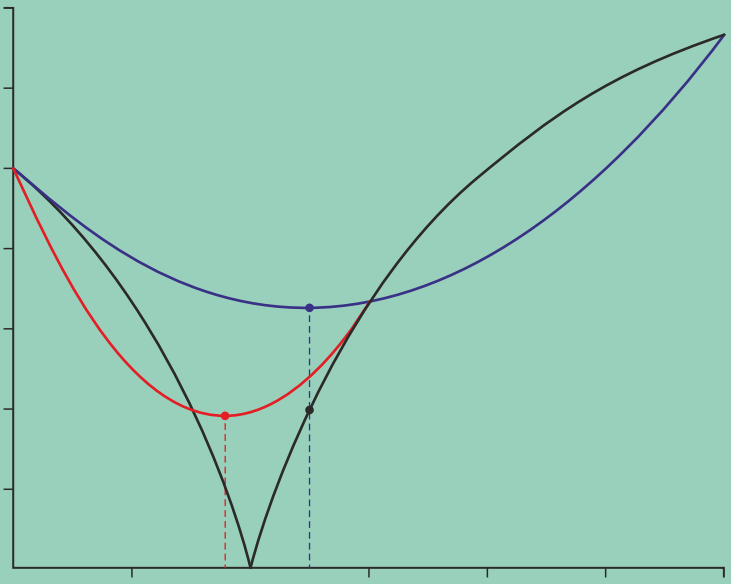
METODY OPTYMALIZACJI

WYBRANE ZAGADNIENIA

Skrypt dla studentów czwartego semestru matematyki stosowanej

Krzysztof Piekarski



Politechnika Białostocka

METODY OPTYMALIZACJI. WYBRANE ZAGADNIENIA

SKRYPT DLA STUDENTÓW CZWARTEGO SEMESTRU MATEMATYKI STOSOWANEJ

Krzysztof Piekarski

Politechnika Białostocka

OFICYNA WYDAWNICZA POLITECHNIKI BIAŁOSTOCKIEJ
BIAŁYSTOK 2024

Recenzent:
prof. dr hab. Eugeniusz Zieniuk

Redaktor naukowy dyscypliny matematyka:
prof. dr hab. Zbigniew Bartosiewicz

Korekta językowa:
Katarzyna Duniewska

Okładka:
Marcin Dominów

ISBN 978-83-68077-08-7 (e-Book)
DOI: 10.24427/978-83-68077-08-7

Oficyna Wydawnicza Politechniki Białostockiej
ul. Wiejska 45C, 15-351 Białystok
e-mail: oficyna.wydawnicza@pb.edu.pl
www.pb.edu.pl

Spis treści

Wstęp

Niniejszy skrypt w zamyśle autora ma służyć przede wszystkim studentom kierunku matematyka stosowana na Wydziale Informatyki Politechniki Białostockiej jako pomoc w lepszym rozumieniu zagadnień przedstawianych na zajęciach w ramach pracowni specjalistycznej z przedmiotu *Metody optymalizacji*, który obecnie jest realizowany na IV semestrze studiów I stopnia. Zagadnienia omawiane w kolejnych rozdziałach są zgodne z treściami programowymi zawartymi w sylabusie do tego przedmiotu. Treści prezentowane w skrypcie mogą być także przydatne dla studentów innych wydziałów zainteresowanych poznaniem i wykorzystaniem podstawowych metod numerycznych optymalizacji statycznej.

Istnieje bardzo bogata literatura dotycząca optymalizacji, zarówno jeżeli chodzi o aspekt teoretyczny, jak i praktyczny. Biorąc pod uwagę to, do kogo adresowane jest prezentowane opracowanie, ograniczymy się tu do zaproponowania jedynie kilku pozycji bibliograficznych mogących stanowić istotne uzupełnienie informacji podanych w skrypcie. Warto zwrócić uwagę Czytelnika zainteresowanego poszerzeniem swojej wiedzy na pozycje, które można uznać obecnie za klasyczne, np. Findeisen, Szymanowski, Wierzbicki (1980) czy Zorychta, Ogryczak (1981). Student poszukujący informacji podanych w bardzo przystępny sposób, ilustrowanych wieloma przykładami, może zainteresować się publikacjami takimi jak np. Kusiak, Danielewska-Tułecka, Oprocha (2019), Stadnicki (2017). Optymalizacja jest jedną z tych dziedzin matematyki, która rozwija się bardzo dynamicznie. Metody stosowane kilkadziesiąt lat temu często ustępują miejsca nowym, zwykle bardziej złożonym, ale jednocześnie znacznie efektywniejszym i bardziej niezawodnym. Metody takie zostały przedstawione przede wszystkim w rozdziale 3. Ich szczegółowy opis można znaleźć np. w Bonnans, Gilbert, Lemaréchal, Sagastizábal (2006), Nocedal, Wright (2006), Gill, Murray, Wright (1981) czy Fletcher (1987).

Teoretyczne podstawy optymalizacji przedstawiane są na wykładzie, natomiast związane z nimi odpowiednie narzędzia matematyczne są wykorzystywane na ćwiczeniach audytoryjnych. W większości praktycznych zastosowań ilość zmiennych decyzyjnych i ograniczeń występujących w zadaniu optymalizacji nie pozwala na jego efektywne rozwiązanie w sposób analityczny. W przypadku programowania nieliniowego, nawet wtedy, gdy rozmiar zadania nie jest zbyt duży, uzyskanie dokładnego rozwiązania może być niemożliwe, właśnie ze względu na nieliniowość funkcji celu lub ograniczeń. Z pomocą przychodzą wówczas numeryczne metody optymalizacji, które pozwalają na otrzymanie przynajmniej rozwiązania przybliżonego. W niniejszym skrypcie akcent położono na związane z tymi metodami algorytmy, które autor starał się przedstawić w przystępny sposób. Większość algoryt-

mów została zilustrowana odpowiednimi przykładami, w których krok po kroku wykonywane są obliczenia w początkowych iteracjach. Wyniki z kolejnych kilku iteracji są często przedstawione w osobnej tabeli i na wykresie. Czytelnik ma dzięki temu sposobność praktycznego poznania danej metody. W tym celu nie wystarczy jednak tylko przeczytać rozwiązanie przedstawione w przykładzie, ale należy samodzielnie powtórzyć obliczenia tam zawarte. Studenci mogą także spróbować zaimplementować algorytm danej metody, tworząc odpowiednią procedurę obliczeniową. Powyższe działania niewątpliwie pomogą uzyskać większą świadomość, jeżeli chodzi o sposób działania podstawowych metod optymalizacji, ale nie stanowią celu samego w sobie, a jedynie do niego prowadzą. Najważniejszym efektem, który student powinien uzyskać w wyniku uczestnictwa w zajęciach pracowni specjalistycznej z tego przedmiotu, jest umiejętność wykorzystania standardowych procedur optymalizacyjnych.

Pakietem oprogramowania proponowanym do wykorzystania na zajęciach jest MATLAB, ze szczególnym uwzględnieniem procedur dostępnych w bibliotece *Optimization Toolbox*. W skrypcie podane są przykłady użycia takich procedur do rozwiązywania konkretnych zadań optymalizacji. Działanie standardowej procedury zależy od wartości wielu parametrów opcjonalnych, z którymi jest ona wywoływana. Bezcelowe byłoby podawanie opisu wszystkich dostępnych opcji – zainteresowany Czytelnik może znaleźć potrzebne informacje w dokumentacji pakietu MATLAB na oficjalnej stronie internetowej MATLAB (n.d.).

Oprócz umiejętności zastosowania odpowiedniej procedury do rozwiązania zadania optymalizacji ważne są również właściwa interpretacja wyników, ich prezentacja w formie tabel lub wykresów i sformułowanie na tej podstawie poprawnych wniosków oraz porównanie przydatności, a w szczególności szybkości zbieżności różnych metod prowadzących do tego samego celu. Oczekuje się, że student na podstawie wykonanych obliczeń przygotuje sprawozdania, które będą zawierać powyższe elementy. W skrypcie podano konkretne propozycje zadań, których rozwiązania mogą znaleźć się w sprawozdaniach.

Białystok, wrzesień 2023 *Krzysztof Piekarski*

Rozdział 1

Programowanie nieliniowe: minimalizacja funkcji jednej zmiennej

1.1. Warunki konieczne i dostateczne optymalności

W tym rozdziale rozpatrujemy problem minimalizacji funkcji f jednej zmiennej:

$$\min_{x \in D} f(x) \tag{1.1}$$

gdzie $f : \mathbb{R} \to \mathbb{R}$ jest funkcją nieliniową, $D = \mathbb{R}$ lub $D = \langle c; d \rangle$, $c < d$. Dopuszczamy możliwość, że $c = -\infty$ lub $d = +\infty$. Jeżeli $D = \mathbb{R}$, mówimy o zadaniu minimalizacji bez ograniczeń, w przeciwnym razie o zadaniu z ograniczeniami.

W sytuacji, gdy funkcja f ma odpowiednią postać, powyższe zadanie można rozwiązać analitycznie. Przypomnijmy podstawowe fakty z analizy matematycznej, wykorzystywane przy poszukiwaniu ekstremów funkcji jednej zmiennej.

Definicja 1.1. Sąsiedztwem $S(x_0, \delta)$ **punktu** $x_0 \in \mathbb{R}$ **o promieniu** δ *nazywamy sumę przedziałów*

$$S(x_0, \delta) = (x_0 - \delta; x_0) \cup (x_0; x_0 + \delta)$$

Otoczeniem $O(x_0, \delta)$ **punktu** $x_0 \in \mathbb{R}$ **o promieniu** δ *nazywamy przedział*

$$O(x_0, \delta) = (x_0 - \delta; x_0 + \delta)$$

Jeżeli promień sąsiedztwa lub otoczenia nie jest istotny, to piszemy $S(x_0)$ *(odpowiednio* $O(x_0)$*).*

Definicja 1.2. Funkcja f **ma w punkcie** $\hat{x}$ **minimum lokalne** $\iff f(\hat{x}) \leqslant f(x)$ *dla wszystkich* $x \in D \cap O(\hat{x})$*, dla pewnego otoczenia* $O(\hat{x})$*.*
Jeżeli przy tym $f(\hat{x}) < f(x)$ *dla wszystkich* $x \in D \cap S(\hat{x})$*, dla pewnego sąsiedztwa* $S(\hat{x})$*, to mówimy o* **mimimum lokalnym właściwym**.

Definicja 1.3. Funkcja f **ma w punkcie** $\hat{x}$ **maksimum lokalne** $\iff f(\hat{x}) \geqslant f(x)$ *dla wszystkich* $x \in D \cap O(\hat{x})$*, dla pewnego otoczenia* $O(\hat{x})$*.*
Jeżeli przy tym $f(\hat{x}) > f(x)$ *dla wszystkich* $x \in D \cap S(\hat{x})$*, dla pewnego sąsiedztwa* $S(\hat{x})$*, to mówimy o* **maksimum lokalnym właściwym**.

Definicja 1.4. *Minima i maksima lokalne nazywamy* **ekstremami lokalnymi**.

Definicja 1.5. Funkcja f **ma w punkcie** $\hat{x}$ **ekstremum globalne** $\iff$ *nierówności w definicjach 1.2 oraz 1.3 są spełnione dla wszystkich* $x \in D$*.*

Uwaga 1.1. Dla dowolnej funkcji f zachodzi:

$$\min f(x) = -\max(-f(x))$$

dlatego w dalszym ciągu będziemy rozpatrywać przede wszystkim zadania minimalizacji.

Twierdzenie 1.6 (pierwszy warunek konieczny istnienia mimimum). *Niech funkcja f ma minimum lokalne w punkcie $\hat{x} \in D$ i niech istnieje ciągła pochodna pierwszego rzędu funkcji f w pewnym otoczeniu $O(\hat{x})$ punktu $\hat{x}$.*
Jeżeli $\hat{x}$ jest punktem wewnętrznym zbioru $D = \langle c;d \rangle$, to $f'(\hat{x}) = 0$.
Jeżeli $\hat{x} = c$, to $f'(\hat{x}) \geqslant 0$. Jeżeli $\hat{x} = d$, to $f'(\hat{x}) \leqslant 0$.

Wniosek 1.7. *Jeżeli $D = \mathbb{R}$, to funkcja f może mieć minima lokalne tylko w tych punktach, w których pochodna funkcji jest równa zeru lub w których pochodna nie istnieje. W przeciwnym razie należy uwzględnić jeszcze punkty na brzegu zbioru D.*

Twierdzenie 1.8 (drugi warunek konieczny istnienia mimimum). *Niech funkcja f ma minimum lokalne w punkcie $\hat{x} \in D$ i niech istnieje ciągła pochodna drugiego rzędu funkcji f w pewnym otoczeniu $O(\hat{x})$ punktu $\hat{x}$.*
Jeżeli $f'(\hat{x}) = 0$, to $f''(\hat{x}) \geqslant 0$.

Twierdzenie 1.9 (warunek dostateczny istnienia mimimum). *Jeżeli istnieje ciągła pochodna drugiego rzędu funkcji f w pewnym otoczeniu $O(\hat{x})$ punktu $\hat{x}$ oraz jednocześnie:*

1° $f'(\hat{x}) = 0$
2° $f''(\hat{x}) > 0$

to funkcja f ma mimimum lokalne właściwe w punkcie $\hat{x}$.

Ważną klasą funkcji przy poszukiwaniu minimum są funkcje wypukłe.

Definicja 1.10. Funkcja f jest wypukła na zbiorze D $\iff$ *dla dowolnych $x_1, x_2 \in D$ i dowolnej liczby $\alpha \in (0;1)$ zachodzi nierówność:*

$$f(\alpha x_1 + (1-\alpha)x_2) \leqslant \alpha f(x_1) + (1-\alpha) f(x_2)$$

Jeżeli zamiast $\leqslant$ mamy $<$, to **funkcja f jest ściśle wypukła**.

Twierdzenie 1.11. *Jeżeli istnieje ciągła pochodna drugiego rzędu funkcji f, przy czym $f''(x) > 0$ dla $x \in D$, to funkcja f jest ściśle wypukła.*

Twierdzenie 1.12. *Jeżeli funkcja f jest wypukła na zbiorze D, to każde jej minimum lokalne jest jednocześnie minimum globalnym na tym zbiorze.*
Jeżeli funkcja f jest ściśle wypukła, to istnieje co najwyżej jedno minimum lokalne tej funkcji, które jest wówczas także jej mimimum globalnym.

Uwaga 1.2. Nie każda funkcja ściśle wypukła ma mimimum lokalne, np. funkcja $f(x) = \frac{1}{x}$ na zbiorze $D = \langle 1; +\infty)$ jest ściśle wypukła, ale nie ma mimimum.

W przypadku, gdy minimum funkcji nie może być wyznaczone analitycznie, należy wykorzystać odpowiednią metodę przybliżoną. Ogólnie rzecz biorąc, możemy podzielić takie metody na bezgradientowe, które znajdują zastosowanie w przypadku nieróżniczkowalnych funkcji celu oraz na metody gradientowe stosowane wtedy, gdy funkcja celu jest przynajmniej klasy C^1. Idea metod z pierwszej grupy opiera się przede wszystkim na definicji minimum funkcji 1.2, natomiast druga grupa metod wykorzystuje warunki konieczne optymalności 1.6 i ewentualnie 1.8. Opis wybranych metod bezgradientowych znajduje się w podrozdziale 1.3, a gradientowych w 1.4. Wynikiem działania przedstawionych metod jest zwykle wyznaczenie jedynie minimum lokalnego, a nie globalnego.

1.2. Lokalizacja rozwiązań

1.2.1. Metoda ekspansji

Jeżeli w zadaniu minimalizacji (1.1) zbiór D jest nieograniczony, tzn. $D = \mathbb{R}$ lub $c = -\infty$ lub $d = \infty$, to należy wyznaczyć przedział $\langle a; b \rangle$, gdzie $a > -\infty$, $b < \infty$, w którym to minimum występuje. Mówimy wtedy o lokalizacji rozwiązania zadania (1.1). Przeprowadzenie takiej lokalizacji jest konieczne ze względu na wymagania wielu metod przybliżonych wyznaczania minimum funkcji, dla których przedział lokalizacji lub przynajmniej punkt startowy położony w takim przedziale jest jedną z danych wejściowych. Do lokalizacji możemy wykorzystać metodę ekspansji, w której startujemy z pewnego przedziału $\langle x^{(0)}; x^{(1)} \rangle$ (bierzemy $x^{(1)} > x^{(0)}$, nie zakładamy, że w tym przedziale jest minimum). Jeżeli funkcja f jest unimodalna (tzn. jest ciągła i ma dokładnie jedno ekstremum), wystarczy przyjąć $x^{(0)} = 0$. Inaczej jest w przypadku, gdy funkcja f ma wiele ekstremów. Wówczas możemy zastosować np. zmodyfikowaną metodę ekspansji Boxa-Daviesa-Swanna. Poniżej przedstawiamy kolejne kroki niezmodyfikowanej metody ekspansji.

Obliczamy $f(x^{(0)})$ i $f(x^{(1)})$. Możliwe są trzy przypadki:

$1^\circ:\ f(x^{(0)}) = f(x^{(1)})$
Wówczas kończymy obliczenia, przyjmując $a = x^{(0)}$ i $b = x^{(1)}$.

$2^\circ:\ f(x^{(0)}) > f(x^{(1)})$
Oznacza to, że mimimum funkcji f leży na prawo od $x^{(0)}$, bierzemy zatem $x^{(2)} = \alpha x^{(1)}$, gdzie $\alpha > 1$ jest tzw. współczynnikiem ekspansji.

Jeżeli teraz $f(x^{(1)}) \leqslant f(x^{(2)})$, to znaczy, że funkcja f przestała maleć, a zaczęła rosnąć, zatem kończymy obliczenia, przyjmując $a = x^{(0)}$ i $b = x^{(2)}$.

Jeżeli przeciwnie, $f(x^{(1)}) > f(x^{(2)})$, to znaczy, że mimimum funkcji f leży na prawo od $x^{(1)}$, bierzemy zatem $x^{(3)} = \alpha x^{(2)}$ (w następnych krokach, jeżeli są potrzebne, bierzemy $x^{(4)} = \alpha x^{(3)}$, $x^{(5)} = \alpha x^{(4)}$ itd.) Kończymy obliczenia, gdy funkcja f zaczyna rosnąć, tzn. $f(x^{(i)}) \leqslant f(x^{(i+1)})$ dla pewnego $i \in \mathbb{N}$. Przyjmujemy wtedy $a = x^{(i-1)}$ i $b = x^{(i+1)}$.

$3^\circ: \ f(x^{(0)}) < f(x^{(1)})$
Oznacza to, że mimimum funkcji f leży na lewo od $x^{(1)}$, zmieniamy więc przedział $\langle x^{(0)}; x^{(1)} \rangle$ na $\langle -x^{(1)}; x^{(1)} \rangle$.

Jeżeli $f(-x^{(1)}) \geqslant f(x^{(0)})$, to znaczy, że funkcja $f(x)$ rośnie z obu stron $x^{(0)}$, zatem minimum jest w przedziale $\langle -x^{(1)}; x^{(1)} \rangle$. Przyjmujemy więc $a = -x^{(1)}$ oraz $b = x^{(1)}$ i kończymy obliczenia.

Jeżeli $f(-x^{(1)}) < f(x^{(0)})$, to znaczy, że mimimum funkcji f leży na lewo od $-x^{(0)}$, bierzemy zatem $x^{(2)} = \alpha x^{(1)}$.

Jeżeli teraz $f(-x^{(1)}) \leqslant f(-x^{(2)})$, to znaczy, że funkcja f przestała maleć, a zaczęła rosnąć (gdy poruszamy się w lewo), zatem kończymy obliczenia, przyjmując $a = -x^{(2)}$ i $b = x^{(0)}$.

Jeżeli przeciwnie, $f(-x^{(1)}) > f(-x^{(2)})$, to znaczy, że mimimum funkcji f leży na lewo od $-x^{(1)}$, bierzemy zatem $x^{(3)} = \alpha x^{(2)}$ itd.

Kończymy obliczenia, gdy funkcja f zaczyna rosnąć (gdy poruszamy się w lewo), tzn. $f(-x^{(i)}) \leqslant f(-x^{(i+1)})$ dla pewnego $i \in \mathbb{N}$. Przyjmujemy wtedy $a = -x^{(i+1)}$ i $b = -x^{(i-1)}$.

Przykład 1.1. Zlokalizować przedział, w którym funkcja $f(x) = x^2 - 5x + 6$ osiąga minimum. Przyjąć:
a) $x^{(1)} = 1, \alpha = 2$; b) $x^{(1)} = 1, \alpha = 5$; c) $x^{(1)} = 6, \alpha = 1{,}5$.

Rozwiązanie:
Ponieważ funkcja f jest unimodalna, możemy przyjąć $x^{(0)} = 0$.

Ad a)
$f(x^{(0)}) = f(0) = 6, \ f(x^{(1)}) = f(1) = 2$, więc $f(x^{(0)}) > f(x^{(1)})$ – obliczamy $x^{(2)} = \alpha x^{(1)} = 2$
$f(x^{(2)}) = f(2) = 0$, zatem $f(x^{(1)}) > f(x^{(2)})$ – obliczamy $x^{(3)} = \alpha x^{(2)} = 4$
$f(x^{(3)}) = f(4) = 2$, zatem $f(x^{(2)}) < f(x^{(3)})$ – kończymy obliczenia, przyjmując przedział lokalizacji minimum jako $\langle a; b \rangle = \langle x^{(1)}; x^{(3)} \rangle = \langle 1; 4 \rangle$

Ad b)
$f(x^{(0)}) = f(0) = 6, \ f(x^{(1)}) = f(1) = 2$, więc $f(x^{(0)}) > f(x^{(1)})$ – obliczamy $x^{(2)} = \alpha x^{(1)} = 5$

$f(x^{(2)}) = f(5) = 6$, zatem $f(x^{(1)}) < f(x^{(2)})$ – kończymy obliczenia, przyjmując przedział lokalizacji minimum jako $\langle a;b\rangle = \langle x^{(0)};x^{(2)}\rangle = \langle 0;5\rangle$

Ad c)
$f(x^{(0)}) = f(0) = 6$, $f(x^{(1)}) = f(6) = 12$, więc $f(x^{(0)}) < f(x^{(1)})$ – obliczamy
$f(-x^{(1)}) = f(-6) = 72$, więc $f(-x^{(1)}) > f(x^{(0)})$
Kończymy obliczenia, przyjmując przedział lokalizacji minimum jako $\langle a;b\rangle = \langle -x^{(1)};x^{(1)}\rangle = \langle -6;6\rangle$. □

W zmodyfikowanej metodzie ekspansji Boxa-Daviesa-Swanna możemy jako punkt startowy przyjąć dowolne $x^{(0)}$ (najlepiej w pobliżu minimum funkcji f). W kolejnych krokach tej metody obliczamy:

$$x^{(i+1)} = x^{(i)} + 2^i \Delta x \text{ dla } i = 0,1,2,... \tag{1.2}$$

gdzie Δx jest ustaloną liczbą dodatnią, aż zostanie spełniony warunek $f(x^{(i+1)}) \geqslant f(x^{(i)})$. Wtedy jako koniec przedziału lokalizacji minimum przyjmujemy $b = x^{(i+1)}$. Następnie obliczamy:

$$x^{(-i-1)} = x^{(-i)} - 2^i \Delta x \text{ dla } i = 0,1,2,... \tag{1.3}$$

aż będzie spełniony warunek $f(x^{(-i-1)}) \geqslant f(x^{(-i)})$. Wtedy jako początek przedziału lokalizacji minimum przyjmujemy $a = x^{(-i-1)}$.

Przykład 1.2. Wyznaczyć rozłączne przedziały o szerokości nie mniejszej niż 1, w których funkcja $f(x) = x^4 - 8x^3 + 14x^2 + 8x - 15$ osiąga ekstrema.

Rozwiązanie:
Funkcja f ma dwa minima i jedno maksimum (nie jest unimodalna). Zastosujemy więc metodę ekspansji Boxa-Daviesa-Swanna.
Narysujmy wykres funkcji f (patrz rysunek 1.1), aby zorientować się, gdzie należy szukać ekstremów.

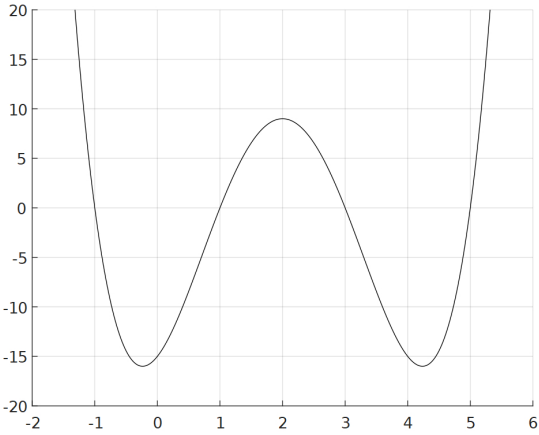
Wyznaczmy najpierw przedziały, w których leżą minima. Niech np. dla pierwszego minimum $x^{(0)} = 0$ i $\Delta x = 0,5$. Wtedy $f(x^{(0)}) = -15$. Otrzymujemy:

$$x^{(1)} = x^{(0)} + 2^0 \Delta x = 0,5, \quad f(x^{(1)}) = f(0,5) = -8,4375 > f(x^{(0)})$$

więc przyjmujemy $b = x^{(1)}$ i obliczamy:

$$x^{(-1)} = x^{(0)} - 2^0 \Delta x = -0,5, \quad f(x^{(-1)}) = f(-0,5) = -14,4375 > f(x^{(0)})$$

Przyjmujemy $a = x^{(-1)}$, zatem zlokalizowaliśmy pierwsze minimum na przedziale $\langle a;b\rangle = \langle -0,5;0,5\rangle$.



Rysunek 1.1: Wykres funkcji $f(x) = x^4 - 8x^3 + 14x^2 + 8x - 15$

Dla drugiego minimum możemy przyjąć np. $x^{(0)} = 4$ i $\Delta x = 0,5$. Wtedy $f(x^{(0)}) = -15$. Otrzymujemy:

$$x^{(1)} = x^{(0)} + 2^0 \Delta x = 4,5, \quad f(x^{(1)}) = f(4,5) = -14,4375 > f(x^{(0)})$$

więc przyjmujemy $b = x^{(1)}$. Obliczamy:

$$x^{(-1)} = x^{(0)} - 2^0 \Delta x = 3,5, \quad f(x^{(-1)}) = f(3,5) = -8,4375 > f(x^{(0)})$$

więc przyjmujemy $a = x^{(-1)}$, zatem zlokalizowaliśmy drugie minimum na przedziale $\langle a;b\rangle = \langle 3,5;4,5\rangle$.

Wyznaczenie maksimum funkcji f jest równoważne wyznaczeniu minimum funkcji $g = -f$. Biorąc pod uwagę tę ostatnią funkcję i przyjmując np. $x^{(0)} = 2, \Delta x = 0,5$, dostaniemy $g(x^{(0)}) = -9$ oraz

$$x^{(1)} = x^{(0)} + 2^0 \Delta x = 2,5, \quad g(x^{(1)}) = g(2,5) = -6,5625 > g(x^{(0)})$$

Przyjmujemy $b = x^{(1)}$. Obliczamy:

$$x^{(-1)} = x^{(0)} - 2^0 \Delta x = 1,5, \quad g(x^{(-1)}) = g(1,5) = -6,5625 > g(x^{(0)})$$

Przyjmujemy $a = x^{(-1)}$, zatem zlokalizowaliśmy maksimum na przedziale $\langle a;b\rangle = \langle 1,5;2,5\rangle$.

Otrzymaliśmy zatem trzy rozłączne przedziały o szerokości nie mniejszej niż 1, takie, że w każdym z nich jest dokładnie jedno ekstremum funkcji f. □

1.3. Metody bezgradientowe

1.3.1. Metoda złotego podziału

Zakładamy, że funkcja f zmiennej x jest ciągła oraz że jej minimum zostało zlokalizowane w przedziale $\langle a;b\rangle$. W metodzie złotego podziału dzielimy przedział $\langle a;b\rangle$ punktami $x^{(1)}$ i $x^{(2)}$ (przy czym $x^{(1)} < x^{(2)}$) w ten sposób, aby jednocześnie były spełnione poniższe warunki:

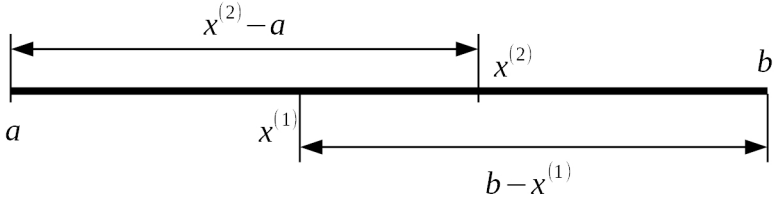
1. Długości przedziałów $\langle a;x^{(2)}\rangle$ oraz $\langle x^{(1)};b\rangle$ są równe, tzn.

$$x^{(2)} - a = b - x^{(1)} \tag{1.4}$$

2. Proporcje długości przedziałów $\langle x^{(1)};b\rangle$ i $\langle a;b\rangle$ oraz $\langle x^{(2)};b\rangle$ i $\langle x^{(1)};b\rangle$ są identyczne, tzn.

$$\frac{b-x^{(1)}}{b-a} = \frac{b-x^{(2)}}{b-x^{(1)}} = \alpha \tag{1.5}$$

gdzie α jest współczynnikiem proporcjonalności.



Rysunek 1.2: Złoty podział odcinka

Z warunku (1.5) wynika, że:

$$b-x^{(1)} = \alpha(b-a) \qquad \text{oraz} \qquad b-x^{(2)} = \alpha(b-x^{(1)}) \tag{1.6}$$

Po dodaniu stronami (1.4) oraz drugiej równości z (1.6) dostajemy:

$$b-a = (\alpha+1)(b-x^{(1)}) \tag{1.7}$$

Uwzględniając teraz pierwszą równość z (1.6), otrzymamy:

$$b-a = (\alpha+1)\alpha(b-a) \tag{1.8}$$

Stąd oczywiście $\alpha^2+\alpha-1=0$. Z konstrukcji złotego podziału odcinka wynika, że $\alpha \in (0;1)$, zatem wartość α jest określona jednoznacznie: $\alpha = \frac{\sqrt{5}-1}{2} \approx 0,618$.

Ze wzoru (1.4) oraz drugiej równości z (1.6) otrzymujemy:

$$\begin{aligned} x^{(1)} &= b-\alpha(b-a) = \\ &= \alpha a + (1-\alpha)b \end{aligned} \qquad \text{oraz} \qquad \begin{aligned} x^{(2)} &= a+\alpha(b-a) = \\ &= (1-\alpha)a + \alpha b \end{aligned} \tag{1.9}$$

W metodzie złotego podziału obliczamy wartości funkcji f w punktach $x^{(1)}$ i $x^{(2)}$.

Jeżeli $f(x^{(1)}) \leqslant f(x^{(2)})$, to zawężamy przedział $\langle a;b\rangle$ do przedziału $\langle a;x^{(2)}\rangle$, tzn. kolejno podstawiamy:

$$b = x^{(2)}, \qquad x^{(2)} = x^{(1)} \qquad \text{i} \qquad x^{(1)} = \alpha a + (1-\alpha)b$$

Jeżeli $f(x^{(1)}) > f(x^{(2)})$, to zawężamy przedział $\langle a;b\rangle$ do przedziału $\langle x^{(1)};b\rangle$, tzn. kolejno podstawiamy:

$$a = x^{(1)}, \qquad x^{(1)} = x^{(2)} \qquad \text{i} \qquad x^{(2)} = (1-\alpha)a + \alpha b$$

Powyższe obliczenia składają się na jedną iterację. Wykonujemy obliczenia do momentu, aż osiągniemy zadaną dokładność ε, tzn. aż będzie spełniony warunek $|b-a| < \varepsilon$. Wówczas jako przybliżone rozwiązanie x_{min} możemy przyjąć dowolną wartość z przedziału $\langle a;b\rangle$ wyznaczonego w ostatniej iteracji, np. $x_{min} = \frac{a+b}{2}$.

Przykład 1.3. Wykonać dwie iteracje metodą złotego podziału w celu wyznaczenia minimum funkcji $f(x) = -\frac{1}{|x|+1}$ na przedziale $\langle -1;2\rangle$.

Rozwiązanie:
Zgodnie z (1.9) obliczamy:

$$\begin{aligned} x^{(1)} &= -\alpha + 2(1-\alpha) \approx -0,618 + 2\cdot(1-0,618) = 0,146 \\ x^{(2)} &= -(1-\alpha) + 2\alpha \approx -0,382 + 2\cdot 0,618 = 0,854 \end{aligned}$$

zatem $f(x^{(1)}) \approx f(0,146) \approx -0,873$, $f(x^{(2)}) \approx f(0,854) \approx -0,539$. Ponieważ $f(x^{(1)}) < f(x^{(2)})$, więc przedział $\langle -1;2\rangle$ zawężamy do przedziału $\langle -1;0,854\rangle$. Jako pierwsze przybliżenie wartości x, dla której funkcja f osiąga minimum, przyjmujemy $x_{min} \approx \frac{-1+0,854}{2} = -0,073$ oraz $f_{min} = f(x_{min}) \approx -0,932$.

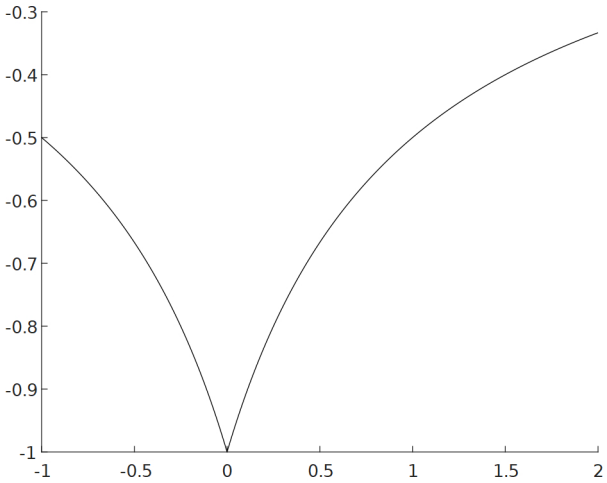
Przechodzimy do drugiej iteracji i w podobny sposób obliczamy:

$$\begin{aligned} x^{(1)} &\approx -\alpha + 0,854(1-\alpha) \approx -0,618 + 0,854\cdot(1-0,618) \approx -0,292 \\ x^{(2)} &\approx -(1-\alpha) + 0,854\alpha \approx -0,382 + 0,854\cdot 0,618 \approx 0,146 \end{aligned}$$

Ponieważ $f(x^{(1)}) \approx f(-0,292) \approx -0,774$, $f(x^{(2)}) \approx f(0,146) \approx -0,873$, więc $f(x^{(1)}) > f(x^{(2)})$, zatem tym razem przedział $\langle -1;0,854\rangle$ zawężamy do przedziału $\langle -0,292;0,854\rangle$. Jako drugie przybliżenie wartości x, dla której funkcja f osiąga

minimum, przyjmujemy $x_{min} \approx \frac{-0,292+0,854}{2} = 0,281$ oraz $f_{min} = f(x_{min}) \approx -0,781$.

Oczywiście ze wzoru funkcji f wynika, że jej minimum jest osiągane w punkcie $\hat{x}_{min} = 0$, przy czym $f(\hat{x}_{min}) = -1$.



Rysunek 1.3: Wykres funkcji $f(x) = -\frac{1}{|x|+1}$ na przedziale $\langle -1;2 \rangle$

W poniższej tabeli przedstawione zostały wyniki działania metody złotego podziału dla pierwszych dziesięciu iteracji:

Tabela 1.1: Wyniki działania metody złotego podziału dla funkcji $f(x) = -\frac{1}{|x|+1}$

Nr iteracji	a	b	x_{min}	ε
1	−1.00000	0.85410	−0.07295	1.85410
2	−0.29180	0.85410	0.28115	1.14590
3	−0.29180	0.41641	0.06231	0.70820
4	−0.29180	0.14590	−0.07295	0.43769
5	−0.12461	0.14590	0.01064	0.27051
6	−0.12461	0.04257	−0.04102	0.16718
7	−0.06075	0.04257	−0.00909	0.10333
8	−0.02129	0.04257	0.01064	0.06386
9	−0.02129	0.01818	−0.00155	0.03947
10	−0.00621	0.01818	0.00599	0.02440

Ponieważ w tym przykładzie znamy dokładną wartość $\hat{x}$, dla której funkcja f osiąga minimum, możemy podać dokładną wartość błędu po każdej iteracji jako $|x_{min} - \hat{x}| =$

$|x_{min}|$. Z przedostatniej kolumny tabeli 1.1 wynika, że błąd nie jest funkcją monotoniczną ilości iteracji i jest znacznie mniejszy od ε. □

Z konstrukcji metody wynika, że oszacowanie błędu ε po każdej iteracji zmniejsza się $\frac{1}{\alpha} \approx 1,618$ raza, zatem metoda złotego podziału jest zawsze zbieżna, a jednocześnie zbieżność ta jest dość wolna. Podstawową zaletą tej metody jest fakt, że można ją stosować w przypadku każdej funkcji ciągłej na przedziale $\langle a;b\rangle$, która ma na tym przedziale minimum lokalne.

1.3.2. Metoda interpolacji kwadratowej

Przyjmujemy takie same założenia jak w przypadku metody złotego podziału oraz dodatkowo zakładamy, że w pobliżu minimum funkcja f może być dobrze przybliżona przez funkcję kwadratową. Wewnątrz przedziału $\langle a;b\rangle$ lokalizacji minimum wybieramy punkt c, zwykle środek tego przedziału, tzn. $c = \frac{1}{2}(a+b)$, i konstruujemy wielomian Q_1 drugiego stopnia, którego wykres przechodzi przez punkty $(a,f(a)),\ (c,f(c)),\ (b,f(b))$, tzn. są spełnione poniższe warunki:

$$Q_1(a) = f(a),\ Q_1(c) = f(c),\ Q_1(b) = f(b) \tag{1.10}$$

Mamy wówczas do czynienia z tzw. interpolacją Lagrange'a funkcji f za pomocą wielomianu Q_1, który jest określony jednoznacznie przez (1.10). Wzór wielomianu ma postać:

$$\begin{aligned} Q_1(x) = f(a)\frac{(x-c)(x-b)}{(a-c)(a-b)} + f(c)\frac{(x-a)(x-b)}{(c-a)(c-b)} + \\ + f(b)\frac{(x-a)(x-c)}{(b-a)(b-c)} \end{aligned} \tag{1.11}$$

Wielomian Q_1 osiąga minimum w punkcie $x^{(1)}$, jeżeli jednocześnie $Q_1'(x^{(1)}) = 0$ oraz $Q_1''(x^{(1)}) > 0$. Z pierwszego z tych warunków wynika, że:

$$x^{(1)} = \frac{1}{2}\frac{(c^2-b^2)f(a) + (b^2-a^2)f(c) + (a^2-c^2)f(b)}{(c-b)f(a) + (b-a)f(c) + (a-c)f(b)} \tag{1.12}$$

a z drugiego:

$$\frac{f(a)}{(a-c)(a-b)} + \frac{f(c)}{(c-a)(c-b)} + \frac{f(b)}{(b-a)(b-c)} > 0 \tag{1.13}$$

Jeżeli zachodzi (1.13), to punkt $x^{(1)}$ określony przez (1.12) przyjmuje się jako pierwsze przybliżenie nieznanego punktu x_{min}, w którym funkcja f osiąga minimum. W zależności od położenia punktu $x^{(1)}$ początkowy przedział lokalizacji $\langle a;b\rangle$ ulega zmianie zgodnie z poniższym schematem.

Jeżeli $x^{(1)} < a$, to przesuwamy przedział $\langle a;b\rangle$ do $\langle x^{(1)};c\rangle$, tzn. kolejno podstawiamy:

$$b = c, \qquad c = a, \qquad a = x^{(1)}$$

Jeżeli $a < x^{(1)} < c$, to zawężamy przedział $\langle a;b\rangle$ do $\langle a;c\rangle$, tzn. kolejno podstawiamy:

$$b = c, \qquad c = x^{(1)}$$

Jeżeli $c < x^{(1)} < b$, to zawężamy przedział $\langle a;b\rangle$ do $\langle c;b\rangle$, tzn. kolejno podstawiamy:

$$a = c, \qquad c = x^{(1)}$$

Jeżeli $x^{(1)} > b$, to przesuwamy przedział $\langle a;b\rangle$ do $\langle c;x^{(1)}\rangle$, tzn. kolejno podstawiamy:

$$a = c, \qquad c = b, \qquad b = x^{(1)}$$

Po wyznaczeniu w ten sposób nowego przedziału $\langle a;b\rangle$ konstruujemy kolejny wielomian drugiego stopnia Q_2, wyznaczamy punkt $x^{(2)}$, w którym ten wielomian osiąga minimum, i sprawdzamy, czy $|x^{(2)} - x^{(1)}| < \varepsilon$ dla zadanej dokładności ε. Jeżeli ten warunek nie jest spełniony, wykonujemy kolejne iteracje, aż dla pewnego wielomianu Q_k otrzymamy $|x^{(k)} - x^{(k-1)}| < \varepsilon$. Wówczas kończymy obliczenia, przyjmując przybliżoną wartość x, dla której funkcja f osiąga minimum, jako $x_{min} = x^{(k)}$.

Przykład 1.4. Wykonać dwie iteracje metodą interpolacji kwadratowej dla danych z przykładu 1.3.

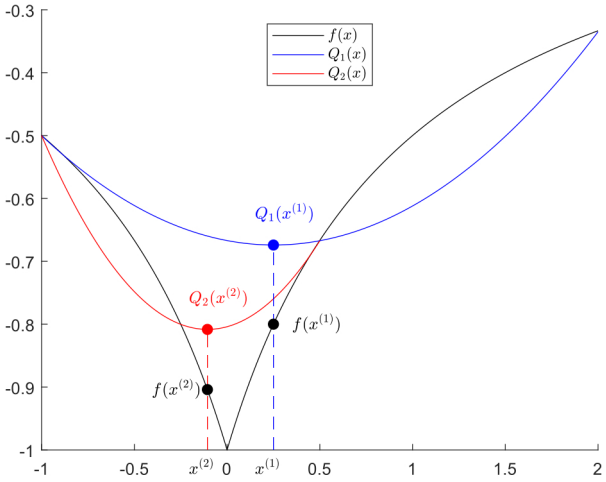
Rozwiązanie:
Przyjmujemy c jako środek przedziału lokalizacji minimum funkcji f, tzn. $c = \frac{1}{2}(-1+2) = 0,5$. Wyznaczenie postaci wielomianu Q_1 nie jest konieczne do wyznaczenia kolejnych przybliżeń rozwiązania optymalnego, ale będzie przydatne do lepszego zobrazowania sposobu działania metody na wykresie. Po podstawieniu danych do wzoru (1.11) otrzymujemy:

$$Q_1(x) = \frac{1}{9}x^2 - \frac{1}{18}x - \frac{2}{3} \approx 0,111x^2 - 0,056x - 0,667$$

Z kolei z (1.12) otrzymujemy pierwsze przybliżenie wartości x, dla której funkcja f osiąga minimum: $x^{(1)} = 0,25$, przy czym $f(x^{(1)}) = -0,8$.
Ponieważ $-1 = a < x^{(1)} < c = 0,5$, zawężamy przedział $\langle -1;2\rangle$ do $\langle -1;0,5\rangle$ i przechodzimy do drugiej iteracji. Nowa wartość $c = \frac{1}{2}(-1+0,5) = -0,25$, zatem z (1.11) ponownie otrzymujemy:

$$Q_2(x) \approx 0,385x^2 + 0,081x - 0,804$$

Z (1.12) dostajemy kolejne przybliżenie rozwiązania zadania: $x^{(2)} \approx -0,106$ i $f(x^{(2)}) \approx -0,904$.



Rysunek 1.4: Ilustracja działania metody interpolacji kwadratowej funkcji $f(x) = -\frac{1}{|x|+1}$ na przedziale $\langle -1;2 \rangle$

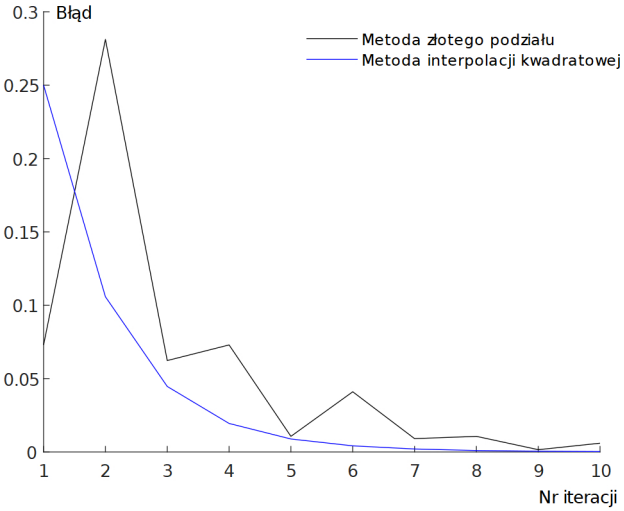
Wyniki działania metody interpolacji kwadratowej dla pierwszych dziesięciu iteracji zostały przedstawione w poniższej tabeli:

Tabela 1.2: Wyniki działania metody interpolacji kwadratowej dla funkcji $f(x) = -\frac{1}{|x|+1}$

Nr iteracji	a	b	x_{min}	ε
1	−1.00000	0.50000	0.25000	
2	−0.25000	0.50000	−0.10577	0.35577
3	−0.25000	0.12500	0.04464	0.15041
4	−0.06250	0.12500	−0.01943	0.06407
5	−0.06250	0.03125	0.00883	0.02826
6	−0.01562	0.03125	−0.00417	0.01300
7	−0.01562	0.00781	0.00202	0.00619
8	−0.00391	0.00781	−0.00099	0.00301
9	−0.00391	0.00195	0.00049	0.00149
10	−0.00098	0.00195	−0.00025	0.00074

Z wyników w tabeli 1.2 możemy wyciągnąć wniosek, że przynajmniej w przypadku rozważanym w przykładzie zarówno błąd $|x_{min} - \hat{x}| = |x_{min}|$, jak i jego oszacowanie ε maleją monotonicznie w zależności od ilości iteracji. □

Na wykresie poniżej przedstawiono faktyczne wartości błędów $|x_{min} - \hat{x}| = |x_{min}|$ zebrane z tabel 1.1 i 1.2.



Rysunek 1.5: Porównanie wartości błędów w metodach złotego podziału i interpolacji kwadratowej funkcji $f(x) = -\frac{1}{|x|+1}$ na przedziale $\langle -1;2 \rangle$

Otrzymane wyniki wskazują na szybszą zbieżność metody interpolacji kwadratowej.

1.4. Metody gradientowe

1.4.1. Metoda interpolacji sześciennej Davidona

Ideą tej metody jest przybliżenie funkcji f w przedziale $\langle a;b \rangle$, lokalizującym minimum funkcji, wielomianem trzeciego stopnia $Q_1(x) = Ax^3 + Bx^2 + Cx + D$. Na wielomian i funkcję nakładamy następujące warunki:

$$Q_1(a) = f(a),\ Q_1'(a) = f'(a),\ Q_1(b) = f(b),\ Q_1'(b) = f'(b) \tag{1.14}$$

Mamy tu do czynienia z tzw. interpolacją Hermite'a funkcji f za pomocą wielomianu Q_1. Jak można zauważyć, w warunkach (1.14) wykorzystujemy wartości f',

zatem powinniśmy założyć, że funkcja $f \in C^1(\mathbb{R})$. Warunki (1.14) są wystarczające, żeby określić współczynniki A,B,C,D wielomianu Q_1. Możemy je zapisać w postaci układu czterech równań liniowych, który ma jednoznaczne rozwiązanie:

$$\begin{cases} Aa^3+Ba^2+Ca+D = f(a) \\ 3Aa^2+2Ba+C \quad\;\; = f'(a) \\ Ab^3+Bb^2+Cb+D = f(b) \\ 3Ab^2+2Bb+C \quad\;\; = f'(b) \end{cases} \tag{1.15}$$

Po wyznaczeniu współczynników A,B,C,D z powyższego układu obliczamy wartość $x^{(1)}$, dla której wielomian Q_1 osiąga minimum. Można łatwo sprawdzić, że

$$x^{(1)} = \frac{-B+\sqrt{B^2-3AC}}{3A}$$

Obliczamy $f'(x^{(1)})$.

Jeżeli $f'(x^{(1)}) < 0$, to zawężamy przedział $\langle a;b\rangle$ do $\langle x^{(1)};b\rangle$, tzn. tworzymy nowy przedział $\langle a;b\rangle$, podstawiając $a = x^{(1)}$.

Jeżeli $f'(x^{(1)}) > 0$, to zawężamy przedział $\langle a;b\rangle$ do $\langle a;x^{(1)}\rangle$, tzn. tworzymy nowy przedział $\langle a;b\rangle$, podstawiając $b = x^{(1)}$.

Powyższe działania składają się na pierwszą iterację. Jeżeli $|f'(x^{(1)})| < \varepsilon$, gdzie ε jest zadaną dokładnością obliczeń (np. $\varepsilon = 0,001$), to kończymy obliczenia. W przeciwnym razie przechodzimy do kolejnej iteracji, konstruując na nowym przedziale $\langle a;b\rangle$ wielomian Q_2 itd.

Uwaga 1.3. Wraz ze zmniejszaniem się przedziału $\langle a;b\rangle$ w kolejnych iteracjach, wyznacznik układu równań (1.15) dąży do zera. Zadanie polegające na rozwiązaniu takiego układu jest wtedy źle uwarunkowane, tzn. mała zmiana wartości współczynników układu może powodować dużą zmianę rozwiązania. Oznacza to, że w takim przypadku współczynniki A,B,C,D wielomianu mogą zostać wyznaczone z dużym błędem, który będzie wynikał z błędów zaokrągleń współczynników układu równań.

Przykład 1.5. Zlokalizowano minimum funkcji $f(x) = -\frac{1}{x^2+1}$ w przedziale $\langle -1;2\rangle$. Wykonać jedną iterację metodą interpolacji sześciennej Davidona. Oszacować błąd.

Rozwiązanie:
Wyznaczamy współczynniki wielomianu Q_1 z układu równań:

$$\begin{cases} -A+B-C+D \quad = f(-1) \\ 3A-2B+C \quad\;\; = f'(-1) \\ 8A+4B+2C+D = f(2) \\ 12A+4B+C \quad\;\; = f'(2) \end{cases}$$

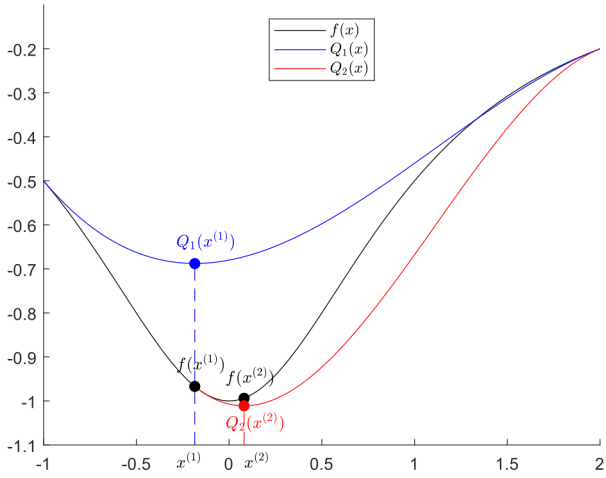
czyli

$$\begin{cases} -A+B-C+D &= -0,5 \\ 3A-2B+C &= -0,5 \\ 8A+4B+2C+D &= -0,2 \\ 12A+4B+C &= -0,16 \end{cases}$$

Stąd $A=-0,06,\ B=0,2,\ C=0,08,\ D=-0,68$, a zatem $Q_1(x)=-0,06x^3+0,2x^2+0,08x-0,68$. Wielomian Q_1 osiąga minimum dla:

$$x^{(1)}=\frac{-B+\sqrt{B^2-3AC}}{3A}\approx -0,185, \qquad \text{przy czym} \qquad f(x^{(1)})\approx -0,967$$

Obliczamy $f'(x^{(1)})\approx -0,345<0$, zatem zawężamy przedział $\langle -1;2\rangle$ do $\langle -0,185;2\rangle$. Wartość błędu to $|f'(x^{(1)})|\approx 0,345$.



Rysunek 1.6: Ilustracja działania metody interpolacji sześciennej Davidona funkcji $f(x)=-\frac{1}{x^2+1}$ na przedziale $\langle -1;2\rangle$

Tabela 1.3 przedstawia wyniki działania metody interpolacji sześciennej Davidona dla pierwszych dziesięciu iteracji. Podobnie jak poprzednio przyjęliśmy oznaczenie $x_{min}=x^{(k)}$ dla $k=1,\ldots$, tzn. przez x_{min} rozumiemy przybliżoną wartość x, dla której funkcja f osiąga minimum, otrzymaną po k-tej iteracji. Oczywiście testowana funkcja $f(x)=-\frac{1}{x^2+1}$ osiąga minimum dla $\hat{x}=0$, możemy zatem faktyczną wartość błędu obliczyć jako $|x_{min}-\hat{x}|=|x_{min}|$.

