Package 'conStruct'

August 22, 2018

Version 1.0.0

Title Models Spatially Continuous and Discrete Population Genetic Structure

Description A method for modeling genetic data as a combination of discrete layers, within each of which relatedness may decay continuously with geographic distance. This package contains code for running analyses (which are implemented in the modeling language rstan) and visualizing and interpreting output. See the paper for more details on the model and its utility.

License GPL-3

Encoding UTF-8

LazyData true

ByteCompile true

Depends R (>= 3.4.0), Rcpp (>= 0.12.18), methods

Imports rstan (>= 2.17.3), rstantools (>= 1.5.0), caroline, gtools, foreach, parallel, doParallel

LinkingTo StanHeaders (>= 2.17.2), rstan (>= 2.17.3), BH (>= 1.66.0-1), Rcpp (>= 0.12.18), RcppEigen (>= 0.3.3.4.0)

SystemRequirements GNU make

NeedsCompilation yes

RoxygenNote 6.1.0

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

conStruct-package *The 'conStruct' package.*

Description

A method for modeling genetic data as a combination of discrete layers, within each of which relatedness may decay continuously with geographic distance. This package contains code for running analyses (which are implemented in the modeling language rstan) and visualizing and interpreting output. See the paper (doi: 10.1534/genetics.118.301333) for more details on the model and its utility.

References

G.S. Brdburd, G.M. Coop, and P.L. Ralph (2018). Inferring continuous and discrete population genetic structure across space. Genetics. doi: 10.1534/genetics.118.301333 Stan Development Team (2018). RStan: the R interface to Stan. R package version 2.17.3. http://mc-stan.org

---

calculate.layer.contribution
Calculate layer contribution

Description

calculate.layer.contribution

Usage

```
calculate.layer.contribution(conStruct.results, data.block,
  layer.order = NULL)
```

Arguments

conStruct.results	The list output by a conStruct run for a given MCMC chain.
data.block	A data.block list saved during a conStruct run.
layer.order	An optional vector giving the order in which the layers of conStruct.results are read.

Details

This function takes the results of a conStruct analysis and calculates the relative contributions of each layer to total covariance.

This function calculates the contribution of each layer to total covariance by multiplying the within-layer covariance in a given layer by the admixture proportions samples draw from that layer. The relative contribution of that layer is this absolute contribution divided by the sum of those of all other layers. A layer can have a large contribution if many samples draw large amounts of admixture from it, or if it has a very large within-layer covariance parameter (phi), or some combination of the two. Layer contribution can be useful for evaluating an appropriate level of model complexity for the data (e.g., choosing a value of K or comparing the spatial and nonspatial models).

Value

This function returns a vector giving the relative contributions of the layers in the analysis.

---

conStruct *Run a conStruct analysis.*

Description

conStruct runs a conStruct analysis of genetic data.

Usage

```
conStruct(spatial = TRUE, K, freqs, geoDist = NULL, coords,
  prefix = "", n.chains = 1, n.iter = 1000, make.figs = TRUE,
  save.files = TRUE)
```

Arguments

spatial	A logical indicating whether to perform a spatial analysis. Default is TRUE.
K	An integer that indicates the number of layers to be included in the analysis.
freqs	A matrix of allele frequencies with one column per locus and one row per sample. Missing data should be indicated with NA.
geoDist	A full matrix of geographic distance between samples. If NULL, user can only run the nonspatial model.
coords	A matrix giving the longitude and latitude (or X and Y coordinates) of the samples.
prefix	A character vector giving the prefix to be attached to all output files.
n.chains	An integer indicating the number of MCMC chains to be run in the analysis. Default is 1.
n.iter	An integer giving the number of iterations each MCMC chain is run. Default is 1e3. If the number of iterations is greater than 500, the MCMC is thinned so that the number of retained iterations is 500 (before burn-in).
make.figs	A logical value indicating whether to automatically make figures once the analysis is complete. Default is TRUE.
save.files	A logical value indicating whether to automatically save output and intermediate files once the analysis is complete. Default is TRUE.

Details

This function initiates an analysis that uses geographic and genetic relationships between samples to estimate sample membership (admixture proportions) across a user-specified number of layers.

This function acts as a wrapper around a STAN model block determined by the user-specified model (e.g., a spatial model with 3 layers, or a nonspatial model with 5 layers). User-specified data are checked for appropriate format and consistent dimensions, then formatted into a data.block, which is then passed to the STAN model block. Along with the conStruct.results output described above, several objects are saved during the course of a conStruct call (if save.files=TRUE). These are the data.block, which contains all data passed to the STAN model block, model.fit, which is unprocessed results of the STAN run in stanfit format, and the conStruct.results,

which are saved in the course of the function call in addition to being returned. If make.figs=TRUE, running conStruct will also generate many output figures, which are detailed in the function make.all.the.plots in this package.

