Package: DRpower (via r-universe)

June 9, 2024

Type Package

Title Study design and analysis for pfhrp2/3 deletion prevalence studies

Version 1.0.2

Description This package can be used in the design and/or analysis stages of Plasmodium falciparum pfhrp2/3 deletion prevalence studies. We assume that the study takes the form of a clustered prevalence survey, meaning the data consists of a numerator (number of deletions found) and denominator (number tested) over multiple clusters. We are interested in estimating the study-level prevalence, i.e. over all clusters, while accounting for the possibility of high intra-cluster correlation. The analysis approach uses a Bayesian random effects model to estimate prevalence and intra-cluster correlation. The approach to power analysis is simulation-based, running the analysis many times on simulated data and estimating empirical power. This method can be used to establish a minimum sample size required to achieve a given target power.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.2.3

BugReports `https://github.com/mrc-ide/DRpower/issues`

Imports dplyr, extraDistr, magrittr, Rcpp, knitrProgressBar, ggplot2, tidyr

Suggests knitr, cowplot, kableExtra, testthat (>= 3.0.0), tidyverse

Config/testthat/edition 3

LinkingTo Rcpp

VignetteBuilder knitr

Depends R (>= 3.5.0)

Repository https://plasmogenepi.r-universe.dev

RemoteUrl https://github.com/mrc-ide/DRpower

RemoteRef HEAD

RemoteSha 250c08a6aa457c5b5a0be099b315fd1523f27970

Contents

`check_DRpower_loaded` *Check that DRpower package has loaded successfully*

Description

Simple function to check that DRpower package has loaded successfully. Prints "DRpower loaded successfully!" if so.

Usage

```
check_DRpower_loaded()
```

`df_sim` *Summary of simulations from the threshold analysis*

Description

This object was produced by running the function `get_power_threshold()` over a wide range of parameter combinations. This data.frame contains the results of these simulations attached to the parameter values used in simulation. The most obvious use of this object is in constructing power curves over a range of sample sizes (see `?power_curve()`).

Usage

```
data(df_sim)
```

Format

A data.frame of 547200 rows and 19 columns. The first 13 columns give parameter combinations that were used in simulating and analysing data. The "reps" column gives the number of times simulation was repeated, and "seed" gives the value of the seed that was used at the start of this loop (to ensure reproducibility). "prev_thresh" gives the prevalence threshold used in hypothesis testing. The final three columns give the estimates power over simulations along with lower and upper 95% CIs calculated using the method of Clopper and Pearson (1934).

Examples

```
data(df_sim)
```

`df_ss` *Minimum sample sizes for the threshold analysis*

Description

This object was produced by finding the point at which `df_sim` crossed the target power threshold of 80% (see details).

Usage

```
data(df_ss)
```

Format

A data.frame of 6840 rows and 15 columns. The first 14 columns give parameter combinations that were used in simulating and analysing data. The final "N_opt" column gives the optimal sample size to achieve a power of 80%.

Details

Minimum sample sizes were calculated as follows:

1. Find the value of N that crosses the threshold, and the value of N preceding it that does not.
2. Do linear interpolation between these two values to get the estimated sample size at the threshold.
3. Deal with special cases of N always being below the target power or always above the target power.
4. Some additional manual wrangling of final results. Ensure that N always decreases with increasing numbers of clusters (this is not always the case due to random variation).

Examples

```
data(df_ss)
```

`DRpower` *The DRpower app for design and analysis of Plasmodium falciparum pfhrp2/3 data*

Description

This package can be used in the design and/or analysis stages of Plasmodium falciparum pfhrp2/3 deletion prevalence studies. We assume that the study takes the form of a clustered prevalence survey, meaning the data consists of a numerator (number tested) and denominator (number of deletions found) over multiple clusters. We are interested in estimating the study-level prevalence, i.e. over all clusters, while accounting for the possibility of high intra-cluster correlation. The analysis approach uses a Bayesian random effects model to estimate prevalence and intra-cluster correlation. The approach to power analysis is simulation-based, running the analysis many times on simulated data and estimating empirical power. This method can be used to establish a minimum sample size required to achieve a given target power.

