Universiteit Leiden

Master Computer Science

Reducing Evaluation Time for a Multiobjective Energy System Sizing Problem using Dynamic Simulation Time

Name:	M.A. Voogt
Student ID:	s1542125
Date:	23/11/2023
Specialisation:	Computer Science and Advanced Data Analytics
1st supervisor:	Thomas Bäck
2nd supervisor:	Niki van Stein
External supervisors:	Tobias Rodemann Felix Lanfermann

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

A hybrid energy system integrates multiple energy sources, unlike systems that rely solely on a single source. However, incorporating these diverse energy sources can be intricate due to the variable nature of renewable energy supply. An approach to address these energy fluctuations is by incorporating energy storage, enhancing system resilience but also increasing costs. Another option would be to rely more on conventional energy sources, which is cheap but potentially causes more pollution to the environment. Achieving the optimal balance between cost, resilience, and environmental impact in these energy systems is challenging due to the absence of a universal solution. As such, this optimization can be approached as a Multi Objective Optimization Problem. In this thesis we experiment with evolutionary optimization and simulations to solve such a problem.

When using evolutionary optimization to solve real world problems using a large simulation or computation as an evaluation function, the individual evaluation time can severely limit the number of evaluations and quality of the solutions within a set time limit. Speeding up these simulations is therefore essential but the new simulation still has to represent the original problem in order to generate relevant solutions. In this research we will be looking at a real world optimization problem that uses a simulator as an evaluation function for a hybrid energy system for an office space. Our goal is to try to keep quantitatively equivalent simulation results while reducing the simulated period.

Continuing with the hybrid energy system optimization problem and framework from [10], we explore different options to reduce the simulation time for the evaluation function. To do this we change the simulated period from a full year to a single month to represent the full year. We also change this simulated month dynamically every generation, essentially creating a dynamic fitness function.

The results from these experiments cannot directly match the quality of the baseline results in terms of hypervolume and IGD+ score which means we cannot directly reduce simulation time. However, we do explore an interesting possibility to start an optimization run using the dynamic fitness function and switching later to the original, in order to acquire a better solution set in the same amount of time.

Contents

1 - Introduction

A hybrid energy system is an energy system which integrates multiple sources of energy, opposite to systems that have only a single energy source. Often these multiple energy sources refer to a combination of renewable and conventional sources. Renewable sources are for example wind or photovoltaic energy, while conventional sources are for example gas or coal.

Incorporating these different energy sources can be quite complex since the supply of renewable energy is not constant but fluctuates with nature. For example photovoltaic systems do not provide energy at night. To mitigate these fluctuations it is possible to add a form of energy storage to the system, which increases resilience but also increases costs. The other option would be to rely more on conventional energy sources, which is cheap but potentially causes more pollution to the environment.

Optimizing these energy systems in terms of cost, resilience and environmental impact is difficult since there is no one-size-fits-all solution. This energy system sizing optimization is something that can be formulated as a Multi Objective Optimization Problem. These optimization problems can be solved using simulations and evolutionary optimization, which is what we experiment with in this thesis.

In evolutionary optimization, the evaluation of individual solutions can take up a lot of time, especially in the case of a real world problem consisting of a large simulation or computation. Speeding up these evaluations is essential to finding a better solution within a reasonable time such that the research takes up less computational and/or financial resources. For a simulation, finding a way to simplify or shorten the simulation can be difficult since the simplified simulation still has to represent the original simulation and the real world.

This looks at a simulation for a building energy system consisting of multiple modules that is to be optimized for four different objectives. Specifically we are using and modifying the simulator and optimization problem from [10]. This simulation usually simulates a full year of real time, but for this research the simulation time is changed dynamically to represent different months. The idea is that when shortening the simulation, it is possible to do more evaluations in the same time. Simulating only a single month or a combination of months instead of a full year can theoretically save a lot of time if the results are similar to the original simulation.

A downside of this approach however, is a potential loss of accuracy. Removing a large part of the year from the simulation means removing a large amount of data to base a solution on. This can result in lower quality solutions, since they might be insufficient or outright irrelevant for the part of the year that was not simulated. In this thesis we experiment with various options for decreasing the simulation time, and check how well they represent the full year. Our hypothesis however is that, due to the mentioned downside, the results will be worse compared to the original simulation.

This brings us to our research questions: For the given energy system, is it possible to get a qualitatively equivalent result when simulating a shorter period of time? And if the results are worse, how much worse are they? To answer these questions we have a few subquestions that need answering as well: Which single month is the most representative of the full year simulation? Can we get a similar result if we were to dynamically change the simulated month every generation? And in what order should we change this month?

In this thesis we firstly briefly present an overview about hybrid energy systems and the specific system from [10], which we use for our experiments. Then we will present our experiments and results.

2 - Literature Overview

In this chapter we discuss hybrid energy systems, the need for sizing optimization using optimization algorithms and the energy system optimization problem from Dr. Rodemann which we have taken as the basis for our research.

2.1 Hybrid Energy Systems

Hybrid energy systems are energy systems which have multiple sources of energy to satisfy their load demands. Often these multiple sources refer to a combination of renewable energy sources (e.g., photovoltaic (PV), wind) and conventional energy sources based on carbon-based fuel (e.g., diesel, coal).

A downside of most renewable energy sources is that they do not provide a consistent supply of energy, but are stochastic instead: they fluctuate with nature. For example a PV system only generates energy during the day, with output peaking at around noon. But this results in a mismatch between supply and demand if there is demand for energy during the evening or night. Because of this, there is a need for a separate energy source necessary to solve this issue. For example a connection to the electricity grid, or a generator in the case of remote locations. These sources provide a steady supply of energy but are often more polluting and the fuel requirements for a generator are costly. These two types of energy sources can be used together to mitigate each others' weaknesses, resulting in a hybrid energy system.

To improve the efficiency of these energy systems it is possible to add various modules. For example a way to partially mitigate the previously mentioned supply and demand mismatch with a PV system, is to add a battery. This battery can store surplus energy for times of excess demand and makes the system less dependent on other, potentially more polluting, energy sources [8].

To generate both electricity and heat from fuel, a Combined Heat and Power (CHP, Cogeneration) plant is more efficient compared to a regular generator. The CHP system recovers the waste heat that is required for generating the electricity, and allows it to be used for heating. A way to preserve generated heat would be thermal energy storage (TES, heat storage). The purpose of this is the same as with the battery system for electricity: to store energy when there is more supply than demand to improve efficiency [11].

Combining these energy systems can result in a hybrid energy system to allow for the needs of the user. The use cases range from reducing grid dependence to save money, to complete off grid living in remote areas. When building these solutions, the combination of components and size of the system can become quite complex. In the next section we discuss techniques for the so called sizing of energy systems.

2.2 Hybrid Energy System Sizing Optimization

The additional components mentioned above are essential for the efficiency and self-sufficiency of the energy system, but also add complexity to designing a hybrid energy system. There is no "one-size-fits-all" solution since the environment that provides natural sources of energy differs per location. Combined with different load requirements, there is a need for custom energy system solutions. There are different methods for the so called sizing of energy systems [7] [14], but for this thesis we will focus on the usage of AI related methods. Specifically the usage of optimization algorithms to find solutions, and simulation tools to verify them.

Combining all variables from the environment and energy consumption with the load demands and budgetary constraints allows us to formulate the sizing of an energy system as a multi-objective optimization problem. Examples of possible objectives are load satisfaction, reliance on grid, investment costs and operation costs over time. These objectives can be calculated from formulas which take into account data from the local natural energy supply and energy demands, and variable parameters which represent the components of the energy system (e.g., PV size, battery size). The combination of formulas is the model for a specific energy system and has to be optimized for its objectives. To perform this optimization it is possible to use optimization algorithms. The model becomes the evaluation function with parameters and objectives [12] [15].

An alternative for the usage of a simulator is replacing it with a surrogate model. This model is trained using the input and output of the real simulation to mimic it, such that it can predict the performance of the candidate solutions. Running the surrogate instead of the actual evaluation function is significantly faster. A downside is the fact that it is only a prediction and therefore less accurate compared to the original evaluation function. Another downside is the training of the surrogate for which you will need actual data, and there is risk of overfitting which reduces accuracy.

A surrogate for the simulator for this research (from [10]) has recently become available [1], but at the time that this research was executed it was not fully developed yet, hence we were not able to make use of this surrogate.

2.3 Hybrid Energy System Optimization Problem

This research project is based on the energy system optimization research of Dr. Rodemann [10]. In his research he tests various optimization algorithms on a real world energy system optimization problem in terms of performance and practical relevance. For this research we experiment with this optimization problem and simulator framework to change the simulated period for faster function evaluations. Here we discuss his original optimization framework and in the next chapter we outline the changes we made to it for this research.

The goal of the optimization problem is as follows: find the best configurations in the form of a Pareto front of the energy system for a company office if an investment was made to upgrade the current system configuration.

2.3.1 Optimization Problem

The energy system configuration consists of different modules, e.g. heat storage, PV system and battery which interact with each other and receive external weather and energy consumption data inside a simulator. The simulator is explained further in the next section. For this energy system there are ten parameters which influence these modules.

The parameters are as follows:

1. PV inclination angle [0–45°]
2. PV orientation angle [120–250°]
3. PV peak power [0–450 kW]
4. Battery capacity [5–400 kWh]
5. Minimum battery State of Charge [0–40%]
6. Maximum battery State of Charge [50–95%]
7. Battery charge threshold [-500–0 kW]
8. Battery discharge threshold [300–700 kW]
9. Heat storage volume [1–5 m^3]
10. Upper CHP operation threshold [10–25°]

This company office in this case is located in Germany, and is not as isolated as some of the locations mentioned in the research papers cited in the previous section. The office is connected to the electricity grid as primary power source and does not require diesel generators to satisfy load demand. The construction of the PV system and other modules are considered purely for saving on yearly energy costs and reducing CO_2 emissions instead of necessity.

The aforementioned parameters have to be optimized for the five objectives of this optimization problem:

- Investment Cost [Euro]: An addition of the costs for the PV system, Battery, and Heat storage, based on their sizes set by the parameters.
- Yearly Costs [Euro]: An addition of the yearly costs for additional electricity needed from the grid, gas consumption, CHP maintenance and peak electricity fee from the grid, minus the reimbursements for feeding electricity back to the grid and government subsidy for the CHP.

- Yearly CO_2 [t]: The total amount of CO_2 emissions from the gas consumption and the electricity from the grid.
- Resilience [s]: The amount of time the company could operate if there suddenly was no energy from the grid available at the worst possible time (which is a worst case scenario).
- Battery Lifetime [Years]: The system uses a Li-Ion Battery which degrades over time. The lifetime is the number of years it takes before the State-Of-Health falls to 90% of original capacity. The simulation computes the State-Of-Health from which this Lifetime is derived.

Investment cost, yearly cost and yearly CO_2 are objectives that need to be minimized while resilience and battery lifetime are to be maximized.

Since these different objectives each have their own scale, the original research makes use of desirability functions to visualize the results. These functions take the raw values of the objectives and convert them into numbers between 0.0 and 1.0 to signify how desirable they are, with 0.0 as most desirable and 1.0 as least desirable.

2.3.2 Energy System Simulator

The actual simulation model is made using a custom library in SimulationX [4] simulation software using the Modelica [3] simulation language. To simulate a full year of energy system usage the model uses two datasets: A data set containing weather data for a certain location, and a dataset with energy consumption for an office building. Both of these datasets contain real world data which were measured for a full year. For the experiments the weather data set corresponding to the weather in Offenbach was chosen, but it can be swapped out for weather data of a different location. Like the weather data, the energy consumption data can also be swapped out to represent a different situation.

To fit in the framework this model is exported as a standalone executable which takes the parameters as input and produces data from which the final objectives are calculated. From these raw objectives the desirabilities are calculated and these are returned to an optimization algorithm which can then calculate a new set of parameters. This is how the simulation becomes the evaluation function for the optimization problem and this loop is visualized in Figure 2.1.

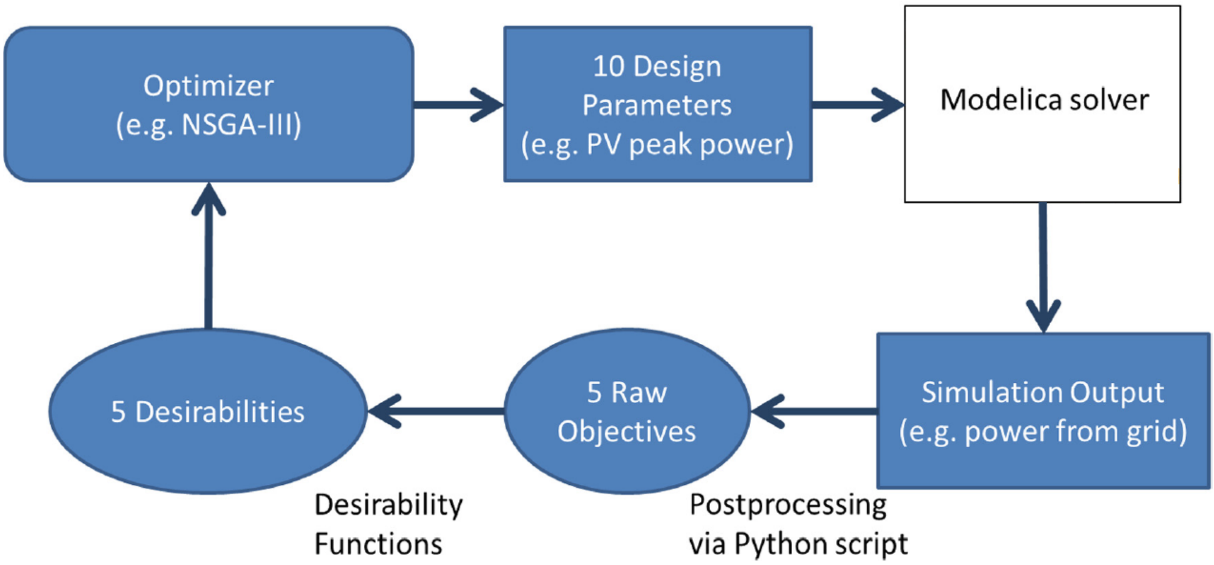


Figure 2.1: Optimization loop from [10]

3 - Research Setup

This research project is based on the earlier energy system optimization research of Dr. Rodemann [10] which we discussed in the previous chapter. This chapter discusses the changes made to his energy system simulator and framework, which is used as a starting point, and how the results are evaluated.

3.1 Energy System Optimization Problem – Changes

The goal of the optimization problem is the same: find the best configurations of the energy system for a company office if an investment was made to upgrade the current situation, in the form of a Pareto front.

The ten parameters remain the same as described in section 2.3, but we removed the Battery Lifetime objective to simplify the problem.

It is expected that the objective for investment cost scales linearly with all other objectives. For example an energy system configuration with lower yearly costs will require a larger investment to build and the same goes for a version with a large resilience or low CO_2 emissions. This is because larger PV systems or larger heat storage are more expensive to build. The relationship between the resilience and the yearly CO_2 emissions should also be linear since a larger battery improves the self-consumption of solar energy since it can store more excess energy when demand is low. This energy can then be used later to reduce grid energy usage and thus reducing CO_2.

Another feature that is removed to simplify the problem is the conversion from raw objectives to desirabilities. In [10] the optimization problem outputs the desirability of an objective to the optimization algorithm instead of the raw objective value. The conversion from raw value to desirability is done using a formula containing certain control point constants that are set differently for each objective. For this conversion to properly function these variables have to be tuned manually for each objective which should be done by the decision maker in consultation with an expert. The benefit of having desirabilities instead of raw objective values is that it makes it easier to see how good a solution is compared to others for people who are not experts.

We removed this conversion to desirabilities since we do not have this expert to help tune the functions. In addition, the desirabilities normally scale from 0.0 (worst) to 1.0 (best). However, since we are minimizing, our desirabilities run from 1.0 (worst) to 0.0 (best), which is the opposite of how they are intended to be understood. Since this is confusing it is another reason to remove this feature.

Figure 3.1 contains a schematic overview of the new framework optimization loop which reflects this change. If we compare this figure with the original loop in figure 2.1 we can see the simplification of the framework with the removal of the desirabilities.

3.2 Framework Software and Optimizer

The original optimization problem is integrated with the PlatEMO V1.5 [13] framework. We upgraded the Matlab function to be compatible with PlatEMO V2.5. PlatEMO is a Matlab based evolutionary multi-objective optimization platform containing many optimization algorithms and benchmark problems. It also allows for users to create their own optimization problems and run these with the available algorithms, which is what we are using it for in our case.

The selection of which optimizers to test in [10] was arbitrary to a certain degree, based on a broad range of optimization approaches. From these tested optimizers we chose IBEA [16] since it was the best performing optimizer from Dr. Rodemann's research. The IBEA algorithm employs quality indicators to guide the selection

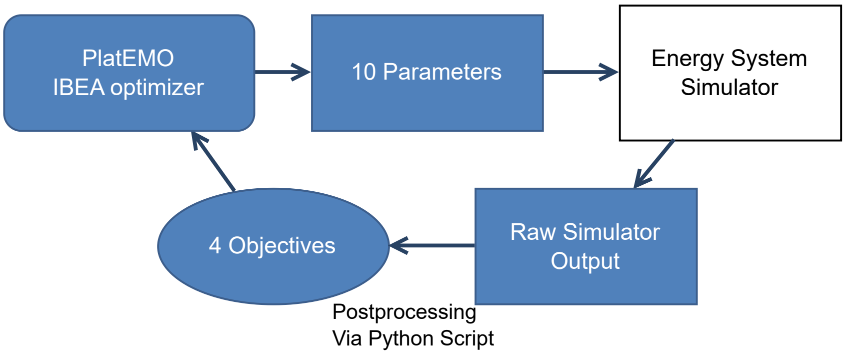


Figure 3.1: New Optimization Loop Overview without desirabilities

and ranking of individuals for mating and survival. It is not our goal to benchmark algorithms, but to evaluate the performance differences with different simulated periods.

3.3 Problem Statement

Running the simulation with a full year of simulated time takes roughly five minutes. This means that when using for example a population size of 100, the evaluation time of a single generation will take between eight an nine hours. To calculate the time it takes for a single optimization run we will have to multiply this number by the number of generations for the run. If we take for example 100 generations, we will have a total simulation time of over a month. For statistical purposes this experiment has to be repeated as well. From this we can conclude that it can take many months to gather results using this setup.

Simulating a single month instead of a full year will take significantly less time, varying between 1,5 to 2 minutes for a single evaluation. However the results of this might not be representative of a full year, depending on the chosen month. This is because the weather and energy consumption differs greatly between months. For example in winter the PV system produces less electricity since there is a shorter period of sunlight, and in summer there is relatively more energy consumption due to the use of air-conditioning. These conditions are present in the datasets and have a large effect on the modules in the simulator and how they interact with each other and the objectives. If we look at the example of the PV system in winter again, we might see that the battery is not filled completely due to less energy availability, which results in a lower resilience.

The interaction between modules is also what limits us in how short the simulated period can be, while retaining sensible results. For example the amount of electricity that is stored in the batteries can have an influence over a longer period of time since it carries over to the next day or week. To properly see the influence of this, we will need to simulate a longer period.

3.4 Cluster Optimization

Before starting the experiments we made an optimization to the framework to speed up the time it takes to evaluate a single generation. The evaluation function was upgraded to work in parallel on a computer cluster such that all individuals of a single generation can be evaluated in parallel instead of sequentially. This makes the total evaluation time for a generation equal to the slowest evaluation of all individuals in the generation. For our example from last section it essentially removes the multiplication by 100 for the individuals, putting us back at eight hours for a single run with 100 generations.

The parallel evaluation is set up in such a way that eight individuals are evaluated on a single compute node, with the only hard limit after that being the size of the computer cluster.
To achieve this, the parameters are split and labeled for each individual evaluation for the new generation, after they are generated by the algorithm. All raw output files and generated objectives are labeled to match the parameters. This way the correct objectives are read for the given parameters.

Current Implementation Downsides

The current implementation of the parallel evaluation has a few downsides which can hinder the performance of experiments with this framework.

To begin, the current implementation is limited in the compatibility with optimization algorithms. It only works with algorithms where the size of the new generation is constant due to the way it distributes the function evaluations over the computer cluster. This could be fixed very easily by turning the parameter for population size into a variable outside of the optimization algorithm to modify the node allocation of the computer cluster. It should however be considered that using an algorithm that evaluates only one new solution in a new generation would only benefit from this parallel implementation for the first full generation.

