Dokumentacja końcowa projekt 8

Kamil Bańkowski, Piotr Baranowski

June 2025

1 Opis problemu

Celem projektu było zbadanie wydajności i efektywności algorytmu CMA-ES w wersji podstawowej w porównaniu z modyfikacją polegającą na inicjalizacji p_σ wektorem losowym. Wersja standardowa została porównana z inicjalizacją wektorem losowym z wielowymiarowego rozkładu normalnego oraz rozkładu jednostajnego na $[-1.0, 1.0]^n$. W celu zbadania efektywności algorytmów wykorzystano funkcje testowe: Ackleya, Bohachevskiego, Griewanka, Rastrigina, Schwefla, elipsoidalną, kwadratową oraz HappyCat.

2 Rozwiązanie

W ramach projektu zaimplementowany został algorytm CMA-ES wraz z modyfikacjami i testami w języku C++ z wykorzystaniem biblioteki Eigen 3.3. Analiza wyników oraz tworzenie wykresów zostały przeprowadzone w języku Python 3.10 z wykorzystaniem następujących bibliotek:

1. pandas – wczytywanie danych w formacie CSV,
2. numpy – wyznaczanie wariancji,
3. scipy.stats – implementacje testów Shapiro-Wilka, Levene'a oraz ANOVA,
4. statsmodels.stats.multicomp – implementacja testu HSD Tukeya,
5. scikit_posthocs – implementacja testu Dunna,
6. matplotlib.pyplot – tworzenie wykresów.

Proces budowania odbywa się z wykorzystaniem narzędzia CMake. Dodatkowo, aby uprościć złożoność komend niezbędnych do konfiguracji, budowania, testowania oraz analizy wyników, funkcjonalności te zostały wyabstrahowane jako cele (targets) narzędzia Make. Szczegóły znajdują się w pliku README.md.

W celu zapewnienia przenaszalności przygotowano plik Dockerfile, umożliwiający zbudowanie obrazu Dockerowego zawierającego wszystkie niezbędne zależności do odtworzenia rozwiązania.

3 Testy

Efektywność algorytmów została poprównana na podstawie zwracanych średnich wartości funkcji poddawanej optymalizacji oraz krzywych zbieżności. W tym celu wykonane zostały dwa rodzaje pomiarów.

W przypadku pierwszego dla każdej funkcji testowej oraz każdego rozważanego wymiaru losowany był punkt startowy. Następnie dla każdej wersji algorytmu CMA-ES przeprowadzona została próba 20-stu uruchomień z tym samym punktem początkowym. Tak zebrane wyniki zapisywane były w formacie `csv` w pliku `test/<ziarno>/compare-<nazwa-funkcji>-<wymiar>`. Następnie na podstawie tych danych przeprowadzone zostały testy sprawdzające równość średnich/median zwracanych wyników między badanymi algorytmami. Wpierw jednak przeprowadzane byly testy normalności rozkładów Shapiro-Wilka oraz równości wariancji Levene'a. Jeśli otrzymywane rozkłady pochodziły z rozkładu normalnego i ich wariancje były równe (spełaniały założenia testu ANOVA) weryfikacja równości średnich przeprowadzana była na podstawie testu ANOVA. W przeciwnym wypadku, o ile wariancja rozkładu nie była równa 0 przeprowadzany był test Kruskala-Wallisa. Jeśli z testu ANOVA wynikało, że przynajmniej jedna średnia jest różna od pozostałych, to wykonywany był test HSD Tukeya do porównania średnich parami. W analogicznej sytuacji, w przypadku gdy z testu Kruskala-Wallisa wynikało, że przynajmniej jedna mediana jest znacząco różna od pozostałych, wykonywany był test Dunna z poprawką Bonferroniego. Wyniki tych testów zapisywane były w `compare/<ziarno>/compare-<nazwa-funkcji>-<wymiar>-statistics`. Dodatkowo, w celu wizualizacji badanych rozkładów zapisywane były wykresy skrzynkowe badanych rozkładów w pliku `compare/<ziarno>/compare-<nazwa-funkcji>-<wymiar>-plot.png`.

Drugi rodzaj pomiarów polegał również na wylosowaniu jednego punktu startowego dla każdej funkcji testowej i wymiaru. Natomiast, w odróżnieniu do pierwszego rodzaju, każdy z algorytmów były uruchamiany tylko raz z tym samym punktem początkowym, a zapisywanymi danymi były wartości funkcji celu w kolejnych iteracjach działania algorytmów. Zebrane pomiary zapisywane były w pliku `test/<ziarno>/convergence-<nazwa-funkcji>-<wymiar>`. Następnie w pliku `convergence/<ziarno>/convergence-<nazwa-funkcji>-<wymiar>-plot.png` generowane i zapisywane były wykresy zbieżności na podstawie zebranych danych.

