

Package ‘simDNAmixtures’

July 23, 2025

Title Simulate Forensic DNA Mixtures

Version 1.1.2

Description Mixed DNA profiles can be sampled according to models for probabilistic genotyping. Peak height variability is modelled using a log normal distribution or a gamma distribution. Sample contributors may be related according to a pedigree.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, naturalsort, pedprobr, xml2

Suggests rmarkdown, knitr, pedtools, testthat (>= 3.0.0), readr,
readxl, withr

Config/testthat/edition 3

Depends R (>= 4.1.0)

LazyData true

VignetteBuilder knitr

URL <https://mkruijver.github.io/simDNAmixtures/>

NeedsCompilation no

Author Maarten Kruijver [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-6890-7632>>)

Maintainer Maarten Kruijver <maarten.kruijver@esr.cri.nz>

Repository CRAN

Date/Publication 2025-04-14 14:10:01 UTC

Contents

allele_specific_stutter_model	2
drop_model	4
gamma_model	5
gf_configuration	7
global_stutter_model	8

kits	9
log_normal_model	9
read_allele_freqs	11
read_size_regression	12
read_STRmix_kit_settings	13
read_stutter_exceptions	14
read_stutter_regression	15
read_wide_table	15
sample_contributor_genotypes	16
sample_drop_model	17
sample_gamma_model	18
sample_genotype	20
sample_log_normal_model	20
sample_log_normal_parameters	21
sample_log_normal_stutter_variance	23
sample_LSAE	23
sample_many_pedigree_genotypes	24
sample_mixtures	25
sample_mixtures_fixed_parameters	28
sample_mixtures_from_genotypes	29
sample_mixture_from_genotypes	32
sample_offspring	33
sample_pedigree_genotypes	34
SMASH_to_wide_table	35
stutter_type	36
Index	38

allele_specific_stutter_model

Stutter model where the expected stutter rate depends on the allele and locus

Description

Stutter model where the expected stutter rate depends on the allele and locus

Usage

```
allele_specific_stutter_model(  
  stutter_types,  
  size_regression,  
  sex_locus_name = "AMEL"  
)
```

Arguments

stutter_types List. See [stutter_type](#).
 size_regression Function, see [read_size_regression](#).
 sex_locus_name Character vector, defaults to "AMEL".

Details

When a `pg_model` is constructed (see [gamma_model](#)), a stutter model can optionally be applied. The allele specific stutter model is commonly used with a log normal model. The expected stutter ratio for a parent allele at a locus is obtained from a linear regression of observed stutter ratios against allele length. For some loci or alleles the linear model may not be satisfactory. To override the expected stutter rates for specific alleles, a list of exceptions can be used. See [stutter_type](#) for more detail.

Value

Object of class `stutter_model` to be used by e.g. [log_normal_model](#).

See Also

[global_stutter_model](#) for a stutter model where the expected stutter ratio does not depend on the locus or parent allele.

Examples

```
# we will define an allele specific stutter model for back stutter only

# prepare stutter regression
filename_bs_regression <- system.file("extdata",
  "GlobalFiler_Stutter_3500.txt", package = "simDNAmixtures")
bs_regression <- read_stutter_regression(filename_bs_regression)

# prepare exceptions, i.e. where does the regression not apply?
filename_bs_exceptions <- system.file("extdata",
  "GlobalFiler_Stutter_Exceptions_3500.csv", package = "simDNAmixtures")
bs_exceptions <- read_stutter_exceptions(filename_bs_exceptions)

# prepare a stutter type
backstutter <- stutter_type(name = "BackStutter", delta = -1,
  stutter_regression = bs_regression,
  stutter_exceptions = bs_exceptions)

# assign stutter model
size_regression <- read_size_regression(system.file("extdata",
  "GlobalFiler_SizeRegression.csv", package = "simDNAmixtures"))
bs_model <- allele_specific_stutter_model(list(backstutter), size_regression)
bs_model
```

drop_model	<i>Defines a (semi-continuous) drop model</i>
------------	---

Description

Defines a (semi-continuous) drop model

Usage

```
drop_model(dropout_probabilities, drop_in_rate = 0, freqs, model_settings)
```

Arguments

dropout_probabilities	Numeric vector with values between 0 and 1. Dropout probabilities for each contributor.
drop_in_rate	Numeric vector of length one. Expected number of drop-ins per locus. Default is 0.
freqs	Optionally a list with allele frequencies (needed when drop_in_rate > 0). See read_allele_freqs .
model_settings	List. Possible parameters: <ul style="list-style-type: none"> • locus_names. Character vector. • size_regression. Function, see read_size_regression.

Details

Define the classes semi-continuous drop-model. The model may then be used to sample DNA profiles using the [sample_mixture_from_genotypes](#) function. Alternatively, to sample many models and profiles in one go with parameters according to a specified distribution, the [sample_mixtures](#) function can be used.

Value

Object of class pg_model.

References

Slooten, K. (2017). Accurate assessment of the weight of evidence for DNA mixtures by integrating the likelihood ratio. Forensic Science International: Genetics, 27, 1-16. doi:10.1016/j.fsigen.2016.11.001

See Also

[sample_mixture_from_genotypes](#), [sample_mixtures](#), [gamma_model](#), [log_normal_model](#).

Examples

```
gf <- gf_configuration()
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))
settings <- list(locus_names = gf$autosomal_markers, size_regression = gf$size_regression)

model <- drop_model(dropout_probabilities = c(0.1),
                    drop_in_rate = 1e-3,
                    freqs = freqs, model_settings = settings)

g <- sample_contributor_genotypes(contributors = "U1", freqs = freqs,
                                loci = settings$locus_names)

# genotype
g

# sample with dropout
sample_mixture_from_genotypes(g, model)
```

gamma_model

*Defines a gamma model for peak height variability***Description**

Defines a gamma model for peak height variability

Usage

```
gamma_model(
  mixture_proportions,
  mu,
  cv,
  degradation_beta = rep(1, length(mixture_proportions)),
  LSAE = stats::setNames(rep(1, length(model_settings$locus_names)),
    model_settings$locus_names),
  model_settings
)
```

Arguments

mixture_proportions	Numeric vector with the mixture proportion for each contributor.
mu	Numeric. Expectation of a full heterozygote contributing allele peak height.
cv	Numeric. Coefficient of variation of a full heterozygote contributing allele peak height

degradation_beta	Numeric Vector of same length as mixture_proportions. Degradation slope parameters for each contributor. Defaults to 1 for each contributor (i.e. not degraded)
LSAE	Numeric vector (named) with Locus Specific Amplification Efficiencies. See sample_LSAE . Defaults to 1 for each locus.
model_settings	List. Possible parameters: <ul style="list-style-type: none"> • locus_names. Character vector. • detection_threshold. Numeric vector (named) with Detection Thresholds. • size_regression. Function, see read_size_regression. • stutter_model. Optionally a stutter_model object that gives expected stutter heights. See global_stutter_model.

