Package 'benchmarkfdrData2019'

June 20, 2024

Title Data and Benchmarking Results from Korthauer and Kimes et al. (2019)

Version 1.19.0

BugReports https://github.com/stephaniehicks/benchmarkfdrData2019/issues

Description Benchmarking results for experimental and simulated data sets used in Korthauer and Kimes et al. (2019) to compare methods for controlling the false discovery rate.

Depends R (>= 3.6.0), SummarizedExperiment, ExperimentHub

Imports utils

Suggests rmarkdown, knitr, BiocStyle, testthat, SummarizedBenchmark, dplyr, ggplot2, rlang

License MIT + file LICENSE

Encoding UTF-8

LazyData true

biocViews SingleCellData, ExperimentData, RNASeqData, ExpressionData, ExperimentData, ExperimentHub

VignetteBuilder knitr

RoxygenNote 6.1.1

git_url https://git.bioconductor.org/packages/benchmarkfdrData2019

git_branch devel

git_last_commit 23fee96

git_last_commit_date 2024-04-30

Repository Bioconductor 3.20

Date/Publication 2024-06-20

Author Stephanie Hicks [aut, cre] (<https://orcid.org/0000-0002-7858-0231>), Keegan Korthauer [aut] (<https://orcid.org/0000-0002-4565-1654>), Patrick Kimes [aut] (<https://orcid.org/0000-0001-6819-9077>)

Maintainer Stephanie Hicks <shicks19@jhu.edu>

Contents

`chipseq-cbp` *ChIP-seq, CBP dataset results*

Description

Results for case study on differential peak calling in CREB-binding protein (CBP) knockout and wild-type mice described in Korthauer and Kimes et al. (2019). Analyses were performed using the `csaw` package.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `cbp-csaw-benchmark`: using csaw with region width as covariate.
- `cbp-csaw-uninf-benchmark`: using csaw with uninformative covariate.
- `cbp-csaw-cov-benchmark`: using csaw with mean coverage as covariate.

See Also

`chipseq-h3k4me3`

Examples

```
`cbp-csaw-benchmark`(metadata = TRUE)

## Not run:
`cbp-csaw-benchmark`()
`cbp-csaw-uninf-benchmark`()
`cbp-csaw-cov-benchmark`()

## End(Not run)
```

---

chipseq-h3k4me3 *ChIP-seq, H3K4me3 dataset results*

Description

Results for case study on differential peak calling in human H3K4me3 ChIP-seq data from ENCODE described in Korthauer and Kimes et al. (2019). Analyses were performed using either `DESeq2` applied to predefined regions or with the `csaw` package.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `h3k4me3-promoters-benchmark`: using predefined regions with mean coverage as covariate.
- `h3k4me3-csaw-benchmark`: using csaw with region width as covariate.
- `h3k4me3-csaw-uninf-benchmark`: using csaw with uninformative covariate.
- `h3k4me3-csaw-cov-benchmark`: using csaw with mean coverage as covariate.

See Also

chipseq-cbp

Examples

```
`h3k4me3-promoters-benchmark`(metadata = TRUE)

## Not run:
`h3k4me3-promoters-benchmark`()
`h3k4me3-csaw-benchmark`()
`h3k4me3-csaw-uninf-benchmark`()
`h3k4me3-csaw-cov-benchmark`()

## End(Not run)
```

---

gsa-human *Gene Set Analysis, human dataset results*

Description

Results for case study on gene set analysis using human dataset described in Korthauer and Kimes et al. (2019). Analyses were performed using GSEA.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- human-benchmark: using gene set size as covariate.
- human-uninf-benchmark: using uninformative covariate.

See Also

gsa-human

Examples

```
`human-benchmark`(metadata = TRUE)

## Not run:
`human-benchmark`()
`human-uninf-benchmark`()

## End(Not run)
```

gsa-mouse *Gene Set Analysis, mouse dataset results*

Description

Results for case study on gene set analysis using mouse dataset described in Korthauer and Kimes et al. (2019). Analyses were performed using GSEA.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- mouse-benchmark: using gene set size as covariate.
- mouse-uninf-benchmark: using uninformative covariate.

See Also

gsa-human

Examples

```
`mouse-benchmark`(metadata = TRUE)

## Not run:
`mouse-benchmark`()
`mouse-uninf-benchmark`()

## End(Not run)
```

gwas-bmi *Genome-wide Association Study, BMI dataset results*

Description

Results for case study on genome-wide associate study meta-analysis using a BMI dataset from the GIANT Consortium described in Korthauer and Kimes et al. (2019).

