



POLITECHNIKA WARSZAWSKA

WYDZIAŁ MATEMATYKI  
I NAUK INFORMACYJNYCH



PRACA DYPLOMOWA MAGISTERSKA

**ANALIZA MOŻLIWOŚCI WYKORZYSTANIA  
W ALGORYTMIE CMA-ES WIEDZY  
O OGRANICZENIACH KOSTKOWYCH**

AUTOR:

INŻ. ROBERT JAKUBOWSKI

PROMOTOR:

DR HAB. INŻ. JAROSŁAW ARABAS  
PROF. NZW. PW

WARSZAWA MAJ 2016

.....

podpis promotora

.....

podpis autora

# Spis treści

<b>1</b>	<b>Streszczenie</b>	<b>5</b>
<b>2</b>	<b>Wstęp</b>	<b>6</b>
2.1	Cel pracy . . . . .	6
<b>3</b>	<b>Algorytm CMA-ES</b>	<b>7</b>
3.1	Opis algorytmu . . . . .	7
3.2	Macierz kowariancji . . . . .	7
3.3	Wartość oczekiwana rozkładu . . . . .	8
3.4	Długość kroku . . . . .	8
3.5	Generowanie punktów . . . . .	8
3.6	Selekcja punktów . . . . .	8
<b>4</b>	<b>Techniki uwzględniania ograniczeń</b>	<b>9</b>
4.1	Transformacje rozwiązań . . . . .	9
4.1.1	Metoda klasyczna . . . . .	9
4.1.2	Rzutowanie . . . . .	10
4.1.3	Reinicjacja . . . . .	10
4.1.4	Próbkowanie . . . . .	10
4.1.5	Zawijanie . . . . .	10
4.1.6	Odbicie . . . . .	11
4.2	Błędzenie przypadkowe . . . . .	11
4.3	Metoda przeprowadzania testów . . . . .	11
4.3.1	Wartość oczekiwana generowanego punktu . . . . .	12
4.3.2	Rozkład prawdopodobieństwa generowanych punktów . . . . .	12
4.4	Wyniki testów . . . . .	13
4.5	Wnioski . . . . .	23
<b>5</b>	<b>Metodyka testowania metod optymalizacji globalnej</b>	<b>24</b>
5.1	Funkcje benchmarkowe . . . . .	24
5.2	Techniki porównywania wyników . . . . .	24
<b>6</b>	<b>Weryfikacja wpływu technik uwzględniania ograniczeń na efektywność CMA-ES</b>	<b>26</b>
6.1	Cel testów . . . . .	26

6.2	Metoda przeprowadzenia testów . . . . .	26
6.3	Wyniki testów . . . . .	26
6.4	Wnioski . . . . .	29
6.5	Porównanie wariantów trasformacji rozwiązań . . . . .	29
<b>7</b>	<b>Podsumowanie</b>	<b>31</b>
7.1	Wyniki . . . . .	31
7.2	Możliwości rozwoju . . . . .	31

## **1. Streszczenie**

Praca bada wpływ różnych metod uwzględniania ograniczeń w algorytmie CMA-ES. Analizuje, czy można wykorzystać informacje o ograniczeniach kostkowych w tymże algorytmie.

Na początku pracy opisany został algorytm CMA-ES, jako teoretyczne wyjście do badań. W kolejnym rozdziale można znaleźć informacje na temat ograniczeń. Są tam opisane poszczególne techniki. Na bazie błędzenia przypadkowego przeanalizowane są charakterystyczne cechy metod.

Po teoretycznych rozważaniach opisane są testy. W ostatnim rozdziale zebrane są obserwacje poczynione w pracy. Rozdzielił ten, to również wskazanie możliwości poprawy algorytmu CMA-ES.

## **2. Wstęp**

Algorytmy ewolucyjne potrafią rozwiązać problemy o których niewiele wiadomo, ale te klasyczne nie dostosowują się do charakterystyki optymalizowanej funkcji. W większości z nich, podczas generowania punktów, rozkład prawdopodobieństwa jest stały. Z tego faktu wynika problem ustalenia parametrów początkowych. Na przeciw temu wychodzi algorytm CMA-ES, który w swej idei ma dopasowywać się do badanej funkcji.

Rozwinięcie akronimu CME-ES podpowiada, w jaki sposób jest to realizowane: Covariance Matrix Adaptation - Evolution Strategy (adaptacja macierzy kowariancji - strategia ewolucyjna). Jest to algorytm oparty na technikach ewolucyjnych, a punkty losowane są na podstawie macierzy kowariancji, która jest w każdej iteracji dostosowywana do aktualnej sytuacji przeszukiwań. Takie rozwiązanie sprawia, że dobór parametrów początkowych nie jest tak istotny jak w innych algorytmach ewolucyjnych.

CMA-ES jest algorytmem dość skomplikowanym w szczegółach - można o tym przeczytać między innymi w rozdziale 3. Mimo kilkunastu lat istnienia ciągle jednak nie jest do końca zbadany. Jednym z pól do badań są ograniczenia kostkowe.

W wielu problemach napotykamy na ograniczenia - materiały mają wytrzymałość, komputery skończoną pamięć, a czasem ogranicza prędkość światła. Temat ten jest więc istotny i poświęcenie jemu uwagi może zaowocować w przyszłości.

W aktualnej wersji algorytmu CMA-ES punkty, które znajdują się poza ograniczeniem, są na to ograniczenie rzutowane (dokładniej o tej metodzie można przeczytać w rozdziale 4.1.2). Jest to tylko jedna z metod uwzględniania ograniczeń. Warto przyjrzeć się też innym.

### **2.1. Cel pracy**

Celem pracy jest sprawdzenie, w jaki sposób można poprawić algorytm CMA-ES. Analiza będzie skupiała się na ograniczeniach kostkowych.

### 3. Algorytm CMA-ES

#### 3.1. Opis algorytmu

Zgodnie z tym, co zostało ustalone we wstępnie, algorytm CMA-ES jest w swej idei podobny do klasycznych algorytmów ewolucyjnych. Inny jest sposób generowania punktów. Można to zauważyć w poniższym pseudokodzie opisującym algorytm CMA-ES:

inicjalizuj zmienne

dopóki nie są spełnione warunki stopu:

    generuj punkty

$$x_k^{t+1} \sim m^t + \sigma^t N(0, C^t)$$

    oblicz wartości funkcji w wygenerowanych punktach

    posortuj punkty

$$f(x_{1:\lambda}^{t+1}) \leq f(x_{2:\lambda}^{t+1}) \leq \dots \leq f(x_{\lambda:\lambda}^{t+1})$$

    przeprowadź selekcję punktów

    oblicz wartość oczekiwana rozkładu

$$m^{t+1} = m^t + c_m \sum_{i=1}^{\mu} w_i (x_{i:\lambda}^{t+1} - m^t)$$

    oblicz długość kroku

    oblicz macierz kowariancji

Ten pseudokod zawiera kilka pojęć, które nie zostały jeszcze omówione. Ich opis znajduje się w kolejnych podrozdziałach.

#### 3.2. Macierz kowariancji

Pierwszą zmienną, która będzie opisana jest macierz kowariancji. Wszystkie punkty są generowane poprzez losowanie na podstawie macierzy kowariancji. Oznacza to, że nie istnieje bezpośredni związek pomiędzy punktami - w klasycznych algorytmach ewolucyjnych zazwyczaj można było mówić o rodzinach nowo wygenerowanego punktu.

Rola dotychczas wygenerowanych punktów jest jednak istotna w algorytmie CMA-ES. Mają one wpływ na adaptację macierzy kowariancji. Odbywa się to w następujący

sposób: z populacji punktów wybiera się  $\mu$  punktów (zazwyczaj połowę), która zwraca lepsze wartości funkcji celu. Następnie przypisuje się im wagi - w zależności od tego, jak dobry jest punkt. Na postawie tych danych wyznaczana jest empiryczna macierz kowariancji.

### **3.3. Wartość oczekiwana rozkładu**

Punkty generowane są na podstawie rozkładu określonego macierzą kowariancji wokół wartości oczekiwanej. Sama macierz nie mówi jednak nic o wartości oczekiwanej. Wy maga to oddzielnego wyznaczenia tej wartości.

Podobnie jak w macierzy kowariancji, wybiera się  $\mu$  punktów. Dla tych punktów obliczana jest średnia ważona. Ta średnia jest nową wartością oczekiwana.

