

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

January 30, 2026

**Title** Household Transmission Chain Simulation and Estimation

**Version** 0.0.0.9000

**Description** Tools to simulate and analyze household transmission chains with viral-load dynamics. Generates synthetic household data with role-specific susceptibility and infectivity and optional seasonal forcing, constructs person-day and test-time tables, and estimates community and within-household transmission parameters via Bayesian inference using Stan. Includes utilities to preprocess user-supplied testing data, summarize secondary attack rates and transmission timelines, and creat diagnostic plots.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Imports** data.table,  
dplyr,  
ggplot2,  
tibble,  
tidyr,  
utils,  
rstan,  
stats

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

**VignetteBuilder** knitr

**Config/testthat/edition** 3

## Contents

build_person_day_table . . . . .	2
build_stan_household_arrays . . . . .	3
dataframe_to_household_list . . . . .	4
data_summarization . . . . .	4
generate_synthetic_data_one . . . . .	5
GenSyn . . . . .	7
households_to_long_tests . . . . .	10

infectious_time_imputation . . . . .	10
main_parameter_estimation_pipeline . . . . .	11
postprocess_stan_fit . . . . .	13
prepare_stan_data . . . . .	14
prepare_stan_households_from_user_data . . . . .	15
print.GenSynResult . . . . .	16
print.TransmissionChainResult . . . . .	16
running_parameter_estimation . . . . .	17
run_household_stan . . . . .	18
simulate_households . . . . .	19
summarize_individuals . . . . .	21
TransmissionChainAnalysis . . . . .	21

<b>Index</b>	<b>24</b>
--------------	-----------

---

build\_person\_day\_table

*Construct person-day long data*

---

## Description

Expands individual timelines into daily rows and computes within-household infectious counts by infector role for likelihood-based estimation.

## Usage

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

## Arguments

`dt` data.table from [infectious\\_time\\_imputation](#).

`tmax` Integer; maximum day index.

`cases_t` Numeric of length `tmax + 1`; community intensity for days 0..`tmax`.

`covariate_cols` Character; names of covariates to carry (scalars or day series).

## Value

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 requested covariates.

---

 build\_stan\_household\_arrays

*Build Stan data arrays from household list*


---

## Description

Converts a list of per-household data frames into Stan-ready arrays.

## Usage

```
build_stan_household_arrays(
  households,
  T_max = 365L,
  seasonal_forcing_list = NULL,
  alpha_comm_by_role = 0.005,
  beta1 = 0.2,
  beta2 = 0.6,
  V_ref = 1000,
  reference_phi = 1,
  reference_kappa = 1,
  g_peak_day = 2,
  g_width = 4
)
```

## Arguments

households	List of per-household data frames.
T_max	Integer; maximum time horizon.
seasonal_forcing_list	Named list of seasonal forcing vectors.
alpha_comm_by_role, beta1, beta2, V_ref	Numeric; model parameters.
reference_phi, reference_kappa	Numeric; reference parameters.
g_peak_day, g_width	Numeric; infectivity profile parameters.

## Value

Named list for Stan.

---

dataframe\_to\_household\_list

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


---

### Description

Splits a long-format test table into a list of per-household data.tables, preserving the required columns and (optionally) any extra 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

df	Data frame with at least household ID, individual ID, role, test date, infection status, and community risk columns.
hh_col, id_col, role_col, date_col, inf_col, comm_col	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").
keep_extra_cols	Logical; keep additional user columns (default TRUE).

### Value

List of data.tables, one per household (household ID placed first).

---

data\_summarization

*Summarize infection episodes (one row per individual)*


---

### Description

Collapses long testing records to episode-level features and optional covariate summaries. Supply either a single data frame df with the six core columns, or provide all six vectors individually. An optional covariate frame can be merged.

**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**

**df** Data frame with columns HH, individual\_ID, role, test\_date, infection\_status, community\_risk. Ignored when all six vector inputs are provided.

**Household\_ID, Individual\_ID, Household\_role, Sample\_test\_days, Infectious\_status, Community\_rate\_infection** Vectors used only when df is NULL.

**Covariate\_DataFrame** Optional data frame to merge before summarizing; joined by available keys among c("HH", "individual\_ID", "test\_date") (at least individual\_ID required).

