Package: coiaf (via r-universe)

June 4, 2024

Type Package

Title Complexity of Infection Estimation with Allele Frequencies

Version 0.1.2

Maintainer Aris Paschalidis <aris_paschalidis@brown.edu>

Description Provides a direct method for estimating complexity of infection using easily calculated measures from sequence read depth data.

License MIT + file LICENSE

URL https://bailey-lab.github.io/coiaf/, https://github.com/bailey-lab/coiaf

BugReports https://github.com/bailey-lab/coiaf/issues

Depends R (>= 3.5.0)

Imports boot, broom, cli, dplyr, ggplot2, glue, Hmisc, lifecycle, magrittr, mathjaxr, purrr, rlang (>= 0.4.11), stringr, tibble, tidyr, utils

Suggests bench, here, knitr, patchwork, pbapply, rmarkdown, roxygen2, spelling, testthat (>= 3.0.0), vdiffr, withr

VignetteBuilder knitr

RdMacros mathjaxr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.2

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

RemoteUrl https://github.com/bailey-lab/coiaf

RemoteRef HEAD

RemoteSha 3df5509ceb4dd095ff5910a9d8a18a5e3551e149

Contents

`bootstrap_ci` *Generate bootstrapped CI*

Description

Generate bootstrapped confidence interval for COI estimates.

Usage

```
bootstrap_ci(
  data,
  max_coi = 25,
  seq_error = 0.01,
  coi_method = c("variant", "frequency"),
  solution_method = c("discrete", "continuous"),
  use_bins = FALSE,
  bin_size = 20,
  replicates = 100,
  parallel = FALSE,
  ncpus = 8
)

## Default S3 method:
bootstrap_ci(
  data,
```

```
  max_coi = 25,
  seq_error = 0.01,
  coi_method = c("variant", "frequency"),
  solution_method = c("discrete", "continuous"),
  use_bins = FALSE,
  bin_size = 20,
  replicates = 100,
  parallel = FALSE,
  ncpus = 8
)

## S3 method for class 'sim'
bootstrap_ci(
  data,
  max_coi = 25,
  seq_error = 0.01,
  coi_method = c("variant", "frequency"),
  solution_method = c("discrete", "continuous"),
  use_bins = FALSE,
  bin_size = 20,
  replicates = 100,
  parallel = FALSE,
  ncpus = 8
)
```

Arguments

`data`	The data for which the COI will be computed.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`coi_method`	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
`solution_method`	Whether to estimate discrete or continuous COIs.
`use_bins`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.
`bin_size`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.
`replicates`	The number of bootstrap replicates.
`parallel`	Whether to parallelize the confidence interval calculation. Note that parallelization only works on non-Windows machines.
`ncpus`	The number of processes to be used in parallel operation.

Value

A `tibble()` with columns:

coi The mean COI.

bias Bias of the statistic.

std.error The standard error of the statistic.

conf.low The lower 95% confidence interval.

conf.high The upper 95% confidence interval.

See Also

`boot::boot()`, `boot::boot.ci()`, `broom::tidy.boot()`

Examples

```
sim_data <- sim_biallelic(coi = 5, plmaf = runif(100, 0, 0.5))
bootstrap_ci(sim_data, solution_method = "continuous")
```

`compute_coi` *Predict the COI*

Description

Predict the COI of the sample.

Usage

```
compute_coi(
  data,
  data_type,
  max_coi = 25,
  seq_error = 0.01,
  bin_size = 20,
  comparison = "overall",
  distance = "squared",
  coi_method = "variant",
  use_bins = FALSE
)
```

Arguments

`data`	The data for which the COI will be computed.
`data_type`	The type of the data to be analyzed. One of `"sim"` or `"real"`.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`bin_size`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.

`comparison` **[Deprecated]** This argument is no longer supported; this function will compare the theoretical curve and sample curve for all PLMAFs.

`distance` **[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.

`coi_method` The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".

`use_bins` **[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.

