Package 'SmokingHistoryGenerator'

June 14, 2026

Type Package

Title R Package for the Smoking History Generator

Version 7.0.0

Date 2026-06-13

Maintainer John Clarke <john.clarke@cornerstonenw.com>

Description Efficient R interface to the Cancer Intervention and Surveillance Modeling Network (CISNET) Smoking History Generator microsimulation engine, which synthesizes individual smoking histories (initiation, cessation, intensity) and ages at death from calibrated initiation, cessation, cigarettes-per-day, and mortality tables. The wrapper exposes fixed-cohort and population data-frame simulation, multi-threaded segmentation, reproducible pseudo-random streams (L'Ecuyer RngStream MRG32k3a or Matsumoto--Nishimura Mersenne Twister), legacy CLI-style configuration files, and portable YAML configuration save/load with optional split smoking and mortality parameter bundles. Methods follow Jeon et al. (2012) <doi:10.1111/j.1539-6924.2011.01775.x>. Random number generators: Matsumoto and Nishimura (1998) <doi:10.1145/272991.272995>; L'Ecuyer (1999) <doi:10.1287/opre.47.1.159>; L'Ecuyer et al. (2002) <doi:10.1287/opre.50.6.1073.358>.

RoxygenNote 7.3.3

URL https://github.com/NCI-CISNET/shg-r

Imports methods, Rcpp (>= 1.0.2), yaml

Suggests testthat (>= 3.0.0), glue, httr2

Config/testthat/edition 3

LinkingTo Rcpp

Encoding UTF-8

License GPL-3

NeedsCompilation yes

Author John Clarke [aut, cre] (Author and maintainer of SHG R package wrapper for the SHG),
Ben Racine [aut] (Co-author of the original SHG),

Martin Krapcho [aut] (Co-author of the original SHG),
Alexander Gaenko [aut] (Co-author of the original SHG),
Makoto Matsumoto [ctb, cph] (Mersenne Twister mt19937
(src/mersenne_class.*); copyright notice in source),
Takuji Nishimura [ctb, cph] (Mersenne Twister mt19937
(src/mersenne_class.*); copyright notice in source),
Pierre L'Ecuyer [ctb, cph] (RngStream MRG32k3a (src/RngStream.*);
copyright notice in source)

Repository CRAN

Date/Publication 2026-06-14 06:50:02 UTC

Contents

`SmokingHistoryGenerator-package`

R Package for the Smoking History Generator

Description

Efficient R interface to the Cancer Intervention and Surveillance Modeling Network (CISNET) Smoking History Generator microsimulation engine, which synthesizes individual smoking histories (initiation, cessation, intensity) and ages at death from calibrated initiation, cessation, cigarettes-per-day, and mortality tables. The wrapper exposes fixed-cohort and population data-frame simulation, multi-threaded segmentation, reproducible pseudo-random streams (L'Ecuyer RngStream MRG32k3a or Matsumoto–Nishimura Mersenne Twister), legacy CLI-style configuration files, and portable YAML configuration save/load with optional split smoking and mortality parameter bundles. Methods follow Jeon et al. (2012) <doi:10.1111/j.1539-6924.2011.01775.x>. Random number generators: Matsumoto and Nishimura (1998) <doi:10.1145/272991.272995>; L'Ecuyer (1999) <doi:10.1287/opre.47.1.159>; L'Ecuyer et al. (2002) <doi:10.1287/opre.50.6.1073.358>.

Details

Default calibrated inputs ship under `system.file("extdata", "2018", package = "SmokingHistoryGenerator")` as `smok/*.csv` and `mort/*.csv` (NHIS-1965-2018 csv-partial). Set `input_data_folder` and per-table filenames accordingly. Wide legacy `.txt` tables remain supported. Full NHIS-style tables are distributed separately (Zenodo; the installed README is at `system.file("README.md", package = "SmokingHistoryGenerator")`).

Author(s)

John Clarke [aut, cre] (Author and maintainer of SHG R package wrapper for the SHG), Ben Racine [aut] (Co-author of the original SHG), Martin Krapcho [aut] (Co-author of the original SHG), Alexander Gaenko [aut] (Co-author of the original SHG), Makoto Matsumoto [ctb, cph] (Mersenne Twister mt19937 (src/mersenne_class.*); copyright notice in source), Takuji Nishimura [ctb, cph] (Mersenne Twister mt19937 (src/mersenne_class.*); copyright notice in source), Pierre L'Ecuyer [ctb, cph] (RngStream MRG32k3a (src/RngStream.*); copyright notice in source)

Maintainer: John Clarke <john.clarke@cornerstonenw.com>

References

CISNET modelling applications that use the Smoking History Generator are listed on the CISNET website at https://cisnet.cancer.gov/.

See Also

R package source: https://github.com/NCI-CISNET/shg-r

`getConfig` *Get SHG Configuration*

Description

Returns the current configuration of the SHG instance as an R list. Can include debug information when debug=TRUE.

Arguments

debug	Logical. If TRUE, includes additional debug information such as RNG state fingerprint, package version, system info, and memory usage. If not provided, defaults to FALSE.

Details

Get current SHG configuration

Value

A list containing the current intent configuration including: config_version, rng_strategy, number_of_segments, num_threads, seeds, input file paths (including mortality_filename), smok_params_source, mort_params_source, and mort_params_type (from load_params, else NA), immediate_cessation_year, inferred cohort_year (single-cohort runs; otherwise NA), repeat/race/sex after runSimFromFixedValues (otherwise NA), and timestamp. This method returns currently applied values (including unresolved auto values such as -1 for segments/threads). Use `getReproConfig()` to export effective runtime values from the last completed simulation. seeds always returns concrete values (explicit user seeds or defaults). If debug=TRUE, also includes rng_state_fingerprint, package_version, package_source, r_version, platform, and memory_usage.

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg$rng_strategy <- "RngStream"
shg$number_of_segments <- 4
config <- shg$getConfig()
names(config)
```

---

getReproConfig *Get Reproducibility Configuration*

Description

Returns a configuration list that captures effective runtime settings from the last completed simulation.

Arguments

debug	Logical. If TRUE, includes additional debug information such as RNG state fingerprint, package version, system info, and memory usage. If not provided, defaults to FALSE.

Details

Get reproducibility-focused SHG configuration from last run

Value

A list like `getConfig()` for the last completed simulation, but with `number_of_segments` as the effective segment count used and **without** `num_threads` (thread count must not affect simulation outcomes for fixed seeds and segment layout; consumers default to auto threads when reloading). Errors if no simulation has completed on the instance.

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg$runSimFromFixedValues(500, 0, 0, 1950)
repro <- shg$getReproConfig()
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg$runSimFromFixedValues(500, 0, 0, 1950)
repro <- shg$getReproConfig()
names(repro)
```

`get_data_shape` *get_data_shape method*

Description

Returns a list containing information about the shape/dimensions of the current input data files. It reads the configured parameter files directly and does not require running a simulation first.

Value

A list with data shape information including races, sexes, cohorts, age ranges, cohort boundaries, and CPD statistics.

A named list describing loaded parameter tables: counts of races, sexes, and cohorts; age ranges for initiation, cessation, mortality, and CPD; mortality calendar years; CPD intensity groups; and CPD row load/skip counts. Intended for validation before running simulations.

`LegacyRunWebVersion` *LegacyRunWebVersion method*

Description

This method offers a way to configure and run a simulation from an input configuration file. Rather than return a R DataFrame, it produces results in an output file. It works in the same as calling the CLI version of the Smoking History Generator with a single input file parameter.

Arguments

`input_file_name`
: Path to a Legacy web-style configuration file. Paths inside the file are resolved relative to the R process working directory (the `input_data_folder` property is ignored). Sample text configs live under `tests/testdata/legacy-web-examples/` in the package source; for installed use, build a config with absolute paths from `system.file("extdata", "2018", package = "SmokingHistoryGenerator")`.

Value

No return value. Called for side effects: runs the CLI-style engine and writes semicolon-separated results to `OUTPUTFILE` and diagnostics to `ERRORFILE` as specified in the configuration file (properties on the R object are ignored).

Examples

```
shg <- new(SHGInterface)
d <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
tf <- tempfile(fileext = ".txt")
writeLines(c(
  "RNGSTRATEGY=RngStream",
  "RNGSTREAM_SEED=12345,12345,12345,12345,12345,12345",
  "RACE=0", "SEX=0", "YOB=1950", "CESSATION_YR=0", "REPEAT=100",
  paste0("INIT_PROB=", file.path(d, "smoking", "initiation.csv")),
  paste0("CESS_PROB=", file.path(d, "smoking", "cessation.csv")),
  paste0("MORTALITY_PROB=", file.path(d, "mortality", "acm.csv")),
  paste0("CPD_DATA=", file.path(d, "smoking", "cpd.csv")),
  paste0("OUTPUTFILE=", tempfile("out_", fileext = ".txt")),
  paste0("ERRORFILE=", tempfile("err_", fileext = ".txt"))
), tf)
shg$LegacyRunWebVersion(tf)
```

Rcpp_SHGInterface *Rcpp SHG Interface Class*

Description

This module provides an Rcpp interface to the Smoking History Generator (SHG) application, including intent-oriented config methods (`getConfig`/`useConfig`) and reproducibility export (`getReproConfig`).

Details

Rcpp SHG Interface Class

Value

The exported reference class `SHGInterface` (see `SHGInterface`): an external pointer to the C++ engine used for all simulations.

Rcpp_SHGInterface-class

Class "Rcpp_SHGInterface"

Description

This is a description.

Extends

Class "C++Object" (Rcpp base class; see **Rcpp**), directly.

All reference classes extend the S4 virtual class envRefClass; see **methods**.

Note

Further Notes

Author(s)

I am the author

References

These are my references

See Also

See also Sample

Examples

```
showClass("Rcpp_SHGInterface")
```

runSimFromDataFrame *runSimFromDataFrame method*

Description

runSimFromDataFrame offers a way to configure and run a simulation from an existing R dataframe. It returns a dataframe of simulated smoking histories with the same number of rows and order as the input dataframe.

Arguments

dfPopulation	The input dataframe with named columns for race, sex, and birth_cohort

Details

On Windows, `output_file` (direct disk output) cannot be combined with multi-threaded execution (`num_threads` not equal to 1). The call stops with an error before loading inputs or writing files. Use the default in-memory DataFrame return value, or set `num_threads <- 1` to write a file.

Value

If `attach_run_info = FALSE`, a `data.frame` with one row per input individual (same order) and columns `smoking_initiation_age` (`-999` = never smoker), `smoking_cessation_age`, `age_at_death`, and `cigarettes_per_day`. Constant `race`, `sex`, or `birth_cohort` are omitted when uniform. If `attach_run_info = TRUE`, the same four-component bundle as `shg_run` (`results`, `original_config`, `repro_config`, `run_info`); see that help page for definitions.

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
pop <- data.frame(race = 0, sex = 0, birth_cohort = 1950)
hist <- shg$runSimFromDataFrame(pop)
head(hist)
```

`runSimFromFixedValues` *runSimFromFixedValues method*

Description

runSimFromFixedValues offers a way to configure and run a simulation from fixed values for race, sex, and birth year cohort rather than passing a data frame. It returns a dataframe of simulated smoking histories for n individuals.

Arguments

`repeat`	The number of individuals to simulate
`race`	(default = 0 and refers to all races combined)
`sex`	(0 for male, 1, for female)
`cohort_year`	(four digit birth cohort year)

Value

If `attach_run_info = FALSE`, a `data.frame` of `repeat` simulated individuals with columns `smoking_initiation_age` (`-999` = never smoker), `smoking_cessation_age`, `age_at_death`, and `cigarettes_per_day`. If `attach_run_info = TRUE`, the same four-component bundle as `shg_run`; see that help page for definitions.

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
hist <- shg$runSimFromFixedValues(500, 0, 0, 1950)
head(hist)
```

SHGInterface *SHGInterface*

Description

The SHG Interface class provides an Rcpp interface to the Smoking History Generator (SHG)

Details

SHGInterface Class

Value

An `SHGInterface` reference object (R6-style external pointer) wrapping the C++ simulation engine. Configure fields, then call simulation methods such as `runSimFromFixedValues` or `runSimFromDataFrame`; use `getConfig` / `useConfig` for settings and `shg_load_params` (via `load_params()`) for parameter bundles.

Fields

`number_of_segments` Number of segments to use for simulation. Use -1 for auto-calculation (default), 1 for single segment, or N > 1 for explicit segment count. Auto-calculation uses: min(cores * 10, repeat / 1000). Note: MersenneTwister RNG is restricted to 1 segment.

`num_threads` Thread count: -1 = auto (all cores, multi-threaded), 1 = single-threaded, N = use N threads. Default: -1. Note: MersenneTwister RNG requires num_threads = 1.

`rng_strategy` 'RngStream' for MRG32k3a (default) or 'MersenneTwister' for Mersenne Twister RNG. 'RngStream' is recommended for reproducibility especially with multi-threaded simulations. Note: MersenneTwister RNG is restricted to single-segment, non-parallel execution due to limitations in maintaining IID properties across segments.

`input_data_folder` Set or get the base folder for input data files

`initiation_filename` Set or get the initiation filename

`cessation_filename` Set or get the cessation filename

`mortality_filename` Set or get the mortality probabilities filename (e.g. acm.csv or ocm-excl-lung-cancer.csv)

`smok_params_source` URL or local path of the last load_params() smoking zip (empty if unset)

`mort_params_source` URL or local path of the last load_params() mortality zip (empty if unset)

`mort_params_type` Mortality table from last load_params(): acm or ocm (empty if unset)

`params_cache_dir` Read-only. Directory where load_params() stores extracted bundles (same as shg_params_cache_dir()). Delete this folder to clear the cache manually.

`cpd_filename` Set or get the cpd filename

`immediate_cessation_year` Set or get Immediate Cessation Year; If 0, no immediate cessation

`mt_seeds` Set or get MersenneTwister seeds. Must be a numeric vector of exactly 4 values (one for each stream: initiation, cessation, life table, individual). If not set, default seeds are used. Only used when rng_strategy is "MersenneTwister".

`rngstream_seed` Set or get RngStream seed. Must be a numeric vector of exactly 6 values (a single seed vector that generates 4 substreams, one for each stream: initiation, cessation, life table, individual). If not set, default seed is used. Only used when rng_strategy is "RngStream".

`shg_apply_config` *Apply a sparse or complete configuration (defaults first, then overlay)*

Description

Resets the instance with `shg_reset_defaults`, then applies `config`. When `smok_params_source` and `mort_params_source` are set, derived table paths are stripped and parameters are restored via `shg_load_params` (same idea as `shg_load_config`). Otherwise settings are applied with `shg$useConfig()` only; explicit `input_data_folder` / per-table filenames in `config` are preserved.

Usage

```
shg_apply_config(shg, config = list())
```

Arguments

`shg`	An `SHGInterface` instance.
`config`	Named list (may be empty).

Value

shg, invisibly.

See Also

`shg_reset_defaults`, `shg_load_config`

shg_clear_params_cache *Clear the SHG parameter cache*

Description

Clear the SHG parameter cache

Usage

```
shg_clear_params_cache()

clear_params_cache()
```

Value

Invisibly, the cache path that was removed (`character`, length one), or `character()` if the directory did not exist (a message is printed in that case). Called for side effects when clearing disk cache; return value is mainly for scripting.

shg_config_bundle *Build a config list suitable for inspection or advanced serialization*

Description

Returns `shg$getConfig()`: engine fields, provenance, and run metadata when available (e.g. after `runSimFromFixedValues`).

Usage

```
shg_config_bundle(shg)
```

Arguments

`shg`	An `SHGInterface` instance.

Value

A plain list (see `shg_save_config` for portable YAML).

See Also

`shg_save_config`, `shg_load_config`, `shg_run`

shg_load_config *Load engine state and parameters from a YAML config file*

Description

Reads the YAML file, applies engine settings with `useConfig()`, then restores parameter tables via `shg_load_params` when the cache is missing or stale (using `smok_params_source`, `mort_params_source`, and `mort_params_type` stored in the file).

Usage

```
shg_load_config(shg, path)

shg_use_config_bundle(shg, path)
```

Arguments

`shg`	An `SHGInterface` instance.
`path`	Path to a YAML file produced by `shg_save_config` or compatible hand-written YAML.

Details

Private GitHub downloads use the `GITHUB_PAT` environment variable when needed (same as `shg_load_params`).

Value

The parsed config list (same object to pass to `shg_run` / `runSim`). Return value is visible so you can assign: `config <- shg_load_config(shg, "my-run.yml")`.

See Also

`shg_save_config`, `shg_run`

shg_load_params *Load SHG smoking and mortality parameter bundles and configure the instance*

Description

Downloads (or reuses locally cached copies of) separate **shg-params** smoking and mortality release zips, merges them into an engine layout under the cache, and sets `input_data_folder` plus relative input filenames on the `SHGInterface` instance.

Each zip uses the **shg-params** release layout (`params/` CSVs plus `metadata.yml`). The simulator expects `smok/*.csv` and `mort/*.csv` under one folder; this function materializes that tree from the two zips.

Usage

```
shg_load_params(
  shg,
  smoking_url = NULL,
  mortality_url = NULL,
  mort_params_type = c("acm", "ocm")
)
```

Arguments

shg	An SHGInterface instance.
smoking_url	URL or local path to the smoking .zip bundle.
mortality_url	URL or local path to the mortality .zip bundle.
mort_params_type	"acm" (**default**) or "ocm".
	For private GitHub-hosted zips, set GITHUB_PAT before downloading.

Value

The SHGInterface instance, invisibly.

Download timeouts

Options shg.params.download.timeout_sec (default 600) and shg.params.download.connect_sec (default 60) control HTTPS transfers when **httr2** is installed.

---

shg_params_cache_dir *Return the directory where downloaded parameter sets are cached*

Description

Return the directory where downloaded parameter sets are cached

Usage

```
shg_params_cache_dir()
```

Value

A length-one character path (visible). Same location as the read-only params_cache_dir field on SHGInterface. Extracted smoking and mortality bundles from shg_load_params are stored under this directory (via tools::R_user_dir(..., "cache")).

shg_params_summary *Summarize currently configured SHG parameter tables*

Description

Returns a compact "shape" summary of the currently configured parameter files (races, sexes, cohorts, age ranges, and CPD coverage). This works after either `shg_load_params()` or manual file-path configuration on an `SHGInterface` instance.

Usage

```
shg_params_summary(shg)
```

Arguments

shg	An `SHGInterface` instance.

Value

A named list with nested sections `initiation`, `cessation`, `mortality`, and `cpd`, plus top-level dimensions/cohort metadata for convenience.

The `cpd$note` field summarizes whether initiation rows below the CPD minimum age are effectively ignorable (all zeros and/or dots), or if there are non-zero initiation values that may indicate a mismatch.

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg_params_summary(shg)
```

shg_reset_defaults *Reset an SHG instance to factory defaults*

Description

Restores the same engine fields as a freshly constructed `SHGInterface` (package extdata paths, default RNG strategy, auto segments/threads, cleared seeds and bundle provenance).

Usage

```
shg_reset_defaults(shg)
```

Arguments

shg	An `SHGInterface` instance.

Value

shg, invisibly.

See Also

shg_apply_config

---

shg_run *Run a fixed cohort simulation from a config list*

Description

shg_run() and SHGInterface$runSim() call the same implementation. Validates required keys and calls runSimFromFixedValues. If repeat, individuals, and N are all omitted, repeat defaults to 1000L. If config is a single string, it is treated as a path and read with read_yaml (use after shg_load_config with the returned list is preferred).

Usage

```
shg_run(shg, config, attach_run_info = TRUE)
```

Arguments

shg	An SHGInterface instance.
config	Named list from shg_load_config, or a YAML path.
attach_run_info	If TRUE (default), returns a run bundle list; set to FALSE to return only the simulation data.frame.

Value

If attach_run_info is FALSE, the data.frame from runSimFromFixedValues. If TRUE, a list with four components:

results Simulation data.frame (see runSimFromFixedValues).

original_config Intent list passed into the run (cohort scalars, smok_params_source, mort_params_source, mort_params_type, engine options); for runSim/shg_run, the config list or parsed YAML.

repro_config Effective post-run settings from getReproConfig (resolved segments/threads, RNG, paths, bundle provenance, cohort metadata).

run_info Execution metadata (UTC time, host, R and package/engine versions; built by internal .shg_build_run_info()).

See Also

shg_load_config, shg_save_config

shg_save_config *Save a portable reproducibility config as YAML*

Description

Writes a YAML file containing smok_params_source, mort_params_source, mort_params_type, engine settings (RNG, seeds, effective segment count), fixed-run parameters (repeat, race, sex, cohort_year), and immediate_cessation_year. Omits derived paths so the bundle stays portable; those paths are restored by shg_load_params.

Usage

```
shg_save_config(shg, path, quiet = FALSE, results = NULL)
```

Arguments

shg	An SHGInterface instance.
path	Destination file path (usually .yml or .yaml).
quiet	If TRUE, suppress the explanatory message printed on save.
results	Optional simulation data.frame; when supplied, the YAML includes a results block (content_md5, summary) and a single repro_digest (MD5 of engine settings plus R session string) for verification (see run bundles from shg_run with attach_run_info = TRUE). Summary uses never_smokers / ever_smokers with count and fraction (YAML reserves bare n); ever_smokers$cpd_mode is the most common rounded CPD among ever smokers. Mean/sd blocks use n_obs (finite values excluding -999). Initiation and cessation means are among ever smokers only; age_at_death$ever_smokers holds death-age stats (not the same list as top-level ever_smokers). age_at_death subgroup n_obs can be smaller if age is missing or sentinel for some individuals. The simulator encodes never smokers with smoking_initiation_age == -999, not NA; NA initiation is excluded from never_smokers, top-level ever_smokers, and those ever-only means.

Details

Prefer the method form shg$save_config(path) (same implementation). The functional form shg_save_config(shg, path) is a convenience wrapper.

Saving reads shg$getReproConfig(debug = FALSE) after your workflow. Portable save is allowed only when the **last completed simulation** on this instance was runSimFromFixedValues — a subsequent runSimFromDataFrame (population run) clears that until you run runSimFromFixedValues again. Use shg$last_completed_sim_was_fixed_cohort() to test programmatically.

The run scalars (repeat, race, sex, cohort_year) come from that fixed cohort run. Engine fields (number_of_segments, rng_strategy, seeds) reflect **effective** values from it when you used defaults or auto settings for segments. Thread count is intentionally omitted from the portable repro

file (outcomes must not depend on it). Optional `results` adds content hashes and compact numeric summaries for verification. If `results` is omitted, the YAML has no `results` block and no `repro_digest` (only engine and cohort fields for portability).

If the last run was not a fixed cohort simulation, or fixed cohort metadata are missing, saving fails with an error.

Value

`path`, invisibly.

See Also

`shg_load_config`, `shg_run`

Examples

```
shg <- new(SHGInterface)
shg$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg$smok_params_source <- "example-smoking"
shg$mort_params_source <- "example-mortality"
shg$mort_params_type <- "acm"
sim <- shg$runSimFromFixedValues(500, 0, 0, 1950)
tf <- tempfile(fileext = ".yml")
shg_save_config(shg, tf, results = sim)
```

---

shg_write_config_yaml *Write a configuration list to YAML*

Description

Strips audit-only keys when present, then drops redundant input paths when `smok_params_source` and `mort_params_source` are set (same idea as portable save). Sparse lists serialize as-is (minimal keys only).

Usage

```
shg_write_config_yaml(config, path)
```

Arguments

`config`	Named list (`original_config`, `repro_config`, or any intent list).
`path`	Output file path.

Details

Parameter provenance and table paths are grouped under a `params` mapping when present (`smok_params_source`, `mort_params_source`, `mort_params_type`, and/or `input_data_folder` with per-table filenames). `shg_load_config` and `shg_apply_config` accept nested or flat keys.

For full portable fixed-cohort bundles, `config` should include both parameter sources and complete `repeat`, `race`, `sex`, `cohort_year` (see `shg_save_config`).

Value

path, invisibly.

---

useConfig *Use SHG Configuration*

Description

Configures an existing SHG instance from a configuration object (typically obtained from getConfig()).

Arguments

config	A list containing configuration parameters. Must include config_version. All parameters are validated.

Details

Configure SHG instance from config object

This method validates the config_version and all parameters before setting them. Unknown fields are warned about but allowed for future compatibility. Missing optional fields use defaults. Fields are applied in an order suitable for round-trips from getConfig(): number_of_segments and num_threads are set before rng_strategy (so switching to Mersenne Twister does not message when the saved list already has single-threaded settings), then seeds, then paths and other options. If the list has deprecated run_multi_threaded but no num_threads, it is mapped: FALSE -> num_threads = 1, TRUE -> num_threads = -1. If both are present, num_threads wins. If the list updates local input paths (input_data_folder or any per-table filename) but omits smok_params_source, mort_params_source, and mort_params_type, any previously recorded bundle provenance is cleared for the omitted key(s) so metadata cannot refer to an older zip after retargeting inputs.

Value

No return value. Called for side effects: updates fields on the SHGInterface instance to match config (typically from getConfig).

Examples

```
shg1 <- new(SHGInterface)
shg1$input_data_folder <- system.file("extdata", "2018", package = "SmokingHistoryGenerator")
shg1$rng_strategy <- "RngStream"
shg1$number_of_segments <- 4
config <- shg1$getConfig()
shg2 <- new(SHGInterface)
shg2$useConfig(config)
shg2$rng_strategy
```

Index