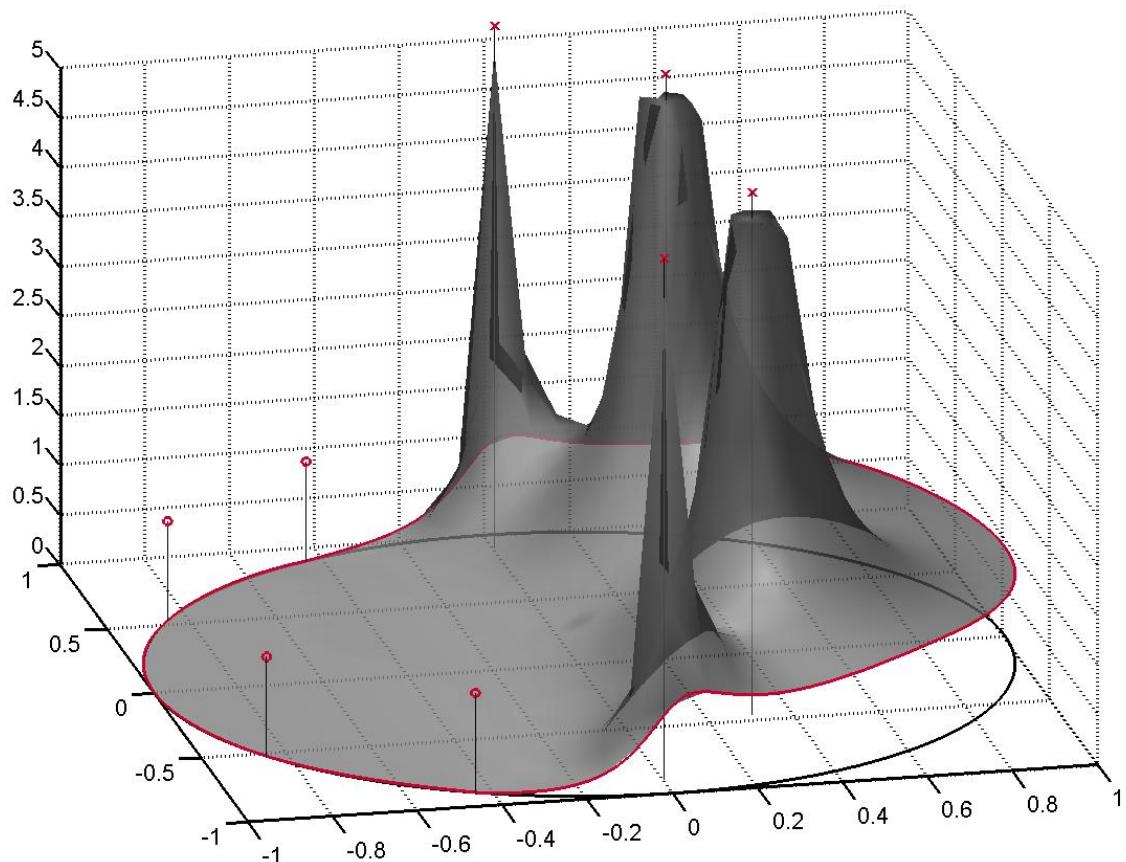


Übungsaufgaben und Python-Kurs zu Digitale Signalverarbeitung auf FPGAs

Prof. Dr. Christian Münker



Sommersemester 2016
15. März 2016

google@chipmuenk.de

Übungsaufgaben und Pythonkurs zu Digitale Signalverarbeitung auf FPGAs

(c) 2008-2016 Prof. Dr. Christian Münker

Überarbeitete Version vom 15. März 2016.

(c) der genialen XKCD-Comics in diesem Skript:
Randall Munroe (<https://xkcd.com/>)
unter CC BY-NC 2.5 Lizenz (<http://creativecommons.org/licenses/by-nc/2.5/>).
Die individuellen URLs sind unter jeder Abbildung abgedruckt.

Diese Unterlagen wurden mit L^AT_EX verfasst, weil ...

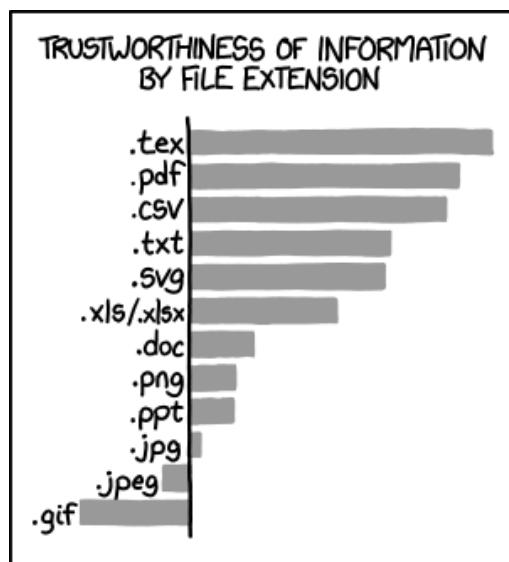


Abb. 1.: [<http://xkcd.com/1301/>]

Inhaltsverzeichnis

Abkürzungs- und Begriffsverzeichnis	ix
Einleitung	1
i Hinweise zu diesem Kurs	1
ii Allgemeine Hinweise zu Python und Matlab	3
I Übungsaufgaben	9
1 LTI: LTI-Systeme im Zeitbereich	11
1.1 * Filterung abgetasteter Signale	18
1.2 * Faltung mit einfachem FIR-Filter	20
1.3 * Systemeigenschaften aus Impulsantwort	20
1.4 * Umformung von SFGs mit graphischen Methoden	21
1.5 * Kritischer Pfad und Ressourcenverbrauch 1	21
1.6 * Kritischer Pfad und Ressourcenverbrauch 2	22
1.7 * Transponierte Systeme zu Aufgabe 1.6	22
1.8 * Allgemeine Fragen zu FPGAs	22
2 LTF: LTI-Systeme im Frequenzbereich	25
2.1 * Filterung des Sensorsignals im Frequenzbereich	28
2.2 * Frequenzgang einfacher FIR-Filter	28
2.3 * Systemfunktion aus Impulsantwort	29
2.4 * Systemeigenschaften aus Übertragungsfunktion	29
2.5 * Systemeigenschaften aus Pol-/Nullstellenplan	30
2.6 * Frequenzgang aus Signalflussdiagramm	30
2.7 * Einfaches Moving Average Filter	31
2.8 * Allgemeine IIR-Struktur	31
2.9 IIR-Filter zweiter Ordnung	32
2.10 Moving Average Filter / rect-Fenster	33
2.11 Notchfilter	33
3 DFT: Diskrete Fouriertransformation und FFT	35
3.1 * Allgemeine Fragen zu DFT und FFT	40
3.2 * DFT des MA-Filters	40
3.3 * Frequenzauflösung der DFT	41
3.4 * Rechenaufwand für DFT und FFT	41
3.5 # Fourierreihe und synchrone DFT	42
3.6 * DFT periodischer Signale mit Python / Matlab	42
3.7 Kohärente DFT einer Rechteckschwingung	43
3.8 Fourier-Analyse mit Rechteck-Fensterung	44

4 FIL: Einfache digitale Filter und FIR-Filterentwurf	45
4.0 * Filterentwurf für Sensorsignal	52
4.1 * Filterspezifikationen	52
4.2 * Amplitudengang linearphasiger Filter	53
4.3 * Maximal-, minimal- und linearphasige Filter	53
4.4 * FIR Halbbandfilter	54
4.5 Filterentwurf	55
4.6 * Filtertransformationen	56
4.7 * Filterimplementierung auf FPGA	56
4.8 * Verständnisfragen zu Filtern	56
5 FIX: Wortlängeneffekte und Fixpoint-Systeme im Zeitbereich	59
5.1 * Analog-Digital-Wandlung eines Drucksensors	63
5.2 * Moving Average Filter mit endlicher Wortbreite	63
5.3 * FIR Filter mit Quantisierung	64
5.4 FIR Filter mit Skalierung	65
5.5 * IIR-Filter mit Skalierung	66
5.6 IIR-Filter mit Quantisierung und kleinen Grenzzyklen	67
5.7 Integrator mit endlicher Wortbreite	67
5.8 * Verständnisfragen zu Filtern mit Quantisierung	68
6 NOI: Wortlängeneffekte im Frequenzbereich: Quantisierungsrauschen	71
6.1 * Quantisierungsrauschen bei Analog-Digital-Wandlung	79
6.2 * SQNR eines quantisierten Signals	79
6.3 * Quantisierungsrauschen im FIR-Filter	80
6.4 * DFT von Breitbandsignalen	81
7 SMP: Abtastung und Downsampling	83
7.1 * Spektren abgetasteter Signale	85
7.2 Analog-Digital-Wandlung mit Oversampling	85
7.3 * Analog-Digital-Wandlung mit Oversampling und nachfolgender Dezimation	86
7.4 * Zweistufige Dezimation	86
7.5 Multiraten-Tiefpassfilter	87
8 INP: Upsampling, Interpolation und Digital-Analog Wandlung	89
8.1 * Wiederholspektren und Images	91
8.2 * Verschiedene Arten der Digital-Analog-Wandlung	91
8.3 * Ideales Interpolationsfilter	94
8.4 * Abtastratenerhöhung mit Nullenstopfen	95
8.5 Oversampling DAC	95
8.6 Zweistufige Interpolation	97
9 SRC: Abtastratenwandlung	99
9.1 Einfache Abtastratenwandlung	101
9.2 * Ideale Abtastratenwandlung	102
9.3 Abtastratenwandlung mit Aliasing im Übergangsbereich	103
10 CIC: Cascaded Integrator-Comb Filter	105
10.1 + CIC-Filter	105
11 IIR: Rekursive Filter	109

11.0	* Filterentwurf für Sensorsignal	111
11.1	* Filtertransformationen	111
11.2	Bilineare Transformation	112
11.3	Resonator	112
12	ZST: Zustandsraum	113
12.0	Python / Matlab	115
12.1	Zustandsgleichungen aus SFG	115
12.2	SFG aus Zustandsgleichungen	116
II	Musterlösungen	117
M1	LTI-Systeme im Zeitbereich	119
M1.1	Filterung abgetasteter Signale	119
M1.2	Faltung mit einfachem FIR-Filter	126
M1.3	Systemeigenschaften aus Impulsantwort	127
M1.4	* Umformung von SFGs mit graphischen Methoden	128
M1.5	Kritischer Pfad und Ressourcenverbrauch	128
M1.6	* Kritischer Pfad und Ressourcenverbrauch 2	130
M1.7	Transponierte Systeme zu Aufgabe 1.6	131
M1.8	Allgemeine Fragen zu FPGAs	132
M2	LTF: LTI-Systeme im Frequenzbereich	135
M2.1	* Filterung des Sensorsignals im Frequenzbereich	135
M2.2	Frequenzgang einfacher FIR-Filter	143
M2.3	* Systemfunktion aus Impulsantwort	145
M2.4	* Systemeigenschaften aus Übertragungsfunktion	145
M2.5	* Systemeigenschaften aus Pol-/Nullstellenplan	148
M2.6	* Systemeigenschaften aus Signalflussdiagramm	149
M2.7	* Einfaches Moving Average Filter	150
M2.8	* Allgemeine IIR-Struktur	154
M2.9	IIR-Filter zweiter Ordnung	161
M2.10	Moving Average Filter / rect-Fenster	164
M2.11	Notchfilter	165
M3	DFT: Diskrete Fouriertransformation und FFT	169
M3.1	* Allgemeine Fragen zu DFT und FFT	169
M3.2	* DFT des MA-Filters	170
M3.3	* Frequenzauflösung der DFT	171
M3.4	* Rechenaufwand für DFT und FFT	172
M3.5	# Fourierreihe und synchrone DFT	173
M3.6	* DFT periodischer Signale mit Python / Matlab	178
M3.7	Kohärente DFT einer Rechteckschwingung	182
M3.8	Fourier-Analyse mit Rechteck-Fensterung	183
M4	FIL: Einfache digitale Filter und FIR-Filterentwurf	187
M4.1	* Filterspezifikationen	187
M4.2	* Amplitudengang linearphasiger Filter	190
M4.3	* Maximal-, minimal- und linearphasige Filter	193
M4.4	* FIR Halbbandfilter	201

M4.5 Filterentwurf	203
M4.6 * Filtertransformationen	203
M4.7 Filterimplementierung auf FPGAs	205
M4.8 Verständnisfragen zu Filtern	206
M5 FIX: Wortlängeneffekte und Fixpoint-Systeme im Zeitbereich	207
M5.1 * Analog-Digital-Wandlung eines Drucksensors	207
M5.2 * Moving Average Filter mit endlicher Wortbreite	208
M5.3 * FIR Filter mit Quantisierung	209
M5.4 FIR Filter mit Skalierung	212
M5.5 IIR-Filter mit Skalierung	213
M5.6 IIR-Filter mit Quantisierung und kleinen Grenzzyklen	216
M5.7 Integrator mit endlicher Wortbreite	217
M5.8 * Verständnisfragen zu Filtern mit Quantisierung	219
M6 NOI: Wortlängeneffekte im Frequenzbereich: Quantisierungsrauschen	221
M6.1 Quantisierungsrauschen Analog-Digital-Wandlung	221
M6.2 * SQNR eines quantisierten Signals	222
M6.3 * Quantisierungsrauschen im FIR-Filter	223
M6.4 * DFT von Breitbandsignalen	225
M7 SMP: Abtastung und Downsampling	229
M7.1 * Spektren abgetasteter Signale	229
M7.2 Analog-Digital-Wandlung mit Oversampling	230
M7.3 * Analog-Digital-Wandlung mit Oversampling und nachfolgender Dezimation	231
M7.4 * Zweistufige Dezimation	231
M7.5 Multiraten-Tiefpassfilter	233
M8 INP: Upsampling, Interpolation und Digital-Analog Wandlung	237
M8.1 * Wiederholspektren und Images	237
M8.2 * Verschiedene Arten der Digital-Analog-Wandlung	237
M8.3 * Ideales Interpolationsfilter	240
M8.4 * Abtastratenerhöhung mit Nullenstopfen	241
M8.5 * Oversampling DAC	241
M8.6 Zweistufige Interpolation	243
M9 SRC: Abtastratenwandlung	245
M9.1 * Einfache Abtastratenwandlung	245
M9.2 * Ideale Abtastratenwandlung	246
M9.3 Abtastratenwandlung mit Aliasing im Übergangsbereich	251
M10 CIC: Cascaded Integrator-Comb Filter	253
M10.1 + CIC-Filter	253
M11 IIR: Rekursive Filter und -entwurf	257
M11.1 * Filtertransformationen	257
M11.2 Bilineare Transformation	259
M11.3 Resonator	261
M12 ZST: Zustandsraum	263
M12.1 Zustandsgleichungen aus SFG	263

M12.2 SFG aus Zustandsgleichungen	264
III Anhang	267
A Wichtige Formeln	269
A.1 Komplexe Ebene	269
A.2 Trigonometrische und Eulersche Identitäten	269
A.3 Exponentialfunktion und Logarithmus	270
A.4 Quadratische Gleichung	270
A.5 Summenformeln	270
A.5.1 Geometrische Reihe	270
A.5.2 Verwandte Reihen	272
A.5.3 Einfache Reihen	272
A.6 Reihenentwicklung	273
A.7 Ein paar Integrale	273
B Fourier-Transformation	275
B.1 Fourier-Transformation kontinuierlicher Signale	275
B.2 Fourier-Transformation diskreter Signale	277
C z-Transformation	279
Abbildungsverzeichnis	280
Tabellenverzeichnis	285
Listings	287
Literaturempfehlungen	289
Literaturverzeichnis	294

Abkürzungs- und Begriffsverzeichnis

Begriff	Erklärung
ADC	Analog-to-Digital-Converter
Aliasing	Bei der Abtastung eines Signals mit f_S werden Spektralanteile oberhalb $f_S/2$ ins → Basisband abgebildet, diesen Vorgang nennt man A. Die hierdurch entstandenen Verfälschungen lassen sich im Allgemeinen nicht rückgängig machen; vor Abtastung oder Reduktion der Abtastrate muss daher die Bandbreite des Signals mit einem <i>Anti-Aliasingfilter</i> auf $f_S/2$ begrenzt werden, damit das <i>rightarrow</i> Nyquistkriterium erfüllt ist.
Basisband	Mit f_S abgetastete Signale haben ein Spektrum, das periodisch mit f_S ist (-> Wiederholspektren). Der Frequenzbereich $-f_S/2 \dots f_S/2$ (oder, seltener, $0 \dots f_S$) wird Basisband genannt. Das Spektrum reellwertiger Signale ist symmetrisch zu $f = 0$, so dass hier bereits das positive Basisband $0 \dots f_S/2$ die vollständige Information enthält.
BP(-Filter)	Bandpass(-Filter)
BS	Bandsperre
CIC	Cascaded Integrator Comb (Filter)
DAC	Digital-to-Analog Converter
Dezimation	Verringerung der Abtastrate durch Wegwerfen von Samples (→ Downsampling) mit vorheriger Filterung zur Vermeidung von → Aliasing. In der Literatur werden beide Begriffe oft synonym verwendet.
Downsampling	Verringerung der Abtastrate, indem $R - 1$ von R Abtastwerten weggeworfen und nur jedes R -te Sample übernommen wird (→ Dezimation).
DSP	Digital Signal Processing oder Processor
DF	→ Direktform [Filter]
DFT	Diskrete Fourier Transformation
Direktform-Filter	Eine Filterstruktur, deren Koeffizienten <i>direkt</i> aus der Übertragungsfunktion $H(z)$ oder der Differenzengleichung abgeleitet werden können.
DGL	Differentialgleichung
DZGL	Differenzengleichung

Begriff	Erklärung
DTFT	Discrete-Time Fourier Transform; Fouriertransformierte von unendlich ausgedehnten, abgetasteten Zeitsignalen. Aufgrund der unendlichen Anzahl von Abtastpunkten hat die DTFT nur theoretische Bedeutung, auf Computern kann nur eine DFT oder FFT mit einer endlichen Punktzahl durchgeführt werden.
FFT	Fast Fourier Transform: Verschiedene Algorithmen zur effizienten Berechnung der → DFT. Die effizientesten Algorithmen basieren auf der Zerlegung der DFT in möglichst kleine Teilstücke und arbeiten daher am Besten wenn die FFT eine Länge von $N_{FFT} = 2^m$ (Radix-2) bzw. $N_{FFT} = 4^m$ (Radix-4) hat.
FIR-Filter	Finite Impulse Response-Filter; ein Filter mit endlicher Impulsantwort, d.h. in der Regel ein nicht-rekursives Filter (Ausnahme: CIC-Filter mit rekursiver Struktur aber endlicher Impulsantwort)
FPGA	Field Programmable Gate Array: Integrierte Bausteine mit programmierbaren Logikverknüpfungen. Im Gegensatz zu CPLDs (Complex Programmable Logic Devices) erlauben FPGAs die Realisierung wesentlich umfangreicherer und flexiblerer Systeme zur Datenverarbeitung.
HP(-Filter)	Hochpass(-Filter)
Interpolation	Allgemein Abschätzung von Werten zwischen Datenpunkten (z.B. Messwerte), wird hier verwendet für Ermittlung von zusätzlichen Abtastwerten bei Erhöhung der Abtastrate.
IIR-Filter	Infinite Impulse Response Filter; ein Filter mit unendlicher Impulsantwort. Das ist nur möglich mit einem rekursiven Filter.
Image	Wenn die Abtastrate durch Nullenstopfen von f_S auf If_S erhöht wird, liegen im neuen Basisband $-If_S/2 \dots If_S/2$ Kopien des alten Basisbands $-f_S/2 \dots f_S/2$, die sich im Gegensatz zu → Wiederholpektren unabhängig beeinflussen lassen.
Kritischer Pfad	In einem synchron getakteten (Teil-)System (= alle Register bekommen den gleichen Takt) müssen beim Eintreffen der nächsten Taktflanke die Daten an allen Registereingängen stabil anliegen. Der <i>kritische Pfad</i> mit der längsten Verzögerungszeit τ_{max} von einem Registerausgang zu einem Registereingang bestimmt daher die maximal zulässige Taktfrequenz des gesamten Systems, $f_{S,max} = 1/\tau_{max}$. Die Verzögerungszeit setzt sich zusammen aus allen arithmetischen Operationen (Additionen, Multiplikationen, ...) im jeweiligen Pfad. Achtung: Die Werte am Eingang $x[n]$ werden von einem Registerausgang außerhalb des betrachteten Teilsystems geliefert, ebenso werden die Daten am Ausgang $y[n]$ von einem weiteren Registereingang erwartet. Auch diese Pfade müssen daher berücksichtigt werden!

Begriff	Erklärung
Leistung	Bei nachrichtentechnischen Betrachtungen werden Leistungen oft in V^2 angegeben, was physikalisch natürlich nicht korrekt ist. Diese schlampige Nomenklatur kann man wie folgt rechtfertigen: Stellt man sich vor, dass das Signal z.B. an einem 50Ω Abschlusswiderstand abfällt, erhält man die zugehörige „physikalische Leistung“. Meist geht es bei nachrichtentechnischen Betrachtungen aber sowieso um Verstärkungs- und Signal-Rauschleistungsverhältnisse, dann fallen (gleiche) Impedanzen aus der Rechnung heraus.
LHP	Left Half-Plane; linke Halbebene der s -Ebene
LTI-System	Linear Time-Invariant System; Linear: System, bei dem eine Skalierung der Eingangsgrößen zur gleichen Skalierung der Ausgangsgrößen führt (Gegenbeispiele: Quadrierer, Logarithmierer, Begrenzer). Zeitinvariant: System, bei dem eine Verschiebung / Verzögerung der Eingangsgröße zu genau der gleichen Verschiebung des Ausgangssystem führt. Gegenbeispiel: Downampler.
MIMO	Multiple-Input, Multiple-Output [System]: System mit mehreren Eingangs- und / oder Ausgangsgrößen. Beispiele: Mehrgrößenregelung, WLAN-System mit mehreren Antennen.
Notch-Filter	Sehr schmalbandige Bandsperre (notch: engl. für Kerbe), um typischerweise eine bestimmte Störfrequenz (z.B. 50 Hz) zu unterdrücken.
Nutz-band(breite)	Der Frequenzbereich des Signals, in dem die Information enthalten ist.
Nyquist-Frequenz	Halbe Abtastfrequenz
Nyquist-Kriterium	Wenn die Bandbreite eines Signals kleiner ist als die halbe Abtastfrequenz, kann das Signal aus der abgetasteten Darstellung fehlerfrei rekonstruiert werden. Das Nyquist-Kriterium ist dann erfüllt, andernfalls tritt → Aliasing auf.
Nyquist-Wandler	→ ADC oder → DAC, dessen Nutzband im Gegensatz zu einem -> Oversampling-Wandler nur bis zur → Nyquist-Frequenz geht.
Nyquist-Zone	Ein Teil des → Wiederholspektrums von $\pm[kf_S/2 \dots (k+1)f_S/2]$
Rekursivfilter	Filter mit Rückkopplung (im Gegensatz zum -> Transversalfilter). Dadurch haben R. meist eine unendlich ausgedehnte Impulsantwort und werden auch → IIR-Filter genannt.
Out-of-Band Noise	Rauschen außerhalb des Nutzbands (aber innerhalb des Basisbands, das gibt es nur bei → Oversampling). Out-of-Band Noise kann herausgefiltert werden, ohne das Nutzband zu beeinflussen.
Oversampling	Überabtastung; ein Signal mit Bandbreite f_N , das mit $2f_S > f_N$ abgetastet wurde. Durch digitale Filterung kann das → Out-of-Band Noise entfernt und so das -> SNR des Signals verbessert werden.
Oversampling Wandler	-> ADC oder -> DAC, der mit einer (meist deutlich) höheren Abtastfrequenz arbeitet, als vom -> Nyquist-Kriterium gefordert.

Begriff	Erklärung
Quantisierung	Umwandlung eines wertkontinuierlichen in ein werdiskretes (= quantisiertes) Signal. Dieser Vorgang ist nichtlinear und mit einem Informationsverlust verbunden. Wird die Wortbreite eines quantisierten Signals reduziert, spricht man von -> Requantisierung.
Requantisierung	Reduktion der Wortbreite eines quantisierten Signals. Wie bei der -> Quantisierung ist dieser Vorgang nichtlinear und mit einem Informationsverlust verbunden.
RHP	Right Half-Plane; rechte Halbebene der s -Ebene
SFDR	Spurious-Free Dynamic Range; der Dynamikbereich zwischen stärkster Störlinie und Nutzsignal im Spektrum
SFG	→ Signalflussgraph oder Signal Flow Graph
SISO	Single-Input, Single-Output [System]
SNR	Signal-To-Noise Ratio; Verhältnis von Signalleistung und gesamter Störleistung
SOS	Second-Order Section; Teilfilter zweiter Ordnung. Durch die Aufteilung von IIR-Filters höherer Ordnung in Teilfilter zweiter Ordnung werden die Filter deutlich robuster gegen Rundungsfehler.
Signalflussgraph	(SFG); Graphische Darstellung des Informationsflusses in einem System. Bei digitalen Systemen (im Gegensatz zu analogen Schaltungen) entspricht der SFG genau dem Blockschaltbild.
TP(-Filter)	Tiefpass(-Filter)
transponieren	Transformation eines -> SFGs in einen anderen SFG mit gleicher Übertragungsfunktion durch Vertauschen von Ein- und Ausgang, Umdrehen sämtlicher Signalflussrichtungen und Vertauschen von Summations- und Abzweigungsknoten.
Transversalfilter	Filter ohne -> rekursiven Anteil. Die Impulsantwort eines T. hat somit eine endliche Länge, es wird daher auch -> FIR-Filter genannt.
WDF	Wave Digital oder Wellen-Digitalfilter
Wiederholspektren	Mit f_S abgetastete Signale haben Spektren, die periodisch mit f_S sind, d.h. das -> Basisbandspektrum wiederholt sich im Abstand von f_S . Die Wiederholungen lassen sich im Gegensatz zu -> Images nicht unabhängig von einander beeinflussen!
Zero Packing	Nullenstopfen; Auffüllen der fehlenden Zwischenwerte mit Nullen bei Erhöhung der Abtastrate
Zero Padding	Anhängen von Nullen an eine Datensequenz um z.B. auf 2^m Werte zu kommen und damit eine schnelle Radix-2 FFT zu ermöglichen. Außerdem erhält man mit einer längeren DFT auch eine feiner aufgelöste Darstellung (!) des Spektrums.

Begriff	Erklärung
ZOH	Zero-Order-Hold; Bei Abtastung wird der letzte Wert gehalten (zeitkontinuierliches Eingangssignal) bzw. wiederholt (zeitdiskretes Eingangssignal). Bei analogen Schaltungen auch als Sample & Hold bekannt.

Formelzeichen	Bedeutung	Einheit
A	Verstärkung; Dämpfung	1
A_{DB}	log. Verstärkung oder Welligkeit im Durchlassband	1
A_{SB}	minimale Dämpfung im Sperrband	1
B	Signalbandbreite	Hz
B_N	Rauschbandbreite	Hz
E{·}	Statistischer Erwartungswert	$\underline{\quad}^1$
F	Normierte Frequenz, $F = fT_S$	1
F_{DB}	Normierte Eckfrequenz des Durchlassbands	1
F_{SB}	Normierte Eckfrequenz des Sperrbands	1
I	Interpolationsfaktor, d.h. Faktor, um den die Abtastrate erhöht wird	1
L	Loop gain, Schleifenverstärkung	1
M	Anzahl (Koeffizienten, Taps); Ordnung	1
N	Anzahl (Nullstellen); Ordnung	1
N	Rauschleistung	W bzw. V ²
N_{FFT}	Anzahl der FFT bzw. DFT-Stützstellen	1
N_Q	Quantisierungsrauschleistung	W, V ² , etc. ¹
N'_Q	Quantisierungsrauschleistungsdichte	W/Hz etc. (s.o.)
P	(Nachrichtentechnische) Leistung	W bzw. V ²
R	Dezimationsfaktor; Faktor, um den die Abtastrate reduziert wird	1
S	Signalleistung	W bzw. V ²
T_S	Abtastperiode, $T_S = 1/f_S$	s
W	Wortbreite	(bits)
WF	Fractional Part der Wortbreite	s.o.
WI	Integer Part der Wortbreite	s.o.
c_Q	Codewert eines Quantisierers	1
di_N(x)	Dirichlet-Kernel (periodische si-Funktion); $di_N(x) := \sin(Nx/2)/N \sin(x/2)$	1
e_N	Rauschleistung	W bzw. V ²
f	Frequenz	Hz

Formelzeichen	Bedeutung	Einheit
f_S	Abtastfrequenz, $f_S = 1/T_S$	Hz
f_{DB}	Eckfrequenz des Durchlassbands	Hz
f_{SB}	Eckfrequenz des Sperrbands	Hz
q	LSB, Quantisierungsschritt	— ¹
s	Komplexe Frequenz; $s = \sigma + j\omega$	rad/s
$\text{si}(x)$	si- oder sinc-Funktion; $\text{si}(x) := \sin(x)/x$	1
x	Eingangsgröße, Signal	— ¹
x_Q	quantisierte Eingangsgröße; $x_Q = qc_Q$	— ¹
z	Komplexe Frequenz; $z = e^{sT_S}$	rad/s
z_0	Nullstelle	rad/s
z_∞	Polstelle	rad/s
Δ	Differenz; Determinante oder charakteristische Gleichung eines Graphen/Systems	—
$\epsilon[n]$	Quantisierungsfehler; $\epsilon[n] = x_Q[n] - x[n]$	— ¹
Ω	Normierte Kreisfrequenz; $\Omega = 2\pi f T_S$	1
ω	Kreisfrequenz; $\omega = 2\pi f$	s ⁻¹

¹Abhängig von der Dimension des Signals

Einleitung

i. Hinweise zu diesem Kurs

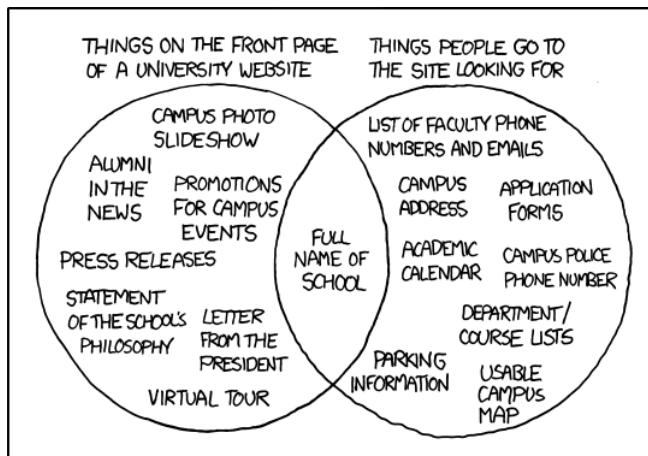


Abb. 0.2.: Was Sie in diesen Unterlagen finden [<https://xkcd.com/773/>]

Übungen: Im Teil I werden zunächst für jedes Kapitel ein paar grundlegende Begriffe und Formeln und Python / Matlab - Kommandos erläutert. Übungen, die mit einem Stern (*) gekennzeichnet sind, sind für das Verständnis absolut grundlegend. Übungen mit einem Plus (+) sind für Interessierte.

Die Musterlösungen in Teil II sind sehr ausführlich, vor allem die „gesternten“ Übungen dienen gleichzeitig als „Skriptersatz“.

Anhang: Hier (Teil III) finden Sie u.a. Tabellen zu Fourier-, Laplace und z-Transformation - diese Tabellen brauchen Sie nur für einzelne Aufgaben, aber nicht in der Prüfung.

Matlab / Python-Files finden Sie auf der Filehosting Plattform **GitHub** unter https://github.com/chipmuenk/dsp_fpga - Sie können alle Files per Zip-File herunterladen oder (empfohlen) mit **git** das Projekt auf Ihre lokale Festplatte „clonen“ (Kurzanleitung ebenfalls unter obigem Link). Dann können Sie bei Updates automatisch Ihre lokalen Files auf den neuesten Stand bringen lassen. Sie können sich auch ganz bequem anschauen, was sich geändert hat, mit den Files herumspielen und wenn's nicht mehr funktioniert, den Orginalzustand wieder herstellen. **git** wird auch immer mehr in der Softwareindustrie eingesetzt, das ist eine weitere wertvolle Fähigkeit für Ihren Lebenslauf ...

Das Gleiche gilt für Python und Matlab - nutzen Sie (nicht nur deshalb!) auf jeden Fall eins von beiden für kursbegleitende Simulationen: Sie bekommen ein besseres Verständnis für DSV, mehr Likes auf Facebook und ein glänzendes Fell.

Praktikum: Das Praktikum ist freiwillig, es vertieft prüfungsrelevante Kenntnisse der Vorlesung, bietet Ihnen die Gelegenheit den Stoff vor der Prüfung noch einmal durchzugehen und lässt

Sie mit Hilfe von Simulink / Xilinx System Generator selbst einen FPGA programmieren und in Betrieb nehmen.

Videos: Zu einigen Kapiteln habe ich Videos erstellt zu dem Stoff, den ich aus dem Bachelorstudium als bekannt voraussetze. Die Folien dazu haben Sie bekommen oder ich händige Sie vorher aus. Im Moodle - Kurs finden Sie die Links zu den Videos. Bitte schauen Sie sich die Videos selbstständig *vor* der Vorlesung an, vor allem wenn die digitale Signalverarbeitung bei Ihnen schon etwas in Vergessenheit geraten ist. Im Hörsaal gehe ich dann nur gemeinsam mit Ihnen Übungen und Fragen dazu durch („flipped classroom“) und mache ggf. eine Vorlesungseinheit, die auf diesen Grundlagen aufbaut.

Literatur: Zusätzliche Literatur benötigen Sie eigentlich nur, wenn Sie mehr zu bestimmten Themen wissen wollen oder wenn Ihnen meine Unterlagen zu chaotisch sind. In der Vorlesung weise ich ggf. auf zusätzliches Material hin.

Andere Kurse: Wenn Sie mit meinem Vorlesungsstil nicht gut zurecht kommen, versuchen Sie doch mal den edX-Kurs (MIT) ELEC301x „Discrete Time Signals and Systems“ (<https://www.edx.org/course/rice/rice-elec301x-discrete-time-signals-1032>) oder den Coursera Kurs „Digital Signal Processing“ (<https://www.coursera.org/course/dsp>)

... und jetzt viel Spaß mit dem Kurs:

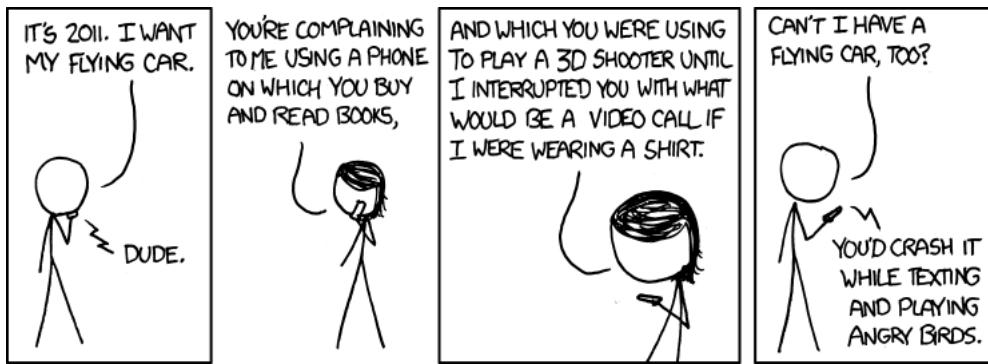


Abb. 0.3.: Darf's etwas mehr sein? [<http://xkcd.com/864/>]

ii. Allgemeine Hinweise zu Python und Matlab

Och nöö ...

... noch eine Skriptsprache? Muss das denn sein? Reichen denn nicht schon Matlab, Javascript, Perl, Ruby, ... ? Genau das war meine Reaktion als mir Anfang 2012 ein Werkstudent Python als Ersatz für Matlab™ vorschlug. Zu diesem Zeitpunkt hatte ich bereits ein gutes Jahr erfolglos nach Open Source Alternativen zu Matlab™ gesucht, um den Lizenzproblemen beim Einsatz in der Lehre und vor allem bei Kooperationen mit mittelständischen Firmen zu entkommen. In einem ersten Selbstversuch brauchte ich dann ca. 8 Stunden, um ohne Vorkenntnisse ein

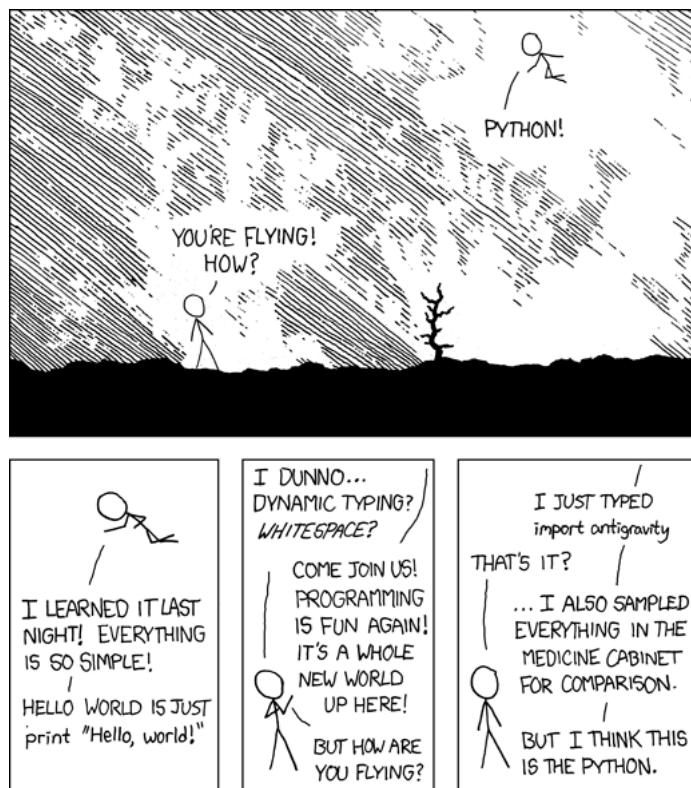


Abb. 0.4.: Python hebt ab! [<http://xkcd.com/353/>]

Matlab™-Skript zum digitalen Filterentwurf mit ca. 200 Zeilen Code erfolgreich nach Python zu „übersetzen“. Das überzeugte mich, zumal die Qualität der erzeugten Grafiken locker mit denen von Matlab™ mithalten konnte (Abb. 0.5).

Wer nutzt es?

Python ist weit verbreitet, laut TIOBE-Index (<http://tiobe.com>) war es die Programmiersprache der Jahre 2007 und 2010 mit der höchsten Zuwachsrate an Nutzern, sie steht 2012 auf Platz 8 der Beliebtheitsskala, 2015 auf Platz 5. Interessant ist es auch einen Blick in die Jobangebote der Python Gruppen von Xing und LinkedIn zu werfen. Laut <http://blog.codeeval.com/codeevalblog/2015> ist Python seit 4 Jahren die bei Arbeitgebern am meisten gefragte Programmiersprache.

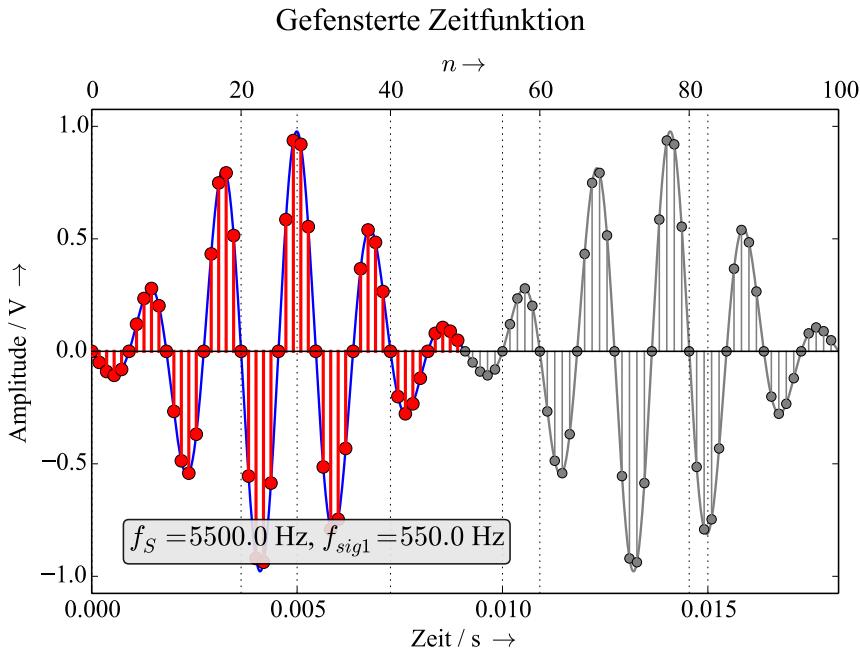


Abb. 0.5.: Beispiel für Plot, erstellt mit Python

Python ist auch die Sprache #1 für die Programmierausbildung an amerikanischen Universitäten laut <http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext> (2014). Ende 2015 überholte „learn python“ in Google Trends „learn java“ (<https://dzone.com/articles/learn-python-overtakes-learn-java>).

Python wurde auf <http://bit.ly/1jBjyWI> im Januar 2014 als „Beste Programmiersprache für Einsteiger“ gekrönt. Als Hauptgründe wurden genannt:

Lesbarkeit: Man wird es entweder lieben oder hassen, Code durch Einrückungen anstelle geschweifter Klammern zu strukturieren; es hilft aber vor allem Anfängern lesbaren Code zu schreiben. Jedenfalls besseren als z.B. mit Perl, dessen Code manchmal aussieht als ob er von einer fluchenden Comicfigur stammt ...

Universalität: Python ist im Gegensatz zu MatlabTM eine „richtige“ Programmiersprache, mit mächtigen Funktionen für Datenbank- und webbasierte Operationen, I/O - Funktionen, Software-Test, professionelle GUI-Erstellung etc. Es ist damit auch für größere Projekte geeignet. MatlabTM merkt man an einigen Stellen an, dass es über mehr als 20 Jahre gewachsen ist und nicht auf einer „richtigen“ Programmiersprache basiert - der Blog <http://abandonmatlab.wordpress.com> ist eine polemische aber unterhaltsame Auseinandersetzung mit den Schwächen von MatlabTM.

Batteries included: Sehr viele Funktionen werden bereits von den mitgelieferten Python Standard Libraries abgedeckt (unbedingt anschauen: <http://docs.python.org/2/library/>) oder lassen sich leicht mit einem zusätzlichen Modul nachrüsten.

Multi-Paradigm: Python ist eine objektorientierte Sprache, die Objektorientiertheit aber nicht erzwingt. Prozedurale oder funktionale Programmierung sind genauso möglich.

Cross-Plattform: Python und GUI-Bibliotheken wurden auf verschiedenste Plattformen portiert. Neben den „großen drei“ (Windows, Linux, OS X) werden u.a. auch Android und iOS (Kivy) und Raspberry Pi unterstützt.

Support: Bei aktiven Open Source Projekten gibt es meist sehr gute und rasche Unterstützung durch die Community, mindestens genauso gut und eher schneller als bei kommerziellen Produkten.

Web-Based Computing: Mit IPython kann man interaktive Notebooks erstellen, vergleichbar mit den Worksheets von MathCAD, Mathematica oder Maple. Diese Notebooks können als Files weitergegeben oder auf einem Server bereitgestellt und dann im Browser bearbeitet werden (Webbased Computing). Für die Lehre ergeben sich damit völlig neue Möglichkeiten.

Für diesen Kurs sollten Sie unbedingt Python oder Matlab installieren, um selbst mit den Übungen „herumzuspielen“. Unter Windows empfehle ich **Winpython** (<https://winpython.github.io/>), unter OS X oder Linux **Anaconda** (<https://store.continuum.io/cshop/anaconda/>).

Um die Codeschnipsel in den Übungen kurz zu halten, werden immer die Header Listing 1 und 2 für Python bzw. Matlab-Code verwendet (und nicht mitabgedruckt):

```

1 #!/usr/bin/env python
2 # -*- coding: iso-8859-15 -*-
3 #=====
4 # _common_imports_v3_py.py
5 #
6 # Einfaches Code-Beispiel zum Kapitel "xxx", Übungsaufgabe yyy
7 #
8 # Importiere Module zur
9 # - Erzeugung von Zufallszahlen:      numpy.rnd
10 # - (Inverse) FFT:                  numpy.fft
11 # - Signalverarbeitung            scipy.signal
12 # - Interpolation                 scipy.interp
13 # (c) 2014-Feb-04 Christian Münker - Files zur Vorlesung "DSV auf FPGAs"
14 #=====
15 from __future__ import division, print_function, unicode_literals # v3line15
16
17 import numpy as np
18 import numpy.random as rnd
19 from numpy import (pi, log10, exp, sqrt, sin, cos, tan, angle, arange,
20                    linspace, array, zeros, ones)
21 from numpy.fft import fft, ifft, fftshift, ifftshift, fftfreq
22 import scipy.signal as sig
23 import scipy.interpolate as intp
24
25 import matplotlib.pyplot as plt
26 from matplotlib.pyplot import (figure, plot, stem, grid, xlabel, ylabel,
27                                 subplot, title, clf, xlim, ylim)
28
29 import dsp_fpga_lib as dsp
30 #----- v3line30
31 # Ende der gemeinsamen Import-Anweisungen

```

Lst. 1: Gemeinsamer Python-Header für diesen Kurs

```

1 %=====
2 %
3 % Common Matlab header for "DSV auf FPGA" Code
4 %
5 %

```

```

6 % (c) 2013 Christian Münker - Files zur Vorlesung "DSV auf FPGAs"
7 %=====
8 set(0,'DefaultAxesColorOrder',[0.8 0 0.2; 0 1 0; 0 0 1], ...
9     'DefaultAxesLineStyleOrder','--|:-|-.');
10
11 set(0,'DefaultAxesUnits','normalized');
12 set(0,'DefaultAxesFontSize',16);
13 set(0,'defaultTextFontSize',16);
14 set(0,'defaultLineMarkerSize', 6);
15
16 set(0,'defaultaxeslinewidth',2);
17 set(0,'defaultlinelinewidth',2);
18 close all; % alle Plot-Fenster schließen
19 clear all; % alle Variablen aus Workspace löschen
20 %

```

Lst. 2: Defaultsettings für hübschere Matlab-Plots

Namespaces: In Matlab stehen die Kommandos aller installierten Toolboxen zur Verfügung, in Python werden die gewünschten Module mit eigenen *Namespaces* importiert. So werden nicht gleichnamige Funktionen unabsichtlich überschrieben / überladen und es bleibt transparent, woher die einzelnen Funktionen stammen. Je nach Anforderung gibt es verschiedene Varianten:

```

1 import numpy      # importiere NumPy Modul mit eigenem Namespace
2 x = numpy.pi      #
3
4 import numpy as np          # importiere Modul bzw.
5 import numpy.random as rnd   # Untermodul mit abgekürztem Namespace
6 x = np.pi * rnd.normal()    # EMPFOHLENE VARIANTE !!
7
8 from numpy import *        # Import aller Funktionen eines Moduls in
9                  # gemeinsamen Namespace (nur für interaktives
10                 # Arbeiten empfohlen: "'besudelt'" den Namespace)
11
12 from numpy import pi,log10  # Kompromiss: Import oft benutzter Funk-
13 x = pi * log10(1000)       # tionen in gemeinsamen Namespace

```

Python-Listings sind ohne die Encoding- und Import-Statements in Listing 1 abgedruckt, um Platz zu sparen. Die Files zum Download im Moodle Kurs sind aber natürlich vollständig. Zum Einstieg mit Python können Sie sich den Kurs unter <http://scipy-lectures.github.io/> anschauen, um innerhalb kürzester Zeit durchzustarten!

Matlab-Listings wurden ohne die Voreinstellungen für Fontgrößen in Listing 2 etc. abgedruckt, ebenso ohne `clear all;` und `close all;` Anweisungen.

Wichtige Einstellungen für die Python Entwicklungsumgebung Spyder:

Tools -> Python Path Manager: Binden Sie den Pfad der Library `dsp_fpga_lib` ein!

Unter Tools -> Preferences:

- Passen Sie das **Global Working Directory** an
- Stellen Sie unter Run ein „Execute in a new dedicated Python console“ und „Always show Run Settings dialog on a first file run“. Diese Einstellungen können über Run -> Configure jederzeit umgestellt werden.

- Unter **Editor -> Advanced Settings** können Sie das Template für neu erstellte Python Files anpassen
- Wählen Sie unter **IPython Console -> Graphics** die Einstellung **Backend: Automatic**, damit Plots in eigenen Fenster geöffnet werden (und nicht als Mini-Bildchen in der Console).

Teil I.

Übungsaufgaben

1. LTI: LTI-Systeme im Zeitbereich

Inhalt

Dieses Kapitel beschäftigt sich mit zeitdiskreten, linearen zeit-invarianten (*Linear Time-Invariant, LTI*) Systemen im *Zeitbereich* und deren Darstellung als Signalflussgraph (SFG). Nach dem Durcharbeiten dieses Kapitel sollten Sie die folgenden Eigenschaften dieser Systeme verstanden haben und in einander überführen können:

Impulsantwort aus SFG und Systemfunktion

Differenzengleichung aus SFG und Systemfunktion

Kritischer Pfad und Ressourcenverbrauch aus SFG

Filterstrukturen (Direktformfilter Typ 1 und 2 und transponierte Filter) aus Systemfunktion und umgekehrt

Moodle:

Bitte schauen Sie sich das Video und die zugehörigen Folien zu Kapitel 1 vor der Vorlesungs/Übungsstunde an.

Bei größeren Lücken empfehle ich zur Vorbereitung das Skript von Prof. Geng (-> Moodle) oder ein anderes einführendes Buch zur zeitdiskreten Signalverarbeitung.

Theorie

Signalflussgraphen (SFG)

Die Analyse komplizierter *linearer*¹ Systeme wird durch *Signalflussgraphen* (SFG) vereinfacht, die die Topologie des unabhängigen linearen Gleichungssystems visualisieren. Die Variablen des Gleichungssystems werden durch die Knoten des Graphen repräsentiert, die verbindenden Zweige (auch Kanten genannt) definieren deren Zusammenhang (Abb. 1.1). Die Visualisierung erleichtert die Manipulation und Auflösung des Gleichungssystems. SFGs wurden in den 1950ern zur Analyse linearer Netzwerke eingeführt. Seitdem werden SFGs zur Lösung der unterschiedlichsten ingenieurwissenschaftlichen Probleme erfolgreich eingesetzt, wie z.B. zur Errechnung der Ausfallwahrscheinlichkeiten im NASA Apollo Projekt oder für die symbolische Analyse von Analogschaltungen. In diesem Kurs werden SFGs für die Analyse von zeitdiskreten Systemen in der z -Ebene verwendet.

Ein SFG ist ein **gerichteter Graph**, d.h. eine Menge von **Knoten** und **gerichteten Zweigen**. Jeder Zweig beginnt und endet auf einem Knoten x_i (das kann auch der gleiche sein) und hat

¹Ohne diese wichtige Voraussetzung sind die im Folgenden geschilderten Methoden zur Umformung von Netzwerken nicht gültig!

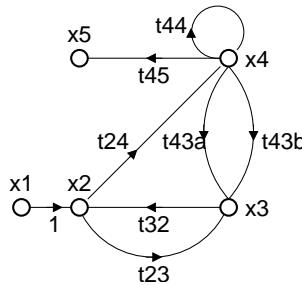
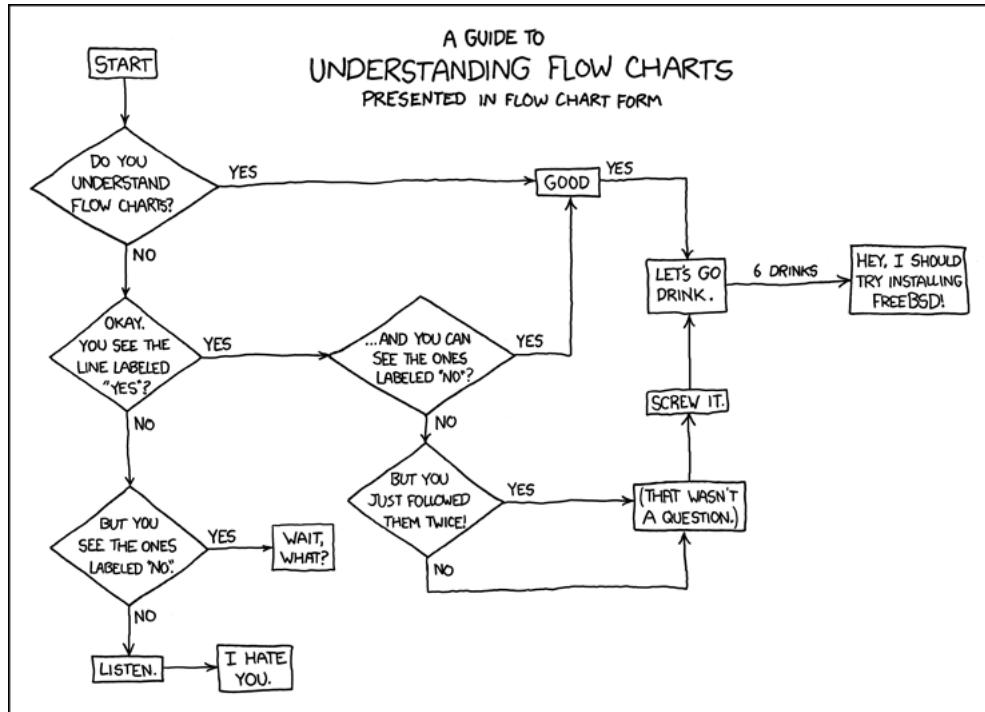


Abb. 1.1.: Beispiel für einen Signalflussgraphen

einen Verstärkungsfaktor (Gewicht), die Zweigverstärkung oder **Zweig-Transmittanz** t_{ij} . Eine **Schleife** ist eine Folge von Zweigen bei der jeder Knoten zu genau zwei Zweigen der Schleife gehört, von denen einer auf dem Knoten beginnt und einer endet. In einer Schleife erster Ordnung kann jeder Knoten von jedem anderen Knoten erreicht werden. Ähnlich einer Schleife erster Ordnung ist ein **Vorwärtspfad** eine Abfolge von Zweigen zwischen einem Start- und einem Endknoten. Dabei ist jeder Knoten mit genau zwei Zweigen des Pfads verbunden, außer dem Startknoten, der keinen Eingangszweig hat und dem Endknoten, der keinen Ausgangszweig hat. Eine Schleife der Ordnung m ist eine Menge von m Schleifen erster Ordnung, die sich nicht berühren, d.h. Schleifen ohne gemeinsame Knoten. Die **Schleifenverstärkung** L einer Schleife erster Ordnung bzw. die **Pfadverstärkung** P eines Pfades ist definiert als das Produkt aller Zweigverstärkungen entlang der Schleife bzw. des Pfads. Die Schleifenverstärkung einer Schleife höherer Ordnung ist das Produkt der Schleifenverstärkungen aller Schleifen erster Ordnung.

Abb. 1.2.: Flowchart als Flowchart [<http://xkcd.com/518/>]

Drei einfache Regeln definieren das Verhalten von SFGs:

Richtung: Ein Signal fließt entlang eines Zweigs in die Richtung, die durch den Pfeil angezeigt wird und wird mit der Transmittanz des Zweigs multipliziert, z.B. $x_5 = t_{45}x_4$ in Abb. 1.1.

Senke: Ein Knotensignal ist die Summe aller am Knoten ankommenden Signale, z.B. $x_3 = t_{23}x_2 + t_{43a}x_4 + t_{43b}x_4$ in Abb. 1.1.

Quelle: Das Knotensignal wird an jeden abgehenden Zweig angelegt.

Anmerkung: Verwechseln Sie nicht ein SFG mit einem *Zustandsdiagramm* oder *Automatengraphen* - auch wenn die Form und z.T. die Bezeichnungen die gleichen sind: Bei letzteren entsprechen die Knoten Zuständen und die Zweige Booleschen Bedingungen für den Übergang von einem Zustand in einen anderen!

Transformationen von SFGs

Von diesen drei Regeln können einige nützliche Regeln zur Umformung von SFGs abgeleitet werden (Abb. 1.3(a) - 1.5(b)). Mit deren Hilfe können Zweige eines SFGs zusammengefasst und Knoten eines SFGs der Reihe nach eliminiert werden, um zum Schluss die Übertragungsfunktion zwischen zwei Knoten zu erhalten.

Zusammenfassen von seriellen und parallelen Zweigen

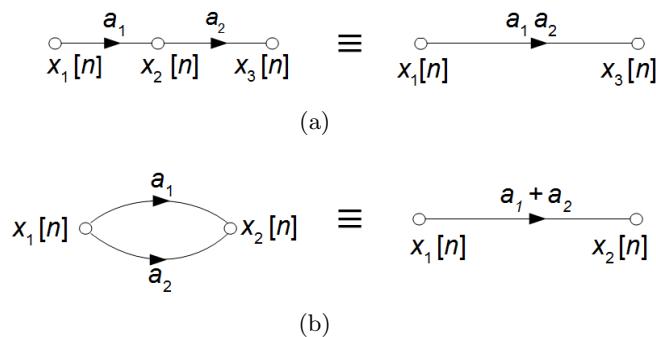


Abb. 1.3.: Zusammenfassen von seriellen (a) und parallelen (b) Zweigen

Diese Regeln erhält man über $x_3 = a_2x_2 = a_2(a_1x_1) = a_1a_2x_1$ und darüber, dass die Signale aller einlaufenden Zweige im Knoten addiert werden.

Distributivität („Schieben über Knoten“)

Die Regeln zur Distributivität (= Reihenfolge von Multiplikation und Addition darf vertauscht werden) in Abb. 1.4 ergeben sich aus der Linearität des Systems (die ja vorausgesetzt worden war). Diese Regeln sind sehr praktisch, um einen Verstärkungsfaktor über einen Knoten zu „verschieben“. Dabei darf ein Register als Multiplikation mit z^{-1} betrachtet werden!

Transponieren

Man *transponiert* einen Graph, indem man nacheinander die folgenden Operation durchführt:

- Umdrehen sämtlicher Signalrichtungen.
- Ersetzen von Verzweigungen durch Addierer.

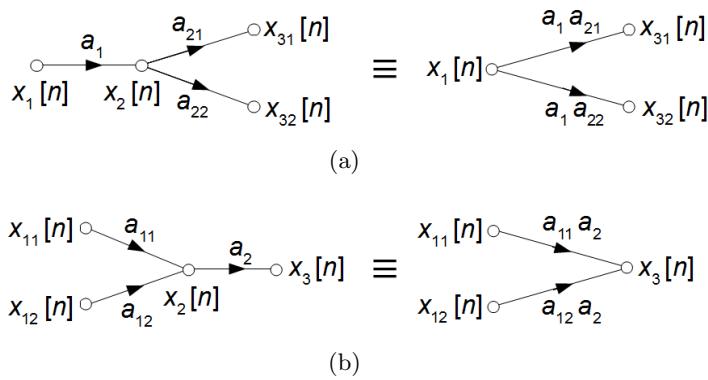


Abb. 1.4.: Distributivität

- Ersetzen von Addierern durch Verzweigungen. Bei Subtrahierern muss das negative Vorzeichen ersetzt werden, entweder durch Umkehr des Vorzeichens eines Koeffizientenmultipliziers im Pfad oder durch Umwandeln eines Addierers am Ende des Pfads in einen Subtrahierer (jeweils falls vorhanden).
- Vertauschen von Eingangs- und Ausgangsknoten.

Schleifen

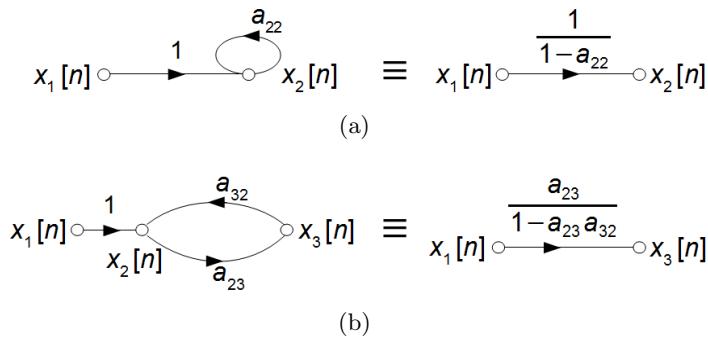


Abb. 1.5.: Eliminieren von Schleifen, (a) Eigenschleifen und (b) allgemeine Schleifen

Die Regeln in Abb. 1.5 zum Eliminieren von Schleifen ergeben sich durch Aufstellen und Auflösen der Knotengleichungen (bekannt aus der Regelungstechnik).

Direktform-Systeme

Abb. 1.8 zeigt Systeme zweiter Ordnung in Direktform 1 (DF1) und 2 (DF2) - Struktur mit der gleichen Systemfunktion

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}.$$

Die Koeffizienten für eine Realisierung in DF1- oder DF2 Struktur können direkt² aus einer gegebenen Übertragungsfunktion abgelesen werden. Versuchen Sie zu Übungszwecken aus den SFGs die Systemfunktion abzuleiten.

²Daher der Name ...

Kritischer Pfad

In einem synchron getakteten (Teil-)System (= alle Register bekommen den gleichen Takt) müssen beim Eintreffen der nächsten Taktflanke die Daten an allen Registereingängen stabil anliegen. Der *kritische Pfad* mit der längsten Verzögerungszeit τ_{max} von einem Registerausgang zu einem Registereingang bestimmt daher die maximal zulässige Taktfrequenz des gesamten Systems, $f_{S,max} = 1/\tau_{max}$. Die Verzögerungszeit setzt sich zusammen aus allen arithmetischen Operationen (Additionen, Multiplikationen, ...) im jeweiligen Pfad. Beachten Sie dabei:

- Werte am Eingang $x[n]$ werden von einem Registerausgang außerhalb des betrachteten Teilsystems geliefert, ebenso werden die Daten am Ausgang $y[n]$ von einem weiteren Registereingang des nachfolgenden Funktionsblocks erwartet (siehe z.B. Abb. M1.6). Auch diese Pfade müssen u.U. bei der Bestimmung des kritischen Pfads berücksichtigt werden!
- Ein Addierer mit drei Eingängen besteht eigentlich aus zwei Addierern mit je zwei Eingängen, die in unterschiedlicher Reihenfolge kombiniert werden können: $(a+b)+c = a+(b+c)$ - daraus können sich abhängig von der Implementierung unterschiedliche kritische Pfade ergeben.

Die Länge des kritischen Pfads und damit die maximale Taktrate können sich verschiedene Implementierungen der gleichen Übertragungsfunktion deutlich unterscheiden. Das gleiche gilt auch für den Verbrauch an Hardware-Ressourcen (Register, Addierer, Multiplizierer).

Python: Befehle und Programme zu Kap. 1

Funktion	Python	Matlab	Beschreibung
Faltung	<code>y = np.convolve(x,h)</code>	<code>y = conv(x,h)</code>	Falte Vektoren x und h miteinander.
Nullstellen	<code>r = np.roots(p)</code>	<code>r = roots(p)</code>	Finde die Wurzeln r eines Polynoms mit den Koeffizienten p .
Polynom	<code>p = np.poly(r)</code> <code>y1 = np.polyval(p, x1)</code>	<code>p = poly(r)</code> <code>y1 = polyval(p, x1)</code>	Finde die Koeffizienten p eines Polynoms mit den Wurzeln r . Bestimme den Wert $y1$ des Polynoms mit den Koeffizienten p an der Stelle $x1$
Grafik			
Fenster	<code>fig = plt.figure(1)</code>	<code>figure(1)</code>	Öffne ein Fenster
Subfigure	<code>ax = fig.subfigure(211)</code>	<code>subfigure(211)</code>	Erstelle Subfigures
Linienplot	<code>ax.plot(x,y)</code>	<code>plot(x,y)</code>	Zeichne Linienplot
Stemplot	<code>ax.stem(x,y)</code>	<code>stem(x,y)</code>	Zeichne Impulsplot

Tab. 1.1.: Spezielle Befehle zu Kap. 1

Anmerkungen zu Tab. 1.1:

Allgemein: In Python finden sich die meisten Funktionen zur Signalverarbeitung im Numpy Paket (abgekürzt mit `np`) oder im Signal Processing Modul von Scipy (`scipy.signal`), in Matlab in der Signal Processing Toolbox.

convolve Optional: mode=“same“ oder „valid“ um Randeffekte zu unterdrücken.

Filename	Beschreibung
<code>LTI_faltung</code>	Einfaches Code-Beispiel zur Demonstration von Faltung
<code>LTI_tricks</code>	Tricks zu Grafik und Plotting: Subplots, Definition und Simulation von IIR-Filtern, Interpolation ...
<code>LTI_sensor_py</code>	Python-Musterlösung zur Übungsaufgabe Aufgabe 1.1
<code>LTI_periodizitaet</code>	Periodizität von abgetasteten Signalen
Directory „Basics“	
<code>fibonacci</code>	Rekursive Berechnung der Fibnonacci-Folge
<code>running_sine</code>	“Laufender Sinus“ mit Messung der Frames pro Sekunde
<code>sum_series</code>	Berechne endliche geometrische Reihe, drucke Ergebnisse mit Formatierung
<code>Grundsignale</code>	Beispiele für Darstellung verschiedener 2D- und 3D-Funktionen
Directory „Demos“	
<code>matplotlib_Tex</code>	Plot mit TeX-Formatierung der Labels
<code>interpolate_noise</code>	Demonstration einiger Mathe- und Plotfähigkeiten von Python
Directory ”3D-Plots“	
<code>LTF_3D_plots</code>	3D-Plots
<code>matplotlib-3D-plots</code>	Surfaceplot mit Colorbar einer einfachen 3D-Funktion
<code>charges_fieldlines</code>	3D-Visualisierung des elektrischen Feldes einer oder mehrerer Punktladungen
<code>magnetic_field-lines</code>	Feldlinien eines magnetischen Dipols (Stromschleife) [Mayavi]
<code>B_field_visualize</code>	Zeige das B-Feld eines Helmholtz-Spulenpaars mit Schnittebene und Iso-Surface [Mayavi]
<code>mlab_inter-active_example</code>	3D-Kurve, die über Slider parametrisiert werden kann [Mayavi, Traits]

Tab. 1.2.: Simulationsfiles zu Kap. 1

Der wichtigste Python- bzw. Matlab-Befehl für dieses Kapitel ist die **Faltung**, `np.convolve(x, h)` bzw. `conv(x, h)` in Listing 1.1 bzw. 1.2, Zeile 4. Die beiden Mini-Programmchen zeigen außerdem, wie man Signale definiert und plottet.

```

1 # ... Ende der gem. import-Anweisungen
2 h = [0.25, 0.5, 0.25] # Impulsantwort
3 x = [1, 1, 1, 1, 1] # Eingangssignal
4 y = np.convolve(x, h)
5 n = arange(len(y)) # n = 0 ... len(y)-1
6 figure(1)
7 stem(n, y, 'r'); grid(True)
8 xlabel(r'$n \rightarrow$')
9 ylabel(r'$y[n] \rightarrow$')
10 title(r'Faltung $y[n] = x[n] \star \{1; 1; 1; 1; 1\}$')
11 plt.show()

```

Lst. 1.1: Faltung mit Python
(LTI/LTI_faltung_py.py)

```

%
h = [0.25 0.5 0.25];
x = [1, 1, 1, 1, 1];
y = conv(x, h); % Faltung
n = 0:length(y)-1;
figure(1);
stem(n, y); grid on;
xlabel('n ->');
ylabel('y[n] ->');
title('Faltung');
%
```

Lst. 1.2: Faltung mit Matlab
(LTI/LTI_faltung_m.m)

Ein paar weitere nützliche Tricks und Anweisungen zeigen Listing 1.3 und 1.4, u.a. wie man Subplots erzeugt, im Zeitbereich filtert und die Impulsantwort auch von rekursiven Systemen bestimmt.

Die Interpolation von Signalen ist in Python anders implementiert als in Matlab: In Python definiert man zunächst eine Funktion auf Basis der vorhandenen Stützpunkte und des gewählten Interpolationsverfahrens und berechnet damit die Funktionswerte für die neuen unabhängigen Werte):

```

1 # ... Ende der Import-Anweisungen
2 # -- Impulse response (lin / log) --
3 f1 = 50; Ts = 5e-3
4 n = arange(0, 50) # sample n
5 t = arange(0, 49., 0.1) # feiner aufgelöst
6 xn = 1.5 + 0.5*cos(2.0*pi*f1*n*Ts) # x[n]
7 b = [0.1, 0]; a = [1, -0.9] # Koeffizienten
8 # H(z) = (0.1 z + 0) / (z - 0.9)
9 [h, k] = dsp.impz(b, a, N = 30) # -> h[k]
10 figure(1)
11 subplot(211)
12 stem(k, h, 'r') # x[n], red stems
13 ylabel(r'$h[k] \rightarrow$'); grid(True)
14 title(r'Impulsantwort $h[n]$')
15 subplot(212)
16 stem(k, 20*log10(abs(h)), 'r')
17 xlabel(r'$k \rightarrow$'); grid(True)
18 ylabel(r'$20 \log |h[k]| \rightarrow$')
19 # ----- Filtered signal -----
20 figure(2);

```

```

% -- Impulse response (lin / log) --
f1 = 50; Ts = 5e-3;
n = [0:49]; % sample n
t = 0:0.1:49; % start/step/stop
xn = 1.5 + 0.5*cos(2.0*pi*f1*n*Ts);
b = [0.1, 0]; a = [1, -0.9];
%
[h, k] = impz(b, a, 30);
figure(1);
subplot(211);
stem(k, h, 'r-');
ylabel('h[k] \rightarrow'); grid on;
title('Impulse Response h[n]');
subplot(212);
stem(k, 20*log10(abs(h)), 'r-');
xlabel('k \rightarrow'); grid on;
ylabel('20 log |h[k]| \rightarrow');
% ----- Filtered signal ---
figure(2);

```

```

21 yn = sig.lfilter(b,a,xn) #filter xn with h
22 f = intp.interp1d(n, yn, kind = 'cubic')
23 yt = f(t) # y(t), interpolated
24 plot(t, yt, color='#cc0000', linewidth=3)
25 stem(n, yn, 'b') # y[n]
26 xlabel(r'$n \rightarrow$'); grid(True)
27 ylabel(r'$y[n] \rightarrow$')
28 title('Filtered Signal')
29 plt.show()      # draw and show the plots

```

Lst. 1.3: Python Tricks zu Kap. 1
(LTI/LTI_tricks_py.py)

```

yn = filter(b,a,xn);
yt = interp1(n, yn, t, 'cubic');
hold on; % don't overwrite plots
plot(t, yt,'color',[0.8,0,0],'LineWidth',3);
stem([0:length(yn)-1],yn,'b');
xlabel('n \rightarrow'); grid on;
ylabel('y[n] \rightarrow');
title('Filtered Signal');
%
```

Lst. 1.4: Matlab Tricks zu Kap. 1
(LTI/LTI_tricks_m.m)

1.1. * Filterung abgetasteter Signale → M1.1

Das Signal eines Temperatursensors ändert sich so langsam, dass es für die Dauer der Messung als Gleichsignal betrachtet werden darf, $x_{Temp}(t) = 1,5$ V. Das Sensorsignal ist mit einer Brummstörung überlagert, $x_n(t) = 0,5 \text{ V} \cos(2\pi \cdot 50 \text{ Hz} \cdot t)$. Das Signal $x(t) = x_{Temp}(t) + x_n(t)$ wird durch Abtastung mit $f_{S1} = 25 \text{ Hz}$, $f_{S2} = 200 \text{ Hz}$, $f_{S3} = 240 \text{ Hz}$ und $f_{S4} = 50 \cdot \pi \text{ Hz}$ in zeitdiskrete Folgen $x_i[k]$ gewandelt. In einem nachgeschalteten digitalen Filter soll die Brummstörung möglichst gut unterdrückt werden.

f_s	n	0	1	2	3	4	5	6	7
25 Hz	$x_1[n]$								
200 Hz	$x_2[n]$								
240 Hz	$x_3[n]$								
50π Hz	$x_4[n]$								
	$y_{MA,4}[n]$								
200 Hz	$y_{MA,3}[n]$								
	$y_{casc}[n]$								
	$y_{Int}[n]$								

Tab. 1.3.: Ergebnisse zu Aufgabe 1.1

a) Abgetastete Signale

Geben Sie für die Abtastraten $f_{S1} \dots f_{S4}$ das zeitdiskrete Signal $x_i[k] = x_{Temp,i}[k] + x_{n,i}[k]$ an und tragen Sie die ersten 8 Werte in Tab. 1.3 ein! Ist $x_i[k]$ periodisch? Wenn ja, nach wievielen Abtastwerten N wiederholt sich $x_i[k]$? Wieviele Perioden L von $x_n(t)$ sind darin enthalten? Skizzieren Sie die abgetasteten Signale, erklären Sie das Ergebnis für $f_{S1} = 25$ Hz!

b) Filterung:

Bestimmen Sie jeweils die folgenden Eigenschaften der beiden Moving Average Systeme und des verlustbehafteten Integrators ($\alpha = 0,5$) in Abb. 1.6:

Impulsantwort: Skizzieren Sie die Impulsantwort $h[k]$ oder schreiben Sie sie auf!

Skalierung: Wie muss die Amplitude des Ausgangssignals skaliert werden, damit der Gleichanteil des Eingangssignals korrekt wiedergegeben wird?

Einschwingen: Ab welchem Sample ist das Ausgangssignal eingeschwungen?

Ausgangssignal: Berechnen Sie für die Eingangsfolge $x_2[k]$ ($f_{S2} = 200$ Hz) das Ausgangssignal $y[k]$ für $k = 0 \dots 7$ und tragen Sie die Werte in Tab. 1.3 ein.

Brummunterdrückung: Versuchen Sie aus den diskreten Werten $y[k]$ eine geschlossene Beschreibung für den eingeschwungenen Zustand zu finden und daraus die Brummunterdrückung abzuschätzen.

Ordnung (nur System Abb. 1.6(a)): Die Ordnung des Filters wird jetzt auf $N = 3$ reduziert. Passen Sie die Skalierung an, bestimmen Sie das Ausgangssignal $y_{b,3}[k]$ und tragen Sie die Werte für $k = 0 \dots 7$ in Tab. 1.3 ein. Erklären Sie das Ergebnis!

Kaskadiertes MA Filter: Durch welches nicht-kaskadierte Filter man das System in Abb. 1.6(b) ersetzen? Welchen Nachteil hat die nicht-kaskadierte Form?

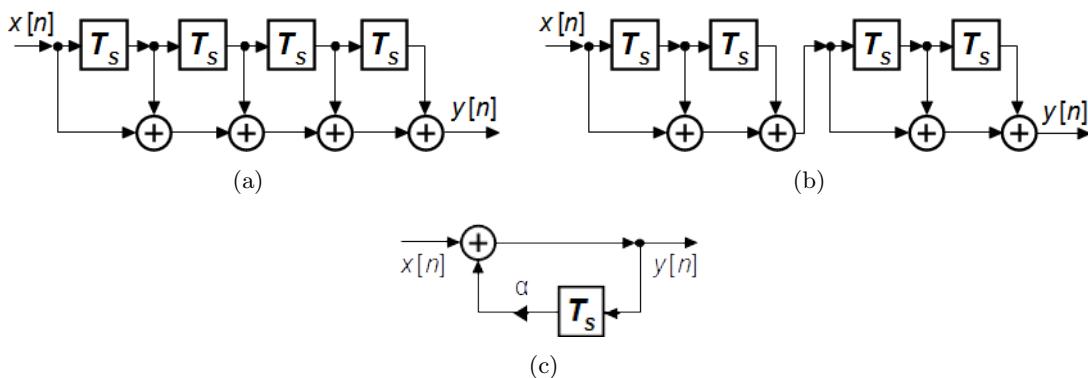


Abb. 1.6.: Moving Average (MA) (a), kaskadiertes MA - Filter (b) und verlustbehafteter (gedämpfter) Integrator (c)

- c) **Simulieren** Sie abgetastete Eingangssignale, Impulsantwort und die Ausgangssignale der Systeme mit Matlab / Python.
- d) Welchen **kritischen Pfad** haben die Filterimplementierungen und welche maximale Taktfrequenzen resultieren daraus? Eine Addition benötigt 2 ns, eine Multiplikation 10 ns.

- e) Wieviele **Taktzyklen pro Sample** benötigt ein uC, auf dem die Filter jeweils implementiert werden sollen? Pro Taktzyklus kann der uC nur eine Rechenoperation (Add, Multiply, ...) durchführen. Operationen zur Speicherverwaltung (Schieberegister, Ringbuffer etc.) sollen vernachlässigt werden.
- f) Wie könnte eine **effiziente Implementierung des Moving Average Filters** in Software oder Hardware aussehen? Schauen Sie sich dazu an, wie sich das Ausgangssignal mit jedem Sample ändert.
- g) Wie könnte ein System zur **Unterdrückung des Gleichanteils** (z.B. Offset) von Wechselsignalen aussehen?

1.2. * Faltung mit einfachem FIR-Filter → M1.2

- a) Ermitteln Sie aus der Struktur des FIR-Filters in Abb. 1.7 die Impulsantwort $h[n]$ und skizzieren Sie diese.

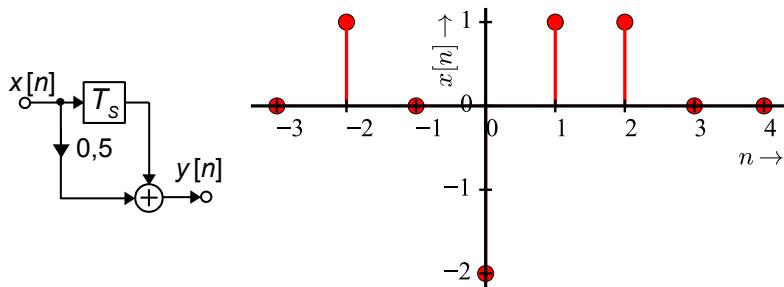


Abb. 1.7.: FIR-Filter (a) mit Eingangssignal $x[n]$ (b) zu Aufgabe 1.2 und 2.2

- b) Geben Sie die Differenzengleichung für das Filter an.
- c) Ermitteln Sie Ausgangssignal $y[n]$ für das Eingangssignal $x[n] = \{1; 0; -2; 1; 1\}$ mit $n = -2, \dots, 2$ (Abb. 1.7b) durch diskrete Faltung $x[n] * h[n]$ oder $h[n] * x[n]$. Skizzieren Sie das Ausgangssignal $y[n]$.

Welche Reihenfolge ist für die Berechnung der Faltung ist bei FIR-Filttern besser geeignet?

1.3. * Systemeigenschaften aus Impulsantwort → M1.3

Die folgenden Systeme werden durch ihre Impulsantwort $h[n]$ mit $n = 0, 1, \dots$ beschrieben.

- a) $h_A[n] = \{0,25; 0,5; 0,25\}$
- b) $h_B[n] = \{0,25; -0,5; 0,25\}$
- c) $h_C[n] = \{0,25; 0,5; 0,75\}$

Sind die Systeme kausal und stabil? Erwarten Sie eher Hochpass- oder Tiefpassverhalten aufgrund der Impulsantwort? Zeichnen Sie eine Hardware-Implementierung auf. Gibt es mehr als eine Hardware-Realisierung?

1.4. * Umformung von SFGs mit graphischen Methoden → M1.4

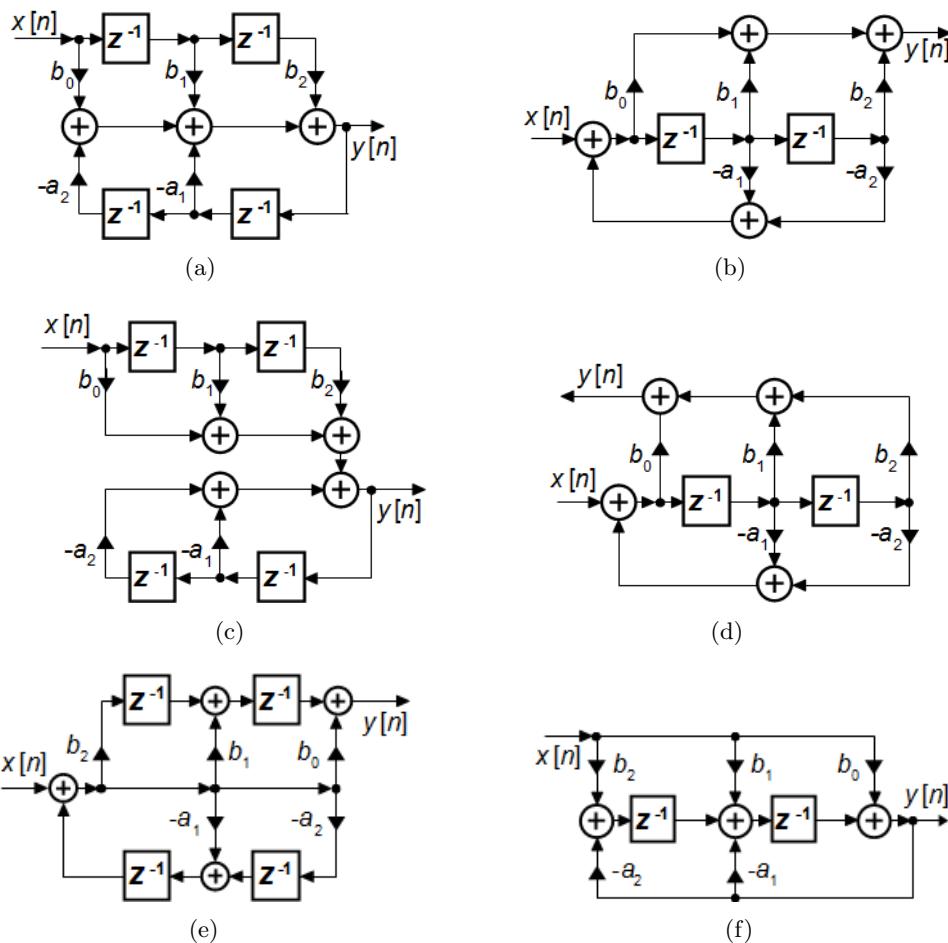


Abb. 1.8.: IIR-Systeme zweiter Ordnung in Direktform 1 (a), (c) und (e, transponiert) und Direktform 2 (b), (d) und (f, transponiert) zu Aufgabe 1.4, 1.5 und 1.6

ML fehlt noch!!

- Zeigen Sie, dass man die SFGs der Systeme in Abb. 1.8 mit folgenden Operationen in einander überführen kann:
 - „Verschieben über Knoten“ (Distributivität)
 - Transponieren
- Versuchen Sie, durch „genaues Hinschauen“ die Übertragungsfunktion der SFGs zu erkennen.

1.5. * Kritischer Pfad und Ressourcenverbrauch 1 → M1.5

In Abb. 1.8 sind Systeme zweiter Ordnung in verschiedenen Arten von Direktform dargestellt. Bestimmen Sie Ressourcenverbrauch (= Anzahl Addierer, Multiplizierer, Register) der Systeme,

den kritischen Pfad und die dadurch begrenzte Taktfrequenz sowie die Anzahl der Multiplikationen pro Input Sample (MPIS). Eine Multiplikation soll $\tau_{mul} = 12$ ns und eine Addition oder Subtraktion $\tau_{add} = 3$ ns dauern.

Bestimmen Sie die Verzögerung des kritischen Pfads der Systeme für die Ordnung N . Berechnen Sie daraus die max. Taktfrequenz für $N = 8$ für schnellstes und langsamstes System.

1.6. * Kritischer Pfad und Ressourcenverbrauch 2 → M1.6

Bestimmen Sie die für die Systeme in Abb. 1.9 den Ressourcenverbrauch (Addierer, Multiplizierer, Register) und die Verzögerungszeit τ_{krit} des kritischen Pfads sowie die maximale Taktfrequenz. Eine Multiplikation soll $\tau_{mul} = 12$ ns und eine Addition oder Subtraktion $\tau_{add} = 3$ ns dauern.

Vergleichen Sie Ressourcenverbrauch und kritischen Pfad der Systeme zweiter Ordnung Abb. 1.9b) bis c) mit dem eines Systems in Direktform 2 (Abb. 1.8(d)).

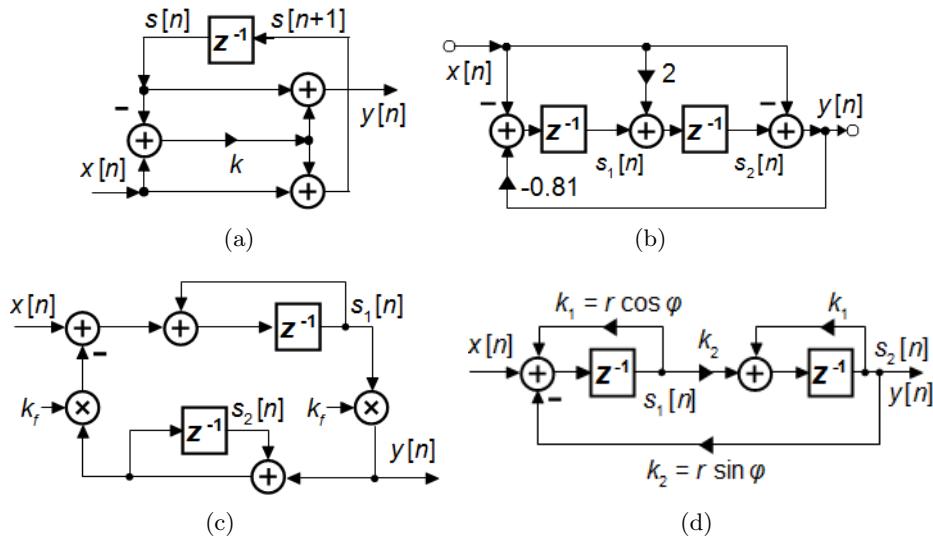


Abb. 1.9.: Filterstrukturen zu Aufgabe 1.6, 1.7, 2.6 und 12.1

1.7. * Transponierte Systeme zu Aufgabe 1.6 → M1.7

Transponieren Sie die Systeme aus Abb. 1.9 und bestimmen Sie wiederum Ressourcenverbrauch und kritischen Pfad / max. Taktfrequenz (Angaben siehe Aufgabe 1.6).

1.8. * Allgemeine Fragen zu FPGAs → M1.8

- Nennen Sie Vor- und Nachteile von FPGAs im Vergleich zu ASICs, DSPs und uCs.
- Warum werden Multiplizierer und Speicher oft als optimierte Hardmakros auf FPGAs implementiert? Welchen Nachteil haben Hardmakros?

- c) Welche Technologien werden zur Herstellung von FPGAs heutzutage überwiegend verwendet? Welche Vor- und Nachteile haben diese?

2. LTF: LTI-Systeme im Frequenzbereich

Inhalt

Dieses Kapitel beschäftigt sich mit LTI-Systemen im *Frequenzbereich*. Nach dem Durcharbeiten dieses Kapitel sollten Sie die folgenden Eigenschaften von zeitdiskreten LTI-Systemen verstanden haben und diese u.a. aus der Systemfunktion $H(z)$ bestimmen können:

Pol-/Nullstellendiagramme in der z -Ebene

Stabilität und Kausalität

Komplexer Frequenzgang $H(e^{j\Omega})$

Betrag- und Phasengang $|H(e^{j\Omega})|$ und $\angle H(e^{j\Omega})$

Gruppenlaufzeit $\tau_g(H(e^{j\Omega}))$

Außerdem sollten Sie wissen, was ein Moving Average Filter ist und wie Sie aus dem P/N Diagramm die Charakteristik und den Betragsgang eines Systems abschätzen können.

Moodle:

Bitte schauen Sie sich das Video und die zugehörigen Folien zu Kapitel 2 vor der Vorlesungs/Übungsstunde an.

Python: Befehle und Programme zu Kap. 2

Definition der Systemfunktion

Aus der Systemfunktion $H(z)$ können die meisten Eigenschaften des Systems abgeleitet (Impulsantwort, komplexer Frequenzgang, P/N-Diagramm,...) werden. Daher werden Systeme in Python / Matlab meist ausgehend von der Systemfunktion definiert:

$$H(z) = \frac{2z - 1}{z^2 + 0,81} = \frac{2z^{-1} - z^{-2}}{1 + 0,81z^{-2}} = 2 \frac{z - 0,5}{(z + 0,9j)(z - 0,9j)}$$
$$b = [2, 1]; a = [1, 0, 0.81] \quad N = [0.5]; P = [-0.9j, 0.9j]; k = 2$$

b,a sind die Koeffizienten von Zähler- und Nennerpolynom der Systemfunktion $H(z)$. Ausgehend von der Polynomform mit *positiven* Exponenten werden die Koeffizienten in *fallender Ordnung* angegeben, ansonsten in aufsteigender Ordnung. Vergessen Sie nicht Koeffizienten mit Wert Null anzugeben!

Bei FIR-Systemen muss **a** nicht angegeben werden (Default: **a = 1**), ansonsten muss die Ordnung des Nennerpolynoms mindestens gleich groß der des Zählerpolynoms sein. Bei einem All-Pole System muss **b = 1** angegeben werden.

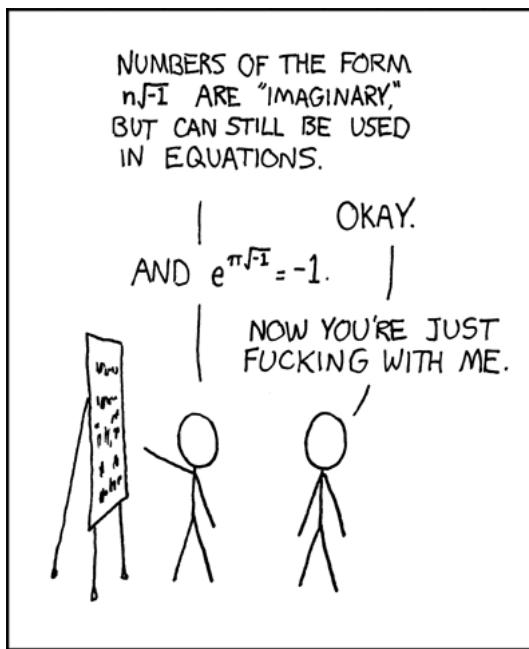


Abb. 2.1.: Imaginäre Erklärungen [<http://xkcd.com/179/>]

N, P sind Null- und Polstellen der Systemfunktion in Produktform, ggf. muss ein Vorfaktor **k** angegeben werden. Bei FIR-Systemen muss **P** nicht angegeben werden (Default: **P = 0**).

Achtung: P/N müssen in Matlab als *Spaltenvektoren* angegeben werden, also mit Semikolons als Trenner, damit `zplane` weiß was gemeint ist:

`N = [0.5]; P = [-0.9*j; 0.9*j]; k = 2;` N/P als *Spaltenvektor* (**Matlab**)

In Python wird nicht unterschieden (Semikolons als Trenner sind Syntaxfehler), die meisten Funktionen entscheiden selbstständig welche Form für einen eindimensionalen Vektor sinnvoll ist und transponieren ihn bei Bedarf.

Umrechnung zwischen beiden Formen ist möglich mit den aus dem letzten Kapitel bekannten Befehlen `N = roots(b); P = roots(a);` und in die andere Richtung
`b = k * poly(P); a = poly(N);`

Funktion	Python	Matlab	Beschreibung
Impulsantwort	<code>h,t = dsp.impz(b,a)</code>	<code>[h,t] = impz(b,a)</code>	Impulsantwort des durch b,a definierten Systems.
Filterung	<code>y = sig.lfilter(b,a,x)</code>	<code>y = filter(b,a,x)</code>	Filtere das Signal x mit dem durch b,a definierten System.
Anfangsbedingungen	<code>zi = sig.lfilter_zi(b,a)</code>		Berechne Ausgangszustand zi , der eingeschwungenem Zustand des Filters entspricht
	<code>zi = sig.lfiltic(b, a, y, x=None)</code>	<code>zi = filtic(b,a,x[,y])</code>	Berechne eingeschwungenen Zustand des Filters
Zero-Phase Filterung	<code>y = sig.filtfilt(b,a,x)</code>	<code>y = filtfilt(b,a,x)</code>	Filterung ohne Phasengang/-Gruppenlaufzeit mit dem durch b,a definierten System.

Funktion	Python	Matlab	Beschreibung
Frequenzgang (komplex)	<code>[W,H] = sig.freqz(b,a)</code>	<code>[H,W] = freqz(b,a)</code>	Berechne den komplexen Frequenzgang entlang des Einheitskreises.
Pole und Nullstellen	<code>dsp.zplane(b,a)</code>	<code>zplane(b,a)</code>	Plotte P/N - Plan ausgehend von Koeffizienten b, a .
	<code>dsp.zplane(N,P,zpk=True)</code>	<code>zplane(N,P)</code>	Plotte P/N - Plan ausgehend von Null/Polstellen N, P .
	<code>[b,a] = sig.zp2tf(z,p,k)</code>	<code>[b,a] = zp2tf(z,p,k)</code>	Umrechnung $N, P, k \rightarrow b, a$
	<code>[z,p,k] = sig.tf2zpk(b,a)</code>	<code>[z,p,k] = tf2zpk(b,a)</code>	Umrechnung $b, a \rightarrow N, P, k$

Tab. 2.1.: Spezielle Befehle zu Kap. 2

Anmerkungen zu Tab. 2.1:

impz Optional: Anzahl der Punkte, Abtastfrequenz.

filtfilt Vorwärts/Rückwärtsfilterung eines Datenblocks mit dem gleichen Filter eliminiert den Phasengang (und damit auch die Gruppenlaufzeit), das funktioniert natürlich nur offline mit Datenblöcken, da man die Zeit nicht rückwärts laufen lassen kann!

(l)filter Achtung: In Python ist `filter(function, iterable)` eine Standard-Funktion, die Elemente aus einer Liste herausfiltert. Mit `sig.lfilter(b,a,x,zi)` können optional Anfangsbedingungen übergeben werden (genauso in Matlab).

freqz gibt den *komplexen* Frequenzgang $H(e^{j\Omega})$ des Systems in Abhängigkeit von der normierten Kreisfrequenz zurück. Daraus können mit `abs(H)` bzw. `angle(H)` leicht Betrag bzw. Phase abgeleitet werden, also z.B. `[H,W] = freqz(b,a); plot(W / (2 * np.pi), abs(H))`.

Optionale Parameter:

Python: `freqz(b, a=1, worN=None, whole = False)`
`worN=1024` nimmt 1024 Datenpunkte
`whole = True`: Plotten über den gesamten Frequenzbereich $\Omega = 0 \dots 2\pi$

Matlab: `freqz(b,a,n=1024, fs=1000)`
`fs=1000`: Angabe der Abtastfrequenz zur Skalierung des zurückgegebenen Frequenzvektors in physikalischen Frequenzen

zplane gibt es in Python on Haus aus bislang nicht, Sie finden die Funktion in meiner dsp - Lib.

zp2tf ist speziell für zeitdiskrete Systeme, für zeitkontinuierliche Systeme gibt es die Funktionen **zp2tf**

Filename	Beschreibung
LTF_sensor_simple	Grundlegender Code zu Aufgabe 2.1 „Filterung des Sensorsignals im Frequenzbereich“
LTF_sensor	Code zu Aufgabe 2.1 „Filterung des Sensorsignals im Frequenzbereich“ mit aufwändigerer Grafik

Filename	Beschreibung
LTF_IIR_allgemein	Code zu Aufgabe 2.8 „Allgemeine IIR-Struktur“
LTF_ML_Notchfilter	Musterloesung zu Aufgabe 2.11 „Notchfilter“
LTF_system-properties	Anzeige von Pol-Nullstellenplan, Amplitudengang, Phasengang, 3D-Übertragungsfunktion etc. eines zeitdiskreten Systems, definiert über Zähler- und Nennerkoeffizienten (Polynomform)

Tab. 2.2.: Simulationsfiles zu Kap. 2

Experimente und Lesestoff zu Kap. 2

Zero-Phase Filtering <http://www.embedded.com/design/configurable-systems/4008847/DSP-Tricks-Doing-Zero-phase-filtering>

2.1. * Filterung des Sensorsignals im Frequenzbereich → M2.1

Die Systeme aus Aufgabe 1.1 sollen nun im Frequenzbereich untersucht werden. Bestimmen Sie für die Systeme aus Abb. 1.6 die folgenden Eigenschaften und überprüfen Sie die Rechnungen durch Simulationen in Python / Matlab. Beim verlustbehafteten Integrator soll $\alpha = 0,9$ gelten.

- a) Komplexer Frequenzgang $H(e^{j\Omega})$ und Betragsgang $|H(e^{j\Omega})|$
- b) DC-Verstärkung $H(\Omega = 0)$
- c) Unterdrückung des Störsignals bei $f = 50$ Hz für $f_{S,2} = 200$ Hz und $f_{S,3} = 240$ Hz
- d) Pol/Nullstellen-Diagramm
- e) Phasengang $\varphi(e^{j\Omega})$ und Gruppenlaufzeit $\tau_g(e^{j\Omega})$ Welche Werte erhält man bei $f = 50$ Hz und $f_{S,2} = 200$ Hz?

2.2. * Frequenzgang einfacher FIR-Filter → M2.2

Das FIR Filter aus Aufgabe 1.2 mit $h[n] = \{0,5; 1\}$ soll jetzt in der Frequenzebene untersucht werden.

- a) Bestimmen Sie die Übertragungsfunktion $H(z)$ und ihre Nullstellen und Pole. Zeichnen Sie diese in der komplexen z -Ebene ein.
- b) Bestimmen Sie den komplexen Frequenzgang $H(e^{j2\pi f})$ sowie seinen Betrag und seine Phase. Berechnen Sie diese Werte für $f = 0, 0,25f_S$ und $0,5f_S$.

2.3. * Systemfunktion aus Impulsantwort → M2.3

Ermitteln Sie für die Systeme aus Aufgabe 1.3 die Übertragungsfunktionen $H(z)$, bestimmen Sie Pole / Nullstellen und zeichnen Sie sie auf.

Welches Frequenzverhalten erwarten Sie aufgrund der Lage der Pol- und Nullstellen? Überprüfen Sie Ihre Antwort mit einer Matlab / Python Simulation, nutzen Sie dazu den Befehl `freqz`.

2.4. * Systemeigenschaften aus Übertragungsfunktion → M2.4

Die folgenden Systeme sind durch ihre Systemfunktion $H(z)$ definiert:

a) $H_a(z) = \frac{(z - 1)(z + 1)}{z^2}$

b) $H_b(z) = z^{-1} + z^{-3}$

c) $H_c(z) = (z - 1)^3$

d) $H_d(z) = \frac{1}{z^2 + 0,64}$

Bestimmen Sie jeweils

- Kausalität und Stabilität: Wie kann man ggf. nichtkausale Systeme in kausale Systeme umwandeln?
- Impulsantwort
- Pole und Nullstellen: Wandeln Sie hierzu die Übertragungsfunktion in ihre Produktform um
- Betrags- und Phasengang: Versuchen Sie, einen Faktor e^{jx} so auszuklammern, dass sich die restlichen Terme zu Sinus- und Cosinusfunktionen zusammenfassen lassen.

und zeichnen Sie:

- Pol- / Nullstellenplan
- Betragsgang: Nutzen Sie hierfür auch den P/N-Plan und berechnen Sie $H(f = 0)$, $H(f = f_S/4)$ und $H(f = f_S/2)$
- Eine Hardware-Implementierung

Überprüfen Sie die Ergebnisse auch mit Matlab / Python - Simulationen.

2.5. * Systemeigenschaften aus Pol-/Nullstellenplan → M2.5

Schätzen Sie folgende Eigenschaften der in Abb. 2.2 durch ihre Pol/Nullstellenpläne gegebenen Systeme ab:

- a) FIR oder IIR - Filter
- b) Kausalität
- c) Stabilität
- d) Frequenzverhalten

Wandeln Sie instabile Systeme durch Streichen von möglichst wenigen Pol- und/oder Nullstellen in stabile Systeme um! Wandeln Sie nicht-kausale Systeme durch Streichen oder Ergänzen von Pol- oder Nullstellen in kausale Systeme um. Die Frequenzcharakteristik soll sich dabei nicht ändern.

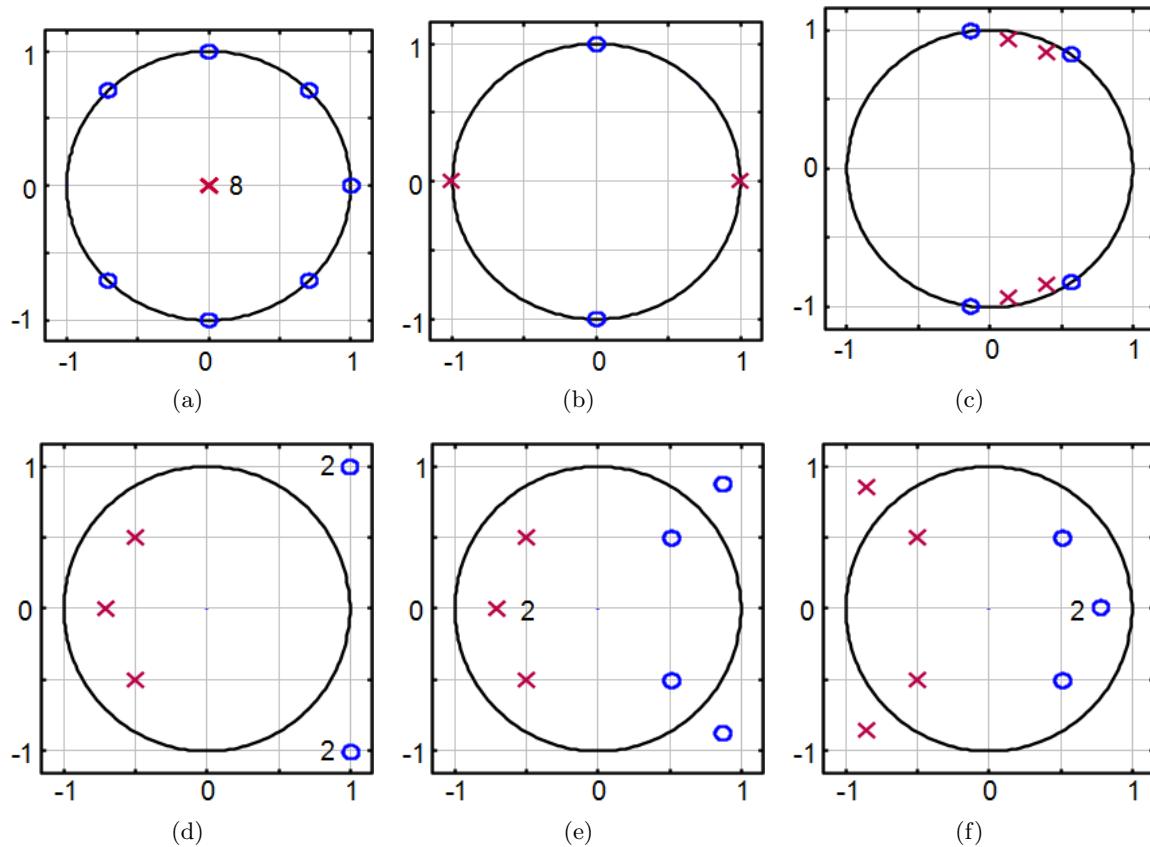


Abb. 2.2.: Pol-/Nullstellenpläne zu Aufgabe 2.5

2.6. * Frequenzgang aus Signalflussdiagramm → M2.6

Bestimmen Sie die für die Systeme in Abb. 1.9:

1. Systemfunktion $H(z)$ und Pol-/Nullstellen

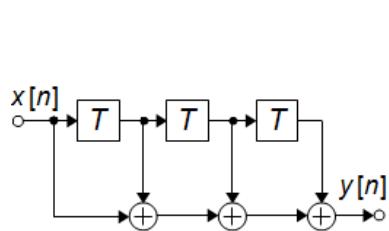
2. Charakteristik im Frequenzbereich (Hochpass, Tiefpass, ...)

Wenn Sie sich noch unsicher sind im Aufstellen der Systemfunktion aus einem SFG, können Sie als zusätzliche Übungsaufgaben versuchen, die Systemfunktion der transponierten Systeme (Abb. M1.8) zu bestimmen (die natürlich identisch sein muss mit der ursprünglichen Systemen).

2.7. * Einfaches Moving Average Filter → M2.7

Gegeben ist die im Abb. 2.3(a) gezeigte FIR-Filterstruktur.

- Ermitteln Sie Impulsantwort $h[n]$ und Systemfunktion $H(z)$ des Filters.
- Die Simulation ergibt den Amplitudengang $|H(e^{j\Omega})|$ in Abb. 2.3(b). Berechnen Sie $H(f = 0)$ sowie die Nullstellen von $H(z)$.
- Ermitteln Sie den komplexen Frequenzgang $H(e^{j\Omega})$ und den Amplitudengang $|H(e^{j\Omega})|$
- Ermitteln Sie den Phasengang $\varphi(e^{j\Omega})$ sowie die Gruppenlaufzeit $\tau_g(e^{j\Omega})$.
- Bestimmen Sie die Systemfunktion und den komplexen Frequenzgang für den Fall, dass die gegebene Filterstruktur auf N Taps erweitert wird. Der Vorfaktor wird dabei so gewählt, dass $H(f = 0) = 1$ ist. Welchen Amplitudengang nähert man mit $N \rightarrow \infty$ immer mehr an? Wozu könnte dieses Filter dienen?
- Schreiben Sie Python / Matlab Code zur Darstellung des Frequenzgangs eines MA-Filter mit $N = 32$



(a)

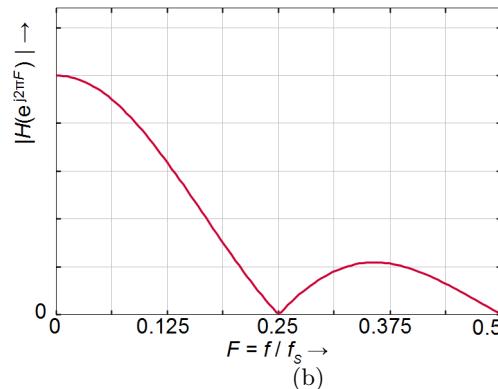
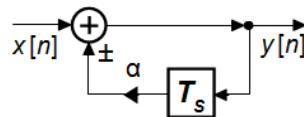


Abb. 2.3.: FIR-Filterstruktur (a) und Amplitudengang (b) zu Aufgabe 2.7

2.8. * Allgemeine IIR-Struktur → M2.8

Ausgehend von der nebenstehenden allgemeinen IIR-Struktur sollen die Eigenschaften der folgenden Varianten untersucht werden:



- a) Addierer, ein Verzögerungsglied T_S
- b) Subtrahierer, ein Verzögerungsglied T_S
- c) Addierer, zwei Verzögerungsglieder T_S
- d) Subtrahierer, zwei Verzögerungsglieder T_S

Die Eigenschaften sollen für alle vier Varianten jeweils für die Koeffizientenwerte $\alpha = \alpha_1 = 0,9$ und $\alpha = \alpha_2 = 0,5$ bestimmt werden.

1. Bestimmen Sie zunächst die Übertragungsfunktion $H(z)$ der vier Varianten.
2. Geben Sie für jede Variante an bzw. skizzieren Sie:
 - den Filtertyp (TP, HP, BP, BS)
 - das Pol-Nullstellen-Diagramm
 - Minimum und Maximum der Betragsfunktion $|H(e^{j2\pi F})|_{min}$ und $|H(e^{j2\pi F})|_{max}$ und bei welcher Frequenz diese auftreten.
3. Skizzieren Sie den Verlauf der Betragsfrequenzgänge $|H(e^{j2\pi F})|$ im Basisband $0 \leq F \leq 0,5$ ohne $H(e^{j2\pi F})$ formelmäßig abzuleiten
4. Leiten Sie die Betragsfrequenzgänge formelmäßig her
5. Skizzieren Sie die Impulsantworten, versuchen Sie die Impulsantworten zu berechnen [nicht prüfungsrelevant für Systeme c) und d)]
6. Schreiben Sie ein Python / Matlab-Skript, um P/N-Diagramm, Frequenzgang und Impulsantwort darzustellen. Nutzen Sie hierzu die Befehle `zplane`, `freqz` und `impz`.

2.9. IIR-Filter zweiter Ordnung → M2.9

Gegeben ist das IIR-Filter in Abb. 2.4(a).

- a) Geben Sie die Differenzengleichung des IIR-Filters an.
- b) Ermitteln Sie aus der Differenzengleichung die Impulsantwort $h[n]$ für $n = 0, \dots, 6$.
- c) Ermitteln Sie für das in Abb. 2.4(b) skizzierte Eingangssignal $x[n]$ das Ausgangssignal $y[n]$ für $n = 1, \dots, 5$ als Summe von bewerteten Impulsantworten und skizzieren Sie es.

Im Weiteren gelte eine Abtastperiode von $T_S = 10 \mu s$:

- d) Ermitteln Sie die Übertragungsfunktion $H(z)$ in Polynomform und ihre Nullstellen und Pole. Zeichnen Sie diese in der komplexen z-Ebene ein. Um welche Art von Filter handelt es sich (TP, HP, BP, BS)?
- e) Ermitteln Sie Betrag und Phase des komplexen Frequenzgangs $H(e^{j2\pi f/f_S})$ bei der Frequenz $f = 25 \text{ kHz}$ direkt aus der Polynomform (nicht ausmultiplizieren!)

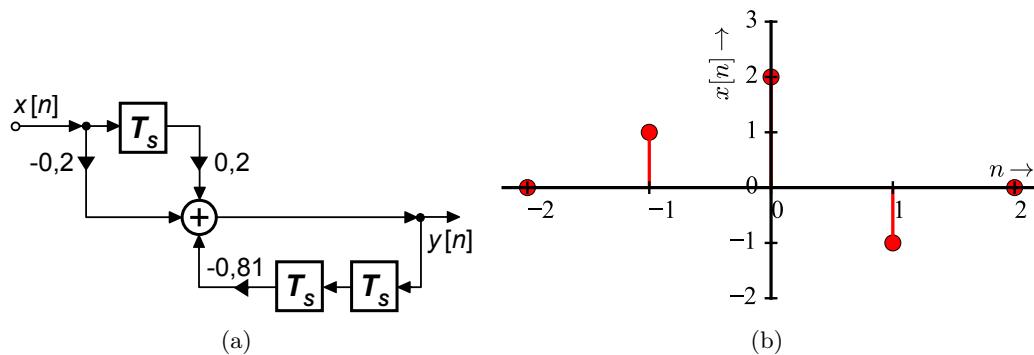


Abb. 2.4.: IIR-Filter (a) mit Eingangssignal (b) zu Aufgabe 2.9

- f) Stellen Sie $H(z)$ nun in der Produktform dar und berechnen Sie nochmals die im vorigen Punkt gefragten Größen.
- g) Wie müsste man welche Filterkoeffizienten ändern, damit bei unveränderter Filtercharakteristik die Verstärkung bei 25 kHz den Betrag 1 hat?
- h) Skizzieren Sie auf dem Angabenblatt unter Verwendung der Ergebnisse von e) bzw. f) das Ausgangssignal $y[n]$ (eingeschwungen) für periodische Eingangssignal in Abb. 2.5 (Abtastwerte einer 25 kHz-Schwingung mit Scheitelwert 1). Zeichnen Sie hierzu zunächst die sinusförmige 25 kHz-Schwingung am Ausgang ein und entnehmen Sie daraus die Abtastwerte.

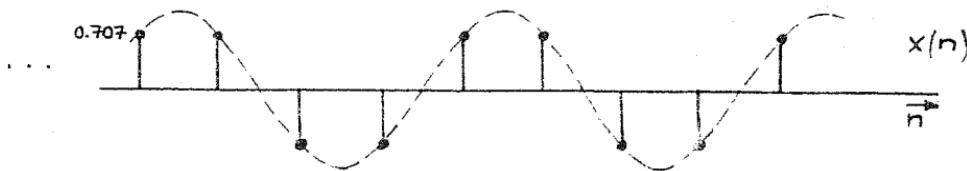


Abb. 2.5.: Periodisches Eingangssignal zu Aufgabe 2.9h)

2.10. Moving Average Filter / rect-Fenster → M2.10

Es sollen die Eigenschaften von Moving Average Filtern mit der Länge $N = 32$ und 1024 verglichen werden, die mit einer Taktrate von $f_S = 10,24$ MHz betrieben werden. Berechnen Sie jeweils:

- a) die Betragsantwort des Frequenzgangs $|H(f)|$ und die Verstärkung $|H(f = 0)|$
- b) die Frequenz, bei der der Frequenzgang die erste Nullstelle hat und die Amplitude des größten Nebenzipfels relativ zum Hauptzipfel (näherungsweise)

2.11. Notchfilter → M2.11

Ein mit $f_S = 10$ kHz abgetastetes Audio-Signal enthält einen starken Störton bei $f_1 = 3$ kHz, der vollständig unterdrückt werden soll. Bestimmen Sie für die folgenden drei Ansätze jeweils

Übertragungsfunktion $H(z)$, Betragsgang $|H(e^{j\Omega})|$ und dessen Größe bei den Frequenzen $f = 0$, $f = f_S/2$ und $f = f_1 \pm 100$ Hz. überprüfen Sie die Rechnung jeweils mit Matlab.

- a) Entwerfen Sie durch Platzieren von Nullstellen ein einfaches reellwertiges FIR-Filter, das diese Frequenz vollständig unterdrückt. Welche minimale Ordnung hat das Filter?
- b) Welchen Nachteil hat dieses einfache Filter?
- c) Ersetzen Sie die einfachen Nullstellen durch doppelte Nullstellen. Werden die Nachteile von H_1 dadurch vermieden?
- d) Durch Hinzufügen von Polen bei dem gleichen Winkel wie die Nullstellen lässt sich das Verhalten von H_1 deutlich verbessern. Entwerfen Sie ein Notchfilter H_3 mit zwei Null- und zwei Polstellen mit dem Polradius $r = 0,95$.