Tabela 1.3: Wyniki działania metody interpolacji sześciennej Davidona dla funkcji $f(x) = -\frac{1}{x^2+1}$

Nr iteracji	a	b	x_{min}	ε
1	-0.18466	2	-0.18466	0.34536
2	-0.18466	0.08056	0.08056	0.15906
3	-0.18466	0.00146	0.00146	0.00292
4	-0.18466	4.341×10^{-5}	4.341×10^{-5}	8.682×10^{-5}
5	-0.18466	1.297×10^{-6}	1.297×10^{-6}	2.594×10^{-6}
6	-0.18466	3.875×10^{-8}	3.875×10^{-8}	7.749×10^{-8}
7	-0.18466	1.158×10^{-9}	1.158×10^{-9}	2.315×10^{-9}
8	-0.18466	3.459×10^{-11}	3.459×10^{-11}	6.918×10^{-11}
9	-0.18466	1.033×10^{-12}	1.033×10^{-12}	2.067×10^{-12}
10	-0.18466	3.086×10^{-14}	3.086×10^{-14}	6.172×10^{-14}

□

Uwaga 1.4. Zaproponowany sposób szacowania błędu poprzez obliczanie wartości $\varepsilon = |f'(x^{(k)})|$ istotnie różni się od sposobu podanego wcześniej, tzn. poprzez obliczanie $|x^{(k)} - x^{(k-1)}|$. W przypadku stosowania wzoru $|x^{(k)} - x^{(k-1)}|$ może się zdarzyć, że kolejne przybliżenia $x^{(k-1)}$ i $x^{(k)}$ będą blisko siebie, ale daleko od $\hat{x}$, dla którego funkcja f osiąga minimum. Wówczas oczywiście wzór $|x^{(k)} - x^{(k-1)}|$ daje niewłaściwą informację o popełnionym błędzie. Stosowanie wzoru $|f'(x^{(k)})|$ może też prowadzić do niewłaściwych wniosków w przypadku, gdy funkcja f ma więcej niż jedno ekstremum lub gdy jej wykres ma asymptotę poziomą (wówczas $\lim_{x^{(k)} \to \pm\infty} |f'(x^{(k)})| = 0$).

1.4.2. Metoda siecznych

Zakładamy, że funkcja $f \in C^1(\mathbb{R})$. Korzystamy z faktu, że minimum funkcji f w takim przypadku może wystąpić tylko w takim punkcie $\hat{x}$, w którym spełnione jest równanie

$$f'(\hat{x}) = 0$$

Niech $\langle a;b \rangle$ oznacza przedział, na którym zostało zlokalizowane minimum funkcji f, czyli rozwiązanie powyższego równania. Prowadzimy sieczną wykresu funkcji $y = f'(x)$ przechodzącą przez punkty $(a, f'(a))$ i $(b, f'(b))$. Równanie tej siecznej ma postać:

$$y = \frac{f'(b) - f'(a)}{b - a}(x - a) + f'(a)$$

Punkt $x^{(1)}$ przecięcia osi poziomej przez tę sieczną przyjmujemy jako pierwsze przybliżenie rozwiązania równania $f'(\hat{x}) = 0$. Z równania siecznej otrzymujemy:

$$x^{(1)} = a - f'(a)\frac{b-a}{f'(b)-f'(a)}$$

Sprawdzamy, czy osiągnęliśmy wystarczającą dokładność (i w związku z tym możemy zakończyć obliczenia) – identycznie jak w metodzie interpolacji sześciennej. Jeżeli nie, to zmieniamy przedział $\langle a;b\rangle$ na:

- $\langle x^{(1)};b\rangle$ w przypadku, gdy $f'(x^{(1)}) < 0$ (wtedy w przedziale $\langle x^{(1)};b\rangle$ jest taki punkt x_{min}, że $f'(x_{min}) = 0$);
- $\langle a;x^{(1)}\rangle$ w przypadku, gdy $f'(x^{(1)}) > 0$ (wtedy w przedziale $\langle a;x^{(1)}\rangle$ jest taki punkt x_{min}, że $f'(x_{min}) = 0$).

Przykład 1.6. Wykonać dwie iteracje metodą siecznych dla danych z przykładu 1.5. Oszacować błąd po każdej iteracji.

Rozwiązanie:
Obliczamy $f'(x) = \frac{2x}{(x^2+1)^2}$. Ponieważ $f'(-1) = -0,5$, $f'(2) = 0,16$, prowadzimy przez punkty $(-1;-0,5)$ i $(2;0,16)$ pierwszą sieczną, która ma równanie:

$$y = 0,22x - 0,28$$

i przecina oś poziomą w punkcie $x^{(1)} = \frac{0,28}{0,22} \approx 1,273$. Otrzymujemy $f'(x^{(1)}) \approx 0,371$, zatem zmieniamy przedział $\langle -1;2\rangle$ na $\langle -1;1,273\rangle$. Błąd po pierwszej iteracji wynosi:

$$|f'(x^{(1)})| \approx 0,371$$

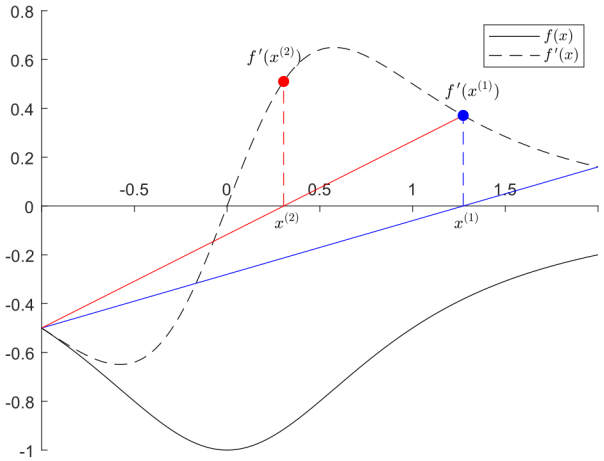
Następnie prowadzimy sieczną przez punkty $(-1;-0,5)$ oraz $(1,273;0,371)$ o równaniu:

$$y \approx 0,383x - 0,117$$

przecinającą oś poziomą w punkcie $x^{(2)} \approx \frac{0,117}{0,383} \approx 0,305$. Otrzymujemy $f'(x^{(2)}) \approx 0,510 > 0$, zatem zmieniamy przedział $\langle -1;1,273\rangle$ na $\langle -1;0,305\rangle$. Błąd po drugiej iteracji wynosi:

$$|f'(x^{(2)})| \approx 0,510$$

Tabela 1.4 przedstawia wyniki działania metody siecznych dla pierwszych ośmiu iteracji. W ósmej iteracji została osiągnięta wartość dokładna.



Rysunek 1.7: Ilustracja działania metody siecznych dla funkcji $f(x) = -\frac{1}{x^2+1}$ na przedziale $\langle -1; 2 \rangle$

Tabela 1.4: Wyniki działania metody siecznych dla funkcji $f(x) = -\frac{1}{x^2+1}$

Nr iteracji	a	b	x_{min}	ε
1	-1.00000	1.27273	1.27273	0.37087
2	-1.00000	0.30487	0.30487	0.51044
3	-0.35431	0.30487	-0.35431	0.55936
4	-0.00965	0.30487	-0.00965	0.01929
5	-0.00965	0.00181	0.00181	0.00361
6	-2.734	0.00181	-2.734×10^{-7}	5.467×10^{-7}
7	-2.734	1.784×10^{-12}	1.784×10^{-12}	3.567×10^{-12}
8	-2.734	0	0	0

□

1.4.3. Metoda Newtona

Przyjmujemy punkt startowy $x^{(0)}$ w pobliżu minimum funkcji f, wybierając np. początek albo koniec przedziału, na którym zostało zlokalizowane to minimum. Zakładamy, że funkcja $f \in C^2(\mathbb{R})$ oraz że $f''(x^{(0)}) > 0$. Oznacza to w szczególności, że funkcja f jest wypukła w pewnym otoczeniu punktu $x^{(0)}$. Przybliżamy funkcję f wielomianem drugiego stopnia Q_1, korzystając z rozwinięcia funkcji f w szereg Taylora wokół punktu $x^{(0)}$, tzn.:

$$Q_1(x) = f(x^{(0)}) + f'(x^{(0)})(x - x^{(0)}) + \frac{1}{2} f''(x^{(0)})(x - x^{(0)})^2$$

Punkt $x^{(1)}$, w którym wielomian Q_1 osiąga minimum, przyjmujemy jako pierwsze przybliżenie punktu $\hat{x}$, w którym funkcja f osiąga minimum. Można łatwo sprawdzić, że:

$$x^{(1)} = x^{(0)} - \frac{f'(x^{(0)})}{f''(x^{(0)})} \tag{1.16}$$

Sprawdzamy, czy osiągnęliśmy wystarczającą dokładność (i w związku z tym możemy zakończyć obliczenia) – identycznie jak w metodzie interpolacji sześciennej. Jeżeli nie, to przechodzimy do drugiej iteracji, wyznaczając kolejny wielomian drugiego stopnia Q_2 z rozwinięcia funkcji f wokół punktu $x^{(1)}$.

Uwaga 1.5. Z postaci wzoru (1.16) wynika, że wyznaczenie konkretnej postaci wielomianów $Q_1, Q_2, \ldots$ nie jest konieczne do przeprowadzenia obliczeń metodą Newtona, a jedynie jest przydatne do lepszego zrozumienia idei tej metody.

Przykład 1.7. Wykonać dwie iteracje metodą Newtona dla funkcji z przykładu 1.5. Jako punkt startowy przyjąć $x^{(0)} = 0,4$. Oszacować błąd po każdej iteracji.

Rozwiązanie:
Mamy $f'(x) = \frac{2x}{(x^2+1)^2}$ i $f''(x) = -\frac{6x^2-2}{(x^2+1)^3}$. Wynika stąd, że funkcja f jest wypukła w przedziale $\left(-\sqrt{\frac{1}{3}}; \sqrt{\frac{1}{3}}\right) \approx (-0,577; 0,577)$. Oczywiście $x^{(0)}$ należy do tego przedziału, zatem możemy wykonać pierwszą iterację. Wielomian Q_1 przybliżający funkcję f ma postać:

$$\begin{aligned} Q_1(x) &= f(0,4) + f'(0,4)(x-0,4) + \frac{1}{2}f''(-1)(x-0,4)^2 \approx \\ &\approx 0,333x^2 + 0,328x - 1,047 \end{aligned}$$

Minimum tego wielomianu jest osiągane w punkcie

$$x^{(1)} = x^{(0)} - \frac{f'(x^{(0)})}{f''(x^{(0)})} \approx \frac{-0,328}{0,666} \approx -0,492$$

Błąd po pierwszej iteracji wynosi:

$$|f'(x^{(1)})| \approx 0,638$$

Następnie konstruujemy wielomian:

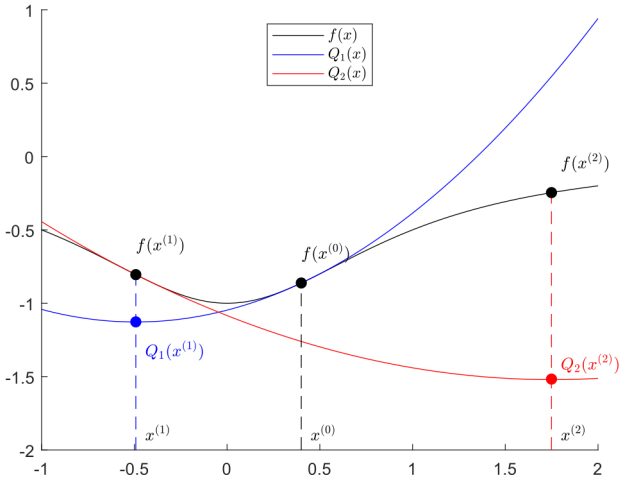
$$\begin{aligned} Q_2(x) &\approx f(-0,492) + f'(-0,492)(x+0,492) + \frac{1}{2}f''(-0,492)(x+0,492)^2 \approx \\ &\approx 0,142x^2 - 0,498x - 1,084 \end{aligned}$$

Minimum wielomianu Q_2 jest osiągane w punkcie

$$x^{(2)} = x^{(1)} - \frac{f'(x^{(1)})}{f''(x^{(1)})} \approx \frac{0,497}{0,285} \approx 1,749$$

Błąd po drugiej iteracji wynosi:

$$|f'(x^{(2)})| \approx 0,212$$



Rysunek 1.8: Ilustracja działania metody Newtona dla funkcji $f(x) = -\frac{1}{x^2+1}$

W tabeli 1.5 przedstawione zostały wyniki działania metody Newtona dla pierwszych dziesięciu iteracji. Kolumna ε_1 zawiera wartości błędu liczone jako $|f'(x^{(k)})|$. Liczby w tej kolumnie tworzą ciąg malejący do zera, co sugeruje, że zbliżamy się do minimum funkcji, jednak wartości w kolumnie x_{min} pokazują, że w istocie w kolejnych iteracjach oddalamy się od $\hat{x} = 0$, gdzie funkcja f faktycznie to minimum osiąga. Funkcja f ma asymptotę poziomą $y = 0$, zatem mamy do czynienia z sytuacją przedstawioną w uwadze 1.4. W kolumnie ε_2 zostały zapisane wartości błędu liczone jako $|x^{(k)}) - x^{(k-1)}|$. Począwszy od trzeciej iteracji liczby w tej kolumnie są coraz większe, co wskazuje na rozbieżność metody Newtona w tym przypadku. Można zauważyć, że wprawdzie dla dwóch pierwszych iteracji jest spełniony warunek wypukłości funkcji f w pewnych otoczeniach punktów $x^{(k)}$, jednak w następnych iteracjach już nie, gdyż $x^{(k)} \notin \left(-\sqrt{\frac{1}{3}}; \sqrt{\frac{1}{3}}\right)$ dla $k \geqslant 2$, zatem odpowiednie wielomiany Q_k dla $k \geqslant 2$ osiągają nie minimum, ale maksimum.

Tabela 1.5: Wyniki działania metody Newtona dla funkcji $f(x) = -\frac{1}{x^2+1}$ i punktu startowego $x^{(0)} = 0,4$

Nr iteracji	x_{min}	ε_1	ε_2
1	−0.492 31	0.637 92	0.892 31
2	1.748 91	0.212 34	2.241 22
3	2.617 11	0.084 96	0.868 18
4	3.668 06	0.035 11	1.050 87
5	5.014 96	0.014 67	1.346 90
6	6.776 30	0.006 16	1.761 44
7	9.101 14	0.002 59	2.324 83
8	12.183 88	0.001 09	3.082 74
9	16.281 73	0.000 46	4.097 95
10	21.736 31	0.000 19	5.454 68

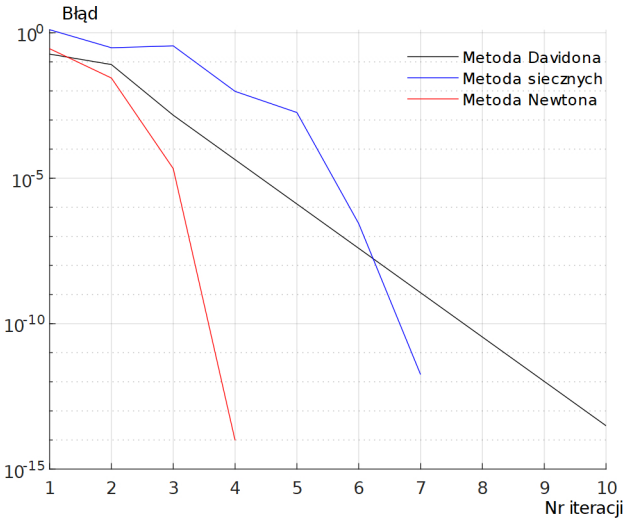
Odpowiednie obliczenia dla tej samej funkcji f i punktu startowego $x^{(0)} = 0,3$ przedstawiamy w tabeli 1.6. Tym razem metoda Newtona jest zbieżna, a wartości błędów liczone obydwoma sposobami szybko maleją do zera. Jak widać, dobór punktu startowego może mieć bardzo duże znaczenie przy poszukiwaniu ekstremum.

Tabela 1.6: Wyniki działania metody Newtona dla funkcji $f(x) = -\frac{1}{x^2+1}$ i punktu startowego $x^{(0)} = 0,3$

Nr iteracji	x_{min}	ε_1	ε_2
1	−0.147 95	0.283 35	0.447 95
2	0.013 86	0.027 72	0.161 81
3	-1.066×10^{-5}	2.133×10^{-5}	0.013 87
4	4.850×10^{-15}	9.700×10^{-15}	1.066×10^{-5}
5	0	0	4.850×10^{-15}
6	0	0	0

□

Poniższy wykres przedstawia faktyczne wartości błędów $|x_{min} - \hat{x}| = |x_{min}|$ w metodach interpolacji sześciennej Davidona, siecznych i Newtona zebrane z tabel 1.3, 1.4 i 1.6. Ze względu na czytelność prezentowanych wyników na osi *OY* zastosowano skalę logarytmiczną.



Rysunek 1.9: Porównanie wartości błędów w metodach interpolacji sześciennej Davidona, siecznych i Newtona zastosowanych do funkcji $f(x) = -\frac{1}{x^2+1}$

Otrzymane wyniki wskazują na to, że najszybciej zbieżna jest metoda Newtona. Zbieżność metody siecznych jest początkowo najwolniejsza, ale począwszy od siódmej iteracji otrzymujemy tą metodą wynik dokładniejszy niż przy zastosowaniu metody Davidona. Nie należy jednak na tej podstawie formułować ogólnych wniosków. Przekonuje nas o tym wcześniej analizowany przykład 1.7, w którym zbieżność metody była zależna od wyboru punktu startowego.

1.5. Procedury w MATLAB-ie

Podstawowe procedury w MATLAB-ie służące do rozwiązywania zadań optymalizacji są dostępne poprzez pakiet *Optimization Toolbox*. Zadania minimalizacji funkcji jednej zmiennej można rozwiązywać w MATLAB-ie przede wszystkim za pomocą procedury `fminbnd`. Szczegółowy opis składni i przykłady wykorzystania każdej ze standardowych procedur MATLAB-a możemy znaleźć w dokumentacji programu. W skrypcie zostaną przedstawione jedynie pewne możliwości zastosowania omawianych procedur.

Cechy charakterystyczne procedury `fminbnd`:

- funkcja celu powinna być ciągła;
- przed wywołaniem procedury należy zlokalizować przedział, na którym funkcja celu osiąga minimum;

- procedura wyznacza jedynie minima lokalne;
- w kolejnych iteracjach wykorzystywane są metody bezgradientowe: złotego podziału lub interpolacji kwadratowej.

Przykład 1.8. Korzystając z procedury `fminbnd`, wyznaczyć minimum funkcji $f(x) = -\frac{1}{|x|+1}$ dla $x \in \langle -1;2 \rangle$.

Rozwiązanie:
Przed zastosowaniem procedury `fminbnd` tworzymy plik **f.mlx** lub **f.m** zawierający definicję funkcji f:

Listing 1.1: Definicja funkcji f

```
function y=f(x)
  y = -1./(abs(x)+1);
end
```

W oddzielnym pliku umieszczamy wywołanie procedury `fminbnd`, np. w taki sposób:

Listing 1.2: Wywołanie procedury `fminbnd`

```
a = -1;
b = 2;
opcje = optimset('Display','iter','Maxiter',10);
[x,y] = fminbnd(@f,a,b,opcje)
```

W kodzie 1.2 wywoływana jest także procedura `optimset` umożliwiająca zmianę standardowych opcji przekazywanych na wejściu procedury `fminbnd`. Kolejne argumenty procedury `optimset` tworzą pary typu nazwa opcji i jej wartość. Para `'Display','iter'` oznacza, że procedura `fminbnd` wyświetli na ekranie nie tylko wynik końcowy, ale także wyniki pośrednie po każdej iteracji, natomiast para `'Maxiter',10` oznacza, że zostanie wykonanych tylko dziesięć iteracji. W wyniku działania powyższego kodu otrzymamy:

Listing 1.3: Wynik działania procedury `fminbnd`

```
 Func-count     x          f(x)          Procedure
     1       0.145898    -0.872678        initial
     2       0.854102    -0.539345        golden
     3      -0.291796    -0.774116        golden
     4       0.112461    -0.898908        parabolic
     5     -0.0278674    -0.972888        parabolic
     6      -0.128679    -0.885991        golden
     7    -0.00346001    -0.996552        parabolic
     8      0.0218814    -0.978587        parabolic
     9     -0.0012953    -0.998706        parabolic
    10     0.00439011    -0.995629        parabolic
    11     0.00087633    -0.999124        golden
```

```
Exiting: Maximum number of iterations has been exceeded
         - increase MaxIter option.
         Current function value: -0.999124
```

W pierwszej kolumnie wydruku jest podany numer iteracji, natomiast w ostatniej informacja o zastosowanej metodzie: złotego podziału (`golden`) lub interpolacji kwadratowej (`parabolic`). □

Uwaga 1.6. Do wyznaczenia minimum funkcji jednej zmiennej możemy wykorzystać także dwie inne procedury z pakietu *Optimization Toolbox*: `fminsearch` lub `fminunc`, jednak w odróżnieniu od procedury `fminbnd` przeznaczone są one przede wszystkim do wyznaczania minimum lokalnego funkcji wielu zmiennych, dlatego bardziej szczegółowo zostaną one przedstawione w następnym rozdziale.

1.6. Zadania do samodzielnego opracowania

1.6.1. Programowanie w MATLAB-ie

Zadanie 1.1. Utworzyć procedury realizujące metody: ekspansji Boxa-Daviesa-Swanna, złotego podziału, interpolacji kwadratowej, interpolacji sześciennej Davidona, siecznych i Newtona.

Wywołanie procedur:

- `[a,b]=m_ekspansji(@f,t0,Delta_t)`, gdzie `f` jest nazwą procedury obliczającej wartość funkcji celu, `t0`, `Delta_t` oznaczają parametry zmodyfikowanej metody ekspansji Boxa-Daviesa-Swanna, `a` i `b` stanowią odpowiednio lewy i prawy koniec przedziału lokalizacji minimum;
- `blad = m_zloty_podzial(@f,a,b,n)`, gdzie `n` – ilość iteracji do wykonania, `blad` – wektor wartości oszacowań błędu po kolejnych iteracjach;
- `blad = m_int_kwad(@f,a,b,n);`
- `blad = m_Davidon(@f,@fp,a,b,n)`, gdzie `fp` jest nazwą procedury obliczającej wartość pierwszej pochodnej funkcji celu;
- `blad = m_siecznych(@f,@fp,a,b,n);`
- `blad = m_Newtona(@f,@fp,@fd,x0,n)`, gdzie `fd` jest nazwą procedury obliczającej wartość drugiej pochodnej funkcji celu, a `x0` jest punktem startowym.

Każda z procedur, z wyjątkiem metody ekspansji, powinna wyświetlać na ekranie wyniki po każdej iteracji.

1.6.2. Sprawozdanie

Każda z poniższych funkcji ma przynajmniej jedno minimum lokalne na podanym przedziale:

1. $f(x) = x + \sin x,\ x \in \langle -5;5\rangle$
2. $f(x) = x - \cos x,\ x \in \langle -5;5\rangle$
3. $f(x) = x\cos x,\ x \in \langle -5;5\rangle$
4. $f(x) = x^2 \sin x,\ x \in \langle -3;3\rangle$
5. $f(x) = x^2 \cos x,\ x \in \langle -3;5\rangle$
6. $f(x) = -xe^{-x},\ x \in \langle -1;5\rangle$
7. $f(x) = e^x \sin x,\ x \in \langle -3;3\rangle$
8. $f(x) = e^x \cos x,\ x \in \langle -2;5\rangle$
9. $f(x) = \frac{\sin x}{x^2+1},\ x \in \langle -3;3\rangle$
10. $f(x) = \frac{\cos x}{x^2+1},\ x \in \langle -3;3\rangle$
11. $f(x) = (\sin x)\ln(x^2+x+1)$, $x \in \langle -2;10\rangle$
12. $f(x) = (\cos x)\ln(x^2+x+1)$, $x \in \langle -2;10\rangle$
13. $f(x) = x\sin(10x),\ x \in \langle 0;10\rangle$
14. $f(x) = x^2 \sin(10x),\ x \in \langle -5;0\rangle$
15. $f(x) = x^2 \ln(x^2+x+1),\ x \in \langle -2;10\rangle$
16. $f(x) = \sin x^2,\ x \in \langle -3;3\rangle$
17. $f(x) = -\cos x^2,\ x \in \langle -2;3\rangle$
18. $f(x) = x^2 \sin x^2,\ x \in \langle 0;5\rangle$
19. $f(x) = x^2 \cos x^2,\ x \in \langle 0;5\rangle$
20. $f(x) = x\sin x^2 + x^2\cos x$, $x \in \langle -5;5\rangle$
21. $f(x) = x^2 \sin x + x\cos x^2$, $x \in \langle -5;5\rangle$

Zadanie 1.2. Narysować wykres funkcji f danej powyższym wzorem na zadanym przedziale. Przetestować standardową procedurę MATLAB-a `fminbnd` dla tej funkcji i przedziału (nie należy dokonywać lokalizacji poszczególnych ekstremów). Czy wynikiem działania procedury jest najmniejsza wartość na zadanym przedziale?

Poniżej podane są wzory wielomianów czwartego stopnia. Każdy z nich ma trzy różne ekstrema lokalne.

1. $f(x) = x^4 - 4x^3 - 2x^2 + 16x + 5$
2. $f(x) = x^4 - 8x^2 + 4x - 3$
3. $f(x) = 3x^4 - 16x^3 + 48x + 2$
4. $f(x) = 3x^4 + 2x^3 - 15x^2 - 18x - 5$
5. $f(x) = 3x^4 + 14x^3 + 9x^2 - 30x + 1$
6. $f(x) = 3x^4 - 22x^3 + 45x^2 - 30x - 7$
7. $f(x) = 6x^4 + 40x^3 + 69x^2 - 18x + 4$
8. $f(x) = 3x^4 + 38x^3 + 165x^2 + 270x - 3$
9. $f(x) = 3x^4 - 8x^3 - 30x^2 + 60x + 6$
10. $f(x) = 3x^4 - 32x^3 + 90x^2 - 12x - 1$
11. $f(x) = x^4 - 12x^3 + 46x^2 - 60x + 2$
12. $f(x) = x^4 - 8x^3 - 8x^2 + 100x - 3$
13. $f(x) = 21x^4 - 32x^3 - 36x^2 + 60x + 4$
14. $f(x) = 3x^4 + 4x^3 - 36x^2 + 36x - 5$
15. $f(x) = 3x^4 + 4x^3 - 12x^2 + 6x + 6$
16. $f(x) = 5x^4 + 4x^3 - 6x^2 - 4x - 7$
17. $f(x) = 15x^4 + 4x^3 - 24x^2 - 12x - 1$
18. $f(x) = 3x^4 + 2x^3 - 9x^2 - 6x + 2$
19. $f(x) = 9x^4 - 20x^3 - 18x^2 + 12x - 3$
20. $f(x) = 3x^4 - 4x^3 - 9x^2 + 12x + 4$
21. $f(x) = x^4 + 2x^3 - 2x^2 - x - 5$

Zadanie 1.3. Zlokalizować wszystkie ekstrema ustalonej funkcji f danej powyższym wzorem metodą ekspansji Boxa-Daviesa-Swanna (dobrać tak parametry tej metody,

aby powstały rozłączne przedziały lokalizacji o szerokości nie mniejszej niż 0,5). Następnie wyznaczyć przybliżone wartości punktów, w których funkcja f osiąga ekstremum, stosując dwie wybrane metody gradientowe i metodę bezgradientową implementowaną przez standardową procedurę MATLAB-a `fminbnd`. Przyjąć różne dopuszczalne wartości oszacowania błędu: $\varepsilon = 10^{-1}$, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}. Rezultaty obliczeń przedstawić w odpowiednich tabelach i na wykresach. Na podstawie otrzymanych wyników porównać szybkość zbieżności poszczególnych metod.

Postać tabeli:

Rodzaj ekstremum: (minimum/maksimum)
Parametry metody ekspansji:
Przedział lokalizacji ekstremum:

ε	Metoda ...			Metoda ...			Metoda `fminbnd`		
	Ilość iteracji	x_{min}	$f(x_{min})$	Ilość iteracji	x_{min}	$f(x_{min})$	Ilość iteracji	x_{min}	$f(x_{min})$
10^{-1}									
10^{-2}									
10^{-3}									
10^{-4}									
10^{-5}									

W odpowiednim wierszu tabeli należy umieścić wyniki otrzymane po tej iteracji, w której po raz pierwszy oszacowanie błędu było nie większe niż ε.

Wykres powinien przedstawiać zależność funkcji oszacowania błędu ε od ilości iteracji n. W jednym układzie współrzędnych należy umieścić trzy wykresy dla danego ekstremum, po jednym dla każdej z metod (na osi pionowej należy zastosować skalę logarytmiczną). Jeżeli to możliwe, należy przyjąć taką maksymalną liczbę iteracji n_{max}, aby $\varepsilon(n_{max}) \leqslant 10^{-10}$.

Rozdział 2

Programowanie nieliniowe: minimalizacja funkcji wielu zmiennych bez ograniczeń

2.1. Warunki konieczne i dostateczne optymalności

W tym rozdziale będziemy rozważać problem minimalizacji funkcji f wielu zmiennych, który możemy zapisać następująco:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \tag{2.1}$$

gdzie $\mathbf{x} = [x_1, x_2, \ldots x_n]^T$, a $f : \mathbb{R}^n \to \mathbb{R}$ jest funkcją nieliniową. Takie zadanie będziemy nazywać zadaniem programowania nieliniowego bez ograniczeń i oznaczać (ZPNbo).

Podobnie jak w przypadku funkcji jednej zmiennej możemy sformułować warunki optymalności dla powyższego zadania. Odpowiednie definicje i twierdzenia wymagają uogólnienia dla przypadku wielowymiarowego – ich nowe wersje, zresztą dobrze znane z kursu analizy matematycznej, przypominamy poniżej.

Definicja 2.1. Sąsiedztwem $S(\mathbf{x}_0, \delta)$ **punktu** $\mathbf{x}_0 \in \mathbb{R}^n$ **o promieniu** δ *nazywamy zbiór wszystkich takich* $\mathbf{x} \in \mathbb{R}^n$*, że* $||\mathbf{x} - \mathbf{x}_0|| < \delta$*, przy czym* $\mathbf{x} \neq \mathbf{x}_0$.
Otoczeniem $O(\mathbf{x}_0, \delta)$ **punktu** $\mathbf{x}_0 \in \mathbb{R}^n$ *nazywamy zbiór:*

$$O(\mathbf{x}_0, \delta) = S(\mathbf{x}_0, \delta) \cup \mathbf{x}_0$$

Definicje ekstremów 1.2–1.5 z rozdziału 1 pozostają identyczne, oczywiście z uwzględnieniem nowej definicji sąsiedztwa i otoczenia punktu, przy czym przyjmujemy $D = \mathbb{R}^n$.

Twierdzenie 2.2 (pierwszy warunek konieczny istnienia mimimum). *Jeżeli funkcja f ma mimimum lokalne w punkcie $\hat{\mathbf{x}}$ i istnieją ciągłe pochodne cząstkowe pierwszego rzędu funkcji f w pewnym otoczeniu $O(\hat{\mathbf{x}})$ punktu $\hat{\mathbf{x}}$, to $\nabla f(\hat{\mathbf{x}}) = \mathbf{0}$.*

Wniosek 2.3. *Funkcja f może mieć minima lokalne tylko w tych punktach, w których gradient funkcji jest wektorem zerowym lub w których gradient nie istnieje.*

Twierdzenie 2.4 (drugi warunek konieczny istnienia mimimum). *Jeżeli funkcja f ma mimimum lokalne w punkcie $\hat{\mathbf{x}}$ i istnieją ciągłe pochodne cząstkowe drugiego rzędu funkcji f w pewnym otoczeniu $O(\hat{\mathbf{x}})$ punktu $\hat{\mathbf{x}}$, to hesjan $\mathbf{H}$ funkcji f w punkcie $\hat{\mathbf{x}}$ jest nieujemnie określony, tzn.*

$$\mathbf{d}^T \mathbf{H} f(\hat{\mathbf{x}}) \mathbf{d} \geqslant 0 \qquad \textit{dla dowolnych} \qquad \mathbf{d} \in \mathbb{R}^n$$

Twierdzenie 2.5 (warunek dostateczny istnienia mimimum). *Jeżeli istnieją ciągłe pochodne cząstkowe drugiego rzędu funkcji f w pewnym otoczeniu $O(\hat{\mathbf{x}})$ punktu $\hat{\mathbf{x}}$ oraz jednocześnie:*

1° $\nabla(\hat{\mathbf{x}}) = \mathbf{0}$,
2° *hesjan funkcji f w punkcie $\hat{\mathbf{x}}$ jest dodatnio określony, tzn.*

$$\mathbf{d}^T \mathbf{H} f(\hat{\mathbf{x}}) \mathbf{d} > 0 \qquad \text{dla dowolnych} \qquad \mathbf{d} \in \mathbb{R}^n,\ \mathbf{d} \neq \mathbf{0}$$

to funkcja f ma mimimum lokalne w punkcie $\hat{\mathbf{x}}$.

Z twierdzenia Schwarza o pochodnych mieszanych funkcji $f \in C^2$ wynika, że dla dowolnego $\mathbf{x} \in \mathbb{R}^n$ hesjan $\mathbf{H} f(\mathbf{x})$ jest macierzą symetryczną.

Twierdzenie 2.6. *Następujące warunki są równoważne:*

1° $\mathbf{H} f(\mathbf{x})$ *jest macierzą dodatnio określoną;*
2° *wszystkie minory główne macierzy $\mathbf{H} f(\mathbf{x})$ są dodatnie;*
3° *istnieje rozkład Cholesky'ego macierzy $\mathbf{H} f(\mathbf{x})$ na iloczyn $\mathbf{L}\mathbf{L}^T$, gdzie $\mathbf{L}$ jest macierzą trójkątną dolną, której wszystkie elementy na głównej przekątnej są dodatnie.*

Powyższe twierdzenie pozwala na łatwe sprawdzenie warunku dodatniej określoności macierzy. W przypadku, gdy stopień macierzy nie jest zbyt duży i możemy wykonać obliczenia analitycznie, korzystamy zwykle z warunku 2°. Jeżeli obliczenia wykonywane są na komputerze, łatwiejsze jest sprawdzenie warunku 3°.

2.2. Metody poszukiwań prostych (bezgradientowe)

W przypadku, gdy f jest funkcją nieróżniczkowalną, możemy zastosować do wyznaczenia przybliżonego rozwiązania zadania (2.1) jakąś metodę bezgradientową. Takie metody wykorzystują jedynie wartości funkcji celu, w przeciwieństwie do metod gradientowych, które wymagają wyznaczenia w każdej iteracji gradientu, a w niektórych przypadkach także hesjanu funkcji celu. Ważną grupę wśród metod bezgradientowych stanowią tzw. metody poszukiwań prostych, do których zalicza się m.in. metody:

- sympleksu Neldera-Meada,
- Hooke'a-Jeevesa,
- Rosenbrocka,
- Powella.

Szczegółowy opis algorytmów tych metod można znaleźć np. w Findeisen et al. (1980). Ze względu na to, że metoda Neldera-Meada została zaimplementowana w MATLAB-ie, poniżej zostanie przedstawiony jej algorytm.

W metodzie Neldera-Meada wykorzystuje się pojęcie sympleksu, który rozumiemy jako wielościan w przestrzeni $\mathbb{R}^n$ o najmniejszej możliwej liczbie wierzchołków, tzn. $n+1$. W przypadku, gdy minimalizujemy funkcję dwóch zmiennych, to $n=2$, zatem sympleksem jest trójkąt, w przypadku funkcji trzech zmiennych sympleksem jest czworościan.

Algorytm metody:

1° Generujemy $n+1$ wierzchołków $\mathbf{x}_1,\ldots,\mathbf{x}_{n+1}$ sympleksu tak, aby był on wielościanem foremnym lub zbliżonym do foremnego.

2° Obliczamy wartości funkcji f we wszystkich wierzchołkach i przenumerowujemy je tak, aby większemu numerowi odpowiadała wyższa wartość funkcji f, tzn.

$$f(\mathbf{x}_1) \leqslant f(\mathbf{x}_2) \leqslant \ldots \leqslant f(\mathbf{x}_n) \leqslant f(\mathbf{x}_{n+1})$$

3° Wyznaczamy środek ciężkości $\mathbf{c}$ sympleksu w przestrzeni $\mathbb{R}^{n-1}$, który powstaje z sympleksu dotychczasowego przez odrzucenie najgorszego wierzchołka, tzn. $\mathbf{x}_{n+1}$. Otrzymujemy:

$$\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$$

4° Zastępujemy punkt $\mathbf{x}_{n+1}$ punktem $\mathbf{x}_r$ leżącym na półprostej o początku w $\mathbf{x}_{n+1}$ i przechodzącej przez $\mathbf{c}$, tzn.

$$\mathbf{x}_r = \mathbf{c} + \alpha(\mathbf{c} - \mathbf{x}_{n+1})$$

Współczynnik α ustalamy w zależności od wyniku porównania wartości $f(\mathbf{x}_r)$ z wartościami $f(\mathbf{x}_1)$, $f(\mathbf{x}_n)$ i $f(\mathbf{x}_{n+1})$. Najpierw bierzemy $\alpha = 1$ (odpowiada to odbiciu symetrycznemu punktu $\mathbf{x}_{n+1}$ względem punktu $\mathbf{c}$). Możliwe są cztery przypadki:

a) $f(\mathbf{x}_r) < f(\mathbf{x}_1)$
Oznacza to, że punkt $\mathbf{x}_r$ jest lepszy od dotychczas najlepszego, w związku z czym warto zwiększyć α, bo być może w tym kierunku dostaniemy jeszcze lepszy punkt (odpowiada to operacji ekspansji). Przyjmujemy $\alpha = 2$, otrzymując punkt $\mathbf{x}_s$. Jeżeli faktycznie $f(\mathbf{x}_s) < f(\mathbf{x}_r)$, to $\mathbf{x}_{n+1}$ zastępujemy punktem $\mathbf{x}_s$. Jeżeli jednak $\mathbf{x}_s$ nie jest lepszy od $\mathbf{x}_r$, czyli $f(\mathbf{x}_r) \leqslant f(\mathbf{x}_s)$, to $\mathbf{x}_{n+1}$ zastępujemy punktem $\mathbf{x}_r$.

b) $f(\mathbf{x}_1) \leqslant f(\mathbf{x}_r) < f(\mathbf{x}_n)$
Pozostajemy przy $\alpha = 1$ i $\mathbf{x}_{n+1}$ zastępujemy punktem $\mathbf{x}_r$.

c) $f(\mathbf{x}_n) \leqslant f(\mathbf{x}_r) < f(\mathbf{x}_{n+1})$
W tym przypadku $\mathbf{x}_r$ jest wprawdzie lepsze od $\mathbf{x}_{n+1}$, które chcemy odrzucić,

ale gorsze od najgorszego punktu $\mathbf{x}_n$, który ma pozostać w następnej iteracji. W takim przypadku przyjmujemy $\alpha = 0,5$ (odpowiada to operacji tzw. ściagnięcia na zewnątrz), otrzymując punkt $\mathbf{x}_z$. Jeżeli $f(\mathbf{x}_z) < f(\mathbf{x}_r)$, to $\mathbf{x}_{n+1}$ zastępujemy punktem $\mathbf{x}_z$. W przeciwnym razie przechodzimy do punktu 6°.

d) $f(\mathbf{x}_r) \geqslant f(\mathbf{x}_{n+1})$
Wówczas $\mathbf{x}_r$ jest gorsze od punktu $\mathbf{x}_{n+1}$, który ma zostać usunięty z sympleksu. Bierzemy teraz $\alpha = -0,5$ (odpowiada to operacji tzw. ściagnięcia do wewnątrz), otrzymując punkt $\mathbf{x}_w$. Jeżeli $f(\mathbf{x}_w) < f(\mathbf{x}_{n+1})$, to $\mathbf{x}_{n+1}$ zastępujemy punktem $\mathbf{x}_w$. W przeciwnym razie przechodzimy do punktu 6°.

5° Sprawdzamy, czy sympleks jest wystarczająco mały, tzn., czy

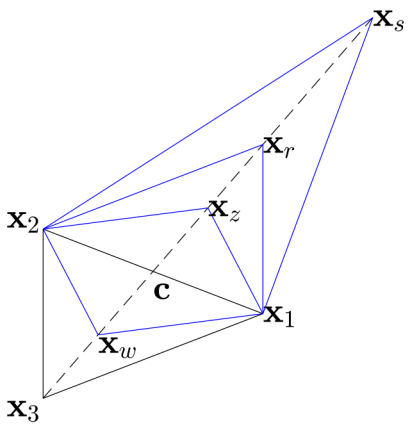
$$\max_{i=2,\ldots,n+1} \|\mathbf{x}_i - \mathbf{x}_1\| < \varepsilon$$

dla zadanej dokładności ε, gdzie $\|\cdot\|$ oznacza normę euklidesową. Jeżeli tak, to kończymy obliczenia, przyjmując jako rozwiązanie optymalne $\mathbf{x}_1$. W przeciwnym razie wracamy do punktu 2°.

6° Zostawiamy tylko najlepszy wierzchołek, czyli $\mathbf{x}_1$, a pozostałe zmieniamy według wzoru:

$$\mathbf{x}_i = \mathbf{x}_1 + 0,5 \cdot (\mathbf{x}_i - \mathbf{x}_1),\ i = 2,\ldots,n+1$$

co odpowiada zmniejszeniu sympleksu. Wracamy do punktu 5°.



Rysunek 2.1: Sposób modyfikacji sympleksu w metodzie Nealdera-Meada (przypadek $n = 2$)

2.3. Metody kierunków poprawy (gradientowe)

Do wyznaczenia $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, gdzie $\mathbf{x} = [x_1, \ldots, x_n]^T$, a $f : \mathbb{R}^n \to \mathbb{R}$ jest funkcją nieliniową, wykorzystuje się często metody optymalizacji z grupy tzw. metod kierunków poprawy. Zaliczamy do nich m.in.:

- metodę najszybszego spadku;
- wielowymiarową metodę Newtona;
- metody gradientów sprzężonych, np. standardową, Fletchera-Reevesa, Polaka-Ribiere'a;
- metody zmiennej metryki, czyli tzw. quasi-newtonowskie, np. Davidona-Fletchera-Powella (DFP) czy Broydena-Fletchera-Goldfarba-Shanno (BFGS).

Idea wszystkich tych metod jest podobna. Postępujemy w nich według poniższego schematu:

1° Przyjmujemy punkt startowy $\mathbf{x}^{(0)} \in \mathbb{R}^n$.

2° Wyznaczamy w punkcie startowym kierunek poprawy $\mathbf{d}^{(0)}$, tzn. kierunek, w którym wartość funkcji f maleje (wykorzystujemy w tym celu gradient funkcji f, a w niektórych metodach także hesjan).

3° Minimalizujemy funkcję f w wyznaczonym kierunku, tzn. szukamy $\min_{\tau\in\mathbb{R}} f(\mathbf{x}^{(0)} + \tau\mathbf{d}^{(0)})$. Jest to zadanie minimalizacji funkcji jednej zmiennej $g(\tau) = f(\mathbf{x}^{(0)} + \tau\mathbf{d}^{(0)})$, a do jego przybliżonego rozwiązania możemy zastosować poznane w rozdziale 1 metody.

4° Zakładając, że rozwiązanie zadania minimalizacji funkcji jednej zmiennej τ jest osiągane dla $\hat{\tau}^{(0)}_{min}$, określamy kolejne przybliżenie rozwiązania pierwotnego zadania optymalizacji $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$ jako $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \hat{\tau}^{(0)}_{min}\mathbf{d}^{(0)}$.

5° Obliczamy błąd przybliżenia jako $||\nabla f(\mathbf{x}^{(1)})||$. Jeżeli jest wystarczająco mały, kończymy obliczenia, przyjmując $\mathbf{x}^{(1)}$ jako rozwiązanie optymalne. W przeciwnym razie wracamy do punktu 2°, przyjmując $\mathbf{x}^{(1)}$ zamiast $\mathbf{x}^{(0)}$ itd.

Proces określony w punkcie 3° nazywamy ***minimalizacją kierunkową***. W i-tej iteracji wartość $\hat{\tau}^{(i)}_{min}$ minimalizującą funkcję $g(\tau) = f(\mathbf{x}^{(i)} + \tau\mathbf{d}^{(i)})$ teoretycznie można by uzyskać z warunku $|g'(\hat{\tau}^{(i)}_{min})| = 0$. Dokładne spełnienie tego warunku przy wykonywaniu obliczeń numerycznych jest jednak praktycznie niemożliwe. Możemy zatem zamiast tego sprawdzić, czy $|g'(\tau)| \leqslant \varepsilon$ dla pewnej małej dodatniej wartości ε. Ponieważ $\mathbf{d}^{(i)}$ jest kierunkiem poprawy, dla wszystkich τ z pewnego przedziału $(0; \tau_R)$ zachodzi $g(\tau) < g(0)$. Możemy zatem na podstawie tych dwóch warunków podjąć decyzję o tym, czy bieżąca wartość τ jest wystarczająco dobrym przybliżeniem $\hat{\tau}^{(i)}_{min}$, sprawdzając to w następujący sposób:

1. Jeżeli $g(\tau) < g(0)$ i $|g'(\tau)| \leqslant \varepsilon$, to przyjmujemy τ jako przybliżenie $\hat{\tau}^{(i)}_{min}$.

2. Jeżeli $g(\tau) \geqslant g(0)$ lub $g'(\tau) > \varepsilon$, to należy zmniejszyć τ.
3. Jeżeli $g(\tau) < g(0)$ i $g'(\tau) < -\varepsilon$, to należy zwiększyć τ.

Zauważmy, że naszym celem jest przede wszystkim wyznaczenie minimum funkcji f, która jest funkcją wielu zmiennych. Okazuje się, że niekoniecznie do osiągnięcia tego celu potrzebne jest wyznaczanie z dużą dokładnością przybliżenia wartości $\hat{\tau}_{min}^{(i)}$, tak jak to zostało przedstawione powyżej. Taka minimalizacja kierunkowa wymaga zwykle wielu obliczeń wartości funkcji f i jej gradientu. W związku z tym obecnie często stosuje się tzw. niedokładną minimalizację kierunkową, która jest mniej czasochłonna i pozwala na wyznaczenie całego przedziału wartości $\langle \tau_L; \tau_R \rangle$, $\tau_L > 0$, przy czym każda z nich może zostać przyjęta za wystarczająco dobre przybliżenie $\hat{\tau}_{min}^{(i)}$. Istnieją różne reguły stosowane w niedokładnej minimalizacji kierunkowej. Poniżej przedstawimy dwie z nich: regułę Wolfe'a i regułę Goldsteina.

Obie reguły są bardzo podobne. W obu występują parametry $m_1 \in (0; 0,5)$ i $m_2 \in (0,5; 1)$. W regule Wolfe'a decyzję podejmujemy na podstawie sprawdzenia poniższych warunków:

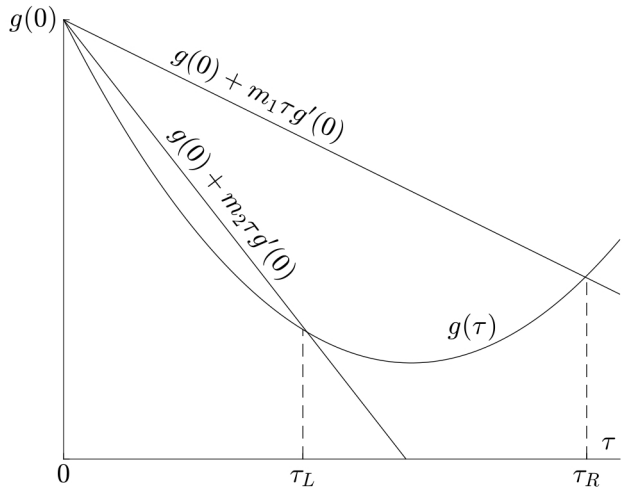
1. Jeżeli $g(\tau) \leqslant g(0) + m_1 \tau g'(0)$ i $g'(\tau) \geqslant m_2 g'(0)$, to przyjmujemy τ jako przybliżenie $\hat{\tau}_{min}^{(i)}$.
2. Jeżeli $g(\tau) > g(0) + m_1 \tau g'(0)$, to należy zmniejszyć τ.
3. Jeżeli $g(\tau) \leqslant g(0) + m_1 \tau g'(0)$ i $g'(\tau) < m_2 g'(0)$, to należy zwiększyć τ.

Reguła Goldsteina różni się od reguły Wolfe'a jedynie zastąpieniem $g'(\tau)$ przez odpowiedni iloraz różnicowy. Taka modyfikacja ma jednak istotne znaczenie wówczas, gdy obliczanie gradientu funkcji f jest czasochłonne (mamy $g'(\tau) = [\nabla f(\mathbf{x}^{(i)} + \tau \mathbf{d}^{(i)})]^T \mathbf{d}^{(i)}$). Odpowiednie warunki można wówczas zapisać w następującej postaci:

1. Jeżeli $g(\tau) \leqslant g(0) + m_1 \tau g'(0)$ i $g(\tau) \geqslant g(0) + m_2 \tau g'(0)$, to przyjmujemy τ jako przybliżenie $\hat{\tau}_{min}^{(i)}$.
2. Jeżeli $g(\tau) > g(0) + m_1 \tau g'(0)$, to należy zmniejszyć τ.
3. Jeżeli $g(\tau) \leqslant g(0) + m_1 \tau g'(0)$ i $g(\tau) < g(0) + m_2 \tau g'(0)$, to należy zwiększyć τ.

Z rysunku 2.2 wynika, że stosując regułę Goldsteina, akceptujemy jako wystarczające przybliżenie $\hat{\tau}_{min}^{(i)}$ wszystkie takie wartości τ, które należą do przedziału $\langle \tau_L; \tau_R \rangle$. Jeżeli $\tau < \tau_L$, stwierdzamy, że krok τ jest zbyt mały, zaś jeżeli $\tau > \tau_R$, stwierdzamy, że jest zbyt duży. Przedstawiony powyżej sposób postępowania nazywamy także ***dwuskośnym testem Goldsteina***. Czasami ograniczamy się do badania tylko nierówności $g(\tau) \leqslant g(0) + m_1 \tau g'(0)$. Wówczas mówimy o ***jednoskośnym teście Goldsteina***. W takim przypadku każdą wartość $\tau \in (0; \tau_R \rangle$ przyjmujemy jako odpowiednie przybliżenie $\hat{\tau}_{min}^{(i)}$.

W przypadku standardowej metody gradientów sprzężonych często zamiast minimalizacji kierunkowej stosuje się inny sposób uzyskania kolejnego przybliżenia $\mathbf{x}^{(k)}$, tak jak to przedstawiono w przykładzie 2.3. Algorytmy wyżej podanych metod



Rysunek 2.2: Interpretacja geometryczna reguły Goldsteina

różnią się poza tym jedynie realizacją punktu 2°, czyli sposobem wyznaczania kierunku poprawy $\mathbf{d}^{(k)}$, co jednak wpływa istotnie na złożoność obliczeniową algorytmu i szybkość zbieżności metody.

W poszczególnych metodach kierunek poprawy $\mathbf{d}^{(k)}$ w $(k+1)$-szej iteracji jest wyznaczany z następujących wzorów:

- w metodzie najszybszego spadku:

$$\mathbf{d}^{(k)} = -\nabla f(\mathbf{x}^{(k)})$$

- w wielowymiarowej metodzie Newtona:

$$\mathbf{d}^{(k)} = -(\mathbf{H}^{-1}f)(\mathbf{x}^{(k)})\nabla f(\mathbf{x}^{(k)}), \qquad \text{gdzie } \mathbf{H}f\text{– hesjan funkcji } f$$

- w metodach gradientów sprzężonych:

w I iteracji	$\mathbf{d}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$
w kolejnych iteracjach	$\mathbf{d}^{(k)} = -\nabla f(\mathbf{x}^{(k)}) + \gamma_{k-1}\mathbf{d}^{(k-1)}$

przy czym

  – w metodzie standardowej:

$$\gamma_{k-1} = \frac{[\nabla f(\mathbf{x}^{(k)})]^T \mathbf{H}f(\mathbf{x}^{(k)})\mathbf{d}^{(k-1)}}{[\mathbf{d}^{(k-1)}]^T \mathbf{H}f(\mathbf{x}^{(k)})\mathbf{d}^{(k-1)}}$$

– w metodzie Fletchera-Reevesa:

$$\gamma_{k-1} = \frac{[\nabla f(\mathbf{x}^{(k)})]^T \nabla f(\mathbf{x}^{(k)})}{[\nabla f(\mathbf{x}^{(k-1)})]^T \nabla f(\mathbf{x}^{(k-1)})}$$

– w metodzie Polaka-Ribiere'a:

$$\gamma_{k-1} = \frac{[\nabla f(\mathbf{x}^{(k)}) - \nabla f(\mathbf{x}^{(k-1)})]^T \nabla f(\mathbf{x}^{(k)})}{[\nabla f(\mathbf{x}^{(k-1)})]^T \nabla f(\mathbf{x}^{(k-1)})}$$

- w metodach zmiennej metryki:

$$\mathbf{d}^{(k)} = -\mathbf{V}^{(k)} \nabla f(\mathbf{x}^{(k)}, \quad \text{gdzie } \mathbf{V}^{(\mathbf{k})} \text{– macierz przybliżająca } (\mathbf{H}^{-1} f)(\mathbf{x}^{(k)})$$

przy czym za $\mathbf{V}^{(0)}$ przyjmujemy dowolną macierz symetryczną i dodatnio określoną, np. $\mathbf{V}^{(0)} = \mathbf{I}$ (macierz jednostkowa) oraz:

– w metodzie DFP:

$$\mathbf{V}^{(k)} = \mathbf{V}^{(k-1)} + \frac{\mathbf{r}^{(k-1)}[\mathbf{r}^{(k-1)}]^T}{[\mathbf{r}^{(k-1)}]^T \mathbf{s}^{(k-1)}} - \frac{\mathbf{V}^{(k-1)} \mathbf{s}^{(k-1)} [\mathbf{s}^{(k-1)}]^T \mathbf{V}^{(k-1)}}{[\mathbf{s}^{(k-1)}]^T \mathbf{V}^{(k-1)} \mathbf{s}^{(k-1)}}$$

– w metodzie BFGS:

$$\mathbf{V}^{(k)} = \mathbf{V}^{(k-1)} + \left(1 + \frac{[\mathbf{s}^{(k-1)}]^T \mathbf{V}^{(k-1)} \mathbf{s}^{(k-1)}}{[\mathbf{s}^{(k-1)}]^T \mathbf{r}^{(k-1)}}\right) \frac{\mathbf{r}^{(k-1)}[\mathbf{r}^{(k-1)}]^T}{[\mathbf{s}^{(k-1)}]^T \mathbf{r}^{(k-1)}} - \\ - \frac{\mathbf{V}^{(k-1)} \mathbf{s}^{(k-1)} [\mathbf{r}^{(k-1)}]^T + \mathbf{r}^{(k-1)} [\mathbf{s}^{(k-1)}]^T \mathbf{V}^{(k-1)}}{[\mathbf{s}^{(k-1)}]^T \mathbf{r}^{(k-1)}}$$

gdzie $\mathbf{r}^{(k-1)} = \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}$, $\mathbf{s}^{(k-1)} = \nabla f(\mathbf{x}^{(k)}) - \nabla f(\mathbf{x}^{(k-1)})$

Z przedstawionych wzorów wynika, że najprostsza w stosowaniu jest metoda najszybszego spadku. Niestety w wielu przypadkach jej zbieżność jest bardzo wolna. Znacznie lepsza okazuje się zwykle wielowymiarowa metoda Newtona, jednak jej główną wadą jest konieczność wyznaczenia hesjanu funkcji celu i jego odwrotności w każdej iteracji. W przypadku, gdy funkcja celu f zależy od dużej ilości zmiennych lub gdy jej wzór jest skomplikowany, wyznaczenie hesjanu i jego odwrotności stanowi istotny problem. Za metody pośrednie, z jednej strony ze względu na szybkość zbieżności, zaś z drugiej ze względu na złożoność obliczeń, można uznać metody gradientów sprzężonych. W standardowym wariancie istnieje wprawdzie konieczność wyznaczania hesjanu, ale nie trzeba go odwracać, natomiast w pozostałych przedstawionych wariantach hesjan nie występuje. Metody zmiennej metryki łączą zalety wielowymiarowej metody Newtona i metod gradientów sprzężonych: ich szybkość zbieżności jest zwykle porównywalna z szybkością w metodzie New-

tona, a zamiast odwrotności hesjanu wyznacza się jej przybliżenia w kolejnych iteracjach. Należy zaznaczyć, że wprowadzenie niedokładnej minimalizacji kierunkowej spowodowało zmniejszenie roli metody DFP na rzecz BFGS – ta ostatnia jest obecnie częściej stosowana.

