

Inhaltsverzeichnis

1 Grundlagen	3
1.1 Digitale Nachrichtenübertragung	3
1.2 Kanalcodierung	6
1.3 Der AWGN-Kanal	8
2 Decodierung	10
2.1 Aufgabenstellung	10
2.2 Wortfehleroptimalität	11
2.3 Bitfehleroptimalität	12
2.4 Ausblick: Mobilfunkkanäle	20
Literaturverzeichnis	23

Kapitel 1

Grundlagen

1.1 Digitale Nachrichtenübertragung

Das Ziel der Kommunikation zwischen zwei Kommunikationspartnern ist das Austauschen von Informationen. Zur Übermittlung der Information von dem einen Partner, genannt Quelle, zum anderen Partner, genannt Senke, werden Nachrichten verwendet, die über ein Medium, genannt Kanal, übermittelt werden.

Die Information einer Nachricht ist subjektiv und technisch nicht faßbar, da sie in der Regel nur im Kontext der Kommunikationspartner verständlich ist. Eine Nachricht FEUER etwa übermittelt eine andere Information zwischen den Kommunikationspartnern Brandmelder und Feuerwehr als zwischen den Kommunikationspartnern Offizier und Soldat.

Unter einer Nachricht hingegen wird im technischen Umfeld eine Zeichenfolge aus einem endlichen Zeichenvorrat verstanden, der im allgemeinen beliebig sein kann, etwa Schrift, Bilder, Symbole, Sprache, Klänge, etc. In der digitalen Nachrichtentechnik wird ein binärer Zeichenvorrat verwendet¹.

Bei der Übermittlung der Nachricht über einen physikalischen Kanal, etwa ein Glasfaserkabel, ein Koaxialkabel, ein Compact-Disc-Leselaser, eine Mobilfunk- oder Richtfunkstrecke, wird eine physikalische Repräsentation der Nachricht als Signal notwendig, welches während der Übermittlung physikalischen Störungen unterliegt. Daher erhält die Seite der Senke ein gestörtes Signal, aus dem sich die übertragene Nachricht je nach Störungsgrad eventuell nicht oder verfälscht rekonstruieren läßt.

Aus diesem Grund wird der Nachricht gezielt Redundanz hinzugefügt (durch die sogenannte Kanalcodierung) und diese mitübertragen. Mit Hilfe der Redundanz der empfangenen Signale ist es die Aufgabe des Kanaldecodierers, die ursprüngliche Nachricht zu rekonstruieren.

Die Abbildung 1.1 zeigt den schematischen Aufbau einer digitalen Nachrichtenübertragungsstrecke für Nachrichten, die mit Blockcodes codiert sind.

- **Quelle:** Von der Quelle aus sollen Nachrichten zur Senke geschickt werden.

¹Statt binären Zeichenvorräten (Alphabeten) können auch Alphabete verwendet werden, deren Elemente Wörter aus einem binären Alphabet sind. Die Konstruktion von Reed-Solomon-Codes beruht etwa auf solchen Symbolzeichensätzen, deren Elementzahl eine Zweierpotenz ist.

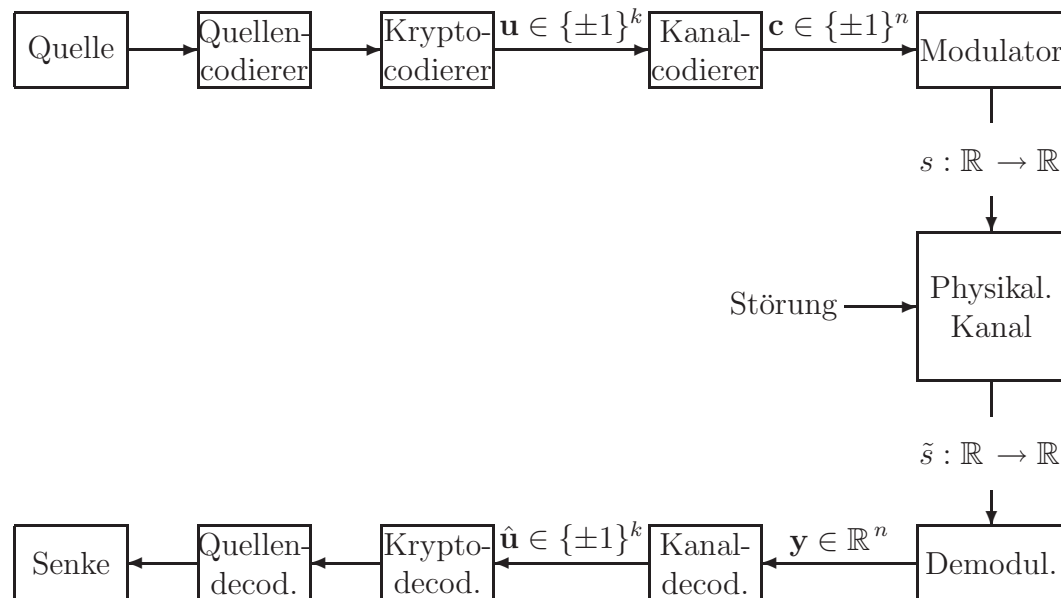


Abbildung 1.1: Digitale Nachrichtenübertragung

Die Darstellung der Nachricht an der Quelle ist beliebig und für die weiteren Betrachtungen nicht relevant. In Mobilfunkanwendungen kann zum Beispiel als Quelle ein Sprecher beziehungsweise Sprache angenommen werden.

- **Quellencodierer:** Der Quellencodierer hat die Aufgabe, die Nachrichten der Quelle so in binäre Wertefolgen (oder Zeichenketten im allgemeinen Fall) zu transformieren, daß weder Informationen verlorengehen noch unnötige Redundanzen codiert werden.
- **Kryptocodierer:** Die optionale Komponente des Kryptocodierers verschlüsselt die vom Quellencodierer kommenden Nachrichten, um lesenden oder schreibenden (verfälschenden) Zugriff von Unbefugten zu verhindern. Die Ausgabe $\mathbf{u} \in \{\pm 1\}^k$ des Kryptocodierers ist ein Binärwort aus dem binären Zeichenvorrat $\{\pm 1\}$. Ein Kryptocodierer bietet häufig die sogenannte „perfekte Sicherheit“ wenn alle kryptocodierten Wörter mit gleicher Wahrscheinlichkeit auftreten.
- **Kanalcodierer:** Da bei der Übertragung der Nachricht über den physikalischen Kanal mit Störungen zu rechnen ist, fügt der Kanalcodierer der Nachricht gezielt Redundanz hinzu, um dem Empfänger die Rekonstruktion der ursprünglichen Nachricht zu ermöglichen. Der Ausgang des Kanalcodierers ist ein kanalcodiertes Wort $\mathbf{c} \in \{\pm 1\}^n$, $n \geq k$, welches zum Beispiel mit Hilfe eines binären linearen Blockcodes aus $\mathbf{u}$ erzeugt werden kann.
- **Modulator:** Die Übertragung der Nachricht über ein physikalisches Medium

erfordert eine physikalische Repräsentation der Nachricht als zeitkontinuierliches Signal (hier dargestellt als Funktion $s : \mathbb{R} \rightarrow \mathbb{R}$).

- **Physikalischer Kanal:** Der physikalische Kanal ist ein (weitestgehend beliebiges) physikalisches Medium, etwa ein Glasfaserkabel, ein Koaxialkabel, ein Compact-Disc-Leselaser, eine Mobilfunk- oder Richtfunkstrecke, etc. Bei der Übertragung des Signals s kommt es zu Störungen, so daß der Kanalausgang ein verfälschtes Signal $\tilde{s}$ ist.
- **Demodulator:** Der Demodulator ist das Gegenstück zum Modulator und demoduliert das empfangene Signal zu einem reellwertigen Vektor $\mathbf{y} \in \mathbb{R}^n$. In diesen Vektor fließen das abgesandte kanalcodierte Wort $\mathbf{c} \in \{\pm 1\}^n$ und die Kanalstörung ein. Da jetzt keine „harten“ Bits mehr vorliegen, spricht man auch von Soft-Werten. Häufig wird im Modulator noch ein Entscheider realisiert, der die reellen Werte zu $+1$ beziehungsweise -1 rundet. Dies ermöglicht zwar die Verwendung sehr effizienter Decodierverfahren (realisiert durch Schieberegister), ist aber durch die Rundung mit einem erheblichen Informationsverlust verbunden.
- **Kanaldecodierer:** Die Aufgabe des Kanaldecodierers besteht in der Rekonstruktion der ursprünglichen Nachricht $\mathbf{u} \in \{\pm 1\}^k$ unter Verwendung der in $\mathbf{y} \in \mathbb{R}^n$ enthaltenen Informationen und dem Wissen über die Kanalcodierung sowie über die Eigenschaften des Störeinflusses. Der Ausgang $\hat{\mathbf{u}} \in \{\pm 1\}^k$ des Kanaldecodierers sollte mit möglichst hoher Wahrscheinlichkeit mit der ursprünglichen Nachricht $\mathbf{u} \in \{\pm 1\}^k$ identisch sein.
- **Kryptodecodierer:** Der Kryptodecodierer ist das Gegenstück zum Kryptocodierer, das heißt, die quellencodierte Nachricht wird hier wieder entschlüsselt.
- **Quellendecodierer:** Der Quellendecodierer bereitet die empfangene Nachricht für die Verarbeitung durch die Senke auf. Im Mobilfunk erzeugt der Quellendecodierer im allgemeinen Sprache.
- **Senke:** Die Senke ist der (beliebig geartete) Empfänger der Nachricht.

