



Universidade Federal de Pernambuco
Centro de Informática

Graduação em Engenharia da Computação

**Simulador de transmissão ADSL2+ para
análise de erros nas camadas física e de
enlace**

Rodrigo Perazzo Rabelo

Trabalho de Graduação

Recife
14 de dezembro de 2010

Universidade Federal de Pernambuco
Centro de Informática

Rodrigo Perazzo Rabelo

**Simulador de transmissão ADSL2+ para análise de erros nas
camadas física e de enlace**

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Engenheiro da Computação.

Orientador: Prof. Djamel Fawzi Hadj Sadok
Co-orientador: Glauco Estácio Gonçalves

Recife
14 de dezembro de 2010

Dedico esse trabalho à minha família e meus amigos.

Agradecimentos

Agradeço a todos os professores envolvidos na minha formação básica e acadêmica, principalmente no tocante à minha formação como cidadão capaz de retribuir à sociedade, com o meu trabalho, todo investimento e confiança dada. Agradeço àqueles que diretamente ou indiretamente me influenciaram a tentar seguir o caminho da pesquisa, que hoje é minha grande fonte de aprendizado e perspectiva de futuro. Agradeço aos meus colegas de graduação, com os quais me reuni inúmeras vezes para discutir e aprender os novos conceitos que frequentemente surgiram durante o curso, bem como para implementar projetos nas mais difíceis horas e nos dias em que a última coisa que se queria fazer era estar na faculdade. Também agradeço a eles a companhia nas horas de lazer, porque rir com eles foi combustível nessa caminhada. Agradeço aos colegas do GPRT que muito me ajudaram a esclarecer idéias e sugerir muitas outras. Agradeço finalmente, e especialmente, ao prof. Djamel Sadok e ao Doutorando Glauco Gonçalves pela orientação perspicaz e atuante neste trabalho.

Resumo

No provimento do acesso à Internet banda larga, é comum a prática de não dar garantias da qualidade do serviço. Devido a viabilidade técnica e financeira para a implantação das tecnologias DSL, o número de usuários vem crescendo rapidamente e a administração individual da qualidade das linhas se torna progressivamente mais difícil. Por conta de seu funcionamento complexo, o desconhecimento do exato comportamento das configurações de proteção pode levar a efeitos indesejados e usuários insatisfeitos. Este trabalho apresenta uma ferramenta de simulação capaz de aferir a qualidade da transmissão de dados de uma linha ADSL2+, oferecendo medidas de desempenho para a camada física e de enlace. A ferramenta permite ainda uma vasta gama de configurações desde o tipo e nível dos ruídos mais comuns às linhas até a configuração dos parâmetros de proteção das mesmas. Seu principal objetivo é agilizar o processo de escolher parâmetros de proteção para linhas das quais se conhece um conjunto mínimo de características.

Palavras-chave: Internet banda larga, DSL, ferramenta de simulação, qualidade de serviço.

Abstract

In the provision of broadband Internet access, it is common practice not to provide assurance of service quality. Due to technical and financial feasibility for deployment of DSL technologies, the number of customers is growing rapidly and the management of the individual quality of the lines becomes progressively more difficult. Because of its complex behavior, the ignorance of the exact purpose of protection settings can lead to undesired effects and unhappy users. This paper presents a simulation tool capable of measuring the quality of ADSL2plus data transmission line, providing performance measures for the physical and the data link layers. The tool also allows a wide range of configurations from the type and level of common noises to the configuration parameters of data protection. Its main objective is to streamline the process of choosing the protection parameters for lines of which are known a minimum set of features.

Keywords: Broadband Internet access, DSL, Simulation Tool, QoS.

Sumário

1	Introdução	1
1.1	Objetivo do trabalho	3
1.2	Visão geral do trabalho	3
2	Tecnologias DSL	4
2.1	Introdução ao DSL	4
2.2	Modulação de Amplitude em Quadratura	5
2.3	Multiplexação Por Divisão em Frequência Ortogonal	8
2.4	Algoritmo de Alocação de <i>bits</i>	10
2.5	Ruídos presentes na comunicação	11
2.6	Códigos corretores e Reed-Solomon	12
2.7	Sincronização DSL	15
2.8	Trabalhos relacionados	15
3	Descrição do Simulador	17
3.1	Base Conceitual	17
3.2	Passos da Sincronização DSL	18
3.2.1	Cálculo do SNR	18
3.2.2	Algoritmo de alocação de <i>bits</i>	20
3.2.3	Escolhendo os parâmetros de framing	21
3.2.4	Ordenação das sub-portadoras	21
3.3	Codificador e Decodificador Reed-Solomon	21
3.4	Interleaving e Deinterleaving	21
3.5	Mapeamento na constelação QAM e interpretação QAM	22
3.6	Modulação e Demodulação	23
3.7	Meio Físico	24
3.7.1	Ruído Gaussiano Branco Aditivo	25
3.7.2	Ruído Impulsivo	26
3.8	Contagem de erros	27
3.9	Interface da ferramenta	27
3.9.1	Interface dos resultados	29
4	Avaliação do Simulador	31
4.1	Validação	31
4.2	Análise Qualitativa	35
4.3	Resultados e Análise	37

4.3.1	Erros antes da decodificação Reed-Solomon	37
4.3.2	Erros após a decodificação Reed-Solomon	39
4.3.3	Erros na camada de enlace	39
4.4	Cenários de uso da ferramenta	42
4.4.1	Predição de Taxa	42
4.4.2	Otimização pela diminuição do erro	43
5	Conclusão	45
5.1	Versatilidade do simulador	45
5.2	Trabalhos Futuros	46
A	Códigos implementados	47
A.1	Algoritmo de alocação de bits	47
A.2	Procedimento para escolha dos parametros de framing	48
A.3	Algoritmo de Interleaving	49
A.4	Algoritmo para gerar as constelações QAM	50
	Referências	53

Lista de Figuras

1.1	Crescimento do número de conexões ADSL no Brasil [2].	1
2.1	Diagrama DSL.	4
2.2	Exemplo de modulação em amplitude[11].	6
2.3	Exemplo de uma constelação 16-QAM[12].	7
2.4	Representação da função Q[15].	8
2.5	Sub-portadoras no esquema OFDM [17].	9
2.6	Forma de onda de um ruído impulsivo de 1 rajada[20].	11
2.7	<i>Crosstalk</i> FEXT e NEXT [19].	12
3.1	Módulos do simulador.	18
3.2	Potência limite nas faixas de operação do DSL [18].	19
3.3	Curva de atenuação por frequência para uma linha telefônica com 2km de extensão.	20
3.4	Interface principal da ferramenta de simulação.	28
3.5	Interface de resultados da ferramenta de simulação.	30
4.1	Diagrama da plataforma de testes.	32
4.2	Histograma do erro no cenário 3.	33
4.3	Alocação de bits no cenário 2.	34
4.4	Alocação de bits no cenário 4.	35
4.5	Distribuição dos erros num cenário sem interleaving.	38
4.6	Distribuição dos erros num cenário com interleaving.	38
4.7	Distribuição dos erros após a decodificação do Reed-Solomon.	39
4.8	Contador de erros do cabeçalho ATM com ruído AWGN.	40
4.9	Contador de erros do cabeçalho ATM com ruído impulsivo.	41

Lista de Tabelas

1.1	Número de modificações e novas versões dos padrões publicados pela ITU-T.	2
2.1	Valores de R e D permitidos pelo padrão ADSL2+.	14
3.1	Constelação 8-QAM dos sistemas DSL.	23
3.2	Relação entre os 2 MSBs de X e Y com o vetor de b bits.	24
3.3	Resumo dos parâmetros configuráveis da interface principal.	29
4.1	Cinco cenários distintos para validação do algoritmo de alocação.	32
4.2	Dados do teste da média zero para o cenário 3.	34
4.3	Comparação dos simuladores.	36
4.4	Valores dos parâmetros de proteção para os experimentos de HEC.	40

CAPÍTULO 1

Introdução

O acesso à Internet, hoje, em altas taxas de recepção de dados e de forma prática é realidade e até uma exigência. A taxa de transmissão de dados, com tecnologias cabeadas, pode chegar até 200 Mbit/s [1, p. 3] e mesmo tecnologias de Internet móvel já ultrapassaram as conexões discadas de 56 kbit/s. Com esse avanço, o serviço de banda larga¹ em geral (por meio de fios telefônicos, fibra ótica ou por ondas eletromagnéticas) se tornou muito atrativo nos últimos anos, tanto para os consumidores residenciais como para os empresariais, que representam 23% dos usuários [2].

As tecnologias DSL (*Digital Subscriber Line*) são disparadas as mais viáveis e com maior aceitação dos consumidores. Sua viabilidade se dá pelo fato de operarem sobre uma infraestrutura já pronta e funcional desde o final do século XIX, a infraestrutura de telefonia. Desta forma, todos os usuários de telefonia tornaram-se usuários potenciais para esta nova tecnologia, o que impulsionou o crescimento do mercado logo após o seu surgimento. Em pesquisa realizada em dezembro de 2009, pela IDC (*International Data Corporation*) Brasil com o patrocínio da Cisco do Brasil, estimou-se que existam mais de 34 milhões de conexões de banda larga na América Latina e a previsão de conexões para o Brasil em 2010 é de 15 milhões [2]. Na figura 1.1 abaixo, é possível ver que, no segundo semestre de 2009 (2H09), pelo menos 7,8 milhões de conexões de banda larga já eram conexões ADSL (*Asymmetric DSL*).

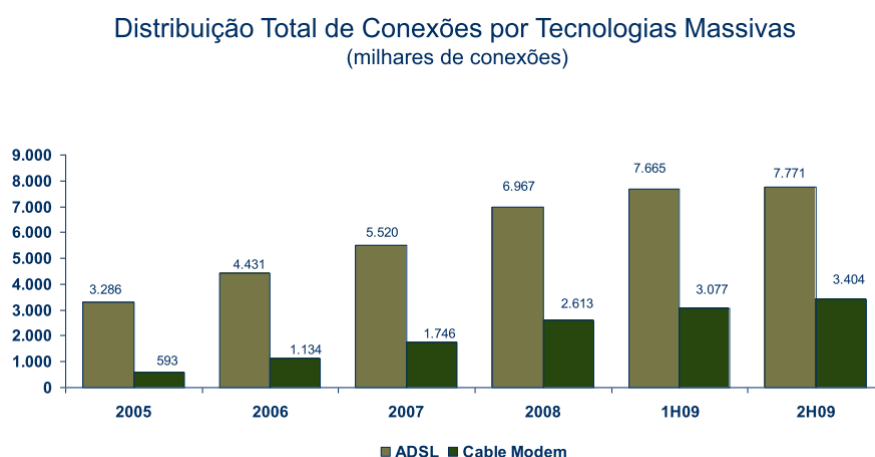


Figura 1.1 Crescimento do número de conexões ADSL no Brasil [2].

Devido à popularidade e expansão do serviço de banda larga com DSL, grandes esforços e

¹Pela IDC Brasil, um serviço de acesso à Internet é considerado de banda larga a partir de taxas superiores à 128 kbit/s [2].

estudos têm sido realizados para garantir mais qualidade e capacidade das transmissões realizadas com essas tecnologias. A evolução de suas propriedades pode ser notada pela constante preocupação por parte da indústria em atualizar os documentos de recomendação. Órgãos como IEEE (*Institute of Electrical and Electronics Engineers*), ETSI (*European Telecommunications Standards Institute*) e ITU-T (*International Telecommunication Union - Telecommunication Standardization Sector*) coordenam a publicação de tais documentos em diversas áreas da comunicação. A ITU-T é atualmente a principal responsável pela padronização das tecnologias DSL e já publicou 10 versões e 41 modificações somente para os padrões ADSL, VDSL (*Very-high-bitrate DSL*) e suas variações. Um levantamento mais detalhado dessas publicações pode ser visto na tabela 1.1 [3].

Padrão	Versões	Adaptações e correções	Período de publicação
ADSL	1	5	1999 a 2003
ADSL2	3	16	2002 a 2009
ADSL2+	3	9	2003 a 2009
VDSL	2	1	2001 a 2004
VDSL2	1	10	2006 a 2010

Tabela 1.1 Número de modificações e novas versões dos padrões publicados pela ITU-T.

Apesar desses esforços, todo cliente desse tipo de serviço sabe que eventualmente é preciso entrar em contato com a central de atendimento para registrar algum tipo de reclamação. Na lista de reclamações dos consumidores paulistas em relação ao serviço de telefonia fixa, 70% referem-se à problemas relacionados com a Internet banda larga. Entre as irregularidades mais frequentes, estão o fornecimento de taxa de dados aquém da contratada e a deficiência de sinal [4].

Os ISPs (*Internet Service Providers*), por sua vez, possuem inúmeras conexões sob sua administração, com particularidades por região, por distância, por taxa de transmissão, e assim cresce a complexidade de se alcançar um atendimento eficaz. Outro fator que agrava a ineficácia dos atendimentos, é que muitos dos equipamentos de rede nas centrais telefônicas e nas próprias residências dos clientes reportam informações pouco úteis sobre a qualidade do serviço empregado. Métricas, por exemplo, que contabilizam o número de segundos com uma quantidade mínima de erros, num determinado intervalo de tempo, são bastante comuns. Para os operadores dos ISPs, pouco dá para deduzir a partir dessas métricas, pois não é possível saber se todas as informações ou apenas alguns poucos blocos de *bytes* foram perdidos dentro daqueles segundos de transmissão.

Obviamente todo esse cenário é financeiramente desfavorável, já que clientes que não obtêm retorno em suas ligações ficam insatisfeitos, reclamam (como já comentado), protestam e cancelam os serviços contratados. Em setembro de 2009, a Fundação Procon-SP autuou a Telefônica em decorrência das panes dos serviços de Internet e voz. Só pelo serviço de Internet, a prestadora responderá 3 processos administrativos; um por cada interrupção do serviço nos meses de fevereiro, abril e maio. Ao final de cada processo administrativo, a Telefônica poderá ser multada em até R\$ 3,2 milhões [5]. Do segundo semestre de 2005 até o primeiro

trimestre de 2007, a tecnologia DSL apresentou queda de 5% em sua representação no mercado de banda larga, indicando uma tendência dos consumidores a experimentar outros tipos de tecnologia para o acesso à internet [6, 7].

Surge a necessidade, então, da criação de ferramentas de auxílio à administração dos serviços e melhoria da qualidade dos mesmos. Bem como a preocupação por informações mais precisas sobre a situação atual da linha telefônica do assinante e padronização dessas informações para os diversos fabricantes de equipamentos [8].

1.1 Objetivo do trabalho

O principal objetivo deste trabalho é desenvolver uma ferramenta de simulação que reproduza, com fidelidade, o comportamento da tecnologia ADSL2+ e que seja capaz de computar medidas de desempenho para camada física e de enlace, incluindo distribuições de probabilidade dos erros nas principais etapas do processo de transmissão de dados.

Como objetivo secundário, seu funcionamento deve depender da especificação de parâmetros de configuração usuais aos DSLAMs (*DSL Access Multiplexers*) e das características exclusivas da linha telefônica em análise, como: extensão dos cabos de cobre até a central telefônica e ruídos presentes. Em terceiro lugar, objetiva-se dar suporte ao processo da escolha dos parâmetros de proteção para garantir melhores níveis de qualidade de serviço.

1.2 Visão geral do trabalho

O presente trabalho segue com a revisão bibliográfica no Capítulo 2, intitulado Tecnologias DSL, explicando parte do funcionamento do DSL e trabalhos relacionados. No Capítulo seguinte, o desenvolvimento deste trabalho que descreve em detalhes o simulador. A validação do simulador aparece no Capítulo 4, seguida pelos resultados obtidos em testes. Por fim, a conclusão do trabalho no Capítulo 5.

CAPÍTULO 2

Tecnologias DSL

Neste Capítulo será explicado gradualmente o *modus operandi* das tecnologias DSL. Na seção 2.1, uma breve introdução ao DSL é apresentada. A partir daí, nas seções seguintes, as principais técnicas de telecomunicações e os algoritmos utilizados para contornar problemas do canal telefônico (e melhor utilizá-lo), serão apresentados com um leve rigor matemático afim de situar o leitor sobre o funcionamento geral da transmissão de dados.

2.1 Introdução ao DSL

A sigla DSL (Digital Subscriber Line) faz referência a uma série de tecnologias que provêm a transmissão digital de dados por meio dos cabos de uma rede de telefone local. O seu desenvolvimento começou na década de 80 como parte da especificação da ISDN (*Integrated Services Digital Network*) publicada em 1984 pelo CCITT (*Comité Consultatif International Téléphonique et Télégraphique*) e pela ITU (*International Telecommunication Union*). O primeiro padrão DSL orientado aos consumidores é o ADSL (*Asymmetric DSL*), que possui essa denominação por garantir taxas de transmissão diferentes para as direções *downstream* (ISP-Residência) e *upstream* (Residência-ISP). Desenvolvido pelos empregados da Bellcore em 1988, a tecnologia coloca sinais digitais de banda larga acima da banda existente do sinal analógico de voz sobre as instalações convencionais de cabeamento de par trançado das companhias telefônicas [9]. A figura 2.1 mostra um diagrama simples das instalações de um sistema DSL.

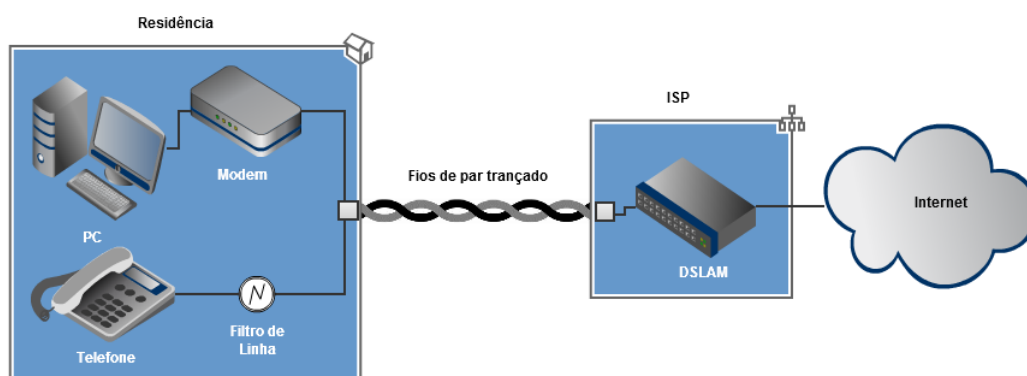


Figura 2.1 Diagrama DSL.

Como os serviços compartilham do mesmo meio físico, não pode haver interferência entre eles. Assim a tecnologia DSL precisa trabalhar em frequências não contidas no intervalo de

200 Hz a 3.4 kHz, sabida como faixa média da voz humana e, conseqüentemente, faixa de operação do telefone [10, p.120]. Já as instalações do telefone na residência precisam incluir um filtro de linha, que vai filtrar o conteúdo do sinal referente a telefonia, como mostra a figura.

Fora dessa faixa de frequência, por conta da forma como foram projetados os circuitos de telefonia, a atenuação (perda gradual de intensidade do sinal) percebida é alta e cresce a medida que frequências mais altas são utilizadas. Logo esse fator precisa também ser levado em conta na definição de um processo de transmissão sobre esse meio físico.

