Package: IntrinsicScan (via r-universe)

October 13, 2026

Type Package

Title Label-Blind Feature-Wise Intrinsic Binary Partition Scanning

Version 0.99.4

Description Scans continuous molecular features for intrinsic binary partitions without using phenotype labels to determine the split. For each feature, observations are divided into two one-dimensional states by selecting the split that minimizes the total within-state L1 loss, defined here as the sum of absolute distances between each observation and the medoid of its assigned state. Thus, unlike a squared-error objective, deviations are measured according to their absolute magnitude rather than being squared. Binary phenotype labels are used only after the partition is fixed to orient the two states, quantify phenotype concordance using orientation-free Youden J, perform exact conditional inference, and optionally assess resampling stability or fixed-partition permutation calibration. The method is intended for preprocessed continuous omics measurements and does not imply that a fitted partition represents true biological bimodality or discrete biological states.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.5.0)

Imports Biobase, S4Vectors, SummarizedExperiment, withr

Suggests BiocStyle, knitr, rmarkdown, leukemiasEset, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

biocViews GeneExpression, Microarray, RNASeq, Classification, Clustering

URL https://github.com/memedesimo98/IntrinsicScan

BugReports https://github.com/memedesimo98/IntrinsicScan/issues

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.1

Config/pak/sysreqs zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-10-10 07:28:35 UTC

RemoteUrl https://github.com/bioc/IntrinsicScan

RemoteRef HEAD

RemoteSha 8c2ac4e72f77bdf1f8d304b69aa0ee56fa47b8f4

Contents

intrinsicPermutation 2
intrinsicResults 3
intrinsicScan 6
topIntrinsic 8

Index **9**

`intrinsicPermutation` *Fixed-partition permutation calibration*

Description

Permutes phenotype labels while keeping label-blind full-sample partitions fixed.

Usage

```
intrinsicPermutation(
  x,
  nPermutations = 1000L,
  cutoffs = c(0.8, 1),
  seed = 1L,
  verbose = TRUE
)
```

Arguments

`x`	An IntrinsicScan result created with keepStates=TRUE.
`nPermutations`	Number of label permutations.
`cutoffs`	J cutoffs for global-null counts.
`seed`	Random seed.
`verbose`	Print progress.

Value

The updated SummarizedExperiment with maxT FWER results.

Examples

```
x <- rbind(
    feature_A = c(1, 1.1, 1.2, 1.3, 1.4, 5, 5.1, 5.2, 5.3, 5.4),
    feature_B = c(1, 2, 3, 2, 3, 2, 3, 4, 3, 4)
)
colnames(x) <- paste0("S", seq_len(ncol(x)))
y <- rep(c("control", "case"), each = 5)
fit <- intrinsicScan(
    x, y, positive = "case", keepStates = TRUE, verbose = FALSE
)
fit <- intrinsicPermutation(
    fit, nPermutations = 10L, seed = 1L, verbose = FALSE
)
intrinsicResults(fit)
```

`intrinsicResults` *Extract IntrinsicScan feature results*

Description

Extract IntrinsicScan feature results

Usage

```
intrinsicResults(x)
```

Arguments

`x`	A result returned by `intrinsicScan()`.

Value

An `S4Vectors::DataFrame` with one row per analyzed feature, ordered by decreasing orientation-free Youden J, increasing Benjamini-Hochberg-adjusted exact-J P value, and decreasing relative gap. Columns describe the fitted intrinsic partition, phenotype concordance, statistical evidence, ranking, and, when requested, resampling stability.

Result columns

The table returned by `intrinsicResults()` contains one row per analyzed feature. Columns are grouped conceptually as follows.

Feature and partition definition

- `feature_id`: feature identifier.

- `valid`: whether a valid two-state partition could be fitted.
- `reason`: reason a feature could not be partitioned, if applicable.
- `threshold`: midpoint between the two adjacent observed values defining the selected low/high split.
- `low_medoid`, `high_medoid`: medoids of the fitted low and high states.
- `medoid_distance`: difference between the high- and low-state medoids.
- `gap`: distance between the two adjacent observations on either side of the selected threshold.
- `relative_gap`: `gap` divided by the feature interquartile range (IQR); `NA` when the IQR is zero or non-finite.
- `within_l1_cost`: minimized total within-state L1 loss, i.e. the sum of absolute distances between observations and their assigned state medoid.
- `high_state_size`: number of samples assigned to the fitted high state.

Phenotype concordance

- `J`: orientation-free Youden index. Values range from 0 to 1, with larger values indicating stronger concordance between the intrinsic partition and phenotype labels.
- `balanced_accuracy`: mean of sensitivity and specificity after orienting the fitted states relative to the phenotype-positive class.
- `sensitivity`: sensitivity after post-hoc orientation of the states.
- `specificity`: specificity after post-hoc orientation of the states.
- `accuracy`: fraction of samples correctly classified after orientation.
- `positive_state`: `"high"` or `"low"`, indicating which fitted state is associated with the phenotype-positive class.
- `exact_full_concordance`: whether the fixed partition achieves both sensitivity and specificity equal to 1 within numerical tolerance.

