

Sprawozdanie

Obliczenia naukowe - lista 4

Kamil Król

244949

Zadanie 1

Celem tego zadania było napisanie funkcji w języku Julia, która oblicza ilorazy różnicowe. Dodatkowym wymaganiem było nieużywanie tablicy dwuwymiarowej.

Dane:

\mathbf{x} – wektor długości $n + 1$ zawierający węzły x_0, \dots, x_n ,

\mathbf{f} – wektor długości $n + 1$ zawierający wartości interpolowanej funkcji w poprzednio podanych węzłach tj. $f(x_0), \dots, f(x_n)$.

Oczekiwany wynik:

\mathbf{fx} – wektor długości $n + 1$ zawierający obliczone ilorazy różnicowe

Opis:

Najpierw przyjrzyjmy się temu w jaki sposób można obliczyć ilorazy różnicowe. Poniżej znajduje się wzór rekurencyjny pozwalający na obliczenie ilorazu różnicowego k -tego rzędu.

dla $k = 0$

$$f[x_i] = f(x_i),$$

dla $k = 1$

$$f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i},$$

dla $k > 1$

$$f[x_i, x_{i+1}, \dots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \dots, x_{i+k}] - f[x_i, x_{i+1}, \dots, x_{i+k-1}]}{x_k - x_i}.$$

Ważnym faktem jest to, że wartość ilorazu różnicowego nie zależy od kolejności węzłów (x_i). Kolejny użyteczny fakt to to, że znajomość węzłów x_i i wartości funkcji $f(x_i)$ (a więc też ilorazów różnicowych zerowego rzędu tj. $f[x_i] = f(x_i)$) pozwala, przy użyciu powyższego wzoru rekurencyjnego, na stworzenie tzw. tablicy ilorazów różnicowych dla wyższych rzędów. Przyjmując, że $d_{ik} = f[x_i, x_{i+1}, \dots, x_{i+k}]$ można wyrazić ją w następujący sposób:

$$\begin{array}{cccccc} d_{0,0} & d_{0,1} & d_{0,2} & \dots & d_{0,k-1} & d_{0,k} \\ d_{1,0} & d_{1,1} & d_{1,2} & \dots & d_{1,k-1} & \\ \dots & \dots & \dots & \dots & & \\ d_{k-1,0} & d_{k-1,1} & d_{k-1,2} & & & \\ d_{k-1,0} & d_{k-1,1} & & & & \\ d_{k,0} & & & & & \end{array}$$

Pierwsza intuicja co do zaprogramowania funkcji obliczającej ilorazy różnicowe to użycie macierzy – tablicy dwuwymiarowej. Zastanówmy się najpierw czy można to zrobić bardziej efektywnie i jakich danych z powyższej tablicy ilorazów różnicowych potrzebujemy. Interesujące dla nas są tylko dane w pierwszym wierszu tej tablicy. Jeśli dodatkowo zauważymy, że każda kolumna zależy tylko i wyłącznie od poprzedniej

kolumny możemy zaproponować rozwiązanie używające tablicy jednowymiarowej. W pierwszym kroku powinniśmy zapisać wartości pierwszej kolumny do jednowymiarowej tablicy. Te dane już mamy, ponieważ są to wartości funkcji w danych węzłach. (Przypomnijmy, że $f[x_i] = f(x_i)$). Następnie w każdym kolejnym kroku powinniśmy wpisywać odpowiednie wartości z kolejnych kolumn na ostatnie miejsca w tablicy. W rezultacie w naszej tablicy otrzymamy tylko wartości ilorazów z pierwszego wiersza.

Pseudokod algorytmu

Algorytm 1: Obliczanie ilorazów różnicowych

```
function ilorazyRoznicowe(x, f)
    for i ← 1 to length(f) do
        | f[x][i] ← f[i]
    for i ← 1 to length(f) do
        for j ← length(f) downto i do
            | f[x][j] ← (f[x][j] - f[x][j - 1]) / (x[j] - x[j - i])
    return f[x]
```

Zadanie 2

Celem tego zadania było napisanie funkcji obliczającej wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego algorytmu Hornera, która działa w czasie liniowym ($O(n)$).

