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

Códigos Turbo Quaternários

José da Silva Barros

Dissertação submetida à Faculdade de Engenharia Elétrica e de Computação da UNICAMP como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.

Banca Examinadora:

Prof. Dr. Renato Baldini Filho (Orientador) - FEEC/UNICAMP

Prof. Dr. Dalton Soares Arantes - FEEC/UNICAMP

Prof. Dr. Lee Luan Ling - FEEC/UNICAMP

Prof. Dr. Carlos Eduardo Câmara - USF

Campinas, 25 de Julho de 2003.

A Deus, pela vida e saúde.

A meus pais e irmãos, por tudo que temos vivido.

A Gininha, minha doce e constante saudade nesses três últimos anos.

A Osmário e Zé Neri*(In Memorian)*, por tudo que ainda representam para todos nós.

Agradecimentos

- A Deus, pelo êxito nessa caminhada cheia de alegrias e percalços.

- A meus pais, Manoel Barros e Josefa Barros, por sua incomparável lição de vida, amor, trabalho e dedicação.

- A meus irmãos e cunhados Marly, Ilza, Marlúcia, Nailza, Mauro, Nair, Rita, Rosiane, Emanoel, Julcimar e Junior e também a meus sobrinhos Mayara, Antônio, Myrelle, Cynara, Lucas, Larissa, Juninho e Marinho, pela amizade, solidariedade e companheirismo que fazem de nós um exemplo de família.

- Um agradecimento especial a Amauri e Regina e Juliene e Edson, pelo apoio e companhia que compartilhamos nesses anos em Campinas.

- A minha noiva Genaldir, pelo seu amor, carinho e compreensão nos momentos de ausência em que tive de me dedicar ao trabalho.

- A Tia Bibi, pelas brincadeiras, gentilezas e especialmente pelas orações que sempre fez por mim.

- A todos os meus amigos e parentes que, passando por minha vida, contribuíram de alguma forma para a minha carreira.

- Ao casal Vánio e Bete com seus filhos Izabela e Marcos (e também a dona Elena), Givaldo e Auriene com sua filha Aninha, João e Zuleica, Gilvan e Luzia, pela amizade, incentivo e representatividade nordestina aqui em Campinas.

- Um agradecimento especial ao meu orientador, Professor Renato Baldini, pela dedicação, paciência e boa vontade desprendidas ao meu trabalho de Mestrado.

- Aos Professores Dalton Soares Arantes (FEEC/UNICAMP), Lee Luan Ling (FEEC/UNICAMP), Carlos Eduardo Câmara (USF), membros da Banca Examinadora.

- Aos amigos da pós graduação, especialmente Rodrigo Ramos, Tarciana, Gonzalo, Magno, Márzio e Waldomiro, que contribuíram muito para a realização deste trabalho.

- Aos professores e funcionários da FEEC, pelo apoio e dedicação.

- Aos professores do departamento de matemática da UFAL, em especial aos professores Antônio Carlos, Adonai e José Carlos, pelo incentivo e respeito que sempre tiveram por mim.

- À FAPEAL (Fundação de Apoio a Pesquisa do Estado de Alagoas), pelo apoio concedido de 12 meses de bolsa de estudo para cursar o mestrado.

Resumo

O objetivo deste trabalho é apresentar uma abordagem matemática e estatística usada na construção de códigos turbo quaternários e analisar seu desempenho com relação aos melhores códigos turbo binários vistos na literatura. Para estes códigos, foram analisados o *esquema de codificação turbo,* que consiste de dois codificadores convolucionais sistemáticos recursivos quaternários, concatenados em paralelo e separados por um entrelaçador do tipo s-aleatório, e o *esquema de decodificação turbo,* que é formado por um algoritmo de decodificação iterativo quaternário, baseado na decodificação MAP (máximo a posteriori). Foram realizadas simulações para comprovar a semelhança do desempenho entre os casos binário e quaternário.

Abstract

The aim of this work is to present a mathematical and statistical model to build quaternary turbo codes and analyze its performance in relation to that obtained with the best binary turbo codes in the literature. Two turbo schemes were presented: a *coding scheme*, that consists of two quaternary systematic recursive convolutional encoders separated by s-random interleaver; and a *decoding scheme* based on the maximum *a posteriori* decoding algorithm - MAP. Simulations were carried out to confirm the similarity between the results of the quaternary and the binary schemes.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Apresentação do Estudo

Desde que Ungerboeck introduziu os conceitos de códigos treliça, em 1982 [5], os códigos turbo binários apresentados por C. Berrou, A. Glavieux e P. Thitimajashima em 1993 [1], representam a mais importante descoberta para o estudo de códigos na área de teoria da informação e codificação, por apresentar desempenho em seu esquema turbo (codificação e decodificação) bem próximo ao limitante de Shannon para o canal *AWGN* (*Aditive White Gaussian Noise*).

O esquema de codificação turbo é composto por dois codificadores convolucionais sistemáticos recursivos (sobre $\mathbb{Z}_2$), de taxa de codificação igual a 1/2, separados por um entrelaçador de N *bits* e concatenados em paralelo junto com um mecanismo opcional de puncionamento. A modulação considerada é a *BPSK* (*Binary Phase Shift Keying*).

O esquema de decodificação é composto por um algoritmo de decodificação de máximo *a posteriori* (*MAP*), símbolo a símbolo, que utiliza os conceitos de código treliça [4], para calcular a informação *a posteriori* que é usada no processo de decodificação iterativa turbo.

O processo de decodificação iterativa [1] usa o algoritmo de decodificação *MAP* em cada decodificador componente e tem a função de retirar a informação extrínseca do estágio de decodificação anterior e usá-la como informação *a priori* no próximo estágio de decodificação,

uma vez que este processo geralmente produz uma diminuição na probabilidade de erro de símbolo (ou *bit*) a cada iteração.

A partir da descoberta dos códigos turbo binários, houve um grande aumento no estudo destes códigos, no sentido de encontrar codificadores e entrelaçadores que proporcionem um melhor desempenho com uma complexidade de decodificação menor.

Utilizando o conhecimento adquirido sobre esses códigos turbo binários, propomos a seguir os códigos turbo quaternários.

1.2 Proposta

Com base nos estudos feitos sobre os códigos turbo binários [1], [2], [15], [3], [7] e [6], isto é, os códigos turbo definidos sobre o campo (corpo) dos números inteiros módulo-2, $\mathbb{Z}_2$, apresentamos os códigos turbo quaternários, ou seja, os códigos turbo definidos sobre os anéis de inteiros módulo-4, $\mathbb{Z}_4$.

Os códigos turbo quaternários têm uma esquematização similar à apresentada para o caso binário [1]. Todavia, por estarmos trabalhando nos anéis de inteiros módulo-4, onde a operação inversa da multiplicação não é definida, devemos estabelecer algumas restrições sobre os polinômios da matriz geradora correspondente aos codificadores que compõem o esquema de codificação turbo.

O esquema de codificação turbo proposto é composto por dois codificadores convolucionais sistemáticos recursivos quaternários, que obedecem a algumas limitações sobre o polinômio de realimentação ou polinômio do denominador da matriz geradora $g(D)$. Estes codificadores são concatenados em paralelo e separados por um entrelaçador, juntamente com um mecanismo opcional de puncionamento. Temos ainda que o esquema de modulação é o *4-PSK* (*Quaternary Phase Shift Keying*), pois esta modulação se adequa perfeitamente com os símbolos de $\mathbb{Z}_4$, e o canal considerado é o *AWGN*. Vemos também que o bom desempenho dos códigos turbo quaternários está diretamente associado à boa escolha dos componentes desse esquema

de codificação.

O algoritmo de decodificação de máximo *a posteriori* (MAP) fornece a informação *a posteriori*, a partir dos conceitos de código treliça e dos conceitos de distância euclidiana no cálculo da probabilidade de transição da métrica do ramo.

O processo de decodificação iterativa utiliza o algoritmo de decodificação MAP (quaternário) em cada decodificador componente, para retirar a informação extrínseca do estágio de decodificação anterior e utilizá-la como informação *a priori* no próximo estágio de decodificação. Este artifício de retirar a informação extrínseca do decodificador anterior e utilizá-la como informação *a priori* no próximo decodificador é o que caracteriza a decodificação iterativa e fornece uma diminuição na probabilidade de erro de símbolo (ou bit), que é proporcional ao "crescimento"do número de iterações.

1.3 Organização da Tese

Este trabalho utiliza a sistematização do esquema de codificação turbo binário para uma nova classe de códigos, os códigos convolucionais sistemáticos recursivos sobre $\mathbb{Z}_4$, além de utilizar os conceitos de distância euclidiana ao quadrado no esquema de decodificação turbo quaternário. A tese está organizada em 5 capítulos.

No Capítulo 1 apresentamos sinteticamente as propostas e objetivos que serão realizados neste trabalho.

No Capítulo 2 introduzimos o esquema de codificação turbo quaternário, que inicia com as definições matemáticas que garantem a existência (construção) dos códigos convolucionais sistemáticos recursivos quaternários, e segue com o esquema de codificação turbo sobre $\mathbb{Z}_4$, descrevendo seus componentes, tais como: codificador, entrelaçador e puncionador. Ainda relacionado ao entrelaçador, mostramos resultados simulados sobre sua aleatoriedade (ou sobre o espalhamento dos seus elementos) e incluímos, também nesse capítulo, uma rápida descrição do modulador e do canal utilizado.

No Capítulo 3 apresentamos o algoritmo de decodificação de máximo *a posteriori* (MAP), onde obtemos as expressões da probabilidade *a posteriori*, usando as propriedades estatísticas e os conceitos de código treliça [1] e [4]. Em seguida, deduzimos a expressão recursiva direta, $\alpha_k(s)$, e a expressão recursiva reversa, $\beta_k(s)$, e também o cálculo da métrica do ramo $\gamma_k(s', s)$. E, finalmente, fazemos um resumo das principais expressões do algoritmo MAP ($\alpha_k(s)$, $\beta_k(s)$ e $\gamma_k(s', s)$) que são usadas para calcular a informação *a posteriori*.

No Capítulo 4 descrevemos o processo de decodificação iterativa. Esse capítulo é iniciado com uma abordagem dos conceitos de distância euclidiana, nas expressões da probabilidade de transição da métrica do ramo, para, em seguida, mostrarmos que a informação *a posteriori* é a soma da informação *a priori*, da informação sistemática e da informação extrínseca. O passo final desse capítulo é exibir o funcionamento do esquema de decodificação iterativa turbo (a cada iteração), fazendo uso dos conceitos de informação intrínseca, extrínseca e *a posteriori*.

No Capítulo 5 mostramos os resultados das simulações realizadas para os códigos turbo sobre $\mathbb{Z}_4$, que dependem de fatores de construção dos componentes destes códigos, tais como: o grau dos polinômios da matriz geradora do codificador convolucional sistemático recursivo, o comprimento da seqüência de informação a ser entrelaçada, a aleatoriedade ou espalhamento entre os elementos dos entrelaçadores, a taxa de codificação e o número de iterações utilizadas no esquema de decodificação. Desse modo, descrevemos aspectos relacionados às simulações, analisando os resultados de forma comparativa e, finalmente, apresentamos as conclusões e sugestões para trabalhos futuros.

Capítulo 2

Codificação Turbo Quaternária

2.1 Introdução

Neste capítulo, fazemos um estudo que contém uma representação geral de todo esquema de codificação turbo quaternária. Iniciamos definindo algumas propriedades matemáticas que garantem a existência e construção dos códigos convolucionais, que, por sua vez, são de fundamental importância para a construção dos códigos convolucionais sistemáticos recursivos, usados no esquema de codificação turbo quaternário. Este esquema de codificação turbo, que é composto por dois codificadores idênticos, separados por um entrelaçador, juntamente com um puncionador, está estruturado na Figura 2.1.

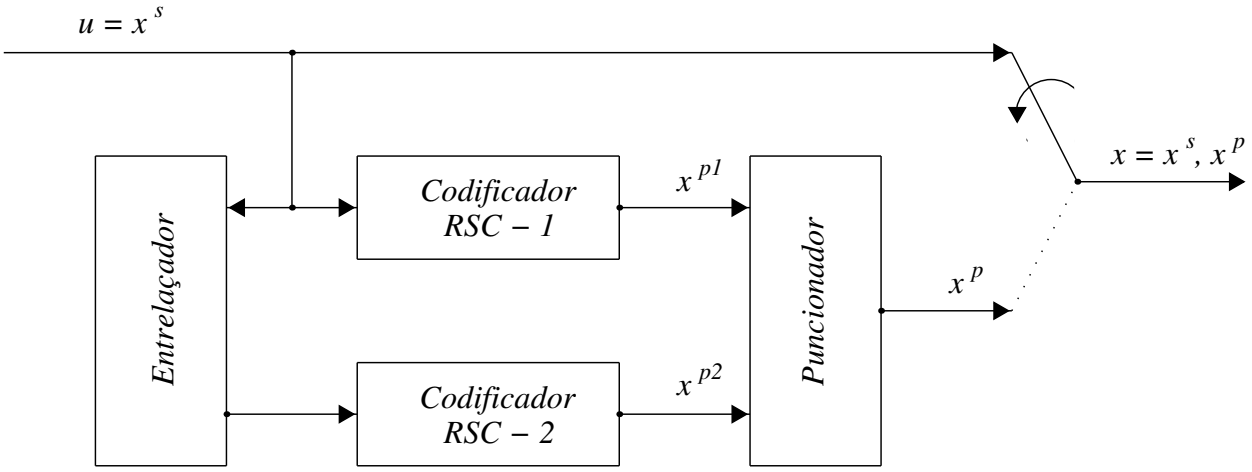
Além disso, precisamos construir codificadores e entrelaçadores a partir de propriedades matemáticas e estatísticas que proporcionem melhor desempenho nos códigos turbo. O puncionador utilizado tem a função de reduzir a taxa do código e a modulação *4-PSK* foi escolhida por ter um casamento (adequação) melhor com os símbolos quaternários sobre o canal *AWGN*.

2.2 Esquema de Codificação Turbo

Inicialmente, descrevemos um esquema padrão de codificação turbo (codificador turbo), constituído por dois codificadores convolucionais idênticos, de taxa 1/2, separados por um en-

trelaçador de N-símbolos, juntamente com um mecanismo opcional de puncionamento. Esses dois codificadores constituintes são codificadores convolucionais, da variedade sistemático recursiva e concatenados em paralelo, conforme a Figura 2.1.

O esquema de codificação turbo quaternário que propomos é equivalente ao esquema binário [1] e [15]; todavia, o esquema proposto trabalha com símbolos do alfabeto quaternário. Desse modo, o entrelaçador e o puncionador podem ser considerados os mesmos do caso binário, enquanto que os codificadores serão substituídos por codificadores quaternários [8], [9], [11], [16], [12] e [13], mostrados nas deduções posteriores.



$u = x^s$	Seqüência de símbolos de informação ou de símbolos sistemáticos
x^{p1}	Seqüência de símbolos de paridade proveniente do codificador $RSC - 1$
x^{p2}	Seqüência de símbolos de paridade proveniente do codificador $RSC - 2$
x^p	Seqüência de símbolos de paridade proveniente dos codificadores $RSC - 1$ e $RSC - 2$
$x = x^s, x^p$	Seqüência código (símbolos de informação e símbolos de paridade)

Figura 2.1: Esquema de codificação turbo.

Usamos agora as subseções seguintes para descrever os componentes individuais do diagrama de codificação turbo, mostrado na Figura 2.1, iniciando com os codificadores.

2.2.1 Codificador Convolucional Sistemático Recursivo

Iniciamos o estudo dos códigos turbo quaternários dando um tratamento matemático e estatístico a todos os componentes do esquema turbo, proposto sobre o anel de inteiros módulo-4, $\mathbb{Z}_4$. Primeiro, começamos definindo os fundamentos matemáticos para a construção dos códigos convolucionais sistemáticos e, a partir destes, construímos os códigos convolucionais sistemáticos recursivos.

Seja $\mathbb{Z}_4$ um anel de inteiros comutativo com identidade multiplicativa e seja $\mathbb{Z}_4[D]$ um anel dos polinômios com coeficientes em $\mathbb{Z}_4$, onde os polinômios de $\mathbb{Z}_4[D]$ são expressos pela série de *Laurent* [20]

$$f(D) = \sum_{i=0}^{n} f_i D^i,$$

com $f_i \in \mathbb{Z}_4$. O coeficiente mínimo de um polinômio não nulo, $f(D)$, é o coeficiente da menor potência de D com coeficiente não nulo.

Agora, considere $L(D)$ como sendo o conjunto formado por polinômios racionais do tipo $\frac{q(D)}{p(D)}$, onde $q(D)$ e $p(D) \in \mathbb{Z}_4[D]$ e o coeficiente mínimo de $p(D)$ é inversível em $\mathbb{Z}_4$.

Logo, o conjunto $L(D)$, módulo a relação de equivalência,

$$\frac{q(D)}{p(D)} \equiv \frac{q_1(D)}{p_1(D)} \text{ se, e somente se, } q(D) \cdot p_1(D) = q_1(D) \cdot p(D), \tag{2.1}$$

é um anel comutativo com identidade multiplicativa, chamado anel das funções racionais na variável D.

Desse modo, chamamos de codificador convolucional com taxa de codificação k/n sobre o anel de funções racionais $L(D)$, o mapeamento linear

$$\begin{aligned} L(D)^k &\longrightarrow L(D)^n \\ u(D) &\longmapsto v(D), \end{aligned}$$

que pode ser expresso como

$$v(D) = u(D)G(D),$$

onde $G(D)$ é uma matriz $k \times n$ (chamada matriz função de transferência ou matriz geradora), com elementos em $L(D)$, cujas linhas são linearmente independentes sobre $L(D)$. E, ainda, $u(D)$ é o polinômio correspondente à seqüência de informação e $v(D)$ é o polinômio correspondente à seqüência código.

Um conjunto

$$C = \{u(D)G(D) \mid u(D) \in L(D)^k\},$$

onde $G(D)$ é a matriz geradora de um codificador convolucional sobre $L(D)$, é um código convolucional com taxa de codificação k/n sobre $\mathbb{Z}_4$. Assim, por exemplo, dadas a seqüência de informação

$$u(D) = 3 + 2D + D^2 + D^4 + 2D^5,$$

e a matriz geradora, 1×2,

$$G(D) = \begin{bmatrix} 1 + D + 3D^2 & 2 + D + 2D^2 \end{bmatrix},$$

temos que o código convolucional com taxa de codificação $1/2$ sobre $\mathbb{Z}_4$ é

$$\begin{aligned} v(D) &= u(D)G(D) \\ &= \begin{bmatrix} 3 + D + 3D^3 + 3D^5 + D^6 + 2D^7 & 2 + 3D + 2D^2 + D^3 + D^5 \end{bmatrix}. \end{aligned}$$

Agora, seja $L_r(D)$ um subanel de $L(D)$ (chamado anel das funções realizáveis sobre $\mathbb{Z}_4$), consistindo dos elementos da classe de equivalência que contém uma representação $\dfrac{q(D)}{p(D)}$, com $p(0) = 1$.

Uma matriz geradora convolucional $G(D)$, $k \times n$, é dita como sistemática se ela possuir uma

submatriz, $k \times k$, que é uma matriz identidade. Assim, sendo por exemplo,

$$G(D) = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & D & 2+2D \end{bmatrix},$$

esta matriz geradora convolucional $G(D)$, 2×4, possui uma submatriz, 2×2, formada por suas duas primeiras colunas, que é uma matriz identidade. Logo, $G(D)$ é uma matriz sistemática.

Portanto, um código convolucional C sobre $\mathbb{Z}_4$ é sistemático se ele possuir uma matriz geradora sistemática. O teorema a seguir garante este resultado.

Teorema 2.2.1 *Um código convolucional C sobre o anel $\mathbb{Z}_4$ é sistemático se, e somente se, ele possui uma matriz geradora, $G(D)$, $k\times n$, que possui uma submatriz $k\times k$ cujo determinante é uma unidade em $L_r(D)$ - anel das funções realizáveis sobre $\mathbb{Z}_4$.*

Prova.

($\Rightarrow$) Se $G(D)$ é sistemática, então $G(D) = \begin{bmatrix} I_k & \bar{G}(D) \end{bmatrix}$. Portanto, o determinante de I_k é uma unidade em $\mathbb{Z}_4$ (e em $L_r(D)$).

($\Leftarrow$) Se $G(D)$ possui uma submatriz $k \times k$, $A(D)$, cujo determinante é uma unidade em $L_r(D)$, então, sem perda de generalidade, seja $G(D) = \begin{bmatrix} A(D) & B(D) \end{bmatrix}$. Como $A(D)$ possui determinante unitário em $L_r(D)$, então $A(D)$ possui uma inversa $A^{-1}(D)$ em $L_r(D)$ e

$$G_{sist}(D) = A^{-1}(D)G(D) = \begin{bmatrix} I_k & \bar{B}(D) \end{bmatrix}$$

é uma matriz geradora equivalente para o código C. ■

Com base no que foi apresentado sobre os códigos convolucionais sistemáticos, construímos agora os códigos convolucionais sistemáticos recursivos sobre $\mathbb{Z}_4$.

Assim, um codificador convolucional sistemático recursivo-*RSC (Recursive Systematic Convolutional)*, com taxa de codificação k/n sobre o anel de funções de realizáveis $L_r(D)$, é um

mapeamento linear

$$\begin{aligned} L_r(D)^k &\longrightarrow L_r(D)^n \\ u(D) &\longmapsto v(D), \end{aligned}$$

que pode ser expresso como

$$v(D) = u(D)G_r(D),$$

onde $G_r(D)$ é uma matriz $k \times n$ da forma

$$G_r(D) = \left[\begin{array}{cc} I_k & X(D) \end{array} \right].$$

Observe que I_k é uma matriz identidade $k \times k$, $X(D)$ é uma matriz $k \times (n\text{-}k)$ com elementos cuja representação é $\dfrac{q(D)}{p(D)}$ em $L_r(D)$ e $p(0) = 1$ em $\mathbb{Z}_4$.

Exemplo: Considere o codificador convolucional não sistemático representado pela matriz geradora

$$G(D) = \left[\begin{array}{cc|cc} 1 & 1+D & 2 & 1 \\ 0 & 1+2D & D & 3+2D \end{array} \right].$$

Podemos transformar este codificador, não sistemático, em um codificador RSC, como se segue. Seja

$$Q(D) = \left[\begin{array}{cc} 1 & 1+D \\ 0 & 1+2D \end{array} \right],$$

a submatriz $k \times k$ (2×2), formada pelas duas primeiras colunas da matriz $G(D)$ e seja

$$Q^{-1}(D) = \left[\begin{array}{cc} 1 & \dfrac{3+3D}{1+2D} \\ 0 & \dfrac{1}{1+2D} \end{array} \right],$$

a inversa a direita da submatriz $Q(D)$. Multiplicando $Q^{-1}(D)$ por $G(D)$, obtemos a matriz

sistemática recursiva

$$G_r(D) = Q^{-1}(D)G(D) = \begin{bmatrix} 1 & 0 & \dfrac{2+3D+3D^2}{1+2D} & \dfrac{2+D+2D^2}{1+2D} \\ 0 & 1 & \dfrac{D}{1+2D} & \dfrac{3+2D}{1+2D} \end{bmatrix}.$$

Este exemplo mostra a transformação da matriz geradora de um codificador convolucional não sistemático numa matriz geradora de um codificador convolucional sistemático recursivo. Assim, temos agora a definição do código RSC:

Definição 2.2.2 *O conjunto*

$$C = \{u(D)g(D) \mid u(D) \in L_r(D)^k\},$$

é um código RSC com taxa de codificação k/n sobre $\mathbb{Z}_4$, onde $g(D) = G_r(D)$ é a matriz geradora de um codificador RSC com elementos em $L_r(D)$.

