Neste capítulo abordaremos as técnicas anteriormente mencionadas, que são o foco deste trabalho, PLL (Phase-Locked Loop), análise de prony, ADALINE e RLS (Recursive Least Mean Squares), bem como as convencionais DFT e STFT. Ressaltamos que esta será apenas uma abordagem superficial, não entraremos, portanto, com profundidade nos assuntos tratados.

0.1 Análise espectral

Análise espectral considera o problema de encontrar a energia de um sinal, finito, distribuída em função de frequência, o que pode ser feito com métodos paramétricos ou não paramétricos. Métodos paramétricos assumem conhecimento do modelo com o qual se gerou o sinal em análise. Por exemplo, em alguns capítulos, como o de análise de Prony, assumiremos que nosso sinal é composto unicamente por un número conhecido de exponenciais complexas amortecidas. Para solucionar um problema de estimação paramétrica, a única coisa que temos quer fazer (o que ainda não é fácil, na maioria dos casos) é encontrar os parâmetros supostos. Por outro lado, métodos não paramétricos se baseiam unicamente em transformações do domínio temporal para o domínio da frequência, utilizando filtros que nos dão a energia do sinal contida em uma determinada faixa do espectro, caso da DFT. Quando temos uma boa estimação de nosso modelo, os métodos paramétricos tendem a dar melhores resultados, enquanto que se pouco sabemos sobre o sinal analisado, ou se o mesmo não está bem modelado, os não paramétricos podem ser a melhor opção.

Algumas vezes é conveniente tratar os sinais de maneira determinística, e o faremos neste trabalho, entretanto, como abordagem mais geral, se pode tratá-los com enfoque probabilístico, isto é: admitimos que não se pode de maneira alguma prever os valores assumidos por um sinal ao longo do tempo, mas se pode estimar suas características por meios estatísticos. Lembramos também que embora sinais reais sejam o caso mais comum na maioria das aplicações e em nosso caso são praticamente regra, não necessariamente o tratamento de sinais complexos é muito mais complicado. Apenas por uma questão de praticidade e para deixar o trabalho mais conciso, não demonstraremos os teoremas utilizados para entradas complexas quando não for necessário.[11]

0.2 Densidade espectral de energia de um sinal determinístico

Seja $y_c(t)$ um sinal contínuo no tempo e $y[n],\ n = 0, \pm1, \pm2, ...$ um sinal discreto tal que $y[n] = y_c(nT_s)$. Consideremos que $y[n]$ tem energia finita. Podemos dizer então que a DTFT de y está definida como:

$$Y(w) = \sum_{n=-\infty}^{+\infty} y[n]e^{-jwn} \tag{1}$$

Onde w é denominada frequência angular medido em radianos/segundo, e pode ser relacionada com seu valor físico da seguinte maneira $\bar{w} = wT_s$. A densidade espectral de potência do sinal $y(t)$ pode ser dada então por:

$$S(w) = |Y(w)|^2 \tag{2}$$

Podemos interpretar este resultado de algumas formas, mas uma bem simples é lembrar que a DTFT de uma cossenoide é igual a soma de duas deltas refletidas no espectro de frequência:

$$\sum_{n=-\infty}^{+\infty} cos(2\pi f\ n)e^{-jwn} = \pi\delta_w(2\pi f) + \pi\delta_w(-2\pi f)\ \ w\ \in [0, 2\pi] \tag{3}$$

Uma relação muito parecida se estabelece no caso de uma senoide. Desta maneira, se imaginamos $yc(t)$ composta unicamente de uma soma de senoides, seu espectro fica muito bem explicitado com a equação XX.

Uma outra relação importante se dá utilizando a autocorrelação de $y[n]$:

$$r_y[k] = \sum_{n=-\infty}^{+\infty} y[n]y^*[n-k] \tag{4}$$

$$\begin{aligned} \sum_{k=-\infty}^{+\infty} r_y[k]e^{-jwk} &= \sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} y[n]y^*[n-k]e^{-jwk} = \\ &\sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} y[n]y^*[n-k]e^{-jwn}e^{jw(n-k)} = \\ \left[\sum_{n=-\infty}^{+\infty} y[n]e^{-jwn}\right]&\left[\sum_{s=-\infty}^{+\infty} y[s]e^{-jws}\right]^* = Y(w)Y(w)^* \end{aligned} \tag{5}$$

Vemos então que a DTFT da autocorrelação de um sinal é igual sua densidade espectral de energia.

0.2.1 Densidade espectral de potência

Para a maioria dos sinais com os quais lidamos, não temos um modelo puramente determinístico que os reproduza, não sabemos o valor exato que tomarão no futuro, e tampouco podemos estender seu valor até o infinito, ou analisar infinitas amostras do mesmo. Para os casos reais, é mais conveniente lidar com sequencias aleatórias ao longo do tempo, em que cada realização da sequência tem associada uma distribuição de probabilidade (até o momento de sua ocorrência, quando esta colapsa em algum valor). Para a maioria dos casos, consideraremos que estamos lidando com processos estacionários em sentido amplo[12]. Para este trato aleatório, não é regra que nossos sinais tenham energia finita, mas é possível ao invés disso, estimar sua densidade de potência.

Assumamos agora que $y[n]$ é uma sequência de variáveis aleatórias, com o mesmo domínio da sequência anterior, e que $E[y[n]] = 0$ para todo n. Tendo todas as variáveis aleatórias média nula. A função de correlação de $y[n]$ está definida como:

$$r_y[k] = E[y[n]\, y[n-k]^*] \tag{6}$$

Definimos então a densidade espectral de potência como:

$$\phi(w) = \sum_{k=-\infty}^{+\infty} r_y[k] e^{-jwk} \tag{7}$$

Uma segunda definição para a PSD (*Power Spectrum Density*) pode ser dada como a DTFT do sinal com o número amostras tendendo ao infinito da seguinte forma:

$$\phi(w) = \lim_{N\to\infty} \frac{1}{N} \left| \sum_{k=0}^{N-1} y[k] e^{-jwk} \right|^2 \tag{8}$$

Ambas definições são equivalentes. Veremos mais adiante que uma das formas principais de se calcular a PSD é por meio de estimações da função de autocorrelação do sinal em questão.