Dane:

- \mathbf{x} – wektor długości $n + 1$ zawierający węzły x_0, \dots, x_n ,
- \mathbf{fx} – wektor długości $n + 1$ zawierający ilorazy różnicowe,
- \mathbf{t} – punkt, w którym należy obliczyć wartość wielomianu.

Oczekiwany wynik:

- \mathbf{nt} – wartość wielomianu w punkcie t

Opis:

Wzór na wielomian interpolacyjny Newtona N_n pokazuje w jaki sposób zależy on od funkcji f . Wzór ten można przedstawić używając ilorazów różnicowych:

$$N_n(x) = \sum_{i=0}^n f[x_0, x_1, \dots, x_i] \prod_{j=0}^{i-1} (x - x_j).$$

Z numerycznego punktu widzenia takie przedstawienie wielomianu interpolacyjnego jest bardzo atrakcyjne. Zauważmy, że w sytuacji kiedy chcielibyśmy dodać nowe węzły (x_i, y_i) możemy to zrobić korzystając ze wcześniej policzonych $d_k = f[x_0, x_1, \dots, x_k]$. Kluczowa jest tu własność ilorazów różnicowych mówiąca, że wartość ilorazu nie zależy od kolejności węzłów. Kolejną zaletą takiego zapisu jest to, że wartość tego wielomianu możemy łatwo obliczyć korzystając z uogólnionego algorytmu Hornera. Sposób w jaki można to zrobić przedstawiono poniżej.

$$\begin{aligned} w_n(x) &:= f[x_0, x_1, \dots, x_n] \\ w_k(x) &:= w_{k+1}(x - x_k) + f[x_0, x_1, \dots, x_k] \quad (k = n - 1, n - 2, \dots, 0) \\ N_n(x) &= w_0(x) \end{aligned}$$

Pseudokod algorytmu

Algorytm 2: Obliczanie wartości wielomianu interpolacyjnego w punkcie t .

```
function warNewton( $x, fx, t$ )  
   $n \leftarrow \text{length}(fx)$   
   $nt \leftarrow fx[n]$   
  for  $i \leftarrow n - 1$  downto 1 do  
     $nt \leftarrow fx[i] + (t - x[i]) \times nt$   
  return  $nt$ 
```

Zadanie 3

Celem tego zadania było napisanie funkcji obliczającej współczynniki a_0, \dots, a_n postaci naturalnej wielomianu interpolacyjnego dla zadanych współczynników $d_0 = f[x_0], d_1 = f[x_0, x_1], \dots, d_n = f[x_0, \dots, x_n]$ tego wielomianu w postaci Newtona oraz węzłów x_0, \dots, x_n . Ponadto funkcja miała działać w czasie $O(n^2)$.

Dane:

x – wektor długości $n + 1$ zawierający węzły x_0, \dots, x_n ,

fx – wektor długości $n + 1$ zawierający ilorazy różnicowe.

Oczekiwany wynik:

a – wektor długości $n + 1$ zawierający obliczone współczynniki postaci naturalnej.

Opis:

Przypomnijmy, że wartości d_0, d_1, \dots, d_n są współczynnikami wielomianu interpolacyjnego w postaci Newtona. Punktem wyjściowym to wyprowadzenia algorytmu będzie uogólniony algorytm Hornera. Najpierw jednak zapiszmy wielomian interpolacyjny w postaci Newtona:

$$p(x) = d_0 + \underbrace{(x - x_0)(d_1 + (x - x_1)(d_2 + \dots + (x - x_{n-2}) \overbrace{(d_{n-1} + (x - x_{n-1}) d_n)}^{W_{n-1}})) \dots)}_{W_1} \dots$$