2.2 Modulação de Amplitude em Quadratura

A primeira técnica utilizada na transmissão de dados pelo canal telefônico, sem interferências com o serviço de telefonia, é baseada na idéia de modulação. No serviço de telefonia, a voz é transmitida na mesma frequência do sinal de voz; já com a modulação é possível escolher a frequência em que os dados serão transmitidos sem impor restrições ao que se quer transmitir.

Modulação é o processo de transportar uma informação, na forma de um sinal (analógico ou digital), dentro de outro sinal que pode ser transmitido fisicamente numa frequência desejada. O processo consiste em alterar uma ou mais características do sinal portador de acordo com o sinal modulador.

Por razões práticas envolvidas com a transmissão física, o sinal portador normalmente assume a forma de uma onda senoidal, ou seja, seus valores no tempo seguem uma função Seno ou Cosseno. Podemos, então, escrever a função do sinal portador no domínio do tempo da seguinte forma:

$$f(t) = A \cos(2\pi ft + \theta) \quad (2.1)$$

Existem três propriedades básicas da forma de onda do sinal portador que podem ser controladas afim de carregar alguma informação, são elas a amplitude (A), a fase (θ) e a frequência (f). No caso específico da modulação em amplitude, o sinal modulador, que é a informação a ser transmitida, controla o parâmetro A da função 2.1, que também vai variar no tempo. É possível reescrever a função 2.1 assim:

$$f(t) = A(t) \cos(2\pi ft + \theta) \quad (2.2)$$

No outro lado da comunicação, basta medir no sinal recebido a variação da amplitude máxima de cada ciclo no tempo e o sinal formado será a função $A(t)$, ou seja, a informação transmitida. A figura 2.2 mostra um sinal portador (*Carrier*) senoidal, sendo modulado por um exemplo de sinal modulador (*Signal*) a ser transmitido.

Nas tecnologias DSL, escolheu-se trabalhar com a modulação de amplitude em quadratura, QAM (*Quadrature Amplitude Modulation*), que é um tipo de modulação em amplitude, mas com dois sinais portadores, ou duas portadoras, transmitidos juntos. As duas portadoras são transmitidas na mesma frequência, mas defasadas de 90° uma da outra, o que torna os sinais ortogonais, ou seja, se uma das portadoras é um cosseno, a outra é um seno. Reformulando a equação 2.2, podemos representar um sinal QAM da seguinte maneira:

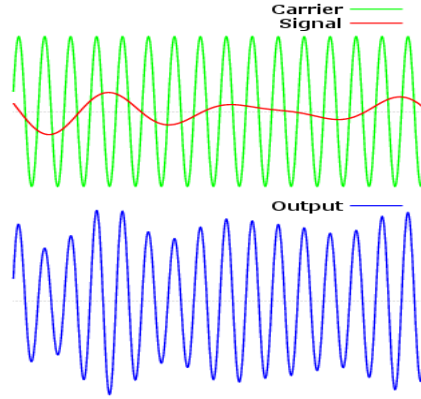


Figura 2.2 Exemplo de modulação em amplitude[11].

$$S(t) = I_i(t) \cos(2\pi ft) + Q_i(t) \sin(2\pi ft) \quad (2.3)$$

Cada portadora é modulada por uma função distinta e no outro lado da comunicação o sistema deve ser capaz de recuperar as duas funções $I_i(t)$ e $Q_i(t)$. Podemos reformular a equação 2.3 em função de uma única senóide, o que demonstra que o caso QAM é semelhante ao caso de uma única portadora. Reformulando, teremos:

$$S(t) = A_i(t) \sin(2\pi ft + \theta_i) \quad (2.4)$$

Onde:

$$A_i = \sqrt{I_i^2 + Q_i^2}, \tan(\theta_i) = \frac{I_i}{Q_i}$$

Como as duas funções são ortogonais, a demodulação do sinal QAM é feita multiplicando as mesmas funções cosseno e seno, para obter as informações de I e Q respectivamente, e depois filtrando as componentes de alta frequência geradas pela relação trigonométrica: $\sin(2a) = 2 \sin(a) \cos(a)$. Mais detalhes da demodulação podem ser vistos em [10, p.153].

Quando as partes da comunicação acordam em trabalhar com *streams* binárias, ou seja, quando a informação transmitida é digital, apenas alguns valores de amplitude são válidos; deixamos de ter um sinal contínuo modulando o sinal portador e passamos a modular discretamente. Nesse cenário um diagrama útil para a interpretação do QAM, é sua constelação.

Os valores de amplitude I e Q são as coordenadas de um plano cartesiano, e todos os possíveis pontos indicados no plano formam a constelação QAM. As constelações QAM podem ser referenciadas por M-QAM, onde $M = 2^b$ e b é o número de *bits* utilizado na modulação. Um exemplo de uma constelação QAM é mostrado na figura 2.3.

Cada ponto no plano representa na comunicação uma sequência específica de *bits* e é frequentemente chamado de símbolo QAM. O primeiro passo no processo de modulação é a codificação dos *bits* em símbolos [10, p. 148]. Essa codificação é a escolha de quais pontos na constelação vão representar quais sequências de *bits* e é frequentemente chamada de mapeamento ou código. Existem mapeamentos bastante conhecidos como o caso do *Gray Mapping*

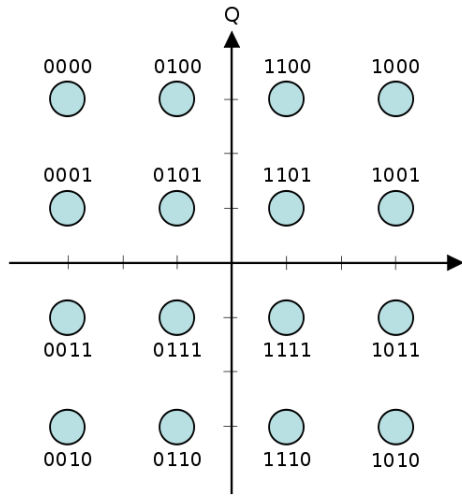


Figura 2.3 Exemplo de uma constelação 16-QAM[12].

e sua regra de mapeamento pode ser encontrada em [10, tabela 6.1 e 6.2, p.149], mas no caso das tecnologias DSL, os seus padrões de implementação definem mapeamentos próprios [13, p. 93].

Toda comunicação está suscetível a erros e esses normalmente são causados por alguma fonte externa que modifica, *a priori*, aleatoriamente pelo menos uma das características do sinal que está sendo transmitido. Na etapa da modulação QAM um erro ocorre quando há uma interpretação equivocada do símbolo transmitido, ou seja, as amplitudes I e Q recebidas são diferentes das enviadas, o que, dependendo das proporções, invalida a informação original.

A escolha da quantidade de *bits*, o mapeamento usado e o grau de separação entre os símbolos podem alterar a probabilidade com que os erros de símbolo acontecem. A probabilidade de erro da demodulação de um símbolo QAM que carrega um número par de *bits* e está sob efeito de um ruído gaussiano, pode ser calculada assim [14]:

$$P_{\text{symb}} = 4 \left(1 - \frac{1}{\sqrt{M}} \right) Q \left(\sqrt{\frac{3\text{SNR}}{(M-1)}} \right) \quad (2.5)$$

Onde M é a quantidade de pontos na constelação QAM, como já explicado; o SNR (*Signal-To-Noise Ratio*) é a relação da potência do sinal transmitido e a potência do ruído presente na linha; e $Q(x)$ (não confundir com a representação da modulação QAM) é a função que calcula a probabilidade que uma variável aleatória normal, com média igual a zero e desvio padrão igual a um, irá obter um valor maior que x , representada na figura 2.4.

Quanto maior o número de *bits* utilizados na modulação, mantendo-se inalterado o SNR, maior o M , menor o argumento x da função Q e, logo, maior a probabilidade de erro de demodulação. Essa é uma das razões pela qual no planejamento do ADSL2 e alguns padrões DSL, por exemplo, impôs-se um limite de 15 *bits* [13, p. 44], o que já produz uma constelação com 2^{15} possíveis pontos. Uma análise detalhada das probabilidades de erro para os diferentes cenários desses parâmetros é o tema do trabalho em [14].

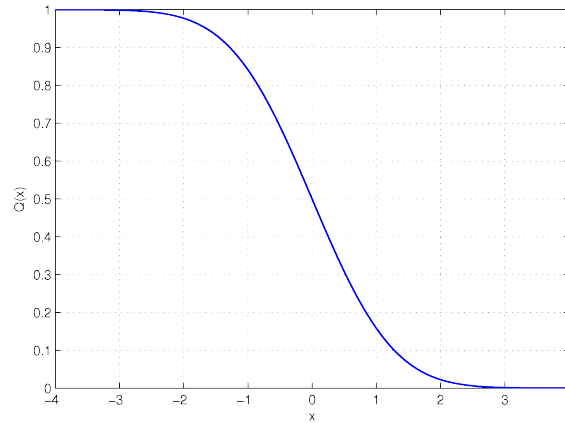


Figura 2.4 Representação da função Q[15].

2.3 Multiplexação Por Divisão em Frequência Ortogonal

Como observado na seção anterior, a modulação QAM permite a transmissão de dados em torno de uma única frequência f (ver equação 2.3) e para transmitir dados em taxas mais altas do que 15 *bits* por símbolo, seria necessário aumentar a potência da transmissão, aumentando o SNR, para que a probabilidade de erro não caísse. Porém, de acordo com o teorema de Shannon [16, p. 389], a capacidade máxima de transmissão de informações de um canal com banda limitada é :

$$C = B \log\left(1 + \frac{S}{N}\right) \quad (2.6)$$

Onde C é a capacidade do canal, B é banda espectral alocada, S é a potência empregada no sinal a transmitir e N é a potência do ruído presente no canal. Logo, é possível observar que C cresce mais rápido quando se aumenta B , utilizando para transmitir mais portadoras em mais frequências, do que quando se aumenta S .

Diante do exposto, é mais viável, se possível, investir em uma transmissão que utiliza uma banda maior e para isso uma outra técnica de comunicação digital complementa a base da transmissão DSL, que é a multiplexação por divisão em frequência ortogonal (*Orthogonal frequency-division multiplexing* - OFDM). Também conhecida por DMT (*Discrete Multi-Tone modulation*), é um tipo de multiplexação que permite transmitir simultaneamente vários sinais, atribuindo para cada sinal faixas de frequência bem definidas não-sobrepostas e de forma que as frequências escolhidas tornam as portadoras de cada faixa ortogonais entre si.

Como as portadoras são ortogonais entre si, em cada uma delas é possível transmitir uma portadora QAM distinta, sem qualquer interferência em suas faixas vizinhas. Nesse cenário, cada portadora QAM alocada é chamada de sub-portadora. O conjunto com todas as sub-portadoras já moduladas e combinadas no tempo, formam outro sinal e que é frequentemente chamado de símbolo OFDM ou DMT. A figura 2.5 ilustra o espectro de um sistema OFDM; note que por conta da ortogonalidade o espectro contínuo de uma sub-portadora assume o valor zero exatamente na frequência de pico das sub-portadoras vizinhas.

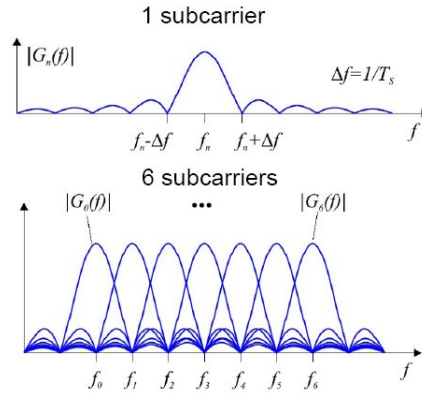


Figura 2.5 Sub-portadoras no esquema OFDM [17].

O processo de representar um determinado sinal, que está originalmente descrito num domínio em outro, é conseguido através do operador matemático chamado de transformada. A transformada mais usada para transcrever sinais no domínio do tempo, perturbações do meio físico, em componentes frequenciais é a transformada de Fourier, que é descrita assim:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t) e^{-j2\pi\xi t} dt, \text{ para todo } \xi \text{ real.} \quad (2.7)$$

A variável t representa o tempo em segundos e a variável ξ a frequência em hertz (Hz). Como é possível perceber, para uma representação perfeita de um sinal no domínio da frequência, é preciso toda a sua descrição no tempo, de $-\infty$ a ∞ . Como sempre há limites físicos de largura de banda e requisitos da transmissão, como atraso máximo, estamos interessados na transformada discreta de Fourier de curta-duração. Discretizando a equação 2.7 e assumindo uma amostragem da função no domínio do tempo para um determinado intervalo, temos:

$$X_k = \sum_{n=0}^{T-1} x_n e^{-j\frac{2\pi kn}{T}} \quad (2.8)$$

Onde T é o número de amostras do sinal que será analisado e k é a frequência da componente calculada. Devido a forma como a transformada é descrita, utilizando a exponencial como base do espaço, uma propriedade importante pode ser observada. Da identidade de Euler, podemos afirmar que:

$$e^{-j2\pi\alpha} = \cos(2\pi\alpha) + j\sin(2\pi\alpha) = 1 + 0 = 1, \text{ para } \alpha \text{ inteiro.} \quad (2.9)$$

Assim, fixando um k e escolhendo $k' = T - k$, podemos observar a seguinte simetria da transformada:

$$\begin{aligned}
e^{-j\frac{2\pi k'n}{N}} &= e^{-j\frac{2\pi(N-k)n}{N}} \\
&= e^{-j\frac{2\pi Nn - 2\pi kn}{N}} \\
&= e^{j\frac{2\pi kn}{N}} e^{-j2\pi n} \\
&= e^{j\frac{2\pi kn}{N}}.
\end{aligned}$$

O que mostra que $X_k = X_{N-k}^*$. Esse fato é utilizado na implementação do OFDM, pois para transmitir, por exemplo, dados modulados em S sub-portadoras, serão necessários $2S$ valores no cálculo, como mostrado em [10, p. 187]. No esquema da OFDM, cada X_k é um número complexo e representa uma sub-portadora, portanto é igual a um ponto na constelação da modulação QAM: $X_k = I_k - jQ_k$. Um transmissor OFDM vai calcular a inversa da transformada 2.8 para enviar o símbolo DMT e o receptor vai calcular a própria transformada 2.8, demodulando o símbolo DMT e recuperando os valores das sub-portadoras QAM.

2.4 Algoritmo de Alocação de *bits*

Como explicado até agora, várias sub-portadoras serão espalhadas por uma banda, que deve se iniciar após as frequências do serviço de telefonia, e para cada sub-portadora uma faixa de frequência será separada. No padrão ADSL2+, por exemplo, as faixas de frequência são espaçadas de $\Delta f = 4.3125 \text{ kHz}$ e sua banda começa em 4.3125 kHz e estende-se até 2.2 MHz [18]. A banda é dividida em no máximo 512 sub-portadoras, compartilhadas entre as direções *upstream* e *downstream*.

A quantidade de *bits* que cada sub-portadora irá transportar é decidido a partir de um algoritmo de alocação de *bits*. A fórmula do número de *bits* alocados na portadora k , que pode ser obtida a partir da equação 2.6 [10, p.166, p.204], dada uma probabilidade de erro desejada Pe , é a seguinte:

$$B_k = \log_2 \left(\frac{SNR_k \gamma_c}{\Gamma_{Pe} \gamma_m} + 1 \right) \quad (2.10)$$

Onde SNR_k é a relação sinal-ruído da sub-portadora k , γ_c é o ganho de codificação devido ao uso de códigos FEC (*Forward Error Correcting Codes*, ver seção 2.6), γ_m é uma margem de segurança, conhecida como SNR_{margin} ou *Noise margin*, que diminui o número de *bits* para diminuir a probabilidade de erro; e o Γ_{Pe} , chamado de *Shannon Gap*, é o argumento necessário para baixar a quantidade de *bits* a um nível que garanta, com base na equação 2.5, que a probabilidade do erro de *bit* fique abaixo de 10^{-7} (BER ou *bit error rate*) [10].

2.5 Ruídos presentes na comunicação

Nas seções anteriores foi citada a presença de ruídos na linha e sua participação em algumas formulações, mas ainda nada foi explicado sobre eles. Em toda comunicação existem ruídos, que podem ser vistos como perturbações não programadas que causam efeitos indesejados ao sistema. Esses ruídos impedem a transmissão em taxas de transmissão de dados altíssimas com alta qualidade; comportamento representado pelo limite teórico formulado por Shannon, ver equação 2.6. Nos sistemas DSL, os ruídos mais comuns são: ruído térmico, ruído impulsivo e *crosstalk*.

O ruído térmico é causado pela natureza física dos circuitos do canal telefônico, que aquecem (como o termo sugere) e alteram os valores que passam pelo canal. Sua representação no domínio da frequência é quase plana, o que significa, que seu efeito é praticamente igual em todas as frequências. No domínio do tempo sua amplitude assume os valores de uma gaussiana e por possuir efeito aditivo e espectro plano é chamado de AWGN (*Additive White Gaussian Noise*).

O ruído impulsivo é uma interferência eletromagnética não-estacionária que consiste em ocorrências aleatórias de picos de energia, pulsos, com amplitude aleatória e conteúdo espectral. Também pode surgir em rajadas, vários pulsos seguidos, e nesse caso normalmente é chamado de REIN (*Repetitive Electrical Impulse Noise*). As causas de suas ocorrências são diversas e variam de eventos de colocar/retirar o gancho do telefone, ruídos das instalações da rede elétrica de casa ou do escritório, veículos de transporte, até ruídos atmosféricos causando descargas elétricas [19]. Um exemplo da forma de onda de um pulso característico de ruído impulsivo pode ser visto na figura 2.6.

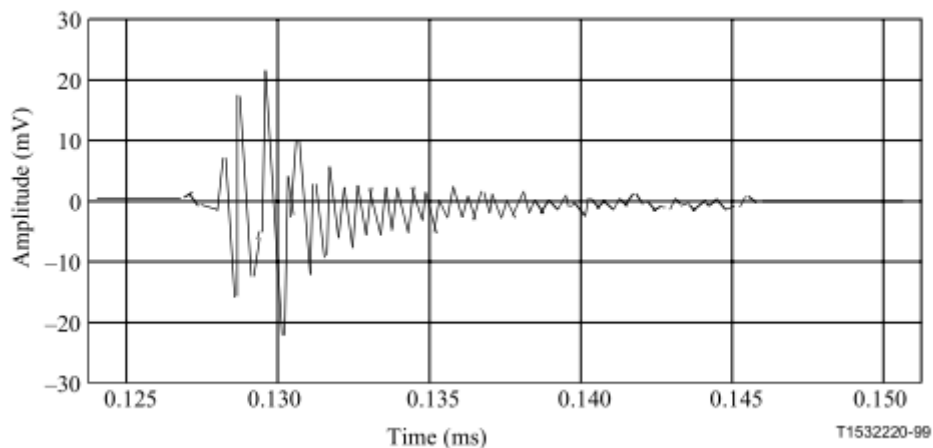


Figura 2.6 Forma de onda de um ruído impulsivo de 1 rajada[20].

Normalmente os pulsos que compõem o ruído impulsivo são de curta duração, o exemplo da figura acima tem duração total de 0.02 ms, o que corresponde à algo em torno de 10% da duração do símbolo DMT especificado nos padrões DSL [13].

