Lab 4

Generated by Doxygen 1.8.14

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 bound Struct Reference

`#include <populationV2.h>`

Public Attributes

- double **upper**
- double **lower**

3.1.1 Detailed Description

Defines bounds for use in fitness function calculations.

The documentation for this struct was generated from the following file:

- populationV2.h

3.2 Firefly Class Reference

`#include <firefly.h>`

Public Member Functions

- double runFirefly ()
- Firefly (int fitnessFunc, int ns, int dim, double upper, double lower, int iterations, double alpha, double beta0, double betaMin, double gamma)
- Firefly (string params)
- ~Firefly ()

Private Attributes

- int ns

  Solution Size.
- int dim

  Number of Dimensions.
- int fitnessFunc

  Fitness Function to be operated on.
- bound bounds

  Function Bounds.
- double **beta0**
- double **betaMin**
- double **gamma**
- double **alpha**
- int iterations

  Number of iterations to run the algorithm.

3.2.1 Detailed Description

This class handles the execution of the firefly algorithm

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Firefly() [1/2]

```
Firefly::Firefly (
            int fitnessFunc,
            int ns,
            int dim,
            double upper,
            double lower,
            int iterations,
            double alpha,
            double beta0,
            double betaMin,
            double gamma )
```

Initialize Class in code

Parameters

fitnessFunc	int
ns	int
dim	int
upper	double
lower	double
iterations	int
alpha	double
beta0	double
betaMin	double
gamma	double

3.2.2.2 Firefly() [2/2]

```
Firefly::Firefly (
            string params )
```

Initialize Class using file param

Parameters

param	string

3.2.2.3 ~Firefly()

```
Firefly::~Firefly ( )  [inline]
```

Default Destructor

3.2.3 Member Function Documentation

3.2.3.1 runFirefly()

```
double Firefly::runFirefly ( )
```

Run an instance of firefly and return the best fitness

Returns

double

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

- firefly.h
- firefly.cpp

3.3 ParticleSwarm Class Reference

```
#include <particleSwarm.h>
```

Public Member Functions

- double runParticleSwarm ()
- ParticleSwarm (int fitnessFunc, int ns, int dim, double upper, double lower, int iterations, double c1, double c2)
- ParticleSwarm (string param)
- ~ParticleSwarm ()

Private Member Functions

- void particleRow (ThreadPointers tp)
- void allocateVelocity (double **&v)

Private Attributes

- int ns

  Population Size.
- int dim

  Dimensions.
- int fitnessFunc

  Fitness Function.
- bound bounds

  Fitnes Function Bounds.
- int iterations

  Number of Iterations.
- double **c1**
- double c2

  Personal Best, Global Best Modifiers.
- mutex lock

  Thread Lock.
- condition_variable cv

  Locks all threads to stay on same iteration and update Global best.
- int activeThreads

  Counts threads still running.

3.3.1 Detailed Description

The particle swarm class handles the data and threads for the Particle Swarm Algorithm

3.3.2 Constructor & Destructor Documentation

3.3.2.1 ParticleSwarm() [1/2]

```
ParticleSwarm::ParticleSwarm (
            int fitnessFunc,
            int ns,
            int dim,
            double upper,
            double lower,
            int iterations,
            double c1,
            double c2 )
```

Initialize Class in code

Parameters

fitnessFunc	int
ns	int
dim	int
upper	double
lower	double
iterations	int
c1	double
c2	double

3.3.2.2 ParticleSwarm() [2/2]

```
ParticleSwarm::ParticleSwarm (
            string param )
```

Initialize Class using file param

Parameters

param	string

3.3.2.3 ~ParticleSwarm()

```
ParticleSwarm::~ParticleSwarm ( )
```

Default Destructor

3.3.3 Member Function Documentation

3.3.3.1 allocateVelocity()

```
void ParticleSwarm::allocateVelocity (
            double **& v )  [private]
```

Create Velocity vector

Parameters

v	double**&

3.3.3.2 particleRow()

```
void ParticleSwarm::particleRow (
            ThreadPointers tp )  [private]
```

Update a single row of particles in the swarm.

Parameters

tp	ThreadPointers
this	ParticleSwarm

3.3.3.3 runParticleSwarm()

```
double ParticleSwarm::runParticleSwarm ( )
```

Run an instance of ParticleSwarm returns best fitness

Returns

double

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

- particleSwarm.h
- particleSwarm.cpp

3.4 Population Struct Reference

```
#include <populationV2.h>
```

Public Member Functions

- void generatePopulation (double min, double max)
- int getBestSolutionIndex ()
  Returns index of the best particle.
- void emptyPopulation ()
  Create a new blank population.
- void updatePopulationFitness ()
  Update the population fitness.
- void printPopulation ()
  Output the population to the console.
- void deletePopulation ()
  Delete the population.
- void sortByFitnessDecending ()
  Sort Highest Fitness First.
- void copyPopulation (Population *pop)
- Population (int populationSize, int dim, int fitnessFunc)
  Create the struct with population size, dim and f.