$\underbrace{\hspace{15em}}_{W_0}$

Idea jest bardzo podobna do wyprowadzania uogólnionego algorytmu Hornera w poprzednim zadaniu. Teraz przyjrzyjmy się zaznaczonym wielomianom W_k . Zauważmy też, że ich stopnie rosną (patrzac od góry do dołu).

$$\begin{aligned} W_n(x) &= d_n \\ W_{n-1}(x) &= d_{n-1} + (x - x_{n-1})W_n \\ &\dots = \dots \\ W_k(x) &= d_k + (x - x_k)W_{k+1} \text{ dla } 0 \leq k < n \\ &\dots = \dots \\ W_0(x) &= p_0(x) \end{aligned}$$

Dodatkowo mamy też, że $\deg(W_k) = \deg(W_{k+1}) + 1$. Obie te obserwacje są kluczowe dla wyprowadzenia algorytmu. Przypomnijmy, że wartości d_0, d_1, \dots, d_n oraz x_0, \dots, x_n są danymi, a więc są znane. Zastanówmy się jak obliczyć współczynniki wielomianu W_{n-1} w postaci naturalnej. Mamy $\deg(W_{n-1}) =$

$\deg(W_n) + 1 = 0 + 1 = 1$, a zatem W_{n-1} możemy ogólnie zapisać jako $W_{n-1}(x) = a_0x^0 + a_1x^1$. Z drugiej strony patrząc na tabelę wyżej możemy go zapisać jako

$$W_{n-1}(x) = d_{n-1} + (x - x_{n-1})W_n = d_{n-1} + (x - x_{n-1})d_n = d_{n-1} + xd_n - x_{n-1}d_n = (d_{n-1} - d_nx_{n-1})x^0 + (d_n)x^1$$

Otrzymaliśmy współczynniki naturalne wielomianu W_{n-1} . Konkretniej mamy, że $a_0 = d_{n-1} - d_nx_{n-1}$ i $a_1 = d_n$. Zróbmy to samo dla W_{n-2} .

$$\begin{aligned} W_{n-2}(x) &= d_{n-2} + (x - x_{n-2})W_{n-1} = d_{n-2} + (x - x_{n-2})(a_0x^0 + a_1x^1) = \\ &= (d_{n-2} - x_{n-2}a_0)x^0 + (a_0 - a_1x_{n-2})x^1 + a_1x^2 \end{aligned}$$

Obliczyliśmy współczynniki W_{n-2} w postaci naturalnej. Jeśli ten wielomian zapiszemy jako $W_{n-2} = b_0x^0 + b_1x^1 + b_2x^2$ to współczynniki będą następujące: $b_0 = d_{n-2} - x_{n-2}a_0$, $b_1 = a_0 - a_1x_{n-2}$, $b_2 = a_1$. Widzimy zatem, że współczynniki przy najwyższej potędze wielomianów W_{n-1} i W_{n-2} są sobie równe. Ogólnie mamy, że współczynniki przy najwyższej potędze dla wielomianów W_k i W_{k-1} są sobie równe. Inna kluczowa obserwacja to fakt, że licząc współczynniki naturalne wielomianu W_{n-2} korzystaliśmy tylko z danych i ze współczynników naturalnych wielomianu W_{n-1} . Podobnie obliczając współczynniki wielomianu W_{n-1} korzystaliśmy tylko z danych i współczynników wielomianu W_n . Widzimy zatem, że współczynniki naturalne wielomianu W_k jesteśmy w stanie obliczyć znając współczynniki wielomianu W_{k+1} . Oznacza to, że możemy to zrobić używając jednej tablicy. Ponadto jesteśmy w stanie zrobić to w czasie $O(n)$. W szczególności możemy obliczyć współczynniki wielomianu $W_0 = p_0$ w postaci naturalnej licząc kolejno współczynniki wielomianów $W_n, W_{n-1}, \dots, W_1, W_0$ każdy w czasie $O(n)$ co daje łączny czas $O(n^2)$. Teraz dla podsumowania pseudokod.

