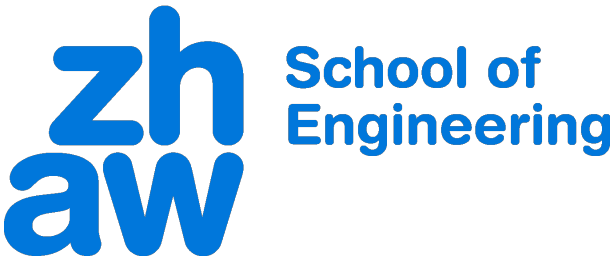
ZHAW Zurich University of Applied Sciences
Winterthur



Zusammenfassung DSV1
Studienwochen 1-14

Written by: Severin Sprenger
(w/ inputs from lienhyan)
June 17, 2026
Zf. DSV1 SW 1-14

1 dB-Rechnen

für Spannungen: $20 \cdot \log_{10}\left(\frac{U_1}{U_0}\right)$

für Ströme: $20 \cdot \log_{10}\left(\frac{I_1}{I_0}\right)$

für Leistungen: $10 \cdot \log_{10}\left(\frac{P_1}{P_0}\right)$

$$\text{dBm} \implies 10 \cdot \log_{10}\left(\frac{P_1}{1\ \text{mW}}\right)$$
$$\text{dBW} \implies 10 \cdot \log_{10}\left(\frac{P_1}{1\ \text{W}}\right)$$
$$\text{dBV} \implies 10 \cdot \log_{20}\left(\frac{U_1}{1\ \text{V}}\right)$$
$$\text{dBmV} \implies 10 \cdot \log_{20}\left(\frac{U_1}{1\ \text{mV}}\right)$$

2 Faltung

$$(s_1 * s_2)(t) = \int_{-\infty}^{\infty} s_1(u) \cdot s_2(t-u)\, du$$

2.1 Schnelle Faltung (Overlap-Add / Save)

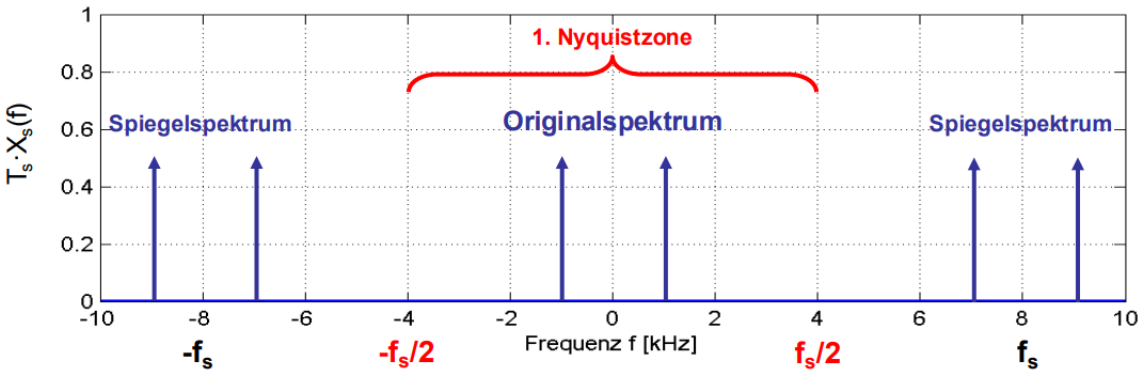
Um zirkulare Faltungsfehler zu vermeiden, muss die FFT-Länge N für einen Signalblock der Länge L und eine Filterimpulsantwort der Länge M sein:

$$N \geq L + M - 1$$

3 Ideale Abtastung

$$x_s(t) = x(t) \cdot \sum_{n=-\infty}^{\infty} \delta(t - n \cdot T_s)$$
$$X_s(f) = X(f) * \left(\frac{1}{T_s} \cdot \sum_{n=-\infty}^{\infty} \delta(f - n \cdot f_s)\right)$$

$$\sum_{n=-\infty}^{\infty} \delta(t - n \cdot T_s) = \sum_{n=-\infty}^{\infty} \frac{1}{T_s} \cdot \exp(j \cdot 2 \cdot \pi \cdot n \cdot f_s \cdot t)$$
$$\implies X_s(f) = \frac{1}{T_s} \cdot \sum_{n=-\infty}^{\infty} X(f - n \cdot f_s)$$



3.1 Signal-to-Quantization-Noise Ratio

$$\text{SQNR} \approx B \cdot 6.02 \text{ dB} + 1.76 \text{ dB}$$
$$\Delta = \frac{U_{max} - U_{min}}{2^B}$$
$$\sigma_e^2 = \frac{\Delta^2}{12}$$

$$\Delta \rightarrow \text{Quantisierungsstufe}, \sigma_e^2 \rightarrow \text{Quantisierungsfehler}$$

Oversampling Gain:
Jede Verdopplung der Abtastrate (OSR) bringt ≈ 3 dB mehr SQNR (effektiv 0.5 Bit).

$$OSR = \frac{f_{os}}{2 \cdot f_{max}} \quad \text{(Oversampling Ratio)}$$
$$\Delta SQNR = 10 \cdot \log_{10}(OSR) \text{ dB}$$

4 DFT

$$m \in [0,\, N-1]$$
$$X[m] = \sum_{n=0}^{N-1} x[n] \cdot \exp\left(-j \cdot \frac{2 \cdot \pi}{N} \cdot m \cdot n\right)$$
$$f[m] = \frac{m \cdot f_s}{N}, \qquad \in [0,\, f_s[$$
$$\Delta f = \frac{f_s}{N}$$
$$\text{Normierte DFT: } c_m = \frac{X[m]}{N}$$
$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \cdot \sum_{m=0}^{N-1} |X[m]|^2$$

4.1 Zero-Padding

Erhöht die Darstellungsauflösung (interpoliert), aber **nicht** die physikalische Frequenzauflösung:

$$\Delta f_{display} = \frac{f_s}{N_{total}}$$

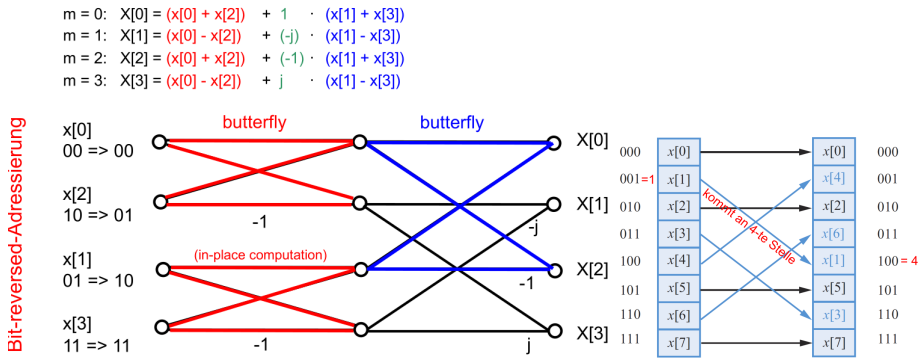
4.2 Symmetrie der DFT (für reelle Signale)

