

Modulare Polynomarithmetik: Evaluation und Interpolation

Zwei Polynome $f(X)$ und $g(X)$ vom Grad ≤ 3 sollen miteinander multipliziert werden. Das Produkt $h(X) = f(X)g(X)$ ist ein Polynom vom Grad ≤ 7 .

```
> f := X -> 5*X^3+3*X^2-4*X+3;
      f := X -> 5 X^3 + 3 X^2 - 4 X + 3
> g := X -> 2*X^3-5*X^2+7*X-2;
      g := X -> 2 X^3 - 5 X^2 + 7 X - 2
> h := unapply(expand(f(X)*g(X)),X);
      h := X -> 10 X^6 - 19 X^5 + 12 X^4 + 37 X^3 - 49 X^2 + 29 X - 6
```

Um $h(X)$ nach dem Schema von Evaluation und Interpolation zu berechnen, genügen also 8 Interpolationsstellen, die im Prinzip beliebig gewählt werden können.

Die Interpolationspunkte:

```
> points := [seq(X,X=0..7)];
      points := [0, 1, 2, 3, 4, 5, 6, 7]
```

Die Werte der Polynome $f(X)$ und $g(X)$ an den Interpolationspunkten:

```
> valuesf := map(f,points);
      valuesf := [3, 7, 47, 153, 355, 683, 1167, 1837]
> valuesg := map(g,points);
      valuesg := [-2, 2, 8, 28, 74, 158, 292, 488]
```

Die Werte des Polynoms $h(X)$ sind die Produkte der entsprechenden Werte von $f(X)$ und $g(X)$

```
> zip((x,y)-> x*y,valuesf,valuesg);
      [-6, 14, 376, 4284, 26270, 107914, 340764, 896456]
```

Durch Interpolation erhält man das gesuchte Polynom:

```
> interp(points,%,X);
      10 X^6 - 19 X^5 + 12 X^4 + 37 X^3 - 49 X^2 + 29 X - 6
```

Man hätte die Interpolationspunkte auch zufällig wählen können. Beispielsweise mit Hilfe eines Zufallsgenerators. Man muss sich dann nur vergewissern, dass die Punkte paarweise verschieden sind.

```

> die := rand(-99..99);

die := proc()
local t;
global _seed;
  _seed := irem(a * _seed, p);
  t := _seed;
  to concats do _seed := irem(a * _seed, p); t := s * t + _seed end do;
  irem(t, divisor) + offset
end proc
> randpoints := [seq(die(),X=0..7)];
      randpoints := [-85, -55, -37, -35, 97, 50, 79, 56]
> valuesf := map(f,randpoints);
valuesf := [-3048607, -822577, -249007, -210557, 4591207, 632303, 2483605, 887267]
> valuesg := map(g,randpoints);
valuesg := [-1264972, -348262, -108412, -92122, 1778978, 237848, 955424, 335942]
> zip((x,y)-> x*y,valuesf,valuesg);

[3856402494004, 286472311174, 26995346884, 19396931954, 8167656246446,
150392003944, 2372895823520, 298070250514]
> interp(randpoints,%,X);
      10 X6 - 19 X5 + 12 X4 + 37 X3 - 49 X2 + 29 X - 6

```

Bei der Diskreten Fouriertransformation der Ordnung n wählt man als Interpolationspunkte die sogenannten n -ten Einheitswurzeln in der komplexen Ebene. Das sind die n Lösungen der Gleichung $X^n = 1$, diese liegen auf dem Einheitskreis; es handelt sich um die n (verschiedenen) Potenzen der sogenannten "primitiven" n -ten Einheitswurzel $e^{2\pi i/n}$.