W ramach testów użyto funkcji i obszarów losowania punktów początkowych przedstawionych w tabeli 1 oraz parametrów przedstawionych w tabeli 2. Wybór wartości parametrów odbył się na podstawie artykułu [3].

Zarówno w przypadku pierwszego jak i drugiego rodzaju pomiarów badane wymiary wynosiły kolejno $1, 5, 10, 20$ oraz 50.

Nazwa funkcji celu	**Obszar inicjalizacji punktu początkowego**
Ackley	$[1.0,\ 30.0]^n$
Bohachevsky	$[1.0,\ 15.0]^n$
Griewank	$[10.0,\ 600.0]^n$
Rastrigin	$[1.0,\ 5.0]^n$
Schwefel	$[-500.0,\ 300.0]^n$
Ellipsoid	$[-100.0,\ 100.0]^n$
Sphere	$[-100.0,\ 100.0]^n$
HappyCat	$[-40.0,\ 100.0]^n$

Tabela 1: Funkcje testowe i obszary inicjalizacji punktu początkowego użyte w testach. Wzory na wartości funkcji Ackleya, Bohachevskiego, Griewanka, Rastrigina oraz Schwefela znajdują się w artykule [2]. Formułę na wartość funkcji eliptycznej można znaleźć w pracy [4] (zaimplementowana wersja funkcji eliptycznej w odróżnieniu od oryginału jest przeskalowana przez $10^{\frac{6}{N}}$). Natomiast wzór na wartość funkcji kwadratowej oraz HappyCat można znaleźć w kolejno artykule [3] oraz pracy [1].

Nazwa parametru	**Wartość (początkowa)**
λ	$4 + \lfloor 3 \cdot \log(N) \rfloor$
μ	$\lfloor \frac{\lambda}{2} \rfloor$
N c_1	$\frac{2}{(N+1.3)^2 + \mu}$
c_s	$\frac{\mu + 2}{N + \mu + 5}$
c_μ	$\min\left(1 - c_1, \frac{2(\mu - 2 + \frac{1}{\mu})}{(N+2)^2 + \mu}\right)$
d_s	$1 + c_s + 2 \cdot \max\left(0, \sqrt{\frac{\mu - 1}{N+1}} - 1\right)$
σ	$0.3 \cdot (r_{\text{upper}} - r_{\text{lower}})$
c_c	$\frac{2}{N+1}$
d_1	10^{-8}
d_2	10^{-10}
`max_iter`	$\lfloor 1000 \log(N+1) \rfloor$

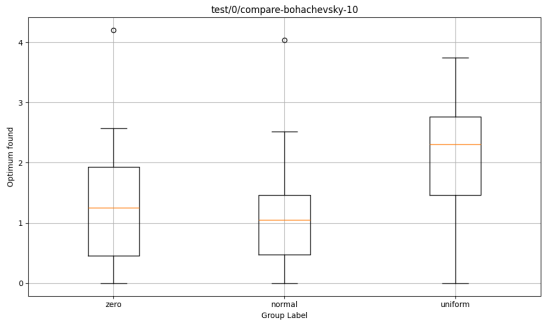
Tabela 2: Parametry użyte do testowania algorytmu CMA-ES i jego modyfikacji. Użyte we wzorze N oznacza wymiar przestrzeni przeszukiwań, natomiast $[r_{\text{upper}}, r_{\text{lower}}]^n$ jest obszarem inicjalizacji punktu startowego dla danej funkcji testowej. Współczynniki d_1, d_2 są współczynnikami warunku Gilla zakończenia obliczeń, natomiast `max_iter` jest maksymalną liczbą generacji algorytmu CMA-ES.

4 Wyniki

W bieżącej sekcji przedstawione zostały wyniki z przeprowadzonych testów i pomiarów.

4.1 Porównanie jakości

W znakomitej większości przypadków test ANOVA lub test Kruskala-Wallisa nie potwierdził fałszywości hipotezy mówiącej o równości średnich/median na poziomie istotności 0.05. Wartości p-value w zależności od testu wahały się między 0.0553, a 0.9519. Odstępstwem od tej tendencji jest test z 10-cio wymiarową funkcją Bohachevskiego, dla której test Kruskala-Wallisa wykazał, że istnieje istotna różnica między miedianami zwracanych wyników w zależności od wyboru algorytmu. Wartość p-value dla tego testu wyniosła 0.0132 oraz istonie gorszą modyfikacją w porównaniu do pozostałych metod okazała się ta z rozkładem jednostajnym (na postawie testu Dunna). Poddane badaniu rozkłady można zobaczyć na rysunku 1. Dodatkowo poziomy istotności porównywanych średnich w ramach testu Dunna przedstawione zostały w tabeli 3.



Rysunek 1: Wykresy skrzynkowe dla badanych wariantów algorytmu CMA-ES dla 10-cio wymiarowej funkcji Bohachevskiego.

