Raport projektu zaliczeniowego

Autor: Adrian Żerebiec

Data wykonania: 24.01.2023

Wybrałem dwa algorytmy spośród zaproponowanych:

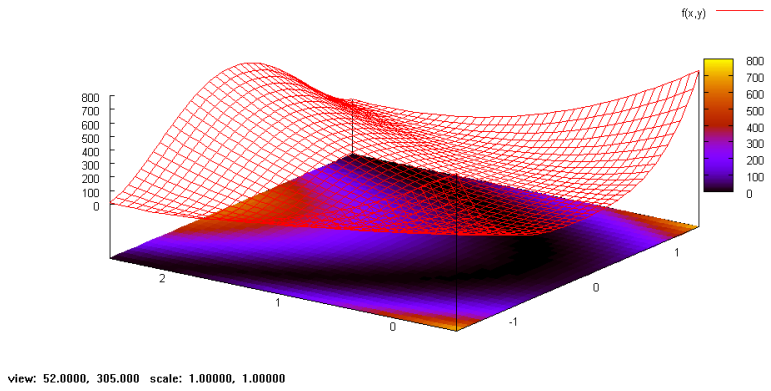
- Poszukiwanie przypadkowe (Pure Random Search)

- Wielokrotny start (Multi-start)

Oraz dwie funkcje z pakietu Smoof:

- Funkcję Rosenbrocka

$$f(x) = \sum_{i=1}^{N-1} \left[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2\right] \quad \forall x \in \mathbb{R}^N.$$

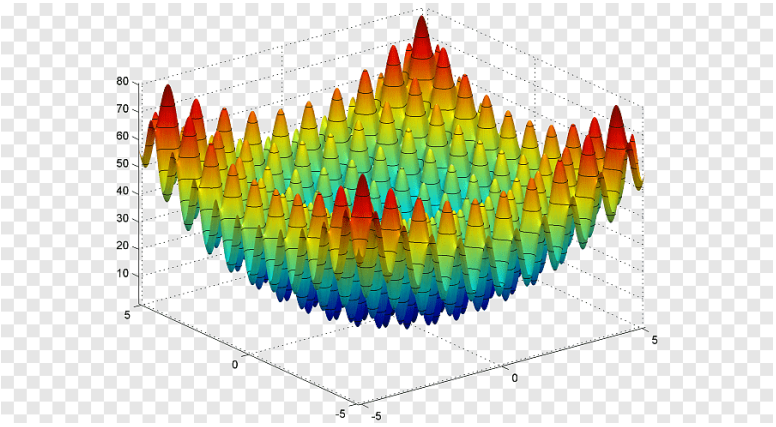


Krótki opis:

Funkcja Rosenbrocka funkcja niewypukła używana w optymalizacji jako test dla algorytmów optymalizacji. Zwana jest też ze względu na swój kształt „doliną Rosenbrocka” lub „funkcją bananową Rosenbrocka”.Funkcja ta jest popularnie używana do przedstawiania zachowań algorytmów optymalizacji. Minimum globalne funkcji znajduje się wewnątrz długiego, parabolicznego wgłębienia funkcji.

Funkcję Ackleya

$$f(x,y) = -20\exp\left[-0.2\sqrt{0.5(x^2+y^2)}\right] - \exp[0.5\,(\cos 2\pi x + \cos 2\pi y)] + e + 20$$



Krótki opis:

W optymalizacji matematycznej funkcja Ackleya jest funkcją niewypukłą używaną jako problem testu wydajności dla algorytmów optymalizacji. Został on zaproponowany przez Davida Ackleya w jego rozprawie doktorskiej z 1987 roku.

Poszukiwanie przypadkowe:

Pure Random Search zdefiniowałem następująco:

```r
PRS <- function(dims, number_of_points, fn){
  number_of_dimensions = nrow(dims)

  points <- matrix(runif(number_of_dimensions*number_of_points), ncol=number_of_dimensions)
  points <- apply(points, 2, function(x) x*(dims[,2]-dims[,1])+dims[,1])

  vals <- apply(points, 1, fn)
  min_index <- which.min(vals)
  min_val <- vals[min_index]
  return(min_val)
}
```

Funkcja ta służy do znajdowania minimum funkcji w danym zbiorze wymiarów. Generuje ona macierz punktów wybranych losowo w zadanym zbiorze wymiarów i następnie oblicza wartość funkcji dla każdego z punktów. Znajduje ona minimum funkcji i zwraca jego wartość.

Wielokrotny start:

Multi-start został zdefiniowany następująco:

```r
MS <- function(dims, number_of_points, fn){
  num_dims = nrow(dims)
  minimum_point <- rep(0, num_dims)
  minimum <- Inf

  for(i in 1:number_of_points){
    p <- runif(num_dims, dims[1,1], dims[1,2])
    p <- optim(par = p, fn = fn, method = "L-BFGS-B", lower = dims[,1], upper = dims[,2])$par
    val <- fn(p)
    if (val < minimum){
      minimum_point <- p
      minimum <- val
    }
  }
  return(minimum)
}
```

Funkcja ta również służy do znajdowania minimum funkcji w danym zbiorze wymiarów. Generuje ona losowe punkty w danym zbiorze wymiarów i wykorzystuje algorytm Neldera-Meada do znalezienia punktu minimum. Następnie zwraca wartość funkcji w tym punkcie.

