Package ‘RPA’

March 12, 2024

Type Package
Title RPA: Robust Probabilistic Averaging for probe-level analysis
Encoding UTF-8
Version 1.58.0
Date 2023-04-07
Depends R (>= 3.1.1), affy, BiocGenerics, BiocStyle, methods, rmarkdown
Imports phyloseq
Suggests knitr, parallel
biocViews GeneExpression, Microarray, Preprocessing, QualityControl
Description Probabilistic analysis of probe reliability and differential gene expression on short oligonucleotide arrays.
License BSD_2_clause + file LICENSE
URL https://github.com/antagomir/RPA
BugReports https://github.com/antagomir/RPA
VignetteBuilder knitr
RoxygenNote 7.2.3
git_url https://git.bioconductor.org/packages/RPA
git_branch RELEASE_3_18
git_last_commit 8684bc9
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-11
Author Leo Lahti [aut, cre] (<https://orcid.org/0000-0001-5537-637X>)
Maintainer Leo Lahti <leo.lahti@iki.fi>
### R topics documented:

- RPA-package .................................................. 3
- calculate.rpa ............................................. 3
- collect.hyperparameters ................................. 4
- d.update.fast ............................................. 5
- estimate.affinities ...................................... 6
- estimate.hyperparameters .............................. 7
- frpa .......................................................... 8
- get.batches ................................................. 10
- get.probe.matrix ......................................... 10
- get.probe.parameters .................................... 12
- get.probeset ................................................ 13
- hyperparameter.update ................................. 14
- levelmap ..................................................... 15
- n.phylotypes.per.oligo .................................. 16
- online.quantile .......................................... 17
- probe.parameters.tolist ............................... 18
- probe.performance ....................................... 18
- probefile ................................................... 19
- probetable .................................................. 20
- retrieve.probesets ...................................... 21
- rpa .......................................................... 21
- rpa.complete ............................................. 23
- rpa.fit ....................................................... 25
- RPA.iteration ............................................. 27
- rpa.online ................................................ 29
- rpa.plot ..................................................... 31
- RPA.preprocess .......................................... 32
- rpa.summarize ........................................... 33
- rpaplot ..................................................... 34
- sample.probeset ......................................... 36
- summarize.batch ........................................ 37
- summarize.batches ...................................... 38
- summarize.rpa ............................................ 39
- summarize.sum ........................................... 40
- summarize_probedata ................................... 41
- updating.hyperparameters ............................. 42

**Index**

- 44
Description

Brief summary of the RPA package

Details

Package: RPA
Type: Package
Version: See sessionInfo() or DESCRIPTION file
Date: 2008-2016
License: FreeBSD
LazyLoad: yes

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#

```
calculate.rpa                  RPA with HITChip
```

Description

Fit RPA for HITChip.

Usage

```
calculate.rpa(level, phylo, oligo.data)
```
Arguments

- **level**: level
- **phylo**: phylo
- **oligo.data**: oligo.data

Value

RPA preprocessed data

Author(s)

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation("microbiome")

---

**Description**

Collect probe-level parameters during online-learning from the batch files.

**Usage**

```r
collect.hyperparameters(
  batches,
  unique.run.identifier,
  save.batches.dir,
  save.batches,
  verbose = TRUE
)
```

**Arguments**

- **batches**: batch list
- **unique.run.identifier**: Batch file identifier string
- **save.batches.dir**: Batch file directory
- **save.batches**: Logical. Determines whether batches are available.
- **verbose**: verbose
d.update.fast

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
# hpe <- collect.hyperparameters(batches, unique.run.identifier, save.batches.dir, save.batches)

d.update.fast  Fast d update

Description
Computes weighted average over the probes, weighted by their inverse probe-specific variances.

Usage
d.update.fast(St, s2)

Arguments
St                    probes x samples data matrix
s2                    variances for the probes

Details
Returns summarized probeset-level weighted average

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
#
Description

Usage
estimate.affinities(dat, a)

Arguments
- dat: Input data set: probes x samples.
- a: Estimated expression signal from RPA model.

Details
To estimate means in the original data domain let us assume that each probe-level observation $x$ is of the following form: $x = d + v + \text{noise}$, where $x$ and $d$ are vectors over samples, $v$ is a scalar (vector with identical elements) noise is Gaussian with zero mean and probe-specific variance parameters $\tau^2$. Then the parameter $\mu$ will indicate how much probe-level observation deviates from the estimated signal shape $d$. This deviation is further decomposed as $\mu = \mu\.real + \mu\.probe$, where $\mu\.real$ describes the 'real' signal level, common for all probes $\mu\.probe$ describes probe affinity effect. Let us now assume that $\mu\.probe \sim N(0, \sigma\.probe)$. This encodes the assumption that in general the affinity effect of each probe tends to be close to zero. Then we just calculate ML estimates of $\mu\.real$ and $\mu\.probe$ based on particular assumptions. Note that this part of the algorithm has not been defined in full probabilistic terms yet, just calculating the point estimates. Note that while $\tau^2$ in RPA measures stochastic noise, and NOT the affinity effect, we use it here as a heuristic solution to weigh the probes according to how much they contribute to the overall signal shape. Intuitively, probes that have little effect on the signal shape (i.e. are very noisy and likely to be contaminated by many unrelated signals) should also contribute less to the absolute signal estimate. If no other prior information is available, using stochastic parameters $\tau^2$ to determine probe weights is likely to work better than simple averaging of the probes without weights. Also in this case the probe affinities sum close to zero but there is some flexibility, and more noisy probes can be downweighted.