Value

This function returns a list with one entry for each chain run (specified with n.chains). The entry for each chain is named "chain_X" for the Xth chain. The components of the entries for each are detailed below:

- posterior gives parameter estimates over the posterior distribution of the MCMC.
  - n.iter number of MCMC iterations retained for analysis (half of the n.iter argument specified in the function call).
  - lpd vector of log posterior density over the retained MCMC iterations.
  - nuggets matrix of estimated nugget parameters with one row per MCMC iteration and one column per sample.
  - par.cov array of estimated parametric covariance matrices, for which the first dimension is the number of MCMC iterations.
  - gamma vector of estimated gamma parameter.
  - layer.params list summarizing estimates of layer-specific parameters. There is one entry for each layer specified, and the entry for the kth layer is named "Layer_k".
    - alpha0 vector of estimated alpha0 parameter in the kth layer.
    - alphaD vector of estimated alphaD parameter in the kth layer.
    - alpha2 vector of estimated alpha2 parameter in the kth layer.
    - mu vector of estimated mu parameter in the kth layer.
    - layer.cov vector of estimated layer-specific covariance parameter in the kth layer.
  - admix.proportions array of estimated admixture proportions. The first dimension is the number of MCMC iterations, the second is the number of samples, and the third is the number of layers.
- MAP gives point estimates of the parameters listed in the posterior list described above. Values are indexed at the MCMC iteration with the greatest posterior probability.
  - index.iter the iteration of the MCMC with the highest posterior probability, which is used to index all parameters included in the MAP list
  - lpd the greatest value of the posterior probability
  - nuggets point estimate of nugget parameters
  - par.cov point estimate of parametric covariance
  - gamma point estimate of gamma parameter
  - layer.params point estimates of all layer-specific parameters
  - admix.proportions point estimates of admixture proportions.

---

conStruct.data *Example dataset used in a* conStruct *analysis*

Description

A simulated dataset containing the allele frequency and sampling coordinate data necessary to run a conStruct analysis.

Usage

```
conStruct.data
```

Format

A list with two elements:

allele.frequencies a matrix with one row for each of the 36 samples and one column for each of 10,000 loci, giving the frequency of the counted allele at each locus in each sample

coords a matrix with one row for each of the 36 samples, in the same order as that of the allele frequency matrix, and two columns, the first giving the x-coordinate (or longitude), the second giving the y-coordinate (or latitude)

---

make.admix.pie.plot *Make admixture pie plot*

Description

`make.structure.plot` makes a map of pie plots showing admixture proportions across layers.

Usage

```
make.admix.pie.plot(admix.proportions, coords, layer.colors = NULL,
  radii = 2.7, add = FALSE, x.lim = NULL, y.lim = NULL,
  mar = c(2, 2, 2, 2))
```

Arguments

`admix.proportions`	A `matrix` of admixture proportions, with one row per sample and one column per layer.
`coords`	`matrix` of sample coordinates, with one row per sample and two columns giving (respectively) the X and Y plotting coordinates.
`layer.colors`	A `vector` of colors to be used in plotting results for different layers. Users must specify one color per layer. If `NULL`, the plot will use a pre-specified vector of colors.
`radii`	A `vector` of numeric values giving the radii to be used in plotting admixture pie plots. If the number of values specified is smaller than the number of samples, radii values will be recycled across samples. The default is 2.7.
`add`	A `logical` value indicating whether to add the pie plots to an existing plot. Default is `FALSE`.
`x.lim`	A `vector` giving the x limits of the plot. The default value is `NULL`, which indicates that the range of values given in the first column of `coords` should be used.
`y.lim`	A `vector` giving the y limits of the plot. The default value is `NULL`, which indicates that the range of values given in the second column of `coords` should be used.
`mar`	A `vector` giving the number of lines of margin specified for the four sides of the plotting window (passed to `par`). Default value, which is only used if `add=FALSE`, is `c(2,2,2,2)`.

Details

This function takes the output from a conStruct analysis and makes a map of pie plots showing admixture proportions across layers, where each sample is represented as a pie chart, and the proportion of the pie of each color represent that sample's admixture proportion in that layer.

Value

This function has only invisible return values.

---

make.all.the.plots *Make output plots*

Description

`make.all.the.plots` makes figures from the output from a conStruct analysis.

Usage

```
make.all.the.plots(conStruct.results, data.block, prefix,
  layer.colors = NULL)
```

Arguments

`conStruct.results`	The list output by a `conStruct` run.
`data.block`	A `data.block` list saved during a `conStruct` run.
`prefix`	A character vector to be prepended to all figures.
`layer.colors`	A `vector` of colors to be used in plotting results for different layers. Users must specify one color per layer. If `NULL`, plots will use a pre-specified vector of colors.

