Package: scf (via r-universe)

October 19, 2026

Title Analyzing the Survey of Consumer Finances

Version 1.2.0

Description Analyze public-use micro data from the Survey of Consumer Finances. Provides tools to download prepared data files, construct replicate-weighted multiply imputed survey designs, compute descriptive statistics and model estimates, and produce plots and tables. Methods follow design-based inference for complex surveys and pooling across multiple imputations. See the package website and the code book for background.

License MIT + file LICENSE

URL https://github.com/jncohen/scf

BugReports https://github.com/jncohen/scf/issues

Depends R (>= 3.6)

Imports ggplot2, haven, httr, stats, survey, utils, quantreg

Suggests dplyr, hexbin, kableExtra, knitr, mitools, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs make libssl-dev libx11-dev zlib1g-dev

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

Date/Publication 2026-10-19 20:15:38 UTC

RemoteUrl https://github.com/jncohen/scf

RemoteRef HEAD

RemoteSha b0f2f97d0f70988963b5764976153be07f432fa0

Contents

scf . . . 3
scf_corr . . . 6
scf_design . . . 7
scf_download . . . 8
scf_freq . . . 9
scf_glm . . . 11
scf_implicates . . . 12
scf_load . . . 13
scf_logit . . . 14
scf_mean . . . 15
scf_median . . . 16
scf_MIcombine . . . 17
scf_model_result_methods . . . 20
scf_nominal . . . 21
scf_ols . . . 23
scf_pctile_sum . . . 24
scf_percentile . . . 26
scf_plot_bbar . . . 27
scf_plot_cbar . . . 29
scf_plot_dbar . . . 30
scf_plot_dist . . . 31
scf_plot_hex . . . 32
scf_plot_hist . . . 33
scf_plot_smooth . . . 35
scf_prop_test . . . 36
scf_quantreg . . . 37
scf_ratio . . . 40
scf_real . . . 41
scf_regtable . . . 42
scf_se . . . 43
scf_subset . . . 44
scf_theme . . . 45
scf_ttest . . . 46
scf_update . . . 48
scf_update_by_implicate . . . 49
scf_variance . . . 50
scf_xtab . . . 52

Index **53**

scf *Analyzing Survey of Consumer Finances Public-Use Microdata*

Description

Functions to analyze the Federal Reserve's Survey of Consumer Finances (SCF) public-use microdata. The package stores the SCF's five implicates and replicate weights in one object (`scf_mi_survey`) and provides means, medians, percentiles, frequencies, cross-tabs, ratios, tests, regression models, and plots.

It is written for analysts who know standard statistics but not the details of multiply imputed or survey-weighted data.

Methodological Background

The SCF is a detailed source of data on U.S. household finances. It is nationally representative, oversamples wealthy households, and fills in missing answers by multiple imputation. Valid estimates need three things:

- Sampling weights. The SCF uses a dual-frame probability sample, so estimates must use the weights.
- Replicate weights. Each household has 999 bootstrap replicate weights, built by the Fed, for estimating sampling variance.
- Multiple imputation. Each household appears in five implicates. Estimates are pooled across them to get point estimates and standard errors.

The package handles all three. See scf_variance for how standard errors are computed.

Workflow

For the methods behind these functions, see Cohen (2026).

1. Download the data with `scf_download()`.
2. Load it with `scf_load()`, which returns a `scf_mi_survey` object.
3. Change variables with `scf_update()` or `scf_update_by_implicate()`, and keep a subset of households with `scf_subset()`.
4. Describe the data with `scf_mean()`, `scf_median()`, `scf_percentile()`, `scf_freq()`, `scf_xtab()`, `scf_ratio()`, `scf_corr()`, and `scf_pctile_sum()`.
5. Test hypotheses with `scf_ttest()` and `scf_prop_test()`.
6. Fit models with `scf_ols()`, `scf_logit()`, `scf_glm()`, and `scf_quantreg()`, and tabulate them with `scf_regtable()`.
7. Plot with `scf_plot_dist()`, `scf_plot_hist()`, `scf_plot_dbar()`, `scf_plot_cbar()`, `scf_plot_bbar()`, `scf_plot_smooth()`, and `scf_plot_hex()`. Change the look of the plots with `scf_theme()`.
8. Convert raw `x` dollar variables to real dollars with `scf_real()`, and dollar results to nominal survey-year dollars with `scf_nominal()`.
9. See the estimate from each implicate with `scf_implicates()`, and pool your own estimates with `scf_MIcombine()`.

Households

The SCF's unit is the primary economic unit: the person or couple at the economic center of the household and everyone financially dependent on them. The Fed calls this unit a "family"; this package calls it a household.

Dollars

Summary variables made by the Fed, such as `income`, `networth`, and `asset`, are in the dollars of one year in every survey year, so they can be compared across years as they are. That year is the Fed's latest survey: 2025 for files downloaded since October 2026, 2022 before that. `scf_load()` records it as `dollar_year`.

Raw survey variables, named `x` followed by a number, are in nominal dollars: dollars of the year the survey asked about. They are not adjusted for inflation. Convert them with `scf_real()` before comparing years or combining them with summary variables. To report a summary variable in nominal dollars, use `scf_nominal()` on the result.

Data Object

Functions work on a `scf_mi_survey` object, made by `scf_design()` via `scf_load()`. Use `scf_design()` directly only for designs built outside `scf_load()`. It is a list with:

- `mi_design`: five `survey::svrepdesign()` objects, one per implicate
- `year`: the survey year
- `n_households`: the weighted number of households
- `dollar_year`: the year whose dollars the summary variables are in

Imputed Missing Data

The SCF fills in missing answers by multiple imputation (Kennickell 1998). A model of the observed data is used to draw five plausible values for each missing answer, giving five complete data sets, called implicates. The spread across implicates shows how uncertain the filled-in values are. See scf_variance and `scf_MIcombine()`.

Mock Data for Testing

CRAN limits a package to 5 MB, and one wave of the SCF is over 100 MB, so the package cannot include the data. Examples and tests use a small mock data set (`scf2022_mock_raw.rds` in `inst/extdata/`): the first 200 households of each 2022 implicate, with the variables the examples use. It is not suitable for analysis or inference.

Plot Theme

All plots use `scf_theme()`. See its help page to change the theme.

Teaching

The package can be used to teach survey analysis and missing data, because each step can be inspected:

- Each implicate's design is in `x$mi_design[[i]]`.
- Each implicate's data is in `x$mi_design[[i]]$variables`.
- Results keep implicate-level estimates (see `scf_implicates()`).
- Standard errors come from replicate weights and the spread across implicates, so students can follow how each part adds to the total.

Author(s)

Joseph N. Cohen, CUNY Queens College

References

Cohen JN. Analyzing the Survey of Consumer Finances with **scf**: Methodology, User Guide, and Package Validation. Working paper, CUNY Queens College; 2026. https://academicworks.cuny.edu/qc_pubs/695/

Barnard J, Rubin DB. Small-sample degrees of freedom with multiple imputation. *Biometrika*. 1999;86(4):948–955. doi:10.1093/biomet/86.4.948

Board of Governors of the Federal Reserve System. Codebook for the 2025 Survey of Consumer Finances. https://www.federalreserve.gov/econres/scfindex.htm

Bricker J, Henriques AM, Moore KB. Updates to the sampling of wealthy families in the Survey of Consumer Finances. Finance and Economics Discussion Series 2017-114. Board of Governors of the Federal Reserve System; 2017.

Kennickell AB. Multiple imputation in the Survey of Consumer Finances. Board of Governors of the Federal Reserve System; 1998.

Kennickell AB. Multiple imputation in the Survey of Consumer Finances. *Statistical Journal of the IAOS*. 2017;33(1):143–151. doi:10.3233/SJI160278

Kennickell AB, McManus DA, Woodburn RL. Weighting design for the 1992 Survey of Consumer Finances. Board of Governors of the Federal Reserve System; 1996. https://www.federalreserve.gov/Pubs/OSS/oss2/papers/weight92.pdf

Little RJA, Rubin DB. *Statistical Analysis with Missing Data*. 3rd ed. Wiley; 2019. ISBN: 9780470526798.

Lumley T. Analysis of complex survey samples. *Journal of Statistical Software*. 2004;9(8):1–19. doi:10.18637/jss.v009.i08

Lumley T. *Complex Surveys: A Guide to Analysis Using R*. Wiley; 2010. ISBN: 9781118210932.

Lumley T. survey: Analysis of complex survey samples. R package. https://CRAN.R-project.org/package=survey

Rubin DB. *Multiple Imputation for Nonresponse in Surveys*. Wiley; 1987.

See Also

Useful links:

- https://github.com/jncohen/scf
- Report bugs at https://github.com/jncohen/scf/issues

`scf_corr` *Estimate Correlation Between Two Continuous Variables in SCF Microdata*

Description

Estimates the weighted Pearson correlation between two variables, with a t-test of zero correlation.

