Ciência e Tecnologia da Fala 2024/1

➢ Estudo Dirigido

- Arthur de Oliveira Lima - 2019027318
- Melchior Augusto Syrio de Melo - 2024686065

I. Introdução

Este estudo dirigido tem como objetivo a resolução de problemas básicos do processamento de sinais de fala. Os exercícios e projetos a seguir têm como base os arquivos e funções de áudio do professor James H. McClellan.
Os exercícios foram elaborados utilizando-se MATLAB 2024a e podem ser acessados através do link abaixo:

https://github.com/Artlima1/speech-processing-study

II. Introduction to Speech Processing

A. Project 1 - Speech Segmentation

Exercício 1.1 - *Phonetic Representation of text*

A representação fonética da frase “Oak is strong and also gives shade” segue a seguir:

OW K IH S S T R EN H AX D AO S OW G IH V S SH EY D

Exercício 1.2 - Phonetic Labeling Using Waveform Plots

B. Project 2 - Preemphasis of Speech

Este projeto foi desenvolvido no arquivo “introduction-to-speech-processing/*preemphasis_of_speech.m*”

Exercício 2.1 - Preliminary Analysis

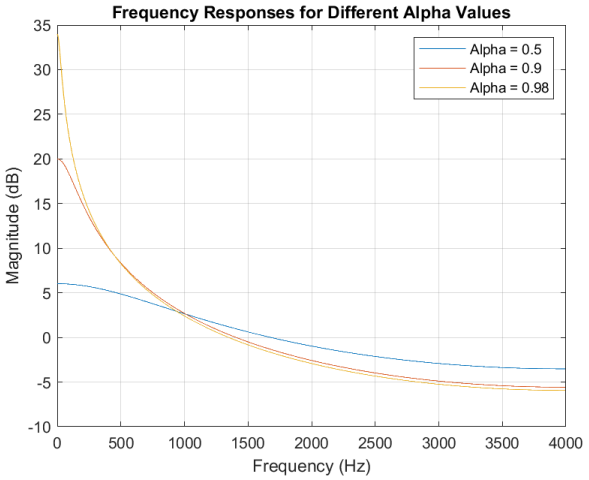
Sabendo que:

$$H(z) = 1 / (1 - alpha*z^{-1})$$

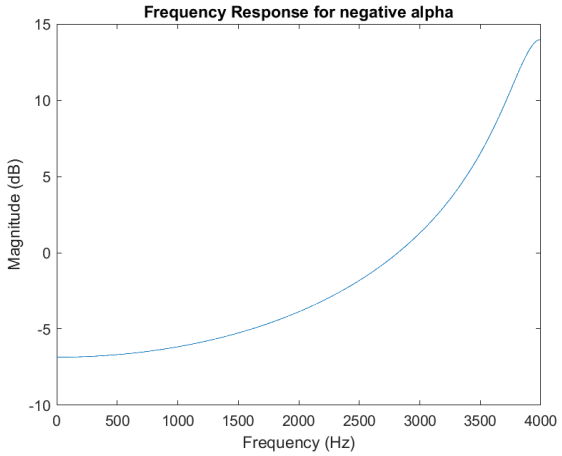
Pode-se determinar analiticamente que:

$$h[n] = alpha^n * u[n]$$

O gráfico abaixo evidencia a resposta em frequência do sistema para diferentes valores de α:

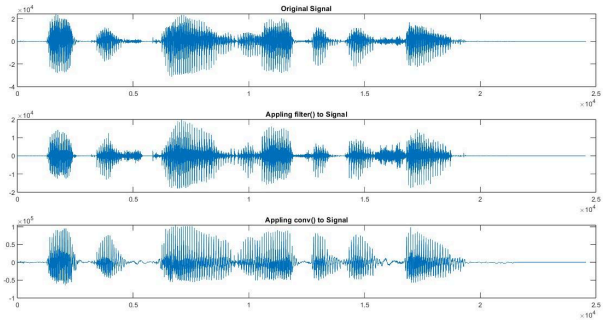


Além disso, para que a resposta evidencia as frequências baixas, pode-se escolher um valor negativo para alpha, como mostra o gráfico abaixo:

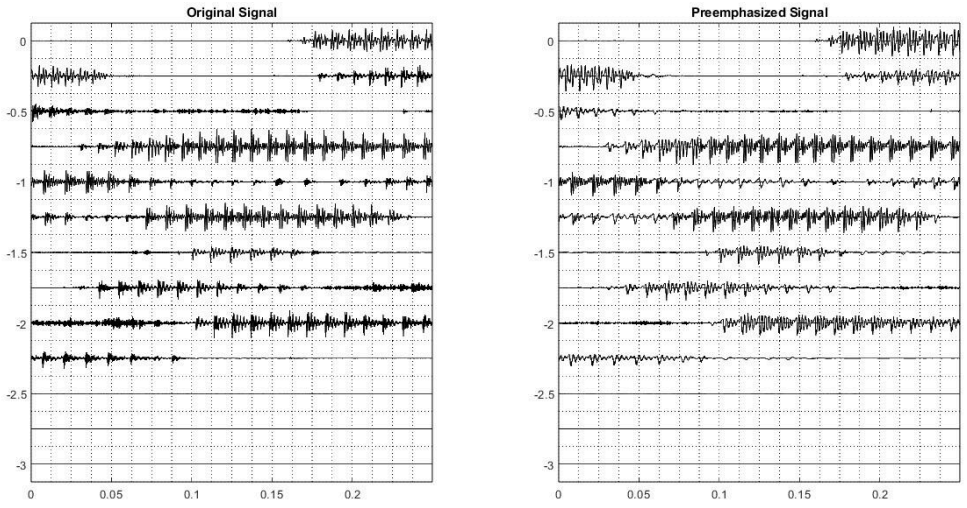


Exercício 2.2 - MATLAB Implementation

Abaixo, é possível visualizar o efeito de aplicação de um filtro de pré-ênfase no sinal usando as funções *filter()* e *conv()*.



Exercício 2.3 - Plotting the Premphasized Signal

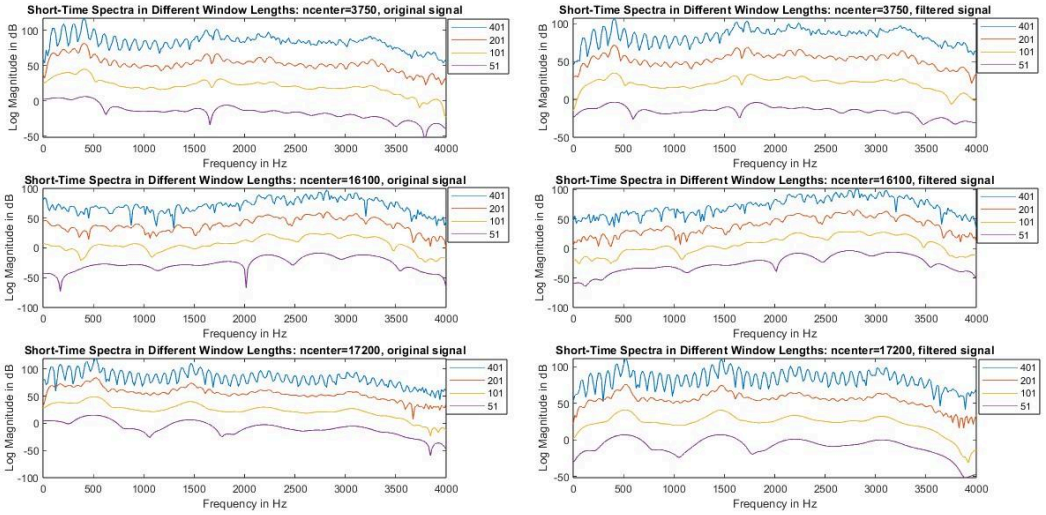


C. Short-Time Fourier Analysis

Este projeto foi desenvolvido no arquivo "introduction-to-speech-processing/*short_time_fourier_transform.m*"

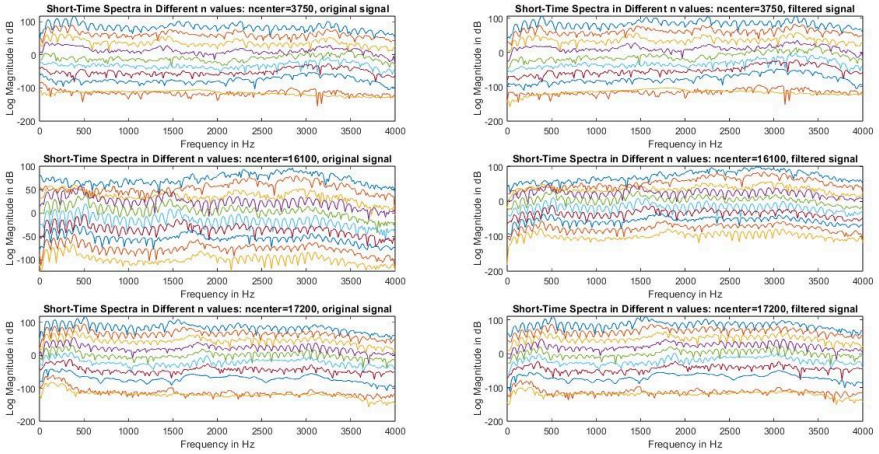
Exercício 3.1 - Effect of Window Length

É possível perceber que, à medida em que se aumenta o tamanho da janela, ganha-se na resolução em frequência, ao passo que se perde resolução temporal.



