**Statystyka – projekt zaliczeniowy**

**Tomasz Janeczko**

**Dominik Marek**

**I Wybrane funkcje i algorytmy**

* **Funkcja Rosenbrocka – funkcja niewypukła mająca wyłącznie wartości nieujemne używana do testowania algorytmów optymalizacji**
* **Funkcja Michalewicza – funkcja wielomodalna używana do testowania algorytmów optymalizacji**
* **Algorytm MS – uznawany za skuteczną metodę analizy wielu sekwencji**
* **Algorytm PRS – prymitywna technika optymalizacji globalnej**

**II Średnie wyniki**

* **Funkcja Rosenbrocka**
* **2 wymiary**

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Na podstawie otrzymanych średnich wyników dla funkcji Rosenbrocka można zauważyć, że algorytm MS radzi sobie bardzo dobrze (wyniki są bardzo małe), zaś PRS nie jest już tak dokładny, ale nadal są to niewielkie wartości.

* **10 wymiarów**



Na podstawie otrzymanych średnich wyników dla funkcji Rosenbrocka można zauważyć, że algorytm MS nadal radzi sobie bardzo dobrze (wyniki są bardzo małe), zaś PRS jest już niedokładny i są to spore wartości.

* **20 wymiarów**



Na podstawie otrzymanych średnich wyników dla funkcji Rosenbrocka można zauważyć, że algorytm MS nadal radzi sobie bardzo dobrze (wyniki są bardzo małe), zaś PRS jest już niedokładny i są to spore wartości.

* **Funkcja Michalewicza**
* **2 wymiary**

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Na podstawie otrzymanych średnich wyników dla funkcji Michalewicza można zauważyć, że oba algorytmy radzą sobie bardzo dobrze i mają stosunkowo zbliżony wynik.

* **10 wymiarów**

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Można zauważyć, że pojawiła się widoczna różnica między wynikami obu algorytmów.

* **20 wymiarów**

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Na podstawie otrzymanych średnich wyników dla funkcji Michalewicza można zauważyć, że ponownie niższe wyniki są uzyskiwane w przypadku algorytmu MS.

**III Analiza histogramów oraz wykresów pudełkowych**

* **Funkcja Rosenbrocka**
* **2 wymiary**

**Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie**Obraz zawierający tekst, zrzut ekranu, diagram

Opis wygenerowany automatycznie

W przypadku histogramu można zauważyć, że algorytm MS poradził sobie bardzo dobrze.

Wykres pudełkowy potwierdza postawione wcześniej tezy, że algorytm MS ma wyniki bardziej skupione wokół mniejszych wartości.

* **10 wymiarów**

**Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie**Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie

W przypadku histogramu można zauważyć, że algorytm MS poradził sobie niesamowicie dobrze, zaś algorytm PRS już rozjechał się znacząco. Wykres pudełkowy potwierdza postawione wcześniej tezy oraz pokazuje różność wyników algorytmu PRS.

* **Obraz zawierający tekst, zrzut ekranu, diagram, wyświetlacz

  Opis wygenerowany automatycznie20 wymiarów**

**Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie**

W przypadku histogramu można zauważyć, że algorytm MS poradził sobie niesamowicie dobrze, zaś algorytm PRS już rozjechał się znacząco. Wykres pudełkowy potwierdza postawione wcześniej tezy oraz pokazuje różność wyników algorytmu PRS.

* **Funkcja Michalewicza**
* **2 wymiary**

**Obraz zawierający tekst, zrzut ekranu, wyświetlacz, diagram

Opis wygenerowany automatycznie**Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie

W przypadku histogramu można zauważyć, że algorytm PRS poradził sobie bardzo dobrze i ma większość wyników skoncentrowanych między -1.80 a -1.75. Wykres pudełkowy pokazuje zaś, że wyniki są zdecydowanie bardziej rozrzucone w przypadku algorytmu PRS.

* **10 wymiarów**

Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, diagram

Opis wygenerowany automatycznie

W przypadku histogramu można zauważyć, że algorytm PRS poradził sobie nadal dobrze i ma rozłożone wyniki w odróżnieniu od algorytmu MS, który ma większość wyników skoncentrowanych między -7.5 a 6.5. Wykres pudełkowy potwierdza nam zaś, że wyniki są zdecydowanie bardziej rozrzucone w przypadku algorytmu MS, co się po raz pierwszy pojawiło w naszych rozważaniach.

* **20 wymiarów**

Obraz zawierający tekst, zrzut ekranu, diagram, wyświetlacz

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie

W przypadku histogramu można zauważyć, że algorytm MS ma bardzo podobny rozkład jak algorytm PRS, tylko na różnych wartościach. Wykres pudełkowy pokazuje nam, że znowu wyniki są bardziej rozrzucone w przypadku algorytmu MS.

