Package 'HARplus'

October 19, 2025

Title Enhanced R Package for 'GEMPACK' .har and .sl4 Files

Version 1.1.0

Description Provides tools for processing and analyzing .har and .sl4 files, making it easier for 'GEMPACK' users and 'GTAP' researchers to handle large economic datasets. It simplifies the management of multiple experiment results, enabling faster and more efficient comparisons without complexity. Users can extract, restructure, and merge data seamlessly, ensuring compatibility across different tools. The processed data can be exported and used in 'R', 'Stata', 'Python', 'Julia', or any software that supports Text, CSV, or 'Excel' formats.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

BugReports https://github.com/bodysbobb/HARplus/issues/

URL https://github.com/bodysbobb/HARplus/, https://www.pattawee-pp.com/HARplus/

Imports openxlsx, haven, stats, utils, tidyr, tools, tidyselect

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Pattawee Puangchit [aut, cre]

Maintainer Pattawee Puangchit <ppuangch@purdue.edu>

Repository CRAN

Date/Publication 2025-10-19 05:10:21 UTC

Contents

`compare_var_structure` *Compare Variable Structures Across SL4 and HAR Objects*

Description

Compares variable structures across multiple SL4 and HAR datasets to ensure compatibility. Identifies matching and mismatched variable structures, helping users diagnose inconsistencies.

Usage

```
compare_var_structure(variables = NULL, ..., keep_unique = FALSE)
```

Arguments

`variables`	Character vector. Variable names to compare. Use `NULL` or `"ALL"` to compare all variables.
`...`	Named SL4 or HAR objects to compare.
`keep_unique`	Logical. If `TRUE`, returns unique variable structures across datasets instead of checking for compatibility. Default is `FALSE`.

Details

- Verifies whether variables have consistent structures across multiple datasets.
- Ensures correct ordering of dimensions and checks for structural compatibility.
- If `keep_unique = TRUE`, returns a list of unique variable structures instead of performing a direct comparison.
- Useful for merging or aligning datasets before further processing.
- Helps detect differences in variable dimensions, which may arise due to model updates or dataset variations.

Value

A list containing:

- `match`: A data frame listing variables with identical structures across datasets.
- `diff`: A data frame listing variables with mismatched structures, useful for debugging and alignment.
- If `keep_unique = TRUE`, instead of `match` and `diff`, returns a data frame with distinct variable structures across datasets.

Author(s)

Pattawee Puangchit

See Also

`get_var_structure`, `get_dim_patterns`, `get_dim_elements`

Examples

```
# Import sample data:
har_data1 <- load_harx(system.file("extdata", "TAR10-WEL.har", package = "HARplus"))
har_data2 <- load_harx(system.file("extdata", "SUBT10-WEL.har", package = "HARplus"))

# Compare structure for a single variable across multiple datasets
compare_var_structure("A", har_data1, har_data2)

# Compare structure for multiple variables across multiple datasets
comparison_multiple <- compare_var_structure(c("A", "E1"), har_data1, har_data2)

# Extract unique variable structures across multiple datasets
unique_vars <- compare_var_structure("ALL", har_data1, har_data2, keep_unique = TRUE)
```

---

export_data *Export Data to Various Formats (CSV/STATA/TEXT/RDS/XLSX)*

Description

Exports structured SL4 or HAR data to multiple file formats, including CSV, Stata, TXT, RDS, and XLSX. Supports nested lists, automatic subfolder creation, and multi-sheet Excel exports.

Usage

```
export_data(
  data,
  output_path,
  format = "csv",
  prefix = "",
```

```
  create_subfolder = FALSE,
  multi_sheet_xlsx = FALSE,
  xlsx_filename = NULL,
  report_output = FALSE
)
```

Arguments

`data`	A list or data frame. The SL4 or HAR data to export.
`output_path`	Character. The base output directory or file path.
`format`	Character. The export format (`"csv"`, `"stata"`, `"txt"`, `"rds"`, `"xlsx"`). Default is `"csv"`.
`prefix`	Character. An optional prefix added to exported file names. Default is `""` (empty).
`create_subfolder`	Logical. If `TRUE`, creates a subfolder for each format. Default is `FALSE`.
`multi_sheet_xlsx`	Logical. If TRUE, exports lists as multi-sheet XLSX files.
`xlsx_filename`	An optional filename for the XLSX file (used when `multi_sheet_xlsx = TRUE`).
`report_output`	Logical. If TRUE, generates an export report.