### **3.4. Długość kroku**

Generowanie punktów z macierzy kowariancji pozwala na skalowanie tego rozkładu poprzez dodanie jednego parametru. Ten parametr jest nazywany długością kroku. W algorytmie CMA-ES długość kroku obliczana jest na podstawie kierunku przemieszczania się wartości oczekiwanej rozkładu. Długość kroku zwiększa się, gdy wartość oczekiwana przemieszcza się w tym samym kierunku. Skraca natomiast, gdy kierunek jest zmienny.

### **3.5. Generowanie punktów**

Punkty są generowane poprzez wylosowanie wektora liczb na postawie macierzy kowariancji. Wylosowany wektor jest skalowany przez długość kroku, a na koniec jest przesuwany o wektor równy wartości oczekiwanej rozkładu.

### **3.6. Selekcja punktów**

Selekcja punktów, to wybór  $\mu$  punktów spośród całej populacji. Wybierane są te punkty, które mają lepszą wartość funkcji. Punktom przypisywane są również wagi wykorzystywane do wyznaczania wartości oczekiwanej.

## 4. Techniki uwzględniania ograniczeń

Niektóre problemy optymalizacyjne posiadają ograniczenia. Szukając rozwiązania należy zapewnić, że będzie ono dopuszczalne. Bazując na [1] techniki uwzględniania ograniczeń można podzielić ze względu na wykorzystanie:

- definicji przestrzeni przeszukiwań - zapewnienie, że podczas krzyżowań, mutacji i innych zmian punktów, żaden z punktów nie wypadnie poza przestrzeń przeszukiwań,
- modyfikacji funkcji celu - zmienienie funkcji celu tak, aby funkcja celu dla punktów spoza ograniczeń zwracały gorsze wyniki,
- transformacji rozwiązań - punkty, które są poza ograniczeniami zostają zamieniane na punkty, które znajdują się w ograniczeniach.

W tej pracy skupiono się na transformacji rozwiązań.

### 4.1. Transformacje rozwiązań

Istnieje wiele technik transformacji rozwiązań spoza ograniczeń na dopuszczalne. W kolejnych podrozdziałach znajdują się metody transformacji rozwiązań, które były badane na potrzeby tej pracy. Każda z technik jest opisana słownie oraz pseudokodem. Opis słowny zawiera wyjaśnienie, co się dzieje z punktem, który znalazł się poza ograniczeniem. W pseudokodzie zastosowano następujące oznaczenia:

- $x$  - punkt, który poddajemy naprawie
- $x'$  - rodzin punktu  $x$ , czyli z punktu  $x'$  z zadanym rozkładem został wygenerowany punkt  $x$
- $x(i)$  - wartość  $i$ -tej współrzędnej punktu  $x$
- $lb$  - ograniczenie dolne
- $ub$  - ograniczenie górne

#### 4.1.1. Metoda klasyczna

Nowy punkt zostaje odrzucony i wraca do poprzedniej pozycji.

$$x = x'$$

#### 4.1.2. Rzutowanie

Punkt jest transformowany do najbliższego punktu, który spełnia ograniczenie. Oznacza to, że dla każdej współrzędnej sprawdzany jest warunek zawierania się w ograniczeniach. Wartości współrzędnych, dla których nie jest on spełniony, zmieniane są na wartości ograniczeń (odpowiednio dolne lub górne), które są najbliższe.

```
dla każdej współrzędnej i  
jeżeli lb(i) > x(i)  
    x(i) = lb(i)  
jeżeli ub(i) < x(i)  
    x(i) = ub(i)
```

#### 4.1.3. Reinicjacja

Punkt jest przenoszony do pozycji początkowej.

```
x = x0
```

#### 4.1.4. Próbkowanie

Punkt jest powtórnie generowany do momentu, aż spełni ograniczenie kostkowe.

```
dopóki !w_ograniczeniach(x)  
    x = losuj(x')
```

#### 4.1.5. Zawijanie

Dla każdej współrzędnej sprawdzane są warunki na ograniczenie. W przypadku współrzędnych, na których punkt jest poza ograniczeniem, różnica pomiędzy ograniczeniem a wartością współrzędnej punktu jest zapamiętywana. Tę różnicę odkładamy na przeciwnym ograniczeniu po stronie, która jest wewnątrz ograniczenia. W tym miejscu znajduje się nowa wartość współrzędnej punktu. W intuicyjny sposób można to wyjaśnić tak, że dla punktów nie ma ograniczeń, a przestrzeń przeszukiwań po każdym wymiarze jest jakby "zawinięta".

```
dla każdej współrzędnej i  
jeżeli lb(i) > x(i)  
    x(i) = ub(i) - (lb(i) - x(i))  
jeżeli ub(i) < x(i)  
    x(i) = lb(i) + (x(i) - ub(i))
```

#### 4.1.6. Odbicie

Dla każdej współrzędnej sprawdzane są warunki na ograniczenie. W przypadku współrzędnych, na których punkt jest poza ograniczeniem, wartość punktu tej współrzędnej jest symetrycznie odbita względem ograniczenia, którego warunek został złamany.

```
dla każdej współrzędnej i
    jeżeli lb(i) > x(i)
        x(i) = x(i) + 2 * (lb(i) - x(i))
    jeżeli ub(i) < x(i)
        x(i) = x(i) - 2 * (x(i) - ub(i))
```

## 4.2. Błądzenie przypadkowe

Można się spodziewać, że algorytm CMA-ES dla funkcji stałej będzie zachowywał się analogicznie do błądzenia przypadkowego. Takie założenie skłoniło autora, żeby zbadać błądzenie przypadkowe z ograniczeniami. Błądzenie przypadkowe jest algorytmem prostszym niż CMA-ES, więc umożliwia szybsze testowanie i wyciąganie wniosków.

Niech  $X_1, X_2, \dots$  będą niezależnymi n-wymiarowymi zmiennymi losowymi o wartości oczekiwanej równej  $[0]^n$ . Błądzeniem przypadkowym nazywamy sekwencję punktów  $S_1, S_2, \dots$ , takich że:

$$S_1 = X_1, S_i = S_{i-1} + X_i \quad (1)$$

Zmienne losowe mogą być realizowane w różny sposób. Mogą to być wektory o rozkładzie normalnym, jednostajnym lub innym.

## 4.3. Metoda przeprowadzania testów

Celem testów jest zaobserwowanie, jaki wpływ na symulację ma metoda uwzględniania ograniczeń. Do tak postawionego celu można podejść na różne sposoby. Wybrano 2 cechy, które miały najlepiej zrealizować cel:

1. Wartość oczekiwana generowanego punktu (razem z poprawą).
2. Rozkład prawdopodobieństwa generowanych punktów.

Sposoby testowania obu cech są opisane w kolejnych podrozdziałach.

#### 4.3.1. Wartość oczekiwana generowanego punktu

Błądzenie przypadkowe jest dobrze określone. W tym algorytmie wartością oczekiwana losowanej liczby  $X_i$  jest 0. Implikuje to fakt, że wartością oczekiwana  $S_i$  jest  $S_{i-1}$ . Ustalenie ograniczeń kostkowych powoduje, że nadal losujemy liczbę  $X_i$ , której wartość oczekiwana jest równa 0. Dodajemy jednak metody naprawy. W związku z tym wartością oczekiwana  $S_i$  nie zawsze jest  $S_{i-1}$ .

Dla każdego punktu  $p$  przestrzeni przeszukiwań można przypisać wektor  $\vec{d}$ . Początek  $\vec{d}$  znajduje się w  $p$ . Założymy teraz, że  $p$  jest punktem  $S_i$  pewnego błądzenia przypadkowego z ograniczeniami. Koniec wektora  $\vec{d}$  znajduje się w wartości oczekiwanej punktu  $S_{i+1}$ .

Wykres, który będzie zawierał wektory  $\vec{d}$ , pozwoli na określenie charakterystyki przemieszczania się punktów w poszczególnych częściach przestrzeni przeszukiwań.

### Implementacja

Do wizualizacji wybrano symulację, w której punkty posiadają 2 wymiary. Pozwala to na przejrzystą wizualizację wektorów dla każdego z punktów. Pokazuje też zachowanie wektorów w rogach - punktach, które są blisko ograniczeń.

Dla każdego z 2 wymiarów wybrano kilkanaście, równoodległych punktów. W skład tych punktów wchodziły też punkty znajdujące się na ograniczeniach. Następnie stworzono 2 pętle po tych punktach (jedna zagnieżdzona w drugiej). W ten sposób uzyskano regularnie rozłożone punkty w dwóch wymiarach.