**covariate\_cols** Optional character vector restricting which covariates from Covariate\_DataFrame are summarized.

**Details**

Records are ordered by HH, individual\_ID, test\_date. Episodes are runs of infection\_status == 1; first-episode fields refer to the earliest run.

**Value**

A data frame with one row per individual including: HH, individual\_ID, episode counts and timing fields (e.g., infection.detected.start, infection.detected.end, infection.true.duration, last\_negative), infection.infectious.day (comma-separated days), community.risk, role, and (if present) basic summaries for covariates.

**See Also**

[summarize\\_individuals](#), [main\\_parameter\\_estimation\\_pipeline](#)

---

generate\_synthetic\_data\_one

*Generate standardized synthetic data for one household*

---

**Description**

Simulates test-day observations for one household over a date window with community seasonality, within-household transmission, adaptive testing, baseline/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 written to HH.

`hh.size` Integer; household size.

`tests.per.week` Integer; tests per person per week.

`p.comm.base.infant.fix` Numeric; baseline community infection prob/day (infant).

`p.comm.multiplier.sibling`, `p.comm.multiplier.parent`,  
`p.comm.multiplier.elder` Numeric; community multipliers by role.

`p.hh.base.infant` Numeric; baseline within-household infection prob/day (infant source).

`p.hh.multiplier.sibling`, `p.hh.multiplier.parent`,  
`p.hh.multiplier.elder` Numeric; within-household multipliers by source role.

`p.imm.base.sibling`, `p.imm.base.parent`, `p.imm.base.elder` Numeric; baseline immunity at day 1.

`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-infants.  
`p.detect`  
 Numeric; detection probability if infected on a test day.  
`amplitude, phase`  
 Numeric; seasonality parameters for community risk.  
`start_date, end_date`  
 Date; simulation window.  
`Covariates`  
 Logical; generate additional covariates.  
`Covariates_list`  
 Character; covariate names.  
`Covariate_specifications`  
 List; optional per-covariate specs.

### Value

Data frame with columns `HH`, `individual_ID`, `role`, `test_date` (`1 = start_date`), `infection_status`, `community_risk`, and optional covariates.

---

 GenSyn

*GenSyn: household simulation & estimation wrapper*


---

### Description

Runs `main_parameter_estimation_pipeline()` to simulate household data, fit via Bayesian Stan, and assemble a "GenSynResult".

### Usage

```

GenSyn(
  n_households = 50,
  print_plots = TRUE,
  plots = c("daily", "weekly", "timeline", "sar"),
  index_vl_column = "vl_test",
  start_date = as.Date("2024-01-01"),
  end_date = as.Date("2024-12-31"),
  seasonal_forcing_list = NULL,
  max_days = 365,
  alpha_comm_by_role = 0.005,
  beta1 = 0.3,
  beta2 = 0.05,
  delta = 0,
  phi_by_role = c(adult = 1, child = 7, toddler = 7, elderly = 4),

```

```

kappa_by_role = c(adult = 1, child = 1.5, toddler = 1.5, elderly = 1),
latent_shape = 3,
latent_scale = 0.5,
infectious_shape = 3,
infectious_scale = 1,
resolve_shape = 1.5,
resolve_scale = 0.5,
peak_day = 1,
width = 4,
ptrans_threshold = 0.5,
detect_threshold_log10 = 1,
detect_threshold_Ct = 40,
surveillance_interval = 7,
test_daily = FALSE,
viral_testing = "viral load",
V_ref = 3,
V_rho = 2.5,
Ct_50 = 40,
Ct_delta = 2,
VL_params_list = default_VL_params,
Ct_params_list = default_Ct_params,
household_profile_list = default_household_profile,
stan_file = "model.stan",
stan_chains = 4,
stan_iter = 2000,
stan_warmup = 1000,
stan_control = list(adapt_delta = 0.99, max_treedepth = 20),
stan_init = "random",
stan_refresh = 50,
stan_cores = 4
)

```

### Arguments

n_households	Integer; number of households to simulate.
print_plots	Logical; print plots if produced.
plots	Character vector of plot names ("daily", "weekly", "sar", "timeline") or "all".
index_vl_column	Character; viral-load column name for plotting (default "vl_test").
start_date, end_date	Date; study window.
seasonal_forcing_list	Optional named list of role vectors; seasonal forcing.
max_days	Integer; maximum simulated days.
alpha_comm_by_role	Numeric; baseline community acquisition rate.
beta1, beta2	Numeric; transmission coefficients.
delta	Numeric; household size scaling exponent.
phi_by_role	Named numeric vector; susceptibility multipliers by role.