Details

Compare the within sample allele frequency (WSMAF) and the population level allele frequency (PLMAF) of the sample to what a theoretical WSMAF and PLMAF should look like. By examining the sample's WSMAF and PLMAF to the theoretical WSMAF and PLMAF, an estimation can be made about what the COI of the sample is. We refer to the sample's WSMAF vs PLMAF as the "sample curve" and refer to the theoretical WSMAF vs PLMAF as the "theoretical curve." To determine the predicted COI value, one of three different methods can be selected:

`end` Determines the distance between the theoretical and sample curve at a PLMAF of 0.5. The COI is whichever theoretical COI curve has the smallest distance to the simulated data.

`ideal` Determines the distance between the theoretical and sample curve at the ideal PLMAF. The ideal PLMAF is calculated by looking at the change between the COI of i and the COI of $i - 1$ and finding the PLMAF for which this distance is maximized. The COI is whichever theoretical COI curve has the smallest distance to the simulated data at the ideal PLMAF.

`overall` Determines the distance between the theoretical and simulated curve for all PLMAFs. Computes the distance between the theoretical curves and the simulated curve. The COI is whichever theoretical curve has the smallest distance to the simulated curve. There is an option to choose one of several distance metrics:

- `abs_sum`: Absolute value of sum of difference.
- `sum_abs`: Sum of absolute difference.
- `squared`: Sum of squared difference.

Value

A list of the following:

- `coi`: The predicted COI of the sample.
- `probability`: A probability density function representing the probability of each COI.

compute_coi_regression *Compute COI based on residuals of all loci against theoretical curves*

Description

Compute COI based on residuals of all loci against theoretical curves

Usage

```
compute_coi_regression(
  data,
  data_type,
  max_coi = 25,
  seq_error = 0.01,
  distance = "squared",
  coi_method = "variant",
  seq_error_bin_size = 20
)
```

Arguments

data	The data for which the COI will be computed.
data_type	The type of the data to be analyzed. One of "sim" or "real".
max_coi	A number indicating the maximum COI to compare the simulated data to.
seq_error	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
distance	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
coi_method	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
seq_error_bin_size	Number of loci in smallest bin for estimating sequence error

Value

A list of the following:

- coi: The predicted COI of the sample.
- probability: A probability density function representing the probability of each COI.

`cont_sensitivity` *Continuous sensitivity analysis*

Description

Runs several iterations of a full COI sensitivity analysis with varying parameters.

Usage

```
cont_sensitivity(
  repetitions = 10,
  coi = 3,
  max_coi = 25,
  plmaf = runif(1000, 0, 0.5),
  coverage = 200,
  alpha = 1,
  overdispersion = 0,
  relatedness = 0,
  epsilon = 0,
  seq_error = 0.01,
  bin_size = 20,
  comparison = "overall",
  distance = "squared",
  coi_method = "variant",
  use_bins = FALSE
)
```

Arguments

`repetitions`	The number of times each sample will be run.
`coi`	Complexity of infection.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`plmaf`	Vector of population-level minor allele frequencies at each locus.
`coverage`	Coverage at each locus. If a single value is supplied then the same coverage is applied over all loci.
`alpha`	Shape parameter of the symmetric Dirichlet prior on strain proportions.
`overdispersion`	The extent to which counts are over-dispersed relative to the binomial distribution. Counts are Beta-Binomially distributed, with the beta distribution having shape parameters $\frac{p}{overdispersion}$ and $\frac{1-p}{overdispersion}$.
`relatedness`	The probability that a strain in mixed infections is related to another. The implementation is similar to relatedness as defined in THE REAL McCOIL simulations (doi:10.1371/journal.pcbi.1005348): "... simulated relatedness (r) among lineages within the same host by sampling alleles either from an existing lineage within the same host (with probability r) or from the population (with probability (1-r))."

epsilon	The probability of a single read being miscalled as the other allele. This error is applied in both directions.
seq_error	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
bin_size	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing plaf.
comparison	**[Deprecated]** This argument is no longer supported; this function will compare the theoretical curve and sample curve for all PLMAFs.
distance	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
coi_method	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
use_bins	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing plaf.