Value
A vector with probe-specific affinities.

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")
estimate.hyperparameters

See Also
rpa.fit

Examples

# mu <- estimate.affinities(dat, a)

estimate.hyperparameters

estimate.hyperparameters

Description

Hyperparameter estimation.

Usage

estimate.hyperparameters(
  sets = NULL,
  probe.parameters = list(alpha = 2, beta = 1),
  batches,
  cdf = NULL,
  bg.method = "rma",
  epsilon = 0.01,
  load.batches = FALSE,
  save.hyperparameter.batches = FALSE,
  mc.cores = 1,
  verbose = TRUE,
  normalization.method = "quantiles",
  save.batches.dir = ".",
  unique.run.identifier = NULL,
  set.inds = set.inds
)

Arguments

sets Probesets to handle. All probesets by default.
probe.parameters User-defined priors. May also include quantile.basis
batches Data batches for online learning
cdf CDF probeset definition file
bg.method Background correction method
epsilon Convergence parameter
load.batches Logical. Load preprocessed data whose identifiers are picked from names(batches). Assuming that the same batch list (batches) was used to create the files in online.quantiles function.
save.hyperparameter.batches

Save hyperparameters for each batch into files using the identifiers with batch name with -hyper.RData suffix.

mc.cores

Number of cores for parallel computation

verbose

Print progress information

normalization.method

Normalization method

save.batches.dir

Specify the output directory for temporary batch saves.

unique.run.identifier

Define identifier for this run for naming the temporary batch files. By default, a random id is generated.

set.inds

Probeset indices

Value

alpha: Hyperparameter alpha (same for all probesets); betas: Hyperparameter beta (probe-specific); variances: Probe-specific variances (beta/alpha)

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#

```
fampa
```

Description

Frozen-RPA preprocessing using precalculated probe parameters.

Usage

```
fampa(
  abatch = NULL,
  probe.parameters = NULL,
  verbose = FALSE,
  cdf = NULL,
  cel.files = NULL,
  cel.path = NULL,
)```
mc.cores = 1,
      summarize.with.affinities = FALSE
  )

Arguments

abatch
  An AffyBatch object.

probe.parameters
  A list with tau2 (probe variance), quantile.basis (basis for quantile normalization
  in log2 domain), and optionally affinity (probe affinities). The probe.parameters$tau2
  and probe.parameters$affinity are lists, each element corresponding to a probe-
  set and containing a parameter vector over the probes. The quantile.basis is a
  vector over the probes, the probes need to be listed in the same order as in tau2
  and affinity. probe.parameters can be optionally provided as a data frame.

verbose
  Print progress information during computation.

cdf
  Specify an alternative CDF environment. Default: none.

cel.files
  List of CEL files to preprocess.

cel.path
  Path to CEL file directory.

mc.cores
  Number of cores for parallelized processing.

summarize.with.affinities
  Use affinity estimates in probe summarization step. Default: FALSE.

Details

fRPA function to preprocess Affymetrix CEL files with RPA using precalculated (frozen) probe
parameters.

Value

Preprocessed expression matrix in expressionSet format

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

See Also

rpa, AffyBatch, ExpressionSet

Examples

# eset <- frpa(abatch, probe.parameters)
get.batches

Split data into batches

description

Get probe matrix.

Usage

get.batches(items, batch.size = NULL, shuffle = FALSE)

Arguments

- **items**: A vector of items to be split into batches.
- **batch.size**: Batch size. The last batch may contain less elements than the other batches which have batch.size elements each.
- **shuffle**: Split the elements randomly in the batches.

Value

A list. Each element corresponds to one batch and contains a vector listing the elements in that batch.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#
get.probe.matrix

Usage

get.probe.matrix(
  cels, 
  cdf = NULL, 
  quantile.basis, 
  bg.method = "rma", 
  normalization.method = "quantiles", 
  batch = NULL, 
  verbose = TRUE
)

Arguments

cels List of CEL files to preprocess

cdf Specify an alternative CDF environment

quantile.basis Pre-calculated basis for quantile normalization in log2 domain

bg.method Specify background correction method. See bgcorrect.methods() for options.

normalization.method normalization method

batch batch

verbose Print progress information during computation

Details

Returns background-corrected, quantile normalized log2 probes x samples matrix

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#
get.probe.parameters

Description
Get probe-level hyperparameter from batch files

Usage
get.probe.parameters(
  affinities,
  unique.run.identifier,
  save.batches.dir = ".",
  mode = "list"
)

Arguments
affinities      probe affinities
unique.run.identifier
                Batch file identifier string
save.batches.dir
                Batch file directory
mode            "list" or "table"

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
# df <- get.probe.parameters(unique.run.identifier, save.batches.dir = ".", mode = "list")
get.probeset

Description
Get probeset matrix.

Usage
get.probeset(name, level, taxonomy, probedata, log10 = TRUE)

Arguments
- name: name
- level: taxonomic level
- taxonomy: taxonomy
- probedata: oligos vs. samples preprocessed data matrix; absolute scale
- log10: Logical. Logarithmize the data TRUE/FALSE

Value
probeset data matrix

Author(s)
Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References
See citation('microbiome')

Examples
#taxonomy <- GetPhylogeny('HITChip', 'filtered')
data.dir <- system.file("extdata", package = "microbiome")
#probedata <- read_hitchip(data.dir, "rpa")$probedata
#ps <- get.probeset('Akkermansia', 'L2', taxonomy, probedata)
Description

Update hyperparameters Update shape (alpha) and scale (beta) parameters of the inverse gamma distribution.

Usage

hyperparameter.update(dat, alpha, beta, th = 0.01)

Arguments

dat A probes x samples matrix (probeset).
alpha Shape parameter of inverse gamma density for the probe variances.
beta Scale parameter of inverse gamma density for the probe variances.

th Convergence threshold.

Details

Shape update: alpha <- alpha + T/2; Scale update: beta <- alpha * s2 where s2 is the updated variance for each probe (the mode of variances is given by beta/alpha). The variances (s2) are updated by EM type algorithm, see s2.update.

Value

A list with elements alpha, beta (corresponding to the shape and scale parameters of inverse gamma distribution, respectively).

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

See Also

s2.update, rpa.online
Examples