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `bmi-maf-benchmark`: using minor allele frequency as covariate.
- `bmi-uninf-benchmark`: using uninformative covariate.
- `bmi-samplesize-benchmark`: using sample size as covariate.

Examples

```
`bmi-maf-benchmark`(metadata = TRUE)

## Not run:
`bmi-maf-benchmark`()
`bmi-uninf-benchmark`()
`bmi-samplesize-benchmark`()

## End(Not run)
```

`microbiome-baxter` *Microbiome, Baxter dataset results*

Description

Results for case study on 16S microbiome differential testing of Baxter et al. (2016) dataset described in Korthauer and Kimes et al. (2019). Wilcoxon rank-sum test was used to test for differential abundance between samples from patients with colorectal cancer and patients with no colonic lesions.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `baxter-genus-benchmark`: testing at genus-level using ubiquity as covariate.
- `baxter-genus-log-ubiquity-benchmark`: testing at genus-level using log-ubiquity as covariate.
- `baxter-genus-mean-abun-benchmark`: testing at genus-level using mean abundance as covariate.
- `baxter-genus-mean-uninf-benchmark`: testing at genus-level using uninformative covariate.
- `baxter-otus-benchmark`: testing at OTU-level using ubiquity as covariate.
- `baxter-otus-log-ubiquity-benchmark`: testing at OTU-level using log-ubiquity as covariate.

References

- Baxter, N. T., Ruffin, M. T., Rogers, M. A., & Schloss, P. D. (2016). Microbiota-based model improves the sensitivity of fecal immunochemical test for detecting colonic lesions. Genome Medicine, 8(1), 37.

See Also

microbiome-schubert, microbiome-goodrich, microbiome-papa, microbiome-enigma

Examples

```
`baxter-genus-benchmark`(metadata = TRUE)

## Not run:
`baxter-genus-benchmark`()
`baxter-genus-log-ubiquity-benchmark`()
`baxter-genus-mean-abun-benchmark`()
`baxter-genus-mean-uninf-benchmark`()
`baxter-otus-benchmark`()
`baxter-otus-log-ubiquity-benchmark`()

## End(Not run)
```

---

microbiome-enigma *Microbiome, ENIGMA dataset results*

Description

Results for case study on 16S microbiome Spearman correlation testing of ENIGMA dataset described in Korthauer and Kimes et al. (2019). Spearman correlation test was used to test for correlation between OTU-level microbial abundances and various geochemical and physical measurements.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- enigma-al-otus-benchmark: testing for correlation with Al measurements using ubiquity as covariate.
- enigma-al-otus-abun-benchmark: testing for correlation with Al measurements using mean abundance as covariate.

- enigma-al-otus-uninf-benchmark: testing for correlation with Al measurements using uninformative covariate.
- enigma-ph-otus-benchmark: testing for correlation with pH measurements using ubiquity as covariate.
- enigma-ph-otus-abun-benchmark: testing for correlation with pH measurements using mean abundance as covariate.
- enigma-ph-otus-uninf-benchmark: testing for correlation with pH measurements using uninformative covariate.
- enigma-so4-otus-benchmark: testing for correlation with SO4 measurements using ubiquity as covariate.
- enigma-so4-otus-abun-benchmark: testing for correlation with SO4 measurements using mean abundance as covariate.
- enigma-so4-otus-uninf-benchmark: testing for correlation with SO4 measurements using uninformative covariate.

See Also

microbiome-schubert, microbiome-baxter, microbiome-goodrich, microbiome-papa

Examples

```
`enigma-al-otus-benchmark`(metadata = TRUE)

## Not run:
`enigma-al-otus-benchmark`()
`enigma-al-otus-abun-benchmark`()
`enigma-al-otus-uninf-benchmark`()
`enigma-ph-otus-benchmark`()
`enigma-ph-otus-abun-benchmark`()
`enigma-ph-otus-uninf-benchmark`()
`enigma-so4-otus-benchmark`()
`enigma-so4-otus-abun-benchmark`()
`enigma-so4-otus-uninf-benchmark`()

## End(Not run)
```

---

microbiome-goodrich *Microbiome, Goodrich dataset results*

Description

Results for case study on 16S microbiome differential testing of Goodrich et al. (2014) dataset described in Korthauer and Kimes et al. (2019). Wilcoxon rank-sum test was used to test for differential abundance between samples from lean (body mass index, BMI < 25) and obsese (BMI > 30) individuals.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- goodrich-genus-benchmark: testing at genus-level using ubiquity as covariate.
- goodrich-genus-abun-benchmark: testing at genus-level using mean abundance as covariate.
- goodrich-genus-uninf-benchmark: testing at genus-level using uninformative covariate.
- goodrich-otus-benchmark: testing at OTU-level using ubiquity as covariate.