Value

A list of the following:

- predicted_coi: A dataframe of the predicted COIs. COIs are predicted using compute_coi(). Each column represents a separate set of parameters. Each row represents a predicted COI. Predictions are done many times, depending on the value of repetitions.
- probability:A list of matrices containing the probability that our model predicted each COI value. Each row contains the probability for a different run. The first row contains the average probabilities over all the runs.
- param_grid: The parameter grid. The parameter grid is all possible combinations of the parameters inputted. Each row represents a unique combination.
- boot_error: A dataframe containing information about the error of the algorithm. The first column indicates the COI that was fed into the simulation. The other columns indicate the mean absolute error (mae), the lower and upper bounds of the 95% confidence interval and the bias.

---

disc_sensitivity *Discrete Sensitivity analysis*

Description

Runs several iterations of a full COI sensitivity analysis with varying parameters.

Usage

```
disc_sensitivity(
  repetitions = 10,
  coi = 3,
  max_coi = 25,
  plmaf = runif(1000, 0, 0.5),
  coverage = 200,
  alpha = 1,
  overdispersion = 0,
  relatedness = 0,
  epsilon = 0,
  seq_error = 0.01,
  bin_size = 20,
  comparison = "overall",
  distance = "squared",
  coi_method = "variant",
  use_bins = FALSE
)
```

Arguments

`repetitions`	The number of times each sample will be run.
`coi`	Complexity of infection.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`plmaf`	Vector of population-level minor allele frequencies at each locus.
`coverage`	Coverage at each locus. If a single value is supplied then the same coverage is applied over all loci.
`alpha`	Shape parameter of the symmetric Dirichlet prior on strain proportions.
`overdispersion`	The extent to which counts are over-dispersed relative to the binomial distribution. Counts are Beta-Binomially distributed, with the beta distribution having shape parameters $\frac{p}{overdispersion}$ and $\frac{1-p}{overdispersion}$.
`relatedness`	The probability that a strain in mixed infections is related to another. The implementation is similar to relatedness as defined in THE REAL McCOIL simulations (doi:10.1371/journal.pcbi.1005348): "... simulated relatedness (r) among lineages within the same host by sampling alleles either from an existing lineage within the same host (with probability r) or from the population (with probability (1-r))."
`epsilon`	The probability of a single read being miscalled as the other allele. This error is applied in both directions.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`bin_size`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.
`comparison`	**[Deprecated]** This argument is no longer supported; this function will compare the theoretical curve and sample curve for all PLMAFs.

distance	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
coi_method	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
use_bins	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing plaf.

Value

A list of the following:

- predicted_coi: A dataframe of the predicted COIs. COIs are predicted using compute_coi(). Each column represents a separate set of parameters. Each row represents a predicted COI. Predictions are done many times, depending on the value of repetitions.
- probability:A list of matrices containing the probability that our model predicted each COI value. Each row contains the probability for a different run. The first row contains the average probabilities over all the runs.
- param_grid: The parameter grid. The parameter grid is all possible combinations of the parameters inputted. Each row represents a unique combination.
- boot_error: A dataframe containing information about the error of the algorithm. The first column indicates the COI that was fed into the simulation. The other columns indicate the mean absolute error (mae), the lower and upper bounds of the 95% confidence interval and the bias.

---

error_plot *Error plot*

Description

Creates a plot showing the error of the sensitivity analysis.

Usage

```
error_plot(
  data,
  fill = "coi",
  fill_levels = NULL,
  title = NULL,
  legend_title = fill,
  legend.position = "right",
  second_fill = NULL
)
```

Arguments

`data`	The data to be plotted.
`fill`	The variable the data will be separated by.
`fill_levels`	The levels for the fill variable.
`title`	The title of the plot. Default to `NULL`.
`legend_title`	The text for the legend. Default to `NULL`.
`legend.position`	The position of the legend. One of `"none"`, `"left"`, `"right"`, `"bottom"`, `"top"`.
`second_fill`	Indicates if there will be a second fill variable and what it will be.

Details

Plots are created using `ggplot2::geom_col()`, which creates a simple bar plot. The mean absolute error is plotted in various colors, according to what parameter is being tested. In addition the 95% confidence interval is shown as black vertical lines.

See Also

`ggplot2::geom_col()` for more information on bar plots and the ggplot2 website.