```
# Generate and fit toydata, learn hyperparameters
#set.seed(11122)
#P <- 11   # number of probes
#N <- 5000 # number of arrays
#real <- sample.probeset(P = P, n = N, shape = 3, scale = 1, mu.real = 4)
#dat <- real$dat # probes x samples#
#
## Set priors
#alpha <- 1e-2
#beta <- rep(1e-2, P)
## Operate in batches
#step <- 1000
#for (ni in seq(1, N, step)) {
#    batch <- ni:(ni+step-1)
#    hp <- hyperparameter.update(dat[,batch], alpha, beta, th = 1e-2)
#    alpha <- hp$alpha
#    beta <- hp$beta
#}
## Final variance estimate
#s2 <- beta/alpha
#
## Compare real and estimated variances
#plot(sqrt(real$tau2), sqrt(s2), main = cor(sqrt(real$tau2), sqrt(s2))); abline(0,1)
```

---

taxonomy

Description

Map taxa between hierarchy levels.

Usage

```
levelmap(taxa = NULL, from, to, tax.table)
```

Arguments

- **taxa**: taxa to convert; if NULL then considering all taxa in the tax.table
- **from**: convert from taxonomic level
- **to**: convert to taxonomic level
- **tax.table**: tax.table

Value

- **mappings**
Author(s)

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation('microbiome')

Description

Check number of matching phylotypes for each probe

Usage

n.phylotypes.per.oligo(taxonomy, level)

Arguments

taxonomy oligo - phylotype matching data.frame
level phylotype level

Value

number of matching phylotypes for each probe

Author(s)

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation("microbiome")
online.quantile

online.quantile Quantile normalization tools for online preprocessing. Estimate quantiles for quantile normalization based on subset of the data (random, or specified by the user).

Usage

online.quantile(abatch, n)

Arguments

abatch AffyBatch
n Numeric: number of random samples to use to define quantile basis. Vector: specify samples to be used in quantile basis calculation.

Details

"online.quantile": Ordinary quantile normalization is exhaustively memory-consuming in large data sets. Then the quantiles can be calculated based on subset of the data to allow efficient normalization. This function can also be used to investigate effect of subset size to convergence of the quantile estimates;"qnorm.basis.online": sweeps through the data in batches to calculate the basis for quantile normalization (average over sorted profiles).

Value

"online.quantile": AffyBatch; "qnorm.basis.online": a vector containing the basis for quantile normalization.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#
probe.parameters.tolist

Description
Convert probe parameter table into a list format

Usage
probe.parameters.tolist(probe.parameters)

Arguments
probe.parameters
A data.frame with alpha, betas, tau2, affinities, quantile.basis

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
# df <- probe.parameters.tolist(probe.parameters.table)

probe.performance

Description
Provide a table of probe-level parameter estimates (affinity and stochastic noise) for RPA output.

Usage
probe.performance(probe.parameters, abatch, sets = NULL)

Arguments
probe.parameters
List with affinities and variances for the probesets
abatch
Affybatch used in the analysis
sets
Specify the probesets to include in the output. Default: All probesets
Value

Data frame of probe-level parameter estimates

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

probeplot plot RPA results and probe-level data for a specified probeset.

Description

probeplot Plot RPA results and probe-level data for a specified probeset.

Usage

probeplot(
  dat,
  highlight.probes = NULL,
  pcol = "darkgrey",
  hcol = "red",
  cex.lab = 1.5,
  cex.axis = 1,
  cex.main = 1,
  cex.names = 1,
  main = "",
  ...
)

Arguments

dat Background-corrected and normalized data: probes x samples.
highlight.probes Optionally highlight some of the probes (with dashed line)
pcol Color for probe signal visualization.
hcol Color for probe highlight
cex.lab Label size adjustment parameters.
cex.axis Axis size adjustment parameters.
cex.main Title size adjustment parameters.
cex.names Names size adjustment parameters.
main Title text.
... Other parameters to pass for plot function.
Details

Plots the preprocessed probe-level observations, estimated probeset-level signal, and probe-specific variances. It is also possible to highlight individual probes and external summary measures.

Value

Used for its side-effects. Returns probes x samples matrix of probe-level data plotted on the image.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

# df <- probetable(probe.parameters)
**retrieve.probesets**

**Retrieve probesets**

**Description**

List probes for each probeset in taxonomic data.

**Usage**

```r
retrieve.probesets(tax.table, level = "species", name = NULL)
```

**Arguments**

- `tax.table`: data.frame with oligo - phylotype mapping info
- `level`: phylotype level for probesets
- `name`: specify phylotypes to check (optional)

**Value**

A list. Probes for each phylotype.

**Author(s)**

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

**References**

See citation('microbiome')

**Examples**