Dla każdego z tych punktów przeprowadzono następujące obliczenia. 1000 razy uruchomiono symulację jednego kroku algorytmu błądzenia przypadkowego (z rozkładem normalnym) z odpowiednią naprawą. Otrzymano tysiąc punktów, z których następnie obliczono średnią arytmetyczną. Tak uzyskany punkt potraktowano jako wartość oczekiwana i wyrysowano wektor  $\vec{d}$ .

#### 4.3.2. Rozkład prawdopodobieństwa generowanych punktów

Punkty w błądzeniu przypadkowym poruszają się chaotycznie. Ta losowość jest częściowo uporządkowana na ograniczeniach. W zależności od metody naprawy, punkty

zachowują się inaczej, a rozkład prawdopodobieństwa jest zniekształcony. Autor postanowił przyjrzeć się jak wygląda rozkład prawdopodobieństwa wystąpienia punktu z perspektywy całej symulacji.

Dla każdej metody naprawy opisanej w 4.1 jednokrotnie uruchomiono algorytm błędzenia przypadkowego jednego punktu. Oddziennie symulowano również błędzenie z rozkładem normalnym oraz jenostajnym na przedziale  $[-0.5; 0.5]$  (przedział co najmniej kilkukrotnie krótszy od ograniczeń kostkowych). Po każdej iteracji nowo wygenerowany punkt zapisywano w tablicy. Z tak przygotowanej tablicy generowano histogram.

### Implementacja

Dla każdej z metod naprawy przygotowano oddzielny skrypt. Pseudokod skryptu zamieszczono poniżej.

```
x - błędzący punkt
punkty - tablica wszystkich położen punktu x
iteracje - liczba iteracji podana jako parametr
i = 0
dopóki i < iteracje
    d = wektor wylosowany z zadanym rozkładem
    x = x + d
    jeżeli x jest poza ograniczeniem
        popraw x
    dodaj x do tablicy punkty
    i = i + 1
```

## 4.4. Wyniki testów

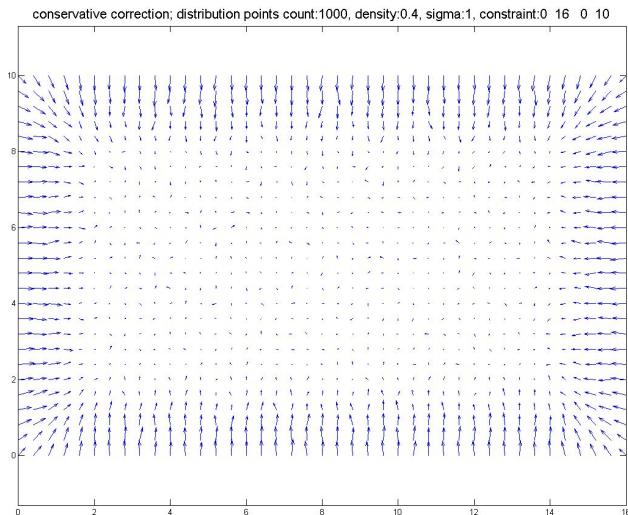
Wykresy zamieszczone w tym podrozdziale są 2 rodzajów. Są to wykresy wektorów  $\vec{d}$  wartości oczekiwanych oraz histogramy wystąpień punktu.

Histogramy były tworzone poprzez symulację skryptów z liczbą iteracji wynoszącą 2-100 milionów. Szerokość przedziałów jest różna i była dobierana tak, aby jak najlepiej przedstawić interesujące fakty. Wszystkie wykresy, które pokazują wyniki błędzenia przypadkowego w dwóch wymiarach zostały przetworzone w następujący sposób: otrzymane wyniki symulacji powielono poprzez symetryczne odbicie ich względem osi X, osi Y oraz początku układu współrzędnych. Wykonano ten zabieg, aby zwiększyć

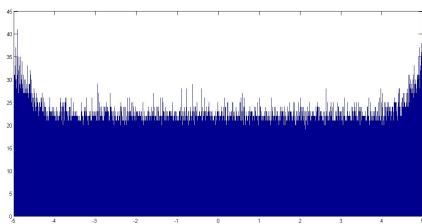
dokładność wyników. Nie wpływają one na badania, ponieważ punkty 0 są środkami ograniczeń obu osi, a z perspektywy algorytmu nie ma znaczenia, w której ćwiartce znajduje się aktualnie punkt.

### Metoda klasyczna

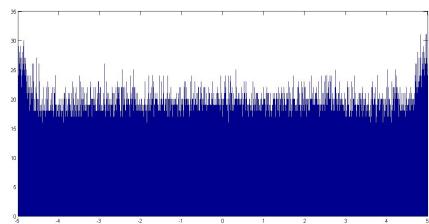
Wektory wartości oczekiwanych, które są blisko ograniczeń są skierowane od ograniczenia. Histogramy z kolei pokazują, że punkty z większym prawdopodobieństwem występują przy ograniczeniach. Takie zachowanie wynika z tego, że potomkowie punktów blisko granicy mogą ”wypaść” poza ograniczenie. Po naprawie dziecko będzie tym samym punktem.



Rysunek 1: Wartość oczekiwana punktu

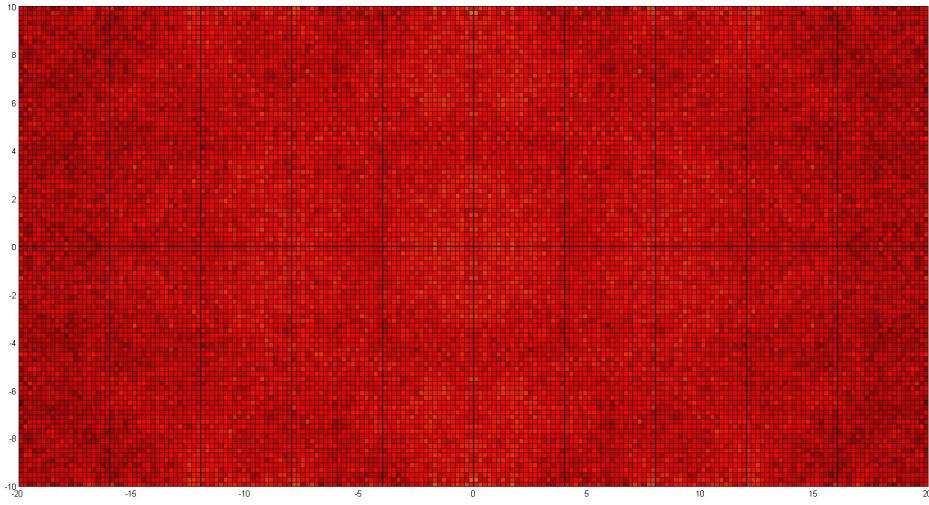


(a) Generowanie z rozkładem normalnym



(b) Generowanie z rozkładem jednostajnym

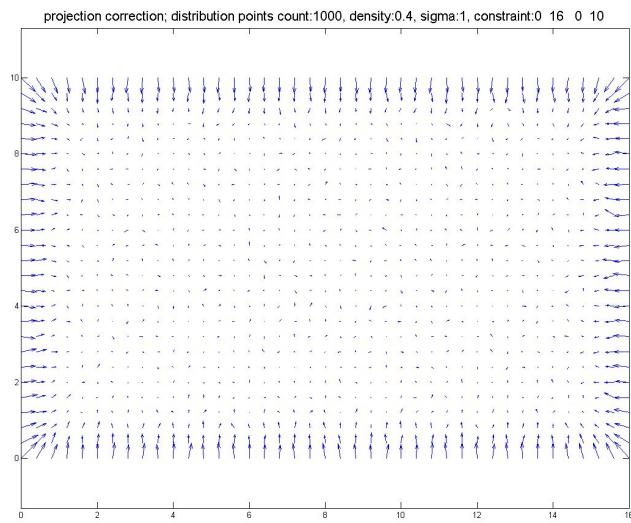
Rysunek 2: Rozkład prawdopodobieństwa punktów; 1 wymiar



