Informatyka, studia dzienne, II st. semestr I

Metody Obliczeniowe Optymalizacji **2010/2011**

Prowadzący: mgr inż. Łukasz Chomątek czwartek, 14:15

Data oddania: _____ Ocena: _____

Michał Janiszewski 169485
Leszek Wach 169513

Zadanie 2: Optymalizacja kierunkowa*

1. Cel zadania

Celem zadania było napisanie programu, który dla dowolnej funkcji dwóch zmiennych znajdzie jej minimum wzdłuż wyznaczonego odcinka – dokona minimalizacji kierunkowej. Operacja ta ma odbywać się z wykorzystaniem trzech kryteriów:

1. Armijo,
2. Wolfa,
3. Goldensteina.

2. Metoda rozwiązania

W celu rozwiązania zadania stworzyliśmy skrypt programu Matlab. Skrypt prezentuje użytkownikowi okienko GUI, w którym należy wprowadzić parametry:

funkcja funkcja, która poddawana będzie optymalizacji, oznaczona jako F,
start punkt początkowy, oznaczany jako p_0,
direction kierunek; wektor od punktu początkowego do końcowego, oznaczany jako d,
$\mathbf{c_1}$, $\mathbf{c_2}$ parametry określające zachowanie się metod[1],
ρ współczynnik zmiany kroku,

* SVN: `https://serce.ics.p.lodz.pl/svn/labs/moo/lcjp_cz1415/wacjan/Zadanie2@116`

[1] Znaczenie tych parametrów różni się pomiędzy metodami.

ϵ pożądana dokładność rozwiązania,
należy także wybrać jedno z dostępnych kryteriów.

Zadaniem programu jest poszukiwanie pewnej wartości kroku λ, która wyraża krotność wektora kierunku rozpoczynającego się w punkcie startowym. Krok ten będzie odpowiadał znalezionemu minimum. Zależność pomiędzy wartością λ, a wartością funkcji F prezentuje poniższy wzór:

$$f(\lambda) = F(p_0.x + \lambda \cdot d.x, p_0.y + \lambda \cdot d.y) \tag{1}$$

Należy zauważyć, że wartościom kroku należącym odcinka $\overline{p_0, p_0 + d}$ odpowiadają wartości [0, 1].

Poszukiwanie wartości kroku jest metodą iteracyjną, która w każdej iteracji przybliża wartość rozwiązania, aż do osiągnięcia zadanego warunku stopu – w tym przypadku jest to kryterium stacjonarości:

$$|\nabla f(\lambda)| \leq \epsilon \tag{2}$$

Punktem p_k nazywać będziemy przybliżenie rozwiązania wyznaczone w k-tej iteracji, zaś symbolem λ_k oznaczać będziemy krok podjęty w k-tej iteracji. Zależności pomiędzy tymi zmiennymi prezentują równania:

$$\lambda = \sum_{i=0}^{n} \lambda_i \tag{3}$$

gdzie n oznacza całkowitą ilość iteracji oraz:

$$p_k = p_0 + \sum_{i=0}^{k} \lambda_i \cdot d \tag{4}$$

3. Kryteria

Poza wymaganiem poprawy[2], stosowane są dodatkowe kryteria, które spełniają rolę kryterium stopu. Dzięki ich wykorzystaniu zapewniamy dokładniejsze i szybsze znalezienie rozwiązania.

Każde z kryteriów zaimplementowane jest jako rekurencyjna metoda, której działanie przedstawia następujący algorytm:

1. Pobierz od użytkownika wszystkie dane,
2. podstaw $\lambda_0 = 0$
3. jak λ_k podstaw znak pochodnej $sign(f(\lambda))$,
4. sprawdź wybraną metodą, czy $f(\lambda + \lambda_k)$ spełnia warunki,
5. jeśli spełnia: podstaw $\lambda_k = \rho \cdot \lambda_k$, przejdź do 4,
6. jeśli nie jest spełniony warunek stacjonarości, rekurencyjnie przejdź do 3,
7. zakończ obliczenia, zwróć λ i ilość wywołań.

3.1. Kryterium Armijo

Najprostszy test, kryterium Armijo, sprawdza czy spadek wartości funkcji w danej iteracji jest nie mniejszy niż jej pochodna przemnożona przez obrany parametr c_1. Warunek ten można zapisać w następujący sposób:

$$f(\lambda + \lambda_k) \leq f(\lambda) + c_1 \cdot \lambda_k \cdot f'(\lambda) \tag{5}$$

[2] Wartość $f(\lambda)$ dla każdego kolejnego oszacowania λ powinna być mniejsza.

