Package ‘BIGr’

June 27, 2024

Title (B)reeding (I)nsight (G)enomics Functions for Polypoid and Diploid Species

Version 0.3.1

Author Alexander M. Sandercock, Josue Chinchilla-Vargas, Shufen Chen, Manoj Sapkota, Meng Lin, Dongyan Zhao, and Breeding Insight Team

Maintainer Alexander M. Sandercock <ams866@cornell.edu>

Description This package contains the functions developed within Breeding Insight to analyze diploid and polyploid breeding and genetic data.

License Apache License 2.0

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 4.0.0)

Imports doParallel,
dplyr,
foreach,
Rdpack (>= 0.7),
readr (>= 2.1.5),
reshape2 (>= 1.4.4),
tidyr (>= 1.3.1),
vcfR (>= 1.15.0)

RdMacros Rdpack

R topics documented:

calculate_Het *Calculate Observed Heterozygosity from a Genotype Matrix*

Description

This function calculates the observed heterozygosity from a genotype matrix. It assumes that the samples are the columns, and the genomic markers are in rows. Missing data should be set as NA, which will then be ignored for the calculations. All samples must have the same ploidy.

Usage

```
calculate_Het(geno, ploidy)
```

Arguments

geno	Genotype matrix or data.frame
ploidy	The ploidy of the species being analyzed

Value

A dataframe of observed heterozygosity values for each sample

calculate_MAF *Calculate Minor Allele Frequency from a Genotype Matrix*

Description

This function calculates the allele frequency and minor allele frequency from a genotype matrix. It assumes that the Samples are the columns, and the genomic markers are in rows. Missing data should be set as NA, which will then be ignored for the calculations. All samples must have the same ploidy.

Usage

```
calculate_MAF(df, ploidy)
```

Arguments

df	Genotype matrix or data.frame
ploidy	The ploidy of the species being analyzed

Value

A dataframe of AF and MAF values for each marker

capture_diversity.Gmat

Estimate Minimum Number of Individuals to Sample to Capture Population Genomic Diversity (Genotype Matrix)

Description

This function can be used to estimate the number of individuals to sample from a population in order to capture a desired percentage of the genomic diversity. It assumes that the samples are the columns, and the genomic markers are in rows. Missing data should be set as NA, which will then be ignored for the calculations. All samples must have the same ploidy. This function was adapted from a previously developed Python method (Sandercock et al., 2023) (https://github.com/alex-sandercock/Capturing_genomic_diversity/)

Usage

```
capture_diversity.Gmat(
  df,
  ploidy,
  r2_threshold = 0.9,
  iterations = 10,
  sample_list = NULL,
  parallel = FALSE,
  save.result = TRUE
)
```

Arguments

df	Genotype matrix or data.frame with the count of alternate alleles (0=homozygous reference, 1 = heterozygous, 2 = homozygous alternate)
ploidy	The ploidy of the species being analyzed
r2_threshold	The ratio of diversity to capture (default = 0.9)
iterations	The number of iterations to perform to estimate the average result (default = 10)
sample_list	The list of samples to subset from the dataset (optional)
parallel	Run the analysis in parallel (True/False) (default = FALSE)
save.result	Save the results to a .txt file? (default = TRUE)

Value

A data.frame with minimum number of samples required to match or exceed the input ratio

References

Sandercock, A. M., Westbrook, J. W., Zhang, Q., & Holliday, J. A. (2024). The road to restoration: Identifying and conserving the adaptive legacy of American chestnut. PNAS (in press).

check_ped *Evaluate Pedigree File for Accuracy*

Description

Check a pedigree file for accuracy and output suspected errors

Usage

```
check_ped(ped.file)
```

Arguments

ped.file	path to pedigree text file. The pedigree file is a 3-column pedigree tab separated file with columns labeled as id sire dam in any order

Details

check_ped takes a 3-column pedigree tab separated file with columns labeled as id sire dam in any order and checks for:

- Ids that appear more than once in the id column
- Ids that appear in both sire and dam columns
- Direct (e.g. parent is a offspring of his own daughter) and indirect (e.g. a great grandparent is son of its granchild) dependencies within the pedigree.
- Individuals included in the pedigree as sire or dam but not on the id column and reports them back with unknown parents (0).

When using check_ped, do a first run to check for repeated ids and parents that appear as sire and dam. Once these errors are cleaned run the function again to check for dependencies as this will provide the most accurate report.

Note: This function does not change the input file but prints any errors found in the console.