Wenn das Eingangssignal $x[n]$ rein reell ist, ist das Spektrum um die Nyquist-Frequenz ($f_s/2$ bzw. Index $N/2$) symmetrisch:

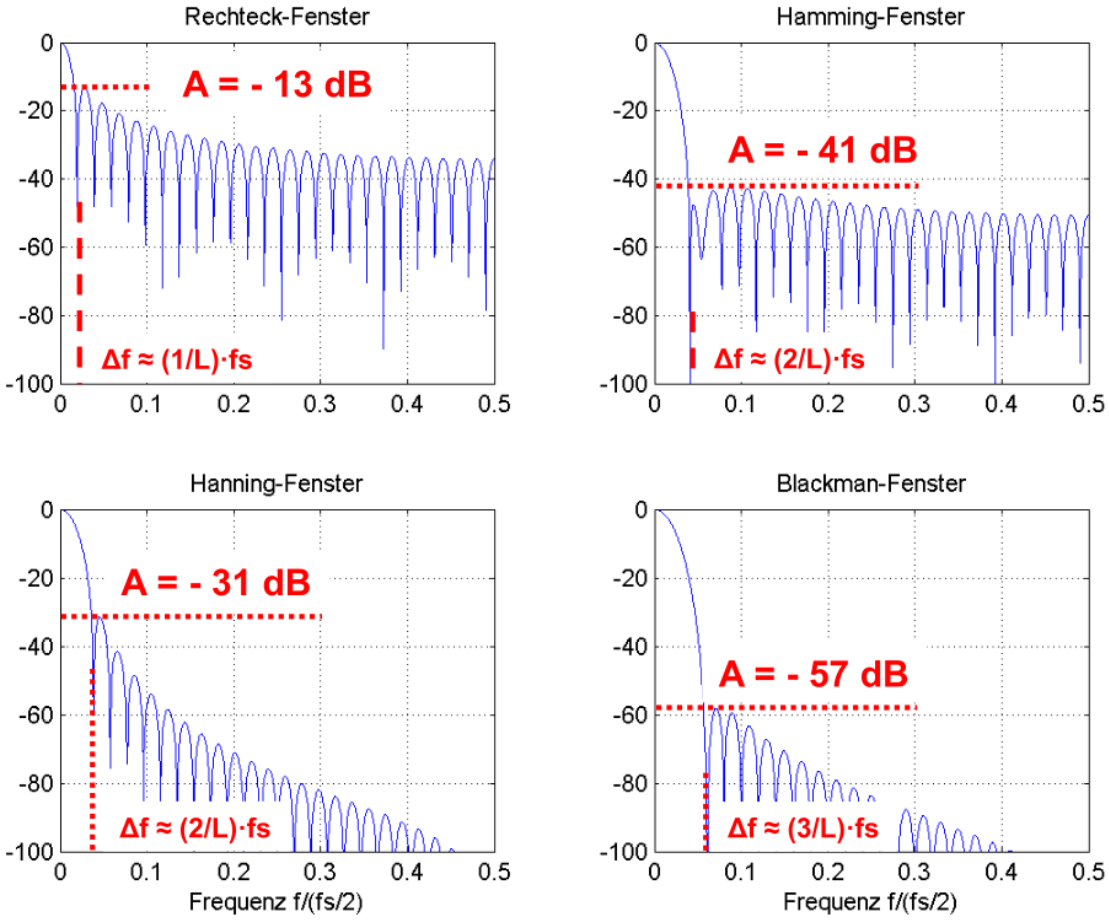
- **Betrag (gerade Symmetrie):** $|X[m]| = |X[N-m]|$
- **Phase (ungerade Symmetrie):** $\angle X[m] = -\angle X[N-m]$

Trick zum Skizzieren: Berechne nur die Werte von $m = 0$ bis $m = N/2$. Die Werte von $N/2+1$ bis $N-1$ sind exakt gespiegelt!

5 FFT (radix 2)



6 Window Functions



6.1 Fensterfunktionen (Zeitbereich)

Für FIR-Filterentwurf zentriert um $n = 0$ (Wertebereich: $-N/2 \leq n \leq N/2$):

$$\begin{aligned}
\textbf{Rechteck:} \quad & w[n] = 1 \\
\textbf{Hanning:} \quad & w[n] = 0.5 + 0.5 \cdot \cos\left(\frac{2\pi n}{N}\right) \\
\textbf{Hamming:} \quad & w[n] = 0.54 + 0.46 \cdot \cos\left(\frac{2\pi n}{N}\right) \\
\textbf{Blackman:} \quad & w[n] = 0.42 + 0.5 \cdot \cos\left(\frac{2\pi n}{N}\right) + 0.08 \cdot \cos\left(\frac{4\pi n}{N}\right)
\end{aligned}$$

Achtung: Bei DFT-Anwendungen ($0 \leq n \leq N-1$) wird oft das Minus-Zeichen ($-$) in den Cosinus-Termen verwendet, da das Fenster dort am Rand beginnt!

6.2 FIR-Filterentwurf (Fenstermethode)

Schritt-für-Schritt Anleitung zur Berechnung der Filterkoeffizienten b_k:

1. **Ideale Impulsantwort $h_d[n]$ berechnen:** Berechnung der unendlich langen, idealen Impulsantwort (z.B. für einen Tiefpass mit Grenzfrequenz f_{DB}) zentriert um $n = 0$:

$$\begin{aligned}
h_d[n] &= \frac{\sin\left(n \cdot \pi \cdot \frac{f_{DB}}{f_s/2}\right)}{n \cdot \pi} \quad \text{für } n \neq 0 \\
h_d[0] &= \frac{f_{DB}}{f_s/2} \quad \text{(nach L'Hôpital)}
\end{aligned}$$