Details

- Supports exporting data in `"csv"`, `"stata"`, `"txt"`, `"rds"`, and `"xlsx"` formats.
- Handles nested lists and exports each data frame individually.
- Optionally creates subfolders for each format (`create_subfolder = TRUE`).
- Customizes file names using `prefix`.
- When multi_sheet_xlsx = TRUE, all exported data is stored in a **single Excel workbook**, with each dataset as a separate sheet.
- If exporting to Stata (`"stata"` format), column names containing `.` will be replaced with `_` to ensure compatibility.
- If `multi_sheet_xlsx = TRUE`, list elements are exported as separate sheets in a single XLSX file.
- The function creates necessary directories if they do not exist.

Value

A list containing the file paths of the exported data.

Author(s)

Pattawee Puangchit

See Also

`pivot_data`, `get_data_by_var`, `get_data_by_dims`

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))

# Extract data
data_multiple <- get_data_by_var(c("qo", "pca"), sl4_data)

# Export
export_data(data_multiple, file.path(tempdir(), "output_directory"),
            format = c("csv", "xlsx", "stata", "txt", "rds"),
            create_subfolder = TRUE,
            multi_sheet_xlsx = TRUE)
```

`get_data_by_dims` *Extract Data by Dimension Patterns from SL4 or HAR Objects*

Description

Retrieves structured data from SL4 or HAR objects based on specified dimension patterns. Supports multiple experiments and merging datasets while maintaining structured dimension metadata.

Usage

```
get_data_by_dims(
  patterns = NULL,
  ...,
  experiment_names = NULL,
  subtotal_level = FALSE,
  rename_cols = NULL,
  merge_data = FALSE,
  pattern_mix = FALSE
)
```

Arguments

`patterns`	Character vector. Dimension patterns to extract. Use `"ALL"` or `NULL` to extract all available patterns.
`...`	One or more SL4 or HAR data objects loaded using `load_sl4x()` or `load_harx()`.
`experiment_names`	Character vector. Names assigned to each dataset. If `NULL`, names are inferred.
`subtotal_level`	Character or logical. Determines which decomposition levels to retain: • `"total"`: Keeps only `"TOTAL"` values. • `"decomposed"`: Keeps only decomposed values (excludes `"TOTAL"`). • `"all"`: Keeps all rows.

- TRUE: Equivalent to "all", retaining both "TOTAL" and decomposed values.
- FALSE: Equivalent to "total", keeping only "TOTAL" values.

rename_cols	Named vector. Column name replacements (c("old_name" = "new_name")).
merge_data	Logical. If TRUE, attempts to merge data across multiple experiments. Default is FALSE.
pattern_mix	Logical. If TRUE, allows flexible pattern matching, ignoring dimension order. Default is FALSE.

Details

- Extracts variables matching specified dimension patterns.
- Allows for flexible pattern matching (pattern_mix = TRUE).
- Supports merging data across multiple experiments (merge_data = TRUE).
- Provides column renaming functionality (rename_cols).
- Handles subtotal filtering (subtotal_level), controlling whether "TOTAL" or decomposed values are retained.

Value

A structured list of extracted data:

- If merge_data = FALSE, returns a named list where each element corresponds to an experiment.
- If merge_data = TRUE, returns a named list of all merged data

Author(s)

Pattawee Puangchit

See Also

get_data_by_var, group_data_by_dims

Examples

```
# Import sample data:
sl4_data <- load_sl4x(
  system.file("extdata", "TAR10.sl4", package = "HARplus")
)
sl4_data1 <- load_sl4x(
  system.file("extdata", "SUBT10.sl4", package = "HARplus")
)

# Extract data for a single dimension pattern
data_single_pattern <- get_data_by_dims(
  "comm*reg",
  sl4_data
)
```

```
# Extract multiple dimension patterns
data_multiple_patterns <- get_data_by_dims(
  c("comm*reg", "REG*ACTS"),
  sl4_data
)

