Package: glyfun (via r-universe)

May 20, 2026

Title Glycan-Centric Functional Enrichment Analysis

Version 0.1.0

Description Provides functional enrichment analysis for glycoproteomics data, including both protein-centric and glycan-centric approaches. The 'enrich_xxx()' functions answer ``Which functions are enriched for proteins with dysregulated glycosylation?'' (traditional protein-level enrichment), while 'enrich_gc_xxx()' functions answer ``Which functions are enriched for dysregulated glycan traits?'' (glycan-centric, linking glycan traits like core-fucosylation with functional annotations). Supports both Over Representation Analysis (ORA) and Gene Set Enrichment Analysis (GSEA) with common databases including GO, KEGG, Reactome, WikiPathways, DO (Disease Ontology), and NCG (Network of Cancer Genes). Integrates seamlessly with 'glydet' (for site-specific derived traits) and 'glystats' (for differential expression analysis).

License MIT + file LICENSE

Suggests org.Hs.eg.db, ReactomePA, DOSE, GOSemSim, glyexp, glyrepr, glyclean, glydet, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://glycoverse.github.io/glyfun/

Imports checkmate, cli, clusterProfiler, dplyr, glystats (>= 0.8.0), purrr, rlang, tibble, tidyselect

VignetteBuilder knitr

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libglpk-dev make libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev zlib1g-dev

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

Date/Publication 2026-04-25 05:50:12 UTC

RemoteUrl https://github.com/glycoverse/glyfun

RemoteRef v0.1.0

RemoteSha 307b4163e143da99f14d5bd025d305d007a03f58

Contents

detected_universe . . . 2
enrich_gc_gsea_do . . . 3
enrich_gc_gsea_go . . . 5
enrich_gc_gsea_kegg . . . 8
enrich_gc_gsea_ncg . . . 10
enrich_gc_gsea_reactome . . . 13
enrich_gc_gsea_wp . . . 15
enrich_gc_ora_do . . . 18
enrich_gc_ora_go . . . 20
enrich_gc_ora_kegg . . . 22
enrich_gc_ora_ncg . . . 24
enrich_gc_ora_reactome . . . 26
enrich_gc_ora_wp . . . 29
enrich_gsea_do . . . 31
enrich_gsea_go . . . 33
enrich_gsea_kegg . . . 35
enrich_gsea_ncg . . . 37
enrich_gsea_reactome . . . 39
enrich_gsea_wp . . . 41
enrich_ora_do . . . 43
enrich_ora_go . . . 45
enrich_ora_kegg . . . 47
enrich_ora_ncg . . . 49
enrich_ora_reactome . . . 50
enrich_ora_wp . . . 52

Index **55**

`detected_universe` *Helper function to prepare the* `universe` *parameter*

Description

This function extracts all detected proteins in a `glyexp::experiment()` or a `glystats` result. It can be readily passed to the `universe` parameter of all `glyfun` functions.

Usage

```
detected_universe(x)
```

Arguments

x A glyexp::experiment() or a glystats result.

Value

A character vector of protein UniProt IDs.

Examples

```
library(glyexp)
universe <- detected_universe(real_experiment)
length(universe)
universe[1:5]
```

---

enrich_gc_gsea_do *Glycan-Centric Disease Ontology (DO) Gene Set Enrichment Analysis*

Description

Performs glycan-centric Disease Ontology (DO) Gene Set Enrichment Analysis (GSEA). It ranks proteins within each glycan trait separately, then compares enriched disease terms across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_do(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  ont = "HDO",
  organism = "hsa",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

dea_res Differential analysis result. Can be one of:

- Result from glystats::gly_limma() (two groups), glystats::gly_ttest(), or glystats::gly_wilcox(), called on an glyexp::experiment() of "traitproteomics" type.
- A tibble with the following columns:

- protein: Uniprot ID of proteins
- trait: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
- site: The glycosylation site.
- p_val: p-values, preferably adjusted p-values
- log2fc: log2 of fold change

rank_by
: Criteria for ranking proteins. One of the following:
- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

aggr
: Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

ont
: One of "HDO" (Human Disease Ontology), "MPO" (Mammalian Phenotype Ontology), or "VDO" (Vector Disease Ontology). Passed to ont of DOSE::gseDO(). Defaults to "HDO".

organism
: "hsa" (Homo sapiens) or "mmu" (Mus musculus). Passed to organism of DOSE::gseDO(). Defaults to "hsa".

