

# Package ‘Household.Transmission.Chain.Data.Analysis’

October 23, 2025

**Title** Household Transmission Chain Simulation and Estimation

**Version** 0.0.0.9000

**Description** This package provides a framework for household transmission chain analysis by a streamlined pipeline to simulate household transmission data and to estimate community and within-household infection risks from either synthetic or user-supplied data. The workflow builds person-day tables, impute infection timelines with Gamma-distributed delays, and fits penalized models with optional covariates. Post-processing summarizes mean estimates, uncertainty, bias, and relative bias, Two user functions cover the main tasks: GenSyn() to simulate data and run the full estimation pipeline (with optional summaries), and TransmissionChainAnalysis() to run the same estimation pipeline on user data.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Imports** data.table,  
stats,  
utils

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

**VignetteBuilder** knitr

**Config/testthat/edition** 3

## Contents

build_person_day_table . . . . .	2
dataframe_to_household_list . . . . .	3
data_summarization . . . . .	4
generate_synthetic_data_one . . . . .	6
generate_synthetic_data_standardized . . . . .	8
GenSyn . . . . .	10
infectious_time_imputation . . . . .	14
main_parameter_estimation_pipeline . . . . .	15
postprocessing_estimates . . . . .	18
print.GenSynResult . . . . .	19

print.TransChainResult . . . . .	20
running_parameter_estimation . . . . .	21
simulate_households . . . . .	22
summarize_individuals . . . . .	24
TransmissionChainAnalysis . . . . .	25
<b>Index</b>	<b>28</b>

---

build\_person\_day\_table

*Construct long format person day data*

---

## Description

Builds a person day table for likelihood based parameter estimation by expanding individual level timelines into daily observations and computing same household infectious counts by infector role.

## Usage

```
build_person_day_table(dt, tmax, cases_t, covariate_cols = character(0))
```

## Arguments

**dt** data.table. Individual level data with infection or imputation results from [infectious\\_time\\_imputation](#)

**tmax** Integer. Maximum day index

**cases\_t** Numeric vector of length tmax + 1. Community intensity on days 0 to tmax.

**covariate\_cols** Character vector. Names of row level covariates to copy.

## Value

A long format data.table with columns agegrp2, agegrp3, agegrp4, n\_inf, n\_inf\_infant, n\_inf\_sibling, n\_inf\_adult, n\_inf\_elder, cases, event, ID\_indiv, ID\_hh, day, and the requested covariates.

## Examples

```
## Not run:
long_dt <- build_person_day_table(imputed, tmax = 200, cases_t = rep(0.001, 201))
head(long_dt)

## End(Not run)
```

---

`dataframe_to_household_list`*Convert a user data frame to a list of household tables*

---

## Description

Splits a long format data frame of test observations into a list of household level data tables. The function preserves required columns and allows any extra user provided covariates.

## Usage

```
dataframe_to_household_list(  
  df,  
  hh_col = "HH",  
  id_col = "individual_ID",  
  role_col = "role",  
  date_col = "test_date",  
  inf_col = "infection_status",  
  comm_col = "community_risk",  
  keep_extra_cols = TRUE  
)
```

## Arguments

<code>df</code>	Data frame. Must contain at least columns for household ID, individual ID, role, test date, infection status, and community risk.
<code>hh_col, id_col, role_col, date_col, inf_col, comm_col</code>	Character. Column names for household ID (default "HH"), individual ID ("individual_ID"), role ("role"), test date ("test_date"), infection status ("infection_status"), and community risk ("community_risk").
<code>keep_extra_cols</code>	Logical. If TRUE (default), retain additional covariate columns.

## Value

A list of data tables, one per household. Columns are ordered with household ID first.

## Examples

```
## Not run:  
hh_list <- dataframe_to_household_list(df)  
length(hh_list); names(hh_list)[1]; head(hh_list[[1]])  
  
## End(Not run)
```

---

data_summarization	<i>Summarize infection episodes (per individual)</i>
--------------------	--

---

## Description

Aggregates long-format testing records into one row per individual with episode counts, first/last positive within the first episode, inferred infectious-day span, last pre-positive negative, and optional covariate summaries. Accepts either a single data frame `df` containing the six required core columns, or the six vectors individually. Optionally merges an external covariate data frame before summarizing.

## Usage

```
data_summarization(
  df = NULL,
  Household_ID = NULL,
  Individual_ID = NULL,
  Household_role = NULL,
  Sample_test_days = NULL,
  Infectious_status = NULL,
  Community_rate_infection = NULL,
  Covariate_DataFrame = NULL,
  covariate_cols = NULL
)
```

## Arguments

<code>df</code>	Data frame with at least the columns <code>HH</code> , <code>individual_ID</code> , <code>role</code> , <code>test_date</code> , <code>infection_status</code> , <code>community_risk</code> . When <code>df</code> is supplied, the vector arguments below are ignored.
<code>Household_ID</code>	Integer vector. Household identifiers; used only when <code>df</code> is <code>NULL</code> .
<code>Individual_ID</code>	Integer vector. Individual identifiers; used only when <code>df</code> is <code>NULL</code> .
<code>Household_role</code>	Character vector. Household role for each row (e.g., "infant", "sibling", "adult", "elder"); used only when <code>df</code> is <code>NULL</code> .
<code>Sample_test_days</code>	Integer or numeric vector. Testing day index for each record (relative day); used only when <code>df</code> is <code>NULL</code> .
<code>Infectious_status</code>	Integer or logical vector (0/1). Test-time infection indicator; used only when <code>df</code> is <code>NULL</code> .
<code>Community_rate_infection</code>	Numeric vector. Community risk/intensity at each record; used only when <code>df</code> is <code>NULL</code> .
<code>Covariate_DataFrame</code>	Data frame of additional covariates to merge prior to summarization. The join uses the best available keys among <code>c("HH", "individual_ID", "test_date")</code> . At minimum, <code>individual_ID</code> must be present (preferably with <code>HH</code> and/or <code>test_date</code> ).
<code>covariate_cols</code>	Character vector of covariate names to summarize (after merging). If <code>NULL</code> (default), all non-core columns are summarized.

