# Implementação de um *Software-Defined Radio* com processamento em tempo real usando sinais acústicos e MATLAB

Fábio G. Fernandes†, Cristiano M. Panazio‡, Danilo Zanatta Filho†

***Resumo*— Este artigo propõe o uso do *software* MATLAB® e seu *Toolbox* Data Acquisition (DAQ) como uma plataforma simples e acessível para a prática de processamento de sinais em tempo-real. Ilustramos a proposta através da análise da implementação em software de um enlace digital de comunicação, também conhecido como *software-defined radio*, com processamento em tempo-real e que utiliza sinais acústicos como forma de transmissão.**

***Palavras-Chave*— MATLAB, tempo-real, modem acústico, rádio definido por software, projetos, processamento digital de sinais.**

***Abstract*— This paper proposes the use of the MATLAB® software and its Data Acquisition (DAQ) toolbox as a simple and accessible platform for the practice of signal processing in real-time. We illustrate this proposal through the analysis of an implementation of a software-defined radio with real-time processing and using acoustic signals.**

***Keywords*— MATLAB, real-time, acoustic modem, software-defined radio, projects, digital signal processing.**

## I. Introdução

Pode-se dizer que o ensino de processamento digital de sinais (PDS) em laboratório e, em especial, voltado para telecomunicações, apresenta basicamente dois grandes problemas: o preço das placas de desenvolvimento e a especificidade das mesmas e de sua programação. No primeiro caso, as placas baseadas em DSP (Digital Signal Processor) ou FPGA (Field Programmable Gate Array), juntamente com seus *softwares*, custam centenas ou até mesmo milhares ou dezenas de milhares de dólares. Tal fato pode facilmente inviabilizar a aquisição de um número adequado de placas para atender satisfatoriamente vários alunos. Além disso, tais placas necessitam conhecimentos muitas vezes específicos de sua configuração, além de um bom conhecimento de linguagem *Assembly* ou C/C++ no caso do DSP e de VHDL (Very High Speed Integrated Circuit (VHSIC) Hardware Description Language) ou Verilog para FPGA[1]. Desta forma, gasta-se uma boa parte do tempo ensinando essas especificidades, que evidentemente tem serventia para a vida profissional dos alunos, mas que tendem a fugir do escopo do curso, que é o ensino de PDS e princípios de telecomunicações.

Por outro lado, a maioria das universidades dispõe do *software* MATLAB e boa parte dos alunos de engenharia elétrica tem contato com este em várias disciplinas, em especial automação e controle, PDS e, evidentemente, telecomunicações. O MATLAB é uma linguagem de alto-nível e de fácil aprendizagem e sintaxe. Ela conta ainda com um ótimo sistema de geração de gráficos e funções nativas ou em *toolboxes*, muito úteis para várias aplicações. Entre elas está a possibilidade de gerar e tratar sinais de áudio, sejam eles na forma de arquivo através do *toolbox audio*, sejam diretamente na placa de som do PC (ou placas de aquisição) e em tempo-real, usando o *toolbox Data Acquisition* (DAQ). Vale ressaltar a facilidade de alterar rapidamente o código dos programas e a capacidade de mudar valores de variáveis em tempo-real durante o processamento, permitindo, por exemplo, alterar coeficientes de filtros, passos de adaptação, entre outros. Ainda, é possível visualizar com o próprio PC os resultados em tempo-real, dispensando a presença de caros osciloscópios ou analisadores de espectro.

Somada a essas características, a capacidade de processamento das CPUs dos PCs vem aumentado de forma vertiginosa, permitindo que operações antes destinadas a DSPs ou FPGAs possam ser realizadas sem problemas pelos atuais PCs. Vale citar como exemplo o sistema GNU Radio [1], que é um rádio definido por *software*, também conhecido por *Software-Defined Radio* (SDR), onde apenas o *down-conversion*, o *up-conversion* e alguns processos de filtragem são feitas em um módulo composto de um simples FPGA. As demais operações (*e.g.*, controle automático de ganho, modulação/demodulação digital, codificação e decodificação de canal) são realizadas pela CPU do PC em sinais cujas largura de faixa pode chegar a alguns megahertz [1]. Sendo assim, mesmo com a ineficiência computacional do MATLAB, uma linguagem interpretada e não compilada, a CPU de um PC relativamente atual é capaz de prover poder de cálculo suficiente para tratar em tempo-real o sinal obtido através da placa de som.

Inspirados neste contexto, foi proposto, como trabalho para um aluno, a implementação da camada física de um SDR operando com sinais acústicos, em tempo-real, e que inclui os principais blocos de um sistema de comunicação digital: modulação/demodulação, sincronismo temporal, de freqüência e de fase, e equalização adaptativa.

