Simulated Annealing optimization on DFs

Simulated annealing is a probabilistic technique used for approximating the global optimum of a given function. It is inspired by the annealing process in metallurgy, where material is heated and then slowly cooled to increase its strength and reduce defects. This report presents the results of simulated annealing optimization on 14 dynamic functions (DF1 to DF14) with 75 dimensions. Time of the functions was fixed to t=50.

The simulated annealing algorithm is implemented with the following parameters:

- Maximum iterations: 10 000
- Initial temperature: 100
- Cooling rate: 0.99
- Neighbor range: 0.1

The algorithm starts by initializing a solution within the bounds of the function. It then iteratively perturbs the current solution and decides to accept the new solution based on the criterion. The acceptance probability is determined by the difference in function values and the current temperature. The temperature is gradually decreased to make the algorithm more selective in accepting new solutions as it progresses.

The results of the optimization are shown below:

Function	Optimum Value
DF1	1.18
DF2	0.9
DF3	1.68
DF4	2.78
DF5	1.3
DF6	60.95
DF7	30.76
DF8	0.61
DF9	1.25
DF10	1.52
DF11	1.21
DF12	1.5
DF13	2.62
DF14	1.32

The results indicate that the simulated annealing algorithm can find solutions for most dynamic functions. However, the algorithm's performance may vary depending on the choice of parameters and the function's characteristics. For instance, the algorithm struggles to find a low-valued optimum for DF6 and DF7, suggesting that these functions may require a different approach or parameter tuning. The time it took for running some functions also differed, suggesting that some functions can be easier optimized than others.

In conclusion, simulated annealing is a powerful and versatile optimization technique that can be used to solve complex optimization problems, including those involving dynamic functions. This experiment shows the algorithm's potential but also highlights the need for careful parameter selection and problem-specific adjustments.

Genetic Algorithm

Genetic algorithms (GAs) are a type of evolutionary algorithm inspired by the principles of natural selection and genetics. They are used to find approximate solutions to optimization and search problems. This report presents the results of simulated annealing optimization on 14 dynamic functions (DF1 to DF14) with 75 dimensions. Time of the functions was fixed to t=50.

The genetic algorithm is implemented with the following parameters:

- Population size: 100
- Number of generations: 10 000
- Mutation rate: 0.01
- Tournament size: 5

The GA implementation consists of several key components:

1. **Generate Individual**: Creates a random individual within the problem's variable bounds.
2. **Compute Fitness**: Evaluates the fitness of an individual by summing up the evaluation results from the problem.
3. **Selection**: Uses tournament selection to choose parents for the next generation.
4. **Crossover**: Combines two parents to produce offspring using single-point crossover.
5. **Mutation**: Introduces random variations into an individual with a certain probability.
6. **Genetic Algorithm Loop**: Runs the GA over multiple generations, evolving the population towards better solutions.

The results of the optimization are shown below:

Function	Best Fitness Value
DF1	1.015
DF2	0.76
DF3	1.14
DF4	2.06
DF5	0.967
DF6	2.14
DF7	20.117

DF8	0.4
DF9	1.033
DF10	2.027
DF11	1.015
DF12	1.096
DF13	2.01
DF14	0.959

The GA demonstrated its capability to optimize a range of dynamic problems with varying degrees of success. While it performed well on some problems (e.g., DF8), it struggled with others (e.g., DF7). Further experimentation and hybrid approaches could help in obtaining better results across all problems.

The implemented Genetic Algorithm efficiently searches for optimal solutions by evolving a population of candidate solutions over multiple generations. The use of tournament selection, crossover, and mutation ensures diversity and convergence towards better solutions. The algorithm was applied to 14 dynamic optimization problems, with the best solutions recorded and saved for analysis.