Central Washington University, Department of Computer Science

CS471 Project 1

Nick Rohde

6^{th} April, 2018

1 Benchmark Functions

1.1 Overview

For this project, we implemented fifteen standard benchmarking functions for optimization algorithms; namely, Schwefel's Function, De Jong's 1^{st} Function, Rosenbrock's Saddle, Rastrigin's Function, Griewangk's Function, Sine Envelope Sine Wave Function, Stretch V Sine Wave Function, Ackley's Function (1), Ackley Function (2), Egg Holder, Rana's Function, Pathological Function, Michalewicz's Function, Master's Cosine Wave, and Schekel's Foxhole. These functions were implemented for to be used in benchmarking of algorithms for future projects.

1.2 Testing

Testing of these functions was done using a set of known optimal points, as well as random data. The results of the optimal points can be found in table 2.1; some functions do not have a known optimal point, thus, this table is only partially filled in. Results for the random data can be found in table 2.2. All random data was generated with the STL implementation of the mersenne twister (mt19937) and the numbers were generated in the range for each function. Furthermore, 100 samples were generated for all tests, and all statistical data displayed in table 2.2 is the average of the 100 data points.

2 Analysis

2.1 Optimal Points

Table 2.1 shows the results of the tests run with the known optimal points. Here we can see that the program correctly calculated all known global optima of the functions without error.

2.2 Random Data

On the other hand, the random data mostly generated garbage output (as is to be expected), however, the Sine Envelope Sine Wave function produced values fairly close to the optimums (-13.4235, -28.3385, -43.2535), also of note is that the standard deviation for this function was extremely small in all cases, with a minimum of 0.07 and a maximum of 14.05. Master's Cosine Wave also produced very clustered data with even smaller standard deviations than the Sine Envelope Sine Wave, ranging from 0.0031 to 0.04.

Another interesting part of the results is the execution time of the functions. All functions lost at most one magnitude in execution time when the input size was trippled, with some only increasing by a small fraction, namely De Jong's Function only increased by $2 \times 10^{-}7$ seconds when the dimensions went from 10 to 30.

Table 2.1: Optimal Point Results

Function	D_{10}	Expected	D_{20}	Expected	D_{30}	Expected
f_1	-4189.83	-4189.83	-8379.66	-8379.66	-12569.5	-12569.5
f_2	0	0	0	0	0	0
f_3	0	0	0	0	0	0
f_4	-2000	-2000	-4000	-4000	-6000	-6000
f_5	0	0	0	0	0	0
f_6	*	*	*	*	*	*
f_7	0	0	0	0	0	0
f_8	*	*	*	*	*	*
f_9	0	0	0	0	0	0
f_{10}	*	*	*	*	*	*
f_{11}	*	*	*	*	*	*
f_{12}	*	*	*	*	*	*
f_{13}	*	*	*	*	*	*
f_{14}	-9	-9	-19	-19	-29	-29
f_{15}	*	*	*	*	*	*

** denotes that the optimal point is unknown*

Table 2.2: Computation comparison for 10, 20, and 30 Dimensions

Problem	D_{10}					D_{20}					D_{30}				
	Avg	Median	Range	SD	T(s)	Avg	Median	Range	SD	T(s)	Avg	Median	Range	SD	T(s)
f_1	-61.82	-46.80	3177.65	596.77	2.77×10^{-7}	-147.22	1.73	4926.18	917.83	6.63×10^{-7}	-303.71	-268.39	5861.72	1131.09	1.16×10^{-6}
f_2	859142	838788	1.75×10^{6}	237719	1.40×10^{-7}	2.64×10^{6}	1.81×10^{6}	3.20×10^{6}	925823	$2.29 \times 10-7$	5.21×10^{6}	2.62×10^{6}	5.08×10^{6}	2.67×10^{6}	3.39×10^{-7}
f_3	1.18×10^{13}	1.11×10^{13}	3.30×10^{13}	5.07×10^{12}	2.47×10^{-8}	3.79×10^{13}	2.63×10^{13}	6.22×10^{13}	1.47×10^{13}	9.58×10^{-8}	7.79×10^{13}	4.08×10^{13}	9.09×10^{13}	3.92×10^{13}	1.88×10^{-7}
f_4	870095	862871	1.84×10^{6}	242235	1.23×10^{-7}	2.62×10^{6}	1.77×10^{6}	3.85×10^{6}	942880	4.96×10^{-7}	5.22×10^{6}	2.60×10^{6}	5.41×10^{6}	2.66×10^{6}	1.02×10^{-6}
f_5	219.84	207.35	441.22	69.07	3.83×10^{-7}	661.70	451.62	871.89	238.64	6.84×10^{-6}	1315.61	669.64	1229.62	668.70	2.32×10^{-6}
f_6	-4.52	-4.50	9.47	0.07	1.40×10^{-7}	-14.05	-9.50	19.63	4.52	5.44×10^{-7}	-28.60	-14.51	29.41	14.05	1.13×10^{-6}
f_7	260.92	260.03	516.79	30.19	1.09×10^{-6}	809.14	554.86	1058.76	265.18	3.18×10^{-6}	1658.45	849.75	1709.94	810.81	6.28×10^{-6}
f_8	2946.27	2904.35	5900.96	484.65	3.45×10^{-7}	9053.53	6136.60	12351	3028.79	$9.64 \times 10-7$	18323.30	9219.26	18275.40	9106.18	1.84×10^{-6}
f_9	194.55	194.82	386.62	2.45	5.54×10^{-7}	604.25	409.68	820.07	194.57	1.63×10^{-6}	1230.71	626.90	1251.74	604.26	3.31×10^{-6}
f_{10}	1.33	-15.76	4025.12	785.80	6.77×10^{-7}	-12.44	17.68	6478.28	1215.58	1.98×10^{-6}	-168.11	-59.29	7074.71	1604.07	3.98×10^{-6}
f_{11}	216.19	231.67	3699.49	675.03	4.89×10^{-7}	644.44	454.87	4076.84	905.86	1.45×10^{-6}	1225.32	720.60	5010.82	1270.95	2.85×10^{-6}
f_{12}	4.50	4.50	9.04	0.11	3.49×10^{-7}	13.98	9.50	18.95	4.50	1.03×10^{-6}	28.48	14.50	28.93	13.99	2.05×10^{-6}
f_{13}	-0.0736	-0.0024	4.13	0.80	9.44×10^{-7}	-0.0119	0.0964	7.29	1.18	2.73×10^{-6}	0.0929	0.0393	7.19	1.35	5.41×10^{-6}
f_{14}	0.0031	0	0.41	0.04	9.23×10^{-7}	0.0031	0	1.18×10^{-26}	0.0031	2.69×10^{-6}	0.0031	0	5.00×10^{-13}	0.0031	5.31×10^{-6}
f_{15}	-3.35×10^{-5}	-3.16×10^{-5}	0.00011	9.98×10^{-6}	3.25×10^{-7}	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Mean	7.87×10^{11}	7.40×10^{11}	3.43×10^{12}	3.38×10^{11}	4.52×10^{-7}	2.71×10^{12}	1.88×10^{12}	4.44×10^{12}	1.05×10^{12}	1.75×10^{-7}	5.56×10^{12}	2.91×10^{12}	6.49×10^{12}	2.80×10^{12}	2.66×10^{-6}

[1] 3.2GHz AMD Ryzen 7 1700X, 16 GB RAM