Optymalizacja nieliniowa

Część I. Optima funkcji jednej zmiennej bez ograniczeń

- Wprowadzenie
- Metody bezpośredniego przeszukiwania
- Wstępne oszacowanie przedziału poszukiwań
- Metody gradientowe

WPROWADZENIE

Optymalizacja - dziedzina matematyki zajmująca się znajdowaniem minimów i maksimów wyrażenia nazywanego funkcją celu (lub kryterium jakośći, kryterium optymalizacji, funkcjonałem jakości).

Rys historyczny

- III wiek p.n.e (Euklides) - poszukiwanie najkrótszej drogi łączącej dwa punkty

- XVIII/XIX wiek (Gauss) - metoda największego spadku (pierwsza technika optymalizacyjna)

- pierwsza połowa XX wieku (Goerge B. Dantzig) - programowanie liniowe, algorytm sympleks („ojciec" terminu optymalizacja)

- druga połowa XX wieku - rozkwit teorii optymalizacji (stymulowany rozwojem technik komputerowych)

Ogólny sposób postępowania dla zagadnień optymalizacyjnych

- opracowanie modelu matematycznego analizowanego procesu
- zdefiniowanie funkcji celu (i ewentualnie ograniczeń)
- poszukiwanie optymalnego rozwiązania (zastosowanie jednej z metod optymalizacji)

Załóżmy, że dane są:

- funkcja celu $f\colon \mathbb{R}^n \to \mathbb{R}$,
- zbiór $X_d \subset \mathbb{R}^n$.

Zadaniem optymalizacji (programowania matematycznego) jest poszukiwanie takiego elementu $x^{opt} \in X_d$, że

$f(x^{opt}) \leqslant f(x)$ dla każdego $x \in X_d$ (poszukiwanie minimum)

lub

$f(x^{opt}) \geqslant f(x)$ dla każdego $x \in X_d$ (poszukiwanie maksimum)

Algorytmy optymalizacyjne (algorytmy numeryczne) wyznaczają pewne rozwiązanie x^*, które ma być przybliżeniem optimum globalnego, tzn. wartość $f(x^*)$ powinna być dobrym przybliżeniem wartości $f(x^{opt})$.

Uwaga. Znalezione optimum globalne (minimum albo maksimum) jest globalne jedynie (!) na zbiorze X_d. Może nie być to otimum globalne w całej dziedzinie.

Problemy z ekstremami lokalnymi

W sytuacjach praktycznych często w obszarze poszukiwań oprócz minimum (maksimum) globalnego występują również minima (maksima) lokalne, w których funkcja celu przyjmuje wartość „gorszą" niż w optimum globalnym.

Występowanie w obszarze poszukiwań więcej niż jednego ekstremum jest problematyczne, ponieważ algorytmy optymalizacyjne „grzęzną" w tych ekstremach lokalnych i nie docierają do ekstremum globalnego.

Uwaga. Wszystkie algorytmy optymalizacyjne znajdują zazwyczaj ekstrema lokalne (albo w najgorszym razie nie znajdują niczego).

Optima funkcji jednej zmiennej

Na potrzębę na dalszej części wykładu ograniczymy się teraz do przypadku jednowymiarowego. O funkcji celu $f : X \to \mathbb{R}$ będziemy zakładać, że jest ciągła. Uwarunkowane to jest tym, że znalezienie optimum funkcji nieciągłej jest trudne i wymaga stosowania złożonych algorytmów.

Przykład 1. *Dla funkcji*

$$f(x) = \begin{cases} x^2 + 3 & dla \quad x \neq 1 \\ 2 & dla \quad x = 1 \end{cases} ;$$

wszystkie podstawowe algorytmy, zastosowane do przedziału $[-1, 2]$ *wskażą minimum w punkcie* $x = 0$. *Nie jest to zgodne z prawdą, ponieważ minimum globalne w tym przedziale znajduje się w punkcie* $x = 1$.

W dalszej części wykładu ograniczymy się tylko i wyłącznie do zaprezentowania metod optymalizacyjnych służących do wyznaczenia minimum funkcji celu.

Uwaga. Rozwiązanie optymalne x^{opt} jest minimum globalnym funkcji celu f wtedy i tylko wtedy, gdy x^{opt} jest maksimum globalnym funkcji $-f$.

Warunki stopu

W zastosowaniach praktycznych metod optymalizacyjnych rzadkim jest przypadek, kiedy trafiamy dokładnie na rozwiązanie optymalne dające rzeczywiste minimum funkcji celu. Jest to spowodowane, m. in. ograniczoną dokładnością arytmetyki numerycznej z jakiej korzystamy w technikach komputerowych. Dlatego też musimy zadowolić się rozwiązaniem bliskim, w sensie przyjętych kryteriów, rozwiązaniu optymalnemu. Najczęściej stosowanymi warunkami stopu (kryteriami oceny jakości rozwiązania przybliżonego) są testy zbieżności:

- $\|x^{(i+1)} - x^{(i)}\|_2 \leqslant \delta$,
- $|f(x^{(i+1)}) - f(x^{(i)})| \leqslant \varepsilon$.

METODY BEZPOŚREDNIEGO PRZESZUKIWANIA
(metody bezgradientowe)

Algorytmy bezgradientowe

- korzystają jedynie ze znajomości wartości funkcji celu
- charakteryzują się względną prostotą
- nie wymagają obliczania wartości pochodnych funkcji celu (co może być dużą zaletą)
- zazwyczaj są wolno zbieżne (potrzebują większej liczby iteracji).

