

Metody Numeryczne N07

Jakub Kurek

1. Wstęp

Celem zadania jest znalezienie rozwiązania układu równań:

$$\begin{cases} -\frac{u_{n-1}-2u_n+u_{n+1}}{h^2} + 2u_n(u_n^2 - 1) = 0 & n = 1 \dots N-1 \\ u_0 = 0 \\ u_N = 1 \end{cases}$$

gdzie $h = \frac{20}{N-1}$.

Program liczący rozwiązania został napisany w C++23 przy użyciu biblioteki *Eigen* w wersji 5.0.1. Wykresy zostały stworzone w języku Python przy użyciu biblioteki matplotlib. Cały kod wykorzystywany do obliczeń i rysowania wykresów znajduje się w repozytorium na *GitHub*.

2. Wielowymiarowa metoda Newtona

Niech funkcja $g : \mathbb{R}^N \rightarrow \mathbb{R}^N$ będzie funkcją klasy co najmniej C^1 (wszystkie pochodne cząstkowe są funkcjami ciągłymi zmiennej x). Rozważamy równanie:

$$g(x) = 0$$

W zależności od parametrów układ równań może mieć różną ilość rozwiązań.

Rozwijając funkcję g w szereg Taylora do pierwszego rzędu otrzymujemy:

$$g(x + \delta x) \simeq g(x) J \delta x$$

gdzie J jest jacobianem funkcji g :

$$J(x)_{ij} = \partial \frac{g_i}{\partial x_j} |_x$$

Aby znaleźć się w punkcie spełniającym równanie $g(x + \delta x) = 0$:

$$\delta x = -J^{-1}g(x)$$

Prowadzi to do iteracji:

$$x_{k+1} = x_k - J^{-1}(x_k)g(x_k)$$

Jacobian trzeba obliczać w każdym kroku iteracji. (rozwiązanie innego układu równań liniowych).

Zamiast rozwiązywać układ równań algebraicznych można rozwiązać funkcję $G : \mathbb{R}^n \rightarrow \mathbb{R}$, ponieważ minimalizacja jest łatwiejsza:

$$G(x) = \frac{1}{2} \|g(x)\|^2 = \frac{1}{2} (g(x))^T g(x)$$

Globalne minimum funkcji $G = 0$ odpowiada rozwiązaniu układu równań, jednak funkcja G może mieć wiele minimów lokalnych (lub globalne minimum może nie istnieć)

Rozwiązujeć równanie $g(x) = 0$ metodą Newtona krok iteracji wynosi:

$$\delta x = -J^{-1}g$$

oraz mamy:

$$\frac{\partial G}{\partial x_i} = \sum_j J_{ji} g_j$$

więc:

$$\nabla G = J^\top g$$

Funkcja G po wykonaniu kroku Newtona:

$$(\nabla G)^T \delta x = g^T J(-J^{-1})g = -g^T g < 0$$

Co pokazuje, że kierunek kroku Newtona jest lokalnym kierunkiem spadku G . Jednak przesunięcie się o pełną długość kroku Newtona nie musi prowadzić do spadku G . Postępujemy więc:

Obliczamy δx oraz ustawiamy wagę początkową $w = 1$.

Dopóki nie znajdziemy poprawy ($G(x_{\text{test}}) < G(x_i)$):

- $x_{\text{test}} = x_i + w\delta x$
- Jeśli $G(x_{\text{test}}) < G(x_i)$:
 - ▶ $x_{i+1} = x_{\text{test}}$
 - ▶ Przechodzimy do kolejnej iteracji
- W przeciwnym przypadku:
 - ▶ $w = w * \alpha$ gdzie $\alpha < 1$
 - ▶ Wracamy do obliczania x_{test} .

Metoda ta jest zawsze zbieżna do jakiegoś minimum funkcji G , ale niekoniecznie do jej minimum globalnego. Jeśli znajdziemy minimum lokalne $G_{\min} > \varepsilon$, rozpoczynamy metodę z innym warunkiem początkowym. Jeśli kilka warunków początkowych nie daje pożdanego rozwiązania należy się poddać.

Im lepszy warunek początkowy tym szansa na znalezienie rozwiązania jest większa. Należy zainwestować naszą wiedzę o funkcji g w jego znalezienie.

2.1. Jakobian

W naszym przyadku Jakobian pochodnych cząstkowych ma postać trójdiagonalną, gdyż nieliniowy składnik zależy tylko od kolejnych 3 zmiennych (inne pochodne cząstkowe będą się zerować).