O ruído de *crosstalk*, que traduzido significa conversa cruzada, é causado quando mais de

uma linha de uma mesma central ou região operando com potências altas causam interferências umas nas outras, induzindo correntes indesejadas que alteram os valores originais que passam pelo canal telefônico. Existem dois tipos de *crosstalk*, o NEXT (*Near End Crosstalk*) e o FEXT (*Far End Crosstalk*). O NEXT ocorre quando um transmissor num dos lados interfere nas linhas desse mesmo lado e já o FEXT ocorre quando um transmissor de um lado induz correntes nos pares de fio do outro lado da comunicação. A figura 2.7 ilustra o comportamento explicado.

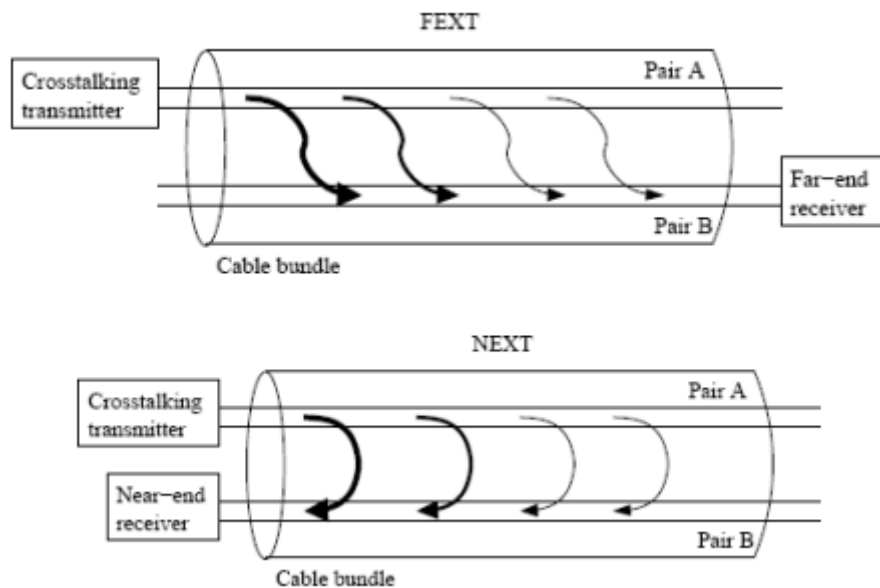


Figura 2.7 *Crosstalk* FEXT e NEXT [19].

No presente trabalho apenas os ruídos térmico e impulsivo são utilizados por afetarem cada linha de uma central independentemente. O efeito do ruído térmico pode ser mensurado por sua variância ou sua potência em dBm/Hz. Já o ruído impulsivo pode ser identificado pelo tamanho de sua rajada, mesmo no caso de tamanho 1, e sua taxa de repetição no tempo (ex: 50 repetições por segundo ou 50 Hz).

2.6 Códigos corretores e Reed-Solomon

Como comentado anteriormente, toda comunicação está suscetível a erros, causados ou não por ruídos presentes, e, dependendo de quão severos esses são, a comunicação pode até mesmo não se concluir. Para isso, toda uma área de conhecimento foi criada para desenvolver códigos que corrigissem ou, pelo menos, detectassem esses erros.

Um código é uma regra para converter um pedaço de informação (por exemplo uma letra, uma palavra, um *bit*) em outra forma de representação não necessariamente do mesmo tipo da original. Códigos corretores nada mais são do que aqueles códigos que se preocupam em transformar a representação da informação em algo que possua características próprias e que lembre a informação original. Assim se o código gerado chega no outro lado da comunicação

ponto a ponto com poucos ou alguns erros, ainda será possível interpretar o que era pretendido transmitir.

Para entender melhor, considere o seguinte exemplo. Imagine um satélite no espaço que se comunica com alguma estação na terra. Isto é, ele recebe ou envia mensagens através do espaço. Suponha que a estação envia uma mensagem binária para ele, por exemplo 00110. Assim essa mensagem é transformada por um modulador que produz ondas que o satélite vai receber e demodular para recuperar a mensagem. A idéia é que interferências no espaço podem provocar interpretações erradas na demodulação e que por sua vez causa *bits* errados na mensagem recebida. Ainda com o exemplo de que 00110 foi enviada, uma mensagem errada 10110 pode ter sido recebida. Como uma retransmissão consome um tempo razoável, um jeito mais seguro de enviar é usando algum código corretor. Adicionando redundância na mensagem, é possível detectar ou mesmo recuperar a mensagem original. Por exemplo, se todo *bit* da mensagem puder ser repetido 3 vezes, então a mensagem 00110 será codificada em 000 000 111 111 000. Desse jeito, se 010 000 111 111 000 for recebida a mensagem correta pode facilmente ser decodificada selecionando o *bit* com maior ocorrência em cada tripla [21, p. 8].

O código corretor usado nas tecnologias DSL, é o *Reed-Solomon*, ou simplesmente código RS, que é um código do tipo FEC, *forward error correcting*, que funciona adicionando redundância em blocos de informações. Ele também é usado na correção dos erros em mídias de CD e DVD devido à sua performance em erros mais dispersos, como riscos ou arranhões.

O *Reed-Solomon* trabalha atrelando a informação a ser codificada à um polinômio, onde cada *byte* num bloco de informações de tamanho definido, digamos de K bytes, representa um dos coeficientes do polinômio. A idéia é avaliar o polinômio em mais do que K pontos, digamos N , e usar o resultado como o código. Assim mesmo que alguns *bytes* ou pontos cheguem errados, é possível encontrar os coeficientes do polinômio grau K que melhor representam aquele conjunto de N pontos. O tamanho do código final pode ser expresso como:

$$N = K + R \quad (2.11)$$

Onde R é o número de pontos que foram avaliados além do número mínimo K . R é justamente a redundância incluída pelo código.

Uma noção importante para interpretação de códigos é a idéia de espaço das mensagens e dos códigos. Utilizando o exemplo do satélite, a mensagem 00110 está no espaço unidimensional, pois cada elemento da mensagem pode ser localizado em única dimensão. Enquanto que o código 000 000 111 111 000, está no espaço tridimensional e cada possível código é um ponto de um cubo.

Nos códigos RS, as mensagens estão no espaço q^K , onde q é número de possíveis valores da mensagem e os códigos estão num espaço q^N . Como o espaço dos códigos é sempre maior que o espaço de mensagens e a função de codificação é uma função injetora, existem pontos no espaço de códigos que possuem correspondentes no espaço de mensagens e outros que não.

Os códigos são construídos de forma que os pontos válidos do espaço, ou conjunto imagem da função de codificação, estejam distantes¹ por no mínimo $R + 1$ bytes uns dos outros. Assim, tais códigos são capazes de recuperar até $\frac{R}{2}$ bytes de erros, usando a decodificação do ponto

¹Entendendo como distância o número de *bytes* diferentes entre os códigos.

recebido pelo ponto válido mais próximo. Note que para cada escolha de K e R um esquema de códigos diferente é montado.

Outro detalhe dos códigos de *Reed-Solomon*, é que seus polinômios são avaliados sobre no corpo finito de Galois de 256 elementos [13, p. 49]. Corpos são estruturas matemáticas se comportando com as mesmas regras da aritmética usual e próxima à nossa intuição cotidiana. Um corpo é um conjunto de elementos onde as operações de adição, subtração, multiplicação ou divisão estão sujeitas as leis da comutatividade, distributividade, associatividade e outras propriedades "usuais"[21, p. 32]. Um corpo finito possui a mesma estrutura, só que com uma quantidade de números finita. Note que para isso ser possível é necessário o uso de alguma noção modular às operações. No corpo finito de Galois(256), todo número pode ser representado por 8 *bits* ou 1 *byte* ($256 = 2^8$).

Como os códigos *Reed-Solomon* possuem essa limitação de correção até $\frac{R}{2}$ *bytes* de erros, erros concentrados em *bytes* próximos na *stream* da comunicação, não conseguem ser completamente corrigidos. Para evitar isso, uma técnica, chamada *interleaving*, é aplicada para reorganizar os *bytes* antes do envio.

O *Interleaving* é um método para rearranjar os dados de forma não contínua, afim de espalhar os erros sobre os vários blocos de informação, aumentando assim a performance dos códigos precedentes à ele. Nos sistemas ADSL2 e ADSL2+, por exemplo, o *Interleaving* é controlado por um parâmetro de proteção D (*Interleave Depth*) que informa o quão profunda será a rearrumação dos *bytes* e funciona assim: cada *byte* i de um bloco de N *bytes*, onde $0 \leq i \leq N - 1$, é atrasado ou deslocado de $(D - 1) * i$ posições [13, p.50]. Nesse esquema, após o *interleaving*, os *bytes* originalmente de um mesmo bloco ficam distantes entre si de D posições. Esse método tem como contrapartida a inserção de uma atraso na comunicação dada a necessidade de embaralhar e desembaralhar os blocos.

Os valores R e N do código *Reed-Solomon* e o valor D , do *Interleaving*, são passados como parâmetros para o bloco de *framing* no processo de transmissão dos sistemas DSL. Cada padrão DSL define os valores possíveis para cada parâmetro de *framing*, assim chamados R, D e N . A tabela 2.1 mostra os valores permitidos para cada parâmetro no padrão ADSL2+ [18]. Os valores para N foram obtidos depois uma certa manipulação com outros parâmetros irrelevantes ao presente trabalho.

Parâmetro	Valores válidos
N	$0 \leq N \leq 255$
R	0, 2, 4, 6, 8, 10, 12, 14 ou 16
D	1, 2, 4, 8, 16, 32, 64, 128 e 256. Se $R = 0$ então $D = 1$

Tabela 2.1 Valores de R e D permitidos pelo padrão ADSL2+.

Ao final da aplicação de um código corretor a um sistema de comunicação, existe uma medida que informa o quanto os dados ficaram protegidos, essa medida é o *coding gain* ou ganho de codificação. O ganho de codificação, no DSL, é medido baseado na seguinte pergunta: *Quantos bits é possível adicionar a cada sub-portadora sem que, devido ao processo de correção posterior, se altere a probabilidade de erro final?*. Essa medida, normalmente mensurada

em dB, entra como parâmetro no algoritmo de alocação de *bits*, como já comentado na seção 2.4.

2.7 Sincronização DSL

Nos sistemas DSL existem, fundamentalmente, duas unidades envolvidas no processo todo: uma no lado da central telefônica, chamada de DSLAM (*DSL Access Multiplexer*), e uma unidade no lado do cliente, conhecida simplesmente como modem (*Modulator Demodulator*).

Uma fase importante no *modus operandi* dos sistemas DSL, é a sincronização do DSLAM com o modem antes de iniciar a transmissão. Nessa fase, as unidades trocam mensagens, seguindo um protocolo padrão, e ao final decidem os valores de todos parâmetros e como a partir daquele momento a transmissão ocorrerá. Acontecem na sincronização, principalmente, a execução do algoritmo de alocação de *bits* e a escolha dos parâmetros de *framing*.

Como o algoritmo de alocação de *bits* depende do *SNR* percebido no outro lado da comunicação, então a unidade que executa esse algoritmo é a unidade receptora; que ao final envia o resultado à unidade transmissora. No caso *downstream*, é o modem que faz esse papel, e no caso *upstream*, o DSLAM.

Existem alguns parâmetros que são ajustados diretamente no DSLAM que influenciam no algoritmo de alocação de *bits* e na escolha dos parâmetros de *framing*. No algoritmo de alocação o principal parâmetro configurável é o *SNR margin*, que semanticamente representa o quanto o ruído recebido pode aumentar (em dB) em relação à potência de ruído que o sistema é projetado para tolerar[13], sua contribuição no algoritmo pode ser observada na equação 2.10. Já na escolha dos parâmetros de *framing*, os principais são: o nível de proteção contra ruídos impulsivos (*Minimum Impulse Noise Protection - INP_{min}*) e o atraso máximo tolerado (*Delay_{max}*). O *INP_{min}* expressa quantos símbolos DMT, no mínimo, estão protegidos contra erros consecutivos dentro de uma janela com a quantidade de símbolos que comportam *D* blocos FEC, e sua formula é expressa por:

$$INP_{min} \leq \frac{4RD}{L} \quad (2.12)$$

Onde *R* e *D* são os parâmetros de *framing*, já comentados na seção 2.6, e *L* é o somatório de *bits* transmitidos em cada uma das sub-portadoras QAM do símbolo DMT. Já o *Delay_{max}* depende do *interleaving*, que espalha as informações e na recepção dos dados precisa esperar mais símbolos DMT para completar os blocos FEC. Sua fórmula é:

$$Delay_{max} \geq \frac{2DN}{L} \quad (2.13)$$

Assim os parâmetros de *framing* escolhidos precisam atender a essas duas restrições.

2.8 Trabalhos relacionados

De uma forma geral, em pesquisas de sistemas de comunicação (o que inclui os sistemas DSL), devido ao custo de obter equipamentos especializados na área, a simulação é uma alternativa

de avaliação que deve ser fortemente considerada [22, 23].

Um exemplo de pesquisa que utiliza essa alternativa é o trabalho desenvolvido em [24]. Cujo objetivo foi a qualificação das linhas ADSL2+ com relação a taxa de *bits*. Outras tecnologias DSL também entraram no projeto, mas aparentemente não puderam ser validadas. A simulação é baseada em modelos analíticos de atenuação para algumas configurações possíveis de linha, incluindo tipos diversos de cabos de par trançado, e três modelos analíticos dos efeitos do ruído *crosstalk*. O trabalho se restringe a qualificar as linhas ADSL2+, não avaliando aspectos da transmissão de dados mesmo.

Outro exemplo do uso dessa técnica de avaliação nas tecnologias DSL, é o modelo de simulação para ADSL implementado no ambiente *Simulink* do MATLAB e incluído no pacote de funções e blocos de comunicação intitulado: *Communications Toolbox* [25, 26]. Nele toda a transmissão é simulada, desde da geração e envio dos dados até o recebimento e contagem dos erros. Como a maioria dos modelos criados no *Simulink*, o simulador tem um ótimo desempenho, simulando o envio de 6.6×10^6 *bits* em 30 segundos e apresenta algumas medidas de desempenho enquanto a simulação ocorre, como o BER (*bit Error Rate*) e o FER (*Frame Error Rate*). Apesar de sua boa performance, a interface de modificação dos parâmetros é complicada (é necessário acessar cada bloco específico e mudar suas configurações), a maioria dos parâmetros de configuração oferecidos não estão acessíveis na prática dos ISPs e o modelo assume apenas a existência do ruído AWGN.

Descrição do Simulador

Neste Capítulo os detalhes de como foi desenvolvido o simulador serão expostos. Na seção 3.1 a base conceitual do simulador é discutida e nela uma visão geral dos módulos implementados é apresentada. Nas seções seguintes (3.2 a 3.8) são descritos os algoritmos e detalhes adicionais da implementação no ambiente do MATLAB (*MATrix LABoratory*) [25]. Por último a seção 3.9, descreve a interface desenvolvida para controlar as entradas da ferramenta e a visualizar os resultados das simulações.

Na implementação da ferramenta como um todo, além das funções padrão do MATLAB, foram utilizadas algumas funções pertencentes ao *Communication Toolbox* (CT) [26], um pacote de funções especialmente desenvolvidas para trabalhar com sistemas de comunicação. Por simplificação, deste ponto em diante serão referenciadas como funções do CT.

3.1 Base Conceitual

Para alcançar o objetivo proposto do presente trabalho, ver seção 1.1, o simulador precisa ser desenvolvido tentando aproximar o comportamento de um sistema ADSL2+ real; e para isso todos os conceitos vistos no Capítulo 2 precisam ser implementados. Para garantir, especificamente, a capacidade de computar medidas de desempenho na camada física a partir de mudanças nas configurações de proteção, é necessário que parâmetros como INP_{min} , $Delay_{max}$ e SNR_{margin} estejam disponíveis para manipulação e que os processos internos do simulador estejam de acordo com as restrições por eles impostas. Vale lembrar que, devido a importância comercial da transmissão dos dados na direção da central até o cliente, optou-se por desenvolver esta ferramenta apenas no sentido *downstream*, mas todas as etapas da transmissão no sentido *upstream* podem ser trivialmente adaptadas a partir dos módulos implementados neste trabalho. Seguindo as seções do Capítulo 2, a figura 3.1 sintetiza os módulos de execução implementados no simulador.

Os módulos estão estruturados em 5 blocos maiores, são esses: Etapas de sincronização, Transmissor, Meio físico, Receptor e Cálculo de resultados. A ordem em que estão organizados na figura 3.1 sugere a ordem natural da transmissão nos sistemas DSL. No primeiro bloco se encontram os módulos onde as configurações iniciais são definidas e sincronizadas entre o modem e o DSLAM. Na seção 3.2, cada módulo desses é discutido detalhadamente. No segundo e quarto bloco, podemos encontrar os módulos que executam os algoritmos pertinentes ao transmissor e receptor dos dados, respectivamente. Nas seções 3.3, 3.4, 3.5 e 3.6, a ida e a volta de cada um dos módulos da camada física do sistema ADSL2+ estão esmiuçados, afim de garantir o entendimento pleno do leitor. No terceiro bloco está o módulo responsável pelo

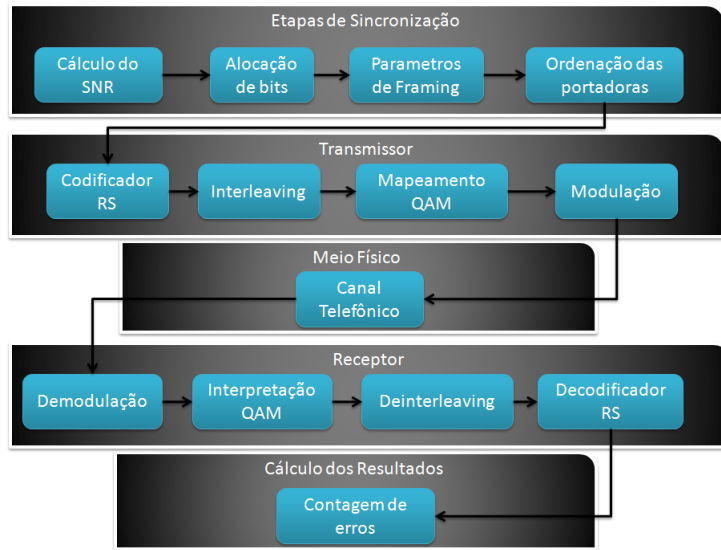


Figura 3.1 Módulos do simulador.

meio físico em si, principalmente no que diz respeito à aplicação dos ruídos e é visto na seção 3.7. Último bloco e módulo se refere à contagem dos erros dentro da ferramenta ao final de cada simulação, seus pormenores são apresentados na seção 3.8.

Elementos aleatórios foram introduzidos no processo para garantir a validade dos resultados, já que, como num sistema real, a existência deles é inevitável. Alguns desses elementos aleatórios são: *stream* de *bits* a ser transmitida, atenuação exata a ser inserida em cada faixa de frequência, potência exata do ruído térmico presente na linha e momento exato em que ruídos impulsivos não-estacionários afetam o símbolo DMT. Todos os eventos acima, exceto a geração da *stream* de dados, variam a partir de um patamar mais um valor sorteado de uma distribuição normal com $\mu = 0$ e $\sigma = 1$.