Funkcja określająca wymiary:

```r
get_bounds <- function(f, number_of_dimensions){
  lower = attr(f,"par.set")$pars$x$lower
  upper = attr(f,"par.set")$pars$x$upper
  array(c(lower, upper), c(number_of_dimensions,2))
}
```

Funkcja ta jest używana do pobierania wymiarów funkcji celu. Funkcja ta przyjmuje funkcję celu i liczbę wymiarów jako argumenty. Następnie pobiera górne i dolne granice funkcji celu ze zestawu parametrów funkcji celu. Na końcu zwraca tablicę dwuwymiarową składającą się z górnych i dolnych granic dla każdego wymiaru.

Funkcje obliczające: średnie, odchylenie standardowe oraz przedziały ufności

```r
get_mean <- function(values) {
  mean <- mean(values)
  return(mean)
}

get_standard_deviation <- function(values) {
  sd <- sd(values)
  return(sd)
}

get_confidence_interval <- function(values, mean, sd, number_of_runs) {
  confidence <- c(mean - qnorm(.975)*sd/sqrt(number_of_runs), mean + qnorm(.975)*sd/sqrt(number_of_runs))
  return(confidence)
}

get_budget <- function(calls_count, number_of_runs, number_of_points) {
  budget <- calls_count %/% number_of_runs - number_of_points
  return(budget)
}

get_prs_num_runs <- function(number_of_runs, budget, number_of_points) {
  prs_num_runs <- (number_of_runs * budget) %/% number_of_points
  return(prs_num_runs)
}
```

Co więcej mamy także funkcję definiującą budżet dla PRS oraz liczbę wykonań tego algorytmu.

Główna funkcja

Funkcja ta jest używana do wygenerowania wyników pomiarowych ze wspomnianych wyżej algorytmów optymalizacji. Funkcja ta przyjmuje liczbę przebiegów, liczbę punktów, liczbę wymiarów, funkcję celu i nazwę funkcji jako argumenty. Następnie tworzy funkcję z licznikiem, aby zliczyć liczbę wywołań funkcji celu. Następnie generuje wyniki przebiegu algorytmów. Funkcja oblicza średnie, odchylenie standardowe i przedział ufności dla wyników Monte Carlo i algorytmu przeszukiwania położenia, a także budżet wywołań funkcji celu. Na końcu funkcja zwraca listę zawierającą średnie, odchylenie standardowe i przedział ufności dla obu algorytmów.

```r
get_results <- function(number_of_runs, number_of_points, number_of_dimensions, tested_function, function_name){
  calls_count <- 0
  tetster_function <- function(x) {
    calls_count <<- calls_count + 1
    tested_function(x)
  }

  dimensions <- get_bounds(tested_function, number_of_dimensions)
  calls_count <- 0

  ms <- get_ms(number_of_runs, dimensions, number_of_points, function_name, tetster_function)
  ms_mean <- get_mean(ms)
  ms_sd <- get_standard_deviation(ms)
  ms_confidence <- get_confidence_interval(ms, ms_mean, ms_sd, number_of_runs)
  budget <- get_budget(calls_count, number_of_runs, number_of_points)

  prs_num_runs <- get_prs_num_runs(number_of_runs, budget, number_of_points)
  prs <- get_prs(prs_num_runs, dimensions, number_of_points,function_name, tested_function)
  prs_mean <- get_mean(prs)
  prs_sd <- get_standard_deviation(prs)
  prs_confidence <- get_confidence_interval(prs, prs_mean, prs_sd, prs_num_runs)

  #testing <- t.test(prs, ms)
  return(list(ms_mean = ms_mean, prs_mean = prs_mean, ms_sd=ms_sd, prs_sd=prs_sd,
              ms_confidence=ms_confidence, prs_confidence=prs_confidence))
}
```

