Package: GenoPop (via r-universe)

October 13, 2024

Title Genotype Imputation and Population Genomics Efficiently from Variant Call Formatted (VCF) Files

Version 0.9.3

Description Tools for efficient processing of large, whole genome genotype data sets in variant call format (VCF). It includes several functions to calculate commonly used population genomic metrics and a method for reference panel free genotype imputation, which is described in the preprint Gurke & Mayer (2024) <doi:10.22541/au.172515591.10119928/v1>.

Date 2024-09-09

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.1

Imports Rsamtools, GenomicRanges, foreach, doParallel, parallel, missForest, IRanges

Suggests knitr, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Marie Gurke [aut, cre] (<https://orcid.org/0000-0001-9901-424X>)

Maintainer Marie Gurke <margurke@gmail.com>

Date/Publication 2024-09-11 16:10:06 UTC

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

RemoteUrl https://github.com/cran/GenoPop

RemoteRef HEAD

RemoteSha 20b1cfa4a4c7c3723e207eef018f4e981e7ad389

Contents

`Dxy` *Dxy*

Description

This function calculates the average number of nucleotide differences per site (Dxy) between two populations from a VCF file (Nei & Li, 1979 (https://doi.org/10.1073/pnas.76.10.5269)). Handling missing alleles at one site is equivalent to Korunes & Samuk, 2021 (https://doi.org/10.1111/1755-0998.13326). The function calculates the number of monomorphic sites using the sequence length and the number of variants in the VCF file. This assumes, that all sites not present in the VCF file are invariant sites, which will underestimate Pi, because of commonly done (and necessary) variant filtering. However, otherwise this calculation would only work with VCF files that include all monomorphic sites, which is quite unpractical for common use cases and will increase computational demands significantly. If you happen to know the number of filtered our sites vs the number of monomorphic sites, please use the number of monomorphic + the number of polymorphic (number of variants in your VCF) sites as the sequence length to get the most accurate estimation of Pi. (This does not work for the window mode of this function, which assumes the sequence length to be the window size.) For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
Dxy(
  vcf_path,
  pop1_individuals,
  pop2_individuals,
  seq_length,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
```

```
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
pop1_individuals	Vector of individual names belonging to the first population.
pop2_individuals	Vector of individual names belonging to the second population.
seq_length	Length of the sequence in number of bases, including monomorphic sites (used in batch mode only).
batch_size	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.

Value

In batch mode (no window_size or skip_size provided): The average number of nucleotide substitutions per site between the individuals of two populations (Dxy). In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'Dxy', representing the average nucleotide differences within each window.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
pop1_individuals <- c("tsk_0", "tsk_1", "tsk_2")
pop2_individuals <- c("tsk_3", "tsk_4", "tsk_5")
total_sequence_length <- 999299  # Total length of the sequence
# Batch mode example
dxy_value <- Dxy(vcf_file, pop1_individuals, pop2_individuals, total_sequence_length)
# Window mode example
dxy_windows <- Dxy(vcf_file, pop1_individuals, pop2_individuals, seq_length = total_sequence_length,
                   window_size = 100000, skip_size = 50000)
```

FixedSites *FixedSites*

Description

This function counts the number of sites fixed for the alternative allele ("1") in a VCF file. It processes the file in two modes: the entire file at once or in specified windows across the genome. For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach but tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
FixedSites(
  vcf_path,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  batch_size = 10000,
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`threads`	Number of threads to use for parallel processing.
`write_log`	Logical, indicating whether to write progress logs.
`logfile`	Path to the log file where progress will be logged.
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
`window_size`	Size of the window for windowed analysis in base pairs (optional). When specified, `skip_size` must also be provided.
`skip_size`	Number of base pairs to skip between windows (optional). Used in conjunction with `window_size` for windowed analysis.
`exclude_ind`	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Details

The function has two modes of operation:

1. Batch Mode: Processes the entire VCF file in batches to count the total number of fixed sites for the alternative allele. Suitable for a general overview of the entire dataset.

2. Window Mode: Processes the VCF file in windows of a specified size and skip distance. This mode is useful for identifying regions with high numbers of fixed sites, which could indicate selective sweeps or regions of low recombination.

Value

In batch mode (no window_size or skip_size provided): A single integer representing the total number of fixed sites for the alternative allele across the entire VCF file. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'FixedSites', representing the count of fixed sites within each window.

Examples