Tabela 3: Wartości p-value testu Dunna dla testu z 10-cio wymiarową funkcją Bohachevskiego.

	Normal	**Uniform**	**Zero**
Normal	1.0000	0.0133	1.0000
Uniform	0.0133	1.0000	0.1148
Zero	1.0000	0.1148	1.0000

Wartym wspomnienia testem jest funkcja Bohachevskiego, w ramach którego każde uruchomienie zwróciło optimum globalne równe 0. Oczywiście w jego przypadku średnie rozkładów są takie same, natomiast nie możliwy był do przeprowadzenia zarówno test ANOVA, jak i test Kruskala-Wallisa.

Wykresy skrzynkowe oraz wyniki dla pozostałych testów można wygenerować poleceniem w głównym katalogu projektu, a otrzymane wyniki zostaną zapisane w katalogu `comparison/0/`.

```
make clean config build test comparison
```

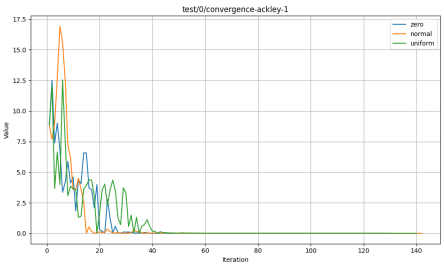
4.2 Wykresy zbieżności

W bieżącej sekcji na rysunkach od 2 do 9 przedstawione zostały wykresy zbieżności dla wybranych funkcji testowych.

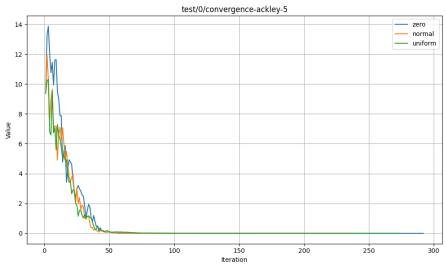
W przypadku funkcji Ackleya można zaobserwować, że wersji podstawowej algorytmu przy każdej próbie udało się znaleźć minimum globalne natomiast w przypadku modyfikacji znalezione rozwiązanie było tylko minimum lokalnym. Można również zauważyć, że badane warianty mają podobne tempo zbieżności, a znakomitą większość czasu algorytm spędza na stabilizacji. Analogiczne obserwacje tyczą się funkcji Bohachevskiego, dla której w przypadku jednowymiarowym faza eksploatacji zajęła cały czas działania algorytmów. Niekiedy jednak CMA-ES jest w stanie szybciej się ustabilizować, co można zaobserwować w przypadku 5-cio wymiarowym, jednak warto tu zaznaczyć, że znalezione rozwiązanie nie jest optimum globalnym. Podobną zależność mozna zauważyć w przypadku jednowymiarowej funkcji Griewanka. Natomiast odwrotne zachowanie obserwowane jest w przypadku 5-cio wymiarowej funkcji Rastrigina, dla której tempo zbieżności jest tym dłuższe im gorsze optimum jest eksploatowane.

Szczególnie ciekawymi wykresami zbieżności niezależnie od wymiaru są te wykonane dla uruchomień algorytmów dla funkcji Schwefela. W jej przypadku znajdowane optima istotnie różnią się od siebie, a czas działania algorytmów znacząco różni się. Wynika to jednak z faktu, że funkcja ta w $\mathbb{R}^n$ nie jest ograniczona z dołu zatem znajdowane optima poza obszarem inicjalizacji punktu startowego mogą się istotnie różnić.

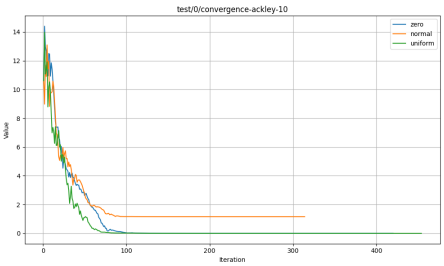
Zbieżność badanych algorytmów w przypadku funkcji elipsoidalnej, kwadratowej oraz HappyCat nie różni się znacząco między wariantami algorytmu CMA-ES. Owszem czas trwania eksploatacji jest inny w zależności od uruchomionego wariantu, ale fazy eksploracji pod względem liczby generacji nie są istotnie różne.