0.2.2 Periodograma e correlograma

Uma forma bastante simples de estimar a PSD seria:

$$\hat{\phi_p}(w) = \frac{1}{N} \left| \sum_{k=0}^{N-1} y[k] e^{-jwk} \right|^2 \tag{9}$$

A qual chamamos periodograma. Outra forma poderia ser:

$$\hat{\phi_c}(w) = \sum_{k=-(N-1)}^{N-1} r_y[k] e^{-jwk} \tag{10}$$

A qual chamamos correlograma. Dois estimadores para a autocorrelação serão apresentados abaixo, considerando $n = 0, 1, 2, ..., N-1$:

$$\hat{r}_y(k) = \frac{1}{N-k} \sum_{n=k}^{N-1} y[n]\, y^*[n-k] \tag{11}$$

$$\hat{r}_y(k) = \frac{1}{N} \sum_{n=k}^{N-1} y[n]\, y^*[n-k] \tag{12}$$

O primeiro estimador é chamado estimador padrão não viesado, e o segundo é um estimador viesado. Lembramos que é assim chamado (viesado) um estimador cuja esperança não é o que visa estimar. Fato é também que $\hat{\phi_p}$ é igual a $\hat{\phi_p}$ se $\hat{r_y}$ for estimada com estimador viesado.

Muitas vezes o estimador em XX é preterido ao de XX, porque para muitos sinais a autocorrelação para valores grandes de k (valores considerados longe do 0) é muito baixa, e portanto não ajudaria fazer a correção do fator de normalização. Entretanto esse pode não ser bem o nosso caso, porque estaremos considerando nossos sinais muitas vezes como senoides puras, as quais apresentam correlação em trechos periódicos. Em realidade, não iremos sequer adentrar muito em estimadores de correlação. Partiremos logo para análise de métodos relacionados a DFT. As vantagens e desvantagens destes métodos todos podem ser vistos à luz de processos estocásticos, ou pela análise das características do método em si. Tentaremos mostrar um pouco de ambos, os utilizando da maneira que for mais conveniente.

0.3 DFT

A transformada discreta de Fourier (DFT) de uma sequência discreta x[n] de tamanho N é definida como[5]:

$$X[k] = \sum_{n=0}^{n=N-1} x[n] e^{(-2jnk\pi/N)}\ ,\ k = 0, 1, 2, ..., N-1 \tag{13}$$

Podemos dizer que a sequência X[k] é a DFT de x[n]. As duas sequências tem o mesmo tamanho, e X[k] representa o mapeamento das frequências de

x[n] supondo estas estacionárias e com período igual ao analisado. Sendo Ts o tempo de amostragem da sequência x[n], temos a seguinte relação:

$$Rs = \frac{1}{TsN} \tag{14}$$

Onde Rs é igual a resolução de X[k]. Temos o mapeamento dado como:

$$fk = Rsk \ , \ k = 0, 1, 2, ..., N/2 \tag{15}$$

Percebemos que a DFT age como uma série de Fourier, onde valores absolutos de X[k] são vistos como a amplitude de uma determinada frequência múltipla da componente fundamental f1, que é igual a resolução Rs. Para um sinal contendo apenas harmônicos de f1, e eventualmente algum valor médio, podemos extrair perfeitamente seus valores de amplitude e fase. Entretanto, para um sinal que possui componentes inter-harmônicos, não é possível fazer o mesmo. Neste caso ocorre o chamado *espalhamento*. Já se pode notar algumas deficiências da DFT. Também temos problemas caso não seja amostrado um período exato do sinal analisado, ou um múltiplo inteiro de um período, podendo a DFT levar a crer, por exemplo, que existe um valor médio no sinal, e outros conteúdos equivocados. Este fenômeno é denominado *spectral leackage*[5].

Como a DFT pressupõe um sinal estacionário, ela não é adequada para análise de sinais variantes no tempo. Para tanto podemos utilizar de uma DFT de janela deslizante. Para uma janela de tamanho N, temos:

$$X[k, m] = \sum_{n=0}^{n=N-1} x[m-n]e^{(-2jnk\pi/N)} \ , \ k = 0, 1, 2, ..., N-1 \tag{16}$$

Desta forma, para cada m¿N, temos uma janela de tamanho N onde será analisado o sinal. Existem algoritmos mais eficazes para efetuar este tipo de cálculo de forma recursiva, sem precisar calcular toda a DFT como se estivéssemos diante de uma amostra completamente nova:

$$X[k, m] = C\{X[k, m-1]e^{j2\pi k/N} + (x[m] - x[m-N])e^{j2\pi k/N}\} \ , \ k = 1, 2, ..., N/2 \tag{17}$$

C=1/N para k=N/2 e igual a 2/N para os demais valores. O valor de k foi restringido levando-se em conta a simetria dos sinais reais.[Não sei de onde veio essa equação, vou ver depois]

Pode-se usar diferentes tipos de janelas além da anterior, que é uma janela quadrada, onde todos os termos da sequência tem igual peso no cálculo da DFT. O uso de outras janelas ajuda na amenização do *leackage*. Algumas janelas estão expostas abaixo.

Janela triangular:

$$w_{tri}(n) = 1 - \frac{|2n - N + 1|}{N - 1} \ , \ 0 \leq n \leq N - 1 \tag{18}$$

Janela de Hamming:

$$w_{hm}(n) = 0.54 - 0.46cos(\frac{2\pi n}{N - 1}) \ , \ 0 \leq n \leq N - 1 \tag{19}$$

O uso deste tipo de janelamento atenua eventuais descontinuidades nas extremidades do sinal, auxiliando na medição dos parâmetros. Aplicar uma janela ao sinal significa deslizar a sequência formada por w por x[n], multiplicando termo a termo. Desta forma poderíamos reescrever a equação XXX como:

$$X[k, m] = \sum_{n=0}^{n=N-1} x[m - n] * w(n) * e^{(-2jnk\pi/N)} \ , \ k = 0, 1, 2, ..., N - 1 \tag{20}$$

podemos ver nas figuras o efeito da aplicação do janelamento.