Im Falle des Beispiels arbeitet man mit $n=8$.

```

> dftpoints := [solve(X^8=1)];
dftpoints := [-I, I, -1, 1, -1/2*sqrt(2) - 1/2*I*sqrt(2), 1/2*sqrt(2) + 1/2*I*sqrt(2), -1/2*sqrt(2) + 1/2*I*sqrt(2), 1/2*sqrt(2) - 1/2*I*sqrt(2)]

```

Die Liste der 8 Punkte ist nicht in fortlaufenden Potenzen geordnet. Um das explizit zu machen:

```

> omega8 := (1+I)/sqrt(2);
      omega8 := (1/2 + 1/2*I)*sqrt(2)
> dftpoints := [seq(evalc(omega8^k),k=0..7)];
dftpoints := [1, 1/2*sqrt(2) + 1/2*I*sqrt(2), I, -1/2*sqrt(2) + 1/2*I*sqrt(2), -1, -1/2*sqrt(2) - 1/2*I*sqrt(2), -I, 1/2*sqrt(2) - 1/2*I*sqrt(2)]

```

Die Polynome $f(X)$ und $g(X)$ und des Produktes $h(X)$ haben an diesen Stellen komplexe Werte.

```
> valuesf := map(evalc,map(f,dftpoints));

valuesf := [7, -9/2*sqrt(2)+3+I(1/2*sqrt(2)+3), -9I, 9/2*sqrt(2)+3+I(1/2*sqrt(2)-3), 5,
9/2*sqrt(2)+3+I(-1/2*sqrt(2)+3), 9I, -9/2*sqrt(2)+3+I(-1/2*sqrt(2)-3)]
> valuesg := map(evalc,map(g,dftpoints));

valuesg := [2, 5/2*sqrt(2)-2+I(9/2*sqrt(2)-5), 3+5I, -5/2*sqrt(2)-2+I(9/2*sqrt(2)+5), -16,
-5/2*sqrt(2)-2+I(-9/2*sqrt(2)-5), 3-5I, 5/2*sqrt(2)-2+I(-9/2*sqrt(2)+5)]
> valuesh := map(evalc,map(h,dftpoints));

valuesh := [14, 11/2*sqrt(2)-18+I(-59+85/2*sqrt(2)), 45-27I, -11/2*sqrt(2)-18+I(59+85/2*sqrt(2)),
-80, -11/2*sqrt(2)-18+I(-59-85/2*sqrt(2)), 45+27I, 11/2*sqrt(2)-18+I(59-85/2*sqrt(2))]
```

Die Werte von $h(X)$ erhält man auch durch Bildung der Produkte der entsprechenden Werte von $f(X)$ und $g(X)$

```
> zip((x,y)-> evalc(x*y),valuesf,valuesg);

[14, 11/2*sqrt(2)-18+I(-59+85/2*sqrt(2)), 45-27I, -11/2*sqrt(2)-18+I(59+85/2*sqrt(2)), -80,
-11/2*sqrt(2)-18+I(-59-85/2*sqrt(2)), 45+27I, 11/2*sqrt(2)-18+I(59-85/2*sqrt(2))]
```

Die Berechnung (der Koeffizienten) von $h(X)$ kann man in diesem Fall statt durch Interpolation viel einfacher durch Anwendung der inversen Fouriertransformation machen. Dazu interpretiert man die Folge der Funktionswerte als Koeffizienten eines Polynoms $H(X)$. (Jetzt kommt es auf die richtige Reihenfolge an!)

```
> convert(zip((x,y)->x*y,valuesh,[seq(X^k,k=0..7)]),'+');

14 + (11/2*sqrt(2)-18+I(-59+85/2*sqrt(2)))X + (45-27I)X^2
+ (-11/2*sqrt(2)-18+I(59+85/2*sqrt(2)))X^3 - 80X^4
+ (-11/2*sqrt(2)-18+I(-59-85/2*sqrt(2)))X^5 + (45+27I)X^6
+ (11/2*sqrt(2)-18+I(59-85/2*sqrt(2)))X^7
> H := unapply(%,X);
```