Details

Define a gamma model for peak height variability with the parametrisation as described by Bleka et al. The model may then be used to sample DNA profiles using the [sample_mixture_from_genotypes](#) function. Alternatively, to sample many models and profiles in one go with parameters according to a specified distribution, the [sample_mixtures](#) function can be used.

Value

Object of class `pg_model`.

References

Bleka, Ø., Storvik, G., & Gill, P. (2016). EuroForMix: An open source software based on a continuous model to evaluate STR DNA profiles from a mixture of contributors with artefacts. *Forensic Science International: Genetics*, 21, 35-44. doi:10.1016/j.fsigen.2015.11.008

See Also

[log_normal_model](#).

Examples

```
# read allele frequencies
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))

gf <- gf_configuration()

# define the gamma model for peak heights
model <- gamma_model(mixture_proportions = 1, mu = 1000.,
                    cv = 0.1, model_settings = gf$gamma_settings_no_stutter)

# sample a single source profile (1-person 'mixture')
u1 <- sample_contributor_genotypes("U1", freqs, loci = gf$autosomal_markers)
sample <- sample_mixture_from_genotypes(u1, model)
```

```
# peaks follow a gamma distribution with an expected height of
# 1,000 for heterozygous alleles; 2,000 for homozygotes
hist(sample$Height)

# the gamma distribution is more obvious if many samples are taken
many_samples <- replicate(n = 1e2,
                          sample_mixture_from_genotypes(u1, model),
                          simplify = FALSE)

hist(sapply(many_samples, function(x) x$Height))
```

gf_configuration	<i>Stutters and size regressions for a GlobalFiler 3500 kit</i>
------------------	---

Description

A dataset containing default parameters and settings for a GlobalFiler 3500 kit.

Usage

```
gf_configuration()
```

Format

A list of:

autosomal_markers Names of autosomal markers in the GlobalFiler kit

repeat_length_by_marker Named numeric with STR repeat length by locus name

size_regression See [read_size_regression](#)

stutters List of 4 stutter types, to be used with [allele_specific_stutter_model](#)

stutter_model For convenience, a pre-defined [allele_specific_stutter_model](#)

log_normal_settings Settings corresponding to a log normal model with all stutter types

log_normal_settings_fwbw Settings corresponding to a log normal model with backward and forward stutter only

gamma_settings Settings corresponding to a gamma model with all stutter types

gamma_settings_no_stutter Settings for a gamma model without stutter

global_stutter_model	<i>Global stutter model where the expected stutter rate is constant across alleles and loci</i>
----------------------	---

Description

Global stutter model where the expected stutter rate is constant across alleles and loci

Usage

```
global_stutter_model(
  back_stutter_rate,
  forward_stutter_rate,
  size_regression,
  sex_locus_name = "AMEL"
)
```

Arguments

back_stutter_rate
Numeric. (Optional)

forward_stutter_rate
Numeric. (Optional)

size_regression
Function, see [read_size_regression](#).

sex_locus_name Character vector, defaults to "AMEL".

Details

When a `pg_model` is constructed (see [gamma_model](#)), a stutter model can optionally be applied. In the global stutter model, the expected stutter rate is constant across all loci and for all parent alleles.

Value

Object of class `stutter_model` to be used by e.g. [gamma_model](#).

See Also

[allele_specific_stutter_model](#) for a stutter model where the expected stutter rate depends on the allele and locus.

Examples

```
# the stutter model needs a size regression to determine fragment length
# of stutter products
size_regression <- read_size_regression(system.file("extdata",
"GlobalFiler_SizeRegression.csv", package = "simDNAmixtures"))
```



```
# define a stutter model with an expected back stutter rate of 10%
stutter_model <- global_stutter_model(back_stutter_rate = 0.1,
                                     size_regression = size_regression)

stutter_model
```

kits

Properties such as loci, dye, sizes for most standard forensic kits

Description

A dataset containing the properties of forensic DNA kits.

Usage

```
kits
```

Format

A list of data frames containing variables such as:

Panel Kit name

Marker

Allele

Size Fragment length

Color Dye colour

Source

<https://github.com/oyvble/euroformix/blob/master/inst/extdata/kit.txt>

log_normal_model

Defines a log normal model for peak height variability

Description

Defines a log normal model for peak height variability

Usage

```
log_normal_model(
  template,
  degradation = rep(0, length(template)),
  LSAE = stats::setNames(rep(1, length(model_settings$locus_names)),
    model_settings$locus_names),
  c2,
  k2,
  model_settings
)
```

Arguments

template	Numeric vector
degradation	Numeric vector of same length as template. Degradation parameters for each contributor.
LSAE	Numeric vector (named) with Locus Specific Amplification Efficiencies. See sample_LSAE . Defaults to 1 for each locus.
c2	Numeric. Allele variance parameter.
k2	Optionally a numeric vector with stutter variance parameters. See sample_log_normal_stutter_variance .
model_settings	List. Possible parameters: <ul style="list-style-type: none"> • locus_names. Character vector. • degradation_parameter_cap. Numeric. • c2_prior. Numeric of length two with shape and scale. • LSAE_variance_prior. Numeric of length one. • detection_threshold. Numeric vector (named) with Detection Thresholds. Defaults to 50 for each locus. • size_regression. Function, see read_size_regression. • stutter_model. Optionally a stutter_model object that gives expected stutter heights. See global_stutter_model. • stutter_variability. Optionally peak height variability parameters for stutters. Required when stutter_model is supplied.

Details

Define a log normal model for peak height variability with the parametrisation as described by Bright et al. The model may then be used to sample DNA profiles using the [sample_mixture_from_genotypes](#) function. Alternatively, to sample many models and profiles in one go with parameters according to a specified distribution, the [sample_mixtures](#) function can be used.