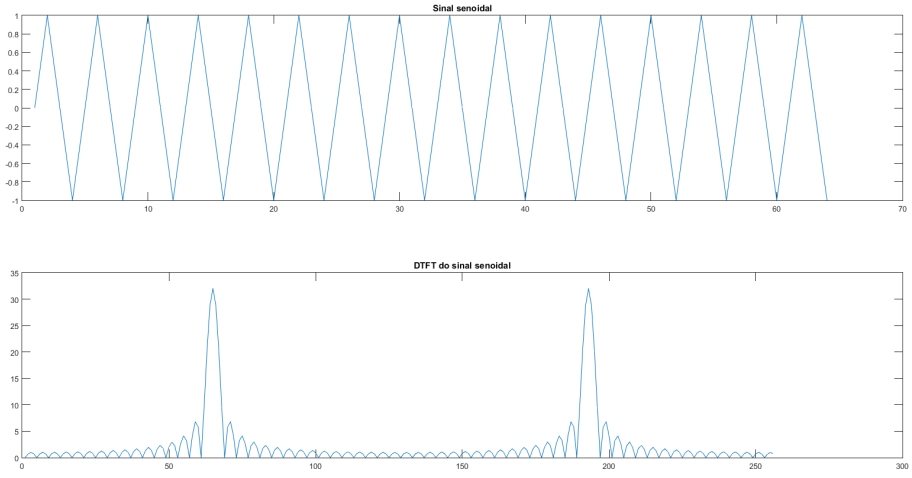


Figura 1: legenda aqui

0.4 Análise de Prony

A análise de Prony foi desenvolvida em 1795 de modo a explicar a espanção de gases. Ela se assemelha em parte a DFT, entretanto se propõe a ajustar uma soma de exponenciais complexas amortecidas a uma sequência de dados

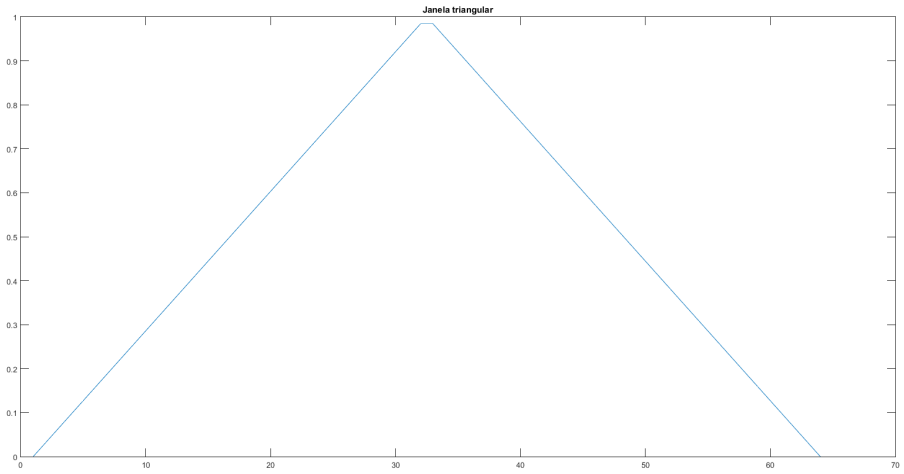


Figura 2: legenda aqui

igualmente espaçados, ao passo de que a DFT apenas estima as exponeciais complexas e em subdivisões predeterminadas da frequência de amostragem. A análise de prony não é somente uma técnica de análise de sinais, mas também de indentificação de sistemas amplamente utilizada em sistemas de potência, área biomédica, processamento de fala, decaimento radioativo entre outas. A análise de Prony é conhecida por não se comportar muito bem quando um sinal contém ruído, a técnica não faz distinção entre sinal e ruído, e também ajusta as exponenciais ao ruído presente. Seguindo o proposto em [3], temos o seguinte para um sinal $y[k]$ imaginando M exponenciais para se ajustarem a 2M amostras, respeitando o teorema da amostragem:

$$y[k] = \sum_{m=1}^{M} A_m e^{(\alpha_m + j2\pi f_m)(k-1)\Delta t + j\sigma_m} \ , \ k = 1, 2, ..., 2M \tag{21}$$

Onde fm é a frequência das exponenciais, Δt é o tempo de amostragem, α_m é o coeficiente de amortecimento, e σ_m é o ângulo de defasagem. Para o método proposto, estamos apenas interessados em saber quais são as frequências presentes no sinal. Se imaginamos nosso sistema sendo auto regressivo (AR) de ordem M, temos que em qualquer instante de k é possível prever o valor $y[k]$ considerando apenas as M amostras anteriores deste mesmo sinal. Podemos então montar uma equação de diferenças, que neste caso também é um modelo de predição linear[4]:

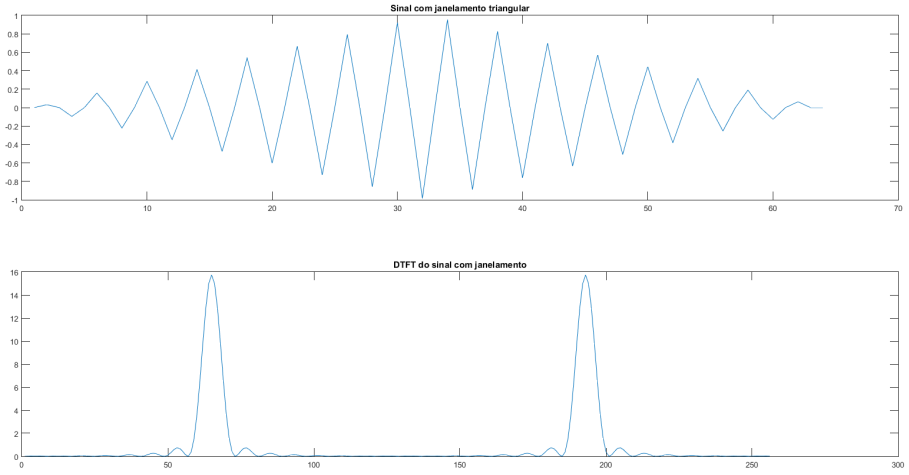


Figura 3: legenda aqui

$$y[k] = \sum_{m=1}^{M} a_m y[k-m] \tag{22}$$

É sabido que exponenciais complexas são soluções para tal modelo. Se tivermos em mãos os valores dos coeficientes a podemos montar o polinômio característico da equação e as raízes deste polinômio (exponenciais complexas) são soluções para nosso problema. De posse dos coeficientes a devemos então encontrar as raízes do polinômio seguinte:

$$P(z) = z^M - a_1 z^{M-1} - ... - a_M z^0 \tag{23}$$

Como os coeficientes a são reais, as raízes complexas estão em pares complexos conjugados, desta maneira cada par complexo representa uma possível senóide já que $cos(wt) = \frac{e^{jwt}+e^{-jwt}}{2}$:

$$f_m = atan(Im\{z_m\}/Re\{z_m\})2\pi fs \tag{24}$$

Onde ${}_zm$ é uma das raízes e fs é a frequência de amostragem. Reparemos que as frequências f_m são frequências de possíveis soluções do sistema, não necessariamente elas estarão presentes. Qualquer combinação linear dessas senióides também é solução do sistema AR planteado. Para encontrar a forma da solução real, é necessário fazer uso das condições iniciais conhecidas de nosso sistema. No caso, as próprias amostras $y[k]$ as quais temos acesso.

