Package 'eCV'

January 19, 2024

Title Enhanced Coefficient of Variation and IDR Extensions for Reproducibility Assessment

Version 0.0.2

Description Reproducibility assessment is essential in extracting reliable scientific insights from high-throughput experiments. While the Irreproducibility Discovery Rate (IDR) method has been instrumental in assessing reproducibility, its standard implementation is constrained to handling only two replicates. Package 'eCV' introduces an enhanced Coefficient of Variation (eCV) metric to assess the likelihood of omic features being reproducible. Additionally, it offers alternatives to the Irreproducible Discovery Rate (IDR) calculations for multi-replicate experiments. These tools are valuable for analyzing high-throughput data in genomics and other omics fields. The methods implemented in 'eCV' are described in Gonzalez-Reymundez et al., (2023) <doi:10.1101/2023.12.18.572208>.

URL https://github.com/eclipsebio/eCV

BugReports https://github.com/eclipsebio/eCV/issues

License GPL (>= 3)

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.2.1

Depends R (>= 3.5.0), idr (>= 1.3), mvtnorm (>= 1.1.3), future (>= 1.4.0), future.apply (>= 1.9.0)

Imports stats, utils,

Suggests knitr, rmarkdown, testthat (>= 3.0.0), tidyverse

Language en-US

Config/testthat/edition 3

NeedsCompilation no

Author Agustin Gonzalez-Reymundez [aut, cre]

Maintainer Agustin Gonzalez-Reymundez <agustin.gonrey@eclipsebio.com>

Repository CRAN

Date/Publication 2024-01-19 20:40:02 UTC

R topics documented:

---

eCV *Assess Reproducibility via Enhanced Coefficient of Variation*

Description

This function estimates an "enhanced" coefficient of variation (eCV) to measure the likelihood of an omic feature being reproducible. The eCV is calculated as |SD^2 - Mean^2| / Mean^2, a metric that decreases with noise among replicates and increases with the mean intensity.

Usage

```
eCV(x, max.ite = 10000, n_threads = 1)
```

Arguments

x	A numeric matrix with rows representing omic features and columns representing sample replicates. Numeric values should be positive and reflect significance (not necessarily p-values).
max.ite	Number of samples from the null distribution (numeric). Defaults to 1e4.
n_threads	Number of threads for parallel computing. Numeric. Defaults to 1.

Details

Inferences are made based on the probabilities of eCV values originating from the group of reproducible features. It assumes that reproducible features follow a prior Normal distribution with dimension r (number of replicates). Pseudo replicates are sampled using a Probabilistic Bootstrap, assuming that the global mean vector and variance-covariance matrix across features are close to the prior's hyperparameters. eCV is computed for each random sample. The proportion of times the observed eCV is lower than or equal to the eCV from random samples is then taken as the probability of the omic feature belonging to the group of reproducible features.

Value

Returns a list with two elements:

- **ecv:** Numeric vector with the estimated eCV values for each omic feature.
- **post_prob:** Posterior probability values.

Examples

```
library(eCV)
set.seed(42)
# Simulate data.
out <- simulate_data(scenario = 1,n_features=1e3)

# Run eCV
ecv_out <- eCV(x = out$sim_data, max.ite = 100)


# Plot results.
library(tidyverse)

out$sim_data %>%
as.data.frame() %>%
mutate(`eCV Prob` = ecv_out$post_prob) %>%
ggplot(aes(x = `Rep 1`, y = `Rep 2`, color = `eCV Prob`)) +
 geom_point(size = 1) +
 scale_color_gradientn(colors=c( "#009CA6", "#D5DADD", "#F4364C"))+
 theme_classic()
```

---

gIDR *Estimate IDR with a General Mixture Model*

Description

This function builds upon ‘idr::est.IDR‘ to extend the Li et al. (2011) copula mixture model to accommodate an arbitrary number of replicates. The term "General" in this context alludes to the assumption of a general multivariate Normal distribution of dimension "n," equal to the number of sample replicates. This assumption essentially allows the pseudo-likelihood approach in Li et al. (2011) to be extended to any number of replicates. This is achieved by modifying the "E" and "M" steps of an expectation maximization algorithm to use a multivariate Normal Distribution instead.

Usage

```
gIDR(x, mu, sigma, rho, p, eps = 0.001, max.ite = 30)
```

Arguments

x	Numeric matrix with rows representing the number of omic features and columns representing the number of sample replicates. The numeric values must be positive and represent significance (not necessarily p-values).
mu	Starting value for the mean of the reproducible component. Numeric.
sigma	Starting value for the standard deviation of the reproducible component.
rho	Starting value for the correlation coefficient of the reproducible component.
p	Starting value for the proportion of the reproducible component.

eps	Stopping criterion. Iterations stop when the increment of the log-likelihood is less than eps times the log-likelihood. Default is 0.001.
max.ite	Maximum number of iterations. Default is 30.

Value

Returns a list of three elements:

idr A numeric vector of the local IDR (Irreproducible Discovery Rate) for each observation (i.e., estimated conditional probability for each observation to belong to the irreproducible component).

