

# Relatório Atividade M4

Leonardo Santos

## Objetivo

O objetivo desta atividade foi aplicar o modelo polinomial com memória para modelar o comportamento de um amplificador de potência, estimar os coeficientes em ponto flutuante e avaliar o impacto da quantização desses coeficientes em ponto fixo com resoluções entre 2 e 20 bits. Adicionalmente, buscou-se realizar a extração direta dos coeficientes a partir de dados quantizados utilizando técnicas de otimização inteira.

## Metodologia

A metodologia seguiu as seguintes etapas:

1. Carregamento dos sinais de excitação e resposta do amplificador.
2. Definição da estrutura do modelo polinomial com memória (ordem e profundidade fixas).
3. Estimação dos coeficientes em ponto flutuante via método dos mínimos quadrados.
4. Quantização dos coeficientes para resoluções de 2 a 20 bits.
5. Avaliação do desempenho por meio do NMSE antes e após a quantização.
6. Extração direta dos coeficientes quantizados utilizando otimização inteira.

## Resultados em Ponto Flutuante

Os coeficientes estimados em ponto flutuante foram:

$$\begin{array}{lll} 0.9735 + 0.0090j & 0.0463 + 0.2087j & -0.2370 - 0.1145j \\ 0.0086 + 0.0544j & -0.5830 - 0.4581j & 0.3378 + 0.3323j \\ -0.1865 - 0.1295j & 0.8158 + 0.4302j & -0.3608 - 0.2919j \end{array}$$

O NMSE obtido foi:

$$-25.50 \text{ dB}$$

A figura a seguir compara a resposta do modelo ajustado com o sinal medido:

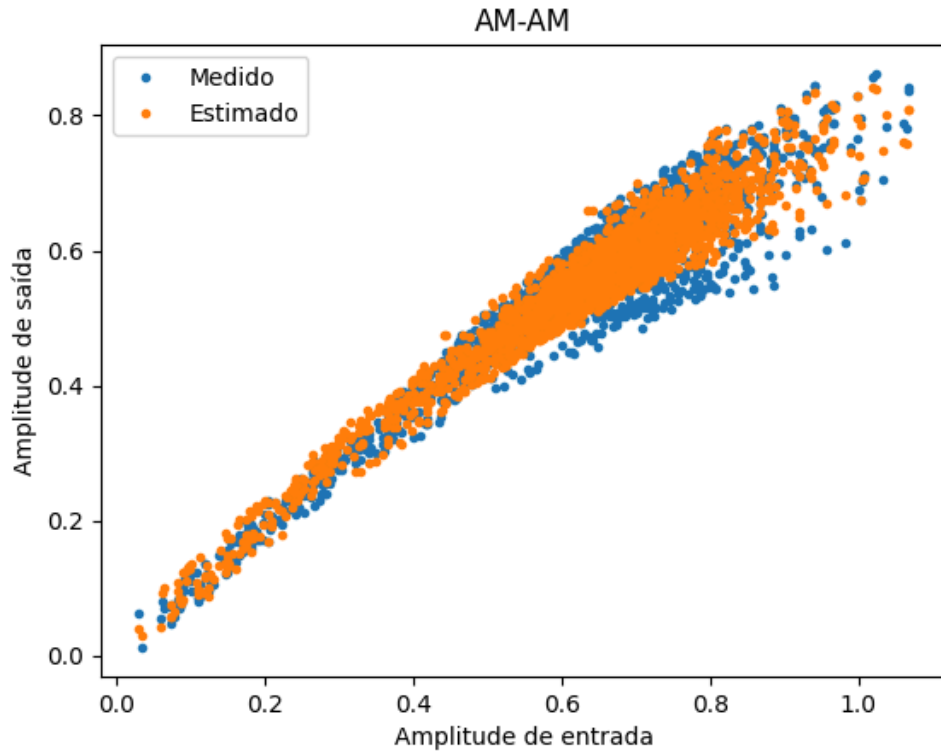


Figura 1: Comparação entre saída medida e modelada (ponto flutuante)

## Efeito da Quantização (2 a 20 bits)

A quantização foi realizada conforme:

$$\text{coef}_q = \text{round}(\text{coef} * 2^N)$$

onde  $N$  é o número de bits de precisão fracionária.

