

Package ‘GTAPViz’

July 21, 2025

Title Automating 'GTAP' Data Processing and Visualization

Version 1.1.3

Description Tools to streamline the extraction, processing, and visualization of Computable General Equilibrium (CGE) results from 'GTAP' models. Designed for compatibility with both .har and .sl4 files, the package enables users to automate data preparation, apply mapping metadata, and generate high-quality plots and summary tables with minimal coding. 'GTAPViz' supports flexible export options (e.g., Text, CSV, 'Stata', or 'Excel' formats). This facilitates efficient post-simulation analysis for economic research and policy reporting. Includes helper functions to filter, format, and customize outputs with reproducible styling.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

BugReports <https://github.com/bodysbobb/GTAPViz/issues/>

URL <https://bodysbobb.github.io/GTAPViz/>

Imports HARplus, ggplot2, dplyr, tidyr, openxlsx, colorspace,
grDevices, scales, utils, methods, stringdist, stats, glue,
openxlsx2

VignetteBuilder knitr

Suggests knitr, rmarkdown, devtools

Depends R (>= 3.5)

NeedsCompilation no

Author Pattawee Puangchit [aut, cre]

Maintainer Pattawee Puangchit <ppuangch@purdue.edu>

Repository CRAN

Date/Publication 2025-04-19 12:42:05 UTC

Contents

add_mapping_info	2
auto_gtap_data	4
comparison_plot	7
convert_units	9
create_export_config	11
create_plot_style	12
create_title_format	17
detail_plot	18
get_all_config	21
get_color_palette	22
pivot_table_with_filter	23
rename_value	25
report_table	26
sort_plot_data	28
stack_plot	30
Index	33

add_mapping_info	<i>Add Mapping Information to GTAP Data</i>
------------------	---

Description

Adds descriptions and unit information to GTAP data based on a specified mapping mode. Supports external mappings or default GTAPv7 mappings, allowing users to enrich datasets with standardized metadata.

Adds **description** and **unit** information to GTAP data structures based on a specified mapping mode. This function supports internal GTAPv7 mappings, external mappings, or a combination of both.

Usage

```
add_mapping_info(  
  data_list,  
  external_map = NULL,  
  mapping = "GTAPv7",  
  description_info = TRUE,  
  unit_info = TRUE  
)
```

Arguments

- data_list A list or nested data structure containing GTAP output data frames.
- external_map Optional data frame. External mapping must include columns: "Variable", "Description", and "Unit".

mapping	Character. Mapping mode for assigning metadata to variables. Options: • "GTAPv7": Use GTAPv7 internal definitions (default). • "Yes": Use only the provided external_map. • "Mix": Use external definitions first, then fallback to GTAPv7 for missing values. • "No": Skip mapping entirely.
description_info	Logical. If TRUE, adds or updates variable descriptions. Default: TRUE.
unit_info	Logical. If TRUE, adds or updates unit information. Default: TRUE.

Details

The mapping argument supports:

Value

The same data structure as input with added "Description" and "Unit" columns, if applicable.

Author(s)

Pattawee Puangchit

See Also

[convert_units](#), [rename_value](#)

Examples

```
# Load GTAP SL4 data
input_path <- system.file("extdata/in", package = "GTAPviz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

# Add mapping using GTAPv7 defaults
gtap_data <- add_mapping_info(sl4.plot.data, mapping = "GTAPv7")

# Use a custom mapping file
my_map <- data.frame(
  Variable = c("qgdp", "EV"),
  Description = c("Real GDP", "Welfare"),
  Unit = c("percent", "million USD")
)
gtap_data <- add_mapping_info(sl4.plot.data, external_map = my_map, mapping = "Mix")
```

auto_gtap_data

Process GTAP Data Automation with Flexible Output Options

Description

Processes GTAP data from sl4 and har files with options for exporting and preparing plot-ready data.

Usage

```
auto_gtap_data(
  experiment,
  input_path = NULL,
  output_path = NULL,
  sl4_suffix = "",
  har_suffix = "",
  process_sl4_vars = NULL,
  process_har_vars = NULL,
  mapping_info = "GTAPv7",
  sl4_mapping_info = NULL,
  har_mapping_info = NULL,
  sl4_extract_method = "get_data_by_dims",
  har_extract_method = "get_data_by_var",
  sl4_priority = NULL,
  har_priority = NULL,
  sl4_convert_unit = NULL,
  har_convert_unit = NULL,
  decimals = 4,
  rename_columns = TRUE,
  region_select = NULL,
  sector_select = NULL,
  subtotal_level = FALSE,
  plot_data = TRUE,
  output_formats = NULL,
  sl4_output_name = "sl4.plot.data",
  har_output_name = "har.plot.data",
  macro_output_name = "GTAPMacro",
  add_scenario_ranking = FALSE,
  rank_column = "ScenarioRank"
)
```

Arguments

experiment	Character vector. Case names to process.
input_path	Character. Path to the input folder.
output_path	Character. Path to the output folder.