Usage

```
scf_corr(scf, var1, var2, variance = "fed")

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

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`var1`	One-sided formula specifying the first variable
`var2`	One-sided formula specifying the second variable
`variance`	`"fed"` (default) or `"rubin"`: how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.
`object`	A `scf_corr` object returned by `scf_corr()`.
`...`	Currently unused; included for S3 generic compatibility.

Details

Use it to measure how closely two variables move together, such as income and net worth. r runs from -1 to 1: the sign gives the direction, values near 0 mean little linear association. A small p-value means the population correlation is unlikely to be zero. r measures linear association only, is sensitive to outliers, and does not adjust for other variables; for that, use `scf_ols()`.

In each implicate, the weighted correlation is

$$r = \frac{\sum w_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum w_i (x_i - \bar{x})^2 \sum w_i (y_i - \bar{y})^2}}$$

where w_i is the household weight and $\bar{x}$, $\bar{y}$ are weighted means. The sampling variance comes from the 999 replicate weights (see scf_variance). The five estimates are pooled (see scf_MIcombine). The t-statistic is r divided by its standard error.

Value

An object of class `scf_corr`, containing:

results Data frame with pooled correlation estimate, standard error, t-statistic, degrees of freedom, p-value, and minimum/maximum values across implicates.

imps Named vector of implicate-level correlations.

aux Variable names used in the estimation.

See Also

`scf_plot_hex()`, `scf_ols()`

Examples

```
# Correlate income and net worth
corr <- scf_corr(scf2022, ~income, ~networth)
print(corr)
summary(corr)
```

`scf_design` *Construct an SCF Multiply-Imputed Survey Object*

Description

Wraps a list of implicate designs into an `scf_mi_survey` object.

Usage

```
scf_design(design, year, n_households, dollar_year = NULL)
```

Arguments

`design`	A list of five `survey::svrepdesign()` objects (one per implicate).
`year`	Numeric SCF survey year (e.g., 2022).
`n_households`	Numeric total U.S. households represented in `year`.
`dollar_year`	Year whose dollars the Fed's summary variables, such as `income` and `networth`, are in. If `NULL`, it is found from the data by comparing `wageinc` with `x5702`.

Details

`scf_load()` calls this function. Call it directly only when building implicate designs outside the download-and-load workflow. Each element of `design` must be a `survey::svrepdesign()` object.

Value

An object of class "scf_mi_survey" with:

mi_design List of replicate-weighted designs (one per implicate).

year SCF survey year.

n_households Estimated number of U.S. households.

dollar_year Year whose dollars the summary variables are in, or NA if unknown.

See Also

scf_load(), scf_update()

Examples

```
# Construct scf_mi_survey object
obj <- scf_design(
  design = scf2022$mi_design,
  year = 2022,
  n_households = scf2022$n_households
)
class(obj)
length(obj$mi_design)
```

---

scf_download *Download and Prepare SCF Microdata for Local Analysis*

Description

Downloads SCF public-use data from the Federal Reserve and saves one .rds file per year for scf_load().

Usage

```
scf_download(years = seq(1989, 2025, 3), overwrite = FALSE, verbose = TRUE)
```

Arguments

years	Integer vector of SCF years to download (e.g., c(2016, 2019)). Defaults to all 13 waves, 1989 to 2025. Years that are not survey years are ignored.
overwrite	Logical. If TRUE, re-download and overwrite existing .rds files. Default is FALSE.
verbose	Logical. If TRUE, display progress messages. Default is TRUE.

Details

Run it once per survey year, before `scf_load()`. Requires an internet connection. Files go to the working directory; years already on disk are skipped unless `overwrite = TRUE`.

For each year, the function downloads the full public data, the summary extract, and the replicate weights; merges them by household; multiplies each replicate weight by its multiplicity factor; and saves the five implicates as a list.

The Fed puts the summary variables of every year, such as `income` and `networth`, in dollars of its latest survey: 2025 for files downloaded since October 2026, 2022 before that. Each file records this as its dollar year, which `scf_load()` reads.

Value

A character vector of paths to the `.rds` files written to disk (one per year). Each file contains a list of five implicate data frames ready for use with `scf_load()`.

References

Board of Governors of the Federal Reserve System. Codebook for the 2025 Survey of Consumer Finances. https://www.federalreserve.gov/econres/files/codebk2025.txt

See Also

`scf_load()`, `scf_design()`, `scf_update()`

Examples

```
if (interactive()) {
  # Download the 2025 survey into the working directory, then load it
  scf_download(2025)
  scf2025 <- scf_load(2025)
}
```

---

scf_freq *Estimate the Frequencies of a Discrete Variable from SCF Microdata*

Description

Estimates the share of households in each category of a discrete variable, overall or by group.

Usage

```
scf_freq(scf, var, by = NULL, percent = TRUE, variance = "fed")
```

Arguments

scf	A scf_mi_survey object from scf_load().
var	A one-sided formula specifying a categorical variable (e.g., ~racecl).
by	Optional one-sided formula specifying a discrete grouping variable (e.g., ~own).
percent	Logical. If TRUE (default), scales results and standard errors to percentages.
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it to describe how households divide across categories, such as education levels. Each estimate is the share of households in a category; with by, shares sum to 1 (or 100) within each group. A share plus or minus two standard errors gives a rough 95 percent range. No tests are reported. For two-way tables, use scf_xtab().

Proportions are estimated in each implicate with survey::svymean() and pooled across implicates (see scf_variance). Numeric variables with more than 25 distinct values are treated as continuous and give an error.

Value

A list of class "scf_freq" with:

results Pooled category proportions and standard errors, with columns group, category, proportion, se_proportion, and n. Without by, group is "All". With percent = TRUE (the default), proportion and se_proportion are percentages. n is the number of sample households (unweighted) in each row, averaged over implicates. Groups of 10 or fewer households give a warning.

imps A named list of implicate-level proportion estimates.

aux Metadata about the variable and grouping structure.

See Also

scf_xtab(), scf_plot_dist()

Examples

```
# Proportions of homeownership
scf_freq(scf2022, ~own)

# Homeownership proportions by education
scf_freq(scf2022, ~own, by = ~edcl)
```

`scf_glm` *Estimate Generalized Linear Model from SCF Microdata*

Description

Fits a generalized linear model to each implicate with `survey::svyglm()` and pools the results with Rubin's rules.

Usage

```
scf_glm(object, formula, family)
```

Arguments

`object`	A `scf_mi_survey` object from `scf_load()`.
`formula`	A valid model formula, e.g., `rich ~ age + factor(edcl)`.
`family`	A GLM family object such as `poisson()`, `quasibinomial()`, or `Gamma()`. Required; there is no default.

Details

Use it for models that `scf_ols()` and `scf_logit()` do not cover: counts with `poisson()`, shares between 0 and 1 with `quasibinomial()`, or positive, skewed amounts with `Gamma(link = "log")`. For linear regression use `scf_ols()`; for logistic regression with odds ratios use `scf_logit()`. `family` sets the link and error distribution and must be given.

Coefficients are on the link scale (log-odds for `binomial()`, log counts for `poisson()`). A positive coefficient means the outcome rises with the predictor, holding the others constant. The p-value tests whether a coefficient is zero, using a t distribution with Rubin's degrees of freedom. AIC is averaged across implicates; compare it only across models of the same outcome, where lower is better. Pooling is described in `scf_MIcombine()`.

Each distinct warning from `svyglm()` is reported once, with the implicates it came from. "non-integer #successes in a binomial glm!", which survey weights always produce, is dropped. The function stops if the model fails in any implicate.

Value

An object of class `"scf_glm"` and `"scf_model_result"` with:

results A data frame of pooled coefficients, standard errors, t-values, p-values, and significance stars. P-values use a t distribution with Rubin's degrees of freedom (see scf_MIcombine).

fit Fit statistics: mean and SD of AIC (`AIC`, `AIC.sd`) across implicates, pseudo-R-squared (`pseudo_r2`, `pseudo_r2.sd`; binomial models only), observations per implicate (`nobs`), and their mean (`nobs_mean`).

models A list of implicate-level `svyglm` model objects.

call The matched function call.

See Also

scf_ols(), scf_logit(), scf_regtable()

Examples

```
# Run logistic regression
model <- scf_glm(scf2022, own ~ hhsex, family = binomial())
summary(model)
```

---

scf_implicates *Extract Implicate-Level Estimates from SCF Results*

Description

Returns the five implicate-level estimates behind an scf result.

Usage

```
scf_implicates(x, long = FALSE)
```

Arguments

x	A result object containing implicate-level estimates. Three types are supported: **Descriptive results** Objects from scf_freq(), scf_mean(), scf_median(), scf_percentile() (not method = "stack"), and scf_ratio(). Also scf_corr() (one correlation per implicate) and scf_xtab() (weighted cell counts per implicate). **Survey statistic results** Objects whose $imps slot contains svystat objects (e.g., raw outputs from survey::svymean()). **Model results** Objects from scf_ols(), scf_glm(), scf_logit(), and scf_quantreg().
long	Logical. If TRUE, returns stacked data frame. If FALSE, returns list.

Details

Use it to check how much imputation affects a result. Implicate estimates close together mean imputation adds little uncertainty; estimates far apart mean the result depends on imputed values, and its standard error is larger. Works on results from the descriptive functions and the models.

Value

A list of implicate-level data frames, or a single stacked data frame if long = TRUE.

Examples

```
# Extract implicate-level results
out <- scf_freq(scf2022, ~own)
scf_implicates(out, long = TRUE)
```

`scf_load` *Load SCF Data as Multiply-Imputed Survey Designs*

Description

Loads `.rds` files made by `scf_download()` as `scf_mi_survey` objects.

Usage

```
scf_load(min_year, max_year = min_year, data_directory = ".")
```

Arguments

`min_year`	Integer. First SCF year to load (1989 to 2025).
`max_year`	Integer. Last SCF year to load. Defaults to `min_year`.
`data_directory`	Character. Directory containing the `.rds` files. Defaults to the current working directory `"."`. For examples and tests, use `tempdir()` to avoid leaving files behind.

Details

Start every analysis here and assign the result, for example `scf2022 <- scf_load(2022)`. Printing the object shows the year, the number of households represented, and the number of implicates.

Each file holds five implicate data frames with the weight `wgt` and replicate weights `wt1b1` to `wt1b999`. Each implicate's design uses `wgt * 5`, because the Fed divides `wgt` by 5 so the five implicates sum to the population. The `wgt` column is not changed.

The object records the dollar year of the summary variables, which `scf_real()` and `scf_nominal()` use. Files made by older versions of the package do not store it, so it is found from the data. Loading years in different dollar years gives a warning.

Value

Invisibly, a `scf_mi_survey` object (or a named list of them if several years are loaded), with elements `mi_design`, `year`, `n_households`, and `dollar_year`, and a logical `mock` attribute.

See Also

`scf_download()`, `scf_design()`, `scf_update()`, `survey::svrepdesign()`

Examples

```
scf2022 <- scf_load(2022)
scf2022
```

`scf_logit` *Estimate Logistic Regression Model using SCF Microdata*

Description

Fits a logistic regression to each implicate and pools the results with Rubin's rules. Returns odds ratios by default.

Usage