Details

This function takes the output from a conStruct analysis and generates a number of plots for visualizing results and diagnosing MCMC performance.

This function produces a variety of plots that can be useful for visualizing results or diagnosing MCMC performance. The plots made are by no means an exhaustive, and users are encouraged to make further plots, or customize these plots as they see fit. For each plot, one file is generated for each MCMC chain (specified withe the `n.chains` argument in the function `conStruct`. The plots generated (as .pdf files) are:

- Structure plot - STRUCTURE-style plot, where each sample is represented as a stacked bar plot, and the length of the bar plot segments of each color represent that sample's admixture proportion in that layer. Described further in the help page for `make.structure.plot`.
- Admixture pie plot - A map of samples in which each sample's location is denoted with a pie chart, and the proportion of a pie chart of each color represents that sample's admixture in each layer. Described further in the help page for `make.admix.pie.plot`
- model.fit.CIs - A plot of the sample allelic covariance shown with the 95% credible interval of the parametric covariance for each entry in the matrix.

- layer.covariances - A plot of the layer-specific covariances overlain unto the sample allelic covariance.
- Trace plots - Plots of parameter values over the MCMC.
  - lpd - A plot of the log posterior probability over the MCMC.
  - nuggets - A plot of estimates of the nugget parameters over the MCMC.
  - gamma - A plot of estimates of the gamma parameter over the MCMC.
  - layer.cov.params - Plots of estimates of the layer-specific parameters over the MCMC.
  - admix.props - A plot of estimates of the admixture proportions over the MCMC.

Value

This function has only invisible return values.

`make.structure.plot` *Make STRUCTURE output plot*

Description

`make.structure.plot` makes a STRUCTURE-style plot from the output from a conStruct analysis.

Usage

```
make.structure.plot(admix.proportions, mar = c(2, 4, 2, 2),
  sample.order = NULL, layer.order = NULL, sample.names = NULL,
  sort.by = NULL, layer.colors = NULL)
```

Arguments

`admix.proportions`	A `matrix` of admixture proportions, with one row per sample and one column per layer.
`mar`	A `vector` of plotting margins passed to `par`. Default is `c(2,4,2,2)`, which tends to look good.
`sample.order`	A `vector` giving the order in which sample admixture proportions are to be plotted, left to right. If `NULL`, samples are plotted in the order they occur in `admix.proportions`.
`layer.order`	A `vector` giving the order in which layers are plotted, bottom to top. If `NULL`, layers are plotted in the order they occur in `admix.proportions`.
`sample.names`	Vector of names to be plotted under each sample's admixture proportion bar plot. The index of a sample's name should be the same as the index of the sample's row in `admix.proportions`. If `NULL`, no names are printed.
`sort.by`	An `integer` giving the column index of the `admix.proportions` matrix to be used in determining sample plotting order. If specified, samples are plotted from left to right in increasing order of their membership in that layer. If `NULL`, samples are plotted in the order they occur in `admix.proportions`.
`layer.colors`	A `vector` of colors to be used in plotting results for different layers. Users must specify one color per layer. If `NULL`, the plot will use a pre-specified vector of colors.

Details

This function takes the output from a conStruct analysis and makes a STRUCTURE-style plot, where each sample is represented as a stacked bar plot, and the length of the bar plot segments of each color represent that sample's admixture proportion in that layer.

Value

This function has only invisible return values.

`match.layers.x.runs` *Match layers up across independent conStruct runs*

Description

`match.layers.x.runs`

Usage

```
match.layers.x.runs(admix.mat1, admix.mat2, admix.mat1.order = NULL)
```

Arguments

`admix.mat1`	A `matrix` of estimated admixture proportions from the original `conStruct` analysis, with one row per sample and one column per layer.
`admix.mat2`	A `matrix` of estimated admixture proportions from a second `conStruct` analysis, with one row per sample and one column per layer, for which the layer order is desired. Must have equal or greater number of layers to `admix.mat1`.
`admix.mat1.order`	An optional `vector` giving the order in which the layers of `admix.mat1` are read.

Details

This function takes the results of two independent `conStruct` analyses and compares them to identify which layers in a new analysis correspond most closely to the layers from an original analysis.

This function compares admixture proportions in layers across independent `conStruct` runs, and compares between them to identify the layers in `admix.mat2` that correspond most closely to those in `admix.mat1`. It then returns a vector giving an ordering of `admix.mat2` that matches up the order of the layers that correspond to each other. This can be useful for:

1. Dealing with "label switching" across independent runs with the same number of layers;
2. Plotting results from independent runs with different numbers of layers using consistent colors (e.g., the "blue" layer shows up as blue even as `K` increases);
3. Examining results for multimodality (i.e., multiple distinct solutions with qualitatively different patterns of membership across layers).

