Interpolacja wielomianowa

Michał Fica

November 2021

1 Wielomian interpolacyjny Lagrange'a

1.1 Interpolacja Lagrange'a

Wielomian w postaci Lagrange'a służy do przybliżenia funkcji f w sytuacji, gdy znane są wartości tej funkcji w punktach $x_0, x_1, ..., x_n$. Dla wielomianu stopnia n wybiera się $n + 1$ punktów $x_0, x_1, \ldots, x_n$ i wielomian przyjmuje postać:

$$w(x) = \sum_{i=0}^{n} y_i \cdot \prod_{j=0 \land j \neq i}^{n} (x - x_j)/(x_i - x_j) \tag{1.1}$$

gdzie

$$y_i = f(x_i)$$

Tak zdefiniowany wielomian jest wielomianem interpolacyjnym. Niech

$$L_i(x) := \prod_{j=0 \land j \neq i}^{n} (x - x_j)/(x_i - x_j)$$

Wielomian L_i zeruje się w każdym z węzłów poza x_i, natomiast wartość wielomianu L_i w punkcie x_i jest równa :

$$L_i(x_i) = \prod_{j=0 \land j \neq i}^{n} (x_i - x_j)/(x_i - x_j) = \prod_{j=0 \land j \neq i}^{n} 1 = 1$$

Dlatego :

$$w(x_i) = \sum_{i=0}^{n} f(x_i) \cdot L_i(x_i) = f(x_1) \cdot 0 + \ldots + f(x_i) \cdot 1 + \ldots + f(x_n) \cdot 0 = f(x_i)$$

1.2 Postać barycentryczna

Postać (1.1) wielomianu interpolacyjnego ma znaczenie głównie teoretyczne, dla celów praktycznych zaleca się używać formy barycentrycznej, która wygląda następująco:

$$w(x) = [\sum_{i=0}^{n} (w_i \cdot f(x_i))/(x - x_i)]/[\sum_{i=0}^{n} (w_i/(x - x_i)]$$

2 Postać Newtona wielomianu

2.1 Iloraz różnicowy

Wielomian w postaci Newtona zawiera w swojej konstrukcji ilorazy różnicowe. Definiuje się je w następujący sposób:

$$f[x_i, \ldots, x_{i+j+1}] = \sum_{i=0}^{k} (f(x_i)/\prod_{j=0, j \neq i}^{k} (x_i - x_j))$$

Iloraz różnicowy można także określić następującym wzorem rekurencyjnym:

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

$$f[x_i, \ldots, x_{i+j+1}] = (f[x_{i+1}, \ldots, x_{i+j+1}] - f[x_i, \ldots, x_{i+j}])/(x_{i+j+1} - x_i)$$

2.2 Postać Newtona

Aby przybliżyć funkcję f wielomianem w postaci Newtona stopnia n wybiera się $n+1$ punktów $x_0, x_1, ..., x_n$ i buduje się następujący wielomian:

$$w(x) = a_0 + \sum_{i=1}^{n} a_i \cdot \prod_{j=0}^{i-1}(x - x_j)$$

gdzie

$$a_i = f[x_0, \ldots, x_i]$$

2.3 Obliczanie wartości, uogólniony schemat Hornera

Wielomian $w(x)$ można przekształcić do postaci, z której jasno wynika, w jaki sposób obliczać wartość tego wielomianu w danym punkcie.

$$\begin{aligned} w(x) &= a_0 + a_1(x - x_0) + a_1(x - x_0)(x - x_1) + \ldots + a_n(x - x_0)\ldots(x - x_n) \\ &= a_0 + (x - x_0)(a_1 + a_2 \cdot (x - x_1) + \cdots + a_n(x - x_1)\cdots(x - x_n)) \\ &= a_0 + (x - x_0)(a_1 + \cdots (a_{n-2} + (x - x_{n-2})(a_{n-1} + (x - x_{n-1})(a_n)))\cdots) \end{aligned}$$

Zatem algorytm wygląda następująco:

$$w_n = a_n$$

$$w_k = w_{k+1} \cdot (x - x_k) + a_k, (k = n-1, \ldots, 0)$$

$$w(x) = w_0$$

3 Konwersja postaci Newtona na postać Lagrange'a

3.1 Opis zadania

Wprowadzając pewne oznaczenia postać Lagranga wielomianu można zapisać w następującej formie:

$$w(x) = \sum_{i=0}^{n} \sigma_i \prod_{j=0 \wedge j \neq i}^{n} (x - x_j)$$

$$\text{gdzie } \sigma_i{:=}w_i \cdot y_i, w_i := 1/\prod_{j=0,j}^{n}(t_i - t_j), i \in \{0, .., n\}$$