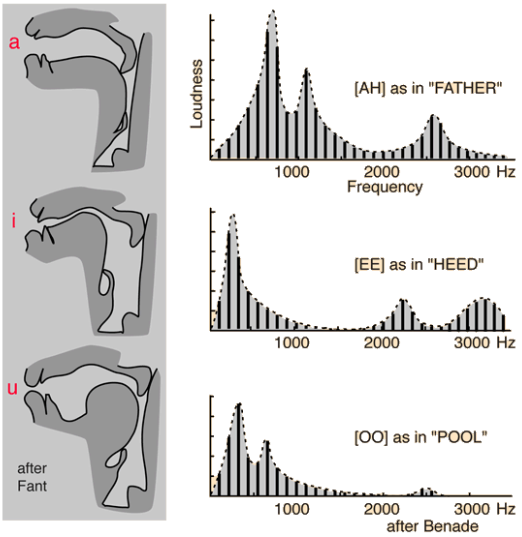
Exercício 3.2 - Effect of Window Position

Abaixo se encontram os gráficos da fft para diferentes posições de janela.



É possível perceber nos gráficos, especialmente com ncenter=16100, a forma como o filtro evidencia altas frequências, principalmente pelo efeito de amortecimento das frequências mais baixas.

III. Speech Modeling



Fonte Georgia State University: http://hyperphysics.phy-astr.gsu.edu/hbase/Music/vowel.html

A. Project 1 - Glottal Pulse Models

Este projeto foi desenvolvido nos arquivos "*speech-modeling/glottalE.m*", "*speech-modeling/glottalR.m*", "*speech-modeling/glottal_pulse_models.m*" e "*speech-modeling/rPulsesCases.m*"

Exercício 1.1 - Exponential Model

Aplicando a transformada inversa em

$$G(z) = (-a*e*\ln(a)*z^{-1}) / (1-az^{-1})^{-2}$$

É possível obter

$$g[n] = -e*\ln(a)*n*a^{n}*u[n]$$

Então, foi desenvolvido no arquivo "*speech-modeling/glottalE.m*" uma função que retorna o um N samples do pulso exponencial e sua respectiva resposta em frequência.

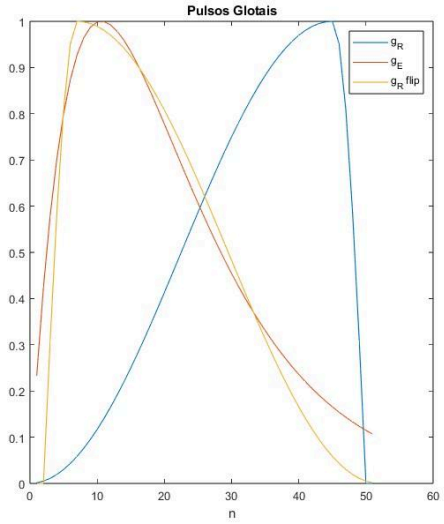
Exercício 1.2 - Rosenberg Model

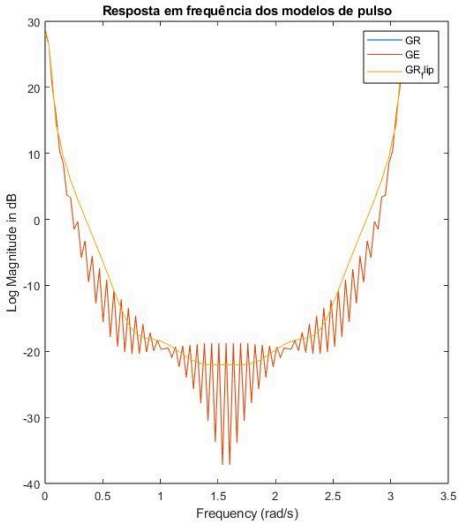
Então, foi desenvolvido no arquivo "*speech-modeling/glottalR.m*" uma função que retorna o um N samples do pulso glotal do modelo de Rosenberg e sua respectiva resposta em frequência.

Exercício 1.3 - Comparison of Glottal Pulse Models

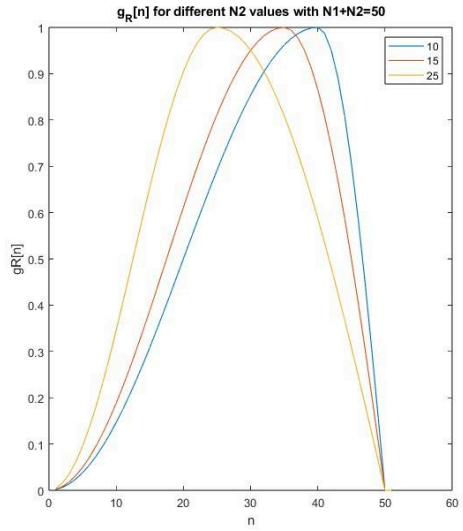
Este exercício usa dos arquivos anteriores, mas foi executado no arquivo "*speech-modeling/glottal_pulse_models.m".*

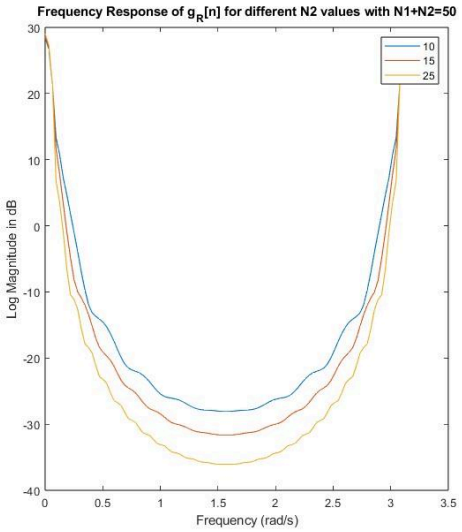
Em primeiro lugar, compara-se a os pulsos glotais no domínio do tempo e no domínio da frequência:



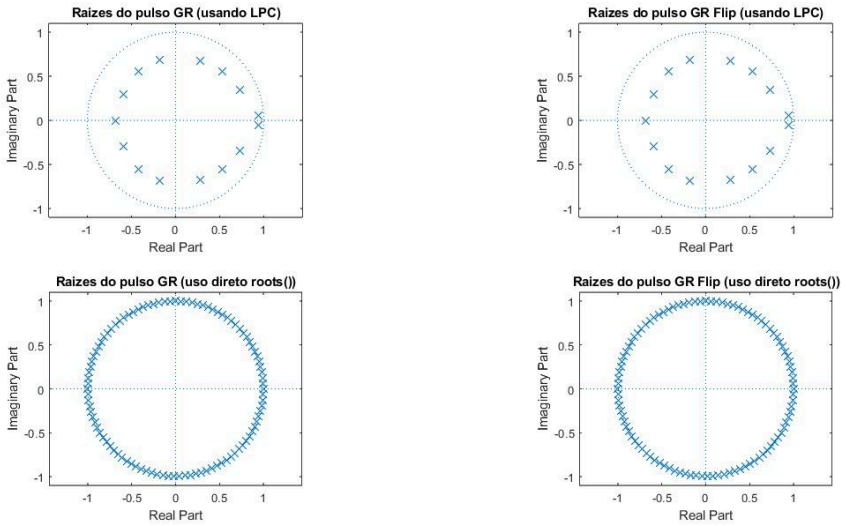


Interessante enxergar que, embora haja diferença no tempo, ao se "flipar" o pulso glotal no modelo de Rosenberg, o módulo de sua resposta em frequência é o mesmo. Isso acontece por serem um par de fase-máxima e fase-mínima.
Então, faz-se uma comparação do efeito da mudança do N2 no modelo de Rosenberg:





Ele altera a atenuação em frequência (tanto a atenuação máxima quanto o "slope").
Por último, deveria ser realizado um plot das raízes no plano Z. Deveria ser usada a função *roots()*, mas ela exige como argumento um array de coeficientes. Nosso modelo de Rosenberg não fornece os coeficientes de sua resposta em frequência, mas sua FFT diretamente. Por conta disso, não conseguimos obter as raízes para o plot no plano Z. Tentamos usar o LPC para obter os coeficientes, mas mesmo assim o resultado do plano Z foi diferente do esperado. Ainda assim, exibimos abaixo os plots para o *roots()* diretamente em GR e usando o LPC.