Public Attributes

- int populationSize

  Population Size.
- int dim

  Dimensions.
- int fitnessFunc

  Which fitness function will be used.
- double * fitness

  Fitnesses of the members in the population.
- double ** population = nullptr

  Population.

3.4.1 Detailed Description

Defines a population populationSize by dim.

Creates a population to be used in Differential Evolution and Genetic Algorithm. The population has populationSize members each with dim values in them.

3.4.2 Member Function Documentation

3.4.2.1 copyPopulation()

```
void Population::copyPopulation (
            Population * pop )
```

The constructor for Population. Sets the population size, dim and fitness function.

Parameters

populationSize	int
dim	int
fitnessFunc	int

3.4.2.2 generatePopulation()

```
void Population::generatePopulation (
            double min,
            double max )
```

Create a population with bounds min and max

Parameters

min	double
max	double

Returns

void

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

- populationV2.h
- populationV2.cpp

3.5 ThreadPointers Struct Reference

`#include <particleSwarm.h>`

Public Attributes

- Population * p
  Pointer to the current population.
- Population * pBest
  Pointer to personalBest.
- double ** v
  Particle Velocity.
- double * gBest
  Global Best.
- double * gBestFitness
  Global best Fitness.
- int iterations
  Number of Iterations.
- int row
  Row thread will be working on.
- mt19937_64 * rng
  Same number generator for rand functions.

3.5.1 Detailed Description

Defines data sent to threads in ParticleSwarm::runParticleSwarm()

The documentation for this struct was generated from the following file:

- particleSwarm.h

Chapter 4

File Documentation

4.1 benchmarksV2.cpp File Reference

Function definitions for the function headers in benchmarksV2.h.

```
#include "benchmarksV2.h"
#include <iostream>
#include <fstream>
```

Functions

- double **schwefels** (double input[ ], int dim)
- double **deJong1st** (double input[ ], int dim)
- double **rosenbrock** (double input[ ], int dim)
- double **rastrigin** (double input[ ], int dim)
- double **griewangk** (double input[ ], int dim)
- double **sineEnvelopSine** (double input[ ], int dim)
- double **stretchVSine** (double input[ ], int dim)
- double **ackleys1** (double input[ ], int dim)
- double **ackleys2** (double input[ ], int dim)
- double **eggHolder** (double input[ ], int dim)
- double **rana** (double input[ ], int dim)
- double **pathological** (double input[ ], int dim)
- double **michalewicz** (double input[ ], int dim)
- double **mastersCosine** (double input[ ], int dim)
- double **shekelsFoxholes** (double input[ ], int dim)

Variables

- double(∗ **benchmarkFunctions** [15])(double input[ ], int dim)

4.1.1 Detailed Description

Function definitions for the function headers in benchmarksV2.h.

4.1.2 Variable Documentation

4.1.2.1 benchmarkFunctions

```
double(* benchmarkFunctions[15])(double input[], int dim)
```

Initial value:

```
= { schwefels, deJong1st, rosenbrock, rastrigin, griewangk, sineEnvelopSine,
stretchVSine, ackleys1, ackleys2, eggHolder, rana, pathological,
michalewicz, mastersCosine, shekelsFoxholes }
```

4.2 benchmarksV2.h File Reference

Defines benchmark function headers.

```
#include <vector>
#include <math.h>
#include <time.h>
#include <chrono>
#include <string>
#include <random>
```

Functions

- double **schwefels** (double input[], int dim)
- double **deJong1st** (double input[], int dim)
- double **rosenbrock** (double input[], int dim)
- double **rastrigin** (double input[], int dim)
- double **griewangk** (double input[], int dim)
- double **sineEnvelopSine** (double input[], int dim)
- double **stretchVSine** (double input[], int dim)
- double **ackleys1** (double input[], int dim)
- double **ackleys2** (double input[], int dim)
- double **eggHolder** (double input[], int dim)
- double **rana** (double input[], int dim)
- double **pathological** (double input[], int dim)
- double **michalewicz** (double input[], int dim)
- double **mastersCosine** (double input[], int dim)
- double **shekelsFoxholes** (double input[], int dim)

Variables

- const double **FOXHOLEC** [30]
- const double **FOXHOLEA** [30][10]
- const int **NUMBENCHMARKS** = 15
- double(∗ **benchmarkFunctions** [NUMBENCHMARKS])(double input[], int dim)

4.2.1 Detailed Description

Defines benchmark function headers.

This file defines the benchmark function headers, as well as the constants for Shekel's Foxholes and benchmark↩ Functions, a pointer to the 15 benchmark functions.