Por opção, não foram implementados módulos com execução em paralelo, assim uma etapa só começa sua execução após o término da anterior. O que originalmente contraria um pouco a execução natural dos sistemas de comunicação, mas não afetará os resultados obtidos, uma vez que os algoritmos individuais são executados exatamente como no caso de uma transmissão real e os resultados relativos à atraso dos dados não serão investigados.

3.2 Passos da Sincronização DSL

3.2.1 Cálculo do SNR

Para o cálculo do SNR de cada faixa de frequência, é preciso saber, como o nome diz, a potência final do sinal enviado e a potência do ruído presente. A potência final do sinal enviado é calculada (em dB) a partir da potência de envio menos a atenuação inserida pelo canal.

Então o primeiro passo é definir a potência de envio. Os padrões da ITU definem as potências máximas que o sinal DMT pode assumir em cada faixa de frequência, afim de evitar

interferências no serviço de telefone, no tráfego de dados na direção *upstream* (sentido modem-DSLAM), bem como interferência em outras linhas DSL da mesma central [13, 18, 1]. O conjunto desses limites, formam a máscaras de potência. A figura 3.2, mostra a máscara de potência definida para o funcionamento do DSL em linhas telefônicas.

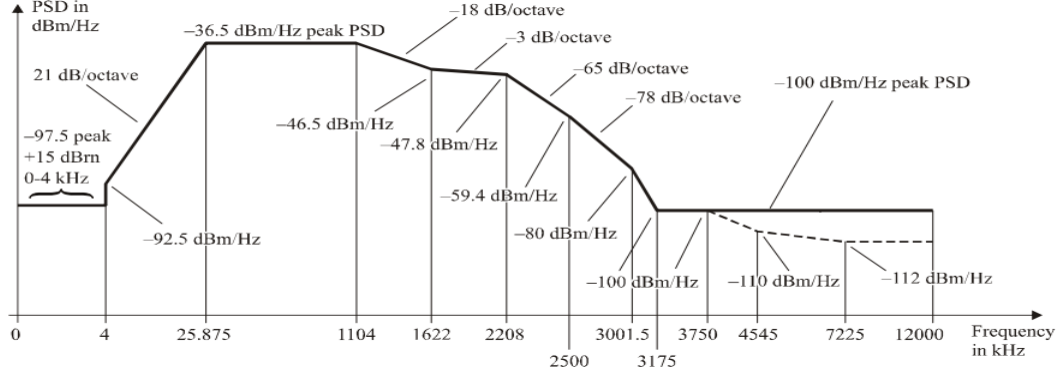


Figura 3.2 Potência limite nas faixas de operação do DSL [18].

As potências estão definidas na forma de PSD (*Power Spectral Density*) em unidades de dBm/Hz (Potência relativa à 1 miliwatt por unidade de frequência). De uma faixa para outra, é indicada a taxa de decaimento da potência em dB/oitava, denotando que a potência decai desse fator (em dB) enquanto a frequência dobra seu valor.

Além disso, os padrões especificam a existência de pontos de controle, que modelam essa máscara, livres para a implementação de cada fabricante. Para implementação dos pontos de controle na ferramenta de simulação, realizou-se experimentos com centenas de sincronizações de uma linha ADSL2+ em laboratório. Nestes experimentos variaram a distância entre a central telefônica e o cliente através de um emulador de linhas físicas. Mais detalhes sobre o ambiente de experimentação em laboratório serão dados no Capítulo 4.

O segundo passo é estimar a atenuação que o sinal DMT sofrerá, dependendo da distância escolhida para o modem do cliente, e depois, por fim, o SNR para cada faixa de frequência no momento da sincronização. Como dito na seção 2.7, o equipamento que mede a atenuação do canal depende da direção de transmissão dos dados; no caso *downstream* é o modem do cliente. Para tal estimativa, foram usados como modelos os gráficos de atenuação do emulador de linhas físicas do ambiente de experimentação. A figura 3.3 mostra a atenuação inserida para uma distância de 2km.

Os pontos dos gráficos para as distâncias de 1km, 2km e 3km foram carregados no ambiente MATLAB e depois os pontos foram interpolados com auxílio da função *Piecewise cubic Hermite interpolation*, que preserva monotonicidade e a forma dos dados [25].

Finalmente, após a obtenção dos valores da potência de transmissão e atenuação, o cálculo do SNR de cada faixa de frequência segue a fórmula abaixo:

$$SNR_i = \frac{faixa_i^{10} \left(\frac{atenuacao_i + mascara_i}{1000} \right)}{2\sigma_{ruído_i}^2}$$

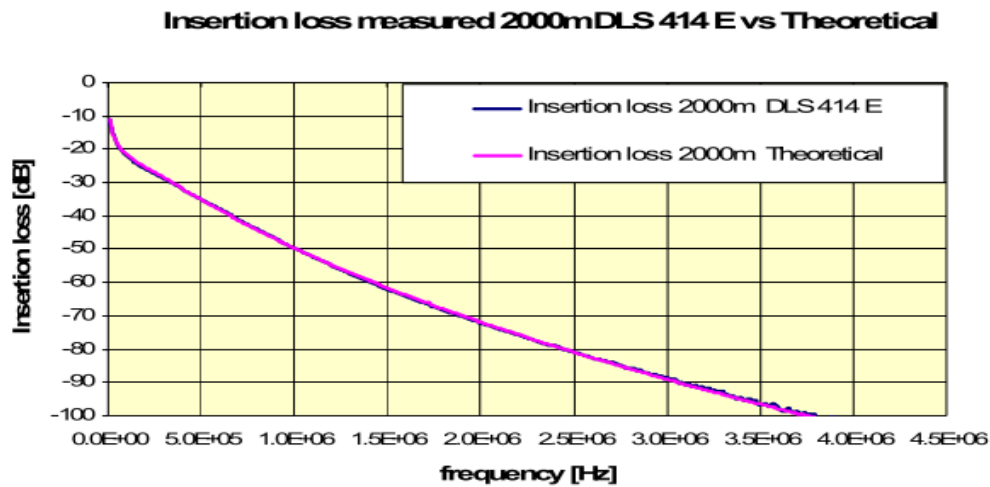


Figura 3.3 Curva de atenuação por frequência para uma linha telefônica com 2km de extensão.

A $faixa_i$ representa a banda de frequência da sub-portadora i (obtida da multiplicação do índice i pelo espaçamento frequencial mínimo definido no padrão ADSL2 [13]), o parâmetro $\sigma_{ruído_i}^2$ representa a variância configurada para o ruído na faixa i . Ao final o SNR obtido está na unidade linear, já que os valores de atenuação e potência foram convertidos.

3.2.2 Algoritmo de alocação de *bits*

Seguindo as especificações do padrão [18] e a sugestão de algoritmo em [10], o algoritmo de alocação de *bits* precisa seguir alguns passos descritos no próximo quadro. Para maiores detalhes, o código apresentado no apêndice A.1 mostra o procedimento implementado no simulador.

1. Calcula os *bits* para cada faixa de frequência
2. Calcula o ganho de potência necessário para arredondar o número de *bits*
3. Verifica se a potência ultrapassa o máximo permitido
 - (a) Se sim, arredonda para baixo
4. Recalcula os *bits* com o novo SNR
5. Verifica se alguma portadora está com mais de 15 *bits*
 - (a) Se sim, limita as portadoras a este número de *bits*

3.2.3 Escolhendo os parâmetros de framing

A escolha dos parâmetros de *framing* N , R e D devem respeitar os limites das equações 2.12 e 2.13 a partir dos valores de INP_{min} e $Delay_{max}$, como visto na seção 2.6. Entretanto, há situações em que as restrições são atendidas por mais de um conjunto de valores dos parâmetros. No desenvolvimento deste simulador, optou-se por escolher os parâmetros de *framing* N , R e D , de modo a garantir a melhor taxa líquida de dados; o que normalmente implica em diminuir a redundância R do bloco FEC.

O procedimento para escolha dos parâmetros, testa todos os valores possíveis sob as restrições e filtra aqueles que as infringem. Para tanto, primeiramente os possíveis R e D são encontrados e depois, usando esses valores, o conjunto de valores válidos para N é encontrado. Depois, dentre todos os possíveis N e R , escolhe-se aquele com menor relação $\frac{R}{N}$. No apêndice A.2, um algoritmo que implementa essas escolhas é apresentado.

3.2.4 Ordenação das sub-portadoras

A ordenação das sub-portadoras refere-se a ordem em que os *bits* dos blocos FEC serão alocados no símbolo DMT. No simulador, este parâmetro pode ser escolhido ao começo da simulação entre 2 opções: sequencial ou aleatória. No esquema sequencial, a portadora que ocupa a faixa de frequência mais baixa é a primeira e as próximas faixas seguem a ordem crescente. No esquema aleatório, a cada simulação um vetor aleatório é gerado e cada posição k indica qual sub-portadora será carregada na iteração k da modulação.

3.3 Codificador e Decodificador Reed-Solomon

A implementação do codificador e decodificador Reed-Solomon foi feita usando respectivamente as funções *rsenc* e *rsdec* do CT. As funções recebem como parâmetro a mensagem a ser codificada, ou decodificada, e os parâmetros N e K definidos na equação 2.11. A mensagem que é passada como parâmetro precisa estar definida no corpo finito de *Galois*, e para isso a função *gf* do CT é usada. Essa última, recebe como parâmetro a mensagem em formato de *string* e um inteiro identificando o corpo finito de *Galois*.

3.4 Interleaving e Deinterleaving

Para implementar o *interleaving* e o *deinterleaving* definido em [13, p.50] e comentado na seção 2.6, foram usadas respectivamente as funções *intrlv* e *deintrlv* do CT. As funções recebem como parâmetro um *array* com a estrutura a ser embaralhada e um vetor especificando a ordem final dos dados. Dessa forma, foi preciso implementar, separadamente, um algoritmo que retornasse como saída o vetor de posições após o procedimento de *interleaving*. Os passos que o algoritmo faz estão no quadro abaixo:

1. Gera um vetor α com inteiros a partir de 1 até o número de *bytes* a ser transmitido.
2. Verifica se o parâmetro N , da equação 2.11, é ímpar.
 - (a) Se não, a cada N inteiros adiciona ao vetor α um inteiro com valor -1.
3. Gera um outro vetor β vazio com o mesmo tamanho de α .
4. Para cada inteiro na posição k do vetor α faz:
 - (a) Coloca o inteiro na posição $(D - 1) * (k \text{ modulo } N)$ do vetor β .
5. Ao final, remove de β todos os inteiros com valor -1 e dá como saída β .

É preciso adicionar *bytes* extras ao vetor de posições, porque como cada *byte* é atrasado de $(D - 1) * i$ posições, sendo o i o índice do *byte* ($0, 1, \dots, N - 1$), N e D precisam ser primos entre si. Caso contrário, em algum momento as novas posições de dois *bytes* distintos coincidiram e a informação seria perdida no *interleaving*. O algoritmo implementado que realiza essa operação é apresentado no apêndice A.3.

3.5 Mapeamento na constelação QAM e interpretação QAM

As funções do CT, *modem.genqammod* e *modem.genqamdmod*, criam objetos dos moduladores e demoduladores para uma determinada constelação QAM, transportando um número fixo de *bits*. Assim, como o número máximo de *bits* é 15, 30 objetos ao total são criados na inicialização da ferramenta e posteriormente reaproveitados em cada simulação. Os objetos possuem atributos como: número de pontos da constelação e os próprios pontos (X, Y) das constelações; e os métodos *modulate* e *demodulate*.

Como os padrões DSL definem uma regra de mapeamento própria [13, p. 81, seção 8.6.3], isto é, diferente das regras clássicas já oferecidas pelas funções do CT, foi preciso implementar código a parte para gerar as constelações referentes a cada quantidade de *bits* e passá-las como parâmetro das funções. As constelações seguem as seguintes regras:

1. Constelações com número par de *bits*: Os inteiros X e Y do ponto da constelação (X, Y) deverão ser determinados a partir dos b *bits* $\{v_{b-1}, v_{b-2}, \dots, v_1, v_0\}$ como segue. X e Y são inteiros ímpares com representações binárias complemento a 2 $(v_{b-1}, v_{b-3}, \dots, v_1, 1)$ e $(v_{b-2}, v_{b-4}, \dots, v_0, 1)$, respectivamente. Os *bits* mais significantes (MSBs - Most Significant *bits*), v_{b-1} e v_{b-2} , são os *bits* de sinal para X e Y , respectivamente.
2. Apenas 1 *bit*: O valor 0 (zero) é representado pelo ponto (1,1) e o valor 1, pelo ponto (-1,-1).
3. Número de *bits* igual a 3:

A constelação segue na tabela 3.1 abaixo:

Valor	X	Y
0	1	1
1	1	-1
2	-1	1
3	-1	-1
4	-3	1
5	1	3
6	-1	-3
7	3	-1

Tabela 3.1 Constelação 8-QAM dos sistemas DSL.

4. Número ímpar de *bits* maior que 3: Os 2 MSBs de X e Y são determinados pelos 5 MSBs dos b *bits*. Seja $c = (b + 1)/2$, então X e Y possuem as representações binárias complemento a 2 $(X_c, X_{c-1}, v_{b-4}, v_{b-6}, \dots, v_3, v_1, 1)$ e $(Y_c, Y_{c-1}, v_{b-5}, v_{b-7}, v_{b-9}, \dots, v_2, v_0, 1)$, onde X_c e Y_c são os *bits* de sinal de X e Y respectivamente. A relação entre $X_c, X_{c-1}, Y_c, Y_{c-1}$ e $v_{b-1}, v_{b-2}, \dots, v_{b-5}$ é mostrada na tabela 3.2.

No apêndice A.4 encontra-se o código responsável pela geração da constelação QAM, seguindo as regras acima.

3.6 Modulação e Demodulação

Para realizar a modulação e a demodulação do símbolo DMT foram usadas, respectivamente, as funções *ifft* e *fft* do CT. As funções recebem um vetor especificando os símbolos QAM ou os valores da amostragem do símbolo DMT e um segundo parâmetro especificando em quantos pontos será calculada a transformada.

Na modulação, para obter valores reais do símbolo DMT a partir dos valores complexos dos símbolos QAM é necessário que esses valores complexos estejam organizados simetricamente num vetor, assim as propriedades da transformada discreta de fourier garantem que a saída será formada apenas por valores reais. Esta simetria, conhecida como simetria hermitiana, pode ser obtida seguindo a formula abaixo para T pontos QAM, cada um representado por $Z_k = X_k + jY_k$:

$$Z_H = \begin{cases} \Re[Z_k] & \text{para } k = T, \\ Z_k & \text{para } k < T, \\ \Im[Z_k] & \text{para } k = T, \\ Z_{k'}^* & \text{para } k' = T - k. \end{cases} \quad (3.1)$$

O vetor Z_H gerado possui $2T$ pontos como comentado na seção 2.3 e sua transformada inversa, $ifft(Z_H)$, tem como saída apenas valores reais. Na demodulação, após a aplicação da função *fft*, a primeira metade já representa os valores das sub-portadoras recuperados.

$v_{b-1}, v_{b-2}, \dots, v_{b-5}$	X_c, X_{c-1}	Y_c, Y_{c-1}
0 0 0 0 0	0 0	0 0
0 0 0 0 1	0 0	0 0
0 0 0 1 0	0 0	0 0
0 0 0 1 1	0 0	0 0
0 0 1 0 0	0 0	1 1
0 0 1 0 1	0 0	1 1
0 0 1 1 0	0 0	1 1
0 0 1 1 1	0 0	1 1
0 1 0 0 0	1 1	0 0
0 1 0 0 1	1 1	0 0
0 1 0 1 0	1 1	0 0
0 1 0 1 1	1 1	0 0
0 1 1 0 0	1 1	1 1
0 1 1 0 1	1 1	1 1
0 1 1 1 0	1 1	1 1
0 1 1 1 1	1 1	1 1
1 0 0 0 0	0 1	0 0
1 0 0 0 1	0 1	0 0
1 0 0 1 0	1 0	0 0
1 0 0 1 1	1 0	0 0
1 0 1 0 0	0 0	0 1
1 0 1 0 1	0 0	1 0
1 0 1 1 0	0 0	0 1
1 0 1 1 1	0 0	1 0
1 1 0 0 0	1 1	0 1
1 1 0 0 1	1 1	1 0
1 1 0 1 0	1 1	0 1
1 1 0 1 1	1 1	1 0
1 1 1 0 0	0 1	1 1
1 1 1 0 1	0 1	1 1
1 1 1 1 0	1 0	1 1
1 1 1 1 1	1 0	1 1

Tabela 3.2 Relação entre os 2 MSBs de X e Y com o vetor de b *bits*.

3.7 Meio Físico

Nesta etapa apenas os ruídos são levados em conta, já que a atenuação foi inserida no recálculo do SNR, substituindo a possível etapa da convolução do sinal transmitido com a resposta ao impulso unitário do meio físico. Os ruídos térmico e impulsivo, são aplicados separadamente e detalhados a seguir.

3.7.1 Ruído Gaussiano Branco Aditivo

Para aplicar o ruído térmico (AWGN) ao sinal é preciso adicionar a cada instante de tempo um valor sorteado de uma variável aleatória que se distribui como uma normal. Porém, é preciso calcular corretamente a variância dessa normal para que o efeito nas portadoras tenha uma variância igual a configurada como entrada do simulador, pois o somatório da transformada discreta de fourier altera as propriedades da variável aleatória aplicada no tempo.

Para isso, suponha que X_k represente o valor demodulado da portadora k , que $x[n]$ represente o valor do sinal DMT e $r[n]$, o valor sorteado para o ruído, ambos no instante de tempo discreto n . No momento da demodulação através da tranformada discreta de fourier, temos:

$$X_k = \sum_{n=0}^{T-1} (x[n] + r[n]) e^{\frac{-j2\pi nk}{T}} \quad (3.2)$$

Como a transformada de fourier é linear, a transformada da soma é a soma das transformadas e podemos reescrever a equação 3.2 assim:

$$X_k = \sum_{n=0}^{T-1} x[n] e^{\frac{-j2\pi nk}{T}} + \sum_{n=0}^{T-1} r[n] e^{\frac{-j2\pi nk}{T}} \quad (3.3)$$

$$= F\{x\} + F\{r\} \quad (3.4)$$

Se definirmos $R_k = F\{r\}$ como o efeito do ruído na portadora k e expandirmos a exponencial em cossenos e senos, podemos calcular a média e a variância do efeito em qualquer uma das componentes real ou imaginária¹. Note que as variáveis aleatórias associadas a cada aplicação do ruído nos instantes de tempo n são independentes e possuem a mesma distribuição de probabilidade, logo a média do efeito do ruído será igual à média individual das variáveis e sua variância será a soma das variâncias individuais.

Como a variância amostral de uma senóide é igual a $\frac{1}{2}$, cada termo do somatório é uma variável aleatória que se distribui como uma normal com média μ e variância $\frac{\sigma^2}{2}$, logo a média e variância do efeito na componente real é:

$$MEDIA(\Re(R_k)) = \mu \quad (3.5)$$

$$VARIANCIA(\Re(R_k)) = \sum_{n=0}^{T-1} \frac{\sigma^2}{2} = \frac{T\sigma^2}{2} \quad (3.6)$$

Assim, finalmente, se o objetivo é afetar as portadoras com um ruído de variância σ^2 é preciso aplicar no tempo um ruído com variância igual a $\frac{2\sigma^2}{T}$.

