Universidade Federal do Paraná
Departamento de Engenharia Elétrica

Pedro Lucca Pereira da Veiga
Adriely Teixeira de Paula

Processamento Digitais de Sinais II
Laboratório 4 - Processo de aprendizagem de neurônios dos tipos Adaline e Perceptron.

Curitiba
2025

1 Exercício 1

1.1 Avalie o arquivo de exemplo "adaline.m", que realiza o treinamento de um neurônio ADALINE para encontrar a equação de uma reta. Mantenha o número de épocas de treinamento fixo em 100 e varie a taxa de aprendizado η de 0,01 a 0,1 e comente os resultados obtidos. Mantenha a taxa de aprendizado η fixa em 0,02 e varie o número de épocas de 100 a 1000, em passos de 200. Comente os resultados. Descreva como o algoritmo de treinamento da Adaline é implementado no código.

O neurônio Adaline é um tipo de neurônio básico utilizado para tratar de medidas contínuas. Seu algoritmo de treinamento, baseado na regra Delta, busca ajustar seus pesos minimizando o erro quadrático médio (MSE) entre a saída da rede e a saída desejada.

Funcionamento do código:

1. Inicialização:
   - Geração de amostras com ruído a partir de uma função linear do tipo
   $$f(x) = ax + b + \text{ruído}$$
   - Inclusão de um termo de *bias* na matriz de entrada;
   - Inicialização dos pesos com valores aleatórios.
2. Treinamento:
   - Em cada época, todas as amostras são apresentadas de forma aleatória.
   - Para cada amostra, realiza-se:
     - Cálculo da saída:
     $$y = W \cdot Xi$$
     - Cálculo do erro:
     $$e = y_d - y,$$
     onde y_d é a saída desejada e y a saída real
     - Atualização dos pesos utilizando a regra:
     $$\Delta W_i = \eta \cdot e \cdot X_i$$
   - Ao final de cada época, calcula-se o erro quadrático médio.
   - O treinamento é encerrado quando a diferença entre erros quadráticos consecutivos for menor que a tolerância especificada ou quando o número máximo de épocas for atingido.

3. Visualização:

   - A cada época, há o ajuste da reta, mostrando a aproximação com a função original.
   - Ao final do treinamento, exibe-se o erro quadrático médio por época.

Variando a taxa de aprendizado η

Observando as figuras 1 e 2, referentes a variação da taxa de aprendizado η, é possível observar que com taxas muito baixas $(\eta = 0.01)$ a atualização dos pesos é muito gradual (necessita de mais épocas para apresentar uma alteração significante) e o erro pode reduzir lentamente. Portanto, tem-se um treinamento mais lento com convergência estável, mas pode não alcançar bons resultados para poucas épocas de treinamento.

Taxas muito altas podem apresentar convergência a um erro mínimo em um número menor de épocas. Contudo, podem levar a oscilações ou instabilidade, já que o algoritmo pode ultrapassar o ponto ótimo.

Nota-se Os valores intermediários oferecem uma boa relação entre a velocidade e estabilidade.

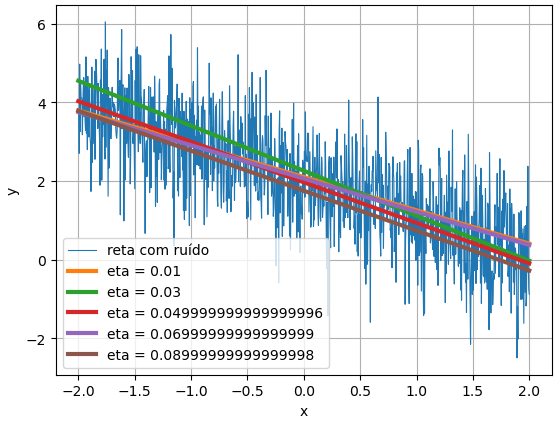

Figure 1: Treinamento Fixo com η Variando

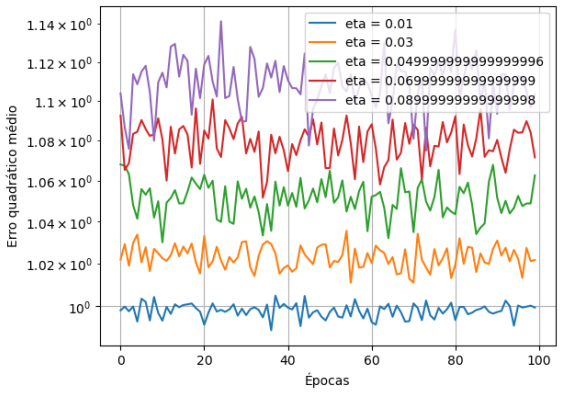

Figure 2: Erro Quadrático Médio com η Variando

Variando o número de épocas

Nos gráficos 3 e 4 estão apresentados os resultados e o erro quadrático médio para diferentes quantidades máximas de épocas de treinamento. Nota-se que o erro quadrático médio diminui rapidamente nas primeiras épocas e mais estável (contido em um intervalo menor de valores) ao final do processo.