sl4_suffix	Character. Custom suffix for SL4 files (e.g., "", "-custom").
har_suffix	Character. Custom suffix for HAR files (e.g., "-WEL").
process_sl4_vars	Character, NULL, or FALSE. Variables to extract from SL4 data: • Character vector: Specific variable names. • NULL: Extract all. • FALSE: Skip SL4 processing.
process_har_vars	Character, NULL, or FALSE. Variables to extract from HAR data: • Character vector: Specific variable names. • NULL: Extract all. • FALSE: Skip HAR processing.
mapping_info	Character. Metadata mode for variable descriptions and units. Options: "GTAPv7" (default), "Yes", "No", "Mix". See add_mapping_info() for full details.
sl4_mapping_info	Data frame or NULL. Mapping for SL4 variables. Must include columns "Variable", "Description", and "Unit".
har_mapping_info	Data frame or NULL. Mapping for HAR variables, structured as above.
sl4_extract_method	Character. SL4 extraction method: "get_data_by_dims", "get_data_by_var", or "group_data_by_dims".
har_extract_method	Character. HAR extraction method. Same options as above.
sl4_priority	Optional list. Required only when sl4_extract_method is "group_data_by_dims". Specifies priority rules for SL4 data grouping.
har_priority	Optional list. Required only when har_extract_method is "group_data_by_dims". Specifies priority rules for HAR data grouping.
sl4_convert_unit	Character or NULL. Optional SL4 unit conversion. Valid options: "mil2bil", "bil2mil", "pct2frac", "frac2pct". See convert_units for details.
har_convert_unit	Character or NULL. Optional HAR unit conversion. Same options as above.
decimals	Integer or NULL. Number of decimal places to round numeric values. Set to NULL to disable rounding.
rename_columns	Logical. If TRUE (default), renames GTAP codes (e.g., "REG" → "Region", "COMM" → "Commodity").
region_select	Optional character vector. Filters data to selected regions. Applies only to the "REG" column, which is fixed and cannot be modified.
sector_select	Optional character vector. Filters data to selected sectors. Applies only to the "ACTS" and "COMM" columns, which are fixed and cannot be modified.
subtotal_level	Logical. If TRUE, includes subtotal rows. Default is FALSE.

plot_data	Logical. If TRUE, generates plot-ready data and assigns to specified variable names.
output_formats	Character vector or list. Output formats for export. Valid values: "csv", "stata", "rds", "txt".
sl4_output_name	Character. Variable name to assign SL4 output. Default: "sl4.plot.data".
har_output_name	Character. Variable name to assign HAR output. Default: "har.plot.data".
macro_output_name	Character. Variable name to assign macro output. Default: "GTAPMacro".
add_scenario_ranking	Logical or "merged". Adds a numeric index for each scenario: • TRUE: Adds a ranking column. • "merged": Also prefixes experiment names with the rank.
rank_column	Character. Name of the ranking column. Default is "ScenarioRank".

Details

- To prepare data for plotting and generating tables within the GTAPViz package, the "Unit" column must be included in the output.
- When using the extraction method "group_data_by_dims", the corresponding priority list must be defined via the sl4_priority or har_priority argument. See [group_data_by_dims](#) for more details.

Value

A processed GTAP-formatted dataset with standardized structure and metadata, ready for analysis or visualization.

Author(s)

Pattawee Puangchit

See Also

[add_mapping_info](#), [convert_units](#), [rename_value](#)

Examples

```
# Input Path:
input_path <- system.file("extdata/in", package = "GTAPViz")

# GTAP Macro Variables from 2 .sl4 Files named (EXP1, EXP2)
# Note: No need to add .sl4 to the experiment name
gtap_data <- auto_gtap_data(experiment = c("EXP1", "EXP2"),
                           har_suffix = "-WEL",
                           input_path = input_path, subtotal_level = FALSE,
                           process_sl4_vars = NULL, process_har_vars = NULL,
                           mapping_info = "GTAPv7", plot_data = TRUE)
```

comparison_plot

*Create Comparative Bar Charts from HAR and SL4 Data***Description**

Generates comparative bar charts using GTAP data. Supports panel facets, split-by grouping, and fully customizable styling and export options.

Input Data**Usage**

```
comparison_plot(
  data,
  filter_var = NULL,
  x_axis_from,
  split_by = "Variable",
  panel_var = "Experiment",
  variable_col = "Variable",
  unit_col = "Unit",
  desc_col = "Description",
  invert_axis = FALSE,
  separate_figure = FALSE,
  var_name_by_description = FALSE,
  add_var_info = FALSE,
  output_path = NULL,
  export_picture = TRUE,
  export_as_pdf = FALSE,
  export_config = NULL,
  plot_style_config = NULL
)
```

Arguments

data	A data frame or list of data frames containing GTAP results.
filter_var	NULL, a vector, a data frame, or a named list specifying filtering conditions. For example: <code>list(Variable = c("EV", "qgdp"), REG = c("USA", "THA"))</code> .
x_axis_from	Character. Column name used for the x-axis.
split_by	Character or vector. - Column(s) used to split plots by (e.g., "REG" or c("COMM", "REG")). - If NULL, a single aggregated plot is generated.
panel_var	Character. Column for panel facets. Default is "Experiment".
variable_col	Character. Column name for variable codes. Default is "Variable".
unit_col	Character. Column name for units. Default is "Unit".

desc_col	Character. Column name for variable descriptions. Default is "Description".
Plot Behavior	
invert_axis	Logical. If TRUE, flips the plot orientation (horizontal bars). Default is FALSE.
separate_figure	Logical. If TRUE, generates a separate plot for each value in panel_var. Default is FALSE.
Variable Display	
var_name_by_description	Logical. If TRUE, uses descriptions instead of variable codes in titles. Default is FALSE.
add_var_info	Logical. If TRUE, appends variable codes in parentheses after the description. Default is FALSE.
Export Settings	
output_path	Character. Directory to save plots. If NULL, plots are returned but not saved.
export_picture	Logical. If TRUE, exports plots as PNG images. Default is TRUE.
export_as_pdf	Logical or "merged". • FALSE (default): disables PDF export. • TRUE: exports each plot as a separate PDF file. • "merged": combines all plots into a single PDF file.
export_config	List. Export options including dimensions, DPI, and background. See create_export_config or get_all_config .
Styling	
plot_style_config	List. Custom plot appearance settings. See create_plot_style or get_all_config .

Details

Please refer to the full plot

Value

A ggplot object or a named list of ggplot objects depending on the separate_figure setting. If export_picture or export_as_pdf is enabled, the plots are also saved to output_path.