Value

Object of class pg_model.

References

Bright, J.A. et al. (2016). Developmental validation of STRmix™, expert software for the interpretation of forensic DNA profiles. Forensic Science International: Genetics, 23, 226-239. [doi:10.1016/j.fsigen.2016.05.007](https://doi.org/10.1016/j.fsigen.2016.05.007)

See Also

[gamma_model](#).

Examples

```
gf <- gf_configuration()
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))

k2 <- sample_log_normal_stutter_variance(gf$log_normal_settings$stutter_variability)

model <- log_normal_model(template = 1e3, c2 = 15, k2 = k2,
                          model_settings = gf$log_normal_settings)
model
```

read_allele_freqs	<i>Read allele frequencies in FSIgen format (.csv)</i>
-------------------	--

Description

Read allele frequencies in FSIgen format (.csv)

Usage

```
read_allele_freqs(filename, remove_zeroes = TRUE, normalise = TRUE)
```

Arguments

filename	Path to csv file.
remove_zeroes	Logical. Should frequencies of 0 be removed from the return value? Default is TRUE.
normalise	Logical. Should frequencies be normalised to sum to 1? Default is TRUE.

Details

Reads allele frequencies from a .csv file. The file should be in FSIgen format, i.e. comma separated with the first column specifying the allele labels and one column per locus. The last row should be the number of observations. No error checking is done since the file format is only loosely defined, e.g. we do not restrict the first column name or the last row name.

Value

Named list with frequencies by locus. The frequencies at a locus are returned as a named numeric vector with names corresponding to alleles.

Examples

```
# below we read an allele freqs file that comes with the package
filename <- system.file("extdata", "FBI_extended_Cauc_022024.csv", package = "simDNAmixtures")
freqs <- read_allele_freqs(filename)
freqs # the output is a list with an attribute named \code{N} giving the sample size.
```

read_size_regression *Reads a size regression file*

Description

Reads a size regression file

Usage

```
read_size_regression(filename, exceptions, repeat_length_by_marker)
```

Arguments

filename	Path to file (character).
exceptions	Optionally a list providing sizes for alleles not covered by the regression. See examples for how this can be used to assign sizes to X and Y at the Amelogenin locus.
repeat_length_by_marker	Optionally a named integer vector with repeat lengths by marker. If not provided, then a .3 allele will not convert to e.g. .75 for a tetranucleotide.

Details

Read a regression file from disk and returns a function that provides the fragment length (bp) for a given locus and allele.

DNA profiles consist of the observed peaks (alleles or stutter products) at several loci as well as the peak heights and sizes. The size refers to the fragment length (bp). A linear relationship exists between the size of a peak and the size. When peaks are sampled in the [sample_mixture_from_genotypes](#) function, a size is assigned using a size regression. The read_size_regression function reads such a regression from disk.

Value

A function that takes a locus name and allele as arguments and returns the size.

Examples

```
filename <- system.file("extdata",
                        "GlobalFiler_SizeRegression.csv",
                        package = "simDNAmixtures")

regression <- read_size_regression(filename)

# obtain size for the 12 allele at the vWA locus
regression("vWA", 12)

# now add AMEL sizes
regression_with_AMEL <- read_size_regression(filename, exceptions = list(
```

```

      AMEL = stats::setNames(c(98.5, 104.5),
                             nm = c("X", "Y"))))

# check that we can obtain size for X at AMEL
stopifnot(regression_with_AMEL("AMEL", "X") == 98.5)

# pass in repeat_length_by_marker for more precise estimates
gf <- gf_configuration()

regression_with_repeat_length <- read_size_regression(filename,
  repeat_length_by_marker = gf$repeat_length_by_marker)

# obtain size for the 15.3 allele at the D1S1656 locus
stopifnot(regression_with_repeat_length("D1S1656", 15.3) ==
  121.628203912362 + 15.75 * 4.2170043570021)

```

read_STRmix_kit_settings

Read STRmix Kit Settings

Description

Read STRmix kit settings from an XML file and associated stutter directory.

Usage

```
read_STRmix_kit_settings(filename, stutters_dir, include_y_loci = FALSE)
```

Arguments

filename	Character vector specifying the path to the XML file containing the profiling kit settings.
stutters_dir	Character vector specifying the directory path where stutter settings files are located.
include_y_loci	Should Y loci be included in the list of locus names? Default is FALSE.

Value

A list containing the following components:

- locus_names. Character vector.
- degradation_parameter_cap. Numeric.
- c2_prior. Numeric of length two with shape and scale.
- LSAE_variance_prior. Numeric of length one.
- detection_threshold. Numeric vector (named) with Detection Thresholds. Defaults to 50 for each locus.
- size_regression. Function, see [read_size_regression](#).

- `stutter_model`. A list representing the stutter model.
- `stutter_variability`. A list representing the stutter variability.

`read_stutter_exceptions`*Reads a stutter exceptions file with overrides for expected stutter ratios*

Description

Reads a stutter exceptions file with overrides for expected stutter ratios

Usage

```
read_stutter_exceptions(filename)
```

Arguments

<code>filename</code>	Character. Path to file.
-----------------------	--------------------------

Details

Reads the file from disk and returns a named numeric vector with stutter ratio exceptions for a given locus and allele.

Value

A named list with the stutter exceptions by locus. For each locus, the exceptions are given as a named numeric with the names corresponding to the parent alleles and the expected stutter rates given as the values.

Examples

```
filename <- system.file("extdata", "GlobalFiler_Stutter_Exceptions_3500.csv",  
                        package = "simDNAmixtures")  
exceptions <- read_stutter_exceptions(filename)  
exceptions$TH01["9.3"]
```

```
read_stutter_regression
```

Reads a stutter regression file

Description

Reads a stutter regression file

Usage

```
read_stutter_regression(filename, min_stutter_ratio = 0.001)
```

Arguments

filename Character. Path to file.
min_stutter_ratio Numeric.

Details

Reads the file from disk and returns a function that provides the expected stutter ratio for a given locus and allele.

Value

A function that takes a locus name and allele as arguments and returns the expected stutter ratio.