Os coeficientes de amortecimento α_m são o módulo das raízes complexas que encontramos. E se alguma raíz não é complexa, isto somente significa que uma exponencial é solução para nossa equação.

$$\alpha_m = |zm| \tag{25}$$

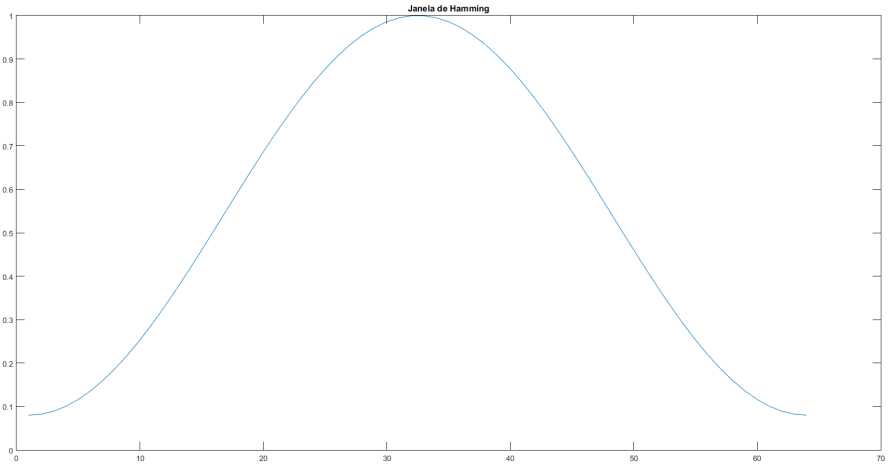


Figura 4: legenda aqui

0.4.1 Solução do modelo de predição linear

Existem diversas formas de solucionar um modelo deste tipo, apresentaremos algumas opções abaixo[9]:

Sistema linear

Conhecendo os valores $y[k]$ que são amostras de nosso sistema analisado, podemos montar um sistema linear considerando a equação anterior:

$$y[k] = \sum_{m=1}^{M} a_m y[k-m] \begin{bmatrix} y[k-1] & y[k-2] & \dots & y[k-M] \\ y[k-2] & y[k-3] & \dots & y[k-M-1] \\ \vdots & & & \vdots \\ y[k-M] & y[k-M-1] & \dots & y[k-2M+1] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix} = \begin{bmatrix} y[k] \\ y[k-1] \\ \vdots \\ y[k-M+1] \end{bmatrix} \tag{26}$$

Ou de forma simplificada:

$$\boldsymbol{Y_k a} = \boldsymbol{y_k} \tag{27}$$

A forma mais simples de resolver este sistema é inverter a matriz $\boldsymbol{Y_k}$:

$$\boldsymbol{a} = \boldsymbol{Y_k}^{-1} \boldsymbol{y_k} \tag{28}$$

Um dos problemas com esta solução é que ela é computacionalmente custosa, já que não está implementada de maneira recursiva, mas pior que isso é o fato de que normalmente temos ruído em nosso sistema, e desta maneira estamos ajustando os valores de $\boldsymbol{a}$ ao ruído também, o que em geral

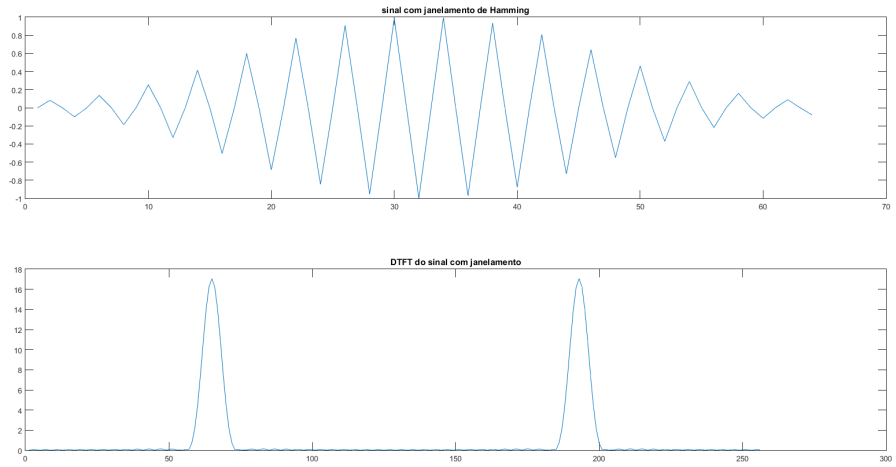


Figura 5: legenda aqui

não é o desejado, e em nosso caso, certamente não é. Desta forma, cada vez que calculamos o vetor $\boldsymbol{a}$, ele pode sair completamente diferente do anterior, nos levando a estimações equivocadas. Modelando $y[k]$ como:

$$y[k] = \sum_{m=1}^{M} a_m y[k-m] + \xi_k \tag{29}$$

Em que ξ_k é ruído branco, temos um modelo estocástico de nosso sinal, que nos possibilita pensar em soluções mais arrojadas.

