

# Advanced Optimization – Final Report

Nevergrad : Adding Multi-objective Benchmarks

Shuqi Deng Master in Data Science and Business Analytics Essec Business School and CentraleSupélec shuqi.deng@studen t-cs.fr	Anna Parshikova Master in Data Science and Business Analytics Essec Business School and CentraleSupélec anna.parshikova@ student-cs.fr	Chia-tien Tang Master in Data Science and Business Analytics Essec Business School and CentraleSupélec chiatien.tang@stu dent-cs.fr	Yuchen Wang Master in Data Science and Business Analytics Essec Business School and CentraleSupélec yuchen.wang@stud ent-cs.fr	Yingxuan Wu Master in Data Science and Business Analytics Essec Business School and CentraleSupélec yingxuan.wu@stud ent-cs.fr
--	--	---	--	--

## Table of Contents

<b>I. Introduction .....</b>	<b>2</b>
<b>A. Problem Definition .....</b>	<b>2</b>
<b>B. Importance of the issue.....</b>	<b>2</b>
<b>C. Potential Applications .....</b>	<b>3</b>
<b>II. Methodology.....</b>	<b>4</b>
<b>A. Identify and Select Benchmarks .....</b>	<b>4</b>
<b>B. Develop Dedicated Classes.....</b>	<b>4</b>
<b>C. Details of the code files: .....</b>	<b>5</b>
<b>D. Algorithm.....</b>	<b>6</b>
<b>E. Connections with prior work.....</b>	<b>6</b>
<b>III. Documentation.....</b>	<b>7</b>
<b>IV. Evaluation.....</b>	<b>9</b>
<b>A. Correctness of Benchmark Functions.....</b>	<b>9</b>
<b>B. Test Coverage.....</b>	<b>10</b>
<b>C. Other Evaluation.....</b>	<b>10</b>
<b>D. Summary_Evaluation .....</b>	<b>11</b>
<b>V. Summary.....</b>	<b>11</b>
<b>VI. Contribution.....</b>	<b>11</b>
<b>VII. Reference &amp; Appendix .....</b>	<b>12</b>

# I. Introduction

This report will focus on the integration of native multi-objective benchmarks in the Nevergrad library. Acknowledging the inherent complexity of real-world optimization problems, we find it crucial to shift from solving single problems to tackling a variety of multi-objective benchmarks as the real-world situations often involve more than one goal. By embracing diverse benchmarks, we can better simulate these complexities by allowing us to evaluate optimization tools in scenarios that better mirror practical decision-making. In essence, broadening the scope is vital for ensuring our optimization solutions are versatile and effective in addressing the diverse and multi-faceted problems. As stated in the original Nevergrad problem #1059 we got inspiration from the suggested web source on multi-objective optimization test functions. As for the key features, they include the deprecation of the Multiobjective Function class in favour of the ConstrainedMultiObjective class, the categorization of benchmarks, and the continued use of the Differential Evolution optimizer. In the end, our collaborative effort extends the library's capacity to evaluate optimization algorithms across a spectrum of realistic scenarios, enhancing its applicability in diverse domains.

## A. Problem Definition

For this project, we chose to work on the Nevergrad problem #1059 related to the enhancement and diversification of multi-objective optimization benchmarks within the Nevergrad library. The existing benchmarks were initially based on mono-objective functions and considering the existing limitation (as they oversimplify complex real-world problems by focusing on a single objective, potentially overlooking the nuanced and multifaceted nature of decision-making scenarios), there was a need to transition to more "native" multi-objective benchmarks. The process will include but not be limited to identifying and selecting the benchmarks that cover a diverse range of characteristics, such as different dimensions, complexities, and features; as well as develop dedicated classes for users to adopt the newer approach of passing a multiobjective loss directly to the optimization process.

## B. Importance of the issue

In the realm of decision-making, particularly in fields like engineering design, finance, and logistics, the significance of multi-objective optimization becomes evident when confronted with the need to balance multiple conflicting objectives. Unlike situations where decisions are solely based on a single criterion, real-world scenarios often demand the consideration of various factors that may compete or align in different ways. Therefore, the application of multi-objective benchmarks is essential for accurately reflecting the intricacies of these decision environments. Within this project, our incorporation of a diverse set of multi-objective benchmarks empowers future users with a more comprehensive toolset thus ensuring that these tools prove robust and effective in navigating the complexities inherent in practical decision-making contexts. By embracing a variety of benchmarks that better mirror real-world scenarios, this project enhances the reliability and applicability of optimization algorithms, contributing to their efficacy across a spectrum of multifaceted decision environments.