Um nun die Performance eines gegebenen digitalen Übertragungssystems (etwa Übertragungsqualität, Energieverbrauch) *im Empfänger* zu verbessern (also ohne jeden Eingriff in die *Rahmenbedingungen* und damit in die definierten *Standards*), sind drei Dinge zu leisten:

- Aufbereitung aller relevanten Informationen über den physikalischen Kanal durch ein stochastisches Kanalmodell.
- Informationstheoretische Interpretation der Ergebnisse der Demodulation (ohne Entscheider).
- Optimale Decodierung unter Berücksichtigung der gegebenen Zeit- und Aufwandsrestriktionen (Übertragungsraten, Chipdesign,...). Da man ohne Entscheider arbeitet, ist der Decodieraufwand im allgemeinen größer als beim

Arbeiten mit Bits. Auf der anderen Seite erhält man einen erheblichen Informationsgewinn.

In Abbildung 1.2 werden die technischen Auswirkungen des Informationsgewinns beim Arbeiten ohne Entscheider veranschaulicht. Dabei gibt es zwei Lesarten:

- Bei einer festen Kanalstörung (als Signal-to-Noise Ratio bezeichnet) werden weniger Decodierungsfehler begangen. Damit kommt es zu einer **Verbesserung der Empfangsqualität**.
- Bei einem festgelegten Fehlerniveau, auch als garantierter Quality-of-Service bekannt, muß weniger Übertragungsenergie aufgewandt werden. Die **Energieeinsparung** kann dabei bis zu 3 Dezibel betragen, was einer Halbierung der Übertragungsenergie entspricht.

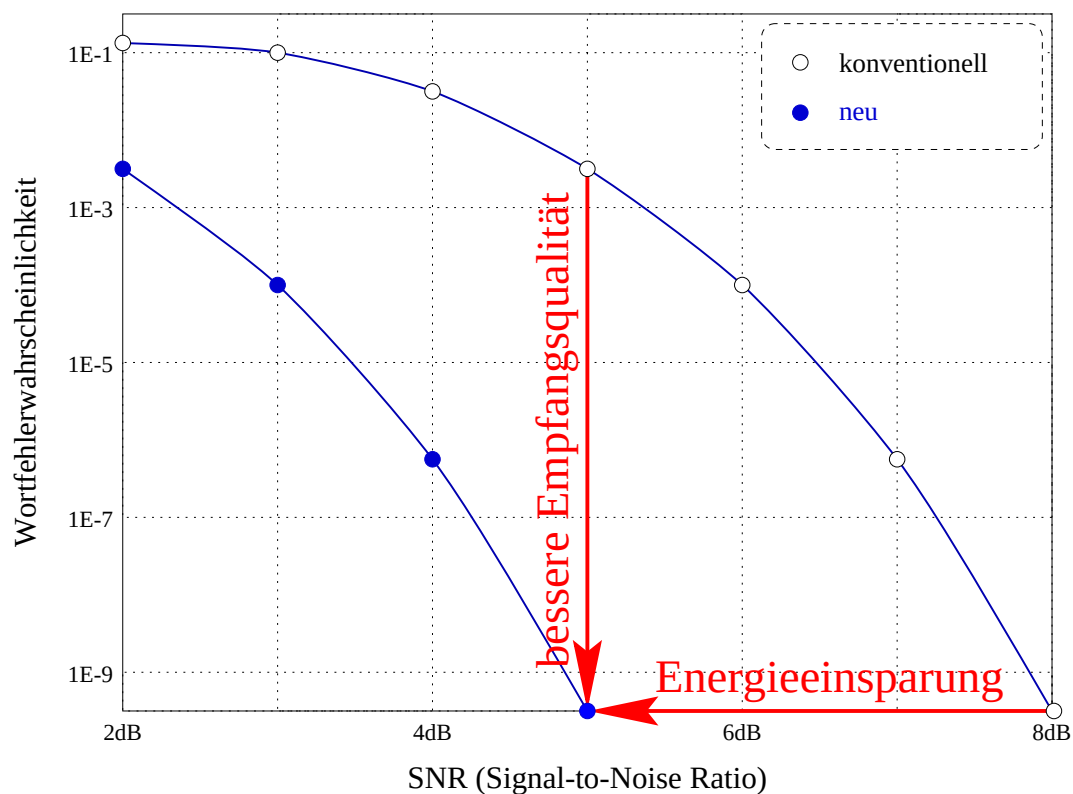


Abbildung 1.2: Verbesserung der Empfangsqualität/Reduktion der Übertragungsenergie

1.2 Kanalcodierung

Um Fehler bei der Übertragung durch einen gestörten physikalischen Kanal korrigieren zu können, fügt der Kanalcodierer jedem Codewort $\mathbf{u} \in \{\pm 1\}^k$ (Ausgabe

des Kryptocodierers) gezielt Redundanz hinzu. Dies kann zum Beispiel dadurch geschehen, daß dem Codewort $\mathbf{u}$ unter festgelegten Regeln $n-k$ Komponenten hinzugefügt werden. Somit erhält man als Ausgabe des Kanalcodierers einen Vektor $\mathbf{c} \in \{\pm 1\}^n$, wobei $c_i = u_i, i = 1, \dots, k$. Für die Berechnung dieser zusätzlichen Komponenten verwendet man die durch die kommutativen Verknüpfungen $\oplus$ und $\odot$ gegebene algebraische Körperstruktur von $\{\pm 1\}$:

$$\begin{array}{rclcl} -1 & \oplus & -1 & = & 1 \\ 1 & \oplus & 1 & = & 1 \\ 1 & \oplus & -1 & = & -1 \\ -1 & \odot & -1 & = & -1 \\ 1 & \odot & -1 & = & 1 \\ 1 & \odot & 1 & = & 1 \end{array} .$$

Für jede zusätzliche Komponente $i \in \{k+1, \dots, n\}$ des zu $\mathbf{u}$ gehörigen Vektors $\mathbf{c}$ wird eine Teilmenge $J_i \subseteq \{1, \dots, k\}$ gebildet und c_i berechnet sich dann zu

$$c_i = \bigoplus_{j \in J_i} u_j, \quad i = k+1, \dots, n. \quad (1.1)$$

Der Kanalcodierer ist somit durch die natürlichen Zahlen k, n und durch die $n-k$ Teilmengen $J_{k+1}, \dots, J_n \subseteq \{1, \dots, k\}$ charakterisiert. Der durch den Kanalcodierer erzeugte Code $\mathcal{C} \subseteq \{\pm 1\}^n$ ist die Menge aller Codewörter $\mathbf{c}$, die durch Anwendung von (1.1) auf alle $\mathbf{u} \in \{\pm 1\}^k$ entsteht. Als Beispiel betrachten wir einen Hamming-Code mit $k = 4, n = 7, J_5 = \{2, 3, 4\}, J_6 = \{1, 3, 4\}$, und $J_7 = \{1, 2, 4\}$. Betrachtet man die Menge $\mathcal{C}$ aller 16 Codewörter dieses Codes und zählt man für je zwei Codewörter die Anzahl der Komponenten, an denen sich diese Codewörter unterscheiden, so stellt man fest, daß sich für diesen Hamming-Code je zwei Codewörter immer an mindestens drei Komponenten unterscheiden. Diese Minimaldistanz $d_{\min}$ eines Codes ist ein wichtiges Gütekriterium. Die eben beschriebene Klasse von Codes wird als die Klasse der binären linearen systematischen Blockcodes bezeichnet, da

- alle Codewörter $\mathbf{c}$ aus der Menge $\{\pm 1\}^n$ sind (binärer Blockcode),
- die ersten k Komponenten von $\mathbf{c}$ den Vektor $\mathbf{u}$ bilden (systematisch),
- die Menge $\mathcal{C}$ aller Codewörter einen k -dimensionalen linearen Unterraum des Vektorraumes $\{\pm 1\}^n$ über $\{\pm 1\}$ bildet (linearer Blockcode).

Die Definition und Beschreibung binärer linearer systematischer Blockcodes wird im wesentlichen mit algebraischen Mitteln durchgeführt. Die wichtigsten Informationen über einen Code werden in dem Tupel (n, k) beziehungsweise in dem Tripel $(n, k, d_{\min})$ zusammengefaßt. Wir werden im folgenden immer von einem binären linearen systematischen Blockcode mit bekannten Teilmengen $J_{k+1}, \dots, J_n$ ausgehen. Die Voraussetzung, daß es sich um einen systematischen Code handelt, ist dabei nicht restriktiv, da zu jedem binären linearen Blockcode ein dazu äquivalenter binärer linearer systematischer Blockcode konstruiert werden kann. Daher wird im

folgenden das Adjektiv 'systematisch' nicht mehr verwendet. Anstelle der Angabe der Teilmengen $J_{k+1}, \dots, J_n$ werden binäre lineare Blockcodes häufig durch eine Generatormatrix $\mathbf{G} \in \{\pm 1\}^{k,n}$ dargestellt. Es gilt dann

$$\mathcal{C} = \{\mathbf{c} \in \{\pm 1\}^n; \mathbf{c}^T = \mathbf{u}^T \mathbf{G}, \mathbf{u} \in \{\pm 1\}^k\}$$

Für den bereits erwähnten Hamming-Code gilt:

$$\mathbf{G} = \begin{pmatrix} -1 & 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & -1 & -1 & -1 \end{pmatrix},$$

wobei die Operationen $\oplus$ und $\odot$ zugrundegelegt sind.

