Package 'gamesGA'

March 1, 2020

Type Package

Title Genetic Algorithm for Sequential Symmetric Games

Version 1.1.3.7

Imports grDevices (>= 3.4.0), graphics (>= 3.4.0), stats (>= 3.4.0), shiny (>= 1.0.0)

Maintainer A. Bradley Duthie <brad.duthie@gmail.com>

Description Finds adaptive strategies for sequential symmetric games using a genetic algorithm. Currently, any symmetric two by two matrix is allowed, and strategies can remember the history of an opponent's play from the previous three rounds of moves in iterated interactions between players. The genetic algorithm returns a list of adaptive strategies given payoffs, and the mean fitness of strategies in each generation.

URL https://bradduthie.github.io/gamesGA/

BugReports https://github.com/bradduthie/gamesGA/issues

Depends R (>= 3.4.0)

License MIT + file LICENSE

LazyData TRUE

RoxygenNote 6.1.1

Suggests testthat

NeedsCompilation yes

Author A. Bradley Duthie [aut, cre] (<https://orcid.org/0000-0001-8343-4995>)

Repository CRAN

Date/Publication 2020-03-01 21:00:02 UTC

R topics documented:

`crossover` *Crossover function*

Description

This function causes uniform crossing over among loci in a population of agents.

Usage

```
crossover(agents, prob = 0.1)
```

Arguments

`agents`	A list of agents on which crossing over will be performed
`prob`	The probability that crossing over happens at a locus

Value

agents The list of agents after crossing over

`games_ga` *Games genetic algorithm:*

Description

This function runs a genetic algorithm that identifies sequential strategies for maximising payoffs given any two by two symmetrical payoff matrix. Simulated players remember three rounds into the past.

Usage

```
games_ga(CC = 3, CD = 0, DC = 5, DD = 1, callC = TRUE,
  generations = 250, rounds = 100, num_opponents = 100,
  cross_prob = 0.05, mutation_prob = 0.05)
```

Arguments

CC	The number of points awarded to a focal agent when the focal agent and its opponent both cooperate
CD	The number of points awarded to a focal agent when the focal agent cooperates and its opponent defects
DC	The number of points awarded to a focal agent when the focal agent defects and its opponent cooperates
DD	The number of points awarded to a focal agent when the focal agent and its opponent both defect
callC	Whether or not the function calls c in the genetic algorithm. While not calling c is an option, the default value is TRUE because using the compiled c code greatly speeds up the genetic algorithm, making the whole program run much faster
generations	The number of generations the genetic algorithm will run before returning selected genotypes and fitness history. Each generation includes some number of rounds of the game that each strategy will play
rounds	The number of rounds of the game that a focal player will play against its opponent before moving on to the next opponent. Rounds are played iteratively against opponents, allowing the history of interactions to affect strategies and therefore total payoffs after all rounds are completed
num_opponents	The number of randomly selected opponents that a focal player will play during the course of one generation; the focal player will play in the same number of rounds with each opponent
cross_prob	A uniform probability of random crossing over event for a focal player's locus with the same locus from another randomly selected player.
mutation_prob	The probability that a given locus will mutate; mutation from 'C' to 'D' occurs with the same probability as 'D' to 'C' (no bias)

Value

A list, the elements of which include: 1. A table of the genomes of strategies and their frequencies in the population and 2. The mean fitness calculated over all players population in each generation. Fitness of one player is the number of points accrued over all rounds and opponents in a generation.

Examples

```
games_ga(CC = 3, CD = 0, DC = 5, DD = 1, generations = 100, rounds = 100)
```

games_ga_gui *Games genetic algorithm (GUI):*

Description

This function launches a graphical user interface (GUI) within a browser for the games_ga function, which runs a genetic algorithm that identifies sequential strategies for maximising payoffs given any two by two symmetrical payoff matrix. Simulated players remember three rounds into the past. Outputs include a table of survivng adaptive strategies and a plot of mean strategy fitness over generations of the genetic algorithm. No arguments are required (or available) for this function – it only launces the GUI.

Usage

```
games_ga_gui()
```

Value

A table (visible in a browser), the elements of which include: 1. A table of the genomes of strategies and their frequencies in the population and 2. A plot showing the mean fitness calculated over all players in the population in each generation. The fitness of one player is the number of points accrued over all rounds and opponents in a generation.

get_fitness *Fitness function*

Description

This function assesses the fitness of each strategy through the use of a sequential game between each focal strategy and a fixed number of random opponents.

Usage

```
get_fitness(history, agents, payoffs, num_opponents, rounds, useC)
```

Arguments

history	A table of all possible prior moves of agents in sequence
agents	A list of agents whose fitness will be assessed
payoffs	A vector of the payoffs for CC (payoffs[1]), CD (payoffs[2]), DC (payoffs[3]) and DD (payoffs[4]) combinations of play choices from the focal player and the opponent
num_opponents	The number of random opponents to match the focal agent against
rounds	The number of rounds that will be played
useC	A TRUE or FALSE value that determines whether or not c will be called to calculate agent fitnesses

Value

fitness A vector in which elements correspond to the accumlated fitness (payoffs) of each agent

---

`mutation` *Mutation function*

Description

This function causes alleles to mutate in agents.

Usage

```
mutation(agents, prob = 0.01)
```

Arguments

`agents`	A list of agents on which mutation will occur
`prob`	The probability that a mutation will occur for any locus

Value

agents The list of agents after mutation

`PD` *PD function*

Description

Returns the number of points (payoff) that a focal player accumlates from one round of a game with an opponent given a payoff vector given a decision "C" or "D" for each player

Usage

```
PD(a1_play, a2_play, payoffs)
```

Arguments

`a1_play`	The play choice of the focal player (0 or 1)
`a2_play`	The play choice of the opponent player (0 or 1)
`payoffs`	A vector of payoffs (length = 4) to the focal player as a consequence of both players playing 0 (payoffs[1]), the focal player only playing 0 (payoffs[2]), the focal player only playing 1 (payoffs[3]), and both players playing 1 (payoffs[4])

Value

fitness The payoff to the focal player (points accumulated from the interaction)

sample_round *Sample round function*

Description

This funciton simulates a sequence of rounds of a game played between two selected agents.

Usage

```
sample_round(agents, foc_agent, opp_agent, pay, rounds = 100)
```

Arguments

agents	A list of agents, two of which will play the game
foc_agent	The index of the first agent that will play the game
opp_agent	The index of the second agent that will play the game
pay	A vector of the payoffs for CC (payoffs[1]), CD (payoffs[2]), DC (payoffs[3]) and DD (payoffs[4]) combinations of play choices from the focal player and the opponent
rounds	The number of rounds that the agents will play the game

Value

round_hist The history of the game played between both agents

tournament *Tournament function*

Description

This function simulates selection of next generation of agents according to their fitness.

Usage

```
tournament(agents, fitness)
```

Arguments

agents	A list of agents to be assessed by fitness
fitness	The fitness vector on which agents will be assessed. Each element in the vector identifies a single agent's fitness

Value

agents A new list of agents selected according to fitness

Index