# Extract all dimension patterns separately from multiple datasets
data_all_patterns <- get_data_by_dims(
  NULL,
  sl4_data, sl4_data1,
  merge_data = FALSE
)

# Merge data for identical patterns across multiple datasets
data_merged_patterns <- get_data_by_dims(
  NULL,
  sl4_data, sl4_data1,
  merge_data = TRUE
)

# Merge data while allowing interchangeable dimensions (e.g., A*B = B*A)
data_pattern_mixed <- get_data_by_dims(
  NULL,
  sl4_data, sl4_data1,
  merge_data = TRUE,
  pattern_mix = TRUE
)

# Retain only "TOTAL" values
data_total_only <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  subtotal_level = "total"
)
data_total_only_alt <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  subtotal_level = FALSE
)

# Retain only decomposed components
data_decomposed_only <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  subtotal_level = "decomposed"
)

# Retain all value levels
data_all_decomp <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  subtotal_level = "all"
```

```
)
data_all_decomp_alt <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  subtotal_level = TRUE
)

# Rename specific columns
data_renamed <- get_data_by_dims(
  "comm*reg",
  sl4_data,
  rename_cols = c(REG = "Region", COMM = "Commodity")
)

# Merge data with custom experiment names
data_merged_experiments <- get_data_by_dims(
  "comm*reg",
  sl4_data, sl4_data1,
  experiment_names = c("EXP1", "EXP2"),
  merge_data = TRUE
)
```

`get_data_by_var` *Extract Variable Data from SL4 or HAR Objects*

Description

Extracts structured data for one or more variables from SL4 or HAR objects, transforming array-like data into a tidy format.

Usage

```
get_data_by_var(
  var_names = NULL,
  ...,
  experiment_names = NULL,
  subtotal_level = FALSE,
  rename_cols = NULL,
  merge_data = FALSE
)
```

Arguments

`var_names`	Character vector. Variable names to extract. Use `"ALL"` or `NULL` to extract all available variables.
`...`	One or more SL4 or HAR data objects loaded using `load_sl4x()` or `load_harx()`.
`experiment_names`	Character vector. Names assigned to each dataset. If `NULL`, names are inferred.

`subtotal_level` Character or logical. Determines which decomposition levels to retain:

- `"total"`: Keeps only `"TOTAL"` values.
- `"decomposed"`: Keeps only decomposed values (excludes `"TOTAL"`).
- `"all"`: Keeps all rows.
- `TRUE`: Equivalent to `"all"`, retaining both `"TOTAL"` and decomposed values.
- `FALSE`: Equivalent to `"total"`, keeping only `"TOTAL"` values.

`rename_cols` Named vector. Column name replacements (`c("old_name" = "new_name")`).

`merge_data` Logical. If `TRUE`, attempts to merge data across multiple experiments. Default is `FALSE`.

Details

- Retrieves specific variables, multiple variables, or all available variables from SL4 or HAR datasets.
- Supports merging data from multiple experiments (`merge_data = TRUE`).
- Allows renaming of column names (`rename_cols`).
- Handles subtotal filtering (`subtotal_level`), controlling whether `"TOTAL"` or decomposed values are retained.

Value

A list of structured data:

- If `merge_data = FALSE`, returns a named list where each element corresponds to an experiment.
- If `merge_data = TRUE`, returns a named list of all merged data

Author(s)

Pattawee Puangchit

See Also

`get_data_by_dims`, , `group_data_by_dims`, `load_sl4x`, `load_harx`

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))
sl4_data1 <- load_sl4x(system.file("extdata", "SUBT10.sl4", package = "HARplus"))

# Extract a single variable
data_qo <- get_data_by_var("qo", sl4_data)

# Extract multiple variables
data_multiple <- get_data_by_var(c("qo", "qgdp"), sl4_data)

# Extract all variables separately from multiple datasets
```

```
data_all <- get_data_by_var(NULL, sl4_data, sl4_data1, merge_data = FALSE)

# Merge variable data across multiple datasets
data_merged <- get_data_by_var(NULL, sl4_data, sl4_data1, merge_data = TRUE)

# Retain only "TOTAL" values, removing decomposed components (subtotal_level = "total" or FALSE)
data_total_only <- get_data_by_var("qo", sl4_data, subtotal_level = "total")
data_total_only_alt <- get_data_by_var("qo", sl4_data, subtotal_level = FALSE)

# Retain only decomposed components, removing "TOTAL" (subtotal_level = "decomposed")
data_decomposed_only <- get_data_by_var("qo", sl4_data, subtotal_level = "decomposed")

# Retain all value levels (subtotal_level = "all" or TRUE)
data_all_decomp <- get_data_by_var("qo", sl4_data, subtotal_level = "all")
data_all_decomp_alt <- get_data_by_var("qo", sl4_data, subtotal_level = TRUE)

# Rename specific columns
data_renamed <- get_data_by_var("qo", sl4_data, rename_cols = c(REG = "Region", COMM = "Commodity"))

# Merge data across multiple datasets with custom experiment names
data_merged_experiments <- get_data_by_var("qo", sl4_data, sl4_data1,
experiment_names = c("EXP1", "EXP2"),
merge_data = TRUE)
```

