KI LaTeX DOKUMENT
Materiały do przedmiotu "Rozwiązywanie zadań odwrotnych"

Metody gradientowe - opis zagadnień związanych z klasycznymi algorytmami optymalizacyjnymi - metoda Newton-CG

dr inż. Konrad M. Gruszka,*

Abstract. Ten dokument prezentuje ogólny opis "krok po kroku" działanie algorytmu gradientowego *Newton-CG* zaprojektowanego do optymalizacji rozkładu temperatur w jednowymiarowym, stacjonarnym modelu transferu ciepła z ustalonymi warunkami brzegowymi.

1 Wprowadzenie

Metoda Newton-CG (Newton-Conjugate Gradient) to iteracyjna metoda optymalizacji, stosowana do znajdowania minimów funkcji nieliniowych. Jest wariantem metody Newtona, który zamiast bezpośredniego rozwiązywania układu równań liniowych dla Hesjana, wykorzystuje metodę sprzężonych gradientów (Conjugate Gradient, CG) do efektywnego rozwiązywania układów równań w dużych wymiarach.

2 Idea metody

Metoda Newtona-CG opiera się na drugim rzędzie rozwinięcia Taylora **funkcji kosztu**. Funkcja kosztu to funkcja matematyczna, która mierzy "koszt" błędu związany z danym zestawem parametrów w problemie optymalizacyjnym. Funkcja kosztu może np mierzyć różnicę między obliczonym rozkładem temperatury a rzeczywistym (docelowym) rozkładem. Inaczej mówiąc metoda Newtona opiera się na przybliżeniu funkcji w pobliżu punktu x_k przy użyciu rozwinięcia w szereg Taylora drugiego rzędu:

$$f(x) \approx f(x_k) + \nabla f(x_k)^T (x - x_k) + \frac{1}{2}(x - x_k)^T H(x_k)(x - x_k) \tag{1}$$

Minimalizując powyższe wyrażenie, uzyskujemy iteracyjny krok aktualizacji:

$$x_{k+1} = x_k - H(x_k)^{-1} \nabla f(x_k) \tag{2}$$

gdzie:
$\nabla f(x_k)$ – gradient funkcji kosztu (kierunek największego spadku),

* Katedra Informatyki, Wydział Inżynierii Mechanicznej i Informatyki (kgruszka@icis.pcz.pl)

$H(x_k)$ – macierz Hesjana (informacja o drugich pochodnych).

Bezpośrednie wyliczanie $H(x_k)^{-1}$ wymaga obliczenia macierzy odwrotnej, co dla dużych układów jest kosztowne. Zamiast tego można rozwiązać układ równań liniowych:

$$H(x_k)p_k = -\nabla f(x_k) \tag{3}$$

dla poszukiwanego kierunku p_k. Metoda Newton-CG wykorzystuje do tego metodę sprzężonych gradientów (CG), która pozwala na efektywne rozwiązanie układu bez jawnego obliczania odwrotności Hesjana.

3 Zastosowanie Newton-CG do problemu odwrotnego w 1D transferze ciepła

Chcemy znaleźć nieznane warunki brzegowe (T_{left},T_{right}), tak aby rozkład temperatury T(x) w jednorodnym materiale odpowiadał zadanemu, rzeczywistemu profilowi T_{target}. Model przewodnictwa cieplnego (równanie Laplace'a dla stanu ustalonego w 1D):

$$\frac{d^2T}{dx^2} = 0$$

z warunkami brzegowymi:

$$T(0) = T_{\text{left}}, \quad T(L) = T_{\text{right}}.$$

Jak Państwo wiecie, jego rozwiązanie analityczne w jednorodnym materiale:

$$T(x) = T_{\text{left}} + \frac{(T_{\text{right}} - T_{\text{left}})}{L}x$$

W naszym przypadku nie znamy T_{left} i T_{right}, ale mamy pewien docelowy rozkład temperatury $T_{target}(x)$.

3.1 Definicja funkcji kosztu

Definiujemy funkcję kosztu jako normę różnicy między symulowanym a rzeczywistym rozkładem temperatury:

$$J(T_{\text{left}}, T_{\text{right}}) = ||T_{\text{model}} - T_{\text{target}}||_2 \tag{4}$$

Powyższa norma (zwana normą euklidesową L_2 dla dwóch wektorów T_{model} oraz T_{target} oblicza się jako:

$$||T_{\text{model}} - T_{\text{target}}||_2 = \sqrt{\sum_{i=1}^{N}(T_{\text{model},i} - T_{\text{target},i})^2} \tag{5}$$

gdzie:

- N – liczba punktów pomiarowych w siatce,
- $T_{model,i}$ – temperatura obliczona w punkcie i,
- $T_{target,i}$ – docelowa temperatura w punkcie i.

Norma L_2 odpowiada średniokwadratowemu błędowi dopasowania, czyli mierzy, jak bardzo obliczony profil temperatury odbiega od rzeczywistego.
Metoda Newton-CG próbuje znaleźć takie wartości T_{left} i T_{right}, które minimalizują błąd $J(T_{left}, T_{right})$, co oznacza:

$$\arg \min_{T_{\text{left}}, T_{\text{right}}} J(T_{\text{left}}, T_{\text{right}}) \tag{6}$$

czyli szukamy takich warunków brzegowych, które najlepiej odwzorują rzeczywisty rozkład temperatury.

