UNIVERSITÄT DER BUNDESWEHR MÜNCHEN
Fakultät für Elektrotechnik und Informationstechnik

Berücksichtigung unterschiedlicher Kanalqualitäten bei der Decodierung

Georg Hoever

Vorsitzender des Promotionsausschusses:	Prof. Dr.rer.nat. Claus Hillermeier
1. Berichterstatter:	Prof. Dr.rer.nat. Dr.-Ing. Stefan Schäffler
2. Berichterstatter:	Prof. Dr.-Ing. Berthold Lankl
3. Berichterstatter:	Prof. Dr.rer.nat. Albert Gilg

Tag der Prüfung: 21.04.2004

Mit der Promotion erlangter akademischer Grad:
Doktor-Ingenieur
(Dr.-Ing.)

München, den 10.07.2004

Inhaltsverzeichnis

Symbolverzeichnis

Im Folgenden werden die in dieser Arbeit benutzten Symbole aufgelistet und erklärt:

a_k, a_l	Vertrauenswert, $= \lvert y_k \rvert$
Burst(k)	Burst, mit dem das k-te Bit eines Codewortes übertragen wird
b	Infowort
$\hat{b}$	Schätzung eines Infoworts
b_l	Infobit bzw. Infowort
c	Codewort
$\hat{c}$	Schätzung eines Codeworts
c_k, c_l	Codebit ($= \pm 1$) bzw. Codewort
$\tilde{c_k}$	vermutliches Codebit, $= \operatorname{sign}(y_k)$
f_k	Dämpfungs-/Fadingwert
f_{Burst}	Burst-abhängiger Dämpfungs-/Fadingwert
K	Anzahl der Bits in einem Codewort
k	eine natürliche Zahl
l	eine natürliche Zahl
μ	Mittelwert einer Normalverteilung
μ_{Burst}	Burst-abhängiger Mittelwert einer Normalverteilung
$\hat{\mu}_{\text{Burst}}$	geschätzter Mittelwert für einen Burst
n_k	Rauschwert (unabhängig normalverteilt)
P	Produkt von Dichtewerten
p	Dichtefunktion
q_k, q_l	Kanalqualitätswert
$\mathbb{R}$	die reellen Zahlen
r_k	Rauschwert (unabhängig normalverteilt)

$S_1(c, y)$, $S_2(c, y)$	Metrik zur Viterbi-Decodierung
σ	Standardabweichung einer Normalverteilung
σ^2	Varianz einer Normalverteilung
σ_{Burst}	Burst-abhängige Standardabweichung einer Normalverteilung
$\hat{\sigma}_{\text{Burst}}$	geschätzte Standardabweichung für einen Burst
$w_0, \ldots, w_3$	Codewörter
x	Sendeleistung
y	empfangene Folge
y_k	empfangener Wert ($\in \mathbb{R}$)
|Burst|	Anzahl der Bits innerhalb eines Bursts

Einleitung und Grundlagen

Im Vergleich zur drahtgebundenen Informationsübermittlung ist die drahtlose Kommunikation prinzipiell größeren Störungen ausgesetzt. Bei der drahtlosen Kommunikation spielen daher Techniken zur Fehlererkennung und Fehlerkorrektur eine wichtige Rolle. Die vorliegende Arbeit beschäftigt sich mit verbesserten Verfahren zur Fehlerkorrektur.

Fehlerursachen sind zum einen zufällige Störungen, z. B. atmosphärisches Rauschen oder Störstrahlungen, zum anderen Überlagerungen, die aus der unterschiedlichen Länge verschiedener Übertragungspfade resultieren. In Abbildung

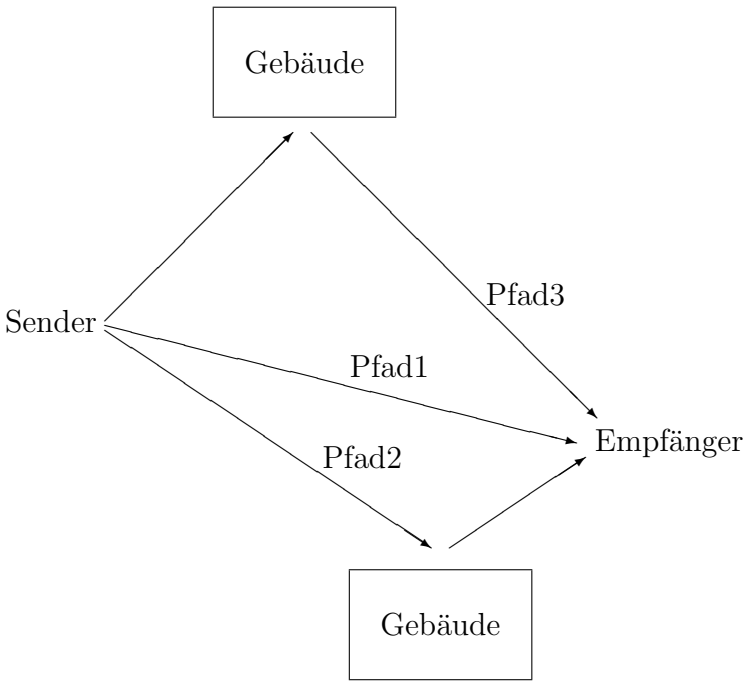

Abbildung 1: Übertragungspfade durch unterschiedliche Reflexionen

1 ist die Länge von Pfad 1 kleiner als die von Pfad 2, die wiederum kleiner ist als die Länge von Pfad 3. Beim Empfänger überlagern sich die Signale der Übertragung auf Grund der unterschiedlich langen Pfade. In ungünstigen Fällen, z. B. falls es zwei gleichberechtigte Übertragungspfade gibt, deren Längen sich gerade um eine halbe Wellenlänge der benutzten Trägerfrequenz unterscheiden, kann es sogar zur Auslöschung des Signals kommen (s. [11]).

Um eine möglichst gute Übertragung zu gewährleisten, hat sich bei GSM (*Global System for Mobile Communications*) ein mehrstufiges Verfahren etabliert, wie es in Abbildung 2 skizziert ist (s. [14, 16]). Im Folgenden wird diese Kette soweit erläutert, wie es für das Verständnis der Arbeit notwendig ist.

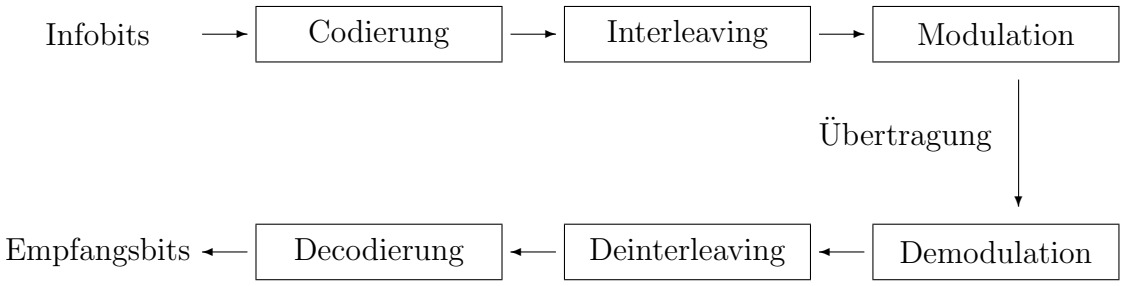

Abbildung 2: Mehrstufiges Verfahren zur Übertragung

Codierung

Am Anfang stehen die Infobits, die die eigentliche Information enthalten und die übermittelt werden sollen. Meistens sind diese Infobits zu Infowörtern gruppiert. Die Codierung fügt den Infobits Redundanz hinzu bzw. transformiert sie in eine Form, die Redundanz beinhaltet. Beispielsweise könnte eine Codierung daraus bestehen, dass jedes Bit des Infoworts dreifach genommen wird; aus dem Infowort $b = 01001$ entsteht so das Codewort c mit

$$c = 000111000000111. \tag{1}$$

Die ursprünglichen Bits müssen dabei nicht unbedingt so klar wie in diesem Beispiel in der Codefolge erscheinen. Bei der GSM-Übertragung ist eine Codierung mittels einer Faltung üblich, bei der benachbarte Bits miteinander gekoppelt werden. Eine derartige Faltung lässt sich durch ein Schieberegister darstellen (s. Abb. 3). Die Folge der Infobits wird sukzessive durch das Schieberegister geführt. In jedem Schritt werden die Bits an den markierten Stellen wie

abgebildet miteinander verknüpft. („+“ bedeutet dabei die Addition modulo 2, bei zwei Bits also eine XOR-Verknüpfung.)

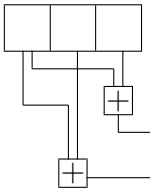

Abbildung 3: Ein Schieberegister

Im Beispiel handelt es sich um ein Schieberegister der Länge 3, bei der jeweils das erste und dritte bzw. das erste und zweite Bit XOR-verknüpft werden (s. Abb. 4). Die Ausgabebits werden zur Codefolge zusammengefasst.

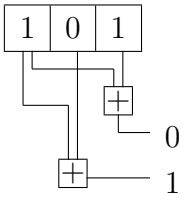

Abbildung 4: Ein Schieberegister mit Input (101) und Output (01)

Aus dem Infowort $b = 01001$ entsteht so die Codefolge $c = 11011011011000$ (s. Abb. 6; ggf. werden den Infobits Nullen vorangestellt bzw. hinzugefügt.)

Der GSM-Standard [1] legt beispielsweise für TCH/FS (*Traffic Channel, Full Speach Rate*) eine Faltung durch ein Schieberegister wie es in Abblidung 5 gezeigt ist, fest.

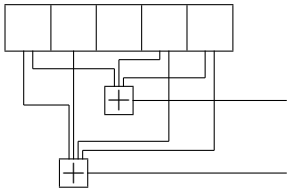

Abbildung 5: Schieberegister zum GSM-Standard TCH/FS

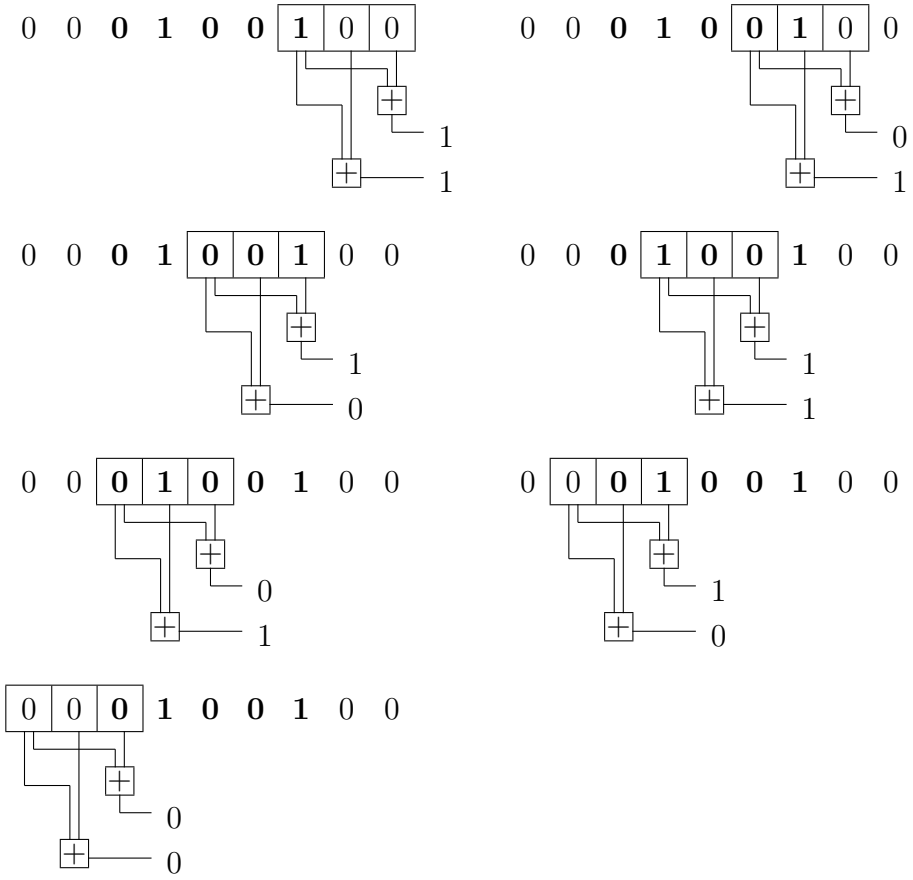

Abbildung 6: Anwendung der Faltung auf die Folge 01001

Interleaving

Wie oben schon angedeutet, gibt es unterschiedliche Arten der Störung. Einerseits gibt es zeitlich schnell schwankende Störungen (atmosphärisches Rauschen), andererseits Dämpfungen (z. B. durch Abschattungen oder Auslöschungen auf Grund von Überlagerungen), die länger andauern. Um bei einer Störung, die mehrere Bits umfasst, nicht sämtliche Informationen über einige Infobits zu verlieren (wie zum Beispiel beim Senden von $c = 000111000000111$ (s. (1)) und einer Störung im Bereich von Bit 5 bis 10 (von rechts gezählt), also einer Empfangsfolge 00011xxxxxx0111 passieren würde), permutiert man die Reihenfolge der Bits in der Codefolge über größere Distanzen. Bei einem Interleaving wie in Abbildung 7 entsteht so aus c die Folge 000100010001100010001

(vorne und hinten Nullen hinzugefügt).

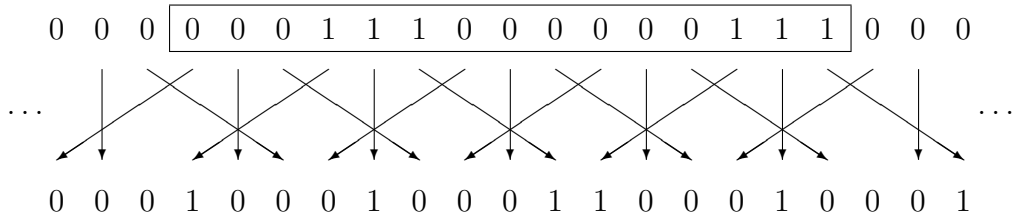

Abbildung 7: Interleaving

Man kann sich überlegen, dass hier sogar bei einer acht Bits umfassenden Störung noch immer das ursprüngliche Infobit rekonstruiert werden kann.

Modulation, Übertragung und Demodulation

Die Modulation legt fest, wie die Bits physikalisch repräsentiert und übertragen werden. Während der Übertragung kommen Störeinflüsse hinzu. Durch unterschiedliche lange Übertragungswege (s. Abb. 1) kann es nicht nur zu Auslöschungen sondern auch zu Überlagerungen kommen, die mehrere Bits betreffen. Die Aufgabe des Demodulators (Equalizers) ist, diese Überlagerungen rückgängig zu machen. Da das konkrete Zustandekommen der Überlagerungen von den äußeren Bedingungen abhängt und schwankt, sowie auf Grund der Störungen, ist nicht immer ein exaktes Zurückberechnen möglich. Der Equalizer gibt daher üblicherweise nicht „harte“ Bits aus, sondern versieht seine Ausgaben mit Vertrauenswerten. Statt einer Ausgabe von Werten aus der Menge $\{0, 1\}$ kann man dies beispielsweise durch Ausgabewerte im Intervall $[0, 1]$ realisieren. So kann beispielsweise eine Ausgabe $0,9$ für eine recht wahrscheinlich gesendete Eins stehen, 0.1 für eine recht wahrscheinliche Null. $0,6$ bedeutet eine unsichere Eins, $0,5$ keine Information.

Üblicherweise wird die Codefolge in einzelnen Bursts, d.h. in Blöcken einer bestimmten Anzahl von Bits, gesendet. Für aufeinanderfolgende Bursts benutzt man dann oft verschiedene Trägerfrequenzen („Frequency-Hopping“). Da eine starke Dämpfung bzw. Auslöschung durch ungünstige Pfad-Interferenzen bei festen Ausbreitungswegen von der benutzten Frequenz abhängt, hat dieses Frequency-Hopping den Vorteil, dass – falls bei einem Burst ungünstige Interferenzen vorliegen – der nächste Burst hoffentlich bessere Übertragungs-

qualitäten aufweist. Zusammen mit dem Interleaving und der eingestreuten Redundanz hofft man dann, trotz der Störungen auf das ursprüngliche Infowort zurückschließen zu können.

Deinterleaving

Die Werte, die der Demodulator liefert, werden entsprechend der umgekehrten Interleaving-Vorschrift zurück sortiert.