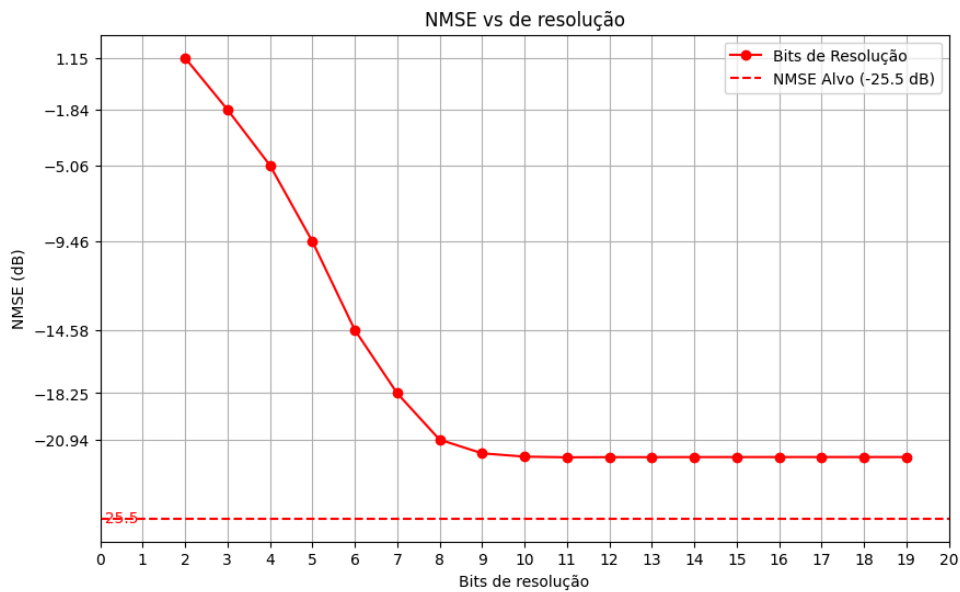


Figura 2: NMSE em função do número de bits de quantização

Observa-se saturação do desempenho a partir de aproximadamente 10 bits, indicando que resoluções superiores trazem ganho marginal.

## Otimização Direta com Coeficientes Quantizados

A tentativa inicial de usar mínimos quadrados com modelo quantizado (truncamentos e aritmética inteira) falhou devido à não-diferenciabilidade da função objetivo, resultando em convergência para solução nula.

A solução adotada foi utilizar o algoritmo **Dual Annealing** (otimização global sem derivadas), tratando os coeficientes diretamente como variáveis inteiras no intervalo permitido pela quantização (ex:  $\pm 255$  para 8 bits com sinal).

Como ponto de partida para a otimização inteira, utilizamos os coeficientes estimados em ponto flutuante (obtidos pelo método dos mínimos quadrados) como chute inicial ( $x_0$ ) no algoritmo de otimização. Esses valores em ponto flutuante são convertidos para a escala inteira adequada seguindo a quantização definida pela precisão (escala  $2^N$ ) ou por regra de arredondamento especificada. Para preservar a natureza discreta do problema, desativamos a busca local em `dual_annealing` (`no_local_search=True`), evitando assim movimentos contínuos na vizinhança que não são compatíveis com a solução inteira.

Embora funcional, essa abordagem apresentou:

- Aumento significativo no tempo de convergência
- Degradação esperada no NMSE devido à restrição discreta

## Resultados com Otimização Inteira (8 bits)

NMSE obtido antes da otimização: **-20.97 dB**

NMSE obtido antes da otimização: **-20.98 dB**

Coeficientes inteiros extraídos (escala  $2^8$ ):

207 + 35j	157 - 233j	-189 + 187j
-68 + 77j	2 + 53j	0 - 17j
80 - 48j	-132 - 92j	158 + 85j

A figura a seguir compara a resposta do modelo com coeficientes quantizados obtidos via otimização inteira com o sinal medido:

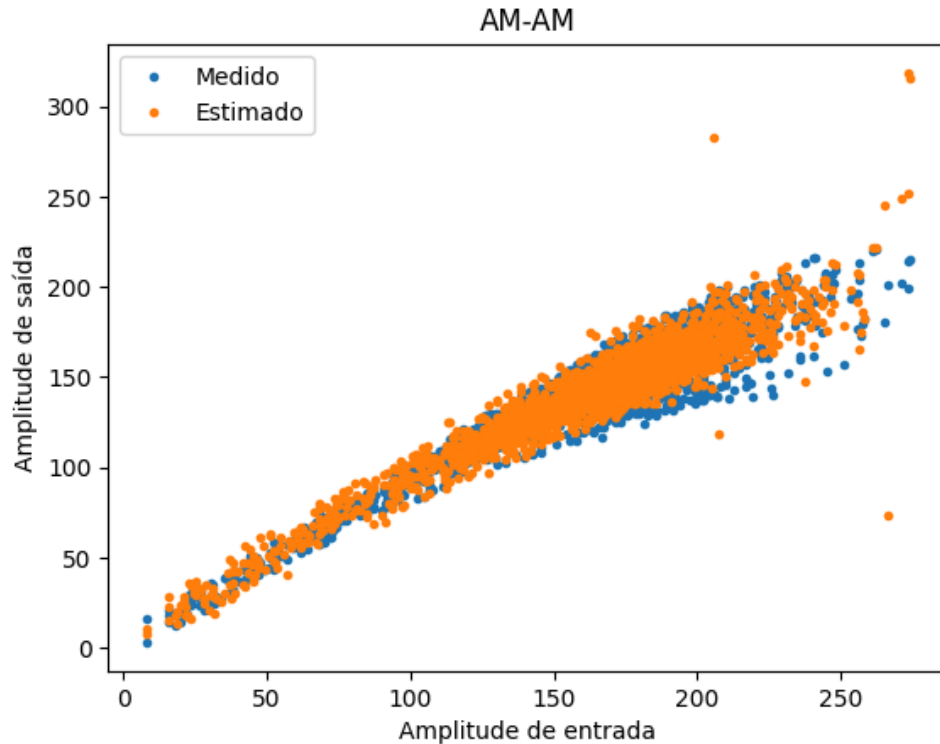


Figura 3: Comparação entre saída medida e modelada (coeficientes quantizados)

No trecho abaixo apresentamos o código principal usado para inicializar a otimização inteira com os coeficientes obtidos por least-squares. Note que `params_float` corresponde ao vetor resultante da estimação em ponto flutuante e que `no_local_search=True` é importante para problemas discretos.

```
ordem = 3
memoria = 2

num_coefs = ordem * (memoria + 1)
total_params = 2 * num_coefs

# Exemplo: coeficientes entre -1.0 e +1.0 em FLOAT
# → em inteiro isso vira:
coef_min = -1.0
coef_max = 1.0

bounds = [
    (coef_min , coef_max)
] * total_params

from scipy.optimize import dual_annealing

result = dual_annealing(
    objetivo,
    bounds,
    x0=params_float.astype(float), # chute inicial: coeficientes do least-squares
    args=(in_data_ext_fixed, out_data_ext_fixed, ordem, memoria, precision),
    maxiter=500,
    initial_temp=5230,
    visit=2.7,
    accept=-5.0,
```

```
no_local_search=True # MUITO IMPORTANTE PARA PROBLEMA DISCRETO  
)
```

## Conclusão

O modelo polinomial com memória foi ajustado com sucesso em ponto flutuante, alcançando NMSE de  $-25.5$  dB. A quantização direta dos coeficientes mostra que 10 bits são suficientes para representar adequadamente a dinâmica do sistema.

A extração direta de coeficientes quantizados via otimização inteira (**Dual Annealing**) é viável, porém resulta em degradação de aproximadamente 8 dB no NMSE (para 8 bits) e aumento significativo no custo computacional. Tais resultados reforçam o trade-off entre precisão, complexidade de implementação e viabilidade em hardware de ponto fixo.