W poniższych przykładach 2.1–2.4 minimalizacja w kolejnych kierunkach jest wykonywana dokładnie, bez użycia metod opisanych w rozdziale 1. Jest to możliwe ze względu na prostą postać funkcji f.

Przykład 2.1. Wykonać minimalizację funkcji $f(x_1,x_2) = x_1^2 + 5x_2^2 - 4x_1x_2 - 2x_2$ w dwóch kolejnych kierunkach metodą najszybszego spadku. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

Rozwiązanie:
Wyznaczamy pierwszy kierunek minimalizacji ze wzoru:

$$\mathbf{d}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$$

Ponieważ

$$\nabla f(\mathbf{x}) = \nabla f(x_1,x_2) = \begin{bmatrix} 2x_1 - 4x_2 \\ 10x_2 - 4x_1 - 2 \end{bmatrix}$$

więc $\nabla f(\mathbf{x}^{(0)}) = \begin{bmatrix} 6 \\ -16 \end{bmatrix}$, zatem

$$\mathbf{d}^{(0)} = \begin{bmatrix} -6 \\ 16 \end{bmatrix}$$

W pierwszej iteracji mimimalizujemy funkcję:

$$\begin{aligned} f(\mathbf{x}^{(0)} + \tau\mathbf{d}^{(0)}) &= f\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \tau\begin{bmatrix} -6 \\ 16 \end{bmatrix}\right) = f\left(\begin{bmatrix} 1-6\tau \\ -1+16\tau \end{bmatrix}\right) = \\ &= f(1-6\tau, -1+16\tau) = \\ &= (1-6\tau)^2 + 5(-1+16\tau)^2 - 4(1-6\tau)(-1+16\tau) - \\ &- 2(-1+16\tau) = 1700\tau^2 - 292\tau + 12 \end{aligned}$$

Jest to trójmian kwadratowy – osiąga minimum w punkcie $\hat{\tau}_{min}^{(0)} = -\frac{b}{2a} = \frac{292}{2\cdot 1700} \approx 0,086$. Stąd

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \hat{\tau}_{min}^{(0)}\mathbf{d}^{(0)} \approx \begin{bmatrix} 1 \\ -1 \end{bmatrix} + 0,086\begin{bmatrix} -6 \\ 16 \end{bmatrix} = \begin{bmatrix} 0,485 \\ 0,374 \end{bmatrix}$$

Obliczamy gradient w punkcie $\mathbf{x}^{(1)}$:

$$\nabla f(\mathbf{x}^{(1)}) \approx \begin{bmatrix} 2 \cdot 0,485 - 4 \cdot 0,374 \\ 10 \cdot 0,374 - 4 \cdot 0,485 - 2 \end{bmatrix} = \begin{bmatrix} -0,526 \\ -0,200 \end{bmatrix}$$

Błąd po pierwszej iteracji wynosi:

$$||\nabla f(\mathbf{x}^{(1)})|| = \sqrt{(-0,526)^2 + (-0,200)^2} \approx 0,563$$

Wyznaczamy teraz drugi kierunek minimalizacji:

$$\mathbf{d}^{(1)} = -\nabla f(\mathbf{x}^{(1)}) \approx \begin{bmatrix} 0,526 \\ 0,200 \end{bmatrix}$$

Szukamy minimum funkcji:

$$\begin{aligned} f(\mathbf{x}^{(1)} + \tau \mathbf{d}^{(1)}) &= f\left(\begin{bmatrix} 0,485 \\ 0,374 \end{bmatrix} + \tau \begin{bmatrix} 0,536 \\ 0,200 \end{bmatrix} \right) = f\left(\begin{bmatrix} 0,485 + 0,526\tau \\ 0,374 + 0,200\tau \end{bmatrix} \right) = \\ &= f(0,485 + 0,526\tau; 0,374 + 0,200\tau) = \\ &= (0,485 + 0,526\tau)^2 + 5(0,374 + 0,200\tau)^2 - \\ &- 4(0,485 + 0,526\tau)(0,376 - 0,176\tau) - 2(0,374 + 0,200\tau) \approx \\ &\approx 0,056\tau^2 - 0,317\tau - 0,539 \end{aligned}$$

Otrzymaliśmy znowu trójmian kwadratowy – osiąga minimum w punkcie $\hat{\tau}^{(1)}_{min} = -\frac{b}{2a} = \frac{0,317}{2 \cdot 0,056} \approx 2,830$. Stąd:

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \hat{\tau}^{(1)}_{min} \mathbf{d}^{(1)} \approx \begin{bmatrix} 0,485 \\ 0,374 \end{bmatrix} + 2,830 \begin{bmatrix} 0,526 \\ 0,200 \end{bmatrix} \approx \begin{bmatrix} 1,974 \\ 0,940 \end{bmatrix}$$

Obliczamy gradient w punkcie $\mathbf{x}^{(2)}$:

$$\nabla f(\mathbf{x}^{(2)}) \approx \begin{bmatrix} 2 \cdot 1,974 - 4 \cdot 0,940 \\ 10 \cdot 0,940 - 4 \cdot 1,974 - 2 \end{bmatrix} = \begin{bmatrix} 0,188 \\ -0,496 \end{bmatrix}$$

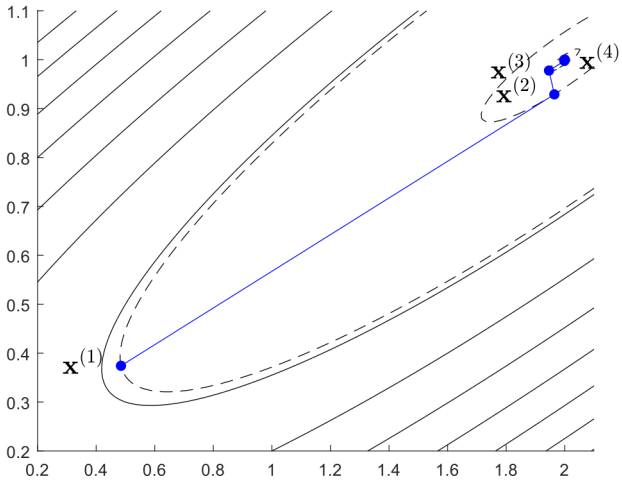
Błąd po drugiej iteracji wynosi:

$$||\nabla f(\mathbf{x}^{(2)})|| = \sqrt{(0,188)^2 + (-0,496)^2} \approx 0,530$$

Tabela 2.1 przedstawia wyniki działania metody najszybszego spadku dla pierwszych ośmiu iteracji.

Tabela 2.1: Wyniki działania metody najszybszego spadku dla funkcji $f(x_1,x_2) = x_1^2 + 5x_2^2 - 4x_1x_2 - 2x_2$

Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	0.484 71	0.374 12	0.562 90
2	1.964 52	0.929 05	0.606 20
3	1.946 24	0.977 80	0.019 97
4	1.998 74	0.997 48	0.021 50
5	1.998 09	0.999 21	0.000 71
6	1.999 96	0.999 91	0.000 76
7	1.999 93	0.999 97	2.513×10^{-5}
8	2.000 00	1.000 00	2.706×10^{-5}



Rysunek 2.3: Poziomice funkcji $f(x_1,x_2) = x_1^2 + 5x_2^2 - 4x_1x_2 - 2x_2$ i rozwiązania przybliżone w początkowych iteracjach metody najszybszego spadku

□

Przykład 2.2. Wykonać minimalizację funkcji $f(x_1,x_2) = x_1^2 + 5x_2^2 - 4x_1x_2 - 2x_2$ w kierunku $\mathbf{d}^{(0)}$ metodą Newtona. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

Rozwiązanie:
Wyznaczamy pierwszy kierunek minimalizacji ze wzoru:

$$\mathbf{d}^{(0)} = -(\mathbf{H}^{-1} f)(\mathbf{x}^{(0)}) \nabla f(\mathbf{x}^{(0)})$$

Funkcja f i punkt $\mathbf{x}^{(0)}$ są takie same jak w przykładzie 2.1, więc gradient funkcji f i jego wartość w punkcie $\mathbf{x}^{(0)}$ też są identyczne.
Hesjan ma postać:

$$\mathbf{H}f(x_1,x_2) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 2 & -4 \\ -4 & 10 \end{bmatrix}$$

Odwracamy hesjan (obliczamy macierz odwrotną do $\mathbf{H}f(x_1,x_2)$):

$$(\mathbf{H}^{-1}f)(x_1,x_2) = \begin{bmatrix} 2{,}5 & 1 \\ 1 & 0{,}5 \end{bmatrix}$$

Stąd:

$$\mathbf{d}^{(0)} = -\begin{bmatrix} 2{,}5 & 1 \\ 1 & 0{,}5 \end{bmatrix} \begin{bmatrix} 6 \\ -16 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

W pierwszej iteracji mimimalizujemy funkcję:

$$\begin{aligned} f(\mathbf{x}^{(0)} + \tau \mathbf{d}^{(0)}) &= f\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \tau \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right) = f\left(\begin{bmatrix} 1+\tau \\ -1+2\tau \end{bmatrix} \right) = f(1+\tau, -1+2\tau) = \\ &= (1+\tau)^2 + 5(-1+2\tau)^2 - 4(1+\tau)(-1+2\tau) - 2(-1+2\tau) = \\ &= 13\tau^2 - 26\tau + 12 \end{aligned}$$

Funkcja kwadratowa, którą otrzymaliśmy, osiąga minimum w punkcie $\hat{\tau}_{min}^{(0)} = -\frac{b}{2a} = \frac{26}{2 \cdot 13} = 1$. Otrzymujemy więc:

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \hat{\tau}_{min}^{(0)} \mathbf{d}^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Gradient w punkcie $\mathbf{x}^{(1)}$ wynosi:

$$\nabla f(\mathbf{x}^{(1)}) = \begin{bmatrix} 2 \cdot 2 - 4 \cdot 1 \\ 10 \cdot 1 - 4 \cdot 2 - 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

zatem osiągnęliśmy rozwiązanie dokładne po pierwszej iteracji. Metoda Newtona przy minimalizacji kwadratowej funkcji wypukłej zawsze prowadzi do rozwiązania dokładnego już po pierwszej iteracji. □

Przykład 2.3. Wykonać minimalizację funkcji $f(x_1,x_2) = x_1^2 + 4x_2^2$ w dwóch kolejnych kierunkach standardową metodą gradientów sprzężonych. Jako punkt startowy

przyjąć $\mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$. Oszacować błąd po każdej iteracji.

Rozwiązanie:
Wyznaczamy pierwszy kierunek minimalizacji ze wzoru:

$$\mathbf{d}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$$

Ponieważ

$$\nabla f(\mathbf{x}) = \nabla f(x_1, x_2) = \begin{bmatrix} 2x_1 \\ 8x_2 \end{bmatrix}$$

więc $\nabla f(\mathbf{x}^{(0)}) = \begin{bmatrix} 2 \\ 16 \end{bmatrix}$, zatem

$$\mathbf{d}^{(0)} = \begin{bmatrix} -2 \\ -16 \end{bmatrix}$$

W pierwszej iteracji wyznaczamy:

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0 \mathbf{d}^{(0)}$$

gdzie

$$\alpha_0 = -\frac{[\mathbf{d}^{(0)}]^T \nabla f(\mathbf{x}^{(0)})}{[\mathbf{d}^{(0)}]^T \mathbf{H} f(\mathbf{x}^{(0)}) \mathbf{d}^{(0)}}$$

Ponieważ

$$\mathbf{H} f(\mathbf{x}) = \begin{bmatrix} 2 & 0 \\ 0 & 8 \end{bmatrix}$$

więc

$$\alpha_0 = -\frac{-260}{2056} \approx 0,126$$

zatem

$$\mathbf{x}^{(1)} \approx \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0,126 \begin{bmatrix} -2 \\ -16 \end{bmatrix} \approx \begin{bmatrix} 0,747 \\ -0,023 \end{bmatrix}$$

Obliczamy gradient w punkcie $\mathbf{x}^{(1)}$:

$$\nabla f(\mathbf{x}^{(1)}) \approx \begin{bmatrix} 2 \cdot 0,747 \\ 8 \cdot (-0,023) \end{bmatrix} \approx \begin{bmatrix} 1,494 \\ -0,187 \end{bmatrix}$$

Błąd po pierwszej iteracji wynosi:

$$||\nabla f(\mathbf{x}^{(1)})|| = \sqrt{1,494^2 + (-0,187)^2} \approx 1,506$$

W drugiej iteracji kierunek $\mathbf{d}^{(1)}$ wyznaczamy ze wzoru:

$$\mathbf{d}^{(1)} = -\nabla f(\mathbf{x}^{(1)}) + \gamma_0 \mathbf{d}^{(0)}$$

gdzie

$$\gamma_0 = \frac{[\nabla f(\mathbf{x}^{(1)})]^T \mathbf{H} f(\mathbf{x}^{(1)}) \mathbf{d}^{(0)}}{[\mathbf{d}^{(0)}]^T \mathbf{H} f(\mathbf{x}^{(1)}) \mathbf{d}^{(0)}} \approx \frac{17,923}{2040} \approx 0,009$$

Stąd

$$\mathbf{d}^{(1)} \approx \begin{bmatrix} -1,494 \\ 0,187 \end{bmatrix} + 0,009 \begin{bmatrix} -2 \\ -16 \end{bmatrix} \approx \begin{bmatrix} -1,512 \\ 0,046 \end{bmatrix}$$

W drugiej iteracji wyznaczamy:

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1 \mathbf{d}^{(1)}$$

gdzie

$$\alpha_1 = -\frac{[\mathbf{d}^{(1)}]^T \nabla f(\mathbf{x}^{(1)})}{[\mathbf{d}^{(1)}]^T \mathbf{H} f(\mathbf{x}^{(1)}) \mathbf{d}^{(1)}}$$

Otrzymujemy:

$$\alpha_1 \approx -\frac{-2,267}{4,588} \approx 0,494$$

zatem

$$\mathbf{x}^{(2)} \approx \begin{bmatrix} 0,747 \\ -0,023 \end{bmatrix} + 0,494 \begin{bmatrix} -1,512 \\ 0,046 \end{bmatrix} \approx \begin{bmatrix} 0,000 \\ 0,000 \end{bmatrix}$$

Oczywiście gradient w punkcie $\mathbf{x}^{(2)}$ wynosi:

$$\nabla f(\mathbf{x}^{(2)}) \approx \begin{bmatrix} 0,000 \\ 0,000 \end{bmatrix}$$

Wyniki obliczeń zostały zapisane z dokładnością do trzech cyfr po przecinku, ale same obliczenia były wykonywane z większą dokładnością. Uzyskany wektor gradientu po drugiej iteracji sugeruje, że otrzymaliśmy wynik dokładny. Tak jest w istocie, gdyż metody gradientów sprzężonych, przy założeniu, że funkcja celu jest funkcją kwadratową i wypukłą, a minimalizacji kierunkowej dokonujemy w sposób dokładny, wymagają wykonania co najwyżej n iteracji, aby otrzymać rozwiązanie dokładne (w przykładzie $n = 2$, ponieważ mamy dwie zmienne niezależne). Ewentualna różnica pomiędzy rozwiązaniami otrzymanym a dokładnym wynika jedynie z błędów zaokrągleń. □

Przykład 2.4. Wykonać minimalizację funkcji $f(x_1, x_2) = x_1^2 + 4x_2^2$ w dwóch kolejnych kierunkach metodą Fletchera-Reevesa. Jako punkt startowy przyjąć

$\mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$. Oszacować błąd po każdej iteracji.

Rozwiązanie:
W pierwszej iteracji mimimalizujemy funkcję:

$$\begin{aligned} f(\mathbf{x}^{(0)} + \tau\mathbf{d}^{(0)}) &= f\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} + \tau \begin{bmatrix} -2 \\ -16 \end{bmatrix}\right) = f\left(\begin{bmatrix} 1-2\tau \\ 2-16\tau \end{bmatrix}\right) = \\ &= f(1-2\tau, 2-16\tau) = (1-2\tau)^2 + 4(2-16\tau)^2 = \\ &= 1028\tau^2 - 260\tau + 17 \end{aligned}$$

Jest to trójmian kwadratowy – osiąga minimum w punkcie $\hat{\tau}^{(0)}_{min} = -\frac{b}{2a} = \frac{260}{2056} \approx 0,126$. Stąd

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \hat{\tau}^{(0)}_{min}\mathbf{d}^{(0)} \approx \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0,126 \begin{bmatrix} -2 \\ -16 \end{bmatrix} = \begin{bmatrix} 0,747 \\ -0,023 \end{bmatrix}$$

Po pierwszej iteracji otrzymaliśmy to samo rozwiązanie przybliżone, co w przykładzie 2.3, więc

$$\nabla f(\mathbf{x}^{(1)}) \approx \begin{bmatrix} 1,494 \\ -0,187 \end{bmatrix}$$

zaś błąd po pierwszej iteracji wynosi:

$$||\nabla f(\mathbf{x}^{(1)})|| \approx 1,506$$

W drugiej iteracji kierunek $\mathbf{d}^{(1)}$ wyznaczamy ze wzoru:

$$\mathbf{d}^{(1)} = -\nabla f(\mathbf{x}^{(1)}) + \gamma_0\mathbf{d}^{(0)}$$

gdzie

$$\gamma_0 = \frac{[\nabla f(\mathbf{x}^{(1)})]^T \nabla f(\mathbf{x}^{(1)})}{[\nabla f(\mathbf{x}^{(0)})]^T \nabla f(\mathbf{x}^{(0)})} \approx \frac{2,267}{260} \approx 0,009$$

Stąd

$$\mathbf{d}^{(1)} \approx \begin{bmatrix} -1,494 \\ 0,187 \end{bmatrix} + 0,009 \begin{bmatrix} -2 \\ -16 \end{bmatrix} \approx \begin{bmatrix} -1,512 \\ 0,046 \end{bmatrix}$$

W drugiej iteracji mimimalizujemy funkcję:

$$\begin{aligned} f(\mathbf{x}^{(1)} + \tau\mathbf{d}^{(1)}) &= f\left(\begin{bmatrix} 0,747 \\ -0,023 \end{bmatrix} + \tau \begin{bmatrix} -1,512 \\ 0,046 \end{bmatrix}\right) = f\left(\begin{bmatrix} 0,747-1,512\tau \\ -0,023+0,046\tau \end{bmatrix}\right) = \\ &= f(0,747-1,512\tau, -0,023+0,046\tau) = \\ &= (0,747-1,512\tau)^2 + 4(-0,023+0,046\tau)^2 \approx \\ &\approx 2,294\tau^2 - 2,267\tau + 0,560 \end{aligned}$$

Powyższy trójmian kwadratowy osiąga minimum w punkcie $\hat{\tau}_{min}^{(1)} = -\frac{b}{2a} \approx \frac{2{,}267}{4{,}588} \approx 0{,}494$. Stąd

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \hat{\tau}_{min}^{(1)}\mathbf{d}^{(1)} \approx \begin{bmatrix} 0{,}747 \\ -0{,}023 \end{bmatrix} + 0{,}494 \begin{bmatrix} -1{,}512 \\ 0{,}046 \end{bmatrix} \approx \begin{bmatrix} 0{,}000 \\ 0{,}000 \end{bmatrix}$$

Otrzymaliśmy rozwiązanie dokładne, identyczne jak w przykładzie 2.3. □

Przykład 2.5. Wykonać minimalizację funkcji $f(x_1, x_2) = x_1^2 + 4x_2^2$ w dwóch kolejnych kierunkach metodą Polaka-Ribiere'a. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$. Oszacować błąd po każdej iteracji.

Rozwiązanie:
W pierwszej iteracji obliczenia są identyczne jak w przykładzie 2.4.

W drugiej iteracji kierunek $\mathbf{d}^{(1)}$ wyznaczamy ze wzoru:

$$\mathbf{d}^{(1)} = -\nabla f(\mathbf{x}^{(1)}) + \gamma_0 \mathbf{d}^{(0)}$$

gdzie

$$\gamma_0 = \frac{[\nabla f(\mathbf{x}^{(1)}) - \nabla f(\mathbf{x}^{(0)})]^T \nabla f(\mathbf{x}^{(1)})}{[\nabla f(\mathbf{x}^{(0)})]^T \nabla f(\mathbf{x}^{(0)})} \approx \frac{2{,}267}{260} \approx 0{,}009$$

Jak widać, dalsze obliczenia są identyczne jak w przykładzie 2.4. □

W przykładach 2.1–2.4 ze względu na postać funkcji f minimum kierunkowe funkcji $g(\tau) = f(\mathbf{x} + \tau\mathbf{d})$ w kierunku $\mathbf{d}$ mogło być wyznaczone dokładnie. Poniższy przykład 2.6 prezentuje działanie metod gradientowych zastosowanych do poszukiwania minimum funkcji $f(x_1, x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$, która w literaturze (np. Branin Jr., 1972) jest znana jako Three-Hump Camel Function. Jest to jedna z funkcji wykorzystywanych do testowania numerycznych metod optymalizacji. Charakterystyczny układ poziomic tej funkcji powoduje problemy związane ze zbieżnością niektórych metod, zwłaszcza przy niekorzystnym wyborze punktu startowego. Funkcja f dana powyższym wzorem ma trzy minima: lokalne w punktach $\hat{\mathbf{x}}_1 \approx (-1{,}74755,\ 0{,}87378)$ i $\hat{\mathbf{x}}_2 = -\hat{\mathbf{x}}_1$, przy czym $f(\hat{\mathbf{x}}_1) = f(\hat{\mathbf{x}}_2) \approx 0{,}29864$, oraz globalne w punkcie $\hat{\mathbf{x}}_3 = (0,\ 0)$, przy czym $f(\hat{\mathbf{x}}_3) = 0$.

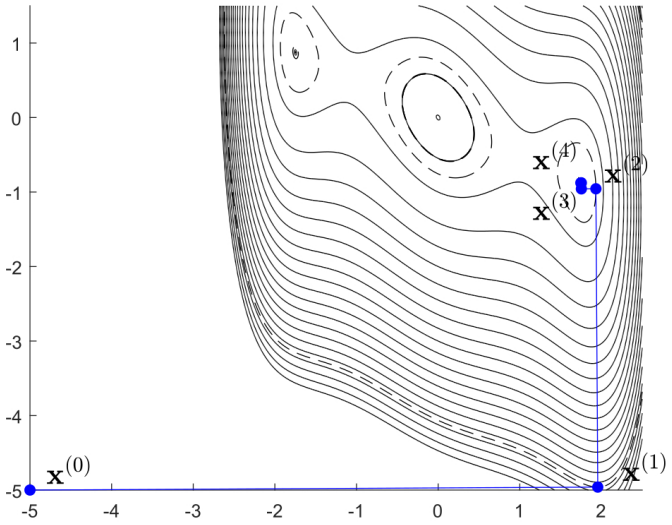
Przykład 2.6. Przetestować działanie metody najszybszego spadku, metod gradientów sprzężonych i wielowymiarowej metody Newtona dla funkcji f określonej powyżej. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = \begin{bmatrix} -5 \\ -5 \end{bmatrix}$.
Rozwiązanie:
Wyniki otrzymane poszczególnymi metodami zostały przedstawione w poniższych

tabelach i na wykresach. W każdym przypadku minimalizację kierunkową przeprowadzono jednowymiarową metodą Newtona z taką dokładnością, że $|g'(\tau)| \leqslant 10^{-5}$, gdzie $g(\tau) = f(\mathbf{x} + \tau\mathbf{d})$, a $\mathbf{d}$ jest kierunkiem minimalizacji.

Na wykresach czarnymi liniami ciągłymi zaznaczono poziomice dla całkowitych wartości funkcji f, a dodatkowo liniami przerywanymi zaznaczono te poziomice, na których następuje zmiana kierunku $\mathbf{d}$ poszukiwań minimum tej funkcji. Wykresy i tabele wskazują na dość szybką zbieżność prawie wszystkich metod z wyjątkiem standardowej metody gradientów sprzężonych, która jest wyraźnie wolniej zbieżna, przy czym dwie spośród testowanych metod zbiegają do jednego, a pozostałe do drugiego minimum lokalnego. Żadna z metod nie jest zbieżna do minimum globalnego. Szybkość zbieżności badanych metod możemy dodatkowo porównać na wykresie 2.9, który przedstawia zależność oszacowania błędu ε od ilości wykonanych iteracji.

Tabela 2.2: Wyniki działania metody najszybszego spadku dla funkcji $f(x_1, x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

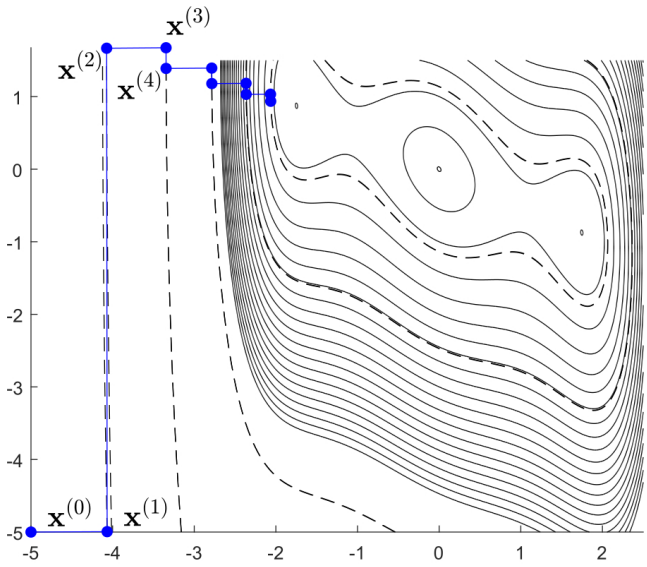
Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	1.955 35	−4.960 25	7.965 30
2	1.932 47	−0.956 48	3.413 60
3	1.754 40	−0.957 50	0.160 60
4	1.753 93	−0.876 75	0.075 87
5	1.747 80	−0.876 79	0.005 77
6	1.747 79	−0.873 89	0.002 73
7	1.747 56	−0.873 89	0.000 21
8	1.747 56	−0.873 78	0.000 10
9	1.747 55	−0.873 78	7.750×10^{-6}
10	1.747 55	−0.873 78	3.622×10^{-6}



Rysunek 2.4: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody najszybszego spadku

Tabela 2.3: Wyniki działania standardowej metody gradientów sprzężonych dla funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

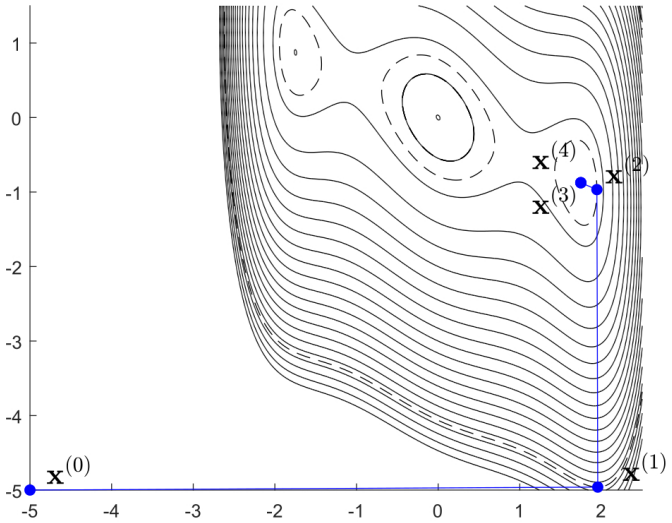
Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	−4.067 14	−4.994 67	851.6855
2	−4.072 93	1.666 36	851.6668
3	−3.345 58	1.672 80	273.571
4	−3.344 99	1.388 05	273.5667
5	−2.785 54	1.392 77	86.676 59
6	−2.784 49	1.179 63	86.674 24
7	−2.365 56	1.182 78	26.756 83
8	−2.363 85	1.031 17	26.756 57
9	−2.065 63	1.032 82	7.818 68
10	−2.063 26	0.935 87	7.820 31



Rysunek 2.5: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach standardowej metody gradientów sprzężonych

Tabela 2.4: Wyniki działania metody Fletchera-Reevesa dla funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

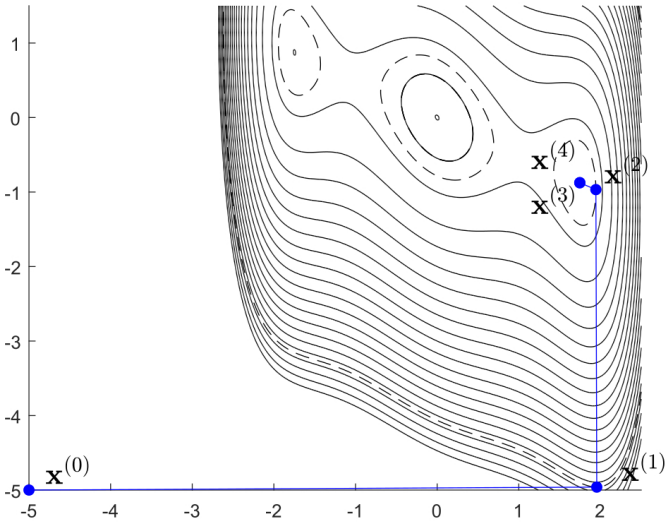
Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	1.955 35	−4.960 25	7.965 30
2	1.944 65	−0.967 32	3.734 74
3	1.747 56	−0.875 55	0.003 91
4	1.747 92	−0.874 78	0.003 87
5	1.747 55	−0.873 78	5.560×10^{-6}
6	1.747 55	−0.873 78	1.226×10^{-6}
7	1.747 55	−0.873 78	1.469×10^{-8}
8	1.747 55	−0.873 78	5.095×10^{-10}
9	1.747 55	−0.873 78	1.511×10^{-10}
10	1.747 55	−0.873 78	7.098×10^{-12}



Rysunek 2.6: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody Fletchera-Reevesa

Tabela 2.5: Wyniki działania metody Polaka-Ribiere'a dla funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

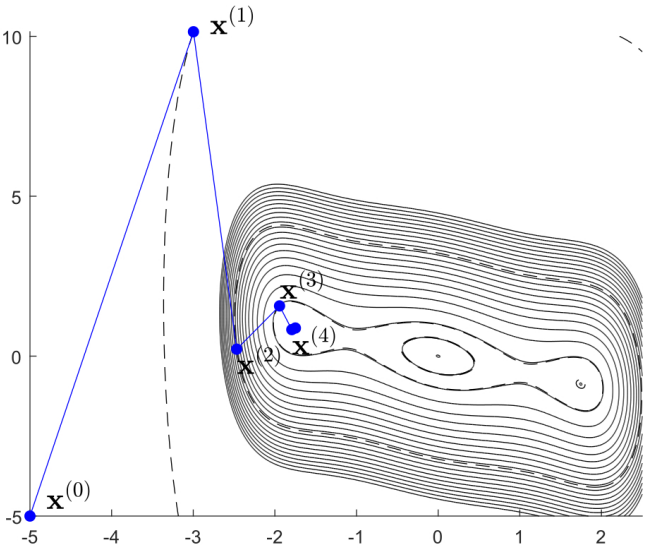
Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	1.955 35	−4.960 26	7.965 23
2	1.944 65	−0.967 32	3.734 74
3	1.747 57	−0.876 15	0.005 22
4	1.747 56	−0.873 78	6.823×10^{-5}
5	1.747 55	−0.873 78	5.044×10^{-9}
6	1.747 55	−0.873 78	5.049×10^{-9}
7	1.747 55	−0.873 78	3.992×10^{-10}
8	1.747 55	−0.873 78	1.926×10^{-10}
9	1.747 55	−0.873 78	1.523×10^{-11}
10	1.747 55	−0.873 78	7.322×10^{-12}



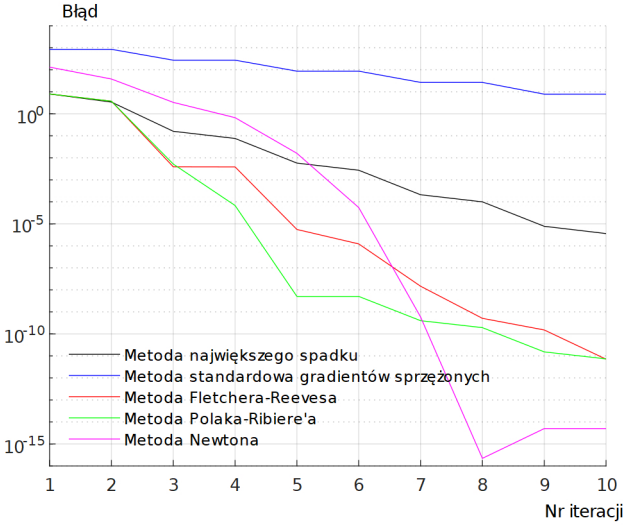
Rysunek 2.7: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody Polaka-Ribiere'a

Tabela 2.6: Wyniki działania wielowymiarowej metody Newtona dla funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

Nr iteracji (k)	$x_1^{(k)}$	$x_2^{(k)}$	$\varepsilon = \|\|\nabla f(\mathbf{x}^{(k)})\|\|$
1	−2.997 49	10.1422	131.8549
2	−2.467 66	0.218 30	38.097 15
3	−1.942 97	1.570 22	3.309 58
4	−1.792 42	0.829 64	0.668 38
5	−1.749 04	0.879 76	0.016 01
6	−1.747 56	0.873 78	5.504×10^{-5}
7	−1.747 55	0.873 78	5.896×10^{-10}
8	−1.747 55	0.873 78	2.220×10^{-16}
9	−1.747 55	0.873 78	4.996×10^{-15}
10	−1.747 55	0.873 78	4.996×10^{-15}



Rysunek 2.8: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach wielowymiarowej metody Newtona



Rysunek 2.9: Porównanie wartości błędów w metodach gradientowych zastosowanych do funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$

Na rysunku 2.9 widzimy, że najwolniejszą dla danych z przykładu jest standardowa metoda gradientów sprzężonych. Metoda Newtona jest szybsza od pozostałych, ale dopiero od ósmej iteracji. □

2.4. Procedury w MATLAB-ie

Jak wspomniano w uwadze 1.6, w MATLAB-ie istnieją dwie podstawowe procedury służące do rozwiązywania wielowymiarowych zadań optymalizacji bez ograniczeń: `fminsearch` i `fminunc`.
Cechy charakterystyczne procedury `fminsearch`:

- Funkcja celu może być nieciągła, jednak wskazane jest, aby punkty nieciągłości nie znajdowały się w pobliżu rozwiązania.
- Obliczenia wykonywane są metodą sympleksu Neldera-Meada opisaną w rozdziale 2.2.

Przykład 2.7. Korzystając z procedury `fminsearch`, wyznaczyć minimum funkcji $f(x_1,x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = [-5,-5]^T$.

Rozwiązanie:
W osobnym pliku **f.mlx** definiujemy funkcję f:

Listing 2.1: Definicja funkcji f

```
function y=f(x)
  y = 2*x(1)^2-1.05*x(1)^4+x(1)^6/6+x(1)*x(2)+x(2)^2;
end
```

Tworzymy także dodatkowy plik **test_minsearch.mlx**:

Listing 2.2: Procedura `test_fminsearch`

```
function [x,f,x_all] = test_fminsearch(x0,n)
  x_all = [];
  opcje = optimset('Display','iter','Maxiter',n,'OutputFcn',@outf);
  [x,f] = fminsearch(@f,x0,opcje);
  function stop = outf(x, optimValues, state)
    stop = false;
    if state=='iter'
      x_all = [x_all;x];
    end
  end
end
```

Wewnątrz zawartej w nim procedury `test_fminsearch` wywołujemy procedurę `fminsearch` poprzedzoną ustawieniem odpowiednich opcji przez `optimset`.

Jest to postępowanie bardzo podobne do tego, które przeprowadziliśmy w przykładzie 1.8 dotyczącym wykorzystania procedury `fminbnd`.

Procedura `fminsearch` wyświetla na ekranie co najwyżej te wyniki pośrednie w kolejnych iteracjach, które widzimy na listingu 2.4. W szczególności nie mamy informacji o współrzędnych punktów $\mathbf{x}^{(k)}$, gdzie k jest numerem iteracji, a podane są jedynie wartości $f(\mathbf{x}^{(k)})$. Aby uzyskać dostęp do brakujących wyników, powinniśmy wykonać dwa dodatkowe kroki. Po pierwsze, w wywołaniu procedury `optimset` należy dopisać jeszcze jedną parę opcji typu nazwa i wartość: `'OutputFcn',@outf`, gdzie `outf` jest nazwą procedury wywoływanej automatycznie przez `fminsearch` w każdej iteracji. Po drugie, należy napisać procedurę `outf`, której zadaniem będzie zapamietanie współrzędnych punktów $\mathbf{x}^{(k)}$. Nagłówek procedury `outf` powinien mieć następującą składnię postaci: `function stop = outf(x, optimValues, state)`. Znaczenie poszczególnych parametrów tej procedury jest następujące: `x` – wektor współrzędnych punktu $\mathbf{x}^{(k)}$ dla ustalonego k, `optimValues` – struktura zawierająca w kolejnych polach informacje wyświetlane w kolumnach na ekranie przez `fminsearch`, `state` – stan algorytmu w chwili wywołania procedury `outf` (`state='iter'` odpowiada wywołaniu bezpośrednio po zakończeniu danej iteracji), `stop` – wartość `'false'` oznacza kontynuację działania `fminsearch` po wyjściu z procedury `outf`. W procedurze `test_fminsearch` przedstawionej na listingu 2.2 wartości $\mathbf{x}^{(k)}$ są zapamiętywane w macierzy `x_all`. Aby macierz ta była widoczna na zewnątrz procedury `outf`, sama procedura powinna być zagnieżdżona wewnątrz innej procedury, tu – `test_fminsearch`.

Wywołanie procedury `test_minsearch` umieszczamy w jeszcze innym pliku:

Listing 2.3: Wywołanie procedury `test_minsearch`

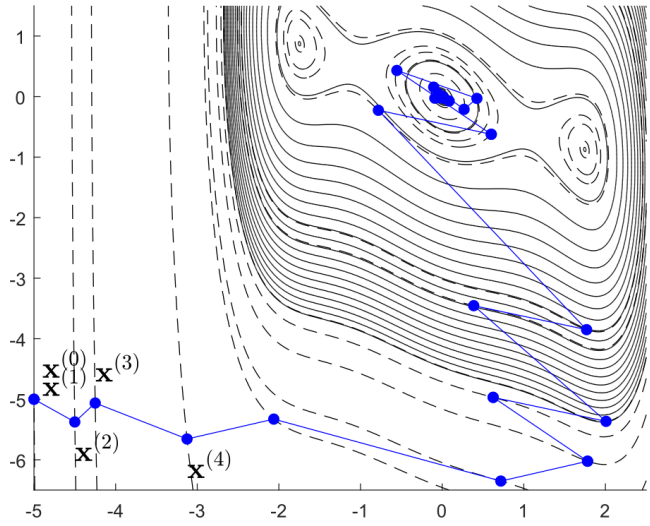
```
x0 = [-5 -5];
[x,f, x_all] = test_fminsearch(x0,10);
```

Ostatni parametr wejściowy oznacza ilość iteracji. W wyniku działania powyższego kodu otrzymujemy:

Listing 2.4: Wynik działania procedury `fminsearch`

```
 Iteration   Func-count     min f(x)         Procedure
     0           1          2047.92
     1           3          2047.92         initial simplex
     2           5          1046.97         expand
     3           7          722.863         expand
     4           9          124.285         expand
     5          11          41.7151         expand
     6          13          36.5531         expand
     7          15          26.6514         reflect
     8          17          26.6514         contract inside
     9          19           22.214         expand
    10          21           22.214         contract inside
```

```
Exiting: Maximum number of iterations has been exceeded
         - increase MaxIter option.
         Current function value: 22.213975
```



Rysunek 2.10: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody Neldera-Meada

□

Wyniki prezentowane są podobnie jak w przypadku procedury `fminbnd`. W ostatniej kolumnie `expand` oznacza operację ekspansji, `reflect` – odbicie, zaś `contract inside` – ściągnięcie do wewnątrz. Analiza wyników przedstawionych na listingu 2.4 oraz ich porównanie z wynikami otrzymanymi w przykładzie 2.6 prowadzą do wniosku, że metoda Neldera-Meada w tym przypadku jest albo wolniej zbieżna od pozostałych, albo rozbieżna. W istocie, po wykonaniu 30 iteracji możemy wnioskować o zbieżności, co więcej, jest to po raz pierwszy zbieżność do minimum globalnego (wcześniej rozpatrywane metody były zbieżne jedynie do minimów lokalnych).

Przyklady 2.8 i 2.9 dotyczą zastosowania procedury `fminunc`.

Cechy charakterystyczne procedury `fminunc`:

- Funkcja celu powinna być różniczkowalna w sposób ciągły.

- W zależności od wyboru użytkownika obliczenia wykonywane są m.in. metodą zmiennej metryki BFGS lub DFP, ewentualnie najszybszego spadku.

Przykład 2.8. Korzystając z procedury `fminunc`, wyznaczyć minimum funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ metodą zmiennej metryki BFGS. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = [-5,-5]^T$.

Rozwiązanie:
W zależności od wybranej opcji gradient funkcji f może być obliczany na podstawie podanego przez użytkownika dokładnego wzoru bądź też pochodne cząstkowe występujące w gradiencie mogą być obliczane w sposób przybliżony za pomocą odpowiednich ilorazów różnicowych. W tym i następnym przykładzie przeprowadzimy obliczenia, posługując się wzorem dokładnym. W tym celu zmodyfikujemy procedurę obliczającą wartość funkcji f. Teraz dodatkowym zadaniem tej procedury będzie wyznaczenie wzoru na gradient tej funkcji (zmienna `grad`). Na listingu 2.5 symbol **`nargout`** jest nazwą standardowej zmiennej zwracającej ilość parametrów wyjściowych, z którymi została wywołana bieżąca procedura z zewnętrznego kodu. Warunek **`nargout`**`>1` w tym przypadku oznacza, że gradient nie jest obliczany, gdy wywołanie procedury `f` w zewnętrznym kodzie ma postać: `y=f(x)` albo `f(x)`.

Listing 2.5: Definicja funkcji f i jej gradientu

```
function [y, grad] =f(x)
  y = 2*x(1)^2-1.05*x(1)^4+x(1)^6/6+x(1)*x(2)+x(2)^2;
  if nargout>1
    grad = [4*x(1)-4.2*x(1)^3+x(1)^5+x(2); x(1)+2*x(2)];
  end
end
```

Podobnie jak poprzednio tworzymy także procedurę `test_fminunc`, z której wywołujemy procedurę `fminunc`. Tym razem opcje wywołania procedury `fminunc` ustawiamy za pomocą procedury `optimoptions` zamiast `optimset`. Nagłówki dwóch ostatnich procedur mają podobną składnię, z tym że dostępne zbiory opcji pokrywają się tylko częściowo. Ponadto, jako pierwszy argument wejściowy wywołania procedury `optimoptions` powinniśmy podać nazwę procedury, której opcje mają dotyczyć, w naszym przypadku będzie to `fminunc`. Na listingu 2.6 warto zwrócić uwagę na opcję `'SpecifyObjectiveGradient'`, której nadano wartość `true`. Oznacza to, że procedura `fminunc` będzie korzystać z dokładnego wzoru na gradient. Gdyby wartość ostatniej opcji była równa `false`, gradient byłby obliczany w sposób przybliżony za pomocą odpowiednich ilorazów różnicowych.

Listing 2.6: Procedura `test_fminunc`

```
function [x,f,x_all] = test_fminunc(x0,n)
  x_all = [];
  opcje = optimoptions('fminunc','Display','iter','MaxIterations',n,
            'OutputFcn',@outf,'SpecifyObjectiveGradient',true);
```

```
  [x,f] = fminunc(@f,x0,opcje);
  function stop = outf(x, optimValues, state)
    stop = false;
    if state=''iter''
      x_all = [x_all;x];
    end
  end
end
```

Po przygotowaniu procedur `f` i `test_fminunc`, przedstawionych na listingach 2.5 i 2.6, możemy wykonać właściwe obliczenia analogicznie jak w przykładzie 2.7.