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to pAdjustMethod of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to pvalueCutoff of downstream enrichment function. Defaults to 0.05.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to minGSSize of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to maxGSSize of downstream enrichment function. Defaults to 500.

exponent
: Weight of each step. Passed to exponent of clusterProfiler::gseGO(). Defaults to 1.

eps
: Epsilon for calculating p-values. Passed to eps of clusterProfiler::gseGO(). Defaults to 1e-10.

seed
: Logical indicating whether to set a random seed for reproducibility. Passed to seed of clusterProfiler::gseGO(). Defaults to FALSE.

Value

A clusterProfiler compareClusterResult object. It can be readily converted to a tibble with tibble::as_tibble(), or visualized with clusterProfiler functions like clusterProfiler::dotplot().

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `DOSE::gseDO()`

`enrich_gc_gsea_go` *Glycan-Centric GO Gene Set Enrichment Analysis*

Description

Performs glycan-centric Gene Ontology (GO) Gene Set Enrichment Analysis (GSEA). Instead of traditional protein-centric enrichment, this function links specific glycan traits (e.g., core-fucosylation, sialylation) to functional annotations. It ranks proteins within each glycan trait separately, then compares enriched biological functions across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_go(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  orgdb = "org.Hs.eg.db",
  ont = "MF",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`orgdb` An OrgDb object. Passed to `OrgDb` of downstream enrichment function.

`ont` Ontology type. Passed to `ont` of `clusterProfiler::enrichGO()`. "BP", "MF", "CC", or "ALL". Defaults to "MF".

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

- `p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.
- `min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.
- `max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.
- `exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.
- `eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.
- `seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `compareClusterResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`
```

```
# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `clusterProfiler::gseGO()`

---

enrich_gc_gsea_kegg *Glycan-Centric KEGG Gene Set Enrichment Analysis*

Description

Performs glycan-centric KEGG pathway Gene Set Enrichment Analysis (GSEA). It ranks proteins within each glycan trait separately, then compares enriched pathways across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_kegg(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "hsa",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.

  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`organism` KEGG organism code. Defaults to "hsa" (Homo sapiens). See `clusterProfiler::gseKEGG()` for details.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

`exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

`eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

`seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `compareClusterResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked

highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `clusterProfiler::gseKEGG()`

---

enrich_gc_gsea_ncg — *Glycan-Centric Network of Cancer Genes (NCG) Gene Set Enrichment Analysis*

Description

Performs glycan-centric Network of Cancer Genes (NCG) Gene Set Enrichment Analysis (GSEA). It ranks proteins within each glycan trait separately, then compares enriched cancer gene sets across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_ncg(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
```

```
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

`exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

`eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

`seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `compareClusterResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `DOSE::gseNCG()`

enrich_gc_gsea_reactome
Glycan-Centric Reactome Gene Set Enrichment Analysis

Description

Performs glycan-centric Reactome pathway Gene Set Enrichment Analysis (GSEA). It ranks proteins within each glycan trait separately, then compares enriched pathways across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_reactome(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "human",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change

- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

aggr	Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".
organism	Reactome organism name. Passed to `organism` of `ReactomePA::gsePathway()`. One of "human", "rat", "mouse", "celegans", "yeast", "zebrafish", "fly". Defaults to "human".
p_adj_method	P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".
p_cutoff	P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.
min_gs_size	Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.
max_gs_size	Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.
exponent	Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.
eps	Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.
seed	Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `compareClusterResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking

and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `ReactomePA::gsePathway()`

---

enrich_gc_gsea_wp *Glycan-Centric WikiPathways Gene Set Enrichment Analysis*

Description

Performs glycan-centric WikiPathways Gene Set Enrichment Analysis (GSEA). It ranks proteins within each glycan trait separately, then compares enriched pathways across traits by running GSEA for each trait.

Usage

```
enrich_gc_gsea_wp(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "Homo sapiens",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`organism` WikiPathways organism name. Passed to `organism` of `clusterProfiler::gseWP()`. Defaults to "Homo sapiens". Use `clusterProfiler::get_wp_organisms()` to see available organisms.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

`exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

`eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

`seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `compareClusterResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_gsea_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like "Which functions are enriched in proteins ranked highly for core-fucosylation changes?" Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function first splits the DEA result by glycan trait, then ranks proteins within each trait separately. For each trait, it applies the same ranking and aggregation logic as `enrich_gsea_go()`, producing one ranked protein list per trait. Those trait-specific ranked lists are then compared with `clusterProfiler::compareCluster()` using its formula interface.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_gsea_go(dea_res)  # or other `enrich_gc_gsea_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `clusterProfiler::gseWP()`

enrich_gc_ora_do *Glycan-Centric Disease Ontology (DO) Over Representation Analysis*

Description

Performs glycan-centric Disease Ontology (DO) Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits to disease associations. It helps answer questions like "Which diseases are enriched in proteins with a specific dysregulated glycan motif?", by grouping differential analysis results by glycan traits and computing disease enrichment for each trait.

Usage