3. DFT: Diskrete Fouriertransformation und FFT

Inhalt

Dieses Kapitel beschäftigt sich mit der Fouriertransformation zeitdiskreter Signale, vor allem mit der *Discrete Fourier Transform* (DFT). Als Grundlagen werden vorausgesetzt:

Kohärente Abtastung: Mit der DFT eines periodischen, abgetasteten Signals über eine ganzzahlige Anzahl Perioden kann man näherungsweise die Fourierreihe des zeitkontinuierlichen Signal berechnen. Hierbei geht es um die Herleitung der DFT aus DTFT und CFT, die Frequenzauflösung in Abhängigkeit von Abtastperiode und Anzahl der Abtastpunkte sowie um die korrekte Skalierung der DFT.

Fast Fourier Transform beschreibt verschiedene effiziente Implementierungen der DFT in Hard- und Software.

Neue Themen

Inkohärente Abtastung: Die DFT eines periodischen Signals über eine gebrochene Anzahl Perioden erzeugt Fehler im Vergleich zur Fourierreihe: Der Leckeffekt beschreibt dass Spektrallinien an und Amplitudenfehler („Lattenzauneffekt“).

Zero-Padding: Durch das Anhängen von Nullen an eine gefensterte Messsequenz kann die Punktezahl auf einen passenden Wert für eine FFT (z.B. $N = 2^r$) erhöht und so die Rechenzeit verringert werden. Durch die höhere Anzahl Punkte werden Zwischenwerte im Spektrum berechnet (Interpolation in der Frequenzebene), so kann der Amplitudenfehler verringert werden.

Fourier-Transformation stationärer Prozesse: Mit Hilfe der DFT kann die spektrale Leistungsdichte (Power Spectral Density, PSD) von stochastischen stationären (zeitlich konstanten Kennwerten) Prozessen (z.B. von periodischen Signalen mit additivem Rauschen) geschätzt werden.

Vorbereitung

Bitte schauen Sie sich bei Bedarf die Videos (deutsch) von Jörg Lovisach auf Youtube an, der sehr schön die Grundlagen von Fourier-Transformation, DFT und FFT erklärt:

„01.04.1 Fourier-Reihe“ (<https://www.youtube.com/watch?v=g5Y25jsliFk>)
„01.04.2 Fourier-Transformation“ (<https://www.youtube.com/watch?v=ug9Co75tQuo>)

„19.03.1 Diskrete Fouriertransformation“ (<https://www.youtube.com/watch?v=ls5MfqLZVbo>)
„19.03.2 DFT mit Fensterung“ (<https://www.youtube.com/watch?v=wd6hAPS4ILc>)
„17.3 FFT in MATLAB(R) und Fensterung“ (https://www.youtube.com/watch?v=7bi5_KC08Kg)

Lesen Sie begleitend zur Vorlesung den Artikel „How to use the FFT and Matlab's pwelch function for signal and noise simulations and measurements“ von Hanspeter Schmid, FHNW/IME, 2012. (<http://www.fhnw.ch/technik/ime/publikationen/2012/> oder auf Moodle).

Theorie

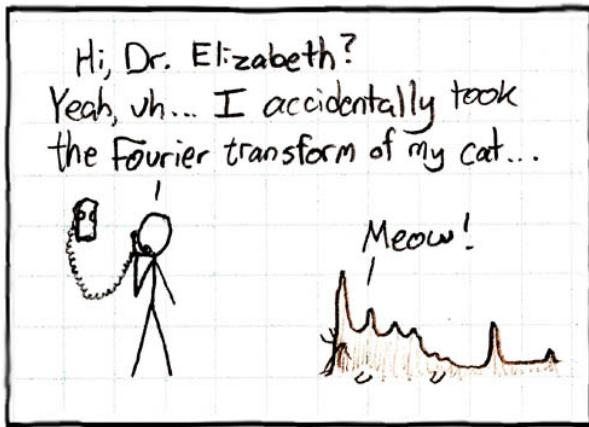


Abb. 3.1.: „That cat has some serious periodic components“. [<http://xkcd.com/26/>]

Energie und Leistung im Zeit- und Frequenzbereich

Wenn die DFT zur Messung und Abschätzung physikalischer Vorgänge verwendet wird, ist es sinnvoll sie so zu skalieren, dass Energie bzw. Leistung eines Signals im Zeit- und Frequenzbereich gleich sind.

Für **Energiesignale** (endliche Zeitspanne) wird die Äquivalenz durch das Parsevalsche Gesetz beschrieben:

$$E = \int_{-\infty}^{\infty} |y(t)|^2 dt = \int_{-\infty}^{\infty} |Y(f)|^2 df = \int_{-\infty}^{\infty} Y(f) \cdot Y^*(f) df$$

Periodische, unendlich ausgedehnte Signale haben unendliche Energie. Bei solchen Signalen betrachtet man einen Signalausschnitt, am besten über eine Periode T_1 mit $y_{T1}(t) = y(t)$ für $|t| < T_1/2$, 0 sonst (Satz von Plancherel):

$$P = \frac{1}{T_1} \int_{-\infty}^{\infty} |y_{T1}(t)|^2 dt = \frac{1}{T_1} \int_{-\infty}^{\infty} |\mathcal{F}(y_{T1}(f))|^2 df$$

Bei statistischen Prozessen ist die Fouriertransformation nicht definiert, hier betrachtet man stattdessen die Fouriertransformierte der Autokorrelationfunktion des Prozesses (Wiener-Kinchine-Theorem).

Bei zeitdiskreten Signalen werden aus den Integralen unendliche (DTFT) bzw. endliche (DFT) Summen:

$$E = \sum_{n=-\infty}^{\infty} |y[n]|^2 = \int_{-1/2}^{1/2} |Y(e^{j2\pi F})|^2 dF \quad (\text{DTFT zeitdiskreter Energiesignale})$$

bzw.

$$P = \frac{1}{N_1} \sum_{n=0}^{N_1-1} |Y[n]|^2 \quad (\text{DFT zeitdiskreter periodischer Signale})$$

Rechenaufwand

Aus der Definitionsgleichung der DFT (B.24) bzw. deren SFG (Abb. 3.2) kann man sofort ablesen, dass die Berechnung von N_{DFT} Frequenzpunkten aus N_{DFT} Zeitpunkten im allgemeinen Fall N_{DFT}^2 komplexe oder $4N_{DFT}^2$ reellwertige Multiplikationen und Additionen (= MAC) erfordert¹. Bei reellwertigen Eingangssignalen sind nur die Hälfte der Rechenoperationen erforderlich. Nutzt man die Symmetrie des Spektrums aus, muss man nur $N/2$ Frequenzpunkte berechnen, so dass nur noch

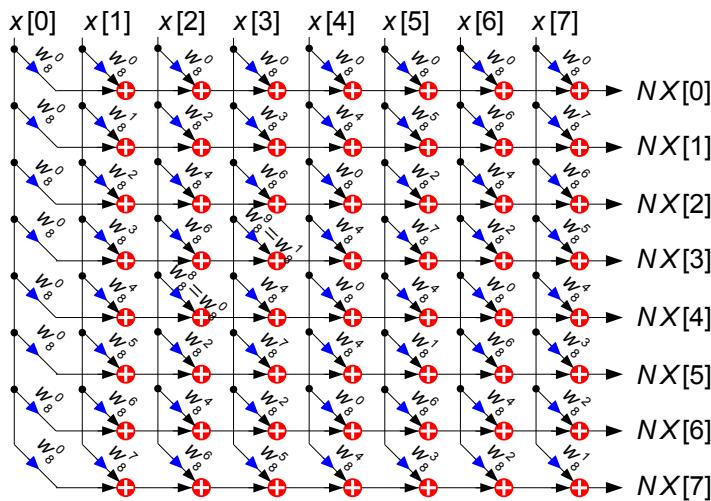


Abb. 3.2.: Signalflussgraph einer DFT mit 8 Punkten

$$N_{MAC,DFT} \approx N_{DFT}^2 \text{ reellwertige Rechenoperationen} \quad (3.1)$$

erforderlich sind. In dieser Abschätzung ist nicht berücksichtigt, dass ein Teil der Rechenoperationen trivial ist (z.B. Multiplikation mit ± 1).

Wenn die Anzahl der Datenpunkte eine Zweierpotenz ist, $N_{FFT} = 2^m$, kann die DFT in kurze Teil-DFTs zerlegt und sehr effizient als sog. *Radix-2 FFT* (Fast-Fourier Transform) berechnet werden, die nur noch ca. $N_{FFT}/2 \log_2 N_{FFT}$ komplexe oder

$$N_{MAC,FFT} \approx 2N_{FFT} \log_2 N_{FFT} \text{ reellwertige Rechenoperationen} \quad (3.2)$$

benötigt. Auch hier werden wieder die Additionen mit den Multiplikationen gemeinsam gerechnet bzw. vernachlässigt.

Der SFG in Abb. 3.3 lässt sich sehr gut mit Pipeliningregistern zwischen den Stufen implementieren, so dass ein sehr hoher Datendurchsatz erreicht werden kann. Die Beschränkung auf

¹Multiplikation und Addition werden als eine Operation gerechnet, da DSPs, FPGAs und die meisten uCs entsprechende Rechenwerke haben.

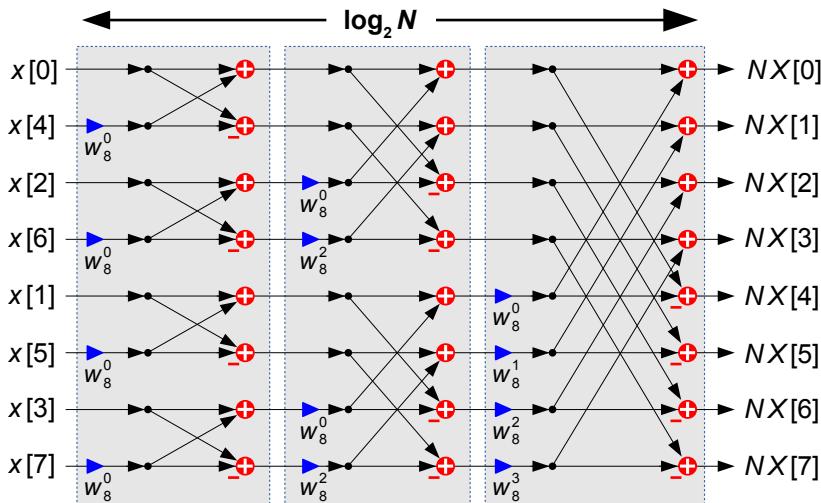


Abb. 3.3.: Signalflussgraph einer FFT mit 8 Punkten

reellwertige Eingangssignale spart hier allerdings kaum Rechenoperationen ein, da die zu verarbeitenden Zwischensignale überwiegend komplex sind. Immerhin müssen nur halb so viele Frequenzpunkte abgespeichert werden. In modernen Rechnerarchitekturen dominieren inzwischen oft die Zeiten für das Verschieben und Abspeichern der Daten.

Eine ausführlichere Darstellung von verschiedenen FFT Algorithmen finden Sie z.B. in <http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/transform/fft.html>

Python: Befehle und Programme zu Kap. 3

Funktion	Python	Matlab	Beschreibung
FFT	<code>X = fft.fft(x, NFFT)</code>	<code>X = fft(x, NFFT)</code>	Berechne DFT X mit NFFT komplexen Werten je Dimension zu NFFT reellen oder komplexen Werten x (ein- oder mehrdimensional)
	<code>fftr(x, NFFT)</code>	--	Berechne DFT zu <i>reeller</i> Zeitfolge x (nur Python)
Zweiseitiges Spektrum	<code>XS = fftshift(X)</code>	<code>XS = fftshift(X)</code>	Verschiebe DFT so, dass der Spektralwert zu $f = 0$ in der Mitte liegt.
Frequenzvektor	<code>f = fftfreq(NFFT, TS)</code>	<code>f = fftfreq(NFFT, TS)</code>	Array mit Frequenzpunkten passend zu den Punkten der FFT für ein zweiseitiges Spektrum (nur Python), also $f = [0, \dots, fs[$.

Tab. 3.1.: Spezielle Befehle zu Kap. 3

Anmerkungen zu Tab. 3.1:

Allgemein: Funktionen rund um DFT/FFT finden sich in Python im FFT Modul von Scipy /Numpy (`numpy.fft`).

„**FFT**“ ist in vielen Programmiersprachen (so auch in Python und Matlab) ein Platzhalter für eine Reihe von Algorithmen, aus denen je nach Randbedingungen (Anzahl der Punkte, reell/komplex etc.) der am besten geeignete ausgewählt wird. Der Befehl kann daher mit einer beliebigen Anzahl Punkte (nicht nur Zweierpotenzen) verwendet werden, auch wenn es für $N_{FFT} = 2^m$ besonders effiziente Algorithmen gibt.

„FFT“ wird oft gleichbedeutend mit „DFT“ verwendet, auch wenn das nicht ganz korrekt ist: DFT beschreibt die mathematische Methode / Transformation, FFT ist eine effizienter Algorithmus zur Berechnung der DFT.

Frequenzvektor: Werte des Frequenzvektor werden beginnend mit $f = 0$ ausgegeben: $f = [0, \dots, f_S[= [0, \dots, f_S/2[, [-f_S/2, \dots, 0[$. Für ein einseitiges Spektrum zwischen $[0, \dots, f_S[$ nimmt man lediglich die erste Hälfte der Punkte, also $S1 = S[0:N_{FFT}/2 - 1]$. Für eine (zweiseitige) Darstellung symmetrisch zu $f = 0$, muss die zweite Hälfte des Vektors mit der ersten vertauscht werden. Dies erledigt der Befehl `fftshift`.

Filename	Beschreibung
Directory „Basics“	
DFT_MA_Filt	Code zu Aufgabe 3.2
DFT_MA_Filt_ML	Musterlösung zu Aufgabe 3.2
DFT_basics_ML	Musterlösung zu Aufgabe 3.5
DFT_Delta_f	Code zu Aufgabe 3.6
DFT_Delta_f_ML	Musterlösung zu Aufgabe 3.6
DFT_coherent.py	Kohärente Abtastung (Periodizität) und DFT
Hauptteil	
DFT_windows	Einfluss von Fensterlänge, -typ und Abtastrate auf das angezeigte Spektrum der DFT
DFT_STFT	Short Time Fourier Transform mit verschiedenen Fenstern und Abtastraten

Tab. 3.2.: Simulationsfiles zu Kap. 3

3.1. * Allgemeine Fragen zu DFT und FFT → M3.1

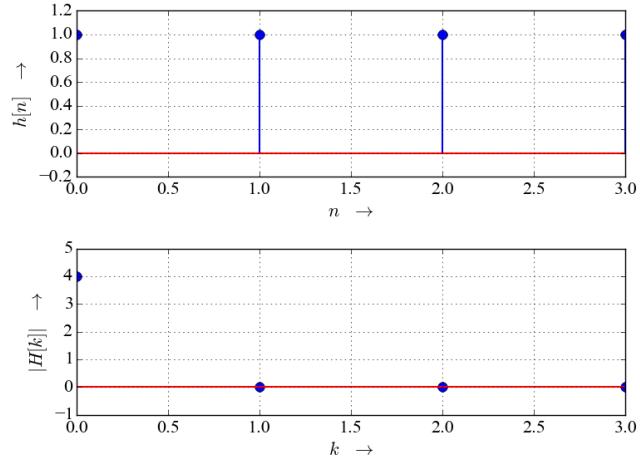
- Nennen Sie die Unterschiede zwischen Fourierreihe, Fouriertransformation, DTFT, DFT und FFT?
- Wie groß ist der Abstand Δf zwischen zwei Frequenzpunkten einer DFT mit N Punkten und der Abtastfrequenz f_S ? Welcher Frequenz f_k entspricht der Frequenzpunkt k ? Welcher Frequenz entspricht der höchste Frequenzpunkt?
- Warum wird die DFT / FFT meist nur in der positiven Hälfte des Nyquistbereichs ($0 \leq f < f_S/2$) ausgewertet?
- Warum benötigt man *immer* eine Fensterung vor Berechnung einer DFT / FFT?

3.2. * DFT eines Moving Average (MA) - Filters → M3.2

Sie wollen aus der Impulsantwort eines MA-Filters der Ordnung $N = 3$, $h_{MA,3} = \{1; 1; 1; 1\}$, dessen Frequenzgang mit Hilfe der DFT bestimmen. Welches Ergebnis erwarten Sie? Sie versuchen dies mit dem Python-Skript in Listing 3.1 und erhalten das dort gezeigte Ergebnis:

```

1 # ... Ende der Import-Anweisungen
2 h = [1,1,1,1]
3 n = arange(len(h))
4 figure(1);
5 subplot(211)
6 stem(n,h)
7 ylabel(r"$h[n] \rightarrow$")
8 xlabel(r"$n \rightarrow$")
9 subplot(212)
10 H = fft(h,256)
11 k = arange(len(H))
12 stem(k,abs(H))
13
14 xlabel(r"$k \rightarrow$")
15 ylabel(r"$|H[k]| \rightarrow$")
16 plt.tight_layout()
17 plt.show()
```



Lst. 3.1: Python Skript zu Aufgabe 3.2
(erster Ansatz)

Abb. 3.4.: Python Plot zu Listing 3.1

Zur Verbesserung der Darstellung versuchen Sie:

- $h = [1,1,1,1,0,0,0,0]; fft(h,8)$
- $fft(h,8)$
- $fft(h,256)$

Interpretieren Sie die Ergebnisse! Entspricht eines davon Ihrer Erwartung?

3.3. * Frequenzauflösung der DFT → M3.3

- a) Was versteht man unter Frequenzauflösung?
- b) Ein Audiosignal soll mit einer FFT mit $N = 2048$ Punkten und einer Frequenzauflösung von $\Delta f = 1$ Hz gemessen werden. Wie groß muss die Abtastrate f_S gewählt werden? Wie groß ist die höchste Frequenz, die noch dargestellt werden kann?

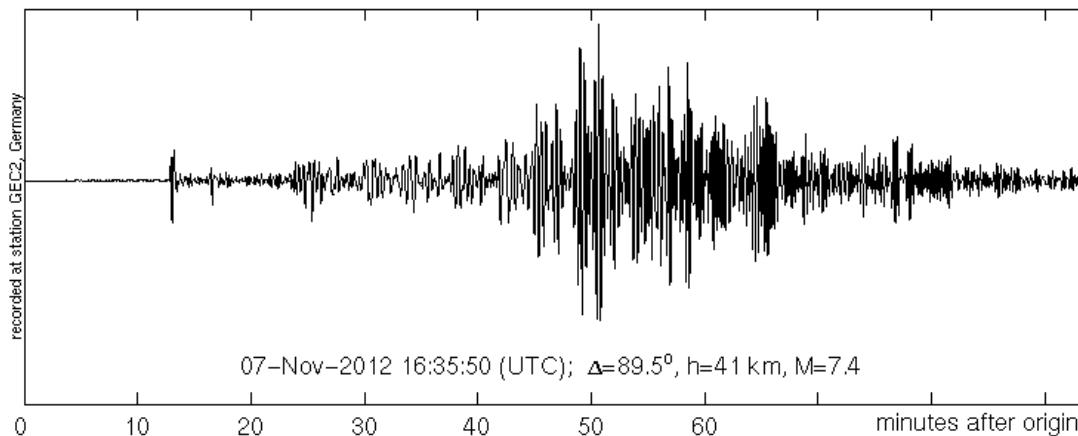


Abb. 3.5.: Seismogramm eines Erdbebens am 7.11.2012 in Guatemala. Mit freundlicher Genehmigung der Bundesanstalt für Geowissenschaften und Rohstoffe [http://www.seismologie.bgr.de/sdac/erdbeben/big_quakes/guatemala_121107_deu.html]

- c) In Abb. 3.5 ist das Seismogramm eines Erdbebens dargestellt, das mit einer Radix-2 FFT spektral analysiert werden soll. Die FFT-Länge kann zwischen 2^8 und 2^{15} gewählt werden, die Abtastrate binär gestuft zwischen $f_S = 0,1 \dots 25,6$ Hz (mit jeweils passendem Anti-Aliasfilter). Interessierende Spektralkomponenten liegen im Bereich $0 \dots 1$ Hz.

Welche Kombination aus FFT-Länge und Abtastrate liefert Ihnen die „besten“ spektralen Informationen, welche Kombination ist am effizientesten unter den angegebenen Randbedingungen?

- d) Bei einer FFT-Länge $N = 2^{11}$ und einer Abtastrate $f_S = 1,6$ Hz tritt eine starke Störlinie bei Frequenzpunkt $k = 620$ auf. Welche Frequenz hat der Störer?

3.4. * Rechenaufwand für DFT und FFT → M3.4

- a) Eine Stereo - Aufnahme der 9. Symphonie von Beethoven mit Laufzeit $T_0 = 74$ min und Abtastrate $f_S = 44,1$ kHz soll vollständig als DFT kodiert dargestellt werden. Welche Länge N der DFT benötigt man hierfür und welche Frequenzauflösung erhält man?
- b) Wieviele reelle Multiplikationen wären zur Berechnung der DFT, wieviele zur Berechnung der FFT notwendig? Bedenken Sie dabei, dass die CD in Stereo aufgenommen wurde.
- c) Wie lange würden die Berechnungen dauern auf einem FPGA mit 512 Hardware - MAC - Units mit jeweils 500 MMAC/s? Die Zeit zum Speichern und Laden der Daten soll vernachlässigt werden.

- d) Welche Rechenleistung in MAC/s ist mindestens erforderlich für eine Real-Time FFT der CD? Nehmen Sie hierfür Fensterlängen von $N_{FFT,1} = 512$ und $N_{FFT,2} = 8192$ an.

3.5. # Fourierreihe und synchrone DFT → M3.5

Das periodische zeitkontinuierliche Signal $s(t) = 1 \text{ V} \cdot (1 + \cos(2\pi 1000 \text{ Hz } t + \pi/4))$ wird dreimal pro Periode abgetastet (das Abtasttheorem ist also erfüllt). Die erste Abtastung erfolgt bei $t = 0$.

- Berechnen und skizzieren Sie zunächst Betrag und Phase der Fourierreihenkoeffizienten c_k des zeitkontinuierlichen Signals $s(t)$ (Continuous Fourier Transform, CFT).
- Berechnen Sie jetzt die ersten drei Werte des abgetasteten Signals $s[n]$. Welchen Wert hat die Abtastperiode T_S ?
- Berechnen und skizzieren Sie Betrag $|S[k]|$ und Phase $\Phi_S[k]$ der DFT dieser drei Abtastwerte $s[n]$, vergleichen Sie das Ergebnis mit der CFT.
- Wieviele Elemente hat die DFT? Wie ordnet man den Index k zu Frequenzwerten zu?
- Was ändert sich, wenn Sie die DFT der ersten 6 Abtastwerte berechnen (gleiche Abtastfrequenz wie vorher)?
- Wie sieht die DFT aus, wenn Sie stattdessen 6 Abtastwerte pro Periode nehmen?
- Wie sieht die DFT aus, wenn Sie stattdessen die ersten drei Abtastwerte durch Anhänger von drei Nullen („zeropadding“) auf 6 Abtastwerte verlängern? Wie müssen Sie den Skalierungsfaktor für die DFT wählen?
- Wieviele Rechenoperationen sind allgemein bei N reellen Abtastwerten zur Berechnung der DFT notwendig?
- Verifizieren Sie Ihre Rechnungen durch Simulation in Python / Matlab, nutzen Sie hierfür den Befehl `fft(x,N_FFT)`.

3.6. * DFT periodischer Signale mit Python / Matlab → M3.6

In Python oder Matlab soll mit einer DFT das Betragsspektrum $|S_y(f)|$ eines Signals $y(t)$ mit

$$y(t) = 1 + 0,5 \sin(2\pi t \cdot 1 \text{ kHz}) + 0,2 \cos(2\pi t \cdot 1,1 \text{ kHz})$$

näherungsweise bestimmt werden.

```

1 # ... Ende der Import-Anweisungen
2 f_S = 51e3; T_S = 1. / f_S
3 N_FFT = 128; t_max = N_FFT * T_S
4 f_a = 1.1e3
5 t = arange(0, t_max, T_S)
6 y = sin(2*pi*t*f_a + pi/2)
7 Sy = fft(y,N_FFT)
8 f = arange(N_FFT)

% 
f_S = 2e3; T_S = 1 / f_S;
N_FFT = 128; t_max = N_FFT*T_S;
f_a = 1e3;
t = 0:T_S:t_max;
y = sin(2*pi*t*f_a);
Sy = fft(y,N_FFT);
f = 0:N_FFT-1;
```

```

9 figure(1); clf()
10 plot(t, y)
11 figure(2); clf()
12 plot(f, Sy); plt.show()

```

Lst. 3.2: DFT mit Python

```

figure(1); clf;
plot(t, y); grid on;
figure(2); clf;
plot(f,Sy); grid on;

```

Lst. 3.3: DFT mit Matlab

- a) Schreiben Sie das **erwartete Betragsspektrum** $|S_y(f)|$ auf (ausführliche Berechnung der CFT ist nicht notwendig - genaues Hinschauen reicht!) und skizzieren Sie es.
- b) **Simulieren Sie das Betragsspektrum** $|S_y(f)|$ mit Hilfe von Listing 3.2 bzw. 3.3 nachdem Sie die Parameter an diese Aufgabe angepasst und ggf. Fehler beseitigt haben.
- c) Was muss gegeben sein, damit das Spektrum **keinen Leckeffekt** zeigt, d.h. damit die erwarteten Spektrallinien sauber dargestellt werden? Wie müssen Sie dazu die Parameter der DFT anpassen? Passen Sie die Frequenzachse so an, dass das Spektrum nicht über den Indices, sondern über den passenden physikalischen Frequenzen dargestellt wird.
- d) Wie können Sie die **Frequenzauflösung der DFT** verbessern? Ändern Sie die Parameter der DFT so, dass Sie die Spektrallinien bei f_a und f_b klar voneinander unterscheiden können.
- e) **Skalieren Sie DFT** jeweils so, dass die Amplituden für ein- und zweiseitiges Spektrum korrekt dargestellt werden. Überprüfen Sie die Skalierung durch Vergleich (analytische Rechnung und Simulation) der Energie bzw. Leistung im Zeitbereich und im Frequenzbereich. Welche Gesamtleistung hat das Signal?
- f) Wie müssen die Parameter geändert werden, damit eine **FFT mit schnellem Radix-2 Algorithmus** mit $N_{FFT} = 2^m$ Punkten verwendet werden kann?

3.7. Kohärente DFT einer Rechteckschwingung → M3.7

Gegeben ist eine zeitkontinuierliche, mittelwertfreie symmetrische Rechteckschwingung mit Grundfrequenz $f_1 = 1/T_1 = 1$ kHz und Amplitude $A = 1$ V:

$$u(t) = \begin{cases} -A & \text{für } -T_1/2 \leq t < 0 \\ +A & \text{für } 0 \leq t < T_1/2 \end{cases} \quad \text{mit } u(t+T_1) = u(t)$$

Die Fourierkoeffizienten der CT-Funktion lauten

$$c_{2k+1} = \frac{2A}{\pi} \cdot \frac{1}{2k+1} \quad \text{und } c_{2k} = 0 \text{ mit } k = 0, 1, 2, \dots$$

- a) Wie lauten die ersten 4 Fourierkoeffizienten c_k ? Leiten Sie obige Formel über die Fourierreihe selbst her!
- b) Die Rechteckfunktion wird jetzt mit 4 kHz abgetastet. Bestimmen Sie die Koeffizienten der DFT? Was könnte der Grund für die Abweichungen zu den Koeffizienten der Fourierreihe sein? Simulieren Sie die DFT für $f_S = 100$ kHz und $N = 100$.

3.8. Fourier-Analyse mit Rechteck-Fensterung → M3.8

Gegeben sei zunächst ein periodisches zeitkontinuierliches Signal $s(t) = A \cos(2\pi f_0 t)$. Nun werde mit einem Rechteckfenster $w(t) = 1$ für $0 \leq t \leq T_1$, $w(t) = 0$ sonst, ein aperiodisches Signal $x(t) = w(t)s(t)$ erzeugt.

- a) Gesucht ist ein analytischer Ausdruck für das Spektrum $X(f)$ (kontinuierliche Fouriertransformation). Es empfiehlt sich, zunächst die Spektren $S(f)$ und $W(f)$ formelmäßig anzugeben und dann den Faltungssatz anzuwenden.
- b) Überlegen Sie sich den prinzipiellen Verlauf von $|X(f)|$ und skizzieren Sie diesen im Bereich $-2 \text{ kHz} \leq f \leq 2 \text{ kHz}$. Wählen Sie dazu die Zahlenwerte $f_0 = 1 \text{ kHz}$, $A = 1$, $T_1 = 2T_0 = 2/f_0$ und überprüfen Sie das Ergebnis mit MATLAB/Octave.
- c) Wie können Sie $|X(f)|$ näherungsweise mit Hilfe einer **Radix-2 FFT** bestimmen?
- d) Wie erhalten Sie mittels der FFT das Spektrum von $s(t)$ ohne störende Effekte?

4. FIL: Einfache digitale Filter und FIR-Filterentwurf

Inhalt

Dieses Kapitel beschäftigt sich mit einfachen allgemeinen Filtern und mit der Definition und dem Entwurf digitaler FIR Filter. IIR-Filter werden gesondert in Kap. 11 behandelt.

Spezifikationen: Wie werden Filter spezifiziert, was bedeuten die Spezifikationen?

Filtereigenschaften: Wie unterscheiden sich FIR und IIR - Filter, welchen Einfluss hat die Wahl des Entwurfsverfahrens auf die Filtereigenschaften?

Linearphasige Filter: Wie erkennt man linearphasige Filter im Zeit- und Frequenzbereich, welche Eigenschaften haben sie?

Halbbandfilter: Wie erkennt man Halbbandfilter im Zeit- und Frequenzbereich, welche speziellen Eigenschaften haben sie?

Theorie

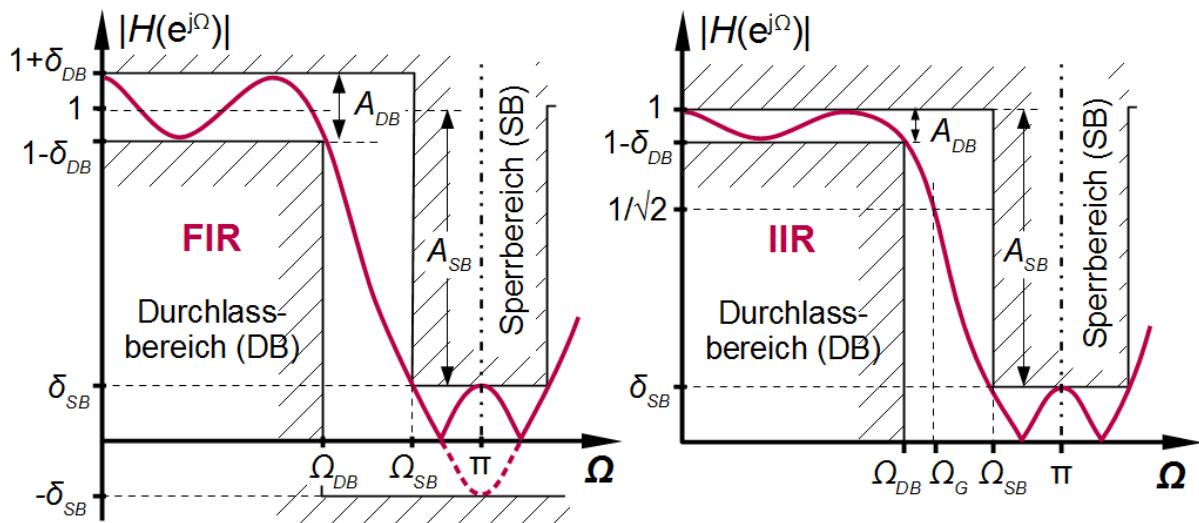


Abb. 4.1.: Filterspezifikationen für FIR und IIR-Filter

Spezifikationen

Abb. 4.1 zeigt die typische Bedeutung der Parameter für FIR- und IIR-Filter: IIR-Filter wurden früher ausschließlich von analogen passiven Filtern abgeleitet (s.u.), deren Durchlassband über die maximale Einfügedämpfung spezifiziert wird und eine maximale Verstärkung von 1 hat. Zu FIR-Filtern gibt es kein analoges Äquivalent, hier wird das DB spezifiziert über die maximale

positive und negative Abweichung (Ripple) δ_{DB} von der nominalen Verstärkung 1. Wenn die nominale Verstärkung $A_0 \neq 1$ ist, muss der lineare Ripple mit A_0 skaliert werden.

Die minimale Amplitudendämpfung im Sperrband in linearer und logarithmischer Schreibweise lässt sich für IIR und FIR-Filter ineinander umrechnen über:

$$A_{SB} = -20 \log_{10} \delta_{SB} \Leftrightarrow \delta_{SB} = 10^{-A_{SB}/20} \quad (4.1)$$

Achtung: A_{SB} bezeichnet die *Dämpfung*, $\delta_{SB} = 0,001 \hat{=} A_{SB} = +60$ dB.

Im Durchlassband muss zwischen IIR und FIR-Filtern unterschieden werden:

$$\text{IIR: } A_{DB} = -20 \log_{10}(1 - \delta_{DB}) \Leftrightarrow \delta_{DB} = 1 - 10^{-A_{DB}/20} \quad (4.2)$$

$$\begin{aligned} \text{FIR: } A_{DB} &= 20 \log_{10}(1 + \delta_{DB}) - 20 \log_{10}(1 - \delta_{DB}) \\ &= 20 \log_{10} \frac{1 + \delta_{DB}}{1 - \delta_{DB}} \Leftrightarrow \delta_{DB} = \frac{10^{A_{DB}/20} - 1}{10^{A_{DB}/20} + 1} \quad (4.3) \\ &\approx 40 \log_{10}(1 + \delta_{DB}) \Leftrightarrow \delta_{DB} \approx 1 - 10^{-A_{DB}/40} \end{aligned}$$

A_{DB} bezeichnet den logarithmierten Ripple und ist ebenfalls positiv.

Achtung: Betrachtet man Leistungen (quadrierte Signale) anstelle von Amplituden, muss wie üblich mit $10 \log_{10} P$ gerechnet werden!

Generell hängt es vom Entwurfverfahren ab, mit welchen Parametern der Frequenzgang spezifiziert wird. So wird z.B. beim Windowed FIR-Entwurfsverfahren die **-6** dB Eckfrequenz als Parameter angegeben, bei den klassischen IIR-Entwürfen dagegen die **-3** dB Eckfrequenz. Bei manchen Entwurfsverfahren werden Filter zu gegebenen Randbedingungen (Eckfrequenzen von DB und SB und Ripple in den Bändern) gesucht.

In diesem Kurs werden die folgenden Symbole für Filterspezifikationen verwendet:

f_S	Abtastfrequenz
f_{DB} (F_{DB})	(Normierte) Eckfrequenz des Durchlassbands (DB)
f_{SB} (F_{SB})	(Normierte) Eckfrequenz des Sperrbands (SB)
δ_{DB}	Ripple (= Welligkeit) im DB
δ_{SB}	Ripple (= Welligkeit) im SB
A_{DB}	Log. Ripple im DB
A_{SB}	Log. Dämpfung im SB

Es wird hier immer von einer Grundverstärkung des Filters im Durchlassband von 1 ($\hat{=} 0$ dB) ausgegangen.

Für einen möglichst entspannten Filterentwurf müssen die Eckfrequenzen von Durchlass- und Stoppband möglichst weit auseinanderliegen (breiter Übergangsbereich)!

Halbbandfilter

Halbbandfilter sind Filter, deren Frequenzgang punktsymmetrisch zu $(\pi/2, H(\Omega = \pi/2))$ verläuft. Es muss also allgemein gelten

$$\begin{aligned} H(\Omega) + H(\pi - \Omega) &= 2H(\pi/2) \\ \text{bzw. } H(\pi/2 - \Omega) + H(\pi/2 + \Omega) &= 2H(\pi/2) \end{aligned}$$

Damit müssen auch die Eckfrequenzen und der Ripple von Pass- und Stopband symmetrisch zu einander sein. Meistens (Filter mit normalisiertem Amplitudengang) ist $H(\pi/2) = 0,5$, aber das ist keine Voraussetzung für ein Halbbandfilter.

Achtung: $H(\Omega)$ ist der komplexe Frequenzgang, nicht der Betragsgang! Da Halbband FIR-Filter aber linearphasig sein müssen, kann man die Rechnung deutlich vereinfachen, indem man die *akausale* Systemfunktion $H_{ak}(z)$ bestimmt: Man eliminiert die lineare Phase aufgrund der mittleren Verzögerung $z^{-N/2}$:

$$\begin{aligned} H_{ak}(z) &= H(z)z^{N/2} \\ \Rightarrow H_{ak}(e^{j\Omega}) &= \underline{H}(e^{j\Omega})e^{j\Omega N/2} \end{aligned}$$

Der so erhaltene Amplitudengang $H_{ak}(e^{j\Omega})$ ist reellwertig, kann aber das Vorzeichen wechseln.

Ein FIR-Filter mit einem solcherart eingeschränkten Amplitudengang hat auch eine spezielle Impulsantwort / Koeffizienten: Halbband-FIR-Filter haben eine symmetrische Impulsantwort (sind also automatisch linearphasig), außer dem mittleren ist jeder zweite Koeffizient Null. Damit das möglich ist, muss die Anzahl L der Koeffizienten ungerade sein mit $L = 4k + 3$ mit $k = 1, 2, \dots$. Die Ordnung $N = L - 1$ muss dementsprechend gerade sein.

Abschätzen der Ordnung

Zur schnellen Abschätzung der Ordnung verschiedener FIR- und IIR-Filterentwürfe können folgende Formeln verwendet werden:

$$N_{FIR} \approx \left\lceil \frac{k_{FIR} f_S}{f_{SB} - f_{DB}} \right\rceil = \left\lceil \frac{k_{FIR}}{F_{SB} - F_{DB}} \right\rceil \text{ mit } k_{FIR} \approx 3 \quad (4.4)$$

$$N_{FIR} \approx \left\lceil \frac{-20 \log_{10} \sqrt{\delta_{SB} \delta_{DB}} - 13}{14,6(F_{SB} - F_{DB})} \right\rceil \quad (4.5)$$

$$N_{IIR} \geq \left\lceil \frac{\sqrt{f_{SB} f_{DB}}}{f_{SB} - f_{DB}} + \frac{a_{SB}}{20 \text{ dB}} \right\rceil \quad (4.6)$$

Die grobe Abschätzung (4.4) ist entnommen aus [Smi99, S. 289] für Windowed Sinc Filter. Der Wert für k_{FIR} hängt leicht vom gewählten Fenster, der gewünschten Sperrdämpfung und dem DB-Ripple ab. Die Näherung passt auch grob für Equiripple-Filter bei einer Sperrdämpfung von $\delta_{SB} = 10^{-4} \hat{=} -80 \text{ dB}$ und einem Ripple von $\delta_{DB} = 0,01 \hat{=} 20 \log_{10}(1 + \delta_{DB}) = 0,086 \text{ dB}$.

(4.5), entwickelt von Kaiser, ist eine Abschätzung für FIR-Filter, die mit Kaiser - Fenster entworfen wurden. Hier gehen auch die gewünschte Sperrdämpfung δ_{SB} und der zulässige Ripple des Durchlassbands δ_{DB} ein [J.F. Kaiser, Nonrecursive Digital Filter Design Using $I_0 - \sinh$ Window function, Proc. IEEE Int. Symp. Circuits and Systems, 20-23, April 1974.].

Bei FIR-Filters kann der Frequenzgang nur über die Nullen beeinflusst werden, deren Anzahl gleich der Ordnung N_{FIR} ist. Die Dämpfung im Sperrbereich wird durch Verteilung von Nullstellen auf dem EK realisiert. Ist die Abtastfrequenz groß im Vergleich zur Eckfrequenz des Sperrbands, werden für das dann breite Sperrband viele Nullstellen benötigt. Die Ordnung ist daher in etwa proportional zur Abtastfrequenz (bei ansonsten unveränderten Parametern).

Der Abschätzung (4.6) liegt zugrunde, dass bei rekursiven Filtern vor allem die Platzierung der Pole (im Durchlassband) den Frequenzgang beeinflusst. Die Ordnung N_{IIR} ist gleich der Anzahl der Pole; im Sperrbereich, weit entfernt von den Polfrequenzen, fällt der Amplitudengang mit 20 dB/Dek. je Pol ab. Bei IIR - Filtern gibt es daher nur eine schwache Abhängigkeit von der Abtastfrequenz. Für elliptische Filter gilt näherungsweise das Gleichheitszeichen, bei anderen Filtertypen (Chebychev, Butterworth, ...) ist N_{IIR} um einen Faktor 1,5 ... 6 größer und hängt stärker von der relativen Bandbreite ab.

Vor allem bei IIR - Filtern, in extremen Fällen auch bei FIR - Filtern, führt eine sehr geringe relative Bandbreite außerdem zu numerischen Problemen. Die Quantisierung der Koeffizienten und der Arithmetik kann daher größere Abweichungen des Frequenzgangs und ein reduziertes SNR zur Folge haben, IIR - Filter können sogar instabil werden. Wenn geringe relative Bandbreiten benötigt werden, ist daher fast immer ein Multiratenfilter (-> Kap. 7) die beste Lösung.

Entwurfsverfahren

Das Ergebnis von Filterentwürfen mit Python / Matlab sind generell Vektoren mit Zähler- und Nennerkoeffizienten (nur bei IIR-Filtern) von $H(z)$ in Polynomschreibweise mit positiven Exponenten. Funktionen zum Filterentwurf sind in Python im Modul `scipy.signal` und in Matlab in der Signal Processing Toolbox enthalten.

Least-Square / Windowed Design / Fourier-Approximation

Dieses Entwurfsverfahren bestimmt die Koeffizienten eines FIR-Filters, indem die unendlich ausgedehnte Impulsantwort eines idealen Filters (rechteckiger Frequenzgang -> si-Zeitfunktion!) approximiert wird, d.h. durch Fensterung auf eine endliche Anzahl Koeffizienten begrenzt wird. Im einfachsten Fall bricht man die Impulsantwort einfach ab (= Rechteckfensterung, „boxcar“). Auf diese Art erhält man für eine gegebene Ordnung den minimalen Fehlerleistung (Least-Square) - das heißt aber nicht minimale Abweichung vom idealen Frequenzgang: Aufgrund der Rechteckfensterung treten Überschwinger an der Bandkante auf! Bei diesem Entwurfsverfahren kann man als einzigen Parameter die -6dB-Grenzfrequenz vorgeben.

Matlab: `firls()`, **Python:** `scipy.signal.firwin()` mit 'boxcar' - Fenster.

In Python findet man alle Filterentwurfsverfahren im Untermodul `scipy.signal`, die Fensterfunktionen unter `scipy.signal.windows` (im Folgenden weggelassen).

Durch eine „weichere“ Fensterfunktion im Zeitbereich erhält man einen Frequenzgang mit geringerem Überschwingen auf Kosten eines breiteren Übergangsbereichs. Vorteilhaft im Gegensatz zum Equirippleverfahren (s.u.) ist, dass die Filterdämpfung bei wachsendem Abstand von der Eckfrequenz je nach Fenster rasch abnimmt, nachteilig ist dass i.a. eine höhere Filterordnung benötigt wird.

Matlab: `firl()`, **Python:** `firwin()`, Default-Fenster ist jeweils 'hamming'.

Fenster in Python: 'bartlett' (= Dreieck), 'blackman', 'blackmanharris', 'boxcar' (= Rechteck), 'flattop', 'gaussian', 'hamming', 'hann' (oder 'hanning'), 'kaiser', 'nuttall', ...

Fenster in Matlab: 'bartlett' = 'triang', 'blackman', 'blackmanharris', 'boxcar' = 'rectwin', 'flattopwin', 'gausswin', 'hamming', 'hann', 'kaiser', 'nuttall', ...

Alle Fensterfunktionen haben als Parameter die Länge des Fenster und die optionale Angabe ob das Fenster symmetrisch (Default) oder periodisch sein soll.

```

1 # Ende der gemeinsamen Import-Anweisungen
2 f_S = 400 # Samplingfrequenz
3 f_DB = 40 # Grenzfrequ. Durchlassband (DB)
4 f_SB = 50 # Grenzfrequenz Sperrband (SB)
5 F_DB = f_DB/(f_S/2) # Frequenzen bezogen
6 F_SB = f_SB/(f_S/2) # auf HALBE Abtastfreq.
7 #
8 A_DB = 0.1 # max. Ripple im DB in dB
9 A_DB_lin = (10**(A_DB/20.0)-1) / \
10   (10**(A_DB/20.0)+1)*2 # und linear
11 A_SB = 60 # min. Dämpfung im SB in dB
12 A_SB_lin = 10**(-A_SB/20.0) # und linear
13 #
14 L = 44 # Manuelle Vorgabe Filterordnung
15 ##### FIR-Filterentwurf #####
16 a = [1] # Nennerpolynom = 1 bei FIR-Filtern
17 === Windowed FIR / Least Square =====
18 F_c = f_DB / (f_S/2) # -6dB - Frequenz
19 b = sig.firwin(L, F_c) # Hamming-Window
20 ## Calculate H(w), w = 0 ... pi, 1024 Pts.
21 [w, H] = sig.freqz(b, a, 1024)
22 # Translate w to physical frequencies:
23 f = w / (2 * pi) * f_S

```

Lst. 4.1: Windowed FIR-Filterentw. mit Python

```

% Samplingfrequenz
f_S = 400; % Samplingfrequenz
f_DB = 40; % Eckfrequenzen DB
f_SB = 50; % und SB
F_DB = f_DB/(f_S/2); % Normierte Frequenzen
F_SB = f_SB/(f_S/2); % bezogen auf f_S / 2
%
A_DB = 0.1; % max. Ripple im DB (log.)
A_DB_lin = (10^(A_DB/20.0)-1)/ ...
  (10^(A_DB/20.0)+1); % und lin.
A_SB = 40; % min. Daempfung im SB in dB
A_SB_lin = 10^(-A_SB/20.0); % und linear
%
L = 44; % Manuelle Vorgabe Filterordnung
%%%%% FIR-Filterentwurf %%%%%%
a = 1;
==== Windowed FIR / Least Square =====
F_c = F_DB / (f_S/2); % -6dB Frequenz
b = fir1(L, F_c, hamming(L+1));
% Calculate H(w), w = 0 ... pi, 1024 Pts.
[H, w] = freqz(b, a, 1024);
% Translate w to physical frequencies:
f = w / (2 * pi) * f_S;

```

Lst. 4.2: Windowed FIR-Filterentwurf mit Matlab

Frequency Sampling

Wenn man von Filterspezifikationen in der Frequenzebene ausgeht, ist Frequency Sampling vermutlich das am einfachsten verständliche und direkteste Filterentwurfverfahren: Der Wunschkoeffizienten wird in gleichmäßigen Abständen (äquidistant) abgetastet, durch inverse Fouriertransformation erhält man sofort die Impulsantwort, die den Filterkoeffizienten des FIR-Filters entsprechen. Auf diese Art können nahezu beliebige Verläufe des Frequenzgangs erzielt werden (auch Multiband, Differenzierer, Hilbert-Filter, ...). Allerdings wird der gewünschte Frequenzgang exakt nur an den Abtastpunkten erreicht - dazwischen können z.T. wilde Überschwingen des Frequenzverlaufs auftreten. Daher muss man unbedingt das Ergebnis des Entwurfs überprüfen, indem man eine „DTFT“ (d.h. eine DFT mit ausreichendem Zero Padding, die zwischen den Frequency Sampling Punkten interpoliert) der Filterkoeffizienten berechnet und mit dem Wunschkoeffizienten vergleicht.

Python: firwin2(), Matlab: fir2():

```

1 #==== Frequency Sampling =====
2 b = sig.firwin2(L, [0, F_DB, F_SB, 1],
3                  [1, 1, 0, 0])

```

Lst. 4.3: Frequency Sampling FIR-Filterentwurf mit Python

```

%==== Frequency Sampling =====
b = fir2(L, [0, F_DB, F_SB, 1], ...
          [1, 1, 0, 0], hamming(L+1));

```

Lst. 4.4: Frequency Sampling FIR-Filterentwurf mit Matlab

Equiripple / Remez / Parks-McClellan / Minimax

Dieses Entwurfsverfahren minimiert den maximalen Fehler (-> **Minimax**) im Durchlass- und Sperrband und schafft so einen gleichmäßigen Ripple (-> **Equiripple**), der die vorgegebenen Spezifikationen voll ausschöpft. Benutzt wird hierfür der Algorithmus von **Parks-McClellan** und das Austauschverfahren von **Remez** - das Verfahren ist in der Literatur daher unter allen vier Namen zu finden. Hier werden im einfachsten Fall die Eckfrequenzen von DB und SB mit den zulässigen Abweichungen (Ripple) vorgegeben. Im Sperrband entspricht der Ripple der minimalen Dämpfung. Durch Angabe von Frequenz / Amplitudenpaaren können aber auch hier relativ beliebige Frequenzgänge vorgegeben werden. Optional können hier die DB und SB Spezifikationen unterschiedlich gewichtet werden.

Matlab: firpm(), Python: remez().

Achtung: Bei Python sind die relativen Frequenzen hier auf f_S , nicht auf $f_S/2$ bezogen (daher wird hier die Abtastfrequenz mit $\text{Hz} = 2$ spezifiziert). Wie bei den zuvor besprochenen Verfahren, kann die Filterordnung manuell angegeben werden. Es gibt aber auch Algorithmen, mit denen man (mehr oder weniger korrekt) die Filterordnung und die Gewichtungen für DB und SB abschätzen kann. Mit den so gewonnenen Angaben startet man das eigentliche Filterdesign. Bei Python ist die Funktion `remezord()` nicht in Scipy enthalten, Sie finden Sie in der dsp - Library auf Moodle.

```

1 #==== REMEZ / Parks-McClellan / Equiripple
2 W_DB = 1; W_SB = 1 # manuelle Ordnung:
3 b = sig.remez(L, [0, F_DB, F_SB, 1],
4                [0, 1], [W_DB, W_SB], Hz = 2)
5 # minimale Ordnung:
6 (L_min,F,A,W) = dsp.remezord([F_DB, F_SB],
7      [1, 0], [A_DB_lin, A_SB_lin], Hz = 2)
8 b = sig.remez(L_min, F, A, W )

```

Lst. 4.5: Equiripple FIR-Filterentw. mit Python

```

%==== REMEZ / Parks-McClellan / Equiripple
W_DB = 1; W_SB = 4; % manuelle Ordnung:
b = firpm(L, [0, F_DB, F_SB, 1],...
           [1, 1, 0, 0], [A_DB, A_SB], [W_DB, W_SB]);
% minimale Ordnung:
[L_min, F , A, W] = firpmord([F_DB, F_SB],...
      [1, 0], [A_DB_lin, A_SB_lin], 2);
b = firpm(L_min, F, A, W);

```

Lst. 4.6: Equiripple FIR-Filterentwurf mit Matlab

Python: Befehle und Programme zu Kap. 4

Funktion	Python	Matlab	Beschreibung
Filterung	<code>h,t = sig.lfilter(b,a)</code>	<code>[h,t] = filter(b,a)</code>	Lineare Filterung mit dem durch <code>b</code> , <code>a</code> definierten System.
FIR-Filter			
Minimax-Filter	<code>sig.remez()</code>	<code>firpm()</code>	Parks-McClellan - FIR Filter

Tab. 4.1.: Spezielle Befehle zu Kap. 4

Anmerkungen zu Tab. 4.1:

Allgemein Siehe auch die Anmerkungen zu Tab. 2.1

Filename	Beschreibung
FIL_Intro	Implementierung verschiedener Filterentwurfsverfahren in Matlab und Python (siehe oben)
FIL_BPSK	Bit Error Rate (BER) bei Störung eines BPSK-kodierten Signals durch additives weißes Rauschen. Es wird gezeigt, welchen Einfluss Grenzfrequenz und Phasenverzerrungen von verschiedenen Tiefpassen auf die BER haben.
FIL-Linphase_Filter	Einfluss von einfachen und doppelten Nullstellen auf den Amplitudengang (zu Aufgabe 4.2)
FIL-Linphase-_Filter_annotation	wie voriges Programm, aber mit zusätzlichen Textausgaben im Plot
FIL-Halfband_design	Entwurf von Halbbandfiltern mit unterschiedlichen Methoden, Vergleich im Zeit- und Frequenzbereich (Aufgabe 4.4)

Tab. 4.2.: Simulationsfiles zu Kap. 4

4.0. * Filterentwurf für Sensorsignal

Brummstörungen bei 50 Hz auf einem Sensorsignal sollen mit einem digitalen Filter um mindestens 60 dB unterdrückt werden. Das Sensorsignal wird mit einem ADC mit $f_S = 400$ Hz abgetastet und hat eine Nutzbandbreite von 0 ... 40 Hz. Das Filter soll einen Amplitudenfehler von maximal ± 1 dB verursachen.

Entwerfen Sie verschiedene FIR mit dem grafischen Entwurfstool pyFDA oder aufbauend auf Listing 4.1 bzw. 4.2 und versuchen Sie, die gestellten Anforderungen zu erfüllen. Die Listings zeigen nur einen Teil der Files, nehmen Sie daher die Files in elektronischer Form und passen Sie sie an die Aufgabenstellung an. Kommentieren Sie für die Simulationen jeweils alle Entwurfsverfahren bis auf eins aus.

Vergleichen Sie die Systeme bezüglich Resourcenverbrauch (Ordnung) und schauen Sie sich die unterschiedlichen Pol/Nullstellenpläne und Betrags- und Phasenverlauf an! Welche Filter haben eine gute Unterdrückung von Störungen bei höheren Frequenzen (z.B. 150 Hz)?

Zu dieser Aufgabe gibt es keine Musterlösung, die Ergebnisse werden in der Vorlesung besprochen. Aufgabe 11.0 behandelt das gleiche Thema mit rekursiven Filtern.

4.1. * Filterspezifikationen → M4.1

- a) Ein Signal soll mit einem FIR- und einem IIR-Filter so gefiltert werden, dass die Ausgangsamplitude im Durchlassbereich um maximal $\pm 10\%$ variiert. Welche Spezifikationen δ_{DB} , A_{DB} muss man dazu in einem Filterentwurfstool angeben?
- b) Ein IIR- und ein FIR-Filter sind spezifiziert mit $A_{DB} = 2$ dB und $A_{SB} = 20$ dB. Die Grenzfrequenzen betragen jeweils $F_{DB} = 0,1$ und $F_{SB} = 0,13$. Wie groß ist jeweils der Ripple im Durchlassband und Sperrband δ_{DB} bzw. δ_{SB} ? Skizzieren Sie den prinzipiellen Verlauf beider Filter für ein Equiripple- und ein Chebychev (Typ 2, Ripple im Sperrband)-Design im linearen und logarithmischen Maßstab!
- c) Die Filter sollen jetzt einem (als ideal angenommenen) Audioverstärker mit einer maximalen Sinusleistung von $P_{nenn} = 80$ W an 8Ω vorgeschaltet werden. In welchem Bereich können die Ausgangsleistung und Ausgangsamplitude im Durchlassband variieren? Welche maximale Ausgangsleistung und -amplitude kann man im Sperrband erwarten?
- d) Einem Sensorsignal im Frequenzbereich $f_{sig} = 0 \dots 20$ Hz mit Amplitude $\hat{u}_{sig} = 1 \dots 50$ mV ist ein Brummen mit $f_{br} = 50$ Hz und $\hat{u}_{br} = 100$ mV überlagert. Die Abtastfrequenz beträgt $f_S = 1$ kHz. Zur Dämpfung der Störkomponente soll jetzt ein Tiefpassfilter entworfen werden.

Der Übergangsbereich zwischen Durchlass- und Sperrband des Filters soll möglichst breit sein. Wie sind die (normierten) Eckfrequenzen für Durchlass- und Sperrband, f_{DB} , f_{SB} , F_{DB} , F_{SB} , Ω_{DB} und Ω_{SB} ?

Bestimmen Sie die logarithmischen Spezifikationen für das Filter, wenn die Brummamplitude am Ausgang des Filters höchstens ein Tausendstel der Signalamplitude betragen darf und die Signalamplitude im Durchlassband maximal um 1% verfälscht werden darf.

Nutzen Sie zur Lösung der Aufgabe die Abbildungen und Formeln auf S. 45f.

4.2. * Amplitudengang linearphasiger Filter → M4.2

Gegeben ist die Impulsantwort $h[n] = \{-1/8, 0, 5/8, 1, 5/8, 0, -1/8\}$ des FIR-Filters F. Die nachstehenden Bilder Abb. 4.2(a) und Abb. 4.2(b) zeigen den Amplitudengang $|H(e^{j2\pi F})|$ im linearem und logarithmischem Maßstab.

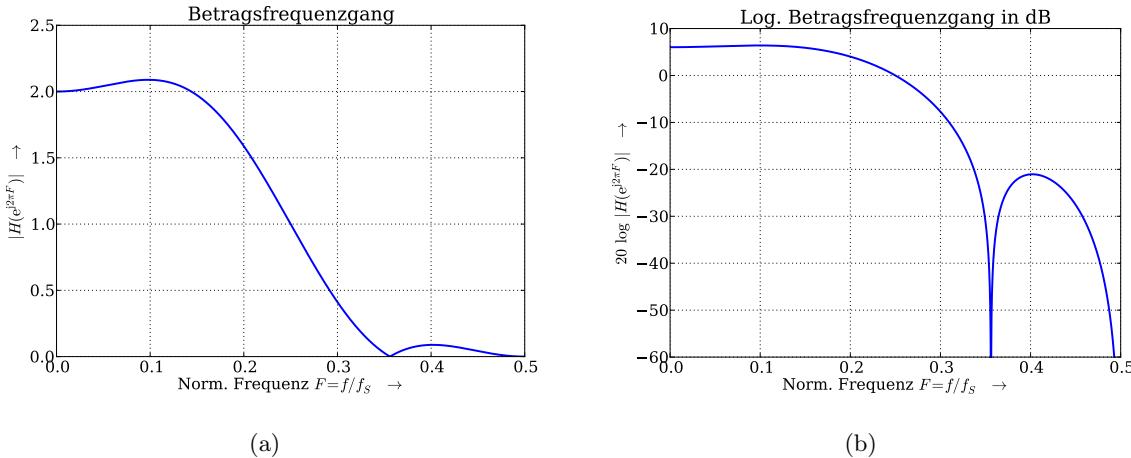


Abb. 4.2.: Amplitudengang des Filters F im (a) linearen und (b) logarithmischen Maßstab

- Ist das Filter linearphasig? Ist es ein Halbbandfilter? Begründen Sie Ihre Aussage!
- Beweisen Sie, dass $H(z)$ bei $f_S/2$ eine doppelte Nullstelle hat.
- Schätzen Sie die Lage von zwei weiteren Nullstellen auf dem Einheitskreis der z -Ebene ab.
- Wie viele Nullstellen sind noch übrig? Wo könnten sie liegen und was bewirken sie? Plotten Sie mit Hilfe eines Matlab / Python-Programms die Betragsfunktion für ein System ohne diese Nullstellen.
- Skizzieren Sie das ganze Pol/Nullstellendiagramm in der z -Ebene. Schreiben Sie ein Python / Matlab - Programm, mit dem Sie das P/N - Diagramm darstellen können

4.3. * Maximal-, minimal- und linearphasige Filter → M4.3

In dieser Aufgabe wird untersucht, wie sich die Platzierung einer Nullstelle auf Minimal- / Maximal- und Linearphasigkeit eines Filters erster bzw. zweiter Ordnung auswirkt. Es wird gezeigt, wie ein komplexwertiges Filter implementiert werden kann.

- Bestimmen Sie allgemein die Systemfunktion $H_1(z)$, Impulsantwort $h_1[n]$, komplexen Frequenzgang $H_1(e^{j\Omega})$, Betragsgang $|H_1(e^{j\Omega})|$ und Phasengang $\angle H_1(e^{j\Omega})$ eines Filters F_1 mit der Nullstelle $z_{0,1} = r_1 e^{j\Omega_1}$ und einem Pol bei $z = 0$.**

Ist das Filter F_1 kausal, reellwertig, minimalphasig, linearphasig?

- Skizzieren Sie $|H_1(e^{j\Omega})|$ und $\angle H_1(e^{j\Omega})$ für die Zahlenwerte $\Omega_1 = \pi/3$ und $r_1 = 2$ im Bereich $0 \leq \Omega \leq 2\pi$ (d.h. $0 \leq f \leq f_S$).**

Berechnen Sie hierzu zunächst Hilfswerte für $|H_1(e^{j\Omega})|$ und $\angle H_1(e^{j\Omega})$ bei den normierten Frequenzen $\Omega = 0$, $\Omega = \pi/3$ und $\Omega = \pi$.

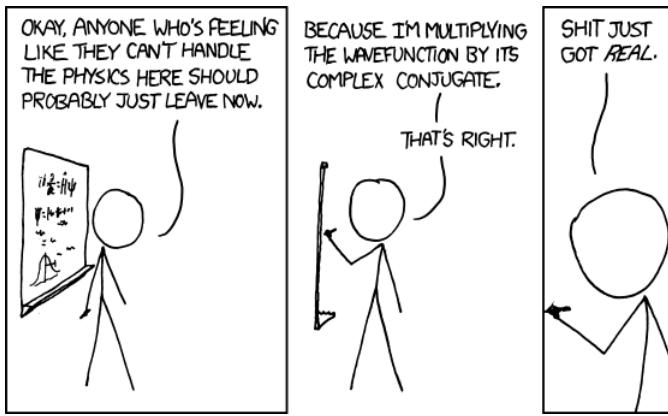


Abb. 4.3.: Konjugiert und komplex [<http://xkcd.com/849/>]

- c) **Leiten Sie aus F_1 durch Spiegeln der Nullstelle $z_{0,1}$ am EK das Filter F_2 ab.**
 Nehmen Sie weiterhin an, dass der Radius der ursprünglichen Nullstelle $r_1 > 1$. Ist das Filter F_2 minimalphasig?

Skalieren Sie $H_2(z)$ so, dass F_1 und F_2 bei $\Omega = \Omega_1$ den gleichen Betrag der Verstärkung haben, $|H_1(e^{j\Omega_1})| = |H_2(e^{j\Omega_1})|$.

Können Sie zeigen, dass $|H_1(e^{j\Omega})| = |H_2(e^{j\Omega})|$ bei allen Frequenzen gilt?

- d) **Skizzieren Sie $|H_2(e^{j\Omega})|$ und $\angle H_2(e^{j\Omega})$** für die Zahlenwerte aus Unterpunkt b), berechnen Sie auch hier zunächst Hilfswerte.
- e) **Konstruieren Sie das Filter F mit $H(z) = H_1(z)H_2(z)$.** Untersuchen Sie, ob F linearphasig ist. Berechnen Sie die den Phasengang $\angle H(e^{j\Omega})$ und die Gruppenlaufzeit $\tau_{g,H}(e^{j\Omega})$ dieses Filters.
- f) **Zeichnen Sie das Blockschaltbild** für eine mögliche Realisierung des Filters F. Das Eingangssignal $x[n]$ sei reell. Wie würden Sie dieses Filter in Hardware implementieren?

4.4. * FIR Halbbandfilter → M4.4

- a) Ausgehend von folgenden **Spezifikationen** soll ein Halbbandfilter entworfen werden:

Abtastrate: $f_S = 24$ kHz

Eckfrequenz des DB: $f_{DB} = 1$ kHz

Ripple des DB: $A_{DB} = 0,1$ dB

Eckfrequenz des SB: $f_{SB} = 10$ kHz

Dämpfung des SB: $A_{SB} = 60$ dB

Passen Sie die obigen Spezifikationen so an, dass Sie ein Halbbandfilter erhalten. Die ursprünglichen Spezifikationen dürfen dabei nicht verletzt, aber verschärft werden. Entwerfen Sie dann das Filter mit Python / Matlab! [Fehlt noch!]

- b) Bestimmen Sie den **Amplitudengang aus der Impulsantwort** des folgenden Filters und zeigen Sie, dass es einen zu $F = 0,25$ symmetrischen Betragsgang hat und somit ein Halbbandfilter ist:

$$h[n] = \{0,1; 0; 0,2; 0,3; 0,2; 0; 0,1\}$$

Welchen Wert hat der Betragsgang bei $F = 0$, $F = 0,25$ und $F = 0,5$? Wie lässt sich die Dämpfung der Frequenzkomponente bei $f_S/2$ verbessern?

Ermitteln Sie hierzu zunächst $H(z)$ und daraus $H(e^{j\Omega})$. Formen Sie $H(e^{j\Omega})$ dann so um, dass man die gefragte Symmetrie erkennen kann. Die Rechnung ist einfacher, wenn Sie die akausale Systemfunktion $H_{ak}(z)$ ermitteln!

- c) Wie müssen Sie die Koeffizienten skalieren, damit gilt $|H(F = 0,25)| = 0,5$?
- d) Zeigen Sie, dass **jeder zweite Koeffizient eines Halbbandfilters Null ist** (außer dem mittleren). Gehen Sie hierzu von der Übertragungsfunktion eines idealen, akausalen TP-Filters aus, mit $\Omega_g = \pi/2$ mit Hilfe der inversen DTFT in den Zeitbereich. Nutzen Sie dazu folgende Formel:

$$X(\Omega) = \begin{cases} 1 & \text{für } 0 \leq |\Omega| < \Omega_g \\ 0 & \text{sonst} \end{cases}$$

$$\bullet \rightsquigarrow x[n] = \frac{1}{2\pi} \int_{-\Omega_g}^{\Omega_g} e^{j\Omega n} = \frac{\Omega_g}{\pi} \frac{\sin(\Omega_g n)}{\Omega_g n} = \frac{\Omega_g}{\pi} \operatorname{si}(\Omega_g n) \quad \text{für } n = 0, 1, 2, \dots$$

- e) Welche Art von Filter erhalten Sie, wenn der **mittlere Koeffizient ebenfalls Null** ist?
- f) Konstruieren Sie aus dem (verbesserten) Filter aus b) ein **Halbband-Hochpassfilter**. Hierzu können Sie die TP -> HP - Transformation verwenden oder überlegen, wie Sie die Koeffizienten variieren müssen.
- g) Es gibt auch **IIR-Halbbandfilter**¹. Überlegen Sie, welche der Ihnen bekannten IIR-Entwurfsverfahren geeignet ist und wie das P/N-Diagramm eines solchen Filters aussehen könnte!

4.5. Filterentwurf → M4.5

Entwerfen Sie mit Python / Matlab ein FIR-Filter nach der Equiripple-Methode und eins nach der Kaiser-Fenster-Methode mit folgenden Spezifikationen:

Das Durchlassband verläuft im Bereich $F = 0,05 \dots 0,5$ und das Stopband zwischen $F = 0 \dots 0,025$ (jeweils normalisiert auf f_S). Die Dämpfung im Sperrband soll mindestens 50 dB betragen, der Ripple im Nutzband höchstens 0,1 dB.

Noch keine Musterlösung!

¹Auch wenn die Symmetriebedingung hier etwas anders definiert ist und natürlich keine Linearphasigkeit vorausgesetzt wird.

4.6. * Filtertransformationen → M4.6

Ein FIR-Filter F1 mit der Impulsantwort $h_1[n] = \{1; 7/4; 1/2; -1/4\}$ hat den Betragsfrequenzgang in Abb. 4.4.

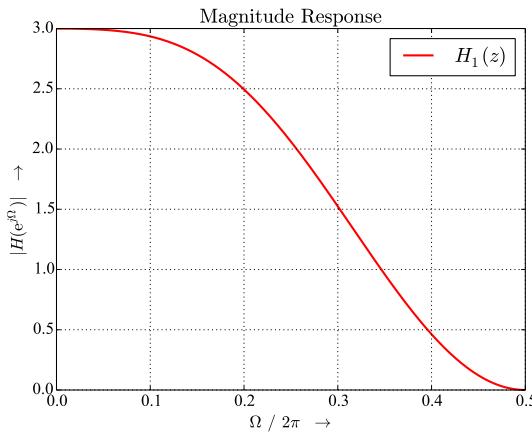


Abb. 4.4.: Betragsfrequenzgang zu Aufgabe 4.6

- a) Berechnen Sie $H_1(f = 0)$ und $H_1(f = f_S/2)$ und beweisen Sie, dass $H_1(f)$ zwei Nullstellen bei $f = f_S/2$ hat. Gibt es weitere Nullstellen? Wieviele, welche?
- b) Aus F1 wird ein neues Filter F2 abgeleitet, indem jedes Verzögerungsglied verdoppelt wird. Geben Sie $h_2[n]$ und $H_2(z)$ an. Wie wirkt sich die Transformation $z^{-1} \rightarrow z^{-2}$ auf den Betragsfrequenzgang aus? Skizzieren Sie den Betragsfrequenzgang von F2.
- c) Erzeugen Sie mittels TP-HP-Transformation $z \rightarrow -z$ aus Filter F1 ein Filter F3. Geben Sie $h_3[n]$ an und skizzieren Sie den zugehörigen Betragsfrequenzgang.

4.7. * Filterimplementierung auf FPGA → M4.7

Ein FIR-Filter 20. Ordnung soll auf einem FPGA mit MAC-Cores implementiert werden, die jeweils $2 \cdot 10^8$ Multiplikationen je Sekunde durchführen können.

- a) Welcher maximale Datendurchsatz in Samples/s lässt sich für ein beliebiges FIR-Filter 20. Ordnung erzielen, wenn nur ein MAC-Core für das Filter eingesetzt werden darf?
- b) Welcher Durchsatz lässt sich bei einem linearphasigen FIR-Filter unter Ausnutzung der Symmetrie erzielen? Welcher Durchsatz ist bei einem Halbbandfilter möglich?
- c) Auf welchen Wert lässt sich der Durchsatz steigern, wenn zwei MAC-Cores verwendet werden dürfen?

4.8. * Verständnisfragen zu Filtern → M4.8

Kreuzen Sie in Tab. 4.3 an, welche Aussage auf welche Filter zutrifft. Kann je nach Systementwurf „Ja“ oder „Nein“ zutreffen, kreuzen Sie beides an.

Aussage	FIR-Filter		IIR-Filter	
	ja	nein	ja	nein
Rekursiv:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Endliche Impulsantwort:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Instabil:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Linearphasig:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Minimalphasig:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Minimalphasig und linearphasig:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Ursprungs:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Einheitskreises:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nullstellen außerhalb des Ursprungs:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nullstellen außerhalb des Einheitskreises:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Halbbandfilter:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Konstante Gruppenlaufzeit:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transponierte Form:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Entwurf über bilineare Transformation:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transversale Struktur:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Abgeleitet von analogen Butterworth-Filtern:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Tab. 4.3.: Verständnisfragen zu Filtern (Aufgabe 4.8)

5. FIX: Wortlängeneffekte und Fixpoint-Systeme im Zeitbereich

Inhalt

Dieses Kapitel beschäftigt sich mit der Repräsentation von Zahlen im Binärsystem und den Auswirkungen einer begrenzten Amplitudenauflösung (Quantisierung) im *Zeitbereich*. Fixpoint Systeme und Fixpoint-Arithmetik lassen sich mit deutlich geringeren Anforderungen an die Hardware implementieren als Floating Point Systeme. Nachteilig ist der deutlich geringere Dynamikbereich bei gleicher Wortlänge und die schlechtere Portierbarkeit.

Grundlagen

Binäre Zahlendarstellung: Integer- und rationale Zahlen mit und ohne Vorzeichen können auch im Binärsystem dargestellt werden. Bei der Umsetzung auf Hardware schränkt die begrenzte Wortlänge den maximal darstellbaren Wertebereich (Full Scale Range, *FSR*) und die maximale Amplitude eines symmetrischen Sinussignals ein.

Quantisierung: Aufgrund der begrenzten Wortlänge können die meisten rationalen Zahlen nicht fehlerfrei in Fixpointsystemen dargestellt werden.

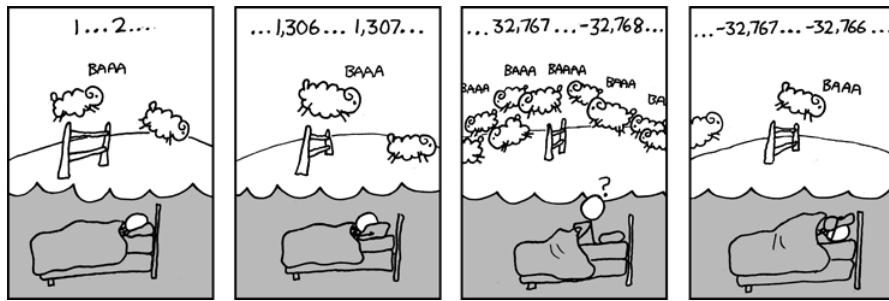
Neue Themen

Filter mit quantisierten Koeffizienten: Bei Fixpoint-Systemen hat die unvermeidliche Quantisierung der Koeffizienten großen Einfluss auf die Filterperformance; Eckfrequenzen und Welligkeit werden verändert, IIR-Filter können sogar instabil werden. Durch Wahl robuster Filtertopologien wie kaskadierter Filter anstatt Direktformfilter lassen sich diese Effekte minimieren.

Fixpoint-Arithmetik: Addition und Multiplikation in Fixpointfiltern lassen die Wortlänge anwachsen. Das ist bei der Auslegung des Systems zu beachten, um Überläufe zu vermeiden.

Requantisierung: In Fixpointfiltern muss daher fast immer requantisiert werden, um das Wortlängenwachstum zu begrenzen.

Grenzzyklen: Die Rückführung des Requantisierungsfehlers bei rekursiven Systemen kann zu Schwingungen führen.

Abb. 5.1.: Schlauflose Programmierer [<http://xkcd.com/571/>]

Theorie

Quantisierungseffekte und Wortlängenwachstum

Bei der Fixed-Point Zahlendarstellung kann man die quantisierte (d.h. auch nicht-lineare) Natur im Gegensatz zum Floating Point Zahlenformat¹ nicht mehr vernachlässigen.

Wertebereich: Der maximal darstellbare Zahlenbereich ist durch Wortlänge W vorgegeben. Achtung: Bei mehrstufigen Filtern können Resonanzüberhöhungen zu Überläufen im Filter führen!

Koeffizientenquantisierung: Die Quantisierung der Filterkoeffizienten ändert die Übertragungsfunktion und kann bei rekursiven Filtern sogar zu Instabilitäten führen. Durch Abschätzungen und Simulationen kann die Anzahl der Fractionalbits so angepasst werden, dass das gewünschte Toleranzschema des Filters erreicht wird.

Requantisierung: Da bei Multiplikationen die Anzahl der Nachkommastellen WF wächst und bei Additionen die Anzahl der Vorkommastellen WI , müssen vor allem bei rekursiven Systemen die Wortlängen im System durch Requantisierung angepasst werden.

Wortlängenwachstum: Bei der Addition von 2 Fixpointzahlen muss man für beliebige Werte ein zusätzliches Integerbit vorsehen, $\Delta WI = 1$. Addiert man N Zahlen benötigt man im worst case

$$\Delta WI = \lceil \log_2 N \rceil$$

zusätzliche Integerbits.

Tipp: Sie können den Logarithmus zur Basis 2 mit dem Taschenrechner berechnen über $\log_2(x) = \log_{10}(x) / \log_{10}(2)$.

Koeffizientenfläche wird die Summe des Betrags aller Koeffizienten eines FIR-Filters genannt:

$$A_b = \sum |b_i| \Rightarrow \Delta WI = \lceil \log_2 A_b \rceil$$

Mit Hilfe der Koeffizientenfläche kann man das Wortlängenwachstum präzise vorhersagen und damit die benötigte Wortbreite des Akkumulators. Kleinere Koeffizienten führen zu kleineren Teilprodukten und damit zu geringerem „bit growth“ des Akkumulators.

¹Floating-Point Zahlen sind zwar auch quantisiert, haben aber eine so große Dynamik, dass Quantisierungseffekte nur selten wahrnehmbar sind.

Requantisierungsmethoden

In Fixpointfiltern muss die Wortlänge zwischendurch fast immer requantisiert werden, um das Wortlängenwachstum zu begrenzen. Hierfür gibt es verschiedene Methoden:

Abschneiden (Truncation) benötigt den geringsten Hardwareaufwand, führt aber zu einem systematischen DC-Fehler (DC-Bias), außerdem kann der Betrag durch Abschneiden größer werden und so zu Grenzzyklen führen.

Runden vermeidet den systematischen DC-Fehler, aber auch durch Runden kann der Betrag ansteigen.

Wertschneiden garantiert, dass der Betrag ist immer kleiner ist als der ursprüngliche Wert. Nachteilig ist der größere Hardware-Aufwand und der größere Quantisierungsfehler.

Kleine und große Grenzzyklen

Bei Requantisierung innerhalb von rekursiven Filtern (und nur bei diesen!) kann die Rückführung des Quantisierungsfehlers Schwingungen oder DC-Offset hervorrufen, die auch anhalten wenn das Eingangssignal wieder auf Null zurückgegangen ist.

Kleine Grenzzyklen: Requantisierung durch Wertschneiden oder Runden kann den *Betrag* vergrößern und so wie eine zusätzliche Verstärkung wirken. Bei rekursiven Systemen kann dies zu Oszillationen der unteren Bits oder zu einem kleinen DC-Offset führen. Diesen Effekt nennt man *kleine Grenzzyklen* (engl.: *small scale zero-input limit cycles*). -> Wertschneiden verringert immer den Betrag und vermeidet so diesen Effekt.

Überlaufeffekte und große Grenzzyklen: Wenn bei Additionen der Wertebereich überschritten wird, springt bei Zweierkomplementarithmetik das Vorzeichen. Dadurch können rückgekoppelte Systeme zu Schwingungen mit voller Amplitude angeregt werden (*große Grenzzyklen*, engl.: *large scale zero-input limit cycles*). Dies wird durch Sättigungslogik (Begrenzung auf minimal bzw. maximal darstellbaren Wert) vermieden.

Python: Befehle und Programme zu Kap. 5

In Python und Matlab ist der Umgang mit binären Fixed-Point Objekten eher exotisch und muss nachgerüstet werden. In Matlab gibt es hierzu die Fixed-Point Toolbox, in Python habe ich selbst die Klasse **Fixed()** geschrieben.

Listing 5.1 zeigt, wie man die Python-Klasse **Fixed** anwendet: Zunächst wird ein Dictionary mit den Quantisierungsoptionen definiert (Anzahl der Integer- und Fractionalbits, Quantisierungs-methode und das Verhalten bei Überlauf).

```

1 # Ende der gemeinsamen Import-Anweisungen
2 import dsp_fpga_fix_lib as fx
3 import time
4 N = 10000; f_a = 1
5 t = linspace(0, 1, N, endpoint=False)
6 a = 1.1 * sin(2 * pi * f_a * t)
7 x = linspace(-2, 2, N, endpoint=False)
8 #
9 #q_obj = (0, 4, 'round', 'sat') # try 'round' ; 'sat'
10 q_obj = {'QI':0, 'QF': 4, 'quant':'fix', 'ovfl': 'sat'} # try 'round' ; 'sat'
```

```

11 #q_obj = {'OVFL':'sat'}
12 fx_a = fx.Fixed(q_obj)
13 fx_x = fx.Fixed(q_obj)
14
15 t_cpu = time.clock()
16
17 aq = fx_a.fix(a) # quantize a
18 xq = fx_x.fix(x) # quantize x
19 print('Anzahl der Überläufe = ', fx_x.N_over)
20
21 print ('Total CPU time: %.5g ms' %((time.clock()-t_cpu)*1000))
22
23 #
24 figure(1)
25 title('Quantisiertes Sinussignal')
26 plot(t, a, label = r'$a(t)$')
27 plt.step(t, aq, where = 'post', label = r'$a_Q(t)$')
28 plot(t, a-aq, label = r'$a(t) - a_Q(t)$')
29 plt.legend(fontsize = 14)
30 xlabel(r'$t \rightarrow$'); ylabel(r'$a \rightarrow$')
31 #
32 figure(2)
33 title('Quantisierungskennlinie')
34 plt.step(x,xq, where = 'post')
35 xlabel(r'$x \rightarrow$'); ylabel(r'$x_Q \rightarrow$')
36 plt.show()

```

Lst. 5.1: Quantisierung eines Sinussignals und Quantisierungskennlinie mit Python
(**FIX/FIX_intro_py.py**)

Beim Instanziieren der Klasse wird das Dictionary als Parameter übergeben, danach wird z.B. der Methode `fix()` der zu quantisierende Wert oder Vektor übergeben. Zurückgegeben werden der quantisierte Wert oder Vektor. Die Anzahl der aufgetretenen Überläufe ist im Attribut `N_over` gespeichert.

Die wichtigsten Quantisierungsmethoden sind dabei '`floor`' (Wertschneiden), '`round`' (Runden) und '`fix`' (Betragsschneiden).

Optionen für das Verhalten beim Verlassen des Wertebereichs sind '`none`' (keine Überläufe; Anzahl der Integerbits wird ignoriert), '`wrap`' (Überläufe mit Zweierkomplementverhalten - „nach ganz groß kommt ganz klein“) und '`sat`' (Begrenzung auf Minimal- bzw. Maximalwert).

Filename	Beschreibung
FIX_intro	Quantisierungskennlinie und Wellenform bei verschiedenen Arten der Quantisierung und Sättigung (Listing 5.1)
FIX_limit_cycles	Grenzzyklen in IIR-Filtern
FIX_pyaudio_basics	Grundgerüst für Echtzeit-Audiosignalverarbeitung in Python
FIX_pyaudio-quantization	Audiodemonstration von Quantisierungs- und Sättigungseffekten
FIX_pyaudio-limit_cycles	Audiodemonstration von Grenzzyklen in rekursiven Filtern (Code ist noch zu langsam für Echtzeit)

Tab. 5.1.: Python-Files zu Kap. 5

5.1. * Analog-Digital-Wandlung eines Drucksensors → M5.1

Der Beschleunigungssensor ADXL 327 kann Beschleunigungen im Bereich $a = \pm 2 \text{ g}$ messen und hat eine Empfindlichkeit von $S_{Sens} = 400 \text{ mV/g}$. Bei $a = 0$ ist die Ausgangsspannung $U_{Sens} = 1,5 \text{ V}$. Die Ausgangsspannung des Sensors wird digitalisiert mit einem nachgeschalteten 8 Bit ADC mit Eingangsspannungsbereich (Full Scale Range) $FSR = 0 \dots 5 \text{ V}$.

- a) Welcher Änderung der Eingangsspannung q_U (= Auflösung) in V entspricht ein LSB des Wandlers? Mit welcher Unsicherheit repräsentiert das Digitalwort einen analogen Wert?
- b) Welche Auflösung bezogen auf die Messgröße q_a in g erhält man, wenn der Wandler direkt an den Sensor angeschlossen wird?
- c) Bei $a = 0$ soll der digitale Ausgangswert des ADCs **0000 0000** sein (im ZK-Format). Wie muss das Ausgangswort dazu digital korrigiert werden?
- d) Welchen digitalen Ausgangswert (nach der Korrektur) liefert der ADC bei einer Beschleunigung von $\pm 1,2 \text{ g}$? Wie werden diese Werte binär dargestellt, welchen Werten entsprechen sie im *normierten Zweierkomplement-Fraktional-Format* (nZKF-Format, Q0.7)?
- e) Welche maximale Amplitude für analoges Signal bzw. Beschleunigung ist digital darstellbar?
- f) Welcher Beschleunigung entspricht das binäre Wort **1000 0001**?
- g) Das Sensorsignal soll jetzt analog vorverstärkt werden, damit es optimal an den Eingangsspannungsbereich des Wandlers angepasst ist. Welche Auflösung q_a in g erhalten Sie dadurch?
- h) Wieviele Codeschritte des Wandlers überstreichen Sie jeweils mit dem Spannungsbereich des Sensors bei beiden Lösungen? Welcher äquivalenten Wortlänge (Effective Number of Bits, ENOB) entspricht das jeweils?
- i) Zur digitalen Filterung des Sensorsignals wird ein MA-Filter der Ordnung 15 ohne Skalierung verwendet. Welche Wortlänge und welches Zahlenformat ist am Ausgang des Filters erforderlich, damit keine Überläufe auftreten? Bei welcher normierten Frequenz liegt die erste Nullstelle des Betragsgangs?

5.2. * Moving Average Filter mit endlicher Wortbreite → M5.2

Ein Moving Average Filter mit der Ordnung 8 (= 8 Register) soll mit einer Eingangswortbreite von 10 bit in Direktform und in transponierter Form (Abb. 5.2) aufgebaut werden, ohne dass im Filter Bits abgeschnitten werden.

- a) (Ohne Quantisierung): Wie lauten die Übertragungsfunktion $H(z)$ und der Frequenzgang $H(e^{j\omega})$? Bei welcher Frequenz liegt die erste Nullstelle im Frequenzgang?
- b) Zeichnen Sie die Wortbreiten in Abb. 5.2 ein. Wie groß ist jeweils die Wortbreite am Ausgang?
- c) Welche Verzögerung hat jeweils der kritische Pfad? Mit welcher Taktrate kann das Filter maximal arbeiten? Nehmen Sie an, dass alle Addierer die gleiche Verzögerung $\tau_{add} = 2 \text{ ns}$ haben und die Setup / Hold-Zeiten der Register vernachlässigt werden können.

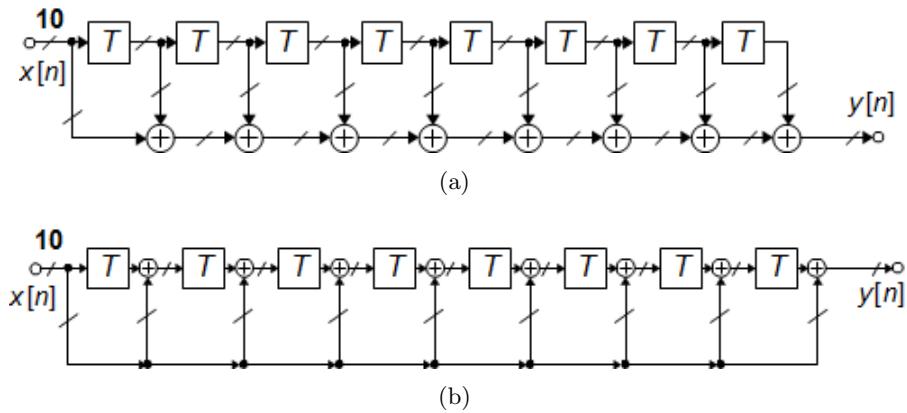


Abb. 5.2.: Moving Average Filter mit Quantisierung zu Aufgabe 5.2, (a) Direktform und (b) transponierte Direktform

- d) Wie viele Flipflops (= Register x Wortbreite) und Volladdierer (= Addierer x Wortbreite) benötigen die beiden Realisierungen?

5.3. * FIR Filter mit Quantisierung → M5.3

Der Entwurf eines FIR-Filters hat die folgenden Koeffizienten geliefert:

$$b = [0.01623, 0, -0.06871, 0, 0.30399, 0.5, 0.30399, 0, -0.06871, 0, 0.01623]$$

Dieses Filter soll in Direktform auf einem FPGA implementiert werden (Abb. 5.3), das hardwa-reoptimierte 18 x 18 Bit Multiplizierer enthält. Die Daten am Eingang $x[n]$ und am Ausgang $y[n]$ in normalisierter Zweierkomplementform haben eine Wortlänge von 16 Bit, $Q_X 0.15 = Q_Y 0.15$ (vorzeichenbehaftet, 15 fractional Bits, Wertebereich $-1 \leq x, y < 1$).

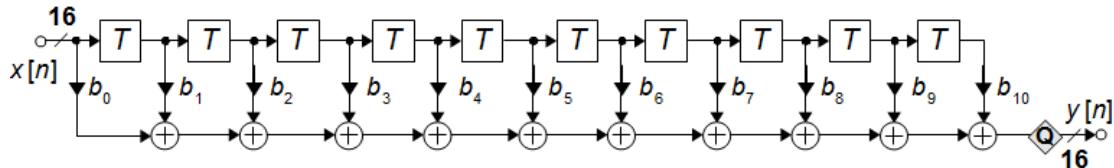


Abb. 5.3.: FIR-Filter mit Quantisierung zu Aufgabe 5.3

Hinweis: Die folgenden Fragen lassen sich mit wenigen Ausnahmen unabhängig voneinander lösen.

- a) Welche Ordnung hat das Filter? Ist das Filter linearphasig? Ist es ein Halbbandfilter? Wenn ja, werden diese Eigenschaften durch Koeffizientenquantisierung gestört?
- b) Wieviele „echte“ Multiplizierer benötigt das Filter? Es sollen Symmetrieffekte und offensichtliche Vereinfachungen ausgenutzt werden. Zeichnen Sie die optimierte Struktur auf!
- c) Welche Verzögerung hat der kritische Pfad dieser Implementierung? Mit welcher Taktrate kann das Filter maximal arbeiten? Nehmen Sie an, dass alle „echten“ Multiplizierer die gleiche Verzögerung $\tau_{mult} = 10$ ns und alle Addierer die gleiche Verzögerung $\tau_{add} = 2$ ns haben. Die Setup / Hold-Zeiten der Register können vernachlässigt werden. Vergleichen Sie diese Werte mit denen eines allgemeinen FIR Filters 10. Ordnung in Direktform (Abb. 5.3).

- d) Welchen Wert hat der Frequenzgang $H(f)$ bei $f = 0$, $f = f_S/4$ und $f = f_S/2$?
- e) Welches Datenformat $Bx.y$ ist optimal (= minimaler Quantisierungsfehler) zur Darstellung der Koeffizienten b geeignet bei einer Gesamtwortlänge $W = 18$ Bit? Alle Koeffizienten sollen das gleiche Format haben. Wie groß ist ein LSB in der gewählten Darstellung?
- f) Quantisieren Sie die Koeffizienten durch Wertschneiden (`floor`) und ermitteln Sie jeweils den Quantisierungsfehler $\epsilon(0, QF)$. Welche Werte ergeben sich für eine Wortbreite der Koeffizienten von 8 Bit? Hinweis: Die Bestimmung mit Hilfe eines Python / Matlab-Skripts ist deutlich einfacher als eine Handrechnung ... Simulieren Sie für diese Darstellung den Frequenzgang des Filters.
- g) Welche Koeffizientenfläche A_i hat das Filter?
- h) Wie muss das Eingangswort (16 bit) vor der Multiplikation (18 x 18 bits) skaliert werden? Welche Bits des Produkts sind (für beliebige Eingangsworte und Koeffizienten) relevant?
- i) Zur Minimierung von Rundungsverlusten soll das Ergebnis erst nach Aufsummierung aller Teilprodukte auf 16 Bit am Ausgang requantisiert werden. Welche Wortbreite muss dazu der Akkumulator haben? Bestimmen Sie die Wortbreite zunächst für beliebige Koeffizienten und danach unter Berücksichtigung der Koeffizientenfläche.
- j) Wie muss das Akkumulatorwort skaliert werden (d.h. welcher Teil wird benutzt, welche Bits werden abgeschnitten), um den Quantisierungsfehler zu minimieren? Es darf dabei kein Überlauf bei beliebigen Eingangswerten $x[n]$ auftreten.

5.4. FIR Filter mit Skalierung → M5.4

Das FIR-Filter F1 aus Aufgabe 4.6 mit der Impulsantwort $h_1[n] = \{1; 7/4; 1/2; -1/4\}$ und dem Betragsfrequenzgang in Abb. 5.4 soll jetzt in Fixpoint-Arithmetik implementiert werden mit einem Wertebereich $-1 < x, y < 1$ am Eingang und Ausgang, das Filter ist in Direktform aufgebaut.

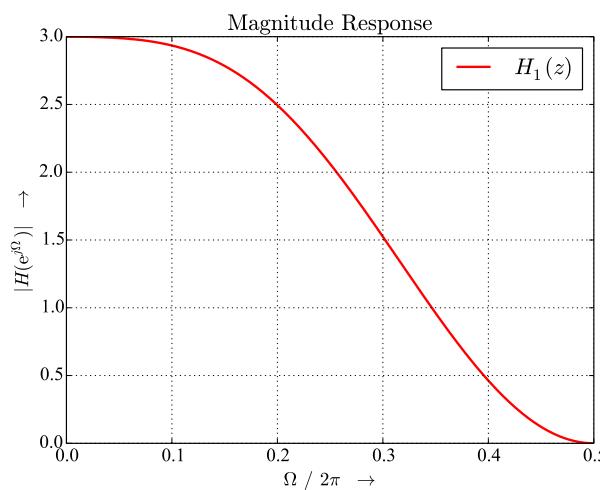


Abb. 5.4.: Betragsfrequenzgang zu Aufgabe 5.4

- a) Skizzieren den Signalfussgraph (die Struktur) des Filters. An welcher Stelle kann im Filter ein Überlauf auftreten?
- b) Bestimmen Sie den Skalierungskoeffizienten $k_{E\infty}$ so, dass für beliebige sinusförmige Eingangssignale oder Gleichspannung ($f = 0$) kein Überlauf entsteht (sog. L^∞ -Skalierung). Mit welcher Eingangsfolge $x_\infty[n]$ wird bei dieser Skalierung $y[n]_{max} = 1$ erreicht?
- c) Bestimmen Sie den Skalierungskoeffizienten k_{E1} so, dass für ein *beliebiges Eingangssignal* kein Überlauf entsteht (sog. L^1 -Skalierung). Überlegen Sie dazu zuerst, welche Eingangsfolge $x_1[n]$ der worst case für das System ist, also bei welcher Eingangsfolge das maximale Ausgangssignal entsteht.

5.5. * IIR-Filter mit Skalierung → M5.5

- a) Ermitteln Sie die **Übertragungsfunktion** $H(z) = Y(z)/X(z)$ der Filterstruktur in Abb. 5.5 in Polynomform.
- b) Ermitteln Sie die **Pole und Nullstellen** von $H(z)$ und tragen Sie sie in ein Pol-Nullstellen-Diagramm ein. Bei welchen Frequenzen nimmt der Amplitudengang den maximalen bzw. minimalen Wert an? Um welchen Filtertyp (HP, TP, BP, BS) handelt es sich?

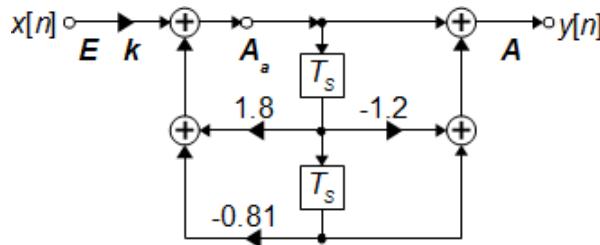


Abb. 5.5.: Rekursive Filterstruktur H zu Aufgabe 5.5

- c) **Skalierung von $H_{Aa}(z)$:** Wie müsste der Faktor $k = k_c$ gewählt werden, damit der Amplitudengang $H_{Aa}(f)$ vom Eingang E zum Punkt Aa den Wert 1 bei $f = 0$ hat?
- d) **Skalierung von $H(z)$:** Wie müsste $k = k_d$ gewählt werden, damit der Amplitudengang $H(f = 0)$ vom Eingang E zum Ausgang A den Wert 1 hat? Kann bei dieser Skalierung ein sinusförmiges oder ein DC - Signal mit Amplitude 1 eine Überschreitung des Wertebereichs $-1 \dots 1$ an einem beliebigen Knoten der Struktur hervorrufen?
- e) **Optimierung der Struktur:** Es soll jetzt der Faktor k_c aus Unterpunkt c) verwendet werden. Der nicht-rekursive Teil der Filterstruktur H soll durch zwei zusätzliche Multiplizierer so verändert werden, dass $H(f = 0) = 1$ ist. Die Filtercharakteristik $H(f)$ und die Lage von Pol- und Nullstellen sollen dabei unverändert bleiben. Skizzieren Sie den nicht-rekursiven Teil des Filters und tragen Sie die Werte der drei Koeffizienten ein.
- f) Geben Sie die **Impulsantwort** $h[n]$ für $n = 0 \dots 3$ an für $k = k_f = 1/128$.
- g) Bestimmen Sie den **Amplitudengang** $H(f)$ und den **Phasengang** $\varphi_H(f)$ bei $f = 0$ und $f = f_s/2$ für die gleiche Struktur.

5.6. IIR-Filter mit Quantisierung und kleinen Grenzzyklen → M5.6

Ein digitales Filter habe die Differenzengleichung $y[n] = x[n] + ay[n - 1]$ mit $a = -0,88$. Am Eingang liege der Einheitsimpuls, $x[n] = \delta[n]$. Das Filter arbeitet mit Festkomma-Arithmetik, zur einfacheren Demonstration von Quantisierungseffekten soll in dieser Aufgabe zunächst angenommen werden, dass das Filter mit dezimaler anstelle binärer Arithmetik arbeitet.

- Bestimmen Sie zunächst die exakte Impulsantwort $h[n]$ für $n = 0 \dots 10$ mit dem Taschenrechner (oder Matlab ...) und notieren Sie die Werte mit drei Nachkommastellen.
- Ermitteln Sie jetzt $h[n]$ für $n = 0 \dots 10$, wenn das Ergebnis der Multiplikation vor der Subtraktion auf eine Nachkommastelle (dezimal) gerundet wird, also z.B. $0,64 \rightarrow 0,6$ und $0,75 \rightarrow 0,8$.
- Ermitteln Sie jetzt $h[n]$ für $n = 0 \dots 10$ für den Fall, dass das Ergebnis der Multiplikation auf eine Nachkommastelle (dezimal) abgeschnitten wird, also z.B. $0,64 \rightarrow 0,6$ und $0,75 \rightarrow 0,7$.
- Das Filter arbeite jetzt mit binärer Arithmetik mit Rundung, die Quantisierungsstufe sei $q = 1$ LSB = $1/16$ (4 Fractionalbits). Es seien $a = -9/16$ und $a = 9/16$. Ermitteln Sie die Impulsantwort für beide Koeffizientenwerte.

5.7. Integrator mit endlicher Wortbreite → M5.7

In dieser Aufgabe soll bestimmt werden, ob und wann die Integratoren in Abb. 5.6 mit einer Wortlänge von 8 bit überlaufen, wenn am Eingang ein Einheitsimpuls $\delta[n]$ angelegt wird. Zum Zeitpunkt $n = 0$ haben alle Integratoren den Wert 0.

Skizzieren Sie außerdem Pol-Nullstellendiagramm und den Frequenzgang (für Integratoren ohne Wortlängenbegrenzungen). Bei welcher Frequenz wird $|H(e^{j\Omega})| = 1$?

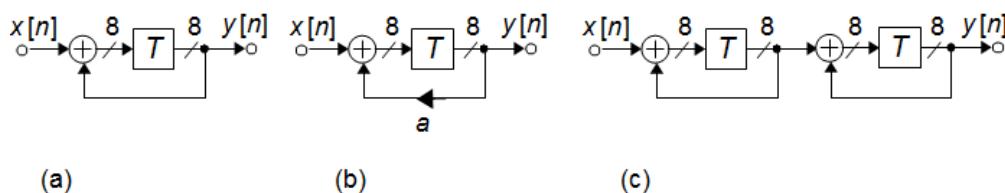


Abb. 5.6.: Integratoren mit endlicher Wortbreite zu Aufgabe 5.7, (a) erster Ordnung, (b) erster Ordnung, verlustbehaftet, (c) zweiter Ordnung

Vor allem die Integratoren b) und c) lassen sich am Einfachsten mit Hilfe der z-Transformation analysieren. Hierfür sind die folgenden Transformationspaare nützlich:

$$\begin{aligned}
 u[n] & \circlearrowleft \bullet \frac{z}{z-1} = \frac{1}{1-z^{-1}} \\
 nu[n] & \circlearrowleft \bullet \frac{z}{(z-1)^2} = \frac{z^{-1}}{(1-z^{-1})^2} \\
 n^2u[n] & \circlearrowleft \bullet \frac{z(z+1)}{(z-1)^3} = \frac{z^{-1}(1+z^{-1})}{(1-z^{-1})^3} \\
 a^n u[n] & \circlearrowleft \bullet \frac{z}{z-a} = \frac{1}{1-az^{-1}}
 \end{aligned}$$

Untersuchen Sie auch die Sprungantwort von Integrator a). Die Sprungantwort von Integrator b) und c) lässt sich über Partialbruchzerlegung berechnen (komplizierter, ohne Musterlösung).

- a) Verwenden Sie jetzt als Eingangssignal xxx - es tritt ein Überlauf auf. Wie unterscheidet sich der Grenzzyklus von dem aus der vorigen Aufgabe?
- b) Überprüfen Sie, dass bei Verwendung von ('sat') keine Grenzzyklen mehr auftreten.

5.8. * Verständnisfragen zu Filtern mit Quantisierung → M5.8

Im Folgenden sollen Filter betrachtet werden, bei denen Koeffizienten und Arithmetik quantisiert sind. Kreuzen Sie in Tab. 5.2 an, welche Aussage auf welche Filter zutrifft. Kann je nach Systementwurf „Ja“ oder „Nein“ zutreffen, kreuzen Sie beides an.

Aussage	FIR-Filter		IIR-Filter	
	ja	nein	ja	nein
Unendliche Impulsantwort:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Stabil:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Linearphasig:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Minimalphasig:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Ursprungs:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Einheitskreises:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nullstellen außerhalb des Ursprungs:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nullstellen außerhalb des Einheitskreises:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Halbbandfilter:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Konstante Gruppenlaufzeit:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transponierte Form:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Entwurf über bilineare Transformation:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transversale Struktur:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Abgeleitet von analogen Butterworth-Filtern:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Kaskadierte Form ist unempfindlicher gegen Quantisierungseffekte als direkte Form:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Grenzzyklen:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Tab. 5.2.: Verständnisfragen zu Filtern mit Quantisierung (Aufgabe 5.8)

6. NOI: Wortlängeneffekte im Frequenzbereich: Quantisierungsrauschen

Inhalt

Dieses Kapitel beschäftigt sich mit der Beschreibung der Signalquantisierung im *Frequenzbereich*. Dabei geht es immer um Quantisierungsrauschen in folgenden Zusammenhängen:

Quantisierung / Analog-Digital Wandlung: Wie kann ich die nicht-lineare Quantisierung eines analogen (= unendlich fein aufgelösten) Signals im statistischen Mittel so beschreiben, dass ich weiterhin Fourier- und z-Transformation anwenden kann?

Requantisierung: Welchen Einfluss hat die Reduktion der Wortbreite eines bereits quantisierten Signals?

(Noise Shaping): Wie kann ich das unvermeidliche Quantisierungsrauschen so formen, dass es mich möglichst wenig stört?

Zur Beschreibung werden folgende wichtige Begriffe und Größen verwendet:

Rauschleistung

Signal- zu Rauschleistungsverhältnis (SQNR)

Rauschleistungsdichte

Effective Number of Bits (ENOB)

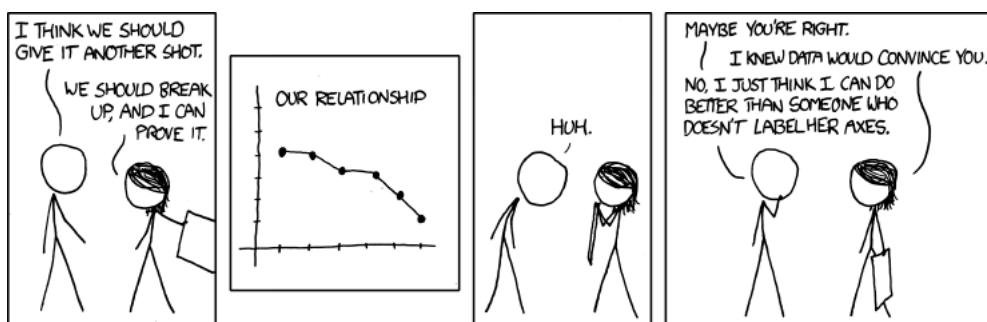


Abb. 6.1.: „And if you labeled your axes, I could tell you exactly how MUCH better“
[<http://xkcd.com/833/>]

Theorie

Quantisierungsrauschen

Im Kap. 5 wurde untersucht, welche *Quantisierungsfehler* $\epsilon[n] = x[n] - x_q[n]$ auftreten, wenn man den unendlich fein aufgelösten Wertebereich eines analogen Signals *diskretisiert* oder *quantisiert*, also auf eine endliche Anzahl $M = 2^W$ von Werten abbildet. Reduziert man die Wortlänge W eines bereits quantisierten Signals, verringert man also die Anzahl der zur Verfügung stehenden Werte, entstehen ebenfalls Quantisierungsfehler. Diesen Prozess nennt man *Requantisierung*. Beide Vorgänge können mit den gleichen mathematischen Methoden untersucht werden. Wichtig ist hier vor allem der Wertebereich oder *Full Scale Range (FSR)* und die Höhe q der Quantisierungsstufen:

$$FSR = (M - 1)q = (2^W - 1)q \approx 2^W q \quad \Leftrightarrow \quad q \approx \frac{FSR}{2^W - 1} \approx \frac{FSR}{2^W} \quad \text{für } 2^W \gg 1 \quad (6.1)$$

Damit keine Übersteuerungen auftreten, ist die maximal darstellbare Amplitude eines symmetrischen Wechselsignals

$$A_{max} = FSR/2 \approx 2^{W-1}q. \quad (6.2)$$

Die maximale Signalleistung eines *sinusförmigen* Signals ist daher

$$S_{max} = A_{max}^2/2 = FSR^2/8. \quad (6.3)$$

Im Kap. 5 wurden diese Vorgänge im Zeitbereich (z.B. Grenzzyklen) betrachtet, und es wurde untersucht wie man Systeme skalieren muss, damit kein Überlauf bzw. Sättigung auftritt. Außerdem wurde betrachtet, welchen Einfluss die Quantisierung der Filterkoeffizienten auf den Frequenzgang des Filters hat. Die Quantisierung der *Koeffizienten* eines linearen Systems $H(z)$ führt zu einem neuen linearen¹ System $H_Q(z)$, das mit den bisherigen Methoden der Signalverarbeitung beschrieben werden kann. Die Quantisierung des *Signals* führt dagegen zu einem

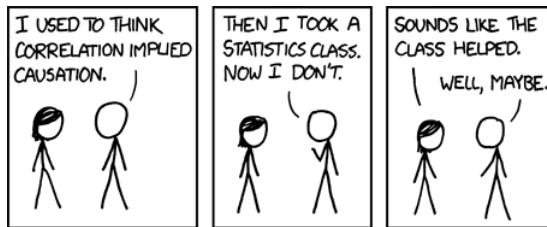


Abb. 6.2.: Korrelation [<http://xkcd.com/552/>]

nicht-linearen System, das mit den üblichen Methoden der Signaltheorie nicht beschrieben werden kann. Daher beschreibt man Quantisierungsfehler als statistischen Prozess und kann so das System trotz der offensichtlichen Nichtlinearität der Quantisierung als linear betrachten und darf die mathematischen Methoden für LTI-Systeme (z -Transformation, Fourier-Transformation etc.) weiterhin verwenden. Der Quantisierungsfehler wird in diesem Kapitel als stationärer ergodischer Zufallsprozess („Quantisierungsrauschen“) betrachtet, der durch seinen Mittelwert m und die Varianz σ^2 beschrieben wird. Die Varianz ist gleich der AC-Leistung des Fehlers, $N_Q = \sigma^2$. Für die gängigen Quantisierungsmethoden *Runden* und *Wertschneiden* (truncation) gilt $\sigma^2 = q^2/12$ (Herleitung z.B. [OS10], [KK09]):

$$q = 2^{-W} FSR \quad \Rightarrow \quad N_Q = \sigma^2 = \frac{q^2}{12} = \frac{FSR^2}{12 \cdot 2^{2W}} \quad (6.4)$$

¹sofern es stabil ist

SQNR

Bei einer vorzeichenbehafteten Repräsentation $X(WI.WF)$ stehen $N = 2^W$ Quantisierungsstufen zur Verfügung mit $WI + 1 + WF = W$. Der darstellbare Zahlenbereich ist $-2^{WI} \leq x \leq 2^{WI} - 2^{-WF} \approx 2^{WI}$. Ein Sinussignal, das den darstellbaren Bereich voll ausnutzt, hat daher eine Amplitude von $A_{max} = 2^{WI}$ und eine Leistung von $S_{max} = A_{max}^2/2 = 2^{2WI}/2$.

Die Rauschleistung eines quantisierten Signals wird bestimmt durch die Größe der Quantisierungsstufe $q = 2^{-WF} = 1$ LSB - je feiner die Quantisierung, desto geringer der Quantisierungsfehler und damit im Mittel das Quantisierungrauschen. Wird das quantisierte Signal durch Runden oder Wertschneiden aus dem ursprünglichen Signal gewonnen, beträgt die Quantisierungsrauschleistung $N_q = q^2/12 = 2^{-2WF}/12 = 2^{-2WF-2}/3$.

Damit ist das maximale² Verhältnis der Leistung eines sinusförmigen Signals zur Quantisierungsrauschleistung, das Signal-to-Quantization Noise Ratio ($SQNR_{max}$) in dB:

$$\begin{aligned} SQNR_{max} &= 10 \log_{10} \frac{S}{N_q} \text{ dB} = 10 \log_{10} \frac{2^{2WI}/2}{2^{-2WF-2}/3} \text{ dB} = 10 \log_{10} \left(\frac{3}{2} 2^{2(WI+1+WF)} \right) \text{ dB} \\ &= 1,76 \text{ dB} + 10 \log_{10} 2^{2W} \text{ dB} = 1,76 \text{ dB} + W \cdot 20 \log_{10} 2 \text{ dB} \\ &= 1,76 \text{ dB} + W \cdot 6,02 \text{ dB} = 98,1 \text{ dB für } W = 16 \end{aligned}$$

Das Verhältnis aus maximaler Signalleistung und Quantisierungsrauschleistung nennt man *maximales Signal-to-Quantization-Ratio* $SQNR_{max}$:

$$\begin{aligned} SQNR_{max} &= \frac{S_{max}}{N_Q} = \frac{FSR^2}{8} \frac{3 \cdot 2^{2W+2}}{FSR^2} = \frac{3}{2} \cdot 2^{2W} \\ &\cong 10 \log 1,5 \text{ dB} + 10 \log 2^{2W} \text{ dB} = (1,76 + 6,02W) \text{ dB} \end{aligned} \quad (6.5)$$

Manchmal werden bei logarithmischen Rechnungen die Pseudo-Einheiten dBV und dBW verwendet, sie sind definiert über $a \text{ dBV} = 20 \log_{10}(A/1 \text{ V})$ bzw. $p \text{ dBW} = 10 \log_{10}(P/1 \text{ W})$.

Beispiel: ADC mit $W = 18$ bit Auflösung und Eingangsbereich von $\pm 1 \text{ V}$

$$\begin{aligned} &\Rightarrow FSR = 2 \text{ V}, A_{max} = 1 \text{ V}, S_{max} = 0,5 \text{ V}^2 \cong -3,01 \text{ dBV} \\ &\Rightarrow q = 2^{-18} \cdot FSR = 7,63 \mu\text{V} \cong -102,4 \text{ dBV}, N_Q = 4,85 \cdot 10^{-12} \text{ V}^2 \cong -113,1 \text{ dBW} \\ &\Rightarrow SQNR_{max} = 1,031 \cdot 10^{-11} \cong 110,1 \text{ dB}. \end{aligned}$$

Anmerkung: Im physikalischen Sinne ist V^2 natürlich keine Leistung. In der Nachrichtentechnik ist es aber üblich quadrierte Größen als Leistung zu bezeichnen, da durch Division durch einen (Abschluss-)Widerstand jederzeit die physikalische Leistung berechnet werden kann. Meist werden sowieso Leistungsverhältnisse betrachtet (SNR , Verstärkung), aus denen der Widerstand herausfällt.

Noch eine Anmerkung: Wenn man von anderen Signalformen ausgeht, ergeben sich auch bei Vollaussteuerung leicht unterschiedliche SNR -Werte: Ein dreieckförmiges Signal hat eine Leistung von $A^2/3$, damit ergibt sich $SQNR_{max} = W \cdot 6,02 \text{ dB}$.

Quantisierung am DAC: Der DAC an sich quantisiert das Signal nicht (und fügt daher auch kein Quantisierungrauschen hinzu), sofern das digitale Signal die gleiche Wortlänge hat wie der DAC. Das ist aber meist nicht der Fall; in der digitalen Signalverarbeitung wird man meist

²d.h. für Vollaussteuerung

ein paar Reservebits zur Verringerung von Rundungsverlusten vorhalten, während beim DAC als analoger Komponente jedes zusätzliche Bit die Kosten drastisch erhöht. Vor dem DAC ist daher in den allermeisten Fällen eine *Requantisierung* notwendig - siehe nächster Abschnitt.

Requantisierung

Bei einem ADC ist klar, was der *FSR* und was die Wortlänge *W* ist und wie groß dementsprechend die Höhe der kleinsten darstellbaren Stufe *q* und das daraus folgende Quantisierungsrauschen ist, da hier physikalische Größen zugeordnet werden können.

Signal- und Rauschleistung von digitalen Signalen können nicht eindeutig angegeben werden, da die Signalwerte keine direkte physikalische Bedeutung haben und die Position des Binärpunkts Interpretationssache ist. Das maximale *SQNR* eines requantisierten Signals kann man trotzdem mit (6.5) bestimmen, da hier nur das Verhältnis beider Größen eingeht; *W* ist dabei ebenfalls die Wortlänge *nach* der (Re-)quantisierung.

Wenn man zu Vergleichszwecken trotzdem *S* und *N_Q* angeben möchte, empfiehlt es sich die Signalleistung wieder abhängig vom *FSR* anzugeben. Bei einer Darstellung mit *WI.WF*, also *WI* Integerbits, *WF* Fractionalbits und einem Vorzeichenbit und insgesamt *WI + 1 + WF = W* bit gilt:

$$\begin{aligned} \mathbf{FSR} &= 2^{WF} - 2^{-WF} - (-2^{WF}) = 2^{WF+1} - 2^{-WF} \approx 2^{WF+1} \text{ für } 2^{-WF} \ll 2^{WF} \\ \Rightarrow A_{\max} &\approx 2^{WF} \quad \text{und} \quad S_{\max} = \frac{A_{\max}^2}{2} \approx \frac{2^{2WF}}{2} \end{aligned} \quad (6.6)$$

Das maximal erreichbare *SQNR* bei der Requantisierung erhält man immer aus der *Gesamtanzahl* *W = WF + WF* der Bits *nach* der Requantisierung!

