Package 'shellgame'

May 27, 2026

Title The Shell Game - Audit Geographic Data Transformations

Version 0.1.1

Description Reveals how data quality silently degrades during geographic transformations while variable labels remain unchanged. Demonstrates that transformation error is agnostic to both the variable (population, income, etc.) and the tool ('R', 'Python', etc.). Provides a reproducible audit framework for quantifying the shift from observed to imputed data at each transformation hop.

License MIT + file LICENSE

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

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

Depends R (>= 4.0.0)

Imports dplyr (>= 1.0.0), ggplot2, janitor, magrittr, rlang, stringr, tidycensus, utils

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

VignetteBuilder knitr

Encoding UTF-8

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Language en-US

NeedsCompilation no

Author Phinn Markson [aut, cre] (ORCID: <https://orcid.org/0000-0002-9169-6095>)

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

Repository CRAN

Date/Publication 2026-05-27 20:10:23 UTC

Contents

`audit_transformation` *Audit geographic transformation*

Description

Main function to audit a complete geographic transformation pipeline. Quantifies the perturbation introduced at each hop and reveals the shell game.

Usage

```
audit_transformation(
  baseline_data,
  zip_zcta_map,
  hud_crosswalk,
  county_fips,
  variable_name = "value",
  value_col = "estimate"
)
```

Arguments

`baseline_data`	Data frame with baseline data at source geography
`zip_zcta_map`	ZIP-ZCTA association crosswalk
`hud_crosswalk`	HUD ZIP-County crosswalk
`county_fips`	Target county FIPS code
`variable_name`	Name of the variable being tracked (for reporting)
`value_col`	Name of the value column in baseline_data

Value

An object of class "shellgame_audit" with audit results

Examples

```
baseline <- data.frame(zcta = c("00001", "00002"), estimate = c(1000, 2000))
zip_zcta <- data.frame(zcta = c("00001", "00002"), zip = c("00010", "00010"))
hud <- data.frame(zip = "00010", county = "99999", tot_ratio = 1)
result <- audit_transformation(
    baseline_data = baseline,
    zip_zcta_map = zip_zcta,
    hud_crosswalk = hud,
    county_fips = "99999",
    variable_name = "population"
)
summary(result)
```

---

check_census_key *Check for Census API key*

Description

Validates that a Census API key is available for tidycensus.

Usage

```
check_census_key(install = FALSE)
```

Arguments

install	Logical, whether to install the key for future sessions

Value

Invisible TRUE if key exists, stops with error if not

---

create_audit_report *Create complete audit report*

Description

Generates all visualizations for an audit.

Usage

```
create_audit_report(
  audit_result,
  zcta_baseline_sf = NULL,
  zcta_geometric_sf = NULL,
  county_sf = NULL
)
```

Arguments

audit_result	A shellgame_audit object
zcta_baseline_sf	Optional: SF object with baseline ZCTAs
zcta_geometric_sf	Optional: SF object with geometric ZCTAs
county_sf	Optional: SF object with county boundary

Value

List of ggplot2 objects

---

extract_perturbed_population

Extract perturbation by receiving county

Description

Returns a data frame of counties that received population redistributed from the target county during the transformation, ordered by magnitude.

Usage

```
extract_perturbed_population(audit_result, top_n = 10)
```

Arguments

audit_result	A shellgame_audit object
top_n	Number of top counties to return (default: 10)

Value

Data frame with columns: county, value

get_zcta_baseline *Get ACS baseline data for ZCTAs*

Description

Fetches ACS 5-year estimates for a specified variable at the ZCTA level using the Census API via the tidycensus package. Requires a Census API key (see https://api.census.gov/data/key_signup.html) and the tidycensus package to be installed.

Usage

```
get_zcta_baseline(variable, year = 2022, zctas = NULL)
```

Arguments

variable	ACS variable code (e.g., "B01001_001" for total population)
year	ACS year (default: 2022)
zctas	Optional character vector of ZCTAs to filter to

Value

Data frame with columns: zcta, estimate, moe

Examples

```
## Not run:
# get_zcta_baseline() retrieves ACS data via the Census API.
# See vignette("data-preparation", package = "geoDeltaAudit") for a full walkthrough.
pop_data <- get_zcta_baseline("B01001_001", year = 2022)

## End(Not run)
```

pad_geoid *Pad GEOID to 5 digits*

Description

Ensures geographic identifiers are zero-padded to 5 digits.

Usage

```
pad_geoid(geoid)
```

Arguments

geoid	Character or numeric vector of geographic identifiers

Value

Character vector of 5-digit zero-padded GEOIDs

Examples

```
pad_geoid(c("123", "45678", 789))
#> [1] "00123" "45678" "00789"
```

`plot_baseline_zctas` *Plot baseline ZCTAs*

Description

Creates a map showing the baseline ZCTAs used in the analysis.

Usage

```
plot_baseline_zctas(zcta_sf, county_sf, title = "Baseline ZCTAs")
```

Arguments

`zcta_sf`	SF object with ZCTA geometries
`county_sf`	SF object with county boundary
`title`	Plot title

Value

A ggplot2 object

`plot_geometric_vs_relationship`
Plot geometric vs relationship membership

Description

Visualizes the discrepancy between geometric intersection and relationship-based membership.