```
scf_logit(object, formula, odds = TRUE)
```

Arguments

`object`	A `scf_mi_survey` object from `scf_load()`.
`formula`	A model formula specifying a binary outcome and predictors, e.g., `rich ~ age + factor(edcl)`.
`odds`	Logical. If `TRUE` (default), exponentiates coefficient estimates to produce odds ratios.

Details

Use it for binary outcomes, such as homeownership. An odds ratio above 1 means higher odds of the outcome per one-unit increase in the predictor; below 1, lower odds. An odds ratio of 1.5 means 50 percent higher odds. The p-value tests an odds ratio of 1 (no association).

Calls `scf_glm()` with `family = binomial()`. With `odds = TRUE`, estimates are exponentiated and standard errors converted to the odds-ratio scale. `vcov()` stays on the log-odds scale; `confint()` computes intervals on that scale and exponentiates them.

Value

An object of class `"scf_logit"` and `"scf_model_result"` with:

results A data frame of pooled estimates (log-odds or odds ratios), standard errors, and test statistics.

fit Fit statistics: mean and SD of AIC (`AIC`, `AIC.sd`) and pseudo-R-squared (`pseudo_r2`, `pseudo_r2.sd`) across implicates, observations per implicate (`nobs`), and their mean (`nobs_mean`).

imps List of implicate-level `svyglm` model objects.

call The matched function call.

See Also

scf_glm(), scf_ols(), scf_MIcombine()

Examples

```
# Run logistic regression
model <- scf_logit(scf2022, own ~ hhsex)
summary(model)
```

---

scf_mean *Estimate Mean in Multiply-Imputed SCF Data*

Description

Estimates the weighted mean of a continuous variable, overall or by group.

Usage

```
scf_mean(scf, var, by = NULL, verbose = FALSE, variance = "fed")
```

Arguments

scf	A scf_mi_survey object from scf_load().
var	A one-sided formula identifying the continuous variable to summarize (e.g., ~networth).
by	Optional one-sided formula specifying a discrete grouping variable for stratified means.
verbose	Logical. If TRUE, include implicate-level results in print output. Default is FALSE.
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it for the average value of a variable in the population or a group. The estimate is in the variable's units. Summary dollar variables, such as networth, are in the data's dollar year (see scf_load()); raw x variables are in nominal dollars (see scf_real()). min and max give the range of the implicate estimates. Wealth and income are right-skewed, so a few very wealthy households pull their means well above their medians; report scf_median() as well.

The mean is estimated in each implicate with survey::svymean() and pooled across implicates (see scf_variance).

Value

A list of class "scf_mean" with:

results Pooled estimates with standard errors and range across implicates. One row per group, or one row total. `n` is the number of sample households (unweighted) in each row, averaged over implicates. Groups of 10 or fewer households give a warning.

imps A named list of implicate-level estimates.

aux Variable and group metadata.

See Also

`scf_median()`, `scf_percentile()`, `scf_xtab()`, `scf_plot_dist()`

Examples

```
# Estimate means
scf_mean(scf2022, ~networth)
scf_mean(scf2022, ~networth, by = ~edcl)
```

---

scf_median *Estimate the Population Median of a Continuous SCF Variable*

Description

Estimates the weighted median of a continuous variable, overall or by group.

Usage

```
scf_median(
  scf,
  var,
  by = NULL,
  verbose = FALSE,
  method = c("implicate", "stack"),
  variance = "fed"
)
```

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`var`	A one-sided formula specifying the continuous variable of interest (e.g., `~networth`).
`by`	Optional one-sided formula for a categorical grouping variable.
`verbose`	Logical; if TRUE, show implicate-level results.
`method`	`"implicate"` (default) or `"stack"`. See Details.
`variance`	`"fed"` (default) or `"rubin"`: how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it for skewed variables such as wealth and income, where it describes the typical household better than the mean. Half of households are below the estimate and half above. Summary dollar variables, such as `networth`, are in the data's dollar year (see `scf_load()`); raw `x` variables are in nominal dollars (see `scf_real()`).

Calls `scf_percentile()` with `q = 0.5`. With `method = "implicate"`, the estimate is the mean of the five implicate medians; the standard error combines sampling and imputation variance (see scf_variance). With `method = "stack"`, the five implicates are stacked and one weighted median is taken, as in the Fed's published tables; no standard error is given. The two can differ in small groups.

Value

A list of class `c("scf_median", "scf_percentile")` with:

results A data frame with pooled medians (`quantile` is 0.5), standard errors, and range across implicates. With `method = "stack"`, `se`, `min`, and `max` are `NA` and `imps` is `NULL`. `n` is the number of sample households (unweighted) in each row, averaged over implicates. Groups of 10 or fewer households give a warning.

imps A list of implicate-level results.

aux Variable and grouping metadata.

See Also

`scf_percentile()`, `scf_mean()`

Examples

```
# Estimate medians
scf_median(scf2022, ~networth)
scf_median(scf2022, ~networth, by = ~edcl)
scf_median(scf2022, ~networth, method = "stack")
```

`scf_MIcombine` *Combine Estimates Across SCF Implicates*

Description

The SCF comes as five implicates: five complete copies of the data that differ only in the values the Fed filled in for missing answers. Every `scf` estimate is computed on each implicate and the five results are combined into a consolidated, final result. This page explains how these results are produced. To see how variance is computed, see scf_variance.

`scf_MIcombine()` does the combining for the regression models. It can also be called directly on any list of implicate results. For univariate and bivariate inferential statistics, the results are

combined within the functions that compute them, such as scf_mean(), scf_ratio(), as these operations can differ depending on the statistic. Call it yourself when you need a statistic or model that scf does not provide: compute it on each implicate and pass the list of results.

Usage

```
scf_MIcombine(results, variances, call = sys.call(), df.complete = Inf)

## S3 method for class 'scf_MIresult'
SE(object, ...)
```

Arguments

results	A list of implicate-level results. Each element is an object with coef() and vcov() methods, or a named numeric vector when variances is given.
variances	Optional list of variance-covariance matrices. If omitted, extracted using vcov().
call	Optional. The originating function call. Defaults to sys.call().
df.complete	Optional degrees of freedom for the complete-data model. Used for small-sample corrections. Defaults to Inf, assuming large-sample asymptotics.
object	A pooled result object of class "scf_MIresult" (for SE()).
...	Not used.

Value

An object of class "scf_MIresult" with components:

coefficients Pooled point estimates across implicates.

variance Pooled variance-covariance matrix.

df Degrees of freedom for each parameter.

missinfo Estimated fraction of missing information for each parameter.

nimp Number of implicates used in pooling.

call The function call.

Supports coef(), vcov(), confint() (t distribution with Rubin's degrees of freedom), and SE().

The estimate

The final estimate is the average of the implicate estimates:

$$\bar{Q} = \frac{1}{m} \sum_{i=1}^{m} Q_i$$

where Q_i is the estimate from implicate i and m is the number of implicates (5).

The standard error

The total variance adds two parts:

$$T = U + \left(1 + \frac{1}{m}\right) B$$

U is the sampling variance, from the replicate weights (see scf_variance). B is the variance of the m implicate estimates around $\bar{Q}$, which measures how much the filled-in values move the answer. The factor $1 + 1/m$ allows for having only five implicates. The standard error is $\sqrt{T}$. The Fed's codebook uses the same formula, written as $U + (6/5)B$.

Tests and confidence intervals

Tests and intervals use a t distribution. Its degrees of freedom are:

$$\nu = (m - 1)\left(1 + \frac{U}{(1 + 1/m)B}\right)^2$$

It depends on the ratio of sampling variance (U) to imputation variance (B). When imputation adds little variance, ν is large and the t is close to normal. When it adds a lot, ν can be small. If `df.complete` is given, the Barnard-Rubin small-sample version is used instead.

The fraction of missing information (`missinfo`) is the share of the total variance that comes from the missing data. Values near 0 mean the imputation adds little uncertainty. Large values mean the result leans heavily on imputed answers.

Regression models

`scf_ols()`, `scf_glm()`, `scf_logit()`, and `scf_quantreg()` fit the model separately on each implicate, then combine the results the same way, with vectors and matrices in place of single numbers:

- The coefficients are the average of the implicate coefficients.
- U is the average of the implicate variance-covariance matrices.
- B is the covariance matrix of the coefficients across implicates.
- $T = U + (1 + 1/m)B$, and each coefficient gets its own ν.

This is what the Fed's codebook does for regressions (the MISECOMP SAS macro). Models always average U over all implicates.

Fit statistics, such as R-squared, are summarized across implicates rather than combined with Rubin's rules. Each model's help page says how.

Groups missing from some implicates

In the descriptive functions, such as `scf_mean()`, a group can be empty in some implicates, for example when a subset uses an imputed variable. Its estimate then combines only the implicates that have it, and m is the number of those implicates. `scf_MIcombine()` itself stops if any implicate result is `NA`.

References

Barnard J, Rubin DB. Small-sample degrees of freedom with multiple imputation. *Biometrika*. 1999;86(4):948–955. doi:10.1093/biomet/86.4.948

Board of Governors of the Federal Reserve System. Codebook for the 2025 Survey of Consumer Finances. https://www.federalreserve.gov/econres/scfindex.htm

Little RJA, Rubin DB. *Statistical Analysis with Missing Data*. 3rd ed. Wiley; 2019. ISBN: 9780470526798.

Rubin DB. *Multiple Imputation for Nonresponse in Surveys*. Wiley; 1987.

Examples

```
# Pool simple survey mean for mock data
outlist <- lapply(scf2022$mi_design, function(d) survey::svymean(~I(age >= 65), d))
pooled  <- scf_MIcombine(outlist)     # vcov/coef extracted automatically
SE(pooled); coef(pooled)
```

`scf_model_result_methods`

S3 Methods for scf_model_result Objects

Description

Generic S3 methods dispatched on objects of class `"scf_model_result"`, as returned by `scf_ols()`, `scf_glm()`, `scf_logit()`, and `scf_quantreg()`. Use them as you would on any fitted model in R.

`coef()` Pooled coefficient estimates (Rubin's Rules).

`vcov()` Pooled variance-covariance matrix. For `scf_logit` it is on the log-odds scale, even when the estimates are odds ratios.

`confint()` Confidence intervals from the t distribution with Rubin's degrees of freedom. For odds ratios, the interval is computed on the log-odds scale and then exponentiated.

`AIC()` Mean AIC across implicates. Compare it only across models fit to the same outcome and data; lower is better.

`residuals()` Residuals from the first implicate model (diagnostic use only).

`predict()` Mean predictions pooled across all five implicate models.

`formula()` The model formula.

`SE()` Pooled standard errors, the square roots of the diagonal of `vcov()`. For `scf_logit` they are on the log-odds scale, unlike `std.error` in `results`, which is on the odds-ratio scale.

Usage

```
## S3 method for class 'scf_model_result'
formula(x, ...)