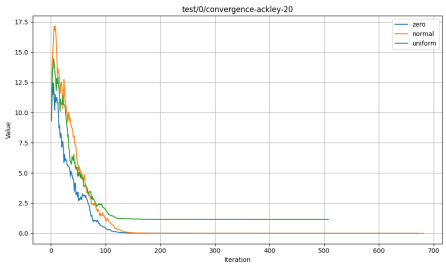
(a) $N = 1$



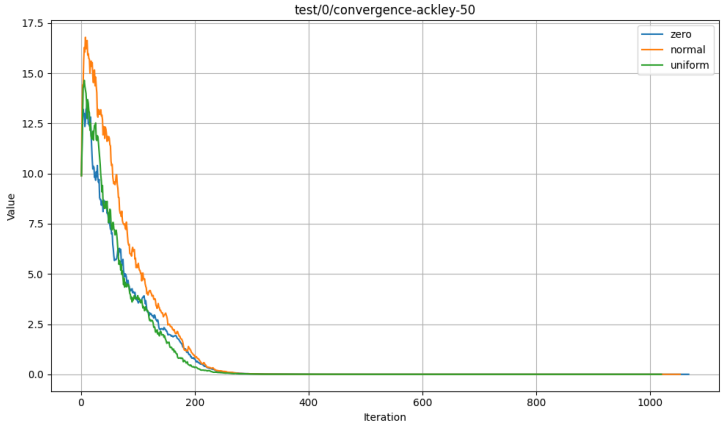
(b) $N = 5$



(c) $N = 10$

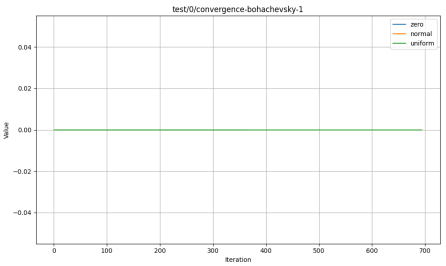


(d) $N = 20$

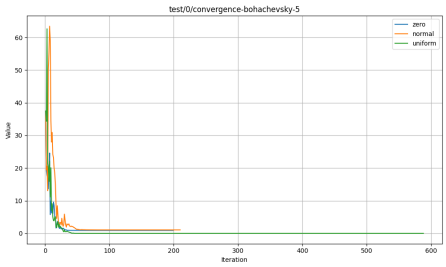


(e) $N = 50$

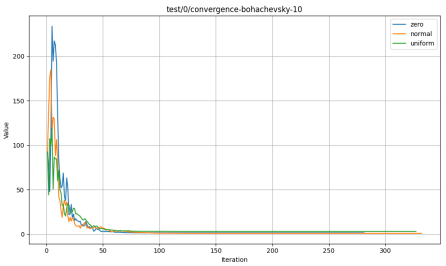
Rysunek 2: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji Ackleya o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



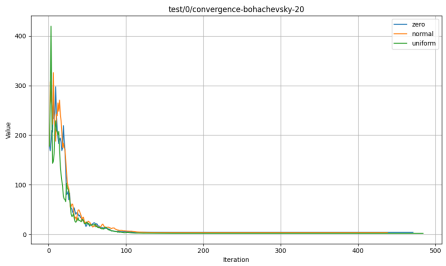
(a) $N = 1$



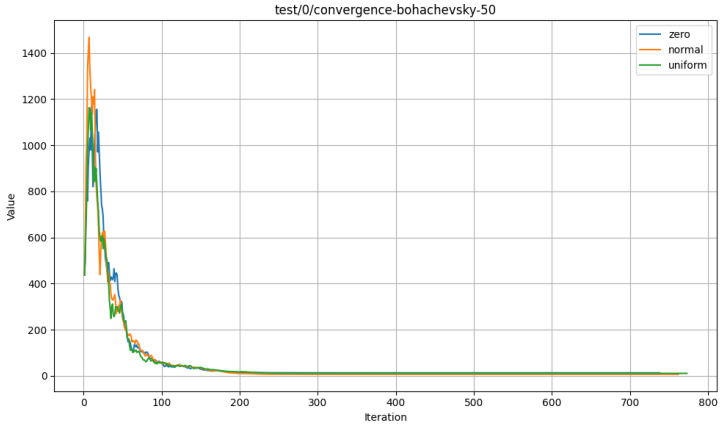
(b) $N = 5$



(c) $N = 10$

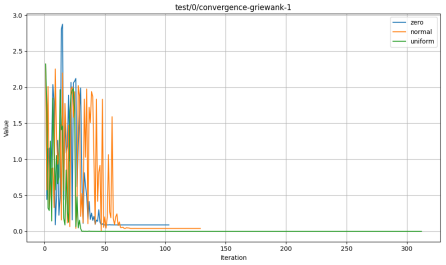


(d) $N = 20$

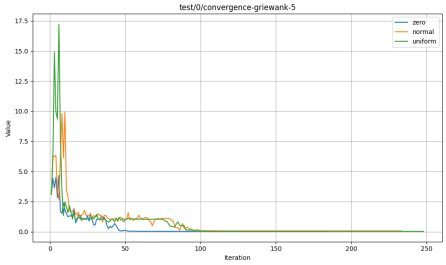


(e) $N = 50$

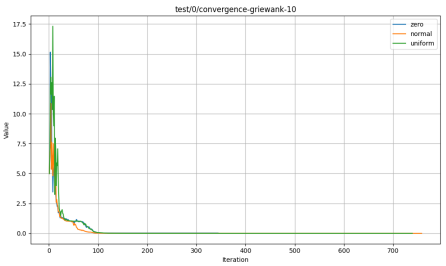
Rysunek 3: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji Bohachevskiego o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



