Analyse, Resynthese und Interpolation von KFZ-Innengeräuschen

Diplomarbeit

von
Christian Feldbauer

eingereicht bei der
Technischen Universität Graz

durchgeführt am
Institut für Elektronische Musik
der
Universität für Musik und darst. Kunst Graz

im Rahmen des interuniversitären Diplomstudiums
Elektrotechnik-Toningenieur

Graz, im Juli 2000

Betreuer und Begutachter:
O.Univ.Prof Mag. DI Dr. Robert Höldrich

Besondern Dank möchte ich meinem Betreuer und Beurteiler, Herrn O.Univ.Prof Mag. DI Dr. Robert Höldrich für die großartige Unterstützung aussprechen. Weiters gebührt Ass.Prof. DI Dr. Gerhard Graber und DI Dr. Martin Pflüger für die Bereitstellung von Literatur und Soundmaterial großer Dank. Bedanken möchte ich mich natürlich auch noch bei meiner Freundin Birgit für ihre Unterstützung und das Korrigieren dieser Arbeit.

Zusammenfassung

Das Ziel dieser Arbeit ist ein KFZ-Innengeräusch-Simulator für Echtzeitbetrieb, um die in der Automobilindustrie üblichen Labor-Hörversuche realitätsnäher und effizienter durchführen zu können. Gegeben sind Aufnahmen aus dem Inneren eines Kraftfahrzeuges für Beschleunigungs- und Verzögerungsvorgänge bei verschiedenen Gaspedalstellungen. Es ist nun erforderlich, zwischen diesen Geräuschsignalen zu interpolieren. Hierzu wird ein einfaches 2-dimensionales Lastzustandsmodell vorgestellt, das den Betriebszustand lediglich mit Motordrehzahl und Gaspedalstellung beschreibt. Weiters erfordert die Interpolation eine Modellierung der Geräusche. Ein Signal wird deshalb in eine deterministische und eine stochastische Komponente zerlegt, die dann nach unterschiedlichen Signalmodellen parametrisiert werden. Die Arbeit beschäftigt sich mit verschiedenen Signalmodellen und dazugehörigen Analyse- und Resynthesemethoden, wie zB. SMS, QUASAR, Heterodyne-Filter, Additive Synthese oder LPC. Große Aufmerksamkeit wurde dem Vold-Kalman Order Tracking Filter gewidmet, das mit dem Verfahren der konjugierten Gradienten und passender Präkonditionierung realisiert wurde.

Abstract

The aim of this work is to develop a realtime simulator for the interior noise of a car for varying load situations. Using this simulator laboratory hearing tests which are common in the automotive industry may be executed more realistically than by using recorded sounds without any interaction with the test persons. Given run-up and slow-down signals for various accelerator pedal positions, it should be possible to interpolate between these prerecorded signals to obtain signals for other load situations. A simplified model for the load situation is defined as a two dimensional parameter space for each gear, with the parameters velocity (RPM) and the position of the accelerator pedal. Choosing a suitable signal model, the signal can be considered as a sum of a deterministic and a stochastic component. Various signal models and methods for the decomposition of deterministic and stochastic components are examined, e.g. SMS, QUASAR, Heterodyne Filter, Additive Synthese or LPC. A Vold-Kalman tracking filter was realized by using the preconditioned conjugate gradient method.

Inhaltsverzeichnis

Einleitung

Diese Arbeit beschäftigt sich mit der Analyse, der Resynthese und der Interpolation (oder Morphing) von Audiosignalen, die aus dem Innenraum eines Kraftfahrzeuges stammen. Das Ziel ist, einen KFZ-Innengeräusch-Simulator für Echtzeitbetrieb zu entwickeln, um die in der Automobilindustrie üblichen Labor-Hörversuche realitätsnäher und effizienter durchführen zu können. Eine Testperson soll dabei die Möglichkeit haben, mittels Gaspedal den Lastzustand beliebig zu variieren, der Simulator soll dann das diesem Betriebszustand entsprechende Geräusch (z.B. das eines KFZ-Prototyps) berechnen. Für Testpersonen stellt ein solcher Simulator eine wesentliche Erleichterung für die Beurteilung der Innengeräusche dar, gegenüber herkömmlichen Tests, bei denen Aufnahmen ohne einer Interaktion vorgespielt werden.

Es ist erforderlich, zwischen den Geräuschsignalen verschiedener Betriebszustände zu interpolieren. Dazu muß ein Modell gefunden werden, welches das Signal auf eine geeignete Weise beschreibt. Analyse bedeutet, die Parameter des Modells zu bestimmen.
Je nach Modell sind die verschiedenen Modifikationen (z.B. Filterung, Time Scale Modification, Pitch Shifting, Morphing, ...) mehr oder weniger leicht durchführbar, was die Wahl des Modells neben weiteren Eigenschaften wie z.B. Kompaktheit oder Rechenaufwand der Resynthese entscheidet. Ein Signalmodell stellt gleichzeitig auch die Resynthesevorschrift dar (z.B. Additive Synthese).

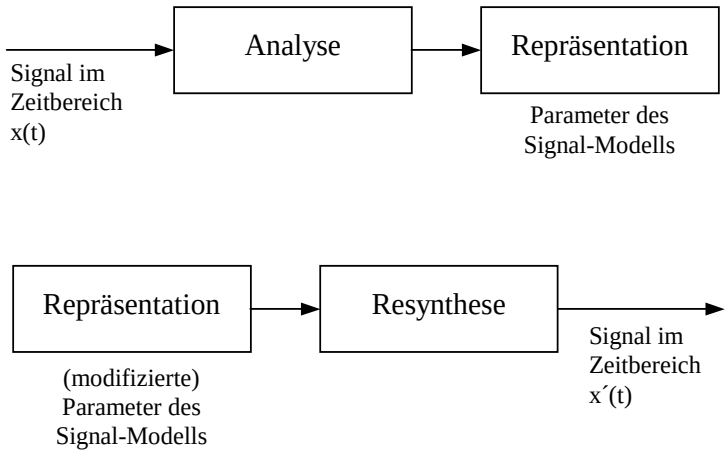


Abb. 0.1: Analyse, Repräsentation, Modifikation und Resynthese

Als genereller Ansatz für die Analyse von KFZ-Innengeräuschen gilt die Zerlegung des Signals in eine deterministische und eine stochastische Signalkomponente.

$$x(t) = x_{det}(t) + x_{sto}(t) \qquad (0.1)$$

Zuerst werden die Parameter der deterministischen Komponente $x_{det}(t)$ bestimmt, sodaß diese isoliert resynthetisiert werden kann.

Die ersten beiden Kapitel beschäftigen sich mit deterministischen Signalmodellen, Analyse- und Resyntheseverfahren. Kapitel 1 stellt grundlegendes Werkzeug zur Verfügung, Schwerpunkt sind Methoden von Xavier Serra (SMS), Robert J. McAulay und Thomas F. Quatieri. Kapitel 2 bearbeitet analysis-by-synthesis –Methoden. Dabei werden die Signalmodellparameter iterativ optimiert, d.h. in jeder Iteration wird das Signal erneut resynthetisiert, und es werden die Parameter mit dem Kriterium, die Differenz zum Originalsignal zu minimieren, modifiziert. Hierzu setzt sich diese Arbeit mit dem von Havard Vold und Jan Leuridan 1993 vorgestellten Vold-Kalman-Filter und dem QUASAR-Signalmodell von Yinong Ding und Xiaoshu Qian auseinander. Das Vold-Kalman Filter ist ein Heterodyne-Filter, das eine Least-Sqares –Aufgabe darstellt. Eine Lösung für dieses Problem mittels der Methode der konjugierten Gradienten mit passender Präkonditionierung wird vorgestellt.
Da die deterministische Komponente eines KFZ-Innengeräusches zum größten Teil durch den Verbrennungsmotor verursacht wird, ist diese harmonisch. Somit sind Heterodyne-Filter das ideale Analysewerkzeug, wenn der Grundfrequenzverlauf bzw. eine Drehzahlreferenz vorliegt.

Wenn $x_{det}(t)$ isoliert bestimmt ist, läßt sich das sogenannte Residuum (Rest oder Differenz) bilden, das als stochastische Komponente aufgefaßt wird und nach einem anderen Modell als die deterministische Komponente parametrisiert wird. Kapitel 3 beschäftigt sich mit dessen Bildung, Modellierung (LPC oder FFT-basiert) und Resynthese.

Für Kapitel 4 und Kapitel 5 wird ein einfaches Signalmodell festgelegt, das wenig Rechenaufwand zur Resynthese benötigt und Morphing leicht ermöglicht. Ein vollständiges Analyse-/Resynthesesystem für KFZ-Geräusche ist in Kapitel 4 zu finden und eine Methode zur Interpolation zwischen Signalen verschiedener Lastzustände in Kapitel 5. Hierzu wird ein einfaches Lastzustandsmodell definiert, das den Betriebszustand eines Kraftfahrzeuges für jeden Gang getrennt lediglich mit Motordrehzahl und Gaspedalstellung beschreibt. Dies bildet die Basis für den ebenfalls in diesem Kapitel vorgestellten Algorithmus für den Echtzeit-Simulator.

Kapitel 1

Bestimmung der deterministischen Komponente

Ein deterministisches Signal ist mathematisch beschreibbar, sodaß eine Vorhersage für jeden Zeitpunkt gemacht werden kann.

1.1 Stationäre deterministische Signale

Ein stationäres Signal liegt dann vor, wenn sich die Parameter mit der Zeit nicht ändern.

Stationäres deterministisches Modell:

$$x(t) = \sum_{k=1}^{K} A_k \cdot \cos(\omega_k t + \varphi_k) \qquad \text{zeitkontinuierliches Signal} \qquad (1.1)$$

$$x(n) = \sum_{k=1}^{K} A_k \cdot \cos(\theta_k n + \varphi_k) \qquad \text{zeitdiskretes Signal} \qquad (1.2)$$

wobei A_k die zeitlich konstante Amplitude, ω_k die ebenfalls konstante Kreisfrequenz und φ_k die Null-Phase des k-ten Teiltones ist. K ist die Anzahl der Teiltöne.

x(n) ist die vereinfachte Schreibweise für $x(n \cdot T_s)$ und stellt das zeitdiskrete Signal dar, das durch Abtastung von x(t) mit dem Abtastintervall T_s entstanden ist, es gilt also: $t = n \cdot T_s$ und $\theta_k = \omega_k \cdot T_s$. θ_k ist die auf 2π normierte Kreisfrequenz in Radiant oder auch das Phaseninkrement. $\theta_k = \pi$ entspricht der Nyquist-Frequenz $\frac{1}{2} f_s$ und somit $\theta_k = 2\pi$ der Abtastfrequenz $f_s = \frac{1}{T_s}$.

Um von der zeitkontinuierlichen in die zeitdiskrete Ebene gelangen zu können, muß stets Bandbegrenzung gegeben sein (Abtasttheorem), deshalb wird die Anzahl der Teiltöne K einen endlichen Wert besitzen.

Bei einem harmonischen Signal sind die Frequenzen der Teiltöne ganzzahlige Vielfache der Grundfrequenz ($\omega_k = k \cdot \omega_1$), das Signal ist dann periodisch mit der Periode $T = \frac{2\pi}{\omega_1}$.

Bei zeitkontinuierlichen periodischen Signalen bietet sich die Entwicklung in Fourier-Reihen an.

1.1.1 Fourier-Reihe

Fourier-Reihe in komplexer Schreibweise:

$$x(t) = \sum_{k=-K}^{K} \underline{A}_k \cdot e^{jk\omega_1 t} \tag{1.3}$$

$$\underline{A}_k = \frac{1}{T} \int_T x(t) \cdot e^{-jk\omega_1 t} \cdot dt \tag{1.4}$$

wobei T die Periodendauer der Grundschwingung mit der Frequenz ω_1 und $\underline{A}_k$ die „komplexe Amplitude“ des k-ten Teiltones ist:

$$\underline{A}_k = \frac{1}{2} A_k \cdot e^{j\varphi_k} \quad \text{für } k > 0 \tag{1.5}$$

$$A_k = 2 \cdot \left|\underline{A}_k\right| \quad \text{und} \quad \varphi_k = \arg(\underline{A}_k) \tag{1.6} \quad (1.7)$$

Der Faktor ½ (bzw. 2) resultiert aus: $\cos(\varphi) = \frac{1}{2}\left(e^{j\varphi} + e^{-j\varphi}\right)$ (1.8)

Allem Anschein nach ist die Entwicklung in eine Fourier-Reihe die ideale Methode zur Parameterbestimmung entsprechend dem stationären deterministischen Modell (1.1) bei zeitkontinuierlichen periodischen Signalen. Allerdings sollte nun davon ausgegangen werden, daß die Signale in zeitdiskreter Form vorliegen.

Bei endlichen (gefensterten) zeitdiskreten Signalen läßt sich eine Diskrete Fourier-Transformation (DFT) berechnen [Oppenheim, Schafer; 1975]:

1.1.2 Diskrete Fourier-Transformation

DFT-Transformationspaar:

$$x(n) = \sum_{m=0}^{N-1} \underline{X}(m) \cdot e^{j(2\pi/N)mn} \qquad \text{für } n = 0, 1, 2, ..., N-1 \qquad (1.9)$$

$$\underline{X}(m) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) \cdot e^{-j(2\pi/N)mn} \qquad \text{für } m = 0, 1, 2, ..., N-1 \qquad (1.10)$$

wobei N die Anzahl der Samples der endlichen Folge x(n) und somit auch die Anzahl der Punkte im diskreten Spektrum $\underline{X}(m)$ ist.

$$\underline{X}(m) = \left|\underline{X}(m)\right| \cdot e^{j \cdot \arg(\underline{X}(m))} \qquad (1.11)$$

$\left|\underline{X}(m)\right|$ ist das Amplitudenspektrum und $\arg(\underline{X}(m))$ das Phasenspektrum, wobei m einen gewissen Frequenzwert bestimmt:

$$f(m) = m \cdot \Delta f = m \cdot \frac{f_s}{N} \qquad (1.12)$$

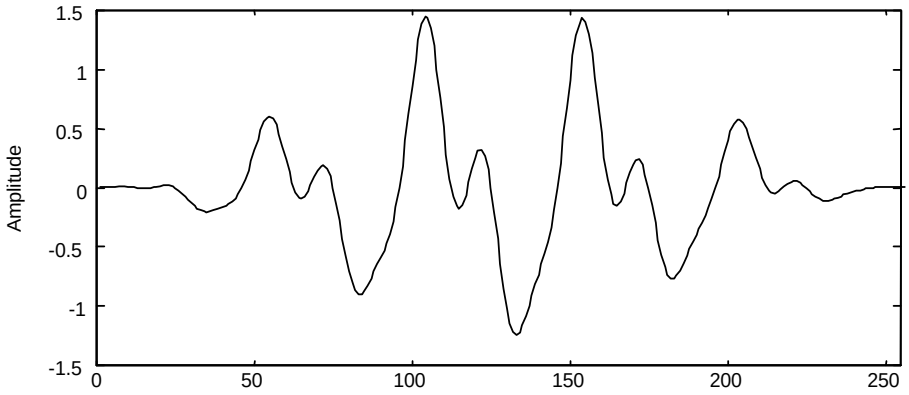
Eine oft verwendete Bezeichnung für m ist Bin-Index oder einfach nur Bin.

Das diskrete Spektrum besteht also aus äquidistanten Linien im Abstand von Δf Hz. Im Vergleich dazu besitzt der Abstand der Spektrallinien bei der Fourier-Reihe den Wert der Grundfrequenz f_1. Bei der DFT ist somit die Länge des Fensters die fundamentale Periodendauer, d.h. die DFT liefert die Fourierkoeffizienten der periodischen Fortsetzung des gefensterten Signals.
Nur dann, wenn die Periode des Signales exakt der Fensterlänge entspricht, erhält man vergleichbare Ergebnisse wie mit Gleichung (1.4), was allerdings nur theoretischer Natur ist.

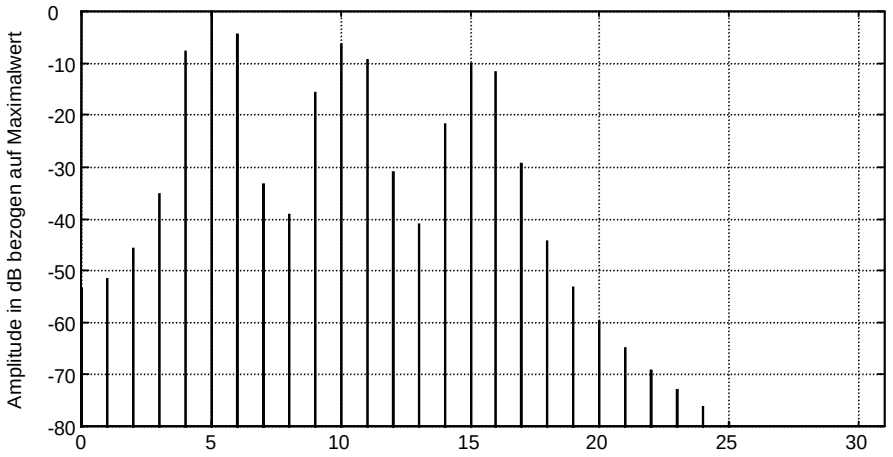
In der Praxis wird versucht, falls das Signal entsprechend stationär ist, das Fenster so groß wie möglich zu gestalten, mindestens soll es sich über drei Signalperioden erstrecken. Je größer es ist, desto besser wird die Frequenzauflösung und desto kleiner der Fehler bei spektralen Schätzungen. Natürlich steigt aber auch der Rechenaufwand an.

Jedoch nicht nur die Länge des Fensters, sondern natürlich auch die Form (Rechteck, Dreieck, von Hann, Hamming, Kaiser, ...) hat einen Einfluß auf das Spektrum, hierzu möchte ich aber auf andere Literatur verweisen (siehe [Oppenheim, Schafer; 1975] und [Haddad, Parsons; 1991]).

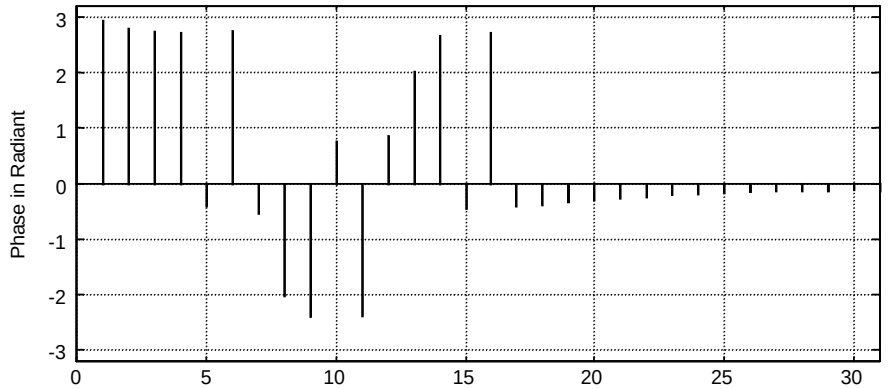
In Abbildung 1.1 ist ein Beispiel zur DFT angefürt. Das (künstliche) Signal enthält drei stationäre Teiltöne, die in einem harmonischen Frequenzverhältnis stehen. In Abb. a) sieht man das mit einem von Hann-Fenster der Länge 256 gefensterte Signal, in b) den Betrag der DFT und in c) die Phase.



a) gefensterte Wellenform (von Hann, N_W = 256 Punkte)
$x(n) = \sin(0.04\pi\, n + \pi/4) + 1/2 \sin(0.08\pi\, n + \pi/2) + 1/3 \sin(0.12\pi\, n)$



b) Die ersten 32 Bins des Amplitudenspektrums der 256-Punkte-DFT



c) Die ersten 32 Bins des Phasenspektrums der DFT

Abb. 1.1: Beispiel zur DFT

1.1.2.1 Zero-Padding

Eine einfache Möglichkeit, die Anzahl der Punkte im Spektrum zu erhöhen, ist das sogenannte „zero-padding“. Hier ist nun die DFT-Punkteanzahl nicht mehr gleich der Fenstergröße.

$N > N_W$

N ... Anzahl der DFT-Punkte
N_W ... Größe des Fensters in Samples

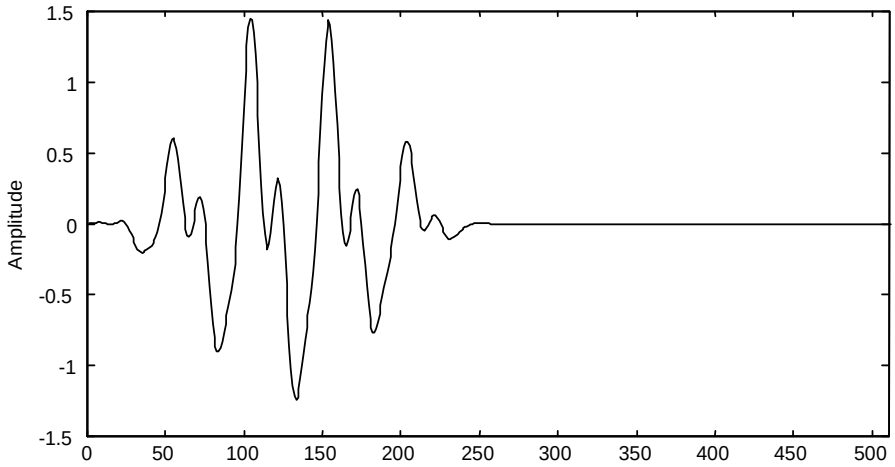
$\frac{N}{N_W}$ ... zero-padding-Faktor

Die restlichen ($N - N_W$) Punkte des DFT-Puffers werden mit Nullen aufgefüllt. Das Ergebnis ist eine Interpolation des Spektrums. Eine Erhöhung der Frequenzauflösung erreicht man dadurch jedoch nicht, da die Länge des Fensters unverändert bleibt. Es liegen nun um den zero-padding-Faktor mehr Bins innerhalb der Keulen des Fensterspektrums, bzw. es vergrößern sich die Breiten der Keulen um diesen Faktor.

Meist bedient man sich der Fast Fourier Transform (FFT) um das Ergebnis der DFT zu bekommen, da auf diese Art beachtlich an Rechenzeit gespart werden kann ($n \cdot ld(n)$ Operationen anstelle von n^2). Die FFT setzt allerdings voraus, daß die Anzahl der Punkte eine Potenz von zwei sein muß ($N = 2^{\mu}$ mit $\mu \in \mathbb{N}$). Man wählt also ein geeignetes N_W und anschließend für N eine Potenz von zwei, die größer oder gleich N_W ist – zero-padding ist also im falle $N > N_W$ unumgänglich.

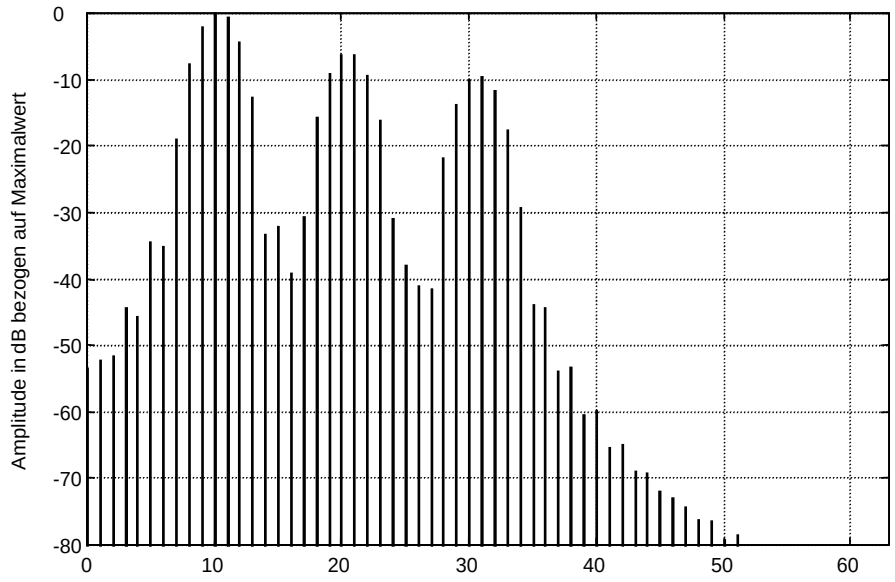
In Abbildung 1.2 sieht man das gleiche Signal wie in Abb. 1.1, mit dem Unterschied, daß hier bei der DFT zero-padding angewendet wurde ($\frac{N}{N_W} = 2$).

Diese Interpolation durch zero-padding bringt aber auch einen Nachteil mit sich, denn bei der in 1.1.2.2 beschriebenen Peak Detection, bei der es darum geht, lokale Maxima im Amplitudenspektrum zu finden, können neben den Hauptkeulen nun auch fälschlicherweise Nebenkeulen detektiert werden, da diese jetzt auch lokale Maxima besitzen können.



a) gefensterte Wellenform mit zero-padding (von Hann mit N_W =256, $\frac{N}{N_W} = 2$)

$x(n) = \sin(0.04\pi\, n + \pi/4) + 1/2 \sin(0.08\pi\, n + \pi/2) + 1/3 \sin(0.12\pi\, n)$



b) Die ersten 64 Bins des Amplitudenspektrum der 512-Punkte-DFT

Abb. 1.2: Beispiel zur DFT mit zero-padding

Doch wie gelangt man nun, ausgehend vom diskreten Spektrum der DFT $\underline{X}(m)$, zu den Parametern des Modells (A_k, ω_k und φ_k) ?

Durch den Einfluß des Fensters wird der k-te Teilton im Allgemeinen nicht als einzelne Linie bei ω_k repräsentiert (ω_k liegt höchstwahrscheinlich zwischen zwei Bins), sondern keulenförmig über mehrere Bins verstreut. Nur im Spezialfall, wenn ω_k ein ganzzahliges Vielfaches von $f_s \cdot 2\pi / N$ ist und ein Rechteckfenster verwendet wurde, wird eine Sinuskomponente mit nur einer Linie dargestellt.

Der Multiplikation von Signal und Fensterfunktion im Zeitbereich entspricht eine Faltung im Frequenzbereich, d.h. das Spektrum des Fensters wird überall dort hin „kopiert“, wo ein Teilton liegt.

Das Betragsspektrum einer Fensterfunktion hat immer eine |sinc(f)|-ähnliche Form, also eine Hauptkeule und wesentlich niedrigere Nebenkeulen. Die wichtigsten Eigenschaften der einzelnen Fenstertypen sind die Breite der Hauptkeule und die Höhe der Nebenkeulen. Ein ideales Fenster hätte eine unendlich schmale Hauptkeule und gar keine Nebenkeulen, also ein Spektrum der Form eines „Dirac-Impulses“ $\delta(f)$. Denn nur dann verursacht die Faltung keine Veränderung. Diese ideale Form würde man jedoch nur bei einer Fensterlänge von unendlich erreichen.
Durch die Breite der Hauptkeule ist es unmöglich (bzw. nur sehr schwer möglich, siehe [Sacchi, Ulrych, Walker; 1998] und [Haddad, Parsons; 1991]), nahe beieinander liegende Komponenten zu trennen.

Abb. 1.3: Betragsspektrum des von Hann-Fensters

Wenn man davon ausgeht, daß das zu Beginn dieses Kapitels vorgestellte Modell dem Signal entspricht, dann darf man annehmen, daß die Teiltöne in der unmittelbaren Nähe jener Bins liegen, wo das Amplitudenspektrum lokale Maxima aufweist.

1.1.2.2 Peak Detection

Um nun die Parameter bestimmen zu können, müssen zuerst die Bins mit lokalem Maximalwert gefunden werden. m_M ist ein Bin-Index mit lokalem Maximun, wenn gilt:

$$\left|\underline{X}(m_M - 1)\right| < \left|\underline{X}(m_M)\right| > \left|\underline{X}(m_M + 1)\right| \qquad (1.13a)$$

Wie schon erwähnt wurde, muß man bei der Verwendung von zero-padding vorsichtig sein, denn auch Nebenkeulen können dann lokale Maxima haben, da die Breite einer Nebenkeule nicht mehr ein Bin ist, sondern $\frac{N}{N_W}$. Oft ist es von Vorteil, das lokale Maximum in einer größeren Umgebung zu definieren (wie auch in [Althoff, Keiler, Zölzer; 1999]):

$$\left|\underline{X}(m_M - 2)\right| < \left|\underline{X}(m_M - 1)\right| < \left|\underline{X}(m_M)\right| > \left|\underline{X}(m_M + 1)\right| > \left|\underline{X}(m_M + 2)\right| \qquad (1.13b)$$

Bei einem realen Signal, welches auch einen stochastischen Anteil besitzt, sind natürlich nicht alle lokalen Maxima Kandidaten für Teiltöne. Es sollten daher einige einschränkende Regeln vereinbart werden:

1) $$\left|\underline{X}(m_M)\right| > A_{MIN} \qquad (1.14)$$

Die Amplitude muß einen absoluten Schwellenwert A_{MIN} überschreiten.