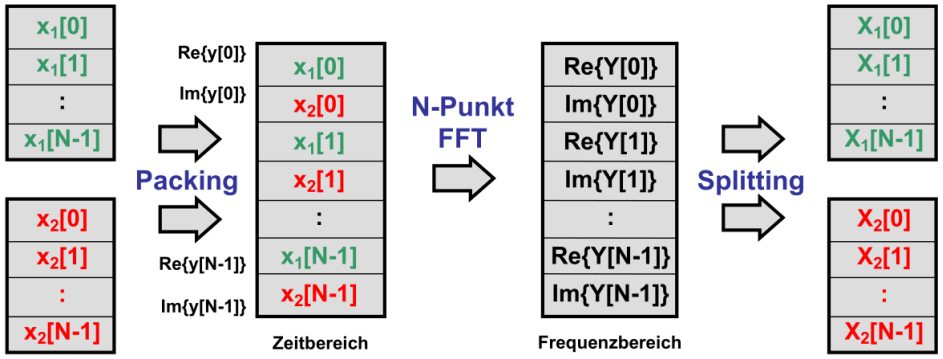
2. **Fensterung (Ausschneiden):** Multiplikation mit der gewählten Fensterfunktion $w[n]$ (Symmetrisch um $n = 0$).

$$h_w[n] = h_d[n] \cdot w[n] \quad \text{für } -\frac{N}{2} \leq n \leq \frac{N}{2}$$

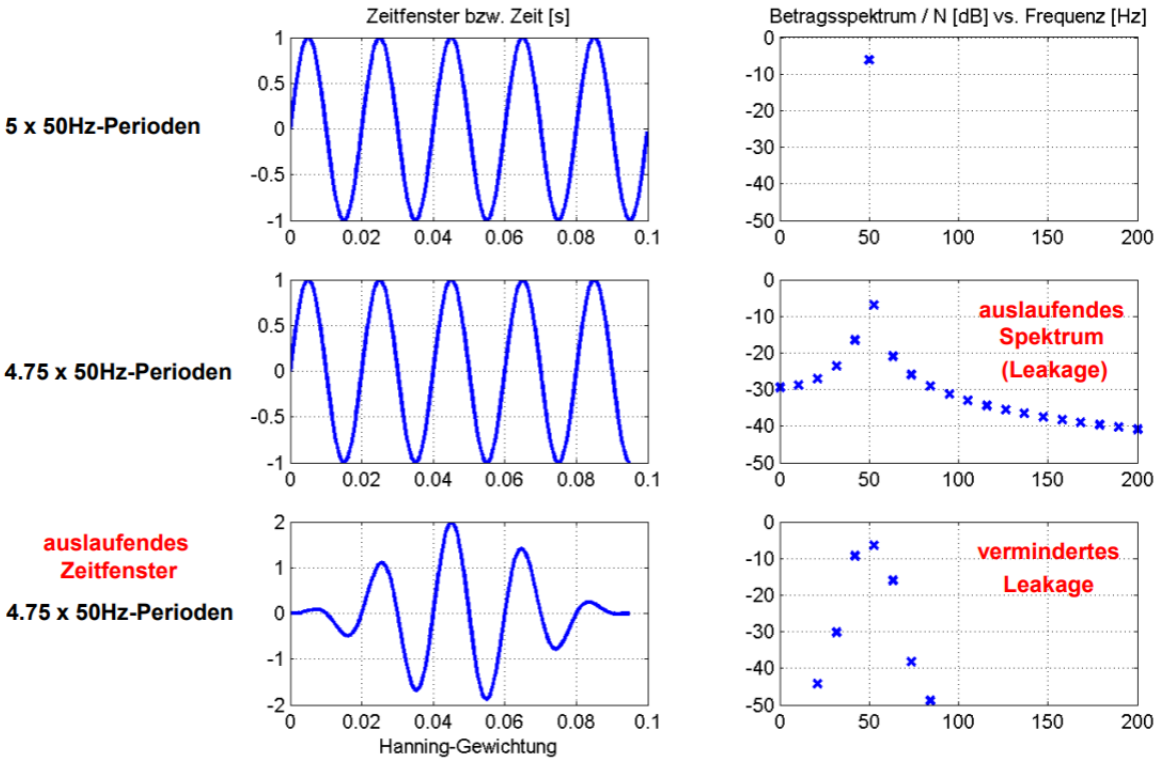
3. **Kausal machen (Zeitverschiebung):** Verschiebung um $N/2$ nach rechts, um kausale Filterkoeffizienten zu erhalten.

$$b_k = h_w\left[k - \frac{N}{2}\right] \quad \text{für } k = 0, 1, \ldots, N$$

7 CFFT



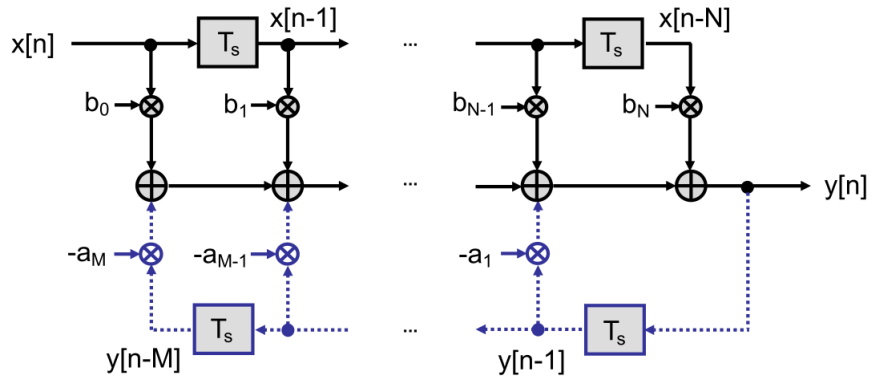
8 Leakage



9 LTD

$$y[n] = \sum_{k=0}^{N} b_k \cdot x[n-k] - \sum_{k=1}^{M} a_k \cdot y[n-k]$$

LTD-System besteht nur aus Elementen für Addition, Multiplikation und Zeitverzögerung



9.1 Sprungantwort (Step Response) und DC-Gain

Die Sprungantwort $s[n]$ ist die Antwort des Systems auf den Einheitssprung $u[n]$ und entspricht der laufenden Summe der Impulsantwort:

$$s[n] = \sum_{k=-\infty}^{n} h[k]$$

Für FIR-Filter ($h[k] = b_k$):
Der stationäre Endwert der Sprungantwort (für $n \geq N$) entspricht der Gleichsignalverstärkung (DC-Gain, bei $f = 0$):

$$s[\infty] = H(e^{j0}) = \sum_{k=0}^{N} b_k$$

Prüfungstipp: Um ein FIR-Tiefpassfilter auf eine DC-Verstärkung von 1 (0 dB) zu normieren, müssen die Koeffizienten so skaliert werden, dass ihre Summe exakt 1 ergibt!

