**SPRING SEMESTER 2021/22**

**COMP2024 Coursework**

|  |  |
| --- | --- |
| Report Type | Individual Reflection |
| Date | 28/04/2022 |
| Group | 6 |
| Name | Tan Zhun Xian |
| Student ID | 20313854 |
| Supervisors | * Tissa Chandesa * Siang Yew Chong |

**Initial Preparations**

As the group leader of the team, I was tasked with coordinating the competition between my group members. Before starting the competition, some initial preparations were needed. To download the source code to be used in the competition, I went to the bbob2010 website [1] specified by the lecturer. The competition is run using the matlab version of the code. I first had to make some changes to the source code so that it fits the specifications of the coursework.

Firstly, the coursework is only run at 5-Dimensions (2-D,3-D,5-D,10-D and 20-D), so I deleted the 40-D in the exampleexperiment.m and exampletiming.m files. Secondly, I downloaded the modified fgeneric.m file from Moodle and replaced the original fgeneric.m file with it. Finally, I set the initial seed of the code to be my student ID which is 20313854. This serves 2 main purposes, which is to ensure fairness among the team members during the competition and to ensure that the results are easily reproduceable. With all the necessary changes made to the code, I sent the code as a template to be used by my group members. All they had to do was to replace the MY\_OPTIMIZER.m file with their selected optimizer and to put the information of their optimizers in the code.

**Optimizers**

Initially, the 4 chosen optimizers were Stochastic Hill-Climbing, CMA-ES, Differential Evolution and IPOP-ACTCMA-ES. As time went on, the chosen optimizers were changed as better optimizers were found. In the end, the group settled on BIPOP-CMA-ES, DEAE, Genetic Algorithm and PSO\_Bounds as the 4 optimizers to be used in the competition.

After setting a deadline for the team, I could then rest easy and wait for their submissions. I am pleased to find that my group was very cooperative and responsive. They submitted their selected optimizers and the results on time to be tested and compared. They also sent in their literature review to be included in the group report.

**Competition**

After all the optimizers and their results were sent in, the competition can officially begin. By reviewing the results, I am pleased to find that results sent and my own results were the same. This shows that the results are genuine. The results were also easily reproducible as the same seed was used in run of the code so the same results were obtained each time.

To extract the results quickly and easily from the code, I wrote a Java program to automate the process. The Java program extracted the results and placed them in 5 distinct blocks of size 24 x 15. This is because the code was run with 24 different functions 15 times in 5 dimensions. The program was run 4 times to extract the results from the 4 group members.

Now that I have the raw results, I can start filling in the fsmap. The Java program above was modified to automatically replace any values smaller than 1.00E-14 with 1.00E-14 and any values greater than 1.00E+03 to 1.00E+03 and save it to a different file. In essence, all the modified values are now between the range of [1.00E-14,1.00E+03]. After modifying the values, the data was copied and pasted into the fsmap of each member. Finally, the total fsmap values was calculated. In the end, BIPOP-CMA-ES won the competition with a score of 10807. DEAE missed first place by a small margin with a score of 10361. It was followed by PSO\_Bounds with a score of 4266. Last place went to GA with a shockingly low score of only 81. So, this concludes the competition.

**Data Collection**

Now that the competition has concluded, I can finally focus on writing the group report. To write a comprehensive group report, I believe I had to collect as much data as I could. Firstly, I used the original results from the competition to make boxplots of the data. We naturally have an idea of the spread of the data by just observing the shape of the boxplots. However, I soon realised that the resulting boxplots were insufficient to convey my point and decided on collecting more data.

Next, I wrote another Java program to automatically calculate the average and standard deviations of the data. This is achieved by using the 15 data points generated in each function evaluation since a function was always run 15 times. I used the data collected to compare the optimizers next to each other. This is done by placing the average and STD of the results of each optimizer next to each other for easy comparisons. This was done for 2-D, 3-D, 5-D, 10-D, 20-D and finally the overall results which is the cumulation of all the results. The best (lowest) results were bolded for easy viewing and the total best (lowest) results were calculated below.

However, I then realised that people were not machines and could not draw any conclusions just by looking at a table full of numbers. So, I used matlab to convert my results to 24 graphs, one for each function, with dimensions on the x-axis and results on the y-axis. This was to allow reader to be able to easily see the changes in results as the number of dimension changes. Matlab was run twice to generate the graphs for average and STD, ending up with 48 graphs.

**Post-Processing**

After I was satisfied with the data collection, I decided to move onto the last step before writing the report which is the post-processing. The post-processing code [1] was provided alongside the source code in the same website. However, certain programs have to be installed before it can be used.

Firstly, a deprecated version of Python (2.x) was needed to run the code. This was because the code was written in Python. The deprecated part comes from the fact that the code was written a decade ago and thus used an older version of Python. Secondly, libraries for Python such as Numpy and Matplotlib were also needed. Finally, a latex program was also needed to generate the reports and thus I went with MiKTeX.

After installing all the required programs, I could finally start the post-processing process. First, I generated the post-processing report for single optimizers for all 4 optimizers. Then, I generate the post-processing report of the comparison of the 4 optimizers. Unfortunately, I was unable to generate post-processing report for the comparison between 2 optimizers. This was due to an error in the code which I was unable to fix because the code is a decade old at this point and there is no one maintaining the code anymore. So, I was unable to compare the top 2 optimizers (BIPOP-CMA-ES and DEAE) and the bottom 2 optimizers (GA and PSO\_Bounds). Nevertheless, with the current post-processing reports on hand, I was finally ready to write the group report.

**Group Report**

Firstly, I made the cover page of the report and include any relevant information in it. This was followed by a table of contents. Then, I gave the definition of a metaheuristic and explained its three main types. This was then followed by the literature review of the 4 optimizers. After that, the optimizer configurations for the competition were included. This finishes the part of the code explaining the optimizers.

Then, I showed the results of the competition by showing the average and STD graphs. This was followed by the full results table below. Links to the boxplots were included as they were deemed to be important enough to be included in the report but not important enough to be shown in the main report.

Next, the observations of the results were given. This was followed by the analysis of the results and then the post-processing. The 4 single post-processing reports and the report on the comparison of the 4 optimizers were also linked in the report. Then, the empirical cumulative distribution (ECDF) of the 4 optimizers for 5-D and 20-D were shown and some observation and analysis were given. Finally, the conclusion and references were given. This concludes the group report.

**Reference**

[1] <https://web.archive.org/web/20210511050037/http://coco.gforge.inria.fr/doku.php?id=bbob-2010-downloads>

[2] <https://numbbo.github.io/data-archive/bbob/>

[3] <http://www.python.org/download/releases/>

[4] <http://sourceforge.net/projects/numpy/>

[5] <http://sourceforge.net/projects/matplotlib/>

[6]