0.4.2 Filtros adaptativos

O Elemento linear adaptativo, ou filtro FIR adaptativo, também chamado de ADALINE, é uma rede neural artificial composta de uma única camada e com função de ativação linear. Sendo $\boldsymbol{x}$ as entradas, $\boldsymbol{w}$ os pesos, b o valor de bias, o ADALINE pode ser implementado matricialmente da seguinte maneira:

$$y = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \end{bmatrix} * \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} + b = \boldsymbol{x} * \boldsymbol{w} + b \tag{30}$$

Filtros adaptativos são considerados sistemas não lineares, portanto a análise de seu comportamento é mais complexa que de a filtros com coeficientes fixos. Por outro lado, pelo fato de serem auto-projetados, por um

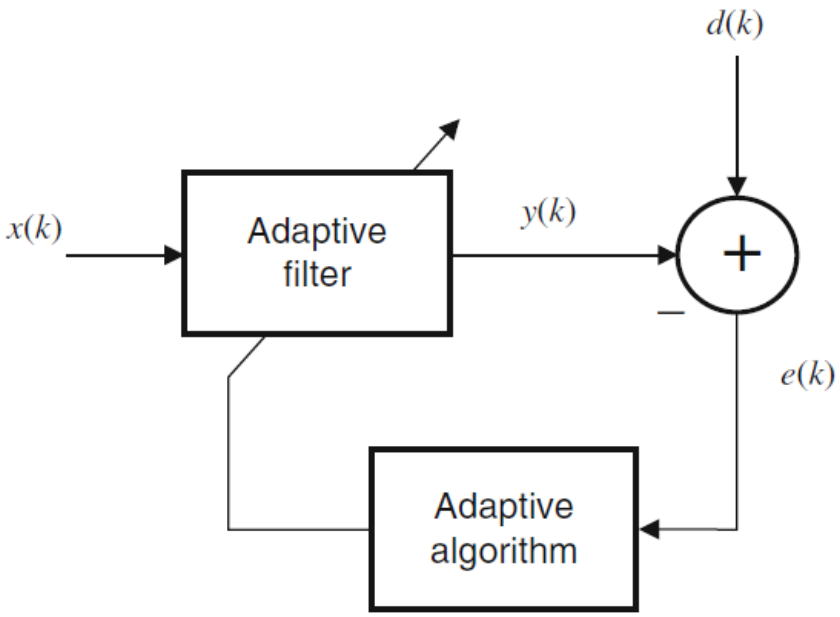


Figura 6: legenda aqui

ponto de vista mais prático, eles podem ser considerados menos complicados que os convencionais em termos de projeto[6].

O desenho usual de um filtro adaptativo pode ser visto na figura 6, onde k é o número da iteração, $x(k)$ é a entrada do filtro, $y(k)$ é o sinal de saída, $d(k)$ é o valor de referência e $e(k) = d(k) - y(k)$ é o valor de erro, necessário para o algoritmo de adaptação do filtro. O algoritmo de adaptação é o prcesso usado para ajustar os coeficientes do filtro de modo a minimizar o erro de acordo com os critétios preestabelecidos. A escolha do algoritmo determina muitos aspectos do processo de adaptação, como a existência de soluções subótimas e complexidade computacional.

0.4.3 Adaptação via gradiente descendente

Uma das formas mais antigas e consagradas de otimizar uma função dentro dos métodos clássicos é seguir a direção do gradiente desta função, com relação as variáveis de interesse, para atualizar os valores destas variáveis de forma iterativa.

Consideremos um sistema discriminado pela série temporal $u[n] n = 0, 1, 2..$

e um filtro de resposta impulsiva real $w_0, w_1 ... w_{M-1}$:

$$y[n] = \sum_{k=0}^{M-1} u[n-k]w_k \tag{31}$$

Todo este desenvolvimento pode ser feito de maneira bastante similar considerando entradas complexas e filtros com coeficientes complexos, mas por questões de simplicidade estaremos focando no caso em que ambos são reais. Imaginemos que desejamos que a saída deste filtro seja uma outra série temporal igualmente real $d[n]$, assim nosso erro seria:

$$e[n] = d[n] - y[n] \tag{32}$$

Considerando o caso de u estocástico, $e[n]$ é uma variável aleatória. Gostaríamos então de minimizar a esperança de $e^2[n]$, definimos então nossa função de custo:

$$J = E[e^2[n]] = E[(d[n] - y[n])^2] \tag{33}$$

Definindo nosso operador gradiente ∇_k:

$$\nabla_k J = -2E[e[n]u[n-k]] \tag{34}$$

Se queremos encontrar o ótimo com relação a esta função de custo, $\nabla_k J$ deve ser igual a zero para todo k, como a esperança do produto de duas variáveis aleatórias é sua correlação, $u[n-k]$ e $e[n]$ devem estar descorrelacionadas para todo k. A este resultados chamamos princípio da ortogonalidade. Um corolário deste princípio é que y também deve ser ortogonal, ou descorrelacionada com $e[n]$.

É mais conveniente para nó trabalhar com a notação matricial destes resultados. Ao vetor coluna que contém as variáveis aleatórias $u[n-k]$, de k=0 a K=M-1, lhe chamamos $\boldsymbol{u}_n$, ao vetor de coeficientes w, igualmente:

$$\boldsymbol{u}_n = \begin{bmatrix} u[n] & u[n-1] & \dots & u[n-M+1] \end{bmatrix}^T$$

$$\boldsymbol{W}_n = \begin{bmatrix} w_0 & w_1 & \dots & w_{M-1} \end{bmatrix}^T$$

Desenvolvendo nossa equação anterior, podemos obter:

$$\nabla J = -2E[e[n]\boldsymbol{u}_n] = 2E[\boldsymbol{u}_n \boldsymbol{U}_n^T \boldsymbol{W} - d[n]\boldsymbol{u}_n] \tag{35}$$

Desta última equação, destacamos que $E[d[n]\boldsymbol{U}_n]$ é o vetor de correlação cruzada entre $d[n]$, que chamaremos de $\boldsymbol{r_{du}}$ e as amostras atrasadas do sinal de entrada.Destacamos também que $E[\boldsymbol{U}_n \boldsymbol{U}_n^T]$, a qual chamaremos $\boldsymbol{R_{uu}}$, é

a matriz MxM de autocorrelação do mesmo sinal. Sendo assim, escrevemos agora:

$$\boldsymbol{R}_{uu}\boldsymbol{w} - \boldsymbol{r}_{du} = 0 \tag{36}$$

De onde podemos concluir que o vetor coeficientes ótimos $\boldsymbol{w}_{opt}$ é:

$$\boldsymbol{w}_{opt} = \boldsymbol{R}_{uu}^{-1}\boldsymbol{r}_{du} \tag{37}$$

Se u é um processo estacionário em sentido amplo, então podemos coletar amostras suficientes para fazer uma boa estimação das matrizes acima, encontrando assim um filtro muito mais apropriado para nossos propósitos. Desta maneira, é possível estimar o conteúdo espectral de u com precisão. Reparemos que este método não é propriamente um gradiente escendente já que encontramos a solução em um passo apenas.

