

Package: isoformic (via r-universe)

October 5, 2026

Title Isoform-Level Biological Interpretation of Transcriptomic Data

Version 0.1.2.9019

Description Isoform-level biological interpretation of transcriptomic data.

License MIT + file LICENSE

URL <https://github.com/luciorq/isoformic>,
<https://luciorq.github.io/isoformic/>

BugReports <https://github.com/luciorq/isoformic/issues>

Depends R (>= 4.3)

Imports arrow, cli, DBI, dplyr, duckdb, fs, ggplot2, ggpubr, ggrepel,
glue, purrr (>= 1.1.0), readr, rlang, S7 (>= 0.2.0), stringr,
tibble, tidyr, utils, vroom, withr

Suggests AnnotationHub, apeglm, BSgenome.Hsapiens.NCBI.GRCh38, carrier
(>= 0.3.0), condathis, curl, DESeq2, fgsea, fishpond,
GenomeInfoDbData, GenomicFeatures, IRanges, knitr, mirai (>= 2.5.1),
org.Hs.eg.db, patchwork, plotgardener, quarto, SummarizedExperiment,
testthat (>= 3.0.0), TxDb.Hsapiens.UCSC.hg38.knownGene, txdbmaker, tximeta,
tximport, viridis

VignetteBuilder quarto

Config/Needs/dev devtools, pkgload, remotes, rcmdcheck, covr,
testthat, usethis, pak, knitr, quarto, pkgdown, roxygen2,
roxygen2md, lintr, styler

Config/Needs/website quarto

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev libx11-dev xz-utils

Repository <https://luciorq.r-universe.dev>

Date/Publication 2026-07-07 14:27:08 UTC

RemoteUrl <https://github.com/luciorq/isoformic>

RemoteRef HEAD

RemoteSha 9ed8b755fd209dc93ac3d15e1ced9062a3727f16

Contents

as_isoformic	2
create_context_data	3
download_reference	4
get_isoformic_cache	6
is_deg_sig	6
IsoformicExperiment	7
join_DEG_DET	9
make_tx_to_gene	10
plot_genomic_context	11
plot_log2FC	12
plot_tx_context	14
plot_tx_expr	14
prepare_annotation	15
prepare_annotation_db	15
prepare_exon_annotation	16
prepare_isoformic_annotation	16
prepare_profile_data	17
run_enrichment	19
salmon_index	21
salmon_quant	22
tx_type_palette	23

Index	24
--------------	-----------

as_isoformic	<i>Convert a SummarizedExperiment to an IsoformicExperiment Object</i>
--------------	--

Description

This function converts a SummarizedExperiment object to an IsoformicExperiment object. It extracts the assays, row data, column data, and metadata from the input object and uses them to create a new IsoformicExperiment object.

Usage

```
as_isoformic(se, annot_path, annot_type = c("gff", "annot_db"))
```

Arguments

se	A SummarizedExperiment object to be converted.
annot_path	Path to the annotation file. This can be a GFF file or the path pre-built annotation database created with [prepare_isoformic_annotation()].
annot_type	Type of the annotation file provided. Options are "gff" for GFF files and "annot_db" for pre-built annotation databases.

create_context_data	<i>Create ContextData Object</i>
---------------------	----------------------------------

Description

This function instantiates a ContextData object containing information about the genomic context of transcripts. It requires a GFF file for gene annotation and constructs a TxDb object from it. The function also prepares an annotation table and updates the transcript names in the TxDb object. The ContextData object can then be used in conjunction with IsoformicExperiment for transcriptomic analyses.