**IV Analiza istotności statystycznej między średnim wynikiem obu algorytmów**

* **Funkcja Rosenbrocka**
* **2 wymiary**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Istnieje znacząca różnica między wynikami obu algorytmów, co widać w dokładności przedziałów ufności.

* **10 wymiarów**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie **Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

Istnieje znacząca różnica między wynikami obu algorytmów, co widać w dokładności przedziałów ufności.

* **20 wymiarów**

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

Istnieje znacząca różnica między wynikami obu algorytmów, co widać w dokładności przedziałów ufności.

* **Funkcja Michalewicza**
* **2 wymiary**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Istnieje widoczna różnica między wynikami obu algorytmów, co widać w dokładności przedziałów ufności, jednak nie jest ona tak wielka jak w przypadku funkcji Rosenbrocka.

* **10 wymiarów**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Po raz pierwszy przedział ufności ma mniejszy rozstęp w przypadku algorytmu PRS, zaś algorytm MS uzyskiwał znaczące niższe wartości.

* **20 wymiarów**

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

Znów przedział ufności posiada mniejszy rozstęp w przypadku algorytmu PRS, zaś algorytm MS uzyskuje całkiem sporo niższe wartości.

**V Implementacja projektu**

*library*("ParamHelpers")  
*library*("checkmate")  
*library*("smoof")  
*library*("ecr")  
*library*("ggplot2")  
  
dim <- 20  
fun <- makeMichalewiczFunction  
  
#Metoda wielokrotnego startu MS  
multi\_start\_function <- function(function\_object, dimensions, lower\_bounds, upper\_bounds, iterations = 100, runs = 50) {  
 total\_calls <- 0  
 scores <- *numeric*(runs)  
 for (run in 1:runs) {  
 smallest\_score <- Inf  
 calls <- 0  
 for (i in 1:iterations) {  
 point <- *runif*(dimensions, min = lower\_bounds, max = upper\_bounds)  
 optim\_result <- *optim*(point, *function\_object*(dimensions), method = "L-BFGS-B", lower = lower\_bounds, upper = upper\_bounds)  
 calls <- calls + optim\_result$counts[1]  
 if (optim\_result$value < smallest\_score) {  
 smallest\_score <- optim\_result$value  
 }  
 }  
 total\_calls <- total\_calls + calls  
 scores[run] <- smallest\_score  
 }  
 *list*(scores, total\_calls / runs)  
}  
  
#Poszukiwanie przypadkowe PRS  
pure\_random\_search\_function <- function(function\_object, dimensions, lower\_bounds, upper\_bounds, iterations, runs = 50) {  
 scores <- *numeric*(runs)  
 for (run in 1:runs) {  
 smallest\_score <- Inf  
 function\_dim <- *function\_object*(dimensions)  
 for (i in 1:iterations) {  
 point <- *runif*(dimensions, min = lower\_bounds, max = upper\_bounds)  
 current\_score <- *function\_dim*(point)  
 if (current\_score < smallest\_score) {  
 smallest\_score <- current\_score  
 }  
 }  
 scores[run] <- smallest\_score  
 }  
 *list*(scores)  
}  
  
param\_set <- *getParamSet*(*fun*(dim))  
lower\_bounds <- *getLower*(param\_set)  
upper\_bounds <- *getUpper*(param\_set)  
ms\_results <- *multi\_start\_function*(fun, dim, lower\_bounds, upper\_bounds)  
prs\_results <- *pure\_random\_search\_function*(fun, dim, lower\_bounds, upper\_bounds, iterations = ms\_results[[2]])  
*print*(*paste*("MS average min for",dim, "dimensions:", *mean*(ms\_results[[1]])))  
*print*(*paste*("PRS average min for",dim, "dimensions:", *mean*(prs\_results[[1]])))  
*par*(mfrow = *c*(1, 2))  
*hist*(ms\_results[[1]], main = "Histogram wynikow MS", xlab = "Wyniki", ylab = "Czestosci", col = "green")  
*hist*(prs\_results[[1]], main = "Histogram wynikow PRS", xlab = "Wyniki", ylab = "Czestosci", col = "pink")  
*par*(mfrow = *c*(1, 1))  
wyniki <- *list*(MS = ms\_results[[1]], PRS = prs\_results[[1]])  
*boxplot*(wyniki, col = *c*("green", "pink"), main = "Wykres pudelkowy wynikow PRS i MS",  
 xlab = "Metoda", ylab = "Wynik")  
test\_result <- *t.test*(ms\_results[[1]], conf.level = 0.95)  
*print*(test\_result)  
test\_result <- *t.test*(prs\_results[[1]], conf.level = 0.95)  
*print*(test\_result)