```
# Batch mode example
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
num_fixed_sites <- FixedSites(vcf_file)

# Window mode example
fixed_sites_df <- FixedSites(vcf_file, window_size = 100000, skip_size = 50000)
```

`Fst` *Fst*

Description

This function calculates the fixation index (Fst) between two populations from a VCF file using the method of Weir and Cockerham (1984). The formula used for this is equivalent to the one used in vcftools –weir-fst-pop (https://vcftools.sourceforge.net/man_latest.html). For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
Fst(
  vcf_path,
  pop1_individuals,
  pop2_individuals,
  weighted = FALSE,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
pop1_individuals	Vector of individual names belonging to the first population.
pop2_individuals	Vector of individual names belonging to the second population.
weighted	Logical, whether weighted Fst or mean Fst is returned (Default = FALSE (mean Fst is returned)).
batch_size	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.

Value

In batch mode (no window_size or skip_size provided): Fst value (either mean or weighted). In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'Fst', representing the fixation index within each window.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
pop1_individuals <- c("tsk_0", "tsk_1", "tsk_2")
pop2_individuals <- c("tsk_3", "tsk_4", "tsk_5")
# Batch mode example
fst_value <- Fst(vcf_file, pop1_individuals, pop2_individuals, weighted = TRUE)
# Window mode example
fst_windows <- Fst(vcf_file, pop1_individuals, pop2_individuals, weighted = TRUE,
                   window_size = 100000, skip_size = 50000)
```

GenoPop_Impute *GenoPop-Impute*

Description

Performs imputation of missing genomic data in batches using the missForest (Stekhoven & Bühlmanm, 2012) algorithm. This function reads VCF files, divides it into batches of a fixed number of SNPs, applies the missForest algorithm to each batch, and writes the results to a new VCF file, which will be returned bgzipped and tabix indexed. The choice of the batch size is critical for balancing accuracy and computational demand. We found that a batch size of 500 SNPs is the most accurate for recombination rates typical of mammalians. For on average higher recombination rates (> 5 cM/Mb) we recommend a batch size of 100 SNPs.

Usage

```
GenoPop_Impute(
  vcf_path,
  output_vcf,
  batch_size = 1000,
  maxiter = 10,
  ntree = 100,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt"
)
```

Arguments

vcf_path	Path to the input VCF file.
output_vcf	Path for the output VCF file with imputed data.
batch_size	Number of SNPs to process per batch (default: 500).
maxiter	Number of improvement iterations for the random forest algorithm (default: 10).
ntree	Number of decision trees in the random forest (default: 100).
threads	Number of threads used for computation (default: 1).
write_log	If TRUE, writes a log file of the process (advised for large datasets).
logfile	Path to the log file, used if write_log is TRUE.

Value

Path to the output VCF file with imputed data.

Examples

```
vcf_file <- system.file("tests/testthat/sim_miss.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim_miss.vcf.gz.tbi", package = "GenoPop")
output_file <- tempfile(fileext = ".vcf")
GenoPop_Impute(vcf_file, output_vcf = output_file, batch_size = 500)
```