A bigger downside is the dependence on the availability of the nodes in the computer cluster. If the cluster is shared within, for example, the company or the university, another person can claim the nodes while they are free between the generations. This means that the job for the next generation will be held until the nodes are available again. If you are running a timed experiment, this results in having less generations evaluated within the time limit, which impacts the final result. To fix this problem a more complex jobscript would have to implemented. This script should reserve all required nodes for the full duration of the experiment and communicate with the optimization algorithm to receive new solutions and send back their solutions.

3.5 Evaluation

To evaluate the produced results two main performance metrics are used: Hypervolume (HV) and the Inverted Generational Distance+ (IGD+) score.

3.5.1 Hypervolume

The hypervolume score of a solution set measures the volume of the dominated objective space relative to a reference point. Generally this reference point is placed such that it is slightly worse than the nadir point (the point with the worst possible objective values in the Pareto front) to make sure that all Pareto optimal solutions have a positive hypervolume contribution. Since this reference point is set by the user, there is no auxiliary information needed about the optimization problem.
The formula for hypervolume is defined as

$$HV(f^{ref}, S) = \Lambda \left(\bigcup_{S_n \in S} [f_1(S_n), f_1^{ref}] \times ... \times [f_m(S_n), f_m^{ref}] \right)$$

where f^{ref} is the reference point, S is the solution set and Λ refers to Lebesgue measure.

Other performance metrics can, for example, require a reference solution set. A reference set contains the solutions that form the known Pareto front of the evaluation function. Since this is not needed to calculate hypervolume, it can also be used for problems that have no reference set available. For our optimization problem we do not have a reference solution set available which makes hypervolume an effective manner to grade our generated solutions.

3.5.2 IGD+

As an additional metric to compare our results we are using Inverted Generational Distance+ [5]. IGD+ is a performance metric that calculates the distance between a reference solution set and a solution set. This distance should be minimized to approach the reference solutions. IGD+ is the modified version of Inverted Generational Distance (IGD) such that it is fully Pareto compliant (which IGD is not). This means that when using IGD it is possible that some dominated solutions can produce a better or equal score when compared to the reference set, while that is not supposed to happen since they are dominated.
The formula for IGD+ is defined as

$$IGD^+(R, S) = \left(\frac{1}{|S|} \sum_{s \in S} \min_{r \in R} d^+(r, s)^p \right)^{\frac{1}{p}} \quad \text{where} \quad d^+(r, s) = \sqrt{\sum_{k=1}^{M} (max\{r_k - s_k, 0\})^2}$$

with reference set R and solution set S where all M objectives must be maximized.

The way IGD+ and IGD calculate this distance is inverted compared to the regular Generational Distance (GD) calculation. This means that whereas GD calculates the distance from points in the solution set to the reference set, IGD calculates the distance between points in the reference set to the closest point in the solution set. This inversion makes the calculation of the distance more accurate compared to the original. For our optimization problem we do not have a reference solution set available. To compute the IGD+ we created a reference set by gathering all non dominated solutions from all runs from all our experiments.

4 - Experiments & Results

4.1 Overview

For this research we devised 10 experiments with different simulation times. The main overview is provided in table 4.1. We first establish a baseline result using the full year simulation. Then we find the month which is the best representation of the full simulation. After that we apply different strategies to dynamically change the simulated month, creating a dynamic fitness function.

Experiment Name	Description
Full Year	Each individual is evaluated with a full year of simulated time for all generations. This is the baseline experiment.
Most Similar Month	Each individual is evaluated with the month that is closest to the baseline experiment in terms of Hypervolume as simulated time for all generations. The most similar month that was found is August.
Single Cycle	All individuals are evaluated with a different month in simulated time every generation, in the order of the year. This cycle repeats every 12 generations. *Example:* Gen1=January, Gen2=Feb, Gen3=Mar ... Gen12=Dec, Gen13=Jan, etc.
Double Cycle	All individuals are evaluated with a different month in simulated time every 2 generations, in the order of the year. This cycle repeats every 24 generations. *Example:* Gen1=Jan, Gen2=Jan, Gen3=Feb, Gen4=Feb ... Gen24=Dec, Gen25=Jan, etc.
Triple Cycle	All individuals are evaluated with a different month in simulated time every 3 generations, in the order of the year. This cycle repeats every 36 generations. *Example:* Gen1=Jan, Gen2=Jan, Gen3=Jan, Gen4=Feb ... Gen36=Dec, Gen37=Jan, etc.
Single Random	All individuals are evaluated with a different random month in simulated time every generation. *Example:* Gen1=Apr, Gen2=Nov, Gen3=Oct, etc.
Double Random	All individuals are evaluated with a different random month in simulated time every 2 generations. This means the randomly chosen month repeats once before selecting a new one. *Example:* Gen1=Apr, Gen2=Apr, Gen3=Nov, Gen4=Nov, Gen5=Oct, etc.
Triple Random	All individuals are evaluated with a different random month in simulated time every 3 generations. This means the randomly chosen month repeats twice before selecting a new one. *Example:* Gen1=Apr, Gen2=Apr, Gen3=Apr, Gen4=Nov, Gen5=Nov, etc.
5x Random	The same as above, but this time the randomly chosen month repeats 5 times.
10x Random	The same as above, but this time the randomly chosen month repeats 10 times.

Table 4.1: Overview of all experiments

All experiments have a 4 hour time limit and 200 individuals per generation. With 8 evaluations in parallel per cluster node, we used 25 nodes. For statistical purposes we repeated all experiments 10 times. After running the experiments, we evaluate every 50th generation of each run with a Full Year evaluation to be able to fairly compare how well the generated solutions perform in the full simulation. We also do this every 5 generations for the first 70 generations for the Random Month experiments to be able to more accurately compare the increase in hypervolume in the first hour.

4.2 Full Year Baseline

This is our baseline experiment in which the simulated time is set to to a full year for each individual.

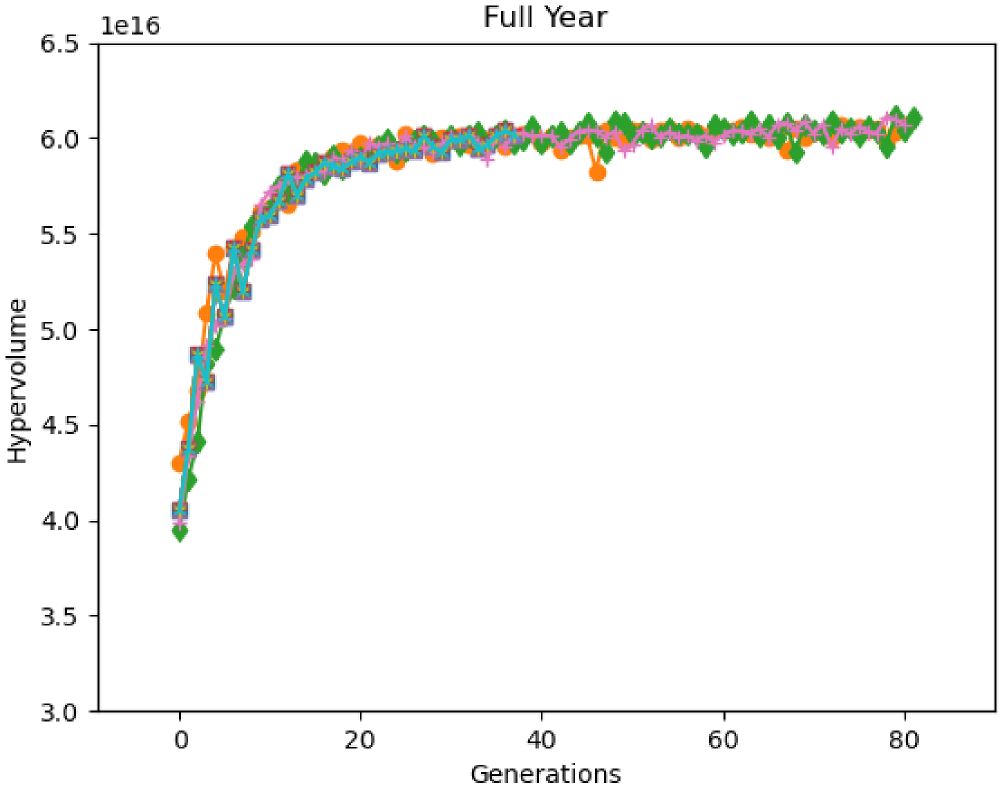
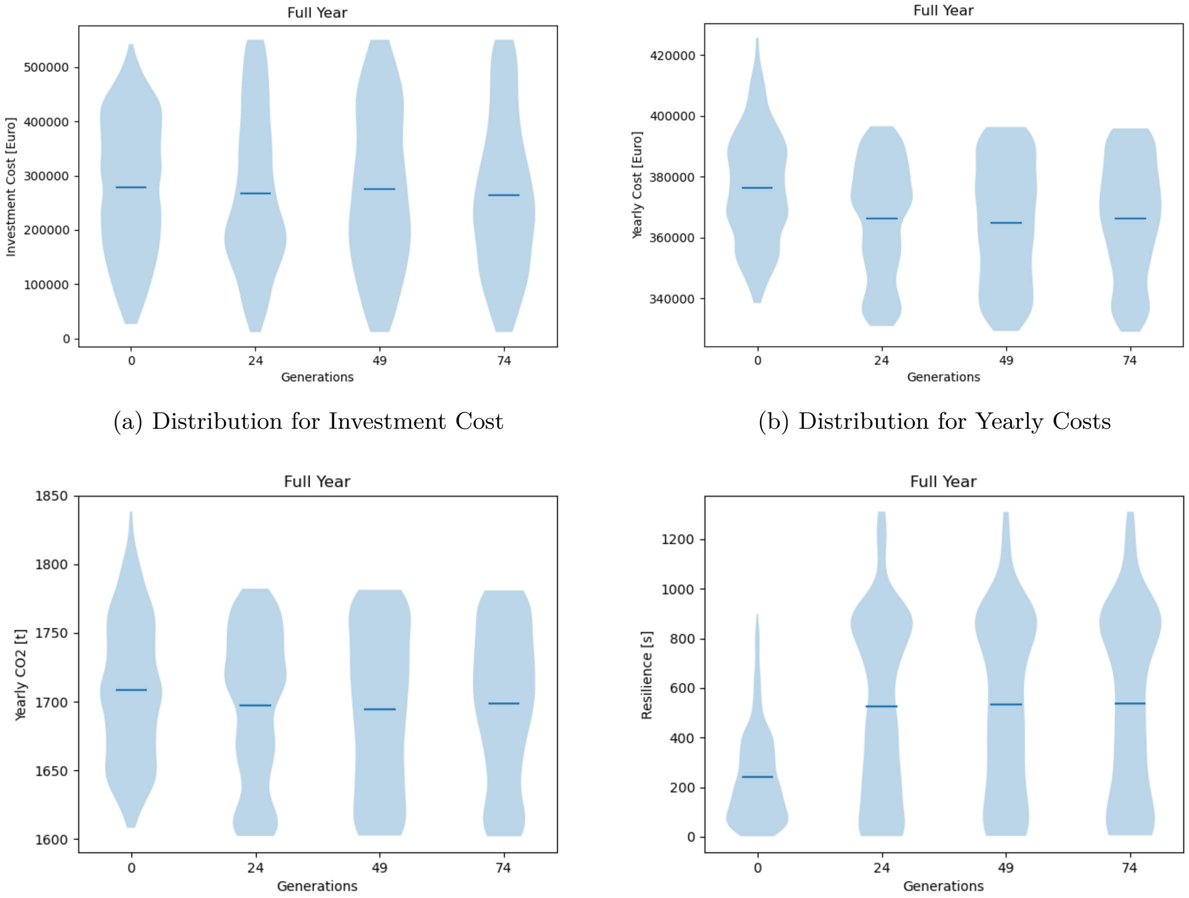


Figure 4.1: Baseline Hypervolume. Each of the ten runs is marked with a separate symbol color

Figure 4.1 contains the hypervolume graph for our baseline experiment. In this graph we observe that within the first 25 generations the hypervolume increases quickly, but after that it begins to plateau just above 6×10^{16}. We will keep this number in mind when comparing the other experiments to this baseline. From now on it is represented as a black dotted line in all other hypervolume graphs.

Another aspect we must take note of here is that, at most, 82 generations can be evaluated in 4 hours with the full year simulation.



(a) Distribution for Investment Cost

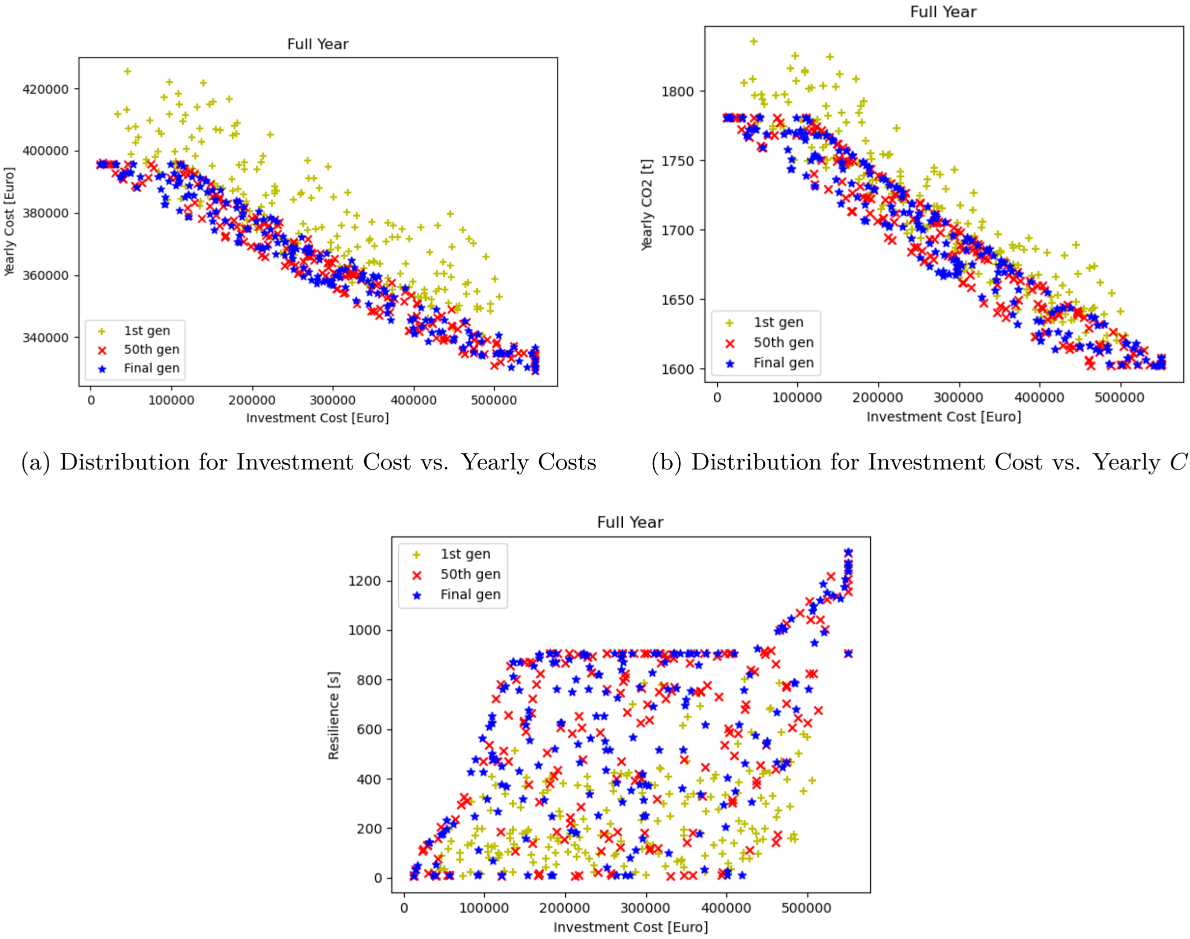
(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure 4.2: Baseline Objective Distributions

In Figure 4.2 the distributions for the different objectives can be seen. In 4.2a and 4.2d the objectives appear to be spreading out more compared to the other figures which have stricter ranges of the distributions for the objectives. For example for the yearly CO_2 it never goes above ~1780t or under ~1600t. Another interesting thing to notice is the very specific shape of the distribution of the resilience objective with very little solutions with more than 1050s resilience.
Another interesting thing to notice in this figure are the ranges of the objective values. The variation in the yearly CO_2 is quite low, compared to investment costs. This means that there might be various ways do achieve a certain amount of yearly CO_2 with different budgets.



(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure 4.3: Baseline Objective Space

In Figure 4.3a and 4.3b, the investment cost scales linearly with yearly costs and yearly CO_2, which is as expected. However the Pareto front for resilience and investment cost in Figure 4.3c has a more distinct rhombus-like form, with only linear scaling in the low budget range and high budget range. A lot of the solutions with the same resilience at around 900s, have very different investment costs, forming a horizontal edge in the Pareto front. This means that there are probably parameters that do not influence resilience, but do increase the investment cost, potentially to improve the other two objectives.

Figure 4.4 shows the parallel coordinate plot of all Pareto optimal solutions of the baseline experiment. The solutions are colored based on their investment cost, with cheaper solutions in blue and expensive solutions in orange. This confirms what we saw earlier in figure 4.3. The investment cost scales inversely with yearly cost and yearly CO_2. We can also see that the cheapest solutions in dark blue will have the least resilience. Only the most expensive solutions can have the most resilience, but they are not concentrated at the maximum and are instead fanning out.

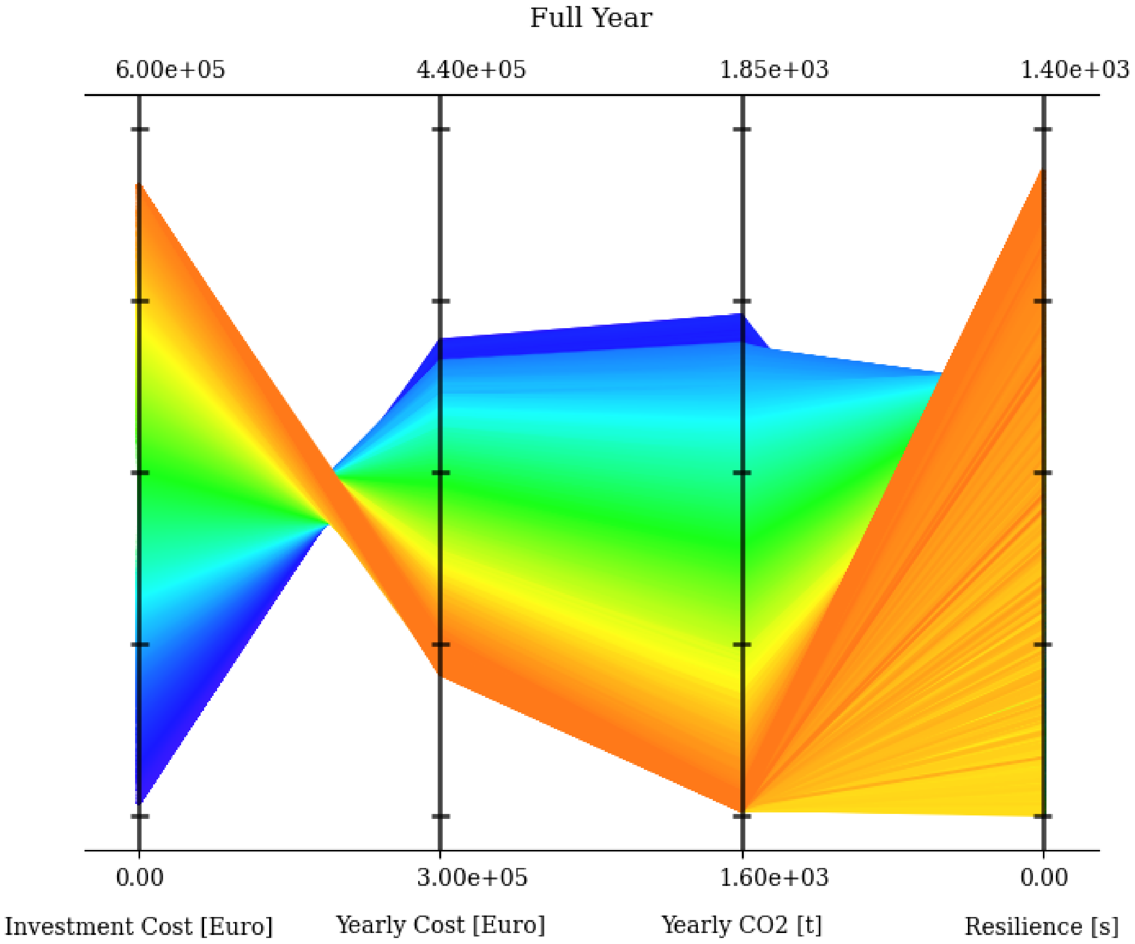


Figure 4.4: Baseline Pareto optimal Parallel Coordinates Plot. The hue of the color scales with Investment Cost. Cheap solutions are blue, expensive solutions are orange.