`get_joint` *Get posterior distribution of both prevalence and the ICC on a grid*

Description

Get posterior distribution of both prevalence and the ICC on a grid. Prevalence is returned in columns, ICC in rows. See also `plot_joint` for how to make a contour plot from this grid.

Usage

```
get_joint(
  n,
  N,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  prev_breaks = seq(0, 1, 0.01),
  ICC_breaks = seq(0, 1, 0.01)
)
```

Arguments

`n, N` the numerator (`n`) and denominator (`N`) per cluster. These are both integer vectors.

`prior_prev_shape1, prior_prev_shape2, prior_ICC_shape1, prior_ICC_shape2`
parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

`prev_breaks, ICC_breaks`
the values at which to evaluate the posterior in both dimensions. Prevalence is returned in columns, ICC in rows.

Examples

```
get_joint(n = c(5, 2, 9), N = c(100, 80, 120))
```

`get_margins` *Margin of error calculations when estimating prevalence from a clustered survey*

Description

Calculate the expected margin of error when estimating prevalence from a clustered survey, or calculate the sample size required to achieve a given target margin of error.

Usage

```
get_margin(N, n_clust, prevalence = 0.2, ICC = 0.05, alpha = 0.05)

get_sample_size_margin(
  MOE,
  n_clust,
  prevalence = 0.2,
  ICC = 0.05,
  alpha = 0.05
)

get_margin_CP(N, n_clust, prevalence = 0.2, ICC = 0.05, alpha = 0.05)

get_sample_size_margin_CP(
  MOE,
  n_clust,
  prevalence = 0.2,
  ICC = 0.05,
  alpha = 0.05,
  N_max = 2000
)
```

Arguments

`N` the number of samples obtained from each cluster, assumed the same over all clusters.

`n_clust` the number of clusters.

`prevalence` the true prevalence of the marker in the population as a proportion between 0 and 1.

`ICC` assumed true intra-cluster correlation (ICC) between 0 and 1.

`alpha` the significance level of the CI.

`MOE` the target margin of error.

`N_max` the largest value of N to consider.

Details

A very common approach when constructing confidence intervals (CIs) from prevalence data is to use the Wald interval:

$$\hat{p} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{N}}$$

where $\hat{p}$ is our estimate of the prevalence, z is the critical value of the normal distribution ($z = 1.96$ for a 95% interval) and N is the sample size. When estimating prevalence from a clustered survey, we need to modify this formula as follows:

$$\hat{p} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{Nc}(1+(n-1)r)}$$

where $\hat{p}$ is the *mean* prevalence over clusters, c is the number of clusters, and r is the intra-cluster correlation (ICC, a value between 0 and 1). The term to the right of the $\pm$ symbol is called the *margin of error* (MOE). We can give this term the name d. The function `get_margin()` returns the values $\hat{p} - d$ and $\hat{p} + d$, i.e. the lower and upper estimates of what our CI will be.

We can also rearrange this formula to get the sample size (N) required to achieve any given MOE:

$$N = \frac{z^2 p(1-p)(1-r)}{cd^2 - z^2 p(1-p)r}$$

The function `get_sample_size_margin()` returns the value of N. Note that in some cases it might not be possible to achieve the specified MOE for any finite sample size due to the ICC introducing too much variation, in which case this formula will return a negative value and the function will return an error.

Although this is a very common approach, it has several weaknesses. First, notice that we sneakily replaced $\hat{p}$ with p when moving to the sample size formula above. This implies that there is no uncertainty in our prevalence estimate, which is not true. Also note that the Wald interval assumes that the sampling distribution of our estimator is Gaussian, which is also not true. The difference between the Gaussian and the true distribution is particularly pronounced when prevalence is at the extremes of the range (near 0% or 100%). Here, the Wald interval can actually include values less than 0 or greater than 1, which are nonsensical.

