Digitale AV-Technik
Prof. Uwe Hahne
Wintersemester 2024/25

Digitale AV-Technik - Aufgabenblatt 09

Thema: Reed-Solomon Codes

Ziele:

- Umgang mit Lagrange Polynomen verstehen
- Grundverständnis für die Funktionsweise von Reed-Solomon Codes

Aufgabe 1: Lagrange-Polynom berechnen

Gegeben sei die folgende Menge von Punkten, durch die ein Polynom verlaufen soll:

$$P_0 = (0, 0), \quad P_1 = (1, 1), \quad P_2 = (2, 8)$$

1. **Bestimmen Sie das Lagrange-Polynom $L(x)$.**

2. **Berechnen Sie den Wert des Polynoms an der Stelle $x = 3$. Dies soll der Punkt P_3 sein.**

3. **Berechnen Sie das Lagrange-Polynom aus P_0, P_1 und dem ermittelten Punkt P_3 um zu prüfen, ob das selbe Polynom herauskommt.**

. . . . . .
. . . . . .
. . . . . .
. . . . . .

. . . . . .
. . . . . .
. . . . . .
. . . . . .

Hinweise:

- Die Lagrange-Basisfunktionen sind definiert als:

$$\ell_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}.$$

- Das Lagrange-Polynom berechnet sich dann durch:

$$L(x) = \sum_{i=0}^{n} y_i \cdot \ell_i(x),$$

Aufgabe 2: Lagrange-Polynom im Code

Prüfen Sie mit dem folgenden Python-Code ihre Rechnung aus Aufgabe 1. Sie können es visuell mit dem Code aus `lagrange.py` machen, indem Sie das Lagrange Polynom für die gegebenen Punkte plotten.

```
def lagrange(x, x_i, y_i):
    n = len(x_i)
    m = len(x)
    y = np.zeros(m)
    for i in range(n):
        p = lagrange_polynomial(i, x, x_i)
        y += y_i[i] * p
    return y

def lagrange_polynomial(j, x, x_i):
    n = len(x_i)
    p = 1
    for m in range(n):
        if m != j:
            p *= (x - x_i[m]) / (x_i[j] - x_i[m])
    return p
```