Tal idéia já havia sido pensada por Hwang em [2], no qual é proposto e relatado o uso do MATLAB para o desenvolvimento de um SDR usando sinais acústicos, a técnica *orthogonal frequency division multiplexing* (OFDM) e modulação 4-QAM (*Quadrature Amplitude Modulation*). Em [3], um outro sistema, que utiliza modulação diferencial QAM, foi implementado em tempo-real. Ainda, temos o caso do projeto do curso EE6058 da Georgia Tech [4] em que dever-se-ia

†Laboratório de Processamento Digital de Sinais para Comunicações (DSPCOM), FEEC, UNICAMP, Campinas, Brasil. ‡Laboratório de Comunicações e Sinais (LCS), Escola Politécnica da Universidade de São Paulo, São Paulo, Brasil. Emails: fgfernandes@yahoo.com.br, cpanazio@lcs.poli.usp.br, daniloz@decom.fee.unicamp.br

[1]Existem ferramentas que permitem codificar tanto o DSP como o FPGA usando o MATLAB/Simulink. Contudo, essas ferramentas costumam ter limitações significativas

projetar um modem acústico, mas com processamento *off-line*.

Nosso projeto difere da proposta de [2], [3] na forma de transmissão/recepção. Enquanto esses conectam diretamente a saída e a entrada da placa de som através de um fio, nós usamos, assim como em [4], a(s) caixa(s) de som para a transmissão e um microfone para a recepção. Neste caso, podemos obter na prática multipercursos e introduzir ruídos e perturbações naturais e não simuladas. Em [2], [3], o canal era simulado através de filtros digitais. Cabe ainda lembrar que como fazemos o processamento em tempo-real, ao se movimentar o microfone em frente das caixas de som (ou vice-versa), causamos desvanecimento, efeito Doppler e variações do canal. Estes fenômenos podem ser observados em tempo-real, diferentemente de [4]. Vale notar que isto é possível graças ao fato de que a velocidade do som é muito inferior a velocidade da luz, o que permite que o movimento de frações de quilômetros por hora causem perturbações consideráveis para sistemas com taxas de alguns quilo-símbolos por segundo.

Cabe ressaltar que, apesar desta proposta ter cunho eminentemente didático, a implementação de modems baseados em sinais acústicos tem sido bastante pesquisado e encontra também aplicações práticas. Citando alguns exemplos, temos o uso em sistemas sub-aquáticos para transmissão de dados de robôs de mergulho [5], [6] e propostas de transmissão de informação embebidas em marca d'água no sinal de áudio [7].

Este trabalho se encontra organizado da seguinte forma. A seção 2 discute os possíveis contextos de aplicação desta proposta. A seção 3 versa sobre como foi feito o processamento em tempo-real através do *toolbox* DAQ. Já a seção 4 detalha a implementação dos blocos que compõem o modem. Na seção 5, expomos os resultados obtidos do sistema implementado. Finalmente, a seção 6 estabelece nossas conclusões e perspectivas.

## II. Aplicações deste trabalho

A implementação de um enlace digital através do MATLAB e placa de som pode ser aplicada a laboratórios, como exercício de disciplinas teóricas, projetos de fim de curso, iniciações científicas, competições entre alunos e mesmo para pesquisa de novas técnicas e algoritmos.

Em especial para o laboratório ou exercício de disciplinas, as funções que compõem o modem podem ser desmembradas em vários blocos que estariam criptografados (*pcode* do MATLAB). Desta forma, os alunos poderiam ir substituindo os blocos que constituem o modem a medida que as experiências/projetos vão ocorrendo. Prevemos distribuir nosso modem neste formato para aqueles que desejarem, e mesmo entregar o código fonte aberto para os professores, de modo a permitir um melhor entendimento do programa, assim como mudanças mais profundas nele.

Cabe lembrar que, apesar de termos implantado um sistema SISO (*Single-Input Single-Output*), a capacidade estereofônica da placa de som permite a implementação de sistemas MIMO (*Multiple-Input Multiple-Output*).

Evidentemente, esta iniciativa de usar a placa de som e MATLAB não é restrita ao ensino de telecomunicações, podendo também ser aplicada ao ensino de PDS, como é feito em [8].

## III. MATLAB como plataforma para processamento digital de sinais em tempo real: o Toolbox DAQ

O *toolbox Data Acquisition* (DAQ) permite gerar interrupções sob diversas condições (*e.g.*, periodicamente, quando for capturado um certo número de amostras, quando amostras atingem um certo limiar de tensão, etc) chamar funções e tratar ou enviar dados em tempo-real para placas de aquisição, incluindo a própria placa de som do PC. Sua configuração é extremamente simples e alguns exemplos estão disponíveis juntamente com o próprio *toolbox*.