Decodierung

Bei der Decodierung wird versucht, möglichst geschickt die durch die Codierung eingestreute Redundanz zu benutzen, um auf das ursprüngliche Infowort zurückzuschließen. Bei festgelegter Codierung kann es mehrere Möglichkeiten der Decodierung geben. Ein Gerätehersteller hat also bei der Decodierung gewisse Freiheiten, auch wenn die Codierung durch einen Standard festgelegt ist. Liegt zum Beispiel eine Codierung wie oben bei (1) zu Grunde, die jedes Infobit verdreifacht, könnte man zur Decodierung die Ausgaben des Equalizers auf 0 bzw. 1 runden und nach Mehrheit entscheiden. Bei einem Empfang von 0.3 - 0.9 - 0.4 würde man so auf 0 - 1 - 0 runden und auf ein Infobit 0 entscheiden. Man könnte auch jeweils die Abstände zu den beiden möglichen Codefolgen 0 - 0 - 0 (beim Infobit 0) und 1 - 1 - 1 (beim Infobit 1) betrachten. Dabei ergibt sich ein Abstand

$$0.3 + 0.9 + 0.4 = 1.6 \quad \text{zu } 0 - 0 - 0$$

und

$$0.7 + 0.1 + 0.6 = 1.4 \quad \text{zu } 1 - 1 - 1,$$

so dass man bei Wahl des kleineren Abstands zu einer 1 decodieren würde. Tatsächlich zeigt sich, dass es unter bestimmten Voraussetzungen am besten ist (s. [6]), den quadratischen Abstand zu minimieren, also

$$0.3^2 + 0.9^2 + 0.4^2 = 1.06$$

versus

$$0.7^2 + 0.1^2 + 0.6^2 = 0.86.$$

In Kapitel 1 wird gezeigt, warum der quadratische Abstand unter gewissen Annahmen die geeignetste Größe zur Minimierung ist.

Der sogenannte Viterbi-Algorithmus (s. [15, 5]) bietet eine effiziente Decodierung für Faltungscodes auf Grundlage einer derartigen Minimierung. Der Viterbi-Algorithmus wird in Anhang B genauer beschrieben.

Die vorliegende Arbeit beschreibt eine im Vergleich zur bei GSM üblichen Vorgehensweise modifizierte Decodierung. Ein gegenüber der Standard-Kanalmodellierung verfeinertes Modell wird betrachtet, das die unterschiedlichen Übertragungsqualitäten bei verschiedenen Bursts berücksichtigt und geschätzte Qualitätswerte in die Decodierung einbezieht. Ähnliche Ansätze werden bei UMTS schon im inneren Empfänger, d.h. dem Pendant des Equalizers, benutzt (s. [2, 8]), während sie bei GSM nicht konsequent umgesetzt werden. Allerdings beruhen alle derartigen Ansätze auf der Schätzung der Übertragungsqualitäten auf Grund von Trainingssequenzen (Piloten), die den Codebits hinzugefügt werden, und die sowohl dem Sender als auch dem Empfänger bekannt sind. Im zweiten Teil dieser Arbeit wird gezeigt, wie man diese Schätzungen auf Basis der Datenbits durchführen und so verlässlichere Ergebnisse und damit eine verbesserte Decodierung erzielen kann.

Verwandte Ansätze, die allerdings beispielsweise auf Markov-Modellen basieren, kein Interleaving berücksichtigen oder gewisse Kenntnisse über den Kanal voraussetzen, gibt es in [9, 13, 17].

Es werden drei Verfahren vorgestellt, wie man in der Praxis diese Qualitätswerte aus den Datenbits schätzen und dann in die Decodierung einfließen lassen kann. Diese Verfahren sind unabhängig von dem konkret benutzten Faltungscode. Sie können ohne Schwierigkeit auf andere Codierungen übertragen werden. Die Darstellung legt ein (Faltungs-) Interleaving zu Grunde, das jeweils zwei benachbarte Codeblöcke miteinander vermischt. Allerdings spielt die konkrete Ausgestaltung des Interleavings keine entscheidende Rolle. Die praktische Umsetzung der Verfahren kann leicht an das jeweilige Interleaving angepasst werden.

Die hier vorgestellte Decodierung betrifft nur die Empfängerseite. Sie kann somit auf verschiedene schon bestehende Standards angewendet werden.

Die Verfahren wurden simulativ mit von Siemens ICM bereitgestellten Testdaten erprobt, wobei konkret eine Codierung, ein Interleaving und eine Übertragung gemäß [1, TCH/FS, Abschnitt 3.1] zu Grunde gelegt wurde. Die entspre-

chenden Ergebnisse zeigen Decodiergewinne von ca. 0,5dB bis zu über 1dB bei den betrachteten Szenarien.

Im Folgenden wird zunächst ein üblicher Codierungs-/Decodierungsablauf beschrieben. Die Kanalmodellierung, d.h. die Annahmen, die hier bzgl. der Störungen zu Grunde gelegt werden, werden vorgestellt und es wird gezeigt, wie sich die oben angedeutete Standard-Decodierung (Minimierung der quadratischen Abweichung) daraus herleitet. Im zweiten Kapitel wird das Modell verfeinert und die Auswirkungen auf die Decodierung dargestellt. Im dritten Kapitel werden drei Verfahren vorgestellt, wie die Kanalqualitätswerte, die man zur verbesserten Decodierung nach dem verfeinerten Modell benötigt, praktisch geschätzt und benutzt werden können. Das vierte Kapitel enthält die Simulationsergebnisse. Der Anhang A beinhaltet exemplarisch Kanalqualitätswerte. Im Anhang B wird der Viterbi-Algorithmus vorgestellt, Anhang C zeigt eine entsprechende Implementierung, die bei der Simulation benutzt wurde, und an der deutlich wird, dass das vorgeschlagene modifizierte Verfahren bei der Decodierung von Faltungscodes einen lediglich in der Metrik modifizierten Viterbi-Algorithmus benutzt.

Kapitel 1

Das Standard-Verfahren

1.1 Das Kanalmodell

Die Abbildung 1.1 gibt nochmals einen Überblick über einen Codierungs-, Sende- und Decodierungsablauf:

Durch die Codierung entstehen aus den Infowörtern Codewörter (im Mobilfunk üblicherweise durch eine Faltungscodierung). Durch die hinzugefügte Redundanz sind die Codewörter länger als die Infowörter. Die Bits der entstehenden Codewörter werden nach einem vorgegebenen Verfahren vermischt (Interleaving). Diese vermischten Bits werden dann burstweise übertragen. Bei dem der Simulation zu Grunde gelegten GSM-Standard wird jeweils ein Infowort zu einem Codewort bestehend aus 456 Bits codiert. Je zwei benachbarte Codewörter werden durch das Interleaving kreuzweise miteinander vermischt. Die entstehenden Blöcke werden in 4 Bursts zu je 114 Bits übertragen. Üblicherweise wird für jeden Burst eine andere Frequenz benutzt (Frequency-Hopping). Mit der Übertragung wird hier der Modulator, der Sender, der Funkkanal, der Empfänger und der Demodulator zusammengefasst. Ferner beschränkt sich die folgende Darstellung auf bipolare Bits. Auch wenn der dargestellte Rechenweg nicht direkt auf höherwertige Modulationsverfahren übertragen werden kann, liegt eine Verallgemeinerung auf solche Verfahren nahe. Bei der hier betrachteten Modellierung werden also die Codebits $c_k \in \{\pm 1\}$ dem Sender zugeführt. Am Empfänger kommen gestörte Werte y_k an, die dann in das Deinterleaving und die Decodierung einfließen. Die Störung wird dadurch modelliert,

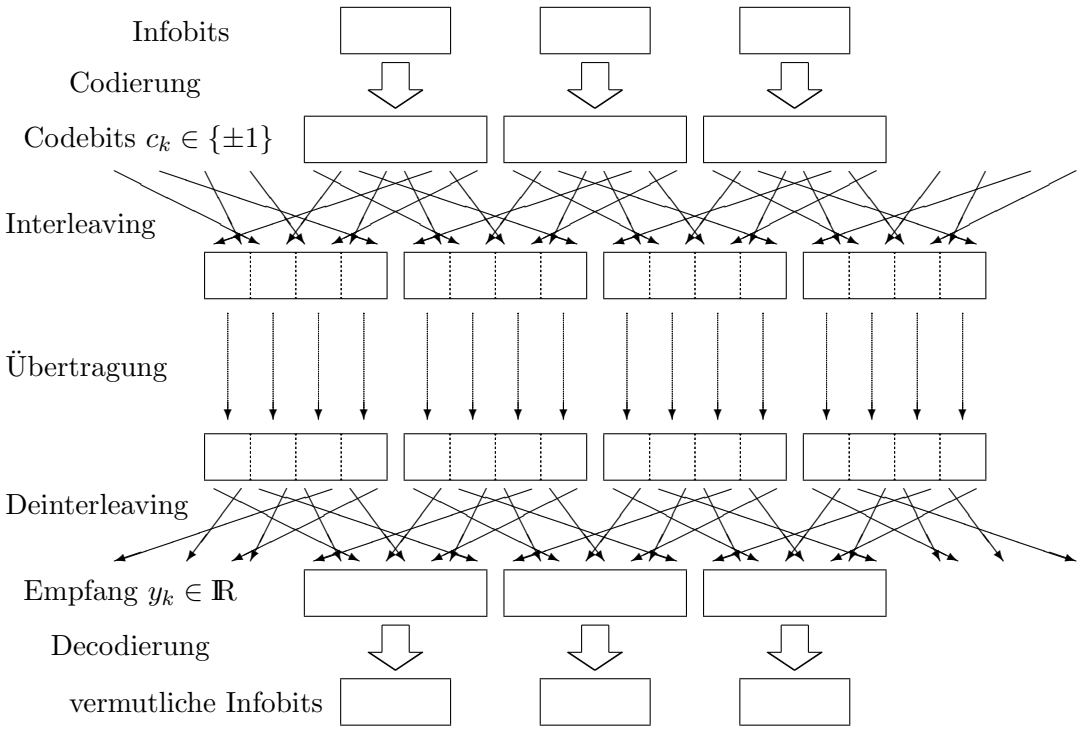

Abbildung 1.1: Übersicht über Codierung und Decodierung

dass nicht Werte $y_k \in \{\pm 1\}$ sondern Werte $y_k \in \mathbb{R}$ empfangen werden. Der Übergang von den c_k zu den y_k wird weiter unten in relativ einfacher Weise mathematisch modelliert, d.h., der komplexe physikalische Prozess der Modulation, des Sendens, der Störung, des Empfangs und der Demodulation werden vereinfacht beschrieben.

Im Folgenden wird ein Bit statt der 0/1-Darstellung stets in bipolarer Weise durch $+1$ oder -1 repräsentiert. Dies erleichtert die folgenden Formeln und Darstellungen. Tatsächlich kann man sich leicht überlegen, dass die Ergebnisse entsprechend auf höherstufige Codierverfahren übertragen werden können. Die Herleitung ist dann aber nicht ganz so übersichtlich wie im bipolaren Fall, weshalb hier im Weiteren nur dieser Fall ausgeführt ist. Während die Info- und Codebits also nur die Werte ± 1 haben können, treten nach der Übertragung störungsbedingt auch Zwischenwerte auf. Vom theoretischen Standpunkt aus kann man für die Empfangswerte y_k sämtliche reelle Zahlen zulassen. In der Praxis liefert der Demodulator allerdings nur einen endlichen Wertebereich:

Außerhalb eines gewissen Intervalls wird auf die Intervallgrenzen abgebildet, innerhalb wird auf festgelegte diskrete Werte gerundet. Die Decodierung benutzt diese sogenannten Softwerte, um die vermutlichen Infobits zu bestimmen.

Das Übertragungs- bzw. Kanalmodell beschreibt den Übergang von den Codebits $c_k \in \{\pm 1\}$ zu den Empfangswerten $y_k \in \mathbb{R}$. In der Realität ist dieser Übergang komplex; die Art der physikalischen Übertragung spielt hier eine Rolle genauso wie die Art der Störung auf der Luftschnittstelle sowie die Verarbeitung des Empfangssignals, s. [10]. Hier soll ein vereinfachtes Kanalmodell zu Grunde gelegt werden, das direkt von den gesendeten Codebits zu Empfangswerten übergeht. Dabei werden zwei Einflüsse modelliert: Einerseits wird die Stärke des Codebits reduziert; dies resultiert – wie in der Einleitung bereits beschrieben – beispielsweise aus unterschiedlichen Laufzeiten verschiedener Übertragungswege (Rayleigh-Fading) und kann mit einem Dämpfungsfaktor f_k modelliert werden. Andererseits stören zufällige Rauscheinflüsse, die man durch einen additiven Term r_k modellieren kann. Man erhält so für die Empfangswerte y_k:

$$y_k = f_k \cdot c_k + r_k.$$

(Zu entsprechenden Kanalmodellen s. [12, 4].)

Betrachtet man keine statistischen Bindungen der Störeinflüsse r_k, so kann man diese beispielsweise als unabhängig normalverteilt (mit Mittelwert 0) annehmen (AWGN-Kanal: *additive white gaussian noise*).

Wegen der unabhängig zu den r_k festgelegten $c_k \in \{\pm 1\}$ sind dann mit r_k auch die Werte $n_k := \frac{r_k}{c_k}$ unabhängig und normalverteilt. Damit ergibt sich für die Empfangswerte y_k:

$$\begin{aligned} y_k &= f_k \cdot c_k + r_k \\ &= \left(f_k + \tfrac{r_k}{c_k}\right) \cdot c_k \\ &= (f_k + n_k) \cdot c_k. \end{aligned}$$

Für das Folgende setzen wir:

$$\begin{aligned} q_k &:= f_k + n_k, \\ \tilde{c_k} &:= \operatorname{sign}(y_k), \\ a_k &:= |y_k|. \end{aligned}$$

Der Wert q_k beschreibt die Qualität des Kanals für das k-te Bit: $q_k = 1$ bedeutet eine ungestörte Übertragung ($y_k = c_k$). Kleinere positive Werte bedeuten gedämpfte / gestörte Übertragung, wobei aber das Vorzeichen von c_k noch erhalten bleibt. Durch große negative Störeinflüsse n_k kann q_k auch negativ werden. Dies bedeutet dann eine Bitumkehr.

Der Wert $\tilde{c_k} \in \{\pm 1\}$ beschreibt das vermutliche Codebit, auf das man allein bei Empfang von y_k schließen würde. Dabei gibt a_k einen Vertrauenswert für diese Entscheidung an: Bei $a_k = 1$ kann man relativ sicher sein, mit $\tilde{c_k}$ richtig zu liegen; bei kleinen Werten ist die Wahrscheinlichkeit für eine Fehlentscheidung des Vorzeichens größer.

Mit den eingeführten Werten kann man y_k darstellen als

$$y_k = q_k \cdot c_k = a_k \cdot \tilde{c_k}. \tag{1.1}$$

Dabei beschreibt die erste Darstellung $y_k = q_k \cdot c_k$ eine Kanalsicht: Der Empfang ergibt sich aus dem gesendeten Bit multipliziert mit dem Qualitätswert des Kanals. Die zweite Darstellung $y_k = a_k \cdot \tilde{c_k}$ spiegelt die Sicht auf der Empfängerseite wider: Vorzeichenentscheidung und Vertrauenswert.

Der Anhang A listet exemplarisch diskretisierte Kanalqualitätswerte auf, wie sie bei Siemens ICM zur Kanalsimulation verwendet werden. Man sieht, dass sich die einzelnen Bursts sehr unterschiedlich verhalten. In der Modellierung kann man das dadurch repräsentieren, dass man die Dämpfungsfaktoren f_k bei einem Burst als relativ konstant ansieht, während sie sich für unterschiedliche Bursts deutlich unterscheiden (können).

In einem einfachen Modell wird für die Decodierung angenommen, dass die q_k unabhängig identisch normalverteilt sind (mit einem Mittelwert μ und einer Standardabweichung σ). Tatsächlich erscheint dies vernünftig, denn q_k enthält mit n_k schon einen normalverteilten Summanden. Die Dämpfungswerte f_k schwanken zwar innerhalb eines Bursts wenig; aufeinanderfolgende Bits eines Codewortes werden allerdings durch das Interleaving auf viele verschiedene Bursts verteilt. Diese haben quasi zufällige unterschiedliche Dämpfungswerte, so dass der Ansatz der unabhängigen Normalverteilung gerechtfertigt ist. (Beim Standard TCH/FS (s. [1]), der den Simulationen und den Ergebnissen von Kapitel 4 zu Grunde liegt, werden jeweils acht aufeinanderfolgende Bits auf acht verschiedene Bursts verteilt.)

Die verfeinerte Kanalmodellierung, die im zweiten Kapitel beschrieben wird, unterscheidet sich von der obigen Modellierung genau in diesem Punkt: der Annahme einer identischen Normalverteilung.