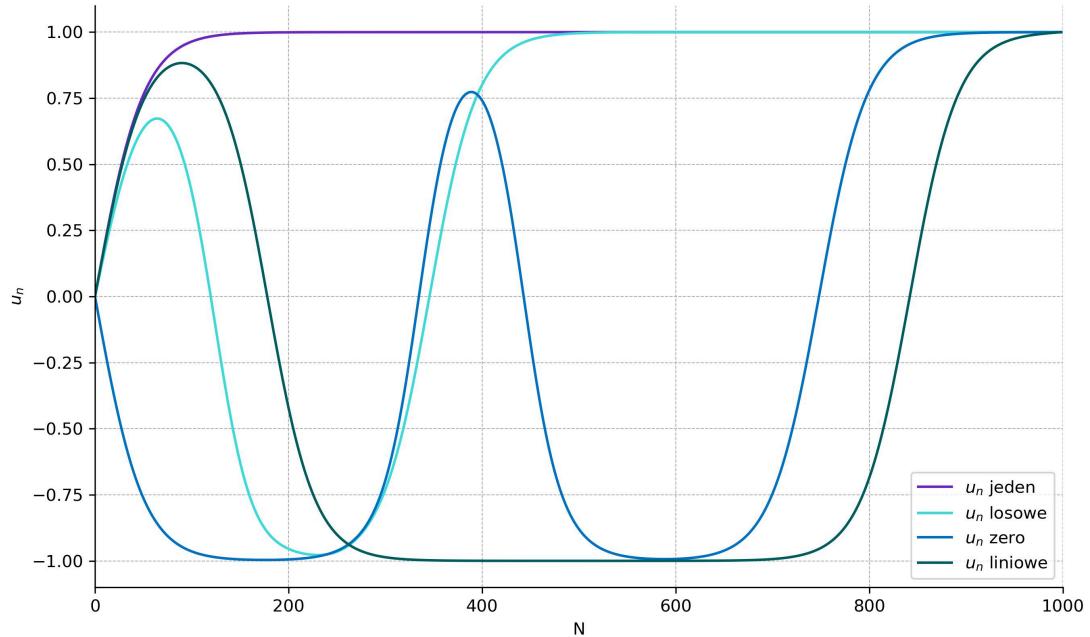
$$J = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ -\frac{1}{h^2} & \frac{\partial f_1}{\partial u_1} & -\frac{1}{h^2} & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -\frac{1}{h^2} & \frac{\partial f_{N-1}}{\partial u_{N-1}} & -\frac{1}{h^2} \\ 0 & \dots & 0 & 0 & 1 \end{pmatrix}$$

Gdzie dla elementów na diagonali

$$\frac{\partial f_n}{\partial u_n} = \frac{\partial}{\partial u_n} \left[-\frac{u_{n-1} - 2u_n + u_{n+1}}{h^2} + 2u_n(u_n^2 - 1) \right] = \frac{2}{h^2} + 6u_n^2 - 2$$

Struktura trójdiagonalna macierzy pozwala nam rozwiązać układ równań $J\delta u = g(u)$ algorytmem Thomasa w czasie $O(N)$.

3. Rozwiązania otrzymane tłumioną wielowymiarową metodą Newtona



Wykres 1: Rozwiązanie równania

4. Podsumowanie

Tłumiona wielowymiarowa Newtona w zależności od warunku początkowego, może osiągać różne minima lokalne, ze względu na nieliniowość członu $-\frac{u_{n-1} - 2u_n + u_{n+1}}{h^2} + 2u_n(u_n^2 - 1)$. W zależność od tego jak wyglądał początkowy wektor przybliżeń u metoda ta dała inne wyniki. Dla początkowego wektora u będącego zapoczątkowanego jedynkami, metoda dość szybko zbiega do ustabilizowania wyników kolejnych wartości u_n jako jeden. Podobnie jest dla wektora losowego, który najpierw wykonuje oscylację do wartości -1 , żeby następnie ustabilizować się w 1 . Dla liniowo zapoczątkowanego wektora $u_n = \frac{1}{N-1}$, wartości rozwiązania przez dość długi czas znajdują się w -1 , żeby na sam koniec zbiec do wartości 1 . Najciekawszy jest wektor u będący początkowo zerami oprócz ostatniego $u_N = 1$, wykonuje on 2 oscylacje do momentu ustabilizowania się końcowych wartości $u_n = 1$. Wykres 1 dobrze pokazuje, że nasze równanie posiada różne rozwiązania, które spełniają założoną dokładność 10^{-6} .