`Heterozygosity` *Heterozygosity Rate*

Description

This function calculates the rate of heterozygosity for samples in a VCF file. (The proportion of heterozygote genotypes.) For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
Heterozygosity(
  vcf_path,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
`threads`	Number of threads to use for parallel processing.
`write_log`	Logical, indicating whether to write progress logs.
`logfile`	Path to the log file where progress will be logged.
`window_size`	Size of the window for windowed analysis in base pairs (optional). When specified, `skip_size` must also be provided.
`skip_size`	Number of base pairs to skip between windows (optional). Used in conjunction with `window_size` for windowed analysis.
`exclude_ind`	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): Observed heterozygosity rate averaged over all loci. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'Ho', representing the observed heterozygosity rate within each window.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
# Batch mode example
Ho <- Heterozygosity(vcf_file)
# Window mode example
Ho_windows <- Heterozygosity(vcf_file, window_size = 100000, skip_size = 50000)
```

OneDimSFS *OneDimSFS*

Description

This function calculates a one-dimensional site frequency spectrum from a VCF file. It processes the file in batches for efficient memory usage. The user can decide between a folded or unfolded spectrum.

Usage

```
OneDimSFS(
  vcf_path,
  folded = FALSE,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  exclude_ind = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
folded	Logical, deciding if folded (TRUE) or unfolded (FALSE) SFS is returned.
batch_size	The number of variants to be processed in each batch (default of 10,000 should be suitable for most use cases).
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
exclude_ind	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

Site frequency spectrum as a named vector

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
sfs <- OneDimSFS(vcf_file, folded = FALSE)
```

Pi *Pi*

Description

This function calculates the nucleotide diversity (Pi) for a sample in a VCF file as defined by Nei & Li, 1979 (https://doi.org/10.1073/pnas.76.10.5269). The formula used for this is equivalent to the one used in vcftools –window-pi (https://vcftools.sourceforge.net/man_latest.html). Handling missing alleles at one site is equivalent to Korunes & Samuk, 2021 (https://doi.org/10.1111/1755-0998.13326). The function calculates the number of monomorphic sites using the sequence length and the number of variants in the VCF file. This assumes, that all sites not present in the VCF file are invariant sites, which will underestimate Pi, because of commonly done (and necessary) variant filtering. However, otherwise this calculation would only work with VCF files that include all monomorphic sites, which is quite unpractical for common use cases and will increase computational demands significantly. If you happen to know the number of filtered our sites vs the number of monomorphic sites, please use the number of monomorphic + the number of polymorphic (number of variants in your VCF) sites as the sequence length to get the most accurate estimation of Pi. (This does not work for the window mode of this function, which assumes the sequence length to be the window size.) For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
Pi(
  vcf_path,
  seq_length,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`seq_length`	Total length of the sequence in number of bases (used in batch mode only).
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).

threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.
exclude_ind	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): Nucleotide diversity (Pi) across the sequence. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'Pi', representing the nucleotide diversity within each window.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
total_sequence_length <- 999299  # Total length of the sequence in vcf
# Batch mode example
pi_value <- Pi(vcf_file, total_sequence_length)
# Window mode example
pi_windows <- Pi(vcf_file, seq_length = total_sequence_length,
                 window_size = 100000, skip_size = 50000)
```

PrivateAlleles *PrivateAlleles*

Description

This function calculates the number of private alleles in two populations from a VCF file. (Alleles which are not present in the other popualtion.) It processes the file in batches or specified windows across the genome. For batch processing, it uses process_vcf_in_batches. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (process_vcf_in_windows).

Usage

```
PrivateAlleles(
  vcf_path,
  pop1_individuals,
  pop2_individuals,
  threads = 1,
  write_log = FALSE,
```

```
  logfile = "log.txt",
  batch_size = 10000,
  window_size = NULL,
  skip_size = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
pop1_individuals	Vector of individual names belonging to the first population.
pop2_individuals	Vector of individual names belonging to the second population.
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
batch_size	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.

Value

In batch mode (no window_size or skip_size provided): A list containing the number of private alleles for each population. In window mode (window_size and skip_size provided): A list of data frames, each with columns 'Chromosome', 'Start', 'End', 'PrivateAllelesPop1', and 'PrivateAllelesPop2', representing the count of private alleles within each window for each population.

Examples

```
# Batch mode example
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
pop1_individuals <- c("tsk_0", "tsk_1", "tsk_2")
pop2_individuals <- c("tsk_3", "tsk_4", "tsk_5")
private_alleles <- PrivateAlleles(vcf_file, pop1_individuals, pop2_individuals)