Ist ein Code durch eine Generatormatrix $\mathbf{G} := (g_{i,j})_{\substack{i=1,\dots,k \\ j=1,\dots,n}}$ gegeben, so läßt sich folgendermaßen feststellen, ob dieser Code systematisch ist: Man betrachtet die $k \times k$ Matrix $\mathbf{A} := (a_{i,j})_{i,j=1,\dots,k}$ mit $a_{i,j} = g_{i,j}$ für alle $i, j = 1, \dots, k$. Der Code ist systematisch, falls die Matrix $\mathbf{A}$ bezüglich der Operationen $\oplus$ und $\odot$ die Einheitsmatrix darstellt, also von der Form

$$\mathbf{A} = \begin{pmatrix} -1 & 1 & \dots & 1 \\ 1 & -1 & 1 & \vdots \\ \vdots & 1 & \ddots & 1 \\ 1 & \dots & 1 & -1 \end{pmatrix}$$

ist.

1.3 Der AWGN-Kanal

In diesem Abschnitt betrachten wir ein spezielles stochastisches Kanalmodell, den AWGN-Kanal (Additive White Gaussian Noise). Bei stochastischen Kanalmodellen werden für die Modellierung die Elemente „Modulation“, „physikalischer Kanal + Störung“ und „Demodulator ohne Entscheider“ zu einer Einheit, dem „Kanal“ zusammengefaßt. Wichtige Parameter des AWGN-Kanalmodells sind einerseits die Energie E_b , die man durch Division der für die Übertragung eines Codewortes $\mathbf{c} \in \{\pm 1\}^n$ verwendeten Signalenergie durch die Anzahl k der Informationsbits $\mathbf{u} \in \{\pm 1\}^k$ erhält, und andererseits die einseitige Rauschleistungsdichte N_0 der Störung im physikalischen Kanal. Ein AWGN-Kanal ist nun folgendermaßen charakterisiert:

- Die Ausgabe $\mathbf{c} \in \{\pm 1\}^n$ des Kanalcodierers wird durch eine $\mathcal{N}(\mathbf{0}, \frac{N_0 n}{2E_b k} \mathbf{I}_n)$ normalverteilte Zufallsvariable additiv überlagert, wobei $\mathbf{I}_n$ die n -dimensionale Einheitsmatrix bezeichnet. Das Ergebnis $\mathbf{y} \in \mathbb{R}^n$ des Demodulators ist somit eine Realisierung einer $\mathcal{N}(\mathbf{c}, \frac{N_0 n}{2E_b k} \mathbf{I}_n)$ normalverteilten Zufallsvariablen $\mathbf{Y}$.
- Für Simulationszwecke ist der Quotient $\frac{N_0}{E_b}$ bekannt.

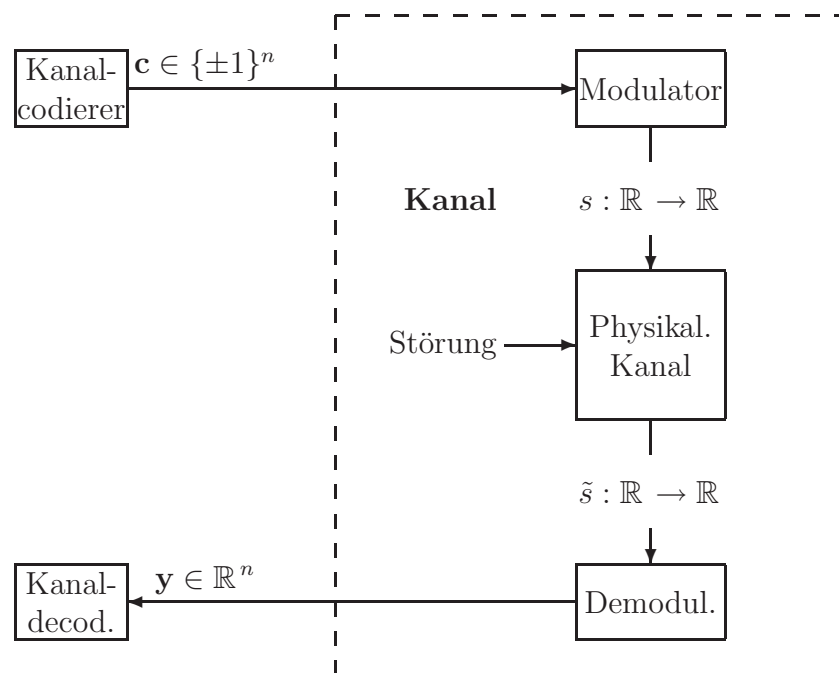


Abbildung 1.3: Kanal

Typisch für stochastische Kanalmodelle ist die Tatsache, daß als Ergebnis der Demodulation einem zu übertragenden Codewort eine Realisierung $\mathbf{y} \in \mathbb{R}^n$ einer speziellen Zufallsvariablen zugeordnet wird. Ein Kanal wird deswegen stochastisch modelliert, weil die Störungen im physikalischen Kanal im Einzelfall nicht vorhersehbar sind, aber in der Vielzahl dennoch gewissen Gesetzmäßigkeiten gehorchen.

Kapitel 2

Decodierung

2.1 Aufgabenstellung

Die Decodieraufgabe besteht mathematisch betrachtet darin, aus der gegebenen Realisierung $\mathbf{y} \in \mathbb{R}^n$ der Zufallsvariablen $\mathbf{Y}$, die in der Praxis der Empfänger liefert, die Informationsbits $\mathbf{u} \in \{\pm 1\}^k$ im noch zu definierenden Sinne „optimal“ zu rekonstruieren. Dies ist eine klassische Aufgabe der mathematischen Statistik und der mathematischen Optimierung. Zur Lösung dieser Aufgabe ist unter spezielle Rahmenbedingungen (Echtzeitanforderungen, Komplexitätsrestriktionen durch Chipdimensionierung) eine Entscheidungsfunktion

$$\begin{aligned}\delta : \mathbb{R}^n &\rightarrow \{\pm 1\}^k, \\ \mathbf{y} &\mapsto \hat{\mathbf{u}} = \delta(\mathbf{y})\end{aligned}$$

zu definieren, die jeder möglichen Realisierung $\mathbf{y} \in \mathbb{R}^n$ eine Rekonstruktion $\hat{\mathbf{u}} \in \{\pm 1\}^k$ der Informationsbits $\mathbf{u} \in \{\pm 1\}^k$ zuordnet. Dabei kann man im wesentlichen zwei verschiedene Strategien verfolgen:

- Wortfehleroptimalität: Berechne $\delta_{WO}(\mathbf{y})$ so, daß die Wahrscheinlichkeit, daß sich in dem rekonstruierten Wort $\hat{\mathbf{u}} \in \{\pm 1\}^k (= \delta_{WO}(\mathbf{y}))$ kein falsches Bit befindet, maximal ist.
- Bitfehleroptimalität: Berechne $\delta_{BO}(\mathbf{y})$ so, daß die Wahrscheinlichkeit für jedes einzelne Bit in $\hat{\mathbf{u}} \in \{\pm 1\}^k (= \delta_{BO}(\mathbf{y}))$, richtig zu sein, maximal ist.

Im Gegensatz zur Bitfehleroptimalität spielt bei der Wortfehleroptimalität die Anzahl der Fehler in $\hat{\mathbf{u}} \in \{\pm 1\}^k$ keine Rolle. Welches Optimalitätskriterium verwendet wird, hängt im allgemeinen von der entsprechenden Anwendung ab. Aufgrund der Komplexität der Aufgabenstellung ist klar, daß die Funktion δ nicht explizit angegeben werden kann, sondern daß die Funktionswerte $\delta(\mathbf{y})$ für jedes im Empfänger auftretende $\mathbf{y} \in \mathbb{R}^n$ durch numerische Verfahren zu berechnen sind.