## Details

- Either `df` or all six vector inputs must be provided; otherwise an error is thrown.
- When merging `Covariate_DataFrame`, name collisions with the six core columns are avoided by renaming the colliding covariate columns with a `"_cv"` suffix.
- Covariate names are normalized to `snake_case` prior to creating summary columns.
- Records are internally ordered by `HH`, `individual_ID`, `test_date`.
- Episode detection is based on runs of `infection_status == 1`; the first episode is used for `*_start`, `*_end`, `*_duration`, and `infection.infectious.day`.

## Value

A data frame with one row per individual containing:

- `HH`: household identifier.
- `individual_ID`: individual identifier (within-household).
- `n.true.infection`: number of infection episodes (runs of 1's).
- `n.detected.infection`: number of detected episodes (same as `n.true.infection` given `infection_status` encodes detection).
- `infection.detected.start`: first positive day of the first episode.
- `infection.detected.end`: last positive day of the first episode.
- `infection.true.duration`: duration (days) of the first episode based on test days (`end - start + 1`).
- `last_negative`: last negative test day prior to the first positive (if any).
- `infection.infectious.day`: comma-separated list of test-day indices during the first episode.
- `community.risk`: mean of `community_risk` across this individual's records.
- `role`: household role (as observed).

If covariates are present, additional columns are appended per covariate (after name normalization to `snake_case`):

- `<cov>_mode`: mode for categorical/logical or binary-numeric covariates.
- `<cov>_first`, `<cov>_mean`, `<cov>_last`: first observed, mean, and last observed value for numeric covariates.
- `<cov>_timevarying`: logical flag indicating whether the covariate varies over time for the individual.

## See Also

[generate\\_synthetic\\_data\\_standardized](#), [summarize\\_individuals](#), [main\\_parameter\\_estimation\\_pipeline](#)

## Examples

```
## Not run:
# Minimal example with a data frame:
df <- data.frame(
  HH = c(1,1,1,1,1,1),
  individual_ID = c(1,1,1,2,2,2),
  role = c("infant","infant","infant","adult","adult","adult"),
  test_date = c(1,2,3,1,2,3),
```

```

infection_status = c(0,1,1,0,0,1),
community_risk = c(0.01,0.02,0.02,0.01,0.01,0.02)
)
data_summarization(df)

# With external covariates to merge and summarize:
covdf <- data.frame(
  HH = c(1,1), individual_ID = c(1,2),
  vacc_status = c(1,0), bmi = c(18.5, 27.2)
)
data_summarization(df, Covariate_DataFrame = covdf)

## End(Not run)

```

---

```
generate_synthetic_data_one
```

*Generate standardized synthetic data for one household*

---

## Description

Simulates testing observations for a single household across a date range with community risk using seasonal forcing, within-household transmission, adaptive testing, baseline and partial immunity, and optional covariates.

## Usage

```

generate_synthetic_data_one(
  household_id,
  hh.size = sample(3:7, 1),
  tests.per.week = 2,
  p.comm.base.infant.fix = 0.001,
  p.comm.multiplier.sibling = 1,
  p.comm.multiplier.parent = 1,
  p.comm.multiplier.elder = 1,
  p.hh.base.infant = 0.1,
  p.hh.multiplier.sibling = 1,
  p.hh.multiplier.parent = 1,
  p.hh.multiplier.elder = 1,
  p.imm.base.sibling = 1e-10,
  p.imm.base.parent = 1e-10,
  p.imm.base.elder = 1e-10,
  partial.immunity.infant = 1e-10,
  partial.immunity.sibling = 1e-10,
  partial.immunity.parent = 1e-10,
  partial.immunity.elder = 1e-10,
  duration.latent = 2,
  duration.infect.inf = 3,
  multiplier.dur.sibpar = 0.5,
  p.detect = 0.999,
  amplitude = 0,
  phase = -0.408,

```

```

start_date = as.Date("2024-09-21"),
end_date = as.Date("2025-04-17"),
Covariates = FALSE,
Covariates_list = c("Vaccination status", "Antibody Level"),
Covariate_specifications = NULL
)

```

## Arguments

`household_id` Integer. Household identifier used to populate the HH column.

`hh.size` Integer. Household size.

`tests.per.week` Integer. Tests per person per week.

`p.comm.base.infant.fix` Numeric. Baseline community infection probability for infants per day.

`p.comm.multiplier.sibling`, `p.comm.multiplier.parent`,  
`p.comm.multiplier.elder` Numeric multipliers relative to the infant baseline for sibling, parent (adult), and elder.

`p.hh.base.infant` Numeric. Baseline within-household transmission probability per infectious infant contact per day.

`p.hh.multiplier.sibling`, `p.hh.multiplier.parent`,  
`p.hh.multiplier.elder` Multipliers relative to the infant baseline for sibling, parent (adult), and elder.

`p.imm.base.sibling`, `p.imm.base.parent`, `p.imm.base.elder` Numeric. Baseline immunity probabilities at day 1 by role.

`partial.immunity.infant`, `partial.immunity.sibling`,  
`partial.immunity.parent`, `partial.immunity.elder` Numeric. Partial-immunity modifiers by role.

`duration.latent` Integer. Mean latent period (days).

`duration.infect.inf` Integer. Mean infectious duration for infants (days).

`multiplier.dur.sibpar` Numeric. Infectious-duration multiplier for non-infant roles.

`p.detect` Numeric. Probability that an infection present on a testing day is detected.

`amplitude`, `phase` Numeric. Seasonality parameters for community risk.

`start_date`, `end_date` Date. Simulation window.

`Covariates` Logical. If TRUE, generate additional covariates.

`Covariates_list` Character vector. Names of covariates to generate.

`Covariate_specifications` List. Optional per-covariate specifications.

**Value**

A data frame with columns:

- HH household ID
- individual\_ID individual index within the household
- role one of "infant", "sibling", "adult", or "elder"
- test\_date day index where 1 maps to start\_date
- infection\_status 0/1 at the test day
- community\_risk community infection intensity on that day
- additional covariate columns when Covariates = TRUE

**Examples**

```
## Not run:
df1 <- generate_synthetic_data_one(1, hh.size = 4, tests.per.week = 2, Covariates = TRUE)
head(df1)

## End(Not run)
```

---

```
generate_synthetic_data_standardized
      (Deprecated) generate_synthetic_data_standardized
```

---

**Description**

This function is retained for backward compatibility. It forwards its arguments to [simulate\\_households](#) and emits a deprecation notice.

