Package: REFT (via r-universe)

May 19, 2026

Type Package

Title Root Exudate Feature Toolkit

Version 0.1.4

Description Provides tools for molecule-oriented and reaction-centred analysis of root exudate datasets. It supports structural matching based on 'PubChem', calculation of molecular descriptors, and inference of candidate microbe-associated metabolic reactions using Kyoto Encyclopedia of Genes and Genomes ('KEGG') identifiers and Enzyme Commission ('EC') numbers. For background on these databases, see Kanehisa et al. (2023) <doi:10.1093/nar/gkac963> and Kim et al. (2023) <doi:10.1093/nar/gkac956>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports readxl, dplyr, purrr, stringr, tibble, writexl, webchem, rlang

Suggests rcdk, rcdklibs

Depends R (>= 4.1.0)

URL https://github.com/gaoguozhen1/REFT

BugReports https://github.com/gaoguozhen1/REFT/issues

NeedsCompilation no

Author Guozhen Gao [aut, cre]

Maintainer Guozhen Gao <gaoguozhen889@gmail.com>

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

Date/Publication 2026-05-19 09:30:24 UTC

RemoteUrl https://github.com/cran/REFT

RemoteRef HEAD

RemoteSha 1df3cfbde4e49ee9ccdf2efe35bd75f124d8c1f2

Contents

REFT 2
reft_calc_descriptors 2
reft_kegg_microbe_run 3
reft_match_smiles 4
reft_run 5
reft_run_simple 6

Index **8**

`REFT` *REFT: Root Exudate Feature Toolkit*

Description

REFT is an R package for batch `PubChem` matching and molecular descriptor calculation from root exudate or metabolomics annotation tables.

`reft_calc_descriptors` *Calculate six molecular descriptors*

Description

Calculate six descriptors from a character vector of SMILES using `rcdk`.

Usage

```
reft_calc_descriptors(smiles)
```

Arguments

`smiles`	A character vector of SMILES.

Value

A tibble with six molecular descriptors.

Examples

```
if (requireNamespace("rcdk", quietly = TRUE)) {
  reft_calc_descriptors("OC(=O)CCC(=O)O")
}
```

`reft_kegg_microbe_run` *Run KEGG microbe-EC-reaction search workflow*

Description

Import a microbial EC annotation table, normalize EC identifiers, extract species names from taxonomy strings, query KEGG for EC-linked reactions, and append reactants, products, and compound formulae. By default, no files are written; set `output_dir` to explicitly request Excel outputs.

Usage

```
reft_kegg_microbe_run(
  input_file,
  ec_col = "EC_Number",
  taxonomy_col = "Taxonomy",
  output_dir = NULL,
  output_file = "microbe_ec_kegg_reactions.xlsx",
  sleep_sec = 0.35,
  verbose = TRUE
)
```

Arguments

`input_file`	Path to input annotation table.
`ec_col`	Column containing EC numbers. Default is `"EC_Number"`.
`taxonomy_col`	Column containing taxonomy strings. Default is `"Taxonomy"`.
`output_dir`	Output directory. If `NULL` (default), no files are written.
`output_file`	Output Excel filename. Default is `"microbe_ec_kegg_reactions.xlsx"`.
`sleep_sec`	Delay between KEGG requests in seconds. Default is `0.35`.
`verbose`	Whether to print progress. Default is `TRUE`.

Value

A named list containing:

results Full result table with EC, microbe, reaction, compounds, and formulae.

ec_to_reaction EC-to-reaction mapping table.

reaction_details Reaction detail table.

compound_table Compound formula table.

Examples

```
toy <- data.frame(
  EC_Number = "1.1.1.1",
  Taxonomy = "k__Bacteria;p__Proteobacteria;g__Escherichia;s__Escherichia_coli"
)
input_file <- tempfile(fileext = ".csv")
utils::write.csv(toy, input_file, row.names = FALSE)
res <- try(
  reft_kegg_microbe_run(input_file, output_dir = tempdir(), sleep_sec = 0,
                        verbose = FALSE),
  silent = TRUE
)
if (!inherits(res, "try-error")) head(res$results)
```

---

reft_match_smiles *Match SMILES from PubChem*

Description

Batch match SMILES using Name, Other Name, KEGG ID, and HMDB ID in order.

