Package 'aftables'

April 22, 2026

Title Create Spreadsheet Publications Following Best Practice

Version 2.0.1

Description Generate spreadsheet publications that follow best practice guidance from the UK government's Analysis Function, available at <https://analysisfunction.civilservice.gov.uk/policy-store/releasing-statistics-in-spreadsheets/>, with a focus on accessibility. See also the 'Python' package 'gptables'.

License MIT + file LICENSE

URL https://best-practice-and-impact.github.io/aftables/, https://github.com/best-practice-and-impact/aftables

BugReports https://github.com/best-practice-and-impact/aftables/issues

Depends R (>= 4.1.0)

Imports openxlsx2, pillar, purrr, dplyr, stringr, tidyselect, tidyr, rlang, utils, yaml, tibble

Suggests covr, knitr, rmarkdown, rstudioapi, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

NeedsCompilation no

Author Matt Dray [aut], Tim Taylor [ctb], Matt Kerlogue [ctb], Crown Copyright [cph], Olivia Box Power [cre, ctb], Zachary Gleisner [ctb]

Maintainer Olivia Box Power <Olivia.BoxPower@dhsc.gov.uk>

Repository CRAN

Date/Publication 2026-04-22 12:10:02 UTC

Contents

`as_aftable` *Coerce To An 'aftable' Object*

Description

Functions to check if an object is an aftable, or coerce it if possible.

Usage

```
as_aftable(x)

is_aftable(x)
```

Arguments

`x`	A data.frame object to coerce.

Value

`as_aftable` returns an object of class aftable if possible. `is_aftable` returns `TRUE` if the object has class aftable, otherwise `FALSE`.

Examples

```
as_aftable(demo_df)
is_aftable(demo_aftable)
```

at_template_aftable *Insert 'create_aftable' template*

Description

Insert at the cursor a template for `create_aftable` from the 'aftables' package, pre-filled with demo data.

Usage

```
at_template_aftable()
```

Value

Empty list. Function is used for side effect.

at_template_workflow *Insert full 'aftables' template workflow*

Description

Insert at the cursor (a) demo templates for cover, contents and notes tables, and (b) a call to `create_aftable` pre-filled with demo data.

Usage

```
at_template_workflow()
```

Value

Empty list. Function is used for side effect.

create_aftable *Create An 'aftable' Object*

Description

Create a new aftable-class object, which is a special data.frame that contains all the information needed in your output spreadsheet. In turn, the object created by this function can be used to populate an 'openxlsx2' wbWorkbook-class object with the function generate_workbook.

Usage

```
create_aftable(
  tab_titles,
  sheet_types = c("cover", "contents", "notes", "tables"),
  sheet_titles,
  blank_cells = NA_character_,
  sources = NA_character_,
  custom_rows = list(NA_character_),
  tables
)
```

Arguments

tab_titles	Required character vector, one value per sheet. Each title will appear literally on each tab of the final spreadsheet output. Keep brief. Letters and numbers only; do not start with a number; use underscores for spaces. For example: 'Cover', 'Contents', 'Notes', 'Table_1'. Will be corrected automatically unless there's an error.
sheet_types	Required character vector, one value per sheet. Sheets that don't contain publication tables ('meta' sheets) should be of type 'contents', 'cover' or 'notes'. Sheets that contain statistical tables of data are type 'tables'.
sheet_titles	Required character vector, one value per sheet. The main title for each sheet, which will appear in cell A1 (top-left corner).
blank_cells	Optional character vector, one value per sheet. A short sentence to explain the reason for any blank cells in the sheet. Supply as NA_character_ if empty. Most likely to be used with sheet type 'tables'.
sources	Optional character vector, one value per sheet. The origin of the data for a given sheet. Supply as NA_character_ if empty. To be used with sheet type 'tables'.
custom_rows	Optional list of character vectors. One list element per sheet, one character vector element per row of pre-table metadata. Supply a list element as NA_character_ if empty. To be used with sheet type 'tables', but can also be used for sheet types 'contents' and 'notes'.
tables	Required list of data.frames (though the cover sheet may be supplied as a list), one per sheet. See details.

Details

How to supply data to the 'tables' argument:

Formats for the elements collected as a list and passed to the 'tables' argument, depending on the sheet type.