A solução acima tem sua utilidade, entretanto, se a natureza do sinal mudar, não teremos mais uma boa aproximação do mesmo, neste caso, é melhor resolver estas equações de forma recursiva, de modo que o filtro possa se adaptar constantemente à mudanças ocorridas.

0.4.4 O algoritmo LMS

Para atualizar o vetor de pesos $\boldsymbol{w}_n$ na direção oposta a do gradiente, definimos um coeficiente de aprendizagem μ. A cada iteração vamos fazer o seguinte:

$$\boldsymbol{w}_n = \boldsymbol{w}_{n-1} - \mu\nabla J$$

$$\boldsymbol{w}_n = \boldsymbol{w}_{n-1} + \mu(\boldsymbol{r}_{du} - \boldsymbol{R}_{uu}\boldsymbol{w}_{n+1}) \tag{38}$$

O que diferencia o LMS de outros métodos de gradiente descendente é a forma de estimar as matrizes $\boldsymbol{R}_{uu}$ e $\boldsymbol{r}_{du}$. Vamos estimâ-las da seguinte forma:

$$\hat{\boldsymbol{R}}_{uu} = \boldsymbol{u}_n\boldsymbol{u}_n^T \tag{39}$$

$$\hat{\boldsymbol{r}}_{du} = \boldsymbol{u}_n d[n] \tag{40}$$

O que pode soar uma heresia, mas faz sentido se pensarmos que apenas temos M amostras de sinal.

0.4.5 O algoritmo NLMS

Uma melhoria no algoritmo anterior pode ser proposta. Se multiplicamos a parte que está entre parênteses na equação XX:

$$\boldsymbol{w}_n = \boldsymbol{w}_{n-1} + \mu \boldsymbol{R}_{uu}^{-1}(\boldsymbol{r}_{du} - \boldsymbol{R}_{uu}\boldsymbol{w}_{n+1}) = \boldsymbol{w}_{n-1} + \mu(\boldsymbol{w}_{opt} - I\boldsymbol{w}_{n-1}) = \boldsymbol{w}_{opt} \quad (41)$$

Em teoria, nosso algoritmo convergiria em apenas uma iteração. Resta apenas o empecilho de que estimar tal matriz. Com estimador que tínhamos antes isto não é possível, porque a matriz estimada $\hat{\boldsymbol{R}}_{uu}$ não é inversível por ser o produto de dois vetores. Mas podemos fazer uma aproximação da forma $\hat{\boldsymbol{R}}_{uu} + \boldsymbol{I}\epsilon$. Que certamente é inversível e pode nos dar uma inversa útil, para valores pequenos de ϵ. E após alguma álgebra[10], que pode ser encontrada em [9], chegamos ao seguinte:

$$\boldsymbol{w}_n = \boldsymbol{w}_{n-1} + \mu \frac{\boldsymbol{u}_n}{|\boldsymbol{u}_n|^2 + \epsilon}(d[n] - \boldsymbol{u}_n \boldsymbol{w}_{n-1}) = \boldsymbol{w}_{n-1} + \mu \frac{\boldsymbol{u}_n}{|\boldsymbol{u}_n|^2 + \epsilon} e[n] \quad (42)$$

A este algoritmo chamamos LMS normalizado, ou NLMS. Há ainda o estudo de convergência para valores pequenos de μ, que também pode ser encontrado em [6][9], tanto para o LMS quanto para o NLMS. Mas de modo geral o NLMS deve convergir respeirada todas as condições estabelecidas anteriormente e escolhendo valores de $\mu < 1$.

0.5 RLS

O método RLS (*Recursive Least Squares*), ou mínimos quadrados recursivo, é um método de adaptação que visa a minimização da soma dos quadrados da diferença entre o sinal de referência e o sinal de saída do filtro em questão (o erro). O RLS pode ser obtido apartir do LMS (*Least Mean Squares*). O RLS é conhecido por possuir rápida convergência, tendo boa performance em sistemas variantes no tempo, como o caso de nosso interesse. Isto vem ao custo de certa complexidade computacional aliada a problemas de estabilidade[6].

Voltando ao modelo do filtro adaptativo da seção anterior, vamos definir $\boldsymbol{x}(k)$ como o vetor contendo N amostras atrasadas de nosso sinal de entrada, ou seja $x(k)$. Assim:

$$\boldsymbol{x}(k) = \begin{bmatrix} x(k-1) & x(k-2) & \dots & x(k-N) \end{bmatrix} \quad (43)$$

Sendo N a ordem do filtro. Da mesma maneira, podemos definir $\boldsymbol{w}(k)$ como o vetor contendo os coeficientes do filtro na k ésima iteração. Assim:

$$\boldsymbol{w}(k) = \begin{bmatrix} w_1(k) \\ w_2(k) \\ \vdots \\ w_N(k) \end{bmatrix} \tag{44}$$

A função objetivo para o RLS é determinística e dada por:

$$\xi(k) = \sum_{i=0}^{k} \lambda^{k-i} \varepsilon^2(i) = \sum_{i=0}^{k} \lambda^{k-i} [d(i) - \boldsymbol{x}(k) * \boldsymbol{w}(k)] \tag{45}$$

Onde ε é o erro a posteriori no instante i. O parâmetro λ é um fator de peso exponencial que deve ser escolhido entre 0 e 1, de modo que $0 << \lambda < 1$. λ representa o fator de esquecimento do erro, quanto menor seu valor, menor será a influência das amostras passadas sobre a atualização atual.