```
enrich_gc_ora_do(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  ont = "HDO",
  organism = "hsa",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)`

	means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.
`ont`	One of "HDO" (Human Disease Ontology), "MPO" (Mammalian Phenotype Ontology), or "VDO" (Vector Disease Ontology). Passed to `ont` of `DOSE::enrichDO()`. Defaults to "HDO".
`organism`	"hsa" (Homo sapiens) or "mmu" (Mus musculus). Passed to `organism` of `DOSE::enrichDO()`. Defaults to "hsa".
`universe`	Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.
`p_adj_method`	P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".
`p_cutoff`	P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.
`q_cutoff`	Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.
`min_gs_size`	Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.
`max_gs_size`	Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

clusterProfiler::compareCluster(), DOSE::enrichDO()

---

enrich_gc_ora_go   *Glycan-Centric GO Over Representation Analysis*

Description

Performs glycan-centric Gene Ontology (GO) Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits (e.g., core-fucosylation, sialylation) to functional annotations. It identifies which biological functions are significantly enriched in proteins exhibiting specific glycosylation changes, grouping the differential analysis results by trait before performing ORA.

Usage

```
enrich_gc_ora_go(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  orgdb = "org.Hs.eg.db",
  ont = "MF",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res`   Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:

- `protein`: Uniprot ID of proteins
- `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
- `site`: The glycosylation site.
- `p_val`: p-values, preferably adjusted p-values
- `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`orgdb` An OrgDb object. Passed to `OrgDb` of downstream enrichment function.

`ont` Ontology type. Passed to `ont` of `clusterProfiler::enrichGO()`. "BP", "MF", "CC", or "ALL". Defaults to "MF".

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`q_cutoff` Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

enrich_gc_xxx() functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", enrich_gc_xxx() answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

clusterProfiler::compareCluster(), clusterProfiler::enrichGO()

---

enrich_gc_ora_kegg *Glycan-Centric KEGG Over Representation Analysis*

Description

Performs glycan-centric KEGG pathway Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits to biological pathways. It helps answer questions like "Which pathways are enriched in proteins with a specific dysregulated glycan motif?", by grouping differential analysis results by glycan traits and computing pathway enrichment for each trait.

Usage

```
enrich_gc_ora_kegg(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "hsa",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`organism` KEGG organism code. Passed to `organism` of `clusterProfiler::enrichKEGG()`. Defaults to "hsa" (Homo sapiens). Common codes: "hsa" (human), "mmu" (mouse), "rno" (rat).

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`q_cutoff` Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `clusterProfiler::enrichKEGG()`

---

enrich_gc_ora_ncg — *Glycan-Centric Network of Cancer Genes (NCG) Over Representation Analysis*

Description

Performs glycan-centric Network of Cancer Genes (NCG) Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits to cancer gene associations. It helps answer questions like "Which cancer gene sets are enriched in proteins with a specific dysregulated glycan motif?", by grouping differential analysis results by glycan traits and computing cancer gene enrichment for each trait.

Usage

```
enrich_gc_ora_ncg(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
```

```
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`q_cutoff` Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `DOSE::enrichNCG()`

`enrich_gc_ora_reactome` *Glycan-Centric Reactome Pathway Over Representation Analysis*

Description

Performs glycan-centric Reactome pathway Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits to biological pathways. It helps answer questions like "Which Reactome pathways are enriched in proteins with a specific dysregulated glycan motif?", by grouping differential analysis results by glycan traits and computing pathway enrichment for each trait.

Usage

```
enrich_gc_ora_reactome(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "human",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff` Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`organism` Reactome organism name. Passed to `organism` of `ReactomePA::enrichPathway()`. One of "human", "rat", "mouse", "celegans", "yeast", "zebrafish", "fly". Defaults to "human".

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`q_cutoff` Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

`clusterProfiler::compareCluster()`, `ReactomePA::enrichPathway()`

enrich_gc_ora_wp *Glycan-Centric WikiPathways Over Representation Analysis*

Description

Performs glycan-centric WikiPathways Over-Representation Analysis (ORA). Instead of traditional protein-centric enrichment, this function links specific glycan traits to biological pathways. It helps answer questions like "Which WikiPathways are enriched in proteins with a specific dysregulated glycan motif?", by grouping differential analysis results by glycan traits and computing pathway enrichment for each trait.