10 FIR

FIR-Filter sind nichtrekursive LTD-Systeme

werden meistens in Transversalstruktur (Direktform 1) realisiert

\+ linearer Phasengang ist realisierbar (keine Signalverzerrungen im Durchlassbereich)

\+ sind immer stabil (alle Pole im Ursprung)

\+ sind toleranter gegenüber Quantisierungseffekten als IIR-Filter

\- weisen viel höhere Filterordnungen als vergleichbare IIR-Filter auf

\- die Zeitverzögerung bzw. Gruppenlaufzeit ist relativ gross

$$h_{FIR}[n] = \{b_0, b_1, \ldots, b_N\}$$

Typ	Symmetrie	Ordnung N	H(0)	H(f_s/2)	Filter
1	sym.	gerade	-	-	alle
2	sym.	ungerade	-	Nullstelle	TP, BP
3	anti-sym.	gerade	Nullstelle	Nullstelle	BP
4	anti-sym.	ungerade	Nullstelle	-	HP, BP

10.1 Gruppenlaufzeit (Group Delay)

$$\tau_g(f) = -\frac{1}{2\pi}\frac{d\phi(f)}{df}$$

Für linearphasige FIR-Filter: $\tau_g = \frac{N}{2} \cdot T_s$

11 IIR

IIR-Filter sind rekursive LTD-Systeme

werden meistens als Biquad-Kaskade realisiert

+ kleine Filterordnung (geringerer Realisierungsaufwand)
+ kleine Zeitverzögerung
- linearer Phasengang für kausale Filter nicht realisierbar
- mehr Probleme mit Quantisierungseffekten als bei FIR-Filter

	Butterworth-Filter	Chebyscheff Filter, Typ 1	Chebyscheff Filter, Typ 2	Elliptisches Filter Cauer	Bessel-Filter
Durchlassbereich	monoton	Rippel	monoton	Rippel	monoton
Sperrbereich	monoton	monoton	Rippel	Rippel	monoton
Steilheit	klein	mittel	mittel	gross	sehr klein
Linearität φ(f)	gross	mittel	mittel	klein	sehr gross

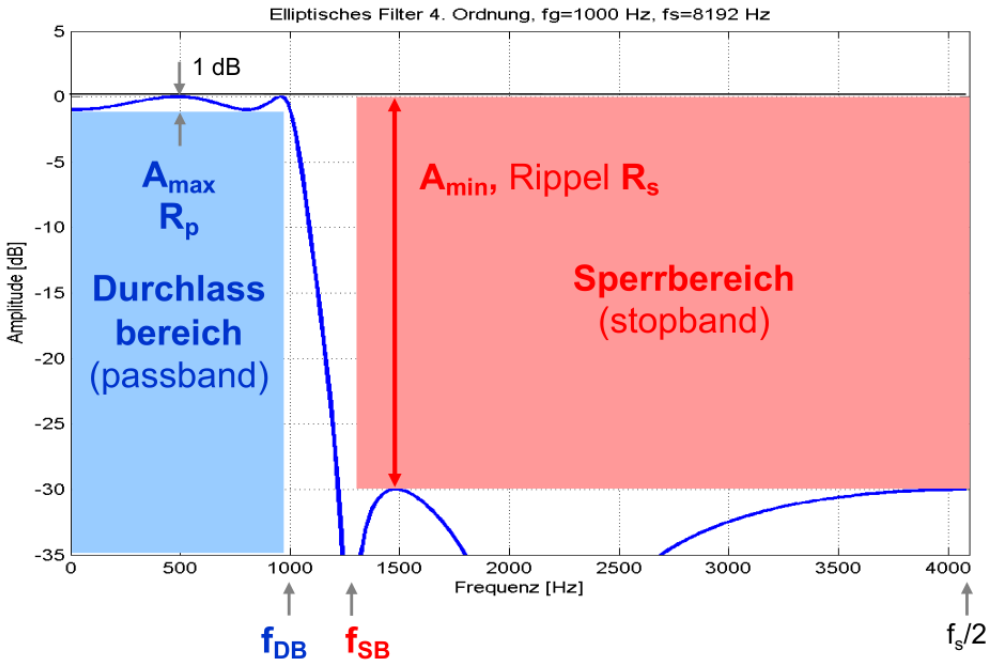
11.1 Bilineare Transformation

$$s = \frac{2}{T_s} \cdot \frac{1 - z^{-1}}{1 + z^{-1}}$$

$$f_a = \frac{1}{\pi \cdot T_s} \cdot \tan\left(\pi \cdot f_d \cdot T_s\right)$$

$f_a \rightarrow$ Vorverzögerte Filterfrequenz

12 Filterspezifikation



12.1 Filter-Transformation (TP → HP Trick)

Ein diskretes Tiefpassfilter kann extrem einfach in ein Hochpassfilter umgewandelt werden, indem das Spektrum um die halbe Abtastfrequenz ($f_s/2$) verschoben wird. Im Zeitbereich entspricht dies der Multiplikation mit $(-1)^n$:

$$h_{HP}[n] = h_{TP}[n] \cdot (-1)^n$$
$$b_{k,HP} = b_{k,TP} \cdot (-1)^k$$

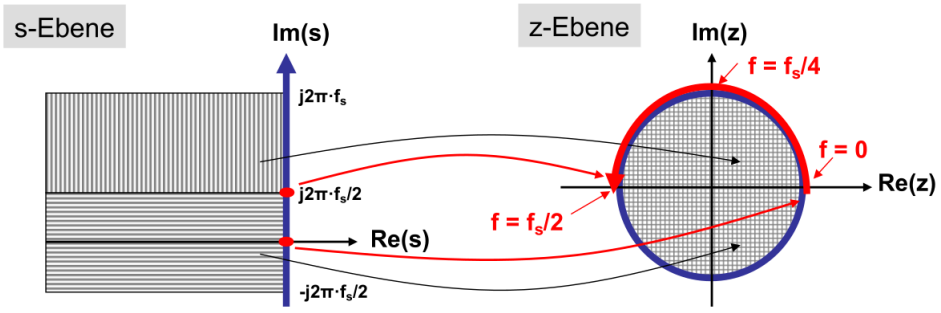
Trick: Drehe einfach bei jedem ungeraden Index ($b_1, b_3, b_5 ...$) das Vorzeichen um!