kappa\_by\_role    Named numeric vector; infectivity multipliers by role.  
 latent\_shape, latent\_scale  
                   Numeric; gamma parameters for latent period.  
 infectious\_shape, infectious\_scale  
                   Numeric; gamma parameters for infectious period.  
 resolve\_shape, resolve\_scale  
                   Numeric; gamma parameters for resolution period.  
 peak\_day, width    Numeric; infectivity profile parameters.  
 ptrans\_threshold  
                   Numeric; transmission potential threshold.  
 detect\_threshold\_log10  
                   Numeric; VL detection threshold in log10.  
 detect\_threshold\_Ct  
                   Numeric; Ct detection threshold.  
 surveillance\_interval  
                   Integer; days between scheduled tests.  
 test\_daily        Logical; switch to daily testing after first detection.  
 viral\_testing     Character; "viral load" or "Ct".  
 V\_ref, V\_rho      Numeric; viral load reference and power.  
 Ct\_50, Ct\_delta   Numeric; Ct-based infectivity parameters.  
 VL\_params\_list    Named list; role-specific VL trajectory parameters.  
 Ct\_params\_list    Named list; role-specific Ct trajectory parameters.  
 household\_profile\_list  
                   Named list; household composition probabilities.  
 stan\_file        Path to a Stan model file.  
 stan\_chains, stan\_iter, stan\_warmup  
                   Integers; Stan sampling controls.  
 stan\_control     List; Stan control list.  
 stan\_init        Character or function; Stan initialization.  
 stan\_refresh     Integer; Stan refresh rate.  
 stan\_cores       Integer; CPU cores for Stan.

### Value

A "GenSynResult" list with elements: \$call, \$n\_households, \$results, \$postprocessing, \$plot\_list.

### See Also

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

### Examples

```
## Not run:
# Simulate and estimate
seasonal_forcing_list <- list(
  adult=rep(0.1,365), child=rep(0.1,365), elderly=rep(0.1,365), toddler=rep(0.1,365)
)
```

```
fit <- GenSyn(
  n_households=50,
  seasonal_forcing_list=seasonal_forcing_list, max_days=365,
  stan_chains=4, stan_iter=2000, stan_warmup=1000, stan_cores=4
)

## End(Not run)
```

---

households\_to\_long\_tests

*Flatten simulator households to a long test-day table*

---

### Description

Expands simulator output to per-person per-day rows suitable for legacy processing and plotting.

### Usage

```
households_to_long_tests(households)
```

### Arguments

**households** List of per-household data frames (legacy simulator). Each should include `individual_ID`, `role`, `infection_status`, `infectious_start`, `infectious_end`. Optional: `infection_time`, `infection_resolved`, scalar `community_risk`, and attribute `test_days`.

### Details

If `test_days` is absent, the horizon is inferred from `infectious_end/infection_resolved` (fall-back to 1). Missing `infection_status` is treated as 0. Household index in the list is used for HH.

### Value

Data frame with columns: HH, `individual_ID`, `role`, `test_date`, `infection_status` and (if available) `community_risk`.

---

infectious\_time\_imputation

*Impute infection timelines from delay distributions*

---

### Description

Imputes infection date, infectious start/end, and component delays using gamma distributions, optionally scaled by covariate functions.

**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 from [summarize\\_individuals](#).

`study_start` Date origin for relative day indices.

`latent_par, report_par, infect_par` Lists with shape and scale for latent, reporting, and infectious periods.

`latent_scale_fn, report_scale_fn, infect_scale_fn` Optional functions taking `dt[idx]` (infected rows) and returning numeric scale multipliers.

**Value**

The input `dt` with columns `latent_delay`, `report_delay`, `infect_period`, `inf_date`, `inf_start_date`, `inf_end_date`, and relative-day variants.

---

main\_parameter\_estimation\_pipeline

*Main parameter estimation pipeline*

---

**Description**

End-to-end workflow for household transmission estimation via Bayesian Stan. Works with simulated or user-provided data.