## C. Potential Applications

Considering the fact that Nevergrad is a library renowned for its optimization tools, the inclusion of the multi-objective benchmarks holds immense potential for addressing practical challenges across various domains. Notably, in engineering design, it aids in optimising systems with conflicting objectives, from maximising performance to minimising costs. In financial portfolio optimization, it facilitates the delicate balance between risk and return, contributing to desirable financial outcomes. Similarly, within supply chain management, the benchmarks assist in optimising networks while navigating conflicting goals like cost minimization and on-time delivery. Furthermore, in environmental planning, the project enables the optimization of plans by balancing conflicting objectives, such as minimising ecological impact while maximising resource utilisation. Through these applications, the enhanced toolset provided by Nevergrad stands poised to elevate optimization solutions in diverse and real-world decision-making scenarios. Overall, we can say that the applications of our solution are virtually limitless, given that the optimization challenges involve multifaceted decision-making scenarios and constraints.

## II. Methodology

We created three files, `__init__.py`, `core.py` and `functions.py` under the folder “nevergrad/nevergrad/functions/multiobjective”. `__init__.py` acts as an interface to access various multi-objective optimization functions, and `core.py` provides the foundational structure with the base class for these functions. `functions.py` implements specific multi-objective optimization functions with constraints as subclasses of the base class.

### A. Identify and Select Benchmarks

Aiming at choosing benchmarks that cover a diverse range of characteristics, such as different dimensions, complexities, and features, and which are based on reputable resources, we specifically focus on the Wikipedia page dedicated to multi-objective optimization test functions(Appendix B.), which was also recommended by the nevergrad benchmark issue opener.

### B. Develop Dedicated Classes

The original class `MultiobjectiveFunction` in the `__init__.py` was deprecated and serves as a notice for users to adopt the newer approach of passing a multiobjective loss directly to the optimization process using `optimizer.tell`. We maintained it in our updated `__init__.py` for backward compatibility.

To fill the vacancy of multi-objective benchmarks, we created a new base class with clearer organisation and documentation called `ConstrainedMultiObjective` for defining multi-objective functions with constraints in `core.py`.

We then created the `function.py` to define several multi-objective optimization functions with constraints as subclasses of `ConstrainedMultiObjective` base class. The functions were defined in the `__call__` methods, and constraints were registered using the `register_cheap_constraint` method when applicable. We categorised the 17 functions(`BinhKorn`, `ChankongHaimes`, `PoloniTwoObjective`, `TestFunction4`, `CTP1`, `ConstrEx`, `FonsecaFleming`, `Kursawe`, `ZDT1`, `ZDT2`, `ZDT3`, `ZDT4`, `ZDT6`, `OsyczkaKundu`, `Viennet`, `Schaffer1`, `Schaffer2`) into four type: 2 Objective Functions with 1 variable, 2 Objective Functions with 2 variables, 2 Objective Functions with more than 2 variables, and 3 Objective Functions with 2 variables.

## C. Details of the code files:

### 1) `_init_.py`

It imports specific multi-objective optimization functions and core classes from the functions module, the `ConstrainedMultiObjective` class from the core module, and necessary error handling and typing modules. It will deprecate a class that raises a `NevergradDeprecationError` when instantiated.

### 2) `core.py`

It has two attributes, 'name' which are names of the multi-objective function, and 'parametrization' which indicates the parameter space for the optimization problem. It also has two methods, '`_init__(self, name: str, parametrization: p.Array)`' which initialises the multi-objective function with a name and parametrization, and '`_call__(self, x: np.ndarray) -> np.ndarray`' which is a placeholder method that should be implemented in subclasses to calculate the objective values based on parameter values.

### 3) `functions.py`

We categorised the functions into four type:

#### 3.1) Two Objective Functions with one variable:

- Schaffer1: Schaffer Function N. 1 with two objectives.
- Schaffer2: Schaffer Function N. 2 with two objectives.

#### 3.2) Two Objective Functions with two variables

- BinhKorn: Binh and Korn function with two objectives and two constraints.
- ChankongHaimes: Chankong and Haimes function with two objectives and two constraints.
- PoloniTwoObjective: Poloni's Two Objective function with two objectives.
- TestFunction4: Test Function 4 with two objectives and three constraints.
- CTP1: CTP1 Function with two objectives and two constraints.
- ConstrEx: Constr-Ex Problem with two objectives and two constraints.