Daqui por diante usamos a notação mais simples para a seqüência código e para a seqüência de informação, conforme o exemplo seguinte.

Exemplo: A Figura 2.2 mostra o diagrama de um codificador RSC que tem matriz geradora

$$g(D) = \begin{bmatrix} 1 & \dfrac{2+D+2D^2}{1+D+3D^2} \end{bmatrix}. \tag{2.2}$$

A seqüência de informação

$$u = x^s = \quad 3\ \ 2\ \ 1\ \ 0\ \ 1\ \ 2\ \ 3\ \ 3\ \ 1\ \ 0\ \ 2\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 2\ \ 2\ \ 3\ \ 2\ \ 1\ \ 0\ \ 1\ \ 2\ \ 1$$

aplicada ao codificador RSC da Figura 2.2 produz a seqüência de saída (seqüência código)

$$x = \begin{array}{|cccccccccccccccccccccccccc} x^s & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^p & 2 & 1 & 3 & 0 & 3 & 0 & 3 & 2 & 1 & 1 & 2 & 0 & 3 & 1 & 2 & 2 & 0 & 1 & 2 & 2 & 2 & 1 & 3 & 3 & 2. \end{array}$$

Como era de se esperar, a seqüência código é composta pela seqüência de informação (símbolos sistemáticos) - nas posições pares, e pela seqüência de paridade - nas posições ímpares. Os dois últimos símbolos sistemáticos são os símbolos que fazem este codificador *RSC*, com esta seqüência de informação, voltar ao estado inicial. Esta é a condição necessária usada no cálculo da expressão recursiva $\beta_k(s)$, que é vista no próximo capítulo.

Note que estamos interessados em um esquema turbo (codificação e decodificação) quaternário que forneça o melhor desempenho possível. Por outro lado, sabemos, a partir de artigos estudados para o caso binário [1], [2], [15], [3], [7], [6], [18] e [14], que os codificadores que propomos (com máxima distância euclidiana mínima ao quadrado entre as palavras código) proporcionam melhor desempenho se suas matrizes geradoras $\left(g(D) = \left[1 \quad \frac{q(D)}{p(D)}\right]\right)$ obedecerem as seguintes restrições:

- O polinômio $p(D)$ de $\frac{q(D)}{p(D)}$ é irredutível e $p(0) = 1$ em $\mathbb{Z}_4$, uma vez que $g(D)$ está definida sobre um anel (anel de *Noetherian* [17], [20] e [16]);

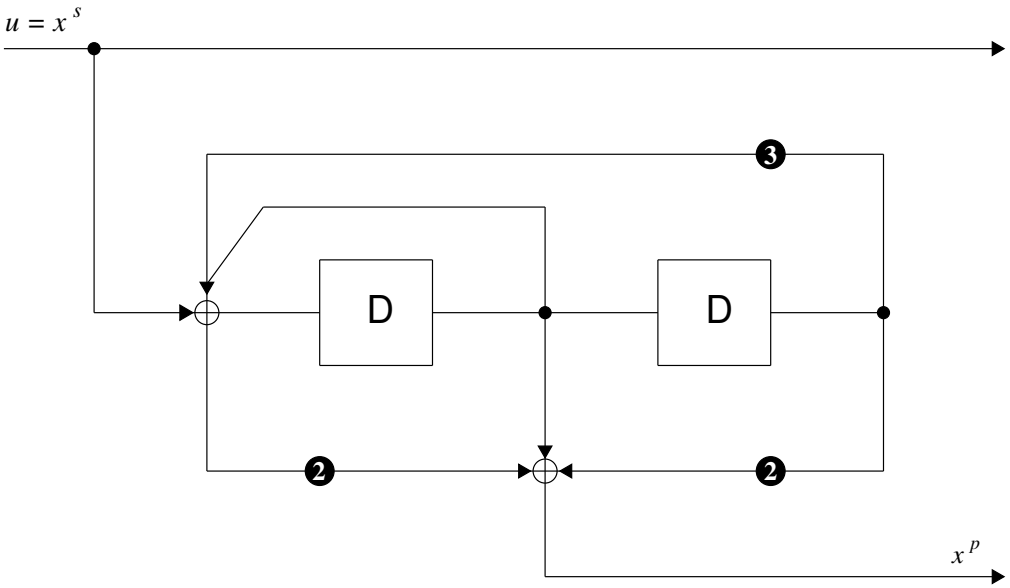


Figura 2.2: Diagrama de um codificador *RSC* quaternário.

Analisamos a seguir as propriedades dos entrelaçadores que proporcionam melhor desempenho nos códigos turbo.

2.2.2 Entrelaçadores

Os entrelaçadores, usados com grande freqüência nos meios de comunicação, têm por objetivo espalhar erros em surto (*burst*), causados pelo ruído impulsivo e pelo desvanecimento (*fading*) seletivo. Sua função clássica é entrelaçar, de forma bastante aleatória, os erros introduzidos no canal, de tal forma que os erros possam ser considerados descorrelacionados, contribuindo assim para que os decodificadores sejam mais eficientes.

Assim, seja

$$\begin{aligned} \pi \;:\; \mathbb{Z} &\longrightarrow \mathbb{Z} \\ i &\longmapsto \pi(i) \quad \forall\, i \in \mathbb{Z} \end{aligned}$$

uma permutação nos inteiros $\mathbb{Z}$. A permutação assim definida é a representação matemática mais simples de um entrelaçador. No entanto, o entrelaçador é formado por um conjunto de operações matemáticas e ou estatísticas onde uma delas (ou a única delas) é a permutação. Desse modo, o entrelaçador é uma função inversível, que recebe em sua entrada N-símbolos de um dado alfabeto, e produz em sua saída os mesmos N-símbolos em uma diferente ordem no tempo. Portanto, sua função no esquema de codificação turbo, Figura 2.1, é tomar cada bloco de N-símbolos de informação que chega em sua entrada e rearranjá-lo em uma forma pseudo aleatória para a codificação *a priori* no codificador *RSC-2*.

Exemplo: O entrelaçador matricial

$$P = \begin{array}{|c|c|c|c|c|} \hline 1 & 2 & 3 & 4 & 5 \\ \hline 6 & 7 & 8 & 9 & 10 \\ \hline 11 & 12 & 13 & 14 & 15 \\ \hline 16 & 17 & 18 & 19 & 20 \\ \hline 21 & 22 & 23 & 24 & 25 \\ \hline \end{array}, \tag{2.3}$$

que ordena seus elementos como linhas da matriz P e os lê como coluna, ou equivalente ao

entrelaçador I_π (ou permutação $\pi(i)$), da Figura 2.3, que transforma a seqüência de informação

$$x^s = u = \begin{array}{cccccccccccccccccccccccc} 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \end{array}$$

na seqüência de informação entrelaçada

$$u' = \begin{array}{cccccccccccccccccccccccc} 3 & 2 & 2 & 0 & 1 & 2 & 3 & 0 & 2 & 0 & 1 & 3 & 0 & 2 & 1 & 0 & 1 & 1 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \end{array}$$

que, por sua vez, inserida no codificador *RSC-2* da Figura 2.1 (com matriz geradora descrita na expressão (2.2)), produz a seqüência código

$$x = \left| \begin{array}{cccccccccccccccccccccccc} x^s & 3 & 2 & 2 & 0 & 1 & 2 & 3 & 0 & 2 & 0 & 1 & 3 & 0 & 2 & 1 & 0 & 1 & 1 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^p & 2 & 1 & 3 & 0 & 3 & 0 & 3 & 2 & 1 & 1 & 2 & 0 & 3 & 1 & 2 & 2 & 0 & 1 & 2 & 2 & 2 & 1 & 3 & 3 & 2 \end{array} \right.$$

Definimos agora algumas propriedades matemáticas e estatísticas que serão utilizadas na construção dos entrelaçadores.

Seja

$$D(I) = \{(\triangle_x, \triangle_y) \in \mathbb{Z}^2 \mid \triangle_x = j - i,\ \triangle_y = \pi(j) - \pi(i),\ 0 \leq i < j < T\},$$

o vetor espalhamento de um entrelaçador I, onde T é o período (comprimento) da permutação. Daí, um entrelaçador possui fator de espalhamento (s, t), se, para $|\triangle_x| < s$, temos $|\triangle_y| \geq t$, onde $s, t \in \mathbb{Z}_+^*$ (conjunto dos números inteiros positivos).

O entrelaçador matricial I_π da Figura 2.3, que possui fator de espalhamento $(s = 6, t = 6)$, não possui espalhamento satisfatório para um conjunto com mais de cinco erros em surto. Como podemos ver, o ruído impulsivo que ocorre nas posições de 07 a 12, quando passa por I_π, só separa as posições de 07 a 11. Veja que $|\triangle_x| = j - i = 12 - 7 = 5$ e $|\triangle_y| = \pi(j) - \pi(i) = 08 - 07 = 1$, ou seja, as posições 07 e 12 ficam juntas nas posições

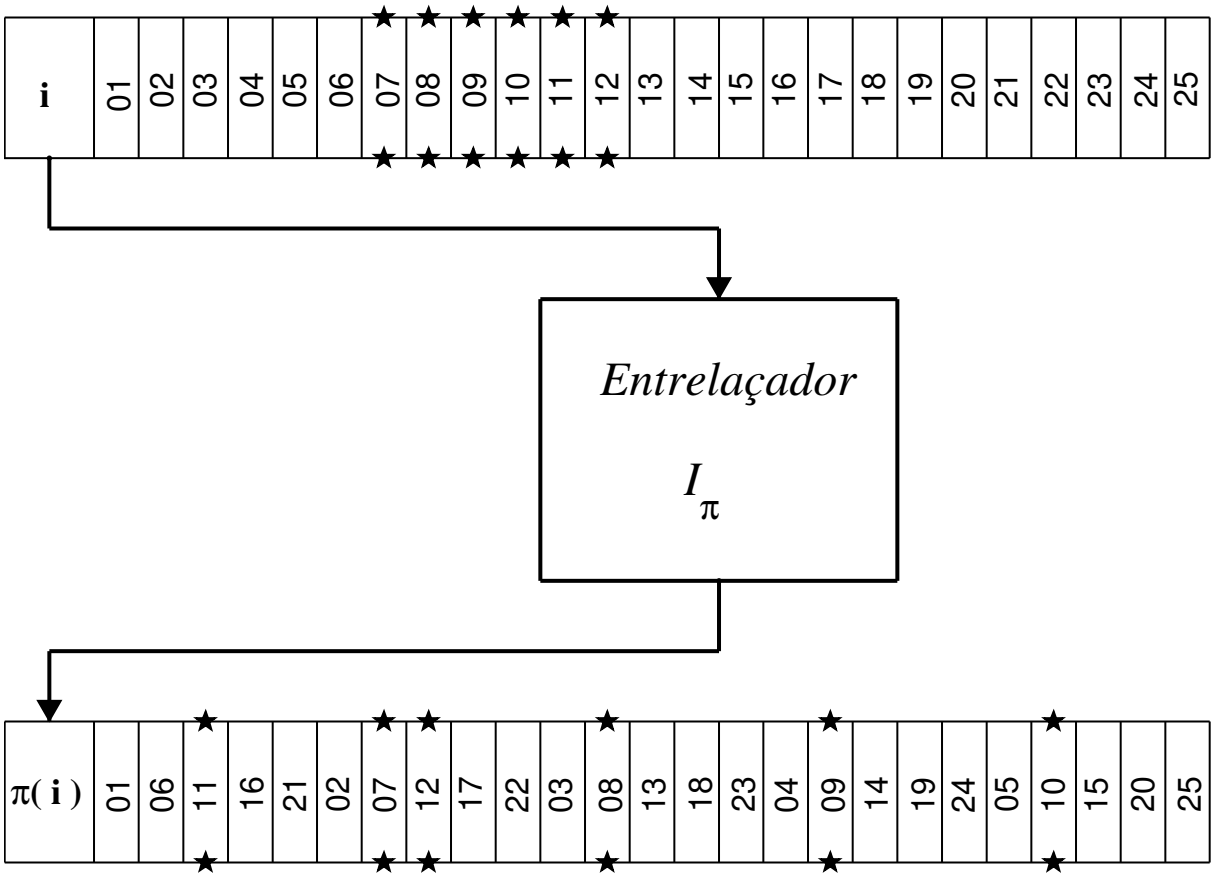


Figura 2.3: Entrelaçador matricial (ou permutação) de bloco (N = 25).

07 e 08, respectivamente.

Esta apresentação permite um melhor entendimento sobre como devemos projetar entrelaçadores capazes de separar conjuntos de erros (em surto ou não) de maneira satisfatória. Ou seja, dado que a distância euclidiana entre as posições de um eventual conjunto de erros (em surto) seja $|\triangle_x| < s$, devemos escolher um entrelaçador que, sempre que possível, separe as posições destes erros à distância euclidiana $|\triangle_y| \geq t$, onde $s \leq t$ e $s, t \in \mathbb{Z}_+^*$. Os entrelaçadores que apresentamos a seguir foram construídos com base nesse estudo para proporcionar melhor separação entre os erros.

Considere o entrelaçador usado por C. Berrou, A. Glavieux e P. Thitimajashima em seu esquema de turbo binário [1], [18], mostrado na Figura 2.4, cuja representação matemática é a seguinte:

Tome $K = 2^k,\ M = 2^m,\ k, m \in \mathbb{Z}_+^*$ (conjunto dos números inteiros positivos) e defina oito números primos:

$$\begin{aligned} p(1) &= 17; \\ p(2) &= 37; \\ p(3) &= 19; \\ p(4) &= 29; \\ p(5) &= 41; \\ p(6) &= 23; \\ p(7) &= 13; \\ p(8) &= 7. \end{aligned}$$

Então para cada número inteiro i, tal que $0 \leq i \leq K \cdot M$. Seja

$$\pi(i) = c(i) + M \cdot r(i),$$

onde

$$\begin{aligned} r(i) &= \text{mod}(p(l+1) \cdot (c_0 + 1) - 1, K); \\ c(i) &= \text{mod}(M/2 + 1) \cdot (r_0 + c_0), M); \\ r_0 &= \text{mod}(i, M); \\ c_0 &= (i - r_0)/M; \\ l &= \text{mod}((r_0 + c_0), 8). \end{aligned}$$

Além do entrelaçador anterior usado por C. Berrou, A. Glavieux e P. Thitimajashima [18] e [19], destacamos também neste trabalho o entrelaçador s-aleatório, conforme descrito a seguir.

A Figura 2.5 mostra um entrelaçador s-aleatório [18] gerado por uma permutação

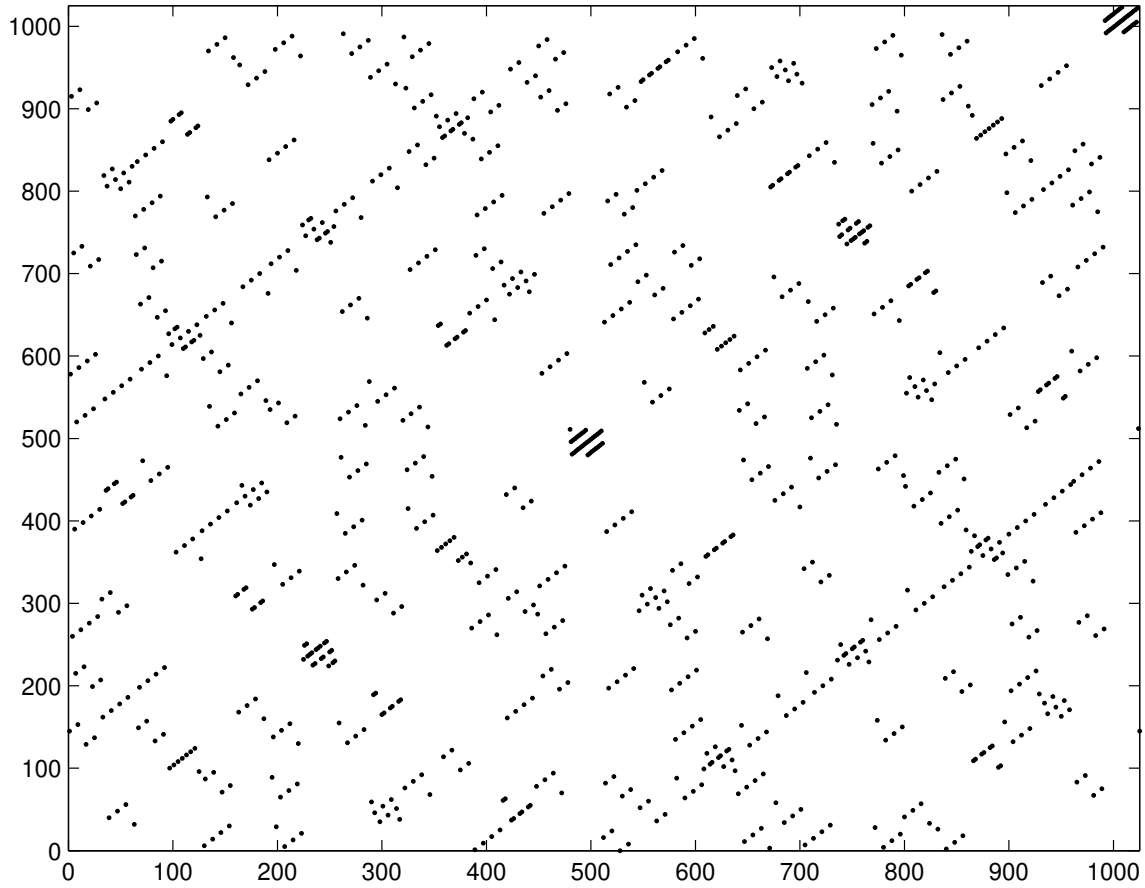


Figura 2.4: Entrelaçador de bloco, $N = 1024$, usado por Berrou-Glavieux.

s-aleatória baseada em uma fonte de ruído aleatório. Por exemplo, um vetor ruído de comprimento N é gerado e a permutação que põe (ordena) o vetor ruído na ordem sorteada (entrelaçada) é usada para gerar o entrelaçador.

Note que os entrelaçadores mostrados nas Figuras 2.4 e 2.5 exigem um estudo que vai além das propriedades apresentadas sobre o espalhamento dos seus elementos, vistas nessa seção. Este estudo, que foi usado para construir esses entrelaçadores, pode ser visto nas referências [18], [1] e [19]. Observe também que o entrelaçador da Figura 2.5 apresenta melhor espalhamento entre seus elementos que o entrelaçador mostrado na Figura 2.4; por isso vamos usar o entrelaçador da Figura 2.5 em nossas simulações.

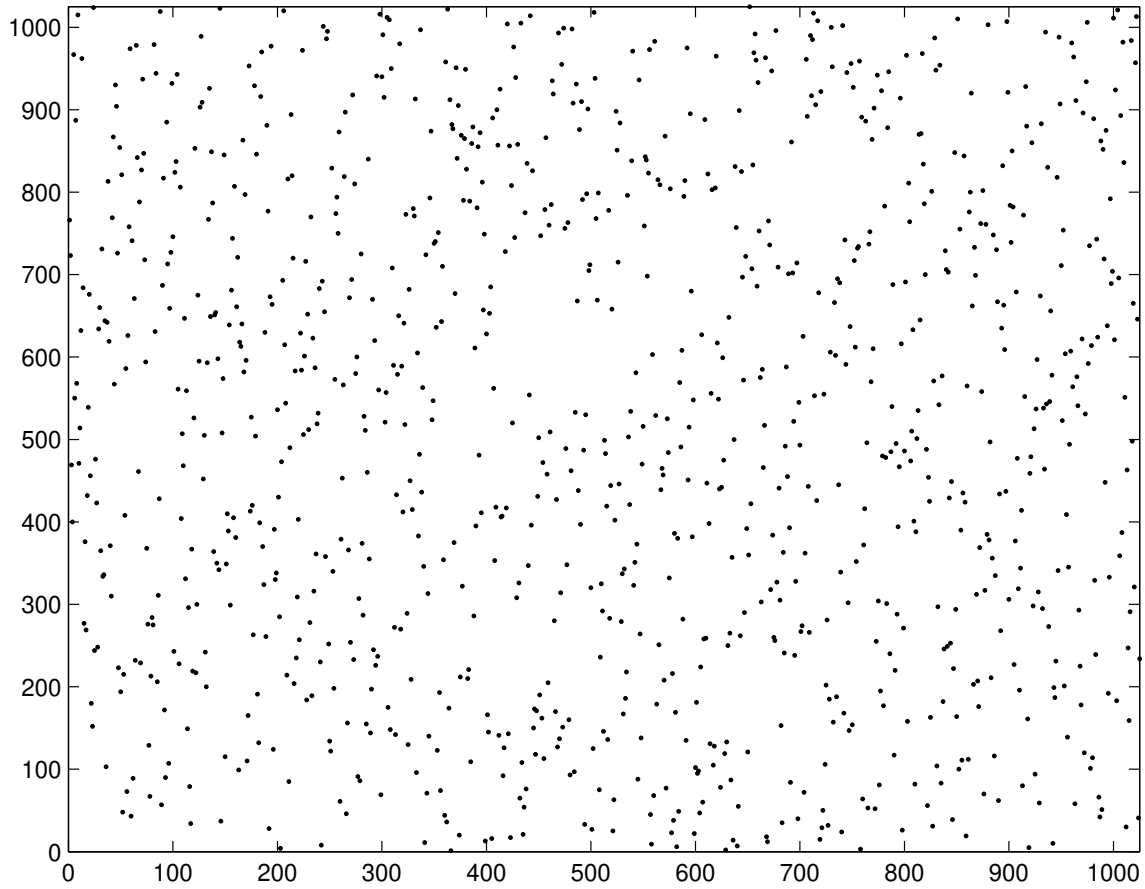


Figura 2.5: Entrelaçador s-aleatório de bloco (N = 1024).

2.2.3 Puncionador

A função do puncionador é apagar periodicamente símbolos de redundância pré-selecionados dos codificadores, aumentando a taxa de codificação.

Com base nos exemplos das expressões (2.2) e (2.3), quando não é usado o puncionador, a taxa de codificação turbo é 1/3 e a seqüência código correspondente ao esquema turbo é dada por:

$$x = \left| \begin{array}{cccccccccccccccccccccccccc} x^s & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^{p1} & 2 & 1 & 3 & 0 & 3 & 0 & 3 & 2 & 1 & 1 & 2 & 0 & 3 & 1 & 2 & 2 & 0 & 1 & 2 & 2 & 2 & 1 & 3 & 3 & 2 \\ x^{p2} & 2 & 1 & 1 & 1 & 0 & 0 & 2 & 3 & 0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 1 & 3 & 2 & 2 & 2 & 1 & 3 & 3 & 2. \end{array} \right.$$

E, quando é usado o puncionador, a taxa de codificação turbo aumenta de 1/3 para 1/2,

apagando-se os símbolos de paridade ímpares, provenientes do codificador *RSC-1*, e os símbolos pares, provenientes do codificador *RSC-2*

$$x = \left| \begin{array}{cccccccccccccccccccccccc} x^s & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^{p1} & 2 & & 3 & & 3 & & 3 & & 1 & & 2 & & 3 & & 2 & & 0 & & 2 & & 2 & & 3 & & 2 \\ x^{p2} & & 1 & & 1 & & 0 & & 3 & & 0 & & 0 & & 0 & & 0 & & 3 & & 2 & & 1 & & 3 & \end{array} \right| ,$$

produzindo assim a seguinte seqüência código[1]

$$x = \left| \begin{array}{cccccccccccccccccccccccc} x^s & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^{p} & 2 & 1 & 3 & 1 & 3 & 0 & 3 & 3 & 1 & 0 & 2 & 0 & 3 & 0 & 2 & 0 & 0 & 3 & 2 & 2 & 2 & 1 & 3 & 3 & 2 \end{array} \right| \tag{2.4}$$

na saída do esquema de codificação turbo, ver Figura 2.1.

