Sprawozdanie

Ćw. 11 – Algorytm poszukiwania harmonii (HS)  
Filip Horst 311257

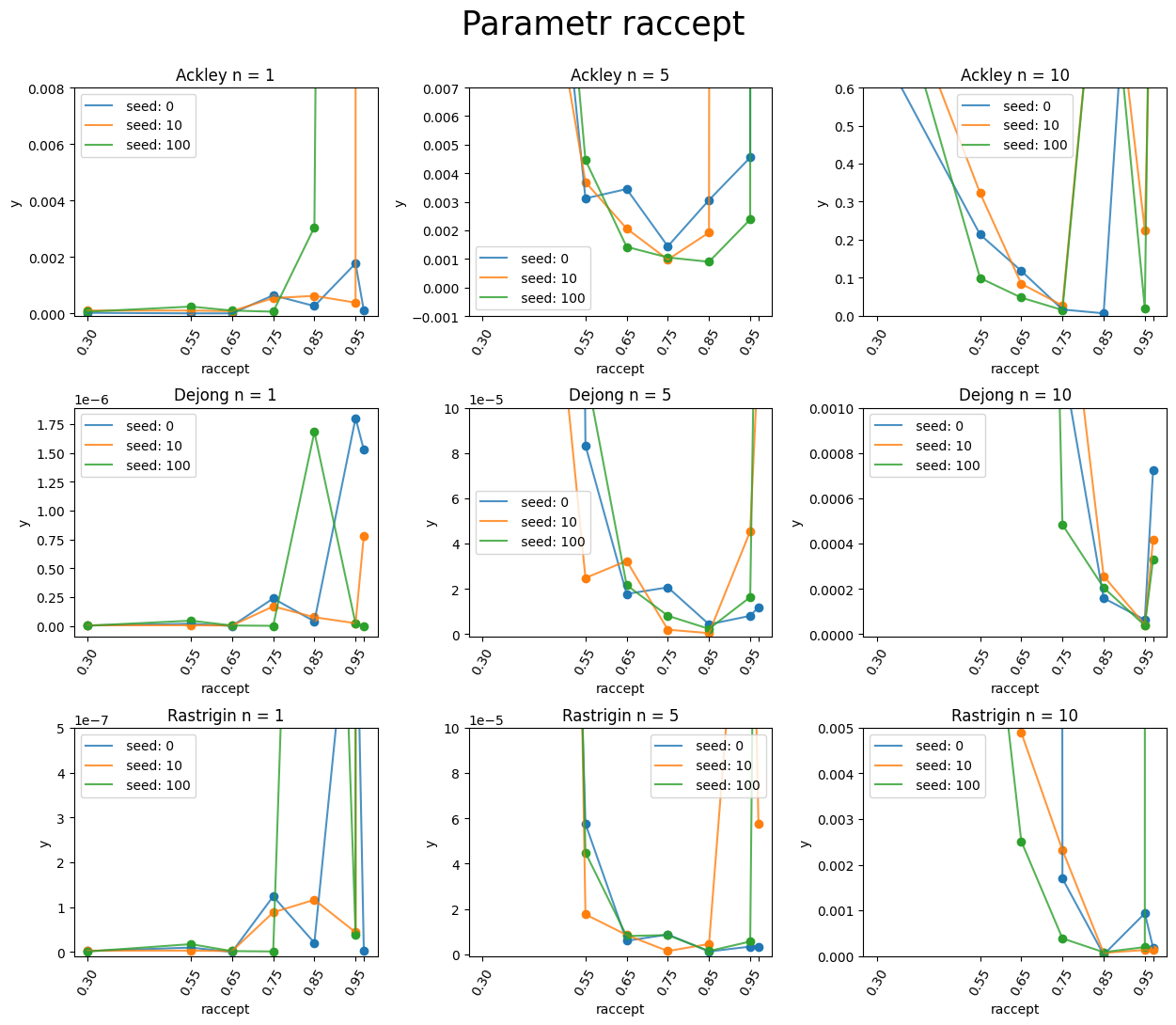
# Analiza zależności parametrów z wynikami – analiza jednoczynnikowa

Pierwszym etapem podobnie jak w poprzednich ćwiczeniach jest analiza jednoczynnikowa. Domyślnymi ustawieniami były:

* Accept\_rate (raccept) 0.95
* Pa\_rate (rpa) 0.7
* Pa\_band\_coef 200
* Popsize 30
* Niter 500

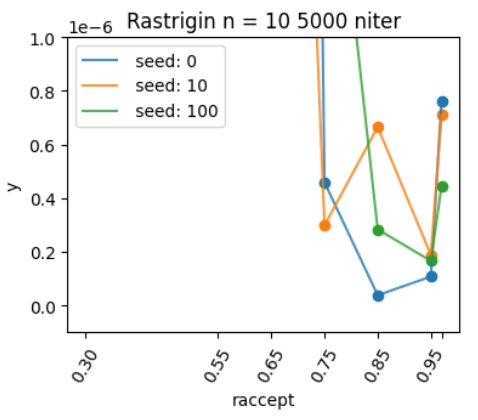
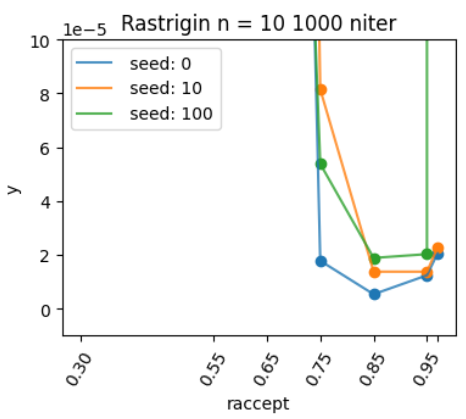
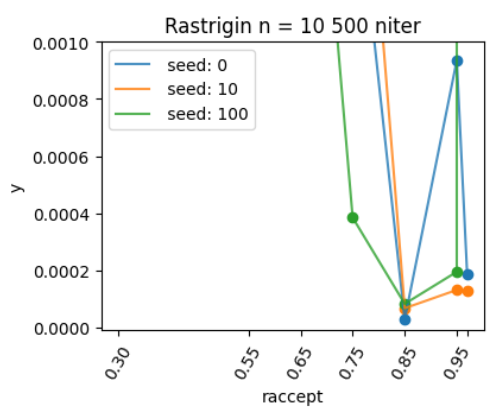
Wszystkie testowane zadania posiadały optimum w punkcie 0, co pozwala na utożsamienie wyniku y z błędem (większe y = większy błąd). Na wykresach przedstawione zostały zależności wyników od ustawień dla trzech różnych ziaren losowych – takie podejście jest trudniejsze w wizualizacji i analizie, ale w przeciwieństwie do użycia miar statystycznych nie trzeba się obawiać, że jakaś zależność zostanie przypadkowo ukryta (jak przy użyciu średniej), bądź utrudniona w analizie (jak przy użyciu mediana + odchylenie itp.).

## Accept\_rate (raccept)



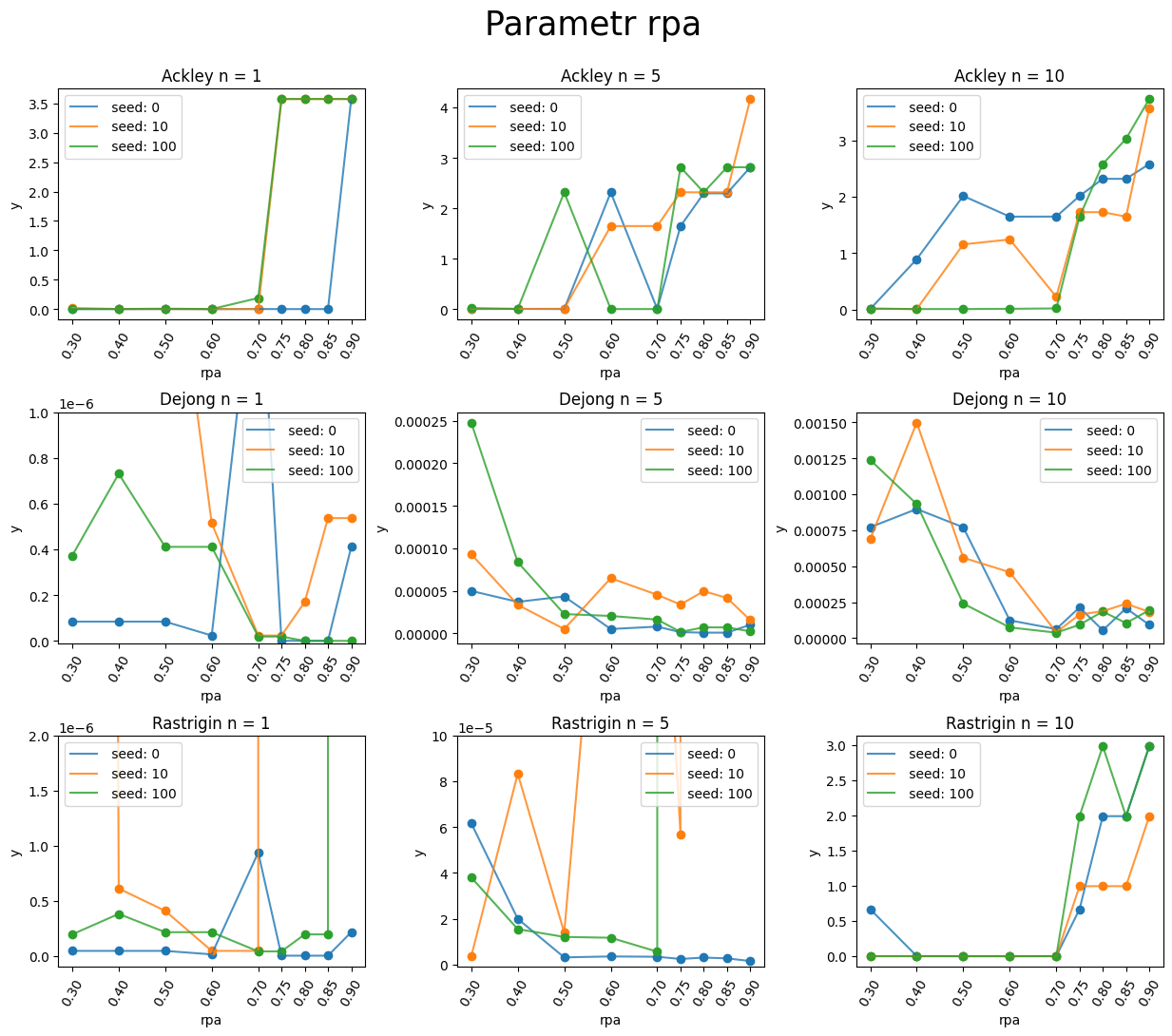
We wszystkich przypadkach oprócz 1-wymiarowej f. Rastrigina zależność wyniku od ustawień raccept przejawia charakter paraboliczny w tym sensie, że dla zbyt małych i zbyt dużych ustawień wynik zaczyna się pogarszać – istnieje pewien przedział dobrych wyników będący kompromisem. Przedział ten jest różny dla konkretnych zadań (inna szerokość i umiejscowienie), ale można powiedzieć, że najbardziej uniwersalnym ustawieniem było 0.85. Inną obserwacją jest to, że dla funkcji o mniejszych wymiarowościach lepiej dobrze sprawdzały się również wartości niższe od 0.85 np. 0.75, natomiast dla 10 wymiarowych funkcji Rastrigina i DeJonga dobre okazywały się wartości wyższe do 0.95 (0.97 już było bardzo słabe dla każdego przypadku). Ta zależność nie jest jednak w pełni zachowana dla 10-wymiarowej funkcji Ackley’a.

Potencjalną hipotezą jest to, że dla łatwiejszych zadań optymalizator jest w stanie dojść w okolice rozwiązania niezależnie od ustawienia raccept (pomijając bardzo skrajne wartości jak np. 0.3), a wtedy zwiększona eksploracja na skutek większej losowości pomaga znaleźć jeszcze lepsze rozwiązanie na bardzo małym obszarze. Pewnym potwierdzeniem tej hipotezy są rzędy wielkości osi Y na wykresach powyżej. Dla łatwiejszych zadań, gdzie skuteczniejsze są niższe ustawienia y są rzędu 1e-5, bądź nawet niższe, podczas gdy dla trudniejszych wyniki są dziesiątki, a nawet setki razy wyższe.

  
Zależność wyniku y od raccept przy różnych niter dla f. Rastrigina n=10.

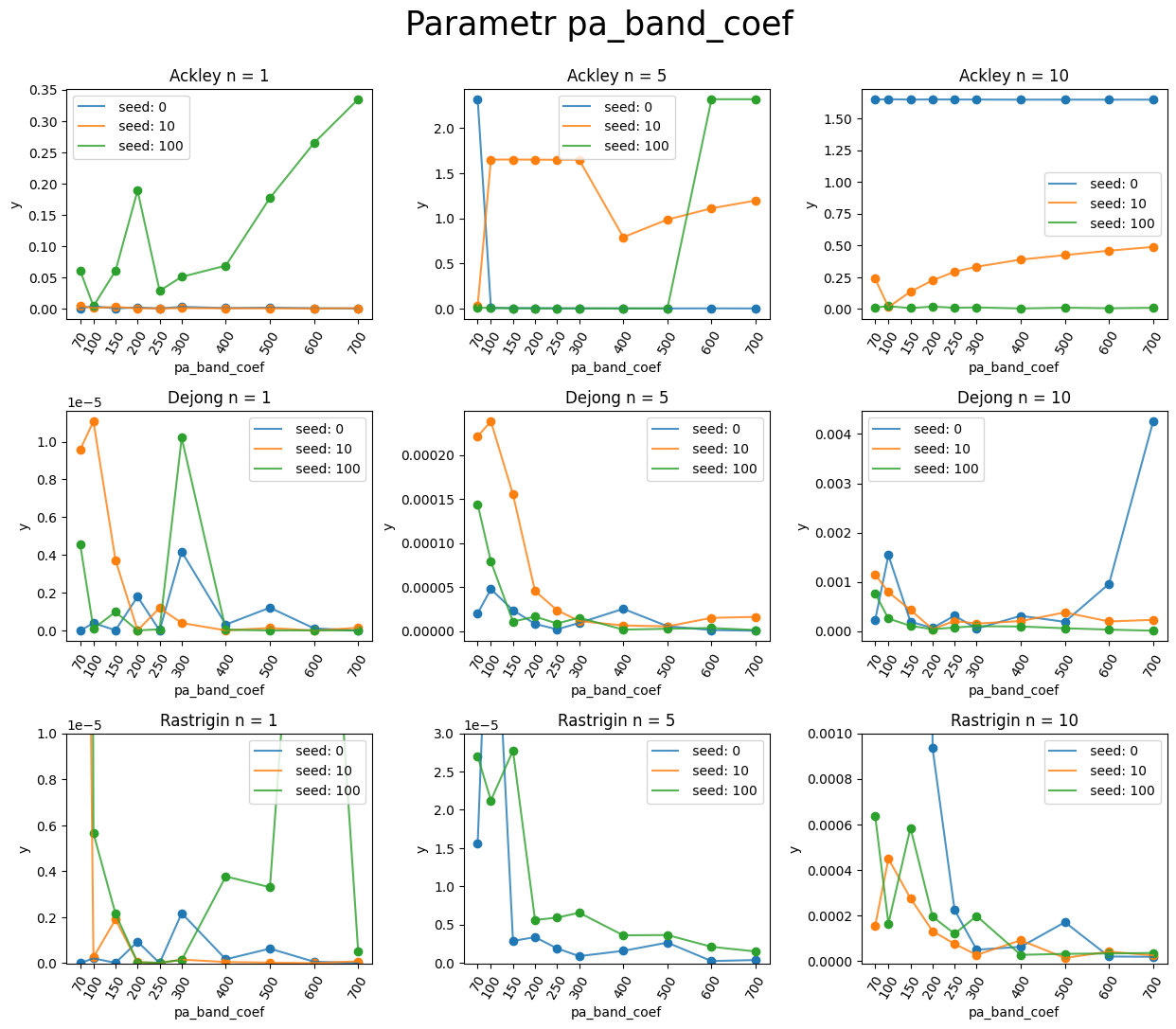
Dodatkowym badaniem było sprawdzenie jakości wyników w różnej liczbie iteracji. Obserwacje są częściowo przeciwne wstępnej hipotezie, ponieważ nawet dla bardzo dużej liczby iteracji charakter wykresu się nie zmienia. Oznacza to, że na optymalną wartość accept\_rate ma przede wszystkim kształt optymalizowanej funkcji.

## Pa\_rate (rpa)



Parametr pa\_rate podobnie ma pewien optymalny zakres wartości. W tym przypadku jednak od razu w oczy rzuca się duża zależność od zadania – np. dla 10 wymiarowych funkcji DeJonga i Rastrigina w pierwszym przypadku występuje zależność w stylu „im więcej tym lepiej” (do wartości ok. 0.7), natomiast dla Rastrigina „im więcej, tym gorzej” (dla wartości > 0.7). Można to zinterpretować jako to, że w funkcji DeJonga lepiej sprawdza się podejście z dodatkową ekploracją, natomiast dla funkcji Rastrigina lepsze jest spokojniejsze podejście rzadko losujące nowe nuty. Najlepiej jest więc wykonać wstępne przeszukanie ogólne dla wykonywanego zadania, by sprawdzić jakie są jego właściwości.

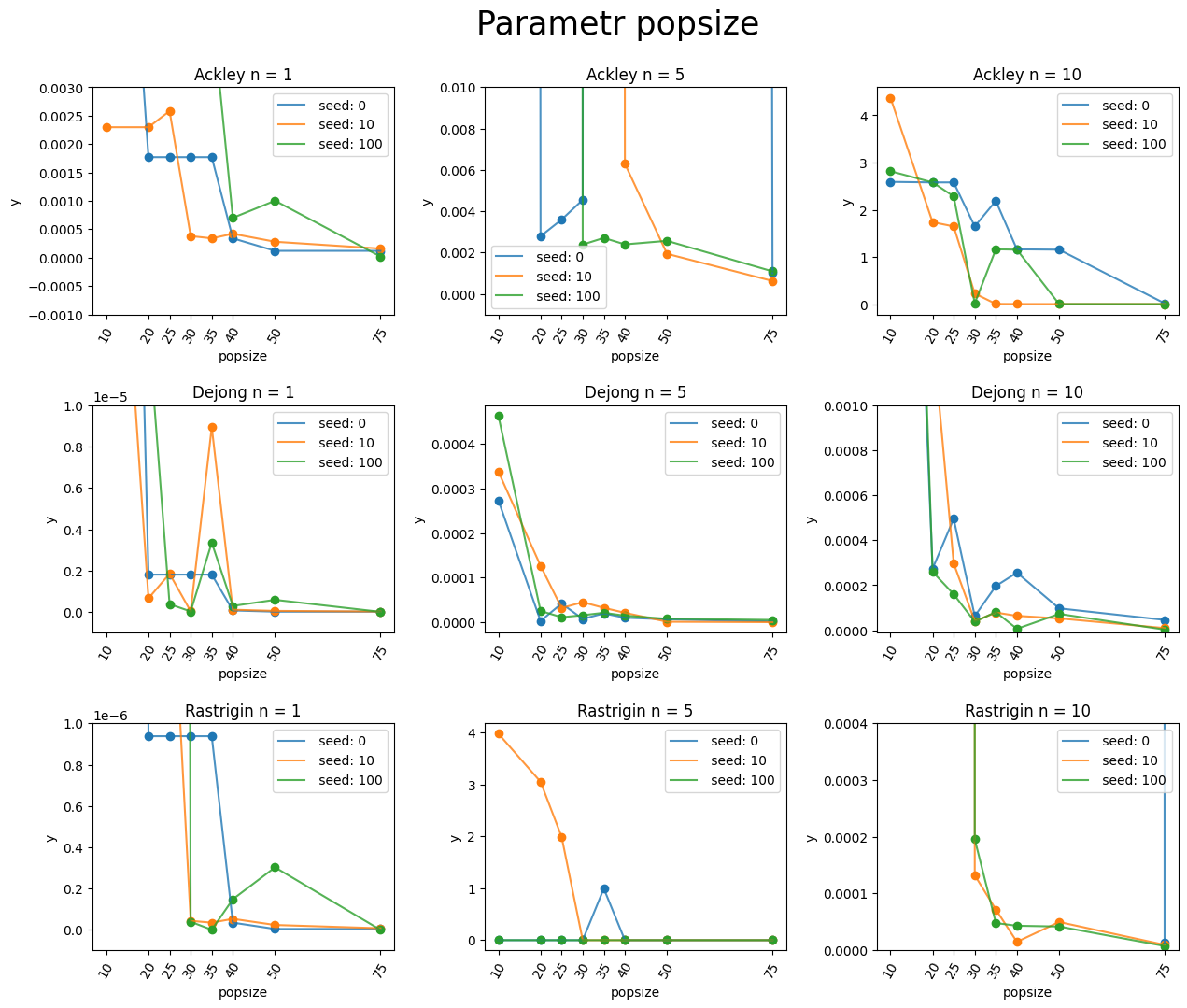
## Pa\_band\_coef



W przypadku tego parametru większość zadań przejawia zależność typu „im więcej, tym lepiej”. Istnieją jednak przypadki, gdzie zwiększanie pogarsza Y (np. Dejong n=10). Jeśli jednak występują takie przypadki to tylko dla pojedynczych ziaren losowych. Jest to logiczne, ponieważ wyższy pa\_band\_coef można utożsamiać z mniejszych krokiem zmian, a więc niefortunny strzał na początku (zależny od ziarna) sprawi, że dla mniejszych kroków harmonia nie zdąży się zbiec do dobrego wyniku w czasie ograniczonym iteracjami.

Podsumowując, wyższe wartości parametru dadzą lepsze wyniki, ale trzeba zachować umiar by nie spowodować zbyt wolnej zbieżności. Można by się było zastanowić nad opłacalnością podejścia hybrydowego, gdzie współczynnik maleje z kolejnymi iteracjami.

## Popsize



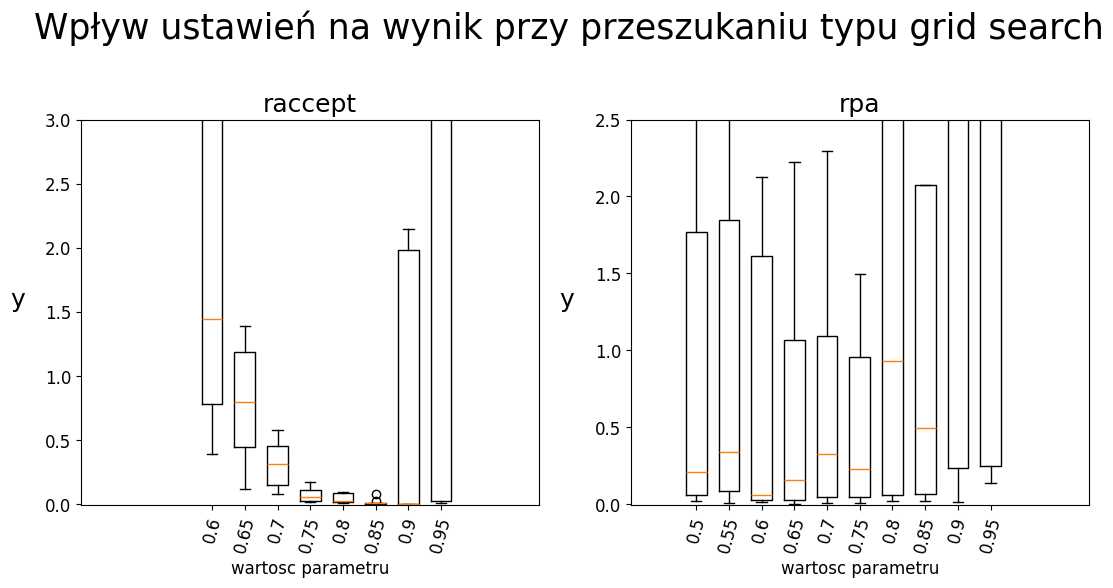
W tym przypadku liczebność populacji zdaje się mieć po jednoznacznie pozytywny wpływ na wyniki. Dla pewnych wartości widać jednak znaczny spadek opłacalności dalszego zwiększania (mała poprawa i wydłużony czas wykonywania), dlatego nie powinno się na ślepo ustawiać wartości maksymalnej.

## Niter

Podobnie jak w każdym z poprzednio badanych algorytmów większa liczba iteracji pomagała osiągnąć lepsze wyniki, ale w pewnym momencie poprawa w stosunku do wydłużonego czasu obliczeń stawała się nieopłacalna. Z powodu prostoty tego parametru wykresy zostały pominięte.

# Podstawowa analiza wieloczynnikowa

Ta analiza obejmuje prosty grid search parametrów accept\_rate (raccept) i pa\_rate (rpa).

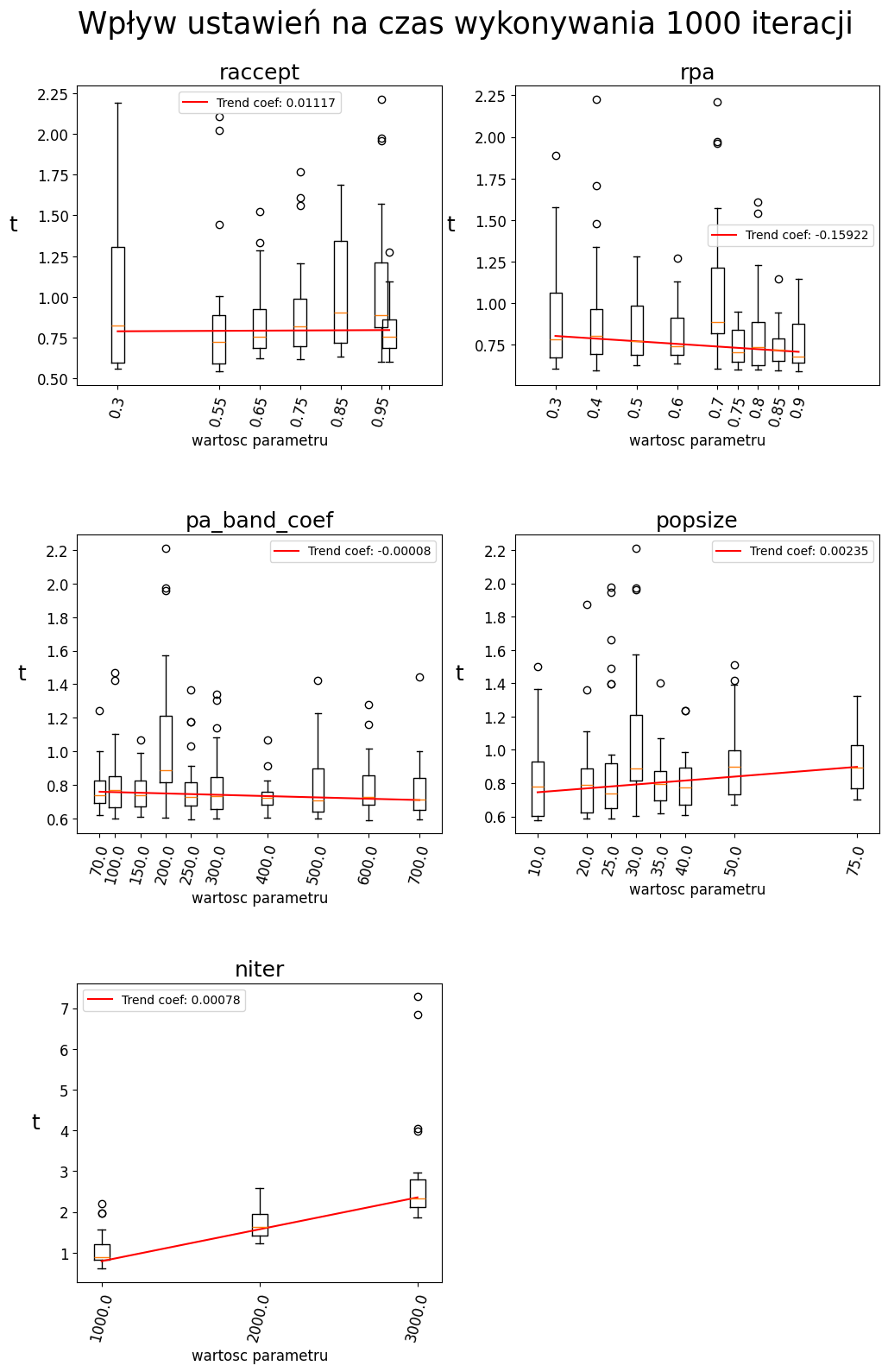


Spośród dwóch zbadanych parametrów zdecydowanie ważniejszy okazuje się być raccept. Łatwo zauważyć, że kiedy jest w swoim optymalnym przedziale (mniej więcej 0.75 – 0.85) to niezależnie od ustawienia rpa wyniki są świetne. Dla parametru rpa po medianach można zaobserwować, że najlepszy jest szeroki przedział niższych wartości, który jak się okazuje jest bardzo podobny do tego uzyskanego w analizie jednoczynnikowej, jednak nie w pełni – w tej wersji zakres zwracający dobre wyniki jest szerszy.

Podsumowując, najlepiej skupić się na badaniu raccept, ponieważ optymalizacja rpa sprowadza się bardziej do sprawdzenia paru ustawień tylko po to, żeby uniknąć kompletnie błędnej wartości jak np. 0.95 – dokładniejsza optymalizacja raczej nie jest opłacalna z powodu małych zysków.

# Wpływ ustawień na czas wykonania

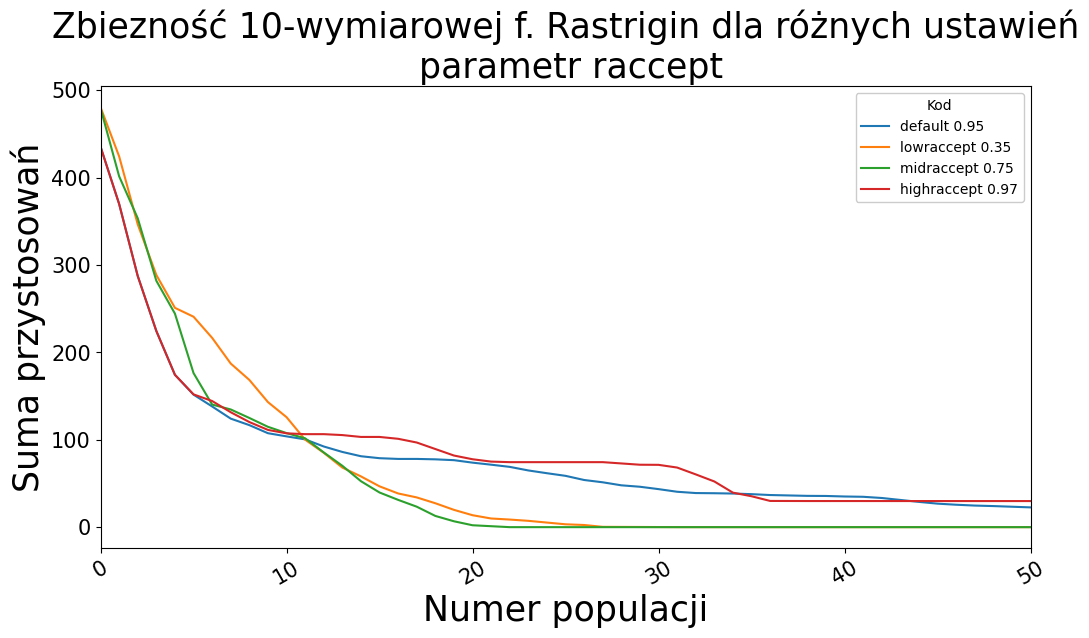
Badanie czasu trwania zostało wykonane na zwiększonej liczbie iteracji równe 1000 (500 więcej), ponieważ przy 500 czasy były na tyle krótkie, że nie dało się odróżnić wpływu parametrów od losowych odchyleń wynikających z fizycznej budowy komputera i współdziałających programów.



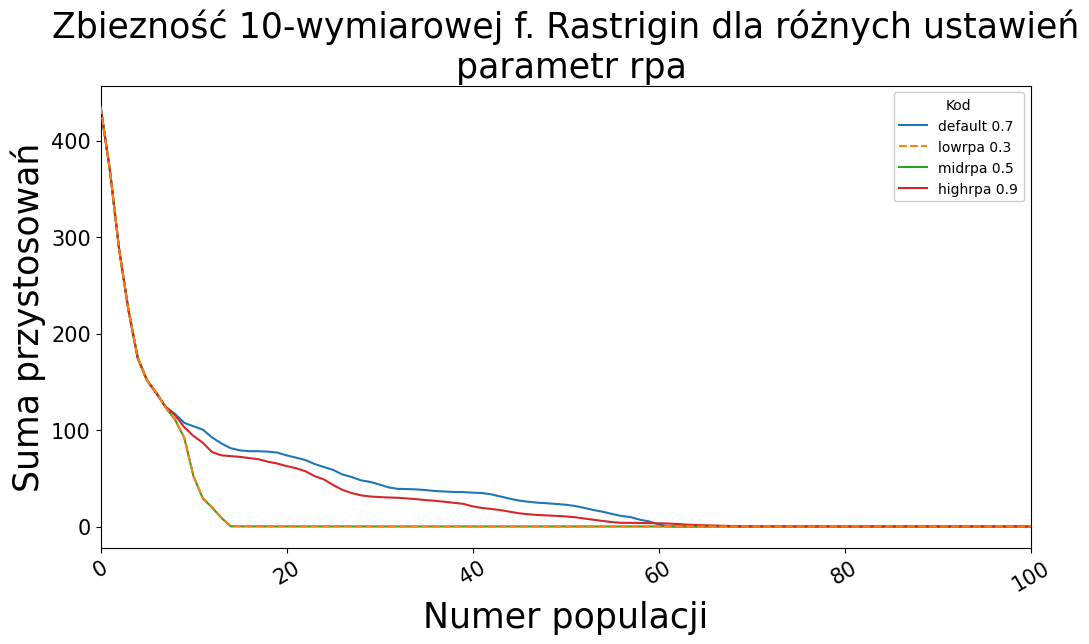
Zdecydowanie największy wpływ na czas wykonanie ma liczba iteracji. Drugi najbardziej wpływowy parametr to zgodnie z oczekiwaniami pop size. Największym zaskoczeniem jest pa\_rate, którego wzrost powoduje przyśpieszenie działania. Hipotezą tego zjawiska jest to, że implementacja losowania nowej nuty jest bardziej obciążająca niż funkcja dodająca szum do już istniejącej. Pozostałe parametry można uznać za nie powodujące znaczących zmian w czasie wykonania.

# Wpływ ustawień na zbieżność populacji

## Accept\_rate



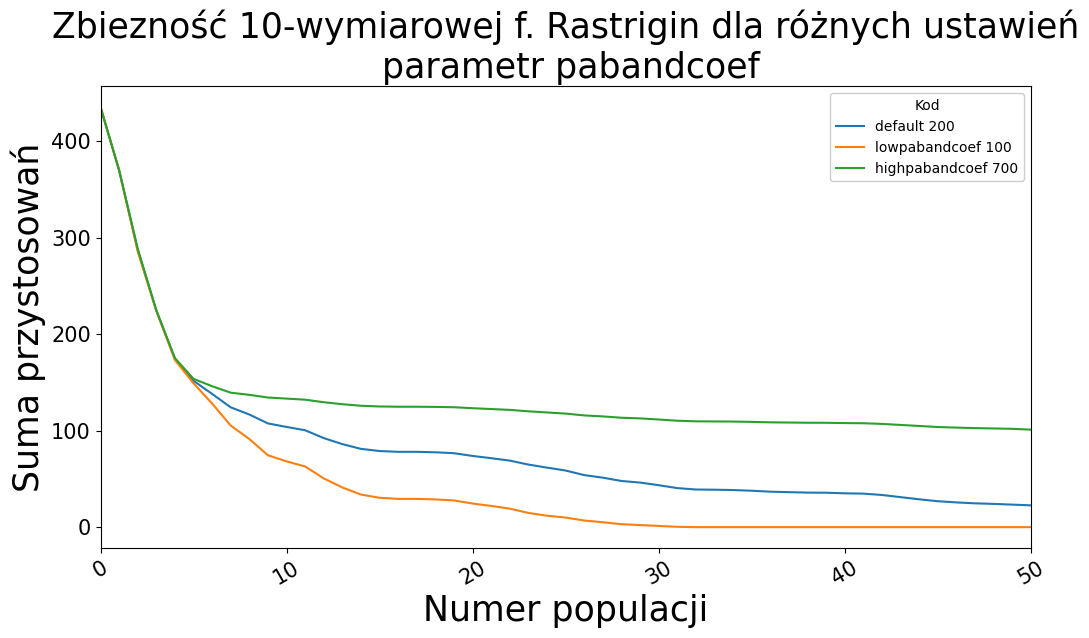
Wyższa wartość accept\_rate sprawia, że do kolejnej harmonii przechodzi więcej niezmienionych nut. Naturalnie oznacza to mniejsze zmiany w kolejnych iteracjach, a co za tym idzie wolniejszy zbieg.



W tym przypadku zakres zwiększono do 100 iteracji, aby upewnić się czy końcowa zbieżność zmienia się dla ustawień. Ponadto, wykresy niskiego (lowrpa) i pośredniego ustawienia (midrpa) nakładały się na siebie całkowicie, dlatego jeden z nich został zaznaczony linią przerywaną (może być to słabo widoczne na wizualizacji).

Dla pa\_rate (rpa) wyższe ustawienie pod kątem zbieżności są lepsze od domyślnego ustawienia 0.7, ale lepsze od niskiego i średniego ustawienia (odp. 0.3 i 0.5). Nie ma więc podstaw do jednoznacznego określenia żadnej zależności między rpa, a zbieżnością algorytmu, poza tym że dla pewnego przedziału algorytm działa lepiej (co już wiadomo z poprzednich sekcji).

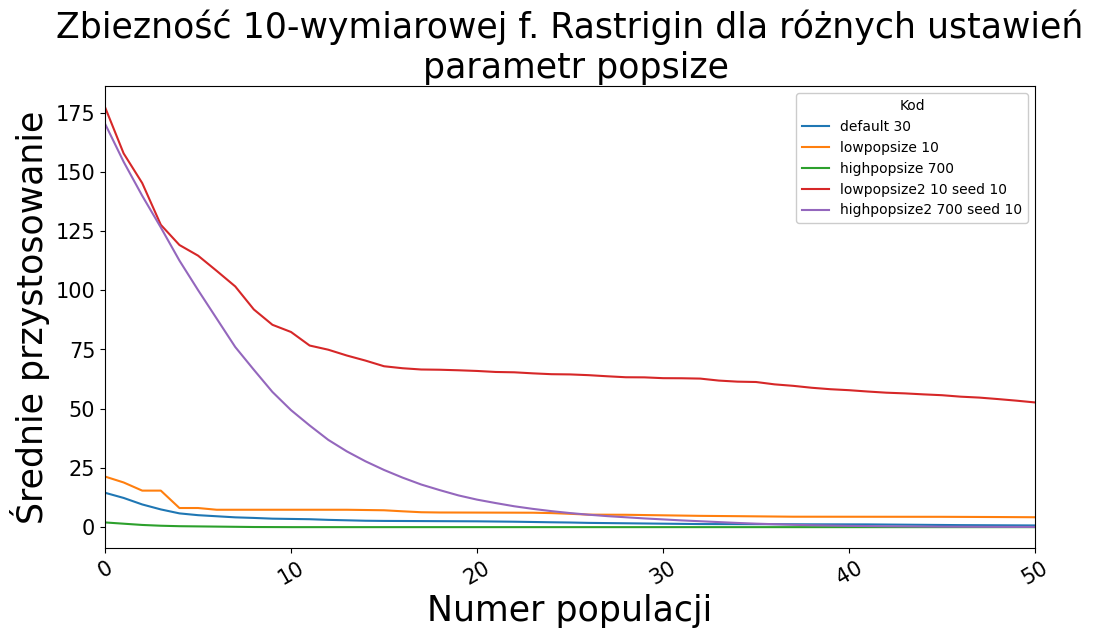
## Pa\_band\_coef

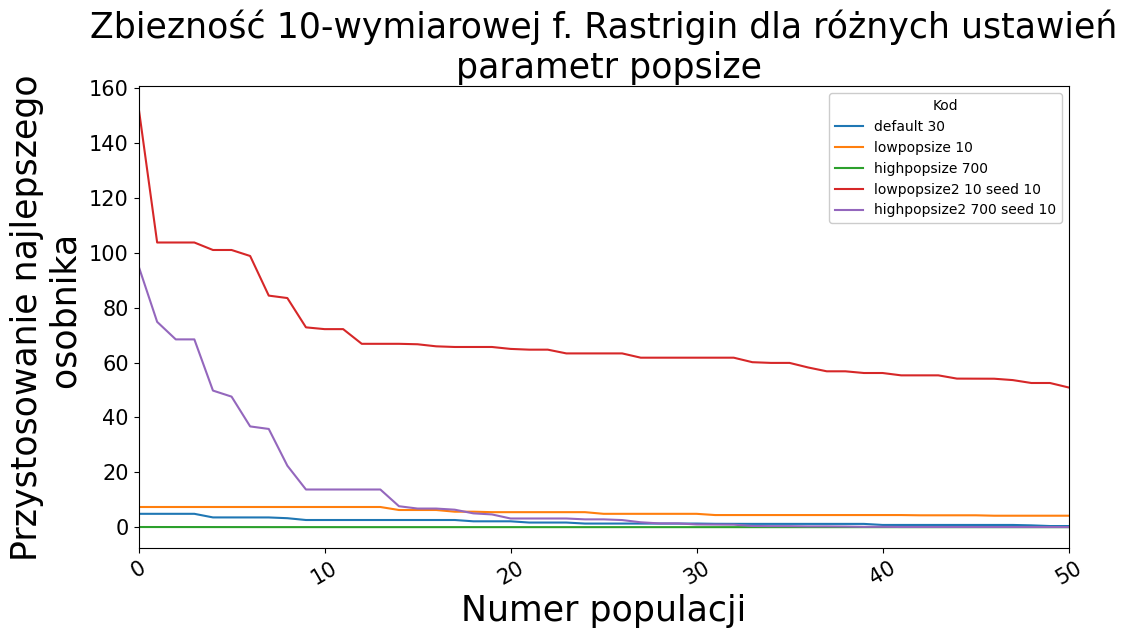


Zgodnie z oczekiwaniami wysokie ustawienie współczynnika, czyli mniejszy krok powoduje wolniejszą zbieżność. Jak wiadomo jednak z wyników i poprzednich badań, harmonia przy powolnym kroku jest bardziej konsekwentna w zbieganiu i pozwala na osiągnięcie dokładniejszych wartości (zakładając, że algorytm zdąży dotrzeć do optimum w danej liczbie iteracji).

## Popsize

W trakcie badania okazało się, że duży wpływ ma ziarno losowe, dlatego na wykresach zostały zawarte dwie wersje (seed 0 oraz seed 10).





Biorąc pod uwagę, że domyślnie użyte parametry raccpet i pa\_rate oferowały w miarę zbalansowane podejście do tworzenia kolejnych harmonii można zaobserwować, że zwiększenie liczebności nut pozwala na szybszy zbieg populacji, szczególnie patrząc na przystosowanie najlepszego osobnika. Jest to naturalne zjawisko, ponieważ więcej nut w harmonii to więcej szans na trafienie losu powodującego modyfikację nuty lub dodanie nowej.

Tak naprawdę zwiększenie liczebności populacji jest bardzo podobne do zwiększenia liczby iteracji, ponieważ również powoduje większą liczbę losowych zmian nut w czasie trwania algorytmu.

# Porównanie

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Funkcja | Wymiary | Accept\_rate | Pa\_rate | Pa\_band\_coef | popsize | niter | seed | Y |
| Ackley | 1 | 0.55 | 0.7 | 200 | 30 | 500 | 0 | 1.465494e-14 |
| Ackley | 5 | 0.95 | 0.7 | 200 | 75 | 500 | 10 | 0.0006212985 |
| Ackley | 10 | 0.95 | 0.7 | 200 | 75 | 500 | 100 | 0.001820866 |
| DeJong | 1 | 0.95 | 0.8 | 200 | 30 | 500 | 100 | 1.232595e-32 |
| DeJong | 5 | 0.95 | 0.7 | 700 | 30 | 500 | 0 | 2.906474e-07 |
| DeJong | 10 | 0.95 | 0.7 | 200 | 75 | 500 | 100 | 3.077976e-06 |
| Rastrigin | 1 | 0.65 | 0.7 | 200 | 30 | 500 | 0 | 0 |
| Rastrigin | 5 | 0.95 | 0.7 | 600 | 30 | 500 | 0 | 2.118647e-07 |
| Rastrigin | 10 | 0.95 | 0.7 | 200 | 75 | 500 | 100 | 6.890985e-06 |

Tabela zawierające najlepsze wyniki HS pochodzące z analizy jednoczynnikowej dla każdego zadania

Co do samych wyników, zdecydowana większość już jest świetna. Wyjątkiem jest tylko funkcja 5 i 10 wymiarowa Ackley’a, która okazała się być problematyczna dla tego algorytmu. Wyniki dało by się poprawić chociażby stosując większa populację i więcej iteracji, jednak takie poprawy nie zostały wprowadzone, by zachować podobieństwo „mocy” ustawień w stosunku do reszty zadań.

W porównaniu do poprzednio badanych algorytmów poszukiwanie harmonii na pewno radzi sobie gorzej od algorytmu rojowego. W odniesieniu do algorytmu ewolucyjnego ciężko jednoznacznie wybrać zwycięzcę, jednak ze względu na uniwersalność i trochę lepszą skuteczność lepszy jest raczej AE. Algorytm poszukiwania harmonii można ocenić jako nowoczesny odpowiednik symulowanego wyżarzania, czyli prosty i bardzo zrozumiały algorytm, który pozwala błyskawicznie osiągnąć całkiem dobre (ale nie najlepsze) wyniki.

# Podsumowanie wszystkich badań