Mając podany wielomian $w \in \prod_n$ w postaci Newtona (tzn. jego współczynniki a_i) należy wyliczyć postać Lagrange'a tego wielomianu tj. znaleźć takie współczynniki σ_i, że:

$$w(x) = \sum_{i=0}^{n} \sigma_i \prod_{j=0 \wedge j \neq i}^{n} (x - x_j)$$

3.2 Algorytm

Na podstawie definicji współczynników a_i otrzymujemy:

$$a_i = \sum_{i=0}^{k} w(x_i) / \prod_{j=0 \wedge j \neq i}^{k} (x_i - x_j) = \sum_{i=0}^{k} \sigma_i \prod_{j=k+1}^{n} (x_i - x_j)$$

Korzystając z powyższej równości nasz problem możemy sprowadzić do rozwiązania następującego układu równań (3.3):

$$\begin{bmatrix} a_0 \\ a_1 \\ a_3 \\ \ldots \\ a_{n-1} \\ a_n \end{bmatrix} = \begin{bmatrix} \prod_{j=1}^{n}(x_0 - x_j) & 0 & 0 & \ldots & 0 \\ \prod_{j=2}^{n}(x_0 - x_j) & \prod_{j=2}^{n}(x_0 - x_j) & 0 & \ldots & 0 \\ \prod_{j=3}^{n}(x_0 - x_j) & \prod_{j=3}^{n}(x_0 - x_j) & \prod_{j=3}^{n}(x_0 - x_j) & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ (x_0 - x_n) & (x_1 - x_n) & (x_2 - x_n) & \ldots & 0 \\ 1 & 1 & 1 & \ldots & 1 \end{bmatrix} \begin{bmatrix} \sigma_0 \\ \sigma_1 \\ \sigma_2 \\ \ldots \\ \sigma_{n-1} \\ \sigma_n \end{bmatrix}$$

Prawa strona k-tego równania składa się z sumy $k+1$ niezerowych składników, z których każdy jest iloczynem pewnych $n-k$ czynników. Algorytm będzie w n krokach przekształcał pewne z tych równań. Wartość lewej strony k-tego równania po i krokach algorytmu będziemy pamiętać w zmiennej $a_k^{(j)}$. Algorytm wygląda następująco:

$$a_k^{(0)} := a_k, k \in \{0, \cdots, n\}$$

$$\text{dla } i \in \{1, \cdots, n\}, k \in \{0, \cdots, i-1\}:$$
$$a_k^{(i)} := a_k^{(i-1)} / (x_k - x_i)$$
$$a_i^{k+1} := a_i^k - a_k^i$$

$$\sigma_i := a_i^{(n)}, i \in \{0, \cdots, n\}$$

W i-tym kroku algorytmu wykonujemy następujące czynności. W i pierwszych równaniach, które składają się tylko z 1 niezerowego składnika $\prod_{j=i}^{n}(x_k - x_i)$ pozbywamy się pierwszego czynnika tego iloczynu. W $i+1$ równaniu usuwamy wszystkie składniki oprócz jednego, sprowadzając je do postaci, w której lewa strona składa się jedynie z 1 niezerowego składnika. Po wykonaniu n kroków algorytmu równania zostaną rozwiązane. Algorytm ten ma złożoność $O(n^2)$.

4 Konwersja postaci Lagrange'a na postać Newtona

4.1 Opis zadania

Dany jest wielomian $w \in \prod_n$ w postaci Lagrange'a, to znaczy znane są jego współczynniki σ_i z punktu (3.1). Celem jest znalezienie współczynników a_i z postaci Newtona tego wielomianu.

4.2 Algorytm

Powtarzając rozumowanie z podpunktu (3.2), otrzymujemy układ równań (3.3), w którym niewiadomymi tym razem są współczynniki a_i natomiast dane są współczynniki σ_i oraz węzły x_i. Będziemy wyliczać kolejne współczynniki a_i w kolejności od a_n do a_0. W k-tym kroku działania algorytmu sumujemy składniki prawej strony (n-k)-tego równania, a następnie uaktualniamy ich wartość.

Algorytm wygląda następująco:

$$\sigma_i^0 := \sigma_i, i \in \{0, \ldots, n\}$$

$$\text{dla } k \in \{0, \ldots, n\}:$$
$$\sigma_{n-k}^{k+1} := \sum_{i=0}^{n-k} \sigma_i^k$$
$$\sigma_i^{k+1} := (x_i - x_{n-k}) \cdot \sigma_i^k \text{ dla } i \in \{n-k, \ldots, 0\}$$

$$a_i := \sigma_i^{n+1-i} \; i \in \{0, \ldots, n\}$$

Algorytm ten ma złożoność $O(n^2)$.