Value

A list of dataframes of error types, and the output printed to the console

Examples

```
##Get list with a dataframe for each error type
#ped_errors <- check_ped(ped.file = "example_ped.txt")

##Access the "messy parents" dataframe result
#ped_errors$messy_parents

##Get list of sample IDs with messy parents error
#messy_parent_ids <- ped_errors$messy_parents$id
#print(messy_parent_ids)
```

dosage2vcf *Convert DArTag Dosage and Counts to VCF*

Description

This function will convert the DArT Dosage Report and Counts files to VCF format

Usage

```
dosage2vcf(dart.report, dart.counts, ploidy, output.file)
```

Arguments

`dart.report`	Path to the DArT dosage report .csv file. Typically contains "Dosage Report" in the file name.
`dart.counts`	Path to the DArT counts .csv file. Typically contains "Counts" in the file name.
`ploidy`	The ploidy of the species being analyzed
`output.file`	output file name and path

Details

This function will convert the Dosage Report and Counts files from DArT into a VCF file. These two files are received directly from DArT for a given sequencing project. The output file will be saved to the location and with the name that is specified. The VCF format is v4.3

Value

A vcf file

Examples

```
## Use file paths for each file on the local system

#The files are directly from DArT for a given sequencing project.
#The are labeled with Dosage_Report or Counts in the file names.

#dosage2vcf(dart.report = "example_dart_Dosage_Report.csv",
 #          dart.counts = "example_dart_Counts.csv",
 #          ploidy = 2,
 #          output.file = "name_for_vcf")

##The function will output the converted VCF using information from the DArT files
```

`dosage_ratios` *Calculate the Percentage of Each Dosage Value*

Description

This function calculates the percentage of each dosage value within a genotype matrix. It assumes that the samples are the columns, and the genomic markers are in rows. Missing data should be set as NA, which will then be ignored for the calculations. All samples must have the same ploidy.

Usage

```
dosage_ratios(data, ploidy)
```

Arguments

`data`	Genotype matrix or data.frame
`ploidy`	The ploidy of the species being analyzed

Value

A data.frame with percentages of dosage values in the genotype matrix

`filterVCF` *Filter a VCF file*

Description

This function will filter a VCF file or vcfR object and export the updated version

Usage

```
filterVCF(
  vcf.file,
  filter.OD = NULL,
  filter.BIAS.min = NULL,
  filter.BIAS.max = NULL,
  filter.DP = NULL,
  filter.MPP = NULL,
  filter.PMC = NULL,
  filter.MAF = NULL,
  filter.SAMPLE.miss = NULL,
  filter.SNP.miss = NULL,
  ploidy,
  output.file = NULL
)
```

Arguments

`vcf.file` vcfR object or path to VCF file. Can be unzipped (.vcf) or gzipped (.vcf.gz).

`filter.OD` Updog filter

`filter.BIAS.min` Updog filter (requires a value for both BIAS.min and BIAS.max)

`filter.BIAS.max` Updog filter (requires a value for both BIAS.min and BIAS.max)

`filter.DP` Total read depth at each SNP filter

`filter.MPP` Updog filter

`filter.PMC` Updog filter

`filter.MAF` Minor allele frequency filter

`filter.SAMPLE.miss` Sample missing data filter

`filter.SNP.miss` SNP missing data filter

`ploidy` The ploidy of the species being analyzed

`output.file` output file name (optional). If no output.file name provided, then a vcfR object will be returned.

Details

This function will input a VCF file or vcfR object and filter based on the user defined options. The output file will be saved to the location and with the name that is specified. The VCF format is v4.3

Value

A gzipped vcf file

Examples

```
## Use file paths for each file on the local system


#filterVCF(vcf.file = "example_dart_Dosage_Report.csv",
 #          filter.OD = 0.5,
 #          ploidy = 2,
 #          output.file = "name_for_vcf")

##The function will output the filtered VCF to the current working directory
```

flip_dosage *Switch Dosage Values from a Genotype Matrix*

Description

This function converts the dosage count values to the opposite value. This is primarily used when converting dosage values from reference based (0 = homozygous reference) to alternate count based (0 = homozygous alternate). It assumes that the Samples are the columns, and the genomic markers are in rows. Missing data should be set as NA, which will then be ignored for the calculations. All samples must have the same ploidy.

Usage

```
flip_dosage(df, ploidy, is.reference = TRUE)
```

Arguments

`df`	Genotype matrix or data.frame
`ploidy`	The ploidy of the species being analyzed
`is.reference`	The dosage calls value is based on the count of reference alleles (TRUE/FALSE)

Value

A genotype matrix

get_countsMADC *Obtain Read Counts from MADC File*

Description

This function takes the MADC file as input and retrieves the ref and alt counts for each sample, and converts them to ref, alt, and size(total count) matrices for dosage calling tools. At the moment, only the read counts for the Ref and Alt target loci are obtained while the additional loci are ignored.

Usage

```
get_countsMADC(madc_file)
```

Arguments

`madc_file`	Path to MADC file

Value

A list of read count matrices for reference, alternate, and total read count values

updog2vcf *Export Updog Results as VCF*

Description

This function will convert an Updog output to a VCF file

Usage

```
updog2vcf(multidog.object, ploidy, output.file)
```

Arguments

`multidog.object`	updog output object with class "multidog" from dosage calling
`ploidy`	The ploidy of the species being analyzed
`output.file`	output file name and path

Details

When performing dosage calling for multiple SNPs using Updog, the output file contains information for all loci and all samples. This function will convert the updog output file to a VCF file, while retaining the information for the values that are commonly used to filter low quality and low confident dosage calls.

Value

A vcf file

References

Updog R package

Index