Usage

```
create_context_data(
  gff_file,
  ...,
  organism,
  orgdb_package,
  bsgenome_package,
  tx_type_palette = NULL
)
```

Arguments

gff_file	Character string specifying the path to a GFF file containing the gene annotation.
...	These dots are for future extensions and must be empty.
organism	Character string specifying the organism name (e.g., "Homo sapiens").
orgdb_package	Character string specifying the name of the organism database package (e.g., "org.Hs.eg.db").
bsgenome_package	Character string specifying the name of the BSgenome package (e.g., "BSgenome.Hsapiens.UCSC.hg38").
tx_type_palette	Named character vector specifying the color palette for transcript types.

download_reference	<i>Download Reference Files from GENCODE</i>
--------------------	--

Description

Downloads reference annotation files from the GENCODE database for human or mouse genomes. Supports downloading GFF, GTF, transcriptome FASTA, and genome FASTA files. The function handles directory creation and checks for existing files to avoid redundant downloads.

Usage

```
download_reference(
  version = "49",
  reference = "gencode",
  organism = c("human", "mouse"),
  file_type = c("gff", "gtf", "fasta", "genome_fasta"),
  output_path = ":cache:",
  timeout_limit = 3600,
  method = "auto"
)
```

Arguments

version	A character string specifying the GENCODE release version. For mouse references, include the letter 'M' in the version string (e.g., "M38"). Default is "49".
reference	A character string specifying the source of the reference file. Currently, only "gencode" is supported. Default is "gencode".
organism	A character string specifying the organism. Valid options are "human" or "mouse".
file_type	A character string specifying the type of file to download. Valid options are "gff", "gtf", "fasta" or "genome_fasta". Defaults to "gff". Note: "fasta" refers to the transcriptome FASTA file. "genome_fasta" refers to the whole genome sequence FASTA file.
output_path	A character string specifying the directory where the downloaded file will be saved. Defaults to ":cache:". Cache path is defined by the <code>[isoformic::get_isoformic_cache()]</code> function.
timeout_limit	A numeric value specifying the maximum time in seconds for the download to complete. This argument takes precedence over <code>options("timeout")</code> . Defaults to 3600 seconds (1 hour).
method	A character string specifying the method used by <code>utils::download.file()</code> . Defaults to "auto".

Details

The function constructs the appropriate download URL based on the specified organism, version, and file type, and downloads the file to the specified output path, being a user cache by default. If the file already exists in the output directory, the function will not download it again and will return the existing file path. The function requires an internet connection and handles timeout settings to prevent download interruptions.

Value

A character string with the full path to the downloaded file.

Note

Currently, only "gencode" reference files are supported. The "mane" reference is not implemented yet.

Examples

```
## Not run:
# Download human GFF file for GENCODE release 49
gff_file <- download_reference(
  version = "49",
  organism = "human",
  file_type = "gff",
  output_path = ":cache:"
)

# Download mouse GFF file for GENCODE release M38
gff_file_mouse <- download_reference(
  version = "M38",
  organism = "mouse",
  file_type = "gff",
  output_path = ":cache:"
)

# Download human transcriptome FASTA file for GENCODE release 49
fasta_file <- download_reference(
  version = "49",
  organism = "human",
  file_type = "fasta",
  output_path = ":cache:"
)

## End(Not run)
```

get_isoformic_cache	<i>Retrieve System-Dependent Cache Path for Isoformic</i>
---------------------	---

Description

Determines the appropriate user cache directory for the isoformic package based on the operating system. On macOS, it avoids using paths with spaces and follows the XDG base directory specification.

Usage

```
get_isoformic_cache(..., ext = NULL)
```

Arguments

...	Additional path components to append to the cache directory.
ext	An optional file extension (e.g., "rds", "csv") to append to the final path.

Details

This function uses the `[tools::R_user_dir()]` function to determine the user cache directory.

Value

A path character string representing the path to the user cache directory for the isoformic package.

is_deg_sig	<i>Annotate Transcripts with Differential Gene Expression Significance</i>
------------	--

Description

Adds a column to a transcript-level differential expression table indicating whether each transcript originates from a gene that is significantly differentially expressed.

Usage

```
is_deg_sig(deg_sig_vector, det_table)
```

Arguments

deg_sig_vector	A character vector containing the names of transcripts from significantly differentially expressed genes.
det_table	A data.frame or tibble containing transcript-level differential expression results, including a transcript_name column.