`get_dim_elements` *Get Dimension Elements from SL4 and HAR Objects*

Description

Extracts and lists unique dimension elements (e.g., `REG`, `COMM`, `ACTS`) from one or more datasets.

Usage

```
get_dim_elements(..., keep_unique = FALSE)
```

Arguments

`...`	One or more structured SL4 or HAR objects containing dimension information.
`keep_unique`	Logical. If `TRUE`, returns only unique dimension elements across inputs. Default is `FALSE`.

Value

A data frame containing unique dimension elements.

Author(s)

Pattawee Puangchit

See Also

get_dim_patterns, get_var_structure

Examples

```
# Import sample data:
sl4_data1 <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))
sl4_data2 <- load_sl4x(system.file("extdata", "SUBT10.sl4", package = "HARplus"))


# Extract dimension elements from a single dataset
get_dim_elements(sl4_data1)

# Extract dimension elements from multiple datasets
get_dim_elements(sl4_data1, sl4_data2)

# Extract unique dimension elements across datasets
get_dim_elements(sl4_data1, sl4_data2, keep_unique = TRUE)
```

---

get_dim_patterns *Get Dimension Patterns from SL4 and HAR Objects*

Description

Extracts and lists unique dimension patterns (e.g., REG*COMM, REG*REG*ACTS) from one or more datasets.

Usage

```
get_dim_patterns(..., keep_unique = FALSE)
```

Arguments

...	One or more structured SL4 or HAR objects containing dimension information.
keep_unique	Logical. If TRUE, returns only unique dimension patterns. Default is FALSE.

Details

- Extracts dimension structure details from the dataset.
- If multiple datasets are provided, combines their dimension information.
- If keep_unique = TRUE, returns only distinct dimension patterns.

Value

A data frame containing:

- DimPattern: The unique dimension patterns.

Author(s)

Pattawee Puangchit

See Also

get_dim_elements, get_var_structure

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))
sl4_data2 <- load_sl4x(system.file("extdata", "SUBT10.sl4", package = "HARplus"))

# Extract dimension patterns
get_dim_patterns(sl4_data)

# Extract only unique dimension patterns across datasets
get_dim_patterns(sl4_data, sl4_data2, keep_unique = TRUE)
```

---

get_var_structure *Get Variable Structure Summary from SL4 and HAR Objects*

Description

Generates a summary of the variables within one or more SL4 or HAR objects, listing their dimension sizes, structures, and optionally, column and observation counts.

Usage

```
get_var_structure(variables = NULL, ..., include_col_size = FALSE)
```

Arguments

variables	Character vector. Variable names to summarize. Use NULL or "ALL" to summarize all variables.
...	One or more SL4 or HAR objects created using load_sl4x() or load_harx().
include_col_size	Logical. If TRUE, includes column and observation counts. Default is FALSE.

Details

- Extracts dimension structures for variables in one or more SL4 or HAR datasets.
- If include_col_size = TRUE, adds column and observation counts.
- Supports multiple datasets and returns results as a named list, with each dataset's summary stored separately.
- Can summarize specific variables or "ALL".

Value

A named list, where each element contains a data frame with:

- `Variable`: The variable name.
- `Dimensions`: The associated dimensions.
- `DimSize`: The number of dimensions.
- `DataShape`: The shape of the data (e.g., `10x20x30`).
- `No.Col`: (Optional) The number of columns.
- `No.Obs`: (Optional) The number of observations.

Author(s)

Pattawee Puangchit

See Also

`get_dim_patterns`, `get_dim_elements`

Examples

