Genetic Algorithm

Generated by Doxygen 1.8.17

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 func Namespace Reference

18 functions

Functions

- float schwefel (vector< float > &x)
- float firstDeJong (vector< float > &x)
- float rosenbrock (vector< float > &x)
- float rastrigin (vector< float > &x)
- float griewangk (vector< float > &x)
- float sineEnvelopeSineWave (vector< float > &x)
- float stretchedVSineWave (vector< float > &x)
- float ackleysOne (vector< float > &x)
- float ackleysTwo (vector< float > &x)
- float eggHolder (vector< float > &x)
- float rana (vector< float > &x)
- float pathological (vector< float > &x)
- float michalewicz (vector< float > &x)
- float mastersCosineWave (vector< float > &x)
- float quartic (vector< float > &x)
- float levy (vector< float > &x)
- float step (vector< float > &x)
- float alpine (vector< float > &x)

4.1.1 Detailed Description

18 functions

4.1.2 Function Documentation

4.1.2.1 ackleysOne()

```
float func::ackleysOne (
            vector< float > & x )
```

Function 8, Implementation of Ackley's One function

Parameters

x	descriptionx Vector of float

4.1.2.2 ackleysTwo()

```
float func::ackleysTwo (
            vector< float > & x )
```

Function 9, Implementation of Ackley's Two function

Parameters

x	descriptionx Vector of float

4.1.2.3 alpine()

```
float func::alpine (
            vector< float > & x )
```

Function 18, Implementation of Alpine function

Parameters

x	descriptionx Vector of float

4.1.2.4 eggHolder()

```
float func::eggHolder (
            vector< float > & x )
```

Function 10, Implementation of Egg Holder function

Parameters

x	descriptionx Vector of float

4.1.2.5 firstDeJong()

```
float func::firstDeJong (
             vector< float > & x )
```

Function 2, Implementation of 1st De Jong's function

Parameters

x	description x Vector of float

4.1.2.6 griewangk()

```
float func::griewangk (
             vector< float > & x )
```

Function 5, Implementation of Griewangk function

Parameters

x	descriptionx Vector of float

4.1.2.7 levy()

```
float func::levy (
             vector< float > & x )
```

Function 16, Implementation of Levy function

Parameters

x	descriptionx Vector of float

4.1.2.8 mastersCosineWave()

```
float func::mastersCosineWave (
             vector< float > & x )
```

Function 14, Implementation of Masters Cosine Wave function

Parameters

x	descriptionx Vector of float

4.1.2.9 michalewicz()

```
float func::michalewicz (
            vector< float > & x )
```

Function 13, Implementation of Michalewicz function

Parameters

x	descriptionx Vector of float

4.1.2.10 pathological()

```
float func::pathological (
            vector< float > & x )
```

Function 12, Implementation of Pathological function

Parameters

x	descriptionx Vector of float

4.1.2.11 quartic()

```
float func::quartic (
            vector< float > & x )
```

Function 15, Implementation of Quartic function

Parameters

x	descriptionx Vector of float

4.1.2.12 rana()

```
float func::rana (
            vector< float > & x )
```

Function 11, Implementation of Rana function

Parameters

x	descriptionx Vector of float

4.1.2.13 rastrigin()

```
float func::rastrigin (
            vector< float > & x )
```

Function 4, Implementation of Rastrigin's function

Parameters

x	descriptionx Vector of float

4.1.2.14 rosenbrock()

```
float func::rosenbrock (
            vector< float > & x )
```

Function 3, Implementation of Rosenbrock's function

Parameters

x	descriptionx Vector of float

4.1.2.15 schwefel()

```
float func::schwefel (
            vector< float > & x )
```

Function 1, Implementation of Schwefel’s function

Parameters

x	descriptionx Vector of float

4.1.2.16 sineEnvelopeSineWave()

```
float func::sineEnvelopeSineWave (
            vector< float > & x )
```

Function 6, Implementation of Sine Envelope Sine Wave function

Parameters

x	descriptionx Vector of float

4.1.2.17 step()

```
float func::step (
            vector< float > & x )
```

Function 147 Implementation of Step function

Parameters

x	descriptionx Vector of float

4.1.2.18 stretchedVSineWave()

```
float func::stretchedVSineWave (
            vector< float > & x )
```

Function 7, Implementation of Stretched V Sine Wave function

Parameters

x	descriptionx Vector of float

Chapter 5

Class Documentation

5.1 DataStats Class Reference

Public Member Functions

- void run ()

  Generate analytical data.

Public Attributes

- float **mean** = INT_MAX
- float **median** = INT_MAX
- float **stand** = INT_MAX
- float **range** [2]
- float **time** = 0
- vector$<$ float $>$ **data**

The documentation for this class was generated from the following files:

- src/data_stats.h
- src/data_stats.cpp

5.2 Genetic_algorithm Class Reference

Public Member Functions

- Genetic_algorithm (float($*$funct)(vector$<$ float $>$ &), float l, float u)

  Solution sets.
- void run ()

  Initialize genetic algorithm to run for a certain function.

Public Attributes

- float **f_best** = INT_MAX
- vector< float > **f_best_history**
- vector< float > solution_best

    Minimum f value during execution.

5.2.1 Constructor & Destructor Documentation

5.2.1.1 Genetic_algorithm()

```
Genetic_algorithm::Genetic_algorithm (
             float(*)(vector< float > &) funct,
             float l,
             float u )
```

Solution sets.