Pseudokod algorytmu

Algorytm 3: Obliczanie współczynników naturalnych wielomianu interpolacyjnego.

```
function naturalna(x, fx)
    n ← length(fx)-1
    a[n] ← fx[n]
    for i ← n - 1 downto 0 do
        a[i] ← fx[i] - a[i + 1] × x[i]
        for j ← i + 1 to n - 1 do
            a[j] ← a[j] - a[j + 1] * x[i]
    return a
```

Zadanie 4

Celem zadania było napisanie funkcji interpolującej zadaną funkcję $f(x)$ w przedziale $[a, b]$ za pomocą wielomianu interpolacyjnego stopnia n w postaci Newtona, a także rysującej wykresy funkcji f oraz otrzymanego wielomianu interpolacyjnego. W interpolacji funkcji należało użyć węzłów równoodległych.

Dane:

f – zadana funkcja,

a, b – przedział interpolacji,

n – stopień wielomianu interpolacyjnego.

Oczekiwany wynik:

– wykres funkcji f oraz wielomianu interpolacyjnego w przedziale $[a, b]$.

Opis:

Na początku wyznaczyłem węzły interpolacyjne x_1, \dots, x_{n+1} w taki sposób aby odległość między nimi

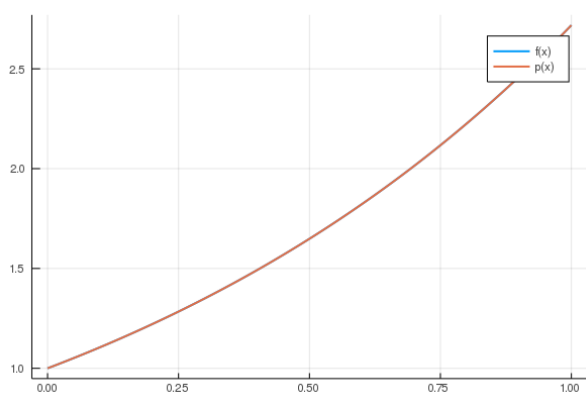
wynosiła $\frac{b-a}{n}$. Następnie obliczyłem wartości funkcji w tych punktach tj. $f(x_1), \dots, f(x_{n+1})$. W celu obliczenia ilorazów różnicowych posłużyłem się funkcją z zadania pierwszego – `ilorazyRoznicowe`. Następnie użyłem funkcji z zadania drugiego tj. `warNewton` do obliczenia wartości wielomianu interpolacyjnego w potrzebnych punktach. W celu uzyskania dokładniejszego wykresu, punkty dla których rysowałem wykres musiałem zagęścić. Zrobiłem to mnożąc dane n razy obrany przeze mnie parametr gęstości równy 40. Dzięki temu uzyskałem dokładniejsze wykresy.

Zadanie 5

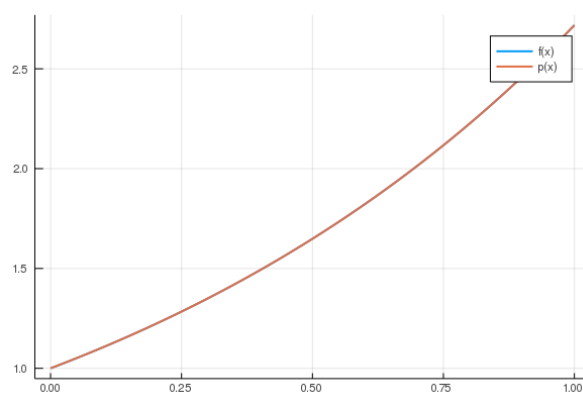
Celem zadania było przetestowanie funkcji `rysujNnfx(f,a,b,n)` (z zadania 4) na następujących przykładach:

- (a) $f(x) = e^x$, $[a, b] = [0, 1]$, $n \in \{5, 10, 15\}$,
- (b) $f(x) = x^2 \sin x$, $[a, b] = [-1, 1]$, $n \in \{5, 10, 15\}$.

