Package 'GABi'

February 19, 2015

Version 0.1

Date 2013-06-19

Title Framework for Generalized Subspace Pattern Mining

Author Ed Curry

Maintainer Ed Curry <e.curry@imperial.ac.uk>

Depends R (>= 2.15.0), hash

Suggests ArrayBin

Description Generalized subspace pattern mining in data arrays, using a genetic algorithm framework.

License GPL (>= 2)

URL http://www.r-project.org,
http://www1.imperial.ac.uk/medicine/people/e.curry/

NeedsCompilation no

Repository CRAN

Date/Publication 2013-09-17 14:47:54

R topics documented:

crossover *Crossover*

Description

Performs crossover genetic operator in GABi genetic algorithm.

Usage

```
crossover(subpop,xoverpoints,pinvert=0)
```

Arguments

`subpop`	A numeric array representing a population of GA solutions. Each row corresponds to a distinct solution
`xoverpoints`	The number of 'crossover points' in the crossover operation (see details).
`pinvert`	The probability with which the order of the subset to be swapped of one of the solutions will be reversed prior to crossover.

Details

Crossover exchanges parts of two GABi solutions, giving rise to two 'offspring' solutions, each containing parts of the two 'parent' solutions. The position of the crossover points (i.e. the limits of the regions of the solutions that will be swapped with each other) is randomly sampled from the whole length of available positions (constrained only by the number of crossover points chosen).

Value

A numeric array representing a population of GA solutions. Each row corresponds to a distinct solution.

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

exchangeSols *Solution exchange between isolated GA subpopulations*

Description

The GABi genetic algorithm employs an 'island' model, in which distinct subpopulations are allowed to evolve in isolation of each other. In order to avoid total isolation, subpopulations may periodically exchange a solution with each other.

Usage

```
exchangeSols(demes,fitnesses,fittestonly,proximity)
```

Arguments

`demes`	List of numeric arrays, each one a distinct subpopulation of GA solutions.
`fitnesses`	List of numeric vectors, each one indicating the fitness of the corresponding solution (row) of the corresponding array in `demes`.
`fittestonly`	Boolean indicating whether or not only the fittest solution from each subpopulation is exchanged.
`proximity`	Boolean indicating whether or not the solution exchange can only occur between 'adjacent' subpopulations (i.e. consecutive elements of the list `demes`) or between any subpopulations.

Details

In the solution exchange process, a predetermined number of randomly selected solutions from each subpopulation (i.e. rows from each numeric array) are swapped with solutions from another subpopulation. See Whitley 1995 for more details on the Island Model of GA populations.

Value

List of numeric arrays, as input `demes` but with some solutions having been exchanged between the subpopulations.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

`featureSelection.basic` *Feature Selection for Block Biclusters in Binary Data*

Description

A feature selection function for the GABi biclustering framework, based on the definition of a bicluster as a block of consistently high values across a submatrix within a binary dataset.

Usage

```
featureSelection.basic(cols)
```

Arguments

`cols`	Numeric vector representing a subset of the columns from `x` across which this solution's bicluster pattern is defined.

Details

A fast feature selection function is vital to the GABi framework of biclustering. In GABi, the bicluster problem is reformulated around the fact that each subset of the columns across a dataset will have one _maximal_ subset of rows that fit a specified pattern, and the submatrix defined by this maximal subset of rows will be the most interesting observation involving that subset of columns. Makes use of `fitnessArgs` a list of parameters in the environment of execution of the biclustering function `GABi`. Notably, the element `consistency` is used to apply a stringency threshold for selecting features (i.e. only those with the proportion of high values across the subset of samples being greater than `consistency`)

Value

Numeric vector representing the features (i.e. rows) from dataset `x` representing the maximal bicluster for the solution encoded by `chr`.

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

`fps` *Fitness Proportional Selection*

Description

Performs selection step in GA process using fitness proportional selection.

Usage

```
fps(population,fitnesses,elitism)
```

Arguments

`population`	Numeric array representing a population of binary GA chromosomes.
`fitnesses`	Numeric vector specifying relative fitnesses of each solution (row) of `population`.
`elitism`	Boolean indicating whether or not the fittest solution should be guaranteed to pass into the next generation.

Details

Fitness Proportional Selection

Value

Numeric array representing a population of binary GA chromosomes. Sampled with replacement from rows of `population`, biased towards each individual solution in direct proportion to its relative fitness.

