Miriam Jańczak 10 grudnia 2017
numer albumu: 229761 prowadzący: dr hab. Paweł Zieliński

Obliczenia naukowe

Lista 4

1 Ilorazy różnicowe

1.1 Opis problemu

Napisanie funkcji obliczającej ilorazy różnicowe na podstawie podanych węzłów i odpowiadających im wartości funkcji w sposób efektywnie wykorzystujący pamięć (tj. nie korzystając z tablicy dwuwymiarowej).

Iloraz różnicowy k-tego rzędu można obliczyć stosując następujący wzór rekurencyjny:

(i) dla $k = 0$
$$f[x_i] = f(x_i),$$

(ii) dla $k = 1$
$$f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i},$$

(iii) dla $k > 1$
$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_k - x_i}.$$

Warto zwrócić uwagę, że taki zapis daje intuicje co do tego, że iloraz różnicowy nie zależy w żaden sposób od kolejności węzłów (x_i). Właściwość ta znajduje zastosowanie przy praktycznym użyciu ilorazów różnicowych.

1.2 Rozwiązanie

Znajomość węzłów x_i i wartości funkcji $f(x_i)$ (czyli także ilorazów $f[x_i] = f(x_i)$ zerowego rzędu) pozwala, za pomocą powyższego wzoru rekurencyjnego, na stworzenie tzw. tablicy ilorazów różnicowych wyższych rzędów. Przyjmując, że $c_{ik} = f[x_i, x_{i+1}, \ldots, x_{i+k}]$ można wyrazić ją w następujący sposób:

$$\begin{array}{llllll} c_{0,0} & c_{0,1} & c_{0,2} & \cdots & c_{0,k-1} & c_{0,k} \\ c_{1,0} & c_{1,1} & c_{1,2} & \cdots & c_{1,k-1} & \\ \cdots & \cdots & \cdots & & & \\ c_{k-1,0} & c_{k-1,1} & & & & \\ c_{k,0} & & & & & \end{array} \tag{1}$$

Można zauważyć, że algorytm obliczania ilorazów różnicowych wynikający bezpośrednio z zastosowania wzoru rekurencyjnego mógłby wykorzystywać stworzoną wyżej dwuwymiarową tablicę

(1). Nie jest on jednak do końca optymalny, gdyż, jak się okazuje, wystarczy użyć jednowymiarowej tablicy fx. Początkowymi wartościami zmiennych fx_i są (przedstawione w tablicy 1) $c_{i,0} = f(x_i)$, a następnymi wartościami $c_{i-1,1}, \ldots, c_{1,i-1}, c_{0,i}$ dla każdej kolejnej zmiennej fx_i. Zauważyć można że zmienne tablicy fx z każdym przejściem tworzą kolejne kolumny tablicy ilorazów różnicowych (1), a wewnątrz każdej kolumny aktualizowane są kolejno od dołu do góry. Taka kolejność wykonywanych obliczeń zapewnia, że tablica fx zawierać będzie na danym etapie ilorazy potrzebne w późniejszych krokach. Opisane postępowanie przedstawia Algorytm 1.

Algorytm 1: Obliczanie ilorazów różnicowych

function `ilorazyRoznicowe(`x,f`)`
 for $i \leftarrow 1$ **to** `length(`f`)` **do**
  $fx[i] \leftarrow f[i]$
 for $i \leftarrow 1$ **to** `length(`f`)` **do**
  for $j \leftarrow$`length(`f`)` **downto** i **do**
   $fx[j] \leftarrow \dfrac{fx[j] - fx[j-1]}{x[j] - x[j-i]}$
 return fx

Dane:

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

`f` – wektor długości $n+1$ zawierający wartości interpolowanej funkcji w węzłach $f(x_0), \ldots, f(x_n)$.

Wyniki:

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

2 Wielomian interpolacyjny

2.1 Opis problemu

Napisanie funkcji obliczającej wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego algorytmu Hornera działającej w czasie liniowym ($O(n)$).