3.2. Kryterium Wolfa

Podstawową wadą kryterium Armijo jest ograniczanie tylko maksymalnej długości kroku, co może prowadzić do bardzo wolnej zbieżności – poprzez stosowanie możliwie najmniejszego kroku λ_k.

Kryterium Wolfa łączy kryterium Armijo z testem krzywizny, który ogranicza wartość kroku od dołu:

$$\begin{aligned} f(\lambda + \lambda_k) &\leq f(\lambda) + c_1 \cdot \lambda_k \cdot f'(\lambda) && (6) \\ f'(\lambda + \lambda_k) &\geq c_2 \cdot f'(\lambda) && (7) \end{aligned}$$

gdzie $c_1 \in (0, 1)$, natomiast $c_2 \in (c_1, 1)$.

Zadaniem tego kryterium jest zagwarantowanie, że w miejscu dużego spadku funkcji, krok nie będzie mały.

3.3. Kryterium Goldsteina

Podobnie jak kryterium Wolfa, kryterium Goldsteina kontroluje zarówno górną jak i dolną granicę kroku zabezpieczając przed zbyt wolnym testowaniem zbieżności.

Tak jak w poprzednim przypadku, również tu wykorzystane jest kryterium Armijo do określania górnej granicy.

$$\begin{aligned} f(\lambda + \lambda_k) &\leq f(\lambda) + c_1 \cdot \lambda_k \cdot f'(\lambda) && (8) \\ f(\lambda + \lambda_k) &\geq f(\lambda) + (1 - c_1) \cdot \lambda_k \cdot f'(\lambda) && (9) \end{aligned}$$

gdzie $c_1 \in (0, \frac{1}{2})$.

4. Wyniki

Poniższe tabele prezentują wyniki otrzymane za pomocą napisanego programu. Otrzymane wyniki zostały porównane z wynikami otrzymanymi za pomocą serwisu WolframAlpha.

4.1. Funkcja testowa 1

Pierwsza funkcja testowa była postaci:

$$F(x, y) = x^2 \cdot \sin(x) + y^2 \cdot \sin(y) \tag{10}$$

Punktem startowym testów był punkt $(-4; -4)$, zaś wektor kierunku miał wartości $[2; 2]$.

Znalezione przez WolframAlpha minimum na zadanym kierunku to w przybliżeniu:

$$F(-2.28892, -2.28892) = -7.89060 \tag{11}$$

4.1.1. Armijo

Wyniki działania metody Armijo prezentuje tabela 1.

Tablica 1. Wyniki działania programu dla funkcji testowej 1 dla kryterium Armijo

c_1	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.1	0.85553	18	-2.28894	-2.28894	-7.8906
0.1	0.3	0.855545	8	-2.28891	-2.28891	-7.8906
0.1	0.5	0.85553	7	-2.28894	-2.28894	-7.8906
0.1	0.7	0.855522	16	-2.28896	-2.28896	-7.8906
0.3	0.1	0.85555	19	-2.2889	-2.2889	-7.8906
0.3	0.3	0.855543	7	-2.28891	-2.28891	-7.8906
0.3	0.5	0.85553	5	-2.28894	-2.28894	-7.8906
0.3	0.7	0.855532	6	-2.28894	-2.28894	-7.8906
0.5	0.1	0.85553	26	-2.28894	-2.28894	-7.8906
0.5	0.3	0.855523	14	-2.28895	-2.28895	-7.8906
0.5	0.5	0.85553	7	-2.28894	-2.28894	-7.8906
0.5	0.7	0.85553	7	-2.28894	-2.28894	-7.8906

4.1.2. Wolf

Wyniki działania metody Wolfa prezentuje tabela 2.

4.1.3. Goldstein

Wyniki działania metody Goldsteina prezentuje tabela 3.

4.2. Funkcja testowa 2

Druga funkcja testowa była postaci:

$$F(x, y) = x^2 \cdot y^2 + \cos(y) + \sin(x) \tag{12}$$

Punktem startowym testów był punkt $(-2; -6)$, zaś wektor kierunku miał wartości $[2; 6]$.