Podstawowy plik z wywołaniem

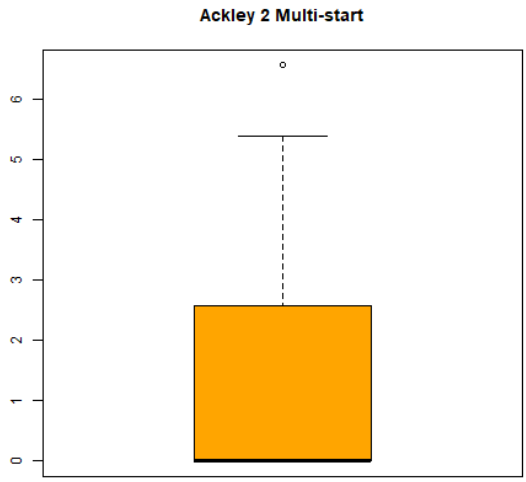
```r
nuberm_of_dimensions_list <- c(2, 10, 20)
number_of_runs <- 50
number_of_points <- 100

# Ackley function
cat("Ackley funtion:\n\n")
file.remove("SolutionAckley.txt")
for (number_of_dimensions in nuberm_of_dimensions_list){
  ackley_function <- makeAckleyFunction(number_of_dimensions)
  result <- get_results(number_of_runs,number_of_points, number_of_dimensions, ackley_function, "Ackley")
  cat("dimensions:", number_of_dimensions, "\n",
      "ms_mean:", result$ms_mean, "\n",
      "ms_sd:", result$ms_sd, "\n",
      "ms_confidence:", result$ms_confidence, "\n",
      "prs_mean:", result$prs_mean, "\n",
      "prs_sd:", result$prs_sd, "\n",
      "prs_confidence:", result$prs_confidence, "\n",
      "difference:", abs(result$ms_mean - result$prs_mean), "\n\n",
      file = "SolutionAckley.txt", append = TRUE)
}
# Rosenbrock function
cat("Rosenbrock funtion:\n\n")
file.remove("SolutionRosenbrock.txt")
for (number_of_dimensions in nuberm_of_dimensions_list){
  rosenbrock_function <- makeRosenbrockFunction(number_of_dimensions)
  result <- get_results(number_of_runs, number_of_points, number_of_dimensions, rosenbrock_function, "Rosenbrock")

  cat("dimensions:", number_of_dimensions, "\n",
      "ms_mean:", result$ms_mean, "\n",
      "ms_sd:", result$ms_sd, "\n",
      "ms_confidence:", result$ms_confidence, "\n",
      "prs_mean:", result$prs_mean, "\n",
      "prs_sd:", result$prs_sd, "\n",
      "prs_confidence:", result$prs_confidence, "\n",
      "difference:", abs(result$ms_mean - result$prs_mean), "\n\n",
      file = "SolutionRosenbrock.txt", append = TRUE)
}
```

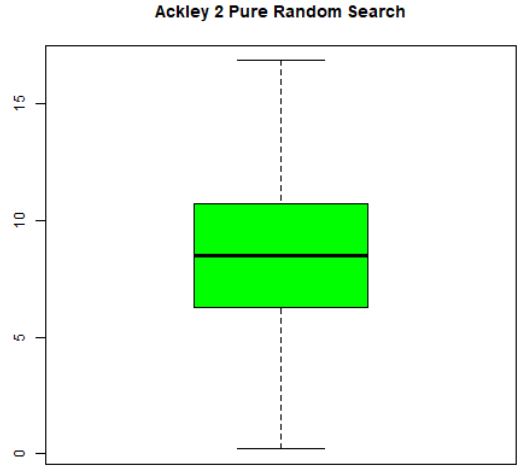
Wyniki działania programu:

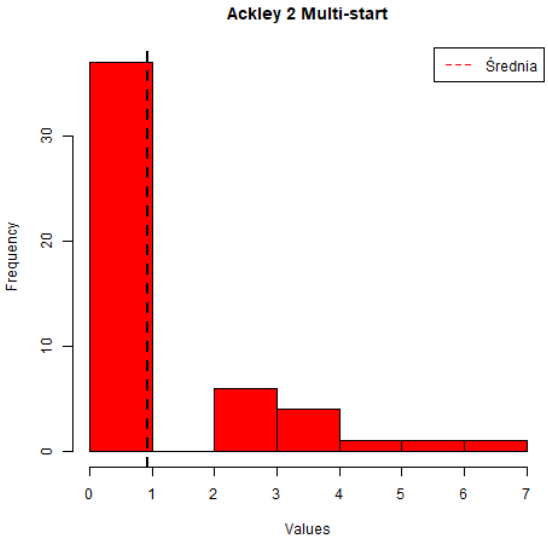
Funkcje Ackleya oraz Rosenbrocka został uruchomione dla wymiarów 2, 10 i 20 oraz dla 100 punktów i 50 uruchomień. Podczas nich zbierałem dane dotyczące średnich, odchyleń, ufności oraz różnicę zdefiniowaną przeze mnie jako wartość absolutna różnic średnich MS oraz PRS. Co więcej tworzyłem w tym samym czasie wykresy oraz histogramy. Dane były zbierane do plików, gdzie możemy zobaczyć jaka np. była średnia dla Ackleya 2D dla PRS. Dodatkowo wykresy oraz histogramy zostały zapisane do katalogów w postaci plików PNG.