$$\begin{aligned}
H := X \rightarrow & 14 + \left(\frac{11}{2}\sqrt{2} - 18 + I\left(-59 + \frac{85}{2}\sqrt{2}\right)\right) X + (45 - 27I) X^2 \\
& + \left(-\frac{11}{2}\sqrt{2} - 18 + I\left(59 + \frac{85}{2}\sqrt{2}\right)\right) X^3 - 80 X^4 \\
& + \left(-\frac{11}{2}\sqrt{2} - 18 + I\left(-59 - \frac{85}{2}\sqrt{2}\right)\right) X^5 + (45 + 27I) X^6 \\
& + \left(\frac{11}{2}\sqrt{2} - 18 + I\left(59 - \frac{85}{2}\sqrt{2}\right)\right) X^7
\end{aligned}$$

Die Auswertung geschieht wiederum an den achten Einheitswurzeln, nun aber im umgekehrten Durchlaufungssinn:

```
> inversedftpoints :=[seq(evalc(omega8^(-k)),k=0..7)];
```

```
inversedftpoints :=
```

```
[1,  $\frac{1}{2}\sqrt{2} - \frac{1}{2}I\sqrt{2}$ , -I,  $-\frac{1}{2}\sqrt{2} - \frac{1}{2}I\sqrt{2}$ , -1,  $-\frac{1}{2}\sqrt{2} + \frac{1}{2}I\sqrt{2}$ , I,  $\frac{1}{2}\sqrt{2} + \frac{1}{2}I\sqrt{2}$ ]
```

```
> valuesH := map(expand,map(evalc,map(H,inversedftpoints)));
```

```
valuesH := [-48, 232, -392, 296, 96, -152, 80, 0]
```

Nach Divisions durch den Skalierungsfaktor 8 erhält man in der Tat die Koeffizienten des Polynoms $h(X)$.

```
> map(x->1/8*x,%);
```

```
[-6, 29, -49, 37, 12, -19, 10, 0]
```

Im Beispielfall $n=8$ kann man exakt rechnen, da ω_8 mittels Wurzeln exakt ausgedrückt werden kann. Das ist im allgemeinen nicht mehr der Fall und man muss die Rechnungen (sofern man nicht in endliche Körper und Ringe ausweicht) mit Fliesskamma-Arithmetik, also mit Rundungsfehlern behaftet, durchführen. Zur Illustration folgt das gleiche Beispiel mit der in Maple eingebauten FFT-Funktion.

Komplexe FFT in Maple

Liste der Koeffizienten des Polynoms $f(X)$:

```
> fcoeffs := array([seq(coeff(f(X),X,k),k=0..7)]);  
fcoeffs := [3, -4, 3, 5, 0, 0, 0, 0]
```

Die Koeffizienten müssen in Real- und Imaginärteile zerlegt werden:

```
> reell := fcoeffs;  
reell := fcoeffs  
> imag := array([0$8]);  
imag := [0, 0, 0, 0, 0, 0, 0, 0]
```

Durchführung der DFT der Ordnung $8 = 2^3$ mittels FFT:

```
> FFT(3,reell,imag);  
8
```

Man erhält im Frequenzbereich die Liste der Werte auch nach Real- und Imaginärteil getrennt:

```
> Freell := copy(reell);  
> print(reell);  
[7.000000001, -3.363961017, 0., 9.363961017, 4.999999999, 9.363961017, 0.,  
-3.363961017]  
> Fimag := copy(imag);  
> print(imag);  
[0, -3.707106773, 8.999999995, 2.292893219, 0., -2.292893219, -8.999999995,  
3.707106773]
```