Value

A tibble with an additional column `DEG_sig` indicating whether the transcript is from a significantly differentially expressed gene ("YES" or "NO"). Row order matches the input `det_table`.

Examples

```
# Sample data
significant_transcripts <- c("transcript1", "transcript3")
det_table <- data.frame(
  transcript_name = c("transcript1", "transcript2", "transcript3", "transcript4"),
  log2FC = c(2.5, -1.2, 0.8, -0.5),
  pvalue = c(0.01, 0.2, 0.03, 0.6)
)

# Annotate transcripts with DEG significance
det_table_annotated <- is_deg_sig(
  deg_sig_vector = significant_transcripts,
  det_table = det_table
)

# View the result
print(det_table_annotated)
```

IsoformicExperiment	<i>IsoformicExperiment Class</i>
---------------------	----------------------------------

Description

The `IsoformicExperiment` class encapsulates the core data structure for transcriptomic analyses in the `isoformic` package. It holds the path to the dataset, sample metadata, and provides access to transcript, gene, and exon annotations through properties. The preferred way to construct an object of this class is through the `IsoformicExperiment()` constructor.

Usage

```
IsoformicExperiment(
  experiment_name = NA_character_,
  data_path = NULL,
  annot_path = NULL,
  assay = NULL,
  col_data = NULL,
  annot_metadata = NULL,
  dea = NULL,
  gsea = NULL,
  tx_type_palette = character(0)
)
```

```

col_data(self, ...)

annot_data_transcripts(self, ...)

annot_data_genes(self, ...)

annot_data_exons(self, ...)

annot_data(self, ...)

annot_row_names(self, ...)

col_names(self, ...)

row_names(self, ...)

tx_to_gene(self, ...)

row_data(self, ...)

tx_annot(self, ...)

de_tx(self, ...)

de_gene(self, ...)

```

Arguments

experiment_name	Character string specifying the name of the experiment. This name is used for caching the assays experiment. If a name is not provided a random identifier is used.
data_path	Character string specifying the path to the data directory.
annot_path	Character string specifying the path to the annotation database directory.
assay	A list of matrices or data frames containing assay data, with transcript IDs as row names and sample IDs as column names. Each element of the list represents a different assay (e.g., TPM, counts).
col_data	A data frame containing sample metadata. First column must be <code>sample_id</code> matching the column names of the assays.
annot_metadata	A list containing metadata about the annotation, such as source, version, and date.
dea	A list containing differential expression analysis results for transcripts and genes.
gsea	A list containing gene set enrichment analysis results.
tx_type_palette	A named character vector specifying the color palette for different transcript types.

self	An IsoformicExperiment object.
...	Additional arguments passed to methods.
annot_data_transcripts	A property that retrieves transcript annotation data.
annot_data_genes	A property that retrieves gene annotation data.
annot_data_exons	A property that retrieves exon annotation data.
annot_data	A property that aggregates transcript, gene, and exon annotation data.

join_DEG_DET	<i>Merge Gene and Transcript Level Differential Expression Tables</i>
--------------	---

Description

Combines gene-level and transcript-level differential expression results into a single table, annotates the combined data with significance labels based on specified cutoffs, and filters transcripts based on their types.

Usage

```
join_DEG_DET(deg_table, det_table, logfc_cut, pval_cut)
```

Arguments

deg_table	A data.frame or tibble containing gene-level differential expression results, including gene_id, gene_name, log2FC, and pvalue columns.
det_table	A data.frame or tibble containing transcript-level differential expression results, including transcript_id, transcript_name, transcript_type, log2FC, and pvalue columns.
logfc_cut	A numeric value specifying the absolute log2 fold-change cutoff for significance.
pval_cut	A numeric value specifying the p-value cutoff for significance.