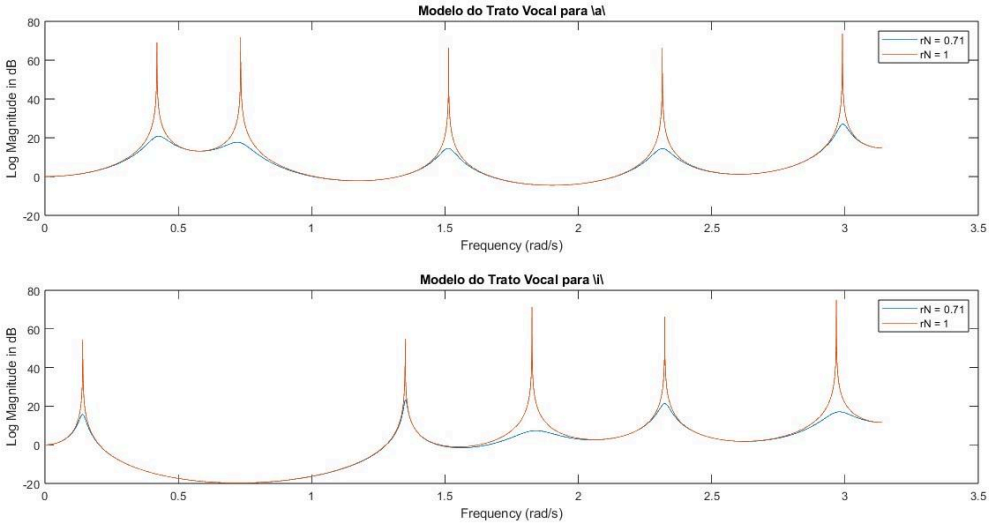


B. Project 2 - Lossless Tube Vocal Tract Models

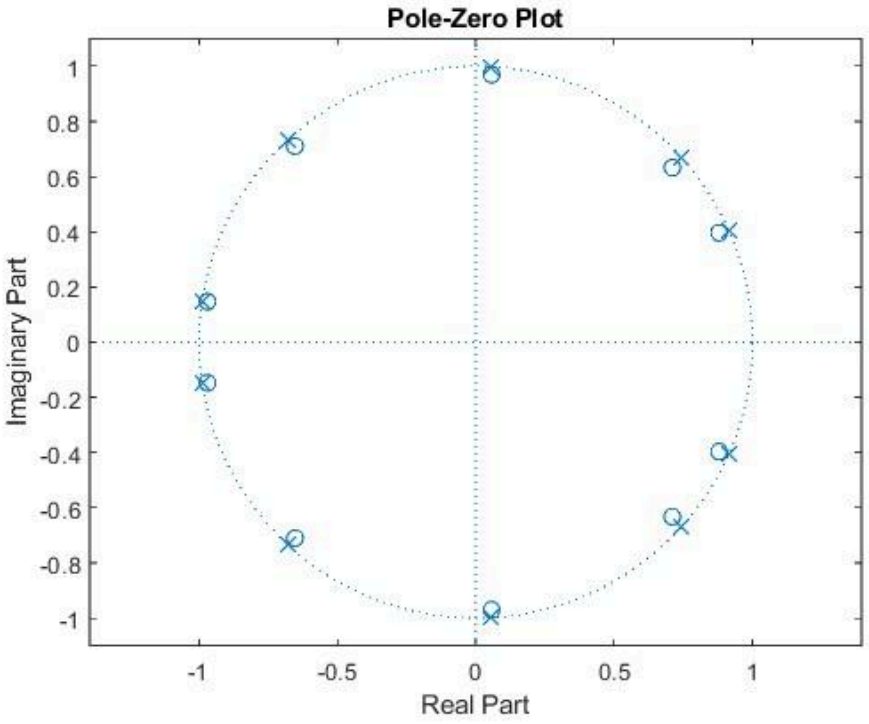
Este projeto foi desenvolvido nos arquivos "*speech-modeling/lossless_tube_vocal_tract_model.m*" e "*speech-modeling/VtoA.m*".

Exercício 2.1 - Frequency Response and Pole-Zero Plot

Utilizando as áreas fornecidas para os fonemas \a\ e \i\, foi possível elaborar o modelo do trato vocal usando a função *atov()*, obtendo-se as seguintes respostas em frequência para diferentes valores de r_N:



Em seguida, exibe-se o diagrama das raízes para o modelo construído para o fonema \a\, com os X's representando o modelo sem perdas e os O's o modelo com perdas.



É possível perceber que, à medida em que se aumentam as perdas, menor o módulo dos polos (mais próximos ao centro do plano Z).

Exercício 2.2 - Finding Model from the System Function

Então, criou-se a função VtoA no arquivo "*speech-modeling/VtoA.m*" para se obter os coeficientes r_k e as áreas para um dado D(z). Usou-se a função para o D(z) fornecido do fonema \a\, obtendo o seguinte resultado:

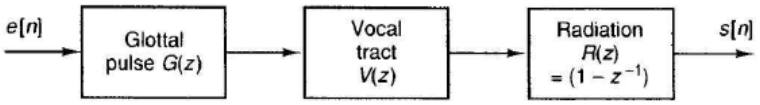
k	1	2	3	4	5	6	7	8	9	10
r_k	0,2381	-0,600	0,4222	0,2381	0,2122	0,2380	0,1035	-0,0667	-0,1666	0,7100
A_k	1,6000	2,6002	0,6500	1,600	2,6001	4,0007	6,5002	8,0013	7,0008	5,0008

C. Project 3 - Vowel Synthesis

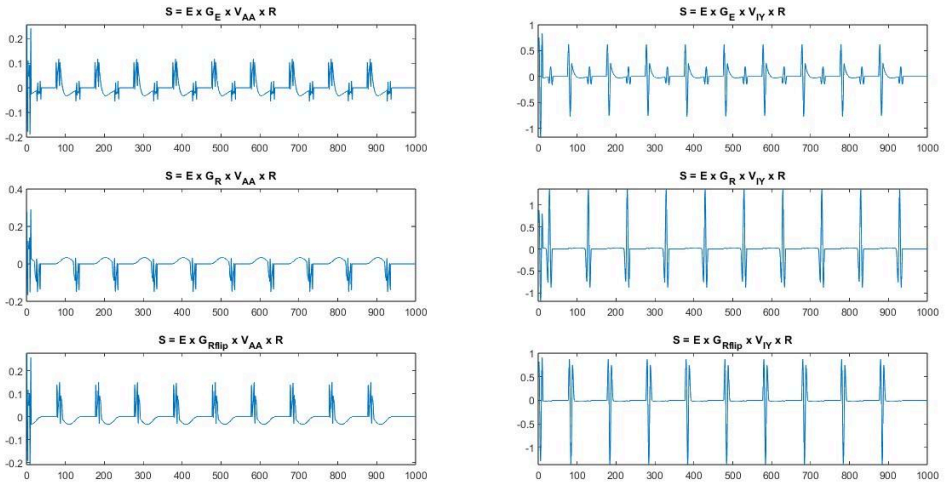
Este projeto foi desenvolvido no arquivo "*speech-modeling/vowel_synthesis.m*".

Exercício 3.1 - Periodic Vowel Synthesis

Usando os módulos desenvolvidos nos últimos exercícios, foi realizada a síntese das vogais \a\ e \i\ a partir do seguinte modelo:

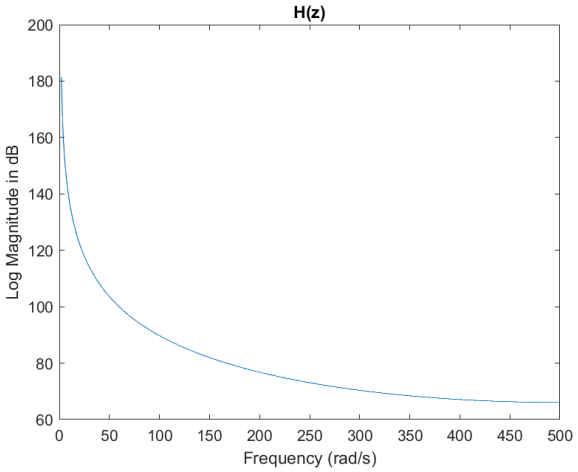


Os resultados obtidos, no domínio do tempo, estão exibidos abaixo:



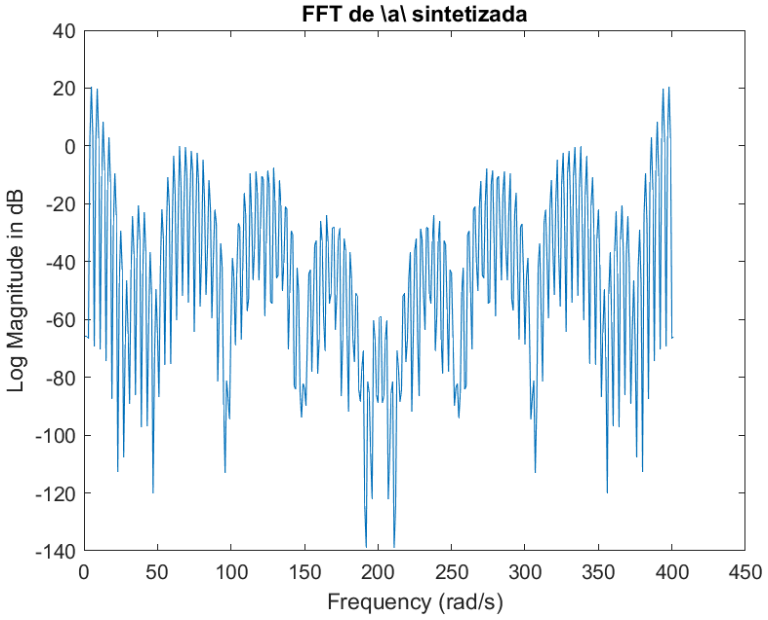
Exercício 3.2 - Frequency Response of Vowell Synthesizer

Exibe-se abaixo a resposta em frequência do sistema H(z):