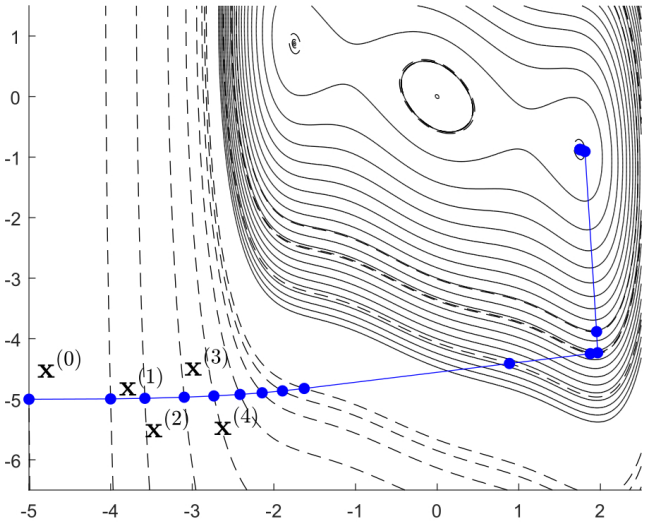
Listing 2.7: Wywołanie procedury `test_minunc`

```
x0 = [-5 -5];
[x,f, x_all] = test_fminunc(x0,20);
```

Listing 2.8: Wynik działania procedury `fminunc` dla metody zmiennej metryki BFGS

```
                                                        First-order
 Iteration  Func-count       f(x)        Step-size       optimality
     0           1          2047.92                      2.62e+03
     1           2          490.787    0.000380952            776
     2           3          246.773              1            415
     3           4          109.877              1            178
     4           5          63.9612              1             83
     5           6           45.105              1           37.6
     6           7          37.6287              1           17.4
     7           8          34.2087              1           11.6
     8           9          32.1269              1           11.3
     9          12          16.5403            5.5           7.93
    10          16          11.3682       0.169675           6.62
.........................................................
    16          26         0.298639              1        0.00355
    17          27         0.298638              1       3.97e-05

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```

Wyniki wyświetlane na ekranie w ostatniej kolumnie (`First-order optimality`) przedstawiają wartości błędu obliczane jako $\|\nabla f(\mathbf{x}^{(k)})\|_\infty$. Po dziesięciu iteracjach wynik jest jeszcze daleki od rozwiązania optymalnego, dopiero po siedemnastej iteracji znajdujemy się w pobliżu minimum, tym razem lokalnego.



Rysunek 2.11: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody zmiennej metryki BFGS

□

Przykład 2.9. Korzystając z procedury `fminunc`, wyznaczyć minimum funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ metodą zmiennej metryki DFP. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = [-5,-5]^T$.

Rozwiązanie:
Jedyną zmianą w kodzie w porównaniu z przykładem 2.8 jest dołączenie w wywołaniu procedury `optimoptions` opcji `'HessUpdate'` i nadanie jej wartości `'dfp'`, co sprawia, że obliczenia zostaną wykonane metodą DFP zamiast domyślnej BFGS. Inną możliwością jest nadanie opcji `'HessUpdate'` wartości `'steepdesc'` – wówczas obliczenia zostaną wykonane metodą najszybszego spadku.

Listing 2.9: Wywołanie procedury `optimoptions` dla metody zmiennej metryki DFP

```
opcje = optimoptions('fminunc','Display','iter','MaxIterations',n,
          'OutputFcn',@outf,'SpecifyObjectiveGradient',true,
          'HessUpdate','dfp');
```

Jak widać na listingu 2.10, wyniki otrzymane metodą DFP w rozpatrywanym zadaniu optymalizacji są bardzo podobne jak w przypadku metody BFGS.

Listing 2.10: Wynik działania procedury `fminunc` dla metody zmiennej metryki DFP

```
                                                        First-order
 Iteration  Func-count       f(x)        Step-size       optimality
     0           1          2047.92                      2.62e+03
     1           2          490.787    0.000380952            776
     2           3          246.773              1            415
     3           4          109.877              1            178
     4           5          63.9613              1             83
     5           6          45.1054              1           37.6
     6           7          37.6297              1           17.4
     7           8          34.2113              1           11.6
     8           9          32.1333              1           11.3
     9          12          16.6001            5.5           7.95
    10          16          11.4106       0.171641           6.63
.............................................................
    19          30         0.298641              1        0.00755
    20          31         0.298639              1        0.00135