References

- Goodrich, J. K. et al. (2014). Human genetics shape the gut microbiome. Cell, 159(4), 789-799.

See Also

microbiome-schubert, microbiome-baxter, microbiome-papa, microbiome-enigma

Examples

```
`goodrich-genus-benchmark`(metadata = TRUE)

## Not run:
`goodrich-genus-benchmark`()
`goodrich-genus-abun-benchmark`()
`goodrich-genus-uninf-benchmark`()
`goodrich-otus-benchmark`()

## End(Not run)
```

---

microbiome-papa *Microbiome, Papa dataset results*

Description

Results for case study on 16S microbiome differential testing of Papa et al. (2012) dataset described in Korthauer and Kimes et al. (2019). Wilcoxon rank-sum test was used to test for differential abundance between samples from patients with GI symptoms but not IBD, and IBD patients.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- papa-genus-benchmark: testing at genus-level using ubiquity as covariate.
- papa-genus-abun-benchmark: testing at genus-level using mean abundance as covariate.
- papa-genus-log-ubiquity-benchmark: testing at genus-level using log-ubiquity as covariate.
- papa-genus-uninf-benchmark: testing at genus-level using uninformative covariate.
- papa-otus-benchmark: testing at OTU-level using ubiquity as covariate.

References

- Papa, E. et al. (2012). Non-invasive mapping of the gastrointestinal microbiota identifies children with inflammatory bowel disease. PloS One, 7(6), e39242.

See Also

microbiome-schubert, microbiome-baxter, microbiome-goodrich, microbiome-enigma

Examples

```
`papa-genus-benchmark`(metadata = TRUE)

## Not run:
`papa-genus-benchmark`()
`papa-genus-abun-benchmark`()
`papa-genus-log-ubiquity-benchmark`()
`papa-genus-uninf-benchmark`()
`papa-otus-benchmark`()

## End(Not run)
```

---

microbiome-schubert *Microbiome, Schubert dataset results*

Description

Results for case study on 16S microbiome differential testing of Schubert et al. (2014) dataset described in Korthauer and Kimes et al. (2019). Wilcoxon rank-sum test was used to test for differential abundance between samples from healthy patients and patients with diarrheal stool samples (including both C. difficile infection, CDI, and non-CDI samples).

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `schubert-otus-benchmark`: testing at OTU-level using ubiquity as covariate.
- `schubert-otus-abun-benchmark`: testing at OTU-level using mean abundance as covariate.
- `schubert-otus-uninf-benchmark`: testing at OTU-level using uninformative covariate.

References

- Schubert, A. M. et al. (2014). Microbiome data distinguish patients with Clostridium difficile infection and non-C. difficile-associated diarrhea from healthy controls. MBio, 5(3), e01021-14.

See Also

`microbiome-baxter`, `microbiome-goodrich`, `microbiome-papa`, `microbiome-enigma`

Examples

```
`schubert-otus-benchmark`(metadata = TRUE)

## Not run:
`schubert-otus-benchmark`()
`schubert-otus-abun-benchmark`()
`schubert-otus-uninf-benchmark`()

## End(Not run)
```

`polyester-insilico` *Polyester simulated dataset results*

Description

Results for differential gene expression analysis using RNA-seq data simulated with the **Polyester** package from yeast data obtained from Schurch et al. (2016) as described in Korthauer and Kimes et al. (2019). Differential expression was tested using **DESeq2**.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `polyester-results-null5`: null 5 vs. 5 comparison.
- `polyester-results-de5`: using unimodal alternative, 2000 DE genes, strong informative covariate, 5 vs. 5 comparison.
- `polyester-results-de5-uninfCov`: using unimodal alternative, 2000 DE genes, uninformative covariate, 5 vs. 5 comparison.
- `polyesterW-results-de5`: using unimodal alternative, 2000 DE genes, weak informative covariate, 5 vs. 5 comparison.
- `polyester-results-null10`: null 10 vs. 10 comparison.
- `polyester-results-de10`: using unimodal alternative, 2000 DE genes, strong informative covariate, 10 vs. 10 comparison.
- `polyester-results-de10-uninfCov`: using unimodal alternative, 2000 DE genes, uninformative covariate, 10 vs. 10 comparison.
- `polyesterW-results-de10`: using unimodal alternative, 2000 DE genes, weak informative covariate, 10 vs. 10 comparison.