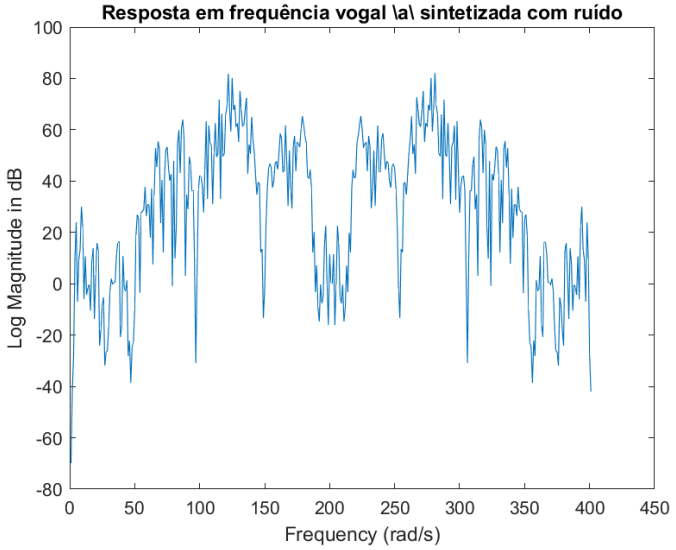
Exercício 3.3 - Short Time Fourier Transform of Synthetic Vowel

Exibe-se abaixo a FFT da vogal \a\ sintetizada, após inserção de ruído:



Exercício 3.4 - Noise Excitation

Exibe-se abaixo a FFT da vogal \a\ sintetizada:



IV. Speech Quantization

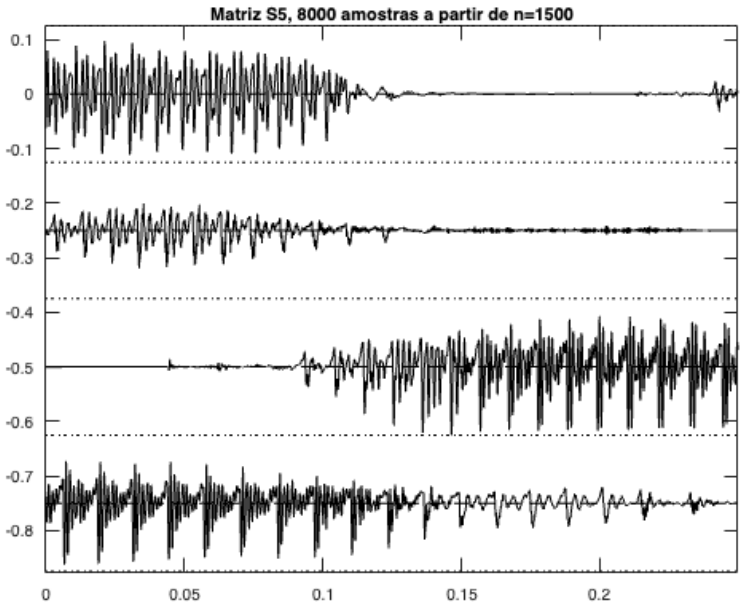
Os códigos gerados nesta seção estão na pasta ‘speech-quantization/’. Os projetos dessa sessão foram escritos em Live-Script de MATLAB.

A. Project 1 - *Speech Properties*

O arquivo de código escrito para a elaboração das figuras e resposta dos exercícios deste projeto encontram-se no caminho ‘speech-quantization/Project1.mlx’.

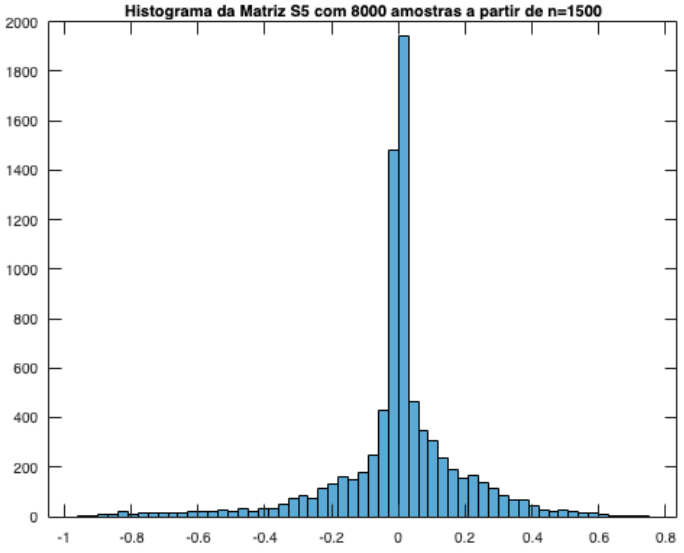
Exercício 1.1 - *Speech Waveform Plotting*

Abaixo segue a imagem utilizando a função *striplot()*, fornecida e de autoria do professor McClellan.



Exercício 1.2 - *Statistical Analysis*

Abaixo segue o histograma da matriz-sinal S5. Os valores de máxima, mínimo, média e média quadrática seguem abaixo. O resultado do histograma é consistente com o observado na forma de onda, posto que o sinal analisado é um sinal de áudio, de caráter oscilatório, limitado em potência e em banda de frequência. Os valores mais próximos ao limite de representação normalizado entre [-1,1] são menos frequentes.

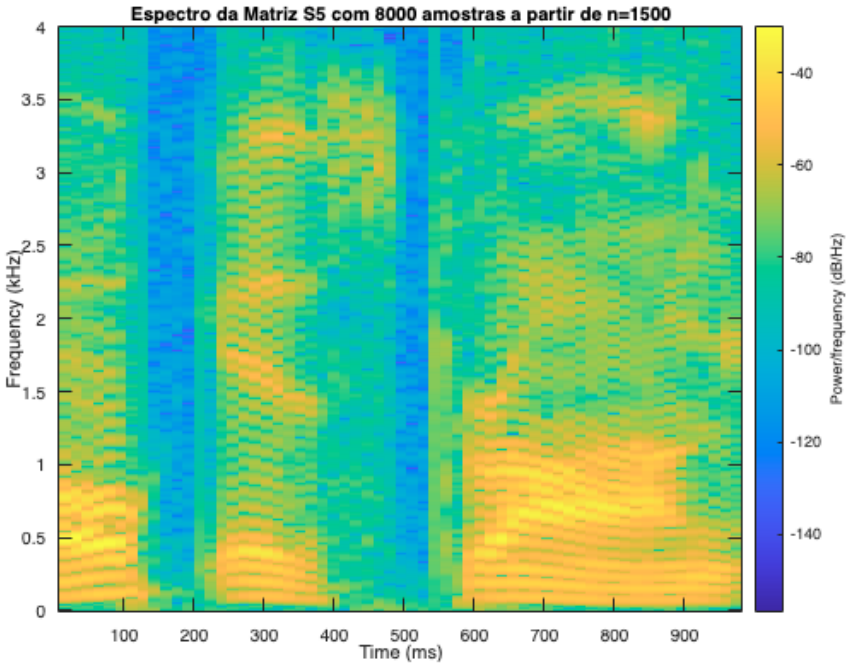



```
maxS5 = max(vetor)
maxS5 = 0.7417
minS5 = min(vetor)
minS5 = -0.9521
avgS5 = mean(vetor)
avgS5 = 2.1649e-04
meanSqrS5 = meansqr(vetor)
meanSqrS5 = 0.0375
```

Exercício 1.3 - *Spectral Analysis*

Abaixo segue o histograma da matriz-sinal S5. Os valores de máxima, mínimo, média e média quadrática seguem abaixo. Ela foi gerada com a função *spectrogram()*. Não foi utilizada a função *spectrum()* indicada pelo estudo posto que esta foi depreciada pelo MATLAB.

Para a geração deste gráfico foi utilizada uma janela de hamming de 256 amostras sobre as 8000 amostras do trecho de sinal indicado pelo estudo.