Mittels inverser Fouriertransformation sollten sich die Koeffizienten von $f(X)$ wiederherstellen lassen. Das gelingt nur bis auf Rundungsfehler!

```
> iFFT(3,reell,imag);  
8  
> op(reell);  
[3.000000000, -3.999999992, 2.999999998, 4.999999989, 0., -.2500000000 10-8,  
.2500000000 10-8, .6250000000 10-8]  
> op(imag);  
[0., -.7500000000 10-9, 0., -.5000000000 10-9, 0., .7500000000 10-9, 0., .5000000000 10-9  
]
```

FFT der Koeffizientenliste von $g(X)$:

```

> gcoeffs := array([seq(coef(g(X),X,k),k=0..7)]);
      gcoeffs := [-2, 7, -5, 2, 0, 0, 0, 0]
> reell := gcoeffs;
      reell := gcoeffs
> imag := array([0$8]);
      imag := [0, 0, 0, 0, 0, 0, 0, 0]
> FFT(3,reell,imag);
      8
> Greell := reell;
> print(reell);
[2, 1.535533906, 3., -5.535533906, -16, -5.535533906, 3., 1.535533906]
> Gimag := imag;
> print(imag);
[0, -1.363961031, -5., -11.36396102, 0, 11.36396102, 5., 1.363961031]

```

Punktweise Multiplikation im Frequenzbereich:

```

> Fcoeffs := zip((x,y) -> x+I*y,Freeell,Fimag);

Fcoeffs := [7.000000001, -3.363961017 - 3.707106773 I, 0. + 8.999999995 I,
9.363961017 + 2.292893219 I, 4.999999999 + 0. I, 9.363961017 - 2.292893219 I,
0. - 8.999999995 I, -3.363961017 + 3.707106773 I]
> Gcoeffs := zip((x,y) -> x+I*y,Greell,Gimag);

Gcoeffs := [2, 1.535533906 - 1.363961031 I, 3. - 5. I, -5.535533906 - 11.36396102 I,
-16, -5.535533906 + 11.36396102 I, 3. + 5. I, 1.535533906 + 1.363961031 I]
> Hcoeffs := zip((x,y)-> evalc(x*y),Fcoeffs,Gcoeffs);

Hcoeffs := [14.000000000, -10.22182538 - 1.104076406 I, 44.99999998 + 26.99999998 I,
-25.77817454 - 119.1040761 I, -79.99999998 - 0. I,
-25.77817454 + 119.1040761 I, 44.99999998 - 26.99999998 I,
-10.22182538 + 1.104076406 I]

```

Vergleich mit der Fouriertransformierten (in floats) $H(X)$ der
Produktpolynoms $h(X)$:

```

> evalf(H(X));
      14. - (10.22182541 - 1.10407638 I) X + (45. - 27. I) X2
      - (25.77817459 - 119.1040764 I) X3 - 80. X4
      - (25.77817459 + 119.1040764 I) X5 + (45. + 27. I) X6
      - (10.22182541 + 1.10407638 I) X7
> Hreal := map(x->Re(x),Hcoeffs);

Hreal := [14.000000000, -10.22182538, 44.99999998, -25.77817454, -79.99999998,
-25.77817454, 44.99999998, -10.22182538]

```

```
> Himag := map(x->Im(x),Hcoeffs);
```

```
Himag := [0., -1.104076406, 26.99999998, -119.1040761, -0., 119.1040761,  
-26.99999998, 1.104076406]
```

Rücktransformation mittels der Werte von $H(X)$:

```
> iFFT(3,Hreal,Himag);
```

8

```
> op(Hreal);
```

```
[-5.999999982, 28.99999992, -48.99999991, 36.99999992, 11.99999998, -18.99999992,  
9.999999938, .5000000000 10-7]
```

```
> op(Himag);
```

```
[0., -.3750000000 10-8, 0., .6250000000 10-8, 0., .3750000000 10-8, 0., -.6250000000 10-8  
]
```

Vergleich mit dem Produktpolynom $h(X)$

```
> h(X);
```

$$10 X^6 - 19 X^5 + 12 X^4 + 37 X^3 - 49 X^2 + 29 X - 6$$