Conforme o exemplo anterior, quando usamos o puncionamento, é mapeado N símbolo de informação em $2N$ símbolos do código, enquanto que, sem o puncionamento, é mapeado N símbolo de informação em $3N$ símbolos do código. Fazemos a seguir uma breve apresentação sobre a modulação *4-PSK*.

2.2.4 Modulador

Considere que a seqüência de entrada do modulador seja composta por símbolos do alfabeto $\theta = \{0, 1, 2, 3\} \in \mathbb{Z}_4$. Assim, o sinal modulado será representado por

$$S_\theta = \left(\cos\left(\frac{\pi\theta}{2} + \frac{\pi}{4}\right), \sin\left(\frac{\pi\theta}{2} + \frac{\pi}{4}\right)\right).$$

Portanto, o modulador transforma os símbolos $\in \mathbb{Z}_4$ em pares ordenados, (n, m), de uma constelação em $\mathbb{R}^2$. Como podemos ver facilmente, os símbolos de $\mathbb{Z}_4$, quando modulados, transformam-se em pontos pertencentes a intersecção da circunferência de raio 1 (energia média

[1]Seqüência obtida com base nos exemplos associados as expressões (2.2) e (2.3).

normalizada em 1) com os eixos de simetria. Isto é, são os pontos

$$0 \equiv \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right),\ 1 \equiv \left(-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right),\ 2 \equiv \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right) \text{ e } 3 \equiv \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right),$$

localizados nos quadrantes do $\mathbb{R}^2$, como mostra a Figura 2.6. A partir daí, estes símbolos são transmitidos pelo canal $AWGN$, que podemos ver melhor no exemplo a seguir.

O modulador recebe, por exemplo, a seqüência do puncionador (ver expressão (2.4))

$$x = \begin{vmatrix} x^s & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 3 & 1 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ x^p & 2 & 1 & 1 & 1 & 0 & 0 & 2 & 3 & 0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 1 & 3 & 2 & 2 & 2 & 1 & 3 & 3 & 2 \end{vmatrix}$$

e produz a seqüência modulada de pares ordenados

$$\bar{x} = \begin{vmatrix} \bar{x}^s & (0.7071, -0.7071) & (-0.7071, -0.7071) & \cdots & (-0.7071, -0.7071) & (-0.7071, 0.7071) \\ \bar{x}^p & (-0.7071, -0.7071) & (-0.7071, 0.7071) & \cdots & (0.7071, -0.7071) & (-0.7071, -0.7071) \end{vmatrix}$$

Em seguida, transmitindo $\bar{x}$ pelo canal $AWGN$, recebemos, devido ao ruído, por exemplo, a seqüência

$$y = \begin{vmatrix} y^s & (1.2148, -0.4456) & (0.2371, -1.5382) & \cdots & (-0.0175, -0.9563) & (-0.7002, -1.0324) \\ y^p & (-1.1094, -0.5743) & (0.0021, -0.0010) & \cdots & (0.0114, -0.8953) & (-0.0002, -1.5438) \end{vmatrix}$$

na entrada do esquema de decodificação turbo, a ser visto nos próximos capítulos.

A partir do que foi exposto, temos ferramentas suficientes para construir um esquema de codificação turbo quaternário, e, no próximo capítulo, apresentamos o algoritmo usado na decodificação da informação produzida pelo esquema de codificação proposto.

$$0 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$$

$$1 = \left(\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$$

$$2 = \left(\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right)$$

$$3 = \left(\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right)$$

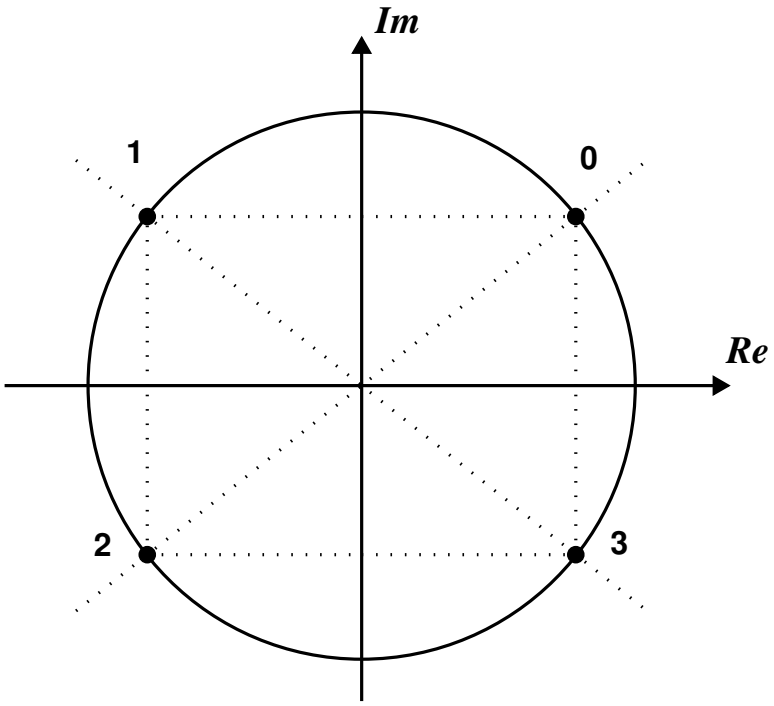


Figura 2.6: Mapeamento entre sinais e símbolos.

Capítulo 3

Algoritmo *MAP* para Códigos Turbo Quaternários

3.1 Introdução

Neste capítulo, apresentamos o algoritmo de decodificação turbo de máximo *a posteriori* - *MAP*, utilizando códigos *RSC* quaternários. Este algoritmo tem por objetivo fornecer a informação *a posteriori* ou o valor do máximo logaritmo da probabilidade *a posteriori*, fazendo uso de propriedades estatísticas e de conceitos de códigos treliça [4], [1] e [2], o qual é desenvolvido da seguinte maneira.

Inicialmente, associamos o símbolo transmitido com a transição entre os estados na treliça (ver expressão (3.2)). Em seguida, usando o fato de que os símbolos da seqüência recebida, y, são independentes entre si, expressamos a probabilidade *a posteriori* como o produto de três probabilidades: $\alpha_k(s)$, $\beta_k(s)$ e $\gamma_k(s', s)$. O próximo passo é encontrar a expressão recursiva direta, $\alpha_k(s)$, a expressão recursiva reversa, $\beta_k(s)$ e a métrica do ramo, $\gamma_k(s', s)$. Desse modo, a informação *a posteriori*, a princípio desconhecida, pode ser expressa em função de probabilidades conhecidas, tais como, a probabilidade *a priori* e a probabilidade de transição da métrica do ramo.

Finalmente, apresentamos um resumo do algoritmo que torna mais claro o uso das expressões $\alpha_k(s)$, $\beta_k(s)$ e $\gamma_k(s', s)$, no cálculo da informação *a posteriori*. O tratamento matemático que é dado a esse capítulo é de fundamental importância na implementação do algoritmo.

3.2 Fundamentos do Algoritmo

A função do algoritmo *MAP* é fornecer a informação *a posteriori* $L(u_k)$, definida da seguinte maneira. Seja

$$L(u_k) = \max(\ln(p(u_k = \theta \mid y))), \tag{3.1}$$

o máximo valor do logaritmo natural da probabilidade *a posteriori* de um símbolo decodificado ser $u_k = \theta \in \{0, 1, 2, 3\}$, dado que a seqüência de símbolos recebida é $y = y_1\ y_2\ \cdots\ y_{k-1}\ y_k\ y_{k+1}\ \cdots\ y_N$.

Portanto, nosso objetivo é expressar a probabilidade *a posteriori*, $p(u_k = \theta \mid y)$, em função de probabilidades conhecidas, tal como, a probabilidade *a priori*, $p(u_k)$, e a probabilidade de transição da métrica do ramo $p(y_k \mid x_k)$, onde x_k é a palavra código transmitida referente ao símbolo de informação u_k e y_k é a palavra código recebida referente a palavra código transmitida x_k.

Começamos com a seguinte proposição, que associa o símbolo transmitido com a transição entre os estados na treliça.

Proposição 3.2.1 *Se as transições entre o estado prévio, $S_{k-1} = s'$, e o estado presente, $S_k = s$, são mutuamente exclusivas (isto é, apenas uma delas pode ter ocorrido na treliça referente ao codificador RSC), então*

$$L(u_k) = \max\left(\ln\left(\frac{\sum\limits_{(s',s)} p(S_{k-1} = s', S_k = s, y)}{p(y)}\right)\right), \tag{3.2}$$

onde (s', s) é o conjunto de transições do estado prévio, $S_{k-1} = s'$, para o estado presente,

$S_k = s$, que pode ocorrer se o símbolo de entrada u_k for igual a θ.

Prova. Usando a regra de *Bayes*, $p(A, B) = p(A \mid B) \cdot P(B)$, e o fato de que, se os estados na treliça são conhecidos, o símbolo de entrada que causa a transição entre o estado prévio e o estado presente será conhecido. Temos, pela expressão (3.1) que:

$$\begin{aligned} L(u_k) &= \max\left(\ln\left(p(u_k = \theta \mid y)\right)\right) \\ &= \max\left(\ln\left(\sum_{(s',s)} p(S_{k-1} = s', S_k = s \mid y)\right)\right) \\ &= \max\left(\ln\left(\sum_{(s',s)} \frac{p(S_{k-1} = s', S_k = s, y)}{p(y)}\right)\right). \end{aligned}$$

■

Para simplificar a notação nas expressões, assumimos que

$$p(S_{k-1} = s', S_k = s, y) = p(s', s, y).$$

Assim, considerando na expressão (3.2) a probabilidade do numerador e usando o fato de que os símbolos da seqüência recebida,

$$y = y_1\ y_2\ \cdots\ y_{k-1}\ y_k\ y_{k+1}\ \cdots\ y_N,$$

são independentes entre si, esta seqüência y pode ser dividida em três partes:

Parte 1: a seqüência recebida antes da transição presente $y_1^{k-1} = y_1\ y_2\ \cdots\ y_{k-1}$;

Parte 2: a palavra código transmitida associada a transição presente $y_k^k = y_k$;

Parte 3: a seqüência recebida depois da transição presente $y_{k+1}^N = y_{k+1}\ y_{k+2}\ \cdots\ y_N$.

Com base na divisão da seqüência y, podemos escrever

$$p(s', s, y) = p(s', s, y_1^{k-1}, y_k, y_{k+1}^N), \tag{3.3}$$

que pode ser expressa como o produto de três probabilidades, $\bar{\alpha}_{k-1}(s')$, $\gamma(s', s)$ e $\beta_k(s)$, conforme veremos no teorema que sucede as definições seguintes:

Definição 3.2.2 *Seja*

$$\bar{\alpha}_{k-1}(s') = p(s', y_1^{k-1}), \tag{3.4}$$

a probabilidade de estar no estado prévio s' no tempo $k-1$ e ter recebido a seqüencia de canal y_1^{k-1}.

Definição 3.2.3 *Seja*

$$\gamma_k(s, s') = p(\{y_k, s\} \mid s'), \tag{3.5}$$

a probabilidade de receber a seqüência de canal y_k e estar no estado presente s, no tempo k, dado que se estava no estado prévio s', no tempo $k-1$.

Definição 3.2.4 *Seja*

$$\bar{\beta}_k(s) = p(y_{k+1}^N \mid s), \tag{3.6}$$

a probabilidade de receber a seqüência futura de canal y_{k+1}^N, dado que se está no estado presente s, no tempo k.

Teorema 3.2.5 *Se o canal é sem memória e os símbolos da seqüência recebida y são independentes entre si, então*

$$p(s', s, y) = \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \bar{\beta}_k(s). \tag{3.7}$$

Prova. Se o canal é sem memória, então a seqüência recebida y_{k+1}^N depende apenas do estado presente s e não do estado prévio s' ou das seqüências presente, y_k, e prévia, y_1^{k-1}. Desse

modo, utilizando a regra de *Bayes*, temos que:

$$\begin{aligned}
p(s', s, y) &= p(s', s, y_1^{k-1}, y_k, y_{k+1}^N) \\
&= p(y_{k+1}^N \mid s', s, y_k, y_1^{k-1}) \cdot p(s', s, y_1^{k-1}, y_k) \\
&= p(y_{k+1}^N \mid s) \cdot p(s', s, y_1^{k-1}, y_k) \\
&= p(s', y_1^{k-1}) \cdot p(\{y_k, s\} \mid s', y_1^{k-1})\} \cdot p(y_{k+1}^N \mid s) \\
&= p(s', y_1^{k-1}) \cdot p(\{y_k, s\} \mid s')\} \cdot p(y_{k+1}^N \mid s) \\
&= \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \bar{\beta}_k(s).
\end{aligned}$$

■

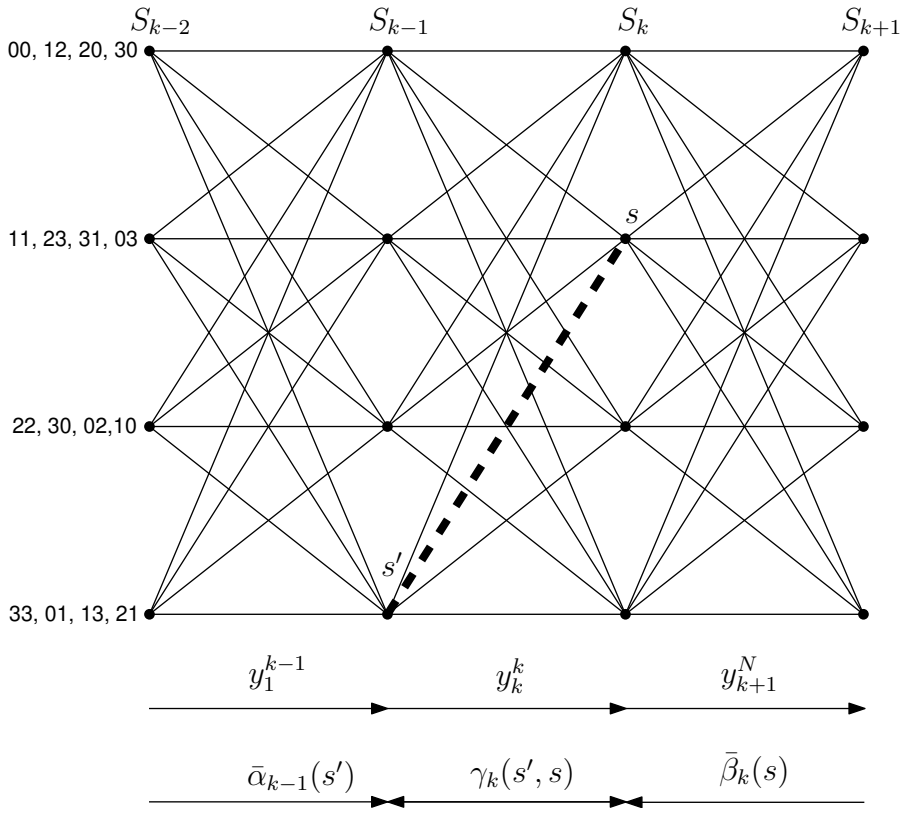


Figura 3.1: Treliça do decodificador *MAP* para o codificador *RSC* com $g(D) = \left[1 \; \frac{2+D}{1+3D}\right]$.

A Figura 3.1, onde as linhas representam as transições com os símbolos de entrada sendo $\theta \in \{0, 1, 2, 3\}$, mostra a divisão da seqüência recebida do canal y e o significado da probabi-

lidade dos três termos $\bar{\alpha}_{k-1}(s'), \gamma_k(s', s)$ e $\bar{\beta}_k(s)$, para a transição na treliça do estado prévio, $S_{k-1} = s'$, para o estado presente, $S_k = s$, conforme é mostrado pela linha em destaque, para o codificador *RSC*, cuja matriz geradora é dada por $g(D) = \left[1 \quad \frac{2+D}{1+3D}\right]$.

Finalmente, substituindo a expressão (3.7) na expressão (3.2), obtemos

$$L(u_k) = \max\left(\ln\left(\frac{\sum\limits_{(s,s')} \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \bar{\beta}_k(s)}{p(y)}\right)\right). \tag{3.8}$$

Calculamos nas seções seguintes as expressões recursivas para $\bar{\alpha}_{k-1}(s')$ e $\bar{\beta}_k(s)$.

3.3 Cálculo da Expressão Recursiva Direta $\bar{\alpha}_k(s)$

Proposição 3.3.1 *Se o canal é sem memória e os símbolos da seqüência recebida y são independentes entre si, então*

$$\bar{\alpha}_k(s) = \sum_{todo\ s'} \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', s). \tag{3.9}$$

Prova. Usando as hipóteses dadas e a regra de *Bayes*, temos que

$$\begin{aligned}
\bar{\alpha}_k(s) &= p(s, y_1^k) \\
&= \sum_{todo\ s'} p(s, s', y_1^{k-1}, y_k) \\
&= \sum_{todo\ s'} p(\{s, y_k\} \mid \{s', y_1^{k-1}\}).p\left(s', y_1^{k-1}\right) \\
&= \sum_{todo\ s'} p(\{s, y_k\} \mid s').p\left(s', y_1^{k-1}\right) \\
&= \sum_{todo\ s'} \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', s).
\end{aligned}$$

■

Supondo que a treliça começa no estado inicial $S_0 = 0$, a condição inicial para esta recursão

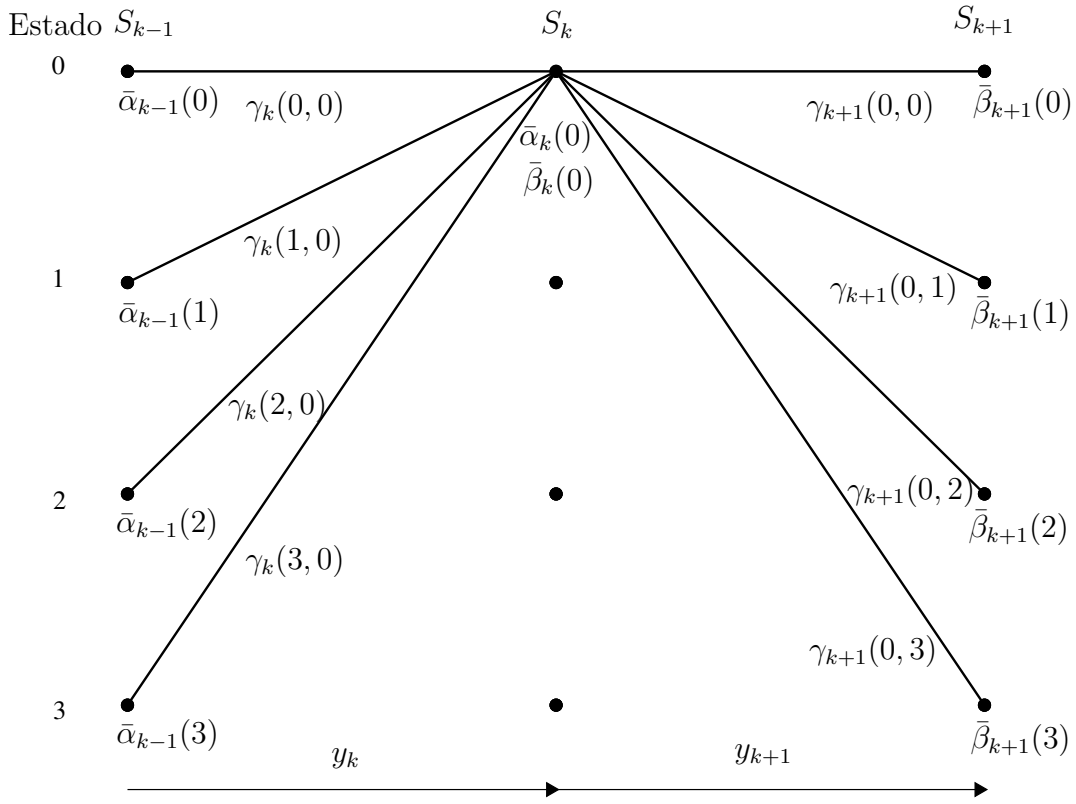


Figura 3.2: Treliça das expressões recursivas $\bar{\alpha}_k(0)$ e $\bar{\beta}_k(0)$.

é:

$$\bar{\alpha}_0(S_0 = s) = \begin{cases} 1 \text{ para } s = 0 \\ 0 \text{ para } s \neq 0. \end{cases} \tag{3.10}$$

A Figura 3.2 mostra todas as transições que chegam e partem do estado $S_k = 0$, na treliça do codificador *RSC*, cuja matriz geradora é $g(D) = \left[1 \quad \frac{2+D}{1+3D}\right]$. Diante disso, o cálculo da expressão recursiva direta $\bar{\alpha}_k(s)$, usando $\bar{\alpha}_{k-1}(s')$ e $\gamma_k(s', s)$, é:

$$\begin{aligned} \bar{\alpha}_k(0) &= \sum_{s'=0}^{3} \bar{\alpha}_{k-1}(s') \cdot \gamma_k(s', 0) \\ &= \bar{\alpha}_{k-1}(0) \cdot \gamma_k(0,0) + \bar{\alpha}_{k-1}(1) \cdot \gamma_k(1,0) + \bar{\alpha}_{k-1}(2) \cdot \gamma_k(2,0) + \bar{\alpha}_{k-1}(3) \cdot \gamma_k(3,0). \end{aligned}$$

3.4 Cálculo da Expressão Recursiva Reversa $\bar{\beta}_k(s)$

Proposição 3.4.1 *Se o canal é sem memória e os símbolos da seqüência recebida y são independentes entre si, então*

$$\bar{\beta}_{k-1}(s') = \sum_{todo\ s} \bar{\beta}_k(s) \cdot \gamma_k(s', s). \tag{3.11}$$

Prova. Similarmente,

$$\begin{aligned} \bar{\beta}_{k-1}(s') &= p\left(y_k^N, s'\right) \\ &= \sum_{todo\ s} \bar{\beta}_k(s) \cdot \gamma_k(s', s). \end{aligned}$$

■

Supondo que o codificador sempre volta ao estado $S_N = 0$, a condição inicial para esta recursão é:

$$\bar{\beta}_N(S_N = s) = \begin{cases} 1 \ \text{ para } s = 0 \\ 0 \ \text{ para } s \neq 0. \end{cases} \tag{3.12}$$

Com base na Figura 3.2, calculamos também a expressão recursiva reversa $\bar{\beta}_k(0)$, a partir de $\bar{\beta}_{k+1}(s)$ e $\gamma_{k+1}(0, s)$, do seguinte modo:

$$\begin{aligned} \bar{\beta}_k(0) &= \sum_{s=0}^{3} \bar{\beta}_{k+1}(s) \cdot \gamma_{k+1}(0, s) \\ &= \bar{\beta}_{k+1}(0) \cdot \gamma_{k+1}(0,0) + \bar{\beta}_{k+1}(1) \cdot \gamma_{k+1}(0,1) + \bar{\beta}_{k+1}(2) \cdot \gamma_{k+1}(0,2) + \bar{\beta}_{k+1}(3) \cdot \gamma_{k+1}(0,3). \end{aligned}$$

3.5 Expressões Recursivas Finais para $\alpha_k(s)$ e $\beta_k(s)$

Voltando à expressão (3.8), verificamos que, se utilizarmos o divisor $p(y)$, caminhamos para um algoritmo que não converge para o limitante de *Shannon* [15] e [7]. Pois, a cada estágio da decodificação, o decodificador que está sendo utilizado recebe a contribuição da informação

extrínseca, vinda do decodificador anterior, para dar origem à informação *a priori*. Diante disso, usamos o artifício de retirar da seqüência recebida y o elemento de ordem k. Isto é, substituímos $p(y)$ por $\dfrac{p(y)}{p(y_k)}$, dando origem ao seguinte corolário.