1.2 Decodierung

Die übliche Art der Decodierung mittels „Soft-Decision“ kann man durch einen Maximum-Likelihood-Ansatz bzw. einem Maximum-a-posteriori-Ansatz mit der zusätzlichen Annahme von gleichwahrscheinlichen Eingabeworten b aus der Modellannahme der Normalverteilung der q_k herleiten. Hier wird eine Herleitung vorgeführt, um dann in Abschnitt 2.2 die Unterschiede bei der verfeinerten Kanalmodellierung deutlich zu machen.

Ausgangspunkt ist eine empfangene Folge $y = y_1 y_2 \dots y_K$. Bekanntermaßen führt ein Maximum-a-posteriori-Ansatz zu einer optimalen Decodierung. In Ermangelung von Informationen über die Wahrscheinlichkeit der Eingabeworte nimmt man diese als gleichwahrscheinlich an und erhält so einen Maximum-Likelihood-Ansatz: Zu einem Infowort b bzw. dem entsprechenden Codewort $c = c(b) = c_1 c_2 \dots c_K$ betrachtet man die Wahrscheinlichkeit, dass durch die Übertragung gerade die Empfangsfolge y entsteht. Man sucht dann das Infowort b, das mit größter Wahrscheinlichkeit zur Empfangsfolge passt. Genauer:

Wird c_k gesendet und y_k empfangen, so muss wegen $y_k = q_k \cdot c_k$ der Kanalqualitätswert q_k gerade gleich $\frac{y_k}{c_k}$ sein. Wegen $c_k \in \{\pm 1\}$ ist $\frac{y_k}{c_k} = y_k \cdot c_k$, also $q_k = y_k \cdot c_k$.

Wegen der angenommenen (μ, σ^2)-Normalverteilung ergibt sich als zugehöriger Dichtewert $p(q_k)$ mit

$$p(q_k) = \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-\frac{(\mu - q_k)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-\frac{(\mu - y_k c_k)^2}{2\sigma^2}}.$$

Damit ergibt sich als Produkt $P(c, y)$ der Dichten zu einem Codewort $c = c_1 c_2 \dots c_K$ und einer Empfangsfolge $y = y_1 y_2 \dots y_K$

$$\begin{aligned} P(c, y) &= \prod_{k=1}^{K} \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-\frac{(\mu - y_k c_k)^2}{2\sigma^2}} \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma} \right)^K \mathrm{e}^{-\sum_{k=1}^{K} \frac{(\mu - y_k c_k)^2}{2\sigma^2}}. \end{aligned}$$

Gesucht ist nun das Infowort $\hat{b}$ bzw. das Codewort $\hat{c} = c(\hat{b}) = \hat{c_1} \hat{c_2} \dots \hat{c_K}$, bei dem $P(c, y)$ maximal ist:

$$\hat{c} = \arg \max_{c \in \mathrm{Code}} P(c, y).$$

(Diese Maximierungsaufgabe erhält man ebenso über den häufig dargestellten Zugang mittels bedingter Wahrscheinlichkeiten.)

Da der Faktor $\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^K$ unabhängig von den c_k ist, braucht er nicht berücksichtigt zu werden:

$$\begin{aligned}\hat{c} &= \arg\max_{c\in\text{Code}} \left[\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^K \mathrm{e}^{-\sum_{k=1}^{K}\frac{(\mu-y_k c_k)^2}{2\sigma^2}}\right] \\ &= \arg\max_{c\in\text{Code}} \left[\mathrm{e}^{-\sum_{k=1}^{K}\frac{(\mu-y_k c_k)^2}{2\sigma^2}}\right].\end{aligned}$$

Die Maximierung der Exponentialfunktion ist nun äquivalent zur Maximierung des Arguments:

$$\hat{c} = \arg\max_{c\in\text{Code}} -\sum_{k=1}^{K}\frac{(\mu-y_k c_k)^2}{2\sigma^2}.$$

Wegen des negativen Vorzeichens führt dies zur Minimierungsaufgabe

$$\begin{aligned}\hat{c} &= \arg\min_{c\in\text{Code}} \sum_{k=1}^{K}\frac{(\mu-y_k c_k)^2}{2\sigma^2} \\ &= \arg\min_{c\in\text{Code}} \sum_{k=1}^{K}\left(\frac{\mu^2}{2\sigma^2} - \frac{2\mu\cdot y_k c_k}{2\sigma^2} + \frac{{y_k}^2{c_k}^2}{2\sigma^2}\right).\end{aligned}$$

Der erste Summand $\frac{\mu^2}{2\sigma^2}$ ist nicht abhängig von c_k, spielt also für die Minimierung keine Rolle. Dies gilt auch für den letzten Summanden $\frac{{y_k}^2{c_k}^2}{2\sigma^2}$, denn da c_k nur die Werte ± 1 annehmen kann, ist immer ${c_k}^2 = 1$. Die Decodierung reduziert sich also wie folgt:

$$\begin{aligned}\hat{c} &= \arg\min_{c\in\text{Code}} \sum_{k=1}^{K}\left(\frac{\mu^2}{2\sigma^2} - \frac{2\mu\cdot y_k c_k}{2\sigma^2} + \frac{{y_k}^2{c_k}^2}{2\sigma^2}\right) \\ &= \arg\min_{c\in\text{Code}} \sum_{k=1}^{K} -\frac{2\mu\cdot y_k c_k}{2\sigma^2} \\ &= \arg\min_{c\in\text{Code}} -\sum_{k=1}^{K}\frac{\mu}{\sigma^2} y_k c_k.\end{aligned}$$

Hier kann der (konstante) Faktor $\frac{\mu}{\sigma^2}$ weggelassen werden. Dabei geht man davon aus, dass für den Mittelwert μ der Normalverteilung gilt: $\mu > 0$. Dies ist plausibel, da eine Bitumkehr ($q_k < 0$) weniger wahrscheinlich sein sollte

als eine wenigstens ansatzweise richtige Übertragung ($q_k > 0$), so dass sich im Erwartungswert ein positiver Wert ergeben sollte. Wegen des negativen Vorzeichens erhält man also schließlich:

$$\hat{c} = \arg \max_{c \in \text{Code}} \sum_{k=1}^{K} y_k c_k. \tag{1.2}$$

Mit (1.1) lässt sich die zu maximierende Summe auch darstellen als

$$\sum_{k=1}^{K} y_k c_k = \sum_{k=1}^{K} a_k \tilde{c_k} c_k.$$

Diese letzte Formel ist auch anschaulich verständlich: Die Summanden $a_k \tilde{c_k} c_k$ sind immer dann positiv, wenn $\tilde{c_k}$ und c_k das gleiche Vorzeichen haben. Wenn es aber kein Codewort $c = c_1 c_2 \ldots c_K$ gibt, dessen Vorzeichen bei jedem k mit dem Vorzeichen $\tilde{c_k}$ der empfangenen Folge übereinstimmt, sollte man bei den Stellen k die Konflikte austragen, an denen die Vertrauenswerte a_k möglichst klein sind.

Man kann durch eine ähnliche Betrachtung zeigen, dass die Maximierung (1.2) genau der Minimierung der Summe der quadratischen Abstände $(y_k - c_k)^2$ entspricht, wie es in der Einleitung erwähnt wurde:

$$\begin{aligned}
& \arg \min_{c \in \text{Code}} \sum_{k=1}^{K} (y_k - c_k)^2 \\
= \; & \arg \min_{c \in \text{Code}} \sum_{k=1}^{K} \left({y_k}^2 - 2 y_k c_k + {c_k}^2 \right) \\
= \; & \arg \min_{c \in \text{Code}} \sum_{k=1}^{K} -2 y_k c_k \qquad \begin{array}{l}(\text{da } {c_k}^2 = 1 \text{ und } y_k \text{ unabhängig} \\ \text{vom gewählten Codewort ist})\end{array} \\
= \; & \arg \max_{c \in \text{Code}} \sum_{k=1}^{K} y_k c_k \\
= \; & \hat{c}.
\end{aligned}$$

Bei einem Faltungscode geschieht eine entsprechende Decodierung effizient mittels des Viterbi-Algorithmus (s. [15, 5]). Dabei brauchen nicht alle möglichen Codewörter durchprobiert und die entsprechenden Summen (1.2) berechnet zu werden, sondern es werden sukzessive Infobit-Folgen bestimmt, die potentiell einen maximalen Wert der Summe (1.2) liefern können. Bitfolgen, die

dies garantiert nicht erreichen, werden so früh wie möglich ausgeblendet. Anhang B beinhaltet eine genaue Beschreibung des Algorithmus. Dabei sieht man, dass der Algorithmus angewendet werden kann, auch wenn die zu maximierende Summe aus anders geformten Summanden gebildet wird. Wählt man z.B. statt der Summe $S_1(c,y) = \sum_{k=1}^{K} y_k c_k$ in (1.2) die Summe $S_2(c,y)$ mit

$$S_2(c,y) = \sum_{k=1}^{K} \delta(y_k, c_k)$$

mit

$$\delta(y_k, c_k) = \begin{cases} 1 & \text{, falls } y_k \text{ und } c_k \text{ gleiches Vorzeichen haben,} \\ 0 & \text{, falls } y_k \text{ und } c_k \text{ unterschiedliches Vorzeichen haben,} \end{cases}$$

zählt also einfach die Stellen mit übereinstimmendem Vorzeichen, so erhält man den sog. Hard-Decision-Viterbi-Decodierer.

Kapitel 2

Modifizierter Ansatz

2.1 Verfeinertes Kanalmodell

In Abschnitt 1.1 wurden die Übertragungseigenschaften durch

$$y_k = f_k \cdot c_k + r_k = (f_k + n_k) \cdot c_k$$

modelliert. Dabei beschreibt f_k einen (über einen Burst eher langsam schwankenden) Dämpfungsfaktor und r_k bzw. n_k eine zufällige Störung (s. auch Anhang A). Die in Abschnitt 1.3 beschriebene Decodierung basiert auf der Annahme, dass die Kanalqualitätswerte $q_k = f_k + n_k$ unabhängig normalverteilt sind. Wegen der Verteilung aufeinander folgender Bits des Codeworts auf verschiedene Bursts (durch das Interleaving) und somit der Einflussnahme verschiedener Dämpfungsfaktoren war diese Annahme einigermaßen plausibel. Dies entspricht soweit dem Stand der Technik und der üblichen Decodierung bei GSM.

In einem verfeinerten Ansatz wird nun berücksichtigt, dass die f_k innerhalb eines Bursts ungefähr konstant sind, zwischen verschiedenen Bursts aber differieren. In [3, S. 49] wird beispielsweise vorgerechnet, dass bei einem fahrenden Auto Signaleinbrüche der Dauer 1,25 ms bei einer bestimten Frequenz auftreten können. Demgegenüber dauert die Übertragung eines Bursts nur 0,577 ms (s. [3, S. 341]).

Diese verfeinerte Modellierung kann man dadurch ausdrücken, dass man die Werte q_k als normalverteilt mit *Burst-abhängigem* Mittelwert μ_{Burst} annimmt.

Denn hat man $f_k \approx f_{\text{Burst}}$, so ergibt sich $q_k \approx f_{\text{Burst}} + n_k$, und da n_k als normalverteilt mit Mittelwert 0 angenommen wird, ergibt sich f_{Burst} als Erwartungswert der q_k über einen Burst. Da auch die Rauscheinflüsse (insbesondere deren Varianz) von der verwendeten Frequenz, also dem Burst, abhängen könnten, wird auch die Standardabweichung σ der n_k und damit der q_k als Burst-abhängig angenommen.

Bei dem verfeinerten Kanalmodell wird also davon ausgegangen, dass die Kanalqualitätswerte q_k unabhängig normalverteilt sind mit Burst-abhängigem Mittelwert μ_{Burst} und Burst-abhängiger Standardabweichung σ_{Burst}.

2.2 Modifizierte Decodierung

Ausgehend von der modifizierten Annahme der Burst-abhängigen Normalverteilung der Kanalqualitätswerte q_k sind nun die Rechnungen in Abschnitt 1.2 auf Änderungen hin zu untersuchen:

Bei entsprechendem Maximum-Likelihood-Ansatz hat man formelmäßig das gleiche Produkt

$$P(c,y) = \prod_{k=1}^{K} \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-\frac{(\mu - y_k c_k)^2}{2\sigma^2}}$$

zu maximieren, genauer das Infowort $\hat{b}$ bzw. das Codewort $\hat{c} = c(\hat{b}) = \hat{c_1}\hat{c_2}\ldots\hat{c_K}$ zu suchen, bei dem P maximal ist:

$$\hat{c} = \arg\max_{c \in \mathrm{Code}} P(c,y).$$

Dabei ist zu beachten, dass μ und σ nun von dem Burst abhängen, mit dem das k-te Bit gesendet wird.

Die Faktoren $\frac{1}{\sqrt{2\pi}\sigma}$ sind wieder von den c_k unabhängig (sie können zwischen den Bursts variieren, haben aber einen Wert, der unabhängig von dem zur Auswertung gewählten Codewort c ist), so dass nach entsprechender Umformung der Exponentialfunktion bei der Bestimmung von $\hat{c}$ die Maximierung auf das Argument der Exponentialfunktion bezogen werden kann:

$$\begin{aligned} \hat{c} &= \arg\max_{c \in \mathrm{Code}} -\sum_{k=1}^{K} \frac{(\mu_{\mathrm{Burst}(k)} - y_k c_k)^2}{2{\sigma_{\mathrm{Burst}(k)}}^2} \\ &= \arg\min_{c \in \mathrm{Code}} \sum_{k=1}^{K} \frac{(\mu_{\mathrm{Burst}(k)} - y_k c_k)^2}{2{\sigma_{\mathrm{Burst}(k)}}^2}. \end{aligned}$$

Nach Ausquadrieren des Zählers erhält man

$$\hat{c} = \arg\min_{c \in \mathrm{Code}} \sum_{k=1}^{K} \left(\frac{\mu^2_{\mathrm{Burst}(k)}}{2{\sigma_{\mathrm{Burst}(k)}}^2} - \frac{2\mu_{\mathrm{Burst}(k)} \cdot y_k c_k}{2{\sigma_{\mathrm{Burst}(k)}}^2} + \frac{{y_k}^2 {c_k}^2}{2{\sigma_{\mathrm{Burst}(k)}}^2} \right).$$

Hier sind wieder die beiden entstehenden quadratischen Terme $\frac{\mu^2_{\mathrm{Burst}(k)}}{2{\sigma_{\mathrm{Burst}(k)}}^2}$ und $\frac{{y_k}^2 {c_k}^2}{2{\sigma_{\mathrm{Burst}(k)}}^2}$ unabhängig von den c_k, so dass sie bei der Minimierung nicht beachtet zu werden brauchen. So bleibt nur der gemischte Term:

$$\hat{c} = \arg\min_{c \in \mathrm{Code}} \sum_{k=1}^{K} -\frac{\mu_{\mathrm{Burst}(k)} \cdot y_k c_k}{{\sigma_{\mathrm{Burst}(k)}}^2}.$$

Bei universalem μ und σ konnte man den Faktor $\frac{\mu}{\sigma^2}$ für die Minimierung weglassen. Hier ist er aber nun k-abhängig und mit c_k verbunden, so dass er weiterhin berücksichtigt werden muss. Unter Beachtung des negativen Vorzeichens erhält man also:

$$\hat{c} = \arg \max_{c \in \text{Code}} \sum_{k=1}^{K} \frac{\mu_{\text{Burst}(k)}}{{\sigma_{\text{Burst}(k)}}^2} y_k c_k = \sum_{k=1}^{K} \frac{\mu_{\text{Burst}(k)}}{{\sigma_{\text{Burst}(k)}}^2} a_k \tilde{c_k} c_k. \tag{2.1}$$

Vorzeichenkonflikte zwischen $\tilde{c_k}$ und c_k sollten also vorrangig an den Stellen ausgetragen werden, an denen einerseits die Vertrauenswerte a_k möglichst klein sind und andererseits auch die Burstqualität relativ schlecht ist (kleiner Mittelwert, große Varianz).

Im Zusammenhang mit UMTS treten Ausdrücke, wie sie in (2.1) vorkommen, insbesondere die Gewichtungsfaktoren $\frac{\mu_{\text{Burst}(k)}}{{\sigma_{\text{Burst}(k)}}^2}$, im inneren Decoder auf (s. [2, 8]). Sie stehen dabei aber im engen Zusammenhang mit der UMTS-eigenen Übertragungseigenschaft der sehr häufigen Anpassung der Sendeleistung („Fast Power Control“).