2) $$\frac{\left|\underline{X}(m_M)\right|}{\max\left(\left|\underline{X}(m)\right|\right)} > a_{MIN} \qquad (1.15)$$

Die Amplitude muß einen relativen Schwellenwert a_{MIN} überschreiten.

3) $$h(m_M) = \frac{2 \cdot \left|\underline{X}(m_M)\right|}{\left|\underline{X}(m_{m-})\right| + \left|\underline{X}(m_{m+})\right|} > h_{MIN} \qquad (1.16)$$

Die Größe $h(m_M)$ gibt Auskunft darüber, wie stark ein lokales Maximum ausgeprägt ist, d.h. wie hoch es im Vergleich zum Mittelwert der benachbarten lokalen Minima ist (siehe [Serra; 1989], $|\underline{X}(m_{m-})|$ linksseitig benachbartes lokales Minimum und $|\underline{X}(m_{m+})|$ rechtsseitiges). Auch diese Relativgröße muß einen Schwellenwert h_{MIN} überschreiten.

4) $$\max\left(\arg\left(\underline{X}(m)\right)\right) - \min\left(\arg\left(\underline{X}(m)\right)\right) < p_{MAX} \;,\quad m = m_M - B \;...\; m_M + B \qquad (1.17a)$$

$$B = \frac{B_W}{2} - 1 \qquad (1.18)$$

Phase Condition: alle Bins, die innerhalb der entsprechenden Hauptkeule liegen, sollen einen annähernd gleichen Phasenwert liefern. B_W ist die Breite der Hauptkeule.

Die Voraussetzung für diese Phase Condition ist die Verwendung eines zero-phase windows (siehe 1.2.3.1). Die Überprüfung auf annähernd gleiche Phasenwerte kann auf verschieden Arten erfolgen. Z.B. kann auch die Abweichung zum Mittelwert herangezogen werden:

$$\left|\text{mean}\left(\arg\left(X(m)\right)\right) - \arg\left(X(m)\right)\right| < p_{MAX}, \quad m = m_M - B \ldots m_M + B \qquad (1.17b)$$

5) $0 < m_{MIN} < m_M < m_{MAX} < N/2$

In manchen Anwendungsfällen kann es auch nützlich sein, den Frequenzbereich einzuschränken, z.B. auf den Hörbereich oder einen anderen relevanten Bereich. Es ergibt sich eine Rechenzeitreduktion. Oft ist aber ein Downsampling vor der DFT zu empfehlen.

6) Die Anzahl der Peaks darf einen gewissen Maximalwert nicht überschreiten.

Dies kann dann von Bedeutung sein, wenn bei der Resynthese möglicherweise nur eine bestimmte Anzahl an Oszillatoren zur Verfügung steht.

7) Psychoakustische Kriterien

Man kann die Anzahl der Parameter-Sätze um einiges verkleinern, wenn Komponenten, die durch benachbarte verdeckt werden, weggelassen werden. Auf diese Weise funktionieren viele Audio-Datenreduktions-Algorithmen.
Es ist auch durchaus vorstellbar, das Amplitudenspektrum vor der Peak Detection entsprechend einer inversen Kurve gleicher Lautstärke zu gewichten, um Komponenten, die vom Menschen mit höherer Empfindlichkeit wahrgenommen werden, bevorzugt zu behandeln.

1.1.2.3 Peak Interpolation

Würde man ω_k direkt aus dem Bin-Index berechnen ($\omega_k = \frac{2\pi}{N} \cdot m_M \cdot f_s$), so wäre der Fehler im schlechtesten Fall sogar $\frac{f_s}{2 \cdot N}$. Mit Hilfe einer parabolischen Interpolation (siehe [Serra; 1989]) läßt sich der Fehler um einiges verringern. Die Berechnung sollte mit logarithmischen Amplitudenwerten (dB) erfolgen, um eine bessere Approximation zu erreichen.

$$\begin{aligned} \alpha &= 20 \log |\underline{X}(m_M - 1)| \\ \beta &= 20 \log |\underline{X}(m_M)| \\ \chi &= 20 \log |\underline{X}(m_M + 1)| \end{aligned} \qquad (1.19)$$

Wir vereinbaren zunächst ein rechtwinkeliges Koordinatensystem, das den Ursprung im Punkt (m_M , 0) und lineare Achsen hat. Die Einheit auf der Abszisse entspricht einem Bin, die auf der Ordinate einem dB. Als nächstes versucht man, folgende Parabel in die drei Punkte (-1 , α), (0 , β) und (1 , χ) zu legen.

$$y(x) = a \cdot (x - p)^2 + b \tag{1.20}$$

Gleichung (1.20) besitzt drei Unbekannte. Mit den drei gegebenen Punkten erhält man ebenfalls drei Gleichungen, und a, b und p lassen sich berechnen. p ist die horizontale Verschiebung des Maximums, die letztendlich von Interesse ist:

$$p = \frac{\alpha - \chi}{2 \cdot (\alpha - 2\beta + \chi)} \tag{1.21}$$

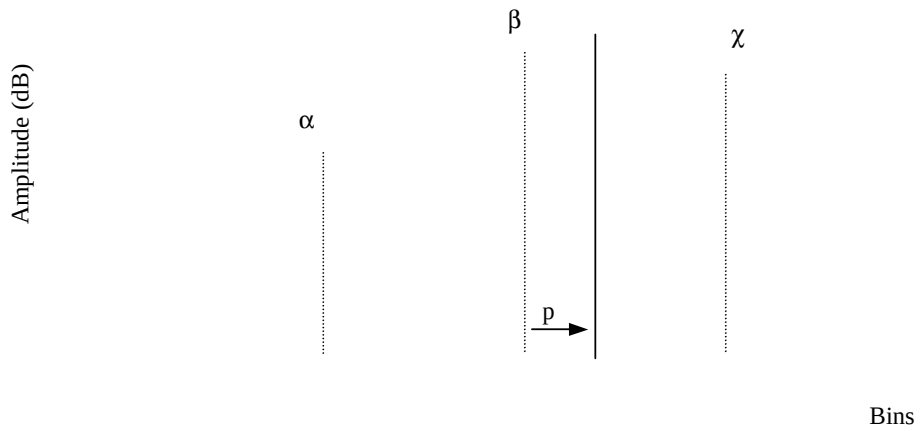


Abb. 1.4: Parabolische Interpolation

Nun läßt sich auch ω_k mit geringem Fehler berechnen:

$$\omega_k = \frac{2\pi}{N} \cdot (m_M + p) \cdot f_s \tag{1.22}$$

bzw. θ_k:

$$\theta_k = \frac{2\pi}{N} \cdot (m_M + p) \tag{1.23}$$

Für A_k und φ_k ist die exakteste Methode, die komplexe Amplitude wie bei der DFT, aber eben nur für den einen Wert (m_M+p) zu bestimmen:

$$\underline{X}(m_M + p) = \frac{1}{N}\sum_{n=0}^{N-1} x(n) \cdot e^{-j\cdot(2\pi/N)\cdot(m_M+p)\cdot n} \tag{1.24}$$

$$A_k = 2 \cdot W_A \cdot \left|\underline{X}(m_M + p)\right| \quad \text{und} \quad \varphi_k = \arg(\underline{X}(m_M + p)) \tag{1.25} \tag{1.26}$$

W_A korrigiert den Einfluß des Fensters auf die Amplitude.

$$W_A = \frac{N}{\sum_{n=0}^{N-1} w(n)} \tag{1.27}$$

w(n) ist die Fensterfunktion. Z.B. ist bei einem Rechteckfenster $W_A = 1$, bei einem von Hann-Fenster (Hanning) ist $W_A = 2$.

In Gleichung (1.24) wird ein Punkt der Fouriertransformierten[♣] des zeitdiskreten endlichen (gefensterten) Signals berechnet. In der Literatur findet man diese Transformierte oft mit DTFT abgekürzt, was für Discrete Time Fourier Transform steht [Haddad, Parsons; 1991]. Oft wird die Fouriertransformierte von x(n) auch mit $X\left(e^{j\theta}\right)$ bezeichnet [Oppenheim, Schafer; 1975].

Es gibt mehrere Möglichkeiten für die Durchführung einer Peak Interpolation. Zum Beispiel kann die Transformierte der Fensterfunktion verwendet werden. In [Althoff, Keiler, Zölzer; 1999] kommt ein Dreieckfenster und dessen Spektrum zum Einsatz.

[♣] Die Fouriertransformierte eines endlichen Signals ist natürlich eine kontinuierliche Funktion. Da x(n) zeitdiskret ist, ist die Transformierte auch periodisch.

1.2 Zeitvariante deterministische Signale

Im Gegensatz zu stationären Signalen sind hier die Parameter mit der Zeit nicht konstant.

Zeitvariantes deterministisches Modell:

$$x_{det}(n) = \sum_{k=1}^{K} A_k(n) \cdot \cos(\varphi_k(n)) \qquad (1.28)$$

$A_k(n)$ ist die sich mit der Zeit ändernde Amplitude (Momentanamplitude) und $\varphi_k(n)$ die Momentanphase des k-ten Teiltones, wobei:

$$\varphi_k(n) = \sum_{m=0}^{n-1} \theta_k(m) + \varphi_k(0) \qquad (1.29)$$

$\theta_k(n)$ stellt die auf 2π normierte Momentan-Kreisfrequenz und $\varphi_k(0)$ die Nullphase des k-ten Teiltones dar.

Es sollte hier erwähnt werden, daß die zeitvarianten Parameter sich nur langsam mit der Zeit verändern dürfen, so daß ein kurzer Signalausschnitt näherungsweise als stationär betrachtet werden kann.

Berechnet man eine DFT des gesamten, aber trotzdem endlichen (gefensterten) Signals, so erhält man zwar ein dichtes Spektrum, aber dieses enthält keine zeitlichen Informationen, d.h. es kann keine Aussage darüber gemacht werden, wann eine bestimmte Frequenzkomponente aufgetreten ist.
Man benötigt also eine Zeit-Frequenz-Repräsentation.
Die am meisten verwendete Zeit-Frequenz-Repräsentation ist die Kurzzeit-Fouriertransformation (Short-Time Fourier Transform, STFT).

1.2.1 Kurzzeit-Fourier-Transformation (STFT)

Um zeitliche Informationen zu erhalten, werden mehrere kürzere Fenster, die sich teilweise überlappen, zur Berechnung von sogenannten Kurzzeitspektren verwendet.

$$\underline{X}_i(m) = \frac{1}{N} \sum_{n=0}^{N-1} w(n) \cdot x(n + i \cdot H) \cdot e^{-j \cdot (2\pi / N) \cdot m \cdot n} \qquad (1.30)$$

für m = 0, 1, 2, ..., N-1

und i = 0, 1, 2, ..., I-1

$\underline{X}_i(m)$ ist das Kurzzeitspektrum des Rahmens mit dem Index i. H gibt an, um wieviele Samples Frame $i+1$ gegenüber i versetzt ist (Hopsize), also stellt i einen zeitlichen Wert dar. N ist die Länge eines Frames und somit die Länge des Fensters w(n) mit eventuell angehängten Nullen im Falle von zero-padding und natürlich auch die Anzahl der Punkte eines Kurzzeitspektrums.

Das Ergebnis einer STFT ist also eine Menge von DFT-Spektren. Mittels einer Inversen Kurzzeit-Fourier-Transformation (siehe 1.2.4) läßt sich das ursprüngliche Signal wieder rekonstruieren. Als einfach durchführbare Modifikation kommt Filterung (Blockfiltertechnik) in Frage. Natürlich ist diese Repräsentation nicht sehr kompakt, wodurch Modifikationen wie Time-Stretching oder Pitch-Shifting nicht einfach zu realisieren sind.

Wir wollen nun aus den Daten der STFT die Parameter für das Signalmodell in Gleichung (1.28) berechnen. Wendet man die in 1.1.2.2 und 1.1.2.3 beschriebene Peak-Auffindung und Interpolation an, so läßt sich für jeden Rahmen eine Menge an Parametersätzen $\{A_p^i, \theta_p^i, \varphi_p^i\}$ berechnen. Die Indizes sagen aus, daß es sich zum Zeitpunkt $n = i \cdot H$ um das p-te relevante lokale Maximum im Amplitudenspektrum handelt. Als P^i bezeichnen wir die Anzahl der Peaks im Frame i.

Als Repräsentation ist diese Wertetriple-Menge jedoch noch immer nicht sehr gut geeignet, da nicht feststeht, ob alle Werte der p-ten Sinuskomponente wirklich zu einem einzigen Teilton gehören. Die Repräsentation soll also auch eine Zusammengehörigkeit der Werte zu einem bestimmten Teilton darstellen.

1.2.1.1 Peak Continuation

Wir wollen hier nun dem Frequenzverlauf eines Teiltones über der Zeit folgen, d.h. Linien finden, die die Punkte θ_p^i und θ_q^{i+1} miteinander verbinden. Bezeichnen wir diese Linien als Frequenz-Trajektorien.

In Abbildung 1.4 wird dies illustriert. Es gibt also Teiltöne, die zu bestimmten Zeitpunkten entstehen und andere, die verschwinden. Es soll für eine beliebiges Signal auch möglich sein, daß sich welche kreuzen.

Es ist nicht möglich, einen für beliebige Signale gültigen Algorithmus zu finden, es können lediglich Regeln aufgestellt werden, die dann bei bestimmten Typen von Signalen mehr oder weniger Wichtigkeit haben. Zum Beispiel können bei Signalen mit harmonischen Frequenzverhältnissen natürlich keine Kreuzungen auftreten.

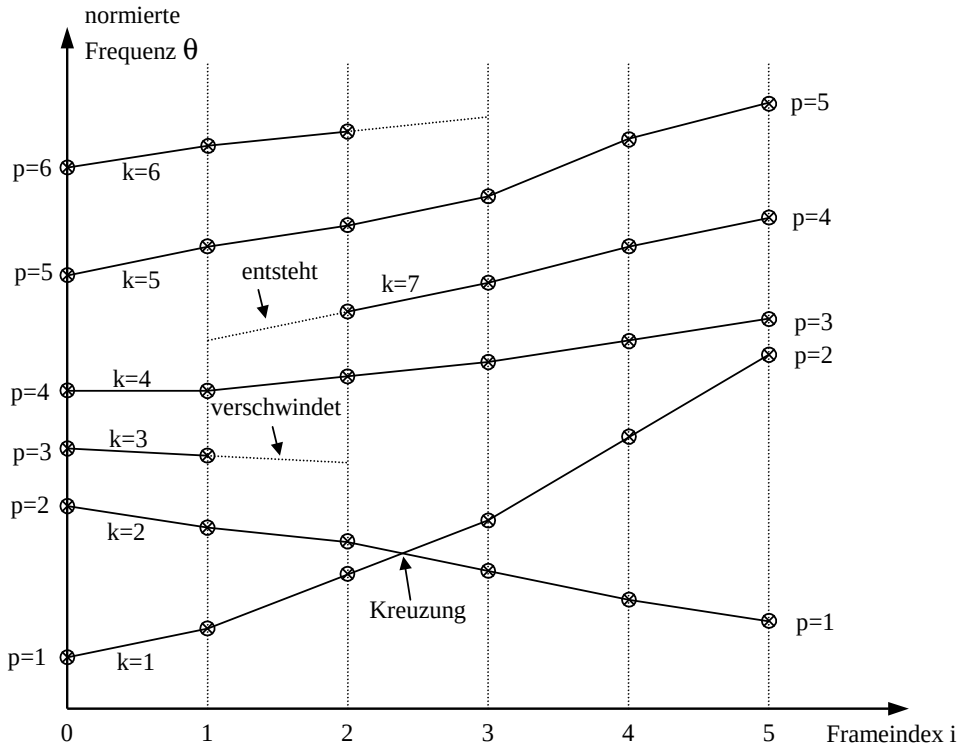


Abb. 1.5: Beispiel eines Teilton-Verlaufs

Initialisierung: Die Werte der detektierten Maxima bei Frame 0 werden direkt den Teiltönen zugewiesen (p = k):

$$\hat{\theta}_k{}^0 = \theta_k{}^0, \quad \hat{A}_k{}^0 = A_k{}^0, \quad \hat{\varphi}_k{}^0 = \varphi_k{}^0 \tag{1.31}$$

Einige Regeln zur Trajektorien-Fortsetzung:

1) $\hat{\theta}_k{}^{i+1} = \theta_p{}^{i+1}$ so daß gilt: $\left|\hat{\theta}_k{}^{i+1} - \hat{\theta}_k{}^{i}\right| = \min$ (1.32)

Die Änderung der Frequenz soll minimal sein.

2) $\hat{\theta}_k{}^{i+1} = \theta_p{}^{i+1}$ so daß gilt: $\left|\hat{\theta}_k{}^{i+1} - 2\cdot\hat{\theta}_k{}^{i} + \hat{\theta}_k{}^{i-1}\right| = \min$ (1.33)

Die Änderung der Steigung der Frequenztrajektorie soll minimal sein.

3) $\hat{\theta}_k{}^{i+1} = \theta_p{}^{i+1}$ so daß gilt: $\left|\hat{A}_k{}^{i+1} - \hat{A}_k{}^{i}\right| = \min$ (1.34)

Die Änderung der dazugehörigen Amplitude soll minimal sein.

4) $\hat{\theta}_k^{\,i+1} = \theta_p^{\,i+1}$ so daß gilt: $\left|\hat{\varphi}_k^{\,i+1} - \hat{\varphi}_k^{\,i} + M \cdot 2\pi - \frac{1}{2}\left(\hat{\theta}_k^{\,i+1} + \hat{\theta}_k^{\,i}\right) \cdot H\right| = \min$ (1.35)

Die Phasendifferenz soll zur mittleren Frequenz proportional sein. M ist eine Ganze Zahl und sorgt für das „Phase Unwrapping“:

$$M = \text{round}\left(\frac{1}{2\pi}\left[\hat{\varphi}_k^{\,i} - \hat{\varphi}_k^{\,i+1} + \frac{1}{2}\left(\hat{\theta}_k^{\,i+1} + \hat{\theta}_k^{\,i}\right) \cdot H\right]\right) \quad (1.36)$$

Die Regeln 1) und 2) können zu sehr unterschiedlichen Ergebnissen führen. 1) sollte dominierend sein bei Signalen, deren Parameter sich sehr langsam mit der Zeit ändern und wenn keine Kreuzungen vorkommen können. 2) sollte dann einen höheren Stellenwert haben, wenn sich bei einem Signal die Frequenz schneller ändern kann und wenn Kreuzungen richtig erkannt werden sollten. 3) und vor allem 4) helfen sehr stark mit, dem k-ten Teilton richtig zu folgen.

Es ist nun mittels iterativer Algorithmen eine optimale Trajektorien-Fortschreitung im Sinne der verschiedenen Fortschreitungs-Regeln zu finden.

Wenn bei Frame i + 1 mehr Maxima detektiert wurden, als bei Frame i ($P^{i+1} > P^i$), dann entstehen hier neue Trajektorien, bzw. wenn bei i + 1 weniger gefunden wurden ($P^{i+1} < P^i$), dann verschwinden welche. Es empfiehlt sich, eine minimale "Lebensdauer" zu definieren. Falls eine Trajektorie diese Minimallänge nicht erreicht, werden die dazugehörigen spektralen Spitzen ignoriert und nicht der deterministischen Signalkomponente zugewiesen.

Mehr zu dem Thema Peak Continuation kann in [McAulay, Quatieri; 1986 (1)], [Serra; 1989], [Serra, Smith; 1990] und [Althoff, Keiler, Zölzer; 1999] gefunden werden.

Sollte die Teiltonverfolgung ein akzeptables Ergebnis geliefert haben, so ist die Analyse der deterministischen Komponente abgeschlossen. Mit dem Parametersatz kann nun mittels Additiver Synthese dieser Signalanteil $x'_{det}(n)$ isoliert resynthetisiert werden.

1.2.2 Additive Synthese

Die Additive Synthese ist die direkte Realisierung von Gleichung (1.28). K Sinusoszillatoren werden mit den zeitvarianten Parametern $A_k(n)$ und $\varphi_k(n)$ angesteuert. In Abbildung 1.6 wird dies veranschaulicht.

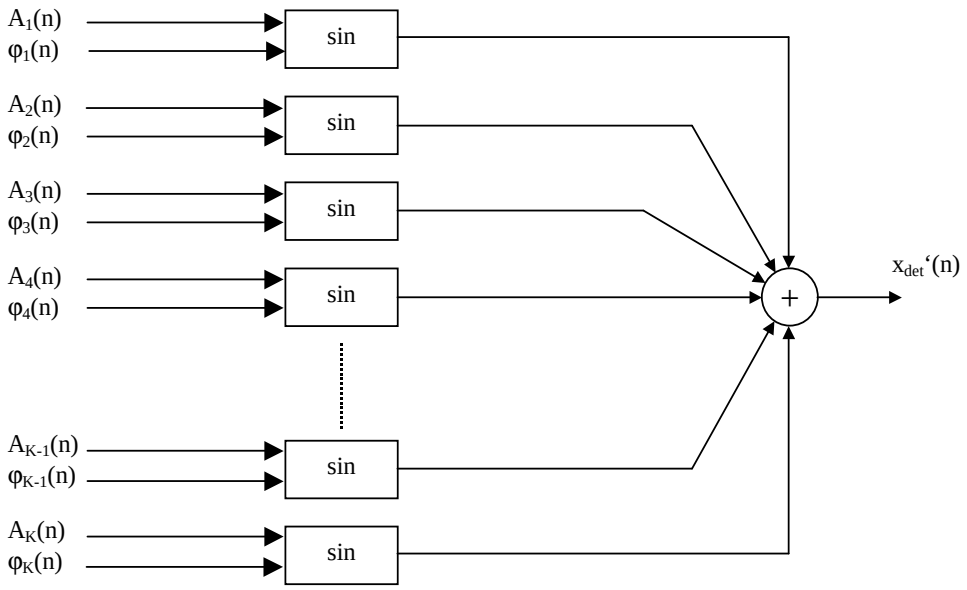


Abb. 1.6: Prinzip der Additiven Synthese

Da nach der zuletzt genannten Parameter-Repräsentation nicht immer alle Teiltöne gleichzeitig vorhanden sein müssen (es können welche entstehen oder verschwinden), sollte eine dynamische Zuweisung zu einem Oszillator erfolgen („Voice Allocator“). Beim Entstehen sollte schon einen Frame früher mit einem linearen Einblenden (fade in) begonnen werden, desgleichen sollte beim Verschwinden im nachfolgenden Frame ein Ausblenden (fade out) geschehen, wie in Abbildung 1.5 durch strichlierte Linien angedeutet wird.
Für diese zusätzlichen Trajektorienpunkte müssen noch Werte definiert werden. Bezeichnen wir den Rahmen, in dem der k-te Teilton zum ersten Mal (bzw. zum letzten Mal) erscheint, mit dem Index i.

Einblenden:

$$\hat{A}_k^{\,i-1} = 0 \tag{1.37}$$

$$\hat{\theta}_k^{\,i-1} = \hat{\theta}_k^{\,i} \tag{1.38}$$

$$\hat{\varphi}_k^{\,i-1} = \mathrm{wrap}\left(\hat{\varphi}_k^{\,i} - \hat{\theta}_k^{\,i} \cdot H\right) \tag{1.39}$$

$$\mathrm{wrap}(\varphi) = \begin{cases} \mathrm{mod}(\varphi, 2\pi), & \mathrm{mod}(\varphi, 2\pi) < \pi \\ \mathrm{mod}(\varphi, 2\pi) - 2\pi, & \mathrm{mod}(\varphi, 2\pi) \geq \pi \end{cases} \tag{1.40}$$

$\mathrm{mod}(x, y)$ ist der Rest der Integer-Division von x/y.

Hier wurde der gleiche Frequenzwert verwendet, man kann aber auch die gleiche Steigung der Frequenz-Trajektorie übernehmen:

$$\hat{\theta}_k^{\,i-1} = 2 \cdot \hat{\theta}_k^{\,i} - \hat{\theta}_k^{\,i+1} \tag{1.41}$$

$$\hat{\varphi}_k^{\,i-1} = \mathrm{wrap}\left(\hat{\varphi}_k^{\,i} - \tfrac{1}{2}\left(3 \cdot \hat{\theta}_k^{\,i} - \hat{\theta}_k^{\,i+1}\right) \cdot H\right) \tag{1.42}$$

Ausblenden mit konstanter Frequenz:

$$\hat{A}_k^{\,i+1} = 0 \tag{1.43}$$

$$\hat{\theta}_k^{\,i+1} = \hat{\theta}_k^{\,i} \tag{1.44}$$

$$\hat{\varphi}_k^{\,i+1} = \mathrm{wrap}\left(\hat{\varphi}_k^{\,i} + \hat{\theta}_k^{\,i} \cdot H\right) \tag{1.45}$$

bzw. mit konstanter Steigung:

$$\hat{\theta}_k^{\,i+1} = 2 \cdot \hat{\theta}_k^{\,i} - \hat{\theta}_k^{\,i-1} \tag{1.46}$$

$$\hat{\varphi}_k^{\,i+1} = \mathrm{wrap}\left(\hat{\varphi}_k^{\,i} + \tfrac{1}{2}\left(3 \cdot \hat{\theta}_k^{\,i} - \hat{\theta}_k^{\,i-1}\right) \cdot H\right) \tag{1.47}$$

Weiters liefert die Repräsentation nur Werte für $n = i \cdot H$. Läßt man die Parameter innerhalb der H Samples konstant, so sind die Wertesprünge zwischen den einzelnen Frames höchstwahrscheinlich hörbar. Es muß also noch eine Interpolation erfolgen.

Die Berechnung der Amplitude $A_k(n)$ stellt kein großes Problem dar, denn eine lineare Interpolation ist völlig ausreichend:

$$A_k(n) = \hat{A}_k^{\frac{n}{\overline{\overline{H}}}} + \frac{\hat{A}_k^{\frac{n}{\overline{\overline{H}}}+1} - \hat{A}_k^{\frac{n}{\overline{\overline{H}}}}}{H} \cdot \left(n - \left(\frac{n}{\overline{\overline{H}}} \cdot H\right)\right) \tag{1.48}$$

$\frac{n}{\overline{\overline{H}}}$ bedeutet, daß es sich um eine Integer-Division handelt.

Ein größeres Problem ist die Berechnung der Momentanphase $\varphi_k(n)$, da $\varphi_k(n)$ und Momemtanfrequenz $\theta_k(n)$ voneinander abhängig sind :

$$\theta_k(t) = \frac{1}{f_s} \frac{d\varphi_k(t)}{dt} \tag{1.49}$$

Aus diesem Zusammenhang ergeben sich auch die Gleichungen (1.29) und (1.35).

1.2.2.1 Keine Auswertung der Phaseninformation

Die einfachste Möglichkeit, die Momentanphase $\varphi_k(n)$ zu rekonstruieren ist, wenn man die Phaseninformationen $\varphi_k(i \cdot H)$ mit Ausnahme der Nullphase $\varphi_k(0)$ gar nicht verwendet. Es wird nur die Momentanfrequenz $\theta_k(i.H)$ interpoliert, wobei wie auch bei der Amplitude eine lineare Interpolation ausreichend ist.

$$\theta_k(n) = \hat{\theta}_k^{\frac{n}{H}} + \frac{\hat{\theta}_k^{\frac{n}{H}+1} - \hat{\theta}_k^{\frac{n}{H}}}{H} \cdot \left(n - \left(\frac{n}{H} \cdot H \right) \right) \tag{1.50}$$

Die Rekonstruktion von $\varphi_k(n)$ erfolgt nach Gleichung (1.29).

