Package 'geoDeltaAudit'

October 19, 2026

Title Quantifying Variable Change Induced by Administrative Boundary Transformations

Version 0.1.1

Description Tools for auditing how analytic variables change when data are transformed across administrative boundary systems. The package is agnostic to data source, variable type, and administrative geography, and is designed to quantify transformation-induced change without attributing blame to any specific boundary definition or allocation scheme.

License MIT + file LICENSE

URL https://github.com/phinnphace/geoDeltaAudit

BugReports https://github.com/phinnphace/geoDeltaAudit/issues

Depends R (>= 4.1.0)

Imports dplyr, janitor, rlang, stringr, tibble,

Suggests knitr, readr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Phinn Markson [aut, cre]

Maintainer Phinn Markson <markson.2@osu.edu>

Repository CRAN

Date/Publication 2026-10-19 09:00:21 UTC

Contents

audit_transform 2
clean_geo_headers 2
prep_hud_crosswalk 3
step_zcta_to_zip_equal 3
step_zip_to_county_totratio 4

Index **5**

`audit_transform` *Audit a sequence of geographic transformations*

Description

testthat::skip("Integration test: run manually (slow / uses real data).") Computes delta_x(VAR) between a baseline and a transformed result while returning diagnostics.

Usage

```
audit_transform(
  df,
  geo_col,
  var_col,
  steps,
  baseline_filter = NULL,
  target_id = NULL
)
```

Arguments

`df`	Input data frame.
`geo_col`	Column containing geography IDs.
`var_col`	Column containing the variable of interest.
`steps`	A list of step functions created by step_* helpers.
`baseline_filter`	Optional function(df) -> filtered df defining baseline membership.
`target_id`	Optional target ID to extract after final step (e.g., "27053").

Value

An object of class `audit_result`.

`clean_geo_headers` *Normalize messy geography headers to standard names*

Description

Normalize messy geography headers to standard names

Usage

```
clean_geo_headers(df, map, keep)
```

Arguments

df	A data frame with geography columns.
map	Named character vector: names are standardized outputs, values are regex patterns of accepted input names.
keep	Character vector of standardized columns to keep.

Value

A tibble with standardized names.

---

prep_hud_crosswalk *Prepare HUD ZIP-to-County crosswalk*

Description

Standardizes HUD crosswalk fields and enforces string IDs.

Usage

```
prep_hud_crosswalk(data, ratio_col = "TOT_RATIO")
```

Arguments

data	Raw HUD crosswalk data frame.
ratio_col	Which HUD ratio to use (default: "TOT_RATIO").

Value

Tibble with columns: zip, county, tot_ratio.

---

step_zcta_to_zip_equal *Step: ZCTA -> ZIP using equal-share allocation*

Description

Given an association table mapping ZCTAs to ZIPs, allocate each ZCTA's values equally across its associated ZIPs.

Usage

```
step_zcta_to_zip_equal(assoc, zcta_col = "zcta", zip_col = "zip")
```

Arguments

assoc	A data frame containing ZCTA-ZIP associations.
zcta_col	Column name in assoc containing ZCTA IDs (ignored if clean_geo_headers matches).
zip_col	Column name in assoc containing ZIP IDs (ignored if clean_geo_headers matches).

Value

A step function suitable for audit_transform().

---

step_zip_to_county_totratio

Step: ZIP -> COUNTY using HUD TOT_RATIO

Description

Allocate ZIP-level values to counties using HUD's TOT_RATIO weights.

Usage

```
step_zip_to_county_totratio(
  hud,
  zip_col = "zip",
  county_col = "county",
  weight_col = "tot_ratio"
)
```

Arguments

hud	A data frame containing ZIP-to-county weights.
zip_col	Column name for ZIP (kept for API symmetry; cleaning is robust).
county_col	Column name for county (FIPS) (kept for API symmetry).
weight_col	Column name for the weight (default "tot_ratio") (kept for API symmetry).

Value

A step function suitable for audit_transform().

Index

audit_transform, 2

clean_geo_headers, 2

prep_hud_crosswalk, 3

step_zcta_to_zip_equal, 3
step_zip_to_county_totratio, 4