Technische Universität München
Institut für Informatik
Prof. Dr. Hans-Joachim Bungartz
Dipl.-Inf. Christoph Riesinger
Dipl.-Math. Jürgen Bräckle

WS 2013/2014

Numerisches Programmieren, Übungen

Musterlösung 5. Übungsblatt: Diskrete Fourier-Transformation, Schnelle Fourier-Transformation

1) Frequenzanalyse

Die Signale s_1, s_2, s_3 sind folgendermassen mit den Frequenzspektren verknüpft:

$$\begin{aligned} s_1 &= e^{3it} &&\leftrightarrow f_3 \\ s_2 &= 0.2\ i + 0.8\ e^{it} &&\leftrightarrow f_1 \\ s_3 &= e^{it} + 0.2\ e^{12it} &&\leftrightarrow f_2, \end{aligned}$$

wobei $t \in [0; 2\pi]$.
Zusätzliche Begründungen:

s_1 einzelne gleichmässige mittelfrequentierte Schwingung $\rightarrow f_3$

s_2 einzelne langsame Schwingung + leichte Verschiebung nach oben $\rightarrow f_1$

s_3 dominante langsame Schwingung + kleine hochfrequentierte Schwingung $\rightarrow f_2$

2) Eigenschaften der diskreten Fourier-Transformation (DFT)

Wiederholung einiger Regeln beim Rechnen mit komplexen Zahlen ($z \in \mathbb{C}$):

- $z = x + iy, \quad x = Re(z),\ y = Im(z)$
- $\overline{z} = x - iy$ (konjugiert Komplexes), $\quad \overline{z \cdot w} = \overline{z} \cdot \overline{w}$
- $|z| = \sqrt{Re(z)^2 + Im(z)^2} \ = \ \sqrt{x^2 + y^2}$
- e^{it} durchläuft den Einheitskreis gegen den Uhrzeigersinn (beginnend bei (1,0)). Außerdem ist die Funktion 2π-periodisch. Daher gilt:

$$\begin{aligned} e^{i \cdot 0} = e^{i \cdot 2k\pi} &= 1 \in \mathbb{R},\ k \in \mathbb{Z} \\ e^{-i \cdot \pi} &= -1 \in \mathbb{R}. \end{aligned}$$

a) Wir betrachten den Ausdruck auf der rechten Seite:

$$\left(\frac{1}{n}\,\overline{IDFT(\overline{v})}\right)_l = \frac{1}{n}\overline{\sum_{k=0}^{n-1}\overline{v_k}\;\omega^{kl}} = \frac{1}{n}\sum_{k=0}^{n-1}\overline{\overline{v_k}\;\omega^{kl}} = \frac{1}{n}\sum_{k=0}^{n-1}\overline{\overline{v_k}}\;\overline{\omega}^{kl}$$
$$= \frac{1}{n}\sum_{k=0}^{n-1} v_k\;\overline{\omega}^{kl} = DFT(v)_l, \quad l=0,\ldots,n-1.$$

b) Es gilt für alle $k=0,\ldots,n-1$:

$$DFT(v+u)_k = \frac{1}{n}\sum_{j=0}^{n-1}(v+u)_j\;\overline{\omega}^{jk} = \frac{1}{n}\sum_{j=0}^{n-1} v_j\;\overline{\omega}^{jk} + \frac{1}{n}\sum_{k=0}^{n-1} u_j\;\overline{\omega}^{kl} = DFT(v)_k + DFT(u)_k.$$

c) Mit Hilfe der geometrischen Summenformel

$$\sum_{j=0}^{n} z^j = \frac{1-z^{n+1}}{1-z}, \quad \text{für } z \neq 1$$

berechnen wir mit $z=\overline{\omega}^l$:

$$\text{für } l=0: \qquad \sum_{k=0}^{n-1}\overline{\omega}^{0k} = \sum_{k=0}^{n-1} e^0 = n$$
$$\text{für } l=1,\ldots,n-1: \qquad \sum_{k=0}^{n-1}\overline{\omega}^{kl} = \frac{1-(\overline{\omega}^l)^n}{1-\overline{\omega}^l} = \frac{1-e^{-i\frac{2\pi}{n}nl}}{1-e^{-i\frac{2\pi}{n}l}} = 0.$$