Beispiel

Betrachtet wird wie in der Einleitung eine Codierung, die jedes Bit verdreifacht. Aus dem Infowort $b = b_1|\, b_2|\, b_3|\, b_4 = +1|-1|-1|+1$ entsteht also

$$c = +1|+1|+1|-1|-1|-1|-1|-1|-1|+1|+1|+1.$$

Als Interleaving wird das in der Einleitung erwähnte kreuzweise Interleaving benutzt (mit auffüllenden „-1“), s. Abb. 2.1

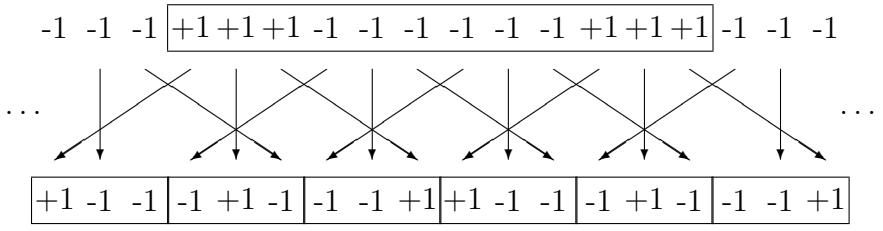

Abbildung 2.1: Interleaving

Die Bits werden in Bursts der Länge 3 gesendet. Für die entsprechenden Mittelwerte gelte

$$\mu_{\text{Burst 1}} = 0.1 \qquad \mu_{\text{Burst 2}} = 0.8 \qquad \mu_{\text{Burst 3}} = 0.3$$

$$\mu_{\text{Burst 4}} = 0.1 \qquad \mu_{\text{Burst 5}} = 0.4 \qquad \mu_{\text{Burst 6}} = 0.5.$$

Die Varianzen werden einheitlich als 1 angenommen. Beispielhafte Kanalqualitätswerte für die einzelnen Bursts sind

Burst 1: - 0.5 | 0.2 | 0.4	Burst 2: 0.8 | 0.7 | 0.8
Burst 3: 0.9 | - 0.1 | - 0.3	Burst 4: 0.6 | - 0.1 | - 0.4
Burst 5: 0.5 | - 0.1 | 0.6	Burst 6: 0.7 | 0.3 | 0.5

Abbildung 2.2 zeigt die empfangene Folge sowie die zurücksortierte Folge (nach dem Deinterleaving).

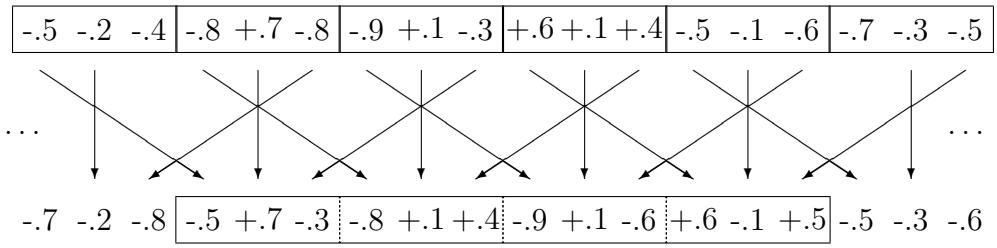

Abbildung 2.2: Empfangene und zurücksortierte Folge

Da in dem Beispiel die einzelnen Infobits durch die Codierung nicht gekoppelt sind, kann man bei der Decodierung die einzelnen zu einem Infobit gehörigen Dreier-Blöcke separat untersuchen. Bei einer Hard-Decision-Decodierung (Beachtung allein des Vorzeichens und dann Mehrheitsentscheidung) würde man auf das Infowort $-1|+1|-1|+1$ schließen, also bei zwei Bits falsch entscheiden.

Betrachtet man die Standard-„Soft"-Metrik (s. 1.2), ergeben sich als entsprechende Summenwerte für den ersten Dreier-Block links

- bei $b_1 = +1$, also einem Codeblock $+1|+1|+1$:

$$(-\,0.5)\cdot(+1) \;+\; (+0.7)\cdot(+1) \;+\; (-\,0.3)\cdot(+1) = -0.1$$

- bei $b_1 = -\,1$, also einem Codeblock $-\,1|-\,1|-\,1$:

$$(-\,0.5)\cdot(-\,1) \;+\; (+0.7)\cdot(-\,1) \;+\; (-\,0.3)\cdot(-\,1) = 0.1.$$

Den maximalen Wert erhält man also auch hier bei der (falschen) Entscheidung zu $b_1 = -1$.

Für den zweiten Dreier-Block erhält man

- bei $b_2 = +1$, also einem Codeblock $+1|+1|+1$:

$$(\text{-}\ 0.8)\cdot(+1)\ +\ (+0.1)\cdot(+1)\ +\ (+0.4)\cdot(+1) = -0.3$$

- bei $b_2 = \text{-}\ 1$, also einem Codeblock - 1|- 1|- 1:

$$(\text{-}\ 0.8)\cdot(\text{-}\ 1)\ +\ (+0.1)\cdot(\text{-}\ 1)\ +\ (+0.4)\cdot(\text{-}\ 1) = 0.3,$$

so dass man (richtig) zu $b_2 = -1$ decodiert. Für die beiden letzten Blöcke rechts erhält man als Summenwerte

	bei $b_k = +1$	bei $b_k = -1$
Codebits 7 bis 9	-1.4	$+1.4$
Codebits 10 bis 12	$+1.0$	-1.0

Maximalen Wert erhält man also richtigerweise bei $b_3 = +1$ und $b_4 = -1$.

Bei der modifizierten Metrik (2.1) wird der entsprechende Mittelwert μ_{Burst} mit in die Decodierung einbezogen. Abbildung 2.3 zeigt zu jedem Bit den Mittelwert des Bursts, mit dem das Bit übertragen wurde.

-.5	+.7	-.3	-.8	+.1	+.4	-.9	+.1	-.6	+.6	-.1	+.5
0.1	0.8	0.3	0.8	0.3	0.1	0.3	0.1	0.4	0.1	0.4	0.5

Abbildung 2.3: Übertragene Bits und zugehörige Burst-Mittelwerte

Als entsprechend gewichtete Summe ergibt sich nun für den ersten Block

- bei $b_1 = +1$, also einem Codeblock $+1|+1|+1$:

$$0.1\cdot(\text{-}\ 0.5)\cdot(+1)\ +\ 0.8\cdot(+0.7)\cdot(+1)\ +\ 0.3\cdot(\text{-}\ 0.3)\cdot(+1) = 0.42$$

- bei $b_1 = \text{-}\ 1$, also einem Codeblock - 1|- 1|- 1:

$$0.1\cdot(\text{-}\ 0.5)\cdot(\text{-}\ 1)\ +\ 0.8\cdot(+0.7)\cdot(\text{-}\ 1)\ +\ 0.3\cdot(\text{-}\ 0.3)\cdot(\text{-}\ 1) = -0.42.$$

Man entscheidet also richtig zu $b_1 = +1$, da die schlechten Kanalqualitätswerte -0.5 bzw. -0.3 (Bitkipper) mit den entsprechend geringen Burstmittelwerten 0.1 bzw. 0.3 gewichtet werden, während der positive Kanalqualitätswert 0.7 stärker in die Entscheidung einbezogen wird.

Für die anderen Bits ergibt sich auch jeweils die richtige Entscheidung:

	bei $b_k = +1$	bei $b_k = -1$
Codebits 4 bis 6	-0.57	$+0.57$
Codebits 7 bis 9	-0.5	$+0.5$
Codebits 10 bis 12	$+0.27$	-0.27

Der in diesem Kapitel beschriebene Ansatz ist für jede Art von Codierung gültig, falls bei der Übertragung qualitativ verschiedene Kanäle benutzt werden.

Im Mobilfunk werden meistens Faltungscodes benutzt. Wie bereits Ende des letzten Kapitels erwähnt, können derartige Codes effizient mittels des Viterbi-Algorithmus decodiert werden. Im letzten Kapitel wurde auch schon darauf hingewiesen, dass es beim Viterbi-Algorithmus nicht auf die konkrete Form der Metrik, d.h. der Form der Summanden der Zielfunktion, ankommt (s. auch Anhang B). Man kann also auch die oben auftretenden Faktoren $\frac{\mu_{\text{Burst}(k)}}{\sigma_{\text{Burst}(k)}{}^2}$ mit berücksichtigen und Summanden wie bei (2.1) zu Grunde legen. So lässt sich - bei bekannten Werten $\mu_{\text{Burst}(k)}$ und $\sigma_{\text{Burst}(k)}$ - mit dem Viterbi-Algorithmus auch das Maximierungsproblem (2.1) effizient lösen. Allerdings sind für den Empfänger die Werte $\mu_{\text{Burst}(k)}$ und $\sigma_{\text{Burst}(k)}$ nicht bekannt und müssen geschätzt werden. Dies stellt eine ernsthafte Herausforderung dar. In der Praxis werden innerhalb eines Bursts oft auch Trainingssequenzen gesendet, die keine Information enthalten, und die dem Empfänger bekannt sind. Diese Trainingssequenzen dienen zur Synchronisation und zur Kanalschätzung im Equalizer. Üblicherweise werden diese Kanalschätzungen aber bei der Decodierung nicht weiter benutzt. Eine Schätzung von μ_{Burst} und σ_{Burst} könnte nun in der Praxis unter Berücksichtigung dieser Trainingssequenzen durchgeführt werden.

Im folgenden Kapitel wird beschrieben, wie man auch ganz ohne Trainingssequenzen, allein auf der Grundlage der empfangenen (Informations-)Daten Schätzungen von μ_{Burst} und σ_{Burst} durchführen kann.

Kapitel 3

Praktische Anwendung

Das grundsätzliche Problem, das sich bei der Auswertung des Maximierungsproblems (2.1) stellt, ist, dass für den Empfänger die Burst-abhängigen Mittelwerte und Varianzen $\mu_{\text{Burst}(k)}$ und $\sigma_{\text{Burst}(k)}$ nicht bekannt sind. Sie müssen also geschätzt werden. Für Mittelwert und Standardabweichung der (als normalverteilt angenommenen) Kanalqualitätswerte q_k hat man die üblichen erwartungstreuen Schätzer

$$\hat{\mu}_{\text{Burst}} = \frac{1}{|\text{Burst}|} \sum_{l \in \text{Burst}} q_l \tag{3.1}$$

und

$$\hat{\sigma}^2_{\text{Burst}} = \frac{1}{|\text{Burst}| - 1} \sum_{l \in \text{Burst}} (q_l - \hat{\mu}_{\text{Burst}})^2. \tag{3.2}$$

(Die Summanden erstrecken sich dabei über die zu einem Burst gehörigen Bits bzw. deren Kanalqualitätswerte und |Burst| bezeichnet die Anzahl der Bits innerhalb eines Bursts.)

Bei den Schätzern (3.1) und (3.2) sind weiterhin die Kanalqualitätswerte q_l auf der Empfangsseite nicht bekannt. Erinnert sei an dieser Stelle nochmals an die beiden Darstellungen des Empfangssignals y_k gemäß (1.1)

$$y_k = q_k \cdot c_k = a_k \cdot \tilde{c_k},$$

wobei die erste Darstellung $y_k = q_k \cdot c_k$ eine Kanalsicht darstellt (gesendetes Bit mal Qualitätswert) und die zweite ($y_k = a_k \cdot \tilde{c_k}$) eine Empfangssicht (Vorzei-

chenentscheidung und Vertrauenswert). Allein aus der Kenntnis von y_k kann man q_k nicht bestimmen.

Um diese Schwierigkeiten zu umgehen, könnte man in der Praxis Trainingsbits verwenden: In einem Burst werden neben den Datenbits weitere fest vorgegebene Bits gesendet. Auf der Empfangsseite kennt man diese Bits c_k, so dass man mittels der Darstellung

$$y_k = q_k \cdot c_k$$

aus den Empfangsdaten y_k die Kanalqualitätswerte als

$$q_k = \frac{y_k}{c_k}$$

berechnen kann. Tatsächlich werden diese Piloten in der Praxis nur im Equalizer benutzt, eine Einflussnahme bei der Decodierung geschieht üblicherweise nicht.

Im Folgenden werden nun Verfahren vorgestellt, auch ohne Trainingsbits, allein auf der Basis der Datenbits, die Kanalwerte μ_{Burst} und σ_{Burst} zu schätzen. Auch bei Vorhandensein von Trainingsbits können diese Verfahren gewinnbringend eingesetzt werden, denn sie erweitern den Stichprobenumfang, auf dessen Basis die Schätzung von μ_{Burst} und σ_{Burst} basiert. (Beim GSM Standard [1, TCH/FS] umfasst ein Burst beispielsweise 114 Datenbits und nur 26 Trainingsbits.)

In der Hoffnung, nur selten einen negativen Kanalqualitätswert q_l (also eine Bitumkehr bei der Vorzeichenbetrachtung) zu haben, liegt ein Ansatz zur Schätzung von μ_{Burst} und σ_{Burst} nahe, der statt der Kanalqualitätswerte q_l die Vertrauenswerte a_l verwendet, denn bei $\tilde{c}_l = c_l$ folgt $q_l = a_l$ (s. (1.1)). Simulationen zeigen, dass sich bei einer Schätzung von μ_{Burst} (σ_{Burst} als konstant angenommen) auf diese Weise bei dem simulierten Szenario kaum Decodiergewinne erzielen lassen. Bei einer zusätzlichen Schätzung von σ_{Burst} gemäß der Formel (3.2) mit q_l ersetzt durch a_l treten sogar Verschlechterungen auf. Dies zeigt, dass man bei Schätzungen vorsichtig vorzugehen hat.

Ein grundsätzliches Verfahren, tatsächlich signifikante Decodiergewinne zu realisieren, ist in Abbildung 3.1 schematisch beschrieben.

Zentrales Element ist dabei die Recodierung von bereits decodierten Wörtern. Durch diese Recodierung erhält man erste Schätzungen ${c_k}^{(1)}$ der tatsächlichen c_k; damit lassen sich dann aus $y_k = q_k \cdot c_k$ die Kanal-Qualitätswerte ${q_k}^{(1)} = \frac{y_k}{{c_k}^{(1)}}$

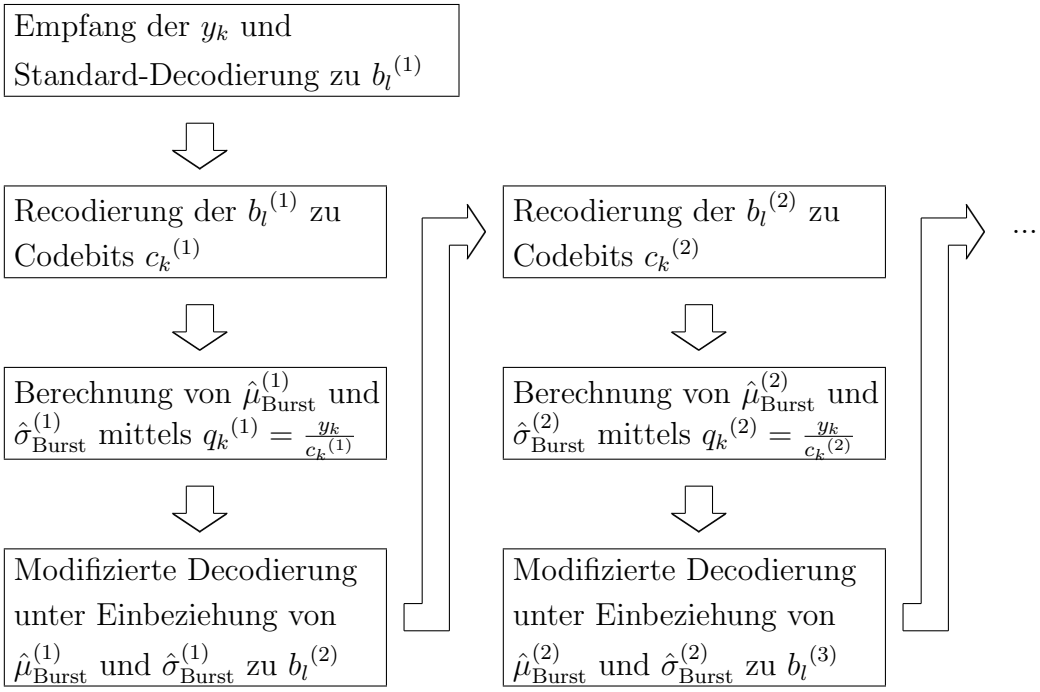

Abbildung 3.1: Kanalschätzung durch Recodierung (ggf. iteriert)

abschätzen. Mit diesen Werten ${q_k}^{(1)}$ berechnet man dann gemäß der obigen Formeln (3.1) bzw. (3.2) die Werte $\hat{\mu}_{\text{Burst}}$ bzw. $\hat{\sigma}_{\text{Burst}}$ und kann dann mit dem modifizierten Algorithmus verbessert decodieren. Ggf. kann man dieses Vorgehen iterieren: Das verbesserte Decodierungsergebnis wird wieder recodiert und dient dann als Grundlage einer verbesserten Schätzung der q_k die wiederum zu einer verbesserten Abschätzung von μ_{Burst} und σ_{Burst} Burst führen u.s.w..