Author(s)

Pattawee Puangchit

See Also

[get_all_config](#), [detail_plot](#), [stack_plot](#), [create_title_format](#)

Examples

```
# Load data
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))
reg_data <- sl4.plot.data[["REG"]]

# Generate plot
plotA <- comparison_plot(
  data = reg_data,
  filter_var = list(Region = "Oceania", Variable = "qgdp"),
  x_axis_from = "Region",
  split_by = "Variable",
  panel_var = "Experiment",
  variable_col = "Variable",
  unit_col = "Unit",
  desc_col = "Description",

  invert_axis = FALSE,
  separate_figure = FALSE,

  var_name_by_description = FALSE,
  add_var_info = FALSE,

  output_path = NULL,
  export_picture = FALSE,
  export_as_pdf = FALSE,
  export_config = create_export_config(width = 20, height = 12),

  plot_style_config = create_plot_style(
    color_tone = "purdue",
    add_unit_to_title = TRUE,
    title_format = create_title_format(
      type = "prefix",
      text = "Impact on"
    ),
  ),
  panel_rows = 2
)
)
```

Description

Converts values in a dataset to different units based on predefined transformations or custom scaling. Supports manual and automatic conversions for economic and trade-related metrics.

Usage

```

convert_units(
  data,
  change_unit_from = NULL,
  change_unit_to = NULL,
  adjustment = NULL,
  value_col = "Value",
  unit_col = "Unit",
  variable_select = NULL,
  variable_col = "Variable",
  scale_auto = NULL
)

```

Arguments

<code>data</code>	A data structure (list, data frame, or nested combination).
<code>change_unit_from</code>	Character vector. Units to be converted (case-insensitive).
<code>change_unit_to</code>	Character vector. Target units corresponding to <code>change_unit_from</code> .
<code>adjustment</code>	Character or numeric vector. Specifies conversion operations (e.g., <code>"/1000"</code> to convert million to billion).
<code>value_col</code>	Character. Column name containing values to adjust (default: <code>"Value"</code>).
<code>unit_col</code>	Character. Column name containing unit information (default: <code>"Unit"</code>).
<code>variable_select</code>	Optional character vector. If provided, only these variables are converted.
<code>variable_col</code>	Character. Column name containing variable identifiers (default: <code>"Variable"</code>).
<code>scale_auto</code>	Optional character vector of predefined conversion rules: <code>"mil2bil"</code>: Converts million USD to billion USD (divides by 1000). <code>"bil2mil"</code>: Converts billion USD to million USD (multiplies by 1000). <code>"pct2frac"</code>: Converts percent to fraction (divides by 100). <code>"frac2pct"</code>: Converts fraction to percent (multiplies by 100).

Details

If both `change_unit_from` and `scale_auto` are provided, the function prompts the user to choose between manual and automatic conversion.

Value

A data structure with values converted to the specified units.

Author(s)

Pattawee Puangchit

See Also

[add_mapping_info](#), [rename_value](#), [sort_plot_data](#)

Examples

```
# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

# Convert million USD to billion USD
gtap_data <- convert_units(sl4.plot.data,
  change_unit_from = "million USD",
  change_unit_to = "billion USD",
  adjustment = "/1000"
)

# Automatic conversion from percent to fraction
gtap_data <- convert_units(sl4.plot.data, scale_auto = "pct2frac")
```

create_export_config *Create an Export Configuration*

Description

Creates a configuration list for controlling plot export settings. This function provides auto-completion for export options.

Usage

```
create_export_config(
  file_name = NULL,
  width = NULL,
  height = NULL,
  dpi = 300,
  bg = "white",
  limitsize = FALSE
)
```

Arguments

file_name	Character. Base name for exported files. Default: "gtap_plots".
width	Numeric. Width of output in inches. Default: NULL (auto-calculated).
height	Numeric. Height of output in inches. Default: NULL (auto-calculated).
dpi	Numeric. Resolution for PNG export. Default: 300.
bg	Character. Background color. Default: "white".
limitsize	Logical. Whether to limit size. Default: FALSE.

Value

A list with export configuration parameters.

Author(s)

Pattawee Puangchit

Examples

```
# Default export configuration
default_export <- create_export_config()

# Custom export configuration
custom_export <- create_export_config(
  file_name = "regional_impacts",
  width = 12,
  height = 8,
  dpi = 600
)
```

create_plot_style	<i>Create a Plot Style Configuration</i>
-------------------	--

Description

Creates a configuration list for plot styling that can be used with GTAPViz plotting functions. This function provides auto-completion for style options while maintaining compatibility with direct list specification.

Usage

```
create_plot_style(
  show_title = TRUE,
  title_face = "bold",
  title_size = 20,
  title_hjust = 0.5,
  add_unit_to_title = TRUE,
  title_margin = c(10, 0, 10, 0),
  title_format = list(type = "standard", text = "", sep = ""),
  show_x_axis_title = TRUE,
  x_axis_title_face = "bold",
  x_axis_title_size = 16,
  x_axis_title_margin = c(25, 25, 0, 0),
  show_x_axis_labels = TRUE,
  x_axis_text_face = "plain",
  x_axis_text_size = 14,
  x_axis_text_angle = 0,
  x_axis_text_hjust = 0,
```

```
x_axis_description = "",
show_y_axis_title = TRUE,
y_axis_title_face = "bold",
y_axis_title_size = 16,
y_axis_title_margin = c(25, 25, 0, 0),
show_y_axis_labels = TRUE,
y_axis_text_face = "plain",
y_axis_text_size = 14,
y_axis_text_angle = 0,
y_axis_text_hjust = 0,
y_axis_description = "",
show_axis_titles_on_all_facets = TRUE,
show_value_labels = TRUE,
value_label_face = "plain",
value_label_size = 5,
value_label_position = "above",
value_label_decimal_places = 2,
show_legend = FALSE,
show_legend_title = FALSE,
legend_position = "bottom",
legend_title_face = "bold",
legend_text_face = "plain",
legend_text_size = 14,
strip_face = "bold",
strip_text_size = 16,
strip_background = "lightgrey",
strip_text_margin = c(10, 0, 10, 0),
panel_spacing = 2,
panel_rows = NULL,
panel_cols = NULL,
theme = NULL,
color_tone = NULL,
color_palette_type = "qualitative",
positive_color = "#2E8B57",
negative_color = "#CD5C5C",
background_color = "white",
grid_color = "grey90",
show_grid_major_x = FALSE,
show_grid_major_y = FALSE,
show_grid_minor_x = FALSE,
show_grid_minor_y = FALSE,
show_zero_line = TRUE,
zero_line_type = "dashed",
zero_line_color = "black",
zero_line_size = 0.5,
zero_line_position = 0,
bar_width = 0.9,
bar_spacing = 0.9,
```

```

    scale_limit = NULL,
    scale_increment = NULL,
    expansion_y_mult = c(0.05, 0.1),
    expansion_x_mult = c(0.05, 0.05),
    all_font_size = 1,
    sort_data_by_value = FALSE,
    plot.margin = c(10, 25, 10, 10)
)