Die von einer (Re-)Quantisierungsstufe erzeugte Quantisierungsrausleistung wird auch hier bestimmt durch den Quantisierungsfehler *q*, also durch die kleinste darstellbare Stufe, die wiederum von der Wortlänge *W* und dem Aussteuerbereich *FSR* abhängt:

$$q = \frac{FSR}{2^W} = \frac{2^{WF+1}}{2^{WF+1+WF}} = 2^{-WF} \quad \Rightarrow \quad N_Q = \frac{q^2}{12} = \frac{2^{-2WF}}{12} \quad (6.7)$$

Wer möchte, kann sich vorstellen, dass die Anzahl der Integerbits den Aussteuerbereich und die Anzahl der Fractionalbits den Quantisierungsfehler bestimmen, auch wenn die Position des „Kommas“ bei Fixpointzahlen willkürlich ist. Dividiert man (6.6) und (6.7) kommt man jedenfalls zum gleichen Ergebnis wie (6.5).

Achtung: An unterschiedlichen Stellen im System (Eingang, Ausgang, nach Multiplikationen) werden Signale meist durch unterschiedliche Quantisierungen repräsentiert.

Effective Number of Bits

N_Q ist unabhängig von *A* - eine Verringerung der Signalamplitude führt daher zwangsläufig zu einem schlechteren *S(Q)NR*! Diese Verschlechterung gegenüber *SQNR_{max}* (auch durch andere Nichtidealitäten des ADCs wie Nichtlinearitäten, thermisches Rauschen, gestörte Spannungsversorgung, Clockjitter, ...) lässt sich als verringerte effektive Wortlänge *Effective Number of Bits (ENOB)* ausdrücken:

$$ENOB = \frac{SNR [\text{dB}] - 1,76 \text{ dB}}{6,02 \text{ dB}} \text{ bit} \quad (6.8)$$

Mit dem *ENOB* kann man verschiedene ADCs oder Signalverarbeitungssysteme mit gleicher (nomineller) Wortbreite vergleichen und abschätzen, wie gut der theoretisch mögliche Signal-Rauschabstand erreicht wird.

Rauschleistungsdichte

Da der Quantisierungsfehler als stationärer ergodischer Zufallsprozess angesehen wird (und dies in Praxis meist auch mit hinreichender Genauigkeit zutrifft), ist das Spektrum des Quantisierungsrauschen weiß. Die Quantisierungsrauschleistung N_Q verteilt sich daher auf das gesamte Basisband $0 \dots f_S$ bzw. $-f_S/2 \dots f_S/2$, die *Rauschleistungsdichte* ist $N'_Q = N_Q/f_S$ und wird in W / Hz angegeben. Aus Rauschleistungsdichte und Bandbreite kann man die Rauschleistung berechnen, z.B. wenn durch ein Filter Teile des Quantisierungsrauschen ausgefiltert wurden.

Anmerkung: Man kann die Rauschleistungsdichte genauso gut für ein einseitiges ($f = 0 \dots f_S/2$) Spektrum angeben. Der Wert für N'_Q wäre dann doppelt so groß, man muss dann aber auch die zweiseitige Rauschbandbreite $-B_N \dots +B_N$ berücksichtigen und kommt insgesamt wieder auf die gleiche Rauschleistung.

Vor allem in der Hochfrequenztechnik werden beide Definitionen für die Rauschleistungsdichte / Power Spectral Density verwendet, man muss daher bei der Interpretation von Messergebnissen aufpassen, was gemeint war.

Anmerkung: Rauschen ist ein breitbandiges Signal, dessen Leistung sich bei Analyse mit einer DFT über alle Bins verteilt. Ändert man die Zahl der Bins ($= N_{FFT}$), ändert sich auch die angezeigte Leistung je Bin. Signale mit Linienspektren wie der Sinuston fallen dagegen idealerweise nur in ein Bin - hier gibt es keinen Zusammenhang zu N_{FFT} . Bei Erhöhung der FFT-Länge sinkt das angezeigte Rauschen („Noise Floor“) immer weiter ab und diskrete Spektrallinien sind klarer erkennbar. Bei konstanter Abtastfrequenz bedeutet das natürlich auch eine immer längere Messzeit.

Bei der Spektralanalyse mit einer N_{FFT} Punkt DFT fallen N_Q/N_{FFT} in jedes Bin ($2N_Q/N_{FFT}$ für einseitige DFT). Die Bandbreite eines Bins ist $\Delta f = f_S/N_{FFT}$. Um die Rauschleistungsdichte aus der DFT direkt abzulesen, muss der angezeigte Wert daher durch diese Bandbreite geteilt werden.

Python: Befehle und Programme zu Kap. 6

Spezielle Python/Matlab-Befehle werden für dieses Kapitel nicht benötigt: Die Quantisierung wurde bereits im Kap. 5 gezeigt, die Darstellung im Frequenzbereich erfolgt mit Hilfe der DFT (Kap. 3). Aufbauend auf Listing 5.1 zeigt Listing 6.1 wie man Quantisierungsrauschen, Rauschleistungsdichte und Rauschleistung in Python simuliert. Ein paar Dinge muss man dabei beachten:

Kohärente DFT: Wenn man periodische Signale quantisiert, sollte man die DFT unbedingt von einer ganzen Anzahl von Perioden bestimmen, ausgeschnitten mit einem Rechteckfenster. Die Anzahl der Perioden sollte eine Primzahl sein; dadurch werden Periodizitäten des Quantisierungsfehlers minimiert und das Rauschspektrum ist gleichmäßiger verteilt.

Amplituden und Leistungen: Die DFT gibt die komplexe *Amplitude* A_k in jedem Frequenzband zurück. Möchte man die *Leistung* im Frequenzband k wissen, muss man $P_k = \underline{A}_k \underline{A}_k^*/2 = |A_k|^2/2$ berechnen! $20 \log_{10}(|A_k|)$ ist 3 dB größer als die erwartete Leistung in dB. In Listing 6.1 wird die DFT durch $\sqrt{2}$ geteilt³, um dieses Problem zu umgehen: Man erhält dann Effektivwerte je Band anstatt der Amplituden, aus denen man direkt die Leistung berechnen kann.

Ein- oder zweiseitiges Spektrum: Möchte man nur physikalische (positive) Frequenzen darstellen, muss man die DFT-Amplituden (Beträge!) bei negativen Indices zu denen bei positiven Indices addieren, um auch im Frequenzbereich korrekte Leistungswerte zu erhalten. Bei reellwertigen Zeitsignalen kann man einfach alle Werte der DFT verdoppeln - bis auf den DC - Wert (Index 0)!

Mittlere Rauschleistungsdichte: Theoretisch sollte die Quantisierungsrauschleistung bei allen Frequenzen gleich sein, praktisch gibt es bei der Quantisierung von sinusförmigen Signalen Korrelationseffekte. Daher mittelt man die Rauschleistung aller Frequenzbänder - *nach der Leistungsberechnung (Quadrieren⁴) und vor dem Logarithmieren!* Die Amplitude des unquantisierten Signals muss man vor der Berechnung im entsprechenden Frequenzband subtrahieren, da man ja nur die Rauschleistung berücksichtigen möchte. Den Index erhält man über $k = f_{sig}/\Delta f = N f_{sig}/f_S$.

Rauschleistung: Hat man die mittlere Rauschleistungsdichte bestimmt, muss man sie nur noch mit $N_{FFT}/2$ multiplizieren - *vor dem Logarithmieren!* - um die gesamte Rauschleistung zu erhalten.

```

1 # Ende der gemeinsamen Import-Anweisungen
2 NOISE_LABEL = True
3 NFFT = 2000; f_a = 2; N_per = 7
4 T_mess = N_per / f_a; T_S = T_mess / NFFT
5 t = linspace(0, T_mess, NFFT, endpoint=False)
6 a = 1.1 * sin(2 * pi * f_a * t)
7 #
8 q_obj = {'QI':1, 'QF': 3, 'quant':'round', 'ovfl': 'wrap'} # versuchen Sie auch 'floor' / 'fix' und 'sat'
9 fx_a = fx.Fixed(q_obj)
10 aq = fx_a.fix(a)
11
12 print('Anzahl der Überläufe = ', fx_a.N_over)
13 #
14 figure(1)
15 title('Quantisiertes Sinussignal und Quantisierungsfehler')
16 plot(t, a, label = r'$a(t)$')
17 plt.step(t, aq, where = 'post', label = r'$a_Q(t)$')
18 plot(t, a-aq, label = r'$a(t) - a_Q(t)$')
19 plt.legend(fontsize = 14)
20 xlabel(r'$t$ \; ; \mathrm{/ \; s} \; \rightarrow$'); ylabel(r'$a \rightarrow$')
21 A_max = 2**q_obj['QI'] - 2**-q_obj['QF']
22 plt.axhline(y = A_max, linestyle = '--', color = 'k')
23 plt.axhline(y = -A_max, linestyle = '--', color = 'k')
24 #
25 Amin = -100 # Unteres Limit in dB für die Darstellung
26 A = abs(2 / sqrt(2) * fft(a) / NFFT)[0:NFFT / 2. - 1] # einseitiges Spektrum,
27 AQ = abs(2 / sqrt(2) * fft(aq) / NFFT)[0:NFFT / 2. - 1] # Effektivwert !
28 A[0] = A[0] * sqrt(2)/2; AQ[0] = AQ[0] * sqrt(2)/2 # korrigiere DC-Wert zurück
29 f = fftfreq(NFFT, T_S)[0:NFFT/2. - 1]      # Frequenzen f. einseitiges Spektrum
30 #
31 fig2 = figure(2)

```

³Achtung: Mit Ausnahme des DC-Werts!!!

⁴Im Gegensatz zu Matlab bewirkt $AQ * AQ$ in Python eine *elementweise* Quadrierung des Vektors.

```

32 title('Spektrum von Eingangs- und quantisiertem Signal')
33 ax2 = fig2.add_subplot(111)
34 ax2.stem(f, 20 * log10(AQ), bottom = Amin, label = r'$A_Q(f)$')
35 ml, sl, bl = stem(f, 20 * log10(A), bottom = Amin, label = r'$A(f)$')
36 plt.setp(ml, 'markerfacecolor', 'r', 'markersize', 10, 'alpha', 0.4) # Marker
37 plt.setp(sl, 'color','r', 'linewidth', 5, 'alpha', 0.4) # Stemline
38 AQ[f_a * T_mess] = abs(AQ[f_a * T_mess] - A[f_a * T_mess]) # Subtrahiere Signal
39 S = 10*log10(A[f_a * T_mess]**2.) # Signalleistung, gemessen
40 N_PSD = 10 * log10(np.average(AQ*AQ)) # mittlere Rauschleistungsdichte
41 NQ = N_PSD + 10*log10(NFFT/2)
42 ENOB = ((S-NQ) - 1.76)/6.02
43 # print(S, NQ, S - NQ)
44 plt.axhline(y = N_PSD) # plotte horiz. Linie mit mittlerer Rauschleist.dichte
45 ax2.set_xlabel(r'$f \text{ ; } \mathbf{Hz} \rightarrow$')
46 ax2.set_ylabel(r'$A_{\text{eff},q}(f) \text{ ; } \mathbf{dB} \rightarrow$')
47 ax2.set_yscale('log')
48 plt.text(1/(4*T_S), Amin,
49         r'$S = %.2f \text{ ; } \mathbf{dB} \text{, } N_Q = %.2f \text{ ; } \mathbf{dB} \text{, } $',
50         r'$SQNR = %.2f \text{ ; } \mathbf{dB} \text{'}(S, NQ, S-NQ) + \n' +
51         r'$ENOB = %.2f \text{ ; } \mathbf{bits} \text{, } N_{FFT} = %d \text{, } f_S = %.2f \text{ ; } $',
52         r'$\mathbf{Hz} \text{'}(ENOB, NFFT, 1./T_S) \text{, } $',
53         ha='center', va='bottom', bbox=dict(facecolor='0.8', alpha=0.8))
54 #
55 plt.legend(fontsize = 14)
56 if NOISE_LABEL:
57     ax2b = ax2.twinx()
58     ax2b.set_yscale('linear')
59     ax2b.set_ylabel(r'$N_q(f) \text{ ; } \mathbf{dBW / Hz} \rightarrow$')
60 plt.show()

```

Lst. 6.1: Quantisierungsrauschen mit Python (`NOI/NOI_intro_py.py`)

Filename	Beschreibung
<code>NOI_intro</code>	Quantisierung im Frequenzbereich (Listing 6.1)
<code>NOI_DFT.py</code>	Skalierung von Signal (Linienspektrum) und Rauschen (Breitbandspektrum) im Frequenzbereich
<code>NOI_DFT_wideband</code>	Code zu Aufgabe 6.4: DFT von Schmal- und Breitbandsignalen
<code>NOI_DFT_wideband_ML</code>	Musterlösung zu Aufgabe 6.4
<code>NOI_QNoise_Filt.py</code>	Testbench zur Simulation von Quantisierungseffekten in „richtigen“ Filtern im Frequenzbereich

Tab. 6.1.: Simulationsfiles zu Kap. 6

Experimente zu Kap. 6

Spielen Sie mit dem Skript herum und versuchen Sie folgende Fragen zu beantworten (keine Musterlösung, wird in der Vorlesung besprochen):

- Vollziehen Sie den Code nach (mit Hilfe der Hinweise oben).
- Welches maximale *SQNR* erwarten Sie bei den Quantisierungsparametern aus Listing 6.1? Warum ergibt die Simulation einen schlechteren Wert? Welchen maximalen Wert können

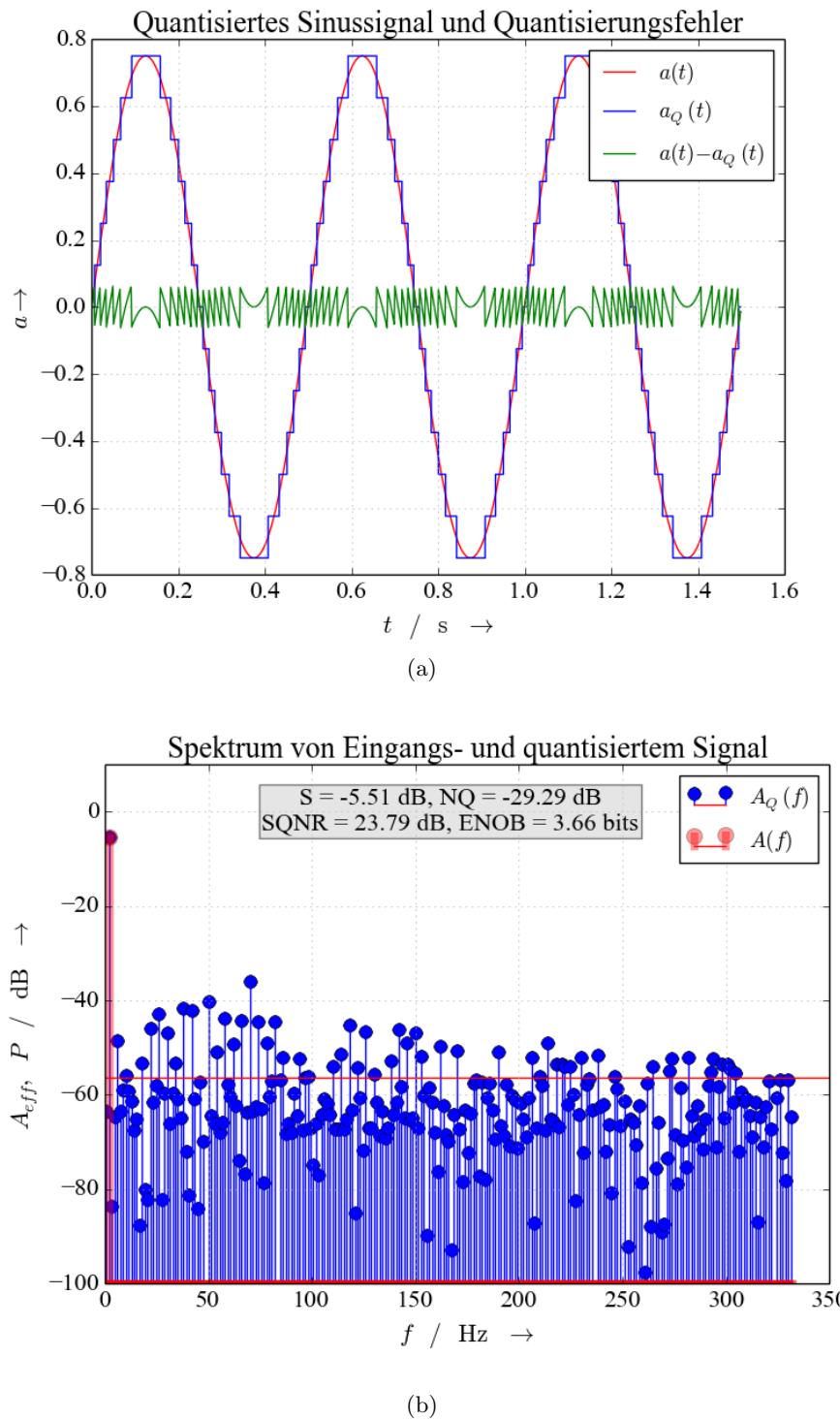


Abb. 6.3.: Auswirkung der Signalquantisierung (a) im Zeit- und (b) im Frequenzbereich (Listing 6.1)

Sie erreichen?

- Legen Sie ein Sinussignal mit der halben maximalen Amplitude an den Quantisierer. Welche Signalleistung erwarten Sie, welches *SQNR* und welche *ENOB*?
- Was passiert, wenn Sie den Quantisierer übersteuern? Vergleichen Sie die Optionen '*sat*' und '*wrap*'.
- Welchen Einfluss auf das Quantisierungsrauschen haben die Quantisierungsmethoden, also z.B. '*floor*', '*round*' und '*fix*'? Schauen Sie sich dabei auch den DC-Wert des Spektrums an.
- Erklären Sie, welchen Einfluss die Anzahl der dargestellten Signalperioden *N_per* im Code auf das dargestellte Spektrum hat.

6.1. * Quantisierungsrauschen bei Analog-Digital-Wandlung → M6.1

Ein ideal sinusförmiges Signal mit $f_1 < f_S/2$ wird mit einem 10 bit Analog-Digital Wandler mit einem Aussteuerbereich von $FSR = 2 \text{ V}$ (Full Scale Range) quantisiert. Eine anschließende Spektralanalyse der digitalen Daten ergibt einen *SNR* von 53 dB.

- a) Wie groß ist das **maximal erreichbare SNR** für sinusförmige Signale mit einem perfekten 10 bit ADC? Welcher effective number of bits (*ENOB*) entspricht das *SNR* von 53 dB?
- b) Welche Gründe könnte es geben für die Abweichung vom maximalen Wert? Nehmen Sie an, dass das Rauschen durch die Quantisierung dominiert wird und schätzen Sie die Amplitude *A* des Sinussignals ab aus der Abweichung zwischen maximalem und tatsächlichen S(Q)NR.
- c) Wie groß sind die **Rauschleistung N_Q** und die **Rauschleistungsdichte N'_Q** des voll ausgesteuerten Wandlers ($A = 1 \text{ V}$) wenn mit $f_{S1} = 10 \text{ kHz}$ bzw. $f_{S2} = 1 \text{ MHz}$ abgetastet wird?
- d) Eine 2048-Punkt FFT mit realem Wandler und sinusförmigem Eingangssignal ergibt das Spektrum in Abb. 6.4. Wie groß sind *SNR* und Amplitude des Signals? Wieviele effektive bits (*ENOB*, effective number of bits) hat der Wandler?

6.2. * SQNR eines quantisierten Signals → M6.2

- a) Wie groß ist das Signal-to-Quantization Noise Ratio (*SQNR*) eines sinusförmigen Eingangssignals, das mit 16 bit quantisiert wurde und das den Wertebereich voll ausnutzt?
- b) Wie groß ist das *SQNR*, wenn die Amplitude den Aussteuerbereich nur zu einem Zehntel ausnutzt?

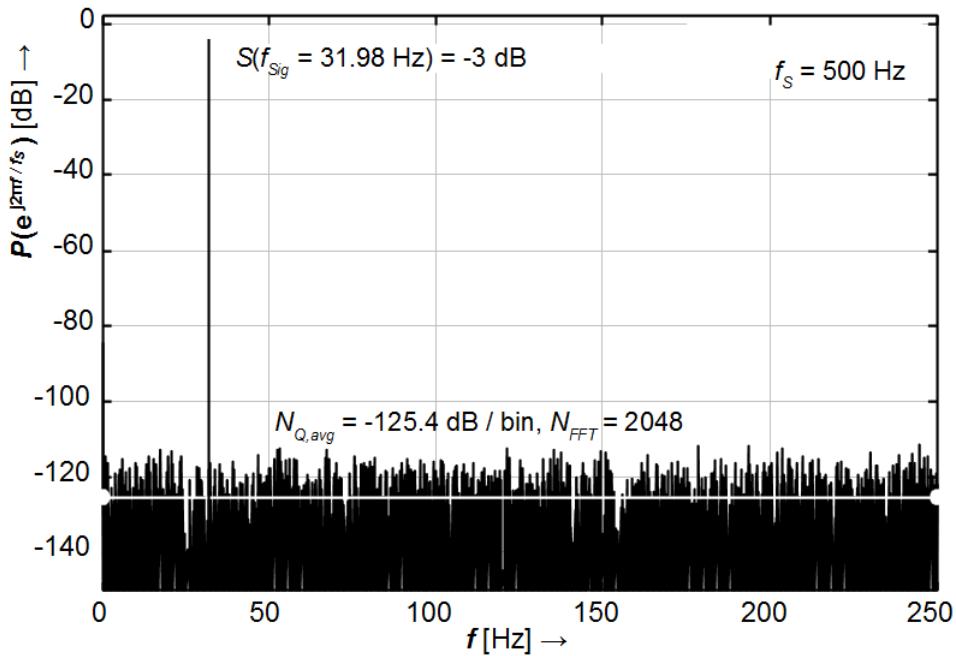


Abb. 6.4.: Spektrale Leistungsdichte am Ausgang eines ADCs

6.3. * Quantisierungsrauschen im FIR-Filter → M6.3

Es soll das Quantisierungsrauschen am Ausgang des Filters aus Aufgabe 5.3 untersucht werden. Die (Re-)Quantisierung erfolgt an allen Stellen durch Runden, der Ein- und Ausgangsspannungsbereich sei $\pm 1 \text{ V}$. Das Signal habe zunächst eine niedrige Frequenz F_1 , bei der gilt $H(F_1) = 1$ und eine Amplitude $A_{in} \approx A_{out} \approx 1$ (es tritt gerade eben noch keine Übersteuerung auf). Vernachlässigen Sie bei der Rechnung alle Rauschquellen außer den jeweils angegebenen. Bestimmen Sie Rauschleistung $N_{Q,Out}$, SNR und $ENOB$ am Ausgang für

- $Q_{DAC}(0.15)$: Requantisierung am Ausgang auf 16 bit vor dem DAC
- $Q_{ADC}(0.14)$: Quantisierung am Eingang durch 15 bit ADC
- $Q_{ADC}(0.14)$ und $Q_{DAC}(0.15)$: Quantisierung am Eingang durch 15 bit ADC und Requantisierung vor dem 16 bit DAC
- $Q_{Mul}(0.15)$: Requantisierung nach allen Multiplizierern auf 16 bit. Wie viele Multiplizierer werden bei diesem Filter wirklich verwendet?

Das Filter wird jetzt wie bei Unterpunkt c) mit einem 15 bit ADC $Q_{ADC}(0.14)$ am Eingang und einem 16 bit DAC $Q_{DAC}(0.15)$ am Ausgang betrieben; mit dieser Konfiguration wurde eine $ENOB = 15,1$ erzielt.

- Auf wieviele bits Q_{Mul} darf nach den Multiplikationen minimal gerundet werden, damit das hierdurch hervorgerufene Quantisierungsrauschen kleiner ist als von ADC und DAC? Welches SNR und welche $ENOB$ ergibt sich in diesem Fall durch die Wirkung aller (Re-)Quantisierer am Ausgang?
- Die Requantisierung nach der Multiplikation soll die $ENOB$ um maximal 0,1 bit verschlechtern. Wieviel Bits sind hierfür notwendig?

- g) Welches **SNR** erhält man für ein sinusförmiges Eingangssignal mit maximaler Amplitude und der Signalfrequenz $F_1 = 1/4$?

6.4. * DFT von Breitbandsignalen → M6.4

In dieser Aufgabe soll ein durch breitbandiges Rauschen gestörtes Signal mit dem Code in Listing 6.2 analysiert werden.

```

1 # ... Ende der Import-Anweisungen
2 f_S = 5e3; T_S = 1. / f_S
3 N_FFT = 1000; t_max = N_FFT * T_S
4 f_a = 1e3; f_b = 1e2;
5 A_a = 5; A_b = 1; NQ = 0.1
6 t = arange(0, t_max, T_S)
7 y = 1 + A_a * sin(2*pi*t*f_a) + A_b * cos(2*pi*t*f_b)
8 n = np.sqrt(NQ) * rnd.randn(len(t))
9 yn = y + n
10 Syn = fft(yn,N_FFT)
11 f = arange(N_FFT)
12 figure(1); clf()
13 plot(t, yn)
14 figure(2); clf()
15 plot(f,abs(Syn)); plt.show()
```

Lst. 6.2: DFT eines Breitbandsignals (erster Ansatz, NOI/NOI_DFT_wideband_py.py)

- a) Welche Leistung S hat das Nutzsignal, welche Leistung N das Rauschsignal in Listing 6.2? Wie groß ist das Signal-to-Noise Ratio SNR ?
- b) Wie groß ist die Rauschleistungsdichte N' ?
- c) Welche mittlere Leistung zeigt die DFT in der Simulation? Überlegen Sie dazu, welche Rauschleistung in jede DFT-Bin fällt.

7. SMP: Abtastung und Downsampling

Inhalt

Dieses Kapitel beschäftigt sich mit der Abtastung (Sampling) zeitkontinuierlicher Signale und dem Downsampling (Reduktion der Abtastrate, d.h. erneute Abtastung) zeitdiskreter Signale. Da man zeitkontinuierliche Signal betrachten kann wie ein Signal, das unendlich schnell abgetastet wurde, können beide Vorgänge mit den gleichen Methoden behandelt werden. Wesentliche Punkte sind dabei:

Aliasing: Wie werden Spektralkomponenten oberhalb der Nyquistfrequenz (= halbe Abtastfrequenz) auf das neue Basisband oder in andere Nyquistzonen abgebildet?

Anti-Aliasfilter: Wie legt man die Spezifikationen von Anti-Alias Filtern fest, um Aliasing zu verhindern?

Dezimation: Wie reduziere ich die Abtastrate, was ist dabei zu beachten?

Oversampling: Welche Vorteile kann durch „zu schnelle“ Abtastung eines zeitkontinuierlichen Signals erzielen?

Multiratenfilter: Wie kann ich Downsampling für den Entwurf effizienter Filter nutzen?

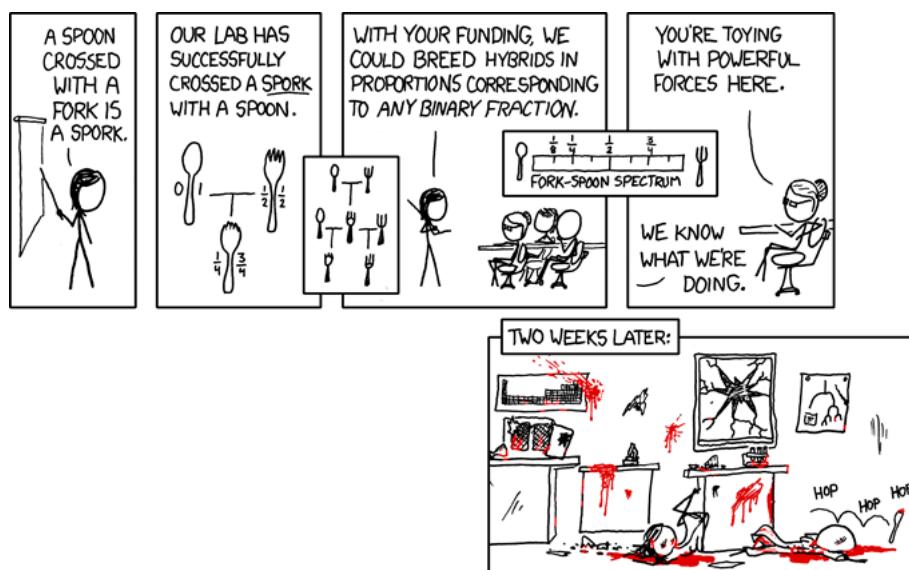


Abb. 7.1.: Abtasten ist gefährlich [<http://xkcd.com/419/>]

Python: Befehle und Programme zu Kap. 7

Für dieses Kapitel benötigt man Befehle, um jeden L -ten Wert eines Arrays zu entnehmen (*Downsampling*), mit Hilfe der Slicing Syntax von Python und Matlab geht das sehr kompakt und elegant. Optional kann die Startphase (also das erste Sample) n_0 spezifiziert werden.

Bei diesem Prozess kann Aliasing auftreten, daher wird üblicherweise vorher tiefpassgefiltert, den gesamten Prozess nennt man dann *Dezimation*.

Funktion	Python	Matlab	Beschreibung
Downsample	<code>y = x[::L]</code>	<code>y = x(1:L:N)</code>	Nimm nur jedes L -te Sample von x , beginnend mit Sample n_0
	<code>y = x[n0::L]</code>	<code>y = x(n0:L:N)</code>	
Dezimation	<code>y = sig.decimate(x, L, n=None, ftype='iir')</code>	<code>y = decimate(x, L [,N,'iir'])</code>	Dezimiere x um den Faktor L

Tab. 7.1.: Spezielle Befehle zu Kap. 7

Anmerkungen zu Tab. 7.1:

downsample Achtung: Unterschiedliche Reihenfolge der Slicing-Parameter bei Python und Matlab; das erste Element eines Arrays hat in Python den Index 0, in Matlab den Index 1!

decimate verbindet Anti-Alias-Filterung und Downsampling miteinander. Die Angabe der Filterordnung N und von 'iir' bzw. `ftype = 'iir'` ist optional. Defaultmäßig wird ein Chebychev-Filter Typ I (IIR) achter Ordnung verwendet. Hier wird die Sequenz vorwärts und rückwärts gefiltert (siehe `filtfilt` bzw. `sig.filtfilt()`) und so eine lineare Phase (und die doppelte Ordnung) erzielt. Da die komplette Sequenz hierfür im Speicher stehen muss, ist diese Methode für lange Sequenzen schlecht geeignet.

Wählt man mit 'fir' bzw. `ftype = 'fir'` ein FIR-Filter, wird defaultmäßig ein Filter 30. Ordnung mit Hamming-Fenster verwendet. Dieses Filter ist linearphasig und wird nur in einer Richtung durchlaufen, die Sequenz muss sich daher nicht komplett im Speicher befinden und kann daher beliebig lang sein.

Achtung: Die Filter haben Ripple, daher können bei einzelnen Frequenzen Überhöhungen auftreten!

Filename	Beschreibung
<code>SMP_sampled_sine</code>	Aliasing: Plotte mehrere sinusförmige Signale mit unterschiedlicher Frequenz und identischer Abtastfunktion
<code>SMP_pyaudio-decimation</code>	Code-Beispiel zur Dezimation (mit / ohne Filterung) von Audio-Signalen

Tab. 7.2.: Simulationsfiles zu Kap. 7

7.1. * Spektren abgetasteter Signale → M7.1

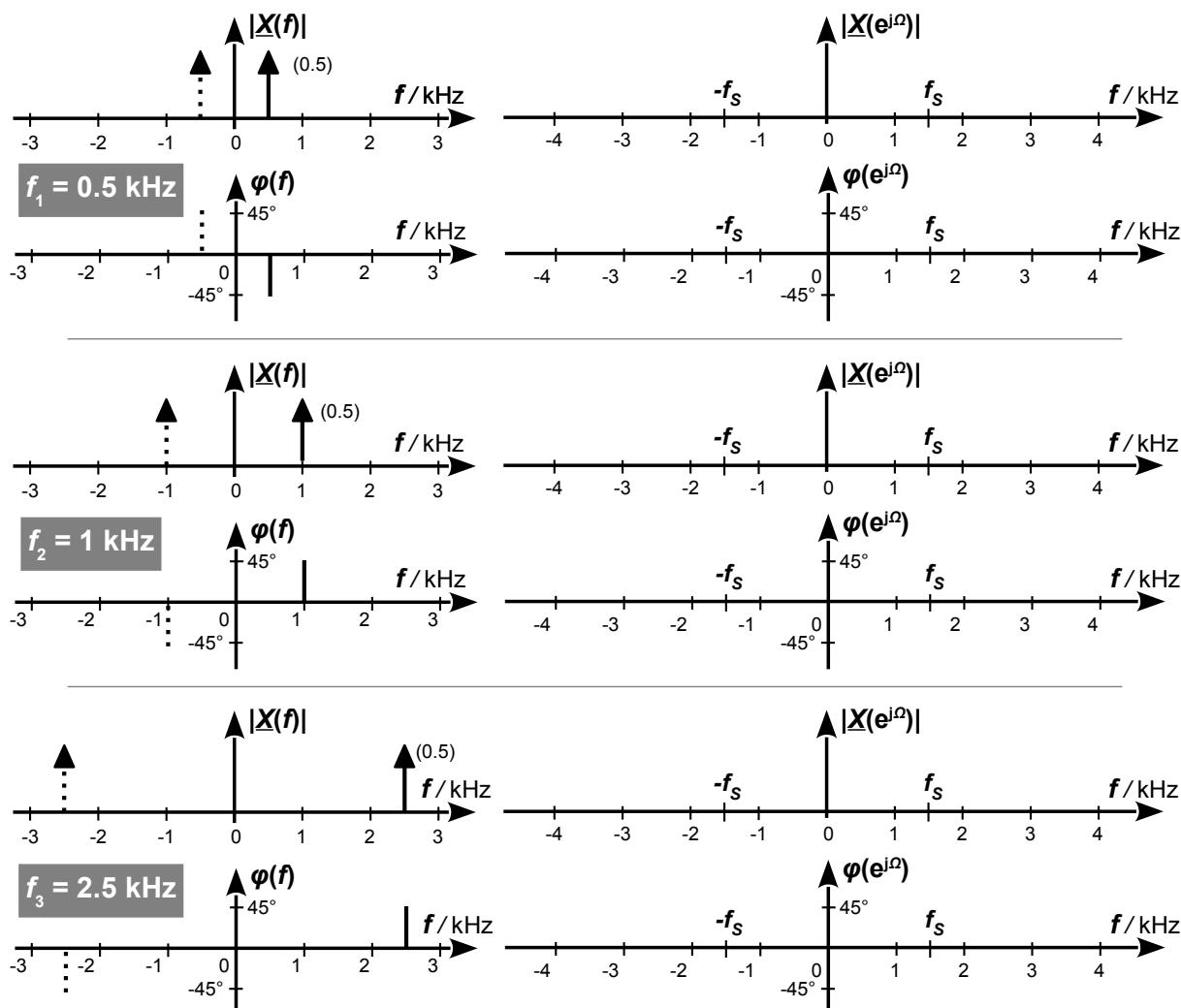


Abb. 7.2.: Amplituden- und Phasenspektren von zeitkontinuierlichen und abgetasteten Signalen zur Aufgabe 7.1.

Auf der linken Seite von Abb. 7.2 sind die Betrags- und Phasenpektrum der drei zeitkontinuierlichen Signale $s_i(t) = \cos(2\pi f_i t + \varphi_i)$
mit $f_1 = 500 \text{ Hz}$, $f_2 = 1000 \text{ Hz}$, $f_3 = 2500 \text{ Hz}$, $\varphi_1 = -45^\circ$, $\varphi_2 = 45^\circ$, $\varphi_3 = 45^\circ$ aufgetragen.

Diese Signale werden jetzt mit $f_S = 1500 \text{ Hz}$ abgetastet. Skizzieren Sie auf der rechten Seite von Abb. 7.2 die Spektren der resultierenden zeitdiskreten Signale. Beschränken Sie sich dabei auf die Komponenten, die durch $\pm 2f_S$, $\pm f_S$ und $f = 0$ entstehen und markieren Sie jeweiligen Spektrallinien farbig!

7.2. Analog-Digital-Wandlung mit Oversampling → M7.2

Ein sinusförmiges Signal wird mit 8 bit Auflösung und der Abtastrate $f_S = 1 \text{ MHz}$ abgetastet, es werde der volle Aussteuerbereich ausgenutzt.

- a) Wie groß ist das maximale SQNR bei Nyquist-Wandlung (maximale Nutzbandbreite f_N)?

- b) Wie groß ist das maximale SQNR bei Oversampling mit $OSR = 10$ und bei $OSR = 50$ und idealer Tiefpassfilterung? Welcher Effective Number of Bits (ENOB) entsprechen die SQNR-Werte?
- c) Welches SQNR lässt sich erzielen mit einem Sigma-Delta-Wandler (ein Bit) erster oder zweiter Ordnung, $OSR = 10$ und $OSR = 50$ und idealer Tiefpassfilterung?

7.3. * Analog-Digital-Wandlung mit Oversampling und nachfolgender Dezimation → M7.3

In einem System A soll ein analoges Signal mit $B = 12$ kHz Bandbreite mit einem idealen ADC mit $f_{S1} = 100$ kHz Abtastrate gewandelt werden.

Alternativ soll das System B verwendet werden mit einem idealen ADC mit $f_{S2} = 1$ MHz Abtastrate und nachfolgender Dezimation um den Faktor $R = 10$. Dabei werden immer 9 aufeinander folgende Samples verworfen. Zwischen ADC und Dezimation wird kein digitales Anti-Antialias / Dezimationsfilter verwendet.

- a) Sind die Ausgangssignale von System A und B identisch? Wird das Spektrum des Signals in System B auf diese Art verfälscht?
- b) Welcher maximale ganzzahlige Dezimationsfaktor R_{max} kann in System B gewählt werden, ohne das Spektrum zu verfälschen?
- c) Warum ist im Allgemeinen ein digitales Filter vor der Dezimationsstufe erforderlich?
- d) Welche Vorteile hat die Kombination von Überabtastung und nachfolgender Dezimation?

7.4. * Zweistufige Dezimation → M7.4

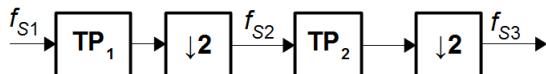


Abb. 7.3.: Zweistufige Dezimation zu Aufgabe 7.4

Ein Analogsignal wird mit vierfacher Überabtastung digitalisiert. Die Abtastfrequenz soll in zwei Stufen um den Faktor 4 reduziert werden (Dezimation). Die Eckfrequenz des Durchlassbands des gesamten Systems in Abb. 7.3 soll ein Drittel der Abtastfrequenz am Ausgang f_{S3} betragen, dies ist auch die Nutzbandbreite $f_N = f_{S3}/3 = f_{S1}/12$ des Eingangssignals. Das Spektrum des Eingangssignals kann als weiß betrachtet werden, daher ist bei der Dezimation auf ausreichende Anti-Alias Filterung zu achten.

- a) Zunächst soll gestattet sein, dass bei beiden Dezimationsstufen Aliasing ins Basisband (aber nicht ins Nutzband!) auftritt. Beide Dezimationsfilter (= Anti-Aliasfilter) sollen als (dezimierende) Halbbandfilter realisiert werden.
- b) Jetzt soll das Basisband nach der zweiten Dezimation frei von Aliasing-Komponenten sein, in der ersten Dezimationsstufe ist Aliasing erlaubt. Das erste Dezimationsfilter soll wieder als Halbbandfilter, das zweite als allgemeines FIR-Filter realisiert werden.

Berechnen Sie für beide Fälle die Eckfrequenzen von Durchlass- und Sperrband der Teilfilter!

Hinweise:

- Am Besten gehen Sie vom Ausgang des Filters aus und überlegen sich, welche Bedingung Ihre Teilfilter jeweils erfüllen müssen, damit am Ausgang kein Aliasing sichtbar wird.
- Halbbandfilter sind Filter, deren Frequenzgang punktsymmetrisch zu $|H(f = f_S/4)| = 0,5 \hat{=} -6 \text{ dB}$ ist. Die Eckfrequenzen von Durchlass- und Sperrband können daher nicht unabhängig voneinander gewählt werden, $f_{DB} = f_S/2 - f_{SB}$. Dezimierende Halbbandfilter verbinden Filterung und Downsampling um den Faktor 2 und nutzen bei der Implementierung aus, dass jeder zweite Koeffizient (bis auf den mittleren) gleich Null ist, sie werden daher gerne in Dezimierungsfiltern genutzt, die mit hoher Taktfrequenz arbeiten müssen.

7.5. Multiraten-Tiefpassfilter → M7.5

In Abb. 7.4 ist ein zweistufiges Multiraten-Tiefpassfilter dargestellt mit den idealisierten¹ Frequenzgängen der Tiefpässe TP_1 und TP_2 . Die Abtastfrequenz am Eingang beträgt $f_{S1} = 48 \text{ kHz}$, beachten Sie die unterschiedlichen Abtastfrequenzen und die endliche Sperrdämpfung der Tiefpässe!

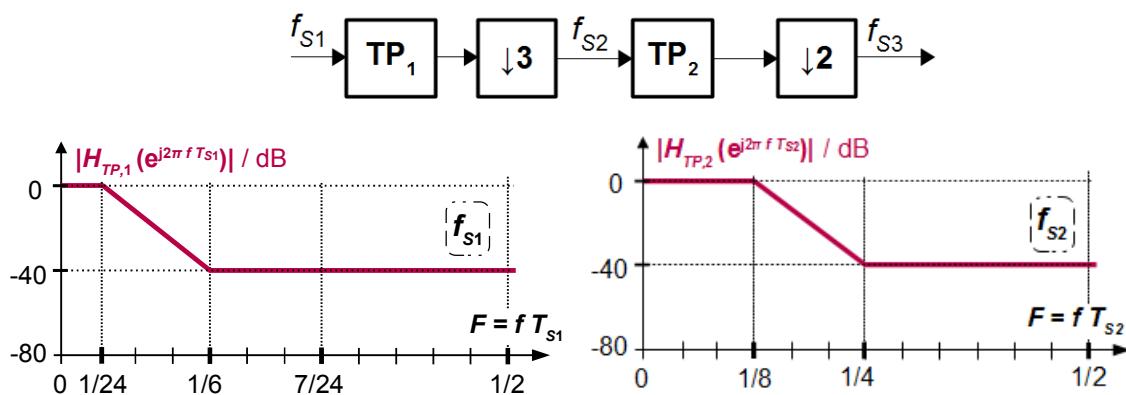


Abb. 7.4.: Multiraten-Tiefpassfilter und Betragsgang der Teilfilter zu Aufgabe 7.5

- Skizzieren Sie den Amplitudengang der gesamten Filterkaskade im Bereich $f = 0 \dots f_{S1}/2$. Bestimmen Sie die Dämpfung der gesamten Kaskade bei der Eckfrequenz des Durchlass- und des Sperrbandes und tragen Sie A_{DB} und A_{SB} in Tab. 7.3 ein.
- Am Eingang werden jetzt ein sinusförmige Testsignale mit den Frequenzen f_1 , f_2 und f_3 angelegt (siehe Tab. 7.3). Berechnen Sie die Frequenzen $f_{out,i}$, bei denen die Testsignale aufgrund von Aliasing am Ausgang der Filterkaskade erscheinen sowie deren Dämpfung und tragen Sie die Daten in Tab. 7.3 ein.
- Ändern Sie die Spezifikationen von TP_1 und / oder TP_2 so ab, dass die gestellten Anforderungen für das Gesamtfilter erfüllt werden.

¹d.h. Durchlass-/Übergangs-/Sperrbereich sind nur schematisch dargestellt

f_{in}	f_{DB}	f_{SB}	f_1	f_2	f_3
	2 kHz	4 kHz	8 kHz	14 kHz	16 kHz
f_{out} (kHz)	2	4			
	A_{DB}	A_{SB}		$A_{al,i}$	
Soll (dB)	0,1	80	80	80	80
Ist (dB)					

Tab. 7.3.: Eigenschaften der Filterkaskade aus Aufgabe 7.5

8. INP: Upsampling, Interpolation und Digital-Analog Wandlung

Inhalt

Dieses Kapitel beschäftigt sich mit der Erhöhung der Abtastrate bei zeitdiskreten Systemen und mit der D/A Wandlung. Beide Prozesse kann man mit den gleichen Methoden behandeln, da man ein analoges Signal als unendlich schnell abgetastet betrachten kann ($T_S \rightarrow 0$):

Upsampling: Erhöhen der Abtastrate, normalerweise durch **Nullenstopfen**, d.h. Auffüllen der fehlenden Werte durch Nullen.

Image: Wenn die Abtastrate durch Nullenstopfen von f_S auf $I f_S$ erhöht wird, liegen im neuen Basisband $-I f_S/2 \dots I f_S/2$ Kopien des alten Basisbands $-f_S/2 \dots f_S/2$, die sich im Gegensatz zu Wiederholsspektrum unabhängig beeinflussen lassen.

Interpolation: Ergänzen von Datenpunkten, um die Auflösung zu erhöhen. Im Zeitbereich entspricht das der „Glättung“ der Daten, im Frequenzbereich einer Dämpfung der Images. Bei **idealer Interpolation** werden alle Images unterdrückt, ohne das ursprüngliche Basisband-Spektrum zu beeinflussen.

Rekonstruktion: Wiederherstellen der „ursprünglichen“ Signalform aus dem abgetasteten Signal - eigentlich das Gleiche wie Interpolation, aber meist bezogen auf die Wiederherstellung von analogen Signalen.

Zero-Order-Hold (ZOH) ist die einfachste (und schlechteste) Form der Interpolation zwischen benachbarten Werten - es wird einfach der letzte Wert solange wiederholt (DT-Systeme) bzw. gehalten (CT-Systeme), bis ein neuer Wert zur Verfügung steht. Diese Operation entspricht im Zeitbereich der Faltung mit einem DT- bzw. CT-Rechteckpuls und im Frequenzbereich der Multiplikation mit einer si-Funktion bzw. einer periodischen si-Funktion (Dirichlet-Kernel, bei DT-Systemen).

<http://demonstrations.wolfram.com/MultirateSignalProcessingUpsampling/>

Theorie

Ein hypothetischer idealer DAC würde analoge Diracstöße produzieren mit einem Gewicht (= Fläche) $s[n]\delta(t-nT_S)$, proportional zum zu wandelnden Digitalwert $s[n]$. Das Ausgangsspektrum eines solchen idealen DACs sähe genauso aus wie das Wiederholsspektrum des digitalen Signals, wiederholt bis zu unendlich hohen Frequenzen. Da Diracstöße eine zeitliche Ausdehnung von 0 und eine unendlich große Amplitude aufweisen, ist diese Art der Wandlung aber technisch nicht realisierbar.

Stattdessen wird in der Praxis eine endliche analoge Spannung generiert, die proportional zum Digitalwert ist, und für die Dauer einer Abtastperiode T_S gehalten (Zero-Order Hold, ZOH). Diese Rechteckfläche kann z.B. den gleichen Wert haben wie die Fläche des Diracstoßes, aber das Spektrum ist unterschiedlich: Die Zero-Order Hold (ZOH) Charakteristik des DACs dämpft das Wiederholspektrum mit einer si-Funktion, deren erste Nullstelle bei $f_{0,min} = f_S$ liegt. Die maximale Dämpfung a_N im Nutzband hat man daher an der oberen Kante des Nutzbands bei f_N .

Python: Befehle und Programme zu Kap. 8

Für dieses Kapitel benötigt man vor allem einen Python- bzw. Matlab-Befehl, um zwischen jeden Wert eines Arrays I Nullen zu „stopfen“ oder jeden Wert I mal zu wiederholen (Zero-Order Hold). Das geht mit Hilfe der Slicing-Schreibweise (siehe Python Kurzanleitung) sehr kompakt und elegant:

Funktion	Python	Matlab	Beschreibung
Upsampling (Zero-Stuffing)	(siehe Listing 8.1)	<code>y = upsample(x,I [,n0])</code>	Füge $I-1$ Nullen zwischen Samples ein
Upsampling (ZOH)	<code>y = np.repeat(x, I)</code>	<code>y = kron(x, ones(1,I))</code>	Wiederhole jedes Sample $I-1$ mal
Interpolation	siehe <code>scipy.interpolate</code>	<code>y = interp(x,r[,n,a])</code>	

Tab. 8.1.: Spezielle Befehle zu Kap. 8

Anmerkungen zu Tab. 8.1:

`upsample` kann auch in Python effizient mit `np.insert` durchgeführt werden.

`kron` ist das Kronecker-Produkt; in Python könnte man genauso schreiben `y = np.kron(x, np.ones(I))`, das ist aber deutlich langsamer als `repeat` oder `tile`.

```

1 import numpy as np
2 a_in = np.arange(1,4) # initial array
3 I = 3; n_0 = 0 # Upsampling factor & start phase
4 # create array of zeros with size a_in * I:
5 a_up = np.zeros(I * a_in.size, dtype=a_in.dtype)
6 # fill in elements of a_in every Ith position:
7 a_up[n_0::I] = a_in
8 #
9 #
10 # Upsampling with ZOH:
11 a_ZOH = np.repeat(a_in,I)
12 
```

```

% 
a_in = [1:3];
I = 3; n_0 = 1
%
a_up = zeros(1, I * length(a_in));
%
a_up(n_0:I:length(a_up)) = a_in;
% alternativ:
a_up = upsample(a_in, I, n_0)
% Zero-Order Hold:
% allocate memory for speed-up:
a_ZOH = zeros(1, I*length(a_in)); 
```

```

13 >>> a_in
14 array([1, 2, 3])
15
16 >>> a_up
17 array([1, 0, 0, 2, 0, 0, 3, 0, 0])
18
19 >>> a_ZOH
20 array([1, 1, 1, 2, 2, 2, 3, 3, 3])

```

Lst. 8.1: Upsampling mit Python

```

a_ZOH = kron(a_in, ones(1,I));
%
%
%
%
%
%
%
```

Lst. 8.2: Upsampling mit Matlab

Soll nicht nur die Abtastrate erhöht werden (Upsampling), sondern auch gleich interpoliert (= Zwischenwerte berechnet bzw. Images entfernt) werden, kann man `a_ip = interp(a_in, I)` (Matlab) verwenden. In Python gibt es hierfür `interp1d()` aus `scipy.interpolate` (siehe <http://docs.scipy.org/doc/scipy/reference/tutorial/interpolate.html>). `np.interp()` kann nur linear interpolieren und ist daher schlecht geeignet.

In Matlab entwirft man mit dem Befehl `b = intfilt(I, p, alpha)` ein linearphasiges Tiefpassfilter für die ideale bandbegrenzte Interpolation einer Sequenz mit Zerostuffing. Alternativ entwirft `intfilt(I, N, 'Lagrange')` ein Filter, das Lagrange-Interpolation der Ordnung `N` durchführt.

8.1. * Wiederholspektren und Images → M8.1

Ein digitales Audiosignal $s[n]$ mit Abtastrate $f_{S1} = 48$ kHz und Nutzbandbreite $f_N = 12$ kHz hat das Basisbandspektrum $S(e^{j2\pi f T_{S1}})$ in Abb. 8.1(a)).

- a) Ergänzen Sie die Wiederholspektren in Abb. 8.1(a) und kennzeichnen Sie das Basisband. **ML unvollständig!**
- b) Die Abtastrate von $s[n]$ wird jetzt durch Nullenstopfen um den Faktor $I = 3$ erhöht. Ergänzen Sie die Wiederholspektren und Images in Abb. 8.1(b), kennzeichnen und bezeichnen Sie beide durch eine eindeutige Schraffur. Tragen Sie die Grenzen des Basisbands ein. **ML unvollständig!**

8.2. * Verschiedene Arten der Digital-Analog-Wandlung → M8.2

Ein digitales Audiosignal $s[n]$ mit Abtastrate $f_{S1} = 48$ kHz und Nutzbandbreite $f_N = 12$ kHz (Basisbandspektrum siehe Abb. 8.1(a)) soll auf verschiedene Arten analog rekonstruiert werden. Die eigentliche D/A Wandlung erfolgt dabei immer mit einfachem analogen Halteglied (Zero-Order Hold):

- a) **Direkte D/A-Wandlung** des digitalen Audiosignals bei der Abtastfrequenz f_{S1} .
- b) **Überabtastung um den Faktor $I = 3$ ohne digitale Interpolation**
Das digitale Audiosignal wird zunächst um den Faktor 3 überabgetastet, zwischen den Signalwerten werden dabei Nullen eingefügt (Nullenstopfen). Die hierdurch reduzierte Leistung im Basisband wird durch Multiplikation um den Faktor 3 ausgeglichen, es findet

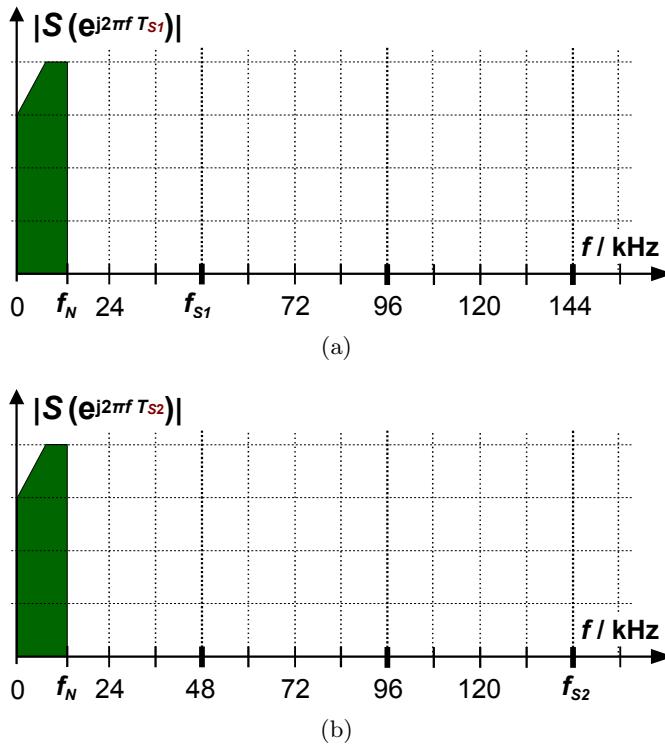


Abb. 8.1.: Spektrum von $s[n]$ vor (a) und nach Abtastratenerhöhung um $I = 3$ (b)

aber keine digitale Interpolation statt. Die D/A Wandlung findet bei der Abtastfrequenz $f_{S2} = 3f_{S1}$ statt.

c) **Überabtastung um den Faktor $I = 3$ mit idealer digitaler Interpolation**

Nach Überabtastung um den Faktor $I = 3$ und Nullenstopfen ist jetzt ein ideales Interpolationsfilter mit Amplitudenkorrektur geschaltet. Die D/A Wandlung findet bei der Abtastfrequenz $f_{S2} = 3f_{S1}$ statt. Was heißt ideale Interpolation? Bestimmen Sie die Eckfrequenzen des Durchlass- und des Sperrbandes für das Interpolationsfilter (möglichst breiter Übergangsbereich).

d) **Überabtastung um den Faktor $I = 3$ mit digitalem Zero-Order Hold**

Nach Überabtastung um den Faktor $I = 3$ und Nullenstopfen wird ($I - 1$) mal der letzte Wert wiederholt (digitales Zero-Order Hold). Die D/A Wandlung findet bei der Abtastfrequenz $f_{S2} = 3f_{S1}$ statt. Wie kann man die Wiederholung des letzten Wertes durch ein einfaches Filter beschreiben?

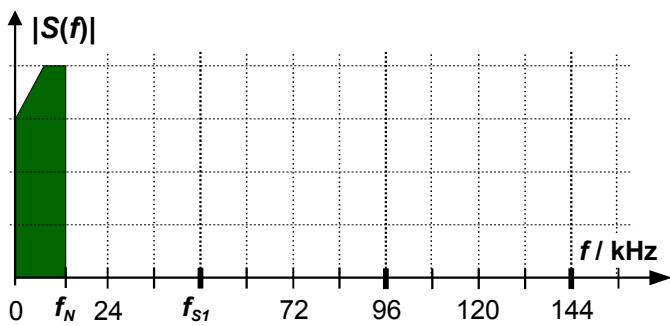
e) Vergleichen Sie die Spektren aus a) - d) und erläutern Sie Unterschiede und Gemeinsamkeiten.

Skizzieren Sie für jeden Fall das Blockschaltbild mit Abtastfrequenzen und ggf. Eckfrequenzen des Filters. Tragen Sie in die Diagramme und Tabellen auf der nächsten Seite ein:

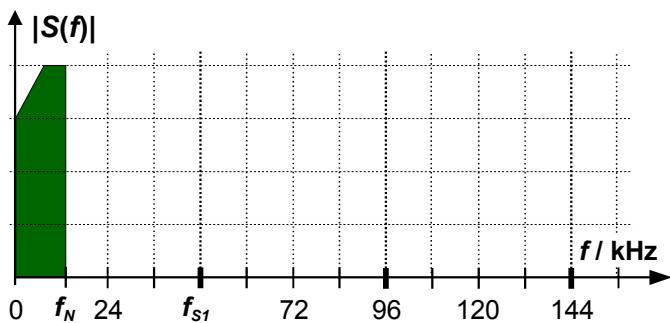
- das **Spektrum** des Audiosignals nach der D/A-Wandlung mittels schraffierter Flächen
- die maximale Dämpfung a_N in dB im Nutzband (bei welcher Frequenz tritt sie auf? warum?)
- die **niedrigste Imagefrequenz** f_{im} und die Dämpfung in dB $a_{im}(f_{im})$ bei dieser Frequenz.

- für Fall d) zusätzlich die Dämpfung des digitalen Interpolationsfilters im Nutzband $a_{N,I}$ und bei der niedrigsten Imagefrequenz $a_{im,I}$

a) Direkte D/A Wandlung

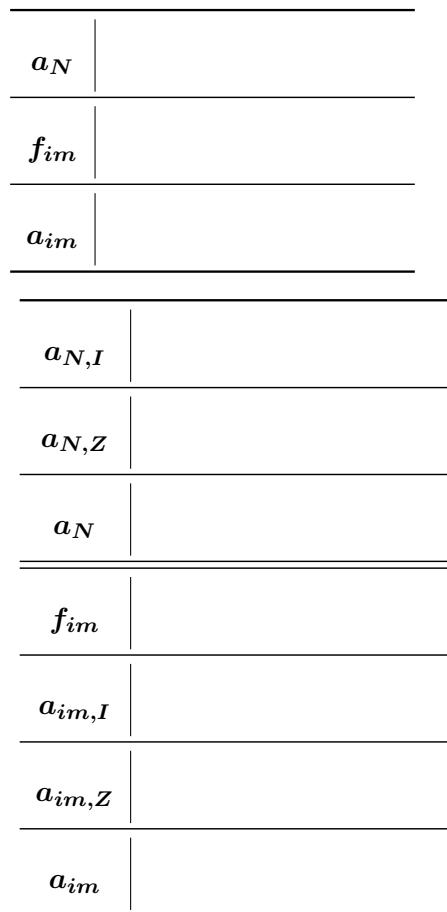
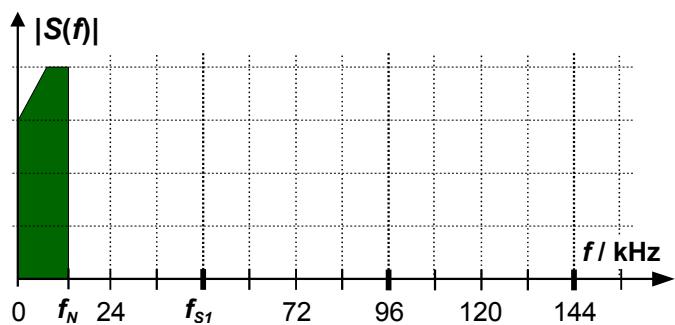
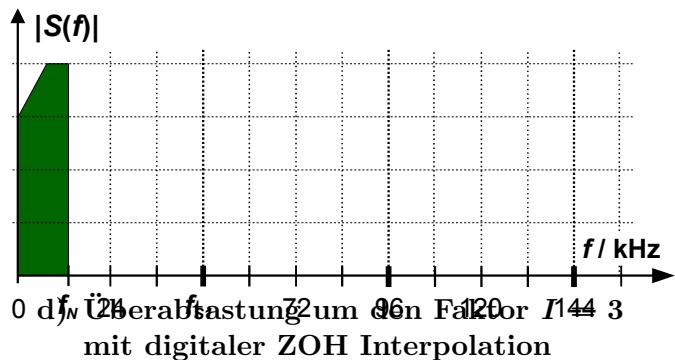


a_N	
f_{im}	
a_{im}	

b) Überabtastung um den Faktor $I = 3$ ohne Interpolation

a_N	
f_{im}	
a_{im}	

c) Überabtastung um den Faktor $I = 3$ mit idealer Interpolation



8.3. * Ideales Interpolationsfilter → M8.3

Wahr oder falsch? Ein ideales Interpolationsfilter ...

- filtert alle Wiederholspektren und Images aus dem Basisband
- filtert alle Wiederholspektren und Images aus dem Nutzband
- hat möglichst weit auseinanderliegende Eckfrequenzen von Sperr- und Durchlassband
- entfernt alle Images
- lässt das DB unverändert
- hat keinen Einfluss auf Wiederholspektren
- lässt Images unverändert
- verhindert Rückfaltungen ins Basisband
- verhindert Rückfaltungen ins Nutzband

- muss im Durchlassband die Verstärkung 1 haben

8.4. * Abtastratenerhöhung mit Nullenstopfen → Aufgabe M8.4

Zeigen Sie mit Hilfe der DFT-Definitionsgleichung, dass eine Erhöhung der Abtastfrequenz mit Nullenstopfen das Spektrum des Nutzbands wiederholt.

$$S[k] = \frac{1}{N} \sum_{n=0}^{N-1} s[n] e^{-j2\pi kn/N} = \frac{1}{3} \sum_{n=0}^2 s[n] e^{-j2\pi kn/3}$$

8.5. Oversampling DAC → Aufgabe M8.5

Ein mit $f_{s1} = 8$ kHz abgetastetes Sprachsignal im Telefonnetz (Nutzband 0 . . . 3,3 kHz) soll zeitkontinuierlich mit einem Zero-Order-Hold (ZOH) - DAC und analogem Tiefpassfilter H_{RKTP} rekonstruiert werden. Als Testsignal wird ein ideal abgetastetes Sinussignal mit $f_{sig} = 3,3$ kHz und $\hat{u} = 5$ V verwendet. Eine Realisierung ohne Oversampling (Abb. 8.2(a)) und eine mit (Abb. 8.2(b)) sollen hinsichtlich der si-Verzerrungen des Testsignals verglichen werden, Imagekomponenten von y_s sollen Amplituden unter 50 mV haben.

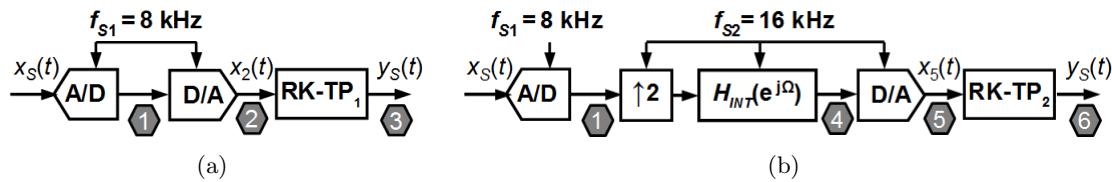


Abb. 8.2.: DAC ohne (a) und mit (b) Oversampling zu Aufgabe 8.5

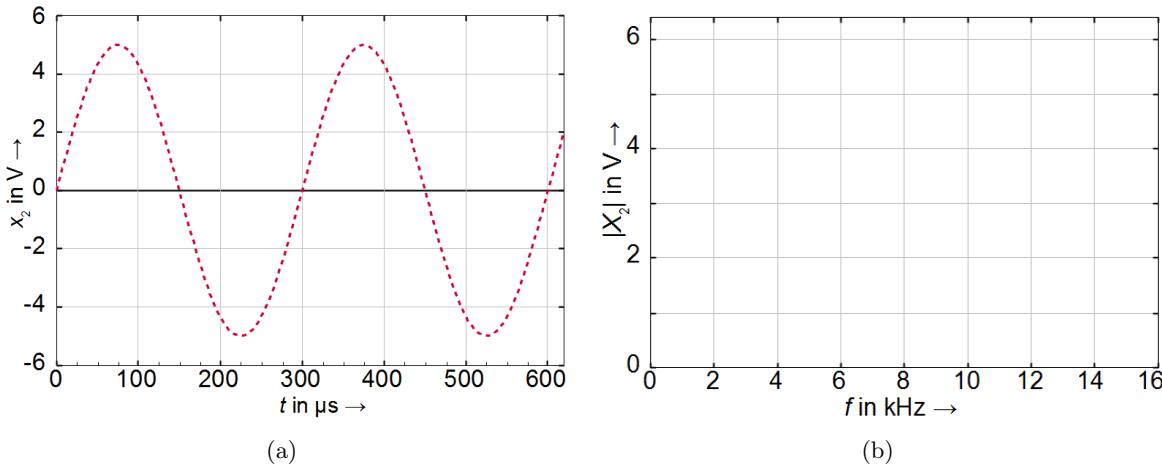


Abb. 8.3.: ZOH-Signal $x_2(t)$ und Amplitudenspektrum $|X_2(f)|$ zu Aufgabe 8.5a

- Skizzieren Sie in Abb. 8.3a) das Signal $x_2(t)$ nach dem ZOH-DAC und in Abb. 8.3b) das dazugehörige Amplitudenspektrum $X_2(f)$. Bei welcher Frequenz tritt das erste Image auf? Wie stark werden das Nutzsignal und das erste Image durch die si-Charakteristik der ZOH-Stufe gedämpft?

- b) Skizzieren Sie in Abb. 8.4a) die Spezifikationen für den Interpolations-Tiefpass aus Abb. 8.2(b)), der als Halbband-FIR-Filter entworfen werden soll. Welche Vorteile bietet eine Implementierung als Halbband-Filter?
 - c) Skizzieren Sie in Abb. 8.5a) das Signal $x_5(t)$ nach dem ZOH-DAC, vernachlässigen Sie dabei die Gruppenlaufzeit des Interpolationsfilters. Skizzieren Sie in Abb. 8.5b) das dazugehörige Amplitudenspektrum $X_5(f)$ bis $f_{S2} = 2f_{S1}$. Bei welcher Frequenz tritt das erste Image auf? Wie stark werden das Nutzsignal und das erste Image durch die si-Charakteristik der ZOH-Stufe gedämpft?
 - d) Skizzieren Sie in Abb. 8.4b) die Spezifikationen für die analogen Rekonstruktions-Tiefpässe $H_{RKTP,1}$ und $H_{RKTP,2}$ aus Abb. 8.2a) und b).

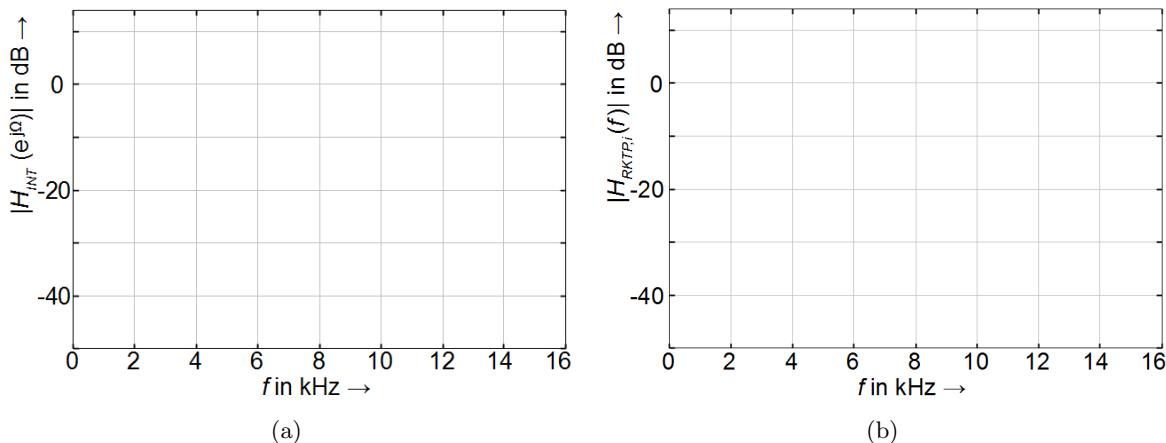


Abb. 8.4.: Spezifikationen für Interpolationsfilter H_{int} (a) und Rekonstruktionstiefpässe $H_{RKTP,i}$ (b) zu Aufgabe 8.5b und d)

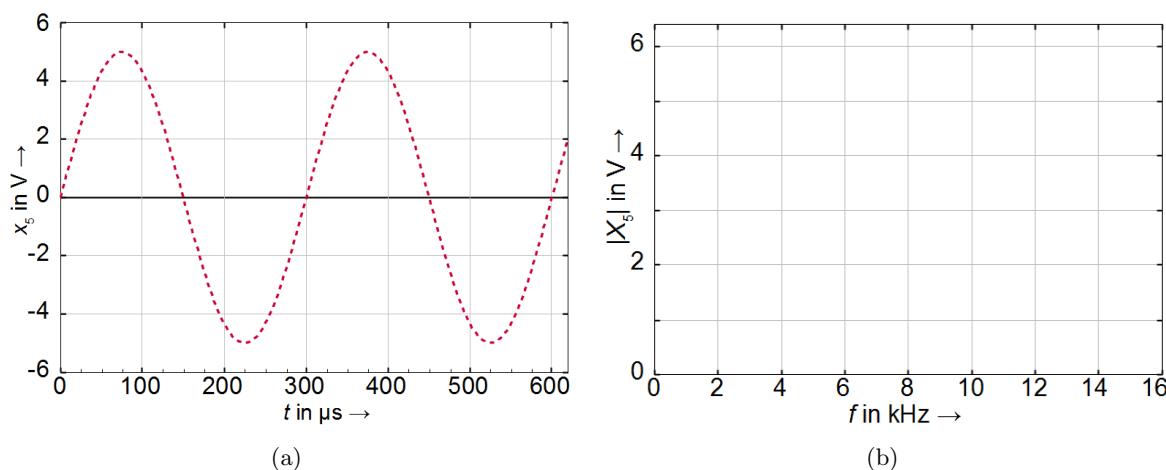


Abb. 8.5.: ZOH-Signal $x_5(t)$ und Amplitudenspektrum $X_5(f)$ zu Aufgabe 8.5c

8.6. Zweistufige Interpolation → M8.6

Abb. 8.6 zeigt die zu Aufgabe 7.4 transponierte zweistufige Interpolation, die Tiefpassfilter sind identisch zu denen aus Aufgabe 7.4.

- Skizzieren Sie den Amplitudengang der Kaskade im logarithmischen Maßstab.
- Am Eingang wird eine Sinusschwingung der Frequenz $f_{S3}/4$ und der Amplitude 10 dBV (d.h. bezogen auf eine Referenzspannung von $1 \text{ V}_{\text{eff}} \hat{=} 0 \text{ dBV}$) angelegt. Berechnen Sie die Frequenzen aller Komponenten am Ausgang der Kaskade im Basisband ($0 \dots f_{S1}/2$ und deren Dämpfung bezogen auf den Eingangspegel).

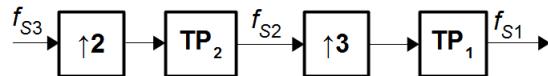


Abb. 8.6.: Zweistufige Interpolation zu Aufgabe 8.6

9. SRC: Abtastratenwandlung

Inhalt

Dieses Kapitel wirft die Ergebnisse der letzten beiden Kapitel in einen Topf, es geht darum die Abtastrate eines Signals um einen rationalen Faktor wie z.B. $7/8$ zu ändern. Wichtiges Thema dabei ist die effiziente Filterung, um Aliasing und unerwünschte Images zu unterdrücken.



Abb. 9.1.: Wenn Ingenieure blitzen [<http://xkcd.com/165/>]

Hinweise:

- Bei normierten Frequenzen in Multiratensystemen muss immer klar sein, auf welche Abtastfrequenz man sich gerade bezieht!
- Sofern nicht anders angegeben, soll Interpolation durch Nullenstopfen erfolgen. Die Korrektur der Verstärkung um den Interpolationsfaktor I erfolgt im nachfolgenden Tiefpass.
- Die Eckfrequenz des Sperrbands soll so *hoch* gewählt werden, dass das Signal gerade noch nicht durch Aliasing oder Images verfälscht wird.
- Die Eckfrequenz des Durchlassbands soll so *niedrig* gewählt werden, dass die Nutzsignalbandbreite gerade noch nicht beschnitten wird.
- Es wäre auch möglich, eine niedrigere Eckfrequenz des Sperrbandes oder eine höhere Eckfrequenz des Durchlassbands zu wählen, aber:

Für einen möglichst entspannten Tiefpassentwurf müssen die Eckfrequenzen von Durchlass- und Stoppband möglichst weit auseinanderliegen (breiter Übergangsbereich)!

Anmerkung: Ein noch entspannterer Filterentwurf ist möglich, wenn Aliasing im Übergangsbereich zulässig ist - dann kann die Eckfrequenz des Sperrbandes noch etwas größer gewählt werden.

Python: Befehle und Programme zu Kap. 9

Funktion	Python	Matlab	Beschreibung
resample	<code>y = sig.resample(x, num, t=None, window=None)</code>	<code>y = resample(x, p, q [, n, beta])</code>	Bringe eine Datensequenz auf eine neue feste Abtastfrequenz.
upfirdn	- - -	<code>y = upfirdn(x,h,p,q)</code>	Upsampling von x um p , Filtern mit h , Downsampling um q

Tab. 9.1.: Spezielle Befehle zu Kap. 9

Anmerkungen zu Tab. 9.1:

Resample (Python): x wird auf num Abtastwerte mit Hilfe der Fourier-Methode entlang der gewählten Achse neu abgetastet. Bei langen Sequenzen sollte man darauf achten, eine Zweierpotenz zu wählen, um die Berechnung der FFT zu beschleunigen! Die Methode ist am besten geeignet für periodische Signale - der letzte und der erste Wert der neu abgetasteten Sequenz „passen zusammen“ und müssen ggf. abgeschnitten werden.

Optional können angegeben werden: t ist ein optionales Array mit den Zeitpunkten bzw. Positionen von x (muss gleiche Länge haben und äquidistant sein, dann wird ein Tuple zurückgegeben mit den neuen Zeitwerten). Außerdem kann spezifiziert werden, mit welchem Fenster die Daten im Frequenzbereich vor dem Zero-Padding ausgeblendet werden.

Resample (Matlab): x wird um den Faktor p/q mit Polyphasenfilterung resampled. Während des Resamplingprozesses wird ein `firls` Anti-Aliasfilter verwendet, das mit einem Kaiserfenster entworfen wird. Optionale Parameter sind der Parameter `beta` des Kaiserfensters (Default `beta = 5`) und die Anzahl `N` der Werte, die vor und nach dem aktuellen Sample $x(k)$ beim Resampling berücksichtigt werden (Default `N = 10`), um Randeffekte abzumildern. Die Länge des resultierenden Arrays ist `ceil(length(x)*p/q)`. Matlab kann im Gegensatz zur Python-Variante auch mit nicht-äquidistanten Sequenzen umgehen. Für Python gibt es eine schnelle Implementierung unter <https://code.google.com/archive/p/upfirdn/>, siehe auch nächster Punkt.

UpFIRDown erhöht die Anzahl der Samples von x durch Nullenstopfen um den Faktor p (Upsampling), filtert mit der Impulsantwort h eines FIR-Filters und reduziert die Anzahl der Samples danach um den Faktor q (Downsampling).

Signal Processing Library (<http://mubeta06.github.io/python/sp/multirate.html>) rüstet ein paar Multirate-Befehle in Python nach, die in `scipy` fehlen, und zwar `downsample`, `upsample`, `upfirdn`, `interp` und `resample`.

Python

Flexibler sind die Interpolatoren aus `scipy.interpolate`: Im Code-Schnipsel Listing 9.1 werden durch kubische Spline-Interpolation nicht-äquidistante Wertepaare der Funktion $y = f(x)$ in äquidistante, dichter abgetastete Wertepaare $y_{\text{new}} = f(x_{\text{new}})$ umgewandelt:

```

1 import numpy as np
2 from scipy import interpolate
3 import matplotlib.pyplot as plt
4 # Definiere y = f(x) mit 13 nicht-äquidistanten Wertepaaren
5 x = np.array([0,1,2,3,4,5,6,6.5,7,7.5,8,8.5,9])
6 y = np.cos(-x**2/4.0)
7 f = interpolate.interp1d(x, y, kind = 'quadratic')

```

```

8
9 xnew = np.linspace(x[0], x[-1], 50) # 50 äquidistante x-Werte
10 ynew = f(xnew) # Berechne interpolierte Werte bei xnew
11
12 plt.plot(x,y, 'bo-')
13 plt.plot(xx,yy, 'g--')
14 plt.show()

```

Lst. 9.1: Interpolation / Resampling mit Python

Natürlich kann man genauso äquidistante in äquidistante Wertepaare umrechnen („normale“ Abtastratenwandlung), es lassen sich verschiedene Arten der Interpolation (ZOH, linear, Spline k-ter Ordnung, ...) wählen. `y` und `ynew` können auch mehrdimensionale Arrays (z.B. Stereosignal) sein, dann werden mehrere Werte zu den gleichen `xnew` Punkten berechnet. Bei langen Sequenzen sollte `assume_sorted = True` gesetzt werden (und die Sequenz monoton sortiert sein), um Sortierzeit zu sparen.

Eine weitere Alternative ist `pandas.DataFrame.resample` aus dem `pandas` Modul (noch nicht selbst ausprobiert).

9.1. Einfache Abtastratenwandlung → M9.1

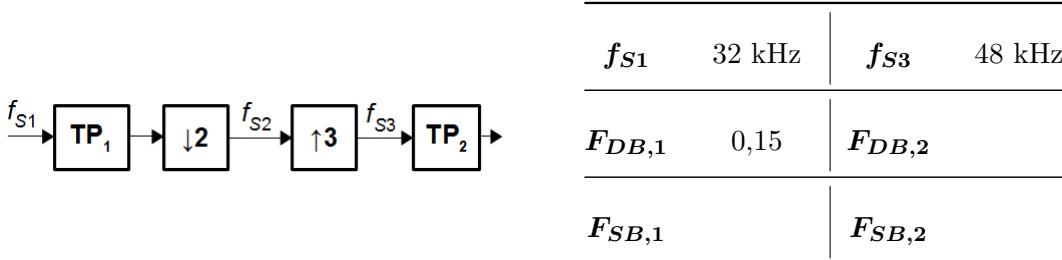


Abb. 9.2.: Synchrone Abtastratenwandlung zu Aufgabe 9.1

In Abb. 9.2 ist eine Kaskade aus Dezimator ($R = 2$) und Interpolator ($I = 3$) zur synchronen Wandlung der Abtastrate von $f_{S1} = 32$ kHz auf $f_{S3} = 48$ kHz dargestellt. Die Durchlassgrenze von TP_1 ist $F_{DB,1} = f_{DB,1}/f_{S1} = 0,15$, die Tiefpässe sperren zunächst ideal.

- Welche Aufgaben haben die Filter TP_1 und TP_2 ?
- Berechnen Sie die Eckfrequenz des Sperrbands von TP_1 $F_{SB,1} = f_{SB,1}/f_{S1}$ so, dass keinerlei Aliasing auftritt.
- Berechnen Sie die Eckfrequenz des Sperrbands von TP_2 $F_{SB,2} = f_{SB,2}/f_{S3}$ so, dass alle Images unterdrückt werden. Berechnen Sie außerdem die Eckfrequenz des Durchlassbands $F_{DB,2} = f_{DB,2}/f_{S3}$.

Die Tiefpässe haben jetzt eine Sperrdämpfung von nur 20 dB, daher treten Aliasingeffekte auf und Images werden nicht mehr vollständig unterdrückt. Als Testsignale werden nacheinander Sinustöne eingespeist mit den Frequenzen $F_{0,\alpha} = f_{0,\alpha}/f_{S1} = 0,1$ und $F_{0,\beta} = f_{0,\beta}/f_{S1} = 0,4$.

- Welche Frequenzkomponenten f/f_{S2} treten am Ausgang des Dezimators, welche Frequenzkomponenten treten f/f_{S3} am Ausgang des Interpolators im jeweiligen Basisband $0 \dots f_{S,i}/2$ auf?

- e) Entwerfen Sie mit Python / Matlab linearphasige Dezimations- und Interpolationsfilter so, dass das Durchlassband des Gesamtsystems keine Welligkeit hat. Die Sperrdämpfung soll größer als 80 dB sein, der Ripple im Durchlassband kleiner als 0,1% sein.

9.2. * Ideale Abtastratenwandlung → M9.2

In Abb. 9.3 ist eine Kaskade aus Dezimator ($R = 5$) und Interpolator ($I = 6$) zur Wandlung der Abtastrate dargestellt. $s_1[n]$ ist das Ausgangssignal eines 8-Bit Oversampling ADCs, der mit der Abtastrate $f_{S1} = 100$ kHz arbeitet, die Nutzbandbreite ist $f_N = 8$ kHz (Abb. 9.4).

Es wird zunächst angenommen, dass die Tiefpässe ideal sperren.

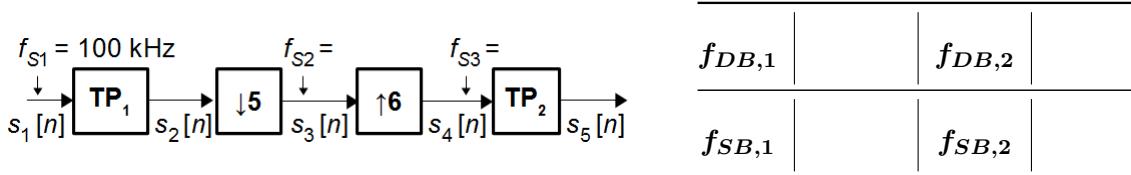


Abb. 9.3.: Abtastratenwandlung zu Aufgabe 9.2 und 9.3

- Geben Sie das **Verhältnis der Abtastraten** f_{S1} , f_{S2} und f_{S3} zueinander an.
- Welche Aufgabe hat TP_1 ? Berechnen Sie die **Eckfrequenzen von TP_1** (Durchlass- und Sperrband F_{DB1} und F_{SB1}) so, dass auch im Übergangsbereich kein Aliasing auftritt. Das Nutzsignal darf nicht beschnitten werden. Skizzieren den Frequenzgang $|H_{TP1}(e^{j2\pi F})|$ von TP_1 sowie die Spektren $|S_2(e^{j2\pi F})|$ und $|S_3(e^{j2\pi F})|$ in Abb. 9.4.
- Welche Aufgabe hat TP_2 ? Bestimmen Sie die **Eckfrequenzen von TP_2** (Durchlass- und Sperrband F_{DB2} und F_{SB2}) so, dass bei maximaler Bandbreite alle Images unterdrückt werden. Skizzieren Sie den Frequenzgang $|H_{TP2}(e^{j2\pi F})|$ von TP_2 sowie die Spektren $|S_4(e^{j2\pi F})|$ und $|S_5(e^{j2\pi F})|$ in Abb. 9.4.

Dezimations- und Interpolationsstufe sollen jetzt vertauscht werden.

- Skizzieren Sie das **Blockschaltbild dieser „vertauschten“ Kaskade** und tragen Sie im gleichen Bild die Eckfrequenzen der Tiefpassfilter ein. Achten Sie dabei auf die richtige Anordnung der Tiefpassfilter für Interpolation und Dezimation - welche Aufgabe hat welches Tiefpassfilter? Kann man das System vereinfachen? Optional: Skizzieren Sie die Spektren und Frequenzgänge (ohne ML).
- Bestimmen Sie **SQNR** und **ENOB** für die Signale $s_1[n]$ bis $s_5[n]$ in beiden Systemen. Gehen Sie dazu von einem sinusförmigen Eingangssignal mit $f_1 < f_N$ aus, das den ADC voll aussteuert und betrachten Sie die Rauschbandbreite an den verschiedenen Punkten im System. Wie ändern sich Rauschleistung und Rauschleistungsdichte bei Erhöhung oder Erniedrigung der Abtastrate, was geschieht in TP_1 und TP_2 ?
- Welche **Vor- und Nachteile** haben die beiden Systeme? Vergleichen Sie dabei die maximal mögliche Bandbreite B_{max} und die höchste Abtastfrequenz $f_{S,max}$ im System. Die Tiefpassfilter sollen als FIR-Filter realisiert werden. Bestimmen Sie die benötigte Ordnung für die beiden Filter mit der Näherungsformel (4.5) und einem Filterentwurfstool und tragen Sie alle Werte in Tab. 9.2 ein. Verwenden Sie Equiripple-Filter mit einer minimalen

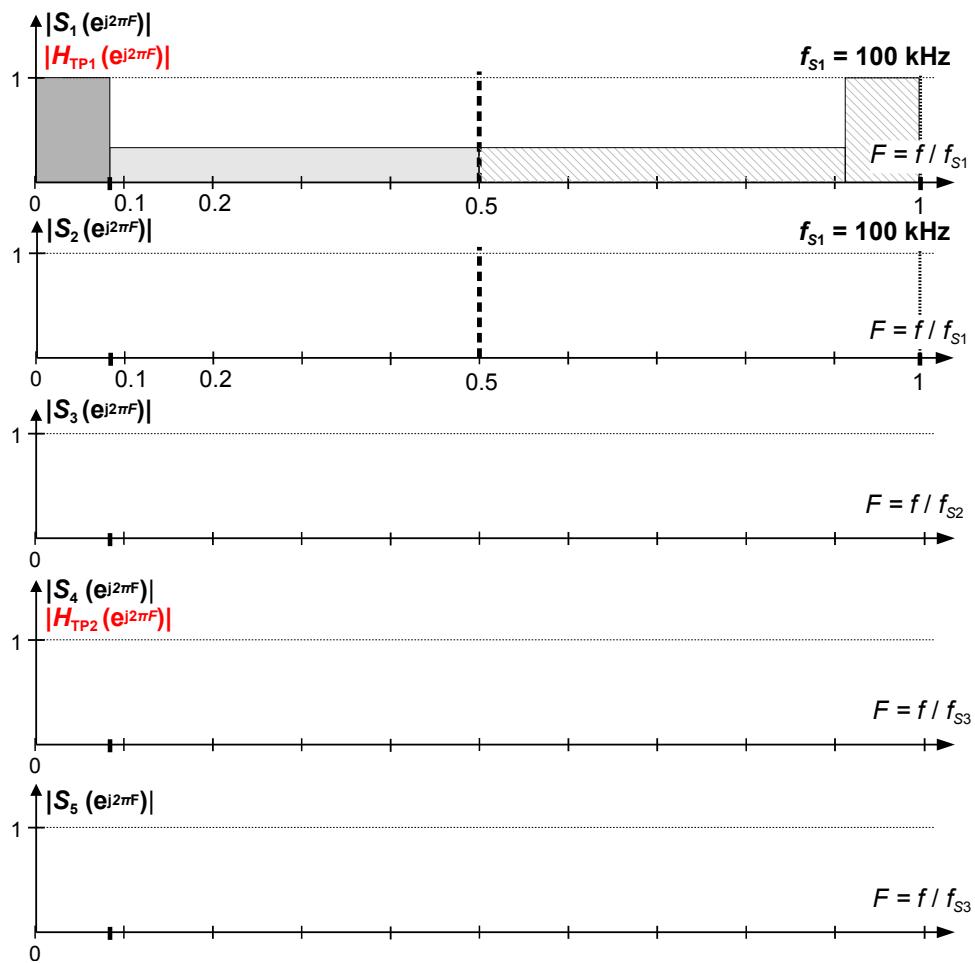


Abb. 9.4.: Spektren und Frequenzgänge zu Aufgabe 9.2

Sperrdämpfung von $\delta_{SB} = 10^{-3} \hat{=} 60$ dB und einem maximalen Ripple von $\delta_{DB} = 10^{-2}$ im Durchlassband.

System	B_{max}	$f_{S,max}$	$N_{FIR,1}$	$N_{FIR,2}$
			\approx	sim
a)				
d) (2 TP)				
d) (1 TP)				—

Tab. 9.2.: Lösungen zu Aufgabe 9.2

9.3. Abtastratenwandlung mit Aliasing im Übergangsbereich → M9.3

Im System Abb. 9.3 soll jetzt für einen entspannteren Filterentwurf bei der Dezimation Aliasing in den Übergangsbereich erlaubt sein.

- a) Bestimmen Sie die hierfür zulässigen Eckfrequenzen von TP_1 .
- b) Welche Spezifikationen werden jetzt für TP_2 benötigt?
- c) Schätzen Sie die Ordnungen $N_{FIR,1}$ und $N_{FIR,2}$ für die Filter ab.

10. CIC: Cascaded Integrator-Comb Filter

Inhalt

Dieses Kapitel beschäftigt sich mit CIC-Filtern. Wesentliche Punkte sind dabei:

Basics: Welche Übertragungsfunktion hat ein CIC-Filter und wie hängt es mit Moving Average Filtern zusammen?

Fixpoint-Effekte: Wie kann ich das Wortwachstum abschätzen?

Multiratenfilter: Wie funktionieren Multiraten - CIC-Filter?

Dezimation: Wieviel Aliasing erhalte ich bei dezimierenden CIC-Filtern?

Interpolation: Wie gut ist die Image-Unterdrückung beim interpolierenden CIC-Filtern?

Python- / Matlab-Befehle für CIC-Filter

Für dieses Kapitel benötigt man die gleichen Befehle wie für Multiratenfilter und Sample-Rate Conversion:

Filename	Beschreibung
CIC_xxx_xxx	CIC-Filter

Tab. 10.1.: Simulationsfiles zu Kap. 10

10.1. + CIC-Filter → M10.1

Ein CIC-Filter zweiter Ordnung ($N = 2$) mit $R = 32$ wird benutzt zur Tiefpassfilterung eines Bitstroms mit der Eingangsratenrate $f_{S,ein} = 26$ MHz und der Nutzbandbreite $f_N = 200$ kHz. Zunächst werde keine Dezimierung benutzt (Abb. 10.1).

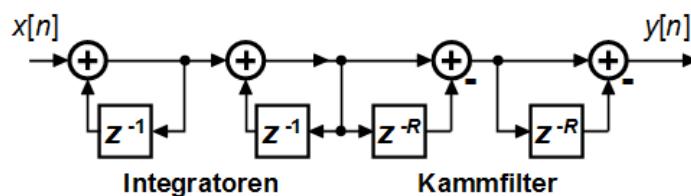


Abb. 10.1.: Zweistufiges CIC-Filter zu Aufgabe 10.1a)

- a) Wie groß ist die Verstärkung des Nutzsignals bei DC, $H_{CIC}(f = 0)$? Wie groß ist die maximale Dämpfung des Nutzsignals bezogen auf $H_{CIC}(f = 0)$? Wie stark wird eine unerwünschte Frequenzkomponente bei $2f_N = 400$ kHz gedämpft?

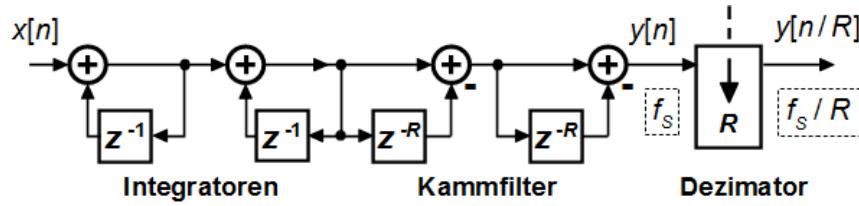


Abb. 10.2.: Zweistufiges CIC-Filter mit Dezimation zu Aufgabe 10.1b)

- b) Der Ausgang des Filters wird jetzt um den Faktor $R = 32$ dezimiert (Abb. 10.2). Wie groß ist die Ausgangstaktrate $f_{S,R}$? Welche maximale Eingangsfrequenz kann ohne Aliasing verarbeitet werden?
- c) Bei welcher Eingangsfrequenz erhält man das stärkste Aliasing ins Nutzband? Wie groß ist die relative Dämpfung (bezogen auf die Verstärkung bei $f = 0$) bei $f_1 = 612.5$ kHz, $f_2 = 812.5$ kHz und $f_3 = 1012.5$ kHz?

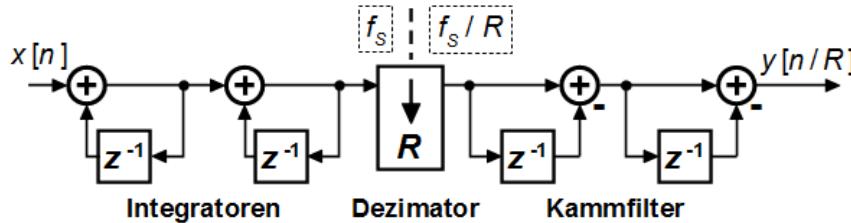


Abb. 10.3.: Zweistufiges CIC-Filter mit Dezimation zu Aufgabe 10.1d) (Hogenauer-Struktur)

- d) Das CIC-Filter wird jetzt in der Hogenauer - Struktur aufgebaut (Abb. 10.3), nur die Integratoren wirken hier als Anti-Aliasingfilter. Wie groß ist deren Dämpfung bei den Frequenzen des vorigen Unterpunkts bezogen auf die Verstärkung bei $f = 0$?
- e) Durch welche Maßnahmen kann man die Alias-Unterdrückung von dezimierenden CIC-Filtern verbessern?
- f) Wie beeinflussen die folgenden Maßnahmen die maximale Dämpfung im Durchlassband, die Dämpfung bei $2f_N$ und die Worst-Case Alias-Dämpfung:
- Halbierung von f_N
 - Verdopplung von R
 - Verdopplung von N
- g) Vergleichen Sie die benötigten Hardware-Ressourcen für die folgenden Filter:
- $N = 2$ kaskadierte Moving Average Filter mit je $R = 32$ Taps
 - CIC-Filter (Abb. 10.1)
 - Dezimierendes CIC-Filter (Abb. 10.2)
 - Dezimierendes CIC-Filter in Hogenauer-Struktur (Abb. 10.3)

Für alle CIC-Filter soll gelten $N = 2$ und $R = 32$.

11. IIR: Rekursive Filter

Inhalt

Dieses Kapitel beschäftigt sich mit der Definition und dem Entwurf rekursiver oder IIR-Filter:

Spezifikationen: Wie werden Filter spezifiziert, was bedeuten die Spezifikationen?

Filtereigenschaften: Wie unterscheiden sich FIR und IIR - Filter, welchen Einfluss hat die Wahl des Entwurfsverfahrens auf die Filtereigenschaften?

Halbbandfilter: Wie erkennt man Halbbandfilter im Zeit- und Frequenzbereich, welche speziellen Eigenschaften haben sie?

Resonator: Was ist ein digitaler Resonator?

Einführung

Entwurfsverfahren für IIR-Filter

Das Ergebnis von Filterentwürfen mit Python / Matlab sind generell Vektoren mit Zähler- und Nennerkoeffizienten (nur bei IIR-Filtern) von $H(z)$ in Polynomschreibweise mit positiven Exponenten. Funktionen zum Filterentwurf sind in Python im Modul `scipy.signal` und in Matlab in der Signal Processing Toolbox enthalten.

Beim „klassischen“ IIR-Filterentwurf entwirft man zunächst ausgehend von den Filterspezifikationen die Übertragungsfunktion $H(s)$ eines analogen Filters, das man auch zeitkontinuierlich implementieren könnte, z.B. als passives *LC*- oder aktives *RC*-Filter. Abhängig von den Anforderungen (max. Ordnung, Dämpfung, Welligkeit, Gruppenlaufzeit, ...) verwendet man ein Bessel, Butterworth, Chebychev oder elliptisches Filter als *Prototypen- oder Referenzfilter*.

Mit bilinearer Transformation wird $H(s)$ auf die zeitdiskrete Übertragungsfunktion $H(z)$ abgebildet. Diese Aufgaben werden glücklicherweise heutzutage von Entwurfssoftware übernommen. Aufgrund der „analogen Vergangenheit“ werden allerdings beim Entwurf von IIR - Filtern die -3dB Grenzfrequenz und die Ordnung vorgegeben und bestimmen dann Dämpfung und die Eckfrequenzen von Durchlass- und Sperrband. Um die Filterspezifikationen zu erfüllen, kann man entweder ausprobieren oder zunächst mit einer Hilfsfunktion Ordnung und Eckfrequenz abschätzen.

Besselfilter

Analoge Besselfilter werden verwendet, wenn eine möglichst konstante Gruppenlaufzeit und damit geringe Phasenverzerrungen benötigt werden. Diesen Vorteil erkauft man sich mit einem

sehr gemächlichen Übergang vom Durchlass- in den Sperrbereich. Digitale Besselfilter werden nur sehr selten eingesetzt, stattdessen verwendet man normalerweise FIR-Filter, die bei gleicher Ordnung ähnliche Dämpfungen erzielen und *keinerlei* Phasenverzerrungen verursachen. In Matlab gibt es daher auch keine Funktion zum Entwurf digitaler Bessel-Filter.

Butterworthfilter

Butterworthfilter haben einen monotonen Verlauf der Verstärkung (kein Ripple) und einen gutmütigen Verlauf der Gruppenlaufzeit. In Matlab und Python gibt es die Hilfsfunktion **buttord()** zur Abschätzung der Grenzfrequenz und Ordnung und **butter()** zum eigentlichen Filterentwurf.

Chebychevfilter

Chebychevfilter werden verwendet, wenn Ripple im Durchlassband (**cheby1()**) oder im Sperrband (**cheby2()**) akzeptabel ist. Dadurch wird ein deutlich steilerer Übergangsbereich bzw. eine geringere Filterordnung erreicht, allerdings auf Kosten eines stärker nichtlinearen Phasengangs. Auch hier gibt es Hilfsfunktionen **cheb1ord()** bzw. **cheb2ord()** zur Abschätzung von Grenzfrequenz und Ordnung.

Elliptische Filter

Wenn Ripple im Durchlassband *und* Sperrband akzeptabel ist, erzielt man mit elliptischen Filtern (**|ellip()**) und **|ellipord()** die geringste Filterordnung für gegebene Spezifikationen.

In Python gibt es einen Wrapper **iirdesign()**, der die vorher beschriebenen Filtertypen '**'ellip'**', '**'cheby1'**', '**'cheby2'**', '**'butter'**', '**'bessel'**' zusammenfasst:

```

1 ##### IIR-Filterentwurf #####
2 === Butterworth Filter=====
3 [Lb,F_b] = sig.buttord(F_DB,F_SB,A_DB,A_SB)
4 #[b, a] = sig.butter(Lb, F_b)
5 === IIR-Wrapper (nur Python) ====
6 #[b, a] = sig.iirdesign(F_DB, F_SB,
7 #                         A_DB, A_SB, ftype='ellip')
```

Lst. 11.1: IIR-Filterentwurf mit Python

```

%%%%% IIR-Filterentwurf %%%%%%
%% Butterworth-Filter
[L_b, F_b] = buttord(F_DB, F_SB, A_DB, A_SB);
[b, a] = butter(L_b, F_b);
%% Elliptisches Filter
[L_e,F_e] = ellipord(F_DB, F_SB, A_DB, A_SB);
[b, a] = ellip(L_e, A_DB, A_SB, F_e);
```

Lst. 11.2: IIR-Filterentwurf mit Matlab

Filename	Beschreibung
FIL_Intro	Implementierung verschiedener Filterentwurfsverfahren in Matlab und Python (siehe oben)
FIL_BPSK	Bit Error Rate (BER) bei Störung eines BPSK-kodierten Signals durch additives weißes Rauschen. Es wird gezeigt, welchen Einfluss Grenzfrequenz und Phasenverzerrungen von verschiedenen Tiefpassen auf die BER haben.

FIL-Linphase_Filter	Einfluss von einfachen und doppelten Nullstellen auf den Amplitudengang (zu Aufgabe 4.2)
FIL-Linphase- _Filter_annotation	wie voriges Programm, aber mit zusätzlichen Textausgaben im Plot
FIL-Halfband_design	Entwurf von Halbbandfiltern mit unterschiedlichen Methoden, Vergleich im Zeit- und Frequenzbereich (Aufgabe 4.4)

Tab. 11.1.: Simulationsfiles zu Kap. 11

Experimente und Lesestoff zu Kap. 11

IIR Filter auf Xilinx FPGAs http://www.xilinx.com/support/documentation/white_papers/wp330.pdf

11.0. * Filterentwurf für Sensorsignal

Aufgabe 4.0 soll jetzt mit rekursiven Filtern gelöst werden.

Brummstörungen bei 50 Hz auf einem Sensorsignal sollen mit einem digitalen Filter um mindestens 60 dB unterdrückt werden. Das Sensorsignal wird mit einem ADC mit $f_S = 400$ Hz abgetastet und hat eine Nutzbandbreite von 0 ... 40 Hz. Das Filter soll einen Amplitudenfehler von maximal ± 1 dB verursachen.

Entwerfen Sie verschiedene FIR und IIR-Filter mit dem graphischen Entwurfstool pyFDA oder aufbauend auf den Skripten Listing 4.1 bzw. 4.2 und versuchen Sie, die gestellten Anforderungen zu erfüllen.

Vergleichen Sie die Systeme bezüglich Resourcenverbrauch (Ordnung) und schauen Sie sich die unterschiedlichen Pol/Nullstellenpläne und Betrags- und Phasenverlauf an! Welche Filter haben eine gute Unterdrückung von Störungen bei höheren Frequenzen (z.B. 150 Hz)?

Zu dieser Aufgabe gibt es keine Musterlösung, die Ergebnisse werden in der Vorlesung besprochen.

11.1. * Filtertransformationen → M11.1

Vergleiche auch Aufgabe 4.6.

Ein IIR-Filter F1 mit der Systemfunktion xxx hat den Betragsfrequenzgang in

- a) Berechnen Sie $H_1(f = 0)$ und $H_1(f = f_S/2)$.
- b) Aus F1 wird ein neues Filter F2 abgeleitet, indem jedes Verzögerungsglied verdoppelt wird. Geben Sie $H_2(z)$ an. Wie wirkt sich die Transformation $z^{-1} \rightarrow z^{-2}$ auf den Betragsfrequenzgang aus? Skizzieren Sie den Betragsfrequenzgang von F2.
- c) Erzeugen Sie mittels TP-HP-Transformation $z \rightarrow -z$ aus Filter F1 ein Filter F3. Geben Sie $h_3[n]$ an und skizzieren Sie den zugehörigen Betragsfrequenzgang.

11.2. Bilineare Transformation → M11.2

Ein in der Audiotechnik verwendetes analoges Tiefenanhebungsfilter mit dem im Bode-Diagramm Abb. 11.1 skizzierten Betragsfrequenzgang soll durch ein digitales IIR-Filter mit einer Abtastfrequenz von $f_S = 32$ kHz ersetzt werden. Dabei soll die Eckfrequenz $f_g = 2$ kHz erhalten bleiben.

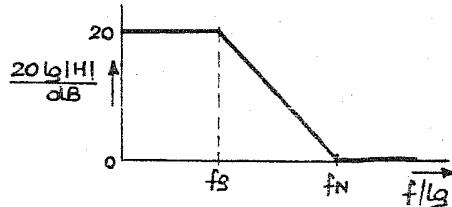


Abb. 11.1.: Amplitudengang $|H(f)|$ des analogen Referenzfilters

- Stellen Sie die analoge Übertragungsfunktion $H(s)$ auf.
- Wenden Sie die Bilineartransformation an, um $H(z)$ des digitalen Filters zu ermitteln.
- Vergleichen Sie die Frequenzen f_N der beiden Realisierungen, bei denen $20 \log_{10} |H(f)| = -3$ dB ist.

11.3. Resonator → M11.3

Systeme mit einem oder zwei komplexen Polen werden auch Resonator genannt, da sie abhängig von den Koeffizienten gedämpfte oder auch ungedämpfte Resonanz zeigen. Die Übertragungsfunktion eines rein rekursiven Systems zweiter Ordnung wird beschrieben durch

$$H(z) = \frac{g_0}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{g_0 z^2}{z^2 + a_1 z^1 + a_2}$$

Dabei sollen a_1, a_2 reelle Koeffizienten sein.

- Leiten Sie aus beiden Formen der Übertragungsfunktion die Differenzengleichung ab und zeichnen Sie die daraus abgeleitete Hardwareimplementierung des Systems auf.
- Zeigen Sie, wie man durch Umformungen des Signalflussgraphen (Verschieben von Verzögerungen etc.) beide Formen in einander überführen kann.
- Bestimmen Sie allgemein die Polstellen des Systems
- Ermitteln Sie den Bereich für die Koeffizienten a_1, a_2 , in dem das System stabil ist und stellen Sie ihn graphisch dar.

12. ZST: Zustandsraum

$$\begin{bmatrix} \cos 90^\circ & \sin 90^\circ \\ -\sin 90^\circ & \cos 90^\circ \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Abb. 12.1.: Matrixtransformation [<http://xkcd.com/184/>]

Allgemeine Hinweise zu Zustandsgleichungen

Eine gute Beschreibung der Zustandsraumdarstellung speziell für die digitale Signalverarbeitung findet sich in [Mey11]. Das entsprechende Kapitel gibt es auch zum Download als PDF.

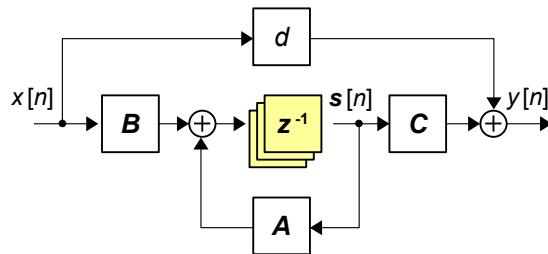


Abb. 12.2.: Allgemeine Struktur eines zeitdiskreten Systems n -ter Ordnung in Zustandsraumbeschreibung mit dem Vektor der Zustandsgrößen $s[n]$

Da die Zustandsraumdarstellung aus der Struktur des Systems abgeleitet wird, haben Systeme mit unterschiedlicher Struktur und gleicher Übertragungsfunktion im Allgemeinen unterschiedliche Zustandsraumdarstellungen.

Zur Bestimmung der Zustandsgleichungen von Abb. xxx (System zweiter Ordnung) bzw. Abb. 12.2 (allgemeiner Fall) müssen die Größen $\mathbf{A}, \mathbf{B}, \mathbf{C}$ und d von folgendem Gleichungssystem ermittelt werden:

$$\begin{aligned} s[n+1] &= \mathbf{As}[n] + \mathbf{B}x[n] \\ y[n] &= \mathbf{Cs}[n] + dx[n] \end{aligned}$$

s bezeichnet dabei die zeitdiskreten Zustandsgrößen (s wie „state variable“), x das Eingangssignal.¹ In dieser Vorlesung werden nur Systeme mit einer Eingangs- und einer Ausgangsgröße behandelt, $x[n], y[n]$ und die entsprechenden Größen in der Frequenzebene sowie d sind daher hier Skalare.

¹In der Regelungstechnik werden Zustandsgrößen üblicherweise mit x und die Eingangsgröße mit u bezeichnet. Wegen Verwechslungsgefahr wurde hier x für die Eingangsgröße beibehalten und die Zustandsgrößen mit s bezeichnet. Da hier nur zeitdiskrete Zustandssysteme betrachtet werden, ist eine Verwechslung mit der komplexen Frequenz s nicht möglich.

Masons Regel

Komplexe Graphen mit mehreren verschachtelten Schleifen lassen sich durch schrittweises Zusammenfassen nur mühsam analysieren. Für diese Fälle kann man mit Hilfe von *Masons [Schleifen-] Regel* (ebenfalls bekannt aus der Regelungstechnik) schnell die Übertragungsfunktion H_{ij} von Knoten x_i zu Knoten x_j bestimmen:

$$H_{ij} = \frac{x_j}{x_i} = \frac{\sum_k^N P_{ij,k} \overbrace{\left(1 + \sum_m (-1)^m [\text{Schleifen } m\text{-ter Ordnung, } \cap \text{ Pfad } k = \emptyset] \right)}^{:= \Delta_{ij,k}}}{\underbrace{1 + \sum_m (-1)^m [\text{Schleifen } m\text{-ter Ordnung}]}_{:= \Delta}} \quad (12.1)$$

Dabei sind die folgenden Begriffe definiert:

- Die *Pfadverstärkung* $P_{ij,k}$ ist das Produkt aller Transmittanzen $t_{ij,k}$ entlang des k -ten Pfads von Knoten x_i zu Knoten x_j .
- Die *Schleifenverstärkung* L_n ist das Produkt aller Transmittanzen in einer Schleife.
- Δ ist die *Determinante* (s.u.) des Graphen.
- $\Delta_{ij,k}$ ist der *Kofaktor* (s.u.) des Pfads P_{ijk} .

Die Summation wird über alle N möglichen Vorwärtspfade von Knoten x_i zu Knoten x_j ausgeführt. Die Determinante wird aus allen verschiedenen Schleifenverstärkungen des Graphen bestimmt:

$$\Delta := 1 - \sum_n^N L_n + \sum_{m,q} \underbrace{L_m L_q}_{\text{nicht-berührend}} - \sum_{r,s,t} \underbrace{L_r L_s L_t}_{\text{nicht-berührend}} + \dots \quad (12.2)$$

Die Produkte aus zwei oder mehreren Schleifenverstärkungen werden aus der Kombination aller Schleifen ohne gemeinsame Knoten (nicht berührend) berechnet. Der Kofaktor Δ_{ijk} wird auf ähnliche Weise berechnet, nur ohne die Schleifen, die Pfad k berühren.

Für den trivialen Fall einer einzelnen Schleife L vereinfacht sich (12.1) zu:

$$H_{ij} = \frac{x_j}{x_i} = \frac{\sum_k^N P_{ij,k}}{\Delta} = \frac{\sum_k^N P_{ij,k}}{1 - L} \quad (12.3)$$

Zur Stabilitätsanalyse der allermeisten Systeme genügt es die Nullstellen der charakteristischen Gleichung Δ zu betrachten, die den Polstellen der der Systemfunktion entsprechen.

12.0. Python / Matlab

Erzeugen des Systems

Aus Koeffizienten: `mysys=tf(b,a,Ts); mysys=fir2sys(b,Ts);` (für FIR-Filter)

Aus Polen und Nullstellen: `mysys=zp(z,p,k,Ts)`

Aus Zustandsvektoren / matrizen: `my_sys = ss (A, B, C ,d, Ts);`

Anmerkung: Ohne Angabe von `Ts` (Sample-Time) wird ein zeitkontinuierliches System erzeugt.

Analyse des Systems

`[b,a]=sys2tf(my_sys); [z,p,k]=sys2zp(mysys); sysout(mysys)`
oder `[b,a]=ss2tf(A,B,C,d); [z,p,k] = ss2pz(A,B,C,d);`

Graphische Darstellung

`bode(mysys); rlocus(mysys); ...`

Manipulation des Systems

Kombination von Subsystemen: `sysadd, syssub, sysmult, ...`

12.1. Zustandsgleichungen aus SFG → M12.1

Bestimmen Sie die Zustandsraumdarstellung

- a) der Systeme aus Aufgabe 1.6, Abb. 1.9.
- b) und der dazu transponierten Systeme aus Aufgabe 1.7, Abb. M1.8.

Die Matrix \mathbf{A}^T des transponierten Systems ist transponiert zur Matrix \mathbf{A} des ursprünglichen Systems - daher stammt auch die Bezeichnung „transponierte Form“ - überprüfen Sie das anhand der gefundenen Lösungen!

ML zu Abb. c) und d) fehlen jeweils noch!

12.2. SFG aus Zustandsgleichungen → M12.2

Gegeben ist folgendes System von Zustandsgleichungen:

$$\begin{aligned}s[n+1] &= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} s[n] + \begin{bmatrix} 0 \\ 1 \end{bmatrix} x[n] \\ y[n] &= \begin{bmatrix} 1 & 1 \end{bmatrix} s[n] + x[n]\end{aligned}$$

- a) Ermitteln Sie die Übertragungsfunktion $H(z) = Y(z)/X(z)$.
- b) Bestimmen Sie die Polstellen von $H(z)$. Ist das System stabil oder instabil?
- c) Geben Sie eine mögliche Realisierung dieses Systems als digitales IIR-Filters an.
- d) Ermitteln Sie aus der unter c) gefundenen Realisierung die Impulsantwort $h[n]$ bis $n = 7$.

Teil II.

Musterlösungen

M1. LTI-Systeme im Zeitbereich

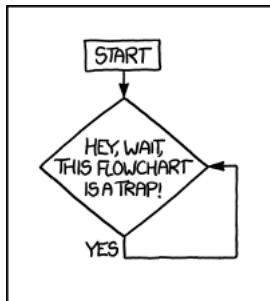


Abb. M1.1.: <http://xkcd.com/1195/>

M1.1. Filterung abgetasteter Signale → Aufgabe 1.1

a) Abgetastete Signale / Periodizität

Nach der Abtastung eines Signals mit der Frequenz $f_S = 1/T_S$ sind Funktionswerte nur noch bei den Abtastzeitpunkten kT_S mit $k = 0, 1, 2, \dots$ definiert. Daher wird aus

$$\begin{aligned} x(t) &= 1,5 \text{ V} + 0,5 \text{ V} \cos \underbrace{(2\pi \cdot 50 \text{ Hz} \cdot t)}_{\varphi(t)} \\ \Rightarrow x(kT_S) &= x[k] = 1,5 \text{ V} + 0,5 \text{ V} \cos \underbrace{(2\pi \cdot 50 \text{ Hz} \cdot kT_S)}_{\varphi[k]} \end{aligned}$$

Setzt man die Abtastfrequenzen $f_{S,i}$ ein, erhält man die zeitdiskreten Folgen $x_i[k]$ (Abb. M1.2a - d):

$$\begin{aligned} x_1[k] &= 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi k \cdot 50 \text{ Hz}/25 \text{ Hz}) = 1,5 \text{ V} + 0,5 \text{ V} \cos(4\pi k) = 2 \text{ V} \\ x_2[k] &= 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi k \cdot 50 \text{ Hz}/200 \text{ Hz}) = 1,5 \text{ V} + 0,5 \text{ V} \cos(\pi k/2) \\ x_3[k] &= 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi k \cdot 50 \text{ Hz}/240 \text{ Hz}) = 1,5 \text{ V} + 0,5 \text{ V} \cos(5\pi k/12) \\ x_4[k] &= 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi k \cdot 50 \text{ Hz}/(50 \cdot \pi \text{ Hz})) = 1,5 \text{ V} + 0,5 \text{ V} \cos(2k) \end{aligned}$$

Man sieht, dass das Abtastsignal für $f_{S,1} = 25$ Hz nicht mehr dem ursprünglichen Signal ähnelt; der Wechselanteil ist verschwunden und der Gleichanteil hat den falschen Wert. Dies ist eine Folge der zu geringen Abtastfrequenz; das Nyquistkriterium wird verletzt und es tritt *Aliasing* auf (→ Kap. 7).