```r
#tax.table <- GetPhylogeny('HITChip')
#sets <- retrieve.probesets(tax.table, 'species', 'Weissella confusa')
```

---

**rpa**

**Description**

Wrapper for RPA preprocessing.
Usage

rpa(
    abatch = NULL,
    verbose = FALSE,
    bg.method = "rma",
    normalization.method = "quantiles.robust",
    cdf = NULL,
    cel.files = NULL,
    cel.path = NULL,
    probe.parameters = NULL,
    mc.cores = 1,
    summarize.with.affinities = FALSE
)

Arguments

abatch An AffyBatch object.
verbose Print progress information during computation.
bg.method Specify background correction method. Default: "rma". See bgcorrect.methods() for other options.
normalization.method Specify quantile normalization method. Default: "pmonly". See normalize.methods(Dilution) for other options.
cdf Specify an alternative CDF environment. Default: none.
cel.files List of CEL files to preprocess.
cel.path Path to CEL file directory.
probe.parameters A list, each element corresponding to a probe set. Each probeset element has the following optional elements: mu (affinity), tau2 (variance), alpha (shape prior), beta (scale prior). Each of these elements contains a vector over the probeset probes, specifying the probe parameters according to the RPA model. If variance is given, it overrides the priors. Can be also used to set user-specified priors for the model parameters. Not used tau2.method = "var". The prior parameters alpha and beta are prior parameters for inverse Gamma distribution of probe-specific variances. Noninformative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta specify an identical inverse Gamma prior for all probes, which regularizes the solution. Can be also specified as lists, each element corresponding to one probeset. May also include quantile.basis
mc.cores Number of cores for parallelized processing.
summarize.with.affinities Use affinity estimates in probe summarization step. Default: FALSE.

Details

RPA preprocessing function. Gives an estimate of the probeset-level mean parameter \( \theta \) of the RPA model, and returns these in an expressionSet object. The choices tau2.method = "robust"
and d.method = "fast" are recommended. With small sample size and informative prior, d.method = "basic" may be preferable. For very large expression data collections, see rpa.online function.

Value

Preprocessed expression matrix in expressionSet format

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

See Also

rpa.online, AffyBatch, ExpressionSet, estimate.affinities, rpa.fit

Examples

# eset <- rpa(abatch)

rpa.complete

Complete RPA preprocessing

Description

RPA preprocessing, also returns probe parameters.

Usage

rpa.complete(
    abatch = NULL,
    sets = NULL,
    epsilon = 0.01,
    tau2.method = "robust",
    d.method = "fast",
    verbose = FALSE,
    bg.method = "rma",
    normalization.method = "quantiles.robust",
    cdf = NULL,
    cel.files = NULL,
    cel.path = NULL,
    probe.parameters = list(),
    mc.cores = 1,
    summarize.with.affinities = FALSE
)
Arguments

abatch
An AffyBatch object.

sets
Probesets for which RPA will be computed.

epsilon
Convergence tolerance. The iteration is deemed converged when the change in all parameters is < epsilon.

tau2.method
Optimization method for tau2 (probe-specific variances). This parameter is denoted by $\tau^2$ in the vignette and manuscript

- "robust": (default) update tau2 by posterior mean, regularized by informative priors that are identical for all probes (user-specified by setting scalar values for alpha, beta). This regularizes the solution, and avoids overfitting where a single probe obtains infinite reliability. This is a potential problem in the other tau2 update methods with non-informative variance priors. The default values alpha = 2; beta = 1 are used if alpha and beta are not specified.
- "mode": update tau2 with posterior mean
- "mean": update tau2 with posterior mean
- "var": update tau2 with variance around d. Applies the fact that tau2 cost function converges to variance with large sample sizes.

d.method
Method to optimize d.

- "fast": (default) weighted mean over the probes, weighted by probe variances. The solution converges to this with large sample size.
- "basic": optimization scheme to find a mode used in Lahti et al. TCBB/IEEE; relatively slow; this is the preferred method with small sample sizes.

verbose
Print progress information during computation.

bg.method
Specify background correction method. Default: "rma". See bgcorrect.methods() for other options.

normalization.method
Specify quantile normalization method. Default: "pmonly". See normalize.methods(Dilution) for other options.

cdf
Specify an alternative CDF environment. Default: none.

cel.files
List of CEL files to preprocess.

cel.path
Path to CEL file directory.

probe.parameters
A list, each element corresponding to a probe set. Each probeset element has the following optional elements: affinity (affinity), tau2 (variance), alpha (shape prior), betas (scale prior). Each of these elements contains a vector over the probeset probes, specifying the probe parameters according to the RPA model. If variance is given, it overrides the priors. Can be also used to set user-specified priors for the model parameters. Not used tau2.method = "var". The prior parameters alpha and beta are prior parameters for inverse Gamma distribution of probe-specific variances. Noninformative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta specify an identical inverse Gamma prior for all probes, which regularizes the solution. Can be also specified as lists, each element corresponding to one probeset. Can also include quantile.basis
rpa.fit

mc.cores Number of cores for parallelized processing.
summarize.with.affinities
Use affinity estimates in probe summarization step. Default: FALSE.

Details
RPA preprocessing function. Gives an estimate of the probeset-level mean parameter \(d\) of the RPA model, and returns these in an expressionSet object. The choices tau2.method = "robust" and d.method = "fast" are recommended. With small sample size and informative prior, d.method = "basic" may be preferable. For very large expression data collections, see rpa.online function.

Value
List with preprocessed expression matrix, corresponding probe parameters, AffyBatch and CDF

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
# eset <- rpa(abatch)

rpa.fit RPA fit

Description
Fit the RPA model.

Usage
rpa.fit(
  dat,
  epsilon = 0.01,
  alpha = NULL,
  beta = NULL,
  tau2.method = "robust",
  d.method = "fast",
  summarize.with.affinities = FALSE
)
Arguments

dat  Original data: probes x samples.
epsilon  Convergence tolerance. The iteration is deemed converged when the change in all parameters is < epsilon.
alpha  alpha prior for inverse Gamma distribution of probe-specific variances. Non-informative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta are specify equal inverse Gamma prior for all probes to regularize the solution. The defaults depend on the method.
beta  beta prior for inverse Gamma distribution of probe-specific variances. Non-informative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta are specify equal inverse Gamma prior for all probes to regularize the solution. The defaults depend on the method.
tau2.method  Optimization method for tau2 (probe-specific variances);
"robust": (default) update tau2 by posterior mean, regularized by informative priors that are identical for all probes (user-specified by setting scalar values for alpha, beta). This regularizes the solution, and avoids overfitting where a single probe obtains infinite reliability. This is a potential problem in the other tau2 update methods with non-informative variance priors. The default values alpha = 2; beta = 1 are used if alpha and beta are not specified.