```

Arguments

<code>show_title</code>	Logical. Show or hide the plot title. Default: TRUE
<code>title_face</code>	Character. Font face ("bold", "plain", "italic"). Default: "bold"
<code>title_size</code>	Numeric. Font size of title. Default: 20
<code>title_hjust</code>	Numeric. Horizontal alignment (0 = left, 1 = right). Default: 0.5
<code>add_unit_to_title</code>	Logical. Append unit to title if applicable. Default: TRUE
<code>title_margin</code>	Numeric vector c(top, right, bottom, left). Default: c(10, 0, 10, 0)
<code>title_format</code>	List or function output. Title formatting options. Can be created with <code>create_title_format()</code> . Default: list(type = "standard", text = "", sep = "")
<code>show_x_axis_title</code>	Logical. Show or hide x-axis title. Default: TRUE
<code>x_axis_title_face</code>	Character. Font face for x-axis title. Default: "bold"
<code>x_axis_title_size</code>	Numeric. Font size of x-axis title. Default: 16
<code>x_axis_title_margin</code>	Numeric vector c(top, right, bottom, left). Default: c(25, 25, 0, 0)
<code>show_x_axis_labels</code>	Logical. Show or hide x-axis labels. Default: TRUE
<code>x_axis_text_face</code>	Character. Font face for x-axis labels. Default: "plain"
<code>x_axis_text_size</code>	Numeric. Font size of x-axis labels. Default: 14
<code>x_axis_text_angle</code>	Numeric. Angle of x-axis labels. Default: 0
<code>x_axis_text_hjust</code>	Numeric. Horizontal justification of x-axis labels. Default: 0
<code>x_axis_description</code>	Character. Optional description for the x-axis. Default: ""
<code>show_y_axis_title</code>	Logical. Show or hide y-axis title. Default: TRUE
<code>y_axis_title_face</code>	Character. Font face for y-axis title. Default: "bold"

`y_axis_title_size`
Numeric. Font size of y-axis title. Default: 16

`y_axis_title_margin`
Numeric vector `c(top, right, bottom, left)`. Default: `c(25, 25, 0, 0)`

`show_y_axis_labels`
Logical. Show or hide y-axis labels. Default: TRUE

`y_axis_text_face`
Character. Font face for y-axis labels. Default: "plain"

`y_axis_text_size`
Numeric. Font size of y-axis labels. Default: 14

`y_axis_text_angle`
Numeric. Angle of y-axis labels. Default: 0

`y_axis_text_hjust`
Numeric. Horizontal justification of y-axis labels. Default: 0

`y_axis_description`
Character. Optional description for the y-axis. Default: ""

`show_axis_titles_on_all_facets`
Logical. Show axis titles on all facets. Default: TRUE

`show_value_labels`
Logical. Show or hide value labels. Default: TRUE

`value_label_face`
Character. Font face for value labels. Default: "plain"

`value_label_size`
Numeric. Font size of value labels. Default: 5

`value_label_position`
Character. Position of value labels ("above", "outside", "top"). Default: "above"

`value_label_decimal_places`
Numeric. Number of decimal places in value labels. Default: 2

`show_legend`
Logical. Show or hide legend. Default: FALSE

`show_legend_title`
Logical. Show or hide legend title. Default: FALSE

`legend_position`
Character. Legend position ("none", "bottom", "right"). Default: "bottom"

`legend_title_face`
Character. Font face for legend title. Default: "bold"

`legend_text_face`
Character. Font face for legend text. Default: "plain"

`legend_text_size`
Numeric. Font size of legend text. Default: 14

`strip_face`
Character. Font face for panel strip. Default: "bold"

`strip_text_size`
Numeric. Font size for panel strip. Default: 16

`strip_background`
Character. Background color of strip. Default: "lightgrey"