Adotamos neste trabalho o procedimento de interrupção baseado num certo número de amostras. Quando a interrupção tem lugar, acionamos as rotinas para tratamento do sinal recebido. Logo, existe uma latência no tratamento das amostras que é proporcional ao número de amostras usadas para disparar a interrupção. Contudo, tal latência fica abaixo de meio segundo e, praticamente, mal é percebida pelo usuário.

## IV. Estudo de caso: enlace digital usando sinais acústicos com processamento em tempo-real

O modem acústico foi implementado com base no modelo da Figura 1. Toda a parte anterior ao conversor D/A e posterior ao conversor A/D foi implementada por *software*. A placa de som (em *hardware*, portanto) foi usada para a interface analógico-digital e o canal de transmissão consistiu dos alto-falantes, do microfone e do meio pelo qual o sinal acústico se propagou.

Todo o processamento feito no transmissor foi implementado em uma função do MATLAB, que divide uma seqüência de bits em blocos, insere as seqüências de treinamento, faz a modulação e a filtragem de transmissão. O processamento do receptor está ligado às funções chamadas pelo DAQ e é efetuado a cada bloco recebido. Transmissão e recepção podem ser executados em um único PC ou em PCs diferentes, independentemente.

### A. O sinal transmitido

O sinal transmitido utiliza modulação QPSK (*Quadrature Phase Shift-Keying*), filtro de transmissão do tipo raiz de cosseno levantado com fator de *roll-off* 0,75, taxa de símbolo de 3675 símbolos/s, seqüência de treinamento de 63 símbolos no início de cada bloco, obtida através de uma seqüência pseudo-aleatória. A freqüência da portadora foi de 7500 Hz e a faixa de freqüência utilizada foi de aproximadamente 6400 Hz. A escolha destes parâmetros se deu em grande parte devido ao fato que o microfone utilizado corta os sinais após a freqüência de 12 kHz. Outro fator que deve ser levado em conta na escolha dos parâmetros é a presença de sinais parasitas de banda estreita, oriundos da própria placa de som ou da fonte de alimentação do PC. Tanto o corte do microfone e os sinais parasitas podem ser facilmente detectados ao se utilizar um "analisador de espectro", implementado pelo programa *demoai_fft* do *toolbox* DAQ.

O tamanho escolhido para cada bloco de dados, e por conseqüência para cada aquisição, é de 17640 amostras. Isto corresponde a 0,4 segundos para a taxa de amostragem usada, que é de 44100 Hz.

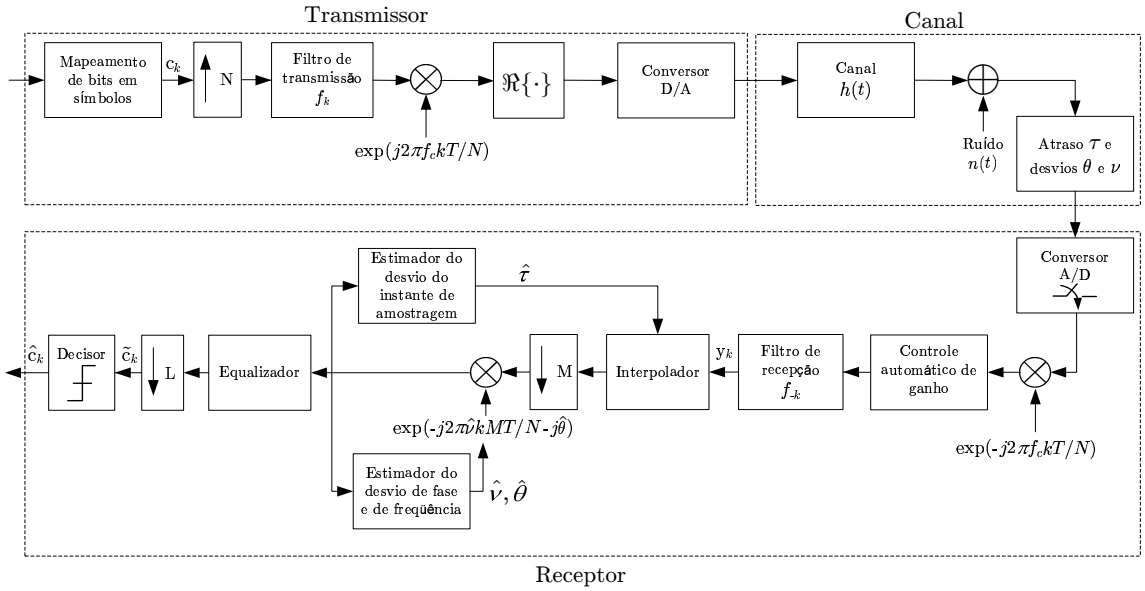


Fig. 1. Modelo do enlace para a implementação do modem acústico. As taxas dos decimadores dependem dos algoritmos de sincronismo ou equalizador (taxa de símbolo ou fracionário) utilizados e devem ser tais que $LM = N$.