An arguably better approach is to construct CIs using the method of Clopper and Pearson (1934). This confidence interval guarantees that the false positive rate is *at least* $alpha$, and in this sense is conservative. It can be asymmetric and does not suffer from the problem of allowing values outside the [0,1] range. To make the Clopper-Pearson interval apply to a multi-cluster survey, we can use the idea of effective sample size, N_e:

$$D_{eff} = 1 + (N-1)r$$

$$N_e = \frac{N}{D_{eff}}$$

We then calculate the Clopper-Pearson CI but using N_e in place of N. The function `get_margin_CP()` returns the expected lower and upper CI limits using the Clopper-Pearson interval, and the function `get_sample_size_margin_CP()` returns the corresponding sample size needed to achieve a certain MOE (the maximum of either lower or upper).

A third option is to use the DRpower Bayesian model to estimate the credible interval of prevalence. See `?get_margin_Bayesian()` for how to do this.

Value

the functions `get_margin()` and `get_margin_CP()` return the expected lower and upper CI limits on the prevalence as percentage. Technically this is not the MOE, as that would be the difference between these limits and the assumed prevalence. However, we feel this is a more useful and more intuitive output.

References

Clopper, C.J. and Pearson, E.S., 1934. The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika, 26, 404–413. doi: 10.2307/2331986.

Examples

```
get_margin(N = 60, n_clust = 3, prevalence = 0.2)

get_sample_size_margin(MOE = 0.07, n_clust = 3, prevalence = 0.2, ICC = 0.01)

get_margin_CP(N = 60, n_clust = 3, prevalence = 0.2)

get_sample_size_margin_CP(MOE = 0.14, n_clust = 3, prevalence = 0.2, ICC = 0.01)
```

get_margin_Bayesian *Margin of error calculations using the Bayesian DRpower model when estimating prevalence from a clustered survey*

Description

As well as comparing against a threshold, the function `get_prevalence()` can be used to estimate a Bayesian credible interval (CrI) on the prevalence. This function returns the margin of error (MOE) we can expect via this method, in terms of the expected lower and upper limits of our credible interval (CrI).

Usage

```
get_margin_Bayesian(
  N,
  prevalence = 0.2,
  ICC = 0.05,
  alpha = 0.05,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  CrI_type = "HDI",
  n_intervals = 20,
  round_digits = 2,
  reps = 100,
  use_cpp = TRUE,
  return_full = FALSE,
  silent = FALSE
)
```

Arguments

`N`	vector of the number of samples obtained from each cluster.
`prevalence`	assumed true prevalence of pfhrp2/3 deletions as a proportion between 0 and 1.
`ICC`	assumed true intra-cluster correlation (ICC) as a value between 0 and 1.

`alpha` the significance level of the credible interval - for example, use `alpha = 0.05` for a 95% interval. See also `CrI_type` argument for how this is calculated.

`prior_prev_shape1, prior_prev_shape2, prior_ICC_shape1, prior_ICC_shape2`

parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

`CrI_type` which method to use when computing credible intervals. Options are "ETI" (equal-tailed interval) or "HDI" (high-density interval). The ETI searches a distance `alpha/2` from either side of the [0,1] interval. The HDI method returns the narrowest interval that subtends a proportion `1-alpha` of the distribution. The HDI method is used by default as it guarantees that the MAP estimate is within the credible interval, which is not always the case for the ETI.

`n_intervals` the number of intervals used in the adaptive quadrature method. Increasing this value gives a more accurate representation of the true posterior, but comes at the cost of reduced speed.

`round_digits` the number of digits after the decimal point that are used when reporting estimates. This is to simplify results and to avoid giving the false impression of extreme precision.

`reps` number of times to repeat simulation per parameter combination.