Außerdem erhalten wir mit dem gleichen Trick:

$$\sum_{k=0}^{n-1}\omega^{kl}\;\overline{\omega}^{kj} = \sum_{k=0}^{n-1}\omega^{k(l-j)} = \sum_{k=0}^{n-1} e^{i\frac{2\pi}{n}k(l-j)} = \begin{cases} n, & \text{für } l=j \\ 0, & \text{für } l\neq j \end{cases}.$$

d) Wir berechnen mit den Ergebnissen aus c):

$$IDFT(DFT(v))_l = \sum_{k=0}^{n-1} c_k\omega^{kl} = \sum_{k=0}^{n-1}\left(\frac{1}{n}\sum_{j=0}^{n-1} v_j\overline{\omega}^{jk}\right)\omega^{kl}$$
$$= \frac{1}{n}\sum_{j=0}^{n-1} v_j\left(\sum_{k=0}^{n-1}\overline{\omega}^{jk}\omega^{kl}\right) \overset{c)}{=} \frac{1}{n}\sum_{j=0}^{n-1} v_j\,(\delta_{jl}\cdot n) = v_l.$$

e) Für die drei Beispielvektoren erhalten wir folgende Ergebnisse:

- $(DFT(a))_k = \frac{1}{n}\sum_{j=0}^{n-1} a_j\overline{\omega}^{jk} = \frac{1}{n}\omega^0 = \frac{1}{n}, \quad \forall k=0,\ldots,n-1,$
- $(DFT(b))_k = \frac{1}{n}\sum_{j=0}^{n-1} b_j\overline{\omega}^{jk} = \frac{i}{n}\sum_{j=0}^{n-1}\overline{\omega}^{jk} \overset{c)}{=} \begin{cases} i, & \text{für } k=0 \\ 0, & \text{für } k=1,\ldots,n-1 \end{cases},$

- $(DFT(c))_k = \frac{1}{n}\sum_{j=0}^{n-1} c_j \overline{\omega}^{jk} = \frac{1}{n}\sum_{j=0}^{n-1}(-1)^j \overline{\omega}^{jk} = \frac{1}{n}\sum_{j=0}^{n-1}(e^{i\pi})^j \overline{\omega}^{jk}$
$= \frac{1}{n}\sum_{j=0}^{n-1}(e^{i\pi})^j e^{-i\frac{2\pi}{n}jk} = \frac{1}{n}\sum_{j=0}^{n-1} e^{-i\frac{2\pi}{n}j(k-n/2)} \overset{c)}{=} \begin{cases} 1, & \text{für } k = \frac{n}{2} \\ 0, & \text{sonst} \end{cases}$.

3) Schnelle Fourier-Transformation (FFT)

a) Mit der direkten Formel (4) für n=4 und $\omega = e^{i\frac{2\pi}{n}}$ erhalten wir:

$$\begin{aligned}
v_0 &= c_0\omega^{0\cdot 0} + c_1\omega^{0\cdot 1} + c_2\omega^{0\cdot 2} + c_3\omega^{0\cdot 3} = c_0\omega^0 + c_1\omega^0 + c_2\omega^0 + c_3\omega^0 \\
v_1 &= c_0\omega^{1\cdot 0} + c_1\omega^{1\cdot 1} + c_2\omega^{1\cdot 2} + c_3\omega^{1\cdot 3} = c_0\omega^0 + c_1\omega^1 + c_2\omega^2 + c_3\omega^3 \\
v_2 &= c_0\omega^{2\cdot 0} + c_1\omega^{2\cdot 1} + c_2\omega^{2\cdot 2} + c_3\omega^{2\cdot 3} = c_0\omega^0 + c_1\omega^2 + c_2\omega^4 + c_3\omega^6 \\
v_3 &= c_0\omega^{3\cdot 0} + c_1\omega^{3\cdot 1} + c_2\omega^{3\cdot 2} + c_3\omega^{3\cdot 3} = c_0\omega^0 + c_1\omega^3 + c_2\omega^6 + c_3\omega^9.
\end{aligned}$$

Aufgrund der Eigenschaften der Exponentialfunktion (2π-Periodizität etc., vgl. Aufg. 1)) gilt: $\omega^4 = e^{i2\pi} = 1,\ \omega^2 = e^{i\pi} = -1$. Daher erhalten wir insgesamt:

$$\begin{aligned}
v_0 &= c_0 + c_1 + c_2 + c_3 \\
v_1 &= c_0 + c_1\omega - c_2 - c_3\omega \\
v_2 &= c_0 - c_1 + c_2 - c_3 \\
v_3 &= c_0 - c_1\omega - c_2 + c_3\omega.
\end{aligned}$$

b) Da die Sortierphase schon in Abb. 3 erledigt ist, müssen wir hier lediglich noch die Kombinationsphase mit dem Butterfly-Operator durchführen.
Dafür gilt:

$$\begin{aligned}
Z_0 &= z_0 + \omega^0 z_2 \\
Z_1 &= z_0 - \omega^0 z_2 \\
Z_2 &= z_1 + \omega^0 z_3 \\
Z_3 &= z_1 - \omega^0 z_3
\end{aligned}$$