Parameter Convergence and Dimensionality Reduction

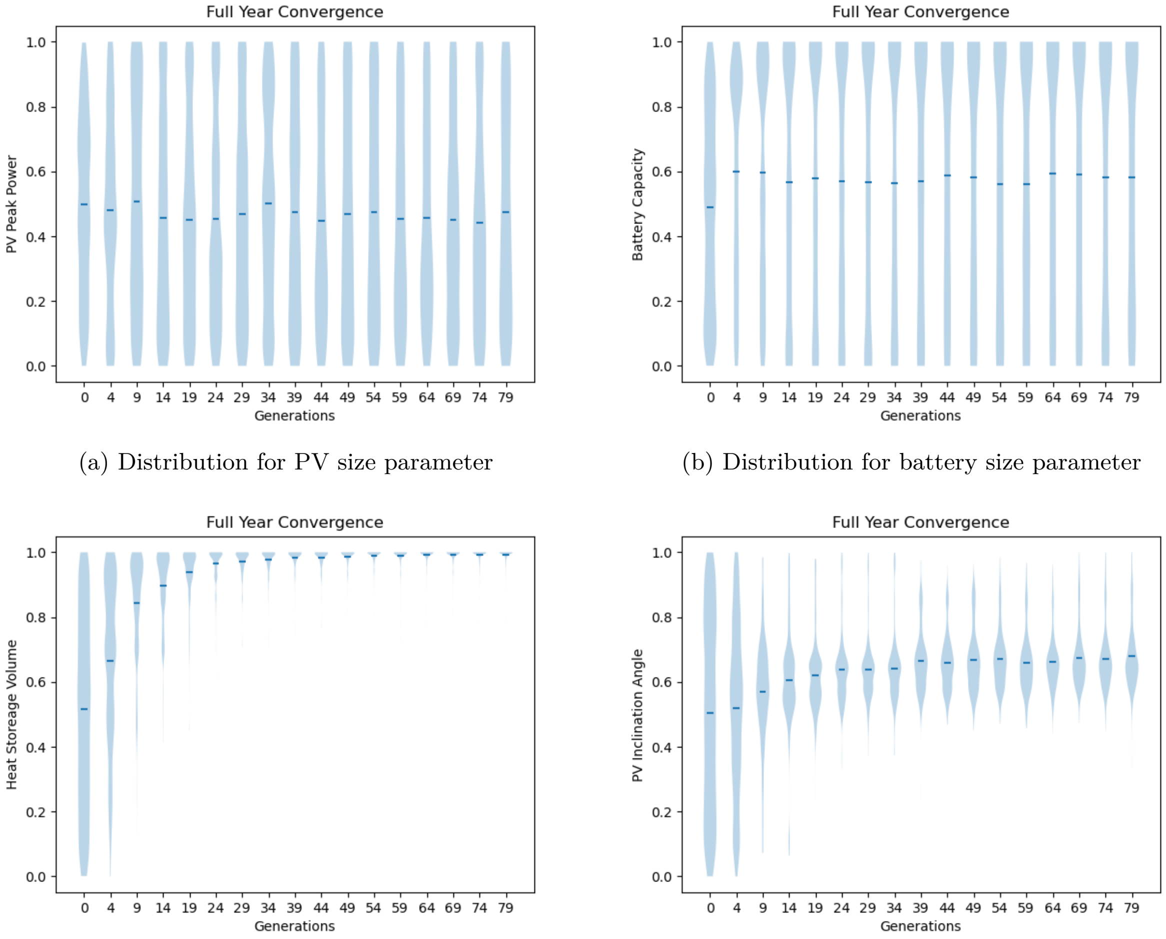
Figure 4.5 contains violinplots that indicate the baseline spread of the three most important parameters. These are the most important since they have a direct influence on most objectives. Here it is clearly visible that the size of the heat storage converges quickly to maximum, while the battery size and especially the PV size stay spread out much more. This can be explained by the fact that expanding the PV systems and batteries will also increase costs and the algorithm keeps cheaper solutions in the population.

The convergence of a parameter usually means that an optimum is found. With this optimum one can perform a dimensionality reduction on the optimization problem by removing the parameter and replacing it with a variable with the optimal value. This reduces the search space for the optimization algorithm which can reduce the amount of time before optimal solutions are found.

When looking at a potential dimensionality reduction for our specific problem, we must note first that this will not influence the real world time it takes to simulate a solution. The potential benefit could be in finding an optimal solution in less generations due to the decreased size of the search space.

Considering the fact that a dimensionality reduction has no influence on our simulation, we could say that this is outside the scope of this thesis, however we will take a look at our parameters to check for options. We have two types of converging parameters: Firstly the parameters that converge at their maximum or minimum value. The heat storage size here is an example. It converges at the maximum value which represents a volume of 5m^3. The fact that it converges here could mean the optimal size of the heat storage is actually larger than 5m^3, but we cannot say for sure. This could mean that the parameter range is insufficient, however there can be constraints from outside which govern the limits of these parameters.

Secondly we have parameters that do converge within their ranges. Examples here are the parameters for the PV angles. These could be better candidates for a dimensionality reduction, however this would reduce the



(a) Distribution for PV size parameter

(b) Distribution for battery size parameter

(c) Convergence for heat storage size parameter

(d) Convergence for PV Inclination angle parameter

Figure 4.5: Baseline parameter convergence. The parameter values are normalized for these plots.

versatility of the optimization problem. As described in section 2.3 the weather and power consumption data for the simulator can be swapped out for different locations. Different parts of the world have a different optimal values for PV angles, therefore setting those to the optimal values for Germany would not work for other office locations. In short, this would enable a dimensionality reduction for this specific location, but it would not apply to others. Also note that we only know the optimal values for these parameters because we do have them as parameters.

4.3 Most Similar Month

To answer one of our secondary research questions, about which month of the year is the most similar to our baseline result, we change the simulated time to a single month of the year and perform two runs. We repeat this for each month of the year and compare the results. We choose the month with the highest hypervolume result to do the full ten runs with. The hypervolume results for every month are available in Appendix A and the IGD+ scores in Appendix B. From this we have chosen August as the most similar month. A larger copy of its HV graph is available in Figure 4.6. Our second best possibility is April, but we have chosen August since it can simulate more generations within the 4 hour time limit. These differences in the number of generations for the individual months can be explained by the fact that some months take longer to simulate due to calculations that are made within the simulation. The complex simulation of the CHP module is one of the reasons for the longer simulation. The CHP is continuously off in summer and continuously on in winter. In spring and autumn it is partially turned on and off and this switching takes time to calculate. A possible conclusion could be that the CHP switching is probably not used during August.

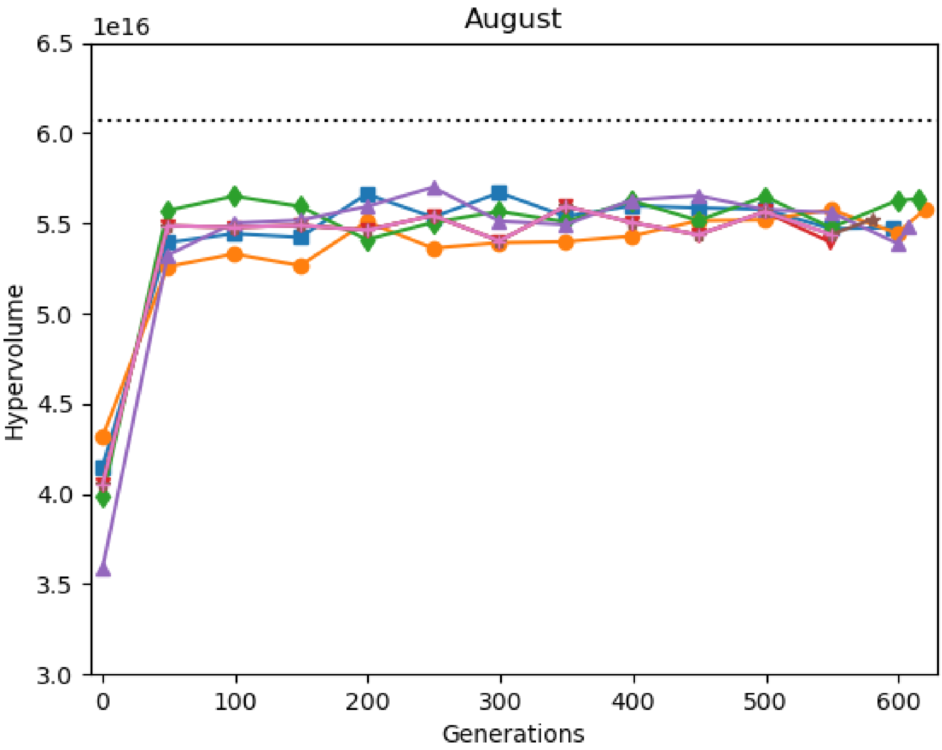
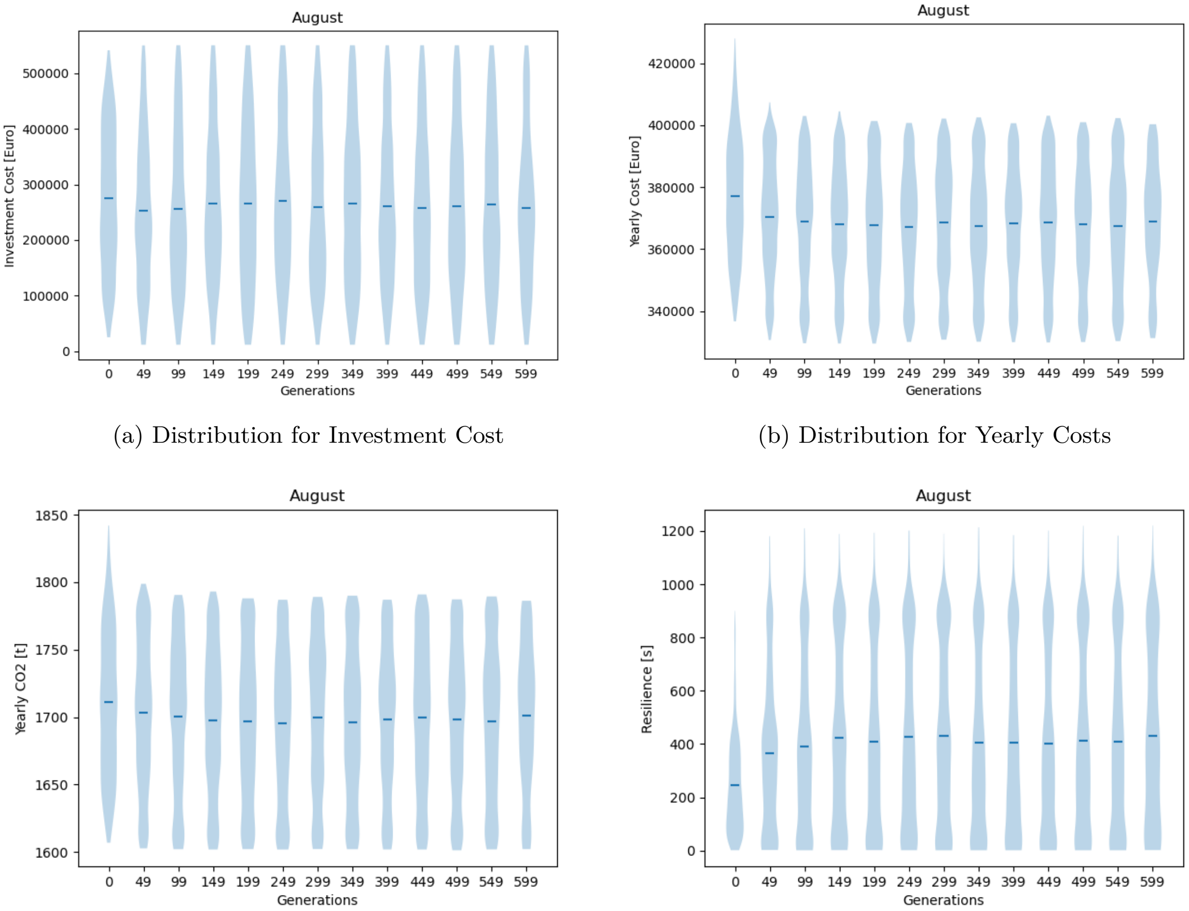


Figure 4.6: August Hypervolume over generations. The dashed line represents the maximum HV reached by the baseline experiment. Each run is represented by a different color line.

In Figure 4.7 the distributions for August are plotted. When comparing them to the baseline there are some noticeable differences and similarities: In Figure 4.7d there is the distinct shape which is much like the one for the baseline result, but it takes more generations to converge into this shape. In the baseline results it is found after 24 generations, here it is 149 generations which means it takes longer to converge to this shape. We also observe that in the last few generations the shape becomes less distinct and more evenly distributed. This means that the algorithm is moving away from the optimal distribution, which is expected since August is not fully representative of the full year and therefore has a different optimum.



(a) Distribution for Investment Cost

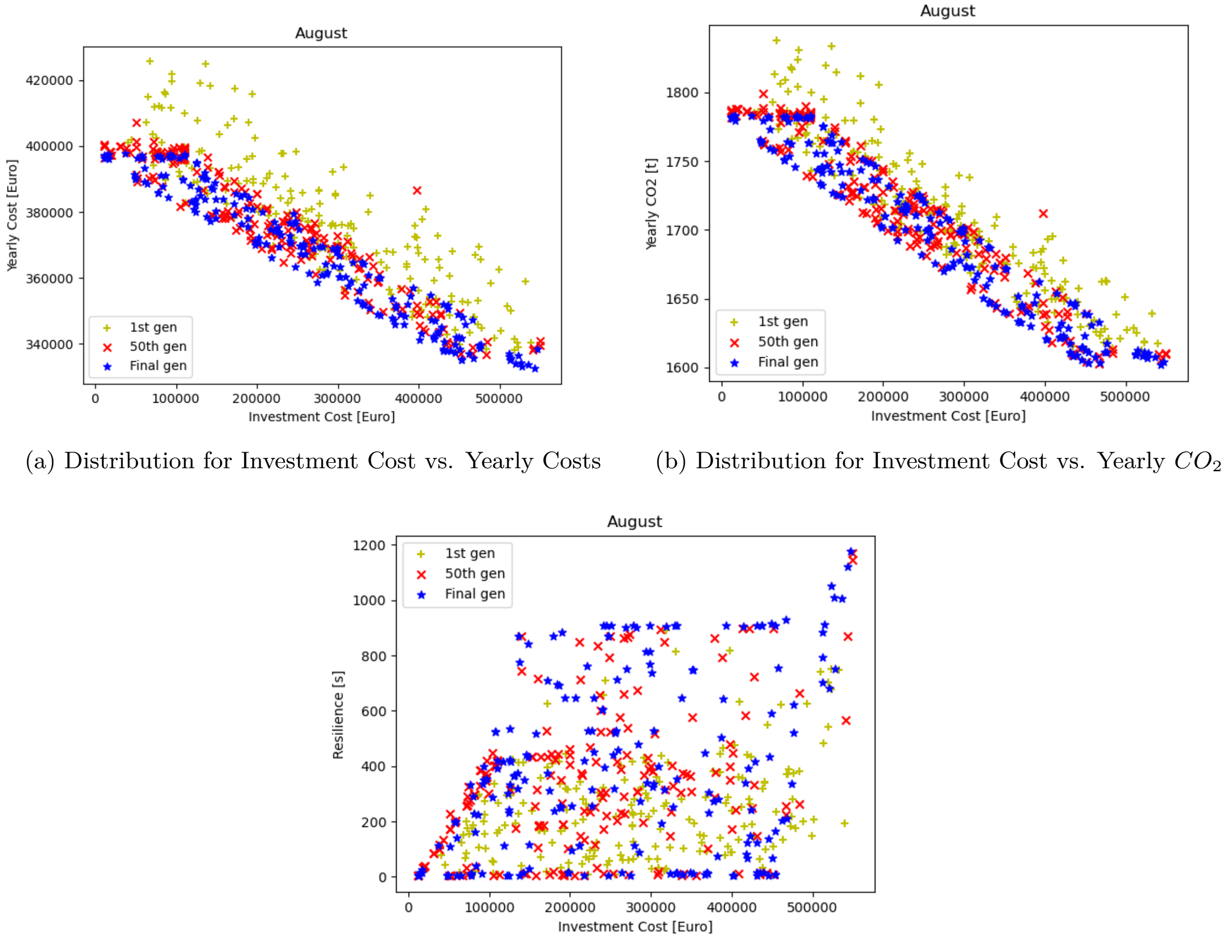
(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure 4.7: August Objective Distributions

Figure 4.8 contains the objective space for August. While 4.8a and 4.8b are very similar to the versions for the baseline experiment, 4.8c is more interesting. Here are significantly less solutions with a resilience of over 400s, while finding a lot of solutions that are close to 0s. This horizontal bottom edge in Figure 4.8c is much more visible here compared to the baseline experiment. Most solutions of the 50th generation are in the area between 400 and 0s resilience, while the final generation is much more spread out. This is in contrast to the baseline experiment where the difference between the 50th and final generation is much smaller. From this and the distribution results it is clear that with less simulation time we need more generations to improve our individual solutions which confirms our choice for August over April.



(a) Distribution for Investment Cost vs. Yearly Costs

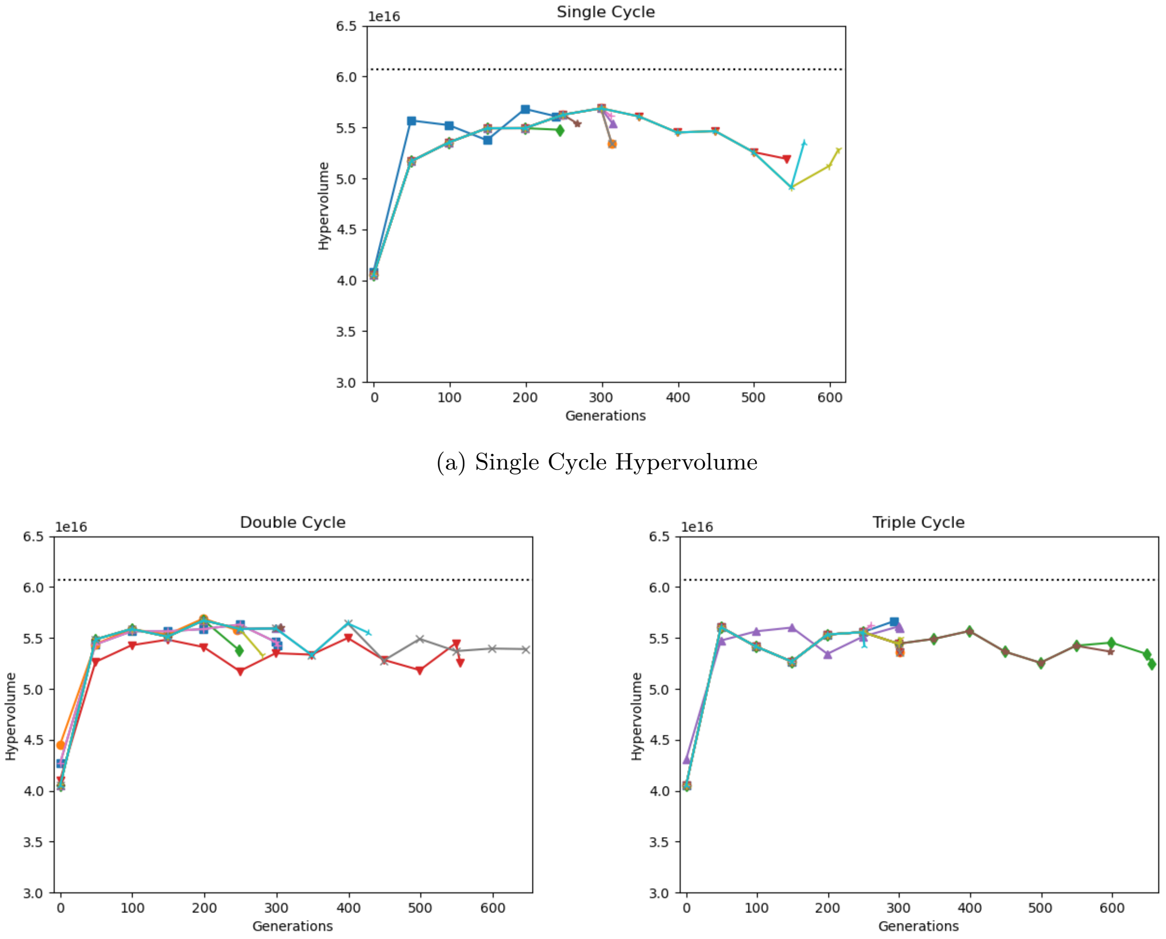
(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure 4.8: August Objective Space

4.4 Year Order Cycle

For this experiment the simulated month was changed every generation starting with January and following the order of the year. This way the optimal point will move every generation, essentially creating a dynamic fitness function, which makes it more difficult for the algorithm to optimize. To try and counteract this problem the experiment was repeated with a double and triple cycle where the same month is repeated two times and three times. The hypothesis is that this could give the optimizer more time to optimize toward a certain optimum before moving it, improving the results.