¹Como a expansão da exponencial não altera a natureza da variável aleatória, a variância e média dos efeitos da componentes real e imaginária serão iguais.

Para gerar o ruído foi usada a função *randn* que sorteia valores que se distribuem como uma normal com média zero e variância 1 e multiplicou-se cada valor pelo o desvio padrão da nova distribuição. O código implementado na ferramenta executa algo similar a equação abaixo:

$$x'[n] = x[n] + \sqrt{\frac{2\sigma^2}{T}} * randn() \quad (3.7)$$

3.7.2 Ruído Impulsivo

Para a injeção do ruído impulsivo são escolhidos 2 parâmetros no começo da simulação: comprimento da rajada ou *burst* e sua taxa de repetição ou frequência; como explicado na seção 2.5. Neste trabalho, a forma de onda do ruído impulsivo é fixa, modelo igual ao distribuído pela ITU-T [20], mas a posição em que sua ocorrência se inicia é aleatória por símbolo DMT, devido a sua natureza aleatória. Por simplificação, as rajadas se limitam a no máximo 10 pulsos seguidos, o que representa o tamanho do ruído REIN, considerando a forma de onda do pulso pré-definida, que pode coincidir inteiramente com o símbolo DMT.

Para a criação de rajadas, simplesmente os pulsos são concatenados em vetores no começo da simulação; já para a implementação da ocorrência do ruído com uma média de repetições fixa por segundo, nomeada como f , e com posição inicial ϕ , o seguinte algoritmo é executado:

1. Sorteia uniformemente um número aleatório x entre 0 e 4000.
2. Compara x com f .
 - (a) Se $x > f$, o ruído não será adicionado.
 - (b) se $x \leq f$, o ruído será adicionado.
3. Se é para adicionar o ruído, sorteia aleatoriamente uma posição inicial ϕ .
4. Soma ruído e sinal.

Como o número de simbolos DMT transmitidos por segundo é igual a 4000, o algoritmo acima adiciona o ruído com uma probabilidade $\frac{f}{4000}$, mantendo a média de ocorrências por segundo próxima de f .

Como citado, para cada injeção do ruído impulsivo no símbolo DMT, é sorteado uma fase aleatória ϕ que indica em que amostra discreta do tempo a injeção se iniciará. Assim, se o comprimento do pulso é t e T é o número de amostras do símbolo DMT, então o intervalo do sorteio para ϕ é:

$$-t + 1 \leq \phi \leq T - 1 \quad (3.8)$$

3.8 Contagem de erros

Durante toda simulação foram sendo armazenados, em vetores distintos, os dados de antes e depois de cada etapa da transmissão, para ao final analisá-los nesta seção. Para contar os erros entre os vetores dos dados transmitidos e recebidos, foram utilizadas as funções *symerr* e *biterr* do CT.

A função *symerr* compara dois vetores de mesmo tamanho e conta em quantas posições os dois vetores estão diferentes. Além do número de posições diferentes, ou erros, dá como saída a razão do número de erros pelo tamanho total do vetor. Essa última saída pode ser interpretada como a frequência relativa dos erros e, quando muitos dados são simulados, aproximada para a probabilidade de erro. A segunda função, *biterr*, faz algo similar à primeira, mas ao invés de comparar as posições, compara a representação binária dos valores de cada posição, indicando ao final quantos *bits* estavam diferentes.

As principais contagens feitas no simulador foram: o número de *bytes* errados em cada bloco FEC antes e depois da decodificação Reed-Solomon, o número total de *bytes* e *bits* errados na mensagem recebida e o número de *bytes* errados localizados nos cabeçalhos das células da camada de enlace, considerando o protocolo ATM (*Asynchronous Transfer Mode*).

Com as primeiras contagens, constrói-se a distribuição dos erros nos blocos FEC, antes e depois da decodificação, e a distribuição binomial a partir da probabilidade do erro de *byte* (segundo parâmetro de saída da função *symerr* do CT). Esses resultados são posteriormente passados à interface, ver seção 3.9.1.

A contagem dos erros nos cabeçalhos das células ATM é feita dividindo a mensagem original e recebida (após os procedimentos de decodificação) em blocos, ou células, de 53 *bytes* e comparando os 5 primeiros *bytes* de cada bloco. Essa contagem representa aproximadamente o contador HEC (*Header Error Check*) da etapa de checagem na camada de enlace.

3.9 Interface da ferramenta

Para melhorar a usabilidade da ferramenta, uma interface gráfica foi desenvolvida e está representada na figura 3.4.

Na interface principal, estão presentes todos os parâmetros de configuração da simulação e esses estão organizados em 4 blocos: *Simulation Choices*, *Features of Subscriber Line*, *Protection* e *Transient Noise*.

No primeiro bloco, estão os parâmetros mais gerais como: a quantidade de *bytes* a ser transmitida e o método de ordenação das sub-portadoras QAM. É possível escolher entre 3 unidades de quantidade de *bytes*: *Kbyte* (1024 *bytes*), *Mbyte* (1024 *Kbytes*) e *Gbyte* (1024 *Mbytes*). Quanto maior a quantidade de *bytes* transmitidos, maior a generalidade dos resultados, mas também maior o tempo de execução da simulação. O método de ordenação das sub-portadoras pode ser configurado para aleatório (*Random*) ou sequencial (*Sequential*), como explicado na seção 3.2.4.

O segundo bloco é alimentado com as características da linha em análise, distância do cliente até a central do provedor e nível do ruído térmico de fundo. Existem 3 opções para distância entre o cliente e o provedor: 1000, 2000 e 3000 metros. Quanto maior a distância, maior

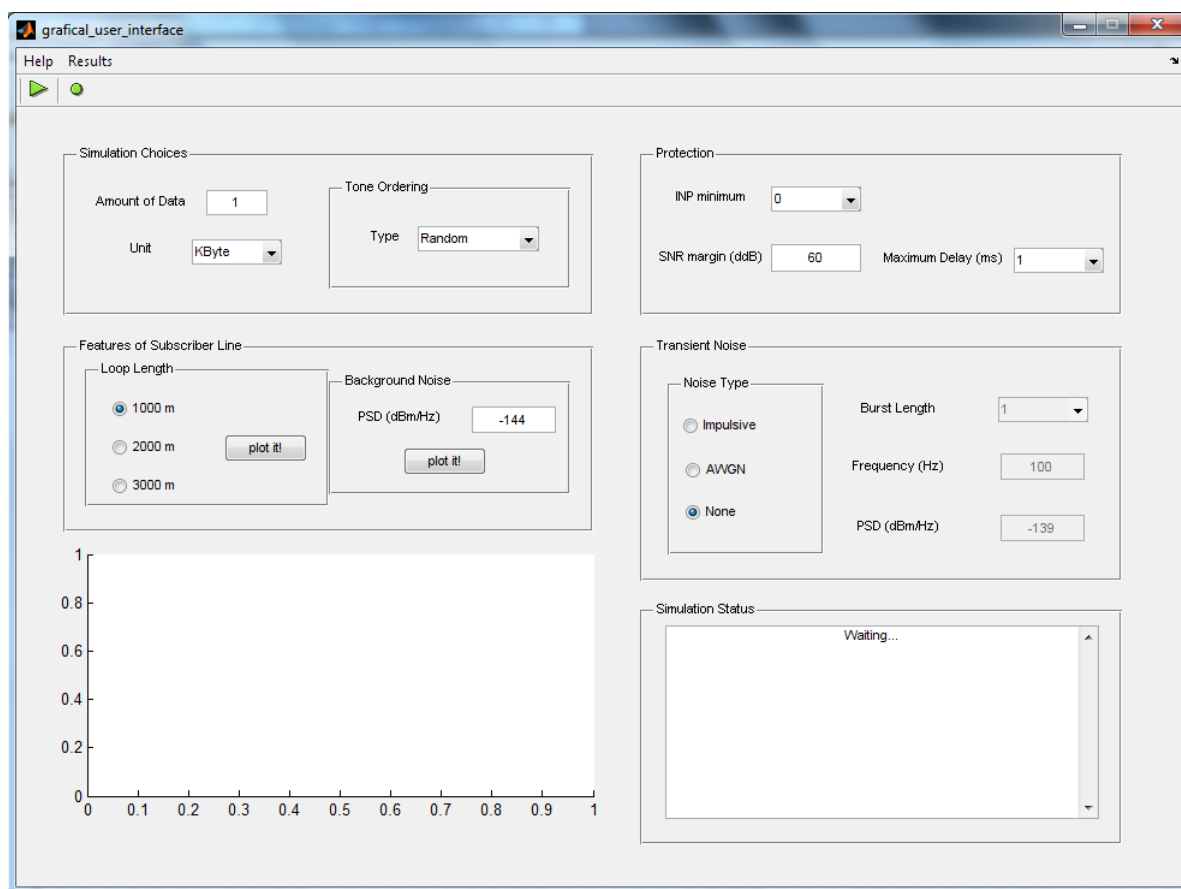


Figura 3.4 Interface principal da ferramenta de simulação.

a atenuação sofrida pelo sinal transmitido. O nível do ruído térmico de fundo fica a critério do usuário da ferramenta, mas este é iniciado com o valor padrão de -144 dBm/Hz. Caso o usuário tente colocar um ruído de fundo muito forte, acima de -80 dBm/Hz, uma mensagem de aviso será apresentada ou um erro será gerado, caso esse valor seja positivo. Este nível representa a variância do efeito do ruído diretamente aplicado às sub-portadoras. Para as duas características da linha, é possível visualizar o comportamento do parâmetro clicando no respectivo botão "*plot it!*".

O terceiro bloco de configurações, apresenta os parâmetros de proteção previamente citados: INP_{min} , $Delay_{max}$ e SNR_{margin} . Alguns valores pré-definidos ficam a disposição para a escolha do usuário. No quarto bloco, as configurações do ruído que é injetado após a sincronização da linha estão disponíveis, exatamente como explicado nas seções 2.5 e 3.7. As restrições do nível do ruído térmico de fundo, também se aplicam ao nível de ruído transiente. Um resumo dos valores de todos os parâmetros pode ser visto na tabela 3.3.

Depois de escolhidos as configurações, é possível simular o cenário clicando em um dos botões verdes de execução logo abaixo do menu. O primeiro executa a simulação completa e o segundo apenas os procedimentos de sincronização da linha. O status da simulação vai sendo impresso no bloco de texto do canto inferior direito da interface a medida que a simulação

Parâmetro	Valores possíveis
<i>Amount of data</i>	Qualquer número positivo
<i>Unit</i>	<i>Kbyte</i> , <i>Mbyte</i> ou <i>Gbyte</i>
<i>Tone Ordering: Type</i>	<i>Random</i> ou <i>Sequential</i>
<i>Loop Length</i>	1000, 2000 ou 3000
<i>Background Noise: PSD</i>	Qualquer valor negativo
<i>INP minimum</i>	0, 0.5, 1,2 e 4
<i>SNR margin</i>	Qualquer valor maior que zero
<i>Maximum Delay</i>	1, 8, 16 e 32
<i>Impulsive: Burst Length</i>	1, 4, 8 e 10
<i>Impulsive: Frequency</i>	Qualquer número positivo
<i>AWGN: PSD</i>	Qualquer valor negativo

Tabela 3.3 Resumo dos parâmetros configuráveis da interface principal.

ocorre. Depois da simulação ter sido finalizada, é necessário clicar no menu "**Results**" e uma outra interface será inicializada. Os detalhes dessa interface seguem na próxima seção.

3.9.1 Interface dos resultados

Nessa interface, 4 tipos de resultados podem ser visualizados: Alocação de *bits*, Constelação QAM, Distribuição dos erros e Contadores. Os gráficos da constelação QAM e o de distribuição dos erros permitem interação com o usuário. No gráfico da constelação QAM, uma das portadoras com o número de *bits* maior que zero é escolhida e os possíveis pontos teóricos em vermelho (**QAM POINT**), os pontos recebidos em azul (**RX POINT**) e os pontos enviados em verde (**TX POINT**). No painel da distribuição dos erros, é possível escolher entre duas opções de gráfico: a distribuição dos erros no bloco FEC antes da decodificação (opção *FEC*) e depois da decodificação do código *Reed-Solomon* (opção *Decoded FEC*). Uma amostra da interface de resultados é apresentada na figura 3.5.

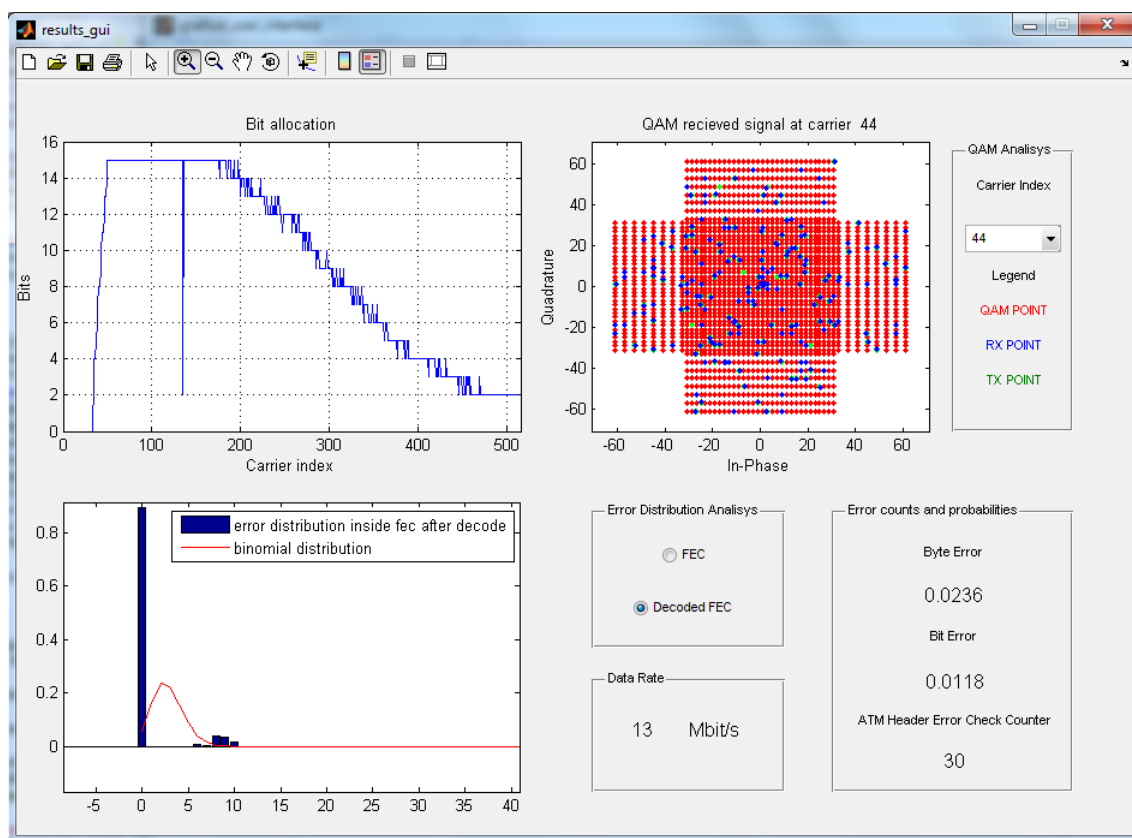


Figura 3.5 Interface de resultados da ferramenta de simulação.

Avaliação do Simulador

4.1 Validação

Neste trabalho, como a ferramenta desenvolvida usa a simulação como técnica de avaliação da performance do sistema DSL, para confiar nos resultados obtidos é necessário a validação dos mesmos por modelos analíticos ou medições de uma plataforma real [23, p. 32]. Já que, mesmo com uma plataforma real, nem todas as informações estão disponíveis para validação, daí a motivação em desenvolver o simulador, a validação será feita em alguns pontos críticos.

A validação de uma simulação consiste em validar três aspectos chave: as premissas utilizadas, as entradas do sistema e suas distribuições e as saídas e conclusões [23, p. 420].

Dentre as premissas identificadas, uma importante a citar é a de que é possível, a partir dos dados referentes a atenuação e configurações do sistema, saber no momento da sincronização a potência aplicada ao sinal transmitido. Os padrões DSL, são documentos de recomendação e não de regulamentação, pois em muitos pontos ou deixam vago ou simplesmente assumem que certos parâmetros e técnicas, quais seus valores ou como implementá-las, são livres para a escolha do fabricante. Técnicas de modelagem do espectro (*spectrum shaping*) a partir de filtros digitais para diminuir efeitos de *crosstalk* ou pontos de controle da máscara de potência não são definidos pela ITU-T. Logo, qualquer que sejam as escolhas feitas no projeto de um simulador, a premissa citada acima não pode ser 100% garantida sem que se perca em generalidade.

As entradas da ferramenta se limitam a definições dos padrões DSL [13, 18] e não há necessidade de validar suas distribuições, já que na maioria são de escolha do operador e sem qualquer comportamento probabilístico. A validação de dados em si, então, será realizada com o resultado da etapa de sincronização, que é a alocação de *bits* nas sub-portadoras.

Para a coleta de dados para comparação, foi utilizada uma plataforma real instalada num laboratório de redes dentro da UFPE, esta está representada na figura 4.1.

A plataforma pode ser separada em quatro ambientes: O de gestão dos experimentos (mais externo), o do cliente, o da linha DSL e o da central telefônica. O ambiente de gestão possui os elementos básicos de uma rede de gestão: um *switch* e uma estação de gestão dos experimentos. A estação de gestão controla o resto dos ambientes, pois nela estão os *scripts* que executam as rotinas dos experimentos e que fazem a comunicação com os instrumentos e equipamentos da plataforma. Todos os *scripts* estão escrito para o *software* do MATLAB.

No ambiente do cliente, estão a estação do cliente (PC comum), que vai receber os dados transmitidos, e o modem *SpeedTouch* da Thomson [27]. O ambiente da linha DSL conta com o gerador de funções e ondas arbitrárias AWG33220A [28], o emulador de linha DLS 410E [29], o injetor de ruídos DLS 5405 e o gerador de ruídos DLS 5500 [30]. Os ruídos impulsivos são gerados pelo AWG33220A e o ruído térmico gerado pelo DLS 5500. O emulador de

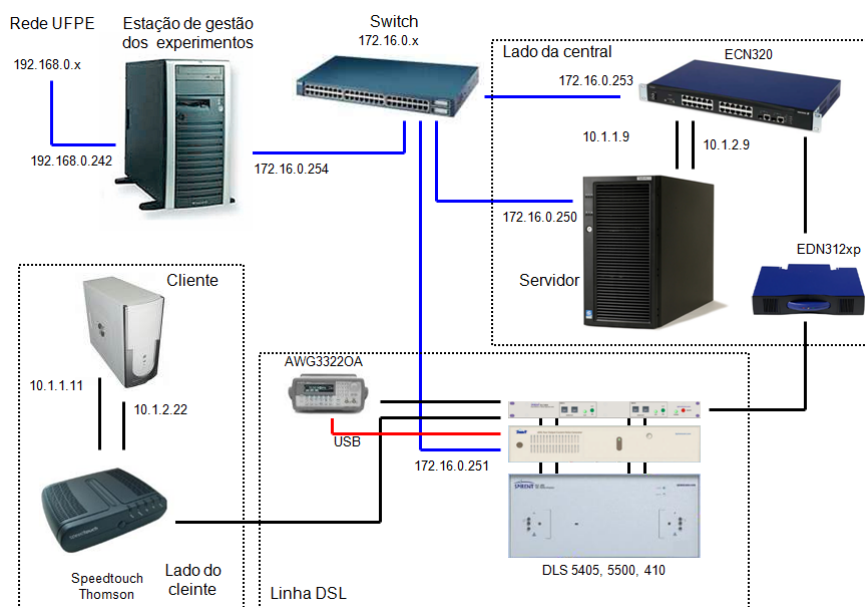


Figura 4.1 Diagrama da plataforma de testes.

linha DLS 410E trabalha com dispositivos eletromecânicos, como relés¹, para emular extensas conexões de fios de par trançado. Seu mecanismo consegue emular uma extensão de até 7 quilômetros em passos de 25 metros. O injetor de ruídos é o equipamento responsável pela injeção aditiva dos ruídos nos sinais transmitidos. Por último, no ambiente da central telefônica estão o DSLAM (EDN312 - *Ethernet DSL Access*), um swith (ECN320 - *Ethernet Controller Node*) e um servidor, onde estão arquivos que serão transmitidos ao cliente. O EDN312 e o ECN320 fazem parte do conjunto *EDA Access Domain* da Ericsson [31].