"mode": update tau2 with posterior mean
"mean": update tau2 with posterior mean
"var": update tau2 with variance around d. Applies the fact that tau2 cost function converges to variance with large sample sizes.
d.method  Method used to optimize d. Options:
"fast": (default) weighted mean over the probes, weighted by probe variances The solution converges to this with large sample size.
"basic": optimization scheme to find a mode used in Lahti et al. TCBB/IEEE; relatively slow; preferred with small sample size.
summarize.with.affinities  Use affinity estimates in probe summarization step. Default: FALSE.

Details

Fits the RPA model, including estimation of probe-specific affinity parameters. First learns a point estimate for the RPA model in terms of differential expression values w.r.t. reference sample. After this, probe affinities are estimated by comparing original data and differential expression shape, and setting prior assumptions concerning probe affinities.

Value

mu: Fitted signal in original data: mu.real + d; mu.real: Shifting parameter of the reference sample; tau2: Probe-specific stochastic noise; affinity: Probe-specific affinities; data: Probeset data matrix; alpha, beta: prior parameters

Author(s)

Leo Lahti <leo.lahti@iki.fi>
RPA.iteration

References
See citation("RPA")

See Also
rpa, estimate.affinities

Examples

# res <- rpa.fit(dat, epsilon, alpha, beta, tau2.method, d.method, affinity.method)

RPA.iteration

RPA iteration

Description
Estimating model parameters d and tau2.

Usage

RPA.iteration(
  S,
  epsilon = 0.001,
  alpha = NULL,
  beta = NULL,
  tau2.method = "fast",
  d.method = "fast",
  maxloop = 1e+06
)

Arguments

S Matrix of probe-level observations for a single probeset: samples x probes.
epsilon Convergence tolerance. The iteration is deemed converged when the change in all parameters is < epsilon.
alpha alpha prior for inverse Gamma distribution of probe-specific variances. Noninformative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta are specify equal inverse Gamma prior for all probes to regularize the solution. The defaults depend on the method.
beta beta prior for inverse Gamma distribution of probe-specific variances. Noninformative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta are specify equal inverse Gamma prior for all probes to regularize the solution. The defaults depend on the method.
tau2.method
Optimization method for tau2 (probe-specific variances).
"robust": (default) update tau2 by posterior mean, regularized by informative priors that are identical for all probes (user-specified by setting scalar values for alpha, beta). This regularizes the solution, and avoids overfitting where a single probe obtains infinite reliability. This is a potential problem in the other tau2 update methods with non-informative variance priors. The default values alpha = 2; beta = 1 are used if alpha and beta are not specified.
"mode": update tau2 with posterior mean
"mean": update tau2 with posterior mean
"var": update tau2 with variance around d. Applies the fact that tau2 cost function converges to variance with large sample sizes.

d.method
Method to optimize d. "fast": (default) weighted mean over the probes, weighted by probe variances. The solution converges to this with large sample size.
"basic": optimization scheme to find a mode used in Lahti et al. TCBB/IEEE; relatively slow; this is the preferred method with small sample sizes.

maxloop
Maximum number of iterations in the estimation process.

Details
Finds point estimates of the model parameters d (estimated true signal underlying probe-level observations), and tau2 (probe-specific variances). Assuming data set S with P observations of signal d with Gaussian noise that is specific for each observation (specified by a vector tau2 of length P), this method gives a point estimate of d and tau2. Probe-level variance priors alpha, beta can be used with tau2.methods 'robust', 'mode', and 'mean'. The d.method = "fast" is the recommended method for point computing point estimates with large sample size.

Value
A list with the following elements: d: A vector. Estimated 'true' signal underlying the noisy probe-level observations.; tau2: A vector. Estimated variances for each measurement (or probe).

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
#
rpa.online

Description

RPA-online for preprocessing very large expression data sets.

Usage

rpa.online(
  cel.path = NULL,
  cel.files = NULL,
  sets = NULL,
  cdf = NULL,
  bg.method = "rma",
  probe.parameters = list(alpha = 1, beta = 1),
  epsilon = 0.01,
  mc.cores = 1,
  verbose = TRUE,
  shuffle = TRUE,
  batch.size = 100,
  batches = NULL,
  save.batches.dir = ".",
  keep.batch.files = FALSE,
  unique.run.identifier = paste("RPA-run-id-", rnorm(1), sep = ""),
  rseed = 23,
  speedup = TRUE,
  summarize.with.affinities = FALSE
)

Arguments

cel.path Path to CEL file directory
cel.files List of CEL files to preprocess
sets Probesets for which RPA will be computed
cdf Specify an alternative CDF environment
bg.method Specify background correction method. See bgcorrect.methods() for options.
probe.parameters Can be used to set user-specified priors for the model parameters alpha, beta. Not used tau2.method = "var". The prior parameters alpha and beta are prior parameters for inverse Gamma distribution of probe-specific variances. Noninformative prior is obtained with alpha, beta -> 0. Not used with tau2.method 'var'. Scalar alpha and beta specify an identical inverse Gamma prior for all probes, which regularizes the solution. Can be also specified as lists, each element corresponding to one probeset. May also include quantile.basis, which should be provided at log2 domain.
epsilon  Convergence tolerance. The iteration is deemed converged when the change in all parameters is < epsilon.
mc.cores  Number of cores for parallel computation
verbose  Print progress information during computation
shuffle  Form random batches
batch.size  Batch size for online mode (rpa.online); the complete list of CEL files will be preprocessed in batches with this size using Bayesian online-updates for probe-specific parameters.
batches  User-defined CEL file batches
save.batches.dir  Output directory for temporary batch saves.
keep.batch.files  Logical. Keep (TRUE) or remove (FALSE) the batch files after preprocessing.
unique.run.identifier  Define identifier for this run for naming the temporary batch files. By default, a random id is generated.
rseed  Random seed.
speedup  Speed up computations with approximations.
summarize.with.affinities  Use affinity estimates in probe summarization step. Default: FALSE.

Details

rpa.online is used to preprocess very large expression data collections based on a Bayesian hyperparameter update procedure. Returns an expressionSet object preprocessed with RPA. Gives an estimate of the probeset-level mean parameter d of the RPA model, and returns these in an expressionSet object. The CEL files are handled in batches to obtain Bayesian updates for probe-specific hyperpriors; after sweeping through the database in batches the results are combined. The online mode is useful for preprocessing very large expression data sets where ordinary preprocessing algorithms fail, without compromises in modelling stage.

Value
List with two elements: an instance of the 'expressionSet' class and probe parameters. For probe.parameters contents, see the probe.parameters input argument.

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

See Also
rpa, AffyBatch, ExpressionSet
**Examples**