Examples

```
filename <- system.file("extdata", "GlobalFiler_Stutter_3500.txt",
                        package = "simDNAmixtures")
regression <- read_stutter_regression(filename)

# obtain the expected stutter ratio for a 12 parent allele at the vWA locus
regression("vWA", 12)
```

```
read_wide_table
```

Read wide table (.txt) with Allele1, Allele2, ... columns as is

Description

Read wide table (.txt) with Allele1, Allele2, ... columns as is

Usage

```
read_wide_table(filename)
```

Arguments

filename Path to txt file.

Value

Dataframe

sample_contributor_genotypes

Sample genotypes for mixture contributors according to allele frequencies

Description

Sample genotypes for mixture contributors according to allele frequencies

Usage

```
sample_contributor_genotypes(
  contributors,
  freqs,
  linkage_map,
  pedigree,
  loci = names(freqs),
  return_non_contributors = FALSE,
  sex_locus_name = "AMEL"
)
```

Arguments

contributors Character vector with unique names of contributors. Valid names are "U1", "U2", ... for unrelated contributors or the names of pedigree members for related contributors.

freqs Allele frequencies (see [read_allele_freqs](#))

linkage_map (optional) A linkage map specifying the recombination fractions between loci. If missing, loci are assumed to be independent. See also [sample_many_pedigree_genotypes](#).

pedigree (optional) [ped](#) object

loci Character vector of locus names (defaults to names attribute of freqs)

return_non_contributors Logical. Should genotypes of non-contributing pedigree members also be returned?

sex_locus_name Character vector, defaults to "AMEL"

Details

For each founder or unrelated person, a genotype is sampled randomly by drawing two alleles from allele frequencies. The non-founders get genotypes by allele dropping, see [sample_pedigree_genotypes](#) for details.

Value

List of DataFrames with genotypes for each pedigree member. See [sample_genotype](#) for the DataFrame format.

Examples

```
# read allele frequencies
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))

# define a pedigree of siblings S1 and S2 (and their parents)
ped_sibs <- pedtools::nuclearPed(children = c("S1", "S2"))

# sample genotypes for a mixture of S1 + U1 + S2
# where U1 is an unrelated person

sample_contributor_genotypes(contributors = c("S1", "U1", "S2"),
                             freqs, pedigree = ped_sibs,
                             loci = gf_configuration()$autosomal_markers)

# now also include AMEL
sample_contributor_genotypes(contributors = c("S1", "S2", "U1"),
                             freqs, pedigree = ped_sibs,
                             loci = c(gf_configuration()$autosomal_markers, "AMEL"))
```

sample_drop_model	<i>Sample drop model(s) with parameters according to priors</i>
-------------------	---

Description

Sample drop model(s) with parameters according to priors

Usage

```
sample_drop_model(
  number_of_contributors,
  sampling_parameters,
  drop_in_rate = 0,
  model_settings
)
```

Arguments

- number_of_contributors
Integer
- sampling_parameters
List. Needs to contain:
- min_dropout_probability. Numeric of length one.
 - max_dropout_probability Numeric of length one.
- drop_in_rate Numeric vector of length one. Expected number of drop-ins per locus. Default is 0.
- model_settings List. See [drop_model](#).

Details

In simulation studies involving many mixed DNA profiles, one often needs to generate various samples with different model parameters. This function samples a drop model with parameters according to prior distributions. The dropout probability for each contributor is sampled uniformly between min_dropout_probability. and max_dropout_probability.

Value

When length(number_of_contributors)==1, a single [drop_model](#) of class pg_model. Otherwise, a list of these.

See Also

[sample_mixtures_fixed_parameters](#) to directly supply parameters of choice for more control

Examples

```
gf <- gf_configuration()

sampling_parameters <- list(min_dropout_probability. = 0., max_dropout_probability. = 0.5)

model <- sample_drop_model(number_of_contributors = 1,
                           sampling_parameters = sampling_parameters,
                           model_settings = list(locus_names = gf$autosomal_markers,
                                                  size_regression = gf$size_regression))
```

sample_gamma_model	<i>Sample gamma model(s) with parameters according to priors</i>
--------------------	--

Description

Sample gamma model(s) with parameters according to priors

Usage

```
sample_gamma_model(number_of_contributors, sampling_parameters, model_settings)
```

Arguments

number_of_contributors

Integer

sampling_parameters

List. Needs to contain:

- min_mu. Numeric of length one.
- max_mu. Numeric of length one.
- min_cv. Numeric of length one.
- max_cv. Numeric of length one.
- degradation_shape1. Numeric of length one.
- degradation_shape2. Numeric of length one.

model_settings List. See [gamma_model](#).

Details

In simulation studies involving many mixed DNA profiles, one often needs to generate various samples with different model parameters. This function samples a gamma model with parameters according to prior distributions. The mean peak height parameter μ is sampled uniformly between min_mu and max_mu. Likewise, the variability parameter cv is sampled uniformly between min_cv and max_cv. The degradation slope parameter β is sampled according to a Beta distribution with parameters degradation_shape1 and degradation_shape2.

Value

When `length(number_of_contributors)==1`, a single [gamma_model](#) of class `pg_model1`. Otherwise, a list of these.

Examples

```
gf <- gf_configuration()

sampling_parameters <- list(min_mu = 50., max_mu = 5e3,
                           min_cv = 0.05, max_cv = 0.35,
                           degradation_shape1 = 10, degradation_shape2 = 1)

model_no_stutter <- sample_gamma_model(number_of_contributors = 2,
                                       sampling_parameters = sampling_parameters,
                                       model_settings = gf$gamma_settings_no_stutter)

model_no_stutter$parameters
```

sample_genotype	<i>Sample a genotype according to allele frequencies</i>
-----------------	--

Description

Sample a genotype according to allele frequencies

Usage

```
sample_genotype(freqs, loci = names(freqs), label = "U")
```

Arguments

freqs	Allele frequencies (see read_allele_freqs)
loci	Character vector of locus names (defaults to names attribute of freqs)
label	Sample name

Details

A genotype is sampled randomly by drawing two alleles from allele frequencies for each locus.

Value

DataFrame with columns Sample Name, Locus, Allele1 and Allele2.