Znalezione przez WolframAlpha minima na tym kierunku to w przybliżeniu:

$$F(-0.47496, -1.42490) = 0.14610 \tag{13}$$
$$F(0.36317, 1.08952) = 0.97471 \tag{14}$$

4.2.1. Armijo

Wyniki działania metody Armijo prezentuje tabela 4.

4.2.2. Wolf

Wyniki działania metody Wolfa prezentuje tabela 5.

4.2.3. Goldstein

Wyniki działania metody Goldsteina prezentuje tabela 6.

5. Wnioski

Podczas testowania stworzonego programu napotkaliśmy duże problemy z takim dobraniem parametrów optymalizacji, aby rozwiązanie zostało w

Tablica 2. Wyniki działania programu dla funkcji testowej 1 dla kryterium Wolfa

c_1	c_2	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.2	0.1	0.855538	5	-2.28892	-2.28892	-7.8906
0.1	0.2	0.3	0.855526	5	-2.28895	-2.28895	-7.8906
0.1	0.2	0.4	0.85553	5	-2.28894	-2.28894	-7.8906
0.1	0.4	0.1	0.855536	8	-2.28893	-2.28893	-7.8906
0.1	0.4	0.3	0.855522	8	-2.28896	-2.28896	-7.8906
0.1	0.4	0.4	0.855541	7	-2.28892	-2.28892	-7.8906
0.1	0.6	0.1	0.855541	10	-2.28892	-2.28892	-7.8906
0.1	0.6	0.3	0.855537	9	-2.28893	-2.28893	-7.8906
0.1	0.6	0.4	0.855541	7	-2.28892	-2.28892	-7.8906
0.3	0.2	0.4	0.85553	5	-2.28894	-2.28894	-7.8906
0.3	0.4	0.1	0.855536	8	-2.28893	-2.28893	-7.8906
0.3	0.4	0.3	0.855522	8	-2.28896	-2.28896	-7.8906
0.3	0.4	0.4	0.855527	7	-2.28895	-2.28895	-7.8906
0.3	0.6	0.1	0.855541	10	-2.28892	-2.28892	-7.8906
0.3	0.6	0.3	0.855536	8	-2.28893	-2.28893	-7.8906
0.3	0.6	0.4	0.855539	8	-2.28892	-2.28892	-7.8906
0.5	0.6	0.1	0.855541	10	-2.28892	-2.28892	-7.8906
0.5	0.6	0.3	0.855536	8	-2.28893	-2.28893	-7.8906
0.5	0.6	0.4	0.855539	8	-2.28892	-2.28892	-7.8906

Tablica 3. Wyniki działania programu dla funkcji testowej 1 dla kryterium Goldsteina

c_1	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.1	0.85553	18	-2.28894	-2.28894	-7.8906
0.1	0.3	0.855545	8	-2.28891	-2.28891	-7.8906
0.1	0.5	0.85553	7	-2.28894	-2.28894	-7.8906
0.1	0.7	0.855522	16	-2.28896	-2.28896	-7.8906
0.2	0.1	0.85553	8	-2.28894	-2.28894	-7.8906
0.2	0.3	0.855536	8	-2.28893	-2.28893	-7.8906
0.2	0.5	0.85553	5	-2.28894	-2.28894	-7.8906
0.2	0.7	0.855547	9	-2.28891	-2.28891	-7.8906
0.3	0.1	0.855534	7	-2.28893	-2.28893	-7.8906
0.3	0.3	0.855538	7	-2.28892	-2.28892	-7.8906
0.3	0.5	0.85553	5	-2.28894	-2.28894	-7.8906
0.3	0.7	0.855532	6	-2.28894	-2.28894	-7.8906

Tablica 4. Wyniki działania programu dla funkcji testowej 2 dla kryterium Armijo

c_1	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.1	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.1	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.2	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.2	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.2	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.3	0.1	0.76251	21	-0.47498	-1.42494	0.146102
0.3	0.3	0.762521	11	-0.474957	-1.42487	0.146102
0.3	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.4	0.1	0.76251	21	-0.47498	-1.42494	0.146102
0.4	0.3	0.762525	11	-0.47495	-1.42485	0.146102
0.4	0.5	0.762512	7	-0.474976	-1.42493	0.146102