```r
# eset <- rpa.online(cel.file.path)
```

**Description**

Plot RPA results and probe-level data for a specified probeset.

**Usage**

```r
rpa.plot(
  x, 
  set, 
  highlight.probes = NULL, 
  pcol = "darkgrey", 
  mucol = "black", 
  ecol = "red", 
  external.signal = NULL, 
  main = NULL, 
  plots = "all", 
  ...
)
```

**Arguments**

- `x`: Output from `rpa.complete` function
- `set`: probeset
- `highlight.probes`: mark probes for highlight
- `pcol`: probe color
- `mucol`: probeset signal color
- `ecol`: external signal color
- `external.signal`: external signal to be plotted on top
- `main`: title
- `plots`: plot type
- `...`: other arguments to be passed

**Details**

Plots the preprocessed probe-level observations, estimated probeset-level signal, and probe-specific variances. It is also possible to highlight individual probes and external summary measures.
Value
Used for its side-effects. Returns probes x samples matrix of probe-level data plotted on the image.

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
#

RPA.preprocess RPA preprocessing

Description
Preprocess AffyBatch object for RPA.

Usage
RPA.preprocess(
  abatch,
  bg.method = "rma",
  normalization.method = "quantiles.robust",
  cdf = NULL,
  cel.files = NULL,
  cel.path = NULL,
  quantile.basis = NULL
)

Arguments
abatch An AffyBatch object.
bg.method Specify background correction method. See bgcorrect.methods(abatch) for options.
normalization.method Specify normalization method. See normalize.methods(abatch) for options. For memory-efficient online version, use "quantiles.online".
cdf The CDF environment used in the analysis.
cel.files List of CEL files to preprocess.
cel.path Path to CEL file directory.
quantile.basis Optional. Basis for quantile normalization. NOTE: required in original, not log2 scale!
Details
Background correction, quantile normalization and log2-transformation for probe-level raw data in abatch. Then probe-level differential expression is computed between the specified ‘reference’ array (cind) and the other arrays. Probe-specific variance estimates are robust against the choice of reference array.

Value
fcmat: Probes x arrays preprocessed differential expression matrix. cind: Specifies which array in abatch was selected as a reference in calculating probe-level differential expression. cdf: The CDF environment used in the analysis. set.ind: Indices for probes in each probeset, corresponding to the rows of fcmat.

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")

Examples
#

rpa.summarize  rpa.summarize

Description
RPA summarization.

Usage
rpa.summarize(dat, affinities, variances, summarize.with.affinities = FALSE)

Arguments
dat Original data: probes x samples.
affinities Probe affinities
variances Probe variances
summarize.with.affinities
Use affinity estimates in probe summarization step. Default: FALSE.

Details
Summarizes the probes in a probe set according to the RPA model based on the given affinity and variance parameters.
Value

A vector. Probeset-level summary signal.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

See Also

rpa

Examples

# res <- rpa.summarize(dat, affinities, variances, summarize.with.affinities = FALSE)

rpaplot

rpaplot Plot RPA results and probe-level data for a specified probeset.

Description

rpaplot Plot RPA results and probe-level data for a specified probeset.

Usage

rpaplot(
  dat,
  mu = NULL,
  tau2 = NULL,
  affinity = NULL,
  highlight.probes = NULL,
  pcol = "darkgrey",
  mucol = "black",
  ecol = "red",
  cex.lab = 1.5,
  cex.axis = 1,
  cex.main = 1,
  cex.names = 1,
  external.signal = NULL,
  main = "",
  plots = "all",
  ...
)
Arguments

- dat: Background-corrected and normalized data: probes x samples.
- mu: probe set signal
- tau2: probe variances
- affinity: probe affinities
- highlight.probes: Optionally highlight some of the probes (with dashed line)
- pcol: Color for probe signal visualization.
- mucol: Color for summary estimate.
- ecol: Color for external signal.
- cex.lab: Label size adjustment parameters.
- cex.axis: Axis size adjustment parameters.
- cex.main: Title size adjustment parameters.
- cex.names: Names size adjustment parameters.
- external.signal: Plot external signal on the probeset. For instance, an alternative summary estimate from another preprocessing method.
- main: Title text.
- plots: "all": plot data and summary, noise and affinity; "data": plot data and summary
- ...: Other parameters to pass for plot function.

Details

Plots the preprocessed probe-level observations, estimated probeset-level signal, and probe-specific variances. It is also possible to highlight individual probes and external summary measures.

Value

Used for its side-effects. Returns probes x samples matrix of probe-level data plotted on the image.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")

Examples

#
sample.probeset

Description
Toydata generator for probeset data.

Usage
sample.probeset(P = 10, n = 20, shape = 1, scale = 1, mu.real = 2)

Arguments
- **P**: Number of probes.
- **n**: Number of samples.
- **shape**: Shape parameter of the inverse Gamma function used to generate the probe-specific variances.
- **scale**: Scale parameters of the inverse Gamma function used to generate the probe-specific variances.
- **mu.real**: Absolute signal level of the probeset.

Details
Generate random probeset with varying probe-specific affinities and variances. The toy data generator follows distributional assumptions of the RPA model and allows quantitative estimation of model accuracy with different options, noise levels and sample sizes. Probeset-level summary estimate is obtained as mu.real + d.

Value
A list with the following elements:
- **dat**: Probeset data: probes x samples
- **tau2**: Probe variances.
- **affinity**: Probe affinities.
- **d**: Probeset signal shape.
- **mu.real**: Probeset signal level.
- **mu**: Probeset-level total signal.

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation("RPA")
Examples

