Sprawozdanie z zadania P2.7

Sławomir Górawski

17 grudnia 2017

1. Cel wykonanych doświadczeń

Załóżmy, że funkcja f ma w przedziale $[a, b]$ funkcję odwrotną i że wartości funkcji f znane są jedynie dla argumentów $a \leqslant x_0 < x_1 < \ldots < x_n \leqslant b \quad (n \in N)$. Chcemy znaleźć przybliżone rozwiązanie równania:

$$f(x) = c \quad (c \in R) \tag{1}$$

leżące w przedziale $[a, b]$.

Pierwszym podejściem, jakie przychodzi na myśl, jest wykorzystanie faktu, że f ma w danym przedziale funkcję odwrotną i wykorzystanie odwrotnej interpolacji. załóżmy, że mamy dane wektory $X = [x_0, x_1, \ldots, x_n]$ (węzły) oraz $Y = [y_0, y_1, \ldots, y_n]$, gdzie $y_i = f(x_i)$ dla $i = 0, 1, \ldots, n$. Możemy przekształcić wyjściowe równanie:

$$f(x) = c$$

do równoważnej postaci:

$$x = f^{-1}(c), \tag{2}$$

oznaczając funkcję odwrotną do f jako f^{-1}.

Znamy wartości f dla $x \in X$, zatem zamieniając oznaczenia osi znamy również wartości f^{-1} dla $x \in Y$ - są to wybrane przez nas węzły X. Przypuśćmy, że dysponujemy funkcją `interpolate(X, Y)`, która zwraca wielomian interpolacyjny przyjmujący w każdym punkcie x_i wartość y_i dla odpowiadającego i. Aby zinterpolować funkcję odwrotną do f wystarczy zamienić miejscami parametry funkcji `interpolate`. Wynikiem wywołania `interpolate(Y, X)` będzie wielomian interpolujący f^{-1}, który oznaczmy jako $L_n^{-1}(x)$. Przybliżonym rozwiązaniem równania (2) będzie $L_n^{-1}(c)$.

Problemem, jaki można zauważyć przy stosowaniu tej metody, jest dobór węzłów. Przy interpolowaniu f^{-1} wektorem węzłów jest bowiem nie X, lecz Y. Niezależnie zatem, jaki układ węzłów obraliśmy, wybierając punkty $x \in X$, układ węzłów faktycznie używany w procesie interpolacji odwrotnej może okazać się zupełnie inny. W przypadku gdy f zachowuje się w zadanym przedziale w sposób zbliżony do funkcji liniowej, nie powinno stanowić to dużego problemu - przekształcenie nie zmieni znacząco ich układu. Dla niektórych funkcji może stanowić to jednak poważny problem, przez co dokładność interpolacji odwrotnej

może znacznie ucierpieć, bowiem układ węzłów zacznie być równie chaotyczny jak zbiór losowych punktów.

Jasne staje się zatem, że jeśli chcemy narzucić konkretny układ węzłów podczas interpolacji, co przy zamiarze znalezienia odpowiednio dokładnego rozwiązania równania (1) może być niezbędne, musimy interpolować wprost funkcję f. Oznaczmy wielomian interpolujący f w zadanych $n+1$ punktach jako $L_n(x)$. Możemy uprościć wtedy problem (1) do następującego:

$$L_n(x) = c \tag{3}$$

Niech $L'_n(x) = L_n(x) - c$. Otrzymujemy równoważną postać równania (3):

$$L'_n(x) = 0 \tag{4}$$

Udało nam się sprowadzić oryginalny problem do równoważnego problemu znalezienia miejsca zerowego wielomianu $L'_n(x)$. Z założenia wiemy, że f ma w danym przedziale funkcje odwrotną. Wynika z tego w szczególności, że na tym przedziale f jest różnowartościowa, zatem rozwiązanie problemu $f(x) = c$ jest jednoznaczne. Aby znaleźć miejsce zerowe $L'_n(x)$, możemy posłużyć się jedną ze znanych metod numerycznych służących do tego celu, na przykład metodą Newtona; konieczne jest jednak ograniczenie x do danego przedziału $[a, b]$, ponieważ zachowanie wielomianu interpolacyjnego poza jego granicami może być nieprzewidywalne.