**Usage**

```
generate_synthetic_data_standardized(
  n_households,
  hh.size = sample(3:7, 1),
  tests.per.week,
  p.comm.base.infant.fix = 0.001,
  p.comm.multiplier.sibling = 1,
  p.comm.multiplier.parent = 1,
  p.comm.multiplier.elder = 1,
  p.hh.base.infant = 0.1,
  p.hh.multiplier.sibling = 1,
  p.hh.multiplier.parent = 1,
  p.hh.multiplier.elder = 1,
  p.imm.base.sibling = 1e-10,
  p.imm.base.parent = 1e-10,
  p.imm.base.elder = 1e-10,
  partial.immunity.infant = 1e-10,
  partial.immunity.sibling = 1e-10,
  partial.immunity.parent = 1e-10,
  partial.immunity.elder = 1e-10,
```



```

duration.latent = 2,
duration.infect.inf = 3,
multiplier.dur.sibpar = 0.5,
p.detect = 0.999,
amplitude = 0,
phase = -0.408,
start_date = as.Date("2024-09-21"),
end_date = as.Date("2025-04-17"),
Covariates = FALSE,
Covariates_list = c("Vaccination status", "Antibody Level"),
Covariate_specifications = NULL
)

```

### Arguments

n_households	Integer. Number of households to simulate.
hh.size	Integer. Household size applied to each simulated household.
tests.per.week	Integer. Tests per person per week.
p.comm.base.infant.fix	Numeric. Baseline community infection probability for infants per day.
p.comm.multiplier.sibling,	p.comm.multiplier.parent,
p.comm.multiplier.elder	
	Numeric multipliers applied to the infant baseline for sibling, parent (adult), and elder, respectively.
p.hh.base.infant	Numeric. Baseline within household transmission probability per infectious infant contact per day.
p.hh.multiplier.sibling,	p.hh.multiplier.parent,
p.hh.multiplier.elder	
	Multipliers relative to the infant baseline for sibling, parent (adult), and elder.
p.imm.base.sibling, p.imm.base.parent, p.imm.base.elder	
	Numeric. Baseline immunity probabilities at day 1 by role.
partial.immunity.infant,	partial.immunity.sibling,
partial.immunity.parent, partial.immunity.elder	
	Numeric. Partial immunity by role.
duration.latent	Integer. Mean latent period in days.
duration.infect.inf	Integer. Mean infectious duration for infants in days.
multiplier.dur.sibpar	Numeric. Infectious duration multiplier for non infant roles.
p.detect	Numeric. Probability that an infection present on a testing day is detected,
amplitude, phase	Numeric. Seasonality parameters for community risk.
start_date, end_date	Date. Simulation window.
Covariates	Logical. If TRUE, generate additional covariates.
Covariates_list	Character vector. Names of covariates to generate.
Covariate_specifications	List. Optional per covariate specifications.

**Details**

Deprecated alias for [simulate\\_households](#).

**Value**

See [simulate\\_households](#).

**See Also**

[simulate\\_households](#)

**Examples**

```
## Not run:
sims <- generate_synthetic_data_standardized(
  n_households = 5,
  simulation_function = generate_synthetic_data_one,
  hh.size = 4,
  tests.per.week = 2
)
length(sims); head(sims[[1]])

## End(Not run)
```

---

GenSyn

*Simulate households, estimate transmission, and (optionally) summarize*

---

**Description**

Generates synthetic household transmission data and runs the full parameter estimation pipeline. By default, it prints a post-processing table comparing the mean estimates to “true” values (bias and relative bias). Optionally, it prints a per-individual data summary produced by the pipeline.

**Usage**

```
GenSyn(
  synthetic_data = TRUE,
  data_summary = FALSE,
  postprocessing = TRUE,
  n_households = 10,
  n_runs = 10,
  hh.size = sample(3:7, 1),
  tests.per.week = 1,
  Covariates = FALSE,
  Covariates_list = c("Vaccination status", "Antibody Level"),
  Covariate_specifications = NULL,
  day_series_covariates = TRUE,
  series_cols = NULL,
  comm_covariate_cols = NULL,
  hh_covariate_cols = NULL,
  hh_by_role = FALSE,
```

```

hh_role_covariate_cols = NULL,
standardize_covariates = TRUE,
lambda_comm = 0.01,
lambda_hh = 0.01,
p.comm.base.infant.fix = 0.002,
p.comm.multiplier.sibling = 1,
p.comm.multiplier.parent = 1,
p.comm.multiplier.elder = 1,
p.hh.base.infant = 0.2,
p.hh.multiplier.sibling = 0.5267686,
p.hh.multiplier.parent = 0.8008933,
p.hh.multiplier.elder = 0.6008933,
p.imm.base.sibling = 1e-10,
p.imm.base.parent = 1e-10,
p.imm.base.elder = 1e-10,
partial.immunity.infant = 1e-10,
partial.immunity.sibling = 1e-10,
partial.immunity.parent = 1e-10,
partial.immunity.elder = 1e-10,
duration.latent = 1,
duration.infect.inf = 2,
multiplier.dur.sibpar = 0.5,
p.detect = 0.999,
amplitude = 2.6581 * 0,
phase = -0.408,
start_date = as.Date("2024-09-21"),
end_date = as.Date("2025-04-17"),
latent_par = list(shape = 2, scale = 1),
report_par = list(shape = 1, scale = 1.5),
infect_par = list(shape = 3, scale = 2),
start_par = c(-6, 0.02, -2, rep(0, 6)),
lambda = 0.01,
lambda0 = 0.2,
lambda_alpha = 5,
delta0_true = qlogis(0.002),
alpha0_true = qlogis(0.2),
true_values = c(delta0 = log(7.148217e-05), gamma2 = log(7.148217e-05 * 4.331956) -
  log(7.148217e-05), gamma3 = log(7.148217e-05 * 1.835466) - log(7.148217e-05), gamma4
  = log(7.148217e-05 * 2) - log(7.148217e-05), alpha0 = log(0.2888953), beta2 =
  log(0.2888953 * 0.5267686) - log(0.2888953), beta3 = log(0.2888953 * 0.8008933) -
  log(0.2888953), beta4 = log(0.2888953 * 0.6008933) - log(0.2888953))
)

```

## Arguments

**synthetic\_data** Logical. Must be TRUE. If FALSE, an error is thrown with guidance to use [TransmissionChainAnalysis](#).