2.2 Wortfehleroptimalität

Betrachtet man die Wortfehleroptimalität, so ergibt sich $\hat{\mathbf{u}} \in \{\pm 1\}^k$ als Lösung eines diskreten Optimierungsproblems, nämlich der aus der mathematischen Statistik bekannten Maximum Likelihood Schätzung (mit $\|\mathbf{y}\|_2^2 := \sum_{i=1}^n y_i^2$)

$$\hat{\mathbf{u}} = \arg \min_{\mathbf{u} \in \{\pm 1\}^k} \|\mathbf{c}(\mathbf{u}) - \mathbf{y}\|_2^2 = \delta_{WO}(\mathbf{y}).$$

Zu jeder empfangenen Nachricht $\mathbf{y} \in \mathbb{R}^n$ ist jeweils ein Wort $\hat{\mathbf{u}} \in \{\pm 1\}^k$ gesucht mit der Eigenschaft

$$\|\mathbf{c}(\hat{\mathbf{u}}) - \mathbf{y}\|_2^2 \leq \|\mathbf{c}(\mathbf{u}) - \mathbf{y}\|_2^2 \quad \text{für alle } \mathbf{u} \in \{\pm 1\}^k.$$

Um nicht alle 2^k möglichen Codewörter untersuchen zu müssen, was bei großen k nicht realisierbar ist, wird nun ein Branch-and-Bound Algorithmus vorgestellt, der die Zahl der zu untersuchenden Codewörter stark reduziert. Zudem kann der Algorithmus zu jeder Zeit ein bis dahin bestes Codewort als Ergebnis liefern. Der in Abbildung 2.1 dargestellte Baum entspricht einem Entscheidungsbaum für

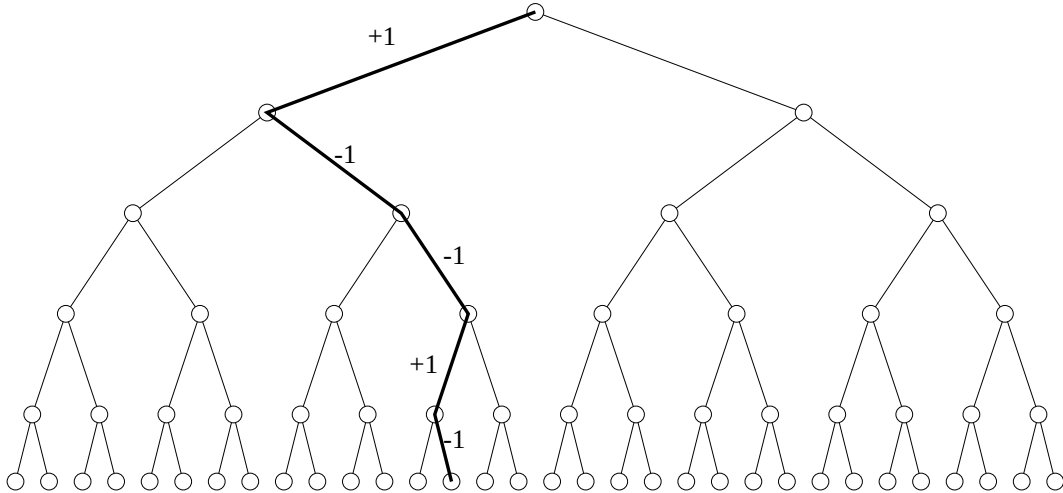


Abbildung 2.1: Branch-and-Bound auf binärem Code

$k = 5$ Informationsbits $u_1, \dots, u_5$, wobei jedes Informationsbit durch eine Ebene im Baum charakterisiert ist. Der angegebene Pfad entspricht somit der Belegung $u_1 = +1, u_2 = -1, u_3 = -1, u_4 = +1, u_5 = -1$. An jeder Verzweigung (Knoten) werden nun untere Schranken für den minimalen Wert der Funktion $\|\mathbf{c}(\mathbf{u}) - \mathbf{y}\|_2^2$ eingetragen, wobei alle Informationsbits bis zu diesem Knoten entsprechend festgelegt sind (Berechnung von Bounds). Betrachtet man zum Beispiel die Verzweigung im angegebenen Pfad der Abbildung 2.1 zwischen $u_2 = -1$ und $u_3 = -1$, so kann die Funktion $\|\mathbf{c}(\mathbf{u}) - \mathbf{y}\|_2^2$ mit $u_1 = +1$ und $u_2 = -1$ für einen systematischen Code (also $c_i = u_i$ für $i = 1, \dots, k$) folgendermaßen nach unten abgeschätzt werden:

$$\|\mathbf{c}(u_1 = +1, u_2 = -1, u_3, u_4, u_5) - \mathbf{y}\|_2^2 \geq (1 - y_1)^2 + (-1 - y_2)^2 + \sum_{i=3}^n (|y_i| - 1)^2.$$

Bewegt man sich auf dem Pfad weiter nach unten, so kann diese Schranke nur schlechter (also größer) werden, da immer mehr Komponenten von $\mathbf{u}$ festgelegt werden. Am Ende des Baumes (also am Blatt) ist $\mathbf{u}$ festgelegt und dort wird dann der entsprechende Funktionswert notiert. Hat man nun für einen festen Wert $\tilde{\mathbf{u}} \in \{\pm 1\}^k$ den Funktionswert $f(\tilde{\mathbf{u}}) = \|\mathbf{c}(\tilde{\mathbf{u}}) - \mathbf{y}\|_2^2$ berechnet, so kann man im Baum von Abbildung 2.1 alle Teilbäume (Branch) streichen, die bei einer Verzweigung starten, deren Schranke größer ist als $f(\tilde{\mathbf{u}})$, denn nach unten werden die Schranken (und damit die Funktionswerte) ja größer. Somit muß man im allgemeinen nicht alle 2^k Möglichkeiten durchprobieren.

Beispiel: Sei $\mathbf{y} = (1.1, -0.5, 0.3, -1.5, 0.7)$ und $\mathbf{c} = (u_1, u_2, u_3, u_1 \oplus u_2, u_2 \oplus u_3)$ und wähle $\tilde{u}_i = \text{sign}(y_i)$, $i = 1, 2, 3$, also $\tilde{u}_1 = +1, \tilde{u}_2 = -1, \tilde{u}_3 = +1$, so gilt:

$$f(\tilde{\mathbf{u}}) = (-0.1)^2 + (-0.5)^2 + 0.7^2 + 0.5^2 + (-1.7)^2 = 3.89.$$

Der Teilbaum beginnend ab $u_1 = -1$ ist wegen $((-1) - 1.1)^2 = 4.41 > 3.89$ schon nicht mehr zu untersuchen.

Manche Codes besitzen eine spezielle Struktur, die jeweils durch ein Trellis-Diagramm gegeben ist. Wendet man das Branch-and-Bound Verfahren auf dieses Trellis-Diagramm an, so nennt man dieses Verfahren Viterbi-Algorithmus. Ein Trellis-Diagramm ist einerseits nicht so komplex wie der obige Entscheidungsbaum, auf der anderen Seite kann das Branch-and-Bound Verfahren auf dem Entscheidungsbaum für alle (systematischen) Blockcodes angewendet werden.

2.3 Bitfehleroptimalität

Für die Bitfehleroptimalität ist für jedes Bit u_i , $i = 1, \dots, k$, folgende Entscheidung zu treffen (die Herleitung ist sehr kompliziert):

$$u_i = \begin{cases} -1 & \text{falls } \ln \left(\frac{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = +1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)}{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = -1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)} \right) < 0 \\ +1 & \text{falls } \ln \left(\frac{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = +1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)}{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = -1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)} \right) \geq 0 \end{cases}.$$

Da die Formeln

$$\ln \left(\frac{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = +1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)}{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_i = -1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)} \right)$$

für jedes $i = 1, \dots, k$ sehr komplex sind ($\mathcal{C}$ bezeichnet die Menge aller 2^k Codewörter), ist man häufig auf Näherungsverfahren angewiesen. Eine mögliche Näherung ergibt sich zum Beispiel durch die Tatsache, daß man die zu berechnenden k reellen Zahlen

$$\beta_1 = \ln \left(\frac{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_1 = +1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)}{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_1 = -1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)} \right), \dots, \beta_k = \ln \left(\frac{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_k = +1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)}{\sum_{\substack{\mathbf{v} \in \mathcal{C} \\ v_k = -1}} \exp \left(-\frac{(\mathbf{y} - \mathbf{v})^T (\mathbf{y} - \mathbf{v})}{\frac{N_0 n}{E_b k}} \right)} \right)$$

folgendermaßen (durch ein nichtlineares Regressionsmodell) approximieren kann:

$$\frac{4E_b k}{N_0 n} \mathbf{y} = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_k \\ \ln \left(\frac{1 + \prod_{j \in J_{k+1}} \frac{\exp(\beta_j) - 1}{\exp(\beta_j) + 1}}{1 - \prod_{j \in J_{k+1}} \frac{\exp(\beta_j) - 1}{\exp(\beta_j) + 1}} \right) \\ \vdots \\ \ln \left(\frac{1 + \prod_{j \in J_n} \frac{\exp(\beta_j) - 1}{\exp(\beta_j) + 1}}{1 - \prod_{j \in J_n} \frac{\exp(\beta_j) - 1}{\exp(\beta_j) + 1}} \right) \end{pmatrix} + \mathbf{e}.$$

Da der Vektor $\mathbf{e}$ ein Maß für die Kanalstörung ist, ist es in Anwendungen ohne Echtzeitanforderungen naheliegend, $\beta_1, \dots, \beta_k$ so zu approximieren, daß $\sum_{i=1}^n e_i^2$ möglichst klein wird. Zu berechnen sind also:

$$\beta_1, \dots, \beta_k = \arg \min_{\mathbf{x} \in \mathbb{R}^k} \left\{ \underbrace{\sum_{i=1}^k \left(x_i - \frac{4E_b k}{N_0 n} y_i \right)^2 + \sum_{i=k+1}^n \left(\ln \left(\frac{1 + \prod_{j \in J_i} \frac{\exp(x_j) - 1}{\exp(x_j) + 1}}{1 - \prod_{j \in J_i} \frac{\exp(x_j) - 1}{\exp(x_j) + 1}} \right) - \frac{4E_b k}{N_0 n} y_i \right)^2}_{\text{Zielfunktion } f} \right\}.$$

Die Zielfunktion f dieses Minimierungsproblems ist beliebig oft stetig differenzierbar. Da

$$f(\mathbf{x}) \geq \sum_{i=1}^k \left(x_i - \frac{4E_b k}{N_0 n} y_i \right)^2$$

für alle $\mathbf{x} \in \mathbb{R}^k$, ist die Existenz eines globalen Minimierers $\mathbf{x}^*$ gesichert.

Im folgenden untersuchen wir iterative Methoden zur lokalen Minimierung einer gegebenen Zielfunktion $f : \mathbb{R}^k \rightarrow \mathbb{R}$, $f \in C^2$. Es werden Algorithmen betrachtet,

die Folgen $\{\mathbf{x}_j\}$, $\{\mathbf{s}_j\}$ und $\{\sigma_j\}$ mit $j \in \mathbb{N}_0$ derart erzeugen, daß für jedes $j \in \mathbb{N}_0$ gilt:

$$\begin{aligned}\mathbf{x}_{j+1} &= \mathbf{x}_j + \sigma_j \mathbf{s}_j \\ f(\mathbf{x}_{j+1}) &< f(\mathbf{x}_j),\end{aligned}$$

wobei $\mathbf{x}_j, \mathbf{s}_j \in \mathbb{R}^k, \sigma_j \in \mathbb{R}^+$. Die Verwendbarkeit dieser Algorithmen entscheidet sich im wesentlichen durch die Wahl der Suchrichtung $\mathbf{s}_j$. Aufgrund der obigen Festlegung ist für jedes $j \in \mathbb{N}_0$

$$\nabla f(\mathbf{x}_j)^T \mathbf{s}_j < 0$$

($\mathbf{s}_j$ sind Abstiegsrichtungen) zu fordern. Ziel ist es, Algorithmen zu konstruieren, die entweder superlinear gegen einen lokalen Minimierer $\bar{\mathbf{x}}$ von f konvergieren, für die also gilt:

$$\lim_{j \rightarrow \infty} \frac{\|\mathbf{x}_{j+1} - \bar{\mathbf{x}}\|_2}{\|\mathbf{x}_j - \bar{\mathbf{x}}\|_2} = 0,$$

oder Algorithmen zu konstruieren, die quadratisch gegen einen lokalen Minimierer $\bar{\mathbf{x}}$ konvergieren, für die also gilt:

$$\lim_{j \rightarrow \infty} \frac{\|\mathbf{x}_{j+1} - \bar{\mathbf{x}}\|_2}{\|\mathbf{x}_j - \bar{\mathbf{x}}\|_2^2} \leq \alpha, \quad \alpha \in \mathbb{R}_0^+.$$

Beim Newton-Verfahren wird die Suchrichtung $\mathbf{s}_j$ als Lösung des linearen Gleichungssystems

$$\nabla^2 f(\mathbf{x}_j) \mathbf{s}_j = -\nabla f(\mathbf{x}_j)$$

für alle $j \in \mathbb{N}_0$ bestimmt. Die Schrittweite σ_j wird im allgemeinen durch den folgenden Algorithmus von Armijo-Goldstein berechnet: Sei $0 < \delta < 0.5$, dann wird das kleinste $\nu \in \mathbb{N}_0$ bestimmt, sodaß mit $\sigma_j := (0.5)^\nu$ gilt:

$$f(\mathbf{x}_j) + \delta \sigma_j \nabla f(\mathbf{x}_j)^T \mathbf{s}_j \geq f(\mathbf{x}_j + \sigma_j \mathbf{s}_j).$$

Befindet man sich nun in einer geeigneten Umgebung eines lokalen Minimierers $\bar{\mathbf{x}}$ von f mit positiv definiter Hessematrix, so ist das Newton-Verfahren durchführbar und die Folge $\{\mathbf{x}_j\}$ konvergiert gegen $\bar{\mathbf{x}}$, wobei nach endlich vielen Schritten stets $\nu = 0$ (Schrittweite gleich 1) gilt. Kann man zudem für die zweite Ableitung von f in dieser Umgebung eine Lipschitzbedingung nachweisen (gegebenenfalls bei $f \in C^3$), so konvergiert die vom Newton-Verfahren erzeugte Folge $\{\mathbf{x}_j\}$ quadratisch gegen $\bar{\mathbf{x}}$. Offensichtlich können mit dieser Methode im allgemeinen nur lokale Minimierer berechnet werden. Die quadratische Konvergenz wird durch einen relativ hohen Aufwand (Berechnung des Gradienten und der Hessematrix in jedem Iterationsschritt) erkaufte. Daher versucht man, die Information zweiter Ordnung durch bisher berechnete Gradienten zu approximieren. Eines der wichtigsten Verfahren in diesem Zusammenhang bildet das BFGS-Verfahren (Broyden, Fletcher, Goldfarb, Shanno-Verfahren). Die Suchrichtung wird durch

$$\mathbf{s}_j = -\mathbf{H}_j \nabla f(\mathbf{x}_j)$$

für alle $j \in \mathbb{N}_0$ berechnet (mit $\mathbf{H}_0 = \mathbf{I}_n$ oder $\mathbf{H}_0$ gleich einer wählbaren, positiv definiten Matrix). Die Matrix $\mathbf{H}_{j+1}$ berechnet sich durch:

$$\mathbf{H}_{j+1} = \mathbf{H}_j + \frac{\mathbf{d}_j^T \mathbf{p}_j + \mathbf{d}_j^T \mathbf{H}_j \mathbf{d}_j}{(\mathbf{d}_j^T \mathbf{p}_j)^2} \mathbf{p}_j \mathbf{p}_j^T - \frac{\mathbf{p}_j \mathbf{d}_j^T \mathbf{H}_j + \mathbf{H}_j \mathbf{d}_j \mathbf{p}_j^T}{\mathbf{d}_j^T \mathbf{p}_j},$$

wobei $\mathbf{d}_j := \frac{\nabla f(\mathbf{x}_{j+1}) - \nabla f(\mathbf{x}_j)}{\|\sigma_j \mathbf{s}_j\|_2}$, $\mathbf{p}_j := \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|_2}$. Die Schrittweite $\sigma_j, j \in \mathbb{N}_0$, wird durch folgenden Algorithmus berechnet: Wähle γ_1 und γ_2 mit $0 < \gamma_1 < \gamma_2 \leq 0.5$ und bestimme σ_j so, daß folgendes gilt:

- $\nabla f(\mathbf{x}_{j+1})^T \mathbf{s}_j \geq \gamma_2 \nabla f(\mathbf{x}_j)^T \mathbf{s}_j$ (somit ist $\mathbf{H}_{j+1}$ positiv definit)
- $f(\mathbf{x}_{j+1}) \leq f(\mathbf{x}_j) + \gamma_1 \sigma_j \nabla f(\mathbf{x}_j)^T \mathbf{s}_j$
- $\sigma_j = 1$, falls möglich.

Ziel der BFGS-Methode ist es, in den positiv definiten Matrizen $\mathbf{H}_j$ die für die Konvergenzgeschwindigkeit relevanten Informationen der zweiten Ableitung von f durch Verwendung der Gradienten

$$\nabla f(\mathbf{x}_0), \nabla f(\mathbf{x}_1), \nabla f(\mathbf{x}_2), \dots$$

zu approximieren. Befindet man sich in einer geeigneten Umgebung eines lokalen Minimierers $\bar{\mathbf{x}}$ von f mit positiv definiter Hessematrix, so konvergiert die vom BFGS-Verfahren erzeugte Folge $\{\mathbf{x}_j\}$ superlinear gegen $\bar{\mathbf{x}}$, falls die zweite Ableitung von f in dieser Umgebung einer Lipschitzbedingung genügt. Die benötigten Auswertungen der Gradienten und Hessematrizen an den entsprechenden Iterationspunkten zur Realisierung des Newton- beziehungsweise BFGS-Verfahrens können durch Automatisches Differenzieren berechnet werden. Auf diese Möglichkeit wird nun genauer eingegangen.