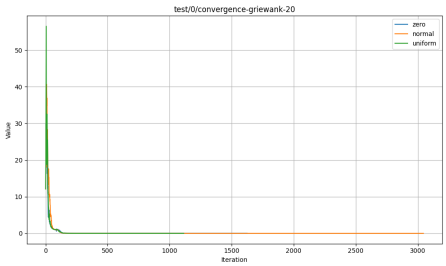
(a) $N = 1$



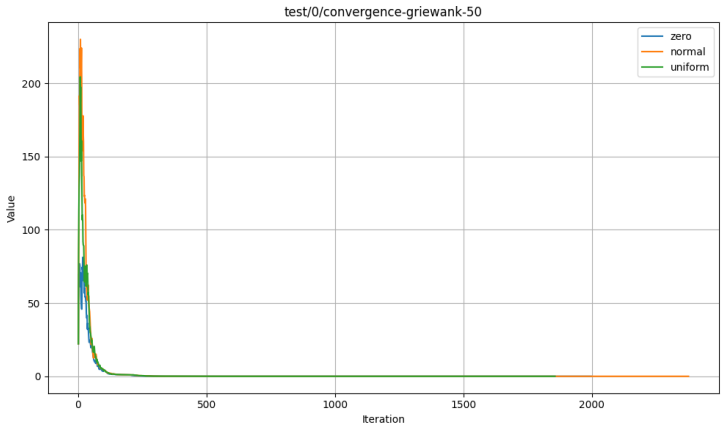
(b) $N = 5$



(c) $N = 10$

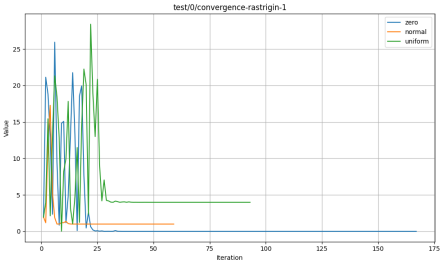


(d) $N = 20$

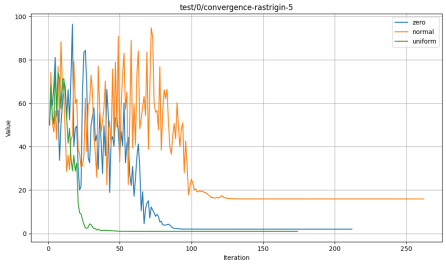


(e) $N = 50$

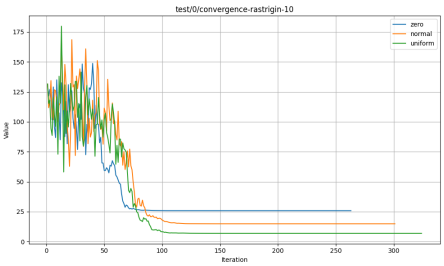
Rysunek 4: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji Griewanka o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