mit $\omega = e^{i\frac{2\pi}{2}}$. Zu beachten ist hierbei die Abhängigkeit des ω von n. In der ersten Kombinationsrunde ergeben IDFT-Lösungen der Größe 2, also $n = 2$. Damit wenden wir nun den Butterfly ein zweites Mal an und erhalten

$$\begin{aligned}
v_0 &= Z_0 + \omega^0 Z_2 = z_0 + z_2 + (z_1 + z_3) = c_0 + c_1 + c_2 + c_3 \\
v_1 &= Z_1 + \omega^1 Z_3 = z_0 - z_2 + \omega\,(z_1 - z_3) = c_0 + c_1\omega - c_2 - c_3\omega \\
v_2 &= Z_0 - \omega^0 Z_2 = z_0 + z_2 - (z_1 + z_3) = c_0 - c_1 + c_2 - c_3 \\
v_3 &= Z_1 - \omega^1 Z_3 = z_0 - z_2 - \omega\,(z_1 - z_3) = c_0 - c_1\omega - c_2 + c_3\omega
\end{aligned}$$

mit $\omega = e^{i\frac{2\pi}{4}}$, was tatsächlich identisch zum Ergebnis der direkten Formel aus a) ist.

4) Sound-Effekte oder: Wozu brauchen wir das überhaupt?

a) In der beiliegenden Grafik ist der Algorithmus schematisch aufgezeichnet.

Für ein beliebiges Eingabesignal ("Incoming Sound Signal" in der Grafik) soll ein Echo durch eine Faltung mit einem Kernel erzeugt werden.

Wie in der schematischen Zeichnung des Faltungskerns ("Kernel") zu sehen ist, gibt es hierbei einen Identitätspeak ("Identity with height = 1"), der alleine angewandt das ursprüngliche Signal wiedergeben (siehe Teilaufgabe b)) würde. In der Zeichnung wurden desweiteren 2 Echoanhebungen ("Echos") eingeführt, die nicht direkt das Echo zurück geben sondern lediglich einen gewissen verzerrten Anteil vom ursprünglichen Signal (daher Höhe > 1).

Um unseren Algorithmus zu vereinfachen nehmen wir an, dass es nur ein Fenster mit doppelter Kernel-Größe gibt ("Filter window"). Der Kernel (und damit das Filter-Fenster) ist in der Praxis kleiner als $[0, T-1]$. Zum Einen führt das zu einer reduzierten Laufzeit (die Faltung findet nur für die Dauer des Filter-Fensters statt), zum Anderen ist es realistisch anzunehmen, dass die Dauer, in der das Echo auftreten kann, kleiner ist als die Spielzeit des Samples, auf das das Echo angewandt werden soll. Verdoppelt man die Fenstergröße, so müssen wir auch die Kernelgröße verdoppeln um eine komponentenweise Faltung im Frequenzraum zu ermöglichen. Zu guter Letzt überführen wir den Kernel schon jetzt in den Frequenzraum, da er sich im weiteren Verlauf der Signalverarbeitung nicht mehr verändert.

i) Signaleingang:
Die Signalverarbeitung erfolgt mit dem Auffüllen des oberen halben Fensterbereichs.

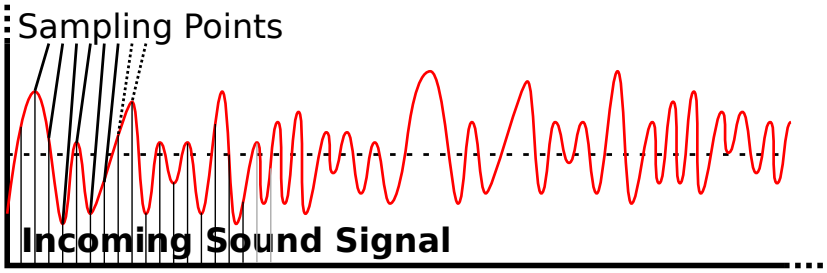
ii) Transformation in den Frequenzraum:
Nachdem ein (weiteres) halbes Fenster an Daten eingelesen wurde und damit die obere Fensterhälfte mit den neuen Daten des Eingabesignals gefüllt ist, verwenden wir unsere FFT um die Signaldaten in den Frequenzraum zu übertragen.

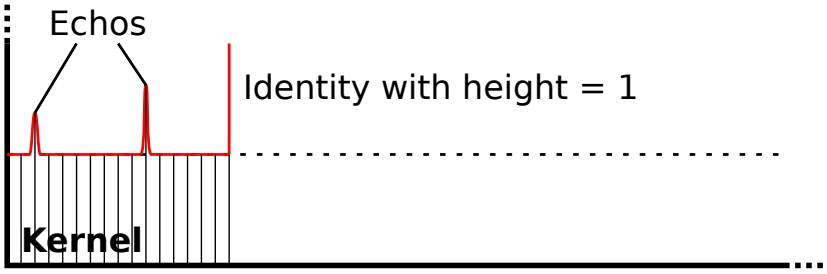
iii) Filter:
Nun können wir komponentenweise den Filter anwenden. Dies benötigt genau soviel Operationen wie das Fenster groß ist, also linear viele Operationen!

