Package ‘AMISforInfectiousDiseases’

January 20, 2025

Title Implement the AMIS Algorithm for Infectious Disease Models

Version 0.1.0

Description Implements the Adaptive Multiple Importance Sampling (AMIS) algorithm, as described by Retkute et al. (2021, <doi:10.1214/21-AOAS1486>), to estimate key epidemiological parameters by combining outputs from a geostatistical model of infectious diseases (such as prevalence, incidence, or relative risk) with a disease transmission model. Utilising the resulting posterior distributions, the package enables forward projections at the local level.

Acknowledgements This work was supported by the NTD Modelling Consortium grant INV-030046, funded by the Bill & Melinda Gates Foundation.

Encoding UTF-8

RoxygenNote 7.3.2

Imports Hmisc, mclust, mnormt, Rcpp, weights

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.1)

Suggests knitr, ggplot2, patchwork, rmarkdown, sf, viridis

VignetteBuilder knitr

URL `https://github.com/drsimonspencer/AMISforInfectiousDiseases-dev`

BugReports `https://github.com/drsimonspencer/AMISforInfectiousDiseases-dev/issues`

NeedsCompilation yes

Author Evandro Konzen [aut] (<`https://orcid.org/0000-0002-6275-1681`>),
Renata Retkute [aut] (<`https://orcid.org/0000-0002-3877-6440`>),
Raiha Browning [aut] (<`https://orcid.org/0000-0002-6175-2244`>),
Thilbault Lestang [aut],
Simon Spencer [aut, cre] (<`https://orcid.org/0000-0002-8375-5542`>),
University of Warwick [cph],
Oxford Research Software Engineering [cph]

Maintainer Simon Spencer <`s.e.f.spencer@warwick.ac.uk`>

Repository CRAN

Date/Publication 2025-01-20 16:02:05 UTC

Contents

`amis` *Run the AMIS algorithm to fit a transmission model to a map*

Description

This implements the AMIS algorithm as described in *Retkute et al. (2021)*. Each iteration of the algorithm produces a set of parameters from a proposal distribution (the prior in the first iteration). For each parameter set, a simulation is run from the transmission model. Then, each preceding simulation is weighted at each location according to the distribution of prevalences (or likelihood function) at that location. A Gaussian mixture model is then fitted to the parameter samples with weights averaged over the active locations (ie locations that have yet to reach the effective sample size target). This Gaussian mixture informs the proposal for the next iteration. The algorithm continues until every location has reached the ESS target, or the maximum number of iterations is reached.

Usage

```
amis(
  prevalence_map,
  transmission_model,
  prior,
  amis_params = default_amis_params(),
  seed = NULL,
  output_dir = NULL,
  initial_amis_vals = NULL
)
```

Arguments

`prevalence_map` For a single timepoint, `"prevalence_map"` can be an $L \times M$ matrix or data frame containing samples from a geostatistical model, where L is the number of locations and M the number of samples per location.

If there are multiple timepoints and/or a parametric `likelihood` function is to be used, `"prevalence_map"` must be a list with T elements, one for each timepoint $t = 1, \dots, T$. Each element must itself be a list with the following objects:

`data` An $L \times M$ matrix as above

`likelihood` (optional) A function taking arguments:

- `data`: A vector of length M_l, where M_l is the number of samples from a geostatistical model for location l or the number of likelihood parameters;
- `sim_prev`: A numeric value for a prevalence simulated from the transmission model;
- `log`: Logical indicating if calculations are to be performed on log scale (specified in `"amis_params"`, see below).

The function `likelihood` must return a numeric value representing the (log-)likelihood of observing a simulated prevalence given the data from a particular location.

The location names are inherited from `rownames(prevalence_map)` if `"prevalence_map"` is a matrix, and from `rownames(prevalence_map[[1]]$data)` if `"prevalence_map"` is a list.

If `likelihood` is not specified, then it is assumed that the data consist of samples from a geostatistical model and a nonparametric method is used. The nonparametric method to be used is specified in `"amis_params"` using the options `breaks`, `delta`, or `sigma` (see `"amis_params"`).

`transmission_model`

A function taking arguments:

- `seeds`: a vector of n seeds;
- `params`: an $n \times d$ matrix of parameter vectors;
- `n_tims`: number of time points.

This function must return an $n \times T$ **matrix** of prevalences (it must be a matrix even when $T = 1$). The vector `seeds` will be the vector of indexes of the simulated samples. If `n_samples` new samples are drawn within each iteration of the AMIS algorithm, then the vector `seeds` will be `1:n_samples` at the first iteration, `(n_samples+1):(2*n_samples)` at the second iteration, and so on.

`prior` A list containing the functions `dprior` and `rprior` (density and random number generator, respectively). The two arguments of `dprior` must be:

- a d-length vector of transmission model parameters; and
- a logical `log` to indicate whether to calculate log-density or not.