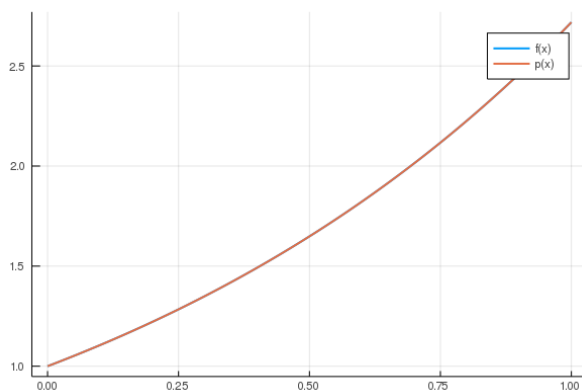
Poniżej narysowane wykresy dla obu funkcji.



(a) $n = 5$



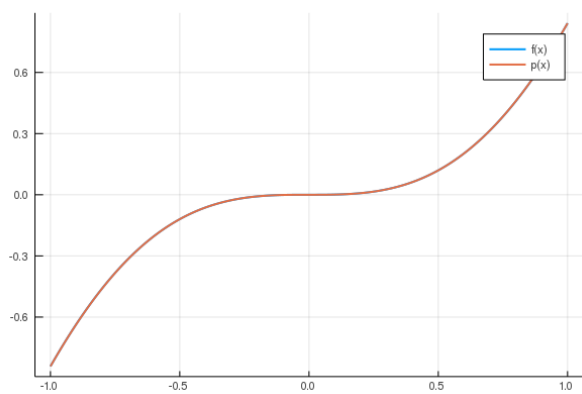
(b) $n = 10$



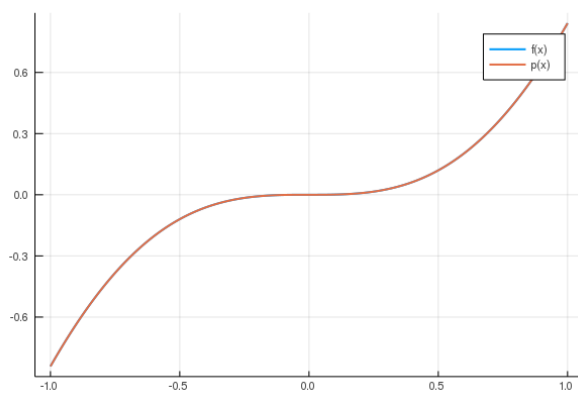
(c) $n = 15$

Wykres funkcji e^x i jej wielomianu interpolacyjnego dla danego stopnia n

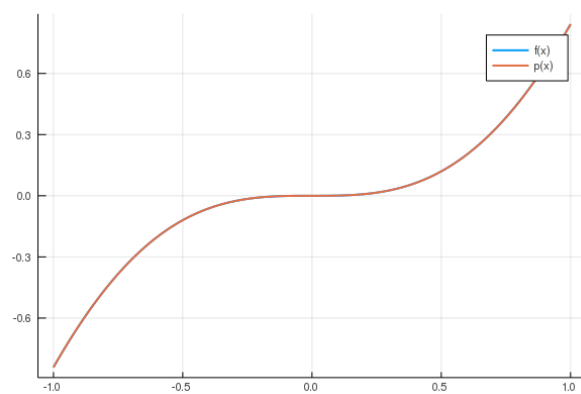
Na przykładach tych funkcji widać, że wybranie równoodległych węzłów dało bardzo dokładne przybliżenia funkcji. Dla żadnego z wykresów nie zaobserwowano rozbieżności. Kolejna rzecz warta zaobserwowania to fakt, że dla wszystkich wartości n funkcje były bardzo dobrze przybliżone.



(a) $n = 5$



(b) $n = 10$



(c) $n = 5$

Wykres funkcji $x^2 \sin x$ i jej wielomianu interpolacyjnego dla danego stopnia n

Zadanie 6

Celem zadania było