(a) Single Cycle Hypervolume

(b) Double Cycle Hypervolume

(c) Triple Cycle Hypervolume

Figure 4.9: Hypervolume for different Year Cycle experiments.

Figure 4.9 shows the hypervolume graphs for the cycle experiments. For the single cycle the hypervolume decreases after a certain point, while the other cycles remain more stable. Searching for an explanation, we notice a comparable behaviour if we cross reference with the single month hypervolume graphs in Appendix A. Here we observe that for certain months the hypervolume decreases after a certain period of time. Combining these results would give us a downward trend on average. A possible hypothesis could be that since all months are included in the cycle, it is influenced by all of them and therefore the hypervolume for the cycle is following this trend.

Appendix E shows all the objective distributions for the cycle experiments. When comparing them to each other it is noticeable that with all three experiments the distinct shape for the resilience objective is lost after about 299 generations. Especially the single cycle finds a lot of solutions with a low resilience in the final generations. As for the yearly costs and CO_2 objectives, all three experiments find a lot of solutions that have high values for those objectives. This correlates with the unstable or decreasing hypervolume in the final 200 generations.

The objective space graphs for the cycle experiments are depicted in Appendix F. The most interesting graph is the one comparing investment cost to resilience. When compared to the baseline experiment it is clear that there are very few solutions with a resilience over 900. Especially for the single cycle experiment which seems to prioritize low investment cost over higher resilience in the final generation. It essentially abandons the Pareto front that was found in the 50th generation with an edge at 900s resilience. An answer could be that it trades high resilience for investment cost to generate a lot of cheap solutions. Though this also increases the other objective values. The exact reason for this is unclear.

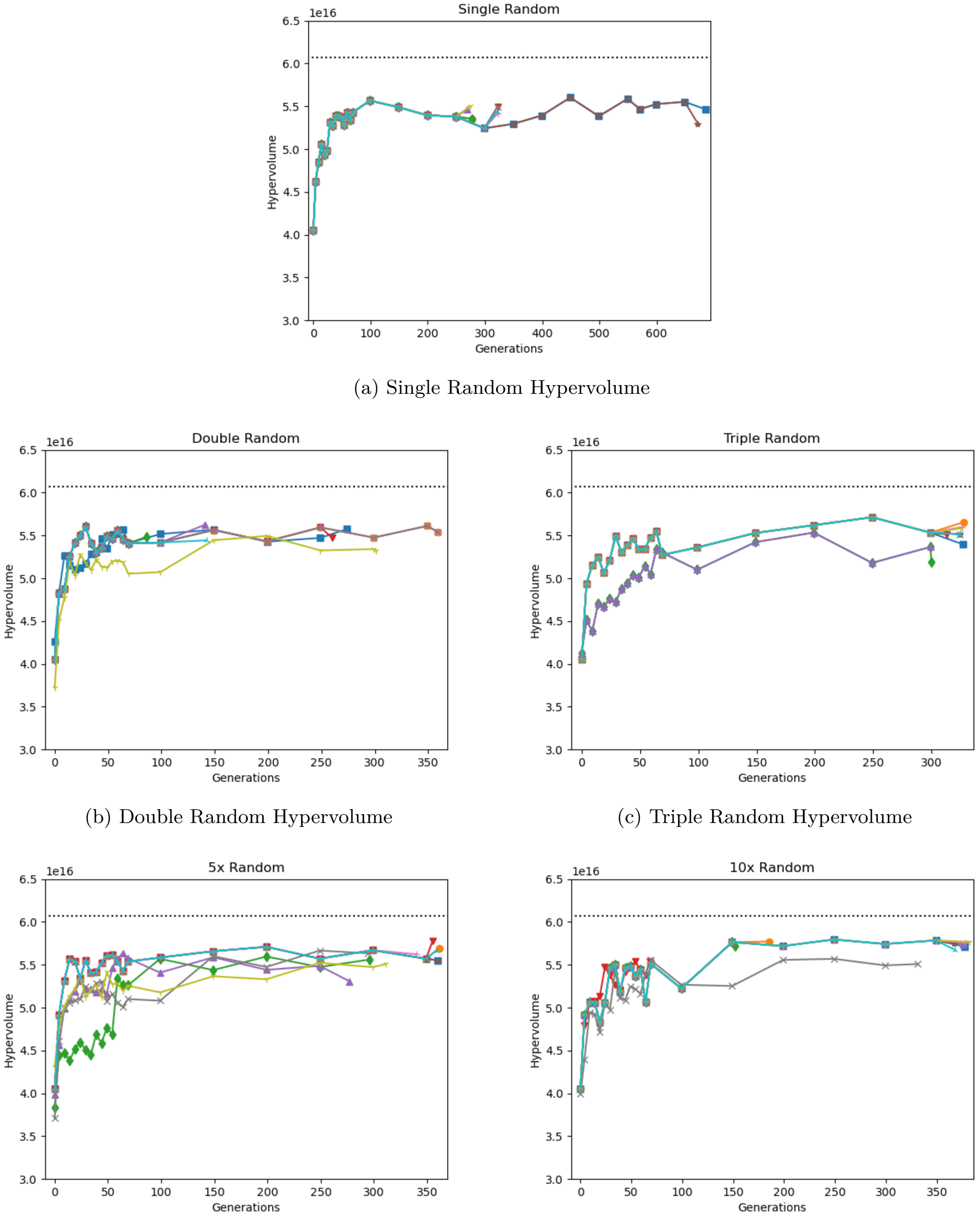
4.5 Random Months

With this experiment the simulated month is randomized every generation, just like with the cycle experiment. This moves the optimal point like a dynamic fitness function, but this time at random. Similarly to the cycle experiment there are versions where the randomly chosen month is not changed for a number of generations. Originally only a double and triple repetition experiment were planned, like with the cycle, but we added a five times and ten times experiment since these repetitions are performing well.
A thing to note is that the randomly chosen month can be the same as the previous month for all these experiments. This can result in a single month being repeated an infinite amount of times, though this has a very low chance of occurring and did not happen during our experiments.

Figure 4.10 contains the hypervolume graphs for the random month experiments. Here it is noticeable that repeating the month multiple times improves the stability and the highest hypervolume score. There might be an optimal number of $N\times$ repetitions, with N smaller than the maximum number of generations, since otherwise it converges to that single month.
Something to notice here is the fact that when the number of times a month is repeated is increased, the number of generations it can do within four hours decreases. This is most likely due to the fact that the different months have different simulation times, since when a month with a longer simulation time gets randomly chosen, it takes longer than simulating a month with a shorter simulation time. When repeating the same month multiple times, this potential difference also gets significantly larger.

Appendix G shows the objective distributions for the random experiments. Here the distinctive shape of the graph for the resilience objective is almost not visible in the single random experiment, but comes through with the increased repetitions. This means that adding repetitions helps to find the Pareto front. The same can also be said for the other objectives.

In Appendix H we can see the objective space graphs for the random experiments plotted. Here again the graphs with investment cost and resilience have the most interesting shape. From these graphs it is clear that adding repetitions helps the optimizer to find the Pareto front. In the single random the solutions are very much spread out. Every repetition, however, finds more and more of the Pareto front.



(a) Single Random Hypervolume

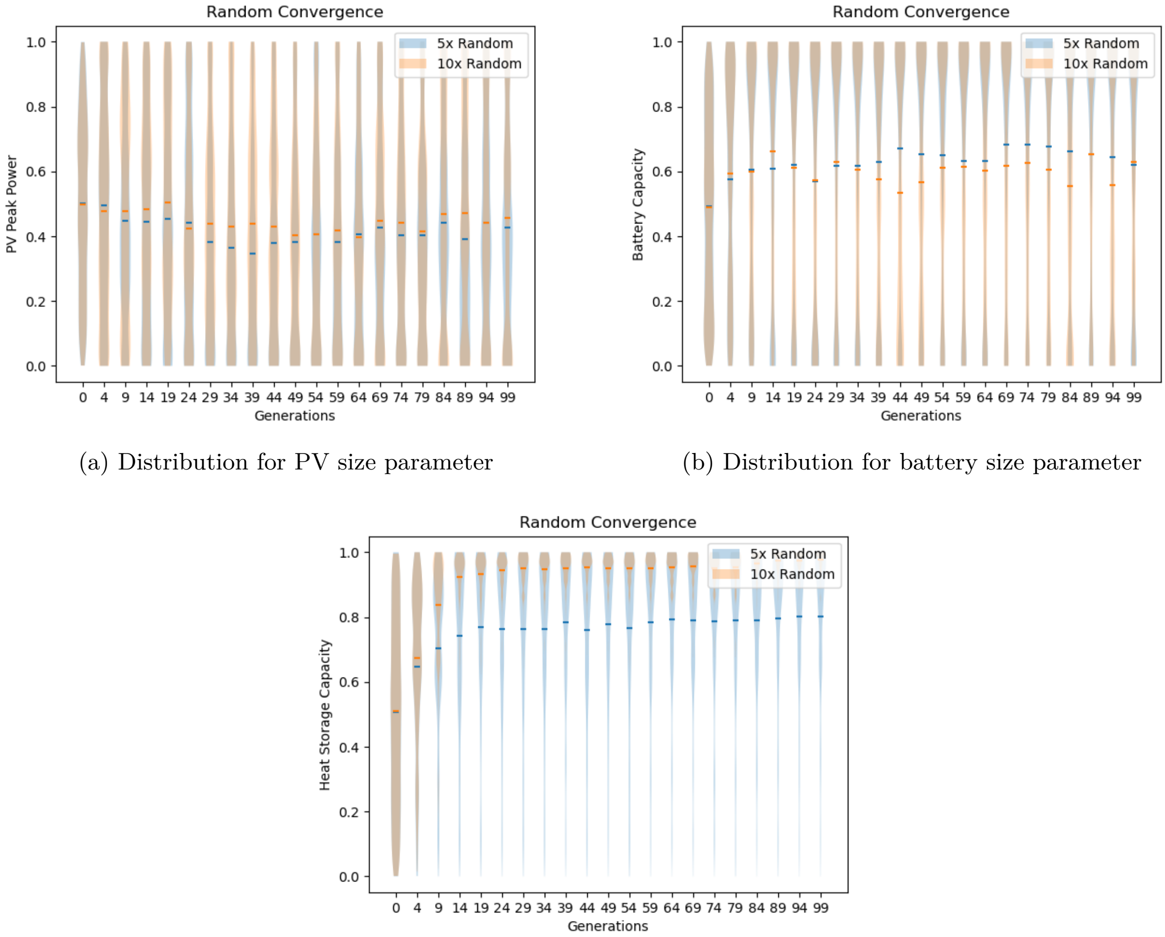
(b) Double Random Hypervolume

(c) Triple Random Hypervolume

(d) 5x Random Hypervolume

(e) 10x Random Hypervolume

Figure 4.10: Hypervolume for different Random Month experiments.



(a) Distribution for PV size parameter

(b) Distribution for battery size parameter

(c) Convergence for heat storage size parameter

Figure 4.11: 5x and 10x Random parameter distributions

Figure 4.11 shows the distribution of the most important parameters for the 5x and 10x Random runs. When comparing to the baseline parameters in Figure 4.5 it is noticeable that for PV and heat storage capacity, the 10x Random run is much closer to the baseline result, but for the battery capacity the 5x Random experiment is closer, though the shapes of the distribution are very similar. The biggest difference can be observed for the heat storage capacity where the 10x Random run quickly finds an optimum at a high capacity heat storage. The fact that it finds this optimum more quickly compared to the 5x Random run can most likely be attributed to the added repetitions of the months.

4.6 IGD+

As an additional comparison metric IGD+ [5] was used. To compute the IGD+ of a solution set it requires the actual optimal Pareto front of the problem as a reference set. However, since this is a real world problem instead of a benchmark function, there is no optimal Pareto front available. Therefore we computed the non dominated solution set from all individual solutions from all runs from all experiments, and used this as the reference Pareto front. The expectation here is that the experiments with the highest hypervolume score have the lowest IGD+ score since they find the most non dominated solutions.

Experiment	IGD+ Score
Full Year	0.98
Most Similar Month	121.97
Single Cycle	200.08
Double Cycle	226.66
Triple Cycle	245.95
Single Random	307.34
Double Random	147.05
Triple Random	187.92
5x Random	146.07
10x Random	146.88

Table 4.2: IGD+ Scores

Table 4.2 contains the IGD+ scores for our different experiments. The most interesting observation is that for the Random experiments the IGD+ score fluctuates when repeating the chosen month more often. These fluctuations have a downward trend instead of a clear improvement every time the amount of repetitions increase. With Double Random the score is lower then with Triple Random. Single Random is higher than both however. The fluctuations can potentially be explained by the randomness of the optimizer.

Another interesting observation is that the IGD+ score for the Cycle experiments increases when you repeat the same month more often. This is not what we initially expected, since the hypervolume of those experiments becomes more stable when increasing the number of repetitions.

4.7 Accelerating HV Development

Looking at all hypervolume graphs from the previous experiments it is clear that they have a common difference to the baseline experiment: The hypervolume increases quickly in the first few generations, but the curve flattens out to a lower final hypervolume score compared to the baseline. This is contrary to the baseline experiment where the increase is much more gradual, but continuously increasing. While the final hypervolume score is not the best, this fast hypervolume increase in the first few generations of these experiments can be exploited if it is faster compared to the baseline. If so, a potential strategy would be to start an optimization run with random months for a short period of time, and then switch over to the regular full year simulation for the remaining evaluations.

To explore this possibility we create a plot for the first hour of the hypervolume graph for the baseline experiment and the 10x Random experiment in Figure 4.12. We choose the 10x Random experiment as comparison since it shows a better hypervolume improvement in the first few generations in Figure 4.10.

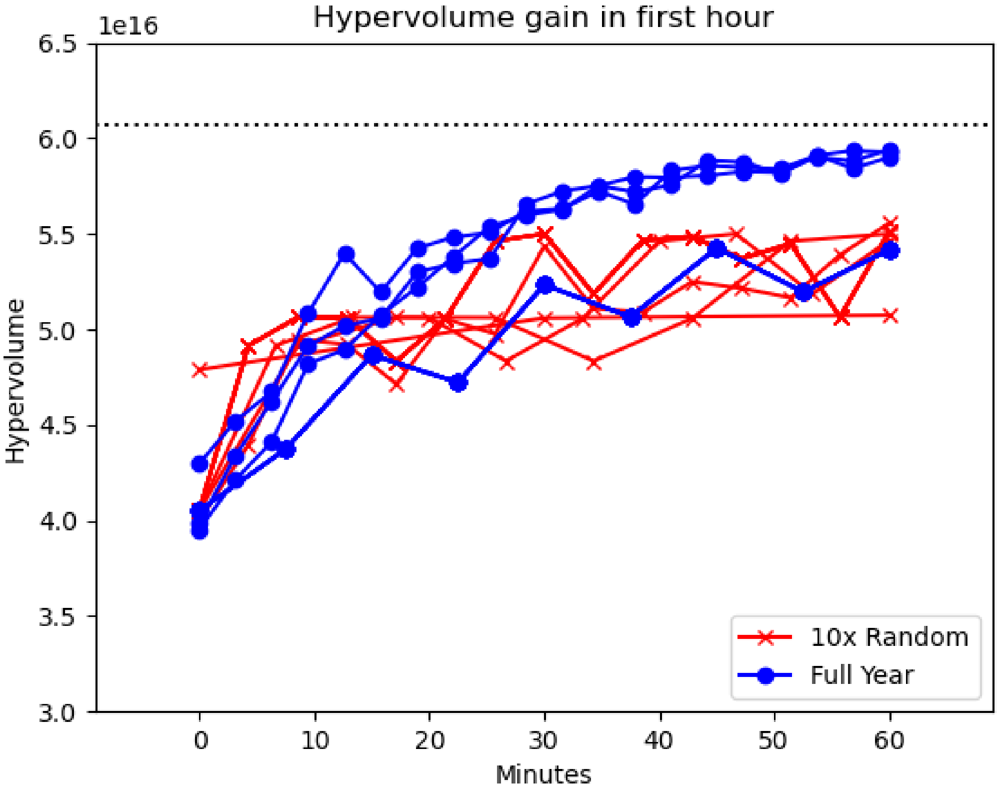


Figure 4.12: Hypervolume gain in the first hour. The blue lines represent the ten runs from the baseline experiment, while the red lines represent the ten runs from the 10x Random experiment.

In Figure 4.12 we observe that for the first 9 to 10 minutes the hypervolume of the 10x Random run increases faster in most runs compared to the baseline experiment. Between 10 and 15 minutes almost all of the baseline runs equal or overtake the 10x Random runs, most of which seem to stagnate or increase only slightly after this time.

To further clarify these observations we calculated the average hypervolume increase over ten minute timeframes. These values are shown in Table 4.3. These values show that, as stated earlier, the hypervolume for the 10x Random run increases faster compared to the baseline experiment. For the next timeframes the average increase becomes very inconsistent, going negative, positive and negative again, indicating the stagnation for these runs. For the baseline experiment the average increase decreases but remains positive for longer. At a certain point it turns negative and positive again. If we look at Figure 4.12 again we observe that one of the baseline runs is not performing as well as the rest. This outlier can explain the inconsistencies in the average increase for the full year simulation.

Timeframe	Avg. incr. 10xR	Avg. incr. FY
0 - 10 m	5.472×10^{15}	3.019×10^{15}
10 - 20 m	-2.245×10^{15}	1.056×10^{15}
20 - 30 m	2.652×10^{15}	1.108×10^{15}
30 - 40 m	-3.912×10^{14}	-7.572×10^{14}
40 - 50 m	-8.681×10^{14}	2.756×10^{14}
50 - 60 m	6.868×10^{14}	1.144×10^{15}

Table 4.3: Average Hypervolume Increase

While these numbers may provide a positive outlook at a first glance, this outlier highlights a potential issue with these results: The randomness of the optimizer has a large influence on these runs. If the outlier from the baseline result was not there, the difference in average HV increase would be much smaller. If we look at Figure 4.12 again we can see a few runs from the 10x Random experiment almost fully overlap with some baseline runs. For those runs the hypervolume increase only equals the baseline experiment. The fact that this occurs shows us that the randomness of the optimizer can play a large role in whether or not the previously mentioned strategy would work.

5 - Conclusions & Discussion

Our main research question was: For the given energy system, is it possible to get a qualitatively equivalent result when simulating a shorter period of time? After running all the experiments and comparing the hypervolume and IGD+ results of our experiments to our baseline, the short answer is no. None of the HV or IGD+ scores of our experiments can directly match the baseline result. Therefore simply replacing the full year simulation with one of the methods from our experiments is not efficient. These results are not unexpected, since we are trying to have the optimization algorithm solve the problem with limited data.

But we do learn a lot about the simulator from our experiments:
Firstly the fact that the month that is most similar to the full year simulation is August. The initial expectation was a spring or autumn month, since the expectation was that those are closer to average in terms of sunshine for the PV system (with summer having more sunshine and winter less sunshine). April is the second best option in terms of hypervolume, but is less attractive because of the amount of generations it manages to complete within four hours. An explanation for this is that the computations for the CHP system are quite complex and require more time. This switching probably happens during April but not in August, resulting in more evaluations. The fact that August can do more generations within four hours is most likely the main reason for the better HV and IGD+ scores compared to April. With a single month the optimizer needs more generations to converge to solutions similar to the baseline which is visible in the objective distributions and objective space.