Podsumujmy teoretyczne wady i zalety proponowanych metod rozwiązania problemu:

1. Metoda odwrotnej interpolacji
    - prostsza w implementacji
    - tylko jedno miejsce pojawienia się błędu: interpolacja
    - szczególnie dopasowana do założeń zadania
    - problem z węzłami interpolacyjnymi
2. Metoda interpolacyjno-iteracyjna
    - nieco trudniejsza w implementacji
    - dwa miejsca pojawienia się błędów: interpolacja i szukanie miejsca zerowego wielomianu interpolacyjnego
    - bardziej uniwersalna
    - dowolność w wyborze węzłów

Z tej wstępnej analizy niekoniecznie wynika wprost, która metoda lepiej sprawdzi się w rozwiązywaniu problemu. Celem doświadczeń jest więc przetestowanie ich i ocena, czy którakolwiek z nich sprawdza się w praktyce, oraz wskazanie na różnice w ich działaniu.

2. Opis użytych metod

W pliku `program.jl` zaimplementowane zostały obie rozważane metody, odpowiednio w postaci funkcji `solve_with_inverse_interpolation` dla metody odwrotnej interpolacji oraz `solve_with_newton` dla metody interpolacyjno-iteracyjnej. Przyjmują one parametry `f`, `c` i `nodes` oznaczające odpowiednio: funkcję f, stałą c z równania (1) oraz kolekcje węzłów interpolacyjnych, zwracają zaś wyliczoną wartość `x` mającą w przyblizeniu spełniać równanie (1). Obie odwołują się do funkcji `interpolate`, która tworzy wielomian interpolacyjny, używając formy Newtona, tj. ilorazów różnicowych jako współczynników. W celu ułatwienia zadania wielomiany przechowywane są jako obiekty klasy `Poly` z biblioteki *Polynomials*, która pozwala na intuicyjne używanie na nich operacji takich jak dodawanie czy mnożenie.

W pliku `program.ipynb` metody te zostały przetestowane dla konkretnych przykładów. Testy oceniają dokładność w odniesieniu do modelowego rozwiązania oraz czas wykonywania funkcji dla różnego doboru węzłów (pod uwagę brane są węzły równoodległe, Czebyszewa i rozmieszczone losowo z jednostajnym rozkładem prawdopodobieństwa). Wyniki przedstawione są w postaci wykresów.

3. Opis wykonanych doświadczeń

3.1 Przykład 1

$$f(x) = x^4 - 3x + 1 \qquad c = 0 \qquad (x \in [-0.8, 0.8])$$
$$x \approx 0.33766676564280153320879 44$$

Dla tego przykładu rozpatrywane były układy 10, 20 i 50 węzłów.

Dla 10 węzłów interpolacja zarówno wprost, jak i odwrotna wydają się być poprawne - wyjątek stanowi interpolacja odwrotna dla 10 węzłów losowych, której wykres w pewnym miejscu znacznie odbiega od wyjściowej funkcji $f(x) = x^4 - 3x + 1$. Prowadzi to do wniosku, że złożenie losowych węzłów z brakiem kontroli nad ich przekształceniem w przypadku interpolacji odwrotnej prowadzi do zupełnie już chaotycznego układu mogącego być źródłem znacznych rozbieżności. Błąd w przypadku interpolacji odwrotnej jest średnio rzędu 10^{-4}, co może nie jest złym wynikiem, jednak dla tej samej ilości węzłów metoda interpolacyjno-iteracyjna daje wynik z błędem tylko 10^{-16}, a czasami nawet z błędem pomijalnym w granicach precyzji arytmetyki.

Zwiększenie ilości węzłów do 20 czy nawet 50 nie wpływa znacząco na efektywność metody interpolacyjno-iteracyjnej: błąd wyniku utrzymuje się w granicach $[10^{-14}, 10^{-17}]$. Z kolei w przypadku zastosowania interpolacji odwrotnej pojawiają się znaczące rozbieżności wielomianu interpolacyjnego z funkcją f. Pomimo tego błąd wyniku zmniejsza się wraz z ilością dodanych węzłów, jednak ciężko jest zauważyć jakieś konkretne korelacje - o dziwo najgorzej wypadają węzły Czebyszewa, pomimo tego, że funkcja przypomina liniową na danym przedziale. Z pewnością można stwierdzić jednak, że metoda interpolacji odwrotnej

zachowuje się znacznie bardziej chaotycznie i nieprzewidywalnie, co jak dotąd nie przemawia na jej korzyść.