#### 3.3) Two Objective Functions with more than two variables

- FonsecaFleming: Fonseca-Fleming function with a configurable number of dimensions.
- Kursawe: Kursawe function with a configurable number of dimensions.
- ZDT1, ZDT2, ZDT3, ZDT4, ZDT6: Zitzler-Deb-Thiele's functions with two objectives and a configurable number of dimensions.
- OsyczkaKundu: Osyczka and Kundu function with two objectives and six constraints.

#### 3.4) Three Objective Functions with two variables

- Viennet: Viennet Function with three objectives.

## D. Algorithm

The primary algorithm used in the existing codebase is the DE (Differential Evolution) optimizer from the Nevergrad library. This optimizer will continue to be employed for the optimization process of the newly integrated multi-objective benchmarks.

## E. Connections with prior work

Our methodology builds upon the existing codebase's structure, where multi-objective optimization problems are formulated using the `ConstrainedMultiObjective` base class. The utilisation of the DE optimizer from Nevergrad aligns with prior work in the codebase. The extension involves integrating additional native multi-objective benchmarks, contributing to a more diverse set of problems for optimization evaluation. This iterative and collaborative approach ensures compatibility with established practices while expanding the scope of optimization challenges addressed by the codebase.

### III. Documentation

As mentioned earlier as an output of our project we created 4 main files such as: `__init__.py`, `core.py`, `functions.py` and `test.py`. To utilise our MultiObjective Benchmark, one must first ensure that Nevergrad is installed. After installing Nevergrad, download the Multiobjective Benchmark files and include them in your project directory following these steps:

1. Clone the nevergrad to local Github (Normally, clone it to your Desktop)
2. Download 4 files (in "Coding Files" folder)
3. Add these 4 files(`__init__.py`, `core.py`, `functions.py` and `test.py`) into this folder:

"nevergrad/nevergrad/functions/multiobjective"

4. Add `example.py` into your Desktop

5. Run this in terminal:

```
export PYTHONPATH=${PYTHONPATH}:"/YOUR PATH/nevergrad"
```

6. Run this to see our constraints are all correct

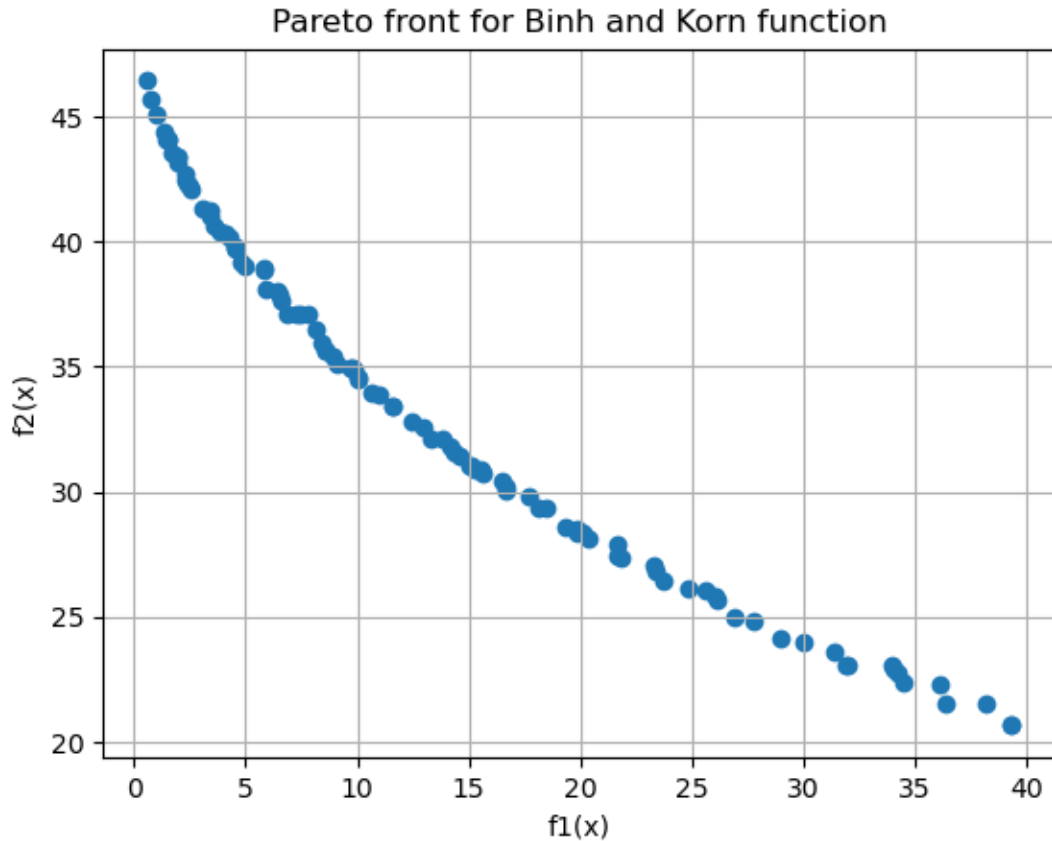
```
python /YOURPATH/nevergrad/functions/multiobjective/test.py
```

