Darstellung von Signalen und Spektren

2

Inhaltsverzeichnis

Zur Verdeutlichung der Signalverarbeitung in einem *Software Defined Radio* (SDR) werden wir die relevanten Signale im Zeitbereich und im Frequenzbereich darstellen; wir sprechen dabei von *Signalen* (Zeitbereich) und *Spektren* (Frequenzbereich). Zur Darstellung verwenden wir das numerische Mathematikprogramm *Matlab* oder das kompatible Programm *GNU Octave*. Da in einem SDR kontinuierliche („analoge“) und diskrete („digitale“) Signale auftreten, in einem numerischen Mathematikprogramm aber nur diskrete Signale in Form von Vektoren mit Abtastwerten verarbeitet werden können, gehen wir in diesem Abschnitt zunächst auf die Darstellung der Signale und die Berechnung und Darstellung der Spektren ein.

2.1 Kontinuierliche und diskrete Signale

Als Beispiel betrachten wir das in Abb. 2.1 gezeigte kontinuierliche Signal

$$x(t) = \sin \omega t = \sin 2\pi f t$$

A. Heuberger und E. Gamm, *Software Defined Radio-Systeme für die Telemetrie*,
DOI 10.1007/978-3-662-53234-8_2

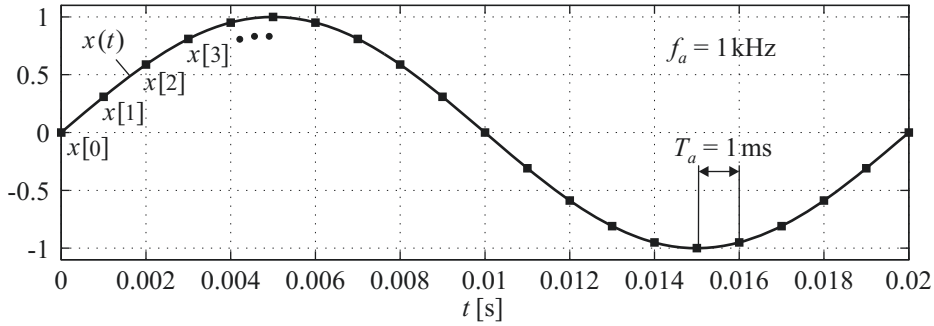

Abb. 2.1 Sinus-Signal als kontinuierliches und als diskretes Signal

mit $\omega = 2\pi f = 2\pi \cdot 50\,\text{Hz}$ und das daraus durch Abtastung mit der Abtastrate $f_a = 1\,\text{kHz}$ bzw. dem Abtastintervall $T_a = 1/f_a = 1\,\text{ms}$ hervorgehende diskrete Signal:

$$x[n] \;=\; x(nT_a) \;=\; \sin 2\pi f\, T_a n$$

In *Matlab*/*Octave* schreiben wir:

```
f_a = 1000;
t_a = 1 / f_a;
f = 50;
t = 0 : t_a : 0.02;
x = sin( 2 * pi * f * t );
```

Damit erhalten wir die Vektoren

$$t = [\ 0.000\ \ 0.001\ \ 0.002\ \ 0.003\ \ 0.004\ \ 0.005\ \ \ldots\ \ 0.020\ ]$$
$$x = [\ 0.000\ \ 0.309\ \ 0.588\ \ 0.809\ \ 0.951\ \ 1.000\ \ \ldots\ \ 0.000\ ]$$

der Länge 21, die den Punkten $x[n]$ des diskreten Signals in Abb. 2.1 entsprechen. Wenn wir das diskrete Signal nun mit dem Befehl

```
plot(t,x);
```

anzeigen, erhalten wir aber nicht die Punkte $x[n]$, sondern das kontinuierliche Signal $x(t)$. Das Programm zeigt also *mehr* an, als wir geliefert haben. Das liegt daran, dass wir dem Programm mit dem Befehl `plot` mitgeteilt haben, dass wir eine Kurve sehen möchten, für die wir aber nur Stützstellen, d. h. Abtastwerte, bereitstellen. Das Programm versucht in diesem Fall, die bereitgestellten Punkte zu einer *möglichst glatten* Kurve zu verbinden. Das funktioniert nur dann gut, wenn die Anzahl der Stützstellen ausreichend hoch ist. Wenn wir dagegen *nur* die Abtastwerte anzeigen wollen, müssen wir den Befehl

```
plot(t,x,'s');
```

verwenden. Die Darstellung in Abb. 2.1 erhalten wir demnach, indem wir beide Darstellungen kombinieren:

```
figure(1);
plot(t,x);
hold on;
plot(t,x,'s');
hold off;
grid on;
axis([ min(t) max(t) -1.1 1.1 ]);
```

Das Abtasttheorem für Tiefpass-Signale besagt, dass die Abtastrate f_a mindestens doppelt so großsein muss wie die maximale, im Signal vorhandene Frequenz f_{max}:

$$\boxed{f_a \geq 2f_{max}} \tag{2.1}$$

Für die Praxis liefert das Theorem in der Regel nur einen groben Anhaltspunkt, da:

- man aus praktischen Gründen, auf die wir im Abschn. 2.3 eingehen, immer eine etwas höhere Abtastrate wählen muss: $f_a > (1.1 \ldots 1.5) \cdot 2f_{max}$;
- bei vielen Signalen keine klar definierbare maximale Frequenz f_{max} existiert, z. B. bei Sprachsignalen.

Deshalb lautet das *praktische Abtasttheorem*:

Die Abtastrate muss so hoch sein, dass das System funktioniert !

Für das Sinussignal in unserem Beispiel gilt $f_{max} = f = 50\,\text{Hz}$. Demnach reicht eine Abtastung mit z. B. $f_a = 1.5 \cdot 2f_{max} = 150\,\text{Hz}$ aus. Die Berechnung der Zwischenwerte, die zur graphischen Darstellung des zugrunde liegenden kontinuierlichen Signals benötigt werden, muss dann mit einem $(\sin x/x)$–Interpolator (*Sinus-x-durch-x-Interpolator*) erfolgen; wir gehen darauf im Abschn. 7.4 noch ausführlich ein. Numerische Mathematikprogramme wie *Matlab* verzichten darauf und verbinden die Punkte statt dessen mit Geraden; Abb. 2.2 zeigt dies für verschiedene Abtastraten. Ein Vergleich der graphischen Darstellung für $f_a = 750\,\text{Hz}$ mit der graphischen Darstellung für $f_a = 1\,\text{kHz}$ in Abb. 2.1 zeigt, dass wir unter *Matlab* in der Tat eine Abtastrate von 1 kHz benötigen, um ein Signal mit einer Frequenz von 50 Hz graphisch ausreichend genau als kontinuierliches Signal darstellen zu können. Daraus folgt allgemein:

▶ In einem numerischen Mathematikprogramm wie Matlab oder Octave muss die Abtastrate eines diskreten Signals mindestens das 20-fache der maximalen Frequenz f_{max} betragen, damit das Programm das zugehörige kontinuierliche Signal graphisch ausreichend genau darstellt.

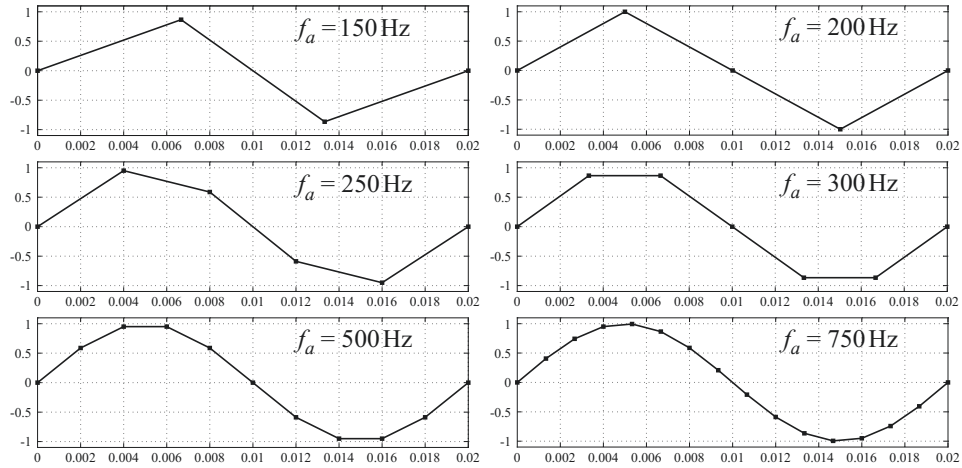

Abb. 2.2 Anzeige in *Matlab*/*Octave* für verschiedene Abtastraten f_a

Das heißt für uns:

► Da ein numerisches Mathematikprogramm wie Matlab oder Octave nur diskrete Signale verarbeiten kann, müssen wir auch die analogen Signale, die in einem SDR auftreten, durch diskrete Signale darstellen und dabei die Abtastrate so hoch wählen, dass das Programm die Signale graphisch als kontinuierliche Signale darstellen kann; dazu müssen wir $f_a \geq 20 f_{max}$ wählen.