**data\_summary** Logical. If TRUE, prints `results$summarized_data`. Default FALSE.

**postprocessing** Logical. If TRUE, prints [postprocessing\\_estimates](#). Default TRUE.

**n\_households** Integer. Number of households to simulate. Default 10.

**n\_runs** Integer. Repeated estimation runs. Default 10.

hh.size	Integer. Household size (constant across HHs unless a random draw is passed). Default <code>sample(3:7, 1)</code> .
tests.per.week	Integer. Tests per person per week (1–3). Default 1.
Covariates	Logical. Generate synthetic covariates. Default FALSE.
Covariates_list	Character vector of covariate names to generate when <code>Covariates=TRUE</code> . Default <code>c("Vaccination status", "Antibody Level")</code> .
Covariate_specifications	List with per-covariate generation specs (type, dist, time_varying, params).
day_series_covariates	Logical. If TRUE, builds day-series list-columns for covariates in the person–day table. Default TRUE.
series_cols	Character vector of covariate names (pre-normalization) to build as day-series; if NULL, all detected covariates are used.
comm_covariate_cols	Character vector of community-level covariates (no intercept) for the likelihood.
hh_covariate_cols	Character vector of household covariates shared across roles.
hh_by_role	Logical. If TRUE, fit role-specific household covariates.
hh_role_covariate_cols	Named list with elements <code>infant</code> , <code>sibling</code> , <code>adult</code> , <code>elder</code> providing covariate names per role when <code>hh_by_role=TRUE</code> .
standardize_covariates	Logical. Z-score non-binary numeric columns in model matrices. Default TRUE.
lambda_comm, lambda_hh	Numeric. L2 penalties for community and household covariate coefficients.
p.comm.base.infant.fix, p.comm.multiplier.sibling, p.comm.multiplier.parent, p.comm.multiplier.elder	Community infection baseline and role multipliers used in simulation.
p.hh.base.infant, p.hh.multiplier.sibling, p.hh.multiplier.parent, p.hh.multiplier.elder	Household transmission baseline and role multipliers used in simulation.
p.imm.base.sibling, p.imm.base.parent, p.imm.base.elder	Baseline immunity probabilities.
partial.immunity.infant, partial.immunity.sibling, partial.immunity.parent, partial.immunity.elder	Breakthrough modifiers for partially immune individuals.
duration.latent, duration.infect.inf, multiplier.dur.sibpar	Natural-history durations used in simulation.
p.detect	Detection probability in testing algorithm.
amplitude, phase	Seasonal modulation of community risk.
start_date, end_date	Simulation window as Date.
latent_par, report_par, infect_par	Lists with Gamma(shape, scale) parameters for imputation of latent delay, reporting delay, and infectious period.
start_par	Numeric vector of initial parameters (auto-expanded if length mismatches model).

<code>lambda</code>	Base L2 penalty for slope-like parameters (age multipliers and household role offsets).
<code>lambda0</code>	Penalty anchoring <code>delta0</code> near <code>delta0_true</code> .
<code>lambda_alpha</code>	Penalty anchoring <code>alpha0</code> near <code>alpha0_true</code> .
<code>delta0_true, alpha0_true</code>	Numeric anchors (logit scale) for community baseline and household baseline.
<code>true_values</code>	Optional named numeric vector of reference values. Names should match columns in <code>theta_mat</code> . Unmatched names are ignored; missing references yield NA bias/relative bias.

## Details

This function is a thin wrapper around [main\\_parameter\\_estimation\\_pipeline](#) with `synthetic_data=TRUE`. Any extra columns generated (e.g., covariates) are automatically carried into the summarization and may be used in the likelihood via the covariate mapping arguments.

## Value

(Invisibly) a list with elements:

- `results`: Output of [main\\_parameter\\_estimation\\_pipeline](#) (raw simulations, summaries, person-day, estimates).
- `postprocessing`: The table returned by [postprocessing\\_estimates](#) (or NULL if `postprocessing=FALSE`).

## Workflow

1. Simulate households and test records.
2. Summarize individuals and impute infection timelines.
3. Build a person-day table and run repeated ML estimation.
4. (Optional) Print data summary and post-processing table.

## See Also

[TransmissionChainAnalysis](#), [main\\_parameter\\_estimation\\_pipeline](#), [postprocessing\\_estimates](#)

## Examples

```
## Not run:
out <- GenSyn(n_households = 50, n_runs = 20, Covariates = TRUE)
names(out)
head(out$results$person_day)

## End(Not run)
```

---

infectious\_time\_imputation

*Impute infection timelines from delay distributions*


---

## Description

Imputes infection date, infectious start and end, and component delays using gamma distributions, optionally scaled by user functions of covariates.

## Usage

```
infectious_time_imputation(
  dt,
  study_start,
  latent_par,
  report_par,
  infect_par,
  latent_scale_fn = NULL,
  report_scale_fn = NULL,
  infect_scale_fn = NULL
)
```

## Arguments

`dt` data.table. Individual level summary from [summarize\\_individuals](#).

`study_start` Date. Reference origin for relative day indices.

`latent_par, report_par, infect_par` Lists with elements shape and scale for the latent, reporting, and infectious period gamma distributions.

`latent_scale_fn, report_scale_fn, infect_scale_fn` Optional functions that take `dt[idx]` for infected rows and return numeric multipliers for the corresponding scales.

## Value

The input `dt` with added or imputed columns: `latent_delay`, `report_delay`, `infect_period`, `inf_date`, `inf_start_date`, `inf_end_date`, and relative day versions `inf_day_rl`, `infectious_day_rl`, `infectious_end_day_rl`.

## Examples

```
## Not run:
imputed <- infectious_time_imputation(
  dt = ind,
  study_start = as.Date("2024-09-21"),
  latent_par = list(shape = 2, scale = 1),
  report_par = list(shape = 1, scale = 1.5),
  infect_par = list(shape = 3, scale = 2)
)

## End(Not run)
```

---

main\_parameter\_estimation\_pipeline

*Main parameter estimation pipeline*


---

## Description

End-to-end pipeline that (i) obtains household testing data (either simulated or user-supplied), (ii) summarizes individuals, (iii) imputes infection timelines, (iv) expands to a person-day table, and (v) runs repeated maximum-likelihood estimation with penalties.