Die cos-Funktion ist mit 2π periodisch, daher ist die kontinuierliche Cosinusfunktion $s(t)$ immer periodisch mit dem Kehrwert der Frequenz $T_1 = 1/f_1$:

$$\begin{aligned} s(t) &= \cos(2\pi f_1 t) = \cos(2\pi f_1 t + 2\pi L) = \cos(2\pi f_1(t + LT_1)) = s(t + LT_1) \quad \text{mit } L \in \mathbb{Z} \\ \Leftrightarrow \varphi(t) &= 2\pi f_1 t = 2\pi f_1 t + 2\pi L - 2\pi L = \varphi(t + LT_1) - 2\pi L \end{aligned}$$

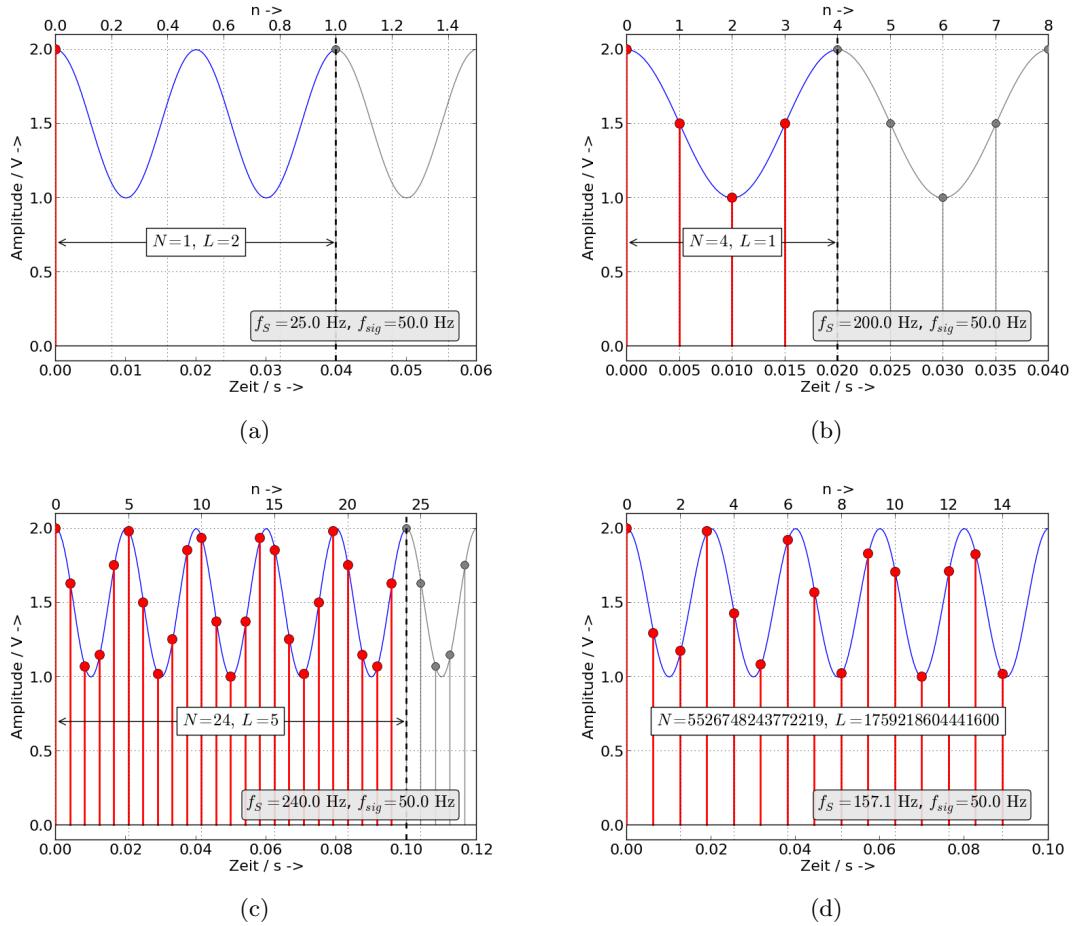


Abb. M1.2.: Eingangssignal $x(t) = 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi \cdot 50 \text{ Hz} \cdot t)$, abgetastet mit verschiedenen Samplingraten f_S

Die Phase $\phi(t)$ steigt linear mit t an (lineare Phase).

Die zeitdiskrete cos-Funktion $s[k]$ ist nur bei $t = kT_S$ definiert; sie ist nur dann periodisch wenn N Abtastwerte L Perioden von $s(t)$ ergeben bzw. wenn $\varphi[N] \bmod 2\pi \equiv 0$:

$$\begin{aligned} \cos(2\pi f_1 k T_S) &\stackrel{!}{=} \cos(2\pi f_1 (k + N) T_S - 2\pi L) \quad \text{mit } L, N \in \mathbb{N}_0 \\ \Rightarrow \varphi(k T_S) &= 2\pi f_1 k T_S \stackrel{!}{=} 2\pi f_1 (k + N) T_S - 2\pi L = \varphi((k + N) T_S) - 2\pi L \end{aligned}$$

In Kurzschreibweise $\varphi[k] := \varphi(k T_s)$:

$$\begin{aligned} \varphi[k] &\stackrel{!}{=} \varphi[k + N] - 2\pi L \\ \Leftrightarrow 2\pi k T_S / T_1 &\stackrel{!}{=} 2\pi(k + N) T_S / T_1 - 2\pi L \\ \Leftrightarrow k T_S / T_1 &\stackrel{!}{=} (k + N) T_S / T_1 - L \end{aligned}$$

Diese Gleichung muss für jeden Zeitpunkt k erfüllt sein (LTI-System), also auch für $k = 0$:

$$0 = N T_S / T_1 - L \Rightarrow N T_S = L T_1$$

Damit die obige Gleichung für ganzzahlige Werte von L, N erfüllbar ist¹, müssen Abtastperiode T_S und Signalperiode T_1 (bzw. Abtastfrequenz f_S und Signalfrequenz f_1) ganzzahlige Vielfache einer gemeinsamen Zeiteinheit T_c (bzw. Frequenz f_c) sein. Sie müssen „kommensurabel“ (lat.: zusammen messbar) sein:

$$NT_S = LT_1 = NLT_c$$

$$\Rightarrow T_c = \text{gcf}(T_S, T_1) \Rightarrow L = T_S/T_c \quad \text{und} \quad N = T_1/T_c$$

oder $f_c = \text{gcf}(f_1, f_S) \Rightarrow L = f_1/f_c \quad \text{und} \quad N = f_S/f_c$

gcf ist der „greatest common factor“, also der größte gemeinsame Faktor oder Teiler von T_1 und T_S (im Allgemeinen nicht ganzzahlig). Damit erhält man die kleinste Lösung für N, L ; Vielfache davon $mL = mNf_1/f_S$ mit $m = 2, 3, \dots$ sind natürlich immer möglich.

Mathematisch ausgedrückt:

$$\exists T_c \in \mathbb{R} \text{ so, dass } T_1 = NT_c \wedge T_S = LT_c \text{ mit } T_1, T_S \in \mathbb{R} \text{ und } N, L \in \mathbb{Z}.$$

$$\Rightarrow \frac{T_1}{T_S} = \frac{f_S}{f_1} = \frac{N}{L} = c \in \mathbb{Q}.$$

Listing M1.3 zeigt eine rekursive Python-Implementierung der gcf - Funktion. Der Variablen b wird dabei in so lange der Rest der ganzzahligen Division a % b zugewiesen (aus numerischen Gründen auf 6 Stellen gerundet) bis dieser Null ist.

f_S/Hz	n	0	1	2	3	4	5	6	7	8	9
25	$x_1[n]$	2,00	2,00	2,00	2,00	2,00	2,00	2,00	2,00	2,00	2,00
200	$x_2[n]$	2,00	1,50	1,00	1,50	2,00	1,50	1,00	1,50	2,00	1,50
240	$x_3[n]$	2,00	1,63	1,07	1,15	1,75	1,98	1,50	1,02	1,25	1,85
50π	$x_4[n]$	2,00	1,29	1,17	1,98	1,43	1,08	1,92	1,57	1,02	1,83
200	$y_{MA,4}[n]$	0,40	0,70	0,90	1,20	1,60	1,50	1,40	1,50	1,60	1,50
	$y_{MA,3}[n]$	0,50	0,88	1,12	1,50	1,50	1,50	1,50	1,50	1,50	1,50
	$y_{casc}[n]$	0,22	0,61	1,11	1,33	1,44	1,50	1,56	1,50	1,44	1,50
	$y_{Int}[n]$	1,00	1,25	1,12	1,31	1,66	1,58	1,29	1,39	1,70	1,60

Tab. M1.1.: Ergebnisse zu Aufgabe 1.1

b) Filterung

i. Moving Average Filter

Das System in Abb. 1.6(a) hat die Ordnung $N = 4$ (d.h. 5 Taps); es bestimmt den gleitenden Mittelwert (*Moving Average*, MA) des Eingangssignals der jeweils letzten 5 Samples. Es hat die Impulsantwort

$$h_{MA4}[n] = \{1; 1; 1; 1; 1\}$$

¹sog. diophantische Gleichung

Wird ein Gleichsignal mit der Amplitude 1 an den Eingang gelegt, ist das Ausgangssignal konstant 5. Daher muss das Signal am Ausgang des Filters durch 5 dividiert werden, damit es korrekt skaliert ist.

Mit dem fünften Eingangsample ($n = 4$) sind alle Register (= Verzögerungsglieder) belegt, das Filter ist damit eingeschwungen. Die berechneten Werten für $y_{MA4}[n]$ passen z.B. zu folgender Gleichung:

$$y_{MA4}[n] = 1,5 \text{ V} + 0,1 \text{ V} \cos(\pi(n - 4)/2) \quad \text{für } n \geq 4$$

Hieraus liest man eine Amplitude von nur noch 100 mV für das überlagerte Brummen ab, das somit somit um den Faktor 5 gedämpft wird ($\cong 20 \log_{10} 1/5 = -14 \text{ dB}$). Der abgelesene Wert ist nur eine Schätzung, für verlässlichere Werte müsste man das Signal interpolieren und so sicherstellen, dass man auch wirklich Maxima / Minima der Funktion getroffen hat. Einfacher bekommt man das gewünschte Ergebnis im Frequenzbereich (Aufgabe 2.1).

Bei einer Ordnung von 3 lautet die Impulsantwort

$$h_{MA3}[n] = \{1; 1; 1; 1\},$$

das Ausgangssignal muss für korrekte Skalierung durch vier geteilt werden. Das Brummen im Ausgangssignal ist jetzt komplett verschwunden. Das liegt daran, dass bei einer Abtastfrequenz von $f_{S,2} = 200 \text{ Hz}$ genau eine Periode des 50 Hz Störsignals in das MA-Filter „passt“, positive und negative Halbwelle heben sich daher immer auf, das Filter hat eine perfekte Brummunterdrückung für diese Signalfrequenz.

ii. Kaskadiertes Moving Average Filter

Das System in Abb. 1.6(b) besteht aus zwei kaskadierten MA-Filtern der Ordnung $N = 2$, es hat daher ebenfalls die Ordnung $N = 4$. Die Impulsantwort

$$h_{kask}[n] = \{1; 2; 3; 2; 1\}.$$

kann man entweder durch Ausprobieren ermitteln oder indem man die Impulsantworten der beiden Teilfilter $h_1[n] = h_2[n] = \{1; 1; 1\}$ mit einander faltet. Wird ein Gleichsignal mit der Amplitude 1 an den Eingang gelegt, ist das Ausgangssignal konstant $1 + 2 + 3 + 2 + 1 = 9$. Daher muss das Signal am Ausgang des Filters durch 9 dividiert werden, damit es korrekt skaliert ist.

Mit dem fünften Eingangsample sind auch hier alle Register belegt, das Filter ist damit eingeschwungen. Hier liest man eine Amplitude von 60 mV ab und erhält

$$y_{kask}[k] = 1,5 \text{ V} - 0,06 \text{ V} \cos(\pi(k - 4)/2) \quad \text{für } k \geq 4.$$

Die Dämpfung des Brummens ist damit ca. $60/500 = 0,12 \cong -18,4 \text{ dB}$.

Das kaskadierte Filter kann durch ein einfaches FIR-Filter ersetzt werden mit den Koeffizienten $h_i = \{1; 2; 3; 2; 1\}$. Ein Nachteil der nicht-kaskadierten Form ist, dass Multiplizierer (wenn auch nur einfache) benötigt werden.

iii. Gedämpfter Integrator

Das gedämpfte Integrator in Abb. 1.6(b) ist ein rekursives System, das eine unendliche Impulsantwort hat. Das macht die manuelle Berechnung aufwändiger bis unmöglich, man kann sich behelfen indem man die Impulsantwort abbricht, also z.B. $h_{Int}[n] = 0,5^i \approx \{1; 0,5; 0,25; 0,125; 0,0625\}$.

Aufgrund der unendlichen Impulsantwort ist das Filter strenggenommen niemals eingeschwungen, man sieht aber aus Simulation oder Rechnung, dass das Ausgangssignal nach einer Weile auf die periodische Sequenz $\{1,7; 1,6; 1,3; 1,4; \dots\}$ zusteuert, aus der man eine Amplitude von 200 mV abschätzen kann.

c) Simulation mit Matlab / Python

Der Code in Listing M1.1 enthält einige Besonderheiten:

Zeile 19: Hier soll im Titel des Plots die Abtastfrequenz automatisch ausgeben werden. Dafür wird im auszugebenden String mit dem Platzhalter `%1.f` das Floating Point Format mit einer Nachkommastelle gesetzt und danach der Wert `1/T_S` übergeben (leicht abweichende Syntax). In Matlab muss man auch für einfache Formatierungen zunächst mit `sprintf()` einen String erzeugen, der hier dann für den Titel verwendet wird.

Zeile 48: Hier sollen die ersten 8 Werte des abgetasteten Signals ins Kommandofenster gedruckt werden. In Python wird dazu der auszugebende Text zwischen Hochkomma-ta übergeben, optional können dahinter weitere Formatierungsanweisungen gegeben werden. Hier wird mit `end=""` der automatische Zeilenvorschub nach dem Printkom-mando unterdrückt. In Matlab muss man auch hier zunächst mit `sprintf()` einen formatierten String erzeugen, der dann über `[str_a, str_b]` mit einem weiteren String verknüpft und mit `disp()` ausgegeben wird.

d) Kritischer Pfad

Der kritische Pfad erstreckt sich beim einfacherem MA und beim kaskadierten MA-Filter über 4 Additionen, entsprechend $\tau_{krit} = 8$ ns. Die maximale Taktfrequenz ist damit $f_{S,max} = 1/\tau_{krit} = 125$ MHz.

Beim kaskadierten MA-Filter könnte der kritische Pfad allerdings leicht auf 2 Additionen verkürzt werden, wenn man ein zusätzliches Register zwischen beide Hälften einsetzt. Damit erhält man eine maximale Taktfrequenz von 250 MHz, auf Kosten einer zusätzlichen Latenz.

Beim gedämpften Integrator sind ein Multiplizierer und ein Addierer im kritischen Pfad, entsprechend $\tau_{krit} = 12$ ns und $f_{S,max} = 1/\tau_{krit} = 83,3$ MHz.

e) Taktzyklen pro Sample

Beim MA und beim kaskadierten MA-Filter müssen pro Sample 4 Additionen durchge-führt und die Samples in einem Ringbuffer neu indiziert werden. Der Aufwand für die Speicherverwaltung soll vernachlässigt werden, daher müssen hier pro Eingangssample 4 Additionen berechnet werden. Beim gedämpften Integrator werden eine Addition und eine Multiplikation benötigt.

f) Effiziente Implementierung

Beim einfachen MA-Filter mit L Taps (Ordnung $N = L - 1$) unterscheidet sich der aktuelle Ausgangswert $y[k]$ vom vorherigen Wert $y[k - 1]$ nur durch den neuen Eingangswert $x[k]$ und Wegfall des letzten Speicherwerts $x[k - (N + 1)]$:

$$\begin{aligned} y_{MA}[k] &= \sum_{i=0}^N x[k - i] = x[k] - x[k - (N + 1)] + \sum_{i=0}^N x[k - 1 - i] \\ &= x[k] - x[k - (N + 1)] + y_{MA}[k - 1] \end{aligned}$$

Diese Gleichung lässt sich durch das Netzwerk in Abb. M1.3(b) implementieren, einer Ketten schaltung aus einem sog. Kammfilter und einem Integrator. Aus numerischen Gründen (\rightarrow Kap. 5 und Kap. 9) werden allerdings meist beide Teilfilter vertauscht, die resultierende Struktur wird *Cascaded Integrator Comb* (CIC) - Filter genannt. Sie benötigt nur zwei Addierer, die auch den kritischen Pfad ausmachen. Ein „normales“ MA-Filter der Ordnung N benötigt im Vergleich N Addierer (Abb. M1.3(a)).

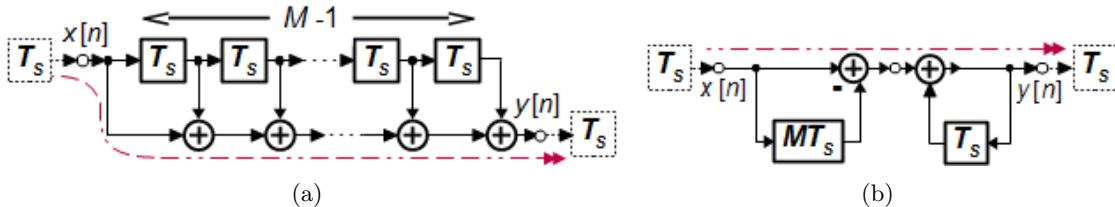


Abb. M1.3.: MA-Filter der Ordnung $M - 1$ und effiziente Implementierung mit kritischen Pfaden

g) Hochpassfilter

Ein Hochpassfilter lässt sich z.B. realisieren, indem man die Koeffizienten eines FIR-Filters wählt als $h[n] = \{1; -1; 1; -1\}$. Wenn die Summe der Koeffizienten Null ist, werden Gleichsignale unterdrückt. Warum ist das so? Berechnen Sie hierzu allgemein das Ausgangssignal eines FIR-Filters, das mit einem Gleichsignal eingeschwungen ist.

```

1 # ... Ende der gem. import-Anweisungen
2 # ----- Define variables -----
3 Ts = 1/240      # sampling period
4 f1 = 50.0        # signal frequency
5 phi0 = 0         # signal initial phase
6 tstep = 1e-3     # time step for "analog" signal
7 Tmax = max(6.0/f1, 10*Ts) # timespan for 6 signal
   periods or 10 Ts
8 N_Ts = Tmax / Ts # number of samples in Tmax
9 # -- Calculate time-vectors and input signals --
10 t = arange(0,Tmax,tstep) # (start,stop,step)
11 n = arange(0,round(N_Ts)) # sample n, step = 1
12 xt = 1.5 + 0.5*cos(2.0*pi*f1*t + phi0) # x(t).
13 xn = 1.5 + 0.5*cos(2.0*pi*f1*n*Ts + phi0) # x[n]
14 #xn = zeros(len(xn)); xn[0] = 1 # Dirac-Stoß
15 # ----- Plot "analog" and sampled signals -----
16 figure(1); grid(True)    # Turn on grid
17 xlabel(r'$t / s \rightarrow$')
18 ylabel(r'$y / V \rightarrow$')
19 title('$x(t) = 1.5 + \\\n0.5 \cos(2 \pi t \cdot 50 \text{Hz})$\\n\\n'
20 $x[n] = 1.5 + 0.5 \cos[2 \pi n \cdot 50 / %.1f]$')
21 %(1./Ts))
22 plot(t, xt, 'b-') # x(t) with blue line
23 stem(n*Ts, xn, linefmt='r-') # x[n], red stems
24 ylim(-0.1, 2.2)  # set y-limits to ymin, ymax
% Variables
Ts = 1/200.0;
f1 = 50.0;
phi0 = 0;
tstep = 1e-3;
Tmax = 6.0/f1;
N_Ts = Tmax / Ts;
%- Calculate input signals
t = 0:tstep:Tmax-tstep;
n = 0:round(N_Ts)-1;
xt=1.5+0.5*cos(2*pi*f1*t+phi0);
xn=1.5+0.5*cos(2*pi*f1*n*Ts+phi0);
%xn = zeros(N_Ts); xn(0) = 1
%--- Plot input signals ---
figure(1);
xlabel('Time (s) ->')
ylabel('Amplitude (V) ->')
ttlstr=strcat('x[n] = 1.5 + ', ...
'0.5 cos[2 pi * 50 / 200 Hz n]');
%
title(ttlstr);
grid on; hold on;
plot(t, xt, 'b');
stem(n*Ts, xn, 'r');
ylim([-0.1 2.2]);

```

```

26 # horizontal line at y = 1.5
27 plt.axhline(1.5, linestyle='--')
28 plt.subplots_adjust(top=0.88,right=0.95)
29 # ----- Impulse response -----
30 figure(2); grid(True)
31 h = [1, 1, 1, 1] # impulse response MA-filter
32 #h = np.convolve([1,1,1],[1,1,1]) # cascaded filt.
33 #h = [1, 0.5, 0.25, 0.125, 0.0625, 0.03125] # ~IIR
34 stem(range(len(h)), h, 'b-') # plot h[n]
35 xlabel(r'$n \rightarrow$');
36 ylabel(r'$h[n] \rightarrow$')
37 title(r'Impulsantwort $h[n]$')
38 # ----- Filtered signal -----
39 figure(3); grid(True)
40 #yn = np.convolve(xn,h) # convolve & scale
41 yn = sig.lfilter([1,0],[1, 0.5],xn) # IIR filter
42 stem(range(len(yn)), yn, 'b') # y[n]
43 xlabel(r'$n \rightarrow$')
44 ylabel(r'$y[n] \rightarrow$')
45 title('Gefiltertes Signal')
46 # ----- Print signal and filtered signal -----
47 print(' n :', end="")
48 for i in range(10): print('%6d' %(i), end="")
49 print('\nx[n]:', end="")
50 for i in range(10): print('%6.2f' %(xn[i]), end="")
51 print('\ny[n]:', end="")
52 for i in range(10): print('%6.2f' %(yn[i]), end="")
53 plt.show()      # draw and show the plots

```

Lst. M1.1: Python Listing zu M1.1

```

% line in DATA coordinates:
line([0 Tmax],[1.5 1.5]);
%
%--- Impulse response -----
figure(2);
%h=[1, 1, 1, 1, 1];
h=conv([1,1,1],[1,1,1]);
%h=[1,0.5, 0.25, 0.125, 0.0625]
stem([0:length(h)-1], h);
xlabel('n ->');
ylabel('h[n] ->');
title('Impulse Response h[n]');
%--- Filtered signal -----
figure(3);
yn=conv(xn,h)/5;
%
stem([0:length(yn)-1], yn);
xlabel('n ->');
ylabel('y[n] ->');
title('Filtered Signal'); grid on;
% Print signal + filtered signal
str_n = sprintf('%6d', [0:11]);
disp(['n = ', str_n]);
xn_s = sprintf('%6.3f', xn(1:12));
disp(['x[n] = ', xn_s]);
yn_s = sprintf('%6.3f', yn(1:12));
disp(['y[n] = ', yn_s]);
%

```

Lst. M1.2: Matlab Listing zu M1.1

```

1 def gcf(a, b):
2 # Greatest Common Factor function for non-integer arguments
3     while b != 0:
4         a, b = b, round((a % b),6) # das heißt: a = b; danach b = round(a % b)
5     return a
6 #
7 T_c = gcf(T_S, T_1)

```

Lst. M1.3: Python-Implementierung der gcf - Funktion zu Aufgabe M1.1

M1.2. Faltung mit einfachem FIR-Filter → Aufgabe 1.2

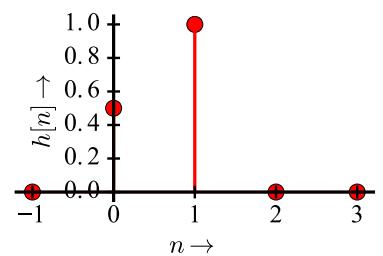
a) Die Impulsantwort $h[n] \dots$

... erhält man als Antwort auf das Eingangssignal

$x[n] = \delta[n] = 1; 0; 0; \dots$ durch „genaues Hinschauen“:

$$h[n] = 0,5; 1.$$

Bei FIR-Filters ist die Impulsantwort identisch mit den Koeffizienten des Filters: $b_0 = 0,5; b_1 = 1$.



b) Die Differenzengleichung ...

... erhält man ebenfalls durch genaues Hinschauen oder mit Hilfe des Faltungsoperators:

$$\begin{aligned} y[n] &= h[n] * x[n] = \sum_{i=-\infty}^{\infty} h[i]x[n-i] = x[n] * h[n] = \sum_{i=-\infty}^{\infty} x[i]h[n-i] \\ &= \sum_{i=0}^{N=1} h[i]x[n-i] = 0,5x[n] + x[n-1] \end{aligned}$$

Da die Impulsantwort endlich ist, muss die Faltungssumme nicht von $n = -\infty \dots \infty$ berechnet werden, sondern nur für die Indices i bei denen $h[i] \neq 0$. Für die Berechnung der DZGL oder des Ausgangssignals von FIR-Filters ist daher im Allgemeinen die Schreibweise $h[n] * x[n]$ besser geeignet.

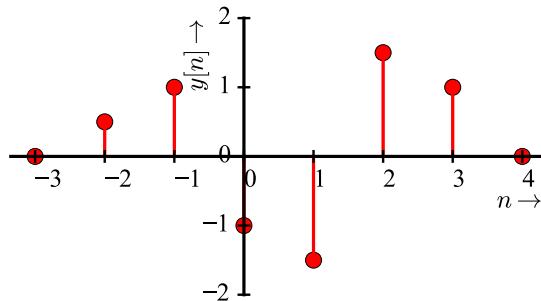


Abb. M1.4.: Ausgangssignal $y[n]$ des FIR-Filters zum Eingangssignal aus Abb. 1.7

c) Das Ausgangssignal $y[n]$ auf ein gegebenes Eingangssignal $x[n] \dots$

... erhält man auf verschiedene Weisen:

DZGL: Direkt aus der DZGL durch Einsetzen von $x[n]$:

$$\begin{aligned} y[n] &= 0,5x[n] + x[n-1] \\ &= 0,5(\delta[n+2] - 2\delta[n] + \delta[n-1] + \delta[n-2]) \\ &\quad + (\delta[n+1] - 2\delta[n-1] + \delta[n-2] + \delta[n-3]) \\ &= 0,5\delta[n+2] + \delta[n+1] - \delta[n] - 1,5\delta[n-1] + 1,5\delta[n-2] + \delta[n-3] \end{aligned}$$

Faltung: Das Ausgangssignal ist die Faltung des Eingangssignals mit der Impulsantwort, also die Summe der mit der Höhe der Eingangsimpulse bewerteten und verschobenen Impulsantworten und lässt sich durch die folgenden Schritte durchführen:

- „Umklappen“ von $h[n]$ oder $x[n]$ (d.h. Bildung von $h[-n]$ oder $x[-n]$)

- Verschieben und Bewerten
- Aufsummieren der Teilprodukte

n	-2	-1	0	1	2	3	4
$x[n]$	1	0	-2	1	1		
$0,5x[n]$	0,5	0	-1	0,5	0,5		
$+ x[n + 1]$			1	0	-2	1	1
$y[n] = \sum h[i] * x[n - i] =$	0,5	1	-1	-1,5	1,5	1	0

Tab. M1.2.: Zeitdiskrete Faltung $y[n] = h[n] * x[n]$ (Aufgabe 1.2)

Wie schon in Unterpunkt b) beschrieben, ist es am einfachsten, wenn man die Eingangssequenz $x[n]$ mit der Impulsantwort $h[n]$ bewertet und verschiebt, da $h[n]$ kurz ist (Tab. M1.2). Filtert man eine kurze Eingangssequenz mit einem IIR-Filter (unendliche Impulsantwort) wie in Aufgabe 2.9, ist die Rechnung leichter in umgekehrter Reihenfolge (Tab. M1.3). Die Ergebnisse sind aber identisch, da die Faltung kommutativ ist.

i	-3	-2	-1	0	1	2	3	4
$x[i]$	1	0	-2	1	1			
n	$\sum x[i]h[n - i] = y[n]$							
-2	$h[-2 - i]$						$0 \cdot 1 + 1 \cdot 0,5 = 0,5$	
-1	$h[-1 - i]$						$1 \cdot 1 + 0 \cdot 0,5 = 1$	
0	$h[0 - i]$						$0 \cdot 1 + (-2 \cdot 0,5) = -1$	
1	$h[1 - i]$						$-2 \cdot 1 + 1 \cdot 0,5 = -1,5$	
2	$h[2 - i]$						$1 \cdot 1 + 1 \cdot 0,5 = 1,5$	
3	$h[3 - i]$						$1 \cdot 1 + 0 \cdot 0,5 = 1$	
4	$h[4 - i]$						$0 \cdot 1 + 0 \cdot 0,5 = 0$	

Tab. M1.3.: Zeitdiskrete Faltung $y[n] = x[n] * h[n]$ (Aufgabe 1.2)

M1.3. Systemeigenschaften aus Impulsantwort → Aufgabe 1.3

Da die Systeme alle durch eine endliche Impulsantwort beschrieben werden, die zum Zeitpunkt $n = 0$ beginnt, sind sie sowohl stabil als auch kausal. Alle Systeme lassen sich durch ein transversales System in Direktform (Abb. M1.5(a)) implementieren, dessen Koeffizienten der Impulsantwort entsprechen, $h[n] = \sum_i h[i]\delta[n - i]$. Bei Systemen a) und b) kann die Symmetrie der Koeffizienten genutzt werden, um die Anzahl der Multiplizierer zu minimieren (Abb. M1.5(b)).

- a) Alle Koeffizienten haben positives Vorzeichen, daher kann das System nur TP-Charakteristik haben (gewichtete Mittelung über die letzten drei Eingangswerte). Dies wird bestätigt durch $H(z = 1) = H(f = 0) = 1$ und $H(z = -1) = H(f = f_S/2) = 0$.
- b) Abwechselnd positive und negative Vorzeichen der Koeffizienten deuten oft auf HP-Charakteristik hin. Um sicher zu gehen, sollte man sich aber auf jeden Fall $H(z = 1)$ und $H(z = -1)$ anschauen - Gegenbeispiele: $\mathbf{b} = [-0.1, 1, -0.1, 1, -0.1]$ (Bandsperre) oder $\mathbf{b} = [-0.05, 1, 1, -0.05]$ (Tiefpass). In dieser Aufgabe bestätigen $H(z = 1) = 0$ und $H(z = -1) = 1$ die HP-Charakteristik.
- c) Wie beim ersten System, deuten rein positive Koeffizienten auf eine TP-Charakteristik hin, $H_C(z = 1) = 1.5$ und $H(z = -1) = -0.5$. Hier ist allerdings eine genauere Analyse notwendig, da das System nicht linearphasig ist (keine Symmetrie der Koeffizienten).

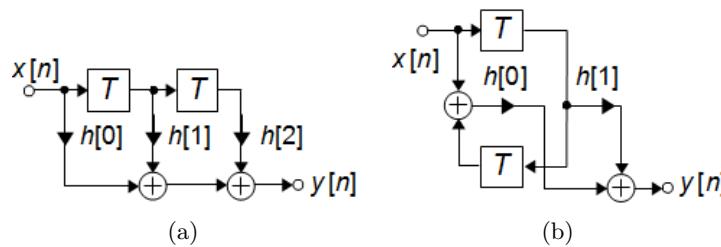


Abb. M1.5.: FIR (transversale) Filter in Direktform (a) und in „gefalteter Form“ (b)

M1.4. * Umformung von SFGs mit graphischen Methoden → Aufgabe 1.4

a) Überführung der SFGs

Man erhält die transponierten Strukturen Abb. 1.8(e) und 1.8(f) durch **Transponieren** der entsprechenden Strukturen, d.h. Vertauschen aller Signalrichtungen, Ersetzen von Addierern durch Verzweigungsknoten und umgekehrt sowie Vertauschen von Eingangs- und Ausgangssignal.

Das **Verschieben über Knoten** ist die graphische Darstellung des Distributivgesetzes (Abb. 1.4).

b) Übertragungsfunktion durch „Hinschauen“

M1.5. Kritischer Pfad und Ressourcenverbrauch → Aufgabe 1.5

Wie man an Abb. M1.6 sieht, können sich die Länge des kritischen Pfads und damit die maximale Taktrate für zwei Implementierungen der gleichen Übertragungsfunktion sowie der Ressourcenverbrauch deutlich unterscheiden:

Die DF1-Form benötigt 4 Register, die DF2-Form 2 Register. Beide Formen benötigen 5 Koeffizientenmultiplizierer und 4 Addierer. Addierer mit 3 Eingängen müssen dabei doppelt gerechnet werden. Der Ressourcenverbrauch der transponierten Formen ist identisch zu den ursprünglichen Strukturen: Die Anzahl der Register und Multiplizierer ändert sich nicht durch das Transponieren. Bei den Strukturen in Abb. M1.6 sind Anzahl der Summierer und Verzweigungsknoten identisch², so dass sich auch hier nichts durch das Transponieren ändert.

²Addierer und Knoten mit 3 Zweigen müssen doppelt gerechnet werden!

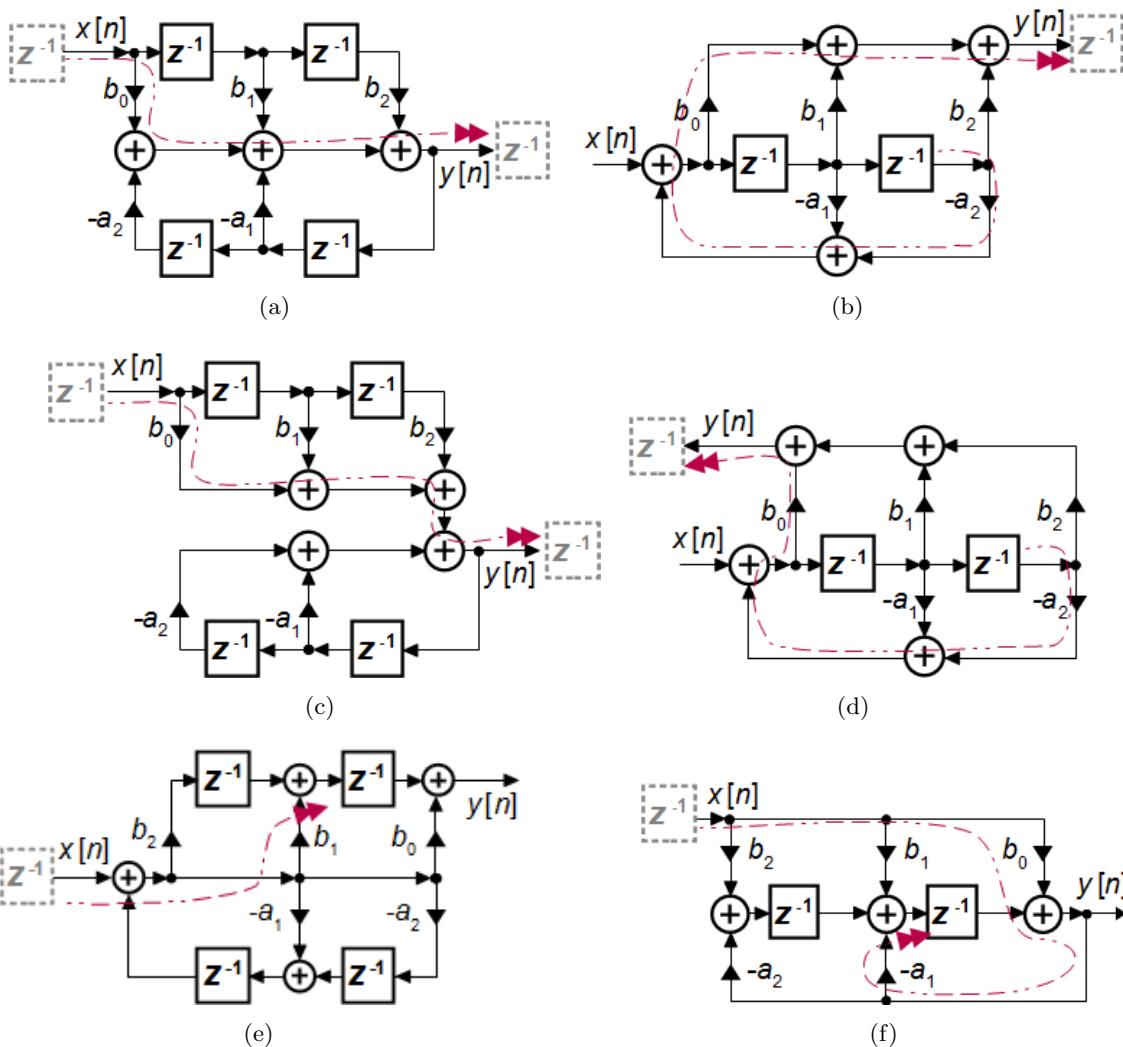


Abb. M1.6.: Systeme zweiter Ordnung in Direktform 1 (a), (c) und (e, transponiert) und Direktform 2 (b), (d) und (f, transponiert) mit kritischen Pfaden

Extra: Ist das bei allen Strukturen so? Versuchen Sie, ein Gegenbeispiel zu konstruieren!

Ein IIR-Filter N -ter Ordnung hat im allgemeinen Fall (keine speziellen Koeffizienten wie 0, 1 oder $1/2$) $N + 1$ Koeffizientenmultiplizierer im nicht-rekursiven oder transversalen Pfad (Zähler). Im rekursiven Pfad (Nenner) ist ein Koeffizient immer 1 (warum?), daher gibt es hier maximal N Multiplizierer. Insgesamt werden daher benötigt:

(T)DF1: $2N$ Register, $2N$ Addierer, $2N + 1$ Koeffizientenmultiplizierer

(T)DF2: N Register, $2N$ Addierer, $2N + 1$ Koeffizientenmultiplizierer

Da die DF2-Struktur den minimalen Ressourcenverbrauch hat, wird sie auch *kanonische* (= minimale) Form genannt.

Der kritische Pfad von Abb. M1.6(a) hängt davon ab, in welcher Reihenfolge der Addierer mit drei Eingängen aufgelöst wird: Den kürzesten Pfad bekommt man, wenn man zuerst die Multiplikationsergebnisse von a_1 und b_1 addiert und das Zwischenergebnis mit dem Ergebnis der Multiplikationen mit a_2 und b_0 aufsummiert.

In Abb. M1.6(c) sind die Additionen bereits „baumartig“ zusammengefasst, der kritische Pfad lässt sich daher nicht weiter verkürzen. Der kritische Pfad von Abb. M1.6(b) ist länger als der von Abb. M1.6(d), da hier alle Addierer im rekursiven *und* im transversalen Pfad durchlaufen werden müssen. Bei den transponierten Strukturen sind Addierer und Register verschachtelt, daher hängt die Länge des kritischen Pfades in Abb. M1.6(e) und M1.6(f) nicht von der Ordnung des Filters ab.

$$\begin{aligned}\tau_{krit,DF1(a,c)} &= \tau_{mul} + 3\tau_{add} = 12 \text{ ns} + 9 \text{ ns} = 21 \text{ ns} & \Rightarrow f_{S,max} &= 48 \text{ MHz} \\ \tau_{krit,DF2(b)}(N=2) &= 2\tau_{mul} + 4\tau_{add} = 24 \text{ ns} + 12 \text{ ns} = 36 \text{ ns} & \Rightarrow f_{S,max} &= 28 \text{ MHz} \\ \tau_{krit,DF2(d)}(N=2) &= 2\tau_{mul} + 3\tau_{add} = 24 \text{ ns} + 9 \text{ ns} = 33 \text{ ns} & \Rightarrow f_{S,max} &= 30 \text{ MHz} \\ \tau_{krit,TDF1(e)}(N=2) &= \tau_{mul} + 2\tau_{add} = 12 \text{ ns} + 6 \text{ ns} = 18 \text{ ns} & \Rightarrow f_{S,max} &= 56 \text{ MHz} \\ \tau_{krit,TDF2(f)}(N=2) &= 2\tau_{mul} + 2\tau_{add} = 24 \text{ ns} + 6 \text{ ns} = 30 \text{ ns} & \Rightarrow f_{S,max} &= 33 \text{ MHz}\end{aligned}$$

Bei Filtern höherer Ordnung machen sich diese Effekte besonders stark bemerkbar. Allgemein:

$$\begin{aligned}\tau_{krit,DF1(a,c)}(N) &= \tau_{mul} + (N+1)\tau_{add} \\ \tau_{krit,DF2(b)}(N) &= 2\tau_{mul} + 2N\tau_{add} & \Rightarrow f_{S,max} &= 13,9 \text{ MHz für } N=8 \\ \tau_{krit,DF2(d)}(N) &= 2\tau_{mul} + (N+1)\tau_{add} \\ \tau_{krit,TDF1(e)}(N) &= \tau_{mul} + 2\tau_{add} & \Rightarrow f_{S,max} &= 56 \text{ MHz für } N=8 \\ \tau_{krit,TDF2(f)}(N) &= 2\tau_{mul} + 2\tau_{add}\end{aligned}$$

Zugrunde liegt aber immer die gleiche Übertragungsfunktion mit identischen Pol- und Nullstellen und Frequenzgang (-> Kap. 2)!!

Bei allen Strukturen müssen 5 Multiplikationen und 4 Additionen pro Inputsample berechnet werden. Unterschiede zwischen verschiedenen Implementierungen können sich dadurch ergeben, dass Symmetrie oder spezielle Koeffizientenwerte ausgenutzt werden (linearphasige Filter, Halbbandfilter -> Kap. 2 und 4) oder durch Änderung der Samplerate im Filter (Kap. 7 - 9). In letzterem Fall können sich auch die Anzahl der Input- und Outputsamples pro Zeiteinheit unterscheiden.

M1.6. * Kritischer Pfad und Ressourcenverbrauch 2 → Aufgabe 1.6

- a) Die Implementierung in Abb. 1.9(a) benötigt 1 Register, 1 Multiplizierer und 3 Addierer. Die maximale Verzögerungszeit von zwei Additionen und einer Multiplikation tritt an zwei Pfaden auf und ist

$$\tau_{krit,a} = \tau_{mul} + 2\tau_{add} = 12 \text{ ns} + 6 \text{ ns} = 18 \text{ ns} \Rightarrow f_{S,max} = 56 \text{ MHz}.$$

- b) Die Implementierung des Systems in Abb. 1.9(b) ist ein rekursives System zweiter Ordnung, benötigt aber nur einen Multiplizierer, da die übrigen Koeffizienten Werte von 0, -1 oder 2 haben und sich ohne Multiplizierer implementieren lassen. Die maximale Verzögerungszeit von zwei Additionen und einer Multiplikation tritt im Rückkopplungspfad auf:

$$\tau_{krit,b} = \tau_{mul} + 2\tau_{add} = 12 \text{ ns} + 6 \text{ ns} = 18 \text{ ns} \Rightarrow f_{S,max} = 56 \text{ MHz}$$

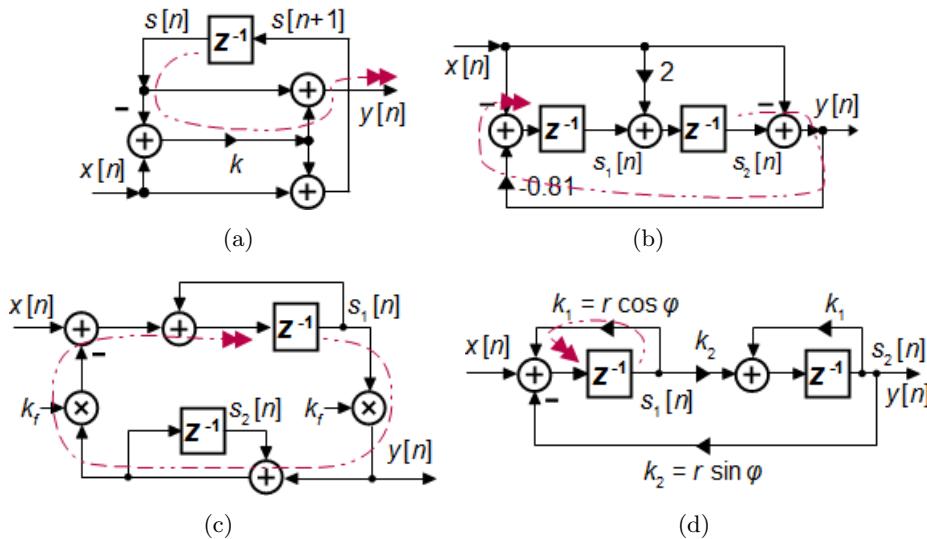


Abb. M1.7.: Kritische Pfade zu Aufgabe M1.6

Die Struktur in Abb. 1.9(b) erhält man durch Transponieren einer DF2-Struktur, sie wird daher *transponierte Direktform 2* (TDF2) genannt. Da hier Register und Addierer abwechseln, hat die TDF2 Struktur kurze kritische Pfade und ist dementsprechend schnell.

- c) Die Implementierung des Systems in Abb. 1.9(c) benötigt 2 Register, 2 Multiplizierer und 3 Addierer. Der kritische Pfad geht über von drei Additionen und zwei Multiplikationen:

$$\tau_{k\text{rit},c} = 2\tau_{mul} + 3\tau_{add} = 24 \text{ ns} + 9 \text{ ns} = 33 \text{ ns} \Rightarrow f_{S,\text{max}} = 30 \text{ MHz}$$

- d) Die Implementierung der Struktur Abb. 1.9(d) benötigt 2 Register, 4 Multiplizierer und 3 Addierer. Die maximale Verzögerungszeit von zwei Additionen und einer Multiplikation tritt im Rückkopplungspfad auf. Der Addierer mit den drei Eingängen muss dabei als zwei normale Addierer gerechnet werden:

$$\tau_{k\text{rit},d} = 1\tau_{mul} + 2\tau_{add} = 12 \text{ ns} + 6 \text{ ns} = 18 \text{ ns} \Rightarrow f_{S,\text{max}} = 56 \text{ MHz}$$

M1.7. Transponierte Systeme zu Aufgabe 1.6 → Aufgabe 1.7

Die transponierten Formen der Systeme aus Abb. 1.9 sind in Abb. M1.8 dargestellt. Die „Spiegelung“ der Darstellung um den gewohnten Signalfluss von links nach rechts zu erhalten, ist z.B. in der Prüfung nicht erforderlich!

Der kritische Pfad des transponierten Systems unterscheidet sich meist von dem des ursprünglichen Systems, Impulsantwort, Übertragungsfunktion, P/N-Diagramm sind bei idealer Arithmetik (keine Quantisierung) aber gleich. Auch die benötigten Hardwareressourcen sind identisch (siehe auch Hinweise zu Aufgabe M1.5). Auch bei dieser Aufgabe sind die Anzahl der Abzweigungen und der Addierer der transponierten und der ursprünglichen Strukturen gleich: Bei *einem* Eingang und *einem* Ausgang (single-input, single-output, SISO) ist das zwangsläufig so, da alle Abzweigungen wieder zu einem Ausgang zusammengeführt werden müssen. Hat das System eine unterschiedliche Zahl von Ein- und Ausgängen (multiple-in, multiple-out, MIMO), kann sich die Anzahl der Abzweigungen und der Addierer unterscheiden.

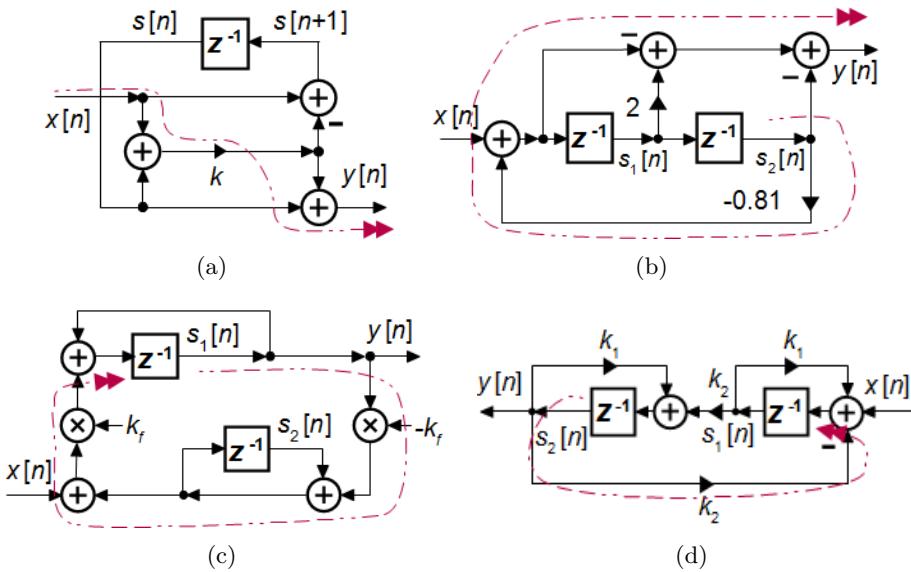


Abb. M1.8.: Transponierte Filterstrukturen zu Abb. 1.9 mit kritischen Pfaden

M1.8. Allgemeine Fragen zu FPGAs → Aufgabe 1.8

- a) **Vor- und Nachteile von FPGAs:** Im Vergleich zu ASICs sind die Einmalkosten (Non-Recurring Expense oder Engineering, NRE) bei FPGAs deutlich günstiger, da die Maskenkosten entfallen und auch die Entwicklungskosten im Allgemeinen dank optimierter Software geringer sind. Aufgrund der noch höheren Flexibilität (optimierte Technologie und Hardware) lassen sich dafür mit ASICs geringere Chipfläche und Verlustleistung (bei gleicher Technologie) sowie noch höhere Performance erzielen. Bei Stückzahlen < 100.000 sind dennoch FPGAs bei vergleichbarer Performance meist deutlich günstiger.

Im Vergleich zu DSPs und uCs sind FPGAs deutlich flexibler (beliebige Wortlängen, bei Bedarf Parallelisierung für extrem hohe Datenraten, verschiedene Taktraten auf dem gleichen Chip) und ermöglichen passgenaue Kompromisse zwischen Datenrate und Leistungsverbrauch. DSPs und uC sind für Standardanwendungen leichter zu entwickeln und im mittleren Performancesegment auch meist kostengünstiger.

- b) **Hardmakros:** Speziell Multiplizierer und Speicher benötigen viel Fläche wenn sie aus Standard FPGA-Blöcken synthetisiert werden. Da beide Komponenten für schnelle Signalverarbeitung immer benötigt werden, wird meist eine gewisse Anzahl layoutoptimiert als Hardmakros im FPGA-Fabric verteilt. Hardmakros sind zwar deutlich schneller, kompakter und verlustleistungssärmer als synthetisierte Softmakros, können aber nicht anderweitig genutzt werden und liegen brach, falls nicht alle Multiplizierer und Speicherblöcke benötigt werden.

- c) **FPGA-Technologien:**

Standard-CMOS: Kann in der jeweils aktuellsten Technologie gefertigt werden, ist daher am schnellsten (oder am kostengünstigsten, wenn ältere Technologien verwendet werden). Nachteile: Das Programmierfile wird in SRAM-Zellen gespeichert, beim Abschalten der Stromversorgung verlieren Speicherzellen den Inhalt und müssen beim

nächsten Power-Up aus externem PROM neu geladen werden -> Auslesen des Programmierfiles möglich, FPGA-Ports sind beim Einschalten u.U. in undefiniertem Zustand. Weiterer Nachteil: Leckströme in SRAM-Zellen erhöhen statische Verlustleistung.

Flash: Programmierinformationen werden in On-Chip Flash-Speicher abgelegt und stehen beim Einschalten sofort zur Verfügung. Nachteil: Komplexere Technologie erhöht Kosten, kleinere Strukturgrößen stehen immer erst ca. 1 1/2 Jahre später als bei Standard-CMOS zur Verfügung.

Antifuse: Radiation-hard (Programmierbits können durch ionisierende Strahlung nicht gekippt werden, Nachteile: In Technologien < 180 nm nicht möglich, keine Neuentwicklungen mehr, nur einmal programmierbar. Seit 2010 werden daher zunehmend rad-hard FPGAs in Flash-Technologie angeboten, die Ausfälle durch redundante Strukturen verhindern sollen. Hierzu werden z.B. Rechnungen in drei identischen Rechenwerken durchgeführt und das richtige Ergebnisse durch Mehrheitsentscheid ausgewählt.

M2. LTF: LTI-Systeme im Frequenzbereich

M2.1. * Filterung des Sensorsignals im Frequenzbereich → Aufgabe 2.1

a) Komplexer Frequenzgang und Betragsgang

MA-Filter der Ordnung 3 und 4

Zur Bestimmung des Frequenzgangs muss $H(z)$ entlang des Einheitskreises berechnet werden, man muss also $z = e^{j\Omega}$ setzen:

$$H_{MA4}(z) = 1 + z^{-1} + z^{-2} + z^{-3} + z^{-4}$$

$$\Rightarrow H_{MA4}(z = e^{j\Omega}) = 1 + e^{-j\Omega} + e^{-2j\Omega} + e^{-3j\Omega} + e^{-4j\Omega} \quad (\text{M2.1})$$

Diese Darstellung des Frequenzgangs ist zwar korrekt, lässt sich aber kaum interpretieren. Vor allem kann man den Betragsfrequenzgang $|H(z = e^{j\Omega})|$ aus dieser Form nicht ohne weiteres bestimmen. Wenn wie in diesem Fall die Koeffizienten von z^{-i} symmetrisch zu einer gedachten Mittellinie liegen, gibt es einen einfachen Trick: Man klammert zunächst den Faktor $z^{-N/2}$ aus $H(z)$ aus (wobei N der Grad des Polynoms ist), so dass auch die Exponenten von z symmetrisch zur Mittellinie liegen („Spiegelpolynom“):

$$H_{MA4}(z) = z^{-2} \left(z^2 + z^1 + \overleftrightarrow{1} + z^{-1} + z^{-2} \right)$$

$$\Rightarrow H_{MA4}(e^{j\Omega}) = e^{-2j\Omega} \left(e^{2j\Omega} + e^{j\Omega} + \overleftrightarrow{1} + e^{-j\Omega} + e^{-2j\Omega} \right)$$

$$= e^{-2j\Omega} \left(\underbrace{e^{2j\Omega} + e^{-2j\Omega}}_{2 \cos 2\Omega} + \underbrace{e^{j\Omega} + e^{-j\Omega}}_{2 \cos \Omega} + 1 \right) \quad (\text{M2.2})$$

Die Terme werden dann paarweise mit (A.3) und (A.4) zu Sinus- oder Cosinusfunktionen (je nach Vorzeichen) zusammengefasst.

$$\Rightarrow H_{MA4}(e^{j\Omega}) = \underbrace{e^{-j2\Omega}}_{\text{lin. Phase}} \underbrace{(2 \cos 2\Omega + 2 \cos \Omega + 1)}_{\text{Amplitudengang} + \text{Vorzeichen}} \Rightarrow H(\Omega = 0) = 5 \quad (\text{M2.3})$$

Bei MA-Filtern (vor allem höherer Ordnung) kommt man mit Hilfe der endlichen geometrischen Reihe (A.15) zu einer übersichtlicheren Darstellung von (M2.1) bzw. (M2.3):

$$H_{MA4}(e^{j\Omega}) = \sum_{k=0}^4 e^{-jk\Omega} = \frac{1 - e^{-j5\Omega}}{1 - e^{-j\Omega}} = \frac{e^{-j5\Omega/2} e^{j5\Omega/2} - e^{-j5\Omega/2}}{e^{-j\Omega/2} e^{j\Omega/2} - e^{-j\Omega/2}}$$

$$= e^{-j2\Omega} \frac{\sin \frac{5\Omega}{2}}{\sin \frac{\Omega}{2}} = 5e^{-j2\Omega} \text{ di}_5(\Omega) \quad (\text{M2.4})$$

$$\text{mit } \text{di}_N(\Omega) = \frac{\sin N\Omega/2}{N \sin \Omega/2} \approx \frac{\sin N\Omega/2}{N\Omega/2} = \text{si}(N\Omega/2)$$

$\text{di}_N(\Omega)$ ist der sogenannte *Dirichlet-Kernel* (A.14), auch *periodische si-Funktion* genannt, der auch bei Spektren von zeitdiskreten Rechteck-Pulsen oder -Fenstern auftaucht. Damit

ist er das direkte Äquivalent zur si-Funktion, die das Spektrum von zeitkontinuierlichen Rechteckpulsen beschreibt (B.8).

Aus (M2.3) oder (M2.4) lässt sich jetzt der Betragsgang ableiten:

$$|H_{MA4}(e^{j\Omega})| = |(2 \cos 2\Omega + 2 \cos \Omega + 1)| = \left| \frac{\sin \frac{5\Omega}{2}}{\sin \frac{\Omega}{2}} \right| = \left| \frac{\sin \frac{5\Omega}{2}}{|\sin \frac{\Omega}{2}|} \right| = |5 \operatorname{di}_5(\Omega)|$$

Zu analogen Ergebnissen kommt man für das MA-Filter der Ordnung 3:

$$\begin{aligned} H_{MA3}(z) &= z^{-3/2} \left(z^{3/2} + z^{1/2} \xrightarrow{\quad} z^{-1/2} + z^{-3/2} \right) \\ \Rightarrow H_{MA3}(e^{j\Omega}) &= e^{-3j\Omega/2} \left(e^{3j\Omega/2} + e^{j\Omega/2} \xrightarrow{\quad} e^{-j\Omega/2} + e^{-3j\Omega/2} \right) \\ &= 2e^{-3j\Omega/2} \left(\cos \frac{3\Omega}{2} + \cos \frac{\Omega}{2} \right) \\ &= \sum_{k=0}^3 e^{-jk\Omega} = e^{-j3\Omega/2} \frac{\sin 2\Omega}{\sin \frac{\Omega}{2}} = 4e^{-j3\Omega/2} \operatorname{di}_4(\Omega) \end{aligned}$$

Auch hier lässt sich leicht die Betragsfunktion ableiten.

Kaskadiertes MA-Filter

Das kaskadierte MA-Filter kann man ausgehend von der äquivalenten Direktform untersuchen

$$\begin{aligned} H_{casc}(z) &= 1 + 2z^{-1} + 3z^{-2} + 2z^{-3} + z^{-4} \\ \Rightarrow H_{casc}(z = e^{j\Omega}) &= 1 + 2e^{-j\Omega} + 3e^{-2j\Omega} + 2e^{-3j\Omega} + e^{-4j\Omega} \\ &= e^{-j2\Omega} (2 \cos 2\Omega + 4 \cos \Omega + 3), \end{aligned}$$

für manche Analysen ist allerdings die kaskadierte Struktur günstiger:

$$\begin{aligned} H_{casc}(z) &= (1 + z^{-1} + z^{-2})^2 \\ \Rightarrow H_{casc}(z = e^{j\Omega}) &= (1 + e^{-j\Omega} + e^{-2j\Omega})^2 \\ &= (e^{-j\Omega})^2 (e^{j\Omega} + 1 + e^{-j\Omega})^2 = e^{-j2\Omega} (2 \cos \Omega + 1)^2 \end{aligned}$$

Von beiden Darstellungen lässt sich leicht die Betragsfunktion ableiten. Um zu zeigen, dass beide Frequenzgänge identisch sind, benötigt man das Theorem $\cos 2x = 2 \cos^2 x - 1$.

Verlustbehafteter Integrator

Beim verlustbehafteten Integrator kann man den Spiegelpolynomtrick leider nicht anwenden, da das Nennerpolynom nicht symmetrisch ist. Hier kann man die komplexen Exponentenfunktionen nur mit Hilfe der Eulerschen Identität (A.2) auflösen:

$$\begin{aligned} H_{int}(z) &= \frac{1}{1 - \alpha z^{-1}} \Rightarrow H_{int}(z = e^{j\Omega}) = \frac{1}{1 - \alpha e^{-j\Omega}} \\ &= \frac{1}{1 - \alpha \cos \Omega + j\alpha \sin \Omega} \end{aligned}$$

Die Betragsfunktion muss man hier „händisch“ ausrechnen:

$$\begin{aligned}|H_{int}(z = e^{j\Omega})| &= \left| \frac{1}{1 - \alpha \cos \Omega + j\alpha \sin \Omega} \right| = \frac{1}{|1 - \alpha \cos \Omega + j\alpha \sin \Omega|} \\&= \frac{1}{\sqrt{(1 - \alpha \cos \Omega)^2 + (\alpha \sin \Omega)^2}} \\&= \frac{1}{\sqrt{1 + \alpha^2 - 2\alpha \cos \Omega}} = \frac{1}{\sqrt{1,81 - 1,8 \cos \Omega}} \quad \text{für } \alpha = 0,9\end{aligned}$$

wobei die Identität $\cos^2 \Omega + \sin^2 \Omega = 1$ ausgenutzt wurde. Man sieht, dass die Betragsfunktion monoton abnimmt wenn Ω von $0 \rightarrow \pi$ geht.

Die korrekte Skalierung bei DC ($\Omega = 0$) kann man mit der unendlichen Reihe (A.12) $\sum_{i=0}^{\infty} 0,5^i = \frac{1}{1-0,5} = 10$ ermitteln; notfalls kann man auch die angenäherte Reihe aufsummieren.

b) DC-Verstärkung

Die DC-Verstärkung eines Systems lässt sich sowohl direkt aus $H(e^{j\Omega})$ als auch aus $H(z)$ bestimmen. Allerdings werden bei der Dirichlet-Form von MA-Systemen bei $\Omega = 0$ Zähler und Nenner 0, der Funktionswert muss dann über die Regel von L'Hospital berechnet werden:

$$H_{MA4}(\Omega = 0) = \lim_{\Omega \rightarrow 0} \frac{\sin 5\Omega/2}{\sin \Omega/2} = \frac{(\sin 5\Omega/2)'}{(\sin \Omega/2)'} \Big|_{\Omega=0} = \frac{\frac{5\Omega}{2} \cos \frac{5\Omega}{2}}{\frac{\Omega}{2} \cos \frac{\Omega}{2}} \Big|_{\Omega=0} = 5$$

Analog dazu berechnet man $H_{MA3}(\Omega = 0) = 4$, $H_{casc}(\Omega = 0) = 9$, $H_{int}(\Omega = 0) = 10$.

Zu den gleichen Ergebnissen kommt man auch schneller, wenn man weiß, dass $H(\Omega = 0) = H(z = 1)$ ist und einfach $z = 1$ in die jeweilige Systemfunktion einsetzt.

c) Dämpfung bei $f = 50$ Hz

Die Dämpfung des Systems bei einer bestimmten Frequenz Ω_1 könnte man wieder direkt aus $H(e^{j\Omega_1})$ bzw. $H(z)$ bestimmen. Allerdings ist hier $z = e^{j\Omega_1}$ im Allgemeinen komplex, was die Rechnung erschwert.

Auf jeden Fall muss aus der Signalfrequenz f_1 (die hier die Frequenz des Störsignals ist) die normierte Frequenz $F_1 = f_1/f_S$ bzw. die normierte Kreisfrequenz $\Omega_1 = 2\pi F_1 = 2\pi f_1/f_S$ berechnet werden:

Hier: $f = 50$ Hz, $f_S = 200$ Hz $\Rightarrow F_1 = 50/200 = 0,25$ und $\Omega_1 = 2\pi F_1 = \pi/2$.

Mit den Ergebnissen von Unterpunkt a) erhält man:

$$\begin{aligned}|H_{MA4}(\Omega_1)| &= |2 \cos 2\Omega_1 + 2 \cos \Omega_1 + 1| \\&= |2 \cos \pi + 2 \cos \pi/2 + 1| = |2 \cdot (-1) + 2 \cdot 0 + 1| = 1 \\&\Rightarrow |H_{MA4}(\Omega_1)| / |H_{MA4}(\Omega = 0)| = 0,2\end{aligned}$$

Analog dazu:

$$\begin{aligned}|H_{MA3}(\Omega_1)| &= 2 \left| \cos \frac{3\Omega_1}{2} + \cos \frac{\Omega_1}{2} \right| = 2 \left| -\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \right| = 0 \\|H_{casc}(\Omega_1)| &= (2 \cos \Omega_1 + 1)^2 = 1 \\|H_{int}(\Omega_1)| &= \frac{1}{\sqrt{1,81 - 1,8 \cos \Omega_1}} = 0,743\end{aligned}$$

f_S (Hz)	25	200	240	50π
F_1	0	$1/4$	$5/24$	$1/\pi$
Ω_1	0	$\pi/2$	$5\pi/12$	2
$ H_{MA,4} (\Omega_1)$	—	0,2	0,043	—
$ H_{MA,3} (\Omega_1)$	—	0	0,21	—
$ H_{casc} (\Omega_1)$	—	0,11	0,26	—
$ H_{int} (\Omega_1)$	—	0,074	0,086	—

Tab. M2.1.: Dämpfungen der verschiedenen Filter aus Aufgabe M2.1 bei $f_1 = 50$ Hz bezogen auf $|H(\Omega = 0)|$ bei unterschiedlichen Abtastfrequenzen

d) Pol- und Nullstellenplan

Zur Bestimmung des P/N-Plans müssen (nicht weiter überraschend) zunächst die Pole und Nullstellen der Systemfunktionen bestimmt werden. Analytisch ist die Zerlegung eines Polynoms N -ter Ordnung in seine Linearfaktoren im Allgemeinen nicht möglich; die Beispiele dieser Aufgabe lassen sich aber noch geschlossen lösen. Zur Bestimmung der Pole und Nullstellen wandelt man am $H(z)$ in die Form mit positiven Exponenten um. Wenn man eine oder mehrere Nullstellen kennt oder erraten kann, kann man die Ordnung des Polynoms durch Polynomdivision reduzieren. MA-Filter kann man durch Umformung der endlichen geometrischen Reihe (A.5.1) in eine Form bringen, aus der man sofort Null- und Polstellen erkennen kann.

MA-Filter der Ordnung 4

Bei diesem Filter kann man Nullstellen nur schwer erraten, dafür lässt es sich bequem als endliche geometrische Reihe darstellen und umformen:

$$H_{MA4}(z) = 1 + z^{-1} + z^{-2} + z^{-3} + z^{-4} = \sum_{k=0}^{N=4} z^{-k} = \frac{1 - z^{-5}}{1 - z^{-1}} = \frac{z^5 - 1}{z^4(z - 1)}$$

$$\Rightarrow z_{0,k} = e^{2\pi k/5} \text{ für } k = 0 \dots 4 \text{ und } z_{\infty,0} = 1 \text{ sowie } z_{\infty,1\dots 4} = 0$$

Die Nullstelle $z_{0,0} = 1$ und die Polstelle $z_{\infty,0} = 1$ heben sich gegenseitig auf.

Für die Auflösung des Zählers wurde benutzt $z^N = 1 = e^{2k\pi} \Leftrightarrow z = e^{j2k\pi/N}$.

MA-Filter der Ordnung 3

Auch diese Systemfunktion lässt sich über die endliche geometrische Reihe umformen wie im letzten Fall (versuchen Sie es!), hier kann man alternativ die Nullstelle bei $z = -1$ erraten, $H_{MA3}(z = -1) = 0$. Durch Polynomdivision durch $(z + 1)$ erhält man ein Polynom zweiter Ordnung, das man leicht in Linearfaktoren zerlegen kann:

$$\begin{aligned}
 H_{MA3}(z) &= 1 + z^{-1} + z^{-2} + z^{-3} && \text{mit} \\
 &= \frac{z^3 + z^2 + z^1 + 1}{z^3} && (z^3 + z^2 + z + 1) \div (z + 1) = z^2 + 1 \\
 &= \frac{(z^2 + 1)(z + 1)}{z^3} && \underline{\quad -z^3 - z^2 \quad} \\
 &\Rightarrow z_{0,0} = -1; z_{0,1} = +j; z_{0,2} = -j; && z + 1 \\
 &z_{\infty,0..2} = 0 && \underline{\quad -z - 1 \quad} \\
 &&& 0
 \end{aligned}$$

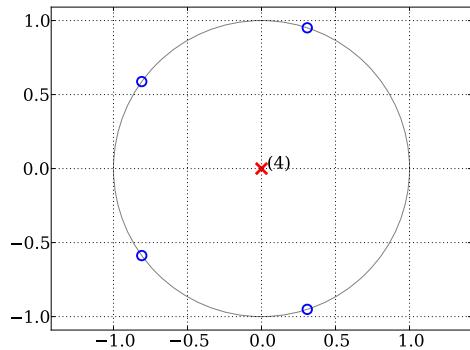
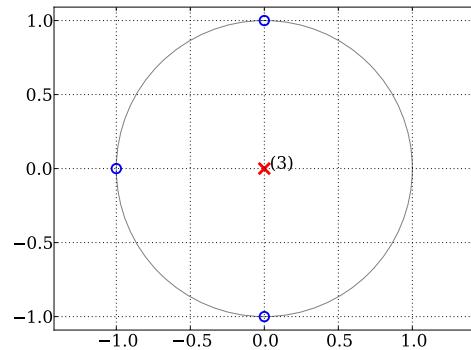
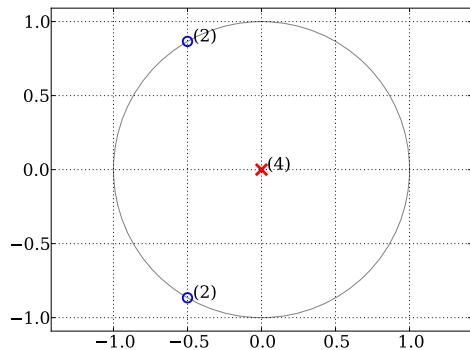
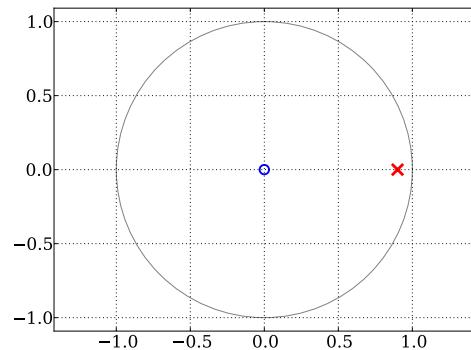
(a) MA, $N = 4$ (b) MA, $N = 3$ (c) Kaskadiertes MA, $2 \times N = 2$ (d) Verlustbehafteter Integrator, $\alpha = 0,9$

Abb. M2.1.: PN-Diagramme zu Aufgabe 2.1

Kaskadiertes MA-Filter

Wie bei der Ermittlung des Betragsgangs geht man auch für die Bestimmung der Pol- und Nullstellen am Einfachsten von der kaskadierten Form aus und löst die quadratischen

Terme mit (A.10) auf:

$$\begin{aligned}
 H_{casc}(z) &= (1 + z^{-1} + z^{-2})^2 = \frac{(z^2 + z + 1)^2}{z^4} \\
 &= \frac{(z + 0,5 \pm j\sqrt{3}/2)^2}{z^4} \\
 \Rightarrow z_{0,0,1} &= -0,5 + j\sqrt{3}/2; \quad z_{0,2,3} = -0,5 - j\sqrt{3}/2; \quad z_{\infty,0...3} = 0
 \end{aligned}$$

Integrator mit Dämpfung

Den Pol- / Nullstellenplan ermittelt man durch „genaues Hinschauen“ auf

$$H_{int}(z) = \frac{1}{1 - \alpha z^{-1}} = \frac{z}{z - \alpha} \Rightarrow z_0 = 0, \quad z_{\infty} = \alpha = 0,9$$

e) Phasengang und Gruppenlaufzeit

Aus (M2.4) lässt sich auch der Phasengang des **MA-Filters der Ordnung 4** ableiten:

$$\begin{aligned}
 \varphi_{MA4}(e^{j\Omega}) &= \angle e^{-j2\Omega} \frac{\sin \frac{5\Omega}{2}}{\sin \frac{\Omega}{2}} = -2\Omega - \angle \sin \frac{5\Omega}{2} - \angle \sin \frac{\Omega}{2} \\
 &= \underbrace{-2\Omega}_{\text{lin. Phase}} + \underbrace{\frac{\pi}{2} \left(\left(1 - \operatorname{sgn} \sin \frac{5\Omega}{2} \right) - \left(1 - \operatorname{sgn} \sin \frac{\Omega}{2} \right) \right)}_{\text{Vorzeichen des Amplitudengangs}} \\
 &= \begin{cases} -2\Omega & \text{für } 0 \leq \Omega < 2\pi/5 & \text{linearphasig mit} \\ \pi - 2\Omega & \text{für } 2\pi/5 < \Omega \leq 4\pi/5 & \text{Phasensprüngen um } \pi \\ 2\pi - 2\Omega & \text{für } 4\pi/5 < \Omega \leq 6\pi/5 & \text{bei } \Omega = 2\pi/5, \\ 3\pi - 2\Omega & \text{für } 6\pi/5 < \Omega \leq 8\pi/5 & \Omega = 4\pi/5, 6\pi/5, \\ 4\pi - 2\Omega & \text{für } 8\pi/5 < \Omega \leq 2\pi & \text{und bei } \Omega = 8\pi/5 \\ \dots & & \end{cases}
 \end{aligned}$$

Um die Phasenverschiebung bei $f_1 = 50$ Hz und $f_{S,2} = 200$ Hz herauszubekommen, muss man nur $\Omega_1 = 2\pi f_1 / f_{S,2} = \pi/2$ in obige Gleichung einsetzen:

$$\varphi_{MA4}(\Omega_1 = \pi/2) = \pi - 2\Omega_1 = 0$$

Bei dieser Frequenz hat das Ausgangssignal also die gleiche Phasenlage wie das Eingangssignal. Bei vier Frequenzen wechselt das Vorzeichen des Amplitudenterms (Nullstelle auf dem EK), bei diesen Frequenzen ändert sich die Phasendrehung um $\pm\pi$. Ob man $+\pi$ oder $-\pi$ annimmt, ist egal, da Unterschiede von 2π auf dem EK nicht sichtbar sind. Da die Phase an dieser Stelle springt, drängt sich die Frage auf: Welche Phase hat man *genau* an dieser Stelle? Die Antwort lautet: egal, da die Amplitude an dieser Stelle Null ist ...

Bei $\Omega = 0$ gibt es keinen Phasensprung - dort wechseln gleichzeitig Zähler- und Nennerfunktion ihr Vorzeichen (Nullstelle und Polstelle heben sich auf).

Die **Gruppenlaufzeit** ist die Ableitung der Phase nach $\partial\omega$ bzw. $\partial\Omega$. Vereinfacht kann man sich vorstellen, dass die Gruppenlaufzeit beschreibt, um welche Zeit ein Pulspaket im Filter verzögert wird. Bei linearphasigen Systemen ist die Gruppenlaufzeit unabhängig von der Frequenz (= wie bei einem Verzögerungsglied); sie ändern nicht die grundsätzliche Impulsform (= keine Dispersion) und sind daher gut geeignet für die Filterung

von amplituden- und phasenmodulierten Signalen. Linearphasige Filter sind immer symmetrisch zur Filtermitte aufgebaut, hier entspricht die Gruppenlaufzeit immer der Laufzeit bis zur (gedachten) Filtermitte. Bei einem linearphasigen FIR-Filter der Ordnung N ($\hat{=} N$ Verzögerungen) ist die Gruppenlaufzeit daher immer $\tau_g = NT_S/2$. An den Stellen, an denen die Phase springt, bereitet die Berechnung der Gruppenlaufzeit oft numerische Probleme (Ausreißer im Plot).

$$\begin{aligned}\tau_g(e^{j\Omega}) &= -\frac{\partial \varphi_{MA4}(e^{j\Omega})}{\partial \omega} = -T_S \frac{\partial \varphi}{\partial \Omega} = -\frac{1}{2\pi} \frac{\partial \varphi}{\partial f} \\ \Rightarrow \tau_{g,MA4}(e^{j\Omega}) &= 2T_S = \frac{2}{200 \text{ Hz}} = 10 \text{ ms}\end{aligned}$$

Zu ähnlichen Ergebnissen kommt man für den **MA-Filter der Ordnung 3**:

$$\varphi_{MA3}(e^{j\Omega}) = \begin{cases} -\frac{3}{2}\Omega & \text{für } 0 \leq \Omega < \pi/2 \quad \text{linearphasig mit} \\ \pi - \frac{3}{2}\Omega & \text{für } \pi/2 < \Omega \leq \pi \quad \text{Phasensprung um } \pi \text{ bei } \Omega = \pi/4 \\ \dots \end{cases}$$

Bei $f_1 = 50 \text{ Hz} \hat{=} \Omega_1 = \pi/2$ ist der Betragsgang 0 und die Phase nicht definiert. Bei geringfügig kleineren Frequenzen beträgt die Phasendrehung $\varphi_{MA3}(\Omega_{1-}) = -3\Omega_{1-}/2 = -3\pi/4$, bei geringfügig höheren Frequenzen $\varphi_{MA3}(\Omega_{1+}) = \pi - 3\Omega_{1+}/2 = \pi/4$.

$$\Rightarrow \tau_{g,MA3}(e^{j\Omega}) = \frac{3}{2}T_S = 7,5 \text{ ms}$$

Beim **kaskadierten MA-Filter** ist die Berechnung der Phase noch einfacher, da hier das Vorzeichen der Amplitudengleichung nicht wechselt (doppelte Nullstellen):

$$\varphi_{kask}(e^{j\Omega}) = -2\Omega \Rightarrow \tau_{g,kask}(e^{j\Omega}) = 2T_S = 10 \text{ ms}$$

Achtung: In Abb. M2.2(c) springt die Phase nur scheinbar um 2π - das liegt nur an der Phasendarstellung des Plot-Programms, das die Phase auf einen Bereich von $\pm\pi$ umbriegt (wrapped).

Beim **verlustbehafteten Integrator** ist die Rechnung komplizierter, da das System nicht linearphasig ist:

$$\begin{aligned}\varphi_{int}(e^{j\Omega}) &= -\angle \frac{1}{1 - \alpha \cos \Omega + j\alpha \sin \Omega} = \angle 1 - \angle(1 - \alpha \cos \Omega + j\alpha \sin \Omega) \\ &= 0 - \text{atan}2 \frac{\alpha \sin \Omega}{1 - \alpha \cos \Omega} = -\arctan \frac{\alpha \sin \Omega}{1 - \alpha \cos \Omega} \\ \Rightarrow \varphi_{int}(\Omega_1 = \pi/2) &= -\arctan \alpha\end{aligned}$$

$$\begin{aligned}\Rightarrow \tau_{g,int}(e^{j\Omega}) &= -\frac{\partial \varphi_{int}(\Omega)}{\partial \Omega} = \partial \left(\arctan \frac{\alpha \sin \Omega}{1 - \alpha \cos \Omega} \right) / \partial \Omega \\ &= -\frac{-\left(\frac{\alpha \sin \Omega}{1 - \alpha \cos \Omega} \right)^2 + \frac{\alpha \cos \Omega}{1 - \alpha \cos \Omega}}{\left(\frac{\alpha \sin \Omega}{1 - \alpha \cos \Omega} \right)^2 + 1}\end{aligned}$$

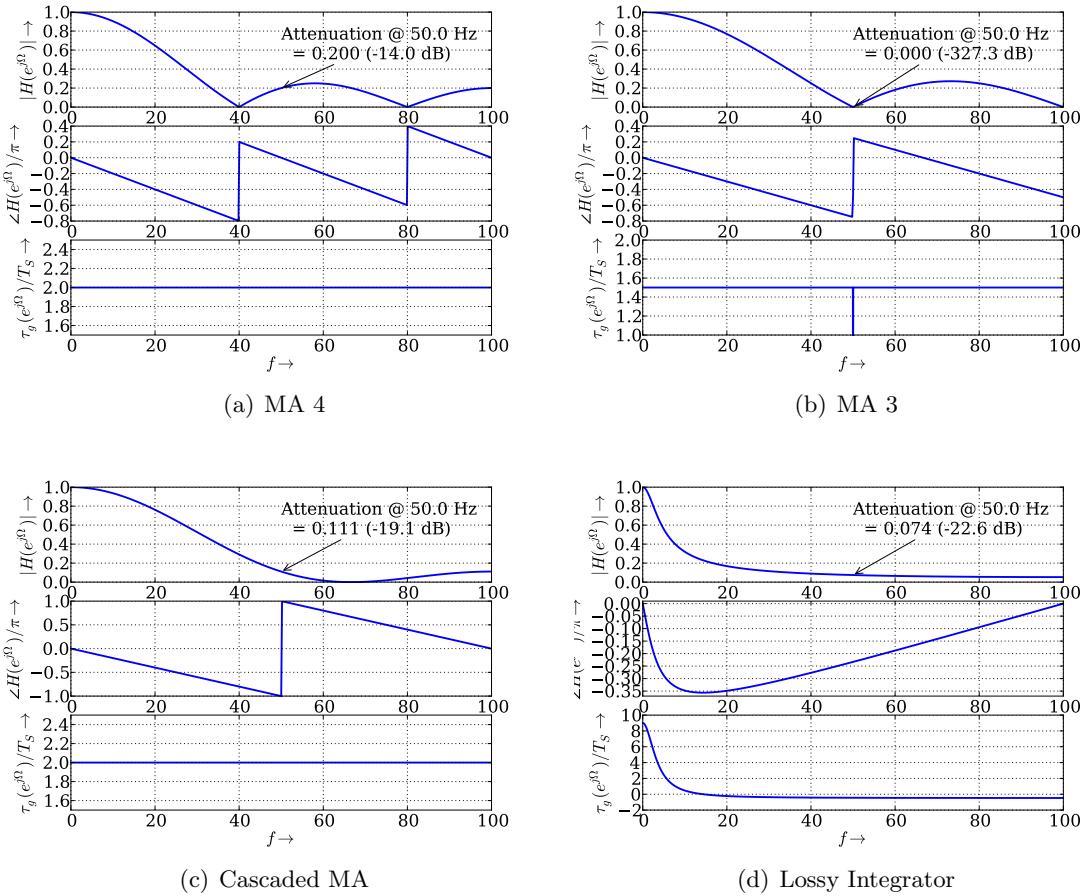


Abb. M2.2.: Betrags- und Phasengang sowie Gruppenlaufzeit zu Aufgabe 2.1

```

1 # ... Ende der Import-Anweisungen
2 Ts = 1/200.0      # sampling period
3 f1 = 50.0         # signal frequency
4 phi0 = 0          # signal initial phase
5 Tmax = 6.0/f1    # time for 6 signal periods
6 N_Ts = Tmax / Ts # No. of samples in Tmax
7 # -- Sampled input signal and filter coeffs.
8 n = arange(0,round(N_Ts)) # sample n
9 xn = 1.5 + 0.5*cos(2.0*pi*f1*n*Ts + phi0)
10 b = np.ones(4)/4; a = 1 # MA-filter, N = 5
11 #b = np.convolve([1,1,1],[1,1,1]); a = 1
12 #b = [1, 0]; a = [1, -0.9] # lossy integr.
13 #
14 # ----- P/Z-Plot -----
15 figure(1)
16 title('Pole/Zero-Plan')
17 dsp.zplane(b,a)
18 # ----- frequency response -----
% Variables
Ts = 1/200.0;
f1 = 50.0;
phi0 = 0;
Tmax = 6.0/f1;
N_Ts = Tmax / Ts;
%- input signal, filt. coeff.
n = 0:round(N_Ts)-1;
xn=1.5+0.5*cos(2*pi*f1*n*Ts+phi0);
b = ones(1,5); a = 1;
%b = conv([1,1,1],[1,1,1]); a = 1;
%b = [1, 0]; a = [1, -0.9];
%
%----- P/Z-Plot -----
figure(1);
title('Pole/Zero-Plan')
zplane(b,a);
% ----- frequency response -----

```

```

19 figure(2)
20 [W, H] = sig.freqz(b, a, whole=0);
21 f = W / (Ts * 2 * pi)
22 (w,Asig) = sig.freqz(b,a, f1*Ts*2*pi)
23 H_mx = np.max(abs(H))
24 H = H / H_mx; Asig = abs(Asig)/H_mx
25 #
26 subplot(311)
27 plot(f,abs(H))
28 ylabel(r'$|H(e^{j \Omega})| \rightarrow$')
29 title('Frequency Response')
30 plt.annotate('Attenuation @ %.1f Hz \n \
31 = %1.3f (%3.1f dB)'%(f1,Asig,20*log10(Asig)),\
32 (f1, Asig),(0.5,0.5),textcoords='axes fraction',\
33 arrowprops=dict(arrowstyle="->"))
34 subplot(312)
35 plot(f, angle(H)/pi)
36 ylabel(r'$\angle H(e^{j\Omega}) / \pi \rightarrow$')
37 subplot(313)
38 tau, w = dsp.grpdelay(b,a, nfft = 2048, Fs = 200,
39 whole=0)
40 plot(w, tau)
41 xlabel(r'$f / Hz \rightarrow$')
42 ylabel(r'$\tau_g(e^{j \Omega})/T_S \rightarrow$')
43 plt.show()      # draw and show the plots

```

Lst. M2.1: Python Listing zu M2.1

```

figure(2);
[H, W] = freqz(b, a);
f = W / (Ts * 2 * pi);
[Asig,w]=freqz(b,a,[f1*Ts*2*pi,0.5]);
H_mx = max(abs(H)); H = H / H_mx;
Asig = abs(Asig(1))/H_mx;
title('Frequency Response H(f)');
subplot(311);
plot(f,abs(H));
ylabel('|H(e^{j \Omega})| ->');
%
%
%
%
%
subplot(312);
plot(f, angle(H)/pi);
ylabel('\angle H(e^{j\Omega}) / \pi');
subplot(313);
[tau, w] = grpdelay(b,a, 2048, 200);
%
plot(w, tau);
xlabel('f \rightarrow');
ylabel('\tau_g(e^{j \Omega})/T_S ->');
%

```

Lst. M2.2: Matlab Listing zu M2.1

M2.2. Frequenzgang einfacher FIR-Filter → Aufgabe 2.2

- a) Null- und Polstellen ermittelt man am besten aus der Systemfunktion mit positiven Exponenten:

$$H(z) = 0,5 + z^{-1} = \frac{0,5z + 1}{z} = 0,5 \frac{z + 2}{z} \quad \Rightarrow \quad z_{0,1} = -2, \quad z_{\infty,1} = 0$$

Mit Null- und Polstellen kann man sofort die Systemfunktion in Produktform angeben

$$H(z) = 0,5 \frac{z - (-2)}{z - 0}$$

obwohl das bei einem System erster Ordnung wie hier nicht besonders viel neue Erkenntnisse liefert. Der P/N-Plan ist dargestellt in Abb. M2.3.

- b) Den Frequenzgang bei physikalischen Frequenzen erhält man entlang des Einheitskreises, d.h. man wertet die Systemfunktion nur aus bei $z = e^{j\Omega} = e^{j2\pi f/f_s}$. Hier startet am einfachsten mit der Systemfunktion mit negativen Exponenten (möglichst wenige Terme)

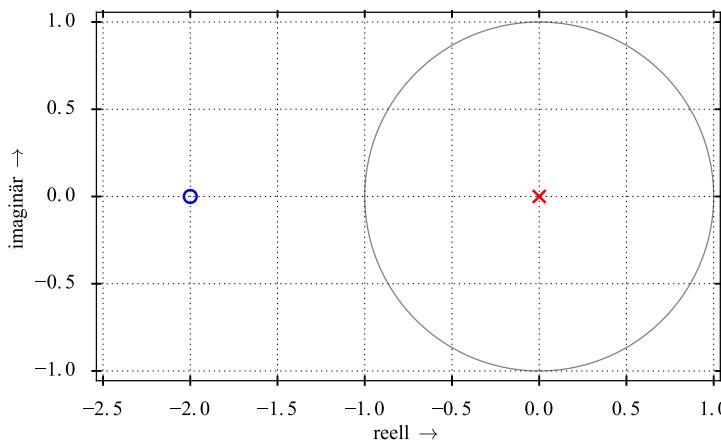


Abb. M2.3.: Pol/Nullstellenplan zu Aufgabe 2.2

und erhält den *komplexen* Frequenzgang¹ durch einfache Ersetzung:

$$\begin{aligned} H(z) &= 0,5 + z^{-1} \\ \Rightarrow H(z = e^{j2\pi f}) &= 0,5 + e^{-j2\pi f/f_S} = 0,5 + \cos 2\pi f/f_S - j \sin 2\pi f/f_S \end{aligned}$$

Hierfür wurde die Eulersche Identität benutzt,

$$e^{\pm j\Omega} = \cos \Omega \pm j \sin \Omega$$

Allerdings ist der komplexe Frequenzgang für Humanoide nicht besonders hilfreich, Betrags- und Phasengang kann man sich viel leichter vorstellen und auch darstellen:

$$\begin{aligned} |H(z = e^{j2\pi f})| &= \sqrt{\Re^2 \{H(e^{j2\pi f})\} + \Im^2 \{H(e^{j2\pi f})\}} \\ &= \sqrt{(0,5 + \cos 2\pi f/f_S)^2 + (-\sin 2\pi f/f_S)^2} \\ &= \sqrt{0,25 + \cos(\cdot) + \cos^2(\cdot) + \sin^2(\cdot)} = \sqrt{1,25 + \cos 2\pi f/f_S} \end{aligned}$$

Für die Phasenberechnung reicht die \arctan -Funktion nicht aus, da sie nur für Argumente zwischen $0 \dots \pi$ eindeutige Ergebnisse liefert, hier muss die atan2 Funktion (siehe Abb. A.1) verwendet werden!

$$\angle H(e^{j2\pi f}) = \arctan \frac{\Im \{H(e^{j2\pi f})\}}{\Re \{H(e^{j2\pi f})\}} = \frac{-\sin 2\pi f/f_S}{0,5 + \cos 2\pi f/f_S}$$

Man sieht, dass der Phasengang hier nicht-linear ist (siehe auch Abb. M2.4).

Um Betrags- und Phasengang an einzelnen Frequenzen zu berechnen, muss man die Frequenz nur in die obigen Gleichungen einsetzen:

$$\begin{aligned} f = 0 : \quad |H(f = 0)| &= \sqrt{1,25 + 1} = 1,5 & \angle H(f = 0) &= \text{atan2} \frac{0}{+0,5} = 0^\circ \\ f = \frac{f_S}{4} : \quad \left|H\left(f = \frac{f_S}{4}\right)\right| &= \sqrt{1,25 + 0} = 1,118 & \angle H\left(f = \frac{f_S}{4}\right) &= \text{atan2} \frac{-1}{0,5} = 63,43^\circ \\ f = \frac{f_S}{2} : \quad \left|H\left(f = \frac{f_S}{2}\right)\right| &= \sqrt{1,25 - 1} = 0,5 & \angle H\left(f = \frac{f_S}{2}\right) &= \text{atan2} \frac{0}{-0,5} = -180^\circ \end{aligned}$$

¹Wie fast immer in dieser Vorlesung, werden komplexe Größen nicht gesondert gekennzeichnet, da im Frequenzbereich die meisten Resultate komplex sind.

Für die speziellen Frequenzen in dieser Aufgabe ist es aber einfacher direkt die entsprechenden Werte für z in $H(z)$ einzusetzen:

$$\begin{aligned} f = 0 &\hat{=} z = 0 & \Rightarrow & H(f = 0) = H(z = 1) = 1,5 \\ f = \frac{f_S}{4} &\hat{=} z = j & \Rightarrow & H(f = \frac{f_S}{4}) = H(z = j) = 0,5 - j \\ f = \frac{f_S}{2} &\hat{=} z = -1 & \Rightarrow & H(f = \frac{f_S}{2}) = H(z = -1) = -0,5 \end{aligned}$$

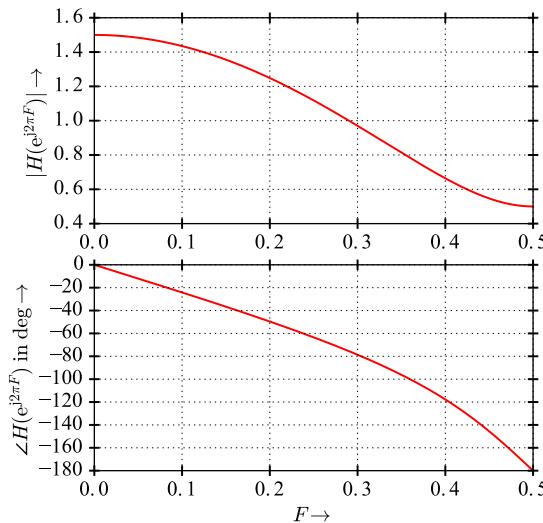


Abb. M2.4.: Betrags- und Phasengang zu Aufgabe 2.2

M2.3. * Systemfunktion aus Impulsantwort → Aufgabe 2.3

Alle Systeme lassen sich durch ein transversales System in Direktform implementieren, dessen Koeffizienten auch die Systemfunktion liefern, $H(z) = \sum h[i]z^{-i}$.

- a) Das System hat TP-Charakteristik: Dies wird bestätigt durch $H(z = 1) = H(f = 0) = 1$ und $H(z = -1) = H(f = f_S/2) = 0$.
- b) Das System hat HP-Charakteristik: $H(z = 1) = H(f = 0) = 0$ und $H(z = -1) = H(f = f_S/2) = 1$.
- c) Wie beim ersten System ergibt sich eher eine TP-Charakteristik: $H_C(z = 1) = H_C(f = 0) = 1,5$ und $H(z = -1) = H(f = f_S/2) = 0,5$. Hier ist allerdings eine genauere Analyse notwendig.

M2.4. * Systemeigenschaften aus Übertragungsfunktion → Aufgabe 2.4

System A ist ein rein transversales System (keine Rückkopplung, Pole nur im Ursprung) und damit stabil. Es ist kausal, da es nur Verzögerungen in der Impulsantwort gibt (bzw. Anzahl der Nullstellen gleich der Anzahl der Polstelle ist, $N = P$). Die

Systemfunktion wird für die Bestimmung der Impulsantwort in die Polynomform mit negativen Exponenten umgewandelt:

$$H_A(z) = \frac{(z-1)(z+1)}{z^2} = 1 - z^{-2} \Rightarrow h_b[n] = \delta[n] - \delta[n-2]$$

Aus der Impulsantwort oder der Systemfunktion in Polynomform mit negativen Exponenten können *direkt* die Koeffizienten eines FIR-Filters in *Direktform* abgelesen werden.

Die Nullstellen bei $z_{0;1,2} = \pm 1$ und die Polstellen bei $z_{\infty;1,2} = 0$ kann man leicht aus der ursprünglichen Form von $H_A(z)$ ablesen, sie führen zu Bandpass-Verhalten.

$$\begin{aligned} H_A(z=1) &= H_A(f=0) = 0, \quad H_A(z=j) = H_A(f=f_S/4) = 2 \\ H_A(z=-1) &= H_A(f=f_S/2) = 0. \end{aligned}$$

Der komplexe Frequenzgang wird entlang des Einheitskreises bestimmt, indem man $z = e^{j\Omega}$ setzt:

$$\begin{aligned} H_A(z = e^{j\Omega}) &= 1 - e^{-2j\Omega} = e^{-j\Omega} (e^{j\Omega} - e^{-j\Omega}) = 2je^{-j\Omega} \sin \Omega \\ \Rightarrow |H_A(z = e^{j\Omega})| &= |2 \sin \Omega| \\ \Rightarrow \angle H_A(z = e^{j\Omega}) &= \pi/2 - \Omega \end{aligned}$$

Durch Ausklammern von $e^{-2j\Omega}$ entsteht hier ein symmetrisches Polynom („Spiegelpolynome“), das sich mit (A.3) zu einer Sinusfunktion zusammenfassen lässt. Die Bestimmung des Phasengangs ist hier besonders einfach, da die Sinusfunktion im Intervall $\Omega = 0 \dots \pi$ das Vorzeichen nicht ändert, so dass dort die Phase nur durch den Vorfaktor $je^{-j\Omega} = e^{j(\pi/2-\Omega)}$ bestimmt wird. Die Phase hat einen konstanten Anteil $\pi/2$ und ist ansonsten proportional zur Frequenz Ω , solche Systeme nennt man *linearphasig*. Nur bei Systemen mit symmetrischen Koeffizienten lassen sich alle Exponentialterme zu Sinus- und Kosinusfunktionen zusammenfassen lassen, darum können nur solche Systeme linearphasig sein!

System B ist stabil (rein transversal, Pole nur im Ursprung) und kausal (nur Verzögerungen in Impulsantwort bzw. Anzahl der Nullstellen kleiner als Anzahl der Polstellen, $N < P$). Die Impulsantwort kann direkt aus der gegebenen Systemfunktion abgelesen werden:

$$H_B(z) = z^{-1} + z^{-3} \Rightarrow h_a[n] = \delta[n-1] + \delta[n-3]$$

Auch hier kann man aus den Koeffizienten bzw. der Impulsantwort $\{1; 0; -1\}$ ein Filter in Direktform ableiten

Aus der Impulsantwort oder der Systemfunktion in Polynomform mit negativen Exponenten können *direkt* die Koeffizienten eines FIR-Filters in *Direktform* abgelesen werden.

Das System hat eine zusätzliche Verzögerung (Latenz) von einer Sampleperiode, die entfernt werden könnte, ohne den Betragsgang zu beeinflussen.

Bestimmung von Pol- und Nullstellen aus der Polynomform mit positiven Exponenten (Erweitern mit z^3):

$$H_B(z) = \frac{z^2 + 1}{z^3} \Rightarrow z_{\infty;1,2,3} = 0; z_{0;1,2} = \pm j$$

Nullstellen bei $z = \pm j$ erzeugen Bandsperre bei $f = f_S/4$.

$$\begin{aligned} H_B(z=1) &= H_B(f=0) = 2 \\ H_B(z=j) &= H_B(f=f_S/4) = 0 \\ H_B(z=-1) &= H_B(f=f_S/2) = -2. \end{aligned}$$

Der komplexe Frequenzgang wird wie bei System A bestimmt:

$$\begin{aligned} H_B(z = e^{j\Omega}) &= e^{-j\Omega} + e^{-3j\Omega} = e^{-2j\Omega} (e^{j\Omega} + e^{-j\Omega}) = 2e^{-2j\Omega} \cos \Omega \\ \Rightarrow |H_B(z = e^{j\Omega})| &= |2 \cos \Omega| \\ \Rightarrow \angle H_B(z = e^{j\Omega}) &= \begin{cases} -2\Omega & \text{für } 0 \leq \Omega < \pi/2 & \text{linearphasig mit} \\ \pi - 2\Omega & \text{für } \pi/2 < \Omega \leq \pi & \text{Phasensprüngen um } \pi \end{cases} \end{aligned}$$

Der Betragsgang lässt sich hier bequem bestimmen, indem man $e^{-2j\Omega}$ ausklammert. Dadurch entsteht ein symmetrisches Polynom, dessen Terme sich zu einer Kosinusfunktion zusammenfassen lassen. Bei der Bestimmung des Phasengangs muss man beachten, dass die Kosinusfunktion bei $\Omega = \pi/2$ das Vorzeichen wechselt, daher springt die Phase dort um π .

System C ist nicht kausal (nur „Verfrühungen“, keine Verzögerungen bzw. $N = 3 > P = 0$), kann aber durch Verzögerung um drei Samples (Multiplikation mit z^{-3}) in ein kausales System H'_C umgewandelt werden:

$$\begin{aligned} H_C(z) &= z^3 - 3z^2 + 3z - 1 \Rightarrow h_c[n] = \delta[n+3] - 3\delta[n+2] + 3\delta[n+1] - \delta[n] \\ H'_C(z) &= H_C(z)z^{-3} = (1 - z^{-1})^3 = 1 - 3z^{-1} + 3z^{-2} - z^{-3} \\ \Rightarrow h_c[n] &= \delta[n] - 3\delta[n-1] + 3\delta[n-2] - \delta[n-3] \end{aligned}$$

Das System ist stabil (rein transversal, Pole nur im Ursprung) und hat eine dreifache Nullstelle bei $z = 1$, es unterdrückt daher sehr stark Gleichanteile (Hochpass). Die Hardwareimplementierung ergibt sich aus der Impulsantwort des verzögerten Systems, $h'_C = \{1; -3; 3; -1\}$.

$$\begin{aligned} H'_C(z=1) &= H'_C(f=0) = 0 \\ H'_C(z=j) &= H'_C(f=f_S/4) = 1 + 3j - 3 - j = -2(1 + j) = 2\sqrt{2}e^{j3\pi/4} \\ H'_C(z=-1) &= H'_C(f=f_S/2) = 8. \end{aligned}$$

$$\begin{aligned} H'_C(z = e^{j\Omega}) &= 1 - 3e^{-j\Omega} + 3e^{-2j\Omega} - e^{-3j\Omega} \\ &= e^{-3/2j\Omega} (e^{j3\Omega/2} - 3e^{j\Omega/2} + 3e^{-j\Omega/2} - e^{-j3\Omega/2}) \\ &= je^{-3/2j\Omega} (2 \sin 3\Omega/2 - 6 \sin \Omega/2) \\ \Rightarrow |H'_C(z = e^{j\Omega})| &= |2 \sin 3\Omega/2 - 6 \sin \Omega/2| \\ \Rightarrow \angle H'_C(z = e^{j\Omega}) &= -\pi/2 - 3/2\Omega \quad \text{für } 0 < \Omega \leq \pi \end{aligned}$$

Die Bestimmung des Phasengangs ist hier nicht trivial, man muss wissen, dass $|6 \sin \Omega/2| \geq |2 \sin 3\Omega/2|$ im ganzen betrachteten Intervall gilt und sich daher das Vorzeichen nicht ändert.

System D ist rekursiv, daher muss zur Untersuchung der Stabilität zunächst die Lage der Polstellen bestimmt werden:

$$H_D(z) = \frac{1}{z^2 + 0,64} = \frac{1}{(z + 0,8j)(z - 0,8j)}$$

Die Polstellen bei $\pm 0,8j$ liegen innerhalb des Einheitskreises, das System ist daher stabil. Die Polstellen bewirken eine Bandpasscharakteristik (Maximum bei $f = f_S/4$). Wegen $P = 2 < N = 0$ (die Nullstellen im Unendlichen werden ignoriert) ist das System kausal.

Die Hardwareimplementierung ergibt sich aus der Polynomform mit negativen Exponenten

$$H_D(z) = \frac{z^{-2}}{1 + 0,64z^{-2}}$$

als rein rekursives System mit zwei Verzögerungsgliedern und einem Koeffizienten mit dem Wert -0,64.

Da das System rekursiv ist, kann man die Impulsantwort in geschlossener Form nur über die inverse z -Transformation berechnen. Iterativ erhält man $h_D[n] = \{1; 0; -0,64; 0; 0,64^2; 0; -0,64^3; \dots\}$ z.B. durch Betrachten der Hardwareimplementierung.

Der Betragsgang kann hier nur über die Eulersche Identität (A.2) bestimmt werden, für den Phasengang ergibt sich eine komplizierte, nichtlineare Lösung, die hier nicht aufgeführt wird.

$$\begin{aligned} H_D(z = e^{j\Omega}) &= \frac{1}{e^{j2\Omega} + 0,64} = \frac{1}{\cos 2\Omega + j \sin 2\Omega + 0,64} \\ \Rightarrow |H_D(z = e^{j\Omega})| &= \frac{1}{\sqrt{\cos^2 2\Omega + 0,64^2 + 1,28 \cos 2\Omega + \sin^2 2\Omega}} \\ &= \frac{1}{\sqrt{1,410 + 1,28 \cos 2\Omega}} \\ \Rightarrow \angle H_D(z = e^{j\Omega}) &= \dots \end{aligned}$$

M2.5. * Systemeigenschaften aus Pol-/Nullstellenplan → Aufgabe 2.5

- a) FIR (nur Polstellen im Ursprung), daher stabil, kausal ($N \leq P$). System hat HP-Verhalten und vier Nullstellen im Frequenzband, auch bei $f = 0$.
- b) IIR, instabil (Polstellen auf EK), kausal. Um ein stabiles System daraus zu erzeugen, müssen beide Polstellen entfernt oder ein wenig in den EK hineingeschoben werden.
- c) IIR, stabil (keine Polstellen außerhalb EK) und kausal, Bandpass.
- d) IIR, stabil und akausal. Ein zusätzlicher Pol im Ursprung (Verzögerung um eine Samplingperiode) macht das System kausal ohne den Frequenzgang zu verändern. Das System hat HP-Charakteristik. Da keine Nullstelle auf dem EK liegt, wird der Frequenzgang nirgendwo Null.
- e) IIR, stabil und kausal, ansonsten wie beim vorigen System.
- f) IIR, instabil und kausal. Zwei Pole außerhalb des EK müssen entfernt werden. Da das System dann akausal wird, müssen zwei Pole im Ursprung ergänzt werden.

M2.6. * Systemeigenschaften aus Signalflussdiagramm → Aufgabe 2.6

- a) Zum Ermitteln der Übertragungsfunktion $H_a(z)$ des Systems Abb. 1.9(a) stellt man am Einfachsten zunächst die Gleichungen für $s[n+1]$ und $y[n]$ bzw. für $zS(z)$ und $y(z)$ auf:

$$\begin{aligned} s[n+1] &= -ks[n] + (1+k)x[n] \quad \circlearrowleft \bullet \quad S(z)z^{+1} = -kS(z) + (1+k)X(z) \\ y[n] &= (1-k)s[n] + kx[n] \quad \circlearrowleft \bullet \quad Y(z) = (1-k)S(z) + kX(z) \quad (\text{M2.5}) \end{aligned}$$

Man erhält daraus die Übertragungsfunktion des Systems (und kann daraus auch direkt die Größen des Zustandsraumsystems ablesen, Aufgabe 12.1):

$$\begin{aligned} S(z)z + kS(z) &= (1+k)X(z) \Rightarrow S(z) = X(z) \frac{1+k}{z+k} \\ \Rightarrow Y(z) &= X(z)(1-k) \frac{1+k}{z+k} + kX(z) = X(z) \frac{1-k^2 + kz + k^2}{z+k} \\ \Rightarrow H_a(z) &= \frac{1+kz}{z+k} = \frac{z^{-1} + k}{kz^{-1} + 1} \Rightarrow z_\infty = -k; z_0 = -\frac{1}{k} \end{aligned}$$

Das System hat eine Pol- und eine Nullstelle, die symmetrisch zum Einheitskreis sind ($z_0 = z_\infty^{-1}$) und damit eine *Allpass*-Charakteristik.

- b) Die Übertragungsfunktion des Systems in Abb. 1.9(b) lautet:

$$H_b(z) = \frac{-z^{-2} + 2z^{-1} - 1}{0,81z^{-2} + 1} = -\frac{1 - 2z + z^2}{0,81 + z^2} \Rightarrow z_{\infty,1,2} = \pm 0,9j; z_{0,1,2} = 1$$

Das System hat eine doppelte Nullstelle bei $z = 1$ ($f = 0$) und je eine Polstelle in der Nähe von $z = \pm j$ ($f = \pm f_S/4$). Aufgrund der konjugiert komplexen Polstellen ($z_{\infty,1} = z_{\infty,2}^*$) ist das System reellwertig mit Hochpass-Charakteristik (doppelte Nullstelle bei DC).

- c) Die Übertragungsfunktion des Systems in Abb. 1.9(c) lässt sich am leichtesten bestimmen, wenn man zunächst die Übertragungsfunktionen der beiden Integratoren / Akkumulatoren mit Multiplikator bestimmt, $H_{I,1}(z) = k_f z^{-1} / (1 - z^{-1}) = k_f / (z - 1)$ und $H_{I,2}(z) = k_f / (1 - z^{-1})$, und daraus dann die gesamte Übertragungsfunktion:

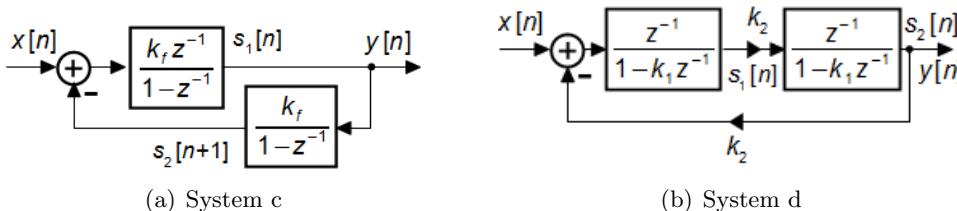


Abb. M2.5.: Blockdiagramme zu Abb. 1.9c) und d)

$$\begin{aligned} Y(z) &= X(z)H_{I,1}(z) - Y(z)H_{I,2}(z)H_{I,2}(z) \\ &= X(Z) \frac{H_{I,1}(z)}{1 + H_{I,1}(z)H_{I,2}(z)} = \frac{\frac{k_f z^{-1}}{1 - z^{-1}}}{1 + \frac{k_f z^{-1}}{1 - z^{-1}} \frac{k_f}{1 - z^{-1}}} \\ \Rightarrow H_c(z) &= \frac{Y(z)}{X(z)} = \frac{k_f z^{-1} (1 - z^{-1})}{1 - (2 - k_f^2) z^{-1} + z^{-2}} \end{aligned}$$

Die Pole von H_c sind:

$$\begin{aligned} z_{\infty 1,2} &= 1 - \frac{k_f^2}{2} \pm \frac{\sqrt{(2-k_f^2)^2 - 4}}{2} = 1 - \frac{k_f^2}{2} \pm j\sqrt{k_f^2 - k_f^4/4} \\ |z_{\infty 1,2}| &= \sqrt{\left(1 - \frac{k_f^2}{2}\right)^2 + k_f^2 - k_f^4/4} = 1 \\ \phi_{\infty 1,2} &= \arg(z_{\infty 1,2}) = \text{atan}2 \frac{\Im\{z_{\infty 1,2}\}}{\Re\{z_{\infty 1,2}\}} = \pm \arctan \frac{\sqrt{k_f^2 - k_f^4/4}}{1 - \frac{k_f^2}{2}} \\ &= \pm \arctan \frac{k_f \sqrt{1 - k_f^2/4}}{1 - \frac{k_f^2}{2}} \approx \pm \arctan k_f \text{ für } |k_f| \ll 1 \end{aligned}$$

Der Betrag der Pole ist genau 1 (auf dem Einheitskreis), unabhängig von k_f . Das System lässt sich also prinzipiell als Oszillator verwenden, dessen Frequenz über k_f eingestellt werden kann. In der Praxis führt das unvermeidliche Abschneiden oder Runden des Multiplikationsergebnisses (→ Kap. 5) dazu dass der Pol nicht genau auf dem EK bleibt; die Oszillation stirbt ab oder wird instabil. Daher sind zusätzliche Maßnahmen zur Stabilisierung der Schwingung notwendig (z.B. Integration und Rückführung des Quantisierungsfehlers). Die endliche Wortlänge des Koeffizienten k_f (Koeffizientenquantisierung) ist bei dieser Struktur übrigens unkritisch: Abweichungen vom idealen Wert führen hier nur zu einer leichten Verschiebung der Resonanzfrequenz.

- d) Das System in Abb. 1.9(d) findet man in der Literatur unter der Bezeichnung **Gold-Rader-Biquad**. Auch hier empfiehlt es sich zunächst die Teilübertragungsfunktion $H_{I,d}(z)$ der beiden gedämpften Integratoren und danach die Gesamtübertragungsfunktion mit Hilfe der Blockdarstellung Abb. M2.5(b) aufzustellen:

$$\begin{aligned} H_{I,d}(z) &= \frac{z^{-1}}{1 - k_1 z^{-1}} = \frac{1}{z - k_1} \\ Y(z) &= (X(z) - k_2 Y(z)) k_2 H_{I,d}^2(z) = X(z) \frac{k_2 H_{I,d}^2(z)}{1 + k_2^2 H_{I,d}^2(z)} \\ \Rightarrow H_d(z) &= \frac{Y(z)}{X(z)} = \frac{k_2}{H_{I,d}^{-2}(z) + k_2^2} = \frac{k_2}{(z - k_1)^2 + k_2^2} \\ &= \frac{k_2}{z^2 - 2k_1 z + k_1^2 + k_2^2} = \frac{k_2}{z^2 - 2r \cos \phi z + r^2} = \frac{k_2}{(z - re^{j\phi})(z - re^{-j\phi})} \\ \Rightarrow z_{\infty 1,2} &= re^{\pm j\phi} = r(\cos \phi \pm j \sin \phi) = k_1 \pm jk_2 \end{aligned}$$

Tiefergehende Informationen zu verschiedenen Implementierungen von IIR-Systemen zweiter Ordnung (Biquad) finden Sie u.a. in [Zöl05].

