```
            / (nodes[i]-nodes[i-j]);
        else {
            poly[i] /= (double)j;
            k[i]--;
        }
    }
```

Etap (4) Wynikiem działania powyższego kodu jest wypełnienie tablicy *poly* wartościami współczynników wielomianu interpolacyjnego w bazie Newtona. Aby przejść do bazy używanej w tradycyjnnym zapisie wielomianów wystarczy wykonać kilka prostych operacji arytmetycznych, które realizuje poniższy kod:

```
for(int j=count-1;j>=0;j--)
    for(int i=j;i<count-1;i++)
        poly[i] = poly[i]-poly[i+1]*nodes[j];
```

Pełny kod źródłowy programu można ściągnąć ze strony:
`http://www.houp.info/mn/hermite.tar.bz2`

Źródło:

1. Kiciak P.: *Podstawy modelowania krzywych i powierzchni. Wydanie II.* WNT, Warszawa 2005, s. 485–491.

Wielomian interpolacyjny Hermite'a

Witold Bołt

15 listopada 2005

1 Sformułowanie zadania

Dla funkcji $f\colon [a,b] \to \mathbb{R}$ danej wzorem:

$$f(t) = e^{2t} + 1$$

wyznaczyć wielomian interpolacyjny Hermite'a na zadanych węzłach. Wypisać wielomian w bazie standardowej (jednomianów) i narysować wykres funkcji oraz wielomianu.

2 Podstawy teoretyczne

2.1 Baza Newtona przestrzeni wielomianów

Wszystkie wielomiany o współczynnikach i wartościach rzeczywistych stopnia co najwyżej n stanowią przestrzeń liniową. Tradycyjnie przyjętą bazą „standardową" tej przestrzeni jest baza składająca się z jednomianów postaci x^j dla $j = 0, 1, \ldots, n$.

Twierdzenie 2.1. *Niech $u_0, u_1, \ldots, u_n$ będą dowolnymi liczbami rzeczywistymi. Wówczas wielomiany postaci:*

$$p_0(t) = 1,\ p_i(t) = (t - u_{i-1})p_{i-1}(t)\ \textit{dla}\ i = 1, \ldots, n+1$$

stanowią bazę przestrzeni wielomianów stopnia co najwyżej $n+1$. Bazę tą nazywamy bazą Newtona przestrzeni wielomianów.

2.2 Wielomian interpolacyjny Hermite'a funkcji f

Twierdzenie 2.2. *Niech dany będzie ciąg liczb $u_0 \leqslant u_1, \leqslant \ldots \leqslant u_n$. Dla dowolnego ciągu liczb $c_0, c_1, \ldots, c_n$, istnieje dokładnie jeden wielomian W stopnia co najwyżej n, taki że jeśli liczba u_k w ciągu $u_0, \ldots, u_n$ występuje r razy, a dokładniej, jeśli $u_k = \cdots = u_{k+r-1}$ i $z\ k > 0$ wynika $u_{k-1} \neq u_k$ oraz $z\ k + r \leqslant n$ wynika $u_{k+r-1} \neq u_{k+r}$ (liczbę u_k nazywamy wtedy węzłem r–krotnym), który spełnia:*

$$W(u_k) = c_k,\ W'(u_k) = c_{k+1},\ \ldots\ ,\ W^{(r-1)}(u_k) = c_{k+r-1}.$$

Definicja 2.3 (Wielomian interpolacyjny Hermite'a funkcji f)**.** Wielomian W stopnia co najwyżej n, nazywamy wielomianem interpolacyjnym Hermite'a funkcji f, jeśli w każdym r–krotnym węźle u_k spełnia równania:

$$W(u_k) = f(u_k),\ W'(u_k) = f'(u_k),\ \ldots\ ,\ W^{(r-1)}(u_k) = f^{(r-1)}(u_k).$$

Z powyższej definicji widzimy, iż pojęcie wielomianu interpolacyjnego Hermite'a jest uogólnieniem pojęcia „zwykłego" wielomianu interpolacyjnego Lagrange'a, który narzucał jedynie równość wartości wielomianu i funkcji w danych punktach. Twierdzenie (2.2) gwarantuje, że wielomian taki wyznaczony jest w sposób jednoznaczny.

