# Package 'polyBreedR'

January 3, 2024

**Title** Genomics-assisted breeding for polyploids (and diploids)

**Version** 0.37

**Author** Jeffrey B. Endelman

**Maintainer** Jeffrey Endelman <endelman@wisc.edu>

**Description** Genomics-assisted breeding for polyploids (and diploids)

**Depends** R (>= 4.0)

**License** GPL-3

**LazyData** true

**RoxygenNote** 7.2.3

**Encoding** UTF-8

**Imports** AGHmatrix, ggplot2, ggrepel, pedigree, grDevices, utils, tidyr, Matrix, methods, rlang, updog, randomForest, vcfR, rrBLUP, data.table

**Suggests** knitr, rmarkdown, asreml

**VignetteBuilder** knitr

## R topics documented:

---

`ADsplit` *Extract read counts from AD string*

---

## Description

Extract read counts from AD string

## Usage

```
ADsplit(AD, ALT, n.core = 1)
```

## Arguments

| | |
|---|---|
| `AD` | array of AD strings |
| `ALT` | TRUE or FALSE (= REF) |
| `n.core` | number of cores |

## Value

integer data with same dimensions as AD

---

`array2vcf` *SNP array to VCF*

---

## Description

Converts output from Genome Studio (Final Report or Wide) to VCF

## Usage

```
array2vcf(array.file, map.file, model.file = NULL, ploidy, vcf.file)
```

## Arguments

| | |
|---|---|
| `array.file` | name of input file with SNP array allele intensities |
| `map.file` | vcf file with map positions for the markers |
| `model.file` | normal mixture model parameters for genotype calls |
| `ploidy` | sample ploidy, for use with `model.file` |
| `vcf.file` | output vcf file |

## Details

Auto-detects whether the input file is a Genome Studio Final Report, which is a "long" format with 9-row header, or in "wide" format, where all the data for each marker is one row. XY values are multiplied by 100

Genotype calls will attempt to be imported from the GS Final Report when `model.file=NULL`. For diploids, columns named "Allele 1 - AB" and "Allele 2 - AB" are expected. For tetraploids, a single column named "Alleles - AB" is expected.

It is assumed that the parameters in `model.file` lead to genotype calls for the dosage of allele B. For a VCF file, genotype calls need to be based on the dosage of ALT. By default, it is assumed that A is the REF allele. For variants where B is REF, include "REF=B" as INFO in the VCF `map.file`.

---

`A_mat` *Additive relationship matrix from pedigree*

---

## Description

Additive relationship matrix from pedigree

## Usage

```
A_mat(ped, ploidy, order.ped = TRUE)
```

## Arguments

| | |
|---|---|
| `ped` | Pedigree in three column format: id, mother, father |
| `ploidy` | 2 or 4 |
| `order.ped` | TRUE/FALSE does the pedigree need to be ordered so that progeny follow parents |

## Details

This is a wrapper that prepares the pedigree in the format required for R package `AGHmatrix` by Amadeu et al. (2016) (cite them if you use this function). A random bivalents model for tetraploid meiosis is assumed.

## Value

Additive relationship matrix (dim: indiv x indiv)

## References

Amadeu et al. (2016) Plant Genome 9, doi:10.3835/plantgenome2016.01.0009

## check_ploidy *Check ploidy*

### Description

Fraction of simplex or triplex markers

### Usage

```
check_ploidy(geno, map)
```

### Arguments

| | |
|---|---|
| geno | Genotype matrix (markers x indiv) |
| map | Data frame with marker map (Marker, Chrom, Position) |

### Details

For every indiv in the genotype matrix, the fraction of markers per chromosome called as simplex or triplex is calculated, which should be low for diploids. A small amount of missing genotype data can be tolerated.