The only argument of `rprior` must be a single integer n that determines the number of samples to draw. `rprior` must produce an $n \times d$ **matrix** of parameters even when $d = 1$. Parameter names are inherited from the `colnames` of the output of `rprior`.

`amis_params` A list containing control parameters for the AMIS algorithm (`default_amis_params()` returns the default values):

`n_samples` Number of new samples drawn within each AMIS iteration. Default to `500`.

`target_ess` Target effective sample size. Default to `500`.

- `max_iters` Maximum number of AMIS iterations. Default to `12`.
- `boundaries` A vector of length two with the left and right boundaries for prevalences. Default to `c(0,1)`. If, for instance, left boundary is zero and there is no right boundary, set `boundaries = c(0,Inf)`.
- `boundaries_param` If specified, it should be a $d \times 2$ matrix with the lower and upper boundaries for the d transmission model parameters. Default to `NULL`.
- `log` Logical indicating if calculations are to be performed on log scale. Default to `TRUE`.
- `delete_induced_prior` Logical indicating whether the induced prior density is to be deleted in the update of weights. Default to `FALSE`.
- `mixture_samples` Number of samples used to represent the weighted parameters in the mixture fitting.
- `df` Degrees of freedom in the t-distributions, used to yield a heavy tailed proposal. Default to `3`.
- `q` Parameter (between 0 and 1) controlling how the weights are calculated for active locations. Default to `0`. See Details below.
- `delta` Optional smoothing parameter if uniform kernel (default) is used. Default to `0.01`.
- `sigma` Optional smoothing parameter if Gaussian kernel is used. Default to `NULL`.
- `breaks` Optional vector specifying the breaks for the histogram. Default to `NULL`. For finite `boundaries`, the first and last entries of `breaks` must be equal to the left and right boundaries, respectively. For non-finite `boundaries`, ensure that the range of `breaks` includes any possible prevalence value.

Uniform kernel is the default method for the density estimator of the likelihood. If `sigma` is provided, then Gaussian kernel will be used instead. If `breaks` is provided, then histogram-based method will be the nonparametric method being used. Note that if `likelihood` is provided in `prevalence_map`, then a parametric method will be implemented.

`seed`	Optional single value interpreted as an integer. It is the seed for the random number generator for the AMIS algorithm. This is not the same as the `seeds` argument passed to `"transmission_model"`.
`output_dir`	A string specifying the local directory where to save outputs after each iteration of the algorithm. At the end of the string, use the correct path separator for your machine's operating system. If the directory is specified, the outputs will be saved in a file called 'amis_output.rds'. Default to `NULL` (i.e. outputs are not saved in a local directory).
`initial_amis_vals`	Optional list of intermittent outputs from a previous run (where at least one iteration was successful). These outputs can be saved by specifying the directory `"output_dir"`.

Details

The average weight of parameter vectors for the set of active locations at iteration i (A_i) has weights determined by how far the effective sample size for location l $\left(\text{ESS}_l^i\right)$ is from the target $\left(\text{ESS}^R\right)$:

$$\bar{w}_j^i = \frac{\sum_{l \in A_i} \left(\text{ESS}^R - \text{ESS}_l^i\right)^q \hat{w}_{lj}^i}{\sum_{l \in A_i} \left(\text{ESS}^R - \text{ESS}_l^i\right)^q}, \qquad q \in [0,1].$$

If $q = 0$ (default), the simple average of individual weights will be calculated. If $q > 0$, more weight will be assigned to locations with low ESS.

Value

A list of class `amis`. If the algorithm completed I iterations, it simulated a total of $N = I \times$ `n_samples`, and therefore the list returned by `amis()` will contain:

`seeds` An N-length vector with the simulation seeds that were used.

`param` An $N \times d$ matrix with the d-dimensional transmission model parameters simulated by the algorithm.

`simulated_prevalences` An $N \times T$ matrix with the simulated prevalences, where T is the number of timepoints.

`weight_matrix` An $N \times L$, where L is the number of locations.

`likelihoods` A $T \times L \times N$ array with the likelihood of observing a simulated prevalence in each location at each time.

`ess` An L-length vector with the final effective sample size (ESS) for each location.

`prevalence_map` List with the prevalence map supplied by the user.

`locations_with_no_data` Vector indicating which locations have no data at any time point.

`components` A list of the mixture components of all iterations, containing:

- `G`: number of components in each iteration;
- `probs`: the mixture weights;
- `Mean`: the locations of the components;
- `Sigma`: the covariance matrices of the components.

`components_per_iteration` A list with the mixture components at each iteration. This object is used in `plot_mixture_components()`.

`ess_per_iteration` An $L \times I$ matrix with with the ESS for each location after each iteration.

