Package ‘starmie’

November 7, 2016

Type Package

Title Population Structure Model Inference and Visualisation

Version 0.1.2

Description Data structures and methods for manipulating output of genetic population structure clustering algorithms. ‘starmie’ can parse output from ‘STRUCTURE’ (see <https://pritchardlab.stanford.edu/structure.html> for details) or ‘ADMIXTURE’ (see <https://www.genetics.ucla.edu/software/admixture/> for details). ‘starmie’ performs model selection via information criterion, and provides functions for MCMC diagnostics, correcting label switching and visualisation of admixture coefficients.

License MIT + file LICENSE

LazyData TRUE

Depends R (>= 3.3.0)

Imports data.table, readr (>= 1.0.0), stringr, tidyr, purrr, ggplot2 (>= 2.1.0), iterpc, combinat, label.switching, proxy, MCL, gridExtra, ggdendro, ggrepel, MCMCpack, parallel, methods, stats

Suggests testthat, knitr, rmarkdown

URL https://github.com/sa-lee/starmie

BugReports https://github.com/sa-lee/starmie/issues

RoxygenNote 5.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Gerry Tonkin-Hill [aut], Stuart Lee [cre, aut]

Maintainer Stuart Lee <lee.s@wehi.edu.au>

Repository CRAN

Date/Publication 2016-11-07 08:12:43
## R topics documented:

<table>
<thead>
<tr>
<th>R function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>admix</td>
<td>2</td>
</tr>
<tr>
<td>admixList</td>
<td>3</td>
</tr>
<tr>
<td>averagePairWiseSimilarityH</td>
<td>3</td>
</tr>
<tr>
<td>averageQ</td>
<td>4</td>
</tr>
<tr>
<td>bestK</td>
<td>4</td>
</tr>
<tr>
<td>clumpak</td>
<td>5</td>
</tr>
<tr>
<td>clumpp</td>
<td>6</td>
</tr>
<tr>
<td>exampleAdmixture</td>
<td>6</td>
</tr>
<tr>
<td>exampleStructure</td>
<td>7</td>
</tr>
<tr>
<td>getClusterAlleleFreqMat</td>
<td>7</td>
</tr>
<tr>
<td>getCompleteAlleleFreqMat</td>
<td>7</td>
</tr>
<tr>
<td>getD</td>
<td>8</td>
</tr>
<tr>
<td>getK</td>
<td>8</td>
</tr>
<tr>
<td>getQ</td>
<td>9</td>
</tr>
<tr>
<td>getStephens</td>
<td>9</td>
</tr>
<tr>
<td>loadAdmixture</td>
<td>9</td>
</tr>
<tr>
<td>loadStructure</td>
<td>10</td>
</tr>
<tr>
<td>plotBar</td>
<td>11</td>
</tr>
<tr>
<td>plotMCMC</td>
<td>11</td>
</tr>
<tr>
<td>plotMDS</td>
<td>12</td>
</tr>
<tr>
<td>plotMultiK</td>
<td>13</td>
</tr>
<tr>
<td>plotTreeBar</td>
<td>13</td>
</tr>
<tr>
<td>runStructure</td>
<td>14</td>
</tr>
<tr>
<td>struct</td>
<td>15</td>
</tr>
<tr>
<td>structList</td>
<td>16</td>
</tr>
</tbody>
</table>

### Index

<table>
<thead>
<tr>
<th>R function</th>
</tr>
</thead>
<tbody>
<tr>
<td>admix</td>
</tr>
</tbody>
</table>

### Description

Constructor for admix object

### Usage

```r
admix()
```

### Value

an admix object which is a list with 6 elements: K: number of clusters estimated by ADMIXTURE
nsamples: number of samples used
nmarkers: number of markers used
Q_df: a data.frame of cluster membership probabilities
P_df: a data.frame of estimated marker frequencies in each inferred population
log_info: a data.frame containing the K, CVerror and logLik of the last model.

---

**admixList**

*Constructor for admixList*

**Description**

Collect many `admix` objects

**Usage**

```r
admixList(...)  
```

**Arguments**

- `...` a list of `admix` objects

**Value**

an admixList object

---

**averagePairWiseSimilarity**

*Average Q matrices.*

**Description**

Average Q matrices.

**Usage**

```r
averagePairWiseSimilarity(Q_list)  
```

**Arguments**

- `Q_list` A list of Q matrices.

**Examples**

```r
# Read in Structure files
multiple_runs_k10 <- exampleStructure("mcmc_diagnostics")
Q_list <- lapply(multiple_runs_k10, getQ)
avgQ <- averagePairWiseSimilarity(Q_list)
```
averageQ  
*Average Q matrices.*

**Description**

Average Q matrices.