Value

A tibble combining gene and transcript differential expression results, with additional columns:

- id: gene or transcript ID.
- name: gene or transcript name.
- transcript_type: type of transcript or "gene" for gene-level entries.
- abs_log2FC: absolute value of log2 fold-change.
- significance: "sig" if significant based on cutoffs, "not_sig" otherwise.

Examples

```
# Sample gene-level data
deg_table <- data.frame(
  gene_id = c("gene1", "gene2"),
  gene_name = c("GeneA", "GeneB"),
  log2FC = c(1.5, -2.0),
  pvalue = c(0.01, 0.04)
)

# Sample transcript-level data
det_table <- data.frame(
  transcript_id = c("tx1", "tx2", "tx3"),
  transcript_name = c("Transcript1", "Transcript2", "Transcript3"),
  transcript_type = c("protein_coding", "lncRNA", "processed_transcript"),
  log2FC = c(1.2, -1.8, 0.5),
  pvalue = c(0.02, 0.03, 0.2)
)

# Merge and annotate differential expression results
deg_det_table <- join_DEG_DET(
  deg_table = deg_table,
  det_table = det_table,
  logfc_cut = 1,
  pval_cut = 0.05
)

# View the result
print(deg_det_table)
```

make_tx_to_gene

*Create Transcript-to-Gene Relationship Table***Description**

Extracts a transcript-to-gene mapping table from GENCODE annotation files, such as the transcriptome FASTA file. Currently, only FASTA files are supported.

Usage

```
make_tx_to_gene(file_path, file_type = c("fasta", "gff", "gtf"))
```

Arguments

file_path	A character string specifying the path to the reference file (e.g., GENCODE FASTA file).
file_type	A character string specifying the type of the reference file. Currently, only "fasta" is supported. Default is "fasta".

Details

The function reads the headers of the FASTA file and extracts relevant information to create a mapping table. For GTF or GFF3 files, support is not yet implemented.

Value

A tibble containing the transcript-to-gene mapping information, including transcript IDs, gene IDs, transcript names, gene names, and transcript types.

Examples

```
## Not run:
# Assuming you have downloaded the GENCODE transcriptome FASTA file:
fasta_file <- download_reference(
  version = "43",
  organism = "human",
  file_type = "fasta",
  output_path = "data-raw"
)

# Create the transcript-to-gene mapping table
tx_to_gene <- make_tx_to_gene(file_path = fasta_file, file_type = "fasta")

# View the first few rows
utils::head(tx_to_gene)

## End(Not run)
```

plot_genomic_context *Plot Transcripts Genomic Context*

Description

Generate a genomic context plot for a specified gene, displaying its transcripts along with their types and annotations. The function utilizes the plotgardener package to create a detailed visualization of the genomic context, including an ideogram, chromosome highlight, and transcript structures. It requires a ContextData object that contains the necessary genomic information and annotations. The plot can be customized with various parameters such as offsets, label limits, and ideogram references.

Usage

```
plot_genomic_context(
  gene_name,
  context_data,
  limit_label = TRUE,
  show_guides = FALSE,
```

```

    y_offset = 0,
    height_offset = 0,
    downstream_offset = 0,
    upstream_offset = 0,
    ideogram_reference = c("hg38", "hg19", "mm11", "mm10", "none")
  )

```

Arguments

gene_name	Character string specifying the name of the gene to plot.
context_data	A ContextData object containing genomic context information.
limit_label	Logical indicating whether to limit the length of transcript labels to avoid overlap (default is TRUE).
show_guides	Logical indicating whether to show guide lines on the plot (default is FALSE).
y_offset	Numeric value to adjust the vertical position of the plot (default is 0).
height_offset	Numeric value to adjust the height of the plot (default is 0).
downstream_offset	Numeric value to extend the downstream region beyond the gene's end position (default is 0).
upstream_offset	Numeric value to extend the upstream region beyond the gene's start position (default is 0).
ideogram_reference	Character string specifying the reference genome for the ideogram. Options include "hg38", "hg19", "mm11", "mm10", or "none" (default is "hg38").