O sinal acústico de entrada é gravado em blocos pelo DAQ, ao mesmo tempo em que o processamento do bloco anterior acontece. Neste modem acústico, o tamanho do bloco gravado é igual ao do bloco transmitido. No entanto, o instante inicial da gravação do bloco de amostras de som não corresponde necessariamente ao início do bloco de dados transmitido, como veremos a seguir. Por este motivo, o seguinte procedimento é feito para garantir que o bloco a ser processado seja realmente um bloco de dados enviado:

1) Um bloco de tamanho fixo de amostras de som é adquirido;
2) Este bloco é filtrado, preservando a faixa de freqüências do modem acústico;
3) A potência do sinal é calculada;
4) Se a potência estiver acima de um dado limiar, o que indica a presença do sinal, o sinal é demodulado e é feita sua correlação com a seqüência de treinamento, procurando o pico que indica o começo de um bloco de dados transmitidos;
5) Se for identificado o começo de um bloco de dados, as amostras recebidas a partir deste ponto são salvas em um *buffer*;
6) Na aquisição das amostras seguintes de som, a quantidade de amostras necessária para completar um bloco de dados é concatenada às amostras guardadas no passo anterior;
7) O bloco de dados completo, salvo no *buffer*, é então processado;
8) A primeira parte do bloco de dados seguinte é gravada;
9) O sistema retorna ao passo 6.

Se a potência calculada no passo 3 estiver abaixo do limiar, nenhum processamento adicional será feito nessa aquisição, e o sistema volta ao passo 1. Este limiar serve para que não seja feito processamento desnecessário na ausência de sinal. No entanto, o valor do limiar depende da sensibilidade do microfone e da amplificação da placa de som. Portanto, para um bom funcionamento, esse parâmetro deve ser ajustado manualmente em função da configuração utilizada ou encontrado de forma adaptativa.

Durante o processamento do bloco de dados no passo 6, é verificado através da correlação se a seqüência de treinamento está presente. Caso ela não esteja, é identificado o final de uma transmissão, e o sistema retorna ao passo 1.

Para tornar possível a transmissão por um canal real, foi necessário implementar algoritmos de sincronização e equalização que garantissem uma boa recepção. Foram utilizados métodos que requerem treinamento para correção de desvios de sincronismo e equalização, de modo a evitar que estes atingissem pontos espúrios de equilíbrio, nos quais a taxa de erro é muito alta.

### *B. Controle automático de ganho*

A potência do sinal recebido está sujeita à variações (*e.g.*, variações da perda de propagação devido ao movimento do microfone etc). Uma vez que o passo de adaptação da maioria dos algoritmos adaptativos é ajustado para uma determinada potência, essas variações podem causar comportamentos indesejáveis, tais como a divergência ou mau rastreamento das mudanças do canal, no caso do equalizador, e escorregamentos de fase e de símbolo, no caso do sincronismo. Desta forma, adota-se um controle automático de ganho (CAG), para que as variações de potência do sinal sejam compensadas, através de uma normalização.

Adotamos, nesta implementação, uma forma bem direta de CAG. Uma vez que dispomos do bloco de dados para o processamento, calculamos a potência deste, normalizando-o em seguida.

### *C. Sincronismo*

Por tratarmos de um canal real (sem fio neste caso), a sincronização da recepção com o sinal recebido é essencial. Desvios de sincronismo temporal e de portadora são inerentes a este tipo de transmissão, nos quais há efeitos como o Doppler, além dos desvios iniciais de sincronismo e da

taxa de amostragem. Como a transmissão e recepção são feitas em tempo real, é possível visualizar os efeitos sobre a comunicação que o deslocamento do microfone ou da caixa e a colocação de anteparos exercem.

A comunicação é feita em blocos de 0,4 segundos e, durante este tempo, os parâmetros de sincronismo podem ter mudanças consideráveis. Algoritmos de malha fechada e de malha aberta foram implementados e o usuário pode compará-los e visualizar as diferenças de desempenho entre eles.

*1) Sincronismo de Portadora:* Como a modulação utilizada no modem acústico foi a QPSK (*Quadrature Phase-Shift Keying*), a informação está contida na fase do símbolo enviado. Daí a necessidade de estimar e corrigir os desvios de sincronismo de portadora do sinal recebido. O uso da modulação DQPSK (*Differential* QPSK) em vez de QPSK eliminaria a necessidade de sincronismo de fase de portadora, mas não de freqüência da portadora.

Para corrigir desvios de fase da portadora foram implementados dois algoritmos: o estimador de fase ML de malha aberta [9] e o *Costas Loop*, um estimador de fase em malha fechada [9].