**Usage**

```
main_parameter_estimation_pipeline(
  user_data = NULL,
  synthetic_data = TRUE,
  n_households = 10,
  start_date = as.Date("2024-01-01"),
  end_date = as.Date("2024-12-31"),
  seasonal_forcing_list = NULL,
  max_days = 365,
  alpha_comm_by_role = 0.005,
  beta1 = 0.3,
  beta2 = 0.05,
  delta = 0,
```

```

phi_by_role = c(adult = 1, child = 7, toddler = 7, elderly = 4),
kappa_by_role = c(adult = 1, child = 1.5, toddler = 1.5, elderly = 1),
latent_shape = 3,
latent_scale = 0.5,
infectious_shape = 3,
infectious_scale = 1,
resolve_shape = 1.5,
resolve_scale = 0.5,
peak_day = 1,
width = 4,
ptrans_threshold = 0.5,
detect_threshold_log10 = 1,
detect_threshold_Ct = 40,
surveillance_interval = 7,
test_daily = FALSE,
viral_testing = "viral load",
V_ref = 3,
V_rho = 2.5,
Ct_50 = 40,
Ct_delta = 2,
VL_params_list = default_VL_params,
Ct_params_list = default_Ct_params,
household_profile_list = default_household_profile,
stan_file = "model.stan",
stan_chains = 4,
stan_iter = 2000,
stan_warmup = 1000,
stan_control = list(adapt_delta = 0.99, max_treedepth = 20),
stan_init = "random",
stan_refresh = 50,
stan_cores = 4
)

```

### Arguments

user_data	Optional; data.frame or list of data.frames.
synthetic_data	Logical; if TRUE, simulate data internally.
n_households	Integer; number of households to simulate.
start_date, end_date	Date; analysis window.
seasonal_forcing_list	Optional named list for seasonality.
max_days	Integer; maximum days.
alpha_comm_by_role	Numeric; community acquisition rate.
beta1, beta2	Numeric; transmission coefficients.
delta	Numeric; household size scaling.
phi_by_role, kappa_by_role	Named numeric vectors.
latent_shape, latent_scale	Numeric; latent period parameters.

infectious\_shape, infectious\_scale  
     Numeric; infectious period parameters.  
 resolve\_shape, resolve\_scale  
     Numeric; resolution period parameters.  
 peak\_day, width   Numeric; infectivity profile.  
 ptrans\_threshold, detect\_threshold\_log10, detect\_threshold\_Ct  
     Numeric; testing thresholds.  
 surveillance\_interval  
     Integer; days between tests.  
 test\_daily       Logical; daily testing after detection.  
 viral\_testing    Character; "viral load" or "Ct".  
 V\_ref, V\_rho     Numeric; viral load parameters.  
 Ct\_50, Ct\_delta   Numeric; Ct parameters.  
 VL\_params\_list, Ct\_params\_list, household\_profile\_list  
     Named lists.  
 stan\_file        Path to Stan model file.  
 stan\_chains, stan\_iter, stan\_warmup  
     Integers; Stan sampling controls.  
 stan\_control     List; Stan control list.  
 stan\_init        Character or function; Stan initialization.  
 stan\_refresh     Integer; Stan refresh.  
 stan\_cores       Integer; CPU cores for Stan.

**Value**

A list with raw\_simulation, stan\_data, fit, posterior\_summary, and diagnostic\_df.

**See Also**

[GenSyn](#)

---

postprocess\_stan\_fit   *Tidy posterior summary for role multipliers*

---

**Description**

Extracts posterior summaries from a stanfit.

**Usage**

```
postprocess_stan_fit(fit)
```

**Arguments**

fit                   A stanfit object.

**Value**

A data frame with posterior summaries.

---

prepare_stan_data	<i>Prepare Stan data from diagnostic testing records</i>
-------------------	--

---

## Description

Converts raw diagnostic testing data into Stan-ready format with the `time_since_infection` matrix for soft-gate latent period modeling.

## Usage

```
prepare_stan_data(
  raw_data,
  seasonal_forcing_list = NULL,
  viral_testing_mode = "viral load",
  T_max = 365,
  V_ref = 3,
  V_rho = 2.5,
  peak_day = 2,
  width = 4,
  alpha_comm_by_role = 0.005,
  reference_phi = 1,
  reference_kappa = 1
)
```

## Arguments

<code>raw_data</code>	Data frame with columns <code>hh_id</code> , <code>person_id</code> , <code>role</code> , <code>day_index</code> , <code>test_result</code> , and optionally <code>pcr_sample</code> .
<code>seasonal_forcing_list</code>	Named list with seasonal forcing vectors by role.
<code>viral_testing_mode</code>	Character; "viral load" or "Ct".
<code>T_max</code>	Integer; maximum time horizon (days).
<code>V_ref</code> , <code>V_rho</code>	Numeric; viral load transmission parameters.
<code>peak_day</code> , <code>width</code>	Numeric; infectivity profile parameters.
<code>alpha_comm_by_role</code>	Numeric; community acquisition rate.
<code>reference_phi</code> , <code>reference_kappa</code>	Numeric; reference parameters.

## Value

Named list suitable for Stan.

---

```
prepare_stan_households_from_user_data
```

*Prepare per-household inputs for the Stan RSV/VL model*

---

## Description

Converts user data to a list of per-household data frames required by the Stan RSV/VL pipeline. Accepts either a long test-day table or a per-person episodes table.

## Usage

```
prepare_stan_households_from_user_data(
  user_data,
  role_levels = c("adult", "child", "elderly", "toddler"),
  vl_mode = c("from_long", "auto"),
  vl_source = c("none", "column", "simulate"),
  vl_column = NULL,
  start_date = NULL,
  end_date = NULL
)
```

## Arguments

user_data	Either: <ul style="list-style-type: none"> <li>• <i>Long format</i>: columns HH, individual_ID, role, test_date, infection_status (optionally a VL column); or</li> <li>• <i>Per-person format</i>: columns hh_id, person_id, role, infection_time, infectious_start, infectious_end (optionally vl_full_trajectory).</li> </ul>
role_levels	Character vector of allowed roles after normalization.
vl_mode	Character; currently used only to trigger trajectory building.
vl_source	Character; one of "none", "column", "simulate".
vl_column	Optional name of the VL column when vl_source = "column".
start_date, end_date	Optional Date bounds.

## Value

A named list of data frames (one per household).

---

<code>print.GenSynResult</code>	<i>Print method for GenSynResult</i>
---------------------------------	--------------------------------------

---

**Description**

Print method for GenSynResult

**Usage**

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

**Arguments**

<code>x</code>	A GenSynResult object.
<code>...</code>	Unused.

**Value**

`x`, invisibly.

---

<code>print.TransmissionChainResult</code>	<i>Print a TransmissionChainResult</i>
--	--

---

**Description**

Nicely prints sections available in a TransmissionChainResult returned by [TransmissionChainAnalysis](#).

**Usage**

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

**Arguments**

<code>x</code>	A TransmissionChainResult object.
<code>...</code>	Passed to or from other methods (unused).

**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

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

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_*`.

---

run_household_stan	<i>Run the Stan household model</i>
--------------------	-------------------------------------

---

### Description

Wrapper around `rstan::sampling()` with reasonable defaults.

### Usage

```
run_household_stan(
  stan_data,
  stan_file = "model.stan",
  chains = 4,
  iter = 2000,
  warmup = 1000,
  control = list(adapt_delta = 0.99, max_treedepth = 20),
  init = "random",
  refresh = 50,
  cores = 4,
  stan_code = NULL,
  package_name = NULL
)
```

### Arguments

stan_data	Named list from <code>build_stan_household_arrays</code> or <code>prepare_stan_data</code> .
stan_file	Path to Stan model file.
chains, iter, warmup	Stan MCMC settings.
control	List passed to <code>rstan::sampling()</code> .
init, refresh, cores	Stan parameters.
stan_code	Optional character string with Stan program code.
package_name	Optional package name for <code>system.file</code> lookup.

**Value**

A stanfit object.

---

simulate_households	<i>Simulate households (RSV/VL engine)</i>
---------------------	--

---

**Description**

Generates synthetic household data for the RSV/VL–Stan pipeline. Returns a list with stacked data frame, per-household list, and long testing records (diagnostic\_df).

**Usage**

```
simulate_households(
  n_households,
  start_date = as.Date("2024-01-01"),
  end_date = as.Date("2024-12-31"),
  max_days = 365,
  seasonal_forcing_list = NULL,
  alpha_comm_by_role = 0.005,
  beta1 = 0.3,
  beta2 = 0.05,
  delta = 0,
  phi_by_role = c(adult = 1, child = 7, toddler = 7, elderly = 4),
  kappa_by_role = c(adult = 1, child = 1.5, toddler = 1.5, elderly = 1),
  latent_shape = 3,
  latent_scale = 0.5,
  infectious_shape = 3,
  infectious_scale = 1,
  resolve_shape = 1.5,
  resolve_scale = 0.5,
  peak_day = 1,
  width = 4,
  ptrans_threshold = 0.5,
  detect_threshold_log10 = 1,
  detect_threshold_Ct = 40,
  surveillance_interval = 7,
  test_daily = FALSE,
  viral_testing = "viral load",
  V_ref = 3,
  V_rho = 2.5,
  Ct_50 = 40,
  Ct_delta = 2,
  VL_params_list = default_VL_params,
  Ct_params_list = default_Ct_params,
  household_profile_list = default_household_profile,
  verbose = FALSE
)
```

**Arguments**

n_households	Integer; number of households.
start_date, end_date	Date study window.
max_days	Integer; simulation horizon (days).
seasonal_forcing_list	Optional named list (adult, child, elderly, toddler) for seasonal forcing.
alpha_comm_by_role	Numeric; baseline community acquisition rate.
beta1, beta2	Numeric; transmission coefficients.
delta	Numeric; household size scaling exponent.
phi_by_role	Named numeric vector; susceptibility multipliers by role.
kappa_by_role	Named numeric vector; infectivity multipliers by role.
latent_shape, latent_scale	Numeric; gamma parameters for latent period.
infectious_shape, infectious_scale	Numeric; gamma parameters for infectious period.
resolve_shape, resolve_scale	Numeric; gamma parameters for resolution period.
peak_day, width	Numeric; infectivity profile parameters.
ptrans_threshold	Numeric; transmission potential threshold.
detect_threshold_log10	Numeric; viral load detection threshold in log10.
detect_threshold_Ct	Numeric; Ct value detection threshold.
surveillance_interval	Integer; days between scheduled tests.
test_daily	Logical; switch to daily testing after first detection.
viral_testing	Character; one of "viral load" or "Ct".
V_ref, V_rho	Numeric; viral load reference and power for transmission.
Ct_50, Ct_delta	Numeric; Ct-based infectivity parameters.
VL_params_list	Named list; role-specific VL trajectory parameters.
Ct_params_list	Named list; role-specific Ct trajectory parameters.
household_profile_list	Named list; household composition probabilities.
verbose	Logical; print progress information.

**Value**

A list with hh\_df (stacked data frame), households (per-household list), and diagnostic\_df (long testing records).

---

summarize\_individuals *Summarize individual-level infection data*

---

### Description

Produces one row per individual with detection windows, inferred infectious windows (relative days), index-case flags, observation bounds, role/age classification, and aggregated covariates. 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 household-level data frames/tables.

`study_start, study_end` Date. Analysis window defining day indices.

`day_series_covariates` Logical; add day-series list-columns (default TRUE).

`series_cols` Character or NULL; covariates to series-encode (default NULL = all).

### Value

`data.table` with one row per individual containing detection and infectious windows, flags, observation bounds, role/age, and covariate summaries.

---

TransmissionChainAnalysis

*TransmissionChainAnalysis: analyze user-supplied household data*

---

### Description

Runs the end-to-end workflow on *user-provided* data using Bayesian Stan estimation. This function does not simulate data; it expects input in tabular form (see Details).

**Usage**

```

TransmissionChainAnalysis(
  user_data,
  plots = c("daily", "weekly", "timeline", "sar"),
  print_plots = FALSE,
  index_vl_column = "vl_test",
  start_date = as.Date("2024-01-01"),
  end_date = as.Date("2024-12-31"),
  seasonal_forcing_list = NULL,
  max_days = 365,
  stan_file = "model.stan",
  stan_chains = 4,
  stan_iter = 2000,
  stan_warmup = 1000,
  stan_control = list(adapt_delta = 0.99, max_treedepth = 20),
  stan_init = "random",
  stan_refresh = 50,
  stan_cores = 4,
  vl_mode = c("auto", "from_long"),
  vl_source = c("column", "simulate", "none"),
  vl_column = NULL,
  role_levels = c("adult", "child", "elderly", "toddler")
)

```

**Arguments**

<code>user_data</code>	A data.frame or a list of data.frames describing household testing or episode data (see Details for accepted shapes). Lists are row-bound internally.
<code>plots</code>	Character vector of plot names to build when compatible data are present: <code>c("daily", "weekly", "timeline", "sar", "all")</code> .
<code>print_plots</code>	Logical; print plots if produced.
<code>index_vl_column</code>	Optional character; name of the viral-load column used by SAR-by-VL plotting. Defaults to "vl_test" when present.
<code>start_date, end_date</code>	Date study window.
<code>seasonal_forcing_list</code>	Optional named list of role vectors for forcing.
<code>max_days</code>	Integer; maximum modeled days.
<code>stan_file</code>	Path to a Stan model file.
<code>stan_chains, stan_iter, stan_warmup</code>	Integers; Stan sampling controls.
<code>stan_control</code>	List; passed to <code>rstan::sampling(..., control = ...)</code> .
<code>stan_init</code>	Character or function; Stan initialization.
<code>stan_refresh</code>	Integer; Stan refresh rate.
<code>stan_cores</code>	Integer; CPU cores for Stan.
<code>vl_mode</code>	One of <code>c("auto", "from_long")</code> indicating how to derive VL.
<code>vl_source</code>	One of <code>c("column", "simulate", "none")</code> indicating the VL source.
<code>vl_column</code>	Optional character; the VL column when <code>vl_source="column"</code> .
<code>role_levels</code>	Character vector; canonical role levels for normalization.

## Details

### Accepted input shapes:

- Long testing table with columns like HH, individual\_ID, role, test\_date, infection\_status, and optionally a VL column.
- Per-person episode table with hh\_id, person\_id, role, infection\_time, infectious\_start, infectious\_end, and optionally a viral-load trajectory.

## Value

An object of class "TransmissionChainResult" containing:

- \$results: Contains fit (the Stan object) and posterior\_summary.
- \$postprocessing: Stan posterior summary.
- \$plot\_list: named list of ggplot objects when built.

## See Also

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

## Examples

```
## Not run:
T_max <- 12
df_person <- data.frame(
  hh_id = c("HH1", "HH1", "HH1", "HH2", "HH2", "HH2"),
  person_id = c(1, 2, 3, 1, 2, 3),
  role = c("adult", "child", "elderly", "adult", "child", "elderly"),
  infection_time = c(2, 4, NA, 1, 3, NA),
  infectious_start = c(3, 6, NA, 2, 5, NA),
  infectious_end = c(8, 9, NA, 7, 9, NA),
  infection_resolved = c(9, 10, NA, 8, 10, NA)
)
seasonal_forcing_list <- list(
  adult = rep(1, T_max), child = rep(1, T_max),
  elderly = rep(1, T_max), toddler = rep(1, T_max)
)
result <- TransmissionChainAnalysis(
  user_data = df_person,
  seasonal_forcing_list = seasonal_forcing_list,
  max_days = T_max,
  stan_chains = 1, stan_iter = 300, stan_warmup = 150
)

## End(Not run)
```

# Index

`build_person_day_table`, [2](#), [17](#)  
`build_stan_household_arrays`, [3](#)  
  
`data_summarization`, [4](#)  
`dataframe_to_household_list`, [4](#)  
  
`generate_synthetic_data_one`, [5](#)  
`GenSyn`, [7](#), [13](#), [23](#)  
  
`households_to_long_tests`, [10](#)  
  
`infectious_time_imputation`, [2](#), [10](#)  
  
`main_parameter_estimation_pipeline`, [5](#),  
[7](#), [9](#), [11](#), [23](#)  
  
`postprocess_stan_fit`, [13](#)  
`prepare_stan_data`, [14](#)  
`prepare_stan_households_from_user_data`,  
[15](#)  
`print.GenSynResult`, [16](#)  
`print.TransmissionChainResult`, [16](#)  
  
`run_household_stan`, [18](#)  
`running_parameter_estimation`, [17](#)  
  
`simulate_households`, [19](#)  
`summarize_individuals`, [5](#), [11](#), [21](#)  
  
`TransmissionChainAnalysis`, [9](#), [16](#), [21](#)