Utilizando a plataforma real descrita acima, foram realizados experimentos de sincronização da linha e coletados dados referentes a 5 cenários distintos e repetidos 5 vezes. Como o algoritmo para alocação de *bits* é realizado para cada uma das 512 sub-portadoras, para cada cenário foram feitas 2560 (5x512) amostras do algoritmo. Os cenários testados estão listados na tabela 4.1. Em todos os cenários foi usado apenas o ruído térmico da linha a um nível de -144dBm/Hz .

Cenário	Extensão da linha	SNR_{margin}	INP_{min}
1	1km	60 ddB	0.5
2	1km	120 ddB	1
3	2km	90 ddB	2
4	3km	60 ddB	0
5	3km	120 ddB	0.5

Tabela 4.1 Cinco cenários distintos para validação do algoritmo de alocação.

¹Tipo de interruptor acionado eletricamente.

Para verificar se os resultados do simulador estão próximos o suficiente da plataforma real será usado o método descrito em [23, p. 208]. Esse método de validação, utiliza o teste da média zero para observações pareadas que, em poucas palavras, consiste em verificar se o erro entre as observações dos sistemas se distribui como uma normal centrada próxima do zero. Se isso ocorrer, os sistemas podem ser ditos como semelhantes.

O método parte de um vetor contendo o erro das observações e calcula um intervalo de confiança onde provavelmente a média real do erro estará. O cálculo desse intervalo exige um nível de confiança de $(100 - \alpha)\%$, de forma que, se 100 observações forem feitas, em $(100 - \alpha)$ delas a média do erro estará contida no intervalo. O intervalo pode ser calculado através da fórmula abaixo [23, p. 208]:

$$\bar{X} \pm Z_{(1-\frac{\alpha}{2})} \frac{S}{\sqrt{N}} \quad (4.1)$$

Onde $\bar{X}$ é média amostral, S é o desvio padrão das amostras, N é quantidade de amostras e $Z_{(1-\frac{\alpha}{2})}$ é o $(1 - \frac{\alpha}{2})$ -quantil de uma distribuição normal unitária. Nessa validação será usado o nível de confiabilidade de 95% e o vetor de erro será calculado pela subtração entre a alocação de *bits* simulada e a real, nessa ordem. Assim, se o erro for positivo, o simulador sobrestimou; e se o erro for negativo, o simulador subestimou.

Vale lembrar que caso o erro não se comporte como uma distribuição normal de alguma forma, mesmo com média bem diferente de zero, não faz sentido algum validar os resultados por esse método. Isso ocorreu, por motivos específicos, em 2 tipos de cenário avaliados, que serão discutidos posteriormente.

No cenário 3 foi observado um comportamento semelhante a uma distribuição normal, como visto na figura 4.2, e o resultado do teste da média zero foi positivo como mostrado na tabela 4.2.

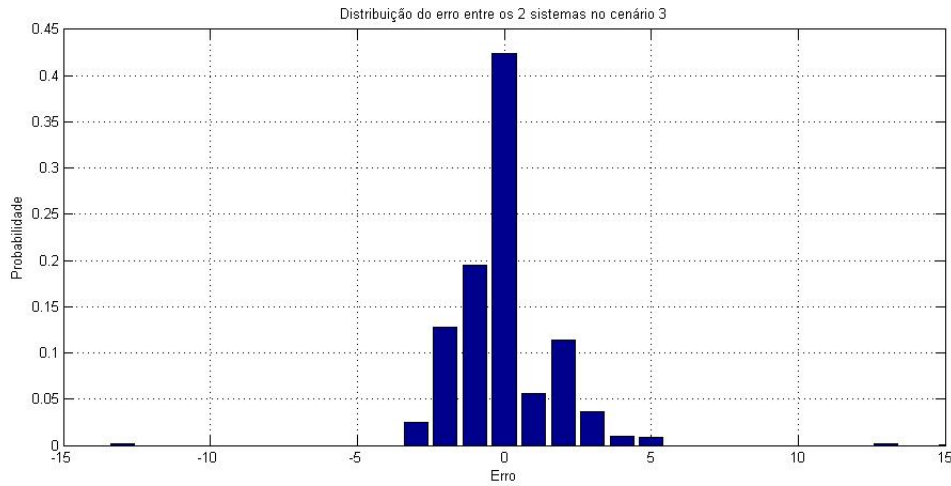


Figura 4.2 Histograma do erro no cenário 3.

Nos outros cenários o simulador estima corretamente parte das baixas frequências, já que a atenuação inserida pelo canal é normalmente menor e existe um limite máximo de *bits*, mas,

Parâmetro	Valor
Média	-0.0539
Desvio padrão	1.6721
Intervalo de Confiança	{-0.1187 0.0109}
Resultado	O zero pertence ao intervalo

Tabela 4.2 Dados do teste da média zero para o cenário 3.

nos restantes das sub-portadoras, ou ele sobrestima a quantidade de *bits*, que é o caso dos cenários 1 e 2, ou ele subestima, que é o caso dos cenários 4 e 5.

Nos cenários 1 e 2, devido à extensão da linha ser menor do que o restante dos cenários, a atenuação é menor e consequentemente a potência recebida no lado do cliente é maior. Isso, de acordo com a equação 2.10, significaria mais *bits* alocados e foi o que o algoritmo do simulador estimou, no entanto, provavelmente devido a alguma restrição de potência, filtro aplicado ou até alguma heurística específica determinada pelos fabricantes dos equipamentos DSL utilizados na plataforma real testada, os resultados foram mais comedidos e menos *bits* foram alocados ao longo do espectro. A figura 4.3 mostra a alocação de bits simulada e da plataforma real no cenário 2.

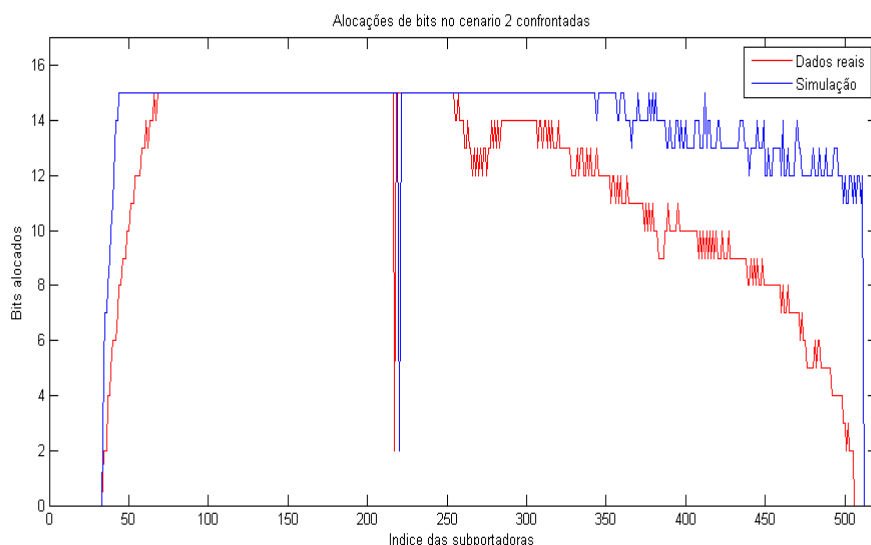


Figura 4.3 Alocação de bits no cenário 2.

Já nos cenários 4 e 5, acontece justamente o contrário, a extensão da linha é maior e devido à atenuação, teoricamente, menos bits seriam alocados, mas não é o que acontece. Neste cenário, por possível escolha dos fabricantes dos DSLAM e do modem, níveis baixos de taxa de *bits* devem ativar técnicas extras de codificação, como o código convolucional de *trellis*, ou alguns limites de potência podem ter sido extrapolados, o que levou a uma subestimativa por parte do simulador. A figura 4.4 mostra a alocação de bits simulada e da plataforma real no cenário 4.

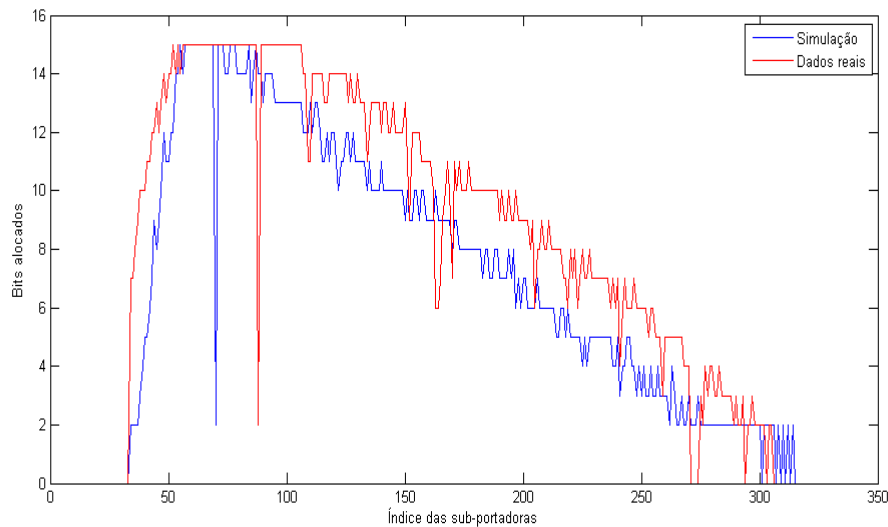


Figura 4.4 Alocação de bits no cenário 4.

4.2 Análise Qualitativa

Para comparar a contribuição de simuladores já desenvolvidos com o deste trabalho, alguns critérios foram elencados e uma análise qualitativa foi realizada. O resultado desta análise se encontra na Tabela 4.3. Os critérios englobam os seguintes tópicos: tecnologias DSL que foram incluídas no projeto do simulador; grau de controlabilidade do simulador e métricas do sistema que o simulador apresenta ao final da simulação. O grau de controlabilidade refere-se à gama de parâmetros de entrada disponíveis para gerar um número razoável de cenários distintos. Depois de avaliados os critérios, notas serão atribuídas (variando entre ruim, regular ou bom), aos seguintes pontos:

Praticidade. Reflete o quão prática é a escolha dos parâmetros de entrada e visualização dos resultados da simulação.

Validade. Reflete o quão válidos ou representativos são os cenários abordados pelo simulador.

Desempenho. Mede o quão rápida é sua execução, ou seja, quanto tempo é levado para uma simulação ser finalizada.

Escopo. Reflete qual a abrangência da simulação, se todas as etapas de uma transmissão DSL foram incluídas no simulador.

Três simuladores serão comparados: o deste trabalho atual, o trabalho em [24] e o simulador disponibilizado pelo software MATLAB no pacote de funções de comunicação [25, 26]. Nas tabela 4.3 são referenciados como SimADSL2+, AnalisadorDeLinha e SimMATLAB, respectivamente.

Métricas	SimADSL2+	AnalizadorDeLinha	SimMATLAB
Padrões	ADSL2+	SHDSL, ADSL{2+}, VDSL	ADSL
Parâmetros de entrada			
Linha	Extensão	Extensão, Tipo e Estrutura	Canal arbitrário
Ruído	AWGN e impulsivo	<i>Crosstalk</i>	AWGN
Proteção	INP_{min} , SNR_{margin} , $Delay_{max}$	SNR_{margin}	N, R e D
Outros	Ordenação das QAM	Tipo da transmissão duplex	—
Saídas do simulador			
PSD	<i>Down</i> e Ruído AWGN	<i>Up</i> , <i>Down</i> e Ruídos	—
Sincronização	Alocação de <i>bits</i> e ganho	—	—
Taxa de <i>bits</i>	Sim	Sim	Não
Erro	<i>BER</i> e <i>FER</i>	—	<i>BER</i> e <i>FER</i>
Distribuição de probabilidade	<i>bytes</i> errados no bloco FEC	—	—
Camada de enlace	Contador de HEC ATM	—	—
Notas dos simuladores / Ruim: 1, Regular: 2 e Bom: 3			
Praticidade	3: Interface Gráfica	3: Interface Web	1: Simulink
Validade	3: Realista	3: Realista	1: Irrealista
Desempenho	2: Depende do cenário	3: Rápido	3: Rápido
Escopo	3: Completo	1: Avaliação da linha	3: Completo

Tabela 4.3 Comparação dos simuladores.

Nos três tópicos da análise, é possível observar pontos onde o simulador deste trabalho (SimADSL2+) se sobressai dos demais. Na praticidade, possui uma interface gráfica simples para manipulação dos parâmetros de entrada. Já o SimMATLAB, é preciso alterar cada bloco do modelo apresentado no simulink, muitas vezes sem explicação clara; por isso sua nota foi Ruim. No quesito parâmetros de entrada, é um dos que trabalha com variáveis comuns ao problema de administração das linhas nos ISPs, pois variáveis como INP_{min} , SNR_{margin} e $Delay_{max}$ estão disponíveis na maioria dos DSLAMs. Já o SimMATLAB utiliza parâmetros que interferem diretamente nos procedimentos padronizados, como variar N e D sem qualquer restrição, e ignora a manipulação dos parâmetros do meio físico, por isso foi considerado irrealista na análise qualitativa. No tópico sobre os parâmetros de saída, é o único que apresenta medidas de desempenho à nível de bloco FEC e da camada de enlace. O simulador AnalizadorDeLinha, só apresenta máscaras de potência (PSD) e taxa de *bits*, nada com relação aos erros; e por isso seu escopo foi considerado Ruim.

Assim podemos dizer que o simulador SimADSL2+, dentre os selecionados, é o que apresenta melhor avaliação sobre os critérios escolhidos. Sua pior avaliação (Regular) refere-se

ao desempenho, pois sua simulação demora muito para cenários com um número razoável de dados transmitidos (a partir de 300 *kbytes*), no entanto é rápido quando a opção de execução se limita aos procedimentos de sincronização.

4.3 Resultados e Análise

Nesta seção os resultados das simulações serão apresentados e discutidos. Vale salientar, reforçando a motivação do trabalho, que os resultados apresentados aqui não estão disponíveis nas implementações atuais de plataformas ADSL2+ reais. Aqueles que estão e que são merecedores de análise, já foram abordados na seção de validação 4.1.

Com a devida contagem de erros explicada na seção 3.8, é possível gerar algumas distribuições dos erros em camadas não acessíveis na plataforma real, como erros dentro do bloco FEC, e observar, dependendo dos parâmetros de proteção da linha, quais são os efeitos dessas mudanças. Somente algumas foram implementadas, mas o simulador dá suporte a várias métricas interessantes, ver seção 5.1. Afim de simplificar o entendimento dos resultados, todos os cenários foram simulados numa linha de 2 quilômetros de extensão e 1 *Mbyte* ou 50 *Kbytes* de dados transmitidos, para experimentos com foco na camada de enlace.

4.3.1 Erros antes da decodificação Reed-Solomon

O INP_{min} e o $Delay_{max}$ estão relacionados com os parâmetros de *framing* R e D pelas equações 2.12 e 2.13; quando $INP_{min} = 0$ e $Delay_{max} = 1$, o modem é forçado a escolher um esquema sem redundância e então $R = 0$. Pela tabela 2.1, quando isso acontece, o valor de D é forçado a ser igual a 1; assim, apenas mudando os parâmetros INP_{min} e o $Delay_{max}$ é possível decidir se haverá ou não a etapa de *interleaving* e codificação [13, 18].

Com mais essa opção de manipular os parâmetros de *framing*, foram feitos experimentos para avaliar o impacto do embaralhamento dos dados nos erros dentro dos blocos FEC. Nesses experimentos a ordenação aleatória das portadoras foi desativada para excluir uma possível contribuição desta etapa nos resultados.

Aplicando apenas o ruído AWGN e configurando os níveis de proteção para $INP_{min} = 0$ e $INP_{min} = 1$, foram contabilizadas as distribuições dos erros nos blocos FEC. As figuras 4.5 e 4.6 mostram os resultados obtidos.

Para analisar a natureza probabilística dos erros, cada resultado acima foi confrontado com a distribuição binomial, que assume a independência dos eventos; e, como pode ser visto, apenas os erros no experimento com *interleaving* se encaixam no perfil da binomial. Essa característica do processo de embaralhamento dos dados é comentada entre os profissionais da área de DSL, muito embora não foi encontrada nenhuma referência demonstrando esse efeito.

Uma suposição razoável do motivo pelo qual o experimento sem *interleaving* toma uma forma bem específica, é a que as sub-portadoras tem probabilidade de erro de símbolo diferentes. A equação 2.5 mostra que a distância entre os pontos da constelação e o SNR medido influenciam nessa probabilidade. Além disso, sabe-se que os padrões DSL definem um limite máximo de 15 *bits* e mesmo que o SNR comporte, a um dado BER, mais do que essa quantidade, só serão transmitidos 15. Assim 15 *bits* serão transmitidos a uma potência que garantiria

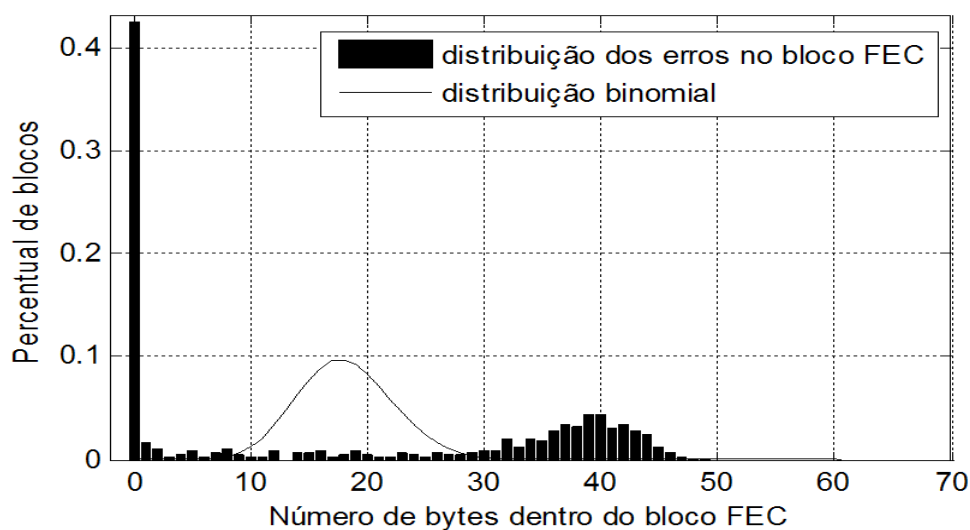


Figura 4.5 Distribuição dos erros num cenário sem interleaving.

