Package 'MOGAMUN'

November 12, 2026

Type Package

Title MOGAMUN: A Multi-Objective Genetic Algorithm to Find Active Modules in Multiplex Biological Networks

Version 1.22.0

Description MOGAMUN is a multi-objective genetic algorithm that identifies active modules in a multiplex biological network. This allows analyzing different biological networks at the same time. MOGAMUN is based on NSGA-II (Non-Dominated Sorting Genetic Algorithm, version II), which we adapted to work on networks.

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData TRUE

Imports stats, utils, RCy3, stringr, graphics, grDevices, RUnit, BiocParallel, igraph

Suggests knitr, markdown

biocViews SystemsBiology, GraphAndNetwork, DifferentialExpression, BiomedicalInformatics, Transcriptomics, Clustering, Network

URL https://github.com/elvanov/MOGAMUN

RoxygenNote 7.1.1

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/MOGAMUN

git_branch RELEASE_3_23

git_last_commit a42eb3d

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Elva-María Novoa-del-Toro [aut, cre] (ORCID: <https://orcid.org/0000-0002-6135-5839>)

Maintainer Elva-María Novoa-del-Toro <elvanov@hotmail.com>

Contents

mogamun_init 2
mogamun_load_data 3
mogamun_postprocess 4
mogamun_run 6

Index **8**

`mogamun_init` *mogamun_init*

Description

initialize evolution parameters

Usage

```
mogamun_init(
    Generations = 500,
    PopSize = 100,
    MinSize = 15,
    MaxSize = 50,
    CrossoverRate = 0.8,
    MutationRate = 0.1,
    JaccardSimilarityThreshold = 30,
    TournamentSize = 2,
    Measure = "FDR",
    ThresholdDEG = 0.05,
    MaxNumberOfAttempts = 3
)
```

Arguments

`Generations`	number of generations to run (default = 500)
`PopSize`	number of subnetworks in the population (default = 100)
`MinSize`	minimum size (no. of nodes) of the subnetworks (default = 15)
`MaxSize`	maximum size (no. of nodes) of the subnetworks (default = 50)
`CrossoverRate`	rate for the crossover (default = 0.8)
`MutationRate`	rate for the mutation (default = 0.1)
`JaccardSimilarityThreshold`	subnetworks over this Jaccard similarity threshold are considered as duplicated (default = 30)
`TournamentSize`	size of the tournament (default = 2)
`Measure`	measure to calculate the nodes scores and to determine which genes are differentially expressed (possible values PValue and FDR, default = FDR)

ThresholdDEG threshold to consider a gene as significantly differerentially expressed. Note: if there is a logFC available, it is also considered |logFC|>1 (default = 0.05)

MaxNumberOfAttempts
maximum number of attempts to find compatible parents (default = 3)

Value

EvolutionParameters

Examples

```
EvolutionParameters <-
    mogamun_init(
        Generations = 1,
        PopSize = 10,
        MinSize = 15,
        MaxSize = 50,
        CrossoverRate = 0.8,
        MutationRate = 0.1,
        JaccardSimilarityThreshold = 30,
        TournamentSize = 2,
        Measure = "FDR",
        ThresholdDEG = 0.05,
        MaxNumberOfAttempts = 3
    )
```

---

mogamun_load_data *mogamun_load_data*

Description

Load the data to process

Usage

```
mogamun_load_data(
    EvolutionParameters,
    DifferentialExpressionPath,
    NodesScoresPath,
    NetworkLayersDir,
    Layers
)
```

Arguments

EvolutionParameters	evolution paramenters returned by mogamun_init()
DifferentialExpressionPath	full path to the differential expression results file (in CSV format). This file must contain at least the columns "gene" with the gene names, and ("PValue" or "FDR"). It can also contain "logFC"
NodesScoresPath	full path to an existing CSV file containing the nodes scores (columns "gene" and "nodescore"). NOTE. If the file does not exist, MOGAMUN will generate it in the provided path with the specified name
NetworkLayersDir	path of the folder that contains the networks that will be the layers of the multiplex. NOTE. Each file must start with a different digit
Layers	string of numbers, where the numbers correspond to the first character of the name of the network files (e.g. "123" builds a multiplex with layers 1, 2, and 3)