Wzór wielomianu interpolacyjnego Newtona można przedstawić używając ilorazów różnicowych, ukazuje on zależność wielomianu interpolacyjnego N_n od funkcji f:

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

Takie przedstawienie jest bardzo użyteczne w obliczeniach numerycznych. Jego zaletą jest to, że dodanie nowych punktów (x_i, y_i) nie narusza obliczonych wcześniej współczynników $c_k = f[x_0, x_1, \ldots, x_k]$. Zauważyć można również, że wartość tak wyrażonego wielomianu można łatwo obliczyć, stosując uogólniony algorytm Hornera. Taki sposób obliczania wielomianu interpolacyjnego Newtona przedstawiono poniżej (2).

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

2.2 Rozwiązanie

Funkcję obliczającą wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ wykorzystującą podany wyżej algorytm Hornera (2) przedstawia Algorytm 2.

Algorytm 2: Obliczanie wartości wielomianu interpolacyjnego w punkcie t.

```
function warNewton(x, fx, t)
    n ← length(fx)
    nt ← fx[n]
    for i ← n − 1 downto 1 do
        nt ← fx[i] + (t − x[i]) · nt
    return nt
```

Dane:

- `x` – wektor długości $n+1$ zawierający węzły $x_0, \ldots, x_n$,
- `fx` – wektor długości $n+1$ zawierający ilorazy różnicowe
- `t` – punkt, w którym należy obliczyć wartość wielomianu

Wyniki:

- `nt` – wartość wielomianu w punkcie t

3 Postać naturalna

3.1 Opis problemu

Napisanie funkcji obliczającej współczynniki $a_0, \ldots, a_n$ postaci naturalnej wielomianu interpolacyjnego dla zadanych współczynników $c_0 = f[x_0], c_1 = f[x_0, x_1], \ldots c_n = f[x_0, \ldots, x_n]$ tego wielomianu w postaci Newtona oraz węzłów $x_0, \ldots, x_n$.

3.2 Rozwiązanie

W celu znalezienia współczynników wielomianu interpolacyjnego w postaci naturalnej jako punkt wyjścia zastosowano uogólniony algorytm Hornera pokazany w poprzednim zadaniu (2). Idea rozwiązania jest bardzo prosta. Łatwo sprawdzić, że w wielomianie interpolacyjnym n-tego stopnia współczynnik a_n przy najwyższej potędze x jest równy c_n. Z tego faktu wynika także, że w_n z uogólnionego algorytmu Hornera jest również równy a_n. Posiadając $a_n = w_n$ w kolejnych krokach algorytmu tworzone będą wartości a_i bazujące na współczynnikach a_{i+1}. Aby znaleźć zależności między kolejnymi a_i algorytm przechodzi po wszystkich w_i od i równego n do 0, tak zmieniając tworzone współczynniki postaci naturalnej, żeby dla każdego w_i doprowadzić w danym momencie do postaci naturalnej. Jak się okazuje uzyskanie takiej postaci przy każdym przejściu nie jest trudne, ponieważ zmiany wprowadzane w każdym a_i są przewidywalne i łatwo jest je wyliczyć. Działanie tej metody prezentuje Algorytm 3.

Algorytm 3: Współczynniki naturalne wielomianu interpolacyjnego.

```
function naturalna(x, fx)
    n ← length(fx)
    a[n] ← fx[n]
    for i ← n − 1 downto 0 do
        a[i] ← fx[i] − a[i + 1] · x[i]
        for j ← i + 1 downto n − 1 do
            a[j] ← a[j] − a[j + 1] ∗ x[i]
    return a
```

Dane:

`x`	–	wektor długości $n+1$ zawierający węzły $x_0, \ldots, x_n$,
`fx`	–	wektor długości $n+1$ zawierający ilorazy różnicowe

Wyniki:

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

4 Interpolacja funkcji i jej wykres

4.1 Opis problemu

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

4.2 Rozwiązanie

Na funkcję `rysujNnfx` zadaną w zadaniu złożyło się kilka najważniejszych czynników. Na początku wyznaczane są węzły interpolacji $(x_1, \ldots, x_{n+1})$, które są rozmieszczone od siebie w odległości $\frac{b-a}{n}$ w przedziale $[a, b]$, a także wartości funkcji f w stworzonych węzłach – $(f(x_1), \ldots, f(x_{n+1}))$. Następnie przy pomocy funkcji `ilorazyRoznicowe` (zadanie 1) obliczone zostały ilorazy różnicowe dla stworzonych wcześniej węzłów. Dla uzyskania dokładniejszych wykresów zarówno wielomian interpolacyjny jak i funkcja f próbkowane są w $20 \cdot (n+1)$ równoodległych punktach, dla których wartości wielomianu obliczane są za pomocą funkcji `warNewton` (zadanie 2). Uzyskane w ten sposób wartości pozwalają na narysowanie wykresu funkcji f oraz jej wielomianu interpolacyjnego.

Funkcja `rysujNnfx`:

Dane:

`f`	–	zadana funkcja
`a, b`	–	przedział interpolacji
`n`	–	stopień wielomianu interpolacyjnego

Wyniki:

– funkcja rysuje wielomian interpolacyjny i interpolowaną funkcję w przedziale $[a, b]$

5 Interpolowanie funkcji e^x i $x^2 sinx$

5.1 Opis problemu

Przetestowanie funkcji `rysujNnfx(f,a,b,n)` (zadanie 4) na następujących przykładach:

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

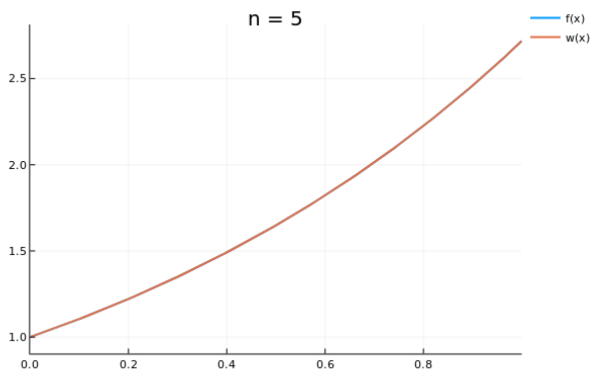
(b) $f(x) = x^2 \sin x$, $[a, b] = [-1, 1]$, $n \in \{5, 10, 15\}$.

5.2 Rozwiązanie

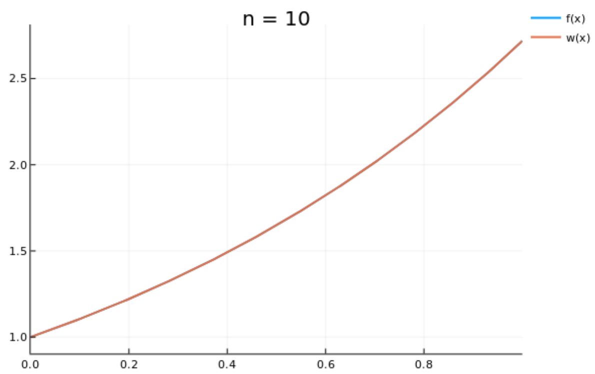
Dla odpowiednich danych wywołano funkcję `rysujNnfx(f,a,b,n)` opisaną w zadaniu 4.

5.3 Wyniki

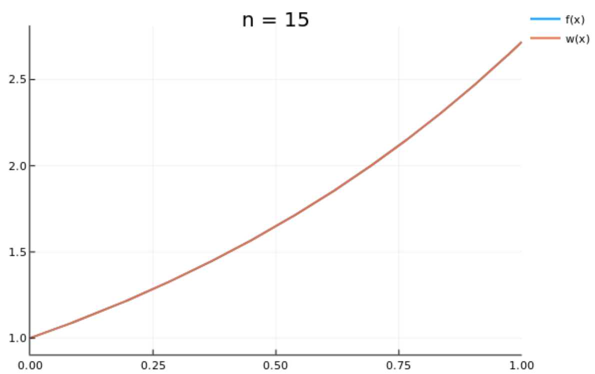
Wykresy otrzymane za pomocą metody `rysujNnfx(f,a,b,n)` prezentują Rysunek 1 i Rysunek 2.