Usage

```
enrich_gc_ora_wp(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "Homo sapiens",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

organism
: WikiPathways organism name. Passed to `organism` of `clusterProfiler::enrichWP()`. Defaults to "Homo sapiens". Use `clusterProfiler::get_wp_organisms()` to see available organisms.

universe
: Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `compareClusterResult` object with additional `glyfun` classes. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

What is glycan-centric enrichment?

In traditional glycoproteomics data analysis, we usually perform differential expression analysis (DEA) on glycoforms, extract proteins that have dysregulated glycosylation, then perform functional enrichment (e.g. GO) on these proteins. This is what `enrich_xxx()` functions do (e.g. `enrich_ora_go()`).

`enrich_gc_xxx()` functions differ in that they link specific glycan traits with functional annotations. Instead of answering the question "Which functions are enriched in dysregulated glycoproteins?", `enrich_gc_xxx()` answers questions like “Which functions are enriched in proteins with dysregulated core-fucosylation?” Higher specificity, deeper insights. By focusing on distinct glycan motifs, it helps you pinpoint the functional relevance of specific glycosylation changes.

Common usage pattern

A common pattern of using this function is:

```
# 1. Use `glydet` to calculate derived traits or motif quantification.
trait_exp <- derive_traits(exp)  # or `quantify_motifs()`

# 2. Perform differential analysis with `glystats`.
```

```
dea_res <- gly_ttest(trait_exp)