Other plotting functions: `sensitivity_plot()`, `world_map()`

`example_real_data` *Example real data*

Description

A small example dataset that contains within-sample allele frequencies (WSAFs) from a sample of individuals.

Format

A matrix of data. The rows of the matrix indicate the sample name and the columns of the matrix indicate the WSAF at each locus.

Source

ftp://ngs.sanger.ac.uk/production/malaria/pfcommunityproject/Pf6/

`likelihood` *Likelihood of a COI*

Description

A function to generate the likelihood of a specific COI value.

Usage

```
likelihood(coi, processed_data, distance = "squared", coi_method = "variant")
```

Arguments

`coi`	The COI for which the likelihood will be generated.
`processed_data`	The processed COI data. This is the output of `process_sim()` or `process_real()`.
`distance`	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
`coi_method`	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".

Details

The likelihood can be thought of the distance between two curves: the "real" COI curve, generated from the inputted data, and the "simulated" COI curve, which depends on the COI value specified. There are three different methods implemented to compute the distance between two curves:

- `abs_sum`: Absolute value of sum of difference.
- `sum_abs`: Sum of absolute difference.
- `squared`: Sum of squared difference.

Value

The likelihood for a specific COI value.

See Also

Other optimization functions: `optimize_coi_regression()`, `optimize_coi()`

`optimize_coi` *Optimize the COI*

Description

A function to compute the COI of inputted data.

Usage

```
optimize_coi(
  data,
  data_type,
  max_coi = 25,
  seq_error = 0.01,
  bin_size = 20,
  distance = "squared",
  coi_method = "variant",
  use_bins = FALSE
)
```

Arguments

`data`	The data for which the COI will be computed.
`data_type`	The type of the data to be analyzed. One of `"sim"` or `"real"`.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`bin_size`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.
`distance`	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
`coi_method`	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
`use_bins`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.

Details

The function utilizes `stats::optim()`. In particular, the function utilizes a quasi-Newton method to compute gradients and build a picture of the surface to be optimized. The function uses a likelihood function as defined by `likelihood()`.

Value

The predicted COI value.

See Also

stats::optim() for the complete documentation on the optimization function.

Other optimization functions: likelihood(), optimize_coi_regression()

`optimize_coi_regression` *Compute COI based on all points fitted to best fitting curve for COI*

Description

Compute COI based on all points fitted to best fitting curve for COI

Usage

```
optimize_coi_regression(
  data,
  data_type,
  max_coi = 25,
  seq_error = 0.01,
  distance = "squared",
  coi_method = "variant",
  seq_error_bin_size = 20
)
```

Arguments

`data`	The data for which the COI will be computed.
`data_type`	The type of the data to be analyzed. One of `"sim"` or `"real"`.
`max_coi`	A number indicating the maximum COI to compare the simulated data to.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`distance`	**[Deprecated]** This argument is no longer supported; this function will solve a weighted least squares minimization problem.
`coi_method`	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".
`seq_error_bin_size`	Number of loci in smallest bin for estimating sequence error

Value

The predicted COI value.

See Also

stats::optim() for the complete documentation on the optimization function.

Other optimization functions: likelihood(), optimize_coi()

plot-simulation *Plot simulated data*

Description

Generate a simple plot visualizing simulated data. Compares the derived WSMAF to the PLMAF.

Usage

```
## S3 method for class 'sim'
autoplot(object, ...)

## S3 method for class 'sim'
plot(x, ...)
```

Arguments

`object`, `x`	An object of class `sim`. Derived from the output of `sim_biallelic()`.
`...`	Other arguments passed on to methods.

See Also

Other simulated data functions: `process_sim()`, `sim_biallelic()`

Examples

```
plot(sim_biallelic(coi = 2))
plot(sim_biallelic(coi = 5))
```

process_real *Process real data*

Description

Generate the COI curve for real data.

Usage

```
process_real(
  wsmaf,
  plmaf,
  coverage,
  seq_error = 0.01,
  bin_size = 20,
  coi_method = "variant"
)
```

Arguments

wsmaf	The within-sample minor allele frequency.
plmaf	The population-level minor allele frequency.
coverage	The read coverage at each locus.
seq_error	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
bin_size	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing plaf.
coi_method	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".