Definicja 2.4 (różnica dzielona)**.** Dla parami różnych liczb $u_0, \ldots, u_n$ **różnice dzielone** funkcji f są określone w następujący sposób:

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

$$f[u_i, \ldots, u_i + k] = \frac{f[u_i, \ldots, u_{i+k-1}] - f[u_{i+1}, \ldots, u_{i+k}]}{u_i - u_{i+k}}$$

Definicja 2.5. Różnica dzielona rzędu k w $(k+1)$–krotnym węźle określona jest wzorem:

$$f[\underbrace{u_i, \ldots, u_i}_{k+1}] = \frac{f^{(k)}(u_i)}{k!}.$$

Etap (2) Funkcja pobierająca dane od użytkownika przygotowuje tablice węzłów. Tablica zawiera liczby: $u_0 \leqslant u_1 \leqslant \ldots \leqslant u_n$. W programie przyjmuje się założenie, że jeśli jakiś węzeł „podany jest" k–krotnie, to dla pierwszego wystąpienia wyliczana jest wartość funkcji, dla drugiego wartość pierwszej pochodnej itd. aż do pochodnej rzędu $k-1$. W tablicy k będziemy więc przechowywać rząd pochodnej do wyliczenia w węźle o danym indeksie. Poniższy fragment kodu wyznacza wartości tablicy k:

```
k[0]=0;
for(int i=1;i<count;i++) {
   if (nodes[i] == nodes[i-1])
      k[i] = k[i-1]+1;
   else
      k[i] = 0;
}
```

Po wyznaczeniu tablicy k jesteśmy gotowi na to aby wyliczać wartości funkcji i pochodnych funkcji f. Wartości te zapiszemy w tablic *poly*, która później zawierać będzie współczynniki naszego wielomianu. Za wyliczenie odpowiednich wartości odpowiada następujący fragment kodu:

```
for(int i=0;i<count;i++)
   poly[i] = f(nodes[i],k[i]);
```

Etap (3) W tym momencie mamy już wszelkie niezbędne informacje do tego by rozpocząć wyliczanie współczynników wielomianu. Wyliczeń dokonujemy na podstawie twierdzenia (2.6) oraz dwóch definicji ilorazów dzielonych.

```
for(int j=1;j<count;j++)
   for(int i=count-1;i>=j;i--) {
      if(k[i]==0)
         poly[i] = (poly[i] - poly[i-1-k[i-1]])
```

- *inter.c* – zawiera funkcję wyliczającą współczynniki wielomianu interpolacyjnego metodą Hermite'a (więcej o tej funkcji w następnym podrozdziale);
- *plot.c* – zawiera funkcje generujące rysunek w formacie Asymptote oraz dokument w formacie LaTeX (funkcje te nie korzystają w żaden sposób z wymienionych formatów – generują jedynie pliki źródłowe).

4.1 Moduł *inter.c* – wyliczanie wielomianu interpolacyjnego

W tym podrozdziale skupimy się na opisaniu modułu inter.c. Moduł ten zawiera dwie funkcję. Pierwszą z nich jest funkcja pomocnicza o nazwie *f*. Funkcja ta służy do wyliczenia wartości zadanej funkcji $f(x) = e^{2x} + 1$ oraz pochodnych dowolnego rzędu z tej funkcji. W tym celu używana jest wbudowana w bibliotekę standardową języka C funkcja *exp*. Oto treść funkcji *f*:

```
double f (double x,int d) {
   return ( (1 << d) * exp(2*x) + ( (d>0) ? 0 : 1 ) );
}
```

Główną funkcją tego modułu jest funkcja *getHermitePolynominal*. Jej działanie można podzielić na cztery etapy:

(1) Przydzielenie pamięci na zmienne. Etap ten nie jest wyjątkowo interesujący, zostanie więc pominięty w dalszym opisie.

(2) Wyznaczenie rzędów pochodnych do wyliczenia w węzłach i wyliczenie wartości funkcji w węzłach.

(3) Wyznaczenie współczynników wielomianu interpolacyjnego w bazie Newtona.

(4) Przejście z bazy Newtona do bazy jednomianów (tzw. bazy potęgowej).

Korzystając z dwóch wzorów z powyższych definicjach, możemy wyliczyć różnicę dzieloną dla dowolnych $u_0, \ldots, u_n$.

Twierdzenie 2.6. *Wielomian interpolacyjny Hermite'a W funkcji f (gdy f jest klasy C^{n+1}) dany jest wzorem:*

$$W(t) = \sum_{i=0}^{n} f[u_0, \ldots, u_i] p_i(t)$$

gdzie p_i to współczynniki bazy Newtona.

Wyznaczenie wielomianu interpolacyjnego Hermite'a sprawdza się więc do wyznaczenia różnic dzielonych $f[u_0, \ldots, u_i]$, które są współczynnikami tego wielomianu w bazie Newtona.