strip_text_margin	Numeric vector c(top, right, bottom, left). Default: c(10, 0, 10, 0)
panel_spacing	Numeric. Spacing between panels. Default: 2
panel_rows	Numeric or NULL. Number of rows in panel layout. Default: NULL
panel_cols	Numeric or NULL. Number of columns in panel layout. Default: NULL
theme	ggplot2 theme or NULL. Custom ggplot theme. Default: NULL
color_tone	Character or NULL. Base color theme. Default: NULL
color_palette_type	Character. Type of color palette ('qualitative', 'sequential', 'diverging'). Default: "qualitative"
positive_color	Character. Color for positive values. Default: "#2E8B57"
negative_color	Character. Color for negative values. Default: "#CD5C5C"
background_color	Character. Background color of plot. Default: "white"
grid_color	Character. Color of grid lines. Default: "grey90"
show_grid_major_x	Logical. Show major grid lines on x-axis. Default: FALSE
show_grid_major_y	Logical. Show major grid lines on y-axis. Default: FALSE
show_grid_minor_x	Logical. Show minor grid lines on x-axis. Default: FALSE
show_grid_minor_y	Logical. Show minor grid lines on y-axis. Default: FALSE
show_zero_line	Logical. Show or hide zero line. Default: TRUE
zero_line_type	Character. Line type ("solid", "dashed", "dotted"). Default: "dashed"
zero_line_color	Character. Color of zero line. Default: "black"
zero_line_size	Numeric. Line thickness of zero line. Default: 0.5
zero_line_position	Numeric. Position of the zero line. Default: 0
bar_width	Numeric. Width of bars. Default: 0.9
bar_spacing	Numeric. Spacing between groups of bars. Default: 0.9
scale_limit	Numeric vector of length 2 or NULL. Manual limits for value axis. Default: NULL
scale_increment	Numeric or NULL. Step size for axis tick marks. Default: NULL
expansion_y_mult	Numeric vector. Y-axis expansion. Default: c(0.05, 0.1)
expansion_x_mult	Numeric vector. X-axis expansion. Default: c(0.05, 0.05)
all_font_size	Numeric. Master control for all font sizes. Default: 1
sort_data_by_value	Logical. Whether to sort data by value. Default: FALSE
plot.margin	Numeric vector c(top, right, bottom, left). Margins around the entire plot. Default: c(10, 25, 10, 10)

Value

A list containing all plot style configuration parameters

Author(s)

Pattawee Puangchit

Examples

```
# Create customized style with title formatting
custom_style <- create_plot_style(
  color_tone = "gtap",
  title_size = 24,
  title_format = create_title_format(
    type = "prefix",
    text = "Impact on",
    sep = "-"
  ),
  bar_width = 0.5,
  x_axis_text_angle = 45
)
```

create_title_format	Create a Title Format Configuration
---------------------	-------------------------------------

Description

Creates a configuration list for controlling plot title formatting. Supports auto-completion for common title format types.

Usage

```
create_title_format(type = "standard", text = "", sep = NULL)
```

Arguments

type	Character. Title format type: "standard": Default (variable + description + unit) "prefix": Adds text before the automatic title "suffix": Adds text after the automatic title "full": Uses only the specified text as the title "dynamic": Builds a title using column values
text	Character. Text content used for prefix, suffix, full, or a template for dynamic.
sep	Character. The separator between components (only used in "prefix" or "suffix" mode). Default is ": ".

Value

A list with title format configuration parameters.

Author(s)

Pattawee Puangchit

Examples

```
# Standard auto-generated title
standard_title <- create_title_format()

# Prefix title
prefix_title <- create_title_format(
  type = "prefix",
  text = "Impact on",
  sep = " "
)

# Dynamic title using column values
dynamic_title <- create_title_format(
  type = "dynamic",
  text = "Impact on {Variable} in {Region}"
)
```

detail_plot	Create Comprehensive Bar Charts from HAR and SLA Data
-------------	---

Description

Generates detailed bar charts to visualize the distribution of impacts across multiple dimensions. Supports top impact filtering, color coding, and fully customizable styling and export options.

Input Data

Usage

```
detail_plot(
  data,
  filter_var = NULL,
  x_axis_from,
  split_by = "Variable",
  panel_var = "Experiment",
  variable_col = "Variable",
  unit_col = "Unit",
  desc_col = "Description",
  invert_axis = TRUE,
  separate_figure = FALSE,
  top_impact = NULL,
```

```

var_name_by_description = FALSE,
add_var_info = FALSE,
output_path = NULL,
export_picture = TRUE,
export_as_pdf = FALSE,
export_config = NULL,
plot_style_config = NULL
)

```

Arguments

data	A data frame or list of data frames containing GTAP results.
filter_var	NULL, a vector, a data frame, or a named list specifying filtering conditions. For example: <code>list(Variable = c("EV", "qgdp"), REG = c("USA", "THA"))</code> .
x_axis_from	Character. Column name used for the x-axis.
split_by	Character or vector. - Column(s) used to split plots by (e.g., "REG" or <code>c("COMM", "REG")</code>). - If NULL, a single aggregated plot is generated.
panel_var	Character. Column for panel facets. Default is "Experiment".
variable_col	Character. Column name for variable codes. Default is "Variable".
unit_col	Character. Column name for units. Default is "Unit".
desc_col	Character. Column name for variable descriptions. Default is "Description".

Plot Behavior

invert_axis	Logical. If TRUE, flips the plot orientation (horizontal bars). Default is FALSE.
separate_figure	Logical. If TRUE, generates a separate plot for each value in panel_var. Default is FALSE.
top_impact	Numeric or NULL. If specified, shows only the top N impactful values; NULL shows all.

Variable Display

var_name_by_description	Logical. If TRUE, uses descriptions instead of variable codes in titles. Default is FALSE.
add_var_info	Logical. If TRUE, appends variable codes in parentheses after the description. Default is FALSE.

Export Settings

output_path	Character. Directory to save plots. If NULL, plots are returned but not saved.
export_picture	Logical. If TRUE, exports plots as PNG images. Default is TRUE.
export_as_pdf	Logical or "merged". - FALSE (default): disables PDF export. - TRUE: exports each plot as a separate PDF file. "merged": combines all plots into a single PDF file.

`export_config` List. Export options including dimensions, DPI, and background. See [create_export_config](#) or [get_all_config](#).

Styling

`plot_style_config` List. Custom plot appearance settings. See [create_plot_style](#) or [get_all_config](#).

Value

A ggplot object or a named list of ggplot objects depending on the `separate_figure` setting. If `export_picture` or `export_as_pdf` is enabled, the plots are also saved to `output_path`.