M2.7. * Einfaches Moving Average Filter → Aufgabe 2.7

- a) Impulsantwort und Systemfunktion

Durch „Hingucken“ erhält man:

$$h_{MA3}[n] = \{1; 1; 1; 1\} \quad \circlearrowleft \bullet \quad H_{MA3}(z) = 1 + z^{-1} + z^{-2} + z^{-3} = \frac{z^3 + z^2 + z + 1}{z^3}$$

Die Systemfunktion eines Moving Average (MA) - Filters entspricht einer geometrischen Reihe (A.5.1) wenn man $a = z^{-1}$ setzt und lässt sich genauso umformen und damit übersichtlicher darstellen, vor allem für MA-Filter höherer Ordnung:

$$\begin{aligned} H(z) &= \sum_{i=0}^{N-1} z^{-i} = \frac{1 - z^{-N}}{1 - z^{-1}} = \frac{z^{-N} - 1}{z^{-1} - 1} \quad \text{für } a \neq 1 \\ \Rightarrow H_{MA3}(z) &= 1 + z^{-1} + z^{-2} + z^{-3} = \sum_{i=0}^3 z^{-i} = \frac{z^{-4} - 1}{z^{-1} - 1} \end{aligned}$$

b) DC-Verstärkung und Nullstellen

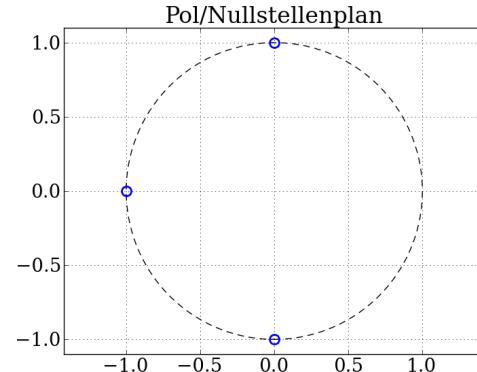
Die DC-Verstärkung erhält man direkt aus der Systemfunktion, indem man setzt

$$H_{MA3}(f = 0) = H(z = 1) = 4.$$

In Abb. 2.3(b) erkennt man eine Nullstelle bei $F = 0,5$ bzw. $f = f_S/2$ oder $z = -1$. Testweises Einsetzen in $H(z)$ bestätigt $H(z = -1) = 0$. Die übrigen Nullstellen ermittelt man durch Polynomdivision:

$$\begin{array}{r} (-z^3 + z^2 + z + 1) \div (z + 1) = z^2 + 1 \\ \hline -z^3 - z^2 \\ \hline z + 1 \\ -z - 1 \\ \hline 0 \end{array}$$

$$\Rightarrow z_{01} = -1; z_{02} = +j; z_{03} = -j;$$



Natürlich kann man bei dieser Aufgabe auch die Nullstelle bei $F = 0,25$ bzw. $z = j$ erkennen. Aber Polynomdivision zu üben ist auch nicht schlecht, oder?

c) Frequenzgang

Zur Bestimmung des Frequenzgangs muss „nur“ $H(z)$ entlang des Einheitskreises berechnet werden, man muss also $z = e^{j\Omega}$ setzen:

$$H(z) = 1 + z^{-1} + z^{-2} + z^{-3} \Rightarrow H(z = e^{j\Omega}) = 1 + e^{-j\Omega} + e^{-2j\Omega} + e^{-3j\Omega} \quad (\text{M2.6})$$

Diese Darstellung des Frequenzgangs ist zwar korrekt, lässt sich aber kaum interpretieren. Vor allem kann man den Betragsfrequenzgang $|H(z = e^{j\Omega})|$ aus dieser Form nicht ohne weiteres bestimmen. Wenn wie in diesem Fall die Koeffizienten von z^{-i} symmetrisch zu einer gedachten Mittellinie liegen, gibt es einen einfachen Trick: Man klammert zunächst den Faktor $z^{-N/2}$ aus $H(z)$ aus (wobei N der Grad des Polynoms ist) so dass auch die Exponenten von z symmetrisch zur Mittellinie liegen („Spiegelpolynom“):

$$\begin{aligned} H(z) &= z^{-\frac{3}{2}} \left(z^{\frac{3}{2}} + z^{\frac{1}{2}} \xrightarrow{\leftrightarrow} z^{-\frac{1}{2}} + z^{-\frac{3}{2}} \right) \\ \Rightarrow H(e^{j\Omega}) &= e^{-\frac{3}{2}j\Omega} \left(e^{\frac{3}{2}j\Omega} + e^{\frac{1}{2}j\Omega} \xrightarrow{\leftrightarrow} e^{-\frac{1}{2}j\Omega} + e^{-\frac{3}{2}j\Omega} \right) \\ &= e^{-\frac{3}{2}j\Omega} \left(\underbrace{e^{\frac{3}{2}j\Omega} + e^{-\frac{3}{2}j\Omega}}_{2 \cos \frac{3}{2}\Omega} + \underbrace{e^{\frac{1}{2}j\Omega} + e^{-\frac{1}{2}j\Omega}}_{2 \cos \frac{\Omega}{2}} \right) \end{aligned} \quad (\text{M2.7})$$

Die Terme werden dann paarweise mit (A.3) und (A.4) zu Sinus- oder Cosinusfunktionen (je nach Vorzeichen) zusammengefasst.

$$\Rightarrow H(e^{j\Omega}) = 2 \underbrace{e^{-j3\Omega/2}}_{\text{lineare Phase}} \underbrace{\left(\cos \frac{3}{2}\Omega + \cos \frac{1}{2}\Omega \right)}_{\text{Amplitudengang + Vorzeichen}} \Rightarrow H(\Omega = 0) = 4 \quad (\text{M2.8})$$

Aus (M2.8) kann man jetzt den Frequenzgang durch Summation der verschiedenen Sinus- bzw. Cosinusfunktionen konstruieren.

Bei MA-Filtern höherer Ordnung kommt man mit Hilfe der endlichen geometrischen Reihe (A.5.1) und $a := e^{-j\Omega}$ zu einer übersichtlicheren Darstellung von (M2.6):

$$\begin{aligned} H_{MA3}(e^{j\Omega}) &= \sum_{k=0}^3 e^{-jk\Omega} = \frac{1 - e^{-j4\Omega}}{1 - e^{-j\Omega}} = \frac{e^{-j4\Omega/2}}{e^{-j\Omega/2}} \frac{e^{j2\Omega} - e^{-j2\Omega}}{e^{j\Omega/2} - e^{-j\Omega/2}} \\ &= e^{-j3\Omega/2} \frac{\sin 2\Omega}{\sin \frac{\Omega}{2}} = 4e^{-j3\Omega/2} D_4(\Omega) \\ \text{mit } D_N(\Omega) &= \frac{\sin N\Omega/2}{N \sin \Omega/2} \approx \frac{\sin N\Omega/2}{\Omega/2} = \frac{\text{si}(N\Omega/2)}{N} \end{aligned} \quad (\text{M2.9})$$

$D_N(\Omega)$ ist der sogenannte *Dirichlet-Kernel* (A.14), auch *periodische si-Funktion* genannt, der immer auftaucht, wenn es um das Spektrum von zeitdiskreten Rechteck-Pulsen oder -Fenstern geht. Damit ist er das direkte Äquivalent zur si-Funktion, die das Spektrum von zeitkontinuierlichen Rechteckpulsen beschreibt (B.8).

Aus (M2.9) lässt sich auch leicht der Betragsgang in einer übersichtlichen Form aufschreiben:

$$|H_{MA3}(e^{j\Omega})| = \left| e^{-j3\Omega/2} \frac{\sin 2\Omega}{\sin \frac{\Omega}{2}} \right| = \left| \frac{\sin 2\Omega}{\sin \frac{\Omega}{2}} \right| = 4 |D_4(\Omega)|$$

Bei $\Omega = 0$ werden Zähler und Nenner 0, der Funktionswert an dieser Stelle wird daher über die Regel von L'Hospital berechnet:

$$H_{MA3}(\Omega = 0) = \lim_{\Omega \rightarrow 0} \frac{\sin 2\Omega}{\sin \Omega/2} = \frac{(\sin 2\Omega)'}{(\sin \Omega/2)'} \Big|_{\Omega=0} = \frac{2\Omega \cos 2\Omega}{\frac{\Omega}{2} \cos \frac{\Omega}{2}} \Big|_{\Omega=0} = 4$$

d) Phasengang und Gruppenlaufzeit

Aus (M2.8) kann man leicht den Phasengang ermitteln:

$$\varphi(e^{j\Omega}) = \begin{cases} -\frac{3}{2}\Omega & \text{für } 0 \leq \Omega < \pi/2 & \text{linearphasig mit} \\ \pi - \frac{3}{2}\Omega & \text{für } \pi/2 < \Omega \leq \pi & \text{Phasensprung um } \pi \text{ bei } \Omega = \pi/2 \end{cases}$$

Bei $\Omega = \pi/2$ (Nullstelle auf dem EK) wechselt das Vorzeichen des Sinus-Terms, bei Frequenzen $\pi/2 < \Omega \leq \pi$ hat man daher eine zusätzliche Phasendrehung von π . Da die Phase an dieser Stelle springt, drängt sich die Frage auf: Welche Phase hat man bei $\Omega = \pi/2$? Die Antwort lautet: egal, da die Amplitude an dieser Stelle Null ist ...

Die Gruppenlaufzeit ist die Ableitung der Phase nach $\partial\omega$:

$$\tau_g(e^{j\Omega}) = -\frac{\partial \varphi}{\partial \omega} = -\frac{-\partial \frac{3}{2}\Omega}{\partial \omega} = \frac{\partial \frac{3}{2}\omega T_S}{\partial \omega} = \frac{3}{2}T_S$$

Bei linearphasigen Filtern wie hier entspricht die Gruppenlaufzeit der Laufzeit bis zur (gedachten) Filtermitte.

e) Erweiterung auf N Taps mit $H(f=0)=1$

Die Systemfunktion eines Moving Average Filters mit N Taps, also der Ordnung $N-1$, und Vorfaktor $1/N$ lässt sich einfach aufstellen als:

$$\begin{aligned} H(z) &= \frac{1}{N} \sum_{i=0}^{N-1} z^{-i} = \frac{1}{N} \frac{1 - z^{-N}}{1 - z^{-1}} \\ \Rightarrow H(e^{j\Omega}) &= \frac{1}{N} \frac{1 - e^{-jN\Omega}}{1 - e^{-j\Omega}} = \frac{e^{-jN\Omega/2}}{Ne^{-j\Omega/2}} \frac{e^{jN\Omega/2} - e^{-jN\Omega/2}}{e^{j\Omega/2} - e^{-j\Omega/2}} = \frac{e^{-j(N-1)\Omega/2}}{N} \frac{\sin N\Omega/2}{\sin \Omega/2} \\ \Rightarrow |H(e^{j\Omega})| &= \frac{1}{N} \left| \frac{\sin N\Omega/2}{\sin \Omega/2} \right| = |\text{di}_N(\Omega)| \end{aligned}$$

Für $N \rightarrow \infty$ nähert sich $H(e^{j\Omega})$ immer mehr einer Dirac-Schar an (A.17), was man in Abb. M2.6 für $N = 32$ bereits gut erkennen kann. Je länger das MA-Filter wird, desto besser filtert es den Gleichanteil des zeitdiskreten Signals heraus. Allerdings steigt gleichermaßen die Gruppenlaufzeit an, das Filter wird immer träger. Für Anwendungen, bei denen ein Signal gemittelt oder geglättet (Suchbegriff: „averaging“ oder „smoothing“) werden soll, gibt es daher eine Vielzahl von Filtern mit unterschiedlichen Kompromissen bzgl. Verzögerungszeit und Frequenzgang.

f) Python / Matlab Code für MA-Filter mit $N = 32$

```

1 # ... imports wie üblich
2 M = 32                      # Anzahl Taps
3 b = ones(1, M)    # (alle Nennerkoeff. = 1)
4 a = M # Zählerkoeff.; hier nur Faktor
5 [w,H] = sig.freqz(b, a, whole=True)
6 # Frequenzgang von F = 0 ... 1 (whole)
7 plot(w / (2*pi), abs(H))
8 grid(True); plt.show()

```

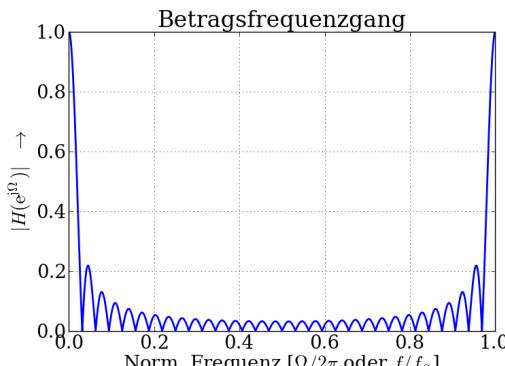
Lst. M2.3: Python Listing zu M2.7f

```

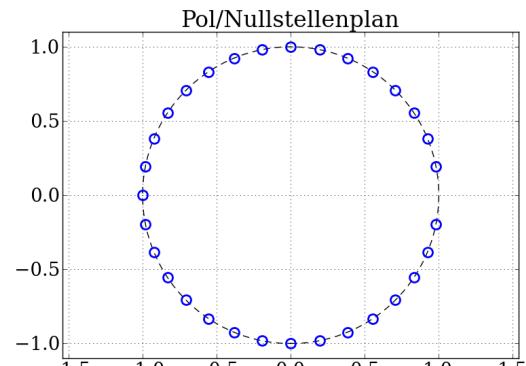
%
M = 32;
b = ones(1,M);
a = M;
[w,H]=freqz(b,a);
%
plot(w/(2*pi), abs(H));
grid on;

```

Lst. M2.4: Matlab Listing zu M2.7f



(a)



(b)

Abb. M2.6.: MA-Filter mit $N = 32$ Taps, (a) Amplitudengang und (b) P/N-Diagramm

M2.8. * Allgemeine IIR-Struktur → Aufgabe 2.8

a) Addierer, eine Verzögerung

Aus dem SFG Abb. M2.7 des Systems bestimmt man zunächst $H_a(z)$, daraus durch Setzen

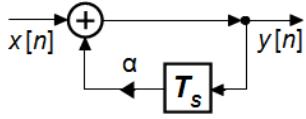


Abb. M2.7.: System a)

$$\begin{aligned}
 H_a(z) &= \frac{1}{1 - \alpha z^{-1}} = \frac{z}{z - \alpha} \Rightarrow z_0 = 0, z_\infty = \alpha \\
 \Rightarrow H_a(e^{j\Omega}) &= \frac{1}{1 - \alpha e^{-j\Omega}} = \frac{1}{1 - \alpha \cos \Omega + \alpha j \sin \Omega} \\
 \Rightarrow |H_a(e^{j\Omega})| &= \frac{1}{\sqrt{1 + \alpha^2 \cos^2 \Omega - 2\alpha \cos \Omega + \alpha^2 \sin^2 \Omega}} \\
 &= \frac{1}{\sqrt{1 + \alpha^2 - 2\alpha \cos \Omega}}
 \end{aligned}$$

von $z = e^{j\Omega}$ den komplexen Frequenzgang $H_a(e^{j\Omega})$ und daraus schließlich den Amplitudenbetragsgang $|H_a(e^{j\Omega})|$. Aus dem P/N Diagramm Abb. M2.8(a) oder aus $|H_a(e^{j\Omega})|$ sieht man, dass das Maximum des Amplitudenbetragsgangs bei $\Omega = 0$ und das Minimum bei $\Omega = \pi$ auftritt, das System ist ein Tiefpass. Man sieht aus Abb. M2.8, dass für $\alpha \rightarrow 1$

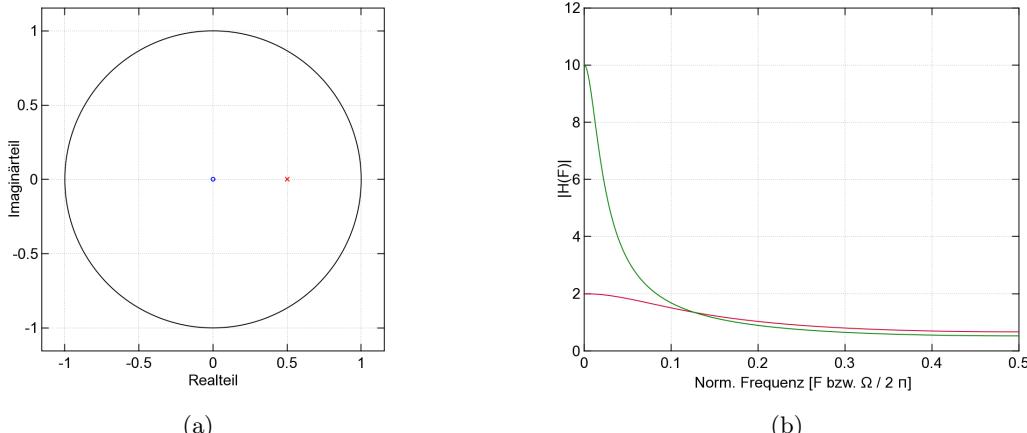


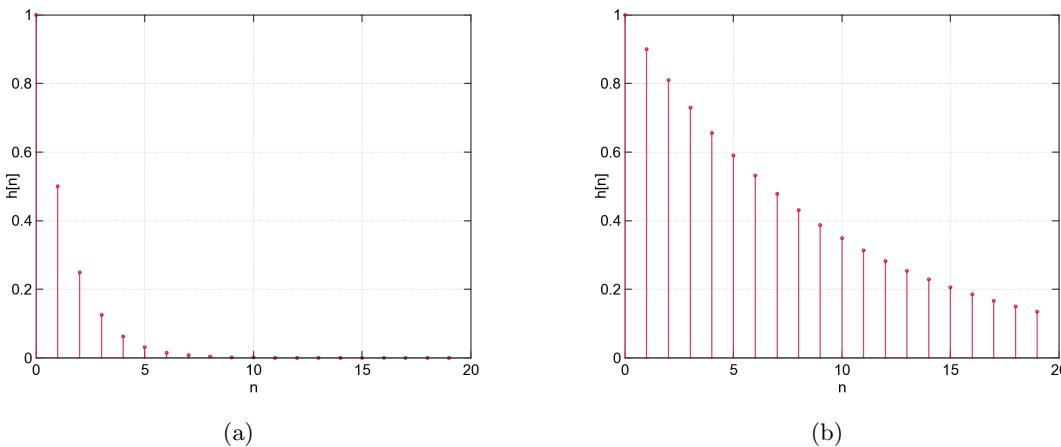
Abb. M2.8.: Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8a

die Überhöhung bei $f = 0$ immer stärker wird, gleichzeitig sinkt die Bandbreite. Maximum und Minimum von $|H_a(e^{j\Omega})|$ lassen sich entweder durch Einsetzen von $\Omega = 0$ bzw. $\Omega = \pi$ oder direkt aus $H_a(z)$ bestimmen:

$$\begin{aligned}
 |H_a(e^{j\Omega})|_{max} &= |H_a(\Omega = 0)| = |H_a(z = 1)| = \frac{1}{1 - \alpha} = \begin{cases} 2 & \text{für } \alpha = 0,5 \\ 10 & \text{für } \alpha = 0,9 \end{cases} \\
 |H_a(e^{j\Omega})|_{min} &= |H_a(\Omega = \pi)| = |H_a(z = -1)| = \frac{1}{1 + \alpha} = \begin{cases} 0,67 & \text{für } \alpha = 0,5 \\ 0,53 & \text{für } \alpha = 0,9 \end{cases}
 \end{aligned}$$

Man sieht aus Abb. M2.9, dass für $\alpha \rightarrow 1$ die Impulsantwort immer länger andauert und ungedämpfter wird. Berechnen lässt sich die Impulsantwort entweder durch „genaues Hinschauen“ direkt aus dem System Abb. M2.7 oder durch inverse z -Transformation:

$$H_a(z) = \frac{z}{z - \alpha} \quad \bullet \sim \circ \quad h_a[n] = \alpha^n u[n]$$

Abb. M2.9.: Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8a**b) Subtrahierer, eine Verzögerung**

Der Lösungsansatz ist der Gleiche wie bei Unterpunkt a), das resultierende System ist

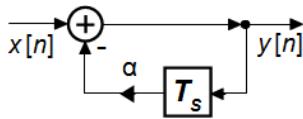


Abb. M2.10.: System b)

$$H_b(z) = \frac{1}{1 + \alpha z^{-1}} = \frac{z}{z + \alpha} \Rightarrow z_0 = 0, z_\infty = -\alpha$$

$$H_b(e^{j\Omega}) = \frac{1}{1 + \alpha e^{-j\Omega}} = \frac{1}{1 + \alpha \cos \Omega - \alpha j \sin \Omega}$$

$$|H_b(e^{j\Omega})| = \frac{1}{\sqrt{1 + \alpha^2 + 2\alpha \cos \Omega}}$$

ein Hochpass mit Maximum von $|H_b(e^{j\Omega})|$ bei $\Omega = \pi$ und Minimum bei $\Omega = 0$.

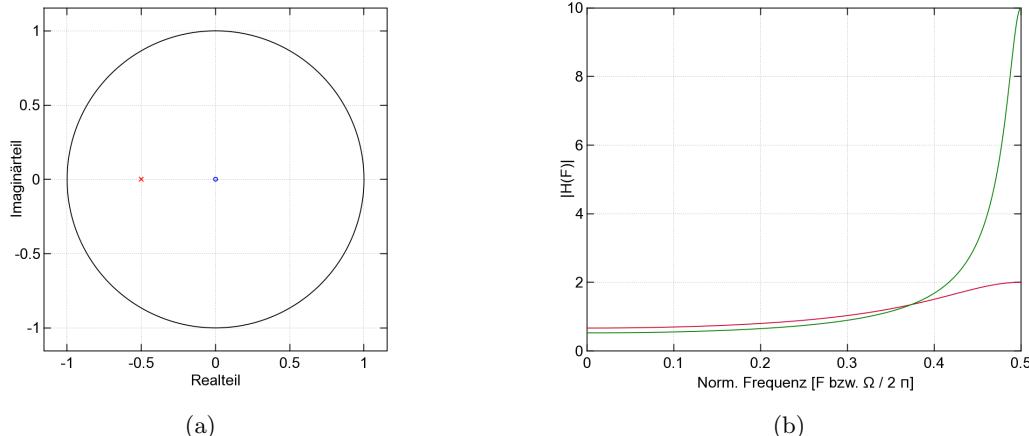


Abb. M2.11.: Pol-Nullstellen-Plan (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8b

$$|H_b(e^{j\Omega})|_{min} = |H_b(\Omega = 0)| = |H_b(z = 1)| = \frac{1}{1 + \alpha} = \begin{cases} 0,67 & \text{für } \alpha = 0,5 \\ 0,53 & \text{für } \alpha = 0,9 \end{cases}$$

$$|H_b(e^{j\Omega})|_{max} = |H_b(\Omega = \pi)| = |H_b(z = -1)| = \frac{1}{1 - \alpha} = \begin{cases} 2 & \text{für } \alpha = 0,5 \\ 10 & \text{für } \alpha = 0,9 \end{cases}$$

Im System Abb. M2.10 liegt das Maximum des Frequenzgangs bei $f_S/2$, dementsprechend

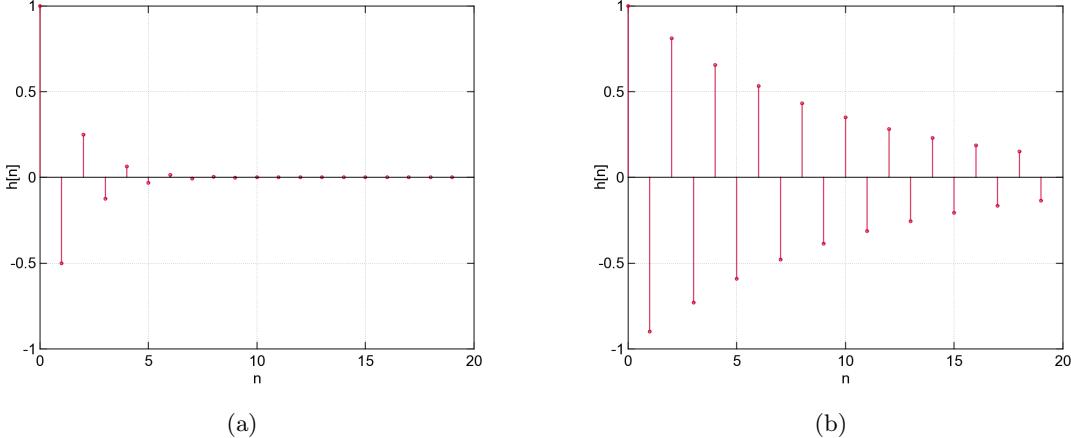


Abb. M2.12.: Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8b

zeigt auch die Impulsantwort eine gedämpfte Oszillation mit $f_S/2$. Berechnung wie bei Unterpunkt a):

$$H_b(z) = \frac{z}{z + \alpha} \quad \bullet \rightsquigarrow \quad h_b[n] = (-\alpha)^n u[n]$$

c) Addierer, zwei Verzögerungen

Die Ersetzung von T_S durch $2T_S$ (bzw. von z^{-1} durch z^{-2}) wirkt sich im Frequenzgang

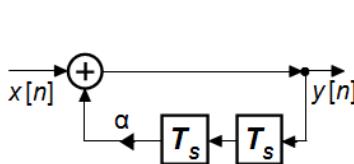


Abb. M2.13.: System c)

$$H_c(z) = \frac{1}{1 - \alpha z^{-2}} = \frac{z^2}{z^2 - \alpha} \Rightarrow z_{0,1,2} = 0, z_{\infty,1,2} = \pm\sqrt{\alpha}$$

$$H_c(e^{j\Omega}) = \frac{1}{1 - \alpha e^{-j2\Omega}} = \frac{1}{1 - \alpha \cos 2\Omega + \alpha j \sin 2\Omega}$$

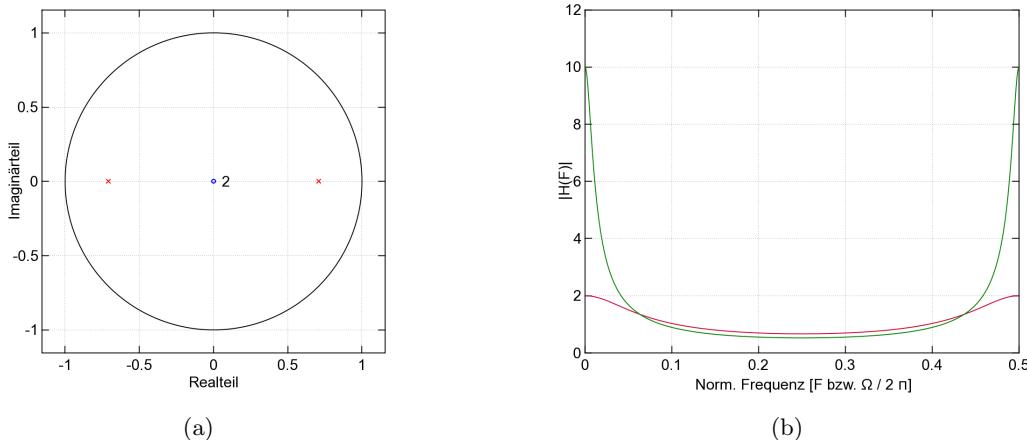
$$|H_c(e^{j\Omega})| = \frac{1}{\sqrt{1 + \alpha^2 \cos^2 2\Omega - 2\alpha \cos 2\Omega + \alpha^2 \sin^2 2\Omega}}$$

$$= \frac{1}{\sqrt{1 + \alpha^2 - 2\alpha \cos 2\Omega}}$$

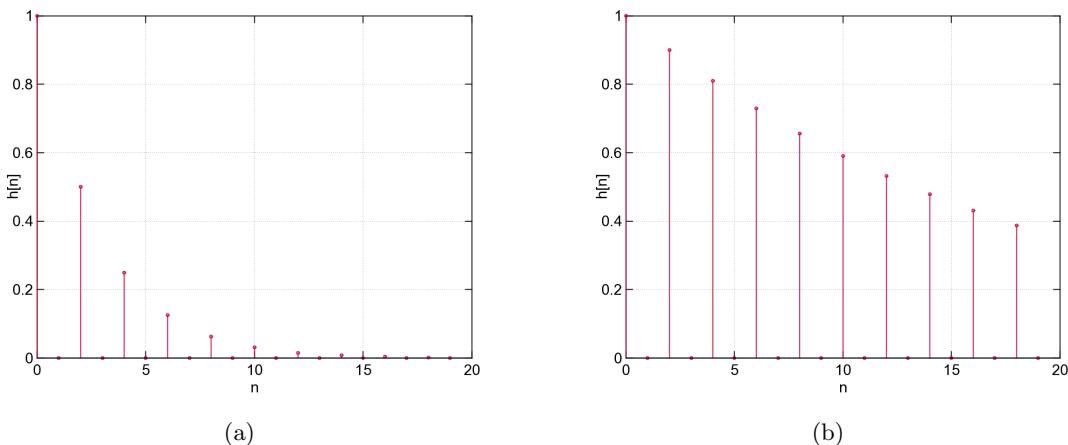
Abb. M2.14(b) aus als Transformation $H(\Omega) \rightarrow H(2\Omega)$, der Frequenzgang wird also im Vergleich zu Abb. M2.8(b) gestaucht. Das resultierende System ist eine Bandsperre mit Maxima bei $\Omega = 0$ und $\Omega = \pi$ und einem Minimum bei $\Omega = \pi/2$. Maximum und Minimum von $|H_c(e^{j\Omega})|$ lassen sich bestimmen wie bei a) und b):

$$|H_c(e^{j\Omega})|_{max} = |H_c(\Omega = 0 \text{ bzw. } \pi)| = |H_c(z = \pm 1)| = \frac{1}{1 - \alpha} = \begin{cases} 2 & \text{für } \alpha = 0,5 \\ 10 & \text{für } \alpha = 0,9 \end{cases}$$

$$|H_c(e^{j\Omega})|_{min} = |H_c(\Omega = \pi/2)| = |H_c(z = j)| = \frac{1}{1 + \alpha} = \begin{cases} 0,67 & \text{für } \alpha = 0,5 \\ 0,53 & \text{für } \alpha = 0,9 \end{cases}$$

Abb. M2.14.: Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8c

In der Impulsantwort Abb. M2.15 erzeugt die zweite Verzögerung (Transformation $z^{-1} \rightarrow z^{-2}$) zusätzliche Samples mit dem Wert Null im Vergleich zu Abb. M2.9. Im Punkt e) wird die Impulsantwort berechnet (nicht prüfungsrelevant).

Abb. M2.15.: Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8c

d) Subtrahierer, zwei Verzögerungen

Auch hier wird der Frequenzgang gestaucht, das resultierende System ist ein Bandpass

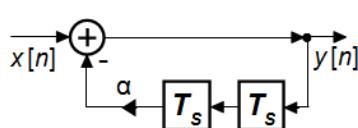


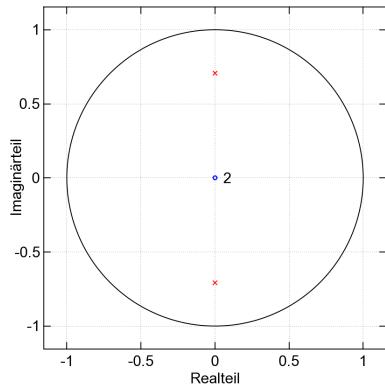
Abb. M2.16.: System d)

$$\begin{aligned}
 H_d(z) &= \frac{1}{1 + \alpha z^{-2}} = \frac{z^2}{z^2 + \alpha} \Rightarrow z_{0,1,2} = 0, z_{\infty,1,2} = \pm j\sqrt{\alpha} \\
 H_d(e^{j\Omega}) &= \frac{1}{1 + \alpha e^{-j2\Omega}} = \frac{1}{1 + \alpha \cos 2\Omega - \alpha j \sin 2\Omega} \\
 |H_d(e^{j\Omega})| &= \frac{1}{\sqrt{1 + \alpha^2 \cos^2 2\Omega + 2\alpha \cos 2\Omega + \alpha^2 \sin^2 2\Omega}} \\
 &= \frac{1}{\sqrt{1 + \alpha^2 + 2\alpha \cos 2\Omega}}
 \end{aligned}$$

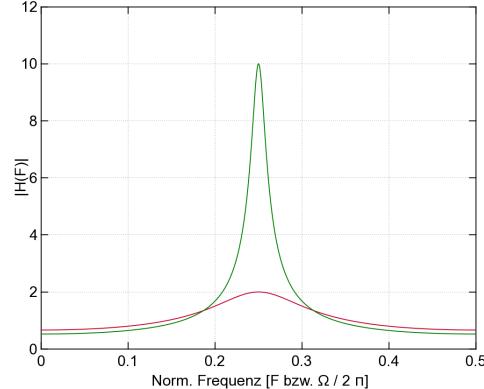
mit Maximum von $|H_d(e^{j\Omega})|$ bei $\Omega = \pi/2$ und Minima $\Omega = 0$ und $\Omega = \pi$.

$$|H_d(e^{j\Omega})|_{max} = |H_d(\Omega = \pi/2)| = |H_d(z = j)| = \frac{1}{1 - \alpha} = \begin{cases} 2 & \text{für } \alpha = 0,5 \\ 10 & \text{für } \alpha = 0,9 \end{cases}$$

$$|H_d(e^{j\Omega})|_{min} = |H_d(\Omega = 0 \text{ bzw. } \pi)| = |H_d(z = \pm 1)| = \frac{1}{1 + \alpha} = \begin{cases} 0,67 & \text{für } \alpha = 0,5 \\ 0,53 & \text{für } \alpha = 0,9 \end{cases}$$

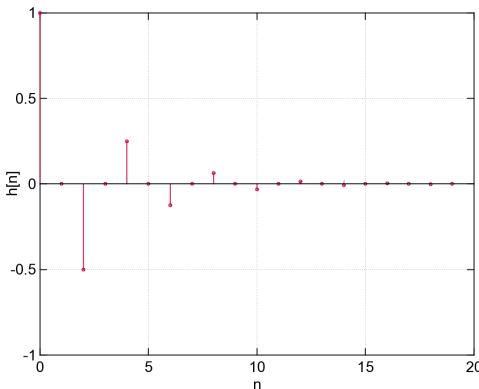


(a)

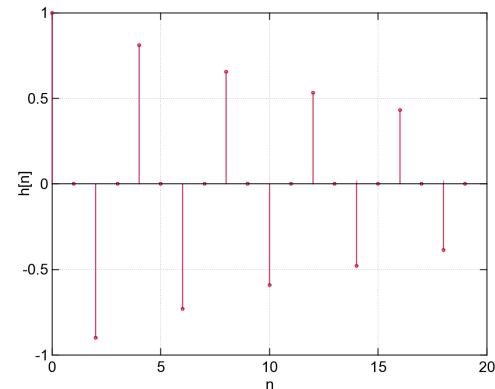


(b)

Abb. M2.17.: Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8d



(a)



(b)

Abb. M2.18.: Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8d

Zusatz: Berechnung der Impulsantwort für System c) und d) (nicht prüfungsrelevant)
Die Impulsantwort von System c) (Abb. M2.13) und d) (Abb. M2.13) kann durch Partialbruchzerlegung bestimmt werden. Dabei werden nacheinander die folgenden Schritte durchgeführt:

Normalisierung: Wenn der Zählergrad N größer oder gleich dem Nennergrad P ist², muss der Bruch zunächst durch eine Polynomdivision in die normalisierte Form gebracht werden.

²Bei praktischen DSV-Systemen ist wie hier in der Aufgabe höchstens $N = P$ möglich, sonst wird das System acausal.

Bestimmung der Pole: Der Nenner muss in Produktform dargestellt werden.

Darstellung in Summenform: Für jeden Pol wird ein Partialbruch mit zu bestimmendem Zählerkoeffizienten aufgestellt, danach werden die Partialbrüche auf einen gemeinsamen Nenner erweitert. Der Nenner ist jetzt wieder identisch zum ursprünglichen Nenner.

Bestimmung der Koeffizienten: Der umgeformte Zähler wird mit dem ursprünglichen gleichgesetzt, die Koeffizienten werden durch Koeffizientenvergleich ermittelt.

System c):

$$\begin{aligned}
 H_c(z) &= \frac{z^2}{z^2 - \alpha} = 1 + \frac{\alpha}{z^2 - \alpha} = 1 + \frac{\alpha}{(z - \sqrt{\alpha})(z + \sqrt{\alpha})} \stackrel{!}{=} 1 + \frac{b_1}{z - \sqrt{\alpha}} + \frac{b_2}{z + \sqrt{\alpha}} \\
 &= 1 + \frac{b_1(z + \sqrt{\alpha}) + b_2(z - \sqrt{\alpha})}{(z - \sqrt{\alpha})(z + \sqrt{\alpha})} \\
 \Rightarrow \quad \alpha &\stackrel{!}{=} b_1(z + \sqrt{\alpha}) + b_2(z - \sqrt{\alpha}) = (b_1 + b_2)z + (b_1 - b_2)\sqrt{\alpha} \\
 \Rightarrow \quad b_1 + b_2 &= 0 \text{ und } b_1 - b_2 = \sqrt{\alpha} \quad \Rightarrow \quad b_1 = \sqrt{\alpha}/2, \quad b_2 = -\sqrt{\alpha}/2 \\
 \Rightarrow \quad H_c(z) &= 1 + \frac{\sqrt{\alpha}}{2(z - \sqrt{\alpha})} - \frac{\sqrt{\alpha}}{2(z + \sqrt{\alpha})} \\
 &\bullet \rightsquigarrow \circ \\
 h_c[n] &= \delta[n] + \sqrt{\alpha}/2 \left((\sqrt{\alpha})^{n-1} - (-\sqrt{\alpha})^{n-1} \right) u[n] = \{1; 0; \alpha; 0; \alpha^2; \dots\} \text{ für } n = 0, 1, \dots \\
 h_c[2n] &= (\sqrt{\alpha})^{2n} u[2n]; \quad h_c[2n+1] = 0 \quad \text{für } n = 0, 1, \dots
 \end{aligned}$$

System d):

$$\begin{aligned}
 H_d(z) &= \frac{z^2}{z^2 + \alpha} = 1 - \frac{\alpha}{z^2 + \alpha} = 1 - \frac{\alpha}{(z - j\sqrt{\alpha})(z + j\sqrt{\alpha})} \stackrel{!}{=} 1 + \frac{b_1}{z - j\sqrt{\alpha}} + \frac{b_2}{z + j\sqrt{\alpha}} \\
 &= 1 + \frac{b_1(z + j\sqrt{\alpha}) + b_2(z - j\sqrt{\alpha})}{(z - j\sqrt{\alpha})(z + j\sqrt{\alpha})} \\
 \Rightarrow \quad -\alpha &\stackrel{!}{=} b_1(z + j\sqrt{\alpha}) + b_2(z - j\sqrt{\alpha}) = (b_1 + b_2)z + (b_1 - b_2)j\sqrt{\alpha} \\
 \Rightarrow \quad (b_1 + b_2)z &= 0 \quad \Rightarrow \quad b_1 = -b_2 \text{ und} \\
 (b_1 - b_2)j\sqrt{\alpha} &= -\alpha \Rightarrow b_1 - b_2 = j\sqrt{\alpha} \quad \Rightarrow \quad b_1 = j\sqrt{\alpha}/2, \quad b_2 = -j\sqrt{\alpha}/2 \\
 \Rightarrow \quad H_d(z) &= 1 + \frac{j\sqrt{\alpha}/2}{z - j\sqrt{\alpha}} - \frac{j\sqrt{\alpha}/2}{z + j\sqrt{\alpha}} \\
 &\bullet \rightsquigarrow \circ \\
 h_d[n] &= \delta[n] + j\sqrt{\alpha}/2 \left((j\sqrt{\alpha})^{n-1} - (-j\sqrt{\alpha})^{n-1} \right) u[n] = \{1; 0; -\alpha; 0; \alpha^2; \dots\} \\
 &\text{für } n = 0, 1, \dots \\
 h_d[2n] &= (\sqrt{\alpha})^{2n} u[2n]; \quad h_d[2n+1] = 0 \quad \text{für } n = 0, 1, \dots
 \end{aligned}$$

Anmerkung:

Die Übertragungsfunktion in Summenform, die man nach der Partialbruchzerlegung erhält, lässt sich in Hardware umsetzen; man nennt dies die *Parallelform*.

¹ # ... Ende der gem. import-Anweisungen

%

```

2 alpha = 0.9; f_S = 1
3 b = [1, 0] # z + 0
4 # b = [1 0 0] # z^2 + 0
5 a = [1, -alpha] # z - 0.9; Add., 1 Verzögerung
6 #a = [1 +alpha] # z + 0.9; Subtr., 1 Verz.
7 #a = [1 0 -alpha] # z^2 - 0.9; Add., 2 Verz.
8 #a = [1 0 +alpha] # z^2 - 0.9; Subtr., 2 Verz.
9 figure(1)
10 dsp.zplane(b,a) # Plotte P/N Diagramm
11 # H(f) entlang der oberen Hälfte des EK:
12 [W,H] = sig.freqz(b,a,1024, f_S)
13 figure(2)
14 plot(W/(2*pi),abs(H),linewidth = 2) # H(F)
15 xlabel(r'$F$ bzw. $\Omega / 2 \pi$')
16 ylabel(r'$|H(F)| \rightarrow$')
17 # Berechne 20 Werte der Impulsantwort:
18 [himp,t] = dsp.impz(b,a,20,f_S)
19 figure(3)
20 (ml, sl, bl) = stem(t,himp) # Impulsantwort
21 plt.setp(ml,'markerfacecolor','r',
22          'markersize',8)
23 plt.setp(sl,'linewidth',2)
24 xlabel('$n$'); ylabel(r'$h[n]$')
25 plt.show()

```

Lst. M2.5: Python Listing zu [M2.8](#)

```

alpha = 0.9; f_S = 1;
b = [1 0]; % z + 0
% b = [1 0 0]; % z^2 + 0
a = [1 -alpha]; % z - 0.9;
%a = [1 +alpha]; % z + 0.9;
%a = [1 0 -alpha]; % z^2 - 0.9;
%a = [1 0 +alpha]; % z^2 - 0.9;
figure(1);
zplane(b,a); % P/N Diagramm
%
[H,F]=freqz(b,a,1024, f_S);
figure(2);
plot(F,abs(H)); % Plotte H(f)
grid on;
xlabel('F bzw. \Omega / 2 \pi');
ylabel('|H(F)|');
% N automatisch:
%[himp,t]=impz(b,a,[],f_S);
[himp,t]=impz(b,a,20,f_S);
figure(3);
h1 = stem(t,himp);
set(h1,'MarkerSize',6,'LineWidth',1.5);
xlabel('n'); ylabel('h[n]');
grid on;

```

Lst. M2.6: Matlab Listing zu [M2.8](#)

M2.9. IIR-Filter zweiter Ordnung → Aufgabe 2.9

a) Differenzengleichung

Das Filter in Abb. 2.4(a) ist ein IIR-Filter zweiter Ordnung in Direktform 1 (Abb. M1.6(a)) mit $b_0 = -0,2$, $b_1 = 0,2$, $a_1 = 0$, $a_2 = -0,81$.

Die Differenzengleichung lautet: $y[n] = -0,2x[n] + 0,2x[n-1] - 0,81y[n-2]$.

b) Impulsantwort aus Differenzengleichung

Zur Bestimmung der Impulsantwort wird der zeitdiskrete Dirakstoß als Eingangsfolge verwendet, alle Register haben den Wert 0 bei $n \leq 0 \Rightarrow y[-1] = y[-2] = 0$:

$$x[n] = \delta[n] = \begin{cases} 1 & \text{für } n = 0 \\ 0 & \text{sonst} \end{cases}$$

$$\Rightarrow h[n] = y[n] = -0,2\delta[n] + 0,2\delta[n-1] - 0,81h[n-2] \Rightarrow$$

$$\begin{aligned} n=0 : \quad h[0] &= -0,2\delta[0] + 0,2\delta[-1] - 0,81h[-2] &= -0,2 \cdot 1 + 0 + 0 &= \mathbf{-0,2} \\ n=1 : \quad h[1] &= -0,2\delta[-1] + 0,2\delta[0] - 0,81h[-1] &= 0 + 0,2 \cdot 1 + 0 &= \mathbf{+0,2} \\ n=2 : \quad h[2] &= -0,2\delta[-2] + 0,2\delta[-1] - 0,81h[0] &= 0 + 0 - 0,81 \cdot (-0,2) &= \mathbf{+0,162} \\ n=3 : \quad h[3] &= \dots - 0,81h[1] &= 0 + 0 - 0,81 \cdot 0,2 &= \mathbf{-0,162} \\ n=4 : \quad h[4] &= \dots - 0,81h[2] &= 0 + 0 - 0,81 \cdot 0,162 &= \mathbf{-0,131} \\ n=5 : \quad h[5] &= \dots - 0,81h[3] &= 0 + 0 - 0,81 \cdot (-0,162) &= \mathbf{+0,131} \\ n=6 : \quad h[6] &= \dots - 0,81h[4] &= 0 + 0 - 0,81 \cdot (-0,131) &= \mathbf{+0,106} \\ &&&\vdots \end{aligned}$$

c) Ausgangssignal über diskrete Faltung

Das Ausgangssignal $y[n]$ eines LTI-Systems erhält man allgemein durch diskrete Faltung von Impulsantwort $h[n]$ und Eingangssignal $x[n]$:

$$y[n] = x[k] * h[k] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

Allerdings hat das System in diesem Fall eine unendlich ausgedehnte Impulsantwort (IIR-Filter), die außerdem nicht in geschlossener Form bestimmt wurde, so dass man auch $y[n]$ nicht in geschlossener Form bestimmen kann. Da das Eingangssignal $x[n]$ nur eine Länge von drei hat, kann $y[n]$ aus den gewichteten und verzögerten Impulsantworten bestimmt werden:

k	$x[k]$	$n =$	-1	0	1	2	3	4	5
-1	1	$x[-1]h[n+1]$	-0,2	0,2	0,162	-0,162	-0,131	0,131	0,106
0	2	$x[0]h[n]$		-0,4	0,4	0,324	-0,324	-0,262	0,262
1	-1	$x[1]h[n-1]$			0,2	-0,2	-0,162	0,162	0,131
$y[n] = \sum x[k]h[n-k] =$			-0,2	-0,2	0,762	-0,038	-0,617	0,031	0,489

d) Übertragungsfunktion

Aus der Differenzengleichung in a) oder direkt aus Abb. 2.4(a) (durch „genaues Hinschauen“) lässt sich leicht die Übertragungsfunktion konstruieren:

$$\begin{aligned} y[n] &= -0,2x[n] + 0,2x[n-1] - 0,81y[n-2] \\ \circlearrowleft \bullet Y(z) &= -0,2X(z) + 0,2X(z)z^{-1} - 0,81Y(z)z^{-2} \\ \Rightarrow H(z) &= \frac{Y(z)}{X(z)} = \frac{-0,2 + 0,2z^{-1}}{1 + 0,81z^{-2}} = \frac{-0,2z^2 + 0,2z^1}{z^2 + 0,81} = -0,2 \frac{z^2 - z^1}{z^2 + 0,81} \end{aligned}$$

Die letzte Form von $H(z)$ ist besonders gut geeignet, um die Pol- und Nullstellen zu ermitteln (positive Exponenten, Koeffizient 1 der höchsten Potenzen in Zähler und Nenner):

$$\text{Nullstellen: } z^2 - z = z(z-1) \stackrel{!}{=} 0 \Rightarrow z_{01} = 0; z_{02} = 1$$

$$\text{Polstellen: } z^2 + 0,81 \stackrel{!}{=} 0 \Rightarrow z^2 = -0,81 \Rightarrow z_{\infty 1,2} = \pm 0,9j$$

Es handelt sich um einen Bandpass (BP); das Maximum von $|H(e^{j\Omega})|$ im Basisband liegt in der Nähe von $\Omega = \pi/2$ ($f = f_s/4$), da dort die Polstelle dem Einheitskreis am nächsten kommt.

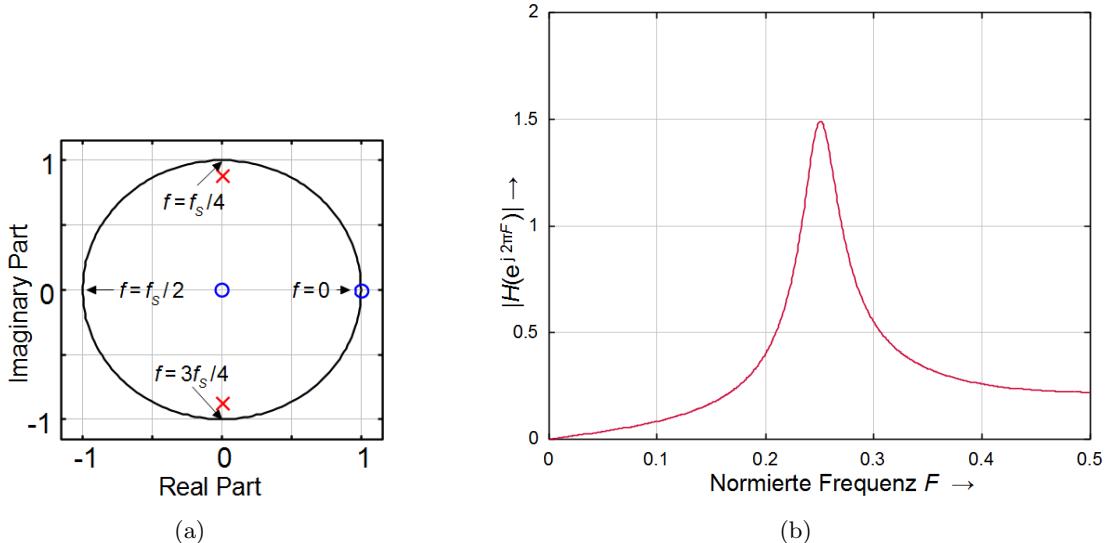


Abb. M2.19.: Pol-Nullstellen-Plan (a) und Amplitudenfrequenzgang (b) zu Abb. 2.4(a)

e) Betrag und Phase bei $f_1 = 25$ kHz

Eine Abtastperiode von $T_S = 10 \mu s$ entspricht einer Abtastfrequenz von $f_S = 1/T_S = 100$ kHz. Damit erhält man die normierte Frequenz:

$$\begin{aligned} F_1 &= f_1/f_S = 1/4; \Omega_1 = 2\pi F_1 = \pi/2 \Rightarrow z_1 = e^{j\Omega_1} = e^{j\pi/2} = j \\ \Rightarrow H(f)|_{f=25 \text{ kHz}} &= H(z)|_{z=j} = -0,2 \frac{j^2 - 0,2j}{j^2 + 0,81} = -0,2 \frac{-1 - j}{-1 + 0,81} = 1,0526 (-1 - j) \end{aligned}$$

$H(z = j)$ hat also einen negativen Real- und einen negativen Imaginärteil. Daher muss der Phasenwinkel im dritten Quadranten liegen, also $-90^\circ > \angle H(z = j) > -180^\circ$ sein. Diese Betrachtung ist wichtig, da

$$\arctan \frac{\Im\{H(z = j)\}}{\Re\{H(z = j)\}} = \arctan \frac{-1,0526}{-1,0526} = \arctan \frac{1,0526}{1,0526} = 45^\circ \hat{=} \pi/4$$

nicht dem Phasenwinkel entspricht - die Arcustangens-Funktion kann nicht zwischen erstem und dritten Quadranten unterscheiden (oder zwischen zweitem und viertem)! Daher gibt es in vielen Programmiersprachen die **atan2**-Funktion, die für den dritten und vierten Quadranten den Arcustangens-Wert um π (bzw. 180°) korrigiert:

$$\angle H(z = j) = \text{atan}2 \frac{\Im\{H(z = j)\}}{\Re\{H(z = j)\}} = \arctan \frac{\Im\{H(z = j)\}}{\Re\{H(z = j)\}} + 180^\circ = 45^\circ + 180^\circ = 225^\circ \hat{=} \frac{5\pi}{4}$$

Eine Korrektur um $-\pi$ bzw. -180° wäre ebenfalls richtig und würde das äquivalente Ergebnis liefern $\angle H(z = j) = -135^\circ \hat{=} -3\pi/4$.

Für die Betragsberechnung ist diese Fallunterscheidung natürlich nicht nötig:

$$|H(z = j)| = \sqrt{\Im^2\{H(z = j)\} + \Re^2\{H(z = j)\}} = 1,0526\sqrt{2} = \mathbf{1,489}$$

f) Übertragungsfunktion in Produktform

Pole und Nullstellen wurden ja bereits in Punkt d) bestimmt, damit kann man $H(z)$ in Produktform sofort aufstellen:

$$H(z) = -0,2 \frac{z(z - 1)}{(z - 0,9j)(z + 0,9j)}$$

Wie im vorigen Unterpunkt sind $f_1 = 25$ kHz und $f_S = 100$ kHz, also gilt wieder $F_1 = 1/4$ und $z_1 = j$:

$$H(z = j) = -0,2 \frac{j(j - 1)}{(j - 0,9j)(j + 0,9j)}$$

Der Gesamtbetrag ergibt sich aus dem Produkt der Teilbeträge, die hier besonders leicht zu bestimmen sind:

$$|H(z = j)| = |-0,2| \frac{|j||j - 1|}{|j - 0,9j||j + 0,9j|} = 0,2 \frac{1 \cdot \sqrt{2}}{0,1 \cdot 1,9} = 1,0526\sqrt{2} = \mathbf{1,489}$$

Die Gesamtphase erhält man durch Aufsummieren der Teilphasen, Beiträge im Nenner müssen subtrahiert werden:

$$\begin{aligned} \angle H(z = j) &= \angle(-0,2) + \underbrace{\angle(j) + \angle(j - 1)}_{\text{Zähler}} - \underbrace{(\angle(j - 0,9j) + \angle(j + 0,9j))}_{\text{Nenner}} \\ &= 180^\circ + 90^\circ + 135^\circ - (90^\circ + 90^\circ) = 225^\circ \hat{=} 5\pi/4 \end{aligned}$$

Anmerkung: Die Berechnungen in diesem Unterpunkt entsprechen der grafischen Abschätzung von Amplituden- und Phasengang aus der Lage von Polen und Nullstellen im P/N-Diagramm. Auch hier werden die Entfernungen (= Beträge) und Winkel der Pole und Nullstellen zum Aufpunkt auf dem Einheitskreis zusammengefasst.

g) Änderung der Filterkoeffizienten

Die Koeffizienten des Zählers dürfen mit einem (einheitlichen!) konstanten Faktor skaliert werden, ohne dass sich die Lage der Nullstellen ändert. Im Nenner wäre dies rechnerisch genauso möglich, ohne dass sich die Lage der Polstellen verschiebt. Allerdings wird dann $a_0 \neq 1$ und die zugehörige Differenzengleichung lässt sich nicht mehr direkt in eine Hardwarestruktur umwandeln. Versuchen Sie es selbst einmal.

Daher kann nur der Zähler, nicht aber der Nenner skaliert werden!

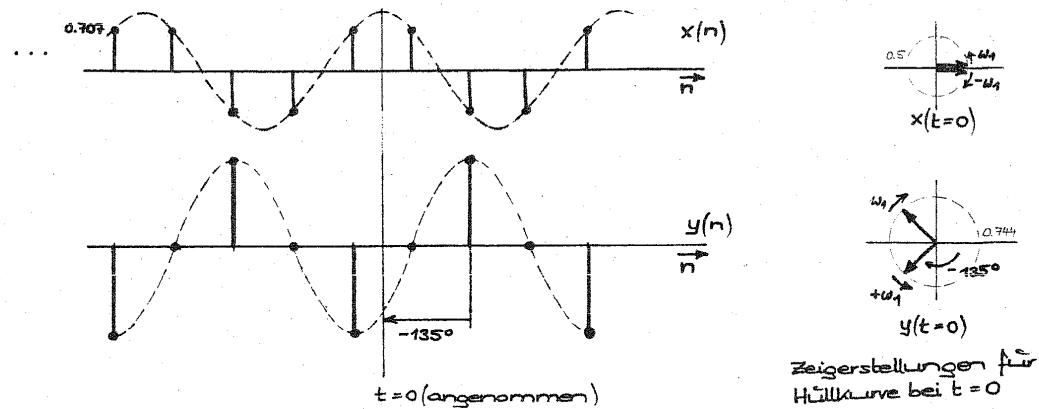
Es wurde bereits berechnet, dass $|H(f)|_{f=25 \text{ kHz}} = |H(z=j)| = 1,489$ ist.

Soll $|H(z=j)| = 1$ gelten, müssen die Zählerkoeffizienten durch 1,489 dividiert werden:

$$b'_0 = b_0 / 1,489 = -0,2 / 1,489 = -0,1344 \quad \text{und} \quad b'_1 = b_1 / 1,489 = 0,2 / 1,489 = 0,1344.$$

h) bei 25 kHz im eingeschwungenen Zustand: Amplitudenverstärkung 1,489

Phase -135° / bei 4 Abtastwerten
pro Periode $\approx 1\frac{1}{2}$ Abtastperioden
Nachteilung)



Kontrolle mittels Differenzengleichung:

$$\begin{aligned} y(n) &= 0 = -0.2 \cdot x(n) + 0.2 \cdot x(n-1) - 0.81 \cdot y(n-2) \\ &\quad \text{gleichgroße Zahlenwerte} \quad 0 \\ y(n) &= 1.489 = -0.2 \cdot x(n) + 0.2 \cdot x(n-1) - 0.81 \cdot y(n-2) = -0.2 \cdot (-\frac{\sqrt{2}}{2}) + 0.2 \cdot \frac{\sqrt{2}}{2} - 0.81 \cdot (-1.489) \end{aligned}$$

M2.10. Moving Average Filter / rect-Fenster → Aufgabe 2.10

Die Stoßantwort und Übertragungsfunktion von Moving Average Filtern der Ordnung $N-1$ (d.h. mit $N-1$ Verzögerungsgliedern) lauten:

$$h[n] = \sum_{m=0}^{N-1} \delta[n-m] \Rightarrow H(z) = \sum_{m=0}^{N-1} z^{-m} = \frac{1-z^{-N}}{1-z^{-1}}$$

Die Summe wurde mit Hilfe der Formel für die endliche geometrische Reihe umgeformt. Die Frequenzantwort wird auf dem Einheitskreis ermittelt durch Setzen von $z = e^{j\omega T_S} = e^{j\Omega}$. Auch wird wieder die Formel für die endliche geometrische Reihe angewendet. Im nächsten Schritt werden aus Zähler und Nenner Exponentialterme so ausgeklammert, dass ein symmetrisches Polynom („Spiegelpolynom“) entsteht, das mit den Eulerschen Identitäten in Sinus bzw. Cosinus-Funktionen umgewandelt werden kann.

$$\begin{aligned} H(e^{j\Omega}) &= \sum_{n=0}^{N-1} e^{-jn\Omega} = \frac{1 - e^{-jN\Omega}}{1 - e^{-j\Omega}} = \frac{e^{-jN\Omega/2}}{e^{-j\Omega/2}} \cdot \frac{e^{+jN\Omega/2} - e^{-jN\Omega/2}}{e^{+j\Omega/2} - e^{-j\Omega/2}} \\ &= e^{-j(N-1)\Omega/2} \frac{\sin(N\Omega/2)}{\sin(\Omega/2)} = \underbrace{e^{-j(N-1)\Omega/2}}_{\text{lineare Phase}} \cdot \underbrace{N}_{H(\Omega=0)} \cdot \underbrace{D_N(\Omega)}_{\text{Amplitudengang}} \end{aligned}$$

Der ausgeklammerte Exponentialterm hat einen Betrag von 1 und beschreibt eine lineare Phase von $(N - 1)\Omega/2$. Damit lautet der Betrag der Frequenzantwort:

$$|H(f)| = \left| \frac{\sin(N\Omega/2)}{\sin(\Omega/2)} \right| = |ND_N(\Omega)| \text{ mit } H(0) = N$$

Für den Einsatz als Moving Average Filter muss also der Ausgangswert noch mit $1/N$ skaliert werden.

Die erste Nullstelle f_0 der Frequenzantwort liegt bei $N\omega T_S/2 = \pi \Rightarrow f_{01} = f_S/N$ bzw. $F_0 = 1/N$ bzw. $\Omega = 2\pi/N$.

Das Maximum des ersten Nebenzipfels liegt in der Mitte³ zwischen erster und zweiter Nullstelle bei $f_{NZ} = 3f_S/2N$.

$$\frac{|H(f_{NZ})|}{H(0)} \approx \frac{1}{N} \left| \frac{\sin(3\pi/2)}{\sin(3\pi/2N)} \right| \approx \frac{1}{N} \frac{1}{3\pi/2N} = \frac{2}{3\pi} \text{ mit } \sin x \approx x \text{ für } x \ll 1$$

Damit ist die Höhe des ersten Nebenzipfels nahezu unabhängig von der Länge des MA - Filters.

	Allgemein	$N = 32$	$N = 1024$
H_0	N	32	1024
f_{01}	f_S/N	320 kHz	10 kHz
$H(f_{NZ})$	$2/(3\pi)$	0,21 $\hat{=} -13,5$ dB	0,21 $\hat{=} -13,5$ dB

Tab. M2.2.: Daten zu Aufgabe 2.10

Anmerkungen:

- Eine Verbesserung der Nebenzipfelunterdrückung kann einfach erreicht werden, indem man mehrere MA-Filter kaskadiert bzw. die Daten das gleiche Filter mehrfach durchlaufen lässt.
- Eine direkte Implementierung eines MA-Filters mit $N - 1 = 1023$ Addierern wäre sehr ineffizient, in der Praxis werden MA-Filter daher fast immer als Cascaded Integrator-Comb (CIC) Filter realisiert (Kap. 5, 7 und 8)
- Die Ergebnisse beschreiben auch das Verhalten eines Rechteckfensters mit $N = 32$ bzw. 1024 Punkten!

M2.11. Notchfilter → Aufgabe 2.11

- a) Am Einfachsten löscht man den Störton aus, indem man eine Nullstelle genau bei der Frequenz $F_1 = f_1/f_S = 0,3$ auf dem Einheitskreis platziert, $z_{0,1} = e^{j2\pi F_1}$. Um ein reellwertiges Filter zu erhalten, muss eine zweite konjugiert-komplexe Nullstelle $z_{0,2} = z_{0,1}^* = e^{-j2\pi F_1}$ hinzugefügt werden. Daher hat das Filter mindestens die Ordnung zwei.

Die Übertragungsfunktion des einfachen Notchfilters lautet

$$H_1(z) = z^{-2}(z - e^{j2\pi F_1})(z - e^{-j2\pi F_1}) = z^{-2}(z^2 - 2z \Re\{e^{j2\pi F_1}\} + 1) = 1 - 2z^{-1} \cos 2\pi F_1 + z^{-2},$$

³Gilt in guter Näherung, da sich die Sinus-Funktion im Nenner zwischen zwei Zählernullstellen nur wenig ändert.

Frequenz- und Betragsgang sind:

$$\begin{aligned} H_1(e^{j\Omega}) &= 1 - 2e^{-j\Omega} \cos 2\pi F_1 + e^{-2j\Omega} = e^{-j\Omega} (e^{j\Omega} - 2 \cos 2\pi F_1 + e^{-j\Omega}) \\ &= 2e^{-j\Omega} (\cos \Omega - \cos 2\pi F_1) \\ \Rightarrow |H_1(e^{j\Omega})| &= 2 |\cos 2\pi F - \cos 2\pi F_1| = 2 |\cos 2\pi F + 0,309| \end{aligned}$$

Man sieht, dass zwar bei $F = F_1$ die gewünschte Auslöschung stattfindet, dass aber der Frequenzgang unsymmetrisch zu F_1 ist.

- b) Die doppelten Nullstellen bei F_1 führen zu der Übertragungsfunktion $H_2(z) = H_1^2(z)$:

$$\begin{aligned} H_2(z) &= z^{-4} (z - e^{j2\pi F_1})^2 (z - e^{-j2\pi F_1})^2 = H_1^2(z) \\ \Rightarrow |H_2(e^{j\Omega})| &= |H_1(e^{j\Omega})|^2 = 4 (\cos 2\pi F - \cos 2\pi F_1)^2 \end{aligned}$$

Der Betragsgang von H_2 weist daher eine stärkere Dämpfung in der Nähe der Notchfrequenz, aber auch eine noch stärkere Asymmetrie auf als der von H_1 .

- c) Die Polstellen sollen den gleichen Winkel aufweisen wie die Nullstellen, $z_{\infty,1} = r_1 e^{j2\pi F_1}$ und $z_{\infty,2} = r_1 e^{-j2\pi F_1}$. Die Übertragungsfunktion lautet damit:

$$\begin{aligned} H_3(z) &= \frac{(z - e^{j2\pi F_1})(z - e^{-j2\pi F_1})}{(z - r_1 e^{j2\pi F_1})(z - r_1 e^{-j2\pi F_1})} = \frac{z^2 - 2z \Re\{e^{j2\pi F_1}\} + 1}{z^2 - 2r_1 z \Re\{e^{j2\pi F_1}\} + r_1^2} \\ &= \frac{z^2 - 2z \cos 2\pi F_1 + 1}{z^2 - 2r_1 z \cos 2\pi F_1 + r_1^2} \end{aligned}$$

Frequenz- und Betragsgang sind:

$$\begin{aligned} H_3(e^{j\Omega}) &= \frac{e^{j\Omega} (e^{j\Omega} - 2 \cos \Omega_1 + e^{-j\Omega})}{e^{j\Omega} (e^{j\Omega} - 2r_1 \cos \Omega_1 + r_1^2 e^{-j\Omega})} = \frac{e^{j\Omega} - 2 \cos \Omega_1 + e^{-j\Omega}}{e^{j\Omega} - 2r_1 \cos \Omega_1 + r_1^2 e^{-j\Omega}} \\ &= \frac{2 (\cos \Omega - \cos \Omega_1)}{\cos \Omega + j \sin \Omega - 2r_1 \cos \Omega_1 + r_1^2 (\cos \Omega - j \sin \Omega)} \\ \Rightarrow |H_3(e^{j\Omega})| &= \frac{2 |\cos \Omega - \cos \Omega_1|}{\sqrt{((1 + r_1^2) \cos \Omega - 2r_1 \cos \Omega_1)^2 + ((1 - r_1^2) \sin \Omega)^2}} \end{aligned}$$

Wer es schafft, $|H_3(e^{j\Omega})|$ übersichtlicher aufzulösen, wird in der nächsten Ausgabe natürlich erwähnt ...

	$H(F = 0)$	$H(F = 0,29)$	$H(F = 0,31)$	$H(F = 0,5)$
$ H_1(F) $	2,618 \equiv 8,36 dB	0,121 \equiv -18,37 dB	0,118 \equiv -18,55 dB	1,382 \equiv 2,81 dB
$ H_2(F) $	6,854 \equiv 16,72 dB	0,015 \equiv -36,74 dB	0,014 \equiv -37,09 dB	1,910 \equiv 5,62 dB
$ H_3(F) $	1,052 \equiv 0,44 dB	0,815 \equiv -1,78 dB	0,815 \equiv -1,78 dB	1,051 \equiv 0,43 dB

Tab. M2.3.: Ergebnisse zu Aufgabe M2.11

```

2 # Definiere Nullstellen auf EK:
3 N = np.asarray([np.exp(1j*0.3*2*pi),
4                 np.exp(-1j*0.3*2*pi)])
5 # Pole: gleicher Winkel, kleinerer Radius:
6 P = 0.95 * N
7 # "Ausmultiplizieren" von P/N ergibt Koeff.
8 b = np.poly(N); a = np.poly(P)
9 figure(1)
10 dsp.zplane(N,P,zpk = True)
11 figure(2)
12 subplot(211)
13 # Frequenzgang an 2048 Punkten:
14 [W,H] = sig.freqz(b,a,2048)
15 F = W / (2* pi)
16 plot(F, 20*log10(abs(H))); grid(True)
17 subplot(212)
18 plot(F,angle(H))
19 xlabel('F ->'); grid(True)
20 plt.tight_layout()
21 # Testfreq. (normierte Kreisfreq.):
22 W_test = array([0, 0.29, 0.3, 0.31, 0.5])*2*pi
23 # Frequenzgang bei Testfrequenzen:
24 [H_test, W_test]=sig.freqz(b,a,W_test)
25 print('H_test = ', H_test)
26 print('|H_test| = ', abs(H_test))
27 print(20*log10(abs(H_test)))
28 plt.show()

```

Lst. M2.7: Python Listing zu [M2.11](#)

```

% Definiere Nullstellen auf EK:
N = [exp(j*0.3*2*pi); ...
      exp(-j*0.3*2*pi)];
% Pole: gleicher Winkel
P = 0.95 * N;
% "Ausmultiplizieren" von P/N -> Koeff.
b = poly(N); a = poly(P);
figure(1);
zplane(N,P);
figure(2);
subplot(211);
% Frequenzgang an 2048 Punkten:
[H,W] = freqz(b,a,2048);
F = W / (2* pi);
plot(F, 20*log10(abs(H))); grid on;
subplot(212);
plot(F,angle(H));
xlabel('F ->'); grid on;
% Testfreq. (normierte Kreisfreq.):
W_test = [0 0.29 0.3 0.31 0.5]*2*pi;
%
% Frequenzgang bei Testfrequenzen:
[H_test, W_test]=freqz(b,a,W_test);
H_test
abs(H_test)
20*log10(abs(H_test))
%
```

Lst. M2.8: Matlab Listing zu [M2.11](#)

M3. DFT: Diskrete Fouriertransformation und FFT

M3.1. * Allgemeine Fragen zu DFT und FFT → Aufgabe 3.1

- a) **Fourierreihe:** Ein periodisches Signal kann durch eine unendliche Summe von Sinus- und Cosinusschwingungen dargestellt werden, deren Frequenzen ganzzahlige Vielfache der Grundschwingung sind. Die Amplituden der Sinus- und Cosinusschwingungen nennt man *Fourierkoeffizienten*, das Spektrum ist ein Linienspektrum.

Fouriertransformation: Ein unendlich ausgedehntes Signal mit endlicher Energie (z.B. ein pulsförmiges Signal das bis auf wenige Ausnahmen den Wert Null hat) kann durch ein unendlich ausgedehntes Integral von Sinus- und Cosinusschwingungen dargestellt werden. Die Gewichtungsfunktion für die Sinus- und Cosinusschwingungen nennt man *Fouriertransformierte*, es ist im Allgemeinen eine kontinuierliche Funktion. Hat das Eingangssignal unendliche Energie (z.B. ein Zufallsprozess), ist die Fouriertransformierte nicht definiert. Eine Aussage über die spektralen Eigenschaften dieses Prozesses lässt sich nur treffen, indem man die Fouriertransformierte der Autokorrelationsfunktion berechnet. Das Ergebnis ist dann die *Leistungsdichtefunktion* mit der Einheit W/Hz.

DTFT: Discrete-Time Fourier Transform; wird vergleichbar mit der konventionellen Fourieranalyse und -synthese auf unendlich ausgedehnte zeitdiskrete Folgen angewendet. Aufgrund der zeitdiskreten Natur des Eingangssignals ist das Spektrum periodisch und im Allgemeinen kontinuierlich (Ausnahme: Das Eingangssignal ist periodisch). Da nur unendlich ausgedehnte Folgen analysiert werden können, ist die DTFT ungeeignet für computerbasierte Analysen.

DFT: Discrete Fourier Transform; aus einer unendlich ausgedehnten Folge wird ein Stück herausgeschnitten (Fensterung), für die DFT tut man dann so, als ob sich das herausgeschnittene Stück unendlich wiederholt. Aufgrund dieser „Periodizität“ ist das resultierende Spektrum ein Linienspektrum; im allgemeinen Fall entsprechen N Werte im Zeitbereich N Werten im Frequenzbereich. Die DFT ist daher eine Blocktransformation, die sehr gut für computerbasierte Berechnungen geeignet ist.

FFT: Fast Fourier Transform. Unter FFT versteht man verschiedene effiziente Algorithmen zur schnelleren Berechnung der DFT. Bei idealer Arithmetik liefern DFT und FFT identische Ergebnisse, allerdings sind für eine N -Punkt DFT N^2 komplexe Multiplikationen notwendig. Das entspricht $4N^2$ reellen Multiplikationen bei komplexwertigen und $2N^2$ reellen Multiplikationen bei reellwertigen Eingangssignalen. Für eine Radix-2 FFT mit N Punkten sind nur ca. $2N \log_2 N$ reelle Multiplikationen erforderlich (die genaue Zahl hängt vom verwendeten Algorithmus ab).

- b) **Frequenzen f_k der Spektrallinien:** Eine DFT mit N Punkten liefert N Frequenzpunkte im Frequenzbereich $[0 \dots f_S]$. Der Abstand der Spektrallinien ist $\Delta f = f_S/N$, die Linien treten auf bei $f_k = k f_S/N$ mit $k = 0, 1, \dots, N-1$ und die höchste Frequenz liegt dementsprechend bei $f_S(N-1)/N$. Bei reellwertigen Eingangssignalen sind die oberen

$N/2$ Frequenzpunkte redundant, da Amplituden- und Phasenspektrum symmetrisch zu $f = 0$ und $f = f_S/2$ sind.

- c) **Auswertung in der positiven Hälfte des Nyquistbereichs** Bei reellwertigen Signalen sind Real- und Imaginärteil bzw. Betrag und Phase der DFT symmetrisch zu $f = 0$, daher genügt es eine Hälfte der Koeffizienten zu kennen. Hierbei haben Realteil und Betrag eine gerade Symmetrie und Imaginärteil und Phase eine ungerade Symmetrie. Bei komplexwertigen Eingangssignalen müssen beide Hälften ausgewertet werden.
- d) Eine **Fensterung vor Berechnung einer DFT / FFT** ist notwendig, da die DFT im Gegensatz zur DTFT eine Blocktransformation ist, die mit einer endlichen Anzahl Samples arbeitet (s.o.). Wenn von vorneherein nur eine endliche Anzahl Samples zur Verfügung steht, kann man sich vorstellen, dass die restlichen (unendlich vielen) Samples alle Null waren (DTFT) oder sich periodisch wiederholen und mit einem Rechteckfenster abgetrennt wurden.

M3.2. * DFT des MA-Filters → Aufgabe 3.2

Sie haben vermutlich $|H(F = 0)| = 4$ und $|H(F = 0,5)| = 0$ erwartet, vielleicht noch eine weitere Nullstelle bei $F = 0,25$. Das liefert die Simulation auch, aber der Verlauf entspricht nicht der erwarteten si-Funktion.

```

1 # ... Ende der Import-Anweisungen
2 h = [1,1,1,1]
3 n = arange(len(h))
4 figure(1);
5 subplot(211)
6 stem(n,h)
7 ylabel(r"$h[n] \rightarrow$")
8 xlabel(r"$n \rightarrow$")
9 subplot(212)
10 H = abs(fft(h,256))
11 k = arange(len(H)/2)
12 stem(k,H[0:len(k)])
13 xlabel(r"$k \rightarrow$")
14 ylabel(r"$|H[k]| \rightarrow$")
15 plt.tight_layout()
16 plt.show()

```

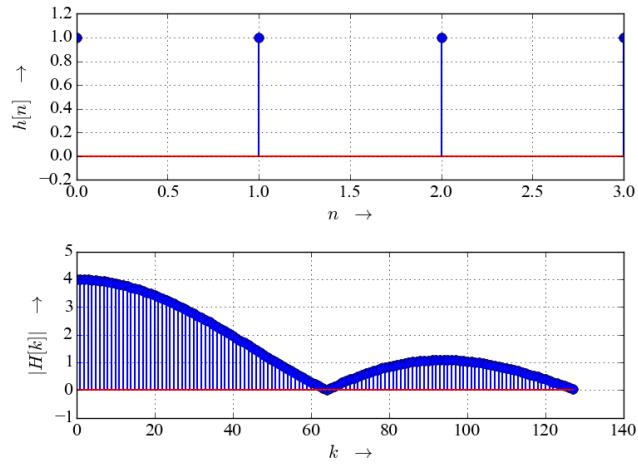


Abb. M3.1.: Korrigiertes Python Skript und Plot zur Simulation des Frequenzgangs zu Aufgabe 3.2

Es gibt mehrere Fallen bei der Anwendung der DFT/FFT auf „echte“ Signale, in die am Anfang fast jeder stolpert:

Die DFT/FFT ...

... lässt sich direkt nur auf periodische Signale anwenden.

Die DFT ist eine Blocktransformation für N Samples zu Zeitpunkten $t_n = nT_S$ mit $n = 0 \dots N - 1$, die N Frequenzpunkte $f_k = kf_1 = kf_S/N$ mit $k = 0 \dots N - 1$ liefert. Ein zeitdiskretes Signal hat ein periodisches Spektrum (Wiederholspektrum), zu einem diskreten (Linien-)Spektrum gehört ein periodisches Zeitsignal! Ein zeitlich begrenztes Signal mit unendlich vielen angehängten Nullen ist ein Fall für die DTFT.

... gibt das gesamte Basisband zurück.

Bei reellwertigen Zeitsignalen ist die FFT im Bereich $f_S/2 \dots f_S$ gespiegelt zu $0 \dots f_S/2$ und liefert keine neuen Erkenntnisse. Meist plottet man daher nur die ersten $N/2$ Frequenzpunkte.

... liefert im Allgemeinen komplexwertige Ergebnisse.

Man muss daher vor dem Plotten Betrag / Phase oder Real- und Imaginärteil der Fouriertransformierten berechnen. Vergisst man das, plottet Python nur den Realteil, die Warnung „Casting complex values to real discards the imaginary part“ in der Konsole kann man leicht übersehen.

... muss für periodische und Gleichsignale skaliert werden, um physikalisch korrekte Werte zu liefern.

Die *Energie* eines Signals bzw. Signalausschnitts bleibt bei der DFT erhalten. Soll mit Hilfe der DFT näherungsweise die CFT bzw. DTFT eines *aperiodischen* Signals (= Energie signal, da die *Energie* definiert ist) bestimmt werden (wie in dieser Aufgabe), darf *nicht* skaliert werden.

Bei einem periodischen oder einem Gleichsignal interessiert die *Leistung* (die zeitliche Ausdehnung und daher die Energie sind ja unendlich groß), die man erhält indem man die Energie pro Zeitabschnitt bestimmt.

Die Fouriertransformierte *periodischer* Signale muss daher bei Matlab und Python durch die Anzahl N der Samples (= die zeitliche Ausdehnung) dividiert werden, damit Amplituden / Leistungen im Zeit- und Frequenzbereich übereinstimmen.¹ Wurde das Signal durch Zeropadding auf die Gesamtlänge L verlängert, darf trotzdem nur mit der ursprünglichen Anzahl N skaliert werden - andernfalls würde die Amplitude der Fouriertransformierten des Signals ja von der Anzahl der angehängten Nullen abhängen.

Der Plot, der von Listing 3.4 erzeugt wird, ist nicht falsch; man kann ihn interpretieren als DFT

- eines Gleichsignals, die noch nicht mit der Anzahl der Datenpunkte skaliert wurde oder
- der Impulsantwort mit vier Abtastpunkten. Die vier Frequenzpunkte sind korrekt, allerdings wäre eine Interpolation zwischen den Punkten praktisch.

In Abb. M3.1 sind die o.g. Korrekturen integriert: Macht man die Anzahl der FFT-Punkte größer als die Anzahl der Datenpunkte (Zeile 10), ergänzen Python und Matlab die fehlenden Punkte automatisch durch Nullen (*Zeropadding*). Mit der `abs()` Funktion wird der Betrag aus Real- und Imaginärteil berechnet, in Zeile 11 wird der Vektor \mathbf{k} über die Hälfte der Frequenzpunkte gebildet, in Zeile 12 wird die erste Hälfte von $|H[k]|$ über diesen Bereich geplottet.

M3.3. * Frequenzauflösung der DFT → Aufgabe 3.3

a) Frequenzauflösung der DFT

Die Frequenzauflösung Δf ist die Breite eines Frequenzintervalls der DFT; sie ist der Kehrwert der Länge des Zeitfensters T_0 (Abtastperiode x Anzahl der Messpunkte), $\Delta f = 1/T_0 = 1/(NT_S) = f_S/N$.

¹Vorsicht: Bei anderen Softwarebibliotheken werden u.U. andere Skalierungen verwendet.

Diese maximale Frequenzauflösung wird nur erreicht, wenn das Signal mit einem Rechteckfenster „ausstanzt“. Andere Fenster, die das Zeitsignal an den Rändern sanfter ausblenden, haben geringeren Amplitudenfehler und Leckeffekt, dafür aber eine höhere -3dB Bandbreite $B = m\Delta f$ mit $m = 0,9$ (Rechteckfenster) . . . $m = 3,7$ (Flattop-Fenster).

b) Abtastfrequenz

Eine Frequenzauflösung von $\Delta f = 1$ Hz entspricht einem Zeitfenster $T_0 = 1/\Delta f = 1$ s. Mit $T_0 = N/T_S = N/f_S$ folgt $f_S = N/T_0 = 2048$ Hz. Die höchste Frequenz, die von der DFT dargestellt wird, ist $f_S/2 = 1024$ Hz. Das ist auch die höchste relevante Frequenzkomponente eines mit f_S abgetasteten reellwertigen Signals, oberhalb von $f_S/2$ wiederholt sich das Spektrum.

c) DFT eines Seismogramms

Die maximal erfassbare Frequenz ist $f_S/2$, die bestmögliche Frequenzauflösung erhält man mit maximaler Anzahl von Messpunkten $N_{max} = 2^{15}$, das resultierende Messfenster hat die Länge $T_{mess} = N_{max}/f_S$.

Relevante Daten wurden über einen Zeitraum von ca. $T_{Data} = 80$ min = 4800 s aufgezeichnet, diesen Zeitraum muss also das Messfenster der DFT erfassen.

Bei $N_{max} = 2^{15} = 32\,768$ und $f_S = 25,6$ Hz ist das Messfenster nur $T_0 = 32\,768/25,6$ Hz = 1280 s lang, erfasst also nur einen Bruchteil der Daten. Um das gesamte Messfenster zu erfassen, darf man höchstens mit

$$f_{S,max} = N_{max}/T_{Data} = 32\,768/4800 \text{ s} = 6,8 \text{ Hz}$$

abtasten. Die nächstkleinere einstellbare Abtastfrequenz ist dann **$f_S = 6,4$ Hz**. Die Frequenzauflösung beträgt dann $\Delta f = f_S/N = 0,2$ mHz, die maximal darstellbare Frequenz ist $f_S/2 = 3,2$ Hz.

Da das gefilterte Seismogramm oberhalb von 1 Hz keine Frequenzkomponenten enthält, wäre eine effizientere Kombination $f_{S2} = 3,2$ Hz, die maximal darstellbare Frequenz $f_{S2}/2 = 1,6$ Hz und $N_{min} \geq T_{Data}f_{S2} = 15360$. Die nächstmögliche Anzahl von Abtastwerten wäre mit $N = 2^{14} = 16384$ nur halb so groß wie vorher.

d) Störfrequenz

Der Abstand der Frequenzpunkte beträgt $\Delta f = f_S/N = 0,78$ mHz, der Frequenzindex $k = 620$ entspricht daher einer Signalfrequenz von $f_k = k\Delta f = kf_S/N = 0,48$ Hz. Solche Störfrequenzen werden von großen Maschinen wie z.B. von Sägewerken erzeugt.

M3.4. * Rechenaufwand für DFT und FFT → Aufgabe 3.4

a) DFT-Länge der 9. Symphonie

Die erforderliche Länge der DFT ist identisch mit der Anzahl der Datenpunkte je Kanal, $N_{DFT} = T_0 f_S = 4440 \text{ s} \cdot 44100 \text{ Hz} = 195\,804\,000$. Die Frequenzauflösung hängt nur mit der Länge des Datenfensters zusammen, $\Delta f = 1/T_0 = 1/4440 \text{ s} = 225 \mu\text{Hz}$. Aus den Koeffizienten der DFT ließe sich theoretisch mit der inversen DFT das Zeitsignal perfekt rekonstruieren, da im Messfenster alle Datenpunkte enthalten sind.

b) Rechenaufwand DFT / FFT

Für jeden Kanal muss eine DFT mit N_{DFT} Punkten berechnet werden, insgesamt sind daher mit (3.1) $N_{MUL,DFT} = 2N_{DFT}^2 = 7,67 \cdot 10^{16}$ Rechenoperationen, für eine FFT sind mit (3.2) immer noch $N_{MUL,FFT} = 2N_{FFT} \log_2 N_{FFT} = 10,8 \cdot 10^9$ reelle Multiplikationen erforderlich.

c) Rechenzeit auf FPGA

Das FPGA hat eine Gesamtrechenleistung von $R_{MUL} = 512 \cdot 500 \text{ MMAC/s} = 256 \text{ GMAC/s}$. Damit ließen sich DFT bzw. FFT berechnen in

$$T_{DFT} = N_{MUL,DFT}/R_{MUL} = 6,0 \cdot 10^5 \text{ s} = 6,9 \text{ d}$$

$$T_{FFT} = N_{MUL,FFT}/R_{MUL} = 0,042 \text{ s.}$$

In der Realität würde die meiste Zeit für das Speichern und Laden der großen Datenmengen benötigt, hier sollte nur gezeigt werden, welche Rechenzeiteinsparung mit der FFT möglich ist.

d) Rechenaufwand für Real-Time FFT

Für Real-Time Signalverarbeitung müssen im zeitlichen Mittel $2f_S$ (Stereo) Samples pro Sekunde verarbeitet werden. Ein Fenster mit N_{FFT} Samples hat eine zeitliche Länge $T_{Mess} = N_{FFT}/f_S$ und benötigt für die FFT $N_{MAC} = N_{FFT} \log_2 N_{FFT}$ reellwertige Rechenoperationen (Multiply-Accumulate, MAC). Damit werden im Mittel

$$N_{MAC}/\text{s} = \frac{N_{MAC}}{T_{Mess}} = f_S \log_2 N_{FFT}$$

Rechenoperationen pro Sekunde benötigt.

	N_{MAC}	T_{mess}	MAC/s
N_{FFT}	$N_{FFT} \log_2 N_{FFT}$	N_{FFT}/f_S	$f_S \log_2 N_{FFT}$
512	4608	11,6 ms	$397 \cdot 10^3$
8192	106496	185,8 ms	$573 \cdot 10^3$

Tab. M3.1.: Anzahl der Rechenoperationen für eine FFT in Abhängig von der Anzahl der Datenpunkte ($f_S = 44,1 \text{ kHz}$, Aufgabe 3.4)

Angeblich wurde die Laufzeit von CDs und der dafür erforderliche Durchmesser so festgelegt, dass die gesamte 9. Symphonie gerade noch auf eine CD passt. Vermutlich ist das aber nur eine gut erfundene Geschichte:

<http://gizmodo.com/5729864/why-the-cd-is-74-minutes-long>.

M3.5. # Fourierreihe und synchrone DFT → Aufgabe 3.5

CT Signal: $s(t) = 1 \text{ V} (1 + \cos(2\pi f_1 t + \pi/4))$ mit $f_1 = 1 \text{ kHz}$

DT Signal: $s[n] = 1 \text{ V} (1 + \cos(2\pi f_1 n T_S + \pi/4)) = 1 \text{ V} (1 + \cos(2\pi n F_1 + \pi/4))$

- a) **Betragsspektrum und Phasenspektrum des zeitkontinuierlichen (CT) Signals**
erhält man im Allgemeinen über die kontinuierliche Fouriertransformation (B.1), bei periodischen Signalen (wie hier) über die Fourierreihe (B.3):

$$\begin{aligned}
c_k &= \frac{1}{T_1} \int_0^{T_1} s(t) e^{-j2k\pi f_1 t} dt = \frac{1}{T_1} \int_0^{T_1} (1 + \cos(2\pi f_1 t + \pi/4)) e^{-j2k\pi f_1 t} dt \\
&= \frac{1}{T_1} \left(\int_0^{T_1} e^{-j2k\pi f_1 t} + \frac{1}{2} \left(e^{j(2\pi f_1 t + \pi/4)} + e^{-j(2\pi f_1 t + \pi/4)} \right) e^{-j2k\pi f_1 t} \right) dt \\
&= \frac{1}{T_1} \left(\int_0^{T_1} e^{-j2k\pi f_1 t} + \frac{1}{2} \left(e^{j(2\pi f_1 t(1-k) + \pi/4)} + e^{-j(2\pi f_1 t(1+k) + \pi/4)} \right) \right) dt \\
&= \begin{cases} 1 \text{ V} & \text{für } k = 0 \quad (\text{DC}) \\ \frac{1}{2} e^{j\pi/4} \text{ V} & \text{für } k = +1 \quad (+1 \cdot f_1 = +1 \text{ kHz}) \\ -\frac{1}{2} e^{j\pi/4} \text{ V} & \text{für } k = -1 \quad (-1 \cdot f_1 = -1 \text{ kHz}) \\ 0 & \text{sonst} \end{cases}
\end{aligned}$$

da die Integrale über die Exponentialfunktion mit Periodizität kT_1 und dem Integrationsintervall von T_1 nur einen Beitrag liefern wenn der Exponent Null ist. Natürlich lässt sich das Spektrum hier auch durch „genaues Hinschauen“ ermitteln, da im Integral nur zwei komplexe Drehzeiger $0,5 \text{ V} \cdot e^{\pm j(2\pi f_1 t + \pi/4)}$ und ein DC-Wert stehen, deren Frequenzen / Beträge / Phasen man direkt ablesen kann.

b) **Abgetastetes Signal und Abtastperiode**

Bei einer Abtastfrequenz f_S und Abtastperiode T_S von

$$f_S = 3f_1 = 3 \text{ kHz} \Rightarrow T_S = (3f_1)^{-1} = 0,333 \text{ ms}$$

erhält man die Datenpunkte

$$\begin{aligned}
\Rightarrow s[0] &= s(0 \cdot T_S) = 1 \text{ V} (1 + \cos(\pi/4)) = 1,7071 \text{ V} \\
s[1] &= 1 \text{ V} \cdot (1 + \cos(2\pi \cdot 1000/3000 + \pi/4)) = 0,0341 \text{ V} \\
s[2] &= 1 \text{ V} \cdot (1 + \cos(2\pi \cdot 2000/3000 + \pi/4)) = 1,2588 \text{ V} \\
s[3] &= 1 \text{ V} \cdot (1 + \cos(2\pi \cdot 3000/3000 + \pi/4)) = s[0] = 1,7071 \text{ V} \\
&\dots
\end{aligned}$$

Das abgetastete Signal ist mit $N = 3$ Abtastwerten periodisch, dabei wird eine ganzzahlige Anzahl $K = 1$ Perioden des CT Signals erfasst. Die Anzahl beider Perioden steht in einem ganzzahligen Verhältnis, daher nennt man die Abtastung *kohärent*.

c) **Betrag und Phase der DFT der drei Abtastwerte**

Die DFT (B.24) ist eine Block-Transformation, die zu $N = 3$ Abtastwerten $N = 3$ Frequenzpunkte berechnet:

$$S[k] = \frac{1}{N} \sum_{n=0}^{N-1} s[n] e^{-j2\pi kn/N} = \frac{1}{3} \sum_{n=0}^2 s[n] e^{-j2\pi kn/3}$$

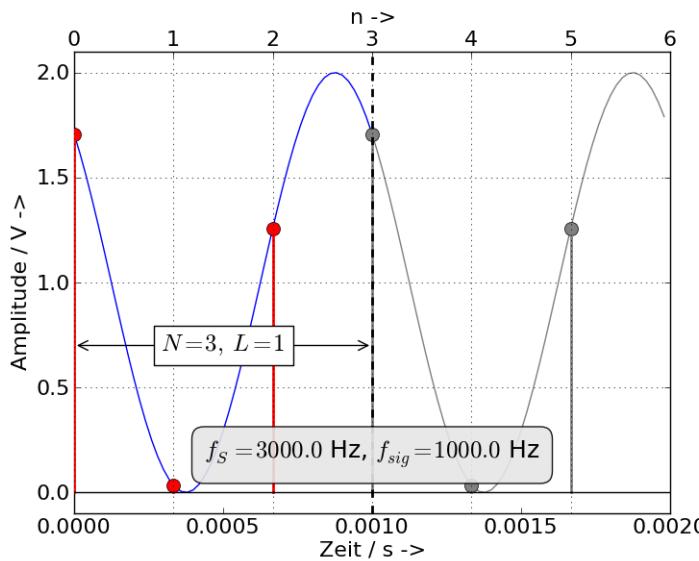


Abb. M3.2.: Eingangssignal $s(t)$ und abgetastetes Signal $s[n]$ (zwei Perioden) zu Aufgabe 3.5

$$\begin{aligned}
 S[0] &= \frac{1V}{3} (1,7071 + 0,0341 + 1,2588) = 1 \text{ V} \\
 S[1] &= \frac{1V}{3} \left(1,7071 + 0,0341e^{-j2\pi/3} + 1,2588e^{-j4\pi/3} \right) \\
 &= \frac{1V}{3} (1,7071 + 0,0341(-0,5 - 0,8660 j) + 1,2589(-0,5 + 0,8660 j)) \text{ V} \\
 &= (0,3536 + 0,3536 j) \text{ V} = 0,5e^{j\pi/4} \text{ V} \\
 S[2] &= \frac{1V}{3} \left(1,7071 + 0,0341e^{-j4\pi/3} + 1,2588 \underbrace{e^{-j8\pi/3}}_{e^{-j2\pi/3}} \right) \\
 &= \frac{1V}{3} (1,7071 + 0,0341(-0,5 + 0,8660 j) + 1,2589(-0,5 - 0,8660 j)) \text{ V} \\
 &= (0,3536 - 0,3536 j) \text{ V} = 0,5e^{-j\pi/4} \text{ V} = S[-1] = S^*[1]
 \end{aligned}$$

Die explizite Berechnung von $S[2]$ wäre also nicht notwendig gewesen, da die DFT mit $N = 3$ periodisch ist und bei reellen Zeitsignalen $S[k] = S^*[-k]$ ist.

Skalierung der Frequenzpunkte:

Im Zeitbereich wird das Signal repräsentiert durch eine Periode der unendlich ausgedehnten Folge $x[n]$ mit N Abtastwerten im Abstand T_S über L Perioden T_1 des CT Signals. Insgesamt wird also eine Zeitspanne $T = NT_S = LT_1$ abgedeckt.

Im Frequenzbereich treten periodische Spektren auf (zeitdiskretes Signal), hier wird das Signal repräsentiert durch die Folge $X[k]$ mit N Frequenzpunkten im Intervall $[0, f_S]$. Der Index k entspricht daher der Frequenz $f_S k/N$, $k = 0 \dots N - 1$, die maximale Frequenzauflösung ist $\Delta f = f_S/N$.

d) Vergleich CFT mit DFT

Man sieht, dass die DFT für periodische, auf $B < f_S/2$ bandbegrenzte CT Signale das gleiche Ergebnis liefern kann wie die CFT. Voraussetzung dazu ist, dass eine ganzzahlige Anzahl L Perioden des CT Signals $s(t)$ so abgetastet wird, dass eine Folge $s[n]$ mit Periode N entsteht. Hierfür müssen Abtastperiode T_S und Signalperiode T_1 „kommensurabel“ (lat.:

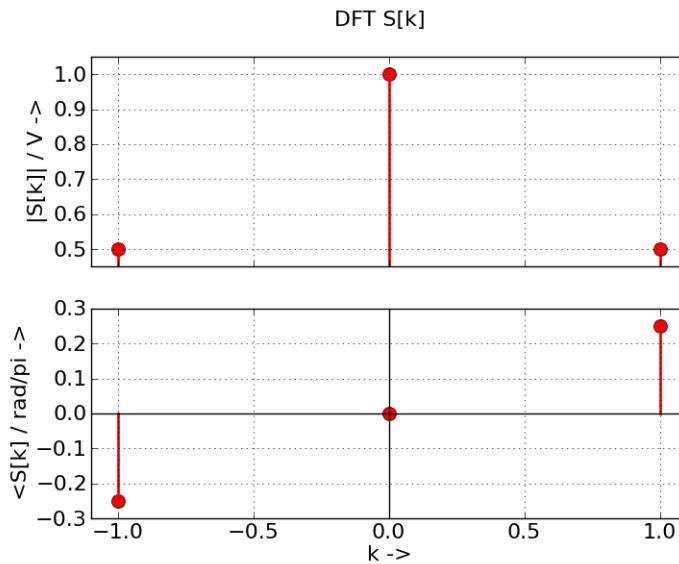


Abb. M3.3.: Betrag und Phase der DFT $S[k]$ für $f_S = 3000$ Hz und $N = 3$ zu Aufgabe 3.5c

zusammen messbar) sein, d.h. ganzzahlige Vielfache einer gemeinsamen Zeiteinheit T_c sein mit

$$NT_S = LT_1 = NLT_c$$

Mathematisch ausgedrückt:

$$\exists T_c \in \mathbb{R} \text{ so, dass } T_1 = NT_c \wedge T_S = LT_c \text{ mit } T_1, T_S \in \mathbb{R} \text{ und } N, L \in \mathbb{Z}.$$

$$\Rightarrow \frac{T_1}{T_S} = \frac{f_S}{f_1} = \frac{N}{L} = c \in \mathbb{Q}.$$

Abtastfrequenz und Signalfrequenz müssen also in einem rationalen Verhältnis stehen. Dies ist in der Praxis nicht immer zu erreichen, da die maximale Anzahl der Abtastpunkte N aus Gründen der Messdauer, des Speicherplatzes und der Rechenzeit begrenzt ist. Besonders effiziente DFT Algorithmen (Fast Fourier Transform, FFT) erhält man für $N = 2^r$, was zusätzliche Einschränkungen mit sich bringt. Siehe hierzu auch Aufgabe 1.1 (Periodizität abgetasteter Signale).

e) DFT mit 3 Abtastwerten pro Periode über zwei Perioden

Berechnet man die DFT über 2 Perioden, wiederholen sich die Abtastwerte:

$$S[0] = \frac{1V}{6} (1,7071 + 0,0341 + 1,2588 + 1,7071 + 0,0341 + 1,2588) = 1 \text{ V}$$

$$S[1] = \frac{1V}{6} \left(1,7071 + 0,0341e^{-j2\pi/6} + 1,2588e^{-j4\pi/6} + 1,7071 \underbrace{e^{-j6\pi/6}}_{=-1} + 0,0341 \underbrace{e^{-j8\pi/6}}_{=-e^{-j2\pi/6}} + 1,2588 \underbrace{e^{-j10\pi/6}}_{=-e^{-j4\pi/6}} \right) = 0 \text{ V}$$

$$S[2] = \frac{1V}{6} \left(1,7071 + 0,0341e^{-j4\pi/6} + 1,2588e^{-j8\pi/6} + 1,7071 \underbrace{e^{-j12\pi/6}}_{=+1} + 0,0341 \underbrace{e^{-j16\pi/6}}_{=+e^{-j4\pi/6}} + 1,2588 \underbrace{e^{-j20\pi/6}}_{=+e^{-j8\pi/6}} \right) = 0,5e^{-j\pi/4} \text{ V}$$

...

Jeder zweite Frequenzpunkt ist also Null. Das kann man leicht auch ohne Rechnung erkennen: Der Index k entspricht Frequenzen von $k/Nf_S = k \cdot 500$ Hz, die Grundfrequenz ist aber 1 kHz. Sinnvoll kann eine solche synchrone Abtastung über mehrere Perioden trotzdem sein: Bei einer rotierenden Welle können so Frequenzkomponenten unterhalb der Grundfrequenz (Subharmonische) detektiert werden. Liefern Frequenzkomponenten unerwarteterweise einen Beitrag, kann das auch ein Hinweis auf eine nicht ganz synchrone Abtastung sein.

f) Sechs Abtastwerte pro Periode

Um sechs Abtastwerte je Periode zu erzielen, muss man hier die Abtastfrequenz auf $f_{S2} = 6$ kHz erhöhen. Mit der Abtastfrequenz erhöht sich der abgedeckte Spektralbereich $-f_{S2}/2 \dots + f_{S2}/2 = \pm 3$ kHz, aber nicht die Frequenzauflösung $f_{S2}/N = 1$ kHz.

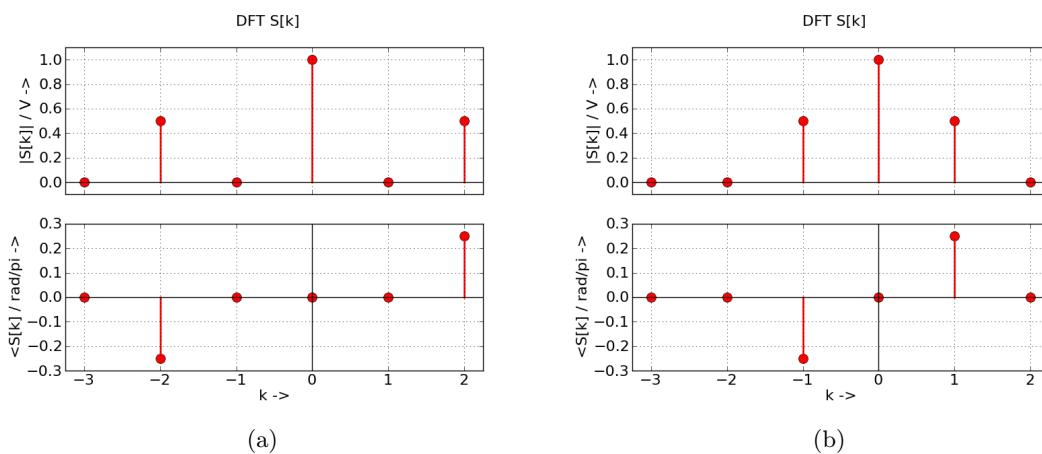


Abb. M3.4.: Betrag und Phase der DFT $S[k]$ für $N = 6$ mit $f_S = 3000$ Hz zu Aufgabe 3.5e (a) und mit $f_S = 6000$ Hz zu Aufgabe 3.5f (b)

g) Zeropadding mit drei Nullen auf insgesamt 6 Abtastwerte

Setzt man die „normale“ Gleichung der DFT an, erhält man (die letzten drei Abtastwerte sind Null):

$$S[0] = \frac{1V}{6} (1,7071 + 0,0341 + 1,2588 + 0 + 0 + 0) = 0,5 \text{ V}$$

Um einen korrekten Skalierungsfaktor zu erhalten, dividiert man bei Zeropadding also nicht durch die Länge N der DFT, sondern durch die Anzahl L der relevanten Abtastwerte:

$$\begin{aligned} S[0] &= \frac{1V}{3} (1,7071 + 0,0341 + 1,2588 + 0 + 0 + 0) = 1 \text{ V} \\ S[1] &= \frac{1V}{3} (1,7071 + 0,0341e^{-j2\pi/6} + 1,2588e^{-j4\pi/6}) = 0,5220e^{-j0,2536\pi} \text{ V} \\ S[2] &= \frac{1V}{3} (1,7071 + 0,0341e^{-j4\pi/6} + 1,2588e^{-j8\pi/6}) = 0,5e^{-j\pi/4} \text{ V} \\ &\dots \end{aligned}$$

Die Werte für $S[0]$ und $S[2]$ sind identisch mit den Werten für $N = 6$ aus Unterpunkt e), der Wert für $S[1]$ ist ein Zwischenwert, der dem Wert der DTFT entspricht.

h) Anzahl der Rechenoperationen einer N -Punkt DFT

Pro Frequenzpunkt müssen alle N reellen Abtastwerte mit komplexen Drehzeigern multipliziert und anschließend aufsummiert werden, es müssen daher jeweils $2N$ reelle Multiplikationen und Additionen durchgeführt werden. Für die Berechnung der gesamten DFT sind also jeweils $2N^2$ reelle Multiplikationen und Additionen notwendigt. Aufgrund der Symmetrie $S[k] = S^*[-k]$ bei reellen Zeitsignalen muss nur die Hälfte der Frequenzpunkte berechnet werden, insgesamt müssen daher N^2 reelle Multiplikationen und Additionen berechnet werden.

i) Simulation in Python / Matlab

Beide Programme erwarten einen Vektor x_n mit den zeitdiskreten Funktionswerten:

```

1 # ... Ende der Import-Anweisungen
2 N_FFT = 3;
3 f_a = 1e3; T_mess = 1. / f_a
4 t = linspace(0,T_mess,N_FFT)
5 xn = 1 + 1 * cos(2*pi*t*f_a)
6 # calculate DFT and scale it with 1/N:
7 Xn = fft(xn)/len(xn)
8 Xn = fftshift(Xn) # center DFT around f = 0
9 # create f-Vector, centered around f = 0:
10 f = fftshift(freq(len(xn),d=1.0/len(xn)))
11 # set phase = 0 for very small magnitudes:
12 for i in range(len(xn)):
13     if abs(Xn[i]/max(abs(Xn))) < 1.0e-10:
14         Xn[i] = 0
15 figure(1)
16 subplot(211)
17 stem(f,abs(Xn))
18 subplot(212)
19 stem(f,angle(Xn)/pi)
20 plt.tight_layout(); plt.show()

```

Lst. M3.1: Python Listing zu Aufgabe 3.5

```

%
N_FFT = 3;
f_a = 1e3; T_mess = 1. / f_a;
t = linspace(0,T_mess-T_mess/NFFT,N_FFT)
xn = 1 + 1 * cos(2*pi*t*f_a);
% calculate DFT and scale it with 1/N:
Xn = fft(xn,length(xn))/length(xn);
%
%
f = linspace(0,1,length(xn));
for i=1:length(Xn)
    if abs(Xn(i))/max(abs(Xn(i)))<1e-10
        Xn(i) = 0;
    endif
endfor
figure(1);
subplot(211);
stem(f,abs(Xn));
subplot(212);
stem(f,angle(Xn)/pi);

```

Lst. M3.2: Matlab Listing zu Aufgabe 3.5

M3.6. * DFT periodischer Signale mit Python / Matlab → Aufgabe 3.6

a) Das erwartete Spektrum des periodischen zeitkontinuierlichen Signals

$$y(t) = 1 + 0,5 \sin(2\pi t \cdot 1 \text{ kHz}) + 0,2 \cos(2\pi t \cdot 1,1 \text{ kHz})$$

ist ein *aperiodisches Linienspektrum* (= *diskrete Linien*). Die Koeffizienten a_k, b_k der *Fourierreihe* entsprechen der Amplitude der Cosinus- und Sinusfunktionen, aus denen sich leicht der Amplitudenbetrag $A_k = \sqrt{a_k^2 + b_k^2}$ oder die Leistung $P_k = A_k^2/2$ der k -ten

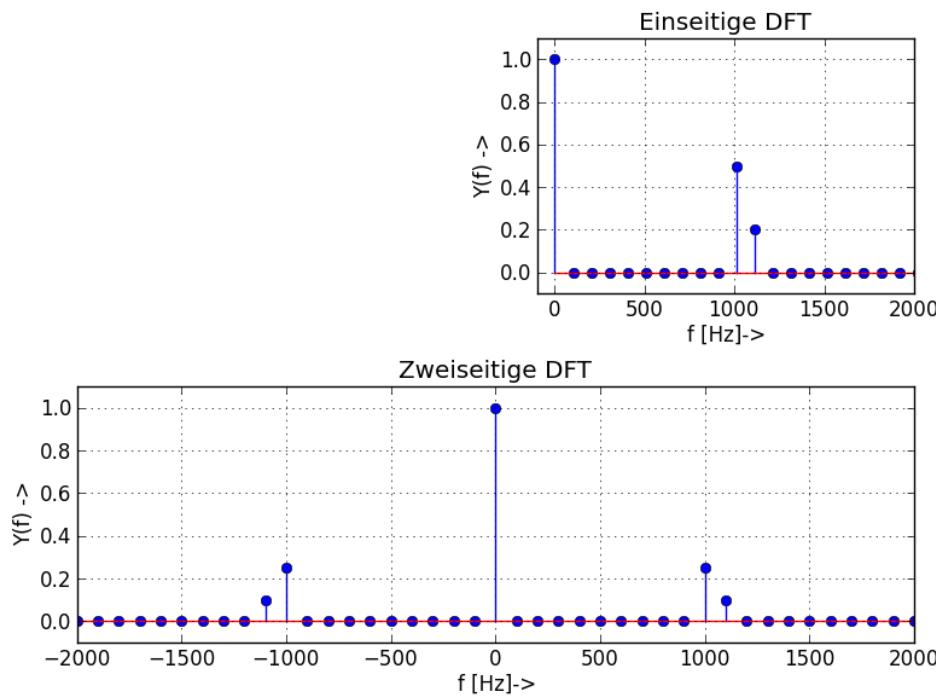


Abb. M3.5.: Erwartetes ein- und zweiseitiges Spektrum zu Aufgabe 3.6

Spektralkomponenten (Linie) berechnen lässt:

$$y(t) = a_0/2 + \sum_{k=1}^{\infty} a_k \cos 2\pi k f_1 t + b_k \sin 2\pi k f_1 t = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi k f_1 t}$$

$$\circlearrowleft Y(f) = a_0/2 + \sum_{k=1}^{\infty} a_k (\delta(f - kf_1) + \delta(f + kf_1)) + j b_k (\delta(f - kf_1) - \delta(f + kf_1))$$

Die Grundfrequenz (größter gemeinsamer Teiler der beiden Frequenzen) ist hier $f_1 = 100$ Hz, dementsprechend sind die Fourierkoeffizienten $a_0 = 2$, $b_{10} = 0,5$ und $a_{11} = 0,2$, der Index bezeichnet dabei die Nummer der Oberwelle.

Aufgetragen als einseitiges ($f \geq 0$) Betragsspektrum $|Y(f)|$ ergeben sich Linien bei $f_0 = 0$ (DC), $f_{10} = 1000$ Hz und $f_{11} = 1100$ Hz mit den Amplitudenbeträgen $A_0 = 1$, $A_{10} = 0,5$ und $A_{11} = 0,2$.

Anmerkung:

Bei reellen Eingangssignalen kann man die Darstellung als *komplexe Fourierreihe*, $f(t) = \sum_{k=-\infty}^{\infty} c_k e^{jk t}$, leicht von der reellen Fourierreihe ableiten (nicht gefragt), da dann gilt

$$c_k = \frac{a_k - jb_k}{2} = c_{-k}^* \quad \text{und} \quad c_0 = \frac{a_0}{2}$$

Die negativen Koeffizienten ($k < 0$) liefern also für reelle Eingangssignale keine zusätzlichen Informationen und werden daher meist nicht dargestellt. Um trotzdem eine korrekte Gesamtleistung bei einseitiger Darstellung ($k \geq 0$) zu bekommen, müssen die Koeffizienten c_k für $k > 0$ verdoppelt werden; c_0 hat bereits den korrekten Wert.

- b) Die **Simulation** von Listing 3.2 bzw. 3.3 mit angepassten Signalfrequenzen liefert zunächst ein verschmiertes Spektrum („Leckeffekt“), in dem man das erwartete Spektrum nur mit etwas Phantasie erkennt. Die Spektrallinien bei f_a und f_b lassen sich nicht unterscheiden.

- c) Der **Leckeffekt** kann durch *kohärente Abtastung* vermieden werden, d.h. indem man die Länge T_{mess} des Messfensters als ganzzahliges Vielfaches der Grundperiode (hier: $T_1 = 1/f_1 = 10$ ms) wählt, $T_{mess} = kT_1$. Im einfachsten Fall führt man die FFT über *eine* Grundschwingung durch und wählt $T_{mess} = T_1 = 10$ ms. Abtastperiode und N_{FFT} müssen passend dazu gewählt werden, damit $T_S = T_{mess}/N_{FFT} = kT_1/T_{mess}$ erfüllt wird.

Ein Array mit passenden Punkten für die Frequenzachse erhält man in Matlab mit `f = linspace(-f_S/2, f_S/2 - f_S/NFFT, NFFT)`. In Python kann man das Gleiche mit der Hilfsfunktion `scipy.fft.freq(NFFT, T_S)` erreichen.

- d) Eine **Verbesserung der Frequenzauflösung** ist nur möglich durch Verlängerung des Messfensters mit $\Delta f = 1/T_{mess} = 1/(N_{FFT}T_S)$. Das ist möglich durch *mehr Abtastpunkte* bei gleichbleibender Abtastrate, also z.B. mit `N_FFT = 200` $\Rightarrow \Delta f = 1/(200T_S) =$

Erhöht man die Abtastrate bei gleichbleibender Anzahl von Abtastpunkten, z.B. mit `T_S = 1e-5`, verringert sich sogar die Frequenzauflösung, da das Messfenster dann kürzer wird.

- e) Für eine **kohärente FFT mit 2^m Punkten** muss die Abtastrate so angepasst werden, dass das Messfenster wieder ein ganzzahliges Vielfaches der Periodenlänge ist, z.B. mit `T_mess = 1e-2; N_FFT = 128; T_S = T_mess / N_FFT; f_S = 1/T_S`. Diese Bedingung ist nicht immer zu erreichen, da oft die Abtastrate durch die verwendete Hardware vorgegeben ist.

f) Leistungsberechnung / Amplitudenskalierung der DFT

Analytische Rechnung

Im Zeitbereich gilt:

$$\begin{aligned} P &= \frac{1}{T_1} \int_{t=0}^{T_1} y^2(\tau) d\tau = \frac{1}{T_1} \int_{t=0}^{T_1} (1 + 0,5 \sin(2\pi \cdot 10f_1\tau) + 0,2 \cos(2\pi \cdot 11f_1\tau))^2 d\tau \\ &= \frac{1}{T_1} \int_{t=0}^{T_1} 1^2 + 0,25 \sin^2(2\pi \cdot 10f_1\tau) + 0,04 \cos^2(2\pi \cdot 11f_1\tau) d\tau \\ &= 1 + 0,125 + 0,02 = A_0^2 + \frac{A_{10}^2}{2} + \frac{A_{11}^2}{2} = 1,145 \end{aligned}$$

Beim Integrieren über eine Periode T_1 liefern alle Terme $\sin 2\pi k f_1 t \cos 2\pi l f_1 t$ den Wert 0, für $k \neq l$ auch die Terme $\sin 2\pi k f_1 t \sin 2\pi l f_1 t$ und $\cos 2\pi k f_1 t \cos 2\pi l f_1 t$. Übrig bleiben daher nur die \sin^2 und \cos^2 Terme, deren Integral sich vergleichsweise leicht berechnen lässt mit

$$\int_{t=0}^{T_1} \cos^2(2\pi k f_1 \tau) d\tau = \frac{\tau}{2} + \frac{\sin(2\pi k f_1 \tau) \cos(2\pi k f_1 \tau)}{2 \cdot 2\pi k f_1} \Big|_{\tau=0}^{T_1} = T_1/2.$$

Für periodische Signale ist der Vergleich von Zeit- und Frequenzbereich einfach, da hier nur die Koeffizienten der Sinus- und Kosinusterme und der Betrag der Spektrallinien verglichen werden müssen.

$$P = \frac{a_0^2}{2} + \sum_{k=1}^{\infty} \frac{a_k^2 + b_k^2}{2} = A_0^2 + \sum_{k=1}^{\infty} \frac{A_k^2}{2}$$

Falls man übrigens vergessen hat, wie das mit dem Integrieren funktioniert, hilft einem das Python Modul **sympy** für symbolische Rechnungen auf die Sprünge:

```

1 from sympy import integrate, cos, pi
2 from sympy.abc import f,x
3 print integrate(cos(x*f)**2, x)
4 print integrate(cos(x*f)**2, x).expand(mul = True)
5 print integrate(cos(2*pi*10*f*x)**2, (x, 0, 1/f))

```

Lst. M3.3: Symbolische Integration mit SymPy

Simulation

Im Zeitbereich nähert man das Integral durch Summation, also mit

$$P = \frac{1}{T_1} \int_{t=0}^{T_1} y^2(\tau) d\tau \approx \frac{1}{T_1} \sum_{n=0}^{T_1/T_S} y^2[n] T_S. \quad (\text{M3.1})$$

In Python / Matlab muss für die korrekte Skalierung das Amplitudenspektrum durch die Anzahl der DFT - Punkte N_{FFT} dividiert werden. Bei einseitigen Spektren über $0 \dots f_S/2$ müssen mit Ausnahme des DC-Werts (Bin 0) außerdem die Amplituden verdoppelt werden (s.o.).