Statistical evidence and descriptive benchmarks

- `exact_J_p`: exact conditional P value for the observed orientation-free J, conditioning on phenotype class sizes and fitted high-state size.
- `exact_J_fdr`: Benjamini-Hochberg-adjusted `exact_J_p`; this is the primary multiple-testing-adjusted inferential quantity used by IntrinsicScan.
- `fisher_p`, `fisher_fdr`: two-sided Fisher exact-test P value for the fixed state-by-phenotype table and its Benjamini-Hochberg adjustment. These are retained as familiar benchmarks rather than as the primary test.
- `orientation_free_auc`: `max(AUC, 1 - AUC)`, so values range from 0.5 to 1 independently of whether higher or lower feature values correspond to the phenotype-positive class.
- `wilcox_p`, `wilcox_fdr`: Wilcoxon rank-sum P value comparing feature values between phenotype classes and its Benjamini-Hochberg adjustment; these are retained as distributional benchmarks.
- `median_difference_positive_minus_negative`: phenotype-positive median minus phenotype-negative median.

- rank_J: feature rank based on decreasing J, then increasing exact-J FDR, then decreasing relative gap.

Optional resampling-stability summaries

These columns are NA for features not selected for stability assessment.

- stability_seed: deterministic feature-specific random seed used for resampling.
- n_resamples_valid: number of requested resamples in which a valid training partition could be fitted.
- n_heldout_valid: number of resamples in which held-out samples from both phenotype classes were available and a held-out J could be calculated.
- threshold_median: median fitted threshold across valid resamples.
- threshold_q025, threshold_q975: 2.5th and 97.5th percentiles of fitted thresholds across valid resamples.
- threshold_iqr: interquartile range of fitted thresholds across valid resamples.
- threshold_iqr_scaled: threshold IQR divided by the full-sample feature IQR.
- median_training_J: median training-set J across valid resamples.
- median_heldout_J: median J obtained after applying the training-derived threshold and phenotype orientation to held-out samples.
- heldout_J_q25, heldout_J_q75: 25th and 75th percentiles of held-out J.
- heldout_J_q025, heldout_J_q975: 2.5th and 97.5th percentiles of held-out J.
- positive_state_stability: fraction of valid resamples in which the training-derived phenotype-positive state agrees with the full-sample orientation.
- state_partition_stability: mean agreement between resample-derived high/low state assignments and the full-sample state partition when the resample-derived threshold is applied to all original samples.
- exact_training_fraction: fraction of valid resamples with training J equal to 1 within numerical tolerance.
- exact_heldout_fraction: fraction of valid held-out evaluations with held-out J equal to 1 within numerical tolerance.
- median_heldout_n_positive, median_heldout_n_negative: median numbers of phenotype-positive and phenotype-negative samples contributing to valid held-out evaluations.

Examples

```
x <- rbind(
  feature_A = c(1, 1.1, 1.2, 1.3, 1.4, 5, 5.1, 5.2, 5.3, 5.4),
  feature_B = c(1, 2, 3, 2, 3, 2, 3, 4, 3, 4)
)
colnames(x) <- paste0("S", seq_len(ncol(x)))
y <- rep(c("control", "case"), each = 5)
fit <- intrinsicScan(x, y, positive = "case", verbose = FALSE)
intrinsicResults(fit)
```

`intrinsicScan` *Scan features for intrinsic binary partitions*

Description

`intrinsicScan()` identifies intrinsic binary partitions in continuous molecular features without using phenotype labels to determine the split. For each feature, it finds the exact global optimum of a one-dimensional two-medoid L1 objective, where L1 loss is the sum of absolute distances between observations and the medoid of their assigned state. Phenotype labels are used only after the partition has been fixed.

Usage

```
intrinsicScan(
  x,
  phenotype,
  positive,
  assay = 1L,
  nResamples = 0L,
  resampleMethod = c("subsample", "bootstrap", "none"),
  resampleFraction = 0.8,
  resampleTop = 500L,
  seed = 1L,
  keepStates = TRUE,
  verbose = TRUE
)
```

Arguments

`x`	A numeric matrix/data.frame with features in rows and samples in columns, a `SummarizedExperiment`, or an `ExpressionSet`.
`phenotype`	Either a vector of binary phenotype labels aligned to the columns of `x`, or (for Bioconductor containers) the name of a column in `colData(x)` / `pData(x)`.
`positive`	Explicit phenotype level to treat as positive when orienting the already-fixed low/high partition.
`assay`	Assay name or index when `x` is a `SummarizedExperiment`.
`nResamples`	Number of internal stability resamples. The default is zero because stability is optional and can be computationally expensive.
`resampleMethod`	One of `"none"`, `"subsample"`, or `"bootstrap"`.
`resampleFraction`	Training fraction for subsampling.
`resampleTop`	Number of top-ranked features for which stability is assessed. May be `Inf`.
`seed`	Master random seed. Feature-specific stability seeds are derived deterministically from this seed and the feature identifier.