- Sheet type 'cover': either (a) a list where each element name is a section header and each element's content is a character vector whose elements will make up separate rows of that section (recommended), or (b) a data.frame with one row per subsection, with one column for section titles and one column for corresponding for that section's body text. For example, you may have a section with the title 'Contact details' that contains an email address and telephone number. You can use line breaks (i.e. '\n') to separate text into paragraphs.
- Sheet type 'contents': one row per sheet, two columns suggested at least (named 'Tab title' and 'Worksheet title').
- Sheet type 'notes': one row per note, two columns suggested (named 'Note number', 'Note text'), where notes are in the form '[note 1]'.
- Sheet type 'tables': a tidy, rectangular data.frame containing the data to be published. It's the user's responsibility to add notes in the form '[note 1]' to column headers, or in a special 'Notes' row.

How to supply hyperlinks:

You can provide text in Markdown link syntax (e.g. '[GOV.UK](https://www.gov.uk)', adding 'mailto:' before an email address) and the containing cell will be rendered as a hyperlink in the output spreadsheet. Note that whole cells will become hyperlinks; there is no support for selected words in a sentence to be rendered as a hyperlink.

Hyperlinks can be supplied in the character strings to three arguments:

- To the 'tables' argument for sheet type 'cover' only. It's recommended to supply the cover information as a list rather than a data.frame, which will allow you to make specific rows within a section (e.g. 'contact us') into hyperlinks.
- To the 'custom_rows' argument for sheets of type 'contents, 'notes' and 'tables'.
- To the 'source' argument for sheets of type 'table' only.

Value

An object with classes 'aftable', 'tbl' and 'data.frame'.

Examples

```
# Prepare some demo tables of information

set.seed(1066)

cover_list <- list(
  "Section 1" = c("First row of Section 1.", "Second row of Section 1."),
  "Section 2" = "The only row of Section 2.",
  "Section 3" = c(
    "[Website](https://best-practice-and-impact.github.io/aftables/)",
    "[Email address](mailto:fake.address@aftables.com)"
  )
)
```

```
contents_df <- data.frame(
  "Sheet name" = c("Notes", "Table_1", "Table_2"),
  "Sheet title" = c(
    "Notes used in this workbook",
    "First Example Sheet",
    "Second Example Sheet"
  ),
  check.names = FALSE
)

notes_df <- data.frame(
  "Note number" = paste0("[note ", 1:2, "]"),
  "Note text" = c("First note.", "Second note."),
  check.names = FALSE
)


table_1_df <- data.frame(
  Category = LETTERS[1:10],
  "Numeric [note 1]" = 1:10,
  "Numeric suppressed" = c(1:4, "[c]", 6:9, "[x]"),
  "Numeric thousands" = abs(round(rnorm(10), 4) * 1e5),
  "Numeric decimal" = abs(round(rnorm(10), 5)),
  "Long name that means that the column width needs to be widened" = 1:10,
  Notes = c(
    "[note 1]", rep(NA_character_, 4), "[note 2]",
    rep(NA_character_, 4)
  ),
  check.names = FALSE
)

table_2_df <- data.frame(Category = LETTERS[1:10], Numeric = 1:10)

# Create 'aftables' object

x <- aftables::create_aftable(
  tab_titles = c("Cover", "Contents", "Notes", "Table_1", "Table_2"),
  sheet_types = c("cover", "contents", "notes", "tables", "tables"),
  sheet_titles = c(
    "The 'aftables' Demo Workbook",
    "Table of contents",
    "Notes",
    "Table 1: First Example Sheet",
    "Table 2: Second Example Sheet"
  ),
  blank_cells = c(
    rep(NA_character_, 3),
    "Blank cells indicate that there's no note in that row.",
    NA_character_
  ),
  custom_rows = list(
    NA_character_,
```

```
    NA_character_,
    "A custom row.",
    c(
      paste0(
        "First custom row [with a hyperlink.]",
        "(https://best-practice-and-impact.github.io/aftables/)"
      ),
      "Second custom row."
    ),
    "A custom row."
  ),
  sources = c(
    rep(NA_character_, 3),
    paste0(
      "[The Source Material, 2024.]",
      "(https://best-practice-and-impact.github.io/aftables/)"
    ),
    "The Source Material, 2024."
  ),
  tables = list(cover_list, contents_df, notes_df, table_1_df, table_2_df)
)

# Test that 'aftable' is one of the object's classes
is_aftable(x)

# Look at the structure of the object
str(x, max.level = 2)
```

`create_config_yaml` *Create an aftables config.yaml*

Description

Copy the example config.yaml file included with aftables to a directory of the user's choice, and optionally open the file for editing. The config.yaml file can be passed to the aftables function `generate_workbook`.