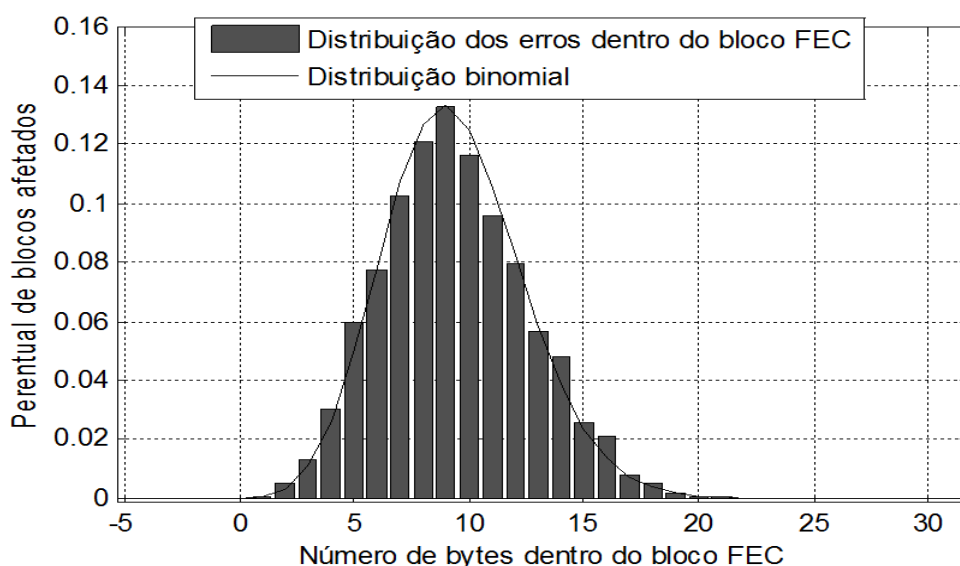


Figura 4.6 Distribuição dos erros num cenário com interleaving.

o mesmo BER para 20 bits, por exemplo, e logo a probabilidade de erro de símbolo especificamente nessas sub-portadoras cai drasticamente. Concluindo, então, que alguns blocos FEC podem estar mais protegidos do que outros.

Para gerar as distribuições binomiais nas duas figuras foram usadas as frequências relativas dos erros de *byte* medidas no experimento como as probabilidades de erro de *byte* p da formula abaixo:

$$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k} \quad (4.2)$$

Onde X é uma variável aleatória representando os erros de *bytes*, k é o número de erros, n é o número total de *bytes* (no nosso caso $n = N$).

4.3.2 Erros após a decodificação Reed-Solomon

Após a decodificação do Reed-Solomon, todos os erros até $\frac{R}{2}$ são corrigidos e os erros a partir de $\frac{R}{2} + 1$ são passados adiante; isso quando esses não coincidem em cair na esfera de aproximação de um código válido no espaço de códigos, ver seção 2.6, pois nesses casos a decodificação é feita errada. Devido ao tamanho do espaço de códigos relacionados aos parâmetros de N e R , a probabilidade de um erro de decodificação acontecer é muito pequena. A figura 4.7 mostra essa distribuição.

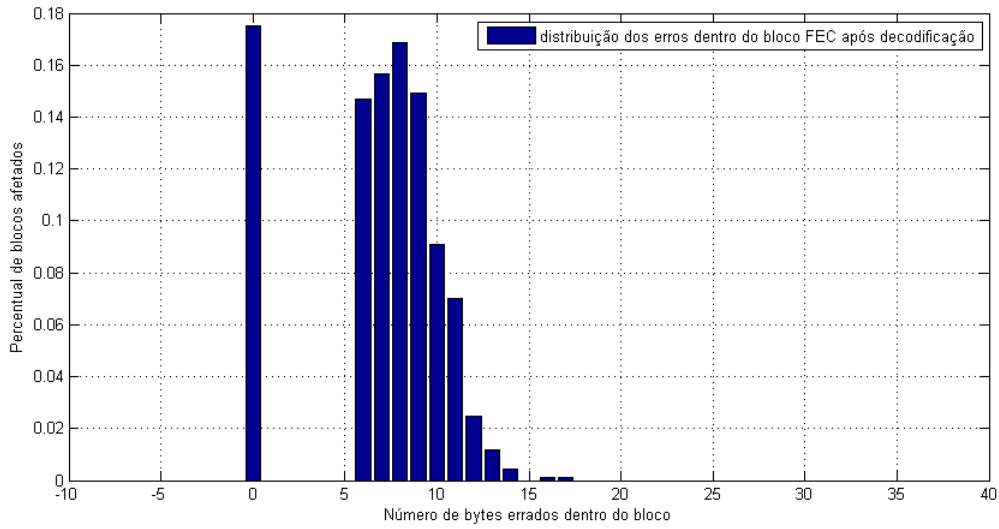


Figura 4.7 Distribuição dos erros após a decodificação do Reed-Solomon.

No cenário da figura 4.7, o parâmetro R é igual a 10, logo todos os erros até 5 *bytes* foram corrigidos e os erros acima disso, permanecem praticamente inalterados. Sendo assim é possível obter o número de blocos de N *bytes* que permaneceram corrompidos após a decodificação, com o seguinte cálculo:

$$FEC_{erro} = FEC_{total} * \sum_{i=\frac{R}{2}+1}^N P(X = i) \quad (4.3)$$

Onde FEC_{total} representa o total de blocos de tamanho N transmitidos e $P(X = i)$ é a probabilidade de i erros de acordo com a equação da binomial 4.2.

4.3.3 Erros na camada de enlace

Os experimentos para observar o comportamento do contador de HEC foram realizados em dois grupos: um com o ruído AWGN a uma potência de -135 dBm/Hz e o outro com o ruído

impulsivo de 1 rajada e 100 repetições por segundos. Para cada um dos grupos, os parâmetros de INP_{min} e SNR_{margin} foram variados de acordo com a tabela abaixo:

Grupo	INP_{min}	SNR_{margin}
AWGN	0.5, 1 e 2	10, 30 e 60
Impulsivo	0.5, 1 e 2	30, 60 e 90

Tabela 4.4 Valores dos parâmetros de proteção para os experimentos de HEC.

Nos experimentos foram transmitidos 50 *Kbytes* de informação, o que nos confere algo em torno de 960 células ATM transmitidas (considerando cada célula ATM com 53 *bytes*). A figura 4.8 mostra os resultados obtidos com os experimentos do primeiro grupo.

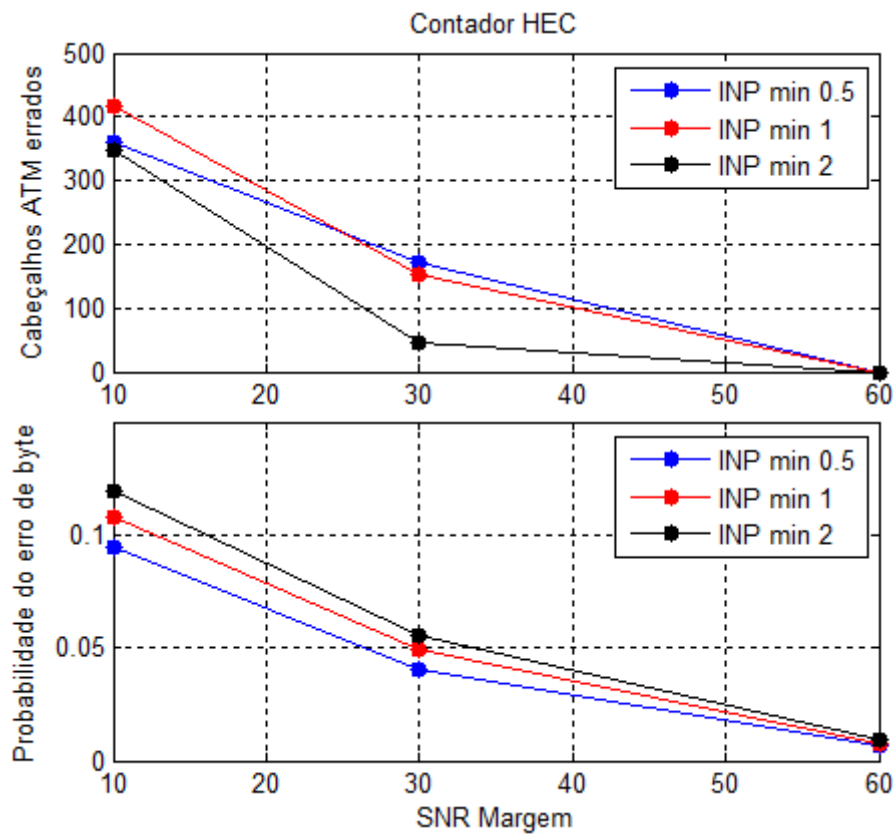


Figura 4.8 Contador de erros do cabeçalho ATM com ruído AWGN.

Os resultados mostram que ao aumentar os parâmetros de proteção a contagem de HEC diminui, apenas na mudança do INP_{min} 0.5 para 1, os valores da contagem de HEC ficaram próximos. Nos três casos de INP_{min} , o parâmetro de SNR_{margin} configurado para 60 dB é suficiente para diminuir a probabilidade de *byte* a um nível de 1%. Isso, juntamente com a codificação, garantiu que praticamente² nenhuma célula fosse repassada com erros. Nos

²Erros ainda podem estar presentes no restante da célula ATM.

cenários com $INP_{min} = 2$, os contadores de HEC somaram cerca de 100 células (10% de todas células transmitidas) a menos que nos cenários com $INP_{min} = 1$. O γ_c , da equação 2.10, aumenta com o aumento do INP_{min} e, conseqüentemente, a probabilidade de erro de símbolo aumenta no nível QAM. No entanto, após a decodificação, um maior INP_{min} faz com que o erro diminua no nível FEC. No segundo gráfico da figura, esse comportamento fica claro, pois as menores probabilidades de erro são dos menores INPs.

No segundo grupo de experimentos, com a mudança do ruído, o comportamento é um pouco diferente. A figura 4.9 apresenta os resultados coletados.

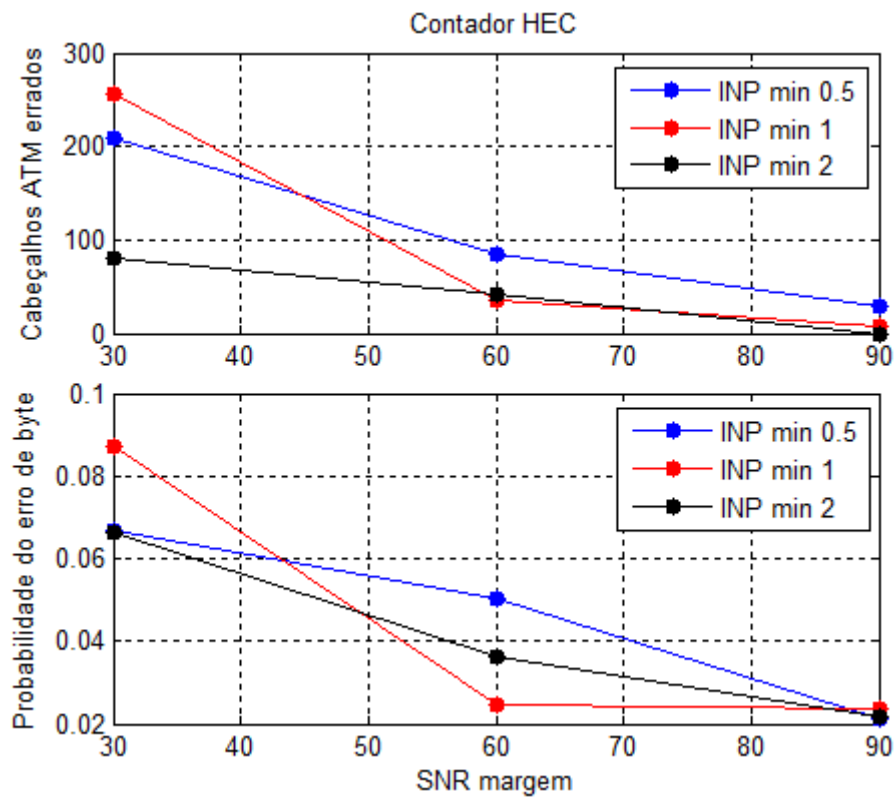


Figura 4.9 Contador de erros do cabeçalho ATM com ruído impulsivo.

A primeira diferença entre os grupos de experimentos é que o parâmetro de SNR_{margin} em 60 ddB não foi suficiente para sanar os erros e a probabilidade de erro de *byte* foi o dobro da observada anteriormente. Somente quando os parâmetros foram configurados para $INP_{min} = 2$ e $SNR_{margin} = 90(ddB)$ foi que o contador de HEC zerou. O segundo gráfico dessa figura, mostra que devido a natureza alatória do ruído impulsivo as probabilidades do erro de *byte* não seguem um comportamento linear com os parâmetros da alocação de *bits* (SNR_{margin} e codificação).

4.4 Cenários de uso da ferramenta

Para poder avaliar a contribuição de uma determinada ferramenta, é preciso apresentar cenários de uso em que a mesma se mostra útil. Para a ferramenta deste trabalho, podemos citar alguns cenários de uso³ em que sua utilidade fica clara. São eles: Predição de taxa de transmissão de dados e Otimização da qualidade pela diminuição da probabilidade de erro. Os dois cenários, ou objetivos, estão interligados com a melhoria da qualidade do serviço de Internet banda larga.

4.4.1 Predição de Taxa

A taxa de transmissão de dados depende de valores que são conhecidos logo após a etapa de sincronização da linha (ver seção 2.7); logo é possível descobrir rapidamente qual será a taxa de transmissão para os novos parâmetros configurados utilizando a segunda opção de execução da ferramenta (*Start Quick Synchronization*). Nessa opção a ferramenta executa apenas os procedimentos de sincronização, como dito na seção 3.9, que duram menos de 1 segundo para serem finalizados. Então, considere a seguinte situação: Um cliente entra em contato com a central de atendimento, questionando sobre uma possível lentidão no acesso à internet. Num caso como esse, um operador pode realizar os seguintes passos:

1. Identificar o cliente;
2. Coletar os parâmetros atualmente configurados para esta determinada linha;
3. Escolher os mesmos parâmetros na interface principal;
4. Simular a ferramenta na segunda opção e verificar a taxa de transmissão alcançada.
 - (a) Se estiver abaixo do contratado, diminuir algum parâmetro de proteção e voltar ao passo 4.
 - (b) Se estiver acima^a ou igual ao contratado, resincronizar a linha com esses parâmetros.
5. Se o problema continuar ou a linha apresentar outros sintomas (queda do sinal DSL), verificar outras fontes do problema.

^aNormalmente existe uma configuração nos DSLAMs que limita a velocidade máxima de *download*, logo se a ferramenta prever algo acima desse patamar, não haverá problema.

Nesse cenário, como as métricas de erro não são verificadas, quer-se cobrir os casos onde a taxa de transmissão está baixa porque a conexão caiu⁴, devido a um ruído forte de curta duração, e foi reconfigurada com parâmetros de proteção altos. Assim quando o ruído deixa de

³Existem outros cenários em que a ferramenta pode ser útil, mas, a primeira vista, os aqui apresentados são os mais importantes.

⁴Isso ocorre quando o modem detecta muitos erros consecutivos e desfaz a sincronização.

estar presente na linha percebe-se que a taxa de transmissão está baixa. Em menos de 1 minuto é possível testar várias possibilidades de taxa e concluir a ocorrência.

4.4.2 Otimização pela diminuição do erro

Nesse cenário, quer-se aumentar a qualidade do serviço diminuindo os erros nos dados recebidos e por isso deve-se analisar algumas configurações de proteção e achar aquela mais adequada para o problema. Como foco da simulação é a qualidade das contagens de erro, é preciso transmitir uma quantidade razoável de *bytes* e assumir algumas características dos ruídos. Agora a ocorrência de um cliente é um pouco diferente e envolve sintomas do tipo: meu modem não sincroniza, minha conexão está caindo repetidamente, entre outros. O algoritmo que o operador iria realizar continua parecido com o anterior, mas envolve outros detalhes:

1. Identificar o cliente;
2. Coletar os parâmetros atualmente configurados para esta determinada linha;
3. Escolher os mesmos parâmetros na interface principal, mas aumentando algum parâmetro de proteção desde o primeiro teste;
4. Escolher transmitir algo em torno de 200 *Kbytes* de informação;
5. Escolher uma configuração para o ruído. Se for a primeira vez, começar com ruídos impulsivos fracos.
6. Simular a ferramenta na primeira opção e verificar a métrica de erro.
 - (a) Se estiver alta, algo igual ou acima de um limiar pré-estabelecido, aumentar algum parâmetro de proteção e voltar ao passo 6.
 - (b) Se estiver baixa, abaixo de desse limiar, resincronizar a linha com os novos parâmetros.
7. Caso o problema persista, volte ao passo 5, escolha um ruído mais forte e continue a bateria de testes.
8. Se o problema for sanado, mas a taxa de transmissão alcançada for baixa, volte ao passo 6 e faça testes diminuindo o parâmetro de SNR_{margin} , adicionando a verificação da uma taxa mínima.

Com uma quantidade razoável de *bytes* sendo transmitidos na primeira opção de simulação, é preciso simular já alguma modificação na linha para evitar perder tempo esperando mais uma simulação. O ruído impulsivo muitas vezes está relacionado à problemas com instalações elétricas, então uma boa tentativa é começar o teste com a frequência do ruído igual ou com algum valor múltiplo da frequência da rede elétrica local⁵. Outros equipamentos de medição

⁵No Brasil a frequência utilizada na rede elétrica é de 60 Hz.

da qualidade de linha podem ajudar escolher quais combinações de ruídos testar.

O limiar pré-estabelecido para a métrica de erro deve ficar a critério do operador e deve depender do tipo de serviço oferecido pelo provedor, entretanto para a maioria das escolhas de limiar o contador de HECs pode ser útil, já que representa os erros na camada ATM, mais próxima da camada IP. É de se esperar que com o tempo seja possível correlacionar esse limiar com os erros em outras camadas da comunicação, melhorando o critério do algoritmo.

Outros resultados da interface podem ajudar na escolha de que parâmetro de proteção é melhor aumentar. Se a distribuição dos erros após a decodificação do *Reed-Solomon*, apresentar uma probabilidade alta de erro para um número de *bytes* maior que o atual $\frac{R}{2}$ (contanto que seja um número próximo), aumentar o parâmetro INP_{min} pode resolver o problema. Se a distância entre os pontos transmitidos e recebidos nas constelações QAM for alta, isso pode indicar, também, que aumentar um pouco o parâmetro de SNR_{margin} pode não resolver e novamente a opção de trabalhar com o INP_{min} deve ser escolhida. Já se a distribuição dos erros nos blocos FEC está concentrada em muitos *bytes* e a distância entre os pontos transmitidos e recebidos nas constelações QAM não for alta, isso acontece quando os pontos recebidos estão centralizados entre dois pontos teóricos, o melhor parâmetro de proteção pode ser justamente o SNR_{margin} .