2.2 Spektrum eines Signals

Das Spektrum eines diskreten Signals wird in der Praxis mit Hilfe der *Schnellen Fourier-Transformation* (*Fast Fourier Transform*, *FFT*) berechnet. Wir beschreiben im folgenden Abschnitt zunächst die Vorgehensweise bei der Berechnung und gehen anschließend auf die Hintergründe und Zusammenhänge ein.

2.2.1 Praktische Berechnung des Spektrums

Die Anwendung der FFT erfordert, dass das zu transformierende Signal eine Länge der Form $N = 2^L$ hat, d. h. N muss eine Potenz von 2 sein; man spricht dann von einer *N-Punkt-FFT*. Auf die Wahl von N gehen wir später noch näher ein. Für das zu transformierende,

im allgemeinen komplexe Signal gilt demnach:

$$\underline{x} \;=\; \Big[\; \underline{x}[0] \;\; \underline{x}[1] \;\; \underline{x}[2] \;\; \underline{x}[3] \;\; \ldots \;\; \underline{x}[N-2] \;\; \underline{x}[N-1] \;\Big]$$

In der Praxis handelt es sich dabei fast immer um einen Ausschnitt aus einem längeren Signal. Damit sich die Unstetigkeiten an den Rändern des Ausschnitts nicht negativ bemerkbar machen, muss man das Signal mit einer *Fenster-Funktion*

$$w \;=\; \Big[\; w[0] \;\; w[1] \;\; w[2] \;\; w[3] \;\; \ldots \;\; w[N-2] \;\; w[N-1] \;\Big]$$

gewichten. Wir verwenden im folgenden das *Blackman-Fenster*, das aufgrund seiner ausgewogenen Eigenschaften zu den am häufigsten verwendeten Fenster-Funktionen gehört.

Für die FFT mit einer Fenster-Funktion w gilt:

$$\boxed{\underline{X}[m] \;=\; \mathrm{FFT}_w\left\{\underline{x}[n]\right\} \;=\; \sum_{n=0}^{N-1} w[n]\,\underline{x}[n]\,e^{-j2\pi nm/N} \quad \text{mit } m = 0,\ldots,N-1} \tag{2.2}$$

Das Spektrum entspricht dem gewichteten Betragsquadrat des FFT-Ergebnisses:

$$\boxed{S_x[m] \;=\; \frac{1}{c_w^2}\left|\underline{X}[m]\right|^2 \quad \text{mit } c_w = \sum_{n=0}^{N-1} w[n]} \tag{2.3}$$

Wir haben bereits erwähnt, dass ein Signalwert $\underline{x}[n]$ bei einer Abtastrate $f_a = 1/T_a$ dem Zeitpunkt $t_n = nT_a$ zuzuordnen ist. Entsprechend ist ein Spektralwert $S_x[m]$ der Frequenz

$$\boxed{f_m \;=\; \frac{mf_a}{N}} \tag{2.4}$$

zuzuordnen. Da man mit $m = 0,\ldots,N-1$ Spektralwerte im Frequenzbereich $0 \le f < f_a$ erhält und das Spektrum eines diskreten Signals wegen

$$S_x[m] \;=\; S_x[m + kN] \qquad \forall k \in \mathcal{Z}$$

mit f_a periodisch ist, entspricht der Frequenzbereich $f_a/2 \le f < f_a$ dem Frequenzbereich $-f_a/2 \le f < 0$; deshalb werden in der Praxis die linke und die rechte Hälfte des Spektrums vertauscht, damit man eine Darstellung im Bereich $-f_a/2 \le f < f_a/2$ erhält. Diese Vertauschung wird *FFT Shift* genannt. Der Wert für die Frequenz Null ist nach der

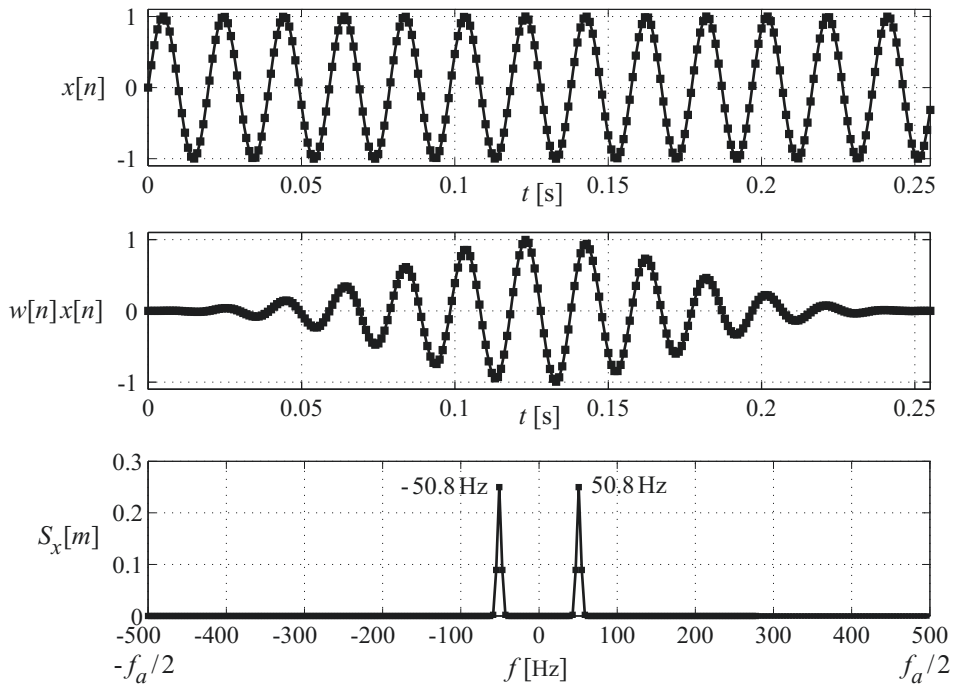

Abb. 2.3 Zeitsignal $x[n]$, gefenstertes Zeitsignal $w[n]\,x[n]$ und Spektrum $S_x[m]$ eines diskreten Sinus-Signals mit $f = 50\,\text{Hz}$, $f_a = 1\,\text{kHz}$ und $N = 256$

Vertauschung durch $S_x[N/2]$ gegeben; negative Frequenzen liegen links davon, positive rechts.

Abb. 2.3 zeigt ein Beispiel für ein Sinus-Signal. In *Matlab* schreiben wir dazu:

```
f_a = 1000;
t_a = 1 / f_a;
N = 256;
f = 13 * f_a / N;
t = ( 0 : N - 1 ) * t_a;
x = sin( 2 * pi * f * t );
w = blackman( N ).';
X_w = fft( w .* x );
c_w = sum( w );
S_x = fftshift( abs( X_w ).^2 / c_w^2 );
f_m = ( -N/2 : N/2 - 1 ) * f_a / N;
figure(1);
plot(f_m,S_x);
grid on;
```

Bei diesem Beispiel haben wir die Signalfrequenz $f = 50\,\text{Hz}$ des letzten Beispiels durch die Signalfrequenz $f = 13 f_a/N \approx 50.8\,\text{Hz}$ ersetzt, die im Frequenzraster der FFT liegt

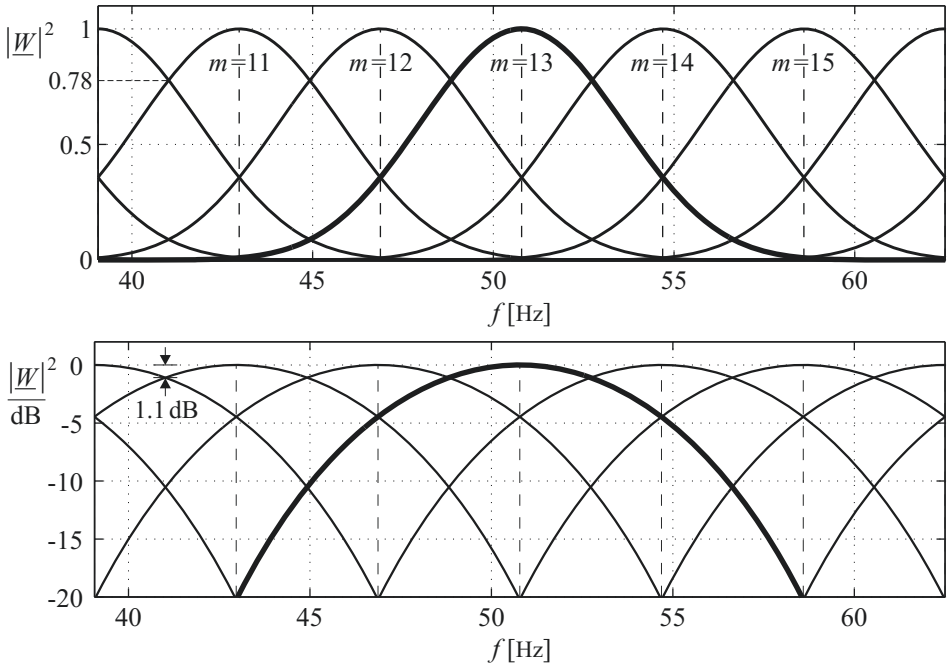

Abb. 2.4 Filterwirkung der FFT mit Blackman-Fenster im Bereich von 50 Hz

($m = 13$). In diesem Fall erhalten wir eine symmetrische Darstellung für die beiden Anteile, die sich aus der Zerlegung

$$x(t) \;=\; \sin(2\pi ft) \;=\; \frac{j}{2}\left(e^{-j2\pi ft} - e^{j2\pi ft}\right)$$

ergeben. Beide Anteile haben eine Amplitude von 0.5 und eine Leistung von 0.25.