Value

A plotgardener object representing the genomic context plot.

plot_log2FC

Plot Log2 Fold-Change Results for Transcripts of Selected Genes

Description

Creates a bar plot of log2 fold-change values for transcripts of a selected gene, differentiating transcript types and significance levels.

Usage

```

plot_log2FC(
  de_data,
  feature,
  feature_column = "gene_name",
  color_palette = NULL
)

plot_log2fc(self, ...)

```

Arguments

de_data	A data.frame or tibble containing combined gene and transcript differential expression results. Should contain columns for log2 fold-change, transcript type, significance, and feature symbols.
feature	A character string specifying the gene name to plot.
feature_column	A character string specifying the column name in de_data that contains gene names. Default is "gene_name".
color_palette	A named character vector specifying colors for different transcript types. If NULL, a default palette will be used.
self	Input object, either a data.frame or an IsoformicExperiment.
...	Additional arguments passed to the method.

Details

The function filters the input table for the selected gene and creates a bar plot of log2 fold-change values. If all transcripts are significant, it plots without adjusting alpha transparency; otherwise, it adjusts alpha based on significance. The function uses predefined colors for transcript types, which can be overridden by providing custom_colors.

Value

A ggplot2 object representing the bar plot.

Examples

```
# Sample data
de_table_long <- data.frame(
  feature_name = c("Transcript1", "Transcript2", "Transcript3", "GeneA"),
  feature_id = c("TX1", "TX2", "TX3", "GENEA"),
  gene_name = c("GeneA", "GeneA", "GeneA", "GeneA"),
  log2FC = c(1.5, -0.5, -2.0, 0.8),
  feature_type = c("protein_coding", "lncRNA", "retained_intron", "gene"),
  is_de = c("yes", "no", "yes", "yes")
)

# Plot log2 fold-change for the selected gene
plot_obj <- plot_log2FC(
  de_data = de_table_long,
  feature = "GeneA",
  feature_column = "gene_name"
)

# Display the plot
print(plot_obj)
```

plot_tx_context	<i>Plot Transcript Genomic Context</i>
-----------------	--

Description

This function plots the genomic context of all transcripts of given genes.

Usage

```
plot_tx_context(exon_table, custom_colors = NULL)
```

Arguments

exon_table	a tibble with exon information. Must contain columns tx_id, exon_left, and exon_right.
custom_colors	a vector of colors to use for each transcript. If not provided, the function will use the default colors. Actually, this argument is *NOT implemented yet.

plot_tx_expr	<i>Plot Transcript per gene expression</i>
--------------	--

Description

Plot Transcript per gene expression

Usage

```
plot_tx_expr(genes_to_plot, profile_data)
```

Arguments

genes_to_plot	a character vector with gene names
profile_data	tibble output from prepare_profile_data

Value

a ggplot object

prepare_annotation	<i>Prepare Annotation</i>
--------------------	---------------------------

Description

Prepare annotation to be imported as rowRanges and rowData for both Genes, Transcripts and Exons based Position Annotation Table. From a GTF or GFF3 annotation file.

Usage

```
prepare_annotation(file_path, file_type = c("gtf", "gff"))
```

Arguments

file_path	Path to annotation file.
file_type	Character indicating the type of file to download. One of "gtf" or "gff". Defaults to "gtf".

prepare_annotation_db	<i>Write Parquet File from GFF</i>
-----------------------	------------------------------------

Description

This function reads a GFF file and writes its contents to a Parquet file using DuckDB.

Usage

```
prepare_annotation_db(input_path, output_path = NULL, file_type = c("gff"))
```

Arguments

input_path	Character string specifying the path to the input GFF file.
output_path	Character string specifying the path to the output Parquet file. If NULL or an empty string, a temporary file will be created.
file_type	Character string specifying the type of the input file. Currently, only "gff" is supported (default is "gff").