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.
<stopping criteria details>
```

□

2.5. Zadania do samodzielnego opracowania

2.5.1. Programowanie w MATLAB-ie

Zadanie 2.1. Zmodyfikować procedurę `m_ekspansji` utworzoną w zadaniu 1.1 tak, aby można było ją wykorzystać do lokalizacji minimum kierunkowego funkcji f dwóch zmiennych w kierunku $\mathbf{d}$. Wzory (1.2) i (1.3) powinny zostać zastąpione odpowiednio przez

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + 2^i \Delta\tau \cdot \mathbf{d} \qquad \text{oraz} \qquad \mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - 2^i \Delta\tau \cdot \mathbf{d} \text{ dla } i = 0, 1, 2, ...$$

a wywołanie zmodyfikowanej procedury powinno mieć postać:
`[t0,t1]=m_ekspansji2(@f,x0,d,Delta_tau)`, gdzie `f` jest nazwą procedury obliczającej wartość funkcji celu, `x0`=$\mathbf{x}^{(0)}$, `d` jest kierunkiem minimalizacji, a `t0` i `t1` stanowią odpowiednio lewy i prawy koniec przedziału lokalizacji minimum w kierunku $\mathbf{d}$.

Zadanie 2.2. Zmodyfikować procedurę `m_Newtona` utworzoną w zadaniu 1.1 tak, aby można było ją wykorzystać do wyznaczenia minimum kierunkowego funkcji f

dwóch zmiennych w kierunku $\mathbf{d}$, czyli minimum funkcji $g(\tau) = f(\mathbf{x}^{(i)} + \tau\mathbf{d})$. W tym celu zastosować dwuskośny test Goldsteina. Wywołanie zmodyfikowanej procedury powinno mieć postać:
`t_min = m_Newtona2(@f,@fp,@fd,x0,d,t0,m)`, gdzie `fp` jest nazwą procedury obliczającej gradient, a `fd` – hesjan funkcji celu, `x0` i `t0` mają znaczenie identyczne jak w metodzie ekspansji, natomiast `m`=$[m_1\ m_2]$, gdzie m_1 i m_2 są parametrami w teście Goldsteina. Parametrem wyjściowym jest taka wartość `t_min`=τ_{min}, którą możemy przyjąć za wystarczająco dobre przybliżenie lokalnego minimum funkcji g.
Wskazówka: korzystając z reguły łańcuchowej, można łatwo sprawdzić, że:

$$g'(\tau) = [\nabla f(\mathbf{x}^{(i)} + \tau\mathbf{d})]^T\mathbf{d} \qquad \text{oraz} \qquad g''(\tau) = \mathbf{d}^T\mathbf{H}f(\mathbf{x}^{(i)} + \tau\mathbf{d})\mathbf{d}$$

Zadanie 2.3. Utworzyć procedury realizujące metody: standardową gradientów sprzężonych, Fletchera-Reevesa, Polaka-Ribiere'a i wielowymiarową Newtona.

Schemat wywołania procedur:

- `[x,blad] = grad_sprz(@f,@fp,@fd,x0,n)`, gdzie `n` – ilość iteracji do wykonania, `x` – macierz $(n+1)\times 2$ zawierająca w każdym wierszu wektor rozwiązania przybliżonego $\mathbf{x}^{(k)}$ po k-tej iteracji, a `blad` – wektor wartości oszacowań błędu po kolejnych iteracjach, natomiast znaczenie pozostałych parametrów pozostaje identyczne jak w zadaniu 2.2;
- `[x,blad] = FR(@f,@fp,@fd,x0,n);`
- `[x,blad] = PR(@f,@fp,@fd,x0,n);`
- `[x,blad] = Newton_Rn(@f,@fp,@fd,x0,n).`

Każda z procedur powinna wyświetlać na ekranie wyniki po każdej iteracji.

2.5.2. Sprawozdanie

W celu sprawdzenia efektywności i niezawodności algorytmów optymalizacyjnych wykorzystuje się często jako funkcje celu takie funkcje testowe, których charakter może powodować trudności obliczeniowe. Są to np. funkcje z wieloma ekstremami lokalnymi lub takie, których zbiór poziomicowy tworzy długie, wąskie doliny albo takie, których wartości na dużym obszarze wokół ekstremum zmieniają się nieznacznie. Wiele przykładów takich funkcji powszechnie stosowanych w optymalizacji można znaleźć np. w Neculai (2008) czy Morè, Garbow, Hilstrom (1981). Oprócz wzoru funkcji ważny jest też sugerowany punkt startowy $\mathbf{x}^{(0)}$ algorytmu. Poniżej podajemy wzory wybranych funkcji testowych i współrzędne punktów startowych.

1. Funkcja Rosenbrocka:

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2-x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-1,2 \;\; 1]^T$$

2. Funkcja White'a i Holsta:

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2-x_1^3)^2, \qquad \mathbf{x}^{(0)} = [-1,2 \;\; 1]^T$$

3. Funkcja Beale'a:

$$\begin{aligned} f(x_1,x_2) = {} & (1,5-x_1(1-x_2))^2 + (2,25-x_1(1-x_2^2))^2 + \\ & + (2,625-x_1(1-x_2^3))^2, \qquad \mathbf{x}^{(0)} = [1 \;\; 0,8]^T \end{aligned}$$

4. Funkcja Himmelblau'a:

$$f(x_1,x_2) = (x_1^2+x_2-11)^2 + (x_1+x_2^2-7)^2, \qquad \mathbf{x}^{(0)} = [1 \;\; 1]^T$$

5. Funkcja Maratos:

$$f(x_1,x_2) = x_1 + 100(x_1^2+x_2^2-1)^2, \qquad \mathbf{x}^{(0)} = [1,1 \;\; 0,1]^T$$

6. Funkcja Freudensteina i Rotha:

$$\begin{aligned} f(x_1,x_2) = {} & (-13+x_1+((5-x_2)x_2-2)x_2)^2 + \\ & + (-29+x_1+((x_2+1)x_2-14)x_2)^2, \qquad \mathbf{x}^{(0)} = [0,5 \;\; -2]^T \end{aligned}$$

7. Funkcja Hieberta:

$$f(x_1,x_2) = (x_1-10)^2 + (x_1x_2-50000)^2, \qquad \mathbf{x}^{(0)} = [0 \;\; 0]^T$$

8. Funkcja trygonometryczna:

$$f(x_1,x_2) = \sum_{i=1}^{2} \left(2 - \sum_{j=1}^{2} \cos x_j + i(1-\cos x_i) - \sin x_i \right)^2, \quad \mathbf{x}^{(0)} = [0,5 \;\; 0,5]^T$$

9. Funkcja Cliffa:

$$f(x_1,x_2) = \left(\frac{x_1-3}{100} \right)^2 - x_1 + x_2 + e^{20(x_1-x_2)}, \qquad \mathbf{x}^{(0)} = [0 \;\; -1]^T$$

10. Funkcja Hagera:

$$f(x_1,x_2)=\sum_{i=1}^{2}(e^{x_i}-\sqrt{i}x_i),\qquad \mathbf{x}^{(0)}=[1\ \ 1]^T$$

11. Pierwsza funkcja Raydana:

$$f(x_1,x_2)=\sum_{i=1}^{2}\frac{i}{10}(e^{x_i}-x_i),\qquad \mathbf{x}^{(0)}=[1\ \ 1]^T$$

12. Druga funkcja Raydana:

$$f(x_1,x_2)=\sum_{i=1}^{2}(e^{x_i}-x_i),\qquad \mathbf{x}^{(0)}=[1\ \ 1]^T$$

13. Funkcja Jennricha i Sampsona:

$$f(x_1,x_2)=\sum_{i=1}^{2}(2+2i-e^{ix_1}-e^{ix_2})^2,\qquad \mathbf{x}^{(0)}=[0{,}3\ \ 0{,}4]^T$$

14. Funkcja Powella:

$$\begin{aligned}f(x_1,x_2,x_3,x_4)&=(x_1+10x_2)^2+5(x_3-x_4)^2+\\&\quad+(x_2-2x_3)^4+10(x_1-x_4)^4,\qquad \mathbf{x}^{(0)}=[3\ \ -1\ \ 0\ \ 1]^T\end{aligned}$$

15. Funkcja Wooda:

$$\begin{aligned}f(x_1,x_2,x_3,x_4)&=100(x_1^2-x_2)^2+(x_1-1)^2+90(x_3^2-x_4)^2+\\&\quad+(1-x_3)^2+10{,}1[(x_2-1)^2+(x_4-1)^2]+\\&\quad+19{,}8(x_2-1)(x_4-1),\qquad \mathbf{x}^{(0)}=[-3\ \ -1\ \ -3\ \ -1]^T\end{aligned}$$

16. Funkcja Browna i Denisa:

$$\begin{aligned}f(x_1,x_2,x_3,x_4)&=\sum_{i=1}^{4}\left[(x_1+0{,}2ix_2-e^{0{,}2i})^2+\right.\\&\quad\left.+(x_3+x_4\sin(0{,}2i)-\cos(0{,}2i))^2\right]^2,\\\mathbf{x}^{(0)}&=[25\ \ 5\ \ -5\ \ -1]^T\end{aligned}$$

Zadanie 2.4. Dana jest funkcja określona jednym z powyższych wzorów 1–16. W przypadku funkcji dwóch zmiennych narysować jej wykres. Wyznaczyć przybliżone wartości punktów, w których funkcja f osiąga minimum, stosując metody:

Neldera-Meada, zmiennej metryki BFGS (zaimplementowane za pomocą standardowych procedur MATLAB-a `fminsearch` oraz `fminunc`) oraz trzy inne metody gradientowe. Obliczenia wykonać dla dwóch różnych punktów startowych, przy czym jednym z nich powinien być punkt podany wraz ze wzorem funkcji. Podać postać gradientu i hesjanu funkcji f. Przyjąć różne dopuszczalne wartości oszacowania błędu: $\varepsilon = 10^{-1},\ 10^{-2},\ 10^{-3},\ 10^{-4},\ 10^{-5}$. Rezultaty obliczeń przedstawić w odpowiednich tabelach i na wykresach. Na podstawie otrzymanych wyników porównać szybkość zbieżności poszczególnych metod i zależność szybkości zbieżności od wyboru punktu startowego.

Postać tabeli:

ε	Metoda ...			
	Ilość iteracji	$x_{1_{min}}$	$x_{2_{min}}$	$f(x_{1_{min}}, x_{2_{min}})$
10^{-1}				
10^{-2}				
10^{-3}				
10^{-4}				
10^{-5}				

W odpowiednim wierszu tabeli należy umieścić wyniki otrzymane po tej iteracji, w której po raz pierwszy oszacowanie błędu było nie większe niż ε.

Wykres powinien przedstawiać zależność funkcji oszacowania błędu ε od ilości iteracji n. W jednym układzie współrzędnych należy umieścić pięć wykresów dla danego punktu startowego, po jednym dla każdej z metod (na osi pionowej należy zastosować skalę logarytmiczną). Wykresy można umieścić w kilku układach współrzędnych, jeżeli przyczyni się to do większej czytelności. Jeżeli to możliwe, należy przyjąć taką maksymalną liczbę iteracji n_{max}, aby $\varepsilon(n_{max}) \leqslant 10^{-10}$.

Rozdział 3

Programowanie nieliniowe: minimalizacja funkcji wielu zmiennych z ograniczeniami

Rozpatrzmy zadanie:

$$\min_{\mathbf{x}\in D} f(\mathbf{x}) \tag{3.1}$$

gdzie $\mathbf{x} = [x_1, \ldots, x_n]^T \in \mathbb{R}^n$, $D \subset \mathbb{R}^n$ jest zbiorem ograniczeń, $f : \mathbb{R}^n \to \mathbb{R}$. W dalszym ciągu będziemy zakładać, że zbiór ograniczeń ma postać:

$$D = \{\mathbf{x} \in \mathbb{R}^n : \ h_k(\mathbf{x}) = 0 \text{ dla } k = 1, \cdots p; \quad g_j(\mathbf{x}) \leqslant 0 \text{ dla } j = 1, \ldots, m\} \tag{3.2}$$

gdzie $h_k : \mathbb{R}^n \to \mathbb{R}$ dla $k = 1, \ldots, p$ oraz $g_j : \mathbb{R}^n \to \mathbb{R}$ dla $j = 1, \ldots, m$.
Jeżeli przynajmniej jedna z funkcji f, h_k, g_j jest nieliniowa, to zadanie (3.1)–(3.2) nazywamy zadaniem programowania nieliniowego z ograniczeniami i oznaczamy (ZPNzo). Równania typu $h_k(\mathbf{x}) = 0$ tworzą ograniczenia równościowe, a nierówności typu $g_j(\mathbf{x}) \leqslant 0$ ograniczenia nierównościowe.

Do numerycznego rozwiązywania zadania (3.1) stosowane są różne metody należące do kilku różnych grup, m.in.:

- metody transformacji zmiennych – mają zastosowanie tylko wtedy, gdy zbiór ograniczeń D jest n-wymiarową kostką, tzn. $D = \{\mathbf{x} \in \mathbb{R}^n : \ a_i \leqslant x_i \leqslant b_i, \ i = 1, \ldots, n\}$, gdzie a_i, b_i – liczby rzeczywiste lub $a_i, b_i = \pm\infty$. Przez odpowiednią zamianę zmiennych sprowadzamy (ZPNzo) do (ZPNbo);
- metody z zastosowaniem modyfikacji kierunków – w pobliżu brzegu zbioru ograniczeń kierunek poszukiwań minimum jest tak modyfikowany, aby był dopuszczalny albo przynajmniej leżał na powierzchni stycznej do ograniczeń;
- metody z zastosowaniem funkcji kary;
- metoda Complex – polega na utworzeniu odpowiedniego sympleksu zawierającego zbiór ograniczeń, a następnie takim jego zmniejszaniu, aby za każdym razem zawierał punkt $\hat{\mathbf{x}}$, w którym funkcja celu f osiąga minimum.

W praktyce często były wykorzystywane metody z zastosowaniem tzw. funkcji kary, reprezentowane przede wszystkim przez metody zewnętrznej, wewnętrznej i przesuwanej funkcji kary. Opiszemy je w kolejnych podrozdziałach. Obecnie ze względu na dużą efektywność na znaczeniu zyskują metody:

- sekwencyjnego programowania kwadratowego,
- zbioru ograniczeń aktywnych,
- obszaru zaufania,
- punktu wewnętrznego.

Każda z wymienionych powyżej grup metod jest reprezentowana w MATLAB-ie, w podrozdziałach 3.3–3.6 umieszczamy także ich opisy. Zainteresowany Czytelnik może znaleźć więcej informacji na ten temat np. w Bonnans et al. (2006), Nocedal Wright (2006), Gill et al. (1981) czy Fletcher (1987).

3.1. Warunki konieczne i dostateczne optymalności

Sformułowanie warunków optymalności w przypadku występowania ograniczeń jest nieco bardziej złożone i wymaga wprowadzenia dodatkowych pojęć.

Definicja 3.1. Ograniczenie nierównościowe $g_j(\mathbf{x}) \leqslant 0$ **jest aktywne w punkcie** $\mathbf{x}_0 \iff g_j(\mathbf{x}_0) = 0$.
Zbiór indeksów wszystkich aktywnych ograniczeń nierównościowych w danym punkcie $\mathbf{x}_0$ *będziemy oznaczać przez* $\mathscr{A}(\mathbf{x}_0)$*, tzn.*

$$\mathscr{A}(\mathbf{x}_0) = \{j : \ g_j(\mathbf{x}_0) = 0\} \tag{3.3}$$

Definicja 3.2. Punkt $\mathbf{x}_0 \in D$ **jest regularny** $\iff$ *wektory* $\nabla h_k(\mathbf{x}_0)$ *dla* $k = 1, \cdots, p$ *oraz* $\nabla g_j(\mathbf{x}_0)$ *dla* $j \in \mathscr{A}(\mathbf{x}_0)$ *stanowią liniowo niezależny układ wektorów.*

Twierdzenie 3.3 (pierwszy warunek konieczny istnienia mimimum; twierdzenie Karusha-Kuhna-Tuckera).
Niech funkcja f *ma minimum lokalne w punkcie* $\hat{\mathbf{x}} \in D$*, przy czym* $\hat{\mathbf{x}}$ *jest punktem regularnym. Niech istnieją ciągłe pochodne cząstkowe pierwszego rzędu funkcji* f*,* h_k *dla* $k = 1, \cdots p$ *oraz* g_j *dla* $j = 1, \ldots, m$*. Wówczas istnieją liczby* $\hat{\lambda}_k \in \mathbb{R}$ *dla* $k = 1, \cdots p$ *oraz* $\hat{\mu}_j \in \mathbb{R}$ *dla* $j = 1, \ldots, m$ *takie, że:*

$1^\circ \ \nabla f(\hat{\mathbf{x}}) + \sum_{k=1}^{p} \hat{\lambda}_k \nabla h_k(\hat{\mathbf{x}}) + \sum_{j=1}^{m} \hat{\mu}_j \nabla g_j(\hat{\mathbf{x}}) = \mathbf{0}$

$2^\circ \ \hat{\mu}_j \geqslant 0$ *dla* $j = 1, \ldots, m$

$3^\circ \ \sum_{j=1}^{m} \hat{\mu}_j g_j(\hat{\mathbf{x}}) = 0$

W teorii optymalizacji istotną rolę odgrywa tzw. ***funkcja Lagrange'a***, którą w przypadku zadania z ograniczeniami równościowymi i nierównościowymi określamy jako:

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(x) + \sum_{k=1}^{p} \lambda_k h_k(\mathbf{x}) + \sum_{j=1}^{m} \mu_j g_j(\mathbf{x}) \tag{3.4}$$

gdzie $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_p]^T$, a $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_m]^T$. Liczby λ_k dla $k = 1, \ldots, p$ oraz μ_j dla $j = 1, \ldots, m$ nazywamy ***mnożnikami Lagrange'a***.

Uwaga 3.1.

1. Warunek 1° tezy Twierdzenia 3.3 może być zapisany krócej z wykorzystaniem funkcji Lagrange'a jako:
$$\nabla_{\mathbf{x}} L(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}) = \mathbf{0} \tag{3.5}$$
przy czym rozumiemy, że $\nabla_{\mathbf{x}} L$ jest tą częścią gradientu funkcji Lagrange'a, w której występują pochodne cząstkowe względem zmiennych $x_1, \ldots, x_n$, natomiast $\hat{\boldsymbol{\lambda}} = [\hat{\lambda}_1, \ldots, \hat{\lambda}_p]^T$, $\hat{\boldsymbol{\mu}} = [\hat{\mu}_1, \ldots, \hat{\mu}_m]^T$.
Można również zauważyć, że ograniczenia równościowe $h_k(\hat{\mathbf{x}}) = 0$ dla $k = 1, \ldots, p$ mogą być zapisane za pomocą funkcji Lagrange'a w postaci
$$\nabla_{\boldsymbol{\lambda}} L(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}) = \mathbf{0} \tag{3.6}$$
a ograniczenia nierównościowe $g_j(\hat{\mathbf{x}}) \leqslant 0$ dla $j = 1, \ldots, m$ w postaci
$$\nabla_{\boldsymbol{\mu}} L(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}) \leqslant \mathbf{0} \tag{3.7}$$
Nierówność wektorów w ostatnim zapisie należy rozumieć w ten sposób, że każda składowa $\nabla_{\boldsymbol{\mu}} L$ jest mniejsza od zera.
2. Punkt 3° tezy oznacza, że $\mu_j g_j(\hat{\mathbf{x}}) = 0$ dla każdego $j = 1, \ldots, m$, gdyż każdy składnik sumy jest niedodatni ze względu na to, że $\mu_j \geqslant 0$ i $g_j(\hat{\mathbf{x}}) \leqslant 0$. W szczególności, jeżeli g_j dla danego j nie jest aktywne w punkcie $\hat{\mathbf{x}}$, to $\mu_j = 0$. Jeżeli g_j jest aktywne w punkcie $\hat{\mathbf{x}}$, to μ_j może być tak dodatnie, jak i równe 0.
3. Twierdzenie 3.3 zostało sformułowane dla ogólnej sytuacji, gdy w (ZPNzo) występują zarówno ograniczenia równościowe, jak i nierównościowe. Jeżeli mamy tylko ograniczenia równościowe, to przyjmujemy $m = 0$ i twierdzenie upraszcza się – nie wystąpią w nim funkcje g_j i związane z nimi liczby μ_j. Podobnie, jeżeli istnieją tylko ograniczenia nierównościowe, przyjmujemy $p = 0$ – wówczas nie wystąpią w twierdzeniu funkcje h_k i związane z nimi liczby λ_k. Analogiczna uwaga dotyczy postaci funkcji Lagrange'a.

Twierdzenie 3.4 (drugi warunek konieczny istnienia mimimum). *Niech funkcja f ma minimum lokalne w punkcie $\hat{\mathbf{x}} \in D$, przy czym $\hat{\mathbf{x}}$ jest punktem regularnym. Niech istnieją ciągłe pochodne cząstkowe drugiego rzędu funkcji f, h_k dla $k = 1, \cdots p$ oraz g_j dla $j = 1, \ldots, m$. Wówczas*

$$\mathbf{d}^T \mathbf{H}_{\mathbf{xx}} L(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}) \mathbf{d} \geqslant 0 \qquad \textit{dla wszystkich wektorów} \qquad \mathbf{d} \in T(\hat{\mathbf{x}})$$

gdzie $\mathbf{H}_{\mathbf{xx}} L$ jest tą częścią hesjanu funkcji Lagrange'a, w której występują wyłącznie pochodne cząstkowe względem zmiennych $x_1, \ldots, x_n$, natomiast

$$T(\hat{\mathbf{x}}) = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{d}^T \nabla h_k(\hat{\mathbf{x}}) = 0 \text{ dla } k = 1, \ldots, p \text{ oraz } \mathbf{d}^T \nabla g_j(\hat{\mathbf{x}}) = 0 \text{ dla } j \in \mathscr{A}(\hat{\mathbf{x}})\}$$

Twierdzenie 3.5 (warunek dostateczny istnienia mimimum). *Jeżeli istnieją ciągłe pochodne cząstkowe drugiego rzędu funkcji f, h_k dla $k = 1, \cdots p$ oraz g_j dla $j = 1, \ldots, m$ w pewnym otoczeniu $O(\hat{\mathbf{x}})$ punktu $\hat{\mathbf{x}} \in D$ oraz istnieją liczby $\hat{\lambda}_k \in \mathbb{R}$ dla $k = 1, \cdots p$ oraz $\hat{\mu}_j \in \mathbb{R}$ dla $j = 1, \ldots, m$ takie, że:*

1° $\nabla f(\hat{\mathbf{x}}) + \sum_{k=1}^{p} \hat{\lambda}_k \nabla h_k(\hat{\mathbf{x}}) + \sum_{j=1}^{m} \hat{\mu}_j \nabla g_j(\hat{\mathbf{x}}) = \mathbf{0}$

2° $\hat{\mu}_j \geqslant 0$ *dla* $j = 1, \ldots, m$

3° $\sum_{j=1}^{m} \hat{\mu}_j g_j(\hat{\mathbf{x}}) = 0$

4° $\mathbf{d}^T \mathbf{H}_{\mathbf{xx}} L(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}) \mathbf{d} > 0$ *dla wszystkich wektorów* $\mathbf{d} \in \tilde{T}(\hat{\mathbf{x}}, \hat{\boldsymbol{\mu}})$, $\mathbf{d} \neq \mathbf{0}$, *gdzie*

$$\tilde{T}(\hat{\mathbf{x}}, \hat{\boldsymbol{\mu}}) = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{d}^T \nabla h_k(\hat{\mathbf{x}}) = 0 \text{ dla } k = 1, \ldots, p \text{ oraz } \mathbf{d}^T \nabla g_j(\hat{\mathbf{x}}) = 0 \text{ dla takich } j \in \mathscr{A}(\hat{\mathbf{x}}), \text{ dla których } \hat{\mu}_j > 0\}$$

to funkcja f ma ścisłe mimimum lokalne w punkcie $\hat{\mathbf{x}}$.

Podobnie, jak w przypadku minimalizacji funkcji jednej zmiennej, rozwiązanie zadania optymalizacji z ograniczeniami funkcji wielu zmiennych można łatwiej uzyskać w przypadku funkcji wypukłych. W dalszym ciągu będziemy wykorzystywać pojęcie zbioru wypukłego.

Definicja 3.6. Zbiór $D \subset \mathbb{R}^n$ jest wypukły $\iff$ *dla dowolnych $\mathbf{x}_1, \mathbf{x}_2 \in D$ i dowolnej liczby $\alpha \in (0;1)$:*

$$\alpha \mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \in D$$

Uwaga 3.2.

1. Powyższa definicja oznacza, że każdy odcinek o końcach w zbiorze wypukłym jest całkowicie zawarty w tym zbiorze.
2. W przypadku jednowymiarowym dowolny zbiór wypukły jest przedziałem liczbowym.

Twierdzenie 1.11, wiążące pojęcie wypukłości funkcji z jej drugą pochodną, przyjmuje obecnie postać:

Twierdzenie 3.7. *Jeżeli istnieją ciągłe pochodne cząstkowe drugiego rzędu funkcji f, przy czym jej hesjan $\mathbf{H}(\mathbf{x})$ jest dodatnio określony dla $\mathbf{x} \in D$, to funkcja f jest ściśle wypukła.*

W niezmienionej postaci zachodzi tutaj także Twierdzenie 1.12.

3.2. Metody z zastosowaniem funkcji kary

3.2.1. Metoda zewnętrznej funkcji kary

Zadanie szukania $\min_{\mathbf{x}\in D} f(\mathbf{x})$ jest równoważne zadaniu szukania $\min_{\mathbf{x}\in\mathbb{R}^n} f_K(\mathbf{x})$, tzn. zadaniu programowania nieliniowego bez ograniczeń, gdzie

$$f_K(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{dla } \mathbf{x} \in D \\ f(\mathbf{x}) + \infty & \text{dla } \mathbf{x} \notin D \end{cases} \tag{3.8}$$

Nieskończoność w ostatnim wzorze jest „karą" za przekroczenie ograniczeń. Możemy zatem zapisać, że

$$f_K(\mathbf{x}) = f(\mathbf{x}) + P(\mathbf{x}) \tag{3.9}$$

gdzie

$$P(\mathbf{x}) = \begin{cases} 0 & \text{dla } \mathbf{x} \in D \\ \infty & \text{dla } \mathbf{x} \notin D \end{cases} \tag{3.10}$$

P nazywamy funkcją kary (krótko: karą), a f_K funkcją z karą.

Realizacja komputerowa takiej minimalizacji jest jednak niemożliwa, zastąpienie nieskończoności przez bardzo dużą liczbę M powoduje bowiem nieciągłość funkcji f_K, co oznacza istotne trudności w rozwiązaniu (ZPNbo). W szczególności nie można wtedy stosować metod gradientowych (najszybszego spadku, Newtona, gradientów sprzężonych, zmiennej metryki). Z tego powodu zamiast powyższej funkcji kary P tworzy się ciąg funkcji kary $(P^{(i)})$ zbieżny do „idealnej" funkcji kary P przedstawionej wzorem (3.10). Ciąg $(P^{(i)})$ powinien być taki, żeby odpowiadający mu ciąg funkcji z karą $(f_K^{(i)})$ był ciągiem funkcji ciągłych i różniczkowalnych dla $\mathbf{x} \in \mathbb{R}$.

Zbiór ograniczeń D określony jest wzorem (3.2). Możemy przyjąć ciąg funkcji kary postaci:

$$P^{(i)}(\mathbf{x}) = \rho^{(i)} \left(\sum_{k=1}^{p} h_k^2(\mathbf{x}) + \sum_{j=1}^{m} [\max\{0, g_j(\mathbf{x})\}]^2 \right) \tag{3.11}$$

gdzie $\rho^{(i)} > 0$ są tzw. współczynnikami funkcji kary. Poszczególne funkcje kary różnią się tylko współczynnikem $\rho^{(i)}$, są równe zeru na zbiorze D i przyjmują wartości większe od zera poza tym zbiorem.

W metodzie zewnętrznej funkcji kary:

1. Startujemy z pewnego punktu $\mathbf{x}^{(0)}$.
2. Przyjmujemy pewną wartość $\rho^{(1)} > 0$.
3. Rozwiązujemy (ZPNbo) postaci

$$\min_{\mathbf{x}\in\mathbb{R}^n} f_K^{(1)}(\mathbf{x}), \quad \text{gdzie} \quad f_K^{(1)}(\mathbf{x}) = f(\mathbf{x}) + P^{(1)}(\mathbf{x}) \tag{3.12}$$

np. jedną z wcześniej poznanych metod, otrzymując jako rozwiązanie $\mathbf{x}_{\min}^{(1)}$ (z tego punktu startujemy w kolejnej iteracji).

4. Przyjmujemy pewną wartość $\rho^{(2)} > \rho^{(1)}$.
5. Rozwiązujemy kolejne (ZPNbo) z funkcją $f_K^{(2)}(\mathbf{x}) = f(\mathbf{x}) + P^{(2)}(\mathbf{x})$ zamiast $f_K^{(1)}$.
6. Kontynuujemy obliczenia dotąd, aż $\|\mathbf{x}_{\min}^{(\ell+1)} - \mathbf{x}_{\min}^{(\ell)}\| \leqslant \varepsilon$, gdzie ε jest dokładnością obliczeń.

Uwaga 3.3.

1. Nie należy przyjmować zbyt dużej wartości pierwszego współczynnika funkcji kary $\rho^{(1)}$, kolejne współczynniki także nie powinny być zbyt szybko zwiększane. W przeciwnym wypadku mogą wystąpić problemy numeryczne z wyznaczeniem rozwiązań pośrednich zadań bez ograniczeń.
2. W poniższych przykładach powyższa uwaga nie ma znaczenia, gdyż wszystkie obliczenia przeprowadzamy tam analitycznie.
3. W metodzie zewnętrznej funkcji kary zbliżamy się do punktu optymalnego zadania z ograniczeniami (3.1) poprzez wartości $\mathbf{x}^{(0)}$, $\mathbf{x}_{\min}^{(1)}$, $\mathbf{x}_{\min}^{(2)}$, ..., które leżą poza zbiorem dopuszczalnym D. Jeżeli ograniczenia muszą być rygorystycznie przestrzegane, należy zastosować inną metodę, np. wewnętrznej funkcji kary.

W poniższych dwóch przykładach ograniczymy się do przypadku, w którym występują wyłącznie ograniczenia nierównościowe.

Przykład 3.1. Wyznaczyć pierwsze przybliżenie $x^{(1)}$ rozwiązania zadania $\min_{x\in D} f(x)$ otrzymane metodą zewnętrznej funkcji kary. Przyjąć następującą funkcję kary:

$$P^{(1)}(x) = \rho^{(1)} \sum_{j=1}^{m} [\max\{0, g_j(x)\}]^q$$

gdzie $q = 2$ lub $q = 3$ oraz $f(x) = x^2 - 4x + 3$, $D = \{x : x \leqslant -1\}$. Zbiór D należy opisać za pomocą jednego ograniczenia. Obliczenia wykonać dla $\rho^{(1)} = 1$, $\rho^{(1)} = 2$ i $\rho^{(1)} = 4$.

Rozwiązanie:
Z postaci zbioru D wynika, że można go opisać za pomocą ograniczenia $g_1(x) = x + 1 \leqslant 0$. Funkcja kary ma zatem postać:

$$P^{(1)}(x) = \rho^{(1)} [\max\{0, g_1(x)\}]^q = \begin{cases} 0 & \text{dla } x \in D \\ \rho^{(1)}[g_1(x)]^q & \text{dla } x \notin D \end{cases}$$

czyli

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla } x \leqslant -1 \\ \rho^{(1)}(x+1)^q & \text{dla } x > -1 \end{cases}$$

W pierwszej iteracji metody zewnętrznej funkcji kary zastępujemy pierwotne zadanie minimalizacji z ograniczeniami $\min\limits_{x\in D} f(x)$ zadaniem minimalizacji bez ograniczeń $\min\limits_{x\in\mathbb{R}} f_K(x)$, gdzie $f_K(x)$ jest funkcją z karą daną wzorem:

$$f_K(x) = f(x) + P^{(1)}(x) = \begin{cases} x^2 - 4x + 3 & \text{dla } x \leqslant -1 \\ x^2 - 4x + 3 + \rho^{(1)}(x+1)^q & \text{dla } x > -1 \end{cases}$$

Funkcja f_K może osiągać minimum tylko dla takiego $x^{(1)}$, dla którego $f'_K(x^{(1)}) = 0$. Obliczamy:

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x \leqslant -1 \\ 2x - 4 + \rho^{(1)} q(x+1)^{q-1} & \text{dla } x > -1 \end{cases}$$

Jeżeli $q = 2$, to

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x \leqslant -1 \\ 2x - 4 + 2\rho^{(1)}(x+1) & \text{dla } x > -1 \end{cases}$$

Dla $x \leqslant -1$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 = 0 \iff x^{(1)} = 2 > -1$, czyli sprzeczność.

Dla $x > -1$ natomiast otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 + 2\rho^{(1)}(x^{(1)} + 1) = 0 \iff 2x^{(1)}(1+\rho^{(1)}) = 4 - 2\rho^{(1)} \iff x^{(1)} = \frac{2-\rho^{(1)}}{1+\rho^{(1)}}$.

Ponieważ funkcja f_K dla $x > -1$ jest kwadratowa z dodatnim współczynnikiem przy x^2, w tak wyznaczonym punkcie $x^{(1)}$ faktycznie otrzymujemy minimum funkcji f_K.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} = 0,5$.
Dla $\rho^{(1)} = 2$ otrzymujemy $x^{(1)} = 0$.
Dla $\rho^{(1)} = 4$ otrzymujemy $x^{(1)} = -0,4$.

Można łatwo przekonać się, wykonując bezpośrednie rachunki, że $\min\limits_{x\in D} f(x)$ jest osiągane dla $\hat{x} = -1$.

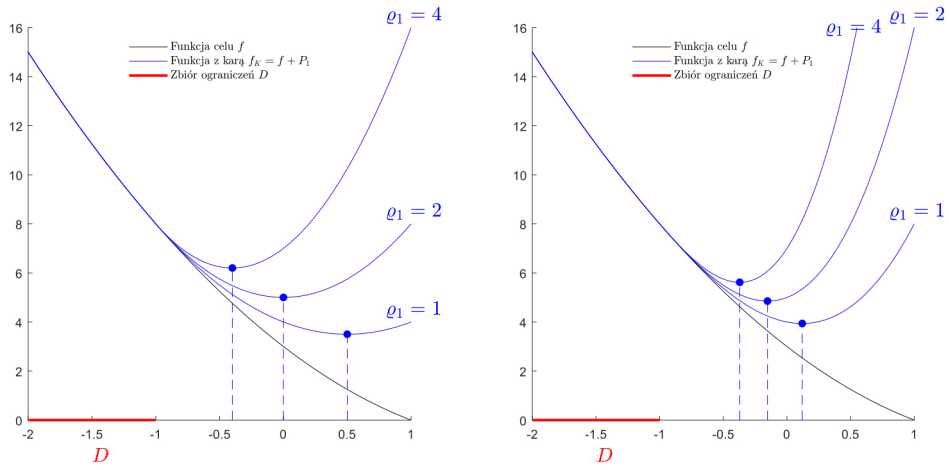
Jeżeli $q = 3$, to

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x \leqslant -1 \\ 2x - 4 + 3\rho^{(1)}(x+1)^2 & \text{dla } x > -1 \end{cases}$$

Dla $x > -1$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 + 3\rho^{(1)}(x^{(1)}+1)^2 = 0 \iff 3\rho^{(1)}(x^{(1)})^2 + (6\rho^{(1)} + 2)x^{(1)} + (3\rho^{(1)} - 4) = 0 \iff x^{(1)} = \frac{-3\rho^{(1)} - 1 - \sqrt{18\rho^{(1)}+1}}{3\rho^{(1)}} \vee x^{(1)} = \frac{-3\rho^{(1)} - 1 + \sqrt{18\rho^{(1)}+1}}{3\rho^{(1)}}$.

Ponieważ funkcja f_K dla $x > -1$ jest wielomianem trzeciego stopnia z dodatnim współczynnikiem przy x^3, jako punkt, w którym jest osiągane minimum, wybieramy $x^{(1)} = \frac{-3\rho^{(1)}-1+\sqrt{18\rho^{(1)}+1}}{3\rho^{(1)}}$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx 0,120$.
Dla $\rho^{(1)} = 2$ otrzymujemy $x^{(1)} \approx -0,153$.
Dla $\rho^{(1)} = 4$ otrzymujemy $x^{(1)} \approx -0,371$.



Rysunek 3.1: Funkcja celu f, funkcja z zewnętrzną karą f_K i zbiór ograniczeń D dla $q = 2$ (po lewej) oraz $q = 3$ (po prawej)

□

Przykład 3.2. Rozwiązać zadanie z przykładu 3.1 dla sytuacji, gdy $D = \{x : x \in \langle 4;5\rangle\}$. Zbiór D opisać za pomocą dwóch ograniczeń.

Rozwiązanie:
Z postaci zbioru D wynika, że można go opisać za pomocą dwóch ograniczeń $g_1(x) = 4 - x \leqslant 0$ i $g_2(x) = x - 5 \leqslant 0$. Funkcja kary ma wtedy postać:

$$P^{(1)}(x) = \rho^{(1)}\left([\max\{0, g_1(x)\}]^q + [\max\{0, g_2(x)\}]^q\right)$$

czyli

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla } x \in \langle 4;5\rangle \\ \rho^{(1)}(4-x)^q & \text{dla } x < 4 \\ \rho^{(1)}(x-5)^q & \text{dla } x > 5 \end{cases}$$

Funkcja z karą f_K jest dana wzorem:

$$f_K(x) = f(x) + P^{(1)}(x) = \begin{cases} x^2 - 4x + 3 & \text{dla } x \in \langle 4;5\rangle \\ x^2 - 4x + 3 + \rho^{(1)}(4-x)^q & \text{dla } x < 4 \\ x^2 - 4x + 3 + \rho^{(1)}(x-5)^q & \text{dla } x > 5 \end{cases}$$

Obliczamy:

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x < 4 \\ 2x - 4 - \rho^{(1)} q(4-x)^{q-1} & \text{dla } x < 4 \\ 2x - 4 + \rho^{(1)} q(x-5)^{q-1} & \text{dla } x > 5 \end{cases}$$

Jeżeli $q = 2$, to

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x \in \langle 4;5\rangle \\ 2x - 4 - 2\rho^{(1)}(4-x) & \text{dla } x < 4 \\ 2x - 4 + 2\rho^{(1)}(x-5) & \text{dla } x > 5 \end{cases}$$

Dla $x \in \langle 4;5\rangle$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 = 0$, czyli sprzeczność.
Dla $x < 4$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 - 2\rho^{(1)}(4 - x^{(1)}) = 0 \iff 2x^{(1)}(1 + \rho^{(1)}) = 4 + 8\rho^{(1)} \iff x^{(1)} = \frac{2+4\rho^{(1)}}{1+\rho^{(1)}}$ – mniejsze od 4 dla dowolnego $\rho^{(1)} > 0$.
Dla $x > 5$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 + 2\rho^{(1)}(x^{(1)} - 5) = 0 \iff 2x^{(1)}(1 + \rho^{(1)}) = 4 + 10\rho^{(1)} \iff x^{(1)} = \frac{2+5\rho^{(1)}}{1+\rho^{(1)}}$ – mniejsze od 5 dla dowolnego $\rho^{(1)} > 0$ (sprzeczne z założeniem, że $x > 5$).

Stąd wynika, że funkcja f_K osiąga minimum dla pewnego $x^{(1)} < 4$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} = 3$.
Dla $\rho^{(1)} = 2$ otrzymujemy $x^{(1)} \approx 3,333$.
Dla $\rho^{(1)} = 4$ otrzymujemy $x^{(1)} = 3,6$.

Można łatwo przekonać się, wykonując bezpośrednie rachunki, że $\min\limits_{x \in D} f(x)$ jest osiągane dla $\hat{x} = 4$.

Jeżeli $q = 3$, to

$$f'_K(x) = \begin{cases} 2x - 4 & \text{dla } x \in \langle 4;5\rangle \\ 2x - 4 - 3\rho^{(1)}(4-x)^2 & \text{dla } x < 4 \\ 2x - 4 + 3\rho^{(1)}(x-5)^2 & \text{dla } x > 5 \end{cases}$$

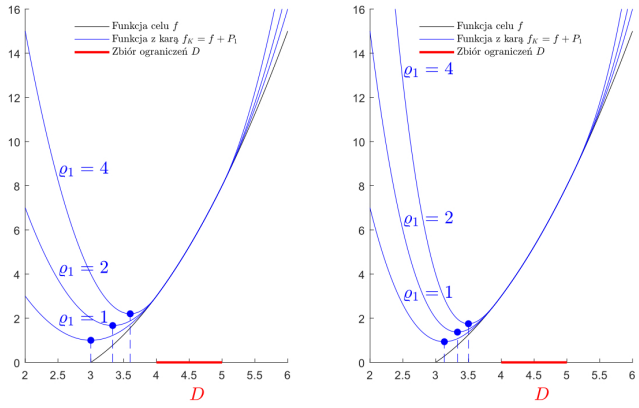
Dla $x < 4$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 - 3\rho^{(1)}(4 - x^{(1)})^2 = 0 \iff$
$-3\rho^{(1)}(x^{(1)})^2 + (24\rho^{(1)} + 2)x^{(1)} - 48\rho^{(1)} - 4 = 0 \iff$
$x^{(1)} = \frac{12\rho^{(1)}+1-\sqrt{12\rho^{(1)}+1}}{3\rho^{(1)}} \vee x^{(1)} = \frac{12\rho^{(1)}+1+\sqrt{12\rho^{(1)}+1}}{3\rho^{(1)}}$.

Ponieważ funkcja f_K dla $x < 4$ jest wielomianem trzeciego stopnia z ujemnym współczynnikiem przy x^3, jako punkt, w którym jest osiągane minimum, przyjmujemy $x^{(1)} = \frac{12\rho^{(1)}+1-\sqrt{12\rho^{(1)}+1}}{3\rho^{(1)}}$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx 3,131$.
Dla $\rho^{(1)} = 2$ otrzymujemy $x^{(1)} \approx 3,333$.
Dla $\rho^{(1)} = 4$ otrzymujemy $x^{(1)} = 3,5$.

Sprawdźmy, czy funkcja f_K nie osiąga minimum także dla $x > 5$. Wtedy:
$f'_K(x^{(1)}) = 0 \iff 2x^{(1)} - 4 + 3\rho^{(1)}(x^{(1)} - 5)^2 = 0 \iff$
$3\rho^{(1)}(x^{(1)})^2 + (2 - 30\rho^{(1)})x^{(1)} + 75\rho^{(1)} - 4 = 0 \iff$
$x^{(1)} = \frac{15\rho^{(1)}-1-\sqrt{1-18\rho^{(1)}}}{3\rho^{(1)}} \vee x^{(1)} = \frac{15\rho^{(1)}-1+\sqrt{1-18\rho^{(1)}}}{3\rho^{(1)}}$.

Dla rozpatrywanych wartości $\rho^{(1)}$ nie istnieją liczby rzeczywiste $x^{(1)}$ wyrażone powyższymi wzorami, zatem punkty $x^{(1)}$ otrzymane dla $x < 4$ są jedynymi, w których funkcja $f_K(x)$ osiąga minimum.



Rysunek 3.2: Funkcja celu f, funkcja z zewnętrzną karą f_K i zbiór ograniczeń D dla $q = 2$ (po lewej) oraz $q = 3$ (po prawej)

□

3.2.2. *Metoda wewnętrznej funkcji kary*

Istnieją dwie podstawowe różnice pomiędzy metodą zewnętrznej i wewnętrznej funkcji kary. Po pierwsze, w zadaniu rozwiązywanym metodą wewnętrznej funkcji kary możemy rozpatrywać wyłącznie ograniczenia nierównościowe. Druga różnica polega na innym określeniu ciągu funkcji kary $(P^{(i)})$. W metodzie wewnętrznej funkcji kary zakładamy, że idealna funkcja kary

$$P(\mathbf{x}) = \begin{cases} 0 \text{ dla } \mathbf{x} \in \mathbb{R} \setminus \partial D \\ \infty \text{ dla } \mathbf{x} \in \partial D \end{cases} \tag{3.13}$$

gdzie ∂D jest brzegiem zbioru D.

Jako ciąg $(P^{(i)})$ zbieżny do idealnej funkcji kary P możemy przyjąć np. funkcje postaci:

$$P^{(i)}(\mathbf{x}) = \begin{cases} 0 & \text{dla } \mathbf{x} \notin D \\ \rho^{(i)} \sum_{j=1}^{m} \left(-\frac{1}{g_j(\mathbf{x})} \right) & \text{dla } \mathbf{x} \in \operatorname{int} D \end{cases} \tag{3.14}$$

albo

$$P^{(i)}(\mathbf{x}) = \begin{cases} 0 & \text{dla } \mathbf{x} \notin D \\ \rho^{(i)} \sum_{j=1}^{m} \left(-\ln(-g_j(x)) \right) & \text{dla } \mathbf{x} \in \operatorname{int} D \end{cases} \tag{3.15}$$

gdzie $\operatorname{int} D$ jest wnętrzem zbioru D.

Uwaga 3.4.

1. W metodzie wewnętrznej funkcji kary zbliżamy się do punktu optymalnego poprzez wartości leżące w zbiorze dopuszczalnym.
2. Jako punkt startowy musimy wybrać punkt ze zbioru dopuszczalnego, tzn. $\mathbf{x}^{(0)} \in D$.
3. Funkcje z karą, tzn. $f_K^{(i)}$, minimalizujemy nie na całej przestrzeni $\mathbb{R}^n$, ale na zbiorze D, jednak ze względu na postać funkcji kary $P^{(i)}$ możemy stosować metody optymalizacji bez ograniczeń.

Przykład 3.3. Wyznaczyć pierwsze przybliżenie $x^{(1)}$ rozwiązania zadania $\min_{x \in D} f(x)$ otrzymane metodą wewnętrznej funkcji kary. Przyjąć funkcję kary

$$P^{(1)}(x) = \rho^{(1)} \sum_{j=1}^{m} \left(-\frac{1}{g_j(x)} \right)$$

oraz $f(x) = 3x + 5$, $D = \{x : x \geqslant 4\}$. Zbiór D opisać za pomocą jednego ograniczenia. Obliczenia wykonać dla $\rho^{(1)} = 1$, $\rho^{(1)} = 0,5$ i $\rho^{(1)} = 0,25$.

Rozwiązanie:
Z postaci zbioru D wynika, że można go opisać za pomocą ograniczenia $g_1(x) = 4 - x \leqslant 0$. Funkcja kary ma postać:

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x \notin D \\ -\frac{\rho^{(1)}}{g_1(x)} & \text{dla} \quad x \in \operatorname{int} D \end{cases}$$

czyli

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x < 4 \\ -\frac{\rho^{(1)}}{4-x} & \text{dla} \quad x > 4 \end{cases}$$

W pierwszej iteracji metody wewnętrznej funkcji kary (identycznie jak w zewnętrznej) zastępujemy pierwotne zadanie minimalizacji z ograniczeniami $\min\limits_{x \in D} f(x)$ zadaniem minimalizacji bez ograniczeń $\min\limits_{x \in \mathbb{R}} f_K(x)$, gdzie f_K jest funkcją z karą daną wzorem:

$$f_K(x) = f(x) + P^{(1)}(x) = \begin{cases} 3x+5 & \text{dla} \quad x < 4 \\ 3x+5-\frac{\rho^{(1)}}{4-x} & \text{dla} \quad x > 4 \end{cases}$$

Ze względu na charakter wewnętrznej metody funkcji kary interesuje nas tylko to minimum, które jest położone wewnątrz zbioru ograniczeń D.

Obliczenia wykonujemy dla $x \in \operatorname{int} D$, czyli $x > 4$:

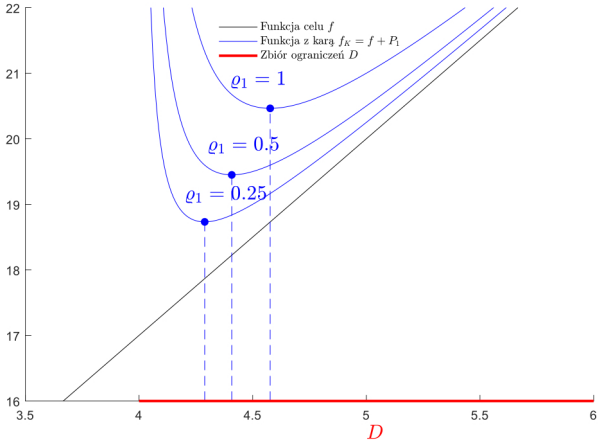
$$f_K'(x) = 3 - \frac{\rho^{(1)}}{(4-x)^2}$$

Dla $x > 4$ otrzymujemy $f_K'(x^{(1)}) = 0 \iff 3 - \frac{\rho^{(1)}}{(4-x^{(1)})^2} = 0 \iff x^{(1)} = 4 - \sqrt{\frac{\rho^{(1)}}{3}} \vee x^{(1)} = 4 + \sqrt{\frac{\rho^{(1)}}{3}}$.

Rozwiązanie $x^{(1)} = 4 - \sqrt{\frac{\rho^{(1)}}{3}} < 4$ odrzucamy. Bierzemy pod uwagę rozwiązanie $x^{(1)} = 4 + \sqrt{\frac{\rho^{(1)}}{3}} > 4$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx 4,577$.
Dla $\rho^{(1)} = 0,5$ otrzymujemy $x^{(1)} \approx 4,408$.
Dla $\rho^{(1)} = 0,25$ otrzymujemy $x^{(1)} \approx 4,289$.

Oczywiście $\min\limits_{x \in D} f(x)$ jest osiągane dla $\hat{x} = 4$.



Rysunek 3.3: Funkcja celu f, funkcja z wewnętrzną karą f_K i zbiór ograniczeń D

□

Przykład 3.4. Wyznaczyć pierwsze przybliżenie $x^{(1)}$ rozwiązania zadania $\min\limits_{x\in D} f(x)$ otrzymane metodą wewnętrznej funkcji kary. Przyjąć następującą funkcję kary:

$$P^{(1)}(x) = \rho^{(1)} \sum_{j=1}^{m} (-\ln(-g_j(x)))$$

oraz $f(x) = 3x+5$, $D = \{x : x \leqslant -1 \ \vee \ x \geqslant 4\}$. Zbiór D opisać za pomocą jednego ograniczenia. Obliczenia wykonać dla $\rho^{(1)} = 1$, $\rho^{(1)} = 0,5$ i $\rho^{(1)} = 0,25$.

Rozwiązanie:
Zbiór D można opisać za pomocą ograniczenia $g_1(x) = -(x+1)(x-4) = -x^2 + 3x + 4 \leqslant 0$.

Funkcja kary przyjmuje postać:

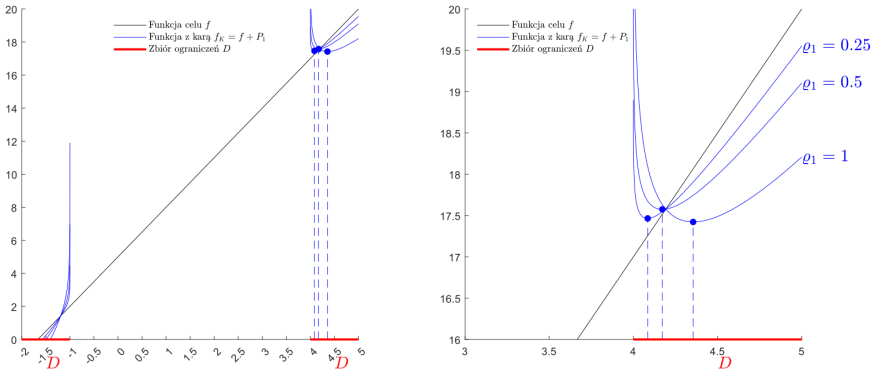
$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x \notin D \\ -\rho^{(1)} \ln(-g_1(x)) & \text{dla} \quad x \in \operatorname{int} D \end{cases}$$

czyli

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x \in (-1;4) \\ -\rho^{(1)} \ln(x^2 - 3x - 4) & \text{dla} \quad x < -1 \ \vee \ x > 4 \end{cases}$$

Funkcja z karą f_K jest określona wzorem:

$$f_K(x) = f(x) + P^{(1)}(x) = \begin{cases} 3x+5 & \text{dla} \quad x \in (-1;4) \\ 3x+5-\rho^{(1)} \ln(x^2 - 3x - 4) & \text{dla} \quad x < -1 \ \vee \ x > 4 \end{cases}$$



Rysunek 3.4: Funkcja celu f, funkcja z wewnętrzną karą f_K i zbiór ograniczeń D (po lewej, powiększenie fragmentu po prawej)

Wykonujemy obliczenia dla $x \in \operatorname{int} D$, czyli $x < -1 \ \vee \ x > 4$:

$$f'_K(x) = 3 - \rho^{(1)} \frac{2x-3}{x^2-3x-4}$$

Dla $x < -1 \ \vee \ x > 4$ otrzymujemy $f'_K(x^{(1)}) = 0 \iff 3 - \rho^{(1)} \frac{2x^{(1)}-3}{(x^{(1)})^2-3x^{(1)}-4} = 0 \iff 3(x^{(1)})^2 - x^{(1)}(9+2\rho^{(1)}) + 3\rho^{(1)} - 12 = 0 \iff x^{(1)} = \frac{2\rho^{(1)}+9-\sqrt{4[\rho^{(1)}]^2+225}}{6} \ \vee \ x^{(1)} = \frac{2\rho^{(1)}+9+\sqrt{4[\rho^{(1)}]^2+225}}{6}$.

Wykonamy najpierw obliczenia dla pierwszego uzyskanego wzoru na $x^{(1)}$, tzn. $x^{(1)} = \frac{2\rho^{(1)}+9-\sqrt{4[\rho^{(1)}]^2+225}}{6}$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx -0,689$.
Dla $\rho^{(1)} = 0,5$ otrzymujemy $x^{(1)} \approx -0,839$.
Dla $\rho^{(1)} = 0,25$ otrzymujemy $x^{(1)} \approx -0,918$.

Wartości te leżą poza zbiorem D, więc je odrzucamy.

Teraz przeprowadzimy obliczenia dla drugiego uzyskanego wzoru na $x^{(1)}$, tzn. $x^{(1)} = \frac{2\rho^{(1)}+9+\sqrt{4[\rho^{(1)}]^2+225}}{6}$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx 4,355$.
Dla $\rho^{(1)} = 0,5$ otrzymujemy $x^{(1)} \approx 4,172$.
Dla $\rho^{(1)} = 0,25$ otrzymujemy $x^{(1)} \approx 4,085$.

Wszystkie otrzymane wartości należą do zbioru D, stanowią zatem rozwiązanie naszego zadania. □

Przykład 3.5. Rozwiązać zadanie z przykładu 3.4 dla sytuacji, w której $f(x) = x^2 - 4x + 3$, $D = \{x : x \leqslant -1\}$. Zbiór D opisać za pomocą jednego ograniczenia.

Rozwiązanie:
Zbiór D można opisać za pomocą ograniczenia $g_1(x) = x + 1 \leqslant 0$.

Funkcja kary przyjmuje postać:

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x \notin D \\ -\rho^{(1)} \ln(-g_1(x)) & \text{dla} \quad x \in \operatorname{int} D \end{cases}$$

czyli

$$P^{(1)}(x) = \begin{cases} 0 & \text{dla} \quad x > -1 \\ -\rho^{(1)} \ln(-x-1) & \text{dla} \quad x < -1 \end{cases}$$

Funkcja z karą f_K jest określona wzorem:

$$f_K(x) = f(x) + P^{(1)}(x) = \begin{cases} x^2 - 4x + 3 & \text{dla} \quad x > -1 \\ x^2 - 4x + 3 - \rho^{(1)} \ln(-x-1) & \text{dla} \quad x < -1 \end{cases}$$

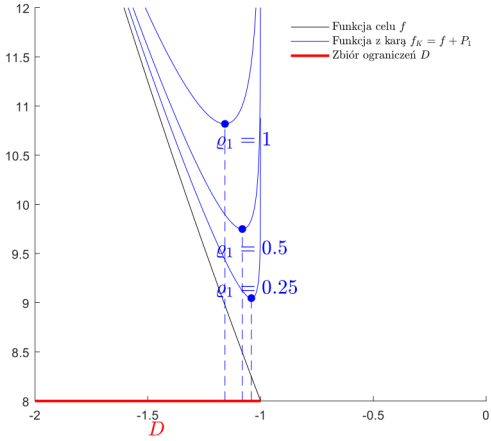
Obliczenia wykonujemy dla $x \in \operatorname{int} D$, czyli $x < -1$:

$$f_K'(x) = 2x - 4 - \rho^{(1)} \frac{1}{x+1}$$

Dla $x < -1$ otrzymujemy $f_K'(x^{(1)}) = 0 \iff 2x^{(1)} - 4 - \rho^{(1)} \frac{1}{x^{(1)}+1} = 0 \iff$
$2(x^{(1)})^2 - 2x^{(1)} - \rho^{(1)} - 4 = 0 \iff x^{(1)} = \frac{1-\sqrt{2\rho^{(1)}+9}}{2} \vee x^{(1)} = \frac{1+\sqrt{2\rho^{(1)}+9}}{2}$.

Drugi wzór na $x^{(1)}$ odrzucamy, gdyż $x^{(1)} = \frac{1+\sqrt{2\rho^{(1)}+9}}{2} > 0 > -1$ dla $\rho^{(1)} > 0$.
Skorzystamy zatem ze wzoru $x^{(1)} = \frac{1-\sqrt{2\rho^{(1)}+9}}{2}$.

Dla $\rho^{(1)} = 1$ otrzymujemy $x^{(1)} \approx -1,158$.
Dla $\rho^{(1)} = 0,5$ otrzymujemy $x^{(1)} \approx -1,081$.
Dla $\rho^{(1)} = 0,25$ otrzymujemy $x^{(1)} \approx -1,041$.



Rysunek 3.5: Funkcja celu f, funkcja z wewnętrzną karą f_K i zbiór ograniczeń D

□

3.2.3. *Metoda przesuwanej funkcji kary*

Przedstawione dotychczas warianty metody funkcji kary charakteryzują się pogarszającym się uwarunkowaniem zadania wraz ze zmianą parametru kary $\rho^{(i)}$ w kolejnych iteracjach. Takiej wady nie ma jednak poniżej opisana metoda, której idea jest podobna do metody zewnętrznej funkcji kary. Istotną różnicą jest przesunięcie kary o zadany wektor. Zamiast zadania z ograniczeniami (3.1)–(3.2) w przesuwanej metodzie funkcji kary rozwiązujemy ciąg zadań bez ograniczeń mający postać:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f_K^{(i)}(\mathbf{x}) \tag{3.16}$$

gdzie

$$f_K^{(i)}(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2}\sum_{k=1}^{p}\sigma_k^{(i)}\left(h_k(\mathbf{x}) + \theta_k^{(i)}\right)^2 + \frac{1}{2}\sum_{j=1}^{m}\rho_j^{(i)}\left[\max\left\{0, g_j(\mathbf{x}) + \eta_j^{(i)}\right\}\right]^2 \tag{3.17}$$

przy czym $\boldsymbol{\sigma}^{(i)} = \left[\sigma_1^{(i)}, \ldots, \sigma_p^{(i)}\right]^T \in \mathbb{R}^p$, $\boldsymbol{\theta}^{(i)} = \left[\theta_1^{(i)}, \ldots, \theta_p^{(i)}\right]^T \in \mathbb{R}^p$,
$\boldsymbol{\rho}^{(i)} = \left[\rho_1^{(i)}, \ldots, \rho_m^{(i)}\right]^T \in \mathbb{R}^m$, a $\boldsymbol{\eta}^{(i)} = \left[\eta_1^{(i)}, \ldots, \eta_m^{(i)}\right]^T \in \mathbb{R}^m$.

W dalszym ciągu, dla uproszczenia zapisu, będziemy pomijać indeks i oznaczający numer iteracji, o ile jego umieszczenie nie będzie bezwzględnie konieczne. Ze względu na przejrzystość dalszych rozważań ograniczymy się najpierw do przy-

padku, w którym występują tylko ograniczenia równościowe. Wówczas

$$f_K(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2}\sum_{k=1}^{p}\sigma_k h_k^2(\mathbf{x}) + \sum_{k=1}^{p}\sigma_k\theta_k h_k(\mathbf{x}) + \frac{1}{2}\sum_{k=1}^{p}\sigma_k\left[\theta_k\right]^2 \tag{3.18}$$

Przyjmując oznaczenia

$$\lambda_k = \sigma_k\theta_k \qquad \text{dla} \qquad k = 1,\ldots,p \tag{3.19}$$

oraz

$$\mathbf{h} = [h_1,\ldots,h_p]^T, \qquad \mathbf{S} = \text{diag}\{\boldsymbol{\sigma}\} \tag{3.20}$$

otrzymujemy:

$$f_K(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T\mathbf{h}(\mathbf{x}) + \frac{1}{2}\mathbf{h}(\mathbf{x})^T\mathbf{S}\mathbf{h}(\mathbf{x}) + \frac{1}{2}\boldsymbol{\theta}^T\mathbf{S}\boldsymbol{\theta} \tag{3.21}$$

Współczynniki λ_k i θ_k można aktualizować w kolejnych iteracjach według wzorów:

$$\lambda_k^{(i+1)} = \lambda_k^{(i)} + \sigma_k^{(i)}h_k(\mathbf{x}^{(i)}) \qquad \text{i} \qquad \theta_k^{(i+1)} = \theta_k^{(i)} + h_k(\mathbf{x}^{(i)}) \tag{3.22}$$

Ostatni składnik we wzorze (3.21) nie zależy od $\mathbf{x}$, więc nie podlega bezpośrednio minimalizacji. Dwa pierwsze składniki stanowią funkcję Lagrange'a $L(\mathbf{x},\boldsymbol{\lambda})$ w przypadku ograniczeń nierównościowych. Funkcję

$$L_A(\mathbf{x},\boldsymbol{\lambda},\mathbf{S}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T\mathbf{h}(\mathbf{x}) + \frac{1}{2}\mathbf{h}(\mathbf{x})^T\mathbf{S}\mathbf{h}(\mathbf{x}) \tag{3.23}$$

nazywamy ***rozszerzoną (uzupełnioną) funkcją Lagrange'a***. Jak widać, w metodzie przesuwanej funkcji kary minimalizujemy $L_A(\mathbf{x},\boldsymbol{\lambda},\mathbf{S})$, dlatego w literaturze określa się ją też jako metodę rozszerzonej funkcji Lagrange'a.

W przypadku, gdy w zadaniu występują tylko ograniczenia nierównościowe, tzn.

$$f_K(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2}\sum_{j=1}^{m}\rho_j\left[\max\left\{0, g_j(\mathbf{x}) + \eta_j\right\}\right]^2 \tag{3.24}$$

przyjmujemy oznaczenie

$$\mu_j = \rho_j\eta_j \qquad \text{dla} \qquad j = 1,\ldots,m \tag{3.25}$$

Wówczas rozszerzoną funkcję Lagrange'a definiujemy jako

$$L_A(\mathbf{x},\boldsymbol{\mu},\mathbf{S}) = f(\mathbf{x}) + \frac{1}{2}\sum_{j=1}^{m}\left(\frac{1}{\rho_j}\left[\max\left\{0,\rho_j g_j(\mathbf{x}) + \mu_j\right\}\right]^2 - \frac{\mu_j^2}{\rho_j}\right) \tag{3.26}$$

Łatwo można zauważyć, że powyższa funkcja Lagrange'a różni się od funkcji f_K, podobnie zresztą jak w przypadku ograniczeń równościowych, jedynie stałym współczynnikiem, zatem znowu minimalizacja funkcji f_K jest równoważna minimalizacji funkcji L_A.

Możemy przyjąć modyfikacje współczynników μ_j i η_j w kolejnych iteracjach np. według wzorów:

$$\mu_j^{(i+1)} = \max\{0, \rho_j^{(i)} g_j(\mathbf{x}^{(i)}) + \mu_j^{(i)}\}, \qquad \eta_j^{(i+1)} = \max\{0, \eta_j^{(i)} + g_j(\mathbf{x}^{(i)})\} \quad (3.27)$$

3.3. Metoda sekwencyjnego programowania kwadratowego

Ze wzoru Taylora wynika, że możemy zapisać przybliżoną równość dla funkcji Lagrange'a określonej wzorem (3.4):

$$L(\mathbf{x}+\mathbf{dx}, \boldsymbol{\lambda}+\mathbf{d\lambda}, \boldsymbol{\mu}+\mathbf{d\mu}) \approx L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) + \sum_{i=1}^{n} \frac{\partial L}{\partial x_i} dx_i + \sum_{k=1}^{p} \frac{\partial L}{\partial \lambda_k} d\lambda_k + \sum_{j=1}^{m} \frac{\partial L}{\partial \mu_j} d\mu_j \quad (3.28)$$

gdzie $\mathbf{dx} = [dx_1, \ldots, dx_n]^T$, $\mathbf{d\lambda} = [d\lambda_1, \ldots, d\lambda_p]^T$, a $\mathbf{d\mu} = [d\mu_1, \ldots, d\mu_m]^T$.
Obliczając składowe gradientu obu stron przybliżonej równości (3.28), otrzymamy kolejno:

$$\begin{aligned} \nabla_{\mathbf{x}} L(\mathbf{x}+\mathbf{dx}, \boldsymbol{\lambda}+\mathbf{d\lambda}, \boldsymbol{\mu}+\mathbf{d\mu}) \approx \nabla f(\mathbf{x}) + \mathbf{H}_{\mathbf{xx}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})\mathbf{dx} + \\ + [\nabla h_1(\mathbf{x}), \ldots, \nabla h_p(\mathbf{x})](\boldsymbol{\lambda}+\mathbf{d\lambda}) + \\ + [\nabla g_1(\mathbf{x}), \ldots, \nabla g_m(\mathbf{x})](\boldsymbol{\mu}+\mathbf{d\mu}) \end{aligned} \quad (3.29)$$

$$\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}+\mathbf{dx}, \boldsymbol{\lambda}+\mathbf{d\lambda}, \boldsymbol{\mu}+\mathbf{d\mu}) \approx [h_1(\mathbf{x}), \ldots, h_p(\mathbf{x})]^T + [\nabla h_1(\mathbf{x}), \ldots, \nabla h_p(\mathbf{x})]^T \mathbf{dx} \quad (3.30)$$

$$\nabla_{\boldsymbol{\mu}} L(\mathbf{x}+\mathbf{dx}, \boldsymbol{\lambda}+\mathbf{d\lambda}, \boldsymbol{\mu}+\mathbf{d\mu}) \approx [g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})]^T + [\nabla g_1(\mathbf{x}), \ldots, \nabla g_m(\mathbf{x})]^T \mathbf{dx} \quad (3.31)$$

Przyjmijmy upraszczające założenie, że równości we wzorach (3.29)–(3.31) są spełnione w sposób dokładny. Przy dodatkowym założeniu, że zadanie (3.1)–(3.2) ma rozwiązanie w punkcie $(\mathbf{x}+\mathbf{dx}, \boldsymbol{\lambda}+\mathbf{d\lambda}, \boldsymbol{\mu}+\mathbf{d\mu})$, z (3.5)–(3.7) oraz (3.29)–(3.31) wynika, że powinien być spełniony poniższy układ równań i nierówności:

$$\begin{cases} \nabla f(\mathbf{x}) + \mathbf{H}_{\mathbf{xx}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \mathbf{dx} + \\ \qquad + [\nabla h_1(\mathbf{x}), \ldots, \nabla h_p(\mathbf{x})](\boldsymbol{\lambda} + \mathbf{d}\boldsymbol{\lambda}) + [\nabla g_1(\mathbf{x}), \ldots, \nabla g_m(\mathbf{x})](\boldsymbol{\mu} + \mathbf{d}\boldsymbol{\mu}) = 0 \\ h_k(\mathbf{x}) + [\nabla h_k(\mathbf{x})]^T \mathbf{dx} = 0 \qquad \text{dla} \qquad k = 1, \ldots, p \\ g_j(\mathbf{x}) + [\nabla g_j(\mathbf{x})]^T \mathbf{dx} \leqslant 0 \qquad \text{dla} \qquad j = 1, \ldots, m \end{cases} \tag{3.32}$$

Układ (3.32) można rozwiązać iteracyjnie, startując z początkowego przybliżenia $(\mathbf{x}^{(0)}, \boldsymbol{\lambda}^{(0)}, \boldsymbol{\mu}^{(0)})$ i uzyskując kolejne przybliżenia rozwiązania układu według wzorów:

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \mathbf{dx}^{(i)}, \quad \boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \mathbf{d}\boldsymbol{\lambda}^{(i)}, \quad \boldsymbol{\mu}^{(i+1)} = \boldsymbol{\mu}^{(i)} + \mathbf{d}\boldsymbol{\mu}^{(i)} \quad \text{dla } i = 0, \ldots \tag{3.33}$$

W metodzie sekwencyjnego programowania kwadratowego w celu wyznaczenia $\mathbf{dx}^{(i)}$ w każdej iteracji rozwiązywane jest następujące zadanie:

$$\min_{\mathbf{d} \in S} \frac{1}{2} \mathbf{d}^T \mathbf{H}_{\mathbf{xx}} L(\mathbf{x}^{(i)}, \boldsymbol{\lambda}^{(i)}, \boldsymbol{\mu}^{(i)}) \mathbf{d} + [\nabla f(\mathbf{x}^{(i)})]^T \mathbf{d} \tag{3.34}$$

gdzie

$$\begin{aligned} S = \{ \mathbf{d} \in \mathbb{R}^n : {} & h_k(\mathbf{x}^{(i)}) + [\nabla h_k(\mathbf{x}^{(i)})]^T \mathbf{d} = 0 \qquad \text{dla} \qquad k = 1, \ldots, p, \\ & g_j(\mathbf{x}^{(i)}) + [\nabla g_j(\mathbf{x}^{(i)})]^T \mathbf{d} \leqslant 0 \qquad \text{dla} \qquad j = 1, \ldots, m \} \end{aligned} \tag{3.35}$$

Rozwiązanie optymalne $\hat{\mathbf{d}}$ zadania (3.34)–(3.35) przyjmujemy jako kierunek poszukiwań wektora $\mathbf{x}^{(i+1)}$ danego wzorem (3.33), tzn. bierzemy $\mathbf{dx}^{(i)} = \alpha^{(i)} \hat{\mathbf{d}}$, przy czym współczynnik $\alpha^{(i)}$ wyznaczamy, wykorzystując pomocniczą funkcję (ang. *merit function*) postaci:

$$\Psi^{(i)}(\mathbf{x}) = f(\mathbf{x}) + \sum_{k=1}^{p} \sigma_k^{(i)} |h_k(\mathbf{x})| + \sum_{j=1}^{m} \rho_j^{(i)} \max\{0, g_j(\mathbf{x})\} \tag{3.36}$$

Obliczenia wykonujemy w następujący sposób:

1. Na początku przyjmujemy $\alpha^{(i)} = 1, \tau_\alpha \in (0; \tau\rangle$, gdzie $\tau \in (0; 1)$.
2. Sprawdzamy, czy nastąpił wystarczający spadek wartości funkcji $\Psi^{(i)}$ przy przejściu od $\mathbf{x}^{(i)}$ do $\mathbf{x}^{(i)} + \alpha^{(i)} \hat{\mathbf{d}}$ (możemy zastosować test jednoskośny Goldsteina).
3. Jeżeli nie, zmniejszamy wartość $\alpha^{(i)}$ według wzoru $\alpha^{(i)} = \tau_\alpha \alpha^{(i)}$ i wracamy do punktu 2. Jeżeli tak, przyjmujemy $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \alpha^{(i)} \hat{\mathbf{d}}$.

W istocie funkcja $\Psi^{(i)}$ dana wzorem (3.36) jest funkcją kary. Jej współczynniki $\sigma_k^{(i)}$ oraz $\rho_j^{(i)}$ w MATLAB-ie wyznaczane są ze wzorów zaproponowanych w Powell (1978):

$$\sigma_k^{(i)} = \begin{cases} \dfrac{\|\nabla f(\mathbf{x})\|}{\|\nabla h_k(\mathbf{x})\|} & \text{dla} \quad i = 0 \\ \max\left\{\lambda_k^{(i)}, \dfrac{\sigma_k^{(i)} + \lambda_k^{(i)}}{2}\right\} & \text{dla} \quad i > 0 \end{cases} \tag{3.37}$$

oraz

$$\rho_j^{(i)} = \begin{cases} \dfrac{\|\nabla f(\mathbf{x})\|}{\|\nabla g_j(\mathbf{x})\|} & \text{dla} \quad i = 0 \\ \max\left\{\mu_k^{(i)}, \dfrac{\rho_k^{(i)} + \mu_k^{(i)}}{2}\right\} & \text{dla} \quad i > 0 \end{cases} \tag{3.38}$$

Przyjmijmy następujące oznaczenia:

$$\begin{aligned} &\mathbf{F} = \mathbf{H}_{\mathbf{xx}} L(\mathbf{x}^{(i)}, \boldsymbol{\lambda}^{(i)}, \boldsymbol{\mu}^{(i)}), && \mathbf{c} = \nabla f(\mathbf{x}^{(i)}) \\ &\mathbf{h} = [h_1(\mathbf{x}^{(i)}), \ldots, h_p(\mathbf{x}^{(i)})]^T, && \mathbf{g} = [g_1(\mathbf{x}^{(i)}), \ldots, g_m(\mathbf{x}^{(i)})]^T \\ &\mathbf{A}_{\mathbf{h}} = [\nabla h_1(\mathbf{x}^{(i)}), \ldots, \nabla h_p(\mathbf{x}^{(i)})], && \mathbf{A}_{\mathbf{g}} = [\nabla g_1(\mathbf{x}^{(i)}), \ldots, \nabla g_m(\mathbf{x}^{(i)})] \end{aligned} \tag{3.39}$$

Wówczas zadanie (3.34)–(3.35) można zapisać w postaci:

$$\min_{\mathbf{d} \in S} \frac{1}{2} \mathbf{d}^T \mathbf{F} \mathbf{d} + \mathbf{c}^T \mathbf{d} \tag{3.40}$$

gdzie

$$\begin{aligned} S = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{A}_{\mathbf{h}}^T \mathbf{d} &= -\mathbf{h} \\ \mathbf{A}_{\mathbf{g}}^T \mathbf{d} &\leqslant -\mathbf{g}\} \end{aligned} \tag{3.41}$$

Mamy zatem do czynienia z minimalizacją kwadratowej funkcji celu przy liniowych ograniczeniach. Tego typu zadania stanowią wyodrębnioną klasę – są to tzw. zadania programowania kwadratowego. Istotna trudność w rozwiązaniu zadania (3.34)–(3.35), a zatem równoważnie (3.40)–(3.41), polega na uwzględnieniu ograniczeń nierównościowych. W MATLAB-ie do rozwiązania zadania (3.40)–(3.41) wykorzystywana jest metoda zbioru ograniczeń aktywnych opisana w podrozdziale 3.4. Poszukiwanie minimum w zadaniu programowania kwadratowego jest wówczas dokonywane nie na całym zbiorze S, ale na jego podzbiorze, w którym oprócz ograniczeń równościowych bierzemy pod uwagę tylko te ograniczenia nierównościowe, które są aktywne w danym punkcie $\mathbf{x}^{(i)}$.

Kolejną kwestią, na którą należy zwrócić uwagę, jest sposób wyznaczenia hesjanu $\mathbf{H}_{\mathbf{xx}} L(\mathbf{x}^{(i)}, \boldsymbol{\lambda}^{(i)}, \boldsymbol{\mu}^{(i)})$. Zwykle oblicza się go w sposób przybliżony, z tych samych powodów, z których robi się to w przypadku minimalizacji bez ograniczeń. Możemy zastosować tu np. wzór wynikający z uogólnienia odpowiedniego wzoru w metodzie zmiennej metryki BFGS. Jeżeli przez $\mathbf{W}^{(i)}$ oznaczymy przybliżenie hesjanu, to

$$\mathbf{W}^{(i+1)} = \mathbf{W}^{(i)} + \frac{\mathbf{s}^{(i)}[\mathbf{s}^{(i)}]^T}{[\mathbf{s}^{(i)}]^T\mathbf{r}^{(i)}} - \frac{\mathbf{W}^{(i)}\mathbf{r}^{(i)}[\mathbf{r}^{(i)}]^T\mathbf{W}^{(i)}}{[\mathbf{r}^{(i)}]^T\mathbf{W}^{(i)}\mathbf{r}^{(i)}} \tag{3.42}$$

gdzie $\mathbf{r}^{(i)} = \mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}$

$$\mathbf{s}^{(i)} = \left(\nabla f(\mathbf{x}^{(i+1)}) + \sum_{k=1}^{p} \lambda_k^{(i)} \nabla h_k(\mathbf{x}^{(i+1)}) + \sum_{j=1}^{m} \mu_j^{(i)} \nabla g_j(\mathbf{x}^{(i+1)})\right) -$$
$$- \left(\nabla f(\mathbf{x}^{(i)}) + \sum_{k=1}^{p} \lambda_k^{(i)} \nabla h_k(\mathbf{x}^{(i)}) + \sum_{j=1}^{m} \mu_j^{(i)} \nabla g_j(\mathbf{x}^{(i)})\right)$$

3.4. Metoda zbioru ograniczeń aktywnych

Jeżeli zadanie programowania nieliniowego jest zadaniem programowania kwadratowego, w którym występuje przynajmniej jedno ograniczenie nierównościowe, to możemy do jego rozwiązania zastosować opisaną poniżej metodę zbioru ograniczeń aktywnych. Metoda ta może zostać wykorzystana w szczególności wewnętrznie w każdej głównej iteracji metody sekwencyjnego programowania kwadratowego do rozwiązania pomocniczego zadania (3.40)–(3.41).

Metoda zbioru ograniczeń aktywnych jest także metodą iteracyjną. Jej istotą jest zmiana w każdej iteracji pierwotnego zbioru ograniczeń S w taki sposób, aby uwzględnione zostały tylko te spośród ograniczeń nierównościowych, które są aktualnie aktywne. Niech $\mathbf{d}^{(0)}$ będzie punktem dopuszczalnym zadania (3.40)–(3.41). Oznaczmy przez $\mathscr{A}(\mathbf{d}^{(0)})$ zbiór indeksów aktywnych ograniczeń nierównościowych w punkcie $\mathbf{d}^{(0)}$, tzn. $(\mathbf{A}_\mathbf{g}^T)_\ell \mathbf{d}^{(0)} = -(\mathbf{g})_\ell$ dla $\ell \in \mathscr{A}(\mathbf{d}^{(0)})$. W pierwszej iteracji rozwiązujemy zatem zadanie:

$$\min_{\mathbf{d} \in S^{(0)}} \frac{1}{2}\mathbf{d}^T\mathbf{F}\mathbf{d} + \mathbf{c}^T\mathbf{d} \tag{3.43}$$

gdzie

$$\begin{aligned} S^{(0)} = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{A}_\mathbf{h}^T\mathbf{d} = -\mathbf{h} \\ (\mathbf{A}_\mathbf{g}^T)_\ell \mathbf{d} = -(\mathbf{g})_\ell \quad \text{dla} \quad \ell \in \mathscr{A}(\mathbf{d}^{(0)})\} \end{aligned} \tag{3.44}$$

W dalszym ciągu będziemy zakładać, że macierz $\mathbf{F}$ jest nieujemnie określona, a $(\mathbf{A}_\mathbf{h}^T)_k$ dla $k = 1, \ldots p$ razem z $(\mathbf{A}_\mathbf{g}^T)_\ell$ dla $\ell \in \mathscr{A}(\mathbf{d}^{(0)})$ stanowią zbiór wektorów liniowo niezależnych. Zbiór $\mathscr{A}(\mathbf{d}^{(0)})$ nie musi zawierać wszystkich indeksów ograniczeń aktywnych, jednak ważne jest, aby spełniony był wyżej podany warunek liniowej niezależności. Zauważmy, że wszystkie ograniczenia w powyższym zadaniu są równościowe. Oznaczmy przez $\hat{\mathbf{d}}^{(0)}$ rozwiązanie optymalne zadania (3.43)–(3.44).

Jeżeli $\hat{\mathbf{d}}^{(0)} \neq \mathbf{d}^{(0)}$, to sprawdzamy, czy $\hat{\mathbf{d}}^{(0)}$ spełnia wszystkie ograniczenia ze zbioru S, czyli także pominięte w pierwszej iteracji pozostałe ograniczenia nierównościowe. W przypadku, gdy faktycznie $\hat{\mathbf{d}}^{(0)} \in S$, przechodzimy do następnej iteracji bez zmiany aktualnego zbioru ograniczeń aktywnych, tzn. $S^{(1)} = S^{(0)}$ i przyjmujemy jako punkt startowy w tej iteracji $\mathbf{d}^{(1)} = \hat{\mathbf{d}}^{(0)}$. W przeciwnym przypadku, tzn. gdy $\hat{\mathbf{d}}^{(0)} \notin S$, w kolejnej iteracji startujemy z $\mathbf{d}^{(1)} = \mathbf{d}^{(0)} + \alpha^{(0)}(\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)})$, przy czym $\alpha^{(0)} \in \langle 0;1 \rangle$. Zauważmy, że dla dowolnego $\alpha^{(0)}$ tak określony wektor $\mathbf{d}^{(1)} \in S^{(0)}$. Wynika to z faktu, że $\mathbf{A}_{\mathbf{h}}^T \hat{\mathbf{d}}^{(0)} = -\mathbf{h}$ oraz $(\mathbf{A}_{\mathbf{g}}^T)_\ell \hat{\mathbf{d}}^{(0)} = -(\mathbf{g})_\ell$ dla $\ell \in \mathscr{A}(\mathbf{d}^{(0)})$, zatem

$$\begin{aligned} &\alpha^{(0)} \mathbf{A}_{\mathbf{h}}^T (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) = \mathbf{0} && \text{oraz} \\ &\alpha^{(0)} (\mathbf{A}_{\mathbf{g}}^T)_\ell (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) = \mathbf{0} && \text{dla} \quad \ell \in \mathscr{A}(\mathbf{d}^{(0)}) \end{aligned}$$

więc

$$\begin{aligned} &\mathbf{A}_{\mathbf{h}}^T (\mathbf{d}^{(0)} + \alpha^{(0)} (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)})) = -\mathbf{h} && \text{oraz} \\ &(\mathbf{A}_{\mathbf{g}}^T)_\ell (\mathbf{d}^{(0)} + \alpha^{(0)} (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)})) = -(\mathbf{g})_\ell && \text{dla} \quad \ell \in \mathscr{A}(\mathbf{d}^{(0)}) \end{aligned}$$

Parametr $\alpha^{(0)}$ wyznaczamy tak, aby w punkcie $\mathbf{d}^{(1)}$ kolejne ograniczenie nierównościowe stało się aktywne, tzn. aby dodatkowo $(\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(1)} = -(\mathbf{g})_q$ dla pewnego $q \notin \mathscr{A}(\mathbf{d}^{(0)})$. Jeżeli $\hat{\mathbf{d}}^{(0)} \notin S$, to istnieje takie $q \notin \mathscr{A}(\mathbf{d}^{(0)})$, że $(\mathbf{A}_{\mathbf{g}}^T)_q \hat{\mathbf{d}}^{(0)} > -(\mathbf{g})_q$. Jednakże $(\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(0)} \leqslant -(\mathbf{g})_q$, więc $(\mathbf{A}_{\mathbf{g}}^T)_q (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) > 0$. Ponadto

$$\begin{aligned} (\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(1)} &= (\mathbf{A}_{\mathbf{g}}^T)_q (\mathbf{d}^{(0)} + \alpha^{(0)} (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)})) = \\ &= (\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(0)} + \alpha^{(0)} (\mathbf{A}_{\mathbf{g}}^T)_q (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) \end{aligned} \tag{3.45}$$

Aby ograniczenie o numerze q stało się aktywne, powinno zachodzić poniższe równanie:

$$(\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(0)} + \alpha^{(0)} (\mathbf{A}_{\mathbf{g}}^T)_q (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) = -(\mathbf{g})_q \tag{3.46}$$

W obliczeniach przyjmujemy, że $\alpha^{(0)}$ jest najmniejszą wartością, dla której jakiekolwiek ograniczenie nierównościowe dla $q \notin \mathscr{A}(\mathbf{d}^{(0)})$ staje się aktywne. Dodatkowo zakładamy, że $\alpha^{(0)} \leqslant 1$. Z (3.46) wynika więc, że

$$\alpha^{(0)} = \min \left\{ 1, \min_{\substack{q \notin \mathscr{A}(\mathbf{d}^{(0)}) \\ (\mathbf{A}_{\mathbf{g}}^T)_q (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)}) > 0}} \frac{(\mathbf{A}_{\mathbf{g}}^T)_q \mathbf{d}^{(0)} + (\mathbf{g})_q}{(\mathbf{A}_{\mathbf{g}}^T)_q (\hat{\mathbf{d}}^{(0)} - \mathbf{d}^{(0)})} \right\} \tag{3.47}$$

Po wyznaczeniu w taki sposób wektora $\mathbf{d}^{(1)}$, w następnej iteracji dołączamy do zbioru $S^{(0)}$ ograniczenie o numerze q, które stało się aktywne w $\mathbf{d}^{(1)}$, a nie było aktywne w $\mathbf{d}^{(0)}$, zatem:

$$S^{(1)} = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{A}_\mathbf{h}^T \mathbf{d} = -\mathbf{h} \\ (\mathbf{A}_\mathbf{g}^T)_\ell \mathbf{d} = -(\mathbf{g})_\ell \quad \text{dla} \quad \ell \in \mathscr{A}(\mathbf{d}^{(0)}) \cup \{q\}\} \tag{3.48}$$

Rozpatrzmy teraz sytuację, w której $\hat{\mathbf{d}}^{(0)} = \mathbf{d}^{(0)}$, tzn. punkt startowy w zadaniu (3.43)–(3.44) jest jednocześnie rozwiązaniem optymalnym tego zadania. Nie oznacza to jednak, że $\hat{\mathbf{d}}^{(0)}$ jest także rozwiązaniem optymalnym zadania ze wszystkimi ograniczeniami równościowymi i nierównościowymi, czyli (3.40)–(3.41). Tak będzie tylko w sytuacji, w której wszystkie mnożniki Lagrange'a $\mu_\ell^{(0)} \geqslant 0$ dla $\ell \in \mathscr{A}(\mathbf{d}^{(0)})$. Wówczas kończymy obliczenia. Ponieważ w zadaniu (3.43)–(3.44) wszystkie ograniczenia są równościowe, w punkcie optymalnym nie musi być spełniony warunek nieujemności związanych z nimi mnożników Lagrange'a. Załóżmy, że istnieje takie $r \in \mathscr{A}(\mathbf{d}^{(0)})$, że $\mu_r^{(0)} < 0$. Oznaczmy $I = \{r \in \{1,\ldots,p+m\} : \mu_r^{(0)} < 0\}$. Pamiętajmy, że $p+m$ jest liczbą wszystkich ograniczeń w zbiorze S. Spośród indeksów związanych z ujemnymi mnożnikami Lagrange'a wybieramy taki indeks q, który odpowiada najmniejszej wartości mnożnika w zbiorze $S^{(0)}$, tzn.

$$\mu_q^{(0)} = \min_{r \in \mathscr{A}(\mathbf{d}^{(0)}) \cap I} \mu_r^{(0)} \tag{3.49}$$

a następnie modyfikujemy zbiór $S^{(0)}$, usuwając z niego ograniczenie nierównościowe o numerze q. Przyjmujemy więc, że:

$$S^{(1)} = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{A}_\mathbf{h}^T \mathbf{d} = -\mathbf{h} \\ (\mathbf{A}_\mathbf{g}^T)_\ell \mathbf{d} = -(\mathbf{g})_\ell \quad \text{dla} \quad \ell \in \mathscr{A}(\mathbf{d}^{(0)}) \backslash \{q\}\} \tag{3.50}$$

i przechodzimy do kolejnej iteracji.

3.5. Metoda obszaru zaufania

Rozpatrzmy najpierw zadanie minimalizacji bez ograniczeń:

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) \tag{3.51}$$

gdzie $\mathbf{x} = [x_1, x_2, \ldots x_n]^T$, a $f : \mathbb{R}^n \to \mathbb{R}$ jest funkcją nieliniową. Załóżmy, że obliczyliśmy wartość funkcji celu f w punkcie $\mathbf{x}^{(0)}$, ale nie jest to punkt optymalny zadania (3.51). W takim przypadku możemy oczywiście wyznaczyć kierunek $\mathbf{d}^{(0)}$, dla którego nastąpi zmniejszenie wartości funkcji f. Jedną z możliwości jest zastosowanie którejś z wcześniej omawianych metod kierunków poprawy. W metodzie obszaru zaufania dążymy także do wyznaczenia takiego kierunku, ale w inny sposób, co prowadzi do innego ciągu $(\mathbf{x}^{(i)})$ przybliżeń rozwiązania optymalnego zadania (3.51). Pod-

stawowym założeniem omawianej metody jest poszukiwanie kierunku $\mathbf{d}^{(i)}$ w tzw. obszarze zaufania, który zwykle jest kulą o promieniu Δ lub elipsoidą w przestrzeni $\mathbb{R}^n$. Zakładamy także, że lokalnie (w otoczeniu $\mathbf{x}^{(i)}$) można dobrze przybliżyć funkcję f za pomocą funkcji kwadratowej q. Korzystając z rozwinięcia funkcji f w szereg Taylora, otrzymujemy:

$$f(\mathbf{x}^{(i)}+\mathbf{d})-f(\mathbf{x}^{(i)}) \approx \frac{1}{2}\mathbf{d}^T\mathbf{H}_{\mathbf{xx}}f(\mathbf{x}^{(i)})\mathbf{d}+[\nabla f(\mathbf{x}^{(i)})]^T\mathbf{d} \tag{3.52}$$

Naturalnym wyborem jest zatem przyjęcie, że

$$q(\mathbf{d}) = \frac{1}{2}\mathbf{d}^T\mathbf{H}_{\mathbf{xx}}f(\mathbf{x}^{(i)})\mathbf{d}+[\nabla f(\mathbf{x}^{(i)})]^T\mathbf{d} \tag{3.53}$$

Konstruujemy pomocnicze zadanie optymalizacji z funkcją q daną wzorem (3.53):

$$\min_{\mathbf{d}\in N} q(\mathbf{d}) \quad \text{gdzie} \quad N=\{\mathbf{d}\in\mathbb{R}^n : \|\mathbf{D}\mathbf{d}\| \leqslant \Delta\} \tag{3.54}$$

przy czym $\mathbf{D}$ jest macierzą diagonalną. W szczególnym przypadku, gdy $\mathbf{D}=\mathbf{I}$ ($\mathbf{D}$ jest macierzą jednostkową), obszar zaufania N jest kulą. Dobór współczynników macierzy $\mathbf{D}$ pozwala na dopasowanie kształtu obszaru zaufania do lokalnego układu poziomic funkcji f. Mimo że funkcja q w zadaniu (3.54) jest kwadratowa, to nie jest to zadanie programowania kwadratowego ze względu na ograniczenie, które nie jest liniowe. Okazuje się jednak, że dobre przybliżenie rozwiązania optymalnego zadania (3.54) można zwykle uzyskać, redukując przestrzeń poszukiwań do dwóch wymiarów. Załóżmy na razie, że macierz $\mathbf{H}_{\mathbf{xx}}f(\mathbf{x}^{(i)})$ jest dodatnio określona. Takie założenie jest słuszne w pobliżu minimum lokalnego funkcji f, ale jeżeli f nie jest wypukła, nie musi być spełnione w każdej iteracji. Odpowiednia podprzestrzeń jest wtedy generowana przez kierunek $\mathbf{d}_1$ najszybszego spadku oraz kierunek $\mathbf{d}_2$ odpowiadający metodzie Newtona, tzn.:

$$\mathbf{d}_1 = -\nabla f(\mathbf{x}^{(i)}), \qquad \mathbf{d}_2 = -[\mathbf{H}_{\mathbf{xx}}f(\mathbf{x}^{(i)})]^{-1}\nabla f(\mathbf{x}^{(i)}) \tag{3.55}$$

Zamiast zadania (3.54) rozwiązujemy zadanie:

$$\min_{\mathbf{d}\in N_0} q(\mathbf{d}) \quad \text{gdzie} \quad N_0=\{\mathbf{d}\in \operatorname{span}\{\mathbf{d}_1,\mathbf{d}_2\} : \|\mathbf{D}\mathbf{d}\| \leqslant \Delta\} \tag{3.56}$$

co jest znacznie prostsze. Oznaczmy przez $\hat{\mathbf{d}}$ rozwiązanie optymalne zadania (3.56).

Jeżeli w kierunku $\hat{\mathbf{d}}$ nastąpiło zmniejszenie wartości funkcji f, tzn. $f(\mathbf{x}^{(i)}+\hat{\mathbf{d}}) < f(\mathbf{x}^{(i)})$, to przyjmujemy $\mathbf{x}^{(i+1)}=\mathbf{x}^{(i)}+\hat{\mathbf{d}}$ i przechodzimy do kolejnej iteracji. W przeciwnym razie zmniejszamy Δ (rozmiar obszaru zaufania) i rozwiązujemy ponownie zadanie (3.56).

Jeżeli przynajmniej jedna wartość własna macierzy $\mathbf{H}_{\mathbf{xx}}f(\mathbf{x}^{(i)})$ jest ujemna, należy przyjąć

$$\mathbf{d}_2 = -[\mathbf{H}_{\mathbf{xx}} f(\mathbf{x}^{(i)}) + \alpha \mathbf{I}]^{-1} \nabla f(\mathbf{x}^{(i)}) \quad \text{dla pewnego} \quad \alpha \in (-\ell_1; -2\ell_1) \tag{3.57}$$

gdzie ℓ_1 jest najmniejszą wartością własną. Taki wybór zapewnia dodatnią określoność macierzy $\mathbf{H}_{\mathbf{xx}} f(\mathbf{x}^{(i)}) + \alpha \mathbf{I}$. Jeżeli przynajmniej jedna wartość własna jest równa zeru, ale nie ma wartości własnych ujemnych, nie bierzemy pod uwagę kierunku $\mathbf{d}_2$ i przyjmujemy

$$\hat{\mathbf{d}} = \frac{\mathbf{d}_1}{\|\mathbf{d}_1\|} \Delta \tag{3.58}$$

Rozpatrzmy teraz zadanie optymalizacji z ograniczeniami (3.1)–(3.2). Aby wyznaczyć kierunek $\hat{\mathbf{d}}$, możemy rozwiązać zadanie podobne do (3.34)–(3.35), które rozważaliśmy w opisie metody sekwencyjnego programowania kwadratowego. Różnica polega na uwzględnieniu dodatkowego ograniczenia związanego z obszarem zaufania. Rozważamy zatem następujące zadanie:

$$\min_{\mathbf{d} \in S \cap B} \frac{1}{2} \mathbf{d}^T \mathbf{H}_{\mathbf{xx}} L(\mathbf{x}^{(i)}, \boldsymbol{\lambda}^{(i)}, \boldsymbol{\mu}^{(i)}) \mathbf{d} + [\nabla f(\mathbf{x}^{(i)})]^T \mathbf{d} \tag{3.59}$$

gdzie:

$$\begin{aligned} S = \{\mathbf{d} \in \mathbb{R}^n : h_k(\mathbf{x}^{(i)}) + [\nabla h_k(\mathbf{x}^{(i)})]^T \mathbf{d} = 0 \quad &\text{dla} \quad k = 1, \ldots, p, \\ g_j(\mathbf{x}^{(i)}) + [\nabla g_j(\mathbf{x}^{(i)})]^T \mathbf{d} \leqslant 0 \quad &\text{dla} \quad j = 1, \ldots, m\} \\ B = \{\mathbf{d} \in \mathbb{R}^n : \|\mathbf{D}\mathbf{d}\| \leqslant \Delta\} \end{aligned} \tag{3.60}$$

Zbiór $S \cap B$ może być jednak pusty dla małych wartości Δ, nawet jeżeli $S \neq \emptyset$. Wraz ze wzrostem Δ tracimy lokalny charakter obszaru zaufania, dlatego zbyt duże zwiększanie wartości tego parametru nie jest wskazane. Jeżeli $S \cap B = \emptyset$, możemy przeskalować zbiór S, tzn. wziąć zamiast S zbiór $S_\theta = \theta S$ dla pewnego $\theta \in (0; 1\rangle$ – tak, aby $S_\theta \cap B \neq \emptyset$.

3.6. Metoda punktu wewnętrznego

Idea metody punktu wewnętrznego opiera się w pewnej mierze na koncepcji obecnej w metodzie wewnętrznej funkcji kary, która została przedstawiona w rozdziale 3.2.2. Nie wymagamy jednak teraz, aby punkt startowy znajdował się w obszarze dopuszczalnym, ponadto będziemy mogli uwzględnić również ograniczenia równościowe. Rozpatrujemy zatem zadanie optymalizacji z ograniczeniami równościowymi i nierównościowymi (3.1)–(3.2). Wprowadzimy do tego zadania najpierw dodatkowe nieujemne zmienne $s_1, \ldots, s_m$, dzięki którym ograniczenia nierównościowe zostaną zamienione na równościowe, a następnie zamiast funkcji celu f weźmiemy funkcję

$$f_\rho(\mathbf{x},\mathbf{s}) = f(\mathbf{x}) - \rho \sum_{j=1}^{m} \ln s_j \tag{3.61}$$

gdzie $\mathbf{s} = [s_1,\ldots,s_m]^T$. Sformułujemy zatem zadanie pomocnicze:

$$\min_{(\mathbf{x},\mathbf{s})\in D_s} f_\rho(\mathbf{x},\mathbf{s}) \tag{3.62}$$

gdzie

$$\begin{aligned} D_s = \{(\mathbf{x},\mathbf{s}) \in \mathbb{R}^{n+m} : & \; h_k(\mathbf{x}) = 0 \quad \text{dla} \quad k = 1,\cdots p; \\ & \; g_j(\mathbf{x}) + s_j = 0 \quad \text{dla} \quad j = 1,\ldots,m; \\ & \; s_j \geqslant 0 \quad \text{dla} \quad j = 1,\ldots,m\} \end{aligned} \tag{3.63}$$

Funkcja f_ρ, dana wzorem (3.61), jest identyczna jak funkcja kary w opisie metody wewnętrznej funkcji kary, teraz jednak zależy od $n+m$ zmiennych – zmienne $s_1,\ldots,s_m$ traktujemy jako niezależne od zmiennych $x_1,\ldots,x_n$. Funkcja Lagrange'a dla zadania (3.62)–(3.63) ma postać:

$$L(\mathbf{x},\mathbf{s},\boldsymbol{\lambda},\boldsymbol{\mu}) = f(x) - \rho \sum_{j=1}^{m} \ln s_j + \sum_{k=1}^{p} \lambda_k h_k(\mathbf{x}) + \sum_{j=1}^{m} \mu_j (g_j(\mathbf{x}) + s_j) \tag{3.64}$$

Ze względu na zgodność z poprzednimi oznaczeniami zachowujemy notację $\mu_1,\ldots,\mu_m$ na oznaczenie mnożników Lagrange'a, mimo że dotyczą one teraz ograniczeń równościowych. Zgodnie z Uwagą 3.1 warunki konieczne Karusha-Kuhna-Tuckera, które muszą być spełnione w punkcie $(\hat{\mathbf{x}},\hat{\mathbf{s}})$ minimalizującym funkcję f_ρ, możemy zapisać w postaci układu równań:

$$\begin{cases} \nabla_{\mathbf{x}} L(\hat{\mathbf{x}},\hat{\mathbf{s}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\mu}}) &= \mathbf{0} \\ \nabla_{\mathbf{s}} L(\hat{\mathbf{x}},\hat{\mathbf{s}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\mu}}) &= \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} L(\hat{\mathbf{x}},\hat{\mathbf{s}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\mu}}) &= \mathbf{0} \\ \nabla_{\boldsymbol{\mu}} L(\hat{\mathbf{x}},\hat{\mathbf{s}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\mu}}) &= \mathbf{0} \end{cases} \tag{3.65}$$

Przeprowadzając podobne rozumowanie jak na początku podrozdziału 3.3, otrzymujemy najpierw:

$$\begin{aligned} L(\mathbf{x}+\mathbf{dx},\mathbf{s}+\mathbf{ds},\boldsymbol{\lambda}+\mathbf{d\lambda},\boldsymbol{\mu}+\mathbf{d\mu}) \approx L(\mathbf{x},\mathbf{s},\boldsymbol{\lambda},\boldsymbol{\mu}) + \sum_{i=1}^{n} \frac{\partial L}{\partial x_i} dx_i + \sum_{j=1}^{m} \frac{\partial L}{\partial s_j} ds_j + \\ + \sum_{k=1}^{p} \frac{\partial L}{\partial \lambda_k} d\lambda_k + \sum_{j=1}^{m} \frac{\partial L}{\partial \mu_j} d\mu_j \end{aligned} \tag{3.66}$$

Podobnie jak w (3.29)–(3.31), stosując dla uproszczenia zapisu oznaczenia (3.39), mamy:

$$\nabla_{\mathbf{x}} L(\mathbf{x}^{(i)}+\mathbf{dx},\mathbf{s}^{(i)}+\mathbf{ds},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}) \approx \mathbf{c}+\mathbf{F}\mathbf{dx}+ \\ +\mathbf{A}_{\mathbf{h}}(\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda})+\mathbf{A}_{\mathbf{g}}(\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}) \tag{3.67}$$

$$\nabla_{\mathbf{s}} L(\mathbf{x}^{(i)}+\mathbf{dx},\mathbf{s}^{(i)}+\mathbf{ds},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}) \approx -\rho\mathbf{S}^{-1}\mathbf{e}+\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}+\rho[\mathbf{S}^{-1}]^2\mathbf{ds} \tag{3.68}$$

$$\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^{(i)}+\mathbf{dx},\mathbf{s}^{(i)}+\mathbf{ds},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}) \approx \mathbf{h}^T+\mathbf{A}_{\mathbf{h}}^T\mathbf{dx} \tag{3.69}$$

$$\nabla_{\boldsymbol{\mu}} L(\mathbf{x}^{(i)}+\mathbf{dx},\mathbf{s}^{(i)}+\mathbf{ds},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}) \approx \mathbf{g}^T+\mathbf{A}_{\mathbf{g}}^T\mathbf{dx}+\mathbf{s}^{(i)}+\mathbf{ds} \tag{3.70}$$

We wzorze (3.67) macierz $\mathbf{F}$ oczywiście zależy także od $\mathbf{s}$, tzn.

$$\mathbf{F}=\mathbf{H}_{\mathbf{xx}}L(\mathbf{x}^{(i)},\mathbf{s}^{(i)},\boldsymbol{\lambda}^{(i)},\boldsymbol{\mu}^{(i)}) \tag{3.71}$$

We wzorze (3.68) przyjęliśmy dodatkowe oznaczenia: $\mathbf{S}$ to macierz diagonalna o współczynnikach $s_1,\ldots,s_m$ na przekątnej, natomiast $\mathbf{e}=[1,\ldots,1]^T\in\mathbb{R}^m$.

W metodzie punktu wewnętrznego w kolejnych iteracjach szukamy przybliżenia punktu optymalnego $(\hat{\mathbf{x}},\hat{\mathbf{s}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\mu}})$, rozwiązując układ równań mający postać:

$$\begin{cases} \nabla_{\mathbf{x}} L(\mathbf{x}^{(i)}+\mathbf{dx}^{(i)},\mathbf{s}^{(i)}+\mathbf{ds}^{(i)},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda}^{(i)},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}^{(i)}) &= \mathbf{0} \\ \nabla_{\mathbf{s}} L(\mathbf{x}^{(i)}+\mathbf{dx}^{(i)},\mathbf{s}^{(i)}+\mathbf{ds}^{(i)},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda}^{(i)},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}^{(i)}) &= \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^{(i)}+\mathbf{dx}^{(i)},\mathbf{s}^{(i)}+\mathbf{ds}^{(i)},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda}^{(i)},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}^{(i)}) &= \mathbf{0} \\ \nabla_{\boldsymbol{\mu}} L(\mathbf{x}^{(i)}+\mathbf{dx}^{(i)},\mathbf{s}^{(i)}+\mathbf{ds}^{(i)},\boldsymbol{\lambda}^{(i)}+\mathbf{d}\boldsymbol{\lambda}^{(i)},\boldsymbol{\mu}^{(i)}+\mathbf{d}\boldsymbol{\mu}^{(i)}) &= \mathbf{0} \end{cases} \tag{3.72}$$

Po pomnożeniu drugiego równania z układu (3.72) przez $\mathbf{S}$ otrzymamy w zapisie macierzowym:

$$\begin{bmatrix} \mathbf{F} & \mathbf{0} & \mathbf{A_h} & \mathbf{A_g} \\ \mathbf{0} & \rho\mathbf{S}^{-1} & \mathbf{0} & \mathbf{S} \\ \mathbf{A}_{\mathbf{h}}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{\mathbf{g}}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{dx}^{(i)} \\ \mathbf{ds}^{(i)} \\ \mathbf{d}\boldsymbol{\lambda}^{(i)}) \\ \mathbf{d}\boldsymbol{\mu}^{(i)}) \end{bmatrix} = \begin{bmatrix} -\mathbf{c}+\mathbf{A_h}\boldsymbol{\lambda}^{(i)}-\mathbf{A_g}\boldsymbol{\mu}^{(i)} \\ \rho\mathbf{e}-\mathbf{S}\boldsymbol{\mu}^{(i)} \\ -\mathbf{h} \\ -\mathbf{g}-\mathbf{s} \end{bmatrix} \tag{3.73}$$

Powyższy układ możemy zapisać równoważnie:

$$\begin{bmatrix} \mathbf{F} & \mathbf{0} & \mathbf{A_h} & \mathbf{A_g} \\ \mathbf{0} & \rho\mathbf{I} & \mathbf{0} & \mathbf{S} \\ \mathbf{A}_{\mathbf{h}}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{\mathbf{g}}^T & \mathbf{S} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{dx}^{(i)} \\ \mathbf{S}^{-1}\mathbf{ds}^{(i)} \\ \mathbf{d}\boldsymbol{\lambda}^{(i)}) \\ \mathbf{d}\boldsymbol{\mu}^{(i)}) \end{bmatrix} = \begin{bmatrix} -\mathbf{c}+\mathbf{A_h}\boldsymbol{\lambda}^{(i)}-\mathbf{A_g}\boldsymbol{\mu}^{(i)} \\ \rho\mathbf{e}-\mathbf{S}\boldsymbol{\mu}^{(i)} \\ -\mathbf{h} \\ -\mathbf{g}-\mathbf{s} \end{bmatrix} \tag{3.74}$$

Macierz układu jest teraz symetryczna, dzięki czemu możemy zastosować szerszą klasę metod do rozwiązania takiego układu.

Jeżeli możemy wyznaczyć rozwiązanie układu (3.74) (jest tak m.in. w przypadku, gdy (3.62)–(3.63) jest lokalnie wypukłe w otoczeniu bieżącego przybliżenia rozwiązania optymalnego), to kolejne przybliżenie wyznaczamy ze wzorów:

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \alpha_{\mathbf{s}}^{(i)} \mathbf{d}\mathbf{x}^{(i)}, \qquad \mathbf{s}^{(i+1)} = \mathbf{s}^{(i)} + \alpha_{\mathbf{s}}^{(i)} \mathbf{d}\mathbf{s}^{(i)} \tag{3.75}$$

$$\boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \alpha_{\boldsymbol{\mu}}^{(i)} \mathbf{d}\boldsymbol{\lambda}^{(i)}, \qquad \boldsymbol{\mu}^{(i+1)} = \boldsymbol{\mu}^{(i)} + \alpha_{\boldsymbol{\mu}}^{(i)} \mathbf{d}\boldsymbol{\mu}^{(i)} \tag{3.76}$$

przy czym w celu wyznaczenia współczynników $\alpha_{\mathbf{s}}^{(i)}$ i $\alpha_{\boldsymbol{\mu}}^{(i)}$ obliczamy najpierw ich maksymalne wartości ze wzorów:

$$\begin{aligned} \alpha_{\mathbf{s}}^{\max} &= \max\{\alpha \in (0;1\rangle : \mathbf{s}^{(i)} + \alpha \mathbf{d}\mathbf{s}^{(i)} \geqslant \varepsilon \mathbf{s}^{(i)}\} \\ \alpha_{\boldsymbol{\mu}}^{\max} &= \max\{\alpha \in (0;1\rangle : \boldsymbol{\mu}^{(i)} + \alpha \mathbf{d}\boldsymbol{\mu}^{(i)} \geqslant \varepsilon \mathbf{s}^{(i)}\} \end{aligned} \tag{3.77}$$

gdzie ε jest małą liczbą dodatnią, np. $\varepsilon = 0,005$. Następnie, analogicznie jak w metodzie sekwencyjnego programowania kwadratowego, sprawdzamy, czy uzyskujemy wystarczający spadek wartości funkcji pomocniczej $\Psi^{(i)}$ (ang. *merit function*) przy przejściu od $\mathbf{x}^{(i)}$ do $\mathbf{x}^{(i)} + \alpha_{\mathbf{s}}^{(i)} \mathbf{d}\mathbf{x}^{(i)}$ i od $\mathbf{s}^{(i)}$ do $\mathbf{s}^{(i)} + \alpha_{\mathbf{s}}^{(i)} \mathbf{d}\mathbf{s}^{(i)}$. Funkcja $\Psi^{(i)}$ ma teraz postać:

$$\Psi^{(i)}(\mathbf{x}, \mathbf{s}) = f(\mathbf{x}) - \rho^{(i)} \sum_{j=1}^{m} \ln s_j + \nu^{(i)}(\|\mathbf{h}\| + \|\mathbf{g} + \mathbf{s}\|) \tag{3.78}$$

Współczynnik $\nu^{(i)}$, który pojawił się w ostatnim wzorze, jest związany z obecnością ograniczeń w zadaniu (3.62)–(3.63). Odpowiednie zwiększanie tego współczynnika w kolejnych iteracjach ma za cel wymuszenie zbieżności kolejnych przybliżeń $(\mathbf{x}^{(i)}, \mathbf{s}^{(i)})$ do punktu, który należy do zbioru dopuszczalnego D_s.

W trakcie obliczeń powinien być też zmniejszany współczynnik funkcji kary ρ od początkowej wartości $\rho^{(0)}$. W podstawowej wersji algorytmu w każdej iteracji wykorzystuje się test:

$$\max(\|\mathbf{c} - \mathbf{A}_{\mathbf{h}} \boldsymbol{\lambda}^{(i)} - \mathbf{A}_{\mathbf{g}} \boldsymbol{\mu}^{(i)}\|, \|\rho^{(i)} \mathbf{e} - \mathbf{S}\boldsymbol{\mu}^{(i)}\|, \|\mathbf{h}\|, \|\mathbf{g} + \mathbf{s}\|) < \rho^{(i)} \tag{3.79}$$

Zauważmy, że w powyższym teście sprawdzamy, czy wszystkie normy wyrazów prawej strony układu równań (3.74) są mniejsze od bieżącej wartości współczynnika kary $\rho^{(i)}$. Jeżeli nierówność w teście nie jest spełniona (algorytm nie osiągnął w danej iteracji wystarczającej dokładności), to wykonujemy kolejną iterację bez zmiany wartości parametru ρ, tzn. $\rho^{(i+1)} = \rho^{(i)}$. W przeciwnym razie zmniejszamy wartość ρ zgodnie ze wzorem: $\rho^{(i+1)} = \sigma \rho^{(i)}$, gdzie σ jest ustaloną liczbą z przedziału $(0;1)$.

W przypadku, gdy zadanie optymalizacji jest niewypukłe, powyższy sposób postępowania nie musi prowadzić do celu, gdyż ciąg rozwiązań przybliżonych $(\mathbf{x}^{(i)})$

może nie być zbieżny do rozwiązania optymalnego $\hat{\mathbf{x}}$, ale do lokalnego maksimum funkcji celu f albo do innego punktu stacjonarnego tej funkcji. Wówczas, jeżeli w otoczeniu punktu $(\mathbf{x}^{(i)}, \mathbf{s}^{(i)})$ zadanie (3.62)–(3.63) nie jest lokalnie wypukłe, należy zastosować przy przejściu z $\mathbf{x}^{(i)}$ do $\mathbf{x}^{(i+1)}$ inny schemat postępowania. Jednym ze sposobów jest wtedy wyznaczenie kierunków $\mathbf{dx}^{(i)}$ i $\mathbf{ds}^{(i)}$ na podstawie rozwiązania pomocniczego zadania optymalizacji:

$$\min_{(\mathbf{dx},\mathbf{ds})\in D_z} \mathbf{c}^T\mathbf{dx} + \frac{1}{2}(\mathbf{dx})^T\mathbf{F}\mathbf{dx} + \rho^{(i)}\mathbf{e}^T\mathbf{S}^{-1}\mathbf{ds} + \frac{1}{2}\rho^{(i)}(\mathbf{ds})^T\mathbf{ds} \tag{3.80}$$

gdzie

$$\begin{aligned} D_z = \{(\mathbf{dx},\mathbf{ds}) \in \mathbb{R}^{n+m} : \; & \mathbf{A}_\mathbf{h}^T\mathbf{dx} = -\mathbf{h}, \\ & \mathbf{A}_\mathbf{g}^T\mathbf{dx} + \mathbf{ds} = -\mathbf{g} - \mathbf{s}, \\ & \|[\mathbf{dx}, \mathbf{S}^{-1}\mathbf{ds}]\| \leqslant \Delta, \\ & \tau\mathbf{s} + \mathbf{ds} \geqslant 0\} \end{aligned} \tag{3.81}$$

przy czym Δ jest promieniem obszaru zaufania określonego przez ograniczenie zapisane w trzecim wierszu (3.81), natomiast $\tau \in (0;1)$. Jednocześnie przyjmujemy, że τ jest bliskie 1, np. $\tau = 0,995$. Jeżeli przez $(\widehat{\mathbf{dx}}, \widehat{\mathbf{ds}})$ oznaczymy przybliżone rozwiązanie powyższego zadania optymalizacji, to podstawiamy $\mathbf{dx}^{(i)} = \widehat{\mathbf{dx}}$, $\mathbf{ds}^{(i)} = \widehat{\mathbf{ds}}$, a następnie obliczamy $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \mathbf{dx}^{(i)}$ oraz $\mathbf{s}^{(i+1)} = \mathbf{s}^{(i)} + \mathbf{ds}^{(i)}$.

3.7. Procedury w MATLAB-ie

Zbiór D ograniczeń zadania optymalizacji (3.1) jest w MATLAB-ie określony jako część wspólna poniższych typów ograniczeń:

- $\mathbf{g}(\mathbf{x}) \leqslant \mathbf{0}$, gdzie $\mathbf{g} = [g_1, \ldots, g_p]^T$ – funkcje nieliniowe, $\mathbf{x} = [x_1, \ldots, x_n]^T$, $\mathbf{0} = [0, \ldots, 0]^T$;
- $\mathbf{h}(\mathbf{x}) = \mathbf{0}$, gdzie $\mathbf{h} = [h_1, \ldots, h_q]^T$ – funkcje nieliniowe;
- $\mathbf{A}\mathbf{x} \leqslant \mathbf{b}$, gdzie $\mathbf{A}$ – macierz $m \times n$ o stałych współczynnikach, $\mathbf{b} = [b_1, \ldots, b_m]^T$;
- $\mathbf{A_{eq}}\mathbf{x} = \mathbf{b_{eq}}$, gdzie $\mathbf{A_{eq}}$ – macierz $r \times n$ o stałych współczynnikach, $\mathbf{b_{eq}} = [b_{eq_1}, \ldots, b_{eq_r}]^T$;
- $\mathbf{xl} \leqslant \mathbf{x} \leqslant \mathbf{xu}$, gdzie $\mathbf{xl}, \mathbf{xu}$ – wektory liczbowe n-elementowe.

Do wyznaczania minimum funkcji wielu zmiennych z ograniczeniami możemy wykorzystać procedurę `fmincon`.

Cechy charakterystyczne procedury `fmincon`:

- Zarówno funkcja celu f, jak i wszystkie funkcje ograniczeń $g_1, \ldots, g_p$ oraz $h_1, \ldots, h_q$ powinny mieć ciągłe pochodne cząstkowe pierwszego rzędu.

- Obliczenia mogą być wykonywane za pomocą jednej z kilku metod opisanych w podrozdziałach 3.3–3.6: punktu wewnętrznego (domyślna), obszaru zaufania, sekwencyjnego programowania kwadratowego i zbioru ograniczeń aktywnych.

Przykłady 3.6–3.9 przedstawiają działanie procedury `fmincon` z wykorzystaniem różnych metod w niej zaimplementowanych.

Przykład 3.6. Korzystając z procedury `fmincon`, wyznaczyć minimum funkcji $f(x_1,x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ na zbiorze ograniczeń $D = \{(x_1,x_2) : x_1^2 + (x_2+2)^2 \leqslant 1\}$ metodą punktu wewnętrznego. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = [0,-2]^T$.

Rozwiązanie:
Mamy do czynienia z jednym nieliniowym ograniczeniem nierównościowym: $g(x_1,x_2) = x_1^2 + (x_2+2)^2 - 1 \leqslant 0$.

Procedura `f` w pliku **f.mlx**, w której definujemy funkcję f i jej gradient, powinna mieć identyczną postać jak w przykładzie 2.8.
Tworzymy także plik **test_fmincon.mlx**, którego zawartość jest prawie identyczna jak pliku **test_fminunc.mlx** z przykładu 2.8. Poniższy listing przedstawia jedynie te linie pliku **test_fmincon.mlx**, które są inne:

Listing 3.1: Procedura `test_fmincon`