Die FFT mit Fenster-Funktion wirkt in diesem Zusammenhang als Filterbank mit N Filtern mit den Mittenfrequenzen f_m. Abb. 2.4 zeigt einen Ausschnitt dieser Filterbank im Bereich von 50 Hz für das verwendete Blackman-Fenster; dabei ist $|\underline{W}|^2$ das Betragsquadrat der Übertragungsfunktion der Filter. Für die Signalfrequenz $f = 50.8$ Hz erhalten wir den korrekten Spektralwert $|\underline{W}|^2 = 1$ (0 dB) im Hauptfilter 13 ($m = 13$) und Nebenwerte mit $|\underline{W}|^2 \approx 0.36$ (–4.4 dB) in den benachbarten Filtern 12 und 14. Bei Signalfrequenzen außerhalb des Frequenzrasters erhalten wir einen zu geringen Spektralwert im Hauptfilter und unsymmetrische Nebenwerte in den benachbarten Filtern. Die maximale Abweichung ergibt sich für Signalfrequenzen, die genau in der Mitte zwischen zwei Rasterfrequenzen liegen; hier liefern die beiden angrenzenden Filter den Spektralwert $|\underline{W}|^2 \approx 0.78$ (–1.1 dB).

Die Länge N der FFT bestimmt die Zeit- und die Frequenzauflösung des Spektrums. Mit zunehmender Länge nimmt die Zeitauflösung ab und die Frequenzauflösung zu. Der

Abstand

$$\boxed{\Delta f \;=\; \frac{f_a}{N} \;=\; RBW} \tag{2.5}$$

der Linien des FFT-Frequenzrasters wird als *Auflösungsbandbreite* (*Resolution Bandwidth*, *RBW*) bezeichnet. Eine weitere, bei der Messung von Rauschleistungen benötigte Bandbreite ist die *Rauschbandbreite* (*Noise Bandwidth*, *NBW*), siehe Abschn. B.1:

$$NBW \;=\; \frac{f_a \sum_{n=0}^{N-1} w^2[n]}{\left(\sum_{n=0}^{N-1} w[n]\right)^2} \tag{2.6}$$

Für ein Blackman-Fenster gilt:

$$NBW \;\approx\; 1.7 \cdot RBW$$

Der exakte Wert hängt von der Länge N ab, das ist für die Praxis aber unbedeutend.

Da ein einzelnes Spektrum eine starke Varianz aufweist, werden in der Praxis mehrere Spektren gemittelt. Das am häufigsten angewendete Mittelungsverfahren ist die in Abb. 2.5 dargestellte Methode von *Welch*, bei der der auszuwertende Signalabschnitt in Blöcke der Länge $N/2$ eingeteilt wird, die dann paarweise ausgewertet werden. Wenn man die Länge des Signalschnitts wählen kann, wählt man ein Vielfaches von $N/2$, so dass kein „Rest“ auftritt. Durch die Mittelung reduziert sich die Zeitauflösung entsprechend. In der Praxis werden stark unterschiedliche Mittelungsfaktoren verwendet; dabei muss man einen Kompromiss zwischen höherer Zeitauflösung auf der einen Seite und geringerer Varianz auf der anderen Seite eingehen. Gegebenenfalls muss man die Länge N und damit die Frequenzauflösung reduzieren, um einen sinnvollen Kompromiss zwischen Zeitauflösung und Varianz des Spektrums zu erzielen.

Bei Spektrum-Analysatoren stellt man die Bandbreite des zu analysierenden Bereichs, die Auflösungsbandbreite *RBW* und den Mittelungsfaktor ein; daraus ergeben sich dann die Länge N der FFT und die Zeitauflösung. Da es keinen Sinn macht, sehr viel mehr Punkte zu berechnen als auf dem Bildschirm des Spektrum-Analysators dargestellt werden können, ist die Länge in der Regel auf 1024, 2048 oder 4096 begrenzt. Bei sehr hohen Analyse-Bandbreiten kann die Länge auch durch die Rechenleistung des FFT-Prozessors begrenzt sein.

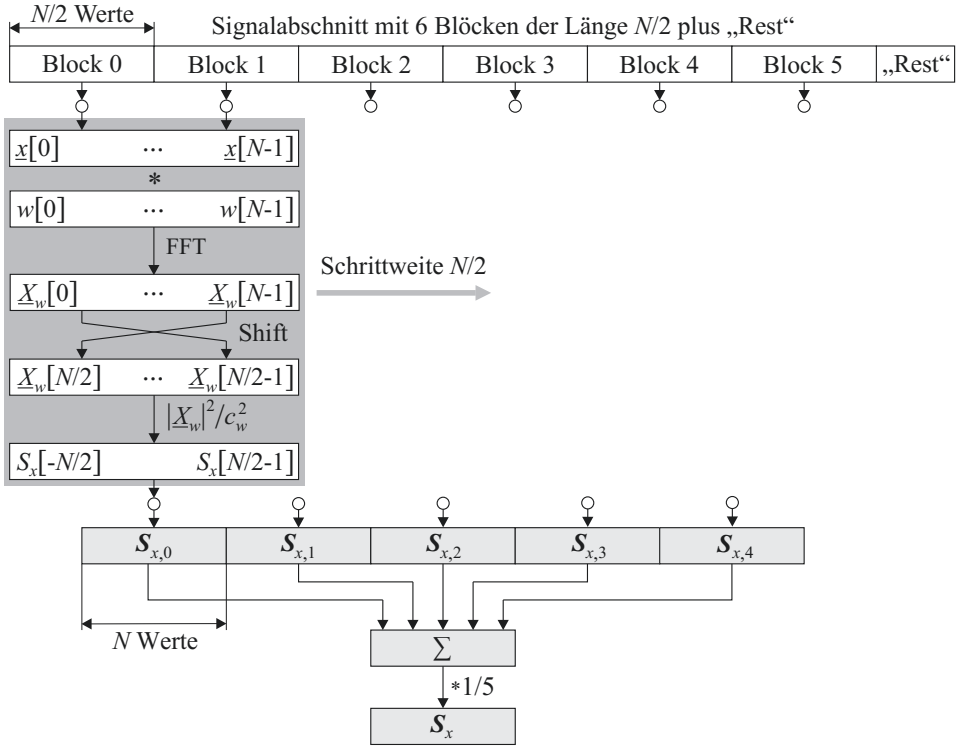

Abb. 2.5 Berechnung des Spektrums eines Signalabschnitts nach der Methode von *Welch*. Wenn möglich wählt man die Länge des Signalabschnitts so, dass kein „Rest“ anfällt

2.2.2 Hintergründe zur Berechnung des Spektrums

Das Spektrum eines Signals entspricht dem Betragsquadrat der Fourier-Transformierten des Signals:

$$S_x(f) \;=\; \left|\underline{X}(f)\right|^2 \quad \text{mit } \underline{X}(f) = \mathcal{F}\left\{\underline{x}(t)\right\} = \int_{-\infty}^{\infty} \underline{x}(t)\, e^{-j2\pi ft}\, dt$$

In der Praxis ist diese Definition aus zwei Gründen nicht direkt anwendbar:

- Ein Signal hat entweder nur eine bestimmte Dauer oder kann nur für eine bestimmte Dauer beobachtet werden. Während der Beobachtungsdauer kann sich die Charakteristik des Signals erheblich ändern, z. B. Sprache und Musik bei einem Rundfunksender. Die Beschreibung des Signals durch *ein* Spektrum wird dem nicht gerecht.

- Die Berechnung erfolgt mit digitalen Schaltkreisen oder digitalen Prozessoren, die nur diskrete Signale verarbeiten können; deshalb muss ein kontinuierliches Signal immer zunächst abgetastet werden, bevor das zugehörige Spektrum berechnet werden kann.