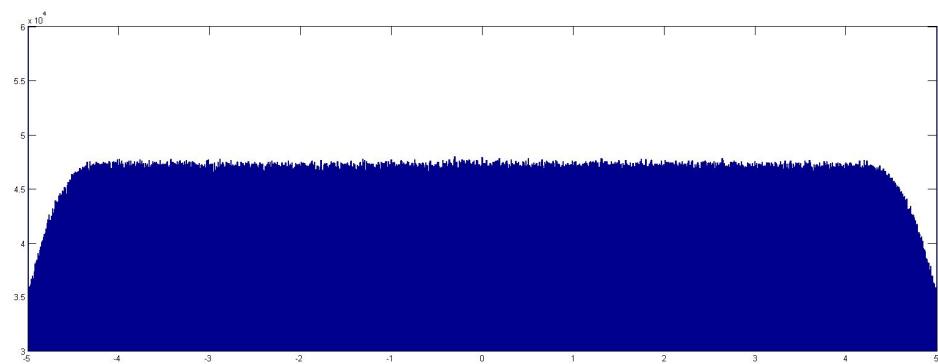
Rysunek 3: Rozkład prawdopodobieństwa punktów; 2 wymiary; generowanie z rozkładem normalnym

## Rzutowanie

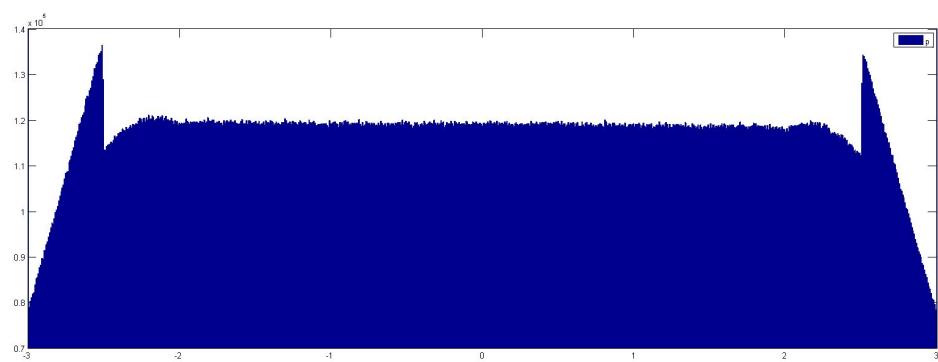
Zgodnie z przewidywaniami największe prawdopodobieństwo wystąpienia punktu jest na ograniczeniu, co bardzo dobrze obrazuje rysunek 7. Na rysunkach 5 i 6 zakres osi Y został zmniejszony do odpowiednio [30000; 60000] oraz [70000; 140000], aby uwypuklić interesujące efekty. W związku z tym, wartości przedziałów brzegowych nie są widoczne - były o kilka rzędów wielkości większe od pozostałych wartości. Warto zwrócić uwagę na niespodziewane zjawisko - wartości przedziałów blisko ograniczeń są mniejsze, niż pozostałe. Oznacza to, że punkty w tych przedziałach występują z mniejszym prawdopodobieństwem, niż pozostałe. Wydawać by się mogło, że powinno być inaczej, ponieważ podczas symulacji punkty często są rzutowane na ograniczenie. Dodatkowo, w rozkładzie jednostajnym widać, że istnieją przedziały, w których punkty występują z większym prawdopodobieństwem. Autorowi nie udało się formalnie uzasadnić tego zjawiska. Intuicja prowadzi do hipotezy, że punkty, które znalazły się poza ograniczeniem, bez naprawy po kilku iteracjach zapełniałyby "dołek". Naprawa natomiast sprawia, że cała dalsza symulacja zostaje niejako przesunięta. W ten sposób powstaje "górkę" w rozkładzie jednostajnym. Brak górek w rozkładzie normalnym można tłumaczyć charakterystyką samego rozkładu - spadek prawdopodobieństwa wraz z oddalaniem się od wartości oczekiwanej.



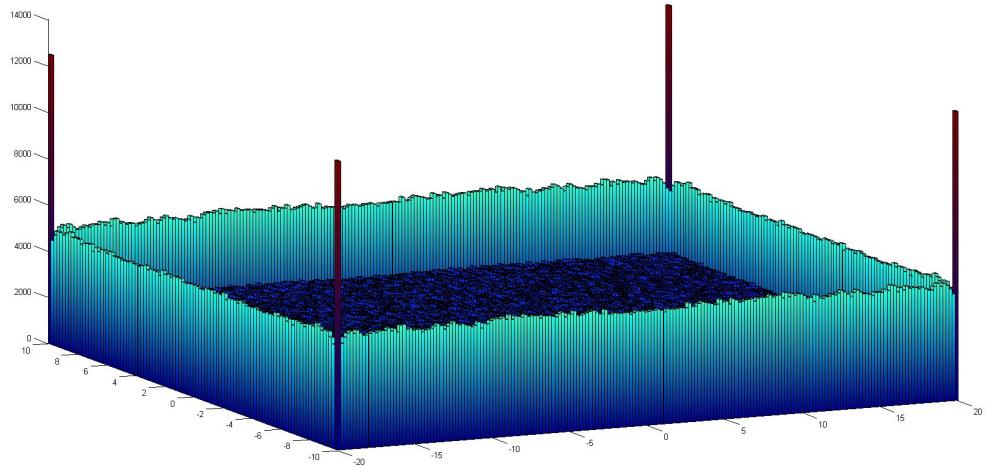
Rysunek 4: Wartość oczekiwana punktu



Rysunek 5: Rozkład prawdopodobieństwa punktów; 1 wymiar; generowanie z rozkładem normalnym



Rysunek 6: Rozkład prawdopodobieństwa punktów; 1 wymiar; generowanie z rozkładem jednostajnym



Rysunek 7: Rozkład prawdopodobieństwa punktów; 2 wymiary; generowanie z rozkładem normalnym

### Reinicjacja

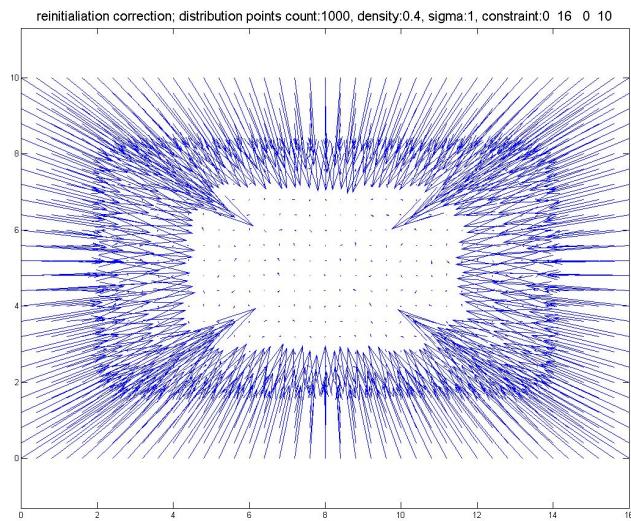
Wykres wartości oczekiwanych jest mało czytelny, ponieważ wartości przesunięć są duże. Ma w tym udział reinicjacja, która przesuwa część punktów na środek przestrzeni przeszukiwań.

Histogramy nie ukazują nic nadzwyczajnego, ponieważ łatwo zauważać pik związanego z reinicjacją oraz wartości histogramu malejące wraz ze zbliżaniem się do ograniczeń. Warto jednak zwrócić uwagę na 2 fakty.

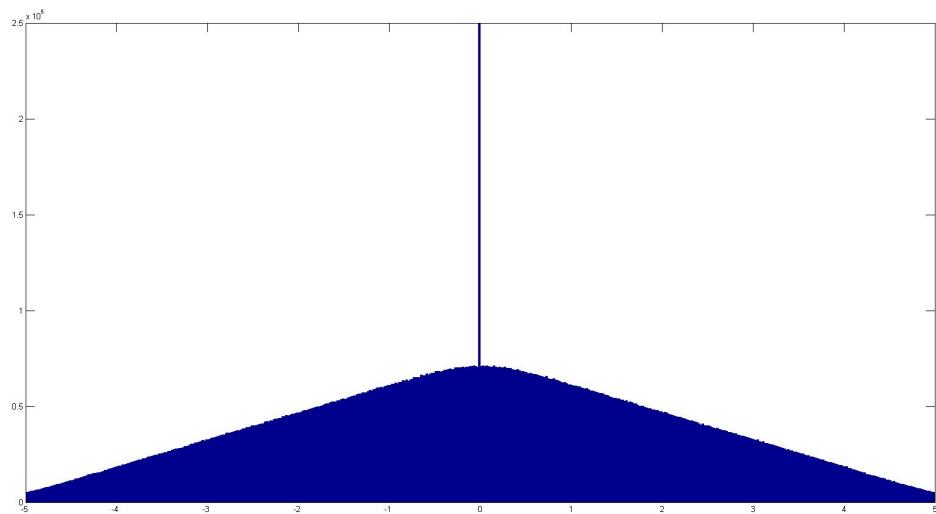
Pierwsze spostrzeżenie, to kształt histogramu. Zarówno dla rozkładu jednostajnego, jak i normalnego w jednym wymiarze jest on stożkowy. Łuk można zauważać tylko blisko punktu środkowego, w pozostałej części spadek jest liniowy. Brakuje charakterystycznego, gaussowskiego przegięcia. Sytuacja jest ciekawsza, gdy występuje więcej wymiarów. Widać wówczas przegięcie (Rysunek 12). Dokładniejsze badania pokazały, że przegięcie nie występuje tylko na jednym wymiarze - tym, który jest relatywnie najkrótszy. Celowo jest użyte słowo relatywnie, ponieważ z perspektywy błędzenia przypadkowego i rozkładu normalnego trzeba brać pod uwagę parametr  $\sigma$  - odchylenie standardowe. Wymiary o małym  $\sigma$  będą relatywnie dłuższe od tych z dużym  $\sigma$ , ponieważ błędzenie będzie wykonywało mniejsze kroki.