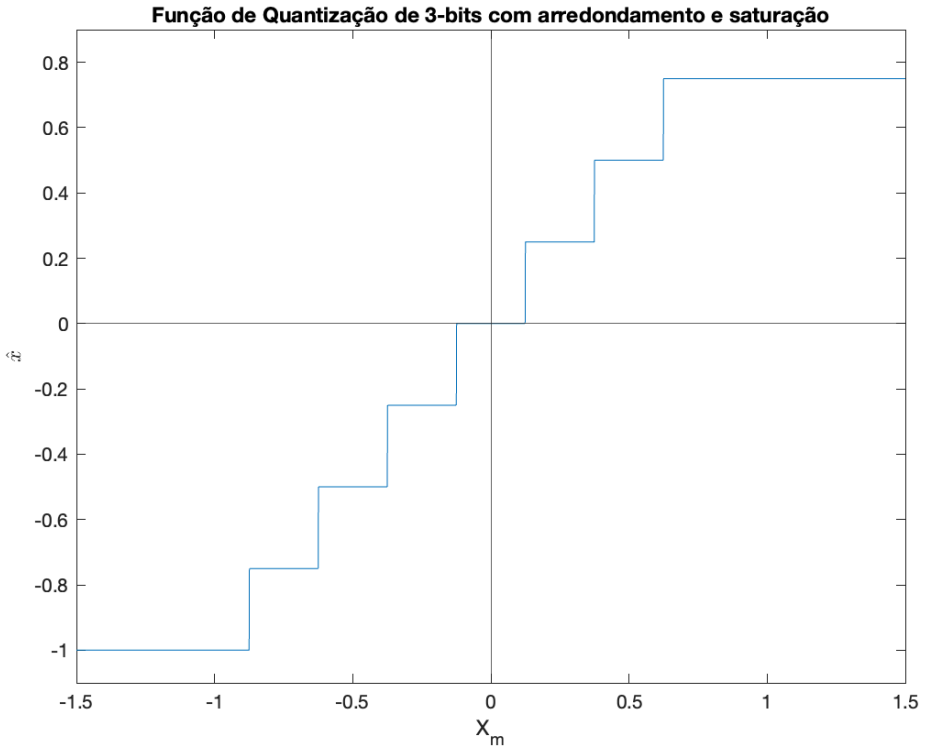


B. Project 2 - *Uniform Quantization*

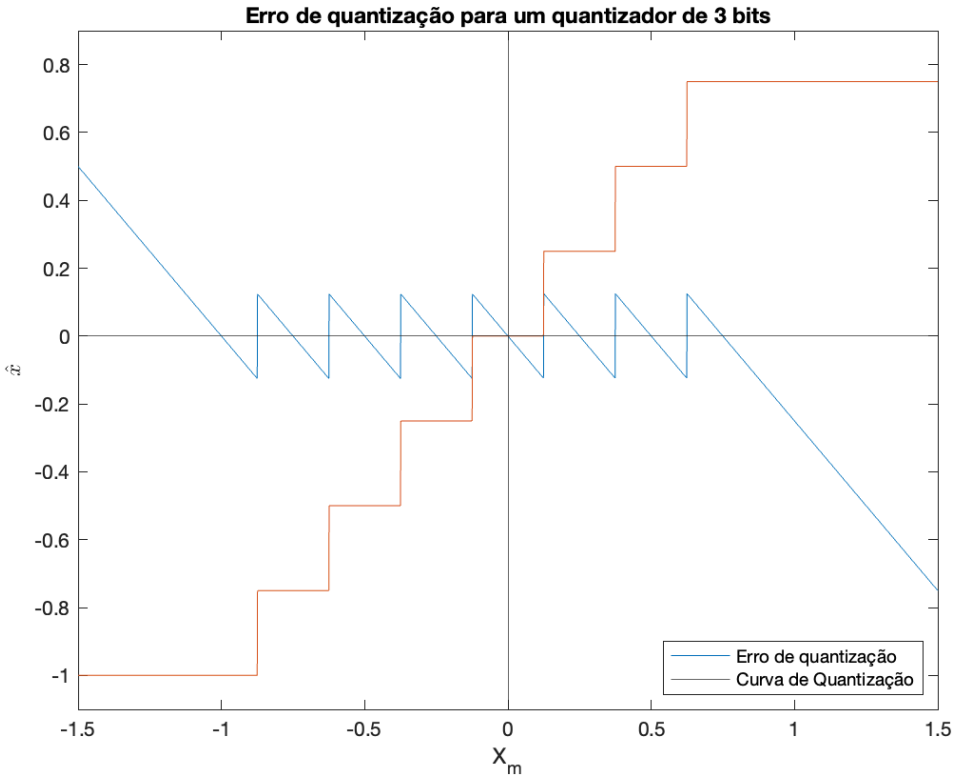
O arquivo de código escrito para a elaboração das figuras e resposta dos exercícios deste projeto encontram-se no caminho ‘speech-quantization/Project2.mlx’.

Exercício 2.1 - *Uniform Quantizer M-File*

A imagem abaixo descreve um quantizador com arredondamento e saturação de 3 bits utilizando a função *fxquant()* provida pelo estudo dirigido.



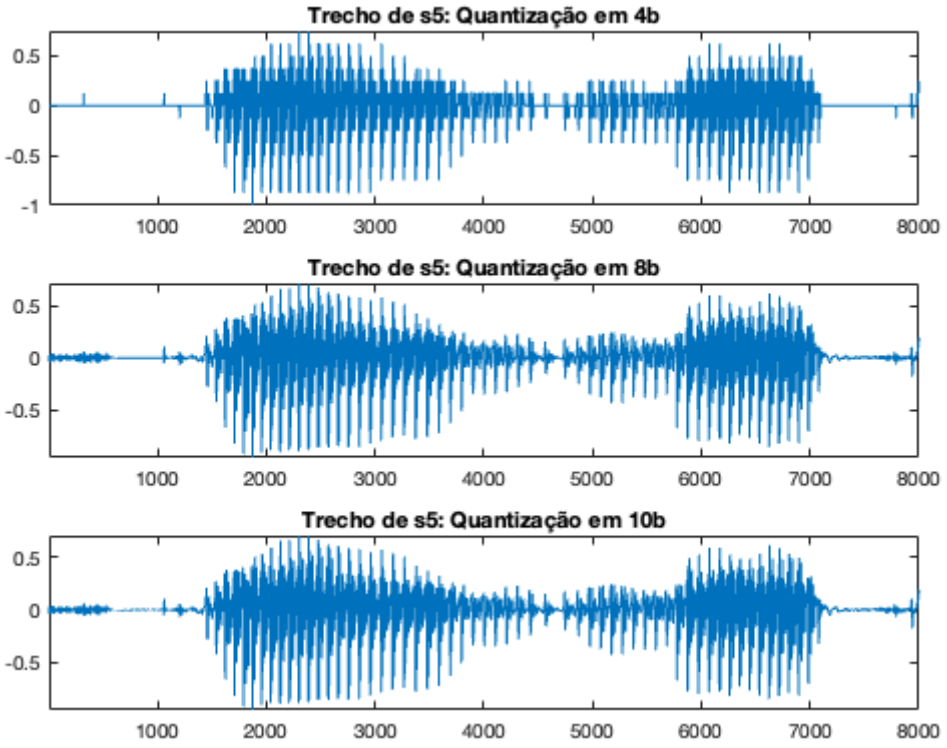
O valor de Δ para esse quantizador é de 0,25, dado como o passo de quantização para 3 bits e o sinal de amplitude -2 a 2. Para um erro de quantização que satisfaça (2 -2), o sinal x deverá ser de no máximo ±15.



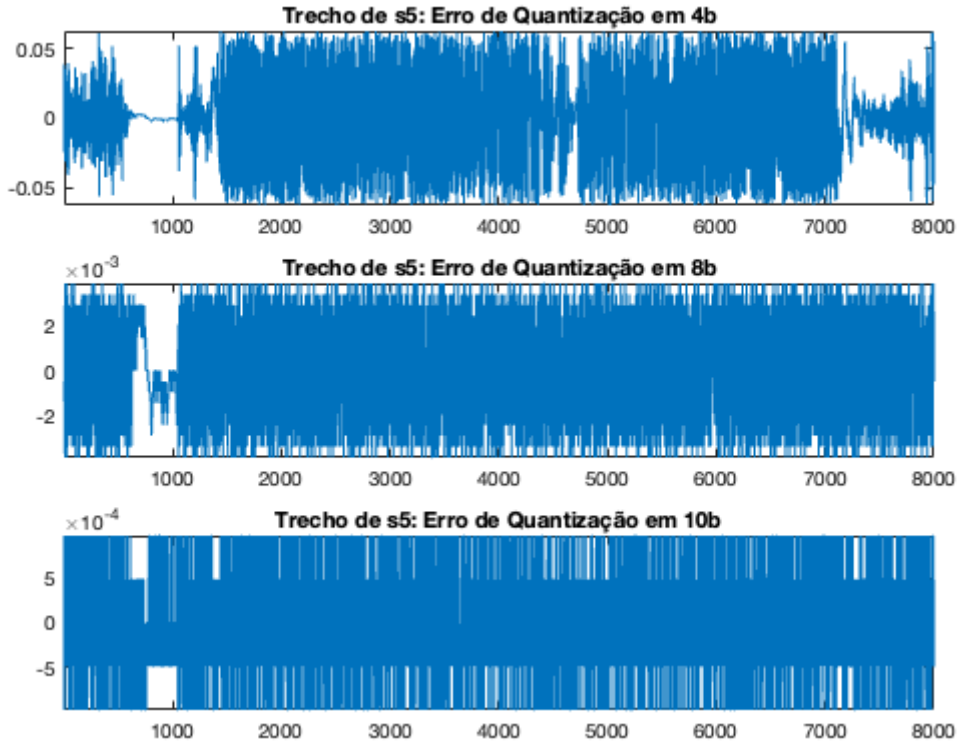
O Gráfico descreve o erro total de um quantizador de 3 bits. Dentro da faixa normal de operação, o erro total será de ±Δ. Ao atingir a faixa de saturação, o quantizador deixa de descrever as variações do sinal e será diretamente proporcional ao desvio da faixa máxima de quantização, posto que o erro é linear.

Exercício 2.2 - *Quantization Experiments*

Abaixo segue experimentos de quantização para 4, 8 e 10 bits, diferentemente dos 12 bits originais do sinal amostrado S5.mat.