Value

List with the data to process

Examples

```
DEGPath <- system.file("extdata/DE/Sample_DE.csv", package = "MOGAMUN")
NodesScoresPath <-
    system.file("extdata/DE/Sample_NodesScore.csv", package = "MOGAMUN")
LayersPath <-
    paste0(system.file("extdata/LayersMultiplex", package = "MOGAMUN"), "/")
EvolutionParameters <- mogamun_init(Generations = 1, PopSize = 10)
LoadedData <-
    mogamun_load_data(
        EvolutionParameters = EvolutionParameters,
        DifferentialExpressionPath = DEGPath,
        NodesScoresPath = NodesScoresPath,
        NetworkLayersDir = LayersPath,
        Layers = "23"
    )
```

---

mogamun_postprocess *mogamun_postprocess*

Description

Postprocess the results: i) calculates the accumulated Pareto front, i.e. the individuals on the first Pareto front after re-ranking the results from multiple runs (NOTE. If there is a single run, the result is the set of individuals in the first Pareto front), ii) filters the networks to leave only the interactions between the genes that are included in the results, iii) generates some plots of interest, such as scatter

plots and boxplots, and iv) (optional) creates a Cytoscape file to visualize the results, merging the subnetworks with a Jaccard similarity coefficient superior to JaccardSimilarityThreshold (NOTE. Make sure to open Cytoscape if VisualizeInCytoscape is TRUE)

Usage

```
mogamun_postprocess(
    ExperimentDir = ".",
    LoadedData = LoadedData,
    JaccardSimilarityThreshold = 70,
    VisualizeInCytoscape = TRUE
)
```

Arguments

ExperimentDir	folder containing the results to be processed. It is the same folder specified as ResultsDir in mogamun_run
LoadedData	list returned by mogamun_load_data()
JaccardSimilarityThreshold	subnetworks over this Jaccard similarity threshold are merged in a single subnetwork
VisualizeInCytoscape	TRUE if you wish to visualize the accumulated Pareto front in Cytoscape, FALSE otherwise

Value

None

Examples

```
DEGPath <- system.file("extdata/DE/Sample_DE.csv", package = "MOGAMUN")
NodesScoresPath <-
    system.file("extdata/DE/Sample_NodesScore.csv", package = "MOGAMUN")
LayersPath <-
    paste0(system.file("extdata/LayersMultiplex", package = "MOGAMUN"), "/")
EvolutionParameters <- mogamun_init(Generations = 1, PopSize = 10)
LoadedData <-
    mogamun_load_data(
        EvolutionParameters = EvolutionParameters,
        DifferentialExpressionPath = DEGPath,
        NodesScoresPath = NodesScoresPath,
        NetworkLayersDir = LayersPath,
        Layers = "23"
    )
ResultsDir <- system.file("SampleResults", package="MOGAMUN")
mogamun_run(
    LoadedData = LoadedData,
    Cores = 1,
    NumberOfRunsToExecute = 1,
```

```
    ResultsDir = ResultsDir
)
mogamun_postprocess(
    ExperimentDir = ResultsDir,
    LoadedData = LoadedData,
    JaccardSimilarityThreshold = 70,
    VisualizeInCytoscape = FALSE
)
```

---

mogamun_run *mogamun_run*

Description

Run the algorithm with the specified values for the evolution parameters

Usage

```
mogamun_run(LoadedData, Cores = 1, NumberOfRunsToExecute = 1, ResultsDir = ".")
```

Arguments

LoadedData	list returned by mogamun_load_data()
Cores	to run MOGAMUN in parallel on the given number of cores (in line with the number of physical processor cores) (default = 1)
NumberOfRunsToExecute	number of runs (default = 1)
ResultsDir	outputs the results in the specified folder

Value

None

Examples

```
DEGPath <- system.file("extdata/DE/Sample_DE.csv", package = "MOGAMUN")
NodesScoresPath <-
    system.file("extdata/DE/Sample_NodesScore.csv", package = "MOGAMUN")
LayersPath <-
    paste0(system.file("extdata/LayersMultiplex", package = "MOGAMUN"), "/")
EvolutionParameters <- mogamun_init(Generations = 1, PopSize = 10)
LoadedData <-
    mogamun_load_data(
        EvolutionParameters = EvolutionParameters,
        DifferentialExpressionPath = DEGPath,
        NodesScoresPath = NodesScoresPath,
        NetworkLayersDir = LayersPath,
        Layers = "23"
```

```
    )
ResultsDir <- system.file("SampleResults", package="MOGAMUN")
mogamun_run(
    LoadedData = LoadedData,
    Cores = 1,
    NumberOfRunsToExecute = 1,
    ResultsDir = ResultsDir
)
```

Index

mogamun_init, 2
mogamun_load_data, 3
mogamun_postprocess, 4
mogamun_run, 6