O estimador de fase de malha aberta implementado é uma técnica que requer treinamento (*Data-Aided* em inglês, ou DA), e é adequada para quando o desvio de fase não varia significativamente entre as seqüências de treinamento. O estimador é definido pela seguinte equação

$$\hat{\theta} = \arg\Big\{\sum c_k^* y_k\Big\} \ , \tag{1}$$

onde $\hat{\theta}$ é a estimativa do desvio de fase, $c_k$ é o *k*-ésimo símbolo da seqüência de treinamento e $y_k$ é o *k*-ésimo símbolo recebido. Em [9] é provado que a variância deste estimador coincide com o limite modificado de Cramér-Rao

$$LMCR(\theta) = \frac{1}{2L_0 E_s/N_0} \ , \tag{2}$$

onde $L_0$ é o número de símbolos observados, $E_s$ é a energia do símbolo e $N_0$ é a variância do ruído.

O método de malha fechada *Costas Loop* [9] é mais adequado para quando o desvio de fase $\theta$ varia ao longo do bloco de transmissão. Ele pode ser chaveado do modo DA para o modo DD (*Decision-Directed*) após o termino da seqüência de treinamento. As seguintes expressões definem seu funcionamento

$$e_k = \Im(c_k^* y_k e^{-j\hat{\theta}_k}) \tag{3}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \gamma e_k \ , \tag{4}$$

onde $\gamma$ é o passo de adaptação do estimador e o sinal de erro $e_k$ decorre da minimização da função de verossimilhança. Para funcionar no modo DD, $c_k^*$ deve ser substituído pelo símbolo decidido $\hat{c}_k^*$. Também devido a sua implementação por *software*, podemos facilmente trocar de algoritmos, parâmetros de adaptação e ver os efeitos no desempenho.

Se houver desvios de freqüência de portadora, que podem ser causados, por exemplo, por efeito Doppler ou diferença na taxa de amostragem entre transmissão e recepção, o *Costas Loop* apresentado acima pode não ser suficiente.

Foram utilizados dois métodos para lidar com desvios de freqüência. O primeiro consiste de uma modificação do *Costas loop*, que transforma a malha em um sistema de segunda ordem ao acrescentar um integrador, desta maneira agindo para anular o erro e sua derivada.

O segundo método é de malha aberta e estima diretamente o desvio de freqüência. Foi proposto por Fitz [10], e requer uma seqüência conhecida de símbolos. A partir do sinal demodulado $y_k = c_k\, e^{(j\,2\,\pi\, f_d\, k\, T + j\theta)} + n_k$, em que $f_d$ é o desvio de freqüência, $\theta$ é o desvio de fase e $n_k$ é o ruído aditivo Gaussiano, faz-se

$$z_k = c_k^*\, y_k \tag{5}$$

$$R(m) = \frac{1}{L_0 - m} \sum_{k=m}^{L_0-1} z_k\, z_{(k-m)}^* \ . \tag{6}$$

em que $R(m)$ é a autocorrelação do sinal auxiliar $z_k$.

Substituindo (5) em (6), temos

$$R(m) = e^{(j\,2\,\pi\, m\, f_d\, T)} \ . \tag{7}$$

O estimador é definido por

$$\hat{f}_d = \frac{1}{\pi N(N+1)T} \sum_{m=1}^{N} \arg\{R(m)\} \ . \tag{8}$$

O valor de $N$ é arbitrário, e está ligado ao máximo desvio estimável $f_{d\,max}$

$$N < \frac{1}{2\,|f_{d\,max}|\,T} \ . \tag{9}$$

*2) Sincronismo temporal:* O sincronismo temporal é essencial para minimizar a interferência intersimbólica (IIS), rastrear as variações de atraso de propagação e/ou erros na taxa de amostragem. As principais causas de desvios de sincronismo temporal presentes no modem acústico são: o fato de o instante inicial de amostragem no receptor não ser necessariamente o instante ótimo; o deslocamento relativo do microfone e do alto-falante, que comprime ou expande o símbolo no tempo; uma pequena diferença de freqüência de amostragem entre reprodução e gravação do sinal acústico. A última faz com que estimadores de desvio de sincronismo temporal de malha aberta tenham pior desempenho, pois o desvio muda constantemente. Isto gera também um desvio constante de freqüência de portadora, já que, no modem acústico, ela é gerada digitalmente, e a correção do desvio é feita após a *down-conversion*. Há, portanto, uma necessidade constante de correção de desvios de sincronismo temporal e de portadora.