13 z-Transformation

$$z = e^{s \cdot T_s}$$
$$z = e^{j \cdot 2 \cdot \pi \cdot f \cdot T_s}$$
$$z = e^{j \cdot \Omega}$$

$$X(s) = \sum_{n=-\infty}^{\infty} x[n] \cdot e^{-n \cdot s \cdot T_s}$$
$$X(z) = \sum_{n=-\infty}^{\infty} x[n] \cdot z^{-n}$$
$$X(f) = \sum_{n=-\infty}^{\infty} x[n] \cdot e^{-j \cdot 2 \cdot \pi \cdot n \cdot f \cdot T_s}$$

$$Z^{-1}\left\{\frac{1}{1 - p \cdot z^{-1}}\right\} = p^n \cdot u[n]$$



14 Übertragungsfunktion (UTF)

$$H(z) = \frac{Y(z)}{X(z)}$$
$$H(z) = \frac{\sum_{k=0}^{N} b_k \cdot z^{-k}}{1 + \sum_{k=1}^{M} a_k \cdot z^{-k}}$$
$$H(z) = \frac{b_0 + b_1 \cdot z^{-1} + \cdots + b_N \cdot z^{-N}}{1 + b_1 \cdot z^{-1} + \cdots + b_M \cdot z^{-M}}$$

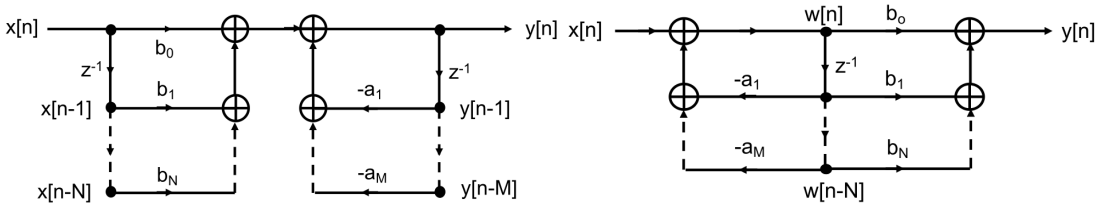
14.1 Direktstrukturen 1 und 2 (IIR-Filter)

Da LTI-Systeme vertauschbar sind, kann die Reihenfolge des nicht-rekursiven (Nullstellen) und rekursiven (Pole) Teils getauscht werden.

- **Direktform 1:** Zwei separate Verzögerungsketten. Benötigt $M + N$ Speicherelemente.
- **Direktform 2 (Kanonische Form):** Gemeinsame Verzögerungskette in der Mitte. Benötigt das Minimum an Speicherelementen: $\max(M, N)$.

Differenzengleichungen für Direktform 2:
Bei Prüfungen wird oft nach dem internen Zustandssignal $w[n]$ (der mittleren Kette) gefragt:

$$w[n] = x[n] - \sum_{k=1}^{M} a_k \cdot w[n-k]$$
$$y[n] = \sum_{k=0}^{N} b_k \cdot w[n-k]$$



14.2 Differenzengleichung

$$y[n] = \sum_{k=0}^{N} b_k \cdot x[n-k] - \sum_{k=1}^{M} a_k \cdot y[n-k]$$
$$Y[z] = \sum_{k=0}^{N} b_k \cdot z^{-k} \cdot X(z) - \sum_{k=1}^{M} a_k \cdot z^{-k} \cdot Y(z)$$

14.3 Stabilität

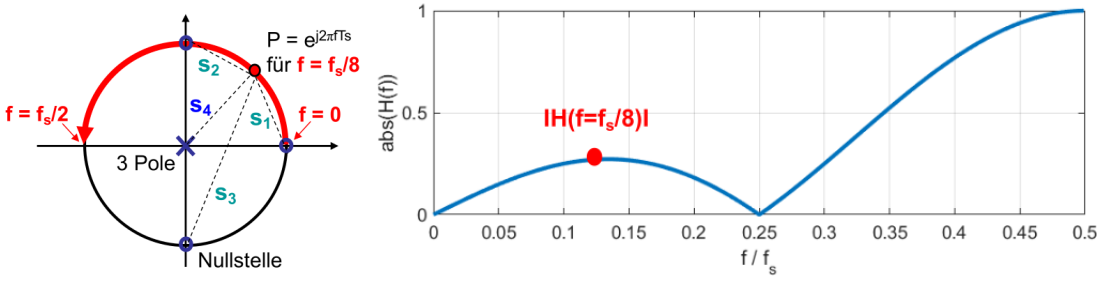
$$\text{BIBO} \rightarrow \sum_{n=-\infty}^{\infty} |h[n]| < \infty$$
$$\text{z-Ebene} \rightarrow |p_k| < 1$$

14.4 Pol-Nullstellen-Darstellung

$$H(z) = b_0 \cdot \frac{\prod_{k=1}^{N}(z - z_k)}{\prod_{k=1}^{M}(z - p_k)} \cdot z^{M-N}$$

$z_k \rightarrow$ Nullstellen

$p_k \rightarrow$ Polstellen



15 Kreuzkorrelation

$$\begin{aligned} r_{xy}[m] &= \sum_{n=-\infty}^{\infty} x[n+m] \cdot y[n] \\ &= \sum_{n=0}^{M-1} x[n+m] \cdot y[n] \end{aligned}$$

$$r_{xy}[m] = r_{xy}[-m]$$

$$y'[m] = a \cdot y[m] \implies r_{xy'}[m] = a \cdot r_{xy}[m]$$

$$y'[m] = y[m] + \mu \implies r_{xy'}[m] = r_{xy}[m] + \sum_{n=0}^{M-1} x[n+m] \cdot \mu$$

$$R_{xy}(z) = X(z) \cdot Y\left(\frac{1}{z}\right)$$

$$R_{xy}(f) = X(f) \cdot Y^*(f)$$

15.1 Laufzeitmessung (Time-Delay Estimation)

Klassisches Radar/Sonar/Echo-Beispiel: Ein Signal $y[n]$ ist eine verzögerte und gedämpfte Version des gesendeten Signals $x[n]$:

$$y[n] = a \cdot x[n - D]$$

- Die **Kreuzkorrelation** $r_{xy}[m]$ hat ihr absolutes Maximum genau bei der Verzögerung (Lag) $m = D$.
- Die physikalische Laufzeit (in Sekunden) berechnet sich dann durch:

$$\tau = D \cdot T_s = \frac{D}{f_s}$$

15.2 Autokorrelation

Falls $x[n] = y[n]$ wird, dann wird $r_{xy}[m]$ als $r_{xx}[m]$ geschrieben. Diese Kreuzkorrelation einer Folge mit sich selbst wird als Autokorrelation bezeichnet.