Initialize genetic algorithm to run for a certain function

Parameters

f	function
l	low bound for x
u	high bound for x

5.2.2 Member Function Documentation

5.2.2.1 run()

```
void Genetic_algorithm::run ( )
```

Initialize genetic algorithm to run for a certain function.

Run genetic algorithm. Status reset before every run.

The documentation for this class was generated from the following files:

- src/genetic_algorithm.h
- src/genetic_algorithm.cpp

5.3 Population Class Reference

```
#include <population.h>
```

Public Member Functions

- void init (int s, int d, float(∗funct)(vector< float > &), int l, int u)
- void generation ()

  Initialize a population.
- void evaluate ()

  Generate random number to fill the population.
- void reset ()

  Evaluate the population the population by first calculating each function's cost, then calculate fitness accordingly.
- void sort_by_cost ()

  Reset the population for new run.

Public Attributes

- vector< vector< float > > **population**
- vector< float > **cost**
- vector< float > **fitness**
- float **fitness_total**

5.3.1 Detailed Description

Population for genetic algorithm. Individual's population of x is represented by vector<float>.

5.3.2 Member Function Documentation

5.3.2.1 evaluate()

```
void Population::evaluate ( )
```

Generate random number to fill the population.

Evaluate the population the population by first calculating each function's cost, then calculate fitness accordingly.

5.3.2.2 generation()

```
void Population::generation ( )
```

Initialize a population.

Generate random number to fill the population.

5.3.2.3 init()

```
void Population::init (
             int s,
             int d,
             float(∗)(vector< float > &) funct,
             int l,
             int u )
```

Initialize a population

Parameters

s	population size
d	population dimension
low	x low bound
high	x high bound

5.3.2.4 reset()

```
void Population::reset ( )
```

Evaluate the population the population by first calculating each function's cost, then calculate fitness accordingly.

reset the population for new run

5.3.2.5 sort_by_cost()

```
void Population::sort_by_cost ( )
```

Reset the population for new run.

pair sort to sort the population by cost

The documentation for this class was generated from the following files:

- src/population.h
- src/population.cpp

Chapter 6

File Documentation

6.1 src/data_stats.h File Reference

data analysis

```
#include <vector>
```

Classes

- class DataStats

6.1.1 Detailed Description

data analysis

Author

Cheng Su (csu@cwu.edu)

Version

0.1

Date

2020-02-05

6.2 src/func.h File Reference

Math functions.

```
#include <stdio.h>
#include <vector>
```

Namespaces

- func

  18 functions

Functions

- float func::schwefel (vector< float > &x)
- float func::firstDeJong (vector< float > &x)
- float func::rosenbrock (vector< float > &x)
- float func::rastrigin (vector< float > &x)
- float func::griewangk (vector< float > &x)
- float func::sineEnvelopeSineWave (vector< float > &x)
- float func::stretchedVSineWave (vector< float > &x)
- float func::ackleysOne (vector< float > &x)
- float func::ackleysTwo (vector< float > &x)
- float func::eggHolder (vector< float > &x)
- float func::rana (vector< float > &x)
- float func::pathological (vector< float > &x)
- float func::michalewicz (vector< float > &x)
- float func::mastersCosineWave (vector< float > &x)
- float func::quartic (vector< float > &x)
- float func::levy (vector< float > &x)
- float func::step (vector< float > &x)
- float func::alpine (vector< float > &x)

6.2.1 Detailed Description

Math functions.

Author

Cheng Su (`csu@cwu.edu`)

Version

0.1

Date

2020-02-05

6.3 src/genetic_algorithm.h File Reference

Genetic algorithm.

```
#include <stdio.h>
#include <vector>
#include "genetic_algorithm.h"
#include "population.h"
```

Classes

- class Genetic_algorithm

6.3.1 Detailed Description

Genetic algorithm.

Author

Cheng Su (csu@cwu.edu)

Version

0.1

Date

2020-02-05

6.4 src/population.h File Reference

Population generation.

```
#include <stdio.h>
#include <vector>
```

Classes

- class Population

6.4.1 Detailed Description

Population generation.

Author

Cheng Su (csu@cwu.edu)

Version

0.1

Date

2020-02-05

6.5 src/run.h File Reference

Run result.

```
#include <stdio.h>
#include <vector>
#include "data_stats.h"
```

Functions

- void setSeed ()

  Set seed for `Mersenne Twister`
- DataStats runFunc (int experiment, string func_name, float(∗f)(vector< float > &), float max, float min)

  Run genetic algorithm for a certain function.
- void output (string func_name, DataStats result, vector< vector< float >> f_best_history)

  Write output file for record.
- void output_all (vector< DataStats > result_best)

  Write output file for all function results.

6.5.1 Detailed Description

Run result.

Author

Cheng Su (`csu@cwu.edu`)

Version

0.1

Date

2020-02-05

6.5.2 Function Documentation

6.5.2.1 output()

```
void output (
            string func_name,
            DataStats result,
            vector< vector< float >> f_best_history )
```

Write output file for record.

Write output file for record

Parameters

func_name	function name
result	result analysis data
f_best_history	minimum value for fx

6.5.2.2 output_all()

```
void output_all (
            vector< DataStats > result_best )
```

Write output file for all function results.

Write output file for all function results

Parameters

result_best	statistics for results for every function

6.5.2.3 runFunc()

```
DataStats runFunc (
            int experiment,
            string func_name,
            float(*)(vector< float > &) f,
            float min,
            float max )
```

Run genetic algorithm for a certain function.

Run genetic algorithm for a certain function

Parameters

experiment	experiment size
func_name	function name
min	low bound for x
max	high bound for x

Index