## S3 method for class 'scf_model_result'
residuals(object, ...)

## S3 method for class 'scf_model_result'
coef(object, ...)

## S3 method for class 'scf_model_result'
vcov(object, ...)

## S3 method for class 'scf_model_result'
AIC(object, k = 2, ...)

## S3 method for class 'scf_model_result'
predict(object, newdata, type = "link", ...)

## S3 method for class 'scf_model_result'
confint(object, parm, level = 0.95, ...)
```

Arguments

x	An object of class "scf_model_result" (for formula).
...	Additional arguments (not used by most methods).
object	An object of class "scf_model_result".
k	Not used. The AIC is the mean of the implicate AICs, with the usual penalty of 2.
newdata	Optional data frame of new observations for predict(). If missing, predictions are made on the original training data.
type	Prediction scale for predict(): "link" (default) or "response". For a logistic model, "response" gives predicted probabilities.
parm	Coefficients to include, by name or position. Defaults to all.
level	Confidence level for confint(). Defaults to 0.95.

---

scf_nominal *Convert SCF Dollar Estimates to Nominal Survey-Year Dollars*

Description

Converts dollar estimates from real dollars to dollars of the survey year.

Usage

```
scf_nominal(x, from_year = NULL, force = FALSE)
```

Arguments

x	An object of class scf_mean, scf_median, scf_percentile, or scf_ttest.
from_year	Integer. Year whose real-dollar units are assumed for the input estimate. Defaults to the dollar year of the data, as recorded by scf_load().
force	Logical. If TRUE, skip the check on the variable.

Details

Estimates, standard errors, and confidence limits are multiplied by the CPI-U-RS for the survey year divided by that for from_year. Works on results from scf_mean(), scf_median(), scf_percentile(), and scf_ttest().

The Fed's summary variables, such as income and networth, are in the dollars of the Fed's latest survey: 2025 for files downloaded since October 2026, 2022 before that (see scf_download()). Raw x variables are already nominal. Income in the summary extract is first moved to survey-year dollars, so converted income is in survey-year dollars, as in the Fed's nominal tables. Medians in Tables 1 and 4 match the Fed within 0.05 percent for 1989 to 2025, except a few medians near zero, where the Fed's rounding is larger than that.

The function stops, unless force = TRUE, if the result is for a raw x variable or for a variable not known to be in real dollars (the Fed's dollar summary variables and names ending in "_real"). Converting twice gives a warning.

Value

The input object with dollar estimates, standard errors, and confidence intervals rescaled to nominal survey-year dollars. Attributes "nominal" and "from_year" are set on the returned object.

References

Board of Governors of the Federal Reserve System. SAS macro for SCF Bulletin variable definitions (bulletin.macro.txt). https://www.federalreserve.gov/econres/files/bulletin.macro.txt

U.S. Bureau of Labor Statistics. Consumer Price Index research series using current methods (CPI-U-RS). https://www.bls.gov/cpi/research-series/r-cpi-u-rs-home.htm

See Also

scf_real(), scf_mean(), scf_median(), scf_percentile(), scf_ttest()

Examples

```
# Convert a mean estimate to nominal survey-year dollars
result <- scf_mean(scf2022, ~networth)
result_nominal <- scf_nominal(result)
```

```
# Works with median and percentile results
med <- scf_median(scf2022, ~networth)
med_nominal <- scf_nominal(med)
```

`scf_ols` *Estimate an Ordinary Least Squares Regression on SCF Microdata*

Description

Fits a linear regression to each implicate with `survey::svyglm()` and pools the results with Rubin's rules.

Usage

```
scf_ols(object, formula)
```

Arguments

`object`	A `scf_mi_survey` object from `scf_load()`.
`formula`	A model formula specifying a continuous outcome and predictor variables (e.g., `networth ~ income + age`).

Details

Use it for continuous outcomes. For binary outcomes use `scf_logit()`; for skewed outcomes such as wealth, consider `scf_quantreg()` or a log transform.

Each coefficient is the expected change in the outcome for a one-unit increase in the predictor, holding the others constant. The p-value tests whether a coefficient is zero, using a t distribution with Rubin's degrees of freedom. R-squared is the share of variance explained and AIC compares models of the same outcome (lower is better); both are averaged across implicates. Pooling is described in `scf_MIcombine()`.

Value

An object of class `"scf_ols"` and `"scf_model_result"` with:

results A data frame of pooled coefficients, standard errors, t-values, p-values, and significance stars.

fit Fit statistics: mean and SD of AIC (`AIC`, `AIC.sd`) and R-squared (`r.squared`, `r.squared.sd`) across implicates, observations per implicate (`nobs`), and their mean (`nobs_mean`).

imps A list of implicate-level `svyglm` model objects.

call The matched call used to produce the model.

See Also

`scf_glm()`, `scf_logit()`, `scf_MIcombine()`

Examples

```
# Run OLS model
model <- scf_ols(scf2022, networth ~ income + age)
summary(model)
```

`scf_pctile_sum` *Summarize SCF Variables by Percentile Group*

Description

Groups households by percentiles of a variable and summarizes a variable within each group.

Usage

```
scf_pctile_sum(
  scf,
  var,
  probs = seq(0, 1, by = 0.1),
  labels = NULL,
  varname = NULL,
  method = c("implicate", "stack"),
  stat = c("mean", "median", "none"),
  stat_var = NULL
)
```

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`var`	A one-sided formula naming the continuous variable to cut (e.g., `~networth`).
`probs`	Numeric vector of values between 0 and 1 defining group boundaries, including 0 and 1 as endpoints. Defaults to deciles (`seq(0, 1, by = 0.1)`).
`labels`	Optional character vector of group labels, length equal to `length(probs) - 1`. If `NULL` (default), labels are generated automatically in the form `"p0-p10"`, `"p10-p20"`, etc.
`varname`	Optional name for the new grouping variable. Defaults to `"{var}_pctile"` (e.g., `"networth_pctile"`).
`method`	Character. One of `"implicate"` (default) or `"stack"`. See Details.
`stat`	`"mean"` (default), `"median"`, or `"none"`. With `"none"`, no statistic is computed and the data are returned with the group variable added.
`stat_var`	Optional one-sided formula naming the variable to summarize within each group. Defaults to `var` if not supplied.

Details

Use it to compare households at different points of a distribution, such as mean net worth of the top 10 percent against the bottom 50 percent. Each row is one group; the estimate is the mean or median of `stat_var` in that group. For the Fed's own groups, such as `nwcat` and `inccat`, use by in `scf_mean()` or `scf_median()` instead.

Households are sorted by the variable and household identifier, and the cumulative weighted population share sets each household's group, as in the Fed's SAS macro. Ties at a boundary are split by identifier, so each group holds its share of the population.

`method = "implicate"` assigns groups within each implicate and returns estimates with standard errors. `method = "stack"` stacks the implicates with weights divided by five, assigns groups once, and computes a weighted statistic without a standard error; it reproduces the Fed's published figures. With `stat = "none"`, the data are returned with the group variable added.

Value

With `stat = "none"`, the `scf_mi_survey` object with the group variable (named by `varname`) added; pass it to by in other functions. With `method = "implicate"`, the result of `scf_mean()` or `scf_median()`. With `method = "stack"`, a data frame with columns `group`, `variable`, and `estimate`.

References

Board of Governors of the Federal Reserve System. SAS macro for SCF Bulletin variable definitions (bulletin.macro.txt). https://www.federalreserve.gov/econres/files/bulletin.macro.txt

See Also

`scf_mean()`, `scf_median()`, `scf_percentile()`, `scf_update_by_implicate()`

Examples

```
# Mean net worth, top 10 vs bottom 90 percent, stack method (fast)
scf_pctile_sum(scf2022, ~networth,
               probs  = c(0, 0.9, 1),
               labels = c("bottom90", "top10"),
               method = "stack")

# Implicate method (default)
scf_pctile_sum(scf2022, ~networth,
               probs  = c(0, 0.9, 1),
               labels = c("bottom90", "top10"))

# Return grouping variable only, no summary statistic (implicate method)
scf2022 <- scf_pctile_sum(scf2022, ~networth,
                          probs  = c(0, 0.9, 1),
                          labels = c("bottom90", "top10"),
                          stat   = "none")
```

`scf_percentile` *Estimate Percentiles in SCF Microdata*

Description

Estimates a weighted percentile of a continuous variable, overall or by group.

Usage

```
scf_percentile(
  scf,
  var,
  q = 0.5,
  by = NULL,
  verbose = FALSE,
  method = c("implicate", "stack"),
  variance = "fed"
)
```

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`var`	A one-sided formula naming the continuous variable to summarize (for example `~networth`).
`q`	Percentile, between 0 and 1. Default 0.5 (median).
`by`	Optional one-sided formula naming a categorical grouping variable. If supplied, the percentile is estimated separately within each group.
`verbose`	Logical. If TRUE, print implicate-level estimates. Default FALSE.
`method`	`"implicate"` (default) or `"stack"`. See Details.
`variance`	`"fed"` (default) or `"rubin"`: how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it for points of a distribution other than the middle, such as the 90th percentile of net worth. With `q = 0.9`, 90 percent of households are below the estimate. Summary dollar variables, such as `networth`, are in the data's dollar year (see `scf_load()`); raw `x` variables are in nominal dollars (see `scf_real()`).

`method = "implicate"` estimates the percentile in each implicate with `survey::svyquantile()` and pools the estimates (see scf_variance). `method = "stack"` stacks the implicates with weights divided by five and computes one weighted percentile without a standard error; it reproduces the Fed's published figures.

Warnings from `survey::svyquantile()` are passed on with the implicate number. One notice, "Not all replicate weight designs give valid standard errors for quantiles", is dropped; it concerns jackknife weights, and the SCF's are bootstrap weights.

Value

An object of class "scf_percentile" containing:

results A data frame containing pooled percentile estimates, pooled standard errors, and implicate min/max values. One row per group (if by is supplied) or one row otherwise. With method = "stack", se, min, and max are NA and imps is NULL. n is the number of sample households (unweighted) in each row, averaged over implicates. Groups of 10 or fewer households give a warning.

imps A list of implicate-level percentile estimates and standard errors.

aux A list containing the variable name, optional group variable name, the quantile requested, the survey year, and the dollar year.

verbose Logical flag indicating whether implicate-level estimates should be printed by print() or summary().

References

Board of Governors of the Federal Reserve System. SAS macro for SCF Bulletin variable definitions (bulletin.macro.txt). https://www.federalreserve.gov/econres/files/bulletin.macro.txt

See Also

scf_median(), scf_mean()

Examples