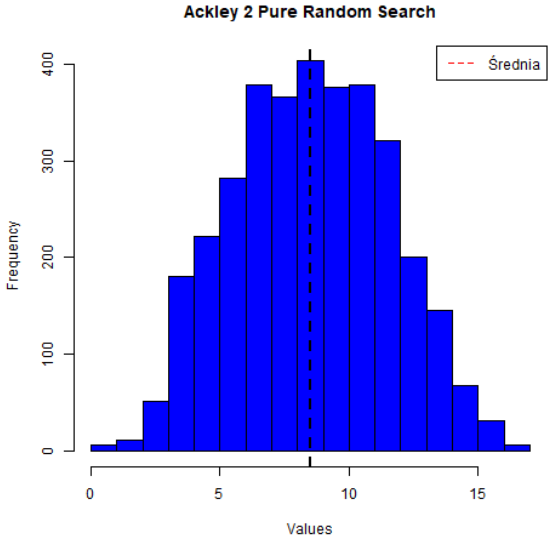
Porównanie dla Ackleya 2D

```
dimensions: 2
ms_mean: 0.932059
ms_sd: 1.711031
ms_confidence: 0.457794 1.406324
prs_mean: 8.522522
prs_sd: 3.005873
prs_confidence: 8.421913 8.623131
difference: 7.590463
```



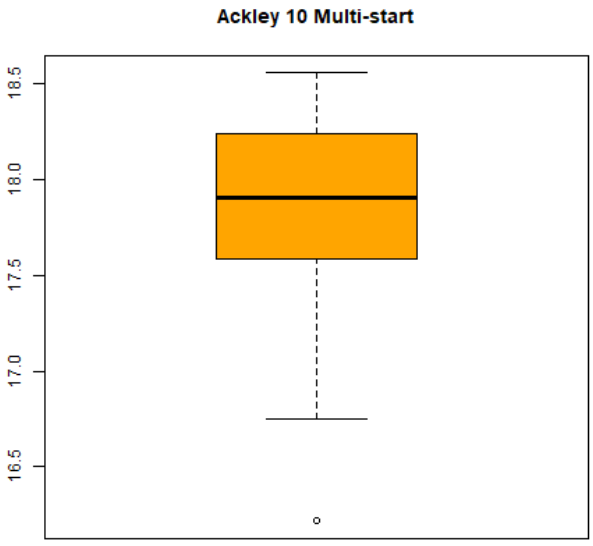


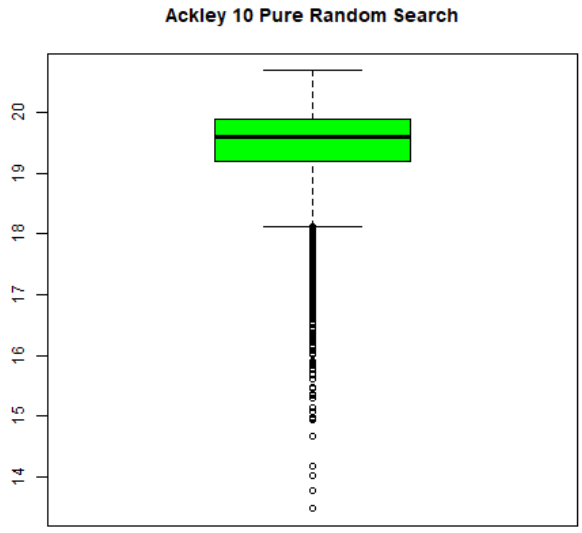


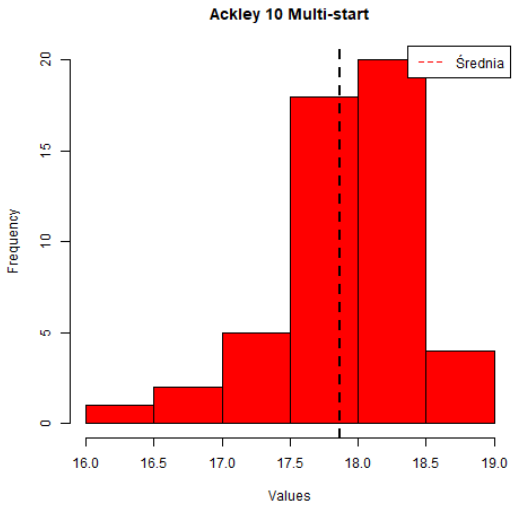


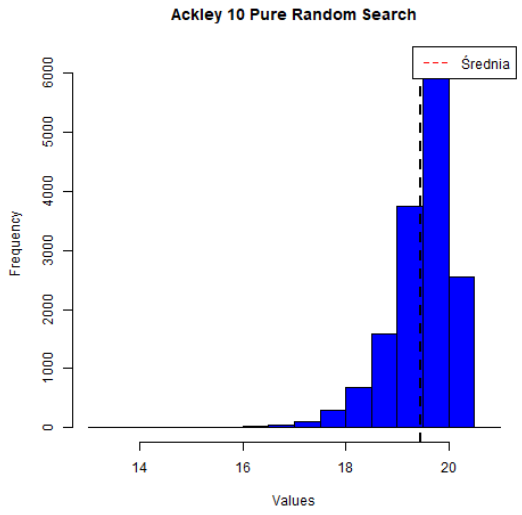
Porównanie dla Ackleya 10D

```
dimensions: 10
 ms_mean: 17.86406
 ms_sd: 0.4970831
 ms_confidence: 17.72628 18.00184
 prs_mean: 19.45927
 prs_sd: 0.6467422
 prs_confidence: 19.44901 19.46952
 difference: 1.595212
```