References

- Schurch, N. J. et al. (2016). How many biological replicates are needed in an RNA-seq experiment and which differential expression tool should you use?. RNA, 22(6), 839-851.

See Also

yeast-insilico

Examples

```
`polyester-results-de5`(metadata = TRUE)

## Not run:
`polyester-results-de5`()
`polyester-results-de5-uninfCov`()
`polyester-results-de10`()
`polyester-results-de10-uninfCov`()
`polyester-results-null5`()
`polyester-results-null10`()
`polyesterW-results-de5`()
`polyesterW-results-de10`()

## End(Not run)
```

rnaseq-brain *RNA-seq, brain dataset results*

Description

Results for case study on differential gene expression analysis between two brain regions (nucleus accumbens and putamen) using data from the GTEx project described in Korthauer and Kimes et al. (2019). Differential expression was tested using **DESeq2**.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- brain-benchmark: using mean counts as covariate.
- brain-uninf-benchmark: using uninformative covariate.

See Also

rnaseq-mir200c

Examples

```
`brain-benchmark`(metadata = TRUE)

## Not run:
`brain-benchmark`()
`brain-uninf-benchmark`()

## End(Not run)
```

rnaseq-mir200c *RNA-seq, mir200c dataset results*

Description

Results for case study on differential gene expression analysis between mir200c knockdown cells and untreated cells described in Korthauer and Kimes et al. (2019). Differential expression was tested using **DESeq2**.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `mir200c-benchmark`: using mean counts as covariate.
- `mir200c-uninf-benchmark`: using uninformative covariate.

See Also

`rnaseq-brain`

Examples

```
`mir200c-benchmark`(metadata = TRUE)

## Not run:
`mir200c-benchmark`()
`mir200c-uninf-benchmark`()

## End(Not run)
```

`scrnaseq-human` *scRNA-seq, human dataset results*

Description

Results for case study on single-cell RNA-seq differential expression analysis using data from diffuse neoplastic infiltrating cells at the migrating front of human glioblastoma as described in Korthauer and Kimes et al. (2019). Differential expression was testing using either the **MAST** package, **scDD** package, or Wilcoxon rank-sum test.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `human-benchmark-mast-det`: testing with **MAST**, using detection rate as covariate.
- `human-benchmark-mast-mean`: testing with **MAST**, using mean counts as covariate.
- `human-benchmark-mast-uninf`: testing with **MAST**, using uninformative covariate.

- human-benchmark-scdd-det: testing with **scDD**, using detection rate as covariate.
- human-benchmark-scdd-mean: testing with **scDD**, using mean counts as covariate.
- human-benchmark-scdd-uninf: testing with **scDD**, using uninformative covariate.
- human-benchmark-wilcox-det: testing with Wilcoxon rank-sum, using detection rate as covariate.
- human-benchmark-wilcox-mean: testing with Wilcoxon rank-sum, using mean counts as covariate.
- human-benchmark-wilcox-uninf: testing with Wilcoxon rank-sum, using uninformative covariate.

See Also

scrnaseq-mouse

Examples

```
`human-benchmark-mast-det`(metadata = TRUE)

## Not run:
`human-benchmark-mast-det`()
`human-benchmark-mast-mean`()
`human-benchmark-mast-uninf`()
`human-benchmark-scdd-det`()
`human-benchmark-scdd-mean`()
`human-benchmark-scdd-uninf`()
`human-benchmark-wilcox-det`()
`human-benchmark-wilcox-mean`()
`human-benchmark-wilcox-uninf`()

## End(Not run)
```

---

scrnaseq-mouse *scRNA-seq, mouse dataset results*

Description

Results for case study on single-cell RNA-seq differential expression analysis using data from mouse cells at NF-kappa-beta activation as described in Korthauer and Kimes et al. (2019). Differential expression was testing using either the **MAST** package, **scDD** package, or Wilcoxon rank-sum test.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `mouse-benchmark-mast-det`: testing with **MAST**, using detection rate as covariate.
- `mouse-benchmark-mast-mean`: testing with **MAST**, using mean counts as covariate.
- `mouse-benchmark-mast-uninf`: testing with **MAST**, using uninformative covariate.
- `mouse-benchmark-scdd-det`: testing with **scDD**, using detection rate as covariate.
- `mouse-benchmark-scdd-mean`: testing with **scDD**, using mean counts as covariate.
- `mouse-benchmark-scdd-uninf`: testing with **scDD**, using uninformative covariate.
- `mouse-benchmark-wilcox-det`: testing with Wilcoxon rank-sum, using detection rate as covariate.
- `mouse-benchmark-wilcox-mean`: testing with Wilcoxon rank-sum, using mean counts as covariate.
- `mouse-benchmark-wilcox-uninf`: testing with Wilcoxon rank-sum, using uninformative covariate.