```
# Estimate the 75th percentile of net worth
scf_percentile(scf2022, ~networth, q = 0.75)

# Estimate the median net worth by ownership group
scf_percentile(scf2022, ~networth, q = 0.5, by = ~own)
```

---

scf_plot_bbar *Stacked Bar Chart of Two Discrete Variables in SCF Data*

Description

Plots a pooled two-way table as stacked bars.

Usage

```
scf_plot_bbar(
  design,
  rowvar,
  colvar,
  scale = c("percent", "count"),
  percent_by = c("total", "row", "col"),
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  fill_colors = NULL,
  row_labels = NULL,
  col_labels = NULL
)
```

Arguments

design	A scf_mi_survey object from scf_load().
rowvar	A one-sided formula for the x-axis grouping variable (e.g., ~edcl).
colvar	A one-sided formula for the stacked fill variable (e.g., ~racecl).
scale	Character. One of "percent" (default) or "count".
percent_by	Character. One of "total" (default), "row", or "col". Sets what the percentages are shares of when scale = "percent".
title	Optional character string for the plot title.
xlab	Optional character string for the x-axis label.
ylab	Optional character string for the y-axis label.
fill_colors	Optional vector of fill colors to pass to ggplot2::scale_fill_manual().
row_labels	Optional named vector to relabel row categories (x-axis).
col_labels	Optional named vector to relabel col categories (legend).

Details

Use it to show how two discrete variables relate, such as homeownership by education. Each bar is a category of rowvar, split by colvar. Heights are percentages of all households, or of each row or column (percent_by), or estimated counts (scale = "count"). Estimates come from scf_xtab().

Value

A ggplot2 object.

Examples

```
# Stacked bar chart: education by ownership
scf_plot_bbar(scf2022, ~own, ~edcl)
```

```
# Column percentages instead of total percent
scf_plot_bbar(scf2022, ~own, ~edcl, percent_by = "col")

# Raw counts (estimated number of households)
scf_plot_bbar(scf2022, ~own, ~edcl, scale = "count")
```

`scf_plot_cbar` *Bar Plot of Summary Statistics by Grouping Variable in SCF Data*

Description

Plots a mean, median, or percentile of a continuous variable for each group.

Usage

```
scf_plot_cbar(
  design,
  yvar,
  xvar,
  stat = "mean",
  method = c("implicate", "stack"),
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  fill = "#0072B2",
  angle = 30,
  label_map = NULL
)
```

Arguments

`design`	A `scf_mi_survey` object from `scf_load()`.
`yvar`	One-sided formula for the continuous variable (e.g., `~networth`).
`xvar`	One-sided formula for the grouping variable (e.g., `~racecl`).
`stat`	`"mean"` (default), `"median"`, or a quantile (numeric between 0 and 1).
`method`	For medians and percentiles, `"implicate"` (default) or `"stack"`. See `scf_percentile()`.
`title`	Plot title (optional).
`xlab`	X-axis label (optional).
`ylab`	Y-axis label (optional).
`fill`	Bar fill color. Default is `"#0072B2"`.
`angle`	Angle of x-axis labels. Default is 30.
`label_map`	Optional named vector to relabel x-axis category labels.

Details

Use it to show how a variable differs across groups, such as median net worth by education. Bar heights are point estimates from `scf_mean()`, `scf_median()`, or `scf_percentile()` with by; run those functions for standard errors. Use `method = "stack"` for medians and percentiles that match the Fed's published tables.

Value

A `ggplot2` object.

See Also

`scf_mean()`, `scf_median()`, `scf_percentile()`, `scf_theme()`

Examples

```
# Visualize 90th percentile of income by education
scf_plot_cbar(scf2022, ~income, ~edcl, stat = 0.9, fill = "#D55E00")
```

`scf_plot_dbar` *Plot Bar Chart of a Discrete Variable from SCF Data*

Description

Plots the percent of households in each category of a discrete variable.

Usage

```
scf_plot_dbar(
  design,
  variable,
  title = NULL,
  xlab = NULL,
  ylab = "Percent",
  angle = 30,
  fill = "#0072B2",
  label_map = NULL
)
```

Arguments

`design`	A `scf_mi_survey` object from `scf_load()`.
`variable`	A one-sided formula specifying a categorical variable (e.g., `~racecl`).
`title`	Optional character string for the plot title. Default: `"Distribution of <variable>"`.
`xlab`	Optional x-axis label. Default: variable name.

ylab	Optional y-axis label. Default: "Percent".
angle	Integer. Rotation angle for x-axis labels. Default is 30.
fill	Fill color for bars. Default is "#0072B2".
label_map	Optional named vector to relabel x-axis category labels.

Details

Use it to show how households divide across categories, such as education levels. Estimates come from scf_freq().

Value

A ggplot2 object representing the pooled bar chart.

See Also

scf_freq(), scf_plot_bbar(), scf_xtab()

Examples

```
# Bar chart of education categories
scf_plot_dbar(scf2022, ~edcl)
```

---

scf_plot_dist *Plot a Univariate Distribution of an SCF Variable*

Description

Plots the percent of households in each category or bin of a variable.

Usage

```
scf_plot_dist(
  design,
  variable,
  bins = 30,
  title = NULL,
  xlab = NULL,
  ylab = "Percent",
  angle = 30,
  fill = "#0072B2",
  labels = NULL
)
```

Arguments

design	A scf_mi_survey object from scf_load().
variable	A one-sided formula specifying the variable to plot.
bins	Approximate number of bins for continuous variables; the breaks are set by pretty(). Default is 30.
title	Optional plot title.
xlab	Optional x-axis label.
ylab	Optional y-axis label. Default is "Percent".
angle	Angle for x-axis tick labels. Default is 30.
fill	Fill color for bars. Default is "#0072B2".
labels	Optional named vector of custom axis labels (for discrete variables only).

Details

Use it for a quick look at any variable. Factors and numeric variables with 25 or fewer values are treated as discrete and plotted with scf_freq(). Continuous variables are binned with the same breaks in every implicate and pooled.

Value

A ggplot2 object.

See Also

scf_theme()

Examples

```
# Distribution of homeownership
scf_plot_dist(scf2022, ~own)

# Distribution of age
scf_plot_dist(scf2022, ~age, bins = 10)
```

---

scf_plot_hex *Hexbin Plot of Two Continuous SCF Variables*

Description

Plots the joint distribution of two continuous variables with hexagonal bins.

Usage

```
scf_plot_hex(design, x, y, bins = 50, title = NULL, xlab = NULL, ylab = NULL)
```

Arguments

design	A scf_mi_survey object from scf_load().
x	A one-sided formula for the x-axis variable (e.g., ~income).
y	A one-sided formula for the y-axis variable (e.g., ~networth).
bins	Integer. Number of hexagonal bins in each direction. Default is 50.
title	Optional character string for the plot title.
xlab	Optional x-axis label. Defaults to the variable name.
ylab	Optional y-axis label. Defaults to the variable name.

Details

Use it in place of a scatterplot when there are too many points to see. Brighter cells hold more households; the color scale is logarithmic. The implicates are stacked with weights divided by five. Requires the **hexbin** package.

Value

A ggplot2 object.

See Also

scf_corr(), scf_plot_smooth(), scf_theme()

Examples

```
# Plot hexbin of income vs. net worth
# Note: mock data has 200 rows per implicate; hexbin output will be sparse.
# Results on full SCF data will show a meaningful joint density.
if (requireNamespace("hexbin", quietly = TRUE)) {
  scf_plot_hex(scf2022, ~income, ~networth)
}
```

---

scf_plot_hist *Histogram of a Continuous Variable in Multiply-Imputed SCF Data*

Description

Plots the pooled share of households in each bin of a continuous variable.

Usage

```
scf_plot_hist(
  design,
  variable,
  bins = 30,
  xlim = NULL,
  title = NULL,
  xlab = NULL,
  ylab = "Proportion of Households",
  fill = "#0072B2"
)
```

Arguments

design	A scf_mi_survey object from scf_load().
variable	A one-sided formula indicating the numeric variable to plot.
bins	Number of bins (default: 30).
xlim	Optional numeric range. Values outside will be included in edge bins.
title	Optional plot title.
xlab	Optional x-axis label. Defaults to the variable name.
ylab	Optional y-axis label. Defaults to "Proportion of Households".
fill	Fill color for bars (default: "#0072B2").

Details

Use it to see the shape of a variable such as age. For very skewed variables, such as net worth, set xlim; values outside it are moved to the nearest endpoint. Empty bins stay on the axis. Bins use the same breaks in every implicate and are pooled with scf_freq().

Value

A ggplot2 object.

See Also

scf_freq(), scf_plot_dbar(), scf_plot_smooth(), scf_update_by_implicate()

Examples

```
# Plot histogram of age
scf_plot_hist(scf2022, ~age, bins = 10)
```

scf_plot_smooth *Smoothed Distribution Plot of a Continuous Variable in SCF Data*

Description

Plots a smoothed curve of the share of households near each value.

Usage