Unter dem Stichwort „decision-feedback“ versteht man das allgemeine Konzept des Zurückfließens von Informationen aus einer bereits durchgeführten Decodierung in den aktuellen Decodierungsprozess. In [9] wird ein solches Konzept beispielsweise verknüpft mit einer Modellierung beruhend auf Markov-Ketten: Die Menge aller Fading-Level (sie entsprechen den hier mit f_k bezeichneten Werten) wird in eine endliche Anzahl von Mengen S_l zerlegt. Die (bedingten) Wahrscheinlichkeiten für $f_k \in S_l$ werden rekursiv unter Zuhilfenahme der empfangenen Werte und (beim „Decision-Feedback Decoder“, [9], Abschnitt 3.1) der recodierten Information berechnet; allerdings wird dazu ein Übergangsmodell für die einzelnen Mengen S_l oder zumindest Informationen über

die stationäre Verteilung des Kanal-Fadings benötigt (s. die Verwendung von $p_\alpha(\gamma)$ in den Gleichungen (5), (7) und (8) aus [9]). Die benutzte Viterbi-Metrik mittelt unter Einbeziehung dieser Wahrscheinlichkeiten über die Mengen S_l die bedingten Wahrscheinlichkeiten, dass y_k empfangen wurde, wenn c_k gesendet wurde und $f_k \in S_l$ ist.

Der Ansatz ist also mit dem hier vorgestellten verwandt, unterscheidet sich aber doch in einigen Punkten: Dem kontinuierlichen Vorgehen in [9] (zu jedem f_k werden Abschätzungen gemacht) steht die blockweise Behandlung der Bursts wie oben gegenüber. Der Ansatz in [9] nutzt das Gedächtnis innerhalb eines Bursts aus, beachtet aber nicht die unterschiedlichen Übertragungsqualitäten der Bursts bei Einsatz von Frequency Hopping. Letzteres ist gerade der wesentliche Punkt der vorliegenden Arbeit. Schließlich sind die Berechnungen in [9] genauer, wenn man die zugrunde liegenden Informationen über den betrachteten Markov-Prozess besitzt. Der obige Ansatz respektiert das Gedächtnis nur in den beiden Größen μ_{Burst} und σ_{Burst}, ist in dieser Hinsicht also gröber, aber dafür ohne Vorwissen zu berechnen.

In den folgenden Abschnitten werden drei Verfahren vorgestellt, wie man ohne Vorwissen unter Einbeziehung des Interleavings ein decision-feedback-Konzept realisieren kann. Die Verfahren werden an Hand eines Interleavings vorgestellt, wie es beispielsweise beim GSM-Standard TCH/FS verwendet wird. Charakteristisch ist dabei, dass jeweils zwei benachbarte Codewörter verschränkt werden und dann burstweise übertragen werden, so wie es in Abbildung 1.1 und in den folgenden Abbildungen angedeutet ist. (Zu Details des Interleavings beim GSM Standard TCH/FS siehe [1].) Die Verfahren lassen sich aber ohne Schwierigkeit auch auf andere Formen des Interleavings übertragen.

3.1 Iterierte vollständige Kanalschätzung

An Abbildung 3.2 wird illustriert und im Folgenden erläutert, wie man vorgehen kann, falls man alle Bursts, die bei der Übertragung eines Codewortes beteiligt sind, vollständig schätzen will. In dieser und den folgenden Abbildungen stehen dabei die Striche innerhalb der übertragenen Wörter bzw. der Codewörter für einzelne Bits und ihre Positionen vor/nach dem Deinterleaving. Ferner bezeichnen c_1, b_1, ... hier und im Folgenden immer ganze Code- bzw. Infowörter.)

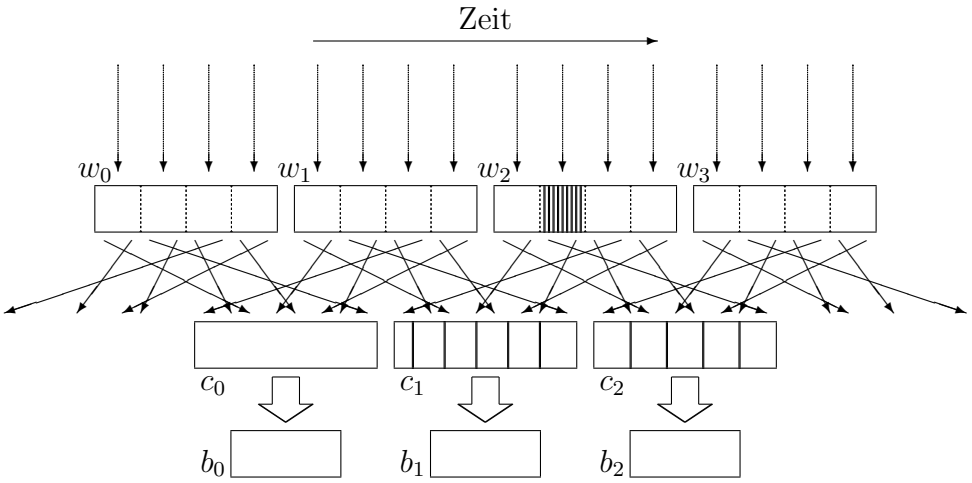

Abbildung 3.2: Bits aus w_2 werden auf c_1 und c_2 verteilt

Ausgegangen wird von einem kontinuierlichen Übertragungsstrom, in dem nun sukzessive quasi iterativ die einzelnen Empfangswörter deinterleaved und dann decodiert werden sollen. (Für den Beginn und das Ende einer Sendeperiode gelten entsprechende Modifikationen.)

Auf Grundlage einer vollständigen Schätzung soll eine verbesserte Decodierung von c_1 zu b_1 durchgeführt werden, nachdem c_0 schon zu b_0 decodiert wurde. Die Bits von c_1 werden übertragen in den Worten w_1 und w_2. Abbildung 3.2 deutet an, dass jedes Wort in 4 Bursts (auf 4 verschiedenen Kanälen) übertragen wird. Insgesamt sind also die acht Bursts, die an den Wörtern w_1 und w_2 beteiligt sind, zu schätzen. Die Bits von w_1 verteilen sich auf c_0 und c_1. Das Wort c_0 wurde in der Vergangenheit schon decodiert, so dass hier eine Recodierung

möglich ist. Die Bits von w_2 verteilen sich hingegen auf c_1 und c_2.Um beispielsweise μ_{Burst} und σ_{Burst} des markierten Bursts auf vollem Stichprobenumfang zu schätzen, braucht man recodierte Bits für die entsprechenden Positionen in c_1 und c_2. Man benötigt also neben einer ersten (Standard-) Decodierung von c_1 auch eine erste (Standard-) Decodierung von c_2. Letztere bezieht auch Bits des Wortes w_3 ein, womit ein Nachteil dieses Verfahrens offenbar wird: Die verbesserte Decodierung zu b_1 kann erst beginnen, wenn w_3 übermittelt wurde.

Das Verfahren lässt sich iterieren: Unter Hinzunahme des nächsten Codewortes lassen sich die Bursts von w_3 schätzen, was zu einer verbesserten Decodierung von c_2 zu b_2 führt. Eine Recodierung liefert dann zuverlässigere Burst-Schätzungen für w_2. Diese resultieren in einer noch besseren Decodierung von c_1 zu b_1.

Der Nachteil dieses Verfahrens ist - neben dem größeren Rechenaufwand wegen mehrfacher Re- und Decodierung - der systematische Zeitverzug: Wie erwähnt ist die verbesserte Decodierung von c_1 zu b_1 erst dann möglich, wenn w_3 übermittelt ist; bei weiteren Iterationen muss die Übermittlung von noch mehr Wörtern abgewartet werden.

In der Simulation bestätigen sich die theoretischen Überlegungen: Eine erste Re- und Decodierung wie beschrieben bringt schon Decodiergewinne von ca. 0,6 bis 0,9 dB. Bei einer zweiten Iteration steigen die Gewinne gegenüber dem Standard-Verfahren auf 0,8 bis 1,0dB und durch weitere Iterationen bis auf 0,9 bis 1,1dB. (Für genauere Simulationsergebnisse s. Abschnitt 4.1.)

3.2 Vereinfachte Kanalschätzung

Bei dem in Abschnitt 3.1 vorgestellten Verfahren wirken sich der hohe Rechenaufwand und der systematische Zeitverzug negativ aus. Das in diesem Abschnitt beschriebene Verfahren zeichnet sich dadurch aus, dass es mit ungefähr dem gleichen Rechenaufwand wie das Standard-Verfahren und ohne Zeitverzug auskommt. Während das Interleaving beim obigen Verfahren eher störte und zu dem erwähnten nachteiligen Zeitverzug führte, wird hier das Interleaving in besonderer Weise positiv ausgenutzt. Das Verfahren ist in Abbildung 3.3 illustriert.

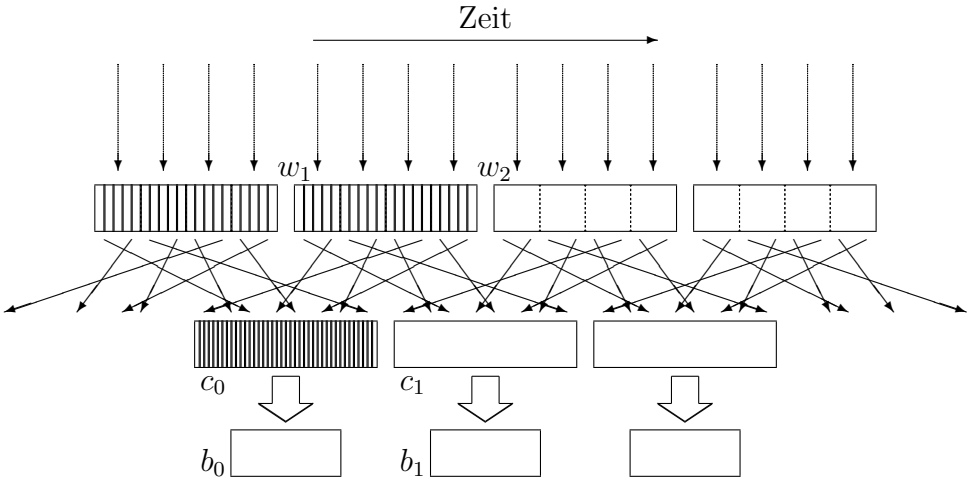

Abbildung 3.3: Schätzung der Bursts aus w_1 durch Recodierung von b_0

Auch hier wird wieder von einem kontinuierlichen Übertragungsstrom aussgegangen, bei dem dann sukzessive die Codewörter decodiert werden:

In der Vergangenheit wurde schon c_0 zu b_0 decodiert. Eine entsprechende Recodierung führt zu Schätzungen der Codebits in c_0 und damit zu Schätzungen der Kanalqualitätswerte bei den entsprechenden Bits. Auf diese Weise erhält man zu ca. der Hälfte der Bits des Wortes w_1 bzw. der vier zugehörigen Bursts Schätzungen. Mittelwerte und Varianzen dieser vier Bursts können nun allein auf Basis dieser abgeschätzten Werte berechnet werden; dies entspricht in etwa einer Schätzung mit halbem Stichprobenumfang im Vergleich zur vollständigen Schätzung beim in Abschnitt 3.1 beschriebenen Verfahren.

Die Mittelwerte zu den Bursts von w_2 können (in Ermangelung weiterer Infor-

mationen) wie in der Einleitung dieses Kapitels beschrieben mit Hilfe der Vertrauenswerte a_l an Stelle der q_l geschätzt werden. Man könnte auch versuchen, auf diese Weise die Varianzen zu schätzen; allerdings wurden dazu noch keine lohnenswerten Verfahren gefunden. Die besten Simulationsergebnisse ergeben sich, wenn als Varianz ein fester Wert genommen wurde (ca. das 1,3-fache der Durchschnitte der Varianzen, die sich bei den Burst-Abschätzungen von w_1 ergeben. Der höhere Varianzwert ist plausibel, denn in den Gewichtungsfaktoren $\frac{\mu_{\text{Burst}}}{\sigma_{\text{Burst}}{}^2}$ resultiert er in einer insgesamt schwächeren Gewichtung der entsprechenden Bursts aus w_2. Dies spiegelt den unsichereren Informationsstand bei der Schätzung der Kanalwerte von w_2 im Vergleich zu der auf recodierten Bits beruhenden Schätzungen bei w_1 wider).

Nachdem so Mittelwerte und Varianzen der beteiligten Bursts geschätzt wurden, kann man nun direkt mit dem modifizierten Verfahren c_1 zu b_1 decodieren. Man benötigt also keine zweite Decodierung sondern kann direkt unter Einbeziehung der wie oben beschrieben geschätzen Werte μ_{Burst} und σ_{Burst} decodieren.

Der Aufwand für eine modifizierte Decodierung (d.h. mit Einbeziehung der Faktoren $\frac{\mu_{\text{Burst}}}{\sigma_{\text{Burst}}{}^2}$ in die Metrik des Viterbi-Algorithmus) ist im Vergleich zur Standard-Decodierung von gleicher Größenordnung. Bei dem hier vorgestellten Verfahren kommt ferner aufwandsmäßig nur eine Recodierung sowie die Abschätzungen der Mittelwerte und Varianzen hinzu. Der diesbezügliche Rechenaufwand ist aber im Vergleich zu einer Decodierung vernachlässigbar gering, so dass man sagen kann, dass das Verfahren mit ungefähr dem gleichen Rechenaufwand wie das Standard-Verfahren auskommt.

Die bei der Simulation erzielten Decodiergewinne im Vergleich zum Standard-Verfahren liegen bei ca. 0,5dB, zu Details s. Abschnitt 4.2.

3.3 Iterierte vereinfachte Schätzung

Der Nachteil des Zeitverzuges beim in Abschnitt 3.1 vorgestellten Verfahren ist ein systematischer. Ein größerer Rechenaufwand kann hingegen mit mehr Rechenleistung ausgeglichen werden. (Schon bei den bisher üblichen Implementierungen ist die Decodierung schneller als die Übertragungsgeschwindigkeit.) Daher wird in diesem Abschnitt ein Verfahren vorgestellt, das (mittels Iteration und damit einem erhöhten Rechenaufwand) bessere Ergebnisse liefert als das in Abschnitt 3.2 beschriebene Verfahren, das aber im Gegensatz zum ersten Verfahren weiterhin ohne Zeitverzug (bei Vernachlässigung der Zeitverzögerung durch Re- und erneute Decodierung) auskommt:

Zur Erläuterung dient Abbildung 3.4: Eine erste Decodierung von c_1 zu b_1 erfolgt genau wie in Abschnitt 3.2 beschrieben. Das Decodierergebnis wird dann wieder recodiert. Auf diese Weise erhält man im Wort w_1 Schätzungen ${c_l}^{(1)}$ zu den bisher noch nicht geschätzten Bits, so dass man die Mittelwerte und Varianzen der entsprechenden Bursts nun mit vollem Stichprobenumfang schätzen kann. Ferner erhält man zu ca. der Hälfte der Bits aus w_2 Schätzungen ${c_l}^{(1)}$. Damit ist eine Schätzung von Mittelwert und Varianz dieser Bursts auf Basis des halben Stichprobenumfangs möglich.

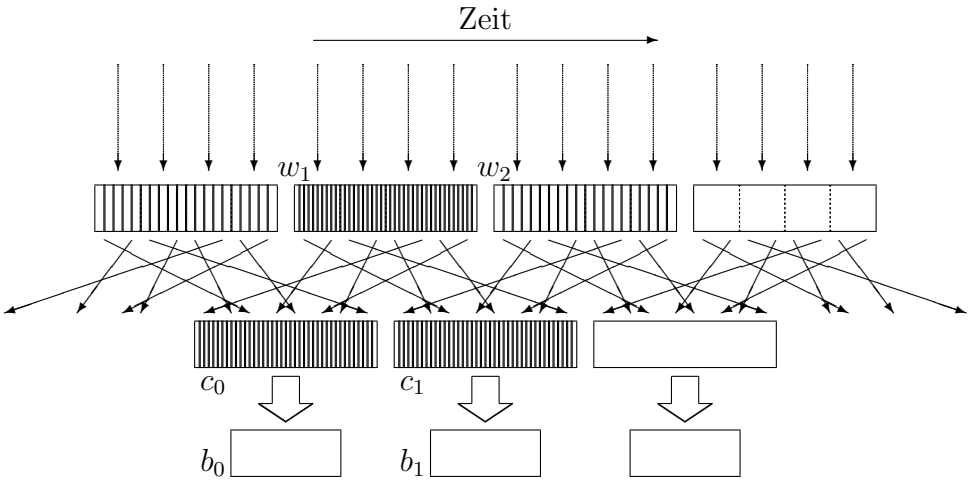

Abbildung 3.4: Recodierung von b_1 ergibt eine verbesserte Schätzgrundlage

Auf diese Weise hat man verbesserte Kanalschätzungen, die bei einer erneuten Decodierung von c_1 zu b_1 ein verbessertes Decodierergebnis zur Folge haben.