1.2.2.2 Kubische Interpolation der Momentanphase

Wir wollen nun versuchen, auch die Phaseninformation zu verwerten. Die folgende Interpolation mit einem kubischen Polynom wurde von McAulay und Quatieri vorgestellt (siehe [McAulay, Quatieri; 1986 (1)] und [McAulay, Quatieri; 1986 (2)]).

$$\varphi_k(i \cdot H + u) = a + b \cdot u + c \cdot u^2 + d \cdot u^3 \quad \text{für} \quad 0 \le u < i \cdot H \tag{1.51}$$

mit:

$$a = \hat{\varphi}_k^{\,i} \tag{1.52}$$

$$b = \hat{\theta}_k^{\,i} \tag{1.53}$$

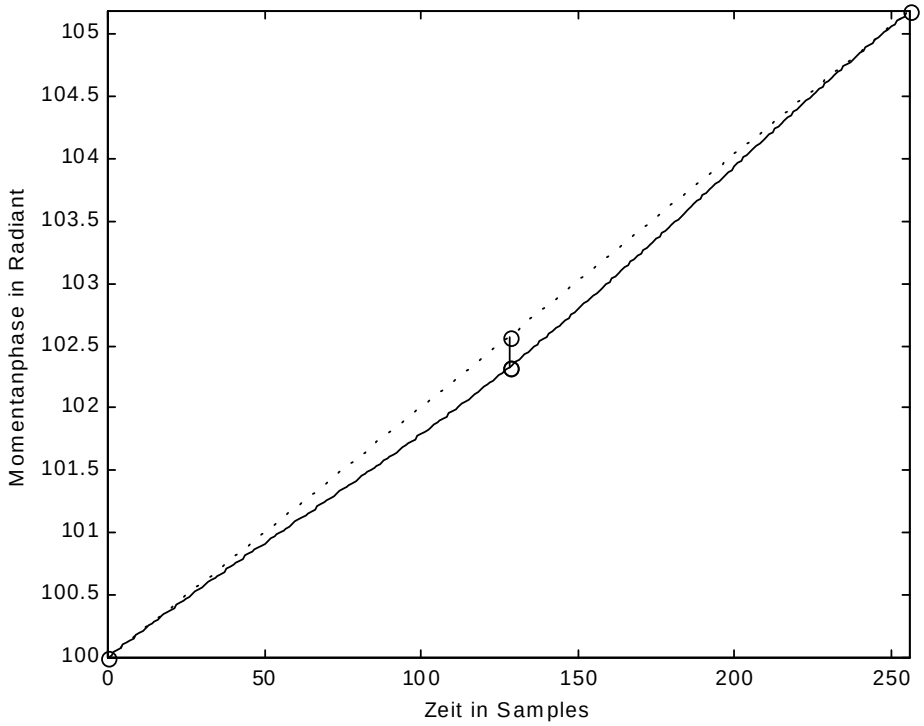
$$c = \frac{3}{H^2}(\hat{\varphi}_k^{\,i+1} - \hat{\varphi}_k^{\,i} - \hat{\theta}_k^{\,i} \cdot H + 2\pi \cdot M) - \frac{1}{H}(\hat{\theta}_k^{\,i+1} - \hat{\theta}_k^{\,i}) \tag{1.54}$$

$$d = -\frac{2}{H^3}(\hat{\varphi}_k^{\,i+1} - \hat{\varphi}_k^{\,i} - \hat{\theta}_k^{\,i} \cdot H + 2\pi \cdot M) + \frac{1}{H^2}(\hat{\theta}_k^{\,i+1} - \hat{\theta}_k^{\,i}) \tag{1.55}$$

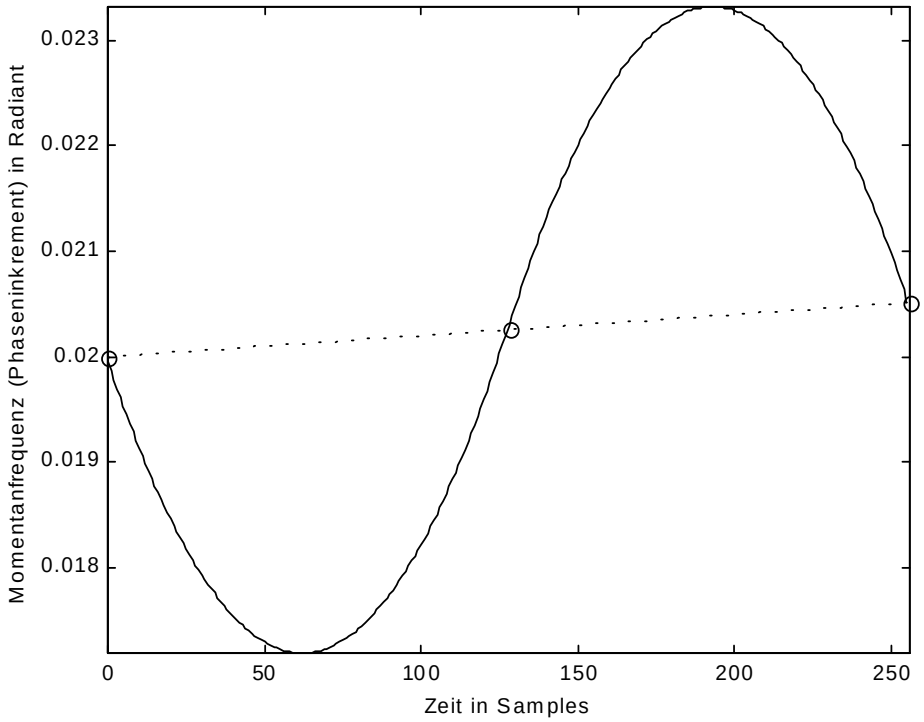
M berechnet sich nach Gleichung (1.35).

Bei Verwendung dieser Interpolation wird garantiert, daß sowohl der Wert der Momentanphase als auch die zeitliche Ableitung an den Rahmengrenzen stimmen, und weiters ergibt sich ein maximal flacher Verlauf, falls M richtig bestimmt worden ist.

In Abbildung 1.6 sieht man ein Beispiel, wo, ausgehend von einem Sweep-Signal, zum Phasenwert des mittleren Frames ein „Meßfehler“ von –0.25 rad hinzuaddiert worden ist. Das Ergebnis ist eine oszillierende Momentanfrequenz.



a) Verlauf der Momentanphase



b) Resultierender Verlauf der Momentanfrequenz

Abb. 1.7: Beispiel zur Kubischen Interpolation der Phase

1.2.3 Inverse Kurzzeit-Fourier-Transformation (ISTFT)

Aus der Kurzzeitspektren-Menge einer STFT läßt sich mit Hilfe der Inversen Kurzzeit-Fourier-Transformation wiederum das ursprüngliche Signal x(n) berechnen, unter der Voraussetzung, daß das Analysefenster w(n) die Bedingung in Gleichung (1.58) erfüllt.

$$x(n) = \frac{1}{W_{oa}} \cdot \sum_{i=0}^{I-1} \mathrm{Shift}^{g}_{i \cdot H, n, N} \left[\sum_{m=0}^{N-1} \underline{X}_i(m) \cdot e^{j \cdot (2\pi / N) \cdot m \cdot g} \right] \tag{1.56}$$

$$\mathrm{Shift}^{g}_{i \cdot H, n, N}[f(g)] = \begin{cases} 0, & n < i \cdot H \\ f(n - i \cdot H), & i \cdot H \le n < i \cdot H + N \\ 0, & n \ge i \cdot H + N \end{cases} \tag{1.57}$$

$$W_{oa}(n) = \sum_{m=-\infty}^{\infty} w(n - m \cdot H) = \mathrm{konst.} = W_{oa} \tag{1.58}$$

Wie aus (1.56) und (1.57) zu sehen ist, werden die überlappenden Signalausschnitte, die durch Rücktransformation eines Kurzzeitspektrums entstanden sind, aufsummiert (Overlap and Add). In Abbildung 1.8 wird dies veranschaulicht.

$W_{oa}(n)$ ist der Amplitudenverlauf der überlappenden und aufsummierten Fensterfunktion. Ist dieser nicht konstant, so wird das Ergebnis der ISTFT genau mit dieser Funktion, die eine Periode von H besitzt, amplitudenmoduliert.
Die Wahl der Sprungweite H muß daher zum verwendeten Fenster passen. Nehmen wir zum Beispiel ein Hanning-Fenster. Damit $W_{oa}(n)$ konstant wird, muß H so gewählt werden, daß $N_w / H = \mu \in \{2, 3, 4, ..., N_w\}$. Wählen wir z.B. $N_W = 256$, dann kann μ aus der Menge {2, 4, 8, 16, 32, 64, 128, 256} gewählt werden, bzw. $H \in \{128, 64, 32, 16, 8, 4, 2, 1\}$. Wählen wir hingegen $N_W = 255$, dann muß $\mu \in \{3, 5, 15, 17, 51, 85, 255\}$ sein, bzw. $H \in \{85, 51, 17, 15, 5, 3, 1\}$.
Beim Rechteckfenster ist es ähnlich, mit der Ausnahme, daß H auch theoretisch gleich N_W sein kann (μ=1).
Bei den meisten anderen Fenstertypen ist die einzige Möglichkeit, damit die Bedingung $W_A(n)$ = konst. eingehalten wird, H = 1. Jedoch kann praktisch auch ein anderer Wert für H gewählt werden, sofern die Welligkeit von $W_{oa}(n)$ gering bleibt, denn eine minimale Amplitudenmodulation wird nicht hörbar sein.

Siehe zu diesem Thema auch Kapitel 3.2, wo neben einem Analysefenster auch ein Synthesefenster verwendet wird.

1.2.3.1 Overlap and Add – Resynthese

Eine Möglichkeit, Additive Synthese ohne einer großen Anzahl an Oszillatoren zu realisieren, ist die sogenannte Overlap and Add – Methode (oder IFFT-Synthesis). Hier werden die Kurzzeitspektren entsprechend dem gewünschten Signal errechnet und anschließend mittels inverser Kurzzeit-Fourier-Transformation in den Zeitbereich gebracht (siehe [Rodet, Depalle; 1992] und [Althoff, Keiler, Zölzer; 1999]). Die Berechnung eines Spektrums erfolgt durch Aufsummieren von verschobenen Fensterspektren W(m).

$$\underline{X}_i(m) = \sum_{k=1}^{K} \hat{A}_k^{\,i} \cdot W\left(m - \frac{N}{2\pi} \cdot \hat{\theta}_k^{\,i}\right) \cdot e^{j \cdot \tilde{\varphi}_k^{\,i}} \tag{1.59}$$

Im Vergleich zur in 1.2.2 beschriebenen Oszillator-Methode ist hier eine Interpolation der Parameter zwischen zwei Rahmen nur mit viel Rechenaufwand möglich, weshalb ich auf diese verzichten möchte. Somit kann auch die Phaseninformation mit Ausnahme bei Frame 0 nicht verwertet werden.

Es ist besonders darauf zu achten, daß die Momentanphase von Frame zu Frame ohne Sprünge verläuft. Diese Phasenkorrektur berücksichtigt $\tilde{\varphi}_k^{\,i}$. Bei i = 0 werden die Analysewerte zugewiesen:

$$\tilde{\varphi}_k^{\,0} = \hat{\varphi}_k^{\,0} \tag{1.60}$$

$$\tilde{\varphi}_k^{\,i+1} = \mathrm{wrap}\left(\tilde{\varphi}_k^{\,i} + \hat{\theta}_k^{\,i} \cdot H\right) \tag{1.61}$$

Wird zum Beispiel ein Hanning-Fenster verwendet, so sieht W(m) folgendermaßen aus:

$$W(m) = W_R(m) + \frac{1}{2}\left(W_R(m-1) + W_R(m+1)\right) \tag{1.62}$$

$W_R(m)$ stellt das Spektrum des Rechteckfensters dar (Dirichlet-Funktion♦):

$$W_R(m) = \frac{\sin(m\pi)}{N_W \cdot \sin\left(\frac{m\pi}{N_W}\right)} \tag{1.63}$$

Es handelt sich hier in (1.62) und (1.63) um reellwertige Spektren, sogenannte „zero-phase windows“, was bedeutet, daß die Fenster im Zeitbereich gerade Symmetrie besitzen. Gerade

♦ Die Dirichlet-Funktion spielt in der zeitdiskreten Signaltheorie die gleiche Rolle wie die sinc-Funktion in der zeitkontinuierlichen.

Symmetrie heißt, daß der zeitliche Nullpunkt in der Fenstermitte liegt, dies ist allerdings nur bei einem ungeraden N_w möglich.

Durch die Reellwertigkeit vereinfacht sich die Berechnung. Die Rechenzeit läßt sich weiters verringern, wenn man die Spektren nur für das analytische Signal berechnet, d.h. es gibt keine Komponenten bei negativen Frequenzen. Man nimmt dann vom Ergebnis der inversen DFT einfach nur der Realteil und multipliziert diesen mit 2.

Es ist auch bei der Analyse sinnvoll, ein zero-phase window zu verwenden, denn dann gelten die aus einem Kurzzeitspektrum ermittelten Parameter (im Speziellen die Phase) nicht wie bisher für den Zeitpunkt wo das Fenster beginnt, sondern für die Fenstermitte. Die Analyse des Rahmens mit dem Index i liefert dann also Werte für den Zeitpunkt $n = i \cdot H + \frac{1}{2} \cdot (N_W - 1)$.
Bei genauerer Untersuchung von Kurzzeitspektren stellt man fest, daß die ermittelten Frequenz- und Amplitudenwerte immer für den Zeitpunkt der Fenstermitte gelten. Man kann dies zeigen, indem man ein Sweep-Signal mit linearem Frequenzverlauf oder ein Signal, bei dem sich die Amplitude linear verändert, transformiert.

Doch wie realisiert man ein Fenster mit negativen Zeiten? Da ein Signalsegment einer DFT als Periode betrachtet werden kann, läßt sich das Signal periodisch fortsetzten. Um einen Puffer für die DFT vorzubereiten, liest man zuerst die Werte ab dem Fenstermaximum ein, anschließend erst die Werte vom Anfang bis zum Maximum. Abbildung 1.9 demonstriert dies bei der zusätzlichen Verwendung von zero-padding.

Eine weitere Möglichkeit, ein zero-phase window zu realisieren, ist, von den Phasenwerten der ungeraden Bins π zu subtrahieren:

$$\varphi(m) = \begin{cases} \arg\left(\underline{X}(m)\right) & , \text{ m gerade} \\ \arg\left(\underline{X}(m)\right) - \pi & , \text{ m ungerade} \end{cases} \qquad (1.64)$$

Abb. 1.8: Die inverse Kurzzeit-Fourier-Transformation. Demonstration von Overlap and Add. Die oberen vier Signale sind durch Rücktransformation der ersten vier Kurzzeitspektren und anschließender Verschiebung an die entsprechende Stelle entstanden, das untere ist die Summe. Die Hopsize beträgt ½ N_W.

Abb. 1.9: Realisierung eines zero-phase windows. Der zero-padding factor ist hier 2.

Kapitel 2

Bestimmung der deterministischen Komponente mittels analysis-by-synthesis–Methoden

2.1 Vold-Kalman Order Tracking Filter

Das Vold-Kalman-Filter ist ein Heterodyne-Filter mit variabler Mittenfrequenz und variabler Bandbreite, es wurde 1993 von Havard Vold und Jan Leuridan [Vold, Leuridan; 1993] vorgestellt. Mit einem solchen Bandpaßfilter lassen sich einzelne Teiltöne eines Signals isolieren. Dies setzt aber voraus, daß der Verlauf der Momentanfrequenz jedes einzelnen Teiltones bereits bekannt ist. Im Speziellen wurde dieses Filter für die Analyse von Geräuschen von Motoren entwickelt, wo oft auf eine Drehzahlreferenz zurückgegriffen werden kann. Bei dieser Kategorie von Geräuschen bezeichnet man die Teiltöne als Ordnungen (Motor-Ordnungen).
Ein Kalman-Filter ist ein verallgemeinerter RLS-Algorithmus (recursive least squares), der mit adaptiven Filtern stark verknüpft ist (siehe [Sayed, Kailath; 1994], [Haddad, Parsons; 1991] oder [Vaseghi; 1996]). Als Ausgang dient ein Ansatz in einem Zustandsraum (State Space Approach). Mit Hilfe eines Kalman-Filters läßt sich der Zustand eines nichtstationären Systems aus mit Rauschen behafteten Messungen bestimmen.

Gehen wir wieder vom bereits aus dem ersten Kapitel bekannten Signalmodell aus. Die deterministische Komponente besteht aus einer Summe von Teiltönen.

$$x_{det}(n) = \sum_{k=1}^{K} A_k(n) \cdot \cos(\varphi_k(n)) \qquad (2.1)$$

Handelt es sich um ein harmonisches Signal, so gilt für die Momentanphase des k-ten Teiltones:

$$\varphi_k(n) = k \cdot \sum_{m=0}^{n-1} \theta_1(m) + \varphi_k(0) \qquad (2.2)$$

Es genügt dann, daß der Verlauf der Grundfrequenz (fundamental frequency) $\theta_1(n)$ bekannt ist, zum Beispiel aus einer Drehzahlreferenz berechnet.

Es handelt sich beim Vold-Kalman-Filter nicht um ein herkömmliches Filter, das z.B. durch eine Differenzengleichung beschreibbar ist, sondern um einen Least Squares Algorithmus. Weiters besitzt es einen nicht-kausalen Charakter, weshalb es nur für "post processing"-Aufgaben verwendet werden kann.

Zunächst ist es erforderlich, einen Teilton als modulierte Trägerschwingung zu betrachten:

$$A_k(n)\cdot\cos(\varphi_k(n)) = m_k(n)\cdot c_k(n) + \overline{m_k(n)}\cdot\overline{c_k(n)} \qquad (2.3)$$

$c_k(n)$ ist der analytische Trägeroszillator (Carrier oder Phasor) des k-ten Teiltones, $\overline{c_k(n)}$ ist der konjugiert komplexe Phasor. Bei einem harmonischen Signal gilt:

$$c_k(n) = e^{j\cdot k\cdot\hat{\varphi}(n)} \quad \text{mit} \quad \hat{\varphi}(n) = \sum_{m=0}^{n-1}\hat{\theta}_1(m) \qquad (2.4)$$

Die Momentanfrequenz dieser komplexen Harmonischen ist die Schätzung der Grundfrequenz $\hat{\theta}_1(m)$. Bei nicht harmonischen Signalen gilt allgemein:

$$c_k(n) = e^{j\cdot\hat{\varphi}_k(n)} \qquad (2.5)$$

$m_k(n)$ ist der unbekannte Modulator des k-ten Teiltones, den es zu bestimmen gilt. Wenn $\varphi_k(n)$ gleich $\hat{\varphi}_k(n)$ ist, dann wird $m_k(n)$ reellwertig, moduliert wird dann also nur die Amplitude. Im Allgemeinen ist aber $\varphi_k(n)$ ungleich $\hat{\varphi}_k(n)$, da nur eine Schätzung des Grundfrequenzverlaufs vorliegt. Somit wird $m_k(n)$ komplexwertig und die Modulation eine Kombination aus Amplituden- und Phasenmodulation.

Der Ansatz für das Vold-Kalman-Filter sind zwei Gleichungen: die Struktur- und die Datengleichung ([Vold, Leuridan; 1993] und [Vold, Mains, Blough; 1997]).

Strukturgleichung:

$$\nabla^{p+1} m_k(n) = \varepsilon_k(n) \qquad (2.6)$$

∇^s steht für den Differenz-Operator der Ordnung s. Ein Vold-Kalman-Filter der Ordnung p (jetzt ist kein Teilton gemeint) benötigt einen Differenz-Operator der Ordnung p + 1. Für ein Filter erster Ordnung nimmt (2.6) folgende Gestalt an:

$$m_k(n-1) - 2\cdot m_k(n) + m_k(n+1) = \varepsilon_k(n) \qquad (2.7)$$

zweite Ordnung:

$$-m_k(n-1) + 3\cdot m_k(n) - 3\cdot m_k(n+1) + m_k(n+2) = \varepsilon_k(n) \qquad (2.8)$$

dritte Ordnung:

$$m_k(n-2) - 4 \cdot m_k(n-1) + 6 \cdot m_k(n) - 4 \cdot m_k(n+1) + m_k(n+2) = \varepsilon_k(n) \qquad (2.9)$$

Eine Minimierung von $\varepsilon_k(n)$ bewirkt eine Glättung des Modulators $m_k(n)$. Der Einfluß der Strukturgleichung ist somit der eines Tiefpaß-Filters.

Datengleichung:

$$x(n) - m_k(n) \cdot c_k(n) = \delta(n) \qquad (2.10a)$$

$$x(n) - \sum_{k \in \text{ Set}} m_k(n) \cdot c_k(n) = \delta(n) \qquad \text{wobei Set} = \{k_1, k_2, ..., k_K\} \qquad (2.10b)$$

In (2.10a) wird nur ein Teilton betrachtet, in (2.10b) mehrere gleichzeitig. Was dies bewirkt möchte ich jedoch erst später klären. Oft sind gar nicht alle Sinuskomponenten (k = 1, ..., K) von Interesse, sondern nur die dominanten, also eine Teilmenge, die wir mit den Indizes $\{k_1, ..., k_K\}$ bezeichnen. $\delta(n)$ kann man als (Teil-) Residuum betrachten. Eine Minimierung bewirkt, daß der Teilton (die Teiltöne) maximale Ähnlichkeit mit dem ursprünglichen Signal x(n) bekommt (bekommen).

Kombination:

Eine Kombination beider Gleichungen, bzw. ein gleichzeitiges Minimieren von $\varepsilon_k(n)$ und $\delta(n)$ wirkt wie ein Heterodyne-Filter: heruntermischen und tiefpaßfiltern.
Dies kann folgendermaßen veranschaulicht werden:

$$\overline{c_k}(n) \cdot x(n) = m_k(n) + \sum_{j \neq k} m_j(n)\, \overline{c_k}(n)\, c_j(n) \qquad (2.11)$$

Das anschließende Tiefpaßfiltern wird, wie bereits erwähnt, durch die Strukturgleichung realisiert, und es bleibt in (2.11) lediglich $m_k(n)$ übrig. Je stärker der Einfluß der Strukturgleichung ist, desto schmäler wird die Bandbreite des Vold-Kalman-Filters.

Ich definiere nun einige Spaltenvektoren und Matrizen, um auf eine bequemere Art ein Gleichungssystem aufzustellen. $\mathbf{m_k}$ beinhaltet die N Samples des unbekannten Modulators des k-ten Teiltones, $\mathbf{c_k}$ ist der dazugehörige Phasor. $\boldsymbol{\varepsilon_k}$ und $\boldsymbol{\delta}$ sind die zu minimierenden Fehlervektoren, $\mathbf{x}$ ist der Signalvektor. Str_p ist die reelle Koeffizientenmatrix der Strukturgleichung eines Filters der Ordnung p. Weiters gilt $C_k = \text{Diag}(\mathbf{c_k})$ und R_k ist der Gewichtsfaktor zwischen der k-ten Strukturgleichung und der Datengleichung. Letzterer kann

entweder ein reeller Skalar r_k sein, oder eine Diagonalmatrix mit dem Diagonalvektor $\mathbf{r_k}$ der Länge N.

$$R_k = \begin{cases} \mathrm{Diag}(\mathbf{r_k}), \text{ zeitvariant} \\ r_k \quad\quad , \text{ zeitinvariant} \end{cases} \tag{2.12}$$

Wählt man $R_k = r_k$ dann erhält man ein zeitinvariantes Filter, bei $R_k = \mathrm{Diag}(\mathbf{r_k})$ hat man die Möglichkeit, die Bandbreite zeitlich zu variieren, somit ist es möglich, z.B. ein Filter mit konstanter Relativbandbreite zu realisieren.
Schreibt man nun Struktur- und Datengleichung als gemeinsames System, so erhält man:

$$\begin{bmatrix} R_k \mathrm{Str}_p \\ C_k \end{bmatrix} \cdot \mathbf{m_k} - \begin{bmatrix} \mathbf{Null} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} R_k \boldsymbol{\varepsilon_k} \\ -\boldsymbol{\delta} \end{bmatrix} \tag{2.13a}$$

Hier wurde wieder nur ein Teilton berücksichtigt, möchte man mehreren gleichzeitig folgen, dann erweitert sich das Gleichungssystem zu

$$\begin{bmatrix} R_{k_1}\mathrm{Str}_p & 0 & \cdots & 0 \\ 0 & R_{k_2}\mathrm{Str}_p & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & R_{k_K}\mathrm{Str}_p \\ C_{k_1} & C_{k_2} & \cdots & C_{k_K} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{m_{k_1}} \\ \mathbf{m_{k_2}} \\ \vdots \\ \mathbf{m_{k_K}} \end{bmatrix} - \begin{bmatrix} \mathbf{Null} \\ \mathbf{Null} \\ \vdots \\ \mathbf{Null} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} R_{k_1}\boldsymbol{\varepsilon_{k_1}} \\ R_{k_2}\boldsymbol{\varepsilon_{k_2}} \\ \vdots \\ R_{k_K}\boldsymbol{\varepsilon_{k_K}} \\ \boldsymbol{\delta} \end{bmatrix} \tag{2.13b}$$

Str_p ist eine reellwertige (N mal N)-Bandmatrix mit den Koeffizienten der Strukturgleichung, wobei N die Anzahl der Samples von x(n) ist. Für ein Filter erster Ordnung sieht Str_p folgend aus:

$$\mathrm{Str}_1 = \begin{bmatrix} -2 & 1 & & & 0 & 0 \\ 1 & -2 & 1 & & & 0 \\ & 1 & \ddots & \ddots & & \\ & & \ddots & \ddots & 1 & \\ 0 & & & 1 & -2 & 1 \\ 0 & 0 & & & 1 & -2 \end{bmatrix} \tag{2.14}$$

Bei ungerader Filterordnung ist diese Matrix symmetrisch. Die (N mal N)-Diagonalmatrix C_k ist nicht reellwertig. Die Elemente von **Null** sind natürlich Nullen.

Das lineare Gleichungssystem in (2.13a) ist überbestimmt: N Unbekannte aber 2N Gleichungen, bzw. $K \cdot N$ Unbekannte und (K+1)N Gleichungen in (2.13b). Gesucht sind also jene $\mathbf{m_k}$, die Gleichung (2.15) am besten im Sinne von Least Squares erfüllen.

$$A \cdot \mathbf{m} \approx \mathbf{b} \tag{2.15}$$

m fasst alle $\mathbf{m_k}$ zusammen, $\mathbf{b} = \left[0, 0, ..., 0, \mathbf{x}^T\right]^T$.

Der Fehlervektor oder auch Residuum genannt ist dann

$$\mathbf{res} = A \cdot \mathbf{m} - \mathbf{b} = \begin{bmatrix} R_{k_1} \boldsymbol{\varepsilon}_{\mathbf{k_1}} \\ \vdots \\ R_{k_K} \boldsymbol{\varepsilon}_{\mathbf{k_K}} \\ -\boldsymbol{\delta} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\varepsilon} \\ -\boldsymbol{\delta} \end{bmatrix} \tag{2.16}$$

Bei Least Squares Aufgaben soll eine Kostenfunktion, die aus dem quadratischen Fehler besteht, minimiert werden. Die Kostenfunktion für das Vold-Kalman-Filter wird zu

$$E = \sum_{n=0}^{K \cdot N-1} |\varepsilon(n)|^2 + \sum_{n=0}^{N-1} |\delta(n)|^2 = \overline{\boldsymbol{\varepsilon}}^T \boldsymbol{\varepsilon} + \overline{\boldsymbol{\delta}}^T \boldsymbol{\delta} \tag{2.17a}$$

oder etwas kürzer:

$$E = \overline{\mathbf{res}}^T \mathbf{res} = \|\mathbf{res}\|^2 \tag{2.17b}$$

2.1.1 Die Normalgleichungen

Wir suchen also jenen Vektor **m**, bei dem E ein Minimum besitzt. Das erreicht man durch Null setzen des Gradienten von E. Das so entstandene Gleichungssystem nennt man die Normalgleichungen.