`prior_density` An N-length vector with the density function evaluated at the simulated parameter values.

`amis_params` List supplied by the user.

`evidence` A list containing an estimate of the log model evidence and corresponding log variance of this estimate for both the full likelihood model (product over all locations), and for each location individually.

References

Retkute, R., Touloupou, P., Basanez, M. G., Hollingsworth, T. D., Spencer, S. E. (2021). *Integrating geostatistical maps and infectious disease transmission models using adaptive multiple importance sampling.* The Annals of Applied Statistics, 15(4), 1980-1998. doi:10.1214/21AOAS1486.

Examples

```
# Define simple "transmission" model where prevalence equals first parameter
transmission_model_identity <- function(seeds, parameters, n_tims=1) {
  return(matrix(parameters[,1], ncol=1))
}

# Generate samples for prevalence map with 3 locations given by B(2,1), B(1,1)=Uniform, B(1,2).
set.seed(123)
L <- 3    # Number of locations
M <- 500 # Number of map samples
prevalence_map <- matrix(NA, L, M)
for (l in 1:L) {
  prevalence_map[l,] <- rbeta(M, max(1,l-1), max(1,3-l))
}
rownames(prevalence_map) <- c("Here","There","Somewhere else")

# Define 2D exponential prior
rprior <- function(n) {
  params <- matrix(NA, n, 2)
  colnames(params) <- c("a","b")
  params[,1] <- rexp(n)
  params[,2] <- rexp(n)
  return(params)
}
dprior <- function(x, log=FALSE) {
  if (log) {
    return(sum(dexp(x, log=TRUE)))
  } else {
    return(prod(dexp(x)))
  }
}
prior <- list(rprior=rprior,dprior=dprior)

# Run AMIS with default control parameters
amis_params <- default_amis_params()
output <- amis(prevalence_map, transmission_model_identity, prior, amis_params, seed=1)

print(output)
summary(output)

original_par <- par(no.readonly = TRUE)
par(cex.lab=1.5, cex.main=1.5, mar=c(5,4.5,4,2)+0.1)

par(mfrow=c(1,2))
plot_mixture_components(output, what = "uncertainty", cex=3)
plot_mixture_components(output, what = "density", nlevels=200)

par(mfrow=c(3,3))
plot(output, what = "a", type="hist", locations=1:L, breaks=100)
plot(output, what = "b", type="hist", locations=1:L, breaks=100)
plot(output, what = "prev", type="hist", locations=1:L, time=1, breaks=100)
```

```
par(mar=c(5,7.5,4,2)+0.1)
par(mfrow=c(1,3))
plot(output, what=c("a","b","prev"), type="CI", locations=1:L, ylab=NA,
     cex=3, lwd=3, measure_central="median", display_location_names=TRUE)

calculate_summaries(output, what="prev", locations=1:L, alpha=0.05)

# Generate new samples from the weighted posterior distributions
new_samples <- sample_parameters(output, n_samples = 200, locations = "Here")
head(new_samples)
plot_hist <- function(column_name){
 hist(new_samples[, column_name], xlab=column_name, main=paste("Histogram of", column_name))
}
par(mfrow=c(1,3))
plot_hist("a")
plot_hist("b")
plot_hist("prevalence")

par(original_par)
```

`calculate_summaries` *Calculate summaries of weighted statistics*

Description

Calculate summaries of weighted statistics

Usage

```
calculate_summaries(
  x,
  what = "prev",
  time = 1,
  locations = NULL,
  alpha = 0.05,
  exceedance_prob_threshold = 0.35
)
```

Arguments

`x`	The output from the function `amis()`.
`what`	What statistic should be calculated the summaries from. It must be either `"prev"` or the name of one of the model parameters. Default to `"prev"`.
`time`	Time point. Only used if `"what"` is set to `"prev"`.
`locations`	Integer vector or location names identifying locations where summaries should be calculated for. If not specified, summary statistics of all locations will be provided.

`alpha`	Numeric value between 0 and 1. Calculations are for the `(alpha/2, 1-alpha/2)`% quantiles.
`exceedance_prob_threshold`	Numeric value. Default to `0.35`, i.e. the probability that the statistic of interest (e.g. prevalence) is higher than `0.35`.

Details

For illustrative examples, see `amis()`.

Value

A list with mean, median, and quantiles of the weighted distribution.

`default_amis_params` *Produce list containing the default AMIS parameters*

Description

For description of AMIS parameters, see argument `amis_params` in `amis()`.

Usage

```
default_amis_params()
```

Value

List containing the default AMIS parameters.