7. Run this command to see if code of `functions.py` is normal or not

```
python Desktop/example.py
```

These steps assume that one has Python installed on one's machine and can run commands in the terminal. Replace ``/YOURPATH/nevergrad`` with the actual path to one's Nevergrad repository. `sometimes small adjustments will be needed` accordingly based on the different operating systems.

Our benchmark also provides functionality for visualising the Pareto front of the optimization results, offering the possibility of analysing the trade-offs and relationships between multiple objectives. An example visualisation script is available in the `example.py` file, demonstrating how to plot the Pareto front for a given function. The graphical representation:



To use the multi-objective optimization functions in Nevergrad, other people can import the functions from the functions module in the `nevergrad.functions.multiobjective` package (at the moment we can make it available for public but this is the ideal scenario). For example, if someone wants to use the BinhKorn function, they can import it as follows:

```
from nevergrad.functions.multiobjective.functions import BinhKorn
```

After importing the desired function, users can create an instance of the function and use it for optimization or analysis according to their requirements (similar can be found in the `example.py` file).



## IV. Evaluation

The evaluation of the multi-objective optimization benchmarks implemented in Nevergrad involves rigorous testing and validation. To verify the correctness and consistency of the benchmark functions across different dimensions and configurations, we have developed a test suite (test.py), leveraging existing code in Nevergrad's repository as reference. This allows us to build upon the foundation laid by others while following Nevergrad's coding standards.

The evaluation can be summarised in the following key points:

### A. Correctness of Benchmark Functions

The primary goal of the evaluation is to ensure that each benchmark function produces the expected output for a given multi-objective optimization function. The correctness is validated by comparing the computed results with predefined expected values.

```
# BinhKorn
def test_Binhkorn(self):
    function = functions.BinhKorn()
    x = np.array([0, 0])
    result = function(x)
    expected = [0, 50]
    self.assertTrue(np.allclose(result, expected), f"Expected {expected}, got {result}")
```

Figure: Example Text Function for Binhkorn

As shown in the example above, each test function is initiated by setting the target function as 'function' and initialising the input x. The outcome of the function is stored in the variable 'result'. Following which we manually calculate the expected result based on the actual formula and store it in the variable 'expected'. The assertTrue() function from the "unittest" package is then employed for a direct comparison between the function's output and our hand-calculated result.

Successful completion of all tests is indicated by a "passed" status in the terminal:

```
Running Tests - test_Binhkorn: 6%|██████████| 1/17 [00:00<00:00, 745.79it/s]
Running Tests - test_CTP1: 12%|██████████| 2/17 [00:00<00:00, 955.75it/s]
Running Tests - test_ChankongHaimies: 18%|██████████| 3/17 [00:00<00:00, 1145.15it/s]
Running Tests - test_ConstrEx: 24%|██████████| 4/17 [00:00<00:00, 1258.70it/s]
Running Tests - test_FonsecaFleming: 29%|██████████| 5/17 [00:00<00:00, 1342.01it/s]
Running Tests - test_Kursawe: 35%|██████████| 6/17 [00:00<00:00, 1403.56it/s]
Running Tests - test_OsyczkaKundu: 41%|██████████| 7/17 [00:00<00:00, 1458.38it/s]
Running Tests - test_PoloniTwoObjective: 47%|██████████| 8/17 [00:00<00:00, 1479.02it/s]
Running Tests - test_TestFunction4: 53%|██████████| 9/17 [00:00<00:00, 1522.37it/s]
Running Tests - test_Viennet: 59%|██████████| 10/17 [00:00<00:00, 1559.39it/s]
Running Tests - test_ZDT1: 65%|██████████| 11/17 [00:00<00:00, 1589.63it/s]
Running Tests - test_ZDT2: 71%|██████████| 12/17 [00:00<00:00, 1617.50it/s]
Running Tests - test_ZDT3: 76%|██████████| 13/17 [00:00<00:00, 1640.82it/s]
Running Tests - test_ZDT4: 82%|██████████| 14/17 [00:00<00:00, 1660.97it/s]
Running Tests - test_ZDT6: 88%|██████████| 15/17 [00:00<00:00, 1668.33it/s]
Running Tests - test_schaffer1: 94%|██████████| 16/17 [00:00<00:00, 1695.48it/s]
Running Tests - test_schaffer2: 100%|██████████| 17/17 [00:00<00:00, 1703.78it/s]
.
-----
Ran 17 tests in 0.016s
OK
```