Author(s)

Pattawee Puangchit

See Also

[comparison_plot](#), [stack_plot](#)

Examples

```
# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

# Prepare Dataframe
sector_data <- sl4.plot.data[["COMM*REG"]]

# Plot
plotB <- detail_plot(
  # === Input Data ===
  data          = sector_data,
  filter_var    = list(Region = "Oceania"),
  x_axis_from   = "Commodity",
  split_by      = "Region",
  panel_var     = "Experiment",
  variable_col  = "Variable",
  unit_col      = "Unit",
  desc_col      = "Description",

  # === Plot Behavior ===
  invert_axis    = TRUE,
  separate_figure = FALSE,
  top_impact     = NULL,

  # === Variable Display ===
  var_name_by_description = TRUE,
  add_var_info            = FALSE,

  # === Export Settings ===
  output_path    = NULL,
  export_picture = FALSE,
```

```

export_as_pdf = FALSE,
export_config = create_export_config(width = 45, height = 20),

# === Styling ===
plot_style_config = create_plot_style(
  positive_color = "#2E8B57",
  negative_color = "#CD5C5C",
  panel_rows = 1,
  panel_cols = NULL,
  show_axis_titles_on_all_facets = FALSE,
  y_axis_text_size = 25,
  bar_width = 0.6,
  all_font_size = 1.1
)
)

```

get_all_config

*Print Plot and Export Configuration Snippets***Description**

Retrieve full configuration code as a list for applying in the plot styling and export settings.

Usage

```

get_all_config(
  plot_style = "default",
  plot_config = TRUE,
  export_config = TRUE
)

```

Arguments

plot_style	Character. Plot style to use (currently only "default" is supported).
plot_config	Logical. If 'TRUE', prints the plot style configuration.
export_config	Logical. If 'TRUE', prints the export configuration.

Details

Once printing into the console, users can simply copy and paste the entire list of configurations, rename it (if needed), and use it in your plot functions directly.

Value

A named list containing the current default values for all GTAPViz configuration options, including plot styles, table formats, and export parameters.

Author(s)

Pattawee Puangchit

Examples

```
# Input Path:
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

# Retrive configurations
get_all_config()
```

get_color_palette	<i>Print and Visualize Themed Color Palettes</i>
-------------------	--

Description

Prints and visualizes predefined color palettes used in GTAPViz. Use ‘color_tone = "all"’ to return a list of callable palette functions.

Usage

```
get_color_palette(color_tone = NULL, palette_type = "qualitative")
```

Arguments

color_tone	Character. Name of the color theme to display (e.g., "gtap", "winter", "fall", or "all").
palette_type	Character. Palette type: "qualitative" (default), "sequential", or "diverging".

Value

A character vector of hex color codes representing the selected color palette. If ‘color_tone = "all"’, returns a list of functions, each generating a specific palette. If ‘color_tone = "list"’, returns a character vector of available palette names.

Author(s)

Pattawee Puangchit

Examples

```
# Get all palettes as callable functions
all_palettes <- get_color_palette("all")
all_palettes$winter()
all_palettes$gtap()

# Visualize specific palettes
get_color_palette("fall", "sequential")
get_color_palette("academic", "diverging")
```

`pivot_table_with_filter`

Export Data as an Excel Pivot Table

Description

Exports a dataset to an Excel file with both raw data and a generated pivot table.

Usage

```
pivot_table_with_filter(
  data,
  filter = NULL,
  rows = NULL,
  cols = NULL,
  data_fields = "Value",
  raw_sheet_name = "RawData",
  pivot_sheet_name = "PivotTable",
  dims = "A4",
  export_table = FALSE,
  output_path = NULL,
  workbook_name = "GTAP_PivotTable.xlsx"
)
```

Arguments

<code>data</code>	Data frame. The dataset to be exported.
<code>filter</code>	Character vector (optional). Columns to be used as filter fields in the pivot table.
<code>rows</code>	Character vector (optional). Columns to be used as row fields in the pivot table.
<code>cols</code>	Character vector (optional). Columns to be used as column fields in the pivot table.
<code>data_fields</code>	Character. The data field(s) to be summarized in the pivot table (default: "Value").
<code>raw_sheet_name</code>	Character. Name of the sheet containing raw data (default: "RawData").
<code>pivot_sheet_name</code>	Character. Name of the sheet containing the pivot table (default: "PivotTable").

<code>dims</code>	Character. Cell reference where the pivot table starts (default: "A3").
<code>export_table</code>	Logical. Whether to save the Excel file (default: TRUE).
<code>output_path</code>	Character. Directory where the file should be saved (default: current working directory).
<code>workbook_name</code>	Character. Name of the output Excel file (default: "GTAP_PivotTable.xlsx").

Details

This function creates an Excel workbook with:

- A raw data sheet (`raw_sheet_name`) containing the provided dataset.
- A pivot table sheet (`pivot_sheet_name`) generated based on specified row, column, and data fields.

If `export = TRUE`, the function saves the workbook to the specified `output_path`.

Value

An excel workbook object containing both raw data and the pivot table.

Author(s)

Pattawee Puangchit

Examples

```
# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

data_pivot_table <- sl4.plot.data[["REG"]]

# Generate Pivot Table with Filter
# Only use columns that exist in the data
pivot_table_with_filter(

  # === Input & Filter Settings ===
  data = data_pivot_table,
  filter = c("Variable", "Unit"), # Allow filtering by variable type and unit

  # === Pivot Structure ===
  rows = c("Region"),             # Rows: Regions (removed "Sector" which doesn't exist)
  cols = c("Experiment"),         # Columns: Experiments
  data_fields = "Value",          # Values to be aggregated

  # === Sheet & Layout ===
  raw_sheet_name = "Raw_Data",    # Sheet name for raw data
  pivot_sheet_name = "Sector_Pivot", # Sheet name for pivot table
  dims = "A3",                   # Starting cell for pivot table

  # === Export Options ===
```



```

export_table = FALSE,
output_path = NULL,
workbook_name = "Sectoral_Impact_Analysis.xlsx"
)

```

rename_value

Rename Values in a Column

Description

Replaces specific values in a column based on a provided mapping file. Supports renaming across nested data structures and preserves factor levels.

