Package 'genetic.algo.optimizeR'

May 8, 2026

Type Package

Title Genetic Algorithm Optimization

Version 0.3.3

Description Genetic algorithm are a class of optimization algorithms inspired by the process of natural selection and genetics. This package is for learning purposes and allows users to optimize various functions or parameters by mimicking biological evolution processes such as selection, crossover, and mutation. Ideal for tasks like machine learning parameter tuning, mathematical function optimization, and solving an optimization problem that involves finding the best solution in a discrete space.

License MIT + file LICENSE

URL https://danymukesha.github.io/genetic.algo.optimizeR/, https://github.com/danymukesha/genetic.algo.optimizeR

BugReports https://github.com/danymukesha/genetic.algo.optimizeR/issues

Imports dplyr, ggplot2, magrittr, rsconnect, stats, stringr, tinytex, biocViews, DiagrammeR

Suggests BiocStyle, knitr, learnr, rmarkdown, spelling, testthat (>= 3.0.0)

Depends R (>= 4.1.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

Language en-US

biocViews ExperimentalDesign, Technology

NeedsCompilation no

Author Dany Mukesha [aut, cre] (ORCID: <https://orcid.org/0009-0001-9514-751X>)

Maintainer Dany Mukesha <danymukesha@gmail.com>

Repository CRAN

Date/Publication 2025-01-24 18:30:07 UTC

Contents

`crossover` *Crossover of selected parents from the fitting population*

Description

This function performs crossover between the selected individuals that fit the best based on the predefined condition(aim/objective).
e.g.: To optimize the function $f(x) = x^2 - 4x + 4$ to find the value of x that minimizes the function.
x: represents a possible value the an individual from the population can have.

Usage

```
crossover(selected_parents, offspring_size)
```

Arguments

`selected_parents`	The list of selected individuals from the population.
`offspring_size`	The number of offspring that the selected population should have.

Value

The output expected should be a list of offspring for the next generation.

Author(s)

Dany Mukesha

Examples

```
population <- c(1, 3, 0)

# Evaluate fitness
fitness <- evaluate_fitness(population)
print("Evaluation:")
print(fitness)

# Selection
selected_parents <- selection(population, fitness, num_parents = 2)
print("Selection:")
print(selected_parents)

# Crossover
offspring <- crossover(selected_parents, offspring_size = 2)
print("Crossover:")
print(offspring)
```

`evaluate_fitness` *Evaluating the fitness of the population*

Description

The function described below applies this equation:

$$f(x) = x^2 - 4x + 4$$

It assesses every individual x of a given population, to then provide an overview of how that population fits the equation.

Usage

```
evaluate_fitness(population)
```

Arguments

`population`	The list of individuals of the population.

Value

The output expected should be a list of $f(x)$ values calculated from individuals in the population.

Author(s)

Dany Mukesha

Examples

```
# example of usage
population <- c(1, 3, 0)
# Evaluate fitness
genetic.algo.optimizeR::evaluate_fitness(population)
```

`initialize_population` *Initialize population*

Description

The function described below creates a population of individuals(values).

Usage

```
initialize_population(population_size, min = -100, max = 100)
```

Arguments

`population_size`	The number of individuals of the population.
`min`	The minimum number an individual could have.
`max`	The maximum number an individual could have.

Details

(Note: the values are random and the population should be highly diversified)
The space of x value is kept integer type and on range from 0 to 3, for simplification.

Value

The output expected should be a list of individuals of the population with the size indicated in the input.

Author(s)

Dany Mukesha

Examples

```
# example of usage
genetic.algo.optimizeR::initialize_population(population_size = 3, min = 0, max = 3)
```

mutation *Mutating the offspring*

Description

The function described below mutates offspring from the selected individuals that fit the best based on the predefined condition(aim/objective).
e.g.: To optimize the function $f(x) = x^2 - 4x + 4$ to find the value of x that minimizes the function.
x: represents a possible value the an individual from the population can have.

Usage

```
mutation(offspring, mutation_rate)
```

Arguments

offspring	The list of offspring.
mutation_rate	The probably of a single offspring to be modified/mutated.

Details

The mutation is needed to increase the diversity in the population and help the next generation close to the fitness.

Value

The output expected should be a list of mutated offspring.

Author(s)

Dany Mukesha

Examples

```
# example of usage
population <- c(1, 3, 0)

# Evaluate fitness.
fitness <- genetic.algo.optimizeR::evaluate_fitness(population)
print("Evaluation:")
print(fitness)

# Selection
selected_parents <- genetic.algo.optimizeR::selection(population, fitness, num_parents = 2)
print("Selection:")
print(selected_parents)

# Crossover
offspring <- genetic.algo.optimizeR::crossover(selected_parents, offspring_size = 2)
```

```
print("Crossover:")
print(offspring)

# Mutation
mutated_offspring <- genetic.algo.optimizeR::mutation(offspring, mutation_rate = 0)
# (no mutation in this example)
print(mutated_offspring)
```

`replacement` *Replacing non-selected individual(s)*

Description

This function replace the individual(s) that was/were not selected (i.e. not the best fit) based on the predefined condition(aim/objective).
e.g.: To optimize the function $f(x) = x^2 - 4x + 4$ to find the value of x that minimizes the function.
x: represents a possible value the an individual from the population can have.

Usage

```
replacement(population, offspring, num_to_replace)
```

Arguments

`population`	The list of individuals of the population
`offspring`	The list of offspring.
`num_to_replace`	The number of selected individuals that should be replaced in the population.

Value

The output expected should be a list of selected individuals that fit the best with the predefined aim.

Author(s)

Dany Mukesha

Examples

```
# example of usage
population <- c(1, 3, 0)

# Evaluate fitness
fitness <- genetic.algo.optimizeR::evaluate_fitness(population)
print("Evaluation:")
print(fitness)

# Selection
selected_parents <- genetic.algo.optimizeR::selection(population, fitness, num_parents = 2)
```

```
print("Selection:")
print(selected_parents)

# Crossover and mutation
offspring <- genetic.algo.optimizeR::crossover(selected_parents, offspring_size = 2)
mutated_offspring <- mutation(offspring, mutation_rate = 0) # (no mutation in this example)
print(mutated_offspring)

# Replacement
population <- genetic.algo.optimizeR::replacement(population, mutated_offspring, num_to_replace = 1)
print("Replacement:")
print(population)
```

`selection` *Selecting the fitting the population*

Description

The function described below selects the individuals that fit the best based on the predefined condition(aim/objective).
e.g.: To optimize the function $f(x) = x^2 - 4x + 4$ to find the value of x that minimizes the function.
x: represents a possible value the an individual from the population can have.

Usage

```
selection(population, fitness, num_parents)
```

Arguments

`population`	The list of individuals of the population
`fitness`	The list of individuals(value) obtained from the function of `genetic.algo.optimizeR` namely 'evaluate_fitness'.
`num_parents`	The number of selected individuals that fit the best with the predefined aim.

Value

The output expected should be a list of selected individuals that fit the best with the predefined condition(aim/objective).

Author(s)

Dany Mukesha

Examples

```
# example of usage
library(genetic.algo.optimizeR)

population <- c(1, 3, 0)

# Evaluate fitness
fitness <- genetic.algo.optimizeR::evaluate_fitness(population)
print("Evaluation:")
print(fitness)

# Selection
selected_parents <- genetic.algo.optimizeR::selection(population, fitness, num_parents = 2)
print("Selection:")
print(selected_parents)
```

Index