Figure2: test.py output when all tests are passed

When there is one or more than one test failed, detailed information about the failed test(s) is displayed.

```
Running Tests - test_Binhkorn: 6%|██████████| 1/17 [00:00<00:00, 975.65it/s]
Running Tests - test_CTP1: 12%|██████████| 2/17 [00:00<00:00, 1104.93it/s]
Running Tests - test_ChankongHaines: 18%|██████████| 3/17 [00:00<00:00, 1274.86it/s]
Running Tests - test_ConstrEx: 24%|██████████| 4/17 [00:00<00:00, 1370.80it/s]
Running Tests - test_FonsecaFleming: 29%|██████████| 5/17 [00:00<00:00, 1437.59it/s]
Running Tests - test_Kursawe: 35%|██████████| 6/17 [00:00<00:00, 1487.69it/s]
Running Tests - test_OsyczkaKundu: 41%|██████████| 7/17 [00:00<00:00, 1535.41it/s]
Running Tests - test_PoloniTwoObjective: 47%|██████████| 8/17 [00:00<00:00, 1550.93it/s]
Running Tests - test_TestFunction4: 53%|██████████| 9/17 [00:00<00:00, 1588.68it/s]
Running Tests - test_Vienet: 59%|██████████| 10/17 [00:00<00:00, 1618.92it/s]
Running Tests - test_ZDT1: 65%|██████████| 11/17 [00:00<00:00, 1645.00it/s]
Running Tests - test_ZDT2: 71%|██████████| 12/17 [00:00<00:00, 1668.49it/s]
Running Tests - test_ZDT3: 76%|██████████| 13/17 [00:00<00:00, 1687.43it/s]
Running Tests - test_ZDT4: 82%|██████████| 14/17 [00:00<00:00, 1663.09it/s]
Running Tests - test_ZDT6: 88%|██████████| 15/17 [00:00<00:00, 1669.62it/s]
Running Tests - test_schaffer1: 94%|██████████| 16/17 [00:00<00:00, 1695.22it/s]
Running Tests - test_schaffer2: 100%|██████████| 17/17 [00:00<00:00, 1702.72it/s]

=====
FAIL: test_ZDT3 (__main__.MultiObjectiveFunctionTests)
-----
Traceback (most recent call last):
  File "/Users/w_yy_cc/Desktop/project/nevergrad/nevergrad/functions/multiobjective/test.py", line 147, in test_ZDT
    3
    self.assertTrue(np.allclose(result, expected), f"Expected {expected}, got {result}")
AssertionError: False is not true : Expected [0, 3], got [0. 1.]

-----
Ran 17 tests in 0.017s
```

Figure3: test.py output when one test failed.

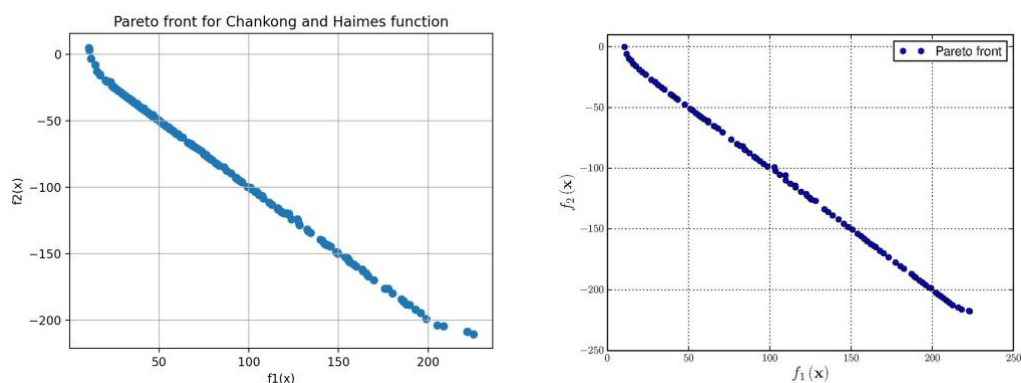


Figure4 Right: pareto front plot generated by our project Left: pareto front sourced from wikipedia

Meanwhile, the pareto front is also used as a testing criterion for our benchmark functions. By visualising and comparing the Pareto front generated by our project with the reference Pareto front sourced from Wikipedia. The alignment between our benchmark functions' Pareto front and the referenced front reinforces the correctness of our implementations (as shown in figure4).