Achtung: Python beginnt Arrays beim Index 0, Matlab beim Index 1. Zum Ausgleich ist bei `a = arange(0,NFFT)` oder `a = x[0:NFFT]` der Wert NFFT bzw. das Element `x[NFFT]` *nicht* mit eingeschlossen, bei den entsprechenden Matlab Kommandos `a = 1:NFFT` bzw. `a = x(1:NFFT)` aber schon!

In Python / Matlab lässt sich die Leistung im Frequenzbereich am elegantesten über das Skalarprodukt aus `Sy` und des dazu konjugierten und transponierten Arrays bestimmen (siehe [M3.4](#) bzw. Listing [M3.5](#)). Die Matlab Syntax ist hier eleganter (wenn auch schwieriger zu verstehen): `Sy'` konjugiert und transponiert `Sy`, `Sy * Sy'` berechnet das Skalarprodukt (da die Arrays transponiert zu einander sind). Aufgrund von Rundungsfehlern hat das Ergebnis einen minimalen imaginären Anteil, den man ggf. noch beseitigen muss, wenn das Ergebnis weiterverarbeitet werden soll.

Alternativ (langsamer) kann man die Leistung auch über `sum(abs(Sy)**2)` (Python) bzw. `sum(abs(Sy).^2)` (Matlab) bestimmen. Die Syntax bewirkt jeweils eine elementweise Quadrierung.

```

1 # ... Ende der Import-Anweisungen
2 f_S = 1e4; T_S = 1/f_S
3 N_FFT = 128; T_mess = N_FFT * T_S
4 f_a = 1e3; f_b = 1.1e3; DC = 1.
5 t = arange(0, T_mess, T_S) # start / stop / step
6 y = DC + 0.5 * sin(2 * pi * t * f_a) \
    + 0.2 * cos(2 * pi * t * f_b)
7 print ('P = ', np.sum(y**2) * T_S / T_mess)
8 figure(1) # two-sided spectrum
9 subplot(212)
10
11 Sy = fft(y, N_FFT) / N_FFT # calculate DFT at
12 # f = [0 ... f_S/2[, [-f_S/2 ... 0[
% f_S = 5e3; T_S = 1 / f_S;
N_FFT = 1000; T_mess = N_FFT * T_S;
f_a = 1e3; f_b = 1.1e3; DC = 1;
t = [0:1:N_FFT-1]*T_S;
y = DC + 0.5 * sin(2*pi*t*f_a) ...
    + 0.2 * cos(2*pi*t*f_b);
fprintf('P = %g\n', sum(y.^2) ...
    * T_S / T_mess);
figure(1); clf;
subplot(212);
Sy = fft(y,N_FFT) / N_FFT;

```

```

13 f = fftfreq(N_FFT, T_S)
14 # freq. points at [0... f_S/2[, [-f_S/2 ... 0[
15 stem(f,abs(Sy)); grid(True)
16 plt.xlim(-2000, 2000); plt.ylim(-0.1, 1.1)
17 xlabel('f [Hz]->'); plt.ylabel('Y(f) ->');
18 title('Zweiseitige DFT')
19 print ('P = ', np.dot(Sy,Sy.conj().T))
20 subplot(222) # one-sided spectrum [0 ... f_S/2[
21 Sy = 2 * fft(y, N_FFT) / N_FFT # ... needs x2
22 Sy[0] = Sy[0] / 2. # adjust DC scaling
23 f = linspace(0, f_S, N_FFT) # f = 0 ... f_S[
24 stem(f[0:N_FFT/2],abs(Sy[0:N_FFT/2])) #... f_S/2[
25 xlabel('f [Hz]->'); plt.ylabel('Y(f) ->')
26 plt.axis([-100,2000,-0.1, 1.1])
27 title('Einseitige DFT')
28 plt.tight_layout(); grid(True); plt.show()

```

Lst. M3.4: Python Listing zu Aufgabe [M3.6](#)

```

f = linspace(-f_S/2, ...
             f_S/2 - f_S/N_FFT, N_FFT);
Sy = fftshift(Sy); % center DC-comp.
stem(f,abs(Sy)); grid on;
xlim([-2000,2000]); ylim([-0.1,1.1]);
xlabel('f [Hz]->'); ylabel('Y(f) ->');
title('Zweiseitige DFT')
fprintf('P = %g\n', Sy * Sy');
subplot(222);
Sy = 2*fft(y,N_FFT)/N_FFT;
Sy(1) = Sy(1)/2;
f = linspace(0, f_S/2, N_FFT/2);
stem(f, abs(Sy(1:N_FFT/2)));
xlabel('f [Hz]->'); ylabel('Y(f) ->');
axis([-100,2000,-0.1, 1.1]);
title('Einseitige DFT'); grid on;

```

Lst. M3.5: Matlab Listing zu [M3.6](#)

M3.7. Kohärente DFT einer Rechteckschwingung → [Aufgabe 3.7](#)

a) Die Fourierkoeffizienten der CT Funktion lauten:

$$c_0 = 0, \quad c_1 = 2A/\pi \approx 0,63A, \quad c_2 = 0, \quad c_3 = 2A/3\pi \approx 0,42A.$$

$$\begin{aligned}
c_k &= \frac{1}{T_1} \int_0^{T_1} u(t) e^{-j2k\pi f_1 t} dt = \frac{A}{T_1} \left[\int_0^{T_1/2} e^{-j2k\pi f_1 t} dt - \int_{-T_1/2}^0 e^{-j2k\pi f_1 t} dt \right] \\
&= \frac{A}{T_1} \left[\frac{e^{-j2k\pi f_1 t}}{-j2k\pi f_1} \Big|_0^{T_1/2} - \frac{e^{-j2k\pi f_1 t}}{-j2k\pi f_1} \Big|_{-T_1/2}^0 \right] \\
&= -\frac{A}{j2k\pi} \left[e^{-j2k\pi f_1 t} \Big|_{t=T_1/2} - 1 - 1 + e^{-j2k\pi f_1 t} \Big|_{t=-T_1/2} \right] = -\frac{A}{jk\pi} \left[e^{-jk\pi} - 1 \right] \\
&= \begin{cases} 0 & \text{für gerade } k \\ \frac{2A}{jk\pi} = -j\frac{2A}{k\pi} & \text{für ungerade } k \end{cases}
\end{aligned}$$

b) Abgetastete Rechteckfunktion

Die abgetastete Funktion erhält man durch

$$u[n] = \begin{cases} -A & \text{für } -1/2 \leq nT_S/kT_1 < 0 \\ +A & \text{für } 0 \leq nT_S/kT_1 < 1/2 \end{cases}$$

Die Koeffizienten der DFT sind gegeben durch:

$$S[k] = \frac{1}{N} \sum_{n=0}^{N-1} u[n] e^{-j2\pi kn/N}$$

Bei einer Abtastung mit $f_S = 4$ kHz benötigt man $N = 4$ Samples für eine synchrone DFT:

$$\begin{aligned} S[0] &= \frac{A}{4} (e^0 + e^0 - e^0 - e^0) = \mathbf{0} \\ S[1] &= \frac{A}{4} (e^0 + e^{-j\pi/2} - e^{-j\pi} - e^{-j3\pi/2}) = \frac{A}{4} (1 - j + 1 - j) = \frac{A}{\sqrt{2}} e^{-j\pi/4} \\ S[2] &= \frac{A}{4} (e^0 + e^{-j\pi} - e^{-j2\pi} - e^{-j3\pi}) = \mathbf{0} \\ S[3] &= S[-1] = S^*[1] = \frac{A}{4} (e^0 + e^{-j3\pi/2} - e^{-j3\pi} - e^{-j\pi9/2}) = \frac{A}{\sqrt{2}} e^{+j\pi/4} \end{aligned}$$

Die Abweichungen zur CFT sind eine Folge von *Aliasing* (Kap. 7): Bei einer Abtastung mit 4 kHz können keine Frequenzkomponenten ≥ 2 kHz dargestellt werden. Sind im ursprünglichen Signal höherfrequente Komponenten enthalten (wie in der Rechteckschwingung), werden diese auf niederfrequente Komponenten abgebildet und verfälschen diese. Erhöht man die Abtastfrequenz, reduziert sich hier der Fehler durch Aliasing, da die Amplituden der Oberschwingungen mit $1/k$ abnehmen. Für eine genaue Analyse müsste man aber das Signal bandbegrenzen auf $f < f_S/2$ (siehe Kap. 7).

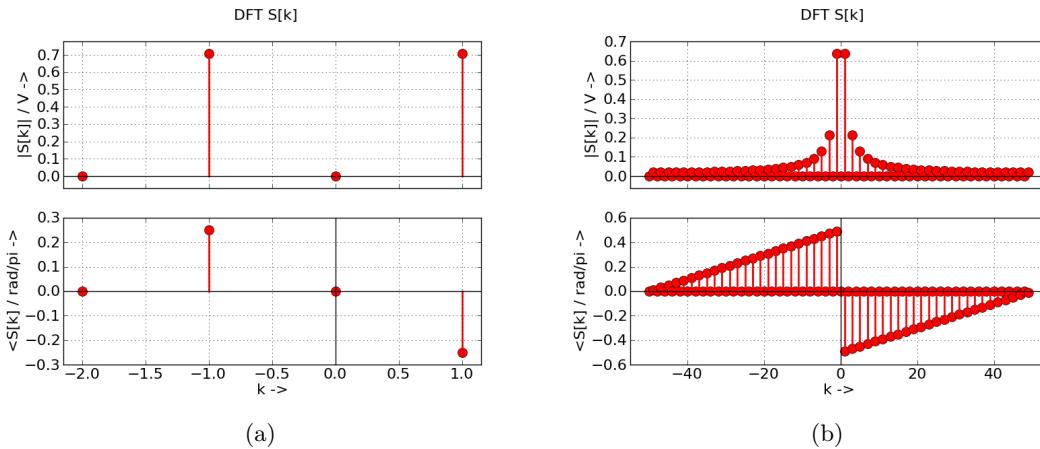


Abb. M3.6.: Betrag und Phase der DFT $S[k]$ für $f_S = 4000$ Hz und $N = 4$ (a) bzw. $f_S = 100$ kHz und $N = 100$ (b) zu Aufgabe 3.7b

M3.8. Fourier-Analyse mit Rechteck-Fensterung → Aufgabe 3.8

a) Spektrum des gefensterten Signals

$$s(t) = A \cos(2\pi f_i t) \circledcirc S(f) = \frac{A}{2} [\delta(f - f_0) + \delta(f + f_0)]$$

oder, über Eulersche Identität und Verschiebungssatz hergeleitet:

$$s(t) = A \cos(2\pi f_i t) = \frac{A}{2} [e^{j2\pi f_0 t} + e^{-j2\pi f_0 t}] \circledcirc S(f) = \frac{A}{2} [\delta(f - f_0) + \delta(f + f_0)]$$

Frequenzantwort der Fensterfunktion $w(t)$:

$$w(t) = \text{rect}\left(\frac{T - T_1/2}{T_1}\right) \circ\bullet W(f) = T_1 \underbrace{\text{si}(\pi f T_1)}_{\text{Verzögerung um } T_1/2} e^{-j2\pi f T_1/2}$$

$$x(t) = w(t) \cdot s(t) \circ\bullet X(f) = W(f) \star S(f)$$

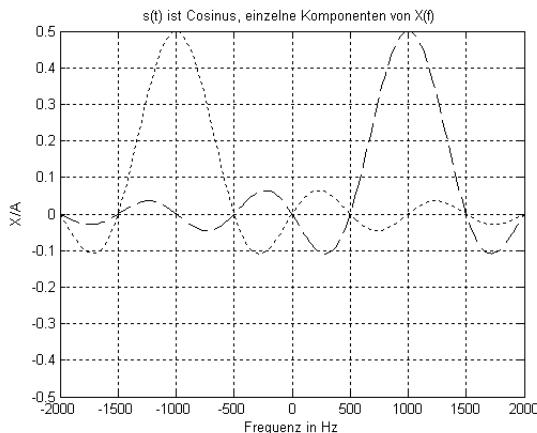
$$\Rightarrow X(f) = \frac{T_1 A}{2} \left\{ e^{-j\pi(f-f_0)T_1} \text{si}[\pi(f-f_0)T_1] + e^{-j\pi(f+f_0)T_1} \text{si}[\pi(f+f_0)T_1] \right\}$$

b) Darstellung des Betragsspektrums von $|X(f)|$

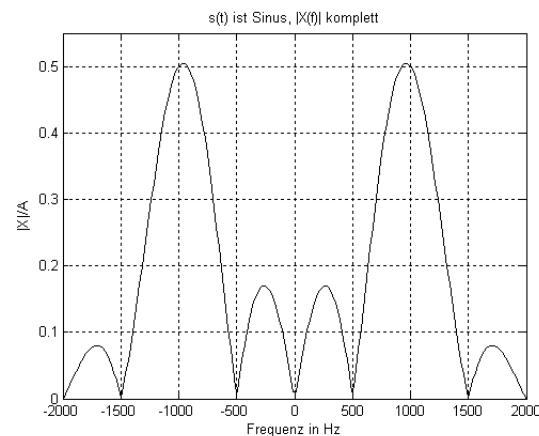
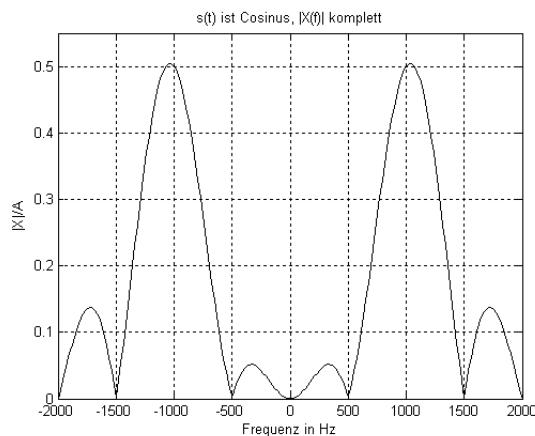
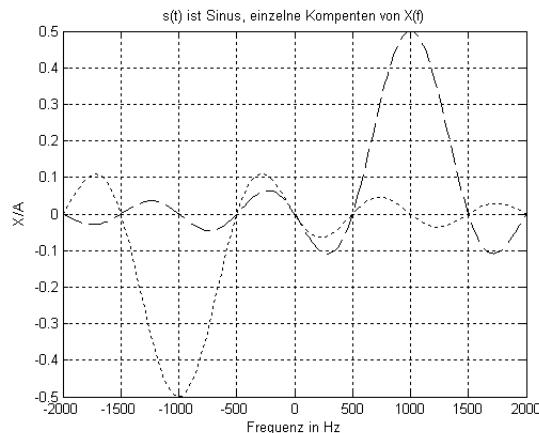
Die Bestimmung des Betragsspektrums fällt deutlich einfacher, wenn man sieht, dass die Verzögerung um $T_1/2$ des Rechteckfensters im Spektrum $W(f)$ lediglich eine lineare Phase $e^{-j2\pi f T_1/2}$ bewirkt. Wählt man daher ein Rechteckfenster $w'(t)$ gleicher Länge, das symmetrisch zur y -Achse ist (gerade Funktion) mit $w'(t) = \text{rect}(T/T_1)$, erhält man eine rein reelle Frequenzantwort $W'(f)$ mit $|W'(f)| = |W(f)|$ und damit auch ein rein reelles Spektrum $X'(f)$. Da $T_1 = 2T_0$ ist, schneiden sowohl $w(t)$ als auch $w'(t)$ zwei ganze Perioden von $s(t)$ mit identischer Startphase aus, daher sind auch die Betragsspektren der aperiodischen Signale identisch:

$$|X(f)| = |X'(f)| = X'(f) = \frac{T_1 A}{2} \{ \text{si}[\pi(f-f_0)T_1] + \text{si}[\pi(f+f_0)T_1] \}$$

$$s(t) = \cos(2\pi f_0 t) \text{ laut Angabe:}$$



$$s(t) = \sin(2\pi f_0 t) \text{ zum Vergleich:}$$



```

1 % Spektrum eines zeitlich begrenzten Cosinus-Signals mit
2 % Zeitfenster T1 in s und Signalfrequenz f0
3 T1=0.002; f0=1000;
4 % Vorzeichen V=1 für Cosinus, V=-1 für Sinus
5 V=1;
6 f=linspace(-2000,2000,200);
7 X1=0.5*(sin(pi*(f-f0)*T1+eps)./(pi*(f-f0)*T1+eps));
8 X2=V*0.5*(sin(pi*(f+f0)*T1+eps)./(pi*(f+f0)*T1+eps));
9 %
10 plot(f,X1,'k-{}-',f,X2,'k:');
11 axis([-2000,2000,-0.5,0.5]); grid;
12 xlabel('Frequenz in Hz'); ylabel('X/A');
13 %
14 figure;
15 plot(f,abs(X1+X2),'k');
16 axis([-2000,2000,0,0.55]); grid;
17 xlabel('Frequenz in Hz'); ylabel('|X|/A');

```

Lst. M3.6: Matlab-Code zu Aufgabe 3.8

c) Näherungsweise Bestimmung von $|X(f)|$

f_S groß genug wählen, um Leckeffekt zu minimieren, dann Zero-Padding um genügend Frequenzpunkte für $X(f)$ zu erhalten (und um auf $N = 2^r$ Punkte für eine effiziente Radix-2 FFT-Berechnung zu kommen)

d) Eine FFT von $s(t)$ ohne Störeffekte

wäre nur möglich mit $N \rightarrow \infty$ Punkten der FFT, da $s(t)$ ein unendlich ausgedehntes Zeitsignal ist! Das entspricht dann einer DTFT mit einer unendlichen Punktzahl, die

nicht numerisch bestimmt werden kann.

Durch Fensterung [$s(t) \rightarrow x(t)$] und Abtastung einer ganzzahligen Anzahl von Perioden kann mit Hilfe der DFT bzw. FFT aus $x(t)$ das Spektrum von $s(t)$ bestimmt werden. Voraussetzung dafür ist, dass die periodische Fortsetzung des Signalausschnitts $x(t)$ identisch mit $s(t)$ ist.

M4. FIL: Einfache digitale Filter und FIR-Filterentwurf

M4.1. * Filterspezifikationen → Aufgabe 4.1

- a) Die **logarithmischen Spezifikationen** lassen sich leicht aus den linearen Spezifikationen berechnen: Eine Schwankung der Ausgangsamplitude im Durchlassbereich von maximal $\pm 10\%$ entspricht beim FIR-Filter $\pm \delta_{DB,FIR} = 0,1$ und beim IIR-Filter $\delta_{DB,IIR} = 0,2$.

$$\text{IIR: } A_{DB,IIR} = -20 \log_{10}(1 - \delta_{DB,IIR}) = 1,94 \text{ dB}$$

$$\begin{aligned} \text{FIR: } A_{DB,FIR} &= 20 \log_{10}(1 + \delta_{DB,FIR}) - 20 \log_{10}(1 - \delta_{DB,FIR}) \\ &= 0,828 \text{ dB} \quad + 0,915 \text{ dB} = 1,74 \text{ dB} \end{aligned}$$

- b) Auch für die **Berechnung der linearen Spezifikationen** aus den logarithmischen Spezifikationen $A_{DB} = 2 \text{ dB}$ und $A_{SB} = 20 \text{ dB}$ müssen FIR und IIR-Filter separat betrachtet werden: Im Sperrband gilt für beide Filter

$$\delta_{SB} = 10^{-A_{SB}/20} = 10^{-1} = 0,1.$$

Im Durchlassband gilt für das IIR-Filter mit (4.2)

$$\delta_{DB,IIR} = 1 - 10^{-A_{DB}/20} = 1 - 10^{-0,1} = 0,206$$

und für das FIR-Filter mit (4.3)

$$\begin{aligned} \delta_{DB,FIR} &= \frac{10^{A_{DB}/20} - 1}{10^{A_{DB}/20} + 1} = 0,115 \\ &\approx 10^{A_{DB}/40} - 1 = 0,122. \end{aligned}$$

Die Näherung für $\delta_{DB,FIR}$ ist schon relativ gut, wird für kleinere Werten von A_{DB} aber noch besser.

Das Equiripple-Filter in Abb. M4.1 benötigt die Ordnung $N = 26$, das Chebychev-Filters (Typ 2) die Ordnung $N = 5$ um die vorgegebenen Spezifikationen zu erfüllen (ermittelt mit Hilfe der Befehle `remezord()` bzw. `cheb2ord()` und durch Ausprobieren).

- c) **Durchlassband:**

Um die **Schwankung der Ausgangsleistung bzw. -spannung** zu berechnen, kann man zunächst wie im vorigen Punkt die linearen Abweichungen der Amplitude $\delta_{DB[A]}$ und der Leistung $\delta_{DB[P]}$ berechnen und mit den Nennwerten multiplizieren. Eine Nennleistung von $P_{nenn} = 80 \text{ W}$ an $R = 8 \Omega$ entspricht einer nominellen Ausgangsamplitude von

$$\hat{u}_{nenn} = \sqrt{2PR} = \sqrt{160 \text{ W} \cdot 8 \Omega} = 35,78 \text{ V.}$$

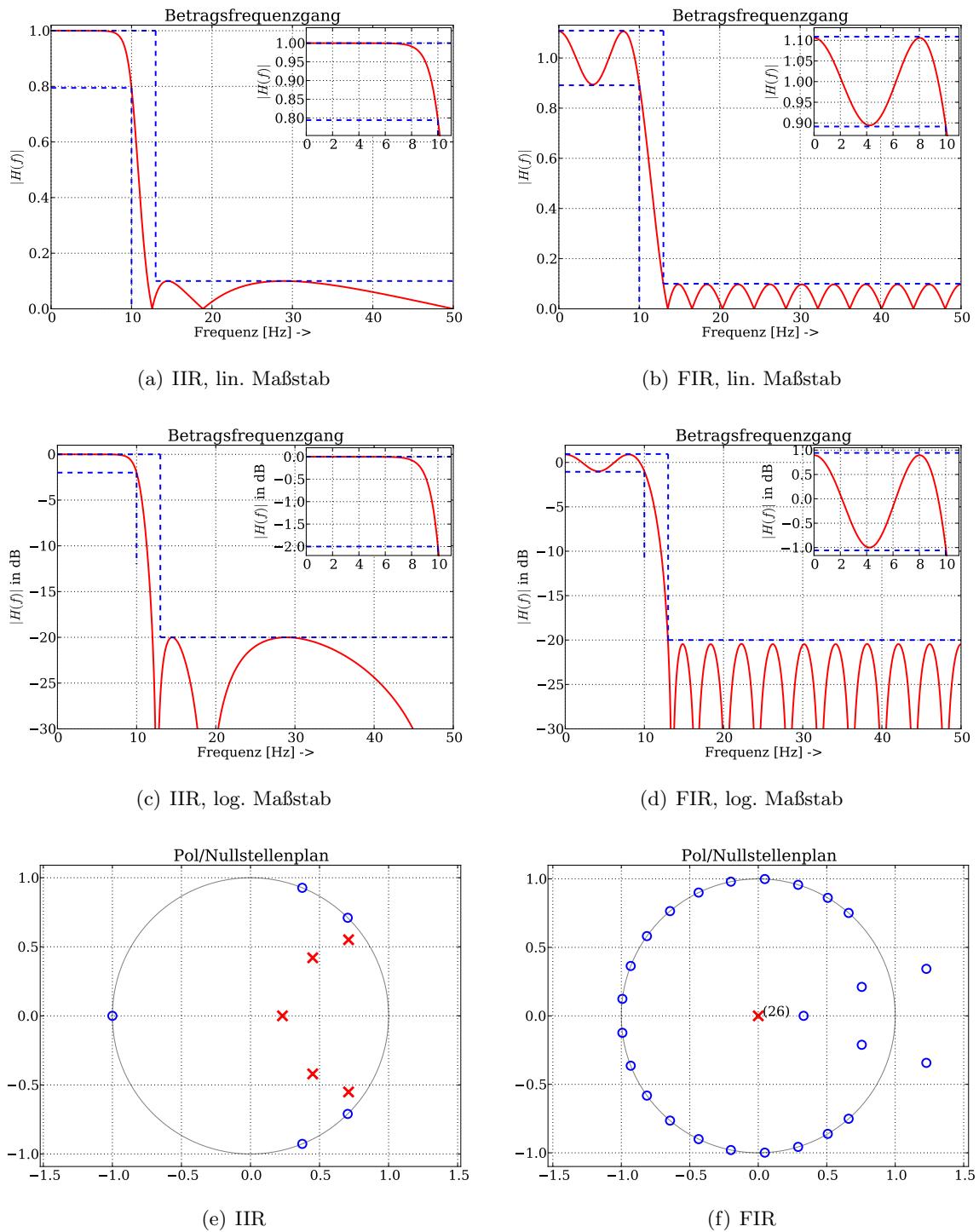


Abb. M4.1.: Frequenzgänge und Spezifikationen im linearen und log. Maßstab sowie P/N-Diagramme für IIR- und FIR-Filter zu Aufgabe 4.1

Wie üblich muss bei Leistungen (quadrierten Signalen) mit $10 \log_{10} P$ und bei Amplituden mit $20 \log_{10} A$ gerechnet werden:

$$\text{Amplitudenvariation: } \delta_{DB,IIR}[A] = 1 - 10^{-A_{DB}/20} = 1 - 10^{-2/20} = 0,206$$

$$\text{Leistungsvariation: } \delta_{DB,IIR}[P] = 1 - 10^{-A_{DB}/10} = 1 - 10^{-2/10} = 0,369$$

Damit können Ausgangsamplitude und -leistung beim IIR-Filter im Durchlassband variieren zwischen:

$$\text{Amplitude: } \hat{u}_{IIR} = \hat{u}_{nenn} \dots \hat{u}_{nenn} (1 - \delta_{DB,IIR[A]}) = 35,78 \dots 28,41 \text{ V}$$

$$\text{Leistung: } P_{IIR} = P_{nenn} \dots P_{nenn} (1 - \delta_{DB,IIR[P]}) = 80 \dots 50,48 \text{ W}$$

Bei einem FIR-Filter gilt analog:

$$\text{Amplitudenvariation: } \pm \delta_{DB,FIR[A]} = \pm \frac{10^{A_{DB}/20} - 1}{10^{A_{DB}/20} + 1} = \pm 0,115$$

$$\text{Leistungsvariation: } \pm \delta_{DB,FIR[P]} = \pm \frac{10^{A_{DB}/10} - 1}{10^{A_{DB}/10} + 1} = \pm 0,226$$

und damit

$$\text{Amplitude: } \hat{u}_{FIR} = \hat{u}_{nenn} (1 \pm \delta_{DB,FIR[A]}) = 39,90 \dots 31,665 \text{ V}$$

$$\text{Leistung: } P_{FIR} = P_{nenn} (1 \pm \delta_{DB,FIR[P]}) = 98,08 \dots 61,92 \text{ W}$$

Wesentlich einfacher ist die Rechnung, wenn man in der logarithmischen Darstellung bleibt:

Bei einem FIR-Filter bedeutet $A_{DB} = 2 \text{ dB}$ dass die Ausgangsleistung im Durchlassband ausgehend von $0 \text{ dB} \hat{=} 80 \text{ W}$ um ca. $\pm 1 \text{ dB} \hat{=} 10^{\pm 1/10}$ schwanken kann.

Sperrband:

Im Sperrband sind die Spezifikationen für IIR- und FIR-Filter identisch definiert. Den Bereich der **Ausgangsspannungen bzw. -leistungen** berechnet man ansonsten genauso wie im Durchlassband ausgehend von $P_{nenn} = 80 \text{ W}$ und $\hat{u}_{nenn} = 35,78 \text{ V}$:

$$\text{Max. Amplitude: } \delta_{DB[A]} = 10^{-A_{SB}/20} = 10^{-20/20} = 0,1$$

$$\text{Max. Leistung: } \delta_{DB[P]} = 10^{-A_{SB}/10} = 10^{-20/10} = 0,01$$

Damit können Ausgangsamplitude und -leistung bei beiden Filtern im Sperrband variieren zwischen:

$$\text{Amplitude: } \hat{u} = 0 \dots \hat{u}_{nenn} \delta_{SB[A]} = 0 \dots 3,58 \text{ V}$$

$$\text{Leistung: } P = 0 \dots P_{nenn} \delta_{SB[P]} = 0 \dots 0,8 \text{ W}$$

d) Filterspezifikationen:

Die Kante des Durchlassbands entspricht der max. Signalfrequenz $f_{sig,max} = 20 \text{ Hz}$, die des Sperrbands der niedrigsten Störfrequenz $f_{br} = 50 \text{ Hz}$. Die Eckfrequenzen für den Filterentwurf sind dementsprechend:

	f_{DB}	f_{SB}	F_{DB}	F_{SB}	Ω_{DB}	Ω_{SB}
Formel	—	—	f_{DB}/f_S	f_{SB}/f_S	$2\pi f_{DB}/f_S$	$2\pi f_{SB}/f_S$
Wert	20 Hz	50 Hz	0,02	0,05	0,126	0,314

Tab. M4.1.: Eckfrequenzen für den Filterentwurf zu Aufgabe 4.1d

Die normalisierte Frequenz F ist einheitenlos, die normalisierte Winkelfrequenz Ω hat die Einheit rad / s / Sample.

Am Eingang des Filters ist die Amplitude des Störsignals doppelt so groß wie die des Nutzsignals, am Ausgang soll sie nur noch ein Tausendstel so groß sein. Daher muss das Störsignal unterdrückt werden um $A_{SB} = 2000 \hat{=} 66$ dB.

Das Nutzsignal soll um maximal $\pm 1\%$ verändert werden, das entspricht

$$\delta_{DB,FIR} = 0,01 \text{ und } A_{DB,FIR} = 20 \log_{10} \frac{1 + \delta_{DB}}{1 - \delta_{DB}} = 0,174 \text{ dB}$$

$$\delta_{DB,IIR} = 0,02 \text{ und } A_{DB,IIR} = -20 \log_{10} 1 - \delta_{DB} = 0,175 \text{ dB}$$

M4.2. * Amplitudengang linearphasiger Filter → Aufgabe 4.2

a) Filtereigenschaften

Das Filter ist linearphasig, weil die Koeffizienten symmetrisch sind (gerade Symmetrie, ungerade Anzahl Koeffizienten → Typ 1).

Das Filter ist ein Halbbandfilter, weil jeder zweite Koeffizient außer dem mittleren gleich Null ist.

b) Nullstelle(n) bei $f_S/2$

Beweis für eine Nullstelle bei $f_S/2$ durch Einsetzen von $z = -1$ in $H(z)$:

⇒ $H(z = -1) = 0$. Um zu zeigen, dass bei $z = z_0$ eine m -fache Nullstelle vorliegt, muss $H(z)$ per Polynomdivision durch $(z - z_0)^m$ dividiert werden, ohne dass ein Rest bleibt:

$$H(z) = \frac{-1 + 5z^{-2} + 8z^{-3} + 5z^{-4} - z^{-6}}{8} = \frac{-z^6 + 5z^4 + 8z^3 + 5z^2 - 1}{8z^6}$$

Hier soll auf eine doppelte Nullstelle bei $z = -1$ untersucht werden, daher wird das Zählerpolynom durch $(z + 1)^2 = z^2 + 2z + 1$ dividiert und untersucht, ob die Division ohne Rest aufgeht:

$$\begin{array}{r}
 \left(-z^6 + 5z^4 + 8z^3 + 5z^2 - 1 \right) \div (z^2 + 2z + 1) = -z^4 + 2z^3 + 2z^2 + 2z - 1 \\
 \underline{-z^6 + 2z^5 + z^4} \\
 \hline
 2z^5 + 6z^4 + 8z^3 \\
 - 2z^5 - 4z^4 - 2z^3 \\
 \hline
 2z^4 + 6z^3 + 5z^2 \\
 - 2z^4 - 4z^3 - 2z^2 \\
 \hline
 2z^3 + 3z^2 \\
 - 2z^3 - 4z^2 - 2z \\
 \hline
 -z^2 - 2z - 1 \\
 z^2 + 2z + 1 \\
 \hline
 0
 \end{array}$$

Da die Division ohne Rest aufgeht, liegt bei $z_{0,1} = z_{0,2} = -1$ eine doppelte Nullstelle vor!

c) Eine weitere Nullstelle auf dem Einheitskreis

kann man aus Abb. 4.2(a) ablesen. Sie liegt ungefähr bei $f_0 = 0,36f_S$ also bei $z_{0,3} = e^{j2 \cdot 0,36\pi}$. Da das System nur reelle Koeffizienten hat, muss es reellwertig sein und daher eine zweite dazu konjugiert-komplexe Nullstelle (= symmetrisch zur x -Achse) haben bei $f_0 = -0,36f_S$ ($z_{0,4} = e^{-j2 \cdot 0,36\pi}$). Die Nullstellen sind einfach, da $|H(e^{j\Omega})|$ an dieser Stelle einen Knick hat, $H(e^{j2\pi F})$ also das Vorzeichen wechselt. Im Gegensatz dazu sieht man bei $|H(e^{j2\pi F})|$ bei $F = 0,5$ (doppelte Nullstelle) keinen Knick.

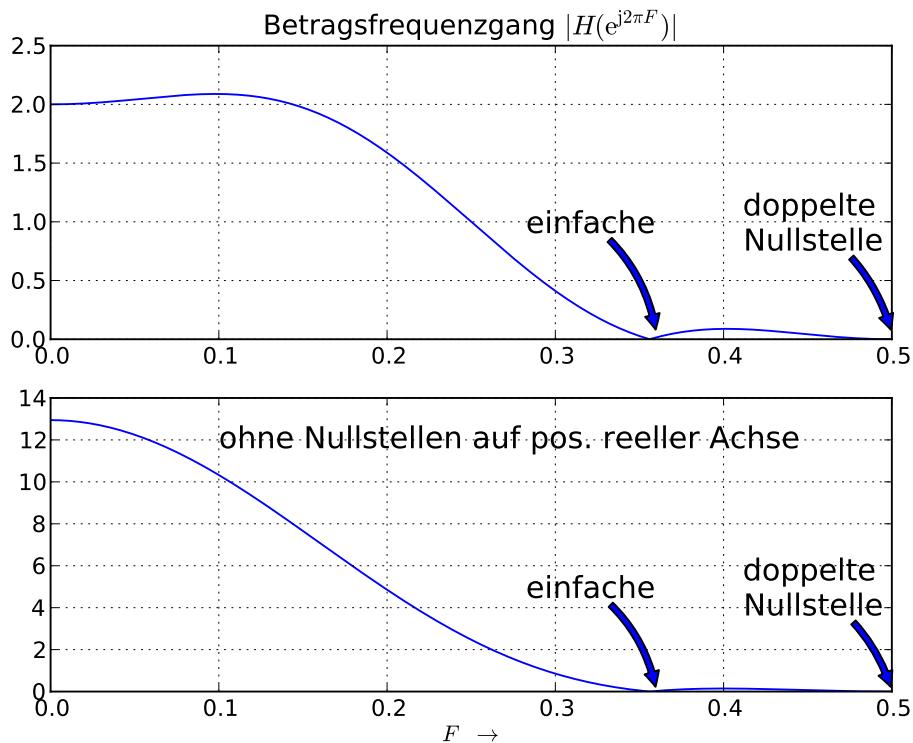


Abb. M4.2.: Amplitudengang mit Kennzeichnung der Nullstellen auf dem Einheitskreis zu Aufgabe 4.2

d) Weitere Nullstellen:

Da das Filter $M = 7$ Koeffizienten hat, hat es die Ordnung $N = 6$ und muss daher auch 6 Nullstellen haben. Damit muss es noch zwei weitere Nullstellen geben, die in $H(e^{j2\pi F})$

i	$z_{0,i}$
1	$-1,0000+0,0000j = e^{j\pi}$
2	$-1,0000+0,0000j = e^{j\pi}$
3	$-0,6180+0,7862j = e^{0,712j\pi}$
4	$-0,6180-0,7862j = e^{-0,712j\pi}$
5	$0,3460+0,0000j = 0,346 \cdot e^0$
6	$2,8901+0,0000j = 2,8901 \cdot e^0$

Tab. M4.2.: Nullstellen des Zählerpolynoms

nicht zu erkennen sind. Da das Filter linearphasig und reellwertig ist, müssen Nullstellen immer konjugiert-komplex und symmetrisch zum Einheitskreis sein. Im Sonderfall $z = -1$ wird das durch eine Nullstelle erreicht (die hier doppelt platziert ist), die Nullstellen auf dem Einheitskreis sind „symmetrisch zu sich selbst“ und treten daher als Paar auf. Die fehlenden zwei Nullstellen müssen daher auf der reellen Achse liegen („konjugiert-komplex zu sich selbst“) und zwar symmetrisch zum Einheitskreis mit $z_{0,5} = r$ und $z_{0,6} = 1/r$. Sie liegen auf der *positiven* reellen Achse, um einen zu frühen Abfall des DBs zu verhindern (vgl. Abb. M4.2 oben und unten). Lässt man die beiden Nullstellen auf der reellen Achse weg, erhöht sich außerdem die DC-Verstärkung.

e) PN-Diagramm

Die Nullstellen in Tab. M4.2 wurden mit Hilfe von Listing M4.1 ermittelt.

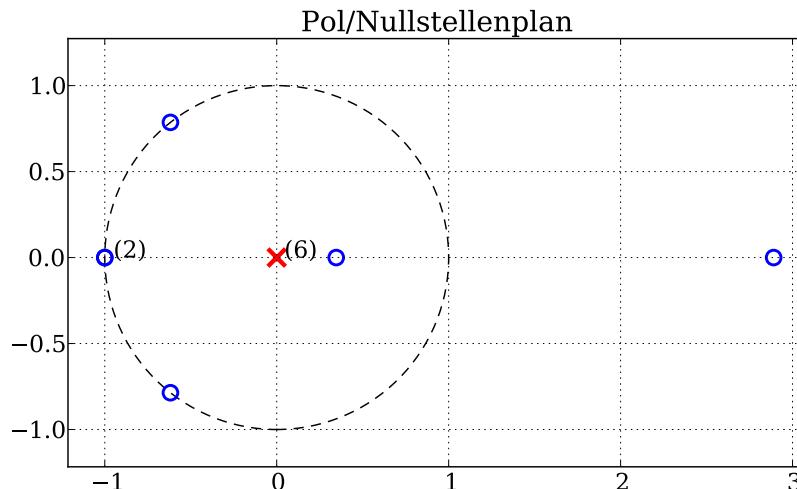


Abb. M4.3.: P/N Diagramm zu Aufgabe 4.2

```

1 # Ende der gemeinsamen Import-Anweisungen
2 f_S = 1
3 bb = [-1/8, 0, 5/8, 1, 5/8, 0, -1/8]
4 aa = 1
5
6 z_0 = np.roots(bb) # Wurzeln des Polynoms,
7
8 print(' i | z_0,i | (polar)')
9 print('-----')
10 for i in range(len(z_0)):
11     print('{0:2d} | {1:15.4f} |{2:6.3f} * e^{(j {3:6.4f} pi)} '\
12         .format(i, z_0[i],abs(z_0[i]),np.angle(z_0[i])/pi))
13     # print('%2d | %15.4f |%6.3f * e^{(j %6.4f pi)}'
14     #     %(i, z_0[i],abs(z_0[i]),np.angle(z_0[i])/pi))
15
16 for i in range(len(z_0)):
17     if z_0[i].real > 0: # Entferne Nullstellen im Durchlassband
18         z_0[i] = 0
19
20 bb_neu = np.poly(z_0) # "Ausmultiplizieren" der Koeffizienten
21
22 [w, H_org] = sig.freqz(bb, aa, 1024)
23 [w, H_neu] = sig.freqz(bb_neu, aa, 1024)
24 f = w / (2 * pi) * f_S
25
26 figure(1)
27 subplot(211)
28 plot(f, abs(H_org)); grid(True)
29 title(r'Betragsfrequenzgang $|H(\mathrm{e}^{\mathrm{j}\Omega F})|$')
30 subplot(212)
31 plot(f,abs(H_neu)); grid(True)
32 xlabel(r'$F \rightarrow$')
33 plt.tight_layout(); plt.show()

```

Lst. M4.1: Python Listing zu Aufgabe 4.2

Das Python3-Format der Print-Anweisung (Zeile 12) ist leider notwendig, da andernfalls die komplexen Werte der Nullstellen nicht richtig angezeigt werden.

M4.3. * Maximal-, minimal- und linearphasige Filter → Aufgabe 4.3

a) Allgemeine Systemfunktion / Übertragungsfunktionen von F_1

Die *Systemfunktion* $H_1(z)$ erhält man einfach durch Einsetzen von Pol- und Nullstelle:

$$H_1(z) = \frac{z - z_{0,1}}{z - z_{\infty,1}} = \frac{z - r_1 e^{j\Omega_1}}{z} = 1 - r_1 e^{j\Omega_1} z^{-1}$$

•→ $h_1[n] = \delta[n] - r_1 e^{j\Omega_1} \delta[n-1] = \delta[n] - r_1 \cos \Omega_1 \delta[n-1] - j r_1 \sin \Omega_1 \delta[n-1]$

Man sieht, dass $H_1(z)$ aufgrund des Pols bei $z = 0$ **kausal** ist, es enthält also keine „Verfrühung“ z^{+1} .

Offensichtlich ist F_1 **nicht reellwertig**, da die Impulsantwort komplexwertig ist. Im Frequenzbereich kann man das erkennen, da es keine konjugiert-komplexe Nullstelle zu $z_{0,1}$ gibt. Reellwertig wäre F_1 nur für $\Omega_1 = 0$ oder $\Omega_1 = \pi$, da dann die Nst. auf der reellen Achse liegt und man die Systemfunktion $H_1(z) = 1 \pm r_1 z^{-1}$ erhält.

Das System hat nur eine einzige Nullstelle, die abhängig davon ob r_1 größer, kleiner oder gleich Null ist, innerhalb oder außerhalb des EKs bzw. auf dem EK liegt. Damit hängt es von r_1 ab, ob das Filter **maximalphasig** oder **minimalphasig** ist.

Da es zu der Nullstelle $z_{0,1}$ keine am EK gespiegelte Nst. gibt, ist das Filter **nur für $r = 1$ linearphasig**, da für diesen Spezialfall die Nst. auf dem EK liegt und $H_1(z) = 1 \pm e^{j\Omega_1} z^{-1}$. Im Zeitbereich kann man Linearphasigkeit erkennen, indem man untersucht, ob die Impulsantwort / Koeffizienten symmetrisch sind. Bei komplexen Koeffizienten muss gelten: $b_i = +b_{N-i}^*$ oder $b_i = -b_{N-i}^*$, wobei N die Ordnung des Filters ist.

Den *komplexen Frequenzgang* $H_1(e^{j\Omega})$ erhält man wie üblich durch Setzen von $z = e^{j\Omega}$:

$$H_1(e^{j\Omega}) = 1 - r_1 e^{j\Omega_1} e^{-j\Omega} = 1 - r_1 e^{j(\Omega_1 - \Omega)} = \underbrace{1 - r_1 \cos(\Omega_1 - \Omega)}_{\text{Realteil}} - r_1 \sin(\Omega_1 - \Omega) j \underbrace{-}_{\text{Imaginärteil}}$$

Den *Betragsgang* erhält man mit

$$\begin{aligned} |H_1(e^{j\Omega})| &= \sqrt{\Re\{H_1(e^{j\Omega})\}^2 + \Im\{H_1(e^{j\Omega})\}^2} \\ &= \sqrt{1 - 2r_1 \cos(\Omega_1 - \Omega) + r_1^2 \cos^2(\Omega_1 - \Omega) + r_1^2 \sin^2(\Omega_1 - \Omega)} \\ &= \sqrt{1 + r_1^2 - 2r_1 \cos(\Omega_1 - \Omega)} \end{aligned} \quad (\text{M4.1})$$

und den Phasengang (siehe auch (A.1) und Abb. A.1) mit

$$\angle H(e^{j\Omega}) = \begin{cases} \arctan \frac{\Im\{H(e^{j\Omega})\}}{\Re\{H(e^{j\Omega})\}} & \text{für } \Re\{H(e^{j\Omega})\} > 0 \\ \arctan \frac{\Im\{H(e^{j\Omega})\}}{\Re\{H(e^{j\Omega})\}} + \pi & \text{für } \Re\{H(e^{j\Omega})\} < 0 \end{cases}$$

Vorsicht: Die Funktion `atan(x)` in Python / Matlab ist die „normale“ Arcustangens-Funktion (keine eingebaute Fallunterscheidung); verwenden Sie zur Phasenberechnung in Python, Matlab, C, ... stattdessen `arctan2(z)` bzw. `atan2(z)` oder `angle(z)`. Die Fallunterscheidung ist notwendig, da der Wertebereich der Arcustangens-Funktion auf $\pm\pi/2$ beschränkt ist und den gleichen Wert liefert, unabhängig davon ob Real- und Imaginärteil beide positiv oder negativ sind.

Damit hängt auch zusammen, dass die Phase an einer Nullstelle von $|H_1(e^{j\Omega})|$ um π springt: Der Betrag von $H(e^{j\Omega})$ wird nur Null, wenn Real- *und* Imaginärteil zu Null werden und damit beide das Vorzeichen wechseln (Ausnahme: doppelte Nullstelle).

Achtung, nicht verwechseln: Nullstellen von $H(e^{j\Omega})$ liegen auf dem EK (physikalische Frequenzen), diese Frequenzen werden vom System vollständig unterdrückt. Nullstellen von $H(z)$ können sich irgendwo in der komplexen Ebene befinden.

Bei dieser Aufgabe gilt dementsprechend:

$$\angle H_1(e^{j\Omega}) = \begin{cases} \arctan \frac{-r_1 \sin(\Omega_1 - \Omega)}{1 - r_1 \cos(\Omega_1 - \Omega)} & \text{für } \Re\{H_1(e^{j\Omega})\} = 1 - r_1 \cos(\Omega_1 - \Omega) > 0 \\ \arctan \frac{-r_1 \sin(\Omega_1 - \Omega)}{1 - r_1 \cos(\Omega_1 - \Omega)} + \pi & \text{für } \Re\{H_1(e^{j\Omega})\} = 1 - r_1 \cos(\Omega_1 - \Omega) < 0 \end{cases} \quad (\text{M4.2})$$

- b) Verlauf von Betrags- und Phasengang von F_1 mit $\Omega_1 = \pi/3$ und $r_1 = 2$ (**maximalphasiges System**)

Die Systemfunktion und der komplexe Frequenzgang von F_1 lauten:

$$H_1(z) = 1 - 2e^{j\pi/3}z^{-1} = 1 - (1 + j\sqrt{3})z^{-1}$$

$$H_1(e^{j\Omega}) = 1 - 2e^{j(\pi/3-\Omega)} = 1 - 2\cos(\pi/3 - \Omega) - j2\sin(\pi/3 - \Omega)$$

Da die einzige Nullstelle außerhalb des Einheitskreises liegt, ist das System **maximalphasig**.

Für die Zeichnung werden zunächst mit $\cos \pi/3 = \frac{1}{2}$ und $\sin \pi/3 = \sqrt{3}/2$ einige Hilfswerte berechnet (Tab. M4.3):

Ω	$\Re\{H_1\}$	$\Im\{H_1\}$	$ H_1 $	$\angle H_1$
0	0	$-\sqrt{3}$	$\sqrt{3}$	-90°
$\pi/3$	-1	0	1	-180°
π	2	$\sqrt{3}$	$\sqrt{7}$	$41^\circ = -319^\circ$

Tab. M4.3.: Hilfswerte für Filter F_1 (Aufgabe 4.3b)

Den Betragsgangs berechnet man mit (M4.1) aus dem letzten Unterpunkt. Die Berechnung des Phasengangs mit (M4.2) ist ungünstig, da hier aufgrund $r_1 > 1$ der Realteil über Ω das Vorzeichen wechselt und eine Fallunterscheidung getroffen werden muss. Eine bessere Darstellung erhält man nach einer kleinen Umformung:

$$H_1(e^{j\Omega}) = 1 - r_1 e^{j(\Omega_1 - \Omega)} = e^{j(\Omega_1 - \Omega)} \left[e^{-j(\Omega_1 - \Omega)} - r_1 \right]$$

$$= e^{j(\Omega_1 - \Omega)} \left[\underbrace{\cos(\Omega_1 - \Omega) - r_1}_{\Re\{H_1\} < 0} - \underbrace{j\sin(\Omega_1 - \Omega)}_{\Im\{H_1\}} \right]$$

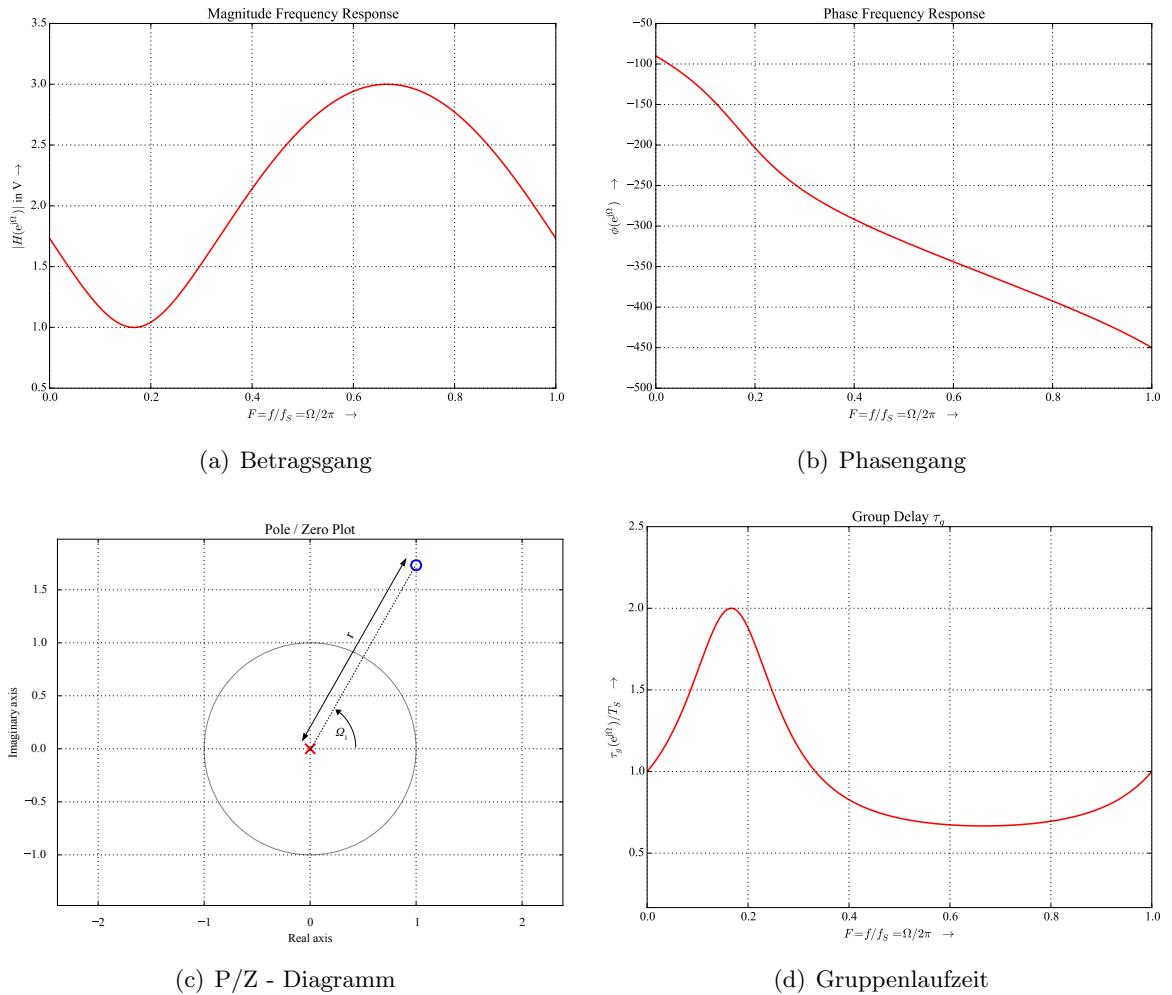
$$\Rightarrow \angle H_1(e^{j\Omega}) = \Omega_1 - \Omega + \arctan \frac{\sin(\Omega_1 - \Omega)}{r_1 - \cos(\Omega_1 - \Omega)} + \pi \quad (\text{M4.3})$$

In den Plots in Abb. M4.4 sieht man Folgendes:

- Bei der Phase kann man nicht zwischen $+41^\circ$ und -319° unterscheiden. Da die Phase immer weiter abfällt (aufgrund der Verzögerung des Filters), ist es sinnvoll sie auch so darzustellen (Option `unwrapped` in Python / Matlab) anstatt sie immer umzubrechen, sobald $\pm 180^\circ \hat{=} \pm \pi$ erreicht sind.
- Die Gruppenlaufzeit erreicht ihr Maximum von $2T_S$ bei $F \approx 1/6$ bzw. $\Omega = \pi/3$. Wird der Filter z.B. innerhalb einer Regelschleife betrieben, kann eine zu hohe Gruppenlaufzeit / Phasendrehung das System destabilisieren.

c) Filter F_2 mit am EK gespiegelter Nullstelle (**minimalphasiges System**)

Die Spiegelung eines Punktes am EK wird mathematisch beschrieben durch $\tilde{z}_1 = 1/z_1^*$. Beim Spiegeln eines Punktes am EK bleibt der Winkel unverändert, $\Omega_2 = \Omega_1$, der neue Radius ist der Kehrwert des alten, $r_2 = 1/r_1$. Damit liegt die Nullstelle von F_2 bei $z_{0,2} = r_2 e^{j\Omega_2} = r_1^{-1} e^{j\Omega_1}$.

Abb. M4.4.: Plots zu maximalphasigem Filter F_1 (Aufgabe 4.3b)

F_2 ist damit bis auf den Skalierungsfaktor c bestimmt:

$$\begin{aligned}
 |H_2(\Omega = \Omega_1)| &= \left| c \left(1 - r_1^{-1} e^{j(\Omega_1 - \Omega)} \right) \right| \stackrel{!}{=} \left| 1 - r_1 e^{j(\Omega_1 - \Omega)} \right| = |H_1(\Omega = \Omega_1)| \text{ mit } r_1 > 1 \\
 &\Rightarrow |c| \left| 1 - r_1^{-1} \right| \stackrel{!}{=} |r_1 - 1| \\
 &\Rightarrow |c| = \left| \frac{r_1 - 1}{1 - r_1^{-1}} \right| = \left| \frac{r_1 - 1}{r_1^{-1}(r_1 - 1)} \right| = r_1 \Rightarrow c = \pm r_1
 \end{aligned}$$

Wählt man $c = +r_1$, erhält man

$$\begin{aligned}
 H_2(z) &= r_1 \left(1 - r_1^{-1} e^{j\Omega_1} z^{-1} \right) = r_1 - e^{j\Omega_1} z^{-1} \\
 \Rightarrow H_2(e^{j\Omega}) &= r_1 - e^{j(\Omega_1 - \Omega)} = r_1 - \cos(\Omega_1 - \Omega) - j \sin(\Omega_1 - \Omega) \\
 \Rightarrow |H_2(e^{j\Omega})| &= \sqrt{1 + r_1^2 - 2r_1 \cos(\Omega_1 - \Omega)}
 \end{aligned}$$

Der Betragsgang von F_2 ist für alle Frequenzen gleich wie der von F_1 (vergleichen Sie mit Aufgabe Punkt a).

Wegen $r_1 > 1 \Rightarrow r_2 < 1$ liegt die einzige Nullstelle innerhalb des EK, F_2 ist **minimalphasig**. Aus M4.2 sieht man, dass der Realteil von H_2 immer positiv ist und die

Fallunterscheidung hier nicht notwendig ist:

$$\angle H_2(e^{j\Omega}) = \arctan \frac{-r_2 \sin(\Omega_1 - \Omega)}{1 - r_2 \cos(\Omega_1 - \Omega)} = -\arctan \frac{\sin(\Omega_1 - \Omega)}{r_1 - \cos(\Omega_1 - \Omega)} \text{ mit } r_2 = 1/r_1$$

d) Verlauf von Betrags- und Phasengang von F_2 mit $\Omega_1 = \pi/3$ und $r_2 = 1/r_1 = 0,5$

Die Systemfunktion und der komplexe Frequenzgang von F_2 lauten (Skalierungsfaktor c nicht vergessen):

$$H_2(z) = 2 - e^{j\pi/3}z^{-1} = 2 - \frac{1}{2}(1 + j\sqrt{3})z^{-1}$$

$$H_2(e^{j\Omega}) = 2 - \frac{1}{2}\cos(\pi/3 - \Omega) - \frac{j}{2}\sin(\pi/3 - \Omega)$$

Auch hier werden zunächst ein paar Hilfswerte mit $\cos \pi/3 = \frac{1}{2}$ und $\sin \pi/3 = \sqrt{3}/2$ berechnet:

Ω	$\Re\{H_2\}$	$\Im\{H_2\}$	$ H_2 $	$\angle H_2$
0	3/2	$-\sqrt{3}/2$	$\sqrt{3}$	-30°
$\pi/3$	1	0	1	0°
π	5/2	$\sqrt{3}/2$	$\sqrt{7}$	19,1°

Tab. M4.4.: Hilfswerte für Filter F_2 (Aufgabe 4.3c)

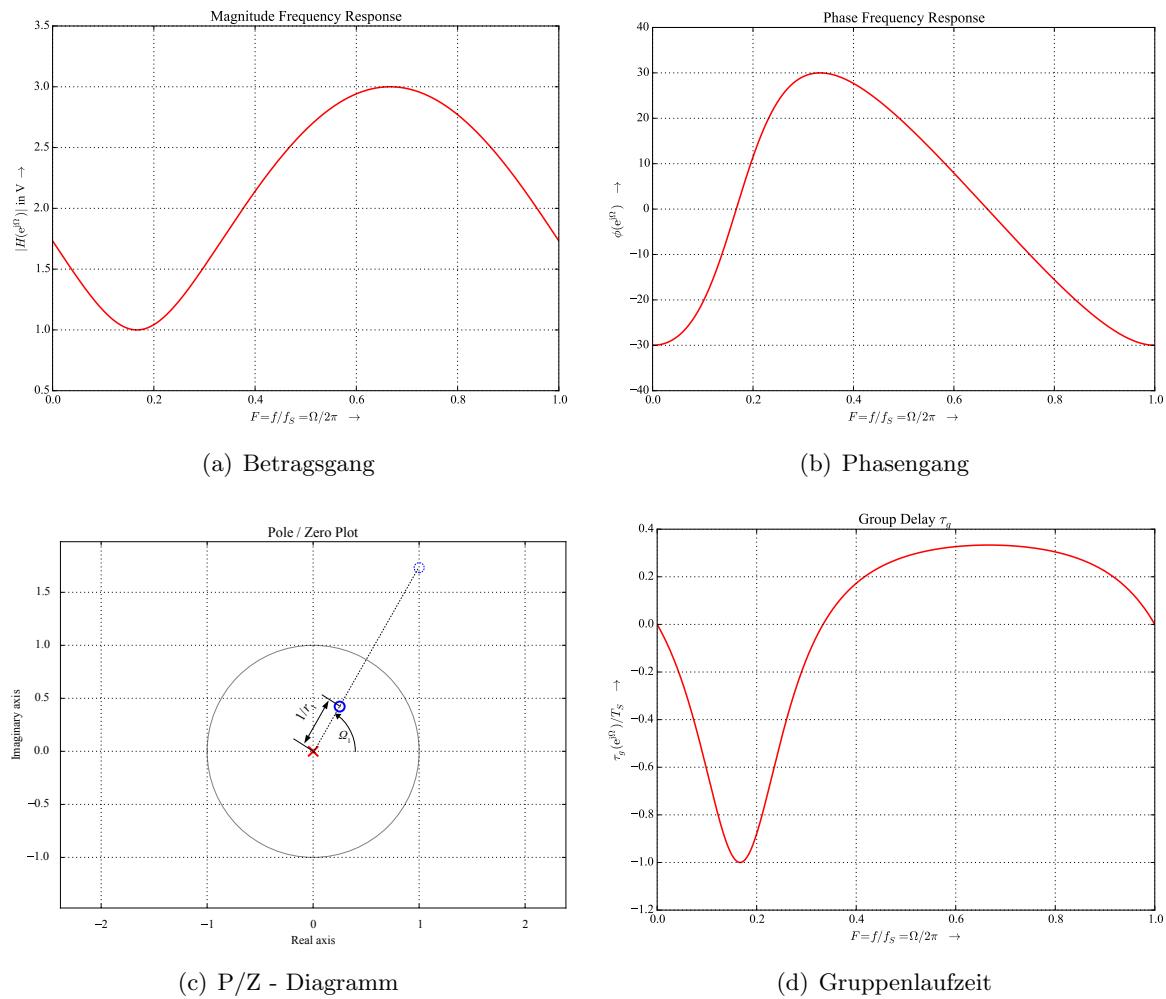
In den Plots in Abb. M4.5 sieht man Folgendes:

- Der Betragsgang ist der gleiche wie der von Filter F_1 (wie schon im vorigen Unterpunkt bewiesen).
- Die Phasendrehung und deren Variation sind deutlich geringer als bei F_1 (*minimalphasiges* System).
- Daher ist die Gruppenlaufzeit auch deutlich geringer (max. $0,3 T_S$).

e) Linearphasigkeit des zusammengesetzten Filters F

Wir stellen die Systemfunktion von $H(z) = H_1(z)H_2(z)$ auf und versuchen, einen Vorfaktor so auszuklammern, dass ein symmetrisches Polynom („Spiegelpolynom“) übrigbleibt:

$$\begin{aligned} H(z) &= H_1(z)H_2(z) = (1 - r_1 e^{j\Omega_1} z^{-1})(r_1 + e^{j\Omega_1} z^{-1}) \\ &= r_1 - (r_1^2 + 1)e^{j\Omega_1} z^{-1} + r_1 e^{j2\Omega_1} z^{-2} \\ &= r_1 e^{j\Omega_1} z^{-1} \left[\underbrace{e^{-j\Omega_1}}_{\tilde{b}_0} z^{+1} - \underbrace{\left(r_1 + \frac{1}{r_1}\right)}_{\tilde{b}_1} \underbrace{+ e^{j\Omega_1}}_{\tilde{b}_2} z^{-1} \right] \end{aligned}$$

Abb. M4.5.: Plots zu minimalphasigem Filter F_2 (Aufgabe 4.3d)

Diese Zerlegung gelingt, $\tilde{b}_0 = \tilde{b}_2^*$, daher kann man das Spiegelpolynom einfach darstellen als Summe aus reellwertigen Sinus¹- und Cosinustermen:

$$\begin{aligned}
 H(e^{j\Omega}) &= r_1 e^{j(\Omega_1 - \Omega)} \left[e^{-j(\Omega_1 - \Omega)} - \left(r_1 + \frac{1}{r_1} \right) + e^{j(\Omega_1 - \Omega)} \right] \\
 &= r_1 e^{j(\Omega_1 - \Omega)} \underbrace{\left[2 \cos(\Omega_1 - \Omega) - \underbrace{\left(r_1 + \frac{1}{r_1} \right)}_{\text{immer } > 2} \right]}_{\text{immer } < 0} \\
 &= -r_1 e^{j(\Omega_1 - \Omega)} \left| 2 \cos(\Omega_1 - \Omega) - \left(r_1 + \frac{1}{r_1} \right) \right|
 \end{aligned}$$

(Den Beweis für $|r + r^{-1}| \geq 2$ finden Sie am Ende der Aufgabe.) Damit lässt sich die Phase einfach ablesen (nicht das negative Vorzeichen vergessen!) als

$$\angle H = \varphi_H = \underbrace{\Omega_1 + \pi}_{\text{konstant}} \underbrace{- \Omega}_{\text{linear}},$$

¹Hier = 0.

man sieht sofort dass die Phase linear ist. Die Gruppenlaufzeit ist konstant:

$$\tau_{g,H} = -\frac{\partial \varphi_H}{\partial \omega} = -\frac{\partial \varphi_H}{\partial \Omega} \frac{\partial \Omega}{\partial \omega} = 1 \cdot \frac{1}{f_S} = T_S \text{ mit } \Omega = 2\pi \frac{f}{f_S}$$

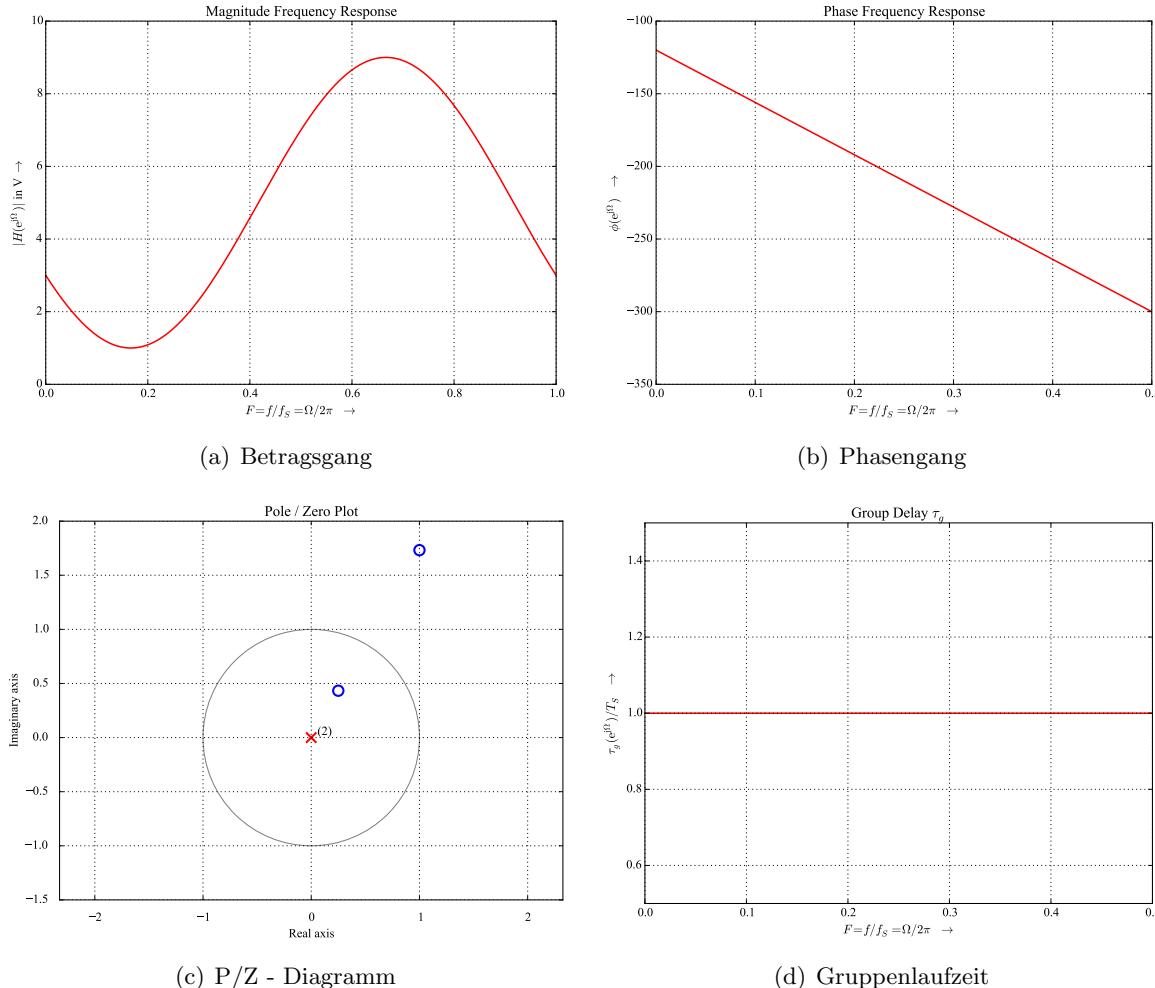


Abb. M4.6.: Plots zu linearphasigem Filter F (Aufgabe 4.3e)

Der Betragsgang lässt ebenfalls leicht aus der obigen Form von $H(e^{j\Omega})$ berechnen, da ja bereits Amplituden- und Phasengang getrennt worden und der Amplitudengang nie das Vorzeichen wechselt:

$$\begin{aligned} |H(e^{j\Omega})| &= r_1 \left| 2 \cos(\Omega_1 - \Omega) - \left(r_1 - \frac{1}{r_1} \right) \right| = (r_1^2 - 1) - 2 \cos(\Omega_1 - \Omega) \\ &= |H_1(e^{j\Omega})| |H_2(e^{j\Omega})| = |H_1(e^{j\Omega})|^2 \end{aligned}$$

Mit den Zahlenwerten aus b) $r_1 = 2$ und $\Omega_1 = \pi/3$ und $\cos \pi/3 = \frac{1}{2}$, $\cos 2\pi/3 = -\frac{1}{2}$ und $\sin \pi/3 = \sin 2\pi/3 = \sqrt{3}/2$ erhält man:

$$H(z) = 2 - 5e^{j\pi/3}z^{-1} + 2e^{j2\pi/3}z^{-2} = 2 - \frac{5}{2} \left(1 + j\sqrt{3} \right) z^{-1} + \left(-1 + j\sqrt{3} \right) z^{-2}$$

Zusammenfassend gilt für das linearphasige Filter als Kombination der beiden Teilfilter:

- Der Betragsgang ist das Produkt der beiden Teilbetragsgänge,
 $|H(e^{j\Omega})| = |H_1(e^{j\Omega})| |H_2(e^{j\Omega})| = |H_1(e^{j\Omega})|^2$.
- Die Gesamtphase ist die Summe der Teilphasen,
 $\angle H(e^{j\Omega}) = \angle H_1(e^{j\Omega}) + \angle H_2(e^{j\Omega}) = \Omega_1 + \pi - \Omega$.
- Die Phase fällt linear ab, die Gruppenlaufzeit ist konstant. Würde man die Gruppenlaufzeit dieses linearphasigen mit den entsprechenden minimal- und maximalphasigen Filtern zweiter Ordnung vergleichen (doppelter Pol innerhalb bzw. außerhalb des EK), würde man sehen, dass gilt $\tau_{g,min} < \tau_{g,lin} < \tau_{g,max}$.

f) Implementierung für Filter F

Für komplexwertige Systeme kann man einen Signalflossgraphen genau wie für reellwertige Systeme zeichnen; in Abb. M4.7a ist das Eingangssignal reellwertig; die Koeffizienten, das Ausgangssignal und ein Teil der Signale (fette Linien) komplexwertig.

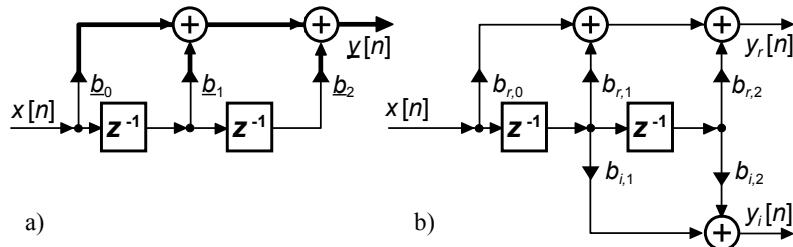


Abb. M4.7.: Blockschaltbild (a) und Implementierung (b) für Filter F (Aufgabe 4.3e)

Die Koeffizienten lauten:

$$\begin{aligned}\underline{b}_0 &= r \\ \underline{b}_1 &= -(r^2 + 1)e^{j\Omega_1} \\ \underline{b}_2 &= re^{j2\Omega_1}\end{aligned}$$

Bei einer Hardwareimplementierung werden Real- und Imaginärteil meist separat verarbeitet (Ausnahme: Floating-Point Unit mit komplexen MAC-Units), dann benötigt man für Real- und Imaginärteil separate Signalfäde, Koeffizienten, Multiplizierer und Addierer. Das Gesamt-Ausgangssignal ist dann $\underline{y}[n] = \underline{y}_r[n] + j\underline{y}_i[n]$. Die Koeffizienten sind hier:

$$\begin{array}{ll} b_0, r = \Re\{b_0\} = r & b_0, i = \Im\{b_0\} = 0 \\ b_1, r = \Re\{b_1\} = -(r^2 + 1) \cos \Omega_1 & b_1, i = \Im\{b_1\} = -(r^2 + 1) \sin \Omega_1 \\ b_2, r = \Re\{b_2\} = r \cos 2\Omega_1 & b_2, i = \Im\{b_2\} = r \sin 2\Omega_1 \end{array}$$

Beweis dass $r + r^{-1} \geq 2$:

$$\begin{aligned}(r - 1)^2 &\geq 0 \quad \text{für } r > 1 \\ \Rightarrow r^2 - 2r + 1 &\geq 0 \\ \Rightarrow r^2 + 1 &\geq 2r \quad | \div r \\ \Rightarrow r + r^{-1} &\geq 2\end{aligned}$$

M4.4. * FIR Halbbandfilter → Aufgabe 4.4

a) Spezifikationen:

Die Bandmitte liegt bei $f_S/4 = 6 \text{ kHz}$. Für ein Halbbandfilter muss gelten $f_S/4 - f_{DB}^! = f_{SB} - f_S/4$ oder $f_{DB}^! = f_S/2 - f_{SB}$. Um diese Bedingung zu erfüllen, muss das Durchlassband auf $f_{DB} = 2 \text{ kHz}$ erweitert werden.

Der vorgegebene Ripple im DB beträgt $A_{DB} = 0,1 \text{ dB}$ oder $\delta_{DB} = 10^{A_{DB}/40} - 1 = 5,7 \cdot 10^{-3}$, der Ripple im SB beträgt $A_{SB} = 60 \text{ dB}$ oder $\delta_{SB} = 10^{-A_{SB}/20} = 10^{-3}$.

Um ein Halbbandfilter zu erhalten, müssen der Verlauf im DB und SB symmetrisch zu einander sein. Daher muss der Ripple im DB angepasst werden auf $\delta_{DB} = \delta_{SB} = 10^{-3}$ oder $A_{DB} = 0,017 \text{ dB}$.

b) Frequenzgang aus Impulsantwort:

Aus der Impulsantwort lässt sich leicht die Systemfunktion $H(z)$ ableiten und durch Ausklammern von z^{-3} in eine Form mit symmetrischen Koeffizienten und Exponenten bringen. Lässt man den Term z^{-3} weg, erhält man die *akausale* Systemfunktion $H_{ak}(z)$. Für diese Systemfunktion gibt es zwar keine Hardwarerealisierung, da die Impulsantwort symmetrisch zu $n = 0$ ist und somit auch Anteile hat bei $n < 0$. Der *Betragsgang* von $H_{ak}(z)$ ist aber identisch mit dem von $H(z)$, da z^{-3} als Verzögerung lediglich den Phasengang beeinflusst.

$$\begin{aligned}
 h[n] &= \{0,1; 0; 0,2; 0,3; 0,2; 0; 0,1\} \\
 \circlearrowleft \bullet \quad H(z) &= 0,1 + 0,2z^{-2} + 0,3z^{-3} + 0,2z^{-4} + 0,1z^{-6} \\
 &= z^{-3} (0,1z^3 + 0,2z + 0,3 + 0,2z^{-1} + 0,1z^{-3}) \\
 \Rightarrow H_{ak}(z) &= 0,1z^3 + 0,2z + 0,3 + 0,2z^{-1} + 0,1z^{-3} \\
 \Rightarrow H_{ak}(z = e^{j\Omega}) &= 0,1e^{j3\Omega} + 0,2e^{j\Omega} + 0,3 + 0,2e^{-j\Omega} + 0,1e^{-j3\Omega} \\
 &= 0,4 \cos \Omega + 0,2 \cos 3\Omega + 0,3
 \end{aligned}$$

Beide Cosinusfunktionen nehmen bei $\Omega = 0$ den Wert 1, bei $\Omega = \pi/2$ den Wert 0 und bei $\Omega = \pi$ den Wert -1 an (multipliziert mit dem jeweiligen Koeffizienten). Beide Cosinus-Funktionen sind punktsymmetrisch zu $(\Omega = \pi/2, 0)$ und werden um den DC-Wert „angehoben“, der durch den mittleren Koeffizienten 0,3 gegeben ist. Es ist also automatisch die Halbbandbedingung erfüllt mit $H(\pi/2 - \Omega) + H(\pi/2 + \Omega) = 2H(\pi/2) = 0,6$.

Es ist also:

- $H_{ak}(\Omega = 0) = H_{ak}(z = 1) = \sum_i h_i$ (gilt immer)
- $H_{ak}(\Omega = \pi/2) = h_{N/2}$, da alle Cosinusfunktionen, aus denen sich der acausal Amplitudengang zusammensetzt, hier Null werden, so dass nur der mittlere Koeffizient übrigbleibt.
- $H_{ak}(\Omega = \pi) = H_{ak}(z = -1) = - \sum_i h_i + 2h_{N/2}$, da alle geraden Koeffizienten Null sind außer dem mittleren.

Insgesamt sind also $H(\Omega = 0) = 0,2 + 0,4 + 0,3 = 0,9$, $H(\Omega = \pi/2) = 0,3$ und $H(\Omega = \pi) = -0,3$

Mit der Zusatzbedingung $H(\Omega = \pi) \stackrel{!}{=} 0$ lässt sich ein verbessertes Tiefpassverhalten erreichen, da dann die Frequenzkomponente bei $f_S/2$ komplett unterdrückt wird. Das erreicht man am Einfachsten durch Anpassung des mittleren Koeffizienten:

$$H(\Omega = 0) + \underbrace{H(\Omega = \pi)}_{\stackrel{!}{=} 0} = 2H(\pi/2) \Rightarrow \sum_i h_i = 2h_{N/2}$$

Die Summe aller Koeffizienten *ohne* $h_{N/2}$ muss also gleich $h_{N/2}$ sein, hier im Beispiel $2h_0 + 2h_2 = 0,6 \stackrel{!}{=} h_3$

Allgemein:

$$\begin{aligned} h[n] &= \{h_0; 0; h_2; 0; \dots; 0; h_{N/2-1}; h_{N/2}; h_{N/2+1}; 0; \dots; 0; h_{N-2}; 0; h_N\} \\ &= \{h_0; 0; h_2; 0; \dots; 0; h_{N/2-1}; h_{N/2}; h_{N/2-1}; 0; \dots; 0; h_2; 0; h_0\} \\ \circlearrowleft \bullet \quad H_{ak}(z) &= h_{N/2} + \sum_{i=0}^{(N-2)/4} h_{2i} \left(z^{2i-N/2} + z^{N/2-2i} \right) \\ \Rightarrow H_{ak}(e^{j\Omega}) &= h_{N/2} + 2 \sum_{i=0}^{(N-2)/4} h_{2i} \cos(2i\Omega) \end{aligned}$$

c) **Skalierung der Koeffizienten so, dass $|\tilde{H}(\Omega = \pi/2)| = 0,5$**

Wie in der vorigen Aufgabe gezeigt, ist

$$\tilde{H}_{ak}(\Omega = \pi/2) = \tilde{h}_{N/2} = 0,5$$

Die Größen mit Tilde gehören zu der Zielübertragungsfunktion. Damit der Zielwert $|\tilde{H}(\Omega = \pi/2)| = 0,5$ erreicht wird, ohne den Verlauf des Frequenzgangs zu ändern, müssen alle Koeffizienten skaliert werden (gerundet auf drei Stellen):

$$\begin{aligned} \tilde{h}_i &= \frac{\tilde{h}_{N/2}}{h_{N/2}} h_i = \frac{0,5}{0,3} h_i \\ &= \{0,167; 0; 0,333; 0,5; 0,333; 0; 0,167\} \\ \circlearrowleft \bullet \quad \tilde{H}(z) &= 0,167 + 0,333z^{-2} + 0,5z^{-3} + 0,333z^{-4} + 0,167z^{-6} \end{aligned}$$

- d) Wenn der **mittlere Koeffizient Null ist**, entfällt die DC-Komponente aus Aufgabepunkt b), damit muss $H(\Omega = 0) + H(\Omega = \pi) = 2H(\Omega = \pi/2) = 0$ sein und es gilt $H(\Omega = 0) = -H(\Omega = \pi)$. Solche Filter haben einen Betragsgang, der symmetrisch zur Nullstelle bei $H(\Omega = \pi/2)$ sind, sie sind also Bandsperren oder Notchfilter (siehe auch Aufgabe 4.6b und Abb. M4.8).
- e) Dass jeder zweite Koeffizient (außer dem mittleren) eines Halbbandfilters immer Null ist, lässt sich mit Hilfe der inversen DTFT zeigen: Die Transformationsformel in der Aufgabenstellung mit $\Omega_g = \pi/2$

$$X_{HB,id}(\Omega) = \begin{cases} 1 & \text{für } 0 \leq |\Omega| < \pi/2 \\ 0 & \text{sonst} \end{cases} \quad \bullet \circlearrowleft \circlearrowright \quad x[n] = \frac{1}{2} \frac{\sin(n\pi/2)}{n\pi/2} \quad \text{für } n = 0, 1, 2, \dots$$

ergibt Null für alle geraden Werte von n . Vom idealen kommt man zu einem realen TP, indem man die unendlich lange Impulsantwort (= die Koeffizienten $h_{HB,id}$) mit einer

geeigneten Fensterfunktion $w[n]$ multipliziert. Die Koeffizienten, die Null sind, bleiben bei dieser Operation Null. Wenn die Fensterfunktion symmetrisch ist, bleibt auch die Symmetrie der Koeffizienten und damit die Halbbandbedingungen erhalten!

- f) Einen **Hochpass-Halbbandfilter** erhält man mit der Transformation $z \rightarrow -z$:

$$\begin{aligned} H_{ak,TP}(z) &= 0,1z^3 + 0,2z + 0,6 + 0,2z^{-1} + 0,1z^{-3} \\ \Rightarrow H_{ak,HP}(z) &= -0,1z^3 - 0,2z + 0,6 - 0,2z^{-1} - 0,1z^{-3} \end{aligned}$$

Beim Halbbandfilter ist die Transformation besonders einfach durchzuführen, da es nur Terme mit ungeraden Exponenten gibt (außer dem mittleren) und daher alle Terme (außer dem mittleren : -) das Vorzeichen wechseln.

- g) **IIR-Halbbandfilter** können nicht linearphasig sein, da sie Pole außerhalb des Ursprungs haben. Es gibt dennoch IIR-Halbbandfilter, allerdings ist hier die Halbbandbedingung anders definiert: Hier muss die *quadrierte* Betragsfunktion symmetrisch zu $\Omega = \pi/2$ verlaufen.

Nicht alle Verfahren eignen sich zum Entwurf: Da Pass- und Stopband symmetrisch zu einander sein müssen, können z.B. keine Chebychev-Approximationen verwendet werden (Ripple im Pass- oder Stopband). Meist werden elliptische Filter verwendet, der resultierende Pol-Nullstellenplan hat Pole auf der imaginären Achse und Nullstellen entlang des Einheitskreises im Sperrband (bei den dazu symmetrischen Frequenzen im Durchlassband ist der Betragsgang 1).

Auch bei IIR Halbbandfiltern ist jeder zweite Koeffizient Null, so dass auch hier knapp die Hälfte der Multiplikationen eingespart werden kann.

M4.5. Filterentwurf → Aufgabe 4.5

Entwurf eines Kaiser Window und eines Equiripple FIR-Filters mit folgenden Spezifikationen:

$$F_{DB} = 0,05, F_{SB} = 0,025, A_{DB} \leq 0,1 \text{ dB}, A_{SB} \geq 50 \text{ dB}$$

Den Spezifikationen entnimmt man, dass es sich um ein Hochpassfilter handelt.

M4.6. * Filtertransformationen → Aufgabe 4.6

Die Systemfunktion $H_1(z)$ erhält man durch z -Transformation von $h_1[n]$:

$$\begin{aligned} h_1[n] &= \{1; 7/4; 1/2; -1/4\} \Rightarrow H_1(z) = 1 + 7/4z^{-1} + 1/2z^{-2} - 1/4z^{-3} \\ &= z^{-3} (z^3 + 7/4z^2 + 1/2z - 1/4) \end{aligned}$$

- a) **Nullstellen von $H_1(z)$**

Eine weitere Nullstelle muss vorliegen, da das Filter dritter Ordnung ist; der Verlauf von $H_1(e^{j2\pi F})$ lässt vermuten dass diese sich bei $F = 0,5$ bzw. $f = f_S/2$ befindet.

Polynomdivision durch $(z + 1)^2 = z^2 + 2z + 1$ ergibt:

$$\begin{array}{r}
 \left(\begin{array}{c} z^3 + \frac{7}{4}z^2 + \frac{1}{2}z - \frac{1}{4} \\ -z^3 - 2z^2 - z \end{array} \right) \div (z^2 + 2z + 1) = z - \frac{1}{4} \\
 \hline
 \begin{array}{c} -\frac{1}{4}z^2 - \frac{1}{2}z - \frac{1}{4} \\ \frac{1}{4}z^2 + \frac{1}{2}z + \frac{1}{4} \end{array} \\
 \hline
 0
 \end{array}$$

Die Polynomdivision zeigt, dass bei $z = -1$ eine doppelte Nullstelle vorliegt und bei $z = +1/4$ eine weitere einfache Nullstelle.

Beweis, dass *eine* (oder mehrere) Nullstellen bei $f_S/2$ vorliegen durch Einsetzen:
 $H_1(f = f_S/2) = H_1(z = -1) = -1 + 7/4 - 1/2 - 1/4 = 0$

$$\begin{aligned}
 H_1(f = 0) &= H_1(z = 1) = 1 + 7/4 + 1/2 - 1/4 = 3 \\
 H_1(f = f_S/2) &= H_1(z = -1) = 1 - 7/4 + 1/2 + 1/4 = 0
 \end{aligned}$$

b) Filter mit jeweils zwei Verzögerungsgliedern

$$h_2[n] = \{1; 0; 7/4; 0; 1/2; 0; -1/4\} \quad \circlearrowleft \quad H_2(z) = 1 + 7/4z^{-2} + 1/2z^{-4} - 1/4z^{-6}$$

Zusätzliche Verzögerungsglieder stauchen den Frequenzgang zusammen. Die Maxima finden sich jetzt bei $f = 0$ und $f = f_S/2$; das Minimum liegt jetzt bei $f_S/4$ (siehe Abb. M4.8) (nicht gefragt).

c) TP-HP-Transformation

durch $H_3(z) = H_1(-z)$ ergibt $H_3(z) = 1 - 7/4z^{-1} + 1/2z^{-2} + 1/4z^{-3}$. Durch die Transformation wird der Frequenzgang $|H_3(f)|$ gegenüber $|H_1(f)|$ um $f_S/2$ verschoben, das Maximum liegt daher jetzt bei $f_S/2$, das Minimum bei $f = 0$ (siehe Abb. M4.8):

$$\begin{aligned}
 H_3(f = 0) &= H_3(z = 1) = 1 - 7/4 + 1/2 + 1/4 = 0 \\
 H_3(f = f_S/2) &= H_3(z = -1) = 1 + 7/4 + 1/2 - 1/4 = 3
 \end{aligned}$$

$$h_3[n] = \{1; -7/4; +1/2; +1/4\}$$

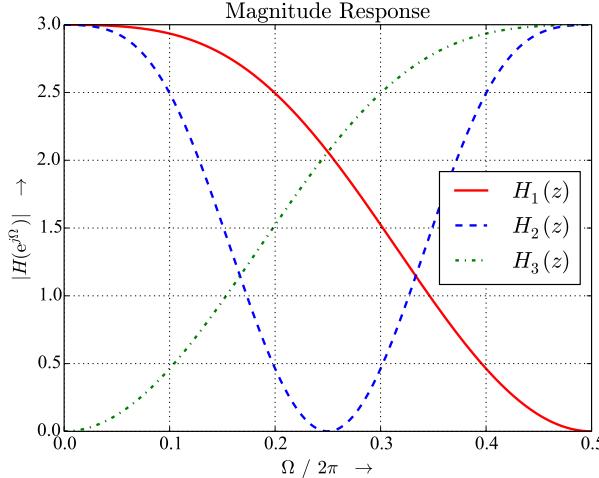


Abb. M4.8.: Betragsfrequenzgänge zu Aufgabe 4.6

M4.7. Filterimplementierung auf FPGAs → Aufgabe 4.7

Ein FIR-Filter 20. Ordnung benötigt im allgemeinen Fall (d.h. es kann keine Symmetrie ausgenutzt werden, um Multiplikationen einzusparen) $N_{FIR} = 21$ Koeffizientenmultiplizierer, die hier mit MAC-Cores implementiert werden sollen.

a) Maximaler Durchsatz mit einem MAC-Core:

Wenn der MAC-Core mit maximaler Frequenz getaktet wird, kann er $f_{MAC} = 2 \cdot 10^8$ Samples je Sekunde mit einem Koeffizientenwert multiplizieren. In einem allgemeinen FIR-Filter 20. Ordnung müssen pro Sample $N_{FIR} = 21$ Multiplikationen durchgeführt werden. Damit können maximal

$$R_a = f_{MAC}/N_{FIR} = 2 \cdot 10^8 \text{S/s}/21 = 9,52 \text{ MS/s}$$

vom Filter verarbeitet werden.

b) Maximaler Durchsatz bei linearphasigen FIR-Filter:

Beim linearphasigen Filter sind die Koeffizientenwerte symmetrisch (gerade oder ungerade) zum mittleren Koeffizienten, die zugehörigen Samples können daher vor der Multiplikation paarweise durch Addition (gerade Symmetrie) oder Subtraktion (ungerade Symmetrie) zusammengefasst werden. In diesem Fall bleiben $N_{FIR,lin} = 11$ Multiplikationen übrig, damit wird ein Durchsatz erzielt von

$$R_b = f_{MAC}/N_{FIR,lin} = 2 \cdot 10^8 \text{S/s}/11 = 18,18 \text{ MS/s.}$$

Bei einem Halbbandfilter ist zusätzlich jeder zweite Koeffizient außer dem mittleren Null, damit können 5 weitere Multiplikationen eingespart werden (s. Aufgabe 5.3) und es werden nur noch $N_{FIR,HB} = 6$ Multiplikationen je Sample benötigt:

$$R_{b,HB} = f_{MAC}/N_{FIR,HB} = 2 \cdot 10^8 \text{S/s}/6 = 33,33 \text{ MS/s.}$$

c) Maximaler Durchsatz mit zwei MAC-Cores:

Die Multiplikationen je Eingangssample können hier zwischen beiden Multiplizierern aufgeteilt werden, der eine übernimmt 5 und der andere 6 Multiplikationen je Eingangssample und begrenzt damit den Durchsatz auf

$$R_c = f_{MAC}/\lceil N_{FIR,lin}/2 \rceil = 2 \cdot 10^8 \text{S/s}/6 = 33,33 \text{ MS/s.}$$

M4.8. Verständnisfragen zu Filtern → Aufgabe 4.8

Aussage	FIR-Filter		IIR-Filter	
	ja	nein	ja	nein
Rekursiv:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Endliche Impulsantwort:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Instabil:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Linearphasig: (1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Minimalphasig: (1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Minimalphasig und linearphasig: (1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Pole außerhalb des Ursprungs:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Einheitskreises:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nullstellen außerhalb des Ursprungs:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nullstellen außerhalb des Einheitskreises:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Halbbandfilter: (2)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Konstante Gruppenlaufzeit:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Transponieren möglich: (3)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Entwurf über bilineare Transformation:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Transversale Struktur:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Abgeleitet von analogen Butterworth-Filters:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Tab. M4.5.: Musterlösung zu Aufgabe 4.8

Erklärungen zu Tab. M4.5

- (1) Linearphasige Filter setzen voraus, dass Nullstellenpaare (oder auch Polstellenpaare) symmetrisch zum Einheitskreis liegen, d.h. Radien von r_1 und $1/r_1$ aufweisen. Bei Polstellenpaaren würde immer eine Polstelle außerhalb des EK liegen, das resultierende Filter wäre instabil. Daher gibt es keine linearphasigen IIR-Filter.
- (2) Halbbandfilter können sowohl mit FIR- als auch mit IIR-Strukturen realisiert werden, allerdings sind letztere nicht linearphasig (siehe auch Aufgabe 4.4). Bei minimalphasigen Filtern müssen alle Null- und Polstellen innerhalb des EK liegen, daher gibt es keine Filter, die gleichzeitig linearphasig und minimalphasig sind.
- (3) Beliebige (lineare) Strukturen können transponiert werden.

M5. FIX: Wortlängeneffekte und Fixpoint-Systeme im Zeitbereich

M5.1. * Analog-Digital-Wandlung eines Drucksensors → Aufgabe 5.1

- a) Die **Auflösung des Wandlers** beträgt $q_U = FSR/2^8 = 5 \text{ V}/256 = 19,53 \text{ mV}$; das ist auch die Unsicherheit, die durch die Quantisierung hervorgerufen wird.
- b) Die **Auflösung des Wandlers bezogen auf die Messgröße** ergibt sich aus

$$q_a = q_U / S_{Sens} = 19,5 \text{ mV}/400 \text{ mV/g} = 5/(256 \cdot 0,4) \text{ g} = 48,83 \cdot 10^{-3} \text{ g}.$$

- c) **Digitale Nullpunktkorrektur**

Bei $a = 0$ liefert der Sensor 1,5 V, die vom ADC in den Code $\tilde{c}_0 = \text{round}(U_0/q_U) = 77$ gewandelt werden. Durch Subtraktion dieses Wertes wird der Beschleunigung $a = 0$ der Digitalwert $c_0 = 0$ zugeordnet und so der analoge Offset von 1,5 V im Digitalen kompensiert.

- d) Den **Digitalwert bei $a = \pm 1,2 \text{ g}$** erhält man jetzt - da der Offset kompensiert ist - einfach durch

$$c(a = \pm 1,2 \text{ g}) = \lfloor a S_{Sens}/q_U \rfloor = \lfloor \pm 1,2 \text{ g} \cdot 400 \text{ mV/g} / 19,53 \text{ mV} \rfloor = \pm 24$$

Binär geschrieben erhält man $+24 \hat{=} 0001\ 1000$ und $-24 \hat{=} 1110\ 1000$ (ZK: Bits invertieren und 1 LSB addieren). Für das nZKF-Format stellt man sich den Fraktionalpunkt nach dem MSB vor bei $0,001\ 1000$ bzw. $1,110\ 1000$, entsprechend einem Links-Shift um $WF = 7$ Stellen. Ein LSB entspricht dann $2^{-WF} = 2^{-7} = 1/128$, im Dezimalsystem repräsentiert die Darstellung die Werte $\pm 24/128 = \pm 0,1875$.

Man kann jetzt einen Skalierungsfaktor zwischen der Sensorgröße a und der digitalen Repräsentation in nZKF Format bestimmen, um einfacher zwischen beiden Welten umrechnen zu können:

$$\frac{x_{nZKF}}{a} = \frac{2^{-WF} a / q_a}{a} = \frac{1}{2^{WF} q_a} = 0,16 \text{ g}^{-1}$$

- e) Die **maximale Eingangsamplitude** des Wandlers muss so gewählt werden, dass weder die minimale noch die maximal darstellbaren Zahlen überschritten werden. Aussteuergrenze des ADCs überschritten wird. Sie entspricht daher ungefähr der Hälfte des FSR , also $A_{max} = 2,5 \text{ V}$.
- f) Das binäre Wort **1000 0001** entspricht $128 + 1 = 129$ in vorzeichenloser („unsigned“) Interpretation bzw. $-128 + 1 = -127$ (-**0111 1111**) in vorzeichenbehafteter („signed“) Interpretation. In nZKF Interpretation repräsentiert das Binärwort den Wert $-127 / 128 = -0,9921875$.

- g) Die **Empfindlichkeit und Auflösung des Sensors mit Vorverstärker** werden durch analoge Verstärkung und Offsetkorrektur maximiert, wenn bei maximaler bzw. minimaler Beschleunigung $a_{max} = \pm 2000 \text{ mg}$ gerade die maximale bzw. minimale Eingangsspannung des ADCs $U_{ADC,max} = 0 \text{ bzw. } 5 \text{ V}$ erreicht wird. Hierfür ist eine Vorverstärkung von

$$A_{pre} = \frac{FSR}{\Delta U_{Sens,max}} = \frac{5 \text{ V}}{\Delta a_{max} S_{Sens}} = \frac{5 \text{ V}}{4g \cdot 0,4 \text{ V g}^{-1}} = 3,125 \text{ erforderlich.}$$

Damit erhöht sich die Empfindlichkeit des Sensors mit Vorverstärker auf

$$S_{Sens,Pre} = \frac{FSR}{\Delta a_{max}} = \frac{5 \text{ V}}{4g} = A_{pre} S_{Sens} = 3,125 \cdot 0,4 \text{ V/g} = 1,25 \text{ V/g}$$

und die Auflösung bezogen auf die Messgröße auf

$$q_{a,Pre} = q_U / S_{Sens,pre} = 19,53 \text{ mV} / 1250 \text{ mV/g} = 15,62 \cdot 10^{-3} \text{ g.}$$

- h) Die **Anzahl der genutzten Codeschritte** des ADCs erhält man, indem man den Bereich der angelegten Spannung durch die Auflösung q_U des ADCs dividiert:

$$\begin{aligned} N_{dir} &= \Delta U_{Sens,max} / q_U = \Delta U_{Sens,max} 2^W / FSR = 1,6 \text{ V} \cdot 256 / 5 \text{ V} = 81,9 \\ \Rightarrow ENOB_{dir} &= \log_2(N_{dir}) = 6,36 \text{ Bits} \\ N_{pre} &= \Delta U_{Sens,pre,max} / q_U = FSR / q_U = 2^8 = 256 \\ \Rightarrow ENOB_{pre} &= \log_2(N_{pre}) = 8 \text{ Bits} \end{aligned}$$

Man gewinnt also mehr als 1,5 Bit effektive Auflösung, indem man die zu wandelnde Spannung richtig an den Eingangsbereich des Wandlers anpasst. Allerdings ist es heutzutage oft billiger, einen höher auflösenden ADC zu verwenden und den *FSR* nicht voll auszunutzen als einen einfacheren ADC mit zusätzlichem Vorverstärker einzusetzen.

- i) Beim **MA-Filter 15. Ordnung** werden insgesamt $N = 16$ Werte aufaddiert, die Wortlänge erhöht sich daher um $\lceil \log_2(16) \rceil = 4$ Bits (bit growth). Der Fall eines DC-Eingangssignals mit $x[n] = 1$ kann nicht vorkommen, da der Wert 1 außerhalb des Wertebereichs liegt, daher erreicht das Aussgangssignal auch nicht ganz den Wert 16. Bei einem MA-Filter mit der Ordnung 16 (also mit 17 Taps) müsste man jedoch die Wortlänge bereits um 5 Bits erhöhen. Die zusätzlichen 4 Bits werden als Integerbits benötigt, um den vergrößerten Wertebereich aufzunehmen, es wird also das Format Q4.7 benötigt.

Ein MA-Filter der Ordnung $N - 1$ hat N gleichverteilte Nullstellen auf dem EK, wobei die Nullstelle bei $z = 1$ durch eine Polstelle kompensiert wird. Die erste Nullstelle liegt daher bei $F_{0,1} = 1/N = 1/16$. Der Betragsgang (nicht gefragt) wird beschrieben durch (-> Kap. 2):

$$|H(f)| = \left| \frac{\sin(N\Omega/2)}{\sin(\Omega/2)} \right| = |ND_N(\Omega)| \text{ mit } H(0) = N$$

M5.2. * Moving Average Filter mit endlicher Wortbreite → Aufgabe 5.2

- a) Übertragungsfunktion und Frequenzgang

$$H(z) = \sum_{i=0}^8 z^{-i} = \frac{1 - z^{-9}}{1 - z^{-1}} = \frac{z^9 - 1}{(z - 1) z^8} \Rightarrow H(e^{j\Omega}) = \frac{\sin 9\Omega/2}{\sin \Omega/2} e^{-j4\Omega}$$

Das Filter ist 8. Ordnung und hat daher 8 Nullstellen, die beim MA-Filter alle auf dem EK liegen. Die Verteilung der Nullstellen lässt sich am Einfachsten bestimmen, wenn man $H(z)$ mit Hilfe der endlichen geometrischen Reihe umformt (siehe oben). Dann sieht man aus $z^9 - 1 = 0$ sofort, dass 9 Nullstellen auf dem EK verteilt liegen bei $\phi = 2k\pi/9; k = \dots, -2, -1, 0, 1, 2, \dots$. Die Nullstelle bei $z = 1$ wird durch den einzigen Pol kompensiert, so dass man wieder die erwarteten 8 Nullstellen erhält. Die erste Nullstelle liegt daher bei $\Omega = 2\pi/9$ oder $f = f_S/9$.

Das gleiche Ergebnis erhält man natürlich aus $H(e^{j\Omega})$, indem man $9\Omega/2 = \pi$ setzt. Aus $H(e^{j\Omega})$ kann man außerdem die lineare Phase von $\phi = -4\Omega$ ablesen, entsprechend einer Gruppenlaufzeit von $\tau_g = 4T_S$ (bis zur Mitte des Filters gerechnet).

b) Wortlängen

Das Wachstum ΔN der benötigten Wortlänge („bit growth“) bei k Summanden ergibt sich aus $\Delta N = \lceil \log_2 k \rceil$ (siehe Abb. M5.1). Die Wortlängen am Ausgang betragen in beiden Fällen 14 bits, siehe Tab. M5.1. In den Abb. M5.1 sind optionale Quantisierer am Ausgang eingezeichnet, die benötigt werden falls mit 10 bit weitergearbeitet werden soll.

k	1	2	3	4	5	6	7	8	9
$\log_2 k$	0	1	1,6	2	2,3	2,6	2,8	3	3,2
ΔN	0	1	2	2	3	3	3	3	4

Tab. M5.1.: Wortlängenwachstum zu Aufgabe M5.2

c) Kritischer Pfad

Bei der Implementierung in Direktform liegen 8 Addierer im kritischen Pfad, $T_{krit,DF} = 8\tau_{add} = 16$ ns und $f_{max,DF} = 1/T_{krit,DF} = 62,5$ MHz. Bei der transponierten Direktform liegt nur ein Addierer im kritischen Pfad, theoretisch könnte das MA-Filter in dieser Implementierung also achtmal so schnell getaktet werden, $f_{max,TDF} = 1/T_{krit} = 500$ MHz!

d) Benötigte Hardware

Direktform: $8 \times 10 = 80$ FlipFlops, $10 + 11 + 2 \times 12 + 4 \times 13 = 97$ FA.

Transponierte Direktform: $10 + 11 + 2 \times 12 + 4 \times 13 = 97$ FlipFlops und FA.

M5.3. * FIR Filter mit Quantisierung → Aufgabe 5.3

a) Allgemeine Filtereigenschaften

Das Filter hat 11 Koeffizienten und daher 10 Verzögerungsglieder -> zehnter Ordnung. Es ist linearphasig (Koeffizienten sind symmetrisch zur Mitte des Filters) und ein Halbbandfilter (jeder zweite Koeffizient außer dem mittleren ist Null). Da Koeffizientenquantisierung die Symmetrie nicht stört und die Nullen Null bleiben, ist auch das quantisierte Filter ein Halbbandfilter. Es wurde berechnet mit dem Kommando

```
b = fir1(10, [0 0,3 0,7 1], [1 1 0 0])
```

entsprechend der Eckfrequenzen des Durchlassbandes $F_{DB} = 0,15$ und des Sperrbandes $F_{SB} = 0,35$, die symmetrisch zu $F = 0,25$ gewählt wurden.

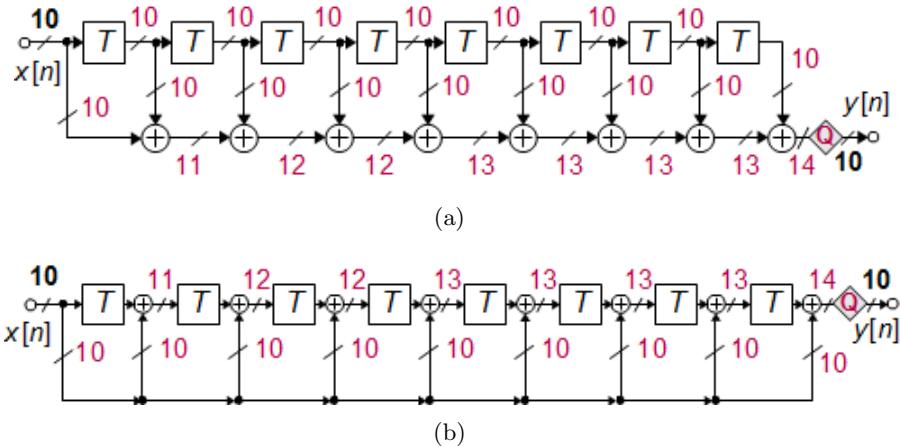


Abb. M5.1.: Moving Average Filter mit Quantisierung zu Aufgabe M5.2, (a) Direktform und (b) transponierte Direktform

b) Hardwareressourcen

In der optimierten Version (Abb. M5.2) werden 6 Addierer und nur drei echte Multiplizierer benötigt; Multiplizierer für identische Koeffizienten können zusammengefasst werden, indem vor der Multiplikation die Datenwörter addiert werden:

$$y[n] = (x[n] + x[n - 10])b_0 + (x[n - 2] + x[n - 8])b_2 + (x[n - 4] + x[n - 6])b_4 + b_5x[n - 5]$$

Der Koeffizient b_5 beträgt genau $1/2$ und kann daher effizient durch Schieben der Bits um eine Position nach rechts realisiert werden.

Im Vergleich dazu werden in der nicht optimierten Version 6 Addierer und 6 Multiplizierer benötigt (auch hier wird der Koeffizient b_5 durch Bitschieben realisiert).

Bei einem allgemeinen FIR-Filter 10. Ordnung werden 10 Addierer und 11 Multiplizierer benötigt.

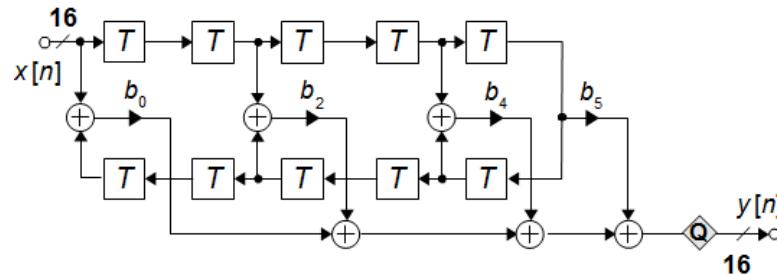


Abb. M5.2.: FIR-Filter mit optimierter Struktur und Quantisierung zu M5.3

c) Kritischer Pfad

Bei einem allgemeinen FIR-Filter 10. Ordnung (Abb. 5.3) ist die Verzögerungszeit des kritischen Pfads $T_{krit} = \tau_{mul} + 10\tau_{add} = 30$ ns, die maximale Taktrate ist daher $f_{max} = 1/T_{krit} = \mathbf{33,3 \text{ MHz}}$.

Wenn man die Bedingung ausnutzt, dass jeder zweite Koeffizient Null ist (Halbbandfilter), ist der kritische Pfad $T_{krit} = \tau_{mul} + 6\tau_{add} = 22$ ns, die maximale Taktrate ist daher $f_{max} = 1/T_{krit} = \mathbf{45,4 \text{ MHz}}$.

Beim optimierten Filter (Abb. M5.2) ist die Verzögerungszeit des kritischen Pfads $T_{krit,opt} = \tau_{mul} + 4\tau_{add} = 18$ ns, die maximale Taktrate ist daher $f_{max,opt} = 1/T_{krit,opt} = 55,6$ MHz.

d) Frequenzgang

$$H(f=0) = H(z=1) = \sum_i b_i = 1,003$$

$$H(f=f_S/2) = H(z=-1) = \sum_i (-1)^{-i} b_i = 0,003$$

$$H(f=f_S/4) = H(z=j) = \sum_i (j)^{-i} b_i = -0,5j$$

e) Koeffizientenformat

Der betragsmäßig größte Koeffizient, $b_5 = 0,5$, muss im gewählten Zahlenformat gerade noch darstellbar sein. Dafür wird das Zahlenformat $B(0,17)$ benötigt, ein LSB ist $q = 2^{-17} = 7,6 \cdot 10^{-6}$. Allgemein lässt sich die Anzahl der benötigten Vorkommastellen bestimmen über $WI = \lceil \log_2(\max(b_i)) \rceil$

f) Quantisierte Koeffizienten: Die quantisierten Koeffizienten in Tab. M5.2 und deren Quantisierungsfehler $\epsilon_i = b_{i,id} - b_{i,Q}$ wurden mit dem Computer berechnet. Der Fehler ist bei positiven Koeffizienten immer größer Null und bei negativen Koeffizienten kleiner Null, da die Koeffizienten durch Wertschneiden bestimmt wurden und er ist natürlich immer kleiner als ein LSB (2^{-17} bzw. 2^{-7}). Die Koeffizienten in Tab. M5.2 sind auf 4 Nachkommastellen gerundet dargestellt, die ϵ_i -Werte wurden aber natürlich aus den quantisierten Koeffizienten mit allen Nachkommastellen ermittelt.

b_i	ideal	$B(0,17)$	$\epsilon(0,17)$	$B(0,7)$	$\epsilon(0,7)$
$b_0 = b_{10}$	0,01623	0,0162277	$2,28 \cdot 10^{-6}$	0,015625	$0,61 \cdot 10^{-3}$
$b_2 = b_8$	-0,06871	-0,0687103	$0,33 \cdot 10^{-6}$	-0,070312	$1,60 \cdot 10^{-3}$
$b_4 = b_6$	0,30399	0,303986	$4,40 \cdot 10^{-6}$	0,296875	$7,11 \cdot 10^{-3}$
b_5	0,5	0,5	0	0,5	0
LSB	—	—	$7,6 \cdot 10^{-6}$	—	$7,8 \cdot 10^{-3}$

Tab. M5.2.: Filterkoeffizienten in verschiedenen Zahlenformaten zu M5.3

```

1 # Ende der gemeinsamen Import-Anweisungen
2 import dsp_fpga_fix_lib as fx
3
4 b = [0.01623, 0, -0.06871, 0, 0.30399, 0.5, 0.30399, 0, -0.06871, 0, 0.01623]
5 #
6 q_obj7 = {'QI':0, 'QF': 7, 'quant':'floor', 'ovfl': 'none'}
7 q_obj17 = {'QI':0, 'QF': 17, 'quant':'floor', 'ovfl': 'none'}
8 #
9 fx_7 = fx.Fixed(q_obj7)
10 fx_17 = fx.Fixed(q_obj17)
11
12 bq7  = fx_7.fix(b) # quantize a
13 bq17 = fx_17.fix(b)

```

```

14 title_str = "    b      | bq(0.17) | eps(0.17) | bq(0.7)   | eps(0.7) "
15 print(title_str, "\n", "-"*len(title_str))
16 for i in range(len(b)):
17     print("{0:8.5f} | {1:10.6g} | {2:9.2E}| {3:9.6f} | {4:9.2E}".format(b[i], bq17[i],
18                           b[i] - bq17[i], bq7[i], b[i] - bq7[i]))

```

Lst. M5.1: Python Listing zu Aufgabe 5.3

- g) Die **Koeffizientenfläche** ist die Summe des Betrags aller Koeffizienten: $A_b = \sum |b_i| = 1,2779$. Dieser Wert wird benötigt, um genauer abzuschätzen welche Wortbreite der Akkumulator für die Teilprodukte haben muss. Kleinere Koeffizienten führen zu kleineren Teilprodukten und damit zu geringerem „bit growth“ des Akkumulators.
- h) **Skalierung der Daten vor Multiplikation:** Es gibt zwei Möglichkeiten die Daten vor der Multiplikation zu skalieren: Das Datenwort kann um zwei Bits nach links geschoben werden, die beiden LSBs werden mit Nullen aufgefüllt. Die zwei LSBs des Produkts sind dann ebenfalls Null und können ohne Genauigkeitsverlust abgeschnitten werden. Alternativ können zwei MSBs ergänzt werden, hier muss allerdings das Vorzeichen beachtet werden und mit '1' oder '0' aufgefüllt werden. Jetzt sind die beiden MSBs des Produkts irrelevant und können weggelassen werden.

In beiden Fällen hat das Produkt $18 + 16 = 34$ relevante Bits.

- i) Im **Akkumulator** müssen bei einem Filter 10. Ordnung im allgemeinen Fall 11 Teilprodukte aufsummiert werden. Das bedeutet einen Bit Growth von $\lceil \log_2(11) \rceil = 4$ Bits, der Akkumulator muss also $34 + 4 = 38$ Bits breit sein. Berücksichtigt man, dass nur 7 Koeffizienten ungleich Null sind, werden nur 3 zusätzliche Bits benötigt. Berücksichtigt man außerdem die Koeffizientenfläche A_b , also den Betrag aller Koeffizienten, benötigt man nur $\lceil \log_2(A_b) \rceil = 1$ zusätzliches Bit.

j) Akkumulatorskalierung

Wenn der Akkumulator dimensioniert wurde wie im vorigen Aufgabenpunkt beschrieben, wird sein Wertebereich voll ausgenutzt. Es werden dann einfach die obersten 16 Bits benutzt (ggf. nach Rundung) und die unteren Bits abgeschnitten.

M5.4. FIR Filter mit Skalierung → Aufgabe 5.4

a) Frequenzgang bei $f = 0$

$$H_1(f=0) = H_1(z=1) = 1 + 7/4 + 1/2 - 1/4 = 3$$

b) Überlauf im Filter

Bei diesem einfachen FIR-Filter kann nur in den Summierern (bzw. im Akkumulator wenn die Summierer zusammengefasst werden) ein Überlauf auftreten.

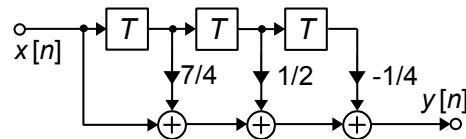


Abb. M5.3.: SFG in Direktform zu Aufgabe 5.4

c) **L^∞ -Skalierung**

heißt: Es tritt an keinem Knoten ein Überlauf bei sinusförmigem Eingangssignal in eingeschwungenem Zustand auf. Aus Frequenzgang ist erkennbar, dass das Maximum bei $H_1(f=0) = L^\infty = H_1(z=1) = 3$ auftritt, also bei der Eingangsfolge $x[n] = \{1; 1; 1; 1\}$.

Der Skalierungsfaktor lautet dementsprechend: $k_{E_\infty} = 1/L^\infty = 1/3$

d) **L^1 -Skalierung**

heißt: Es tritt kein Überlauf bei beliebigem Eingangssignal auf; worst case ist eine Folge von Einheitsimpulsen, die Vorzeichen der Einheitspulse hängen von den Vorzeichen der Koeffizienten ab. $L^1 = \sum |h_1[n]| = 3,5$. Dieser Wert tritt auf bei $x[n] = \{-1; 1; 1; 1\}$. Der Skalierungsfaktor lautet: $k_{E1} = 1/L^1 = 1/3,5$

M5.5. IIR-Filter mit Skalierung → Aufgabe 5.5

a) Übertragungsfunktion $H(z)$

$$H(z) = k \frac{1 - 1,2z^{-1} + z^{-2}}{1 - 1,8z^{-1} + 0,81z^{-2}}$$

b) Pole und Nullstellen

$$\text{Nullstellen: } z_{0;1,2} = 0,6 \pm \sqrt{0,36 - 1} = \mathbf{0,6 \pm j0,8}$$

$$\Rightarrow r_{0;1,2} = \sqrt{0,6^2 + 0,8^2} = \mathbf{1}, \phi_{0;1,2} = \pm \arctan \frac{\Im\{z_0\}}{\Re\{z_0\}} = \frac{\pm 0,8}{0,6} = \pm 53^\circ$$

$$\text{Polstellen: } z_{\infty;1,2} = 0,9 \pm \sqrt{0,81 - 0,81} = \mathbf{0,9} \text{ (doppelte Polstelle)}$$

Achtung: Die obige Formel zur Berechnung des Winkels stimmt nur für die rechte Halbebene - wenn $\Re\{z_0\} < 0$ ist, muss die atan2 - Funktion verwendet werden!

Das Maximum des Amplitudengangs $|H(e^{j\Omega})|$ liegt bei $\Omega = 0$ ($z = 1$) - dort kommt der Einheitskreis dem Pol am nächsten. Da die Nullstellen hier auf dem Einheitskreis liegen, wird dort auch der Amplitudengang minimal. Die Frequenz, bei der das passiert, lässt sich sofort aus dem Winkel der Nullstellen $\phi_{0;1,2}$ bestimmen: $f_{0;1,2} = f_S \phi_{0;1,2}/360^\circ = 0,148f_S$. Das Filter hat also Tiefpasscharakteristik (Abb. M5.4(b)).

c) Skalierung des rekursiven Teils $H_r(z)$ bis Punkt A_a

$$H_r(z) = \frac{k_c}{1 - 1,8z^{-1} + 0,81z^{-2}}$$

Auch das Maximum des Amplitudengangs von $H_r(e^{j\Omega})$ liegt bei $\Omega = 0$ ($z = 1$). Damit der Betrag der Übertragungsfunktion dort den Wert 1 erreicht, muss die Struktur folgendermaßen *skaliert* werden:

$$H_{r,max} = H_r(z=1) = \frac{k_c}{1 - 1,8 + 0,81} = 100k_c \stackrel{!}{=} 1 \Rightarrow k_c = 0,01$$

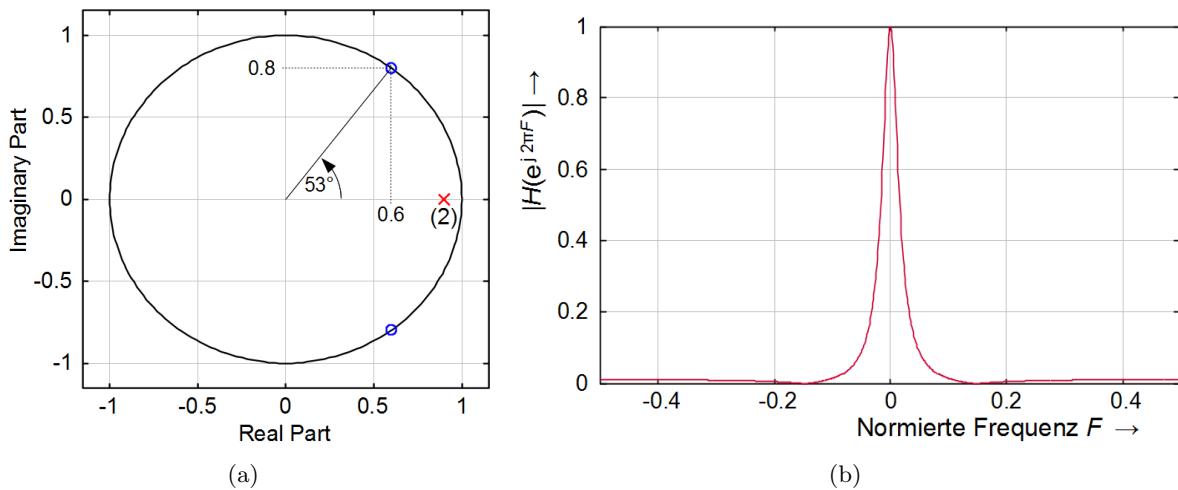


Abb. M5.4.: IIR-Filter mit Skalierung zu Aufgabe M5.5, (a) P/N-Diagramm und (b) Amplitudengang $|H(e^{j2\pi F})|$ für $k = 0,01$

d) Skalierung der gesamten Übertragungsfunktion $H(z)$

$$H(f=0) = H(z=1) = k_d \frac{1 - 1,2 + 1}{1 - 1,8 + 0,81} = 80k_d \stackrel{!}{=} 1 \Rightarrow k_d = 0,0125$$

Legt man mit dieser Skalierung das DC-Signal $x = 1$ an, stellt sich bei A_a im eingeschwungenen Zustand der Wert 1,25 ein. An dieser Stelle gibt es daher entweder einen Überlauf (Zahlenbereich $-1 \dots 1$) oder man muss ein Bit mehr „spendieren“, um einen Zahlenbereich von $-2 \dots 2$ zu erreichen.

e) Optimierung der Struktur

Bei Skalierung mit $k_c = 0,01$ erreicht der Ausgang nicht ganz den Wert 1:

$$H_{max} = H(z=1) = 80k_c = 0,8 \hat{=} 20 \log_{10}(0,8) \text{ dB} = -1,9 \text{ dB}$$

Diese Reduktion der maximalen Aussteuerung verschlechtert also den Signal-Rauschabstand um 1,9 dB. Da im rekursiven Teil bereits die maximale Aussteuerung von 1 erreicht wird (\rightarrow Unterpunkt c), kann nur im nicht-rekursiven Teil die Aussteuerung wieder ausgeglichen werden. Hierzu werden die Koeffizienten aller nicht-rekursiven Koeffizienten mit $1/0,8 = 1,25$ multipliziert (Abb. M5.5).

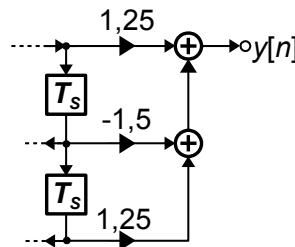


Abb. M5.5.: Optimierte IIR-Filter mit Skalierung des nicht-rekursiven Teils zu Aufgabe M5.5

Achtung: Diese Optimierung ist nur möglich, weil hier der nicht-rekursive Teil dem rekursiven Teil folgt (Direktform 2)!

f) Impulsantwort $h[n]$

Differenzengleichung aus $H(z)$:

$$\begin{aligned} H(z) &= \frac{Y(z)}{X(z)} = \frac{1}{128} \frac{1 - 1,2z^{-1} + z^{-2}}{1 - 1,8z^{-1} + 0,81z^{-2}} \\ \Rightarrow Y(z) &= \frac{1}{128} X(z) [1 - 1,2z^{-1} + z^{-2}] + Y(z) [1,8z^{-1} - 0,81z^{-2}] \\ \Rightarrow y[n] &= \frac{1}{128} \underbrace{[x[n] - 1,2x[n-1] + x[n-2]]}_{\text{nicht-rekursiver Teil}} + \underbrace{[1,8y[n-1] - 0,81y[n-2]]}_{\text{rekursiver Teil}} \end{aligned}$$

Erregung mit $x[n] = \delta[n] = \{1,0,0,\dots\} \Rightarrow$ Impulsantwort $h[n] = y[n]$:

$$\begin{aligned} h[0] &= \frac{1}{128} [1 - 1,2 \cdot 0 + 0] = \frac{1}{128} = 7,8125 \cdot 10^{-3} \\ h[1] &= \frac{1}{128} [0 - 1,2 \cdot 1 + 0] + 1,8h[0] - 0,81 \cdot 0 = 4,6875 \cdot 10^{-3} \\ h[2] &= \frac{1}{128} [0 - 1,2 \cdot 0 + 1] + 1,8h[1] - 0,81h[0] = 9,9219 \cdot 10^{-3} \\ h[3] &= \frac{1}{128} \underbrace{[0 - 1,2 \cdot 0 + 0]}_{=0 \text{ für } n>2} + 1,8h[2] - 0,81h[1] = 14,0625 \cdot 10^{-3} \\ &\vdots \\ h[n] &= 1,8h[n-1] - 0,81h[n-2] \quad \text{für } n > 2 \end{aligned}$$

Anmerkung: Das Filter ist stabil, auch wenn das nach den ersten Werten nicht so aussieht. Die Impulsantwort erreicht das Maximum erst bei $h[10] = 0,024$; der doppelte Pol ist relativ nah am Einheitskreis und verursacht einen kräftigen Überschwinger.

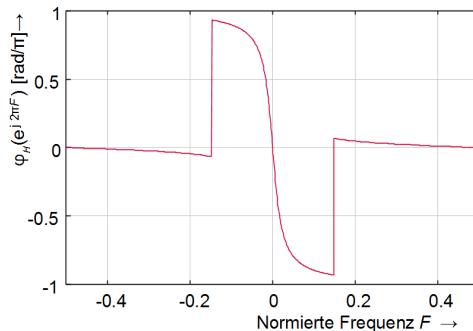


Abb. M5.6.: Phasengang des IIR-Filters zu Aufgabe M5.5

g) Amplituden- und Phasengang von $H(f)$ bei $f = 0$ und $f = f_S/2$

$$\begin{aligned} H(f=0) &= H(z=1) = \frac{1}{128} \cdot \underbrace{80}_{\text{aus d)}} = 0,625 = |H(f=0)|; \quad \varphi_H(f=0) = 0 \\ H\left(f = \frac{f_S}{2}\right) &= H(z = -1) = \frac{1}{128} \cdot \frac{1 - 1,2 \cdot (-1) + (-1)^2}{1 - 1,8(-1) + 0,81 \cdot (-1)^2} \\ &= \frac{1}{128} \cdot \frac{3,2}{3,61} = 6,92 \cdot 10^{-3} = \left|H\left(f = \frac{f_S}{2}\right)\right|; \quad \varphi_H\left(f = \frac{f_S}{2}\right) = 0 \end{aligned}$$

Anmerkung:

Im Allgemeinen ist der Phasengang recht aufwändig zu berechnen (Phasenwinkel des Zählers - Phasenwinkel des Nenners). Solange nur reellwertige Argumente verwendet werden, kann aber $H(z)$ nur reellwertig sein. Damit ist $\varphi_H(z) = 0^\circ$ (wenn $H(z) > 0$) oder $180^\circ \hat{=} \pi$ (falls $H(z) < 0$).

M5.6. IIR-Filter mit Quantisierung und kleinen Grenzzyklen → Aufgabe 5.6**a) Exakte Lösung ohne Quantisierungseffekte**

$$\begin{aligned}
 y[n] &= x[n] - 0,88 y[n-1] \\
 \Rightarrow h[n] &= \delta[n] - 0,88 h[n-1] \\
 h[0] &= \delta[0] = 1 \\
 h[1] &= \delta[1] - 0,88 h[0] = -0,88 \\
 h[2] &= -0,88 h[1] = (-0,88)^2 = 0,774 \\
 &\vdots \\
 h[n] &= (-0,88)^n
 \end{aligned}$$

Die Impulsantwort klingt ab gegen Null für $n \rightarrow \infty$. Zum gleichen Ergebnis gelangt man über die inverse z -Transformation:

$$y[n] = x[n] + ay[n-1] \Rightarrow H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 - az^{-1}} \quad \bullet \rightsquigarrow h[n] = a^n \text{ mit } a = -0,88$$

Das Beispiel zeigt, dass man für einfache Strukturen die Impulsantwort von IIR-Filtern in geschlossener Form auch ohne z -Transformation ableiten kann. Für Systeme höherer Ordnung muss man zunächst $H(z)$ aufstellen und über Partialbruchzerlegung in Terme erster und zweiter Ordnung zerlegen. Aus dieser Darstellung kann man über die inverse z -Transformation die Impulsantwort ermitteln.

b) Quantisieren durch Rundung

Bei Berücksichtigung von Quantisierungseffekten (Nichtlinearitäten!) muss die Impulsantwort Schritt für Schritt bestimmt werden:

$$\begin{aligned}
 h_Q[n] &= \delta[n] - Q\{0,88 h_Q[n-1]\} \\
 \Rightarrow h_Q[0] &= \delta[0] - 0 &= 1 \\
 h_Q[1] &= 0 - Q\{0,88 h_Q[0]\} &= -0,9 \\
 h_Q[2] &= 0 - Q\{0,88 h_Q[1]\} = -Q\{0,88 \cdot (-0,9)\} = -Q\{-0,792\} &= 0,8 \\
 h_Q[3] &= 0 - Q\{0,88 h_Q[2]\} = -Q\{0,88 \cdot 0,8\} = -Q\{0,704\} &= -0,7 \\
 h_Q[4] &= 0 - Q\{0,88 h_Q[3]\} = -Q\{0,88 \cdot (-0,7)\} = -Q\{-0,616\} &= 0,6 \\
 h_Q[5] &= 0 - Q\{0,88 h_Q[4]\} = -Q\{0,88 \cdot 0,6\} = -Q\{0,528\} &= -0,5 \\
 h_Q[6] &= 0 - Q\{0,88 h_Q[5]\} = -Q\{0,88 \cdot (-0,5)\} = -Q\{-0,440\} &= 0,4 \\
 h_Q[7] &= 0 - Q\{0,88 h_Q[6]\} = -Q\{0,88 \cdot 0,4\} = -Q\{0,352\} &= -0,4 \\
 h_Q[8] &= 0 - Q\{0,88 h_Q[7]\} = -Q\{0,88 \cdot (-0,4)\} = -Q\{-0,352\} &= \mathbf{0,4} \\
 \rightarrow & \text{Grenzzyklus!} \\
 \Rightarrow h_Q[n] &= 0,4 \cdot (-1)^n \quad \text{für } n > 5
 \end{aligned}$$

Ab dem 6. Schritt wechselt die Impulsantwort zwischen den Werten +0,4 und -0,4 hin und her, konvergiert also nicht gegen 0 wie beim unquantisierten Filter. Man nennt dies einen *kleinen Grenzzyklus* oder *Quantisierungsgrenzzyklus* (engl. *small (amplitude) limit cycle* oder *quantization limit cycle*) mit der Periode $2T_S$. Einen Grenzzyklus mit der Periode T_S liegt vor, wenn ein konstanter Rest am Ausgang stehen bleibt (vorausgesetzt, das unquantisierte Filter hat eine gegen Null abklingende Impulsantwort). In dieser Aufgabe entsteht der Grenzzyklus durch die Rundung von -0,352 hin zum betragsmäßig größeren Wert -0,4.

Außerdem gibt es noch *große Grenzzyklen* oder *Überlaufgrenzzyklen*, die durch einen Überlauf und den damit verbundenen Vorzeichenwechsel (bei Zweierkomplementdarstellung) entstehen. Überlaufgrenzzyklen lassen sich vermeiden durch Sättigungslogik oder durch reduzierte Aussteuerung.

c) Quantisieren durch Abschneiden

Die Rechnung verläuft prinzipiell genau wie unter b), hier ergibt sich die Impulsantwort:

$$h_Q[n] = \{1; -0,8; 0,7; -0,6; 0,5; -0,4; 0,3; -0,2; 0,1; 0; 0; \dots\}$$

Hier konvergiert das Ausgangssignal nach einer endlichen Anzahl von Schritten gegen Null. Da das einfache Abschneiden oder Wertschneiden auch zu betragsmäßig größeren Werten führen kann, kann unter anderen Randbedingungen auch Wertschneiden zu kleinen Grenzzyklen führen.

d) Filter mit binärer Arithmetik

Die Rechnung verläuft wieder genau wie bei b) und c), für $a = \pm 9/16$ erhält man einen Grenzzyklus mit $T = T_S$ (= konstanter Ausgangswert, $y = 1/16$) bzw. einen Grenzzyklus mit $T = 2T_S$ (= alternierender Ausgangswert, $y = \pm 1/16$).

Weitergehende Informationen zu Grenzzyklen: [KK09], [Wer09], [PBW08]

M5.7. Integrator mit endlicher Wortbreite → Aufgabe 5.7

Der Wertebereich eines vorzeichenbehafteten 8 bit Worts beträgt -128 ... 127. Für ein positives Eingangssignal ist der maximale Ausgangswert des Integrators daher +127.

a) Integrator erster Ordnung - Impulsantwort:

$$y_{a,\delta}[n+1] = y_a[n] + \delta[n] = u[n] \Rightarrow y_{a,\delta}[n] = u[n-1] = \{0; 1; 1; 1; \dots\}$$

z -Transformation:

$$Y_{a,\delta}(z)z = Y_{a,\delta}(z) + 1 \quad \Rightarrow \quad Y_{a,\delta}(z) = \frac{1}{z-1} = \frac{z^{-1}}{1-z^{-1}} \quad \bullet \rightsquigarrow \circ \quad y_{a,\delta}[n] = u[n-1]$$

Die z -Transformierte der Impulsantwort ist die Übertragungsfunktion, $H_a(z) = Y_{a,\delta}(z) = 1/(z-1)$ mit einer Polstelle bei $z = 1$ und einer Nullstelle im Unendlichen. Für Gleichspannungs-Eingangssignale ist die Übertragungsfunktion daher unendlich, $H(f = 0) = H(z = 1) \rightarrow \infty$. $H(f)$ fällt monoton ab zu $H(f = f_S/2) = H(z = -1) = -1/2$.

Berechnung der Frequenz, bei der der Betragsgang 1 wird:

$$\begin{aligned} |H(e^{j\Omega})| &= \left| \frac{1}{e^{j\Omega} - 1} \right| = \left| \frac{1}{\cos \Omega + j \sin \Omega - 1} \right| = \frac{1}{\sqrt{\cos^2 \Omega - 2 \cos \Omega + 1 + \sin^2 \Omega}} = 1 \\ \Rightarrow 2 - 2 \cos \Omega &= 1 \quad \Rightarrow \cos \Omega = 1/2 \quad \Rightarrow \Omega = \arccos 1/2 = \pi/3 \text{ (entspricht } f_S/6) \end{aligned}$$

Sprungantwort (nicht gefragt; Lösung durch „genaues Hinschauen“):

$$y_{a,u}[n+1] = y_{a,u}[n] + u[n] = (n+1)u[n] = \{1; 2; 3; \dots\} \Rightarrow y_{a,u}[n] = nu[n-1] = \{0; 1; 2; \dots\}$$

oder über z -Transformation:

$$zY_{a,u}(z) = Y_{a,u}(z) + \frac{z}{z-1} \Rightarrow Y_{a,u}(z) = \frac{z}{(z-1)^2} = \frac{z^{-1}}{(1-z^{-1})^2} \quad \bullet \rightsquigarrow \circ \quad y_{a,u}[n] = nu[n-1]$$

Der Integrator läuft beim Schritt $n = 128$ über, das Ausgangswort wird dann vorzeichenbehaftet als -128 interpretiert.

b) Verlustbehafteter Integrator - Impulsantwort:

$$\begin{aligned} y_{b,\delta}[n+1] &= ay_{b,\delta}[n] + \delta[n] \\ \circ \rightsquigarrow \bullet \quad zY_{b,\delta}(z) &= aY_{b,\delta}(z) + 1 \Rightarrow Y_{b,\delta}(z) = \frac{1}{z-a} \\ \bullet \rightsquigarrow \circ \quad y_{b,\delta}[n] &= a^{n-1}u[n-1] = \{0; 1; a; a^2; \dots\} \end{aligned}$$

Für $a < 1$ konvergiert die Folge gegen 0.

Die Übertragungsfunktion ist $H_b(z) = Y_{b,\delta}(z) = 1/(z-a)$, die Polstelle liegt bei $z = a$. Das Maximum von $H(f)$ wird auch hier bei $f = 0$ erreicht, hat aber einen endlichen Wert („verlustbehafteter Integrator“), das System ist stabil.

Ähnlich wie in Aufgabenteil a) lässt sich bestimmen, dass $|H(e^{j\Omega})| = 1$ gilt für $\Omega = \arccos a/2$.

c) Integrator zweiter Ordnung - Impulsantwort:

$$y_{c,\delta}[n+1] = y_{c,\delta}[n] + y_{a,\delta}[n] = y_{c,\delta}[n] + u[n-1] = y_{a,u}[n-1] = (n-1)u[n-2]$$

oder über z -Transformation:

$$\begin{aligned} \circ \rightsquigarrow \bullet \quad zY_{c,\delta}(z) &= Y_{c,\delta}(z) + \frac{1}{z-1} \Rightarrow Y_{c,\delta}(z) = \frac{1}{(z-1)^2} = \frac{z^{-2}}{(1-z^{-1})^2} \\ \bullet \rightsquigarrow \circ \quad y_{a,u}[n] &= (n-1)u[n-2] = \{0; 0; 1; 2; 3; \dots\} \end{aligned}$$

Der Integrator läuft beim Schritt $n = 129$ über, das Ausgangswort wird dann vorzeichenbehaftet als -128 interpretiert.

Die Übertragungsfunktion ist $H_c(z) = Y_{c,\delta}(z) = 1/(z-1)^2$, es gibt eine doppelte Nullstelle bei $z = 1$, $H(f=0) = H(z=1) \rightarrow \infty$. $H(f)$ fällt monoton ab zu $H(f=f_S/2) = H(z=-1) = 1/4$.

Genau wie in Aufgabenteil a) ist $|H(e^{j\Omega})| = 1$ bei $\Omega = \pi/3$.

M5.8. * Verständnisfragen zu Filtern mit Quantisierung → Aufgabe 5.8

Aussage	FIR-Filter		IIR-Filter	
	ja	nein	ja	nein
Unendliche Impulsantwort: (1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Stabil: (2)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Linearphasig: (3)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Minimalphasig: (4)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Pole außerhalb des Ursprungs: (5)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Pole außerhalb des Einheitskreises: (2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nullstellen außerhalb des Ursprungs: (5)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nullstellen außerhalb des Einheitskreises: (2)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Halbbandfilter: (3)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Konstante Gruppenlaufzeit: (3)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Transponieren möglich:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Entwurf über bilineare Transformation:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Transversale Struktur:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Abgeleitet von analogen Butterworth-Filtern:	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kaskadierte Form ist unempfindlicher gegen Quantisierungseffekte als direkte Form:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Grenzzyklen: (1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Tab. M5.3.: Musterlösung zu Aufgabe 5.8

Erklärungen zu Tab. M5.3

- a) Bei IIR Filtern führt die endliche Rechengenauigkeit der quantisierten Arithmetik dazu, dass entweder Grenzzyklen auftreten (→ unendliche Impulsantwort) oder dass die Ergebnisse aller Multiplikationen zu Null gerundet / abgeschnitten werden (→ endliche Impulsantwort), siehe Aufgabe 5.6. Bei FIR-Filtern können keine Grenzzyklen auftreten, da sie keine Rückkopplung haben. Hier kann es höchstens vorkommen, dass Koeffizienten beim Quantisieren Null werden und dadurch die Impulsantwort kürzer wird.
- b) Durch Koeffizientenquantisierung verschiebt sich die Lage von Pol- und Nullstellen, damit werden auch $H(f)$ und $H(z)$ verfälscht. Da nur IIR-Filter Pole außerhalb des Ursprungs haben, können auch hier Polstellen aus dem EK herausgeschoben werden und dadurch das System destabilisieren. Nullstellen werden oft auch außerhalb des EK platziert (→

„Linearphasige Filter“); falls sie durch die Quantisierung aus dem EK herauswandern, beeinträchtigt das nicht die Stabilität.

- c) Linearphasige Filter und Halbbandfilter setzen Symmetrien der Koeffizienten voraus. Da die Symmetrie bei der Quantisierung erhalten bleibt, werden auch Linearphasigkeit und/oder die Eigenschaft des Halbbandfilters nicht gestört. Das gleiche gilt für die konstante Gruppenlaufzeit, die alle linearphasigen Filter haben.
- d) Bei minimalphasigen Filtern müssen alle Null- und Polstellen innerhalb des EK liegen. Die Verschiebung der Pol- und Nullstellen durch Quantisierung kann diese Eigenschaft u.U. ändern.
- e) Pole und Nullstellen im Ursprung entsprechen zusätzlichen (positiven oder negativen) Verzögerungen, die durch die Quantisierung nicht beeinflusst werden.

M6. NOI: Wortlängeneffekte im Frequenzbereich: Quantisierungsrauschen

M6.1. Quantisierungsrauschen Analog-Digital-Wandlung → Aufgabe 6.1

- a) **Maximales $SQNR$:** Für einen Nyquist-Wandler ($f_N = f_S/2$) mit W bit Auflösung ist das maximal erreichbare $SQNR$ für sinusförmige Signale mit Vollaussteuerung

$$SQNR_{max} = (6,02W + 1,76) \text{ dB} = 61,96 \text{ dB} \text{ für } W = 10.$$

Umgekehrt kann aus einem gegebenen $S(Q)NR$ die effektive Anzahl von Bits berechnet werden:

$$ENOB = (SNR - 1,76 \text{ dB})/6,02 = 8,51 \text{ bit für } SNR = 53 \text{ dB.}$$

- b) **Verringertes SNR :** Nichtidealitäten des ADC können das SNR verringern oder Signalkomponenten oberhalb der Nyquistfrequenz $f_S/2$ fallen bei der Abtastung zurück ins Nutzband (hier in der Aufgabenstellung ausgeschlossen). Die wahrscheinlichste Ursache ist jedoch, dass der Eingangsspannungsbereich nicht ausgenutzt wird: Die Signalamplitude A bestimmt die Signalleistung S , während die Rauschleistung N_Q unabhängig von A ist.

Ein Aussteuerbereich von $FSR = 2 \text{ V}$ entspricht einer maximalen Signalamplitude $A_{max} = FSR/2 = 1 \text{ V}$.

$$\begin{aligned} SQNR &= \frac{S}{N_Q} = \frac{S}{N_Q} \frac{S_{max}}{S_{max}} = SQNR_{max} \frac{S}{S_{max}} = SQNR_{max} \frac{A^2/2}{A_{max}^2/2} \\ \Rightarrow SQNR[\text{dB}] &= SQNR_{max}[\text{dB}] + 10 \log_{10} \frac{S}{S_{max}} = SQNR_{max}[\text{dB}] + 20 \log_{10} \frac{A}{A_{max}} \\ \Rightarrow A &= A_{max} 10^{(SNR-SQNR_{max})/20} = 1 \text{ V} \cdot 10^{8,78/20} = \mathbf{0,36 \text{ V}} \end{aligned}$$

Hat man bereits die $ENOB$ des ADCs berechnet, könnte man auch daraus die Eingangsamplitude bestimmen, allerdings werden die $ENOB$ normalerweise bei Vollaussteuerung bestimmt.

- c) Bei der (Re-)Quantisierung eines Signals auf eine LSB-Größe von q entsteht ein Fehler, der im zeitlichen Mittel eine **Rauschleistung** $N_Q = q^2/12$ hat. Diese Rauschleistung ist unabhängig von der Abtastfrequenz, so wie die Leistung eines Sinussignals unabhängig von dessen Frequenz ist! Die Rauschleistung ist außerdem unabhängig von der Amplitude des zu quantisierenden Signals.¹

Die Größe eines LSBs hängt ab vom Aussteuerbereich (Full Scale Range, FSR) und der Anzahl der Bits: $q = FSR/(2^W - 1) \approx FSR/2^W$ für $2^W \gg 1$. Hier ist $FSR = 2 \text{ V}$ und $W = 10$:

$$N_Q = \frac{q^2}{12} = \frac{FSR^2}{12 \cdot 2^{2W}} = \frac{4\text{V}^2}{12 \cdot 2^{20}} = 3,17 \cdot 10^{-7} \text{ V}^2$$

¹Die Amplitude muss natürlich so groß sein, dass überhaupt ein paar Quantisierungsstufen ausgesteuert werden.

Das Quantisierungsrauschen soll auch hier als weiß betrachtet werden; so dass sich die Rauschleistung gleichmäßig über das gesamte (positive) Basisband $f_S/2$ verteilt. Die **Rauschleistungsdichte** N'_Q ist

$$N'_Q(f_{S1}) = \frac{N_Q}{f_{S1}/2} = \frac{3,17 \cdot 10^{-7} \text{ W}}{5 \text{ kHz}} = 6,34 \cdot 10^{-11} \text{ W/Hz bzw.}$$

$$N'_Q(f_{S2}) = \frac{N_Q}{f_{S2}/2} = \frac{6,34 \cdot 10^{-7} \text{ W}}{500 \text{ kHz}} = 6,34 \cdot 10^{-13} \text{ W/Hz}$$

Aus der Rauschleistungsdichte N'_Q berechnet man die Rauschleistung N_Q , indem man N'_Q mit der Rauschbandbreite B_N multipliziert: Lässt man z.B. den ADC mit einer Abtastrate von 1 MHz laufen, will damit aber Signale nur bis zu 50 kHz wandeln, kann man das Quantisierungsrauschen zwischen 50 kHz und $f_S/2$ mit einem digitalen Tiefpass der Bandbreite $B = 50 \text{ kHz}$ eliminieren:

$$N_Q = B_N N'_Q(f_{S2}) = B_N \frac{N_Q}{f_{S2}/2} = \frac{50 \text{ kHz}}{500 \text{ kHz}} N_Q = \frac{N_Q}{10} = 3,17 \cdot 10^{-8} \text{ W}$$

Die gesamte Rauschleistung wird also auf ein Zehntel reduziert, was einer Verbesserung der *ENOB* um $10/6 = 1,7$ bit entspricht.

- d) FFT eines ADC-Signals:** Die Gesamtrauschleistung (Quantisierungsrauschen und thermisches Rauschen) verteilt sich auf die $N_{FFT}/2 = 1024$ Bins in Abb. 6.4, im Mittel wurde je Bin gemessen $N' = -125,4 \text{ dBW/Bin}$. Die Gesamtrauschleistung ist daher

$$N = N' N_{FFT}/2 \hat{=} N'[\text{dBW}] + 10 \log_{10}(N_{FFT}/2) = -125,4 \text{ dBW} + 30,1 \text{ dB} = -95,3 \text{ dBW}$$

Die Signalleistung ist $S = -3 \text{ dBW}$, der Wandler ist also voll ausgesteuert (warum?²). Daraus ergeben sich

$$\begin{aligned} SNR &= S - N = 92,3 \text{ dB und} \\ ENOB &= (SNR - 1,76 \text{ dB})/6,02 = 15,0 \text{ bit.} \end{aligned}$$

M6.2. * SQNR eines quantisierten Signals → Aufgabe 6.2

- a) Das maximale Signal-to-Quantization Noise Ratio ($SQNR_{max}$) bei sinusförmigem Signal, das den gesamten Aussteuerbereich ausnutzt, ist gegeben durch**

$$SQNR_{max} = 1,76 \text{ dB} + W \cdot 6,02 \text{ dB} = 98,1 \text{ dB für } W = 16$$

- b) Signal-to-Quantization Noise Ratio $SQNR$ bei reduzierter Signalamplitude**

Wenn $x[n]$ nur ein Zehntel des Aussteuerbereichs ausnutzt, beträgt die Signalleistung nur noch ein Hundertstel, also -20 dB. Das $SQNR$ reduziert sich damit auf 78,1 dB. Das entspricht einer *effective number of bits* (*ENOB*) von **12,7 bit**.

² $S = A^2/2 \hat{=} 20 \log_{10} A - 3 \text{ dB}$. Also muss $A = 0 \text{ dBVpk}$ und damit voll ausgesteuert sein.

M6.3. * Quantisierungsrauschen im FIR-Filter → Aufgabe 6.3

Die Signalleistung am Ein- und Ausgang ist im Durchlassbereich des Filters gleich, da hier der Betragsgang 1 ist. Sie beträgt $S = A^2/2 = FSR^2/8 = 0,5 \text{ W} \hat{=} -3,01 \text{ dBW}$.

Detaillierte Rechenergebnisse finden Sie in Tab. M6.1.

a) Quantisierung am Ausgang auf 16 bit vor DAC $Q_{DAC}(0.15)$

$$N_{Q,DAC} = q^2/12 = 2^{-30}/12 \text{ W} = 7,76 \cdot 10^{-11} \text{ W} \hat{=} 10 \log_{10}(7,76 \cdot 10^{-11}) = -101,1 \text{ dBW}$$

$$\Rightarrow SQNR = S/N_{Q,DAC} = 98,1 \text{ dB}$$

Da dieser Quantisierer direkt am *Ausgang* liegt, gilt $N_{Q,out} = N_{Q,DAC}$, die *ENOB* ist $ENOB = (SQNR - 1,76)/6,02 = 16 \text{ bit}$ (nachrechnen!).

Achtung: Nicht der DAC selbst, sondern die *vorherige* Verringerung der Wortbreite auf 16 bit verursacht das Quantisierungsrauschen!

b) Quantisierung am Eingang durch 15 bit ADC $Q_{ADC}(0.14)$

$$N_{Q,ADC} = q^2/12 = 2^{-28}/12 \text{ W} = 3,10 \cdot 10^{-10} \text{ W} \hat{=} -95,1 \text{ dBW} \Rightarrow ENOB = 15,5 \text{ bit}$$

Auf den ersten Blick scheint das Ergebnis falsch zu sein, da ja nur mit 15 bit am Eingang quantisiert wurde. Da diese Quantisierung am *Eingang* des Filters wirkt, wird das Quantisierungsrauschen mit dem Frequenzgang des Filters bewertet. Das hier betrachtete Filter ist ein Halbbandfilter, daher ist auch die Rauschbandbreite nur $B_N = f_S/4$. Für die ausführliche Rechnung wird zunächst die Rauschleistungsdichte bestimmt, $N'_{Q,ADC} = N_{Q,ADC}/(f_S/2)$.

Daraus erhält man die mit der Bandbreite gewichtete Rauschleistung am Ausgang:
 $N_{Q,out} = B_N N'_{Q,ADC} = N_{Q,ADC}/2$. Mit etwas Übung erkennt man den Faktor 1/2 auch ohne Umweg über die Rauschbandbreite.

Anmerkung: Die Simulation ergibt $SQNR = 95,9 \text{ dB}$ und $ENOB = 15,6 \text{ dB}$, an der Abweichung ist die (zu) einfache Berechnung der Rauschbandbreite schuld: $H(f_S/4) = 1/2$, dementsprechend ist die Ausgangsleistung bei dieser Frequenz nur noch 1/4. Die Bandbreite für die Rauschleistung ist also kleiner als $f_S/4$, das $SQNR$ daher etwas höher als berechnet.

c) Quantisierung durch $Q_{ADC}(0.14)$ und durch $Q_{DAC}(0.15)$

Die Rauschbeiträge von DAC und ADC zum Gesamtrauschen am Ausgang wurden bereits berechnet, daher müssen nur noch beide Leistungen aufaddiert werden (es wird angenommen, dass beide Rauschbeiträge unkorreliert sind):

$$N_{Q,out} = N_{Q,ADC} + N_{Q,DAC} = 3,10 \cdot 10^{-10} \text{ W} + 7,76 \cdot 10^{-11} \text{ W} =$$

d) Quantisierung nach jeder Multiplikation mit 16 bit $Q_{mul}(0.15)$

Die Quantisierung nach den Multiplizierern wirkt sich als Quantisierungsrauschen direkt am Ausgang aus, $N_{Q,out} = 7N_{Q,Mul} = 7N_{Q(0.15)} = 7 \cdot 7,76 \cdot 10^{-11} \text{ W} = 5,43 \cdot 10^{-10} \text{ W}$.

e) Rauschbeiträge von Q_{mul} sollen kleiner sein als die von DAC und ADC:

$$N_{Q,out,mul} = 7N_{Q,Mul} \leq N_{Q,ADC}/2 + N_{Q,DAC} = 2,33 \cdot 10^{-10} \text{ W}$$

$$\Rightarrow N_{Q,Mul} = 2^{-2WF_{mul}}/12 \leq 2,33 \cdot 10^{-10} \text{ W}/7 = 3,33 \cdot 10^{-11} \text{ W}$$

$$\Rightarrow WF_{mul} = \lceil -0,5 \log_2(12 \cdot 3,33 \cdot 10^{-11}) \rceil = 16 \text{ bit}$$

Nach den Multiplizierern muss also auf $Q_{mul}(0.16)$ requantisiert werden, alle 7 Quantisierer erzeugen zusammen eine Rauschleistung von

$$N_{Q,out,mul} = 7N_{Q,Mul} = 7N_{Q(0.16)} = 7 \cdot 1,94 \cdot 10^{-11} \text{ W} = 1,36 \cdot 10^{-10} \text{ W}.$$

Das Gesamtrauschen am Ausgang in diesem Fall ist:

$$N_{Q,out} = 7N_{Q,Mul} + N_{Q,ADC}/2 + N_{Q,DAC} = (1,36 + 2,33) \cdot 10^{-10} \text{ W} = \mathbf{3,69 \cdot 10^{-10} \text{ W}}$$

$$\hat{=} -94,33 \text{ dBW}$$

$$SQNR = -3,01 \text{ dB} - N_{Q,out} = \mathbf{91,31 \text{ dB}}$$

$$ENOB = (SQNR - 1,76 \text{ dB}) / 6,02 \text{ dB/bit} = \mathbf{14,9 \text{ bit}}$$

- f) Die **Rauschbeiträge von Q_{mul}** sollen die Gesamt-*ENOB* aus Unterpunkt c) nur um 0,1 bit, also von *ENOB* = 15,2 bit auf 15,1 bit verringern:

$$ENOB = (SNR \text{ (dB)} - 1,76 \text{ dB}) / 6,02 \text{ dB/bit} \stackrel{!}{=} 15,1 \text{ bit} \Rightarrow SNR = 92,66 \text{ dB}$$

$$\Rightarrow N_{Q,out,max} = -SNR - 3,01 \text{ dB} = -95,67 \text{ dB} \hat{=} 2,71 \cdot 10^{-10}$$

$$= 7N_{Q,Mul,max} + N_{Q,ADC}/2 + N_{Q,DAC}$$

$$\Rightarrow N_{Q,Mul,max} = (2,71 \cdot 10^{-10} - 2,33 \cdot 10^{-10}) / 7 = 5,41 \cdot 10^{-12} 2^{-2WF_{mul}} / 12$$

$$\Rightarrow WF_{mul} = \lceil -0,5 \log_2(12 \cdot 5,41 \cdot 10^{-12}) \rceil = \mathbf{17 \text{ bit}}$$

Nach den Multiplizierern muss also mindestens auf das Format $Q_{mul}(0.17)$ requantisiert werden.

- g) **SQNR für Testsignal mit der Frequenz $F_1 = 1/4$:**

Da das Filter ein Halbbandfilter ist, ist die Signalamplitude am Ausgang des Filters nur noch 1/2, die Leistung dementsprechend nur noch 1/4 $\hat{=} -6,02 \text{ dB}$. Alle *SQNR*-Werte verringern sich also um 6,02 dB, die zugehörigen *ENOB*-Werte um 1 bit.

	(a) Q_DAC	(b) ADC	(c) Q_DAC + ADC	(d) 7 Q_Mul	(e) Alles
$N_{Q,out} / \text{W}$	$7,76 \cdot 10^{-11}$	$1,55 \cdot 10^{-10}$	$2,33 \cdot 10^{-10}$	$5,43 \cdot 10^{-10}$	$3,69 \cdot 10^{-10}$
$N_{Q,out} / \text{dBW}$	-101,10	-98,09	-96,33	-92,65	-94,33
$SQNR / \text{dB}$	98,09	95,08	93,32	89,64	91,31
$ENOB$	16,0	15,5	15,2	14,6	14,9

Tab. M6.1.: Zusammenfassung der Ergebnisse aus Aufgabe M6.3

M6.4. * DFT von Breitbandsignalen → Aufgabe 6.4

a) Signalleistungen

Die Nutzsignale sind Sinussignale, deren Leistung $S = A^2/2$ bei kohärenter DFT auf genau ein Frequenzbin (einseitiges Spektrum) bzw. zwei Frequenzbins (zweiseitiges Spektrum) fällt:

$$\begin{aligned} A_a &= \frac{5}{\sqrt{2}} \text{ V}_{rms} \quad \Rightarrow S_a = A_a^2 = 12,5 \text{ W} \quad \hat{=} \quad 10 \log_{10} \frac{12,5 \text{ W}}{1 \text{ W}} = 10,97 \text{ dBW} \\ A_b &= \frac{1}{\sqrt{2}} \text{ V}_{rms} \quad \Rightarrow S_b = A_b^2 = 0,5 \text{ W} \quad \hat{=} \quad 10 \log_{10} \frac{0,5 \text{ W}}{1 \text{ W}} = -3,01 \text{ dBW} \end{aligned}$$

Die Gesamtleistung von sinusförmigen Signalen ist die Summe der Einzelleistungen (siehe Aufgabe M3.6f):

$$S = S_a + S_b = 13 \text{ W} \hat{=} 11,14 \text{ dBW}$$

b) Rauschleistung und Signal-to-Noise Ratio

Der Rauschanteil von Listing 6.2 ist ein Gauß- oder normalverteilter Zufallsprozess mit Varianz (= AC-Leistung) $\sigma^2 = 0,01 \text{ V}^2 \hat{=} 10 \text{ mW} \hat{=} -20 \text{ dBW}$. Da der Prozess gleichanteilsfrei ist, ist das gleichzeitig die Gesamtleistung. Die Standardabweichung ist die Quadratwurzel der Varianz, sie beträgt $\sigma = 0,1 \text{ V}$.

Das Signal-to-Noise Ratio SNR erhält man aus dem Verhältnis der Gesamtsignal- zur Rauschleistung:

$$SNR = \frac{S}{N} = \frac{13 \text{ W}}{0,01 \text{ W}} = 1300 \hat{=} 11,14 \text{ dBW} - (-20 \text{ dBW}) = 31,14 \text{ dB}$$

c) Die Rauschleistungsdichte N' ist die Rauschleistung pro Hz:

$$\begin{aligned} N' &= N/f_S &= 10 \text{ mW}/5 \text{ kHz} &= 2 \mu\text{W}/\text{Hz} \text{ (zweiseitiges Spektrum)} \\ \text{bzw. } N' &= N/(f_S/2) &= 10 \text{ mW}/2,5 \text{ kHz} &= 4 \mu\text{W}/\text{Hz} \text{ (einseitiges Spektrum)} \end{aligned}$$

d) Die mittlere angezeigte Rauschleistung der DFT in der Simulation ist die Rauschleistung pro Frequenzbin:

$$\begin{aligned} N/\text{bin} &= N/N_{FFT} &= 10 \text{ mW}/2000 \text{ bins} &= 5 \mu\text{W/bin} \text{ (zweis. Spektrum)} \\ \text{bzw. } N/\text{bin} &= N/(N_{FFT}/2) &= 10 \text{ mW}/1000 \text{ bins} &= 10 \mu\text{W/bin} \text{ (eins. Spektrum)} \end{aligned}$$

```

1 # ... Ende der Import-Anweisungen
2 f_S = 5e3; T_S = 1. / f_S
3 N_FFT = 2000; N_FFT_2 = N_FFT / 2
4 t_max = N_FFT * T_S
5 f_1 = 1e3; a_1 = 1
6 NQ = 0.1
7
8 k_1 = N_FFT * f_1 / f_S # Index Frequenz 1
9
10 t = arange(0, t_max, T_S)
11 f_a = 1e3; f_b = 1e2;
12 A_a = 5; A_b = 1; NQ = 0.1
13 t = arange(0, t_max, T_S)
14 y = 1 + A_a * sin(2*pi*t*f_a) + A_b * cos(2*pi*t*f_b)
15 n = np.sqrt(NQ) * rnd.randn(len(t))

```

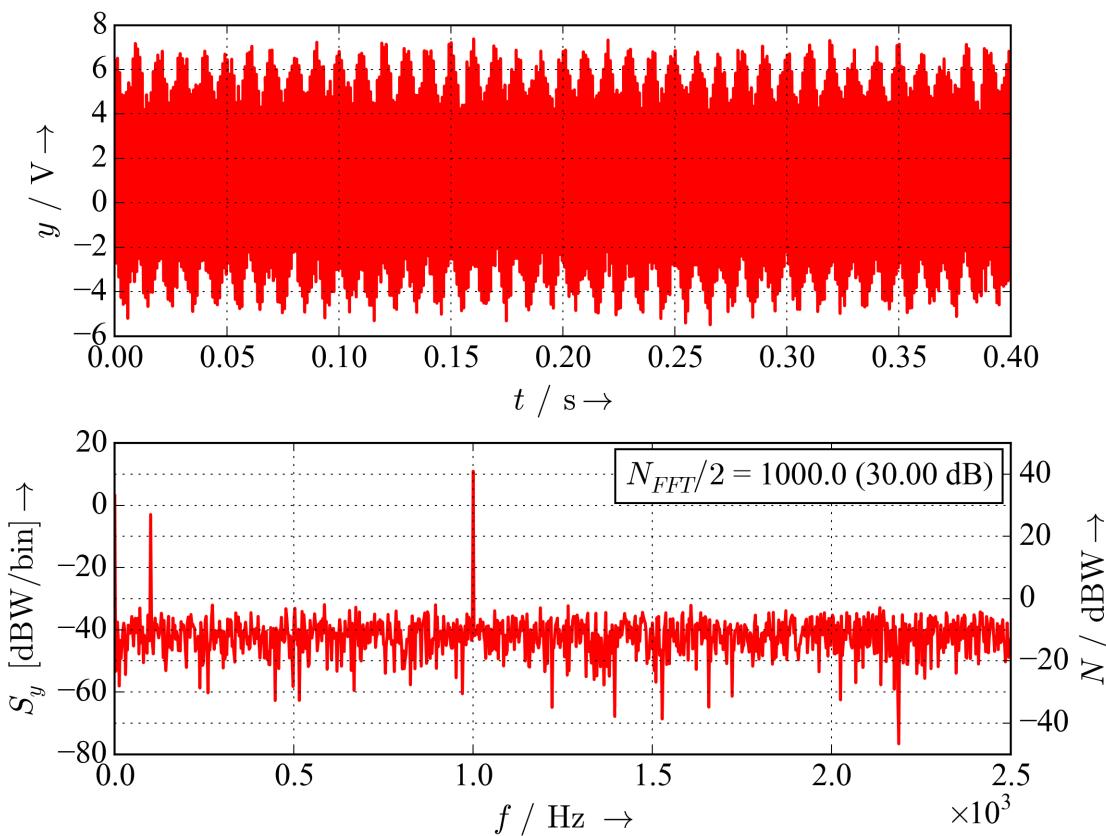


Abb. M6.1.: Einseitiges Spektrum von Schmal- und Breitbandsignalen

```

16 yn = y + n
17 N_t = sum(n*n)/len(t) # Rauschleistung, berechnet im Zeitbereich
18 print ('N =', N_t, 'W =', 10*log10(N_t), 'dBW')
19 print("N' =", N_t / N_FFT_2, 'W/Bin =', 10*log10(N_t / N_FFT_2), 'dBW/Bin')
20 print ('S1 =', a_1*a_1/2, 'W =', 10*log10(a_1*a_1/2), 'dBW')
21 # DFT mit korrekten AMPLITUDEN für EINSEITIGES Spektrum
22 Syn = abs(fft(yn,N_FFT))[0:N_FFT_2]/ N_FFT_2
23 Nn = Syn.copy()
24 Nn[k_1] = Nn[k_1] - a_1
25 Syn = Syn * Syn / 2 # Berechne Leistung (pro bin)
26 Nn = Nn * Nn / 2 # Berechne Leistung (pro bin)
27 N_f = sum(Nn); print(N_f)
28
29 f = fftfreq(N_FFT, T_S)[0:N_FFT_2]
30
31 print (Syn[k_1])
32 #
33 fig1 = figure(1); fig1.clf()
34 ax1 = fig1.add_subplot(211)
35 ax1.plot(t, yn)
36 ax1.set_xlabel(r'$t / \mathbf{\mathit{s}} \rightarrow$')
37
38 ax21 = fig1.add_subplot(212)
39 ax21.plot(f,10*log10(Syn),'r') # Leistung in dBW/bin
40 #ax21.plot(f,10*log10(Syn/N_FFT_2),'b')
41 ax21.set_xlabel(r'$f / \mathbf{\mathit{Hz}} \rightarrow$')
42 ax21.set_ylabel(r'$S_y / \mathbf{\mathit{[dBW/bin]}} \rightarrow$')
43 ylim21 = ax21.get_ylimits()
44 ax22 = ax21.twiny()
45 ax22.set_ylimits(ylim21 + 10*log10(N_FFT_2))
46 ax22.set_ylabel(r'$N / \mathbf{\mathit{[dBW]}} \rightarrow$')
47 ax22.text(0.98,0.9,r'$N_{FFT}/2$, = %s, (%.2f, {dB})$'
```

```
48         %(N_FFT_2, 10*log10(N_FFT_2)),  
49         fontsize=16, ha="right", va="center", linespacing=1.5,  
50         transform = ax22.transAxes,  
51         bbox=dict(boxstyle="square", fc='white'))  
52 fig1.tight_layout(pad = 0.8, h_pad = 0.3)  
54 plt.show()
```

Lst. M6.1: DFT eines Breitbandsignals, verbesserte Lösung
(NOI/NOI_DFT_wideband_ML_py.py)

M7. SMP: Abtastung und Downsampling

M7.1. * Spektren abgetasteter Signale → Aufgabe 7.1

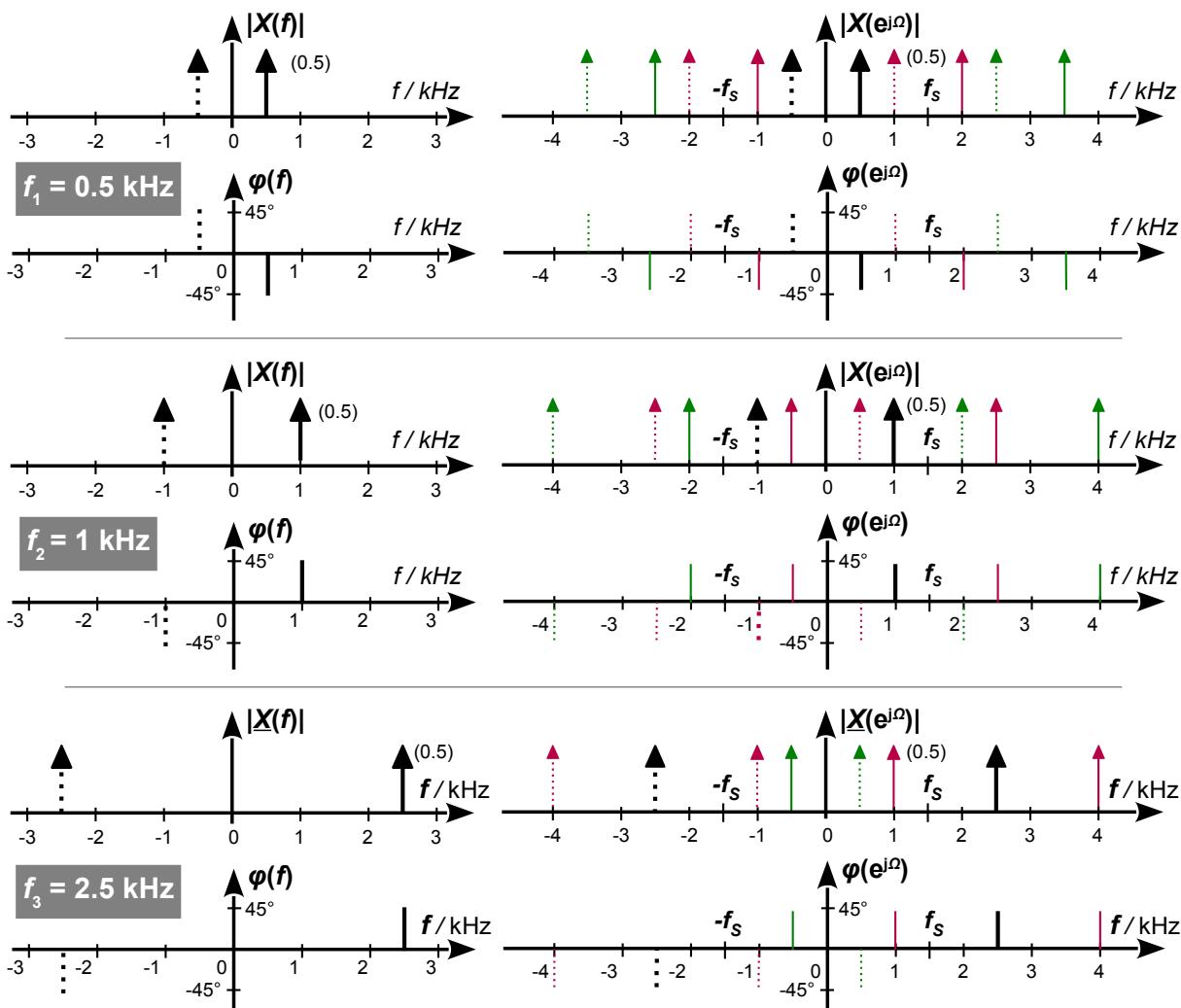


Abb. M7.1.: Amplituden- und Phasenspektren von zeitkontinuierlichen (links) und mit $f_s = 1500 \text{ Hz}$ abgetasteten (rechts) Signalen zu Aufgabe 7.1. Gezeichnet wurden nur die mit $\pm 2f_s, \pm f_s$ und $f = 0$ modulierten Produkte.

Die Spektren des abgetasteten Signals werden konstruiert, indem man jede Linie des Betrags- bzw. Phasenspektrums um $k f_s$ mit $k = \dots, -2, -1, 0, 1, 2, \dots$ verschiebt.

Anmerkung:

Frequenzen und Nullphasen der analogen Signale wurden hier so gewählt, dass sich gleiche Abtastwerte im Zeitbereich (Abb. M7.2) und daher auch gleiche Abtastspektren (Frequenzbereich) für alle drei Signale ergeben. Bei den Signalen mit 1000 Hz und 2500 Hz findet Aliasing

statt! „Fehlende“ Linien in Abb. M7.1 können durch zusätzliche Abtastprodukte aufgrund $\pm 3f_S$ ergänzt werden.

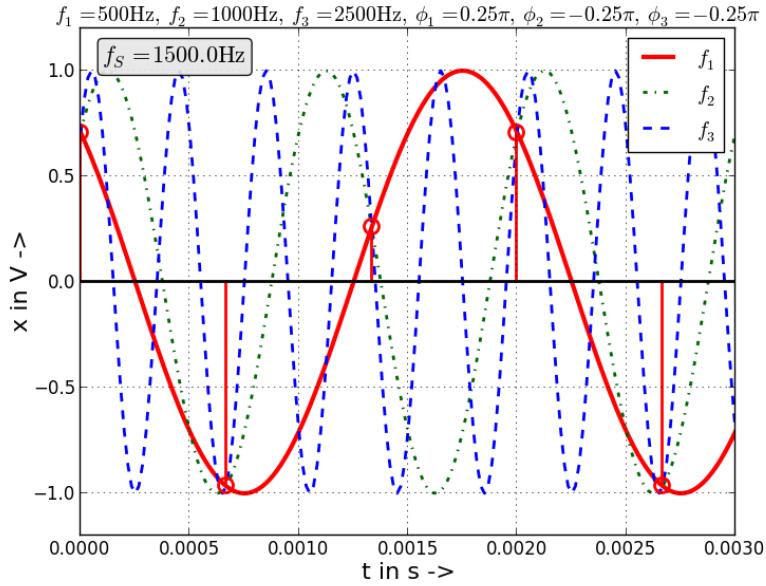


Abb. M7.2.: Zeitkontinuierliche Signale mit identischen Abtastwerten zu Aufgabe 7.1.

M7.2. Analog-Digital-Wandlung mit Oversampling → Aufgabe 7.2

- a) Das **maximale SQNR bei Nyquist-Wandlung** ist gegeben durch

$$SQNR_{max} = (6,02W + 1,76) \text{ dB} = 49,92 \text{ dB} \text{ für } N = 8.$$

- b) Das **maximale SQNR mit Oversampling** ist gegeben durch

$$\begin{aligned} SQNR_{max}(OSR) &= \frac{S}{f_N N'_Q} = \frac{S}{N_Q} \frac{f_S/2}{f_N} = SQNR_{max} \frac{f_S/2}{f_N} = SQNR_{max} OSR \\ &\Rightarrow SQNR[\text{dB}] = SQNR_{max}[\text{dB}] + 10 \log_{10} OSR \\ &\Rightarrow SQNR(OSR = 10) = 59,92 \text{ dB} \quad \hat{=} \quad ENOB = (59,9 - 1,8)/6,0 = 9,7 \\ &\Rightarrow SQNR(OSR = 50) = 66,91 \text{ dB} \quad \hat{=} \quad ENOB = (66,9 - 1,8)/6,0 = 10,9 \end{aligned}$$

- c) Das **maximale SQNR eines Sigma-Delta-Wandlers** erster Ordnung mit einem Bit ist gegeben durch

$$SQNR_{\Sigma\Delta,max} = -6,6 + 30 \log_{10} OSR = 23,4 \text{ dB } (OSR = 10) \text{ bzw. } 44,4 \text{ dB } (OSR = 50)$$

und für einen Wandler zweiter Ordnung durch

$$SQNR_{\Sigma\Delta,max} = -6,6 + 50 \log_{10} OSR = 43,4 \text{ dB } (OSR = 10) \text{ bzw. } 78,3 \text{ dB } (OSR = 50)$$

M7.3. * Analog-Digital-Wandlung mit Oversampling und nachfolgender Dezimation → Aufgabe 7.3

- a) Das Amplitudenspektrum des Signals wird nicht verfälscht bei der Reduktion der Abtastrate auf $f_{S,2}/R$, da das analoge Signal (und damit auch das Basisband des abgetasteten Signals) auf $B = 12 \text{ kHz} < f_{S2}/(2R)$ bandbegrenzt ist.

Zeitsignal und Phasenspektrum beider Signale sind aber nur dann identisch, wenn bei der Dezimation jeweils das gleiche Sample verwendet wird wie bei der Abtastung mit der niedrigen Abtastfrequenz f_{S1} . Andernfalls tritt ein Zeitversatz zwischen den abgetasteten Sequenzen auf.

- b) Maximaler Downsamplingfaktor R_{max} : Ein analoges AA-Filter muss die Bandbreite des analogen Signals vor der Abtastung auf $f_{S1}/2$ begrenzen, damit das Signal nicht durch Aliasing bei der Abtastung verfälscht wird. Wird die Abtastrate des Signals anschließend um den Faktor R verringert (Resampling, Dezimation), entspricht das einer neuerlichen Abtastung mit $f_{S2}/2 = f_{S1}/(2R)$. Daher muss die Bandbreite *vor der Dezimation* geringer als $f_{S2}/2$ sein:

$$\frac{f_{S1}}{2R_{max}} > B \Leftrightarrow R_{max} = \left\lfloor \frac{f_{S1}}{2B} \right\rfloor = 41$$

- c) Digitales Anti-Aliasfilter: Vor der Verringerung der Abtastrate auf $f_{S,2} = f_{S,1}/R$ muss ggf. mit einem digitalen Filter die Bandbreite des abgetasteten Signals auf $f_{S,2}/2 = f_{S,1}/(2R)$ begrenzt werden.

- d) Überabtastung um I mit nachfolgender Dezimation hat folgende Vorteile:

Entspannter AA-Filterentwurf: Die Bandbreite des analogen Signals muss nur auf $1fs/2$ begrenzt werden, hierfür kann ein Filter mit deutlich breiterem Übergangsbereich und daher geringerer Ordnung verwendet werden. Vor der Verringerung der Abtastrate muss u.U. die Bandbreite des abgetasteten Signals reduziert werden, um Aliasing bei der Dezimation zu vermeiden.

Verbessertes SQNR: Das Quantisierungsrauschen $q^2/12$ verteilt sich bei Abtastung mit erhöhter Abtastrate über einen breiteren Frequenzbereich, so dass die Rauschleistungsdichte sinkt. Durch Begrenzung der Bandbreite in einem digitalen Filter (unbedingt notwendig!) wird die Gesamtrauschleistung verringert und so das *SQNR* verbessert.

M7.4. * Zweistufige Dezimation → Aufgabe 7.4

In dieser Aufgabe geht es darum, die Abtastrate eines zeitdiskreten Signals in zwei Stufen von f_{S1} auf $f_{S3} = f_{S2}/2 = f_{S1}/4$ zu reduzieren. Die obere Grenze des Nutzbands ist hier $f_N = f_{S3}/3 = f_{S2}/6 = f_{S1}/12$, das ist auch die *minimale* Frequenz für das Durchlassband beider Tiefpässe, $f_N \leq f_{DB}$, da ansonsten das Nutzband vom Tiefpassfilter „angeschnitten“ wird. Der Übergangsbereich zwischen Durchlass- und Sperrband soll möglichst breit sein, damit man einen möglichst entspannten Tiefpassentwurf (= niedrige Ordnung) erhält. Die Eckfrequenz für das Durchlassband soll daher möglichst gering gewählt werden (im Extremfall $f_N = f_{DB}$), die Eckfrequenz des Sperrbands möglichst groß.

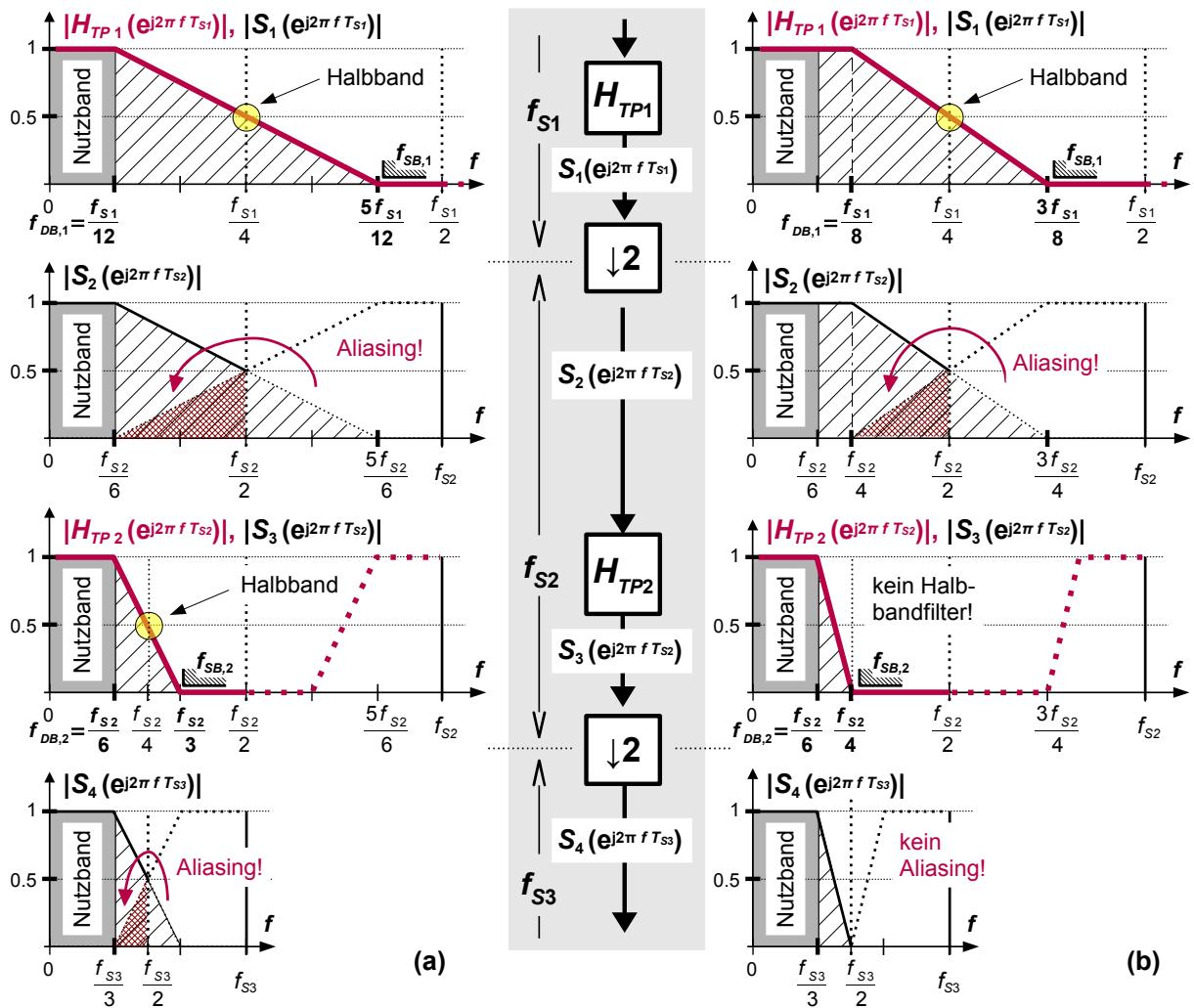


Abb. M7.3.: Zweistufige Dezimation, mit (a) und ohne (b) Aliasing ins Basisband am Ausgang zu Aufgabe 7.4a)

a) Realisierung mit zwei dezimierenden Halbbandfiltern (Abb. M7.3a)

Da es hier keine zusätzlichen Vorgaben gibt, soll die Eckfrequenz für das Durchlassband für beide Tiefpassfilter $f_N = f_{DB} = f_{DB1} = f_{DB2}$ gewählt werden (also möglichst niedrig) damit der Übergangsbereich möglichst breit wird.

TP₁ arbeitet mit f_{S1} . Um Aliasing bei der nachfolgenden ersten Dezimation um $R = 2$ vollständig zu vermeiden, müsste die Grenzfrequenz des Sperrbands bei $f_{SB1} = f_{S1}/4 = f_{S2}/2$ liegen. Hier ist jedoch Aliasing ins Basisband bis an den Rand des Nutzbands gestattet, daher kann die Grenzfrequenz des Sperrbands höher gewählt werden, $f_{SB1} = f_{S1}/2 - f_{DB1} = 5/12f_{S1}$. Mit dieser Wahl sind die Eckfrequenzen von DB und SB symmetrisch zu $f_{S1}/2$ und man kann ein Halbbandfilter verwenden. Frequenzkomponenten mit $f_{S2}/2 < f < f_{SB1}$ werden dann bei der Dezimation $f_{S2} - f > f_N$ abgebildet (eng schraffiert gekennzeichnet).

Entsprechendes gilt für **TP₂**: Hier ist $f_{SB2} = f_{S2}/2 - f_N = f_{S2}/3$. Man sieht, dass Halbbandfilter gut geeignet sind als Anti-Aliasfilter für eine Dezimation um $R = 2$: Mit $f_{DB} = f_N$ wird ja gleichzeitig $f_{SB} = f_S/2 - f_{DB}$ festgelegt, dies ist aber die maximale Grenzfrequenz, bei der gerade noch kein Aliasing ins Nutzband auftritt.

Nachteilig bei dieser Lösung ist, dass noch ein drittes Filter (hier nicht gezeichnet) benötigt wird, das die Aliasing-Komponenten im Frequenzbereich zwischen $f_N = f_{S3}/3$ und f_{S3} „aufräumt“. Dieses Filter müsste extrem steilbandig sein, da ja $f_{S3}/3$ auch die obere Grenze des Nutzbandes ist. In der Praxis wird man daher die Grenzfrequenzen der Sperrbänder beider Dezimationsfilter etwas niedriger (und damit automatisch die der Durchlassbänder etwas höher) legen müssen.

b) Realisierung mit dezimierendem Halbbandfilter und FIR-Filter (Abb. M7.3b)

Die Nachteile der vorigen Lösung werden vermieden, wenn man Aliasing ins Nutzband bei der zweiten Dezimation verbietet und dafür die Filterübergänge etwas steiler auslegt. Am besten beginnt man die Aufgabe vom Ausgang des Multiratenfilters her:

Wenn kein Aliasing ins Basisband bei der Dezimation auf f_{S3} auftreten darf (unterstes Bild von Abb. M7.3b), muss $f_{SB,2} \leq f_{S3}/2 = f_{S2}/4$ sein, mit Hinblick auf minimale Steilheit des Filters also $f_{SB,2} = f_{S2}/4$. Aus dem gleichen Grund ist $f_{DB,2} = f_N = f_{S2}/4$. Der Frequenzbereich unterhalb $f_{S2}/4$ wird in der zweiten Stufe nicht unterdrückt, daher darf in der ersten Stufe des Filters kein Aliasing in den Bereich $0 \dots f_{S2}/4$ auftreten. Die Bedingung für die maximal zulässige Eckfrequenz des Sperrbandes $f_{SB,1}$ lautet daher:

$$f_{S2} - f_{SB,1} = \frac{f_{S1}}{2} - f_{SB,1} = \frac{f_{S2}}{4} = \frac{f_{S1}}{8} \Leftrightarrow f_{SB,1} = \frac{3f_{S1}}{8}$$

Aufgrund der vorgegebenen Symmetrie des Halbbandfilters folgt daraus automatisch $f_{DB,1} = f_{SB,1} - f_{S1}/4 = f_{S1}/8$

M7.5. Multiraten-Tiefpassfilter → Aufgabe 7.5

- a) In Abb. M7.4 ist der **Amplitudengang der Kaskade** aus TP₁ und TP₂ im Bereich $0 \dots f_{S1}/2 = 3f_{S2}/2 = 3f_{S3}$ dargestellt. Hierfür wurden die Amplitudengänge von TP₁ und TP₂ im gleichen Frequenzmaßstab aufgetragen, aufgrund der geringeren Abtastfrequenz sind bei TP₂ im Bereich $f_{S1}/6 \dots f_{S1}/2 (\hat{=} f_{S2}/2 \dots 3f_{S2}/2)$ Wiederholspektren eingetragen. In diesem Bereich sollte daher der Begriff „Amplitudengang“ nur mit Vorsicht benutzt werden: Beim ersten Downsampling werden Frequenzkomponenten aus diesem Bereich in das Basisband $0 \dots f_{S2}/2 = f_{S1}/6 = 8$ kHz abgebildet, nach der zweiten Downsamplingstufe ist das Basisband sogar auf $0 \dots f_{S3}/2 = f_{S2}/4 = f_{S1}/12 = 4$ kHz begrenzt.

Man kann jedoch den „Amplitudengang“ auf folgende Art benutzen: Zunächst ermittelt man mit Hilfe des kaskadierten Frequenzgangs (Abb. M7.4d) die Dämpfung, die eine bestimmte Frequenzkomponente erfährt. Danach errechnet man (oder ermittelt mit dem „Faltungsschema“ Abb. M7.4f) auf welche Ausgangsfrequenz die Eingangsfrequenz abgebildet wird.

- b) Der **Testton bei $f_2 = 7/24f_{S1} = 14$ kHz** wird von TP₁ um 40 dB gedämpft. Da die Frequenz außerhalb des Basisbands von f_{S2} liegt, wird sie auf $\tilde{f}_2 = f_{S2} - f_2 = 1/24f_{S1} = 2$ kHz abgebildet. Bei dieser Frequenz hat TP₂ eine Dämpfung von 0 dB, daher wird bei dieser Frequenz Aliasing insgesamt nur um 40 dB gedämpft (siehe Tab. M7.1) - .
- c) Ein **verbesserter Filterentwurf** muss zwei Probleme des bestehenden Entwurfs beseitigen:

f_{in}	f_{DB}	f_{SB}	f_1	f_2	f_3
	2 kHz	4 kHz	8 kHz	14 kHz	16 kHz
f_{out} (kHz)	2	4	0	2	0
	A_{DB}	A_{SB}		$A_{al,i}$	
Soll (dB)	0,1	80	80	80	80
Ist (dB)	0	53	80	40	40

Tab. M7.1.: Eigenschaften der Filterkaskade aus Aufgabe 7.5

Die **Aliasdämpfung** ist nicht bei allen Eingangsfrequenzen ausreichend: Die Frequenz $f_2 = 14$ kHz wird z.B. beim Downsampling auf $\tilde{f}_2 = f_{S2} - f_2 = 2$ kHz und damit ins Durchlassband von TP₂ abgebildet. Die gewünschte Alias-Unterdrückung von 80 dB bei Eingangsfrequenzen $f_{in} \geq f_{S2} - f_{DB} = 14$ kHz muss daher TP₁ alleine erreichen. Daraus folgt als zusätzliche Spezifikation für TP₁ $A_{SB,1a} = 80$ dB bei $f_{SB,1a} = 14$ kHz. Da die hohe Sperrdämpfung erst bei höheren Frequenzen gefordert ist, ist ein Equiripple-Entwurf für TP₁ vermutlich nicht optimal. Bessere Ergebnisse (= geringere Ordnung) wird ein gefensterter Entwurf liefern, dessen Dämpfung zu höheren Frequenzen hin zunimmt.

Die **Gesamtdämpfung der Kaskade** bei $f_{SB} = 4$ kHz ist ebenfalls zu gering. Das lässt sich auf verschiedene Arten verbessern:

1. Man kann die Eckfrequenz des Sperrbandes von TP₁ auf $f_{SB,1} = 4$ kHz ($\hat{=} F_{SB,1} = 1/12$) reduzieren, dort genügt weiterhin $A_{SB,1} = 40$ dB.
2. Alternativ kann die Sperrdämpfung von TP₂ bei $f_{SB,2} = 4$ kHz ($\hat{=} F_{SB,2} = 1/4$) auf $A_{SB,2} = 67$ dB erhöht werden.

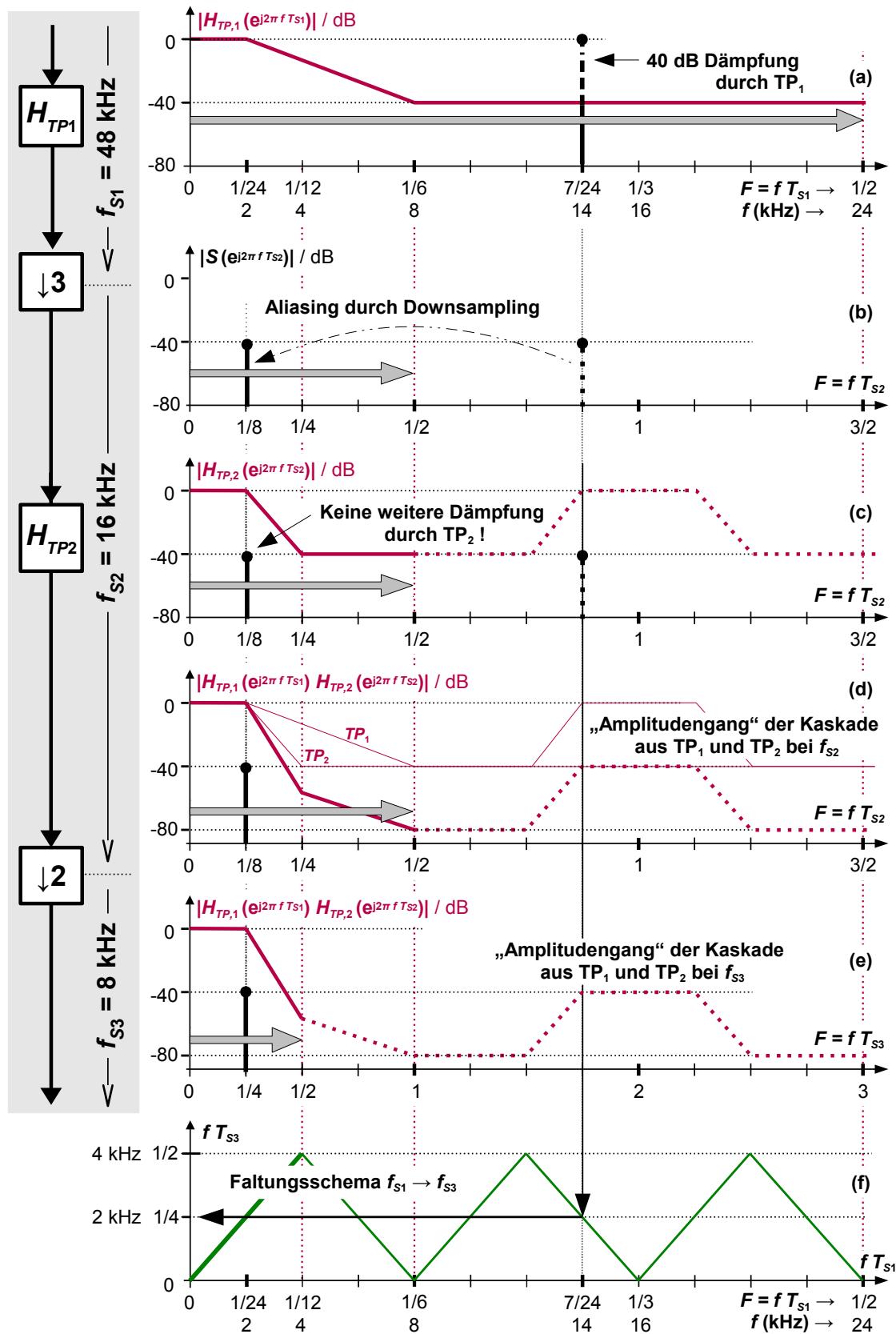


Abb. M7.4.: Frequenzgänge der Multiraten-Filterkaskade zu Aufgabe 7.5. Graue Pfeile symbolisieren die jeweiligen Basisbänder.

M8. INP: Upsampling, Interpolation und Digital-Analog Wandlung

M8.1. * Wiederholspektren und Images → 8.1

In Abb. 8.1) sehen Sie ...

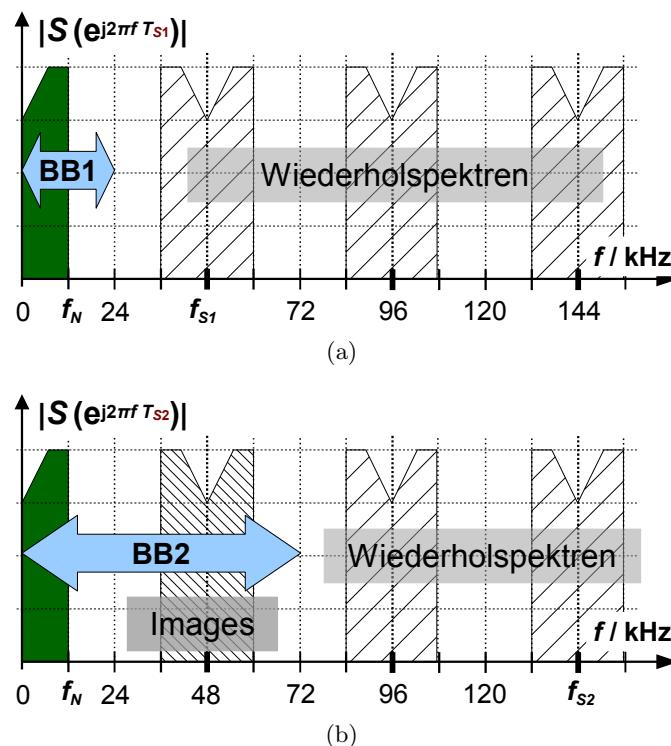
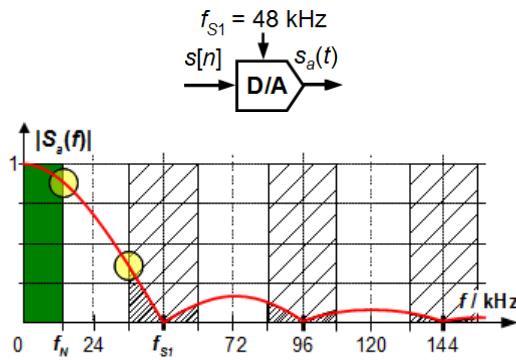


Abb. M8.1.: Spektrum von $s[n]$ (a) und nach Abtastratenerhöhung um $I = 3$ (b)

M8.2. * Verschiedene Arten der Digital-Analog-Wandlung → Aufgabe 8.2

- a) **Direkte D/A-Wandlung des digitalen Audiosignals mit Abtastfrequenz f_{s1}**
 Bei direkter Wandlung wird das DAC-Ausgangssignal jeweils für die Periode T_{s1} gehalten, das Wiederholspektrum des digitalen Signals (schwach schraffiert) wird dadurch mit der Funktion $\text{si}(\pi f T_{s1})$ gewichtet, es bleibt das stark schraffierte Spektrum übrig.



$$a_N \quad \text{si}(\pi f_N/f_{S1}) = \text{si}(\pi/4) = 0,90 \\ \equiv -0,91 \text{ dB}$$

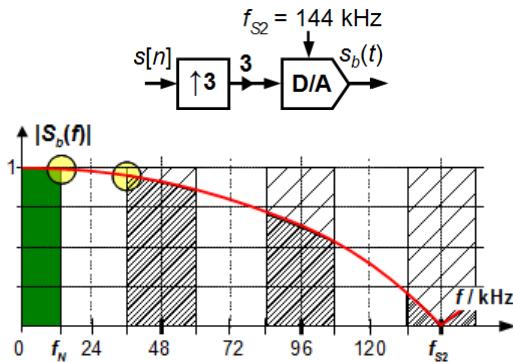
- 8cm- 8cm $f_{im} \quad f_{S1} - f_N = 36 \text{ kHz}$

$$a_{im} \quad \text{si}(\pi f_{im}/f_{S1}) = \text{si}(3\pi/4) = 0,30 \\ \equiv -10,5 \text{ dB}$$

b) Überabtastung um den Faktor $I = 3$ ohne digitale Interpolation

Hier wird das digitale Audiosignal zunächst um den Faktor 3 überabgetastet, zwischen den Signalwerten werden dabei Nullen eingefügt (Nullenstopfen). Dadurch werden die Wiederholspektren bis zur neuen Nyquistfrequenz $f_S/2$ „real“ und könnten separat gefiltert werden (was aber hier nicht getan wird). Außerdem wird das gesamte Spektrum durch das Nullenstopfen um den Faktor 3 gedämpft, dies wird durch ein Verstärkungsglied wieder ausgeglichen.

Die D/A Wandlung bei der dreifachen Abtastfrequenz $f_{S2} = 3f_{S1}$ setzt die digitalen Werte in dreimal schmalere Rechteckpulse um, die eine si-Funktion als Spektrum haben mit der ersten Nullstelle bei $1/f_{S2}$.



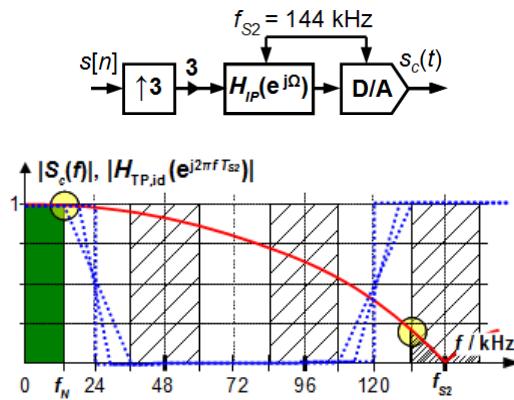
$$a_N \quad \text{si}(\pi f_N/f_{S2}) = \text{si}(\pi/12) = 0,99 \\ \equiv -0,10 \text{ dB}$$

- 7.5cm- 7.5cm $f_{im} \quad f_{S1} - f_N = 36 \text{ kHz}$

$$a_{im} \quad \text{si}(\pi f_{im}/f_{S2}) = \text{si}(\pi/4) = 0,90 \\ \equiv -0,91 \text{ dB}$$

c) Überabtastung um den Faktor $I = 3$ mit idealer digitaler Interpolation

Ideale Interpolation heißt, dass im neuen Basisband $0 \dots f_{S2}/2 = If_{S1}/2$ alle Images des alten Basisbands $0 \dots f_{S1}/2$ unterdrückt werden. Das Nutzband $0 \dots f_N < f_{S1}/2$ muss das Interpolationsfilter ohne Dämpfung passieren können, bei der niedrigsten Imagefrequenz $f_{im} = f_{S1} - f_N$ ist die Dämpfung schon unendlich groß. Die gestrichelten Linien im Bild unten erfüllen alle die Bedingung für ein ideales Interpolationsfilter. Die minimale Eckfrequenz des Durchlassbands beträgt hier $f_{DB} = f_N = 12 \text{ kHz}$, die maximale Eckfrequenz des Sperrbands $f_{SB} = f_{im}$.



$$\begin{aligned} a_N & \quad \text{si}(\pi f_N/f_{S2}) = \text{si}(\pi/12) = 0,99 \\ & \equiv -0,10 \text{ dB} \\ f_{im} & \quad f_{S2} - f_N = 132 \text{ kHz} \\ a_{im} & \quad \text{si}(\pi f_{im}/f_{S2}) = \text{si}(11\pi/12) \equiv -20,9 \text{ dB} \end{aligned}$$

d) Überabtastung um den Faktor $I = 3$ mit digitalem Zero-Order Hold

Die Wiederholung des letzten Abtastwerts (Zero-Order Hold, ZOH) beim Upsampling um $I = 3$ anstelle von Nullenstopfen lässt sich signaltheoretisch beschreiben durch Nullenstopfen, gefolgt von einem Moving-Average (MA) Filter der Länge $L = I = 3$,¹ das mit $f_{S2} = 3f_{S1}$ getaktet wird (Abb. M8.2).

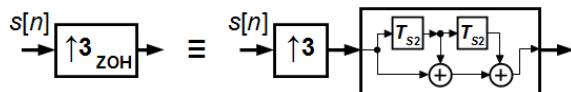
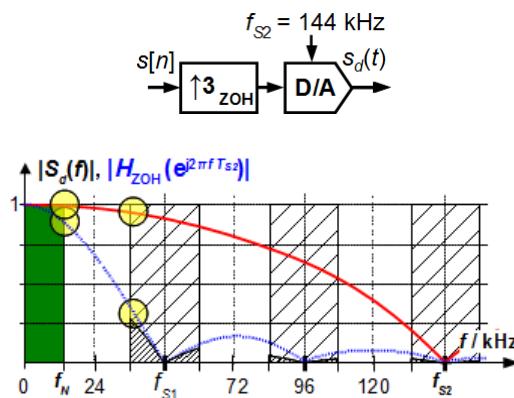


Abb. M8.2.: Ersatzschaltbild für Interpolation mit ZOH zu Aufgabe 8.2 d)

Das MA-Filter hat eine Verstärkung von 3 bei niedrigen Frequenzen und übernimmt damit gleichzeitig die Amplitudenkorrektur nach dem Nullenstopfen. Die Filtercharakteristik eines MA-Filters der Länge $N = 3$ ist alles andere als ideal (vgl. Aufgabe 2.7), sie wird beschrieben durch die Dirichletfunktion $\text{di}_N(\Omega)$ („periodische si-Funktion“):

$$|H_I(e^{j2\pi f T_{S2}})| = 3 \text{ di}_3(2\pi f T_{S2}) = \frac{\sin 3\pi f T_{S2}}{\sin \pi f T_{S2}} = \frac{\sin \pi f T_{S1}}{\sin \pi f T_{S1}/3} \approx 3 \text{ si } \pi f T_{S1}$$

Im neuen Basisband $0 \dots f_{S2}/2 = 3f_{S1}/2$ werden jetzt nicht mehr alle Images des alten Basisbands $0 \dots f_{S1}/2$ unterdrückt, das Nutzband $0 \dots f_N < f_{S1}/2$ wird dagegen bei der Eckfrequenz $f_{DB} = f_N = 12 \text{ kHz}$ durch das „Interpolationsfilter“ schon beträchtlich um $a_{N,I} = 0,9 \text{ dB}$ gedämpft.



$$\begin{aligned} a_{N,I} & \quad \text{di}_3(2\pi f_N/f_{S2}) \approx \text{si}(\pi f_N/f_{S1}) \\ & = \text{si}(\pi/4) \equiv -0,91 \text{ dB} \\ a_{N,Z} & \quad \text{si}(\pi f_N/f_{S2}) = \text{si}(\pi/12) \equiv -0,10 \text{ dB} \\ a_N & \quad a_{N,I} a_{N,Z} = 0,89 \equiv -1,01 \text{ dB} \\ f_{im} & \quad f_{S1} - f_N = 36 \text{ kHz} \\ a_{im,I} & \quad \text{di}_3(2\pi f_{im}/f_{S2}) \approx \text{si}(\pi f_{im}/f_{S1}) \\ & = \text{si}(3\pi/4) \equiv -10,5 \text{ dB} \\ a_{im,Z} & \quad \text{si}(\pi f_{im}/f_{S2}) = \text{si}(\pi/4) \equiv -0,91 \text{ dB} \\ a_{im} & \quad a_{im,I} a_{im,Z} = 0,27 \equiv -11,4 \text{ dB} \end{aligned}$$

¹entsprechend einer Ordnung von $N = 2$

e) Vergleicht man die Spektren aus a) - d) fallen die folgenden Unterschiede und Gemeinsamkeiten auf:

- Bei der direkten D/A-Wandlung (a) und beim Oversampling mit digitalem ZOH (d) wird das Durchlassband am stärksten gedämpft.
- Bei der D/A-Wandlung mit Oversampling ohne Interpolationsfilter (b) werden die Images am schlechtesten unterdrückt.
- Bei der D/A-Wandlung mit Oversampling und perfekter Interpolation (c) werden die Images am besten unterdrückt, nur hier tauchen die ersten Images erst bei $f_{S2} - f_N$ auf. Die Dimensionierung des analogen Rekonstruktionsfilters ist bei (c) daher besonders einfach.
- Die D/A-Wandlung mit Oversampling und digitalem ZOH (d) lässt sich vorteilhaft einsetzen, wenn das Signal $s[n]$ selbst schon oversampled ist ($f_N \ll f_{S1}$): Man benötigt kein digitales Interpolationsfilter wie bei (c) und hat eine bessere Image-Unterdrückung als bei (a) und (b). Die Dämpfung des Nutzbandes fällt nicht so sehr ins Gewicht, da si $\pi f_N T_{S1} \approx 1$.

M8.3. * Ideales Interpolationsfilter → Aufgabe 8.3

Ein ideales Interpolationsfilter (iI)...

- filtert *nicht* die Wiederholspektren aus dem Basisband - im Basisband gibt es definitivonsgemäß keine Wiederholspektren. Da sich die Spektren zeitdiskreter Signale mit der Abtastfrequenz wiederholen (-> Wiederholspektren), ist es unmöglich Wiederholspektren zu beeinflussen (ohne gleichzeitig das Basisband zu ändern)! Images, die bei einer vorangegangenen Erhöhung der Abtastfrequenz entstanden sind, werden von einem iI aus dem Basisband entfernt - **[falsch]**.
- filtert *nicht* die Wiederholspektren aus dem Nutzband, da das Nutzband ein Teil des Basisbands ist und ... (s.o.). Images können u.U. im Nutzband auftreten, diese werden dann vom iI entfernt - **[falsch]**.
- hat möglichst weit auseinanderliegende Eckfrequenzen von Sperr- und Durchlassband - das sollte zwar so sein, damit man einen entspannten Filterentwurf bekommt, ist aber keine Voraussetzung - **[falsch]**.
- entfernt alle Images - das ist die wichtigste Eigenschaft eines iI - **[stimmt]!**
- lässt das DB unverändert - **[stimmt]**.
- hat keinen Einfluss auf Wiederholspektren - **[stimmt]** (s.o.)!
- lässt Images nicht unverändert - im Gegenteil, Images sollen vollständig unterdrückt werden - **[falsch]**.
- verhindert *nicht* Rückfaltungen ins Basisband; Rückfaltungen (Aliasing) treten nur beim Downsampling auf und müssen durch ein Anti-Aliasingfilter vor dem Downsampling verhindert werden - **[falsch]**.
- verhindert Rückfaltungen ins Nutzband - **[falsch]** (s.o.).

- muss im Durchlassband die Verstärkung 1 haben? Nein, oft wird im Interpolationsfilter gleichzeitig die Signalamplitude korrigiert, die durch das Nullenstopfen reduziert wurde - [falsch].

M8.4. * Abtastratenerhöhung mit Nullenstopfen → Aufgabe 8.4

$$S[k] = \frac{1}{N} \sum_{n=0}^{N-1} s[n] e^{-j2\pi kn/N} = \frac{1}{3} \sum_{n=0}^2 s[n] e^{-j2\pi kn/3}$$

M8.5. * Oversampling DAC → Aufgabe 8.5

a) ZOH-Signal und Amplitudenspektrum ohne Oversampling (Abb. M8.3)

Das periodische Amplitudenspektrum $|X_1(e^{j\omega T_{S1}})|$ des zeitdiskreten Signals hat Linien mit $A = 2,5$ V bei $f = nf_{S1} \pm f_{sig}$. Die kritischen Frequenzen bei der Rekonstruktion des analogen Signals sind die höchste Frequenz des Nutzbandes (hier: $f_{sig} = 3,3$ kHz), deren Dämpfung durch den si-Frequenzgang der Zero-Order-Hold (ZOH)-Stufe an anderer Stelle ausgeglichen werden muss und die niedrigste Imagefrequenz (hier: $f_{S1} - f_{sig} = 4,7$ kHz), die unterdrückt werden muss (Tab. M8.1).

Allgemein ist der Frequenzgang eines ZOH-Gliedes gegeben durch

$$\frac{H_{ZOH}(f)}{H_{ZOH}(0)} = \text{si}(\pi f/f_S) = \frac{\sin \pi f/f_S}{\pi f/f_S},$$

die erste Nullstelle tritt also bei $f = f_S$ auf.

Frequenz in kHz	3,3	4,7	11,3	12,7
Dämpfung	0,743	0,521	0,217	0,193
Dämpfung in dB	2,58	5,66	13,28	14,29

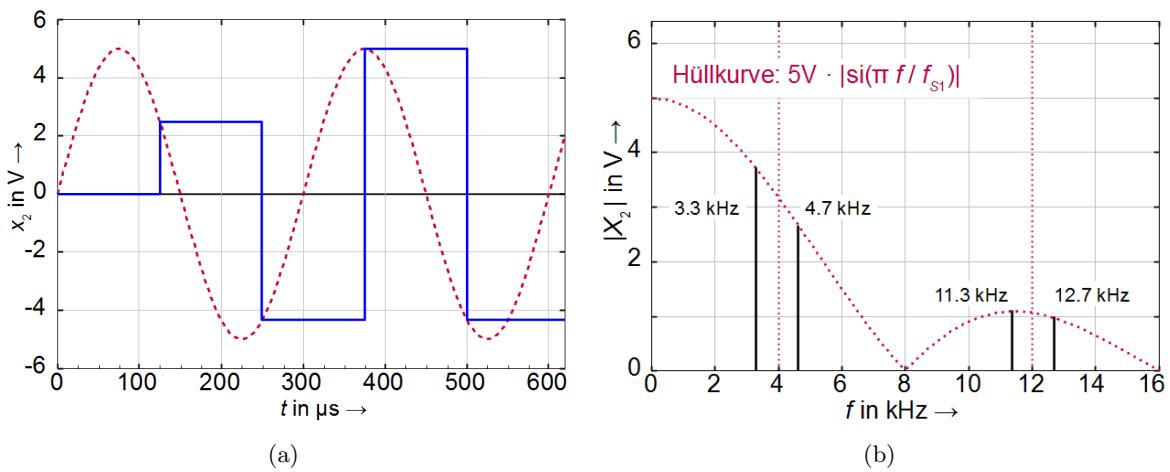
Tab. M8.1.: si-Dämpfungen für DAC ohne Oversampling (Abb. 8.2(a))

b) Spezifikationen für Interpolations-Tiefpass aus Abb. 8.2b)

Der digitale Interpolations-Tiefpass soll Images bei 4,7 kHz bzw. 11,3 kHz unterdrücken, damit das analoge Rekonstruktionsfilter diese Aufgabe nicht übernehmen muss. Der digitale Interpolations-Tiefpass muss eine Sperrdämpfung haben von

$$H_{int}(e^{j\omega T_{S2}}) = 20 \log_{10} \frac{5 \text{ V}}{50 \text{ mV}} = 40 \text{ dB} \quad \text{bei } f_{S1} - f_{sig} = 4,7 \text{ kHz}.$$

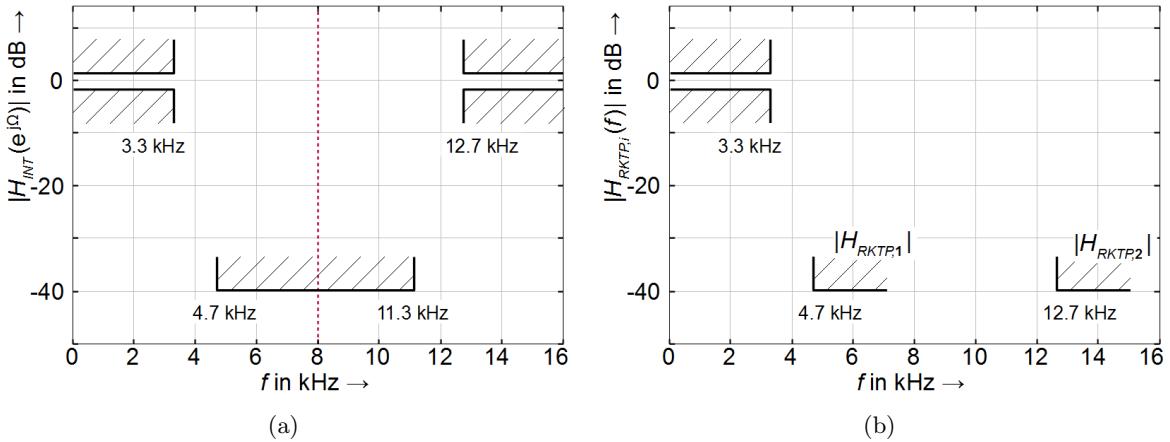
Halbband-Filter haben symmetrische Koeffizienten und sind daher linearphasig. Da außerdem jeder zweite Koeffizient bis auf den mittleren Null ist, können sie mit vergleichsweise geringem Hardware-Aufwand realisiert werden.

Abb. M8.3.: ZOH-Signal $x_2(t)$ und Amplitudenspektrum $|X_2(f)|$ zu Aufgabe 8.5a

Dafür haben Halbband-Filter enge Vorgaben für den Frequenzgang: Bei $|H(f = f_S/4)| = 1/2$ und $|H(f)| = 1 - |H(f_S/2 - f)|$ (jeweils normiert auf $|H(f = 0)| = 1$). In diesem Fall folgt damit z.B.

$$|H(f = f_{SB})| = -40 \text{ dB} \hat{=} 0,01 \Rightarrow |H(f = f_{DB})| = 1 - |H(f = f_{SB})| = 0,99 \hat{=} 0,087 \text{ dB}.$$

Aus der geforderten Sperrdämpfung von 40 dB folgt damit ein Ripple von 1% oder 0,087 dB im Durchlassband.

Abb. M8.4.: Spezifikationen für Interpolationsfilter H_{int} (a) und Rekonstruktionstiefpässe $H_{RKTP,i}$ (b) zu Aufgabe 8.5b und d)

c) ZOH-DAC mit Oversampling: (Abb. M8.5)

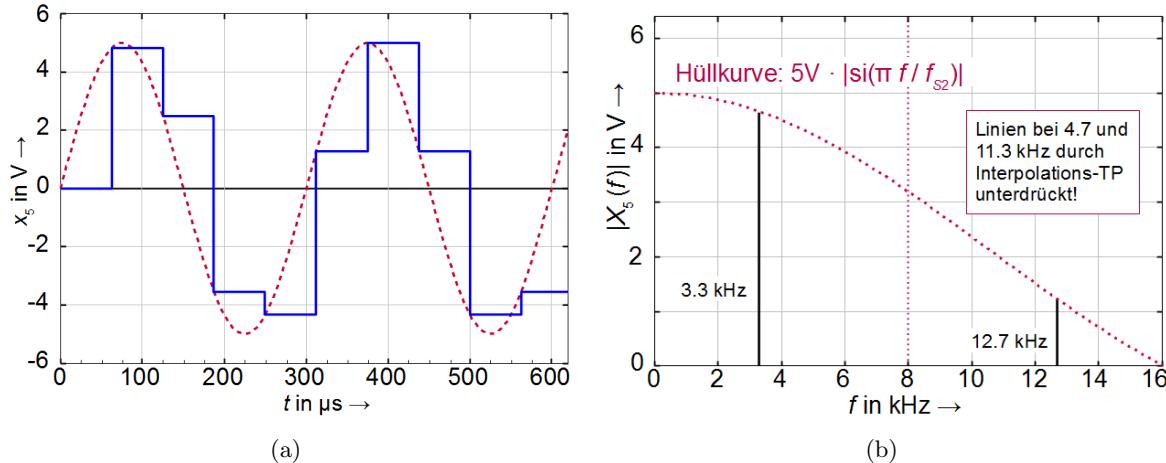
Durch die Erhöhung der Abtastfrequenz tritt das erste Image erst bei $f_{S2} - f_{sig} = 12,7 \text{ kHz}$ auf, auch die Dämpfung der Signalfrequenz fällt geringer aus (Tab. M8.2).

d) Spezifikationen für analoge Rekonstruktions-Tiefpässe aus Abb. 8.2a) und b):

Wie man in Abb. M8.4b) sehen kann, ist für den DAC mit Oversampling ($H_{RKTP,2}$) der Übergangsbereich zwischen Durchlass- und Sperrband breiter. Man kann also ein analoges Rekonstruktionsfilter mit geringerer Ordnung verwenden, um die geforderte Image-Unterdrückung zu erreichen.

Frequenz in kHz	3,3	12,7
Dämpfung	0,931	0,242
Dämpfung in dB	0,62	12,32

Tab. M8.2.: si-Dämpfungen für DAC mit Oversampling (Abb. 8.2(b))

Abb. M8.5.: ZOH-Signal $x_5(t)$ und Amplitudenspektrum $X_5(f)$ zu Aufgabe 8.5c

Die Images werden durch die ZOH-Funktion zusätzlich gedämpft (5,7 dB bzw. 12,3 dB), man kommt daher mit einer entsprechend geringeren Sperrdämpfung aus.

M8.6. Zweistufige Interpolation → Aufgabe 8.6

Der „Amplitudengang“ der Kaskade entspricht genau dem der zweistufigen Dezimation aus Aufgabe 7.4 bis auf die hier erforderliche Amplitudenkorrektur um den Faktor 2 bzw. 3.

Das Spektrum des Eingangssignals enthält im Basisband nur eine Signalkomponente mit der Frequenz $f_{sig,BB} = f_{S3}/4$. Diese Komponente ist im Wiederholspektrum periodisch f_{S3} vorhanden bei $f_{sig,wied} = \pm f_{sig,BB} \pm k f_{S3}$ mit $k = 0, 1, 2, \dots$.

Bei Erhöhung der Abtastfrequenz auf zunächst $f_{S,2} = 2f_{S,3}$ und dann auf $f_{S,1} = 6f_{S,3}$ werden Wiederholspektren im Frequenzbereich $\pm f_S/2$ „echt“, d.h. zu Images, die von den Filtern gedämpft werden.

f/f_{S3}	f/f_{S1}	Dämpfung nach TP2 in dB	Gesamtdämpfung am Ausgang in dB
1/4	1/24	0	0
1/2	1/12	40	$40 + 40/3 = 53,3$
3/4	3/24	40	$40 + 40 \cdot 2/3 = 66,7$
5/4	5/24	40	80
7/4	7/24	0	40
9/4	9/24	0	40
11/4	11/24	40	80

Tab. M8.3.: Dämpfung der Images im zweistufigen Interpolator (Aufgabe 8.6)

M9. SRC: Abtastratenwandlung

M9.1. * Einfache Abtastratenwandlung → Aufgabe 9.1

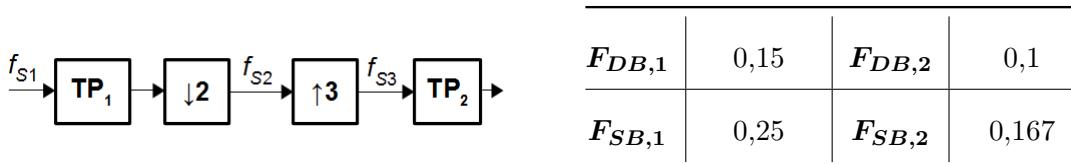
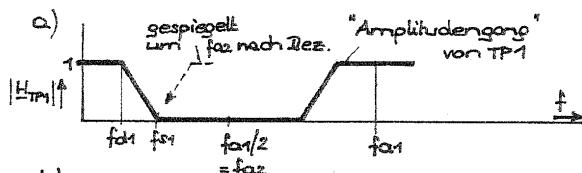
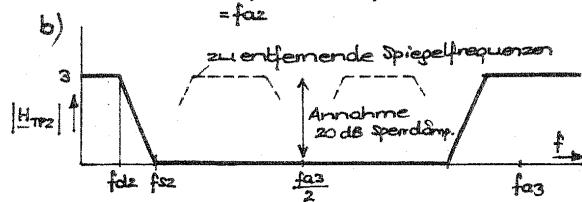


Abb. M9.1.: Synchrone Abtastratenwandlung zu Aufgabe 9.1



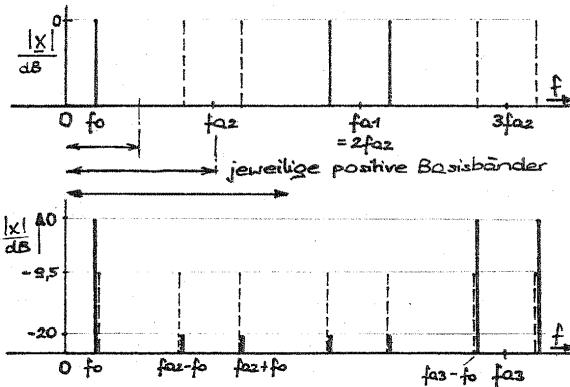
damit kein Aliasing nach Dezimierung auf f_{a2} auftritt, muß

$$f_{S1} = \frac{f_{a2}}{2} = \frac{f_{a1}}{4} \text{ sein } \Rightarrow \frac{f_{S1}}{f_{a1}} = \frac{1}{4}$$



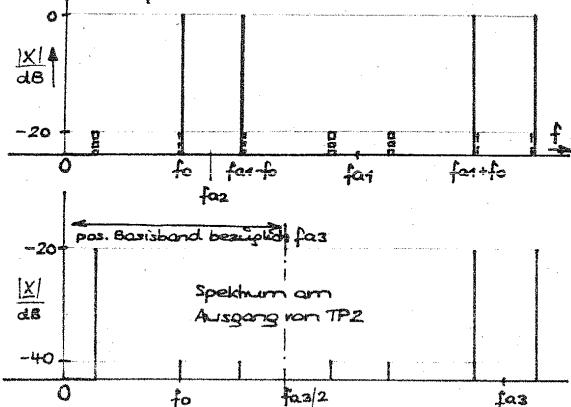
$$\begin{aligned} f_{d1} &= f_{d2} ; \quad f_{d2} = f_{S1} \\ \text{für Toleranzschema beziehen auf } f_{a2} = 3: \\ \frac{f_{d1}}{f_{a3}} &= \frac{f_{d1}}{f_{a1}} \cdot \frac{f_{a1}}{f_{a2}} \cdot \frac{f_{a2}}{f_{a3}} = 0,15 \cdot 2 \cdot \frac{1}{3} = 0,1 \\ \frac{f_{d2}}{f_{a3}} &= \frac{f_{S1}}{f_{a1}} \cdot \frac{f_{a1}}{f_{a2}} \cdot \frac{f_{a2}}{f_{a3}} = 0,25 \cdot 2 \cdot \frac{1}{3} = 0,166 \end{aligned}$$

c) α) Anregung mit Sinus $f_0 = f_{a1}/10$ im Durchlaßbereich von TP 1



Liniens nach TP1 bei $f_0 = 0,1 \cdot f_{a1}$
zusätzliche Liniens nach Dezimierung mit $R=2$ bei $f_0 = 0,2 f_{a2}$

β) Anregung mit Sinus $f_0 = 0,4 f_{a1}$ im Sperrbereich von TP 1



Liniens nach
Erhöhung der Abtastrate um $I=3$
durch Einfügen von Nullen; Amplituden
sinken um Faktor 3 $\hat{=} -9,5$ dB
bei $f_0, f_{a2} \pm f_0$
Liniens nach TP2 unter Annahme
von 20 dB Sperrdämpfung und
Verstärkungsfaktor 3 insgesamt
 \rightarrow unbedämpft bei $f_0 = 0,1 \cdot \frac{2}{3} f_{a2}$
um 20 dB bedämpft bei $f_0/faz = 0,266$ und $0,1$

Liniens am Eingang (bei f_0 im pos. Basisb.)
Liniens nach TP1 um 20 dB gedämpft
zusätzliche Liniens nach Dezimierung
Alias bei $f = f_{a2} - f_0 = 0,2 f_{a2}$!

somit ist Spektrum vor
Abtastraten erhöhung identisch
wie unter α), jedoch um 20 dB gedämpft
es ergeben sich gleiche Spektren
vor und nach TP2 wie unter α),
jedoch um zusätzlich 20 dB gedämpft

M9.2. * Ideale Abtastratenwandlung → Aufgabe 9.2

a) Verhältnis der Abtastraten

Bei Multiratensystemen ist die Abtastfrequenz der verschiedenen Teilsysteme eine wichtige Entwurfsgröße. Hier gilt:

$$\begin{aligned} f_{S1} &= 100 \text{ kHz} \Rightarrow f_{S2} = \frac{f_{S1}}{5} = 20 \text{ kHz} \Rightarrow \frac{f_{S2}}{f_{S1}} = \frac{1}{5} \\ f_{S3} &= 6f_{S2} = \frac{6f_{S1}}{5} = 120 \text{ kHz} \Rightarrow \frac{f_{S3}}{f_{S1}} = \frac{6}{5} \end{aligned}$$

Damit ist die Abtastrate des Ausgangssignals um den Faktor 6/5 höher als die des Eingangssignals.

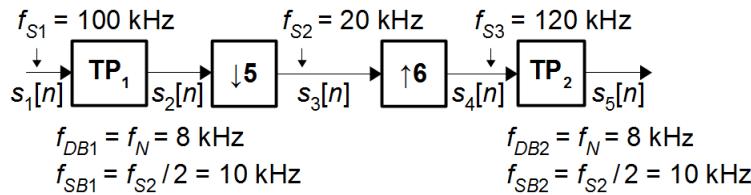


Abb. M9.2.: Abtastratenwandlung zu Aufgabe 9.2 mit Abtast- und Eckfrequenzen

b) Eckfrequenzen von TP_1

Die Dezimation entspricht einer erneuten Abtastung (Resampling) mit geringerer Abtastrate. Vor der Dezimation muss TP_1 alle Frequenzkomponenten außerhalb des *neuen* Basisbands $-f_{S2}/2 \dots + f_{S2}/2$ unterdrücken, da ansonsten *Aliasing* auftritt. TP_1 ist also ein Anti-Aliasingfilter.

$$f_{SB,1} \leq \frac{f_{S2}}{2} = \frac{f_{S1}}{10} = 10 \text{ kHz} \Rightarrow F_{SB,1,max} = \frac{f_{SB,1}}{f_{S1}} = 0,1$$

Die Eckfrequenz des Durchlassbands wird so gewählt, dass das Nutzsignal gerade noch durchgelassen wird. Höhere Eckfrequenzen für das DB würden zwar auch funktionieren, aber der Übergangsbereich zwischen DB $f_{DB,1}$ und Sperrband $f_{SB,1}$ würde schmäler, damit wäre ein Filter höherer Ordnung (= höherer Hardwareaufwand) nötig.

$$f_{DB,1} \geq f_N = 8 \text{ kHz} \Rightarrow F_{DB,1,min} = \frac{f_N}{f_{S1}} = 0,08$$

Abb. M9.3 zeigt den Frequenzgang $|H_{\text{TP}1}(e^{j2\pi F})|$ von TP_1 und das resultierende Spektrum $|S_2(e^{j2\pi F})|$.

c) Eckfrequenzen von TP_2

Die Erhöhung der Abtastrate (Resampling) um den Faktor I wird erzielt durch Einfügen von $I-1$ Nullen zwischen den Abtastwerten (*zero stuffing*). Dadurch entstehen $I-1$ Kopien (*Images*) des alten Basisbandes $-f_{S2}/2 \dots f_{S2}/2$. Nach der Erhöhung der Abtastrate von f_{S2} auf $If_{S2} = f_{S3}$ muss TP_2 alle Frequenzkomponenten außerhalb des *alten* Basisbands $-f_{S2}/2 \dots + f_{S2}/2$ unterdrücken, da ansonsten die *Images* das Signal verfälschen. TP_2 wird daher Anti-Image-, Interpolations- oder Rekonstruktionsfilter genannt.

$$f_{SB,2} \leq \frac{f_{S2}}{2} = f_{SB,1} = 10 \text{ kHz} \Rightarrow F_{SB,1,max} = \frac{f_{SB,1}}{f_{S1}} = \frac{10 \text{ kHz}}{100 \text{ kHz}} = 0,1$$

$$F_{SB,2,max} = \frac{f_{SB,2}}{f_{S3}} = \frac{f_{SB,1}}{6/5 f_{S1}} = \frac{10 \text{ kHz}}{120 \text{ kHz}} = 0,083$$

Normierte Frequenzen müssen immer auf die zugehörige Abtastfrequenz (f_{S1} bzw. f_{S3}) bezogen werden!