# 3. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other `enrich_gc_xxx()` functions
```

See Also

clusterProfiler::compareCluster(), clusterProfiler::enrichWP()

---

enrich_gsea_do   *Disease Ontology (DO) Gene Set Enrichment Analysis*

Description

Performs Disease Ontology (DO) Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_do(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  ont = "HDO",
  organism = "hsa",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from glystats::gly_limma() (two groups), glystats::gly_ttest(), or glystats::gly_wilcox(), called on an glyexp::experiment() of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values

- log2fc: log2 of fold change

rank_by
: Criteria for ranking proteins. One of the following:
  - "log2fc": log2 fold change with signs
  - "abs_log2fc": absolute log2 fold change
  - "log10p": negative log10 p-value
  - "signed_log10p" (default): log10 p-value with signs of log2 fold change
  - "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

aggr
: Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

ont
: One of "HDO" (Human Disease Ontology), "MPO" (Mammalian Phenotype Ontology), or "VDO" (Vector Disease Ontology). Passed to `ont` of `DOSE::gseDO()`. Defaults to "HDO".

organism
: "hsa" (Homo sapiens) or "mmu" (Mus musculus). Passed to `organism` of `DOSE::gseDO()`. Defaults to "hsa".

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

exponent
: Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

eps
: Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

seed
: Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `gseaResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::ridgeplot()`.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use `rank_by` to specify other ranking criteria, such as p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

DOSE::gseDO()

---

enrich_gsea_go *GO Gene Set Enrichment Analysis*

Description

Performs Gene Ontology (GO) Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_go(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  orgdb = "org.Hs.eg.db",
  ont = "MF",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins

- trait: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
- site: The glycosylation site.
- p_val: p-values, preferably adjusted p-values
- log2fc: log2 of fold change

rank_by
: Criteria for ranking proteins. One of the following:
  - "log2fc": log2 fold change with signs
  - "abs_log2fc": absolute log2 fold change
  - "log10p": negative log10 p-value
  - "signed_log10p" (default): log10 p-value with signs of log2 fold change
  - "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

aggr
: Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

orgdb
: An OrgDb object. Passed to OrgDb of downstream enrichment function.

ont
: Ontology type. Passed to ont of clusterProfiler::enrichGO(). "BP", "MF", "CC", or "ALL". Defaults to "MF".

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to pAdjustMethod of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to pvalueCutoff of downstream enrichment function. Defaults to 0.05.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to minGSSize of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to maxGSSize of downstream enrichment function. Defaults to 500.

exponent
: Weight of each step. Passed to exponent of clusterProfiler::gseGO(). Defaults to 1.

eps
: Epsilon for calculating p-values. Passed to eps of clusterProfiler::gseGO(). Defaults to 1e-10.

seed
: Logical indicating whether to set a random seed for reproducibility. Passed to seed of clusterProfiler::gseGO(). Defaults to FALSE.

Value

A clusterProfiler gseaResult object. It can be readily converted to a tibble with tibble::as_tibble(), or visualized with clusterProfiler functions like clusterProfiler::ridgeplot().

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use rank_by to specify other ranking criteria, such as

p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

`clusterProfiler::gseGO()`

---

enrich_gsea_kegg *KEGG Gene Set Enrichment Analysis*

Description

Performs KEGG pathway Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_kegg(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "hsa",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.

- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`organism` KEGG organism code. Defaults to "hsa" (Homo sapiens). See `clusterProfiler::gseKEGG()` for details.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

`exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

`eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

`seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `gseaResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::ridgeplot()`.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use `rank_by` to specify other ranking criteria, such as p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

`clusterProfiler::gseKEGG()`

---

enrich_gsea_ncg — *Network of Cancer Genes (NCG) Gene Set Enrichment Analysis*

Description

Performs Network of Cancer Genes (NCG) Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_ncg(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`rank_by` Criteria for ranking proteins. One of the following:

- "log2fc": log2 fold change with signs
- "abs_log2fc": absolute log2 fold change
- "log10p": negative log10 p-value
- "signed_log10p" (default): log10 p-value with signs of log2 fold change
- "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

`aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

`min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

`max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

`exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.

`eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.

`seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `gseaResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::ridgeplot()`.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use `rank_by` to specify other ranking criteria, such as p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

`DOSE::gseNCG()`

---

enrich_gsea_reactome *Reactome Gene Set Enrichment Analysis*

Description

Performs Reactome pathway Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_reactome(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "human",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

dea_res
: Differential analysis result. Can be one of:
  - Result from glystats::gly_limma() (two groups), glystats::gly_ttest(), or glystats::gly_wilcox(), called on an glyexp::experiment() of "traitproteomics" type.
  - A tibble with the following columns:
    - protein: Uniprot ID of proteins
    - trait: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
    - site: The glycosylation site.
    - p_val: p-values, preferably adjusted p-values
    - log2fc: log2 of fold change

rank_by
: Criteria for ranking proteins. One of the following:
  - "log2fc": log2 fold change with signs
  - "abs_log2fc": absolute log2 fold change
  - "log10p": negative log10 p-value
  - "signed_log10p" (default): log10 p-value with signs of log2 fold change
  - "log2fc_log10p": log2 fold change multiplied by negative log10 p-value

aggr
: Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".

organism
: Reactome organism name. Passed to organism of ReactomePA::gsePathway(). One of "human", "rat", "mouse", "celegans", "yeast", "zebrafish", "fly". Defaults to "human".

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to pAdjustMethod of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to pvalueCutoff of downstream enrichment function. Defaults to 0.05.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to minGSSize of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to maxGSSize of downstream enrichment function. Defaults to 500.

exponent
: Weight of each step. Passed to exponent of clusterProfiler::gseGO(). Defaults to 1.

eps
: Epsilon for calculating p-values. Passed to eps of clusterProfiler::gseGO(). Defaults to 1e-10.

seed
: Logical indicating whether to set a random seed for reproducibility. Passed to seed of clusterProfiler::gseGO(). Defaults to FALSE.

Value

A clusterProfiler `gseaResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::ridgeplot()`.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use `rank_by` to specify other ranking criteria, such as p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

`ReactomePA::gsePathway()`

---

enrich_gsea_wp *WikiPathways Gene Set Enrichment Analysis*

Description

Performs WikiPathways Gene Set Enrichment Analysis (GSEA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_gsea_wp(
  dea_res,
  rank_by = "signed_log10p",
  aggr = "median",
  organism = "Homo sapiens",
  p_adj_method = "BH",
  p_cutoff = 0.05,
  min_gs_size = 10,
  max_gs_size = 500,
  exponent = 1,
  eps = 1e-10,
  seed = FALSE
)
```