# Window mode example
private_alleles_windows <- PrivateAlleles(vcf_file, pop1_individuals, pop2_individuals,
                                          window_size = 100000, skip_size = 50000)
```

SegregatingSites *SegregatingSites*

Description

This function counts the number of polymorphic or segregating sites (sites not fixed for the alternative allele) in a VCF file. It processes the file in batches or specified windows across the genome. For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
SegregatingSites(
  vcf_path,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  batch_size = 10000,
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`threads`	Number of threads to use for parallel processing.
`write_log`	Logical, indicating whether to write progress logs.
`logfile`	Path to the log file where progress will be logged.
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
`window_size`	Size of the window for windowed analysis in base pairs (optional). When specified, `skip_size` must also be provided.
`skip_size`	Number of base pairs to skip between windows (optional). Used in conjunction with `window_size` for windowed analysis.
`exclude_ind`	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): A single integer representing the total number of polymorphic sites across the entire VCF file. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'PolymorphicSites', representing the count of polymorphic sites within each window.

Examples

```
# Batch mode example
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
num_polymorphic_sites <- SegregatingSites(vcf_file)

# Window mode example
polymorphic_sites_df <- SegregatingSites(vcf_file, window_size = 100000, skip_size = 50000)
```

SingletonSites *SingletonSites*

Description

This function counts the number of singleton sites (sites where a minor allele occurs only once in the sample) in a VCF file. It processes the file in batches or specified windows across the genome. For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
SingletonSites(
  vcf_path,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  batch_size = 10000,
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`threads`	Number of threads to use for parallel processing.
`write_log`	Logical, indicating whether to write progress logs.
`logfile`	Path to the log file where progress will be logged.
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
`window_size`	Size of the window for windowed analysis in base pairs (optional). When specified, `skip_size` must also be provided.
`skip_size`	Number of base pairs to skip between windows (optional). Used in conjunction with `window_size` for windowed analysis.

exclude_ind Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): A single integer representing the total number of singleton sites across the entire VCF file. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'SingletonSites', representing the count of singleton sites within each window.

Examples

```
# Batch mode example
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
num_singleton_sites <- SingletonSites(vcf_file)

# Window mode example
vcf_path <- "path/to/vcf/file"
singleton_sites_df <- SingletonSites(vcf_file, window_size = 100000, skip_size = 50000)
```

TajimasD *TajimasD*

Description

This function calculates Tajima's D statistic for a given dataset (Tajima, 1989 (10.1093/genetics/123.3.585)). The formula used for this is equivalent to the one used in vcftools –TajimaD (https://vcftools.sourceforge.net/man_latest.html). The function calculates the number of monomorphic sites using the sequence length and the number of variants in the VCF file. This assumes, that all sites not present in the VCF file are invariant sites, which will underestimate Pi, because of commonly done (and necessary) variant filtering. However, otherwise this calculation would only work with VCF files that include all monomorphic sites, which is quite unpractical for common use cases and will increase computational demands significantly. If you happen to know the number of filtered our sites vs the number of monomorphic sites, please use the number of monomorphic + the number of polymorphic (number of variants in your VCF) sites as the sequence length to get the most accurate estimation of Pi. (This does not work for the window mode of this function, which assumes the sequence length to be the window size.) For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
TajimasD(
  vcf_path,
  seq_length,
```

```
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
seq_length	Total length of the sequence in number of bases (used in batch mode only).
batch_size	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.
exclude_ind	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): Tajima's D value. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'TajimasD', representing Tajima's D within each window.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
total_sequence_length <- 999299  # Total length of the sequence
# Batch mode example
tajimas_d <- TajimasD(vcf_file, total_sequence_length)
# Window mode example
tajimas_d_windows <- TajimasD(vcf_file, seq_length = total_sequence_length,
                              window_size = 100000, skip_size = 50000)
```

TwoDimSFS *TwoDimSFS*

Description

This function calculates a two-dimensional site frequency spectrum from a VCF file for two populations. It processes the file in batches for efficient memory usage. The user can decide between a folded or unfolded spectrum.

