Praktikum II „Numerik am Computer“ FS 2024
Prof. Dr. H. Harbrecht Universität Basel
M. Dujella, A. Heinimann, L. Mehrle, M. Morini, M. Ramm, D. Yusofy

---

Beilage zur Serie 4

Horner-Schema und Polynominterpolation

Problemstellung

Zu gegebenen $n+1$ Stützpunkten (x_i, y_i), $i = 0, 1, \ldots, n$, suchen wir das eindeutig bestimmte Polynom p von maximalem Grad n durch diese Stützpunkte. Das heisst: $p(x_i) = y_i$ für $i = 0, 1, \ldots, n$ (vergleiche „Einführung in die Numerik“).

Das Polynom p kann auf verschiedene Weise dargestellt werden. Im Allgemeinen nimmt man an, dass

$$p(z) = \sum_{j=0}^{n} w_j p_j(z).$$

Dabei sind $w_j \in \mathbb{R}$ die sogenannten Gewichte, p_j bestimmte Polynome und $z \in \mathbb{R}$ die Stelle, an welcher wir das Interpolationspolynom auswerten wollen.

Damit alle Polynome von Grad kleiner oder gleich n auf diese Weise dargestellt werden können, muss die Menge von Polynomfunktionen $\{p_j\}_{j=0}^{n}$ eine Basis vom Raum der Polynome von maximalem Grad n sein (vergleiche „Lineare Algebra“).

Im Folgenden betrachten wir zwei mögliche Darstellungen.

Monomiale Interpolation und Horner-Schema

Bei der monomialen Interpolation sind die $p_j(z)$ Monome, also $p_j(z) = z^j$, und es folgt

$$p(z) = \sum_{j=0}^{n} a_j p_j(z) = \sum_{j=0}^{n} a_j z^j.$$

Um das Interpolationspolynom zu bestimmen, müssen wir die Koeffizienten a_j, $j = 0, 1, \ldots, n$, berechnen. Da die Polynomfunktion p die Stützpunkte (x_i, y_i) für $i = 0, 1, \ldots, n$ passieren soll (*Interpolationsbedingung*), erhalten wir $n+1$ Gleichungen, nämlich:

$$\begin{aligned} a_0 x_0^0 + a_1 x_0^1 + \ldots + a_n x_0^n &= y_0, \\ a_0 x_1^0 + a_1 x_1^0 + \ldots + a_n x_1^n &= y_1, \\ &\vdots \\ a_0 x_n^0 + a_1 x_n^0 + \ldots + a_n x_n^n &= y_n \end{aligned}$$

In Matrixschreibweise entspricht dies dem Gleichungssystem

$$\underbrace{\begin{bmatrix} x_0^0 & x_0^1 & \ldots & x_0^n \\ x_1^0 & x_1^1 & \ldots & x_1^n \\ \vdots & & & \vdots \\ x_n^0 & x_n^1 & \ldots & x_n^n \end{bmatrix}}_{\mathbf{X}} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix}}_{\mathbf{a}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}}_{\mathbf{y}}.$$

Somit ist der Koeffizientenvektor $\mathbf{a}$ gegeben durch $\mathbf{a} = \mathbf{X}^{-1}\mathbf{y}$ (in MATLAB: `a = X\y`).

Die Matrix $\mathbf{X}$ kann in MATLAB spaltenweise aufgebaut werden. Dazu ist es am einfachsten, wenn die Stützstellen x_i als ein Spaltenvektor $\mathbf{x}$ dargestellt werden. Der Code zum Aufbau der Matrix könnte dann folgende Struktur haben:

```
X = zeros(n + 1);
for j = 0:n
    X(:, j + 1) = ...
end
```

Um ein Polynom, welches in monomialer Darstellung vorliegt, an einer bestimmten Stelle auszuwerten, kann man es sich zunutze machen, dass gilt:

$$\begin{aligned} p(z) &= a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z + a_0 \\ &= (((a_n z + a_{n-1})z + a_{n-2})z + \ldots)z + a_0. \end{aligned}$$

Die zweite Form hat den Vorteil, dass sie mit gleich vielen Additionen und weniger Multiplikationen auskommt. Daher ist eine Auswertung in dieser Form schneller. In MATLAB könnte man dies mit Hilfe einer `for`-Schleife umsetzen und diesen Ausdruck von innen nach aussen berechnen, sodass in jedem Schritt eine weitere Klammer verrechnet wird. Dies ergäbe dann zum Beispiel folgende Struktur:

$p = a_n$
for $j = n-1, n-2, \ldots, 0$ **do**
    $p = \ldots$
end for

Dieses Vorgehen zum Auswerten eines Polynoms wird *Horner-Schema* genannt.

Newton-Interpolation

Für die $n+1$ paarweise verschiedenen Stützpunkte (x_i, y_i), $i = 0, 1, \ldots, n$, betrachten wir nun die Newton-Interpolation. Das Polynom p hat die Form

$$p(z) = \sum_{j=0}^{n} c_j N_j(z),$$

wobei N_j das j-te Newton-Polynom ist, d.h. $N_0(z) = 1$ und

$$N_j(z) = \prod_{k=0}^{j-1} (z - x_k).$$

Auch hier werden die Koeffizienten mit Hilfe der Interpolationsbedingung $p(x_i) = y_i$, $i = 0, \ldots, n$ bestimmt. Das daraus entstehende lineare Gleichungssystem können wir schreiben als

$$\begin{aligned} p(x_0) &= c_0 &&= y_0, \\ p(x_1) &= c_0 + c_1(x_1 - x_0) &&= y_1, \\ p(x_2) &= c_0 + c_1(x_2 - x_0) + c_2(x_2 - x_0)(x_2 - x_1) &&= y_2, \\ &\ldots \end{aligned}$$

Die Koeffizienten können mit dem Schema der dividierten Differenzen berechnet werden. Zu gegebenen Stützstellen $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ sind die dividierten Differenzen folgendermassen definiert:

$$\begin{aligned} f[x_i] &:= y_i, && i = 0, 1, 2, \ldots, n, \\ f[x_i, \ldots, x_{i+j}] &:= \frac{f[x_{i+1}, \ldots, x_{i+j}] - f[x_i, \ldots, x_{i+j-1}]}{x_{i+j} - x_i} && 0 \leq i < i + j \leq n. \end{aligned}$$

Die Koeffizienten c_j sind durch diese dividierten Differenzen gegeben (Beweis siehe Vorlesung), nämlich

$$c_j = f[x_0, x_1, \ldots, x_j].$$

Der Algorithmus zur Bestimmung der Koeffizienten lautet:

for $k = 0, 1, \ldots, n$ **do**
  $c_k = y_k$
end for
for $j = 1, 2, \ldots, n$ **do**
  for $k = n, n-1, \ldots, j$ **do**
    $c_k = \dfrac{c_k - c_{k-1}}{x_k - x_{k-j}}$
  end for
end for

Das Interpolationspolynom p kann dann an der Stelle z wieder mit einem Horner-Schema effizient ausgewertet werden.

Allgemeine Informationen befinden sich auf der Website.
Zuletzt editiert am 22. März 2024.