Tablica 5. Wyniki działania programu dla funkcji testowej 2 dla kryterium Wolfa

c_1	c_2	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.2	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.1	0.2	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.1	0.2	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.1	0.4	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.1	0.4	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.1	0.4	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.1	0.6	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.1	0.6	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.1	0.6	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.2	0.4	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.2	0.4	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.2	0.4	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.2	0.6	0.1	0.76251	15	-0.47498	-1.42494	0.146102
0.2	0.6	0.3	0.762509	7	-0.474981	-1.42494	0.146102
0.2	0.6	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.3	0.4	0.1	0.76251	21	-0.47498	-1.42494	0.146102
0.3	0.4	0.3	0.762521	11	-0.474957	-1.42487	0.146102
0.3	0.4	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.3	0.6	0.1	0.76251	21	-0.47498	-1.42494	0.146102
0.3	0.6	0.3	0.762521	11	-0.474957	-1.42487	0.146102
0.3	0.6	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.4	0.6	0.1	0.76251	21	-0.47498	-1.42494	0.146102
0.4	0.6	0.3	0.762525	11	-0.47495	-1.42485	0.146102
0.4	0.6	0.5	0.762512	7	-0.474976	-1.42493	0.146102

Tablica 6. Wyniki działania programu dla funkcji testowej 2 dla kryterium Goldsteina

c_1	ρ	λ	Ilość wywołań	$f(\lambda).x$	$f(\lambda).y$	Minimum
0.1	0.1	0.762522	7	-0.474957	-1.42487	0.146102
0.1	0.3	0.76252	7	-0.47496	-1.42488	0.146102
0.1	0.5	0.762516	12	-0.474968	-1.4249	0.146102
0.2	0.1	0.762521	6	-0.474959	-1.42488	0.146102
0.2	0.3	0.76252	7	-0.47496	-1.42488	0.146102
0.2	0.5	0.762515	8	-0.474969	-1.42491	0.146102
0.3	0.1	0.76252	7	-0.47496	-1.42488	0.146102
0.3	0.3	0.762522	9	-0.474955	-1.42487	0.146102
0.3	0.5	0.762512	7	-0.474976	-1.42493	0.146102
0.4	0.1	0.762516	6	-0.474967	-1.4249	0.146102
0.4	0.3	0.762524	6	-0.474952	-1.42485	0.146102
0.4	0.5	0.762512	5	-0.474976	-1.42493	0.146102

ogóle odnalezione. Duża część testów została odrzucona ze względu na brak wyniku.

Powodem takiego stanu rzeczy może być takie dobranie dostępnych opcji (w tym optymalizowanej funkcji), że krok wywołania jest bardzo mały, a przez to przekroczony jest limit rekurencji.

Innym powodem powyższego błędu może być przyjęta metoda różniczkowania – dokonujemy obliczenia wartości funkcji w trzech punktach i wykorzystując je szacujemy wartość pochodnej. W niektórych przypadkach metoda ta może dawać niepożądane rezultaty.

Dla wszystkich zestawów testowych ilość wymaganych do wykonania kroków jest dość mała, w związku z czym trudno jest jednoznacznie określić istnienie tendencji.

Wydaje się jednak, że najlepsze rezultaty uzyskiwane są dla parametru ρ równego 0,5 dla wszystkich metod.

W przypadku kryterium Armijo efekt zmiany c_1 zależny jest od parametru ρ.

Zmiana tego parametru przy ρ w okolicy 0.5 nie przynosi widocznych zmian, jednak gdy ρ jest małe, zmniejszanie c_1 zmniejsza też wymaganą ilość kroków.

Kryterium Wolfa jest znacznie bardziej podatne na zmianę parametru c_2 – wraz ze wzrostem tej wartości wzrasta konieczna ilość kroków do rozwiązania. Optymalne do dobrania parametry to małe c_1, np. 0,1 oraz ρ w okolicach 0,5.

Kryterium Goldsteina daje najlepsze rezultaty – dochodzi do rozwiązania w średnio najmniejszej ilości kroków. W przypadku tej metody widoczna jest poprawa w ilości kroków wraz z dążeniem parametrów c_1 oraz ρ do 0,5.

Literatura

[1] Grega, Wojciech. *Metody optymalizacji. Wykład 4* [online]. [dostęp: 27 marca 2011]. Dostępny w Internecie: `http://aq.ia.agh.edu.pl/Aquarium/Dydaktyk/Wyklady/MO/2005-06/Wyklad04.PDF`