Arguments

- `dea_res` Differential analysis result. Can be one of:
  - Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
  - A tibble with the following columns:
    - `protein`: Uniprot ID of proteins
    - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
    - `site`: The glycosylation site.
    - `p_val`: p-values, preferably adjusted p-values
    - `log2fc`: log2 of fold change
- `rank_by` Criteria for ranking proteins. One of the following:
  - "log2fc": log2 fold change with signs
  - "abs_log2fc": absolute log2 fold change
  - "log10p": negative log10 p-value
  - "signed_log10p" (default): log10 p-value with signs of log2 fold change
  - "log2fc_log10p": log2 fold change multiplied by negative log10 p-value
- `aggr` Aggregation method for combining multiple scores across different traits and sites for the same protein. One of "median", "mean", or "max". Defaults to "median".
- `organism` WikiPathways organism name. Passed to `organism` of `clusterProfiler::gseWP()`. Defaults to "Homo sapiens". Use `clusterProfiler::get_wp_organisms()` to see available organisms.
- `p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".
- `p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.
- `min_gs_size` Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.
- `max_gs_size` Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.
- `exponent` Weight of each step. Passed to `exponent` of `clusterProfiler::gseGO()`. Defaults to 1.
- `eps` Epsilon for calculating p-values. Passed to `eps` of `clusterProfiler::gseGO()`. Defaults to 1e-10.
- `seed` Logical indicating whether to set a random seed for reproducibility. Passed to `seed` of `clusterProfiler::gseGO()`. Defaults to `FALSE`.

Value

A clusterProfiler `gseaResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::ridgeplot()`.

How it ranks proteins

GSEA requires a ranked list of proteins as input. This function ranks proteins based on the median absolute log2 fold change across all traits and sites. This reflects the overall glycosylation dysregulation degree of each glycoprotein. You can use `rank_by` to specify other ranking criteria, such as p-values or signed log2 fold changes. You can also use `aggr` to specify how to aggregate multiple scores for the same protein across different traits and sites.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gsea_go(dea_res)  # or `enrich_gsea_xxx()` functions
```

See Also

`clusterProfiler::gseWP()`

---

enrich_ora_do — *Disease Ontology (DO) Over Representation Analysis*

Description

Performs Disease Ontology (DO) Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_do(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  ont = "HDO",
  organism = "hsa",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

dea_res
: Differential analysis result. Can be one of:
  - Result from glystats::gly_limma() (two groups), glystats::gly_ttest(), or glystats::gly_wilcox(), called on an glyexp::experiment() of "traitproteomics" type.
  - A tibble with the following columns:
    - protein: Uniprot ID of proteins
    - trait: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
    - site: The glycosylation site.
    - p_val: p-values, preferably adjusted p-values
    - log2fc: log2 of fold change

dea_p_cutoff
: P-value cutoff for statistical significance. Defaults to 0.05. For glystats result input, adjusted p-values are used.

dea_log2fc_cutoff
: Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, c(-1, 1) means "log2fc < -1 or log2fc > 1", and c(-Inf, 1) means "log2fc > 1". Defaults to c(-1, 1).

ont
: One of "HDO" (Human Disease Ontology), "MPO" (Mammalian Phenotype Ontology), or "VDO" (Vector Disease Ontology). Passed to ont of DOSE::enrichDO(). Defaults to "HDO".

organism
: "hsa" (Homo sapiens) or "mmu" (Mus musculus). Passed to organism of DOSE::enrichDO(). Defaults to "hsa".

universe
: Background genes Uniprot IDs, directly passed to universe of downstream enrichment function. If NULL (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use detected_universe() to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to pAdjustMethod of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to pvalueCutoff of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to qvalueCutoff of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to minGSSize of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to maxGSSize of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler enrichResult object. It can be readily converted to a tibble with tibble::as_tibble(), or visualized with clusterProfiler functions like clusterProfiler::dotplot().

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

DOSE::enrichDO()

---

enrich_ora_go   *GO Over Representation Analysis*

Description

Performs Gene Ontology (GO) Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_go(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  orgdb = "org.Hs.eg.db",
  ont = "MF",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from glystats::gly_limma() (two groups), glystats::gly_ttest(), or glystats::gly_wilcox(), called on an glyexp::experiment() of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins

- trait: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
- site: The glycosylation site.
- p_val: p-values, preferably adjusted p-values
- log2fc: log2 of fold change

dea_p_cutoff
: P-value cutoff for statistical significance. Defaults to 0.05. For glystats result input, adjusted p-values are used.

dea_log2fc_cutoff
: Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, c(-1, 1) means "log2fc < -1 or log2fc > 1", and c(-Inf, 1) means "log2fc > 1". Defaults to c(-1, 1).

orgdb
: An OrgDb object. Passed to OrgDb of downstream enrichment function.

ont
: Ontology type. Passed to ont of clusterProfiler::enrichGO(). "BP", "MF", "CC", or "ALL". Defaults to "MF".

universe
: Background genes Uniprot IDs, directly passed to universe of downstream enrichment function. If NULL (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use detected_universe() to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to pAdjustMethod of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to pvalueCutoff of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to qvalueCutoff of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to minGSSize of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to maxGSSize of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler enrichResult object. It can be readily converted to a tibble with tibble::as_tibble(), or visualized with clusterProfiler functions like clusterProfiler::dotplot().

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

clusterProfiler::enrichGO()

---

enrich_ora_kegg *KEGG Over Representation Analysis*

Description

Performs KEGG pathway Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_kegg(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "hsa",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

organism
: KEGG organism code. Passed to `organism` of `clusterProfiler::enrichKEGG()`. Defaults to "hsa" (Homo sapiens). Common codes: "hsa" (human), "mmu" (mouse), "rno" (rat).

universe
: Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `enrichResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

`clusterProfiler::enrichKEGG()`

enrich_ora_ncg *Network of Cancer Genes (NCG) Over Representation Analysis*

Description

Performs Network of Cancer Genes (NCG) Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_ncg(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff`
Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `enrichResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

`DOSE::enrichNCG()`

---

enrich_ora_reactome *Reactome Over Representation Analysis*

Description

Performs Reactome pathway Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_reactome(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "human",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

`dea_res` Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

`dea_p_cutoff` P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

`dea_log2fc_cutoff` Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

`organism` Reactome organism name. Passed to `organism` of `ReactomePA::enrichPathway()`. One of "human", "rat", "mouse", "celegans", "yeast", "zebrafish", "fly". Defaults to "human".

`universe` Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

`p_adj_method` P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

`p_cutoff` P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `enrichResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

`ReactomePA::enrichPathway()`

---

enrich_ora_wp *WikiPathways Over Representation Analysis*

Description

Performs WikiPathways Over-Representation Analysis (ORA) on glycoproteins with dysregulated glycosylation.

Usage

