Atividade 5

Leonardo Santos - GRR20196154

Essa etapa consiste no desenvolvimento do modelo do predistorcedor digital.

Para isso inicialmente e feito o mesmo processo para realização do modelo do PA, porem com as entradas e saidas trocadas, conforme ilustrado pela Figura 1, em valores com virgula flutuante.

```python
M = 3
P = 5

# calculo dos coeficientes do modelo do PA
XX_ext = mp(P, M, in_data_ext)
coefficients, _, _, _ = np.linalg.lstsq(XX_ext, out_data_ext[M:], rcond=None)
# Saida estimada do modelo do PA
predicted_val_ext = XX_ext @ coefficients

# Calculo do modelo do pre-distorcedor
XX_val_pre = mp(P, M, predicted_val_ext)
coefficients_pre, _, _, _ = np.linalg.lstsq(XX_val_pre , in_data_ext[M*2:], rcond=None)
# Saida estimada do modelo do predistorcedor
predicted_val_pre = XX_val_pre @ coefficients_pre
```

Figura 1: Código do para modelo do predistorcedor em virgula flutuante

E o resultado dessa etapa esta ilustrado pela Figura 2 a seguir:

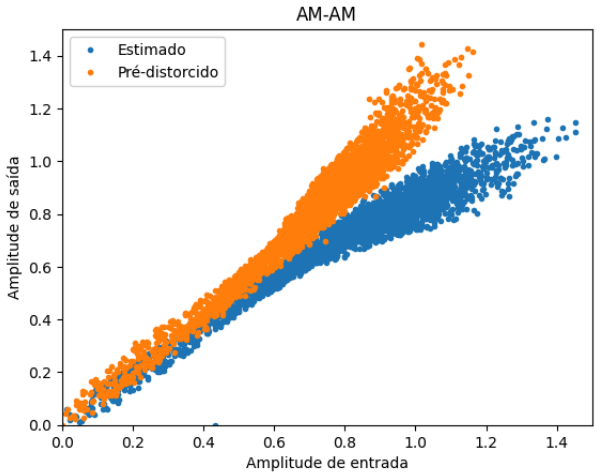

Figura 2: Gráficos entrada e saída em virgul flutuante

Portanto observa-se que a saida do prédistorcedor esta conforme o esperado. Em seguida foram realizados os calculos com os valores para virgula fixa que pode ser utilizando as funções desenvolvidas nas ultima atividade, conforme ilustrado pela Figura 3 a seguir.

```python
# normalização
in_val_fixed = np.round(in_val_norm * (2 ** p_bits))
out_val_fixed = np.round(out_val_norm * (2 ** p_bits))
in_ext_fixed = np.round(in_ext_norm * (2 ** p_bits))
out_ext_fixed = np.round(out_ext_norm * (2 ** p_bits))

# calculo dos coeficientes do modelo do PA
XX_ext_pa = mp(P, M, in_ext_norm)
coefficients_norm_pa, _, _, _ = np.linalg.lstsq(XX_ext_pa, out_ext_norm[M:], rcond=None)

predicted_val_ext = XX_ext_pa @ coefficients_norm_pa

# Calculo dos coeficientes do modelo do pre-distorcedor
XX_ext_pre = mp(P, M, predicted_val_ext)
coefficients_norm_pre, _, _, _ = np.linalg.lstsq(XX_ext_pre , in_ext_norm[M*2:], rcond=None)

# modelo do PA
XX_val_PA = mp_int(P, M, in_val_fixed, p_bits)
predicted_val_fixed_PA, e_bits = MultiplicadorMatrizes(coefficients_norm_pa, XX_val_PA, p_bits)

# modelo do predistocedor
XX_val_pre = mp_int(P, M, in_val_fixed, p_bits)
predicted_val_fixed_pre, e_bits = MultiplicadorMatrizes(coefficients_norm_pre, XX_val_pre, p_bits)
```

Figura 3: Código para modelo do predistorcedor em virgula fixa

E o resultado dessa implementação esta ilustrado pela Figura 4 a seguir:

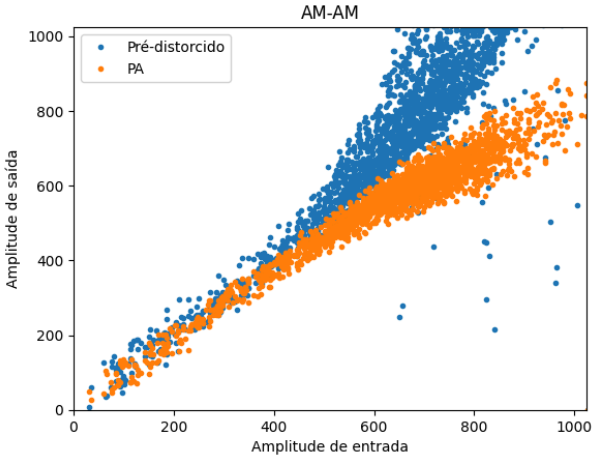

Figura 4: Gráficos entrada e saída em virgula fixa

Em seguida foi feito a simulação do predistorcedor tanto para virgula fixa,com uma resolução de 10 bits, quanto para virgula flutuante cujos os resultados estão ilustrados pelas Figura 5, Figura 6 a seguir:

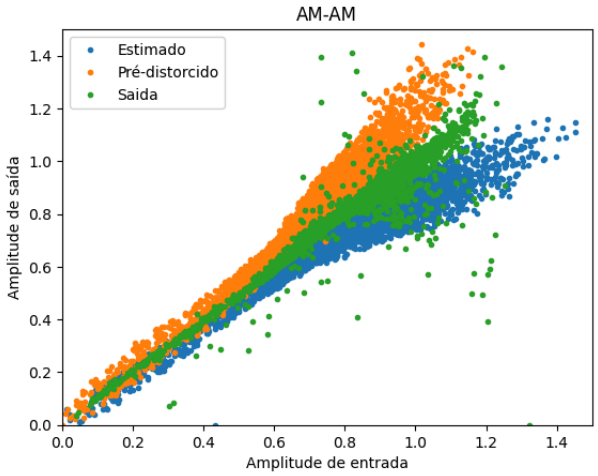

Figura 5: Gráficos entrada e saída em virgula flutuante simulado

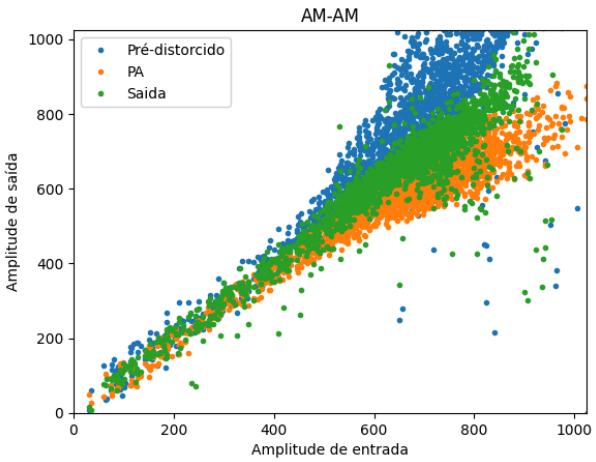

Figura 6: Código para modelo do predistorcedor em virgula fixa simulado