`use_cpp` if `TRUE` (the default) then use an Rcpp implementation of the adaptive quadrature approach that is much faster than the base R method.

`return_full` if `TRUE` then return the complete distribution of lower and upper CrI limits in a data.frame. If `FALSE` (the default) return a summary including the mean and 95% CI of these limits.

`silent` if `TRUE` then suppress all console output.

Details

Estimates MOE using the following approach:

1. Simulate data via the function `rbbinom_reparam()` using known values, e.g. a known "true" prevalence and intra-cluster correlation.
2. Analyse data using `get_prevalence()`. Determine the upper and lower limits of the credible interval.
3. Repeat steps 1-2 `reps` times to obtain the distribution of upper and lower limits. Return the mean of this distribution along with upper and lower 95% CIs. To be completely clear, we are producing a 95% CI on the limits of a CrI, which can be confusing! See *Value* for a clear explanation of how to interpret the output.

Note that we have not implemented a function to return the sample size needed to achieve a given MOE under the Bayesian model, as this would require repeated simulation over different values of `N` which is computationally costly. The appropriate value can be established manually if needed by running `get_margin_Bayesian()` for different sample sizes.

Value

If `return_full` = `FALSE` (the default) returns an estimate of the lower and upper CrI limit in the form of a data.frame. The first row gives the lower limit, the second row gives the upper limit, both as percentages. The first column gives the point estimate, the subsequent columns give the 95% CI on this estimate. If `return_full` = `TRUE` then returns a complete data.frame of all lower and upper CI realisations over simulations.

Examples

```
get_margin_Bayesian(N = c(120, 90, 150), prevalence = 0.15, ICC = 0.01 , reps = 1e2)
```

`get_posterior` *Estimate prevalence and intra-cluster correlation from raw counts*

Description

Takes raw counts of the number of positive samples per cluster (numerator) and the number of tested samples per cluster (denominator) and returns posterior estimates of the prevalence and intra-cluster correlation coefficient (ICC).

Usage

```
get_prevalence(
  n,
  N,
  alpha = 0.05,
  prev_thresh = 0.05,
  ICC = NULL,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  MAP_on = TRUE,
  post_mean_on = FALSE,
  post_median_on = FALSE,
  post_CrI_on = TRUE,
  post_thresh_on = TRUE,
  post_full_on = FALSE,
  post_full_breaks = seq(0, 1, l = 1001),
  CrI_type = "HDI",
  n_intervals = 20,
  round_digits = 2,
  use_cpp = TRUE,
  silent = FALSE
)
```

```
get_ICC(
  n,
  N,
  alpha = 0.05,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  MAP_on = TRUE,
  post_mean_on = FALSE,
  post_median_on = FALSE,
  post_CrI_on = TRUE,
  post_full_on = FALSE,
  post_full_breaks = seq(0, 1, l = 1001),
  CrI_type = "HDI",
  n_intervals = 20,
  round_digits = 4,
  use_cpp = TRUE
)
```

Arguments

n, N
: the numerator (n) and denominator (N) per cluster. These are both integer vectors.

alpha
: the significance level of the credible interval - for example, use alpha = 0.05 for a 95% interval. See also CrI_type argument for how this is calculated.

prev_thresh
: the prevalence threshold that we are comparing against. Can be a vector, in which case the return object contains one value for each input.

ICC
: normally this should be set to NULL (the default), in which case the ICC is estimated from the data. However, a fixed value can be entered here, in which case this overrides the use of the prior distribution as specified by prior_ICC_shape1 and prior_ICC_shape2.

prior_prev_shape1, prior_prev_shape2, prior_ICC_shape1, prior_ICC_shape2
: parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

MAP_on, post_mean_on, post_median_on, post_CrI_on, post_thresh_on, post_full_on
: a series of boolean values specifying which outputs to produce. The options are:

- MAP_on: if TRUE then return the maximum *a posteriori*.
- post_mean_on: if TRUE then return the posterior mean.
- post_median_on: if TRUE then return the posterior median.
- post_CrI_on: if TRUE then return the posterior credible interval at significance level alpha. See CrI_type argument for how this is calculated.