Author(s)

Ed Curry <e.curry@imperial.ac.uk>

GABi *Genetic Algorithm for Generalized Biclustering*

Description

A flexible framework for finding submatrices that are good manifestations of a user-specified pattern from within a numeric (often binary) matrix. The user-defined pattern is specified via feature selection and bicluster desirability evaluation functions (see details).

Usage

```
GABi(x,nSols=0,convergenceGens=40,popsize=256,mfreq=1,xfreq=0.5,
maxNgens=200,keepBest=FALSE,identityThreshold=0.75,
nsubpops=4,experiod=10,diffThreshold=0.9,verbose=FALSE,maxLoop=1,
fitnessArgs=list(consistency=0.8,featureWeights = rowMeans(x, na.rm = TRUE)),
fitnessFun=getFitnesses.entropy,featureSelFun=featureSelection.basic)
```

Arguments

`x`	Numeric data input array used to generate binary output array. Each row of the array represents a different variable.
`nSols`	Number of solutions at which to terminate loop.
`convergenceGens`	Number of generations after which to terminate the GA process within each loop if no improvement to the best solution's fitness is seen.
`popsize`	Total number of solutions to be evolved in GA (divided across `nsubpops` subpopulations.)
`mfreq`	Mutation frequency: probability of flipping each bit in each GA solution is `mfreq/ncol(x)`.
`xfreq`	Crossover frequency: probability of each pair of solutions having the crossover operator being applied.
`maxNgens`	Maximum number of generations in GA process within each loop.
`keepBest`	Boolean specifying whether or not to pass the best solution from each generation unchanged into the next.
`identityThreshold`	Numeric value specifying the proportion of shared columns from `x` above which two biclusters are considered redundant and only the best is output as a solution.
`nsubpops`	Numeric value specifying the number of distinct subpopulations across which to distribute the GAs population of solutions. For more details on the Island Model of GAs, see Whitley 1995. If `nsubpops=1`, a traditional (non-Island) GA will be implemented.

`experiod`	Number of generations after which to exchange solutions between the distinct GA subpopulations. If `experiod` is greater than `maxNgens`, the subpopulations will be kept completely distinct.
`diffThreshold`	Numeric value specifying minimum proportion of values in each row of `x` to be greater than the minimum or less than the maximum value. Included primarily as a filter to remove invariant rows from binary datasets `x`.
`verbose`	Boolean indicating whether or not to print diagnostic messages to R console.
`maxLoop`	Numeric value specifying maximum number of runs of the GA, after which GABi will terminate and return all recovered solutions, even if `nSols` isn't reached.
`fitnessArgs`	List containing arguments to be used in `fitnessFun` and `featureSelFun`.
`fitnessFun`	Function taking argument `chr`, a numeric vector specifying the solution to be evaluated. All other arguments to be used in the function should be specified in `fitnessArgs`. Must return a single numeric value indicating the relative fitness (i.e. 'goodness') of the solution.
`featureSelFun`	Function taking argument `chr`, a numeric vector specifying the solution to be evaluated. All other arguments to be used in the function should be specified in `fitnessArgs`. Must return a numeric vector indicating which features (i.e. rows of `x`) to be included in the bicluster.

Details

GABi uses flexible user-defined (or preset) functions to perform generalized biclustering of a numeric or binary data matrix `x`. It implements a number of features, including an Island Model of population evolution (in which a number of distinct subpopulations are kept isolated for the purposes of selection and crossover) and an iterative loop of solution generation (in which the GA process is rerun with a 'tabu' list, ensuring that previously returned solutions are not selected for in subsequent runs of the GA). Given an appropriate fitness function `fitnessFun` and feature selection function `featureSelFun`, which take a binary chromosome (in which a `1` denotes that the corresponding column of `x` is included in the bicluster) and return a desirability score and a list of the features fitting the bicluster pattern across the specified columns, respectively.

Value

List of biclusters. Each bicluster represents a submatrix satisfying the conditions of the specified pattern, and contains the elements:

`features`	Which rows of the input array `x` are in this bicluster
`samples`	Which columns of the input array `x` are in this bicluster
`score`	Fitness evaluation of this bicluster (can be used to compare the different biclusters output by the algorithm)

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

Examples

```
## create a binary array
x <- array(round(runif(1200)),dim=c(100,12))
## Not run: x

## use GABi to find biclusters
x.bc <- GABi(x,maxNgens=20)
## Not run: x.bc
```

`getFitnesses.basic` *Basic Bicluster Fitness Function*

Description

A fitness function for the GABi biclustering framework, based on the simple principle that the larger the dense submatrix, the more interesting it is to discover.