See Also

`scrnaseq-human`

Examples

```
`mouse-benchmark-mast-det`(metadata = TRUE)

## Not run:
`mouse-benchmark-mast-det`()
`mouse-benchmark-mast-mean`()
`mouse-benchmark-mast-uninf`()
`mouse-benchmark-scdd-det`()
`mouse-benchmark-scdd-mean`()
`mouse-benchmark-scdd-uninf`()
`mouse-benchmark-wilcox-det`()
`mouse-benchmark-wilcox-mean`()
`mouse-benchmark-wilcox-uninf`()

## End(Not run)
```

---

sims-informative-cosine

Simulation study results (cosine covariate)

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'cosine' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with standard Gaussian, t_5, t_{11}, and χ_4 distributed test statistics.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as "informative" and "uninformative", respectively.

Examples

```
`informative-cosine-benchmark-gaussian`(metadata = TRUE)

## Not run:
`informative-cosine-benchmark-gaussian`()
`informative-cosine-benchmark-t11`()
`informative-cosine-benchmark-t5`()
`informative-cosine-benchmark-chisq4`()

res <- `informative-cosine-benchmark-chisq4`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

---

sims-informative-cubic

Simulation study results (cubic covariate)

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'cubic' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with standard Gaussian, t_5, t_{11}, and χ_4 distributed test statistics.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as "informative" and "uninformative", respectively.

Examples

```
`informative-cubic-benchmark-gaussian`(metadata = TRUE)

## Not run:
`informative-cubic-benchmark-gaussian`()
`informative-cubic-benchmark-t11`()
`informative-cubic-benchmark-t5`()
`informative-cubic-benchmark-chisq4`()

res <- `informative-cubic-benchmark-chisq4`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

---

sims-informative-sine *Simulation study results (sine covariate)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'sine' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with standard Gaussian, t_5, t_{11}, and χ_4 distributed test statistics.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as "informative" and "uninformative", respectively.

Examples

```
`informative-sine-benchmark-gaussian`(metadata = TRUE)

## Not run:
`informative-sine-benchmark-gaussian`()
`informative-sine-benchmark-t11`()
`informative-sine-benchmark-t5`()
`informative-sine-benchmark-chisq4`()

res <- `informative-sine-benchmark-chisq4`()
res[[1]]$informative
res[[1]]$uninformative
```

```
## End(Not run)
```

`sims-informative-step` *Simulation study results (step covariate)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'step' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with standard Gaussian, t_5, t_{11}, and χ_4 distributed test statistics.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

Examples

```
`informative-step-benchmark-gaussian`(metadata = TRUE)

## Not run:
`informative-step-benchmark-gaussian`()
`informative-step-benchmark-t11`()
`informative-step-benchmark-t5`()
`informative-step-benchmark-chisq4`()

res <- `informative-step-benchmark-chisq4`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

sims-null *Simulation study results (null settings)*

Description

Results for null simulations used in Korthauer and Kimes et al. (2019) stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with standard Gaussian, t_5, t_{11}, and χ_4 distributed test statistics.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as "informative" and "uninformative", respectively.

Examples

```
`null-benchmark-gaussian`(metadata = TRUE)

## Not run:
`null-benchmark-gaussian`()
`null-benchmark-t11`()
`null-benchmark-t5`()
`null-benchmark-chisq4`()

res <- `null-benchmark-chisq4`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

sims-uasettings *Simulation study results (unimodal, Gaussian-distributed)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'unimodal' series of effect size distributions (Stephens, 2017) with standard Gaussian distributed test statistics, and 'cubic' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. In each simulation, 10% of tests were non-null, i.e. had non-zero effect sizes.

Simulations were performed with 'bimodal', 'flat top', 'skew' and 'spiky' effect size distributions (see corresponding reference for details).

In sims-uasettings-t and sims-uasettings-25, similar settings were considered, but with t_{11} distributed test statistics and 25% of tests being non-null, respectively.

Arguments

metadata	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as "informative" and "uninformative", respectively.

References

- Stephens, M. (2017). False discovery rates: a new deal. Biostatistics, 18(2), 275-294.