IDR A numerical vector of the expected Irreproducible Discovery Rate for observations that are as irreproducible or more irreproducible than the given observations.

est_param Estimated parameters: p, rho, mu, sigma.

References

Q. Li, J. B. Brown, H. Huang, and P. J. Bickel. (2011)

Examples

```
# 1. Show that gIDR reduces to classical IDR for n=2.

# Load required packages.
library(idr)
library(eCV)

# Set seed for RNG.
set.seed(42)

# Simulate data.
out <- simulate_data(scenario = 2, n_features = 1e3)

# Set initial parameter values.
mu <- 2
sigma <- 1.3
rho <- 0.8
p <- 0.7

# Compare IDR and gIDR
idr.out <- est.IDR(x = out$sim_data, mu, sigma, rho, p)
gidr.out <- gIDR(x = out$sim_data, mu, sigma, rho, p)

# Show the results are the same.
all.equal(gidr.out$est_param, idr.out$para)



library(tidyverse)
# Plot results.
out$sim_data %>% as.data.frame() %>%
```

```
mutate(idr = gidr.out$idr) %>%
ggplot(aes(x=`Rep 1`,y=`Rep 2`,color=idr)) +
 geom_point(size=1) +
 scale_color_gradientn(colors=c("#F4364C", "#D5DADD", "#009CA6" ))+
 theme_classic()

#2. Show gIDR for n=10.

out <- simulate_data(scenario = 1, n_reps = 10, n_features = 1e3)
gidr.out <- gIDR(x = out$sim_data, mu, sigma, rho, p)
out$sim_data %>% as.data.frame() %>%
mutate(idr = gidr.out$IDR) %>%
ggplot(aes(x = `Rep 1`, y = `Rep 2`, color = idr)) +
 geom_point(size = 1) +
 scale_color_gradientn(colors=c("#F4364C", "#D5DADD", "#009CA6" ))+
 theme_classic()
```

`mIDR` *Estimate meta Irreproducible Discovery Rate (mIDR).*

Description

This function extends the Li et al. (2011) copula mixture model, originally implemented in idr::est.IDR, to accommodate any number of replicates. It computes the local IDR for all pairwise combinations of replicates. Then it computes a "meta" local IDR score using the formula: 1 - (1 - idr_1)*...*(1 - idr_C(r,2)), where C(r,2) represents the number of all pairwise combinations of scores. Once the meta local IDR is obtained, the expected IDR scores are obtained in the same way as in the traditional IDR procedure.

Usage

```
mIDR(x, mu, sigma, rho, p, eps = 0.001, max.ite = 20, n_threads = 1)
```

Arguments

`x`	A numeric matrix with rows representing the number of omic features and columns representing the number of sample replicates. The numeric values should be positive and represent significance (not necessarily p-values).
`mu`	Starting value for the mean of the reproducible component Numeric.
`sigma`	Starting value for the standard deviation of the reproducible component Numeric.
`rho`	Starting value for the correlation coefficient of the reproducible component Numeric.
`p`	Starting value for the proportion of the reproducible component Numeric.
`eps`	Stopping criterion. Iterations stop when the increment of the log-likelihood is less than "eps" times the log-likelihood. Defaults to 0.001.
`max.ite`	Maximum number of iterations. The default is 30.
`n_threads`	Number of threads for parallel computing. Numeric. Defaults to 1.

Value

Returns a list of two elements:

idr A numeric vector of the local meta IDR for each observation.

IDR A numerical vector of the expected meta IDR for observations that are as irreproducible or more irreproducible than the given observations.

References

Q. Li, J. B. Brown, H. Huang, and P. J. Bickel. (2011) Measuring reproducibility of high-throughput experiments. Annals of Applied Statistics, Vol. 5, No. 3, 1752-1779.

Examples

```
library(eCV)
set.seed(42)

# Simulate data.
out <- simulate_data(scenario = 1, n_reps = 4, n_features = 1e3)

# Set initial parameter values.
mu <- 2
sigma <- 1.3
rho <- 0.8
p <- 0.7

# Get meta local IDR scores.
midr_out <- mIDR(x = out$sim_data, mu, sigma, rho, p)

library(tidyverse)
out$sim_data %>%
  as.data.frame() %>%
  mutate(`Meta idr` = midr_out$idr) %>%
  ggplot(aes(x = `Rep 1`, y = `Rep 2`, color = `Meta idr`)) +
  geom_point(size = 1) +
 scale_color_gradientn(colors=c("#F4364C", "#D5DADD", "#009CA6" ))+
  theme_classic()
```

`mrep_assessment` *Multi-replicate Reproducibility Assessment.*

Description

This function wraps the different methods implemented in the package eCV to assess reproducibility of omic feature values coming from two or more sample replicates. The 'method' argument specifies any of the implemented methods: "IDR", "gIDR", "mIDR", and "eCV".

