

Package ‘ihsMW’

July 29, 2026

Title Clean and Harmonise 'Malawi Integrated Household Survey' Data

Version 1.1.1

Description An offline suite of tools to clean, aggregate, and harmonise data from the 'Malawi Integrated Household Survey' ('IHS'), covering rounds two through six (2004 to 2025). Provides crop-specific unit conversions, stratified winsorization, consumer-price deflation, and automatic cross-round harmonisation for complex survey designs.

License MIT + file LICENSE

Depends R (>= 4.1.0)

URL <https://github.com/vituk123/ihsMW>,
<https://vituk123.github.io/ihsMW/>

BugReports <https://github.com/vituk123/ihsMW/issues>

Imports dplyr (>= 1.1.0), readr (>= 2.1.0), rlang (>= 1.1.0), cli (>= 3.6.0)

Suggests srvyr (>= 1.2.0), survey (>= 4.2.0), testthat (>= 3.0.0),
usethis (>= 2.2.0), pkgdown (>= 2.0.0), knitr (>= 1.40),
rmarkdown (>= 2.20), withr (>= 2.5.0), jsonlite (>= 1.8.0)

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/testthat/edition 3

Language en-US

NeedsCompilation no

Author Vitumbiko Kayuni [aut, cre] (ORCID:
<<https://orcid.org/0009-0008-4331-3839>>)

Maintainer Vitumbiko Kayuni <vitumbikokayuni@gmail.com>

Repository CRAN

Date/Publication 2026-07-29 12:10:01 UTC

Contents

ihs_aggregate	2
ihs_clean	3
ihs_convert_units	4
ihs_crosswalk_check	6
ihs_deflate	6
ihs_harmonise	8
ihs_merge	9
ihs_panel_ids	10
ihs_report	11
ihs_search	12
ihs_standardize_missing	13
ihs_svydesign	14
ihs_winsorize	15
Index	17

ihs_aggregate	<i>Smart Aggregation to Household Level</i>
---------------	---

Description

Automatically detects variable types and applies sensible aggregations, so that individual-level or plot-level modules can be rolled up to the household without hand-writing a ‘summarise()’ call per column.

Usage

```
ihs_aggregate(data, group_col = "case_id")
```

Arguments

data	A data.frame at the individual or plot level
group_col	The column name identifying the household (e.g., "case_id" or "y4_hhid")

Value

A data.frame with one row per value of ‘group_col’.

Aggregation rules

- continuous numeric columns are summed (‘na.rm = TRUE’);
- numeric columns containing only 0, 1 and ‘NA’ are treated as dummies and become 1 if any member has a 1 (a logical OR);
- logical columns become ‘any()’;
- character and factor columns take the single distinct value if there is one, otherwise the distinct values joined with “ | ”;

- anything else is dropped, with a warning naming the column.

Summing is the right default for quantities and expenditures but not for ages or prices - select your columns before calling, or aggregate those yourself.

See Also

[ihc_convert_units](#) to put quantities in comparable units before summing them.

Examples

```
members <- data.frame(
  case_id = c("A", "A", "B", "B", "B"),
  income = c(1000, 500, 200, 300, 250),
  has_radio = c(0, 1, 0, 0, 0),
  district = c("Lilongwe", "Lilongwe", "Blantyre", "Blantyre", "Blantyre")
)
ihc_aggregate(members, group_col = "case_id")
```

ihc_clean

Clean and Harmonise IHS Data

Description

This wrapper function applies standard cleaning procedures to Malawi IHS data. It handles missing value conversions, winsorization of continuous variables, and returns an audit log of all transformations applied.

Usage

```
ihc_clean(
  data,
  winsorize_vars = NULL,
  winsorize_by = NULL,
  probs = c(0.01, 0.99)
)
```

Arguments

data	A data.frame (typically loaded from a '.dta' file)
winsorize_vars	Character vector of continuous variables to winsorize (e.g., consumption, harvest)
winsorize_by	Optional character string of a grouping variable (e.g., region) for stratified winsorization
probs	Numeric vector of length 2 specifying the lower and upper quantiles for winsorization. Default is 'c(0.01, 0.99)'.

Value

A data.frame with cleaning applied. The returned object has an ‘ihs_audit’ attribute containing a log of modifications: initial and final dimensions, how many values were recoded to ‘NA’ in each column, and how many observations each winsorized variable had capped at the lower and upper bound.

See Also

[ihs_standardize_missing](#) and [ihs_winsorize](#), which this function wraps.

Examples

```
survey <- data.frame(
  region = rep(c("North", "South"), each = 50),
  food_exp = c(rnorm(50, 100, 10), rnorm(50, 500, 50))
)
# A refusal code and an implausible outlier
survey$food_exp[1] <- -99
survey$food_exp[60] <- 99999

clean <- ihs_clean(survey, winsorize_vars = "food_exp",
  winsorize_by = "region")

# Winsorized values land in a new `_w` column; the raw column is preserved
head(clean[, c("food_exp", "food_exp_w")])

# Inspect what was changed
str(attr(clean, "ihs_audit"))
```

ihs_convert_units	<i>Convert Agricultural Units to Kilograms</i>
-------------------	--

Description

Converts reported harvest units (e.g., Pails, Ox carts, Heaps) into standard kilograms using official NSO crop-specific conversion factors.

Usage

```
ihs_convert_units(data, qty_col, unit_col, crop_col, unmapped = "warn")
```

Arguments

data	A data.frame.
qty_col	The name of the column containing the quantity.
unit_col	The name of the column containing the unit code.
crop_col	The name of the column containing the crop code.

unmapped Action to take when a crop-unit combination cannot be mapped: "warn" (default), "error", or "ignore".

Details

IHS agriculture modules let households report quantities in whatever unit they actually use, so a raw quantity column mixes kilograms, pails, ox-carts and heaps in the same variable. Summing it directly is meaningless. This function looks up the right factor for each crop, unit, region and shelling condition and returns a comparable kilogram column.

Value

A data.frame with a new column named <qty_col>_kg. Rows whose crop-unit combination is not in the factor table get NA, never a silently wrong number.

Matching rules

Factors vary by region and by whether the crop was shelled, so the lookup resolves in this order:

1. exact match on crop, unit, region and condition;
2. same crop, unit and region, falling back through the condition preference order shelled (1), not applicable (3), unshelled (2);
3. the same two steps against the Central region, used as the national fallback when a crop-unit pair has no factor for the row's own region.

A region column (or ihs_region / survey_region) is detected automatically and used if present. Without one, every row is priced at Central-region factors and the function says so.

See Also

[ihs_aggregate](#) to total the resulting kilogram column to household level.

Examples

```
harvest <- data.frame(
  crop_code = c(1, 1, 1),
  unit_code = c(1, 2, 3),
  quantity  = c(100, 2, 5),
  region    = c(1, 2, 2)
)
ihs_convert_units(harvest, qty_col = "quantity",
                  unit_col = "unit_code", crop_col = "crop_code")

# Unmappable combinations return NA rather than a wrong number
odd <- data.frame(crop_code = 999, unit_code = 999, quantity = 10)
suppressWarnings(
  ihs_convert_units(odd, "quantity", "unit_code", "crop_code")
)
```

ihs_crosswalk_check	<i>Check the comparability of variables across IHS rounds</i>
---------------------	---

Description

Evaluates the completeness and comparability of variables across the available IHS rounds (IHS2, IHS3, IHS4, IHS5, IHS6) using the bundled crosswalk. Use it to see how many variables are genuinely comparable across the full series before committing to a pooled analysis.

Usage

```
ihs_crosswalk_check(verbose = TRUE)
```

Arguments

`verbose` Logical. If TRUE (default), prints a summary report to the console using `cli`.

Value

Invisibly, a tibble containing the full crosswalk with one `ihs*_name` column per round plus an `n_rounds_avail` column counting how many rounds each variable appears in. If `verbose` is TRUE, also prints a summary to the console.

See Also

[ihs_search](#) to look up individual variables.

Examples

```
# Return the crosswalk without printing the report
cw <- ihs_crosswalk_check(verbose = FALSE)
nrow(cw)

# Variables available in every round
sum(cw$n_rounds_avail == max(cw$n_rounds_avail))
```

ihs_deflate	<i>Deflate Nominal Values to Real Values Using CPI</i>
-------------	--

Description

Converts nominal monetary values to real (constant-price) values using Malawi CPI data. By default uses 2019 (the IHS5 fieldwork year) as the reference period, so figures from every round are expressed in 2019 kwacha and can be compared directly.

Usage

```
ihs_deflate(data, value_cols, round = NULL, base_year = 2019)
```

Arguments

<code>data</code>	A data.frame.
<code>value_cols</code>	Character vector of column names containing monetary values to deflate.
<code>round</code>	Character string of the IHS round (e.g., "IHS5"). If NULL (default), auto-detected from an <code>ihs_round</code> column in the data - which is what ihs_harmonise adds, so pooled multi-round data.frames deflate correctly without further arguments.
<code>base_year</code>	Numeric. The base year for deflation. Default is 2019 (IHS5 baseline). Must be a year present in the CPI table.

Details

Each round is deflated using its survey midpoint year: IHS2 = 2004, IHS3 = 2010, IHS4 = 2016, IHS5 = 2019, IHS6 = 2024. The bundled CPI table (`inst/extdata/mw_cpi_annual.csv`) is the World Bank WDI series `FP.CPI.TOTL` for Malawi, rebased to 2019 = 100, covering 2004-2025.

Value

A data.frame with new `*_real` suffixed columns containing deflated values. Original nominal columns are left untouched.

Caveat

This is a national CPI deflator. It does not adjust for spatial price variation between districts or between urban and rural areas. If your analysis is sensitive to spatial price differences, use the spatial price index published with each round's consumption aggregate instead.

See Also

[ihs_harmonise](#), which adds the `ihs_round` column this function reads.

Examples

```
# A pooled data.frame carrying an `ihs_round` column deflates in one call
pooled <- data.frame(
  food_exp = c(1000, 1000, 1000),
  ihs_round = c("IHS4", "IHS5", "IHS6")
)
ihs_deflate(pooled, value_cols = "food_exp")

# Or state the round explicitly for a single-round data.frame
ihs_deflate(data.frame(food_exp = 1000), value_cols = "food_exp",
  round = "IHS6")
```

ihs_harmonise

*Harmonise Raw IHS Data***Description**

Takes a raw data.frame loaded from a Malawi IHS survey round (e.g. from a '.dta' or '.csv' file) and renames its columns to the standard harmonised variable names defined in the bundled crosswalk. This is the first step of almost every 'ihsMW' workflow: once column names are harmonised, the same downstream code works on any round.

Usage

```
ihs_harmonise(data, round = "IHS5", extra = FALSE)
```

Arguments

data	A data.frame, typically read from a '.dta' file using <code>haven::read_dta</code> or a '.csv' using <code>readr::read_csv</code> .
round	A character string specifying the IHS round: one of "IHS2", "IHS3", "IHS4", "IHS5" or "IHS6".
extra	Logical. If FALSE (default), drops columns that are not in the harmonisation crosswalk or standard ID columns. If TRUE, keeps all original columns.

Details

Matching is case-insensitive, so 'HHID' and 'hhid' are treated as the same variable. Each raw column is claimed by at most one crosswalk entry.

Value

A data.frame with columns renamed to standard 'harmonised_name's where applicable, plus an 'ihs_round' column recording which round the rows came from. Stata variable labels are preserved across the rename.

See Also

[ihs_search](#) to find out what a raw variable is called in each round before harmonising.

Examples

```
# A miniature stand-in for a raw IHS file
raw <- data.frame(case_id = c("a", "b"), hhsize = c(4, 6), junk = c(1, 2))

# Harmonise, dropping columns absent from the crosswalk
ihs_harmonise(raw, round = "IHS6")

# Keep everything, including unmapped columns
ihs_harmonise(raw, round = "IHS6", extra = TRUE)
```

ihs_merge

*Merge Multiple Harmonised IHS Data Frames***Description**

Merges two or more harmonised data.frames, auto-detecting common ID columns when by is not specified. Warns on unexpected row expansion from many-to-many joins.

Usage

```
ihs_merge(..., by = NULL, type = "left")
```

Arguments

...	Two or more data.frames to merge.
by	Character vector of columns to join by. If NULL (default), auto-detects common ID columns from a standard set.
type	Join type: "left" (default), "inner", or "full".

Value

A merged data.frame with an ihs_merge_log attribute containing row counts at each merge step.

Auto-detected keys

When by is NULL the join keys are the intersection of case_id, hhid, hh_id, HHID, ea_id and PID with the columns common to every input. The detected keys are always printed - read them, because joining household modules on PID when you meant case_id silently changes your sample.

Examples

```
hh <- data.frame(case_id = c("A", "B", "C"), hhsize = c(4, 6, 3))
ag <- data.frame(case_id = c("A", "B", "D"), harvest_kg = c(120, 340, 90))

# Left join on the auto-detected key, keeping every household
merged <- ihs_merge(hh, ag)
merged

# Only households present in both modules
ihs_merge(hh, ag, type = "inner")

# Row counts at each step
attr(merged, "ihs_merge_log")
```

ihs_panel_ids

*Get Standard Panel ID Columns for an IHS Round***Description**

Returns the standard household, individual, and enumeration area ID column names used in a given IHS round. This is a convenience helper for users building longitudinal panels or merging modules.

Usage

```
ihs_panel_ids(round = "IHS5")
```

Arguments

round A character string specifying the IHS round: one of "IHS2", "IHS3", "IHS4", "IHS5" or "IHS6". Use "all" for a comparison across all rounds.

Value

If a single round is specified, a named character vector of standard ID columns with elements hh_id, indiv_id, ea_id, strata and weight. If "all", a data.frame comparing ID columns across rounds.

Round differences

Two names change across the series and catch people out:

- the household weight is hhweight in IHS2 and IHS3 but hh_wgt from IHS4 onwards;
- the design stratum is stratum in IHS2-IHS5 but strata in IHS6.

[ihs_svydesign](#) handles both automatically.

Examples

```
# Get IHS6 ID columns
ihs_panel_ids("IHS6")

# Compare across all rounds - note the stratum rename in IHS6
ihs_panel_ids("all")
```

ihs_report

*Generate Summary Statistics Table***Description**

Produces a publication-ready summary statistics table for numeric variables in IHS data. Supports grouping by a factor variable and optional survey weights.

Usage

```
ihs_report(data, vars = NULL, by = NULL, weights = NULL)
```

Arguments

<code>data</code>	A data.frame.
<code>vars</code>	Character vector of column names to summarise. If NULL (default), all numeric columns are included.
<code>by</code>	Optional character string of a grouping variable (e.g., "region", "urban").
<code>weights</code>	Optional character string of a column containing survey weights for weighted means and SDs.

Value

A data.frame of summary statistics with columns: variable, n, mean, sd, median, min, max, pct_missing. If by is specified, an additional grouping column is included.

Weighted statistics

When weights is supplied the mean and SD are weighted; n, median, min, max and pct_missing stay unweighted, since they describe the sample rather than the population. For weighted quantiles or design-correct standard errors, build a design with [ihs_svydesign](#) and use the **survey** package.

See Also

[ihs_svydesign](#) for design-based inference.

Examples

```
hh <- data.frame(
  hhsize = c(4, 6, 3, 5, 7, 2),
  food_exp = c(1200, 3400, 900, 2100, 4500, 700),
  region = c("North", "North", "Central", "Central", "South", "South"),
  hh_wgt = c(1.2, 0.8, 1.5, 1.1, 0.9, 1.3)
)

# Basic summary of every numeric column (weight columns are excluded)
ihs_report(hh)
```

```
# Selected variables, grouped by region
ihs_report(hh, vars = "food_exp", by = "region")

# Survey-weighted means and SDs
ihs_report(hh, vars = c("hhsz", "food_exp"), weights = "hh_wgt")
```

ihs_search

Search across all IHS rounds for variables manually mapped

Description

Searches the manual harmonisation crosswalk bundled within ihsMW for specific variables.

Usage

```
ihs_search(keyword, round = NULL, fields = c("name", "label", "module"))
```

Arguments

keyword	A single search string to find (case-insensitive).
round	Limits the search to variables that exist in a specific round. Valid inputs are "IHS2", "IHS3", "IHS4", "IHS5", and "IHS6". Defaults to NULL (all rounds).
fields	A character vector of fields to include in the search. Valid fields are "name", "label", and "module".

Value

A tibble with one row per matching harmonised variable and one ihs*_name column per survey round, plus n_rounds (how many rounds the variable appears in) and needs_review.

Examples

```
ihs_search("consumption")
ihs_search("expenditure", round = "IHS5")
ihs_search("age", fields = c("name", "label"))

# Variables that are new in IHS6
ihs_search("livestock", round = "IHS6")
```

`ihs_standardize_missing`*Standardize Survey Missing Codes*

Description

Converts common survey missing codes (like -99 for "Refused" or -98 for "Don't Know") into standard R 'NA' values to prevent them from skewing numeric calculations. The codes recognised are -99, -98, -97, 999, 998 and 997. Only numeric columns are touched; character columns pass through untouched.

Usage

```
ihs_standardize_missing(data)
```

Arguments

<code>data</code>	A data.frame
-------------------	--------------

Value

A data.frame with missing values standardized, carrying an 'ihs_missing_conversions' attribute recording how many values were recoded in each affected column.

Warning

999 is a legitimate value for some variables - a plot area in square metres, a price in kwacha. Check your variables before relying on the blanket recode, and pass only the affected columns through this function if in doubt.

Examples

```
df <- data.frame(age = c(34, -99, 41, 998), village = c("A", "B", "C", "D"))
clean <- ihs_standardize_missing(df)
clean$age

# How many values were recoded, by column
attr(clean, "ihs_missing_conversions")
```

ihs_svydesign

*Create a Survey Design Object for IHS Data***Description**

Wraps `survey::svydesign()` with automatic detection of standard IHS weight, strata, and PSU columns from harmonised data.

Usage

```
ihs_svydesign(data, weight_col = NULL, strata_col = NULL, psu_col = NULL)
```

Arguments

<code>data</code>	A <code>data.frame</code> of harmonised IHS data.
<code>weight_col</code>	Character. Column name for survey weights. If <code>NULL</code> (default), auto-detected from standard IHS column names.
<code>strata_col</code>	Character. Column name for strata. If <code>NULL</code> (default), auto-detected.
<code>psu_col</code>	Character. Column name for PSU/cluster. If <code>NULL</code> (default), auto-detected.

Value

A `survey.design2` object from the **survey** package.

Detected column names

Weights are looked up as `hh_wgt`, `hhweight`, `weight`, `panelweight`, `hh_wgt_adj` or `hhwght`; strata as `stratum`, `strata`, `strataid` or `strat`; and PSUs as `ea_id`, `psu`, `cluster` or `clusterid`. Matching is case-insensitive, which is what lets the same call work on IHS2-IHS5 (`stratum`, `hhweight/hh_wgt`) and on IHS6, which renamed the `stratum` column to `strata`. The detected names are always printed.

See Also

[ihs_panel_ids](#) for the standard design columns of each round, and [ihs_report](#) for a quick weighted summary table that does not require a design object.

Examples

```
hh <- data.frame(
  case_id = 1:20,
  ea_id   = rep(1:5, each = 4),
  strata  = rep(c("urban", "rural"), each = 10),
  hh_wgt  = runif(20, 0.5, 3),
  food_exp = rnorm(20, 5000, 1000)
)

# Weight, stratum and PSU columns are all detected automatically
```

```
dsgn <- ihw_svydesign(hh)

# Population mean with design-correct standard errors
survey::svymean(~food_exp, dsgn, na.rm = TRUE)
```

ihw_winsorize

Winsorize Continuous Variables

Description

Caps extreme outliers at specified percentiles. Crucially, this function allows for stratified winsorization (e.g., by region) to avoid over-trimming poor/rich areas, and it creates new ‘_w’ suffixed columns to preserve raw data provenance.

Usage

```
ihw_winsorize(data, vars, by = NULL, probs = c(0.01, 0.99))
```

Arguments

data	A data.frame
vars	Character vector of column names to winsorize
by	Optional grouping variable name (e.g., "region") for stratified thresholds
probs	Numeric vector of lower and upper quantiles. Default ‘c(0.01, 0.99)’

Value

A data.frame with new ‘*_w’ columns added, carrying an ‘ihw_winsorized_vars’ attribute recording how many observations were capped at each bound.

Why stratify

A single national 99th percentile treats the richest rural households as outliers because the urban distribution sits far above them. Passing ‘by’ computes the cut-points separately within each stratum, which preserves the shape of both distributions.

Examples

```
df <- data.frame(
  region = rep(c("North", "South"), each = 50),
  cons   = c(rnorm(50, 100, 10), rnorm(50, 500, 50))
)

# Global thresholds
g <- ihw_winsorize(df, vars = "cons", probs = c(0.05, 0.95))

# Thresholds computed within each region
```

```
s <- ihs_winsorize(df, vars = "cons", by = "region", probs = c(0.05, 0.95))

# The raw column is never modified
identical(g$cons, df$cons)
attr(s, "ihs_winsorized_vars")
```

Index

ihs_aggregate, [2](#), [5](#)
ihs_clean, [3](#)
ihs_convert_units, [3](#), [4](#)
ihs_crosswalk_check, [6](#)
ihs_deflate, [6](#)
ihs_harmonise, [7](#), [8](#)
ihs_merge, [9](#)
ihs_panel_ids, [10](#), [14](#)
ihs_report, [11](#), [14](#)
ihs_search, [6](#), [8](#), [12](#)
ihs_standardize_missing, [4](#), [13](#)
ihs_svydesign, [10](#), [11](#), [14](#)
ihs_winsorize, [4](#), [15](#)