Na rysunku 10 można dostrzec też dwa uskoki, które związane są z pikiem w punkcie 0

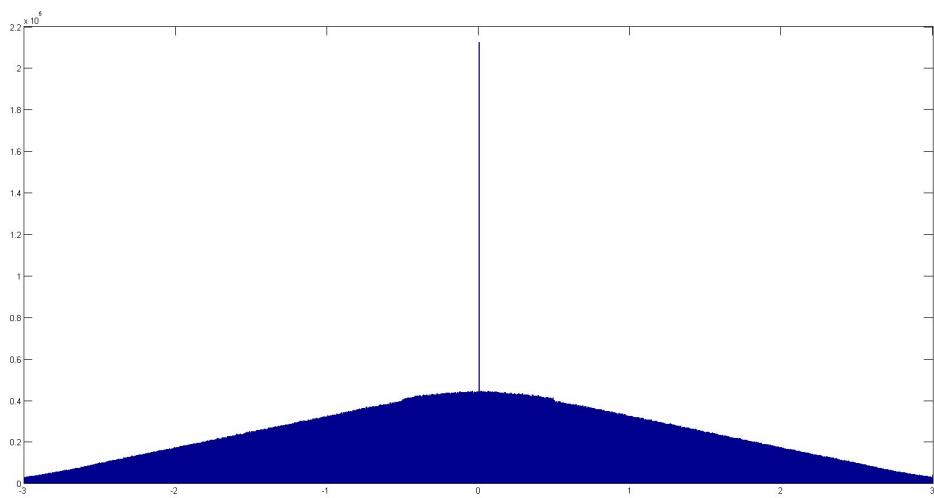
oraz charakterystyką rozkładu jednostajnego.



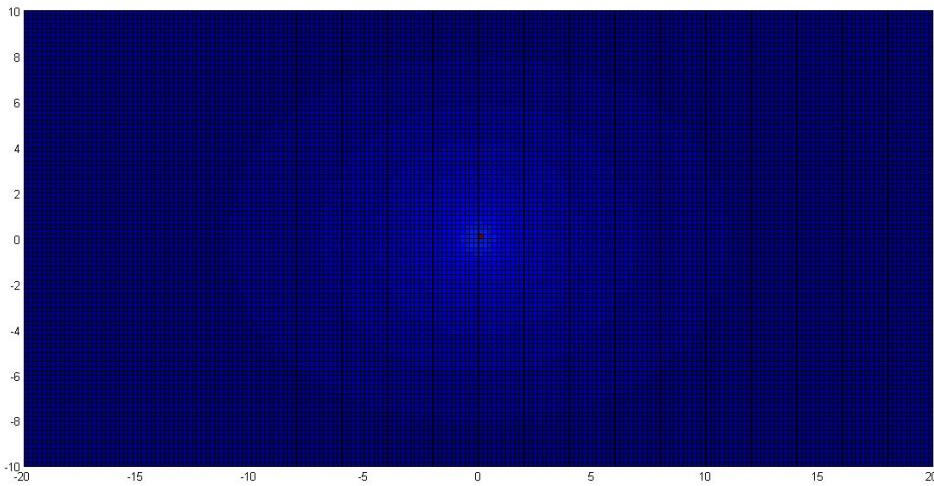
Rysunek 8: Wartość oczekiwana punktu



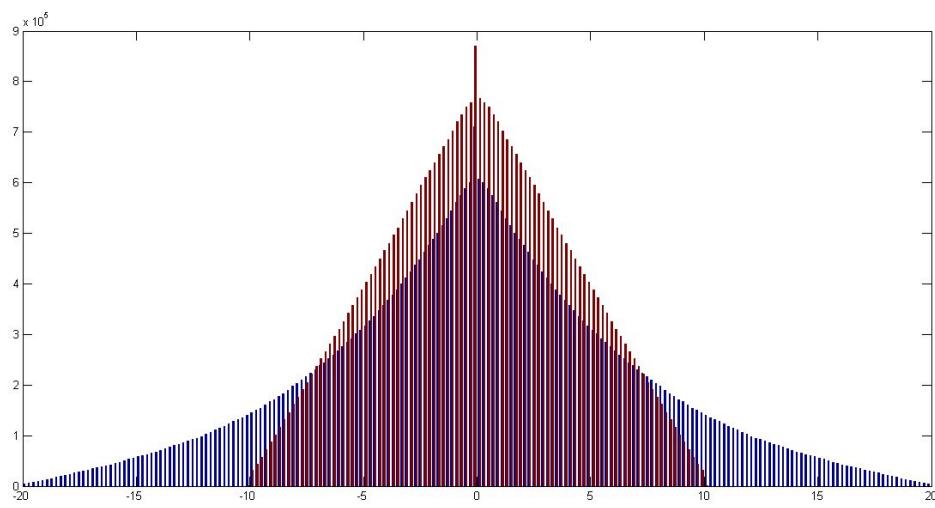
Rysunek 9: Rozkład prawdopodobieństwa punktów; 1 wymiar; generowanie z rozkładem normalnym



Rysunek 10: Rozkład prawdopodobieństwa punktów; 1 wymiar; generowanie z rozkładem jednostajnym



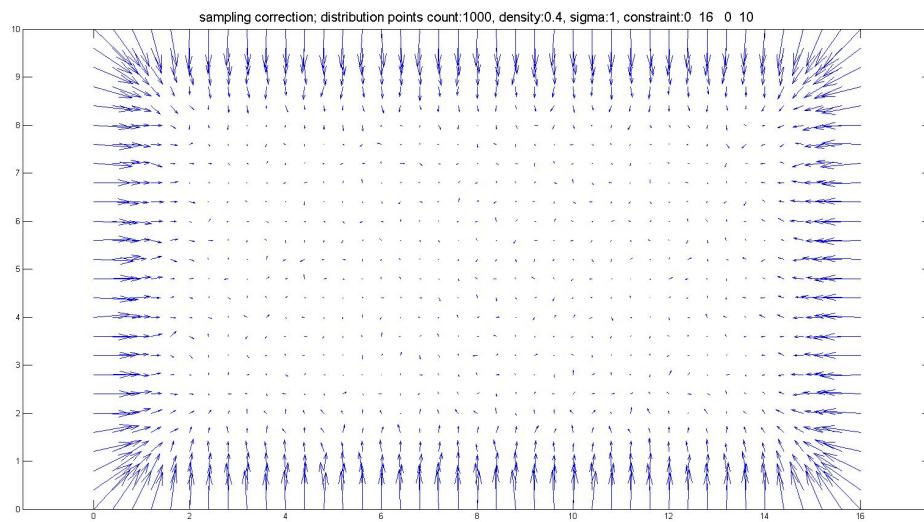
Rysunek 11: Rozkład prawdopodobieństwa punktów; 2 wymiary; generowanie z rozkładem normalnym



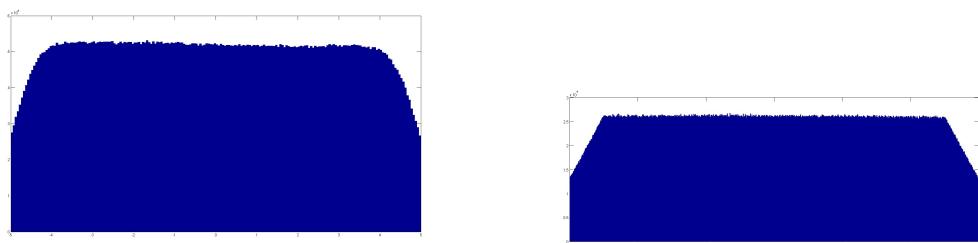
Rysunek 12: Rozkład prawdopodobieństwa punktów; 2 wymiary; generowanie z rozkładem normalnym; oddzielne histogramy dla obu wymiarów

## Próbkowanie

Ten rozkład charakteryzuje się spadkiem wartości prawdopodobieństwa wraz ze zbliżaniem się do ograniczenia. Punkty, które wypadłyby poza ograniczenia oraz ich potomkowie są przesuwane w kierunku środka przedziału.