## B. Test Coverage

As mentioned earlier, 4 types and 17 functions were created.

Accordingly, within the *MultiObjectiveFunctionTests* class, 17 test functions are designed to cover a range of scenarios, including single and multi-dimensional cases. This ensures that the benchmark functions are robust and applicable to a diverse range of optimization challenges.

## C. Other Evaluation

Apart from correctness, the evaluation of our contribution also includes *documentation consistency* and *integration with Nevergrad*. The goal is to create a codebase that is not only modular, maintainable, and readable, but also integrates seamlessly into the Nevergrad framework. Specifically, function names, constraints, and input parameters are cross-checked to guarantee coherence between the codebase and the accompanying documentation.

## D. Summary\_Evaluation

In summary, the comprehensive evaluation presented in this section reflects our dedication to a clean code structure, comprehensive documentation, and adherence to project guidelines. The objective is to create an efficient and user-friendly experience for individuals utilising the multi-objective optimization benchmarks within the Nevergrad library.

## V. Summary

Focusing on integrating native multi-objective benchmarks into the Nevergrad library, we addressed the limitations of single-objective benchmarks, we tried to enhance the toolset and diversify and improve multi-objective optimization benchmarks by categorising multi-objective benchmarks based on the wikipedia page, and utilising the Differential Evolution optimizer. The methodology involves creating new files and classes, with documentation for user implementation. Potential applications span engineering, finance, supply chain management, and environmental planning. The evaluation ensured correctness, test coverage, and integration with Nevergrad. Overall, we tried to extend Nevergrad's optimization capabilities for diverse real-world decision-making scenarios.

## VI. Contribution

All group members actively engaged in every stage of the project's development to ensure a comprehensive and unified effort. However, to guarantee meticulous attention to detail, consistency, and the prevention of any repetition, specific responsibilities were assigned to individual team members for final control in accordance with their designated areas of contribution:

### 1. Motivation & Documentation

Led by Anna Parshikova, this stage received dedicated oversight to maintain clarity and coherence in articulating the project's motivation and methodology.

### 2. Methodology

Chia-tien Tang and Shuqi Deng took charge of overseeing the methodology, ensuring a consistent and cohesive approach in the development process.

### 3. Evaluation

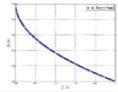
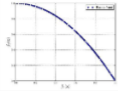
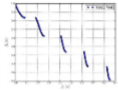
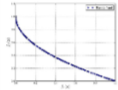
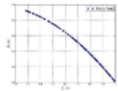
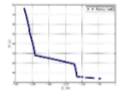
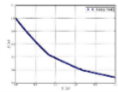
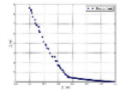
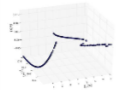
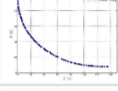
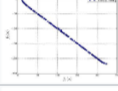
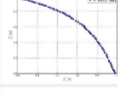
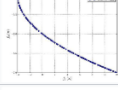
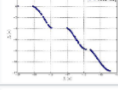
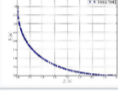
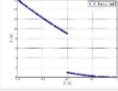
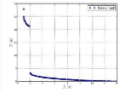
The final control for the evaluation stage was entrusted to Yuchen Wang and Yingxuan Wu, ensuring thoroughness and precision in assessing the outcomes and implications of the project.