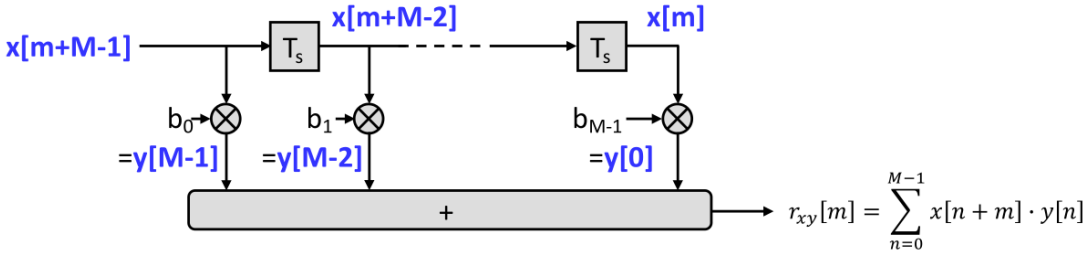
Die Autokorrelation ist ein Mass um die zeitverschonen Ähnlichkeit mit sich selbst zu messen.

Der Wert der Autokorrelation ist wenn $m = 0$ maximal und ist dann gleichwertig wie die Signalleistung.

15.3 Korrelation durch Faltung

$$y'[n] = y[-n]$$
$$(x * y')[m] = r_{xy}[m]$$

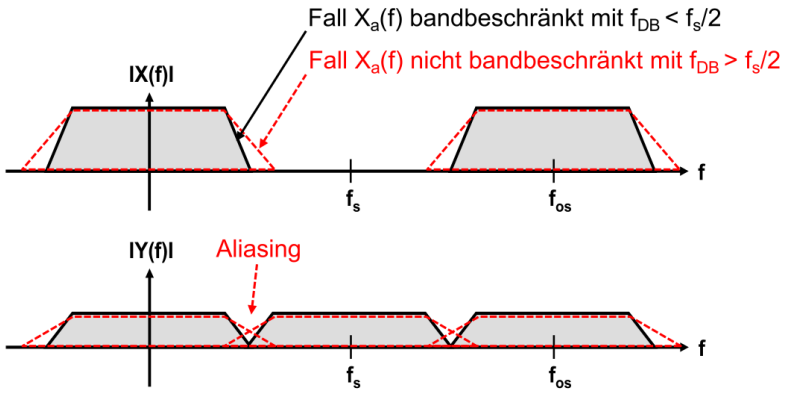
15.4 Korrelation mit FIR-Filter



16 Downsampling

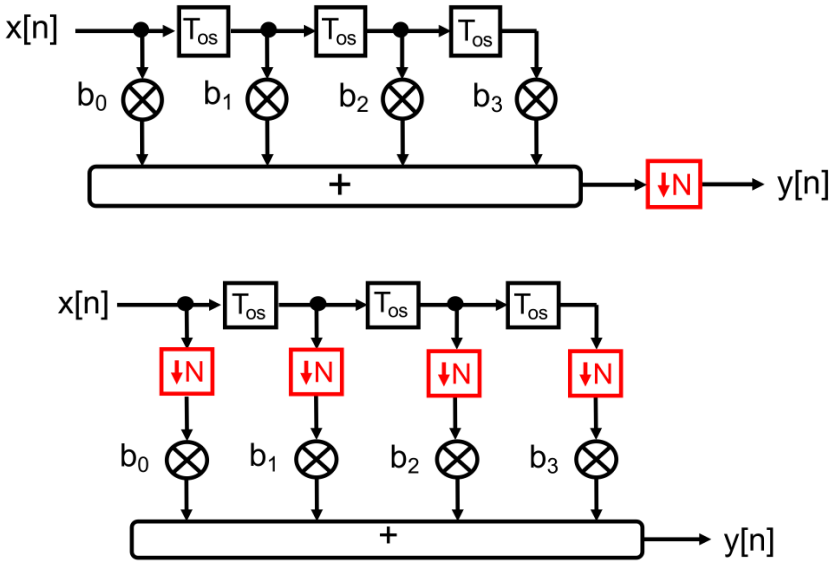
$$x[m] @ f_{os}$$
$$y[m] @ \frac{f_{os}}{N}$$
$$y[m] = x[m * N + n_0]$$
$$n_0 \in [0; N-1]$$

$$X(f) = \frac{1}{T_{os}} \cdot \sum_{n=-\infty}^{\infty} X_a(f - n \cdot f_{os})$$
$$Y(f) = \frac{1}{N \cdot T_{os}} \cdot \sum_{n=-\infty}^{\infty} X_a\left(f - \frac{n \cdot f_{os}}{N}\right)$$



17 Dezimation

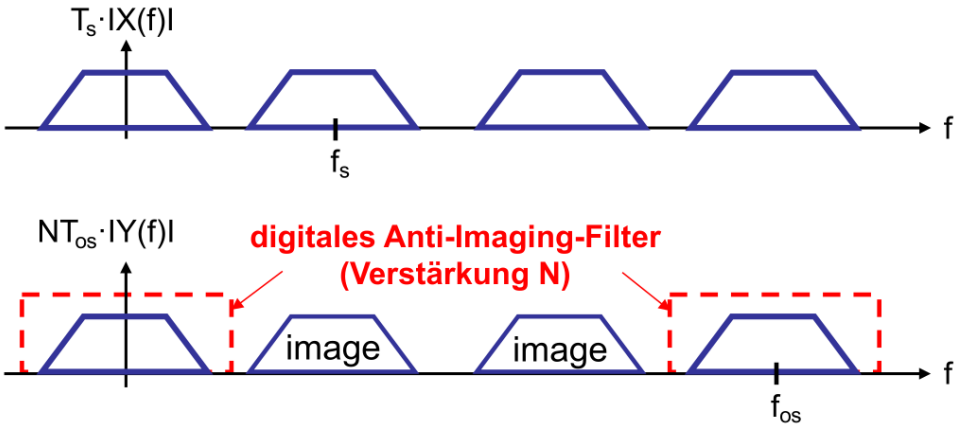
Die Dezimation kombiniert das Prinzip von Downsampling mit einem digitalen Antialiasing-Filter.