```
function [x,f,x_all] = test_fmincon(x0,n)
.............................................................
  opcje = optimoptions('fmincon','Display','iter','MaxIterations',n,
            'OutputFcn',@outf,'SpecifyObjectiveGradient',true,
            'SpecifyConstraintGradient',true,'HessianFcn',@hesjan);
  [x,f] = fmincon(@f,x0,[],[],[],[],[],[],@nielin,opcje);
.............................................................
end
```

W wywołaniu procedury `optimoptions` pojawiły się dwie nowe pary argumentów: `'SpecifyConstraintGradient',true` oraz `'HessianFcn', @hesjan`. Pierwsza z nich oznacza, że gradienty nieliniowych ograniczeń (w tym przykładzie mamy tylko jedno takie ograniczenie) będą obliczane dokładnie na podstawie wzorów podanych przez użytkownika. Podobne znaczenie ma druga para argumentów – dotyczy dokładnego obliczania hesjanu (zarówno funkcji celu, jak i ograniczeń). W tym przypadku `hesjan` jest nazwą funkcji, która powinna być utworzona przez użytkownika.

Wywołanie procedury `fmincon` zawiera sześć pustych argumentów postaci `[]`. Odpowiadają one innym typom ograniczeń, które nie występują w tym przykładzie. Przedostatni argument (`@nielin`) wskazuje na nazwę procedury, która powinna być utworzona przez użytkownika i zawierać definicję funkcji nieliniowych ograniczeń i ich gradientów. W naszym przypadku procedura `nielin` ma postać:

Listing 3.2: Procedura `nielin`

```
function [g,h,grad_g,grad_h] = nielin(x)
  g = x(1)^2+(x(2)+2)^2-1;
  h = [];
  if nargout>2
    grad_g = [2*x(1); 2*(x(2)+2)];
    grad_h = [];
  end
end
```

Mimo że w przykładzie nie ma nieliniowych ograniczeń równościowych, procedura `fmincon` wymaga, aby na wyjściu procedury `nielin` pojawiła się także zmienna `h` reprezentująca takie ograniczenia. Ponieważ zadeklarowaliśmy, że podamy wzory na gradienty nieliniowych ograniczeń, to dodatkowo reprezentujące je zmienne `grad_g` i `grad_h` powinny być przekazane jako trzeci i czwarty argument wyjściowy procedury `nielin`.

Wspomniana wyżej procedura `hesjan` powinna mieć postać:

Listing 3.3: Procedura `hesjan`

```
function H = hesjan(x,lambda)
  Hf = [4-12.6*x(1)^2+5*x(1)^4, 1; 1, 2];
  Hg = [2 0; 0 2];
  H = Hf+lambda.ineqnonlin*Hg;
end
```

Zmienna `Hf` przechowuje wartość hesjanu funkcji celu f, w zmiennej `Hg` zapamiętany jest hesjan funkcji nieliniowych ograniczeń. Zmienna `lambda` jest strukturą zawierającą mnożniki Lagrange'a: pole `ineqnonlin` jest wektorem mnożników dla nieliniowych ograniczeń nierównościowych.

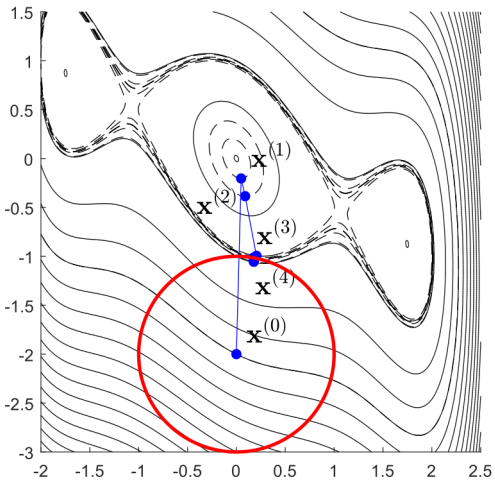
Warto zanotować, że procedura `optimoptions` pozwala na bardzo dużą elastyczność konfiguracji parametrów, z jakimi będzie wywoływana właściwa procedura optymalizacyjna `fmincon`. W szczególności pominięcie opcji `'HessianFcn'` oznacza aproksymację hesjanu, domyślnie za pomocą wzoru takiego jak w metodzie zmiennej metryki BFGS. Należy zaznaczyć, że o ile wyznaczenie wzoru dokładnego nie jest zbyt pracochłonne, lepiej wykorzystać taką możliwość, aby przyspieszyć zbieżność procesu optymalizacji.

Przed wywołaniem procedury `test_fmincon` przyjmujemy współrzędne punktu startowego `x0 = [0,-2]` i otrzymujemy:

Listing 3.4: Wynik działania procedury `fmincon` w metodzie punktu wewnętrznego

```
                                                First-order      Norm of
 Iter F-count            f(x)  Feasibility     optimality         step
    0       1    4.000000e+00    0.000e+00    4.000e+00
    1       2    3.637837e-02    2.228e+00    1.000e-01    1.797e+00
    2       3    1.297634e-01    1.616e+00    2.956e-01    1.858e-01
    3       4    8.675273e-01    5.295e-02    4.489e-01    6.205e-01
    4       6    8.939166e-01    2.171e-02    2.280e-01    2.032e-02
    5       7    9.895645e-01    0.000e+00    7.726e-02    5.023e-02
    6       8    9.354264e-01    0.000e+00    2.199e-02    2.862e-02
    7       9    9.142293e-01    0.000e+00    5.645e-04    1.151e-02
    8      10    9.138648e-01    0.000e+00    2.000e-04    2.053e-04
    9      11    9.136668e-01    0.000e+00    2.033e-06    1.080e-04
   10      12    9.136648e-01    0.000e+00    2.000e-08    1.099e-06

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is
non-decreasing in feasible directions, to within the value
of the optimality tolerance, and constraints are satisfied
to within the value of the constraint tolerance.

<stopping criteria details>
```

W kolumnie `Feasibility` wartość 0 oznacza, że punkt $\mathbf{x}^{(k)}$ uzyskany w k-tej iteracji jest dopuszczalny (leży w zbiorze ograniczeń). Im wartość w tej kolumnie jest większa, tym dalej od brzegu zbioru ograniczeń jest położony punkt $\mathbf{x}^{(k)}$.



Rysunek 3.6: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody punktu wewnętrznego

Kolorem czerwonym na powyższym rysunku zaznaczono brzeg zbioru ograniczeń. □

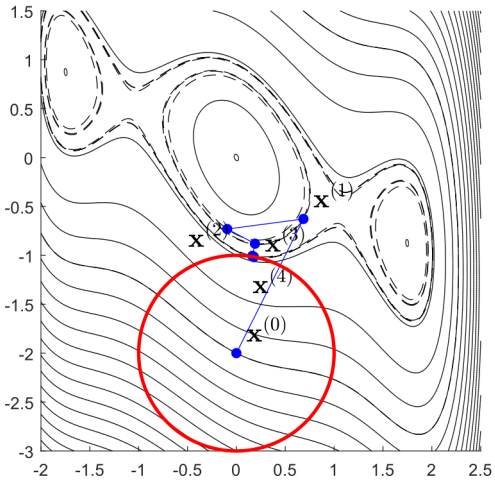
Przykład 3.7. Wykonać obliczenia dla danych z przykładu 3.6 metodą sekwencyjnego programowania kwadratowego.

Rozwiązanie:
Zastosowanie tej metody wymaga dwóch modyfikacji w wywołaniu procedury `optimoptions` względem jej parametrów w przykładzie 3.6. Po pierwsze, powinniśmy zaznaczyć, że korzystamy z innego algorytmu niż domyślny, dodając parę argumentów: `'Algorithm','sqp'`. Po drugie, w metodzie sekwencyjnego programowania kwadratowego zaimplementowanej w MATLAB-ie hesjan funkcji jest zawsze obliczany w sposób przybliżony, dlatego należy usunąć parę argumentów `'HessianFcn',@hesjan`, gdzie `hesjan` jest nazwą procedury wyznaczającej hesjan w sposób dokładny. Po dokonaniu takich zmian otrzymamy wyniki:

Listing 3.5: Wynik działania procedury `fmincon` w metodzie programowania sekwencyjnego

```
 Iter Func-count         Fval   Feasibility     Norm of  First-order
                                                   step   optimality
    0        1    4.000000e+00    0.000e+00    0.000e+00    4.000e+00
    1        6    6.896041e-01    1.353e+00    1.534e+00    9.120e-01
    2        8    6.207447e-01    6.189e-01    7.867e-01    1.214e+00
    3       11    6.792337e-01    2.889e-01    3.212e-01    2.883e-01
    4       13    8.989170e-01    1.621e-02    1.273e-01    2.492e-01
    5       15    9.133657e-01    3.275e-04    1.810e-02    9.227e-03
    6       17    9.136614e-01    3.630e-06    1.905e-03    2.367e-04
    7       19    9.136648e-01    2.963e-10    1.721e-05    5.844e-07

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is
non-decreasing in feasible directions, to within the value
of the optimality tolerance, and constraints are satisfied
to within the value of the constraint tolerance.

<stopping criteria details>
```

Ilość iteracji głównej metody w tym przykładzie jest mniejsza niż przy zastosowaniu algorytmu punktu wewnętrznego, ale całkowita ilość obliczeń wartości funkcji celu większa (kolumna `F-count` na listingu 3.4 i `Func-count` na listingu 3.5). Ponadto wszystkie punkty $\mathbf{x}^{(k)}$ otrzymane w kolejnych iteracjach metody sekwencyjnego programowania kwadratowego leżą poza zbiorem ograniczeń – wynika to z dodatnich wartości w kolumnie `Feasibility`.



Rysunek 3.7: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody sekwencyjnego programowania kwadratowego

□

Przykład 3.8. Wykonać obliczenia dla danych z przykładu 3.6 z wykorzystaniem metody zbioru ograniczeń aktywnych.

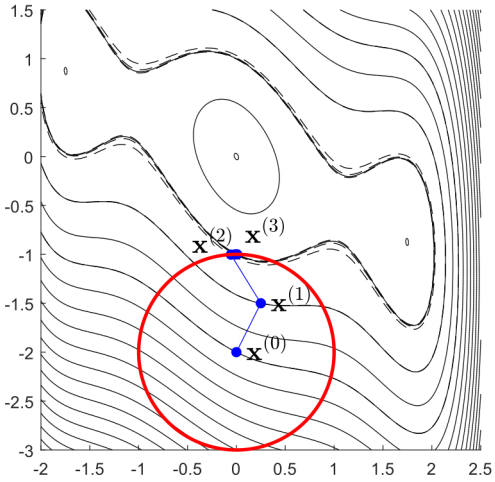
Rozwiązanie:
W MATLAB-ie metoda zbioru ograniczeń aktywnych może zostać wykorzystana jako podrzędna do rozwiązywania zadań programowania kwadratowego, gdy główną jest metoda sekwencyjnego programowania kwadratowego. Można sformułować identyczne uwagi dotyczące modyfikacji wywołania procedury `optimoptions` jak w przykładzie 3.7, z tą różnicą, że teraz odpowiednia para jej argumentów powinna mieć postać: `'Algorithm','active-set'`. Okazuje się, że zastosowanie tej metody do rozwiązania rozpatrywanego zadania optymalizacji powoduje konieczność obliczenia wielu wartości funkcji celu (kolumna `F-count`). Przy domyślnych wartościach parametrów występujących w kryteriach zatrzymania algorytmu działanie procedury `fmincon` zakończy się komunikatem `Solver stopped prematurely`, co wskazuje na to, że rozwiązanie optymalne nie zostało osiągnięte. Możemy zwiększyć limit obliczeń wartości funkcji celu, dołączając do argumentów wywołania procedury `optimoptions` parę: `'MaxFunctionEvaluations',mf`, gdzie `mf` jest liczbą oznaczającą nową wartość limitu. Po obliczeniach otrzymamy:

Listing 3.6: Wynik działania procedury `fmincon` z wykorzystaniem metody zbioru ograniczeń aktywnych

```
                                    Max      Line search   First-order
 Iter F-count        f(x)    constraint   steplength    optimality
                                                        Procedure
    0       1           4            -1
    1       6     1.99594       -0.6875        0.125          2.75
    2       9       1.069     -0.003078          0.5          1.62
    3      14     1.01676    -0.0009915        0.125          1.09
    4      19    0.992398    -0.0003289        0.125         0.953
    5      25    0.982898    -0.0002047       0.0625         0.894
    6      31    0.974544    -0.0001004       0.0625         0.838
    7      37    0.967195    -1.314e-05       0.0625         0.785
    8      45    0.965553    -8.461e-06       0.0156         0.773
    9      53    0.963961    -3.982e-06       0.0156         0.761
   10      62    0.963187    -2.897e-06      0.00781         0.755
.....................................................................
   27     290    0.960254     -3.32e-11     3.05e-05         0.732
   28     307    0.960251    -1.825e-11     3.05e-05         0.732
   29     324    0.960248    -3.291e-12     3.05e-05         0.732
Active inequalities (to within options.ConstraintTolerance = 1e-06):
  lower      upper     ineqlin   ineqnonlin
                                     1

Local minimum possible. Constraints satisfied.

fmincon stopped because the predicted change in the objective
function is less than the value of the function tolerance
and constraints are satisfied to within the value
of the constraint tolerance.

<stopping criteria details>
```

Zwróćmy uwagę na komunikat `Local minimum possible`, który w odróżnieniu od komunikatu `Local minimum found` z poprzedniego przykładu wskazuje na to, że algorytm zakończył działanie niekoniecznie w rozwiązaniu optymalnym. W takim przypadku należy się przyjrzeć przede wszystkim następującym kolumnom na listingu: `f(x)`, `Max constraint` i `First-order optimality Procedure`. Jeżeli zbliżamy się do rozwiązania optymalnego, wartości funkcji celu powinny zmniejszać się w kolejnych iteracjach. Tak jest w istocie, chociaż w kilku ostatnich iteracjach zmiana jest bardzo mała. Ponieważ jednocześnie długości kroku, z którym przemieszczamy się od $\mathbf{x}^{(i)}$ do $\mathbf{x}^{(i+1)}$ (kolumna `Line search steplength`) są bliskie zera, oznacza to, że wykonywanie kolejnych iteracji nie wpływa istotnie na rozwiązanie. Wartości w kolumnie `Max constraint` stanowią miarę przekroczenia ograniczeń – ujemne wartości oznaczają, że żadne z ograniczeń nie zostało przekroczone, natomiast wartość 0 oznacza, że wszystkie ograniczenia są aktywne (w rozważanym zadaniu mamy oczywiście tylko jedno ograniczenie).

Wartości w kolumnie `First-order optimality` Procedure powinny dążyć do zera. W przykładzie brak jest takiej zbieżności, co świadczy o tym, że nie otrzymaliśmy wartości optymalnej.



Rysunek 3.8: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody zbioru ograniczeń aktywnych

□

Przykład 3.9. Korzystając z procedury `fmincon`, wyznaczyć minimum funkcji $f(x_1,x_2) = 2x_1^2 - 1,05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ na zbiorze ograniczeń $D = \{(x_1,x_2) : -1 \leqslant x_1 \leqslant 1, -3 \leqslant x_2 \leqslant -1\}$ metodą obszaru zaufania. Jako punkt startowy przyjąć $\mathbf{x}^{(0)} = [0,-2]^T$.

Rozwiązanie:
Metoda obszaru zaufania zaimplementowana w MATLAB-ie pozwala na uwzględnienie jedynie ograniczeń liniowych. W związku ze zmianą obszaru ograniczeń D w porównaniu z przykładami 3.6–3.8 należy dokonać korekty w pliku **test_fmincon.mlx**. Wywołanie procedury `fmincon` zamieniamy na następujący fragment kodu:

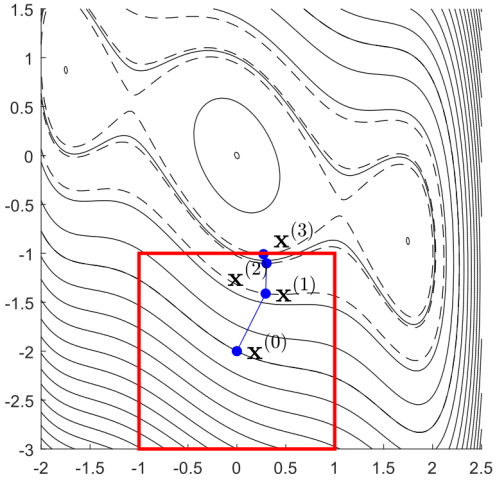
Listing 3.7: Modyfikacja procedury `test_fmincon`

```
xl = [-1, -3];
xu = [1, -1];
[x,f] = fmincon(@f,x0,[],[],[],[],xl,xu,[],opcje);
```

Oczywiście ze względu na zmianę metody należy także w wywołaniu procedury `optimoptions` wstawić parę argumentów:
`'Algorithm','trust-region-reflective'`.
Po wykonaniu obliczeń otrzymamy wyniki:

Listing 3.8: Wynik działania procedury `fmincon` w metodzie obszaru zaufania

```
                          Norm of     First-order
 Iteration  f(x)          step        optimality   CG-iterations
     0            4                                4
     1      1.74312      0.657667          1.04           1
     2      1.05582      0.482499         0.194           1
     3     0.884919      0.295691        0.0147           1
     4     0.870498     0.0904664      0.000214           1
     5     0.870284     0.0111276      4.59e-08           1

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```



Rysunek 3.9: Poziomice funkcji $f(x_1,x_2) = 2x_1^2 - 1{,}05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$ i rozwiązania przybliżone w początkowych iteracjach metody obszaru zaufania

□

3.8. Zadania do samodzielnego opracowania

3.8.1. Programowanie w MATLAB-ie

Zadanie 3.1. Utworzyć procedury `fun`, `gradfun` i `hesfun`, których zadaniami będą odpowiednio: obliczenie wartości funkcji z karą f_K, jej gradientu oraz hesjanu. Procedury powinny działać poprawnie w przypadku, gdy funkcja celu zależy od dwóch zmiennych dla zadanego zbioru ograniczeń D.

Wywołanie procedur:

- `y = fun(met,@f,x,ro,eta)`, gdzie `met` – oznaczenie metody (jeżeli `met="p"`, to metoda przesuwanej funkcji kary, jeżeli `met="w"`, to metoda wewnętrznej funkcji kary), `f` – nazwa procedury obliczającej wartość funkcji celu (bez kary), `x=[x0,y0]` – wektor współrzędnych punktu, w którym ma być obliczona wartość funkcji f_K, `ro,eta` – wektory początkowych współczynników f_K w metodzie przesuwanej funkcji kary lub `ro` – wartość współczynnika $\rho^{(1)}$ w metodzie wewnętrznej funkcji kary, `y` – wartość funkcji f_K;
- `z = gradfun(met,@fp,x,ro,eta)`, gdzie `fp` – nazwa procedury obliczającej gradient funkcji celu (bez kary), `z` – wektor gradientu funkcji f_K;
- `A = hesfun(met,@fd,x,ro,eta)`, gdzie `fd` – nazwa procedury obliczającej hesjan funkcji celu (bez kary), `A` – macierz hesjanu funkcji f_K.

Zadanie 3.2. Utworzyć procedury `przesuw_kara` i `wewn_kara` realizujące odpowiednio metody przesuwanej i wewnętrznej funkcji kary.

Wywołanie procedur:

- `[x,blad] = przesuw_kara(@f,@fp,@fd,x,ro,eta,n_nad)`, gdzie `x` – punkt startowy, `n_nad` – maksymalna ilość iteracji w metodzie funkcji kary, zaś znaczenie parametrów wyjściowych identyczne jak w zadaniu 2.3;
- `[x,blad] = wewn_kara(@f,@fp,@fd,x,ro,n_nad)`.

Wykorzystać procedury `Newton_Rn` z zadania 2.3 oraz `fun`, `gradfun` i `hesfun` z zadania 3.1. Jako podrzędną przyjąć wielowymiarową metodę Newtona. Minimalizacja kierunkowa powinna kończyć się po spełnieniu dwuskośnego testu Goldsteina. Procedury `przesuw_kara` i `wewn_kara` powinny wyświetlać na ekranie wyniki po każdej iteracji.

3.8.2. Sprawozdanie

Podobnie jak w przypadku (ZPNbo), możemy znaleźć w literaturze (np. Hock i Schittkowski, 1981) przykłady problemów testowych, które są wykorzystywane do sprawdzania skuteczności algorytmów optymalizacyjnych w przypadku zadań z ograniczeniami. Poniżej przedstawiamy niektóre z takich problemów:

1.

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2 - x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-2 \;\; 1]^T$$
$$D = \{(x_1,x_2) : x_2 \geqslant 1{,}5\}$$

2.

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2 - x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-2 \;\; 1]^T$$
$$D = \{(x_1,x_2) : x_1x_2 - 1 \geqslant 0,\ x_1 + x_2^2 \geqslant 0,\ x_1 \leqslant 0{,}5\}$$

3.

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2 - x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-2 \;\; 1]^T$$
$$D = \{(x_1,x_2) : x_1 + x_2^2 \geqslant 0,\ x_1^2 + x_2 \geqslant 0,\ -0{,}5 \leqslant x_1 \leqslant 0{,}5,\ x_2 \leqslant 1\}$$

4.

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2 - x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-2 \;\; 1]^T$$
$$D = \{(x_1,x_2) : -x_1 + x_2^2 \geqslant 0,\ x_1^2 - x_2 \geqslant 0,\ -0{,}5 \leqslant x_1 \leqslant 0{,}5,\ x_2 \leqslant 1\}$$

5.

$$f(x_1,x_2) = (1-x_1)^2 + 100(x_2 - x_1^2)^2, \qquad \mathbf{x}^{(0)} = [-2 \;\; 1]^T$$
$$D = \{(x_1,x_2) : x_1 + x_2^2 \geqslant 0,\ x_1^2 + x_2 \geqslant 0,\ x_1^2 + x_2^2 - 1 \geqslant 0,$$
$$-0{,}5 \leqslant x_1 \leqslant 0{,}5\}$$

6.

$$f(x_1,x_2) = x_2 + 10^{-5}(x_2 - x_1)^2, \qquad \mathbf{x}^{(0)} = [10 \;\; 1]^T$$
$$D = \{(x_1,x_2) : x_2 \geqslant 0\}$$

7.

$$f(x_1,x_2) = \frac{1}{3}(x_1+1)^3 + x_2, \qquad \mathbf{x}^{(0)} = [1{,}125 \;\; 0{,}125]^T$$
$$D = \{(x_1,x_2) : x_1 \geqslant 1,\ x_2 \geqslant 0\}$$

8.

$$f(x_1,x_2)=\sin(x_1+x_2)+(x_1-x_2)^2-1,5x_1+2,5x_2, \qquad \mathbf{x}^{(0)}=[0\ \ 0]^T$$
$$D=\{(x_1,x_2):-1,5\leqslant x_1\leqslant 4,\ -3\leqslant x_2\leqslant 3\}$$

9.

$$f(x_1,x_2)=x_1-x_2, \qquad \mathbf{x}^{(0)}=[-10\ \ 10]^T$$
$$D=\{(x_1,x_2):-3x_1^2+2x_1x_2-x_2^2+1\geqslant 0\}$$

10.

$$f(x_1,x_2)=(x_1-5)^2+x_2^2, \qquad \mathbf{x}^{(0)}=[4,9\ \ 0,1]^T$$
$$D=\{(x_1,x_2):-x_1^2+x_2\geqslant 0\}$$

11.

$$f(x_1,x_2)=0,5x_1^2+x_2^2-x_1x_2-7x_1-7x_2, \qquad \mathbf{x}^{(0)}=[0\ \ 0]^T$$
$$D=\{(x_1,x_2):25-4x_1^2-x_2^2\geqslant 0\}$$

12.

$$f(x_1,x_2)=(x_1-2)^2+x_2^2, \qquad \mathbf{x}^{(0)}=[-2\ \ -2]^T$$
$$D=\{(x_1,x_2):(1-x_1)^3-x_2\geqslant 0,\ x_1\geqslant 0,\ x_2\geqslant 0\}$$

13.

$$f(x_1,x_2)=0,01x_1^2+x_2^2, \qquad \mathbf{x}^{(0)}=[2\ \ 2]^T$$
$$D=\{(x_1,x_2):x_1x_2-25\geqslant 0,\ x_1^2+x_2^2-25\geqslant 0,$$
$$2\leqslant x_1\leqslant 50,\ 0\leqslant x_2\leqslant 50\}$$

14.

$$f(x_1,x_2)=0,01x_1^2+x_2^2, \qquad \mathbf{x}^{(0)}=[-1\ \ -1]^T$$
$$D=\{(x_1,x_2):10x_1-x_2-10\geqslant 0,\ 2\leqslant x_1\leqslant 50,\ -50\leqslant x_2\leqslant 50\}$$

15.

$$f(x_1,x_2)=(x_1-10)^3+(x_2-20)^3, \qquad \mathbf{x}^{(0)}=[20,1\ \ 5,84]^T$$
$$D=\{(x_1,x_2):(x_1-5)^2+(x_2-5)^2-100\geqslant 0,$$
$$-(x_1-6)^2-(x_2-5)^2+82,81\geqslant 0,$$
$$13\leqslant x_1\leqslant 100,\ 0\leqslant x_2\leqslant 100\}$$

16.

$$f(x_1,x_2) = (x_1-2)^2 + (x_2-1)^2, \qquad \mathbf{x}^{(0)} = [2\ \ 2]^T$$
$$D = \{(x_1,x_2) : -x_1 - x_2 + 2 \geqslant 0,\ -x_1^2 + x_2 \geqslant 0\}$$

Zadanie 3.3. Dana jest funkcja testowa dwóch zmiennych określona jednym z powyższych wzorów 1–16 wraz ze zbiorem ograniczeń D. Narysować w układzie współrzędnych wykres funkcji celu f z zaznaczonym zbiorem D.

Wyznaczyć przybliżone współrzędne punktów, w których funkcja f osiąga minimum na zbiorze D, stosując przesuwaną i wewnętrzną metodę funkcji kary. W większości podanych przykładów problemów testowych punkt startowy $\mathbf{x}^{(0)} \notin D$, dlatego najpierw jako startowy należy przyjąć dowolny punkt wewnątrz zbioru D (w obu metodach ten sam punkt), a następnie podany w przykładzie punkt $\mathbf{x}^{(0)}$, tylko tym razem w przesuwanej metodzie funkcji kary. W obu metodach należy użyć wielowymiarowej metody Newtona w celu wyznaczenia rozwiązania zadania programowania nieliniowego bez ograniczeń w każdej iteracji głównej metody. Na podstawie wartości funkcji f i f_K trzeba stwierdzić, czy otrzymane rozwiązania znajdują się w zbiorze ograniczeń D, czy poza nim. Porównać szybkość zbieżności poszczególnych metod i sprawdzić, jak wybór punktu startowego wpływa na szybkość zbieżności w przesuwanej metodzie funkcji kary.

Wykonać także obliczenia, korzystając ze standardowej procedury `fmincon`. Zastosować domyślną metodę punktu wewnętrznego i metodę sekwencyjnego programowania kwadratowego. Czy na podstawie otrzymanych wyników można stwierdzić, że te metody są bardziej efektywne od metod funkcji kary?

Rezultaty obliczeń przedstawić w odpowiednich tabelach i na wykresach.
Postać tabeli:

ε	Ilość iteracji	$x_{1_{min}}$	$x_{2_{min}}$	$f(x_{1_{min}}, x_{2_{min}})$	$f_K(x_{1_{min}}, x_{2_{min}})$
10^{-1}					
10^{-2}					
10^{-3}					
10^{-4}					
10^{-5}					

Wykres powinien przedstawiać zależność funkcji oszacowania błędu ε od ilości iteracji n. W jednym układzie współrzędnych należy umieścić cztery wykresy, po jednym dla każdej z metod (na osi pionowej należy zastosować skalę logarytmiczną). Wykresy można narysować w kilku układach współrzędnych, jeżeli przyczyni się to do większej czytelności. Jeżeli to możliwe, należy przyjąć taką maksymalną liczbę iteracji n_{max}, aby $\varepsilon(n_{max}) \leqslant 10^{-10}$.

Rozdział 4

Programowanie liniowe: metoda sympleks

Będziemy rozpatrywać następujące zadanie:

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x} \\ &\mathbf{A}\mathbf{x} = \mathbf{b} \\ &\mathbf{x} \geqslant \mathbf{0} \end{aligned} \tag{4.1}$$

gdzie $\mathbf{x} = [x_1, \ldots, x_n]^T \in \mathbb{R}^n$ – wektor zmiennych decyzyjnych; $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$ – funkcja celu, $f : \mathbb{R}^n \to \mathbb{R}$; $\mathbf{c} = [c_1, \ldots, c_n]^T \in \mathbb{R}^n$ – wektor współczynników funkcji celu; $\mathbf{A}$ – macierz $m \times n$, przy czym $\operatorname{rz}\mathbf{A} = m < n$; $\mathbf{b} = [b_1, \ldots, b_m]^T \in \mathbb{R}^m$; $\mathbf{0} = [0, \ldots, 0]^T \in \mathbb{R}^n$. Zapis $\mathbf{x} \geqslant \mathbf{0}$ oznacza, że $x_1 \geqslant 0, \ldots, x_n \geqslant 0$. Jeżeli ponadto $b_1 \geqslant 0, \ldots, b_m \geqslant 0$, to (4.1) nazywamy zadaniem programowania liniowego (oznaczenie: (ZPL)) w postaci standardowej. Każdy wektor $\mathbf{x}$, który spełnia wszystkie ograniczenia zapisane w (4.1), nazywamy rozwiązaniem dopuszczalnym tego zadania, a zbiór D wszystkich ograniczeń zbiorem rozwiązań dopuszczalnych. Spośród wszystkich rozwiązań dopuszczalnych wybieramy te, dla których funkcja celu osiąga minimum – otrzymujemy w ten sposób zbiór rozwiązań optymalnych D_{opt}, który może zawierać dokładnie jeden element (rozwiązanie zadania (4.1) jest wtedy jednoznaczne), nieskończenie wiele elementów lub może być pusty.

Definicja 4.1. Zadanie programowania liniowego jest nieograniczone $\iff$

$$\inf_D f(\mathbf{x}) = -\infty$$

Zadanie programowania liniowego jest sprzeczne $\iff$ *zbiór rozwiązań dopuszczalnych* $D = \emptyset$.

Uwaga 4.1.

1. Jeżeli (ZPL) jest nieograniczone, to zbiór rozwiązań dopuszczalnych D jest nieograniczony. Odwrotne twierdzenie nie jest prawdziwe.
2. Jeżeli zbiór D jest ograniczony, to (ZPL) ma przynajmniej jedno rozwiązanie optymalne ($D_{opt} \neq \emptyset$).
3. Zbiór rozwiązań optymalnych $D_{opt} = \emptyset \iff$ (ZPL) jest nieograniczone lub jest sprzeczne.

Funkcja celu, tak jak wszystkie ograniczenia występujące w (ZPL), jest liniowa. Zadanie (4.1) możemy zapisać także w postaci rozwiniętej:

$$\min(c_1x_1 + \ldots + c_nx_n)$$
$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots &\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \\ x_1 \geqslant 0, \ldots, x_n \geqslant 0 \end{cases} \tag{4.2}$$

Klasyczną i powszechnie stosowaną metodą rozwiązywania (ZPL) jest metoda sympleks. Istnieje wiele wariantów tej metody, natomiast w dalszych podrozdziałach omówimy dwa: dwufazową oraz zrewidowaną metodę sympleks. Oba omawiane warianty wymagają, aby (ZPL) miało tzw. postać kanoniczną, tzn. aby wśród kolumn macierzy **A** można było znaleźć m różnych kolummn jednostkowych. Postać kanoniczną można uzyskać z postaci standardowej, wprowadzając do (ZPL) tzw. zmienne sztuczne. Najpierw jednak trzeba zwykle sprowadzić (ZPL) do postaci standardowej, stosując tzw. zmienne bilansujące. W kolejnym podrozdziale zostały przedstawione różne przypadki, z którymi możemy mieć do czynienia, jeżeli (ZPL) nie jest w postaci standardowej.

4.1. Sprowadzanie (ZPL) do postaci standardowej

Przypadek 1:

Jeżeli w (ZPL) istnieją ograniczenia nierównościowe postaci:

$$\begin{array}{r} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leqslant b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \leqslant \ldots \end{array}$$

to wprowadzamy nowe zmienne bilansujące $z_1, z_2, \ldots$, co do których zakładamy, że są nieujemne ($z_1 \geqslant 0, z_2 \geqslant 0, \ldots$), aby zamienić nierówności na równania:

$$\begin{array}{r} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + z_1 = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + z_2 = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots = \ldots \end{array}$$

Przykład 4.1. Sprowadzić poniższe (ZPL) do postaci standardowej:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4 & =7 \\ x_2-4x_4 & \leqslant 3 \\ x_1+5x_2-4x_3 & \leqslant 2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0,\ x_4\geqslant 0 & \end{cases}$$

Rozwiązanie:
Wprowadzamy nowe zmienne $z_1\geqslant 0, z_2\geqslant 0$ i zapisujemy (ZPL) w postaci:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4=7 \\ x_2-4x_4+z_1=3 \\ x_1+5x_2-4x_3+z_2=2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0,\ x_4\geqslant 0,\ z_1\geqslant 0,\ z_2\geqslant 0 \end{cases}$$

□

Przypadek 2:

Podobnie jeżeli w (ZPL) istnieją ograniczenia nierównościowe postaci:

$$\begin{array}{c} a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n\geqslant b_1 \\ a_{21}x_1+a_{22}x_2+\cdots+a_{2n}x_n\geqslant b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\geqslant .. \end{array}$$

wprowadzamy nowe zmienne bilansujące $z_1, z_2, \ldots$, co do których zakładamy, że są nieujemne ($z_1\geqslant 0, z_2\geqslant 0, \ldots$), aby zamienić nierówności na równania:

$$\begin{array}{c} a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n-z_1=b_1 \\ a_{21}x_1+a_{22}x_2+\cdots+a_{2n}x_n-z_2=b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots= .. \end{array}$$

Przykład 4.2. Sprowadzić poniższe (ZPL) do postaci standardowej:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4 & \geqslant 7 \\ x_2-4x_4 & =3 \\ x_1+5x_2-4x_3 & \leqslant 2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0,\ x_4\geqslant 0 & \end{cases}$$

Rozwiązanie:
Wprowadzamy nowe zmienne $z_1\geqslant 0, z_2\geqslant 0$ i zapisujemy (ZPL) w postaci:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4-z_1=7 \\ x_2-4x_4 \qquad\qquad\quad =3 \\ x_1+5x_2-4x_3+z_2 \quad =2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0,\ x_4\geqslant 0,\ z_1\geqslant 0,\ z_2\geqslant 0 \end{cases}$$

□

Przypadek 3:

Jeżeli w (ZPL) istnieją tzw. zmienne swobodne, tzn. nie ma któregoś z ograniczeń typu $x_1\geqslant 0, x_2\geqslant 0,\ldots$, to w celu sprowadzenia (ZPL) do postaci standardowej możemy postąpić na dwa sposoby.

Sposób 1.
Dla każdej zmiennej swobodnej x_i wprowadzamy dwie nowe zmienne $u_i\geqslant 0, v_i\geqslant 0$, przyjmujemy

$$x_i=u_i-v_i$$

i zastępujemy x_i tym podstawieniem w funkcji celu i we wszystkich ograniczeniach.

Przykład 4.3. Sprowadzić poniższe (ZPL) do postaci standardowej:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4 = \ 7 \\ x_2-4x_4 \qquad\qquad = \ 3 \\ x_1+5x_2-4x_3 \qquad \leqslant -2 \\ x_1\geqslant 0,\ x_4\geqslant 0 \end{cases}$$

Rozwiązanie:
Ponieważ w trzecim ograniczeniu występuje nierówność, wprowadzamy nową zmienną $z_1\geqslant 0$ i zapisujemy to ograniczenie w postaci:

$$x_1+5x_2-4x_3+z_1=-2$$

Po prawej stronie jest liczba ujemna (-2), zatem trzeba jeszcze pomnożyć to ograniczenie przez -1. Otrzymujemy wtedy:

$$-x_1-5x_2+4x_3-z_1=2$$

Ponadto w rozważanym (ZPL) brak jest ograniczeń $x_2\geqslant 0, x_3\geqslant 0$.
Możemy zatem wprowadzić dodatkowe zmienne $u_2\geqslant 0,\ v_2\geqslant 0,\ u_3\geqslant 0,\ v_3\geqslant 0$, takie, że:

$$x_2=u_2-v_2 \quad \text{i} \quad x_3=u_3-v_3$$

Następnie zastępujemy x_2 i x_3 w funkcji celu i we wszystkich ograniczeniach równościowych. Ostatecznie otrzymujemy:

$$\min(2x_1+3u_2-3v_2-4x_4)$$
$$\begin{cases} x_1+5u_2-5v_2-3u_3+3v_3+x_4 = 7 \\ u_2-v_2-4x_4 = 3 \\ -x_1-5u_2+5v_2+4u_3-4v_3-z_1 = 2 \\ x_1\geqslant 0,\ x_4\geqslant 0,\ u_2\geqslant 0,\ v_2\geqslant 0,\ u_3\geqslant 0,\ v_3\geqslant 0,\ z_1\geqslant 0 \end{cases}$$

□

Sposób 2.
Jeżeli np. x_1 jest zmienną swobodną, to z dowolnego ograniczenia równościowego, np.

$$a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n=b_1$$

wyznaczamy

$$x_1=\tfrac{1}{a_{11}}(b_1-a_{12}x_2-\ldots-a_{1n}x_n)$$

i wstawiamy do wszystkich takich ograniczeń, które w postaci standardowej (ZPL) mają być przedstawione w postaci równań.

Przykład 4.4. Sprowadzić poniższe (ZPL) do postaci standardowej:

$$\min(2x_1+3x_2-4x_4)$$
$$\begin{cases} x_1+5x_2-3x_3+x_4 = 7 \\ x_2-4x_4 = 3 \\ x_1+5x_2-4x_3 = 2 \\ x_1\geqslant 0,\ x_4\geqslant 0 \end{cases}$$

Rozwiązanie:
Z drugiego równania otrzymujemy:

$$x_2=4x_4+3 \tag{4.3}$$

a z trzeciego:

$$\begin{aligned} x_3 &= 0,25(x_1+5x_2-2)=0,25(x_1+20x_4+13)= \\ &= 0,25x_1+5x_4+3,25 \end{aligned} \tag{4.4}$$

Wstawiając (4.3) i (4.4) do funkcji celu i pierwszego ograniczenia równościowego, otrzymamy (ZPL) w postaci standardowej:

$$\min(2x_1 + 8x_4)$$
$$\begin{cases} 0,25x_1 + 6x_4 = 1,75 \\ x_1 \geqslant 0,\ x_4 \geqslant 0 \end{cases}$$

Uwaga 4.2. W rzeczywistości funkcja celu po wyeliminowaniu zmiennych x_2 i x_3 ma postać:

$$f(x_1, x_4) = 2x_1 + 8x_4 + 9$$

jednak dodanie stałej do funkcji celu nie wpływa na wartości zmiennych decyzyjnych w rozwiązaniu optymalnym. Wystarczy zatem rozwiązać powyższe (ZPL) w postaci standardowej z funkcją celu $2x_1 + 8x_4$, a następnie dodać 9 do otrzymanej wartości tej funkcji. □

Przypadek 4:

Jeżeli w (ZPL) jest max zamiast min, to korzystamy ze wzoru:

$$\max f(x_1, x_2, ..., x_n) = -\min(-f(x_1, x_2, ..., x_n))$$

Przykład 4.5. Sprowadzić poniższe (ZPL) do postaci standardowej:

$$\max(2x_1 + 8x_2)$$
$$\begin{cases} x_1 + 6x_2 = 1 \\ x_1 \geqslant 0,\ x_2 \geqslant 0 \end{cases}$$

Rozwiązanie:
Otrzymujemy (ZPL) postaci:

$$\min(-2x_1 - 8x_2)$$
$$\begin{cases} x_1 + 6x_2 = 1 \\ x_1 \geqslant 0,\ x_2 \geqslant 0 \end{cases}$$

Po jego rozwiązaniu należy pamiętać o tym, że minimalna wartość funkcji celu w ostatnim sformułowaniu zadania jest równa **minus** wartości maksymalnej z pierwotnej wersji zadania. □

4.2. Dwufazowa metoda sympleks

W tym wariancie metody sympleks przekształcamy najpierw pierwotne (ZPL) zapisane w postaci standardowej w zadanie pomocnicze w postaci kanonicznej. Pierwsza faza polega na rozwiązaniu zadania pomocniczego. Otrzymujemy w jej końcowym

etapie zadanie równoważne pierwotnemu, ale zapisane także w postaci kanonicznej. To nowe (ZPL) jest rozwiązywane w drugiej fazie. Otrzymany w tej fazie zbiór rozwiązań optymalnych jest jednocześnie zbiorem rozwiązań optymalnych zadania pierwotnego.

Przykład 4.6. Stosując dwufazową metodę sympleks, rozwiązać (ZPL):

$$\min(2x_1+3x_2)$$
$$\begin{cases} x_1+4x_2+x_3=1 \\ x_1+x_2+2x_3=2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0 \end{cases}$$

Rozwiązanie:

I faza:
Jest to zadanie w postaci standardowej. W pierwszej fazie wprowadzamy sztuczne zmienne $y_1\geqslant 0, y_2\geqslant 0$, dodając je do lewych stron odpowiednich ograniczeń równościowych, tak, aby sprowadzić zadanie do postaci kanonicznej. Formułujemy następnie poniższe pomocnicze (ZPL), w którym minimalizujemy sumę zmiennych sztucznych:

$$\min(y_1+y_2)$$
$$\begin{cases} x_1+4x_2+x_3+y_1=1 \\ x_1+x_2+2x_3+y_2=2 \\ x_1\geqslant 0,\ x_2\geqslant 0,\ x_3\geqslant 0,\ y_1\geqslant 0,\ y_2\geqslant 0 \end{cases}$$

Tworzymy początkową tabelę metody sympleks, w której wpisujemy współczynniki ograniczeń i funkcji celu:

	x_1	x_2	x_3	y_1	y_2	b
Pierwsze ograniczenie (wiersz w_1):	1	4	1	1	0	1
Drugie ograniczenie (wiersz w_2):	1	1	2	0	1	2
Funkcja celu (wiersz w):	0	0	0	1	1	0

Liczby zaznaczone niebieskim kolorem tworzą kolumny jednostkowe. Zmieniamy wiersz funkcji celu tak, aby pod kolumnami jednostkowymi otrzymać same zera (w miejscu liczb czerwonych), wykonując działanie na wierszach: $w-w_1-w_2$. Otrzymamy wtedy:

x_1	x_2	x_3	y_1	y_2	b
1	4	1	1	0	1
1	1	2	0	1	2
-2	-5	-3	0	0	-3

Powyższej tabeli odpowiada dopuszczalne rozwiązanie bazowe:

$$\mathbf{x} = [x_1, x_2, x_3, y_1, y_2]^T = [0,0,0,1,2]^T$$

Pierwsze trzy kolumny nie są jednostkowe, dlatego przyjmujemy $x_1 = x_2 = x_3 = 0$, natomiast pozostałe dwie są jednostkowe – stanowią bazę, zatem wartości zmiennych $y_1 = 1$ i $y_2 = 2$ odczytujemy z ostatniej kolumny. Wartość funkcji celu dla powyższego dopuszczalnego rozwiązania bazowego $\mathbf{x}$ wynosi $f(\mathbf{x}) = 3$ (niebieska liczba z przeciwnym znakiem).

Wybieramy najmniejszą ujemną liczbę w ostatnim wierszu (liczba czerwona) – przy tym wyborze nie bierzemy pod uwagę liczby niebieskiej. Ponieważ wybrana w ten sposób liczba jest w drugiej kolumnie, wprowadzamy tę kolumnę do bazy (chcemy uczynić ją kolumną jednostkową).

Obliczamy ilorazy liczb z ostatniej kolumny (zielone) i odpowiednich liczb z kolumny wprowadzanej do bazy (drugiej – liczby zielone):

$$\frac{1}{4}, \frac{2}{1} = 2$$

W obliczeniach bierzemy jednak pod uwagę tylko liczby dodatnie z kolumny wprowadzanej do bazy. Wybieramy z tych ilorazów najmniejszy, czyli $\frac{1}{4}$. Ponieważ powstał on przez dzielenie liczb w pierwszym wierszu, więc wyrzucamy z bazy tę kolumnę, która ma jedynkę właśnie w pierwszym wierszu, czyli kolumnę związaną ze zmienną y_1.

Teraz chcemy, aby druga kolumna stała się kolumną jednostkową. Mnożymy zatem pierwszy wiersz przez $\frac{1}{4}$, aby liczba w ramce stała się równa 1 (działanie: $\frac{1}{4}w_1$):

x_1	x_2	x_3	y_1	y_2	b
$\frac{1}{4}$	1	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$
1	1	2	0	1	2
-2	-5	-3	0	0	-3

Otrzymujemy teraz zera w drugiej kolumnie – tam, gdzie są liczby czerwone (działania: $w_2 - w_1$, $w + 5w_1$):

x_1	x_2	x_3	y_1	y_2	b
$\frac{1}{4}$	1	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$
$\frac{3}{4}$	0	$\boxed{\frac{7}{4}}$	$-\frac{1}{4}$	1	$\frac{7}{4}$
$-\frac{3}{4}$	0	$-\frac{7}{4}$	$\frac{5}{4}$	0	$-\frac{7}{4}$

Powyższej tabeli odpowiada dopuszczalne rozwiązanie bazowe

$$\mathbf{x} = [x_1, x_2, x_3, y_1, y_2]^T = \left[0, \frac{1}{4}, 0, 0, \frac{7}{4}\right]^T$$

Kolumny pierwsza, trzecia i czwarta nie są jednostkowe, dlatego przyjmujemy $x_1 = x_3 = y_1 = 0$. Pozostałe dwie są jednostkowe – stanowią bazę – zatem wartości zmiennych $x_2 = \frac{1}{4}$ i $y_2 = \frac{7}{4}$ odczytujemy z ostatniej kolumny. Wartość funkcji celu dla powyższego dopuszczalnego rozwiązania bazowego $\mathbf{x}$ wynosi $f(\mathbf{x}) = \frac{7}{4}$ (niebieska liczba z przeciwnym znakiem).

Wybieramy najmniejszą ujemną liczbę w ostatnim wierszu (liczba czerwona). Ponieważ liczba ta jest w trzeciej kolumnie, wprowadzamy tę kolumnę do bazy.

Obliczamy ilorazy liczb z ostatniej kolumny (zielone) i odpowiednich liczb z kolumny wprowadzanej do bazy (trzeciej – liczby zielone):

$$\frac{1}{4} : \frac{1}{4} = 1, \quad \frac{7}{4} : \frac{7}{4} = 1$$

Ilorazy są identyczne, więc możemy wyrzucić z bazy kolumnę drugą (związaną ze zmienną x_2) lub piątą (związaną ze zmienną y_2). Zależy nam jednak na tym, aby w bazie nie było kolumn związanych ze zmiennymi sztucznymi, zatem wyrzucamy z bazy kolumnę piątą.

Mnożymy drugi wiersz przez $\frac{4}{7}$, aby liczba w ramce stała się równa 1 (działanie: $\frac{4}{7}w_2$):

x_1	x_2	x_3	y_1	y_2	b
$\frac{1}{4}$	1	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$
$\frac{3}{7}$	0	1	$-\frac{1}{7}$	$\frac{4}{7}$	1
$-\frac{3}{4}$	0	$-\frac{7}{4}$	$\frac{5}{4}$	0	$-\frac{7}{4}$

Otrzymujemy zera w trzeciej kolumnie tam, gdzie są liczby czerwone (działania: $w_1 - \frac{1}{4}w_2, w + \frac{7}{4}w_2$):

x_1	x_2	x_3	y_1	y_2	b
$\frac{1}{7}$	1	0	$\frac{2}{7}$	$-\frac{1}{7}$	0
$\frac{3}{7}$	0	1	$-\frac{1}{7}$	$\frac{4}{7}$	1
0	0	0	1	1	0

Ponieważ w ostatnim wierszu wszystkie liczby są nieujemne, kończymy obliczenia I fazy. Jednocześnie otrzymaliśmy bazę niezwiązaną ze zmiennymi sztucznymi, zatem możemy przejść do fazy II.

II faza:
Teraz bierzemy pod uwagę tylko te kolumny z ostatniej tabeli, które nie są związane ze zmiennymi sztucznymi. Powracamy do pierwotnej funkcji celu, tzn. $f(\mathbf{x}) = 2x_1 + 3x_2$:

x_1	x_2	x_3	b
$\frac{1}{7}$	1	0	0
$\frac{3}{7}$	0	1	1
2	3	0	0

Dalej postępujemy analogicznie, jak w I fazie, tzn. na początku zmieniamy wiersz funkcji celu tak, aby pod kolumnami jednostkowymi otrzymać same zera (w miejscu liczb czerwonych), wykonując działanie na wierszach: $w - 3w_1$. Otrzymujemy:

x_1	x_2	x_3	b
$\frac{1}{7}$	1	0	0
$\frac{3}{7}$	0	1	1
$\frac{11}{7}$	0	0	0

Kończymy obliczenia, gdyż wszystkie liczby w ostatnim wierszu są nieujemne. Otrzymaliśmy rozwiązanie optymalne pierwotnego (ZPL) w punkcie $\mathbf{x} = [0,0,1]^T$. Wartość pierwotnej funkcji celu dla powyższego dopuszczalnego rozwiązania bazowego $\mathbf{x}$ wynosi $f(\mathbf{x}) = 0$ (niebieska liczba z przeciwnym znakiem). □

W powyższym przykładzie otrzymaliśmy rozwiązanie optymalne. Metoda dwufazowa pozwala wykryć również przypadek nieograniczoności lub sprzeczności (ZPL).

Twierdzenie 4.2. *1. Jeżeli w tabeli sympleksowej wszystkie liczby w kolumnie wprowadzanej do bazy są ujemne, pierwotne (ZPL) jest nieograniczone.*
2. Pierwotne (ZPL) jest sprzeczne $\iff$ wartość funkcji celu w rozwiązaniu optymalnym zadania I fazy jest różna od zera.

Przykład 4.7. (zadanie nieograniczone) Stosując dwufazową metodę sympleks, rozwiązać (ZPL):

$$\min(-x_2 - x_3)$$
$$\begin{cases} x_1 - x_2 + x_3 = 1 \\ x_2 - 2x_3 + x_4 = 2 \\ x_1 \geqslant 0,\ x_2 \geqslant 0,\ x_3 \geqslant 0,\ x_4 \geqslant 0 \end{cases}$$

Rozwiązanie:
Zauważmy, że powyższe (ZPL) jest już w postaci kanonicznej ze zmiennymi bazowymi x_1 i x_4. Przechodzimy zatem od razu do II fazy:

x_1	x_2	x_3	x_4	b
1	−1	1	0	1
0	1	−2	1	2
0	−1	−1	0	0

Odczytujemy dopuszczalne rozwiązanie bazowe:

$$\mathbf{x} = [x_1, x_2, x_3, x_4]^T = [1,0,0,2]^T$$

dla którego wartość funkcji celu $f(\mathbf{x}) = 0$ i zgodnie z algorytmem metody sympleks wprowadzamy do bazy kolumnę drugą i usuwamy czwartą (moglibyśmy wprowadzić trzecią i usunąć pierwszą, co jednak prowadziłoby do tego samego końcowego wniosku):

x_1	x_2	x_3	x_4	b
1	0	−1	1	3
0	1	−2	1	2
0	0	−3	1	2

Rozwiązanie dopuszczalne odczytane z ostatniej tabeli to:

$$\mathbf{x} = [x_1, x_2, x_3, x_4]^T = [3,2,0,0]^T$$

z wartością funkcji celu $f(\mathbf{x}) = -2$. W dalszym ciągu nie jest to jednak rozwiązanie optymalne, gdyż w ostatnim wierszu mamy liczbę ujemną -3. Powinniśmy teraz wprowadzić do bazy kolumnę trzecią, ale wszystkie liczby w tej kolumnie są ujemne, więc nie możemy obliczyć odpowiednich ilorazów decydujących o usunięciu z bazy innej kolumny. Na podstawie punktu 1. Twierdzenia 4.2 uznajemy, że rozwiązywane zadanie jest nieograniczone. □

Przykład 4.8. (zadanie sprzeczne) Stosując dwufazową metodę sympleks, rozwiązać (ZPL):

$$\min(x_1 + 2x_2)$$
$$\begin{cases} x_1 - 2x_2 - x_3 = 3 \\ -4x_1 - x_2 + x_3 = 2 \\ x_1 \geqslant 0,\ x_2 \geqslant 0,\ x_3 \geqslant 0 \end{cases}$$

Rozwiązanie:

I faza:
Odpowiednie zadanie pomocnicze ma postać:

$$\min(y_1 + y_2)$$
$$\begin{cases} x_1 - 2x_2 - x_3 + y_1 = 3 \\ -4x_1 - x_2 + x_3 + y_2 = 2 \\ x_1 \geqslant 0,\ x_2 \geqslant 0,\ x_3 \geqslant 0,\ y_1 \geqslant 0,\ y_2 \geqslant 0 \end{cases}$$

Zapiszmy tabelę sympleksową:

x_1	x_2	x_3	y_1	y_2	b
1	−2	−1	1	0	3
−4	−1	1	0	1	2
0	0	0	1	1	0

Po wyzerowaniu elementów w ostatnim wierszu pod kolumnami y_1 i y_2 (działanie $w - w_1 - w_2$) otrzymujemy:

x_1	x_2	x_3	y_1	y_2	b
1	−2	−1	1	0	3
−4	−1	1	0	1	2
3	3	0	0	0	-5

Otrzymaliśmy od razu rozwiązanie optymalne zadania I fazy, ale wartość funkcji celu $y_1 + y_2$ jest równa 5, a więc różna od zera. Na podstawie punktu 2. Twierdzenia 4.2 uznajemy zatem, że pierwotne zadanie jest sprzeczne. □

4.3. Zrewidowana metoda sympleks

Zadanie programowania liniowego w postaci kanonicznej można rozwiązać, stosując także tzw. zrewidowaną metodę sympleks. Sposób wykonywania obliczeń w tym wariancie metody został przedstawiony w poniższym przykładzie 4.9. Okazuje się, że nie wszystkie operacje arytmetyczne wykonywane w dwufazowej metodzie sympleks są konieczne do poprawnego jej działania. W szczególności nie trzeba w każdej iteracji przekształcać całej tablicy sympleksowej, wystarczy wykonać obliczenia na kolumnie wprowadzanej do bazy. Stanowi to istotną korzyść metody zrewidowanej ze względu na ilość obliczeń potrzebnych do uzyskania rozwiązania optymalnego.

Przykład 4.9. Stosując zrewidowaną metodę sympleks, rozwiązać zadanie:

$$\min(x_4 + x_5)$$
$$\begin{cases} x_1 + 4x_2 + x_3 + x_4 = 1 \\ x_1 + x_2 + 2x_3 + x_5 = 2 \\ x_1, ..., x_5 \geqslant 0 \end{cases}$$

Rozwiązanie:
Zapisujemy współczynniki ograniczeń w macierzy:

$$\mathbf{A} = \overset{\begin{matrix} \text{I} & \text{II} & \text{III} & \text{IV} & \text{V} \end{matrix}}{\left[\begin{array}{|ccc|cc|} 1 & 4 & 1 & 1 & 0 \\ 1 & 1 & 2 & 0 & 1 \end{array}\right]} \quad \begin{matrix} \underbrace{\qquad\qquad} \\ \mathbf{A}_N \end{matrix} \quad \begin{matrix} \underbrace{\qquad} \\ \mathbf{B} \end{matrix}$$

i dzielimy ją na dwie części: macierz niebazową $\mathbf{A}_N$ i macierz bazową jednostkową $\mathbf{B}$. Nad macierzą zostały zapisane numery zmiennych odpowiadających poszczególnym jej kolumnom.

Podziałowi macierzy $\mathbf{A}$ odpowiada podział wektora rozwiązania dopuszczalnego powyższego zadania na część niebazową $\mathbf{x}_N$ i bazową $\mathbf{x}_B$:

$$\mathbf{x} = \overset{\begin{matrix} \text{I} & \text{II} & \text{III} & \text{IV} & \text{V} \end{matrix}}{\left[\begin{array}{|ccc|cc|} x_1 & x_2 & x_3 & x_4 & x_5 \end{array}\right]}^T \quad \begin{matrix} \underbrace{\qquad\qquad} \\ \mathbf{x}_N \end{matrix} \quad \begin{matrix} \underbrace{\qquad} \\ \mathbf{x}_B \end{matrix}$$

oraz podział wektora współczynników funkcji celu na część niebazową $\mathbf{c}_N$ i bazową $\mathbf{c}_B$:

$$\mathbf{c} = \overset{\begin{matrix} \text{I} & \text{II} & \text{III} & \text{IV} & \text{V} \end{matrix}}{\left[\begin{array}{|ccc|cc|} 0 & 0 & 0 & 1 & 1 \end{array}\right]}^T \quad \begin{matrix} \underbrace{\qquad\qquad} \\ \mathbf{c}_N \end{matrix} \quad \begin{matrix} \underbrace{\qquad} \\ \mathbf{c}_B \end{matrix}$$

Iteracja nr 1:
Przyjmujemy $\mathbf{x}_N = [0\ 0\ 0]^T$ i obliczamy:

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Wartość funkcji celu:

$$f(\mathbf{x}) = \mathbf{c}_B{}^T \mathbf{x}_B = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = 3$$

Aby zdecydować, którą kolumnę wprowadzić do bazy, obliczamy:

$$\mathbf{p}_N^T = \mathbf{c}_N{}^T - \mathbf{c}_B{}^T \mathbf{B}^{-1}\mathbf{A}_N = [0\,0\,0] - [1\;1] \begin{bmatrix} 1 & 4 & 1 \\ 1 & 1 & 2 \end{bmatrix} = [-2 \overset{\text{I}}{\quad} \boxed{-5} \overset{\text{II}}{\quad} -3 \overset{\text{III}}{\quad}]$$

Wprowadzamy kolumnę II (związaną ze zmienną x_2), gdyż -5 jest najmniejszą liczbą ujemną w wektorze $\mathbf{p}_N$. Gdyby w wektorze $\mathbf{p}_N$ nie było liczb ujemnych, to otrzymalibyśmy rozwiązanie optymalne.

Oznaczmy przez $\mathbf{a}_J$ kolumnę wprowadzaną do bazy. Wyznaczamy wektor:

$$\mathbf{d} = \mathbf{B}^{-1}\mathbf{a}_J = \mathbf{a}_J = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$$

Aby zdecydować, którą kolumnę usunąć z bazy, obliczamy ilorazy współrzędnych wektora $\mathbf{x}_B$ przez odpowiednie współrzędne wektora $\mathbf{d}$. Otrzymujemy:

$$\frac{1}{4}, \quad \frac{2}{1} = 2$$

Wybieramy najmniejszy nieujemny iloraz, czyli $\frac{1}{4}$ – otrzymaliśmy go, dzieląc liczby odpowiadające kolumnie IV, zatem usuwamy ją z bazy. Zmiana bazy w zrewidowanej metodzie sympleks oznacza zamianę miejscami w macierzy $\mathbf{A}$ oraz w wektorach $\mathbf{x}^T$ i $\mathbf{c}^T$ kolumny wprowadzanej i usuwanej. Dostajemy zatem:

$$A = \Big[\overset{\text{I}}{\boxed{\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 2 \end{matrix}}} \overset{\text{IV}}{\overset{\text{III}}{\boxed{\begin{matrix} 4 & 0 \\ 1 & 1 \end{matrix}}}} \overset{\text{II}}{\quad} \overset{\text{V}}{\quad} \Big] \qquad \begin{matrix} \\ \mathbf{A}_N \end{matrix} \quad \begin{matrix} \\ \mathbf{B} \end{matrix}$$

oraz

$$\mathbf{x} = \big[\overset{\text{I}}{\quad} \overset{\text{IV}}{\overset{\text{III}}{\boxed{x_1\; x_4\; x_3}}} \overset{\text{II}}{\boxed{x_2\; x_5}} \overset{\text{V}}{\quad} \big]^T \qquad \begin{matrix} \\ \mathbf{x}_N \end{matrix} \quad \begin{matrix} \\ \mathbf{x}_B \end{matrix}$$

a także

$$\mathbf{c} = \big[\overset{\text{I}}{\boxed{0\;1\;0}} \overset{\text{IV}}{\overset{\text{III}}{\quad}} \overset{\text{II}}{\boxed{0\;1}} \overset{\text{V}}{\quad} \big]^T \qquad \begin{matrix} \\ \mathbf{c}_N \end{matrix} \quad \begin{matrix} \\ \mathbf{c}_B \end{matrix}$$

Iteracja nr 2:
Znowu przyjmujemy $\mathbf{x}_N = [0\;0\;0]^T$ i obliczamy:

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$$

Teraz $\mathbf{B}$ nie jest już macierzą jednostkową. Ponieważ

$$\mathbf{B}^{-1} = \begin{bmatrix} \frac{1}{4} & 0 \\ -\frac{1}{4} & 1 \end{bmatrix}$$

więc

$$\mathbf{x}_B = \begin{bmatrix} \frac{1}{4} & 0 \\ -\frac{1}{4} & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ \frac{7}{4} \end{bmatrix}$$

Wartość funkcji celu:

$$f(\mathbf{x}) = \mathbf{c}_B{}^T \mathbf{x}_B = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{4} \\ \frac{7}{4} \end{bmatrix} = \frac{7}{4}$$

Aby zdecydować, którą kolumnę wprowadzić do bazy, obliczamy:

$$\mathbf{p}_N^T = \mathbf{c}_N{}^T - \mathbf{c}_B{}^T \mathbf{B}^{-1}\mathbf{A}_N = [0\ 1\ 0] - [0\ 1] \begin{bmatrix} \frac{1}{4} & 0 \\ -\frac{1}{4} & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 2 \end{bmatrix} =$$

$$\overset{\text{I}\quad \text{IV}\quad \text{III}}{=} \begin{bmatrix} -\frac{3}{4} & \frac{5}{4} & \boxed{-\frac{7}{4}} \end{bmatrix}$$

Wprowadzamy kolumnę III (związaną ze zmienną x_3), gdyż $-\frac{7}{4}$ jest najmniejszą liczbą ujemną w wektorze $\mathbf{p}_N$.

Wyznaczamy wektor:

$$\mathbf{d} = \mathbf{B}^{-1}\mathbf{a}_J = \begin{bmatrix} \frac{1}{4} & 0 \\ -\frac{1}{4} & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ \frac{7}{4} \end{bmatrix}$$

Obliczamy ilorazy współrzędnych wektora $\mathbf{x}_B$ i odpowiednich współrzędnych wektora $\mathbf{d}$. Otrzymujemy:

$$\frac{1}{4} : \frac{1}{4} = 1, \quad \frac{7}{4} : \frac{7}{4} = 1$$

Ponieważ oba są nieujemne i równe, możemy wyrzucić z bazy kolumnę II lub V, ale naszym celem jest przeniesienie kolumn jednostkowych do macierzy $\mathbf{A}_N$, usuwamy zatem kolumnę V.

Po zmianie bazy:

$$\mathbf{A} = \overset{\text{I IV V II III}}{\left[\begin{array}{|ccc|cc|} \hline 1 & 1 & 0 & 4 & 1 \\ 1 & 0 & 1 & 1 & 2 \\ \hline \end{array}\right]} \quad \begin{matrix} \\ \\ \mathbf{A}_N \end{matrix} \quad \begin{matrix} \\ \\ \mathbf{B} \end{matrix}$$

oraz

$$\mathbf{x} = \overset{\text{I IV V II III}}{\left[\begin{array}{|ccc|cc|} \hline x_1 & x_4 & x_5 & x_2 & x_3 \\ \hline \end{array}\right]}^T \quad \begin{matrix} \\ \mathbf{x}_N \end{matrix} \quad \begin{matrix} \\ \mathbf{x}_B \end{matrix}$$

a także

$$\mathbf{c} = \overset{\text{I IV V II III}}{\left[\begin{array}{|ccc|cc|} \hline 0 & 1 & 1 & 0 & 0 \\ \hline \end{array}\right]}^T \quad \begin{matrix} \\ \mathbf{c}_N \end{matrix} \quad \begin{matrix} \\ \mathbf{c}_B \end{matrix}$$

Iteracja nr 3:
Ponownie $\mathbf{x}_N = [0\ 0\ 0]^T$. Obliczamy:

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$$

Ponieważ

$$\mathbf{B}^{-1} = \begin{bmatrix} \frac{2}{7} & -\frac{1}{7} \\ -\frac{1}{7} & \frac{4}{7} \end{bmatrix}$$

więc

$$\mathbf{x}_B = \begin{bmatrix} \frac{2}{7} & -\frac{1}{7} \\ -\frac{1}{7} & \frac{4}{7} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Wartość funkcji celu wynosi:

$$f(\mathbf{x}) = \mathbf{c}_B{}^T \mathbf{x}_B = \begin{bmatrix} 0\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0$$

Obliczamy:

$$\mathbf{p}_N^T = \mathbf{c}_N{}^T - \mathbf{c}_B{}^T \mathbf{B}^{-1}\mathbf{A}_N = \underset{\text{I IV V}}{[0\ 1\ 1]} - [0\ 0] \begin{bmatrix} \frac{2}{7} & -\frac{1}{7} \\ -\frac{1}{7} & \frac{4}{7} \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} =$$
$$= [0\ \ 1\ \ 1]$$

Ponieważ wszystkie współrzędne wektora $\mathbf{p}_N$ są nieujemne, kończymy obliczenia – otrzymaliśmy rozwiązanie optymalne:

$$\mathbf{x}^\star = \big[\ \overbrace{0\,0\,0}^{\mathbf{x}_N}\ \overbrace{0\,1}^{\mathbf{x}_B}\ \big]^T \qquad \begin{matrix} \text{I} & \text{IV} & \text{V} & \text{II} & \text{III} \end{matrix}$$

albo po uporządkowaniu numerów zmiennych:

$$\mathbf{x}^\star = \big[\,0\,0\,1\,0\,0\,\big]^T \qquad \begin{matrix} \text{I} & \text{II} & \text{III} & \text{IV} & \text{V} \end{matrix}$$

□

4.4. Dualizm w programowaniu liniowym – równoważność zadania pierwotnego i dualnego

Każdemu pierwotnemu (ZPL) odpowiada pewne inne zadanie programowania liniowego (ZD), które nazywane jest zadaniem dualnym do (ZPL). W przypadku, gdy (ZPL) ma postać:

$$\begin{aligned} &\min(c_1x_1+\cdots+c_nx_n)\\ &\begin{cases} a_{11}x_1+\cdots+a_{1n}x_n &\geqslant b_1\\ \quad\cdots\cdots\cdots\cdots\cdots &\geqslant \cdots\\ a_{m1}x_1+\cdots+a_{mn}x_n &\geqslant b_m\\ x_1,\ldots,x_n \geqslant 0 \end{cases} \end{aligned}$$

to opowiadające mu zadanie dualne (ZD) ma postać:

$$\begin{aligned} &\max(b_1\lambda_1+\cdots+b_m\lambda_m)\\ &\begin{cases} a_{11}\lambda_1+\cdots+a_{m1}\lambda_m &\leqslant c_1\\ \quad\cdots\cdots\cdots\cdots\cdots &\leqslant \cdots\\ a_{1n}\lambda_1+\cdots+a_{mn}\lambda_m &\leqslant c_n\\ \lambda_1,\ldots,\lambda_m \geqslant 0 \end{cases} \end{aligned}$$

Krócej możemy zapisać (ZPL) w postaci:

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x}\\ &\mathbf{A}\mathbf{x} \geqslant \mathbf{b}\\ &\mathbf{x} \geqslant \mathbf{0} \end{aligned} \tag{4.5}$$

któremu odpowiada następujące (ZD):

$$\begin{array}{l} \max \mathbf{b}^T \boldsymbol{\lambda} \\ \mathbf{A}^T \boldsymbol{\lambda} \leqslant \mathbf{c} \\ \boldsymbol{\lambda} \geqslant \mathbf{0} \end{array} \tag{4.6}$$

gdzie $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^T$.

Uwaga 4.3.

1. Jeżeli w (ZPL) mamy ograniczenie typu

$$a_{j1}x_1 + \cdots + a_{jn}x_n \leqslant b_j \text{ dla pewnego } j \in \{1, \ldots, m\}$$

   to w (ZD) $\lambda_j \leqslant 0$.
2. Jeżeli w (ZPL) mamy ograniczenie typu

$$a_{j1}x_1 + \cdots + a_{jn}x_n = b_j \text{ dla pewnego } j \in \{1, \ldots, m\}$$

   to w (ZD) zmienna λ_j jest swobodna.
3. Jeżeli w (ZPL) mamy ograniczenie typu $x_i \leqslant 0$ dla pewnego $i \in \{1, \ldots, n\}$, to w (ZD)

$$a_{1i}\lambda_1 + \cdots + a_{mi}\lambda_m \geqslant c_i$$

4. Jeżeli w (ZPL) zmienna x_i dla pewnego $i \in \{1, \ldots, n\}$ jest swobodna, to w (ZD)

$$a_{1i}\lambda_1 + \cdots + a_{mi}\lambda_m = c_i$$

Łatwo zauważyć, że prawdziwe jest:

Twierdzenie 4.3. *Zadaniem dualnym do zadania dualnego jest zadanie pierwotne.*

Istnieje ścisła zależność między rozwiązaniami optymalnymi zadania (ZPL) i (ZD), przedstawiona w poniższych twierdzeniach 4.4–4.6.

Twierdzenie 4.4. *Jeżeli (ZPL) i (ZD) mają rozwiązania dopuszczalne, to oba zadania mają także rozwiązania optymalne. Jeżeli przynajmniej jedno z zadań (ZPL) lub (ZD) nie ma rozwiązania dopuszczalnego, to żadne z nich nie ma rozwiązania optymalnego.*

Twierdzenie 4.5 (słaba dualność). *Jeżeli* $\mathbf{x}$ *jest dowolnym rozwiązaniem dopuszczalnym (ZPL), a* $\boldsymbol{\lambda}$ *jest dowolnym rozwiązaniem (ZD), to zachodzi nierówność* $\mathbf{c}^T\mathbf{x} \geqslant \mathbf{b}^T\boldsymbol{\lambda}$*, tzn. wartość funkcji celu w (ZPL) jest nie mniejsza niż wartość funkcji celu w (ZD).*

Oczywiście kierunek nierówności w Twierdzeniu 4.6 dotyczy przypadku, gdy (ZPL) jest określone wzorem (4.5), a (ZD) wzorem (4.6). W szczególnym przypadku otrzymujemy:

Twierdzenie 4.6 (silna dualność). *Jeżeli istnieją takie rozwiązanie dopuszczalne* $\mathbf{x}^\star$ *zadania (ZPL) oraz rozwiązanie dopuszczalne* $\boldsymbol{\lambda}^\star$ *zadania (ZD), że* $\mathbf{c}^T\mathbf{x}^\star = \mathbf{b}^T\boldsymbol{\lambda}^\star$*, to* $\mathbf{x}^\star$ *jest rozwiązaniem optymalnym (ZPL), a* $\boldsymbol{\lambda}^\star$ *rozwiązaniem optymalnym (ZD).*

Twierdzenie 4.6 pozwala na uzyskanie rozwiązania optymalnego (ZPL) dzięki rozwiązaniu (ZD). Taki sposób postępowania jest szczególnie korzystny w przypadku, gdy w zadaniu pierwotnym liczba zmiennych jest mniejsza niż liczba ograniczeń reprezentowanych przez macierz $\mathbf{A}$. Wówczas rozwiązywanie (ZD) wymaga mniejszego nakładu obliczeń niż bezpośrednie rozwiązywanie (ZPL). Sposób wykorzystania (ZD) do rozwiązania (ZPL) przedstawia poniższy przykład 4.10.

Przykład 4.10. Sformułować zadanie dualne do zadania:

$$\min(x_1+x_2+x_3)$$
$$\begin{cases} x_1+2x_2+x_3 \geqslant 2 \\ 2x_1+x_2-x_3 \geqslant 10 \\ x_1,x_2,x_3 \geqslant 0 \end{cases}$$

Znaleźć rozwiązanie optymalne zadania pierwotnego, rozwiązując zadanie dualne.

Rozwiązanie:
Zadanie dualne ma postać:

$$\max(2\lambda_1+10\lambda_2)$$
$$\begin{cases} \lambda_1+2\lambda_2 \leqslant 1 \\ 2\lambda_1+\lambda_2 \leqslant 1 \\ \lambda_1-\lambda_2 \leqslant 1 \\ \lambda_1,\lambda_2 \geqslant 0 \end{cases}$$

Sprowadzamy je do postaci standardowej:

$$-\min(-2\lambda_1-10\lambda_2)$$
$$\begin{cases} \lambda_1+2\lambda_2+\lambda_3 = 1 \\ 2\lambda_1+\lambda_2+\lambda_4 = 1 \\ \lambda_1-\lambda_2+\lambda_5 = 1 \\ \lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5 \geqslant 0 \end{cases}$$

Jest to jednocześnie postać kanoniczna, mamy bowiem w macierzy ograniczeń kolumny jednostkowe, z których można utworzyć macierz jednostkową. Do rozwiązania ostatnio zapisanej postaci zadania dualnego możemy zatem zastosować zrewidowaną metodę sympleks.

Zapisujemy współczynniki ograniczeń w macierzy:

$$\mathbf{A}_D = \overset{\text{I}\quad\text{II}\quad\text{III}\quad\text{IV}\quad\text{V}}{\left[\begin{array}{|cc|ccc|}\hline 1 & 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 1 \\ \hline\end{array}\right]} \begin{matrix} \\ \\ \\ \\ \mathbf{A}_{DN} \end{matrix} \quad \begin{matrix} \\ \\ \\ \\ \mathbf{B}_D \end{matrix}$$

Otrzymujemy:

$$\boldsymbol{\lambda} = \overset{\text{I}\quad\text{II}\quad\text{III}\quad\text{IV}\quad\text{V}}{\left[\begin{array}{|cc|ccc|}\hline \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 & \lambda_5 \\ \hline\end{array}\right]}^T \begin{matrix} \\ \\ \boldsymbol{\lambda}_N \end{matrix} \quad \begin{matrix} \\ \\ \boldsymbol{\lambda}_B \end{matrix}$$

oraz

$$\mathbf{c}_D = \overset{\text{I}\quad\text{II}\quad\text{III}\,\text{IV}\,\text{V}}{\left[\begin{array}{|cc|ccc|}\hline -2 & -10 & 0 & 0 & 0 \\ \hline\end{array}\right]}^T \begin{matrix} \\ \\ \mathbf{c}_{DN} \end{matrix} \quad \begin{matrix} \\ \\ \mathbf{c}_{DB} \end{matrix}$$

Iteracja nr 1:
Przyjmujemy $\boldsymbol{\lambda}_N = [0\ 0]^T$ i obliczamy:

$$\boldsymbol{\lambda}_B = \mathbf{B}_D^{-1}\mathbf{b}_D = \mathbf{b}_D = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Wartość funkcji celu:

$$f_D(\boldsymbol{\lambda}) = \mathbf{c}_{DB}{}^T \boldsymbol{\lambda}_B = \begin{bmatrix} 0\ 0\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = 0$$

Aby zdecydować, którą kolumnę należy wprowadzić do bazy, obliczamy:

$$\mathbf{p}_{DN}^T = \mathbf{c}_{DN}{}^T - \mathbf{c}_{DB}{}^T \mathbf{B}_D^{-1}\mathbf{A}_{DN} = [-2\ -10] - [0\ 0\ 0] \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 1 & -1 \end{bmatrix} =$$
$$\overset{\text{I}\quad\text{II}}{= \left[\begin{array}{cc} -2 & \boxed{-10} \end{array}\right]}$$

Wprowadzamy kolumnę II (związaną ze zmienną λ_2), gdyż -10 jest najmniejszą liczbą ujemną w wektorze $\mathbf{p}_{DN}$.

Wyznaczamy wektor:

$$\mathbf{d} = \mathbf{B}_D^{-1}\mathbf{a}_J = \mathbf{a}_J = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}$$

Obliczamy ilorazy współrzędnych wektora $\boldsymbol{\lambda}_B$ i odpowiednich współrzędnych wektora $\mathbf{d}$. Otrzymujemy:

$$\frac{1}{2}, \quad 1, \quad -1$$

Wybieramy najmniejszy nieujemny iloraz, czyli $\frac{1}{2}$. Otrzymaliśmy go, dzieląc liczby odpowiadające kolumnie III, zatem usuwamy ją z bazy. Dostajemy:

$$\mathbf{A}_D = \begin{array}{c} \text{I}\ \ \text{III}\quad\ \ \text{II}\ \ \text{IV}\ \text{V} \\ \left[\begin{array}{|cc|cc|} \hline 1 & 1 & 2 & 0 & 0 \\ 2 & 0 & 1 & 1 & 0 \\ 1 & 0 & -1 & 0 & 1 \\ \hline \end{array}\right] \\ \mathbf{A}_{DN} \quad \mathbf{B}_D \end{array}$$

więc

$$\boldsymbol{\lambda} = \begin{array}{c} \text{I}\ \ \text{III}\ \ \text{II}\ \ \text{IV}\ \text{V} \\ \left[\ \boxed{\lambda_1\ \lambda_3}\boxed{\lambda_2\ \lambda_4\ \lambda_5}\ \right]^T \\ \boldsymbol{\lambda}_N \quad \boldsymbol{\lambda}_B \end{array}$$

oraz

$$\mathbf{c}_D = \begin{array}{c} \text{I}\ \ \text{III}\quad\ \ \text{II}\ \ \text{IV}\ \text{V} \\ \left[\ \boxed{-2\ 0}\boxed{-10\ 0\ 0}\ \right]^T \\ \mathbf{c}_{DN} \quad \mathbf{c}_{DB} \end{array}$$

Iteracja nr 2:
Przyjmujemy $\boldsymbol{\lambda}_N = [0\ 0]^T$. Ponieważ

$$\mathbf{B}_D^{-1} = \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix}$$

więc

$$\boldsymbol{\lambda}_B = \mathbf{B}_D^{-1}\mathbf{b}_D = \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{3}{2} \end{bmatrix}$$

Wartość funkcji celu:

$$f_D(\boldsymbol{\lambda}) = \mathbf{c}_{DB}{}^T \boldsymbol{\lambda}_B = \begin{bmatrix} -10 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{3}{2} \end{bmatrix} = -5$$

Obliczamy:

$$\mathbf{p}_{DN}^T = \mathbf{c}_{DN}{}^T - \mathbf{c}_{DB}{}^T \mathbf{B}_D^{-1}\mathbf{A}_{DN} = [-2\ 0] - [-10\ 0\ 0] \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ 1 & 0 \end{bmatrix} =$$

$$\overset{\text{I}\quad\text{III}}{=} [3\ \ 5]$$

Wynika stąd, że rozwiązaniem optymalnym zadania dualnego jest wektor

$$\boldsymbol{\lambda}^\star = [\ \underbrace{\boxed{0\ 0}}_{\boldsymbol{\lambda}_N} \underbrace{\boxed{\tfrac{1}{2}\ \tfrac{1}{2}\ \tfrac{3}{2}}}_{\boldsymbol{\lambda}_B}\ ]^T \quad \overset{\text{I}\ \text{III}\ \text{II}\ \text{IV}\ \text{V}}{}$$

albo po uporządkowaniu numerów zmiennych:

$$\boldsymbol{\lambda}^\star = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & \frac{3}{2} \end{bmatrix}^T \quad \overset{\text{I}\ \ \text{II}\ \text{III}\ \text{IV}\ \ \text{V}}{}$$

Z Twierdzenia 4.6 wynika, że rozwiązanie zadania pierwotnego ma postać:

$$(\mathbf{x}^\star)^T = -\mathbf{c}_{DB}^T \mathbf{B}_D^{-1} = -[-10\ 0\ 0] \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} = [5\ 0\ 0]$$

tzn. $x_1^\star = 5, x_2^\star = 0, x_3^\star = 0$. □

Do wyznaczenia rozwiązania optymalnego (ZPL) na podstawie rozwiązania optymalnego (ZD) albo odwrotnie możemy wykorzystać także:

Twierdzenie 4.7. *Rozwiązania dopuszczalne* $\mathbf{x}$ *oraz* $\boldsymbol{\lambda}$ *odpowiednio (ZPL) i (ZD) są ich rozwiązaniami optymalnymi* $\Longleftrightarrow$

$1^\circ\ (\mathbf{c}-\mathbf{A}^T\boldsymbol{\lambda})^T\mathbf{x}=0$
$2^\circ\ \boldsymbol{\lambda}^T(\mathbf{Ax}-\mathbf{b})=0$

Warunki podane w Twierdzeniu 4.7 nazywamy ***warunkami komplementarności***.

Uwaga 4.4. Z określenia (ZPL) i (ZD) wynika, że wszystkie elementy wektorów $\mathbf{c}-\mathbf{A}^T\boldsymbol{\lambda}$, $\mathbf{Ax}-\mathbf{b}$, $\mathbf{x}$ i $\boldsymbol{\lambda}$ są nieujemne, zatem każda składowa odpowiedniego iloczynu skalarnego tych wektorów w powyższych warunkach 1° i 2° powinna być równa zeru. Oznacza to, że warunki komplementarności mogą być równoważnie zapisane następująco:

$$
\begin{aligned}
&1^\circ\ a_{1i}\lambda_1+\cdots+a_{mi}\lambda_m<c_i \quad\Rightarrow\quad x_i=0 && \forall i\in\{1,\ldots,n\}\\
&2^\circ\ x_i>0 \quad\Rightarrow\quad a_{1i}\lambda_1+\cdots+a_{mi}\lambda_m=c_i && \forall i\in\{1,\ldots,n\}\\
&3^\circ\ a_{j1}x_1+\cdots+a_{jn}x_n>b_j \quad\Rightarrow\quad \lambda_j=0 && \forall j\in\{1,\ldots,m\}\\
&4^\circ\ \lambda_j>0 \quad\Rightarrow\quad a_{j1}x_1+\cdots+a_{jn}x_n=b_j && \forall j\in\{1,\ldots,m\}
\end{aligned}
$$

Przykład 4.11. Korzystając z warunków komplementarności, wyznaczyć rozwiązanie optymalne $[x_1^\star,x_2^\star]$ zadania:

$$
\min(x_1+4x_2)
\begin{cases}
x_1+2x_2\geqslant 10\\
2x_1-x_2\geqslant 5\\
x_1,x_2\geqslant 0
\end{cases}
$$

wiedząc, że $[\lambda_1^\star,\lambda_2^\star]=[1,0]$ jest rozwiązaniem optymalnym zadania dualnego.

Rozwiązanie:
Formułujemy zadanie dualne:

$$
\max(10\lambda_1+5\lambda_2)
\begin{cases}
\lambda_1+2\lambda_2\leqslant 1\\
2\lambda_1-\lambda_2\leqslant 4\\
\lambda_1,\lambda_2\geqslant 0
\end{cases}
$$

Ponieważ $\lambda_1^\star>0$, z punktu 4° Uwagi 4.4:

$$x_1^\star+2x_2^\star=10$$

Z drugiej strony $2\lambda_1^\star-\lambda_2^\star=2<4$, zatem z punktu 1° otrzymujemy $x_2^\star=0$. Ostatecznie zatem $[x_1^\star,x_2^\star]=[10,0]$. □

4.5. Procedury w MATLAB-ie

W MATLAB-ie możemy rozwiązać za pomocą standardowej procedury `linprog` zadanie programowania liniowego postaci:

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x} \\ &\mathbf{A}\mathbf{x} \leqslant \mathbf{b} \\ &\mathbf{A_{eq}}\mathbf{x} = \mathbf{b_{eq}} \\ &\mathbf{xl} \leqslant \mathbf{x} \leqslant \mathbf{xu} \end{aligned} \tag{4.7}$$

przy czym przyjmujemy tu te same oznaczenia co na początku rozdziału 3.7.

Działanie procedury `linprog` rozpoczyna się od wstępnego przekształcenia (ang. *preprocessing*) pierwotnego (ZPL) danego przez (4.7) w celu jego uproszczenia, czyli redukcji zbędnych ograniczeń i zmiennych. W wyniku takiego działania powstaje zadanie:

$$\begin{aligned} &\min \tilde{\mathbf{c}}^T\tilde{\mathbf{x}} \\ &\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}} \\ &\mathbf{0} \leqslant \tilde{\mathbf{x}} \leqslant \mathbf{u} \end{aligned} \tag{4.8}$$

Sprowadzając (4.8) do postaci standardowej, zamieniamy $\tilde{\mathbf{x}} \leqslant \mathbf{u}$ na $\tilde{\mathbf{x}} + \mathbf{z} = \mathbf{u}$ dla $\mathbf{z} \geqslant \mathbf{0}$, co można zapisać także następująco:

$$\begin{aligned} &\min \tilde{\mathbf{c}}^T\tilde{\mathbf{x}} \\ &\begin{bmatrix} \tilde{\mathbf{A}} & \mathbf{0} \\ \mathbf{I} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{b}} \\ \mathbf{u} \end{bmatrix} \\ &\tilde{\mathbf{x}} \geqslant \mathbf{0},\ \mathbf{z} \geqslant \mathbf{0} \end{aligned} \tag{4.9}$$

Można łatwo sprawdzić, że zadanie dualne do (4.9) ma postać:

$$\begin{aligned} &\max[\tilde{\mathbf{b}}^T\ \mathbf{u}^T] \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{w} \end{bmatrix} = \tilde{\mathbf{b}}^T\boldsymbol{\lambda} + \mathbf{u}^T\mathbf{w} \\ &\begin{bmatrix} \tilde{\mathbf{A}}^T & \mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{w} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{c}} \\ \mathbf{0} \end{bmatrix} \\ &\mathbf{w} \leqslant \mathbf{0},\ \mathbf{z} \geqslant \mathbf{0} \end{aligned} \tag{4.10}$$

Algorytm stosowany domyślnie przez procedurę `linprog` wykorzystuje metodę sympleks do rozwiązania zadania dualnego (4.10), aby w konsekwencji otrzymać rozwiązanie zadania pierwotnego (4.8). W każdej iteracji są obliczane tzw. miara niedopuszczalności rozwiązania dualnego, liczona ze wzoru

$$\|\mathbf{A}^T\boldsymbol{\lambda} + \mathbf{w} + \mathbf{z} - \mathbf{c}\|_2 \tag{4.11}$$

oraz miara niedopuszczalności rozwiązania pierwotnego obliczana ze wzoru

$$\| \max\{\mathbf{0}, [\mathbf{xl} - \mathbf{x},\ \mathbf{x} - \mathbf{xu},\ \mathbf{Ax} - \mathbf{b},\ \mathbf{A_{eq}x} - \mathbf{b_{eq}}]\}\|_2 \tag{4.12}$$

Przy czym rozumiemy, że

$$\max\{\mathbf{0},\ \mathbf{y}\} = [\max\{0, y_1\}, \ldots, \max\{0, y_n\}]$$

dla dowolnego wektora $\mathbf{y} = [y_1, \ldots, y_n]^T \in \mathbb{R}^n$ i $\mathbf{0} = [0, \ldots, 0]^T \in \mathbb{R}^n$.

Dodatnia wartość miary wskazuje na to, że dla otrzymanego rozwiązania nie wszystkie ograniczenia zadania optymalizacji są spełnione, zatem rozwiązanie jest niedopuszczalne. Algorytm próbuje wystartować z punktu dopuszczalnego, zatem zwykle wartość miary niedopuszczalności rozwiązania dualnego pozostaje zerowa we wszystkich iteracjach. Inaczej jest z miarą niedopuszczalności rozwiązania pierwotnego, gdyż rozwiązanie bazowe dopuszczalne w zadaniu dualnym nie musi odpowiadać rozwiązaniu bazowemu dopuszczalnemu w zadaniu pierwotnym, chyba że w obu zadaniach są to rozwiązania optymalne.

Przykład 4.12. Korzystając z procedury `linprog`, rozwiązać poniższe (ZPL):

$$\min(x_1 + x_2 + x_3)$$
$$\begin{cases} x_1 + 2x_2 + x_3 \geqslant 2 \\ 2x_1 + x_2 - x_3 \geqslant 10 \\ x_1, x_2, x_3 \geqslant 0 \end{cases}$$

Rozwiązanie:
Wyznaczenie rozwiązania optymalnego nie wymaga definiowania własnych procedur przed wywołaniem standardowej procedury `linprog`. Wszystkie potrzebne dane i obliczenia możemy umieścić w jednym pliku ***.mlx**:

Listing 4.1: Wywołanie procedury `linprog`

```
c = [1,1,1];
A = [-1,-2,-1; -2,-1,1];
b = [-2; -10];
Aeq = [];
Beq = [];
xl = zeros(3,1);
xu = Inf*ones(3,1);
options = optimoptions('linprog','Display','iter');
[x,f] = linprog(c,A,b,Aeq,Beq,xl,xu,options)
```

Oznaczenia zmiennych na listingu 4.1 odpowiadają oznaczeniom we wzorze (4.7). Ponieważ w rozwiązywanym zadaniu nie mamy ograniczeń równościowych, zmiennym `Aeq` i `beq` zostały przypisane puste tablice. Podobnie jak w przypadku wcześniej omawianych procedur standardowych służących do rozwiązywania zadań optymalizacji nieliniowej, także w przypadku procedury `linprog` można zmienić spo-

sób jej działania przez wybór odpowiednich opcji w procedurze `optimoptions`. Wykonanie kodu z listingu 4.1 prowadzi do wyświetlenia na ekranie wyników w postaci:

Listing 4.2: Wynik działania procedury `linprog`

```
LP preprocessing removed 0 inequalities, 0 equalities,
1 variables, and 2 non-zero elements.

 Iter      Time           Fval  Primal Infeas    Dual Infeas
    0     0.004    0.000000e+00   1.019804e+01   0.000000e+00
    1     0.011    5.000000e+00   0.000000e+00   0.000000e+00

Optimal solution found.
x = 3x1
     5
     0
     0

f = 5
```

Na listingu 4.2 w kolumnie `Fval` podane są wartości funkcji celu dla zadania pierwotnego. Kolumna `Primal Infeas` zawiera miarę niedopuszczalności rozwiązania pierwotnego, a kolumna `Dual Infeas` – miarę niedopuszczalności rozwiązania dualnego. □

4.6. Zadania do samodzielnego opracowania

4.6.1. Programowanie w MATLAB-ie

Zadanie 4.1. Utworzyć procedury `ilorazy` oraz `zmien_baze`, które można by zastosować w każdej iteracji dwufazowej metody sympleks do rozwiązania (ZPL) danego w postaci (4.1). Zadaniem procedury `ilorazy` ma być wyświetlenie na ekranie wszystkich ilorazów potrzebnych do podjęcia decyzji, która kolumna tabeli sympleksowej (macierzy ograniczeń **A**) ma być usunięta z bazy. Procedura `zmien_baze` ma dokonać odpowiednich zmian w tabeli sympleksowej przy przejściu do kolejnej bazy.

Wywołanie procedur:

- `ilorazy(A,b,nr_kol_do_bazy)`, gdzie `A` – macierz ograniczeń, `b` – wektor prawej strony, `nr_kol_do_bazy` – numer kolumny macierzy `A` wprowadzanej do bazy;
- `[A,b,c] = zmien_baze(A,b,c,nr_kol_do_bazy, nr_kol_z_bazy)`, gdzie `c` – wektor współczynników funkcji celu,

`nr_kol_z_bazy` – numer kolumny macierzy `A` usuwanej z bazy, przy czym parametry wyjściowe powinny reprezentować macierz `A` i wektory `b` i `c` dla kolejnej tabeli sympleksowej.

Zadanie 4.2. Utworzyć procedurę `zmien_baze_zrew`, którą można by zastosować w każdej iteracji zrewidowanej metody sympleks. Procedura ta ma dokonać przestawienia odpowiednich kolumn w macierzy ograniczeń **A** i wektorze współczynników funkcji celu **c** przy przejściu do kolejnej bazy.

Wywołanie procedury:

- `[AN,B,cN,cB] = zmien_baze_zrew(AN,B,cN,cB, nr_kol_do_bazy,nr_kol_z_bazy)`, gdzie `AN` – część niebazowa macierzy ograniczeń, `B` – macierz bazowa, `cN` oraz `cB` – odpowiednio części niebazowa i bazowa wektora współczynników funkcji celu, przy czym parametry wyjściowe powinny reprezentować macierze `AN` i `B` oraz wektory `cN` i `cB` dla kolejnej tabeli sympleksowej.

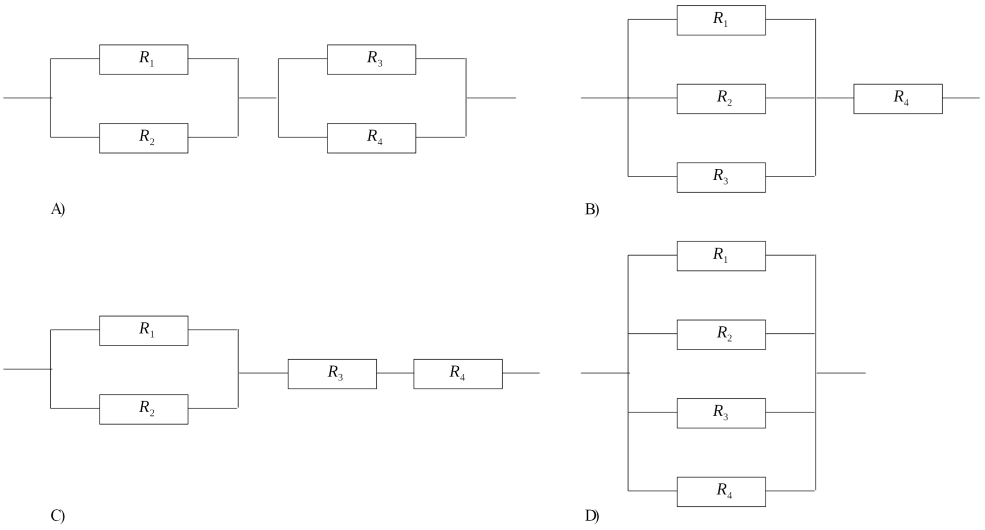
4.6.2. Sprawozdanie

Zadanie 4.3. Dany jest obwód elektryczny złożony z czterech rezystorów, jak na rysunku 4.1. Swobodne końce obwodu podłączamy do źródła prądu stałego. Wartości rezystancji poszczególnych oporników wynoszą: $R_1 = \ldots$, $R_2 = \ldots$, $R_3 = \ldots$, $R_4 = \ldots$, a maksymalne prądy mogące przez nie płynąć wynoszą odpowiednio: $I_1 = \ldots$, $I_2 = \ldots$, $I_3 = \ldots$, $I_4 = \ldots$. Stosując dwufazową metodę sympleks, wyznaczyć maksymalną wartość prądu, który może płynąć przez cały układ elektryczny. Jakie prądy płyną wtedy przez poszczególne oporniki? Wykorzystać procedury `ilorazy` oraz `zmien_baze` utworzone w zadaniu 4.1. Porównać otrzymane wyniki z obliczeniami wykonanymi za pomocą standardowej procedury `linprog`. W rozwiązaniu podać:

- sformułowanie (ZPL) wynikające bezpośrednio z powyższej treści;
- postać standardową (ZPL);
- postać (ZPL) rozwiązywanego w I fazie;
- wyniki otrzymane w I fazie (umieścić je w poniższej tabeli):

Nr tabeli metody sympleks	Bazowe rozwiązanie dopuszczalne	Wartość funkcji celu	Nr kolumny wprowadzanej do bazy	Nr kolumny usuwanej z bazy
0	$\mathbf{x} = [\ldots\ldots]$			
1	$\mathbf{x} = [\ldots\ldots]$			
$\vdots$				

- postać (ZPL) rozwiązywanego w II fazie;
- końcową tabelę metody sympleks w II fazie;
- wyniki otrzymane w II fazie (umieścić je w identycznej tabeli jak w I fazie);
- kod związany z wywołaniem procedury `linprog`.



Rysunek 4.1: Rysunki obwodów elektrycznych do zadania 4.3

Uwaga 4.5.

1. Aby sformułować (ZPL), należy skorzystać z praw Kirchhoffa i z prawa Ohma.
2. Proszę nie redukować ilości zmiennych ani ograniczeń w (ZPL) przed przystąpieniem do jego rozwiązywania.
3. W wektorze prawej strony **b** pojawią się zera. Może to spowodować zapętlenie się algorytmu – jednym ze sposobów uniknięcia takiej sytuacji jest nieznaczna zmiana wartości zerowych w wektorze **b**, która nie powinna mieć istotnego wpływu na rozwiązanie optymalne zadania. Przed przystąpieniem do rozwiązywania (ZPL) należy zamienić wszystkie zera w wektorze **b** na 0,001 (nie dotyczy to procedury `linprog` – tu zostawiamy wartości zerowe).

Zadanie 4.4. Dane jest (ZPL) określone w poniższym zestawie nr …. Stosując zrewidowaną metodę sympleks, wyznaczyć rozwiązanie optymalne. Wykorzystać procedurę `zmien_baze_zrew` utworzoną w zadaniu 4.2. Porównać otrzymane wyniki z obliczeniami wykonanymi za pomocą standardowej procedury `linprog`.
W rozwiązaniu należy podać:

- początkową tabelę sympleksową;
- dla każdej iteracji: $\mathbf{A}_N$, $\mathbf{B}$, $\mathbf{B}^{-}1$, $\mathbf{c}_B$, $\mathbf{c}_N$, $\mathbf{x}_B$, f, $\mathbf{p}_N$, $\mathbf{d}$ oraz numery kolumn – wprowadzanej do bazy i usuwanej z bazy.

Zestawy do zadania 4.4:

1.

$$\min(-5x_1 - 6x_2 - 7x_3 - 8x_4 - 10x_7 - 2x_8)$$
$$\begin{cases} 3x_2 + x_3 + 5x_4 + 7x_5 + 8x_6 + 4x_7 + x_8 + x_9 = 1 \\ 8x_1 + 9x_2 + 7x_3 + 3x_4 + 7x_5 + 2x_6 + 8x_7 + x_{10} = 2 \\ 5x_1 + 9x_2 + x_3 + 7x_4 + 7x_5 + 6x_6 + 7x_7 + 2x_8 + x_{11} = 3 \\ 6x_1 + 4x_2 + 3x_3 + 5x_5 + 6x_6 + x_7 + 2x_8 + x_{12} = 4 \\ 8x_1 + x_2 + 2x_3 + 8x_4 + 9x_5 + 9x_6 + 8x_7 + 7x_8 + x_{13} = 5 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

2.

$$\min(-8x_1 - 12x_2 - 10x_3 - 6x_4 + x_9)$$
$$\begin{cases} 3x_1 + 6x_2 + 6x_3 + 9x_4 + 8x_5 + 3x_6 + 3x_7 + 8x_8 + x_9 = 7 \\ 5x_1 + 8x_2 + 7x_3 + 6x_4 + 9x_5 + 4x_6 + 8x_7 + x_8 + x_{10} = 5 \\ 7x_1 + 5x_2 + 6x_4 + 9x_5 + 9x_6 + 3x_7 + x_{11} = 1 \\ 4x_1 + 3x_2 + 9x_3 + 8x_4 + x_5 + 6x_6 + 4x_8 + x_{12} = 9 \\ x_1 + x_2 + 8x_3 + 5x_4 + 4x_5 + 9x_7 + 5x_8 + x_{13} = 6 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

3.

$$\min(-5x_1 - 4x_2 - 3x_3 - 2x_4 + x_{10})$$
$$\begin{cases} x_1 + 6x_2 + x_3 + 4x_4 + 3x_5 + 7x_6 + 6x_8 + x_9 = 5 \\ x_1 + 4x_2 + 7x_3 + 2x_4 + 9x_5 + 6x_6 + 8x_7 + x_8 + x_{10} + = 9 \\ 6x_3 + 2x_4 + 3x_5 + 8x_6 + 7x_7 + 2x_8 + x_{11} = 7 \\ 9x_1 + x_2 + 9x_3 + x_4 + 2x_5 + 3x_7 + 4x_8 + x_{12} = 2 \\ 5x_1 + 6x_3 + 3x_4 + 3x_5 + 4x_6 + x_8 + x_{13} = 8 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

4.

$$\min(-x_1 - 5x_2 - 3x_3 - 4x_4)$$
$$\begin{cases} 7x_1 + 2x_2 + x_4 + 2x_5 + 3x_6 + 4x_7 + 6x_8 + x_9 = 3 \\ 8x_1 + 6x_3 + 8x_4 + 8x_6 + 5x_7 + 3x_8 + x_{10} = 8 \\ 4x_1 + 3x_2 + 2x_3 + 7x_4 + x_5 + 2x_6 + x_7 + 9x_8 + x_{11} = 11 \\ 4x_2 + 8x_3 + 7x_4 + 6x_7 + 3x_8 + x_{12} = 7 \\ x_1 + 3x_2 + 2x_3 + 8x_5 + 3x_6 + 9x_7 + 3x_8 + x_{13} = 4 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

5.

$$\min(-x_1 - 3x_4 - 7x_5 - 7x_8 + x_{10} + x_{11})$$
$$\begin{cases} x_1 + 4x_2 + 5x_3 + 4x_4 + 3x_5 + 7x_6 + 2x_7 + x_9 = 4 \\ 5x_1 + 7x_2 + 3x_3 + 5x_4 + 8x_5 + 6x_6 + 2x_7 + 6x_8 + x_{10} = 8 \\ 5x_2 + x_3 + 9x_5 + 4x_6 + 5x_7 + 3x_8 + x_{11} = 10 \\ 3x_1 + 2x_2 + x_3 + 5x_4 + 5x_5 + 3x_6 + x_7 + 4x_8 + x_{12} = 3 \\ 5x_1 + 7x_2 + 9x_5 + 9x_7 + 6x_8 + x_{13} = 6 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

6.

$$\min(-x_1 - 3x_2 - 4x_5 - x_7 + 30x_{10} + x_{13})$$
$$\begin{cases} x_1 + 2x_2 + 9x_3 + 8x_4 + 5x_5 + 6x_6 + 6x_7 + 2x_8 + x_9 = 7 \\ 2x_2 + 4x_3 + 8x_4 + 5x_5 + 5x_6 + 9x_7 + 7x_8 + x_{10} = 5 \\ 4x_1 + 4x_2 + 3x_3 + 6x_4 + 4x_5 + 2x_7 + x_8 + x_{11} = 3 \\ 4x_1 + 8x_3 + 8x_4 + 4x_5 + 2x_6 + 4x_7 + 7x_8 + x_{12} = 10 \\ 5x_1 + x_2 + 8x_3 + 9x_4 + 2x_5 + 3x_6 + 3x_7 + 4x_8 + x_{13} = 9 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

7.

$$\min(-3x_1 - 5x_3 - 6x_4 - 4x_7 + x_{10} + x_{11})$$
$$\begin{cases} 2x_1 + 9x_2 + 8x_3 + 3x_4 + x_6 + 4x_7 + x_9 = 4 \\ 3x_1 + 2x_2 + 4x_3 + x_4 + 9x_5 + 9x_6 + 3x_7 + x_8 + x_{10} = 2 \\ 6x_1 + 6x_3 + x_4 + 2x_5 + 7x_6 + 9x_7 + 6x_8 + x_{11} = 3 \\ 8x_1 + 4x_2 + 3x_3 + 4x_4 + 6x_5 + 6x_6 + 2x_7 + 2x_8 + x_{12} = 5 \\ 7x_1 + 2x_3 + 4x_4 + 5x_5 + 6x_6 + 3x_7 + 3x_8 + x_{13} = 7 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

8.

$$\min(-x_1 - x_2 - x_5 - x_6 - 4x_8 + 10x_9 + 10x_{13})$$
$$\begin{cases} 4x_1 + x_2 + 8x_4 + 5x_5 + 8x_6 + 2x_7 + 3x_8 + x_9 = 5 \\ 8x_1 + 8x_2 + 8x_3 + x_4 + 7x_5 + x_6 + 4x_7 + 2x_8 + x_{10} = 1 \\ 7x_1 + 3x_2 + 6x_3 + 4x_4 + 6x_5 + 6x_6 + 2x_7 + 2x_8 + x_{11} = 8 \\ 4x_1 + 3x_2 + 4x_3 + 7x_4 + 7x_5 + 2x_6 + 6x_7 + 6x_8 + x_{12} = 2 \\ 8x_1 + 3x_2 + 4x_3 + 9x_4 + 3x_5 + 5x_7 + 3x_8 + x_{13} = 7 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

9.

$$\min(-2x_1 - 2x_3 - 3x_4 + 2x_9 + x_{10})$$
$$\begin{cases} 3x_1 + 5x_2 + 2x_3 + 5x_4 + 9x_5 + 9x_6 + 5x_7 + 9x_8 + x_9 = 5 \\ 9x_1 + 5x_2 + 8x_3 + 4x_4 + 7x_5 + 5x_6 + 9x_7 + x_{10} = 1 \\ 9x_1 + 6x_2 + 2x_3 + 6x_5 + x_6 + 7x_7 + 8x_8 + x_{11} = 3 \\ 4x_1 + 2x_2 + x_3 + 9x_4 + 3x_5 + 4x_7 + 5x_8 + x_{12} = 2 \\ 6x_1 + 2x_2 + 4x_3 + 2x_4 + 4x_5 + 6x_6 + 8x_7 + 7x_8 + x_{13} = 1 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

10.

$$\min(-x_1 - x_4 - x_6 - x_7 + x_{11} + x_{12})$$
$$\begin{cases} 9x_1 + 7x_2 + 8x_3 + 7x_4 + 3x_5 + 4x_6 + 4x_8 + x_9 = 7 \\ 6x_1 + 4x_2 + 2x_3 + 3x_4 + 6x_5 + 2x_6 + 6x_7 + x_{10} = 5 \\ x_1 + 2x_2 + x_3 + 4x_4 + 5x_6 + 9x_7 + x_8 + x_{11} = 1 \\ x_1 + x_2 + 3x_3 + 5x_4 + 5x_5 + 6x_6 + 3x_7 + 5x_8 + x_{12} = 6 \\ 8x_1 + 4x_3 + x_4 + 8x_6 + 9x_7 + 9x_8 + x_{13} = 4 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

11.

$$\min(-x_1 - x_4 - x_6 + x_9)$$
$$\begin{cases} x_2 + 8x_3 + 7x_5 + 8x_6 + x_7 + 2x_8 + x_9 = 8 \\ 3x_1 + 2x_2 + 9x_3 + 5x_4 + 8x_5 + 2x_6 + 7x_7 + 9x_8 + x_{10} = 4 \\ x_1 + 3x_3 + x_4 + 5x_5 + 7x_6 + 8x_7 + 4x_8 + x_{11} = 2 \\ x_1 + 8x_2 + 9x_3 + 7x_4 + 3x_5 + 7x_6 + 2x_7 + 8x_8 + x_{12} = 10 \\ 5x_2 + 9x_3 + x_4 + 7x_5 + 7x_6 + 6x_7 + 7x_8 + x_{13} = 1 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

12.

$$\min(-2x_3 - 4x_5 - 7x_6 + x_{10} + x_{11} + x_{13})$$
$$\begin{cases} 2x_1 + 6x_2 + 4x_3 + 3x_4 + 5x_6 + 6x_7 + x_8 + x_9 = 4 \\ 2x_1 + 8x_2 + x_3 + 2x_4 + 8x_5 + 9x_6 + 9x_7 + 8x_8 + x_{10} = 1 \\ 7x_1 + 3x_2 + 6x_3 + 6x_4 + 9x_5 + 8x_6 + 3x_7 + 3x_8 + x_{11} = 1 \\ 8x_1 + 5x_2 + 8x_3 + 7x_4 + 6x_5 + 9x_6 + 4x_7 + 8x_8 + x_{12} = 1 \\ x_1 + 7x_2 + 5x_3 + 6x_5 + 9x_6 + 9x_7 + 3x_8 + x_{13} = 2 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

13.

$$\min(-3x_4 - 5x_5 - 6x_7)$$
$$\begin{cases} 4x_2 + 3x_3 + 9x_4 + 8x_5 + x_6 + 6x_7 + x_9 = 5 \\ 4x_1 + x_2 + x_3 + 8x_4 + 5x_5 + 4x_6 + 9x_8 + x_{10} = 8 \\ 5x_1 + x_2 + 6x_3 + x_4 + 4x_5 + 3x_6 + 7x_7 + x_{11} = 2 \\ 6x_1 + x_2 + 4x_3 + 7x_4 + 2x_5 + 9x_6 + 6x_7 + 8x_8 + x_{12} = 4 \\ x_1 + 6x_4 + 2x_5 + 3x_6 + 8x_7 + 7x_8 + x_{13} = 3 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

14.

$$\min(-x_3 - x_5 - 6x_7 + x_{10} + x_{11})$$
$$\begin{cases} 2x_1 + 9x_2 + x_3 + 9x_4 + x_5 + 2x_6 + 3x_8 + x_9 = 1 \\ 4x_1 + 5x_2 + 6x_4 + 3x_5 + 3x_6 + 4x_7 + x_{10} = 3 \\ x_1 + 7x_2 + 2x_3 + x_5 + 2x_6 + 3x_7 + 4x_8 + x_{11} = 1 \\ 6x_1 + 8x_2 + 6x_4 + 8x_5 + 8x_7 + 5x_8 + x_{12} = 2 \\ 3x_1 + 4x_2 + 3x_3 + 2x_4 + 7x_5 + x_6 + 2x_7 + x_8 + x_{13} = 1 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

15.

$$\min(-x_3 - x_5 - x_7 + x_9 + x_{10} + x_{12})$$
$$\begin{cases} 6x_1 + 8x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 + 6x_7 + 2x_8 + x_9 = 1 \\ 2x_1 + 7x_2 + 2x_4 + 4x_5 + 5x_6 + 6x_7 + 3x_8 + x_{10} = 1 \\ 3x_1 + 9x_2 + 3x_3 + 5x_4 + 4x_5 + 8x_6 + x_7 + 4x_8 + x_{11} = 1 \\ x_1 + 2x_2 + 8x_4 + 6x_5 + 4x_6 + 6x_7 + 5x_8 + x_{12} = 1 \\ 9x_1 + 9x_3 + 3x_4 + 2x_5 + 5x_6 + 7x_7 + 7x_8 + x_{13} = 1 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

16.

$$\min(-x_3 - x_5 - x_6 + x_9)$$
$$\begin{cases} x_1 + 6x_2 + 3x_4 + 6x_5 + 8x_6 + 3x_8 + x_9 = 1 \\ 4x_1 + 9x_2 + 4x_3 + 3x_4 + 7x_5 + 3x_6 + 5x_7 + 9x_8 + x_{10} = 1 \\ 8x_1 + 4x_2 + x_3 + 8x_5 + 5x_6 + 8x_7 + 2x_8 + x_{11} = 2 \\ 3x_1 + 8x_2 + 8x_3 + 8x_4 + x_5 + 7x_6 + x_7 + 4x_8 + x_{12} = 3 \\ 2x_1 + 7x_2 + 3x_3 + 6x_4 + 4x_5 + 6x_6 + 6x_7 + 2x_8 + x_{13} = 1 \\ x_i \geqslant 0 \text{ dla } i \in \{1, \ldots, 13\} \end{cases}$$

Rozdział 5

Programowanie całkowitoliczbowe

Przedmiotem rozważań w tym rozdziale będzie zadanie:

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x} \\ &\mathbf{A}\mathbf{x} = \mathbf{b} \\ &\mathbf{d} \leqslant \mathbf{x} \leqslant \mathbf{g} \\ &x_i \in \mathbb{Z} \quad \text{dla pewnych } i \in \{1,\ldots,n\} \end{aligned} \tag{5.1}$$

przy czym założenia dotyczące wektorów $\mathbf{c}$, $\mathbf{x}$, $\mathbf{b}$ i macierzy $\mathbf{A}$ są identyczne jak w rozdziale 4. Dodatkowo przyjmujemy, że $\mathbf{d} = [d_1,\ldots,d_n]^T \in \mathbb{R}^n$ oraz $\mathbf{g} = [g_1,\ldots,g_n]^T \in \mathbb{R}^n$ są wektorami liczb rzeczywistych. Dopuszczamy możliwość, że $d_i = -\infty$ lub $g_i = +\infty$ dla pewnych $i \in \{1,\ldots,n\}$. Zapis $\mathbf{x} \leqslant \mathbf{g}$ oznacza, że $x_1 \leqslant g_1,\ldots,x_n \leqslant g_n$. Podobnie interpretujemy zapis $\mathbf{d} \leqslant \mathbf{x}$. Zadanie (5.1) przy powyższych założeniach nazywamy zadaniem programowania całkowitoliczbowego (oznaczenie: (ZPC)). Jeżeli w zadaniu (5.1) ostatnie założenie dotyczy wszystkich zmiennych decyzyjnych, tzn. $x_i \in \mathbb{Z}$ dla wszystkich $i \in \{1,\ldots,n\}$, to mówimy o zadaniu czystym, a jeżeli tylko niektórych zmiennych, to mamy do czynienia z zadaniem mieszanym.

Zadanie (5.1) bez ostatniego założenia, czyli

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x} \\ &\mathbf{A}\mathbf{x} = \mathbf{b} \\ &\mathbf{d} \leqslant \mathbf{x} \leqslant \mathbf{g} \end{aligned} \tag{5.2}$$

jest zadaniem programowania liniowego, które rozpatrywaliśmy w rozdziale 4. Jak wiemy, możemy je rozwiązać np. metodą sympleks. Dołączenie warunku całkowitoliczbowości powoduje, że zmienia się typ zadania, którego rozwiązanie wymaga teraz zastosowania innych metod. Wyjątkiem jest sytuacja, kiedy rozwiązaniem optymalnym zadania programowania liniowego (5.2) jest taki wektor $\mathbf{x}^\star$, którego odpowiednie elementy są całkowite, zgodnie z warunkiem całkowitoliczbowości podanym w (5.1). Wówczas $\mathbf{x}^\star$ jest jednocześnie rozwiązaniem optymalnym zadania (5.1).

W przeciwnym razie należy zastosować jedną z metod rozwiązywania (ZPC). Najprostszym sposobem rozwiązania (ZPC) jest obliczenie wartości funkcji celu we wszystkich punktach $\mathbf{x}$ należących do zbioru ograniczeń i wybranie namniejszej spośród tych wartości. Taki sposób może być skuteczny wyłącznie wtedy, gdy zbiór ograniczeń jest skończony i niezbyt liczny (oznacza to w szczególności, że wszystkie zmienne x_i powinny przyjmować wartości całkowite). Ponadto fakt przynależności do zbioru ograniczeń powinien być łatwy do sprawdzenia, co zwykle ma miejsce

tylko wtedy, gdy liczba n zmiennych decyzyjnych jest niewielka. Prowadzi to nas do wniosku, że powyższy sposób w praktyce jest niezbyt użyteczny. Powszechnie stosowaną jest metoda podziału i ograniczeń, inną, mniej uniwersalną, jest metoda cięć. Obie są przedstawione w kolejnych podrozdziałach. W każdej z nich konstruowany jest ciąg zadań pomocniczych $L_0, \ldots, L_t$, z których każde jest zadaniem programowania liniowego, które możemy rozwiązać metodą sympleks, jednak w tej sytuacji jest to metoda podrzędna.

Warto zwrócić uwagę na istotne różnice pomiędzy rozwiązaniami optymalnymi (ZPC) i (ZPL) określonymi odpowiednio przez (5.1) i (5.2):

1. Jeżeli D jest zbiorem ograniczeń (ZPL), to rozwiązanie optymalne (ZPL) leży zawsze na jego brzegu, natomiast rozwiązanie optymalne (ZPC) może znajdować się także wewnątrz tego zbioru.
2. Jeżeli ilość rozwiązań optymalnych (ZPC) jest skończona, może istnieć więcej niż jedno takie rozwiązanie.
3. Przypuśćmy, że wyznaczyliśmy rozwiązanie optymalne $\mathbf{x}^\star$ zadania (5.2). Zaokrąglenie wartości niecałkowitych do całkowitych w $\mathbf{x}^\star$ zwykle nie spowoduje, że otrzymamy rozwiązanie optymalne $\mathbf{x}_C^\star$ zadania (5.1). Tak wyznaczone przybliżenie punktu $\mathbf{x}_C^\star$ może znajdować się nawet poza zbiorem ograniczeń D.

5.1. Metoda podziału i ograniczeń

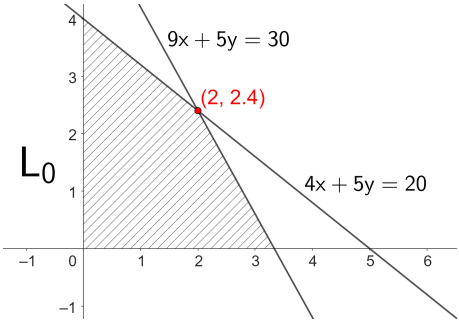
Do rozwiązywania zarówno czystych, jak i mieszanych (ZPC) możemy stosować metodę podziału i ograniczeń. Wersja metody podana poniżej służy do wyznaczania maksimum funkcji celu.

Przykład 5.1. Rozwiązać zadanie:

$$\text{(ZPC):} \qquad \max(x+y) \quad \begin{cases} 4x+5y \leqslant 20 \\ 9x+5y \leqslant 30 \\ x \geqslant 0,\ y \geqslant 0 \\ x, y \in \mathbb{Z} \end{cases}$$

Rozwiązanie:
Rozpatrujemy najpierw pomocnicze zadanie L_0, które różni się od zadania (ZPC) jedynie pominięciem ograniczeń $x, y \in \mathbb{Z}$, czyli ma postać:

$$L_0: \qquad \max(x+y) \quad \begin{cases} 4x+5y \leqslant 20 \\ 9x+5y \leqslant 30 \\ x \geqslant 0,\ y \geqslant 0 \end{cases}$$



Rysunek 5.1: Interpretacja geometryczna zadania L_0

Na rysunku 5.1 przedstawiono interpretację geometryczną zadania L_0. Kolorem czerwonym zaznaczono jego rozwiązanie optymalne.

Funkcja celu ma postać $f(x,y) = x+y$, zatem optymalna (maksymalna) jej wartość w zadaniu L_0 wynosi:

$$f^\star = f(x^\star, y^\star) = x^\star + y^\star = 2 + 2{,}4 = 4{,}4$$

Ponieważ otrzymane rozwiązanie optymalne $x^\star, y^\star$ nie jest całkowitoliczbowe, w zadaniu L_0 otrzymujemy ograniczenie górne na optymalną wartość (całkowitą, gdyż współczynniki funkcji f są całkowite) funkcji celu pierwotnego (ZPC): $f_C \leqslant 4$.

Rozdzielamy zadanie L_0 na kolejne zadania L_1 i L_2. Możemy dokonać podziału ze względu na każdą zmienną, która nie spełnia warunku całkowitoliczbowości w rozwiązaniu optymalnym zadania L_0, czyli w naszym przypadku tylko ze względu na zmienną y (w L_0 $y^\star$ nie jest całkowite). Utworzenie zadania L_1 polega na dołączeniu do zbioru ograniczeń zadania nadrzędnego, czyli L_0, ograniczenia $y \leqslant [\hat{y}]$, gdzie $[\hat{y}]$ oznacza część całkowitą liczby $\hat{y}$. Analogicznie tworzymy zadanie L_2, dołączając ograniczenie $y \geqslant [\hat{y}] + 1$.

Otrzymujemy zatem:

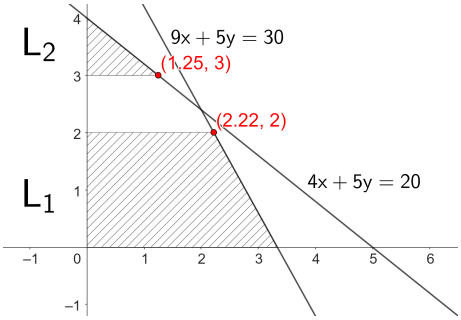
$$L_1: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_0 \text{ oraz} \\ y \leqslant 2 \end{cases}$$

Podobnie zbiór ograniczeń w zadaniu L_2 powinien przyjąć postać:

$$L_2: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_0 \text{ oraz} \\ y \geqslant 3 \end{cases}$$

Optymalna wartość funkcji celu w zadaniu L_1 wynosi:

$$f^\star = f(x^\star, y^\star) = x^\star + y^\star = 2{,}22 + 2 = 4{,}22$$



Rysunek 5.2: Interpretacja geometryczna zadań L_1 i L_2

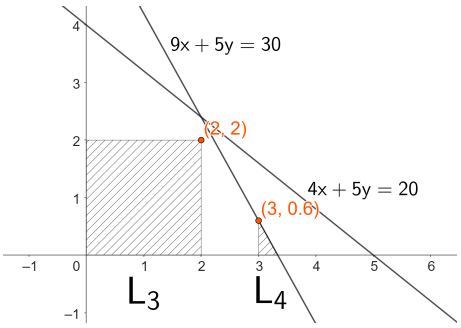
Optymalna wartość funkcji celu w zadaniu L_2 wynosi:

$$f^\star = f(x^\star, y^\star) = x^\star + y^\star = 1,25 + 3 = 4,25$$

Zadanie L_1 rozdzielamy na L_3 i L_4, dołączając dodatkowe ograniczenie na zmienną x (gdyż $x^\star$ w L_1 nie jest całkowite). Zadania L_3 i L_4 mają zbiory ograniczeń:

$$L_3: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_1 \text{ oraz} \\ x \leqslant 2 \end{cases}$$

$$L_4: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_1 \text{ oraz} \\ x \geqslant 3 \end{cases}$$



Rysunek 5.3: Interpretacja geometryczna zadań L_3 i L_4

W zadaniu L_3 rozwiązanie optymalne $x^\star, y^\star$ **jest** całkowitoliczbowe, zaś optymalna wartość funkcji celu wynosi:

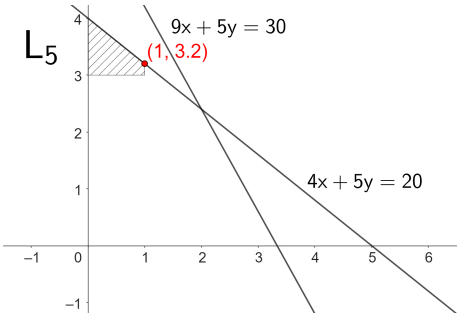
$$f^\star = f(x^\star, y^\star) = x^\star + y^\star = 2 + 2 = 4$$

Oznacza to, że optymalna wartość funkcji celu w pierwotnym (ZPC) na pewno nie jest mniejsza (szukamy maksimum) od otrzymanej optymalnej wartości funkcji celu w zadaniu L_3, czyli $f_C \geqslant 4$. Otrzymaliśmy ograniczenie dolne na f_C. Jednocześnie $f_C \leqslant 4$ (na podstawie rozwiązania L_0), zatem $f_C = 4$ – rozwiązanie optymalne zadania L_3 jest więc jednocześnie rozwiązaniem optymalnym pierwotnego (ZPC).

Zadanie L_2, ponieważ jego rozwiązanie nie jest całkowitoliczbowe, rozdzielamy na zadania L_5 i L_6, dołączając nowe ograniczenie na zmienną x :

$$L_5: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_2 \text{ oraz} \\ x \leqslant 1 \end{cases}$$

$$L_6 \text{ (zbiór ograniczeń jest pusty)}: \quad \begin{cases} \text{wszystkie ograniczenia zadania } L_2 \text{ oraz} \\ x \geqslant 2 \end{cases}$$



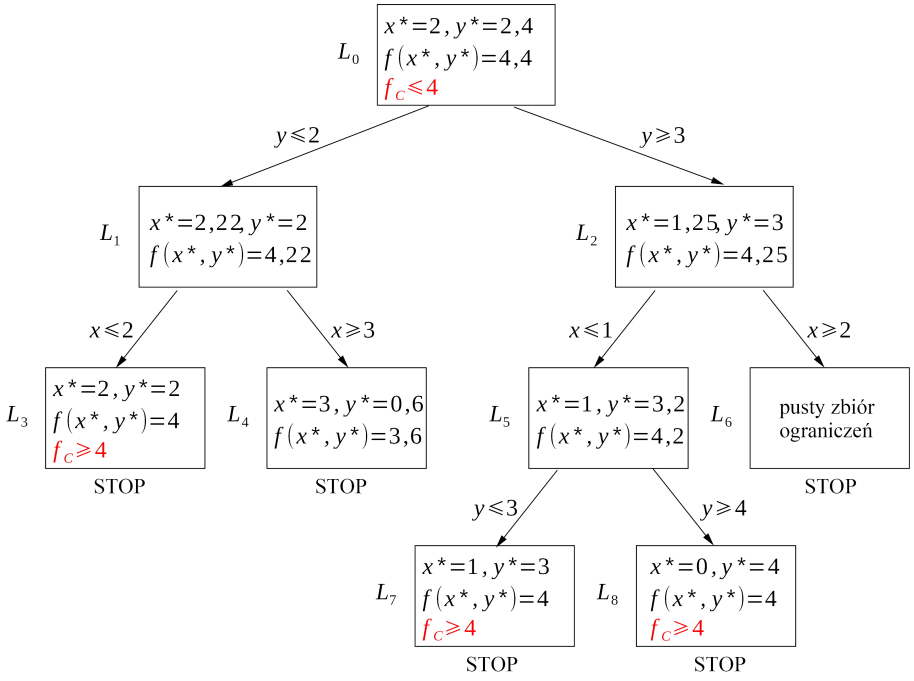
Rysunek 5.4: Interpretacja geometryczna zadania L_5

Zadania L_4 nie trzeba rozdzielać, ponieważ wartość optymalna funkcji celu dla tego zadania wynosi $3,6$, zatem jest mniejsza od optymalnej wartości f_C dla pierwotnego ZPC. Nie jest w związku z tym możliwe, abyśmy uzyskali wyższą wartość funkcji celu na podzbiorze zbioru ograniczeń dla zadania L_4.

W wyniku podziału zadania L_5 na zadania L_7 i L_8 uzyskamy dwa kolejne rozwiązania optymalne pierwotnego (ZPC): $[x^\star, y^\star] = [1,3]$ i $[x^\star, y^\star] = [0,4]$.

Oznacza to, że pierwotne (ZPC) ma trzy rozwiązania optymalne: $[2,2], [1,3]$ i $[0,4]$, dla których $f_C = 4$.

Podział zadania L_0 na zadania podrzędne został przedstawiony także na poniższym diagramie (rysunek 5.5). Nad strzałkami podano ograniczenia, które należy dołączyć do zadania nadrzędnego, aby uzyskać zadanie podrzędne.



Rysunek 5.5: Diagram ilustrujący podział zadania L_0

W rozpatrywanym przykładzie wszystkie rozwiązania optymalne zadań L_k, dla których zarówno $x^\star \in \mathbb{Z}$, jak i $y^\star \in \mathbb{Z}$, są jednocześnie rozwiązaniami optymalnymi (ZPC). Nie zawsze jednak musi tak być – może się zdarzyć, że rozwiązanie optymalne całkowitoliczbowe pewnego zadania L_q nie jest jednocześnie rozwiązaniem optymalnym zadania (ZPC), gdyż istnieje inne zadanie L_r, $r \neq q$, które też ma rozwiązanie optymalne całkowitoliczbowe, ale o wyższej wartości funkcji celu $f(x^\star, y^\star)$ niż w zadaniu L_q. □

Uwaga 5.1.

1. Ograniczenie górne (typu $f_C(\mathbf{x_{opt}}) \leqslant a$) otrzymujemy na podstawie wartości optymalnej zadania L_0.
2. Ograniczenie dolne (typu $f_C(\mathbf{x_{opt}}) \geqslant b$) otrzymujemy na podstawie wartości optymalnej zadania L_k, o ile rozwiązanie optymalne zadania L_k jest całkowitoliczbowe.
3. Jeżeli zadanie L_k dla pewnego k jest sprzeczne lub całkowitoliczbowe, to go nie rozdzielamy.
4. Jeżeli $f(\mathbf{x_{opt}}^{L_k})$ dla pewnego k jest mniejsza od aktualnego dolnego ograniczenia, to także nie rozdzielamy zadania L_k.

5.2. Metoda cięć

W przypadku czystego (ZPC) możemy zastosować także tzw. metodę cięć. Poniższa wersja dotyczy minimalizacji funkcji celu.

Algorytm metody:

1. Rozwiązujemy zadanie L_0 (identyczne jak w metodzie podziału i ograniczeń) metodą sympleks. Jeżeli rozwiązanie optymalne zadania L_0 jest całkowitoliczbowe, kończymy obliczenia, w przeciwnym razie przechodzimy do punktu 2.
2. Z ostatniej tablicy sympleksowej zadania L_0 wybieramy taki wiersz o numerze i_0, w którym element prawej strony (b_{i_0}) ma największą część ułamkową ($b_{i_0} - [b_{i_0}]$ jest największe).
3. Na podstawie wiersza i_0 konstruujemy nowe ograniczenie (równanie cięcia) w postaci:
$$\sum_{j \in I_N} \left([a_{i_0 j}] - a_{i_0 j} \right) x_j + x_{n+1} = [b_{i_0}] - b_{i_0} \tag{5.3}$$
gdzie I_N jest zbiorem indeksów zmiennych niebazowych w ostatniej tablicy sympleksowej zadania L_0 (wprowadziliśmy nową zmienną x_{n+1} i zakładamy, że $x_{n+1} \geqslant 0$).
4. Równanie (5.3) dołączamy do ograniczeń zadania L_0, tworząc nowe zadanie L_1 (ogólnie L_{k+1} z L_k), wracamy do punktu 1 z zadaniem L_k zamiast L_0.

Uwaga 5.2.

1. Dołączenie ograniczenia (5.3) geometrycznie powoduje odcięcie ze zbioru ograniczeń pewnych rozwiązań niecałkowitoliczbowych, ale pozostawia w odciętym zbiorze wszystkie rozwiązania całkowitoliczbowe.
2. Po dołączeniu (5.3) do ostatniej tablicy sympleksowej zadania L_k otrzymujemy od razu rozwiązanie bazowe zadania L_{k+1}, ale jest ono niedopuszczalne ($[b_{i_0}] - b_{i_0} < 0$). Wyznaczenie rozwiązania optymalnego zadania L_{k+1} wymaga zatem zastosowania dualnej metody sympleks.

Przykład 5.2. Wykonać jedną iterację metodą cięć w celu rozwiązania zadania:

$$\text{(ZPC):} \qquad \min(-x-y)$$
$$\begin{cases} 5x+3y+z=15 \\ x+2y+t=6 \\ x \geqslant 0,\ y \geqslant 0,\ z \geqslant 0,\ t \geqslant 0 \\ x,y,z,t \in \mathbb{Z} \end{cases}$$

Rozwiązanie:
Na podstawie ostatniej tabeli sympleksowej dla zadania L_0, mającej postać:

x	y	z	t	b
1	0	$\frac{2}{7}$	$-\frac{3}{7}$	$\frac{12}{7}$
0	1	$-\frac{1}{7}$	$\frac{5}{7}$	$\frac{15}{7}$
0	0	$\frac{1}{7}$	$\frac{2}{7}$	$\frac{27}{7}$

tworzymy nowe ograniczenie:

$$\left(\left[\frac{2}{7}\right]-\frac{2}{7}\right)z+\left(\left[-\frac{3}{7}\right]-\left(-\frac{3}{7}\right)\right)t+u_1=\left[\frac{12}{7}\right]-\frac{12}{7}$$

czyli

$$-\frac{2}{7}z-\frac{4}{7}t+u_1=-\frac{5}{7}$$

Dołączamy nowe ograniczenie do wcześniejszych ograniczeń i otrzymujemy zadanie:

$$L_1: \qquad \min(-x-y)$$
$$\begin{cases} 5x+3y+z=15 \\ x+2y+t=6 \\ -\frac{2}{7}z-\frac{4}{7}t+u_1=-\frac{5}{7} \\ x\geqslant 0,\ y\geqslant 0,\ z\geqslant 0,\ t\geqslant 0, u_1\geqslant 0 \end{cases}$$

Rozwiązanie optymalne zadania L_1 to wektor: [2,25 1,25 0 1,25 0], zatem nie jest to rozwiązanie całkowitoliczbowe. Aby rozwiązać pierwotne zadanie (ZPC), należy więc wykonać kolejną iterację (być może nie jedną), tzn. dołączyć kolejne ograniczenie i wprowadzić kolejną zmienną $u_2\geqslant 0$ do zadania L_1, tworząc zadanie L_2 itd. □

5.3. Procedury w MATLAB-ie

Mieszane lub czyste (ZPC) można rozwiązać w MATLAB-ie za pomocą standardowej procedury `intlinprog`. Takie zadanie powinno mieć postać:

$$\begin{aligned} &\min \mathbf{c}^T\mathbf{x} \\ &\mathbf{Ax}\leqslant\mathbf{b} \\ &\mathbf{A_{eq}x}=\mathbf{b_{eq}} \\ &\mathbf{xl}\leqslant\mathbf{x}\leqslant\mathbf{xu} \\ &x_i\in\mathbb{Z} \text{ dla } i\in I \end{aligned} \tag{5.4}$$

gdzie I jest pewnym podzbiorem zbioru indeksów $\{1,\ldots,n\}$ zmiennych decyzyjnych. Pozostałe oznaczenia są identyczne jak na początku podrozdziału 3.7. Nie

wszystkie typy ograniczeń muszą wystąpić w zadaniu (5.4), można bowiem rozpatrywać np. zadania bez ograniczeń równościowych albo bez ograniczeń nierównościowych.

Działanie procedury `intlinprog` jest dosyć złożone. Można wyróżnić w nim następujące etapy:

1. Wstępna analiza zadania L_0 (według oznaczeń z rozdziału 5.1) odpowiadającego (ZPC) danemu przez (5.4). Proces ten ma na celu ewentualne uproszczenie zadania L_0, czyli redukcję ilości zmiennych decyzyjnych lub zbędnych ograniczeń, jeżeli jest to możliwe.
2. Rozwiązanie uproszczonego w taki sposób zadania L_0.
3. Wstępna analiza (ZPC) danego przez (5.4). Cel tego procesu jest podobny jak w punkcie 1.
4. Wygenerowanie dodatkowych ograniczeń nierównościowych prowadzących do zmniejszenia zbioru ograniczeń, na którym poszukiwane jest rozwiązanie optymalne (ZPC).
5. Zastosowanie metody podziału i ograniczeń.
6. Wykorzystanie metod heurystycznych na każdym etapie metody podziału i ograniczeń, które mają na celu wyznaczenie jakiegokolwiek lub poprawę najlepszego dotychczas wyznaczonego rozwiązania całkowitoliczbowego. Heurystyka oznacza w tym przypadku, że otrzymane rozwiązanie całkowitoliczbowe nie zawsze będzie najlepsze z możliwych do uzyskania w danym kroku. Zwykle jednak zastosowanie metody z tej grupy prowadzi do przyspieszenia procesu obliczeń i nie wymaga przeszukiwania całego drzewa zadań programowania liniowego L_k w celu wyznaczenia rozwiązania optymalnego (ZPC) danego przez 5.4.

Niekoniecznie wszystkie powyższe etapy muszą wystąpić w konkretnym zadaniu – jeżeli w którymś z etapów zostanie znalezione rozwiązanie optymalne (ZPC) albo okaże się, że takie rozwiązanie nie istnieje, procedura `intlinprog` kończy działanie.

Podobnie jak w przypadku innych procedur optymalizacyjnych MATLAB-a, możemy wpływać na działanie procedury `intlinprog` poprzez ustawienie odpowiednich opcji w procedurze `optimoptions`. W szczególności możliwa jest ingerencja w każdy z sześciu etapów działania procedury `intlinprog` wymienionych powyżej, np. przypisanie opcji:

1. `LPPreprocess` wartości `'none'` oznacza, że zostanie pominięta wstępna analiza zadania L_0;
2. `RootLPAlgorithm` wartości `'primal-simplex'` oznacza, że zadania L_k będą rozwiązywane zwykłą metodą sympleks (standardowo są rozwiązywane dualną metodą sympleks);
3. `IntegerPreprocess` wartości `'none'` oznacza, że wstępna analiza (ZPC) będzie przeprowadzona tylko w bardzo podstawowym zakresie;

4. `CutGeneration` wartości `'none'` oznacza, że nie będą generowane dodatkowe ograniczenia nierównościowe prowadzące do zmniejszenia zbioru ograniczeń;
5. `BranchRule` odpowiedniej wartości określa sposób wyboru składowej niecałkowitej w rozwiązaniu zadania L_k w metodzie podziału i ograniczeń, względem której należy dokonać dalszego podziału;
6. `Heuristics` odpowiedniej wartości określa, jaka metoda heurystyczna zostanie użyta do poprawy rozwiązania całkowitoliczbowego.

Przykład 5.3. Korzystając z procedury `intlinprog`, rozwiązać poniższe (ZPC):

$$\text{(ZPC):} \qquad \min(-x-y)$$
$$\begin{cases} 4x+5y \leqslant 20 \\ 9x+5y \leqslant 30 \\ x \geqslant 0,\ y \geqslant 0 \\ x,y \in \mathbb{Z} \end{cases}$$

Rozwiązanie:
Tak określone zadanie jest równoważne (ZPC) rozpatrywanemu w przykładzie 5.1. W MATLAB-ie możemy je rozwiązać w sposób przedstawiony na listingu 5.1:

Listing 5.1: Wywołanie procedury `intlinprog`

```
c = [-1,-1];
A = [4,5; 9,5];
b = [20; 30];
Aeq = [];
Beq = [];
xl = zeros(2,1);
xu = Inf*ones(2,1);
x0 = [];
int = 1:2;
options = optimoptions('intlinprog','Display','iter');
[x,f] = intlinprog(c,int,A,b,Aeq,Beq,xl,xu,x0,options)
```

Widzimy, że instrukcje prowadzące do rozwiązania są bardzo podobne do tych, które zostały użyte w przykładzie 4.12 do rozwiązania (ZPL) za pomocą procedury `linprog`. Należy jednak zwrócić uwagę na istotne różnice: w wywołaniu procedury `intlinprog` występują dwa dodatkowe parametry wejściowe. Jednym z nich jest `int` – wektor indeksów tych zmiennych, które w rozwiązaniu optymalnym (ZPC) mają być całkowite. Drugi to `x0` – wektor startowy mający wpływ na działanie niektórych metod heurystycznych, o których wspomniano powyżej. Odpowiedni wybór wektora `x0`, który powinien być dopuszczalny, może skrócić czas potrzebny na rozwiązanie (ZPC), może też prowadzić do wyznaczenia innego rozwiązania, jeżeli istnieje więcej niż jedno. Jeżeli nie chcemy wskazywać wektora startowego, w kodzie programu należy przypisać zmiennej `x0` pustą tablicę.

W wyniku działania procedury `intlinprog` z powyższymi wartościami parametrów otrzymamy wynik w postaci:

Listing 5.2: Wynik działania procedury `intlinprog`

```
LP:                Optimal objective value is -4.400000.

Heuristics:        Found 1 solution using ZI round.
                   Upper bound is -4.000000.
                   Relative gap is 0.00%.


Optimal solution found.

Intlinprog stopped at the root node because
the objective value is within a gap tolerance
of the optimal value,
options.AbsoluteGapTolerance = 0 (the default
value). The intcon variables are
integer within tolerance,
options.IntegerTolerance = 1e-05 (the default
value).
x = 2x1
     2
     2

f = -4
```

Powyższy komunikat oznacza w szczególności, że procedura `intlinprog` wyznaczyła jedno rozwiązanie optymalne (ZPC): $\mathbf{x} = [2,2]^T$, dla którego wartość funkcji celu $f_C = -4$. Dodatkowa informacja to wartość optymalna zadania L_0: $f = -4,4$. □

5.4. Zadania do samodzielnego opracowania

5.4.1. Sprawozdanie

Zadanie 5.1. Dane jest (ZPC) określone w poniższym zestawie nr .... Stosując metodę podziału i ograniczeń, wyznaczyć wszystkie rozwiązania optymalne. Przedstawić pełny podział zadania L_0 na L_1, L_2 itd., tzn. nie dokonywać dalszego podziału tylko w przypadku, gdy zadanie L_k ma rozwiązanie całkowitoliczbowe lub jest sprzeczne. Jeżeli w rozwiązaniu optymalnym danego zadania L_k obie zmienne (x i y) przyjmują wartości niecałkowite, to dalszego podziału należy dokonać ze względu na zmienną x. Wyjaśnić, biorąc pod uwagę górne i dolne ograniczenia na wartość funkcji celu f_C w (ZPC), których zadań moglibyśmy nie dzielić. Do rozwiązania każdego zadania L_k zastosować standardową procedurę `linprog`. Porównać otrzymane wy-

niki z obliczeniami wykonanymi za pomocą standardowej procedury `intlinprog`. W rozwiązaniu podać:

- diagram przedstawiający podział zadania L_0 na L_1, L_2 itd.,
- zbiór ograniczeń dla każdego zadania L_k oraz ograniczenia górne i dolne na wartość f_C funkcji celu w (ZPC) (przedstawić w poniższej tabeli):

Zadanie	Postać zbioru ograniczeń	Ograniczenie górne lub dolne na wartość f_C
L_0		
L_1		
L_2		
$\vdots$		

- kod związany z wywołaniem procedury `intlinprog`.

Zestawy do zadania 5.1 (należy przyjąć dodatkowo, że $x, y \in \mathbb{Z}$):

1.

$$\max(-x+4y)$$
$$\begin{cases} x-8y+105 \geqslant 0 \\ 19x-8y-53 \geqslant 0 \\ -11x-8y+169 \leqslant 0 \\ x-8y+63 \leqslant 0 \\ 19x-8y-163 \leqslant 0 \\ -11x-8y+238 \geqslant 0 \end{cases}$$

2.

$$\max(-8y)$$
$$\begin{cases} 8x-8y+48 \geqslant 0 \\ -27x-8y+281 \leqslant 0 \\ -2x-8y+95 \leqslant 0 \\ 8x-8y-12 \leqslant 0 \\ -27x-8y+427 \geqslant 0 \\ -2x-8y+138 \geqslant 0 \end{cases}$$

3.

$$\max(6x-9y)$$
$$\begin{cases} x-8y+112 \geqslant 0 \\ 20x-8y-46 \geqslant 0 \\ -10x-8y+170 \leqslant 0 \\ x-8y+70 \leqslant 0 \\ 20x-8y-160 \leqslant 0 \\ -10x-8y+236 \geqslant 0 \end{cases}$$

4.

$$\max(7x-y)$$
$$\begin{cases} 20x-8y-69 \geqslant 0 \\ -11x-8y+173 \leqslant 0 \\ x-8y+62 \leqslant 0 \\ 20x-8y-182 \leqslant 0 \\ -11x-8y+240 \geqslant 0 \\ x-8y+103 \geqslant 0 \end{cases}$$

5.

$$\max(7x-8y)$$
$$\begin{cases} 17x-8y-11 \geqslant 0 \\ -12x-8y+162 \leqslant 0 \\ -8y+68 \leqslant 0 \\ 17x-8y-108 \leqslant 0 \\ -12x-8y+234 \geqslant 0 \\ -8y+109 \geqslant 0 \end{cases}$$

6.

$$\max(-6x-3y)$$
$$\begin{cases} 50x-8y-220 \geqslant 0 \\ -7x-8y+135 \leqslant 0 \\ 3x-8y+53 \leqslant 0 \\ 50x-8y-484 \leqslant 0 \\ -7x-8y+189 \geqslant 0 \\ 3x-8y+97 \geqslant 0 \end{cases}$$

7.

$$\max(2x+3y)$$
$$\begin{cases} 4x-8y+76 \geqslant 0 \\ 740x-8y-5845 \geqslant 0 \\ -5x-8y+125 \leqslant 0 \\ 4x-8y+28 \leqslant 0 \\ 740x-8y-9695 \leqslant 0 \\ -5x-8y+174 \geqslant 0 \end{cases}$$

8.

$$\max(-9x)$$
$$\begin{cases} 5x-8y+66 \geqslant 0 \\ -116x-8y+978 \leqslant 0 \\ -4x-8y+114 \leqslant 0 \\ 5x-8y+16 \leqslant 0 \\ -116x-8y+1579 \geqslant 0 \\ -4x-8y+160 \geqslant 0 \end{cases}$$

9.

$$\max(7x-2y)$$
$$\begin{cases} 3x-8y+73 \geqslant 0 \\ 112x-8y-847 \geqslant 0 \\ -6x-8y+127 \leqslant 0 \\ 3x-8y+27 \leqslant 0 \\ 112x-8y-1435 \leqslant 0 \\ -6x-8y+177 \geqslant 0 \end{cases}$$

10.

$$\max(3y)$$
$$\begin{cases} -36x-8y+332 \leqslant 0 \\ -3x-8y+97 \leqslant 0 \\ 7x-8y-1 \leqslant 0 \\ -36x-8y+520 \geqslant 0 \\ -3x-8y+141 \geqslant 0 \\ 7x-8y+56 \geqslant 0 \end{cases}$$

11.

$$\max(-6x+3y)$$
$$\begin{cases} 28x-8y-140 \geqslant 0 \\ -9x-8y+151 \leqslant 0 \\ 2x-8y+46 \leqslant 0 \\ 28x-8y-292 \leqslant 0 \\ -9x-8y+210 \geqslant 0 \\ 2x-8y+89 \geqslant 0 \end{cases}$$

12.

$$\max(8x)$$
$$\begin{cases} 102x-8y-570 \geqslant 0 \\ -6x-8y+124 \leqslant 0 \\ 3x-8y+40 \leqslant 0 \\ 102x-8y-1103 \leqslant 0 \\ -6x-8y+175 \geqslant 0 \\ 3x-8y+87 \geqslant 0 \end{cases}$$

13.

$$\max(-2x-5y)$$
$$\begin{cases} 13x-8y+6 \geqslant 0 \\ -15x-8y+190 \leqslant 0 \\ -x-8y+74 \leqslant 0 \\ 13x-8y-76 \leqslant 0 \\ -15x-8y+275 \geqslant 0 \\ -x-8y+116 \geqslant 0 \end{cases}$$

14.

$$\max(2x-6y)$$
$$\begin{cases} 17x-8y-28 \geqslant 0 \\ -12x-8y+170 \leqslant 0 \\ -8y+65 \leqslant 0 \\ 17x-8y-127 \leqslant 0 \\ -12x-8y+243 \geqslant 0 \\ -8y+107 \geqslant 0 \end{cases}$$

15.

$$\max(-2x-5y)$$
$$\begin{cases} 10x-8y+23 \geqslant 0 \\ -20x-8y+229 \leqslant 0 \\ -2x-8y+81 \leqslant 0 \\ 10x-8y-45 \leqslant 0 \\ -20x-8y+339 \geqslant 0 \\ -2x-8y+123 \geqslant 0 \end{cases}$$

16.

$$\max(-x-2y)$$
$$\begin{cases} 5x-8y+64 \geqslant 0 \\ -116x-8y+1052 \leqslant 0 \\ -4x-8y+117 \leqslant 0 \\ 5x-8y+14 \leqslant 0 \\ -116x-8y+1653 \geqslant 0 \\ -4x-8y+163 \geqslant 0 \end{cases}$$

17.

$$\max(8x+4y)$$
$$\begin{cases} 8x-8y+43 \geqslant 0 \\ -26x-8y+268 \leqslant 0 \\ -2x-8y+92 \leqslant 0 \\ 8x-8y-19 \leqslant 0 \\ -26x-8y+405 \geqslant 0 \\ -2x-8y+135 \geqslant 0 \end{cases}$$

18.

$$\max(-9x+4y)$$
$$\begin{cases} -129x-8y+1001 \leqslant 0 \\ -4x-8y+115 \leqslant 0 \\ 5x-8y+25 \leqslant 0 \\ -129x-8y+1673 \geqslant 0 \\ -4x-8y+162 \geqslant 0 \\ 5x-8y+74 \geqslant 0 \end{cases}$$

19.

$$\max(-2x-2y)\quad \begin{cases} 9x-8y+43 \geqslant 0 \\ -24x-8y+254 \leqslant 0 \\ -2x-8y+91 \leqslant 0 \\ 9x-8y-20 \leqslant 0 \\ -24x-8y+383 \geqslant 0 \\ -2x-8y+133 \geqslant 0 \end{cases}$$

20.

$$\max(9x+3y)\quad \begin{cases} 210x-8y-1390 \geqslant 0 \\ -6x-8y+120 \leqslant 0 \\ 4x-8y+31 \leqslant 0 \\ 210x-8y-2484 \leqslant 0 \\ -6x-8y+169 \geqslant 0 \\ 4x-8y+78 \geqslant 0 \end{cases}$$

21.

$$\max(-x-y)\quad \begin{cases} 7x-8y+41 \geqslant 0 \\ -35x-8y+348 \leqslant 0 \\ -3x-8y+92 \leqslant 0 \\ 7x-8y-16 \leqslant 0 \\ -35x-8y+531 \geqslant 0 \\ -3x-8y+136 \geqslant 0 \end{cases}$$

Zadanie 5.2. Dane jest (ZPC) określone w poniższym zestawie nr …. Stosując metodę cięć, wyznaczyć jedno z rozwiązań optymalnych. Do rozwiązania każdego zadania L_k zastosować standardową procedurę `linprog`.
W rozwiązaniu należy podać:

- postać dodatkowych ograniczeń dołączanych do kolejnych zadań L_k,
- rozwiązania optymalne zadań L_k (przedstawić w poniższej tabeli):

Zadanie	Rozwiązanie optymalne L_k	Wartość funkcji celu
L_0	$x^\star = \ldots,\ y^\star = \ldots$	
L_1	$x^\star = \ldots,\ y^\star = \ldots$	
L_2	$x^\star = \ldots,\ y^\star = \ldots$	
$\vdots$		

- rozwiązanie optymalne (ZPC).

Zestawy do zadania 5.2 (należy przyjąć dodatkowo, że $x \geqslant 0, y \geqslant 0, z \geqslant 0, t \geqslant 0$ oraz $x, y, z, t \in \mathbb{Z}$):

1.

$$\min(-x-y)\quad \begin{cases} 5x+3y+z = 15 \\ 3x+5y+t = 16 \end{cases}$$

3.

$$\min(-x-y)\quad \begin{cases} 7x+5y+z = 35 \\ 5x+7y+t = 36 \end{cases}$$

5.

$$\min(-x-y)\quad \begin{cases} 9x+7y+z = 63 \\ 7x+9y+t = 64 \end{cases}$$

2.

$$\min(-x-y)\quad \begin{cases} 6x+4y+z = 24 \\ 4x+6y+t = 25 \end{cases}$$

4.

$$\min(-x-y)\quad \begin{cases} 8x+6y+z = 48 \\ 6x+8y+t = 49 \end{cases}$$

6.

$$\min(-x-y)\quad \begin{cases} 10x+8y+z = 80 \\ 8x+10y+t = 81 \end{cases}$$

7.

$$\min(-x-y)\begin{cases}11x+9y+z=\ \ 99\\ 9x+11y+t=100\end{cases}$$

8.

$$\min(-x-y)\begin{cases}12x+10y+z=120\\ 10x+12y+t=121\end{cases}$$

9.

$$\min(-x-y)\begin{cases}13x+11y+z=143\\ 11x+13y+t=144\end{cases}$$

10.

$$\min(-x-y)\begin{cases}14x+12y+z=168\\ 12x+14y+t=169\end{cases}$$

11.

$$\min(-x-y)\begin{cases}15x+13y+z=195\\ 13x+15y+t=196\end{cases}$$

12.

$$\min(-x-y)\begin{cases}16x+14y+z=224\\ 14x+16y+t=225\end{cases}$$

13.

$$\min(-x-y)\begin{cases}17x+15y+z=255\\ 15x+17y+t=256\end{cases}$$

14.

$$\min(-x-y)\begin{cases}18x+16y+z=288\\ 16x+18y+t=289\end{cases}$$

15.

$$\min(-x-y)\begin{cases}19x+17y+z=323\\ 17x+19y+t=324\end{cases}$$

16.

$$\min(-x-y)\begin{cases}20x+18y+z=360\\ 18x+20y+t=361\end{cases}$$

17.

$$\min(-x-y)\begin{cases}21x+19y+z=399\\ 19x+21y+t=400\end{cases}$$

18.

$$\min(-x-y)\begin{cases}22x+20y+z=440\\ 20x+22y+t=441\end{cases}$$

19.

$$\min(-x-y)\begin{cases}23x+21y+z=483\\ 21x+23y+t=484\end{cases}$$

20.

$$\min(-x-y)\begin{cases}24x+22y+z=528\\ 22x+24y+t=529\end{cases}$$

21.

$$\min(-x-y)\begin{cases}25x+23y+z=575\\ 23x+25y+t=576\end{cases}$$

Bibliografia

Bonnans, J. F., Gilbert, J. C., Lemaréchal, C., Sagastizábal, C. A. (2006). *Numerical optimization*. Berlin: Springer-Verlag.

Branin Jr., F. H. (1972). Widely convergent method of finding multiple solutions of simultaneous nonlinear equations. *IBM Journal of Research and Development*, *16*(5), 504-522.

Findeisen, W., Szymanowski, J., Wierzbicki, A. (1980). *Teoria i metody obliczeniowe optymalizacji*. Warszawa: PWN.

Fletcher, R. (1987). *Practical methods of optimization*. New York: John Wiley & Sons.

Gill, P. E., Murray, W., Wright, M. H. (1981). *Practical optimization*. London, New York: Academic Press.

Hock, W., Schittkowski, K. (1981). Test examples for nonlinear programming codes. *Lecture Notes in Economics and Mathematical Systems*, *187*.

Kusiak, J., Danielewska-Tułecka, A., Oprocha, P. (2019). *Optymalizacja. Wybrane metody z przykładami zastosowań*. Warszawa: PWN.

MATLAB. (n.d.). https://www.mathworks.com.

Morè, J. J., Garbow, B. S., Hilstrom, K. E. (1981). Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, *7*(1), 17-41.

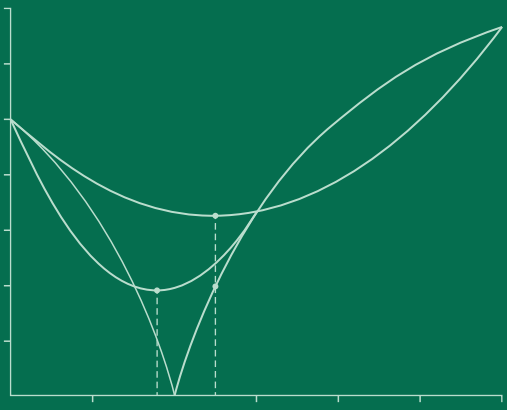
Neculai, A. (2008). An unconstrained optimization test functions collection. *Advanced Modeling and Optimization*, *10*(1), 147-161.

Nocedal, J., Wright, S. J. (2006). *Numerical optimization*. New York: Springer Nature.

Powell, M. J. D. (1978). A fast algorithm for nonlinearly constrained optimization calculations. *Numerical Analysis, Lecture Notes in Mathematics*, *630*, 144-157.

Stadnicki, J. (2017). *Teoria i praktyka rozwiązywania zadań optymalizacji z przykładami zastosowań technicznych*. Warszawa: PWN.

Zorychta, K., Ogryczak, A. (1981). *Programowanie liniowe i całkowitoliczbowe*. Warszawa: WNT.



Politechnika
Białostocka