- post_thresh_on: if TRUE then return the posterior probability of being above the threshold(s) specified by prev_thresh.
- post_full_on: if TRUE then return the full posterior distribution, produced using the adaptive quadrature approach, at breaks specified by post_full_breaks.

post_full_breaks
a vector of breaks at which to evaluate the full posterior distribution (only if post_full_on = TRUE). Defaults to 0.1% intervals from 0% to 100%.

CrI_type
which method to use when computing credible intervals. Options are "ETI" (equal-tailed interval) or "HDI" (high-density interval). The ETI searches a distance alpha/2 from either side of the [0,1] interval. The HDI method returns the narrowest interval that subtends a proportion 1-alpha of the distribution. The HDI method is used by default as it guarantees that the MAP estimate is within the credible interval, which is not always the case for the ETI.

n_intervals
the number of intervals used in the adaptive quadrature method. Increasing this value gives a more accurate representation of the true posterior, but comes at the cost of reduced speed.

round_digits
the number of digits after the decimal point that are used when reporting estimates. This is to simplify results and to avoid giving the false impression of extreme precision.

use_cpp
if TRUE (the default) then use an Rcpp implementation of the adaptive quadrature approach that is much faster than the base R method.

silent
if TRUE then suppress all console output.

Details

There are two unknown quantities in the DRpower model: the prevalence and the intra-cluster correlation (ICC). These functions integrate over a prior on one quantity to arrive at the marginal posterior distribution of the other. Possible outputs include the maximum *a posteriori* (MAP) estimate, the posterior mean, posterior median, credible interval (CrI), probability of being above a set threshold, and the full posterior distribution. For speed, distributions are approximated using an adaptive quadrature approach in which the full distribution is split into intervals and each intervals is approximated using Simpson's rule. The number of intervals used in quadrature can be increased for more accurate results at the cost of slower speed.

Examples

```
# basic example of estimating prevalence and
# ICC from observed counts
sample_size <- c(80, 110, 120)
deletions <- c(3, 5, 6)

get_prevalence(n = deletions, N = sample_size)
get_ICC(n = deletions, N = sample_size)
```

`get_power_presence` *Calculate power when testing for presence of deletions*

Description

Calculates power directly for the case of a clustered prevalence survey where the aim is to detect the presence of *any* deletions over all clusters. This design can be useful as a pilot study to identify priority regions where high deletion prevalence is likely. Note that we need to take account of intra-cluster correlation here, as a high ICC will make it more likely that we see zero deletions even when the prevalence is non-zero.

Usage

```
get_power_presence(N, prevalence = 0.01, ICC = 0.05)
```

Arguments

`N`	vector of the number of samples obtained from each cluster.
`prevalence`	assumed true prevalence of pfhrp2/3 deletions as a proportion between 0 and 1.
`ICC`	assumed true intra-cluster correlation (ICC) between 0 and 1.

Examples

```
get_power_presence(N = c(120, 90, 150), prevalence = 0.01, ICC = 0.1)
```

`get_power_threshold` *Estimate power when testing prevalence against a threshold*

Description

Estimates power when conducting a clustered prevalence survey and comparing against a set threshold. Estimates power empirically via repeated simulation. Returns an estimate of the power, along with lower and upper 95% confidence interval of this estimate.

Usage

```
get_power_threshold(
  N,
  prevalence = 0.1,
  ICC = 0.05,
  prev_thresh = 0.05,
  rejection_threshold = 0.95,
  ICC_infer = NULL,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  n_intervals = 20,
  round_digits = 2,
  reps = 100,
  use_cpp = TRUE,
  silent = FALSE
)
```

Arguments

`N` vector of the number of samples obtained from each cluster.