Examples

```
# below we read an allele freqs and sample a genotype
filename <- system.file("extdata", "FBI_extended_Cauc_022024.csv",
  package = "simDNAmixtures")
freqs <- read_allele_freqs(filename)
sample_genotype(freqs, loci = c("D3S1358", "vWA"))
```

sample_log_normal_model

Sample log normal model(s) with parameters according to priors

Description

Sample log normal model(s) with parameters according to priors

Usage

```
sample_log_normal_model(
  number_of_contributors,
  sampling_parameters,
  model_settings
)
```

Arguments

number_of_contributors

Integer

sampling_parameters

List. Needs to contain:

- min_template. Numeric of length one.
- max_template. Numeric of length one.
- degradation_shape. Numeric of length one.
- degradation_scale. Numeric of length one.

model_settings List. See [log_normal_model](#).**Details**

In simulation studies involving many mixed DNA profiles, one often needs to generate various samples with different model parameters. This function samples a log normal model with parameters according to prior distributions. The template parameter for each contributor is sampled uniformly between min_template and max_template. The degradation parameter for each contributor is sampled from a gamma distribution with parameters degradation_shape and degradation_scale.

Value

When length(number_of_contributors)==1, a single [log_normal_model](#) of class pg_model. Otherwise, a list of these.

Examples

```
gf <- gf_configuration()

sampling_parameters <- list(min_template = 50., max_template = 1000.,
                           degradation_shape = 2.5, degradation_scale = 1e-3)

model_no_stutter <- sample_log_normal_model(number_of_contributors = 1,
                                           sampling_parameters = sampling_parameters,
                                           model_settings = gf$log_normal_settings)
```

sample_log_normal_parameters

Sample log normal model parameters according to priors

Description

Sample log normal model parameters according to priors

Usage

```
sample_log_normal_parameters(  
  number_of_contributors,  
  sampling_parameters,  
  model_settings  
)
```

Arguments

number_of_contributors

Integer

sampling_parameters

List. Needs to contain:

- min_template. Numeric of length one.
- max_template. Numeric of length one.
- degradation_shape. Numeric of length one.
- degradation_scale. Numeric of length one.

model_settings List. See [log_normal_model](#).

Details

In simulation studies involving many mixed DNA profiles, one often needs to generate various samples with different model parameters. This function samples a log normal model with parameters according to prior distributions. The template parameter for each contributor is sampled uniformly between min_template and max_template. The degradation parameter for each contributor is sampled from a gamma distribution with parameters degradation_shape and degradation_scale.

Value

When `length(number_of_contributors)==1`, a single [log_normal_model](#) of class `pg_model`. Otherwise, a list of these.

Examples

```
gf <- gf_configuration()  
  
sampling_parameters <- list(min_template = 50., max_template = 1000.,  
                             degradation_shape = 2.5, degradation_scale = 1e-3)  
  
parameters <- sample_log_normal_parameters(number_of_contributors = 1,  
                                             sampling_parameters = sampling_parameters,  
                                             model_settings = gf$log_normal_settings)
```

sample_log_normal_stutter_variance

Sample log normal stutter variance parameters according to priors

Description

Sample log normal stutter variance parameters according to priors

Usage

```
sample_log_normal_stutter_variance(log_normal_stutter_variability)
```

Arguments

```
log_normal_stutter_variability
```

List of variability parameters. See [gf_configuration](#) for an example.

Value

Named numeric with stutter variance parameter for all stutter types. Names are k2 concatenated with the name of the stuter type. See example.

Examples

```
gf <- gf_configuration()
log_normal_stutter_variability <- gf$log_normal_settings$stutter_variability
k2 <- sample_log_normal_stutter_variance(log_normal_stutter_variability)
```

sample_LSAE

Sample Locus Specific Amplification Efficiency (LSAE) according to prior

Description

Sample Locus Specific Amplification Efficiency (LSAE) according to prior

Usage

```
sample_LSAE(LSAE_variance, locus_names)
```

Arguments

```
LSAE_variance
```

Numeric. See [gf_configuration](#) for an example.

```
locus_names
```

Character vector.

Details

In the Bright et al. log normal model, the expected peak height includes a multiplicative factor for the locus (marker). These factors are called the LSAEs (Locus Specific Amplification Efficiencies). In the model, the prior for the log10 of LSAE is normal with mean 0. The variance can be specified.

Value

Named numeric with LSAEs for each locus (names).

Examples

```
gf <- gf_configuration()
lsae <- sample_LSAE(LSAE_variance = gf$log_normal_settings$LSAE_variance_prior,
                    locus_names = gf$autosomal_markers)

# the barplot shows that some loci amplify better than others
barplot(lsae, las=2)
```

sample_many_pedigree_genotypes

Sample (many) genotypes for pedigree members according to allele frequencies by allele dropping and possibly taking linkage into account by simulating recombination.

Description

Sample (many) genotypes for pedigree members according to allele frequencies by allele dropping and possibly taking linkage into account by simulating recombination.

Usage

```
sample_many_pedigree_genotypes(
  pedigree,
  freqs,
  loci = names(freqs),
  unrelated_names = character(),
  linkage_map,
  number_of_replicates = 1L,
  sex_locus_name = "AMEL",
  return_transmission_vectors = FALSE
)
```

Arguments

pedigree	ped object
freqs	Allele frequencies (see read_allele_freqs)
loci	Character vector of locus names (defaults to names attribute of freqs)

`unrelated_names` Character vector with names of any additional unrelated persons. Defaults to length zero.

`linkage_map` A linkage map specifying cM distances between loci. If missing, loci are assumed to be independent.

`number_of_replicates` An integer specifying the number of replicate genotype samples to generate. Defaults to 1.

`sex_locus_name` Character vector, defaults to "AMEL"

`return_transmission_vectors` Should transmission vectors be returned as an attribute? These are usually not of interest, so the default is FALSE.