Conclusão

A família de tecnologias DSL consegue alcançar altas taxas de transmissão de dados, em comparação com outros serviços de acesso à internet banda larga, e está em ascensão na América Latina. Também devido a sua viabilidade de implantação, sua predominância no mercado se mantém e o número de clientes cresce continuamente.

Nos últimos 10 anos, a indústria se mostrou preocupada e atuante na evolução da qualidade dessas tecnologias, mas não foi suficiente para prevenir que as falhas na administração dos ISPs e as poucas informações reportadas sobre a qualidade do serviço pelos equipamentos de rede elevassem o número de reclamações contra as empresas ao nível de 70%, tendo como consequência, processos judiciais e multas da ordem de milhões de reais.

Ferramentas de auxílio à administração dos serviços podem ser vistas como uma das principais necessidades desse enorme mercado que começa a ter sérios problemas financeiros. Dessa forma, este presente trabalho, se mostra perfeitamente alinhado à demanda do mercado de serviços de banda larga.

Ao final do processo de validação, comparação e resultados é possível também afirmar que o objetivo inicial deste trabalho foi alcançado; já que o simulador apresenta saídas fiéis ao sistema ADSL2+ e, talvez, maiores detalhamentos das etapas apenas iriam ocasionar melhorias marginais ou até mesmo perda de generalidade. Outra constatação sobre o simulador desenvolvido é que ele vem ocupar, como ferramenta, um espaço vazio entre os demais simuladores destacando-se na capacidade de reunir uma boa validade de resultados com um escopo bem completo da transmissão de dados. Por fim, ficou comprovada a capacidade de apresentar como saída os comportamentos da camada física e de enlace em função das variáveis de proteção configuradas.

5.1 Versatilidade do simulador

Seguindo uma ordem de importância e afim de deixar simples e clara a interface, apenas alguns resultados foram inteiramente implementados nela. Entretanto, como explicado na seção 3.8, em todas as etapas da simulação as estruturas de dados são armazenadas antes e depois de cada um dos módulos para análise futura. Assim, a quantidade de resultados viáveis com os dados armazenados é muito maior do que aqueles atualmente apresentados na interface.

A variedade dos dados armazenados ao final da simulação é enorme e incluem dados do tipo: mensagem original e recebida, blocos FEC enviados e recebidos, blocos FEC antes e depois da decodificação, *stream* de *bits* antes e depois do *Interleaving*, símbolos DMT enviados e recebidos, símbolos QAM enviados e recebidos, dentre outros. Métricas secundárias de erro e

desempenho também são calculadas, mas não apresentadas, como por exemplo: probabilidade teórica do erro de símbolo QAM em cada sub-portadora no momento da sincronização e aplicação do ruído transiente e a distância efetiva entre os pontos das constelações de cada uma das sub-portadoras.

Então, avaliações que pudessem envolver, por exemplo, a localização temporal dos erros ou o intervalo entre erros também estariam disponíveis com poucos trechos de código extra. Uma das avaliações que merece atenção, mas ao final do trabalho acabou não sendo incluída na ferramenta, é a distribuição dos erros de *bit* a nível DMT. Gerada a partir da contagem dos bits errados em cada símbolo DMT, essa pode mostrar características do ruído que não são triviais, pois é um resultado de um nível ainda mais baixo que o FEC.

5.2 Trabalhos Futuros

Alguns pontos do projeto do simulador precisam ser melhorados, principalmente com relação ao seu desempenho, como ficou claro na comparação com outros simuladores disponíveis. Considerar transcrever o código, ou parte dele, em modelos válidos para o ambiente *simulink* (mantendo a comunicação da interface com os módulos) ou até mesmo mudar a estratégia de simulação para realizar execuções em paralelo, podem ser consideradas como bons pontos de partida. A inclusão de mais métricas de erro apresentadas na interface, como comentado na seção 5.1, pode ajudar o desempenho final do usuário nos cenários de uso explorados na seção 4.4.

Uma interessante alteração a ser avaliada posteriormente, é a possibilidade do usuário escolher exatamente como será feita a ordenação das sub-portadoras e, além disso, definir tipos diferentes de dados a serem transmitidos em sub-portadoras específicas. Isso, visando utilizar algumas sub-portadoras que possuem probabilidade de erro menor do que outras, como brevemente comentado na seção 4.3.1, para dados mais sensíveis à erros como voz ou vídeo. Provedores do serviço de acesso à internet banda larga poderiam avaliar se implementar opções de serviço diferenciado com probabilidades de erro baixíssimas e taxas controladas seria economicamente viável.

Devido a escolha de realizar uma validação confiável e completa, incluindo medições de uma plataforma real do sistema, apenas a tecnologia ADSL2+ foi desenvolvida até o fim. Porém, uma extensão deste trabalho para a tecnologia VDSL2 pode ser trabalhada, mantendo-se praticamente inalterada a estrutura atual do simulador.

A estrutura do simulador apresenta uma forma modular, como visto no capítulo 3, e muitos dos módulos são compartilhados entre as tecnologias ADSL2+ e VDSL2 por utilizarem as mesmas técnicas de comunicação digital para o envio de dados [18, 1]. Dos reajustes necessários para a inclusão no projeto, podemos citar: o algoritmo de *interleaving*, a escolha dos parâmetros de *framing* e a avaliação do canal telefônico (atenuação, PSD, etc).

A tecnologia VDSL2 consegue obter, com a mesma probabilidade de erro, taxas de bits de 200 kbit/s nas duas direções *upstream* e *downstream*. Isso deve-se a banda frequencial disponível ser maior, até 30 MHz, e as direções serem intercaladas [1]. Isso muda um pouco a definição da máscara potência, pequenos detalhes da modulação e também a distância máxima, até onde de fato vale a pena utilizar VDSL2 e não ADSL2+, entre a central e o cliente.

Códigos implementados

A.1 Algoritmo de alocação de bits

```
%% DATA DE CRIAÇÃO: 22/11/2010
%% CRIADOR: Rodrigo Perazzo Rabelo
%% DESCRIÇÃO: Executa o algoritmo de alocação de bits
%% ENTRADA: SNR, margem, codificação e número de portadoras
%% SAÍDA: o número de bits a ser alocado em cada portadora
%%
%first calculation
alloc_bits(1:carriers) = arrayfun(@(i) (log2(1+(SNR(i)...
    *coding_gain)/db2linear(pe_gap+(set_margin/10)))),1:carriers);

%change power for round the number of bits per carrier
gain = (2.^(round(alloc_bits))-1)./(2.^(alloc_bits)-1);

temp_psd = linear2db((1000/Fd).*power.received.*gain);
index = find(temp_psd > MAXPSDMASK);

if(~isempty(index))
    gain(index) = (2.^(floor(alloc_bits(index))-1)./...
        (2.^(alloc_bits(index))-1);
end

SNR = SNR.*gain;

%second calculation
alloc_bits(1:carriers) = int8(arrayfun(@(i) (log2(1+(SNR(i)...
    *coding_gain)/db2linear(pe_gap+(set_margin/10)))),1:carriers));

%limits number of bits
for i=1:carriers
    if(alloc_bits(i) > maxbits)
        alloc_bits(i) = maxbits;
    end
end
end
```

```

%pilot tone
secure = find(alloc_bits == max(alloc_bits));
if(length(secure)> 1)
    middle = round(length(secure)/2);
    pilotindex = secure(middle+ceil((middle/4)*rand));
else
    pilotindex = secure;
end
alloc_bits(pilotindex) = 2;

```

A.2 Procedimento para escolha dos parametros de framing

```

%% DATA DE CRIAÇÃO: 29/03/2010
%% CRIADOR: Rodrigo Perazzo Rabelo
%% DESCRIÇÃO: Executa o algoritmo que escolhe os parâmetros de framing
%% ENTRADA: INPmin, Delaymax e L
%% SAÍDA: R, N e D
%%
function [R_ D_ N_] = calc_frame(max_delay, L, min_inp)

R = [ 0 2 4 6 8 10 12 14 16 ];
D = [ 1 2 4 8 16 32 64 128 256 ];
RDN = [0 1 0];
k = 1;
if(min_inp ~= 0)
    for i=1:length(R)
        for j=1:length(D)
            if( (4*R(i)*D(j)/L) >= min_inp )
                Ns = calc_N(L, R(i), D(j), min_inp, max_delay);
                if(min(Ns) > 0)
                    RDN(k, 1) = R(i);
                    RDN(k, 2) = D(j);
                    N_ = max(Ns);
                    RDN(k, 3) = N_;
                    RDN(k, 4) = R(i)/N_;
                    k = k + 1;
                end
            end
        end
    end
    index = find(RDN(:,4) == min(RDN(:,4)) & RDN(:,4) > 0);
else

```

```

RDN(1,1:2) = [0 1];
RDN(1,3) = calc_N(L, RDN(1,1), RDN(1,2), min_inp, max_delay);
RDN(1,4) = 0;
index = 1;
end

R_ = RDN(index,1);
D_ = RDN(index,2);
N_ = RDN(index,3);

```

A.3 Algoritmo de Interleaving

```

%% DATA DE CRIAÇÃO: 29/03/2010
%% CRIADOR: Rodrigo Perazzo Rabelo
%% DESCRIÇÃO: Encontra as posições dos bytes após o interleaving
%% ENTRADA: N e D
%% SAÍDA: ordem do interleave
%%

function [ output_order, number_of_extra_bytes] = ...
    intrlv_idx_gen(N, D, total_fecs)

if(mod(N,2) == 0)
    input_order = 1:(N+1)*total_fecs;
    temp = zeros(((N+1)*total_fecs),1);

    for i=input_order
        delay = (D-1)*(mod(i-1, (N+1)))+i;
        temp(delay) = i;
    end

    dummy_idx = find(mod(temp,N+1) == 1);
    dummy_bytes = length(dummy_idx);

    for j=1:length(dummy_idx)
        temp(dummy_idx(j):end-1) = temp(dummy_idx(j)+1:end);
        dummy_idx = dummy_idx - 1;
    end

    output_order = temp(1:end-dummy_bytes);

    for k=1:length(output_order)
        if(output_order(k) ~= 0)

```

```

        output_order(k) = output_order(k) - ceil(output_order(k)/(N+1));
    end
end

else
    input_order = 1:N*total_fecs;
    output_order = zeros((N*total_fecs),1);

    for i=input_order
        delay = (D-1)*(mod(i-1,N))+i;
        output_order(delay) = i;
    end
end

zero_idx = find(output_order == 0);
number_of_extra_bytes = length(zero_idx);

counter = N*total_fecs+1;

for i=1:number_of_extra_bytes
    output_order(zero_idx(i)) = counter;
    counter = counter + 1;
end

```

A.4 Algoritmo para gerar as constelações QAM

```

%% DATA DE CRIAÇÃO: 29/03/2010
%% CRIADOR: Rodrigo Perazzo Rabelo
%% DESCRIÇÃO: Gera as constelações QAM
%% ENTRADA: número de bits e tabelas do ADSL2+
%% SAÍDA: Constelação QAM
%%
function [binaryOut, out] = SymbolCode( b , odd_msb_code)
M = 2^b;
out = [];
binaryOut = [];
for inteiro = 0:(M-1)
    binary = de2bi(inteiro,b);
    binaryOut = [binaryOut; binary];
    X2c = [1];
    Y2c = [1];
    if(mod(b,2) == 0)
        %even
    end
end

```

```

for j = 0:(b-1)
    if(mod(j,2) == 0)
        Y2c = [binary(j+1), Y2c];
    else
        X2c = [binary(j+1), X2c];
    end
end

msb = b/2;
else
    %odd
    if(b == 1)
        X2c = [binary, X2c];
        Y2c = [binary, Y2c];

    elseif(b == 3)
        out = [
            1.0000 + 1.0000i
            1.0000 - 1.0000i
            -1.0000 + 1.0000i
            -1.0000 - 1.0000i
            -3.0000 + 1.0000i
            1.0000 + 3.0000i
            -1.0000 - 3.0000i
            3.0000 - 1.0000i];

        out = out';
    else
        for j = 3:(b-1)
            if(mod(j,2) == 0)
                Y2c = [binary(j+1), Y2c];
            else
                X2c = [binary(j+1), X2c];
            end
        end
        index = find(odd_msb_code(:,1) == bi2de(binary(1:5), 'left-msb'));
        X2c = [odd_msb_code(index,2:3), X2c];
        Y2c = [odd_msb_code(index,4:5), Y2c];
    end

    msb = (b+1)/2;
end

```

```
if(b ~= 3)
    X = bi2de(X2c(2:end), 'left-msb');
    Y = bi2de(Y2c(2:end), 'left-msb');

    X = X - 2^(msb)*X2c(1);
    Y = Y - 2^(msb)*Y2c(1);
    out = [out, X+Y*i];
end
end
```

Referências Bibliográficas

- [1] ITU-T. Recommendation G.993.2: Very high speed digital subscriber line transceivers 2 (VDSL2). , 2006. Disponível em: <<http://www.itu.int/rec/T-REC-G.993.2-200602-I/en>>. Acesso em: 26 ago. 2010.
- [2] INTERNATIONAL DATA CORPORATION (IDC). Barômetro cisco de banda larga brasil 2005-2010: Resultados de Dezembro de 2009. , 2009. Disponível em: <http://www.cisco.com/web/BR/assets/docs/Barometro_Brasil_2H09_Site.pdf>. Acesso em: 04 out. 2010.
- [3] ITU-T G-series Recommendations. Disponível em: <<http://www.itu.int/itu-t/recommendations/index.aspx?ser=G>>. Acesso em 05 dez. 2010.
- [4] Internet lidera as reclamações do consumidor. Tele Síntese, São Paulo, 27 ago. 2009. Plantão. Disponível em: <<http://www.telesintese.com.br/index.php/plantao/12967>>. Acesso em: 06 dez. 2010.
- [5] Procon-SP autua Telefônica por falhas.Tele Síntese, São Paulo, 27 ago. 2009. Plantão. Disponível em: <<http://www.telesintese.com.br/index.php/plantao/13145>>. Acesso em: 06 dez. 2010.
- [6] INTERNATIONAL DATA CORPORATION (IDC). Barômetro cisco de banda larga brasil 2005-2010: Resultados de 2006. , 2006. Disponível em: <http://www.cisco.com/web/BR/barometro/barometro_2006.pdf>. Acesso em: 06 dez. 2010.
- [7] INTERNATIONAL DATA CORPORATION (IDC). Barômetro cisco de banda larga brasil 2005-2010: Resultados de Março de 2007. , 2007. Disponível em: <http://www.cisco.com/web/BR/barometro/barometro_1tri_2007.pdf>. Acesso em: 06 dez. 2010.
- [8] TENNYSON, Gary. *Challenges in DSL Network Management*. In: GLOBAL TELECOMMUNICATIONS CONFERENCE, 2008, New Orleans. .. New Orleans: IEEE, 2008. p.1-3.
- [9] P.J. Keyes, R.C. McConnell, and K. Sistanizadeh, ADSL: A New Twisted-Pair Access to the Information Highway, *IEEE Communications Magazine*, April 1995, pp 52-60
- [10] GOLDEN, Philip. et al. *Fundamentals of DSL Technology*. Boca Raton: Auerbach, 2006

- [11] Wikimedia Commons. Amplitude Modulation. Disponível em: <<http://commons.wikimedia.org/wiki/File:Amplitude-modulation.png>>. Acesso em: 11 dez 2010
- [12] Wikipedia, the free encyclopedia. Quadrature Amplitude Modulation. Disponível em: <http://en.wikipedia.org/wiki/File:16QAM_Gray_Coded.svg>. Acesso em: 11 dez 2010.
- [13] ITU-T. Recommendation G.992.3: Asymmetric digital subscriber line (ADSL) transceivers 2 (ADSL2). , 2005. Disponível em: <<http://www.itu.int/rec/T-REC-G.992.3-200501-S/en>>. Acesso em: 26 ago. 2010.
- [14] BODEN, Ralf. *Octet error probability analysis of adsl-dmt codewords*. International journal of eletronics and communcations. Amsterdam, p. 1. 17 jan. 2009.
- [15] Wikipedia, the free encyclopedia. Q-Function. Disponível em: <<http://en.wikipedia.org/wiki/File:Q-function.png>>. Acesso em: 11 dez 2010.
- [16] GALLAGER, Robert G.. *Information theory and reliable communication*. New York: John Wiley & Sons, 1968.
- [17] Wireless Broadband Network. An overview of OFDM. Disponível em <<http://wimaxcomnet.blogspot.com/2008/03/overview-of-ofdm.html>>. Acesso em: 11 dez 2010.
- [18] ITU-T. Recommendation G.992.5: Asymmetric digital subscriber line (ADSL) transceivers - extended bandwidth adsl2 (ADSL2plus). , 2005. Disponível em: <<http://www.itu.int/rec/T-REC-G.992.5-200501-S/en>>. Acesso em: 26 ago. 2010.
- [19] NEDEV, Nedko H.. *Analysis of the impact of impulse noise in digital subscriber line systems*. 2003. Tese (Doutorado em Engenharia) - The University of Edinburgh. Edinburgh
- [20] ITU-T Recommendation G.996.1: Test procedures for digital subscriber line (DSL) transceivers., 2001. Disponível em: <<http://www.itu.int/rec/T-REC-G.996.1-200102-I/en>>. Acesso em: 10 dez. 2010.
- [21] DAGNELIES, Arnaud. *Algebraic soft-decoding of Reed-Solomon codes*. 2007. Tese (Mestrado em Faculty of applied sciences) - Université Catholique de Louvain. Louvain-la-neuve.
- [22] CHEN, Walter Y.. *Dsl : simulation techniques and standards development for digital subscriber lines*. [S.l.]: Macmillan Technical Pub, 1998.
- [23] JAIN, Raj. *The Art of Computer System Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*. New York: John Wiley & Sons, Inc, 1991

- [24] JAREŠ, J. Vodrážka, P.. *Using simulations for qualification of adsl2+ lines*. In: INTERNATIONAL CONFERENCE ON SYSTEMS, SIGNALS AND IMAGE PROCESSING, 2006, Budapeste. .. Budapeste: IEEE 2006. p.1-4.
- [25] MATHWORKS. MATLAB: The Language Of Technical Computing. , 2010. Disponível em: <<http://www.mathworks.com/products/matlab/>>. Acesso em: 26 ago. 2010.
- [26] MATHWORKS. Communications Toolbox. , 2010. Disponível em: <<http://www.mathworks.com/products/communications/>>. Acesso em: 26 ago. 2010.
- [27] Thomson, images & beyond. SpeedTouch 510v6. Disponível em:<<http://www.speedtouch.com.br/st510v6.htm>>. Acesso em: 11 dez 2010.
- [28] Agilent Technologies. 33220A Function/Arbitrary Waveform Generator. Disponível em: <<http://www.home.agilent.com/agilent/product.jsp?nid=-536902257.536883183.00&cc=BR&lc=por>>. Acesso em: 11 dez 2010.
- [29] Spirent Communications – Operating Manual DSL410E – ADSL2+ and ADSL2++ Wireline Simulator.
- [30] Spirent. DLS, for accurate reproduction of last-mile conditions in the lab. Disponível em: <http://www.spirent.com/Broadband/Network_Emulation_Impairments/DLS.aspx>. Acesso em: 11 dez 2010.
- [31] Ericsson. EDA newer version. Disponível em: <http://www.ericsson.com/ourportfolio/products/eda-1200?nav=fgb_101_219|fgb_101_063>. Acesso em: 11 dez 2010.