```
enrich_ora_wp(
  dea_res,
  dea_p_cutoff = 0.05,
  dea_log2fc_cutoff = c(-1, 1),
  organism = "Homo sapiens",
  universe = NULL,
  p_adj_method = "BH",
  p_cutoff = 0.05,
  q_cutoff = 0.2,
```

```
  min_gs_size = 10,
  max_gs_size = 500
)
```

Arguments

dea_res
: Differential analysis result. Can be one of:

- Result from `glystats::gly_limma()` (two groups), `glystats::gly_ttest()`, or `glystats::gly_wilcox()`, called on an `glyexp::experiment()` of "traitproteomics" type.
- A tibble with the following columns:
  - `protein`: Uniprot ID of proteins
  - `trait`: A glycosylation trait (e.g. "TFc" for proportion of core-fucosylated glycans)
  - `site`: The glycosylation site.
  - `p_val`: p-values, preferably adjusted p-values
  - `log2fc`: log2 of fold change

dea_p_cutoff
: P-value cutoff for statistical significance. Defaults to 0.05. For `glystats` result input, adjusted p-values are used.

dea_log2fc_cutoff
: Log2 fold change cutoff statistical significance. A length-2 numeric vector, being negative and positive boundaries, respectively. For example, `c(-1, 1)` means "log2fc < -1 or log2fc > 1", and `c(-Inf, 1)` means "log2fc > 1". Defaults to `c(-1, 1)`.

organism
: WikiPathways organism name. Passed to `organism` of `clusterProfiler::enrichWP()`. Defaults to "Homo sapiens". Use `clusterProfiler::get_wp_organisms()` to see available organisms.

universe
: Background genes Uniprot IDs, directly passed to `universe` of downstream enrichment function. If `NULL` (default), all genes in the data will be used. Another common pattern is to use all detected proteins as backgroud genes. You can use `detected_universe()` to help you.

p_adj_method
: P-value adjustment method. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". Passed to `pAdjustMethod` of downstream enrichment function. Defaults to "BH".

p_cutoff
: P-value cutoff to filter significant terms. Passed to `pvalueCutoff` of downstream enrichment function. Defaults to 0.05.

q_cutoff
: Q-value (FDR) cutoff to filter significant terms. Passed to `qvalueCutoff` of downstream enrichment function. Defaults to 0.2.

min_gs_size
: Minimal size of each gene set for analyzing. Gene sets with fewer genes than this threshold will be excluded. Passed to `minGSSize` of downstream enrichment function. Defaults to 10.

max_gs_size
: Maximum size of each gene set for analyzing. Gene sets with more genes than this threshold will be excluded. Passed to `maxGSSize` of downstream enrichment function. Defaults to 500.

Value

A clusterProfiler `enrichResult` object. It can be readily converted to a tibble with `tibble::as_tibble()`, or visualized with `clusterProfiler` functions like `clusterProfiler::dotplot()`.

Common usage pattern

A common pattern of using this function is:

```
# 1. Perform differential analysis with `glystats`.
dea_res <- gly_ttest(exp)

# 2. Use this function.
go_res <- enrich_gc_ora_go(dea_res)  # or other glyfun functions
```

See Also

`clusterProfiler::enrichWP()`

Index

clusterProfiler::compareCluster(), *5*, *7, 8*, *10*, *12*, *15*, *17*, *20*, *22*, *24*, *26*, *28*, *31*
clusterProfiler::dotplot(), *4*, *7*, *9*, *12*, *14*, *17*, *19*, *21*, *23*, *26*, *28*, *30*, *44*, *46*, *48*, *50*, *52*, *54*
clusterProfiler::enrichGO(), *6*, *21, 22*, *34*, *46, 47*
clusterProfiler::enrichKEGG(), *23, 24*, *48*
clusterProfiler::enrichWP(), *30, 31*, *53, 54*
clusterProfiler::get_wp_organisms(), *16*, *30*, *42*, *53*
clusterProfiler::gseGO(), *4*, *7–9*, *11*, *14*, *16*, *32*, *34–36*, *38*, *40*, *42*
clusterProfiler::gseKEGG(), *9, 10*, *36, 37*
clusterProfiler::gseWP(), *16, 17*, *42, 43*
clusterProfiler::ridgeplot(), *32*, *34*, *36*, *38*, *41*, *43*

detected_universe, 2
detected_universe(), *19*, *21*, *23*, *25*, *27*, *30*, *44*, *46*, *48, 49*, *51*, *53*
DOSE::enrichDO(), *19, 20*, *44, 45*
DOSE::enrichNCG(), *26*, *50*
DOSE::gseDO(), *4, 5*, *32, 33*
DOSE::gseNCG(), *12*, *39*

enrich_gc_gsea_do, 3
enrich_gc_gsea_go, 5
enrich_gc_gsea_kegg, 8
enrich_gc_gsea_ncg, 10
enrich_gc_gsea_reactome, 13
enrich_gc_gsea_wp, 15
enrich_gc_ora_do, 18
enrich_gc_ora_go, 20
enrich_gc_ora_kegg, 22
enrich_gc_ora_ncg, 24
enrich_gc_ora_reactome, 26
enrich_gc_ora_wp, 29
enrich_gsea_do, 31
enrich_gsea_go, 33
enrich_gsea_go(), *5*, *7*, *9, 10*, *12*, *14, 15*, *17*
enrich_gsea_kegg, 35
enrich_gsea_ncg, 37
enrich_gsea_reactome, 39
enrich_gsea_wp, 41
enrich_ora_do, 43
enrich_ora_go, 45
enrich_ora_go(), *19*, *21*, *24*, *26*, *28*, *30*
enrich_ora_kegg, 47
enrich_ora_ncg, 49
enrich_ora_reactome, 50
enrich_ora_wp, 52

glyexp::experiment(), *2, 3*, *6*, *8*, *11*, *13*, *16*, *18*, *20*, *23*, *25*, *27*, *29*, *31*, *33*, *35*, *38*, *40*, *42*, *44, 45*, *47*, *49*, *51*, *53*
glystats::gly_limma(), *3*, *6*, *8*, *11*, *13*, *16*, *18*, *20*, *23*, *25*, *27*, *29*, *31*, *33*, *35*, *38*, *40*, *42*, *44, 45*, *47*, *49*, *51*, *53*
glystats::gly_ttest(), *3*, *6*, *8*, *11*, *13*, *16*, *18*, *20*, *23*, *25*, *27*, *29*, *31*, *33*, *35*, *38*, *40*, *42*, *44, 45*, *47*, *49*, *51*, *53*
glystats::gly_wilcox(), *3*, *6*, *8*, *11*, *13*, *16*, *18*, *20*, *23*, *25*, *27*, *29*, *31*, *33*, *35*, *38*, *40*, *42*, *44, 45*, *47*, *49*, *51*, *53*

ReactomePA::enrichPathway(), *27, 28*, *51, 52*
ReactomePA::gsePathway(), *14, 15*, *40, 41*

tibble::as_tibble(), *4*, *7*, *9*, *12*, *14*, *17*, *19*, *21*, *23*, *26*, *28*, *30*, *32*, *34*, *36*, *38*, *41*, *43, 44*, *46*, *48*, *50*, *52*, *54*