```
# Import data sample:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))
sl4_data1 <- load_sl4x(system.file("extdata", "SUBT10.sl4", package = "HARplus"))

# Get summary for all variables in a single dataset
get_var_structure(data_obj = sl4_data)

# Get summary for specific variables
get_var_structure(c("gdp", "trade"), sl4_data)

# Include column and observation counts
get_var_structure("ALL", sl4_data, include_col_size = TRUE)

# Compare structures across multiple datasets
get_var_structure("ALL", sl4_data, sl4_data1)

# Include column and observation counts across multiple datasets
get_var_structure("ALL", sl4_data, sl4_data1, include_col_size = TRUE)
```

---

group_data_by_dims *Group Data by Dimension Patterns in SL4 or HAR Objects*

Description

Groups extracted SL4 or HAR data based on specified dimension structures and priority rules. Supports automatic renaming, merging, subtotal filtering, and structured metadata handling.

Usage

```
group_data_by_dims(
  patterns = NULL,
  ...,
  priority,
  rename_cols = NULL,
  experiment_names = NULL,
  subtotal_level = FALSE,
  auto_rename = FALSE
)
```

Arguments

patterns	Character vector. Dimension patterns to extract. Use "ALL" or NULL to extract all available patterns.
...	One or more SL4 or HAR objects loaded using load_sl4x() or load_harx().
priority	Named list. Specifies priority dimension elements (c("group_name" = c("dim1", "dim2"))).
rename_cols	Named vector. Column name replacements (c("old_name" = "new_name")).
experiment_names	Character vector. Names assigned to each dataset. If NULL, names are inferred.
subtotal_level	Character or logical. Determines which decomposition levels to retain: • "total": Keeps only "TOTAL" values. • "decomposed": Keeps only decomposed values (excludes "TOTAL"). • "all": Keeps all rows. • TRUE: Equivalent to "all", retaining both "TOTAL" and decomposed values. • FALSE: Equivalent to "total", keeping only "TOTAL" values.
auto_rename	Logical. If TRUE, automatically renames dimensions for consistency. Default is FALSE.

Details

- Groups extracted variables based on dimension elements.
- Applies predefined priority rules to structure the data.
- Allows automatic renaming of dimensions (auto_rename = TRUE).
- Supports merging of grouped data across multiple experiments.
- Handles subtotal filtering (subtotal_level), controlling whether "TOTAL" or decomposed values are retained.

Value

A structured list of grouped data:

- A named list where each element corresponds to a dimension size group (e.g., "2D", "3D").
- Each group contains dimension-grouped data based on priority rules.
- If unmerged data exists, includes a report attribute detailing merge issues.

Author(s)

Pattawee Puangchit

See Also

get_data_by_dims, get_data_by_var, load_sl4x, load_harx

Examples

```
# Import sample data
sl4_data1 <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))
sl4_data2 <- load_sl4x(system.file("extdata", "SUBT10.sl4", package = "HARplus"))

# Case 1: Multiple priority levels (Sector then Region) with auto_rename
priority_list <- list(
  "Sector" = c("COMM", "ACTS"),
  "Region" = c("REG")
)
grouped_data_multiple <- group_data_by_dims(
  patterns = "ALL",
  sl4_data1,
  priority = priority_list,
  auto_rename = TRUE
)

# Case 2: Single priority (Region only) with auto_rename
priority_list <- list("Region" = c("REG"))
grouped_data_single <- group_data_by_dims(
  patterns = "ALL",
  sl4_data1, sl4_data2,
  priority = priority_list,
  auto_rename = TRUE
)

# Case 3: Multiple priorities without auto_rename
priority_list <- list(
  "Sector" = c("COMM", "ACTS"),
  "Region" = c("REG")
)
grouped_data_no_rename <- group_data_by_dims(
  patterns = "ALL",
  sl4_data1,
  priority = priority_list,
  auto_rename = FALSE
)
```

---

load_harx *Load and Process HAR Files with Header Selection*

Description

Reads a GEMPACK HAR file and extracts structured data while maintaining compatibility with standard HAR formats. Provides flexibility in naming conventions and header selection.

Usage