Dada a impossibilidade de manipular a fase da amostragem da placa de som, implementamos uma correção totalmente digital do sincronismo temporal através da utilização de filtros interpoladores digitais [11], [12]. Neste projeto, fazemos uso de um interpolador linear

$$y(kT + \tau) = \tau' y(k'T) + (1 - \tau') y((k'+1)T) \ , \tag{10}$$

onde $y$ é o sinal recebido, $T$ é o período de símbolo, $\tau'$ é a parte fracionária do desvio de sincronismo temporal e $k'$ é a soma de $k$ com a parte inteira de $\tau$, que é o desvio de sincronismo temporal. Optamos por trabalhar com a

interpolação linear de sinais amostrados com o dobro da taxa de símbolo, uma vez que a perda de desempenho neste caso é inferior a 0,05 dB [12].

A correção do desvio depende de sua estimação, que no modem acústico foi implementada de diversas maneiras, que podem ser escolhidas pelo usuário. O estimador de malha aberta implementado é o *Oerder & Meyr* [9], que é um estimador cego. Por isto, ele deve ser utilizado somente quando houver certeza que $|\tau| < T$, para que o estimador não atinja o sincronismo com um símbolo anterior ou posterior. Ele é uma alternativa ao método ML, de maior custo computacional, e é baseado em argumentos heurísticos. O estimador final corresponde à seguinte expressão

$$\hat{\tau} = -\frac{T}{2\pi} \arg\left(\sum_{k=0}^{NL_0} |y(kT_s)|^2 e^{-j2\pi k/N}\right) , \qquad (11)$$

onde $\hat{\tau}$ é o estimador do desvio $\tau$, $N$ é o fator de super-amostragem (que deve ser maior ou igual a 4), $T_s$ é o período de amostragem e $L_0$ é o número de períodos de símbolo utilizados na estimação.

Foi também implementado o estimador de malha fechada *Early-Late Detector* (ELD) [9]. Ele funciona em modo DA durante a seqüência de treinamento e no modo DD no restante do bloco. É baseado no fato de o valor médio da derivada do sinal ser nulo no instante ótimo de amostragem. Por isto, o valor realimentado é proporcional à aproximação da derivada do sinal no ponto desejado:

$$e_k = c_k^* \left[ y(kT + T/2 + \hat{\tau}_k) - y(kT - T/2 + \hat{\tau}_k) \right] . \qquad (12)$$

Este método é uma aproximação do método de máxima verossimilhança, o qual utiliza o verdadeiro valor da derivada.

### *D. Equalização*

No modem acústico estão presentes dois tipos de equalizadores: o linear e o DFE [13], ambos adaptados com o algoritmo *Least Mean Square* (LMS) [13]. Nos testes, como era esperado, fica visível que a escolha do DFE melhora consideravelmente o desempenho do receptor em canais fortemente seletivos em freqüência. Ambos equalizadores foram implementados também em versão fracionária, com filtros de *taps* a cada meio período de símbolo. Notamos que os equalizadores fracionários têm desempenho superior, o que é explicado pelo fato do alto valor de roll-off (excesso de banda) usado e da possibilidade de corrigir algum desvio de sincronismo temporal causado pela presença de seletividade em freqüência do sinal na recepção.

## V. RESULTADOS

Foram realizados diversos testes com o modem acústico em tempo real, com diferentes configurações espaciais do microfone e caixa de som, de modo a visualizar seus efeitos no sinal e na recepção. Os algoritmos de sincronismo utilizados nos testes foram o ELD para sincronismo temporal e o *Costas Loop* de segunda ordem para sincronismo de portadora. Eles foram escolhidos por terem bom desempenho nas situações em que os parâmetros variam muito ao longo do tempo, que

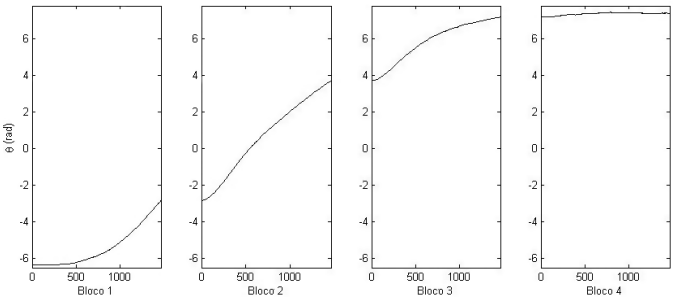


Fig. 2. Comportamento do desvio de fase ao longo de 4 blocos de dados consecutivos, à medida que o microfone se aproxima da caixa de som

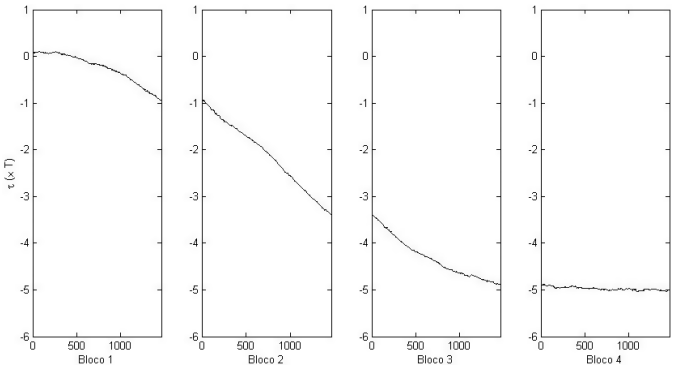


Fig. 3. Comportamento do desvio temporal ao longo dos mesmos 4 blocos da Figura 2

é o que verificamos na prática. O equalizador utilizado foi o DFE fracionário.