3.2 Obliczanie gradientu numerycznie

Gradient wyznaczamy numerycznie przy użyciu różnic skończonych:

$$\frac{\partial J}{\partial T_{\text{left}}} \approx \frac{J(T_{\text{left}} + \epsilon, T_{\text{right}}) - J(T_{\text{left}} - \epsilon, T_{\text{right}})}{2\epsilon} \tag{7}$$

Tutaj ϵ oznacza mały krok różniczkowy, w praktyce jest to wartość rzędu 10^{-5}.

3.3 Hesjan

Hesjan (macierz Hessego) to macierz drugich pochodnych cząstkowych funkcji skalarnej wielu zmiennych. W matematyce i optymalizacji służy do analizy wypukłości funkcji i oceny zachowania jej drugich pochodnych.
Dla funkcji $f : \mathbb{R}^n \to \mathbb{R}$ Hesjan definiuje się jako:

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Jak obliczyć Hesjan numerycznie? Hesjan można oszacować numerycznie przy użyciu skończonych różnic. Powszechnie stosowana formuła to:

$$H_{i,j} \approx \frac{f(x + e_i + e_j) - f(x + e_i) - f(x + e_j) + f(x)}{\epsilon^2}$$

gdzie:
e_i i e_j to wektory bazowe zaburzeń w odpowiednich wymiarach,
ε to ponownie mały krok różniczkowy.

4 Implementacja algorytmu

Metoda newton-CG jest zainmplementowana w bibliotece *scipy*. Aby z niej skorzystać wystarczy zaimportować **minimize** z *scipy.optimize*

Aby zrealizować metodę należy postępować zgodnie z poniższym schematem działania

(1) Wygeneruj docelowy rozkład temperatury
   - wykorzystaj wcześniej napisaną funkcję *simulate_heat_transfer(N, T0, TN, max iter, tolerance=None)* dla jednorodnego, jednowymiarowego przypadku

(2) Określ funkcję kosztu $J(T_{left}, T_{right})$ *def cost_function(boundary_conditions):* która zwróci normę różnicy T_{model} -T_{target} (patrz np.linalg.norm())

(3) Oblicz gradient funkcji kosztu *def numerical_gradient(f, x, epsilon=1e-5)*

(4) Oblicz Hesjan funkcji kosztu *def numerical_hessian(f, x, epsilon=1e-5):*

Funkcja gradientu:

$$\frac{\partial J}{\partial x_i} \approx \frac{J(x + \epsilon e_i) - J(x - \epsilon e_i)}{2\epsilon} \tag{8}$$

Hesjan funkcji kosztu aproksymowany metodą różnic skończonych:

$$H_{i,j} = \frac{J(x + e_i + e_j) - J(x + e_i) - J(x + e_j) + J(x)}{\epsilon^2} \tag{9}$$

Aby stworzyć funkcję **numerical_hessian(f, x, epsilon=1e-5)** numerycznie aproksymującą Hesjan funkcji f w punkcie x używając metody skończonych różnic, należy krok po kroku:

(1) **Inicjalizacja**: Utworzyć pustą macierz hessian o wymiarze $n \times n$ (gdzie n to liczba zmiennych w x: n=len(x)).

(2) **Utworzyć macierz perturbacji**: *perturb* to macierz jednostkowa I (patrz np.eye()) pomnożona przez ϵ, co pozwala na łatwe modyfikowanie każdej współrzędnej x.

(3) **Podwójna pętla** po indeksach (i, j):
   - Obliczyć wartości funkcji w różnych kombinacjach przesunięć:
     - $f(x + e_i + e_j)$ - wartość funkcji przy zaburzeniu dwóch zmiennych.
     - $f(x + e_i)$ - wartość funkcji przy zaburzeniu tylko jednej zmiennej.
     - $f(x + e_j)$ - wartość funkcji przy zaburzeniu drugiej zmiennej.
     - $f(x)$ - wartość funkcji w oryginalnym punkcie. Obliczyć element macierzy Hesjanowej według wzoru skończonych różnic.
     - Zwrócić macierz Hesjana.

(4) Optymalizuj metodą **Newton-CG** z *scipy*

(5) Odtwórz warunki brzegowe

(6) Wizaulizuj i wyświetl wyniki

5 Podsumowanie

W niniejszym opracowaniu przedstawiono metodę gradientową Newton-CG. Implementacja sprowadziła się do:

- Rozwiązywania przewodnictwa ciepła – symulacja stacjonarnego przepływu ciepła metodą różnic skończonych.
- Generowania docelowego rozkładu temperatury.
- Definiowania funkcji kosztu jako różnicy między modelem a rzeczywistością.
- Obliczenia gradientu i Hesjana numerycznie.
- Optymalizacji metodą Newtona-CG, aby odtworzyć rzeczywiste warunki brzegowe.
- Porównania wyników wizualnie i numerycznie.