4.2.2 Variable Documentation

4.2.2.1 FOXHOLEA

```
const double FOXHOLEA[30][10]
```

Initial value:

```
= { { 9.681,0.667,4.783,9.095,3.517,9.325,6.544,0.211,5.122,2.02 },
{ 9.4,2.041,3.788,7.931,2.882,2.672,3.568,1.284,7.033,7.374 },
{ 8.025,9.152,5.114,7.621,4.564,4.711,2.996,6.126,0.734,4.982 },
{ 2.196,0.415,5.649,6.979,9.510,9.166,6.304,6.054,9.377,1.426 },
{ 8.074,8.777,3.467,1.863,6.708,6.349,4.534,0.276,7.633,1.567 },
{ 7.650,5.658,0.720,2.764,3.278,5.283,7.474,6.274,1.409,8.208 },
{ 1.256,3.605,8.623,6.905,4.584,8.133,6.071,6.888,4.187,5.448 },
{ 8.314,2.261,4.24,1.781,4.124,0.932,8.129,8.658,1.208,5.762 },
{ 0.226,8.858,1.42,0.954,1.622,4.698,6.228,9.096,0.972,7.637 },
{ 7.305,2.228,1.242,5.928,9.133,1.826,4.06,5.204,8.713,8.247 },
{ 0.652,7.027,0.508,4.876,8.807,4.632,5.808,6.937,3.291,7.016 },
{ 2.699,3.516,5.847,4.119,4.461,7.496,8.817,0.69,6.593,9.789 },
{ 8.327,3.897,2.017,9.57,9.825,1.15,1.395,3.885,6.354,0.109 },
{ 2.132,7.006,7.136,2.641,1.882,5.943,7.273,7.691,2.88,0.564 },
{ 4.707,5.579,4.08,0.581,9.698,8.542,8.077,8.515,9.231,4.67 },
{ 8.304,7.559,8.567,0.322,7.128,8.392,1.472,8.524,2.277,7.826 },
{ 8.632,4.409,4.832,5.768,7.05,6.715,1.711,4.323,4.405,4.591 },
{ 4.887,9.112,0.17,8.967,9.693,9.867,7.508,7.77,8.382,6.74 },
{ 2.44,6.686,4.299,1.007,7.008,1.427,9.398,8.48,9.95,1.675 },
{ 6.306,8.583,6.084,1.138,4.350,3.134,7.853,6.061,7.457,2.258 },
{ 0.652,2.343,1.37,0.821,1.31,1.063,0.689,8.819,8.833,9.07 },
{ 5.558,1.272,5.756,9.857,2.279,2.764,1.284,1.677,1.244,1.234 },
{ 3.352,7.549,9.817,9.437,8.687,4.167,2.57,6.54,0.228,0.027 },
{ 8.798,0.88,2.37,0.168,1.701,3.68,1.231,2.39,2.499,0.064 },
{ 1.46,8.057,1.337,7.217,7.914,3.615,9.981,9.198,5.292,1.224 },
{ 0.432,8.645,8.774,0.249,8.081,7.461,4.416,0.652,4.002,4.644 },
{ 0.679,2.8,5.523,3.049,2.968,7.225,6.73,4.199,9.614,9.229 },
{ 4.263,1.074,7.286,5.599,8.291,5.2,9.214,8.272,4.398,4.506 },
{ 9.496,4.83,3.15,8.27,5.079,1.231,5.731,9.494,1.883,9.732 },
{ 4.138,2.562,2.532,9.661,5.611,5.5,6.886,2.341,9.699,6.5 } }
```

4.2.2.2 FOXHOLEC

```
const double FOXHOLEC[30]
```

Initial value:

```
= { 0.806,0.517,0.1,0.908,0.965,0.669,0.524,0.902,0.351,0.876,0.462,
0.491,0.463,0.741,0.352,0.869,0.813,0.811,0.0828,0.964,0.789,0.360,0.369,
0.992,0.332,0.817,0.632,0.883,0.608,0.326 }
```

4.3 firefly.cpp File Reference

Defines functions for the firefly class.

```
#include "firefly.h"
#include <fstream>
```

4.3.1 Detailed Description

Defines functions for the firefly class.

4.4 firefly.h File Reference

Definition for the Firefly class.

```
#include "populationV2.h"
```

Classes

- class Firefly

4.4.1 Detailed Description

Definition for the Firefly class.

Firefly Algorithm behaves like fireflies do while they mate. This class runs the firefly algorithm

4.5 particleSwarm.cpp File Reference

Defines functions for the ParticleSwarm class.

```
#include "particleSwarm.h"
#include <windows.h>
```

4.5.1 Detailed Description

Defines functions for the ParticleSwarm class.

4.6 particleSwarm.h File Reference

Defines ParticleSwarm and Thread Pointers.

```
#include "populationV2.h"
#include <fstream>
#include <thread>
#include <condition_variable>
```

Classes

- struct ThreadPointers
- class ParticleSwarm

4.6.1 Detailed Description

Defines ParticleSwarm and Thread Pointers.

Index