Usage

```
create_config_yaml(path = getwd(), open_config = rlang::is_interactive())
```

Arguments

`path`	optional character string containing directory to copy the config.yaml file. Defaults to current working directory.
`open_config`	optional logical whether to open the copy of config.yaml for editing in the current R session.

Details

If there is an existing config.yaml file in the destination directory this function will not overwrite it.

Contents of example config.yaml:

```
aftables:
  default:
    workbook_properties:
      author: "aftables"
      title: "aftables example workbook"
      keywords:
        - "aftables1"
        - "aftables2"
        - "aftables3"
      subject: "aftables example subject"
      category: "aftables example category"
      comments: "aftables example comments"
    workbook_format:
      base_font_name: "Arial"
      base_font_size: 12
      table_header_size: 12
      sheet_heading_size: 16
      sheet_subheading_size: 14
      cellwidth_default: 16
      cellwidth_wider: 32
      nchar_break: 50
  workbook1:
    workbook_properties:
      category: "aftables workbook 1 category"
      keywords:
        - "aftablesworkbook1"
  workbook2:
    workbook_properties:
      title: "aftables workbook 2"
      category: "aftables workbook 2 category"
      keywords:
        - "aftablesworkbook2"
```

All configurations must be placed below an `aftables` key. `aftables` should be followed by a `default` key and/or custom keys (e.g. `workbook1`).

The `default` key settings will be read by all calls to `generate_workbook` which use this config.yml. This allows you to share settings when generating multiple workbooks in one script.

Custom key settings (e.g. `workbook1`) will only be used by `generate_workbook` when the key is provided as the `config_name` argument. This allows you to specify settings for a specific workbook. Custom key settings will be preferred over the `default` settings.

Keys below workbook_properties will appear in the Excel workbook when it is saved using `wb_save` from openxlsx2. They can be found in the file properties or the workbook information.

Keys below workbook_format will be applied to the contents of the workbook. The values of base_font_name and base_font_size define the default font name and size used by the workbook. All text not formatted as a table header, sheet subheading or sheet heading will use the default settings. Font sizes of sheet headings, sheet subheadings, and table header rows will use the values of sheet_heading_size, sheet_subheading_size and table_header_size respectively, and they will additionally be formatted as bold.

The values of cellwidth_default, cellwidth_wider and nchar_break are used to define column widths. The units of cellwidth_default and cellwidth_wider are the column width values used by Excel. All columns widths are set by default to use the cellwidth_default value. If the number of characters in a column header or the contents of a column exceeds the value of nchar_break aftables will set the column width to the value of cellwidth_wider. Users can avoid text wrapping in columns or column headers by setting the value of nchar_break based on their data or the content of their column headers.

Not all workbook configuration options need to be set. Required settings are documented in generate_workbook.

Examples

```
# Use default arguments to copy `aftables` `config.yaml` file to the current
# working directory without opening the copied file for editing:

## Not run:
create_config_yaml(open_config = FALSE)
## End(Not run)
```

---

demo_aftable *A Demo 'aftables' Object*

Description

A pre-created 'aftables' object ready to be converted to an 'openxlsx2' wbWorkbook-class object with generate_workbook.

Usage

```
demo_aftable
```

Format

A data.frame with 6 rows and 7 columns:

tab_title Character. Text to appear on each sheet's tab.

sheet_type Character. The content type for each sheet: 'cover', 'contents', 'notes', or 'tables'.

sheet_title Character. The title that will appear in cell A1 (top-left) of each sheet.

blank_cells Character. An explanation for any blank cells in the table.

custom_rows List-column of character vectors. Additional arbitrary pre-table information provided by the user.

source Character. The origin of the data, if relevant.

table List-column of data.frames (apart from the cover, which is a list) containing the statistical tables.

`demo_df` *A Demo 'data.frame' Object*

Description

A pre-created data.frame ready to be converted to an aftables-class object with `as_aftable` and then an 'openxlsx2' wbWorkbook-class object with `generate_workbook`.