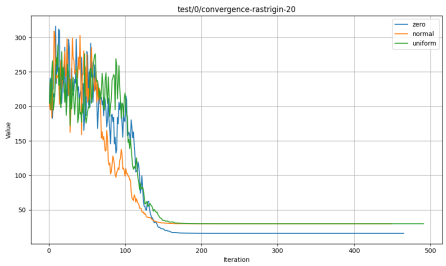
(a) $N = 1$



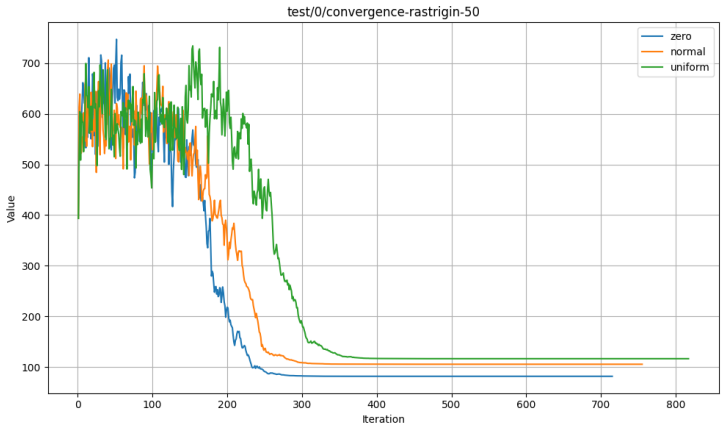
(b) $N = 5$



(c) $N = 10$

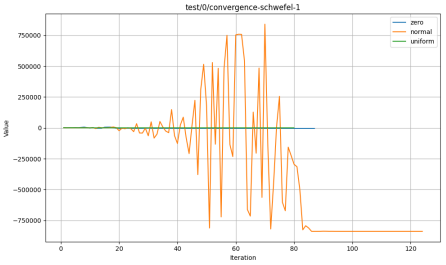


(d) $N = 20$

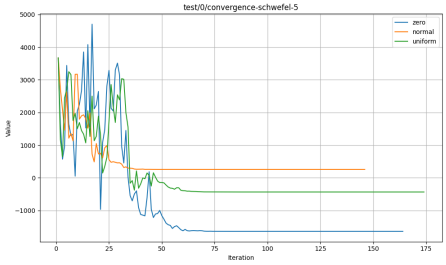


(e) $N = 50$

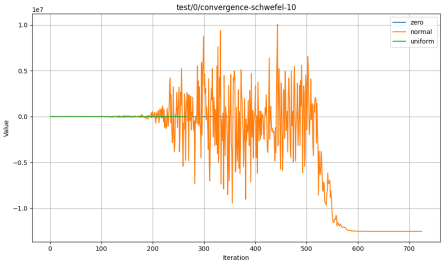
Rysunek 5: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji Rastrigina o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



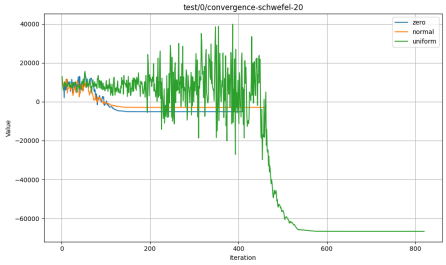
(a) $N = 1$



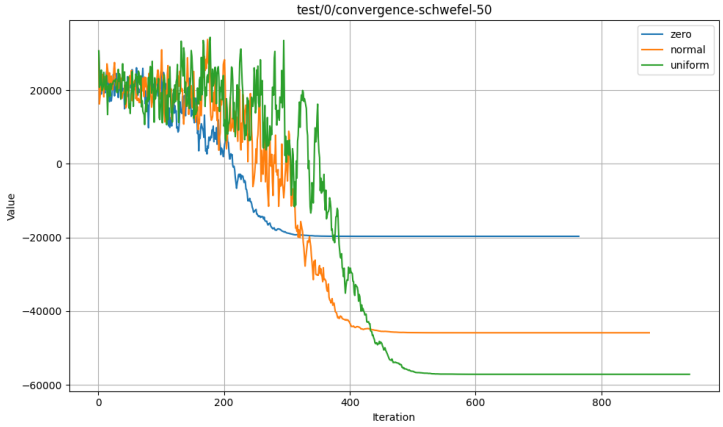
(b) $N = 5$



(c) $N = 10$

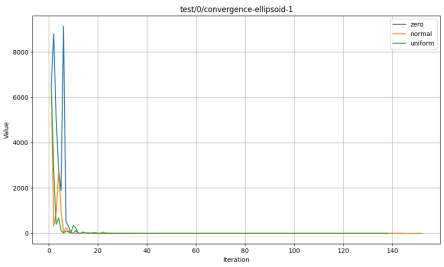


(d) $N = 20$

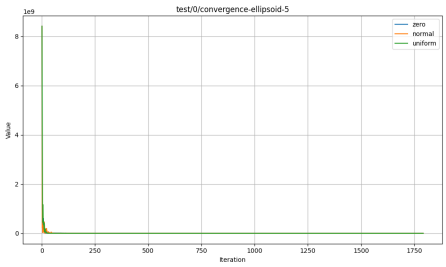


(e) $N = 50$

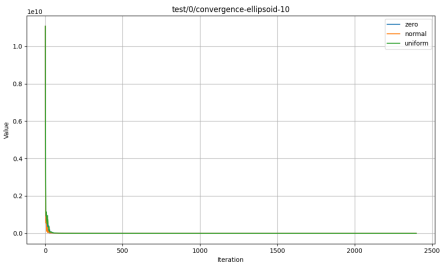
Rysunek 6: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji Schwefela o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