Die Eckfrequenz des Durchlassbandes (DB) $f_{DB,2}$ von TP_2 wird so groß gewählt, dass gerade noch alle Frequenzkomponenten des Nutzbandes durchgelassen werden, sie hat daher den gleichen Wert wie $f_{DB,1}$:

$$f_{DB,2} = f_{DB,1} = 8 \text{ kHz} \Rightarrow F_{DB,1} = \frac{f_{DB,1}}{f_{S1}} = \frac{8 \text{ kHz}}{100 \text{ kHz}} = 0,08$$

$$F_{DB,2} = \frac{f_{DB,2}}{f_{S3}} = \frac{f_{DB,1}}{6/5 f_{S1}} = \frac{8 \text{ kHz}}{120 \text{ kHz}} = 0,067$$

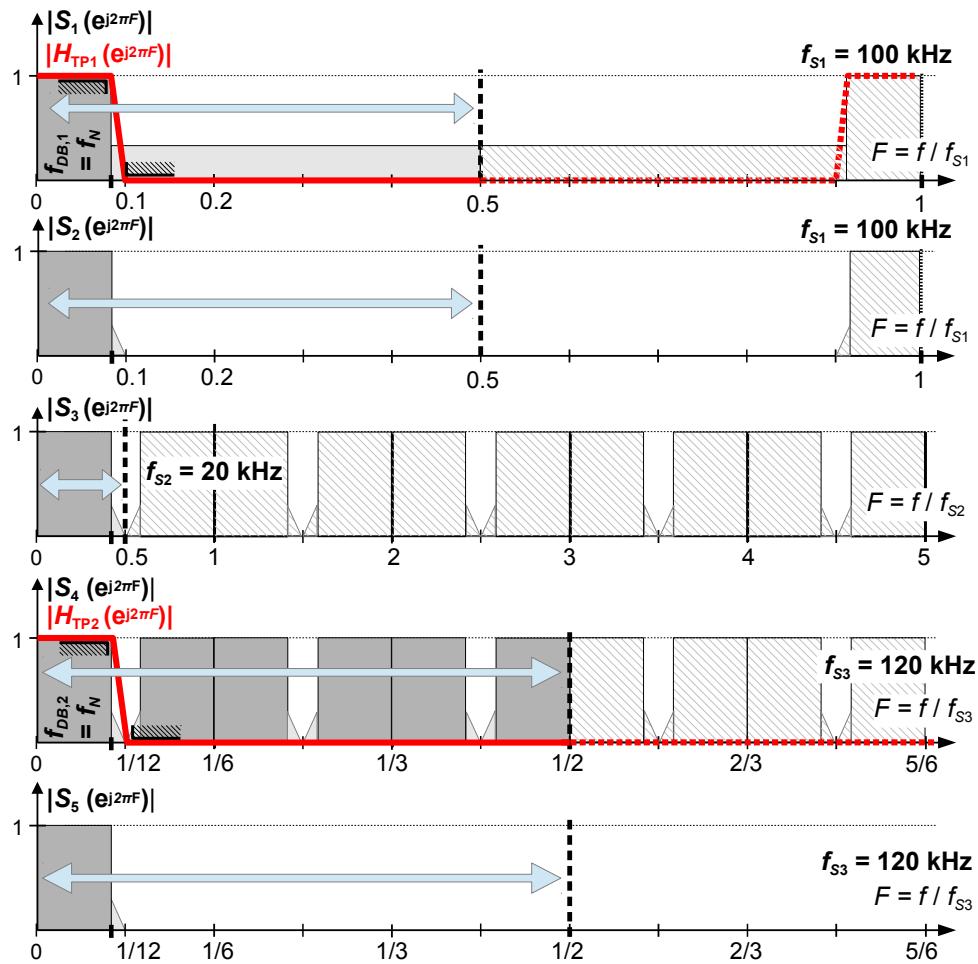


Abb. M9.3.: Spektren und Frequenzgänge (Pfeile kennzeichnen das pos. Basisband) zu Aufgabe 9.2

Die absoluten Eckfrequenzen von TP₁ und TP₂ sind identisch; aufgrund der unterschiedlichen Abtastfrequenzen ergeben sich aber anderen *relative* Grenzfrequenzen - und damit auch unterschiedliche Filterentwürfe für beide Filter!

d) Blockschaltbild von „vertauschter“ Kaskade

Bei Vertauschung von Dezimator und Interpolator müssen auch Dezimations- und Interpolationsfilter ihren Platz tauschen (Abb. M9.4)! Da hier zunächst die Abtastrate erhöht

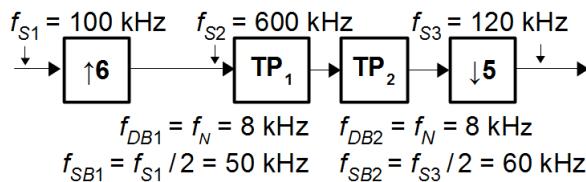


Abb. M9.4.: Alternative Abtastratenwandlung zu Aufgabe 9.2 mit vertauschter Dezimation und Interpolation

wird, wird TP₁ jetzt benötigt um die Images oberhalb $f_{S1}/2$ zu entfernen (Interpolations- oder Rekonstruktionsfilter). Vor der Dezimation wird ein weiteres Filter benötigt, um Frequenzanteile oberhalb $f_{S3}/2$ zu entfernen (Anti-Aliasingfilter). Mit der Vertauschung von Dezimation und Interpolation ändern sich auch die Spezifikationen für die Eckfrequenzen des Sperrbereichs (Abb. M9.4). Die absoluten Eckfrequenzen für die Durchlassbereiche

bleiben gleich; die höheren Abtastfrequenzen von TP_1 und TP_2 führen aber im Vergleich zu Abb. M9.2 geringeren *relativen Grenzfrequenzen*.

$$\begin{aligned} f_{DB,1} = f_{DB,2} = f_N = 8 \text{ kHz} &\Rightarrow F_{DB,1} = F_{DB,2} = f_N/f_{S2} = 8/600 = 1,33 \cdot 10^{-2} \\ f_{SB,1} = f_{S1}/2 = 50 \text{ kHz} &\Rightarrow F_{SB,1} = f_{SB,1}/f_{S2} = 50/600 = 8,33 \cdot 10^{-2} \\ f_{SB,2} = f_{S3}/2 = 60 \text{ kHz} &\Rightarrow F_{SB,2} = f_{SB,2}/f_{S2} = 60/600 = 0,1 \end{aligned}$$

Beide Tiefpassfilter werden normalerweise in einem Filter zusammengefasst (Abb. M9.5) um Hardware einzusparen. Dieses Filter muss die jeweils schärfere Anforderung für Durchlass- und Sperrbereich erfüllen.

$$\begin{aligned} f_{DB} &= \max(f_{DB,1}, f_{DB,2}) = 8 \text{ kHz} \Rightarrow F_{DB} = f_{DB}/f_{S2} = 1,33 \cdot 10^{-2} \\ f_{SB} &= \min(f_{SB,1}, f_{SB,2}) = 50 \text{ kHz} \Rightarrow F_{SB} = f_{SB}/f_{S2} = 8,33 \cdot 10^{-2} \end{aligned}$$

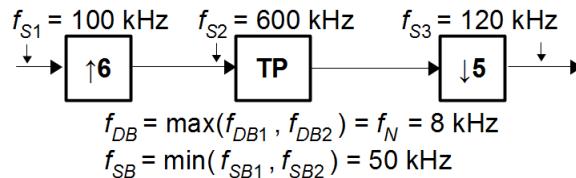


Abb. M9.5.: Alternative Abtastratenwandelung zu Aufgabe 9.2 mit zusammengefasstem TP-Filter

e) *SQNR* und *ENOB*

Die gesamte *Rauschleistung* N_Q , die bei einer (Re-)Quantisierung entsteht, hängt nur von der Größe des LSBs ab, nicht von der Abtastrate f_S , genausowenig wie die Leistung eines Sinustons von dessen Frequenz abhängt. Es ist daher $N_Q = q^2/12$ mit $q = FSR/2^W$ nur vom Aussteuerbereich (Full Scale Range, *FSR*) und von der Anzahl der Quantisierungsstufen 2^W abhängig. Das maximale *SQNR* (= bei maximaler Aussteuerung) ist unabhängig vom *FSR*, da hier die maximale Signalleistung und damit ebenfalls der Aussteuerbereich eingeht (siehe auch Aufgabe 6.1).

Die *Rauschleistungsdichte* $N'_Q = N_Q/f_S$, die bei der (Re-)Quantisierung entsteht, hängt von der Bandbreite des Basisbands f_S ab: Nach dem Parcevalschen Theorem sind die Signalleistung im Zeit- und Frequenzbereich identisch, tastet man das gleiche Signal mit höherer Abtastrate f_{S2} ab, verteilt sich daher die Rauschleistung N_Q über die größere Rauschbandbreite $B_{N2} = f_{S2}$, es gilt $N_Q = N'_{Q1}f_{S1} = N'_{Q2}f_{S2}$. Da es sich um abgetastete (zeitdiskrete!) Signale mit periodischen Spektren handelt, ist B_N natürlich höchstens gleich f_S .

Bei *Reduktion der Abtastrate* um den Faktor R ändern sich die Spektralamplituden nicht, allerdings werden Komponenten außerhalb des *neuen* Basisbands $-f_S/2R \dots + f_S/2R$ in dieses zurückgefaltet und verursachen so Verzerrungen und eine Verschlechterung des *SQNR* (*Aliasing*), die nicht wieder korrigiert werden können! Hier ist also eine Begrenzung der Bandbreite auf $B = f_S/2R$ vor der Reduktion der Abtastrate erforderlich (*Anti-Aliasfilter*).

Bei *Erhöhung der Abtastrate* um den Faktor I auf $I f_S$ durch Nullenstopfen werden die Wiederholspektren des Nutzsignals und die des Quantisierungsrauschen real bis $I f_S/2$.

Die Amplituden beider Spektralanteile werden um $1/I$ reduziert¹, das $SQNR$ im alten Basisband $-f_s/2 \dots +f_s/2$ ändert sich daher nicht. Nach der Erhöhung der Abtastrate muss die Bandbreite des Signals daher auf das alte Basisband begrenzt werden, da die Spektralkomponenten zwischen $f_s/2 \dots If_s/2$ Verzerrungen verursachen und das $SQNR$ verschlechtern.

Eine *nachträgliche Erhöhung der Abtastrate* verbessert also nicht das $SQNR$, bei ungeeigneter Filterung kann es sogar deutlich schlechter werden! Das funktioniert nur bei gleichzeitigem (Re-)Sampling und (Re-)Quantisierung, z.B. bei der Abtastung analoger Signale, da hier das neu entstehende Quantisierungsrauschen umso „dünner“ verteilt wird, je größer die neue Abtastrate ist.

Reduziert man aber die äquivalente Rauschbandbreite B_N durch Tiefpassfilterung ohne dabei das Nutzsignal zu beeinträchtigen (das geht nur bei Überabtastung, also wenn für die Bandbreite des Signals gilt $B < f_s/2!$), reduziert man auch die gesamte Rauschleistung, also $N_2 = N_1 B_{N,2}/B_{N,1}$. Das $SQNR$ verbessert sich im gleichen Verhältnis, da S unverändert ist.

Zur Berechnung des $SQNR$ an verschiedenen Stellen der Signalverarbeitungskette muss man also das $SQNR$ des Wandlers am Eingang $s_1[n]$ des Multiratensystems bestimmen und es dann mit dem Verhältnis der Rauschbandbreiten $B_{N,i}$ an verschiedenen Stellen des Systems skalieren²:

$$SQNR_1 = (6,02W + 1,76) \text{ dB} = 49,92 \text{ dB} \text{ für } W = 8.$$

Die Rauschbandbreite von s_1 beträgt $2B_{N1} = f_{S1} = 100 \text{ kHz}$. Daraus ergibt sich z.B.

$$SQNR_2 = SQNR_1 + 10 \log \frac{100 \text{ kHz}}{20 \text{ kHz}} = (49,92 + 6,99) \text{ dB} = 56,91 \text{ dB}.$$

Signal	f_s	$2B_N$	$SQNR / \text{dB}$	$ENOB$
$s_1[n]$	100 kHz	100 kHz	49,92	8,0
$s_2[n]$	100 kHz	20 kHz	56,91	9,2
$s_3[n]$	20 kHz	20 kHz	56,91	9,2
$s_4[n]$	120 kHz	—	— ³	—
$s_5[n]$	120 kHz	20 kHz	56,91	9,2

Tab. M9.1.: Werte zu Abb. M9.2

f) Vergleich der Systeme

Der größte Nachteil von **System Abb. M9.2** liegt darin, dass die Signalbandbreite B des Gesamtsystems beschränkt wird durch die niedrige Nyquistfrequenz $f_{S2}/2 = f_{S1}/2R$ nach der Dezimation. Damit ist diese Form der Abtastratenwandlung im Allgemeinen nur für Oversampling-Systeme geeignet, bei denen die Nutzbandbreite viel geringer als die

¹Diese Amplitudenreduktion wird üblicherweise im nachfolgenden Interpolationsfilter ausgeglichen und daher oft gar nicht erwähnt.

²Da nur die *Verhältnisse* der Rauschbandbreiten betrachtet werden, ist es gleich ob man die Rauschleistungsdichte über $-f_s/2 \dots f_s/2$ oder über $0 \dots f_s/2$ bestimmt.

³Vor einer $SQNR$ -Betrachtung müssen zunächst die Images entfernt werden.

Signal	f_S	$2B_N$	$SQNR / \text{dB}$	$ENOB$
$s_1[n]$	100 kHz	100 kHz	49,92	8,0
$s_2[n]$	600 kHz	—	(s.o.)	—
$s_3[n]$	600 kHz	100 kHz	49,92	8,0
$s_4[n]$	600 kHz	100 kHz	49,92	8,0
$s_5[n]$	120 kHz	100 kHz	49,92	8,0

Tab. M9.2.: Werte zu Abb. M9.4 bzw. M9.5

Abtastrate f_{S1} ist. Durch die Verringerung der Nutzbandbreite verbessert sich als positiver Nebeneffekt das $SQNR$.

Die Abschätzung der Filterordnung liefert folgende Ergebnisse:

$$N_{FIR,1} \approx \left\lceil \frac{-20 \log_{10} \sqrt{\delta_{SB} \delta_{DB}} - 13}{14,6(F_{SB,1} - F_{DB,1})} \right\rceil = \left\lceil \frac{50 - 13}{14,6 \cdot 0,1 - 0,08} \right\rceil = 127 \quad (\text{Matlab: } 127)$$

$$N_{FIR,2} \approx \left\lceil \frac{-20 \log_{10} \sqrt{\delta_{SB} \delta_{DB}} - 13}{14,6(F_{SB,2} - F_{DB,2})} \right\rceil = \left\lceil \frac{50 - 13}{14,6 \cdot (0,083 - 0,067)} \right\rceil = 153 \quad (\text{Matlab: } 153)$$

Der Vergleich mit der einfachen Abschätzung (4.4) zeigt deren größeren Fehler:

$$N_{FIR,1} \approx \left\lceil \frac{3f_{S1}}{f_{SB,1} - f_{DB,1}} \right\rceil = \left\lceil \frac{3}{F_{SB,1} - F_{DB,1}} \right\rceil = \left\lceil \frac{300 \text{ kHz}}{(10 - 8) \text{ kHz}} \right\rceil = 150$$

$$N_{FIR,2} \approx \left\lceil \frac{3f_{S3}}{f_{SB,2} - f_{DB,2}} \right\rceil = \left\lceil \frac{3}{F_{SB,1} - F_{DB,1}} \right\rceil = \left\lceil \frac{360 \text{ kHz}}{(10 - 8) \text{ kHz}} \right\rceil = 180$$

Der größte Nachteil von System **Abb. M9.4** bzw. **Abb. M9.5** liegt in der hohen „Zwischenfrequenz“ $f_{S2} = If_{S1}$. Da die Nutzbandbreite des Signals im ganzen System gleich bleibt, bleibt auch der $SQNR$ unverändert. Es wird nur ein Tiefpassfilter mit vergleichsweise geringen Anforderungen benötigt, das gleichzeitig als Dezimations- und Interpolationsfilter dient:

$$N_{FIR} = N_{FIR,1} \approx \left\lceil \frac{50 - 13}{14,6 \cdot (8,33 \cdot 10^{-2} - 1,33 \cdot 10^{-2})} \right\rceil = 37 \quad (\text{Matlab: } 36)$$

$$N_{FIR,2} \approx \left\lceil \frac{50 - 13}{14,6 \cdot (0,1 - 1,33 \cdot 10^{-2})} \right\rceil = 30 \quad (\text{Matlab: } 29)$$

M9.3. Abtastratenwandlung mit Aliasing im Übergangsbereich → Aufgabe 9.3

a) Eckfrequenzen TP_1

Die Eckfrequenzen des Durchlassbereichs für TP_1 und TP_2 bleiben unverändert. Die

System	B_{max}	$f_{S,max}$	$N_{FIR,1}$		$N_{FIR,2}$	
			\approx	sim	\approx	sim
a)	$f_{S1}/2R = 10 \text{ kHz}$	120 kHz	127	127	153	153
d), 2 TP	$\min(f_{S1}/2, f_{S3}/2) = 50 \text{ kHz}$	$I f_{S1} = 600 \text{ kHz}$	37	36	30	29
d), 1 TP	$\min(f_{S1}/2, f_{S3}/2) = 50 \text{ kHz}$	$I f_{S1} = 600 \text{ kHz}$	37	36	—	—

Tab. M9.3.: Ergebnisse zu Aufgabe 9.2

Sperrfrequenz von TP₁ darf jetzt so gewählt werden, dass bei Abtastung mit f_{S2} Alias-Komponenten mit der niedrigsten Frequenz bei $f = f_N = f_{DB,1}$ entstehen:

$$f_{DB,1} = f_N = 8 \text{ kHz} \Rightarrow F_{DB,1} = \frac{f_N}{f_{S1}} = \frac{8 \text{ kHz}}{100 \text{ kHz}} = 0,08$$

$$f_{SB,1} = f_{S2} - f_N = 20 \text{ kHz} - 8 \text{ kHz} = 12 \text{ kHz} \Rightarrow F_{SB,1} = \frac{f_{SB,1}}{f_{S1}} = \frac{12 \text{ kHz}}{100 \text{ kHz}} = 0,12$$

b) Eckfrequenzen TP₂

Die Spezifikationen für TP₂ ändern sich hier nicht, TP₂ soll ja nur die Images außerhalb des Basisbands $\pm f_{S2}/2$ unterdrücken.

c) Filterordnung

$$N_{FIR,1} \approx \left\lceil \frac{50 - 13}{14,6 \cdot (0,12 - 0,08)} \right\rceil = 64 \quad (\text{Matlab: } 64)$$

Durch das Zulassen von Aliasing ins Übergangsband kann hier die Ordnung von TP₁ halbiert werden!

M10. CIC: Cascaded Integrator-Comb Filter

M10.1. + CIC-Filter → Aufgabe 10.1

Die Übertragungsfunktion eines CIC-Filters ohne Dezimation ist identisch mit der eines Moving Average (MA) Filters mit R Taps (Ordnung $R - 1$). CIC-Filter höherer Ordnung $N > 1$ entsprechen N kaskadierten MA Filtern. Die Übertragungsfunktion ist genau wie beim MA-Filter ein Dirichlet-Kernel $\text{di}_R(x)$ (periodische si-Funktion):

$$\begin{aligned} |H_{CIC}(e^{j2\pi fT_S})|^N &= |\text{di}_R(2\pi fT_S)|^N = \left| \frac{\sin(R\pi fT_S)}{\sin(\pi fT_S)} \right|^N \approx \left| \frac{\sin(R\pi fT_S)}{\pi fT_S} \right|^N \quad \text{für } \pi fT_S \ll 1 \\ &= R^N \left| \frac{\sin(R\pi fT_S)}{R\pi fT_S} \right|^N = R^N |\text{si}(R\pi fT_S)|^N \end{aligned}$$

a) CIC-Filter ohne Dezimation:

Die Verstärkung eines CIC-Filters zweiter Ordnung ($N = 2$) mit $R = 32$ bei DC ist

$$|H_{CIC}(f = 0)| = \left| \frac{\sin(R\pi fT_S)}{\sin(\pi fT_S)} \right|_{f=0}^N = R^2 = 32^2 = 1024 \hat{=} 60,2 \text{ dB.}$$

Im Folgenden wird der Frequenzgang relativ zur Verstärkung bei DC bestimmt und zur einfacheren Rechnung durch die si-Funktion angenähert:

$$\begin{aligned} |H_{CIC}(f = 200 \text{ kHz})|^2 &\approx R^2 \left| \frac{\sin(R\pi fT_S)}{R\pi fT_S} \right|_{f=200 \text{ kHz}}^2 = |H_{CIC}(f = 0)| \underbrace{\left| \frac{\sin(R\pi fT_S)}{R\pi fT_S} \right|}_{\Delta H_{CIC}(f=200 \text{ kHz})}^2 \\ &= 32^2 \cdot \left| \frac{\sin(32\pi 200 \text{ kHz}/26 \text{ MHz})}{32\pi 200 \text{ kHz}/26 \text{ MHz}} \right|^2 = 32^2 \cdot 0,902^2 \\ &\hat{=} 60,2 \text{ dB} - 1,8 \text{ dB} \\ \Rightarrow \Delta H_{CIC}(f = 200 \text{ kHz}) &= -1,8 \text{ dB} \\ |H_{CIC}(f = 400 \text{ kHz})|^2 &\approx 32^2 \cdot \left| \frac{\sin(32\pi 0,4/26)}{32\pi 0,4/26} \right|^2 = 32^2 \cdot 0,646^2 \hat{=} 60,2 \text{ dB} - 7,6 \text{ dB} \\ \Rightarrow \Delta H_{CIC}(f = 400 \text{ kHz}) &= -7,6 \text{ dB} \end{aligned}$$

An diesen beiden Werten kann man die Nachteile von CIC-Filters erkennen: Der Verstärkungsabfall bei der Eckfrequenz des Durchlassbands beträgt hier bereits 1,8 dB (si-Verzerrung) bei nur geringer Dämpfung von Komponenten außerhalb des DBs (hier: 7,6 dB bei der doppelten Eckfrequenz des DB).

- b) **CIC-Filter mit Dezimation um R :** Das Eingangssignal wird vor der Dezimation mit dem CIC-Filter aus dem vorigen Aufgabenpunkt bewertet, das CIC-Filter dient also als Anti-Aliasing Filter. Die Abtastfrequenz nach der Dezimation am Ausgang ist $f_{S,Aus} = f_{S,Ein}/R = 26 \text{ MHz}/32 = 812,5 \text{ kHz}$, das Basisband geht also bis $f_{S,Aus}/2 = f_{S,Ein}/2R = 406,25 \text{ kHz}$.

- c) **Aliasing beim CIC-Filter mit Dezimation um R :** Frequenzkomponenten bei f außerhalb des Basisbands werden bei der Dezimation ins Basisband zurückgefaltet auf die Aliasfrequenz

$$f_A = f - kf_{S,Ein}/R = f - kf_{S,Aus} \quad \text{mit } k = 0, \pm 1 \pm 2, \dots \text{ und } |f_A| < f_{S,Aus}/2$$

Da das CIC-Filter nur einen relativ schwachen Dämpfungsverlauf hat, muss bei diesem Filtertyp ein gewisses Maß an Aliasing toleriert werden. Aus dem Frequenzgang des CIC-Filters Abb. ?? (dargestellt für $R = 8$) erkennt man, dass der Worst-Case an der Kante des Nutzbandes auftritt, Komponenten bei $f = f_{S,Aus} - f_N$ werden zurückgefaltet auf die Aliasfrequenz $f_A = f_N = f_{S,Aus} - f$. Komponenten bei $f = 612,5$ kHz werden hier im Beispiel auf $f_A = f_N = 200$ kHz zurückgefaltet.

Bei höheren Frequenzen nimmt die Dämpfung der periodischen si-Funktion rasch zu und erreicht bei $f = f_{S,Aus} = 812,5$ kHz das Maximum. Auch eine Signalkomponente bei $f = f_{S,Aus} + f_N = 1012,5$ kHz wird auf $f_N = 200$ kHz zurückgefaltet, wird aber etwas stärker gedämpft als bei $f = 612,5$ kHz. Die Dämpfung wird genauso berechnet wie in Unterpunkt a). Zum leichteren Rechnen wird hier die normierte Frequenz F eingeführt, bezogen auf die Abtastfrequenz am *Ausgang*:

$$F = \frac{Rf}{f_{S,Ein}} = \frac{f}{f_{S,Aus}} = \frac{f_A + kf_{S,Aus}}{f_{S,Aus}} = F_A + k \quad \text{mit } k = 0, \pm 1 \pm 2, \dots \text{ und } |F_A| < 0,5$$

Damit kann man die Übertragungsfunktion des CIC-Filters schreiben als

$$\left| H_{CIC}(e^{j2\pi f/f_{S,Ein}}) \right|^N = \left| \frac{\sin(R\pi f/f_{S,Ein})}{\sin(\pi f/f_{S,Ein})} \right|^N = \left| \frac{\sin(\pi f/f_{S,Aus})}{\sin(\pi f/(Rf_{S,Aus}))} \right|^N = \left| \frac{\sin(\pi F)}{\sin(\pi F/R)} \right|^N$$

Bezogen auf eine Nutzfrequenzkomponente bei gleicher Frequenz im Basisband, $f_N = f_A$, werden die Alias-Komponenten unterdrückt um

$$\begin{aligned} A_{supr} &= \frac{\left| H_{CIC}(e^{j2\pi f_N/f_{S,Ein}}) \right|^N}{\left| H_{CIC}(e^{j2\pi f/f_{S,Ein}}) \right|^N} = \left| \frac{\sin(\pi F_N)}{\sin(\pi F_N/R)} \right|^N \cdot \left| \frac{\sin(\pi F/R)}{\sin(\pi F)} \right|^N \\ &= \left| \frac{\sin(\pi F_A)}{\sin(\pi F_A/R)} \frac{\sin(\pi F/R)}{\sin(\pi F)} \right|^N \quad \text{mit } f_A = f_N \\ &= \left| \frac{\sin(\pi F_A)}{\sin(\pi F_A/R)} \frac{\sin(\pi(F_A+k)/R)}{\sin(\pi(F_A+k))} \right|^N \quad \text{mit } F = F_A + k \\ &= \left| \frac{\sin(\pi(F_A+k)/R)}{\sin(\pi F_A/R)} \right|^N \quad \text{mit } \sin(\pi(F_A+k)) = \sin(\pi F_A) \end{aligned}$$

- d) **Hogenauer-Struktur:** Hier findet die Dezimation nach N kaskadierten Integratorstufen statt mit Amplitudengang

$$\left| H_I(e^{j2\pi f T_{S,Ein}}) \right|^N = \left| \frac{1}{2 \sin \pi f T_{S,Ein}} \right|^N = \left| \frac{1}{2 \sin \pi F/R} \right|^N.$$

Bezogen auf eine Nutzfrequenzkomponente bei gleicher Frequenz im Basisband, $f_N = f_A$, werden die Alias-Komponenten unterdrückt um

$$\begin{aligned} A_{supr} &= \frac{|H_I(f_N)|^2}{|H_I(f)|^2} = \frac{|H_I(f_A)|^2}{|H_I(f_A + kf_{S,Ein}/R)|^2} \quad \text{mit } f_A = f_N \\ &= \left| \frac{2 \sin \pi(f_A + kf_{S,Ein}/R) T_{S,Ein}}{2 \sin \pi f_A T_{S,Ein}} \right|^2 = \left| \frac{\sin \pi(F_A + k)/R}{\sin \pi F_A/R} \right|^2 \quad \text{mit } F_A = Rf_A/f_{S,Ein} \end{aligned}$$

f [kHz]	100	200	400	412,5	612,5	712,5	812,5	1013	1425
F	0,123	0,246	0,492	0,508	0,754	0,861	1	1,246	1,754
$\Delta H_{CIC}(f)$ [dB]	-0,44	-1,8	-7,6	-8,1	-21,2	-34,5	∞	-29,9	-35,8
Basisband					Aliasing				
f_A [kHz]	100	200	400	400	200	100	0	200	200
F_A	0,123	0,246	0,492	0,492	0,246	0,123	0	0,246	0,246
$\Delta H_{CIC}(f_A)$ [dB]	-0,44	-1,8	-7,6	-7,6	-1,8	-0,44	∞	-1,8	-1,8
A_{Supr} [dB]	0	0	0	0,5	19,4	34,1	∞	28,1	34,0

Tab. M10.1.: CIC-Filter mit $f_{S,Ein} = 26$ MHz, $N = 2$ und $R = 32$

Man erhält die gleiche Unterdrückung von Alias-Komponenten wie beim CIC-Filter mit nachgeschalteter Dezimierung (Tab. M10.1). Dieses erstaunliche Ergebnis wird von der *Noble-Identität* vorhergesagt: Ein Filter mit $H_1(z^{-R})$ (hier: das Kammfilter), gefolgt von einer Dezimation um R ist identisch mit einer Dezimation um R , gefolgt von einem Filter $H_1(z^{-1})$.

e) Änderung der CIC-Parameter:

Halbierung von f_N : Mit einem größeren Verhältnis zwischen Nutzbandbreite und Abtastfrequenz am *Ausgang* verbessert man gleichzeitig die Alias-Unterdrückung und verringert die Dämpfung des Nutzsignals - siehe Werte in Tab. M10.1 für $f_N = 100$ kHz und $f = 712,5$ kHz. Es fallen jetzt mehr Alias-Komponenten in den „Don't Care“ - Bereich zwischen f_N und $f_{S,Aus}/2$, wo sie später ausgefiltert werden können.

Verdopplung von R : Mit Erhöhung des Dezimationsverhältnisses R verringert sich die Grenzfrequenz des Tiefpasses, damit werden höherfrequente Störungen stärker gedämpft. Allerdings wird gleichzeitig das Basisband schmäler, damit nimmt Aliasing zu.

Verdopplung von N : Mit Erhöhung der Ordnung des Filters erhöht sich nach obigen Formeln die Dämpfung der Alias-Komponenten A_{Supr} , bei einer Verdopplung von N verdoppelt sich A_{Supr} im logarithmischen Maßstab ebenfalls. Allerdings erhöht sich auch die Dämpfung des Nutzsignals an der Bandkante, was an anderer Stelle der Signalverarbeitung kompensiert werden muss.

f) Vergleich der benötigten Hardware-Ressourcen:

Moving Average Filter: $N(R - 1) = 62$ Addierer und $N(R - 1) = 62$ Register.

CIC-Filter (Abb. 10.1): $N = 2$ Addierer und Register für die Integratoren, $N = 2$ Addierer und $NR = 64$ Register für die Kammfilter, insgesamt 4 Addierer und 66 Register.

Dezimierendes CIC-Filter (Abb. 10.2): Identisch mit nicht-dezimierendem CIC-Filter, in praktischen Implementierungen wird ein zusätzliches Register für die Downsampling-Stufe benötigt.

Dezimierendes CIC-Filter in Hogenauer-Struktur (Abb. 10.3): Jeweils $N = 2$ Addierer und Register für Integrator und Kammfilter sowie ein weiteres Register für die Downsampling-Stufe, insgesamt werden daher 4 Addierer und 5 Register benötigt.

Achtung: Die Übertragungsfunktion von dezimierenden und nicht-dezimierenden CIC-Filtern ist nur für Frequenzen bis $f = f_{S,ein}/2R$ identisch, darüber tritt beim dezimierenden CIC-Filter Aliasing auf.

Detailliertere Informationen zu dezimierenden CIC-Filtern finden Sie unter [Hog81].

M11. IIR: Rekursive Filter und -entwurf

M11.1. * Filtertransformationen → Aufgabe 11.1

a) Nullstellen von $H_1(z)$

$$h_1[n] = \{1; 7/4; 1/2; -1/4\} \Rightarrow H_1(z) = 1 + 7/4z^{-1} + 1/2z^{-2} - 1/4z^{-3} \\ = z^{-3} (z^3 + 7/4z^2 + 1/2z - 1/4)$$

Eine weitere Nullstelle muss vorliegen, da das Filter dritter Ordnung ist; der Verlauf von $H_1(e^{j2\pi F})$ lässt vermuten dass diese sich bei $F = 0,5$ bzw. $f = f_S/2$ befindet. Beweis, dass *eine* (oder mehrere) Nullstellen bei $f_S/2$ vorliegen durch Einsetzen: $H_1(f = f_S/2) = H(z = -1) = -1 + 7/4 - 1/2 - 1/4 = 0$

Polynomdivision durch $(z + 1)^2 = z^2 + 2z + 1$ ergibt:

$$\begin{array}{r} (z^3 + \frac{7}{4}z^2 + \frac{1}{2}z - \frac{1}{4}) \div (z^2 + 2z + 1) = z - \frac{1}{4} \\ \underline{-z^3 - 2z^2 - z} \\ -\frac{1}{4}z^2 - \frac{1}{2}z - \frac{1}{4} \\ \underline{\frac{1}{4}z^2 + \frac{1}{2}z + \frac{1}{4}} \\ 0 \end{array}$$

Die Polynomdivision zeigt, dass bei $z = -1$ eine doppelte Nullstelle vorliegt und bei $z = +1/4$ eine weitere einfache Nullstelle.

b) Filter mit jeweils zwei Verzögerungsgliedern

$$h_2[n] = \{1; 0; 7/4; 0; 1/2; 0; -1/4\} \quad \circlearrowleft \bullet \quad H_2(z) = 1 + 7/4z^{-2} + 1/2z^{-4} - 1/4z^{-6}$$

Zusätzliche Verzögerungsglieder stauchen den Frequenzgang zusammen. Die Maxima finden sich jetzt bei $f = 0$ und $f = f_S/2$; das Minimum liegt jetzt bei $f_S/4$ (siehe Abb. M11.1). (nicht gefragt)

c) TP-HP-Transformation

durch $H_3(z) = H_1(-z)$ ergibt $H_3(z) = 1 - 7/4z^{-1} + 1/2z^{-2} + 1/4z^{-3}$. Durch die Transformation wird der Frequenzgang $|H_3(f)|$ gegenüber $|H_1(f)|$ um $f_S/2$ verschoben, das Maximum liegt daher jetzt bei $f_S/2$, das Minimum bei $f = 0$ (siehe Abb. M11.1):

$$H_3(f = 0) = H_3(z = 1) = 1 - 7/4 + 1/2 + 1/4 = 0$$

$$H_3(f = f_S/2) = H_3(z = -1) = 1 + 7/4 + 1/2 - 1/4 = 3$$

$$h_3[n] = \{1; -7/4; +1/2; +1/4\}$$

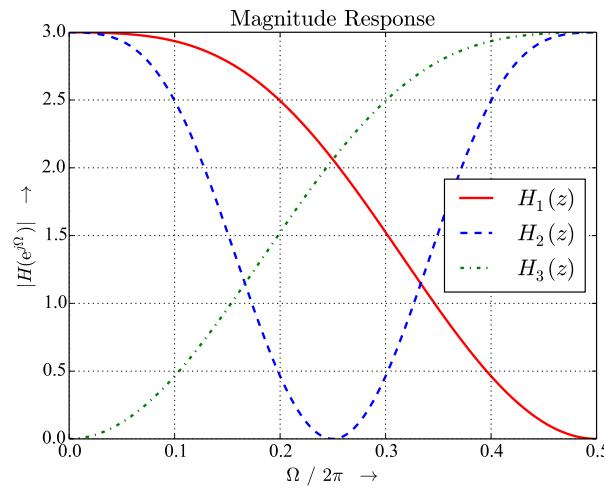


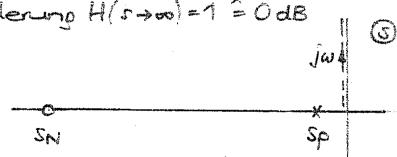
Abb. M11.1.: Betragsfrequenzgänge zu Aufgabe 11.1

M11.2. Bilineare Transformation → Aufgabe 11.2

a) Ansatz $H(s) = \frac{s - s_N}{s - s_P}$ erfüllt bereits Forderung $H(s \rightarrow \infty) = 1 \hat{=} 0 \text{ dB}$

$$H(s=0) = \frac{s_N}{s_P} = V_0 \stackrel{!}{=} 10 \hat{=} 20 \text{ dB}$$

$$H(j\omega) = \frac{j\omega + V_0 \omega_0}{j\omega + \omega_0} = V_0 \frac{1 + \frac{j\omega}{V_0 \omega_0}}{1 + \frac{j\omega}{\omega_0}}$$

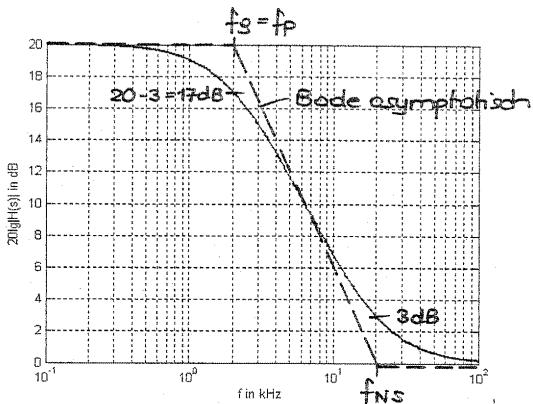


$$s_P = -\omega_P \stackrel{!}{=} -\omega_0 \\ s_N = -\omega_N = -V_0 \omega_0$$

ergibt Bodediagramm wie gefordert mit $f_g = 2 \text{ kHz}$, wenn

$$\omega_0 = 2\pi f_g = 12,566 \cdot 10^3 \text{ s}^{-1}$$

Betragsfrequenzgang $\rightarrow |H(j\omega)|$ des analogen Tiefenanhobungsfilters mittels MATLAB-Programm (umsetzung)



b) Forderung für digitales Filter: $s_{2g} = 2\pi \frac{f_g}{f_a} = 2\pi \frac{2 \text{ kHz}}{32 \text{ kHz}} = \frac{\pi}{8}$

führt mittels Bilineartransformation auf ein

analoges Referenzfilter mit $F_{P,\text{Ref}} = F_{N,\text{Ref}} = \frac{f_a}{\pi} \tan\left(\frac{\pi f_g}{2}\right) = \frac{32}{\pi} \tan\left(\frac{\pi}{16}\right) = 2.026 \text{ kHz}$

welches im übrigen prinzipiell nicht identisch ist mit Filter unter a)!

$H(z)$ aus $H_{\text{Ref}}(s) = \frac{s - s_N}{s - s_P}$ mit $s_P = -2\pi F_{P,\text{Ref}} = -12,73 \cdot 10^3 \text{ s}^{-1}$; $\frac{s_N}{s_P} = V_0 = 10$

durch Substitution $s = 2f_a \frac{z-1}{z+1}$ ergibt

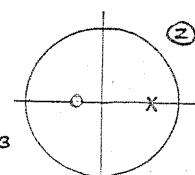
$$H(z) = \frac{2f_a \frac{z-1}{z+1} - s_N}{2f_a \frac{z-1}{z+1} - s_P} = \frac{2(z-1) - (z+1)s_N/f_a}{2(z-1) - (z+1)s_P/f_a} = \frac{(2-s_N/f_a)z + (-2-s_N/f_a)}{(2-s_P/f_a)z + (-2-s_P/f_a)}$$

$$H(z) = \frac{(\omega_N/f_a + 2)z + (\omega_N/f_a - 2)}{(\omega_P/f_a + 2)z + (\omega_P/f_a - 2)} \stackrel{!}{=} \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}} = b_0 \frac{z - z_N}{z - z_P}$$

$$\text{mit } b_0 = \frac{\omega_N/f_a + 2}{\omega_P/f_a + 2} = \frac{12,73/32 + 2}{12,73/32 + 2} = 2.4932$$

$$b_1 = \frac{\omega_N/f_a - 2}{\omega_P/f_a + 2} = \frac{12,73/32 - 2}{12,73/32 + 2} = 0.825$$

$$a_1 = \frac{\omega_P/f_a - 2}{\omega_P/f_a + 2} = \frac{12,73/32 - 2}{12,73/32 + 2} = -0.668$$



-8-

Alternative: MATLAB-Funktion zur Ausführung der Bilineartransformation (s. Programm):

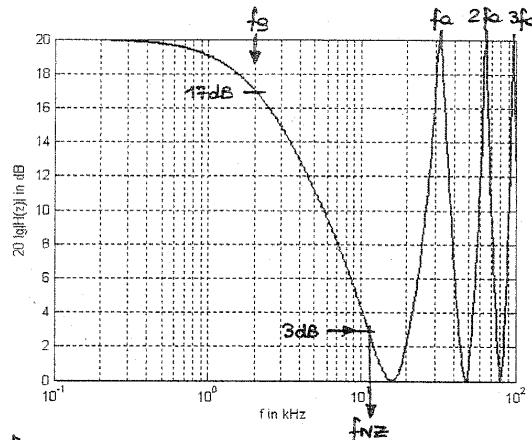
BILINEAR Bilinear transformation with optional frequency prewarping.

[NUMd,DEND] = BILINEAR(NUM,DEN,Fs) converts the s-domain transfer function specified by Z, P, and K to a z-transform discrete equivalent obtained from the bilinear transformation:

$H(z) = H(s) \mid s = 2*Fs*(z-1)/(z+1)$ where NUM and DEN are row vectors containing numerator and denominator transfer function coefficients, NUM(s)/DEN(s), in descending powers of s, transforms to z-transform coefficients NUMd(z)/DEND(z).

BILINEAR accepts an optional additional input argument that specifies prewarping. For example, [NUMd,DEND] = BILINEAR(NUM,DEN,Fs,Fp) applies prewarping before the bilinear transformation so that the frequency responses before and after mapping match exactly at frequency point Fp (match point Fp is specified in Hz).

Betragsfrequenzgang des
digitalen Filters
mittels MATLAB - Programm (s. unten)



c) Vergleich der „Nullstellen – Eckfrequenzen“ f_N

analoges Filter : $f_{N\text{ref}} = V_0 f_p = 10 \cdot 2 \text{ kHz} = 20 \text{ kHz}$

digitales Filter : f_{Nz} über Umkehrformel der Bilinear-Transformation aus $f_{N\text{ref}}$ des analogen Referenzfilters:

$$f_{N\text{ref}} = V_0 F_{P\text{ref}} = 10 \cdot 2,026 = 20,26 \text{ kHz}$$

$$f_{Nz} = \frac{f_a}{\pi} \arctan \left(\pi \frac{f_{N\text{ref}}}{f_a} \right) = \frac{32}{\pi} \arctan \left(\pi \frac{20,26}{32} \right) = 11,25 \text{ kHz}$$

Frequenzbereich wird von $0 \leq f < \infty$ beim analogen Referenzfilter auf $0 \leq f \leq \frac{f_a}{2}$ des digitalen Filters gestaucht!

Verwendetes MATLAB-Programm:

```
V0dB=20; % Verstärkung in dB bei f=0
V0 = 10^(V0dB/20); % = 10
fa = 32; % Abtastrate in kHz
fg = 2; % "Eckfrequenz" in kHz
Wg = 2*pi*fg; % = 12.566
bs = [1 V0*Wg];
as = [1 Wg];
f = logspace(-1,2,300);
Hs = freqs(bs,as,2*pi*f); % Übertragungsfunk. analog
semilogx(f,20*log10(abs(Hs)),'Linewidth',2);
grid; xlabel('f in kHz'); ylabel('20lg|H(s)| in dB');
[bzt,azt] = bilinear(bs,as,fa,2); % Bilineartransform.
Hz= freqz(bzt,azt,f,fa); % Übertragungsfunktion digital
figure;
semilogx(f,20*log10(abs(Hz)),'Linewidth',2);
grid; xlabel('f in kHz'); ylabel('20 lg|H(z)| in dB');

Wnt = V0*Wgt; % = 127.3
fn=fa/pi*atan(Wnt/(2*fa)); % = 11.255

% Im Weiteren zur numerischen Kontrolle:
Fgref = fa/pi*tan(pi*fg/fa); % = 2.0261
Wpt = 2*pi*Fgref; % = 12.73
b0z = (Wnt/fa+2)/(Wpt/fa+2); % = 2.4932
b1z = (Wnt/fa-2)/(Wpt/fa+2); % = 0.82502
a1z = (Wgt/fa-2)/(Wpt/fa+2); % = -0.66818
bz = [b0z b1z]; % identisch mit bzt oben
az = [1 a1z]; % identisch mit azt oben

ftest=[fg,V0*fg,fn];
HsDB=20*log10(abs(freqs(bs,as,2*pi*ftest)));
HzDB=20*log10(abs(freqz(bz,az,test,fa)))
```

f / kHz	2	11.25	20
20 lg Hs in dB	17.03	2.967	-
20 lg Hz in dB	17.03	-	2.967

3dB - Werte nicht exakt
wegen Überlagerung der
beiden Grad 1 - Funktionen:

M11.3. Resonator → Aufgabe 11.3

a) Differenzengleichung und Implementierung

$$H(z) = \frac{Y(z)}{X(z)} = \frac{g_0}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$\Rightarrow Y(z) = g_0 X(z) - a_1 z^{-1} Y(z) - a_2 z^{-2} Y(z) \bullet\circ y[n] = g_0 x[n] - a_1 y[n-1] - a_2 y[n-2]$$

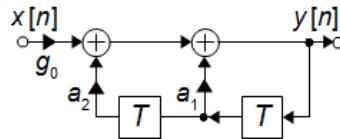


Abb. M11.2.: Resonator in Direktform Typ 1

$$H(z) = \frac{Y(z)}{X(z)} = \frac{g_0 z^2}{z^2 + a_1 z^1 + a_2}$$

$$\Rightarrow Y(z) z^2 = g_0 z^2 X(z) - a_1 z^1 Y(z) - a_2 Y(z) \bullet\circ y[n+2] = g_0 x[n+2] - a_1 y[n+1] - a_2 y[n]$$

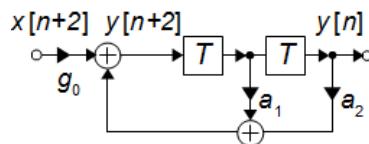


Abb. M11.3.: Resonator in Direktform Typ 2

b) Überführen von DF1 in DF2

Verschieben Sie die beiden Verzögerungen über Knoten und Addierer, beachten Sie dabei die entsprechenden Regeln (s. Folien zu Kapitel 1).

c) Lage der Polstellen

$$H(z) = \frac{g_0}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{g_0 z^2}{(z - z_{\infty,1})(z - z_{\infty,2})} \quad (\text{M11.1})$$

Die Wurzeln der characteristischen Gleichung Δ bestimmen die Pole $z_{\infty,i}$ des Systems (M11.2):

$$\Delta = 1 + a_1 z^{-1} + a_2 z^{-2} \Rightarrow z_{\infty,i} = -\frac{a_1}{2} \pm \sqrt{\frac{a_1^2}{4} - a_2} \quad (\text{M11.2})$$

d) Stabilitätsbereich:

i) Komplexe Pole

Für $a_2 > a_1^2/4$ hat das System zwei konjugiert komplexe Pole $z_{\infty,1} = z_{\infty,2}^* = r_{\infty} e^{\pm j\theta_{\infty}}$ (geschrieben in Polarform mit Polradius r_{∞} (M11.3) und Polwinkel θ_{∞} (M11.4)).

$$r_{\infty}^2 = r_{\infty,1}^2 = r_{\infty,2}^2 = |z_{\infty,i}|^2 = \frac{a_1^2}{4} + \left(a_2 - \frac{a_1^2}{4}\right) = a_2 \quad (\text{M11.3})$$

$$\theta_{\infty,i} = \arctan \frac{\Im\{z_{\infty,i}\}}{\Re\{z_{\infty,i}\}} = \pm \arctan \frac{\sqrt{a_2 - a_1^2/4}}{a_1/2} = \pm \theta_{\infty} \quad (\text{M11.4})$$

$$\Leftrightarrow a_1 = -\Re\{z_{\infty,i}\} = -2r_{\infty} \cos \theta_{\infty} \quad \text{und} \quad a_2 = r_{\infty}^2 \quad (\text{M11.5})$$

Das System ist stabil wenn alle Pole innerhalb des Einheitskreises (EK) liegen, also für einen Polradius $r_\infty < 1$.

ii) Reelle Pole (nicht gefragt)

Bei reellen Polen ($a_2 < a_1^2/4$), ist das System stabil wenn [ZB95]

$$a_1 > -1 - a_2 \quad \text{and} \quad a_1 < 1 + a_2 \quad \text{for} \quad a_2 < \frac{a_1^2}{4} \quad (\text{M11.6})$$

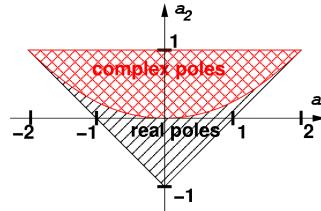


Abb. M11.4.: Stabilitätsgebiet eines Systems zweiter Ordnung mit reellen Koeffizienten a_1, a_2

Die Stabilitätsbedingung (M11.7) für komplexe Pole kann direkt aus (M11.3) abgeleitet werden und als *Stabilitätsdreieck* in Abb. M11.4 dargestellt werden.

$$a_2 < 1 \quad \text{for} \quad a_2 > \frac{a_1^2}{4} \quad (\text{M11.7})$$

Man kann jetzt die Übertragungsfunktion in Polarform aufstellen und kann so einen Resonator mit gewünschtem Polwinkel (Resonanzfrequenz) dimensionieren (M11.8).

$$H(z) = \frac{g_0}{(1 - r_\infty e^{j\theta_\infty} z^{-1})(1 - r_\infty e^{-j\theta_\infty} z^{-1})} = \frac{g_0}{1 - 2r_\infty \cos \theta_p z^{-1} + r_\infty^2 z^{-2}} \quad (\text{M11.8})$$

M12. ZST: Zustandsraum

M12.1. Zustandsgleichungen aus SFG → Aufgabe 12.1

In dieser Aufgabe sollen die Zustandsgleichungen der System in Abb. 1.9 und der dazu transponierten Systeme in Abb. M1.8 bestimmt werden:

a) Systeme aus Aufgabe 1.6 (Abb. 1.9 a - d)

1. Allpass erster Ordnung in WDF-Form Abb. 1.9(a):

Hier handelt es sich um ein System erster Ordnung. Die Matrizen und Vektoren degenerieren daher hier zu Skalaren und die Zustandsraumdarstellung ist nicht besonders sinnvoll. Am Einfachsten erhält man die Zustandsgrößen, indem man das Gleichungssystem direkt in der geeigneten Form aufstellt. Die Zustandsmatrix \mathbf{A} erhält man dann, indem man verfolgt wie der neue Wert der Zustandsvariablen ($s[n+1]$) vom alten ($s[n]$) abhängt:

$$\begin{aligned} s[n+1] &= As[n] + Bx[n] \quad \circlearrowleft \bullet \quad S(z)z^{+1} = AS(z) + BX(z) \\ y[n] &= Cs[n] + dx[n] \quad \circlearrowleft \bullet \quad Y(z) = CS(z) + dX(z) \end{aligned}$$

Die Größen des Zustandsraumsystems kann man dann aus dem „ausgefüllten“ Gleichungssystem (M2.5) direkt ablesen:

$$\mathbf{A} = A = -k, \quad \mathbf{B} = B = 1 + k, \quad \mathbf{C} = C = 1 - k \text{ und } d = k.$$

2. Transponierter IIR Filter in Direktform Abb. 1.9(b):

$$\begin{aligned} \begin{bmatrix} s_1[n+1] \\ s_2[n+1] \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & -0,81 \\ 1 & 0 \end{bmatrix}}_{=A} \begin{bmatrix} s_1[n] \\ s_2[n] \end{bmatrix} + \underbrace{\begin{bmatrix} -0,19 \\ 2 \end{bmatrix}}_{=B} x[n] \\ y[n] &= \underbrace{\begin{bmatrix} 0 & -1 \end{bmatrix}}_{=C} \begin{bmatrix} s_1[n] \\ s_2[n] \end{bmatrix} + \underbrace{(-1)}_{=d} x[n] \end{aligned}$$

3. Resonator oder LDI-Struktur Abb. 1.9(c):

- Fehlt noch -

4. Rader-Gould Struktur Abb. 1.9(d):

- Fehlt noch -

b) Transponierte Systeme aus Aufgabe 1.6 (Abb. M1.8 a - d)

1. Transponierter Allpass erster Ordnung in WDF-Form Abb. M1.8(a):

Die Matrizen des ursprünglichen und des transponierten Systems sind hier identisch: $\mathbf{A} = -k$, $\mathbf{B} = 1 + k$, $\mathbf{C} = 1 - k$, $d = k$. Ebenso sind die benötigten Hardweareressourcen und der kritische Pfad identisch.

2. IIR Filter in Direktform Abb. M1.8(b):

$$\begin{bmatrix} s_1[n+1] \\ s_2[n+1] \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0,81 & 0 \end{bmatrix} \begin{bmatrix} s_1[n] \\ s_2[n] \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} x[n]$$

$$y[n] = \begin{bmatrix} -1 & 2 \end{bmatrix} \begin{bmatrix} s_1[n] \\ s_2[n] \end{bmatrix} + (-1)x[n]$$

$$\Rightarrow \mathbf{A}^T = \begin{bmatrix} 0 & 1 \\ -0,81 & 0 \end{bmatrix}, \mathbf{B}^T = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \mathbf{C}^T = \begin{bmatrix} -1 & 2 \end{bmatrix} \text{ und } d^T = -1$$

Man sieht dass tatsächlich gilt $\mathbf{A}^T = (\mathbf{A})^T$, die Übertragungsfunktion beider Systeme ist aber natürlich identisch!

3. Transponierte Resonator- oder LDI-Struktur Abb. M1.8(c):

- Fehlt noch -

4. Transponierte Rader-Gould Struktur Abb. M1.8(d):

- Fehlt noch -

Überprüfung mit Matlab:

Nutzen Sie hierfür die folgenden Befehle:

```
A = [0 1;-0,81 0]; B = [0; 1]; C = [-1 2]; D = -1;
my_sys = ss (A, B, C, D); % Definiere System aus Matrizen oder
my_sys = tf(b a); % Koeffizienten
step(my_sys); % oder
step (A, B, C, D); % Stoßantwort
lsim (A, B, C, D, u, t); % Transiente Antwort auf Eingangssignal u
% (t ist Zeitvektor)
[y,t,x]=lsim (A, B, C, D, u, t); % oder Speicherung in Ausgangs-
% und Zustandsvektoren y, t, x
bode(A, B, C, D)
```

M12.2. SFG aus Zustandsgleichungen → Aufgabe 12.2

Gegeben ist:

$$\begin{aligned} \mathbf{s}[n+1] &= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \mathbf{s}[n] + \begin{bmatrix} 0 \\ 1 \end{bmatrix} x[n] \\ y[n] &= \begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{s}[n] + x[n] \end{aligned}$$

Gesucht sind:

Impulsantwort $h[n]$, Übertragungsfunktion $H(z)$ und Stabilität des Systems

a) Setze:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & 1 \end{bmatrix} \text{ und } d = 1$$

$$\begin{aligned} \mathbf{s}[n+1] &= \mathbf{As}[n] + \mathbf{Bx}[n] \quad \circlearrowleft \quad z^{+1}\mathbf{S}(z) = \mathbf{AS}(z) + \mathbf{BX}(z) \\ y[n] &= \mathbf{Cs}[n] + dx[n] \quad \circlearrowleft \quad Y(z) = \mathbf{CS}(z) + dX(z) \end{aligned}$$

Bestimmen von $Y(z)$:

$$\Rightarrow z^{+1}\mathbf{S}(z) - \mathbf{AS}(z) = (\mathbf{E}z^{+1} - \mathbf{A}^T)\mathbf{S}(z) = \mathbf{BX}(z) \text{ mit } \mathbf{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ (Einheitsmatrix)}$$

$$\Rightarrow \mathbf{S}(z) = (\mathbf{E}z^{+1} - \mathbf{A})^{-1} \mathbf{BX}(z)$$

$$\Rightarrow Y(z) = \mathbf{CS}(z) + dX(z) = \mathbf{C}(\mathbf{E}z^{+1} - \mathbf{A})^{-1} \mathbf{BX}(z) + dX(z)$$

Nebenrechnung:

$$(\mathbf{E}z^{+1} - \mathbf{A})^{-1} = \left(\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \right)^{-1} = \begin{bmatrix} z & -1 \\ -1 & z-1 \end{bmatrix}^{-1} = \frac{\begin{bmatrix} z-1 & 1 \\ 1 & z \end{bmatrix}}{z^2 - z - 1}$$

Die Übertragungsfunktion $H_1(z)$ wird jetzt aus dem Quotient von $Y(z)$ und $X(z)$ bestimmt:

$$\begin{aligned} H_1(z) &= \frac{Y(z)}{X(z)} = \frac{\mathbf{C}(\mathbf{E}z^{+1} - \mathbf{A})^{-1} \mathbf{BX}(z) + dX(z)}{X(z)} = \mathbf{C}(\mathbf{E}z^{+1} - \mathbf{A})^{-1} \mathbf{B} + d \\ &= \frac{\mathbf{C} \begin{bmatrix} z-1 & 1 \\ 1 & z \end{bmatrix} \mathbf{B}}{z^2 - z - 1} + d = \frac{\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} z-1 & 1 \\ 1 & z \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}}{z^2 - z - 1} + 1 = \frac{\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ z \end{bmatrix}}{z^2 - z - 1} + 1 \\ &= \frac{1+z}{z^2 - z - 1} + 1 = \frac{z^2}{z^2 - z - 1} = \frac{1}{1 - z^{-1} - z^{-2}} \end{aligned}$$

b) Polstellen (= Nullstellen des Nenners) ermitteln für Stabilitätsuntersuchung:

$$1 - z^{-1} - z^{-2} = 0 \Leftrightarrow z^2 - z - 1 = 0 \quad \Leftrightarrow \quad z_{\infty,1,2} = \frac{1}{2} \pm \frac{\sqrt{5}}{2} = \begin{cases} 1,6180 \\ -0,6180 \end{cases}$$

Die Polstelle $z_{\infty,1} = 1,618$ liegt außerhalb des Einheitskreises, somit ist das Stabilitätskriterium nicht erfüllt!

- c) $H_1(z)$ beschreibt ein rein rekursives System (IIR-Filter), das sich leicht als Signalfußplan Abb. M12.1(a) darstellen lässt:

$$H_1(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 - z^{-1} - z^{-2}} \Rightarrow Y(z) = \frac{X(z)}{1 - z^{-1} - z^{-2}}$$

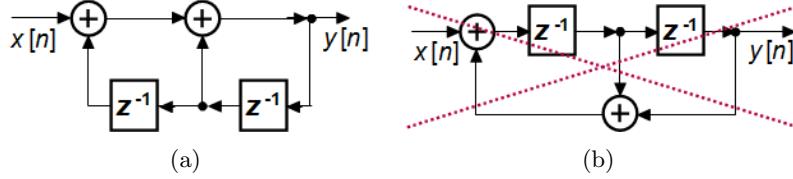


Abb. M12.1.: Signalfußplan zu Aufgabe 12.1c; (a) korrekte und (b) falsche Lösung

Anmerkung: Der Amplitudenfrequenzgang und die Polstellen von System Abb. M12.1(a) und Abb. M12.1(b) sind identisch. System Abb. M12.1(b) hat aber zwei Verzögerungen im direkten Signalpfad, die zwei zusätzlichen Nullstellen im Ursprung entsprechen, $z_{0,1} = z_{0,2} = 0$. Abb. M12.1(b) hat daher eine zusätzliche lineare Phase¹ von $\phi(e^{j\Omega}) = -2\Omega$. Die Übertragungsfunktion lautet (selbst nachprüfen!):

$$H_2(z) = \frac{Y(z)}{X(z)} = \frac{z^{-2}}{1 - z^{-1} - z^{-2}} \Rightarrow Y(z) = \frac{z^{-2}X(z)}{1 - z^{-1} - z^{-2}}$$

- d) Aus dem Signalfußplan Abb. M12.1(a) lässt sich direkt die Differenzengleichung herauslesen:

$$y[n] = x[n] + y[n-1] + y[n-2]$$

Wird die Eingangsfolge $x[n]$ durch $\delta[n]$ ersetzt, ergibt sich die rekursive Definition der Impulsantwort

$$h_1[n] = \delta[n] + h[n-1] + h[n-2] = 1,1,2,3,5,8,13,21, \dots$$

Auch im Zeitbereich lässt sich sofort erkennen, dass das System nicht stabil ist. Die Zahlenfolge ist die sogenannte *Fibonacci-Folge*, die z.B. angeblich für Verschlüsselungen eingesetzt werden kann [Dan Brown, *Sakrileg*]. Siehe auch http://en.wikipedia.org/wiki/Fibonacci_number.

Anmerkung: System Abb. M12.1(b) hat die Differenzengleichung

$$y[n] = x[n-2] + y[n-1] + y[n-2]$$

und die Impulsantwort

$$h_2[n] = \delta[n-2] + h[n-1] + h[n-2] = 0,0,1,1,2,3,5,8,13,21, \dots$$

¹Da das Filter rekursiv ist, ist die Gesamtphase aber nichtlinear.

Teil III.

Anhang

A. Wichtige Formeln

A.1. Komplexe Ebene

$$r = |z| = \sqrt{x^2 + y^2}; \quad x = \Re\{z\} \text{ und } y = \Im\{z\}$$

$z = x + jy = re^{j\phi}$ mit

$$\phi = \arg(z) = \begin{cases} \arctan \frac{y}{x} & \text{für } x > 0 \quad \bullet \circ \\ \arctan \frac{y}{x} + \pi & \text{für } x < 0, y \geq 0 \quad \blacksquare \\ \arctan \frac{y}{x} - \pi & \text{für } x < 0, y < 0 \quad \square \\ \frac{\pi}{2} & \text{für } x = 0, y > 0 \quad \blacktriangle \\ -\frac{\pi}{2} & \text{für } x = 0, y < 0 \quad \blacktriangledown \end{cases} \quad (\text{A.1})$$

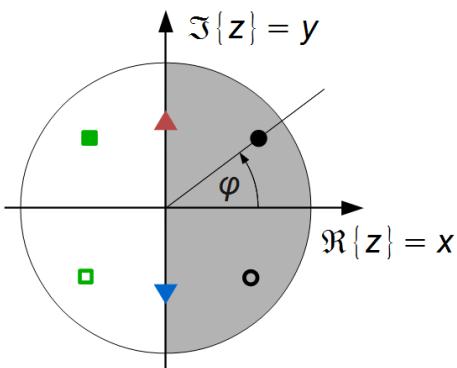


Abb. A.1.: Komplexe z -Ebene

A.2. Trigonometrische und Eulersche Identitäten

$$e^z = \cos z + j \sin z \quad (\text{A.2})$$

$$\sin(z) = \frac{e^{jz} - e^{-jz}}{2j} \quad (\text{A.3})$$

$$\cos(z) = \frac{e^{jz} + e^{-jz}}{2} \quad (\text{A.4})$$

$$\tan(z) = \frac{\sin(z)}{\cos(z)} = -j \frac{e^{jz} - e^{-jz}}{e^{jz} + e^{-jz}} = -j \frac{1 - e^{-2jz}}{1 + e^{-2jz}} \quad (\text{A.5})$$

$$(\text{A.6})$$

A.3. Exponentialfunktion und Logarithmus

$$e^z = \sum_{n=0}^{\infty} \frac{z^n}{n!} \approx 1 + z \quad \text{für } |z| \ll 1 \quad (\text{A.7})$$

$$\ln(z) = \ln|z| + j(\arg(z) + 2k\pi); \quad k \in \mathbb{Z} \quad \text{wegen Periodizität der Exponentialfunktion} \quad (\text{A.8})$$

$$= \sum_{n=0}^{\infty} \frac{2}{2n+1} \left(\frac{z-1}{z+1} \right)^{2n+1} \approx 2 \frac{z-1}{z+1} \quad \text{für } z \approx 1 \quad \text{im Hauptzweig } (k=0) \quad (\text{A.9})$$

A.4. Quadratische Gleichung

Die quadratische Gleichung $x^2 + px + q = 0$ hat die Wurzel(n)

$$x_{1,2} = \frac{-p \pm \sqrt{p^2 - 4q}}{2} \quad (\text{A.10})$$

in den folgenden Bereichen

i)	$p^2 < 4q$	\Rightarrow	zwei komplexe Lösungen
ii)	$p^2 = 4q$	\Rightarrow	eine reelle Lösung
iii)	$p^2 > 4q$	\Rightarrow	zwei reelle Lösungen
iv)	$p > 0, p^2 \neq 4q$	\Rightarrow	zwei Lösungen in der linken Halbebene
v)	$p < 0, p^2 \neq 4q$	\Rightarrow	zwei Lösungen in der rechten Halbebene

Die Gleichung $x^N \pm a = 0$ lässt sich auflösen mit Hilfe der Identität $1 \equiv e^{j2\pi k}$ für alle $k \in \mathbb{Z}$:

$$x^N - a = 0 \Leftrightarrow x_{0,k} = \begin{cases} \sqrt[N]{a} e^{j2\pi k/N} & \text{für } a \in \mathbb{R} > 0 \\ \sqrt[N]{|a|} e^{j(2\pi k+1)/N} & \text{für } a \in \mathbb{R} < 0 \end{cases}; \quad k = 0 \dots N-1 \quad (\text{A.11})$$

A.5. Summenformeln

A.5.1. Geometrische Reihe

Bei zeitdiskreten LTI-Systemen durchläuft das Signal eine endliche (transversale Systeme) bzw. unendliche (rekursive Systeme) Anzahl von Verzögerungen, die in der z -Ebene mit z^{-k} beschrieben werden. Die Kombination aus verschiedenen verzögerten Kopien des Signals lässt sich mit Hilfe der Summenformeln für die geometrische Reihe bestimmen:

$$\text{Unendlich:} \quad \sum_{k=0}^{\infty} z^k = \frac{1}{1-z} \quad \text{für } |z| < 1 \quad (\text{A.12})$$

Ersetzt man z durch z^{-1} , erhält man die Systemfunktion eines Integrators.

$$\text{Endlich: } \sum_{k=0}^{N-1} z^k = \frac{1-z^N}{1-z} = \frac{z^N - 1}{z - 1} \quad \text{für } z \neq 1 \quad (\text{A.13})$$

Analog zum vorigen Fall beschreibt diese Formel die Systemfunktion eines MA-Filters der Ordnung $N - 1$. Man sieht, dass man den gleichen Frequenzgang erhält, wenn man ein Kammfilter der Ordnung N mit $H_1(z) = 1 - z^{-N}$ und einen Integrator mit $H_2(z) = 1/(1 - z^{-1})$ kaskadiert.

Um den Frequenzgang eines MA-Filters zu berechnen, ersetzt man $z = e^{j\Omega}$ und erhält:

$$\begin{aligned} \text{Dirichlet: } \text{di}_N(2\Omega) &= \frac{1}{N} \sum_{k=-(N-1)}^{N-1} e^{jk\Omega} = \frac{1}{N} \frac{e^{jN\Omega} - e^{-jN\Omega}}{e^{j\Omega} - e^{-j\Omega}} \\ &= \begin{cases} \frac{\sin N\Omega}{N \sin \Omega} & \text{für } \Omega \neq k\pi \\ (-1)^{k(N-1)} & \text{sonst} \end{cases} \end{aligned} \quad (\text{A.14})$$

Der Betrag des Dirichlet-Kernels (die Dirichlet-Funktion ist etwas völlig anderes!) entspricht dem Betragsspektrum eines N -Punkte Rechteckfensters bzw. eines Moving Average Filters der Länge N (Ordnung $N - 1$), skaliert mit $1/N$:

$$\begin{aligned} h[n] &= \frac{1}{N} \sum_{k=0}^{N-1} \delta[n - k] \\ \circledast H(e^{j\Omega}) &= \frac{1}{N} \sum_{k=0}^{N-1} e^{jk\Omega} = \frac{e^{jN\Omega} - 1}{e^{j\Omega} - 1} = \frac{e^{-j(N-1)\Omega/2}}{N} \frac{e^{jN\Omega/2} - e^{-jN\Omega/2}}{e^{j\Omega/2} - e^{-j\Omega/2}} \\ \Rightarrow |H(e^{j\Omega})| &= |\text{di}_N(\Omega)| = \begin{cases} \frac{\sin N\Omega/2}{N \sin \Omega/2} & \text{für } \Omega \neq 2k\pi \\ (-1)^{k(N-1)} & \text{sonst} \end{cases} \end{aligned} \quad (\text{A.15})$$

Es gilt ferner:

$$\text{di}_N(\Omega) = \frac{\sin N\Omega/2}{N \sin \Omega/2} \approx \frac{\sin N\Omega/2}{N\Omega/2} = \text{si}(N\Omega/2) \quad \text{für } \Omega \ll 1 \quad (\text{A.16})$$

$$\text{di}_{N \rightarrow \infty}(\Omega) = \sum_{k=-\infty}^{\infty} \delta(\Omega - 2k\pi) \quad (\text{Dirac-Schar, ohne Beweis}) \quad (\text{A.17})$$

A.5.2. Verwandte Reihen

Unendlich: $\sum_{k=0}^{\infty} ka^k = \frac{a}{(a-1)^2}$ für $|a| < 1$ (A.18)

Endlich: $\sum_{k=0}^{N-1} ka^k = \frac{(N-1)a^{N+1} - Na^N + a}{(a-1)^2}$ für $a \neq 1$ (A.19)

Unendlich: $\sum_{k=0}^{\infty} k^2 a^k = \frac{a(1+a)}{(a-1)^3}$ für $|a| < 1$ (A.20)

Endlich: $\sum_{k=0}^{N-1} k^2 a^k = \frac{(N-1)^2 a^{N+2} - (2N^2 - 2N - 1)a^{N+1} + N^2 a^N - a^2 - a}{(a-1)^3}$ für $a \neq 1$ (A.21)

A.5.3. Einfache Reihen

Mit Hilfe der folgenden einfachen Reihen kann man z.B. Rechnungen zu Fensterfunktionen für die Fouriertransformation einfacher durchführen.

Lineare Reihe $\sum_{k=1}^n k = \frac{n(n+1)}{2}$ (A.22)

Quadratische Reihe $\sum_{k=1}^n k^2 = \frac{n(n+1)(2n+1)}{6}$ (A.23)

Kubische Reihe $\sum_{k=1}^n k^3 = \left(\frac{n(n+1)}{2}\right)^2$ (A.24)

Lösungen für beliebige ganzzahlige Exponenten gibt die Faulhabersche Formel.



Abb. A.2.: Gefährliche Fußnoten [<http://xkcd.com/1184/>]

A.6. Reihenentwicklung

Zur Annäherung der trigonometrischen Funktionen werden die Taylorreihen um den Punkt 0 herum berechnet (MacLaurinsche Reihe), x muss dabei in Bogenmaß (rad) angegeben werden:

$$\sin(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!} = x - \frac{x^3}{6} + \frac{x^5}{120} \mp \dots \quad (\text{A.25})$$

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!} = 1 - \frac{x^2}{2} + \frac{x^4}{24} \mp \dots \quad (\text{A.26})$$

$$\tan(x) = x + \frac{x^3}{3} + \frac{2x^5}{15} + \dots \quad \text{für } |x| < \frac{\pi}{2} \quad (\text{A.27})$$

$$\arctan(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{2k+1} = x - \frac{x^3}{3} + \frac{x^5}{5} \mp \dots \quad \text{für } |x| \leq 1 \text{ und } x \neq \pm j \quad (\text{A.28})$$

A.7. Ein paar Integrale

$$\int_{t=0}^{T_1} \cos^2(2\pi k f_1 \tau) d\tau = \frac{\tau}{2} + \frac{\sin(2\pi k f_1 \tau) \cos(2\pi k f_1 \tau)}{2 \cdot 2\pi k f_1} \Big|_{\tau=0}^{T_1} = T_1/2 \quad (\text{A.29})$$

$$\int_{t=0}^{T_1} \sin^2(2\pi k f_1 \tau) d\tau = \frac{\tau}{2} - \frac{\sin(2\pi k f_1 \tau) \cos(2\pi k f_1 \tau)}{2 \cdot 2\pi k f_1} \Big|_{\tau=0}^{T_1} = T_1/2 \quad (\text{A.30})$$

B. Fourier-Transformation

B.1. Fourier-Transformation kontinuierlicher Signale

$$\text{Fourierintegral: } s(t) = \int_{-\infty}^{\infty} S(f) e^{j2\pi f t} df \quad \circ \bullet \quad S(f) = \int_{-\infty}^{\infty} s(t) e^{-j2\pi f t} dt \quad (\text{B.1})$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega t} d\omega \quad \circ \bullet \quad S(\omega) = \int_{-\infty}^{\infty} s(t) e^{-j\omega t} dt \quad (\text{B.2})$$

Anmerkung: Aufgrund der Symmetrie und da die Integration jeweils über $-\infty$ bis ∞ läuft, ist die Wahl der Vorzeichen in der Hin- und Rücktransformation im Prinzip willkürlich. Auch der Vorfaktor kann beliebig auf Hin- und Rückrichtung aufgeteilt werden. In diesem Skript wird der Faktor immer so gewählt, dass $S(0)$ dem DC-Wert entspricht, in der Literatur (und in Software-Bibliotheken!) gibt es aber auch andere Definitionen.

Die **Fourierreihe** erhält man für Signale, die mit $T_1 = 1/f_1$ periodisch sind. Hier ergeben sich diskrete Linienspektren mit komplexen Koeffizienten $c_k \equiv S[kf_1]$:

$$\text{Fourierreihe: } s(t) = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi k f_1 t} \quad \circ \bullet \quad c_k \equiv S[kf_1] = \frac{1}{T_1} \int_{-T_1/2}^{T_1/2} s(t) e^{-j2k\pi f_1 t} dt; \quad k \in \mathbb{Z} \quad (\text{B.3})$$

$$= \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos 2\pi k f_1 t + b_k \sin 2\pi k f_1 t \quad \circ \bullet \quad a_k \equiv \Re \{S[kf_1]\} = \frac{1}{T_1} \int_{-T_1/2}^{T_1/2} s(t) \cos(2k\pi f_1 t) dt; \quad k \in \mathbb{N}_0 \quad (\text{B.4})$$

$$b_k \equiv \Im \{S[kf_1]\} = \frac{1}{T_1} \int_{-T_1/2}^{T_1/2} s(t) \sin(2k\pi f_1 t) dt; \quad k \in \mathbb{N} \quad (\text{B.5})$$

Ähnlichkeit:	$a \cdot s(bt)$	○—●	$\frac{a}{ b } S\left(\frac{f}{b}\right)$	(B.6)
Diracstoß:	$\delta(t)$	○—●	1	(B.7)
Rect-Puls:	$\text{rect}\left(\frac{t}{T_w}\right)$	○—●	$T_w \frac{\sin \pi T_w f}{\pi T_w f} = T_w \text{si}(\pi T_w f)$	(B.8)
si-Puls:	$\text{si}(f_0 t)$	○—●	$\frac{1}{f_0} \text{rect} \frac{f}{f_0}$	(B.9)
Period. δ – Funktion:	$\sum_{n=-\infty}^{\infty} \delta(t - nT)$	○—●	$\frac{1}{T} \sum_{n=-\infty}^{\infty} \delta\left(f - \frac{n}{T}\right)$	(B.10)
Ideale Abtastung:	$s(t) \sum_{n=-\infty}^{\infty} \delta(t - nT)$	○—●	$\frac{1}{T} \sum_{n=-\infty}^{\infty} S\left(f - \frac{n}{T}\right)$	(B.11)
Cosinus-Fkt.:	$\cos 2\pi f_0 t$	○—●	$\frac{1}{2} [\delta(f - f_0) + \delta(f + f_0)]$	(B.12)
Sinus-Fkt.:	$\sin 2\pi f_0 t$	○—●	$\frac{j}{2} [\delta(f - f_0) - \delta(f + f_0)]$	(B.13)
BP-Transformation:	$s(t) \cdot \cos 2\pi f_0 t$	○—●	$\frac{1}{2} [S(f - f_0) + S(f + f_0)]$	(B.14)
Frequenzverschiebung:	$s(t) \cdot e^{j2\pi f_0 t}$	○—●	$S(f - f_0)$	(B.15)
Verzögerung:	$s(t - \Delta T)$	○—●	$S(f) e^{-j2\pi f \Delta T} = S(f) e^{-jf \Delta \phi T_0}$	(B.16)
			mit $\Delta \phi \doteq \frac{2\pi \Delta T}{T_0}$	
Integrator:	$\int_{-\infty}^t s(\tau) d\tau$	○—●	$\frac{S(f)}{j2\pi f} + \frac{1}{2} S(0) \delta(f)$	(B.17)
Einheitssprung:	$u(t)$	○—●	$\frac{\delta(f)}{2} + \frac{1}{j2\pi f}$	(B.18)
ZOH:	$\text{rect}\left(\frac{t}{\tau}\right) * \left[s(t) \sum_{n=-\infty}^{\infty} \delta(t - nT) \right]$	○—●	$\frac{\tau}{T} \text{si}(\pi \tau f) \sum_{n=-\infty}^{\infty} S\left(f - \frac{n}{T}\right)$	(B.19)

$$\text{Period. Signal: } \sum_{n=-\infty}^{\infty} s(t - nT) = s(t) \star \sum_{n=-\infty}^{\infty} \delta(t - nT) \circledcirc \bullet \frac{S(f)}{T} \sum_{n=-\infty}^{\infty} \delta\left(f - \frac{n}{T}\right) \quad (\text{B.20})$$

$$\text{Period. rect-Signal: } \text{rect}\left(\frac{t}{T_w}\right) \star \sum_{n=-\infty}^{\infty} \delta(t - nT) \circledcirc \bullet \sum_{n=-\infty}^{\infty} \frac{T_w}{T} \sin\left(\frac{\pi n T_w}{T}\right) \delta\left(f - \frac{n}{T}\right) \quad (\text{B.21})$$

B.2. Fourier-Transformation diskreter Signale

Die Fouriertransformation einer *unendlichen* mit T_S abgetasteten äquidistanten Folge von Abtastwerten $s[n] \equiv s[nT_S]$ ist die **Discrete Time Fourier Transform** oder **DTFT**:

$$\text{DTFT: } s[n] \equiv s[nT_S] = \int_{-\infty}^{\infty} S(f) e^{j2\pi f n T_S} df \quad \circledcirc \bullet \quad S(f) = \sum_{n=-\infty}^{\infty} s[n] e^{-j2\pi f n T_S} \quad (\text{B.22})$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega n T_S} d\omega \quad \circledcirc \bullet \quad S(\omega) = \sum_{n=0}^{N-1} s[n] e^{-j\omega n T_S} \quad (\text{B.23})$$

Für eine periodische Funktion mit N Samples pro Periode T_1 erhält man die **Diskrete Fourier Transformation** oder **DFT**. Mit $T_1 = NT_S$ bzw. $f_1 = f_S/N$ kann die DFT aus der Fourierreihe oder aus der DTFT abgeleitet werden:

$$\begin{aligned} \text{DFT: } s[n] \equiv s[nT_S] &= \sum_{k=0}^{N-1} S[kf_S/N] e^{j2\pi n T_S k f_S / N} \quad \circledcirc \bullet \quad S[k] \equiv S[kf_1] = \frac{1}{N} \sum_{n=0}^{N-1} s[nT_S] e^{-j2\pi n T_S k f_S / N} \\ &= \sum_{k=0}^{N-1} S[k] e^{j2\pi k n / N} = \sum_{k=0}^{N-1} S[k] W_k^n &= \frac{1}{N} \sum_{n=0}^{N-1} s[n] e^{-j2\pi k n / N} = \frac{1}{N} \sum_{n=0}^{N-1} s[n] W_k^{-n} \quad (\text{B.24}) \end{aligned}$$

mit $W_k \equiv e^{j2\pi k / N}$

C. z -Transformation

Definition:	$x[n]$	o—•	$\mathcal{L}\{x[n]\} = X(z) = \sum_{n=-\infty}^{\infty} z^{-n} x[n]$	(C.1)
Linearität:	$a_1 x_1[n] + a_2 x_2[n]$	o—•	$a_1 X_1(z) + a_2 X_2(z)$	(C.2)
Diracstoß:	$\delta[n]$	o—•	1	(C.3)
Faltung:	$x_1[k] * x_2[k]$	o—•	$X_1(z)X_2(z)$	(C.4)
Verzögerung um k:	$x[n - k]$	o—•	$X(z)z^{-k}$	(C.5)
Spiegelung:	$x[-n]$	o—•	$X(-z)$	(C.6)
Einheitssprung:	$u[n]$	o—•	$\frac{z}{z - 1} = \frac{1}{1 - z^{-1}}$	(C.7)
Rampe:	$nu[n]$	o—•	$\frac{z}{(z - 1)^2} = \frac{z^{-1}}{(1 - z^{-1})^2}$	(C.8)
Exponentielle Folge:	$a^n u[n]$	o—•	$\frac{z}{z - a} = \frac{1}{1 - az^{-1}}$	(C.9)
Exponentielle Folge:	$a^{n-1} u[n]$	o—•	$\frac{1}{z - a} = \frac{z^{-1}}{1 - az^{-1}}$	(C.10)
Upsampling um L:	$y[n] = \begin{cases} x[n/L] & \text{für } n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{sonst (Nullenstopfen)} \end{cases}$	o—•	$Y(z) = X(z^L)$	(C.11)
Downsampling um R:	$y[n] = x[Rn]$	o—•	$Y(z) = \frac{1}{R} \sum_{k=0}^{R-1} X(z^{1/R} e^{j2\pi k/R})$	(C.12)

Abbildungsverzeichnis

1	XKCD: Latex rules	ii
0.2	XKCD: Was Sie in diesen Unterlagen finden	1
0.3	XKCD: Darf's etwas mehr sein?	2
0.4	XKCD: Python hebt ab!	3
0.5	Beispiel für Plot, erstellt mit Python	4
1.1	Beispiel für einen Signalflussgraphen	12
1.2	XKCD: Flowchart als Flowchart	12
1.3	Zusammenfassen von seriellen (a) und parallelen (b) Zweigen	13
1.4	Distributivität	14
1.5	Eliminieren von Schleifen, (a) Eigenschleifen und (b) allgemeine Schleifen	14
1.6	Moving Average (MA) (a), kaskadiertes MA - Filter (b) und verlustbehafteter (gedämpfter) Integrator (c)	19
1.7	FIR-Filter mit Eingangssignal	20
1.8	IIR-Systeme zweiter Ordnung mit identischer Systemfunktion in Direktform 1 und Direktform 2	21
1.9	Filterstrukturen zu Aufgabe 1.6, 1.7, 2.6 und 12.1	22
2.1	XKCD: Imaginäre Erklärungen	26
2.2	Pol-/Nullstellenpläne zu Aufgabe 2.5	30
2.3	FIR-Filterstruktur und Amplitudengang	31
2.4	IIR-Filter (a) mit Eingangssignal (b) zu Aufgabe 2.9	33
2.5	Periodisches Eingangssignal zu Aufgabe 2.9h)	33
3.1	XKCD: That cat has some serious periodic components	36
3.2	Signalflussgraph einer DFT mit 8 Punkten	37
3.3	Signalflussgraph einer FFT mit 8 Punkten	38
3.4	Python Plot zu Listing 3.1	40
3.5	Seismogramm eines Erdbebens am 7.11.2012 in Guatemala. Mit freundlicher Genehmigung der Bundesanstalt für Geowissenschaften und Rohstoffe [http://www.seismologie.bgr.de/sdac/erdbeben/big_quakes/guatemala_121107_deu.html]	41
4.1	Filterspezifikationen für FIR und IIR-Filter	45
4.2	Amplitudengang des Filters F im (a) linearen und (b) logarithmischen Maßstab	53
4.3	XKCD: Konjugiert <i>und</i> komplex	54
4.4	Betragsfrequenzgang zu Aufgabe 4.6	56
5.1	XKCD: Schlaflose Programmierer	60
5.2	Moving Average Filter mit Quantisierung	64
5.3	FIR-Filter mit Quantisierung zu Aufgabe 5.3	64

5.4	Betragsfrequenzgang zu Aufgabe 5.4	65
5.5	Rekursive Filterstruktur H zu Aufgabe 5.5	66
5.6	Integratoren mit endlicher Wortbreite	67
6.1	XKCD: And if you labeled your axes	71
6.2	XKCD: Korrelation	72
6.3	Auswirkung der Signalquantisierung im Zeit- und Frequenzbereich	78
6.4	Spektrale Leistungsdichte am Ausgang eines ADCs	80
7.1	XKCD: Abtasten ist gefährlich	83
7.2	Amplituden- und Phasenspektren von zeitkontinuierlichen und abgetasteten Signalen	85
7.3	Zweistufige Dezimation zu Aufgabe 7.4	86
7.4	Multiraten-Tiefpassfilter und Betragsgang der Teilfilter zu Aufgabe 7.5	87
8.1	Spektrum von $s[n]$ vor (a) und nach Abtastratenerhöhung um $I = 3$ (b)	92
8.2	DAC ohne (a) und mit (b) Oversampling zu Aufgabe 8.5	95
8.3	ZOH-Signal $x_2(t)$ und Amplitudenspektrum $ X_2(f) $ zu Aufgabe 8.5a	95
8.4	Spezifikationen für Interpolationsfilter H_{int} (a) und Rekonstruktionstiefpässe $H_{RKTP,i}$ (b) zu Aufgabe 8.5b und d)	96
8.5	ZOH-Signal $x_5(t)$ und Amplitudenspektrum $X_5(f)$ zu Aufgabe 8.5c	96
8.6	Zweistufige Interpolation zu Aufgabe 8.6	97
9.1	XKCD: Wenn Ingenieure blinken	99
9.2	Synchrone Abtastratenwandlung zu Aufgabe 9.1	101
9.3	Abtastratenwandlung zu Aufgabe 9.2 und 9.3	102
9.4	Spektren und Frequenzgänge zu Aufgabe 9.2	103
10.1	Zweistufiges CIC-Filter zu Aufgabe 10.1a)	105
10.2	Zweistufiges CIC-Filter mit Dezimation zu Aufgabe 10.1b)	106
10.3	Zweistufiges CIC-Filter mit Dezimation zu Aufgabe 10.1d) (Hogenauer-Struktur)	106
11.1	Amplitudengang $ H(f) $ des analogen Referenzfilters	112
12.1	XKCD: Matrixtransformation	113
12.2	Allgemeine Struktur eines zeitdiskreten Systems n -ter Ordnung in Zustandsraumbeschreibung mit dem Vektor der Zustandsgrößen $s[n]$	113
M1.1	XKCD: Flowchart	119
M1.2	Eingangssignal $x(t) = 1,5 \text{ V} + 0,5 \text{ V} \cos(2\pi \cdot 50 \text{ Hz} \cdot t)$, abgetastet mit verschiedenen Samplingraten f_S	120
M1.3	MA-Filter der Ordnung $M - 1$ und effiziente Implementierung mit kritischen Pfaden	124
M1.4	FIR-Filter mit Eingangssignal	126
M1.5	FIR Filter in Direktform und in „gefalteter Form“	128
M1.6	Systeme zweiter Ordnung in Direktform 1 und 2 mit kritischen Pfaden	129
M1.7	Kritische Pfade zu Aufgabe M1.6	131
M1.8	Transponierte Filterstrukturen zu Abb. 1.9 mit kritischen Pfaden	132
M2.1	PN-Diagramme zu Aufgabe 2.1	139
M2.2	Betragss- und Phasengang sowie Gruppenlaufzeit zu Aufgabe 2.1	142
M2.3	Pol/Nullstellenplan zu Aufgabe 2.2	144

M2.4	Betrag- und Phasengang zu Aufgabe 2.2	145
M2.5	Blockdiagramme zu Abb. 1.9c) und d)	149
M2.6	MA-Filter mit $N = 32$ Taps, (a) Amplitudengang und (b) P/N-Diagramm	153
M2.7	System a)	154
M2.8	Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8a	154
M2.9	Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8a	155
M2.10	System b)	155
M2.11	Pol-Nullstellen-Plan (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8b	155
M2.12	Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8b	156
M2.13	System c)	156
M2.14	Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8c	157
M2.15	Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8c	157
M2.16	System d)	157
M2.17	Pol-Nullstellen-Plan für $\alpha = 0,5$ (a) und Amplitudenfrequenzgang (b) zu Aufgabe 2.8d	158
M2.18	Impulsantwort für $\alpha = 0,5$ (a) und $\alpha = 0,9$ (b) zu Aufgabe 2.8d	158
M2.19	Pol-Nullstellen-Plan (a) und Amplitudenfrequenzgang (b) zu Abb. 2.4(a)	162
M3.1	Korrigiertes Python Skript und Plot zur Simulation des Frequenzgangs zu Aufgabe 3.2	170
M3.2	Eingangssignal $s(t)$ und abgetastetes Signal $s[n]$ (zwei Perioden) zu Aufgabe 3.5	175
M3.3	Betrag und Phase der DFT $S[k]$ für $f_S = 3000$ Hz und $N = 3$ zu Aufgabe 3.5c	176
M3.4	Betrag und Phase der DFT $S[k]$ für $N = 6$ mit $f_S = 3000$ Hz zu Aufgabe 3.5e (a) und mit $f_S = 6000$ Hz zu Aufgabe 3.5f (b)	177
M3.5	Erwartetes ein- und zweiseitiges Spektrum zu Aufgabe 3.6	179
M3.6	Betrag und Phase der DFT $S[k]$ für $f_S = 4000$ Hz und $N = 4$ (a) bzw. $f_S = 100$ kHz und $N = 100$ (b) zu Aufgabe 3.7b	183
M4.1	Frequenzgänge und Spezifikationen im linearen und log. Maßstab sowie P/N-Diagramme für IIR- und FIR-Filter zu Aufgabe 4.1	188
M4.2	Amplitudengang mit Nullstellen	191
M4.3	P/N-Diagramm	192
M4.4	Plots zu maximalphasigem Filter F_1 (Aufgabe 4.3b)	196
M4.5	Plots zu minimalphasigem Filter F_2 (Aufgabe 4.3d)	198
M4.6	Plots zu linearphasigem Filter F (Aufgabe 4.3e)	199
M4.7	Blockschaltbild (a) und Implementierung (b) für Filter F (Aufgabe 4.3e)	200
M4.8	Betragsfrequenzgänge zu Aufgabe 4.6	204
M5.1	Moving Average Filter mit Quantisierung	210
M5.2	FIR-Filter mit optimierter Struktur und Quantisierung zu M5.3	210
M5.3	SFG in Direktform zu Aufgabe 5.4	212
M5.4	IIR-Filter mit Skalierung	214
M5.5	Optimiertes IIR-Filter mit Skalierung des transversalen Teils	214
M5.6	Phasengang des IIR-Filters	215
M6.1	Einseitiges Spektrum von Schmal- und Breitbandsignalen	226
M7.1	Amplituden- und Phasenspektren von zeitkontinuierlichen und abgetasteten Signalen	229

M7.2	Zeitkontinuierliche Signale mit identischen Abtastwerten	230
M7.3	Zweistufige Dezimation, mit (a) und ohne (b) Aliasing ins Basisband am Ausgang zu Aufgabe 7.4a)	232
M7.4	Frequenzgänge der Multiraten-Filterkaskade	235
M8.1	Spektrum von $s[n]$ (a) und nach Abtastratenerhöhung um $I = 3$ (b)	237
M8.2	Ersatzschaltbild für Interpolation mit ZOH zu Aufgabe 8.2 d)	239
M8.3	ZOH-Signal $x_2(t)$ und Amplitudenspektrum $ X_2(f) $ zu Aufgabe 8.5a	242
M8.4	Spezifikationen für Interpolationsfilter H_{int} (a) und Rekonstruktionstiefpass $H_{RKTP,i}$ (b) zu Aufgabe 8.5b und d)	242
M8.5	ZOH-Signal $x_5(t)$ und Amplitudenspektrum $X_5(f)$ zu Aufgabe 8.5c	243
M9.1	Synchrone Abtastratenwandlung zu Aufgabe 9.1	245
M9.2	Abtastratenwandlung zu Aufgabe 9.2	247
M9.3	Spektren und Frequenzgänge zu Aufgabe 9.2	248
M9.4	Alternative Abtastratenwandlung zu Aufgabe 9.2 (vertauschte Dezimation und Interpolation)	248
M9.5	Alternative Abtastratenwandlung zu Aufgabe 9.2 (zusammengefasstes TP-Filter)	249
M11.1	Betragsfrequenzgänge zu Aufgabe 11.1	258
M11.2	Resonator in Direktform Typ 1	261
M11.3	Resonator in Direktform Typ 2	261
M11.4	Stabilitätsgebiet eines Systems zweiter Ordnung	262
M12.1	Signalfussplan zu Aufgabe 12.1c; (a) korrekte und (b) falsche Lösung	266
A.1	Komplexe z -Ebene	269
A.2	XKCD: Gefährliche Fußnoten	272

Tabellenverzeichnis

1.1	Spezielle Befehle zu Kap. 1	15
1.2	Simulationsfiles zu Kap. 1	16
1.3	Ergebnisse zu Aufgabe 1.1	18
2.1	Spezielle Befehle zu Kap. 2	27
2.2	Simulationsfiles zu Kap. 2	28
3.1	Spezielle Befehle zu Kap. 3	38
3.2	Simulationsfiles zu Kap. 3	39
4.1	Spezielle Befehle zu Kap. 4	51
4.2	Simulationsfiles zu Kap. 4	51
4.3	Verständnisfragen zu Filtern (Aufgabe 4.8)	57
5.1	Python-Files zu Kap. 5	62
5.2	Verständnisfragen zu Filtern mit Quantisierung (Aufgabe 5.8)	69
6.1	Simulationsfiles zu Kap. 6	77
7.1	Spezielle Befehle zu Kap. 7	84
7.2	Simulationsfiles zu Kap. 7	84
7.3	Eigenschaften der Filterkaskade aus Aufgabe 7.5	88
8.1	Spezielle Befehle zu Kap. 8	90
9.1	Spezielle Befehle zu Kap. 9	100
9.2	Lösungen zu Aufgabe 9.2	103
10.1	Simulationsfiles zu Kap. 10	105
11.1	Simulationsfiles zu Kap. 11	111
M1.1	Ergebnisse zu Aufgabe 1.1	121
M1.2	Zeitdiskrete Faltung $y[n] = h[n] * x[n]$ (Aufgabe 1.2)	127
M1.3	Zeitdiskrete Faltung $y[n] = x[n] * h[n]$ (Aufgabe 1.2)	127
M2.1	Dämpfungen der verschiedenen Filter aus Aufgabe M2.1 bei $f_1 = 50$ Hz	138
M2.2	Daten zu Aufgabe 2.10	165
M2.3	Ergebnisse zu Aufgabe M2.11	166
M3.1	Anzahl der Rechenoperationen für eine FFT in Abhängig von der Anzahl der Datenpunkte ($f_S = 44,1$ kHz, Aufgabe 3.4)	173
M4.1	Eckfrequenzen für den Filterentwurf zu Aufgabe 4.1d	190
M4.2	Nullstellen des Zählerpolynoms	192

M4.3 Hilfswerte für Filter F ₁ (Aufgabe 4.3b)	195
M4.4 Hilfswerte für Filter F ₂ (Aufgabe 4.3c)	197
M4.5 Musterlösung zu Aufgabe 4.8	206
M5.1 Wortlängenwachstum zu Aufgabe M5.2	209
M5.2 Filterkoeffizienten in verschiedenen Zahlenformaten zu M5.3	211
M5.3 Musterlösung zu Aufgabe 5.8	219
M6.1 Zusammenfassung der Ergebnisse aus Aufgabe M6.3	224
M7.1 Eigenschaften der Filterkaskade aus Aufgabe 7.5	234
M8.1 si-Dämpfungen für DAC ohne Oversampling (Abb. 8.2(a))	241
M8.2 si-Dämpfungen für DAC mit Oversampling (Abb. 8.2(b))	243
M8.3 Dämpfung der Images im zweistufigen Interpolator (Aufgabe 8.6)	244
M9.1 Werte zu Abb. M9.2	250
M9.2 Werte zu Abb. M9.4 bzw. M9.5	251
M9.3 Ergebnisse zu Aufgabe 9.2	252
M10.1 CIC-Filter mit $f_{S,Ein} = 26$ MHz, $N = 2$ und $R = 32$	255

Listings

1	Gemeinsamer Python-Header für diesen Kurs	5
2	Defaultsettings für hübschere Matlab-Plots	5
1.1	Faltung mit Python	17
1.2	Faltung mit Matlab	17
1.3	Python Tricks zu Kap. 1 (LTI/LTI_tricks_py.py)	17
1.4	Matlab Tricks zu Kap. 1 (LTI/LTI_tricks_m.m)	17
3.1	Python Skript zu Aufgabe 3.2 (erster Ansatz)	40
3.2	DFT mit Python	42
3.3	DFT mit Matlab	42
4.1	Windowed FIR-Filterentw. mit Python	49
4.2	Windowed FIR-Filterentwurf mit Matlab	49
4.3	Frequency Sampling FIR-Filterentwurf mit Python	50
4.4	Frequency Sampling FIR-Filterentwurf mit Matlab	50
4.5	Equiripple FIR-Filterentw. mit Python	50
4.6	Equiripple FIR-Filterentwurf mit Matlab	50
5.1	Quantisierung eines Sinussignals und Quantisierungskennlinie mit Python (FIX/FIX_intro_py.py)	61
6.1	Quantisierungsrauschen mit Python (NOI/NOI_intro_py.py)	76
6.2	DFT eines Breitbandsignals (erster Ansatz, NOI/NOI_DFT_wideband_py.py)	81
8.1	Upsampling mit Python	90
8.2	Upsampling mit Matlab	90
9.1	Interpolation / Resampling mit Python	100
11.1	IIR-Filterentwurf mit Python	110
11.2	IIR-Filterentwurf mit Matlab	110
M1.1	Python Listing zu M1.1	124
M1.2	Matlab Listing zu M1.1	124
M1.3	Python-Implementierung der gcf - Funktion zu Aufgabe M1.1	125
M2.1	Python Listing zu M2.1	142
M2.2	Matlab Listing zu M2.1	142
M2.3	Python Listing zu M2.7f	153
M2.4	Matlab Listing zu M2.7f	153
M2.5	Python Listing zu M2.8	159
M2.6	Matlab Listing zu M2.8	159
M2.7	Python Listing zu M2.11	166
M2.8	Matlab Listing zu M2.11	166

code/DFT/Basics/DFT_MA_Filt_ML_py.py	170
M3.1 Python Listing zu Aufgabe 3.5	178
M3.2 Matlab Listing zu Aufgabe 3.5	178
M3.3 Symbolische Integration mit SymPy	181
M3.4 Python Listing zu Aufgabe M3.6	181
M3.5 Matlab Listing zu M3.6	181
M3.6 Matlab-Code zu Aufgabe 3.8	185
M4.1 Python Listing zu Aufgabe 4.2	193
M5.1 Python Listing zu Aufgabe 5.3	211
M6.1 DFT eines Breitbandsignals, verbesserte Lösung (NOI/NOI_DFT_wideband_ML_py.py)	225

Literaturempfehlungen

Grundlagen - kurz und knapp

[Gen11a] N. Geng, *Aufgabensammlung zu Signale und Systeme Teil 2: Zeitdiskrete Signale und Systeme*, Hochschule München, Feb. 2011, 128 S.

[Gen11b] _____, *Skriptum zu Signale und Systeme Teil 2: Zeitdiskrete Signale und Systeme*, Hochschule München, Feb. 2011, 128 S.

Inhalt von Skript und Aufgabensammlung zur Vorlesung „Signale und Systeme“ - wird als bekannt vorausgesetzt!

[Leh] Lehrstuhl für Nachrichtentechnik an der TU München, *Ein Lerntutorial für Nachrichtentechnik im world wide web*, <http://www.lntwww.de/>.

Online-Tutorials zu verschiedenen Themen („Bücher“) der Nachrichtentechnik. Für die Vorbereitung auf diese Vorlesung besonders empfohlen:

- Signaldarstellung (ganzes Buch, vor allem Kap. 5)
- Stochastische Signaltheorie (nur Kap. 3)

[vG08] D. C. von Grünigen, *Digitale Signalverarbeitung - Bausteine, Systeme, Anwendungen*, 4. ed., Carl Hanser Verlag, München, 2008, 347 S., 29 €.

Schwerpunkte: Signale, DFT und Filter, ohne Multiratensysteme. Besonders empfohlen für Einsteiger und zum Selbststudium. Mit Matlab- und Labview- Übungen.

[Wer08] M. Werner, *Signale und Systeme*, Vieweg + Teubner, 2008.

Knappe Zusammenfassung der Grundlagen der analogen und digitalen Signalverarbeitung, das Buch geht auch kurz auf Abtastung, Interpolation und Resampling ein, auch Quantisierungsrauschen und allgemein stochastische Prozesse werden kurz behandelt. Gut geeignet, um „verschüttetes“ DSV und Nachrichtentechnikwissen wieder aufzufrischen, auch begleitend zur Vorlesung. Da der Stoff hier recht knapp abgehandelt wird, wird eher das vorige Buch zum Selbststudium empfohlen.

[Wer09] M. Werner, *Digitale Signalverarbeitung mit Matlab: Grundkurs*, 4. ed., Vieweg + Teubner, Wiesbaden, 2009, 294 S., 29,90 €.

Knapper aber anschaulicher Grundlagenkurs, auch zum Selbststudium geeignet. Ohne Multiratensysteme, MATLAB-Programme zum Download. Achtung - der zweite Band heißt sehr ähnlich - „Digitale Signalverarbeitung mit MATLAB-Praktikum“, beschäftigt sich aber mit spezielleren Themen (u.a. Zustandsraumdarstellung digitaler Systeme, Skalierung und Koeffizientenquantisierung, adaptive Systeme)!

[Dob07] G. Doblinger, *Zeitdiskrete Signale und Systeme - Eine Einführung in die grundlegenden Methoden der digitalen Signalverarbeitung*, 1st ed., J. Schlembach Fachverlag, Wilburgstetten, 2007, 230 S., 24,90 €.

Knapp, aber gut erklärt mit vielen Übungen. Matlab / Octave - Beispiele zum Download. Mit Einführung in Multiratensysteme, aber ohne Wortlängeneffekte.

- [Orf10] S. J. Orfanidis, *Introduction to Signal Processing*, 2. ed., Prentice Hall, Inc., 2010, 783 S., <http://www.ece.rutgers.edu/~orfanidi/intro2sp>.

Ein Klassiker der Signalverarbeitung, der inzwischen auch als PDF unter obiger URL kostenlos erhältlich ist. Die Themen werden in eher ungewöhnlicher Reihenfolge behandelt; wenn Sie mit anderen Büchern nicht zureckkommen, versuchen Sie mal dieses hier, es werden auch Multiratensysteme behandelt, Wortlängeneffekte werden nur angerissen. Viele Übungsaufgaben mit Lösungen (extra File)!

- [Smi99] S. W. Smith, *The scientist and engineer's guide to digital signal processing*, 2nd ed., California Technical Publishing, San Diego, USA, 1999, <http://www.DSPguide.com>.

Sehr gute und ausführliche Erklärungen, vor allem zu DFT und Filtern! Online kostenlos erhältlich.

- [Smi07] J. O. Smith, *Introduction to digital filters*, <http://ccrma.stanford.edu/~jos/filters05/>, Sep. 2007, 480 S., 46 \$.

Sehr gut erklärte Grundlagen zu digitalen Filtern, auch zu Filtertopologien, mit Matlab-Beispielen. Online unter obiger URL kostenlos (im HTML-Format).

Grundlagen - ausführlicher

- [OS10] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*, 3rd ed., Pearson, 2010, 1120 S., ab 50 €.

Didaktisch exzellente Einführung über die digitale Signalverarbeitung (über 35 Jahre „gereift“) mit vielen Übungsaufgaben. Auch für das Selbststudium gut geeignet, auch auf Deutsch und in der Hochschulbibliothek erhältlich. Begleitender Kurs mit gleichem Titel bei MIT OpenCourseWare (ocw.mit.edu), Course 6.341.

- [KK09] K.-D. Kammerer and K. Kroschel, *Digitale Signalverarbeitung*, 7. ed., Vieweg + Teubner, Wiesbaden, 2009, 588 S., 39,90 €.

Sehr gute, in die Tiefe gehende Darstellung der Grundlagen (ohne Multiratensysteme). Außerdem Verfahren zur Spektralschätzung (nicht Teil der Vorlesung). Umfangreicher MATLAB-Übungsteil.

- [Kar12] U. Karrenberg, *Signale - Prozesse - Systeme: Eine multimediale und interaktive Einführung in die Signalverarbeitung*, 6. ed., Springer, 2012, 532 S. mit DVD, 49,95 €.

Didaktisch sehr guter DSV-Kurs, beginnend bei den Grundlagen bis hin zu Quell- und Kanalkodierung und neuronalen Netzen. Großer Wert wird auf den Zusammenhang zwischen Zeit- und Frequenzbereich gelegt. Zu allen Kapiteln gibt es auf einer mitgelieferten DVD umfangreiche Experimente und Übungsaufgaben in DASYLab, einem Labview-ähnlichen System. Auf der DVD sind auch die Software (und eine Lizenz) enthalten.

- [HQ07] J. Hoffmann and F. Quint, *Signalverarbeitung mit MATLAB und Simulink: Anwendungsorientierte Simulationen*, Oldenbourg, 2007, 436 S., 39,80 €.

Schwerpunkt auf Entwurf von digitalen Filtern, inkl. Multiratensystemen und Filterbänken. Gliederung ähnlich wie Vorlesung DSV auf FPGAs. Nicht den Vorgänger (ähnlicher Titel, Addison-Wesley) kaufen!

Spezielle und weiterführende Literatur

- [Mey07] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*, 3 ed., Springer, Berlin, 2007, 774 S., ca. 90 €
Das Standardwerk für schnelle digitale Signalverarbeitung auf FPGAs.
Das Standardwerk für schnelle digitale Signalverarbeitung auf FPGAs.
- [Zöl05] U. Zölzer, *Digitale Audiosignalverarbeitung*, 3. ed., Teubner, Wiesbaden, Germany, 2005, 328 S., 44,90 €.
Das Standardwerk zur digitalen Audiosignalverarbeitung. Ausführliche Beschreibung der Vor- und Nachteile verschiedener rekursiver Filterstufen.
- [ZA11] U. Zölzer and X. Amatriain, *DAFX: digital audio effects*, 2. ed., Wiley, 2011, 624 S., 85 €.
Auf letzterem Buch aufbauend, ein weiteres Standardwerk, diesmal zu digitalen Audioeffekten.
- [Mer10] A. Mertins, *Signaltheorie: Grundlagen der Signalbeschreibung, Filterbänke, Wavelets, Zeit-Frequenz-Analyse, Parameter- und Signalschätzung*, 2. ed., Vieweg+Teubner Verlag, 2010, 393 S., 34,95 €.
Signaltheorie verschiedener Transformationen (Fourier-, Kurzzeitfourier-, Wavelet- ...), nicht-lineare Transformationen, Filterbänke. Für alle, die es immer schon etwas genauer wissen wollten ...

Spezielle Literatur zu Multiratensystemen

- [Har04] F. J. Harris, *Multirate signal processing for communication systems*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004, 478 S., ca. 90 €.
Sehr praxisnahes Buch speziell zu Multiratensystemen und effizienten Filtern.
- [CR75] R. Crochiere and L. Rabiner, *Optimum FIR digital filter implementations for decimation, interpolation, and narrow-band filtering*, Acoustics, Speech and Signal Processing, IEEE Transactions on **23** (1975), no. 5, 444 – 456.
Generelle Theorie und Implementierung von Multistage Dezimatoren und Interpolatoren, Anwendung für schmalbandige Filter
- [RC75] L. Rabiner and R. Crochiere, *A novel implementation for narrow-band FIR digital filters*, Acoustics, Speech and Signal Processing, IEEE Transactions on **23** (1975), no. 5, 457 – 464.
Detailliertere Beschreibung von schmalbandigen Filtern in Multiraten-Implementierung

- [CR81] R. E. Crochiere and L. R. Rabiner, *Interpolation and decimation of digital signals - a tutorial review*, Proc. of the IEEE **69** (1981), 300–331, Reprinted in „Oversampling Delta-Sigma Data Converters“, IEEE Press.

Sehr gute (und sehr tiefgehende) Einführung zu Dezimierung und Interpolation

Literaturverzeichnis

- [CR75] R. Crochiere and L. Rabiner, *Optimum FIR digital filter implementations for decimation, interpolation, and narrow-band filtering*, Acoustics, Speech and Signal Processing, IEEE Transactions on **23** (1975), no. 5, 444 – 456.
- [CR81] R. E. Crochiere and L. R. Rabiner, *Interpolation and decimation of digital signals - a tutorial review*, Proc. of the IEEE **69** (1981), 300–331, Reprinted in „Oversampling Delta-Sigma Data Converters“, IEEE Press.
- [Dob07] G. Doblinger, *Zeitdiskrete Signale und Systeme - Eine Einführung in die grundlegenden Methoden der digitalen Signalverarbeitung*, 1st ed., J. Schlembach Fachverlag, Wilburgstetten, 2007, 230 S., 24,90 €.
- [Gen11a] N. Geng, *Aufgabensammlung zu Signale und Systeme Teil 2: Zeitdiskrete Signale und Systeme*, Hochschule München, Feb. 2011, 128 S.
- [Gen11b] _____, *Skriptum zu Signale und Systeme Teil 2: Zeitdiskrete Signale und Systeme*, Hochschule München, Feb. 2011, 128 S.
- [GG04] A. Groth and H. G. Göckler, *Multiratensysteme abtastratenumsetzung und digitale filterbänke*, Schlembach Fachverlag, 2004, 539 S., 39,90 €.
- [Har04] F. J. Harris, *Multirate signal processing for communication systems*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004, 478 S., ca. 90 €.
- [Hog81] E. Hogenauer, *An economical class of digital filters for decimation and interpolation*, IEEE Trans. Acoust., Speech, Signal Processing **29** (1981), no. 2, 155–162.
- [HQ07] J. Hoffmann and F. Quint, *Signalverarbeitung mit MATLAB und Simulink: Anwendungsorientierte Simulationen*, Oldenbourg, 2007, 436 S., 39,80 €.
- [Kar12] U. Karrenberg, *Signale - Prozesse - Systeme: Eine multimediale und interaktive Einführung in die Signalverarbeitung*, 6. ed., Springer, 2012, 532 S. mit DVD, 49,95 €.
- [KK09] K.-D. Kammeyer and K. Kroschel, *Digitale Signalverarbeitung*, 7. ed., Vieweg + Teubner, Wiesbaden, 2009, 588 S., 39,90 €.
- [Leh] Lehrstuhl für Nachrichtentechnik an der TU München, *Ein Lerntutorial für Nachrichtentechnik im world wide web*, <http://www.lntwww.de/>.
- [Mer10] A. Mertins, *Signaltheorie: Grundlagen der Signalbeschreibung, Filterbänke, Wavelets, Zeit-Frequenz-Analyse, Parameter- und Signalschätzung*, 2. ed., Vieweg+Teubner Verlag, 2010, 393 S., 34,95 €.
- [Mey07] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*, 3 ed., Springer, Berlin, 2007, 774 S., ca. 90 €
Das Standardwerk für schnelle digitale Signalverarbeitung auf FPGAs.
- [Mey11] M. Meyer, *Signalverarbeitung: Analoge und digitale Signale, Systeme und Filter*, Vieweg + Teubner, 2011.

- [Orf10] S. J. Orfanidis, *Introduction to Signal Processing*, 2. ed., Prentice Hall, Inc., 2010, 783 S., <http://www.ece.rutgers.edu/~orfanidi/intro2sp>.
- [OS10] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*, 3rd ed., Pearson, 2010, 1120 S., ab 50 €.
- [PBW08] R. K. Pally, A. Baghdasarya, and G. Wheelock, *A tutorial on limit cycle oscillations in digital filters*, Research report for course ece 5620, Virginia Tech, USA, 2008.
- [RC75] L. Rabiner and R. Crochiere, *A novel implementation for narrow-band FIR digital filters*, Acoustics, Speech and Signal Processing, IEEE Transactions on **23** (1975), no. 5, 457 – 464.
- [Smi99] S. W. Smith, *The scientist and engineer's guide to digital signal processing*, 2nd ed., California Technical Publishing, San Diego, USA, 1999, <http://www.DSPguide.com>.
- [Smi07] J. O. Smith, *Introduction to digital filters*, <http://ccrma.stanford.edu/~jos/filters05/>, Sep. 2007, 480 S., 46 \$.
- [vG08] D. C. von Grünigen, *Digitale Signalverarbeitung - Bausteine, Systeme, Anwendungen*, 4. ed., Carl Hanser Verlag, München, 2008, 347 S., 29 €.
- [Wer08] M. Werner, *Signale und Systeme*, Vieweg + Teubner, 2008.
- [Wer09] M. Werner, *Digitale Signalverarbeitung mit Matlab: Grundkurs*, 4. ed., Vieweg + Teubner, Wiesbaden, 2009, 294 S., 29,90 €.
- [ZA11] U. Zölzer and X. Amatriain, *DAFX: digital audio effects*, 2. ed., Wiley, 2011, 624 S., 85 €.
- [ZB95] U. Zölzer and T. Boltze, *Parametric digital filter structures*, Proc. 99th Audio Engineering Society (AES) Convention, Sep. 1995.
- [Zö05] U. Zölzer, *Digitale Audiosignalverarbeitung*, 3. ed., Teubner, Wiesbaden, Germany, 2005, 328 S., 44,90 €.