For the cycle and random experiments we expected that they would get closer to the baseline results when repeating the chosen month more often, since the optimizer has more time to optimize in the direction of the optimal points for that month.
For the cycle experiments this hypothesis was proven to be incorrect. The HV decreases over time and IGD+ scores increase when increasing the amount of repetitions.
The random experiments however do seem to improve when increasing the amount of times a month is repeated. It is more clearly visible in the HV results since the IGD+ scores for double, 5x and 10x random are within margin of error from each other. There might be an optimal amount of repetitions for the month in this case which could be interesting to find.

However a surprising observation is that when combining one of these methods with the full year simulation there might be a way to accelerate the hypervolume development within the constrained time. When using for example the 10x random method for the first few generations it is possible to get to a certain level of HV much faster compared to the full year simulation. Once this level is reached or when the HV is visibly stagnating, it could switch to the full year simulation. This might result in reaching an equal amount of HV in a shorter time or a higher amount in the same time. For this simulator the point where switching back would be a good idea is reached fairly quickly, within a few generations. But this might be interesting for more complex simulations that take more time since the time saved might scale with the total simulation time. While this seems optimistic, more statistical research would be required for this method. It would have to be consistently better or equal to fully switch to this strategy.

In conclusion we did not find a straight up more or at least equally efficient method to replace the time consuming full year simulation, but combining the repeated random month with the full simulation could result in a faster initial hypervolume increase. Saving computational time by simply reducing the simulation time might sound very attractive to practitioners, but this thesis proves this not to be a viable approach.

5.1 Future Work

For the future of this project there are a few more experiments relating to the previously described findings that could be done using the current setup.

To begin with an experiment could be set up to see if the earlier mentioned strategy to speed up hypervolume development works properly. We would have to do some statistical analysis to see if it is an improvement for most runs, and how much better it is. We would also have to finetune the point at which to switch over to the full year simulation. To do this a hypervolume calculation could be added after evaluating each generation with a single month. This would be compared with the HV of the previous generation and if the difference is smaller than a certain threshold, the evaluation function could switch to the full year evaluation. Afterwards we can compare this to the baseline experiment. Another option would be to do a few different short runs to figure out when exactly the HV will stagnate with the different month-swapping strategies, and how much better it is compared to the full year. Lastly for optimization scenarios like this, it might be interesting to create a dynamic optimization scenario from the beginning. For example it can start from a simpler simulation with less simulation time or data and over time extend to the full simulation, slowly extending or adding in data every generation. This way at least in the first few generations time can be saved using the simpler simulation, while potentially still achieving good final results.

Besides these experiments there are a couple of updates and upgrades that are already being worked on. In the first place the simulator model and optimization problem are to be expanded to include electric company vehicles and a charging system with a charging algorithm that can decide to charge or discharge cars based on available electricity. New objectives would include the amount of electric cars (as opposed to ICE cars) and the amount of charge those cars get. New parameters to influence these would be related to the amount of charging stations and fast charging stations, amongst others.

This new model is finished already but when testing it the real world simulation time was significantly longer due to the complexity of the charging controller. This would mean that there would be even less generations within a time limit available for optimization. To alleviate this problem, experiments with a surrogate AI to replace the simulation are underway [1]. Some potential problems with this surrogate could be the fact that it still requires input and output data generated by the framework for training, which could take some time to generate. Also changing locations by switching out weather data and consumption data would require generating a new training dataset for the surrogate. But all in all after gathering the data, using the surrogate is much faster compared to running the simulation and could save a lot of time, even if it is not fully accurate.

For the optimizer there are also a few interesting options that should be tried. Instead of using IBEA, a state of the art Efficient Global Optimization (EGO) [6] algorithm could be used, for example SMS-EGO [9]. This uses a Kriging model to essentially predict where the most optimal points are and optimizes quickly in their direction. These kinds of EGO algorithms are very efficient and could find an optimal configuration in less time.

Another option would be to use a specialized or enhanced algorithm for dynamic fitness functions [2] for our experiments where the simulated period is changed every generation. It would be interesting to find out if these algorithms would perform better in these cases.

6 - Acknowledgements

I would like to thank all my supervisors for their great advice and patience with me.
I am also very grateful to HRI for providing me with this opportunity to do research with them.

I would also like to thank to my parents for their support and moving me to and from Germany, and my aunt for letting me live with her for a while to write this thesis.

For helping me write and improve my thesis I would like to thank Alexandra Blank for her great advice and coaching, and Max Doesburg for proofreading and pushing me to finally finish this thesis.

And many thanks to Fons Verbeek for lending me a spot in one of his offices to work at.

Bibliography

[1] Pouya Aghaei Pour, Tobias Rodemann, Jussi Hakanen, and Kaisa Miettinen. Surrogate assisted interactive multiobjective optimization in energy system design of buildings. Optimization and Engineering, 23(1):303–327, 2021.

[2] Jürgen Branke and Hartmut Schmeck. Designing evolutionary algorithms for dynamic optimization problems. Natural Computing Series Advances in Evolutionary Computing, page 239–262, 2003.

[3] P. Fritzson and P. Bunus. Modelica - a general object-oriented language for continuous and discrete-event system modeling and simulation. Proceedings 35th Annual Simulation Symposium. SS 2002, 2002.

[4] ESI Group. SimulationX Software. `https://www.esi-group.com/products/system-simulation`. [accessed 01-04-2021].

[5] Hisao Ishibuchi, Hiroyuki Masuda, Yuki Tanigaki, and Yusuke Nojima. Modified distance calculation in generational distance and inverted generational distance. Lecture Notes in Computer Science Evolutionary Multi-Criterion Optimization, page 110–125, 2015.

[6] D.R. Jones, M. Schonlau, and W.J Welch. Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13, page 455–492, 1998.

[7] R. Luna-Rubio, M. Trejo-Perea, D. Vargas-Vázquez, and G.J. Ríos-Moreno. Optimal sizing of renewable hybrids energy systems: A review of methodologies. Solar Energy, 86(4):1077–1088, 2012.

[8] Aeidapu Mahesh and Kanwarjit Singh Sandhu. Hybrid wind/photovoltaic energy system developments: Critical review and findings. Renewable and Sustainable Energy Reviews, 52:1135–1147, 2015.

[9] Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$ -metric selection. Parallel Problem Solving from Nature – PPSN X Lecture Notes in Computer Science, page 784–794, 2008.

[10] Tobias Rodemann. A comparison of different many-objective optimization algorithms for energy system optimization. Applications of Evolutionary Computation Lecture Notes in Computer Science, page 3–18, 2019.

[11] J.M. Sala. 20 - thermal energy storage (tes) systems for cogeneration and trigeneration systems. In Advances in Thermal Energy Storage Systems, pages 493–509. Woodhead Publishing, 2015.

[12] H. Suryoatmojo. Artificial Intelligence Based Optimal Configuration of Hybrid Power Generation System. PhD dissertation, Graduate school of science and technology, Japan, Department of computer science and engineering, Kumamoto University, 2010.

[13] Ye Tian, Ran Cheng, Xingyi Zhang, and Yaochu Jin. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization. IEEE Computational Intelligence Magazine, 12(4):73–87, 2017.

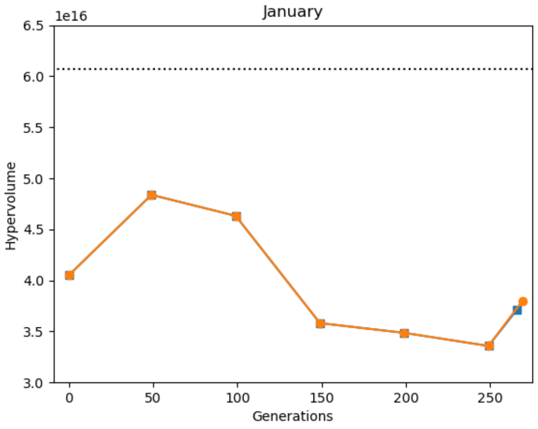
[14] Subho Upadhyay and M.P. Sharma. A review on configurations, control and sizing methodologies of hybrid energy systems. Renewable and Sustainable Energy Reviews, 38:47–63, 2014.

[15] A. Yahiaoui, F. Fodhil, K. Benmansour, M. Tadjine, and N. Cheggaga. Grey wolf optimizer for optimal design of hybrid renewable energy system pv-diesel generator-battery: Application to the case of djanet city of algeria. Solar Energy, 158:941–951, 2017.

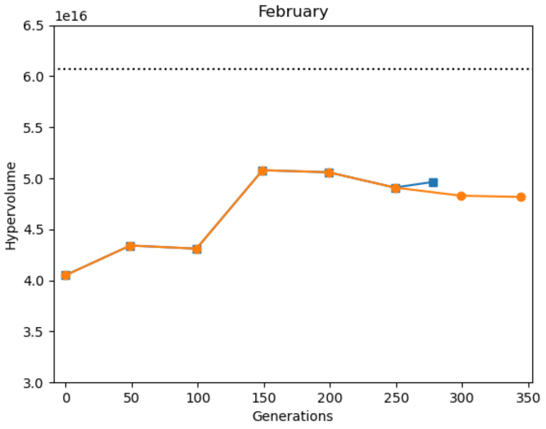
[16] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. Lecture Notes in Computer Science Parallel Problem Solving from Nature - PPSN VIII, page 832–842, 2004.

Appendices

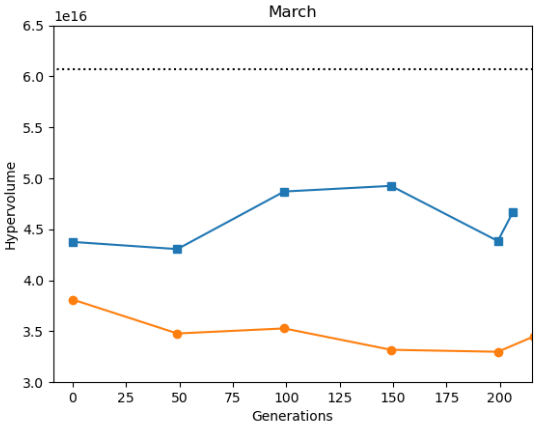
A - Individual Month Hypervolume Graphs



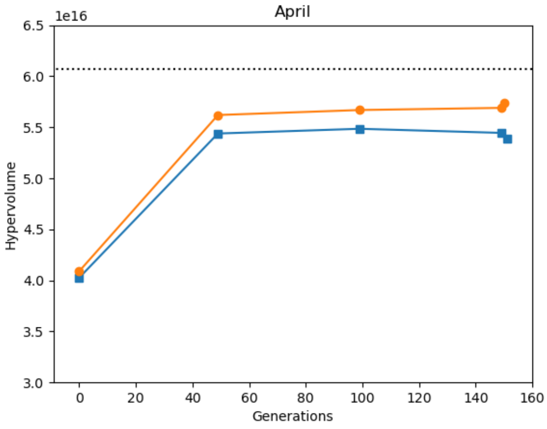
(a) January HV



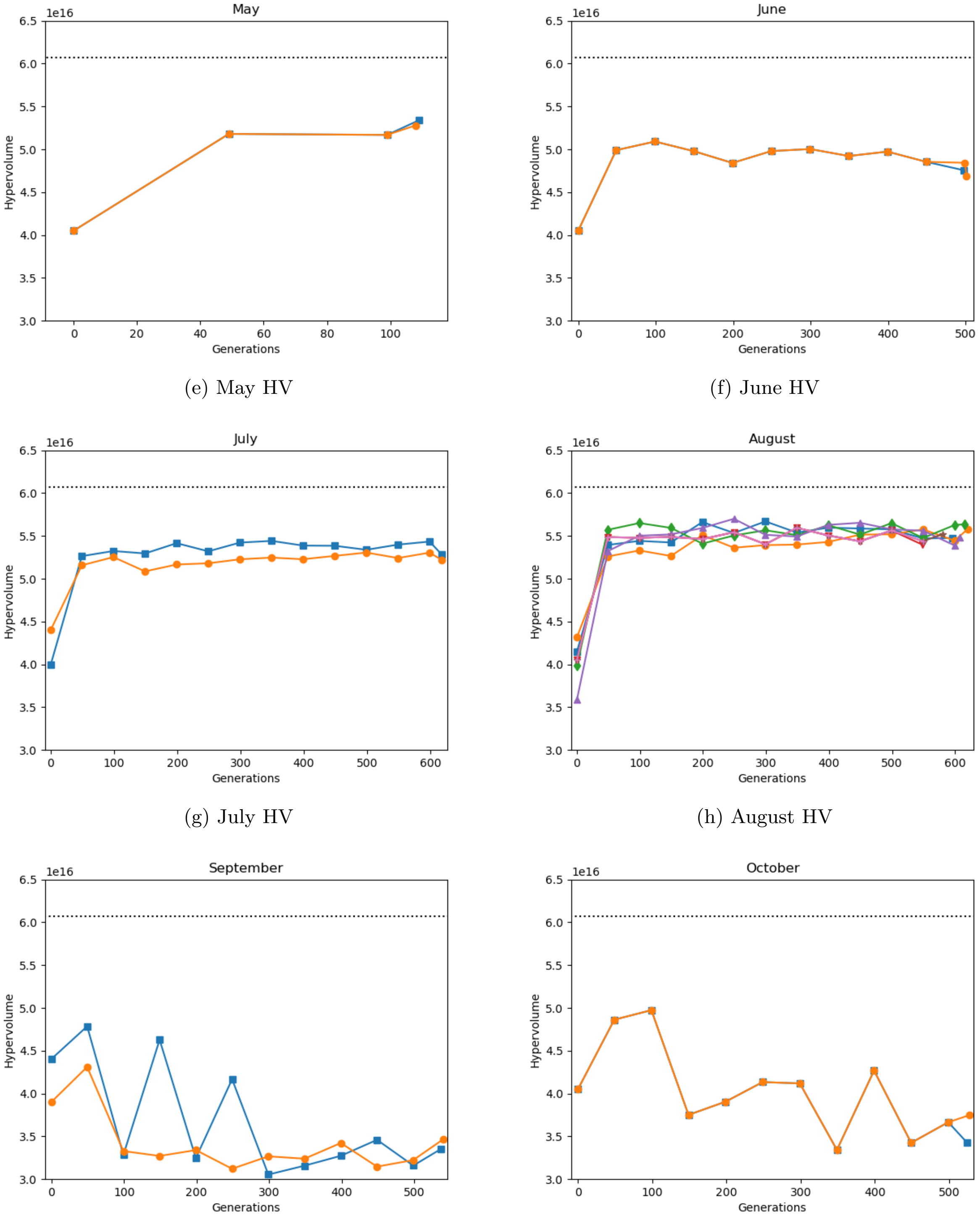
(b) February HV



(c) March HV



(d) April HV



(e) May HV

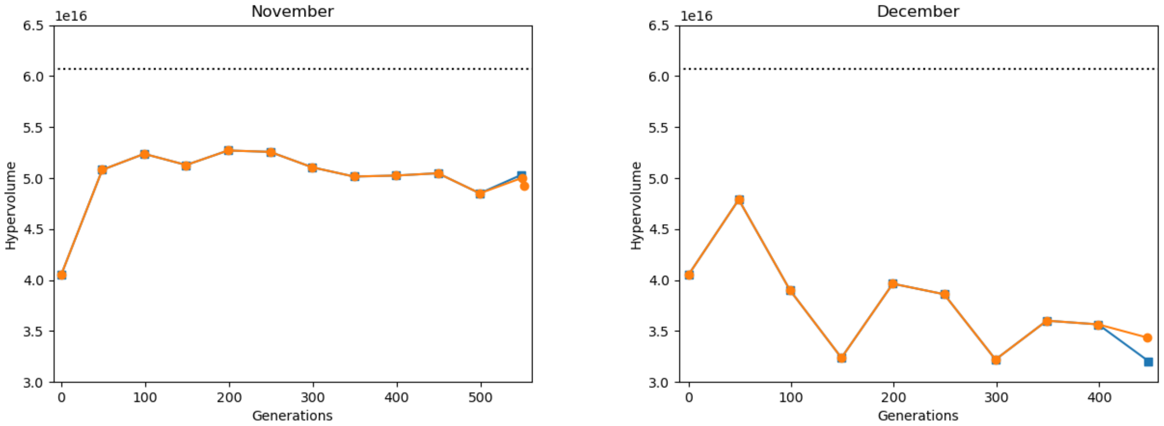
(f) June HV

(g) July HV

(h) August HV

(i) September HV

(j) October HV



(k) November HV

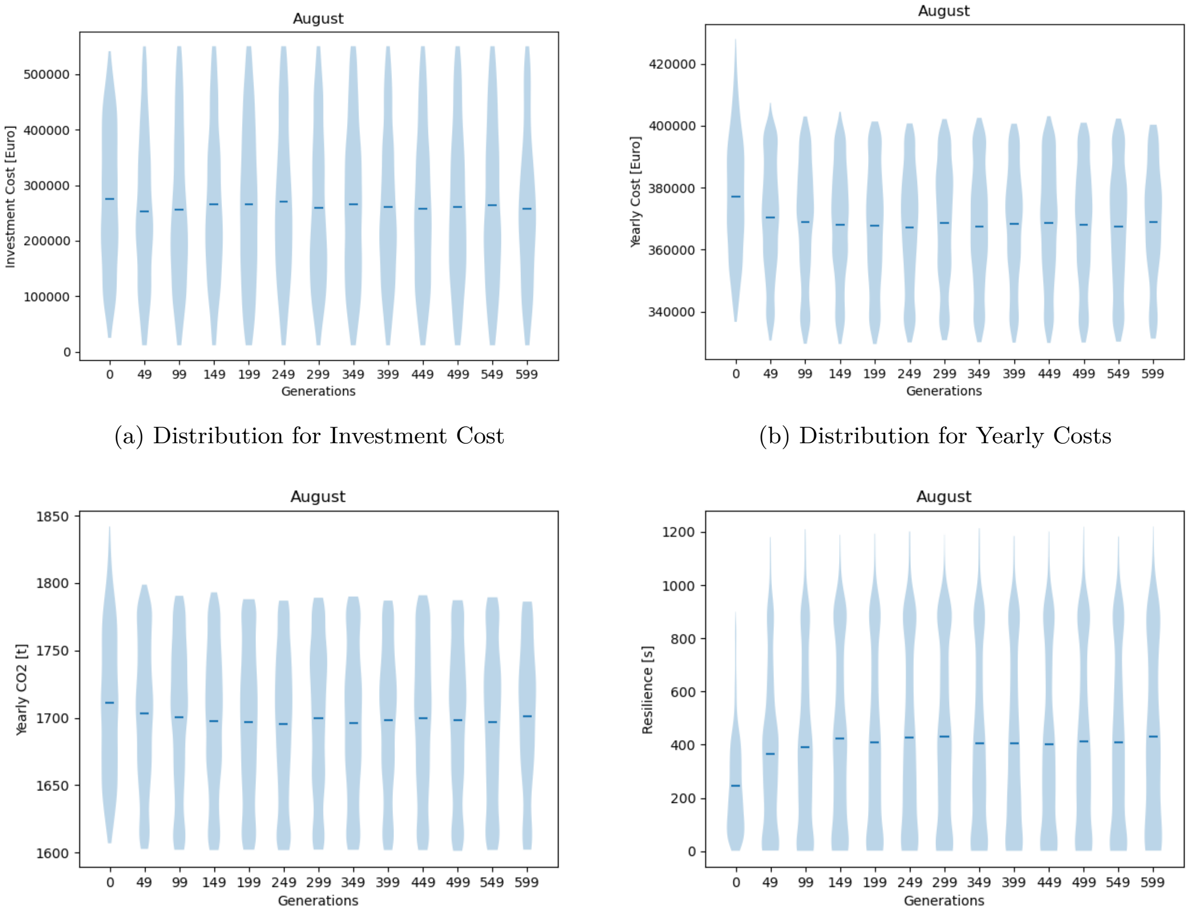
(l) December HV

B - Individual Month IGD+ Scores

Month	IGD+ Score
January	884.70
February	727.52
March	532.50
April	247.01
May	544.01
June	814.29
July	474.14
August	121.97
September	642.99
October	664.49
November	804.48
December	1107.78