Unsere Kostenfunktion ist zwar reellwertig, hängt aber von einem komplexwertigen Vektor **m** ab. Berechnen wollen wir nun den Gradienten von E:

$$\nabla E(\mathbf{m}) = \begin{bmatrix} \dfrac{\partial E(\mathbf{m})}{\partial M(0)} \\ \vdots \\ \dfrac{\partial E(\mathbf{m})}{\partial M(N-1)} \end{bmatrix} \tag{2.18}$$

Und da ergibt sich bereits ein Problem, da E keine analytische Funktion ist (wegen Betrag bzw. Konjugation). Man könnte getrennt nach Real- und Imaginärteil ableiten oder man interpretiert die Kostenfunktion als

$$E\left(\overline{\mathbf{m}}, \mathbf{m}\right) = \left(\overline{\mathbf{m}}^T \overline{A}^T - \overline{\mathbf{b}}^T\right)\left(A\,\mathbf{m} - \mathbf{b}\right) \tag{2.19}$$

und leitet nur nach $\overline{\mathbf{m}}$ oder nach $\mathbf{m}$ ab. Denn auch diese beiden Ableitungen sind konjugiert, geht die eine nach **Null**, so auch die andere.

$$\nabla E\left(\mathbf{m}\right) = \frac{\partial E\left(\overline{\mathbf{m}}, \mathbf{m}\right)}{\partial \overline{\mathbf{m}}} = \overline{A}^T \mathbf{res} \tag{2.20}$$

Durch nullsetzen erhält man:

$$\overline{A}^T A\,\mathbf{m} - \overline{A}^T \mathbf{b} = \mathbf{0} \quad \Rightarrow \quad \boxed{\overline{A}^T A\,\mathbf{m} = \overline{A}^T \mathbf{b}} \tag{2.21}$$

Dieses neue Gleichungssystem ist nicht mehr überbestimmt, K·N Gleichungen und K·N Unbekannte. Weiters ist die Koeffizientenmatrix $\overline{A}^T A$ symmetrisch (bzw. hermitesch).
$\overline{A}^T A\,\mathbf{m} = \overline{A}^T \mathbf{b}$ wird in der Literatur als die Normalgleichungen von $A \cdot \mathbf{m} \approx \mathbf{b}$ (aber auch von $A \cdot \mathbf{m} = \mathbf{b}$) bezeichnet. Ein Nachteil von $\overline{A}^T A$ ist bei manchen Problemstellungen die schlechte Konditionierung (die Konditionierungszahl von $\overline{A}^T A$ ist das Quadrat jener von A).

Ich möchte hier die Normalgleichungen des Vold-Kalman-Filters anführen:

$$\begin{bmatrix} B_{k_1} & D_{1,2} & \cdots & D_{1,K} \\ D_{2,1} & B_{k_2} & \cdots & D_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ D_{K,1} & D_{K,2} & \cdots & B_{k_K} \end{bmatrix} \mathbf{m} = \begin{bmatrix} \overline{C}_{k_1}\mathbf{x} \\ \overline{C}_{k_2}\mathbf{x} \\ \vdots \\ \overline{C}_{k_K}\mathbf{x} \end{bmatrix} \tag{2.22}$$

mit

$$B_k = Str_p^T R_k^2 Str_p + I \tag{2.23}$$

wobei I die Einheitsmatrix ist und den Koppelmatrizen

$$D_{u,v} = \mathrm{Diag}(\overline{\mathbf{c}}_{\mathbf{k_u}} \circ \mathbf{c}_{\mathbf{k_v}}) \tag{2.24}$$

Der Vektor auf der rechten Seite in (2.22) beinhaltet die für die einzelnen Trägerfrequenzen heruntergemischten Signale.

Wenn wir die Teiltöne einzeln und nicht gekoppelt extrahieren möchten, bekommen wir anstelle von (2.22) eine Menge kleinerer Gleichungssysteme:

$$B_{k_i}\,\mathbf{m}_{\mathbf{k_i}} = \overline{C}_{k_i}\,\mathbf{x} \qquad \text{für } i = 1, ..., K \tag{2.25}$$

Hier verzichtet man auf die Koppelelemente. Der Vorteil der Koppelung ist, daß nahe beieinander liegende Sinuskomponenten (oder sich kreuzende) ohne Schwebungseffekte extrahiert werden können. Der Nachteil ist das größere und weniger spärlich besetzte Gleichungssystem.

Eine direkte Lösung der Gleichung (2.22) bzw. (2.25) ist wahrscheinlich nicht möglich, denn N ist im Allgemeinen sehr groß. Als Alternative muß ein iteratives Lösungsverfahren angewendet werden.

Quadratische Form

An dieser Stelle möchte ich zeigen, daß das Aufsuchen des Minimums der Quadratischen Form (2.26) äquivalent ist mit dem Lösen des linearen Gleichungssystems $A\,\mathbf{x} = \mathbf{b}$.

$$Q(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\,\mathbf{x} - \mathbf{b}^T\mathbf{x} + c \tag{2.26}$$

Der Gradient von Q(**x**) ist

$$\nabla Q(\mathbf{x}) = \frac{1}{2}A^T\mathbf{x} + \frac{1}{2}A\,\mathbf{x} - \mathbf{b} \tag{2.27}$$

Wenn A eine symmetrische (bzw. hermitesche) Matrix ist, dann vereinfacht sich der Gradient zu

$$\nabla Q(\mathbf{x}) = A\,\mathbf{x} - \mathbf{b} = \mathbf{res} \tag{2.28}$$

Nullsetzen liefert folglich $A\,\mathbf{x} = \mathbf{b}$.

Ist A weiters positiv definit, so ist garantiert, daß Q bei **x** ein globales Minimum besitzt.

Wie leicht zu sehen ist, ist es einfacher Q(**x**) zu minimieren anstelle von $\|\mathbf{res}\|^2$. Deshalb sollten zur Lösung des Vold-Kalman-Filters zuerst die Normalgleichungen gebildet werden, um eine symmetrische Matrix zu erhalten und anschließend die einfache quadratische Form in (2.26) mit einem Standard-Minimierungsverfahren (zB. CG) minimiert werden.

2.1.2 **Gradientenmethode** (Cauchy´s method of steepest descent (SD))

Eine der ältesten und einfachsten Minimierungsmethoden ist das Verfahren des steilsten Abstieges. Hier nähert man sich schrittweise entgegen der Richtung des Gradienten an ein Minimum der Kostenfunktion E(**x**) (siehe [Kreyszig; 1999]). Mit $^{(i)}$ weise ich auf den i-ten Iterationsschritt hin. Zu Beginn muß ein Startvektor $\mathbf{x}^{(0)}$ gewählt werden. Die Iterierten berechnen sich nach

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \alpha^{(i)} \cdot \nabla \mathrm{E}\left(\mathbf{x}^{(i)}\right) \tag{2.29}$$

Die Berechnung von $\alpha^{(i)}$ erfolgt durch Aufsuchen des Minimums der Funktion g, die nur mehr von $\alpha^{(i)}$ abhängt.

$$\mathrm{g}\left(\alpha^{(i)}\right) = \mathrm{E}\left(\mathbf{x}^{(i+1)}\right) = \mathrm{E}\left(\mathbf{x}^{(i)} - \alpha^{(i)} \cdot \nabla \mathrm{E}\left(\mathbf{x}^{(i)}\right)\right) \tag{2.30}$$

$$\frac{\mathrm{dg}\left(\alpha^{(i)}\right)}{\mathrm{d}\alpha^{(i)}} = 0 \quad \Rightarrow \quad \alpha^{(i)} = \cdots \tag{2.31}$$

Wenn nun die einfache quadratische Form aus (2.26) unsere Kostenfunktion ist, bzw. wir die Lösung von $\mathrm{A}\,\mathbf{x} = \mathbf{b}$ suchen, wobei A symmetrisch und positiv definit ist, dann vereinfachen sich die Schritte der Gradientenmethode zu

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \alpha^{(i)} \cdot \mathbf{res}^{(i)} \tag{2.32}$$

mit

$$\alpha^{(i)} = \frac{\mathbf{res}^{(i)^{\mathrm{T}}} \mathbf{res}^{(i)}}{\mathbf{res}^{(i)^{\mathrm{T}}} \mathrm{A}\, \mathbf{res}^{(i)}} \tag{2.33}$$

wobei

$$\mathbf{res}^{(i)} = \mathrm{A}\,\mathbf{x}^{(i)} - \mathbf{b} \tag{2.34}$$

Um Rechenzeit zu sparen kann der Residuumvektor auch rekursiv berechnet werden:

$$\mathbf{res}^{(i+1)} = \mathrm{A}\left(\mathbf{x}^{(i)} - \alpha^{(i)} \mathbf{res}^{(i)}\right) - \mathbf{b} = \mathbf{res}^{(i)} - \alpha^{(i)} \mathrm{A}\, \mathbf{res}^{(i)} \tag{2.35}$$

Anstelle des Matrix-Vektor-Produktes $\mathrm{A}\,\mathbf{x}^{(i)}$ wird jetzt $\mathrm{A}\,\mathbf{res}^{(i)}$ benötigt, das bereits in Gleichung (2.33) verwendet wurde.

Erfreulich ist, daß pro Iterationsschritt lediglich eine Matrix-Vektor-Multiplikation als einzige aufwendigere Operation durchgeführt werden muß. Bei einer spärlich besetzten Matrix A muß diese erst gar nicht aufgespannt werden, womit Speicherplatz gespart werden kann.

Der große Nachteil dieser Methode ist, daß sie im Allgemeinen sehr langsam konvergiert, weshalb sie nur selten Anwendung findet. Wesentlich öfters wird hingegen die Methode der konjugierten Gradienten verwendet.

2.1.3 **Methode der konjugierten Gradienten** (Conjugate Gradients Method (CG))

Auch hier wollen wir wieder die quadratische Form $Q(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\,\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$ minimieren. Bei jedem Schritt nähern wir uns entgegen dem Richtungsvektor $\mathbf{d}^{(i)}$ an die Lösung $\mathbf{x}$, $\mathbf{x}^{(0)}$ wird wieder gewählt.

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \alpha^{(i)} \cdot \mathbf{d}^{(i)} \tag{2.36}$$

Der erste Schritt verläuft gleich wie bei SD, also ist $\mathbf{d}^{(0)} = \nabla Q\left(\mathbf{x}^{(0)}\right) = \mathbf{res}^{(0)}$. Für alle weiteren Richtungsvektoren muß jedoch folgende Bedingung gelten (siehe [Golub, Ortega; 1993]):

$$\mathbf{d}^{(i)^T} A\,\mathbf{d}^{(j)} = 0, \quad i \neq j \tag{2.37}$$

Das bedeutet, daß diese Vektoren orthogonal bezüglich A, bzw. konjugiert relativ zu A sind. Weiters muß gelten, daß sie ungleich dem Nullvektor sind. Wenn A eine (N mal N)-Matrix ist, dann können maximal N Vektoren die Bedingung (2.37) erfüllen, und tatsächlich konvergieren für jeden Startvektor $\mathbf{x}^{(0)}$ die Iterierten gegen die Lösung von $A\,\mathbf{x} = \mathbf{b}$ in nicht mehr als N Schritten, falls A eine reelle, symmetrische, positiv definite Matrix ist.
Durch diese obere Grenze an Iterationen kann dieses Verfahren auch als direktes Lösungsverfahren betrachtet werden.

Aus (2.37) folgt für die Berechnung der Richtungsvektoren (ohne Herleitung)

$$\mathbf{d}^{(i+1)} = \beta^{(i)}\mathbf{d}^{(i)} - \mathbf{res}^{(i+1)} \tag{2.38}$$

$$\beta^{(i)} = \frac{\mathbf{res}^{(i+1)^T}\mathbf{res}^{(i+1)}}{\mathbf{res}^{(i)^T}\mathbf{res}^{(i)}} \tag{2.39}$$

Die Berechnung von $\alpha^{(i)}$ ist ähnlich wie bei SD:

$$\alpha^{(i)} = \frac{\mathbf{res}^{(i)^T}\mathbf{res}^{(i)}}{\mathbf{d}^{(i)^T} A\,\mathbf{d}^{(i)}} \tag{2.40}$$

Auch das Residuum läßt sich wieder rekursiv berechnen:

$$\mathbf{res}^{(i+1)} = A\left(\mathbf{x}^{(i)} - \alpha^{(i)}\mathbf{d}^{(i)}\right) - \mathbf{b} = \mathbf{res}^{(i)} - \alpha^{(i)} A\,\mathbf{d}^{(i)} \qquad (2.41)$$

Ein Grund für die Beliebtheit von CG ist, daß es schnell konvergiert und einfach zu realisieren ist. Wie bei SD ist auch hier die einzige aufwendigere Operation pro Iteration ein Matrix-Vektor-Produkt: $A\,\mathbf{d}^{(i)}$.

Beim Vold-Kalman-Filter zum Folgen von nur einem Teilton ist die Matrix A eine spärlich besetzte Bandmatrix. Daß dieses möglicherweise alle Speicherkapazitäten sprengende Ungetüm gar nicht erst aufgespannt werden muß, möchte ich anhand eines Filters erster Ordnung zeigen, bei dem die Matrix $A = B_k = Str_1 R_k^2 Str_1 + I$ ist. Das Matrix-Vektor-Produkt $A\,\mathbf{d}$ läßt sich folgendermaßen berechnen: $A\,\mathbf{d} = Str_1\left(\mathbf{r_k} \circ \left(\mathbf{r_k} \circ (Str_1\,\mathbf{d})\right)\right) + \mathbf{d}$, wobei $\mathbf{v} \circ \mathbf{w}$ für das Schur-Produkt (Element für Element) steht. Zweimal ist die Operation $\mathbf{w} = Str_1\,\mathbf{v}$ durchzuführen, was ebenfalls ohne eine Matrix möglich ist: $w_n = v_{n-1} - 2v_n + v_{n+1}$. Die Randzonen müssen dazu mit Nullen erweitert werden.

Nach einigen Versuchen hat sich gezeigt, daß unser Gleichungssystem sehr schlecht konditioniert ist, d.h. die Iterierten konvergieren nur langsam. Man stellt dann auch fest, daß die Anzahl der Iterationen nicht mit N beschränkt ist, wenn N die Anzahl der Unbekannten ist. Diese Beschränkung ist nämlich nur theoretischer Natur, da exaktes Rechnen vorausgesetzt wird. In der Praxis sind Rechenoperationen nur mit endlicher Genauigkeit möglich, es kommt zu Rundungsfehlern. Wenn das System sehr schlecht konditioniert ist (die Konditionierungszahl von A ist sehr groß), ist der Lösungsprozeß extrem empfindlich gegen Rundungsfehler.
Eine Verbesserung kann mit der sogenannten Präkonditionierung erreicht werden.

2.1.4 Präkonditionierung

Wir lösen anstelle von $A\mathbf{x} = \mathbf{b}$ das durch die Kongruenztransformation $\hat{A} = S\,A\,S^T$ erhaltene System $\hat{A}\hat{\mathbf{x}} = \hat{\mathbf{b}}$ mit $\hat{\mathbf{x}} = S^{-T}\mathbf{x}$ und $\hat{\mathbf{b}} = S\mathbf{b}$, sodaß gilt: $\mathrm{cond}(\hat{A}) < \mathrm{cond}(A)$.
Ohne weiterer Herleitung möchte ich hier den vollständigen präkonditionierten CG-Algorithmus zum Lösen von $A\mathbf{x} = \mathbf{b}$ vorstellen (siehe [Golub, Ortega; 1993]):

Wähle $\mathbf{x}^{(0)}$. Setze $\mathbf{res}^{(0)} = A\mathbf{x}^{(0)} - \mathbf{b}$

Löse $M\hat{\mathbf{r}}^{(0)} = \mathbf{res}^{(0)}$. Setze $\mathbf{d}^{(0)} = \hat{\mathbf{r}}^{(0)}$

Wiederhole für k = 0, 1, ...

$$\alpha^{(k)} = \frac{\hat{\mathbf{r}}^{(k)^T}\mathbf{res}^{(k)}}{\mathbf{d}^{(k)^T} A\,\mathbf{d}^{(k)}}$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha^{(k)}\mathbf{d}^{(k)}$$

$$\mathbf{res}^{(k+1)} = \mathbf{res}^{(k)} - \alpha^{(k)} A\,\mathbf{d}^{(k)}$$

Löse $M\hat{\mathbf{r}}^{(k+1)} = \mathbf{res}^{(k+1)}$

Konvergenztest, Iterationsabbruch

$$\beta^{(k)} = \frac{\hat{\mathbf{r}}^{(k+1)^T}\mathbf{res}^{(k+1)}}{\hat{\mathbf{r}}^{(k)^T}\mathbf{res}^{(k)}}$$

$$\mathbf{d}^{(k+1)} = \beta^{(k)}\mathbf{d}^{(k)} - \hat{\mathbf{r}}^{(k+1)} \qquad (2.42)$$

Die Matrix M ist durch $M = \left(S^T S\right)^{-1}$ definiert und hat somit symmetrisch und positiv definit zu sein. Wir bezeichnen sie als Präkonditionierungsmatrix. Für die Wahl von M gibt es zwei allgemeine Kriterien: erstens sollte M ähnlich zu A sein. Wählt man M = A kann die Konditionierungszahl von $\hat{A}$ auf eins reduziert werden. Zweitens muß M einfach zu invertieren sein, da $M\hat{\mathbf{r}}^{(k)} = \mathbf{res}^{(k)}$ in jedem Iterationsschritt gelöst werden muß. Offensichtlich stehen diese beiden Kriterien in einem Konflikt zueinander, denn ließe sich A leicht invertieren, so wäre die gesamte Auseinandersetzung mit iterativen Lösungsmethoden überflüssig.

Kehren wir nun wieder zum Vold-Kalman-Filter zurück. Wenn wir auf die Koppelung verzichten, ist die Matrix des Gleichungssystems $A = Str_p^T R_k^2 Str_p + I$. Verzichten wir nicht, so stellt diese Matrix trotzdem den dominierenden Teil in der Hauptdiagonale dar. Für sehr große R_k gewinnt $Str_p^T Str_p$ an Bedeutung. Eine gute Näherung für diese Matrix ist:

$$Str_p^T Str_p \approx \left(L\,U\right)^{p+1} = M \qquad (2.43)$$

mit

$$L = \begin{bmatrix} 1 & & & & 0 \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ 0 & & & -1 & 1 \end{bmatrix} \quad \text{und} \quad U = L^T = \begin{bmatrix} 1 & -1 & & & 0 \\ & 1 & -1 & & \\ & & 1 & \ddots & \\ & & & \ddots & -1 \\ 0 & & & & 1 \end{bmatrix} \qquad (2.44)$$

Bei einem Filter erster Ordnung ist die Näherung beinahe perfekt:

$$\mathrm{Str}_1^{\mathrm{T}}\mathrm{Str}_1=\begin{bmatrix} 5 & -4 & 1 & & & 0 \\ -4 & 6 & -4 & 1 & & \\ 1 & -4 & 6 & \ddots & \ddots & \\ & 1 & \ddots & \ddots & -4 & 1 \\ & & \ddots & -4 & 6 & -4 \\ 0 & & & 1 & -4 & 5 \end{bmatrix} \approx \left(\mathrm{L}\,\mathrm{U}\right)^2 = \begin{bmatrix} \underline{2} & \underline{-3} & 1 & & & 0 \\ \underline{-3} & 6 & -4 & 1 & & \\ 1 & -4 & 6 & \ddots & \ddots & \\ & 1 & \ddots & \ddots & -4 & 1 \\ & & \ddots & -4 & 6 & -4 \\ 0 & & & 1 & -4 & 5 \end{bmatrix} \tag{2.45}$$

Auch bei höheren Ordnungen sind die Unterschiede gering und nur im Randbereich. Somit wäre die Forderung nach Ähnlichkeit zu A erfüllt. Weiters sollte nun die Inversion einfach durchzuführen sein:

$$\mathrm{M}^{-1} = \left(\mathrm{U}^{-1}\mathrm{L}^{-1}\right)^{p+1} \tag{2.46}$$

$$\mathrm{L}^{-1} = \begin{bmatrix} 1 & & & & 0 \\ 1 & 1 & & & \\ 1 & 1 & 1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \quad \text{und} \quad \mathrm{U}^{-1} = \left(\mathrm{L}^{-1}\right)^{\mathrm{T}} \tag{2.47}$$

Die Lösung der Nebensysteme in (2.42) erfordert eine Matrix-Vektor-Multiplikation mit M^{-1}, die Dank der trivialen Dreiecksmatrix L^{-1} sehr einfach durch Berechnung von Kumulativsummen durchführbar ist.

Die Multiplikation von L^{-1} mit einem Vektor **v** ist:

$$\mathrm{L}^{-1}\mathbf{v} = \mathrm{cumsum}(\mathbf{v}) \tag{2.48}$$

und die von U^{-1} mit **v**:

$$\mathrm{U}^{-1}\mathbf{v} = \mathrm{flip}\left(\mathrm{cumsum}\left(\mathrm{flip}(\mathbf{v})\right)\right) \tag{2.49}$$

wobei

$$\mathrm{cumsum}(\mathbf{v}) = \begin{bmatrix} \mathrm{v}_1 \\ \mathrm{v}_1 + \mathrm{v}_2 \\ \vdots \\ \mathrm{v}_1 + \mathrm{v}_2 + \ldots + \mathrm{v}_N \end{bmatrix} \tag{2.50}$$

und

$$\mathrm{flip}(\mathbf{v}) = \begin{bmatrix} \mathrm{v}_N \\ \mathrm{v}_{N-1} \\ \vdots \\ \mathrm{v}_1 \end{bmatrix} \tag{2.51}$$

Somit ergibt sich für eine Multiplikation mit M^{-1}:

$$\hat{\mathbf{r}}^{(k)} = \ldots U^{-1}L^{-1}\,\mathbf{res}^{(k)} = \ldots \mathrm{flip}\left(\mathrm{cumsum}\left(\mathrm{flip}\left(\mathrm{cumsum}(\mathbf{res}^{(k)})\right)\right)\right) \qquad (2.52)$$

Anmerken möchte ich hier, daß cumsum(**v**) bloß (N-1) Additionen benötigt.

Wie großartig diese Präkonditionierung den Lösungsvorgang beschleunigt, möchte ich in Abbildung 2.1 zeigen. Hier sieht man für ein Filter erster Ordnung die ursprüngliche Kostenfunktion aus Gleichung (2.17), die in jedem Iterationsschritt zusätzlich berechnet wurde. Die obere Kurve resultiert aus einem gewöhnlichen CG-Algorithmus (M = I), die untere aus einem präkonditionierten mit M nach Gleichung (2.43). Die Anzahl der Unbekannten (N) ist 10000 und der Gewichtungsfaktor (r) ist 10^5. PCG findet die Lösung in diesem Beispiel bereits nach ungefähr 50 Iterationen, CG hingegen benötigt unvergleichbar länger, beziehungsweise erreicht die Lösung aufgrund der schlechten Konditionierung gar nie.

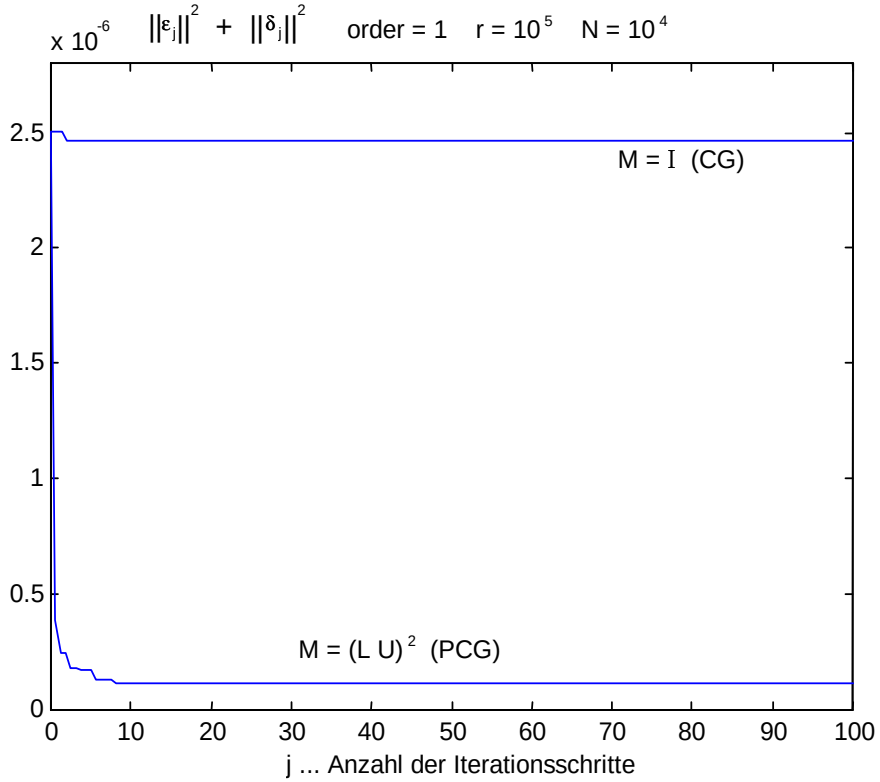


Abb. 2.1: Auswirkung der Präkonditionierung (Darstellung der zu minimierenden Ur-Kostenfunktion in Abhängigkeit von der Anzahl der Iterationsschritte)

Eine weitere positive Auswirkung der Präkonditionierung ist, daß besonders bei Filtern mit kleiner Bandbreite (große Werte für r) der Lösungsvorgang enorm verkürzt wird, bei CG hingegen wird mit größer werdendem r die Konditionierung immer schlechter.

Vergleicht man allerdings den Rechenzeitaufwand eines Vold-Kalman-Filters, das mittels oben angeführtem PCG-Algorithmus gelöst wurde, mit dem eines herkömmlichen Heterodyne-Filters, sieht man, daß ersterer viel größer ist. Es stellt sich nun die Frage, ob es effizientere Methoden zur Lösung gibt, z.B. mittels Prädiktor.

2.1.5 Eigenschaften des Vold-Kalman-Filters

Wie bereits öfters erwähnt, kann mit dem Gewichtungsfaktor R_k die Bandbreite beeinflußt werden. Nach zahlreichen Versuchen und Messungen ergaben sich folgende Zusammenhänge:

Bandbreite

Für ein Filter erster Ordnung berechnet sich die -3 dB Bandbreite nach

$$B_{3dB} = 1.58 \quad r^{-1/2} \tag{2.53}$$

und für eines zweiter Ordnung nach

$$B_{3dB} = 1.70 \quad r^{-1/3} \tag{2.54}$$

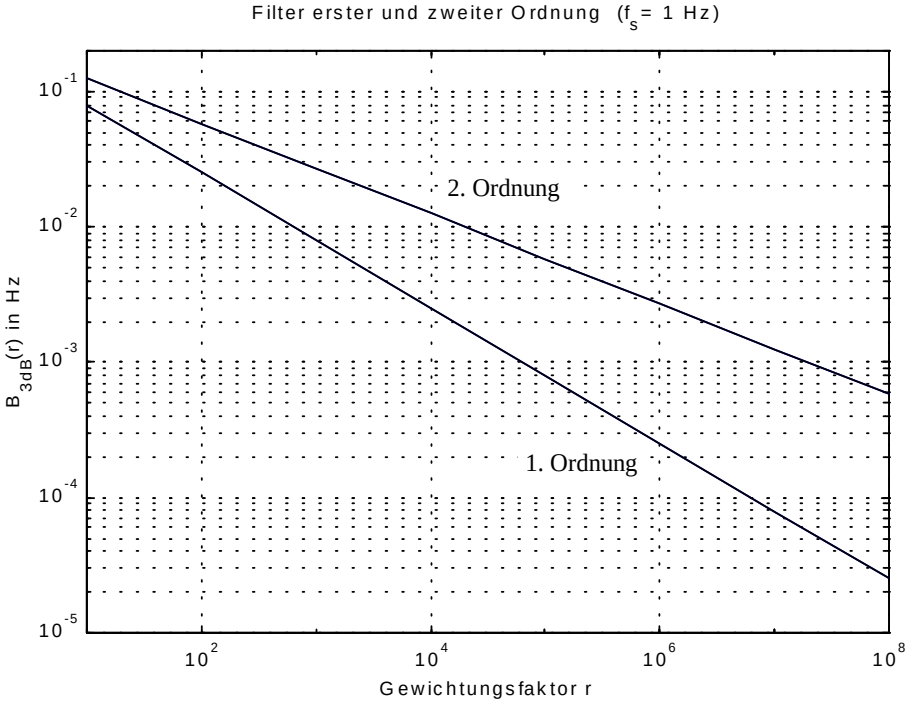


Abb. 2.2: Bandbreite als Funktion vom Gewichtungsfaktor r

Die Einheit von B_{3dB} ist in (2.53) und (2.54) in Radiant, um unabhängig von der Abtastrate zu bleiben. Eine Umrechnung in Hz erfordert eine Multiplikation mit $f_s/2\pi$. In doppellogarithmischer Darstellung erhält man jeweils eine Gerade, wie in Abbildung 2.2 gezeigt wird.

Filterform

Wie auch bei anderen Filtern werden die Flanken mit höherer Ordnung steiler.

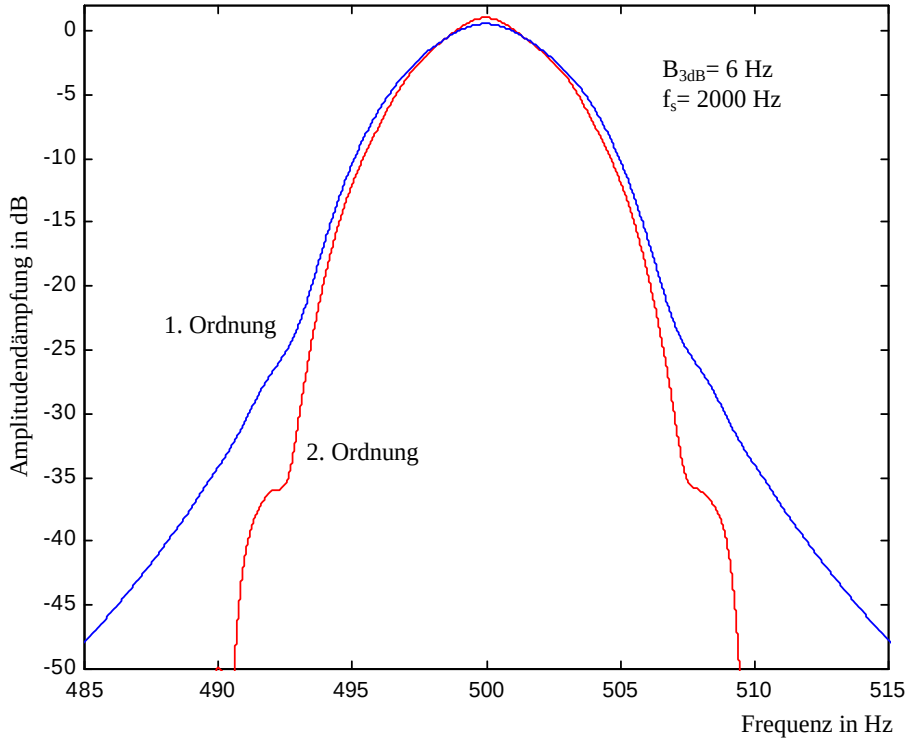


Abb. 2.3: Form der Filter erster und zweiter Ordnung

Verhalten im Zeitbereich

Filter mit geringer Bandbreite haben schlechtes zeitliches Verhalten, d.h. plötzlichen Änderungen der Amplitude des Signals kann nur sehr langsam gefolgt werden.In Abbildung 2.4 sieht man das Ergebnis eines Vold-Kalman-Tiefpasses erster Ordnung (kein Heruntermischen des Signals, Phasor ist konstante 1-Folge). Als Eingangssignal wurde die eingezeichnete Stufe verwendet. Deutlich zu erkennen ist hier, daß ein Vold-Kalman-Filter in den Randbereichen Fehler verursacht, da auch dort ein Einschwingen stattfindet. Weiters wird hier auch der akausale Charakter sichtbar.

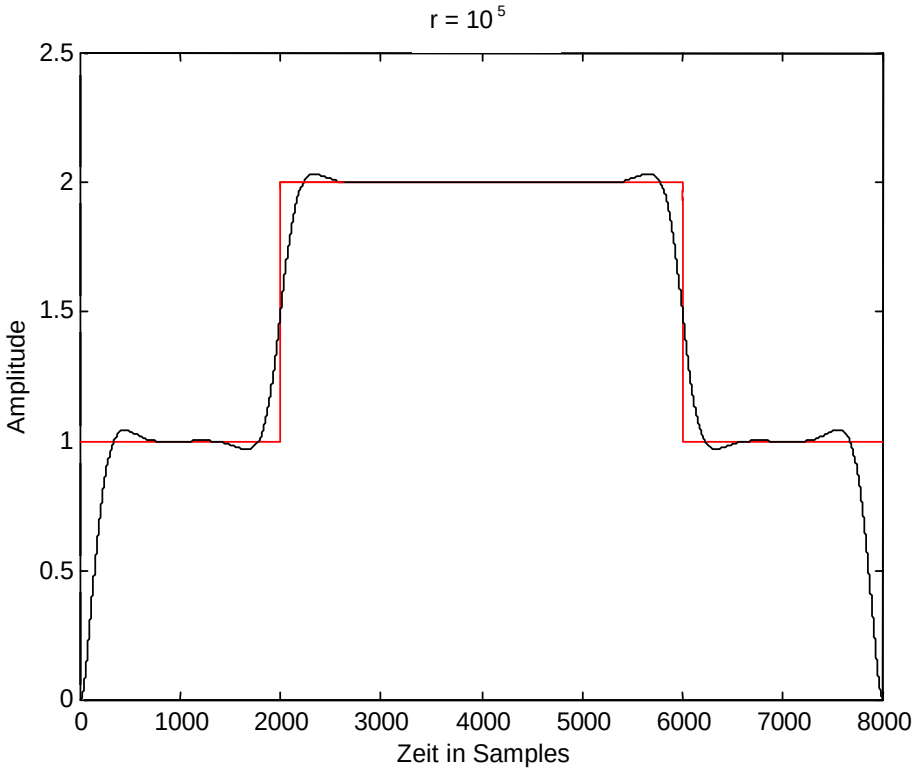


Abb. 2.4: Zeitliches Verhalten eines Vold-Kalman-Tiefpasses

Versuche dieser Art dienten zum Messen der 10% - 90% Anstiegszeit T_A. In diesem Beispiel wäre dies die Anzahl der Samples zwischen der Stelle wo die Amplitude 1.1 beträgt und jener Stelle bei 1.9.
Für ein Filter erster Ordnung konnte folgender Zusammenhang zwischen T_A und B_{3dB} gefunden werden:

$$B_{3dB} \cdot T_A = 4.51 \qquad (2.55)$$

wobei die Einheit für B_{3dB} Radiant ist, die für T_A Samples. Weiters läßt sich T_A in Abhängigkeit vom Gewichtungsfaktor r nach Gleichung (2.59) berechnen.

$$T_A = 2.85 \quad r^{1/2} \qquad (2.56)$$

Für ein Filter zweiter Ordnung gilt:

$$B_{3dB} \cdot T_A = 4.75 \qquad (2.57)$$

und

$$T_A = 2.80 \quad r^{1/3} \qquad (2.58)$$

Die Umrechnung von T_A in Sekunden erfordert natürlich eine Multiplikation mit dem Abtastintervall $T_S = 1 / f_S$.

Abschließend sollte noch erwähnt werden, daß die Meßergebnisse mit großen Streuungen auftraten und daher die Formeln (2.53) bis (2.58) noch nicht als endgültige Referenzen angesehen werden sollten. Mehr über die Eigenschaften des Vold-Kalman-Filters ist in [Vold, Herlufsen, u.a.; 1999] zu finden.

Gekoppeltes Heterodyne-Filter

Wie schon öfters erwähnt, können mehrere Teiltöne gleichzeitig extrahiert werden, indem mehrere Strukturgleichungen und eine umfangreichere Datengleichung als gemeinsames Gleichungssystem angeschrieben werden. In den Normalgleichungen treten dann die Koppelelemente $D_{u,v}$ auf, die bei separatem Filtern hingegen weggelassen werden.
Das folgende Beispiel soll den Vorteil des Koppelns veranschaulichen. Das dazu verwendete Signal wurde künstlich erzeugt und besitzt zwei Sinuskomponenten. In Abbildung 2.5a sieht man die sich kreuzenden Frequenzverläufe. Die Amplitude der ersten ist konstant 2, die der zweiten von 0.5 auf 1 linear ansteigend. Als Filter verwendete ich ein Vold-Kalman-Filter erster Ordnung mit $r = 10^4$ (das entspricht bei $f_s = 200$ Hz einer 3dB-Bandbreite von 0.5 Hz). Die Ergebnisse des ungekoppelten Filters sind in Abbildung 2.5b zu sehen, es treten hier im Bereich der Frequenzverlauf-Kreuzungen erhebliche Schwebungseffekte auf. Auch ein beliebiges anderes Heterodyne-Filter hätte die gleichen Probleme in diesem Bereich. In Abbildung 2.5c ist der Amplitudenverlauf des gekoppelten Filters dargestellt, der frei von Schwebungseffekten ist.

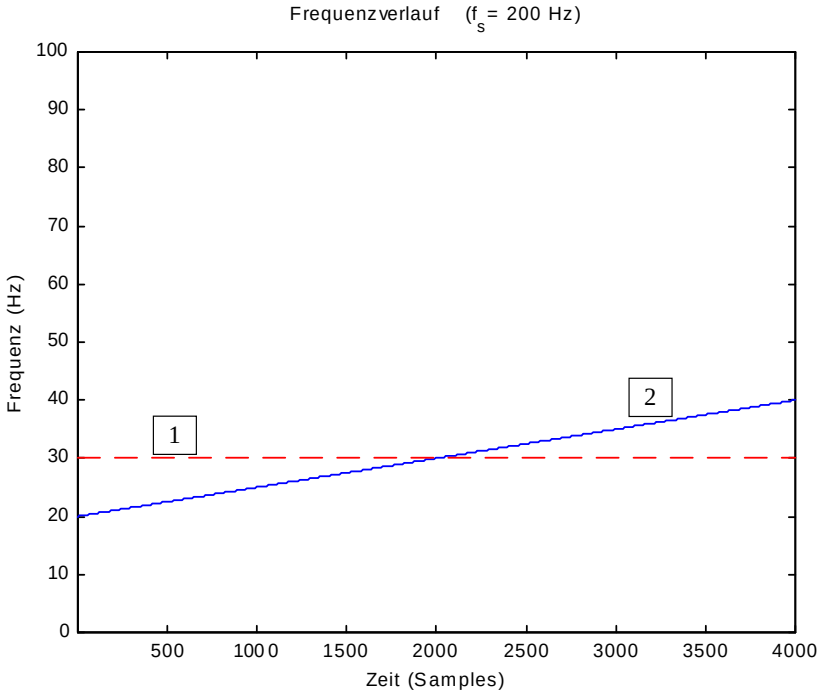


Abb. 2.5a: Frequenzverlauf der beiden Sinuskomponenten

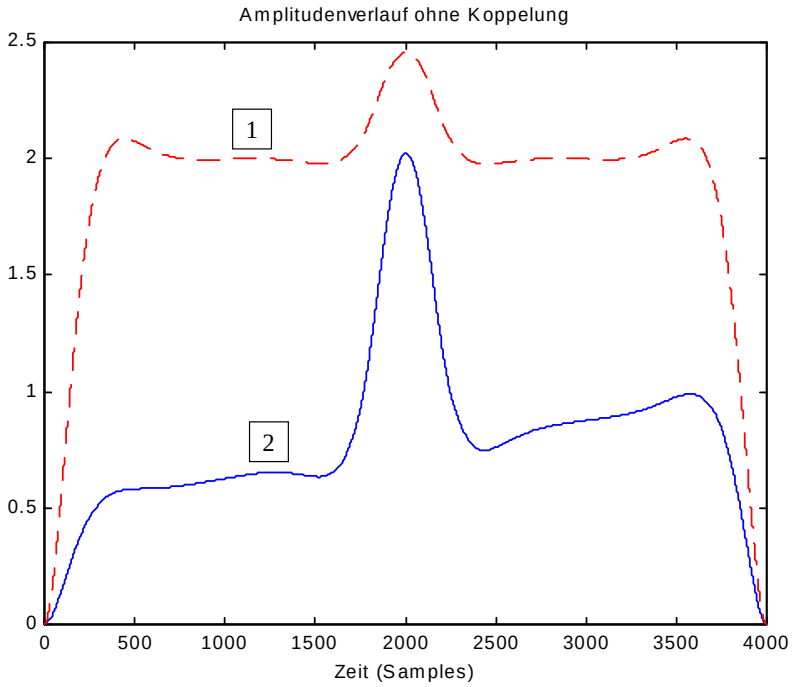


Abb. 2.5b: Ergebnis ohne Koppelung

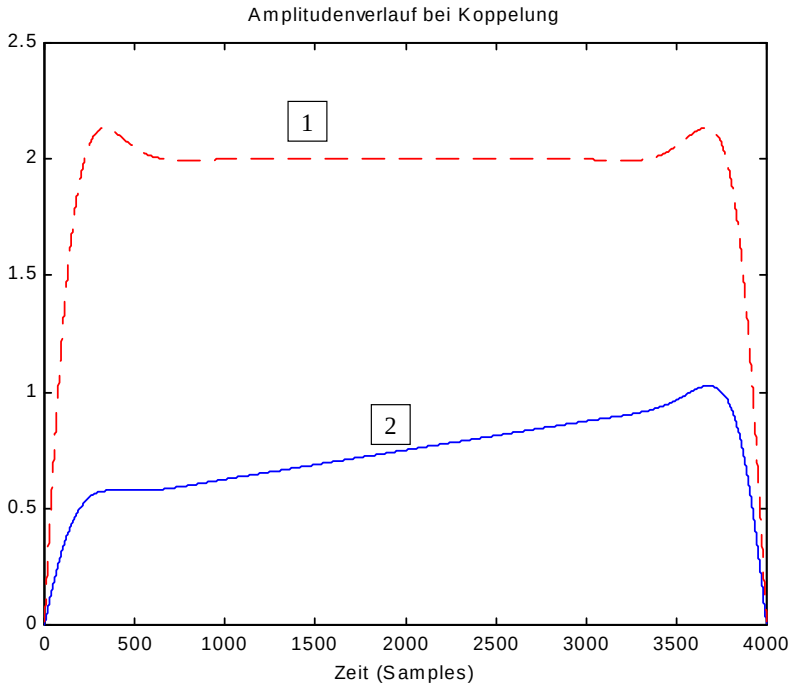


Abb. 2.5c: Ergebnis mit Koppelung

Diese Eigenschaft der Unterdrückung der Schwebungseffekte ist bei manchen Problemstellungen wünschenswert, so daß man auch gerne den Mehrbedarf an Rechenzeit beim Vold-Kalman-Filters in Kauf nimmt. Bei Aufgaben, bei denen die Teiltöne nicht nahe beieinander liegen oder sich nicht kreuzen, gelangt man mit herkömmlichen Heterodyne-Filtern zu vergleichbar guten Ergebnissen und benötigt jedoch viel weniger Rechenzeit.

2.2 Das QUASAR Signalmodell

Yinong Ding und Xiaoshu Qian stellten 1996 das sogenannte „Quadratic Polynomial-Phase Sinusoid and Residual Signal Model“, kurz QUASAR, vor (siehe [Ding, Qian; 1996] und [Ding, Qian; 1997]). Genau wie bisher wird versucht, zuerst die deterministische Signalkomponente zu bestimmen und anschließend das Residuum zu bilden und zu modellieren (siehe hierzu Kapitel 3). Im Abschnitt 1.2.2.2 wurde für die Momentanphase die kubische Interpolation nach Mc Aulay und Quatieri verwendet, beim QUASAR Signalmodell wird die Momentanphase frameweise als quadratisches Polynom beschrieben.

Wir schreiben die deterministische Komponente wieder als

$$x_{det} = \sum_{k=1}^{K} A_k(n) \cdot \cos\left(\varphi_k(n)\right) \tag{2.59}$$

Für die Momentanphase des k-ten Teiltones gilt nun

$$\varphi_k(n) = \theta_k n + \phi_k(n) \tag{2.60}$$

wobei θ_k die (stationäre) Nominalfrequenz und $\phi_k(n)$ die Phasenabweichung ist. Erstere wird aus einem Periodogramm des Originalsignals ermittelt (auch die Anzahl der Teiltöne K kann daraus abgelesen werden).
Die Phasenabweichung wird als stückweise quadratisches Polynom betrachtet:

$$\phi_k^i(n) = a_k^i + b_k^i n + c_k^i n^2 \tag{2.61}$$

wobei $\phi_k^i(n)$ der i-te Frame von $\phi_k(n)$ ist.

Ein stückweise quadratisches Polynom kann als Linearkombination von quadratischen Basis-Splines (kurz B-Splines) geschrieben werden:

$$\phi_k(n) = \sum_{i=-2}^{I-1} \beta_k^i \Theta^i(n) \tag{2.62}$$

Für die Amplitude reicht eine lineare Interpolation zwischen den Frames. Diese stückweise lineare Funktion kann man ebenfalls als Überlagerung von B-Splines schreiben, allerdings von linearen B-Splines.

$$A_k(n) = \sum_{i=0}^{I} \alpha_k^i \Lambda^i(n) \tag{2.63}$$

Die Abbildung 2.6 illustriert diese Basis-Spline-Funktionen.

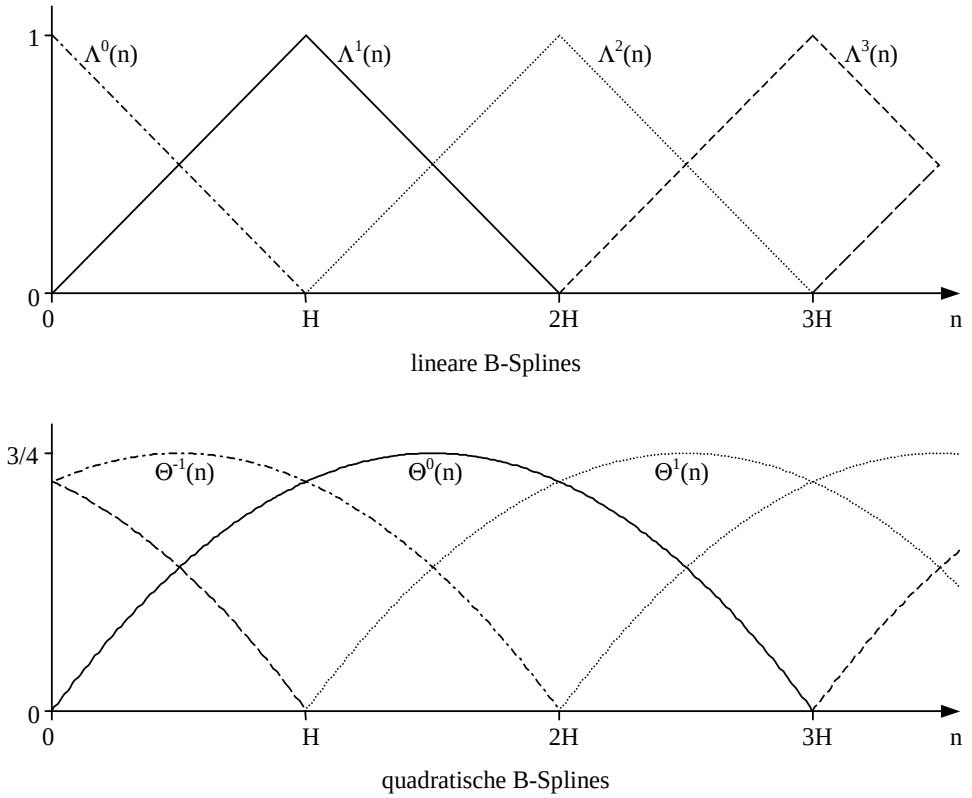


Abb. 2.6: lineare und quadratische Basis-Spline-Funktionen

Durch die B-Spline-Repräsentation wird garantiert, daß die Momentanphase und die Momentanamplitude stetig über Framegrenzen verlaufen.

Der Vorteil der quadratischen Phase ist, daß weniger Daten gespeichert werden müssen: anstelle von Momentanfrequenz und Momentanphase wird pro Frame nun nur mehr der Koeffizient β_k^i benötigt.

Der mit dem QUASAR-Modell verknüpfte Analysealgorithmus ist eine „analysis-by-synthesis“ Methode. D.h. die zu bestimmenden Parameter α_k^i und β_k^i werden in jedem Schritt eines Optimierverfahrens verändert. Das Optimier-Kriterium ist die Minimierung der Energie des Residuums.

$$\min_{\alpha_k^i,\beta_k^i} \sum_{n=0}^{N-1} \left(x(n) - x'_{det}(n)\right)^2 \qquad (2.64)$$

$x'_{det}(n)$ ist das Ergebnis der Resynthese mit den Parametern der vorhergegangenen Schätzung. Die Initialisierungswerte für α_k^i und β_k^i erhält man durch Verwenden von Heterodyne-Filtern, die auf den Nominalfrequenzen θ_k angesetzt werden und anschließendes B-Spline-Fitting.

Leider wurde von Ding und Qian nicht näher auf die Optimieraufgabe eingegangen, sie erwähnten lediglich, daß ein Gauss-Newton-Algorithmus verwendet werden könne. Weiters meinten die Autoren, daß die Optimierung oft ganz verzichtbar sei, da die Ergebnisse der Heterodyne-Filter meist bereits ausreichend gut wären. Allerdings dürfte dann keine Rede mehr von „analysis-by-synthesis“ sein.

Kapitel 3

Bestimmung und Modellierung der stochastischen Komponente

3.1 Bildung des Residuums

Nachdem der deterministische Anteil eines Signales mittels Methoden, die in den letzten beiden Kapiteln beschrieben wurden, ermittelt worden ist, soll nun die stochastische Komponente bestimmt werden.

Beschrieben wird der deterministische Teil durch die Parameter des Signalmodells. Nach anschließender Resynthese (z.B. mittels Additiver Synthese) liegt die Signalkomponente im Zeitbereich vor. Da das ursprüngliche Signal die Summe aus deterministischem und stochastischem Anteil ist ($x(n) = x_{det}(n) + x_{sto}(n)$) liegt die Überlegung, eine Subtraktion durchzuführen, nicht fern.

3.1.1 Subtraktion im Zeitbereich

Wenn $x_{det}(n)$ ausreichend genaue Phasenkohärenz zu x(n) besitzt, wird die Subtraktion von $x_{det}(n)$ und x(n) zielführend sein.

$$res(n) = x(n) - x_{det}(n) \tag{3.1}$$

Eine notwendige Bedingung für Phasenkohärenz ist Gleichung (3.2):

$$\sum_n |res(n)|^2 < \sum_n |x(n)|^2 \tag{3.2}$$

D.h. die Differenz muß eine geringere Energie haben als das ursprüngliche Signal.

Die Differenz (oder das Residuum) enthält höchstwahrscheinlich noch immer deterministische Anteile, denn die deterministische Komponente $x_{det}(n)$ kann meist nicht vollständig bestimmt werden. Grund dafür sind Fehler, wie zum Beispiel eine zu geringe Anzahl an Teiltönen, ein fehlerbehafteter Verlauf der Grundfrequenz, Einfluß von Transienten oder Interpolationsfehler.
Halten sich diese Fehler in Grenzen und besitzt res(n) deutlich weniger Energie als x(n), so kann man das Residuum durchaus als die stochastische Komponente betrachten.

$$x_{sto}(n) = res(n) \tag{3.3}$$

Bei den Methoden, die in Kapitel 2 beschrieben wurden, wird versucht, das Residuum zu minimieren. Hier ist also Phasenkohärenz garantiert gegeben, und die Bildung des Residuums nach Gleichung (3.1) wird zu einem vernünftigen Ergebnis führen.

Wird jedoch zum Beispiel eine Additive Synthese ohne Auswertung der Phaseninformation zur Resynthese verwendet, so ist die nötige Phasenkohärenz nicht gegeben und eine Subtraktion im Zeitbereich versagt.

3.1.2 Subtraktion im Frequenzbereich

Wenn die erforderliche Phasenkohärenz fehlt, kann eine Subtraktion nur im Frequenzbereich und nur im Amplitudenspektrum erfolgen (siehe [Serra, Smith; 1990]).

Zuerst werden x(n) und $x_{det}(n)$ mittels Short-Time Fourier Transform in den Frequenzbereich gebracht:

$$\underline{X}_i(m) = STFT\{x(n)\} \tag{3.4}$$

$$\underline{D}_i(m) = STFT\{x_{det}(n)\} \tag{3.5}$$

Es sollte eine Fensterfunktion gewählt werden, die sich verschoben überlagert zu einer Konstanten aufsummiert, z.B. ein von Hann-Fenster (siehe 1.2.3, Gleichung (1.56)).

Anschließend erfolgt die Subtraktion, Frame für Frame, Bin für Bin.

$$\left|\underline{R}_i(m)\right| = \left|\left|\underline{X}_i(m)\right| - \left|\underline{D}_i(m)\right|\right| \tag{3.6}$$

$$\arg(\underline{R}_i(m)) = \arg(\underline{X}_i(m)) \tag{3.7}$$

Das Phasenspektrum des Residuums ist das unveränderte Phasenspektrum des Originalsignals.

Zuletzt wird das Residuum in den Zeitbereich zurücktranformiert:

$$res(n) = \frac{1}{W_{oa}} \cdot iSTFT\{\underline{R}_i(m)\} \tag{3.8}$$

W_{oa} korrigiert den Einfluß des Fensters.

Bei idealen Bedingungen ist theoretisch nur ein Analysefenster und kein Synthesefenster[♣] erforderlich. Versuche haben jedoch gezeigt, daß auf ein Synthesefenster nicht verzichtet werden kann, da unter realen Umständen sonst ein "Knistern" oder "Prasseln" hörbar wird.

Fenstert man also bei der Analyse und bei der Resynthese mit der gleichen Funktion, so wird das Signal insgesamt zweimal mit dieser multipliziert, die effektive Fensterfunktion ist nun das Quadrat der ursprünglichen. Deshalb ist dann von Bedeutung, daß das Quadrat der Fensterfunktion sich verschoben überlagert zu einer Konstanten aufsummiert.
Ist w(n) ein Fenster, das die Gleichung (1.56) erfüllt (z.B. Hanning), dann verwenden wir zur Analyse und zur Resynthese die Wurzel aus w(n).

$$w_A(n) = w_S(n) = \sqrt{w(n)} \tag{3.9}$$

Ist w(n) ein Hann-Fenster, dann ist $w_A(n)$ und $w_S(n)$ eine Sinushalbwelle. Der Korrekturfaktor ist dann:

$$W_{oa} = \frac{N}{2 \cdot H} \quad \text{mit} \quad \frac{N}{H} \in \{2, 3, 4, ...\} \tag{3.10}$$

Möchte man zur Analyse und zur Resynthese unterschiedliche Fenster verwenden, so wird die Bedingung in Gleichung (1.56) ersetzt durch:

$$W_{oa}(n) = \sum_{m=-\infty}^{\infty} w_A(n - m \cdot H) \cdot w_S(n - m \cdot H) = \text{konst.} = W_{oa} \tag{3.11}$$

3.2 Modellierung und Resynthese der stochastischen Komponente

Die stochastische Komponente (bzw. das Residuum) liegt nun als Signal im Zeitbereich vor. Ein stochastisches Signal ist vollständig durch ein Leistungsdichtespektrum (LDS) beschreibbar, daher genügt auch ein Amplitudenspektrum. Nimmt man an, daß keine deterministischen Anteile mehr im Residuum enthalten sind, so läßt sich dieses Signal als gefiltertes weißes Rauschen interpretieren, wobei das Filter natürlich zeitvariant ist. Die stochastische Komponente ist also zeitvariant gefärbtes Rauschen.

[♣] kein Fenster meint natürlich kein anderes als das Rechteckenster

3.2.1 FFT-Filter mit Rauscherregung

Wie bereits erwähnt, beinhaltet das Amplitudenspektrum die gesamte Information, also transformieren wir die stochastische Komponente mittels Kurzzeit-Fouriertransformation in den Frequenzbereich. Wurde das Residuum durch Subtraktion im Frequenzbereich gebildet, so erübrigt sich dieser Schritt natürlich, da $\underline{R}_i(m)$ bereits vorliegt.

$$\underline{R}_i(m) = \mathrm{STFT}\{x_{sto}(n)\} \tag{3.12}$$

Da $x_{sto}(n)$ ein reellwertiges Signal ist, genügt es, pro Frame nur die Hälfte des Amplitudenspektrums zu speichern, da die zweite Hälfte wegen gerader Symmetrie redundant ist.

$$R_i(m) = \left|\underline{R}_i(m)\right| , \quad \begin{array}{l} m = 0, 1, 2, ..., N/2 \text{ wenn N gerade} \\ m = 0, 1, 2, ..., (N-1)/2 \text{ wenn N ungerade} \end{array} \tag{3.13}$$

$R_i(m)$ benötigt allerdings noch immer eine große Menge an Speicherplatz, weswegen eine weitere Datenreduktion erfolgen sollte. Serra und Smith empfehlen eine einfache Liniensegment-Approximation. Dazu wird das Spektrum in gleichgroße Frequenzbänder unterteilt und jeweils nur der maximale Amplitudenwert gespeichert. Zur Rekonstruktion wird zwischen diesen Werten, die in der Bandmitte plaziert sind, linear interpoliert.
Ein großer Nachteil dieser Methode ist, daß das rekonstruierte Signal im Allgemeinen mehr Energie besitzt, als das ursprüngliche. Mehr zu dieser Variante ist zu finden in [Serra, Smith; 1990] und [Serra; 1989].

Eine Verbesserung kann erzielt werden, wenn anstelle des Maximalwertes eines Bandes die Wurzel des Energiemittelwertes verwendet wird.

$$\hat{R}_i(q) = \sqrt{\frac{1}{U} \cdot \sum_{u=0}^{U-1} R_i(u + q \cdot U)^2} , \quad q = 0, 1, ..., Q-1 \tag{3.14}$$

Q ist die Anzahl der Frequenzbänder, U die Anzahl der Bins innerhalb eines Bandes.
Zur Rekonstruktion kann dann auch auf die lineare Interpolation verzichtet werden, es genügt, diesen RMS-Wert für alle Bins eines Bandes zu verwenden.

$$R'_i(m) = \hat{R}_i\left(\frac{m}{\overline{\overline{U}}}\right) \tag{3.15}$$

$\frac{m}{\overline{\overline{U}}}$ steht hier wieder für eine Integer-Division.

Denkbar ist auch eine Anpassung der Bandbreiten an die Frequenzgruppen des menschlichen Gehörs, z.B. 1 oder ½ Bark breite Bänder.

Eine weitere Speicherplatzeinsparung kann erreicht werden, wenn die Werte $\hat{R}_i(q)$ als Datentypen, die nur wenige Bits benötigen, gespeichert werden. Dazu logarithmieren wir zuerst $\hat{R}_i(q)$, um z.B. auf dB umzurechnen:

$$L_i(q) = 20 \cdot \log\left(\frac{\hat{R}_i(q)}{M_i}\right) \tag{3.16}$$

$$M_i = \max(\hat{R}_i(q)) \tag{3.17}$$

$L_i(q)$ kann nur Werte kleiner gleich Null annehmen und wird im Allgemeinen mit Gleitkommazahlen dargestellt. Eine mögliche Umrechnung in B-Bit-Integer ist in (3.18) angeführt, dabei werden Werte, die mehr als um Δ dB unter M_i liegen, nicht mehr berücksichtigt:

$$\tilde{L}_i(q) = \text{round}\left(\frac{2^B - 1}{\Delta} \cdot \text{clip}_{<0}\left(L_i(q) + \Delta\right)\right) \tag{3.18}$$

$$\text{clip}_{<L}(x) = \begin{cases} L, & x < L \\ x, & \text{sonst} \end{cases} \tag{3.19}$$

Bei B = 8 Bit und Δ = 64 dB erhält man eine Quantisierung von ¼ dB. Das sollte mehr als genügend sein. Dagegen ist bei B = 4 Bit und einem dürftigen Dynamikbereich von Δ = 32 dB nur mehr eine Quantisierung von 2 dB möglich.

Die Rekonstruktion geschieht nach (3.20) und (3.21):

$$L_i'(q) = \tilde{L}_i(q) \cdot \frac{\Delta}{2^B - 1} - \Delta \tag{3.20}$$

$$\hat{R}_i'(q) = \begin{cases} 0 & , \quad \tilde{L}_i(q) = 0 \\ M_i \cdot 10^{\frac{L_i'(q)}{20}} & , \quad \text{sonst} \end{cases} \tag{3.21}$$

Wie hier ersichtlich ist, muß natürlich auch der Skalierungswert M_i für jeden Frame gespeichert werden, um die Werte rekonstruieren zu können.

Zur Resynthese wird wieder eine inverse Kurzeit-Fouriertransformation verwendet, vorher müssen jedoch noch die komplexen Kurzzeitspektren berechnet werden. Wie schon erwähnt, betrachten wir die stochastische Komponente als zeitvariant gefiltertes weißes Rauschen. Das Phasenspektrum von weißem Rauschen besteht aus gleichverteilten♠ Zufallszahlen im Intervall $[-\pi, \pi[$, die Filterung bzw. Färbung ergibt sich durch Verwendung der gespeicherten Amplitudenspektren. Um Rechenzeit zu sparen, berechnen wir nur Spektren des analytischen Signals.

♠ im Unterschied zum Signal im Zeitbereich, welches aus normalverteilten Zufallszahlen besteht

$$\underline{R}_i'(m) = \begin{cases} R_i'(m) \cdot e^{j\cdot(\pi - \mathrm{rand}(2\pi))} & , 0 \le m \le N/2 \text{ (bzw. } (N-1)/2) \\ 0 & , N/2 \text{ (bzw. } (N-1)/2) < m < N \end{cases} \tag{3.22}$$

$$x_{sto}'(n) = \frac{2 \cdot W_{A,sq}}{W_{oa,sq}} \cdot \mathrm{Re}\left\{ \mathrm{iSTFT}\left\{ \underline{R}_i'(m) \right\} \right\} \tag{3.23}$$

Der Faktor 2 in Gleichung (3.23) berücksichtigt, daß es sich um ein analytisches Signal handelt. Aus dem selben Grund ist auch vom Ergebnis der iSTFT der Realteil zu nehmen.
$W_{A,sq}$ korrigiert den Einfluß des Analysefensters auf die Amplitude und berechnet sich nach (3.24).

$$W_{A,sq} = \sqrt{\frac{N}{\sum_{n=0}^{N-1} w_A(n)^2}} \tag{3.24}$$

Die sich überlappenden Signalblöcke, die durch Rücktransformation der Kurzzeitspektren entstehen, sind nicht kohärent. D.h. es addieren sich die Energien und nicht die Amplituden. Deshalb erscheint in Gleichung (3.24) der quadratische Mittelwert im Unterschied zu Gleichung (1.25), wo der arithmetische verwendet wird.
Aus dem gleichen Grund muß bei der iSTFT in (3.23) unbedingt ein Synthesefenster verwendet werden, das sich quadratisch zu einer Konstanten aufsummiert, also folgende Bedingung erfüllt:

$$W_{oa,sq}(n) = \sum_{m=-\infty}^{\infty} w_S(n - m \cdot H)^2 = \text{konst.} = W_{oa,sq} \tag{3.25}$$

Verwendet man für $w_S(n)$ eine Sinushalbwelle, so berechnet sich der Korrekturfaktor $W_{oa,sq}$ nach:

$$W_{oa,sq} = \sqrt{\frac{N}{2 \cdot H}} \quad \text{mit} \quad \frac{N}{H} \in \{2, 3, 4, \ldots\} \tag{3.26}$$

Vergleicht man (3.26) mit (3.10), so fällt als Unterschied natürlich die Wurzel auf. Die Erklärung dafür wurde bereits oben gegeben.

Leicht durchführbare Modifikationen sind Time Scale Modifications. Um das Signal zeitlich zu dehnen bzw. zu stauchen, muß bei der iSTFT (overlap and add) lediglich eine andere Hopsize verwendet werden, als bei der STFT. Bezeichnen wir mit S die Hopsize der iSTFT und mit H die der STFT. ξ_t ist der Faktor, um den das Signal gedehnt bzw. gestaucht werden soll.

$$S = \mathrm{round}(\xi_t \cdot H) \tag{3.27}$$

S muß natürlich, wie auch H, eine Ganze Zahl sein, somit ist der effektive Skalierungsfaktor nicht beliebig.

$$\xi_{t,eff} = S/H \tag{3.28}$$

Der Korrekturfaktor $W_{oa,sq}$ berechnet sich jetzt mit S anstelle von H.

$$W_{oa,sq} = \sqrt{\frac{N}{2 \cdot S}} \tag{3.29}$$

Im Allgemeinen summieren sich die überlagerten Fenster jetzt nicht mehr zu einer Konstanten, weshalb eine Amplitudenmodulation auftreten wird. Somit sind einer zeitlichen Dehnung Grenzen gesetzt. Ist z.B. S größer als N, so findet gar keine Überlappung mehr statt, das Ergebnis wird dann ein zerhacktes Signal sein.

Eine Möglichkeit, dieses Problem zu umgehen, ist eine Interpolation der Repräsentationsdaten. Dadurch wird dann S wieder gleich H. In Abbildung 3.1 wird diese Interpolation veranschaulicht. Es soll das Signal in diesem Beispiel auf das 0.8-fache gestaucht werden. Im obersten Graphen sind die überlappenden Analysefenster zu sehen, darunter die Zeitpunkte, für welche die Parameter eines Frames gelten und in der dritten Zeile die Zeitpunkte, für die die interpolierten Werte gültig sein sollen.

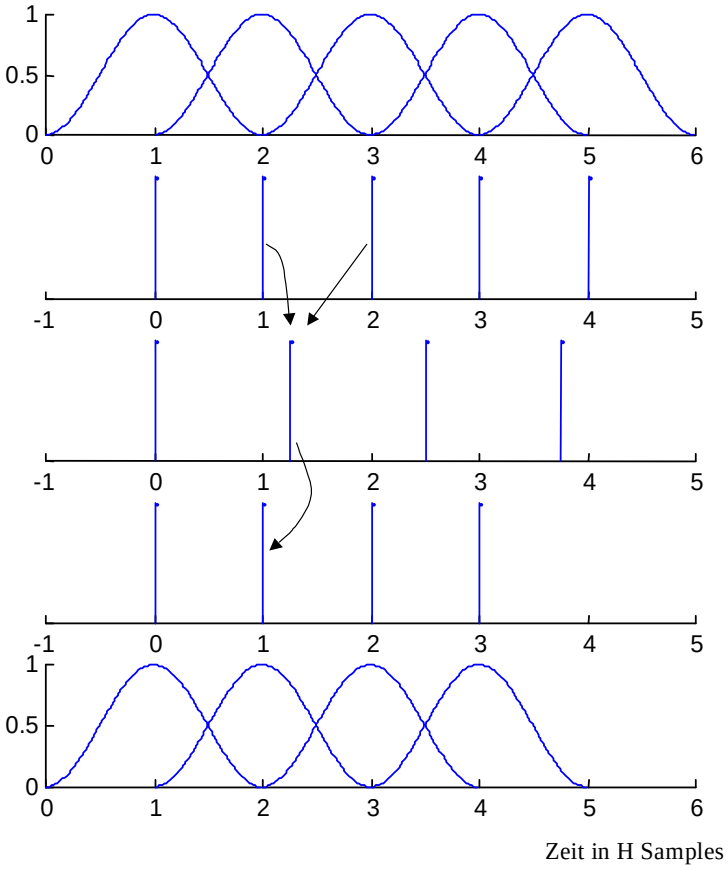


Abb. 3.1: Veranschaulichung einer Time Scale Modification mittels Interpolation der Repräsentation ($\xi_t = 0.8$)

Der interpolierte Parametersatz mit dem Index j gilt für den Zeitpunkt

$$t(j) = \frac{j}{\xi_t} \quad \text{(in Frames)} \tag{3.30}$$

Bezeichnen wir die ursprüngliche Repräsentation mit $R_i(q)$ und die interpolierte mit $R^t_j(q)$. Eine lineare Interpolation reicht aus:

$$R^t_j(q) = R_u(q) + (R_{u+1}(q) - R_u(q)) \cdot \left(\frac{j}{\xi_t} - u \right) \quad \text{mit} \quad u = \text{trunc}\left(\frac{j}{\xi_t} \right) \tag{3.31}$$

3.2.2 LPC-Filter mit Rauscherregung

Mittels Linearer Prädiktion (siehe Anhang A) lassen sich die Filterkoeffizienten eines IIR-Polstellenfilter bestimmen. Man erhält also für jeden Frame eine Approximation des Amplitudenspektrums durch ein Polynom der Ordnung p. Wendet man LPC an der stochastischen Komponente an, so erhält man eine sehr kompakte Repräsentation. Zur Resynthese wird dieses zeitvariante Filter mit weißem Rauschen erregt.

Zur Realisierung des zeitvarianten Filters möchte ich auf zwei Methoden näher eingehen:

- Verwendung eines Lattice-Filters mit zeitvarianten, interpolierten Koeffizienten
- Verwendung von zeitinvarianten Filtern und Overlap-Add

Verwendung eines Lattice-Filters mit zeitvarianten, interpolierten Koeffizienten

Wie im Anhang A nachzulesen ist, können die Koeffizienten eines Lattice-Filters problemlos zwischen den einzelnen Frames interpoliert werden, ohne daß es zu Stabilitätsproblemen kommt, vorausgesetzt, die Koeffizienten bei den Framegrenzen ergeben stabile Filter.

Als Eingangssignal verwenden wir weißes Rauschen, d.h. normalverteilte Zufallszahlen mit Mittelwert gleich Null und Varianz gleich Eins.

$$exc_i(n) = \text{randn}(n), \; n = 0, 1, 2, ..., S\text{-}1 \tag{3.32}$$

Die Synthese-Framelänge (hier gleich der Synthese-Hopsize) bezeichnen wir mit S.

$$y_i(n) = \text{lattice}\left(exc_i(n), G_i(n), k_{1,i}(n), k_{2,i}(n), ..., k_{p,i}(n)\right) \tag{3.33}$$

Zu beachten ist, daß die Zustandsvariablen des Filters von Frame i bei $n = S - 1$ auf Frame $i + 1$ bei n = 0 übernommen werden.
Für die Filterkoeffizienten ist eine lineare Interpolation völlig ausreichend:

$$G_i(n) = G_i + \frac{G_{i+1} - G_i}{S} \cdot n \tag{3.34}$$

$$k_{v,i}(n) = k_{v,i} + \frac{k_{v,i+1} - k_{v,i}}{S} \cdot n \tag{3.35}$$

Zuletzt müssen nur mehr die Blöcke aneinander gereiht werden:

$$y(n) = W_{A,sq} \cdot y_{\underline{\underline{n}}}\left(\mathrm{mod}(n,S)\right) \tag{3.36}$$

$W_{A,sq}$ korrigiert wieder den Einfluß des Analysefensters (siehe Gleichung (3.24)). Durch diese Korrektur sollte nun das Ergebnis der Resynthese y(n) die gleiche Leistung besitzen wie das ursprüngliche Signal $x_{sto}(n)$.

Möchte man eine Time Scale Modification durchführen, muß für S nur ein anderer Wert gewählt werden als für H. Das Problem einer unerwünschten Amplitudenmodulation tritt hier nicht auf.

Verwendung von zeitinvarianten Filtern und Overlap-Add

Die zeitvariante Filterung mit interpolierten Koeffizienten ist ziemlich rechenaufwendig. Deshalb möchte ich nun eine weitere Möglichkeit beschreiben, die weniger Rechenzeit erfordert.
Anstelle die Koeffizienten zu interpolieren, verwenden wir nun für jeden Frame ein zeitinvariantes Filter, multiplizieren die entstehenden Signalblöcke mit einem Synthesefenster und überlagern sie anschließend auf die gleiche Weise wie bei einer iSTFT.

$$exc_i(n) = \mathrm{randn}(n),\ n = 0, 1, 2, ..., M\text{-}1 \tag{3.37}$$

M ist die Syntheseframelänge.

$$y_i(n) = w_S(n) \cdot \mathrm{filter}\left(exc_i(n), G_i, k_{1,i}, k_{2,i}, ..., k_{p,i}\right) \tag{3.38}$$

$$y(n) = \frac{W_{A,sq}}{W_{oa,sq}} \cdot \sum_{i=0}^{I-1} \mathrm{Shift}^{g}_{i \cdot S,n,M}\left[y_i(g)\right] \tag{3.39}$$

Diese Gleichung beschreibt das sogenannte "Overlap and Add". S ist die Synthese-Hopsize, der Operator Shift wurde bereits in Kapitel 1.2.3 definiert.

Für die Korrekturfaktoren $W_{A,sq}$ und $W_{oa,sq}$ gelten die selben Überlegungen wie in Kapitel 3.2.1 (siehe Gleichungen (3.24) und (3.29)).

Auch hier sind Time Scale Modifications leicht realisierbar. Man wählt lediglich für S einen Wert ungleich H. Damit nun aber das Problem einer Amplitudenmodulation nicht auftritt, muß M abhängig von S gewählt werden. Denn im Unterschied zu einem STFT–iSTFT–System kann hier die Syntheseframelänge M ungleich der Analyseframelänge N sein (siehe Gleichung (3.40)). Es ist dann auch eine Interpolation überflüssig.
Verwenden wir zur Analyse z.B. ein Hann-Fenster und zur Synthese eine Sinushalbwelle, dann soll folgendes gelten:

$$\frac{N}{H} = \frac{M}{S} = k \in \{2, 3, 4, ...\} \tag{3.40}$$

3.2.3 LPC-Filter mit Multi-Puls-Erregung

In den letzten beiden Abschnitten wurde weißes Rauschen zur Erregung eines Filters verwendet. Wenn allerdings das zu modellierende Signal noch beachtliche deterministische und transiente Anteile besitzt, wird das Ergebnis mit einem sogenannten „noise-driven“ Filter wahrscheinlich nicht sehr zufriedenstellend sein. Bessere Ergebnisse können mit einer Multi-Puls-Erregung erzielt werden.
Die lineare Prädiktion mit Multi-Puls-Erregung (Multi Pulse Excitation Linear Prediction, MPLP) verwendet wieder ein Filter H(z), das mittels herkömmlicher linearer Prädiktion bestimmt wird, allerdings folgt ein weiterer Algorithmus zur Bestimmung einer optimalen Pulsfolge als Eingangssignal für das Filter. Wie nun zu erkennen ist, sind bei dieser Art der Kodierung also auch Daten für das Erregungssignal notwendig.

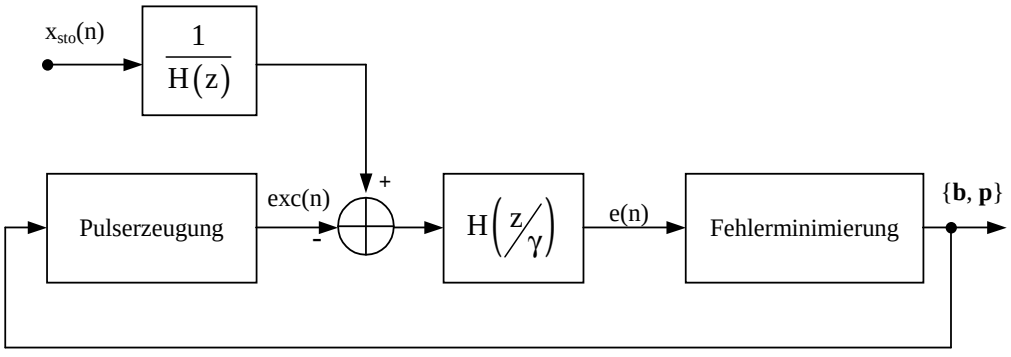


Abb. 3.2: Blockschaltbild des MPLP-Analysealgorithmus

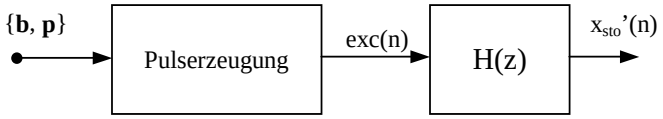


Abb. 3.3: Blockschaltbild der MPLP Resynthese

Abbildung 3.2 zeigt den MPLP-Analysealgorithmus, der ein optimales Erregungssignal exc(n) iterativ berechnet, d.h. es handelt sich um eine „analysis-by-synthesis“ Methode. Im Vektor **p** sind die Positionen der Pulse innerhalb eines Frames gespeichert (Sample-Indizes), **b** beinhaltet die Amplitude dieser Pulse. {**b**, **p**} soll so bestimmt werden, daß der Fehler, der die Differenz aus Erregung und invers gefiltertem Eingangssignal ist, minimal wird. $H(z/\gamma)$ wirkt als „error-shaping filter“. Eine genauere Beschreibung dieses Algorithmus ist zu finden in [Ding, Qian; 1996], wo MPLP zur Modellierung des Residuums beim QUASAR-Signalmodell vorgeschlagen wird.

Wie bereits erwähnt, liefert MPLP bessere Ergebnisse als ein „noise-driven“ Filter. Im Speziellen, wenn man bei der Resynthese die deterministische und die stochastische Komponente wieder zusammenfügt, verschmelzen diese besser, d.h. sie klingen nicht einzeln und somit synthetisch steril, sondern stehen in Bezug zueinander.
Allerdings sind auch Nachteile zu nennen: Modifikationen wie Timestretching, Pitchshifting oder Interpolation zwischen verschiedenen Sounds (Morphing) sind nicht mehr einfach zu bewerkstelligen und es müssen zusätzlich Daten für die Erregung gespeichert werden.

Kapitel 4

Analyse-/Resynthesesytem für KFZ-Innengeräusche

4.1 Alternatives Heterodyne-Filter

Die deterministische Komponente bei Kfz-Innengeräuschen besteht zum größten Teil aus dem Motorgeräusch und kann somit als ein harmonisches Signal betrachtet werden. Ist der zeitliche Verlauf der Grundfrequenz bekannt, z.B. aus einer Drehzahlreferenz berechnet, so lassen sich die einzelnen Teiltöne einfach mittels Heterodyne-Filter bestimmen. Abbildung 4.1 zeigt das Prinzip eines solchen Filters: heruntermischen und tiefpaßfiltern. $h_{LP}(n)$ ist die Impulsantwort des Tiefpasses, $\hat{\varphi}_k(n)$ ist die Schätzung der Momentanphase des k-ten Teiltones, die nach Gleichung (4.2) berechnet wird, und $A_k(n)$ ist die komplexe Amplitude oder Einhüllende. Interpretieren wir einen Partialton als modulierten Träger, so stellt ein Heterodyne-Filter den Demodulator dar.
Kapitel 2 beschäftigt sich ausführlich mit einem solchen Filter, dem Vold-Kalman-Filter, das die großartige Eigenschaft besitzt, durch gleichzeitige Verfolgung der einzelnen Teiltöne (Koppelung) Schwebungseffekte zu unterdrücken, d.h. die Energie wird „gerecht“ zwischen den einzelnen Partialtönen aufgeteilt. Dieser Vorteil kommt allerdings nur dann zum Tragen, wenn die einzelnen Frequenztrajektorien nahe beieinander liegen oder sich kreuzen. Bei Geräuschen von Verbrennungsmotoren gibt es keine Kreuzungen, und die Teiltöne liegen nur bei sehr niedriger Drehzahl nahe beieinander. Der große Nachteil des Vold-Kalman-Filters ist der Rechenzeitbedarf, weshalb ich in diesem Abschnitt ein alternatives Hetrodyne-Filter beschreiben werde.

Das folgende Beispiel soll demonstrieren, daß auch in extremen Situationen (niedrige Drehzahl) herkömmliche Filter ausreichend sind.

Beispiel: die Drehzahl eines 4-Zylinder-/4-Takt-Motors bei Standgas sei 900 U/min. Dies entspricht einer Frequenz von 15 Hz für die erste Motorordnung. Da die Teiltöne als ½-Ordnungen auftreten, ist der Abstand 7.5 Hz. Zweifache Anwendung eines IIR-Filters der Ordnung 5 mit Chebyshev-Charakteristik liefert bei einer auf 3 kHz reduzierten Abtastrate ein völlig akzeptables Ergebnis (Toleranzschema des Chebyshev-Tiefpasses der Ordnung 5: Paßband: -1.5 dB bis 2.4 Hz, Sperrband: -30 dB ab 3.6 Hz). Durch die zweifache Anwendung verdoppeln sich die Dämpfungswerte und man erhält für das Heterodyne-Bandpaßfilter eine –3 dB-Bandbreite von 4.8 Hz und eine –60 dB-Bandbreite von 7.2 Hz.

Der Vorteil eines IIR-Filters ist der geringe Rechenzeitbedarf, der Nachteil die nichtlineare Phase. Da unser Analysevorgang nicht in Echtzeit zu geschehen hat, können wir akausale Methoden anwenden. D.h. die Daten liegen bereits vollständig vor und wir betreiben „post processing“. Dadurch ist es möglich, ein nullphasiges IIR-Filter zu erhalten.

Nullphasiges Forwärts/Rückwärts-IIR-Filter

Um ein nullphasiges IIR-Filter zu bekommen, wendet man ein herkömmliches IIR-Filter zuerst vorwärts und anschließend rückwärts an. In einer MATLAB-ähnlichen Notation läßt sich das einfach ausdrücken:

$$\mathbf{y} = \mathrm{flip}\left(\mathrm{filter}\left(\mathrm{flip}\left(\mathrm{filter}(\mathbf{x})\right)\right)\right) \tag{4.1}$$

Durch die zweimalige Filterung ergeben sich für die Dämpfung doppelt so hohe Werte als in den Spezifikationen des Filters angegeben (Kaskade, Verdoppelung der Ordnung).

Natürlich braucht auch dieses Filter eine gewisse Einschwingzeit, weshalb in den Randbereichen Reserven vorhanden sein sollten.

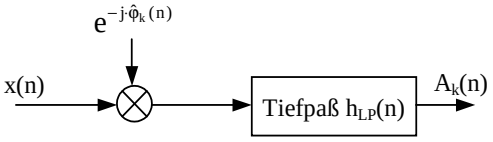


Abb. 4.1: Heterodyne-Filter

$$\hat{\varphi}_k(n) = \frac{2\pi k}{f_S} \sum_{m=0}^{n-1} \hat{f}_1(m) \tag{4.2}$$

Möchte man ein Filter mit variabler Bandbreite, so empfielt sich die Realisierung des IIR-Filters mittels Lattice-Struktur (siehe hierzu 3.2.2 und A.1.1).

Die Wahl der Bandbreite ist ein sehr wichtiger Punkt. Einerseits soll sie so gering sein, daß ausschließlich die Energie des gewünschten Teiltones berücksichtigt wird und nicht die von benachbarten bzw. von Rauschen. Andererseits passiert es bei zu geringer Bandbreite leicht, daß schon bei kleinen Fehlern in der Schätzung der Momentanfrequenz der Teilton vorübergehend „verloren geht“, d.h. er liegt für kurze Zeit im Sperrband des Filters.

4.2 Bestimmung des Grundfrequenzverlaufes

Um ein Heterodyne-Filter einsetzen zu können, sind Schätzungen der Momentanfrequenzverläufe der einzelnen Teiltöne erforderlich. Bei einem harmonischen Signal genügt eine Schätzung der Grundfrequenz. Da wir uns mit Geräuschen von Verbrennungsmotoren beschäftigen, ist es naheliegend, eine eventuell vorhandene Drehzahlreferenz zu verwenden.

4.2.1 Verwendung einer Drehzahlreferenz

Sollten es die Umstände ermöglichen, zusätzlich zum KFZ-Geräusch das Zündsignal aufzuzeichnen, so sind nur noch kleine Nachbearbeitungsschritte erforderlich, um eine Schätzung der Fundamentalfrequenz zu erhalten. Da ich im Rahmen meiner Arbeit jedoch leider keine Gelegenheit hatte, Referenzsiganle zur Hilfe zu nehmen, möchte ich auf [Vold, Leuridan; 1993] und [Vold, Mains, Blough; 1997] verweisen.

4.2.2 Bestimmung ohne Drehzahlreferenz

Wenn keine Drehzahlreferenz zur Verfügung steht, muß mit aufwendigeren Methoden versucht werden, den Grundfrequenzverlauf mehr oder weniger automatisiert zu ermitteln. Abbildung 4.2 zeigt den Absolutbetrag einer STFT eines Hochlaufgeräusches, das im Innenraum des Fahrzeuges (am Beifahrersitz auf Kopfhöhe) aufgenommen wurde. Um die Darstellung zu verbessern, wurden Grauwerttransformationen durchgeführt (siehe hierzu [Sonka, Hlavac, Boyle; 1999]). Der Motordrehzahlverlauf steigt hier von etwa 960 U/min auf ungefähr 6480 U/min monoton an, was einer Frequenz von 16 Hz bis 108 Hz entspricht. Diese nach Gleichung (4.3) berechnete Frequenztrajektorie bezeichnen wir als erste Motorordnung. In der Abbildung ist sehr deutlich die zweite Motorordnung sehen, die von 32 Hz bis 216 Hz ansteigt. Es handelt sich bei diesem Beispiel um ein Kraftfahrzeug mit einem 4-Takt-Ottomotor mit 4 Zylindern. Dieser Typ von Motor produziert zwei Explosionen pro Umdrehung, was die Dominanz dieser zweiten Ordnung erklärt. Andererseits benötigt ein vollständiger Zyklus zwei volle Umdrehungen (ein einzeln betrachteter Zylinder wird nur jede zweite Umdrehung gezündet). Dies erklärt die Existenz von halben Ordnungen. Bei diesem Motortyp entspricht die Frequenz der Ordnung ½ der Grundfrequenz im Sinne von Fourier (Gleichung (4.4)).

$$f_{(1.\,Ord)}(n) = \frac{RPM(n)}{60} \tag{4.3}$$

$$f_1(n) = \tfrac{1}{2} f_{(1.\,Ord)}(n) = \frac{RPM(n)}{120} \qquad \text{, gilt für 4-Takt-/4-Zylinder-Motor} \qquad (4.4)$$

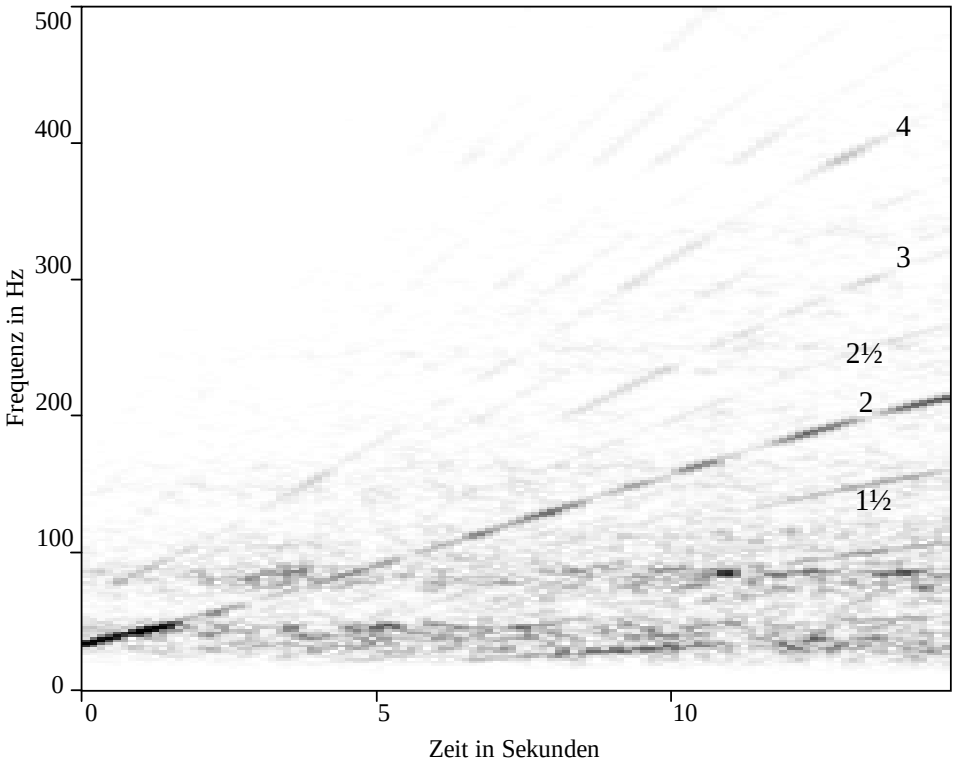


Abb. 4.2: Betrag der STFT eines KFZ-Innengeräusches bei einem 100%-Hochlauf im 2. Gang (zur besseren Darstellung wurden Grauwert-transformationen durchgeführt)

Da bei diesem Motortyp die zweite Motorordnung am stärksten ausgeprägt ist, sollte der Frequenzverlauf dieser bestimmt und anschließend durch vier geteilt werden. Es können auch mehrere Trajektorien verfolgt werden, je nach Qualität der einzelnen. In diesem Fall kann man durch Mittelung auf die Grundfrequenz schließen. In Gleichung (4.5) geschieht eine durch die Amplitude der Teiltöne gewichtete Mittelung.

$$\hat{f}_1(n) = \sum_{k=1}^{K} \frac{f_k(n)}{k} \frac{A_k(n)}{\sum_{k=1}^{K} A_k(n)} \qquad (4.5)$$

Es sind hier nur die wirklich dominierenden Teiltöne von Interesse, z.B. bis zur 6. Motorordnung, weshalb die Abtastrate des Geräuschsignals stark reduziert werden sollte, um Rechenzeit zu sparen (z.B. auf 2 kHz). Wie aus Abbildung 4.2 erkennbar ist, gewinnen höhere Ordnungen mit steigender Drehzahl immer mehr an Bedeutung. Bei niedriger Drehzahl ist hier nur die 2. und 4. Motorordnung zu sehen.

Rauschen erschwert die Analyse, weshalb sich die Trajektorien womöglich nur stückweise verfolgen lassen. Bei sehr tiefen Frequenzen sieht man in Abbildung 4.2 durchgehende Rauschbänder, das sogenannte „Booming“. Der Grund für siese Bänder sind markante

Resonanzen des Innenraumes. Man kann in dieser Abbildung deutlich sehen, daß die Amplitude der zweiten Motorordnung in diesen Frequenzbereichen besonders groß ist.

Wendet man die von Serra beschriebene „Peak Detection“ (siehe 1.1.2.2 und [Serra; 1989]) bei diesem KFZ-Innengeräusch an, so erhält man den in Abbildung 4.3 dargestellten Peak Plot. Auch nach längerem Probieren findet man keine Grenzwerte, die zu einem wirklich zufriedenstellenden Ergebnis führen, es werden entweder zu viele spektrale Spitzen detektiert, oder es verschwinden auch jene, die eigentlich von Interesse sind. Eine Verbesserung kann mit Hilfe von Nachbearbeitungsschritten aus der Bildverarbeitung erzielt werden, wie zum Beispiel das Entfernen von vereinzelt auftretenden Punkten, die höchstwahrscheinlich nicht zu Sinuskomponenten gehören, oder das Vervollständigen von Linien (Line Relaxation). Hierzu möchte ich auf [Sonka, Hlavac, Boyle; 1999] verweisen.

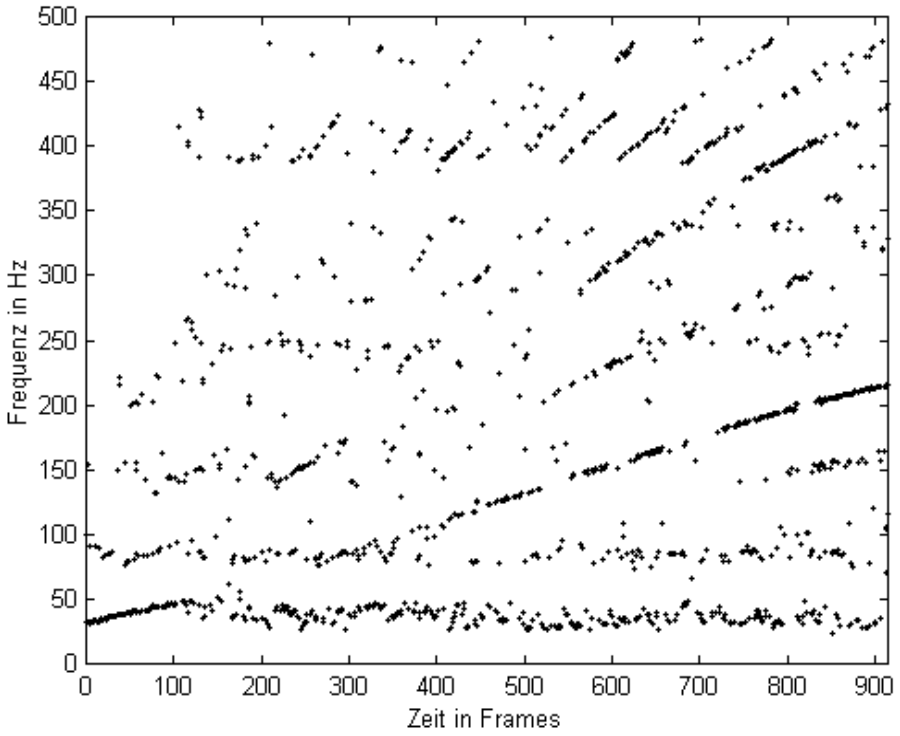


Abb. 4.3: Ergebnis der Peak Detection nach 1.1.2.2.
Verwendete Werte: f_s = 1 kHz, N = 128, H = 16, W = 127, Fensterfunktion: Kaiser (β = 5.5, B_W = 4 bin), a_{MIN} = -35 dB, h_{MIN} = 4, p_{MAX} = 0.5 rad

Eine vollständig automatisierte Detektion des Grundfrequenzverlaufes ist nur mit aufwendigen Algorithmen und viel Rechenaufwand erreichbar. Bei einfacheren Methoden sind unterstützende manuelle Schritte erforderlich. Es kann zum Beispiel Anfangs- und Endwert des Drehzahlverlaufs bekanntgegeben werden, oder es können Bereiche oder Liniensegmente manuell selektiert werden.

Läßt sich der Verlauf nicht vollständig extrahieren, so sollten diese Lücken z.B. mittels kubischer Spline-Interpolation gefüllt werden. Weiters ist eine Interpolation erforderlich, um $\hat{f}_1(n)$ bei der Samplerate zu bekommen, bei der das Heterodyne-Filter eingesetzt werden soll.

4.3 Analysesystem

In diesem Abschnitt möchte ich ein komplette System zur Analyse von KFZ-Geräuschen beschreiben. Wir versuchen, das ursprüngliche Signal x(n), das bei einer Samplerate f_S von angenommen 48 kHz vorliegt, in eine deterministische und eine stochastische Komponente zu zerlegen. Diese beiden Teile werden anschließend mit unterschiedlichen Methoden parametrisiert.
Für die deterministische Komponente treffen wir die Vereinfachung, daß sie ausschließlich durch den Verbrennungsmotor verursacht wurde, weshalb von einem harmonischen Signal ausgegangen werden kann und somit der Verlauf der Grundfrequenz zur Analyse ausreicht. Alle Signalinhalte, die nicht vom Motor produziert werden, weisen wir der stochastischen Komponente zu. Für die Trennung wird zuerst der deterministische Anteil bestimmt und anschließend das Residuum durch Subtraktion im Zeitbereich berechnet.
Abbildung 4.4 zeigt das Blockschaltbild des Analysesystems. Der erste Schritt ist eine Reduzierung der Abtastrate auf beispielsweise $f_{S,dec} = 6$ kHz, da die Motorordnungen nur bis etwa 1.5 kHz bis 2 kHz unverdeckt bleiben. Dadurch kann die Rechenzeit der folgenden Schritte gering gehalten werden. Zur Sampleratenkonversion empfiehlt sich eine mehrstufige Anwendung einer Reduktion um jeweils ½ oder ⅓ mit einem FIR-Filter, dessen Delay kompensiert wird.

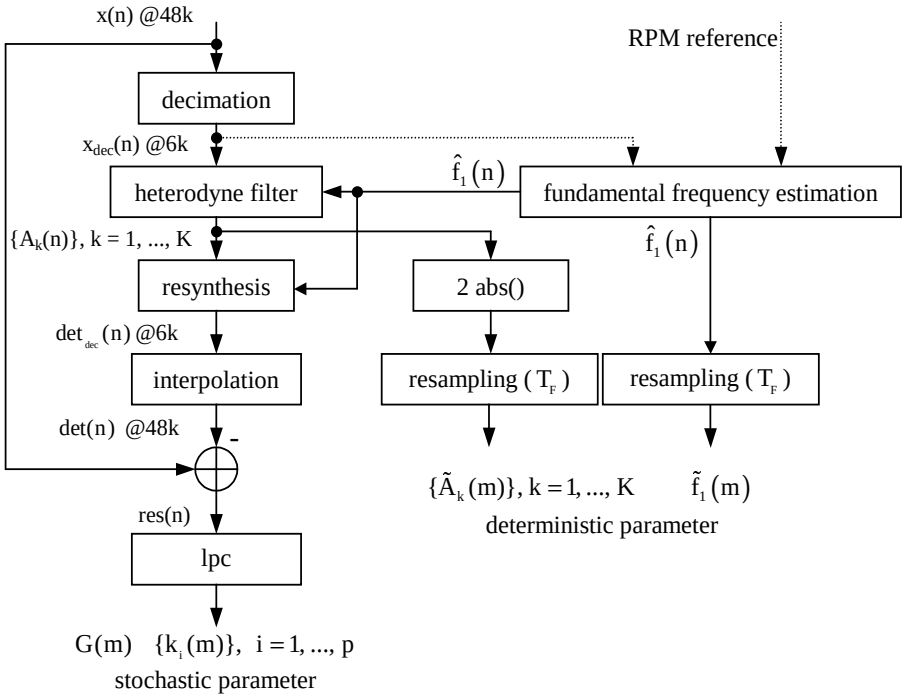


Abb. 4.4: Blockschaltbild des Analysesystems

Zur Bestimmung der deterministischen Parameter wird das in Abschnitt 4.1 beschriebene Heterodyne-Filter verwendet. Zuvor muß allerdings noch eine Schätzung des Grundfrequenzverlaufes $\hat{f}_1(n)$ berechnet werden (siehe 4.2), damit die Momentanphase $\hat{\varphi}_k(n)$ nach Gleichung (4.5) bestimmt werden kann.
Das Heterodyne-Filter liefert die komplexen Amplituden $A_k(n)$ für die Teiltöne k = 1, ..., K (siehe Gleichung (4.6)) bei einer Samplerate von $f_{S,dec}$. Die Anzahl an Partialtönen K ist mit einem Wert von 20 bis 30 ausreichend.

$$\hat{\varphi}_k(n) = \frac{2\pi k}{f_{S,dec}} \sum_{m=0}^{n-1} \hat{f}_1(m) \tag{4.5}$$

$$A_k(n) = \left(x_{dec}(n) \cdot e^{-j\,\hat{\varphi}_k(n)} \right) \otimes h_{LP}(n) \tag{4.6}$$

Das Resynthesesystem (4.4) soll möglichst wenig Rechenaufwand erfordern, deshalb wird auch nur der Absolutbetrag von $A_k(n)$ verwendet. Weiters, da die Bandbreite des Tiefpasses mit der Impulsantwort $h_{LP}(n)$ nur wenige Hz beträgt, ist die Einhüllende $A_k(n)$ sehr langsam veränderlich und kann mit einem enorm großen Abtastintervall abgetastet werden. Dieses Intervall wird als Framelänge T_F bezeichnet und sollte in der Größenordnung von etwa 10 ms liegen (anders ausgedrückt sollte die Framerate $f_F = \frac{1}{T_F}$ ungefähr 100 Hz betragen).

$$H_{dec} = \frac{f_{S,dec}}{f_F} \in \mathbb{N} \tag{4.7}$$

Die erforderliche Hopsize (Framelänge in Samples) muß natürlich eine Ganze Zahl ergeben. Wir wählen für $H_{dec} = 64$ bei $f_{S,dec} = 6\,\text{kHz}$, was eine Framerate von 93.75 Hz ergibt.
Als Parameter der deterministischen Komponente speichern wir also eine stark abtastratenreduzierte Version des Betrages von $A_k(n)$:

$$\tilde{A}_k(n) = \left| A_k(n \cdot H_{dec}) \right| \tag{4.8}$$

Nicht nur die Amplituden sind Teil dieser Parameter, sondern auch der Grundfrequenzverlauf. Dieser wird ebenfalls auf f_F reduziert:

$$\tilde{f}_1(n) = \hat{f}_1(n \cdot H_{dec}) \tag{4.9}$$

Um das Residuum berechnen zu können, muß der deterministische Anteil zuerst resynthetisiert werden. Dazu verwenden wir das Ergebnis der Heterodyne-Filter, also die komplexwertigen, nicht dezimierten $A_k(n)$ und den Grundfrequenzverlauf $\hat{f}_1(n)$, um die Momentanphasen $\hat{\varphi}_k(n)$ zu erhalten. Letztere wurden allerdings bereits in den Heterodyne-Filtern verwendet. Gleichung (4.10) beschreibt diese komplexwertige Additive Synthese, wir nehmen den zweifachen Realteil des analytischen Signals.

$$\mathrm{det}_{dec}(n) = 2 \cdot \mathrm{real}\left(\sum_{k=1}^{K} A_k(n)\, e^{j \hat{\varphi}_k(n)} \right) \tag{4.10}$$

Der nächste Schritt ist die Interpolation von $f_{S,dec}$ auf f_S. Dazu sollte wie bei der Reduktion die Konversion um kleine Faktoren (2 oder 3) in mehreren Stufen durchgeführt werden. Da nun $\mathrm{det}'(n)$ und $x(n)$ mit der gleichen Samplerate vorliegen, kann die Subtraktion erfolgen, und man erhält das Residuum $\mathrm{res}(n)$:

$$\mathrm{res}(n) = x(n) - \mathrm{det}(n) \tag{4.11}$$

Modelliert wird das Residuum mittels LPC (siehe Kapitel 3 und Anhang A). Man verwendet wieder die gleiche Framerate f_F wie bei den deterministischen Parametern, womit wir eine Hopsize H von 512 Samples bei f_S = 48 kHz erhalten. Das Ergebnis von LPC sind Filterkoeffizienten. Es ist für das Resynthesesystem von Vorteil, wenn diese bereits als Reflexionskoeffizienten k_i mit i = 1, ..., p vorliegen. Die Ordnung des Filters p soll mindestens 20 sein.

4.4 Resynthesesystem

Zur Resynthese der deterministischen Komponente verwenden wir eine einfache Additive Synthese. Als Parameter liegen nur reellwertige Amplituden vor, d.h. wir beziehen keine Phaseninformation zur Interpolation ein – eine linieare Interpolation sowohl für Momentanamplituden als auch für Momentanphasen ist ausreichend. In Abschnitt 1.2.2.1 wurde diese einfache und rechenzeitsparende Methode bereits beschrieben.

Momentanamplituden und Momentanfrequenz berechnen sich nach

$$A_k'(n) = \tilde{A}_k\left(\frac{n}{=}{H}\right) + \frac{\tilde{A}_k\left(\frac{n}{=}{H}+1\right) - \tilde{A}_k\left(\frac{n}{=}{H}\right)}{H} \cdot \left(n - \left(\frac{n}{=}{H} \cdot H \right) \right) \tag{4.12}$$

und

$$f_1'(n) = \tilde{f}_1\left(\frac{n}{=}{H}\right) + \frac{\tilde{f}_1\left(\frac{n}{=}{H}+1\right) - \tilde{f}_1\left(\frac{n}{=}{H}\right)}{H} \cdot \left(n - \left(\frac{n}{=}{H} \cdot H \right) \right) \tag{4.13}$$

Die Hopsize H ist hier wie bei der Modellierung des Residuums zu wählen. Bei den vorgeschlagenen Werten beträgt H = 512 Samples bei f_S = 48 kHz.

Anzumerken ist hier noch, daß das Resynthesesystem eine Latenz von zumindest einer Framelänge T_F besitzt, da zur Interpolation innerhalb eines Frames (m) bereits die Werte des nächsten Frames (m+1) bekannt sein müssen.
Die Momentanphasen werden wieder nach der bereits mehrmals vorgestellten Gleichung errechnet:

$$\varphi'_k(n) = \frac{2\pi k}{f_S} \sum_{m=0}^{n-1} f'_1(m) \qquad (4.14)$$

Die Zeitsignale der einzelnen Teiltöne können nun berechnet werden, da Betrag und Phase bekannt sind. Die Summe aller ergibt das deterministische Signal $x'_{det}(n)$.

$$x'_{det}(n) = \sum_{k=1}^{K} A'_k(n) \cos\left(\varphi'_k(n)\right) \qquad (4.15)$$

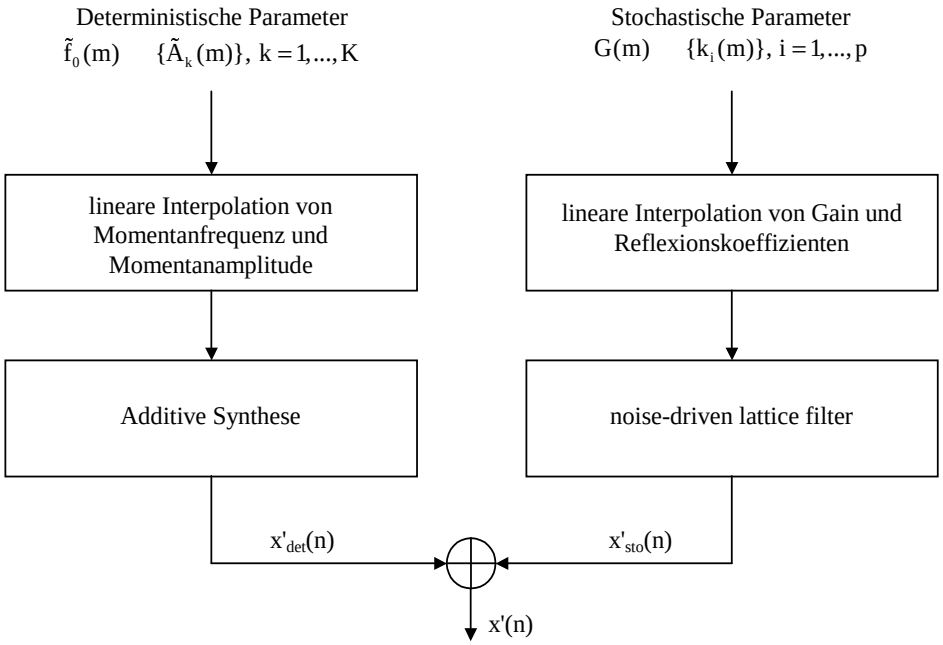


Abb. 4.5: Blockschaltbild des Resynthesesystems

Zur Resynthese der stochastischen Komponente verwenden wir ein mit weißem Rauschen gespeistes zeitvariantes Lattice-Filter. Dazu müssen die Reflexionskoeffizienten $k_i(m)$ und Gain G(m) linear interpoliert werden. Diese Methode wurde bereits in Abschnitt 3.2.2 detailliert beschrieben. Der Ausgang dieses Filters liefert das stochastische Signal $x'_{sto}(n)$, die Summe von $x'_{det}(n)$ und $x'_{sto}(n)$ ergibt das vollständige resynthetisierte KFZ-Innengeräusch.

$$x'(n) = x'_{det}(n) + x'_{sto}(n) \qquad (4.16)$$

Das Blockschaltbild in Abbildung 4.5 veranschaulicht das in diesem Abschnitt Beschriebene.

Kapitel 5

Echtzeit-Simulator für KFZ-Innengeräusche

Das Ziel dieser Arbeit ist die Entwicklung eines Simulators für KFZ-Innengeräusche. Eine Testperson, die Innengeräusche eines Kraftfahrzeuges bewerten soll, bekommt die Möglichkeit, nach Wunsch mehr oder weniger Gas zu geben. Der Simulator errechnet dann in Echtzeit das diesem Lastzustand, bzw. dieser Lastzustandsänderung entsprechende Geräusch möglichst naturgetreu.
Als Ausgangsmaterial dienen Aufnahmen bei verschiedenen Lastzuständen. Um nun die Geräusche bei unterschiedlichsten Lasten zu erhalten ist eine Interpolation erforderlich, wofür zuallererst ein einfaches Lastzustandsmodell gefunden werden muß.

5.1 Lastzustandsmodell

Zur Vereinfachung betrachten wir zuerst nur Lastzustände bei konstantem Gang. Weiters nehmen wir gleichbleibende Straßenverhältnisse und –steigung an. Auf eine sehr einfache Art und Weise läßt sich der Lastzustand eines Kraftfahrzeuges mit nur zwei Parametern beschreiben: Motordrehzahl v(t) und Gaspedalstellung p(t). Spannen wir mit diesen beiden eine Ebene (p-v-Ebene) auf, so kann der Lastzustand zu jedem Zeitpunkt t als Punkt dargestellt werden.
Bei einer bestimmten Gaspedalstellung erreicht das Fahrzeug (nach theoretisch unendlich langer Zeit) eine zugehörige bestimmte Geschwindigkeit, vorausgesetzt, das Triebwerk wird nicht in Grenzzustände gebracht. Das soll bedeuten, bei höheren Gängen wird diese Zuordnung von Gaspedalstellung und Stationärgeschwindigkeit (bzw. Stationärdrehzahl) normalerweise zutreffen, bei niedrigen Gängen nur bis zu einer gewissen Pedalstellung, da eine Maximaldrehzahl erreicht wird.
Abbildung 5.1 zeigt diese Zuordnung für ein fiktives Kraftfahrzeug bei einem niedrigen Gang (dick-punktierte Linie). Bezeichnen wir diese Linie als Stationärlinie $v_{ST}(p)$. Die Zuordnung gilt hier nur bis zu einer Gasstellung von etwa ¾, da dann bereits die Maximaldrehzahl erreicht ist.

Die Stationärlinie teilt diese einfache Lastzustandsebene in zwei Bereiche: Beschleunigungs- und Verzögerungsbereich, oder Zug- und Schubbereich. Folgendes Beispiel soll diese Zusammenhänge verdeutlichen: das Fahrzeug bewegt sich mit konstanter Geschwindigkeit bei 2220 RPM und 30 % Gas. Diesem Betriebszustand entspricht der Punkt A in der p-v-Ebene von Abbildung 5.1. Der Fahrer erhöht nun abrupt das Gas auf 47 %. Im ersten Moment ändert sich die Drehzahl noch nicht, wir befinden uns im Punkt B. Dieser Punkt liegt oberhalb der Stationärlinie und das Fahrzeug beschleunigt. Wird die neue Gaspedalstellung beibehalten, so erhöht sich die Drehzahl solange, bis wieder die Stationärlinie bei 3330 RPM erreicht wird (Punkt C). Bei einer anschließenden Gasverminderung auf die ursprünglichen 30 % geschieht ähnliches. Der Motor reduziert die Drehzahl ausgehend von Punkt D, bis wieder der stationäre Zustand A erreicht ist.

