UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE COMUNICAÇÕES

Técnicas Autodidatas e Soluções de Baixa Complexidade para Equalização e Estimação Turbo

Autor
Murilo Bellezoni Loiola

Orientador
Prof. Dr. João Marcos Travassos Romano
Co-orientador
Dr. Renato da Rocha Lopes

Banca Examinadora:

- Prof. Dr. João Marcos Travassos Romano (FEEC/UNICAMP)
- Prof. Dr. Bartolomeu Ferreira Uchôa Filho (EEL/UFSC)
- Prof. Dr. Amauri Lopes (FEEC/UNICAMP)
- Prof. Dr. Dalton Soares Arantes (FEEC/UNICAMP)

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Campinas, 23 de fevereiro de 2005

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Loiola, Murilo Bellezoni
L834t Técnicas autodidatas e soluções de baixa complexidade para equalização e estimação turbo / Murilo Bellezoni Loiola. --Campinas, SP: [s.n.], 2005.

Orientadores: João Marcos Travassos Romano, Renato da Rocha Lopes.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Filtros digitais (Matemática). 2. Identificação de sistemas. 3. Equalizadores (Eletrônica). 4. Processamento de sinais. 5. Teoria da Codificação. 6. Sistemas difusos. I. Romano, João Marcos Travassos. II. Lopes, Renato da Rocha. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Titulo em Inglês: Blind techniques and low-complexity solutions for turbo equalization and estimation.
Palavras-chave em Inglês: Digital filters (mathematics), System identification, Equalizers (electronics), Signal processing, Coding theory e Fuzzy systems
Área de concentração: Telecomunicações e Telemática
Titulação: Mestre em Engenharia Elétrica
Banca examinadora: Bartolomeu Ferreira Uchoa Filho, Amauri Lopes e Dalton Soares Arantes
Data da defesa: 23/02/2005

Resumo

Este trabalho considera o uso de receptores iterativos, que realizam conjuntamente as tarefas de equalização, decodificação e estimação de canal, para mitigar os efeitos introduzidos pelo meio de transmissão no sinal enviado. Primeiramente, realizamos um estudo comparativo de equalizadores turbo com complexidade exponencial e com complexidade reduzida, procurando estabelecer as vantagens e limitações de cada um. Em seguida, abordamos o problema da estimação cega de canais de comunicação em um contexto iterativo. Para que o estimador de canal possa se beneficiar da robustez introduzida pelo código corretor de erros, incluímos os algoritmos de estimação de canal na malha de realimentação do equalizador turbo. Propusemos então a utilização de algoritmos de mínimos quadrados rápidos para realizar a estimação e mostramos, através de simulações, as vantagens em se utilizar tal esquema. Por fim, tendo em vista a aplicação específica de equalização de canais da rede elétrica, propusemos um equalizador *fuzzy* iterativo cujo desempenho supera o de um equalizador *fuzzy* convencional.

Abstract

This work concerns the use of iterative receivers, which jointly perform equalization, decoding and channel estimation, to mitigate the impairments introduced by the transmission medium into the transmitted signal. First, we make a comparative study of exponential-and reduced-complexity turbo equalizers and establish the advantages and the limitations of each one. The problem of blind channel estimation is dealt with in the sequel. In order to make the channel estimator benefit from the error correction capabilities of the codes, we introduce the channel estimation algorithms in the feedback loop of the turbo equalizers. Then, we propose the use of fast least squares algorithms to estimate the channel and show by simulations the advantages of using such approach. Finally, regarding the specific application of power-line channel equalization, we propose an iterative fuzzy equalizer, which outperforms a conventional fuzzy equalizer.

Aos meus pais,
Manoel e Rita

Agradecimentos

Talvez tão difícil quanto escrever esta dissertação foi encontrar palavras apropriadas para expressar minha gratidão a todos que, direta ou indiretamente, contribuíram com esta realização. Espero que esses agradecimentos possam, ao menos, retribuir um pouco da atenção que me foi dada.

Aos meus queridos pais, Manoel e Rita, por todo o carinho e compreensão. Certamente palavras são insuficientes para agradecer aqueles sem os quais nada disso teria sido possível ou teria valido a pena e cujo amor sempre me inspirou e incentivou.

Às minhas irmãs, Edna e Ednéia, pelo incentivo e zelo. Com elas, meu prazer em viver é certamente muito maior.

Ao meu orientador João Marcos Travassos Romano, pela orientação dedicada, de uma qualidade técnica inquestionável, e pela amizade e convivência extremamente agradável desde os tempos da iniciação científica.

Ao meu co-orientador Renato da Rocha Lopes, pela amizade, por todas as discussões, técnicas ou não, e pelas valiosas sugestões que muito contribuíram para o desenvolvimento desta dissertação. Sua enorme contribuição faz, no meu entendimento, este trabalho tão dele quanto meu.

Ao professor Bartolomeu Ferreira Uchôa Filho, pela disponibilidade, pelo interesse demonstrado e pelo rigor ao elaborar suas sugestões.

Ao professor Amauri Lopes, por suas valiosas sugestões e pela revisão criteriosa deste trabalho.

Ao professor Dalton Soares Arantes, pelas sugestões e discussões enriquecedoras durante o todo desenvolvimento desta dissertação.

Aos amigos Rafael Ferrari e Ricardo Suyama, pelo companheirismo desde o início da graduação.

Aos demais amigos do Laboratório de Processamento de Sinais para Comunicações: Aline, Charles, Cristiano Cruz, Cristiano Panazio, Cris, Cynthia, Danilo, Dayan, Glauco, Gustavo, Leonardo, Maurício Sol, Moisés, Romis e Tarciano; pela ajuda e pelos agradáveis momentos que passamos juntos.

Aos funcionários da FEEC, pelo apoio indispensável.

Aos amigos da turma de graduação EE97, por todo o apoio e incentivo.

À Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP, pelo apoio financeiro.

Conteúdo

Lista de Figuras

Abreviaturas

APP:	*A Posteriori Probabilities* – Probabilidades *a posteriori*
AWGN:	*Additive White Gaussian Noise* – Ruído aditivo branco e gaussiano
BCJR:	*Bahl, Cocke, Jelinek e Raviv*
BER:	*Bit Error Rate* – Taxa de erro de bit
BPSK:	*Binary Phase Shift Keying*
COST:	*European Co-operation in the field of Scientific and Technical research*
DFE:	*Decision Feedback Equalizer*
E_b/N_0:	Razão entre a energia de bit e a energia de ruído
FAEST:	*Fast a Posteriori Error Sequential Technique*
FIR:	*Finite Impulse Response* – Resposta ao impulso finita
FK:	*Fast Kalman* – Algoritmo rápido de Kalman
FTF:	*Fast Transversal Filter* – Filtro transversal rápido
GSM:	*Global System for Mobile communications*
HFAEST:	*Hard FAEST* – Algoritmo FAEST alimentado por decisões abruptas
HFK:	*Hard FK* – Algoritmo FK alimentado por decisões abruptas
HFTF:	*Hard FTF* – Algoritmo FTF alimentado por decisões abruptas
HLMS:	*Hard LMS* – Algoritmo LMS alimentado por decisões abruptas
HMKF:	*Hard MKF* – Algoritmo MKF alimentado por decisões abruptas
IIS:	Interferência Intersimbólica

LLR:	*Log-Likelihood Ratio*
LMS:	*Least-Mean-Squares*
MAP:	*Maximum a Posteriori Probability* – Máxima probabilidade a posteriori
MLSE:	*Maximum Likelihood Sequence Estimation* – Estimação de seqüência de máxima verossimilhança
MKF:	*Modified Kalman Filter* – Filtro de Kalman Modificado
MSE:	*Mean Squared Error* – Erro quadrático médio
MMSE:	*Minimum Mean Squared Error* – Erro quadrático médio mínimo
MIMO:	*Multiple-Input Multiple-Output* – Várias entradas e várias saídas
PLC:	*Power Line Communications*
PSK:	*Phase Shift-Keying*
QAM:	*Quadrature Amplitude Modulation* – Modulação de Amplitude em Quadratura
PSP:	*Per-Survivor Processing* – Processamento por sobrevivente
RBF:	*Radial Basis Functions* – Funções de base radial
RLS:	*Recursive Least-Squares*
RSC:	*Recursive Systematic Convolutional*
SFAEST:	*Soft FAEST* – Algoritmo FAEST alimentado por decisões suaves
SFE:	*Soft-Feedback Equalizer*
SFK:	*Soft Fast Kalman* – Algoritmo FK alimentado por decisões suaves
SFTF:	*Soft FTF* – Algoritmo FTF alimentado por decisões suaves
SMKF:	*Soft MKF* – Algoritmo MKF alimentado por decisões suaves
SISO:	*Soft-Input Soft-Output* – Entrada e saída suaves
SLMS:	*Soft LMS* – Algoritmo LMS alimentado por decisões suaves
SNR:	*Signal-to-Noise Ratio* – Relação sinal-ruído
SOVA:	*Soft Output Viterbi Algorithm* – Algoritmo de Viterbi com saída suave
WSSUS:	*Wide Sense Stationary Uncorrelated Scattering*

1

Introdução

Desde os primórdios da civilização, a capacidade de se comunicar teve papel fundamental no convívio e desenvolvimento social de nossa espécie. A cada etapa do desenvolvimento humano, a necessidade de comunicação entre pontos cada vez mais distantes impulsionava a evolução dos sistemas de telecomunicações. O progresso destes sistemas foi tão grande que, hoje, podemos transmitir informações rapidamente entre praticamente quaisquer pontos do planeta. Radiodifusão de sinais de TV, telefones celulares, conexão de alta velocidade à Internet, auxílio à navegação, transmissão de dados por fibras ópticas, por satélites e pela rede elétrica são apenas alguns exemplos que mostram a importância e a influência que os sistemas de comunicações atuais exercem em nossas vidas.

Seja para o trabalho ou para o entretenimento, a utilização cada vez mais intensa de sistemas de comunicações digitais exige a busca por técnicas capazes de promover

um aumento da taxa de transmissão de dados e uma melhora na qualidade do sinal recebido. Podemos dizer que o principal obstáculo para a concretização desses objetivos é o canal de comunicação, o meio físico pelo qual se propaga o sinal que carrega as informações. Além de introduzir ruído, o canal pode fazer com que o sinal transmitido sofra um espalhamento temporal, fazendo com que dados enviados num determinado instante passem a interferir com dados transmitidos em outros instantes [1]. Este fenômeno, chamado de interferência intersimbólica (IIS), é um dos principais fatores que limitam o desempenho dos sistemas de comunicações, provocando redução da confiabilidade e/ou da taxa de transmissão. Exemplos de fatores responsáveis pelo surgimento da IIS são a propagação por múltiplos percursos, como em sistemas de comunicações móveis, e a largura de faixa limitada, como em canais telefônicos.

Para aumentar a imunidade do sinal transmitido ao ruído, freqüentemente são utilizados códigos corretores de erros [2]. Estes códigos acrescentam uma redundância controlada à mensagem enviada, possibilitando ao receptor detectar e/ou corrigir erros ocorridos durante a transmissão. Graças a essa capacidade de correção, os sistemas de comunicação digital codificados podem operar com uma potência de transmissão menor que os sistemas não codificados. Já para minimizar os efeitos da IIS, o receptor normalmente emprega um filtro denominado equalizador [1,3].

Em esquemas clássicos de recepção, os equalizadores não fazem uso da redundância introduzida pela codificação de canal, sendo equalização e decodificação realizadas separadamente. Uma alternativa interessante para a melhoria significativa do desempenho do receptor surgiu com a equalização turbo [4]. Nesta nova abordagem, inspirada na decodificação dos chamados códigos turbo, o equalizador utiliza a saída do decodificador como uma informação *a priori* sobre os símbolos enviados. Desta forma, o equalizador pode aprimorar suas saídas, que serão empregadas pelo decodificador para produzir estimativas mais refinadas da mensagem transmitida, recomeçando o ciclo. É exatamente esta troca iterativa de informações entre equalizador e decodificador que faz com que os equalizadores turbo alcancem uma taxa de erro de bit muito menor que aquela obtida pelos receptores não-iterativos con-

vencionais.

Os equalizadores turbo normalmente usam o algoritmo MAP (*Maximum a Posteriori Probabilities*) desenvolvido por Bahl, Cocke, Jelinek e Raviv (BCJR) [5] tanto para a equalização quanto para a decodificação. O grande inconveniente deste algoritmo, quando usado como equalizador, é que sua complexidade computacional cresce exponencialmente com a memória do canal, tornando-o proibitivo para canais com uma resposta ao impulso longa. Este fato tem motivado a busca por equalizadores de baixa complexidade como aqueles em [6–8], que utilizam, com sucesso, filtros lineares para equalizar o sinal recebido. Dentre os equalizadores turbo estudados neste trabalho, os de complexidade reduzida receberão atenção especial.

Em geral, os equalizadores turbo supõem que o canal é conhecido. No entanto, em situações reais, como nas comunicações móveis, o canal nem sempre é conhecido. Assim, é necessário estimar os parâmetros desconhecidos para que se possa continuar empregando os equalizadores turbo. Esta estimação pode ou não ser realizada através de uma seqüência de treinamento, que nada mais é que um conjunto de símbolos conhecido pelo receptor e incorporado à mensagem transmitida. Uma desvantagem do uso de seqüências de treinamento é a inerente perda de eficiência. Os métodos de estimação conhecidos como não-supervisionados, autodidatas ou cegos, por sua vez, não utilizam seqüências de treinamento, mas apenas algumas características estatísticas do sinal transmitido. Assim, todo o sinal recebido corresponde à informação "útil". É exatamente por isso que os métodos cegos vêm sendo bastante estudados.

Nos receptores iterativos existe a possibilidade de incorporar o algoritmo de estimação entre o decodificador e o equalizador. Desta forma, a robustez introduzida pelo código corretor de erros pode ser usada para refinar as estimativas do canal, obtendo assim um estimador turbo, cujo desempenho supera o do receptor no qual a estimação de canal é feita separadamente da equalização e da decodificação. O foco principal desta dissertação é, portanto, o estudo de estimadores turbo cegos.

Um outro problema comum nos sistemas de comunicações móveis e de transmissão de dados pela rede elétrica é a presença de ruído impulsivo. Devido às carac-

terísticas particulares deste sinal, equalizadores não-lineares se mostram muito mais eficientes que os equalizadores lineares. Dentre os inúmeros filtros não-lineares, os sistemas *fuzzy* surgem como uma das principais alternativas graças à sua capacidade intrínseca de lidar com incertezas. Assim, neste trabalho investigaremos a utilização de um filtro *fuzzy* num esquema iterativo para mitigar os efeitos do ruído impulsivo de grande amplitude em canais PLC (*Power Line Communications*).

O objetivo principal deste trabalho é, portanto, estudar técnicas cegas e soluções de baixa complexidade para a equalização e estimação turbo, procurando avaliar as vantagens e limitações de cada alternativa analisada. Um outro objetivo do trabalho é realizar um estudo preliminar de técnicas iterativas para a redução dos efeitos do ruído impulsivo em sistemas de comunicações digitais.

1.1 Organização da Dissertação

O restante desta dissertação se organiza da seguinte maneira:

- **Capítulo 2** – *Fundamentos de Codificação, Equalização e Estimação de Canal*
  Apresenta o modelo do sistema que será utilizado no restante do trabalho e descreve alguns conceitos básicos sobre códigos corretores de erros e códigos turbo. Também é dada uma visão geral do problema de equalização e estimação de canal.

- **Capítulo 3** – *Equalização Turbo*
  Os princípios de funcionamento dos equalizadores turbo são apresentados em detalhes. Alguns equalizadores de complexidade exponencial e de complexidade reduzida são estudados, dando atenção especial aos de baixa complexidade. Através de simulações, faz-se um estudo comparativo entre os diversos equalizadores turbo estudados.

- **Capítulo 4** – *Estimação, Equalização e Decodificação Iterativas*
  Aborda o problema da equalização turbo quando o canal não é conhecido

pelo receptor. Apresenta um esquema de receptor cego iterativo que realiza conjuntamente as tarefas de equalização, decodificação e estimação de canal. Neste capítulo propõe-se a utilização de algoritmos de mínimos quadrados rápidos para estimar o canal e mostra-se, através de simulações, as vantagens em se empregar este esquema.

- **Capítulo 5** – *Equalização e Decodificação Iterativas Usando Filtros Fuzzy*
  Os conceitos básicos sobre os filtros *fuzzy* são abordados. Propõe-se, então, um equalizador *fuzzy* iterativo, cujo desempenho é superior ao de um equalizador *fuzzy* convencional. Para avaliar o desempenho do equalizador não-linear iterativo, considerou-se a equalização de canais PLC com forte presença de ruído impulsivo.

- **Capítulo 6** – *Conclusões e Perspectivas*
  Este capítulo conclui a dissertação apresentando as considerações finais e mostrando as perspectivas de trabalhos futuros.

2

Fundamentos de Codificação, Equalização e Estimação de Canal

2.1 Sistemas de Comunicação e Modelo em Banda Base

O objetivo principal de qualquer sistema de comunicação é transportar informação do emissor ao destinatário. Para cumprir tal tarefa, um sistema de comunicação deve possuir certos elementos que tornam a transmissão de informação possível. A Fig. 2.1 mostra o esquema de blocos de um sistema de comunicação genérico.

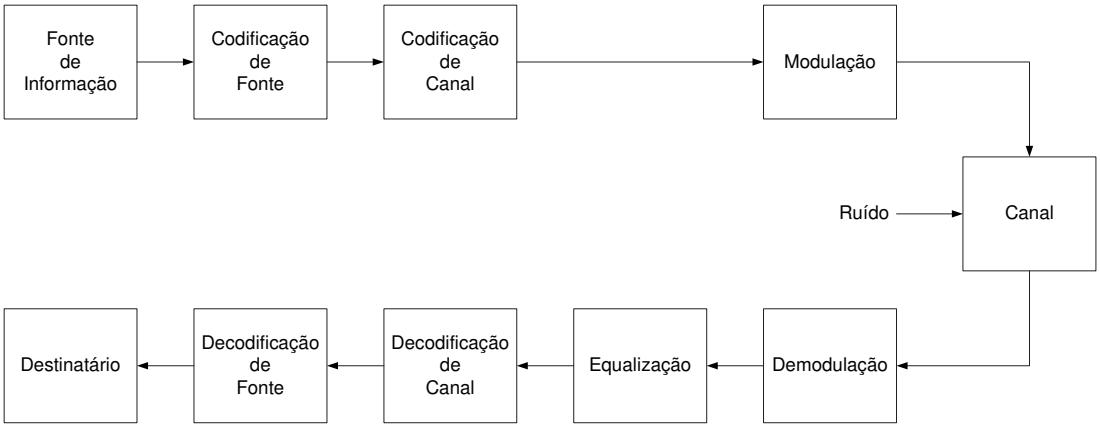


Figura 2.1: Esquema de um sistema de comunicação digital.

Nesta figura, a fonte de informação, ou emissor, pode ser uma pessoa ou uma máquina, como um computador pessoal, por exemplo. A saída da fonte de informação, que é a mensagem a ser enviada ao destinatário, pode ser tanto uma forma de onda contínua, como no caso da voz, quanto uma seqüência de símbolos, como em um texto. A função da codificação de fonte é transformar essa mensagem em uma seqüência de bits, chamados de bits de informação. Esta transformação é feita de maneira a minimizar o números de bits por unidade de tempo necessários para representar a mensagem. Já a codificação de canal insere uma redundância controlada na seqüência resultante da codificação de fonte de modo que o receptor possa detectar e/ou corrigir erros ocorridos durante a transmissão.

O modulador, conforme mostrado na Fig. 2.1, recebe a seqüência do codificador de canal e transforma esses dados em formas de onda adequadas para a transmissão pelo canal, que é o meio físico de propagação das informações. Canais de transmissão típicos incluem linhas telefônicas, fibras ópticas, o ar e o espaço. Porém, na sua viagem pelo meio de transmissão, a informação é corrompida por ruídos naturais e/ou produzidos pelo homem. Além disso, o canal pode distorcer as formas das ondas transmitidas.

Na recepção, o demodulador faz o papel inverso ao do modulador, gerando um sinal amostrado a partir das formas de onda distorcidas e corrompidas pelo canal.

O equalizador procura compensar as distorções impostas pelo canal. A seqüência equalizada, como mostra a Fig. 2.1, é então tratada pelo decodificador de canal, que emprega a redundância introduzida na codificação de canal para detectar e, se possível, corrigir eventuais erros. Por fim, a decodificação de fonte utiliza a saída do decodificador de canal para obter uma estimativa da mensagem original, que será entregue ao destinatário.

Nesta dissertação, iremos considerar o modelo em banda-base a tempo discreto do sistema de comunicação da Fig. 2.1. Assim, modulador, canal e demodulador são agrupados em um só bloco, formando um canal discreto equivalente. Esta combinação pode ser bem modelada por um filtro do tipo FIR (*Finite Impulse Response*), cuja saída é afetada por um ruído aditivo, branco e gaussiano de média nula. Consideraremos também que os bits na saída do codificador de canal são mapeados para os símbolos da constelação BPSK (*Binary Phase Shift Keying*). Além disso, consideraremos que a fonte de informação e a codificação de fonte foram combinadas num único bloco, bem como a decodificação de fonte e o destinatário. A Fig. 2.2 apresenta o diagrama de blocos do sistema de comunicação simplificado.

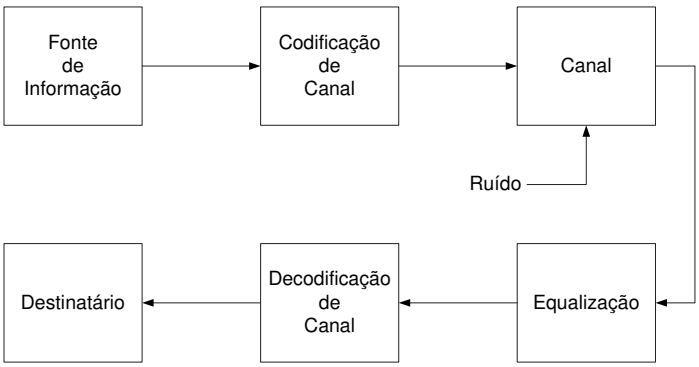


Figura 2.2: Sistema de comunicação digital simplificado.

Nas próximas seções iremos discutir brevemente algumas técnicas de estimação, equalização, codificação e decodificação de canal.

2.2 Codificação e Decodificação de Canal

As técnicas de codificação de canal são utilizadas para proteção de informações digitais transmitidas por um canal ruidoso e com interferências, permitindo reduzir a potência de transmissão requerida para se alcançar uma taxa de erro desejada. Isso é possível pois o código acrescenta bits de redundância à mensagem original.

Podemos dividir os códigos utilizados na codificação de canal em duas grandes classes: os códigos de bloco e os códigos convolucionais. Como o próprio nome sugere, os códigos de bloco codificam blocos de bits de informação. Já os códigos convolucionais podem codificar os bits de informação seqüencialmente ou em blocos.

A principal diferença entre estas duas categorias é a existência de memória no codificador. Nos códigos de bloco, cada operação de codificação depende somente das entradas atuais do codificador, sendo independente das entradas anteriores. Em contraste, num código convolucional cada seqüência de saída do codificador depende não apenas das entradas atuais mas também de um certo número de entradas anteriores, ou seja, o código apresenta memória. Neste trabalho, apresentaremos apenas alguns conceitos básicos relativos aos códigos convolucionais. Maiores informações sobre esta classe de códigos e sobre os códigos de bloco podem ser encontradas em [1, 2, 9].

Após a apresentação dos códigos convolucionais, esta seção discutirá os fundamentos dos códigos turbo, fonte de inspiração para as técnicas de equalização turbo.

2.2.1 Códigos Convolucionais

Os códigos convolucionais têm sido amplamente utilizados em uma enorme gama de aplicações, como as comunicações sem fios e as comunicações por satélite, por exemplo. Sua popularidade vem de sua estrutura simples e da possibilidade de implementação prática de métodos de decodificação por máxima verossimilhança [2, 9].

A partir de uma seqüência de bits de informação, o codificador de um código convolucional com k entradas e n saídas, $n > k$, produz uma seqüência de bits codificados conhecida como *palavra código*. A menor distância de Hamming entre

quaisquer duas palavras código de um código convolucional é conhecida como *distância livre*, ou d_{free} [2]. Quanto maior a distância livre, maior a capacidade do código de detectar/corrigir erros ocorridos durante a transmissão. A razão $R = k/n$ é chamada de taxa de codificação e indica a quantidade de redundância introduzida na mensagem.

Como já dito anteriormente, a saída depende não apenas da entrada num certo instante mas também das entradas em ν instantes anteriores. Logo, o codificador possui *memória* de ordem ν. Para uma taxa R fixa, códigos mais robustos podem ser construídos, até um certo limite, pelo aumento do número de elementos de memória. Um exemplo de codificador convolucional de taxa $R = 1/2$ pode ser visto na Fig. 2.3, onde os blocos marcados com a letra D são os elementos de memória e o operador $\oplus$ indica adição módulo-2. Na Fig. 2.3, o bit de saída c^1, num instante k, é gerado pela adição módulo-2 dos bits m_k e m_{k-2} enquanto que o bit c^2 é formado pela soma de m_k, m_{k-1} e m_{k-2}.

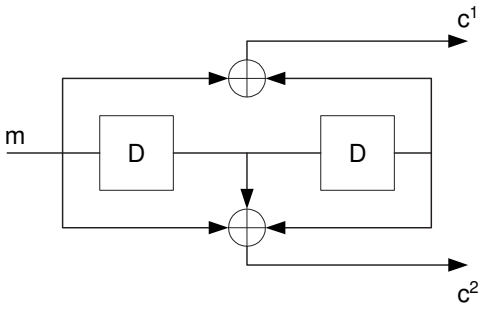


Figura 2.3: Exemplo de codificador.

Sendo a adição módulo-2 uma operação linear, o codificador é um sistema linear. Conseqüentemente, cada seqüência de saída ($\mathbf{c}^1$ e $\mathbf{c}^2$ na Fig. 2.3) é obtida através da convolução[1] da seqüência de entrada, $\mathbf{m}$, com as respectivas "respostas ao impulso". Estas respostas ao impulso são obtidas observando os bits codificados gerados pela entrada $\mathbf{m} = (1\ 0\ 0\ \cdots)$. Para o codificador do exemplo, as respostas ao impulso são: $\mathbf{g_1} = (1\ 0\ 1)$ e $\mathbf{g_2} = (1\ 1\ 1)$ ou $\mathbf{g_1} = 5$ e $\mathbf{g_2} = 7$, em octal.

[1] Daí a designação de códigos *convolucionais*.

Levando-se em conta a linearidade, a operação de convolução temporal pode ser substituída, como na transformada Z, por uma multiplicação em um domínio transformado. Então, representando a seqüência de entrada como um polinômio da forma $\mathbf{m}(D) = m_0 + m_1 D + m_2 D^2 + \cdots$ e a resposta ao impulso como $\mathbf{g}(D) = g_0 + g_1 D + \cdots + g_\nu D^\nu$, a seqüência de saída é dada por $\mathbf{c}(D) = \mathbf{m}(D)\,\mathbf{g}(D)$. Nestas expressões, D pode ser interpretado como um operador de atraso e as potências de D denotam o número de unidades de tempo que um bit é atrasado em relação ao bit inicial da seqüência. Assim, para o codificador da Fig. 2.3, pode-se escrever $\mathbf{g_1}(D) = 1 + D^2$ e $\mathbf{g_2}(D) = 1 + D + D^2$. Estes polinômios são conhecidos como polinômios geradores e fornecem informações sobre a estrutura do código.

Como os codificadores convolucionais são circuitos seqüenciais, seu funcionamento pode ser descrito por um diagrama de estados, sendo os estados definidos como o conteúdo dos elementos de memória. Cada um dos 2^ν estados apresenta 2^k ramos de chegada e 2^k ramos de saída, sendo k o número de entradas do codificador. O diagrama de estados do codificador representado na Fig. 2.3 pode ser visto na Fig. 2.4. Neste diagrama, S0, S1, S2 e S3 são os estados; as setas indicam transição entre dois estados enquanto que o primeiro dígito do conjunto x/yz define qual o bit de entrada (informação) e os dois dígitos seguintes, a saída do codificador.

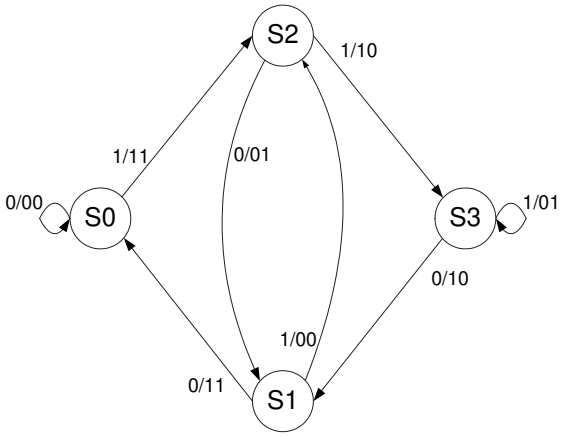


Figura 2.4: Diagrama de estados para o código (5,7).

Como pode ser observado, é possível determinar qual o estado atual do sistema conhecendo apenas o estado anterior e o bit de entrada que provocou a transição de estados. Logo, a codificação pode ser vista como um *processo de Markov* e sua evolução temporal representada através de um diagrama em treliça, como o da Fig. 2.5, que corresponde a um bloco de informação de 3 bits na entrada do codificador da Fig. 2.3 seguido por mais dois bits que forçam o codificador a voltar ao estado inicial $S0$.

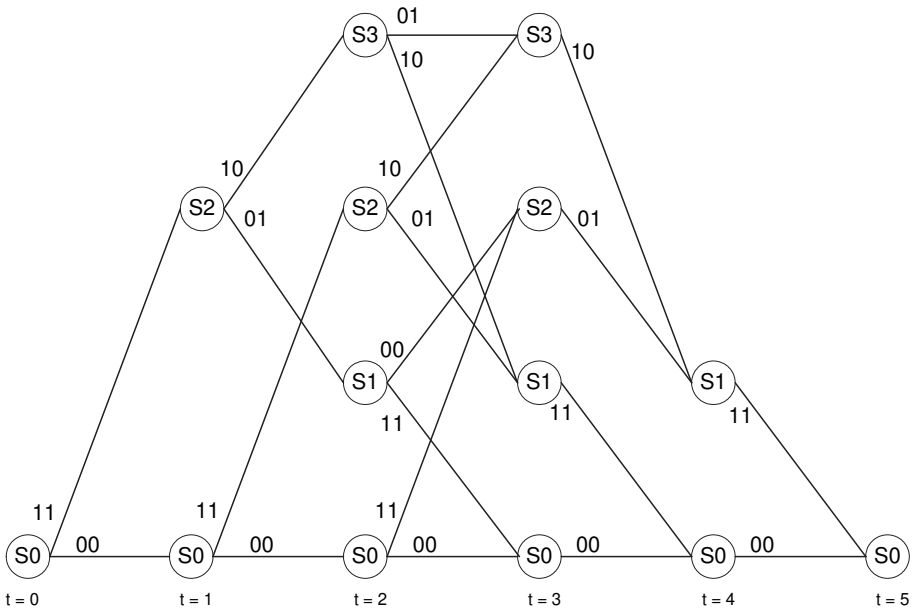


Figura 2.5: Diagrama em treliça para o código (5,7).