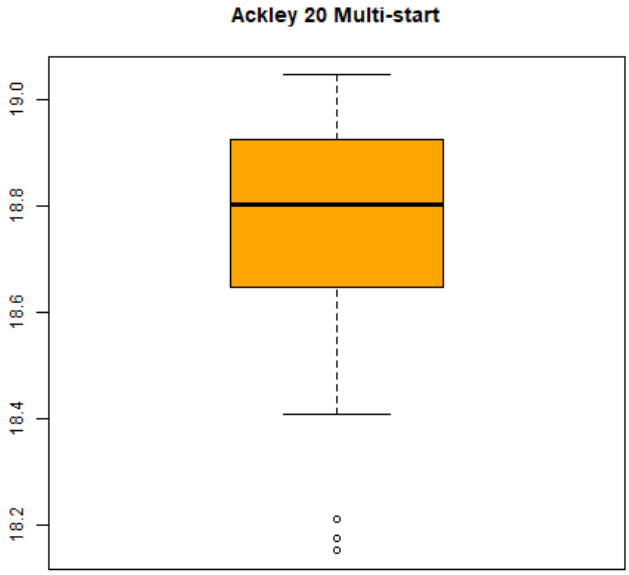


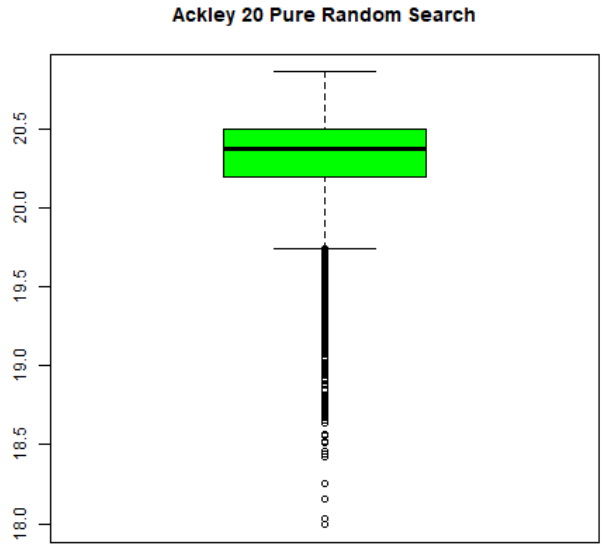


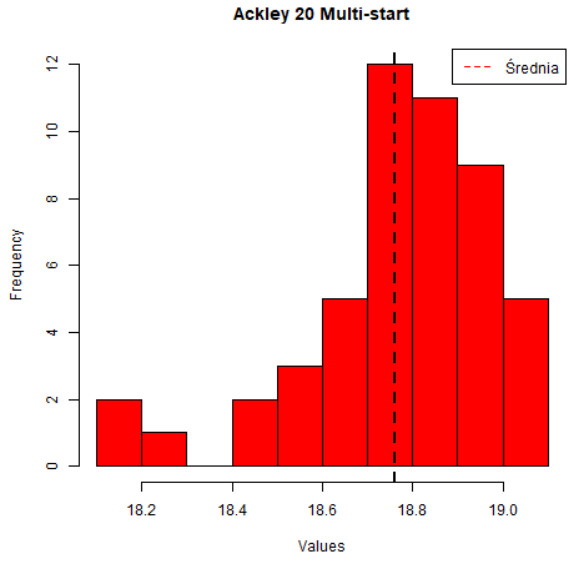


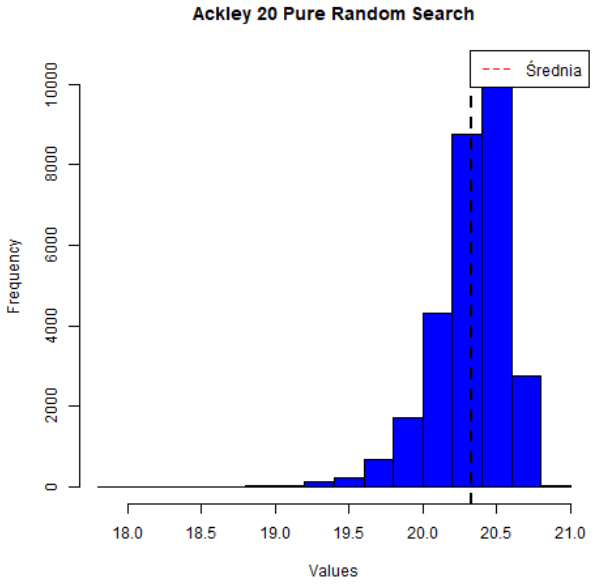
Porównanie dla Ackleya 20D

```
dimensions: 20
ms_mean: 18.76067
ms_sd: 0.2120858
ms_confidence: 18.70189 18.81946
prs_mean: 20.32624
prs_sd: 0.2582978
prs_confidence: 20.32327 20.3292
difference: 1.565562
```