Após alguma álgebra que pode ser encntrada em [6], temos o seguinte:

$$\boldsymbol{w}(k) = \boldsymbol{S}_d(k) * \boldsymbol{p}_d(k) \tag{46}$$

$$\boldsymbol{S}_d = \lambda^{-1} \left[\boldsymbol{S}_d(k-1) - \frac{\boldsymbol{S}_d(k-1)\boldsymbol{x}^T(k)\boldsymbol{x}(k)\boldsymbol{S}_d}{\lambda + \boldsymbol{x}(k)\boldsymbol{S}_d(k)\boldsymbol{x}^T(k)} \right] \tag{47}$$

$$\boldsymbol{p}_d(k) = \lambda \boldsymbol{p}_d(k-1) + d(k) * \boldsymbol{x}(k) \tag{48}$$

Boas recomendações de inicialização de $\boldsymbol{S}_d$ e $\boldsymbol{p}_d$ são $\boldsymbol{S}_d = \delta \boldsymbol{I}$ e $\boldsymbol{p}_d = [0\ 0\ ...\ 0]^T$. Nas quais δ pode ser o inverso da potência estimada do sinal.

0.6 PLL

Um Phase Locked Loop digital, assim como sua versão analógica, visa determinar os parâmetros de um processo estocáscico como uma onda senoidal. Desta forma o PLL tenta extrair parâmetros de Amplitude, fase e

frequência. Temos diversas variantes digitais e analógicas[7], a variante utilizada neste trabalho foi proposta por Ziarani em 2004[8]. Uma breve demostração será realizada abaixo. Seja $u(t)$ uma função variante no tempo, e $y(t) = Asin\phi(t)$ um sinal periódico senoidal sendo A a amplitude do sinal e $\phi(t)$ sua fase, tendo este sinal uma frequência constante, podemos escrever $\phi(t) = wt + \delta$, sendo w igual a frequência e δ um valor de fase constante. Podemos escrever de maneira mais geral:

$$u(t) = \sum_{i=0}^{\infty} A_i sin\phi_i(t) + n(t) \tag{49}$$

Onde $n(t)$ representa um distúrbio, como um ruído. Na realidade, todos os parâmetros podem variar com o tempo, é conveniente escrever então um conjunto $\Psi(t) = [A(t), w(t), \delta(t)]$ o qual contém todos os parâmetros de um possível sinal $y(t)$.

Definimos a função d:

$$d^2(t, \Psi(t)) = [u(t) - y(t, \Psi(t))]^2 = e^2(t) \tag{50}$$

Assim sendo o conjunto $\Psi(t)$ ótimo o que minimiza a função d^2 que adotaremos como função de custo $J(\Psi(t), t)$. Os parâmetros $\Psi(t)$ podem ser estimados por meio de gradiente descendente.

$$\frac{d\Psi(t)}{dt} = -\boldsymbol{\mu}\frac{\partial J(\Psi(t), t)}{\partial \Psi(t)} \tag{51}$$

$$\boldsymbol{\mu} = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \tag{52}$$

Com $\frac{d\Psi(t)}{dt}$ denotando o vetor na dierção do qual são atualizados os valores de $\Psi(t)$ a cada iteração. E $\boldsymbol{\mu}$ a matriz diagonal contendo as constantes de atualização referentes a cada parâmetro. Como estaremos lidando com estimações agora, usaremos a notação $\hat{\Psi}(t)$:

$$\begin{bmatrix} \frac{d\hat{A}(t)}{dt} \\ \frac{d\hat{w}(t)}{dt} \\ \frac{d\hat{\delta}(t)}{dt} \end{bmatrix} = -\boldsymbol{\mu} \begin{bmatrix} \frac{\partial e^2(t)}{\partial \hat{A}} \\ \frac{\partial e^2(t)}{\partial \hat{w}} \\ \frac{\partial e^2(t)}{\partial \hat{\delta}} \end{bmatrix} \tag{53}$$

Podemos obter então o seguinte:

$$\frac{d\hat{A}(t)}{dt} = 2m_1 e(t) sin(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{54}$$

$$\frac{d\hat{w}(t)}{dt} = 2m_2 e(t)\hat{A}(t)tcos(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{55}$$

$$\frac{d\hat{\delta}(t)}{dt} = 2m_3 e(t)\hat{A}(t)cos(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{56}$$

$$\frac{d\hat{\phi}(t)}{dt} = \hat{w}(t) + \frac{\hat{\delta}(t)}{dt} \tag{57}$$

Por conta do fator temporal t que aparece solto na equação referente a $\hat{w}$, esse sistema é variante no tempo, o que o torna instável. No entanto uma solução heurística é substituir t por uma constante $m4$, que pode ser absorvida pela constante $m2$, tornando o sistema invariante no tempo e fazendo com que este sistema de equações seja útil na prática. Desta maneira obtemos o seguinte:

$$\frac{d\hat{A}(t)}{dt} = 2\mu_1 e(t)sin(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{58}$$

$$\frac{d\hat{w}(t)}{dt} = 2\mu_2 e(t)\hat{A}(t)tcos(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{59}$$

$$\frac{d\hat{\phi}(t)}{dt} = \hat{w}(t) + 2\mu_3 e(t)\hat{A}(t)cos(\int_0^t \hat{w}(\tau)d\tau + \hat{\delta}(t)) \tag{60}$$

Usando o método *Euler Forward*:

$$A[n+1] = A[n] + \mu_1 T_s e[n]sen(\phi(t)) \tag{61}$$

$$w[n+1] = w[n] + \mu_2 T_s e[n]cos(\phi(t)) \tag{62}$$

$$\phi[n+1] = \phi[n] + T_s w[n] + \mu_3 T_s e[n]cos(\phi(t)) \tag{63}$$

Desta forma obtemos o conjunto de equações discretas que utilizaremos para rastrear frequências no restante do trabalho.

0.7 Processamento Multitaxa

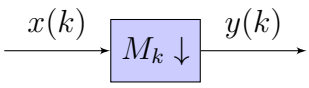
Em determinadas situações, é desejável alterar a taxa de amostragem de um sinal para realizar certos tipos de processamento. Por exemplo, veremos em alguns resultados nas próximas sessões que o algoritmo PLL descrito anteriormente é sensível a taxa de amostragem em relação a frequência rastreada, portanto é conveniente mantê-la mais ou menos constante. Também veremos que processar sinais na taxa necessária para abranger todo o espectro que se quer analisar é extremamente custoso computacionalmente e não é garantia

de melhores resultados. Por isso deve-se considerar o uso de uma estrutura multitaxa que diminua a frequência base.