## VII. Reference & Appendix

A. CodeFiles:<https://drive.google.com/drive/folders/1UEoQBlOzzzfejl2nXkGbBr6FzOiKXhQg?usp=sharing>

B. Test functions for multi-objective optimization :

[https://en.wikipedia.org/wiki/Test\\_functions\\_for\\_optimization](https://en.wikipedia.org/wiki/Test_functions_for_optimization)

Zitzler-Deb-Thiele's function N. 1: <sup>[29]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 1 + \frac{1}{20} \sum_{i=2}^{30} x_i \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} \end{cases}$	$\begin{cases} 0 \leq x_1 \leq 1, \\ 1 \leq i \leq 30. \end{cases}$
Zitzler-Deb-Thiele's function N. 2: <sup>[29]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 1 + \frac{1}{20} \sum_{i=2}^{30} x_i \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \left( \frac{f_1(\mathbf{x})}{g(\mathbf{x})} \right)^2 \end{cases}$	$\begin{cases} 0 \leq x_1 \leq 1, \\ 1 \leq i \leq 30. \end{cases}$
Zitzler-Deb-Thiele's function N. 3: <sup>[29]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 1 + \frac{1}{20} \sum_{i=2}^{30} x_i \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} - \left( \frac{f_1(\mathbf{x})}{g(\mathbf{x})} \right) \sin(10\pi f_1(\mathbf{x})) \end{cases}$	$\begin{cases} 0 \leq x_1 \leq 1, \\ 1 \leq i \leq 30. \end{cases}$
Zitzler-Deb-Thiele's function N. 4: <sup>[29]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 91 - \sum_{i=2}^{10} (x_i^2 - 10 \cos(4\pi x_i)) \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} \end{cases}$	$\begin{cases} 0 \leq x_1 \leq 1, \\ -5 \leq x_i \leq 5, \\ 2 \leq i \leq 10 \end{cases}$
Zitzler-Deb-Thiele's function N. 6: <sup>[29]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(6\pi x_1) \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 1 + 9 \left[ \frac{\sum_{i=2}^{10} x_i}{9} \right]^{0.35} \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \left( \frac{f_1(\mathbf{x})}{g(\mathbf{x})} \right)^2 \end{cases}$	$\begin{cases} 0 \leq x_1 \leq 1, \\ 1 \leq i \leq 10. \end{cases}$
Oycszka and Kundu function: <sup>[28]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2 - (x_4 - 4)^2 - (x_5 - 1)^2 \\ f_2(\mathbf{x}) = \sum_{i=1}^5 x_i^2 \end{cases}$	$\begin{cases} g_1(\mathbf{x}) = x_1 + x_2 - 2 \geq 0 \\ g_2(\mathbf{x}) = 6 - x_1 - x_2 \geq 0 \\ g_3(\mathbf{x}) = 2 - x_2 + x_1 \geq 0 \\ g_4(\mathbf{x}) = 2 - x_1 + 3x_2 \geq 0 \\ g_5(\mathbf{x}) = 4 - (x_1 - 3)^2 - x_4 \geq 0 \\ g_6(\mathbf{x}) = (x_1 - 3)^2 + x_6 - 4 \geq 0 \end{cases}$ $\begin{cases} 0 \leq x_1, x_2, x_6 \leq 10 \\ 1 \leq x_3, x_5 \leq 5, \\ 0 \leq x_4 \leq 6. \end{cases}$
CTP1 function (2 variables): <sup>[30][34]</sup>		Minimize $\begin{cases} f_1(x, y) = x \\ f_2(x, y) = (1 + y) \exp\left(-\frac{x}{1+y}\right) \end{cases}$	$\begin{cases} s.t. - \begin{cases} g_1(x, y) = \frac{f_1(x, y)}{0.858 \exp(-0.544 f_1(x, y))} \geq 1 \\ g_2(x, y) = \frac{f_2(x, y)}{0.729 \exp(-0.295 f_1(x, y))} \geq 1 \end{cases} \end{cases}$ $0 \leq x, y \leq 1.$
Constr-Ex problem: <sup>[4]</sup>		Minimize $\begin{cases} f_1(x, y) = x \\ f_2(x, y) = \frac{1-y}{x} \end{cases}$	$\begin{cases} s.t. - \begin{cases} g_1(x, y) = y + 9x \geq 6 \\ g_2(x, y) = -y + 9x \geq 1 \end{cases} \end{cases}$ $\begin{cases} 0.1 \leq x \leq 1, \\ 0 \leq y \leq 5 \end{cases}$