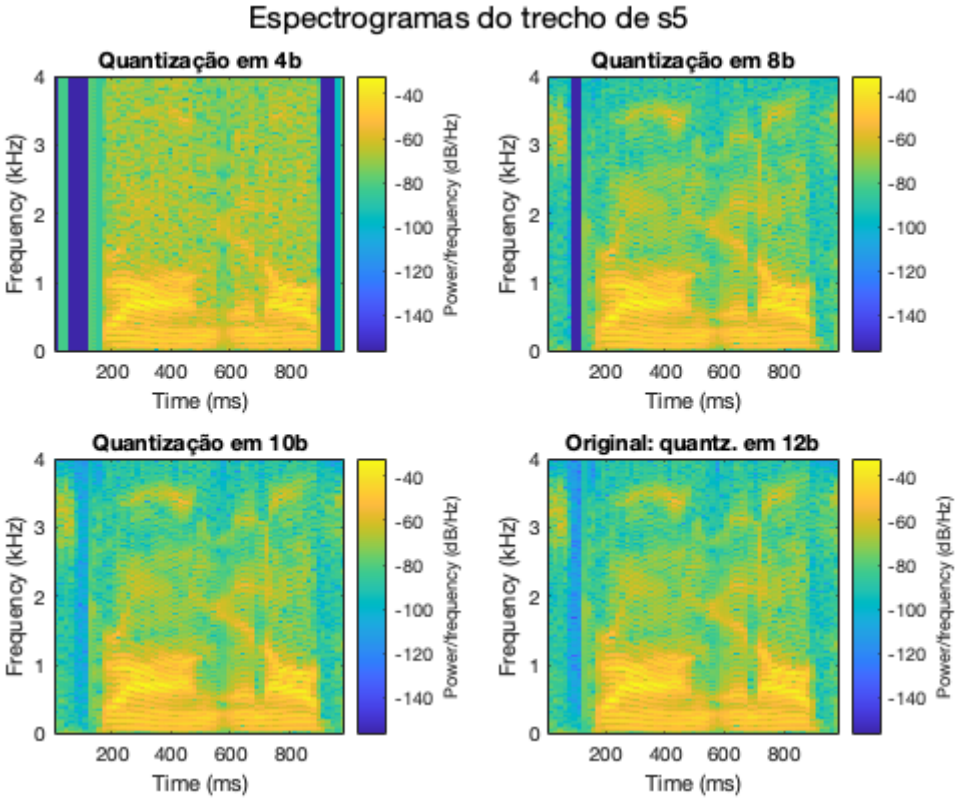


Já na figura abaixo, onde estão os erros de quantização para cada um dos novos sinais amostrados. É possível observar que com a maior redução da taxa de bits, há um desvio maior do sinal original. Conforme aumenta-se a taxa de bits, observa-se a redução no valor absoluto do desvio em cada uma das amostras do sinal. Em 4 bits temos desvios da ordem de 10^{-2}. Já em 8b e 10b temos 10^{-3} e 10^{-4} respectivamente. Posto que o sinal oscila dentro da faixa [-1, 1] de quantização, em 4 bits o erro chega a superar 5% do sinal original em algumas amostras.



Exercício 2.3 - *Spectral Analysis of Quantization Noise*

Na figura abaixo encontram-se os espectrogramas dos sinais re-quantizados em 4, 8 e 10 bits, junto com uma comparação do sinal original quantizado em 12 bits.



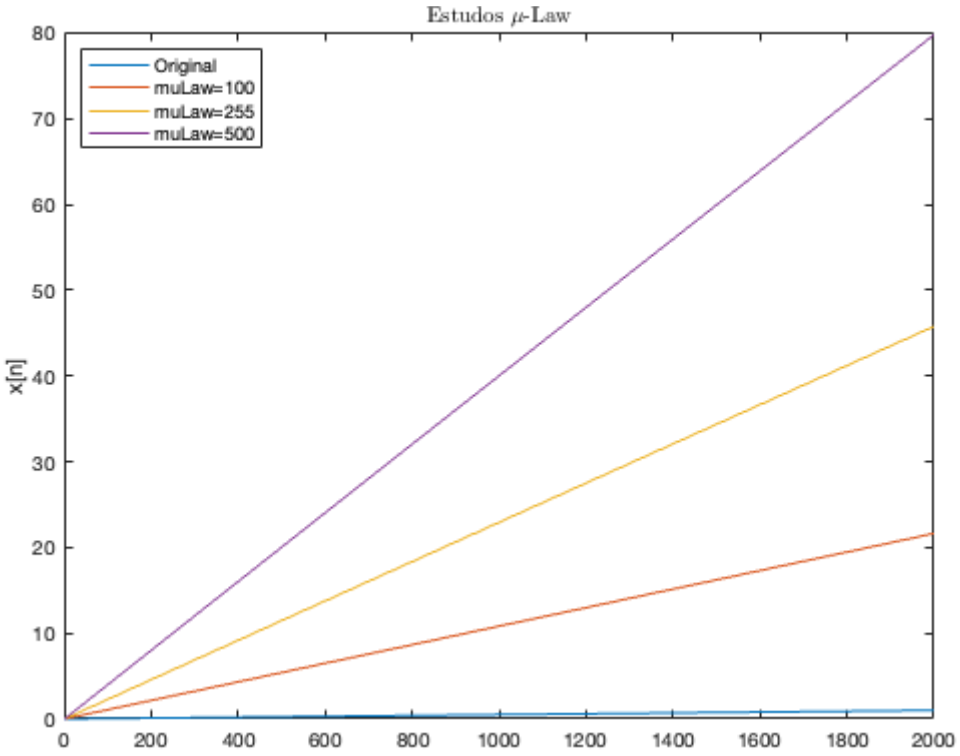
Podemos assumir que o ruído de fundo possui uma característica de ruído branco. O ruído branco é decorrente da de uma distribuição de probabilidade independente para cada uma das amostras. O erro de quantização vai variar de $\pm\Delta/2$.

C. Project 3 - *u-Law Companding*

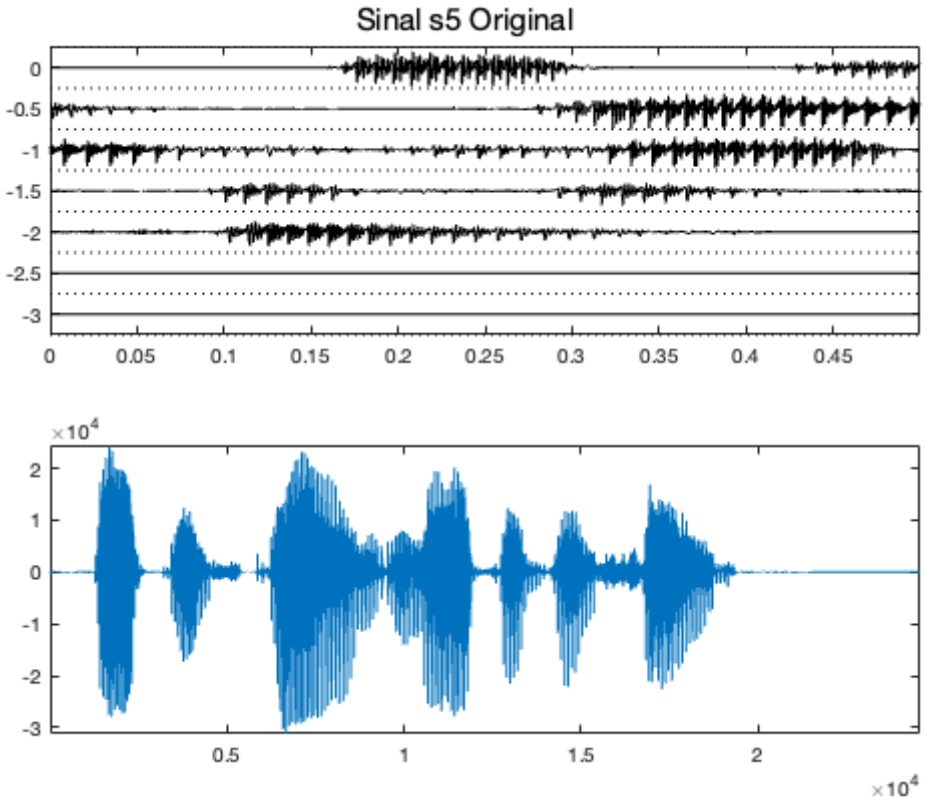
O arquivo de código escrito para a elaboração das figuras e resposta dos exercícios deste projeto encontram-se no caminho 'speech-quantization/Project3.mlx'.

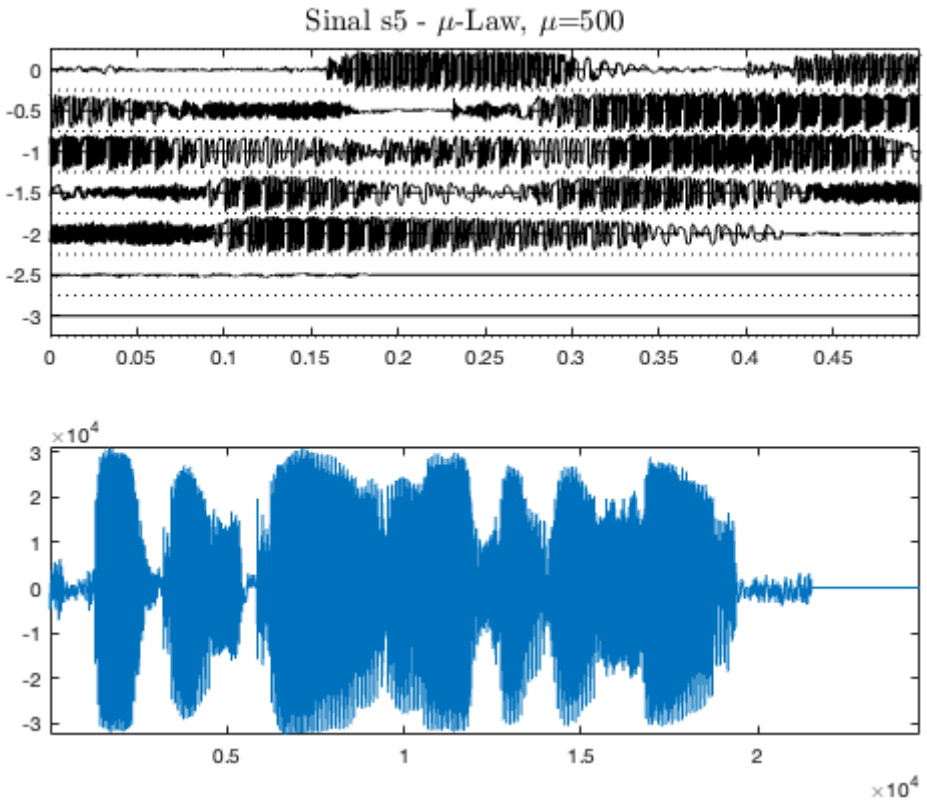
Exercício 3.1 - *u-Law Compressor*

A imagem abaixo descreve a característica *u* para valores de 100, 255 e 500, junto ao vetor de entrada original. Foi utilizada a função *mulaw()* fornecida para aplicar ao sinal S5, resultando nas figuras subsequentes.