(a) $n = 5$

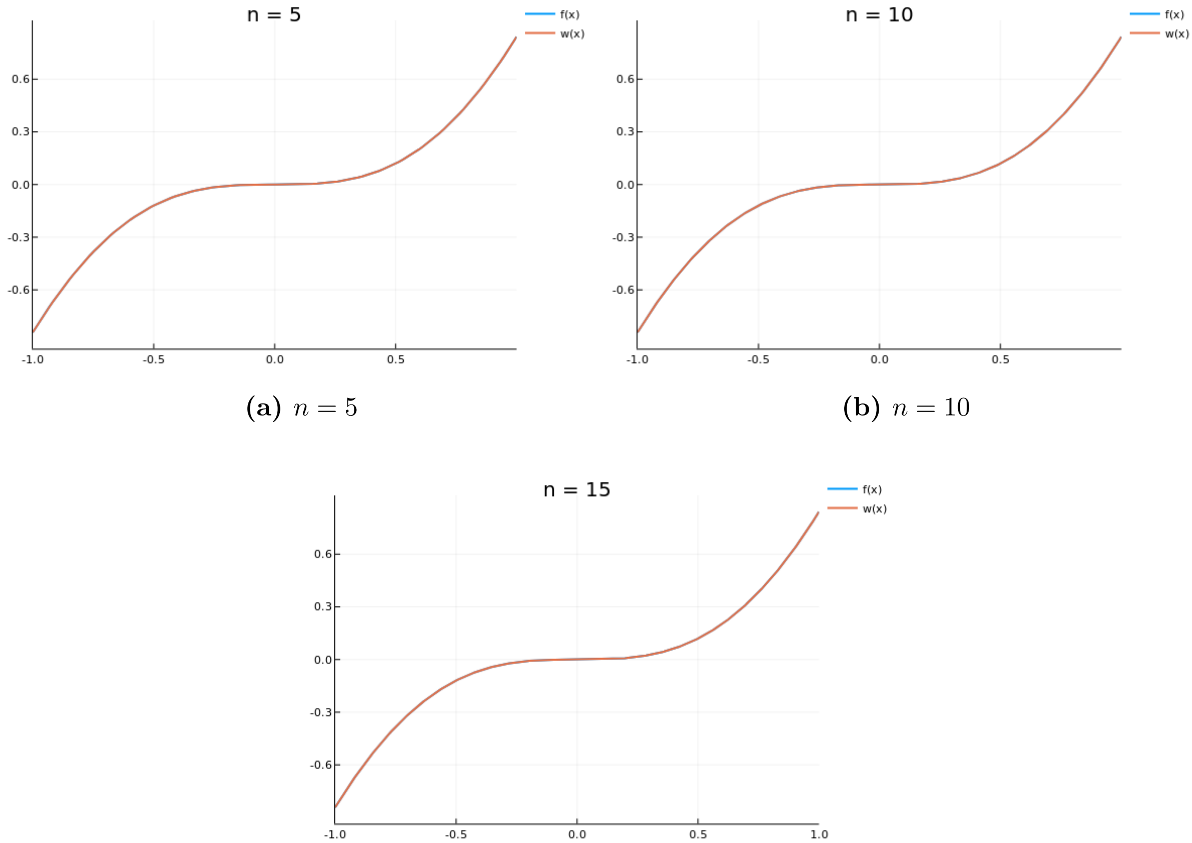


(b) $n = 10$



(c) $n = 15$

Rysunek 1: Wykres e^x i jej wielomianu interpolacyjnego dla danego stopnia n



(a) $n = 5$ (b) $n = 10$ (c) $n = 5$

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

5.4 Wnioski

Zarówno dla funkcji e^x jak i również $x^2 \sin x$ na zadanych przedziałach wielomiany interpolacyjne są bardzo bliskie interpolowanym funkcją, na wykresach 1 i 2 nie zaobserwowano rozbieżności. Dla $n = 5$ uzyskane wartości różnią się dopiero na szóstym miejscu po przecinku, a dla większych n ma jeszcze dalszych. Widać zatem że w tym przypadku zastosowanie równoodległych węzłów interpolacji (zadanie 4) dało bardzo dobre przybliżenia interpolowanych funkcji.