See Also

sims-uasettings-t, sims-uasettings-25

Examples

```
`uasettings-benchmark-bimodal`(metadata = TRUE)

## Not run:
`uasettings-benchmark-bimodal`()
`uasettings-benchmark-flattop`()
`uasettings-benchmark-skew`()
`uasettings-benchmark-spiky`()

res <- `uasettings-benchmark-spiky`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

---

sims-uasettings-25 *Simulation study results (unimodal, 25% non-null)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'unimodal' series of effect size distributions (Stephens, 2017) with standard Gaussian distributed test statistics, and 'cubic' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. In each simulation, 25% of tests were non-null, i.e. had non-zero effect sizes. Simulations were performed with 'bimodal', 'flat top', 'skew' and 'spiky' effect size distributions (see corresponding reference for details).

In `sims-uasettings`, similar settings were considered, but with 10% of tests being non-null.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

References

- Stephens, M. (2017). False discovery rates: a new deal. Biostatistics, 18(2), 275-294.

See Also

`sims-uasettings`

Examples

```
`uasettings-25-benchmark-bimodal`(metadata = TRUE)

## Not run:
`uasettings-25-benchmark-bimodal`()
`uasettings-25-benchmark-flattop`()
`uasettings-25-benchmark-skew`()
`uasettings-25-benchmark-spiky`()

res <- `uasettings-25-benchmark-spiky`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

`sims-uasettings-t` *Simulation study results (unimodal, t-distributed)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) with 'unimodal' series of effect size distributions (Stephens, 2017) with t_{11} distributed test statistics, and 'cubic' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. In each simulation, 10% of tests were non-null, i.e. had non-zero effect sizes. Simulations were performed with 'bimodal', 'flat top', 'skew' and 'spiky' effect size distributions (see corresponding reference for details).

In `sims-uasettings` and `sims-uasettings-25`, similar settings were considered, but with standard Gaussian distributed test statistics and 25% of tests being non-null, respectively.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

References

- Stephens, M. (2017). False discovery rates: a new deal. Biostatistics, 18(2), 275-294.

See Also

`sims-uasettings`, `sims-uasettings-25`

Examples

```
`uasettings-t-benchmark-bimodal`(metadata = TRUE)

## Not run:
`uasettings-t-benchmark-bimodal`()
`uasettings-t-benchmark-flattop`()
`uasettings-t-benchmark-skew`()
`uasettings-t-benchmark-spiky`()

res <- `uasettings-t-benchmark-spiky`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

`sims-varyinginfo-discrete` *Simulation study results (varying information, discrete)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) across varying covariate informativeness, using standard Gaussian distributed test statistics, and informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. An informative covariate with tunable informativeness was defined with a rough informativeness parameter, delta, defined between 0 and 100. Simulations were performed with delta values of 0, 20, 40, 60, 80, and 100. The conditional probability of a test being null was defined by a discrete, non-smooth, function of the informative covariate.

In `sims-varyinginfo-smooth`, similar settings were considered, but using a smooth relationship between the condition probability of a test being null and the informative covariate.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

See Also

`sims-varyinginfo-smooth`

Examples

```
`varyinginfo-discrete-benchmark-level00`(metadata = TRUE)

## Not run:
`varyinginfo-discrete-benchmark-level00`()
`varyinginfo-discrete-benchmark-level20`()
`varyinginfo-discrete-benchmark-level40`()
`varyinginfo-discrete-benchmark-level60`()
`varyinginfo-discrete-benchmark-level80`()
`varyinginfo-discrete-benchmark-level100`()

res <- `varyinginfo-discrete-benchmark-level100`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

`sims-varyinginfo-smooth` *Simulation study results (varying information, smooth)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) across varying covariate informativeness, using standard Gaussian distributed test statistics, and informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. An informative covariate with tunable informativeness was defined with a rough informativeness parameter, delta, defined between 0 and 100. Simulations were performed with delta values of 0, 20, 40, 60, 80, and 100. The conditional probability of a test being null was defined by a smooth function of the informative covariate.