3 Opis działania programu

Program komunikuje się z użytkownikiem w sposób interakcyjny. Najpierw użytkownik podaje dwie liczby rzeczywiste a i b określające krańce przedziału $[a, b]$, a następnie podaje liczbę węzłów (która musi być nie mniejsza niż 2). W programie przyjmuje się założenie, że a staje się pierwszym węzłem natomiast b ostatnim. Kolejną rzeczą jest wybór sposobu podziału odcinka $[a, b]$.

- Wybranie opcji *rozkład jednorodny* powoduje że przedział zostanie podzielony na odcinki równej długości (zgodnie z podaną wcześniej liczbą węzłów). Użytkownik w takim wypadku zostanie poproszony o podanie jednej (stałej dla wszystkich węzłów) krotności.
- Wybranie opcji *rozkład jednorodny ze zmienną krotności* powoduje, podobnie jak poprzednio, że odcinek $[a, b]$ zostanie podzielony na odcinki o równej długości, jednak w odróżnieniu od przypadku poprzedniego, dla każdego z węzłów oddzielnie można podać różną krotność.

- Wybranie opcji *rozkład niejednorodny* umożliwia użytkownikowi podział przedziału $[a, b]$ na dowolne odcinki i przypisanie każdemu z węzłów dowolnej krotności.

Po wprowadzeniu danych program wylicza wielomian interpolacyjny i zapisuje go w pliku wyjściowym. Domyślnie wynik zapisywany jest w pliku o nazwie *plot.tex*[1]. Jest to plik w formacie LaTeX, który wewnętrznie korzysta z pakietu Asymptote. Wygenerowany dokument zawiera:

- wzór i dziedzinę funkcji f,
- wzór wielomianu interpolacyjnego W,
- wykres obu tych funkcji.

Aby, na podstawie tegoż pliku wygenerować dokument w formacie PDF najlepiej wykorzystać dostarczony skrypt powłoki o nazwie *genPDF.sh*, który należy uruchomić w następujący sposób:

```
./genPDF.sh plik
```

jeśli nasz dokument jest zapisany w pliku o nazwie: *plik.tex*. Do poprawnego działania skrypt *genPDF.sh* wymaga programów: latex, asymptote (komenda asy), dvips oraz ps2pdf. Wynikiem działania skryptu jest powstanie pliku *plik.pdf*.

3.1 Przykładowy wynik działania programu

Poniżej przedstawiono wynik działania programu, dla przedziału $[-1, 1]$ podzielonego na 5 równoodległych węzłów o stałej krotności 2.

Funkcja $f\colon [-1.0, 1.0] \to \mathbb{R}$ dana wzorem:

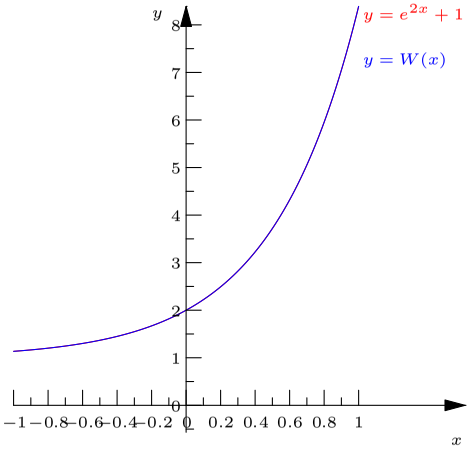
$$f(x) = e^{2x} + 1.$$

[1] Aby zapisać wynik w innym pliku należy uruchomić program z jednym parametrem będącym nazwą pliku wynikowego.

Wielomian interpolacyjny[2]:

$$W(x) = +0.0015 \cdot x^9 + 0.0071 \cdot x^8$$
$$+0.0253 \cdot x^7 + 0.0883 \cdot x^6 + 0.2667 \cdot x^5 + 0.6669 \cdot x^4$$
$$+1.3333 \cdot x^3 + 2.0000 \cdot x^2 + 2.0000 \cdot x^1 + 2.0000 \cdot x^0$$

Wykres:



4 Opis struktury programu

Program został napisany w języku C (a dokładniej w wersji C99 tegoż języka). Kod źródłowy programu podzielony jest na części (moduły):

- *main.c* – zawiera główną funkcję programu, wywołuje ona funkcje pobierającą dane, wyliczającą wielomian oraz generującą wyjście;
- *getdata.c* – zawiera funkcję pobierającą dane od użytkownika;

[2] Aby poprawić czytelność, współczynniki wielomianu są wypisywane z dokładnością do 4 miejsc po przecinku.