Usage

```
reft_match_smiles(
  data,
  name_col = "Name",
  other_col = "Other_name(Kegg_name)",
  hmdb_col = "HMDB_ID",
  kegg_col = "Kegg_ID"
)
```

Arguments

data	A data frame containing query columns.
name_col	Compound name column.
other_col	Alternative name column.
hmdb_col	HMDB ID column.
kegg_col	KEGG ID column.

Value

A data frame with matching log and SMILES.

Examples

```
dat <- data.frame(
  Name = "Glutarate",
  `Other_name(Kegg_name)` = NA,
  HMDB_ID = NA,
  Kegg_ID = NA,
  check.names = FALSE
)
res <- try(reft_match_smiles(dat), silent = TRUE)
if (!inherits(res, "try-error")) head(res)
```

`reft_run` *Run the full REFT workflow*

Description

Import an Excel table, clean query fields, match SMILES from PubChem, calculate six molecular descriptors, and optionally write Excel outputs. By default, no files are written; set `output_dir` to explicitly request Excel outputs.

Usage

```
reft_run(
  input_file,
  name_col = "Name",
  other_col = "Other_name(Kegg_name)",
  hmdb_col = "HMDB_ID",
  kegg_col = "Kegg_ID",
  output_dir = NULL,
  output_desc_file = "metabolites_6_descriptors.xlsx",
  output_unmatched_file = "unmatched_smiles.xlsx",
  output_log_file = "pubchem_match_log.xlsx",
  verbose = TRUE
)
```

Arguments

`input_file`	Path to the input Excel file.
`name_col`	Column name for compound name. Default is `"Name"`.
`other_col`	Column name for alternative name. Default is `"Other_name(Kegg_name)"`.
`hmdb_col`	Column name for HMDB identifier. Default is `"HMDB_ID"`.
`kegg_col`	Column name for KEGG identifier. Default is `"Kegg_ID"`.
`output_dir`	Output directory. If `NULL` (default), no files are written.
`output_desc_file`	Final descriptor Excel filename.

`output_unmatched_file`
: Unmatched records Excel filename.

`output_log_file`
: PubChem match log Excel filename.

`verbose`
: Whether to print progress. Default is `TRUE`.

Value

A named list with three data frames:

descriptors Final annotation table with SMILES and six descriptors.

unmatched Rows that could not be matched to SMILES.

match_log Unique-query matching log from PubChem.

Examples

```
toy <- data.frame(
  Name = "Glutarate",
  `Other_name(Kegg_name)` = NA,
  HMDB_ID = NA,
  Kegg_ID = NA,
  check.names = FALSE
)
if (requireNamespace("rcdk", quietly = TRUE)) {
  input_file <- tempfile(fileext = ".xlsx")
  writexl::write_xlsx(toy, input_file)
  res <- try(reft_run(input_file, output_dir = tempdir(), verbose = FALSE),
             silent = TRUE)
  if (!inherits(res, "try-error")) head(res$descriptors)
}
```

---

reft_run_simple *Run REFT with default column names*

Description

A simplified wrapper around `reft_run()` for the common case where the input file already uses the default column names. By default, no files are written; set `output_dir` to explicitly request Excel outputs.

Usage

```
reft_run_simple(input_file, output_dir = NULL, verbose = TRUE)
```

Arguments

`input_file`
: Path to the input Excel file.

`output_dir`
: Output directory. If `NULL` (default), no files are written.

`verbose`
: Whether to print progress. Default is `TRUE`.

Value

Same as `reft_run()`.

Examples

```
toy <- data.frame(
  Name = "Glutarate",
  `Other_name(Kegg_name)` = NA,
  HMDB_ID = NA,
  Kegg_ID = NA,
  check.names = FALSE
)
if (requireNamespace("rcdk", quietly = TRUE)) {
  input_file <- tempfile(fileext = ".xlsx")
  writexl::write_xlsx(toy, input_file)
  res <- try(reft_run_simple(input_file, output_dir = tempdir(), verbose = FALSE),
             silent = TRUE)
  if (!inherits(res, "try-error")) head(res$descriptors)
}
```

Index

REFT, 2
reft_calc_descriptors, 2
reft_kegg_microbe_run, 3
reft_match_smiles, 4
reft_run, 5
reft_run_simple, 6