Dieses Vorgehen lässt sich iterieren: Eine erneute Recodierung führt zu verbesserten Schätzungen ${c_l}^{(2)}$, diese wieder zu besseren Schätzwerten für die Mittelwerte und Varianzen und so fort.

Die Simulation zeigt eine Vergrößerung des Decodiergewinns von 0,5dB beim Verfahren nach Abschnitt 3.2 auf ca. 0,7 bis 0,8dB mit einer zusätzlichen Re- und Decodierung. Weitere Iterationen bringen nur noch marginale Verbesserungseffekte (s. Abschnitt 4.3).

3.4 Gegenüberstellung der Verfahren

Im Folgenden werden die drei Verfahren nochmals bzgl. Aufwand, Zeit und Decodiergewinn zusammengefasst:

- Verfahren 1: Iterierte vollständige Kanalschätzung
  - − mehrfache Decodierung
  - − systematischer Zeitverzug
  - \+ Decodiergewinne bis zu 1,1 dB
- Verfahren 2: Vereinfachte Kanalschätzung
  - \+ einfache Decodierung
  - \+ kein Zeitverzug
  - − Decodiergewinne um 0,5 dB
- Verfahren 3: Iterierte vereinfachte Kanalschätzung
  - − mehrfache Decodierung
  - \+ kaum Zeitverzug
  - \+ Decodiergewinne von ca. 0,8 dB

Kapitel 4

Simulationsergebnisse

In diesem Kapitel werden die Ergebnisse vorgestellt, die bei einer Simulation der im vorigen Abschnitt vorgestellten Verfahren gewonnen wurden. Simuliert wurde eine Codierung und Decodierung nach GSM-Standard. Die Codierung erfolgte analog TCH/FS mit „Parity and Tailing“, „Convolutional Encoder“, „Interleaving“ und „Mapping on a Burst“ (s. [1], Abschnitte 3.1.2 bis 3.1.4): Jeweils 260 Infobits werden zu 456 Codebits codiert, genauer: Zu den ersten 50 Bits („class 1a“-Bits) werden 3 Parity-Bits erzeugt. Diese werden zusammen mit den ersten 182 Infobits codiert mit einem Faltungscode der Rate 1/2 (s. Abbildung 5 der Einleitung). Die letzten 78 Infobits werden ohne Redundanz in die Codefolge übernommen. Jeweils benachbarte Codewörter werden durch das Interleaving verschränkt. Diese Bitfolgen werden in Bursts der Länge 114 (zuzüglich zweier Kanal-Kontrollbits sowie von 26 Trainingsbits) gesendet.

Die Störungen bei der Übertragung wurden mit Hilfe von Stördaten modelliert, die freundlicherweise von Siemens ICM zur Verfügung gestellt wurden und auch dort zur Kanalsimulation eingesetzt werden. Auszüge aus den Daten sind in Anhang A abgedruckt. Es gab fünf verschiedene Datensätze, die jeweils unterschiedliche Sendeleistungen modellieren, genauer: die sich um jeweils 2dB bei der Sendeleistung unterschieden. Da die absolute Sendeleistung für die Ergebnisse nicht relevant ist, werden die Ergebnisse zu den unterschiedlichen Datensätzen in den folgenden Abschnitten mit x, $x + 2$, $x + 4$ bezeichnet. Sie beziehen sich nur auf die ersten drei Datensätze, da sich bei der Simulation mit den beiden letzten Datensätzen so wenige Fehler ereigneten, dass sich keine

statistisch abgesicherten quantitativen Aussagen bzgl. der Decodiergewinne ableiten ließen.

Die Decodierung wurde mit einem Viterbi-Algorithmus durchgeführt. Eine Beschreibung des Viterbi-Algorithmus findet sich in Anhang B. Ein Auszug des zur Simulation verwendeten C++-Programmcodes ist in Anhang C abgedruckt.

Das Standard-Verfahren bezeichnet in den folgenden Abschnitten jeweils die Decodierung mit der üblichen Metrik (die Soft-Werte respektierend, s. (1.2)), die modifizierten Verfahren eine Decodierung mit einer Metrik, wie sie in Kapitel 2 erläutert wurde (zusätzliche Einbeziehung der Kanal-Schätzwerte, s. (2.1)). Die Parity-Bits zu den „class 1a“-Bits wurden in allen Fällen bei der Decodierung nicht als zusätzliche Information berücksichtigt. Die modifizierten Versionen benutzten jeweils Kanalabschätzungen für die einzelnen Bursts, wobei gemäß dem Ziel, allein aus den Datenbits die Kanalschätzungen durchzuführen, die eingefügten Kanal-Kontrollbits nicht berücksichtigt wurden.

Die Bitfehlerrate (BER=Bit error rate) bei den folgenden Ergebnissen bezieht sich allein auf die durch die Faltung codierten Bits, d.h. auf die ersten 182 Bits eines Infoworts („class 1“-Bits).

Die im folgenden aufgeführten numerischen Resultate stützen sich auf eine Simulation von jeweils 20000 übertragenen Infoworten, also ca. 10 Millionen übertragener Bits.

4.1 Iterierte vollständige Kanalschätzung

Abbildung 4.1 zeigt die Bitfehlerraten zu den verschiedenen Sendeleistungen für das Standardverfahren (gepunktet) sowie für das in Abschnitt 3.1 beschriebene modifizierte Verfahren mit 1 bis 5 Iterationen (wobei die Kurven für 3 bis 5 Iterationen fast zusammenfallen). Abbildung 4.2 zeigt den Decodiergewinn bzgl. des Standardverfahrens bei verschiedenen Iterationszahlen und bei den unterschiedlichen Sendeleistungen. Abbildung 4.3 listet die genauen Werte auf, auf denen die Diagramme beruhen.

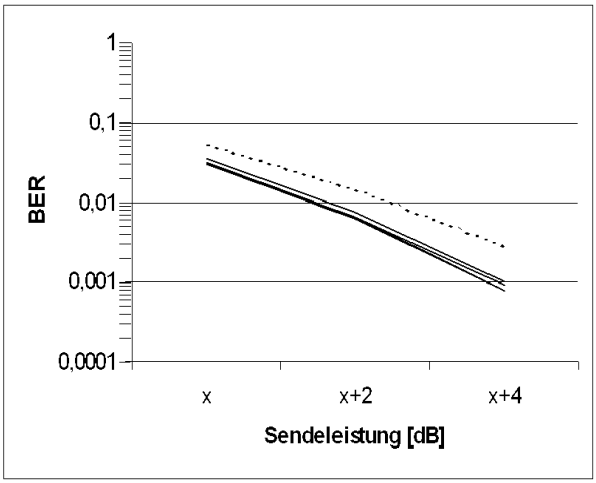

Abbildung 4.1: Bitfehlerraten beim ersten Verfahren

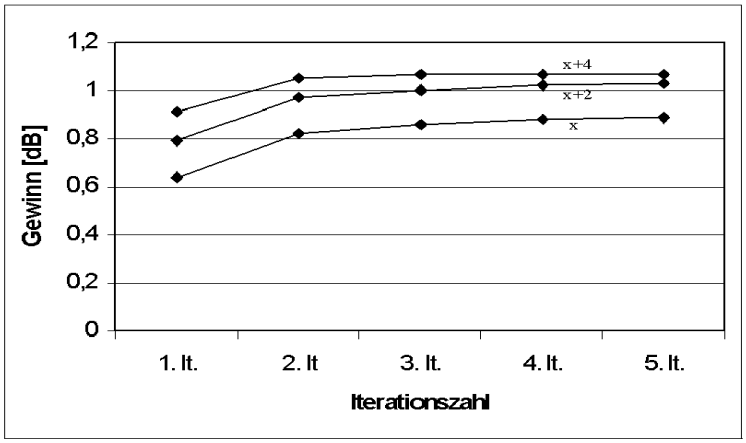

Abbildung 4.2: Gewinn [dB] bei verschiedenen Iterationen des ersten Verfahrens

		Sendeleistung		
		x	$x+2$	$x+4$
BER[%]	Standard	5,39	1,46	0,28
	1. Iteration	3,55	0,76	0,10
	2. Iteration	3,16	0,65	0,09
	3. Iteration	3,07	0,64	0,08
	4. Iteration	3,03	0,63	0,08
	5. Iteration	3,01	0,62	0,08
Gewinn[dB]	1. Iteration	0,64	0,79	0,91
	2. Iteration	0,82	0,97	1,05
	3. Iteration	0,86	1,00	1,07
	4. Iteration	0,88	1,02	1,07
	5. Iteration	0,89	1,03	1,07

Abbildung 4.3: Numerische Ergebnisse beim ersten Verfahren

Man erkennt einen deutlichen Decodiergewinn schon nach einer Iteration, der mit der zweiten Iteration nochmals signifikant gesteigert wird. Weitere Iterationen bringen nur noch kleinere Verbesserungen.

4.2 Vereinfachte Kanalschätzung

Abbildung 4.4 zeigt grafisch und Abbildung 4.5 numerisch die Ergebnisse für die in Abschnitt 3.2 beschriebene vereinfachte Kanalschätzung bei den unterschiedlichen Sendeleistungen. Bei der Bitfehlerrate sind in Abbildung 4.4 zum Vergleich die Ergebnisse des Standardverfahrens aufgeführt (gestrichelt). Die letzte Zeile in Abbildung 4.5 gibt den Decodiergewinn des modifizierten Verfahrens gegenüber dem Standardverfahren an.

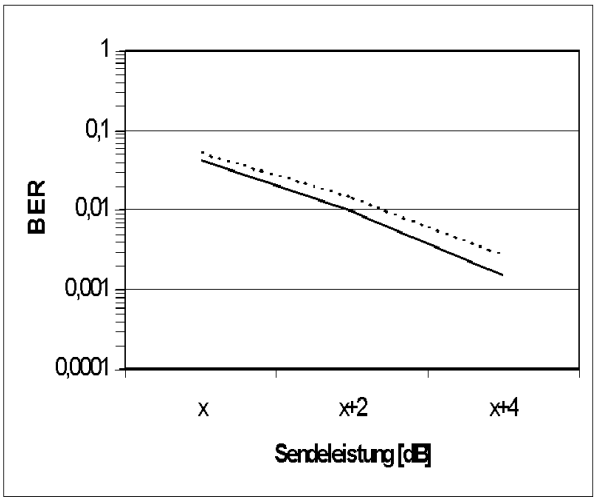

Abbildung 4.4: Bitfehlerraten beim zweiten Verfahren

		Sendeleistung		
		x	$x+2$	$x+4$
BER[%]	Standard	5,39	1,46	0,28
	verbessert	4,06	0,95	0,15
Gewinn[dB]		0,44	0,51	0,55

Abbildung 4.5: Numerische Ergebnisse beim zweiten Verfahren

Man erhält bei den zu Grunde gelegten Daten also auch bei dem einfachen Verfahren (ohne Zeitverzögerung und ohne signifikatnen Mehraufwand) eine Verbesserung von ca. 0,5 dB.

4.3 Iterierte vereinfachte Schätzung

Abbildung 4.6 zeigt die Bitfehlerraten zu den verschiedenen Sendeleistungen für das Standardverfahren (gepunktet), für das modifizierte Verfahren mit vereinfachter Kanalschätzung wie in Abschnitt 3.2 beschrieben (gestrichelt) sowie für die in Abschnitt 3.3 beschriebene iterierte Schätzung mit 1 bis 4 Iterationen, wobei diese fast zusammenfallen. Abbildung 4.7 zeigt den Decodiergewinn bzgl. des Standardverfahrens bei verschiedenen Modifikationen und bei den unterschiedlichen Sendeleistungen. Abbildung 4.8 listet die genauen Werte auf, auf denen die Diagramme beruhen.

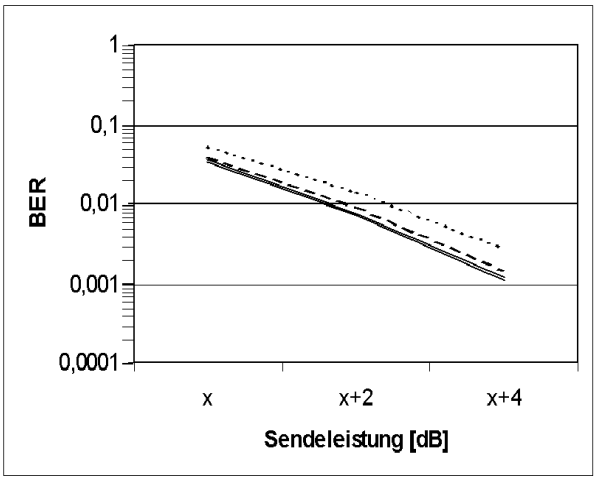

Abbildung 4.6: Bitfehlerraten beim dritten Verfahren

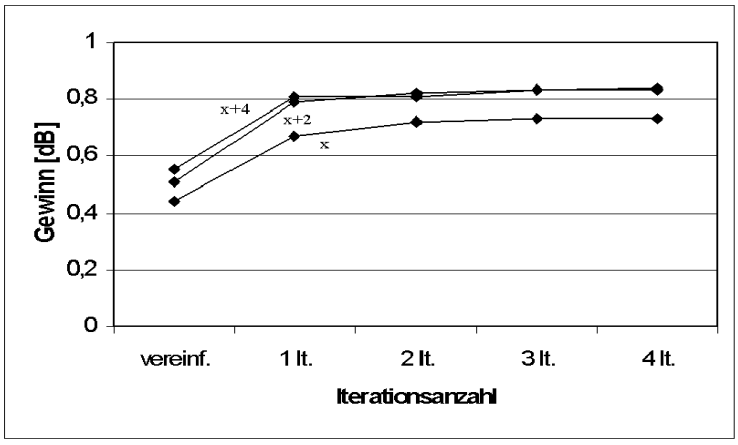

Abbildung 4.7: Gewinn [dB] bei verschiedenen Iterationen des dritten Verfahrens

		Sendeleistung		
		x	$x+2$	$x+4$
BER[%]	Standard	5,39	1,46	0,28
	vereinfacht	4,06	0,95	0,15
	mit 1 It.	3,48	0,76	0,12
	mit 2 It.	3,38	0,74	0,11
	mit 3 It.	3,34	0,73	0,11
	mit 4 It.	3,34	0,73	0,11
Gewinn[dB]	vereinfacht	0,44	0,51	0,55
	mit 1 It.	0,67	0,79	0,81
	mit 2 It.	0,72	0,82	0,83
	mit 3 It.	0,73	0,83	0,83
	mit 4 It.	0,73	0,84	0,83

Abbildung 4.8: Numerische Ergebnisse beim dritten Verfahren

Man erkennt bei einer zusätzlichen Iteration einen deutlich größeren Decodiergewinn als beim vereinfachten Verfahren gemäß Abschnitt 3.2. Weitere Iterationen bringen nur noch verhältnismäßig kleine Gewinnsteigerungen.

4.4 Zusammenfassung und Einordnung der Simulationsergebnisse

In diesem Abschnitt werden die durch die drei Verfahren erreichten Bitfehlerraten bzw. Decodiergewinne zusammengefasst und verglichen. Abbildung 9 listet die entsprechenden Daten für eine Standard-Decodierung sowie für die beschriebenen Verfahren auf. Es wurden jeweils gerundete Werte benutzt für die Ergebnisse von 2 bis 3 Iterationen bei Verfahren 1 und 1 bis 2 Iterationen bei Verfahren 2.

		Sendeleistung		
		x	$x+2$	$x+4$
Standard	BER[%]	5,4	1,5	0,3
Verfahren 1	BER[%]	3,1	0,6	0,08
	Gewinn[dB]	0,9	1,0	1,1
Verfahren 2	BER[%]	4,1	1,0	0,15
	Gewinn[dB]	0,44	0,51	0,55
Verfahren 3	BER[%]	3,4	0,74	0,11
	Gewinn[dB]	0,7	0,8	0,8

Abbildung 4.9: Zusammenfassung der Ergebnisse

Zur Einordnung der Ergebnisse kann man die Bitfehlerraten vergleichen mit denen, die man bei einer Decodierung mittels (2.1) erhält, wenn man die tatsächlichen Werte μ_{Burst} und σ_{Burst} für die Bursts einsetzt, z.B., indem man sie bei der Simulation als Mittelwerte bzw. Varianzen der eingespeisten Kanalqualitätswerte berechnet. Man erhält dann die folgenden Werte:

	Sendeleistung		
	x	$x+2$	$x+4$
BER[%]	2,11	0,42	0,06

Abbildung 4.10: Ergebnisse bei vollständiger Kanalkenntnis

Die Verfahren realisieren also ca. die Hälfte des Gewinns, der mit dem in Kapitel 2 erläuterten Ansatz bei vollständiger Kanalkenntnis möglich wäre.