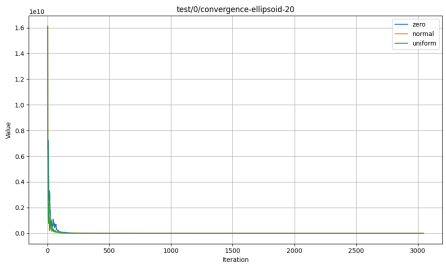
(a) $N = 1$



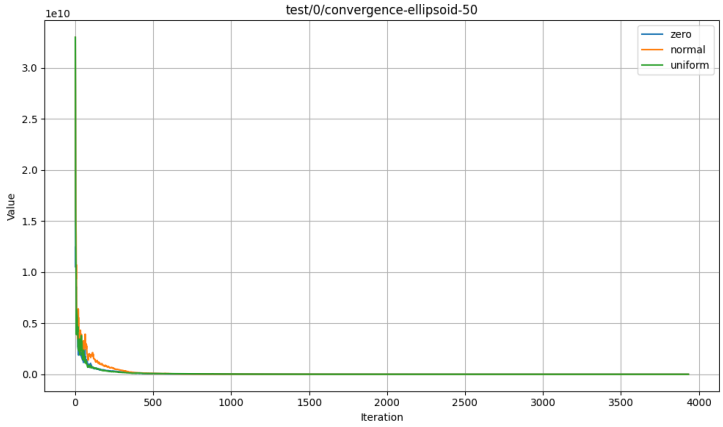
(b) $N = 5$



(c) $N = 10$

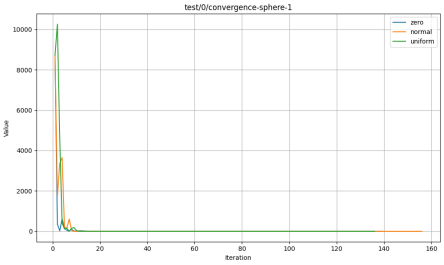


(d) $N = 20$

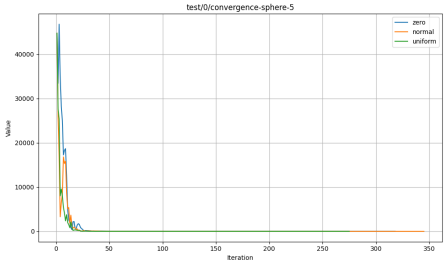


(e) $N = 50$

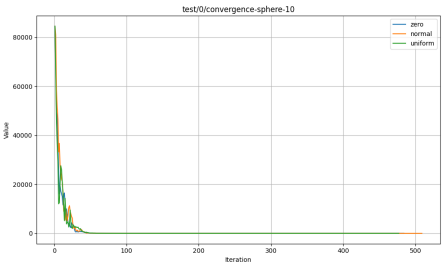
Rysunek 7: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji elispoidalnej o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



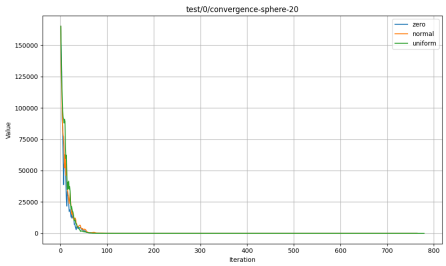
(a) $N = 1$



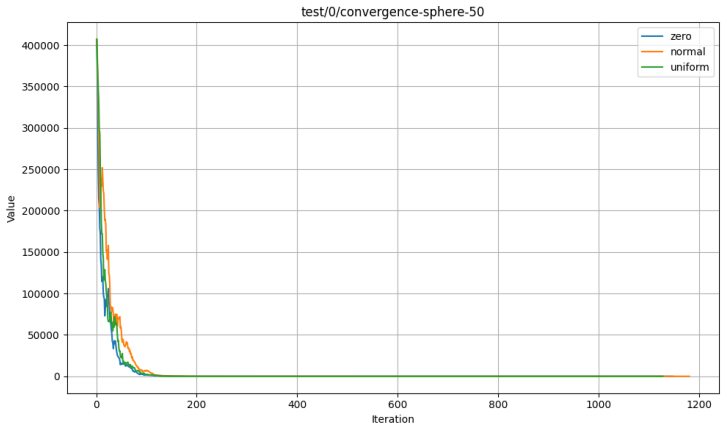
(b) $N = 5$



(c) $N = 10$

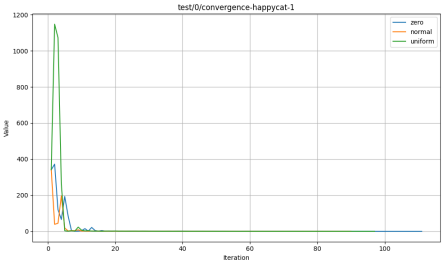


(d) $N = 20$

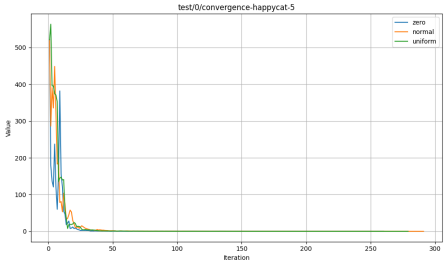


(e) $N = 50$

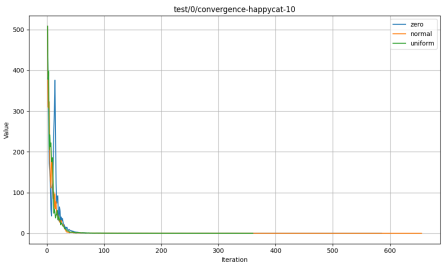
Rysunek 8: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji kwadratowej o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.