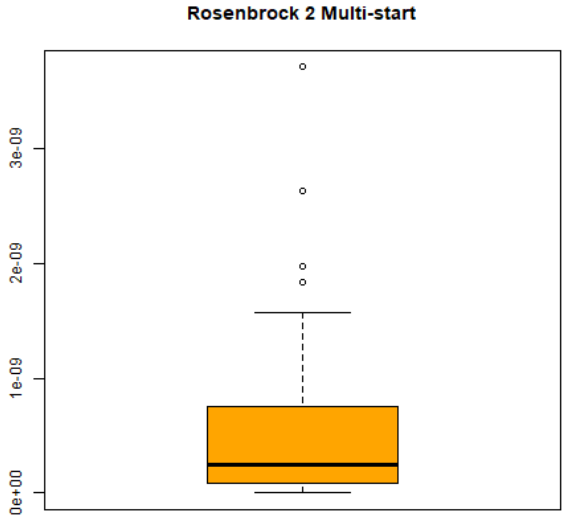


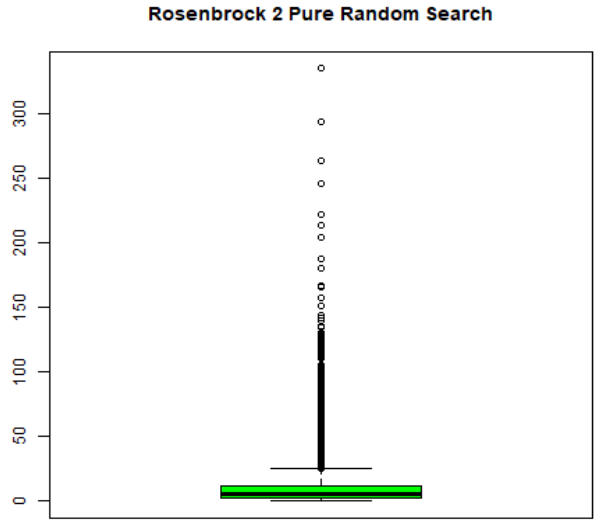


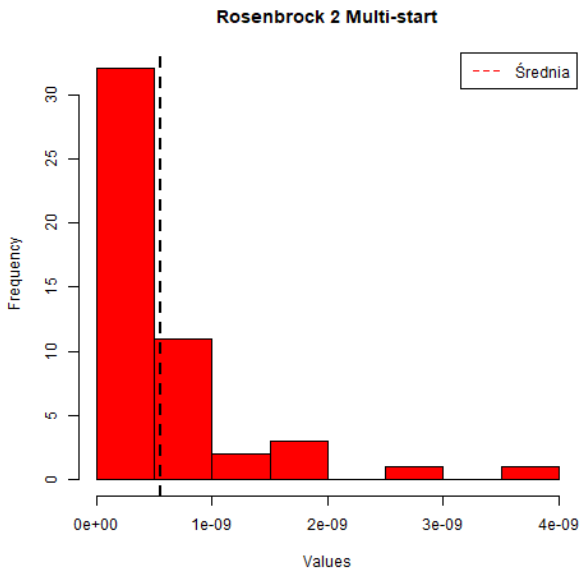


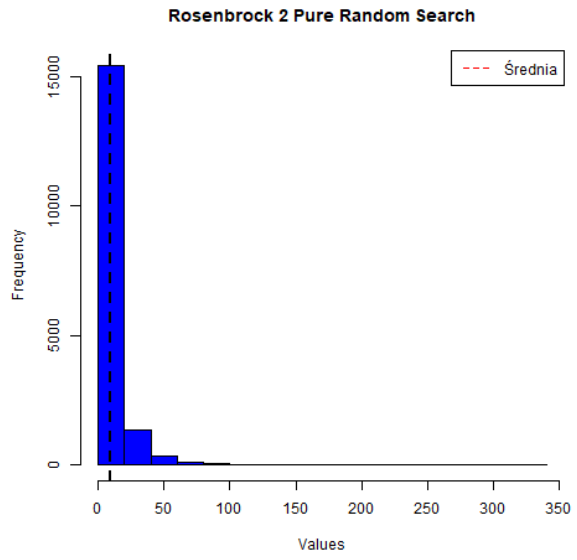
Porównanie dla Rosenbrocka 2D

```
dimensions: 2
  ms_mean: 5.541402e-10
  ms_sd: 7.438106e-10
  ms_confidence: 3.479702e-10 7.603102e-10
  prs_mean: 9.448288
  prs_sd: 14.2989
  prs_confidence: 9.235449 9.661127
  difference: 9.448288
```