Usage

```
plot_geometric_vs_relationship(
  zcta_baseline_sf,
  zcta_geometric_sf,
  county_sf,
  title = "Geometric vs Relationship Membership"
)
```

Arguments

`zcta_baseline_sf`	SF object with baseline ZCTAs (relationship-based)
`zcta_geometric_sf`	SF object with all geometrically intersecting ZCTAs
`county_sf`	SF object with county boundary
`title`	Plot title

Value

A ggplot2 object

`plot_transformation_perturbation` *Plot transformation perturbation*

Description

Creates a simple bar chart showing baseline vs recovered values.

Usage

```
plot_transformation_perturbation(audit_result)
```

Arguments

`audit_result`	A shellgame_audit object

Value

A ggplot2 object

`prep_hud_crosswalk` *Prepare HUD ZIP-County crosswalk data*

Description

Standardizes HUD crosswalk data with proper column names and formatting.

Usage

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

Arguments

data	Raw HUD crosswalk data frame
ratio_col	Name of the ratio column to use (default: "TOT_RATIO")

Value

Data frame with standardized columns: zip, county, tot_ratio

Examples

```
raw <- data.frame(ZIP = "00010", COUNTY = "99999", TOT_RATIO = 1)
result <- prep_hud_crosswalk(raw)
```

---

prep_zip_zcta *Prepare ZIP-ZCTA crosswalk data*

Description

Standardizes ZIP-ZCTA crosswalk data with proper column names and formatting.

Usage

```
prep_zip_zcta(data, zip_col = NULL, zcta_col = "zcta")
```

Arguments

data	Raw ZIP-ZCTA crosswalk data frame
zip_col	Name of the ZIP code column (default: "ZIP_CODE" or "zip")
zcta_col	Name of the ZCTA column (default: "zcta")

Value

Data frame with standardized columns: zcta, zip

Examples

```
raw <- data.frame(ZIP_CODE = c("00010", "00010"), zcta = c("00001", "00002"))
result <- prep_zip_zcta(raw)
```

print.shellgame_audit *Print method for shellgame_audit*

Description

Print method for shellgame_audit

Usage

```
## S3 method for class 'shellgame_audit'
print(x, ...)
```

Arguments

x	A shellgame_audit object
...	Additional arguments (ignored)

Value

Invisibly returns the input object. Called for side effects (console output).

run_full_transformation *Run full transformation pipeline*

Description

Executes both hops: ZCTA → ZIP → County. Tracks the complete swap from observed to imputed data.

Usage

```
run_full_transformation(
  baseline_data,
  zip_zcta_map,
  hud_crosswalk,
  value_col = "estimate",
  county_fips = NULL
)
```

Arguments

baseline_data	Data frame with ZCTA-level baseline data
zip_zcta_map	ZIP-ZCTA association table
hud_crosswalk	HUD ZIP-County crosswalk
value_col	Name of value column in baseline_data
county_fips	Optional county FIPS to filter final result

Value

List with intermediate and final results

Examples

```
baseline <- data.frame(zcta = c("00001", "00002"), estimate = c(1000, 2000))
zip_zcta <- data.frame(zcta = c("00001", "00002"), zip = c("00010", "00010"))
hud <- data.frame(zip = "00010", county = "99999", tot_ratio = 1)
result <- run_full_transformation(baseline, zip_zcta, hud,
    value_col = "estimate", county_fips = "99999")
```

---

summary.shellgame_audit *Summary method for shellgame_audit*

Description

Summary method for shellgame_audit

Usage

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

Arguments

object	A shellgame_audit object
...	Additional arguments (ignored)

Value

Invisibly returns the input object. Called for side effects (console output).

---

transform_zcta_to_zip *Transform ZCTA data to ZIP level*

Description

Performs the first hop: ZCTA → ZIP using association-based allocation. This is where the first swap occurs: observed data → imputed data.

Usage

```
transform_zcta_to_zip(baseline_data, zip_zcta_map, value_col = "estimate")
```

Arguments

baseline_data	Data frame with columns: zcta, and a value column
zip_zcta_map	Data frame with columns: zcta, zip
value_col	Name of the value column in baseline_data (default: "estimate")

Value

Data frame with columns: zip, value (allocated to ZIP level)

Examples

```
baseline <- data.frame(zcta = c("00001", "00002"), estimate = c(1000, 2000))
zip_zcta <- data.frame(zcta = c("00001", "00002"), zip = c("00010", "00010"))
result <- transform_zcta_to_zip(baseline, zip_zcta, value_col = "estimate")
```

---

transform_zip_to_county

Transform ZIP data to County level

Description

Performs the second hop: ZIP → County using HUD TOT_RATIO allocation. This is where the second swap occurs: further imputation via proxy.

Usage

```
transform_zip_to_county(zip_data, hud_crosswalk, county_fips = NULL)
```

Arguments

zip_data	Data frame with columns: zip, value
hud_crosswalk	Data frame with columns: zip, county, tot_ratio
county_fips	Optional FIPS code to filter to specific county

Value

Data frame with columns: county, value (allocated to county level)

Examples

```
zip_data <- data.frame(zip = "00010", value = 3000)
hud <- data.frame(zip = "00010", county = "99999", tot_ratio = 1)
result <- transform_zip_to_county(zip_data, hud, county_fips = "99999")
```

Index