6 (done - 2pkt)

24 November, 2023 23:46

L7.6. 2 punkty Jak wiadomo, język programowania PWO++ ma bogatą bibliotekę funkcji i procedur numerycznych. Wśród nich jest m.in. procedura `DD_Table(x,f)` znajdująca z dokładnością bliską maszynowej **ilorazy różnicowe** $f[x_0]$, $f[x_0,x_1]$, $\ldots$, $f[x_0,x_1,\ldots,x_n]$, gdzie `x`$:=[x_0,x_1,\ldots,x_n]$ jest wektorem parami różnych liczb rzeczywistych, a `f` – daną funkcją. Niestety procedura ta ma pewną wadę, mianowicie n **musi być mniejsze** niż 21. W jaki sposób, wykorzystując procedurę `DD_Table` **tylko raz**, można **szybko** wyznaczyć ilorazy różnicowe $f[z_0]$, $f[z_0,z_1]$, $\ldots$, $f[z_0,z_1,\ldots,z_{20}]$, $f[z_0,z_1,\ldots,z_{20},z_{21}]$, gdzie $z_i \neq z_j$ dla $i \neq j$, $0 \leq i,j \leq 21$.

Uwagi. Rozwiązania, w których **dwukrotnie** używa się procedury `DD_Table` lub wykorzystuje się **jawny wzór** na iloraz różnicowy **nie wchodzą w grę**.

Wiemy że wzór z definicji wygląda tak: $w(x) = a_0 + \sum_{i=1}^{n} a_i \prod_{j=0}^{i-1}(x - x_j) = a_0 + a_1(x-x_0) + a_2(x-x_1)(x-x_0) + \ldots + a_n(x-x_{n-1})\cdots(x-x_1)(x-x_0)$

Czyli możemy zapisać: $L_{n+1}(x) = L_n(x) + F[x_0, x_1, \ldots, x_n, x_{n+1}] \cdot \prod_{i=0}^{n}(x - x_i)$

tego szukamy

Mamy:

I $L_{21}(x) = L_{20}(x) + k(x-x_0)\ldots(x-x_n)$ - ze wzoru

II $F(x_{21}) = L_{21}(x_{21})$ - bo interpolujemy w tym punkcie

Użyciem DD_Table obliczamy więc

$F[z_0], F[z_0,z_1], \ldots, F[z_0,\ldots,z_{20}]$

i mamy:

$f(x_{21}) = L_{20}(x_{21}) + k(x_{21}-x_0)\ldots(x_{21}-x_{20})$

co przekształcamy by wyznaczyć k:

$$k = \frac{F(x_{21}) - L_{20}(x_{21})}{(x_{21}-x_0)\cdots(x_{21}-x_{20})}$$