## Usage

```
main_parameter_estimation_pipeline(
  user_data = NULL,
  synthetic_data = TRUE,
  n_households = 10,
  n_runs = 10,
  hh.size = sample(3:7, 1),
  tests.per.week = 1,
  Covariates = FALSE,
  Covariates_list = c("Vaccination status", "Antibody Level"),
  Covariate_specifications = NULL,
  day_series_covariates = TRUE,
  series_cols = NULL,
  comm_covariate_cols = NULL,
  hh_covariate_cols = NULL,
  hh_by_role = FALSE,
  hh_role_covariate_cols = NULL,
  standardize_covariates = TRUE,
  lambda_comm = 0.01,
  lambda_hh = 0.01,
  p.comm.base.infant.fix = 0.002,
  p.comm.multiplier.sibling = 1,
  p.comm.multiplier.parent = 1,
  p.comm.multiplier.elder = 1,
  p.hh.base.infant = 0.2,
  p.hh.multiplier.sibling = 0.5267686,
  p.hh.multiplier.parent = 0.8008933,
  p.hh.multiplier.elder = 0.6008933,
  p.imm.base.sibling = 1e-10,
  p.imm.base.parent = 1e-10,
  p.imm.base.elder = 1e-10,
  partial.immunity.infant = 1e-10,
  partial.immunity.sibling = 1e-10,
  partial.immunity.parent = 1e-10,
  partial.immunity.elder = 1e-10,
  duration.latent = 1,
  duration.infect.inf = 2,
  multiplier.dur.sibpar = 0.5,
  p.detect = 0.999,
  amplitude = 2.6581 * 0,
```

```

phase = -0.408,
start_date = as.Date("2024-09-21"),
end_date = as.Date("2025-04-17"),
latent_par = list(shape = 2, scale = 1),
report_par = list(shape = 1, scale = 1.5),
infect_par = list(shape = 3, scale = 2),
start_par = c(-6, 0.02, -2, rep(0, 6)),
lambda = 0.01,
lambda0 = 0.2,
lambda_alpha = 5,
delta0_true = qlogis(0.002),
alpha0_true = qlogis(0.2)
)

```

### Arguments

user_data	List or data.frame. User-provided observations. If a list, it should contain one data frame per household; if a single data.frame, it must contain at least columns HH, individual_ID, role, test_date, infection_status, and community_risk. Ignored when synthetic_data = TRUE. Default NULL.
synthetic_data	Logical. If TRUE (default), simulate data using <a href="#">generate_synthetic_data_standardized</a> via <a href="#">simulate_households</a> ; otherwise, convert user_data with <a href="#">dataframe_to_household_list</a> .
n_households	Integer. Number of households to simulate (used only when synthetic_data = TRUE). Default 10.
n_runs	Integer. Number of repeated estimation runs. Default 10.
hh.size	Integer. Household size for simulation. Default <code>sample(3:7, 1)</code> .
tests.per.week	Integer. Tests per person per week (1–3). Default 1.
Covariates	Logical. If TRUE, generate additional covariates in simulation.
Covariates_list	Character vector. Names of covariates to generate.
Covariate_specifications	List. Per-covariate specifications; see <a href="#">generate_synthetic_data_one</a> for schema and defaults.
day_series_covariates	Logical. If TRUE, build day-series list-columns for covariates during summarization (see <a href="#">summarize_individuals</a> ).
series_cols	Character vector or NULL. Names (pre-normalization) of covariates to include as day-series; if NULL, uses all detected covariates.
comm_covariate_cols	Character vector. Column names included in the community risk model (no intercept).
hh_covariate_cols	Character vector. Household covariates shared across roles.
hh_by_role	Logical. If TRUE, allow role-specific household covariates.
hh_role_covariate_cols	List. Named list with elements infant, sibling, adult, elder, each a character vector of columns; falls back to hh_covariate_cols when missing.
standardize_covariates	Logical. Z-score non-binary numeric columns in model matrices.



lambda_comm, lambda_hh	Numeric. L2 penalties for community and household covariate coefficients. Defaults 0.01.
p.comm.base.infant.fix	Numeric. Baseline community infection probability per day for infants.
p.comm.multiplier.sibling, p.comm.multiplier.elder	Numeric multipliers applied to p.comm.base.infant.fix by role.
p.hh.base.infant	Numeric. Baseline within-household transmission probability per infectious infant contact (per day).
p.hh.multiplier.sibling, p.hh.multiplier.elder	Multipliers relative to infant baseline by role.
p.imm.base.sibling, p.imm.base.parent, p.imm.base.elder	Numeric. Baseline (day-1) immunity probabilities by role.
partial.immunity.infant, partial.immunity.parent, partial.immunity.elder	Numeric. Breakthrough scaling (0–1) if previously immune, by role.
duration.latent	Integer. Mean latent period (days).
duration.infect.inf	Integer. Mean infectious duration for infants (days). Other roles use duration.infect.inf * multiplier.dur.sibpar.
multiplier.dur.sibpar	Numeric. Infectious duration multiplier for non-infant roles.
p.detect	Numeric. Detection probability on testing days.
amplitude, phase	Numeric. Seasonality parameters (cosine forcing) for community risk.
start_date, end_date	Date. Simulation/analysis window (inclusive of start_date).
latent_par, report_par, infect_par	Lists with elements shape and scale for the latent delay, reporting delay, and infectious period gammas.
start_par	Numeric vector. Initial parameter vector. If not tailored to the covariate layout, it is resized internally when passed to the optimizer.
lambda	Numeric. Base L2 penalty for gamma (age) and z_* (role offsets).
lambda0, lambda_alpha	Numeric. Penalties pulling delta0 and alpha0 toward delta0_true and alpha0_true.
delta0_true, alpha0_true	Numeric anchors for the intercept penalties (logit scale).

## Details

When `synthetic_data = TRUE`, data are produced by [generate\\_synthetic\\_data\\_standardized](#) via [simulate\\_households](#). When `FALSE`, user\_data is converted with [dataframe\\_to\\_household\\_list](#). Covariate names are normalized to snake\_case during summarization.