Dem ersten Punkt kann man dadurch Rechnung tragen, dass man das Signal mit einem *Fenster* bewertet, d. h. einen Teil des Signals herausgreift. An die Stelle der normalen Fourier-Transformation tritt dann die *Kurzzeit-Fourier-Transformation* (*Short-Time Fourier Transform*) mit der Fenster-Funktion $w(t)$ und der Verschiebung t_M, die der Mitte des herausgegriffenen Teils entspricht:

$$\underline{X}(f, t_M) \;=\; \int_{-\infty}^{\infty} w(t - t_M)\, \underline{x}(t)\, e^{-j2\pi ft}\, dt$$

Für das *gefensterte* Signal gilt demnach:

$$\underline{x}_w(t, t_M) \;=\; w(t - t_M)\, \underline{x}(t)$$

Abb. 2.6 zeigt ein Beispiel. Wir können nun das *Kurzzeit-Spektrum*

$$S_x(f, t_M) \;=\; \left|\underline{X}(f, t_M)\right|^2$$

eines Signals für eine Folge von Verschiebungen t_M berechnen und damit die zeitliche Änderung des Signals auch im Frequenzbereich erfassen. Da eine praktische Berechnung immer unter Verwendung einer Fenster-Funktion erfolgt und deshalb jedes praktisch berechnete Spektrum ein Kurzzeit-Spektrum ist, lässt man den Zusatz *Kurzzeit* weg und spricht nur vom *Spektrum*.

Als Fenster-Funktionen werden Funktionen verwendet, die nur in einem vorgegebenen Intervall ungleich Null sind; dadurch wird die Berechnung auf dieses Intervall begrenzt. Das Ergebnis der Berechnung hängt von der Form der Fenster-Funktion ab; deshalb werden in der Praxis zahlreiche verschiedene Fenster-Funktionen verwendet, die entweder gezielt für eine bestimmte Aufgabe optimiert wurden oder einen guten Kompromiss für mehrere verschiedene Aufgaben darstellen. Die Eigenschaften dieser Fenster-Funktionen werden in Vorlesungen oder Büchern über Digitale Signalverarbeitung ausführlich beschrieben. Wir verwenden in unseren Beispielen das *Blackman-Fenster*.

Wir kommen nun zum zweiten Punkt: dem Übergang zu diskreten Signalen und – damit verbunden – zu diskreten Fenster-Funktionen. Der Übergang erfolgt durch Abtastung. In der Signaltheorie wird die Abtastung durch Multiplikation mit einem Dirac-Impulskamm

$$SAMPLE\,(t, T_a) \;=\; \sum_{n=-\infty}^{\infty} \delta_0(t - nT_a)$$

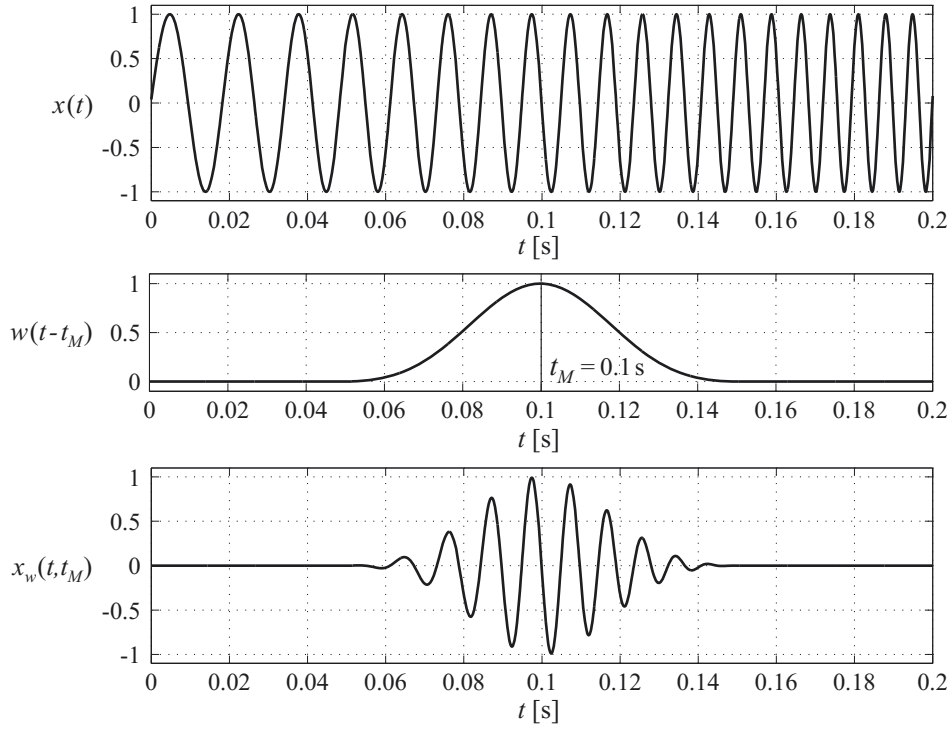

Abb. 2.6 Fensterung eines Signals mit einer Fenster-Funktion $w(t)$ bei einer Verschiebung von $t_M = 0.1\,\mathrm{s}$

beschrieben:

$$\underline{x}_S(t) \;=\; \underline{x}(t) \cdot SAMPLE\,(t, T_a) \;=\; \sum_{n=-\infty}^{\infty} \underline{x}(nT_a)\,\delta_0(t - nT_a)$$

Die Gewichte der Dirac-Impulse entsprechen den Werten des diskreten Signals:

$$\underline{x}[n] \;=\; \underline{x}(nT_a)$$

Man kann nun die Fourier-Transformation für diskrete Signale aus der Fourier-Transformation für kontinuierliche Signale ableiten:

$$\underline{X}_S(f) \;=\; \mathcal{F}\left\{\underline{x}_S(t)\right\} = \int_{-\infty}^{\infty} \underline{x}_S(t)\, e^{-j2\pi ft}\, dt$$

$$= \int_{-\infty}^{\infty} \left(\sum_{n=-\infty}^{\infty} \underline{x}(nT_a)\, \delta_0(t - nT_a) \right) e^{-j2\pi ft}\, dt$$

$$= \sum_{n=-\infty}^{\infty} \underline{x}[n]\, e^{-j2\pi f\, T_a n}$$

Wir wenden auch hier wieder eine Fenster-Funktion an. Diskrete Fenster-Funktionen $w[n]$ haben in der Praxis immer eine geradzahlige Länge N. Als Bezugspunkt wird hier nicht die Mitte, sondern der erste Wert verwendet; ohne Verschiebung gilt demnach:

$$w \;=\; \Big[\; w[0] \;\; w[1] \;\; w[2] \;\; \ldots \;\; w[N-2] \;\; w[N-1] \;\Big]$$

Bei einer Verschiebung um $t_M = n_M T_a$ erhalten wir die Fourier-Transformierte:

$$\underline{X}_w(f, n_M) \;=\; \sum_{n=n_M}^{n_M+N-1} w[n - n_M]\, \underline{x}[n]\, e^{-j2\pi f\, T_a n}$$

Sie ist periodisch mit der Periode $f_a = 1/T_a$, da

$$e^{-j2\pi f\, T_a n} \;=\; e^{-j2\pi (f + m f_a)\, T_a n} \qquad \forall m \in \mathcal{Z}$$

gilt. Auch diese Fourier-Transformierte müssen wir in der Praxis wieder als diskrete Folge darstellen, d. h. wir berechnen sie nur für bestimmte Werte der Frequenz f. Dabei erweist es sich als sinnvoll, $N = 2^L$ Werte der Form

$$f_m \;=\; \frac{m f_a}{N} \quad \text{mit } m = 0, 1, \ldots, N-1$$

zu verwenden, weil die Berechnung dann mit Hilfe der *Schnellen Fourier-Transformation* (*Fast Fourier Transform*, *FFT*) erfolgen kann:

$$\underline{X}_w(f_m, n_M) \;=\; \sum_{n=n_M}^{n_M+N-1} w[n - n_M]\, \underline{x}[n]\, e^{-j2\pi mn/N}$$

Für das Spektrum gilt dann:

$$\tilde{S}_x(f_m, n_M) \;=\; \left| \underline{X}_w(f_m, n_M) \right|^2$$

Da eine zeitliche Verschiebung eines Signals nur eine Phasendrehung der zugehörigen Fourier-Transformierten bewirkt und die Phase aufgrund der Betragsquadrat-Bildung nicht

in das Spektrum eingeht, können wir anstelle des Fensters auch das Signal verschieben; dann gilt:

$$\tilde{S}_x(f_m, n_M) = \left| \sum_{n=0}^{N} w[n]\, \underline{x}[n + n_M]\, e^{-j2\pi mn/N} \right|^2$$

Für ein konstantes Signal $\underline{x}[n] = 1\ \forall n$ erhalten wir für $f_m = 0$:

$$\tilde{S}_x(0, n_M) = \left| \sum_{n=0}^{N} w[n] \right|^2$$

Da das Spektrum bei $f_m = 0$ in diesem Fall Eins betragen muss, müssen wir also noch mit dem Quadrat der Summe der Fensterkoeffizienten normieren:

$$S_x(f_m, n_M) = \frac{\tilde{S}_x(f_m, n_M)}{c_w^2} \quad \text{mit } c_w = \sum_{n=0}^{N} w[n]$$

Damit haben wir die Berechnungsvorschrift für ein einzelnes Spektrum.