Funkcja unimodalna

Definicja 1. Funkcja ciagła jednej zmiennej jest *unimodalna*, gdy istnieje punkt c taki, że f jest malejąca dla $x < c$ i rosnąca dla $x > c$.

Uwaga. W praktyce wystarczy nam znajomość faktu, że minimalizowana funkcja jest unimodalna na przedziale.

Metoda przeszukiwania potrójnego

Niech dana będzie ciągła i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$.

Wybieramy dwa punkty $c, d \in (a, b)$ $(c < d)$ tak, aby dzieliły przedział $[a, b]$ na trzy równe części, tzn.

$$c = (2a + b)/3$$
$$d = (a + 2b)/3.$$

Zauważy, że wyjściowy przedział $[a, b]$ możemy zawęzić porównując wartości funkcji f w punktach c i d, mianowicie:

- jeżeli $f(c) < f(d)$, wówczas minimum leży w przedziale $[a, d]$
- jeżeli $f(c) > f(d)$, wówczas minimum leży w przedziale $[c, b]$
- jeżeli $f(c) = f(d)$, wówczas minimum leży w obu powyższych przedziałach (możemy dowolnie wybrać jeden z nich).

Powyższe rozumowanie możemy kontynuować dla tak zawężonego przedziału, w efekcie otrzymując jeszcze węższy przedział.

Załóżmy dalej, że oczekujemy, że obliczone minimum funkcji celu będzie oszacowane z dokładnością ε. Oznacza to, że jeżeli x^* przybliża dokładne minumum $x^{(opt)}$, to wymagamy aby

$$|x^{(opt)} - x^*| < \varepsilon.$$

Zatem opisany wyżej proces skracania początkowego przedziału, trzeba będzie wielokrotnie powtarzać do momentu kiedy krańce zawężonego przedziału $[a, b]$ będą spełniały warunek

$$b - a < 2\varepsilon.$$

Wtedy warunek ten spełniony jest przez punkt

$$x^* = (a + b)/2.$$

Metoda złotego podziału

Bardziej efektywnym sposobem doboru punktów pośrednich, które wyznaczają podział przedziału poszukiwań jest tzw. przeszukiwanie według złotego podziału. Idea metody opiera się na założeniu, że w każdym kroku obliczeń długość przedziału zmniejszana jest w stałym stosunku równym α, gdzie

$$\alpha = \frac{\sqrt{5}-1}{2} \approx 0.6180339 \quad \text{(złota liczba)}.$$

Przypomnienie. *Złota liczba* jest to dodatnia liczba niewymierna będąca rozwiązaniem równania $x^2 - x - 1 = 0$ równa

$$\varphi = \frac{1+\sqrt{5}}{2} \approx 1.6180339.$$

Czasami złotą liczbą określa się też liczbę odwrotną

$$\frac{1}{\varphi} = \frac{2}{1+\sqrt{5}} = \frac{\sqrt{5}-1}{2} = \varphi - 1 \approx 0.6180339.$$

Niech dana będzie ciągła i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$. W startowej (zerowej) iteracji przyjmujemy:

$$a^{(0)} = a \text{ i } b^{(0)} = b,$$

a następnie wybieramy dwa różne punkty $c^{(0)}, d^{(0)} \in [a^{(0)}, b^{(0)}]$ takie, że $c^{(0)} < d^{(0)}$.

Przedział poszukiwań ma zmniejszać się, z iteracji na iterację, w stałym stosunku α. Oznacza to, że w każdej iteracji, wybrane punkty pośrednie $c^{(i)}, d^{(i)}$ muszą dzielić i-ty przedział $[a^{(i)}, b^{(i)}]$ tak, aby spełniony był warunek

$$\frac{b^{(i)} - c^{(i)}}{b^{(i)} - a^{(i)}} = \frac{d^{(i)} - a^{(i)}}{b^{(i)} - a^{(i)}} = \alpha.$$

Z ostatniego warunku możemy wyznaczyć wartości punktów $c^{(i)}$ i $d^{(i)}$ w każdej iteracji, mamy

$$c^{(i)} = b^{(i)} - \alpha\left(b^{(i)} - a^{(i)}\right) = \alpha a^{(i)} + (1-\alpha)b^{(i)},$$
$$d^{(i)} = a^{(i)} + \alpha\left(b^{(i)} - a^{(i)}\right) = (1-\alpha)a^{(i)} + \alpha b^{(i)}.$$

Jeżeli w i-tej iteracji zachodzi

$$f(c^{(i)}) < f(d^{(i)}),$$

wówczas w następnej iteracji poszukiwania będą prowadzone w przedziale $[a^{(i)}, d^{(i)}]$. Będziemy wtedy mieli

$$a^{(i+1)} = a^{(i)},$$
$$b^{(i+1)} = d^{(i)},$$
$$c^{(i+1)} = \alpha a^{(i+1)} + (1-\alpha) b^{(i+1)},$$
$$d^{(i+1)} = (1-\alpha) a^{(i+1)} + \alpha b^{(i+1)} = (1-\alpha) a^{(i)} + \alpha d^{(i)} = \cdots = c^{(i)}.$$

Uwaga. Jak widać w $(i+1)$-szej iteracji jest potrzeba wyznaczenia tylko jednego (!) nowego punktu, tzn. $c^{(i+1)}$. Pozostałe punkty znamy z poprzedniej iteracji.

Analogiczna sytuacja ma miejsce, jeżeli w i-tej iteracji zachodzi

$$f(c^{(i)}) > f(d^{(i)}).$$

Wtedy w następnej iteracji poszukiwania będą prowadzone w przedziale $[c^{(i)}, b^{(i)}]$. Będziemy wtedy mieli

$$\begin{aligned}
a^{(i+1)} &= c^{(i)}, \\
b^{(i+1)} &= b^{(i)}, \\
c^{(i+1)} &= \alpha a^{(i+1)} + (1-\alpha) b^{(i+1)} = \alpha c^{(i)} + (1-\alpha) b^{(i)} = \cdots = d^{(i)}, \\
d^{(i+1)} &= (1-\alpha) a^{(i+1)} + \alpha b^{(i+1)}.
\end{aligned}$$

Uwaga. W tym przypadku w $(i+1)$-szej iteracji jest również potrzeba wyznaczenia tylko jednego nowego punktu, tzn. $d^{(i+1)}$. Pozostałe punkty są już znane z poprzedniej iteracji.

Istotność wyboru parametru α

Zwróćmy uwagę, że wybór wartości parametru α nie jest dowolny. Okazuje się, że powyższy algorytm może działać jedynie dla „złotej liczby".

Dla pierwszego przypadku, w $(i+1)$-szej iteracji mamy

$$\alpha = \frac{d^{(i+1)} - a^{(i+1)}}{b^{(i+1)} - a^{(i+1)}} = \frac{c^{(i)} - a^{(i)}}{d^{(i)} - a^{(i)}} = \cdots = \frac{1-\alpha}{\alpha}.$$

Stąd otrzymujemy równanie

$$\alpha^2 + \alpha - 1 = 0,$$

którego jedynym dodatnim rozwiązaniem jest złota liczba

$$\alpha = \frac{\sqrt{5}-1}{2}.$$

Dla drugiego przypadku sytuacja jest podobna.

Metoda Fibonacciego

Metoda Fibonacciego, podobnie jak dwie wcześniejsze, opiera się na idei zawężania początkowego przedziału poszukiwań. Wyjściowy przedział poszukiwań o długości L zawężamy aż do uzyskania przedziału finalnego długości ε, co odpowiada zakładanej dokładności obliczeń. Zawężanie odbywa się w oparciu o zależności dyktowane przez wyrazy ciągu Fibonacciego.

W metodzie tej stosunek długości $\frac{b^{(i+1)} - a^{(i+1)}}{b^{(i)} - a^{(i)}}$ nie jest stały, lecz zmienia się w każdej iteracji w zależności od kolejnych liczb w ciągu Fibonacciego.

Przypomnienie. Ciągiem Fibonacciego nazywamy ciąg liczb naturalnych ϕ_1, ϕ_2, ..., gdzie $\phi_1 = \phi_2 = 1$, a dalsze wyrazy są określone zależnością:

$$\phi_{n+2} = \phi_{n+1} + \phi_n,$$

czyli wyrazami ciągu Fibonacciego sa liczby:

$$1, 1, 2, 3, 5, 8, 13, \ldots \ .$$

Ciąg Fibonacciego możemy przedstawić również jawnie, wtedy

$$\phi_n = \frac{1}{\sqrt{5}} \left(\left[\frac{1+\sqrt{5}}{2} \right]^n - \left[\frac{1-\sqrt{5}}{2} \right]^n \right).$$

Na etapie wstępnym ustalamy początkowy przedział poszukiwań $[a, b]$, a następnie dla zadanej dokładności obliczeń ε szukamy najmniejszego wyrazu ciągu Fibonacciego ϕ_k, dla którego spełniona jest nierówność:

$$\phi_k > \frac{L}{\varepsilon}, \tag{1}$$

gdzie $L = b - a$.

Kolejny etap polega na iteracyjnym zmniejszaniu długości przedziału poszukiwań, aż do uzyskania zadanej dokładności. Zmniejszanie to odbywa się według następującej zasady.

W i-tej iteracji wewnątrz przedziału $[a^{(i)}, b^{(i)}]$ umieszczamy symetrycznie dwa punkty $c^{(i)}$ i $d^{(i)}$ tak, aby spełniona była zależność:

$$b^{(i)} - d^{(i)} = c^{(i)} - a^{(i)}. \tag{2}$$

Spełnianie tej zależności przez punkty wewnętrzne oznacza, że muszą być one równo oddalone od końców aktualnego przedziału poszukiwań.

Punkt wewnętrzny $c^{(i)}$ wybieramy w następujący sposób:

$$c^{(i)} = b^{(i)} - \beta^{(i)} \left(b^{(i)} - a^{(i)} \right),$$

gdzie $\beta^{(i)} = \frac{\phi_{k-i-1}}{\phi_{k-i}}$, a k jest numerem najmniejszego wyrazu ciągu Fibonacciego spełniającego nierówność (1).

Położenie punktu $d^{(i)}$ możemy wyznaczyć na podstawie zależności (2), mianowicie

$$d^{(i)} = a^{(i)} + b^{(i)} - c^{(i)}.$$

Ogólnie, w i-tej iteracji mamy obliczone punkty $c^{(i)} < d^{(i)}$. W celu wybrania odpowiedniego (zawężonego) przedziału do dalszych poszukiwań minimum, porównujemy wartości funkcji celu w tych punktach. Mogą wystąpić dwa przypadki.

Jeżeli $f(c^{(i)}) < f(d^{(i)})$, wówczas w następnej iteracji poszukiwania będą prowadzone w przedziale $[a^{(i)}, d^{(i)}]$. Będziemy wtedy mieli

$$a^{(i+1)} = a^{(i)}, \; b^{(i+1)} = d^{(i)}.$$

Jeżeli natomiast $f(c^{(i)}) \geqslant f(d^{(i)})$, wtedy w następnej iteracji poszukiwania będą prowadzone w przedziale $[c^{(i)}, b^{(i)}]$. Będziemy wtedy mieli

$$a^{(i+1)} = c^{(i)}, \; b^{(i+1)} = b^{(i)}.$$

W metodzie Fibonacciego z góry można ustalić liczbę iteracji potrzebnych do osiągnięcia zadanej dokładności optymalizacji.

W każdym kroku iteracji początkowy przedział jest skracany o odpowiedni ułamek. I tak w i-tej iteracji długość zaktualizowanego przedziału będzie wynosić:

$$b^{(i+1)} - a^{(i+1)} = \frac{\phi_{k-i-1}}{\phi_{k-i}} \left(b^{(i)} - a^{(i)} \right).$$

Jeżeli k jest numerem wyrazu w ciągu Fibonacciego, spełniającego warunek (1), wtedy po wykonaniu $k-2$ iteracji początkowy przedział będzie miał długość:

$$\frac{\phi_{k-1}}{\phi_k} \cdot \frac{\phi_{k-2}}{\phi_{k-1}} \cdot \frac{\phi_{k-3}}{\phi_{k-2}} \ldots \frac{\phi_3}{\phi_4} \cdot \frac{\phi_2}{\phi_3} \cdot (b-a) = \frac{b-a}{\phi_k}.$$

Na podstawie (1) mamy zatem

$$\frac{b-a}{\phi_k} < \varepsilon.$$

Oznacza to, że w $k-2$ iteracji aktualny przedział, w którym znjduje się minimum, ma długość mniejszą od zadanej dokładności ε.

Ponadto w ostatniej iteracji (tzn. dla $i = k - 3$, ponieważ zaczynamy iterowanie od $i = 0$) mamy

$$\beta^{(k-3)} = \frac{\phi_{k-(k-3)-1}}{\phi_{k-(k-3)}} = \frac{\phi_2}{\phi_3} = \frac{1}{2},$$

co implikuje, że

$$c^{(k-3)} = d^{(k-3)}.$$

Oznacza to jednocześnie, że ostatni zdublowany punkt wewnętrzny $c^{(k-3)}$ aktualnego przedziału poszukiwań jest jego środkiem. Możemy go zatem użyć jako dobre przybliżenie dokładnego minimum.

Uwaga. Należy wziąć pod uwagę fakt, że $k-2$ iteracje są wystarczające w przypadku, gdy obliczenia są ścisłe. W praktyce, ze względu na błedy zaokrągleń, nie zawsze daje się to uzyskać. Z tego powodu, aby zapewnić wymaganą dokładność, nieznacznie zwiększa się liczbę iteracji, np. o jedną.

Metoda oparta na interpolacji Lagrange'a

Metoda ta korzysta z faktu, że każdą funkcję możemy lepiej lub gorzej przybliżyć wielomianem stopnia drugiego. Metoda jest efektywna dla funkcji, których wykres jest zbliżony do paraboli. Dla innych funkcji będzie ona mało skuteczna albo wręcz rozbieżna.

W każdym kroku algorytmu obliczane są wartości funkcji celu w trzech punktach (w kroku zerowym są to zazwyczaj początek, środek i koniec przedziału poszukiwań). Przez te punkty prowadzony jest wielomian stopnia drugiego wyznaczany w oparciu o wzór interpolacyjny Lagrange'a. Minimum tego wielomianu wyznacza czwarty punkt pozwalający zawęzić przedział poszukiwań.

Niech dana będzie ciągła i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$. Załóżmy, że w i-tej iteracji mamy trzy punkty $a < c < b$ oraz że znamy wartości $f(a), f(c), f(b)$. Wyznaczamy wielomian drugiego stopnia przechodzący przez te punkty, mamy

$$w(x) = \sum_{k=1}^{3} f(x_k) \prod_{j=1, j\neq k}^{3} \frac{(x_j - x)}{(x_j - x_k)},$$

gdzie $x_1 = a, x_2 = c, x_3 = b$.

Wielomian ten osiąga minimum w wierzchołku paraboli, tzn. dla

$$d = \frac{1}{2} \cdot \frac{f(a)(c^2 - b^2) + f(c)(b^2 - a^2) + f(b)(a^2 - c^2)}{f(a)(c - b) + f(c)(b - a) + f(b)(a - c)}.$$

Uwaga. Jeżeli wyznaczony punkt d jest poza zbiorem $(a, c) \cup (c, b)$, wtedy algorytm nie jest zbieżny. Jest to spowodowane tym, że dla $d = c$ mamy tylko trzy punkty podziału i nie możemy zawęzić przedziału poszukiwań. A w pozostałych przypadkach wypadamy poza przedział poszukiwań.

Załóżmy, że $d \in (a, c) \cup (c, b)$. Jeżeli znamy wartość $f(d)$ możemy zawęzić przedział poszukiwań i powtórzyć procedurę. Zawężanie przedziału odbywa się w następujący sposób:

Jeżeli $a^{(i)} < d^{(i)} < c^{(i)}$ oraz $f(d^{(i)}) < f(c^{(i)})$, wówczas

$$\begin{aligned} a^{(i+1)} &= a^{(i)}, \\ c^{(i+1)} &= d^{(i)}, \\ b^{(i+1)} &= c^{(i)}. \end{aligned}$$

Jeżeli $a^{(i)} < d^{(i)} < c^{(i)}$ oraz $f(d^{(i)}) \geqslant f(c^{(i)})$, wówczas

$$\begin{aligned} a^{(i+1)} &= d^{(i)}, \\ c^{(i+1)} &= c^{(i)}, \\ b^{(i+1)} &= b^{(i)}. \end{aligned}$$

Jeżeli $c^{(i)} < d^{(i)} < b^{(i)}$ oraz $f(d^{(i)}) < f(c^{(i)})$, wówczas

$$\begin{aligned} a^{(i+1)} &= c^{(i)}, \\ c^{(i+1)} &= d^{(i)}, \\ b^{(i+1)} &= b^{(i)}. \end{aligned}$$

Jeżeli $c^{(i)} < d^{(i)} < b^{(i)}$ oraz $f(d^{(i)}) \geqslant f(c^{(i)})$, wówczas

$$\begin{aligned} a^{(i+1)} &= a^{(i)}, \\ c^{(i+1)} &= c^{(i)}, \\ b^{(i+1)} &= d^{(i)}. \end{aligned}$$

Jeżeli długość przedziału $[a^{(i)}, b^{(i)}]$ maleje w kolejnych iteracjach, to metoda jest zbieżna do minimum funkcji celu w przedziale $[a^{(0)}, b^{(0)}]$.

Zawężanie przedziału poszukiwań kontynuujemy w opisany sposób do momentu, aż długość tego przedziału będzie mniejsza od zadanej dokładności ε.

W opisanej metodzie sugeruje się rozszerzenie warunku stopu o dodatkowy składnik

$$|d^{(i)} - d^{(i-1)}| < \gamma.$$

Spowodowane jest to tym, że w niektórych sytuacjach zwężenia przedziału $[a^{(i)}, b^{(i)}]$ są nieznaczne, ponieważ punkty $d^{(i)}$ niewiele się zmieniają w kolejnych iteracjach. *Uwaga.* Współczynnik dokładności γ powinien być dużo mniejszy niż ε (np. 100 razy).

WSTĘPNE OSZACOWANIE PRZEDZIAŁU POSZUKIWAŃ

Zaprezentowane wyżej metody poszukują minimum lokalnego funkcji na zadanym przedziale. W sytuacjach w których wiemy, że funkcja celu posiada minimum na zbiorze $\mathbb{R}$ i chcemy je znaleźć, potrzebujemy odpowiedniego przedziału startowego. Zły (przypadkowy) wybór przedziału początkowego w procedurze optymalizacyjnej, może spowodować, że nie będzie ona zbieżna.

Przedstawione poniżej metody ekspansji służą do wstępnego określenia przedziału $[a, b]$, w którym znajduje się minimum funkcji unimodalnej.

Metoda ekspansji

Metoda rozpoczyna poszukiwania od punktu $x_0 = 0$. Startujemy podając dowolny punkt $x_1 > 0$ i współczynnik ekspansji $\alpha > 1$. Następnie obliczamy i porównujemy wartości funkcji f w punktach x_0 i x_1.

Możliwe są trzy przypadki:
1. Jeżeli $f(x_0) = f(x_1)$, to poszukiwane minimum znajduje się w przedziale $[x_0, x_1]$.
2. Jeżeli $f(x_0) > f(x_1)$, to kierunek poszukiwań minimum jest dobry i następny punkt wybieramy w tym samym kierunku, tzn.

$$x_2 = \alpha x_1.$$

Uwaga. Zauważmy, że musi być $\alpha > 1$ (!), aby zapewnić ekspansję kolejnych punktów (przyrost długości kolejnych kroków).

Jeżeli $f(x_1) \leqslant f(x_2)$, to kończymy poszukiwania ponieważ minimum musi znajdować się w przedziale $[x_0, x_2]$.

Jeżeli natomiast $f(x_1) > f(x_2)$, to poszukiwania kontynuujemy w tym samym kierunku, a kolejne punkty obliczmy według wzoru:

$$x_{i+1} = \alpha^i x_i,$$

gdzie $i = 2, 3, \ldots, m$ są numerami kolejnych kroków (iteracji). Poszukiwania kontynuujemy do momentu znalezienia takiego punktu dla którego spełniony będzie warunek:

$$f(x_i) \leqslant f(x_{i+1}),$$

ponieważ wtedy możemy stwierdzić, że minimum znajduje się w przedziale $[x_{i-1}, x_{i+1}]$.

3. Jeżeli $f(x_0) < f(x_1)$, to kierunek poszukiwań nie jest dobry i minimum należy poszukiwać w kierunku przeciwnym (na lewo od 0). Przyjmujemy zatem

$$x_1 := -x_1$$

Jeżeli wówczas

$$f(x_1) \geqslant f(x_0),$$

to następuje koniec poszukiwań, ponieważ minimum musi znajdować się w przedziale $[x_1, -x_1]$. W przeciwnym wypadku poszukiwania są prowadzone w kierunku ujemnym, analogicznie jak wyżej dla pkt. 2.

Metoda ekspansji Boxa-Daviesa-Swanna (BDS)

(dwupunktowa)

W zastosowaniach praktycznych ustalenie wstępnego przedziału poszukiwań w otoczeniu zera może nie być możliwe. Zasadna jest modyfikacja algorytmu ekspansji dla punktu początkowego różnego od zera.

Taką modyfikację zawiera algorytm zaproponowany przez Boxa, Daviesa i Swanna. Różnica w stosunku do wcześniejszego algorytmu ekspansji polega na tym, że startujemy z dowolnego punktu x_0, a współczynnik ekspansji jest stale równy 2.

Minimum funkcji może znajdować się zarówno na prawo, jak i na lewo od punktu x_0. Dlatego będziemy testować wartości funkcji celu po obu jego stronach.

Poszukiwania zaczynamy od sprawdzenia otoczenia prawostronnego punktu x_0, tzn. porównujemy wartości funkcji w punktach:

$$x_{i+1} = x_0 + 2^i \delta,$$

gdzie δ oznacza ustaloną z góry długość kroku, a $i = 0, 1, 2, \ldots$ Poszukiwanie prowadzimy tak długo, aż natrafimy na punkt w którym spełniony będzie warunek:

$$f(x_{i+1}) \geqslant f(x_i).$$

Punkt x_{i+1} wyznacza górną granicę przedziału poszukiwań, tzn. $b = x_{i+1}$.

Dolną granicę przedziału poszukiwań wyznaczamy w podobny sposób, przy czym w procesie iteracyjnym wykorzystujemy wzór:

$$x_{i+1} = x_0 - 2^i\delta, \qquad i = 0, 1, 2, \ldots \ .$$

Proces poszukiwania dolnej granicy prowadzimy do momentu znalezienia punktu w którym ponownie

$$f(x_{i+1}) \geqslant f(x_i).$$

Punkt ten wyznacza dolną granicę przedziału poszukiwań, tzn. $a = x_{i+1}$.

Metoda ekspansji Boxa-Daviesa-Swanna (BDS)
(trójpunktowa)

Startujemy od podania dowolnego punktu x_0 i długości kroku poszukiwań δ. Procedura rozpoczyna działanie od porównania wartości funkcji f w trzech punktach: $x_0 - \delta, x_0, x_0 + \delta$.

Możliwe są trzy przypadki:
1. Jeżeli w punkcie x_0 wartość funkcji jest mniejsza niż w pozostałych dwóch punktach, to w przedziale $[x_0 - \delta, x_0 + \delta]$ musi znajdować się poszukiwane minimum.
2. Jeżeli w punkcie x_0 wartość funkcji jest większa niż w pozostałych dwóch punktach, to w przedziale $[x_0 - \delta, x_0 + \delta]$ musi znajdować się maksimum lokalne - procedura przerywa działanie.
3. Ostatnim przypadkiem jest sytuacja kiedy wartość funkcji celu w punkcie x_0 leży pomiędzy wartościami w pozostałych punktach - tutaj rozpoczyna się ekspansja.

Ekspansja odbywa się w kierunku spadku wartości funkcji, tzn. w lewo lub w prawo na osi liczbowej. Załóżmy, że zachodzi

$$f(x_0 - \delta) < f(x_0) < f(x_0 + \delta).$$

Oznacza to, że kierunek poszukiwań minimum będzie ujemny (minimum znajduje się na lewo od x_0).

Wyznaczamy teraz kolejne trzy punkty, w których porównamy wartości funkcji celu. Punkty te wyznaczamy według zasady:

$$x_1 = x_0,$$
$$x_2 = x_0 - \delta,$$
$$x_3 = x_0 - 2 \cdot \delta.$$

Uwaga. Dwa punkty mamy już z poprzedniego etapu - musimy wyznaczyć tylko trzeci.

Ogólnie, jeżeli w i-tej iteracji ekspansji mamy trzy punkty

$$x_1^{(i)}, x_2^{(i)}, x_3^{(i)}$$

dla których zachodzi

$$f(x_3^{(i)}) < f(x_2^{(i)}) < f(x_1^{(i)}),$$

wtedy w kolejnej iteracji porównujemy wartości funkcji f w punktach

$$\begin{aligned} x_1^{(i+1)} &= x_2^{(i)}, \\ x_2^{(i+1)} &= x_3^{(i)}, \\ x_3^{(i+1)} &= x_0 - 2^i \cdot \delta. \end{aligned}$$

Uwaga. Dwa punkty i wartości funkcji f w tych punktach znamy z poprzedniej iteracji.

Jeżeli w którejś iteracji otrzymamy

$$f(x_3^{(k)}) > f(x_2^{(k)}),$$

to algorytm kończy działanie z sukcesem - minimum znajduje się w przedziale $[x_3^{(k)}, x_1^{(k)}]$.

Analogicznie postępujemy kiedy wartości funkcji maleją w kierunku dodatnim, tzn.

$$f(x_0 - \delta) > f(x_0) > f(x_0 + \delta).$$

Minimum musi znajdować się na prawo od x_0 - robimy ekspansję w prawo.

METODY GRADIENTOWE

Metody przedstawione wyżej, do swego działania, wykorzystywały jedynie wartości badanej funkcji. Algorytmy gradientowe do wyznaczania minimum funkcji celu potrzebują informacji o jej pochodnej.

Metody gradientowe są zazwyczaj szybciej zbieżne, ale ich ograniczeniem jest konieczność spełniania warunku różniczkowalności przez minimalizowaną funkcję.

Metoda bisekcji

Algorytm bisekcji, wykorzystywany do wyznaczania miejsc zerowych dowolnej funkcji, może być również stosowany do poszukiwania minimum funkcji celu.

Metoda bisekcji jest najprostszym algorytmem gradientowym. Poszukuje ona minimum lokalnego funkcji f na zadanym przedziale $[a, b]$.

Niech dana będzie różniczkowalna i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$. Algorytm bisekcji działa na zasadzie zawężania początkowego przedziału poszukiwań do momentu osiągnięcia takiej jego długości, która odpowiada zadanej dokładności ε.

Zawężanie odbywa się w następujący sposób. W bieżącym przedziale $[a, b\ ]$ wyznaczamy jego środek, tzn. punkt $m = \dfrac{a+b}{2}$ i obliczamy $f'(m)$.

Mogą wystąpić trzy przypadki:

- jeżeli $f'(m) < 0$, to minimum musi znajdować się w przedziale $[m, b]$;
- jeżeli $f'(m) > 0$, to minimum musi znajdować się w przedziale $[a, m]$;
- jeżeli $f'(m) = 0$, to sytuacja jest nieco bardziej złożona. Jeżeli dodatkowo wiemy, że funkcja jest wypukła, to możemy stwierdzić, że minimum znajduje się w punkcie m. Jeżeli tego nie wiemy, to możemy nieznacznie zaburzyć m i kontynuować obliczenia z tak skorygowanym punktem.

Następnie powtarzamy powyższe rozumowanie dla zaktualizowanego przedziału poszukiwań, aż do osiągnięcia zadanej dokładności.

Uwaga. Algorytm bisekcji do zawężenia przedziału poszukiwań, w każdej iteracji, potrzebuje tylko jednego punktu wewnętrznego - inaczej niż ma to miejsce dla metod bezgradientowych.

Uwaga. W metodzie bisekcji, w każdej iteracji przedział poszukiwań jest skracany o połowę. Algorytm wydaje się być zatem szybszy od poprzednio podanych. Tak rzeczywiście jest jeżeli znamy pochodną funkcji f i koszt obliczenia wartości pochodnej w punkcie m jest podobny do kosztu obliczenia wartości samej funkcji. Jeżeli natomiast pochodna jest wyznaczana numerycznie, to metoda może być mniej efektywna niż metody bezgradientowe.

Metoda Newtona

Metoda Newtona jest bardziej efektywną procedurą niż metoda bisekcji. Do jej stosowania, oprócz pierwszej pochodnej, wymagana jest znajomość również drugiej pochodnej funkcji celu.

W procedurze optymalizacyjnej Newtona wyznaczamy ciąg kolejnych przybliżeń, w którym każdy kolejny wyraz jest bliższy szukanego minimum.

Metoda opiera się na aproksymacji optymalizowanej funkcji wielomianem stopnia drugiego, wyznaczanym w oparciu o wzór Taylora.

Przypomnienie. Wzór Taylora jest ważnym wzorem dotyczącym funkcji z $C^n[a,b]$, którym często posługujemy się w analizie numerycznej.

Twierdzenie 1. *Jeżeli $f \in C^n[a,b]$ i $f^{(n+1)}$ istnieje w przedziale otwartym (a,b), to dla dowolnych punktów c i x z przedziału domkniętego $[a,b]$ mamy*

$$f(x) = \sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(c)(x-c)^k + E_n(x),$$

gdzie reszta $E_n(x)$ może być wyrażona wzorem

$$E_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi)(x-c)^{n+1},$$

dla pewnego punktu ξ leżącego między c i x.

Niech dana będzie dwukrotnie różniczkowalna i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$.

Zauważmy, że stosując wzór Taylora możemy, dla dowolnych $x, c \in [a, b]$, aproksymować funkcję f w następujący sposób

$$f(x) \approx f(c) + f'(c)(x - c) + \frac{1}{2} f''(c)(x - c)^2.$$

Zamieniając x na $x + h$ oraz c na x dostajemy

$$f(x + h) \approx f(x) + f'(x)h + \frac{1}{2} f''(x)h^2. \tag{3}$$

Uwaga. Krok h musi być tak dobrany, aby $x + h \in [a, b]$.

Niech $x^{(i)}$ będzie punktem (tzn. pewnym przybliżeniem rzeczywistego minimum) otrzymanym w i-tej iteracji procedury Newtona, wtedy na podstawie wzoru (3) mamy

$$f(x^{(i)} + h) \approx f(x^{(i)}) + f'(x^{(i)})h + \frac{1}{2}f''(x^{(i)})h^2.$$

W celu otrzymania kolejnego (lepszego) przybliżenia musimy wyznaczyć optymalną długość kroku h. Zauważmy, że prawa strona powyższej równości jest funkcją kwadratową zmiennej h. Jeżeli jest $f''(x^{(i)}) > 0$, to ta funkcja kwadratowa posiada minimum. Możemy zatem przybliżyć minimum naszej funkcji celu przez minimum tej funkcji kwadratowej, mamy

$$x^{(i+1)} = x^{(i)} + h = x^{(i)} - \frac{f'(x^{(i)})}{f''(x^{(i)})}.$$

Procedurę rozpoczynamy od podania punktu startowego $x^{(0)}$ (pierwszego przybliżenia) i dokładności obliczeń ε. Iterowanie kontynuujemy do momentu kiedy

$$\left|x^{(i)} - x^{(i-1)}\right| < \varepsilon.$$

Uwaga. Powyższy warunek stopu nie gwarantuje dokładności przybliżenia rozwiązania optymalnego. Mówi on jedynie, że korekta otrzymanego rozwiązania w stosunku do poprzedniego jest mała.

Uwaga. Metoda Newtona, podobnie jak metoda bisekcji, nie korzysta z informacji o wartości badanej funkcji.

Uwaga. Punkt do którego dąży ciąg kolejnych przybliżeń $(x^{(i)})$, to miejsce zerowe pochodnej funkcji f. W szczególności oznacza to, że algorytm Newtona może przybliżać maksimum lub punkt przegięcia badanej funkcji.

Uwaga. Zaletą algorytmu Newtona jest jego szybka zbieżność.

Uwaga. W metodzie Newtona kluczową rolę odgrywa dobry wybór punktu początkowego. Dla źle wybranego punktu algorytm może nie być zbieżny.

Metoda siecznych

Metoda siecznych stanowi modyfikację metody Newtona pozwalającą na uniknięcie konieczności wyznaczania drugiej pochodnej.

Niech dana będzie różniczkowalna i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$. Jeżeli założymy, że została wykonana przynajmniej jedna iteracja algorytmu, to dysponujemy dwoma punktami $x^{(i)}$ oraz $x^{(i-1)}$, a także wartościami pochodnej w tych punktach. Możemy zatem przybliżyć $f''(x^{(i)})$ za pomocą współczynnika kierunkowego siecznej przechodzącej przez punkty $(x^{(i)}, f'(x^{(i)}))$ oraz $(x^{(i-1)}, f'(x^{(i-1)}))$.

Mamy

$$f''(x^{(i)}) \approx \frac{f'(x^{(i)}) - f'(x^{(i-1)})}{x^{(i)} - x^{(i-1)}},$$

zatem w algorytmie Newtona, w i-tej iteracji, wzór na przybliżenie szukanego minimum przyjmuje postać

$$x^{(i+1)} = x^{(i)} - f'(x^{(i)}) \frac{x^{(i)} - x^{(i-1)}}{f'(x^{(i)}) - f'(x^{(i-1)})}.$$

Uwaga. Metoda siecznych w procesie optymalizacji wykorzystuje tylko pierwszą pochodną, ale do rozpoczęcia optymalizacji potrzebuje dwóch punktów startowych.

Reguła Armijo i metoda Newtona-Armijo

Algorytm Newtona stara się w maksymalnie szybko przejść do optimum badanej funkcji, co powoduje jego niestabilność. Spowodowane jest to tym, że w niektórych sytuacjach, krok prowadzący do kolejnego przybliżenia jest za duży i algorytm staje się rozbieżny.

Reguła Armijo podaje zasadę wyznaczania kroku d optymalizacji, który gwarantuje sukcesywne zmniejszanie się wartości funkcji.

Niech dana będzie różniczkowalna i unimodalna funkcja $f : \mathbb{R} \to \mathbb{R}$ dla której szukamy minimum w przedziale $[a, b]$. Jeżeli $x_0 \in [a, b]$ jest punktem w którym $f'(x_0) \neq 0$ i $\alpha \in (0, 1)$, wtedy zachodzi tzw. warunek Armijo:

$$\exists_d \quad f(x_0 + d) \leqslant f(x_0) + d\alpha f'(x_0).$$

Uwaga. Zauważmy, że warunek Armijo gwarantuje istnienie odpowiedniego kroku d, ale nie podaje sposobu jego wyznaczenia. W praktyce startujemy od wybranej wielkości kroku d i dokonujemy sprawdzenia czy zachodzi dla niego warunek Armijo. Jeżeli nie to zmniejszamy wartość kroku do ρd, gdzie $\rho \in (0, 1)$. Procedurę zmniejszania powtarzamy do momentu, aż warunek zostanie spełniony.

Metoda Newtona-Armijo

Metoda optymalizacyjna Newtona-Armijo jest prostą modyfikacją algorytmu Newtona wykorzystującą warunek Armijo. Modyfikacja dotyczy dwóch zmian:
1. długości kroku startowego w każdej iteracji procedury;
2. zasad akceptacji znalezionego punktu.

Ad. 1. Jeżeli druga pochodna minimalizowanej funkcji jest, w bieżącym przybliżeniu dodatnia, to startowa wartość kroku d z warunku Armijo wyznaczana jest, na początku każdej iteracji, zgodnie z algorytmem Newtona. Jeżeli druga pochodna dla bieżącego przybliżenia, nie jest dodatnia, to przyjmujemy długość kroku d równą 1 w kierunku spadku wartości badanej funkcji.

Uwaga. W przypadku funkcji jednej zmiennej, kierunek wzrostu wartości funkcji, wyznacza znak pochodnej w punkcie x.

Ad. 2. Procedura Newtona-Armijo oprócz informacji o pochodnej badanej funkcji, wykorzystuje również informację o jej wartościach. Mianowicie w każdym punkcie będącym kandydatem na kolejne przybliżenie szukanego minimum, badana jest wartość funkcji celu i jest on przyjmowany jedynie wtedy, gdy spełnia warunek Armijo.

Uwaga. Wprowadzone modyfikacje algorytmu skutkują tym, że jest on bardziej stabliny niż algorytm Newtona i zbiega do poszukiwanego minimum za każdym razem (tzn. dla każdego punktu początkowego).