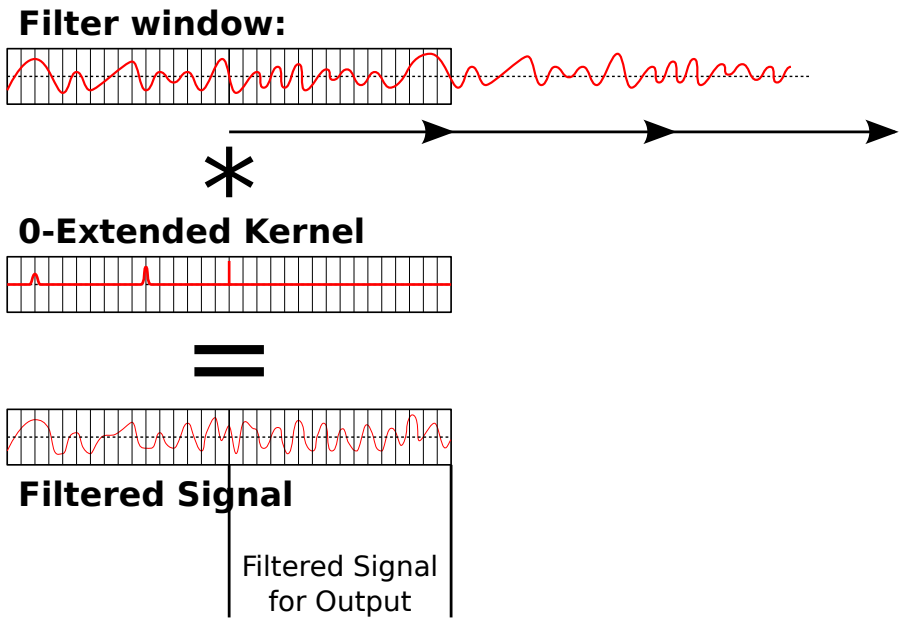
iv) Rücktransformation:
Nach dem Filtern transformieren wir die gefilterten Daten wieder in den Ortsraum (iFFT) und finden in der oberen Hälfte des Fensters unser gewünschtes gefiltertes Signal.

v) Verschieben des Fensters:
Als letzter Schritt muss das Fenster "verschoben" werden was wir einfach dadurch erreichen können indem wir den oberen Teil des Fensters in den unteren Bereich kopieren. Solange noch Eingabedaten vorliegen, wird das Signal ab Schritt 1 weiterverarbeitet.

Ein Beispielprogramm findet sich auf der Website zu dieser Vorlesung (numpro_echo).







b)

$$A(t) = \sum_{i_t=0}^{T-1} I(i_t) \cdot K(t - i_t)$$

Wegen $\kappa_a = \delta_{a,0}$ muss auch $t - i_t = 0$ und $t = i_t$ gelten. Damit erhalten wir:

$$A(t) = \sum_{i_t=0}^{T-1} I(i_t) \cdot \delta_{t-i_t,0} = I(t)$$

c) Zur Veranschaulichung betrachten wir für $c = -6$ den Identitätspeak des Kernel an der

Stelle -6. Veranschaulicht würde das bedeuten, dass das Signal um 6 Samplings versetzt ausgegeben wird (was sich allerdings später als falsch herausstellt!)

Mit

$$A(t) = \sum_{i_t=0}^{T-1} E(i_t) \cdot K(t - i_t)$$

und $\kappa_a = \delta_{a,c}$ erhalten wir somit

$$A(t) = \sum_{i_t=0}^{T-1} E(i_t) \cdot \delta_{t-i_t,c}$$

und mit $t - i_t = c \Leftrightarrow i_t = t - c$

$$A(t) = E(t - c)$$

Was bedeutet das nun für unseren Kernel? Statt dass wir z. B. $c = -6$ einsetzen und damit eine um 6 Zeitschritte versetzte Ausgabe erwarten, bekommen wir das Signal zurück, dass 6 Abtastpunkte in der Zukunft liegt. D. h., dass wir unseren Kernel vor der Transformation in den Frequenzraum an der Achse $x = 0$ spiegeln müssen.

Für Interessierte sei hier auf die Vorlesung AWR 1 verwiesen bei der noch schnellere Methoden (z. B. komponentenweise Faltung mit reelen Zahlen statt komplexen) gezeigt werden: [1].

[1] `http://www5.in.tum.de/wiki/index.php/Algorithms_of_Scientific_Computing_-_Summer_11`