```
load_harx(
  file_path,
  coefAsname = FALSE,
  lowercase = FALSE,
  select_header = NULL
)
```

Arguments

`file_path`	Character. The file path to the HAR file.
`coefAsname`	Logical. If `TRUE`, replaces four-letter headers with coefficient names when available. Default is `FALSE`.
`lowercase`	Logical. If `TRUE`, converts all variable names to lowercase. Default is `FALSE`.
`select_header`	Character vector. Specific headers to read; if `NULL`, all headers are read. Example: `select_header = c("A", "E1")`.

Details

- Uses `load_harplus()` internally for efficient HAR file reading.
- Allows optional conversion of variable names to lowercase (`lowercase = TRUE`).
- Supports coefficient-based naming (`coefAsname = TRUE`).
- Enables selective header extraction via `select_header = c("A", "E1")`.
- Returns structured data with explicit dimension names and sizes.

Value

A structured list containing:

- `data`: Extracted HAR variable data stored as matrices, arrays, or vectors.
- `dimension_info`: A list with:
  - `dimension_string`: A textual representation of dimensions (e.g., "REG*COMM*YEAR").
  - `dimension_names`: The names of each dimension.
  - `dimension_sizes`: The size of each dimension.

Author(s)

Pattawee Puangchit

See Also

`load_sl4x`, `get_data_by_var`, `get_data_by_dims`

Examples

```
# Path to example files
har_path <- system.file("extdata", "TAR10-WEL.har", package = "HARplus")

# Basic loading
har_data <- load_harx(har_path)

# Load with coefficient names
har_data_coef <- load_harx(har_path, coefAsname = TRUE)

# Load with lowercase names
har_data_lower <- load_harx(har_path, lowercase = TRUE)

# Load specific headers
har_selected <- load_harx(har_path, select_header = c("A", "E1"))

# Load with multiple options
har_combined <- load_harx(har_path,
                          coefAsname = TRUE,
                          lowercase = TRUE,
                          select_header = c("A", "E1"))
```

---

load_sl4x *Load and Process SL4 Files with Enhanced Options*

Description

Reads an SL4 file and processes its structured data into an enhanced SL4 object. Extracts structured variable information, dimensions, and handles subtotal columns.

Usage

```
load_sl4x(file_path, lowercase = FALSE, select_header = NULL)
```

Arguments

file_path	Character. The full path to the SL4 file to be read.
lowercase	Logical. If TRUE, converts all variable names to lowercase. Default is FALSE.
select_header	Character vector. Specific headers to extract; if NULL, all headers are read.

Details

- Uses load_harplus() internally for optimized SL4 file reading.
- Extracts variable names, dimension structures, and metadata.
- Converts variable names to lowercase if lowercase = TRUE.
- Allows the selection of specific headers using select_header.
- Returns structured data with explicit dimension names and sizes.

Value

A structured list containing:

- `data`: Extracted SL4 variable data, stored as arrays or matrices.
- `dimension_info`: A list with:
  - `dimension_string`: A textual representation of dimensions (e.g., "REG*COMM*YEAR").
  - `dimension_names`: The names of each dimension.
  - `dimension_sizes`: The size of each dimension.

Author(s)

Pattawee Puangchit

See Also

`load_harx`, `get_data_by_var`, `get_data_by_dims`

Examples

```
# Path to example files
sl4_path <- system.file("extdata", "TAR10.sl4", package = "HARplus")

# Basic loading
sl4_data <- load_sl4x(sl4_path)

# Load with lowercase names
sl4_data_lower <- load_sl4x(sl4_path, lowercase = TRUE)

# Load specific headers
sl4_selected <- load_sl4x(sl4_path, select_header = c("qo", "qgdp"))
```

`pivot_data` *Pivot Data from SL4 or HAR Objects*

Description

Transforms long-format SL4 or HAR data into wide format by pivoting selected columns. Supports both single data frames and nested lists.

Usage