Table B.1: Single Month IGD+ Scores

C - August Objective Distributions



(a) Distribution for Investment Cost

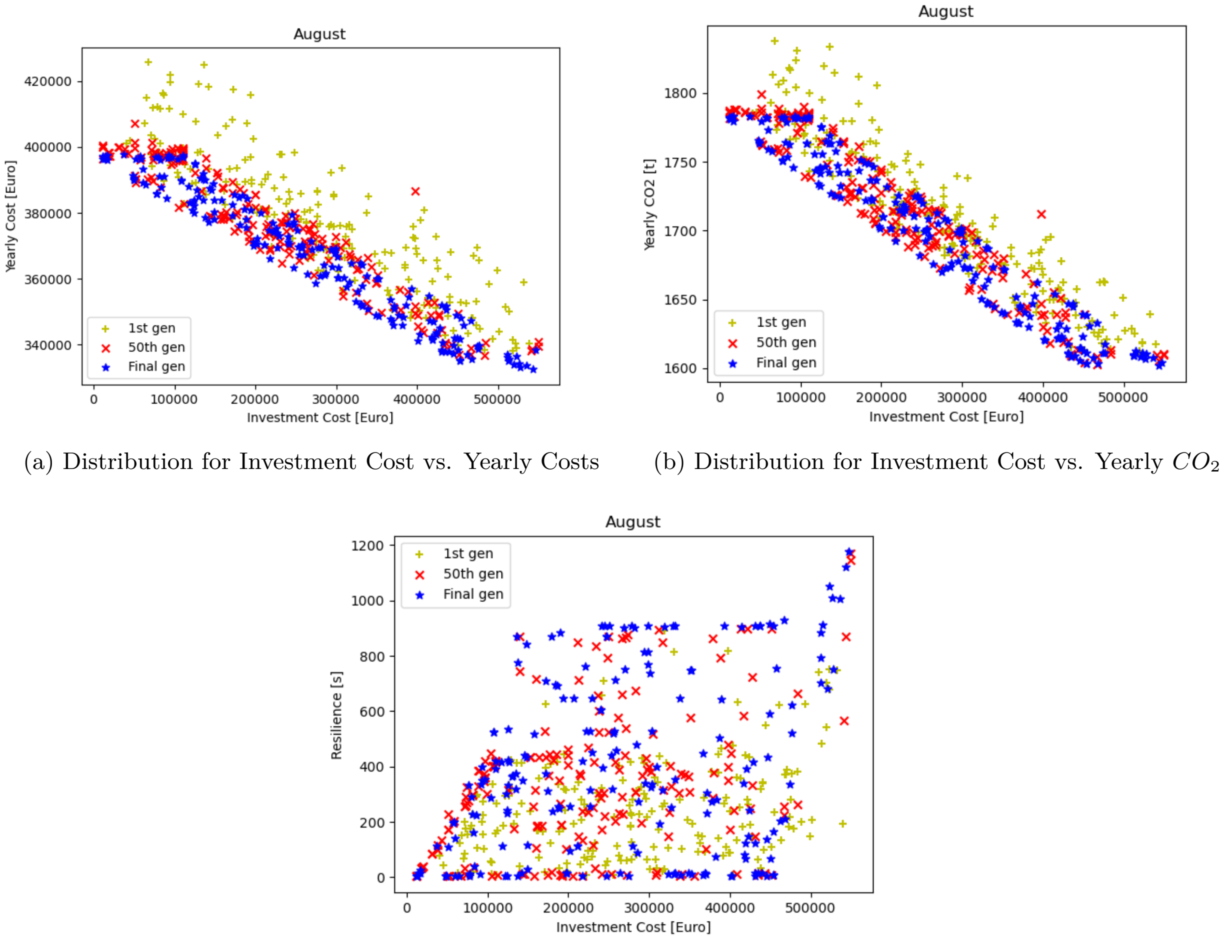
(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure C.1: August Objective Distributions

D - August Objective Space



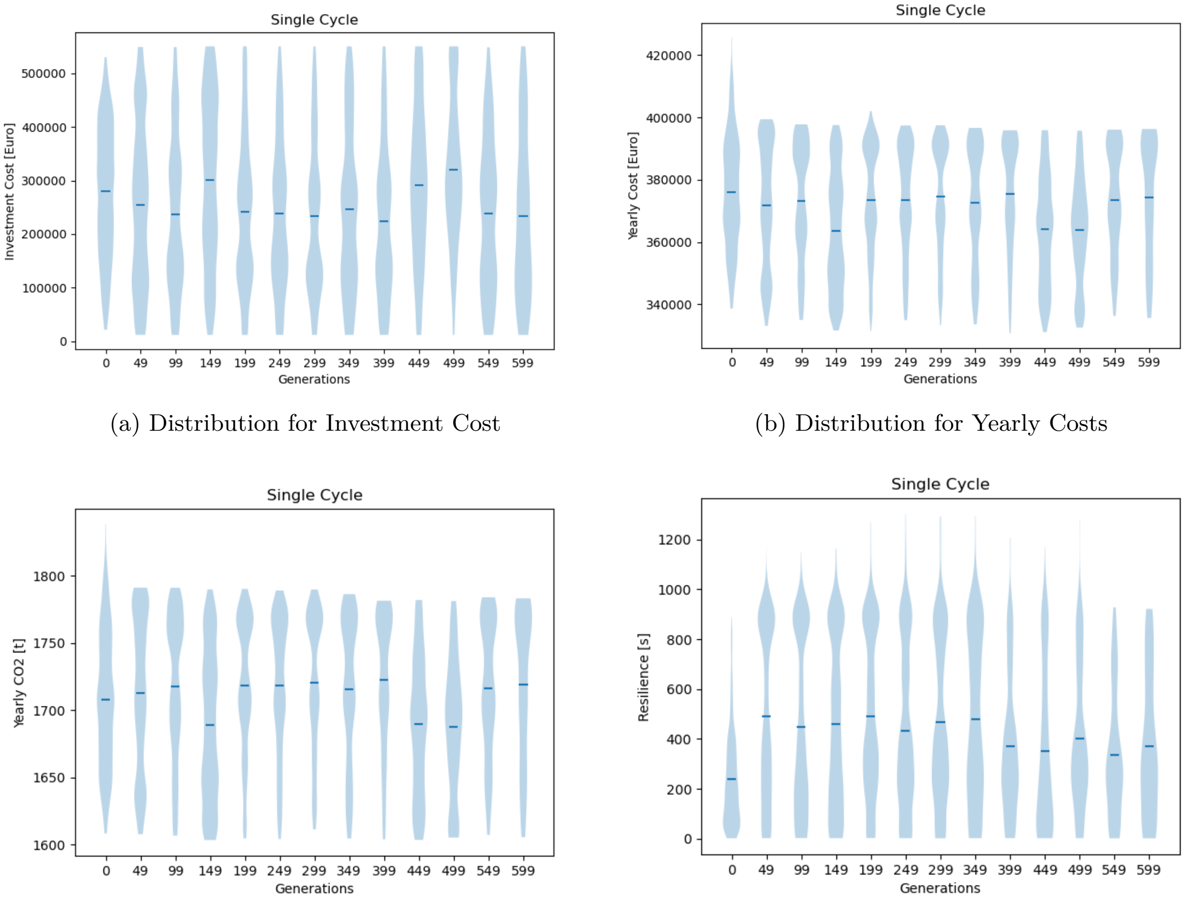
(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure D.1: August Objective Space

E - Cycle Objective Distributions



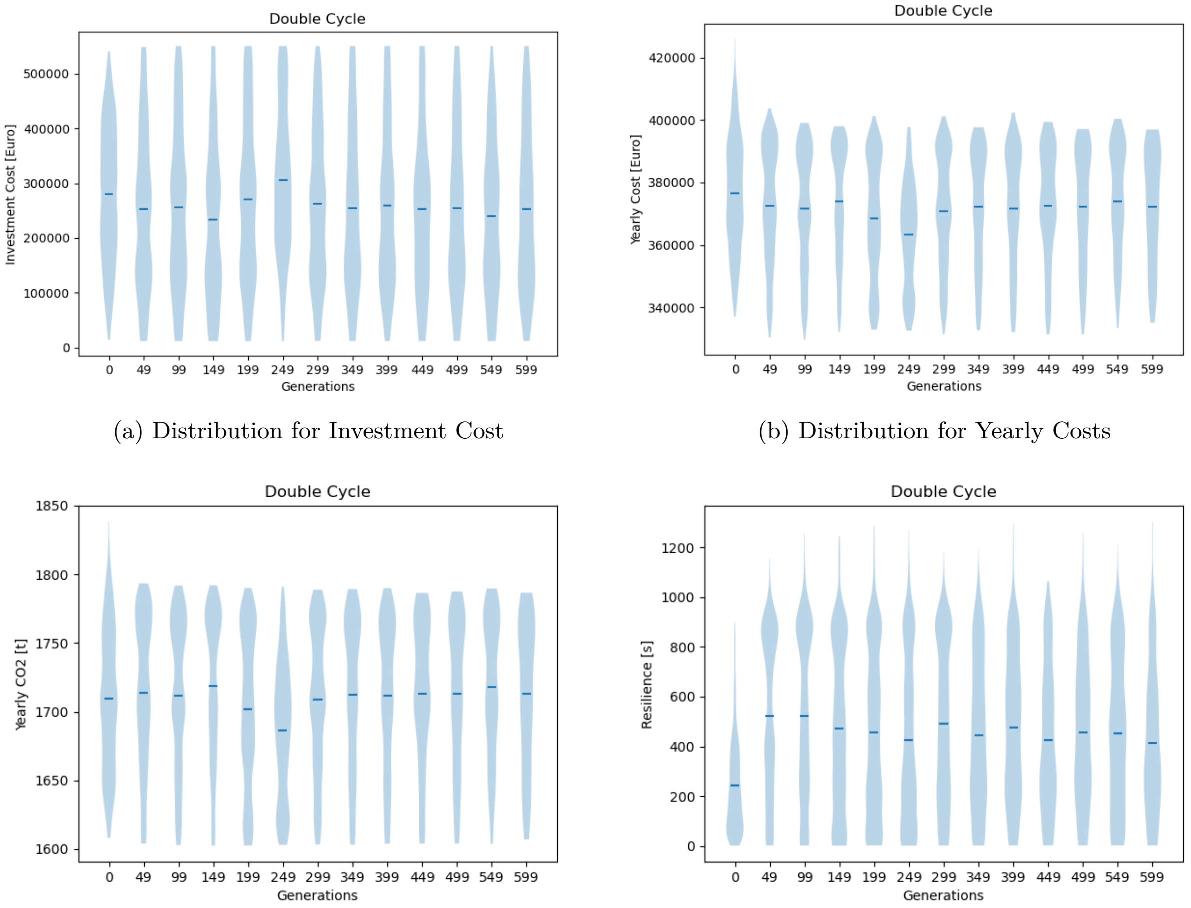
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure E.1: Single Cycle Objective Distributions



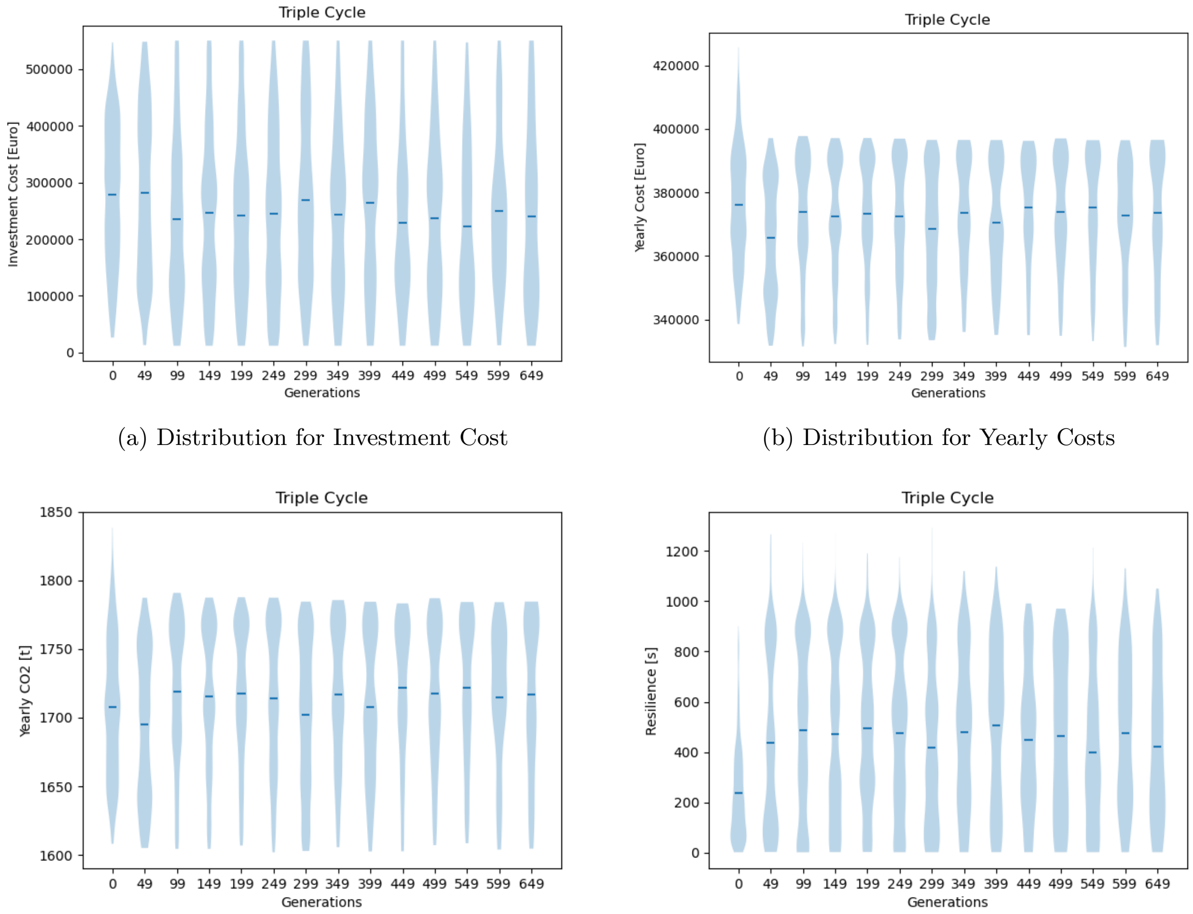
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure E.2: Double Cycle Objective Distributions



(a) Distribution for Investment Cost

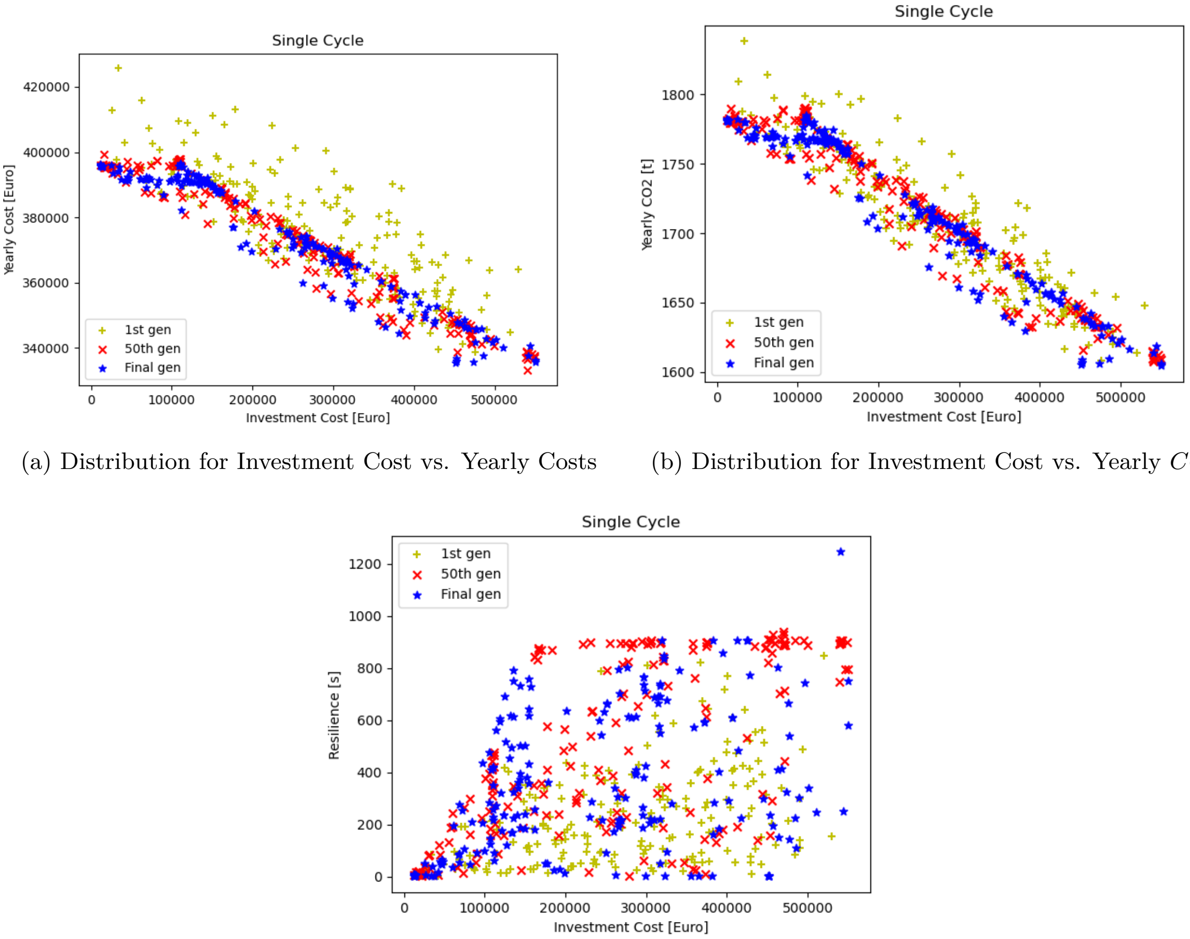
(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure E.3: Triple Cycle Objective Distributions

F - Cycle Objective Space

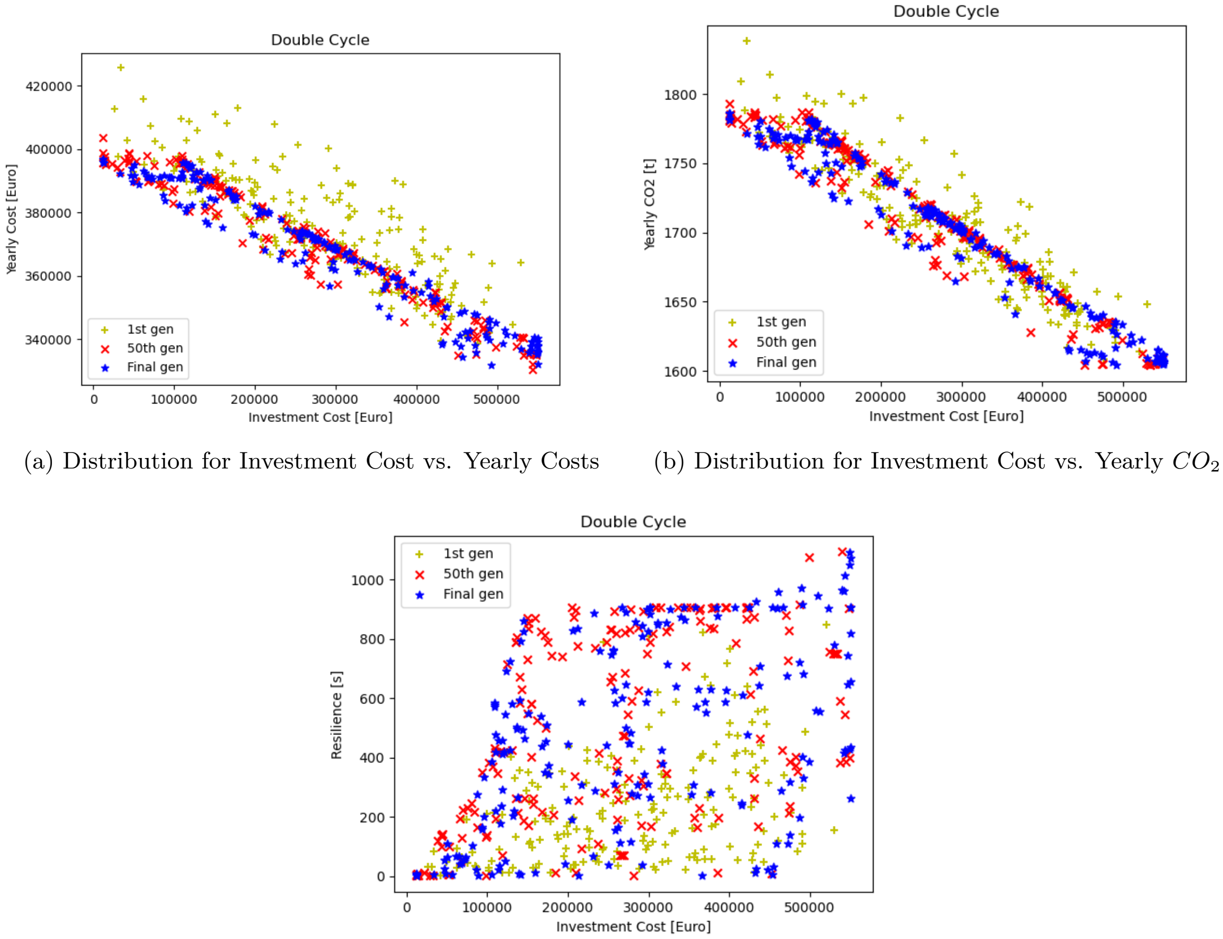


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure F.1: Single Cycle Objective Space