### Value

List containing

**mat** Matrix (indiv x chrom) of results

**plot** ggplot2 barplot

## check_trio *Check markers for parent-offspring trio*

### Description

Check markers for parent-offspring trio

### Usage

```
check_trio(parentage, geno, ploidy)
```

### Arguments

| | |
|---|---|
| parentage | Data frame with three columns: id, mother, father |
| geno | Matrix of allele dosages: markers x indiv |
| ploidy | 2 or 4 |

### Details

Computes the percentage of markers at which the two parents and offspring have incompatible allele dosages (for tetraploids, the random bivalents model is used). For dihaploid offspring of a single tetraploid parent, use `ploidy = 4` and "haploid" for the father in `parentage`, as well as a diploid (0,1,2) genotype for the offspring. A small amount of missing genotype data can be tolerated.

### Value

Data frame with the percentage of incompatible markers for each trio

---

## dart2vcf *Convert DArTag to VCF*

---

### Description

Convert DArTag to VCF

### Usage

```
dart2vcf(counts.file, dosage.file, vcf.file, ploidy, first.data.row = 9)
```

### Arguments

| | |
|---|---|
| `counts.file` | DArTag collapsed counts file |
| `dosage.file` | DArTag dosage file |
| `vcf.file` | name of VCF output file |
| `ploidy` | ploidy |
| `first.data.row` | default is 9 for DArTag format |

### Details

Two input files expected. counts.file is the two-row collapsed counts file, whereas dosage.file has one row per target, with chrom and position in columns 4 and 5. DArT reports dosage of REF, whereas VCF standard is based on dosage of ALT. The dosage is exported as GT field in VCF.

Duplicate samples are renamed by appending the "Target ID".

---

## gbs *Genotype calls for GBS*

---

### Description

Genotype calls for genotype-by-sequencing (GBS) data

### Usage

```
gbs(in.file, out.file, ploidy, bias = TRUE, n.core = 1, silent = FALSE)
```

### Arguments

| | |
|---|---|
| in.file | VCF input file |
| out.file | VCF output file |
| ploidy | ploidy |
| bias | TRUE/FALSE, whether to estimate allelic bias |
| n.core | number of cores |
| silent | TRUE/FALSE |

### Details

VCF input file must contain AD field. Variants with more than 2 alleles are coerced to zero DP, so better to filter them out first.

Posterior mode and mean genotypes are added as GT and DS fields. GQ is also added based on probability of posterior mode. Binomial calculation uses R/updog package (Gerard et al. 2018) with "norm" prior. Previous INFO is discarded; adds NS, DP.AVG, AF.GT, AB, OD, SE.

### Value

nothing

---

## geno_call *Genotype calls*

---

### Description

Genotype calls based on a normal mixture model

### Usage

```
geno_call(
  data,
  filename,
  model.ploidy = 4L,
  sample.ploidy = 4L,
  min.posterior = 0,
  transform = TRUE
)
```

### Arguments

| | |
|---|---|
| data | matrix (markers x id) of input values for the normal mixture model |
| filename | CSV filename with the model parameters |
| model.ploidy | 2 or 4 (default) |
| sample.ploidy | 2 or 4 (default) |
| min.posterior | minimum posterior probability (default 0) for genotype call |
| transform | TRUE (default) or FALSE whether to apply arcsin square root transformation |

### Details

The first column of the CSV input file should be the SNP ID, followed by columns for the normal distribution means, standard deviations, and mixture probabilities. Genotype calls are based on the maximum a posteriori (MAP) method. If the posterior probability of the MAP genotype is less than `min.posterior`, then `NA` is returned for that sample. By default, an arcsin square root transformation is applied to the input values to match the approach used by R package fitPoly. To use a tetraploid mixture model for diploid samples, set `sample.ploidy = 2` and `model.ploidy = 4`.