See Also

[read_allele_freqs](#)

Examples

```
# load allele frequencies
freqs <- read_allele_freqs(system.file("extdata",
                                       "FBI_extended_Cauc_022024.csv",
                                       package = "simDNAmixtures"))

# define a pedigree with two full siblings: S1 and S2
ped_fs <- pedtools::nuclearPed(children = c("S1", "S2"))

# define two linked loci
linkage_map <- data.frame(chromosome = c(12, 12),
                          locus = c("vWA", "D12S391"),
                          position = c(15.15, 26.63))

# sample genotypes ignoring linkage
sample_many_pedigree_genotypes(pedigree = ped_fs, freqs = freqs,
                               loci = c("vWA", "D12S391"),
                               number_of_replicates = 10)

# sample genotypes taking linkage into account
sample_many_pedigree_genotypes(pedigree = ped_fs, freqs = freqs,
                               loci = c("vWA", "D12S391"),
                               linkage_map = linkage_map,
                               number_of_replicates = 10)
```

sample_mixtures	<i>Sample mixtures with random genotypes and random parameters according to priors</i>
-----------------	--

Description

Sample mixtures with random genotypes and random parameters according to priors

Usage

```
sample_mixtures(
  n,
  contributors,
  freqs,
  linkage_map,
  sampling_parameters,
  model_settings,
  sample_model,
  pedigree,
  results_directory,
  seed,
  number_of_replicates = 1L,
  write_non_contributors = FALSE,
  tag = "simulation",
  silent = FALSE
)
```

Arguments

<code>n</code>	Integer. Number of samples.
<code>contributors</code>	Character vector with unique names of contributors. Valid names are "U1", "U2", ... for unrelated contributors or the names of pedigree members for related contributors.
<code>freqs</code>	Allele frequencies (see read_allele_freqs)
<code>linkage_map</code>	(optional) A linkage map specifying the recombination fractions between loci. If missing, loci are assumed to be independent. See also sample_many_pedigree_genotypes .
<code>sampling_parameters</code>	List. Passed to the <code>sample_model</code> function.
<code>model_settings</code>	List. Passed to the <code>sample_model</code> function.
<code>sample_model</code>	Function such as sample_log_normal_model .
<code>pedigree</code>	(optionally) ped object. Contributors can be named pedigree members.
<code>results_directory</code>	(optionally) Character with path to directory where results are written to disk.
<code>seed</code>	(optionally) Integer seed value that can be used to get reproducible runs. If results are written to disk, the 'Run details.txt' file will contain a seed that can be used for reproducing the result.
<code>number_of_replicates</code>	Integer. Number of replicate simulations for each sample.
<code>write_non_contributors</code>	Logical. If TRUE, sampled genotypes for non-contributing pedigree members will also be written to disk. Defaults to FALSE.

tag	Character. Used for sub directory name when results_directory is provided.
silent	Logical. If TRUE, then no message will be printed about where the output (if any) was written to disk.

Value

If results_directory is provided, this function has the side effect of writing results to disk.

Return value is a list with simulation results:

- call matched call
- smash DataFrame with all samples in SMASH format (see [SMASH_to_wide_table](#))
- samples Detailed results for each sample
- parameter_summary DataFrame with parameters for each sample

Examples

```
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))
gf <- gf_configuration()

sampling_parameters <- list(min_mu = 50., max_mu = 5e3,
                           min_cv = 0.05, max_cv = 0.35,
                           degradation_shape1 = 10, degradation_shape2 = 1)

mixtures <- sample_mixtures(n = 2, contributors = c("U1", "U2"), freqs = freqs,
                           sampling_parameters = sampling_parameters,
                           model_settings = gf$gamma_settings_no_stutter,
                           sample_model = sample_gamma_model)

# sample a mixture of two siblings taking into account

linkage_map <- data.frame(chromosome = c("12", "12"),
                          locus = c("vWA", "D12391"),
                          position = c(16.56662766, 29.48590551))

ped_sibs <- pedtools::nuclearPed(children = c("Sib1", "Sib2"))

sibs_mix <- sample_mixtures(n = 1, contributors = c("Sib1", "Sib2"),
                           freqs = freqs,
                           linkage_map = linkage_map,
                           pedigree = ped_sibs,
                           sampling_parameters = sampling_parameters,
                           model_settings = gf$gamma_settings_no_stutter,
                           sample_model = sample_gamma_model)

# an example using the semi-continuous drop model

drop_model_sampling_parameters <- list(min_dropout_probability. = 0.,
                                       max_dropout_probability. = 0.5)

drop_model_settings <- list(locus_names = gf$autosomal_markers,
```

```

size_regression = gf$size_regression)

mixtures <- sample_mixtures(n = 2, contributors = c("U1", "U2"), freqs = freqs,
                           sampling_parameters = drop_model_sampling_parameters,
                           model_settings = drop_model_settings,
                           sample_model = sample_drop_model)

```

sample_mixtures_fixed_parameters

Sample mixtures with known genotypes and fixed parameters

Description

Sample mixtures with known genotypes and fixed parameters

Usage

```

sample_mixtures_fixed_parameters(
  genotypes,
  parameter_summary,
  model_settings,
  results_directory,
  seed,
  tag = "simulation",
  silent = FALSE
)

```

Arguments

genotypes	List of contributor genotypes. See sample_contributor_genotypes .
parameter_summary	DataFrame with parameters for each sample.
model_settings	List. Passed to the sample_model function.
results_directory	(optionally) Character with path to directory where results are written to disk.
seed	(optionally) Integer seed value that can be used to get reproducible runs. If results are written to disk, the 'Run details.txt' file will contain a seed that can be used for reproducing the result.
tag	Character. Used for sub directory name when results_directory is provided.
silent	Logical. If TRUE, then no message will be printed about where the output (if any) was written to disk.

Value

If results_directory is provided, this function has the side effect of writing results to disk.

Return value is a list with simulation results:

- call matched call
- smash DataFrame with all samples in SMASH format (see [SMASH_to_wide_table](#))
- samples Detailed results for each sample
- parameter_summary DataFrame with parameters for each sample

See Also

[sample_mixtures](#)

Examples

```
# simulate autosomal samples with fixed parameters (from csv) and refs (from csv)
parameter_summary <- utils::read.csv(system.file(
  "extdata", "Example_2p_Parameter_Summary.csv",
  package = "simDNAmixtures"))

gf <- gf_configuration()

filename_refs <- system.file("extdata", "Example_References_DB.csv",
                             package = "simDNAmixtures")
db_refs <- utils::read.csv(filename_refs, check.names = FALSE)

genotypes <- simDNAmixtures:::.wide_references_to_allele_tables(db_refs)

samples <- sample_mixtures_fixed_parameters(genotypes = genotypes,
                                             parameter_summary = parameter_summary,
                                             model_settings = gf$log_normal_bwfw_settings,
                                             seed = 1)
```

sample_mixtures_from_genotypes

Sample mixtures with known genotypes and random parameters according to priors

Description

Sample mixtures with known genotypes and random parameters according to priors

Usage

```
sample_mixtures_from_genotypes(
  n,
  genotypes,
  sampling_parameters,
  model_settings,
  sample_model,
  results_directory,
  seed,
  number_of_replicates = 1L,
  tag = "simulation",
  silent = FALSE
)
```