Analisaremos a seguir as implicações de um abaixamento na frequência de amostragem:

0.7.1 Downsampling

Uma estrutura downsampling tem a forma seguinte:



Representamos a estrutura como um sistema que reproduz em sua saída apenas amostras onde k é múltiplo de M_k. Podemos escrever $y[n] = x[nM_k]$ $n = 1, 2, 3...$ Considerando que $x[k]$ foi amostrado com a frequência de Nyquist fs, o efeito final é o mesmo que se tivéssemos amostrado x com fs/M_k, podemos dizer que vamos observar aliasing em nosso resultado final. Haverá portanto sobreposição no espectro de frequência referente a $X(w)$ quando analisarmos $Y(w)$. Entretanto, é possível prever onde estarão localizadas as componentes espectrais presentes no sinal completo que estamos subamostrando, e embora não possamos saber exatamente qual é o valor correspondente as frequências originais, podemos saber no mínimo qual é sua soma.

Uma demonstração formal dos efeitos observados pode ser encontrada em[5], aqui faremos uma abordagem mais intuitiva. Imaginemos uma componente espectral de $x[k]$ referente a w_0. Imagine que fazemos uma subamostragem neste sinal, e agora sua frequência de amostragem que era f_s se torna f_s/M_k. Agora, se $w_0 < f_s/(2Mk)$ essa as componentes dessa frequência não saem do lugar, elas continuam onde estavam anteriormente. Isso não significa que seu espectro não seja alterado nas frequências mais baixas que a nova taxa de amostragem, somente quer dizer que elas sofrem interferência no mesmo lugar onde estavam antes. Agora, se a frequência de interesse é maior que a nova frequência de amostragem dividida por 2, então ela não poderá de forma alguma ser encontrada no mesmo local, pois simplesmente não há como visualizá-la com DFT ou DTFT. Mas então onde vão parar essas frequências? Bom, onde pararia qualquer uma que sofre aliasing. Ela dá voltas no círculo de frequências. O circulo de frequências sempre vai de 0 a 2π, e as frequências sempre caminham nele no sentido anti-horário. Para cada vez que a frequência de interesse (a qual chamaremos f_i) for maior que fs, a frequência em questão dá uma volta no círculo. Um algoritmo simples será apresentado quando expusermos o método do PLL multitaxa, mas por

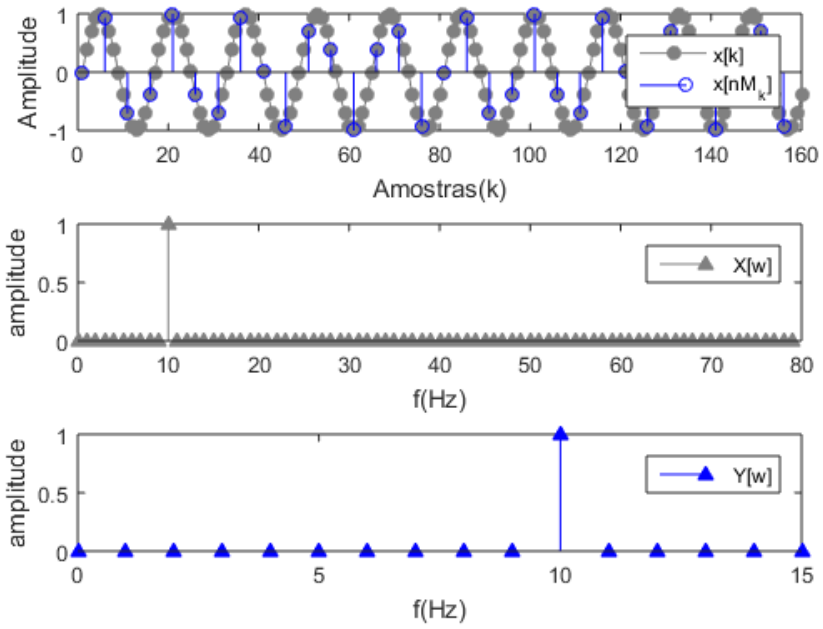


Figura 7: Downsampling para $f_s = 160Hz$, $f_0 = 10Hz$, $M_k = 5$

hora podemos saber que o ângulo referente à nova frequência será o resto da divisão f_i/f_s e multiplicamos por 2π.

No exemplo da figura XX, temos $fs = 160Hz$ e $M_k = 5$, nossa frequência de interesse é 10Hz, a frequência de amostragem depois da subamostragem seria ainda maior que $2f_i$, então ela não se move, como observado na figura. Entretanto, no caso de $M_k = 10$ a coisa muda, calculando:

$$w_i = rest(f_i\ M_k,\ f_s)2\pi \tag{64}$$

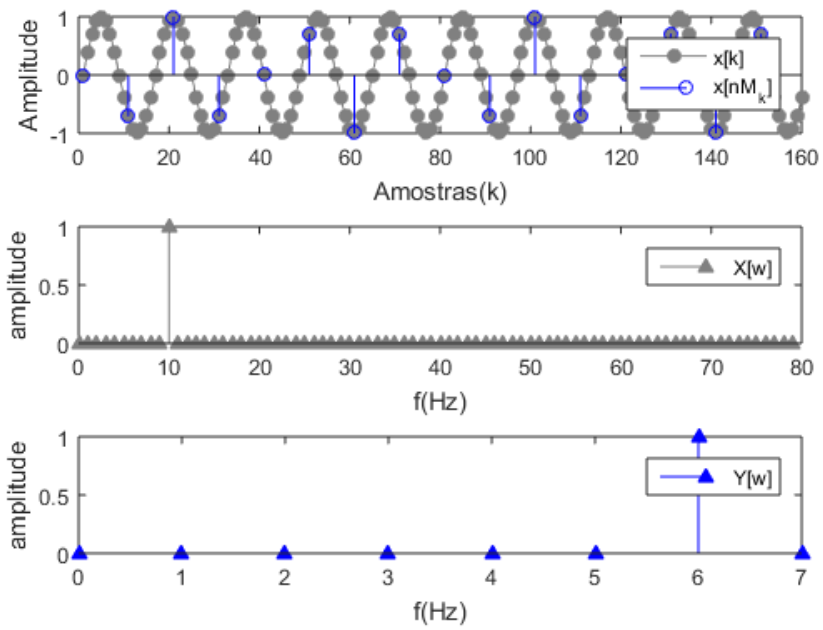


Figura 8: Downsampling para $f_s = 160Hz$, $f_0 = 10Hz$, $M_k = 10$