Usage

```
getFitnesses.basic(chr)
```

Arguments

`chr`	Numeric vector representing a GA solution to the biclustering problem (i.e. a subset of the columns from `x` across which to look for the pattern).

Details

A fitness function is fundamental to the success of a GA. In this case, `getFitnesses.basic` evaluates the desirability of biclusters by multiplying the number of columns from dataset `x` (argument for function `GABi`) that displaying a consistent block of `1`s involving the features that are observed to fit this pattern. Makes use of `fitnessArgs` a list of parameters in the environment of execution of the biclustering function `GABi`. Notably, the element `consistency` is used to apply a stringency threshold for selecting features (i.e. only those with the proportion of high values across the subset of samples being greater than `consistency`).

Value

Numeric value representing fitness score for the solution encoded by `chr`.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

`getFitnesses.entropy` *Entropy-based Bicluster Fitness Function*

Description

A fitness function for the GABi biclustering framework, based on the principle that the less likely a bicluster would be observed by chance, the more interesting it is to discover that bicluster.

Usage

```
getFitnesses.entropy(chr)
```

Arguments

`chr`	Numeric vector representing a GA solution to the biclustering problem (i.e. a subset of the columns from `x` across which to look for the pattern).

Details

A fitness function is fundamental to the success of a GA. In this case, `getFitnesses.entropy` evaluates the desirability of biclusters by estimating the probability of a given selection of the columns from dataset `x` (argument for function `GABi`) displaying a consistent block of `1`s involving the features that are observed to fit this pattern. Makes use of `fitnessArgs` a list of parameters in the environment of execution of the biclustering function `GABi`. Notably, the element `featureWeights` is a numeric vector encoding the probability of any randomly selected column of the input matrix `x` having a high value of the corresponding row. This is used in the entropy calculation for the corresponding bicluster. And the element `consistency` is used to apply a stringency threshold for selecting features (i.e. only those with the proportion of high values across the subset of samples being greater than `consistency`).

Value

Numeric value representing fitness score for the solution encoded by `chr`.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

`mutation` *Mutation*

Description

Performs mutation genetic operator in GABi genetic algorithm.

Usage

```
mutation(pop, mfreq)
```

Arguments

`pop`	Numeric array representing a population of binary GA chromosomes.
`mfreq`	Numeric value defining probability with which each bit in `pop` will be flipped.

Details

Mutation randomly flips each bit in the GA chromosome with some specified (low) probability.

Value

Numeric array representing a population of binary GA chromosomes, derived from `pop` but potentially altered by the mutation process.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

`plotBicluster` *Bicluster Visualization*

Description

Creates a heatmap contrasting the data values for a bicluster's features across the bicluster with those features' values across the remainder of the dataset.

Usage

```
plotBicluster(x,dataset,col=gray(seq(from=1,to=0,length=100)))
```

Arguments

`x`	Bicluster object returned from `GABi`
`dataset`	Numeric matrix in which bicluster `x` was discovered
`col`	Colors to use in heatmap

Details

Uses `heatmap` to generate a false color image that illustrates the data values corresponding to the bicluster `x`. Default color is grayscale. A black/white bar along the top of the heatmap indicates the extent of bicluster membership (i.e. columns of the heatmap below the black bar come from data columns that were in the bicluster).

Value

An object returned by `heatmap`. That is, invisibly, a list with components:

`rowInd`	row index permutation vector as returned by `order.dendrogram`
`colInd`	column index permutation vector

Author(s)

Ed Curry `<e.curry@imperial.ac.uk>`

`reproduction` *Reproduction*

Description

Wrapper to apply the genetic operators (crossover and mutation) in the GABi GA.

Usage

```
reproduction(population,xfreq,mfreq,xoverpoints,pinvert,elitism)
```

Arguments

`population`	Numeric array representing a population of binary GA chromosomes.
`xfreq`	Crossover frequency: probability of each pair of solutions having the crossover operator being applied.
`mfreq`	Mutation frequency: probability of flipping each bit in each GA solution is `mfreq/ncol(x)`.
`xoverpoints`	The number of 'crossover points' in the crossover operation (see details).
`pinvert`	The probability with which the order of the subset to be swapped of one of the solutions will be reversed prior to crossover.
`elitism`	Boolean indicating whether or not the fittest solution should be guaranteed to pass into the next generation.

Details

Reproduction is the stage of a GA in which the solutions that are selected to be passed onto the next generation have the potential to be altered to construct new, and potentially better, solutions.

Value

Numeric array representing a population of binary GA chromosomes, derived from `population`.

Author(s)

Ed Curry <`e.curry@imperial.ac.uk`>

Index