2.2.3 Beispiel

Für das Beispiel verwenden wir das in Abb. 2.7 gezeigte Mehrfrequenzsignal, das zur Wahl einer Telefonnummer in einem analogen Telefonsystem verwendet wird. Das Signal besteht aus 15 Abschnitten mit je einem Tonpaar. Am Zeitsignal können wir das aber nur bedingt erkennen. Die Abtastrate beträgt $f_a = 8\,\text{kHz}$. Vor der Berechnung haben wir noch Rauschen addiert, das in Abb. 2.7 nicht enthalten ist. Abb. 2.8 zeigt ein Spektrum aus dem ersten Tonpaar. Die Töne haben jeweils die Amplitude 0.5, den Effektivwert $0.5/\sqrt{2} = 0.3535$ und die Leistung 0.125; folglich haben die beiden Anteile eines Tons jeweils die Leistung 0.0625 bzw. $10 \log_{10} 0.0625 = -12\,\text{dB}$.

Etwas schwieriger ist die Bestimmung der Leistung beim Rauschen. Wir haben dem Signal weißes Rauschen mit einer Leistung von 0.01 hinzugefügt. In *Matlab* verwendet man dazu den Befehl `randn`, der ein weißes Rauschsignal mit einer Leistung von Eins erzeugt, das durch Skalierung mit $\sqrt{P_n}$ auf die Leistung P_n umgerechnet wird; wir schreiben also:

```
% ... das Nutzsignal sei im Vektor x gegeben ...
P_n = 0.01;
n = sqrt( P_n ) * randn( 1, length(x) );
x = x + n;
```

Bei weißem Rauschen verteilt sich die Rauschleistung im langfristigen Mittel gleichmäßig auf den Frequenzbereich $-f_a/2 < f < f_a/2$, d. h. auf einen Bereich mit der Bandbreite f_a;

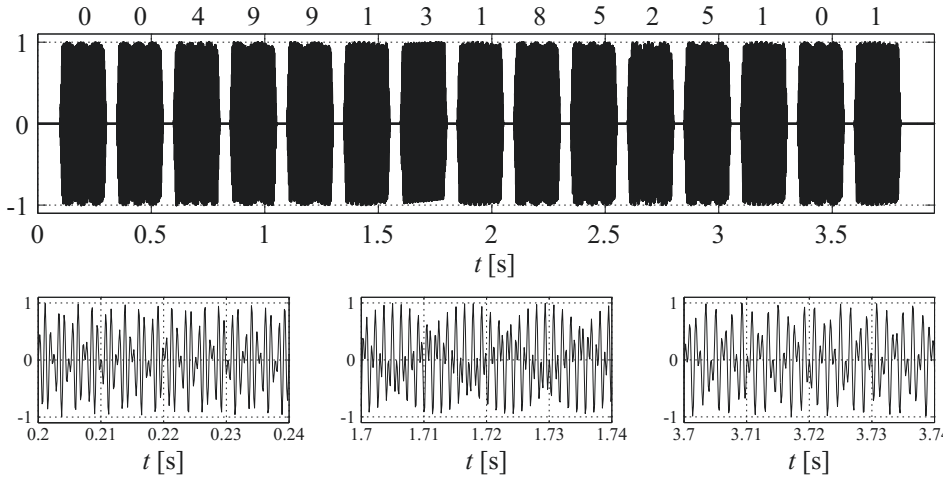

Abb. 2.7 Mehrfrequenzsignal (MFV) zur Wahl der Telefonnummer 004991318525101 in einem analogen Telefonsystem

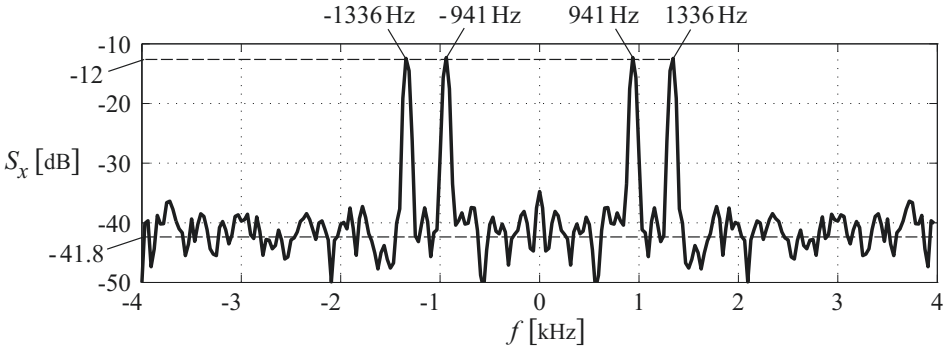

Abb. 2.8 Spektrum aus dem ersten Tonpaar

deshalb beträgt die *Rauschleistungsdichte*:

$$S_n(f) \;=\; \frac{P_n}{f_a} \;=\; \frac{0.01}{8\,\text{kHz}} \;=\; 1.25 \cdot 10^{-6}\,\text{Hz}^{-1}$$

Für ein Blackman-Fenster-erhalten wir bei einer FFT-Länge von $N = 256$ die Auflösungsbandbreite

$$RBW \;=\; \frac{f_a}{N} \;=\; \frac{8\,\text{kHz}}{256} \;=\; 31.25\,\text{Hz}$$

und die Rauschbandbreite:

$$NBW \;\approx\; 1.7 \cdot RBW \;\approx\; 53\,\text{Hz}$$

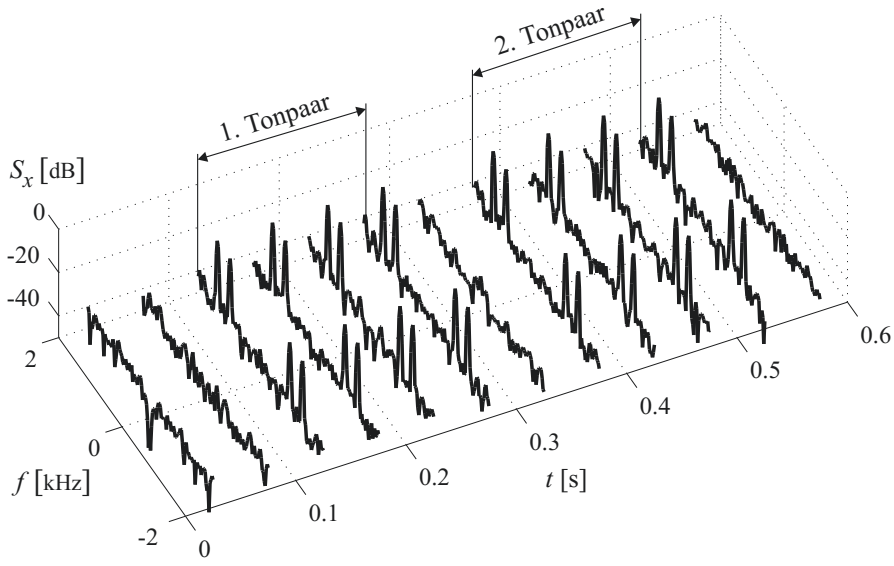

Abb. 2.9 3-dimensionale Darstellung der Spektren der ersten beiden Tonpaare

In jedes Filter der FFT-Filterbank fällt demnach im langfristigen Mittel die Rauschleistung:

$$P_n(f_m) = S_n(f) \cdot NBW \;\approx\; 1.25 \cdot 10^{-6}\,\mathrm{Hz}^{-1} \cdot 53\,\mathrm{Hz} \;\approx\; 6.6 \cdot 10^{-5}$$

$$\Rightarrow \quad 10 \log_{10} P_n(f_m) \;\approx\; -41.8\,\mathrm{dB}$$

Diesen Wert können wir in Abb. 2.8 ablesen. Bei einer Messung mit einem Spektral-Analysator wird entsprechend *rückwärts* gerechnet.

Wir haben in diesem Beispiel Signalabschnitte mit einer Länge von 400 Abtastwerten ausgewertet. Das entspricht bei einer Abtastrate von 8 kHz einer Signaldauer von 50 ms. Für jeden Abschnitt haben wir nach der Welch-Methode zwei Spektren der Länge 256 berechnet und gemittelt. Da wir dazu nur 384 Abtastwerte benötigen, wurden in jedem Abschnitt die letzten 16 Abtastwerte nicht berücksichtigt; hier war uns aber eine *runde* Dauer von 50 ms wichtiger.