18 Upsampling

$$x\,[m]\;@f_s$$
$$y\,[m]\;@N \cdot f_s$$
$$y\,[n] = \begin{cases} x\,[m] & \text{für} n = m \cdot N \\ 0 & \text{sonst} \end{cases}$$

Da Upsampling lediglich Nullen einfügt, bleibt das Spektrum des Signals unverändert. Dies hat jedoch den Effekt, dass die Spiegelspektren des originalen Signals als Images im neuen Signal auftreten.

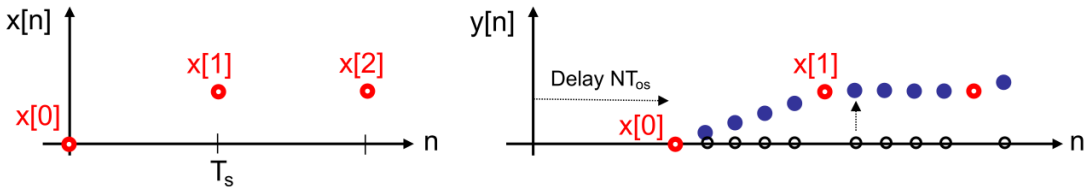


19 Interpolation

Die Interpolation verbindet Upsampling mit einem Anti-Imaging-Filter. Der Filter wird als Tiefpass mit $f_{DB} < \frac{f_s}{2}$ entworfen.

19.1 Lineare Interpolation

In gewissen Anwendungen genügt auch wenn eine lineare Interpolation verwendet wird.



$$H(f) = \frac{1}{N} \cdot \left(\frac{\sin\left(\frac{\pi \cdot f}{f_s}\right)}{\sin\left(\frac{\pi \cdot f}{f_{os}}\right)} \right)^2$$

Aufgrund des Frequenzverhaltens der linearen Interpolation, kann diese nur in Fällen eingesetzt werden wo die Frequenzen im Signal sehr viel kleiner als $\frac{f_s}{2}$ sind.

19.2 Polyphasen-Zerlegung

Erhöht die Recheneffizienz, da die Filterung mit der tieferen Taktrate durchgeführt werden kann.

$$H(z) = \sum_{k=0}^{N-1} z^{-k} \cdot H_k(z^N)$$

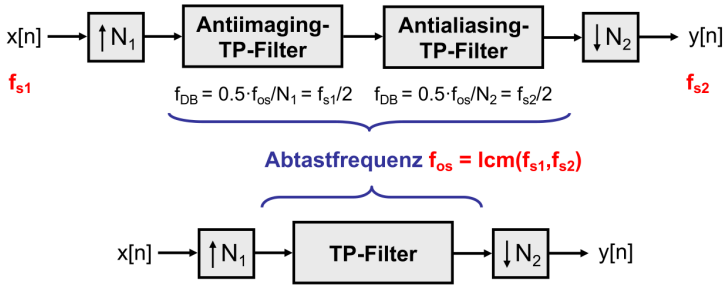
Für N=2 (Gerade/Ungerade Indexierung):

$$H(z) = H_0(z^2) + z^{-1} \cdot H_1(z^2)$$

20 Rationale Ratenverhältnisse

Um rationale Ratenverhältnisse zu erreichen, wird ein signal zuerst mit N_1 Interpoliert und anschliessend mit N_2 Dezimiert. Dabei kann auch das Anti-Imaging-Filter und das Antialiasing-Filter zu einem Filter kombiniert werden.

$$\frac{N_1}{N_2} = \frac{f_{s2}}{f_{s1}}$$



21 CIC-Filter

Besonders effiziente Dezimations-/Interpolationsfilter, da sie **keine Multiplikatoren** benötigen (nur Addierer, Subtrahierer und Verzögerungen).

$$H(z) = \left(\frac{1 - z^{-R}}{1 - z^{-1}}\right)^{M}$$

$R \rightarrow$ Ratenänderungsfaktor, $M \rightarrow$ Anzahl der Stufen.