Community intensity cases\_t used by the person-day builder is generated internally as a smoothed, scaled seasonal signal over the analysis window.

**Value**

A list with elements:

- `raw_simulation`: list of household data frames (simulated or user-provided).
- `summarized_data`: individual-level summary table (from [summarize\\_individuals](#) then [infectious\\_time\\_imputation](#)).
- `person_day`: long person-day table (from [build\\_person\\_day\\_table](#)).
- `estimates`:  $n\_runs \times p$  matrix of parameter estimates (from [running\\_parameter\\_estimation](#)).

**See Also**

[simulate\\_households](#), [generate\\_synthetic\\_data\\_standardized](#), [summarize\\_individuals](#), [infectious\\_time\\_imputation](#), [build\\_person\\_day\\_table](#), [running\\_parameter\\_estimation](#), [postprocessing\\_estimates](#)

**Examples**

```
## Not run:
res <- main_parameter_estimation_pipeline(
  synthetic_data = TRUE,
  n_households = 20, n_runs = 5,
  hh.size = 4, tests.per.week = 2,
  delta0_true = qlogis(0.002), alpha0_true = qlogis(0.2)
)
str(res$estimates)

## End(Not run)
```

---

postprocessing\_estimates

*Summarize and compare estimates against reference values*

---

**Description**

Computes mean, SD, SE across runs and (optionally) bias and relative bias versus supplied reference (true) values for each parameter.

**Usage**

```
postprocessing_estimates(
  theta_mat,
  true_values = c(delta0 = log(7.148217e-05), gamma2 = log(7.148217e-05 * 4.331956) -
    log(7.148217e-05), gamma3 = log(7.148217e-05 * 1.835466) - log(7.148217e-05), gamma4
    = log(7.148217e-05 * 2) - log(7.148217e-05), alpha0 = log(0.2888953), beta2 =
    log(0.2888953 * 0.5267686) - log(0.2888953), beta3 = log(0.2888953 * 0.8008933) -
    log(0.2888953), beta4 = log(0.2888953 * 0.6008933) - log(0.2888953))
)
```

**Arguments**

theta_mat	Numeric matrix. Parameter estimates from multiple runs (e.g., output of <a href="#">running_parameter_estimation</a> ). Must be non-empty.
true_values	Optional named numeric vector of reference values. Names should match columns in theta_mat. Unmatched names are ignored; missing references yield NA bias/relative bias.

**Details**

Rows with incomplete estimates are dropped using the first column as a sentinel. If theta\_mat lacks column names, generic names "par1", "par2", ... are assigned. The number of retained runs is stored in the "n\_runs" attribute of the returned table.

**Value**

A data.table with one row per parameter and columns:

- Parameter: column name from theta\_mat
- Estimate: mean across runs
- SD: standard deviation across runs
- SE: standard error (SD / sqrt(n\\_runs))
- True: supplied reference value (if any)
- Bias: Estimate - True
- RelBias: Bias / |True| (NA when True is 0 or missing)
- Block, Role: simple parameter grouping labels for readability

**See Also**

[main\\_parameter\\_estimation\\_pipeline](#), [running\\_parameter\\_estimation](#)

**Examples**

```
## Not run:
est <- matrix(rnorm(50), nrow = 10)
colnames(est) <- c("delta0", "gamma2", "alpha0", "z_sib", "theta_comm_x")
postprocessing_estimates(est, true_values = c(delta0 = -9.5, alpha0 = -1.2))

## End(Not run)
```

---

print.GenSynResult      *Print a GenSynResult*

---

**Description**

Nicely prints sections available in a GenSynResult returned by [GenSyn](#). If present, the per-individual data summary and the post-processing table are shown; otherwise concise guidance is printed.

**Usage**

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

**Arguments**

x                      A GenSynResult object, typically the result of [GenSyn](#).  
 ...                    Passed to or from other methods (unused).

**Details**

This S3 method is invoked when a GenSynResult is printed (e.g., typing the object at the console or calling print(x)). The method does not perform any computation and does not modify global state; it only formats and prints components that were included in the returned object.

**Value**

x, returned invisibly.

**See Also**

[GenSyn](#)

---

```
print.TransChainResult
```

*Print a TransChainResult*

---

**Description**

Nicely prints sections available in a TransChainResult returned by [TransmissionChainAnalysis](#). If present, the per-individual data summary and the post-processing table are shown; otherwise concise guidance is printed.

**Usage**

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

**Arguments**

x                      A TransChainResult object, typically the result of [TransmissionChainAnalysis](#).  
 ...                    Passed to or from other methods (unused).

**Details**

This S3 method is invoked when a TransChainResult is printed (e.g., typing the object at the console or calling print(x)). The method performs no computation and does not modify global state; it only formats and prints components included in the returned object.

**Value**

x, returned invisibly.

**See Also**[TransmissionChainAnalysis](#)

---

running\_parameter\_estimation*Run penalized maximum likelihood parameter estimation*

---

**Description**

Repeats optimization n\_runs times from jittered starts to estimate community and household transmission parameters with optional covariate effects and quadratic penalties.

**Usage**

```
running_parameter_estimation(  
  long_dt,  
  n_runs,  
  start_par = NULL,  
  lambda,  
  lambda0,  
  lambda_alpha,  
  delta0_true,  
  alpha0_true,  
  comm_covariate_cols = NULL,  
  hh_covariate_cols = NULL,  
  hh_by_role = FALSE,  
  hh_role_covariate_cols = NULL,  
  standardize_covariates = FALSE,  
  lambda_comm = lambda,  
  lambda_hh = lambda,  
  verbose = TRUE  
)
```

**Arguments**

long_dt	data.table or data.frame. Person day data from <a href="#">build_person_day_table</a> .
n_runs	Integer. Number of optimization runs for multi start.
start_par	Numeric vector or NULL. Initial parameters. If NULL, a vector of the correct length is initialized with delta0_true and alpha0_true in the intercept slots and zeros elsewhere.
lambda	Numeric. Base L2 penalty on gamma age effects and z_* role offsets.
lambda0, lambda_alpha	Numeric. Penalty strengths that pull delta0 and alpha0 toward delta0_true and alpha0_true.
delta0_true, alpha0_true	Numeric anchors for the intercept penalties on the logit scale.
comm_covariate_cols	Character vector of community risk covariate names with no intercept.