### Value

matrix of allele dosages (0,1,2,..ploidy) with dimensions markers x individuals

---

## `get_pedigree` *Generate pedigree*

---

### Description

Generate pedigree for a set of individuals

### Usage

```
get_pedigree(id, pedfile, delim = ",", na.string = "NA", trim = TRUE)
```

### Arguments

| | |
|---|---|
| `id` | Vector of names of individuals |
| `pedfile` | Name of pedigree file |
| `delim` | Delimiter for the pedigree file (default is "," for CSV) |
| `na.string` | String used for NA in the pedigree file (default is "NA") |
| `trim` | TRUE/FALSE whether to trim pedigree (see Details) |

### Details

Finds ancestors of individuals in a three-column pedigree file (id,mother,father). The id column can be the identifier for an individual or cross. String matches must be exact or based on the naming convention crossID-progenyID. The returned pedigree is ordered using R package `pedigree` so that offspring follow parents. When `trim` is TRUE (default), the pedigree is trimmed to remove ancestors with only one offspring (which are not needed to compute the pedigree relationship matrix).

### Value

Data frame with columns id, mother, father

## GT2DS *Convert GT to ALT allele dosage (DS)*

### Description

Convert GT to ALT allele dosage (DS)

### Usage

```
GT2DS(GT, diploidize = FALSE, n.core = 1)
```

### Arguments

| | |
|---|---|
| GT | GT string |
| diploidize | TRUE/FALSE |
| n.core | number of cores |

### Details

If diploidize is TRUE, data are recoded as a diploid 0,1,2.

### Value

integer data with same dimensions as GT

## GvsA *Plot G vs. A*

### Description

Plot marker-based vs. pedigree-based additive relationship coefficients

### Usage

```
GvsA(
  parentage,
  G,
  A,
  filename = NULL,
  thresh.G = Inf,
  thresh.A = 0.5,
  Gmax = NULL,
  Amax = NULL
)
```

### Arguments

| | |
|---|---|
| parentage | Data frame of individuals to plot, with 3 columns: id,mother,father |
| G | Genomic relationship matrix |
| A | Pedigree relationship matrix |
| filename | Name of PDF file to save the results (optional for one individual) |
| thresh.G | Threshold above which names are displayed (default Inf) |
| thresh.A | Threshold above which names are displayed (default 0.5) |
| Gmax | Upper limit for y-axis for plotting. If NULL, maximum value in G is used. |
| Amax | Upper limit for x-axis for plotting. If NULL, maximum value in A is used. |

### Details

Useful for finding and correcting pedigree errors. If the G or A coefficient for an individual exceeds the threshold, its name is displayed in the figure. If parentage contains one individual, by default a ggplot2 variable will be returned, but the result can also be written to file. If multiple individuals are present, a filename is required.

---

## G_mat *Additive genomic relationships*

---

### Description

Relationship matrix for additive effects with bi-allelic markers

### Usage

```
G_mat(geno, ploidy)
```

### Arguments

| | |
|---|---|
| geno | Matrix of allele dosages (markers x indiv) |
| ploidy | Any even integer (2,4,6,...) |

### Details

Additive effects are based on the traditional orthogonal decomposition of genetic variance in panmictic populations (Fisher 1918; Kempthorne 1957; Endelman et al. 2018). Missing genotype data is replaced with the population mean.

### Value

G matrix

### References

Fisher (1918) Trans. Roy. Soc. Edin. 52:399-433.

Kempthorne (1957) An Introduction to Genetic Statistics.

Endelman et al. (2018) Genetics 209:77-87.

`impute` *Impute missing data for bi-allelic markers*

## Description

Impute missing data for bi-allelic markers

## Usage

```
impute(
  in.file,
  out.file,
  ploidy,
  method,
  geno,
  min.DP = 1,
  max.missing,
  params = NULL,
  n.core = 1
)
```

## Arguments

| | |
|---|---|
| `in.file` | VCF input file |
| `out.file` | VCF output file |
| `ploidy` | ploidy |
| `method` | One of the following: "pop","EM","RF" |
| `geno` | One of the following: "GT","DS" |
| `min.DP` | genotypes below this depth are set to missing (default=1) |
| `max.missing` | remove markers above this threshold, as proportion of population |
| `params` | list of method-specific parameters |
| `n.core` | multicore processing |

## Details

Assumes input file is sorted by position. Markers with no genetic variance are removed.

`method="pop"` imputes with the population mean for `geno="DS"` and population mode for `geno="GT"`.