`plot.amis` *Plot histogram or credible interval of weighted distributions given a model fitted by `amis()`*

Description

Plot histogram or credible interval of weighted distributions given a model fitted by `amis()`

Usage

```
## S3 method for class 'amis'
plot(
  x,
  what = "prev",
  type = "hist",
  locations = 1,
  time = 1,
  measure_central = "mean",
  order_locations_by = NULL,
  display_location_names = FALSE,
  alpha = 0.05,
  breaks = 500,
  cex = 1,
  lwd = 1,
  xlim = NULL,
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  ...
)
```

Arguments

`x`	The output from the function `amis()`.
`what`	What posterior distribution should be plotted. It can be `"prev"` (default) for plotting prevalences, or one of the parameter names.
`type`	Type of plot. It can be `"hist"` (default) for histogram, or `"CI"` for credible intervals
`locations`	Integer vector or location names identifying locations the plots are made for. Default to `1` (first location).
`time`	Integer index identifying the timepoint. Default to `1`.
`measure_central`	Measure of central tendency for credible interval plots. It can be `"mean"` (default) or `"median"`.
`order_locations_by`	How the credible intervals of multiple locations should be ordered. If `NULL` (default), locations are displayed according to the argument `"locations"`. Otherwise, it must be either `"prev"` or one of the parameter names, and then the locations are ranked by the corresponding measure of central tendency.
`display_location_names`	Logical indicating whether location names are to be shown or not in credible interval plots. Default to `FALSE`.
`alpha`	Numeric value between 0 and 1 indicating the endpoints of the credible intervals, which are evaluated at (alpha/2, 1-alpha/2)% quantiles. Default (`0.05`) will create 95% credible intervals.

`breaks`	Argument passed to `wtd.hist()` for histogram plots. Default to `500`.
`cex`	Argument passed to plots of credible intervals. Default to `1`.
`lwd`	Argument passed to plots of credible intervals. Default to `1`.
`xlim`	The x limits of the plots. For for credible intervals of multiple statistics (i.e. `length(what)>1`), it must be either `NULL` or a list with the x limits for each statistic. Default to `NULL`.
`main`	Title for the plot.
`xlab`	Lable for the x axis.
`ylab`	Lable for the y axis.
`...`	Other graphical parameters passed to `wtd.hist()`.

Details

For illustrative examples, see `amis()`.

Value

A plot.

`plot_mixture_components` *Wrapper function for* `plot.Mclust()`

Description

Wrapper function for `plot.Mclust()`

Usage

```
plot_mixture_components(
  x,
  what = "uncertainty",
  iteration = NULL,
  datapoints = "proposed",
  main = NULL,
  xlim = NULL,
  ylim = NULL,
  ...
)
```

Arguments

`x`	The output from the function `amis()`.
`what`	A string specifying the type of plot requested: `"uncertainty"` A plot of classification uncertainty (default) `"density"` A plot of estimated density `"BIC"` A plot showing BIC values used to choose the number of components
`iteration`	Integer indicating which iteration the plot should be about. If `NULL` (default), the plot will be for the final iteration. See more details in `plot.Mclust()`.
`datapoints`	A string specifying what the datapoints should represent in the plot of classification uncertainty: `"proposed"` datapoints will represent the samples simulated from the mixture. The colours indicate which mixture components the samples were simulated from. `"fitted"` datapoints will show the samples that the mixture model was fitted to, i.e. weighted samples from the previous iteration. The colour of a datapoint indicates the most likely mixture component the sample belongs to.
`main`	Title of the plot. If `NULL`, the default title will be displayed. Set to `NA` for omitting title.
`xlim`	The x limits of the plots. Default to `NULL`
`ylim`	The y limits of the plots. Default to `NULL`.
`...`	Other arguments to match the `plot.Mclust()` function.

Details

For illustrative examples, see `amis()`.

Value

A plot for model-based clustering results.

`print.amis` *Print method for object of class* `amis`

Description

Print method for object of class `amis`

Usage

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

Arguments

`x`	The output from the function `amis()`.
`...`	Other arguments to match the generic `print()` function

Details

For illustrative examples, see `amis()`.

Value

Brief description of data and model specifications used to run `amis()`.

`sample_parameters` *Sample parameters from their weighted distributions given a model fitted by* `amis()`

Description

Sample parameters from their weighted distributions given a model fitted by `amis()`

Usage

```
sample_parameters(x, n_samples = 200, locations = 1)
```

Arguments

`x`	The output from the function `amis()`.
`n_samples`	Number of samples to draw. Default to `200`.
`locations`	Integer identifying the locations. Default to `1`.

Details

For illustrative examples, see `amis()`.

Value

Matrix with parameter values and corresponding prevalences for each location.

`summary.amis` *Summary method for object of class* `amis`

Description

Summary method for object of class `amis`

Usage

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

Arguments

`object`	The output from the function `amis()`.
`...`	Other arguments to match the generic `summary()` function

Details

For illustrative examples, see `amis()`.

Value

Summary statistics of the fitted model.

Index