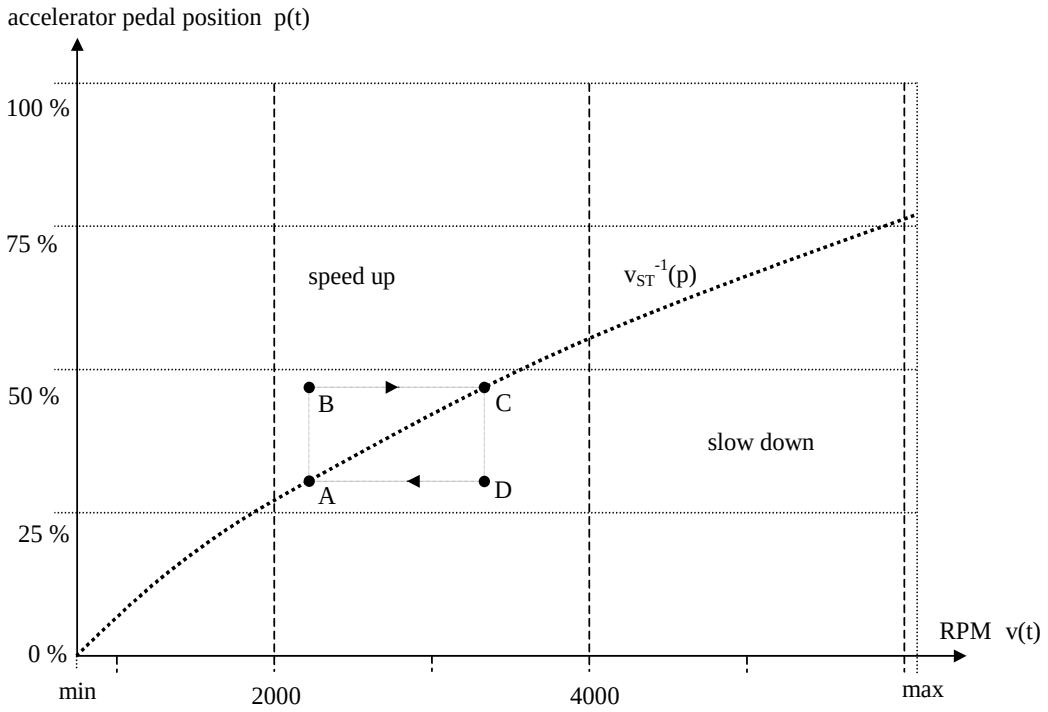


Abb. 5.1: Das Lastzustandsmodell: Drehzahl v und Gaspedalstellung p spannen die Lastzustandsebene auf

Wichtig ist, daß der Verlauf der Drehzahl immer eine stetige Funktion ergeben muß, sowohl als Funktion der Zeit als auch als Funktion der Gaspedalstellung. D.h. die Funktion muß differenzierbar sein: $\frac{dv(t)}{dt}$ und $\frac{dv(p)}{dp}$ existieren. Es kann in der Lastzustandsebene also keine horizontalen Sprünge geben.

5.2 Interpolation von KFZ-Innengeräuschen

Nehmen wir an, daß Innengeräusche für Beschleunigungs- und Verzögerungsvorgänge bei verschiedenen Gaspedalstellungen aufgenommen worden sind. Während einer Aufnahme bleibt die jeweilige Gasstellung konstant. Wir betrachten nun die Geräusche bei den beiden Pedalpositionen p_1 und p_2. Ziel ist, das KFZ-Geräusch für eine Gasstellung p zu berechnen, die zwischen p_1 und p_2 liegt. Abbildung 5.2 zeigt dieses Vorhaben in der p-v-Ebene, links soll ein Beschleunigungsgeräusch berechnet werden und rechts eines für Verzögerung.

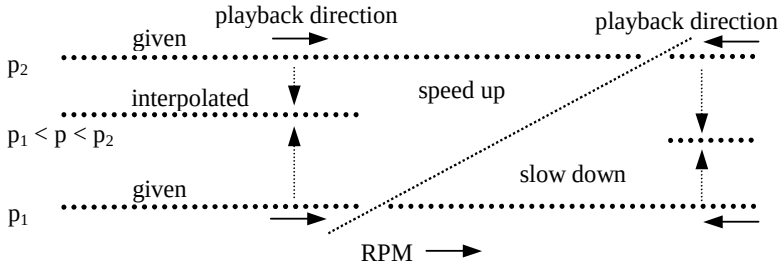


Abb. 5.2: Veranschaulichung der Interpolation in der Lastzustandsebene

Die Soundfiles werden mit dem in Abschnitt 4.3 beschriebenen Analysesystem modelliert, es liegen Frames mit den deterministischen und stochastischen Parametern vor. Zeichnet man diese Frames in die p-v-Ebene ein, so ergibt sich in der Nähe der Stationärlinie eine hohe Framedichte, da dort $\left|\frac{dv(t)}{dt}\right|$ sehr klein ist. Je weiter wir uns entfernen, umso geringer wird die Dichte, $\left|\frac{dv(t)}{dt}\right|$ wird immer größer. Punkte auf der Stationärlinie besitzen theoretisch eine unendlich hohe Framedichte.

Wir wollen nun die Signalmodellparameter für die Gaspedalstellung p durch Interpolation der Werte für p_1 und p_2 berechnen. Dazu werden Frames mit gleicher Drehzahl v bzw. mit gleicher Grundfrequenz $\tilde{f}_1(m)$ herangezogen. Es ist also für ein jedes gegebenes KFZ-Geräusch die Umkehrfunktion von $\tilde{f}_1(m)$ erforderlich, die wir als $m(f_1)$ bezeichnen, um die entsprechenden Frames finden zu können. Eine lineare Interpolation für jeden einzelnen Parameter ist ausreichend. Mit $w_p(m)$ in Gleichung (5.1) ist ein beliebiger Modellparameter gemeint, z.B. die Amplitude eines Teiltones oder ein Reflexionskoeffizient.

$$w_p\left(m_p(f_1)\right) = w_{p_1}\left(m_{p_1}\left(f_1\right)\right) + \frac{w_{p_2}\left(m_{p_2}\left(f_1\right)\right) - w_{p_1}\left(m_{p_1}\left(f_1\right)\right)}{p_2 - p_1} \cdot \left(p - p_1\right) \qquad (5.1)$$

Diese Interpolation läßt sich problemlos in Drehzahlbereichen durchführen, in denen beide gegebenen Sounds definiert sind. In der Nähe der Stationärlinie wird es allerdings problematisch: ab der Stationärdrehzahl bei p_1 ist nur mehr der Sound bei p_2 definiert. Abbildung 5.3 zeigt diesen Bereich, in dem Gleichung (5.1) nicht angewendet werden kann. Auf keinen Fall darf eine Interpolation zwischen einem Beschleunigungs- und einem Verzögerungsgeräusch gemacht werden.

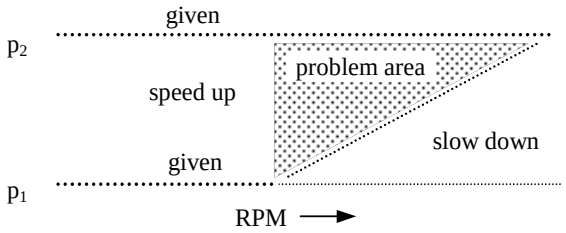


Abb. 5.3: Der Problembereich bei der Interpolation

Die einfachste Möglichkeit, dieses Problem zu lösen, ist die Verwendung der Frames im Stationärpunkt von p_1 auch bei höheren Drehzahlen. Das setzt natürlich voraus, daß die Aufnahmen auch noch einen Bereich mit gleichbleibender Geschwindigkeit besitzen.

Ein weiteres Problem stellt die Umkehrfunktion $m(f_1)$ dar. Die Originalfunktion $\tilde{f}_1(m)$ ist reellwertig, es treten Werte mit minimalen Unterschieden auf. Um $m(f_1)$ als Tabelle implementieren zu können, ist eine gröbere Quantisierung erforderlich, da die Speicherkapazität sonst nicht ausreicht. Dies führt jedoch zum Verlust von Frames. Man muß daher auf einen Suchalgorithmus zurückgreifen, der mit einer knapp gehaltenen Tabelle $\hat{m}(\ddot{f}_1)$ beschleunigt wird. Wir suchen nun also den Index m desjenigen Frames, der eine zum gewünschten Wert $\dot{f}_1$ minimal abweichende Grundfrequenz besitzt. Abbildung 5.4 zeigt den Algorithmus.

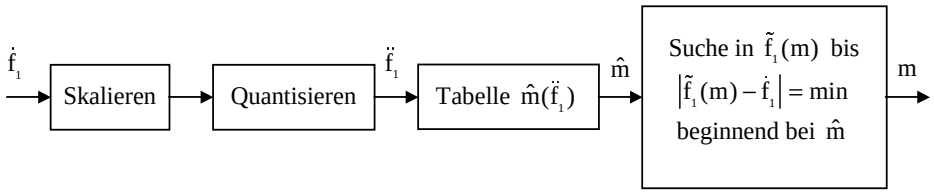


Abb. 5.4: Implementierung der Umkehrfunktion von $\tilde{f}_1(m)$

Die lineare Interpolation der Parameter von Frames mit gleicher Drehzahl liefert noch kein fertiges Soundfile. Es stellt sich die Frage, bei welchen Werten für $\dot{f}_1$ die Interpolation durchgeführt werden soll. Der zeitliche Abstand zwischen den Frames des zu berechnenden Geräuschs muß natürlich ebenso konstant sein wie auch bei den gegebenen. Es ist also der neue Grundfrequenzverlauf $\dot{f}_1(m)$ erforderlich. Bei der Gaspedalstellung p soll die Beschleunigung bzw. die Verzögerung andere Werte besitzen als bei p_1 oder p_2. Verwenden wir auch hierzu eine lineare Interpolation:

$$\dot{f}_1(m) = f_1^{p_1}(m) + \frac{f_1^{p_2}(m) - f_1^{p_1}(m)}{p_2 - p_1} \cdot (p - p_1) \qquad (5.2)$$

Zu Beachten ist, daß die beiden gegebenen Aufnahmen unterschiedlich lange dauern: bei mehr Gas wird die Stationärgeschwindigkeit später erreicht als bei weniger Gas. Die Folge $f_1^{p1}(m)$ muß daher im Stationärbereich verlängert werden.

Nachdem $\dot{f}_1(m)$ bestimmt worden ist, kann die Berechnung der restlichen Parameter nach Gleichung (5.1) erfolgen.

5.3 Algorithmus für den Simulator

Für den Simulator sind Beschleunigungs- und Verzögerungsgeräuschaufnahmen bei konstanter Gaspedalstellung erforderlich. Zum Beispiel bei 0 %, 20 %, 40%, 60 %, 80 % und 100 %.

Betrachten wir nun eine Realtime-Anwendung. Die Gaspedalstellung bleibt nicht konstant, sondern kann sich zu jedem Zeitpunkt verändern. Abbildung 5.5 zeigt das Struktogramm für den Algorithmus basierend auf dem Analyse- und Resynthesesystem von Kapitel 4. Ein jeder Durchlauf durch diese Schleife benötigt die Dauer eines Frames T_F. Wir beginnen mit einer Initialdrehzahl, zum Beispiel mit der Minimaldrehzahl. Das System muß jeden Zyklus die Pedalstellung erneut abfragen. Nachdem nun sowohl die Momentandrehzahl als auch die Momentanpedalstellung bekannt sind, können die beiden Soundfiles, die für eine Interpolation während dieses Frames herangezogen werden sollen, ermittelt werden. Es existieren für ein jedes Soundfile die entsprechenden Daten, um die Umkehrfunktion von $\tilde{f}_1(m)$ implementieren zu können. Somit können nun auch die für die Momentandrehzahl passenden Frames m_1 und m_2 der beiden gegebenen Geräusche gefunden werden. In diesem Schritt soll auch überprüft werden, ob der momentane Lastzustand in einem Problembereich liegt. Ist das der Fall, so soll ein entsprechender Frame aus dem Stationärbereich von p_1 für m_1 gewählt werden. Einer Interpolation der deterministischen und stochastischen Signalmodellparameter nach Gleichung (5.1) steht nichts mehr im Weg. Der nächste Schritt ist die Resynthese. Da wegen der linearen Interpolation zwei zeitlich benachbarte Frames erforderlich sind, um Signale bei voller Samplerate zu erhalten, kommt es zu einer

Verzögerung (Latenz) von einer Framedauer T_F. Deshalb kann im Initialdurchlauf noch keine Resynthese erfolgen. In allen weiteren Zyklen geschieht die Resynthese nach Abschnitt 4.4.
Zum Schluß muß noch die Momentandrehzahl für den nächsten Durchlauf berechnet werden. Verwenden wir dazu die auf m_1 bzw. m_2 nachfolgenden Frames der beiden gegebenen Sounds:

$$\dot{f}_1 = f_1^{p_1}(m_1+1) + \frac{f_1^{p_2}(m_2+1) - f_1^{p_1}(m_1+1)}{p_2 - p_1} \cdot (p - p_1) \qquad (5.3)$$

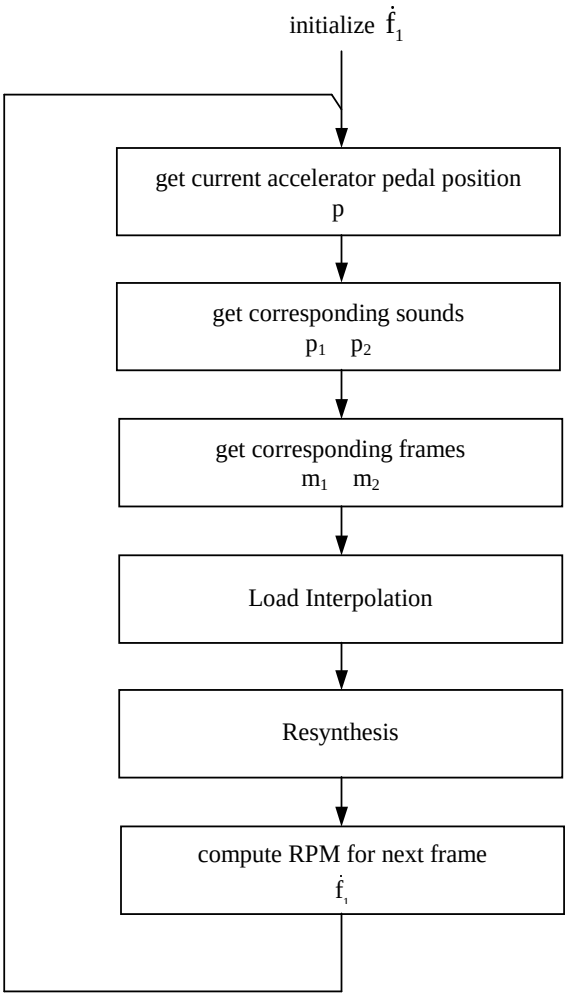


Abb. 5.5.: Struktogramm des Algorithmus
für den Echtzeit-KFZ-Geräusch-Simulator

Anhang A

Lineare Prädiktion (LPC)

Lineare Prädiktion spielt bei der Sprachanalyse eine sehr wichtige Rolle. Man geht dabei davon aus, daß ein Signal exc(n) durch ein zeitvariantes Filter geformt wird und das fertige Sprachsignal x(n) liefert. Das Eingangssignal, auch Erregung (Excitation) genannt, soll bei stimmhaften Lauten ein Pulszug mit entsprechender Frequenz sein, womit die Stimmbänder des Menschen nachgebildet wären. Bei stimmlosen Lauten ist die Erregung weißes Rauschen. Dieses Signal wird anschließend gefiltert, was beim Menschen im Vokaltrakt geschieht.

Durch die Lineare Prädiktion werden die Parameter des Filters bestimmt. Es ist somit möglich, mit geringen Datenmengen Sprache zu übertragen oder zu speichern (linear predictive coding – LPC), es müssen lediglich die Parameter des Filters und die Grundfrequenz der Erregung, falls es sich um einen stimmhaften Laut handelt, übermittelt werden. LPC stellt also eine kompakte Repräsentation für Sprachsignale dar, Modifikationen wie Pitch-Shifting oder Time-Stretching sind ohne Aufwand durchführbar.

LPC ist aber nicht nur für Sprache verwendbar, mit einem Pulszug als Erregung lassen sich beliebige harmonische Signale codieren. Mit weißem Rauschen kann zum Beispiel die stochastische Komponente eines Signals modelliert werden.

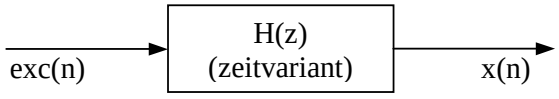


Abb. A.1: Modell für Lineare Prädiktion

Da es sich um zeitvariante Parameter handelt, unterteilen wir das Signal in kurze Segmente (Frames) und nehmen an, daß das System innerhalb eines Segments stationär ist. $y_i(n)$ ist das Segment mit dem Index i und der Länge N, H ist wieder die Hopsize:

$$x_i(n) = x(n + i \cdot H) \quad \text{für} \quad 0 \le n < N \qquad \text{(A.1)}$$

Aus Gründen der Vereinfachung wird von nun an der Frame-Index weggelassen, wir betrachten nur mehr ein bestimmtes Segment.

Nehmen wir an, daß das Filter sich durch folgende Übertragungsfunktion modellieren läßt [Vaseghi; 1996]:

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k \cdot z^{-k}} \qquad (A.2)$$

Es handelt sich hier um ein Filter, dessen Charakteristik nur von den Polstellen abhängt, die Nullstellen sind fix bei z = 0 (Allpol-Filter). Die Anzahl der Polstellen ist p. Möchte man ein Filter, das r Resonanzen besitzt, so muß p mindestens $2 \cdot r$ sein.

Die dazugehörige Differenzengleichung des Systems ist:

$$x(n) = \sum_{k=1}^{p} a_k \cdot x(n-k) + G \cdot exc(n) \qquad (A.3)$$

Man versucht nun, x'(n) aus den letzten p Samples vorherzusagen:

$$x'(n) = \sum_{k=1}^{p} \alpha_k \cdot x(n-k) \qquad (A.4)$$

Das setzt natürlich voraus, daß das Signal in sich korreliert.

Es ergibt sich jedoch ein Prädiktorfehler:

$$e(n) = x(n) - x'(n) = x(n) - \sum_{k=1}^{p} \alpha_k \cdot x(n-k) \qquad (A.5)$$

Im Idealfall, wenn man die Systemparameter exakt bestimmen kann, minimiert sich der Fehler zu:

$$e(n) = G \cdot exc(n) \quad \text{wenn} \quad \{\alpha_k\} = \{a_k\} \qquad (A.6)$$

Wir wollen nun versuchen, die Parameter $\{\alpha_k\}$ so zu bestimmen, daß der quadratische Fehler E minimal wird.

$$E = \sum_{n} |e(n)|^2 \Rightarrow \min. \qquad (A.7)$$

Dies erreicht man durch Null setzen der partiellen Ableitungen:

$$\frac{\partial E}{\partial \alpha_k} = 0, \qquad 1 \le k \le p \qquad (A.8)$$

Nun hat man ein Gleichungssystem mit p Gleichungen und p Unbekannten. Nach weiterer Berechnung (siehe [Vaseghi; 1996], [Haddad, Parsons; 1991]) nimmt dieses Gleichungssystem folgende Form an:

$$\begin{bmatrix} R(0) & R(1) & R(2) & ... & R(p-1) \\ R(1) & R(0) & R(1) & ... & R(p-2) \\ R(2) & R(1) & R(0) & ... & R(p-3) \\ ... & ... & ... & ... & ... \\ R(p-1) & R(p-2) & R(p-3) & ... & R(0) \end{bmatrix} \cdot \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ ... \\ \alpha_p \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \\ ... \\ R(p) \end{bmatrix} \qquad \text{(A.9)}$$

wobei R(n) eine "Kurzzeit-Autokorrelationsfolge" (Schätzung der Autokorrelationsfolge) ist:

$$R(n) = \sum_{m=0}^{H-1-n} x(m) \cdot x(m+n) \qquad \text{(A.10)}$$

Bestimmung von G (ohne Herleitung):

$$G = \sqrt{R(0) - \sum_{k=1}^{p} \alpha_k \cdot R(k)} \qquad \text{(A.11)}$$

Um mit dieser Autokorrelations-Methode ein gutes Ergebnis zu erzielen, empfiehlt es sich, die Signalausschnitte mit einem Fenster zu multiplizieren, das an den Enden ausblendet (also kein Rechteckfenster).

Das Gleichungssystem in (A.9) enthält eine reguläre Töplitz-Matrix, eine sehr effiziente Methode dieses Gleichungssystem zu lösen, ist die Levinson´sche Rekursion.

A.1 Die Levinson´sche Rekursion (Levinson-Durbin-Algorithmus)

Die Koeffizienten $\{\alpha_k^{(p)}\}$ eines Prädiktorfilters der Ordnung p können durch jene $\{\alpha_k^{(p-1)}\}$ des Filters der Ordnung p-1 ausgedrückt werden:

$$\begin{bmatrix} 1 \\ -\alpha_1^{(p)} \\ \vdots \\ -\alpha_{p-1}^{(p)} \\ -\alpha_p^{(p)} \end{bmatrix} = \begin{bmatrix} 1 \\ -\alpha_1^{(p-1)} \\ \vdots \\ -\alpha_{p-1}^{(p-1)} \\ 0 \end{bmatrix} + k_p \cdot \begin{bmatrix} 0 \\ -\alpha_{p-1}^{(p-1)} \\ \vdots \\ -\alpha_1^{(p-1)} \\ 1 \end{bmatrix} \tag{A.12}$$

Der Skalar k_p heißt Reflexionskoeffizient. Das kommt daher, daß der erste Vektor rechts vom Gleichheitszeichen in Gleichung (A.12) die Vorwärts-Prädiktorkoeffizienten und der zweite Vektor die Rückwärts-Prädiktorkoeffizienten enthält.

Ohne weiterer Herleitung (siehe [Vaseghi; 1996], [Haddad, Parsons; 1991]) möchte ich nun den fertigen Algorithmus vorstellen.

Levinson-Durbin-Algorithmus

Um nun das Gleichungssystem (A.9) recheneffizient zu lösen, geht man folgendermaßen vor:

$$E^{(0)} = R(0) \tag{A.13}$$

For i = 1, 2, ..., p

$$\Delta^{(i-1)} = R(i) - \sum \alpha_k^{(i-1)} \cdot R(i-k) \tag{A.14}$$

$$k_i = -\frac{\Delta^{(i-1)}}{E^{(i-1)}} \tag{A.15}$$

$$\alpha_i^{(i)} = k_i \tag{A.16}$$

For j = 1, ..., i-1

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \cdot \alpha_{i-j}^{(i-1)} \tag{A.17}$$

$$E^{(i)} = \left(1 - k_i^2\right) \cdot E^{(i-1)} \tag{A.18}$$

Mit den Filterkoeffizienten $\{\alpha_i\}$ kann ein Filter in Direkt-Form realisiert werden. Bei dieser Filtertopologie gibt es allerdings den Nachteil, daß es zu Stabilitätsproblemen kommen kann, wenn man versucht, die Koeffizienten von Frame zu Frame zu interpolieren. Für eine Aussage über Stabilität ist es erforderlich, das Nennerpolynom von H(z) zu faktorisieren, sodaß eine Überprüfung der Polradien möglich ist.
Bei einem Filter in Lattice-Form, bei dem die Reflexionskoeffizienten $\{k_i\}$ verwendet werden, gibt es dieses Problem nicht.

A.1.1 Allpol-Brückenfilter (All-pole Lattice Filter)

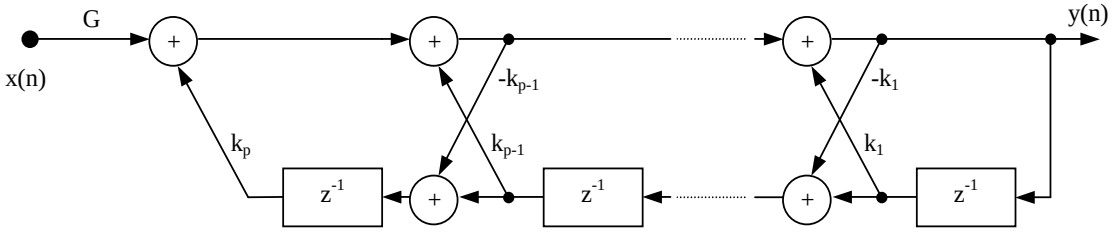


Abb. A.2: Allpol-Filter in Lattice-Form

Die Reflexionskoeffizienten sind ebenfalls ein Ergebnis des oben angeführten Levinson-Durbin-Algorithmus. Außerdem dient dieser Algorithmus zur Umrechnung von $\{\alpha_i\}$ in $\{k_i\}$.

Um bei einem Filter in Lattice-Form die Stabilität zu überprüfen, genügt es festzustellen, ob alle Reflexionskoeffizienten kleiner als eins sind.

$k_i < 1$ für i = 1, 2, ..., p $\Rightarrow$ Filter ist stabil (A.19)

Quellenverzeichnis

[Althoff, Keiler, Zölzer; 1999]
Rasmus Althoff, Florian Keiler, Udo Zölzer: "Extracting Sinusoids From Harmonic Signals", Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99), Dec. 1999

[Ding, Qian; 1996]
Yinong Ding, Xiaoshu Qian: "Processing of Musical Tones Using a Combined Quadratic Polynomial-Phase Sinusoid and Residual (QUASAR) Signal Model", Journal of the Audio Engineering Society, Vol. 45, No. 7/8, July/August 1997

[Ding, Qian; 1997]
Yinong Ding, Xiaoshu Qian: "Sinusoidal and Residual Decomposition and Residual Modeling of Musical Tones Using the QUASAR Signal Model", Proceedings ICMC, 1997

[Golub, Ortega; 1993]
Gene Golub, James M. Ortega: "Scientific Computing, Eine Einführung in das wissenschaftliche Rechnen und Parallele Numerik", Deutsche Übersetzung von Rolf Dieter Grigorieff, B. G. Teubner Stuttgart, 1996

[Haddad, Parsons; 1991]
Richard A. Haddad, Thomas W. Parsons: "DIGITAL SIGNAL PROCESSING – Theory, Applications, and Hardware", Computer Science Press, New York, 1991

[Kreyszig; 1999]
Erwin Kreyszig: "Advanced Engineering Mathematics", 8th Edition, John Wiley & Sons Inc., 1999

[McAulay, Quatieri; 1986 (1)]
Robert J. McAulay, Thomas F. Quatieri: "Speech Analysis/Synthesis Based on a Sinusoidal Representation", IEEE Transaction on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No. 4, August 1986

[McAulay, Quatieri; 1986 (2)]
Robert J. McAulay, Thomas F. Quatieri: "Speech Transformations Based on a Sinusoidal Representation", IEEE Transaction on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No. 6, December 1986

[Oppenheim, Schafer; 1975]
Alan V. Oppenheim, Ronald W. Schafer: "Digital Signal Processing", ISBN: 0-13-214635-5, Prentice Hall, Inc., New Jersey, USA, 1975

[Rodet, Depalle; 1992]
X. Rodet, P. Depalle: "Spectral Envelopes and Inverse FFT Synthesis", Presented at the 93rd AES Convention San Francisco, October 1992

[Sacchi, Ulrych, Walker; 1998]
Mauricio D. Sacchi, Tadeusz J. Ulrych, Colin J. Walker: "Interpolation and Extrapolation Using a High-Resolution Discrete Fourier Transform", IEEE Transactions on Signal Processing, Vol. 46, No. 1, January 1998

[Sayed, Kailath; 1994]
Ali H. Sayed, Thomas Kailath: "A State-Space Approach to Adaptive RLS Filtering", IEEE Signal Processing Magazine, July 1994

[Serra; 1989]
Xavier Serra: "A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition", Dissertation, Dep. of Music, Stanford University, October 1989

[Serra, Smith; 1990]
Xavier Serra, Julius Smith: "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition", Computer Music Journal, Vol. 14, No. 4, P. 12-24, Winter 1990

[Sonka, Hlavac, Boyle; 1999]
M. Sonka, V. Hlavac, R. Boyle: "Image Processing, Analysis, and Machine Vision", PWS Publishing, ISBN 0-534-95393-X, 1999

[Vaseghi; 1996]
Saeed V. Vaseghi: "Advanced Signal Processing and Digital Noise Reduction", Queen´s University of Belfast, UK, ISBN: 0-471-95875-1, John Wiley & Sons Ltd and B.G. Teubner, 1996

[Vold, Leuridan; 1993]
Havard Vold, Jan Leuridan: "High Resolution Order Tracking at Extreme Slew Rates, Using Kalman Tracking Filters", SAE Paper Number 931288, 1993

[Vold, Mains, Blough; 1997]
Havard Vold, Michael Mains, Jason Blough: "Theoretical Foundations for High Performance Order Tracking with the Vold-Kalman Tracking Filter", SAE Paper Number 972007, 1997

[Vold, Herlufsen, u.a.; 1999]
Havard Vold, Henrik Herlufsen, Svend Gade, H. Konstantin-Hansen: "Characteristics of the Vold-Kalman Order Tracking Filter", Sound and Vibration, April 1999