**hh\_covariate\_cols**  
 Character vector of household covariates shared across roles.

**hh\_by\_role**  
 Logical. If TRUE, allow role specific household covariates through `hh_role_covariate_cols`.

**hh\_role\_covariate\_cols**  
 Named list with elements `infant`, `sibling`, `adult`, `elder` that give covariate names per role. Falls back to `hh_covariate_cols` if a role list is missing.

**standardize\_covariates**  
 Logical. Z score non binary numeric columns in model matrices.

**lambda\_comm, lambda\_hh**  
 Numeric. L2 penalties for community and household covariate coefficients.

**verbose**  
 Logical. If TRUE, print notes on dropped or unknown covariates and initialization.

### Value

A numeric matrix of dimension `n_runs` by `n_parameters` with column names that match the parameter layout, for example `delta0`, `gamma2`, `alpha0`, `z_sib`, and `theta_*`.

### Examples

```
## Not run:
est <- running_parameter_estimation(
  long_dt, n_runs = 5,
  lambda = 0.01, lambda0 = 0.2, lambda_alpha = 5,
  delta0_true = qlogis(0.002), alpha0_true = qlogis(0.2)
)
colnames(est)

## End(Not run)
```

---

simulate_households	<i>Simulate synthetic household data (list output)</i>
---------------------	--

---

### Description

Simulates independent households by repeatedly calling a single household simulation function (default: [generate\\_synthetic\\_data\\_one](#)). Returns a list of long format data frames, one per household.

### Usage

```
simulate_households(
  n_households,
  simulation_function = generate_synthetic_data_one,
  hh.size = sample(3:7, 1),
  tests.per.week,
  p.comm.base.infant.fix = 0.001,
  p.comm.multiplier.sibling = 1,
  p.comm.multiplier.parent = 1,
  p.comm.multiplier.elder = 1,
  p.hh.base.infant = 0.1,
```

```

p.hh.multiplier.sibling = 1,
p.hh.multiplier.parent = 1,
p.hh.multiplier.elder = 1,
p.imm.base.sibling = 1e-10,
p.imm.base.parent = 1e-10,
p.imm.base.elder = 1e-10,
partial.immunity.infant = 1e-10,
partial.immunity.sibling = 1e-10,
partial.immunity.parent = 1e-10,
partial.immunity.elder = 1e-10,
duration.latent = 2,
duration.infect.inf = 3,
multiplier.dur.sibpar = 0.5,
p.detect = 0.999,
amplitude = 0,
phase = -0.408,
start_date = as.Date("2024-09-21"),
end_date = as.Date("2025-04-17"),
Covariates = FALSE,
Covariates_list = c("Vaccination status", "Antibody Level"),
Covariate_specifications = NULL
)

```

## Arguments

`n_households` Integer. Number of households to simulate.

`simulation_function` Function. The single household simulation function to call.

`hh.size` Integer. Household size applied to each simulated household.

`tests.per.week` Integer. Tests per person per week.

`p.comm.base.infant.fix` Numeric. Baseline community infection probability for infants per day.

`p.comm.multiplier.sibling`, `p.comm.multiplier.parent`,  
`p.comm.multiplier.elder` Numeric multipliers applied to the infant baseline for sibling, parent (adult), and elder, respectively.

`p.hh.base.infant` Numeric. Baseline within household transmission probability per infectious infant contact per day.

`p.hh.multiplier.sibling`, `p.hh.multiplier.parent`,  
`p.hh.multiplier.elder` Multipliers relative to the infant baseline for sibling, parent (adult), and elder.

`p.imm.base.sibling`, `p.imm.base.parent`, `p.imm.base.elder` Numeric. Baseline immunity probabilities at day 1 by role.

`partial.immunity.infant`, `partial.immunity.sibling`,  
`partial.immunity.parent`, `partial.immunity.elder` Numeric. Partial immunity by role.

`duration.latent` Integer. Mean latent period in days.

`duration.infect.inf` Integer. Mean infectious duration for infants in days.

multiplier.dur.sibpar  
 Numeric. Infectious duration multiplier for non infant roles.

p.detect  
 Numeric. Probability that an infection present on a testing day is detected,

amplitude, phase  
 Numeric. Seasonality parameters for community risk.

start\_date, end\_date  
 Date. Simulation window.

Covariates  
 Logical. If TRUE, generate additional covariates.

Covariates\_list  
 Character vector. Names of covariates to generate.

Covariate\_specifications  
 List. Optional per covariate specifications.

### Value

A list of length `n_households`. Each element is a data frame with columns `HH`, `individual_ID`, `role`, `test_date`, `infection_status`, `community_risk`, and any generated covariates.

### See Also

[generate\\_synthetic\\_data\\_one](#), [generate\\_synthetic\\_data\\_standardized](#)

### Examples

```
## Not run:
sims <- simulate_households(5, hh.size = 4, tests.per.week = 2)
length(sims); head(sims[[1]])

## End(Not run)
```

---

`summarize_individuals` *Summarize individual level infection data*

---

### Description

Produces one row per individual with infection windows, index case flags, observation bounds, and covariate summaries. Optionally builds day series list columns for selected covariates.

### Usage

```
summarize_individuals(
  raw_dt,
  study_start,
  study_end,
  day_series_covariates = TRUE,
  series_cols = NULL
)
```



**Arguments**

`raw_dt` List of data frames or data tables, one per household.

`study_start, study_end` Date. Analysis window used for relative day indexing.

`day_series_covariates` Logical. If TRUE, attach day series list columns for covariates.

`series_cols` Character vector or NULL. Which covariates to build day series for. If NULL, all detected covariates are considered.

**Value**

A `data.table` with one row per individual including infection detection start and end, inferred infectious window using relative day indices, index case flag, observation window, role and age category, and per person covariate summaries.

**Examples**

```
## Not run:
ind <- summarize_individuals(sims, as.Date("2024-09-21"), as.Date("2025-04-17"))
head(ind)

## End(Not run)
```

---

TransmissionChainAnalysis

*Estimate transmission parameters from user data*


---

**Description**

Runs the full parameter-estimation pipeline on user-provided long-format testing data. By default, prints a post-processing table (mean estimates, SD/SE, bias, relative bias). Optionally prints the per-individual summary produced by the pipeline.