6 Interpolowanie funkcji $|x|$ i $\frac{1}{1+x^2}$ – zjawisko rozbieżności

6.1 Opis problemu

Przetestowanie funkcji `rysujNnfx(f,a,b,n)` (zadanie 4) na następujących przykładach:

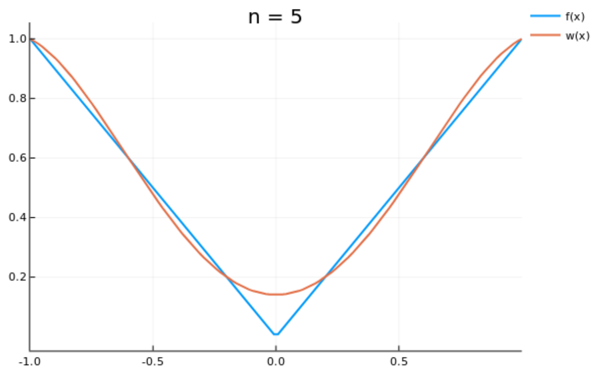
(a) $f(x) = |x|$, $[a, b] = [-1, 1]$, $n \in \{5, 10, 15\}$,
(b) $f(x) = \frac{1}{1+x^2}$, $[a, b] = [-5, 5]$, $n \in \{5, 10, 15\}$.

6.2 Rozwiązanie

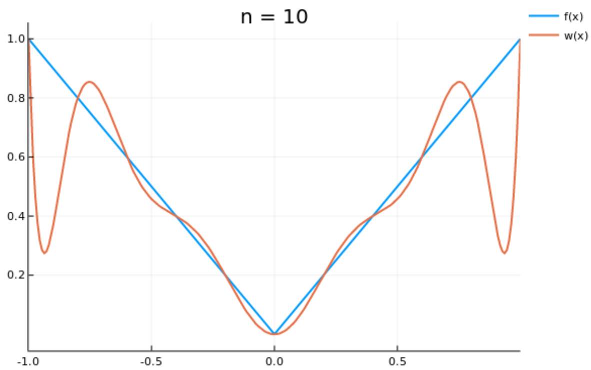
Dla odpowiednich danych wywołano funkcję `rysujNnfx(f,a,b,n)` opisaną w zadaniu 4.

6.3 Wyniki

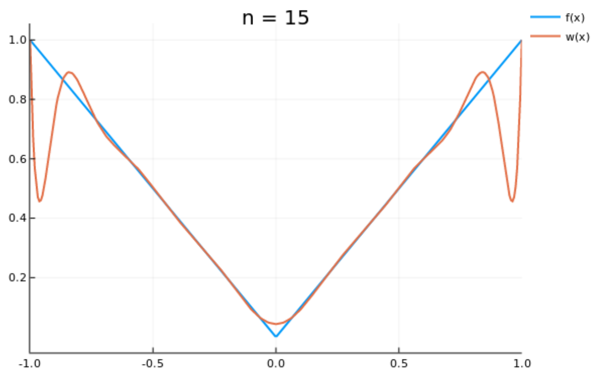
Wykresy otrzymane za pomocą metody `rysujNnfx(f,a,b,n)` prezentują Rysunek 3 i Rysunek 4.