Usage

```
TwoDimSFS(
  vcf_path,
  pop1_individuals,
  pop2_individuals,
  folded = FALSE,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  exclude_ind = NULL
)
```

Arguments

vcf_path	Path to the VCF file.
pop1_individuals	Vector of individual names belonging to the first population.
pop2_individuals	Vector of individual names belonging to the second population.
folded	Logical, deciding if folded (TRUE) or unfolded (FALSE) SFS is returned.
batch_size	The number of variants to be processed in each batch (default of 10,000 should be suitable for most use cases).
threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
exclude_ind	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

Two-dimensional site frequency spectrum as a matrix.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
pop1_individuals <- c("tsk_0", "tsk_1", "tsk_2")
pop2_individuals <- c("tsk_3", "tsk_4", "tsk_5")
sfs_2d <- TwoDimSFS(vcf_file, pop1_individuals, pop2_individuals, folded = TRUE)
```

`WattersonsTheta` *WattersonsTheta*

Description

This function calculates Watterson's Theta, a measure for neutrality, from a VCF file (Watterson, 1975 (https://doi.org/10.1016/0040-5809(75)90020-9)). The function calculates the number of monomorphic sites using the sequence length and the number of variants in the VCF file. This assumes, that all sites not present in the VCF file are invariant sites, which will underestimate Pi, because of commonly done (and necessary) variant filtering. However, otherwise this calculation would only work with VCF files that include all monomorphic sites, which is quite unpractical for common use cases and will increase computational demands significantly. If you happen to know the number of filtered our sites vs the number of monomorphic sites, please use the number of monomorphic + the number of polymorphic (number of variants in your VCF) sites as the sequence length to get the most accurate estimation of Pi. (This does not work for the window mode of this function, which assumes the sequence length to be the window size.) For batch processing, it uses `process_vcf_in_batches`. For windowed analysis, it uses a similar approach tailored to process specific genomic windows (`process_vcf_in_windows`).

Usage

```
WattersonsTheta(
  vcf_path,
  seq_length,
  batch_size = 10000,
  threads = 1,
  write_log = FALSE,
  logfile = "log.txt",
  window_size = NULL,
  skip_size = NULL,
  exclude_ind = NULL
)
```

Arguments

`vcf_path`	Path to the VCF file.
`seq_length`	The length of the sequence in the data set (used in batch mode only).
`batch_size`	The number of variants to be processed in each batch (used in batch mode only, default of 10,000 should be suitable for most use cases).

threads	Number of threads to use for parallel processing.
write_log	Logical, indicating whether to write progress logs.
logfile	Path to the log file where progress will be logged.
window_size	Size of the window for windowed analysis in base pairs (optional). When specified, skip_size must also be provided.
skip_size	Number of base pairs to skip between windows (optional). Used in conjunction with window_size for windowed analysis.
exclude_ind	Optional vector of individual IDs to exclude from the analysis. If provided, the function will remove these individuals from the genotype matrix before applying the custom function. Default is NULL, meaning no individuals are excluded.

Value

In batch mode (no window_size or skip_size provided): Watterson's theta value normalized by the sequence length. In window mode (window_size and skip_size provided): A data frame with columns 'Chromosome', 'Start', 'End', and 'WattersonsTheta', representing Watterson's theta within each window normalized by the window length.

Examples

```
vcf_file <- system.file("tests/testthat/sim.vcf.gz", package = "GenoPop")
index_file <- system.file("tests/testthat/sim.vcf.gz.tbi", package = "GenoPop")
total_sequence_length <- 999299  # Total length of the sequence
# Batch mode example
wattersons_theta <- WattersonsTheta(vcf_file, total_sequence_length)
# Window mode example
wattersons_theta_windows <- WattersonsTheta(vcf_file, seq_length = total_sequence_length,
                                            window_size = 100000, skip_size = 50000)
```

Index