Arguments

<code>n</code>	Integer. Number of samples.
<code>genotypes</code>	List of contributor genotypes. See sample_contributor_genotypes .
<code>sampling_parameters</code>	List. Passed to the <code>sample_model</code> function.
<code>model_settings</code>	List. Passed to the <code>sample_model</code> function.
<code>sample_model</code>	Function such as sample_log_normal_model .
<code>results_directory</code>	(optionally) Character with path to directory where results are written to disk.
<code>seed</code>	(optionally) Integer seed value that can be used to get reproducible runs. If results are written to disk, the 'Run details.txt' file will contain a seed that can be used for reproducing the result.
<code>number_of_replicates</code>	Integer. Number of replicate simulations for each sample.
<code>tag</code>	Character. Used for sub directory name when <code>results_directory</code> is provided.
<code>silent</code>	Logical. If TRUE, then no message will be printed about where the output (if any) was written to disk.

Value

If `results_directory` is provided, this function has the side effect of writing results to disk.

Return value is a list with simulation results:

- `call` matched call
- `smash` DataFrame with all samples in SMASH format (see [SMASH_to_wide_table](#))
- `samples` Detailed results for each sample
- `parameter_summary` DataFrame with parameters for each sample

See Also

[sample_mixtures](#)

Examples

```

# define two known GlobalFiler genotypes
K1 <- data.frame(
  `Sample Name` = "K1",
  Locus = c("D3S1358", "vWA", "D16S539", "CSF1PO", "TPOX", "D8S1179",
            "D21S11", "D18S51", "D2S441", "D19S433", "TH01", "FGA",
            "D22S1045", "D5S818", "D13S317", "D7S820", "SE33",
            "D10S1248", "D1S1656", "D12S391", "D2S1338"),
  Allele1 = c("14", "15", "11", "11", "8",
              "11", "28", "14", "11", "14", "9.3",
              "21", "15", "9", "9", "10", "14",
              "14", "12", "21", "20"),
  Allele2 = c("16", "18", "13", "11", "11",
              "12", "30", "16", "12", "14", "9.3",
              "24", "16", "11", "12", "10", "18",
              "14", "15", "22", "24"), check.names = FALSE
)

K2 <- data.frame(
  `Sample Name` = "K2",
  Locus = c("D3S1358", "vWA", "D16S539", "CSF1PO", "TPOX", "D8S1179",
            "D21S11", "D18S51", "D2S441", "D19S433", "TH01", "FGA",
            "D22S1045", "D5S818", "D13S317", "D7S820", "SE33",
            "D10S1248", "D1S1656", "D12S391", "D2S1338"),
  Allele1 = c("16", "16", "10", "13", "8",
              "11", "27", "17", "10", "13", "6",
              "23", "16", "11", "11", "9", "22.2",
              "14", "12", "22", "21"),
  Allele2 = c("16", "17", "12", "14", "8",
              "13", "30", "18", "11", "14", "6",
              "25", "17", "11", "11", "12", "28.2",
              "15", "14", "22", "23"), check.names = FALSE
)

# first sample three replicates of a low-level profile of K1 only
gf <- gf_configuration()

sampling_parameters <- list(min_template = 75., max_template = 75,
                           degradation_shape = 2.5, degradation_scale = 1e-3)

single_source_results <- sample_mixtures_from_genotypes(n = 1,
  genotypes = list(K1), sampling_parameters = sampling_parameters,
  number_of_replicates = 3, sample_model = sample_log_normal_model,
  model_settings = gf$log_normal_bwfw_settings)

# now sample two mixtures of K1 and K2 with two replicates each

mix_results <- sample_mixtures_from_genotypes(n = 2,
  genotypes = list(K1, K2), sampling_parameters = sampling_parameters,
  number_of_replicates = 3, sample_model = sample_log_normal_model,
  model_settings = gf$log_normal_bwfw_settings)

```

sample_mixture_from_genotypes

Sample mixture profile with provided genotypes

Description

Sample mixture profile with provided genotypes

Usage

```
sample_mixture_from_genotypes(genotypes, model, sample_name = "mixture")
```

Arguments

genotypes	List of contributor genotypes. See sample_contributor_genotypes .
model	pg_model object.
sample_name	Character. Defaults to "mixture".

Details

A mixture profile is sampled according to the provided pg_model (see [gamma_model](#), [log_normal_model](#) and [sample_contributor_genotypes](#)).

Value

DataFrame with at least SMASH columns (see [SMASH_to_wide_table](#)). Depending on the chosen pg_model (e.g. [gamma_model](#) or [log_normal_model](#)), other columns with further details about the simulation are returned as well.

See Also

[sample_mixtures](#) for a function that samples many mixtures in one go.

Examples

```
# read allele frequencies and kit data
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))

gf <- gf_configuration()

# define a pedigree of siblings S1 and S2 (and their parents)
ped_sibs <- pedtools::nuclearPed(children = c("S1", "S2"))

# sample genotypes for a mixture of S1 + U1 + S2
# where U1 is an unrelated person
genotypes <- sample_contributor_genotypes(contributors = c("S1", "U1", "S2"),
    freqs, pedigree = ped_sibs, loci = gf$autosomal_markers)
```



```
# define a gamma model for peak heights
gamma_model <- gamma_model(mixture_proportions = c(0.5, 0.3, 0.2), mu = 1000.,
                             cv = 0.1, model_settings = gf$gamma_settings_no_stutter)

# sample mixture from genotypes
mix <- sample_mixture_from_genotypes(genotypes, gamma_model)
```

sample_offspring	<i>Sample offspring from two parental genotypes</i>
------------------	---

Description

Sample offspring from two parental genotypes

Usage

```
sample_offspring(father, mother, label = "Child")
```

Arguments

father	DataFrame (see sample_genotype)
mother	DataFrame (see sample_genotype)
label	SampleName of child (character)

Details

A genotype is sampled according to Mendelian inheritance. That is, one of two alleles of a parent is passed down to the offspring.