Ein Spektrum-Analysator stellt die berechneten Spektren sukzessive mit einer Rate von $1/0.05 = 20$ Bildern pro Sekunde dar. Die ersten 12 Spektren sind in Abb. 2.9 als 3D-Graphik dargestellt; dabei haben wir den dargestellten Frequenzbereich auf ± 2 kHz begrenzt. Eine weitere, häufig verwendete Darstellung ist das in Abb. 2.10 gezeigte *Spektrogramm*, bei der die Spektren farb- oder grau-codiert in einer t–f–Ebene dargestellt werden.

Es fällt auf, dass bei der Berechnung der Leistungsanzeige der Töne die Länge N der FFT nicht eingeht, bei der Berechnung der Leistungsanzeige des Rauschens dagegen

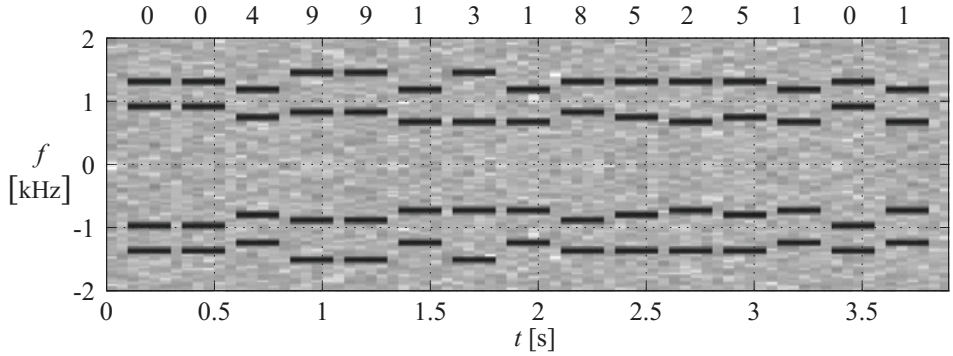

Abb. 2.10 2-dimensionale Darstellung der Spektren in Form eines Spektrogramms

schon. Das hängt mit dem Verhältnis der Bandbreiten der verschiedenen Signalanteile – hier jeweils zwei Töne und Rauschen – zur Auflösungsbandbreite zusammen:

- Ein konstanter Ton hat die Bandbreite Null und fällt damit immer als Ganzes in eines der Filter der FFT-Filterbank; deshalb ändert sich die angezeigte Leistung nicht, wenn man die Länge N bzw. die Auflösungsbandbreite ändert. Das gilt auch für Signale mit endlicher Bandbreite, solange ihre Bandbreite kleiner als die Auflösungsbandbreite bleibt.
- Bei breitbandigen Signalen, deren Leistung in mehrere Filter fällt, hängt die angezeigte Leistung von der Rauschbandbreite der FFT-Filterbank ab. Hier ändern sich die angezeigten Werte, wenn man die Länge N ändert. Bei einer Verdoppelung von N halbiert sich die Rauschbandbreite; dadurch nehmen die angezeigten Werte um 3 dB ab.

Abb. 2.11 zeigt dies am Beispiel eines Sinus-Signals mit Rauschen. Die Anzeige der Leistung des Sinus-Signals bleibt unverändert, während die Anzeige der Rauschleistung mit

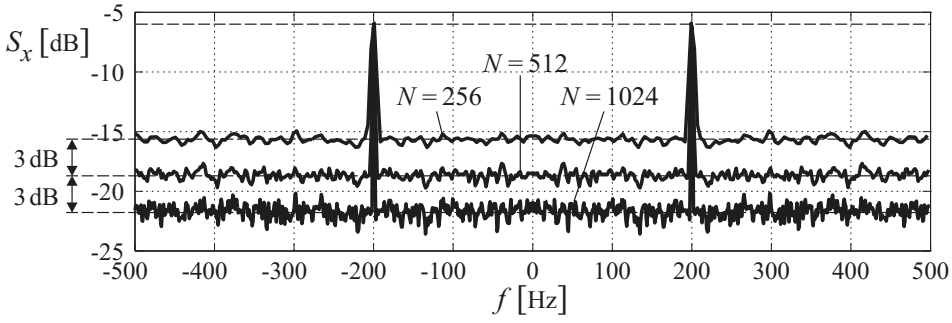

Abb. 2.11 Spektren eines Sinus-Signals mit Rauschen für verschiedene FFT-Längen

zunehmender FFT-Länge abnimmt. Da wir für alle drei Fälle einen Signalabschnitt gleicher Länge verwendet haben, nimmt mit zunehmender FFT-Länge auch die Anzahl der nach der Welch-Methode gemittelten Spektren ab, so dass der Verlauf der angezeigten Rauschleistung aufgrund der größeren Varianz *rauer* wird. Ist das Sinus-Signal deutlich schwächer als hier angenommen, kann es in der Anzeige des Rauschens *versinken* und auch bei hoher FFT-Länge nicht mehr erkannt werden. In diesem Fall kann nur ein längerer Signalabschnitt Abhilfe schaffen.

In der Praxis werden häufig zwei Messungen durchgeführt: eine Messung mit hoher FFT-Länge zur Ermittlung der sinus-förmigen Anteile und eine Messung mit geringer FFT-Länge zur exakteren Ermittlung der Rauschleistungsdichte.

2.3 Unterabtastung und Überabtastung

Eine der wichtigsten Signalverarbeitungsoperationen in einem SDR ist die Änderung der Abtastrate (*Resampling*) diskreter Signale durch Unterabtastung oder Überabtastung. Da wir diese Operationen in den folgenden Beispielen immer wieder anwenden werden, gehen wir im folgenden auf die einfachsten Realisierungen unter *Matlab* ein; effizientere Realisierungen behandeln wir im Kap. 7.

Bei einer Unterabtastung wird die Abtastrate von $f_{a,high}$ auf $f_{a,low} = f_{a,high}/M$ reduziert. Die Unterabtastung selbst erfolgt unter *Matlab*, indem man aus einem Signalvektor mit der Abtastrate $f_{a,high}$ jeden M-ten Werte entnimmt:

```
x_low = x_high( 1 : M : end );
```

Vorher müssen wir das Signal aber so filtern, dass das Abtasttheorem auch nach der Unterabtastung erfüllt ist, sofern das nicht ohnehin schon der Fall ist. Bei einer Überabtastung wird die Abtastrate von $f_{a,low}$ auf $f_{a,high} = Mf_{a,low}$ erhöht, indem zunächst zwischen je zwei Abtastwerten $M - 1$ Nullen eingefügt werden. In *Matlab* erfolgt dies mit Hilfe des Kronecker-Produkts `kron`:

```
x_high = kron( x_low, [ 1 zeros(1,M-1) ] );
```

Hier muss anschließend auf jeden Fall eine Filterung erfolgen, die die eingefügten Nullen durch interpolierte Werte ersetzt. Das Filter muss dabei die Gleichverstärkung M besitzen, damit die Leistung des Signals trotz der eingefügten Nullen erhalten bleibt.

Abb. 2.12 zeigt die Blockschaltbilder und die Spektren für die beiden Operationen am Beispiel $M = 4$. Bei der Unterabtastung dient die Filterung der Unterdrückung unerwünschter Signalanteile, die über den Alias-Effekt in den Nutzbereich am Ausgang fallen würden. Bei der Überabtastung erhält man durch das Einfügen der Nullen eine periodische

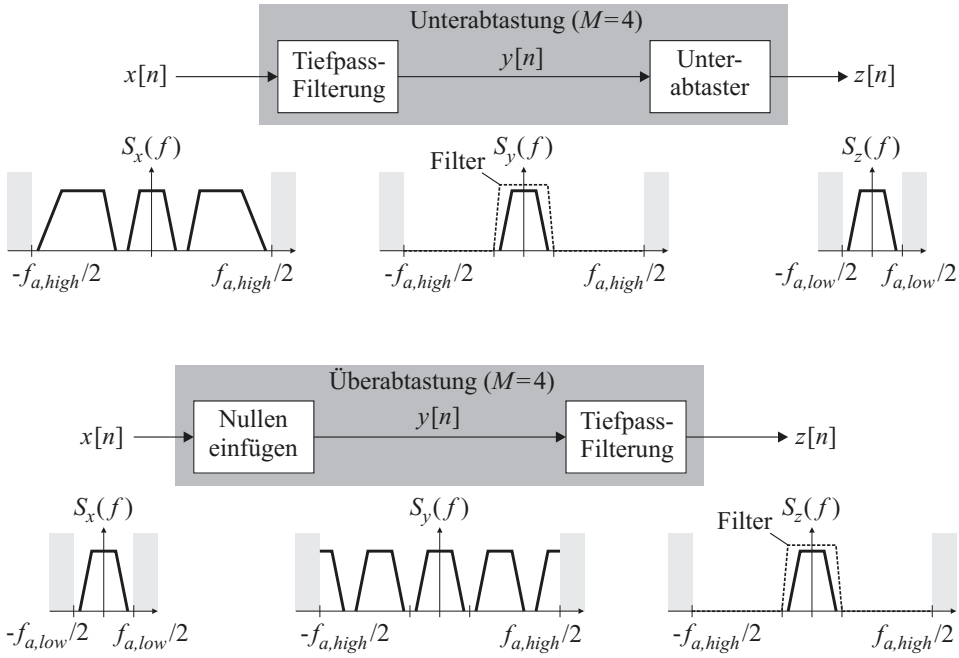

Abb. 2.12 Unterabtastung und Überabtastung am Beispiel $M = 4$

Fortsetzung des Nutzbereichs, aus der der ursprüngliche Nutzbereich durch Filterung herausgeschnitten werden muss.