3.2 Przykład 2

$$f(x) = ln(x^2 + 4) \qquad c = ln(4.2) \qquad (x \in [0, 5])$$
$$x = \sqrt{0.2}$$

Dla tego przykładu rozpatrywane były układy 10 i 20 węzłów.

Dla 10 węzłów widać już brak precyzji w odwzorowaniu wykresu f w przypadku metody interpolacji odwrotnej. Metoda interpolacyjno-iteracyjna zachowuje się stabilnie, a rząd błędu jej wyniku jest dwa razy niższy (10^{-4}).

Podwojenie ilości węzłów spowodowało drastyczne odstępstwa interpolacji odwrotnej od f (rzędu 10^4 w pojedynczym przypadku), co wpłynęło na wyniki (choć najbardziej znaczące błędy pojawiały się w oddaleniu od przecięcia wykresów f i c). Z kolei metoda interpolacyjno-iteracyjna tylko skorzystała na zwiększeniu ilości węzłów i błąd jej wyników zmniejszył się dla każdego układu, osiągając wartości od 10^{-7} do 10^{-9}. Podsumowując, kolejny przykład potwierdza tezę z pierwszego.

3.3 Przykład 3

$$f(x) = \frac{1}{1 + 25x^2} \qquad c = 0.5 \qquad (x \in [0, 1])$$
$$x = 0.2$$

Dla tego przykładu testowane były układy 5, 20 i 30 węzłów.

Jest wiadome, że funkcja z tego przykładu okazywała się trudna w normalnej interpolacji. W przypadku interpolacji odwrotnej jest to jednak znacznie bardziej widoczne i na każdym z 9 wykresów można zauważyć, jak wielomian interpolacyjny znacznie odbiega od wykresu funkcji f. Metoda interpolacyjno-iteracyjna zachowuje się stabilnie (na co wpływ może mieć ograniczony przedział, na jakim ta znana z trudności funkcja jest interpolowana).

Błędy dla metody interpolacyjno-iteracyjnej są średnio rzędu 10^{-2} dla 5 węzłów, 10^{-8} dla 20 i 10^{-10} dla 30. Z kolei błędy w metodzie interpolacji odwrotnej są w większości przypadków nie do zaakceptowania, zaś dodawanie węzłów tylko je zwiększa.

4. Wnioski

Po przetestowaniu w praktyce działania dwóch algorytmów nasuwa się wniosek, że dopasowana do założeń tego zadania metoda interpolacji odwrotnej nie zdaje egzaminu. Pomimo tego, że teoretycznie jest prosta oraz intuicyjna, dla konkretnych przykładów generuje duże błędy, a dokładność interpolacji przez nią

dokonywanej pozostawia bardzo wiele do życzenia. Zwiększenie ilości węzłów nie poprawia jej zachowania - w wielu przypadkach jest dokładnie na odwrót. Podsumowując, wykorzystanie jej, nawet do problemu który spełnia założenie istnienia funkcji odwrotnej, nie jest raczej dobrym pomysłem w jakimkolwiek przypadku.

Metoda ta porównana została z bardziej uniwersalną metodą interpolacyjno-iteracyjną, która znacznie lepiej sprawdziła się w powierzonej jej roli. Błędy były znacznie niższego rzędu, większa ilość węzłów zwykle przekładała się na większą precyzję wyniku, a wykonywana podczas obliczeń interpolacja dobrze odwzorowywała zadaną funkcję. Mając to na uwadze, rekomenduję wykorzystywanie metody tej lub podobnej do rozwiązywania problemów takich, jak ten postawiony w zadaniu.

Dodatkowym wnioskiem płynących z testów może być to, że stosowanie losowego układu węzłów raczej nie jest dobrym pomysłem w typowych zastosowaniach, a zwłaszcza przy interpolacji odwrotnej. Z kolei węzły Czebyszewa, lepsze w teorii od innych, przekładają się również na lepszą dokładność w praktyce.