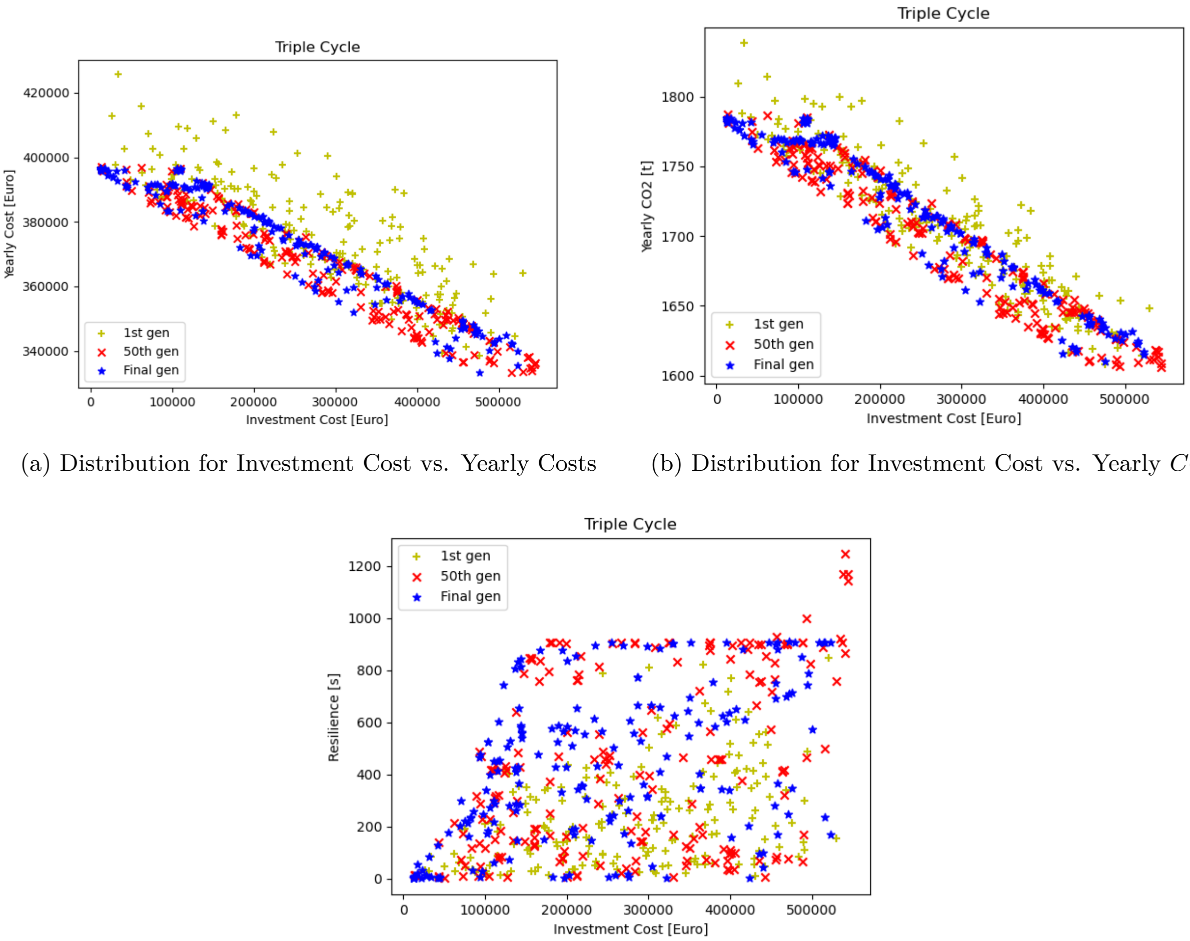


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure F.2: Double Cycle Objective Space



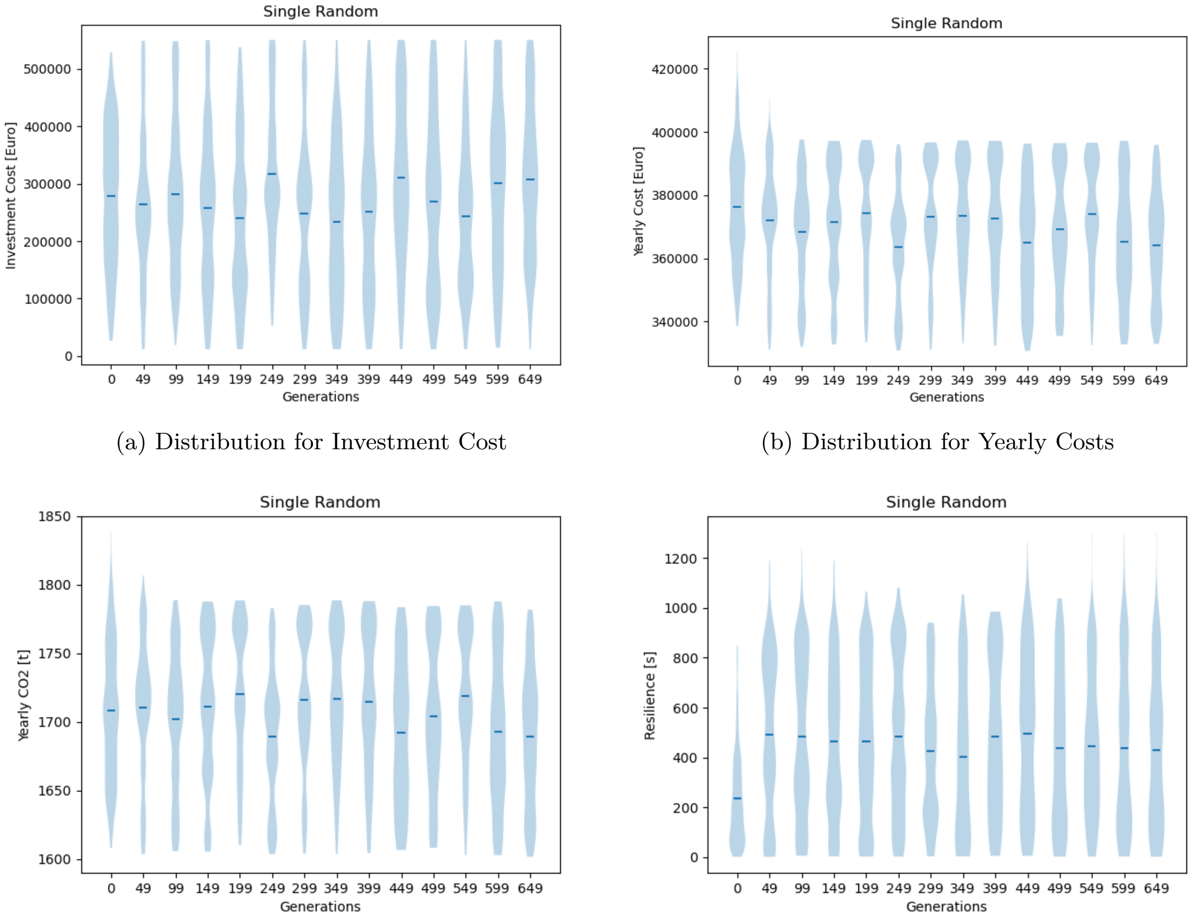
(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure F.3: Triple Cycle Objective Space

G - Random Objective Distributions



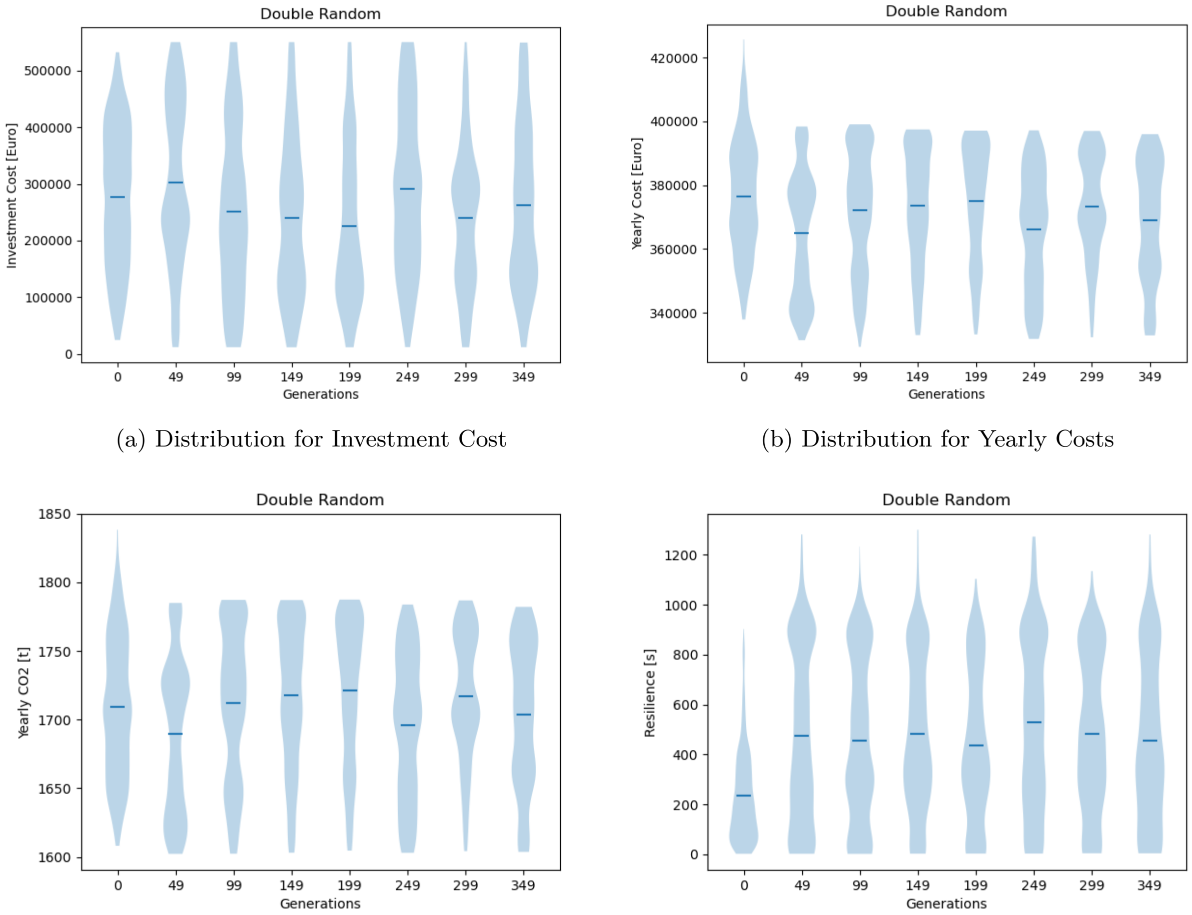
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure G.1: Single Random Objective Distributions



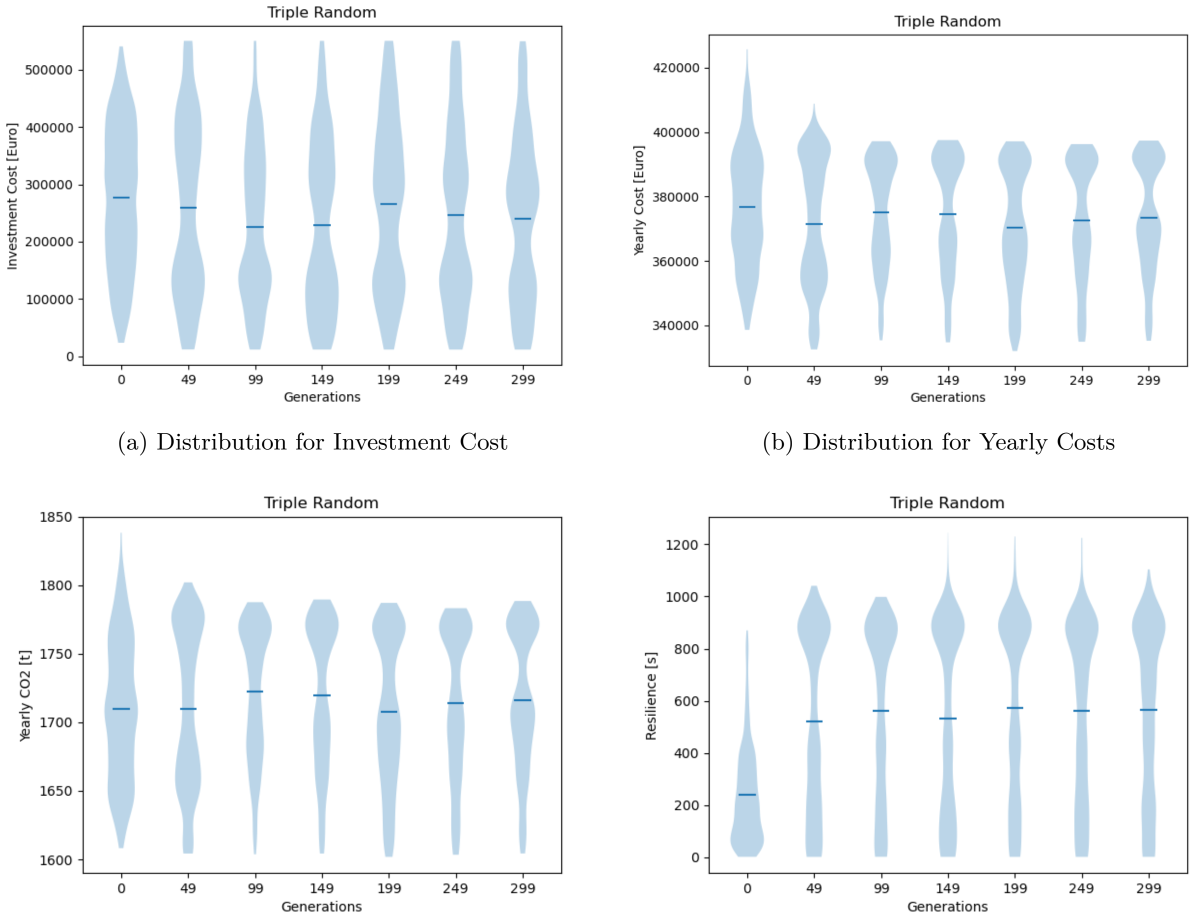
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure G.2: Double Random Objective Distributions



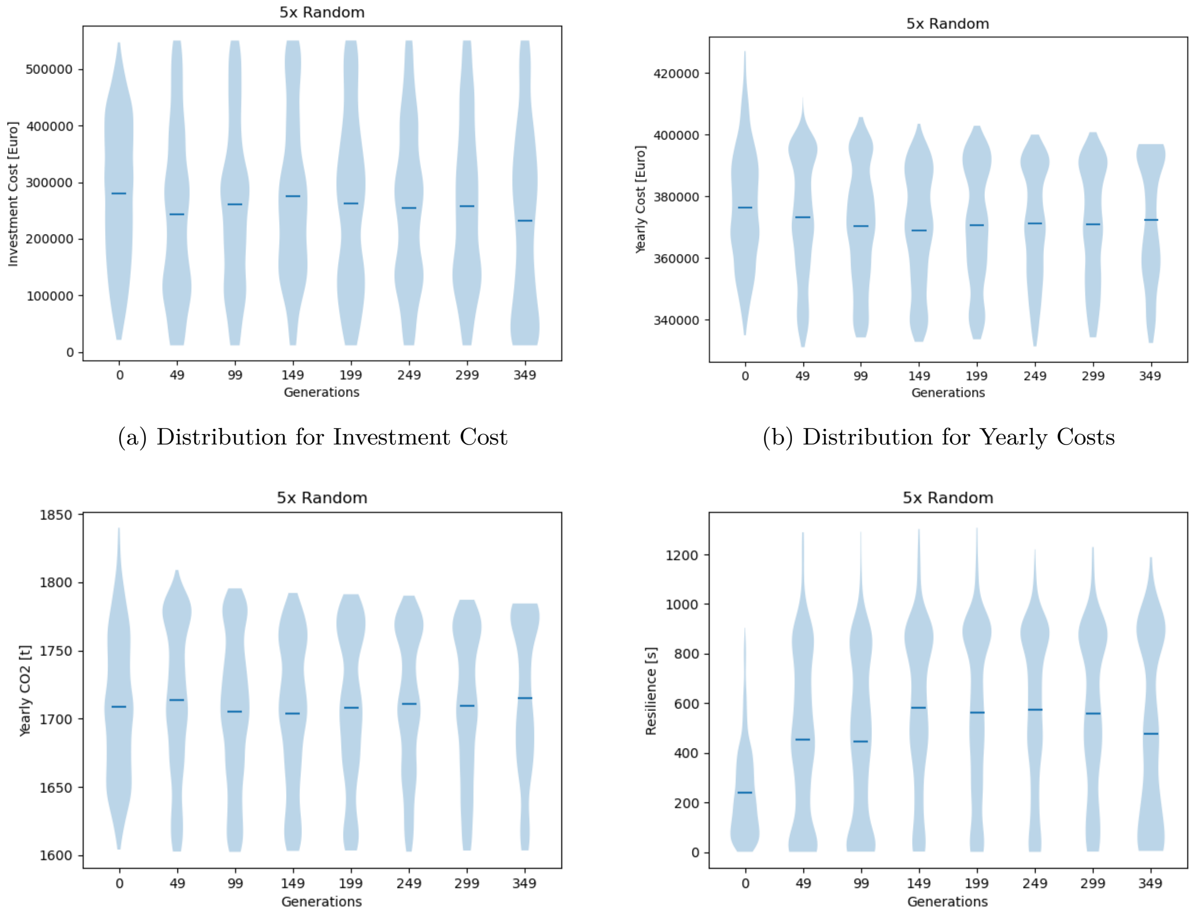
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure G.3: Triple Random Objective Distributions



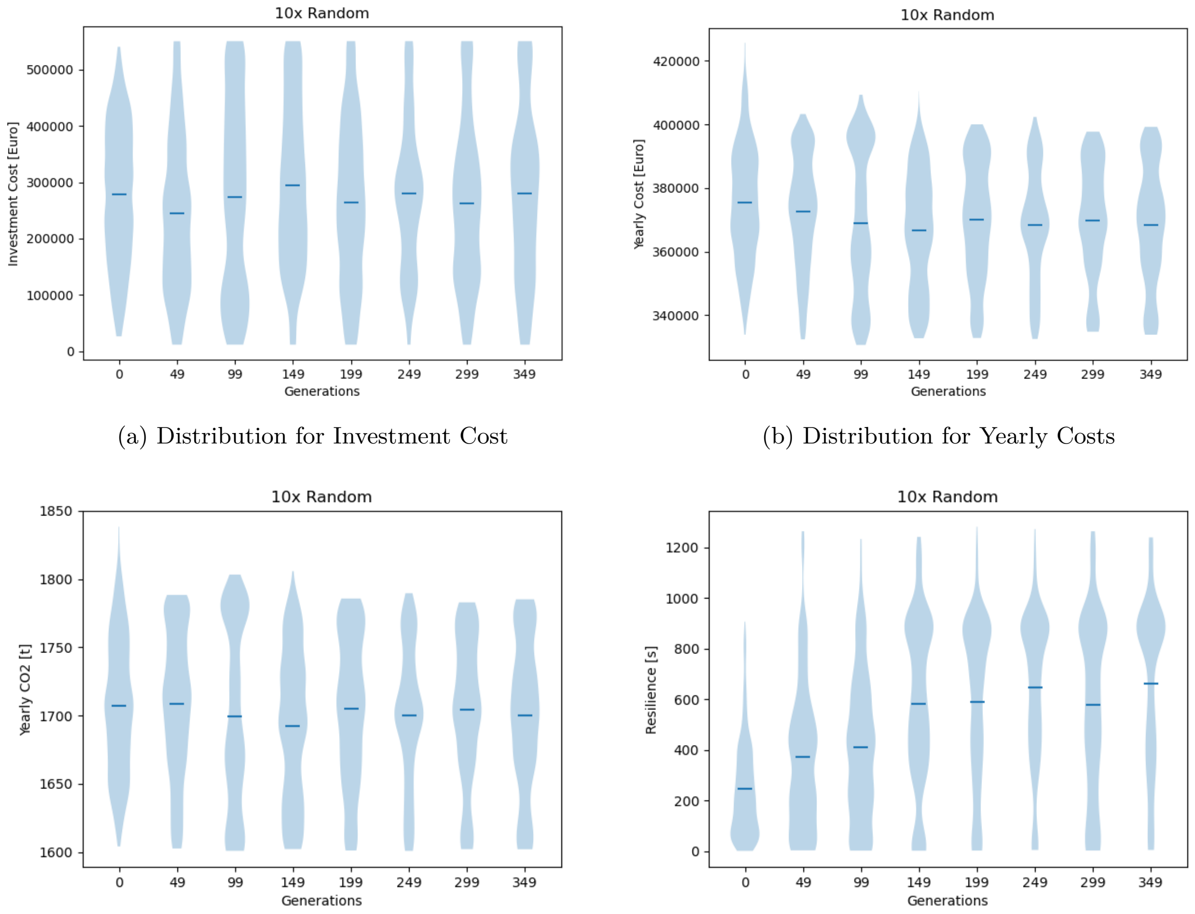
(a) Distribution for Investment Cost

(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure G.4: 5x Random Objective Distributions