Sei $f : \mathbb{R}^k \rightarrow \mathbb{R}$ eine zweimal stetig differenzierbare Funktion. Mit

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_k} \end{pmatrix} \in \mathbb{R}^k$$

beziehungsweise

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \dots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_k} \\ \vdots & & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_k \partial x_1} & \dots & \frac{\partial^2 f(\mathbf{x})}{\partial x_k \partial x_k} \end{pmatrix} \in \mathbb{R}^{k,k}$$

bezeichnen wir den Gradienten beziehungsweise die Hessematrix der Funktion f an der Stelle $\mathbf{x}$. Ziel ist es nun, einen Algorithmus zu entwickeln, der für eine möglichst allgemeine Funktion f und für jeden Punkt $\mathbf{x} \in \mathbb{R}^k$ als Eingabe das

Tripel $(f(\mathbf{x}), \nabla f(\mathbf{x}), \nabla^2 f(\mathbf{x}))$ als Ausgabe berechnet. Wir beginnen mit einfachen Spezialfällen. Für die konstanten Funktionen $f(\mathbf{x}) \equiv c$ erhalten wir:

$$\nabla f(\mathbf{x}) = \mathbf{0}, \quad \nabla^2 f(\mathbf{x}) = \mathbf{0}$$

für alle $\mathbf{x} \in \mathbb{R}^k$. Ist die Funktion f eine Projektion, also $f(\mathbf{x}) = x_i$, $i \in \{1, \dots, k\}$, so folgt:

$$\nabla f(\mathbf{x}) = i\text{-te Einheitsvektor}, \quad \nabla^2 f(\mathbf{x}) = \mathbf{0}.$$

Unser gesuchter Algorithmus ist also für konstante Funktionen und für Projektionen unmittelbar gegeben und sehr einfach zu implementieren. Diese Überlegungen bilden die Basis für alles Folgende. Wir betrachten nun zwei übliche Methoden, um neue Funktionen aus gegebenen Funktionen zu erhalten, nämlich die rationale Zusammensetzung und die Zusammensetzung mit einer Bibliotheksfunktion. Seien also

$$a : \mathbb{R}^k \rightarrow \mathbb{R} \quad \text{und} \quad b : \mathbb{R}^k \rightarrow \mathbb{R}$$

zweimal stetig differenzierbare Funktionen. Sei ferner $f : \mathbb{R}^k \rightarrow \mathbb{R}$ eine der Funktionen $a + b$, $a - b$, $a \cdot b$ und $a : b$. Bei $a : b$ wird zudem $b > 0$ oder $b < 0$ vorausgesetzt. Somit ist f ebenfalls zweimal stetig differenzierbar und wir erhalten für alle $\mathbf{x} \in \mathbb{R}^k$ im Falle $f = a + b$:

$$\begin{aligned} \nabla f(\mathbf{x}) &= \nabla a(\mathbf{x}) + \nabla b(\mathbf{x}) \\ \nabla^2 f(\mathbf{x}) &= \nabla^2 a(\mathbf{x}) + \nabla^2 b(\mathbf{x}), \end{aligned}$$

im Falle $f = a - b$:

$$\begin{aligned} \nabla f(\mathbf{x}) &= \nabla a(\mathbf{x}) - \nabla b(\mathbf{x}) \\ \nabla^2 f(\mathbf{x}) &= \nabla^2 a(\mathbf{x}) - \nabla^2 b(\mathbf{x}), \end{aligned}$$

im Falle $f = a \cdot b$:

$$\begin{aligned} \nabla f(\mathbf{x}) &= b(\mathbf{x}) \cdot \nabla a(\mathbf{x}) + a(\mathbf{x}) \cdot \nabla b(\mathbf{x}) \\ \nabla^2 f(\mathbf{x}) &= b(\mathbf{x}) \cdot \nabla^2 a(\mathbf{x}) + a(\mathbf{x}) \cdot \nabla^2 b(\mathbf{x}) + \nabla b(\mathbf{x}) \cdot \nabla a(\mathbf{x})^T + \nabla a(\mathbf{x}) \cdot \nabla b(\mathbf{x})^T, \end{aligned}$$

und im Falle $f = a : b$

$$\begin{aligned} \nabla f(\mathbf{x}) &= (\nabla a(\mathbf{x}) - f(\mathbf{x}) \cdot \nabla b(\mathbf{x})) : b(\mathbf{x}) \\ \nabla^2 f(\mathbf{x}) &= (\nabla^2 a(\mathbf{x}) - f(\mathbf{x}) \cdot \nabla^2 b(\mathbf{x}) - \nabla b(\mathbf{x}) \cdot \nabla f(\mathbf{x})^T - \\ &\quad - \nabla f(\mathbf{x}) \cdot \nabla b(\mathbf{x})^T) : b(\mathbf{x}). \end{aligned}$$

Sind nun die Funktionen a , b konstante Funktionen oder Projektionen, so kann für jedes $\mathbf{x} \in \mathbb{R}^k$ das Tripel $(f(\mathbf{x}), \nabla f(\mathbf{x}), \nabla^2 f(\mathbf{x}))$ unmittelbar mit den obigen Rechenregeln aus den Tripeln $(a(\mathbf{x}), \nabla a(\mathbf{x}), \nabla^2 a(\mathbf{x}))$ und $(b(\mathbf{x}), \nabla b(\mathbf{x}), \nabla^2 b(\mathbf{x}))$ berechnet werden. Wir haben somit einen Algorithmus, der als Eingabe die Stelle $\mathbf{x} \in \mathbb{R}^k$, die Tripel $(a(\mathbf{x}), \nabla a(\mathbf{x}), \nabla^2 a(\mathbf{x}))$, $(b(\mathbf{x}), \nabla b(\mathbf{x}), \nabla^2 b(\mathbf{x}))$ und die Operation $\eta \in \{+, -, \cdot, :\}$ benötigt und als Ausgabe das Tripel $(f(\mathbf{x}), \nabla f(\mathbf{x}), \nabla^2 f(\mathbf{x}))$

liefert. Durch wiederholte Anwendung dieses Algorithmus' lassen sich alle rationalen Funktionen behandeln.

Beispiel: $f : \mathbb{R}^2 \rightarrow \mathbb{R}, \mathbf{x} \mapsto \frac{x_1 \cdot x_2}{x_2^2 + 1}$. Wir betrachten die folgenden Funktionen:

$$\begin{aligned} f_1 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_1, \\ f_2 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_2, \\ f_3 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto 1, \\ f_4 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_1 \cdot f_2, \\ f_5 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_2 \cdot f_2, \\ f_6 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_5 + f_3, \\ f_7 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_4 : f_6. \end{aligned}$$

Offensichtlich ist $f = f_7$. Wendet man die oben skizzierte Vorgehensweise auf die Funktionen $f_1, \dots, f_7$ an der Stelle $\tilde{\mathbf{x}} = (0, 0)^T$ an, so erhält man:

$$\begin{aligned} \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, f_1 \right) &\mapsto \left(0, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) = (f_1, \nabla f_1, \nabla^2 f_1)(\tilde{\mathbf{x}}) \\ \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, f_2 \right) &\mapsto \left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) = (f_2, \nabla f_2, \nabla^2 f_2)(\tilde{\mathbf{x}}) \\ \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, f_3 \right) &\mapsto \left(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) = (f_3, \nabla f_3, \nabla^2 f_3)(\tilde{\mathbf{x}}) \\ \left. \begin{aligned} &\left(0, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \\ &\left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \\ &\{\cdot\} \end{aligned} \right\} &\mapsto \left(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) = (f_4, \nabla f_4, \nabla^2 f_4)(\tilde{\mathbf{x}}) \\ \left. \begin{aligned} &\left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \\ &\left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \\ &\{\cdot\} \end{aligned} \right\} &\mapsto \left(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} \right) = (f_5, \nabla f_5, \nabla^2 f_5)(\tilde{\mathbf{x}}) \\ \left. \begin{aligned} &\left(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} \right) \\ &\left(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \\ &\{+\} \end{aligned} \right\} &\mapsto \left(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} \right) = (f_6, \nabla f_6, \nabla^2 f_6)(\tilde{\mathbf{x}}) \end{aligned}$$

$$\left\{ \begin{array}{l} \left(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) \\ \left(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} \right) \\ \{:\} \end{array} \right\} \mapsto \left(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) = (f, \nabla f, \nabla^2 f)(\tilde{\mathbf{x}})$$

Bei der Zusammensetzung mit einer Bibliotheksfunktion betrachtet man zweimal stetig differenzierbare Funktionen

$$\xi : \mathbb{R} \rightarrow \mathbb{R} \quad \text{und} \quad a : \mathbb{R}^k \rightarrow \mathbb{R}.$$

Sei ferner $f : \mathbb{R}^k \rightarrow \mathbb{R}$ gegeben durch $f = \xi \circ a$, wobei ξ eine Bibliotheksfunktion (zum Beispiel $\sin$, $\cos$, $\exp$ u.s.w.) ist und wobei a eine Funktion ist, die wir bereits automatisch differenzieren können, dann ist die Funktion f ebenfalls zweimal stetig differenzierbar und der bisher entwickelte Algorithmus ist unter der Voraussetzung erweiterbar, daß die Ableitungen erster und zweiter Ordnung für die Bibliotheksfunktionen an jeder Stelle algorithmisch auswertbar sind. Es gilt:

$$\begin{aligned} \nabla f(\mathbf{x}) &= \xi'(a(\mathbf{x})) \cdot \nabla a(\mathbf{x}) \\ \nabla^2 f(\mathbf{x}) &= \xi''(a(\mathbf{x})) \cdot \nabla a(\mathbf{x}) \cdot \nabla a(\mathbf{x})^T + \xi'(a(\mathbf{x})) \cdot \nabla^2 a(\mathbf{x}). \end{aligned}$$

Wir erhalten somit einen erweiterten Algorithmus, der als Eingabe die Stelle $\mathbf{x} \in \mathbb{R}^k$, das Tripel $(a(\mathbf{x}), \nabla a(\mathbf{x}), \nabla^2 a(\mathbf{x}))$ und die Funktion ξ benötigt und als Ausgabe das Tripel $(f(\mathbf{x}), \nabla f(\mathbf{x}), \nabla^2 f(\mathbf{x}))$ liefert. Durch wiederholte Anwendung dieses Algorithmus' lassen sich eine große Klasse von Funktionen behandeln.

Beispiel: $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, $\mathbf{x} \mapsto \sin(x_1 \cdot x_2) + \exp(x_2)$. Wir betrachten die folgenden Funktionen:

$$\begin{aligned} f_1 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_1, \\ f_2 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_2, \\ f_3 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_1 \cdot f_2, \\ f_4 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto \sin(f_3), \\ f_5 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto \exp(f_2), \\ f_6 : \mathbb{R}^2 &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_4 + f_5. \end{aligned}$$

Offensichtlich ist $f = f_6$. Wendet man die bisher skizzierte Vorgehensweise auf die Funktionen $f_1, \dots, f_6$ an der Stelle $\tilde{\mathbf{x}} = (\pi, 0)^T$ an, so erhält man:

$$\left(\begin{pmatrix} \pi \\ 0 \end{pmatrix}, f_1 \right) \mapsto \left(\pi, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) = (f_1, \nabla f_1, \nabla^2 f_1)(\tilde{\mathbf{x}})$$

$$\begin{aligned}
 & \left(\left(\begin{pmatrix} \pi \\ 0 \end{pmatrix}, f_2 \right) \mapsto \left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) = (f_2, \nabla f_2, \nabla^2 f_2)(\tilde{\mathbf{x}}) \right. \\
 & \left. \left(\begin{pmatrix} \pi, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \right\} \\
 & \left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \Bigg\} \mapsto \left(0, \begin{pmatrix} 0 \\ \pi \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) = (f_3, \nabla f_3, \nabla^2 f_3)(\tilde{\mathbf{x}}) \\
 & \quad \quad \quad \{\cdot\} \\
 & \left(0, \begin{pmatrix} 0 \\ \pi \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) \Bigg\} \mapsto \left(0, \begin{pmatrix} 0 \\ \pi \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) = (f_4, \nabla f_4, \nabla^2 f_4)(\tilde{\mathbf{x}}) \\
 & \quad \quad \quad \{\sin\} \\
 & \left(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \Bigg\} \mapsto \left(1, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right) = (f_5, \nabla f_5, \nabla^2 f_5)(\tilde{\mathbf{x}}) \\
 & \quad \quad \quad \{\exp\} \\
 & \left(0, \begin{pmatrix} 0 \\ \pi \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right) \Bigg\} \\
 & \left(1, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right) \Bigg\} \mapsto \left(1, \begin{pmatrix} 0 \\ 1 + \pi \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \right) = (f, \nabla f, \nabla^2 f)(\tilde{\mathbf{x}}) \\
 & \quad \quad \quad \{+\}
 \end{aligned}$$

Hat man also die Funktion $f : \mathbb{R}^k \rightarrow \mathbb{R}$ durch eine charakterisierende Sequenz

$$\begin{aligned}
 & f_1 : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_1, \\
 & & \dots & \\
 & f_k : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto x_k, \\
 & f_{k+1} : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto c_1, \\
 & & \dots & \\
 & f_{k+d} : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto c_d, \\
 & f_{k+d+1} : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_{\alpha_i} \eta_i f_{\gamma_i}, \\
 & \eta_i \in \{+, -, \cdot, :\}, & \alpha_i, \gamma_i &\in \{1, \dots, k+d\} \\
 & & \text{oder} & \\
 & f_{k+d+1} : \mathbb{R}^k \rightarrow \mathbb{R}, & \mathbf{x} &\mapsto \xi(f_{\gamma_i}) \\
 & \xi \text{ ist eine Bibl.-Fkt.}, & \gamma_i &\in \{1, \dots, k+d\} \\
 & & \dots &
 \end{aligned}$$

$$\begin{aligned}
f_{k+d+s} : \mathbb{R}^k &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto f_{\alpha_i} \eta_i f_{\gamma_i}, \\
\eta_i &\in \{+, -, \cdot, : \}, & \alpha_i, \gamma_i &\in \{1, \dots, k+d+s-1\} \\
&& \text{oder} & \\
f_{k+d+s} : \mathbb{R}^k &\rightarrow \mathbb{R}, & \mathbf{x} &\mapsto \xi(f_{\gamma_i}) \\
\xi \text{ ist eine Bibl.-Fkt.}, & & \gamma_i &\in \{1, \dots, k+d+s-1\} \\
f_{k+d+s} &= & f &
\end{aligned}$$

dargestellt, wobei $c_1, \dots, c_d$ reelle Konstanten bezeichnen, so ist diese Funktion f durch den eben entwickelten Algorithmus an jeder Stelle $\mathbf{x} \in \mathbb{R}^k$ automatisch differenzierbar. Für die computergestützte Erstellung charakterisierender Sequenzen gibt es verschiedene Compiler mit unterschiedlichen Eingabeformaten für die Funktion f .

Algorithmen zur globalen Optimierung sind nur mit Resultaten der stochastischen Analysis herzuleiten und numerisch sehr aufwendig. Eine interessante Anwendung dieser numerischen Verfahren ist durch die Funkübertragung von Raumsonden gegeben. Um etwa Bilder vom Neptun zu übertragen, hat die Raumsonde Voyager mit einer Sendeleistung von 18,2 Watt lediglich die Leistung einer Taschenlampe zur Verfügung. Bei einer Strecke von $4,4 \cdot 10^9$ km und einem Signal-Rausch-Verhältnis $\frac{E_b}{N_0}$ von 2,5 dB beträgt die empfangene Energie pro Binärzeichen auf der Erde $9,5 \cdot 10^{-22}$ Wattsekunden. Eine Einsparung bei $\frac{E_b}{N_0}$ von 1dB (etwa durch verbesserte Decodierverfahren) führt heute zu einer Kostenersparnis von etwa 75 Millionen Dollar. Die entsprechend höhere numerische Komplexität ist anhand dieser Fakten (insbesondere der Flugzeit von 12 Jahren von der Erde bis zum Neptun) von untergeordneter Bedeutung, da für dieses Anwendungsbeispiel keine strikten zeitlichen Vorgaben für die Decodierung gegeben sind. Mit Decodierverfahren basierend auf globaler Optimierung können bis zu 3dB Energieeinsparung bei gleicher Übertragungsqualität gegenüber klassischen Decodierverfahren (Schieberegister) erreicht werden.

2.4 Ausblick: Mobilfunkkanäle

Beim Mobilfunk gelangen die vom Sender abgestrahlten elektromagnetischen Wellen aufgrund von Hindernissen häufig nicht (nur) über den direkten Weg zur Empfangsantenne. Vielmehr setzen sich die empfangenen Wellen vorwiegend aus einer Vielzahl von reflektierten, gebeugten und gestreuten Wellen zusammen. Ein typisches Beispiel für diese so genannte Mehrwegeausbreitung ist in Abbildung 2.2 für den terrestrischen Mobilfunkkanal dargestellt. Bedingt durch die Mehrwegeausbreitung besteht das Empfangssignal aus einer Überlagerung von unterschiedlich stark gedämpften, verzögerten und phasenverschobenen Versionen des Sendesignals, welche sich gegenseitig auslöschen beziehungsweise verstärken. Daher kann je nach Phasenlage der einzelnen Teilwellen die Überlagerung konstruktiv oder destruktiv erfolgen. Außerdem kommt es durch die Mehrwegeausbreitung bei der digitalen

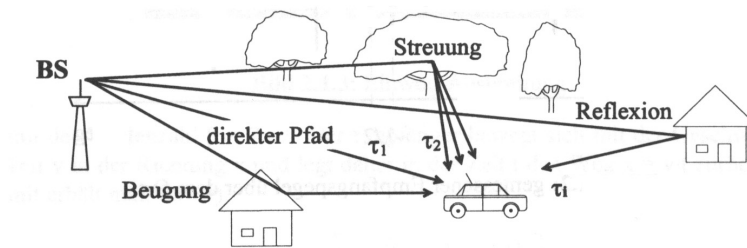


Abbildung 2.2: Mehrwegeausbreitung

Datenübertragung zu einer Veränderung der Impulsform bis hin zu mehreren, einzeln trennbaren Impulsen. Dieser Effekt wird als Impulsdispersion bezeichnet. Die Größe der Impulsdispersion hängt von den Laufzeitdifferenzen und den Amplitudenverhältnissen der Teilwellen ab.

Neben der Mehrwegeausbreitung wirkt sich auch der Dopplereffekt negativ auf die Übertragungseigenschaften des Mobilfunkkanals aus. Durch die Bewegung der Mobilstation kommt es aufgrund des Dopplereffektes zu einer Frequenzverschiebung der einzelnen Teilwellen. Der Einfallswinkel - definiert durch die Einfallsrichtung der Teilwelle und die Bewegungsrichtung der Mobilstation (Abbildung 2.3) - bestimmt die Dopplerfrequenz (Frequenzverschiebung) einer Teilwelle. Bedingt durch den

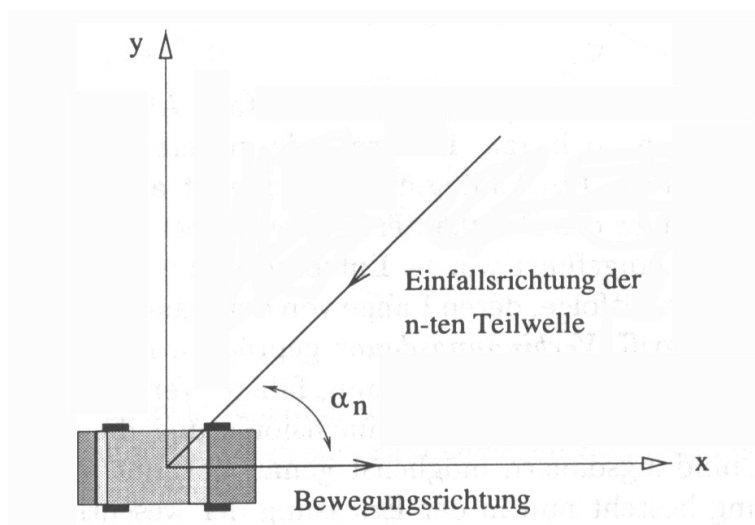


Abbildung 2.3: Dopplereffekt

Dopplereffekt erfährt das Spektrum eines Sendesignals auf dem Übertragungsweg eine Frequenzverbreiterung. Dieser Effekt wird als Frequenzdispersion bezeichnet. Die Größe der Frequenzdispersion hängt im wesentlichen von der maximalen Dopplerfrequenz und den Amplituden der einzelnen empfangenen Teilwellen ab. Im Zeitbereich äußert sich der Dopplereffekt dadurch, daß die Impulsantwort des Kanals zeitvariant wird.

Moderne Methoden zur stochastischen Modellierung von Mobilfunkkanälen zeichnen sich insbesondere dadurch aus, daß sie in der Lage sind, nicht nur die Wahrscheinlichkeitsdichten der Amplitude und Phase zu modellieren, sondern auch Aussagen über die Pegelunterschreitungsrates und die mittlere Fadingdauer bereitzustellen, die für Bündelfehler verantwortlich sind. Da zudem eine additive Überlagerung mit einem weißen Rauschen als Störung zu berücksichtigen ist, wird das Empfangssignal somit durch die Verknüpfung verschiedener stochastischer Prozesse als sehr komplexer stochastischer Prozeß mit parameterabhängiger Kovarianzstruktur modelliert. Das vollständige stochastische Kanalmodell erhält man schließlich durch Berücksichtigung des Demodulators, was im allgemeinen einer komplizierten stochastischen Integration entspricht.

Letztendlich erhält man als Ergebnis des Empfängers die Realisierung einer komplizierten Zufallsvariablen mit parameterabhängiger Kovarianzstruktur. Die entsprechenden Parameter sind online in speziellen zeitlichen Abständen zu schätzen. Daher ist es heute immer noch üblich, Decodierverfahren nur im Zusammenhang mit AWGN-Kanälen und unter Verwendung ganz spezieller Codeklassen zu untersuchen. Die wesentlichen Unterschiede zwischen AWGN- und Mobilfunkkanälen lassen sich folgendermaßen zusammenfassen:

- Bei Mobilfunkkanälen ist es im allgemeinen nicht möglich, die Verteilungen der Zufallsvariablen, die das Ergebnis des Empfängers repräsentieren, exakt zu berechnen. Man ist also auf statistische Kenngrößen oder numerische Approximationen angewiesen.
- Die statistischen Kenngrößen der eben erwähnten Verteilungen sind nicht zeitinvariant, müssen also im Kanalmodell als Parameter modelliert werden und sind vom Decodierer in gewissen zeitlichen Abständen neu anzupassen.

Obwohl man heute versucht, diese Schwierigkeiten - etwa durch den Einsatz von Equalizern - zu umgehen, bleiben diese Probleme relevant, da einerseits die verwendeten Equalizer - aus Sicht der Mathematik - nicht optimal sind, andererseits die Aufgabe des Equalizers idealerweise durch die Decodierung ohne großen Zusatzaufwand übernommen werden könnte.

Literaturverzeichnis

- H. Bauer, *Wahrscheinlichkeitstheorie*. de Gruyter-Verlag, 4. Auflage 1991.
- A. Beutelspacher, *Kryptologie*. Vieweg-Verlag, 3. Auflage 1993.
- J. F. Böhme, *Stochastische Signale*. Teubner-Verlag, 1993.
- M. Bossert, *Kanalcodierung*. Teubner-Verlag, 1992.
- W. B. Davenport und W. L. Root, *An Introduction to the Theory of Random Signals and Noise*. McGraw-Hill, 1958.
- H. Fischer, *Automatic Differentiation: Parallel computation of function, gradient, and Hessian matrix*. Parallel Computing **13**, 1990.
- H. Fischer, *Automatisches Differenzieren*. In: J. Herzberger(Ed.), *Eine Einführung in das Scientific Computing*. Akademie-Verlag, 1995.
- H. Fischer, B. Riedmüller und S. Schäffler(Eds.), *Applied Mathematics and Parallel Computing*. Physica-Verlag, 1996.
- H. Fischer und H. Warsitz, *Complexity investigations concerning the derivative of a rational function*. Technical Report 138, IAMS, TU München, 1995.
- R. Fletcher, *Practical Methods of Optimization*. Wiley, 1993.
- B. Friedrichs, *Kanalcodierung*. Springer-Verlag, 1996.
- W. Fumy und H. P. Rieß, *Kryptographie*. Oldenbourg-Verlag, 2. Auflage 1994.
- C. Großmann und J. Terno, *Numerik der Optimierung*. Teubner-Verlag, 1993.
- Hänsler, *Statistische Signale*. Springer-Verlag, 1991.
- J. Hagenauer und P. Hoher, *Concatenated Viterbi-decoding*. in: Proc. 4. Joint Swedish-Soviet Int. Workshop on Inf. Theory, Gotland, Sweden, 1989.

- J. Hagenauer, E. Offer und L. Papke, *Iterative Decoding of Binary Block and Convolutional Codes*. IEEE Transactions on Information Theory **42**, 1996.
- C. R. P. Hartmann und L. D. Rudolph, *An Optimum Symbol-by-Symbol Decoding Rule for Linear Codes*. IEEE Transactions on Information Theory **22**, 1976.
- W. Heise und P. Quattrocchi, *Informations- und Codierungstheorie*. Springer-Verlag, 3. Auflage 1995.
- J. Herzberger(Ed.), *Eine Einführung in das Scientific Computing*. Akademie-Verlag, 1995.
- P. Kosmol, *Methoden zur numerischen Behandlung nichtlinearer Gleichungen und Optimierungsaufgaben*. Teubner-Verlag, 1993.
- J. Lodge, P. Hoeher und J. Hagenauer, *The decoding of multidimensional codes using separable MAP 'filters'*. in: Proc. Biennial Symp. on Communications, Queen's University, Kingston, Canada, 1992.
- J. Lodge, R. Young, P. Hoeher und J. Hagenauer, *Separable MAP 'filters' for the decoding of product and concatenated codes*. in: Proc. IEEE Int. Conf. on Comm., Geneva, Switzerland, 1993.
- H. D. Lüke, *Signalübertragung*. Springer-Verlag, 6. Auflage 1995.
- J. G. Proakis, *Digital Communications*. McGraw-Hill, 3. Auflage 1995.
- K. Ritter und S. Schäffler, *A Stochastic Method for Constrained Global Optimization*. SIAM Journal on Optimization **4**, 1994.
- H. Rohling, *Einführung in die Informations- und Codierungstheorie*. Teubner-Verlag, 1995.
- S. Schäffler, *Global Optimization Using Stochastic Integration*. Roderer-Verlag, 1995.
- S. Schäffler, *Unconstrained Global Optimization Using Stochastic Integration*. Optimization **35**, 1995.
- S. Schäffler, *Large Scale Global Optimization on Transputer Networks*. In: H. Fischer, B. Riedmüller und S. Schäffler(Eds.), *Applied Mathematics and Parallel Computing*. Physica-Verlag, 1996.
- S. Schäffler und T. F. Sturm, *Wahrscheinlichkeitstheorie und Statistik I für*

Mathematiker. Skriptenreihe des IAMS, Band 5, TU München, 1994.

S. Schäffler und T. F. Sturm, *Wahrscheinlichkeitstheorie und Statistik II für Mathematiker*. Skriptenreihe des IAMS, Band 6, TU München, 1995.

G. A. F. Seber und C. J. Wild, *Nonlinear Regression*. Wiley, 1989.

A. N. Širjaev, *Wahrscheinlichkeit*. VEB-Verlag, 1988.

P. Spellucci, *Numerische Verfahren der nichtlinearen Optimierung*. Birkhäuser-Verlag, 1993.

A. Törn und A. Žilinskas, *Global Optimization*. North-Holland, 1987.

M. Ulbrich und S. Ulbrich, *Automatic Differentiation: A Structure-Exploiting Forward Mode with Almost Optimal Complexity for Kantorovič Trees*. In: H. Fischer, B. Riedmüller und S. Schäffler(Eds.), *Applied Mathematics and Parallel Computing*. Physica-Verlag, 1996.