**Usage**

```
TransmissionChainAnalysis(
  user_data,
  data_summary = FALSE,
  postprocessing = TRUE,
  n_runs = 10,
  day_series_covariates = TRUE,
  series_cols = NULL,
  comm_covariate_cols = NULL,
  hh_covariate_cols = NULL,
  hh_by_role = FALSE,
  hh_role_covariate_cols = NULL,
  standardize_covariates = TRUE,
  lambda_comm = 0.01,
  lambda_hh = 0.01,
  start_date = as.Date("2024-09-21"),
```

```

end_date = as.Date("2025-04-17"),
latent_par = list(shape = 2, scale = 1),
report_par = list(shape = 1, scale = 1.5),
infect_par = list(shape = 3, scale = 2),
start_par = c(-6, 0.02, -2, rep(0, 6)),
lambda = 0.01,
lambda0 = 0.2,
lambda_alpha = 5,
delta0_true = qlogis(0.002),
alpha0_true = qlogis(0.2),
true_values = NA
)

```

### Arguments

<code>user_data</code>	Data frame (preferred) with the required columns listed above; alternatively, a list of per-household data frames which will be row-bound. Must not be NULL.
<code>data_summary</code>	Logical. If TRUE, prints <code>results\$summarized_data</code> . Default FALSE.
<code>postprocessing</code>	Logical. If TRUE (default), prints <a href="#">postprocessing_estimates</a> .
<code>n_runs</code>	Integer. Number of repeated estimation runs. Default 10.
<code>day_series_covariates</code>	Logical. If TRUE, builds day-series list-columns for covariates in the person-day table. Default TRUE.
<code>series_cols</code>	Character vector of covariate names (pre-normalization) for which to build day-series; if NULL, all detected covariates are used.
<code>comm_covariate_cols</code>	Character vector of community-model covariates (no intercept).
<code>hh_covariate_cols</code>	Character vector of household-model covariates shared across roles.
<code>hh_by_role</code>	Logical. If TRUE, use role-specific household covariates.
<code>hh_role_covariate_cols</code>	Named list with elements <code>infant</code> , <code>sibling</code> , <code>adult</code> , <code>elder</code> providing covariate names per role when <code>hh_by_role=TRUE</code> .
<code>standardize_covariates</code>	Logical. Z-score non-binary numeric columns in model matrices. Default TRUE.
<code>lambda_comm, lambda_hh</code>	Numeric. L2 penalties for community and household covariate coefficients.
<code>start_date, end_date</code>	Date. Study window used for summarization and imputation if <code>test_date</code> is provided as Date.
<code>latent_par, report_par, infect_par</code>	Lists with Gamma(shape, scale) parameters for imputation of latent delay, reporting delay, and infectious period.
<code>start_par</code>	Numeric vector of initial parameters (auto-expanded if length mismatches model).
<code>lambda</code>	Base L2 penalty for slope-like parameters (age multipliers and household role offsets).
<code>lambda0</code>	Penalty anchoring <code>delta0</code> near <code>delta0_true</code> .
<code>lambda_alpha</code>	Penalty anchoring <code>alpha0</code> near <code>alpha0_true</code> .

`delta0_true, alpha0_true` Numeric anchors (logit scale) for community baseline and household baseline.

`true_values` Optional named numeric vector of reference values. Names should match columns in `theta_mat`. Unmatched names are ignored; missing references yield NA bias/relative bias.

## Details

This wrapper calls [main\\_parameter\\_estimation\\_pipeline](#) with `synthetic_data=FALSE` and passes `user_data` through [dataframe\\_to\\_household\\_list](#) internally. If `user_data` is a list of per-household data frames, it is row-bound before processing. Character covariates are internally factorized via model matrices when used in the likelihood.

## Value

(Invisibly) a list with elements:

- `results`: Output of [main\\_parameter\\_estimation\\_pipeline](#) (summaries, person-day, estimates).
- `postprocessing`: The table returned by [postprocessing\\_estimates](#) (or NULL if `postprocessing=FALSE`).

## Required columns in `user_data`

- `HH` — household identifier (integer)
- `individual_ID` — individual identifier within household (integer)
- `role` — one of "infant", "sibling", "adult", "elder"
- `test_date` — test day (integer or Date; Date will be converted to relative day)
- `infection_status` — 0/1 infectious status at `test_date`
- `community_risk` — numeric community infection intensity at `test_date`

Any additional columns are treated as candidate covariates and may be included in the likelihood via `comm_covariate_cols`, `hh_covariate_cols`, or `hh_role_covariate_cols`.

## See Also

[GenSyn](#), [main\\_parameter\\_estimation\\_pipeline](#), [data\\_summarization](#), [postprocessing\\_estimates](#)

## Examples

```
## Not run:
# Suppose df is your long-format dataset with required columns:
fit <- TransmissionChainAnalysis(
  user_data = df,
  n_runs = 20,
  comm_covariate_cols = c("cases"),
  hh_covariate_cols   = c("vaccination_status_mode")
)
fit$postprocessing

## End(Not run)
```

# Index

`build_person_day_table`, [2](#), [18](#), [21](#)

`data_summarization`, [4](#), [27](#)

`dataframe_to_household_list`, [3](#), [16](#), [17](#),  
[27](#)

`generate_synthetic_data_one`, [6](#), [16](#), [22](#),  
[24](#)

`generate_synthetic_data_standardized`,  
[5](#), [8](#), [16–18](#), [24](#)

`GenSyn`, [10](#), [19](#), [20](#), [27](#)

`infectious_time_imputation`, [2](#), [14](#), [18](#)

`main_parameter_estimation_pipeline`, [5](#),  
[13](#), [15](#), [19](#), [27](#)

`postprocessing_estimates`, [11](#), [13](#), [18](#), [18](#),  
[26](#), [27](#)

`print.GenSynResult`, [19](#)

`print.TransChainResult`, [20](#)

`running_parameter_estimation`, [18](#), [19](#), [21](#)

`simulate_households`, [8](#), [10](#), [16–18](#), [22](#)

`summarize_individuals`, [5](#), [14](#), [16](#), [18](#), [24](#)

`TransmissionChainAnalysis`, [11](#), [13](#), [20](#), [21](#),  
[25](#)