```r
# real <- sample.probeset(P = 10, n = 20, shape = 1, scale = 1, mu.real = 2)
```

Description

Summarize batch.

Usage

```r
summarize.batch(
  q,
  set.inds,
  probe.parameters = list(),
  epsilon,
  verbose = FALSE,
  mc.cores = 1,
  summarize.with.affinities = FALSE
)
```

Arguments

- `q`: Background corrected, quantile-normalized, log2 probes x samples matrix
- `set.inds`: Indices for each probeset, corresponding to `q` matrix
- `probe.parameters`: A list, each element corresponding to a probe set. Each probeset element has the following elements: affinity, variance and optionally alpha and beta priors. Each of these elements contains a vector over the probeset probes, specifying the probe parameters according to the RPA model. If variances are given, that overrides the priors.
- `epsilon`: Convergence tolerance. The iteration is deemed converged when the change in all parameters is < `epsilon`.
- `verbose`: Print progress information during computation.
- `mc.cores`: Number of cores for parallel processing
- `summarize.with.affinities`: Use affinity estimates in probe summarization step. Default: FALSE.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("RPA")
summarize.batches

Examples
#

summarize.batches

Description
Summarize batches.

Usage
def summarize.batches(
    sets = NULL,
    probe.parameters = list(),
    batches,
    load.batches = FALSE,
    mc.cores = 1,
    cdf = NULL,
    bg.method = "rma",
    normalization.method = "quantiles",
    verbose = TRUE,
    save.batches.dir = ".",
    unique.run.identifier = NULL,
    save.batches = FALSE,
    set.inds,
    speedup = FALSE,
    summarize.with.affinities = FALSE
)

Arguments

- `sets` Probesets to summarize
- `probe.parameters` Optional probe parameters, including priors.
- `batches` Data batches for online learning
- `load.batches` Logical. Load precalculated data for the batches.
- `mc.cores` Number of cores for parallel computation
- `cdf` CDF for alternative probeset definitions
- `bg.method` Background correction method
- `normalization.method` Normalization method
- `verbose` Print progress information
- `save.batches.dir` Specify the output directory for temporary batch saves.
Define identifier for this run for naming the temporary batch files. By default, a
random id is generated.

Save batches?

Probeset indices

Speed up calculations with approximations.

Use affinity estimates in probe summarization step. Default: FALSE.

Sweeps through the batches. Summarizes the probesets within each batch based on the precalculated model parameter point estimates.

Expression matrix: probesets x samples.

Leo Lahti <leo.lahti@iki.fi>

See citation("RPA")

#

RPA summarization

Probeset summarization with RPA for taxonomic data.

summarize.rpa(
  taxonomy,
  level,
  probedata,
  verbose = TRUE,
  probe.parameters = NULL
)
Arguments

- **taxonomy**: oligo - phylotype matching data.frame
- **level**: taxonomic level for the summarization.
- **probedata**: preprocessed probes x samples data matrix in absolute domain
- **verbose**: print intermediate messages
- **probe.parameters**: Optional. If probe.parameters are given, the summarization is based on these and model parameters are not estimated. A list. One element for each probeset with the following probe vectors: affinities, variances

Value

List with two elements: abundance.table (summarized data matrix in absolute scale) and probe.parameters (RPA probe level parameter estimates)

Author(s)

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation("microbiome")

---

**summarize.sum**

*Sum-based probe summarization*

**Description**

Probeset summarization with the standard sum method.

**Usage**