Value

Invisible path to the created Parquet file.

```
prepare_exon_annotation
```

Prepare Exon based Position Annotation Table

Description

Prepare Exon based Position Annotation Table

Usage

```
prepare_exon_annotation(gene_name, file_path, file_type = c("gff", "gtf"))
```

Arguments

gene_name	String or vector of gene names to extract.
file_path	Path to annotation file.
file_type	A character string specifying the type of file to download. Valid options are "gff", "gtf", "fasta" or "genome_fasta". Defaults to "gff". Note: "fasta" refers to the transcriptome FASTA file. "genome_fasta" refers to the whole genome sequence FASTA file.

```
prepare_isoformic_annotation
```

Write Feature Annotation to Parquet Files

Description

This function reads an annotation file and parse feature annotation to Parquet files each level of required feature (i.e. gene, transcript, and exon).

Usage

```
prepare_isoformic_annotation(
  input_path,
  output_path = NULL,
  file_type = c("gff")
)
```

Arguments

input_path	Character string specifying the path to the input GFF file.
output_path	Character string specifying the path to the output directory where Parquet files are written. If NULL or an empty string, the cache directory will be used.
file_type	Character string specifying the type of the input file. Currently, only "gff" is supported (default is "gff").

Value

Invisible path to the created Parquet file.

prepare_profile_data *Prepare Data for Gene and Transcript Expression Profile Plot*

Description

This function processes gene and transcript-level expression data, along with differential expression results, to prepare a tidy data frame suitable for plotting expression profiles across different sample groups.

Usage

```
prepare_profile_data(
  txi_gene = NULL,
  txi_transcript,
  sample_metadata,
  tx_to_gene,
  de_result_gene,
  de_result_transcript,
  var,
  var_levels,
  gene_col = "gene_name",
  tx_col = "transcript_name",
  pvalue_cutoff = 0.05,
  lfc_cutoff = 1,
  use_fdr = TRUE
)
```

Arguments

txi_gene	A tibble or tximport output containing gene-level expression abundances. If NULL, gene-level abundances will be summarized from txi_transcript. Default is NULL.
txi_transcript	A tibble or tximport output containing transcript-level expression abundances.
sample_metadata	A data.frame or tibble containing sample metadata. The first column should contain sample names matching the column names in txi_gene and txi_transcript.
tx_to_gene	A data.frame or tibble containing transcript-to-gene mapping information. Must include columns specified by gene_col and tx_col.
de_result_gene	A data.frame or tibble containing differential expression results at the gene level. Must include gene_name, log2FC, and qvalue columns.
de_result_transcript	A data.frame or tibble containing differential expression results at the transcript level. Must include transcript_name, log2FC, and qvalue columns.

<code>var</code>	A string specifying the column name in <code>sample_metadata</code> that indicates the grouping variable (e.g., treatment, condition).
<code>var_levels</code>	A character vector specifying the levels of <code>var</code> to include in the contrasts.
<code>gene_col</code>	A string specifying the column name in <code>tx_to_gene</code> that contains gene names. Default is "gene_name".
<code>tx_col</code>	A string specifying the column name in <code>tx_to_gene</code> that contains transcript names. Default is "transcript_name".
<code>pvalue_cutoff</code>	A numeric value specifying the p-value cutoff for determining significant differential expression. Default is 0.05.
<code>lfc_cutoff</code>	A numeric value specifying the log2 fold-change cutoff for determining significant differential expression. Default is 1.
<code>use_fdr</code>	A logical value indicating whether to use the false discovery rate (qvalue) instead of p-value for significance cutoff. Default is TRUE.

Details

The function combines gene and transcript expression data with differential expression results to generate a tidy data frame. It filters significant genes and transcripts based on specified cutoffs and prepares the data for plotting expression profiles across specified sample groups.

Value

A tibble containing processed expression data and differential expression flags, ready for plotting.