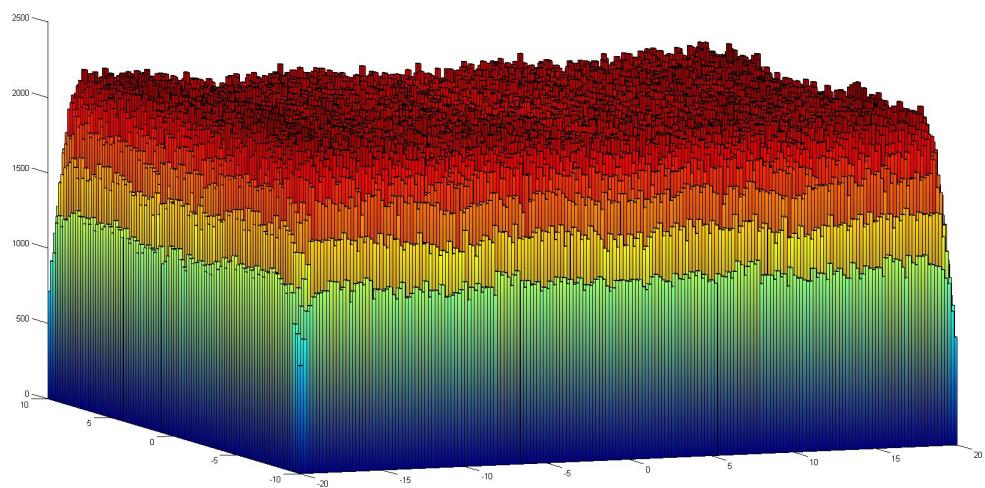


Rysunek 13: Wartość oczekiwana punktu



(a) Generowanie z rozkładem normalnym      (b) Generowanie z rozkładem jednostajnym

Rysunek 14: Rozkład prawdopodobieństwa punktów; 1 wymiar

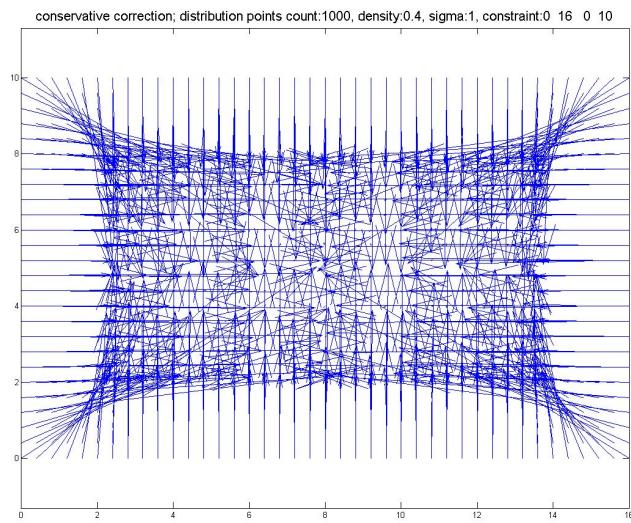


Rysunek 15: Rozkład prawdopodobieństwa punktów; 2 wymiary; generowanie z rozkładem normalnym

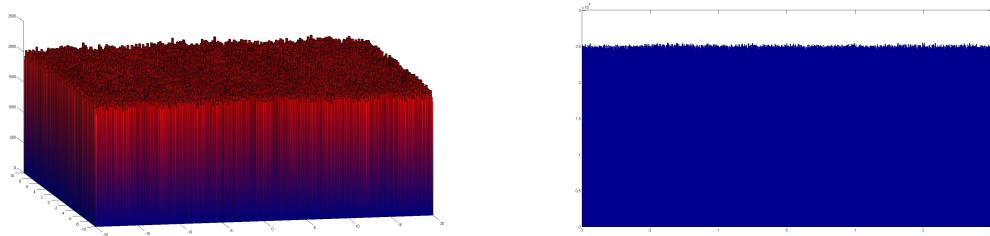
## Zawijanie

Wykres przesunięć wartości oczekiwanych jest bardzo nieczytelny. Jest to implikacji dużych przesunięć punktów - te, które znajdują się poza ograniczeniem, wędrują na drugą stronę wykresu.

Uzasadniając wykresy histogramów warto wyobrazić sobie, że symulacja przeprowadzana jest bez ograniczeń, a następnie wykres z wynikami przecinany jest w równych odstępach wzdłuż każdej osi. Na koniec tak pocięte części łączone są w jeden wykres. Takie zachowanie symuluje zawijanie. W związku z tym nie dziwi fakt, że histogramy przedstawiają rozkład jednostajny z szumem.



Rysunek 16: Wartość oczekiwana punktu



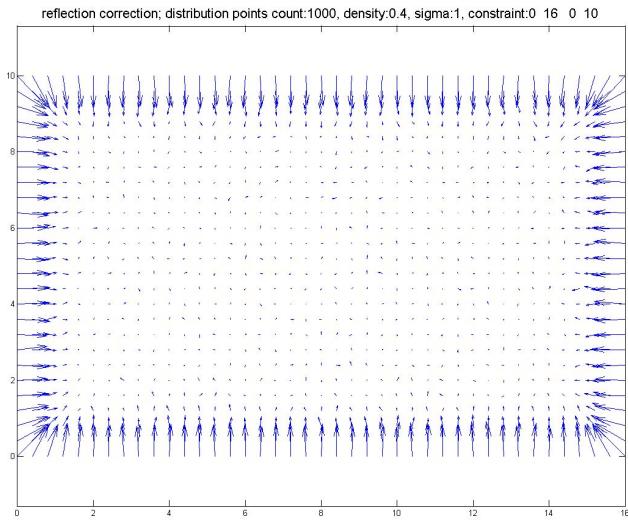
(a) 1 wymiar; generowanie z rozkładem normalnym      (b) 2 wymiary; generowanie z rozkładem jednostajnym

Rysunek 17: Rozkład prawdopodobieństwa punktów;

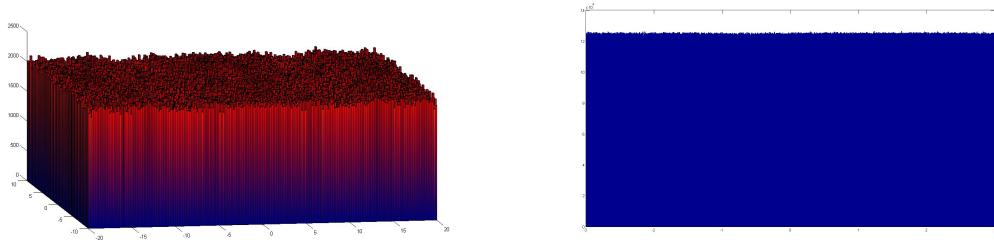
## Odbicie

Podobnie jak zawijanie, odbicie zwraca histogram rozkładu normalnego z szumem. W tej sytuacji nieco trudniej o analogię, lecz po chwili zastanowienia nie dziwi kształt poniższych wykresów.

Od zawijania różni się natomiast wykres wartości oczekiwanych, ponieważ w odbiciu nie ma znaczących przesunięć punktów.



Rysunek 18: Wartość oczekiwana punktu



(a) 1 wymiar; generowanie z rozkładem normalnym      (b) 2 wymiary; generowanie z rozkładem jednostajnym

Rysunek 19: Rozkład prawdopodobieństwa punktów;

## 4.5. Wnioski

W większości metod zachowanie pojedynczego punktu przekłada się na globalne zachowanie całego algorytmu. Testy pokazały cechy, które nie były oczywiste podczas prostej analizy algorytmów.

Z perspektywy algorytmu CMA-ES warto zwrócić uwagę przede wszystkim na metody zawijania i reinicjacji. W tych metodach wartości oczekiwane punktów znajdują się daleko od rodzica. Oznacza to, że w algorytmie CMA-ES macierz kowariancji może znacząco się zniekształcać.