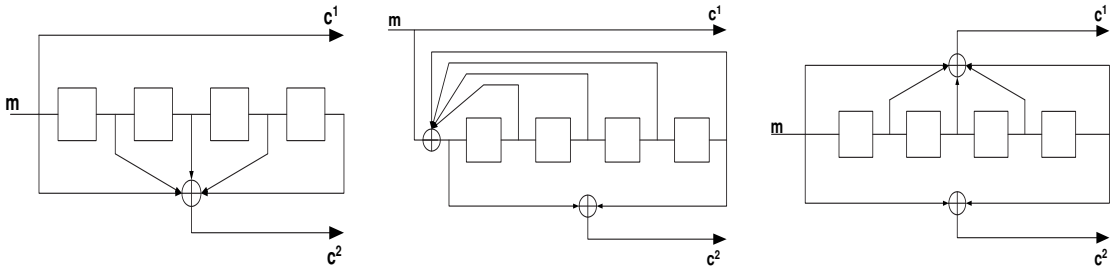
Numa treliça, os ramos de saída de um estado estão associados a cada uma das possíveis entradas do codificador. Assim, para o nosso exemplo, o ramo de saída inferior está associado ao bit de informação "0" enquanto que o ramo de saída superior está associado ao bit de informação "1". Os rótulos de cada ramo de saída mostram os bits codificados gerados pelos respectivos bits de informação. Além disso, cada palavra código é representada por um caminho único pela treliça, ou seja, cada palavra código é obtida através da concatenação dos bits de saída do codificador em cada transição de estado. Como será visto, a representação em

treliça de um código é de extrema importância no processo de decodificação.

Podemos ainda dividir os códigos convolucionais em não-sistemáticos e sistemáticos, recursivos e não-recursivos.

Num código sistemático, as primeiras k saídas são réplicas exatas das k entradas (bits de informação). As demais saídas do codificador correspondem aos *bits de paridade*. Em um código sistemático recursivo (RSC - *Recursive Systematic Convolutional*), os bits de paridade são gerados através de uma malha de realimentação. Como será visto na seção 2.2.2, esta é a classe de códigos empregada nos códigos turbo. Já nos códigos não-sistemáticos, nenhuma das saídas é exatamente igual a qualquer uma das entradas.

A Fig. 2.6(a) ilustra o codificador de um código convolucional sistemático de geradores $\mathbf{g}_1 = 20$ e $\mathbf{g}_2 = 37$. Já a Fig. 2.6(b) mostra o codificador do código convolucional sistemático recursivo com $\mathbf{g}_1 = 37$ e $\mathbf{g}_2 = 21$ enquanto que a Fig. 2.6(c) apresenta o de um código convolucional não-sistemático com $\mathbf{g}_1 = 37$ e $\mathbf{g}_2 = 21$. Vale notar que os códigos sistemático recursivo e o não-sistemático possuem os mesmos polinômios geradores, ou seja, podem ser descritos pelo mesmo diagrama em treliça, diferindo apenas nos bits de entrada e de saída associados a cada ramo dessa treliça.



(a) sistemático (20,37) (b) sistemático recursivo (37,21) (c) não-sistemático (37,21)

Figura 2.6: Exemplos de codificadores convolucionais.

Porém, o emprego de técnicas de codificação só pode ser útil se for possível recuperar a mensagem original a partir da informação codificada. Assim, deve existir uma relação biunívoca, dada pelos polinômios geradores, entre mensagem e palavra

código. A tarefa do decodificador é, portanto, produzir uma estimativa da mensagem original baseada no sinal recebido. Isto equivale a dizer que o decodificador precisa encontrar o caminho da treliça que gerou a palavra código enviada. Dessa forma, é necessário escolher um dentre os inúmeros caminhos possíveis.

Uma *regra de decodificação* é uma estratégia de escolha de uma palavra código para cada seqüência recebida. Um erro de decodificação ocorre quando a mensagem estimada difere da mensagem inicial.

Uma regra de decodificação freqüentemente utilizada na decodificação de códigos convolucionais é a que minimiza a probabilidade de erro de seqüência, isto é, $\min P(\hat{\mathbf{c}} \neq \mathbf{c})$. Pode-se mostrar [1, 2, 9] que para minimizar $P(\hat{\mathbf{c}} \neq \mathbf{c})$, as palavras código $\hat{\mathbf{c}}$ devem ser escolhidas de forma a maximizar a probabilidade *a posteriori* de $\mathbf{c}$, ou seja,

$$\hat{\mathbf{c}} = \arg\max_{\mathbf{c}} P(\mathbf{c}|\mathbf{r}), \tag{2.1}$$

onde $\mathbf{r}$ é a seqüência ruidosa recebida. Esse critério ótimo é conhecido como MAP (*Maximum a Posteriori Probability*).

Utilizando a regra de Bayes, a probabilidade *a posteriori* $P(\mathbf{c}|\mathbf{r})$ pode ser expressa como

$$P(\mathbf{c}|\mathbf{r}) = \frac{p(\mathbf{r}|\mathbf{c})\,P(\mathbf{c})}{P(\mathbf{r})}. \tag{2.2}$$

O termo $P(\mathbf{c})$ é a probabilidade *a priori* de ocorrência da palavra código $\mathbf{c}$. Já que $P(\mathbf{r})$ não depende de $\mathbf{c}$, a equação (2.1) é reescrita da seguinte maneira:

$$\hat{\mathbf{c}} = \arg\max_{\mathbf{c}} p(\mathbf{r}|\mathbf{c})\,P(\mathbf{c}). \tag{2.3}$$

A função $p(\mathbf{r}|\mathbf{c})$ é conhecida como função de verossimilhança. Comparando (2.3) e (2.1), vemos que um decodificador que maximize a verossimilhança é equivalente ao decodificador MAP quando as palavras código são consideradas equiprováveis, isto é, quando $P(\mathbf{c})$ é igual para todo $\mathbf{c}$. Em muitos sistemas, a probabilidade de ocorrência de cada palavra código não é conhecida pelo receptor, o que torna a decodificação MAP impossível. Nestes casos, a decodificação por máxima verossimilhança é a melhor das estratégias disponíveis. Na prática, a decodificação por

máxima verossimilhança é realizada pelo algoritmo de Viterbi [1, 2, 9], cuja implementação é mais simples que a do decodificador MAP.

Na maioria dos sistemas de comunicação digital usando codificação binária (somente 2 símbolos são produzidos: "0" e "1") o sinal recebido é quantizado antes de ser decodificado. Assim, a entrada do decodificador é composta por uma seqüência de "0's " e "1's " e o dispositivo realiza a chamada *decodificação abrupta* (*Hard Decoding*). Quando a entrada do decodificador possuir mais de dois níveis ou mesmo não for quantizada, tem-se a *decodificação suave* (*Soft Decoding*).

Embora os decodificadores que realizam decodificação abrupta sejam de implementação mais simples, seu desempenho não é tão bom quanto daqueles que empregam a decodificação suave uma vez que não utilizam toda a informação disponível no sinal recebido. Isto fica evidente a partir da observação da Fig 2.7, onde simulou-se a transmissão de blocos de 150 bits de informação por um canal AWGN (*Additive White Gaussian Noise*). Os bits de informação foram codificados pelo código convolucional sistemático recursivo de taxa $R = 1/2$ e geradores (7,5).

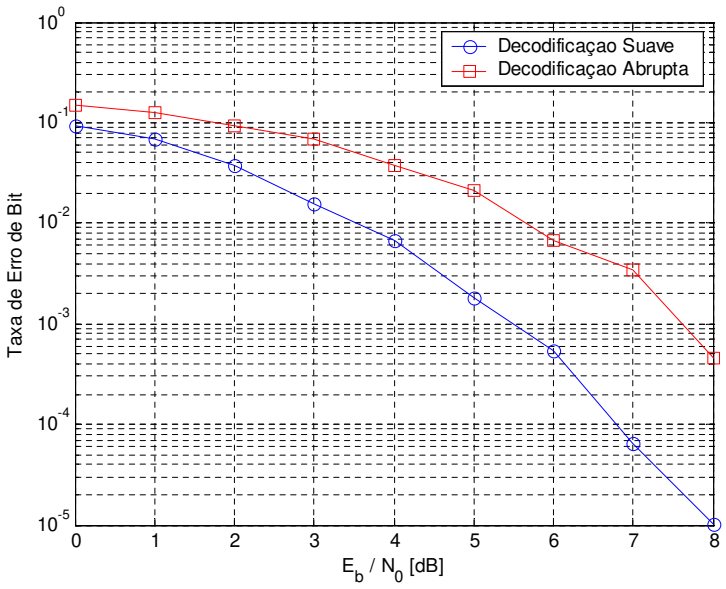


Figura 2.7: Comparação entre decodificação abrupta e suave para o código (7,5).

Os cálculos feitos em [1] mostram que a diferença de desempenho entre a decodificação abrupta e a decodificação suave é de cerca de 2 dB, confirmando os resultados da Fig. 2.7.

Na próxima seção, apresentaremos simplificadamente os códigos turbo, que são os precursores das técnicas iterativas investigadas neste trabalho.

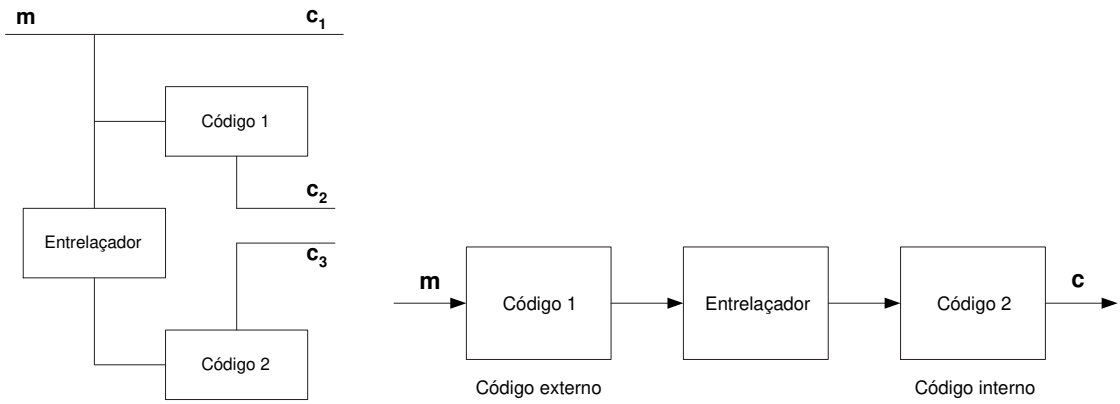
2.2.2 Códigos Turbo

Em 1993, uma nova classe de códigos convolucionais cujo desempenho se aproxima do limite de Shannon foi apresentada à comunidade científica [10]. Esses novos códigos, chamados de códigos turbo, representaram um dos maiores avanços na teoria de codificação de canal da última década, evoluindo muito rapidamente devido às suas potencialidades, combinando algumas idéias novas com conceitos e algoritmos "esquecidos". Hoje, já existem padrões definidos para a utilização dos códigos turbo em sistemas de telefonia celular de terceira geração [11].

Como será visto, não há nada de "turbo" na codificação, sendo o procedimento iterativo de decodificação que dá nome aos códigos. Embora esta dissertação não tenha como objetivo o estudo desses códigos, é importante conhecer seus princípios de funcionamento, principalmente os com concatenação serial, uma vez que eles serviram de inspiração para a equalização turbo, tendo estas duas técnicas muitas similaridades.

O primeiro codificador de um sistema turbo [10] era composto pela concatenação paralela de dois códigos convolucionais sistemáticos recursivos idênticos separados por um entrelaçador (*Interleaver*). No entanto, os códigos convolucionais sistemáticos recursivos também podem ser concatenados serialmente [12], conforme indicado na Fig. 2.8.

O entrelaçador, ou permutador, é de fundamental importância nos códigos turbo. Na codificação, sua função é reduzir o número de palavras com baixo peso de Hamming [13]. Além disso, a permutação dos bits permite "criar" códigos longos a partir de códigos convolucionais curtos, isto é, com poucos elementos de memória. Isto é



(a) Paralela (b) Serial

Figura 2.8: Exemplos de concatenação de códigos convolucionais.

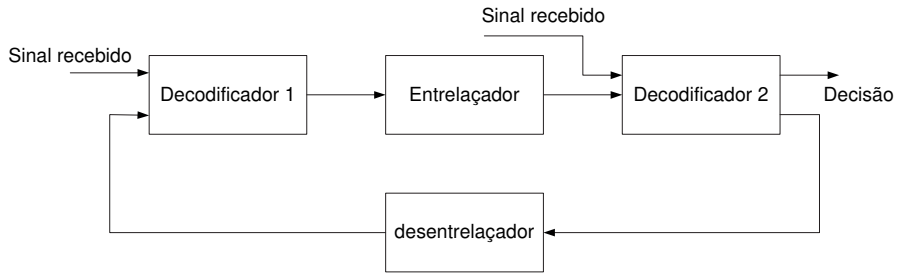
importante uma vez que somente códigos longos podem se aproximar do limite de Shannon [14].

A decodificação é realizada pela concatenação serial de dois ou mais decodificadores, cada um deles associado a um dos codificadores. Como a mensagem é codificada por mais de um codificador antes da transmissão, o receptor pode refinar suas decisões trocando informações, de maneira iterativa, entre os decodificadores. Isto quer dizer que uma mesma seqüência observada passa várias vezes pelo processo de decodificação antes de se produzir a decisão final sobre o bloco de dados. A Fig. 2.9 ilustra os esquemas de decodificação correspondentes aos dois tipos de concatenação da Fig. 2.8.

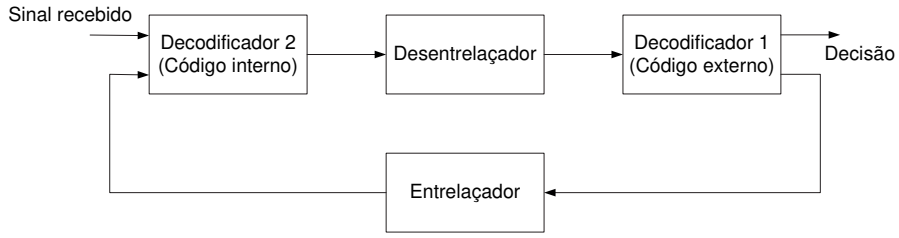
Como já foi mencionado na subseção 2.2.1, a decodificação suave tem um desempenho melhor que a decodificação abrupta. Logo, os dois decodificadores devem ser capazes de produzir decisões suaves para as etapas seguintes de decodificação. Essas decisões suaves são expressas por meio do logaritmo da razão de verossimilhança, ou LLR (*Log-Likelihood Ratio*), definido da seguinte maneira para uma modulação BPSK:

$$L\left(m_k\right) \triangleq \ln \frac{P\left(m_k = +1|\mathbf{r}\right)}{P\left(m_k = -1|\mathbf{r}\right)}, \tag{2.4}$$

onde m_k é o bit da mensagem no instante k e $\mathbf{r}$ é a seqüência recebida. Convém



(a) Paralela



(b) Serial

Figura 2.9: Exemplos de decodificação iterativa.

notar que sign $[L(m_k)]$ fornece a decisão MAP, $\hat{m}_k$, e o módulo de $L(m_k)$ indica a confiabilidade da decisão tomada estar correta [14, 15].

Como pode ser notado na Fig. 2.9(a), a saída de um decodificador é usada pelo outro como uma informação *a priori* sobre o sinal recebido. Esta informação, também chamada de informação *extrínseca*, corresponde a uma informação incremental sobre o bit m_k obtida através da decodificação de todos os outros bits do bloco de dados com exceção do bit sistemático recebido no instante k. Em outras palavras, o decodificador processa os bits recebidos ao redor do bit sistemático m_k e a informação *a priori* a respeito de m_k, juntamente com as restrições impostas pelo código, para gerar uma nova LLR sobre m_k.

Uma das chaves para o excelente desempenho da decodificação iterativa reside no fato de os decodificadores trocarem entre si apenas informações extrínsecas. Isto garante que cada decodificador utilize como informação *a priori* somente valores que não foram gerados diretamente por ele próprio na iteração anterior, evitando que a

malha de realimentação tenha *feedback* positiva, o que poderia gerar instabilidade do algoritmo.

Ainda é possível mostrar [10, 14–17], a partir da equação (2.4) e da regra de Bayes, que a LLR produzida na saída de cada decodificador pode ser escrita como a soma de três termos: um proveniente do canal (seqüência recebida), outro vindo do conhecimento *a priori* das probabilidades de ocorrência dos bits e o último correspondente à informação extrínseca gerada durante a decodificação. Logo, uma vez calculadas as LLR's, é possível obter as informações extrínsecas simplesmente subtraindo de $L(m)$ os termos correspondentes ao canal e à probabilidade *a priori*.

Algoritmos de decodificação que aceitam como entrada estimativas *a priori* da probabilidade de ocorrência dos bits da mensagem e produzem novas estimativas dessas probabilidades, condicionadas ao recebimento de toda a seqüência (probabilidades *a posteriori*), são chamados de algoritmos SISO (*Soft-Input, Soft-Output*). Exemplos desses algoritmos são o BCJR-MAP [5,9,10,14–17] e o SOVA (*Soft Output Viterbi Algorithm*) [9,14,18]. Uma diferença entre eles reside no fato que os estados estimados pelo algoritmo de Viterbi com saída suave devem formar um caminho factível através da treliça enquanto que os estados estimados pelo algoritmo MAP não estão, necessariamente, conectados.

A seguir, descreveremos apenas o algoritmo MAP por dois motivos: é ótimo no sentido de minimizar a probabilidade de erro de bit (o algoritmo de Viterbi minimiza a probabilidade de erro de palavra) e é o mais utilizado em esquemas de decodificação e equalização turbo.

O algoritmo MAP símbolo-a-símbolo, mais conhecido como BCJR, foi proposto por Bahl *et al* [5] em 1974. Ele calcula a probabilidade *a posteriori* de cada transição de estado bem como dos bits de mensagem de um processo de Markov dada uma seqüência ruidosa observada. Quando usado para a decodificação turbo, o algoritmo BCJR estima as probabilidades *a posteriori* dos bits de mensagem, isto é, $P(m_k = +1|\mathbf{r})$ e $P(m_k = -1|\mathbf{r})$ para a modulação BPSK, e calcula o logaritmo da razão de verossimilhança de acordo com a equação (2.4), que pode ser reescrita da

seguinte maneira:

$$L\left(m_k\right) = \ln \frac{P\left(m_k = +1|\mathbf{r}\right)}{P\left(m_k = -1|\mathbf{r}\right)} = \ln \frac{\sum\limits_{\{(s',s):m_k=+1\}} p\left(s', s, \mathbf{r}\right)}{\sum\limits_{\{(s',s):m_k=-1\}} p\left(s', s, \mathbf{r}\right)}, \tag{2.5}$$

onde s' e s representam os estados da treliça nos instantes $k-1$ e k, respectivamente. Os somatórios do numerador e do denominador são realizados para todas as transições existentes entre os estados s' e s quando o bit de informação é, respectivamente, $m_k = +1$ e $m_k = -1$.

Usando as propriedades de um processo de Markov, é possível mostrar [5] que as probabilidades conjuntas $p\left(s', s, \mathbf{r}\right)$ podem ser escritas como o produto de três termos independentes:

$$\begin{aligned} p\left(s', s, \mathbf{r}\right) &= p\left(s', \mathbf{r}_{j<k}\right) \cdot p\left(s, \mathbf{r}|s'\right) \cdot p\left(\mathbf{r}_{j>k}|s\right) \\ &= \underbrace{p\left(s', \mathbf{r}_{j<k}\right)}_{\alpha_{k-1}(s')} \cdot \underbrace{P\left(s|s'\right) \cdot p\left(r_k|s', s\right)}_{\gamma_k(s',s)} \cdot \underbrace{p\left(\mathbf{r}_{j>k}|s\right)}_{\beta_k(s)} \cdot \end{aligned} \tag{2.6}$$

Aqui, $\mathbf{r}_{j<k}$ representa a seqüência de símbolos recebidos do instante inicial até o instante $k-1$, enquanto que $\mathbf{r}_{j>k}$ é seqüência entre o instante $k+1$ e o final da treliça.

Os valores de $\alpha_k\left(s\right)$ e $\beta_k\left(s\right)$, podem ser calculados recursivamente da seguinte maneira:

$$\alpha_k\left(s\right) = \sum_{s'} \gamma_k\left(s', s\right) \cdot \alpha_{k-1}\left(s'\right), \tag{2.7a}$$

$$\beta_{k-1}\left(s'\right) = \sum_{s} \gamma_k\left(s', s\right) \cdot \beta_k\left(s\right). \tag{2.7b}$$

Considerando uma treliça de duração finita que inicia e termina no estado zero (s_0), $\alpha_k\left(s\right)$ e $\beta_k\left(s\right)$ são inicializados como segue:

$$\begin{aligned} \alpha_{inicio}\left(0\right) = 1 \qquad e \quad \alpha_{inicio}\left(s\right) = 0 \quad \forall s \neq 0 \\ \beta_{final}\left(0\right) = 1 \qquad e \quad \beta_{final}\left(s\right) = 0 \quad \forall s \neq 0 \end{aligned} \tag{2.8}$$

O algoritmo BCJR realiza o cálculo dos valores de α_k, indo do instante inicial ao final (recursão *forward*). Já os valores de β_k são obtidos deslocando-se do final para o início da treliça (recursão *backward*).

O termo $\gamma_k(s', s)$ nas equações (2.7) depende da saída atual do canal e, conforme pode ser observado na equação (2.6), é expresso em função de $P(s|s')$ e da probabilidade condicional $p(r_k|s', s)$ [16, 17]:

$$P(s|s') = P(m_k) = \begin{cases} \frac{e^{La_k}}{1+e^{La_k}} & para\ m_k = 1 \\ \frac{1}{1+e^{La_k}} & para\ m_k = -1 \end{cases} \tag{2.9}$$

e

$$p(r_k|s', s) = \prod_{l=1}^{n} \frac{1}{\sigma_n\sqrt{2\pi}} e^{\left(-\frac{1}{2\sigma_n^2}(r_{kl} - ac_{kl})^2\right)} \tag{2.10}$$

onde La_k é a informação *a priori* produzida pelo outro decodificador, $n = 1/R$ é o número de bits gerados na saída do codificador num instante k (considerando códigos turbo baseados em códigos convolucionais sistemáticos recursivos de taxa $1/n$), $c_{k,l},\ l = 1,\ \cdots\ n$, são as saídas do codificador na transição entre os estados s' e s no instante k, $r_{k,l}$ são os símbolos recebidos correspondentes a $c_{k,l}$, R é a taxa de codificação, σ_n^2 é a variância do ruído e a é a amplitude do desvanecimento do canal ($a = 1$ para um canal AWGN).

É importante salientar que na concatenação serial o decodificador externo não tem acesso às saídas do canal e, portanto, $p(r_k|s', s)$ se reduz a uma constante. Logo, $\gamma_k(s', s)$ passa a depender somente da informação *a priori*.

A saída do decodificador MAP pode, portanto, ser reescrita da seguinte forma:

$$L(m_k) = \ln \frac{\sum_{\{(s',s):m_k=+1\}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{\{(s',s):m_k=-1\}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}. \tag{2.11}$$

As referências [9, 10, 14–17] apresentam uma descrição detalhada do uso do algoritmo BCJR na decodificação dos códigos turbo.

2.3 Equalização

Como definido na seção 2.1, o canal de transmissão é o meio físico pelo qual são transportadas as informações. Na maioria dos sistemas de comunicação digital o sinal transmitido sofre um espalhamento temporal devido às características dispersivas do canal, fazendo com que dados transmitidos num determinado instante passem a interferir com dados transmitidos em outros instantes. Este fenômeno, chamado de interferência intersimbólica (IIS), é um dos principais fatores que limitam o desempenho dos sistemas de comunicações, provocando redução da confiabilidade e/ou da taxa de transmissão. Para minimizar os efeitos da IIS sobre o sinal recebido normalmente se emprega um filtro, denominado equalizador, no receptor.

Para que o equalizador possa remover toda a IIS do sinal recebido, sua função de transferência deve inverter a função de transferência do canal, isto é,

$$W(z) = \frac{1}{H(z)}, \tag{2.12}$$

onde $H(z)$ é a transformada Z do canal e $W(z)$ é a transformada Z do equalizador. Este critério de projeto do equalizador é chamado de *Zero Forcing* pois ele força a IIS a zero nos instantes de amostragem. O problema desse equalizador é que ele é altamente suscetível ao fenômeno conhecido como *noise enhancement*, isto é, uma amplificação demasiada do ruído quando o canal apresenta nulos espectrais.

Existem vários outros critérios de projeto de equalizadores nos quais se verifica uma redução do fenômeno de *noise enhancement*. As duas principais abordagens são o filtro de Wiener e o estimador de seqüência de máxima verossimilhança.

O filtro de Wiener é um método para obtenção dos coeficientes ótimos de um filtro linear discreto que minimiza o erro quadrático médio entre o sinal desejado e a saída do equalizador. Já o estimador de seqüência de máxima verossimilhança, ou MLSE (*Maximum Likelihood Sequence Estimator*), é uma estrutura não-linear que minimiza a probabilidade de erro de palavra. Nas duas próximas subseções, apresentaremos brevemente estas estruturas.

2.3.1 Filtro de Wiener

Para obter a solução de Wiener, vamos considerar o equalizador como sendo um filtro linear do tipo FIR (*Finite Impulse Response*) com coeficientes $\mathbf{w}$ e vetor de entrada $\mathbf{r}$ definidos da seguinte maneira:

$$\begin{aligned} \mathbf{w} &= \left[\begin{array}{cccc} w_0 & w_1 & \cdots & w_{M-1} \end{array} \right]^{\mathrm{T}} \\ \mathbf{r}\left(k\right) &= \left[\begin{array}{cccc} r\left(k\right) & r\left(k-1\right) & \cdots & r\left(k-M+1\right) \end{array} \right]^{\mathrm{T}} \end{aligned} \tag{2.13}$$

onde $M-1$ é a ordem do filtro, $r\left(k\right)$ é a k-ésima saída do canal e T indica transposição. Como o equalizador é um filtro linear, sua saída no instante k, $z\left(k\right)$, é calculada pela convolução do sinal de saída do canal pelos coeficientes do filtro, isto é,

$$z\left(k\right) = \sum_{i=0}^{M-1} w_i^* r\left(k-i\right) = \mathbf{r}^{\mathrm{H}}\left(k\right)\mathbf{w}\left(k\right) = \mathbf{w}^{\mathrm{H}}\left(k\right)\mathbf{r}\left(k\right), \tag{2.14}$$

onde H indica transposição hermitiana.

Se o sinal desejado no instante k é $d\left(k\right)$, então o sinal de erro na saída do equalizador é

$$e\left(k\right) = d\left(k\right) - z\left(k\right) = d\left(k\right) - \mathbf{w}^{\mathrm{H}}\left(k\right)\mathbf{r}\left(k\right). \tag{2.15}$$

A potência do sinal de erro pode ser obtida através do cálculo do valor quadrático médio de $e\left(k\right)$. Assim, podemos utilizar como critério de projeto do equalizador a minimização da potência do erro, ou seja,

$$\min_{\mathbf{w}} \mathrm{E}\left[e\left(k\right)e^*\left(k\right)\right] = \min_{\mathbf{w}} \mathrm{E}\left[\left|e\left(k\right)\right|^2\right]. \tag{2.16}$$

Este critério é mais conhecido como MMSE (*Minimum Mean-Squared Error*).

O vetor dos coeficientes ótimos, $\mathbf{w}_{otimo}$, é então expresso por [3, 19]

$$\mathbf{w}_{otimo} = \mathbf{R}^{-1}\mathbf{p}, \tag{2.17}$$

sendo $\mathbf{R} = \mathrm{E}\left[\mathbf{r}\left(k\right)\mathbf{r}^{\mathrm{H}}\left(k\right)\right]$ a matriz de autocorrelação do sinal recebido e $\mathbf{p} = \mathrm{E}\left[\mathbf{r}\left(k\right)d^*\left(k\right)\right]$, o vetor de correlação cruzada entre o sinal recebido e o sinal desejado. Como a função custo $J = \mathrm{E}\left[\left|e\left(k\right)\right|^2\right]$ é quadrática em $\mathbf{w}$, ela possui apenas um mínimo. Logo, o vetor dos coeficientes ótimos é único.

2.3.2 Estimador de Seqüência de Máxima Verossimilhança (MLSE)

Existem casos de canais em que o desempenho do equalizador linear é insatisfatório ou necessita de muitos coeficientes para conseguir um bom resultado. Nessas situações, técnicas não-lineares podem ter um desempenho muito melhor. Dentre as soluções mais conhecidas destaca-se o estimador de seqüência de máxima verossimilhança [19], ou MLSE, que é ótimo no sentido de minimização de probabilidade de erro de palavra.

Conforme mostrado na subseção 2.2.1, minimizar a probabilidade de erro de seqüência equivale a maximizar a função de verossimilhança para uma seqüência ruidosa observada quando os símbolos transmitidos são considerados equiprovavéis.

Modelando o canal de transmissão como um filtro linear discreto do tipo FIR, o processo de introdução de IIS se assemelha ao processo de codificação convolucional, onde a memória do canal, μ, é análoga à memória do codificador convolucional [20]. Conseqüentemente, o canal pode ser representado por um diagrama em treliça e a solução MLSE corresponde ao caminho desta treliça que esteja mais próximo da seqüência observada.

Para medir a proximidade entre o sinal recebido e os caminhos da treliça usa-se a distância euclidiana. A *métrica do ramo* é a distância entre os símbolos recebidos e os símbolos associados ao ramo da treliça em questão. A *métrica do caminho*, que é a soma das métricas dos ramos pertencentes àquele caminho, indica a distância total da seqüência recebida em relação à seqüência gerada por esse caminho. Para cada estado, o ramo com a menor métrica acumulada (menor distância) é chamado de *sobrevivente*, sendo os demais denominados *competidores*.

Portanto, podemos definir o critério MLSE como a minimização da métrica dos caminhos:

$$\hat{\mathbf{x}}(k) = \arg\min_{\mathbf{x}} \sum_{i=0}^{N} \left| r(i) - \sum_{j=0}^{\mu} h(j)\, x(i-j) \right|^2 \tag{2.18}$$

onde $\hat{\mathbf{x}} = \begin{bmatrix} \hat{x}_0 & \hat{x}_1 & \cdots & \hat{x}_N \end{bmatrix}$ é a estimativa da seqüência transmitida $\mathbf{x}$ e $h(k)$

são os coeficientes do canal. A maneira trivial de achar a solução MLSE é calcular a métrica de cada caminho possível na treliça e escolher aquele de menor valor. Contudo, o custo computacional disso pode ser extremamente elevado, o que ocorre para canais com muitos estados. Uma forma eficiente de se resolver este problema é através da utilização do algoritmo de Viterbi [21].