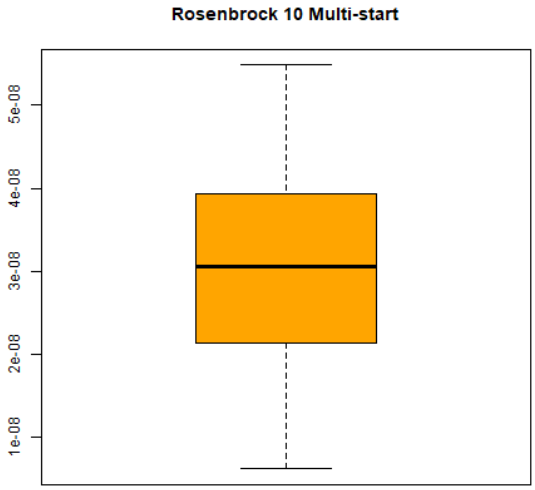


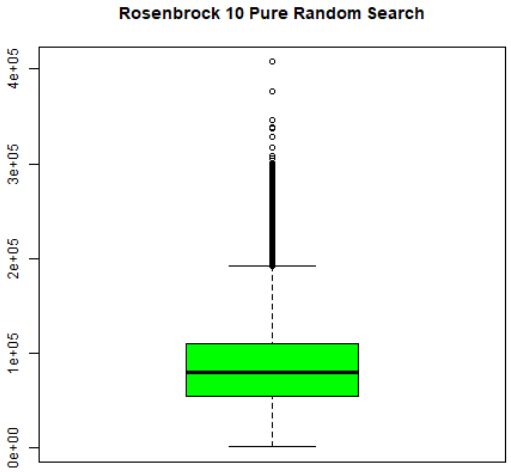


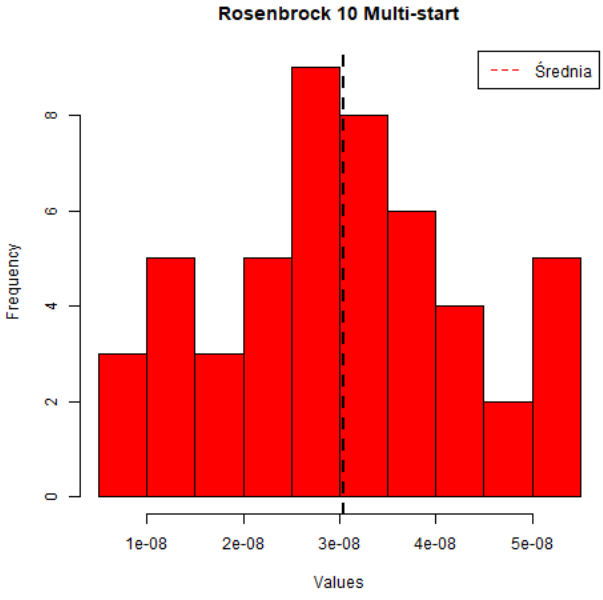


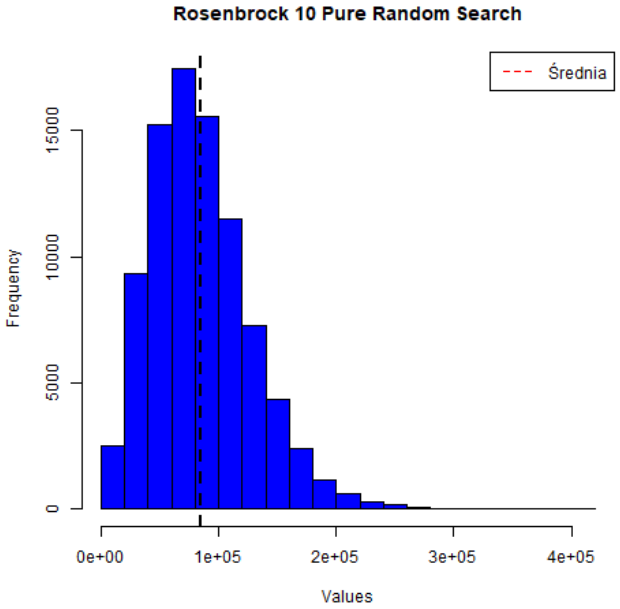
Porównanie dla Rosenbrocka 10D:

```
dimensions: 10
  ms_mean: 3.04489e-08
  ms_sd: 1.301214e-08
  ms_confidence: 2.684218e-08 3.405562e-08
  prs_mean: 84653.26
  prs_sd: 42319.39
  prs_confidence: 84373.43 84933.09
  difference: 84653.26
```