In `sims-varyinginfo-discrete`, similar settings were considered, but using a discrete, non-smooth, relationship between the condition probability of a test being null and the informative covariate.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

See Also

`sims-varyinginfo-discrete`

Examples

```
`varyinginfo-smooth-benchmark-level00`(metadata = TRUE)

## Not run:
`varyinginfo-smooth-benchmark-level00`()
`varyinginfo-smooth-benchmark-level20`()
`varyinginfo-smooth-benchmark-level40`()
`varyinginfo-smooth-benchmark-level60`()
`varyinginfo-smooth-benchmark-level80`()
`varyinginfo-smooth-benchmark-level100`()

res <- `varyinginfo-smooth-benchmark-level100`()
res[[1]]$informative
res[[1]]$uninformative
```

```
## End(Not run)
```

`sims-varyingntests` *Simulation study results (varying number of tests)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) across varying number of hypotheses tested, using standard Gaussian distributed test statistics, and 'sine' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with 100, 500, 1000, 5000, 10000, and 50000 tests.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

Examples

```
`varyingntests-benchmark-n100`(metadata = TRUE)

## Not run:
`varyingntests-benchmark-n100`()
`varyingntests-benchmark-n500`()
`varyingntests-benchmark-n1000`()
`varyingntests-benchmark-n5000`()
`varyingntests-benchmark-n10000`()
`varyingntests-benchmark-n50000`()

res <- `varyingntests-benchmark-n100`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

`sims-varyingpi0` *Simulation study results (varying null proportion, Gaussian-distributed)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) across varying null proportion of hypotheses with 20000 tests, standard Gaussian distributed test statistics, and 'sine' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 95%, and 99% null tests.

In `sims-varyingpi0-t`, similar settings were considered, but t_{11} distributed test statistics.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

See Also

`sims-varyingpi0-t`

Examples

```
`varyingpi0-benchmark-nullprop05`(metadata = TRUE)

## Not run:
`varyingpi0-benchmark-nullprop05`()
`varyingpi0-benchmark-nullprop10`()
`varyingpi0-benchmark-nullprop20`()
`varyingpi0-benchmark-nullprop30`()
`varyingpi0-benchmark-nullprop40`()
`varyingpi0-benchmark-nullprop50`()
`varyingpi0-benchmark-nullprop60`()
`varyingpi0-benchmark-nullprop70`()
`varyingpi0-benchmark-nullprop80`()
`varyingpi0-benchmark-nullprop90`()
`varyingpi0-benchmark-nullprop95`()
`varyingpi0-benchmark-nullprop99`()

res <- `varyingpi0-benchmark-nullprop99`()
res[[1]]$informative
res[[1]]$uninformative
```

```
## End(Not run)
```

`sims-varyingpi0-t` *Simulation study results (varying null proportion, t-distributed)*

Description

Results for simulations used in Korthauer and Kimes et al. (2019) across varying null proportion of hypotheses with 20000 tests, t_{11} distributed test statistics, and 'sine' informative and paired uninformative covariates stored as a list of SummarizedBenchmark objects from 100 replications. Simulations were performed with 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 95%, and 99% null tests.

In `sims-varyingpi0`, similar settings were considered, but standard Gaussian distributed test statistics.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

The object is a list of length 100 corresponding to the 100 replications. Each entry in the list is a named list of two SummarizedBenchmark objects. The results based on using an informative and uninformative covariate are stored as `"informative"` and `"uninformative"`, respectively.

See Also

`sims-varyingpi0`

Examples

```
`varyingpi0-t-benchmark-nullprop05`(metadata = TRUE)

## Not run:
`varyingpi0-t-benchmark-nullprop05`()
`varyingpi0-t-benchmark-nullprop10`()
`varyingpi0-t-benchmark-nullprop20`()
`varyingpi0-t-benchmark-nullprop30`()
`varyingpi0-t-benchmark-nullprop40`()
`varyingpi0-t-benchmark-nullprop50`()
`varyingpi0-t-benchmark-nullprop60`()
`varyingpi0-t-benchmark-nullprop70`()
`varyingpi0-t-benchmark-nullprop80`()
`varyingpi0-t-benchmark-nullprop90`()
`varyingpi0-t-benchmark-nullprop95`()
`varyingpi0-t-benchmark-nullprop99`()
```

```
res <- `varyingpi0-t-benchmark-nullprop99`()
res[[1]]$informative
res[[1]]$uninformative