**Usage**

`averageQ(Q_list)`

**Arguments**

- `Q_list`  
  A list of Q matrices.

**Examples**

```r
# Read in Structure files
multiple_runs_k10 <- exampleStructure("mcmc_diagnostics")
Q_list <- lapply(multiple_runs_k10, getQ)
avgQ <- averageQ(Q_list)
```

---

**bestK**  
*Determine a suitable K value from multiple Structure runs*

**Description**

Determine a suitable K value from multiple Structure runs

**Usage**

`bestK(x, method, plot = TRUE)`

**Arguments**

- `x`  
  A `structList` or `admixList` object.
- `method`  
  The method used to calculate the best K either 'evanno' or 'structure', not required for `admixList` objects.
- `plot`  
  Whether or not to generate diagnostic plots

**Details**

If the K values are not ordered or there an even numbers of runs per K the 'structure' method will be implemented and the 'evanno' method to compute delta K will not be returned in the output.
Value

a data.frame containing with columns containing the $L_k$, AIC, BIC, DIC and deltaK for \texttt{structList}. If an \texttt{admixList} was given a data.frame returning the log information will be supplied. If plot = TRUE a ggplot object is printed for the method of interest.

Examples

```r
multi_K <- exampleStructure("multiple_runs")
# Run the evanno method and display diagnostic plots.
evanno_results <- bestK(multi_K, method = "evanno")
# Run the default structure method and display diagnostic plots
structure_results <- bestK(multi_K, "structure")
# find 'best' K according to results
deltaK <- evanno_results$variable == 'delta K'
max_deltaK <- which(evanno_results$value == max(evanno_results$value[deltaK], na.rm = TRUE))
evanno_results[max_deltaK, ]
LK <- structure_results$variable == 'L(K)'
max_LK <- which(structure_results$value == max(structure_results$value[LK], na.rm = TRUE))
structure_results[max_LK, ]
# admixture example
multi_K_admix <- exampleAdmixture()
bestK(multi_K_admix)
```

---

clumpak

Run the CLUMPP algorithms.

Description

Run the CLUMPP algorithms.

Usage

```r
clumpak(Q_list, method = "none")
```

Arguments

- **Q_list**
  A list of of Q matrices.
- **method**
  The method the algorithm uses to infer the correct permutations. One of 'greedy' or 'greedyLargeK' or 'stephens' or 'none'

Examples

```r
# Read in Structure files
multiple_runs_k10 <- exampleStructure("mcmc_diagnostics")
Q_list <- lapply(multiple_runs_k10, getQ)
clumpak_results <- clumpak(Q_list)
```
clumpp | Run the CLUMPP algorithms.

Description
Run the CLUMPP algorithms.

Usage
clumpp(Q_list, method = "greedy", iter = 100)

Arguments
Q_list | A list of of Q matrices.
method | The algorithm to use to infer the correct permutations. One of 'greedy' or 'greedyLargeK' or 'stephens'
iter | The number of iterations to use if running either 'greedy' or 'greedyLargeK'

Examples
# use multiple K=3 runs
cl_data <- exampleStructure("clumpp")
print(cl_data)
Q_list <- lapply(cl_data, getQ)
clumppy <- clumpp(Q_list)

elementAdmixture | Example admixture runs

Description
Example admixture runs

Usage
elementAdmixture()
exampleStructure

Example structure objects

Description
load structure objects for different starmie functions

Usage
exampleStructure(example_type)

Arguments
example_type  a character string either "multiple_runs", "clumpp" or "mcmc_diagnostics" or "barplot"

getClusterAlleleFreqMat

Retrieve estimated within-cluster allele frequencies

Description
Retrieve estimated within-cluster allele frequencies

Usage
getClusterAlleleFreqMat(x)

Arguments
x  a struct or admix object.

getCompleteAlleleFreqMat

Retrieve estimated population allele frequencies

Description
Retrieve estimated population allele frequencies

Usage
getCompleteAlleleFreqMat(x)

Arguments
x  a struct or admix object.
getD

Accessor methods for struct objects

Description
getD Return the number of free parameters in STRUCTURE model

Usage
getD(structure_obj)
getPosterior(structure_obj)
getFitStats(structure_obj)
getMCMC(structure_obj)

Arguments
structure_obj a struct object

Functions
• getPosterior: Return the estimated log posterior probability (L_k) from a struct object
• getFitStats: Return the estimated mean and variance of estimated log-likelihood from a struct object
• getMCMC: Return non-burn in MCMC iterations.

getK

Retrieve the assumed number of populations from struct or admix objects.