## **5. Metodyka testowania metod optymalizacji globalnej**

Funkcje posiadają charakterystyczne cechy, m.in. monotoniczność, liczba ekstremów, ciągłość. Metody optymalizacji globalnej również posiadają swoje charakterystyczne własności. W zależności od przypadku, wymagania stawiane algorytmowi mogą być różne: szybkości działania, przeszukanie całej przestrzeni, czy też dokładne określenie ekstremum. Połączenie tego wszystkiego powoduje, że trudno jednoznacznie określić, która metoda optymalizacji jest najlepsza. Warto jednak posiadać wiedzę w których przypadkach metoda przynosi lepsze rezultaty.

Zapewne zgodziliby się z tym stwierdzeniem twórcy konkursów takich jak CEC, czy BBOB. Tworzą oni szereg funkcji benchmarkowych, które mają na celu empirycznie pokazać silne i słabe strony algorytmów w konkretnych sytuacjach.

### **5.1. Funkcje benchmarkowe**

Funkcje benchmarkowe, to złożone funkcje. Skomplikowanie polega między innymi na wielu ekstremach funkcji lub specyficzny umieszczeniu minimum globalnego. Dzięki nim można porównać algorytmy.

### **5.2. Techniki porównywania wyników**

Wybranie funkcji benchmarkowych nie wystarczy, aby stwierdzić który z algorytmów ewolucyjnych jest lepszy. Sa to algorytmy niedeterministyczne - poszukiwanie rozwiązania za każdym razem będzie wyglądało inaczej nawet przy takich samych parametrach. Oznacza to, że dla wiarygodności wyników należy testy uruchamiać wielokrotnie. Posiadając takie wyniki należy posłużyć się wybraną metodą, aby porównać skuteczność algorytmów. W tej pracy do porównywania został wybrany test Wilcooxona, ponieważ można go użyć do porównywania par obserwacji oraz ten test pokazuje różnice cech.

## **Test Wilcoxona**

Test Wilcoxona jest testem nieparametrycznym - nie jest potrzebna wiedza na temat rozkładu populacji [2]. Ten test sprawdza, czy istnieje statycznie istotna różnica między dwoma zbiorami, które były losowane z pewnym rozkładem prawdopodobieństwa. Hipoteza dotyczy median wygenerowanych zbiorów. Hipoteza zerowa  $H_0$  brzmi "Różnica median zbiorów wynosi 0". Na podstawie sprawdzenia hipotezy możemy wywnioskować, czy 2 rozkłady prawdopodobieństwa generują dane istotnie różne. Do przedstawienia wyników wielu testów Wilcoxona wykorzystuje się tabele.

## **6. Weryfikacja wpływu technik uwzględniania ograniczeń na efektywność CMA-ES**

### **6.1. Cel testów**

Zrealizowanie celu całej pracy wymaga sprawdzenia, jak można usprawnić algorytm CMA-ES. Trudno byłoby to zrealizować bez przetestowania samego algorytmu.

Celem testów będzie sprawdzenie charakterystycznych zachowań macierzy kowariancji. Aby zrealizować ten cel zostaną przeprowadzone testy, które sprawdzają rozkład prawdopodobieństwa punktów. Porównana zostanie także skuteczność algorytmu CMA-ES w zależności od sposobu uwzględniania ograniczeń.

### **6.2. Metoda przeprowadzenia testów**

Testy przeprowadzano poprzez jednokrotne uruchomienie zmodyfikowanego algorytmu CMA-ES na funkcji stałej, losowej oraz kwadratowej. Modyfikacja polegała na usunięciu warunków stopu z pętli głównej algorytmu. Wynikała ona z tego, że po kilku iteracjach algorytm się zatrzymywał. Nie modyfikowano natomiast sposobu uwzględniania ograniczeń. Algorytm był przerywany po kilkuset iteracjach na podstawie logów w konsoli. Wszystkie wygenerowane przez algorytm punkty zostały przedstawione na histogramie (podobnie jak przy błędzeniu przypadkowym).

Do przeprowadzania testów została użyta biblioteka przygotowana przez Nikolausa Hansena, współautora algorytmu CMA-ES. Tak jak w przypadku błędnego przypadkowego, wykorzystano implementację w języku MATLAB [6].

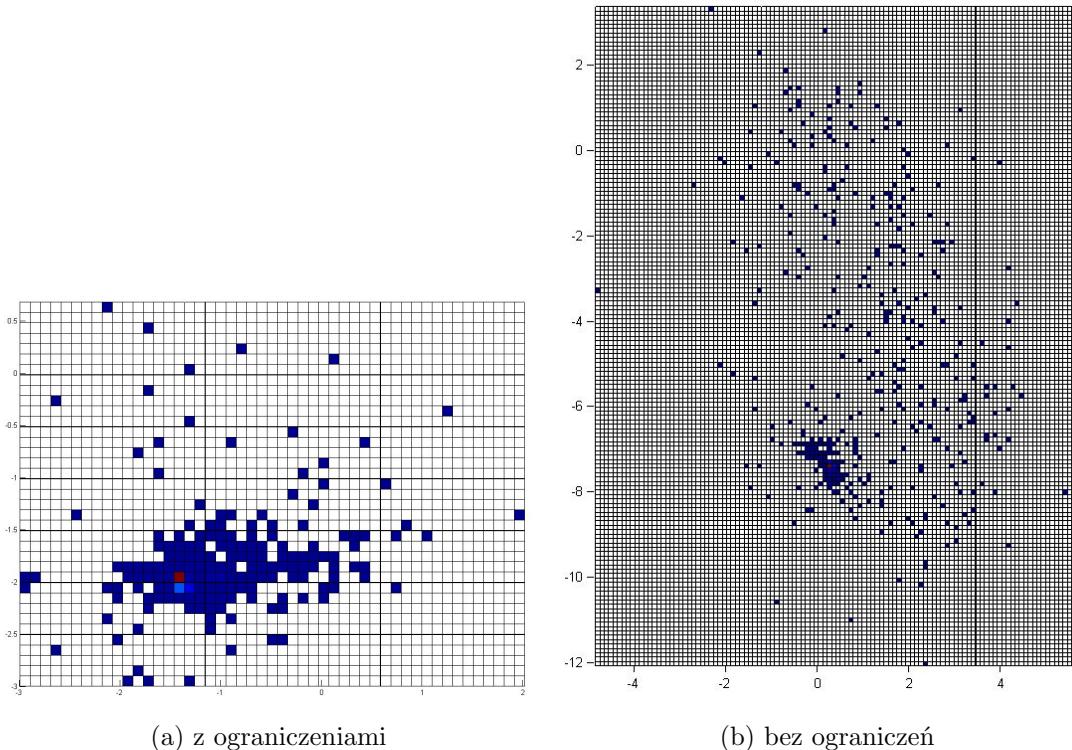
### **6.3. Wyniki testów**

#### **Funkcja stała**

Wbrew oczekiwaniom algorytm CMA-ES uruchomiony na funkcji stałej ma tendencje do zmniejszania kroku algorytmu, więc po kilkuset iteracjach przemieszczanie się wartości oczekiwanej jest znikome. Obrazują to rysunki w tym podrozdziale. Widać na nich fragmenty, w których punkty były losowane z dużą różnicą, obszar gdzie różnice

były małe oraz miejsce, w którym algorytm znalazł się i prawie nie poruszał (czerwony kwadrat)