`prevalence` assumed true prevalence of pfhrp2/3 deletions as a proportion between 0 and 1. If a vector of two values is given here then prevalence is drawn uniformly from between these limits independently for each simulation. This allows power to be calculated for a composite hypothesis.

`ICC` assumed true intra-cluster correlation (ICC) as a value between 0 and 1.

`prev_thresh` the threshold prevalence that we are testing against (5% by default).

`rejection_threshold`
the posterior probability of being above the prevalence threshold needs to be greater than `rejection_threshold` in order to reject the null hypothesis.

`ICC_infer` the value of the ICC assumed in the inference step. If we plan on estimating the ICC from our data, i.e. running `get_prevalence(ICC = NULL)` (the default), then we should also set `ICC=NULL` here (the default). However, if we plan on running `get_prevalence()` with ICC set to a known value then we should insert this value here as `ICC_infer`.

`prior_prev_shape1`, `prior_prev_shape2`, `prior_ICC_shape1`, `prior_ICC_shape2`
parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

`n_intervals` the number of intervals used in the adaptive quadrature method. Increasing this value gives a more accurate representation of the true posterior, but comes at the cost of reduced speed.

`round_digits` the number of digits after the decimal point that are used when reporting estimates. This is to simplify results and to avoid giving the false impression of extreme precision.

`reps` number of times to repeat simulation per parameter combination.

`use_cpp` if `TRUE` (the default) then use an Rcpp implementation of the adaptive quadrature approach that is much faster than the base R method.

`silent` if `TRUE` then suppress all console output.

Details

Estimates power using the following approach:

1. Simulate data via the function `rbbinom_reparam()` using known values (e.g. a known "true" prevalence and intra-cluster correlation).
2. Analyse data using `get_prevalence()` to determine the probability of being above `prev_thresh`.
3. If this probability is above `rejection_threshold` then reject the null hypothesis. Encode this as a single correct conclusion.
4. Repeat steps 1-3 many times. Count the number of simulations for which the correct conclusion is reached, and divide by the total number of simulations. This gives an estimate of empirical power, along with upper and lower 95% binomial CIs via the method of Clopper and Pearson (1934).

Note that this function can be run even when `prevalence` is less than `prev_thresh`, although in this case what is returned is not the power. Power is defined as the probability of *correctly* rejecting the null hypothesis, whereas here we would be incorrectly rejecting the null. Therefore, what we obtain in this case is an estimate of the false positive rate.

References

Clopper, C.J. and Pearson, E.S., 1934. The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika, 26, 404–413. doi: 10.2307/2331986.

Examples

```
get_power_threshold(N = c(120, 90, 150), prevalence = 0.15, ICC = 0.1 , reps = 1e2)
```

---

get_sample_size_presence

Get minimum sample size when testing for presence of deletions

---

Description

Calculates the minimum sample size required per cluster to achieve a certain power for the case of a clustered prevalence survey where the aim is to detect the presence of *any* deletions over all clusters (see `?get_power_presence()`). Assumes the same sample size per cluster.

Usage

```
get_sample_size_presence(
  n_clust,
  target_power = 0.8,
  prevalence = 0.01,
  ICC = 0.05,
  N_max = 2000
)
```

Arguments

`n_clust`	the number of clusters.
`target_power`	the power we are aiming to achieve.
`prevalence`	assumed true prevalence of pfhrp2/3 deletions as a proportion between 0 and 1.
`ICC`	assumed true intra-cluster correlation (ICC) between 0 and 1.
`N_max`	the maximum allowed sample size. Sample sizes are only explored up to this value, after which point an error is returned.

Examples

```
get_sample_size_presence(n_clust = 5, prevalence = 0.01, ICC = 0.1)
```

`get_sample_size_table` *Get pre-computed sample size tables*

Description

Produce a sample size table giving the minimum sample size per cluster for given values of the ICC and the prevalence threshold against which we are comparing.