Usage

```
rename_value(data, column_name = NULL, mapping.file)
```

Arguments

data	Data structure (data frame, list, or nested combination).
column_name	Character. Column to modify. If 'NULL', the function extracts it from 'mapping.file'.
mapping.file	Data frame with "OldName" and "NewName" columns for renaming.

Value

The same data structure with specified values replaced.

Author(s)

Pattawee Puangchit

See Also

[add_mapping_info](#), [convert_units](#), [sort_plot_data](#)

Examples

```

# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
har.plot.data <- readRDS(file.path(input_path, "har.plot.data.rds"))

# Rename variables in a dataset
mapping_welfare <- data.frame(
  ColumnName = "COLUMN",
  OldName = c("alloc_A1", "ENDWB1", "tech_C1", "pop_D1", "pref_G1", "tot_E1", "IS_F1"),
  NewName = c("Alloc Eff.", "Endwb", "Tech Chg.", "Pop", "Perf", "ToT", "I-S"),
  stringsAsFactors = FALSE
)

```

```
)

har.plot.data <- rename_value(har.plot.data, mapping.file = mapping_welfare)
```

report_table

Generate a Structured Report Table

Description

Transforms multiple datasets into wide-format tables based on defined pivot columns, hierarchical grouping, and renaming rules. Supports optional subtotal filtering and exporting to Excel.

Usage

```
report_table(
  data_list,
  pivot_col,
  total_column = FALSE,
  export_table = FALSE,
  separate_file = FALSE,
  output_path = NULL,
  sheet_names = NULL,
  include_units = FALSE,
  component_exclude = NULL,
  group_by = NULL,
  rename_cols = NULL,
  var_name_by_description = TRUE,
  add_var_info = FALSE,
  decimal = 2,
  unit_select = NULL,
  separate_sheet_by = NULL,
  subtotal_level = FALSE,
  repeat_label = FALSE,
  workbook_name = "detail_results",
  add_group_line = FALSE
)
```

Arguments

data_list	A named list of data frames to process.
pivot_col	A named list specifying the column to pivot into a wide format for each dataset. Each dataset can have only one pivot column. Example: <code>pivot_col = list(A = "COLUMN", E1 = "PRICES")</code>
total_column	Logical. If TRUE, adds a "Total" column summing numeric values.
export_table	Logical. If TRUE, saves the output as an Excel file.
separate_file	Logical. If TRUE, saves each dataset as a separate Excel file.

output_path	Character. Directory for saving Excel files when export_table = TRUE.
sheet_names	Optional named list for custom sheet names.
include_units	Logical. If TRUE, includes "Unit" as a grouping column if applicable.
component_exclude	Optional character vector specifying pivoted values to exclude.
group_by	A named list defining hierarchical grouping for each dataset. The order of columns in each list determines the priority. Example: group_by = list(A = list("Experiment", "REG"), E1 = list("Experiment", "REG", "COMM"))
rename_cols	A named list for renaming columns across all datasets. Example: rename_cols = list("REG" = "Region", "COMM" = "Commodities", "Experiment" = "Scenario")
var_name_by_description	Logical. If TRUE, replaces variable codes with descriptions when available.
add_var_info	Logical. If TRUE, appends variable codes in parentheses after descriptions.
decimal	Numeric. Number of decimal places for rounding values.
unit_select	Optional character. Specifies a unit to filter the dataset.
separate_sheet_by	Optional column name to split sheets in Excel. If defined, each unique value in the specified column gets its own sheet. Example: separate_sheet_by = "Scenario".
subtotal_level	Logical. If TRUE, includes all subtotal values; otherwise, keeps only TOTAL rows.
repeat_label	Logical. If TRUE, repeats the first group column in exports for clarity.
workbook_name	Character. Name of the Excel workbook (without extension).
add_group_line	Logical. If TRUE, adds a thin line after each group in the exported table.

Details

This function requires a data list and can generate multiple output tables in a single setup. That is, all data frames within the list can be processed simultaneously. See the example for how to generate two data frames at once from the data list `sl4.plot.data`, which is obtained via `auto_gtap_data(plot_data = TRUE)`.

Value

If `export_table = TRUE`, tables are saved as Excel files.

Author(s)

Pattawee Puangchit

See Also

[add_mapping_info](#), [convert_units](#), [rename_value](#), [pivot_table_with_filter](#)

Examples

```
# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
sl4.plot.data <- readRDS(file.path(input_path, "sl4.plot.data.rds"))

report_table(
  data_list = sl4.plot.data,

  # === Table Structure ===
  pivot_col = list(
    REG = "Variable",
    "COMM*REG" = "Commodity"
  ),
  group_by = list(
    REG = list("Experiment", "Region"),
    "COMM*REG" = list("Experiment", "Variable", "Region")
  ),
  rename_cols = list("Experiment" = "Scenario"),

  # === Table Layout & Labels ===
  total_column = FALSE,
  decimal = 4,
  subtotal_level = FALSE,
  repeat_label = FALSE,
  include_units = TRUE,
  var_name_by_description = TRUE,
  add_var_info = TRUE,
  add_group_line = FALSE,

  # === Export Options ===
  separate_sheet_by = "Unit",
  export_table = FALSE,
  output_path = NULL,
  separate_file = FALSE,
  workbook_name = "Comparison Table Default"
)
```

sort_plot_data

Sort GTAP Plot Data

Description

Sorts data frames in a GTAP plot list structure based on specified column orders. Works with data frames, lists of data frames, or nested data structures.