Value

DataFrame (see [sample_genotype](#))

Examples

```
# below we read an allele freqs and sample a genotype
filename <- system.file("extdata", "FBI_extended_Cauc_022024.csv",
                         package = "simDNAmixtures")
freqs <- read_allele_freqs(filename)

# sample parents
father <- sample_genotype(freqs, loci = c("D3S1358", "vWA"))
mother <- sample_genotype(freqs, loci = c("D3S1358", "vWA"))

# sample child
child <- sample_offspring(father, mother)
```

sample_pedigree_genotypes

Sample genotypes for pedigree according to allele frequencies by allele dropping.

Description

Sample genotypes for pedigree according to allele frequencies by allele dropping.

Usage

```
sample_pedigree_genotypes(pedigree, freqs, loci = names(freqs))
```

Arguments

pedigree	ped object
freqs	Allele frequencies (see read_allele_freqs)
loci	Character vector of locus names (defaults to names attribute of freqs)

Details

For each founder, a genotype is sampled randomly by drawing two alleles according to allele frequencies. Alleles for the rest of the pedigree are then obtained by allele dropping: [sample_offspring](#) is invoked for each non-founder.

Value

List of DataFrames with genotypes for each pedigree member. See [sample_genotype](#) for the DataFrame format.

See Also

[sample_many_pedigree_genotypes](#) for a function that takes linkage into account.

Examples

```
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
                                     package = "simDNAmixtures"))
gf <- gf_configuration()

ped_sibs <- pedtools::nuclearPed(children = c("S1", "S2"))

sibs_genotypes <- sample_pedigree_genotypes(ped = ped_sibs,
                                           freqs = freqs, loci = gf$autosomal_markers)
```

SMASH_to_wide_table	<i>Converts SMASH (SampleName, Marker, Allele, Size, Height) data to a wide table</i>
---------------------	---

Description

Converts SMASH (SampleName, Marker, Allele, Size, Height) data to a wide table

Usage

```
SMASH_to_wide_table(x)
```

Arguments

`x` DataFrame with SampleName, Marker, Allele, Size, Height columns

Value

DataFrame with columns: Sample Name, Marker, Allele 1, Allele 2, ..., Size 1, Size 2, ..., Height 1, Height 2, ...

Examples

```
# generate example data in SMASH form
freqs <- read_allele_freqs(system.file("extdata", "FBI_extended_Cauc_022024.csv",
package = "simDNAmixtures"))
gf <- gf_configuration()

sampling_parameters <- list(min_mu = 50., max_mu = 5e3,
                           min_cv = 0.05, max_cv = 0.35,
                           degradation_shape1 = 10, degradation_shape2 = 1)

mixtures <- sample_mixtures(n = 2, contributors = c("U1"), freqs = freqs,
                           sampling_parameters = sampling_parameters,
                           model_settings = gf$gamma_settings_no_stutter,
                           sample_model = sample_gamma_model)

# convert from SMASH to wide table
wide_table <- SMASH_to_wide_table(mixtures$smash)
```

stutter_type	<i>Defines a stutter type to be used in the allele specific stutter model.</i>
--------------	--

Description

Defines a stutter type to be used in the allele specific stutter model.

Usage

```
stutter_type(
  name,
  delta,
  applies_to_all_loci = TRUE,
  stutter_regression,
  stutter_exceptions,
  applies_to_loci,
  repeat_length_by_marker
)
```

Arguments

name	Character. Name of the stutter, e.g. "BackStutter"
delta	Numeric. When length one, repeat units gained (lost when negative). When length two, the second element is the number of base pairs gained (lost).
applies_to_all_loci	Logical. Defaults to TRUE.
stutter_regression	Function. See read_stutter_regression .
stutter_exceptions	Optionally a list. See read_stutter_exceptions .
applies_to_loci	Optionally a character vector of locus names to which this stutter type applies.
repeat_length_by_marker	Optionally a named integer vector with repeat lengths by marker. Only needed when delta is of length two.

Details

When a `pg_model` is constructed (see [log_normal_model](#)), a stutter model can optionally be applied.

Value

Object of class `stutter_type` to be passed to [allele_specific_stutter_model](#).

Examples

```
filename_bs_exceptions <- system.file("extdata",  
  "GlobalFiler_Stutter_Exceptions_3500.csv", package = "simDNAmixtures")  
bs_exceptions <- read_stutter_exceptions(filename_bs_exceptions)  
  
filename_bs_regression <- system.file("extdata",  
  "GlobalFiler_Stutter_3500.txt", package = "simDNAmixtures")  
bs_regression <- read_stutter_regression(filename_bs_regression)  
  
backstutter <- stutter_type(name = "BackStutter", delta = -1,  
  stutter_regression = bs_regression,  
  stutter_exceptions = bs_exceptions)
```

Index

* **datasets**
 kits, 9

allele_specific_stutter_model, 2, 7, 8, 36

drop_model, 4, 18

gamma_model, 3, 4, 5, 8, 10, 19, 32

gf_configuration, 7, 23

global_stutter_model, 3, 6, 8, 10

kits, 9

log_normal_model, 3, 4, 6, 9, 21, 22, 32, 36

ped, 16, 24, 26, 34

read_allele_freqs, 4, 11, 16, 20, 24–26, 34

read_size_regression, 3, 4, 6–8, 10, 12, 13

read_STRmix_kit_settings, 13

read_stutter_exceptions, 14, 36

read_stutter_regression, 15, 36

read_wide_table, 15

sample_contributor_genotypes, 16, 28, 30, 32

sample_drop_model, 17

sample_gamma_model, 18

sample_genotype, 17, 20, 33, 34

sample_log_normal_model, 20, 26, 30

sample_log_normal_parameters, 21

sample_log_normal_stutter_variance, 10, 23

sample_LSAE, 6, 10, 23

sample_many_pedigree_genotypes, 16, 24, 26, 34

sample_mixture_from_genotypes, 4, 6, 10, 12, 32

sample_mixtures, 4, 6, 10, 25, 29, 30, 32

sample_mixtures_fixed_parameters, 18, 28

sample_mixtures_from_genotypes, 29

sample_offspring, 33, 34

sample_pedigree_genotypes, 17, 34

SMASH_to_wide_table, 27, 29, 30, 32, 35

stutter_type, 3, 36