Corolário 3.5.1 *Se os símbolos da seqüência recebida y são independentes entre si, então*

$$\frac{p(y)}{p(y_k)} = p\left(y_1^{k-1}\right) \cdot p\left(y_{k+1}^{N} \mid y_1^{k}\right). \tag{3.13}$$

Prova. Usando a regra de *Bayes* temos que:

$$\begin{aligned} \frac{p(y)}{p(y_k)} &= \frac{p\left(y_1^{k}, y_{k+1}^{N}\right)}{p(y_k)} \\ &= \frac{p\left(y_{k+1}^{N} \mid y_1^{k}\right) \cdot p\left(y_1^{k}\right)}{p(y_k)} \\ &= p\left(y_1^{k-1}\right) \cdot p\left(y_{k+1}^{N} \mid y_1^{k}\right). \end{aligned}$$

■

Teorema 3.5.2 *Se substituirmos $p(y)$ por $\dfrac{p(y)}{p(y_k)}$, então*

$$L(u_k) = \max\left(\ln\left(\sum_{(s',s)} \alpha_{k-1}(s') \cdot \gamma_k(s',s) \cdot \beta_k(s)\right)\right), \tag{3.14}$$

onde $\alpha_k(s)$ e $\beta_k(s)$ são as novas probabilidades modificadas definidas como:

$$\begin{aligned} \alpha_k(s) &\triangleq \frac{\bar{\alpha}_k(s)}{p\left(y_1^{k}\right)}; \\ \beta_k(s) &\triangleq \frac{\bar{\beta}_k(s)}{p\left(y_{k+1}^{N} \mid y_1^{k}\right)}. \end{aligned} \tag{3.15}$$

Prova. Pelas expressões (3.8) e (3.13), temos que

$$\begin{aligned} L(u_k) &= \max\left(\ln\left(\sum_{(s',s)}\frac{\bar{\alpha}_{k-1}(s')\cdot\gamma_k\left(s',s\right)\cdot\bar{\beta}_k(s)}{p(y)}\right)\right) \\ &= \max\left(\ln\left(\sum_{(s',s)}\frac{\bar{\alpha}_{k-1}(s)\cdot\gamma_k\left(s',s\right)\cdot\bar{\beta}_k(s)\cdot p(y_k)}{p(y)}\right)\right) \\ &= \max\left(\ln\left(\sum_{(s',s)}\frac{\bar{\alpha}_{k-1}(s')}{p\left(y_1^{k-1}\right)}\cdot\gamma_k\left(s',s\right)\cdot\frac{\bar{\beta}_k(s)}{p\left(y_{k+1}^{N}\mid y_1^k\right)}\right)\right) \\ &= \max\left(\ln\left(\sum_{(s',s)}\alpha_{k-1}(s')\cdot\gamma_k\left(s',s\right)\cdot\beta_k(s)\right)\right). \end{aligned}$$

■

Desse modo, as expressões recursivas finais de $\alpha_k(s)$ e $\beta_k(s)$ são dadas pelas proposições seguintes.

Proposição 3.5.3 *Se o canal é sem memória, os símbolos da seqüência recebida y são independentes entre si, e vale a expressão*

$$p\left(y_1^k\right)=\sum_{todo\ s}\bar{\alpha}_k(s), \tag{3.16}$$

então

$$\alpha_k(s)=\frac{\sum\limits_{todo\ s'}\alpha_{k-1}(s')\cdot\gamma_k\left(s',s\right)}{\sum\limits_{todo\ s}\sum\limits_{todo\ s'}\alpha_{k-1}(s')\cdot\gamma_k\left(s',s\right)}. \tag{3.17}$$

Prova. Pela expressão (3.4), temos que $p\left(y_1^k\right) = \sum_{todo\ s} \bar{\alpha}_k(s)$; e, pelas expressões (3.15) e (3.9), segue-se que:

$$\begin{aligned}
\alpha_k(s) &= \frac{\bar{\alpha}_k(s)}{p\left(y_1^k\right)} \\
&= \frac{\bar{\alpha}_k(s)}{\sum_{todo\ s} \bar{\alpha}_k(s)} \\
&= \frac{\sum_{todo\ s'} \bar{\alpha}_{k-1}(s') \cdot \gamma_k\left(s', s\right)}{\sum_{todo\ s} \sum_{todo\ s'} \bar{\alpha}_{k-1}(s') \cdot \gamma_k\left(s', s\right)} \\
&= \frac{\sum_{todo\ s'} \alpha_{k-1}(s') \cdot \gamma_k\left(s', s\right)}{\sum_{todo\ s} \sum_{todo\ s'} \alpha_{k-1}(s') \cdot \gamma_k\left(s', s\right)}.
\end{aligned}$$

■

Proposição 3.5.4 *Se o canal é sem memória, os símbolos da seqüência recebida y são independentes entre si, e vale a expressão*

$$p\left(y_k^N \mid y_1^{k-1}\right) = \frac{p\left(y_1^k\right) \cdot p\left(y_{k+1}^N \mid y_1^k\right)}{p\left(y_1^{k-1}\right)},$$

então

$$\beta_{k-1}(s) = \frac{\sum_{todo\ s} \beta_k(s) \cdot \gamma_k(s', s)}{\sum_{todo\ s} \sum_{todo\ s'} \alpha_{k-1}(s') \cdot \gamma_k\left(s', s'\right)}. \tag{3.18}$$

Prova. Pelas expressões (3.15), (3.11), (3.16) e (3.9), e, pela expressão da hipótese dada, temos que:

$$\begin{aligned}
\beta_{k-1}(s) &= \frac{\bar{\beta}_k(s)}{p\left(y_k^N \mid y_1^{k-1}\right)} \\
&= \frac{\sum\limits_{todo\ s} \bar{\beta}_k(s) \cdot \gamma_k(s', s) \cdot p\left(y_1^{k-1}\right)}{p\left(y_1^k\right) \cdot p\left(y_{k+1}^N \mid y_1^k\right)} \\
&= \frac{\sum\limits_{todo\ s} \beta_k(s) \cdot \gamma_k(s', s) \cdot p\left(y_1^{k-1}\right) \cdot p\left(y_{k+1}^N \mid y_1^k\right)}{\sum\limits_{todo\ s} \sum\limits_{todo\ s'} \bar{\alpha}_{k-1}(s') \cdot \gamma_k\left(s', s\right) \cdot p\left(y_{k+1}^N \mid y_1^k\right)} \\
&= \frac{\sum\limits_{todo\ s} \beta_k(s) \cdot \gamma_k(s', s) \cdot p\left(y_1^{k-1}\right)}{\sum\limits_{todo\ s} \sum\limits_{todo\ s'} \alpha_{k-1}(s') \cdot \gamma_k\left(s', s\right) \cdot p\left(y_1^{k-1}\right)} \\
&= \frac{\sum\limits_{todo\ s} \beta_k(s) \cdot \gamma_k(s', s)}{\sum\limits_{todo\ s} \sum\limits_{todo\ s'} \alpha_{k-1}(s') \cdot \gamma_k\left(s', s\right)}.
\end{aligned}$$

■

Observe que $\alpha_k(s)$ e $\beta_k(s)$ são calculados recursivamente pelas equações (3.17) e (3.18) e possuem as mesmas condições iniciais que seus complementos naturais, dados em (3.10) e (3.12) para $\bar{\alpha}_k(s)$ e $\bar{\beta}_k(s)$, respectivamente.

Finalmente, calculamos agora a métrica do ramo $\gamma_k\left(s', s\right)$. Este termo é considerado a principal ferramenta no cálculo do valor do máximo logaritmo da probabilidade *a posteriori*, por ser ele o termo que possui as expressões básicas que nos permitem, não apenas calcular, mais deduzir resultados importantes sobre todo o esquema de decodificação iterativa.

3.6 Cálculo dos Valores de $\gamma_k\left(s', s\right)$

Corolário 3.6.1 *Se o canal é sem memória e os símbolos da seqüência recebida y são independentes entre si, então*

$$\gamma_k(s', s) = p(y_k \mid x_k) \cdot p(u_k). \tag{3.19}$$

Prova. Usando a regra de *Bayes* temos que:

$$\begin{aligned}
\gamma_k(s', s) &= p(s, y_k \mid s') \\
&= \frac{p(s, y_k, s')}{p(s')} \\
&= \frac{p(y_k \mid s, s') \cdot p(s', s)}{p(s')} \\
&= \frac{p(y_k \mid s, s') \cdot p(s \mid s') \cdot p(s')}{p(s')} \\
&= p(y_k \mid x_k) \cdot p(u_k),
\end{aligned}$$

onde:

u_k - símbolo de entrada necessário que causa a transição na treliça do estado prévio, s', para o estado presente, s;

x_k - palavra código transmitida associada com esta transição;

y_k - palavra código recebida associada com a palavra código transmitida x_k.

$p(u_k)$ - probabilidade *a priori* deste símbolo;

■

Assim, a probabilidade de transição, $\gamma_k(s', s)$, é dada pelo produto das probabilidades:

- $p(u_k)$ - probabilidade *a priori* do símbolo de entrada u_k necessário para a transição, derivada da saída do decodificador componente do esquema de decodificação iterativa, a ser visto no próximo capítulo.

- $p(y_k \mid x_k)$ - probabilidade de receber a seqüência de canal y_k dado que a palavra código x_k associada com esta transição foi transmitida. Esta probabilidade é derivada a partir da seqüencia de símbolos recebida, y, e dos símbolos obtidos a partir da treliça.

Desse modo, apresentamos as expressões $\alpha_{k-1}(s')$, $\beta_k(s)$ e $\gamma_k(s', s)$, necessárias para calcular a informação *a posteriori*, conforme será mostrado no resumo a seguir.

3.7 Resumo do Algoritmo *MAP*

Este resumo, conforme mostra a Figura 3.3, apresenta de forma mais clara as expressões mais importantes descritas no algoritmo MAP, usadas para calcular a informação *a posteriori* $L(u_k)$, ou ainda, para decodificar a seqüência recebida do canal y, sendo organizado da seguinte maneira:

1- A probabilidade de transição, $p(y_k \mid x_k)$, é usada juntamente com a probabilidade *a priori*, $p(u_k)$, para calcular a métrica do ramo, $\gamma_k(s', s)$, de acordo com a expressão (3.19). Em seguida, calculamos as expressões recursivas $\alpha_k(s)$ e $\beta_k(s)$.

2- A partir da métrica do ramo $\gamma_k(s', s)$, dada na expressão (3.19), e da condição inicial $\alpha_k(0)$, dada na expressão (3.10), calculamos a expressão recursiva direta $\alpha_{k-1}(s')$, dada na expressão (3.17).

3- Com o uso da métrica do ramo $\gamma_k(s', s)$, dado na expressão (3.19), da expressão recursiva direta $\alpha_{k-1}(s')$, dada na expressão (3.17), e da condição inicial $\beta_k(0)$, dada na expressão (3.12), calculamos a expressão recursiva reversa $\beta_k(s)$, dada na expressão (3.18).

4- Finalmente, com o uso das expressões $\alpha_{k-1}(s')$, $\gamma_k(s', s)$ e $\beta_k(s)$, dadas nas expressões (3.17), (3.19) e (3.18), respectivamente, calculamos a informação *a posteriori*, dada na expressão (3.14), como explicitado na Figura 3.3.

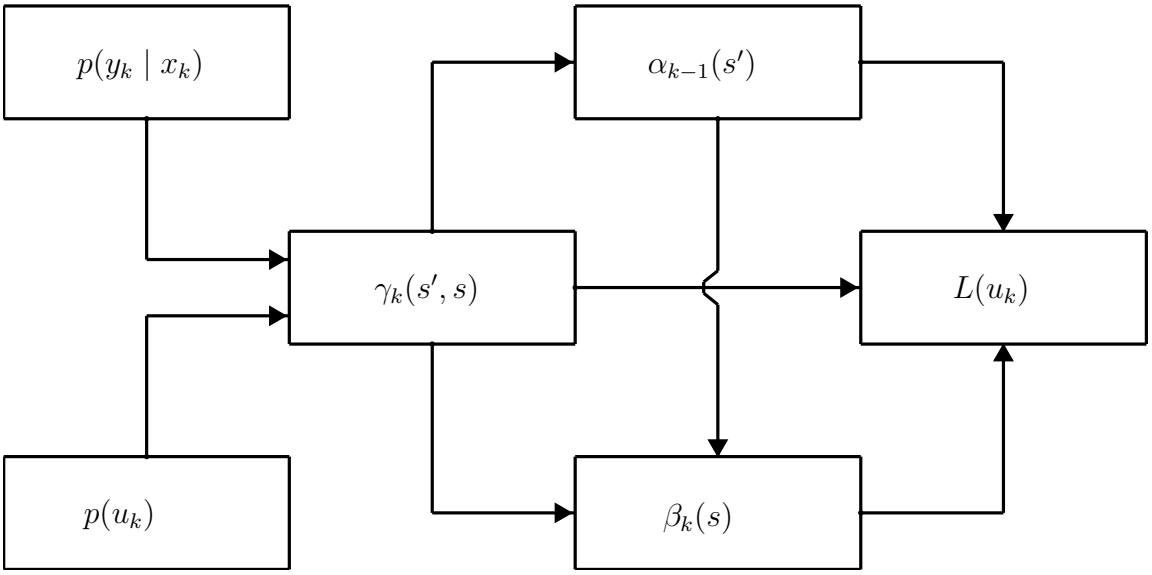


Figura 3.3: Resumo das principais expressões usadas no algoritmo *MAP*.

Capítulo 4

Decodificação Iterativa Turbo Quaternária

4.1 Introdução

Apresentamos neste capítulo o processo de decodificação iterativa turbo sobre $\mathbb{Z}_4$. Este processo de decodificação, esquematizado na Figura 4.2, é composto por dois decodificadores componentes, dois entrelaçadores e um desentrelaçador. Seu objetivo é calcular a informação extrínseca na saída do estágio de decodificação anterior (decodificador anterior), e usá-la como informação *a priori* na entrada do próximo estágio de decodificação (próximo decodificador), para reduzir a probabilidade de erro de símbolo (ou bit) à medida em que for crescendo o número de iterações.

Os conceitos de distância euclidiana e o fato de que o código convolucional é sistemático são usados, na probabilidade de transição da métrica do ramo $\gamma_k(s', s)$, para expressar a informação *a posteriori* (expressão (4.4)) como a soma da informação *a priori*, da informação sistemática ($\exp\left(L_c \cdot D_E^2(y_k^s, x_k^s)\right)$) e da informação extrínseca, calculadas em cada estágio da decodificação (cada decodificador), ver Figura 4.2.

4.2 Um Estágio da Decodificação Turbo

Nesta seção, expressamos a probabilidade de transição (função densidade de probabilidade) em função dos símbolos sistemáticos e dos símbolos de paridade e, em seguida, deduzimos as expressões de informação intrínseca, extrinseca e *a posteriori*.

Sendo x_k a palavra código transmitida e n_k o vetor ruído cujas componentes são variáveis aleatórias gaussianas com média zero e variância $\sigma^2 = \dfrac{N_0}{2}$ (ruído aditivo gaussiano branco), a palavra código recebida é dada por

$$y_k = x_k + n_k.$$

Além disso, sabendo que cada codificador RSC do esquema decodificação turbo tem taxa de codificação 1/2. A palavra código transmitida é dada por

$$x_k = x_k^s \; x_k^p = u_k \; x_k^p,$$

e a palavra código recebida é dada por

$$y_k = y_k^s \; y_k^p.$$

Ou seja, a palavra código transmitida é composta por um símbolo de informação (símbolo sistemático) e por um símbolo de paridade; e, a palavra código recebida é composta por um 'símbolo' associado ao símbolo de informação transmitido e por um 'símbolo' associado ao símbolo de paridade transmitido.

Assim, se o canal é sem memória e gaussiano, então, condicionados aos símbolos transmitidos, os símbolos recebidos também são gaussianos com função densidade de probabilidade dada por

$$p(y_k \mid x_k) \;\; = \;\; p(y_k^s \mid x_k^s) \cdot p(y_k^p \mid x_k^p), \tag{4.1}$$

onde

x_k^s - símbolo sistemático da palavra código transmitida x_k;

x_k^p - símbolo de paridade da palavra código transmitida x_k;

y_k^s - 'símbolo' recebido correspondente ao símbolo transmitido x_k^s;

y_k^p - 'símbolo' recebido correspondente ao símbolo transmitido x_k^p.

Agora, considerando a modulação *4-PSK*, os símbolos da palavra código transmitida (x_k^s e x_k^p) e os símbolos da palavra código recebida (y_k^s e y_k^p) são pontos do plano cartesiano $\mathbb{R}_2$ ($A \times B$), ou seja, são pares ordenados do tipo:

$$x_k^s = (x_{k_A}^s,\ x_{k_B}^s)$$

$$x_k^p = (x_{k_A}^p,\ x_{k_B}^p)$$

$$y_k^s = (y_{k_A}^s,\ y_{k_B}^s)$$

$$y_k^p = (y_{k_A}^p,\ y_{k_B}^p).$$

Portanto, sendo

$$\begin{aligned} D_E^2(X,\ Y) &= \parallel Y - X \parallel^2 \\ &= (y_A - x_A)^2 + (y_B - x_B)^2, \end{aligned} \tag{4.2}$$

a distância euclidiana ao quadrado entre os dois pontos $X = (x_A,\ x_B)$ e $Y = (y_A,\ y_B)$ do plano bidimensional $\mathbb{R}_2$ ($A \times B$), a função densidade de probabilidade da expressão (4.1) é dada por.

$$\begin{aligned} p(y_k \mid x_k) &= p(y_k^s \mid x_k^s) \cdot p(y_k^p \mid x_k^p) \\ &= \frac{1}{\pi N_0} \cdot \exp\left(\frac{-\parallel y_k^s - x_k^s \parallel^2}{N_0}\right) \cdot \exp\left(\frac{-\parallel y_k^p - x_k^p \parallel^2}{N_0}\right) \\ &= \frac{1}{\pi N_0} \cdot \exp\left(\frac{-\parallel (y_{k_A}^s,\ y_{k_B}^s) - (x_{k_A}^s,\ x_{k_B}^s) \parallel^2}{N_0}\right) \cdot \exp\left(\frac{-\parallel (y_{k_A}^p,\ y_{k_B}^p) - (x_{k_A}^p,\ x_{k_B}^p) \parallel^2}{N_0}\right) \\ &= \frac{1}{\pi N_0} \cdot \exp\left(-\frac{(y_{k_A}^s - x_{k_A}^s)^2 + (y_{k_B}^s - x_{k_B}^s)^2}{N_0}\right) \cdot \exp\left(-\frac{(y_{k_A}^p - x_{k_A}^p)^2 + (y_{k_B}^p - x_{k_B}^p)^2}{N_0}\right), \end{aligned}$$

onde N_0 é a variância do ruído.

Para simplificar a notação da expressão da função densidade de probabilidade usaremos a distância euclidiana ao quadrado, $D_E^2(\cdot\ ,\ \cdot)$, entre símbolos, deixando subentendido que cada símbolo corresponde a um ponto de $\mathbb{R}_2$.

Desse modo, a função densidade de probabilidade (expressão (4.1)) é dada por

$$\begin{aligned} p(y_k \mid x_k) &= \frac{1}{\pi N_0} \cdot \exp\left(\frac{- \parallel y_k^s - x_k^s \parallel^2}{N_0}\right) \cdot \exp\left(\frac{- \parallel y_k^p - x_k^p \parallel^2}{N_0}\right) \\ &= \frac{1}{\pi N_0} \cdot \exp\left(L_c \cdot D_E^2\left(y_k^s,\ u_k\right)\right) \cdot \exp\left(L_c \cdot D_E^2\left(y_k^p,\ x_k^p\right)\right), \end{aligned} \tag{4.3}$$

onde $L_c = \dfrac{-1}{N_0}$ e $u_k = x_k^s$.

A Figura 4.1 mostra as distâncias entre os possíveis símbolos transmitidos $\theta = \{0, 1, 2, 3\}$ e o símbolo recebido $r = (1.6664, -2.3332)$, ou seja,

$$\begin{aligned} 0 &\equiv \left(+\frac{1}{\sqrt{2}}, +\frac{1}{\sqrt{2}}\right) \Longrightarrow D_E^2(0, r) &&= 10.1636 \\ 1 &\equiv \left(-\frac{1}{\sqrt{2}}, +\frac{1}{\sqrt{2}}\right) \Longrightarrow D_E^2(1, r) &&= 14.8769 \\ 2 &\equiv \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right) \Longrightarrow D_E^2(2, r) &&= 8.2777 \\ 3 &\equiv \left(+\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right) \Longrightarrow D_E^2(3, r) &&= 3.5644. \end{aligned}$$

Portanto, supondo que a palavra código recebida seja, por exemplo,

$$r\ r = (1.6664, -2.3332)\ \ (1.6664, -2.3332)$$

e supondo que $r\ r$ seja a primeira palavra código a ser decodificada, podemos admitir que a palavra código decodificada (transmitida) é 3 3, pois a menor distância, $D_E^2(3, r) = 3.5644$, produz maior probabilidade de transição, $p(y_k \mid x_k)$. Isto justifica melhor o fato de que decidir pela maior probabilidade *a posteriori* $L(u_k)$ é similar a decidir pela menor distância euclidiana ao quadrado.