O primeiro teste realizado teve como objetivo analisar o comportamento do modem acústico com o deslocamento do microfone. As Figuras 2 e 3 mostram a evolução dos desvios de fase e tempo estimados enquanto o microfone era aproximado da caixa de som. Podemos ver que o valor do desvio de fase aumenta e o valor do desvio temporal diminui, o que pode ser explicado pelo efeito Doppler. Ao aproximarmos o microfone da caixa de som, a freqüência de portadora percebida no microfone aumenta, elevando assim o desvio de fase. Como a aproximação causa compressão do sinal no tempo, os pulsos passam a ter duração menor, e o instante ótimo de amostragem passa a acontecer cada vez mais cedo. Por conseqüência, o desvio temporal diminui de valor.

Em outro teste, o microfone e caixa de som permaneceram parados, enquanto uma placa metálica circular de 15 cm de diâmetro foi aproximada do microfone e em seguida retirada.

Na Figura 4, vemos os gráficos de resposta em freqüência do canal estimado ao longo de 4 blocos. No bloco 1, o anteparo estava sendo aproximado do microfone. Nos blocos 2 e 3, ele estava próximo do microfone. No bloco 4, ele foi afastado. Podemos notar que a atenuação máxima do canal aumenta significativamente nos blocos 2 e 3, em relação aos blocos 1 e 4, devido à propagação multi-percurso causada pelo anteparo.

Foi testado ainda outro canal de transmissão, ainda com microfone e caixa de som parados, mas com um tubo plástico de 5 cm de diâmetro em volta do microfone, de modo a gerar IIS. O canal estimado tem a resposta em freqüência mostrada na Figura 5. Podemos ver que há freqüências que são atenuadas em até 17dB.

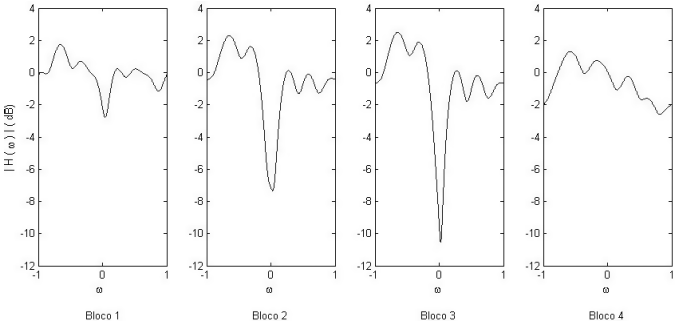


Fig. 4. Resposta em freqüência do canal estimado, durante a aproximação de um anteparo

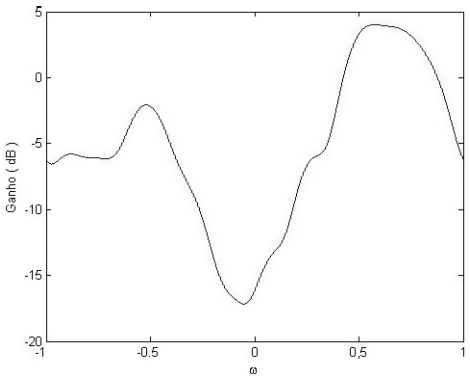


Fig. 5. Magnitude da resposta em freqüência do canal, quando o microfone está no interior do tubo plástico

O objetivo deste teste é avaliar a mudança no atraso de treinamento do equalizador no erro quadrático médio. Para isto, o sinal gravado neste teste foi processado de duas maneiras diferentes: sem atraso de treinamento e com atraso de 3 períodos de símbolo. O filtro de alimentação do DFE usado tinha tamanho de 4 períodos de símbolo. A diferença entre os resultados é visível na comparação da Figura 6, em que 6-a mostra maior espalhamento que 6-b. O erro quadrático médio calculado caiu de 0,0799 para 0,0327 com a adição do atraso de treinamento, ou seja, um ganho de aproximadamente 4dB. Outros testes, com canais menos seletivos em freqüência, não mostraram ser tão dependentes do atraso de treinamento quanto este.

### A. Utilização da CPU

Apenas o processamento do sinal utiliza cerca de 60% a 70% de um Pentium IV de 3 GHz, com 512 MB de memória RAM, com tempo de processamento por bloco de 0,26 s. Ao se plotar os resultados ao fim de cada bloco, o uso da CPU salta para até 90%, e leva 0,33s para processar um bloco. Isto mostra que o uso de gráficos deve ser feito de forma parcimoniosa. Mesmo com os gráficos, no entanto, o processamento neste PC foi mais rápido que a duração do bloco, que é de 0,4s.