The `admix.mat1.order` argument can be useful when running this function to sync up plotting colors/order across the output of more than two `conStruct` runs.

Value

This function returns a vector giving the ordering of the layers in admix.mat2 that maximizes similarity between admix.mat1 and re-ordered admix.mat2.

---

structure2conStruct *Convert a dataset from STRUCTURE to conStruct format*

Description

structure2conStruct converts a STRUCTURE dataset to conStruct format

Usage

```
structure2conStruct(infile, start.loci, missing.datum, outfile)
```

Arguments

infile	The name and path of the file in STRUCTURE format to be converted to conStruct format.
start.loci	The index of the first column in the dataset that contains genotype data.
missing.datum	The character or value used to denote missing data in the STRUCTURE dataset (often -9).
outfile	The name and path of the file containing the conStruct formatted dataset to be generated by this function.

Details

This function takes a population genetics dataset in STRUCTURE format and converts it to conStruct format. The STRUCTURE file must have one row per individual and two columns per locus, and can only contain bi-allelic SNPs.

This function takes a STRUCTURE format data file and converts it to a conStruct format data file. The STRUCTURE dataset should be in the ONEROWPERIND file format, with one row per individual and two columns per locus (this function therefore can only be applied to diploid organisms). The first column of the STRUCTURE dataset should be individual names. There may be any number of other columns that contain non-genotype information before the first column that contains genotype data, but there can be no extraneous columns at the end of the dataset, after the genotype data. The genotype data should be bi-allelic single nucleotide polymorphisms (SNPs).

Value

This function returns an allele frequency data matrix that can be used as the freqs argument in a conStruct analysis run using conStruct. It also saves this object as an .RData file so that it can be used in future analyses.

x.validation *Run a conStruct cross-validation analysis*

Description

x.validation runs a conStruct cross-validation analysis

Usage

```
x.validation(train.prop = 0.9, n.reps, K, freqs = NULL,
  data.partitions = NULL, geoDist, coords, prefix, n.iter,
  make.figs = FALSE, save.files = FALSE, parallel = FALSE,
  n.nodes = NULL)
```

Arguments

train.prop	A numeric value between 0 and 1 that gives the proportions of the data to be used in the training partition of the analysis. Default is 0.9.
n.reps	An integer giving the number of cross- validation replicates to be run.
K	A numeric vector giving the numbers of layers to be tested in each cross-validation replicate. E.g., K=1:7.
freqs	A matrix of allele frequencies with one column per locus and one row per sample. Missing data should be indicated with NA.
data.partitions	A list with one element for each desired cross-vlaidation replicate. This argument can be specified instead of the freqs argument if the user wants to provide their own data partitions for model training and testing. See the model comparison vignette for details on what this should look like.
geoDist	A matrix of geographic distance between samples. If NULL, user can only run the nonspatial model.
coords	A matrix giving the longitude and latitude (or X and Y coordinates) of the samples.
prefix	A character vector giving the prefix to be attached to all output files.
n.iter	An integer giving the number of iterations each MCMC chain is run. Default is 1e3. If the number of iterations is greater than 500, the MCMC is thinned so that the number of retained iterations is 500 (before burn-in).
make.figs	A logical value indicating whether to automatically make figures during the course of the cross-validation analysis. Default is FALSE.
save.files	A logical value indicating whether to automatically save output and intermediate files once the analysis is complete. Default is FALSE.
parallel	A logical value indicating whether or not to run the different cross-validation replicates in parallel. Default is FALSE. For more details on how to set up runs in parallel, see the model comparison vignette.
n.nodes	Number of nodes to run parallel analyses on. Default is NULL. Ignored if parallel is FALSE. For more details in how to set up runs in parallel, see the model comparison vignette.

Details

This function initiates a cross-validation analysis that uses Monte Carlo cross-validation to determine the statistical support for models with different numbers of layers or with and without a spatial component.

Value

This function returns (and also saves as a .Robj) a `list` containing the standardized results of the cross-validation analysis across replicates. For each replicate, the function returns a list with the following elements:

- `sp` - the mean of the standardized log likelihoods of the "testing" data partition of that replicate for the spatial model for each value of K specified in `K`.
- `nsp` - the mean of the standardized log likelihoods of the "testing" data partitions of that replicate for the nonspatial model for each value of K specified in `K`.

In addition, this function saves two text files containing the standardized cross-validation results for the spatial and nonspatial results (prefix_sp_xval_results.txt and prefix_nsp_xval_results.txt, respectively). These values are written as matrices for user convenience; each column is a cross-validation replicate, and each row gives the result for a value of `K`.

Index