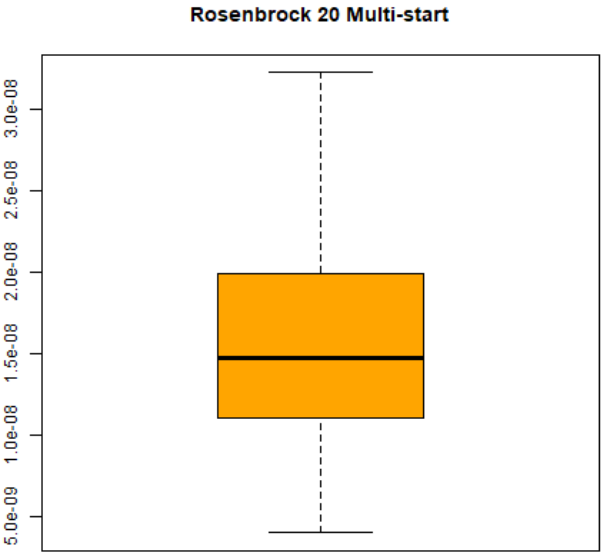


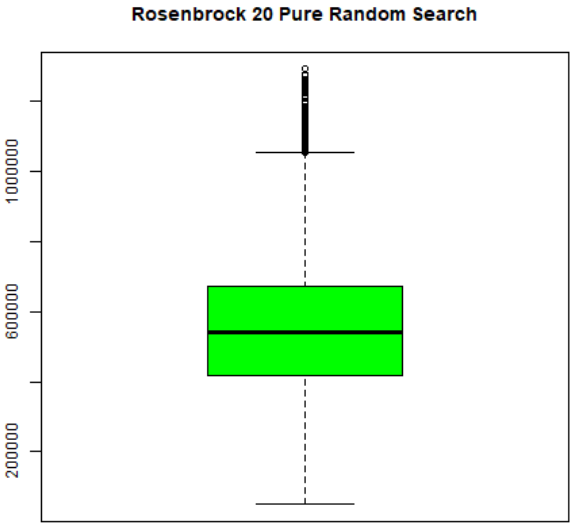


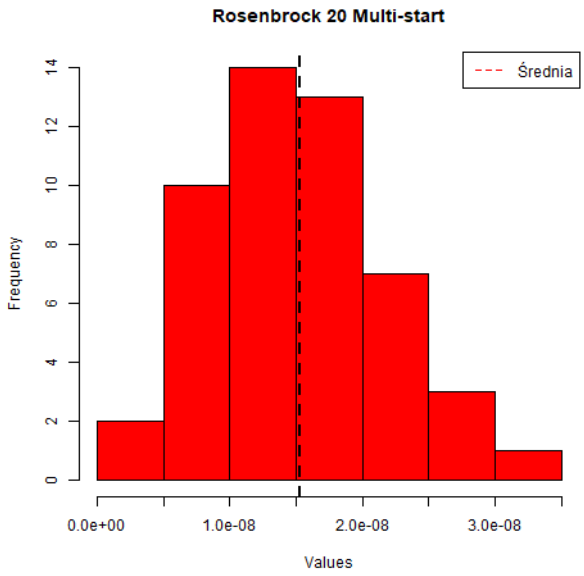


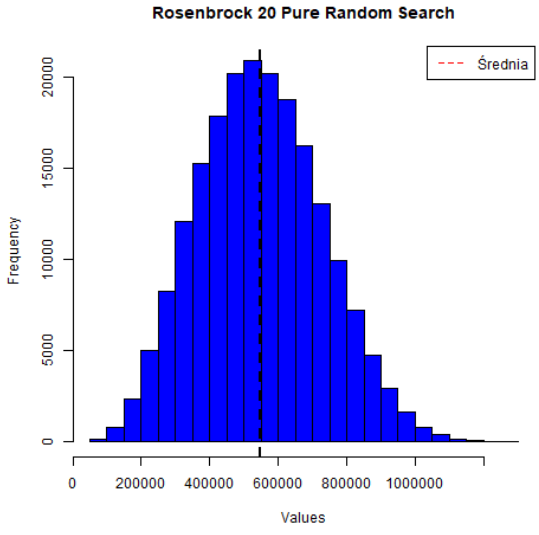
Porównanie dla Rosenbrocka 20D

```
dimensions: 20
ms_mean: 1.532369e-08
ms_sd: 6.389606e-09
ms_confidence: 1.355261e-08 1.709476e-08
prs_mean: 548510.6
prs_sd: 181062.8
prs_confidence: 547715.3 549305.9
difference: 548510.6
```









Analiza statystyczna

Używamy funkcji t.test i konstruujemy 95-procentowe przedziały ufności dla różnic średnich. Dla testów statystycznych za hipotezę zerową przyjmujemy stwierdzenie, że średnie z obu algorytmów są równe. Hipoteza alternatywna to oczywiście stwierdzenie, że są one różne.

Wartości uzyskane dzięki wykorzytsaniu funkcji t.test

Wartość	Rosenbrock 2D	Rosenbrock 10D	Rosenbrock 20D	Ackley 2D	Ackley 10D	Ackley 20D
Mean of PRS	9.339409e+00	8.449998e+04	5.490841e+05	8.574625	19.45217	20.32772
Mean of MS	7.237551e-10	2.920326e-08	1.612208e-08	0.923986	17.92257	18.65230
P Value	<2.2e-16	<2.2e-16	<2.2e-16	<2.2e-16	< 2.2e-16	< 2.2e-16
Confidence	9.127747 - 9.551071	84219.54 - 84780.42	548291.3 - 549876.8	6.93696 - 8.016245	1.511886 -1.87166	1.523916- 1.641636

W powyższej tabelce widać, że p-wartości są znacznie mniejsze od przyjętego poziomu istotności (0.05). Zatem każdą z postawionych hipotez zerowych (mówiących o równości średnich) musimy odrzucić.

Podsumowanie

Powyższe analizy pozwalają nam stwierdzić iż mimo teoretycznych podobieństw PRS jak i MS są kompletnie od siebie różne. Należy zatem dobierać je zależnie od sytuacji, badanej funkcji jak i celów jakie przyświecają nam podczas tworzenia analizy. Dla lepsze analizy moglibyśmy wziąć także inne funkcje do zbadania jak algorytmy sobie z nimi poradzą, gdyż ze sporą pewnością możemy stwierdzić że zależnie od charakterystyki funkcji powinniśmy dobierać algorytm.