Aus den Kanalqualitätswerten kann man eine Abschätzung der Kanalkapazität herleiten:

Die Angaben der Kanalqualitätswerte sind ganze Zahlen zwischen -127 und 127 (s. Anhang A). Modelliert man den Kanal durch die Eingangssymbole $a \in \{-1, +1\}$ und die Ausgangssymbole $b \in \{-127, -126, \ldots, 126, 127\}$, so kann man durch eine Statistik über die Folge der Kanalqualitätswerte die bedingten Wahrscheinlichkeiten $P(b|a)$ abschätzen. Für $a = +1$ ergibt sich $P(b|1)$ direkt als Anteil von b an den Qualitätswerten. Beispielsweise ist so $P(127|1) = 0.29$, $P(126|1) = 0.00243$, ... bei der kleinsten Sendeleistung x und z.B. $P(127|1) = 0.40$ bei $x + 2$. Aus Symmetriegründen ist $P(b| - 1) = P(-b|1)$.

Falls man die Gedächtniseffekte des Kanals unberücksichtigt lässt, also zu einem eigentlich schlechteren Kanal übergeht, kann man mit diesen Werten nun die (bedingten) Entropien und die Transinformation $I(A, B)$ berechnen, in Abhängigkeiten von den Eingangswahrscheinlichkeiten $p(a = 1)$ und $p(a = -1)$ (s. [7]). Die Kanalkapazität erhält man als maximale (bzgl. der Eingangswahrscheinlichkeiten) Transinformation. (Man kann nachrechnen, dass wegen der Symmetrie dieses Maximum bei $p(a = 1) = p(a = -1) = \frac{1}{2}$ angenommen wird.)

Man erhält so die folgenden Kanalkapazitäten, die wegen des unberücksichtigten Gedächtnisses eine untere Schranke darstellen:

	Sendeleistung		
	x	$x + 2$	$x + 4$
Kanalkapazität	0,61	0,72	0,81

Abbildung 4.11: Berechnete Kanalkapazitäten

Man sieht, dass auch bei dem Datensatz mit der niedrigsten Sendeleistung die Kanalkapazität oberhalb von $\frac{1}{2}$ liegt. Nach dem Shannonschen Kanalkapazitätssatz ist damit theoretisch eine beliebig fehlerarme Codierung mit einer Rate 1/2, wie sie den Simulationen zugrunde lag, möglich (s. [7]).

Anhang A

Beispieldaten

Dieser Anhang enthält einen Auszug aus den Daten, die bei der Simulation als Kanalqualitätswerte zu Grunde gelegt wurden. Die Daten wurden freundlicherweise von Siemens ICM zur Verfügung gestellt. Da die Ausgaben des Demodulators gewöhnlich nur 4- oder 8-Bit Zahlen sind, sind auch diese Simulationswerte auf ganze Zahlen zwischen -127 und +127 diskretisiert. Durch eine Division mit 127 erhält man Werte im Intervall $[-1, 1]$, die man dann als Kanalqualitätswerte wie in Kapitel 1 beschrieben benutzen kann. Eine Skalierung ändert allerdings nichts an der Lösung der betrachteten Minimierungsprobleme, wie man leicht durch ähnliche Überlegungen sieht, wie sie in den Abschnitten 1.2 und 2.2 bei den Umformulierungen der Zielfunktionen durchgeführt wurden, so dass man diese Kanalqualitätswerte direkt zur Berechnung der entsprechenden Ausdrücke (1.2) und (2.1) benutzen kann.

Die Daten simulieren eine Übertragung, bei der jeweils 114 Bits in einem Burst enthalten sind. Man sieht deutlich die unterschiedliche Qualität aufeinanderfolgender Bursts (durch Linien voneinander getrennt): Der erste Burst bietet eine sehr gute Übertragungsqualität, der zweite eine schlechtere, der dritte dann wieder eine etwas bessere. Es gibt sehr schlechte Bursts (z.B. der achte abgedruckte Burst, der einen Mittelwert von nur $\mu_{\text{Burst}} \approx 6,2$ aufweist), aber dann auch wieder sehr gute (der vorletzte abgedruckte).

Die Daten spiegeln also die in den Kapiteln 1 und 2 gemachten Annahmen wider, nämlich Burst-abhängige Empfangsschwankungen, die durch das Rayleigh-Fading verbunden mit Frequency-hopping erklärt werden knnen. Ferner kann

man nachrechnen, dass die Werte innerhalb eines Bursts, also bei vorgegebenem Mittelwert, nicht korreliert sind. Dies entspricht der AWGN-Annahme in den Kapiteln 1 und 2.

110	126	126	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127
127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127
127	127	127	127	127	127	127	127	127	127	99	127	127	127	127	127	127	127	127
127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	110
127	127	127	127	127	127	127	125	127	127	127	127	127	118	127	127	127	127	127
127	127	97	127	127	127	127	127	127	114	127	127	127	127	127	91	121	113	46
29	40	37	37	59	-50	48	-67	-38	87	-46	-83	-28	100	72	-55	-28	-33	-22
73	42	59	-75	78	-10	11	93	10	19	100	93	-35	76	-78	52	-78	78	71
-37	-83	-61	97	99	99	99	99	82	-71	35	3	93	-46	51	-54	70	-60	81
70	32	40	62	-20	-15	-30	9	82	63	-48	-49	45	-36	-14	-1	33	57	29
46	-28	-9	33	77	72	-33	-15	56	52	41	44	66	-39	-48	-66	50	-41	-38
-46	8	41	-18	82	-5	36	58	33	52	55	-17	81	81	70	-36	31	4	40
34	77	53	68	127	127	48	82	127	127	114	33	45	127	127	93	127	56	127
99	127	127	63	127	127	127	127	127	127	127	127	56	127	37	99	65	127	104
41	36	119	6	80	65	48	127	9	18	18	84	41	127	127	105	30	127	21
36	120	53	115	85	121	31	120	15	42	41	33	98	124	54	54	116	-9	52
104	127	124	103	127	84	109	127	43	127	127	127	127	106	127	31	104	18	18
126	77	68	69	90	88	4	4	86	90	127	30	127	55	51	97	26	23	23
20	10	8	6	-1	18	26	34	49	52	24	35	26	49	41	0	-1	20	55
70	29	10	10	25	39	5	9	32	20	14	35	32	45	19	8	-9	20	-6
9	-6	23	47	37	27	43	9	13	20	39	48	50	63	54	40	27	29	28
-10	-5	21	0	0	-7	24	13	40	0	-4	15	5	29	0	1	14	16	27
34	-1	10	15	3	33	25	53	45	25	46	35	44	32	43	15	14	3	24
6	4	34	12	30	48	16	-5	-13	11	22	18	12	4	36	39	10	10	24
-2	-4	-28	-30	14	-26	18	0	36	-10	6	-35	-3	-33	-37	14	8	54	-25
-26	44	0	29	-7	7	-8	16	56	3	-67	-27	-21	48	45	-16	8	27	75
-17	27	7	9	-57	31	1	52	-28	41	-18	-27	-11	-57	44	45	20	-59	-27
5	-25	-3	2	-3	-23	54	17	48	-15	8	-19	-21	38	-16	-29	22	32	-16
58	22	3	-30	-18	-5	11	-26	12	13	-9	-13	-21	4	49	-45	-3	44	59
60	59	-34	-57	11	-1	-24	10	57	56	31	-56	-6	29	-9	-4	45	12	-12

54	11	53	-17	17	53	24	21	32	26	34	30	-22	30	10	13	-14	35	32
24	-36	13	58	80	7	-10	-18	29	39	79	4	14	16	3	53	29	32	31
-33	64	21	18	38	-7	29	70	79	20	80	65	26	-15	-35	49	36	39	20
4	71	5	12	22	57	-24	28	41	-7	59	15	16	57	4	-14	-10	23	24
12	-24	66	30	33	-36	63	21	72	72	18	71	71	11	-3	9	7	-18	14
-21	72	29	9	27	-27	52	-5	54	69	56	71	4	12	24	-19	38	-9	35
61	103	91	87	50	12	72	127	127	69	127	113	127	127	64	116	99	23	36
89	108	127	111	31	47	127	127	127	88	11	51	17	53	48	85	33	45	96
27	24	107	30	127	85	12	33	50	69	75	87	114	111	108	32	72	87	127
65	97	53	120	87	125	36	111	50	90	58	127	50	127	127	70	17	32	25
107	117	58	63	58	83	127	49	127	99	0	90	88	92	114	59	127	77	127
106	92	63	45	48	31	8	99	43	86	33	41	15	49	22	99	50	17	39
4	-6	-5	29	-17	-19	-34	14	0	27	12	12	9	11	-16	-26	25	-13	-16
-23	6	-10	-2	-4	38	-4	-2	-14	13	17	4	14	5	28	-1	2	14	5
4	5	-5	-16	-1	16	-10	7	17	12	15	11	-6	-8	15	0	-6	4	21
32	22	-10	-8	-2	1	7	-11	-11	17	33	23	-17	-14	21	-17	6	27	14
19	0	27	17	14	-14	-11	12	28	-18	9	25	13	11	19	0	45	12	18
44	32	-20	21	25	-19	18	-7	30	0	7	5	16	28	21	-12	-8	21	6
68	39	-6	45	45	79	64	80	79	62	20	38	79	79	30	33	40	50	72
-9	17	25	70	69	64	79	27	80	21	79	15	21	20	51	10	80	79	10
80	15	79	31	80	79	74	46	80	60	79	7	4	80	60	42	75	21	65
81	76	4	46	5	60	81	72	82	33	81	82	81	68	82	5	51	81	73
82	33	81	82	33	58	81	81	49	35	81	76	6	34	36	36	70	46	82
73	81	77	82	69	45	53	60	8	82	65	64	43	49	59	9	56	29	15
26	-1	3	28	20	35	23	41	50	7	36	60	34	37	20	56	24	23	43
61	38	45	35	75	17	35	29	32	39	44	54	41	56	11	63	60	38	63
50	39	46	20	-10	51	51	25	70	50	15	43	-1	57	-6	32	-6	33	20
27	19	39	20	4	20	5	25	26	45	29	35	29	31	39	24	5	43	29
40	22	34	20	21	43	-1	57	14	32	37	37	24	4	-11	13	26	33	4
10	-4	5	11	12	10	7	35	34	58	37	38	23	41	10	30	-2	20	-1
7	27	41	18	62	82	16	42	-8	-4	-49	25	0	-2	42	22	-65	-65	25
78	-17	44	-8	-48	30	-51	-63	-31	50	30	31	38	-21	-37	-46	-18	-81	8
0	37	43	32	54	51	25	-9	62	-51	-81	-38	5	25	82	-21	58	80	70
-34	-27	7	8	-20	30	-9	10	-5	5	61	16	-21	14	-11	34	23	-13	-43
-51	-2	23	-63	21	-62	40	-15	-60	62	57	25	-25	12	-40	-59	15	5	5
12	29	-60	46	-42	24	-19	-2	-31	19	29	-2	27	14	9	39	-16	-3	1

-19	32	32	76	18	12	36	88	44	127	127	67	72	83	32	80	5	30	18
35	6	55	69	9	7	65	77	-5	49	91	53	52	49	60	127	127	127	49
117	59	28	27	32	37	52	98	39	77	41	38	40	24	29	38	51	119	127
74	127	89	-13	13	47	122	127	91	90	56	10	120	127	113	55	7	3	95
88	87	56	37	84	84	127	127	78	-2	42	13	54	54	33	33	18	6	8
84	109	69	72	127	120	66	56	95	52	14	1	15	37	39	31	27	52	55
127	106	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127
127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127
127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127	127
127	127	127	98	62	127	119	127	127	127	80	127	127	114	127	121	127	127	124
127	115	127	127	127	99	127	127	127	127	127	127	63	106	127	127	127	87	127
81	127	127	127	127	87	80	127	127	127	105	127	127	127	127	104	125	51	111
17	15	5	-48	-6	30	3	-7	7	12	6	-11	-11	30	15	-29	-24	15	8
26	-21	-4	24	34	-20	-28	11	-23	-18	-23	-3	40	-22	-30	-1	4	6	-35
12	-31	-35	17	4	9	13	-3	-6	28	-4	27	37	12	9	10	-18	-16	30
21	-48	-54	-66	60	38	0	-19	-46	-43	41	-46	-24	49	-17	-19	49	9	24
-17	-64	61	-66	-43	34	-15	-56	46	-52	4	-13	27	17	-57	27	8	-14	-17
12	55	-27	-54	-33	40	16	41	19	-57	-73	-80	-86	70	60	10	-29	-8	-1

Anhang B

Der Viterbi-Algorithmus

Dieser Anhang enthält eine kurze Beschreibung des Viterbi-Algorithmus, da dieser Algorithmus wesentlich ist zur effizienten Lösung der Maximierungsaufgaben (1.2) und (2.1). Eine Beschreibung des Viterbi-Algorithmus findet man beispielsweise auch in [3, 14]. Die Implementierung, die zur Simulation benutzt wurde, ist in Anhang C abgedruckt.

Der Viterbi Algorithmus dient zur Decodierung einer durch eine Faltungscodierung codierten Bitfolge. Er wird hier zunächst an Hand des schon in der Einleitung eingeführten Faltungscodes erläutert, der durch das Schieberegister in Abbildung B.1 festgelegt ist.

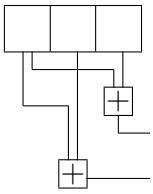

Abbildung B.1: Das zur Codierung benutzte Schieberegister

Die Funktionsweise des Algorithmus macht man sich am besten am Bild des sogenannten Trellis klar (s. Abbildung B.2).

Zu Beginn ist das Schieberegister mit Nullen initialisiert. Das erste Infobit wird nun von links hineingeschoben. Dabei entstehen zwei Output-Bits: Beim Einschieben einer Null (also 000 im Schieberegister) ist der Output 00, beim

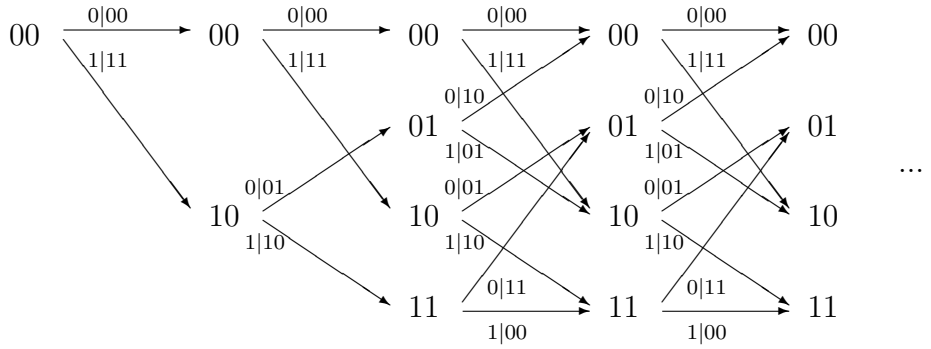

Abbildung B.2: Trellis

Einschieben einer Eins (100 im Schieberegister) ist er 11. Nachdem man die Ausgabe abgelesen hat, ist das rechte Bit im Schieberegister nicht mehr wichtig. Für die Zukunft relevant sind nur noch die linken beiden Bits. Dort steht nun 00 oder 10. Der bis hierhin beschriebene Zusammenhang ist in Abbildung B.2 links dargestellt: Ausgehend von 00 (nur die beiden linken Bits des initialisierten Schieberegisters) wird eine 0 oder 1 eingeschoben (jeweils die linke Zahl an den von 00 ausgehenden Pfeilen), womit eine Ausgabe 00 oder 11 erzeugt wird (das rechte Zahlenpaar an den Pfeilen). Man gelangt so zu den Zuständen 00 bzw. 10. Von dort aus geht es entsprechend weiter: Von 00 gelangt man durch Einschieben von 0 bzw. 1 in das Schieberegister (also eine Ausgabe 00 bzw. 11) zu 00 bzw. 10, und von 10 gelangt man durch Einschieben von 0 bzw. 1 (also eine Ausgabe 01 bzw. 10) zu 01 bzw. 11. Nun sind alle möglichen Registerzustände erreicht, und der entsprechende Übergang beginnt sich zu wiederholen.