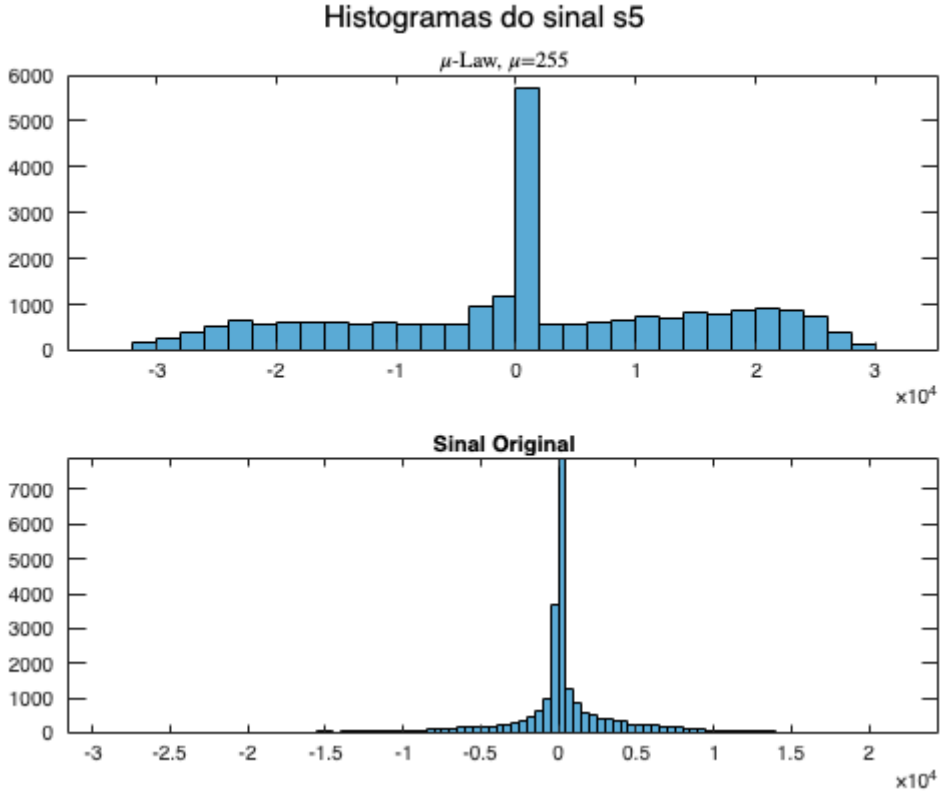


É possível observar a diferença entre o sinal S5 Original e o que teve a compressão u-Law aplicada. As amostras de menor intensidade são agora representadas com valores absolutos maiores, fazendo com que a faixa de representação do sinal em bits seja melhor aproveitada e possibilitando uma representação do sinal com menos bits por palavra. .



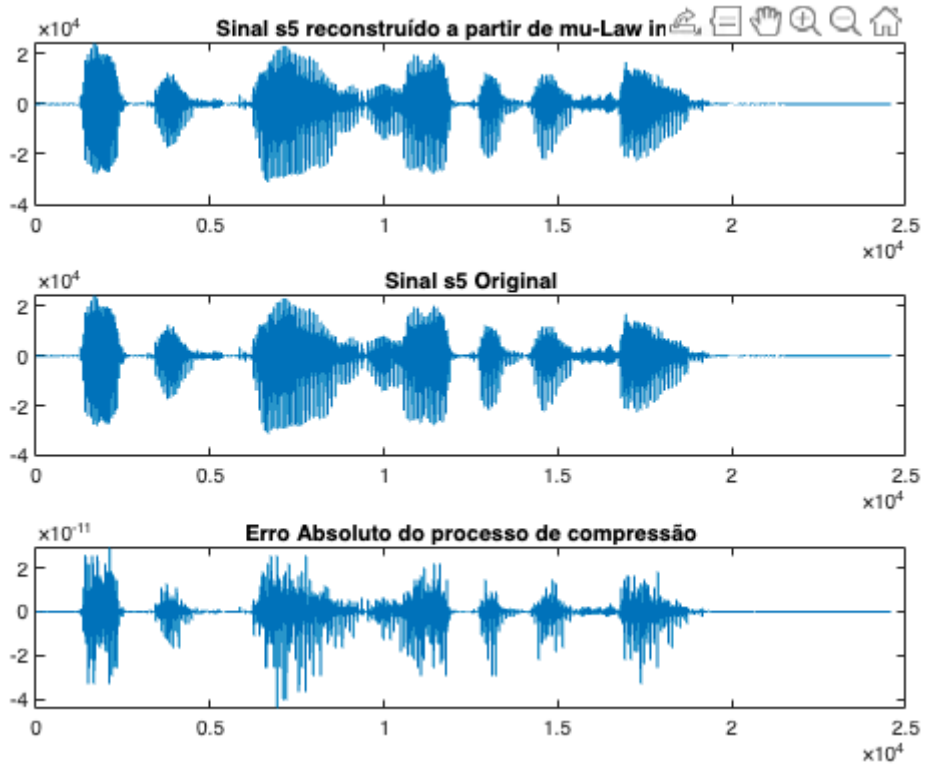


O histograma dos sinais originais e do aplicado u-Law para u=500, presentes na figura abaixo reforçam a mudança na representação do sinal: temos uma quantidade de amostras maior com valor absoluto mais longe de zero.

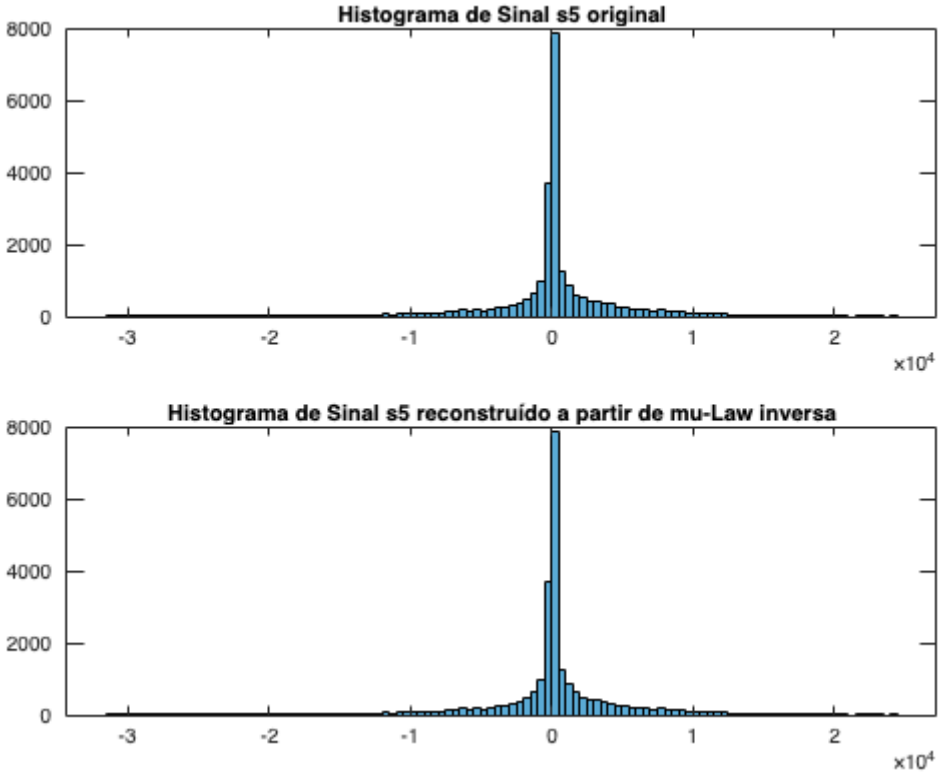


Por fim, foi escrita a função *mulawinv.m*, também disponibilizada. Aplicou-se essa função ao sinal com u-law. Abaixo pode-se ver a forma de onda reconstruída, o sinal original e o erro do

processo de reversão da compressão mu-law. Nota-se que o erro introduzido pelo processo é muito pequeno, da ordem de 10^{-11}.



