Laboratorium - Wydział Elektroniki	**Technika optymalizacji**
Autorzy: 1. [132652] **Karol Kozłowski** *(EIT / ESA)* 2. [132750] **Karol Nikścin** *(EIT / ESA)*	Wrocław, dnia 16-01-2006 Prowadzący: Dr inż. Ewa Szlachcic
Program optymalizujący metodą simpleksu Neldera-Meada	**PROJEKT**

1. Sformułowanie zadania i omówienie algorytmu.

Optymalizacja jest to zagadnienie matematyczne, którego zadaniem jest znalezienie najlepszego rozwiązania zadanego problemu.

Metoda optymalizacji Neldera-Meada zwanej również metodą pełzającego simplesku, jest efektem prac dwóch naukowców J. A. Nelder'a i R. Mead'a. Opis tej metody został przez nich po raz pierwszy opublikowany w roku 1965 w czasopiśmie „Computer Journal" w artykule noszącym tytuł „A Simplex Method for Function Minimization„. Metoda ta jest typowym przedstawicielem metod poszukiwań prostych i po mimo swojego wieku nadal jest często stosowana ze względu na swoją prostotę. Polega ona na odpowiednim przekształcaniu n wymiarowego simpleksu w przestrzeni $n+1$ wymiarowej za pomocą odpowiednich przekształceń do otrzymania najbardziej optymalnego rozwiązania (za optymalne rozwiązanie przyjmuje się najczęściej takie rozwiązanie, w którym odległości pomiędzy wierzchołkami są mniejsze, bądź równe dokładności obliczeń, dlatego odległości te w początkowym simpleksie powinny być znacznie większe od dokładności obliczeń). Metoda korzysta z 3 przekształceń – odbicia, ekspansji oraz kontrakcji. Podstawowymi parametrami optymalizacji są: współczynnik odbicia $\alpha = 1$, współczynnik kontrakcji $0 < \beta < 1$, współczynnik ekspansji $\gamma > 1$ oraz dokładność obliczeń ε.

2. Informacje o programie

Program został napisany w środowisku "Borland C++ Bulider 6" zaimplementowany został w nim muParser (http://muparser.sourceforge.net/) dostępny bezpłatnie w internecie.

3. Przykłady testowe

- Funkcja Engwalla – punkty startowe generowane losowo z przedziału [-5, 5] – algorytm zawsze znajduje minimum, najczęściej w <100 iteracjach (dla parametrów początkowych, przy zmianie parametrów liczba iteracji spada, bądź rośnie – np. dla $\beta = 0{,}25$) i $\gamma = 2$ liczba iteracji wynosi od 20 do 30).
- Funkcja Powella – punkty startowe generowane losowo z przedziału [-5,5] – algorytm poprawnie znajduje optimum, zależność parametrów algorytmu od ilości iteracji jest podobna jak w poprzednim przykładzie
- Funkcja Zangwilla (trudna dla przyjętej metody) – nie wiem dlaczego ta funkcja jest trudna, optimum było zawsze znajdowane, bez względu na przyjęte parametry początkowe.

4. Szczegółowe informacje o programie

Aktualnie program działa dla maksymalnie 10 zmiennych i 100 punktach simpleksu (takie były założenia, może to jednak być łatwo rozszerzone). Jako parametry programu podajemy:

- minimalizowaną funkcję celu – jedyne ograniczenie to możliwości muParsera[1]
- wymiar simpleksu – jest on automatycznie ustawiany na wartość $2n+1$, gdzie n to liczba zmiennych funkcji celu, może on być zmieniony ręcznie lecz nie na wartość mniejszą od $n+1$.
- podstawowe parametry metody takie jak współczynnik odbicia $(\alpha = 1)$, współczynnik kontrakcji $(\beta = 0,8)$, współczynnik ekspansji $(\gamma = 1,2)$ oraz dokładność obliczeń $(\varepsilon = 0,01)$. W nawiasach podano parametry domyślne, któremogą byćłatwo zmienione z poziomu GUI.
- Punkt startowy – można go podać ręcznie, bądź wygenerować losowo podając przedziały w jakich mają być generowane zmienne.

Postęp algorytmu można kontrolować ręcznie – krok po kroku, bądź użyć opcji automatycznej optymalizacji, która zatrzyma się po znalezieniu optimum (odległość pomiędzy punktami simpleksu będzie mniejsza od przyjętej wartości ε) lub po osiągnięciu określonego limitu iteracji (domyślnie 500). Kryterium największej odległości może być jednak wyłączone aby umożliwić łatwe wykonanie kolejnych iteracji, jeżeli zaistniała by taka potrzeba.

Niestety na chwilę obecną nie udało nam się zaimplementować żadnej wizualizacji w programie

5. Dyskusja wpływy punktu startowego na działanie metody

Wybór punktów startowych ma bardzo duży wpływ na ilość iteracji, generalnie im bliżej minimum znajdują się punkty startowe to algorytm potrzebuje tym mniej iteracji na dojście do celu. Czasami nieprawidłowy dobór punktów startowych powoduje błędne działanie algorytmu czego efektem są przypadki „zapadania się" simpleksu do nieistniejącego minimum – występowało to tylko w przypadku funkcji niewypukłych, lecz po zresetowaniu punktów startowych i uruchomieniu algorytmu optimum było znajdowane, być może gdybyśmy zastosowali w algorytmie mechanizm „periodycznej odnowy" o którym mowa jest w literaturze, problem ten zostałby uniknięty.

6. Literatura:

- A.Stachurski, P. Wierzbicki - „Podstawy optymalizacj" - Oficyna Wydawnicza Politechniki Warszawskiej 2001
- D. Byatt, I. Coope, C. Price - „40 Years of the Nelder-Mead Algorithm" University of Cantenbury, New Zeland 2003
- Liczne opracowania w internecie takie jak: Simplexverfahren nach NELDER-MEAD **(http://tinyurl.com/d9ate)**

1 Do sprawozdania załączona jest lista funkcji obsługiwanych przez muParser