```r
summarize.sum(
  taxonomy,
  level,
  probedata,
  verbose = TRUE,
  downweight.ambiguous.probes = TRUE
)
```

**Arguments**

- **taxonomy**: oligo - phylotype matching data.frame
- **level**: taxonomic level for the summarization.
- **probedata**: preprocessed probes x samples data matrix in absolute domain
- **verbose**: print intermediate messages
- **downweight.ambiguous.probes**: Downweight probes with multiple targets
**Value**

List with two elements: abundance.table (summarized data matrix in absolute scale) and probe.parameters used in the calculations

**Author(s)**

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

**References**

See citation("microbiome")

---

**summarize_probedata**  *Summarize probedata*

**Description**

Summarize phylogenetic microarray probe-level data from given input folder.

**Usage**

```r
summarize_probedata(
  data.dir = NULL,
  probedata = NULL,
  taxonomy = NULL,
  level,
  method,
  probe.parameters = NULL
)
```

**Arguments**

- `data.dir`  Data folder.
- `probedata`  probe-level data matrix in absolute domain
- `taxonomy`  probe taxonomy
- `level`  Summarization level
- `method`  Summarization method

**Value**

data matrix (taxa x samples)

**Author(s)**

Contact: Leo Lahti <microbiome-admin@googlegroups.com>
References

See citation('microbiome')

Examples

```r
## Not run:
#library(microbiome)
data.directory <- system.file("extdata", package = "microbiome")
# Read oligo-level data (here: simulated example data)
#probedata <- read_hitchip(data.directory, method = "frpa")$probedata
# Read phylogeny map
# NOTE: use phylogeny.filtered for species/L1/L2 summarization
# Load taxonomy from output directory
#taxonomy <- GetPhylogeny("HITChip", "filtered")
# Summarize oligos into higher level phylotypes
#dat <- summarize_probedata(
#  probedata = probedata,
#  taxonomy = taxonomy,
#  method = "rpa",
#  level = "species")
#
## End(Not run)
```

---

**updating.hyperparameters**

`updating hyperparameters`

**Description**

Hyperparameter update.

**Usage**

`updating.hyperparameters(`

```r
q,
set.inds,
verbose,
mc.cores = 1,
alpha,
betas,
epsilon
```

**Arguments**

- `q` probes x samples matrix
- `set.inds` Probe set indices
- `verbose` Print progress information
**updating.hyperparameters**

- **mc.cores**: Number of cores for parallel computation
- **alpha**: alpha hyperparameter
- **betas**: beta hyperparameters
- **epsilon**: Convergence parameter

**Value**

List with the following elements: alpha, betas, s2s (variances)

**Author(s)**

Leo Lahti <leo.lahti@iki.fi>

**References**

See citation("RPA")

**Examples**

#
Index

* **internal**
  levelmap, 15
  retrieve.probesets, 21

* **methods**
  d.update.fast, 5
  frpa, 8
  get.probe.matrix, 10
  probeplot, 19
  rpa, 21
  rpa.complete, 23
  rpa.online, 29
  rpa.plot, 31
  RPA.preprocess, 32
  rpaplot, 34
  summarize.batch, 37

* **package**
  RPA-package, 3

* **utilities**
  calculate.rpa, 3
  collect.hyperparameters, 4
  estimate.affinities, 6
  estimate.hyperparameters, 7
  get.batches, 10
  get.probe.parameters, 12
  get.probeset, 13
  hyperparameter.update, 14
  n.phylotypes.per.oligo, 16
  online.quantile, 17
  probe.parameters.tolist, 18
  probe.performance, 18
  probetable, 20
  rpa.fit, 25
  RPA.iteration, 27
  rpa.summarize, 33
  sample.probeset, 36
  summarize.batches, 38
  summarize.rpa, 39
  summarize.sum, 40
  summarize_probedata, 41
  updating.hyperparameters, 42
  calculate.rpa, 3
  collect.hyperparameters, 4
  d.update.fast, 5
  estimate.affinities, 6
  estimate.hyperparameters, 7
  frpa, 8
  get.batches, 10
  get.probe.matrix, 10
  get.probe.parameters, 12
  get.probeset, 13
  hyperparameter.update, 14
  levelmap, 15
  n.phylotypes.per.oligo, 16
  online.quantile, 17
  probe.parameters.tolist, 18
  probe.performance, 18
  probeplot, 19
  probetable, 20
  retrieve.probesets, 21
  RPA (RPA-package), 3
  rpa, 21
  RPA-package, 3
  rpa.complete, 23
  rpa.fit, 25
  RPA.iteration, 27
  rpa.online, 29
  rpa.plot, 31
  RPA.preprocess, 32
  rpa.summarize, 33
  rpaplot, 34