Usage

```
mrep_assessment(x, method = "eCV", param, n_threads = 1)
```

Arguments

x	A numeric matrix with rows representing the number of omic features and columns representing the number of sample replicates. The numeric values should be positive and represent significance (not necessarily p-values).
method	The name of the method used to assess reproducibility. Character. Possible values are "IDR", "gIDR", "mIDR", and "eCV". Defaults to "eCV".
param	List specifying the initial values for the parameters used by the specified method. If method is any of the IDR variants, param must be a named list with "mu", "sigma", "rho", "p", "eps", and "max.ite". If method = "eCV", param only needs "max.ite".
n_threads	Number of threads for parallel computing. Numeric. Default to 1. Only used when method is mIDR or eCV.

Details

The "IDR" method calls the traditional IDR, as implemented in the package idr (idr::est.IDR). Regardless of the number of replicates given to the function, when method="IDR", only the first two are used. Any of the other methods are meant to be used when r >= 2. Both gIDR and mIDR reduce to traditional IDR if r = 2.

Value

A list with two elements

rep_index A numeric vector with values between zero and one, with smaller values indicating higher reproducibility

method String storing the name of the method used

Examples

```
library(eCV)

# Simulate data
set.seed(42)
out <- simulate_data(scenario = 2, n_reps = 4, n_features = 1e3)

# Define parameters for each method.
params <- list(
  eCV = list(max.ite = 100),
  gIDR = list(
     mu = 2,
     sigma = 1.3,
     rho =0.8,
      p = 0.7,
    eps = 1e-3,
```

```
    max.ite = 50),
  mIDR = list(
     mu = 2,
     sigma = 1.3,
     rho =0.8,
      p = 0.7,
    eps = 1e-3,
    max.ite = 50))

# Create a list to store results
results <- NULL
methods <- c("eCV", "gIDR", "mIDR")
for (method in methods) {
  rep_index <- mrep_assessment(x = out$sim_data,
                               method = method,
                               param = params[[method]])$rep_index
  new_rows <- data.frame(value = rep_index,
                         Method = method,
                         group = out$sim_params$feature_group)
  results <- rbind(results, new_rows)
}

# Plot results

library(tidyverse)
results %>%
mutate(group = ifelse(group == 1,"FALSE","TRUE")) %>%
ggplot(aes(x=Method, y = value,fill=group)) +
scale_fill_manual(values = c( "#009CA6" , "#F4364C")) +
geom_boxplot() +
theme_classic() +
labs(y="Reproducibility assessment", fill="Reproducible\nfeature")
```

`simulate_data` *Simulates omic features into reproducible and irreproducible groups*

Description

This function is an extension of the copula mixture model simulations presented in Li et al. (2011). It generates samples of n_features pairs of omic features for n_reps (>=2) replicates. The state of each omic feature (i.e., reproducible or irreproducible) is determined by sampling from a binomial variable K with a vector of probabilities, P. The vector P represents the mixing probability between two multivariate normal distributions. The elements of P are associated with reproducibility. For example, if K can only assume two values, say 0 or 1, then K can represent groups of reproducible or irreproducible features.

Usage

```
simulate_data(n_reps = 2, n_features = 10000, scenario = 1)
```

Arguments

n_reps	Number of sample replicates. Numeric. Defaults to 2.
n_features	Number of omic features to simulate. Numeric. Defaults to 1e4.
scenario	Combination of parameters' values defining scenarios in Li et al. (2011). Numeric. Possible values are 1, 2, 3, or 4. Defaults to 1.

Details

The dimension of each normal distribution is determined by the number of replicates, r. The "scenario" argument controls the values of the parameters according to the simulation scenarios outlined in Li et al. (2011) (Table I in the article). Scenario 1 corresponds to a situation where reproducible features are highly correlated and exceed the number of irreproducible features. Scenario 2 corresponds to a situation where the reproducible features are less than the irreproducible ones and exhibit low correlation. Scenario 3 represents situations where reproducible features are less than irreproducible ones but still highly correlated. Scenario 4 is a generalization of Scenario 1, with the addition of a component of "reproducible noise" consisting of highly correlated but low-intensity features.

Value

Returns a list of two elements:

- **sim_data:** Matrix of dimensions n_features x n_reps with the simulated numerical values for each feature.
- **sim_params:** List with all the parameter values.

References

Q. Li, J. B. Brown, H. Huang, and P. J. Bickel. (2011) Measuring reproducibility of high-throughput experiments. Annals of Applied Statistics, Vol. 5, No. 3, 1752-1779.

Examples

```
library(eCV)
set.seed(42)
out <- simulate_data(scenario = 1)

library(tidyverse)
out$sim_data %>% as.data.frame() %>%
mutate(`Features group` = as.character(out$sim_params$feature_group)) %>%
ggplot(aes(x=`Rep 1`,y=`Rep 2`,color=`Features group`)) +
 geom_point(size=1, alpha=0.5) +
 scale_color_manual(values = c( "#009CA6" , "#F4364C")) +
 theme_classic()
```

Index