Das ideale Filter für die Tiefpass-Filterung wäre der aus der Signaltheorie bekannte *ideale Tiefpass* mit der Übertragungsfunktion

$$\underline{H}(f) = \begin{cases} 1 & \text{für } -f_{a,low}/2 \le f \le f_{a,low}/2 \\ 0 & \text{für } f_{a,low}/2 < |f| < f_{a,high}/2 \end{cases}$$

und der zeitdiskreten Impulsantwort:

$$h[n] = \frac{1}{f_{a,high}} \int_{-f_{a,high}/2}^{f_{a,high}/2} \underline{H}(f)\, e^{j2\pi nf/f_{a,high}}\, df = \frac{1}{f_{a,high}} \int_{-f_{a,low}/2}^{f_{a,low}/2} e^{j2\pi nf/f_{a,high}}\, df$$

$$= \frac{1}{\pi n} \sin\left(\pi n \frac{f_{a,low}}{f_{a,high}}\right) = \frac{1}{\pi n} \sin\left(\frac{\pi n}{M}\right) \quad \text{mit } M = \frac{f_{a,high}}{f_{a,low}}$$

2.3.1 Einfache Berechnung von Tiefpass-Filtern

In der Praxis wird ein FIR-Filter verwendet; dazu müssen wir die unendlich lange Impulsantwort $h[n]$ auf einen Bereich $-N_h \leq n \leq N_h$ beschränken, so dass wir ein Filter mit $N = (2N_h + 1)$ Koeffizienten erhalten. In *Matlab* schreiben wir dazu:

```
% ... die Abtastraten f_a_high und f_a_low seinen gegeben ...
N_h = <vorgegebener Wert>;
M = f_a_high / f_a_low;
n = 1 : N_h;
h = sin( pi * n / M ) ./ ( pi * n );
h = [ fliplr(h) 1/M h ];
```

Abb. 2.13 zeigt ein Beispiel für den Betragsfrequenzgang eines *abgeschnittenen* idealen Tiefpass-Filters. Der Verlauf ist aufgrund der durch das Abschneiden verursachten Welligkeit im Durchlassbereich und der geringen Dämpfung im Sperrbereich unbefriedigend. Wir können aber eine deutliche Verbesserung erreichen, indem wir die Koeffizienten mit einer Fenster-Funktion bewerten. Dafür eignet sich das *Kaiser-Fenster*, da es einen Parameter β besitzt, mit dem man einen Kompromiss zwischen der Steilheit der Filterflanken und der Sperrdämpfung erzielen kann; dabei gilt für die Sperrdämpfung a an der Flanke des Filters:

$$a \approx \beta \cdot 10\,\text{dB}$$

In *Matlab* führen wir die Bewertung mit

```
beta = <Sperrdämpfung an der Flanke in dB / 10>;
h = h .* kaiser( length(h), beta ).';
```

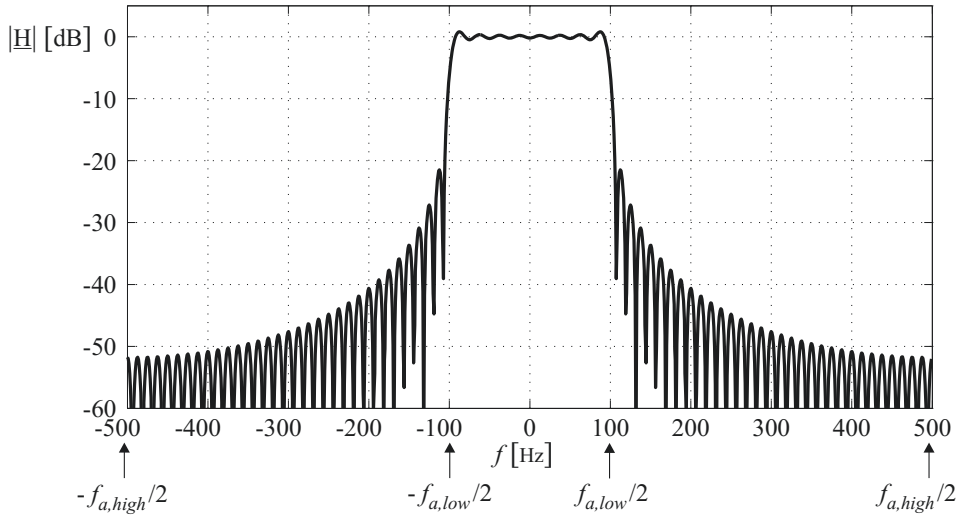

Abb. 2.13 Betragsfrequenzgang eines *abgeschnittenen* idealen Tiefpass-Filters mit $f_{a,high} = 1000\,\text{Hz}$, $f_{a,low} = 200\,\text{Hz}$ und $N_h = 40$ (81 Filter-Koeffizienten)

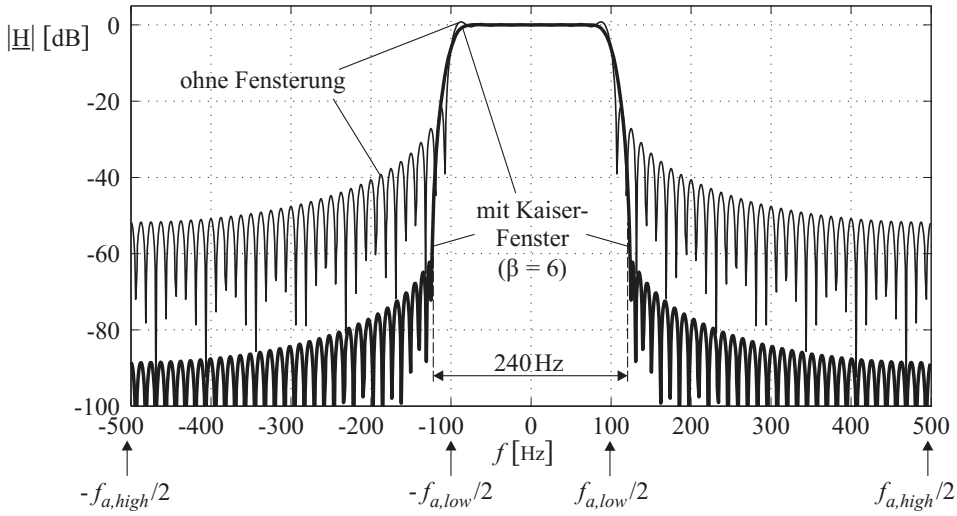

Abb. 2.14 Betragsfrequenzgang des Filters aus Abb. 2.13 vor und nach der Bewertung mit einem Kaiser-Fenster mit $\beta = 6$

durch; Abb. 2.14 zeigt das Ergebnis. Durch die Fensterung nimmt allerdings die Bandbreite des Filters zu. Das können wir kompensieren, indem wir anstelle von M einen etwas größeren Wert $\tilde{M} \approx (1.1 \ldots 1.2)\, M$ einsetzen. Anschließend müssen wir die resultierende Breite des Durchlassbereichs prüfen, die von der zulässigen Dämpfung an den Rändern des Durchlassbereichs abhängt.

Wir haben demnach drei Parameter, mit denen wir das Filter an unsere Anforderungen anpassen können:

- die Bereichsgrenze N_h, die die Anzahl N der Koeffizienten des Filters bestimmt;
- den Parameter β des Kaiser-Fensters, mit dem wir die Sperrdämpfung einstellen;
- den Bandbreiten-Faktor $\tilde{M}/M \approx 1.1 \ldots 1.2$, mit dem wir die Bandbreite justieren.