Usage

```
sort_plot_data(
```


stack_plot

*Create Stacked Bar Charts for Decomposition Analysis***Description**

Generates stacked bar charts to visualize value compositions across multiple dimensions. Supports both stacked and unstacked layouts for decomposition analysis, with full control over grouping, faceting, top-impact filtering, and export styling.

Input Data**Usage**

```
stack_plot(
  data,
  filter_var = NULL,
  x_axis_from,
  stack_value_from,
  split_by = NULL,
  panel_var = "Experiment",
  variable_col = "Variable",
  unit_col = "Unit",
  desc_col = "Description",
  invert_axis = FALSE,
  separate_figure = FALSE,
  show_total = TRUE,
  unstack_plot = FALSE,
  top_impact = NULL,
  var_name_by_description = FALSE,
  add_var_info = FALSE,
  output_path = NULL,
  export_picture = TRUE,
  export_as_pdf = FALSE,
  export_config = NULL,
  plot_style_config = NULL
)
```

Arguments

data	A data frame or list of data frames containing GTAP results.
filter_var	NULL, a vector, a data frame, or a named list specifying filtering conditions. For example: <code>list(Variable = c("EV", "qgdp"), REG = c("USA", "THA"))</code> .
x_axis_from	Character. Column name used for the x-axis.
stack_value_from	Character. Column containing stack component categories (e.g., "COMM" for commodities).
split_by	Character or vector.

	• Column(s) used to split plots by (e.g., "REG" or c("COMM", "REG")). • If NULL, a single aggregated plot is generated.
panel_var	Character. Column for panel facets. Default is "Experiment".
variable_col	Character. Column name for variable codes. Default is "Variable".
unit_col	Character. Column name for units. Default is "Unit".
desc_col	Character. Column name for variable descriptions. Default is "Description".
Plot Behavior	
invert_axis	Logical. If TRUE, flips the plot orientation (horizontal bars). Default is FALSE.
separate_figure	Logical. If TRUE, generates a separate plot for each value in panel_var. Default is FALSE.
show_total	Logical. If TRUE, displays total values above stacked bars. Default is TRUE.
unstack_plot	Logical. If TRUE, creates separate bar plots for each x_axis_from value instead of stacking. Default is FALSE.
top_impact	Numeric or NULL. If specified, shows only the top N impactful values; NULL shows all.
Variable Display	
var_name_by_description	Logical. If TRUE, uses descriptions instead of variable codes in titles. Default is FALSE.
add_var_info	Logical. If TRUE, appends variable codes in parentheses after the description. Default is FALSE.
Export Settings	
output_path	Character. Directory to save plots. If NULL, plots are returned but not saved.
export_picture	Logical. If TRUE, exports plots as PNG images. Default is TRUE.
export_as_pdf	Logical or "merged". • FALSE (default): disables PDF export. • TRUE: exports each plot as a separate PDF file. • "merged": combines all plots into a single PDF file.
export_config	List. Export options including dimensions, DPI, and background. See create_export_config or get_all_config .
Styling	
plot_style_config	List. Custom plot appearance settings. See create_plot_style or get_all_config .

Value

A ggplot object or a named list of ggplot objects depending on the separate_figure setting. If export_picture or export_as_pdf is enabled, the plots are also saved to output_path.

Author(s)

Pattawee Puangchit

See Also

[comparison_plot](#), [detail_plot](#)

Examples

```
# Load Data:
input_path <- system.file("extdata/in", package = "GTAPViz")
har.plot.data <- readRDS(file.path(input_path, "har.plot.data.rds"))

# Prepare Dataframe
welfare.decomp <- har.plot.data[["A"]]

# Plot
plotC <- stack_plot(
  # === Input Data ===
  data          = welfare.decomp,
  filter_var    = list(Region = "Oceania"),
  x_axis_from   = "Region",
  stack_value_from = "COLUMN",
  split_by      = FALSE,
  panel_var     = "Experiment",
  variable_col  = "Variable",
  unit_col      = "Unit",
  desc_col      = "Description",

  # === Plot Behavior ===
  invert_axis    = FALSE,
  separate_figure = FALSE,
  show_total     = TRUE,
  unstack_plot   = FALSE,
  top_impact     = NULL,

  # === Variable Display ===
  var_name_by_description = TRUE,
  add_var_info            = FALSE,

  # === Export Settings ===
  output_path    = NULL,
  export_picture = FALSE,
  export_as_pdf  = FALSE,
  export_config  = create_export_config(width = 28, height = 15),

  # === Styling ===
  plot_style_config = create_plot_style(
    color_tone      = "gtap",
    panel_rows      = 2,
    panel_cols      = NULL,
    show_legend     = TRUE,
    show_axis_titles_on_all_facets = FALSE
  )
)
```


Index

`add_mapping_info`, [2](#), [6](#), [11](#), [25](#), [27](#), [29](#)
`add_mapping_info()`, [5](#)
`auto_gtap_data`, [4](#)

`comparison_plot`, [7](#), [20](#), [32](#)
`convert_units`, [3](#), [5](#), [6](#), [9](#), [25](#), [27](#), [29](#)
`create_export_config`, [8](#), [11](#), [20](#), [31](#)
`create_plot_style`, [8](#), [12](#), [20](#), [31](#)
`create_title_format`, [8](#), [17](#)

`detail_plot`, [8](#), [18](#), [32](#)

`get_all_config`, [8](#), [20](#), [21](#), [31](#)
`get_color_palette`, [22](#)
`group_data_by_dims`, [6](#)

`pivot_table_with_filter`, [23](#), [27](#)

`rename_value`, [3](#), [6](#), [11](#), [25](#), [27](#), [29](#)
`report_table`, [26](#)

`sort_plot_data`, [11](#), [25](#), [28](#)
`stack_plot`, [8](#), [20](#), [30](#)