Aplicando os conceitos de informação intrínseca e extrínseca usados por C. Berrou, A. Glavieux e P. Thitimajashima [1] nas expressões que compõem a informação *a posteriori*,

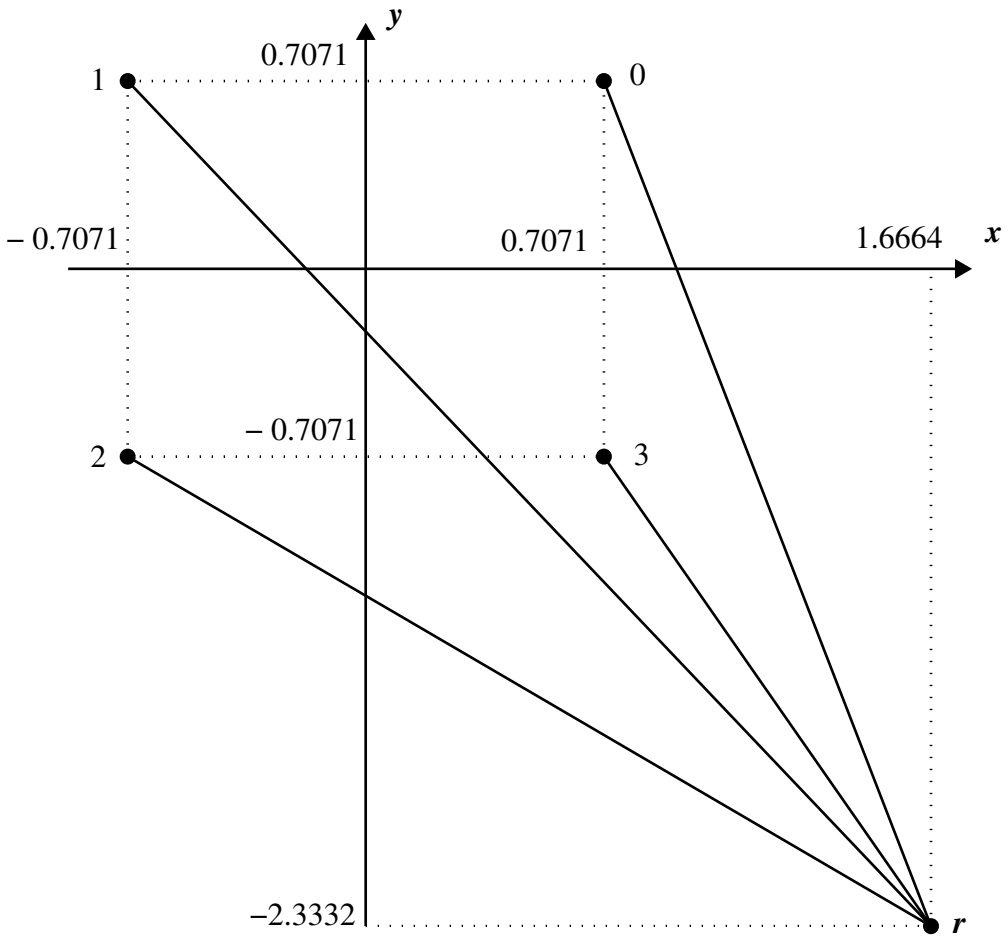


Figura 4.1: Distância euclidiana ao quadrado.

mostramos que a informação *a posteriori*, $L(u_k)$, calculada através do algoritmo MAP, será dividida em três termos:

1. $L^i(u_k)$ - informação intrínseca ou informação *a priori*;

2. $L_c \cdot D_E^2\left(y_k^s, u_k\right)$ - informação sistemática;

3. $L^e(u_k)$ - informação extrínseca, de acordo com os resultados a seguir.

Corolário 4.2.1 *Se os símbolos da seqüencia recebida y são independentes entre si e o canal é gaussiano, sem memória e com modulação* 4-PSK*, então*

$$\gamma_k(s, s') = \exp\left(L_c \cdot D_E^2\left(y_k^s, u_k\right)\right) \cdot \gamma_k^e(s, s') \cdot p(u_k),$$

onde

$$\gamma_k^e(s,s') = \frac{1}{\pi N_0} \cdot \exp\left(L_c \cdot D_E^2\left(y_k^p, x_k^p\right)\right).$$

Prova. Usando a expressão (4.3) da proposição anterior na expressão da probabilidade de transição da métrica do ramo (3.19), temos que:

$$\begin{aligned}
\gamma_k(s,s') &= p(y_k \mid x_k) \cdot p(u_k) \\
&= \exp\left(L_c \cdot D_E^2\left(y_k^s, u_k\right)\right) \cdot \frac{1}{\pi N_0} \cdot \exp\left(L_c \cdot D_E^2\left(y_k^p, x_k^p\right)\right) \cdot p(u_k) \\
&= \exp\left(L_c \cdot D_E^2\left(y_k^s, u_k\right)\right) \cdot \gamma_k^e(s,s') \cdot p(u_k).
\end{aligned}$$

■

Teorema 4.2.2 *Se valem as hipóteses do corolário anterior, então*

$$L(u_k) = L^i(u_k) + L_c \cdot D_E^2\left(y_k^s, u_k\right) + L^e(u_k), \tag{4.4}$$

onde

$$L^i(u_k) = \ln(p(u_k)),$$

é informação intrínseca, e

$$L^e(u_k) = \ln\left(\sum_{(s,s')} \alpha_{k-1}(s') \cdot \beta_k(s) \cdot \gamma_k^e(s,s')\right),$$

é a informação extrínseca.

Prova. Aplicando o resultado do corolário anterior na expressão final da informação a

posteriori (3.14), temos que

$$\begin{aligned} L(u_k) &= \max\left(\ln\left(\sum_{(s,s')}\alpha_{k-1}(s')\cdot\beta_k(s)\cdot\gamma_k(s,s')\right)\right) \\ &= \max\left(\ln\left(\sum_{(s,s')}\alpha_{k-1}(s')\cdot\beta_k(s)\cdot\exp\left(L_c\cdot D_E^2\left(y_k^s,u_k\right)\right)\cdot\gamma_k^e(s,s')\cdot p(u_k)\right)\right) \\ &= \max\left(\ln\left(\sum_{(s,s')}\alpha_{k-1}(s')\cdot\beta_k(s)\cdot\gamma_k^e(s,s')\right)+L_c\cdot D_E^2\left(y_k^s,u_k\right)+\ln\left(p(u_k)\right)\right) \\ &= \max\left(L_e(u_k)+L_c\cdot D_E^2\left(y_k^s,u_k\right)+L^i(u_k)\right). \end{aligned}$$

■

Com base no que foi apresentado poderemos expor com maiores detalhes o significado dos termos:

Informação *a priori* - referenciada também como informação intrínseca, a informação *a priori* sobre um símbolo é a informação conhecida antes de iniciar a decodificação da seqüência de símbolos recebida na entrada do decodificador;

Informação *extrínseca* - em contraste com a referenciada informação intrínseca, a informação extrínseca sobre o símbolo u_k é a informação fornecida por um decodificador, baseado na seqüência de símbolos recebida e na informação *a priori*. Ou seja, com base na expressão (4.4), a informação extrínseca é calculada do seguinte modo: informação *a posteriori*, $L\left(u_k\right)$, menos a informação sistemática, $\left(L_c\cdot D_E^2\left(y_k^s,u_k\right)\right)$, menos a informação *a priori* $L^i(u_k)$.

Informação *a posteriori* - a informação *a posteriori* sobre um símbolo é a informação que o decodificador fornece levando em conta toda informação que entra no decodificador sobre o símbolo u_k.

A partir da informação *a posteriori* $L(u_k)$, retiramos a informação extrínseca, de acordo com

a expressão (4.4), para usar no próximo estágio de decodificação [1].

4.3 Decodificação Iterativa Turbo

Na seção anterior, vimos como se realiza um estágio da decodificação, isto é, como se obtém as expressões da informação *a posteriori*, da informação *a priori* e da informação extrínseca. Nesta seção, veremos que o processo de decodificação iterativa turbo baseia-se em retirar a informação extrínseca de um estágio de decodificação anterior e usá-la como informação *a priori* no outro estágio de decodificação para que, à medida em que for crescendo o número de iterações, vá diminuindo a probabilidade de erro.

Assim, note que a Figura 4.2 organiza a seqüência recebida do canal y em duas partes:

$y^1 = y^{s1}, y^{p1}$ **- Seqüência proveniente do codificador *RSC-1*** - recebida pelo primeiro decodificador componente, contém a versão recebida dos símbolos sistemáticos, y^{s1}, e dos símbolos de paridade, y^{p1}, provenientes do primeiro codificador;

$y^2 = y^{s2}, y^{p2}$ **- Seqüência proveniente do codificador *RSC-2*** - recebida pelo segundo decodificador componente, contém a versão entrelaçada dos símbolos sistemáticos, y^{s2}, e dos símbolos de paridade, y^{p2}, provenientes do segundo codificador.

Processo Iterativo: considere inicialmente o primeiro decodificador componente na primeira iteração. Este decodificador recebe a seqüência de canal, y^1, e produz uma estimativa da informação *a posteriori* $L_{11}(u_k)$ dos símbolos de dado u_k, onde $k \in \{1, \cdots, N\}$ e N é o comprimento da seqüência de informação. Note que o subscrito 11 de $L_{11}(u_k)$ indica que estamos na primeira iteração e no primeiro decodificador e, ainda, nesta primeira iteração a informação *a priori* que o primeiro decodificador componente recebe é $\ln(p(u_k = \theta)) = \ln(1/4)$.

[1] Se a decodificação ocorreu no primeiro decodificador, basta entrelaçar a informação extrínseca, obtida na saída deste decodificador, para obter a informação *a priori* a ser usada no segundo decodificador. Entretanto, se a decodificação ocorreu no segundo decodificador, devemos desentrelaçar a informação extrínseca, obtida na saída deste decodificador, para obter a informação *a priori* a ser usada no primeiro decodificador.

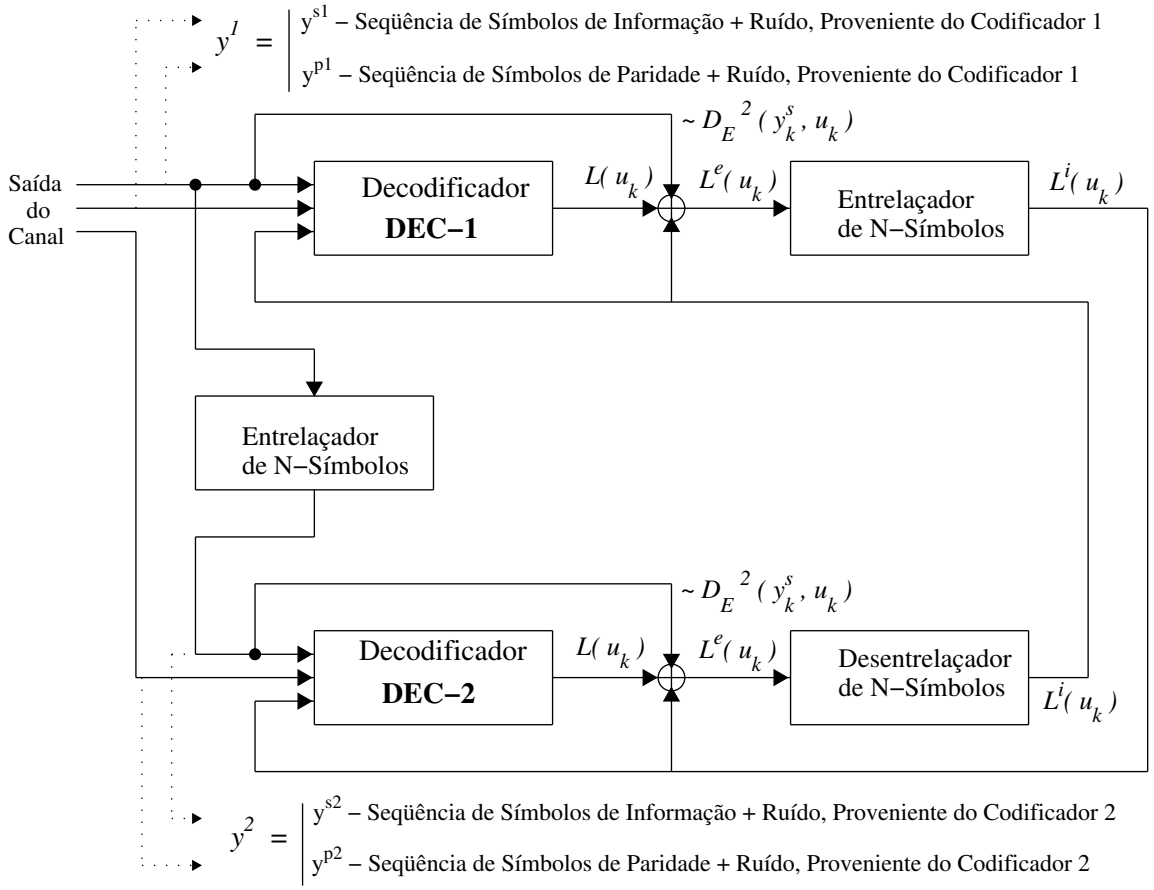


Figura 4.2: Esquema de decodificação turbo.

O segundo decodificador componente recebe a seqüência de canal y^2 junto com a informação extrínseca entrelaçada do primeiro decodificador componente e fornece uma estimativa da informação *a posteriori* $L_{12}(u_k)$ dos símbolos de dado u_k, onde o subscrito 12 de $L_{12}(u_k)$ indica que estamos na primeira iteração e no segundo decodificador componente. Note que a informação extrínseca entrelaçada, utilizada pelo segundo decodificador componente é, na verdade, a informação *a priori* obtida com o entrelaçamento da expressão: informação *a posteriori*, $L_{11}(u_k)$, menos informação *a priori*, $\ln(p(u_k = \theta)) = \ln(1/4)$, menos a informação sistemática, $L_c \cdot D_E^2\,(y_k^s, u_k)$, todas do primeiro decodificador componente. Esta informação *a priori*[2] (usada no segundo decodificador) é calculada através da expressão (4.4) na saída do primeiro decodifi-

[2]A informação extrínseca calculada na saída do primeiro decodificador, após ser entrelaçada, é usada como informação *a priori* na entrada do segundo decodificador. E de forma análoga, a informação extrínseca calculada na saída do segundo decodificador, após ser desentrelaçada, é usada como informação *a priori* na entrada do primeiro decodificador e assim por diante.

cador componente.

Na segunda iteração, o primeiro decodificador componente novamente processa sua seqüência recebida de canal y^1. Mas, dessa vez, ele também possui a informação *a priori* $L^i(u_k)$ fornecida pela porção extrínseca $L^e(u_k)$ da informação *a posteriori* $L_{12}(u_k)$ calculada pelo segundo decodificador componente, na primeira iteração. Portanto, ele pode produzir uma melhor informação *a posteriori* $L_{21}(u_k)$. Note que $L_{21}(u_k)$ é a informação *a posteriori* na segunda iteração e no primeiro decodificador componente.

A segunda iteração continua no segundo decodificador componente usando uma melhor informação *a posteriori* $L_{21}(u_k)$ do primeiro decodificador para obter, através da expressão (4.4), a melhor informação *a priori* $L^i(u_k)$ que será usada junto com a seqüência recebida de canal y^2 para calcular $L_{22}(u_k)$.

Este processo iterativo continua e, a cada iteração, em média a taxa de erro de bit diminui para uma mesma relação sinal ruído - SNR (*signal to noise ratio*).

O exemplo apresentado aqui mostra a importância do processo iterativo, ou seja, como a confiabilidade sobre o símbolo decodificado melhora a cada passo da decodificação, ou como a taxa de erro de bit versus E_b/N_0 diminui a cada iteração.

Exemplo: considere a seqüência de informação que possui $N = 100$ símbolos distribuídos através das linhas da Tabela 4.1. Esta seqüência, depois de ser processada pelo esquema de codificação da Figura 2.1 (para o codificador RSC com matriz geradora $g(D) = \left[1 \; \frac{2+D+2D^2}{1+D+3D^2}\right]$ e taxa de codificação turbo 1/2), passa pelo canal que introduz erros devido ao ruído e produz a seqüência de informação da Tabela 4.2 na saída do primeiro decodificador componente do esquema de decodificação turbo da Figura 4.2. A nova seqüência apresentada na Tabela 4.2, para o primeiro decodificador componente e na primeira iteração, contém 23 erros entre os 100 símbolos decodificados e estes erros estão destacados conforme a Tabela referenciada. Já a seqüência da Tabela 4.3, obtida na saída do segundo decodificador componente e na primeira iteração, contém 7 erros a menos que a seqüência apresentada na Tabela 4.2.

Na segunda iteração, as seqüências das Tabelas 4.4 e 4.5, obtidas nas saídas do primeiro e do segundo decodificador componente, contêm 15 erros e 11 erros, respectivamente.

Finalmente, a Tabela 4.6 apresenta a seqüência que contém apenas 1 erro em 100 símbolos decodificados, na saída do primeiro decodificador componente e na terceira iteração. Enquanto a Tabela 4.7, apresenta a seqüência com 0 (zero) erro para 100 símbolos decodificados na saída do segundo decodificador componente e na terceira iteração.

Assim, o processo de decodificação iterativa torna o algoritmo de decodificação *MAP* bastante eficiente na correção de erros dos símbolos decodificados, proporcionando, em média, uma diminuição na probabilidade de erro de símbolo (ou *bit*) a cada iteração, ver Figura 4.3, que apresenta os pontos referentes à taxa de erro de bit em $1.1735e-001$, $5.6122e-002$ e $5.1020e-003$ versus $E_b/N_0 = 1.25\ \ dB$ correspondentes à primeira, segunda e terceira iteração, respectivamente.

Portanto, o processo de decodificação iterativa, que é considerado a principal ferramenta do algoritmo de decodificação turbo, contribui para que este algoritmo seja considerado um dos mais eficientes na correção de erros para uma baixa relação sinal ruído, como pode ser comprovado a partir dos resultados apresentados no capítulo seguinte.

Seqüência de informação a ser transmitida																			
1	2	1	2	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	3
3	1	3	1	2	2	0	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	1	2	1	1	1	3	2	1	2	1	1	2	3	3	1	2	1	0
2	1	2	0	1	2	2	3	1	1	2	3	1	0	0	1	1	3	1	2
1	3	2	2	1	3	2	2	2	1	0	1	1	0	0	2	0	2	3	0

Tabela 4.1: Seqüência de informação de comprimento $N = 100$ símbolos.

Primeiro decodificador - primeira iteração																			
1	2	1	③	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	①
②	1	3	1	2	2	①	3	2	2	⓪	2	②	1	⓪	1	0	0	0	⓪
3	2	②	2	1	1	⓪	②	2	1	2	1	1	①	3	⓪	⓪	2	1	0
2	1	③	0	1	2	2	3	②	1	2	3	②	0	①	1	1	②	③	2
1	3	2	2	1	3	2	2	2	1	0	⓪	③	0	0	2	①	2	3	0

Tabela 4.2: 100 Símbolos decodificados - 23 erros ocorridos.

Segundo decodificador - primeira iteração																			
1	2	1	③	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	①
②	1	3	1	2	2	①	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	②	2	1	1	⓪	3	2	1	2	1	1	2	3	3	⓪	2	1	0
2	1	③	0	1	2	2	3	1	1	2	3	1	③	①	⓪	⓪	②	1	2
1	3	2	2	1	3	2	2	2	1	0	⓪	③	0	0	2	①	2	3	0

Tabela 4.3: 100 Símbolos decodificados - 16 erros ocorridos.

Primeiro decodificador - segunda iteração																			
②	2	1	③	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	①
②	③	②	1	①	2	②	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	1	2	1	1	1	3	2	1	2	1	1	2	3	3	1	2	1	0
2	1	2	0	1	2	2	3	1	1	2	3	1	③	①	⓪	1	②	1	2
1	3	2	2	1	3	2	2	2	1	0	⓪	③	0	0	2	①	2	3	0

Tabela 4.4: 100 Símbolos decodificados - 15 erros ocorridos.

Segundo decodificador - segunda iteração																			
1	2	1	③	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	①
②	③	3	1	①	2	0	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	1	2	1	1	1	3	2	1	2	1	1	2	3	3	1	2	1	0
2	1	2	0	1	2	2	3	1	1	2	3	1	③	③	⓪	1	3	1	2
1	3	2	2	③	3	2	2	2	1	0	⓪	③	0	0	2	0	2	3	0

Tabela 4.5: 100 Símbolos decodificados - 11 erros ocorridos.

Primeiro decodificador - terceira iteração																			
1	2	1	2	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	3
3	1	3	1	①	2	0	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	1	2	1	1	1	3	2	1	2	1	1	2	3	3	1	2	1	0
2	1	2	0	1	2	2	3	1	1	2	3	1	0	0	1	1	3	1	2
1	3	2	2	1	3	2	2	2	1	0	1	1	0	0	2	0	2	3	0

Tabela 4.6: 100 Símbolos decodificados - 1 erro ocorrido.

Segundo decodificador - terceira iteração																			
1	2	1	2	3	1	0	2	3	1	2	2	2	3	2	0	1	1	3	3
3	1	3	1	2	2	0	3	2	2	1	2	1	1	3	1	0	0	0	3
3	2	1	2	1	1	1	3	2	1	2	1	1	2	3	3	1	2	1	0
2	1	2	0	1	2	2	3	1	1	2	3	1	0	0	1	1	3	1	2
1	3	2	2	1	3	2	2	2	1	0	1	1	0	0	2	0	2	3	0

Tabela 4.7: 100 Símbolos decodificados - 0 erro ocorrido.

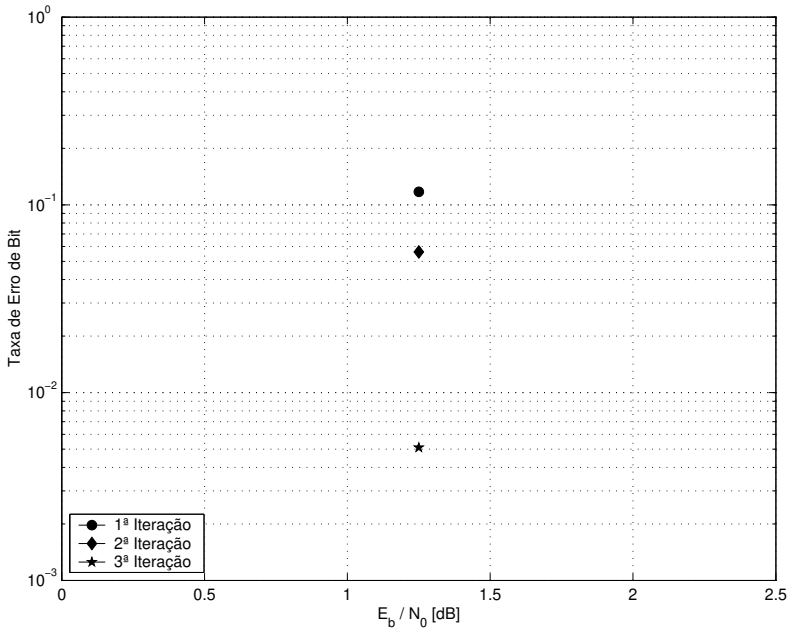


Figura 4.3: Efeito da variação do número de iterações para $N = 100$ *bits*, taxa de codificação turbo 1/2 e $g(D) = \left[1 \quad \dfrac{2 + D + 2D^2}{1 + D + 3D^2}\right]$, sobre a curva de $p_b(e) \times E_b/N_0$.

Capítulo 5

Resultados e Conclusões

5.1 Introdução

Neste capítulo, apresentamos as curvas da taxa de erro de símbolo versus E_b/N_0 referentes as simulações realizadas e também as conclusões e sugestões para trabalhos futuros dentro da classe dos códigos turbo quaternários. As simulações foram realizadas sobre o esquema turbo proposto, considerando-se a variação de alguns parâmetros, tais como: o comprimento da seqüência de informação, a taxa do codificador turbo, o codificador *RSC* e o número de iterações. A variação destes parâmetros está diretamente associada com o desempenho dos códigos turbo.

5.2 Resultados das Simulações

Inicialmente, organizamos os componentes do esquema turbo, analisando a contribuição dada por cada um deles para as simulações, e, em seguida, discutimos os resultados simulados. Os componentes do esquema turbo serão organizados conforme a seguir.

Os codificadores convolucionais constituintes, que a princípio têm a função de introduzir símbolos de redundância no sistema para que o esquema de decodificação seja capaz de corrigir eventuais erros ocorridos durante a transmissão, são codificadores *RSC* idênticos, com taxa de

codificação igual a 1/2, com matriz geradora do tipo $g(D) = \left[1 \ \frac{g_2(D)}{g_1(D)}\right]$, e organizados conforme o esquema de codificação turbo mostrado na Figura 2.1.