(a) $n = 5$



(b) $n = 10$

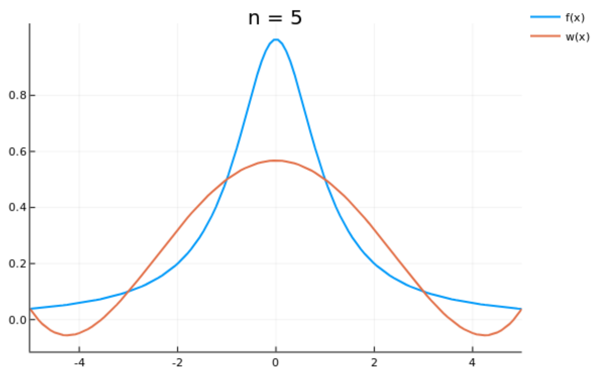


(c) $n = 15$

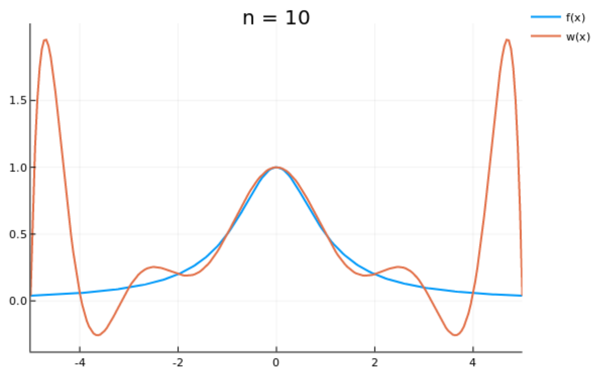
Rysunek 3: Wykres $|x|$ i jej wielomianu interpolacyjnego dla danego stopnia n

6.4 Wnioski

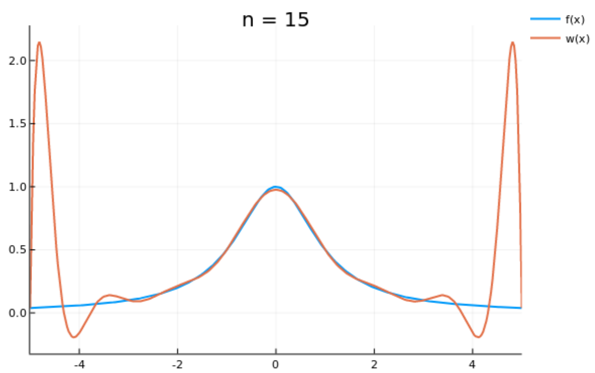
W obu przypadkach można zaobserwować wyraźne rozbieżności szczególnie na końcach przedziałów. Funkcja $|x|$ nie jest różniczkowalna i w największej mierze to odpowiada za odchylenia w tym przypadku. Natomiast dla funkcji $\frac{1}{1+x^2}$ możemy zaobserwować zjawisko Rungego – pogorszenie jakości interpolacji wielomianowej pomimo zwiększenia liczby węzłów. Można by przypuszczać, że ciąg wielomianów interpolacyjnych p_n dla coraz większych n będzie jednostajnie zbieżny do f. Jednak w tym przypadku zaobserwować można efekt wprost odwrotny. Dzieje się tak z powodu pewnej osobliwości tej funkcji na osi urojonej w pobliżu przedziału $[-5, 5]$. Istnieje jednak ogólny sposób na poprawę interpolacji. Zauważyć można, że zastosowanie równoodległych węzłów sprawia, że tam gdzie interpolacja funkcji jest trudniejsza przypada stosunkowo niewielka liczba punktów (tyle samo co w innych częściach dziedziny). Oczywiste jest także to, że wielomian wysokiego stopnia charakteryzuje się dużą liczbą zer, a równocześnie szybko rozbiega do nieskończoności, dlatego niejako ”wije się” przyjmując skrajne wartości. Widać zatem że tam, gdzie wielomian potencjalnie mógłby uciekać, należałoby zwiększyć ilość węzłów. Aby uniknąć powstałych błędów warto zastosować wielomiany oparte na węzłach Czebyszewa (rosyjski matematyk, który sformułował problem dokładności interpolacji jako problem znalezienia



(a) $n = 5$



(b) $n = 10$



(c) $n = 5$

Rysunek 4: Wykres $\frac{1}{1+x^2}$ i jej wielomianu interpolacyjnego dla danego stopnia n

wielomianu, który najlepiej przybliżałby zero na danym przedziale), które mają dużo mniejsze oscylacje, gdyż oparte są na węzłach mocno zagęszczonych przy końcach przedziału. Warto pamiętać, że dla dokładniejszego przybliżenia funkcji wielomianem interpolacyjnym należy dobrać węzły w sposób optymalny np. stosując zera wielomianu Czebyszewa.