O algoritmo de Viterbi encontra o caminho com a menor métrica (mais verossímil) se movendo seqüencialmente pela treliça, comparando as métricas de todos os caminhos entrando em cada estado e armazenando apenas os caminhos sobreviventes, juntamente com suas métricas. Esse algoritmo pode, então, ser descrito da seguinte maneira, onde μ é a memória do canal e N é o tamanho da mensagem:

1. Começando no tempo $j = \mu$, calcule a métrica parcial para o único caminho entrando em cada estado. Armazene os sobreviventes e as respectivas métricas.

2. Incremente j de 1 unidade. Calcule a métrica parcial para todos os caminhos entrando em um estado somando a métrica de cada ramo com aquela do sobrevivente do instante anterior. Para cada estado, armazene o novo sobrevivente e sua métrica, eliminando todos os outros caminhos.

3. Se $j < N + \mu$, repita o passo 2. Caso contrário, pare.

As referências [1, 20] dão um descrição detalhada do equalizador MLSE com o algoritmo de Viterbi.

2.4 Estimação de Canal

Na seção anterior, apresentamos o equalizador MLSE e sua implementação através do algoritmo de Viterbi. Como pode ser visto em (2.18), o cálculo das métricas dos ramos depende dos coeficientes do canal, isto é, o equalizador ótimo deve conhecer precisamente o canal para poder recuperar as informações transmitidas.

A dependência em relação aos coeficientes do canal não é exclusiva do equalizador MLSE. Vários outros algoritmos, como veremos no capítulo 3, necessitam dessa informação.

O problema é que nem sempre o canal é conhecido *a priori*. Além disso, algumas características do canal podem variar com o tempo. Logo, é necessário um algoritmo de estimação de canal que seja capaz de seguir essas variações, ou seja, que deve ser capaz de se adaptar. Nesta seção, apresentaremos dois algoritmos clássicos da teoria de filtragem adaptativa para estimar os coeficientes do canal: o algoritmo LMS e o filtro de Kalman.

O algoritmo LMS é uma versão estocástica do método do gradiente enquanto que o filtro de Kalman é um filtro linear recursivo ótimo, no sentido MMSE, cuja formulação matemática está baseada em equações de estados. Neste trabalho, consideraremos apenas canais modelados como filtros lineares do tipo FIR. É importante mencionar que equalizadores lineares adaptativos também podem ter seus coeficientes atualizados pelos dois algoritmos em questão.

2.4.1 Algoritmo LMS

A solução de Wiener, calculada em (2.17), demanda a inversão da matriz de autocorrelação do sinal recebido, o que pode ter um custo computacional alto quando o filtro possui muitos coeficientes. Para solucionar este problema, podemos utilizar métodos iterativos, como o método do gradiente [3]. A idéia é usar o vetor gradiente do critério MSE (*Mean-Squared Error*) para obter, iterativamente, os coeficientes do filtro até a convergência para a solução de Wiener.

Dessa forma, se $\mathbf{h}(k) = \begin{bmatrix} h_0(k) & h_1(k) & \cdots & h_\mu(k) \end{bmatrix}^{\mathrm{T}}$ é o vetor com as estimativas do canal num instante k e $\mathbf{u}(k)$ é o vetor de símbolos transmitidos (seqüência de treinamento) que produz $r(k)$ na saída do canal, podemos definir o erro de estimação como

$$e(k) = r(k) - \mathbf{h}^{\mathrm{H}}(k)\,\mathbf{u}(k). \tag{2.19}$$

Nesta equação, supõe-se que a seqüência de treinamento é conhecida pelo receptor.

A fim de minimizar a potência do erro, o vetor $\mathbf{h}(k)$ é ajustado na direção oposta à do gradiente de $\mathrm{E}\left[|e(k)|^2\right]$, ou seja,

$$\mathbf{h}(k+1) = \mathbf{h}(k) - \frac{\mu_{adapt}}{2} \frac{\partial}{\partial \mathbf{h}(k)} \mathrm{E}\left[|e(k)|^2\right], \tag{2.20}$$

onde o passo de adaptação, μ_{adapt}, controla a velocidade de convergência do algoritmo e o erro residual após convergência.

O gradiente do erro quadrático médio é facilmente calculado como

$$\frac{\partial}{\partial \mathbf{h}(k)} \mathrm{E}\left[|e(k)|^2\right] = 2\left[\mathbf{R}(k)\,\mathbf{h}(k) - \mathbf{p}(k)\right], \tag{2.21}$$

sendo $\mathbf{R} = \mathrm{E}\left[\mathbf{u}(k)\,\mathbf{u}^{\mathrm{H}}(k)\right]$ e $\mathbf{p} = \mathrm{E}\left[\mathbf{u}(k)\,r^*(k)\right]$.

Na prática, o gradiente deve ser computado a partir dos dados recebidos uma vez que não se tem um conhecimento *a priori* da matriz de autocorrelação $\mathbf{R}$ e do vetor de correlação cruzada $\mathbf{p}$. A solução mais simples é substituir $\mathbf{R}$ e $\mathbf{p}$ pelas respectivas estimativas instantâneas:

$$\hat{\mathbf{R}}(k) = \mathbf{u}(k)\,\mathbf{u}^{\mathrm{H}}(k), \tag{2.22a}$$

$$\hat{\mathbf{p}}(k) = \mathbf{u}(k)\,r^*(k). \tag{2.22b}$$

Assim,

$$\mathbf{h}(k+1) = \mathbf{h}(k) - \frac{\mu_{adapt}}{2} \frac{\partial}{\partial \mathbf{h}(k)} |e(k)|^2 \tag{2.23}$$

e o novo vetor gradiente é calculado como

$$\frac{\partial}{\partial \mathbf{h}(k)} |e(k)|^2 = -2e^*(k)\,\mathbf{u}(k). \tag{2.24}$$

Tem-se, então

$$e(k) = r(k) - \mathbf{h}^{\mathrm{H}}(k)\,\mathbf{u}(k), \tag{2.25a}$$

$$\mathbf{h}(k+1) = \mathbf{h}(k) + \mu_{adapt} e^*(k)\,\mathbf{u}(k). \tag{2.25b}$$

O algoritmo representado por (2.25) é chamado de LMS (*Least Mean Square*) e corresponde a uma versão estocástica do método do gradiente. É possível mostrar [3] que o algoritmo LMS oscila em torno da solução de Wiener.

2.4.2 Filtro de Kalman

O filtro de Kalman [3, 19, 22] é um filtro linear ótimo, no sentido de minimizar o erro quadrático médio entre a saída do filtro e um sinal desejado. Sua formulação matemática é descrita em termos de equações de estado. Outra característica deste filtro é o cálculo recursivo da solução, o que o torna computacionalmente eficiente.

Para derivar o filtro de Kalman, consideremos o canal como um sistema dinâmico linear e discreto representado pelas seguintes equações de estado:

$$\mathbf{h}(k+1) = \mathbf{F}(k+1,k)\,\mathbf{h}(k) + \mathbf{n}_1(k), \tag{2.26a}$$

$$r(k) = \mathbf{h}^{\mathrm{H}}(k)\,\mathbf{u}(k) + n_2(k). \tag{2.26b}$$

A equação (2.26a) é chamada de equação de processo. O *vetor de estados*, denotado por $\mathbf{h}(k)$, é definido como qualquer vetor que descreva precisamente o comportamento do sistema dinâmico. No nosso caso, $\mathbf{h}(k)$ é o vetor com as estimativas dos coeficientes do canal. Já $\mathbf{F}(k+1,k)$ é a *matriz de transição de estados* que relaciona os estados do sistema entre os instantes $k+1$ e k e depende do modelo adotado para o canal. O vetor $\mathbf{n}_1(k)$ representa o *ruído de processo* e é modelado como sendo branco, de média nula e com matriz de correlação definida por:

$$\mathrm{E}\left[\mathbf{n}_1(k)\,\mathbf{n}_1^{\mathrm{H}}(j)\right] = \begin{cases} \mathbf{Q}_1(k) & , k = j \\ \mathbf{0} & , k \neq j \end{cases}. \tag{2.27}$$

A equação de observação, (2.26b), relaciona o estado atual e a saída do sistema. O vetor $\mathbf{u}(k)$ é conhecido como *vetor de observação* e corresponde à seqüência de treinamento enviada. Já $n_2(k)$ é chamado de *ruído de observação* e é modelado da mesma forma que $\mathbf{n}_1(k)$, sendo a correlação dada como:

$$\mathrm{E}\left[n_2(k)\,n_2^{\mathrm{H}}(j)\right] = \begin{cases} Q_2(k) & , k = j \\ 0 & , k \neq j \end{cases}. \tag{2.28}$$

Além disso, os processos $\mathbf{n}_1(k)$ e $n_2(k)$ são estatisticamente independentes.

A filtragem de Kalman consiste, portanto, em se obter estimativas do vetor de estados $\mathbf{h}$, num instante k, usando toda a seqüência de dados observados até aquele

instante. O algoritmo que realiza esta tarefa é obtido a partir das equações de estado (2.26) e de um desenvolvimento matemático elegante [3, 22], sendo expresso pelo seguinte conjunto de equações:

$$\begin{aligned}
&\mathbf{G}(k) = \mathbf{F}(k+1,k)\,\mathbf{K}(k,k-1)\,\mathbf{u}^{\mathrm{H}}(k)\left[\mathbf{u}(k)\,\mathbf{K}(k,k-1)\,\mathbf{u}^{\mathrm{H}}(k) + Q_2(k)\right]^{-1} \\
&\alpha(k) = r_k - \mathbf{h}^{\mathrm{H}}(k|\mathbf{r}_{k-1})\,\mathbf{u}(n) \\
&\mathbf{h}(k+1|\mathbf{r}_k) = \mathbf{F}(k+1,k)\,\mathbf{h}(k|\mathbf{r}_{k-1}) + \mathbf{G}(k)\,\alpha(k) \\
&\mathbf{K}(k) = \mathbf{K}(k,k-1) - \mathbf{F}(k,k+1)\,\mathbf{G}(k)\,\mathbf{u}(k)\,\mathbf{K}(k,k-1) \\
&\mathbf{K}(k+1,k) = \mathbf{F}(k+1,k)\,\mathbf{K}(k)\,\mathbf{F}^{\mathrm{H}}(k+1,k) + \mathbf{Q}_1(k)
\end{aligned} \tag{2.29}$$

Nestas equações, $\mathbf{G}(k)$ é o vetor do ganho de Kalman, $\mathbf{h}(k|\mathbf{r}_{k-1})$ é o vetor com a predição dos coeficientes do canal no instante k dados os sinais recebidos até o instante $k-1$, $\mathbf{K}(k,k-1)$ é a *matriz de correlação do erro de predição de estado*, que fornece uma descrição estatística do erro na predição de $\mathbf{h}(k|\mathbf{r}_{k-1})$, e $\mathbf{r}_k$ é o vetor com os sinais observados até o instante k.

2.5 Conclusão

Este capítulo apresentou, de forma bastante sucinta, alguns princípios básicos de codificação, decodificação, equalização e estimação de canais de comunicação. Esses conceitos serão utilizados no restante da dissertação.

3

Equalização Turbo

No capítulo anterior, dissemos que para aumentar a robustez de sistemas de transmissão ao ruído geralmente são empregados códigos corretores de erros, que introduzem uma redundância ao sinal transmitido. Entretanto, tradicionalmente esta redundância não é utilizada pelos equalizadores, sendo equalização e decodificação realizadas separadamente. Uma alternativa interessante para a melhoria significativa do desempenho do receptor surgiu com a equalização turbo. Nessa nova abordagem, equalização e decodificação são realizadas conjuntamente através de um procedimento iterativo por bloco, inspirado na decodificação dos códigos turbo.

Em geral, as taxas de erro de bit resultantes da equalização turbo são bem menores que aquelas obtidas pelas soluções tradicionais, em que equalização e decodificação são realizadas separadamente. Em outras palavras, a equalização turbo permite que se alcance uma certa taxa de erro de bit com uma potência de trans-

missão menor que a das técnicas usuais (não iterativas). Desde que foi apresentada pela primeira vez em 1995 [4], a equalização turbo vem sendo amplamente estudada, principalmente com o objetivo de redução da complexidade dos algoritmos envolvidos.

Conforme mencionado anteriormente, o canal discreto equivalente da Fig. 3.1 pode ser visto como um "código convolucional" não-sistemático, não-binário, de taxa R = 1 e possivelmente variante no tempo [4, 23–25].

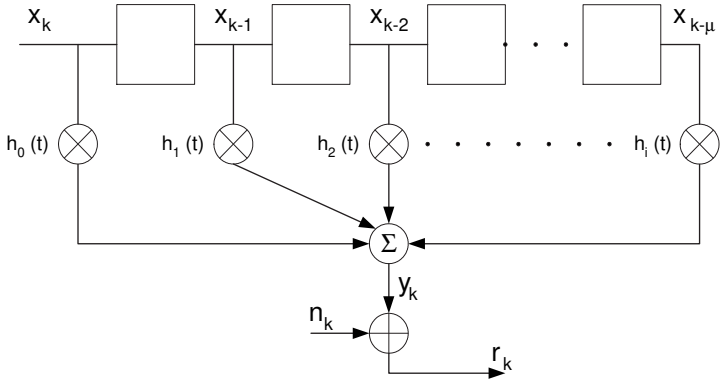


Figura 3.1: Modelo equivalente do canal.

Deste ponto de vista, codificação de canal e o próprio canal formariam um esquema de códigos serialmente concatenados, como pode ser visto na Fig. 3.2. A semelhança deste esquema com aquele da Fig.2.8(b) é evidente. Logo, podemos empregar uma estrutura iterativa ("turbo"), como a da Fig. 2.9(b), no receptor.

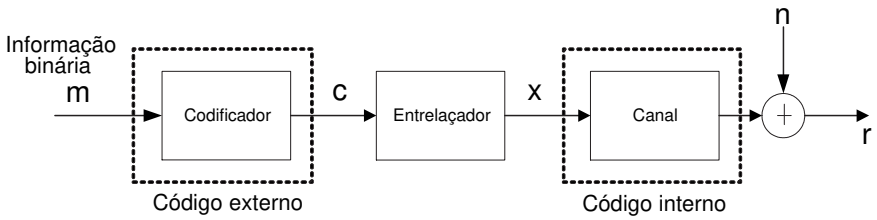


Figura 3.2: Concatenação do código corretor de erros com o canal.

Nesta abordagem, o equalizador tem o papel de "decodificar" o código interno, ou seja, o canal, enquanto que o decodificador externo faz a decodificação de canal. O esquema geral de um receptor iterativo pode ser visto na Fig. 3.3.

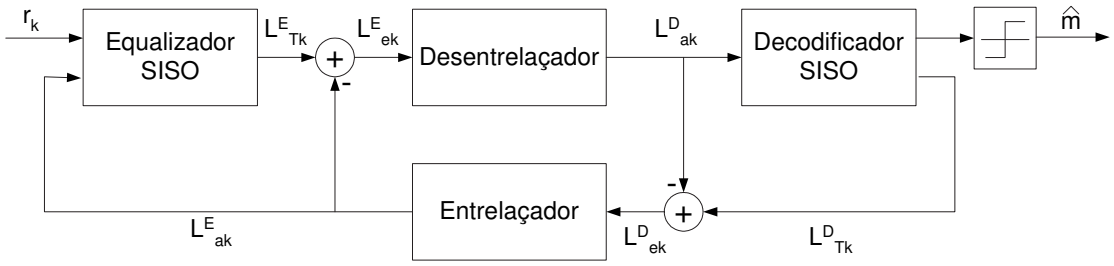


Figura 3.3: Equalizador turbo.

Assim como nos códigos turbo, os componentes do receptor iterativo devem ser capazes de trocar entre si decisões suaves de forma a utilizar toda a informação disponível no sinal. Como já explicitado, um dispositivo que aceita informações suaves na entrada, na forma de probabilidades *a priori* da ocorrência dos símbolos, e fornece decisões suaves é chamado de dispositivo SISO. Assim, na Fig. 3.3, L representa informações suaves na forma de LLR, os sobrescritos "E" e "D" se referem, respectivamente, ao equalizador e ao decodificador, o subscrito "T" representa as LLR´s (2.4), o subscrito "e" indica informação extrínseca, o subscrito "a", informação *a priori* e o subscrito "k" é o índice temporal. Os valores L^D_{Tk} correspondem as probabilidades *a posteriori* dos símbolos de saída do codificador enquanto que $\hat{m}$ é obtida a partir das probabilidades *a posteriori* dos símbolos de entrada do codificador.

Podemos descrever resumidamente o funcionamento do equalizador turbo da Fig. 3.3 da seguinte maneira: na primeira iteração, o equalizador não dispõe de nenhuma informação *a priori* sobre a ocorrência dos bits transmitidos, ou seja, os valores L^E_{ak} são nulos. Logo, o equalizador produz as LLR dos bits codificados utilizando apenas a seqüência ruidosa observada $\mathbf{r}$. Esses valores são desentrelaçados e passados ao decodificador, que gera novas estimativas das LLR's. Em seguida, as informações extrínsecas são calculadas, subtraindo das LLR's a informação *a priori*

L_{ak}^{D}, e entrelaçadas para o correto ordenamento dos bits, terminando a primeira iteração. Nas iterações seguintes, todo o ciclo se repete, com as estimativas da mensagem sendo refinadas a cada iteração. Esse processo permite a remoção de quase toda a IIS mesmo para canais com desvanecimentos profundos [4, 7, 8].

Aqui também é importante que apenas a informação extrínseca, isto é, incremental, seja passada entre os dispositivos SISO, evitando instabilidade no sistema. O entrelaçador tem a função de espalhar o erro por todo o bloco de dados [4, 7–9] além de promover o correto ordenamento dos bits.

Os algoritmos normalmente empregados tanto para a equalização quanto para a decodificação são o BCJR [5, 9, 23, 26] e suas versões simplificadas, conhecidas como Log-Map e Max-Log-MAP [9, 14, 16, 27]. O grande inconveniente dessas soluções é que sua complexidade computacional cresce exponencialmente com o comprimento do canal. O problema fica ainda maior uma vez que na equalização turbo, equalização e decodificação são realizadas várias vezes para um mesmo bloco de dados. Surge então a necessidade de se procurar soluções mais simples. Muitos esforços têm sido dedicados a esse respeito e esquemas utilizando equalizadores lineares [6–8, 25, 28] têm sido aplicados com sucesso.

Nas próximas seções, descreveremos os algoritmos com complexidade exponencial e alguns equalizadores de complexidade reduzida. Em seguida, apresentaremos comparações entre os diversos algoritmos estudados em várias condições de simulação de forma a estabelecer as vantagens e limitações de cada um deles.

3.1 Equalizadores com Complexidade Exponencial

3.1.1 Algoritmo MAP

O equalizador ótimo que minimiza a probabilidade de erro de bit é realizado pelo algoritmo BCJR, já descrito na subseção 2.2.2. Este equalizador calcula estimativas das probabilidades *a posteriori* (APP – *A Posteriori Probabilities*) de cada bit codificado, x_k, levando em consideração informações sobre todos os outros símbolos da

seqüência recebida $\mathbf{r}$, ou seja,

$$L(x_k) = \ln \frac{P(x_k = +1|\mathbf{r})}{P(x_k = -1|\mathbf{r})} = \ln \frac{\sum_{\{(s',s):x_k=+1\}} p(s', s, \mathbf{r})}{\sum_{\{(s',s):x_k=-1\}} p(s', s, \mathbf{r})}. \tag{3.1}$$

A principal diferença entre o algoritmo utilizado para a equalização e o usado para a decodificação está na forma de se calcular o termo $p(r_k|s', s)$ em (2.6). Para o equalizador, esta probabilidade pode ser escrita como

$$p(r_k|s', s) = \exp\left(-\frac{1}{2\sigma_n^2} \cdot \left| r_k - \sum_{i=0}^{\mu} h_i(t) \cdot x_{k-i} \right|^2\right). \tag{3.2}$$

Nesta equação, $h_i(t)$, $i = 0 \cdots \mu$, são os coeficientes do canal, σ_n^2 é a variância do ruído aditivo branco e gaussiano de média nula e x_{k-i} são os símbolos associados à transição entre os estados s e s'.

Outra diferença entre os algoritmos BCJR para equalização e para decodificação reside no cálculo das saídas. Enquanto o equalizador computa apenas as probabilidades *a posteriori* dos símbolos de entrada do canal, o decodificador calcula as probabilidades *a posteriori* dos símbolos de saída e de entrada do codificador. Estas últimas fornecem a estimativa MAP da mensagem transmitida. Além disso, cada ramo da treliça do canal está associado a somente um símbolo de saída enquanto que cada ramo da treliça do codificador está associado a múltiplas saídas. As referências [15, 23, 26] fornecem uma boa descrição do algoritmo BCJR aplicado a equalização.

3.1.2 Algoritmo Log-MAP

O algoritmo MAP da subseção anterior apresenta alta complexidade computacional uma vez que requer o cálculo de um grande número de exponenciais e multiplicações. O BCJR pode, ainda, apresentar instabilidade devido à representação numérica com precisão finita. Uma forma de diminuir a carga computacional e a sensibilidade numérica do algoritmo MAP consiste em realizar todas as operações

no domínio logarítmico. O algoritmo assim obtido é chamado de Log-MAP e pode ser descrito como [14, 16, 27, 29]:

$$\ln \alpha_k(s) = \ln \sum_{s'} \exp\{\ln \gamma_k(s', s) + \ln \alpha_{k-1}(s')\} \tag{3.3a}$$

$$\ln \beta_k(s') = \ln \sum_{s} \exp\{\ln \gamma_{k+1}(s', s) + \ln \beta_{k+1}(s)\} \tag{3.3b}$$

$$\begin{aligned} L(x_k) &= \ln \sum_{\{(s',s):x_k=+1\}} \exp\{\ln \alpha_{k-1}(s') + \ln \gamma_k(s', s) + \ln \beta_k(s)\} - \\ &- \ln \sum_{\{(s',s):x_k=-1\}} \exp\{\ln \alpha_{k-1}(s') + \ln \gamma_k(s', s) + \ln \beta_k(s)\}. \end{aligned} \tag{3.3c}$$

As expressões (3.3) podem ser calculadas mais facilmente através da utilização do *logaritmo Jacobiano*, definido como

$$\ln\left(e^{\delta_1} + e^{\delta_2}\right) = \max(\delta_1, \delta_2) + \ln\left(1 + e^{-|\delta_1 - \delta_2|}\right). \tag{3.4}$$

A partir de (3.4), a expressão $\ln\left(e^{\delta_1} + \ldots + e^{\delta_n}\right)$ pode ser calculada recursivamente [27] supondo que $\delta = \ln\left(e^{\delta_1} + \ldots + e^{\delta_{n-1}}\right)$ é conhecido. Assim,

$$\begin{aligned} \ln\left(e^{\delta_1} + \ldots + e^{\delta_n}\right) &= \ln\left(\Delta + e^{\delta_n}\right) \quad \text{com } \Delta = e^{\delta_1} + \ldots + e^{\delta_{n-1}} = e^{\delta} \\ &= \max(\ln \Delta, \delta_n) + f_c(|\ln \Delta - \delta_n|) \\ &= \max(\delta, \delta_n) + f_c(|\delta - \delta_n|). \end{aligned} \tag{3.5}$$

onde $f_c(|\delta - \delta_n|) = \ln\left(1 + e^{-|\delta - \delta_n|}\right)$. O comportamento do termo $\ln\left(1 + e^{-|\delta_1 - \delta_2|}\right)$ em relação ao módulo da diferença entre δ_1 e δ_2 pode ser observado na Fig. 3.4.

Fica evidente pela Fig. 3.4 que o termo em questão tende a zero à medida que a diferença entre δ_1 e δ_2 aumenta. Logo, ao invés de calcularmos essa função para cada valor de δ_1 e δ_2, podemos consultá-la em uma tabela unidimensional construída previamente. Em [27], verificou-se que apenas 8 valores de $|\delta_1 - \delta_2|$ na faixa de 0 a 5 são suficientes para um excelente desempenho do algoritmo. Também é possível mostrar [9, 27] que o algoritmo Log-MAP calculado através do logaritmo Jacobiano é equivalente ao BCJR.

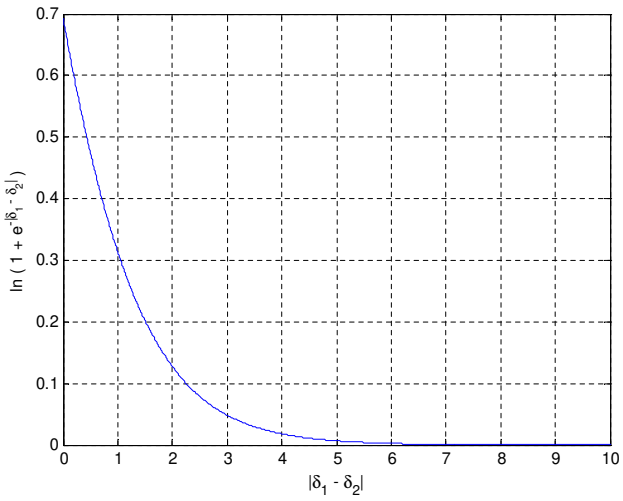


Figura 3.4: Comportamento de $\ln\left(1+e^{-|\delta_1-\delta_2|}\right)$.

3.1.3 Algoritmo Max-Log-MAP

Ainda considerando o comportamento de $\ln\left(1+e^{-|\delta_1-\delta_2|}\right)$, uma aproximação razoável é obtida simplesmente ignorando este termo. Então, o logaritmo Jacobiano é escrito como

$$\ln\left(e^{\delta_1}+e^{\delta_2}\right) \approx \max\left(\delta_1,\delta_2\right) \tag{3.6}$$

e o algoritmo assim obtido, descrito pelas equações (3.7), é conhecido como Max-Log-MAP [14, 16, 27, 29].

$$\ln\alpha_k\left(s\right) = \max_{s'}\left\{\ln\gamma_k\left(s',s\right)+\ln\alpha_{k-1}\left(s'\right)\right\} \tag{3.7a}$$

$$\ln\beta_k\left(s'\right) = \max_{s}\left\{\ln\gamma_{k+1}\left(s',s\right)+\ln\beta_{k+1}\left(s\right)\right\} \tag{3.7b}$$

$$\begin{aligned} L\left(x_k\right) = & \max_{\{(s',s):x_k=+1\}}\left\{\ln\alpha_{k-1}\left(s'\right)+\ln\gamma_k\left(s',s\right)+\ln\beta_k\left(s\right)\right\} - \\ - & \max_{\{(s',s):x_k=-1\}}\left\{\ln\alpha_{k-1}\left(s'\right)+\ln\gamma_k\left(s',s\right)+\ln\beta_k\left(s\right)\right\}. \end{aligned} \tag{3.7c}$$

Como conseqüência da aproximação feita em (3.6), o Max-Log-MAP considera a cada instante apenas dois caminhos da treliça do canal: o melhor para um bit transmitido $x_k=+1$ e o melhor para $x_k=-1$, sendo que um deles é o caminho de

máxima verossimilhança. Desta maneira, as decisões abruptas produzidas pelo Max-Log-MAP são idênticas às produzidas pelo algoritmo de Viterbi [9, 27]. Já o MAP e o Log-MAP levam em consideração todos os caminhos da treliça para calcular as LLR dos bits transmitidos.

3.2 Equalizadores com Complexidade Reduzida

Infelizmente, os três algoritmos apresentados anteriormente possuem uma complexidade computacional que cresce exponencialmente com o comprimento do canal. A idéia é, então, substituir esses algoritmos por equalizadores mais simples cuja complexidade não seja proibitiva. Dentre as várias soluções existentes podemos citar aquelas propostas em [6–8, 25], que descreveremos brevemente nas próximas subseções. Nesta dissertação, o equalizador proposto em [6], chamado de SFE, receberá uma atenção especial pois ele apresenta resultados melhores que os obtidos por outros equalizadores SISO de complexidade semelhante. Além disso, é razoável pensar que qualquer outro equalizador SISO de complexidade maior que a do SFE e menor que a do MAP provavelmente terá um desempenho intermediário entre o dessas duas estruturas.

3.2.1 Cancelamento de Interferência

Os equalizadores SISO lineares propostos em [7, 8, 25] se baseiam na estrutura mostrada na Fig. 3.5. Nesta estrutura, o sinal recebido num instante k, r_k, é filtrado

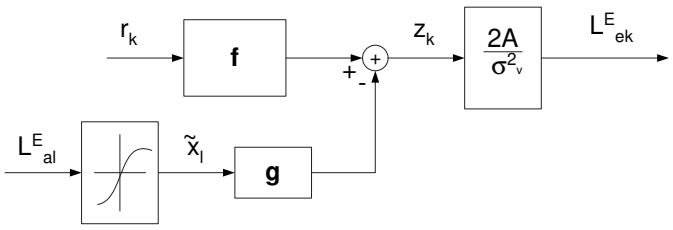


Figura 3.5: Cancelamento de interferência.

por um filtro linear $\mathbf{f}$, cuja saída contém IIS residual. Para cancelá-la, podemos utilizar a informação *a priori*, L_{ak}^{E}, para produzir estimativas suaves $\{\tilde{x}_{l\neq k}\}$ dos símbolos interferentes $\{x_{l\neq k}\}$, uma vez que

$$\tilde{x}_l = \mathrm{E}\left[x_l | L_{al}^{E}\right]. \tag{3.8}$$

Os valores de $\tilde{x}_l$ assim obtidos correspondem a estimativas de mínimo erro quadrático médio de x_l [28].

Lembrando que para uma modulação binária

$$L_{al}^{E}(x_l) = \ln \frac{P(x_l = +1)}{P(x_l = -1)} \quad \text{e} \quad P(x_l = +1) + P(x_l = -1) = 1,$$

obtemos:

$$\tilde{x}_l = \mathrm{E}\left[x_l | L_{al}^{E}\right] = \tanh\left(\frac{L_{al}^{E}}{2}\right). \tag{3.9}$$

O filtro $\mathbf{g}$ usa os valores de $\tilde{x}_l$ para estimar a IIS residual na saída de $\mathbf{f}$. Assim, a subtração indicada na Fig. 3.5 reduz o nível de interferência no sinal equalizado.

Uma vez que a saída do equalizador será usada para estimar o símbolo transmitido no instante k, a influência deste símbolo não deve ser cancelada. Isto é feito restringindo o coeficiente central de $\mathbf{g}$ a zero. Assim, a saída do equalizador no instante k não depende de $L_{ak}^{E}(x_k)$. Portanto, z_k pode ser usado para produzir informações extrínsecas. Para tanto, podemos escrever a saída do equalizador como

$$z_k = Ax_k + v_k, \tag{3.10}$$

onde $A = \mathrm{E}[z_k x_k] = \sum_l h_l f_{-l}$ é o ganho do canal equivalente e v_k é um ruído de variância σ_v^2, independente de x_k, e que inclui os efeitos do ruído do canal e da IIS residual. Supondo que v_k é uma variável aleatória gaussiana, podemos calcular a informação extrínseca L_{ek}^{E} da seguinte maneira [28]:

$$L_{ek}^{E} = \frac{2Az_k}{\sigma_v^2}. \tag{3.11}$$

O equalizador proposto em [7] considera que as decisões suaves de (3.9) estão corretas, isto é, $\tilde{x}_k = x_k$. Portanto, $\mathbf{f}$ é escolhido como o filtro casado ao canal, ou

seja, $f_k = h_{-k}$. Já os equalizadores apresentados em [8, 25] são escolhidos segundo o critério MMSE e dependem da informação *a priori*. Conseqüentemente, seus coeficientes devem ser computados para cada símbolo transmitido, resultando num equalizador variante no tempo e cuja complexidade computacional é quadrática em relação ao comprimento de $\mathbf{f}$.

3.2.2 Soft-Feedback Equalizer

O SFE (*Soft-Feedback Equalizer*), proposto em [6,28] e apresentado na Fig. 3.6, é uma estrutura SISO de cancelamento de interferência que possui algumas características similares às de um DFE (*Decision-Feedback Equalizer*) [1, 3]. Diferentemente dos equalizadores descritos na seção 3.2.1, que utilizam apenas a informação *a priori* fornecida pelo decodificador para cancelar toda a interferência sobre o símbolo z_k, o SFE combina as informações extrínsecas produzidas por ele próprio com as informações *a priori* fornecidas pelo decodificador para obter estimativas $\bar{x}_l = \mathrm{E}\left[x_l | L^E_{Tk}\right]$ mais confiáveis dos símbolos interferentes x_l, $l < k$.

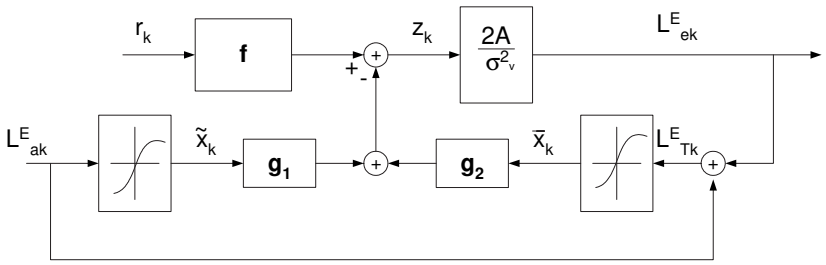


Figura 3.6: *Soft-feedback equalizer.*

Para calcular os coeficientes do SFE, vamos primeiramente escrever a saída z_k como

$$z_k = \mathbf{f}^{\mathrm{T}}\mathbf{r}_k - \mathbf{g}_1^{\mathrm{T}}\tilde{\mathbf{x}}_k - \mathbf{g}_2^{\mathrm{T}}\bar{\mathbf{x}}_k, \tag{3.12}$$

onde $\mathbf{f} = [f_{-M_1}, \ldots f_{M_2}]^{\mathrm{T}}$, $\mathbf{r}_k = [r_{k+M_1}, \ldots r_{k-M_2}]^{\mathrm{T}}$, $\mathbf{g}_1 = [g_{-M_1}, \ldots g_{-1}]^{\mathrm{T}}$ é estritamente anticausal, $\mathbf{g}_2 = [g_1, \ldots g_{M_2+\mu}]^{\mathrm{T}}$ é estritamente causal, $\tilde{\mathbf{x}}_k = [\tilde{x}_{k+M_1}, \ldots \tilde{x}_{k+1}]^{\mathrm{T}}$,

$\bar{\mathbf{x}}_k = [\bar{x}_{k-1}, \ldots \bar{x}_{k-M_2-\mu}]^{\mathrm{T}}$, μ é o comprimento do canal, M_1 e M_2 determinam os comprimentos dos filtros e o sobrescrito T denota transposição. Das definições de $\tilde{\mathbf{x}}_k$ e $\bar{\mathbf{x}}_k$ nota-se que, num instante k, o SFE tenta cancelar a interferência causada por M_1 símbolos futuros e $M_2 + \mu$ símbolos passados. Em outras palavras, a saída do filtro $\mathbf{f}$ apresenta IIS residual de M_1 símbolos futuros e $M_2 + \mu$ símbolos anteriores ao instante k.

Uma vez que $\tilde{\mathbf{x}}_k$ e $\bar{\mathbf{x}}_k$ são aproximadamente iguais a $\mathbf{x}_k$ e os símbolos transmitidos não são correlacionados, podemos escrever $\mathrm{E}[\tilde{x}_k x_j] = \mathrm{E}[\bar{x}_k x_j] = \mathrm{E}[\tilde{x}_k \bar{x}_j] = 0$, $j \neq k$. Portanto, conforme mostrado em [28], os filtros lineares $\mathbf{f}$, $\mathbf{g}_1$ e $\mathbf{g}_2$ que minimizam o erro quadrático médio $\mathrm{E}\left[|z_k - x_k|^2\right]$ entre a saída do equalizador e o símbolo transmitido são dados por:

$$\mathbf{f} = \left(\mathbf{H}\mathbf{H}^{\mathrm{T}} - \frac{\alpha_1^2}{E_1}\mathbf{H}_1\mathbf{H}_1^{\mathrm{T}} - \frac{\alpha_2^2}{E_2}\mathbf{H}_2\mathbf{H}_2^{\mathrm{T}} + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{h}_0, \tag{3.13}$$

$$\mathbf{g}_1 = (\alpha_1/E_1)\,\mathbf{H}_1^{\mathrm{T}}\mathbf{f}, \tag{3.14}$$

$$\mathbf{g}_2 = (\alpha_2/E_2)\,\mathbf{H}_2^{\mathrm{T}}\mathbf{f}. \tag{3.15}$$

onde $\mathbf{H}$ é a matriz de convolução do canal,

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \ldots & h_\mu & 0 & 0 & \ldots & 0 \\ 0 & h_0 & h_1 & \ldots & h_\mu & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & \ldots & \ldots & h_0 & h_1 & \ldots & h_\mu \end{bmatrix}, \tag{3.16}$$

de dimensões $M \times (M + \mu)$, $M = M_1 + M_2 + 1$ e

$$E_1 = \mathrm{E}\left[|\tilde{x}_k|^2\right], \tag{3.17}$$

$$E_2 = \mathrm{E}\left[|\bar{x}_k|^2\right], \tag{3.18}$$

$$\alpha_1 = \mathrm{E}\left[\tilde{x}_k x_k\right], \tag{3.19}$$

$$\alpha_2 = \mathrm{E}\left[\bar{x}_k x_k\right]. \tag{3.20}$$

Numerando as colunas da matriz de convolução do canal como $\mathbf{H} = [\mathbf{h}_{-M_1}, \ldots \mathbf{h}_{M_2+\mu}]$, o vetor $\mathbf{h}_0$ é a coluna de índice zero, $\mathbf{H}_1 = [\mathbf{h}_{-M_1}, \ldots \mathbf{h}_{-1}]$ e $\mathbf{H}_2 = [\mathbf{h}_1, \ldots \mathbf{h}_{M_2+\mu}]$.

A informação extrínseca L_{ek}^E na saída do SFE pode ser computada através de (3.11) para uma modulação BPSK, substituindo A por $\mathbf{f}^{\mathrm{T}}\mathbf{h}_0$ e σ_v^2 por $A(1-A)$ [6, 28]. Assim:

$$L_{ek}^E = \frac{2z_k}{1 - \mathbf{f}^{\mathrm{T}}\mathbf{h}_0}. \tag{3.21}$$

Para calcular os valores de E_1 e α_1, vamos supor que L_{ak}^E é produzida por um canal AWGN equivalente, ou seja, $L_{ak}^E = \gamma_p (x_k + w_k)$, onde w_k é um ruído aditivo, branco e gaussiano de variância σ_w^2, independente da seqüência transmitida, do ruído do canal de transmissão e da saída do equalizador. O valor $\gamma_p = 2/\sigma_w^2$ é proporcional à relação sinal-ruído do canal equivalente que gerou L_{ak}^E e está diretamente ligado à qualidade da informação *a priori*. Condicionando x_k a 1 [6, 28], obtém-se $E_1 = \mathrm{E}\left[|\tilde{x}_k|^2 | x_k = 1\right]$, $\alpha_1 = \mathrm{E}\left[\tilde{x}_k | x_k = 1\right]$ e L_{ak}^E é uma variável aleatória gaussiana de média γ_p e variância $2\gamma_p$, isto é, $L_{ak}^E \sim \mathcal{N}(\gamma_p, 2\gamma_p)$. Portanto,

$$\alpha_1 = \Psi_1(\gamma_p), \tag{3.22}$$

$$E_1 = \Psi_2(\gamma_p), \tag{3.23}$$

onde

$$\Psi_1(\gamma) = \mathrm{E}\left[\tanh(u/2)\right], u \sim \mathcal{N}(\gamma, 2\gamma), \tag{3.24}$$

$$\Psi_2(\gamma) = \mathrm{E}\left[\tanh^2(u/2)\right], u \sim \mathcal{N}(\gamma, 2\gamma). \tag{3.25}$$

As expressões em (3.24) e (3.25) não apresentam uma solução fechada mas podem ser calculadas por algoritmos numéricos simples.

Da Fig. 3.6, nota-se que $L_{Tk}^E = L_{ek}^E + L_{ak}^E$. Considerando a aproximação gaussiana para a informação extrínseca L_{ek}^E feita em (3.10) e (3.11), chega-se a

$$L_{Tk}^E = (\gamma_p + \gamma_e) x_k + \gamma_p w_k + \gamma_e v_k, \tag{3.26}$$

com $\gamma_e = 2A^2/\sigma_v^2$ proporcional à relação sinal-ruído do canal equivalente que produziu L_{ek}^E. Supondo que $x_k = 1$, tem-se $L_{Tk}^E \sim \mathcal{N}(\gamma_p + \gamma_e, 2(\gamma_p + \gamma_e))$ e

$$\alpha_2 = \Psi_1(\gamma_p + \gamma_e), \tag{3.27}$$

$$E_2 = \Psi_2(\gamma_p + \gamma_e). \tag{3.28}$$

Porém, para calcular E_1, E_2, α_1 e α_2, os valores de γ_p e γ_e precisam ser estimados. Quando o SFE é empregado num equalizador turbo, podemos utilizar o estimador proposto em [30] para computar γ_p e γ_e em cada iteração turbo a partir dos respectivos valores na iteração anterior. Dessa forma, se $z_k = Ax_k + v_k$ como em (3.10), γ_e é computado iterativamente a partir das estimativas iniciais $\hat{A}_0$ e $\hat{\sigma}_0^2$ da seguinte maneira:

$$\begin{aligned}
\hat{A}_i &= \frac{1}{L}\sum_{k=0}^{L-1} \tanh\left(\frac{\hat{A}_{i-1} z_k}{\hat{\sigma}_{i-1}^2}\right) z_k, \\
\hat{\sigma}_i^2 &= \frac{1}{L}\sum_{k=0}^{L-1} \left\| \hat{A}_i \operatorname{sign}(z_k) - z_k \right\|^2, \\
\hat{\gamma}_e^{(i)} &= \frac{2\hat{A}_i^2}{\hat{\sigma}_i^2},
\end{aligned} \tag{3.29}$$

onde o índice $i > 0$ se refere à iteração turbo e L é o tamanho do bloco de dados recebidos. Uma estimativa para γ_p é conseguida substituindo z_k por L_{ak}^E nas equações acima.

Os valores iniciais $\hat{A}_0$ e $\hat{\sigma}_0^2$ necessários ao cálculo de $\hat{\gamma}_e^{(1)}$ são obtidos a partir do procedimento iterativo [6] descrito a seguir:

$$\begin{aligned}
\mathbf{f}_j &= \left(\mathbf{H}\mathbf{H}^{\mathrm{T}} - \frac{\Psi_1^2\left(\gamma_e^{(j)}\right)}{\Psi_2\left(\gamma_e^{(j)}\right)} \mathbf{H}_2\mathbf{H}_2^{\mathrm{T}} + \sigma^2\mathbf{I} \right)^{-1} \mathbf{h}_0, \\
\gamma_e^{(j)} &= 2\mathbf{f}_j^{\mathrm{T}}\mathbf{h}_0 / \left(1 - \mathbf{f}_j^{\mathrm{T}}\mathbf{h}_0\right).
\end{aligned} \tag{3.30}$$

Nestas equações, considerou-se $E_1 = 1$ e $\alpha_1 = 0$ uma vez que na primeira iteração turbo não há informação *a priori* disponível e, portanto, $\gamma_p = 0$. É possível

mostrar [28] que este procedimento iterativo converge muito rapidamente, freqüentemente após três iterações, quando se inicializa γ_e com zero. Já para calcular $\hat{\gamma}_p^{(1)}$ em (3.29), considera-se $\hat{\sigma}_0^2 = 2\hat{A}_0$, que reflete a aproximação gaussiana feita anteriormente para L_{ak}^E.

É importante notar que as aproximações gaussianas para L_{ak}^E e L_{ek}^E fazem com que os coeficientes do SFE sejam invariantes no tempo, o que leva a uma complexidade computacional por símbolo proporcional ao número de coeficientes do filtro $\mathbf{f}$. As soluções em [8], como já mencionado na seção 3.2.1, possuem coeficientes variantes no tempo e uma complexidade por símbolo que depende do quadrado do números de coeficientes do equalizador.

Da Fig. 3.6, podemos ver que os filtros $\mathbf{g}_1$ e $\mathbf{g}_2$ são alimentados, respectivamente, com estimativas $\tilde{x}_k$ e $\bar{x}_k$ dos símbolos interferentes. Como L_{ak}^E e L_{Tk}^E são dadas na forma de LLR, um mapeamento como aquele de (3.9) é necessário. Logo,

$$\begin{aligned} \tilde{x}_k &= \mathrm{E}\left[x|L_{ak}^E\right] = \tanh\left(L_{ak}^E/2\right), \\ \bar{x}_k &= \mathrm{E}\left[x|L_{Tk}^E\right] = \tanh\left(L_{Tk}^E/2\right). \end{aligned} \tag{3.31}$$

Uma análise cuidadosa de (3.13) – (3.15) revela que, para certos valores de γ_p e γ_e, o SFE se reduz a outros equalizadores bastante conhecidos. Assim, quando γ_p e γ_e são muito pequenos, $\alpha_1^2/E_1 \to 0$ e $\alpha_2^2/E_2 \to 0$ e nenhum cancelamento de interferência é feito. Neste caso, o SFE se reduz a um equalizador MMSE linear $\mathbf{f}$.

Já quando $\gamma_p \to 0$ e $\gamma_e \to \infty$, O SFE se reduz a um DFE convencional pois γ_p pequeno implica em informação *a priori* não confiável e, portanto, nenhum cancelamento da IIS precursora. Contudo, γ_e grande indica que a saída do equalizador é confiável e pode ser usada para cancelar efetivamente a IIS pós-cursora.

Por fim, quando $\gamma_p \to \infty$, o SFE se reduz ao equalizador da Fig. 3.5. Isto ocorre porque, quando γ_p tem um valor alto, o equalizador tem acesso a estimativas confiáveis de todos os símbolos interferentes e, portanto, pode cancelar perfeitamente sua interferência.

Se analisarmos o comportamento das funções $\Psi_1(\gamma)$ e $\Psi_2(\gamma)$, poderemos simplificar ainda mais o cálculo dos coeficientes do SFE. Conforme mostrado em [28],

as razões α_1/E_1 e α_2/E_2 podem ser bem aproximadas pelo valor constante 1 sem prejudicar a saída do equalizador. Conseqüentemente, os filtros aproximados são calculados como

$$\begin{aligned} \mathbf{f} &= \left(\mathbf{H}\mathbf{H}^{\mathrm{T}} - \alpha_1 \mathbf{H}_1 \mathbf{H}_1^{\mathrm{T}} - \alpha_2 \mathbf{H}_2 \mathbf{H}_2^{\mathrm{T}} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{h}_0, \\ \mathbf{g}_1 &= \mathbf{H}_1^{\mathrm{T}} \mathbf{f}, \\ \mathbf{g}_2 &= \mathbf{H}_2^{\mathrm{T}} \mathbf{f}. \end{aligned} \tag{3.32}$$

Nesta dissertação, todas as simulações feitas com o SFE utilizaram o conjunto de equações simplificadas (3.32).

Os trabalhos originais [6, 28] computam a função $\Psi_1(\gamma)$ através de algoritmos numéricos. Neste trabalho, propomos a utilização da seguinte aproximação empírica, baseada em [31], para $\Psi_1(\gamma)$:

$$\Psi_1(\gamma) = \begin{cases} 0,4808\gamma + 1.10^{-4} & ,\gamma < 0,2 \\ 1 - \exp\left(B - A\gamma^G\right) & ,\gamma \geq 0,2 \end{cases}, \tag{3.33}$$

com $A = 0,4527$, $B = 0,0218$ e $G = 0,86$. O uso desta aproximação permite uma redução do custo computacional além de simplificar o cálculo dos coeficientes do SFE.

Verificamos que a utilização desta aproximação no lugar do cálculo exigido por (3.24) não alterou o desempenho do SFE. Isto fica claro na Fig. 3.7, que mostra a função $\Psi_1(\gamma)$ calculada por (3.24) e por (3.33). Notamos que (3.33) só deixa de ser uma boa aproximação para $\Psi_1(\gamma)$ quando γ é inferior 10^{-3}. Porém, valores tão pequenos de γ são uma indicação de que as informações *a priori* não são confiáveis e portanto, como já discutido acima, pouco contribuem para o cálculo da saída do SFE.

3.3 Simulações

Nesta seção, realizamos um estudo comparativo do desempenho dos equalizadores turbo apresentados anteriormente em vários cenários de simulação. Procuramos

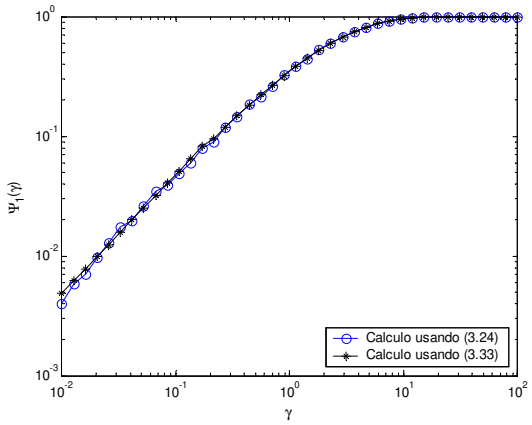
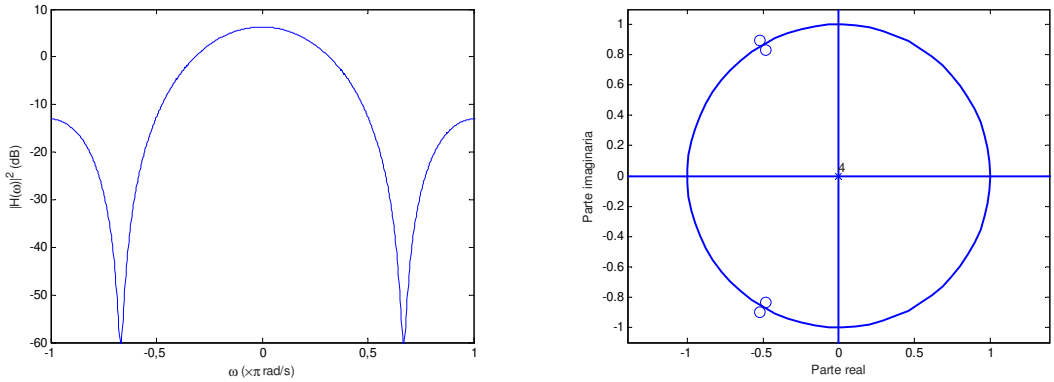


Figura 3.7: Função $\Psi_1(\gamma)$.

analisar os efeitos do canal, do código corretor de erros e do tamanho do entrelaçador sobre os resultados.

Em todas as simulações foi considerada a transmissão de 10^6 bits de informação para cada relação sinal-ruído. Os entrelaçadores foram gerados aleatoriamente, os bits codificados foram mapeados para os símbolos da constelação BPSK antes da transmissão e o ruído foi considerado como aditivo, branco, gaussiano, de média nula e de potência definida pela relação sinal-ruído. A decodificação foi realizada pelo algoritmo Log-MAP e o receptor possuía perfeito conhecimento do canal e da potência do ruído. Além disso, para cada bloco de dados recebido, o receptor realizou 15 iterações turbo.

Para verificarmos a influência do tamanho do entrelaçador no desempenho dos equalizadores turbo, consideramos que os bits de informação foram codificados pelo código convolucional sistemático recursivo de taxa 1/2 e geradores (7,5) e então transmitidos pelo canal $0,227 + 0,46z^{-1} + 0,688z^{-2} + 0,46z^{-3} + 0,227z^{-4}$, cujas características podem ser observadas nas Figs 3.8(a) e 3.8(b).



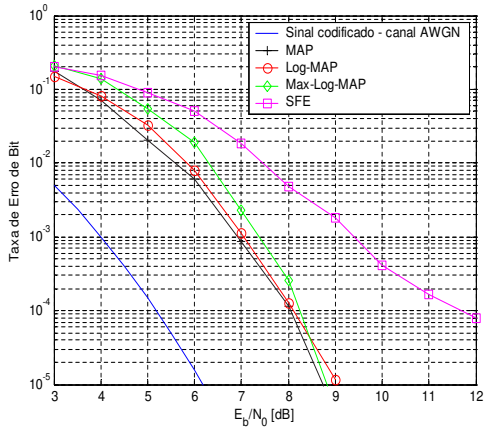
(a) Resposta em freqüência (b) Zeros

Figura 3.8: Características do canal $0,227+0,46z^{-1}+0,688z^{-2}+0,46z^{-3}+0,227z^{-4}$.

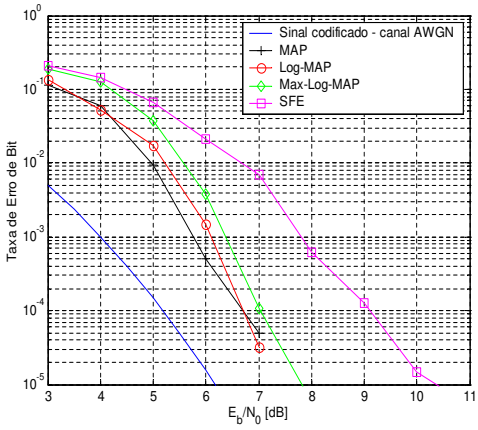
A Fig. 3.9 apresenta os resultados obtidos após a última iteração com os equalizadores MAP, Log-MAP, Max-Log-MAP e SFE ($M_1 = 10$ e $M_2 = 4$) para entrelaçadores de tamanho 2^i, $i = 8, \ldots, 12$. Também está traçada a curva de um sinal codificado em um canal AWGN, que não introduz IIS.

Como pode ser observado, o algoritmo Log-MAP apresenta praticamente o mesmo desempenho do MAP para todos os tamanhos de entrelaçadores simulados. Para um entrelaçador de 256 bits, estes dois algoritmos estão a cerca de 3 dB da curva do canal AWGN. Já para um entrelaçador de comprimento 2048, o MAP e o Log-MAP conseguem remover toda a IIS para E_b/N_0 maiores que 5 dB. Com um entrelaçador de 4096 bits, os dois algoritmos removem a IIS a partir de 4,5 dB.

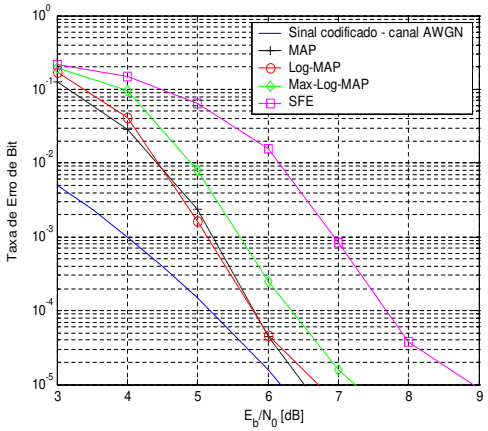
O algoritmo Max-Log-MAP tem cerca da metade da complexidade computacional do Log-MAP [9, 16] e apresenta uma perda de aproximadamente 0,4 dB em relação ao MAP e ao Log-MAP para todos os entrelaçadores simulados numa taxa de erro de bit, BER (*Bit Error Rate*), de 10^{-3}. Para o entrelaçador de 2048 bits, o Max-Log-MAP remove a IIS a partir de aproximadamente 6 dB. Porém, ao se dobrar o tamanho do entrelaçador, o equalizador apresenta um ganho de cerca de 0,5 dB, alcançando a curva de um sinal codificado transmitido num canal AWGN a partir de 5,5 dB.



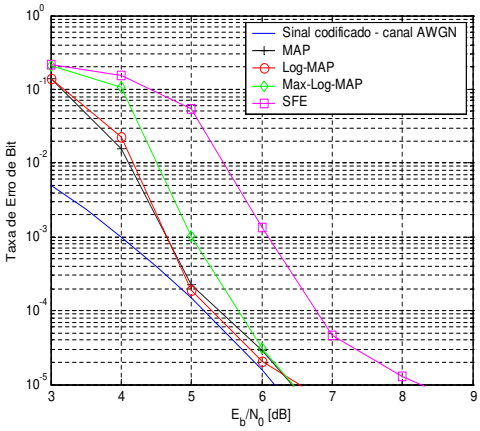
(a) Entrelaçador de 256 bits



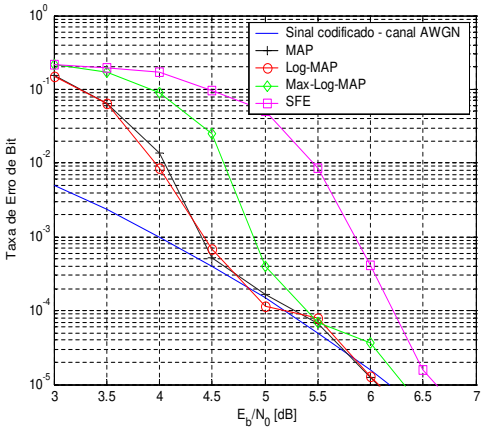
(b) Entrelaçador de 512 bits



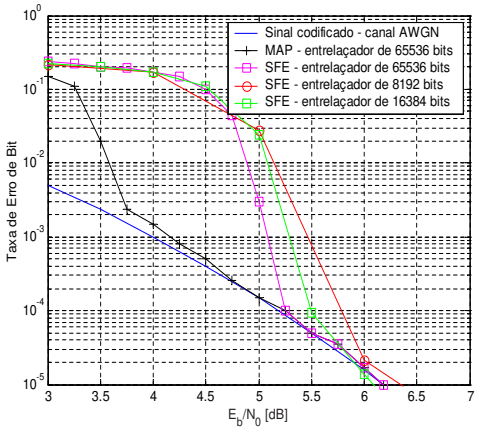
(c) Entrelaçador de 1024 bits



(d) Entrelaçador de 2048 bits



(e) Entrelaçador de 4096 bits



(f) Entrelaçadores de vários tamanhos

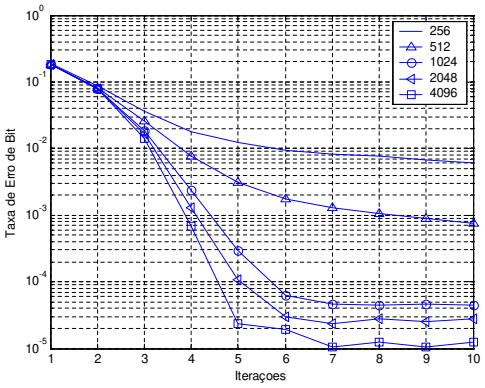
Figura 3.9: Comparação entre os algoritmos MAP, Log-MAP, Max-Log-MAP e SFE.

Já o desempenho do SFE em relação ao MAP varia com o tamanho do entrelaçador. Assim, para um entrelaçador de 256 bits a perda de desempenho para uma BER de 10^{-3} é de cerca de 2,5 dB, indo para mais ou menos 2 dB com um entrelaçador de 512 bits. Usando um entrelaçador de 1024 bits, o SFE fica 1,8 dB atrás do MAP, reduzindo esta distância para 1,4 dB com um entrelaçador de tamanho 4096. Por fim, a Fig. 3.9(f) mostra que, para um entrelaçador de 65536 bits, o SFE tem uma perda de mais ou menos 1 dB em relação ao BCJR. Ainda analisando a Fig. 3.9(f) percebemos que o ganho alcançado ao se passar de um entrelaçador de 8192 bits para um de 65536 bits é de aproximadamente 0,35 dB para uma BER de 10^{-3}. Isto indica que, a partir de um certo ponto, o aumento do tamanho do entrelaçador não conduz a uma melhora de desempenho que justifique o aumento da carga computacional.

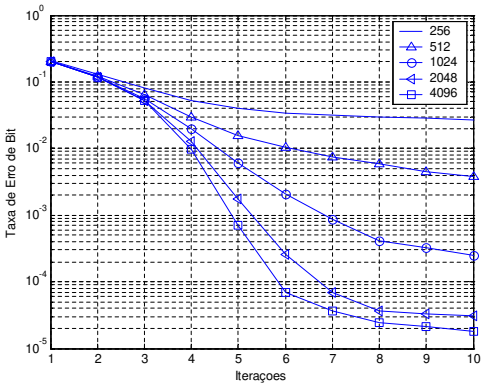
Convém notar que, independente do equalizador empregado, entrelaçadores maiores permitem alcançar uma determinada taxa de erro com um número menor de iterações. Isto ocorre porque, como esse canal apresenta uma IIS severa, entrelaçadores pequenos não são capazes de distribuir os erros por todo o bloco de dados [7, 32]. Conseqüentemente, toda a mensagem é corrompida e a detecção é prejudicada.

Para visualizarmos o comportamento dos algoritmos, traçamos na Fig. 3.10 a taxa de erro de bit dos equalizadores MAP, Max-Log-MAP e SFE para $E_b/N_0 = 6$ dB em função do números de iterações turbo realizadas. Nesta figura, vemos claramente que quanto maior o comprimento do entrelaçador menor o número de iterações necessárias para se alcançar uma determinada BER. Assim, para uma BER de 10^{-3} e um entrelaçador de 4096 bits, o equalizador MAP necessita de 4 iterações, enquanto que o Max-Log-MAP e o SFE devem realizar, respectivamente, 5 e 12 iterações. Porém, mesmo realizando um número maior de iterações, o SFE pode ter uma complexidade computacional menor que a do MAP. Em [28] é feita uma comparação entre as complexidades destes dois equalizadores.

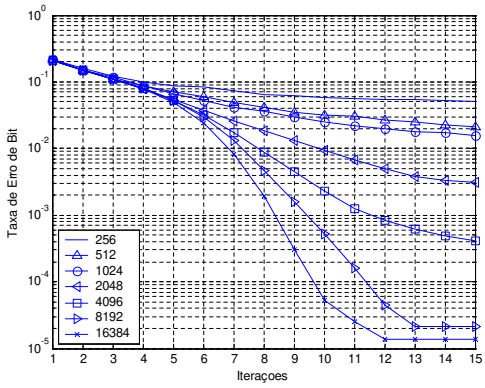
É exatamente esta redução de complexidade que permite que o SFE seja aplicado à equalização de canais com resposta ao impulso longa, onde os algoritmos



(a) MAP



(b) Max-Log-MAP



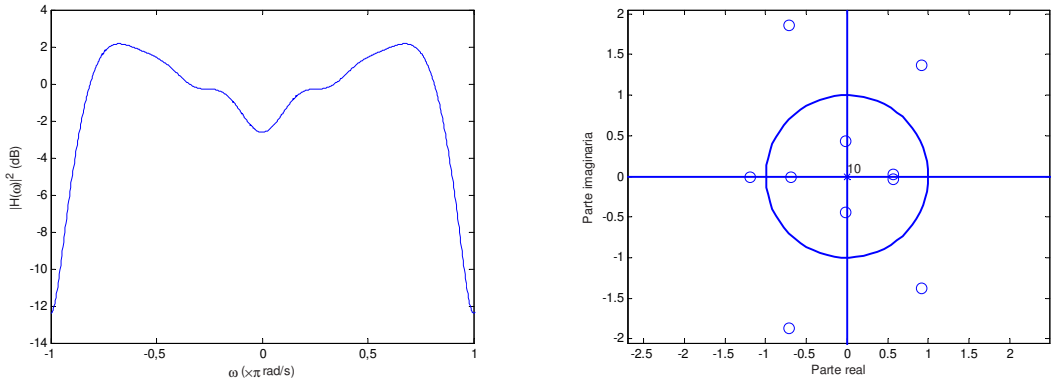
(c) SFE

Figura 3.10: BER para $E_b/N_0 = 6$ dB em função das iterações e do tamanho dos entrelaçadores.

de complexidade exponencial são inviáveis. Como exemplo, consideremos a transmissão de blocos de 128 bits de informação, codificados pelo código convolucional sistemático recursivo (7,5) e transmitidos pelo canal $0,07 + 0,03z^{-1} + 0,21z^{-2} + 0,36z^{-4} + 0,72z^{-5} - 0,5z^{-6} - 0,21z^{-7} + 0,07z^{-8} - 0,05z^{-9} + 0,04z^{-10}$.

As características deste canal estão na Fig. 3.11. O SFE empregado possuía $M_1 = 8$ e $M_2 = 4$. A Fig. 3.12 apresenta os resultados obtidos para 10 iterações do receptor. A diferença de desempenho entre a primeira e a última iterações eviden-

ciam o ganho proporcionado pela equalização turbo. Convém lembrar que a curva para a primeira iteração corresponde ao desempenho de um receptor convencional onde equalização e decodificação são realizadas separadamente.



(a) Resposta em freqüência (b) Zeros

Figura 3.11: Características do canal $0,07+0,03z^{-1}+0,21z^{-2}+0,36z^{-4}+0,72z^{-5}-0,5z^{-6}-0,21z^{-7}+0,07z^{-8}-0,05z^{-9}+0,04z^{-10}$.

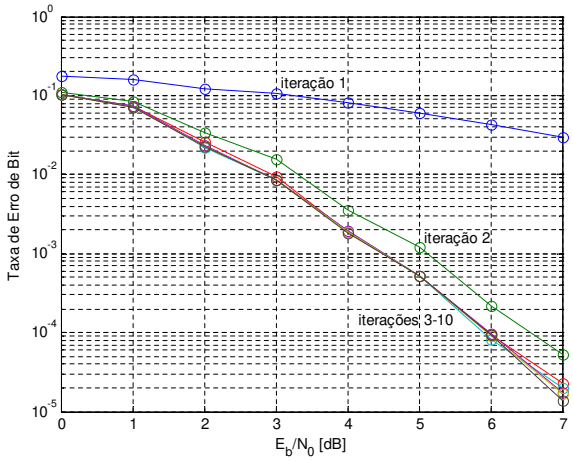
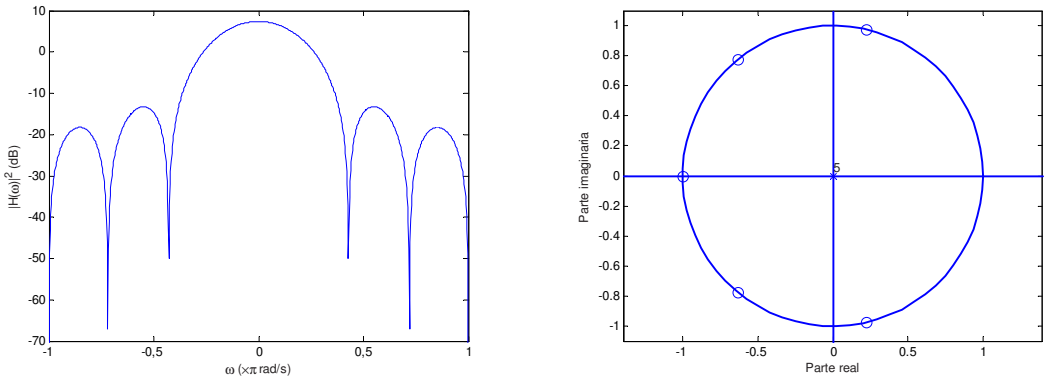


Figura 3.12: BER do SFE para o canal $0,07+0,03z^{-1}+0,21z^{-2}+0,36z^{-4}+0,72z^{-5}-0,5z^{-6}-0,21z^{-7}+0,07z^{-8}-0,05z^{-9}+0,04z^{-10}$.

Comparando as Figs. 3.9(a) e 3.12 notamos que o desempenho dos equalizadores turbo também depende das características do canal de transmissão. Quanto mais severa a IIS introduzida pelo canal, mais difícil é a sua remoção do sinal recebido. O canal $0,23 + 0,42z^{-1} + 0,52z^{-2} + 0,52z^{-3} + 0,42z^{-4} + 0,23z^{-5}$, cuja resposta em freqüência é mostrada na Fig. 3.13(a), é o canal de 6 coeficientes que causa a maior degradação de desempenho para o detector de seqüência de máxima verossimilhança [33]. Na Fig. 3.14 estão as curvas obtidas após a 15ª iteração considerando a transmissão de blocos de 2048 bits codificados. O SFE usado neste cenário possuía $M_1 = 15$ e $M_2 = 10$.



(a) Resposta em freqüência (b) Zeros

Figura 3.13: Características do canal $0,23+0,42z^{-1}+0,52z^{-2}+0,52z^{-3}+0,42z^{-4}+0,23z^{-5}$.

Pela Fig. 3.14, vemos que a degradação de desempenho do SFE em relação ao MAP e ao Log-MAP é de aproximadamente 3 dB para uma BER de 10^{-3}. Esta perda é bem maior que a apresentada na Fig. 3.9(d), onde se utilizou o mesmo código e o mesmo entrelaçador. O algoritmo Log-MAP continua com um desempenho igual ao do MAP.

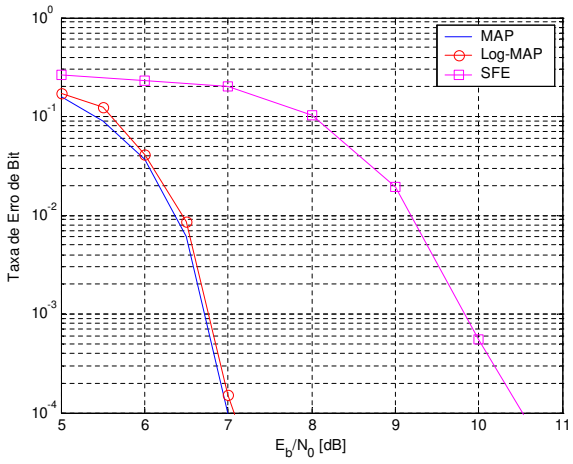
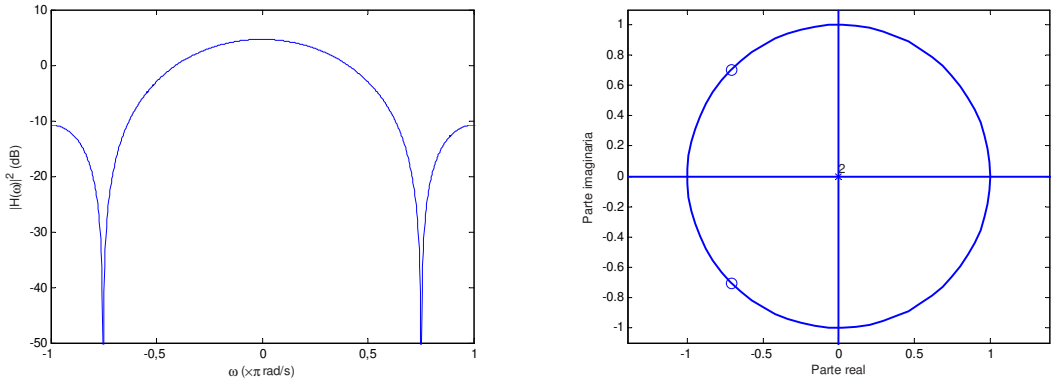


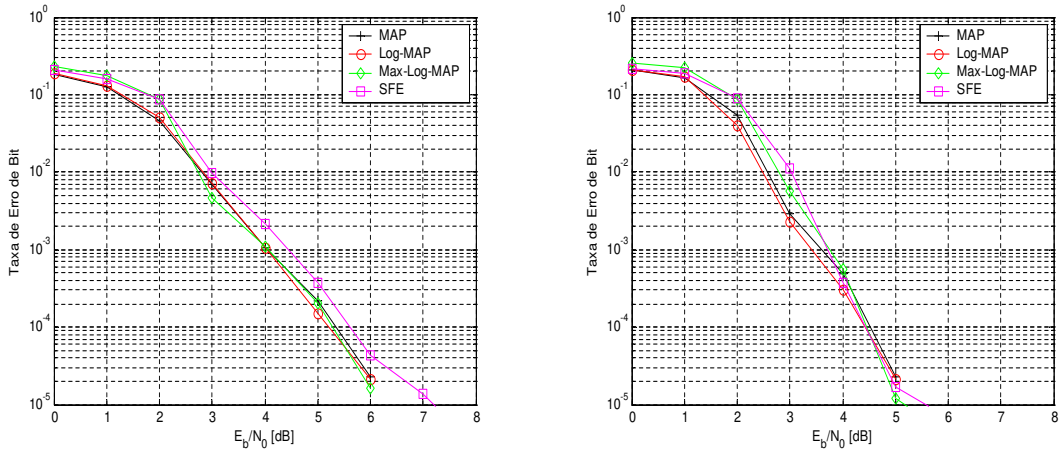
Figura 3.14: Desempenho de alguns equalizadores turbo para o canal $0,23 + 0,42z^{-1} + 0,52z^{-2} + 0,52z^{-3} + 0,42z^{-4} + 0,23z^{-5}$.

Para verificar a influência do código corretor de erros sobre o desempenho dos equalizadores turbo estudados, simulamos a transmissão de blocos de 1024 bits codificados através do canal $0,5+0,71z^{-1}+0,5z^{-2}$, cujas características se encontram nas Figs. 3.15(a) e 3.15(b). Utilizamos códigos RSC de geradores (7,5) e (23,35). Os gráficos resultantes das simulações estão nas Figs. 3.16(a) e 3.16(b).



(a) Resposta em freqüência (b) Zeros

Figura 3.15: Características do canal $0,5 + 0,71z^{-1} + 0,5z^{-2}$.



(a) RSC (7,5) (b) RSC (23,35)

Figura 3.16: Desempenho dos equalizadores turbo para o canal $0,5 + 0,71z^{-1} + 0,5z^{-2}$ para diferentes códigos.

Como podemos notar nestas figuras, para um mesmo código a diferença de desempenho entre os equalizadores é muito pequena. Isto ocorre uma vez que o canal considerado introduz uma IIS menos severa, permitindo que ela seja quase completamente removida do sinal recebido, mesmo para relações sinal-ruído baixas. Por outro lado, fica claro pelas Figs. 3.16(a) e 3.16(b) que códigos melhores produzem melhores resultados. Para o código (7,5), os equalizadores MAP, Log-MAP e Max-Log-MAP atingem uma BER de 10^{-3} a 4 dB enquanto que, para o código (23,35), esta taxa de erro é alcançada a aproximadamente 3,6 dB. O SFE se comporta exatamente como os outros três algoritmos para $E_b/N_0 > 4$ dB quando o código (23,35) é empregado. Já com o código (7,5), o SFE apresenta uma perda de cerca de 0,3 dB em todas as relações sinal-ruído simuladas.

A grande maioria dos trabalhos em equalização turbo utiliza códigos convolucionais sistemáticos recursivos. Estes códigos, como mencionado no capítulo 2, possuem a mesma estrutura dos códigos não-sistemáticos de mesmos geradores. Em [34] são feitas simulações demonstrando que, na equalização turbo, estas duas classes de códigos levam a um mesmo desempenho do receptor.

3.3.1 Aplicações em Comunicações Móveis

Atualmente, um dos grandes desafios impostos aos sistemas celulares é viabilizar a transmissão de dados, voz e imagem a altas taxas e com probabilidade de erro reduzida. A pesquisa em equalização turbo pode representar uma solução a esse problema.

Os canais de banda larga utilizados em sistema celulares de 3ª e 4ª gerações sofrem severas distorções causadas pelo desvanecimento, que atinge diferentemente cada componente de freqüência do sinal enviado (desvanecimento seletivo em freqüência). Para esses canais, a equalização turbo apresenta resultados bastante superiores ao de técnicas tradicionais, como pode ser observado em [4, 7, 23, 32, 35].

Para avaliar o desempenho de um equalizador turbo neste contexto, simulamos a transmissão de 802 blocos de 1248 bits de informação, equivalentes a 8 *slots* GSM, através de 4 modelos de canais ditados pelo COST207 [36]. O equalizador turbo empregado utilizou o SFE com $M_1 = 8$ e $M_2 = 4$ e os bits de informação foram codificados pelo código RSC de taxa 1/2 e geradores (7,5). Além disso, a transmissão dos bits codificados foi realizada usando um pulso cosseno levantado com fator de *roll-off* igual a 0,35. A Fig. 3.17 mostra as curvas obtidas após a 5ª iteração para um móvel se deslocando a 60 km/h.

Notamos, na Fig. 3.17, que o desempenho do receptor nos ambientes *typical rural*, *typical urban* e *bad urban* é praticamente o mesmo até uma relação sinal-ruído de 4 dB. A partir deste ponto, as taxas de erro de bit alcançadas no ambiente *typical rural* são menores que aquelas alcançadas nos outros canais, o que já era esperado uma vez que este canal possui um número reduzido espalhadores. Assim, para uma BER de 10^{-4}, o receptor no ambiente *typical urban* tem uma perda de aproximadamente 1 dB em relação ao canal *typical rural*, enquanto que o canal *bad urban*, tem uma degradação de cerca de 2 dB. Também podemos perceber que o desempenho do equalizador turbo no ambiente *hilly terrain* é muito inferior ao desempenho mostrado no demais ambientes. Isto ocorre porque o canal *hilly terrain* apresenta uma IIS severa, distorcendo significativamente o sinal transmitido.

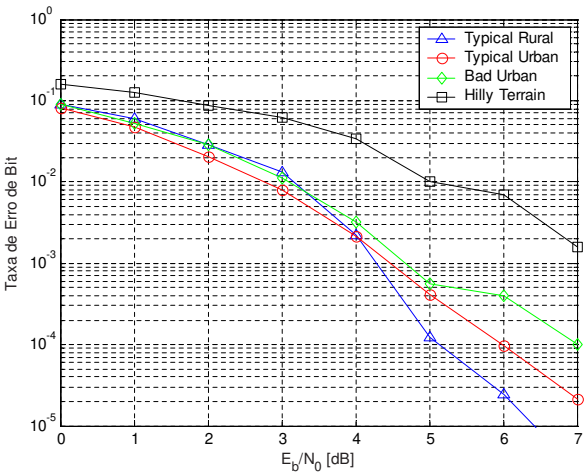


Figura 3.17: Desempenho do equalizador turbo com o SFE para um móvel a 60 km/h.

Além do desvanecimento, outro fenômeno típico dos ambientes sem fio e que degrada significativamente o sinal recebido é o ruído impulsivo. Aqui também, a equalização turbo parece uma solução promissora, como podemos observar em [37], onde é desenvolvido um equalizador Bayesiano cego próprio para ambientes com este tipo de ruído.

3.4 Conclusão

Neste capítulo, apresentamos os conceitos básico da equalização turbo. Neste novo paradigma, equalização e decodificação são realizadas conjuntamente através de um procedimento iterativo. Com isso, o equalizador pode se beneficiar da capacidade de correção de erros do código para remover a IIS de uma maneira mais efetiva. Os principais equalizadores de complexidade exponencial foram apresentados, bem como alguns equalizadores de complexidade reduzida. Dentre estes se destaca o SFE, uma técnica de cancelamento de interferência que utiliza a própria saída do equalizador para obter estimativas mais confiáveis dos símbolos interferentes. Para

reduzir a complexidade computacional e simplificar o cálculo dos coeficientes do SFE, propusemos uma aproximação para a função $\Psi_1(\gamma)$. Além disso, verificamos através de simulações a influência do código corretor de erros, do canal e do tamanho do entrelaçador sobre o desempenho do receptor.

Todos os equalizadores descritos neste capítulo pressupõem conhecimento preciso dos coeficientes do canal e da potência do ruído aditivo. Quando estas informações não estão disponíveis, é necessário estimá-las para que se possa continuar empregando as estruturas estudadas. Uma possível solução consiste em incluir um algoritmo de estimação de canal na malha de realimentação de um equalizador turbo. Desta maneira, o estimador de canal pode também se beneficiar da redundância introduzida pelo código. Esta abordagem será discutida em detalhes no próximo capítulo.

4

Estimação, Equalização e Decodificação Iterativas

Todos os equalizadores turbo descritos no capítulo anterior supõem que os coeficientes do canal discreto equivalente e a potência do ruído aditivo são conhecidos perfeitamente pelo receptor. No entanto, em situações reais, como nos canais de comunicações móveis, isto nem sempre ocorre. Para solucionar este problema, podemos vislumbrar duas alternativas: desenvolver equalizadores SISO adaptativos ou obter estimativas dos parâmetros desconhecidos para que se possam empregar os equalizadores descritos no capítulo 3.

Em ambas as soluções, podemos ou não recorrer ao uso de seqüências de treinamento. Estas seqüências nada mais são que um conjunto de símbolos pré-definidos e conhecidos pelo receptor que são incluídos no sinal transmitido. O receptor uti-

liza esta seqüência e o sinal recebido correspondente para calcular os coeficientes do equalizador ou estimar o canal. Como o receptor conhece o sinal enviado, a seqüência de treinamento, este método é dito supervisionado. Uma desvantagem dos métodos supervisionados é que nem todo o sinal enviado corresponde à informação que se deseja transmitir, causando assim uma perda de eficiência. Além disso, nos casos em que o canal varia rapidamente, é necessário repetir o treinamento com a devida freqüência.

Já os métodos não-supervisionados, também conhecidos como autodidatas ou cegos, não utilizam seqüências de treinamento para equalizar ou estimar o canal, mas apenas algumas características estatísticas do sinal transmitido. Assim, todo o sinal recebido corresponde à informação "útil". Por este motivo, todo o restante deste capítulo será dedicado ao estudo de técnicas cegas.

Como mencionado previamente, uma primeira alternativa para a equalização turbo quando o canal e a potência do ruído não são conhecidos consiste no desenvolvimento de equalizadores SISO cegos. Contudo, a obtenção de equalizadores cegos que aceitem informações *a priori* na forma de LLR e forneçam informações extrínsecas na saída não é tão simples. Desta maneira, a solução normalmente escolhida é aquela que estima os parâmetros desconhecidos do canal para então usá-los nos equalizadores turbo.

Uma primeira abordagem, adotada em [35, 37, 38], consiste em utilizar o sinal na saída do equalizador para realizar a estimação do canal. O receptor proposto em [35], que tem complexidade exponencial, utiliza as saídas de um equalizador Max-Log-MAP para alimentar um estimador de canal adaptado pelo algoritmo LMS. Já em [38], o equalizador usa o processamento por sobrevivente (PSP - *Per-Survivor Processing*) para calcular estimativas do canal para cada um dos caminhos sobreviventes da treliça, o que pode ser proibitivo. O equalizador proposto em [37] usa um método de Monte Carlo, denominado *Gibbs sampler*, para fazer uma estimação Bayesiana conjunta do canal, da variância do ruído e dos sinais transmitidos. A idéia básica é gerar amostras aleatórias a partir da distribuição conjunta *a posteriori* $p\left(\mathbf{h}, \sigma_n^2, \mathbf{x}|\mathbf{r}\right)$ e então estimar qualquer distribuição marginal usando estas

amostras. É possível mostrar [37] que, no limite, as estimativas dessas distribuições convergem para as distribuições marginais exatas.

Nos receptores iterativos, no entanto, há a possibilidade de incluir os algoritmos de identificação na malha de realimentação do receptor. Conseqüentemente, o estimador de canal pode se beneficiar do código corretor de erros, aprimorando as estimativas dos parâmetros desconhecidos a cada iteração. Este esquema, mostrado na Fig. 4.1 e já utilizado em outros trabalhos, será empregado em todos os receptores turbo cegos estudados neste capítulo.

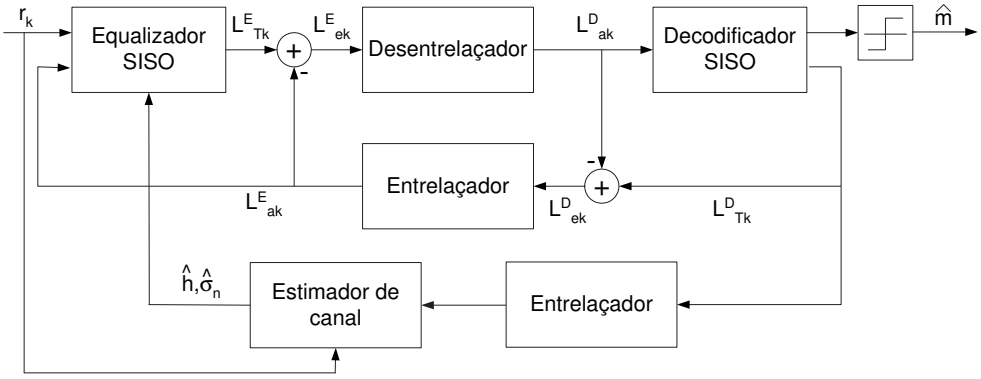


Figura 4.1: Esquema geral de um estimador turbo.

Em linhas gerais, podemos descrever o funcionamento do receptor da Fig. 4.1 da seguinte maneira: a partir de uma estimativa inicial para o canal e a variância do ruído, o equalizador gera informações extrínsecas e as passa para o decodificador, que por sua vez, também produz estimativas das probabilidades *a posteriori* de cada bit transmitido. Estas probabilidades *a posteriori*, juntamente com os sinais recebidos, são usados pelo estimador de canal para produzir novas estimativas dos parâmetros desejados, que serão utilizados pelo equalizador na iteração seguinte. De posse de estimativas mais precisas, o equalizador produz saídas mais confiáveis, refinando a decodificação e, conseqüentemente, a estimação, recomeçando o ciclo. Como já foi dito anteriormente, nos restringiremos ao estudo de receptores cegos. Porém, nada impede que a estrutura da Fig. 4.1 seja empregada em receptores supervisionados

ou em receptores semicegos, que são assim chamados por utilizarem não apenas a seqüência de treinamento, mas toda a seqüência recebida, para estimar o canal.

É interessante notar que, mesmo quando o estimador turbo funciona sem supervisão, o estimador de canal se comporta como um algoritmo supervisionado, no qual a seqüência de treinamento corresponde às estimativas dos símbolos transmitidos, obtidas a partir das LLR's fornecidas pelo decodificador, e o sinal desejado é o próprio sinal recebido. Deste modo, podemos empregar algoritmos tradicionalmente classificados como supervisionado, como o LMS e o filtro de Kalman, descritos respectivamente nas subseções 2.4.1 e 2.4.2, para realizar a estimação cega do canal.

Muitos trabalhos, como aqueles em [28, 30, 39–41], exploram a estrutura do receptor da Fig. 4.1 para realizar a estimação cega do canal e dos sinais transmitidos. Em [28, 30], o estimador proposto consiste numa simplificação do algoritmo EM (*Expectation-Maximization*) que apresenta uma complexidade por símbolo proporcional ao comprimento do canal. Já em [39, 40], o estimador desenvolvido é um filtro de Kalman que incorpora as informações suaves geradas pelo decodificador e possui uma complexidade quadrática em relação ao comprimento do canal. Uma modificação do algoritmo RLS (*Recursive Least Squares*) é empregada para estimar o canal em [41].

Assim como ocorre com os equalizadores SISO usados na equalização turbo, a busca por algoritmos de identificação de baixa complexidade computacional vem se intensificando. Dentre as várias soluções existentes, vamos centrar nossa atenção naquelas que empregam filtros de Kalman e suas variações. Dentre os inúmeros trabalhos estudados sobre estimação turbo, apenas [39, 40] utilizam essa classe de filtros. Mais precisamente, propomos neste capítulo a utilização de algoritmos de mínimos quadrados rápidos, como o *Fast Kalman* [42–44], o FTF (*Fast Transversal Filter*) [44,45] e o FAEST (*Fast a Posteriori Error Sequential Technique*) [44,46] para estimar os coeficientes do canal num esquema como o da Fig. 4.1. Estes algoritmos, embora computacionalmente eficientes, não vêm sendo explorados no contexto de identificação iterativa.

Nas próximas seções, descreveremos brevemente os algoritmos rápidos e o filtro

de Kalman modificado de [39, 40]. Nessas descrições, o vetor de entrada $\mathbf{x}$ corresponde às estimativas dos sinais transmitidos obtidas com as LLR's geradas pelo decodificador. Em seguida, apresentaremos comparações entre estes algoritmos e o LMS, empregados num arranjo como o da Fig. 4.1, em diversos cenários de simulação. Ao término do capítulo, estará claro que é possível obter receptores cegos de baixa complexidade e bom desempenho usando o SFE em conjunto com um dos algoritmos rápidos.

4.1 Filtro de Kalman Modificado

O estimador de canal proposto em [39, 40] consiste, de fato, numa modificação do filtro de Kalman convencional na qual as informações suaves produzidas pelo decodificador são incorporadas à descrição estatística do canal. No desenvolvimento do algoritmo, a entrada do estimador é tratada como um sinal estocástico. Isto quer dizer que o sinal x_k que alimenta o filtro é escrito como $x_k = \bar{x}_k + \tilde{x}_k$, onde $\bar{x}_k$ é o valor médio de x_k e $\tilde{x}_k$ é um "ruído" de média zero e variância σ_x^2. Sendo L_{Tk}^D a LLR produzida pelo decodificador, $\bar{x}_k$ e σ_x^2 são facilmente obtidos através das expressões

$$\bar{x}_k = \mathrm{E}\left[x|L_{Tk}^D\right] = \tanh\left(L_{Tk}^D/2\right), \tag{4.1a}$$

$$\sigma_{x,k}^2 = \mathrm{E}\left[(x-\bar{x})^2\,|L_{Tk}^D\right] = 1-\bar{x}_k^2. \tag{4.1b}$$

Portanto, podemos escrever as equações de estado deste filtro de Kalman modificado como

$$\begin{aligned} \mathbf{x}_k &= \bar{\mathbf{x}}_k + \tilde{\mathbf{x}}_k, \\ \mathbf{h}_{k+1} &= \mathbf{F}\mathbf{h}_k + \mathbf{B}\mathbf{n}_{1,k}, \\ r_k &= \mathbf{h}_k^{\mathrm{H}}\bar{\mathbf{x}}_k + n_{2,k}, \end{aligned} \tag{4.2}$$

onde supõe-se que $\mathrm{E}\left[\mathbf{n}_{1,k+j}\mathbf{n}_{1,k}^{\mathrm{H}}\right] = \mathbf{Q}_{n_1}\delta_j$ e $n_{2,k} = \mathbf{h}_k^{\mathrm{H}}\tilde{\mathbf{x}}_k + n_k$, sendo n o ruído AWGN do canal.

É importante salientar que o ruído $n_{2,k}$ na equação de observação incorpora na estrutura do filtro de Kalman as informações suaves fornecidas pelo decodificador.

Além disso, supondo um modelo de canal WSSUS (*Wide Sense Stationary Uncorrelated Scattering*), isto é, estacionário no sentido amplo com propagação em múltiplos percursos e espalhadores descorrelacionados, é possível mostrar [39, 40] que

$$\mathrm{E}\left[n_{2,k+j}n_{2,k}^{*}\right] = \mathrm{tr}\left\{\mathbf{Q}_{n_2}\right\}\delta_j, \tag{4.3}$$

com $\mathbf{Q}_{n_2} = \mathbf{R}_k\mathbf{S}_{k,k} + \sigma_n^2$, $\mathbf{R}_k = \mathrm{E}\left[\mathbf{h}_k\mathbf{h}_k^{\mathrm{H}}\right]$, $\mathbf{S}_{k,k} = \mathrm{E}\left[\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k^{\mathrm{H}}\right]$ e $\mathrm{tr}\left\{\cdot\right\}$ sendo o traço de uma matriz.

Definindo $\mathbf{x}_k = \left[x_k, \ldots, x_{k-\mu+1}\right]^{\mathrm{T}}$, $\bar{\mathbf{x}}_k = \left[\bar{x}_k, \ldots, \bar{x}_{k-\mu+1}\right]^{\mathrm{T}}$, μ como o comprimento do canal, $\mathbf{K}_{k|k-1} = \mathrm{E}\left[\varepsilon_{k|k-1}\varepsilon_{k|k-1}^{\mathrm{H}}\right]$, $\varepsilon_{k|k-1} = \mathbf{h}_k - \hat{\mathbf{h}}_{k|k-1}$, $\hat{\mathbf{h}}_{i|j}$ como as estimativas do canal feitas no instante i a partir dos dados disponíveis até o instante j e $\mathbf{G}_k$ como o ganho de Kalman, obtém-se o filtro de Kalman modificado [39, 40], apresentado na Tab. 4.1.1. Este algoritmo, assim como o filtro de Kalman clássico, apresenta uma complexidade computacional proporcional ao quadrado do número de coeficientes do estimador de canal.

Algoritmo 4.1.1 Filtro de Kalman modificado

$$\hat{\mathbf{h}}_{k|k-1} = \mathbf{F}\hat{\mathbf{h}}_{k-1|k-1}$$
$$\alpha_k = r_k - \hat{\mathbf{h}}_{k|k-1}\bar{\mathbf{x}}_k$$
$$\mathbf{R}_k = \hat{\mathbf{h}}_{k|k-1}\hat{\mathbf{h}}_{k|k-1}^{\mathrm{H}} + \mathbf{K}_{k,k-1}$$
$$\mathbf{Q}_{n_2} = \mathrm{diag}\left[\sigma_{x,k}^2, \cdots, \sigma_{x,k-\mu+1}^2\right]\mathbf{R}_k + \sigma_n^2$$
$$\mathbf{G}_k = \frac{\mathbf{K}_{k,k-1}\mathbf{x}_k}{\mathrm{tr}\left\{\mathbf{Q}_{n_2}\right\} + \bar{\mathbf{x}}_k^{\mathrm{H}}\mathbf{K}_{k,k-1}\bar{\mathbf{x}}_k}$$
$$\hat{\mathbf{h}}_{k|k} = \hat{\mathbf{h}}_{k|k-1} + \mathbf{G}_k\alpha_k^{*}$$
$$\mathbf{K}_{k+1,k} = \mathbf{F}\left[\mathbf{I} - \mathbf{G}_k\bar{\mathbf{x}}_k^{\mathrm{H}}\right]\mathbf{K}_{k,k-1}\mathbf{F}^{\mathrm{H}} + \mathbf{B}\mathbf{Q}_{n_1}\mathbf{B}^{\mathrm{H}}$$

4.2 Algoritmos Rápidos

O filtro de Kalman clássico, que é ótimo no sentido de minimizar o erro quadrático médio entre o sinal de entrada e o sinal desejado, possui complexidade computacional proporcional ao quadrado do número de coeficientes do estimador de canal. Um primeiro passo na redução da complexidade deste algoritmo foi dado por Ljung, Morf e Falconer [42] com a proposição do primeiro algoritmo rápido de mínimos quadrados, denominado "*Fast Kalman*". O *Fast Kalman*, assim como outros algoritmos rápidos, tem complexidade proporcional ao número de coeficientes do estimador de canal. Esta redução é alcançada graças à exploração da natureza seqüencial do sinal de entrada. De fato, o vetor de entrada $\mathbf{x}_k$ pode ser obtido, a menos do primeiro elemento, por um simples deslocamento nos elementos de $\mathbf{x}_{k-1}$. Outra característica comum aos algoritmos rápidos estudados é que eles são matematicamente equivalentes ao filtro de Kalman tradicional.

De forma geral, os algoritmos rápidos podem ser descritos como formados por um conjunto de quatro filtros transversais, $\mathbf{A}_k$, $\mathbf{B}_k$, $\hat{\mathbf{h}}_k$ e $\mathbf{G}_k$, todos excitados pela mesma seqüência de entrada $\mathbf{x}_k$ [45]. Aqui, $\mathbf{A}_k$ é um preditor progressivo, $\mathbf{B}_k$ é um preditor regressivo, $\hat{\mathbf{h}}_k$ é o vetor com as estimativas dos coeficientes do canal e $\mathbf{G}_k$ é o vetor com o ganho de Kalman.

A seguir, na Tab. 4.2.1 da pag. 66, expomos o algoritmo *Fast Kalman*. Nestas equações, r_k é o sinal recebido no instante k, w é o fator de esquecimento, μ é o comprimento do canal, $\mathbf{x}_k = [x_k, \cdots, x_{k-\mu+1}]^{\mathrm{T}}$, x_k é a estimativa do bit transmitido no instante k obtida a partir da LLR produzida pelo decodificador e E_{ak} é a energia do erro de predição progressiva ε_{ak}, dado por $\varepsilon_{ak} = x_k - \mathbf{A}_k^{\mathrm{H}}\mathbf{x}_{k-1}$. A dedução detalhada deste algoritmo rápido pode ser encontrada em [42, 43].

Já o FTF e o FAEST consistem basicamente em modificações do *Fast Kalman*, com ligeira redução na complexidade computacional. Para o FTF, é comum se definir o vetor de ganho dual:

$$\tilde{\mathbf{G}}_k = \frac{1}{\gamma_k}\mathbf{G}_k, \tag{4.4}$$

onde $\gamma_k = \varepsilon_k / e_k$, conhecida como *variável de verossimilhança* ou *variável de ante-*

Algoritmo 4.2.1 Fast Kalman

`disponível no instante k-1`

$$\mathbf{x}_{k-1}, \hat{\mathbf{h}}_{k-1}, \mathbf{A}_{k-1}, \mathbf{B}_{k-1}, \mathbf{G}_{k-1}, E_{a,k-1}$$

`recebido no instante k`

$$x_k, r_k$$

`atualização dos preditores`

$$\begin{aligned}
e_{ak} &= x_k - \mathbf{A}_{k-1}^{\mathrm{H}} \mathbf{x}_{k-1} \\
\mathbf{A}_k &= \mathbf{A}_{k-1} + e_{ak}^* \mathbf{G}_{k-1} \\
\varepsilon_{ak} &= x_k - \mathbf{A}_k^{\mathrm{H}} \mathbf{x}_{k-1} \\
E_{a,k} &= w E_{a,k-1} + e_{ak}^* \varepsilon_{ak} \\
\begin{bmatrix} \mathbf{M}_k \\ m_k \end{bmatrix} &= \begin{bmatrix} 0 \\ \mathbf{G}_{k-1} \end{bmatrix} + \frac{\varepsilon_{ak}}{E_{a,k}} \begin{bmatrix} 1 \\ -\mathbf{A}_k \end{bmatrix} \\
e_{bk} &= x_{k-\mu} + \mathbf{B}_{k-1}^{\mathrm{H}} \mathbf{x}_k \\
\mathbf{G}_k &= \frac{1}{1 - m_k e_{bk}^*} \left[\mathbf{M}_k + m_k \mathbf{B}_{k-1} \right] \\
\mathbf{B}_k &= \mathbf{B}_{k-1} + e_{bk}^* \mathbf{G}_k
\end{aligned}$$

`atualização do estimador de canal`

$$\begin{aligned}
\alpha_k &= r_k - \hat{\mathbf{h}}_{k-1}^{\mathrm{H}} \mathbf{x}_k \\
\hat{\mathbf{h}}_k &= \hat{\mathbf{h}}_{k-1} + \mathbf{G}_k \alpha_k^*
\end{aligned}$$

cipação, é a razão entre os erros de predição *a posteriori* e *a priori*. Essa variável recebe esta última denominação pois permite que os erros *a posteriori* sejam obtidos antes mesmo do que os filtros que os determinam. O algoritmo FTF, desenvolvido em [45], é apresentado na Tab. 4.2.2 da pag. 68.

O FAEST, proposto em [46] e descrito na Tab. 4.2.3 da pag. 69, é muito semelhante ao FTF, diferindo apenas no uso da variável de verossimilhança. Ao invés de γ_k, o FAEST emprega

$$\zeta_k = \frac{1}{\gamma_k}. \tag{4.5}$$

A tabela abaixo [43, 45] ilustra a carga computacional dos algoritmos estudados.

Algoritmo	Multiplicações	Somas
Gradiente	2μ	2μ
Fast Kalman	$10\mu + 4$	$12\mu + 5$
FTF	$7\mu + 11$	$7\mu + 3$
FAEST	$7\mu + 10$	$7\mu + 4$
Kalman convencional	$3\mu^2 + 3\mu$	$2\mu^2 + 2\mu + 1$

Porém, é necessário estimar não apenas os coeficientes do canal mas também a variância do ruído. Para tanto, em conjunto com os vários algoritmos de estimação estudados, empregamos o seguinte estimador [28]:

$$\hat{\sigma}_n^2 = \frac{1}{L} \sum_{k=0}^{L-1} \left| r_k - \hat{\mathbf{x}}^{\mathrm{T}} \hat{\mathbf{h}} \right|^2, \tag{4.6}$$

onde L é o número de símbolos enviados e $\hat{\mathbf{x}}$ é o vetor com as estimativas dos sinais transmitidos obtidas a partir de decisões abruptas sobre as LLR's do decodificador, isto é, $\hat{x}_k = \operatorname{sign}\left[L_{Tk}^D\right]$.

Algoritmo 4.2.2 FTF

disponível no instante k-1

$$\mathbf{x}_{k-1}, \hat{\mathbf{h}}_{k-1}, \mathbf{A}_{k-1}, \mathbf{B}_{k-1}, \tilde{\mathbf{G}}_{k-1}, \gamma_{k-1}, E_{a,k-1}, E_{b,k-1}$$

recebido no instante k

$$x_k, r_k$$

atualização dos preditores

$$\begin{aligned}
e_{ak} &= x_k - \mathbf{A}_{k-1}^{\mathrm{H}} \mathbf{x}_{k-1} \\
\begin{bmatrix} \tilde{\mathbf{M}}_k \\ \tilde{m}_k \end{bmatrix} &= \begin{bmatrix} 0 \\ \tilde{\mathbf{G}}_{k-1} \end{bmatrix} + \frac{e_{ak}}{wE_{a,k-1}} \begin{bmatrix} 1 \\ -\mathbf{A}_{k-1} \end{bmatrix} \\
\varepsilon_{ak} &= \gamma_{k-1} e_{ak} \\
\mathbf{A}_k &= \mathbf{A}_{k-1} + \varepsilon_{ak}^* \tilde{\mathbf{G}}_{k-1} \\
\tilde{\mathbf{G}}_k &= \tilde{\mathbf{M}}_k + \tilde{m}_k \mathbf{B}_{k-1} \\
E_{ak} &= wE_{a,k-1} + e_{ak} \varepsilon_{ak}^* \\
e_{bk} &= wE_{b,k-1} \tilde{m}_k \\
\gamma_{1k} &= \gamma_{k-1} \frac{wE_{ak-1}}{E_{a,k}} \\
\gamma_k &= \frac{\gamma_{1k}}{1 - \gamma_{1k} \tilde{m}_k e_{bk}^*} \\
\varepsilon_{bk} &= \gamma_k e_{bk} \\
\mathbf{B}_k &= \mathbf{B}_{k-1} + \varepsilon_{bk}^* \tilde{\mathbf{G}}_k \\
E_{bk} &= wE_{b,k-1} + e_{bk} \varepsilon_{bk}^*
\end{aligned}$$

atualização do estimador de canal

$$\begin{aligned}
\alpha_k &= r_k - \hat{\mathbf{h}}_{k-1}^{\mathrm{H}} \mathbf{x}_k \\
\hat{\mathbf{h}}_k &= \hat{\mathbf{h}}_{k-1} + \gamma_k \alpha_k^* \tilde{\mathbf{G}}_k
\end{aligned}$$

Algoritmo 4.2.3 FAEST

disponível no instante k-1

$$\mathbf{x}_{k-1}, \hat{\mathbf{h}}_{k-1}, \mathbf{A}_{k-1}, \mathbf{B}_{k-1}, \tilde{\mathbf{G}}_{k-1}, \zeta_{k-1}, E_{a,k-1}, E_{b,k-1}$$

recebido no instante k

$$x_k, r_k$$

atualização dos preditores

$$\begin{aligned}
e_{ak} &= x_k - \mathbf{A}_{k-1}^{\mathrm{H}} \mathbf{x}_{k-1} \\
\begin{bmatrix} \tilde{\mathbf{M}}_k \\ \tilde{m}_k \end{bmatrix} &= \begin{bmatrix} 0 \\ \tilde{\mathbf{G}}_{k-1} \end{bmatrix} + \frac{e_{ak}}{wE_{a,k-1}} \begin{bmatrix} 1 \\ -\mathbf{A}_{k-1} \end{bmatrix} \\
\varepsilon_{ak} &= \frac{e_{ak}}{\zeta_{k-1}} \\
\mathbf{A}_k &= \mathbf{A}_{k-1} + \varepsilon_{ak}^* \tilde{\mathbf{G}}_{k-1} \\
\tilde{\mathbf{G}}_k &= \tilde{\mathbf{M}}_k + \tilde{m}_k \mathbf{B}_{k-1} \\
E_{a,k} &= wE_{a,k-1} + e_{ak}\varepsilon_{ak}^* \\
e_{bk} &= wE_{b,k-1}\tilde{m}_k \\
\zeta_{1k} &= \zeta_{k-1} + \frac{|e_{ak}|^2}{wE_{a,k-1}} \\
\zeta_k &= \zeta_{1k} - \tilde{m}_k e_{bk}^* \\
\varepsilon_{bk} &= \frac{e_{bk}}{\zeta_k} \\
\mathbf{B}_k &= \mathbf{B}_{k-1} + \varepsilon_{bk}^* \tilde{\mathbf{G}}_k \\
E_{b,k} &= wE_{b,k-1} + e_{bk}\varepsilon_{bk}^*
\end{aligned}$$

atualização do estimador de canal

$$\begin{aligned}
\alpha_k &= r_k - \hat{\mathbf{h}}_{k-1}^{\mathrm{H}} \mathbf{x}_k \\
\hat{\mathbf{h}}_k &= \hat{\mathbf{h}}_{k-1} + \frac{\alpha_k^*}{\zeta_k} \tilde{\mathbf{G}}_k
\end{aligned}$$

4.3 Condições Iniciais

Como dito anteriormente, o equalizador na Fig. 4.1 necessita de uma estimativa inicial do canal e da variância do ruído na primeira iteração turbo. Nos casos em que uma seqüência de treinamento está disponível, ela pode ser usada para estimar esses parâmetros. Porém, nos receptores cegos, esses valores devem ser obtidos de alguma outra forma. Assim, nesta dissertação, consideramos que o comprimento do canal é conhecido e que a estimativa inicial, $\hat{\mathbf{h}}_{ini}$, tem apenas um coeficiente não-nulo. O valor deste coeficiente é dado por $\hat{\sigma}_{n,ini}$, onde

$$\hat{\sigma}^2_{n,ini} = \frac{1}{2L}\sum_{k=0}^{L-1}|r_k|^2 \tag{4.7}$$

é a estimativa inicial da variância do ruído. Estas condições iniciais estimam a relação sinal-ruído em 0 dB enquanto mantêm os valores de $\hat{\mathbf{h}}_{ini}$ e $\hat{\sigma}_{n,ini}$ consistentes com a energia recebida [28]. Os algoritmos rápidos utilizam, na primeira iteração, a estimativa do canal assim obtida como ponto de partida. A partir da segunda iteração, os estimadores de canal empregam como valores iniciais os coeficientes calculados por eles na iteração anterior.

Já os filtros $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$ e $\tilde{\mathbf{G}}$ são zerados a cada iteração. Para manter a coerência com estas inicializações, γ_0 e ζ_0 devem ser iguais a 1 [44]. Por fim, os valores das energias dos erros de predição devem ser escolhidos de maneira a não comprometer a estabilidade dos algoritmos nos instantes iniciais. Nesta dissertação, fizemos $E_{a,0} = E_0$ e $E_{b,0} = w^{-\mu}E_0$, com $E_0 = 10^{-4}$. É muito importante mencionar que o processamento por blocos realizado nos receptores turbo, juntamente com essas inicializações, garantem a estabilidade dos algoritmos rápidos.

4.4 Simulações

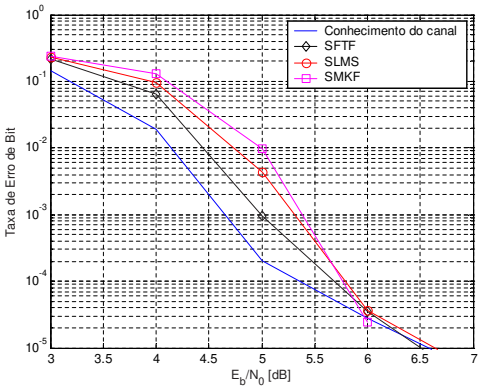
Nesta seção, analisaremos o desempenhos dos algoritmos rápidos, do LMS e do filtro de Kalman modificado como estimadores de canal em receptores turbo cegos

como o da Fig. 4.1. Em todas as simulações, consideramos a transmissão de 10^6 bits de informação, codificados pelo código convolucional sistemático recursivo de taxa 1/2 e geradores $(7, 5)$. Os bits codificados foram mapeados para os símbolos do alfabeto $\{\pm 1\}$ e entrelaçados aleatoriamente antes da transmissão. A equalização foi feita por diferentes equalizadores SISO e a decodificação foi realizada pelo algoritmo Log-MAP.

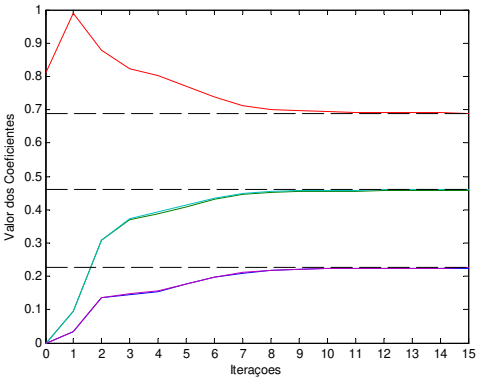
Por conveniência, designaremos o algoritmo *Fast Kalman* simplesmente por FK e o filtro de Kalman modificado por MKF. Além disso, conforme já explicado, a entrada do estimador de canal é produzida a partir da LLR proveniente do decodificador através de uma decisão suave, calculada de acordo com a equação (3.9), ou através de uma decisão abrupta, computada como $\text{sign}\left(L_{Tk}^{D}\right)$. Assim, quando decisões suaves forem utilizadas para alimentar o estimador, acrescentaremos a letra S na frente do nome do algoritmo. Já a letra H é acrescentada ao se alimentar o estimador com decisões abruptas. Desta forma, o algoritmo denominado SFK, por exemplo, representa o *Fast Kalman* alimentado por decisões suaves enquanto que HFK indica o mesmo algoritmo, mas alimentado por decisões abruptas.

O primeiro cenário de simulação consistiu na transmissão de blocos de 1024 bits de informação através do canal $0,227 + 0,46z^{-1} + 0,688z^{-2} + 0,46z^{-3} + 0,227z^{-4}$, cujas características já foram apresentadas na Fig. 3.8. A Fig. 4.2 apresenta os resultados obtidos, após a 20ª iteração, com o equalizador BCJR e alguns algoritmos de estimação de canal. A estimativa inicial, $\hat{\mathbf{h}}_{ini} = \begin{bmatrix} 0 & 0 & \hat{\sigma}_{n,ini} & 0 & 0 \end{bmatrix}$, com $\hat{\sigma}_{n,ini}$ dado por (4.7), considera que a localização do maior coeficiente do canal é conhecida. Como podemos ver na Fig. 4.2(a), os três receptores cegos têm praticamente o mesmo desempenho do receptor com perfeito conhecimento do canal e da variância do ruído para E_b/N_0 maiores que 6 dB. Também fica claro que, para E_b/N_0 menores que 6 dB, o SFTF apresenta taxas de erro mais baixas que o SLMS e o SMKF. Por exemplo, para uma taxa de erro de bit de 10^{-3}, o SFTF tem uma perda de cerca de $0,3$ dB em relação ao receptor com conhecimento do canal, enquanto que os outros dois algoritmos estão a aproximadamente $0,7$ dB. Já as Figs. 4.2(b) e 4.2(c) mostram os valores estimados para os coeficientes do canal e variância do ruído a $E_b/N_0 = 5$ dB

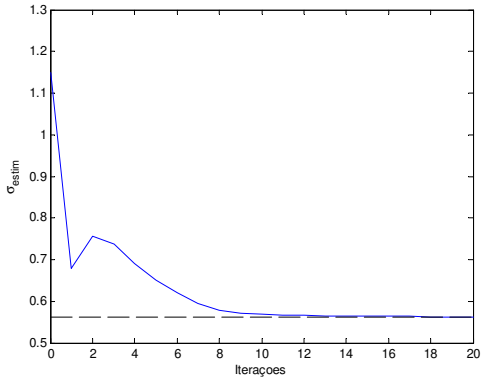
para o SFTF. Vemos que, a partir da 8ª iteração, os coeficientes do canal e o desvio padrão do ruído já são estimados com um erro de aproximação desprezível. Vale notar que a iteração 0 corresponde à estimativa inicial do canal.



(a) Curva de Erro



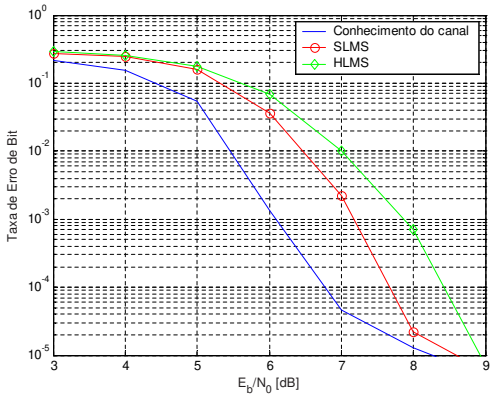
(b) Coeficientes estimados a $E_b/N_0 = 5$ dB com o SFTF



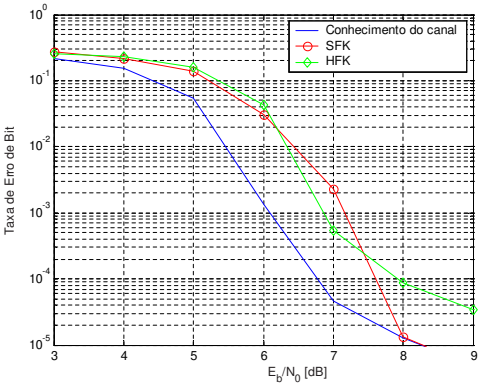
(c) Desvio padrão estimado a $E_b/N_0 = 5$ dB com o SFTF

Figura 4.2: Receptor cego utilizando equalizador MAP e diferentes algoritmos de estimação para o canal $[0,227\ 0,46\ 0,688\ 0,46\ 0,227]$.

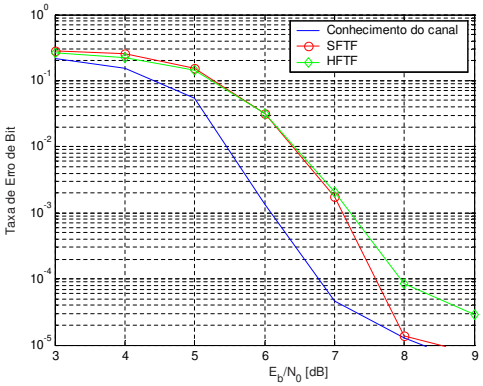
Já a Fig. 4.3 mostra as curvas de erro obtidas por receptores iterativos empregando o equalizador SFE, com $M_1 = 9$ e $M_2 = 4$, e os vários algoritmos de estimação para o mesmo canal e o mesmo número de iterações considerados anteriormente.



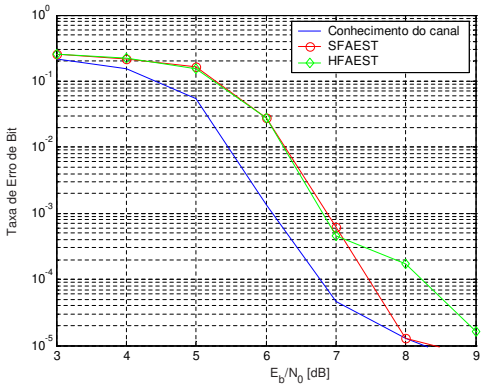
(a) LMS



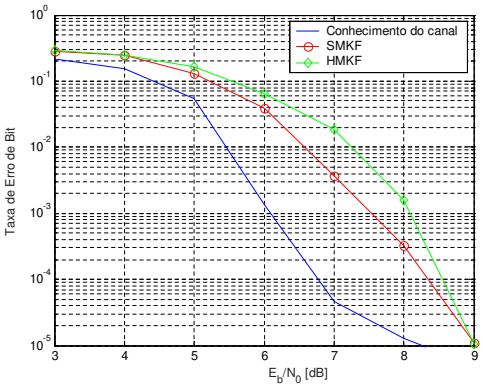
(b) FK



(c) FTF



(d) FAEST

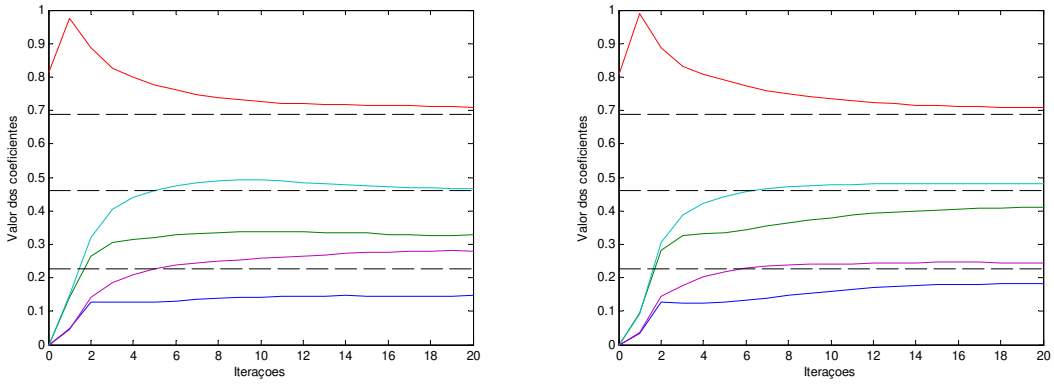


(e) MKF

Figura 4.3: Receptor cego com SFE e vários algoritmos de estimação para o canal $[0,227\ 0,46\ 0,688\ 0,46\ 0,227]$.

Observando as figuras, concluímos que os algoritmos rápidos e o LMS, quando alimentados com decisões suaves, apresentam aproximadamente os mesmos resultados e, a partir de 8 dB, se comportam como o equalizador turbo com perfeito conhecimento do canal. Os algoritmos rápidos, no entanto, têm desempenho superior ao do LMS quando alimentados por decisões abruptas. Em ambos os casos, o filtro de Kalman modificado teve desempenho ligeiramente inferior ao dos demais, embora tenha uma complexidade computacional maior. Como era de esperar, os algoritmos rápidos, por serem equivalentes, têm igual desempenho.

É interessante notar que as estimativas do canal produzidas pelo LMS não são tão boas quanto aquelas geradas pelos algoritmos rápidos, embora todos eles tenham curvas de taxa de erro de bit semelhantes. Isto fica claro na Fig. 4.4, que mostra a evolução das estimativas dos coeficientes do canal com as iterações.

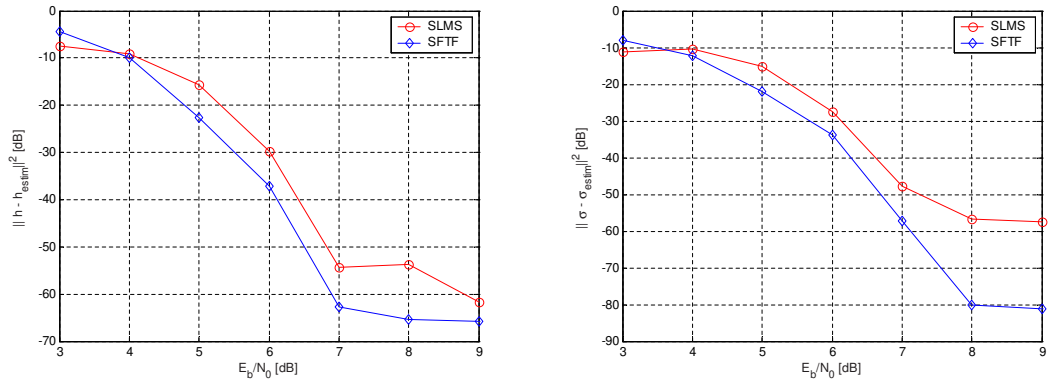


(a) SLMS (b) SFTF

Figura 4.4: Evolução das estimativas do canal para $E_b/N_0 = 5$ dB.

Para verificar que o SFTF realmente produz melhores estimativas que o SLMS traçamos, na Fig. 4.5, os erros de estimação dos coeficientes do canal e do desvio padrão do ruído para os dois algoritmos. A partir destas figuras, vemos claramente a superioridade do SFTF em relação ao SLMS, principalmente na estimação de σ. Isto ocorre pois o cálculo de $\hat{\sigma}$ depende das estimativas do canal, como mostrado em (4.6). Logo, estimativas mais precisas dos coeficientes levam a uma estimativa

mais refinada de σ.



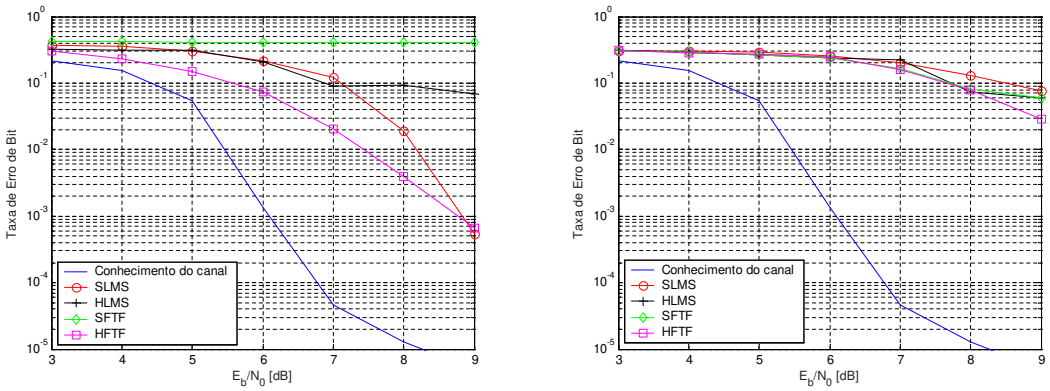
(a) Coeficientes (b) Desvio padrão

Figura 4.5: Erros de estimação para o canal $[0,227\ 0,46\ 0,688\ 0,46\ 0,227]$.

O fato do LMS ter desempenho equivalente ao dos outros algoritmos, mesmo com estimativas menos precisas do canal, pode ser explicado pela presença do código corretor de erros. Erros de estimação nos coeficientes do canal e na variância do ruído podem ser vistos como um ruído aditivo extra corrompendo o sinal recebido. Conseqüentemente, se os erros de estimação forem pequenos e o código for robusto o suficiente para suportar este "ruído" adicional, o desempenho do sistema não sofrerá uma degradação significativa.

Os resultados apresentados até este momento consideram que a posição do coeficiente de maior energia do canal é conhecida, sendo exatamente este o único coeficiente de valor não-nulo em $\hat{h}_{ini}$. Como este conhecimento nem sempre está disponível, procuramos avaliar o impacto de estimativas iniciais diferentes no desempenho do estimador turbo. Na Fig. 4.6, apresentamos os resultados obtidos com o SFE em conjunto com o LMS e o FTF para duas inicializações distintas.

Podemos notar na Fig. 4.6(a) que o desempenho do receptor com o SLMS supera o do receptor com o HLMS para E_b/N_0 maiores que 7 dB. Já o SFTF tem um desempenho muito pior que o do HFTF, que foi o algoritmo que mais se aproximou da curva do equalizador turbo com conhecimento do canal para E_b/N_0 inferiores a



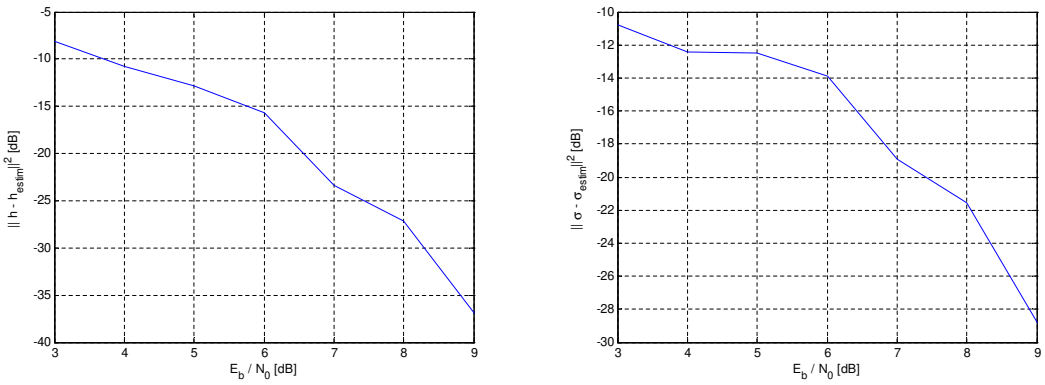
(a) $\hat{\mathbf{h}}_{ini} = (0\ \hat{\sigma}_{n,ini}\ 0\ 0\ 0)$ (b) $\hat{\mathbf{h}}_{ini} = (\hat{\sigma}_{n,ini}\ 0\ 0\ 0\ 0)$

Figura 4.6: Receptor cego com SFE e diferentes inicializações para o canal $[0,227\ 0,46\ 0,688\ 0,46\ 0,227]$.

9 dB. Para a inicialização usada na Fig. 4.6(b), porém, os quatro receptores apresentaram igual desempenho. Fica evidente, portanto, que a inicialização do canal tem um papel fundamental no desempenho dos equalizadores turbo cegos. Quanto mais distante da realidade estiver a estimativa inicial do canal, menos confiáveis serão as decisões produzidas pelo equalizador e pelo decodificador e, conseqüentemente, piores serão as novas estimativas do canal. As Figs. 4.7(a) e 4.7(b) apresentam os erro de estimação dos coeficientes do canal e da variância do ruído para o receptor cego usando o SFE em conjunto com o HFTF para a inicialização da Fig. 4.6(b).

Além disso, vemos que, para os algoritmos rápidos, a alimentação com decisões suaves nem sempre conduz a uma melhora de desempenho. Estudos mais aprofundados devem ser realizados para tentar explicar por que isso ocorre, bem como indicar quando utilizar cada tipo de decisão.

No segundo cenário de simulação, blocos de 2048 bits codificados foram transmitidos pelo canal $0,6491 + 0,6296z^{-1} + 0,3246z^{-2} + 0,1493z^{-3} + 0,2337z^{-4}$, cujos dois primeiros coeficientes têm valores bastante próximos. No receptor, o SFE empregado tinha $M_1 = 10$ e $M_2 = 5$. Além disso, foram realizadas 20 iterações turbo. A Fig. 4.8 mostra os resultados alcançados com $\hat{\mathbf{h}}_{ini} = \begin{bmatrix} \hat{\sigma}_{n,ini} & 0 & 0 & 0 & 0 \end{bmatrix}$.

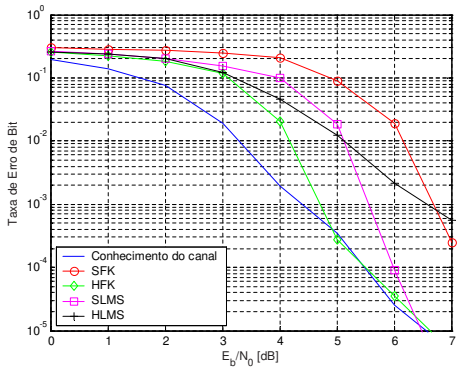


(a) Coeficientes (b) Desvio padrão

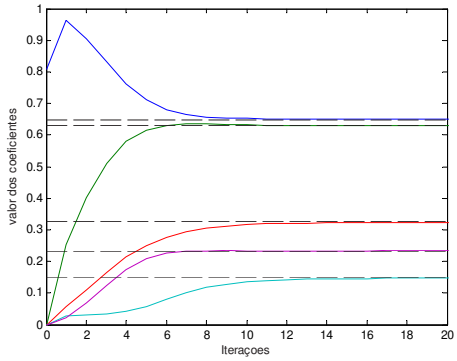
Figura 4.7: Erros de estimação para inicialização $\hat{\mathbf{h}}_{ini} = (\hat{\sigma}_{n,ini}\ 0\ 0\ 0\ 0)$.

Aqui também, o algoritmo rápido alimentado com decisões abruptas é superior aos demais, atingindo a curva do equalizador turbo com perfeito conhecimento do canal a 5 dB. O SLMS, por sua vez, só atinge esta curva em 6,5 dB. Para uma taxa de erro de bit de 10^{-3}, o HFK tem uma perda de cerca de 0,3 dB em relação ao receptor com conhecimento do canal, enquanto que o SLMS está a 0,8 dB do HFK. Já o SFK e o HLMS estão a aproximadamente 1 dB do SLMS. As Figs. 4.8(b) e 4.8(c) apresentam a evolução das estimativas dos coeficientes do canal para $E_b/N_0 = 5$ dB utilizando, respectivamente, o HFK e o SFK. Como vemos, o HFK consegue uma excelente aproximação dos coeficientes já na 10^{a} iteração, identificando o canal com um erro desprezível. Já o SFK não consegue fazer uma boa estimação do canal, convergindo provavelmente para um mínimo local, o que explicaria o fraco desempenho para esta relação sinal-ruído. Como dito anteriormente, um estudo mais aprofundado sobre a convergência dos estimadores turbo cegos usando algoritmos de mínimos quadrados rápidos alimentados por decisões suaves e abruptas deve ser conduzido para tentar elucidar este problema.

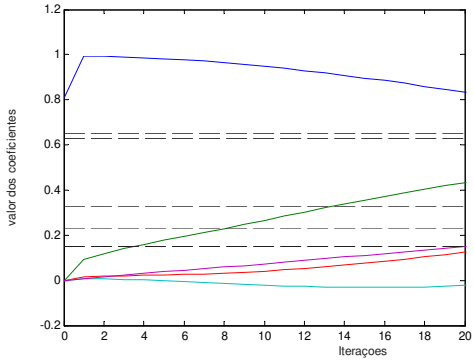
Até este ponto, priorizamos a análise de estimadores turbo de baixa complexidade utilizando o SFE. Porém, para canais curtos, os algoritmos de complexidade exponencial não apresentam um custo computacional muito maior que os algorit-



(a) Curva de Erro



(b) Canal estimado com o HFK para $E_b/N_0 = 5$ dB

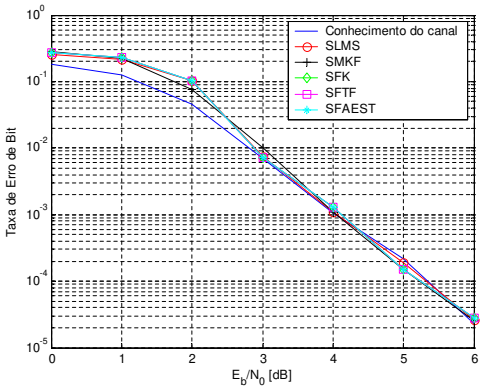


(c) Canal estimado com o SFK para $E_b/N_0 = 5$ dB

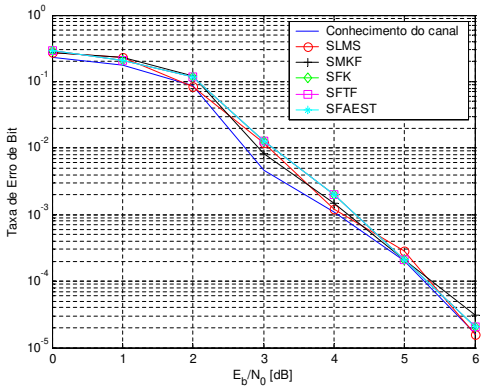
Figura 4.8: Desempenho do receptor cego com o SFE para o canal $[0,6491\ 0,6296\ 0,3246\ 0,1493\ 0,2337]$.

mos de complexidade reduzida. Logo, nesta última etapa de simulações, vamos avaliar o desempenho de receptores cegos usando vários equalizadores e vários algoritmos de estimação de canal. Para tanto, o último cenário de simulação consistiu na transmissão de 1024 bits codificados pelo canal $0,5 + 0,71z^{-1} + 0,5z^{-2}$, cujas características foram apresentadas na Fig. 3.15. A estimativa inicial do canal, $\hat{\mathbf{h}}_{ini} = \begin{bmatrix} 0 & \hat{\sigma}_{n,ini} & 0 \end{bmatrix}$, considera que a posição do coeficiente de maior energia do

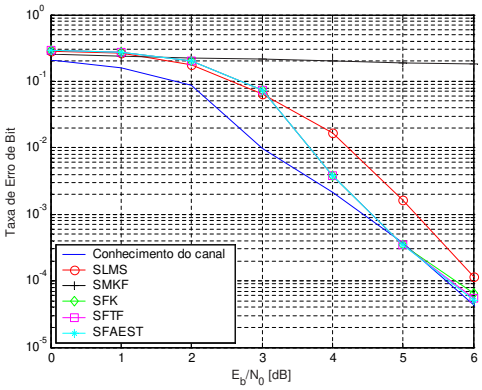
canal é conhecida. A Fig. 4.9 mostra os resultados obtidos com o BCJR, o Max-Log-MAP e o SFE ($M_1 = 6$ e $M_2 = 3$) após a 10^{a} iteração turbo.



(a) BCJR



(b) Max-Log-MAP



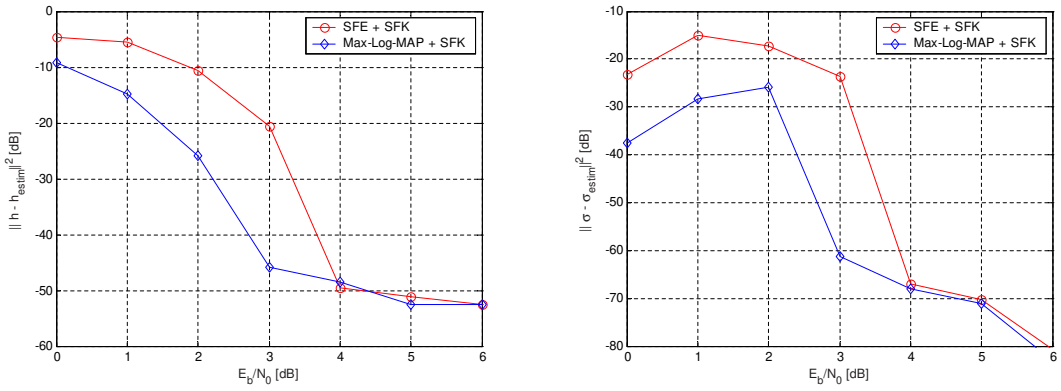
(c) SFE

Figura 4.9: Desempenho para o canal $[0,5\ 0,71\ 0,5]$.

Observamos nestas figuras que todos os algoritmos rápidos atingem a curva do equalizador turbo com perfeito conhecimento do canal para os três equalizadores. Para o BCJR, isto ocorre a $E_b/N_0 = 3$ dB e, para o SFE, a 5 dB. Já com o Max-Log-MAP, os receptores cegos se comportam como o equalizador com conhecimento do canal para todas as relações sinal-ruído simuladas. O SLMS, por sua vez, se comporta exatamente como os algoritmos rápidos somente quando o receptor em-

prega o BCJR e o Max-Log-MAP. Com o SFE, o SLMS apresenta uma perda de cerca de 0,6 dB em relação aos algoritmos rápidos para uma taxa de erro de bit de 10^{-3}. Algo semelhante ocorre com o receptor cego usando o SMKF: em conjunto com os equalizadores de complexidade exponencial, o estimador turbo com SMKF tem desempenho igual aos dos sistemas com os outros algoritmos de estimação. Já em conjunto com o SFE, o SMKF não consegue calcular estimativas do canal que permitam melhorar o desempenho com as iterações. Mais uma vez, portanto, vemos claramente as vantagens em se empregar os algoritmos rápidos para estimar o canal, independente do equalizador SISO escolhido.

Ainda comparando o desempenho dos estimadores turbo com o Max-Log-MAP e com o SFE, apresentamos na Fig. 4.10 os erros na estimação dos coeficientes do canal e no desvio padrão do ruído obtidos com o SFK. Vemos, nestas figuras, que os dois receptores calculam estimativas praticamente idênticas do canal e do desvio padrão do ruído a partir de 4 dB. Para relações sinal-ruído mais baixas, o receptor com o Max-Log-MAP apresenta um desempenho bastante superior, o que é comprovado pelas curvas da Fig. 4.9.

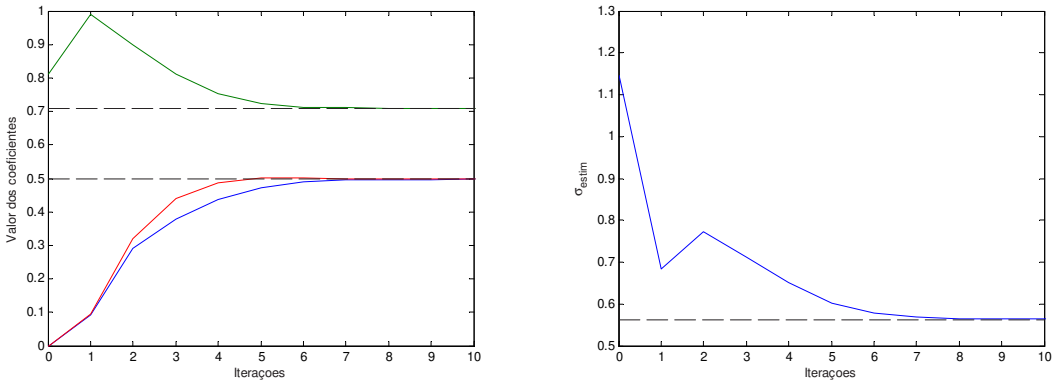


(a) Coeficientes (b) Desvio padrão do ruído

Figura 4.10: Erros de estimação para o canal $[0,5\ 0,71\ 0,5]$.

Para o estimador turbo utilizando o SFE e o SFK mostramos, na Fig. 4.11, as estimativas do canal para uma relação sinal-ruído de 5 dB. Como podemos perceber,

a partir da 7ª iteração o receptor cego já estima os coeficientes do canal e o desvio padrão do ruído com um erro de aproximação desprezível.

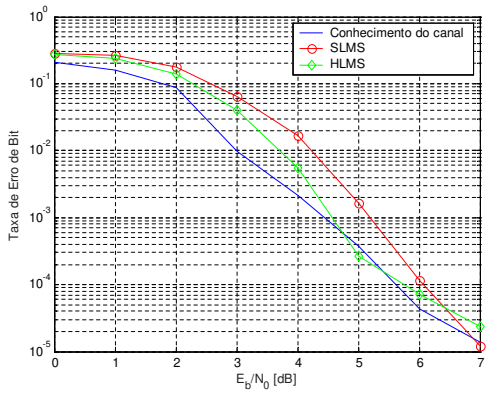


(a) Coeficientes (b) Desvio padrão do ruído

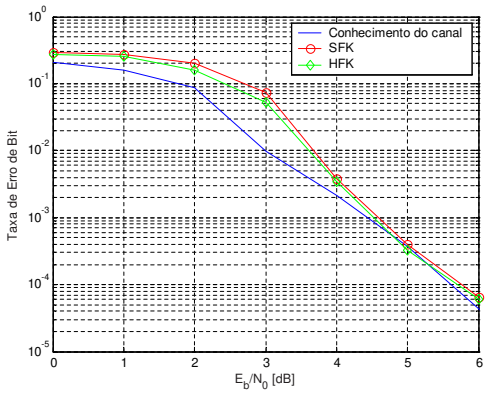
Figura 4.11: Estimativas do canal $[0,5\ 0,71\ 0,5]$ usando SFE e SFK para $E_b/N_0 = 5$ dB.

Em todas as simulações feitas até agora para o canal $0,5+0,71z^{-1}+0,5z^{-2}$, os algoritmos de estimação foram alimentados com estimativas dos símbolos transmitidos obtidas a partir de decisões suaves sobre as LLR's produzidas pelo decodificador. A título de comparação, traçamos na Fig. 4.12 as curvas de taxa de erro de bit para receptores cegos usando o SFE em conjunto com o LMS, o FK e o MKF quando estes são alimentados pelos dois tipos de decisão. Vemos que, para esse canal, o SFK tem o mesmo desempenho do HFK. Já o HLMS apresenta taxas de erro menores que o SLMS, o mesmo ocorrendo entre o HMKF e o SMKF. Estes resultados reforçam a idéia de que um estudo mais aprofundado deve ser feito para determinar em quais situações a utilização de decisões abruptas permite melhorar as estimativas do canal.

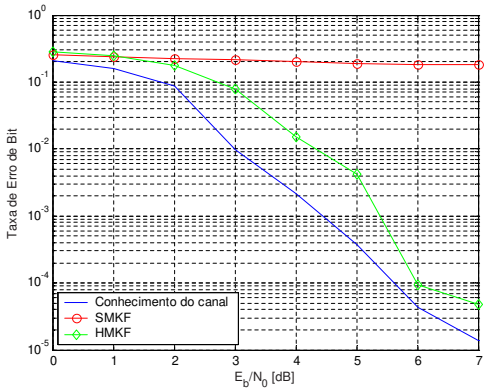
Por fim, analisamos a influência da inicialização no desempenho do estimador turbo para alguns equalizadores e para canal $0,5+0,71z^{-1}+0,5z^{-2}$. Considerando $\hat{\mathbf{h}}_{ini} = \begin{bmatrix} \hat{\sigma}_{n,ini} & 0 & 0 \end{bmatrix}$ notamos, na Fig. 4.13, que o equalizador BCJR é mais sensível às condições iniciais que o SFE e o Max-Log-MAP. Com o BCJR, apenas o receptor usando o SLMS atingiu o desempenho do sistema com conhecimento do



(a) LMS



(b) FK

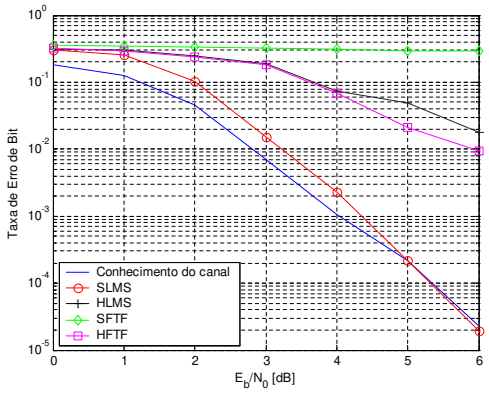


(c) MKF

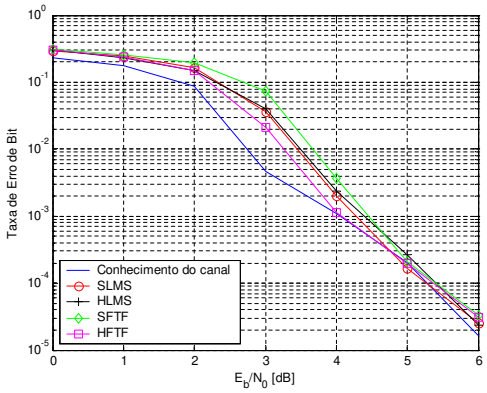
Figura 4.12: Comparação entre decisões suaves e abruptas para o canal $[0,5\ 0,71\ 0,5]$.

canal. Com o SFE, no entanto, esta curva foi atingida pelo SLMS e pelo HFTF. O Max-Log-MAP parece ser o mais robusto dos equalizadores pois todos os receptores cegos que o empregaram se comportam, a partir de 5 dB, como se conhecessem precisamente o canal. Uma possível explicação para esta robustez é dada em [47], onde é mostrado que a saída do equalizador Max-Log-MAP independe da variância do ruído, diferentemente do Log-MAP e do BCJR. Assim, a saída do Max-Log-MAP conteria, além do ruído do canal, apenas o "ruído" acrescentado pelo erro de esti-

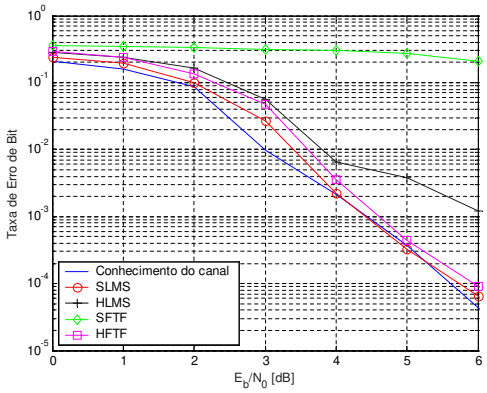
mação dos coeficientes. Já os sinais nas saídas do BCJR e do SFE teriam ainda o ruído produzido pelo erro de estimação da variância. Então, como os sinais na saída do Max-Log-MAP têm uma relação sinal-ruído maior, a decodificação produz estimativas mais confiáveis dos símbolos transmitidos e, conseqüentemente, a estimação de canal é beneficiada.



(a) BCJR



(b) Max-Log-MAP



(c) SFE

Figura 4.13: Inicialização: $\hat{\mathbf{h}}_{ini} = (\hat{\sigma}_{n,ini}\ 0\ 0)$.

A partir dos resultados apresentados nesta seção, vemos que é possível implementar bons receptores iterativos cegos de baixa complexidade computacional utilizando em conjunto o SFE e algum algoritmo rápido, como o FK.

4.5 Conclusão

Neste capítulo, apresentamos possíveis esquemas para realizar a equalização turbo quando o canal não é conhecido pelo receptor. Nestas abordagens, algoritmos de estimação de canal são incluídos na malha de realimentação do receptor para se beneficiarem do código corretor de erros, sendo equalização, decodificação e estimação de canal realizadas conjuntamente através de um procedimento iterativo. Soluções de baixa complexidade computacional, como o algoritmo LMS, foram estudadas. Além disso, propusemos a utilização dos algoritmos de mínimos quadrados rápidos, como o *Fast Kalman*, o FTF e o FAEST para estimar o canal e mostramos, através de simulações, as vantagens em se empregar tais algoritmos. Por fim, concluímos que bons receptores iterativos cegos e de baixa complexidade podem ser obtidos com o uso conjunto do SFE e de algum algoritmo rápido.

5

Equalização e Decodificação Iterativas Usando Filtros Fuzzy

Neste capítulo, vamos propor um equalizador iterativo, inspirado no da Fig. 3.3, que utiliza um filtro *fuzzy*. Esta nova estrutura procura mitigar os efeitos da IIS e do ruído impulsivo em canais de comunicação digital de banda larga, como os canais dos sistemas celulares de 3ª e 4ª gerações, os canais de radiodifusão de TV digital ou como os canais PLC (*Power Line Communications*). Este último sistema utiliza a rede de distribuição de energia elétrica e, recentemente, tem despertado a atenção de muitos pesquisadores pois constitui uma alternativa de baixo custo à transmissão de dados a taxas elevadas.

Várias características dos canais PLC dificultam significativamente sua utilização em larga escala. Dentre elas, podemos citar: atenuações variantes no tempo e

dependentes da freqüência de transmissão e da distância entre transmissor e receptor, que podem chegar a até 60 dB [48]; reflexões causadas por descasamentos de impedância, o que ocasiona a propagação dos sinais por múltiplos percursos; e forte presença de ruído impulsivo, cuja densidade espectral de potência pode ultrapassar em 50 dB a do ruído de fundo [48]. Este ruído impulsivo é gerado por mudanças rápidas e aleatórias nas condições do canal, ocasionadas, por exemplo, pela conexão e desconexão de cargas lineares e não-lineares.

Como a função densidade de probabilidade do ruído impulsivo não é gaussiana, equalizadores lineares, baseados no critério MMSE, não apresentam bom desempenho. Logo, equalizadores não-lineares são normalmente escolhidos. Dentre as várias classes de estruturas não-lineares, os filtros *fuzzy* aparecem como bons candidatos para a minimização do ruído impulsivo porque são construídos para trabalhar com incertezas e podem levar a uma solução quase-ótima, em termos de taxa de erro de bit, com apenas algumas modificações em sua estrutura. Outra alternativa seria empregar o equalizador BCJR, descrito na subseção 3.1.1. Para tanto, porém, o cálculo de $p\left(r_k|s', s\right)$ em (2.6) deve ser modificado de maneira a considerar a distribuição do ruído, o que não é tão simples neste caso.

5.1 Equalizadores *Fuzzy*

Sistemas *fuzzy* são sistemas não-lineares capazes de inferir complexas relações entre as variáveis de entrada e saída de um processo. Esse mapeamento entre entrada e saída é feito a partir de um conjunto de regras construído com informações numéricas e/ou lingüísticas. Como exemplo, suponha que se deseje controlar a temperatura de um ambiente. Algumas das regras que poderiam ser criadas são:

Se temperatura é *alta* **então** ligue ar condicionado
Se temperatura é *baixa* **então** desligue ar condicionado
Se temperatura é *baixa* **e** ar condicionado desligado **então** ligue aquecedor

ou ainda

Se temperatura $> 25^{o}$C **então** ligue ar condicionado
Se temperatura $< 19^{o}$C **então** desligue ar condicionado

O primeiro conjunto de regras utiliza apenas informações lingüísticas, enquanto que o segundo também utiliza informações numéricas. De uma forma genérica, podemos escrever uma regra como

Se antecedente 1 **e/ou** antecedente 2 **e/ou** ... antecedente M **então** conseqüente 1 **e/ou** ... conseqüente N

Cada antecedente e conseqüente é representado por uma função denominada *função de pertinência*. Como o próprio nome sugere, estas funções indicam o grau de pertinência do valor de uma variável em relação ao antecedente/conseqüente.

Os sistemas *fuzzy* trabalham internamente com conjuntos *fuzzy*, que podem ser vistos como conjuntos tradicionais onde as fronteiras não estão perfeitamente definidas. Deste modo, é geralmente necessário "*fuzzificar*" os valores de entrada, isto é, transformar valores pontuais em conjuntos *fuzzy*. Da mesma maneira, precisamos normalmente transformar os conjuntos de saída em valores pontuais, ou seja, "*defuzzificar*". Existem inúmeros métodos de *fuzzificação* e *defuzzificação*, sendo a escolha destes mais um parâmetro de projeto.

Podemos considerar um sistema *fuzzy* como composto por 4 blocos fundamentais, apresentados na Fig. 5.1. Conforme vemos nesta figura, a base de regras é formada pelo conjunto de regras que será usado pelo mecanismo de inferência para fazer o mapeamento entre os conjuntos de entrada e de saída.

A estrutura matemática de um sistema *fuzzy* depende, principalmente, da escolha das estratégias de *fuzzificação* e *defuzzificação*, do tipo de função de pertinência utilizada e de como o mecanismo de inferência deve combinar as regras. Considerando apenas as informações numéricas, um filtro *fuzzy* pode ser descrito como uma generalização das redes neurais com funções de ativação de base radial

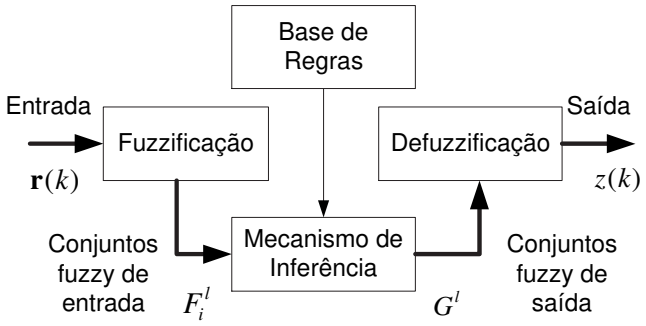


Figura 5.1: Esquema de um sistema *fuzzy*.

(RBF) [49] normalizadas. Conseqüentemente, como mostrado em [49], um filtro *fuzzy* operando como um equalizador transversal permite atingir a solução ótima, em termos de mínima probabilidade de erro.

Modelando o canal PLC como um canal linear, dispersivo e de faixa limitada, temos que sua saída é dada por

$$r\left(k\right)=\sum_{n=0}^{\mu-1}h\left(n\right)x\left(k-n\right)+\nu\left(k\right)=\tilde{r}\left(k\right)+\nu\left(k\right), \tag{5.1}$$

onde $x\left(k\right)$ é o símbolo transmitido no instante k, h é a resposta ao impulso do canal, $\tilde{r}\left(k\right)$ é o sinal sem ruído na saída do canal e ν corresponde à soma de todos os tipos de ruído, incluindo o ruído gaussiano de fundo e o ruído impulsivo.

A saída do filtro *fuzzy*, que realiza então a estimação bayesiana dos símbolos transmitidos, considerando-se modulação binária, *fuzzificação singleton*, funções de pertinência gaussianas, composição max-produto e *defuzzificação* pelo método das alturas [50], é calculada por

$$f(\mathbf{r}(k))=\frac{1}{N_s}(2\pi\sigma_n^2)^{-m}\frac{\left\{\sum\limits_{\tilde{\mathbf{r}}\in\mathbf{C}_d^+}\exp\left(-\frac{\|\mathbf{r}(k)-\tilde{\mathbf{r}}\|^2}{2\sigma_n^2}\right)-\sum\limits_{\tilde{\mathbf{r}}\in\mathbf{C}_d^-}\exp\left(-\frac{\|\mathbf{r}(k)-\tilde{\mathbf{r}}\|^2}{2\sigma_n^2}\right)\right\}}{\sum\limits_{\tilde{\mathbf{r}}\in\mathbf{C}_d}\exp\left(-\frac{\|\mathbf{r}(k)-\tilde{\mathbf{r}}\|^2}{2\sigma_n^2}\right)}. \tag{5.2}$$

Em (5.2), N_s é o número de possíveis valores de saídas do canal sem ruído, σ_n^2 é a variância do ruido aditivo branco e gaussiano de fundo, m é o número de entradas

do equalizador, $\mathbf{C}_d^+ = \{\tilde{\mathbf{r}}(k) | x(k-d) = +1\}$ e $\mathbf{C}_d^- = \{\tilde{\mathbf{r}}(k) | x(k-d) = -1\}$ são, respectivamente, os conjuntos dos valores de saída do canal para símbolos transmitidos iguais a $+1$ e -1, $\mathbf{C}_d = \mathbf{C}_d^+ \cup \mathbf{C}_d^-$, d é o atraso de equalização e $\|\cdot\|$ representa a norma de um vetor.

Como podemos notar em (5.2), a saída do equalizador *fuzzy* é expressa como uma diferença entre termos correspondentes aos dois possíveis símbolos transmitidos. Isto pode ser encarado como um tipo de informação suave, que indica a confiabilidade da decisão. Este fato será muito importante na seção seguinte, onde iremos propor um equalizador *fuzzy* baseado no princípio turbo.

Uma desvantagem deste filtro *fuzzy* é a complexidade computacional, que aumenta exponencialmente com o comprimento do canal. Para reduzir a complexidade, utilizamos um filtro *fuzzy* com realimentação de decisão, inspirado em [49, 51]. A diminuição da carga computacional é possível pois, conforme mostrado em [51], ao considerar que os símbolos estimados, $\hat{x}$, estão corretos, a realimentação de decisão diminui o número de valores pertencentes aos conjuntos $\mathbf{C}_\mathbf{d}^+$ e $\mathbf{C}_d^-$. Na Fig 5.2 é apresentada essa estrutura realimentada, onde Q é a função de decisão. Como o canal PLC varia com o tempo, este equalizador foi implementado adaptativamente, usando o algoritmo de *Back-propagation* [3]. Para isto, é necessária uma seqüência de treinamento periódica para ajustar os parâmetros do sistema *fuzzy*, tais como os centros e dispersões dos conjuntos *fuzzy* de entrada.

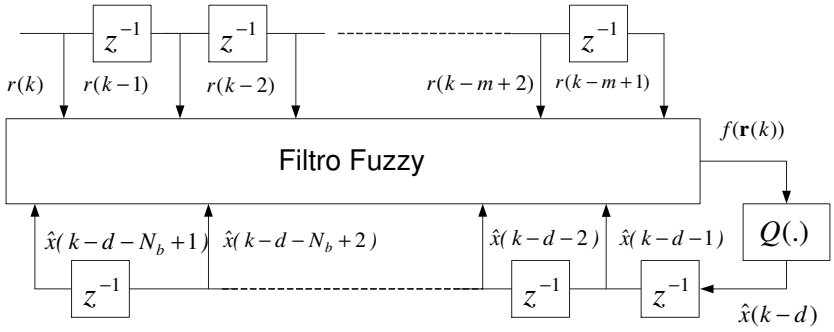


Figura 5.2: Equalizador *fuzzy* com realimentação de decisão.

5.2 Equalizador *Fuzzy* Iterativo

Conforme mencionado na seção anterior, o filtro *fuzzy* da Fig. 5.2 pode, dependendo da escolha das funções de pertinência e dos métodos de inferência, *fuzzificação* e *defuzzificação*, se tornar equivalente ao equalizador ótimo que minimiza a probabilidade de erro. Também mostramos que as saídas deste filtro, embora não sejam LLR's, expressam a confiabilidade do sinal equalizado. Portanto, podemos empregá-lo num esquema iterativo semelhante ao de um equalizador turbo. A Fig. 5.3 mostra este novo equalizador.

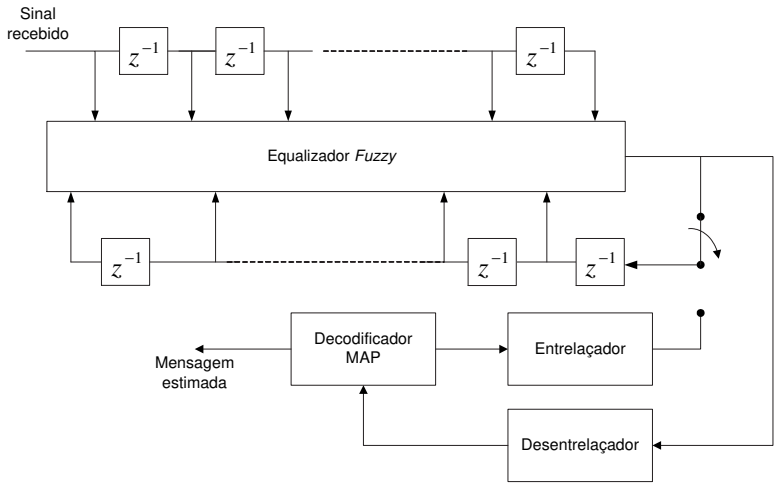


Figura 5.3: Equalizador *fuzzy* iterativo.

Vemos, na Fig. 5.3, que as entradas do ramo de realimentação do equalizador *fuzzy* são alimentadas por estimativas dos símbolos transmitidos obtidas a partir das saídas do próprio equalizador ou a partir das saídas do decodificador. Durante a primeira iteração turbo, não há nenhuma informação *a priori* disponível e, portanto, utilizamos a saída suave do filtro *fuzzy* para a realimentação. Esta configuração corresponde a um equalizador *fuzzy* DFE convencional. Como a partir da segunda iteração estimativas mais confiáveis dos símbolos transmitidos podem ser calculadas com as informações *a priori* fornecidas pelo decodificador, usando (3.9), empregamos essas estimativas para realimentar o filtro.

5.3 Simulações

Para avaliar o desempenho do equalizador *fuzzy* iterativo, foi simulada a transmissão de bits de informação codificados por um código convolucional sistemático recursivo de taxa $R = 1/2$ e geradores $(23, 35)$. Esses bits codificados foram então modulados em BPSK e permutados por um entrelaçador aleatório de tamanho igual a 10000 antes de serem transmitidos.

Foram considerados dois canais PLC para transmissões entre 1 e 2 MHz, cada um deles com 128 coeficientes. As características desses canais podem ser observadas nas Figs. 5.4 e 5.5. Adicionalmente à degradação imposta por estes canais, foi considerado um ruído de fundo aditivo, branco e gaussiano além de forte presença de ruído impulsivo, conforme indicado na Fig. 5.6.

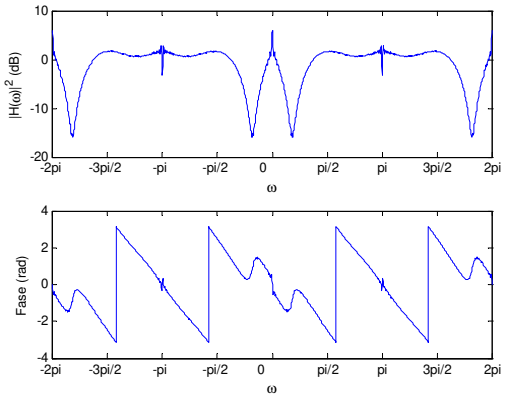


Figura 5.4: Resposta em freqüência do canal 1.

O equalizador *fuzzy* foi testado com 20 entradas, 10 das quais destinadas à realimentação. Além disso, cada entrada possuía 20 funções de pertinência gaussianas. A decodificação foi realizada pelo algoritmo BCJR e, para cada bloco de dados recebidos, foram feitas 10 iterações turbo. As curvas resultantes das simulações estão nas Figs. 5.7 e 5.8. Cada iteração é designada pelo símbolo # em tais curvas. Por motivo de comparação, também foram traçadas as curvas para canais sem IIS e sem ruído impulsivo.

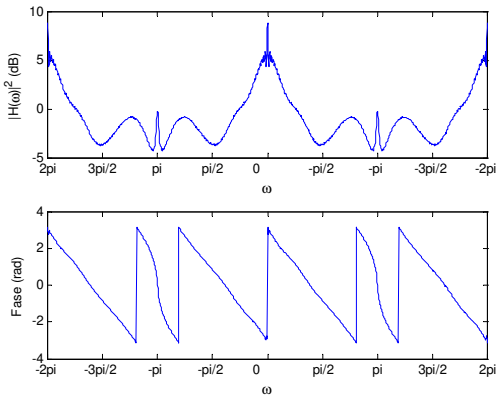


Figura 5.5: Resposta em freqüência do canal 2.

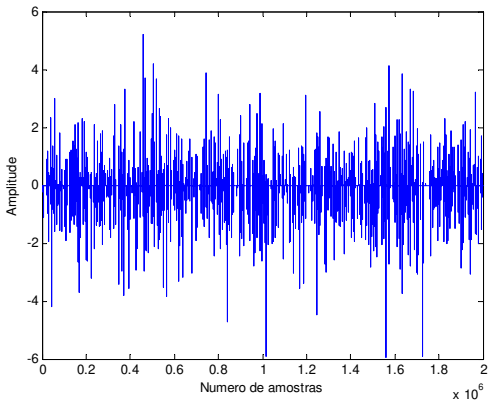


Figura 5.6: Amostras de ruído impulsivo.

Em ambas as figuras, o desempenho obtido na primeira iteração é idêntico àquele conseguido pelo equalizador *fuzzy* e pelo decodificador em um receptor não-iterativo. Para o canal 1, por exemplo, temos um ganho de aproximadamente $3,7$ dB da terceira para a décima iterações considerando uma taxa de erro de bit de 10^{-2}. Já para uma taxa de erro de 10^{-3}, o ganho apresentado entre a quinta e sétima iterações é de cerca de 1 dB. Isto mostra o benefício alcançado pelas iterações.

Embora os resultados demonstrem a superioridade do processo iterativo em relação ao esquema não-iterativo, vemos claramente que o desempenho do equalizador

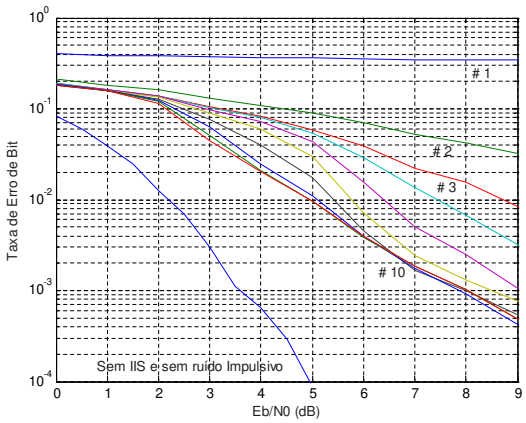


Figura 5.7: Curvas de erro para o canal 1.

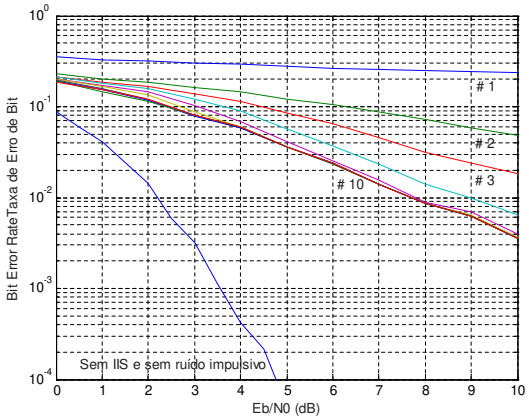


Figura 5.8: Curvas de erro para o canal 2.

fuzzy proposto não se aproxima da curva do canal AWGN, como ocorre com os equalizadores turbo descritos no capítulo 3. Uma possível explicação para isto reside no fato de que a estrutura proposta neste capítulo utiliza a informação *a priori* para cancelar apenas a parte causal da IIS. Conseqüentemente, o sinal na saída do equalizador continua corrompido pela parcela não-causal da IIS. Uma idéia para tentar solucionar este problema é usar filtros *fuzzy* numa estrutura de cancelamento de interferência, como a da Fig. 3.5.

Um outro fator que talvez degrade o desempenho do equalizador iterativo proposto é que ele não produz LLR's, mas uma outra informação de confiabilidade, que é normalizada. Equalizadores turbo usando redes neurais RBF [52], cujo desempenho se aproxima do ótimo, têm estrutura semelhante ao equalizador *fuzzy* iterativo, porém suas saídas não são normalizadas. Um estudo mais aprofundado deve ser realizado de forma a corroborar, ou não, esta possível explicação.

5.4 Conclusão

Neste capítulo, propusemos um esquema de equalização iterativa utilizando uma classe de estruturas não-lineares denominadas filtros *fuzzy*. Mostramos como a informação *a priori* produzida pelo decodificador pode ser incorporadas em tais filtros e que as saídas geradas pelo equalizador indicam a confiabilidade do sinal equalizado. Por fim, simulamos o desempenho do equalizador proposto em canais PLC e observamos que o procedimento iterativo oferece ganhos consideráveis em relação ao receptor não-iterativo usual.

6

Conclusões e Perspectivas

Neste trabalho, propusemos e analisamos técnicas iterativas para equalização e estimação cega de canais de comunicação que se beneficiam da capacidade de correção de erros dos códigos convolucionais. Dedicamos a maior parte da dissertação ao estudo de equalizadores e estimadores turbo de baixa complexidade computacional, visando uma possível aplicação em telefonia celular de 3ª e 4ª gerações.

No capítulo 2, apresentamos alguns conceitos básicos de codificação e decodificação de códigos convolucionais, bem como de códigos turbo. Apresentamos também o algoritmo BCJR, que foi utilizado no restante da dissertação. Introduzimos o problema de equalização e descrevemos dois dos principais critérios de obtenção de equalizadores: o filtro de Wiener e o estimador de seqüência de máxima verossimilhança. Por fim, discutimos brevemente o problema de estimação de canais de comunicação e expomos dois algoritmos clássicos da teoria de filtragem adaptativa:

o algoritmo LMS e o filtro de Kalman.

Os princípios de funcionamento dos equalizadores turbo foram detalhados no capítulo 3. Neste novo paradigma, equalização e decodificação são realizadas conjuntamente através de um processo iterativo. Com isso, o equalizador pode se beneficiar do código para remover a IIS de uma maneira mais efetiva. Dentre os equalizadores de complexidade exponencial existentes, estudamos o BCJR e duas de suas versões simplificadas, o Log-MAP e o Max-Log-MAP. Já dentre as várias propostas de equalizadores SISO de baixa complexidade, como aquelas em [6–8, 25], nos concentramos no SFE, uma técnica de cancelamento de interferência que utiliza a própria saída do equalizador para obter estimativas mais confiáveis dos símbolos interferentes e que possui complexidade computacional proporcional ao número de coeficientes do equalizador. Para reduzir um pouco mais a complexidade computacional e simplificar o cálculo dos coeficientes do SFE, propusemos uma aproximação para a função $\Psi_1(\gamma)$ em (3.24) e mostramos que esta aproximação não degrada o desempenho do receptor. Através de simulações, comparamos os desempenhos dos vários equalizadores turbo e verificamos a influência do código corretor de erros, do canal e do tamanho do entrelaçador sobre o desempenho do receptor. Concluímos que o equalizador turbo usando o SFE garante uma boa relação entre desempenho e complexidade computacional. Além disso, simulações preliminares indicam que o equalizador turbo com o SFE tem um desempenho superior ao das técnicas tradicionais para canais com desvanecimento seletivo em freqüência, comum no ambiente rádio-móvel.

No capítulo 4, estudamos um possível esquema para realizar a equalização turbo quando o canal não é conhecido pelo receptor. Nesta abordagem, um algoritmo de estimação de canal é incluído na malha de realimentação do receptor para se beneficiar da redundância introduzida pelo código corretor de erros. Analisamos soluções autodidatas de baixa complexidade computacional e propusemos a utilização dos algoritmos de mínimos quadrados rápidos para estimar o canal. Através de simulações, mostramos que os estimadores turbo usando os algoritmos rápidos apresentam, geralmente, desempenho equivalente ao do equalizador turbo com perfeito

conhecimento do canal, a partir de uma determinada relação sinal-ruído. Verificamos também que as estimativas produzidas pelo algoritmos rápidos são melhores que aquelas geradas pelo LMS e que algoritmos de estimação alimentados com decisões suaves nem sempre apresentam um melhor desempenho. Por fim, vimos que o equalizador Max-Log-MAP é mais robusto que o SFE e o BCJR a erros de estimação na variância do ruído e concluímos que bons receptores iterativos cegos de baixa complexidade podem ser obtidos com o uso conjunto do SFE e de algum algoritmo rápido.

Já o capítulo 5 aborda o problema de equalização não-linear de canais com forte presença de ruído impulsivo. Após uma breve introdução aos sistemas *fuzzy*, propusemos um esquema de equalização iterativa utilizando equalizadores *fuzzy*. Mostramos que a informação *a priori* produzida pelo decodificador pode ser incorporada em tais filtros através das entradas do ramo de realimentação e que as saídas geradas pelo equalizador indicam a confiabilidade do sinal equalizado. Por fim, simulamos o desempenho do equalizador proposto em canais PLC e observamos que o procedimento iterativo oferece ganhos consideráveis em relação ao receptor *fuzzy* não-iterativo usual.

6.1 Trabalhos Futuros

Um fator fundamental para que a equalização turbo possa ser empregada em futuros equipamentos celulares é o desenvolvimento de equalizadores SISO de baixa complexidade capazes de trabalhar com modulações de vários níveis. Este assunto é abordado em [7, 25, 53], onde são propostos equalizadores lineares para modulações M-QAM e M-PSK. Nesta linha, uma próxima etapa do trabalho é generalizar o SFE para que ele possa ser empregado com outras modulações.

Além disso, os princípios que fundamentam a equalização turbo podem também ser aplicados na resolução de outros problemas comuns nas comunicações sem fio, quase sempre com desempenho superior ao de técnicas clássicas não-iterativas. Sendo o ambiente rádio-móvel compartilhado por inúmeros usuários, técnicas de

múltiplo acesso e de detecção multiusuário são essenciais para o correto encaminhamento dos dados transmitidos. Como mostrado em [15], o problema da detecção multiusuário também é passível de ser atacado por uma implementação "turbo" no receptor. Dentro deste tópico, pretendemos utilizar o SFE para realizar detecção multiusuário iterativa de baixa complexidade.

Outro problema de grande interesse prático em sistemas celulares é a utilização de arranjos de antenas, tanto na transmissão quanto na recepção. Esses arranjos procuram aumentar a capacidade de transmissão além de mitigar os efeitos do desvanecimento causado pela propagação em multipercursos utilizando estratégias de diversidade espacial. Trabalhos recentes, como aqueles em [54, 55], incorporam equalizadores turbo em arranjos de antenas em sistemas MIMO (*Multiple Input - Multiple Output*) apresentando, mais uma vez, desempenho superior ao de técnicas convencionais.

Os estimadores turbo autodidatas considerados neste trabalho consideram que o comprimento do canal é conhecido. Na prática, isto nem sempre ocorre. Se o comprimento do canal for incorretamente estimado, as seqüências nas saídas do desentrelaçador e do decodificador não terão correlação e, portanto, a mensagem estimada não corresponderá à mensagem transmitida. Logo, técnicas para determinação exata da ordem do canal devem ser pesquisadas e incorporadas aos estimadores turbo.

Também mostramos que alimentar os algoritmos rápidos com decisões suaves nem sempre conduz a uma melhora de desempenho. Conseqüentemente, estudos mais aprofundados devem ser realizados para tentar explicar por que isso ocorre, bem como indicar quando utilizar cada tipo de decisão.

Em relação ao equalizador *fuzzy* iterativo, maiores estudos devem ser feitos para verificar se o desempenho geral do receptor pode ser melhorado pela alteração de alguns dos parâmetros de projeto.

Bibliografia

[1] E. A. Lee e D. G. Messerschmit, *Digital Communication*, 2ª ed. Kluwer Academic Press, 1994.

[2] S. Lin e D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, 1983.

[3] S. Haykin, *Adaptive Filter Theory*, 3ª ed. Prentice-Hall, 1996.

[4] C. Douillard, M. Jézéquel, C. Berrou, A. Picart, P. Didier, e A. Glavieux, "Iterative Correction of Intersymbol Interference: Turbo-Equalization," *European Transactions on Telecommunications*, vol. 6, n°. 5, pags. 507–511, setembro-outubro 1995.

[5] L. R. Bahl, J. Cocke, F. Jelinek, e J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, n°. 2, pags. 284–287, março 1974.

[6] R. R. Lopes e J. R. Barry, "Soft-Output Decision-Feedback Equalization with a Priori Information," em *Proceedings IEEE Global Telecommunications Conference, GLOBECOM'03*, vol. 3, dezembro 2003, pags. 1705–1709.

[7] C. Laot, A. Glavieux, e J. Labat, "Turbo Equalization: Adaptive Equalization and Channel Decoding Jointly Optimized," *IEEE Journal on Selected Areas in Communications*, vol. 19, n°. 9, pags. 1744–1752, setembro 2001.

[8] M. Tüchler, R. Koetter, e A. C. Singer, "Turbo Equalization: Principles and New Results," *IEEE Transactions on Communications*, vol. 50, n°. 5, pags. 754–766, maio 2002.

[9] B. Vucetic e J. Yuan, *Turbo Codes – Principles and Applications.* Kluwer Academic Publishers, 2000.

[10] C. Berrou, A. Glavieux, e P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes," em *Proceedings IEEE International Conference on Communications. ICC'93*, vol. 2/3, Genebra, Suíça, 1993, pags. 1064–1070.

[11] M. D. Yacoub, *Wireless Technology: Protocols, Standards, and Techniques.* CRC Press, 2002.

[12] S. Benedetto, D. Divsalar, G. Montorsi, e F. Pollara, "Serial Concatenation of Interleaved Codes: Analysis, Design and Iterative Decoding," *IEEE Transactions on Information Theory*, vol. 44, n°. 3, pags. 909–926, maio 1998.

[13] S. Benedetto e G. Montorsi, "Unveiling Turbo Codes: Some Results on Parallel Concatenated Coding Schemes," *IEEE Transactions on Information Theory*, vol. 42, n°. 2, pags. 409–428, março 1996.

[14] J. P. Woodard e L. Hanzo, "Comparative Study of Turbo Decoding Techniques: An Overview," *IEEE Transactions on Vehicular Technology*, vol. 49, n°. 6, pags. 2208–2233, novembro 2000.

[15] J. Hagenauer, "The Turbo Principle: Tutorial Introduction and State of the Art," em *Proceedings of the International Symposium on Turbo Codes and Related Topics.*, Brest, França, setembro 1997, pags. 1–11.

[16] M. C. Valenti, "Iterative Detection and Decoding for Wireless Communications," Tese de Doutorado, Virginia State University (EUA), julho 1999.

[17] W. E. Ryan, "A Turbo Code Tutorial," acessado em junho 2003, http://www.ece.arizona.edu/ ryan/turbo2c.pdf.

[18] J. Hagenauer e P. Hoeher, "A Viterbi Algorithm with Soft-Decidion Outputs and its Applications," em *Proceedings IEEE Global Telecommunications Conference. GLOBECOM'89*, vol. 3, novembro 1989, pags. 1680–1686.

[19] H. V. Poor, *An Introduction to Signal Detection and Estimation.* Springer-Verlag, 1988.

[20] G. D. Forney, "Maximum Likelihood Sequence Estimator of Digital Sequences in the Presence of Intersymbol Interference," *IEEE Transactions on Information Theory*, vol. IT-18, pags. 363–378, maio 1972.

[21] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, pags. 260–269, abril 1967.

[22] L. Ljung e T. Söderström, *Theory and Practice of Recursive Identification.* Mit Press, 1987.

[23] G. Bauch, H. Khorram, e J. Hagenauer, "Iterative Equalization and Decoding in Mobile Communications Systems," em *Proceedings* 2^{nd} *European Personal Mobile Communications Conference. EPMCC'97*, Bonn, Alemanha, setembro/outubro 1997, pags. 307–312.

[24] A. Picart, P. Didier, e A. Glavieux, "Turbo-Detection: A New Approach to Combat Channel Frequency Selectivity," em *Proceedings IEEE International Conference on Communications. ICC'97*, vol. 3, Montreal, Canadá, junho 1997, pags. 1498–1502.

[25] M. Tüchler, A. C. Singer, e R. Koetter, "Minimum Mean Squared Error Equalization Using A Priori Information," *IEEE Transactions on Signal Processing*, vol. 50, n°. 3, pags. 673–683, março 2002.

[26] P. F. Petit, "Turbo-Equalization for QAM Constellations," Tese de Doutorado, University of South Australia (Australia), agosto 2002.

[27] P. Robertson, E. Villebrun, e P. Hoeher, "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain," em *Proceedings IEEE International Conference on Communications. ICC'95*, Seattle, Estados Unidos, junho 1995, pags. 1009–1013.

[28] R. R. Lopes, "Iterative Estimation, Equalization and Decoding," Tese de Doutorado, Georgia Institute of Technology (EUA), Atlanta, julho 2003.

[29] G. Bauch e V. Franz, "A Comparison of Soft-In/Soft-Out Algorithms for "Turbo-Detection"," em *Proceedings International Conference on Telecommunications, ICT'98* , junho 1998, pags. 259–263.

[30] R. R. Lopes e J. Barry, "Exploiting Error-Control Coding in Blind Channel Estimation," em *Proceedings IEEE Global Communications Conference – GLOBECOM'01* , vol. 2, novembro 2001, pags. 1317–1321.

[31] S. Y. Chung, T. J. Richardson, e R. L. Urbanke, "Analysis of Sum-Product Decoding of Low-Density Parity-Check Codes Using a Gaussian Approximation," *IEEE Transactions on Information Theory*, vol. 47, n°. 2, pags. 657–670, fevereiro 2001.

[32] C. Laot, "Égalisation Autodidacte et Turbo-Égalisation. Application aux canaux sélectifs en fréquence," Tese de Doutorado, L'Université de Rennes 1 (França), julho 1997.

[33] J. Proakis, *Digital Communications*. McGraw-Hill, 1995.

[34] V. D. Trajkovic e P. B. Rapajic, "Turbo Equalization Using Non-Systematic and Recursive Systematic Convolutional Codes," em *Proceedings IEEE Vehicular Technology Conference - VTC'03-Spring*, vol. 3, abril 2003, pags. 2125–2129.

[35] X. M. Chen e P. Hoeher, "Blind Turbo Equalization for Wireless DPSK Systems," em *Proceedings* 4^{th} *International ITG Conference on Source and Channel Coding*, Berlin, Alemanha, janeiro 2002, pags. 371–378.

[36] M. C. Jeruchim, P. Balaban, e K. S. Shanmugan, *Simulation of Communication Systems - Modeling, Methodology, and Techniques*, 2ª ed. Kluwer Academic/Plenum Publishers, 2000.

[37] X. Wang e R. Chen, "Blind Turbo Equalization in Gaussian and Impulsive Noise," *IEEE Transactions on Vehicular Technology*, vol. 50, n°. 4, pags. 1092–1105, julho 2001.

[38] S. Vlahoyiannatos e L. Hanzo, "Blind PSP-Based Turbo Equalization," em *Proceedings IEEE Vehicular Technology Conference - VTC'01-Spring*, vol. 3, 2001, pags. 1858–1862.

[39] S. Song, A. C. Singer, e K. M. Sung, "Turbo Equalization with an Unknown Channel," em *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing - ICAASP'02*, vol. 3, 2002, pags. 2805–2808.

[40] ——, "Soft Input Channel Estimation for Turbo Equalization," *IEEE Transactions on Signal Processing*, vol. 52, n°. 10, pags. 2885–2894, outubro 2004.

[41] M. Tüchler, R. Otnes, e A. Schmidbauer, "Performance of Soft Iterative Channel Estimation in Turbo Equalization," em *Proceedings IEEE International Conference on Communications - ICC'02*, vol. 3, 2002, pags. 1858–1862.

[42] L. Ljung, M. Morf, e D. Falconer, "Fast Calculation of Gain Matrices for Recursive Estimation Schemes," *International Journal of Control*, vol. 27, n°. 1, pags. 1–19, 1978.

[43] D. Falconer e L. Ljung, "Application of Fast Kalman Estimation to Adaptive Equalization," *IEEE Transactions on Communications*, vol. COM-26, n°. 10, pags. 1439–1446, outubro 1978.

[44] J. R. B. Gimenez, "Sobre a Estabilidade Numérica dos Algoritmos de Mínimos Quadrados Rápidos ," Tese de Doutorado, FEEC - UNICAMP, Campinas, janeiro 1995.

[45] J. M. Cioffi e T. Kailath, "Fast, Recursive-Least-Squares Transversal Filters for Adaptive Filtering," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-32, n°. 2, pags. 304–337, abril 1984.

[46] G. Carayannis, D. Manolakis, e N. Kalouptsidis, "A Fast Sequential Algorithm for Least-Squares Filtering and Prediction," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-31, pags. 1394–1402, dezembro 1983.

[47] M. A. Khalighi, "Effect of Mismatched SNR on the Performance of Log-MAP Turbo Detector," *IEEE Transactions on Vehicular Technology*, vol. 52, n°. 5, pags. 1386–1397, setembro 2003.

[48] H. Dai e H. V. Poor, "Advanced Signal Processing for Power Line Communications," *IEEE Communications Magazine*, pags. 100–107, maio 2003.

[49] S. K. Patra e B. Mulgrew, "Efficient Architecture for Bayesian Equalization using Fuzzy Filters," *IEEE Transactions on Circuits and Systems - II, Analog and Digital Signal Processing*, vol. 45, n°. 7, pags. 812–820, julho 1998.

[50] J. M. Mendel, "Fuzzy Logic Systems for Engineering: A Tutorial," em *Proceedings IEEE*, vol. 83, março 1995, pags. 345–377.

[51] S. Chen, B. Mulgrew, e S. McLaughlin, "Adaptive Bayesian Equalizer with Decision Feedback," *IEEE Transactions on Signal Processing*, vol. 41, n°. 9, pags. 2918–2927, setembro 1993.

[52] M. S. Yee, B. L. Yeap, e L. Hanzo, "Radial Basis Function-Assisted Turbo Equalization," *IEEE Transactions on Communications*, vol. 51, n°. 4, pags. 664–675, abril 2003.

[53] A. Dejonghe e L. Vandendorpe, "Turbo-equalization for multilevel modulation: an efficient low-complexity scheme," em *Proceedings IEEE International Conference on Communications - ICC'02*, vol. 3, abril-maio 2002, pags. 1863–1867.

[54] M. Koca e B. C. Levy, "Iterative Space-Time Detection of TCM for Broadband Wireless Channels," em *Proceedings* 3^{rd} *IEEE Signal Processing Workshop on Signal Processing Advances in Wireless Communications - SPAWC'01*, Taiwan, março 2001, pags. 122–125.

[55] T. Abe e T. Matsumoto, "Space-time turbo equalization in frequency-selective MIMO channels," *IEEE Transactions on Vehicular Technology*, vol. 52, n°. 3, pags. 469–475, maio 2003.

A

Artigos Publicados

Artigo publicado relacionado a este trabalho:

- M. B. Loiola, M. V. Ribeiro e J. M. T. Romano, "A Turbo Equalizer Using Fuzzy Filters," em *Proceedings 2004 IEEE International Workshop on Machine Learning for Signal Processing - MLSP'04*, São Luís, Brasil, pags. 695-704, setembro 2004.

Artigos publicados não relacionados a esta dissertação mas produzidos no decorrer do Mestrado:

- R. R. F. Attux, M. B. Loiola, R. Suyama, L. N. de Castro, F. J. Von Zuben e J. M. T. Romano, "Blind Search for Optimal Wiener Equalizers Using an Artificial Immune Network Model," *EURASIP Journal on Applied Signal Processing*, vol. 2003, nº 8, pags. 740-747, julho 2003.

- C. Junqueira, J. B. Destro Filho, D. Zanatta Filho, M. B. Loiola e J. M. T. Romano, "GCS - The Flexible GPS Channel Simulator," *Proceedings GNSS 2003 - The European Navigation Conference*, Graz, Áustria, abril 2003.

- C. Junqueira, J. B. Destro Filho, D. Zanatta Filho, M. B. Loiola e J. M. T. Romano, "A GPS Simulator for Analysis of Channel Impairments in Practical Scenarios," *Proceedings 2002 International Telecommunications Sysmposium - ITS'02*, Natal, Brasil, pags. 559-564, setembro 2002.