O entrelaçador, usado na concatenação em paralelo entre os dois codificadores *RSC-1* e *RSC-2* do esquema de codificação da Figura 2.1, evita que o erro ocorrido em um dado símbolo, associado ao codificador *RSC-1*, ocorra neste mesmo símbolo, associado ao codificador *RSC-2*. Desse modo, nota-se que a presença do entrelaçador no processo de codificação fornecerá maior confiabilidade na decodificação. E, ainda, com base no estudo feito sobre os vários tipos de entrelaçadores, escolhemos, para nossas simulações, o entrelaçador s-aleatório[1], pois como já observado na Figura 2.5, ele apresenta melhores propriedades aleatórias que os outros entrelaçadores estudados.

O Puncionador, serve para aumentar a taxa de codificação do codificador turbo. Assim, o codificador turbo terá taxa 1/2 quando usado o puncionamento, e terá taxa 1/3 quando não usado o puncionamento.

O canal é *AWGN* e a *modulação* é *4-PSK*. O fato de o canal ser *AWGN*, usado junto com o fato do codificador ser *RSC*, permite-nos obter a informação extrínseca a partir da informação *a posteriori*. Isto é, permite-nos construir o processo de decodificação iterativa, considerado a ferramenta mais importante do esquema turbo.

Os decodificadores, são decodificadores máximo *a posteriori*, MAP, e estão organizados de acordo com o esquema de decodificação iterativa turbo quaternário da Figura 4.2. Estes decodificadores atuam de forma cíclica. Isto é, inicialmente, o primeiro decodificador fornece a informação extrínseca que é usada no segundo decodificador, então, este segundo decodificador produz uma nova informação extrínseca que será usada no primeiro decodificador e assim por diante. Este procedimento é chamado de processo iterativo e a cada iteração ele fornece, em média, uma melhor informação *a posteriori* sobre o símbolo decodificado.

Assim, a contribuição dada por cada um destes componentes faz com que o esquema turbo

[1] O entrelaçador s-aleatório tem a vantagem de trabalhar com seqüências de quaisquer comprimento N. Desse modo, podemos utilizar, no esquema turbo proposto, seqüências de informação de quaisquer comprimento N.

seja considerado um dos mais eficientes esquemas de decodificação para a correção de erros. Desse modo, apresentamos agora os resultados (ou as curvas) obtidos por simulações, a partir da variação de parâmetros, tais como: o número de iterações, o comprimento da seqüência de informação, o codificador *RSC*, e a taxa de codificação do código turbo. Estes parâmetros, organizados de acordo com a Tabela 5.1, estão diretamente associados ao desempenho do esquema turbo.

Codificador	C. S. Informação	Taxa Turbo	Figuras
RSC-**A** 4 Estados	$N = 5000$ *Símbolos*	1/2	5.4, 5.8, 5.10 e 5.11
	$N = 10000$ *Símbolos*	1/2	5.4 e 5.9
	$N = 5000$ *Símbolos*	1/3	5.5 e 5.8
	$N = 10000$ *Símbolos*	1/3	5.5 e 5.9
RSC- **B** 16 Estados	$N = 500$ *Símbolos*	1/2	5.2 e 5.6
	$N = 2000$ *Símbolos*	1/2	5.2 e 5.7
	$N = 5000$ *Símbolos*	1/2	5.1, 5.2, 5.10 e 5.11
	$N = 20000$ *Símbolos*	1/2	5.12
	$N = 500$ *Símbolos*	1/3	5.3 e 5.6
	$N = 2000$ *Símbolos*	1/3	5.3 e 5.7
8 Estados	$N = 5000$ *Símbolos*	1/2	5.11

$\longrightarrow$ Codificador *RSC*- **A** com matriz geradora $g(D) = \left[1 \ \frac{2+D}{1+3D}\right]$

$\longrightarrow$ Codificador *RSC*- **B** com matriz geradora $g(D) = \left[1 \ \frac{2+D+2D^2}{1+D+3D^2}\right]$

Tabela 5.1: Parâmetros usados nas simulações.

5.2.1 Análises do Processo Iterativo ou do Número de Iterações

A Figura 5.1 apresenta as curvas das taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: taxa de codificação turbo igual a 1/2, comprimento da seqüência de informação, $N = 5000$ *símbolos*, codificador *RSC*-**B**, e o número de iterações variando de 1 até 15.

Observando as curvas relacionadas à décima quinta, sexta, quarta, segunda e primeira iteração, estas curvas alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 0.81\ dB$, $E_b/N_0 \simeq 1.0\ dB$, $E_b/N_0 \simeq 1.3\ dB$, $E_b/N_0 \simeq 2.2\ dB$ e $E_b/N_0 \simeq 3.8\ dB$, respectivamente. Desse

modo, a curva da décima quinta iteração para taxa de erro de símbolo de 10^{-4} tem um ganho[2] aproximado de 0.19 dB, 0.5 dB, 1.4 dB e 3.0 dB em relação a sexta, quarta, segunda e primeira iteração, respectivamente.

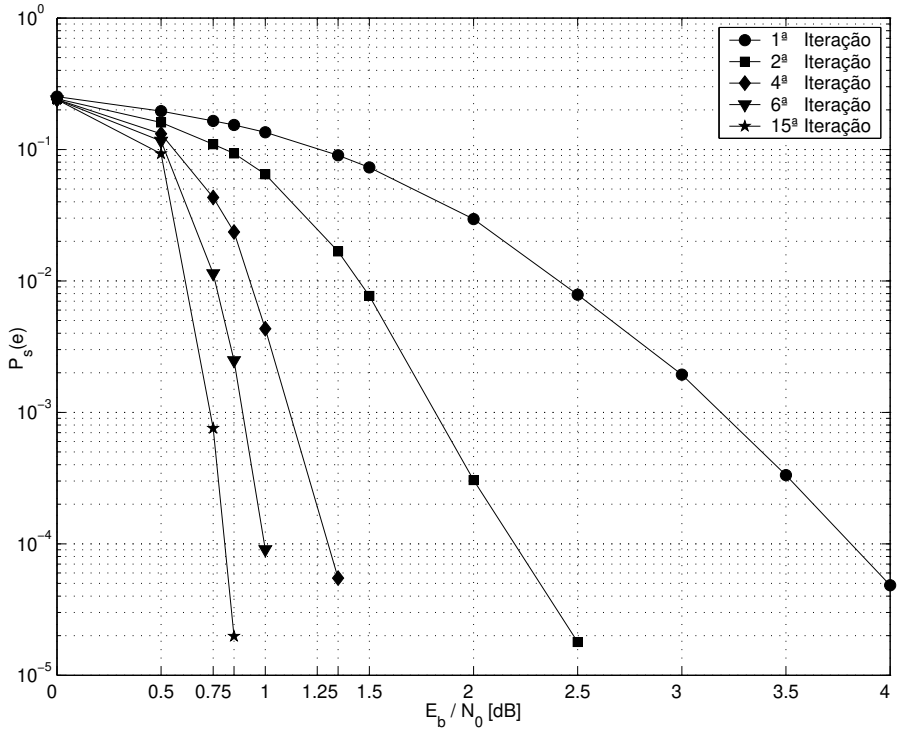


Figura 5.1: Efeito da variação do número de iterações para codificador RSC-**B**, taxa de codificação turbo 1/2 e $N = 5000$ símbolos, sobre a curva de $p_s(e) \times E_b/N_0$.

Desse modo, podemos perceber que o processo iterativo proporciona: um ganho bastante acentuado da primeira até a quarta iteração, um ganho menos acentuado da quarta até a sexta iteração e um ganho pequeno da sexta até a décima quinta iteração. Verificamos também que o processo iterativo produz um ganho de aproximadamente 3.0 dB entre a primeira e a décima quinta iteração para o codificador RSC-**B**, sendo assim considerado a ferramenta mais importante do esquema turbo.

[2]Denotamos como ganho o valor absoluto da diferença entre as duas razões $E_b/N_0(1)$ e $E_b/N_0(2)$, calculada para a mesma taxa de erro de símbolo.

5.2.2 Análises do Comprimento da Seqüência de Informação

A Figura 5.2 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador RSC-**B**, taxa de codificação turbo igual a 1/2, 15 iterações e comprimentos das seqüências de informação $N = 500$ símbolos, $N = 2000$ símbolos e $N = 5000$ símbolos.

Para os comprimentos das seqüências de informação $N = 5000$ símbolos, $N = 2000$ símbolos e $N = 500$ símbolos as curvas referentes a estes comprimentos alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 0.81\ dB$, $E_b/N_0 \simeq 1.01\ dB$ e $E_b/N_0 \simeq 1.46\ dB$, respectivamente. Ou seja, para taxa de erro de símbolo de 10^{-4}, a curva referente ao comprimento $N = 5000$ símbolos apresenta um ganho de aproximadamente 0.20 dB em relação à curva referente ao comprimento $N = 2000$ símbolos e um ganho de aproximadamente 0.65 dB em relação à curva referente ao comprimento $N = 500$ símbolos.

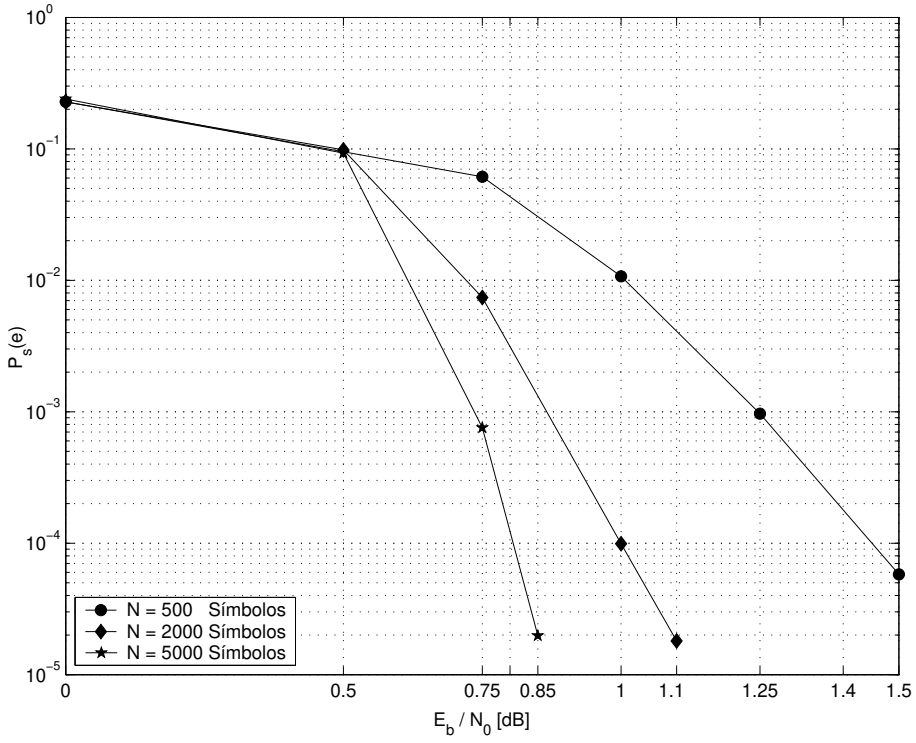


Figura 5.2: Efeito da variação do comprimento da seqüência de informação para taxa de codificação turbo 1/2, codificador RSC-**B** e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.3 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador RSC-**B**, taxa de codificação turbo igual a 1/3, 15 iterações e comprimentos das seqüências de informação $N = 2000$ símbolos e $N = 500$ símbolos.

Note que, para os comprimentos da seqüência de informação $N = 2000$ símbolos e $N = 500$ símbolos, as curvas referentes a estes comprimentos alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 0.36\ dB$ e $E_b/N_0 \simeq 0.77\ dB$, respectivamente. Ou seja, para taxa de erro de símbolo de 10^{-4}, a curva referente ao comprimento $N = 2000$ símbolos apresenta um ganho de aproximadamente 0.41 dB em relação à curva referente ao comprimento $N = 500$ símbolos.

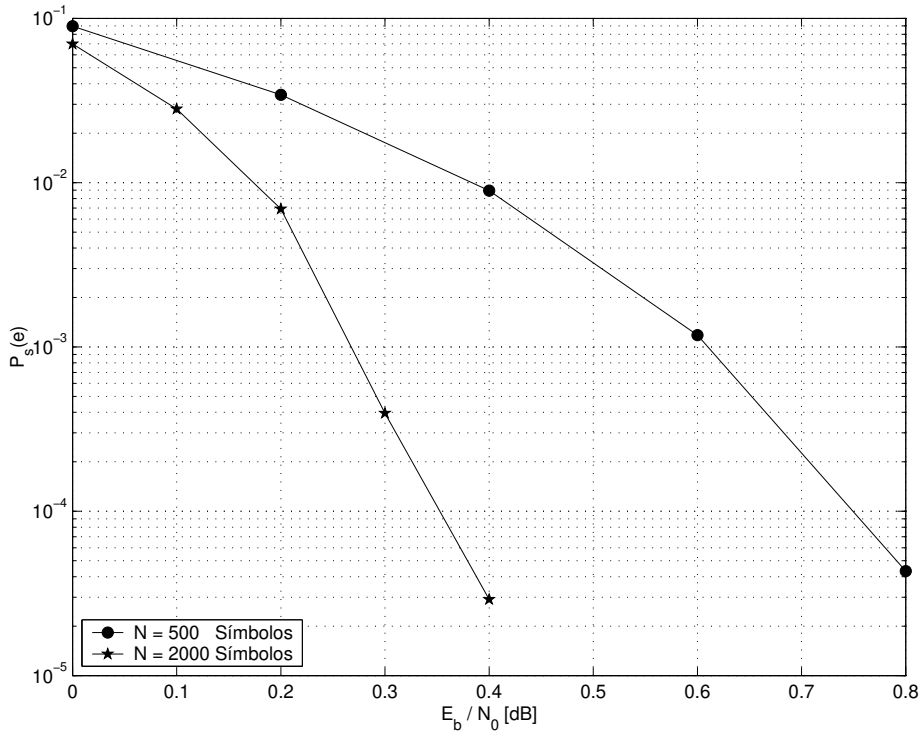


Figura 5.3: Efeito da variação do comprimento da seqüência de informação para taxa de codificação turbo 1/3, codificador RSC-**B** e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.4 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador RSC-**A**, taxa de codificação turbo igual a 1/2, décima quinta iteração e comprimentos das seqüências de informação $N = 10000$ símbolos e $N = 5000$ símbolos.

Note que para os comprimentos da seqüência de informação $N = 10000$ símbolos e $N = 5000$

símbolos as curvas referentes a estes comprimentos alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 1.81\ dB$ e $E_b/N_0 \simeq 1.98\ dB$, respectivamente. Ou seja, para a taxa de erro de símbolo de 10^{-4}, a curva referente ao comprimento $N = 10000$ símbolos, apresenta um ganho de aproximadamente $0.17\ dB$ em relação à curva referente ao comprimento $N = 5000$ símbolos.

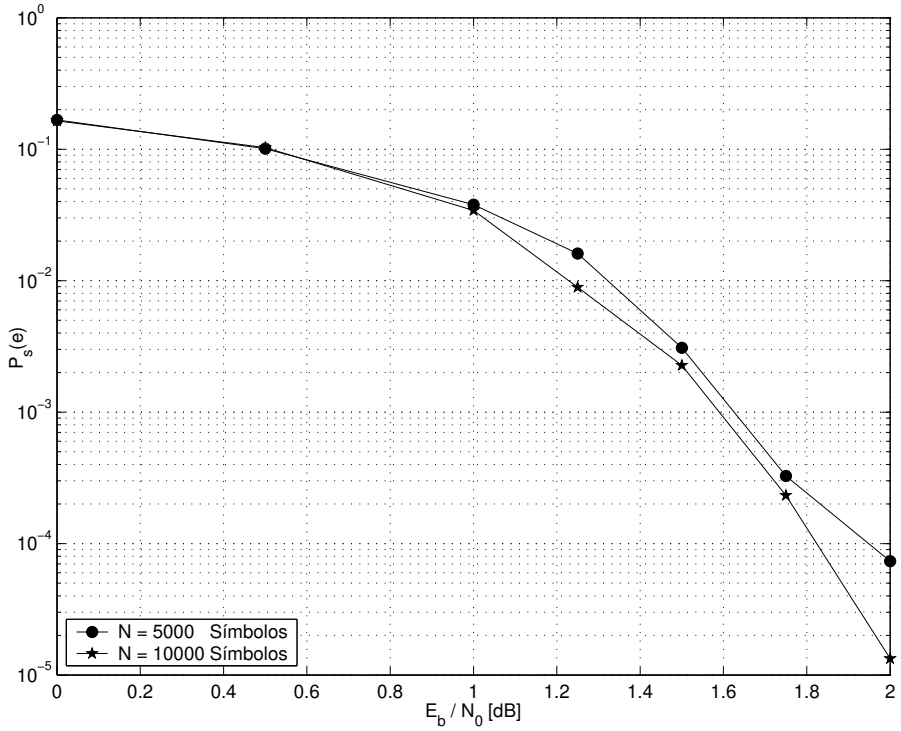


Figura 5.4: Efeito da variação do comprimento da seqüência de informação para codificador *RSC*-**A**, taxa de codificação turbo 1/2 e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.5 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador *RSC*-**A**, taxa de codificação turbo igual a 1/3, décima quinta iteração e comprimentos das seqüências de informação $N = 5000$ símbolos e $N = 10000$ símbolos.

Note que para os comprimentos da seqüência de informação $N = 5000$ símbolos e $N = 10000$ símbolos as curvas referentes a estes comprimentos alcançam a taxa de erro de símbolo em 10^{-4} para $E_b/N_0 \simeq 1.25\ dB$ e $E_b/N_0 \simeq 1.1\ dB$, respectivamente. Ou seja, para a taxa de erro de símbolo de 10^{-4}, a curva referente ao comprimento $N = 10000$ símbolos apresenta um ganho de aproximadamente $0.15\ dB$ em relação à curva referente ao comprimento $N = 5000$ símbolos.

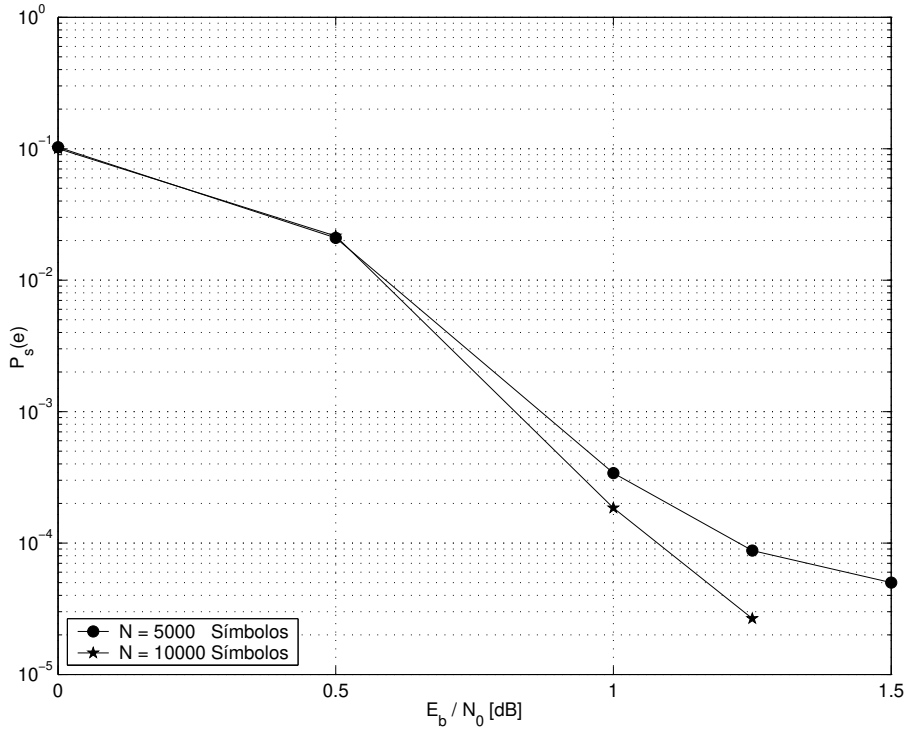


Figura 5.5: Efeito da variação do comprimento da seqüência de informação para codificador *RSC*-**A**, taxa de codificação turbo 1/3 e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

Portanto, as quatro figuras anteriores (5.2, 5.3, 5.4 e 5.5) mostram que aumentando-se o comprimento da seqüência de informação temos um ganho em termos de E_b/N_0 entre as curvas[3]. Por exemplo, entre as curvas apresentadas na Figura 5.2, a curva correspondente ao comprimento $N = 5000$ símbolos apresenta um ganho de aproximadamente 0.65 dB em relação a curva correspondente ao comprimento $N = 500$ símbolos. Logo, podemos admitir que o aumento do comprimento da seqüência de informação deve ser considerado um elemento importante do esquema turbo.

[3]Estima-se que a partir de um determinado valor de N (N grande) não temos ganho significativo entre duas curvas consecutivas , pois o ganho entre curvas consecutivas vai decrescendo com o crescimento do comprimento da seqüência de informação, ver Figura 5.2.

5.2.3 Análises da Variação da Taxa de Codificação Turbo ou Uso do Puncionador

A Figura 5.6 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador RSC-**B**, comprimento da seqüência de informação, $N = 500$ símbolos, 15 iterações e taxas de codificação turbo 1/2 e 1/3.

Como era de se esperar, as curvas referentes às taxas de codificação turbo 1/2 e 1/3 alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 1.46\ dB$ e $E_b/N_0 \simeq 0.75\ dB$, respectivamente. Isto é, a curva cuja taxa de codificação turbo é 1/3 apresenta um ganho de aproximadamente $0.71\ dB$ em relação à curva cuja taxa de codificação turbo é 1/2, para taxa de erro de símbolo de 10^{-4}.

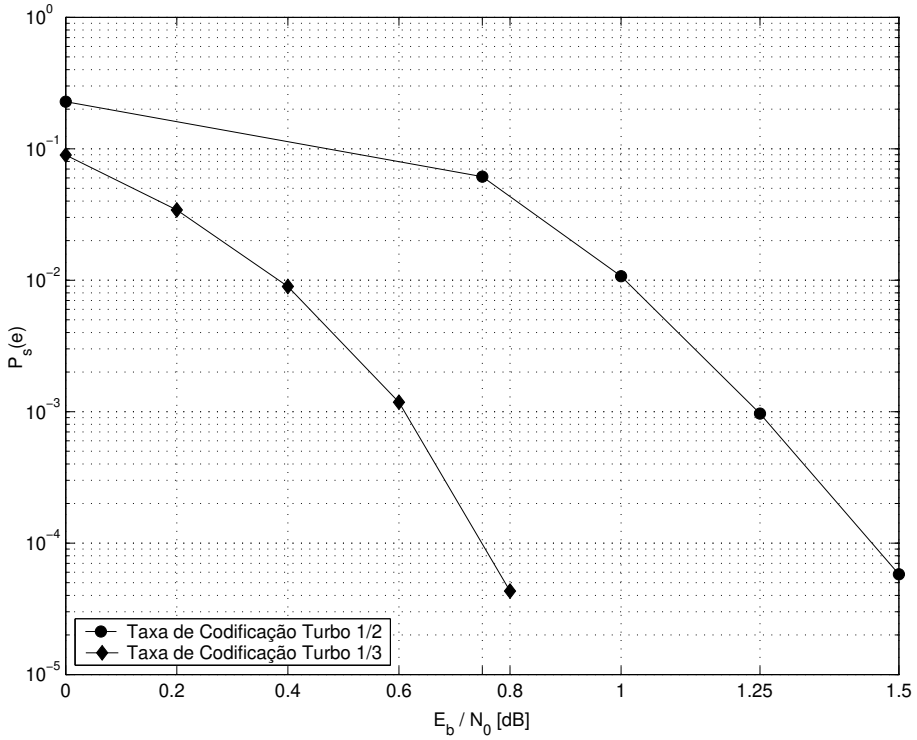


Figura 5.6: Efeito da variação da taxa de codificação turbo para codificador RSC-**B**, comprimento da seqüência de informação $N = 500$ símbolos e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.7 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador RSC-**B**, comprimento da seqüência de informação, $N = 2000$

símbolos, décima quinta iteração e taxas de codificação turbo 1/2 e 1/3.