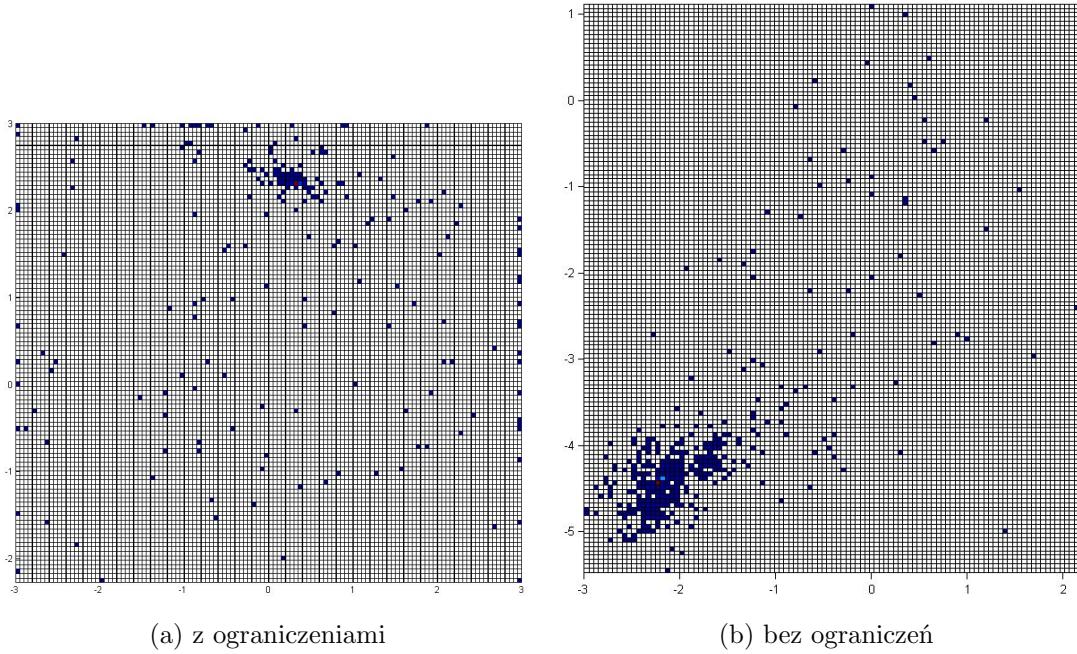
Przeprowadzono testy, w których algorytm posiadał ograniczenie  $[-3; 3]$  na obu wymiarach oraz test bez ograniczeń.



Rysunek 20: Histogram punktów; algorytm CMA-ES; funkcja stała; 2 wymiary

### Funkcja losowa

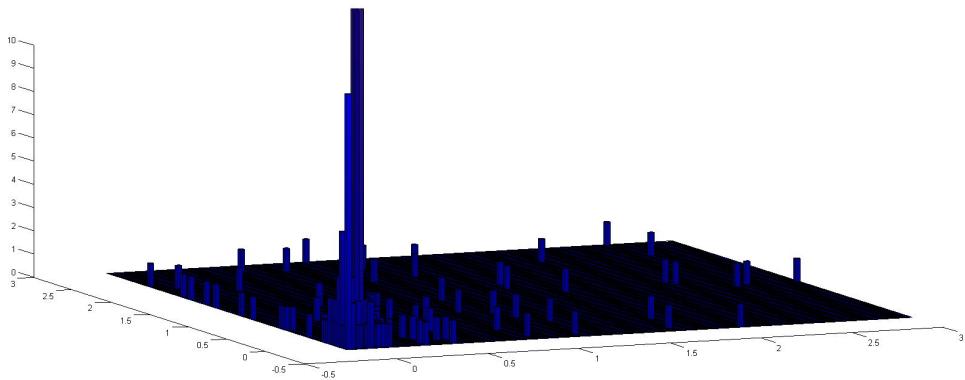
Podobnie jak w przypadku obliczeń funkcji stałej krok algorytmu CMA-ES na funkcji losowej zmniejszał się dość szybko. Poniższe wykresy, podobnie jak w przypadku testowania funkcji stałej, przedstawiają histogram wystąpień punktów w algorytmie CMA-ES z ograniczeniami oraz bez ograniczenia.



Rysunek 21: Histogram punktów; algorytm CMA-ES; funkcja losowa; 2 wymiary

### Funkcja kwadratowa dwuwymiarowa

Badano funkcję o wzorze  $y = x_1^2 + x_2^2$ . Dla tej funkcji zdecydowano się zmienić ograniczenie. Wynosiło ono  $[0,1;3]$  na obu wymiarach. Początkowa wartość oczekiwana wynosiła  $[2;2]$ . Wykres dla tak przygotowanej symulacji znajduje się na rysunku 22 (osi Z została obcięta do przedziału  $[0;10]$ ).



Rysunek 22: Histogram punktów; algorytm CMA-ES; funkcja losowa; 2 wymiary

## **6.4. Wnioski**

Na wykresach, które przedstawiają histogramy punktów podczas symulacji algorytmu CMA-ES na funkcji losowej i stałej, nie udało się zaobserwować żadnych efektów widocznych podczas testowania błądzenia przypadkowego. Autor nie zdecydował się na generowanie wykresów innych metod transformacji rozwiązań, ponieważ nie dałoby to widocznych różnic w wykresach. Zdecydowano się jednak na porównanie za pomocą funkcji CEC dwóch wariantów transformacji rozwiązań.

## **6.5. Porównanie wariantów trasformacji rozwiązań**

Celem testów było sprawdzenie, czy zmiana technik uwzględniania ograniczeń w algorytmie CMA-ES wpływa na rozwiązanie. Do tego porównania wybrano 2 warianty transformacji rozwiązań: ewolucję darwinowską i ewolucję lamarkowską, które zostały opisane poniżej. Dla obu wariantów uruchomiono symulacje algorytmu CMA-ES. Do tego celu wykorzystano funkcje benchmarkowe z konkursu CEC 2013, których dokładny opis znajduje się tutaj [8]. Ten zestaw został wybrany ze względu na duże zróżnicowanie funkcji oraz implementację w języku MATLAB, z której skorzystano [9]. Algorytm CMA-ES (dla obu typów ewolucji) uruchomiono po 50 razy dla każdej funkcji benchmarkowej (30 wymiarów). Algorytm CMA-ES dla każdej symulacji zwrócił najlepszy punkt, który znalazł. W ten sposób dla 28 funkcji uzyskano po 50 wartości. Wyniki zostały porównane testem Wilcooxona, aby sprawdzić, czy w istotny sposób się różnią.

### **ewolucja lamarkowska**

Punkt, który wypadnie poza ograniczenie jest transformowany wewnątrz ograniczenia. Po transformacji jest obliczana wartość funkcji punktu.

### **ewolucja darwinowska**

Punkt nie jest transformowany wewnątrz ograniczenia. Wartość funkcji celu obliczamy w następujący sposób: kopujemy punkt, przekształcamy symetrycznie względem tych ograniczeń, które zostały przekroczone, a na koniec obliczamy wartość funkcji dla tego

punktu.

## Wyniki

Wyniki zostały zebrane w tabelę, w której zastosowano następujące oznaczenia:

- + oznacza, że lepszy okazał się wariant ewolucji lamarkowskiej,
- - oznacza, że lepszy okazał się wariant ewolucji darwinowskiej,
- . oznacza, że wyniki obu wariantów różnią się nieznacznie.

Numer funkcji	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wynik	.	.	.	.	.	.	+	.	.	.	.	.	.	.
Numer funkcji	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Wynik	.	.	.	.	.	.	.	.	.	.	.	.	.	.

## **7. Podsumowanie**

### **7.1. Wyniki**

### **7.2. Możliwości rozwoju**

## Literatura

- [1] *Wykłady z algorytmów ewolucyjnych*, Jarosław Arabas, 2004
- [2] *The Wilcoxon Signed-Rank Test*, Richard Lowry, <http://vassarstats.net/textbook/ch12a.html>
- [3] *Różnicowa implementacja algorytmu CMAES*, Michał Bobowski, 2015
- [4] *An Introduction to Probability Theory and its Applications, Volume II*, William Feller, 1970
- [5] *Completely Derandomized Self-Adaptation in Evolution Strategies w Evolutionary Computation*, 9(2), pp. 159-195, Nikolaus Hansen, Andreas Ostermeier, 2001
- [6] *CMA-ES: Evolution Strategy with Covariance Matrix Adaptation for nonlinear function minimization*, Nikolaus Hansen, [https://www.lri.fr/~hansen/cmaes\\_inmatlab.html](https://www.lri.fr/~hansen/cmaes_inmatlab.html)
- [7] *The CMA evolution strategy: A tutorial.* arXiv:1604.00772v1, Nikolaus Hansen, 2016
- [8] *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*, J. J. Liang, B. Y. Qu, P. N. Suganthan, Alfredo G. Hernández-Díaz, 2013
- [9] *CEC13 Test Function Suite* , Jane Jing Liang, <http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/CEC2013/cec13matlab.zip>

Warszawa, dnia 30 maja 2016

## Oświadczenie

Oświadczam, że pracę magisterską pod tytułem „Analiza możliwości wykorzystania w algorytmie CMA-ES wiedzy o ograniczeniach kostkowych”, której promotorem jest dr hab. inż. Jarosław Arabas prof. nzw. PW, wykonałem samodzielnie, co poświadczam własnoręcznym podpisem.

---