Description
Retrieve the assumed number of populations from struct or admix objects.

Usage
getK(x)

Arguments
x a struct object or admix object.
getQ

Retrieve Q matrix from struct or admix objects.

**Description**

Retrieve Q matrix from struct or admix objects.

**Usage**

getQ(x)

**Arguments**

- **x**
  - a struct or admix object.

getStephens

Use the Stephen’s method to permute sample labels

**Description**

Use the Stephen’s method to permute sample labels

**Usage**

getStephens(Q_list)

**Arguments**

- **Q_list**
  - A list of of Q matrices.

loadAdmixture

Read Admixture Output

**Description**

Read Admixture Output

**Usage**

loadAdmixture(qfile, pfile, logfile = NULL)
loadStructure

Arguments

- **qfile**
  - a valid Q file from ADMIXTURE
- **pfile**
  - a corresponding P file from ADMIXTURE
- **logfile**
  - logfile from corresponding AMIXTURE run

Value

an `admix` object containing the output of an admixture run

Examples

```r
qfin <- system.file("extdata/hapmap3_files", "hapmap3.2.Q", package = "starmie")
pfin <- system.file("extdata/hapmap3_files", "hapmap3.2.P", package = "starmie")
my_admix <- loadAdmixture(qfin, pfin)
# add log file
logfin <- system.file("extdata/hapmap3_files", "log2.out", package = "starmie")
my_admix <- loadAdmixture(qfin, pfin, logfin)
```

Description

Read Structure Output

Usage

```r
loadStructure(filename, logfile = NULL)
```

Arguments

- **filename**
  - a string containing an .out_f file
- **logfile**
  - optional string containing logfile produced by structure (default NULL).

Examples

```r
# read in K = 10 Structure file (both out_f and log file)
k10_r1 <- system.file("extdata/microsat_testfiles", "locprior_K10.out_f", package = "starmie")
k10_log <- system.file("extdata/microsat_testfiles", "chain_K10.log", package = "starmie")
# no log
k10_data <- loadStructure(k10_r1)
k10_data
# with log
k10_data <- loadStructure(k10_r1, k10_log)
k10_data
```
plotBar  

*Generate a barplot of a Structure or Admixture run.*

**Description**

Generate a barplot of a Structure or Admixture run.

**Usage**

```
plotBar(x, populations = NULL, plot = TRUE, facet = TRUE)
```

**Arguments**

- `x`: an object of type `struct` or `admix` or a Q-matrix
- `populations`: a data.frame that contains the sample number as the first column and the population as the second.
- `plot`: if FALSE returns a data.frame for customised plots
- `facet`: whether or not to split the barplot by cluster. This is recommended.

**Examples**

```r
# Read file using K = 6 and plot results
k6_data <- exampleStructure("barplot")
# Generate standard 'structure' barplot
plotBar(k6_data, facet = FALSE)
# adding group information
set.seed(212)
pops <- data.frame(id = k6_data$ancest_df[,1],
  population = sample(letters[1:3], nrow(k6_data$ancest_df), replace = TRUE))
# our facetted structure plot
plotBar(k6_data, pops)
# standard 'structure' bar plot
plotBar(k6_data, pops, facet = FALSE)
# admixture example
k3_data <- exampleAdmixture()[[3]]
plotBar(k3_data)
```

plotMCMC  

*Plot STRUCTURE MCMC chains*

**Description**

Plot non-burn MCMC iterations of STRUCTURE for checking convergence. If plot is set to FALSE a data.frame is returned containing the log likelihood and alpha values over different K and runs and not plot is printed to the device.
plotMDS

Plot principal coordinates from Q-matrix, struct or admix objects

Description
Plot principal coordinates from Q-matrix, struct or admix objects

Usage
plotMDS(x, method = NULL)

Arguments
x a Q-matrix of probability memberships, or struct or admix object
method (default = NULL) string 'nnd' or 'jsd' valid only for struct objects

Details
"nnd" uses the nucleotide distance matrix estimated by STRUCTURE to construct the principal coordinates, sizing the points by the expected heterozygosity within a cluster. "jsd" produces a principal coordinates from the Jensen Shannon Divergence metric as used by the 'ldavis' package and is the default for Q-matrix or admix objects. By default using plotMDS on a struct object will produce principal coordinates on the clusters themselves rather than within samples.
Examples

```r
# struct example
k6_data <- exampleStructure("barplot")
plotMDS(k6_data)
plotMDS(k6_data, method = "jsd")

# admix example
k3_data <- exampleAdmixture()[[3]]
plotMDS(k3_data)
```

---

**plotMultiK**

*Generate a barplot for multiple values of K.*

**Description**

Generate a barplot for multiple values of K.