É possível observar também a o retorno da distribuição original do histograma das amostras do sinal:



D. Project 4 - *Signal-to-Noise-Ratios*

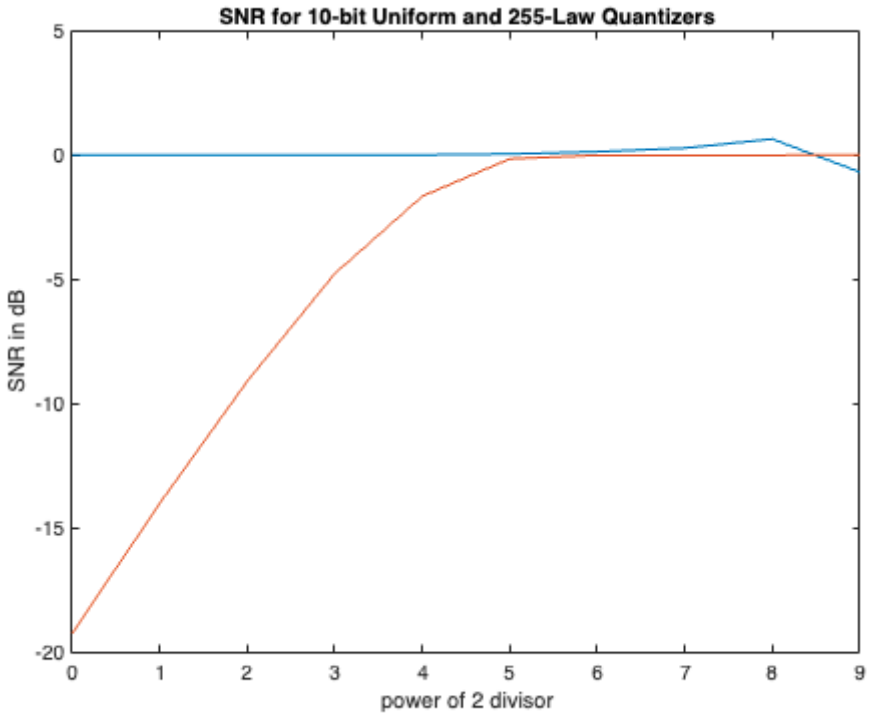
O arquivo de código escrito para a elaboração das figuras e resposta dos exercícios deste projeto encontram-se no caminho 'speech-quantization/Project4.mlx'. Também é possível encontrar as funções '*speech-quantization/snr_mcclellan.m*' e '*speech-quantization/qplot_alt.m'* decorrentes dos exercícios propostos.

Exercício 4.1 - *Signal-to-Noise Computation*

A função solicitada pode ser encontrada em '*speech-quantization/snr_mcclellan.m'*. Aplicando ela para sinais quantizados em 8 e 9 bits, que por sua vez foram computados utilizando-se da função fxquant(), obteve-se um SNR de 37,4 dB e 42,8 dB respectivamente. O sinal s5 foi normalizado entre [-1, 1] de acordo com a documentação para a utilização da função de reamostragem.
Esses valores estão de acordo com o esperado posto que conforme há a redução da palavra de quantização, aumenta-se o ruído do sinal, diminuindo sua proporção de SNR.

Exercício 4.2 - *Comparison of Uniform and u-Law Quantization*

O gráfico a seguir representa o uso da função *qplot()* para um quantizador linear (azul) e para um quantizador u-Law (laranja).
Entretanto, nota-se o comportamento errado da função: um quantizador linear deveria reduzir sua SNR conforme divide-se a amplitude do sinal em cada uma das iterações da função.



Investigando mais a fundo o problema, foi descoberto que a função *qplot()*, por se tratar de uma função da época do MATLAB 4 sendo executada num MATLAB 2024a, esta função utilizou por debaixo dos panos a função ***snr()*** padrão dos *toolboxes* do MATLAB moderno, quando esperava-se pelos autores o uso da função escrita pelos alunos.
A função snr() do MATLAB 2024a de acordo com a documentação utiliza a razão entre a soma das raízes das amostras ao quadrado conforme a equação `r =`

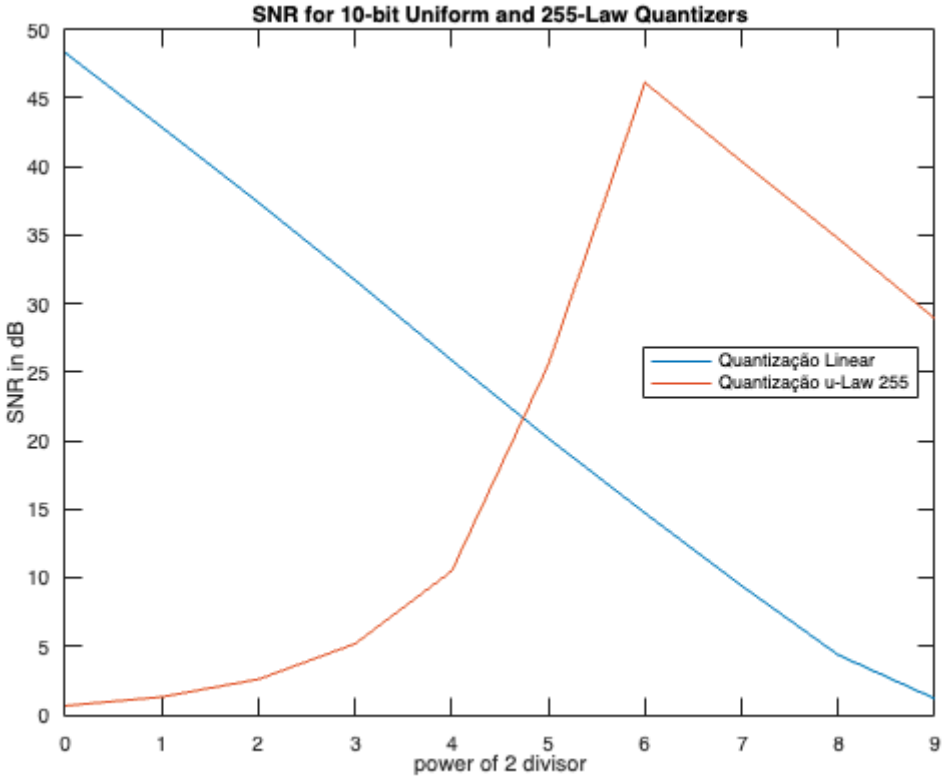
`mag2db(rssq(xi(:))/rssq(y(:)))`, onde r é o SNR do MATLAB 2024 e rssq é o processo de soma já descrito:

$$x_{\mathrm{RSS}} = \sqrt{\sum_{n=1}^{N} |x_n|^2},$$

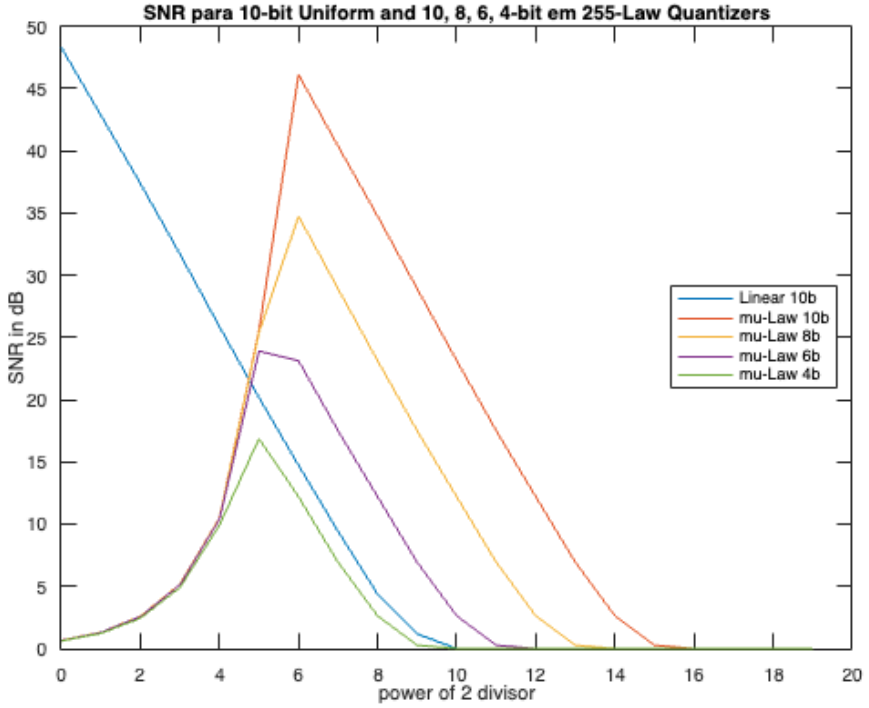
Já a equação do estudo dirigido suprime a raiz quadrada do processo.

$$\mathrm{SNR} = 10 \log \left(\frac{\sum_{n=0}^{L-1} (x[n])^2}{\sum_{n=0}^{L-1} (\hat{x}[n] - x[n])^2} \right) \qquad (4\text{-}1)$$

Corrigindo a função para utilizar a função de SNR escrita, observa-se por meio da figura abaixo o decaimento linear do SNR na quantização uniforme conforme a amplitude das amostras é decrescente, sendo dividida por potências de 2 a cada interação, por 10 iterações. A amplitude final é de uma proporção de 512:1. Esse resultado faz sentido posto que, conforme visto nos projetos anteriores, o ruído de quantização fica mais evidente quanto mais próximo do limiar de quantização o sinal está.



Já na figura abaixo, foi utilizada a função *qplot_alt.m*, derivada de qplot() com o SNR corrigido. Essa função calcula o SNR para o sinal S5 com um quantizador linear de 10b e quantizadores mu-Law de 10, 8, 6 e 4b para comparação. O comportamento é esperado posto que o u-Law é mais eficiente para representar sinais dado que utiliza melhor a faixa de representação da palavra utilizada.



Para um quantizador linear amostrar um sinal numa faixa de amplitude de 64:1 manter ao menos o mesmo SNR de um sinal de 6-bits em u-Law sobre a mesma faixa de amplitude seriam necessários ao menos 12 bits. Em 6-bits o u-Law oferece um SNR de 23,1 dB frente a 14,7 dB em 10 bits de um quantizador linear. Para ter ao menos o mesmo SNR, são necessários 2 bits: