Sprawozdanie

Metody numeryczne

Laboratorium 10.

Poszukiwanie minimum wartości funkcji metodą symulowanego wyżarzania

21.05.2020 r.

Aleksandra Rolka

Celem 10. laboratorium było znalezienie minimum funkcji metodą symulowanego wyżarzania.

1. Wstęp teoretyczny

Optymalizacja (minimalizacja) funkcji

Zadaniem optymalizacji jest poszukiwanie minimum lub maksimum funkcji (wielu zmiennych). W praktyce problem sprowadza się do poszukiwania minimum, czyli takiego punktu, dla którego zachodzi:

$$f: R^n \to R \tag{1}$$

$$min\, f(x) = f(x^*) \Leftrightarrow \bigwedge_{x \in R^n} f(x^*) < f(x) \tag{2}$$

$$x = [x_1, x_2, \dots, x_n]^T \tag{3}$$

z warunkami:

$$g_j(x) \leq 0, \quad j = 1, 2, \dots, m \tag{4}$$

$$h_j(x) \leq 0, \quad j = 1, 2, \dots, r \tag{5}$$

- $f(x), g(x), h(x)$ – funkcje skalarne
- $f(x)$ –funkcja celu, celem jest znalezienie jej minimum (optymalizacja)
- $g(x), h(x)$ –funkcje określające warunki jakie musi spełniać rozwiązanie (więzy) – ograniczają przestrzeń dopuszczalnych rozwiązań

Minimum lokalne, minimum globalne oraz punkt siodłowy funkcji celu

a) punkt x^* stanowi **minimum globalne** funkcji jeśli:

$$\bigwedge_{x \in R^n} \; f(x) \geq f(x^*) \tag{6}$$

b) punkt x^* stanowi **minimum lokalne** funkcji jeśli:

$$\exists \varepsilon: \varepsilon > 0, \varepsilon \in R \quad \bigwedge_{x: ||x - x^*|| < \epsilon} \; f(x) > f(x^*) \tag{7}$$

c) punkt $\boldsymbol{x^* = \begin{pmatrix} x^0 \\ y^0 \end{pmatrix}}$ jest **punktem siodłowym** funkcji jeśli:

$$\exists \varepsilon: \varepsilon > 0, \varepsilon \in R \quad \bigwedge_{\substack{x: ||x - x^0|| < \epsilon \\ y: ||y - y^0|| < \epsilon}} \; f(x, y^0) \leq \; f(x^0, y^0) \leq f(x^0, y) \tag{8}$$

Metoda symulowanego wyżarzania **(jedna z metod Monte Carlo, metoda stochastyczna)**

- **Symulowane wyżarzanie** – algorytm typu Monte Carlo: polega na wielokrotnym losowaniu przesunięć położeń wędrowców. Jeśli w nowym położeniu wędrowca funkcja f przyjmuje mniejszą wartość, to stare położenie wędrowca nadpisywane jest nowym (mówi się wtedy o akceptacji nowego położenia). W przeciwnym wypadku nowe położenie akceptowane jest tylko z pewnym prawdopodobieństwem, zadanym przez rozkład Boltzmanna: przy wysokiej temperaturze jest duże prawdopodobieństwo akceptacji "gorszego" położenia, przy niskiej – niewielkie.
  Sposób działania symulowanego wyżarzania przypomina zjawisko wyżarzania w metalurgii.

Prosty algorytm symulowanego wyżarzania – iteracyjny:

Na początku wybierany jest punkt startowy x_0 i określana jest wartość funkcji celu $f(x_0)$ dla danego wędrowca, ustalana jest również temperatura T oraz ilość kroków M jaką ma wykonać:

1) wylosowanie przemieszczenia Δx generatorem liczb pseudolosowych i określenie $f(x_i + \Delta x)$

2) nowe położenie akceptowane z prawdopodobieństwem $P = 1$, jeśli:

$$f(x_i + \Delta x) \leq f(x_i) \quad \Rightarrow \quad x_{i+1} = x_i + \Delta x \tag{9}$$

w przeciwnym wypadku:

$$f(x_i + \Delta x) > f(x_i) \tag{10}$$

określając prawdopodobieństwo:

$$P = \exp\left(-\frac{f(x_i+\Delta x)-f(x_i)}{T}\right) \tag{11}$$

i losowana jest liczba:

$$X \in U(0,1), \tag{12}$$

nowe położenie akceptowane (z prawdopodobieństwem P), jeśli spełniona jest relacja:

$$X < P \tag{13}$$

3) co M iteracji obniżana jest temperatura, np.:

$$T_{new} = C \cdot T_{old}, \qquad C < 1 \tag{14}$$

co efektywnie zmniejsza prawdopodobieństwo akceptacji położenia o wyższej funkcji kosztu

- kroki 1-3 powtarza się M razy dla każdego wędrowca
- symulację można wykonać dla N wędrowców równolegle – są niezależni
- po jej zakończeniu wyszukiwany jest wędrowca o najniższej funkcji celu

2. Zadanie do wykonania

2.1 Opis problemu

Zadanie: znaleźć minimum funkcji:

$$f(x,y) = \sin(x)\sin(y) - \exp\left(-\left(x+\frac{\pi}{2}\right)^2 - \left(y-\frac{\pi}{2}\right)^2\right) \quad (15)$$

przy pomocy metody symulowanego wyżarzania ze zmienną temperaturą T zaprogramowanej dla N wędrowców. Każdy wędrowiec ma w danej chwili współrzędne (x_i, y_i) (indeks i oznacza numer wędrowca, $i = 0, 1, \ldots, N-1$).

Schemat do wykorzystania w programie:

1. wybór współrzędnych początkowych $(x^{(0)}, y^{(0)})$, przypisanie ich wszystkim wędrowcom,
2. $for\ (iT = 0;\ iT <= 20;\ iT = iT + 1)$ // pętla zewnętrzna obniżająca temperaturę
3. $T = \frac{10}{2^{iT}}$ // ustalenie aktualnej temperatury
4. $for\ (k = 0;\ k < 100;\ k = k + 1)$ // 100 kroków…
5. $for\ (i = 0;\ i < N;\ i = i + 1)$ // ...dla każdego z wędrowców
6. $\Delta x_i = d_{rand}(-1, 1)$ // losowanie przesunięcia Δx_i , Δy_i
7. $\Delta y_i = d\_rand(-1, 1)$ // … z przedziału [−1, 1]
8. $if\ (\ (f\ (x_i + \Delta x_i, y_i + \Delta y_i) < f\ (x_i, y_i)\ )$ // jeżeli nowe położenie jest lepsze..
9. $x_i = x_i + \Delta x_i$ // ...jest akceptowane
10. $y_i = y_i + \Delta y_i$
11. $else\ if\ \left(d_{rand}(0,1) < exp\left(-\frac{f(x_i+\Delta x_i, y_i+\Delta y_i) - f(x_i,y_i)}{T}\right)\right)$
12. $x_i = x_i + \Delta x_i$ // gorsze położenie akceptowane z prawdopodobieństwem
13. $y_i = y_i + \Delta y_i$ // danym przez rozkład Boltzmanna
14. po zakończeniu algorytmu wyszukiwany zostaje wędrowca, dla którego wartość funkcji f jest najmniejsza – jego położenie uznane za znalezione minimum.

d_{rand} − zwraca pseudolosową liczbę zmiennoprzecinkową z przedziału podanego przez argumenty.

Na podstawie powyższego pseudokodu napisano program w języku C znajdujący minimum funkcji (15). Program ten nie korzysta z żadnych dodatkowych bibliotek numerycznych.

Przyjęto następujące wartości parametrów:

- liczba wędrowców $N = 200$
- współrzędne początkowych $(x^{(0)}, y^{(0)}) = (5,5)$
- współrzędne początkowych $(x^{(0)}, y^{(0)}) = (5,5)$
- obszar przeszukiwań minimum $[x_{min}, x_{max}] \times [y_{min}, y_{max}] = [-10, 10] \times [-10, 10]$

Ponadto:

- na wykresie konturowym funkcji (15) nanieść aktualne położenia wędrowców po zakończeniu błądzenia dla trzech przypadków temperatur:

  - ✓ $i_T = 0$ (wówczas $T = 10$),
  - ✓ $i_T = 7$ ($T = 0.078125$)
  - ✓ oraz po zakończeniu algorytmu, tj. dla $i_T = 20$ ($T \approx 0.0000095$).

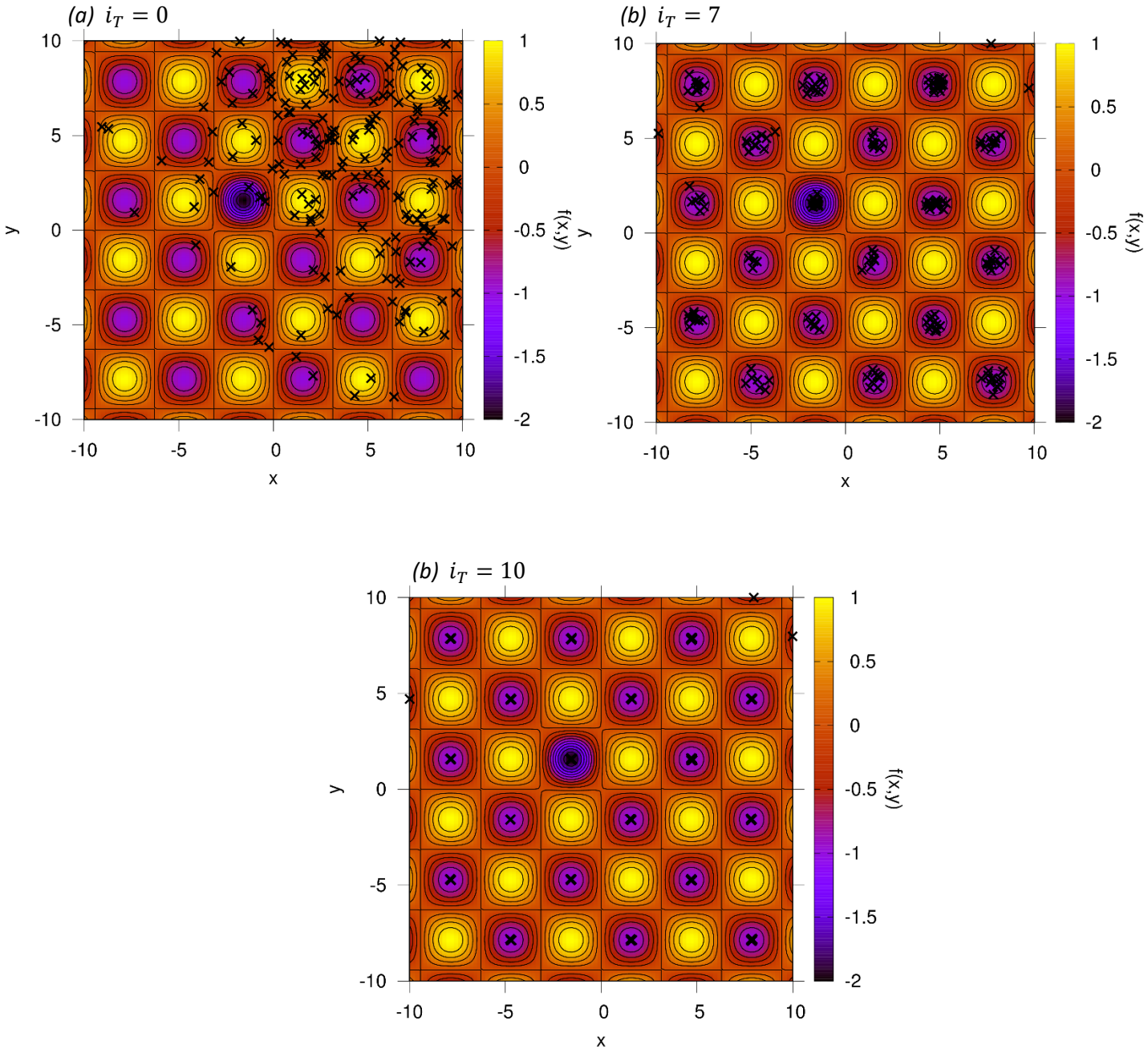
- dla jednego z wędrowców (np. dla pierwszego) wypisać i narysować wartości funkcji $f(x_i, y_i)$ dla wszystkich jego położeń w trakcie działania całego algorytmu

2.2 Wyniki

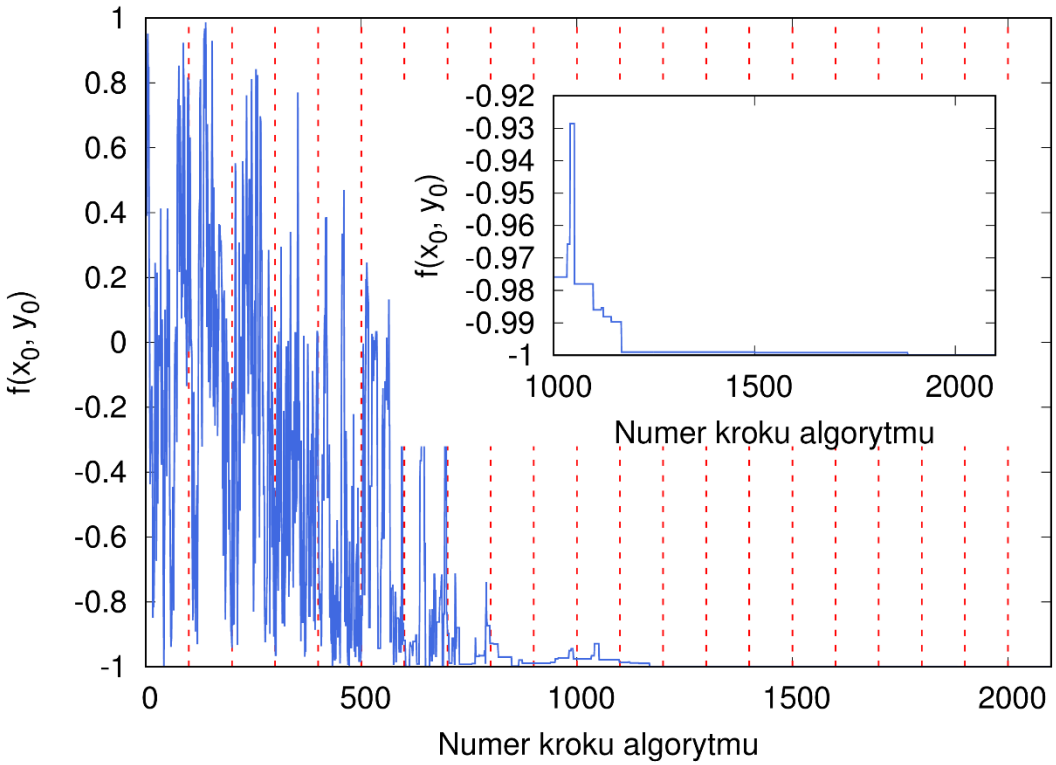
Otrzymane wyniki obliczeń zapisane zostały w plikach **.dat.* Kolejno:

- *w0.dat* - wartość funkcji dla pierwszego wędrowca w każdym z kroków algorytmu,
- *T.dat* - położenia x_i, y_i wszystkich wędrowców dla przypadku i_T *= 0,* i_T *= 7* oraz i_T *= 20*

Następnie na podstawie tych danych za pomocą skryptu Gnuplot'a wygenerowano poniższe wykresy:



Rysunek 1. *Położenia wszystkich wędrowców w trakcie działania algorytmu po zakończeniu błądzenia dla zadanego numeru i_T. Po wykonaniu działania całego algorytmu znaleziono minimum: (x_{min}, y_{min}) ≈ (-1.575539, 1.563222), dla którego $f(x_{min}, y_{min})$ ≈ -1.99988.*



Rysunek 2. *Wartości funkcji f (x, y) dla wszystkich położeń pierwszego wędrowca w trakcie trwania algorytmu. Czerwone, przerywane linie oznaczają miejsca zmiany temperatury (co 100 kroków). Wstawiony wykres: zbliżenie dla końcowych kroków.*

3. Wnioski

Znalezione minimum funkcji ≈ *-1.99988.* jest minimum globalnym o położeniu (x_{min}, y_{min}) ≈ *(-1.575539, 1.563222)*. Przy odpowiedniej ilości wędrowców minimum globalne jest poprawnie wyznaczone. Dla próby, uruchomiono program tylko dla jednego wędrowca, niestety wynik nie był ani trochę zbliżony, nie był on minimum globalnym. Równie ważnym elementem symulacji jest zmniejszanie temperatury. Przy uruchomieniu programu dla stałej temperatury widoczne było, że wyniki nie ulegają polepszeniu, wędrowcy nie zbiegają się w położenia minimów.

Źródła:

- Dr hab inż. Tomasz Chwiej – Notatki do wykładu *„Optymalizacja (minimalizacja) funkcji”*. Dostępne w Internecie: http://galaxy.agh.edu.pl/~chwiej/mn/minimalizacja_1819.pdf