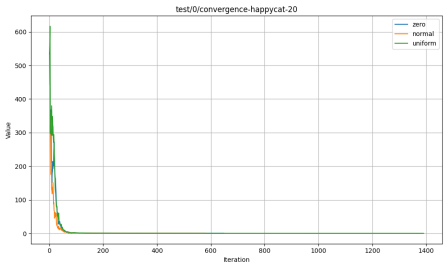
(a) $N = 1$



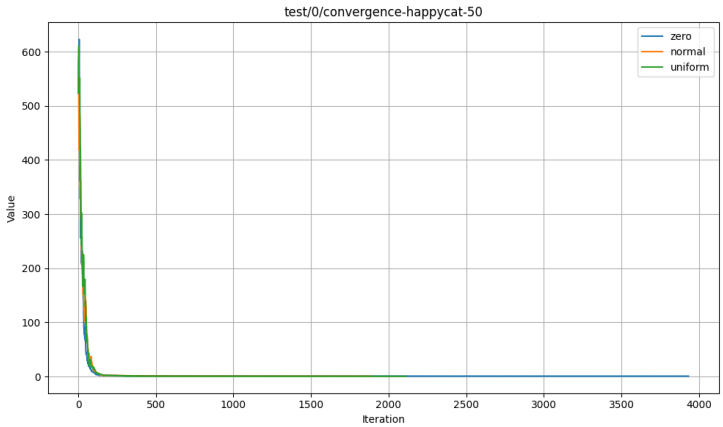
(b) $N = 5$



(c) $N = 10$



(d) $N = 20$



(e) $N = 50$

Rysunek 9: Wykresy zbieżności standardowej wersji algorytmu CMA-ES zaznaczonej na niebiesko, modyfikacji z rozkładem normalnym zaznaczonym na pomarańczowo oraz rozkładem jednostajnym oznaczonym kolorem zielonym. Wykresy przedstawiają przebieg kolejnych epok dla funkcji HappyCat o wymiarze równym (a) 1, (b) 5, (c) 10, (d) 20, (e) 50.

5 Wnioski

Rozważone modyfikacje nie wpływają na średnią jakość algorytmu CMA-ES. Przy pojedynczym uruchomieniu w zadanym punkcie startowym w zależności od metody możemy otrzymać istotnie różne wyniki, co można szczególnie zaobserwować w przypadku funkcji Schwefela, natomiast żadna z wersji algorytmu nie dominuje pod względem jakości rozwiązań. Pojedynczy test z 10-cio wymiarową funkcją Bohachevskiego może składniać ku dalszym badaniom. Natomiast na podstawie tego wyniku można sądzić, że rozważane modyfikacje mogą pogarszać efektywność algorytmu CMA-ES. W połączeniu z większym kosztem obliczeniowym w trakcie inicjalizacji prowadzi to do konkluzji, że inicjalizacja wektora p_σ wektorem zerowym jest najlepszym wariantem.

6 Różnice względem dokumentacji wstępnej

W stosunku do wstępnej specyfikacji projektu zdecydowano się na następujące zmiany.

1. Zmiana biblioteki numerycznej z Boost na Eigen3.3.
2. Rezygnacja z pomiaru i analizy czasu działania algorytmów na rzecz wykresów zbieżności.
3. Zmiana struktury projektu. Podział implementacji na algorytm i testy jest tylko na poziomie systemu plików. Algorytm oraz testy zaimplementowane zostały w osobnych plikach natomiast kompilowane są do jednego pliku wykonywalnego.
4. Rozszerzenie zbioru funkcji testowych o funkcje Schwefela, elipsoidalną, kwadratową oraz HappyCat.
5. Zmiana badanego rozkładu wektora losowego z jednostajnego na hipersześcianu jednostkowego $[0.0, 1.0]^n$, na rozkład jednostajny na hipersześcianie $[-1.0, 1.0]^n$.
6. Wyznaczanie pierwiastka macierzy kowariancji odbywa się za pomocą standardowego algorytmu do wyznaczania rozkładu macierzy na wartości własne, zamiast ortogonalizacji Gramma-Schmidta.

Bibliografia

[1] Hans-Georg Beyer i Steffen Finck. "Happycat–a simple function class where well-known direct search algorithms do fail". W: *International conference on parallel problem solving from nature*. Springer. 2012, s. 367–376.

[2] Nikolaus Hansen i Stefan Kern. "Evaluating the CMA evolution strategy on multimodal test functions". W: *International conference on parallel problem solving from nature*. Springer. 2004, s. 282–291.

[3] Nikolaus Hansen i Andreas Ostermeier. "Completely derandomized self-adaptation in evolution strategies". W: *Evolutionary computation* 9.2 (2001), s. 159–195.

[4] Tristan Marty i in. "LB+ IC-CMA-ES: Two Simple Modifications of CMA-ES to Handle Mixed-Integer Problems". W: *International Conference on Parallel Problem Solving from Nature*. Springer. 2024, s. 284–299.