Usage

```
get_sample_size_table(
  prevalence = seq(0, 0.2, 0.01),
  ICC = 0.05,
  prev_thresh = 0.05
)
```

Arguments

`prevalence`	the assumed true prevalence of pfhrp2/3 deletions in the domain. Allowed values are anything in `seq(0, 0.2, 0.01)`, including vectors of values.
`ICC`	the assumed intra-cluster correlation. Allowed values are" {0, 0.01, 0.02, 0.05, 0.1, 0.2}.
`prev_thresh`	the prevalence threshold against which we are comparing. Allowed values are: {0.05, 0.08, 0.1}.

Details

The function `get_power_threshold()` was run over a large range of parameter combinations and results were stored within the `df_sim` object (see `?df_sim`). These simulations were then used to produce minimum sample size estimates by linear interpolation that were stored within the `df_ss` object (see `?df_ss`). This function provides a simple way of querying the `df_ss` object for given parameter values.

Examples

```
get_sample_size_table()
```

`historical_data` *Data from historical pfhrp2 studies that passed filters for inclusion into an ICC analysis.*

Description

A data.frame of sites that were used to estimate the ICC based on previously published data. These sites passed strict inclusion criteria to ensure they are maximally informative (see details).

Usage

```
data(historical_data)
```

Format

A data.frame of 30 rows and 11 columns. Each row gives a different site that made it through filtering steps in the ICC analysis from historical data. Coluns give geographic properties, sampling times, the number of samples tested and positive for pfhrp2 deletions, and the citation from which the data originates.

Details

The raw dataset of historical pfhrp2/3 studies was downloaded from the WHO malaria threats map on 27 Nov 2023. This spreadsheet can be found in this package in the R_ignore/data folder (see the Github repos) under the name "MTM_PFHRP23_GENE_DELETIONS_20231127_edited.xlss". Note that this spreadsheet has the term "_edited" added to the name because two extra columns were added to the original data: "discard" and "discard_reason". These columns specify certain rows that should be discarded in the original data due to data entry mistakes. The following steps were then taken. All scripts to perform these steps can be found in the same R_ignore folder:

1. Rows were dropped that were identified to discard based on problems in the original data.
2. Filtered to Africa, Asia or South America.
3. Filtered to Symptomatic patients.
4. Filtered to convenience surveys or cross-sectional prospective surveys only.
5. Combined counts (tested and positive) of studies conducted in the same exact location (based on lat/lon) in the same year and from the same source publication. These are considered a single site.
6. Filtered to have 10 or more samples per site. Avoids very small sample sizes which would have very little information from the data and therefore would be driven by our prior assumptions.
7. All sites were mapped to ADMIN1 level by comparing the lat/lon coordinates against a shapefile from GADM version 4.1.0, first administrative unit.

8. Results were combined with studies that contain additional information not reflected in the WHO malaria threats map data. For example, some studies have site-level resolution despite not being apparent in the original data download. These additional studies can be found in the R_ignore/data folder under the name "additional_data.csv".
9. Filter to ADMIN1 regions that contain at least 3 distinct sites within the same year and from the same source publication.

This final filtered dataset is what is available here.

Examples

```
data(historical_data)
```

plot_joint *Contour plot of joint posterior distribution of prevalence and ICC*

Description

Runs get_joint() to obtain the joint posterior distribution of the prevalence and the ICC. Creates a ggplot contour plot object from this result.

Usage

```
plot_joint(
  n,
  N,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  prev_breaks = seq(0, 1, 0.01),
  ICC_breaks = seq(0, 1, 0.01),
  n_bins = 5
)
```

Arguments

n, N
: the numerator (n) and denominator (N) per cluster. These are both integer vectors.

prior_prev_shape1, prior_prev_shape2, prior_ICC_shape1, prior_ICC_shape2
: parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

`prev_breaks`, `ICC_breaks`
: the values at which to evaluate the posterior in both dimensions. Prevalence is returned in columns, ICC in rows.