Examples

```
## Not run:
# Assuming txi_gene, txi_transcript, sample_metadata, tx_to_gene, de_result_gene,
# and de_result_transcript are pre-loaded data frames:

# Prepare data for plotting
if (FALSE) {
  expr_df <- prepare_profile_data(
    txi_gene = txi_gene,
    txi_transcript = txi_transcript,
    sample_metadata = sample_metadata,
    tx_to_gene = tx_to_gene,
    de_result_gene = de_result_gene,
    de_result_transcript = de_result_transcript,
    var = "condition",
    var_levels = c("control", "treatment"),
    gene_col = "gene_name",
    tx_col = "transcript_name",
    pvalue_cutoff = 0.05,
    lfc_cutoff = 1,
    use_fdr = TRUE
  )

  # View the prepared data
```

```

utils::head(expr_df)

# Plotting example (assuming ggplot2 is installed)
library(ggplot2)
ggplot(expr_df, aes(x = condition, y = mean_TPM, fill = DE)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  facet_wrap(~ parent_gene + transcript_type)
}

## End(Not run)

```

run_enrichment

*Run Gene Set Enrichment Analysis for Different Transcript Types***Description**

Performs gene set enrichment analysis (GSEA) on differential expression results for various transcript types, using the fgsea package. The function iterates over specified transcript types, filters the data accordingly, and runs GSEA for each type.

Usage

```

run_enrichment(
  det_df,
  genesets_list,
  tx_to_gene,
  pval_cutoff = 0.05,
  lfc_cutoff = 1
)

```

Arguments

det_df	A data.frame or tibble containing transcript-level differential expression results, including transcript_type, log2FC, and gene_name columns.
genesets_list	A list of gene sets to be used in the enrichment analysis.
tx_to_gene	A data.frame or tibble mapping transcript names to gene names, including transcript_name and gene_name columns.
pval_cutoff	A numeric value specifying the p-value cutoff for the enrichment results. Default is 0.05.
lfc_cutoff	A numeric value specifying the log2 fold-change cutoff for filtering transcripts. Default is 1.

Details

The function defines a list of transcript types and their corresponding labels. It then filters the input differential expression data for each transcript type, ranks the genes by log2 fold-change, and performs GSEA using the fgsea package.

Value

A tibble containing the enrichment analysis results for each transcript type, including pathway names, p-values, adjusted p-values, and the transcript type (experiment).

Examples

```
# Sample differential expression data
det_df <- data.frame(
  gene_name = c(
    "GeneA", "GeneB", "GeneC", "GeneD",
    "GeneA", "GeneA", "GeneB", "GeneC",
    "GeneD", "GeneE", "GeneB", "GeneA"
  ),
  transcript_type = c(
    "protein_coding", "retained_intron",
    "protein_coding_CDS_not_defined", "processed_transcript",
    "protein_coding", "protein_coding",
    "retained_intron", "protein_coding_CDS_not_defined",
    "processed_transcript", "nonsense_mediated_decay",
    "protein_coding", "retained_intron"
  ),
  transcript_name = c(
    "Transcript1", "Transcript2",
    "Transcript3", "Transcript4",
    "Transcript5", "Transcript6",
    "Transcript7", "Transcript8",
    "Transcript9", "Transcript10",
    "Transcript11", "Transcript12"
  ),
  log2FC = c(
    1.5, -2.0, 0.8, -1.2, 2.3, -0.5,
    1.0, -1.5, 0.3, -2.5, 1.8, -0.7
  )
)

# Sample gene sets
genesets_list <- list(
  Pathway1 = c("GeneA", "GeneC", "GeneF"),
  Pathway2 = c("GeneB", "GeneD", "GeneE", "GeneX")
)

# Sample transcript to gene mapping
tx_to_gene <- data.frame(
  transcript_name = det_df$transcript_name,
  gene_name = det_df$gene_name
)

# Run enrichment analysis
fgsea_results_df <- run_enrichment(
  det_df = det_df,
  genesets_list = genesets_list,
  tx_to_gene = tx_to_gene,
```

```
pval_cutoff = 0.05,  
lfc_cutoff = 1  
)  
  
# View the results  
print(fgsea_results_df)
```

salmon_index*Build Salmon Index*

Description

Create a Salmon index from a reference transcriptome FASTA file.