## End(Not run)
```

`yeast-insilico` *Yeast in silico dataset results*

Description

Results for differential gene expression analysis using in silico (computationally) spiked-in data generated from yeast data obtained from Schurch et al. (2016) as described in Korthauer and Kimes et al. (2019). Differential expression was tested using **DESeq2**.

Arguments

`metadata`	logical whether to load metadata (TRUE) or full resource (FALSE). (default = FALSE)

Details

Results for this case study include:

- `yeast-results-null5`: null 5 vs. 5 comparison.
- `yeast-results-de5`: using unimodal alternative, 2000 DE genes, strong informative covariate, 5 vs. 5 comparison.
- `yeastH-results-de5`: using unimodal alternative, 500 DE genes, strong informative covariate, 5 vs. 5 comparison.
- `yeastII-results-de5`: using bimodal alternative, 2000 DE genes, strong informative covariate, 5 vs. 5 comparison.
- `yeastIIH-results-de5`: using bimodal alternative, 500 DE genes, strong informative covariate, 5 vs. 5 comparison.
- `yeastW-results-de5`: using unimodal alternative, 2000 DE genes, weak informative covariate, 5 vs. 5 comparison.
- `yeastHW-results-de5`: using unimodal alternative, 500 DE genes, weak informative covariate, 5 vs. 5 comparison.
- `yeastIIW-results-de5`: using bimodal alternative, 2000 DE genes, weak informative covariate, 5 vs. 5 comparison.
- `yeastIIHW-results-de5`: using bimodal alternative, 500 DE genes, weak informative covariate, 5 vs. 5 comparison.
- `yeast-results-de5-uninfCov`: using unimodal alternative, 2000 DE genes, uninformative covariate, 5 vs. 5 comparison.

- `yeastH-results-de5-uninfCov`: using unimodal alternative, 500 DE genes, uninformative covariate, 5 vs. 5 comparison.
- `yeastII-results-de5-uninfCov`: using bimodal alternative, 2000 DE genes, uninformative covariate, 5 vs. 5 comparison.
- `yeastIIH-results-de5-uninfCov`: using bimodal alternative, 500 DE genes, uninformative covariate, 5 vs. 5 comparison.
- `yeast-results-null10`: null 10 vs. 10 comparison.
- `yeast-results-de10`: using unimodal alternative, 2000 DE genes, strong informative covariate, 10 vs. 10 comparison.
- `yeastH-results-de10`: using unimodal alternative, 500 DE genes, strong informative covariate, 10 vs. 10 comparison.
- `yeastII-results-de10`: using bimodal alternative, 2000 DE genes, strong informative covariate, 10 vs. 10 comparison.
- `yeastIIH-results-de10`: using bimodal alternative, 500 DE genes, strong informative covariate, 10 vs. 10 comparison.
- `yeastW-results-de10`: using unimodal alternative, 2000 DE genes, weak informative covariate, 10 vs. 10 comparison.
- `yeastHW-results-de10`: using unimodal alternative, 500 DE genes, weak informative covariate, 10 vs. 10 comparison.
- `yeastIIW-results-de10`: using bimodal alternative, 2000 DE genes, weak informative covariate, 10 vs. 10 comparison.
- `yeastIIHW-results-de10`: using bimodal alternative, 500 DE genes, weak informative covariate, 10 vs. 10 comparison.
- `yeast-results-de10-uninfCov`: using unimodal alternative, 2000 DE genes, uninformative covariate, 10 vs. 10 comparison.
- `yeastH-results-de10-uninfCov`: using unimodal alternative, 500 DE genes, uninformative covariate, 10 vs. 10 comparison.
- `yeastII-results-de10-uninfCov`: using bimodal alternative, 2000 DE genes, uninformative covariate, 10 vs. 10 comparison.
- `yeastIIH-results-de10-uninfCov`: using bimodal alternative, 500 DE genes, uninformative covariate, 10 vs. 10 comparison.

References

- Schurch, N. J. et al. (2016). How many biological replicates are needed in an RNA-seq experiment and which differential expression tool should you use?. RNA, 22(6), 839-851.

See Also

`polyester-insilico`

Examples

```
`yeast-results-de5`(metadata = TRUE)

## Not run:
`yeast-results-de5`()
`yeast-results-de5-uninfCov`()
`yeast-results-de10`()
`yeast-results-de10-uninfCov`()
`yeast-results-null5`()
`yeast-results-null10`()
`yeastH-results-de5`()
`yeastH-results-de5-uninfCov`()
`yeastH-results-de10`()
`yeastH-results-de10-uninfCov`()
`yeastHW-results-de5`()
`yeastHW-results-de10`()
`yeastII-results-de5`()
`yeastII-results-de5-uninfCov`()
`yeastII-results-de10`()
`yeastII-results-de10-uninfCov`()
`yeastIIH-results-de5`()
`yeastIIH-results-de5-uninfCov`()
`yeastIIH-results-de10`()
`yeastIIH-results-de10-uninfCov`()
`yeastIIHW-results-de5`()
`yeastIIHW-results-de10`()
`yeastIIW-results-de5`()
`yeastIIW-results-de10`()
`yeastW-results-de5`()
`yeastW-results-de10`()

## End(Not run)
```

Index