**Usage**

```r
plotMultiK(x, populations = NULL, plot = TRUE)
```

**Arguments**

- `x`: A `structList` or `admixList` object or a list of Q-matrices.
- `populations`: A data.frame that contains the sample number as the first column and the population as the second.
- `plot`: If FALSE returns a data.frame for customised plots.

**Examples**

```r
cluster_runs <- exampleStructure("multiple_runs")
# Generate barplot
plotMultiK(cluster_runs[3:5])
```

---

**plotTreeBar**

*Generate a barplot of a Structure or Admixture run.*

**Description**

Generate a barplot of a Structure or Admixture run.

**Usage**

```r
plotTreeBar(x, facet = TRUE, dendro = NULL, cut = NULL)
```
Arguments

- **x**: a single cluster run object of type `struct` or `admix` or Q-matrix
- **facet**: whether or not to split the barplot by cluster. This is recommended.
- **dendro**: an object of class `hclust` (defaults to `hclust` with average linkage)
- **cut**: an integer vector output by `cutree` (defaults to `cutree k = ncol(Q)`)

Examples

```r
# Read file using K = 6 and plot results
k6_data <- exampleStructure("barplot")
# our facetted structure plot with tree
plotTreeBar(k6_data)
# standard 'structure' bar plot with tree
plotTreeBar(k6_data, facet = FALSE)
# Admix example
k3_data <- exampleAdmixture()[[3]]
plotTreeBar(k3_data)
```

---

**runStructure**

*Run STRUCTURE in current path*

**Description**

Run STRUCTURE in current path

**Usage**

```r
runStructure(path_to_structure, input_file, main_params, extra_params,
             out_prefix, n_K, n_replicates, n_cores)
```

**Arguments**

- **path_to_structure**: path to structure binary executable (ie. `/usr/bin/structure`)
- **input_file**: file name of input data
- **main_params**: file name of mainparams file for STRUCTURE
- **extra_params**: file name of extraparams file for STRUCTURE
- **out_prefix**: prefix path/name for logging
- **n_K**: number of assumed populations to try
- **n_replicates**: number of replicates
- **n_cores**: number of cores

**Note**

Set `RANDOMIZE = 0` in main params file to avoid using same seed. Haven’t tested on Windows.
struct

Examples

```r
## Not run:
input_file <- system.file("inst/extdata/microsat_testfiles", "locprior.str", package = "starmie")
main_params <- system.file("inst/extdata/microsat_testfiles", "mainparams", package = "starmie")
extra_params <- system.file("inst/extdata/microsat_testfiles", "extraparams", package = "starmie")
runStructure("structure", input_file, main_params, extra_params, "test", 5, 2, 2)

## End(Not run)
```

---

**struct**

*Constructor for struct object*

**Description**

struct object for storing structure run information

**Usage**

```r
struct()
```

**Details**

The **struct** object is a list with 11 elements:

- `K`: number of clusters estimated by Structure
- `run_params`: the run parameters given to the Structure program
- `mem_df`: assigned cluster membership proportions
- `allele_freqs`: Estimated net nucleotide distances between cluster
- `avg_dist_df`: Average distance between individuals
- `fit_stats_df`: Model fit statistics
- `fst_df`: Fst values
- `ancest_df`: Inferred ancestry of individuals
- `clust_allele_list`: Cluster allele frequencies
- `burn_df`: Burn in MCMC iteration output
- `nonburn_df`: Main MCMC iteration output

**Value**

a **struct** object

**See Also**

- `loadStructure` for reading in STRUCTURE out_f files.
- `structList` for manipulating multiple struct objects
structList

Constructor for a structList object

Description

the structList class is a container for storing a collection of struct objects.

Usage

structList(...) 

Arguments

... a list of a struct objects
Index

admix, 2, 3, 7–12, 14
admixlist, 3, 4, 5, 13
averagePairWiseSimilarityH, 3
averageQ, 4
bestK, 4
clumpak, 5
clumpp, 6

eampleAdmixture, 6
eampleStructure, 7

getClusterAlleleFreqMat, 7
getCompleteAlleleFreqMat, 7
getD, 8
getFitStats (getD), 8
getK, 8
getMCMC (getD), 8
getPosterior (getD), 8
getQ, 9
getStephens, 9

loadAdmixture, 9
loadStructure, 10, 15

plotBar, 11
plotMCMC, 11
plotMDS, 12
plotMultiK, 13
plotTreeBar, 13

runStructure, 14

struct, 7–9, 11, 12, 14, 15, 16
structList, 4, 5, 12, 13, 15, 16