## VI. Conclusão e Perspectivas

Mostramos neste artigo a viabilidade de se implementar um enlace digital completo, com processamento em tempo-real, usando um simples PC com placa de som e o *software* MATLAB. Nota-se uma maior motivação dos alunos e professores por conseguirem pôr em prática e observar facilmente os resultados dos conhecimentos que antes ficavam restritos à teoria ou simples simulações passadas como exercícios. Ainda, a implementação prática em MATLAB possibilita que o aluno despenda menos tempo com problemas relacionados ao *hardware* ou *software*, e se concentre mais no funcionamento dos algoritmos.

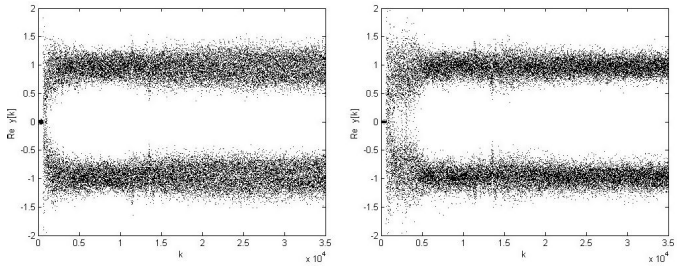


Fig. 6. Parte real do sinal equalizado, para os atrasos de treinamento: (a) 0 e (b) 3T

Vale ressaltar que a aplicação do modem não fica restrita a sinais acústicos, podendo também ser aplicado em sistemas com fio tais como linhas telefônicas e linhas de transmissão de energia (*Power Line Communications*). Uma idéia interessante é a de se usar circuitos passivos que correspondam ao comportamento destas linhas, de modo a melhor controlar os ambientes aos quais o modem é submetido. Ainda é possível, neste caso, controlar a inserção de ruído antes da amostragem na entrada da placa e assim ter uma melhor idéia do desempenho do sistema implementado.

Ainda, é importante lembrar que sistemas MIMO também são implementáveis, graças à capacidade estereofônica da placa.

## Referências

[1] "GNU Radio - GNU FSF project,"em http://www.gnu.org/software/gnuradio/.

[2] Jeng-Kuang Hwang, *An Innovative Communication Design Lab Based on PC Sound Card and MATLAB: a SDR OFDM modem example*. *IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), Hong Kong, Abril 2003.

[3] Jeng-Kuang Hwang, Yu-Lun Chi e Rih Lung Chung, *Design of NDA Differential QAM modem for Real-time Communications A SDR PC based approach* ICCS 2004.

[4] EE6058: Digital Communications II, School of Electrical and Computer Engineering Georgia Institute of Technology , Spring 1999, responsável: Prof. John Barry, http://users.ece.gatech.edu/∼barry/6058/proj

[5] M. Stojanovic, "Recent advances in high-speed underwater acoustic communications" *IEEE Journal of Oceanic Engineering* v. 21, i. 2, p. 125-136, Abril 1996

[6] J. Gomes, V. Barroso, G. Ayela e P. Coince, "An Overview of the ASIMOV Acoustic Communication System" *IEEE Conference and Exhibition OCEANS 2000 MTS*,p. 1633-1637, vol.3, 11-14 setembro 2000.

[7] A.N. Lemma, J. Aprea, W. Oomen e L. van de Kerkhof, "A temporal domain audio watermarking technique"IEEE Transactions on Signal Processing, p. 1088 - 1097, Abril 2003.

[8] ECE1563: Digital Signal Processing, School of Electrical & Computer Engineering of the University of Pittsburgh, Spring 2007, responsável: Prof. J. Robert Boston. http://www.engr.pitt.edu/electrical/faculty-staff/boston/1563/ee1563.htm

[9] Mengali, U. e D'Andrea, A., (1997). "Synchronization Techniques for Digital Receivers", Plenum Press, New York.

[10] M.P.Fitz, Planar Filtered Techniques for Burst Mode Carrier Synchronization, GLOBECOM'91, Phoenix, Arizona, Dezembro 1991.

[11] F.M. Gardner, "Interpolation in digital modems - Part I: Fundamentals,"IEEE Trans. Commun., vol. 41, pp. 502-508, Mar. 1993.

[12] L. Erup, F.M. Gardner, R.A. Harris, "Interpolation in Digital Modems-Part II: Implementation and Performance, "IEEE Trans. Commun., vol. 41, pp. 998-1008, Jun. 1993.

[13] S. Haykin. Adaptive Filter Theory (2nd Edition). Prentice Hall, Englewood Cliffs, NJ, 1990.