`method="EM"` uses parameter "tol" (default is 0.02, see rrBLUP A.mat documentation). Imputed values are truncated if needed to fall in the interval [0,ploidy].

`method="RF"` uses parameters "ntree" (default 100) for number of trees and "nflank" (default 100) for the number of flanking markers (on each side) to use as predictors. Because RF first uses EM to generate a complete dataset, parameter "tol" is also recognized.

impute_L2H *Impute from low to high density markers by Random Forest*

## Description

Impute from low to high density markers by Random Forest

## Usage

```
impute_L2H(
  high.file,
  low.file,
  out.file,
  params = list(),
  exclude = NULL,
  n.core = 1
)
```

## Arguments

| | |
|---|---|
| high.file | name of high density file |
| low.file | name of low density file |
| out.file | name of CSV output file for imputed data |
| params | list of parameters (see Details) |
| exclude | optional, vector of high density samples to exclude |
| n.core | multicore processing |

## Details

Argument params is a list with three named elements: format, n.tree, n.mark. format can have values "GT" (integer dosage) or "DS" (real numbers between 0 and ploidy). Classification trees are used for GT and regression trees for DS. n.tree is the number of trees (default = 100). n.mark is the number of markers to use as predictors (default = 100), chosen based on minimum distance to the target.

The exclude argument is useful for cross-validation.

Both VCF and CSV are allowable input file formats–they are recognized based on the file extension. For CSV, the first three columns should be marker, chrom, pos. The output file is CSV.

Any missing data are imputed separately for each input file at the outset, using the population mean (DS) or mode (GT) for each marker.

## Value

matrix of OOB error with dimensions markers x trees

## impute_PO *Impute from low to high density markers with PolyOrigin*

### Description

Impute from low to high density markers by linkage analysis with PolyOrigin

### Usage

```
impute_PO(
  ped.file,
  high.file,
  low.file,
  low.format = "GT",
  out.file,
  n.thread = 1
)
```

### Arguments

| | |
|---|---|
| ped.file | pedigree file for progeny (must follow PO format) |
| high.file | name of high density file with phased parents |
| low.file | name of low density VCF file with progeny |
| low.format | either "GT" (default) or "AD" |
| out.file | name of CSV output file |

### Details

You must have separately installed PolyOrigin and Julia for this function to work.

The high density file contains phased parental genotypes in PolyOrigin format. The first 3 columns are the genetic map in cM: marker, chrom, position. To output imputed data with physical rather than genetic map positions, including a fourth column named "bp". Subsequent columns are the phased parental genotypes.

VCF is assumed for the low-density file. The pedigree file must follow PolyOrigin format.

The output file contains the posterior maximum geontypes.

A temporary directory "tmp" is created to store intermediate files and then deleted.

## madc *Multi-Allelic Haplotype Counts from DArTag*

### Description

Multi-Allelic Haplotype Counts from the DArTag MADC (Missing Allele Discovery Count) file

### Usage

```
madc(madc.file, marker = "CDF1")
```

### Arguments

| | |
|---|---|
| madc.file | MADC filename |
| marker | Name of marker ("CDF1" is only option so far) |

### Details

Due to multi-allelism, for some trait markers a correct interpretation is not possible using the collapsed counts file; the MADC file is needed. Currently, the only marker implemented is CDF1 for potato DArTag. The CDF1 marker detects the 2C, 2T, and 4 alleles, and all other haplotypes are treated as allele 1. Allele 3 is not detected by the assay.

### Value

matrix of haplotype counts

---

## merge_impute *Merge two genotype matrices and impute missing data*

---

### Description

Merge two genotype matrices and impute missing data by BLUP

### Usage

```
merge_impute(geno1, geno2, ploidy)
```

### Arguments

| | |
|---|---|
| geno1 | Genotype matrix (coded 0...ploidy) with dimensions markers x indiv |
| geno2 | Genotype matrix (coded 0...ploidy) with dimensions markers x indiv |
| ploidy | Either 2 or 4 |

### Details

Designed to impute from low to high density markers. The BLUP method is equivalent to Eq. 4 of Poland et al. (2012), but this function is not iterative. Additional shrinkage toward the mean is applied if needed to keep the imputed values within the range [0,ploidy]. Missing data in the input matrices are imputed with the population mean for each marker. If an individual appears in both input matrices, it is renamed with suffixes ".1" and ".2" and treated as two different individuals. Monomorphic markers are removed.