Details

The function computes whether a SNP is a variant site or not, based on the WSMAF at that SNP. This process additionally accounts for potential sequencing error.

Value

A list of the following:

- data: A tibble with
- plmaf_cut: Breaks of the form [a, b).
- m_variant: The average WSMAF or proportion of variant sites in each segment defined by plmaf_cut.
- bucket_size: The number of loci in each bucket.
- midpoints: The midpoint of each bucket.
- seq_error: The sequence error inferred.
- bin_size: The minimum size of each bin.
- cuts: The breaks utilized in splitting the data. of each COI.

See Also

process_sim() to process simulated data.

`process_sim` *Process simulated data*

Description

Generate the simulated COI curve.

Usage

```
process_sim(sim, seq_error = 0.01, bin_size = 20, coi_method = "variant")
```

Arguments

`sim`	Output of `sim_biallelic()`.
`seq_error`	The level of sequencing error that is assumed. If no value is inputted, then we infer the level of sequence error.
`bin_size`	**[Deprecated]** This argument is no longer supported; to estimate the COI, all data points are used. Data points are not grouped in bins of changing `plaf`.
`coi_method`	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".

Details

Utilize the output of `sim_biallelic()`, which creates simulated data. The PLMAF is kept, and the function computes whether a SNP is a variant site or not, based on the simulated WSMAF at that SNP. This process additionally accounts for potential sequencing error. To check whether the simulated WSMAF correctly indicated a variant site or not, the phased haplotype of the parasites is computed.

Value

A list of the following:

- `data`: A tibble with
- `plmaf_cut`: Breaks of the form `[a, b)`.
- `m_variant`: The average WSMAF or proportion of variant sites in each segment defined by `plmaf_cut`.
- `bucket_size`: The number of loci in each bucket.
- `midpoints`: The midpoint of each bucket.
- `seq_error`: The sequence error inferred.
- `bin_size`: The minimum size of each bin.
- `cuts`: The breaks utilized in splitting the data. of each COI.

See Also

`process_real()` to process real data.

Other simulated data functions: `plot-simulation`, `sim_biallelic()`

`sensitivity_plot` *Sensitivity plot*

Description

Creates a plot of the sensitivity analysis.

Usage

```
sensitivity_plot(
  data,
  dims,
  result_type,
  sub_title = NULL,
  title = NULL,
  caption = NULL
)
```

Arguments

`data`	The data to be plotted.
`dims`	A list representing the number of rows and columns our plots will be split into.
`result_type`	An indicator that indicates if a count or boxplot should be plotted.
`sub_title`	A list of titles for each individual subplot.
`title`	The title of the overall figure.
`caption`	The caption of the overall figure.

Details

Creates a grid of plots. Each plot is created using `ggplot2::geom_count()`. The number of observations at each location is counted and then the count is mapped to point area on the plot.

The x-axis is the true COI, and the y-axis is the estimated COI. The counts are plotted in blue, and red line is drawn with the equation $y = x$. This line indicates where the blue circles should be if the algorithm was 100% correct.

See Also

`ggplot2::geom_count()` for more information on count plots and the ggplot2 website.

Other plotting functions: `error_plot()`, `world_map()`

`sim_biallelic` *Simulate biallelic data*

Description

Simulate biallelic data from a simple statistical model. Inputs include the complexity of infection (COI), population-level minor allele frequencies (PLMAF), and some parameters dictating skew and error distributions. Outputs include the phased haplotypes and the unphased read count and coverage data.

Usage

```
sim_biallelic(
  coi,
  plmaf = runif(10, 0, 0.5),
  coverage = 200,
  alpha = 1,
  overdispersion = 0,
  relatedness = 0,
  epsilon = 0
)
```

Arguments

`coi`	Complexity of infection.
`plmaf`	Vector of population-level minor allele frequencies at each locus.
`coverage`	Coverage at each locus. If a single value is supplied then the same coverage is applied over all loci.
`alpha`	Shape parameter of the symmetric Dirichlet prior on strain proportions.
`overdispersion`	The extent to which counts are over-dispersed relative to the binomial distribution. Counts are Beta-Binomially distributed, with the beta distribution having shape parameters $\frac{p}{overdispersion}$ and $\frac{1-p}{overdispersion}$.
`relatedness`	The probability that a strain in mixed infections is related to another. The implementation is similar to relatedness as defined in THE REAL McCOIL simulations (doi:10.1371/journal.pcbi.1005348): "... simulated relatedness (r) among lineages within the same host by sampling alleles either from an existing lineage within the same host (with probability r) or from the population (with probability (1-r))."
`epsilon`	The probability of a single read being miscalled as the other allele. This error is applied in both directions.