Für unsere weiteren Beispiele verwenden wir $\beta = 6$, $\tilde{M}/M = 1.11$ und $N_h \approx 16\, M$:

```
% ... die Abtastraten f_a_high und f_a_low seinen gegeben ...
M = f_a_high / f_a_low;
N_h = floor( 16 * M );
K = 1.11;
n = 1 : N_h;
h = sin( pi * n / ( K * M ) ) ./ ( pi * n );
h = [ fliplr(h)  1 / ( K * M )  h ] .* kaiser( 2 * N_h + 1, 6 ).';
```

Bei einer zulässigen Dämpfung von 0.1 dB an den Rändern des Durchlassbereichs erhalten wir damit eine Bandbreite:

$$B \approx 0.8 f_{a,low} = \frac{0.8 f_{a,high}}{M}$$

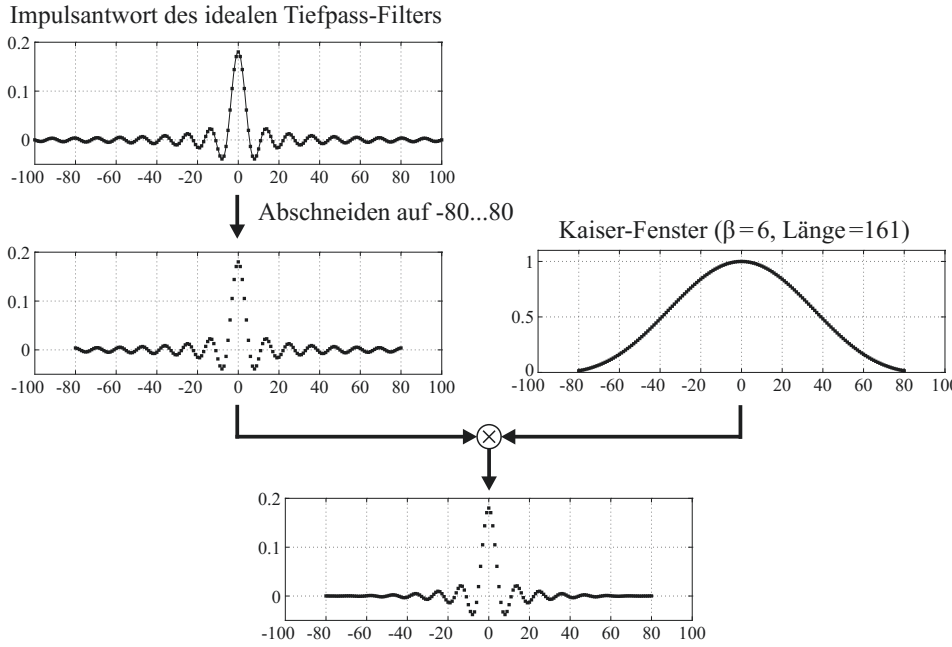

Abb. 2.15 Berechnung der Koeffizienten eines Tiefpass-Filters für $M = 5$ ($N = 161$)

Die –3 dB–Bandbreite beträgt etwa $0.875 f_{a,low}$. Abb. 2.15 zeigt die Berechnung der Koeffizienten eines Tiefpass-Filters für $M = 5$; in diesem Fall erhalten wir

$$N \;=\; 2N_h + 1 \overset{N_h = 16\,M}{=} \;=\; 32\,M + 1 \overset{M=5}{=} 32 \cdot 5 + 1 \;=\; 161$$

Koeffizienten.

Für eine systemtheoretisch korrekte Darstellung als kausales Filter müssen wir die Koeffizienten um N_h Werte nach *rechts* verschieben, damit

$$h[n] \;=\; 0 \quad \text{für } n < 0$$

gilt; dadurch wird der Bereich $n = -N_h \ldots N_h$ nach $n = 0 \ldots 2N_h$ verschoben. Für die Anwendung bedeutet das, dass das Filter eine Verzögerung um N_h Abtastwerte verursacht.

Ein derart berechnetes Filter ist in der Regel nicht optimal. In der Praxis kann man mit dem iterativen *Remez-Verfahren* Filter mit günstigeren Eigenschaften berechnen, muss dazu aber ein Toleranzschema für den Betragsfrequenzgang angeben und ggf. mehrere Versuche unternehmen, bis das Verfahren konvergiert. Für unsere Zwecke reicht die hier vorgestellte nicht-iterative Berechnung, in die nur der Faktor $M = f_{a,high}/f_{a,low}$ als variabler Parameter eingeht, aber vollkommen aus.

2.3.2 Filter-Berechnungsfunktion

In unseren weiteren Beispielen werden wir die Überabtastung nicht nur an den Stellen einsetzen, an denen in einem SDR tatsächlich eine Überabtastung erfolgen muss, sondern auch immer dann, wenn wir von einem diskreten Signal nicht nur die einzelnen Abtastwerte, sondern den Verlauf des zugrunde liegenden kontinuierlichen Signals graphisch darstellen wollen; dazu benötigen wir – wie in Abschn. 2.1 beschrieben – ein diskretes Signal mit einer etwa 10-fach höheren Abtastrate.

Zur bequemeren Nutzung in den folgenden Beispielen fassen wir die Berechnung zu einer eigenen Funktion zusammen:

```
function h = resampling_filter(M)
% h = resampling_filter(M)
%
% FIR lowpass filter for resampling
%
%   M - ratio of sampling rates (M >= 1)

N_h = floor( 16 * M );
n = 1 : N_h;
K = 1.11;
h = sin( pi * n / ( K * M ) ) ./ ( pi * n );
h = [ fliplr(h)  1 / ( K * M )  h ] .* kaiser( 2 * N_h + 1, 6 ).';
```

Das resultierende Filter hat die Gleichverstärkung Eins (0 dB). Um die bei einer Überabtastung erforderliche Gleichverstärkung M zu erhalten, können wir entweder das Ein- oder Ausgangssignal des Filters oder die Filterkoeffizienten mit M multiplizieren. In unseren Beispielen realisieren wir die Multiplikation mit M in der Regel dadurch, dass wir beim Einfügen der Nullen

```
x_high = kron( x_low, [ M zeros(1,M-1) ] );
```

schreiben; dadurch werden die Abtastwerte mit M multipliziert.

Durch das Einfügen von Nullen können wir die Filterung mit den Funktionen `conv` bzw. `filter` durchführen. Das ist für unsere Zwecke ausreichend, für praktische Anwendungen jedoch ineffizient. Das Einfügen von Nullen bei der Überabtastung und das Verwerfen von berechneten Werten bei der Unterabtastung kann durch die Verwendung eines Polyphasen-FIR-Filters vermieden werden. Wir gehen darauf im Abschn. 7.2 ein.

2.4 Berechnung allgemeiner Tiefpass-Filter

Die Berechnung allgemeiner Tiefpass-Filter kann ebenfalls mit dem im vorausgehenden Abschnitt beschriebenen Verfahren erfolgen. In diesem Fall liegt kein Bezug zu einer

Unter- oder Überabtastung vor, d. h. die Abtastraten am Eingang und am Ausgang haben denselben Wert f_a. An die Stelle des Faktors M tritt die auf die Abtastrate f_a bezogene *relative Bandbreite*:

$$B_{rel} = \frac{B}{f_a} = 0.01 \ldots 0.8$$

Zur einfachen Berechnung haben wir die Funktion

```
h = lowpass_filter( B_rel, beta, N )
```

erstellt, bei der neben der relativen Bandbreite `B_rel` optional der Parameter `beta` des Kaiser-Fensters und die Koeffizientenanzahl `N` angegeben werden kann. Letztere beträgt ohne explizite Vorgabe:

$$N \approx \frac{32}{B_{rel}}$$

Wir haben für B_{rel} die willkürliche Untergrenze 0.01 festgelegt, um die maximale Koeffizientenanzahl auf etwa 4000 zu beschränken. Das zur Bandbegrenzung eines Signals mit der Abtastrate f_a auf die Bandbreite B benötigte Filter können wir nun bequem mit

```
h = lowpass_filter( B / f_a );
```

berechnen.

2.5 Zusammenfassung

Im diesem Abschnitt haben wir die zur Darstellung von Signalen und Spektren benötigten Grundlagen behandelt und durch Beispiele verdeutlicht. Nach diesen Grundlagen arbeiten auch moderne Messgeräte wie z. B. Oszilloskope, Spektrum-Analysatoren und spezielle Signalanalysatoren. Die konkreten Berechnungen erfolgen zwar häufig mit Hilfe von optimierten Algorithmen, bei denen bestimmte Zusammenhänge ausgenutzt werden, um den Rechenaufwand zu reduzieren, die zugrunde liegenden Grundprinzipien sind aber dieselben.

http://www.springer.com/978-3-662-53233-1

Software Defined Radio-Systeme für die Telemetrie
Aufbau und Funktionsweise von der Antenne bis zum Bit-Ausgang

Heuberger, A.; Gamm, E.

2017, XIII, 383 S. 288 Abb., Hardcover
ISBN: 978-3-662-53233-1