Usage

```
salmon_index(  
  fasta_path,  
  index_path = "salmon_index",  
  kmer_len = 31,  
  num_threads = 2,  
  env_name = "salmon-env",  
  is_gencode = FALSE,  
  decoy_fasta = NULL,  
  clip_poly_a = TRUE  
)
```

Arguments

<code>fasta_path</code>	Path to the reference transcriptome FASTA file.
<code>index_path</code>	Directory path to save the Salmon index (default is "salmon_index").
<code>kmer_len</code>	K-mer length for the index (default is 31).
<code>num_threads</code>	Number of threads to use (default is 2).
<code>env_name</code>	Name of the conda environment with Salmon installed (default is "salmon-env").
<code>is_gencode</code>	Logical indicating if the FASTA is from GENCODE (default is FALSE).
<code>decoy_fasta</code>	Optional path to a FASTA file containing decoy sequences (default is NULL).
<code>clip_poly_a</code>	Logical indicating whether to clip poly-A tails (default is TRUE).

Value

processx style output list.

salmon_quant	<i>Run Salmon Quantification</i>
--------------	----------------------------------

Description

Perform transcript quantification using Salmon's selective-alignment-based mode from raw RNA-seq reads.

Usage

```
salmon_quant(  
  input_r1,  
  input_r2 = NULL,  
  index_path = "salmon_index",  
  output_dir = "quant_output",  
  num_threads = 8,  
  num_gibbs = 100,  
  min_score_fraction = "0.65",  
  env_name = "salmon-env"  
)
```

Arguments

input_r1	Path to the FASTQ file for read 1 (or single-end reads).
input_r2	Optional path to the FASTQ file for read 2 (paired-end reads).
index_path	Path to the Salmon index directory (default is "salmon_index").
output_dir	Directory to save the quantification output (default is "quant_output").
num_threads	Number of threads to use (default is 8).
num_gibbs	Number of Gibbs samples for uncertainty estimation (default is 100).
min_score_fraction	Minimum score fraction for alignments (default is "0.65").
env_name	Name of the conda environment with Salmon installed (default is "salmon-env").

Value

processx style output list.

tx_type_palette	<i>Transcript Type Color Palette</i>
-----------------	--------------------------------------

Description

Default color palette for transcript types.

Usage

```
tx_type_palette()
```

Index

`annot_data (IsoformicExperiment), 7`
`annot_data_exons (IsoformicExperiment), 7`
`annot_data_genes (IsoformicExperiment), 7`
`annot_data_transcripts (IsoformicExperiment), 7`
`annot_row_names (IsoformicExperiment), 7`
`as_isoformic, 2`

`col_data (IsoformicExperiment), 7`
`col_names (IsoformicExperiment), 7`
`create_context_data, 3`

`de_gene (IsoformicExperiment), 7`
`de_tx (IsoformicExperiment), 7`
`download_reference, 4`

`get_isoformic_cache, 6`

`is_deg_sig, 6`
`IsoformicExperiment, 7`
`IsoformicExperiment(), 7`

`join_DEG_DET, 9`

`make_tx_to_gene, 10`

`plot_genomic_context, 11`
`plot_log2FC, 12`
`plot_log2fc (plot_log2FC), 12`
`plot_tx_context, 14`
`plot_tx_expr, 14`
`prepare_annotation, 15`
`prepare_annotation_db, 15`
`prepare_exon_annotation, 16`
`prepare_isoformic_annotation, 16`
`prepare_profile_data, 17`

`row_data (IsoformicExperiment), 7`
`row_names (IsoformicExperiment), 7`

`run_enrichment, 19`

`salmon_index, 21`
`salmon_quant, 22`

`tx_annot (IsoformicExperiment), 7`
`tx_to_gene (IsoformicExperiment), 7`
`tx_type_palette, 23`