Details

Simulated data are drawn from a simple statistical model:

1. Strain proportions are drawn from a symmetric Dirichlet distribution with shape parameter `alpha`.
2. Phased haplotypes are drawn at every locus, one for each `coi`. The allele at each locus is drawn from a Bernoulli distribution with probability given by the `plmaf`.
3. The "true" within-sample allele frequency at every locus is obtained by multiplying haplotypes by their strain proportions, and summing over haplotypes. Errors are introduced through the equation

$$wsmaf_{error} = wsmaf(1 - e) + (1 - wsmaf)e$$

   where $wsmaf$ is the WSMAF without error and e is the error parameter `epsilon`.
4. Final read counts are drawn from a beta-binomial distribution with expectation w_{error}. The raw number of draws is given by the `coverage`, and the skew of the distribution is given by the `overdispersion` parameter. If the `overdispersion` is equal to zero, then the distribution is binomial, rather than beta-binomial.

Value

An object of class `sim`. Contains a list of tibbles:

- `parameters` contains each parameter and the value used to simulate data.
- `strain_proportions` contains the proportion of each strain.
- `phased_haplotypes` contains the phased haplotype for each strain at each locus.
- `data` contains the following columns:
  - `plmaf`: The population-level minor allele frequency.
  - `coverage`: The coverage at each locus.
  - `counts`: The count at each locus.
  - `wsaf`: The within-sample minor allele frequency.

See Also

Other simulated data functions: `plot-simulation`, `process_sim()`

Examples

```
sim_biallelic(coi = 5)
```

---

theme_coiaf *Custom ggplot2 theme*

Description

Custom ggplot2 theme

Usage

```
theme_coiaf(
  base_size = 10,
  base_family = "",
  base_line_size = base_size/22,
  base_rect_size = base_size/22
)
```

Arguments

base_size	base font size, given in pts.
base_family	base font family
base_line_size	base size for line elements
base_rect_size	base size for rect elements

Examples

```
library("ggplot2")
p <- ggplot(mtcars, aes(x = wt, y = mpg, colour = factor(gear))) +
  geom_point() +
  facet_wrap(~am) +
  geom_smooth(method = "lm", se = FALSE)

p + theme_coiaf()
```

---

theoretical_coi *Theoretical COI*

Description

Generate the theoretical relationship between the WSMAF ($\mathbf{w}$), the PLMAF ($\mathbf{p}$), and the COI (k).

Usage

```
theoretical_coi(
  coi_range,
  plmaf = seq(0, 0.5, length.out = 101),
  coi_method = c("variant", "frequency")
)
```

Arguments

coi_range	The COIs for which the relationship will be generated.
plmaf	The population-level minor allele frequency over which the relationship will be generated.
coi_method	The method we will use to generate the theoretical relationship. The method is either "variant" or "frequency". The default value is "variant".

Value

A tibble() containing the generated values. Each column is named with the COI used. The last column of the tibble contains the PLMAF.

Examples

```
theoretical_coi(1:5)
theoretical_coi(1:5, coi_method = "frequency")
```

---

world_map *World map plot*

Description

Plot a world map showing the COI in each region where reads were sampled from.

Usage

```
world_map(data, variable, label = NULL, alpha = 0.1, breaks = c(1, 2))
```

Arguments

data	The data to be plotted.
variable	The variable the data will plot.
label	The label for the variable.
alpha	The alpha value for the plotted data.
breaks	The breaks for the color scale.

Details

Creates a world map and overlays the COI in each region. The magnitude of the COI is indicated by both the color and the size of the bubble.

See Also

This website for more information on creating bubble graphs in R.

Other plotting functions: error_plot(), sensitivity_plot()

Index