(a) Distribution for Investment Cost

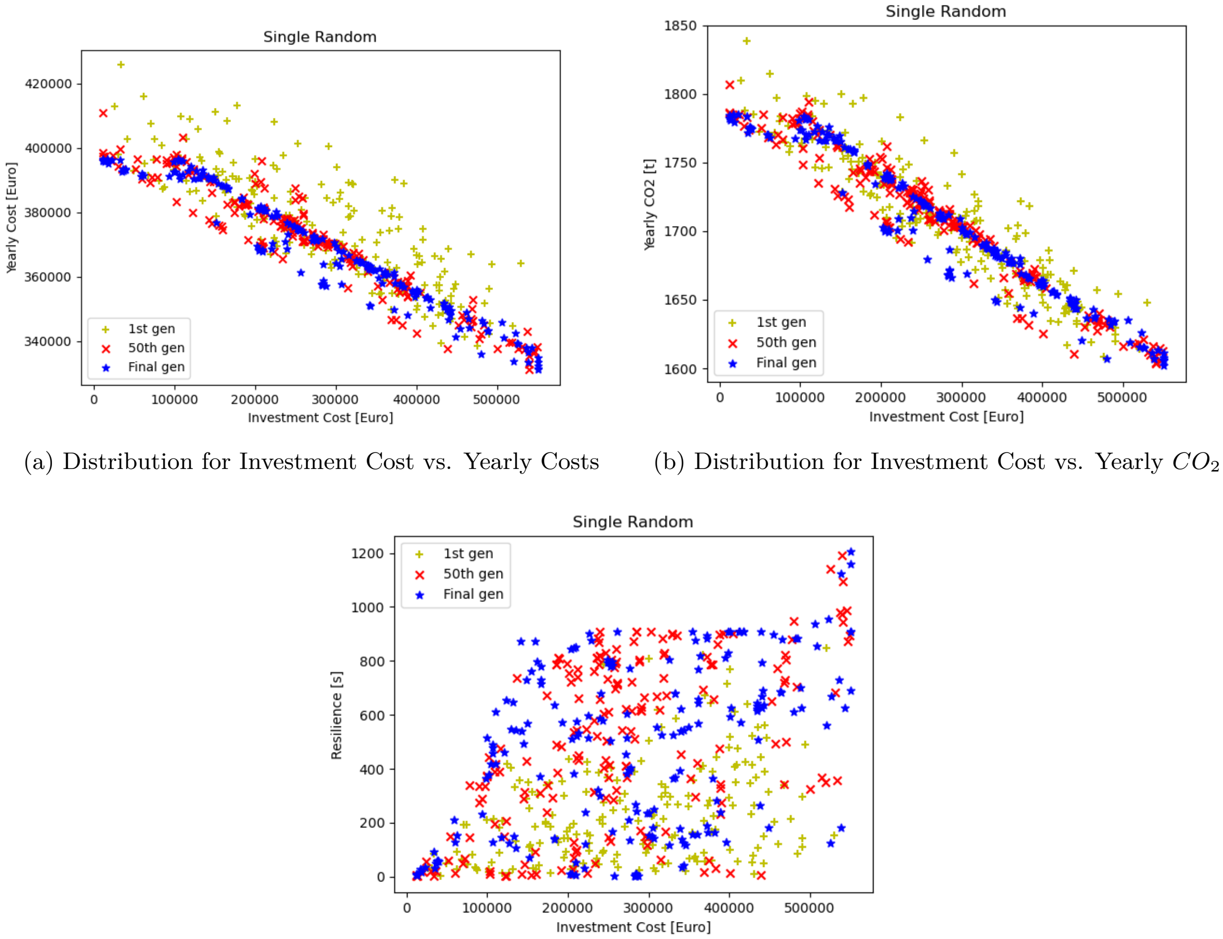
(b) Distribution for Yearly Costs

(c) Distribution for Yearly CO_2

(d) Distribution for Resilience

Figure G.5: 10x Random Objective Distributions

H - Random Objective Space

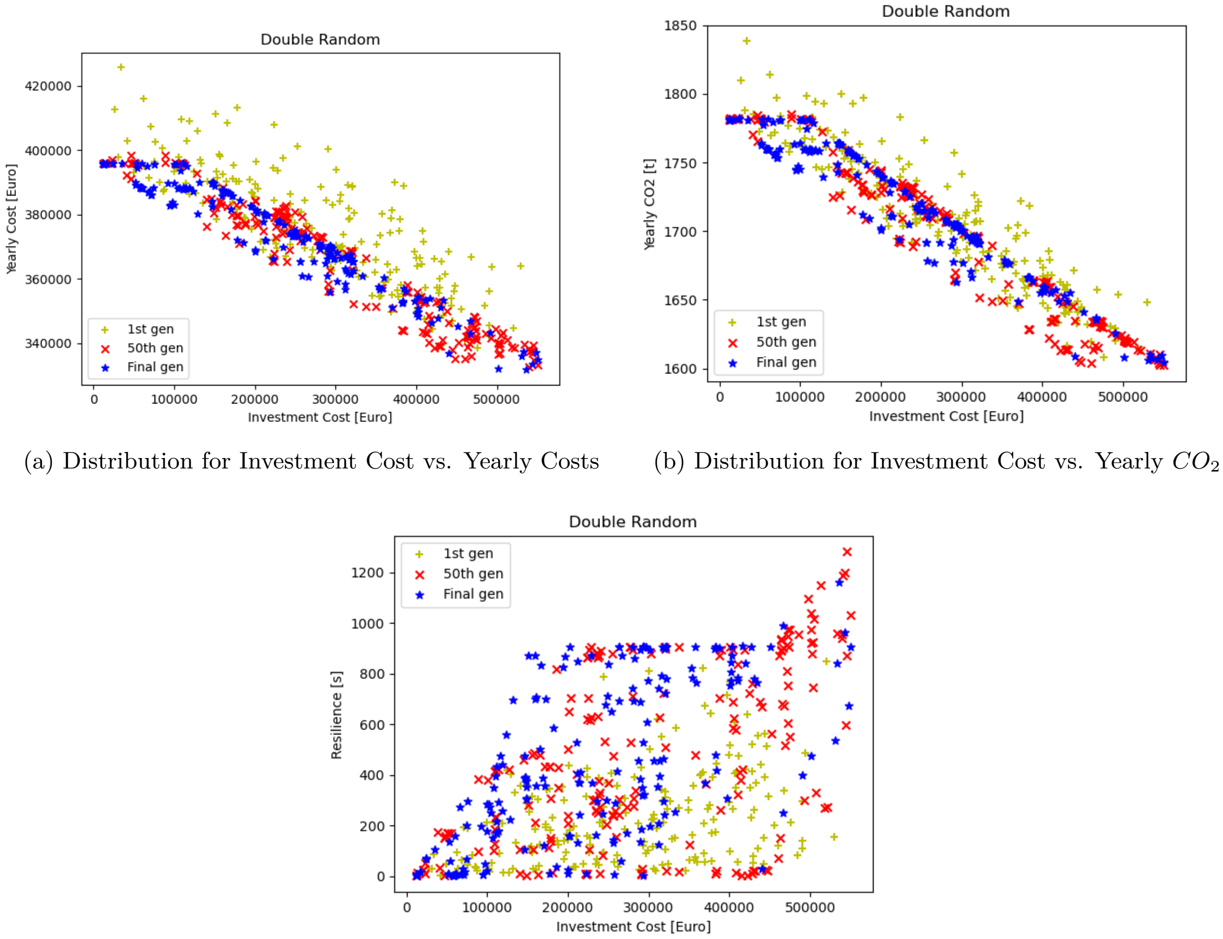


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure H.1: Single Random Objective Space

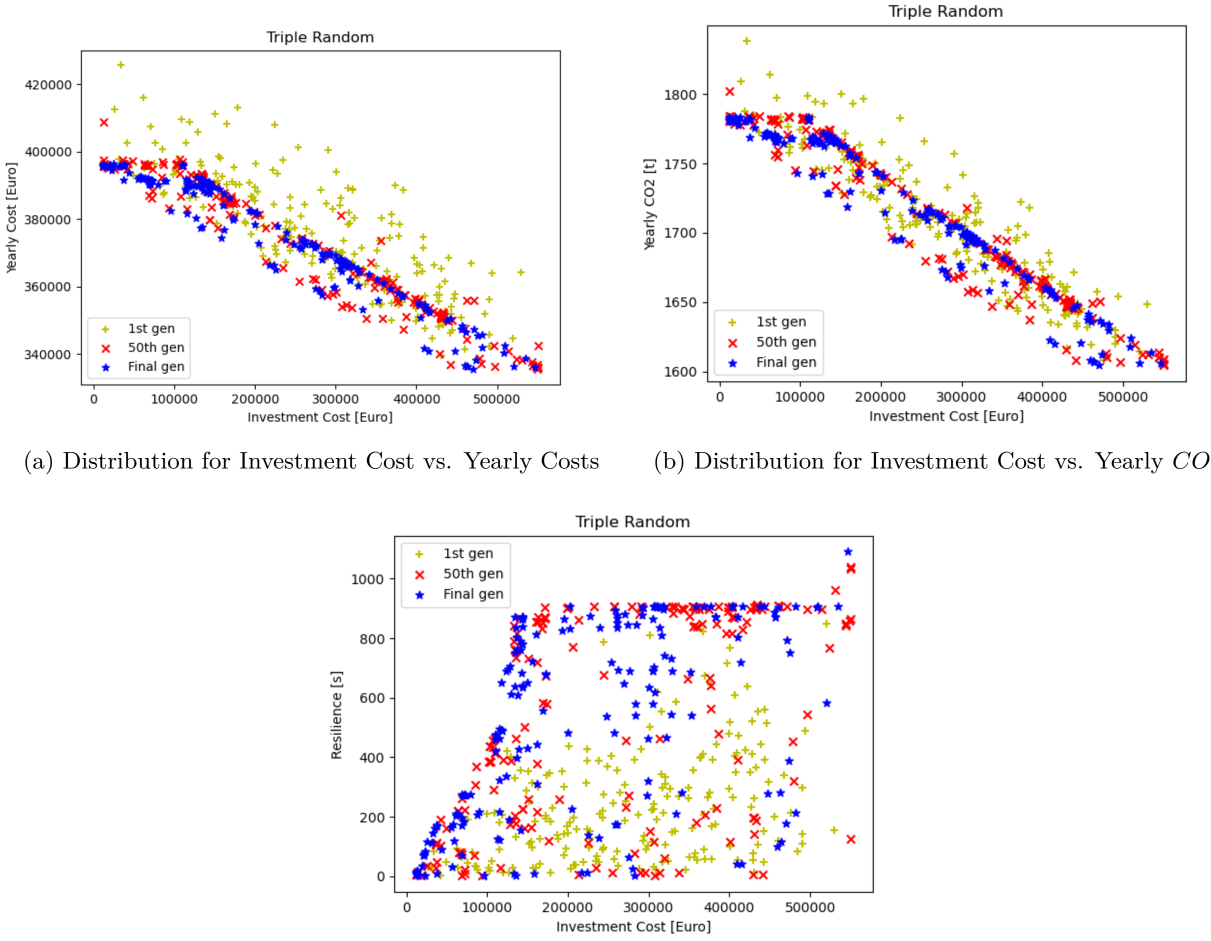


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure H.2: Double Random Objective Space

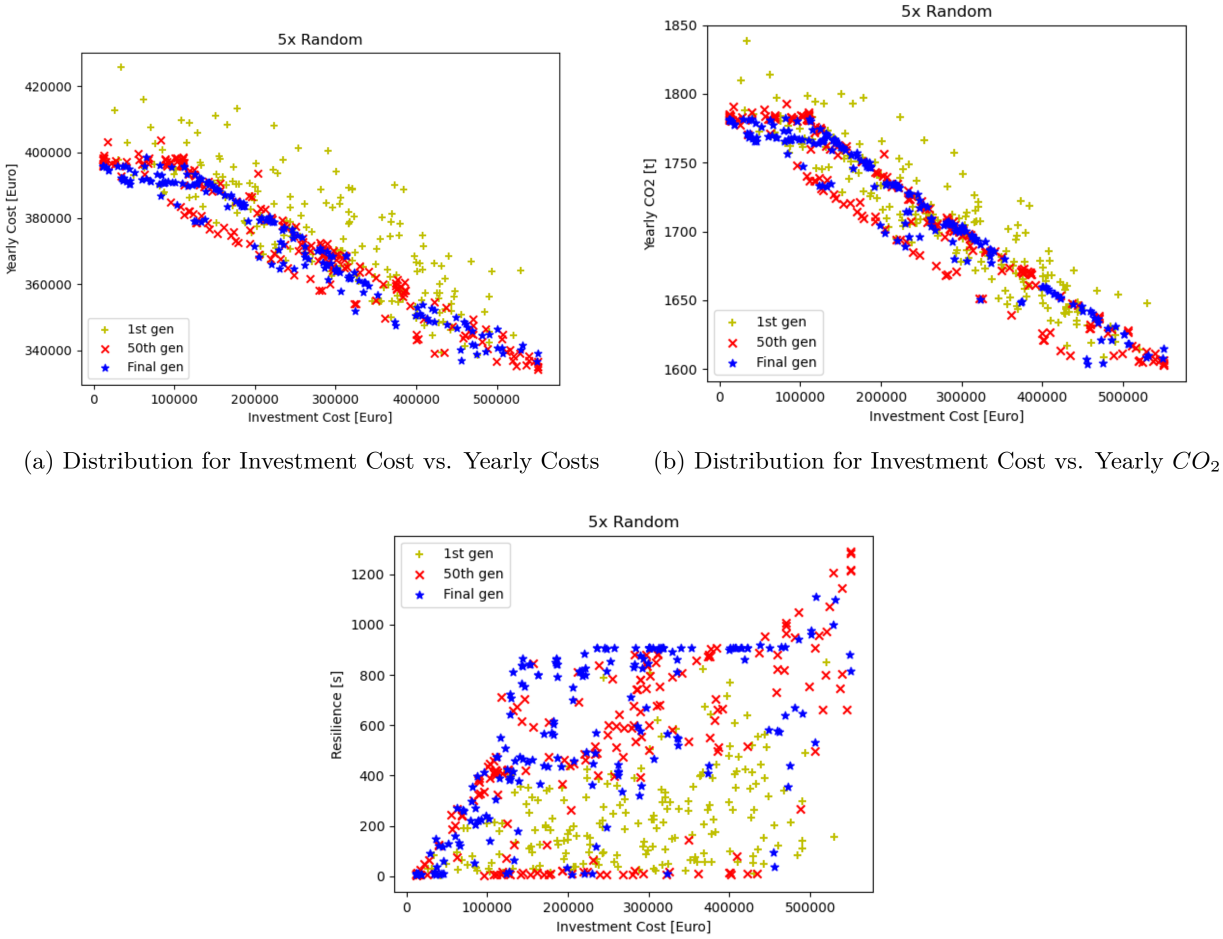


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure H.3: Triple Random Objective Space

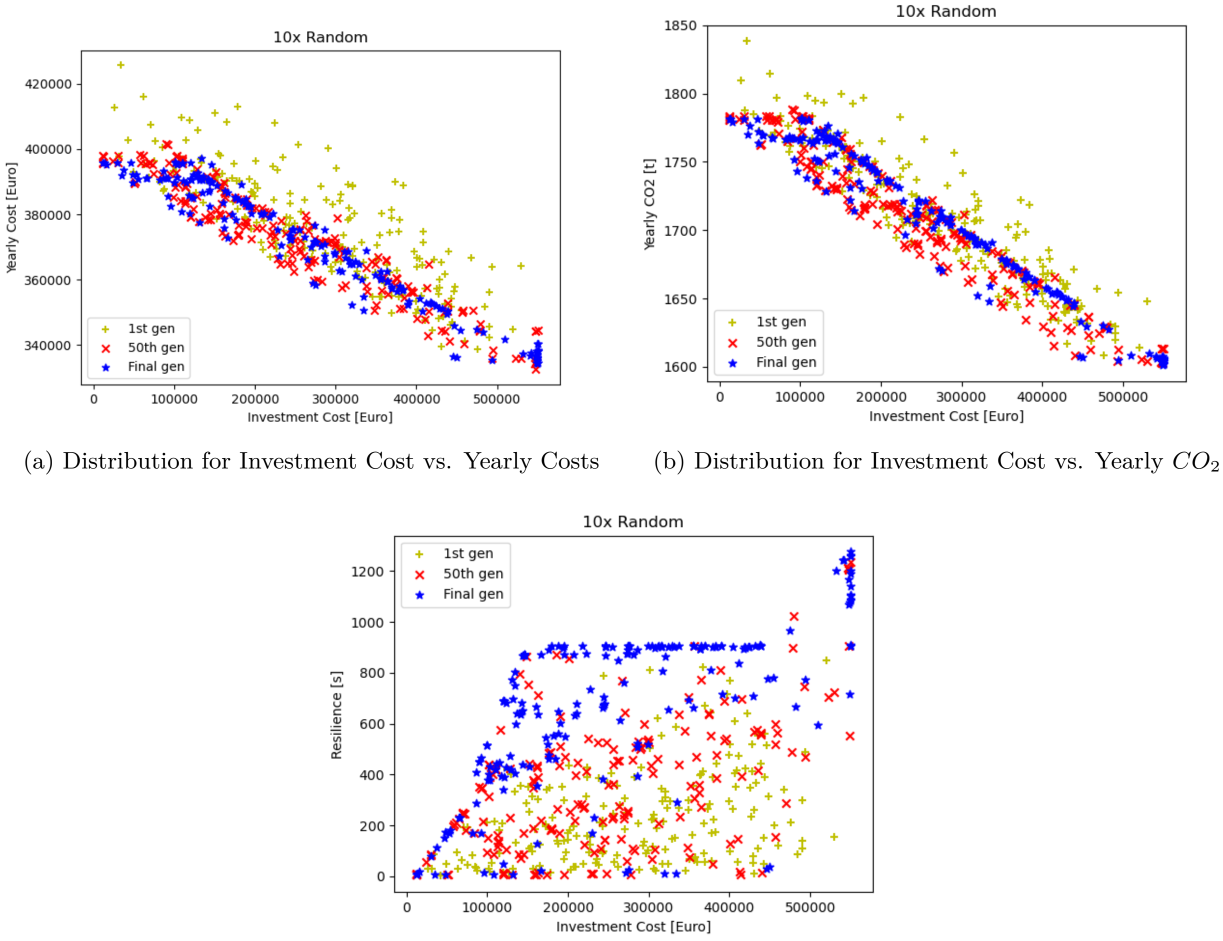


(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure H.4: 5x Random Objective Space



(a) Distribution for Investment Cost vs. Yearly Costs

(b) Distribution for Investment Cost vs. Yearly CO_2

(c) Distribution for Investment Cost vs. Resilience

Figure H.5: 10x Random Objective Space