```
pivot_data(data_obj, pivot_cols, name_repair = "unique")
```

Arguments

`data_obj`	A list or data frame. The SL4 or HAR data to pivot.
`pivot_cols`	Character vector. Column names to use as pivot keys.
`name_repair`	Character. Method for handling duplicate column names (`"unique"`, `"minimal"`, `"universal"`). Default is `"unique"`.

Details

- Uses `tidyr::pivot_wider()` internally to reshape data.
- Allows multiple columns to be pivoted simultaneously.
- Recursively processes nested lists, ensuring all data frames are transformed.

Value

A transformed data object where the specified `pivot_cols` are pivoted into wide format.

Author(s)

Pattawee Puangchit

See Also

`get_data_by_var`, `get_data_by_dims`

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))

# Extract multiple variables
data_multiple <- get_data_by_var(c("qo", "qxs"), sl4_data)

# Pivot a single column
pivoted_data <- pivot_data(data_multiple, pivot_cols = "REG")

# Pivot multiple columns
pivoted_data_multi <- pivot_data(data_multiple, pivot_cols = c("REG", "COMM"))
```

---

pivot_data_hierarchy *Create Hierarchical Pivot Table from SL4 or HAR Objects*

Description

Creates hierarchical pivot tables from structured SL4 or HAR data, with optional Excel export. Supports both single data frames and nested lists, preserving dimension hierarchies.

Usage

```
pivot_data_hierarchy(
  data_obj,
  pivot_cols,
  name_repair = "unique",
  export = FALSE,
  file_path = NULL,
  xlsx_filename = NULL
)
```

Arguments

data_obj	A list or data frame. The SL4 or HAR data to pivot.
pivot_cols	Character vector. Column names to use as pivot keys in order of hierarchy.
name_repair	Character. Method for handling duplicate column names ("unique", "minimal", "universal"). Default is "unique".
export	Logical. If TRUE, exports result to Excel. Default is FALSE.
file_path	Character. Required if export = TRUE. The path for Excel export.
xlsx_filename	Character. The name for the Excel file when using multi_sheet_xlsx. If NULL, uses the name of the dataset. Default is NULL.

Details

- Transforms data into hierarchical pivot format with nested column headers.
- Supports multiple pivot columns in specified order (e.g., REG > COMM).
- Handles both single data frames and nested list structures.
- Optional direct export to Excel with formatted hierarchical headers.
- Uses efficient data processing with tidyr and openxlsx.

Value

A pivoted data object with hierarchical structure:

- If input is a data frame: Returns a hierarchical pivot table.
- If input is a list: Returns a nested list of hierarchical pivot tables.
- If export = TRUE: Invisibly returns the pivoted object after Excel export.

Author(s)

Pattawee Puangchit

See Also

pivot_data

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))

# Extract data
data_multiple <- get_data_by_var(c("qo", "pca"), sl4_data)

# Create hierarchical pivot without export
pivot_hier <- pivot_data_hierarchy(data_multiple,
                                   pivot_cols = c("REG", "COMM"))

# Create and export to Excel in one step
pivot_export <- pivot_data_hierarchy(data_multiple,
                                     pivot_cols = c("REG", "COMM"),
                                     export = TRUE,
                                   file_path = file.path(tempdir(), "pivot_output.xlsx"))
```

---

rename_dims *Rename Dimensions in SL4 or HAR Objects*

Description

Renames dimension and list names in structured SL4 or HAR objects.

Usage

```
rename_dims(data_obj, mapping_df, rename_list_names = FALSE)
```

Arguments

data_obj	A structured SL4 or HAR object.
mapping_df	A two-column data frame where the first column (old) contains the current names, and the second column (new) contains the new names.
rename_list_names	Logical. If TRUE, renames list element names. Default is FALSE.

Details

- Replaces old dimension names with new ones as specified in mapping_df.
- If rename_list_names = TRUE, renames list element names as well.
- Ensures consistency across SL4 and HAR datasets.

Value

The modified SL4 or HAR object with updated dimension names and, optionally, updated list names.

Author(s)

Pattawee Puangchit

See Also

get_data_by_var, get_data_by_dims

Examples

```
# Import sample data:
sl4_data <- load_sl4x(system.file("extdata", "TAR10.sl4", package = "HARplus"))