Der Viterbi-Algorithmus nutzt nun die folgende Beobachtung aus: Sowohl durch eine Infofolge 100 (das rechte Bit als erstes eingeschoben) als auch durch 101 gelangt man zum in Abbildung B.3 markierten Zustand 10. Die entsprechenden Codefolgen bis hierhin sind (von links nach rechts gelesen) 00-00-11 bzw. 11-01-01. Wie die Codefolge sich weiter entwickelt, hängt nun nicht mehr von der Vorgeschichte sondern nur noch von dem erreichten Zustand ab.

Sucht man nun zu einer empfangenen Folge $y_1 y_2 \dots y_K$ das Infowort b, so dass für die entsprechende Codefolge $c(b) = c_1 c_2 \dots c_K$ die Summe

$$\sum_{k=1}^{K} f(c_k, y_k) \tag{B.1}$$

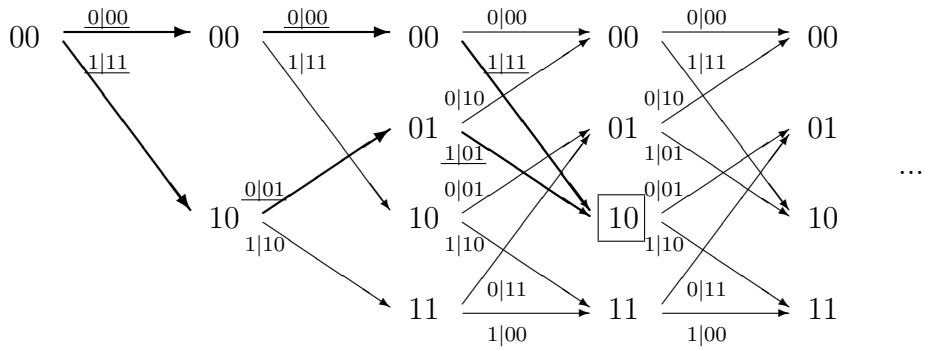

Abbildung B.3: Zwei Pfade zum gleichen Zustand

minimal wird (z.B. $f(c_k, y_k) = (c_k - y_k)^2$, was zur Maximierungsaufgabe (1.2) äquivalent ist, s. Ende Abschnitt 1.2), so braucht man im markierten Zustand für die Zukunft nur die eine der beiden möglichen Vorgeschichten (100 oder 101 bzw. als Codefolge $c_1c_2 \ldots c_6 = 000011$ oder $c_1c_2 \ldots c_6 = 110101$) berücksichtigen, die den kleinsten Wert

$$\sum_{k=1}^{6} f(c_k, y_k)$$

besitzt.

Entsprechend braucht man bei der Decodierung einer Empfangsfolge in jedem Zustand des Trellis nur eine Vorgeschichte zu behalten.

Bei dem simulierten GSM-Standard werden bei der Decodierung den Infobits noch Nullen angehängt, so dass man definitiv in den Nullzustand beim Trellis ausläuft; zum Ende sind dann nicht mehr alle Zustände möglich, s. Abbildung B.4.

Durch das Auswählen des besten Pfades erhält man damit automatisch das beste Infowort b. Ohne Auslaufen in den Nullzustand muss man die Werte an den Zuständen im letzten Schritt vergleichen und nimmt dann das Infowort, das zum bestwertigen Zustand gehört.

Es ist offensichtlich, dass man statt (B.1) zu minimieren, bei Wahl des jeweils größten Wertes eine entsprechende Maximierungsaufgabe lösen kann. Bei anderen Codierungen (z.B. längeren Schieberegistern) verändert bzw. erweitert sich das Trellis entsprechend. Die Funktionsweise des Algorithmus überträgt sich aber ohne Schwierigkeiten.

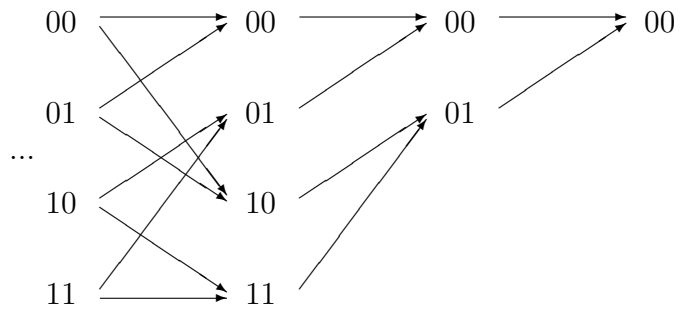

Abbildung B.4: In den Nullzustand auslaufendes Trellis

Wie man sieht, ist der Algorithmus unabhängig von der konkreten Form der Summanden $f(c_k, y_k)$. Wie bereits in Abschnitt 1.2 erwähnt, erhält man mit

$$f(c_k, y_k) = \begin{cases} 1 & \text{, falls } y_k \text{ und } c_k \text{ gleiches Vorzeichen haben,} \\ 0 & \text{, falls } y_k \text{ und } c_k \text{ unterschiedliches Vorzeichen haben,} \end{cases}$$

den Hard-Decision-Viterbi-Decoder und mit

$$f(c_k, y_k) = c_k \cdot y_k$$

einen Algorithmus zum Lösen von (1.2) (Soft-Decision-Viterbi-Decoder).

Bei

$$f(c_k, y_k) = \frac{\mu_{\text{Burst}(k)}}{{\sigma_{\text{Burst}(k)}}^2} \cdot y_k \cdot c_k$$

erhält man einen Algorithmus zur Lösung von (2.1).

Anhang C

Quellcode

Dieser Anhang enthält den Quellcode des Viterbi-Algorithmus, der zur Simulation der in Kapitel 3 beschriebenen Verfahren benutzt wurde, und mit dem die in Kapitel 4 wiedergegebenen Ergebnisse erzielt wurden. Der Code wurde unter Microsoft Visual C++ entwickelt.

Die Eingabe besteht aus dem Codewort `codewort`, das die Soft-Werte der einzelnen Bits enthält, sowie zwei Vektoren `mean` und `sigma`, die jeweils die nach den einzelnen Verfahren berechneten Kanalqualitätswerte μ_{Burst} und σ_{Burst} enthalten (jeweils für den Burst, über den das entsprechende Bit übertragen wurde; ein entsprechendes Deinterleaving dieser Werte wurde vor dem Aufruf von `decoding` durchgeführt).

Die Vektoren `AusgabeBit1` und `AusgabeBit2` enthalten für jeden Zustand des Schieberegisters (als Binärzahl interpretiert) die beiden Ausgabebits der zu Grunde liegenden Faltungscodierung (als 0 oder 1). Diese beiden Vektoren werden zu Beginn der Simulation als globale Variablen angelegt.

Die Funktion `zurueckordnen` ordnet die Bits entsprechend des verwendeten Standards [1, TCH/FS] nach der Auflösung der Faltung zurück und rundet die ohne Redundanz übermittelten class2-Bits.

Der Type `VecInt` ist definiert als `vector<integer>`.

```
decoding(vector<double>& codewort,
         vector<double>& mean,
         vector<double>& sigma)
  // Modifizierte Viterbi-Dekodierung
  // Eingabe: 456er-codeworte, mean und sigma entsprechend der
  //          Schaetzungen an den entsprechenden Bitpositionen
  // Ausgabe: 260er-Infowort, wobei nur die Bits 0,..181
  //          des Infoworts durch Redundanz bestimmt sind.
{
   int k, posU, z, b, j, index, i;
   double soft1, soft2, mean1, mean2, sigma1, sigma2;
   double sum1, sum2, wert;
   int hart1, hart2, nextZ;
   m_anzG = 5; // Laenge des Schieberegisters
   m_anzZ = 16; // Anzahl der (zu speichernden) Zustaende

      // Array mit Fehlerwerten; Wert -99 bedeutet 'nicht besetzt'
   double* fehlerwert = new double[m_anzZ];
      // Array mit aktuellen Fehlerwerten
   double* fehlerwertNeu = new double[m_anzZ];
   for (k=0; k<m_anzZ; k++)
      fehlerwert[k] = -99.0;
   fehlerwert[0] = 0;    // fuer Nullzustand: 0
   vector<VecInt> vorgaenger(m_anzZ);
   for (k=0; k<m_anzZ; k++)
      vorgaenger[k].assign(185+m_anzG);
   vector<VecInt> bit(m_anzZ);
   for (k=0; k<m_anzZ; k++)
      bit[k].assign(185+m_anzG);

   // Schleife ueber die Code-Bits mit Auslaufen in den Null-Zustand
   for (posU = 0; posU<185+m_anzG-1; posU++)
   {
      for (k=0; k<m_anzZ; k++)
         fehlerwertNeu[k] = -99;
         // Welche Bits wurden empfangen?
      soft1 = codewort[2*posU];
      soft2 = codewort[2*posU+1];
      mean1 = mean[2*posU];
      mean2 = mean[2*posU+1];
      sigma1 = sigma[2*posU];
      sigma2 = sigma[2*posU+1];
         // Schleife ueber die Zustaende ...
      for (z=0; z<m_anzZ; z++)
            // ... die besetzt sind
         if (fehlerwert[z]>-1)
```

```
            // Schleife ueber naechstes Bit (=0 oder =1)
         for (b=0; b<=1; b++)
         {
               // Berechnung des neuen Fehlerwertes
            wert = fehlerwert[z];
            hart1 = AusgabeBit1[2*z+b]*2-1;   // als +/-1
            hart2 = AusgabeBit2[2*z+b]*2-1;   // als +/-1
            sum1 = mean1/(sigma1*sigma1)*hart1*soft1;
            sum2 = mean2/(sigma2*sigma2)*hart2*soft2;
            wert += (sum1 + sum2)/100;
            nextZ = (2*z+b)%m_anzZ;      // neue Zustandsnummer
            if (fehlerwertNeu[nextZ]<-1 ||
                  (fehlerwertNeu[nextZ]>-1 &&
                      wert>fehlerwertNeu[nextZ]))
            {
               fehlerwertNeu[nextZ] = wert;
               vorgaenger[nextZ][posU+1] = z;
               bit[nextZ][posU+1] = b;
            }
         }

      for (j=0; j<m_anzZ; j++)
         fehlerwert[j] = fehlerwertNeu[j];
   }

      // Ergebnis: Nullzustand (da Auslaufen)
   vector<int> u(185+m_anzG);
   index = 0;
   for (k=185+m_anzG-1; k>0; k--)
   {
      u[k-1] = bit[index][k];
      index = vorgaenger[index][k];
   }

      // Zurueckordnen zum Infowort und class2-Bits runden
   vector<int> infowort(260,0);
   infowort = zurueckordnen(u,codewort)

   delete fehlerwert;
   delete fehlerwertNeu;
   return infowort;
}
```

Zusammenfassung

Diese Arbeit beschreibt die Ideen zu einer verbesserten Decodierung bei GSM.

Ein gegenüber der Standard-Kanalmodellierung verfeinertes Modell wird betrachtet, das die unterschiedlichen Übertragungsqualitäten verschiedener Kanäle berücksichtigt und diese Qualitätswerte in die Decodierung einbezieht. Dies führt zu einem Minimierungsproblem, das ähnlich dem bei der Standard-Decodierung üblichen Minimierungsproblem ist, allerdings mit geänderter Metrik (abhängig von den Qualitätswerten des Kanals). Bei bekannten Qualitätswerten kann die Decodierung mit einem in der Metrik modifizierten Viterbi-Algorithmus durchgeführt werden.

Es werden drei Verfahren vorgestellt, wie man in der Praxis diese Qualitätswerte allein aus den übertragenen Datenbits durch „decision feedback“, also durch Recodierung von bereits decodierten Wörtern, ohne Einbeziehung von Trainingssequenzen schätzen und dann in eine verbesserte Decodierung einfließen lassen kann:

1) Ein iteriertes Verfahren, das die Qualitätswerte auf dem vollen Stichprobenumfang schätzt. Dieses Verfahren bringt die besten Ergebnisse, hat aber auf Grund des Interleavings einen systematischen Zeitverzug bei der Decodierung zur Folge.

2) Ein vereinfachtes Verfahren, das ohne mehrfache Decodierung und ohne Zeitverzug auskommt.

3) Ein iteriertes vereinfachtes Verfahren als Mischung von 1) und 2).

Die Verfahren wurden simulativ mit von Siemens ICM bereitgestellten Testdaten erprobt. Die entsprechenden Ergebnisse zeigen Decodiergewinne von ca. 0,5dB bis zu über 1dB bei den betrachteten Szenarien.

Literaturverzeichnis

[1] 3GPP TS 45.003 V4.4.0: *3er Generation Partnership Project; Technical Specification Group GERAN; Channel Coding (Release 4)*. Sophia Antipolis, Valbonne, 2001.

[2] S. Benedetto, E. Bigliere: *Principles of Digital Transmission*. Kluwer Academic/Plenum Publishers, New York, 1999.

[3] K. David, T. Benkner: *Digitale Mobilfunksysteme*. B.G. Teubner, Stuttgart, 1996.

[4] J.P. Fonseka, Y. Yin, I. Korn: *Sensitivity to Fading Estimates in Maximal Ratio Combining*. IEEE Proc.-Commun. **150** (4), 269-274, 2003.

[5] G.D. Forney: *The Viterbi Algorithm*. Proc. of the IEEE **61** (3), 268-278, 1973.

[6] J. Hagenauer: *Soft is Better Than Hard*. In: R.E. Blahut, D.J. Costello, U. Maurer, T. Mittelholzer (Hrsg.): Communications and Cryptography - Two Sides of One Tapestry, 155-171, 1994.

[7] R. W. Hamming: *Information und Codierung*. Wiley-VCH, 1987.

[8] N. Kong, L.B. Milstein: *Combined Average SNR of a Generalized Diversity Selection Combining Scheme*. In: Proc. IEEE Int. Conf. Commun., 1556-1560, Atlanta, 1998.

[9] L. Li and A.J. Goldsmith: *A Decision-Feedback Maximum-Likelihood Decoder for Fading Channels*. Proceedings of the IEEE Global Comm. Conf. 332-336, Phoenix, Arizona, 1997.

[10] H.D. Lüke: *Signalübertragung* (7. Auflage). Springer, Berlin/ Heidelberg/New York, 1999.

[11] M. Pätzold: *Mobilfunkkanäle*. Vieweg, Braunschweig/Wiesbaden, 1999

[12] J.G. Proakis: *Digital Communications* (3rd Ed.). McGraw-Hill, New-York, 1995.

[13] M. Riediger, E. Shwedyk: *Communication Receivers Based on Markov Models of the Fading Channel*. IEEE Proc.-Commum. **150** (4), 275-279, 2003.

[14] H. Rohling: *Einführung in die Informations- und Codierungstheorie*. B.G. Teubner, Stuttgart, 1995.

[15] A.J. Viterbi: *Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm*. IEEE Trans. Inform. Theory **IT-13**, 260-269, 1967.

[16] M. Werner: *Nachrichtentechnik* (2. Auflage). Vieweg, Braunschweig, Wiesbaden, 1999.

[17] Q. Zhang, S.A. Kassam: *Finite-State Markov Model for Rayleigh Fading Channels*. IEEE Tran. Comm. **47** (11), 1688-1692, 1999.

Danksagung

An dieser Stelle möchte ich mich herzlich bei Herrn Prof. Stefan Schäffler bedanken, der die fachlichen Anregungen zu dieser Arbeit gegeben hat und mir diese Promotion ermöglicht hat. Mein Dank gilt auch der Siemens AG und insbesondere meinem Fachzentrumsleiter, Herrn Prof. Albert Gilg, für die Erlaubnis, die Ergebnisse, die ich im Rahmen eines Projektes bei Siemens erarbeitet habe, für diese Arbeit verwerten zu dürfen. Ferner möchte ich mich bei allen bedanken, die die Entstehung der Arbeit begleitet und gefördert haben, hier besonders bei Frau Dr. Annelie Stöhr, die mich motiviert hat, am richtigen Punkt den Dingen weiter auf den Grund zu gehen.

Lebenslauf

Name	Georg Hoever
Geburtsdatum	11. Juni 1970
1976 - 1980	Besuch der St. Petrus Grundschule Aldekerk
1980 -1989	Besuch des Friedrich-Spee-Gymnasiums Geldern
18.05.89	Abitur
10/90 - 01/96	Studium der Mathematik an der Universität Karlsruhe (TH)
27.03.92	Diplom-Vorprüfung
25.01.96	Diplom-Hauptprüfung
04/96 - 05/98	Promotion in Mathematik zum Dr. rer. nat. an der Ludwig-Maximilians-Universität München
25.05.98	Promotionsprüfung zum Dr. rer. nat.
04/98 - 09/99	Wissenschaftlicher Assistent an der Universität Regensburg
seit 10/99	Entwicklungsingenieur bei der Siemens AG, Corporate Technology