```
scf_plot_smooth(
  design,
  variable,
  binwidth = NULL,
  xlim = NULL,
  method = "loess",
  span = 0.2,
  color = "blue",
  xlab = NULL,
  ylab = "Percent of Households",
  title = NULL
)
```

Arguments

design	A scf_mi_survey object from scf_load().
variable	A one-sided formula specifying a continuous variable (e.g., ~networth).
binwidth	Optional bin width. Default uses Freedman-Diaconis rule.
xlim	Optional numeric vector of length 2. Limits the plotted range; shares are still computed over all households.
method	Character. Smoothing method: "loess" (default) or "lm".
span	Numeric LOESS span. Default is 0.2. Ignored if method = "lm".
color	Line color. Default is "blue".
xlab	Optional label for x-axis. Defaults to the variable name.
ylab	Optional label for y-axis. Defaults to "Percent of Households".
title	Optional plot title.

Details

Use it to see the shape, skew, or peaks of a variable such as age or income. The curve shows shape, not exact shares. The implicates are stacked with weights divided by five and binned (by default with the Freedman-Diaconis width), and a smoothing curve is fit to the bin shares.

Value

A ggplot2 object.

See Also

scf_plot_hist(), scf_plot_dist(), scf_theme()

Examples

```
# Plot smoothed distribution
scf_plot_smooth(
  scf2022,
  ~networth,
  xlim = c(0, 2e6),
  method = "loess",
  span = 0.25
)
```

---

scf_prop_test *Test a Proportion in SCF Data*

Description

Tests a proportion against a null value, or the difference between two groups.

Usage

```
scf_prop_test(
  design,
  var,
  group = NULL,
  p = NULL,
  alternative = c("two.sided", "less", "greater"),
  conf.level = 0.95,
  variance = "fed"
)
```

Arguments

design	A scf_mi_survey object from scf_load().
var	A one-sided formula indicating a binary variable (e.g., ~rich).
group	Optional one-sided formula indicating a binary grouping variable (e.g., ~female). If omitted, a one-sample test is performed.
p	Null hypothesis value. Defaults to 0.5 for one-sample, 0 for two-sample tests.
alternative	Character. One of "two.sided" (default), "less", or "greater".
conf.level	Confidence level for the confidence interval. Default is 0.95.
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it for yes/no variables, such as whether a household owns its home, and to compare two groups, such as homeownership among men and women. The one-sample estimate is the share with the trait; the two-sample estimate is the first group's share minus the second's. A small p-value means a difference this large is unlikely under the null hypothesis; the confidence interval gives the plausible range.

The variable must be logical or coded 0/1. Proportions are estimated in each implicate with the replicate weights and pooled (see scf_variance). The two-sample test accounts for the covariance between groups.

Value

An object of class `"scf_prop_test"` with:

results Estimate, standard error, t-statistic, degrees of freedom, p-value, confidence interval, and stars. For two-sample tests, the estimate is the first group minus the second.

proportions (Only in two-sample tests) A data frame of pooled proportions by group.

fit A list describing the method, null value, alternative hypothesis, and confidence level.

See Also

`scf_ttest()`, `scf_mean()`, `scf_MIcombine()`

Examples

```
# Wrangle data for example
scf2022 <- scf_update(scf2022,
  rich   = networth > 1e6,
  female = factor(hhsex, levels = 1:2, labels = c("Male","Female")),
  over50 = age > 50
)

# One-sample test
scf_prop_test(scf2022, ~rich, p = 0.10)

# Two-sample test
scf_prop_test(scf2022, ~rich, ~female, alternative = "less")
```

`scf_quantreg` *Estimate a Quantile Regression Model on SCF Microdata*

Description

Fits a weighted quantile regression to each implicate with `quantreg::rq()` and pools the results with Rubin's rules.

Usage

```
scf_quantreg(
  object,
  formula,
  tau = 0.5,
  se = c("replicate", "nid", "iid", "ker", "boot"),
  ...
)
```

Arguments

object	A scf_mi_survey object from scf_load().
formula	A model formula specifying the outcome and predictors, e.g., networth ~ age + factor(edcl). The outcome should be numeric.
tau	Quantile to estimate, between 0 and 1. Default 0.5 (median).
se	Within-implicate covariance method: "replicate" (default), "nid", "iid", "ker", or "boot". See Details.
...	Additional arguments passed to quantreg::rq().

Details

Use it for skewed outcomes such as wealth and income, where the mean is pulled by a few very wealthy households. Each coefficient is the change in the tau quantile of the outcome per one-unit increase in the predictor; with tau = 0.5, the change in the median. Comparing several values of tau shows whether a predictor matters more at the top or bottom of the distribution. Estimate each quantile in a separate call. P-values use a t distribution with Rubin's degrees of freedom. The function stops if the predictors are collinear.

se sets the within-implicate covariance:

"replicate" **(default)** Refits the model with each of the 999 replicate weights (see scf_variance). The only method that uses the SCF's design, and consistent for quantiles (Rust and Rao 1996). About 5,000 fits; several minutes on the full data.

"nid", "iid", "ker", "boot" From quantreg::summary.rq(). These treat the weighted sample as a simple random sample. "nid" allows the error density to vary; "iid" assumes it is constant (Koenker and Bassett 1978); "ker" uses a kernel estimate; "boot" is a pairs bootstrap. Analytic methods are unreliable at quantiles with mass points, such as low quantiles of net worth.

Fit statistics (Koenker and Machado 1999) compare the model with an intercept-only model at the same tau:

rho, rho_null Weighted check-loss of the model and the null model, averaged across implicates. Weights are scaled to mean 1.

r1 $1 - \rho/\rho_{null}$. A local measure of fit at tau, from 0 to 1.

r1_adj $1-(1-R^1)n/(n-p)$. A descriptive adjustment, not derived from Koenker and Machado.

nobs, nobs_mean Observations per implicate and their mean.

Value

An object of class "scf_quantreg" and "scf_model_result" with:

results A data frame of pooled coefficients, standard errors, t-values, p-values, and significance stars.

tau The quantile estimated.

se_method The SE method used.

fit Fit statistics: rho, rho_null, r1, r1_adj, nobs, nobs_mean. See Details.

models A list of implicate-level rq model objects for direct inspection.

imp_vcov A list of implicate-level variance-covariance matrices.

call The matched call.

formula The model formula.

References

Koenker R, Bassett G. Regression quantiles. *Econometrica*. 1978;46(1):33–50. doi:10.2307/1913643

Koenker R, Machado JAF. Goodness of fit and related inference processes for quantile regression. *Journal of the American Statistical Association*. 1999;94(448):1296–1310. doi:10.2307/2669943

Rust KF, Rao JNK. Variance estimation for complex surveys using replication techniques. *Statistical Methods in Medical Research*. 1996;5(3):283–310. doi:10.1177/096228029600500305

See Also

scf_ols(), scf_glm(), scf_MIcombine(), quantreg::rq()

Examples

```
# Median regression of net worth on age and education, with fast standard
# errors; the default, se = "replicate", uses the SCF's design but is slower
m_med <- scf_quantreg(scf2022, networth ~ age + factor(edcl), tau = 0.5,
                      se = "nid")
summary(m_med)

# Access goodness-of-fit statistics
m_med$fit$r1
m_med$fit$r1_adj
```

scf_ratio *Estimate a Ratio of Totals in SCF Microdata*

Description

Estimates the ratio of two population totals, overall or by group.

Usage

```
scf_ratio(scf, num, den, by = NULL, variance = "fed")
```

Arguments

scf	A scf_mi_survey object from scf_load().
num	A one-sided formula for the numerator (e.g., ~wageinc).
den	A one-sided formula for the denominator (e.g., ~income).
by	Optional one-sided formula for a grouping variable.
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it for aggregate shares like those in the Fed's Bulletin, such as financial assets as a share of all assets. An estimate of 0.25 means the numerator total is a quarter of the denominator total. This is not the mean of household ratios.

The ratio is sum(w * num) / sum(w * den), estimated in each implicate with survey::svyratio() and pooled (see scf_variance).

Value

An object of class "scf_ratio" with:

results Pooled ratio, standard error, range across implicates, and group ("All" without by), one row per group.

imps Implicate-level estimates.

aux Variable names and survey year.

See Also

scf_mean(), scf_variance

Examples

```
# Share of all income received by homeowners
scf2022 <- scf_update(scf2022, owner_income = income * (own == 1))
scf_ratio(scf2022, ~owner_income, ~income)
```

scf_real *Convert Raw SCF Dollar Variables to Real Dollars*

Description

Adds versions of raw `x` dollar variables in the dollars of the summary variables to every implicate.

Usage

```
scf_real(scf, vars, income = NULL, suffix = "_real", force = FALSE)
```

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`vars`	A one-sided formula naming raw dollar variables, such as `~x5729 + x3915`.
`income`	Optional one-sided formula naming the variables in `vars` that are income items, reported for the year before the survey.
`suffix`	Text added to each converted variable's name. Defaults to `"_real"`.
`force`	Logical. If `TRUE`, skip the safeguards.

Details

Follows the Fed's SCF Bulletin SAS macro. Amounts are multiplied by the September CPI-U-RS for the data's dollar year (2025 for files downloaded since October 2026; see `scf_download()`) divided by that for the survey year. Income items, reported for the prior year, are first multiplied by the ratio of annual CPI-U-RS in the survey year to the prior year; name them in `income`. In other amounts, the codes -1 and -2 are left unchanged. A converted variable equals the matching summary variable; for example, `x5702` with `income = ~x5702` equals `wageinc`.

The function stops, unless `force = TRUE`, if a variable is not a raw `x` variable, has fewer than 25 distinct values, or the new variable already exists. It cannot tell which variables are income items; check the SCF codebook.

Value

The `scf_mi_survey` object with the converted variables added. Assign it back to keep them.

References

Board of Governors of the Federal Reserve System. SAS macro for SCF Bulletin variable definitions (bulletin.macro.txt). https://www.federalreserve.gov/econres/files/bulletin.macro.txt

U.S. Bureau of Labor Statistics. Consumer Price Index research series using current methods (CPI-U-RS). https://www.bls.gov/cpi/research-series/r-cpi-u-rs-home.htm

See Also

`scf_nominal()`, `scf_update()`

Examples

```
# Wages (x5702, an income item) and the market
# value of directly held stock (x3915) in the data's dollars
scf2022 <- scf_real(scf2022, ~x5702 + x3915, income = ~x5702)
scf_median(scf2022, ~x5702_real)
```

`scf_regtable` *Format and Display Regression Results from Multiply-Imputed SCF Models*

Description

Formats one or more `scf` model results as a side-by-side table.

Usage