# Define a renaming map
mapping_df <- data.frame(
  old = c("REG", "COMM"),
  new = c("Region", "Commodity")
)

# Rename columns in the dataset
rename_dims(sl4_data, mapping_df)

# Rename both columns and list names
rename_dims(sl4_data, mapping_df, rename_list_names = TRUE)
```

---

save_har *Save Data to GEMPACK HAR Format*

Description

Writes one or more data matrices or arrays into a GEMPACK-compatible HAR file format, automatically generating associated 1C set headers.

Usage

```
save_har(
  data_list,
  file_path,
  dimensions,
  value_cols = NULL,
  long_desc = NULL,
  coefficients = NULL,
  export_sets = TRUE,
  lowercase = TRUE,
  dim_order = NULL
)
```

Arguments

`data_list`	A named list of data frames or arrays (names up to 4 characters).
`file_path`	Path to the output HAR file.
`dimensions`	A named list specifying dimension columns for each header. • HAR dimension names follow these column names; rename columns before saving to change them.
`value_cols`	A named vector of value column names. Defaults to "Value".
`long_desc`	A named vector of long descriptions (optional).
`coefficients`	A named vector of coefficient names (optional).
`export_sets`	Logical; if TRUE, exports dimension sets as 1C headers. Default is TRUE.
`lowercase`	Logical; if TRUE, converts elements to lowercase. Default is TRUE.
`dim_order`	Optional dimension ordering specification. Can be: • `NULL` (default): alphabetical A–Z ordering • a data frame with columns matching dimension names • a named list specifying order for each dimension • a character string giving the path to an Excel or CSV file with order definitions

Details

- Supports `1CFULL` (string sets) and `REFULL` (real arrays) headers
- `2IFULL` (integer) and `RESPSE` (sparse) types are not implemented
- Up to seven dimensions and ~2 million elements per chunk
- No practical file-size limit or header count restriction

Value

Invisibly returns a list containing export metadata, including file path and header summary.

Author(s)

Pattawee Puangchit

See Also

`load_harx`, `load_sl4x`

Examples

```
# Example 1: Save a single matrix
har_path <- system.file("extdata", "TAR10-WEL.har", package = "HARplus")
har_data <- load_harx(har_path)
welfare_data <- get_data_by_var("A", har_data)
welfare_data <- welfare_data[["har_data"]][["A"]]
```

```
save_har(
  data_list   = list(WELF = welfare_data),
  file_path   = file.path(tempdir(), "output_single.har"),
  dimensions  = list(WELF = c("REG", "COLUMN")),
  value_cols  = c(WELF = "Value"),
  long_desc   = c(WELF = "Welfare Decomposition"),
  export_sets = TRUE,
  lowercase   = FALSE
)

# Example 2: Save multiple matrices
welfare_data <- get_data_by_var(c("A", "A1"), har_data)
welfare_data <- welfare_data[["har_data"]]

save_har(
  data_list   = list(WELF = welfare_data[["A"]],
                     DECOM = welfare_data[["A1"]]),
  file_path   = file.path(tempdir(), "output_multi.har"),
  dimensions  = list(WELF = c("REG", "COLUMN"),
                     DECOM = c("REG", "ALLOCEFF")),
  value_cols  = list(WELF = "Value",
                     DECOM = "Value"),
  long_desc   = list(WELF = "Welfare Decomposition",
                     DECOM = "Allocative efficiency effect"),
  export_sets = TRUE,
  lowercase   = FALSE
)

# Example 3: Apply mapping file for sorting
mapping <- list(
  REG    = c("RestofWorld", "MENA", "EU_28"),
  COLUMN = c("alloc_A1", "tot_E1")
)

save_har(
  data_list   = list(
    WELF  = welfare_data[["A"]],
    DECOM = welfare_data[["A1"]]
  ),
  file_path   = file.path(tempdir(), "output_sorted.har"),
  dimensions  = list(
    WELF  = c("REG", "COLUMN"),
    DECOM = c("REG", "ALLOCEFF")
  ),
  value_cols  = list(
    WELF  = "Value",
    DECOM = "Value"
  ),
  long_desc   = list(
    WELF  = "Welfare Decomposition",
    DECOM = "Allocative efficiency effect"
  ),
  export_sets = TRUE,
```

```
  dim_order   = mapping,
  lowercase   = FALSE
)
```

Index