As curvas referentes às taxas de codificação turbo 1/2 e 1/3 alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 1.01\ dB$ e $E_b/N_0 \simeq 0.36\ dB$, respectivamente. Isto é, a curva cuja taxa de codificação turbo é 1/3 apresenta um ganho de aproximadamente 0.65 dB em relação à curva cuja taxa de codificação turbo é 1/2, para taxa de erro de símbolo de 10^{-4}.

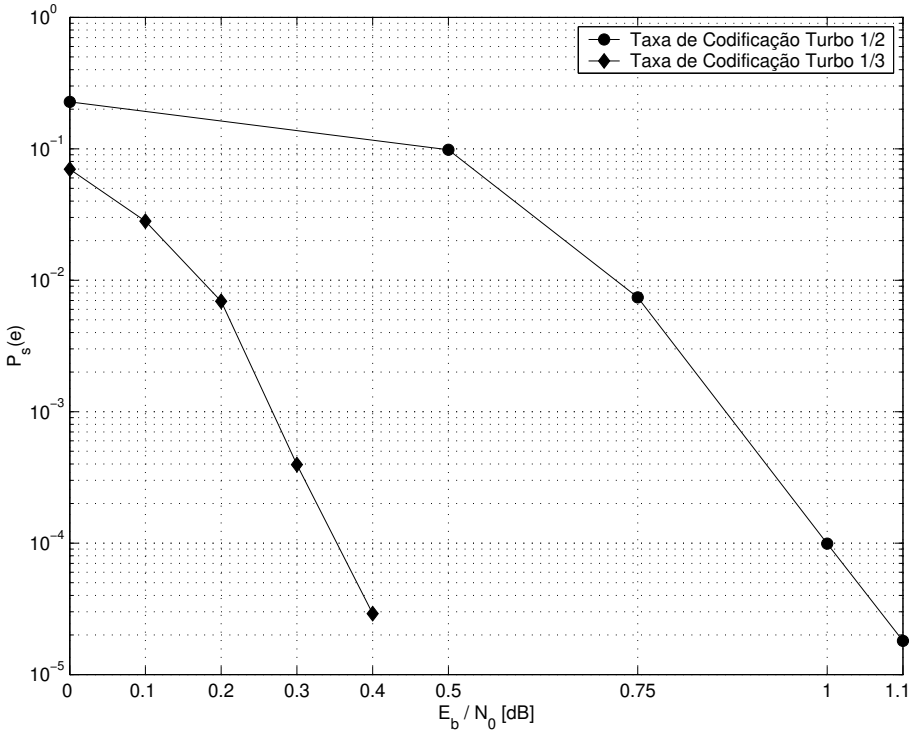


Figura 5.7: Efeito da variação da taxa de codificação turbo para codificador *RSC*-**B**, comprimento da seqüência de informação $N = 2000$ símbolos e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.8 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador *RSC*-**A**, taxas de codificação turbo igual a 1/3 e 1/2, décima quinta iteração e comprimento da seqüência de informação $N = 5000$ símbolos.

As curvas referentes as taxas de codificação turbo 1/2 e 1/3 alcançam a taxa de erro de símbolo em 10^{-4} para $E_b/N_0 \simeq 1.95\ dB$ e $E_b/N_0 \simeq 1.25\ dB$, respectivamente. Ou seja, para a taxa de erro de símbolo de 10^{-4}, a curva referente à taxa de codificação turbo 1/3, apresenta um ganho de aproximadamente 0.70 dB em relação à curva referente à taxa de codificação turbo

1/2.

Na Figura 5.8 nota-se que existe uma saturação das curvas (efeito "floor"(piso)) com o aumento de E_b/N_0, ou seja, para cada curva existe um valor de E_b/N_0 a partir do qual a curva satura, e, este valor de E_b/N_0 para o qual a curva satura varia de acordo com os parâmetros (comprimento da seqüencia de informação, taxa de codificação turbo, número de iteração e codificador *RSC*) utilizados no esquema turbo. Em outras palavras, quanto melhor for o esquema turbo menor é o valor de E_b/N_0 para o qual a curva satura, e, ainda, menor é o valor da taxa de erro de símbolo em que a curva satura, ver Figura 5.8

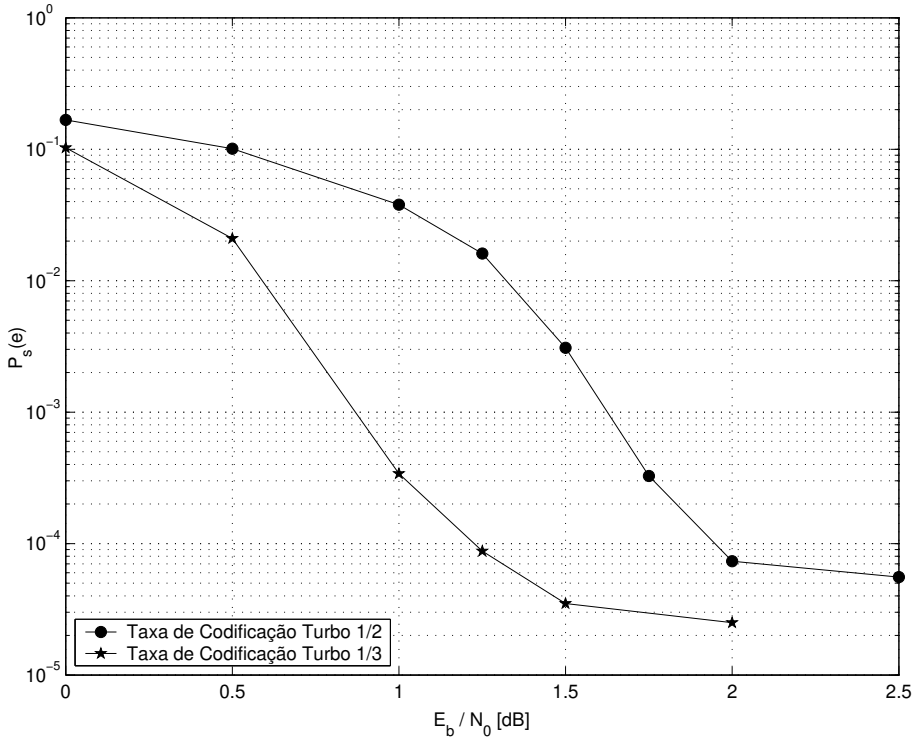


Figura 5.8: Efeito da variação da taxa de codificação turbo para codificador *RSC*-**A**, comprimento da seqüência de informação $N = 5000$ símbolos e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

A Figura 5.9 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificador *RSC*-**A**, taxas de codificação turbo igual a 1/3 e 1/2, décima quinta iteração e comprimento da seqüência de informação $N = 10000$ símbolos.

Note que, para as taxas de codificação turbo 1/2 e 1/3, as curvas referentes a estas taxas

alcançam a taxa de erro de símbolo em 10^{-4} para $E_b/N_0 \simeq 1.81\ dB$ e $E_b/N_0 \simeq 1.10\ dB$, respectivamente. Ou seja, para a taxa de erro de símbolo de 10^{-4}, a curva referente à taxa de codificação turbo 1/3, apresenta um ganho de aproximadamente 0.71 dB em relação à curva referente à taxa de codificação turbo 1/2.

Portanto, as Figuras (5.6, 5.7, 5.8 e 5.9) mostram que a variação da taxa de codificação turbo de 1/2 para 1/3 fornece ganho maior que 0.65 dB da taxa, E_b/N_0, entre as curvas. Assim, o puncionador é considerado uma ferramenta importante do esquema turbo.

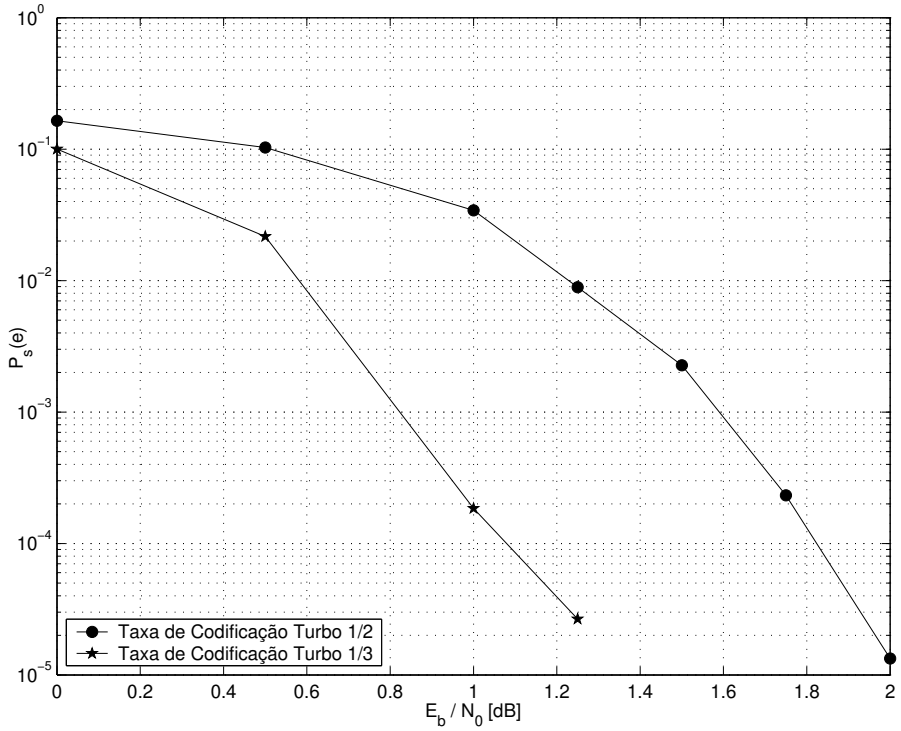


Figura 5.9: Efeito da variação da taxa de codificação turbo para codificador *RSC*-**A**, comprimento da seqüência de informação $N = 10000$ símbolos e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

5.2.4 Análises do Desempenho do Código Constituinte Utilizado em Termos do Número de Estado da Treliça do Código Constituinte

A Figura 5.10 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificadores *RSC*-**A** e *RSC*-**B**, comprimento da seqüência de informação, $N = 5000$ símbolos, décima quinta iteração e taxa de codificação turbo 1/2.

As curvas referentes aos codificadores *RSC*-**A** e *RSC*-**B**, alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 1.98\ dB$ e $E_b/N_0 \simeq 0.81\ dB$, respectivamente. Isto é, a curva referente ao codificador *RSC*-**B** apresenta um ganho de aproximadamente 1.17 dB em relação a curva referente ao codificador *RSC*-**A** para taxa de erro de símbolo de 10^{-4}.

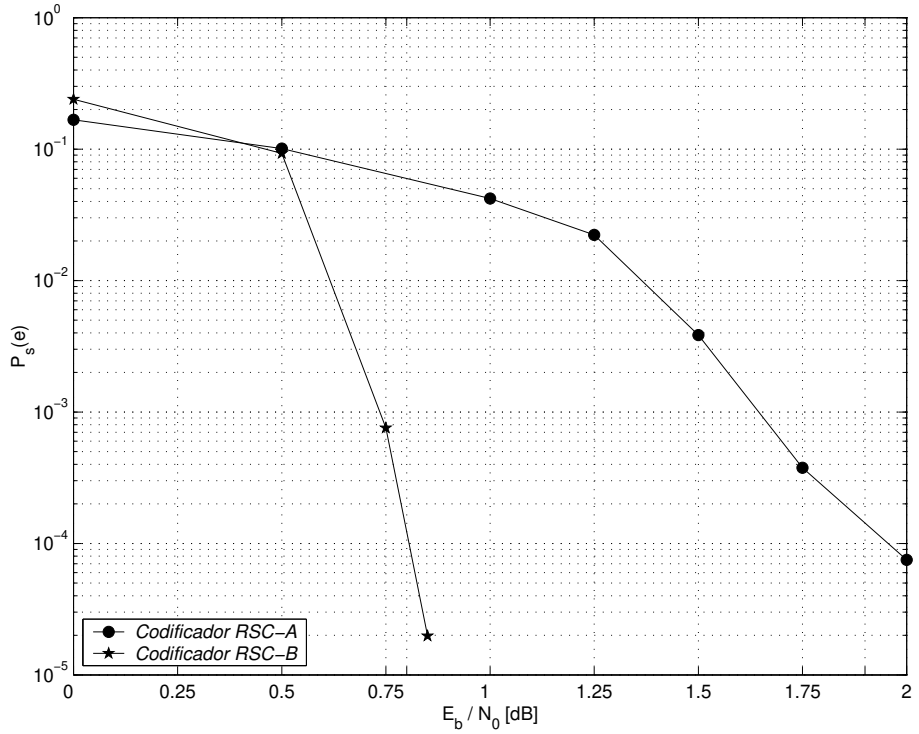


Figura 5.10: Efeito da variação do codificador *RSC* para taxa de codificação turbo 1/2, quinze iterações e $N = 5000$ símbolos sobre a curva de $p_s(e) \times E_b/N_0$.

Portanto, a Figura 5.10 mostra que a variação do codificador *RSC* proporciona um ganho bastante significativo da taxa E_b/N_0. Em outras palavras, a curva referente codificadores *RSC*-

B, apresenta um ganho de aproximadamente 1.17 dB em relação a curva referente codificadores RSC-**A**. Portanto, podemos considerar a escolha do codificador RSC como sendo uma ferramenta muito importante do esquema turbo.

A Figura 5.11 apresenta as curvas da taxa de erro de símbolo versus E_b/N_0 para o esquema turbo que possui: codificadores RSC cujas treliças possuem 4 estados, 8 estados e 16 estados, comprimento da seqüência de informação, $N = 5000$ símbolos, décima quinta iteração e taxa de codificação turbo 1/2.

As curvas correspondentes aos codificadores RSC cujas treliças possuem 4 estados, 8 estados e 16 estados, alcançam a taxa de erro de símbolo de 10^{-4} em $E_b/N_0 \simeq 1.98\ dB$, $E_b/N_0 \simeq 1.96\ dB$ e $E_b/N_0 \simeq 0.81\ dB$, respectivamente. Isto é, a curva referente ao codificador RSC que possui uma treliça com 16 estados apresenta um ganho de aproximadamente 1.17 dB em relação as curvas referentes aos codificadores RSC que possuem treliças com 4 estados e 8 estados para taxa de erro de símbolo de 10^{-4}.

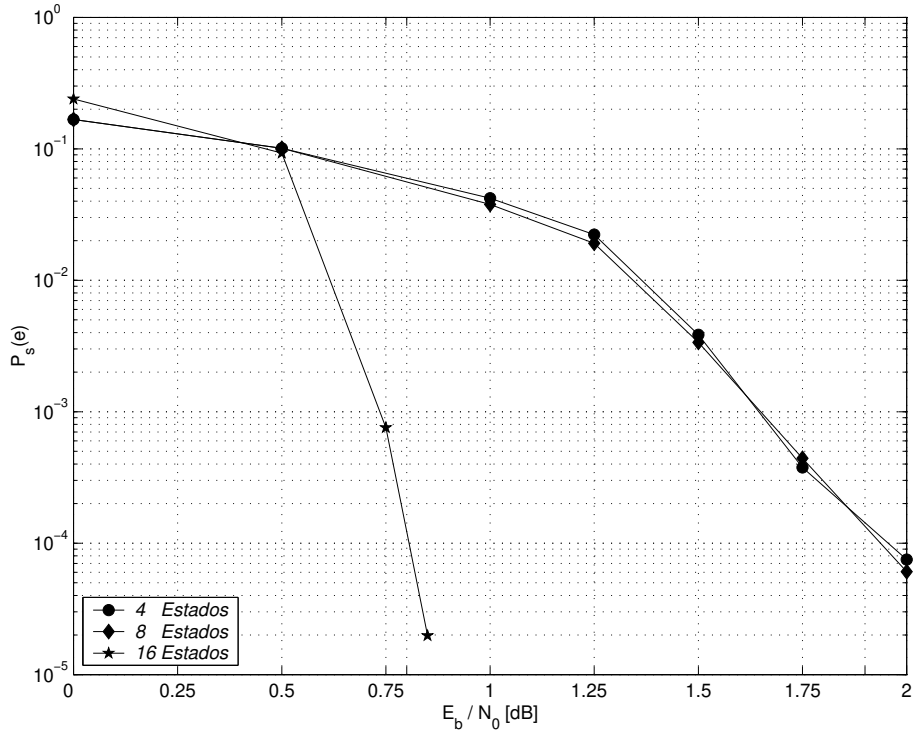


Figura 5.11: Efeito da variação do número de estados da treliça para comprimento da seqüência de informação $N = 5000$ símbolos, taxa de codificação turbo 1/2 e quinze iterações, sobre a curva de $p_s(e) \times E_b/N_0$.

Notemos que a curva referente ao codificador cuja treliça possui 8 estados é equivalente a curva referente ao codificador cuja treliça possui 4 estados. Observe que este fato ocorre porque a máxima distância euclidiana mínima ao quadrado entre as palavras código referentes aos codificadores cujas treliças possuem 4 e 8 estados são iguais. Além disso, a treliça com 8 estados é equivalente a duas treliças com 4 estados ver Apêndice A.

5.3 Conclusão

Neste trabalho, apresentamos uma estrutura matemática e estatística suficiente para a construção de um esquema de codificação e decodificação turbo quaternário. Estudamos os componentes do esquema de codificação turbo (incluindo simulações sobre os entrelaçadores), analisando a contribuição dada por cada um deles para a construção de um esquema de decodificação turbo quaternário que forneça o máximo de confiabilidade sobre os símbolos decodificados. Em seguida, fizemos a implementação do algoritmo decodificação obtendo como resultados as curvas para a probabilidade de erro nas quais notamos que:

- quanto maior for o número de iterações,
- quanto maior for o comprimento da seqüência de informação,
- quanto maior for o número de estados da treliça referente ao codificador RSC com máxima distância euclidiana mínima ao quadrado entre as palavras código, e,
- quanto menor for a taxa de codificação turbo,

mais próxima a curva referente à simulação para o esquema turbo encontra-se do limitante de *Shannon*. Todavia, é importante notar que:

- Existe um determinado valor para o número de iterações tal que, se aumentarmos este valor, aumentamos em conseqüência disto o tempo gasto na decodificação, mas não obtemos um ganho significativo entre as curvas correspondendo ao aumento dessas iterações.

Observando a Figura 5.1 para a taxa de erro de *símbolo* de 10^{-4}, vemos que, enquanto o ganho entre as curvas que correspondem a 1ª e a 2ª iteração é de $\simeq 1.6\ dB$, o ganho entre as curvas que correspondem a 6ª e a 15ª iteração é apenas de $0.19\ dB$. Note também que o valor a partir do qual não teremos mais ganho significativo, se aumentarmos o número de iterações, depende do codificador *RSC*, do comprimento da seqüência de informação e da taxa de codificação turbo;

- Existe um determinado valor para o comprimento da seqüência de informação (este valor depende do codificador *RSC*, do número de iterações e da taxa de codificação turbo), que, a partir deste valor, se aumentarmos o comprimento da seqüência de informação, aumentamos em conseqüência disto o tempo gasto na decodificação, mas o ganho obtido com esse aumento é muito pequeno. Observando a Figura 5.2 para a taxa de erro de *símbolo* de 10^{-4}, temos que o ganho correspondente as curvas referentes aos comprimentos da seqüência de informação $N = 2000$ símbolos e $N = 500$ símbolos é de $0.45\ dB$ e o ganho para às curvas referentes aos comprimentos da seqüência de informação $N = 5000$ símbolos e $N = 2000$ símbolos é de $0.20\ dB$;

- Existe um determinado valor (16 estados) para o número de estados da treliça do codificador *RSC*, que, a partir deste valor, o tempo gasto na decodificação é muito grande, pois o número de estados da treliça do codificador *RSC* quaternário é da ordem de 4^m, onde m é o grau do polinômio da matriz geradora do codificador *RSC*;

- Finalmente, se baixarmos muito a taxa de codificação turbo, ficamos com um esquema turbo que transmite pouca informação e muita redundância.

Portanto, entre a faixa de valores que existe para esses parâmetros, devemos escolher valores que proporcionem sempre baixa taxa de erro de *bit* (ou *símbolo*) versus baixa taxa E_b/N_0, para um esquema turbo com baixa complexidade no algoritmo de decodificação e que transmita o máximo de informação.

Desse modo, a escolha do codificador-*RSC* é um requisito importante no esquema turbo, pois o codificador *RSC*-**A**, apesar de ser o codificador que possui maior distância euclidiana mínima ao quadrado entre as palavras código (entre os codificadores cuja treliça possui 4 estados), não proporciona bons resultados quando comparado com os resultados dos códigos turbo binários [15], [1] vistos na literatura. Por outro lado, o codificador *RSC*-**B**, que possui maior distância euclidiana mínima ao quadrado entre as palavras código (entre os codificadores cuja treliça possui 16 estados), proporciona desempenho semelhante ao desempenho apresentado pelos códigos turbo binários.

Em outras palavras, a curva mostrada no esquema turbo quaternário da Figura 5.12 atinge a taxa de erro de *bit* de 10^{-4} em $E_b/N_0 \simeq 1.43\ dB$ para o comprimento da seqüência de informação $N = 1000$ *bits* e 15 iterações, enquanto que, para o mesmo comprimento da seqüência de informação e o mesmo número de iterações , a curva apresentada para o esquema binário atinge a taxa de erro de *bit* de 10^{-4} em $E_b/N_0 \simeq 1.44\ dB$ (ver Figura 5.13 ou [15]). E, finalmente, para $N = 40000$ *bits* (valor máximo para o qual conseguimos realizar as simulações referentes ao codificador *RSC*-**B**), a curva mostrada no esquema turbo quaternário da Figura 5.12 atinge a taxa de erro de *bit* de 10^{-4} em $E_b/N_0 \simeq 0.71\ dB$ com 15 iterações, taxa de codificação turbo 1/2 e um entrelaçador melhor do que o entrelaçador utilizado por C. Berrou [1]. Portanto, estamos à uma distância de $\simeq 0.06\ dB$ do valor encontrado para o esquema turbo binário (ver Figura 5.14 ou [1]) cujo resultado é o seguinte: para o comprimento da seqüência de informação $N = 65536$ *bits*, taxa de codificação turbo 1/2, a curva apresentada para o esquema binário atinge a taxa de erro de *bit* de 10^{-4} em $E_b/N_0 \simeq 0.65\ dB$.

Logo, como o esquema turbo quaternário (com a modulação *4-PSK*) ocupa a metade da largura de faixa (banda) utilizada no esquema turbo binário (com modulação *2-PSK*), ou seja, para a mesma largura de faixa, o esquema turbo quaternário pode fornecer o dobro da quantidade de informação fornecida no esquema turbo binário, podemos concluir que o uso do esquema turbo quaternário em um sistema de comunicação que possua largura de faixa limitada será mais vantajoso que o esquema turbo binário.