Viennet function:		Minimize $\begin{cases} f_1(x, y) = 0.5(x^2 + y^2) + \sin(x^2 - y^2) \\ f_2(x, y) = \frac{(3x - 2y + 4)^2 + (x - y)^2}{27} + 15 \\ f_3(x, y) = \frac{1}{x^2 + y^2 + 1} - 1.1 \exp(-(x^2 + y^2)) \end{cases}$	$-3 \leq x, y \leq 3.$
Binh and Korn function: <sup>[28]</sup>		Minimize $\begin{cases} f_1(x, y) = 4x^2 + 4y^2 \\ f_2(x, y) = (x - 5)^2 + (y - 5)^2 \end{cases}$	$\begin{cases} s.t. - \begin{cases} g_1(x, y) = (x - 5)^2 + y^2 < 25 \\ g_2(x, y) = (x - 8)^2 + (y + 3)^2 \geq 7.7 \end{cases} \end{cases}$ $\begin{cases} 0 \leq x \leq 5, \\ 0 \leq y \leq 3 \end{cases}$
Chankong and Haimes function: <sup>[16]</sup>		Minimize $\begin{cases} f_1(x, y) = 2 + (x - 2)^2 + (y - 1)^2 \\ f_2(x, y) = 9x - (y - 1)^2 \end{cases}$	$\begin{cases} s.t. - \begin{cases} g_1(x, y) = x^2 + y^2 \leq 225 \\ g_2(x, y) = x - 3y + 10 \leq 0 \end{cases} \end{cases}$ $-20 \leq x, y \leq 20$
Fonseca-Fleming function: <sup>[16]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = 1 - \exp\left[-\sum_{i=1}^n \left(x_i - \frac{1}{\sqrt{n}}\right)^2\right] \\ f_2(\mathbf{x}) = 1 - \exp\left[-\sum_{i=1}^n \left(x_i + \frac{1}{\sqrt{n}}\right)^2\right] \end{cases}$	$\begin{cases} -4 \leq x_i \leq 4, \\ 1 \leq i \leq n \end{cases}$
Test function 4: <sup>[36]</sup>		Minimize $\begin{cases} f_1(x, y) = x^2 - y \\ f_2(x, y) = -0.5x - y - 1 \end{cases}$	$\begin{cases} s.t. - \begin{cases} g_1(x, y) = 6.5 - \frac{x}{y} - y \geq 0 \\ g_2(x, y) = 7.5 - 0.5x - y \geq 0 \\ g_3(x, y) = 30 - 5x - y \geq 0 \end{cases} \end{cases}$ $-7 \leq x, y \leq 4$
Kursawe function: <sup>[20]</sup>		Minimize $\begin{cases} f_1(\mathbf{x}) = \sum_{i=1}^n \left[ -10 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{j=1}^n x_j^2} + x_{i-1}^2\right) \right] \\ f_2(\mathbf{x}) = \sum_{i=1}^n \left[  x_i ^{1.8} + 5 \sin(x_i^2) \right] \end{cases}$	$\begin{cases} -5 \leq x_i \leq 5, \\ 1 \leq i \leq 3. \end{cases}$
Schaffer function N. 1: <sup>[31]</sup>		Minimize $\begin{cases} f_1(x) = x^2 \\ f_2(x) = (x - 2)^2 \end{cases}$	$\begin{cases} -A \leq x \leq A \\ \text{Values of } A \text{ from } 10 \\ \text{to } 10^5 \text{ have been} \\ \text{used successfully.} \\ \text{Higher values of } A \\ \text{increase the difficulty} \\ \text{of the problem.} \end{cases}$
Schaffer function N. 2:		Minimize $\begin{cases} f_1(x) = \begin{cases} -x, & \text{if } x \leq 1 \\ x - 2, & \text{if } 1 < x \leq 3 \\ 4 - x, & \text{if } 3 < x \leq 4 \\ x - 4, & \text{if } x > 4 \end{cases} \\ f_2(x) = (x - 5)^2 \end{cases}$	$-5 \leq x \leq 10.$
Poli's two objective function:		Minimize $\begin{cases} f_1(x, y) = [1 + (A_1 - B_1(x, y))^2 + (A_2 - B_2(x, y))^2] \\ f_2(x, y) = (x + 3)^2 + (y + 1)^2 \end{cases}$ where $\begin{cases} A_1 = 0.5 \sin(1) - 2 \cos(1) + \sin(2) - 1.5 \cos(2) \\ A_2 = 1.5 \sin(1) - \cos(1) + 2 \sin(2) - 0.5 \cos(2) \\ B_1(x, y) = 0.5 \sin(x) - 2 \cos(x) + \sin(y) - 1.5 \cos(y) \\ B_2(x, y) = 1.5 \sin(x) - \cos(x) + 2 \sin(y) - 0.5 \cos(y) \end{cases}$	$-\pi \leq x, y \leq \pi$