### Value

Imputed genotype matrix (markers x indiv)

### References

Poland et al. (2012) Plant Genome 5:103-113.

## readXY *Read SNP array intensity data*

### Description

Read SNP array intensity data

### Usage

```
readXY(filename, skip = 9, output = "ratio")
```

### Arguments

| | |
|---|---|
| filename | filename |
| skip | number of lines to skip before the header line with the column names |
| output | One of three options: "ratio","theta","AD" |

### Details

The first two columns of the tab-delimited input file should be the SNP and Sample ID. Columns labeled "X" and "Y" contain the signal intensities for the two alleles. Use output to specify whether to return the ratio = Y/(X+Y) or theta = atan(Y/X)*2/pi. Option "AD" exports the XY data in the allele depth format for a VCF file ("X,Y"), with the X and Y values multiplied by 100 and rounded to the nearest integer.

### Value

matrix with dimensions markers x individuals

## update_alias *Update names based on alias*

### Description

Update names based on data frame with alias and preferred name

### Usage

```
update_alias(x, alias, remove.space = TRUE, filename = NULL)
```

### Arguments

| | |
|---|---|
| x | Vector of names to update |
| alias | Data frame with two columns: first is the preferred name and second is the alias |
| remove.space | TRUE/FALSE |
| filename | update names in CSV file |

### Details

Parameter `remove.space` indicates whether blank spaces should be removed before string matching.

### Value

Vector with updated names

---

## `vcf2csv` *Convert VCF to CSV*

---

### Description

Convert VCF to CSV

### Usage

```
vcf2csv(vcf.file, csv.file, format)
```

### Arguments

| | |
|---|---|
| `vcf.file` | Input file |
| `csv.file` | Output file |
| `format` | Name of FORMAT key to export, either "GT" or "DS" |

### Value

none

---

## `write_vcf` *Create VCFv4.3 file*

---

### Description

Create VCFv4.3 file

### Usage

```
write_vcf(filename, fixed, geno, other.meta = NULL)
```

### Arguments

| | |
|---|---|
| `filename` | VCF file name |
| `fixed` | character matrix with 8 columns: CHROM, POS, ID, REF, ALT, QUAL, FILTER, INFO |
| `geno` | named list of genotype matrices, see Details |
| `other.meta` | optional, other metadata (without ##) besides INFO and FORMAT keys |

## Details

Several standard INFO key are recognized: ##INFO=<ID=REF,Number=A,Type=Character,Description=\"Array allele (A/B) in reference genome\"> ##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of samples with data"> ##INFO=<ID=DP.AVG,Number=1,Type=Float,Description="Average Sample Depth"> ##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth"> ##INFO=<ID=AB,Number=1,Typ Bias"> ##INFO=<ID=SE,Number=1,Type=Integer,Description="Sequencing Error (PHRED)"> ##INFO=<ID=OD,Nun (PHRED)"> ##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency"> ##INFO=<ID=AF.GT,Numbe Frequency based on GT"> ##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes">" ##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles">" Every element of geno is m x n matrix (m variants, n samples), e.g., AD, GT. The FORMAT field is created from the order and names of geno. Sample names taken from colnames of geno. Metadata for geno is generated from the names of the list: ##FORMAT=<ID=GT,Number=1,Type=String,Description="G ##FORMAT=<ID=AD,Number=R,Type=Integer,Description="Allele Depth"> ##FORMAT=<ID=DP,Number=1,Type= Depth"> ##FORMAT=<ID=DS,Number=1,Type=Float,Description="Posterior Mean Dosage"> ##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">

Any additional metadata should be included without the ## prefix.

# Index