```
scf_regtable(
  ...,
  model.names = NULL,
  digits = NULL,
  auto_digits = is.null(digits),
  labels = NULL,
  output = c("console", "markdown", "latex", "csv"),
  file = NULL
)
```

Arguments

`...`	One or more SCF regression model objects, or a single list of such models.
`model.names`	Optional character vector naming the models. Defaults to `"Model 1"`, `"Model 2"`, etc.
`digits`	Integer. Decimal places for every estimate. If NULL (default), decimal places are chosen by size (see `auto_digits`).
`auto_digits`	Logical; if `TRUE`, uses adaptive decimal places: 0 digits for large numbers (>= 1000), 2 digits for moderate (>= 1), 3 digits for values of at least 0.001, and 2 significant digits below that.
`labels`	Optional named character vector or labeling function to replace term names with descriptive labels.
`output`	Output format: one of `"console"` (print to console), `"markdown"` (print Markdown table for R Markdown), `"latex"` (print LaTeX table for PDF compilation), or `"csv"` (write CSV file).
`file`	File path for CSV output; required if `output = "csv"`.

Details

Use it to report several models together. Each cell shows the estimate, significance stars, and standard error in parentheses. Terms appear in the order first seen; terms absent from a model show "–". Fit statistics follow the coefficients: N for all models, R-squared and AIC for OLS, AIC and pseudo-R-squared for binomial models, and tau, R1, and adjusted R1 for quantile regression. Compare AIC only across models of the same type on the same data.

Value

Invisibly returns a data frame with formatted regression results and fit statistics.

Examples

```
#  Perform OLS regression
m1 <- scf_ols(scf2022, income ~ age)

# Print regression results as a console table
scf_regtable(m1, digits = 2)
```

---

scf_se *Standard Errors of scf Results*

Description

Returns the pooled standard errors of a result.

Usage

```
## S3 method for class 'scf_mean'
SE(object, ...)

## S3 method for class 'scf_percentile'
SE(object, ...)

## S3 method for class 'scf_ratio'
SE(object, ...)

## S3 method for class 'scf_model_result'
SE(object, ...)
```

Arguments

object	A result from one of the functions above.
...	Not used.

Details

Works on results from `scf_mean()`, `scf_median()`, `scf_percentile()`, `scf_ratio()`, the models (`scf_ols()`, `scf_logit()`, `scf_glm()`, `scf_quantreg()`), and `scf_MIcombine()`. The values are the `se` column of the results, named by group, or by variable when there is no `by`. Stack-method percentiles have no standard error and return `NA`. For `scf_logit()`, standard errors are on the log-odds scale, unlike `std.error` in `results`, which is on the odds-ratio scale.

Value

A named numeric vector of standard errors.

See Also

`scf_MIcombine()`, scf_variance

Examples

```
SE(scf_mean(scf2022, ~networth, by = ~edcl))
```

`scf_subset` *Subset an* `scf_mi_survey` *Object*

Description

Keeps households that meet a condition, in every implicate.

Usage

```
scf_subset(scf, expr)
```

Arguments

`scf`	A `scf_mi_survey` object from `scf_load()`.
`expr`	A logical expression used to filter rows, evaluated separately in each implicate's variable frame (e.g., `age < 65 & own == 1`). It can also use variables from the calling environment.

Details

Use it to study one part of the population, such as renters or an age range. Estimates from a subset describe that group only: a mean from `scf_subset(scf2022, age >= 65)` is the mean among senior-headed households. To compare groups side by side, use `by` in the estimating functions instead.

The condition is applied in each implicate separately, so a household with imputed values can be in the subset in some implicates and not others. Rows where the condition is `NA` are dropped. The function stops if no rows meet the condition in an implicate. `n_households` is the weighted count, averaged across implicates.

Value

A new scf_mi_survey object (see scf_design())

See Also

scf_load(), scf_update()

Examples

```
# Filter for working-age households with positive net worth
scf_sub <- scf_subset(scf2022, age < 65 & networth > 0)
scf_mean(scf_sub, ~income)
```

---

scf_theme *Default Plot Theme for SCF Visualizations*

Description

The ggplot2 theme used by all scf_plot_*() functions.

Usage

```
scf_theme(base_size = 13, base_family = "sans", grid = TRUE, axis = TRUE, ...)

scf_activate_theme()
```

Arguments

base_size	Base font size. Defaults to 13.
base_family	Font family. Defaults to "sans".
grid	Logical. Show gridlines? Defaults to TRUE.
axis	Logical. Include axis ticks and lines? Defaults to TRUE.
...	Additional arguments passed to ggplot2::theme_minimal().

Details

Based on theme_minimal() and sized for 5.5 by 5.5 inch figures at 300 dpi. Add a ggplot2::theme() call to change settings. scf_activate_theme() sets it as the session default.

Value

scf_theme() returns a ggplot2 theme object. scf_activate_theme() returns NULL invisibly; called for its side effect of setting the session-wide ggplot2 theme.

See Also

ggplot2::theme(), scf_plot_dist()

Examples

```
library(ggplot2)
ggplot(mtcars, aes(factor(cyl))) +
  geom_bar(fill = "#0072B2") +
  scf_theme()

# Activate globally for the session, then restore the previous theme:
old <- ggplot2::theme_get()
scf_activate_theme()
ggplot2::theme_set(old)
```

---

scf_ttest *T-Test of Means using SCF Microdata*

Description

Tests a mean against a null value, or the difference between two group means.

Usage

```
scf_ttest(
  design,
  var,
  group = NULL,
  mu = 0,
  alternative = c("two.sided", "less", "greater"),
  conf.level = 0.95,
  variance = "fed"
)
```

Arguments

design	A scf_mi_survey object from scf_load().
var	A one-sided formula specifying a numeric variable (e.g., ~income).
group	Optional one-sided formula specifying a binary grouping variable (e.g., ~female).
mu	Numeric. Null hypothesis value. Default is 0.
alternative	Character. One of "two.sided" (default), "less", or "greater".
conf.level	Confidence level for the confidence interval. Default is 0.95.
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it to test a claim about a mean, such as whether mean income differs between two groups. The one-sample estimate is the mean; the two-sample estimate is the first group's mean minus the second's. A small p-value means a difference this large is unlikely under the null hypothesis; the confidence interval gives the plausible range. Means of wealth and income are dominated by the top of the distribution and may not reflect the typical household.

Means are estimated in each implicate with `survey::svymean()` and pooled (see scf_variance). The two-sample test accounts for the covariance between groups.

Value

An object of class `scf_ttest` with:

results A data frame with pooled estimate, standard error, t-statistic, degrees of freedom, p-value, significance stars, and confidence interval. For two-sample tests, the estimate is the first group minus the second.

means Group-specific means (for two-sample tests only).

fit List describing the test type, null hypothesis, confidence level, and alternative.

aux Survey year, variable name, and dollar year, used by `scf_nominal()`.

See Also

`scf_prop_test()`, `scf_mean()`, `scf_MIcombine()`

Examples

```
# Derive analysis vars
scf2022 <- scf_update(scf2022,
  female = factor(hhsex, levels = 1:2, labels = c("Male", "Female")),
  over50 = age > 50
)

# One-sample t-test
scf_ttest(scf2022, ~income, mu = 75000)

# Two-sample t-test
scf_ttest(scf2022, ~income, group = ~female)
```

scf_update *Create or Alter SCF Variables*

Description

Creates or changes variables in every implicate.

Usage

```
scf_update(.object, ...)
```

Arguments

.object	A scf_mi_survey object from scf_load().
...	Named expressions assigning new or modified variables using = syntax. Each must return one value or one value per row. Set a variable to NULL to remove it.

Details

Use it before analysis to recode values, convert types, or create ratios, differences, and indicators. Expressions are evaluated in order in each implicate's data. Set a variable to NULL to remove it. The result is a new object; assign it back to keep the changes. For variables that depend on the distribution within each implicate, such as ranks or percentile cutoffs, use scf_update_by_implicate().

Value

The input scf_mi_survey object with mi_design updated to reflect the new or modified variables. Other elements (year, n_households, dollar_year) and the mock attribute are kept. n_households is not recomputed.

See Also

scf_load(), scf_update_by_implicate(), survey::svrepdesign()

Examples

```
# Create a binary indicator for being over age 50
scf2022 <- scf_update(scf2022,
  over50 = age > 50
)

# Example: Create a log-transformed income variable
scf2022 <- scf_update(scf2022,
  log_income = log(income + 1)
)
```

scf_update_by_implicate *Modify Each Implicate Individually in SCF Data*

Description

Applies a function to each implicate's data frame.

Usage

```
scf_update_by_implicate(object, f)
```

Arguments

object	A scf_mi_survey object from scf_load().
f	A function that takes one implicate's data frame as its sole argument and returns a modified data frame with the same number of rows. The function signature should be function(df) { ... }. The returned data frame is used to rebuild the replicate-weighted survey design.

Details

Use for variables that depend on the distribution within each implicate, such as ranks, implicate-specific quantiles, or group z-scores. The function must return a data frame with the same number of rows. The result is a new object; assign it back to keep the changes.

Value

A modified scf_mi_survey object with updated implicate-level designs.

See Also

scf_update()

Examples

```
# Flag households in the top 10% of net worth,
# using each implicate's weighted 90th percentile as the threshold.
scf2022 <- scf_update_by_implicate(scf2022, function(df) {
  o <- order(df$networth)
  share <- cumsum(df$wgt[o]) / sum(df$wgt)
  threshold <- df$networth[o][which(share >= 0.90)[1]]
  df$top10nw <- df$networth >= threshold
  df
})

# Compute implicate-specific z-scores of income,
# using the weighted mean and standard deviation
```

```
scf2022 <- scf_update_by_implicate(scf2022, function(df) {
  mu <- stats::weighted.mean(df$income, df$wgt)
  sigma <- sqrt(stats::weighted.mean((df$income - mu)^2, df$wgt))
  df$z_income <- (df$income - mu) / sigma
  df
})

# Verify new variable exists
head(scf2022$mi_design[[1]]$variables$z_income)
```

`scf_variance` *Standard Errors Across SCF Implicates*

Description

Every SCF estimate has two sources of error. `scf` adds them to get the standard error, using Rubin's rules (see scf_MIcombine). This page explains the first source, sampling error, and the two ways `scf` can measure it.

Sampling error. The SCF interviews a sample of households, not all of them. A different sample would give a slightly different answer. The Fed measures this with 999 replicate weights: each one re-weights the sample as if it had been drawn again. `scf` re-computes the estimate with each set of replicate weights. The sampling variance U is the spread of those 999 answers: the sum of their squared distances from their mean, divided by 998.

Imputation error. Some answers are missing, and the Fed fills them in five times, giving five implicates. The spread of the five implicate estimates is the imputation variance B. See scf_MIcombine.

Two ways to measure sampling variance

Each implicate has its own sampling variance. The two methods differ only in which ones are used for U.

`"fed"` **(default)** U is the sampling variance from the first implicate. This is what the Fed's own SAS code (the MEANIT macro in the SCF codebook) does for means and medians. The Fed built the replicate weights for the first implicate, so this method uses them on the data they were made for.

`"rubin"` U is the average sampling variance over all five implicates. This is standard multiple-imputation practice (Rubin 1987) and what most software does, including `mitools` and Stata's `mi`. It uses more information, so it is slightly more stable.

For estimates on the whole population, the two methods give nearly the same answer. They can differ for groups whose membership depends on imputed values, such as households grouped by net worth or income. A household can fall in the group in one implicate and out of it in another, so the first implicate's group may not represent the others. The Fed's documentation warns about this case.

Regression models (`scf_ols()`, `scf_glm()`, `scf_logit()`, `scf_quantreg()`) always use `"rubin"`, as the Fed's codebook does for regressions (the MISECOMP macro).

If a group is missing from some implicates, `"fed"` uses the first implicate that has it.

Subsets and groups

Take care when `scf_subset()` or `by =` uses a variable that the Fed imputes, such as income, net worth, assets, or debts. Each implicate can then hold a different set of households. With `"fed"`, the sampling variance comes from the first implicate's set alone, which may not represent the other four. With `"rubin"`, it is averaged over all five sets, so every version of the group counts.

For example, `scf_subset(scf2022, networth > 1e6)` keeps a different set of households in each implicate. The same is true of groups made from imputed variables, such as net worth categories. Use `"rubin"` for analyses like these:

```
rich <- scf_subset(scf2022, networth > 1e6)
scf_mean(rich, ~income, variance = "rubin")

scf_mean(scf2022, ~income, by = ~nwcat, variance = "rubin")
scf_freq(scf2022, ~own, by = ~nwcat, variance = "rubin")
```

If the subset or group uses a variable that is rarely or never imputed, such as age or household sex, the households are nearly the same in every implicate and the two methods agree.

Which to use

- Use the default, `"fed"`, to match the Fed's published methods and figures.
- Use `"rubin"` when groups or subsets depend on imputed values (for example `by = ~nwcat`, or `scf_subset()` on income or net worth), or when results must match other multiple-imputation software.
- Report which method you used. For whole-population estimates the choice rarely matters.

How to set it

Set it with the `variance` argument:

```
scf_mean(scf2022, ~networth, variance = "rubin")
```

The `variance` argument is available in `scf_mean()`, `scf_median()`, `scf_percentile()`, `scf_freq()`, `scf_xtab()`, `scf_corr()`, `scf_ttest()`, `scf_prop_test()`, and `scf_ratio()`. Functions that call these, such as `scf_pctile_sum()` and the plot functions, use `"fed"`.

References

Rubin DB. Multiple Imputation for Nonresponse in Surveys. Wiley, 1987.

Board of Governors of the Federal Reserve System. Codebook for the Survey of Consumer Finances (see the sections on sampling error and the MEANIT SAS macro). https://www.federalreserve.gov/econres/scfindex.htm

scf_xtab *Cross-Tabulate Two Discrete Variables in Multiply-Imputed SCF Data*

Description

Estimates a two-way table of household shares, with standard errors.

Usage

```
scf_xtab(scf, rowvar, colvar, scale = "cell", variance = "fed")
```

Arguments

scf	A scf_mi_survey object from scf_load().
rowvar	A one-sided formula specifying the row variable (e.g., ~edcl).
colvar	A one-sided formula specifying the column variable (e.g., ~racecl).
scale	Character. Proportion basis: "cell" (default), "row", or "col".
variance	"fed" (default) or "rubin": how the sampling variance is combined across implicates. See scf_variance. Most users can leave this alone.

Details

Use it to see how two categorical variables relate, such as homeownership by education. scale sets which shares print: of all households ("cell"), of each row ("row", rows sum to 100 percent and show how each row divides across columns), or of each column ("col").

Shares are estimated in each implicate with the replicate weights and pooled (see scf_variance). Counts come from survey::svytable().

Value

A list of class "scf_xtab" with:

results One row per cell: row, col, rowvar, colvar, prop and se (share of all households), row_share, col_share, row_share_se, and col_share_se.

matrices Matrices of cell, row, and col shares, and their standard errors se, se_row, and se_col.

imps Weighted cell counts for each implicate.

aux List with rowvar and colvar, their labels, and the scale shown by print().

Examples

```
# Cross-tabulate ownership by sex
scf_xtab(scf2022, ~own, ~hhsex, scale = "row")
```

Index

AIC.scf_model_result
(scf_model_result_methods), 20

coef.scf_model_result
(scf_model_result_methods), 20
confint.scf_model_result
(scf_model_result_methods), 20

formula.scf_model_result
(scf_model_result_methods), 20

Gamma(), *11*
ggplot2::theme(), *46*
ggplot2::theme_minimal(), *45*

poisson(), *11*
predict.scf_model_result
(scf_model_result_methods), 20
pretty(), *32*

quantreg::rq(), *37–39*
quantreg::summary.rq(), *38*
quasibinomial(), *11*

residuals.scf_model_result
(scf_model_result_methods), 20

scf, 3
scf-package (scf), 3
scf_activate_theme (scf_theme), 45
scf_activate_theme(), *45*
scf_corr, 6
scf_corr(), *3*, *6*, *12*, *33*, *51*
scf_design, 7
scf_design(), *4*, *9*, *13*, *45*
scf_download, 8
scf_download(), *3*, *13*, *22*, *41*
scf_freq, 9
scf_freq(), *3*, *12*, *31*, *32*, *34*, *51*
scf_glm, 11
scf_glm(), *3*, *12*, *14*, *15*, *19*, *20*, *23*, *39*, *44*, *50*
scf_implicates, 12
scf_implicates(), *3*, *5*
scf_load, 13
scf_load(), *3*, *4*, *6–11*, *14–17*, *22–24*, *26*, *28–30*, *32–36*, *38*, *40*, *41*, *44–46*, *48*, *49*, *52*
scf_logit, 14
scf_logit(), *3*, *11*, *12*, *19*, *20*, *23*, *44*, *50*
scf_mean, 15
scf_mean(), *3*, *12*, *17–19*, *22*, *25*, *27*, *30*, *37*, *40*, *44*, *47*, *51*
scf_median, 16
scf_median(), *3*, *12*, *15*, *16*, *22*, *25*, *27*, *30*, *44*, *51*
scf_MIcombine, *6*, *11*, 17, *50*
scf_MIcombine(), *3*, *4*, *11*, *15*, *23*, *37*, *39*, *44*, *47*
scf_model_result_methods, 20
scf_nominal, 21
scf_nominal(), *3*, *4*, *13*, *41*, *47*
scf_ols, 23
scf_ols(), *3*, *6*, *7*, *11*, *12*, *15*, *19*, *20*, *39*, *44*, *50*
scf_pctile_sum, 24
scf_pctile_sum(), *3*, *51*
scf_percentile, 26
scf_percentile(), *3*, *12*, *16*, *17*, *22*, *25*, *29*, *30*, *44*, *51*
scf_plot_bbar, 27
scf_plot_bbar(), *3*, *31*
scf_plot_cbar, 29
scf_plot_cbar(), *3*
scf_plot_dbar, 30
scf_plot_dbar(), *3*, *34*
scf_plot_dist, 31
scf_plot_dist(), *3*, *10*, *16*, *36*, *46*
scf_plot_hex, 32
scf_plot_hex(), *3*, *7*
scf_plot_hist, 33

scf_plot_hist(), *3*, *36*
scf_plot_smooth, 35
scf_plot_smooth(), *3*, *33, 34*
scf_prop_test, 36
scf_prop_test(), *3*, *47*, *51*
scf_quantreg, 37
scf_quantreg(), *3*, *12*, *19, 20*, *23*, *44*, *50*
scf_ratio, 40
scf_ratio(), *3*, *12*, *18*, *44*, *51*
scf_real, 41
scf_real(), *3, 4*, *13*, *15*, *17*, *22*, *26*
scf_regtable, 42
scf_regtable(), *3*, *12*
scf_se, 43
scf_subset, 44
scf_subset(), *3*, *51*
scf_theme, 45
scf_theme(), *3, 4*, *30*, *32, 33*, *36*
scf_ttest, 46
scf_ttest(), *3*, *22*, *37*, *51*
scf_update, 48
scf_update(), *3*, *8, 9*, *13*, *41*, *45*, *49*
scf_update_by_implicate, 49
scf_update_by_implicate(), *3*, *25*, *34*, *48*
scf_variance, *3, 4*, *6*, *10*, *15–17*, *19*, *26*, *36–38*, *40*, *44*, *46, 47*, 50, *52*
scf_xtab, 52
scf_xtab(), *3*, *10*, *12*, *16*, *28*, *31*, *51*
SE(), *18*
SE.scf_mean *(*scf_se*)*, 43
SE.scf_MIresult *(*scf_MIcombine*)*, 17
SE.scf_model_result *(*scf_se*)*, 43
SE.scf_percentile *(*scf_se*)*, 43
SE.scf_ratio *(*scf_se*)*, 43
summary.scf_corr *(*scf_corr*)*, 6
survey::svrepdesign(), *4*, *7*, *13*, *48*
survey::svyquantile(), *26*
survey::svyratio(), *40*

vcov.scf_model_result *(*scf_model_result_methods*)*, 20