22 Anhang

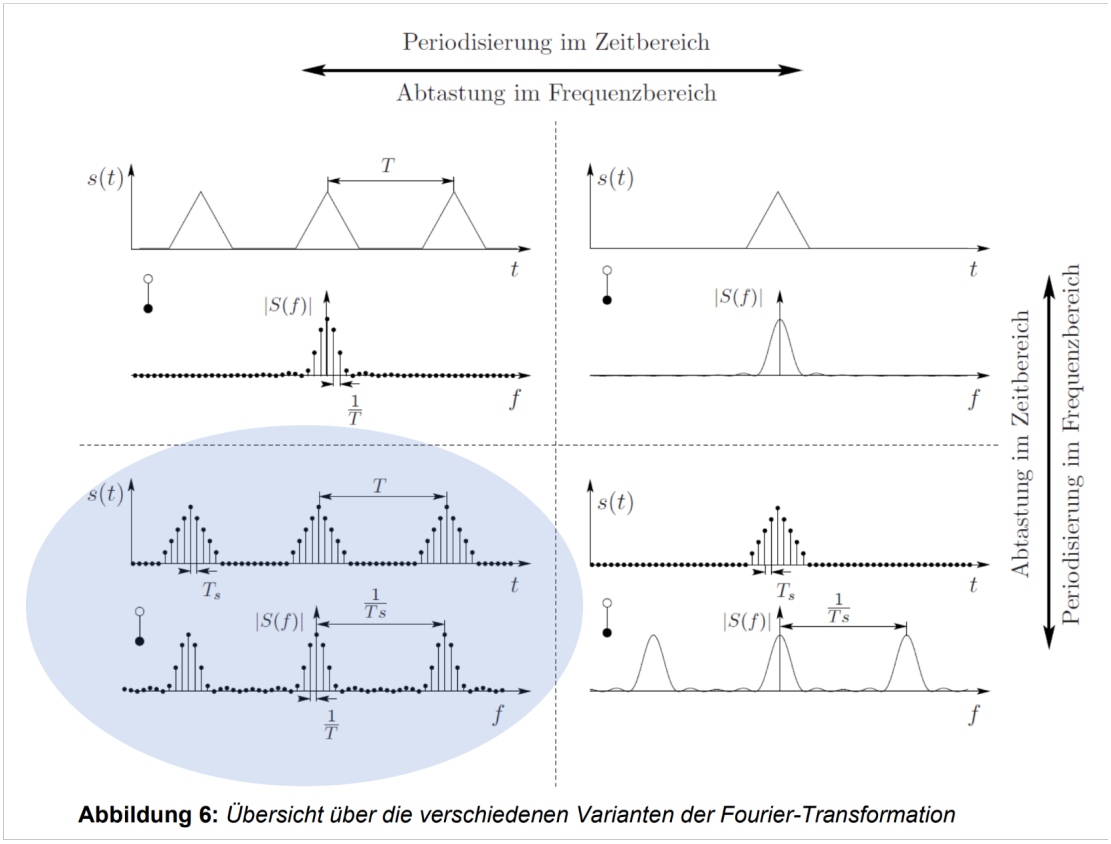
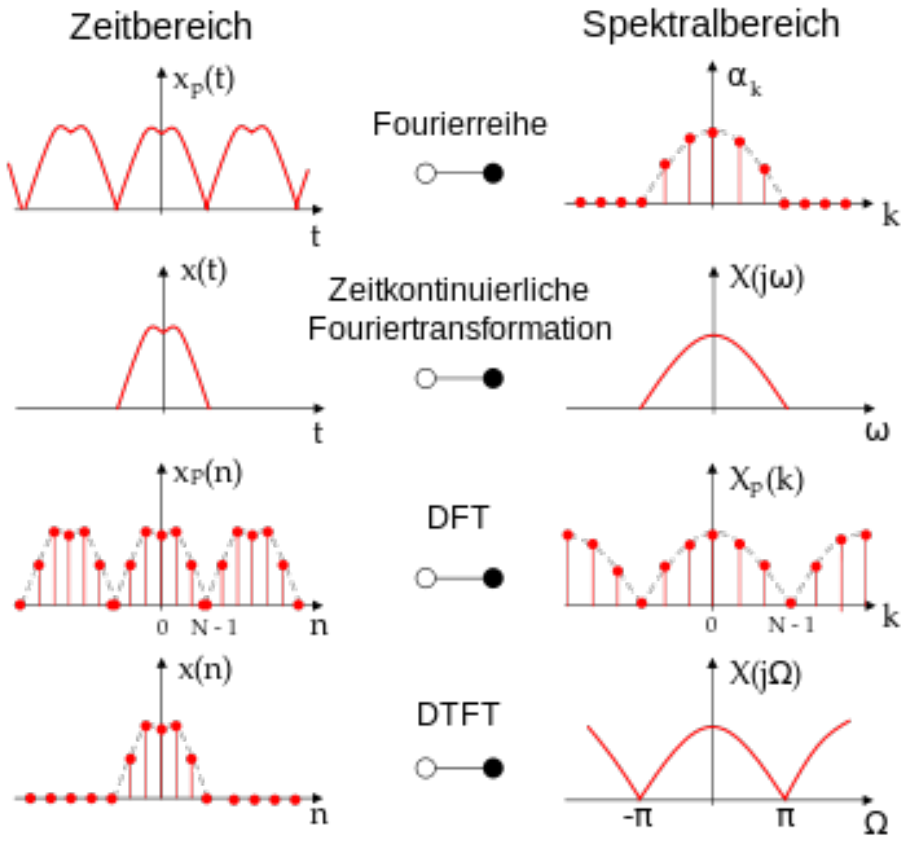


Abbildung 6: *Übersicht über die verschiedenen Varianten der Fourier-Transformation*



22.1 MATLAB Befehlsübersicht

Prüfungsfragen beinhalten oft das Interpretieren von Code. Wichtige Funktionen:

Signale & Transformationen:

- `conv(x, y)`: Lineare Faltung zweier Vektoren (Länge $N_x + N_y - 1$).
- `xcorr(x, y)`: Kreuzkorrelation. Bei `xcorr(x)` Autokorrelation.
- `fft(x, N)` / `ifft(X, N)`: N-Punkt (Inverse) Fast Fourier Transformation.
- `roots(p)`: Findet die Nullstellen eines Polynoms (z.B. für Pol-/Nullstellen).
- `poly(r)`: Erzeugt Polynomkoeffizienten aus gegebenen Nullstellen `r`.

Digitale Filterung & Analyse:

- `filter(b, a, x)`: Filtert Signal `x` mit der Differenzengleichung (Zähler `b`, Nenner `a`).
- `freqz(b, a)`: Berechnet den komplexen Frequenzgang $H(e^{j\Omega})$ des Filters.
- `zplane(b, a)`: Zeichnet das Pol-Nullstellen-Diagramm in der komplexen Ebene.

Filterentwurf:

- `fir1(N, Wn)`: FIR-Filterentwurf (Fenstermethode). *Achtung:* Liefert $N + 1$ Koeffizienten!
- `firpm(N, f, a)`: FIR-Entwurf nach Parks-McClellan (Minimax/Equiripple-Optimierung).
- `butter(N, Wn)`: IIR-Butterworth-Filterentwurf (liefert `b` und `a`). Wn ist auf $f_s/2$ normiert!
- `bilinear(b_a, a_a, fs)`: Bilineare Transformation (Analog → Digital).

Multiraten-Signalverarbeitung:

- `downsample(x, N)` / `upsample(x, N)`: Reine Ratenänderung (ohne Filterung!).
- `decimate(x, N)`: Dezimation **inklusive** internem Anti-Aliasing-Tiefpassfilter.
- `interp(x, N)`: Interpolation **inklusive** internem Anti-Imaging-Tiefpassfilter.

22.2 TI-Nspire CX II-T CAS

Der TI-Nspire hat keine native DSP-Bibliothek, aber seine CAS-Funktionen sind für die z-Transformation und Systemanalyse extrem mächtig:

- **Partialbruchzerlegung (Inverse z-Trafo):**
  Befehl: `expand(Ausdruck, z)`
  Tipp: Den Term zuerst als $X(z)/z$ eingeben und expandieren lassen, danach wieder mit z multiplizieren, um die Form $\frac{z}{z-p}$ zu erhalten!
- **Komplexe Pol-/Nullstellen finden:**
  Befehl: `cZeros(Polynom, z)` oder `cSolve(Gleichung = 0, z)`
  Liefert direkt die komplexen Wurzeln für das Pol-Nullstellen-Diagramm.
- **Komplexe Zahlen umwandeln (Polar ↔ Kartesisch):**
  Am Ende der Zeile `>Polar` oder `>Rect` eingeben (über den Katalog → Complex). Wichtig für Betrag und Phase von $H(e^{j\Omega})$.
- **Faltung / Summen berechnen:**
  Nutze das Summenzeichen Σ im Math-Template für diskrete Summen wie die DFT oder Faltung kurzer Folgen.
- **Listen-Operationen (Vektoren):**
  Arrays in geschweiften Klammern `{1, 2, 3}`. Punktprodukt via `sum({...} * {...})`. Sehr nützlich für Korrelation bei $m = 0$ (Energie).