`n_bins`
: the number of equally spaced breaks in the contour plot. For example, 5 bins creates 4 contour lines at 20%, 40%, 60% and 80% of the maximum value.

Examples

```
plot_joint(n = c(5, 2, 9), N = c(100, 80, 120))
```

plot_posterior *Plot posterior distribution of prevalence and ICC*

Description

These two functions run `get_prevalence()` and `get_ICC()` respectively to obtain the full posterior distribution of the parameter of interest. Then they plot the posterior density along with some useful visualisations including the 95

Usage

```
plot_prevalence(
  n,
  N,
  prev_range = c(0, 1),
  alpha = 0.05,
  prev_thresh = 0.05,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  CrI_type = "HDI",
  n_intervals = 20,
  use_cpp = TRUE
)

plot_ICC(
  n,
  N,
  ICC_range = c(0, 1),
  alpha = 0.05,
  prev_thresh = 0.05,
  prior_prev_shape1 = 1,
  prior_prev_shape2 = 1,
  prior_ICC_shape1 = 1,
  prior_ICC_shape2 = 9,
  CrI_type = "HDI",
  n_intervals = 20,
  use_cpp = TRUE
)
```

Arguments

n, N — the numerator (n) and denominator (N) per cluster. These are both integer vectors.

prev_range — the range of prevalence values explored. Vector of two values giving lower and upper limits, defined between 0 and 1.

alpha — the significance level of the credible interval - for example, use alpha = 0.05 for a 95% interval. See also CrI_type argument for how this is calculated.

prev_thresh — the prevalence threshold that we are testing against (single value only, proportion between 0 and 1).

prior_prev_shape1, prior_prev_shape2, prior_ICC_shape1, prior_ICC_shape2 — parameters that dictate the shape of the Beta priors on prevalence and the ICC. See the Wikipedia page on the Beta distribution for more detail. The default values of these parameters were chosen based on an analysis of historical pfhrp2/3 studies, although this does not guarantee that they will be suitable in all settings.

CrI_type — which method to use when computing credible intervals. Options are "ETI" (equal-tailed interval) or "HDI" (high-density interval). The ETI searches a distance alpha/2 from either side of the [0,1] interval. The HDI method returns the narrowest interval that subtends a proportion 1-alpha of the distribution. The HDI method is used by default as it guarantees that the MAP estimate is within the credible interval, which is not always the case for the ETI.

n_intervals — the number of intervals used in the adaptive quadrature method. Increasing this value gives a more accurate representation of the true posterior, but comes at the cost of reduced speed.

use_cpp — if TRUE (the default) then use an Rcpp implementation of the adaptive quadrature approach that is much faster than the base R method.

ICC_range — the range of ICC values explored. Vector of two values giving lower and upper limits, defined between 0 and 1.

Examples

```
plot_prevalence(n = c(5, 2, 9), N = c(100, 80, 120))

plot_ICC(n = c(5, 2, 9), N = c(100, 80, 120))
```

plot_power *Plot a power curve using pre-computed values*

Description

Runs get_joint() to obtain the joint posterior distribution of the prevalence and the ICC. Creates a ggplot contour plot object from this result.

Usage

```
plot_power(
  n_clust = 5,
  prevalence = 0.1,
  ICC = 0.05,
  prev_thresh = 0.05,
  N_min = 1,
  N_max = 2000
)
```

Arguments

n_clust	the number of clusters. Allowed values are anything in 2:20, including vectors of values.
prevalence	the assumed true prevalence of pfhrp2/3 deletions in the domain. Allowed values are anything in seq(0, 0.2, 0.01), including vectors of values.
ICC	the assumed intra-cluster correlation. Allowed values are" {0, 0.01, 0.02, 0.05, 0.1, 0.2}.
prev_thresh	the prevalence threshold against which we are comparing. Allowed values are: {0.05, 0.08, 0.1}.
N_min, N_max	plotting limits on the x-axis.

Examples

```
plot_power()
```

Index