Usage

```
demo_df
```

Format

A data.frame with 6 rows and 7 columns:

tab_title Character. Text to appear on each sheet's tab.

sheet_type Character. The content type for each sheet: 'cover', 'contents', 'notes', or 'tables'.

sheet_title Character. The title that will appear in cell A1 (top-left) of each sheet.

blank_cells Character. An explanation for any blank cells in the table.

custom_rows List-column of character vectors. Additional arbitrary pre-table information provided by the user.

source Character. The origin of the data, if relevant.

table List-column of data.frames (apart from the cover, which is a list) containing the statistical tables.

`demo_workbook` *A Demo 'Workbook' Object*

Description

A pre-created 'openxlsx2' wbWorkbook-class object generated from an aftables-class object with `generate_workbook`.

Usage

```
demo_workbook
```

Format

An 'openxlsx2' wbWorkbook-class object with 5 sheets.

generate_workbook *Generate A Workbook Object From An 'aftable'*

Description

Populate an 'openxlsx2' wbWorkbook-class object with content from an aftable-class object. In turn, the output can be passed to `wb_save` from 'openxlsx2'

Usage

```
generate_workbook(
  aftable,
  author = NULL,
  title = NULL,
  keywords = NULL,
  config_path = "config.yaml",
  config_name = NULL
)
```

Arguments

`aftable`	An aftable-class object created using `create_aftable` (or `as_aftable`), which contains the data and information needed to create a workbook.
`author`	Optional character string to set the workbook author. Default NULL.
`title`	Optional character string to set the workbook title. Default NULL.
`keywords`	Optional character vector to set the workbook keywords. Default NULL.
`config_path`	Optional character string containing path to config file. Defaults to config.yaml file located in working directory.
`config_name`	Optional character string specifying which configuration to use from config file. Default NULL.

Details

Analysis Function guidance advises workbooks should have the author, title, keywords and language document properties completed. `aftables` provides functionality to set the author, title and keywords properties. See Releasing statistics in spreadsheets for more information including how to set the workbook language.

A config file can be used to set workbook properties and formatting. See `create_config_yaml` for details of how to create a config.yaml file. If author, title or keywords are provided in both the `config.yaml` file and in the `generate_workbook` arguments, the values provided in the function arguments are preferred to those provided in the `config.yaml` file.

Value

An openxlsx2 wbWorkbook-class object.

Examples

```
# Convert an aftable to an openxlsx2 wbWorkbook-class object
# Setting the minimum workbook properties as function arguments
## Not run:
example_workbook <- generate_workbook(
  demo_aftable,
  author = "Example author",
  title = "example workbook",
  keywords = c("keyword1", "keyword2", "keyword3")
)
## End(Not run)

# Use openxlsx2::wb_get_properties to view properties that have been applied
## Not run:
openxlsx2::wb_get_properties(example_workbook)
## End(Not run)

# Save the workbook with openxlsx2::wb_save
## Not run:
openxlsx2::wb_save(example_workbook, "example_workbook.xlsx")
## End(Not run)

# Using config.yaml file to set workbook properties and edit text and cell
# formatting
## Not run:
example_workbook2 <- generate_workbook(
  demo_aftable,
  config_path = system.file("ext-data", "config.yaml", package = "aftables"),
  config_name = "workbook1"
  )
## End(Not run)

# Use openxlsx2::wb_get_properties to view properties that have been applied
## Not run:
openxlsx2::wb_get_properties(example_workbook2)
## End(Not run)
```

`summary.aftable` *Summarise An 'aftable' Object*

Description

A concise result summary of an aftable-class object to see information about the sheet content. Shows a numbered list of sheets with each tab title, sheet type and table dimensions.

Usage

```
## S3 method for class 'aftable'
summary(object, ...)
```

Arguments

`object`	An aftable-class object for which to get a summary.
`...`	Other arguments to pass.

Value

`object` unaltered.

Examples

```
# Print a concise summary of the aftable-class object
summary(demo_aftable)

# Alternatively, look at the structure
str(demo_aftable, max.level = 2)
```

`tbl_sum.aftable` *Provide A Succinct Summary Of An 'aftable' Object*

Description

A brief text description of an aftable-class object.

Usage

```
## S3 method for class 'aftable'
tbl_sum(x, ...)
```

Arguments

`x`	An aftable-class object to summarise.
`...`	Other arguments to pass.

Value

Named character vector.

Examples

```
# Print with description
print(demo_aftable)

# Print description only (package 'tibble' must be installed)
tibble::tbl_sum(demo_aftable)
```

Index