Finalmente, como o algoritmo de decodificação turbo quaternário é computacionalmente mais complexo que o algoritmo utilizado no caso binário, torna-se inadequada a utilização do algoritmo turbo quaternário na realização das simulações para um comprimento da seqüencia de informação $N > 20000$ símbolos. Portanto, sugerimos os trabalhos futuros: o algoritmo de decodificação "*MAX-LOG-MAP*"e o algoritmo de decodificação "*LOG-MAP*"que tem uma complexidade bem menor; Além destes, temos o "Algoritmo de Busca Reduzida", que baseia-se no fato de que: a partir de um certo valor para a taxa E_b/N_0, podemos diminuir o número de iterações utilizado na taxa E_b/N_0 seguinte sem ter perdas significativas no ganho, reduzindo-se assim a complexidade do algoritmo. Estima-se que com estas modificações no algoritmo poderemos utilizar comprimentos das seqüencias da informação maiores para uma complexidade do algoritmo menor.

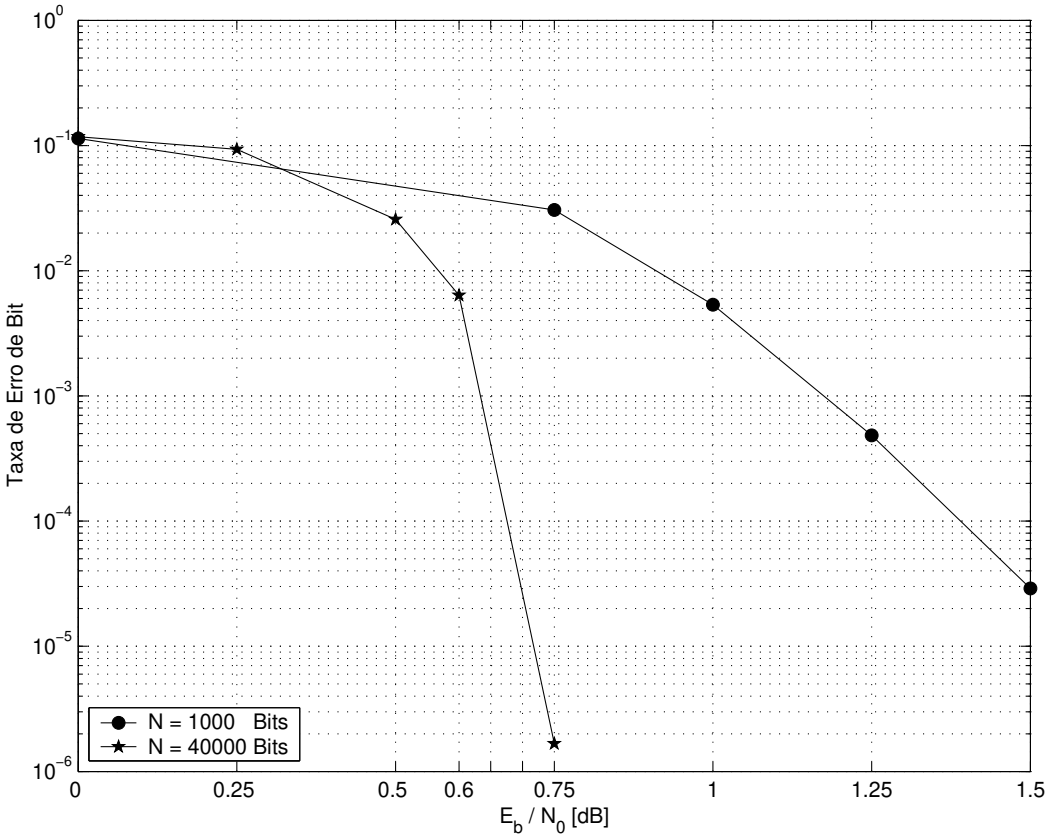


Figura 5.12: Efeito do codificador *RSC*-**B** cuja treliça possui 16 estados, comprimento da seqüência de informação $N = 40000$ *bits*, taxa de codificação turbo 1/2 e quinze iterações, sobre a curva de $p_b(e) \times E_b/N_0$.

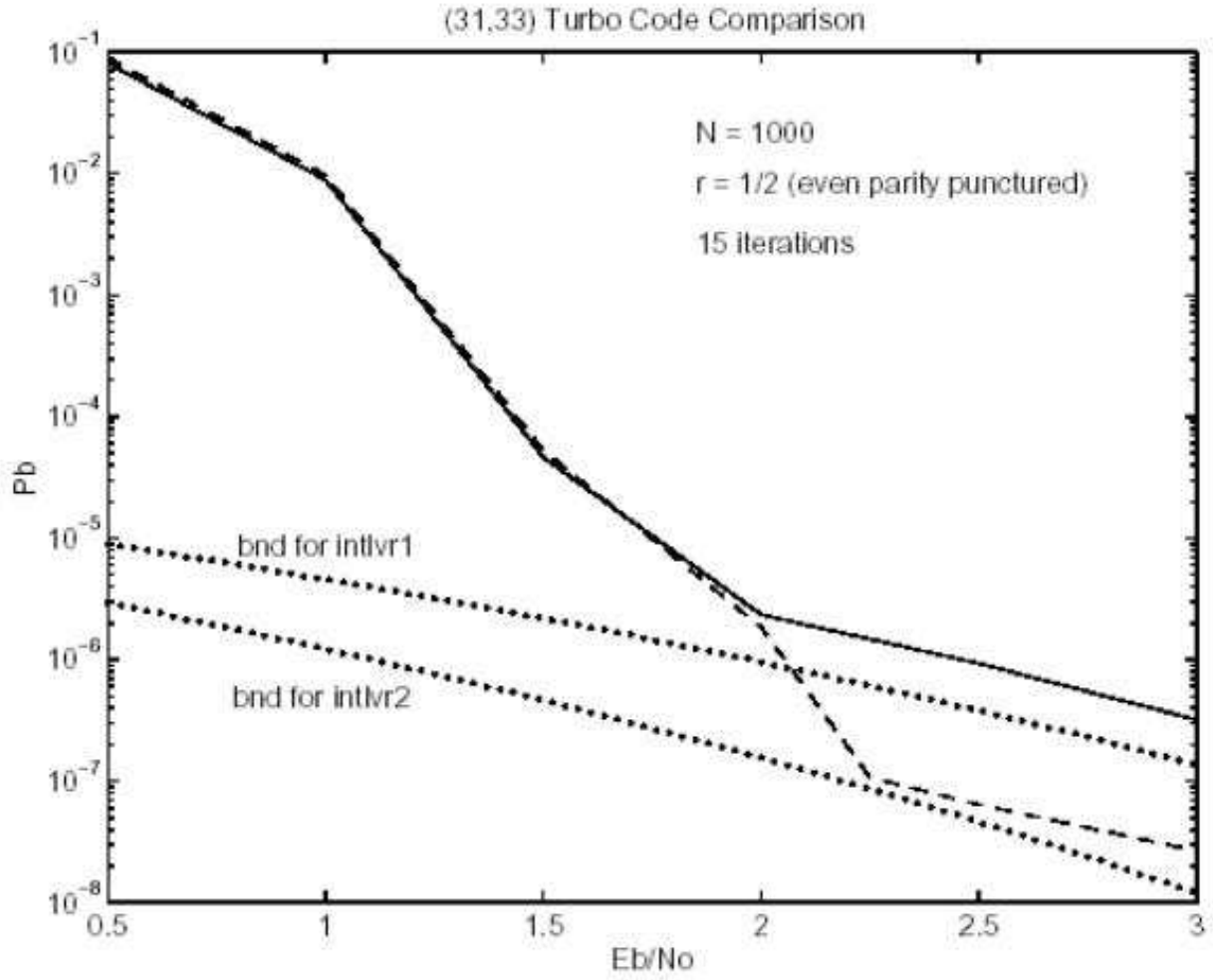


Figura 5.13: Efeito do codificador turbo binário cuja treliça possui 16 estados, comprimento da seqüência de informação $N = 1000$ *bits*, taxa de codificação turbo 1/2, quinze iterações, sobre a curva de $p_b(e) \times E_b/N_0$ [15].

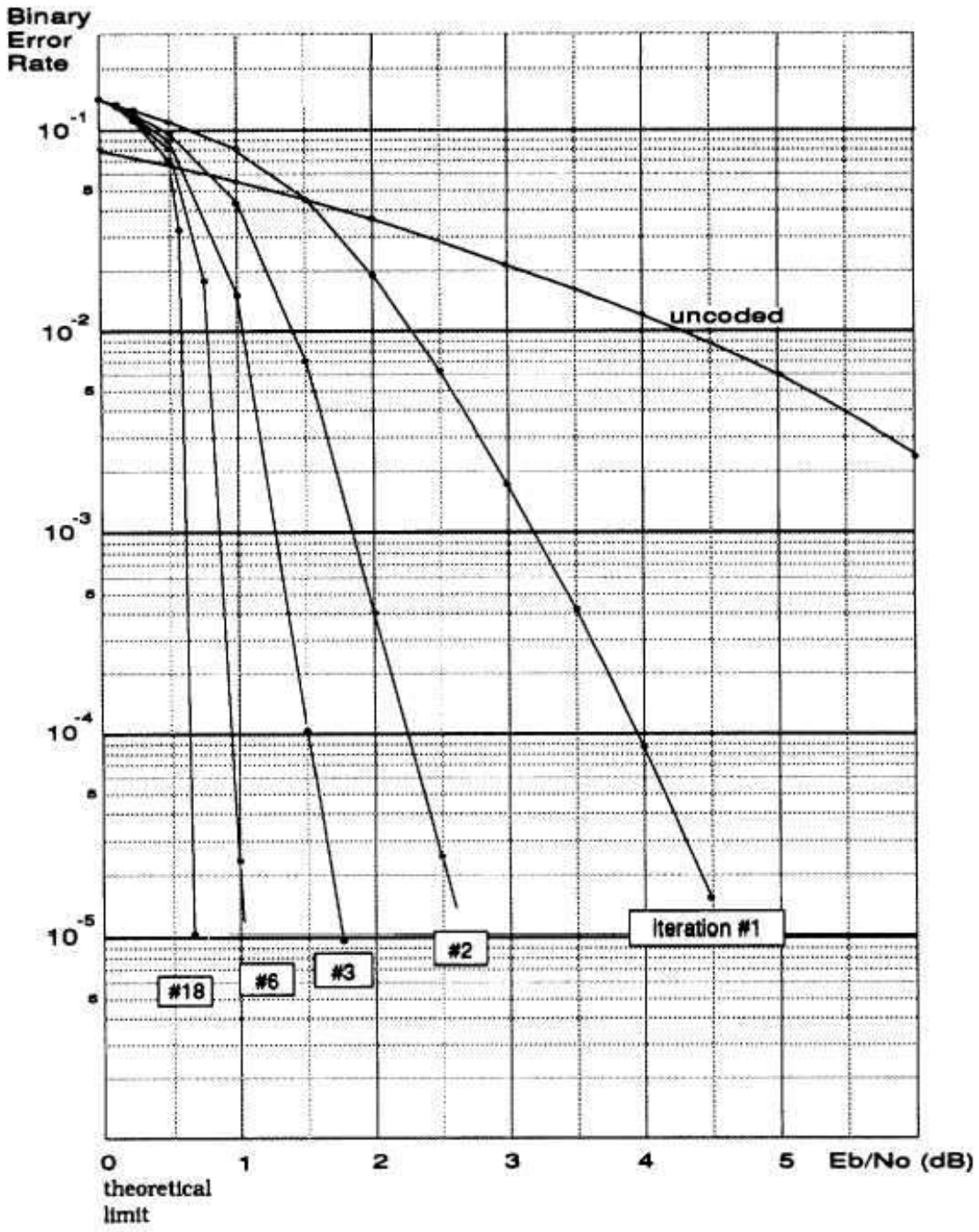


Figura 5.14: Efeito da variação do número de iterações para comprimento da seqüência de informação $N = 65536$ *bits*, taxa de codificação turbo 1/2 e codificador turbo binário cuja treliça possui 16 estados, sobre a curva de $p_b(e) \times E_b/N_0$ [1].

Apêndice A

Aspectos Relacionados aos Codificadores

A.1 Treliça com 4 Estados

O codificador RSC, com matriz geradora $g(D) = \left[1 \ \frac{2+D}{1+3D}\right]$, possui uma treliça com 4 estados cujos elementos são descritos na Tabela A.1.

A Tabela A.1, que está dividida em duas partes, contém os elementos da treliça nos dois sentidos. Ou seja, a primeira parte da Tabela A.1 contém os elementos da treliça posterior[1], que pode ser observado da seguinte maneira: Considerando que a treliça encontra-se no Estado Presente ($EP \in \{0,\ 1,\ 2,\ 3\}$), qual o Próximo Estado ($PE \in \{0,\ 1,\ 2,\ 3\}$) a ser alcançado pela transição, e, qual a Próxima Saída ($PS \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$) associada a esta transição, dado que o símbolo $\theta \in \{0,\ 1,\ 2,\ 3\}$ foi recebido.

A segunda parte da Tabela A.1 contém os elementos da treliça anterior[2], descrita conforme a seguir.

Considerando que a treliça encontra-se no Estado Presente ($EP \in \{0,\ 1,\ 2,\ 3\}$), qual o Estado Anterior ($EA \in \{0,\ 1,\ 2,\ 3\}$) do qual ocorreu a transição para o Estado Presente, e,

[1]Treliça que fornece o Próximo Estado e a Próxima Saída a partir do Estado Presente e do símbolo recebido.
[2]Treliça que fornece o Estado Anterior e a Saída Anterior a partir do Estado Presente e do símbolo recebido.

qual a Saída Anterior ($SA \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$) associada a esta transição, dado que o símbolo $\theta \in \{0,\ 1,\ 2,\ 3\}$ foi recebido.

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	PE	PS	PE	PS	PE	PS	PE	PS
0	0	00	1	12	2	20	3	32
1	3	03	0	11	1	23	2	31
2	2	02	3	10	0	22	1	30
3	1	01	2	13	3	21	0	33

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	EA	SA	EA	SA	EA	SA	EA	SA
0	0	00	1	11	2	22	3	33
1	3	01	0	12	1	23	2	30
2	2	02	3	13	0	20	1	31
3	1	03	2	10	3	21	0	32

EP = Estado Presente
PE = Próximo Estado
PS = Próxima Saída
EA = Estado Anterior
SA = Saída Anterior

Tabela A.1: Treliça posterior e anterior com 4 estados.

A.2 Treliça com 16 Estados

O codificador RSC, com matriz geradora $g(D) = \left[1\ \frac{2 + D + 2D^2}{1 + D + 3D^2}\right]$, possui uma treliça posterior com 16 estados, mostrada na Tabela A.2, e descrita da seguinte maneira: Considerando que a treliça encontra-se no Estado Presente $EP \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$, qual o Próximo Estado $PE \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$ a ser alcançado pela transição, e qual a Próxima Saída $PS \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$ associada a esta transição, dado que o símbolo $\theta \in \{0,\ 1,\ 2,\ 3\}$ foi recebido.

De maneira similar, apresentamos agora a treliça anterior mostrada na Tabela A.3. Isto é, considerando que a treliça encontra-se no Estado Presente $EP \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$, qual o Estado anterior $EA \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$ do qual ocorreu a transição para o

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	*PE*	*PS*	*PE*	*PS*	*PE*	*PS*	*PE*	*PS*
00	00	00	10	12	20	20	30	32
01	30	00	00	12	10	20	20	32
02	20	00	30	12	00	20	10	32
03	10	00	20	12	30	20	00	32
10	11	03	21	11	31	23	01	31
11	01	03	11	11	21	23	31	31
12	31	03	01	11	11	23	21	31
13	21	03	31	11	01	23	11	31
20	22	02	32	10	02	22	12	30
21	12	02	22	10	32	22	02	30
22	02	02	12	10	22	22	32	30
23	32	02	02	10	12	22	22	30
30	33	01	03	13	13	21	23	33
31	23	01	33	13	03	21	13	33
32	13	01	23	13	33	21	03	33
33	03	01	13	13	23	21	33	33

EP = Estado Presente
PE = Próximo Estado
PS = Próxima Saída

Tabela A.2: Treliça posterior com 16 estados.

Estado Presente, e qual a Saída Anterior $SA \in \{00,\ 01,\ 02,\ 03,\ \ldots\ ,\ 32,\ 33\}$ associada a esta transição, dado que o símbolo $\theta \in \{0,\ 1,\ 2,\ 3\}$ foi recebido.

A.3 Treliça com 8 Estados

Analisando o codificador *RSC* com matriz geradora $g(D) = \left[1 \ \frac{2 + D + D^2}{1 + D + 3D^2}\right]$, cujo diagrama é descrito na Figura 2.2, verificamos que fazendo-se a multiplicação por 2 módulo 4, do valor que entra no segundo elemento de memória, o codificador resultante encontrado possui uma treliça que contém apenas 8 estados. Ou seja, a treliça posterior correspondente ao codificador proposto contém apenas os 8 estados: 00 02 10 12 20 22 30 32 e é mostrada na Tabela A.4.

Com base na Tabela A.4 verifica-se que esta treliça posterior com 8 estados, é equivalente a

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	*EA*	*SA*	*EA*	*SA*	*EA*	*SA*	*EA*	*SA*
00	00	00	01	12	02	20	03	32
01	11	03	12	11	13	23	10	31
02	22	02	23	10	20	22	21	30
03	33	01	30	13	31	21	32	33
10	03	00	00	12	01	20	02	32
11	10	03	11	11	12	23	13	31
12	21	02	22	10	23	22	20	30
13	32	01	33	13	30	21	31	33
20	02	00	03	12	00	20	01	32
21	13	03	10	11	11	23	12	31
22	20	02	21	10	22	22	23	30
23	31	01	32	13	33	21	30	33
30	01	00	02	12	03	20	00	32
31	12	03	13	11	10	23	11	31
32	23	02	20	10	21	22	22	30
33	30	01	31	13	32	21	33	33

EP = Estado Presente
EA = Estado Anterior
SA = Saída Anterior

Tabela A.3: Treliça anterior com 16 estados.

duas treliças com 4 estados. Isto é, as linhas correspondentes aos Estados Presentes 00 02 10 12 são iguais as linhas correspondentes aos Estados Presentes 22 20 32 30, respectivamente.

A Tabela A.5, que descreve a treliça anterior para o codificador *RSC* cuja treliça contém apenas 8 estados, também apresenta características semelhantes as da Tabela A.4.

Portanto, este codificador cujas treliças (posterior e anterior) possuem apenas 8 estados (ver Tabelas A.4 e A.5), tem desempenho equivalente ao codificador referente a Tabela A.1 cuja treliça possui 4 estados (ver Figura 5.11). Pois a máxima distância euclidiana mínima ao quadrado entre as palavras código referentes a treliça com 8 estados é igual a máxima distância euclidiana mínima ao quadrado entre as palavras código referentes a treliça com 4 estados.

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	*PE*	*PS*	*PE*	*PS*	*PE*	*PS*	*PE*	*PS*
00	00	00	10	12	20	20	30	32
02	20	02	30	10	00	22	10	30
10	12	03	22	11	32	23	02	31
12	32	01	02	13	12	21	22	33
20	20	02	30	10	00	22	10	30
22	00	00	10	12	20	20	30	32
30	32	01	03	13	12	21	22	33
32	12	03	22	11	32	23	02	31

EP = Estado Presente
PE = Próximo Estado
PS = Próxima Saída

Tabela A.4: Treliça posterior com 8 estados.

	$\theta = 0$		$\theta = 1$		$\theta = 2$		$\theta = 3$	
EP	*EA*	*SA*	*EA*	*SA*	*EA*	*SA*	*EA*	*SA*
00	00	00			02	22		
	22	00			20	22		
02			12	13			10	31
			30	13			32	31
10			00	12			02	30
			22	12			20	30
12	10	03			12	21		
	32	03			30	21		
20	02	02			00	20		
	20	02			22	20		
22			10	11			12	33
			32	11			30	33
30			02	10			00	32
			20	10			22	32
32	12	01			10	23		
	30	01			32	23		

EP = Estado Presente
EA = Estado Anterior
SA = Saída Anterior

Tabela A.5: Treliça anterior com 16 estados.

Bibliografia

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," Proc. Int. Conf. Comm., pp. 1064-1070, May 1993.

[2] C. Berrou, A. Glavieux, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," IEEE Trans. Comm., vol. 44, n. 10, pp. 1261-1271, Oct. 1996.

[3] J. Hagenauer, E. Offer, and L.Papke, "Iterative decoding of binary block and convolutional codes," IEEE Trans. Inform. Theory, vol. 42, pp. 429-445, Mar. 1996.

[4] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," IEEE Trans. Inform. Theory, pp. 284-287, Mar. 1974.

[5] G. Ungerboeck, "Channel coding with multilevel/phase signals," IEEE Trans. Inform. Theory, pp. 55-67, Jan. 1982

[6] P. Robertson, "Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes," Proc. Globecom. pp. 1298-1303, 1994.

[7] D. Divsalar and F. Pollara, "Turbo codes for PCS application," Proc. 1995 Int. Conf. Comm., pp. 54-59.

[8] K. Sayood and J. C. Borkenhagen, "Use of residual redundancy in the design of joint source and channel coders," IEEE Trans. Comm., vol. 39, pp. 838-846, June 1991.

[9] K. Sayood, F. Liu, and J. D. Gibson, "A constrained joint source/channel coder design," IEEE J. Select. Areas Commun., vol. 12, pp. 1584-1593, Dec. 1994.

[10] R. Baldini F. and P. G. Farrell, "Coded Modulation Based on Rings of Integers Modulo-q, Parte-I: Block Codes," IEEE Proc. Commun., pp. 129-136, Vol. 141, n. 3, Jun. 1994.

[11] R. Baldini F. and P. G. Farrell, "Coded Modulation Based on Rings of Integers Modulo-q, Parte-II: Convolutional Codes," IEEE Proc. Commun., pp. 137-142, Vol. 141, n. 3, Jun. 1994.

[12] A. Ruscitto and E. M. Biglieri, "Joint Source and Channel Coding using Turbo Codes over Rings," IEEE Trans. on Comms., vol. 46, n. 8, Aug. 1998.

[13] J. Berkmann, "A Symbol-by-Symbol Map Decoding Rule for Linear Codes over Rings using the Dual Codes," pp. 90, Proc. of ISIT', Cambridge, MA, USA, Aug. 1998, pp. 16-21.

[14] Jason P. Woodard and Lajos Hanzo, "Comparative Study of Turbo Decoding Techniques: An Overview," IEEE Trans. Veh. Technol, Vol. 49, n. 6, pp. 2208-2233, Nov. 2000.

[15] W. E. Ryan, "A Turbo Code Tutorial,"Proc. IEEE Globecom'98, 1998.

[16] Rolf Johannesson, "Some Structural Properties of Convolutional Codes over Rings," IEEE Trans. Inform. Theory, pp. 839-845, Mar. 1998.

[17] M. F. Atiyah and I. G. MacDonald, "Commutative Algebra," Reading, MA: Addison - Wesley, 1969.

[18] Chris Heegard and Stephen B. Wicker, "Turbo Coding," Kluwer Academic Publishers pp. 10-63.

[19] K. S. Andrews, C. Heegard and D. Kozen, "Interleaver Desingn Methods for Turbo Codes", Proceedings of the 1998. International Symposium on Information Theory, pg. 420.

[20] F. Fagnini and S. Zampieri, "System-Theoretic Properties of Convolutional Codes Over Rings," IEEE Trans. Inform. Theory, vol. 47, N^o. 6, Sept. 2001, pp. 2256-2274.