`keepStates`	Logical; store the full high/low state assignment matrix as assay `"state"` in the returned object.
`verbose`	Logical; print progress messages.

Details

For each feature independently, the observed values are sorted and every admissible split between two distinct adjacent values is considered. Each candidate split defines a low and a high state. Its L1 loss is the sum of the absolute distances between each observation and the medoid of its assigned state, and the split with the minimum total L1 loss is selected. The fitted threshold is placed halfway between the two adjacent observed values defining the selected split. If more than one split has the same optimum within numerical tolerance, ties are resolved deterministically, first by preferring the largest gap between adjacent states and then the split closest to the middle of the ordered observations.

Phenotype labels do not participate in fitting the partition. They are used only after the partition has been fixed to determine which state is more associated with the phenotype-positive class and to quantify concordance. Concordance is measured by the orientation-free Youden index $J = |TPR_{high} - FPR_{high}|$, equivalent after orientation to $J = sensitivity + specificity - 1$. Thus, $J = 0$ indicates no phenotype concordance and $J = 1$ indicates perfect concordance in the analyzed samples.

The primary exact-J P value is calculated conditionally on the numbers of phenotype-positive samples, phenotype-negative samples, and samples assigned to the high state. Under exchangeable phenotype labels, the number of phenotype-positive samples in the high state follows a hypergeometric distribution. Multiple-testing correction is performed using the Benjamini-Hochberg procedure.

No normalization, transformation, missing-value imputation, or feature filtering is performed. Inputs must therefore be complete, finite, preprocessed continuous measurements. A high phenotype-concordance score does not by itself establish bimodality or discrete biological states.

Value

A `SummarizedExperiment` containing the analyzed molecular features in assay `"expression"`, reordered according to the IntrinsicScan ranking. Features are ranked first by decreasing orientation-free Youden J, then by increasing Benjamini-Hochberg-adjusted exact-J P value, and finally by decreasing relative gap between the fitted low and high states. If `keepStates = TRUE`, logical high-state assignments are stored in assay `"state"`. Feature-level scan statistics are appended to `rowData()` with prefix `scan_` and can also be retrieved without the prefix using `intrinsicResults()`.

Examples

```
x <- rbind(
  feature_A = c(1, 1.1, 1.2, 1.3, 1.4, 5, 5.1, 5.2, 5.3, 5.4),
  feature_B = c(1, 2, 3, 2, 3, 2, 3, 4, 3, 4)
)
colnames(x) <- paste0("S", seq_len(ncol(x)))
y <- rep(c("control", "case"), each = 5)
fit <- intrinsicScan(x, y, positive = "case", verbose = FALSE)
intrinsicResults(fit)
```

topIntrinsic *Filter top IntrinsicScan features*

Description

Filters and ranks results without collapsing effect, evidence, and stability into a composite category.

Usage

```
topIntrinsic(x, n = 50L, minJ = 0.8, maxFDR = 0.05,
  requireStability = FALSE, minHeldoutJ = NULL,
  minOrientationStability = NULL, minPartitionStability = NULL,
  minExactHeldout = NULL, maxFWER = NULL)
```

Arguments

`x`	An IntrinsicScan result.
`n`	Maximum rows.
`minJ`	Minimum Youden J.
`maxFDR`	Maximum exact-J FDR.
`requireStability`	Require stability estimates.
`minHeldoutJ`	Optional minimum median held-out J.
`minOrientationStability`	Optional minimum orientation stability.
`minPartitionStability`	Optional minimum state-partition stability.
`minExactHeldout`	Optional minimum exact-heldout fraction.
`maxFWER`	Optional maximum maxT FWER P value.

Value

A filtered S4Vectors DataFrame.

Examples

```
x <- rbind(
    feature_A = c(1, 1.1, 1.2, 1.3, 1.4, 5, 5.1, 5.2, 5.3, 5.4),
    feature_B = c(1, 2, 3, 2, 3, 2, 3, 4, 3, 4)
)
colnames(x) <- paste0("S", seq_len(ncol(x)))
y <- rep(c("control", "case"), each = 5)
fit <- intrinsicScan(x, y, positive = "case", verbose = FALSE)
topIntrinsic(fit, n = 2, minJ = 0, maxFDR = 1)
```

Index

intrinsicPermutation, 2
intrinsicResults, 3
intrinsicScan, 6

topIntrinsic, 8