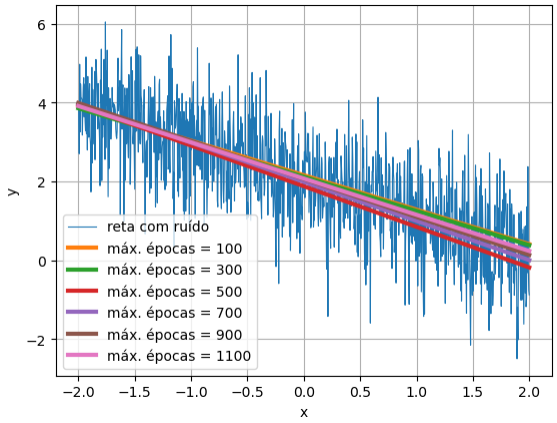

Figure 3: Treinamento Fixo com Épocas Variando

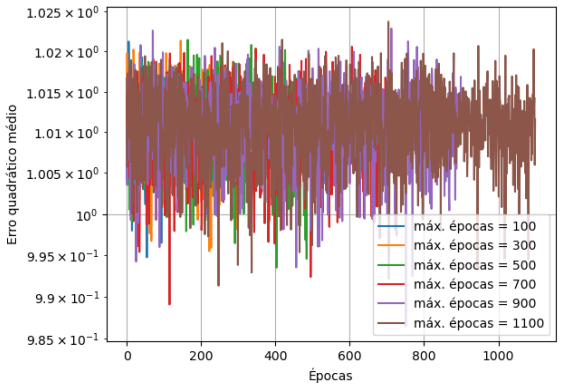

Figure 4: Erro Quadrático Médio entre Épocas

O aumento das épocas tem efeito limitado após certo ponto, o que indica convergência. Utilizar um número extremamente alto de épocas para atingir o mínimo possível de erro pode impedir a capacidade de generalização do modelo (também conhecido como *overfitting*) e, portanto, indesejável.

2 Exercício 2

2.1 Avalie o exemplo de treinamento de um neurônio Perceptron para implementar portas AND e OR.

- Descreva como o algoritmo é implementado.

O algoritmo é implementado da seguinte forma:

1. Definição do problema:
   - O usuário escolhe entre treinar uma porta AND ou OR. As entradas e saídas desejadas (valores verdade da tabela-verdade) são definidos de acordo.
2. Inicialização:
   - Os pesos e os bias são inicializados aleatoriamente.
   - O bias é tratado como uma entrada extra fixa com valor -1.
   - É definida uma taxa de aprendizado η e uma tolerância de erro.
3. Treinamento:
   - A cada época, todos os padrões de entrada são apresentados ao perceptron, em ordem aleatória.

- A saída do perceptron é computada por um limiar (função degrau): se o somatório ponderado for maior ou igual a 0, a saída é 1; caso contrário, 0.
- Calcula-se o erro entre a saída desejada e a saída atual.
- O vetor de pesos é atualizado segundo a regra de Hebb modificada (Perceptron Rule):

$$\Delta W = \eta \cdot erro \cdot entrada$$

- O erro quadrático acumulado por época é armazenado.

4. Parada:
   - O treinamento para quando o erro acumulado da época é menor que a tolerância.
5. Visualização:
   - O código plota a fronteira de decisão, a reta de separação, e o erro por época.

A figura 5 mostra a reta de separação da porta AND, os pontos azuis representam as entradas com saída 0 (falsas) e os vermelhos com saída 1 (verdadeiras), a reta pontilhada representa a fronteira de decisão aprendida pelo Perceptron. É observa que a reta separa corretamente os pontos (1,1) (único com saída 1) dos demais, mostrando que a porta AND é linearmente separável, e o Perceptron conseguiu aprender essa separação com sucesso.

A figura 6 apresenta o erro quadrático por época para porta AND, o erro começa em 3.0, três classificações incorretas na primeira época, e em apenas duas épocas cai para 0. Indicando um aprendizado rápido e eficiente, típico de problemas linearmente separáveis como o AND.

Para a porta OR, a reta da separação na figura 7 mostra os pontos com saída 1 (vermelhos) que possuem ao menos uma entrada 1. Apenas o ponto (0,0) tem saída 0 (azul). A reta de separação corretamente divide o ponto (0,0) dos demais, demonstrando que o modelo aprendeu a lógica da OR, já que, assim com a porta AND, também é linearmente separável.

A figura 8 mostra o erro quadrático por época, o erro começa em 2.0 e cai para 0 em apenas 1 época, demonstrado um aprendizado ainda mais rápido.

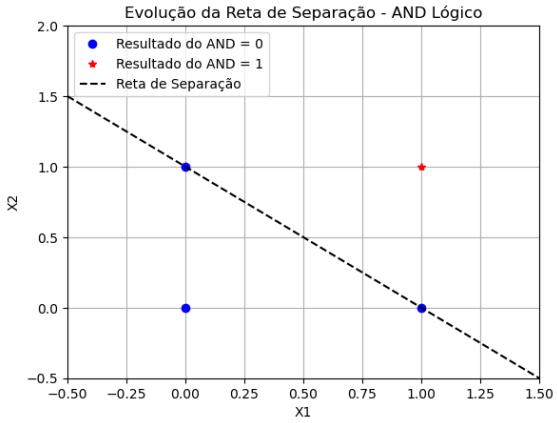

Figure 5: Evolução do Treinamento - Porta AND

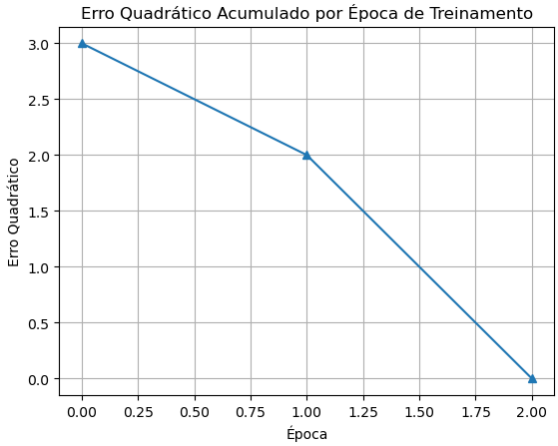

Figure 6: Erro Quadrático Acumulado por Época - Porta AND

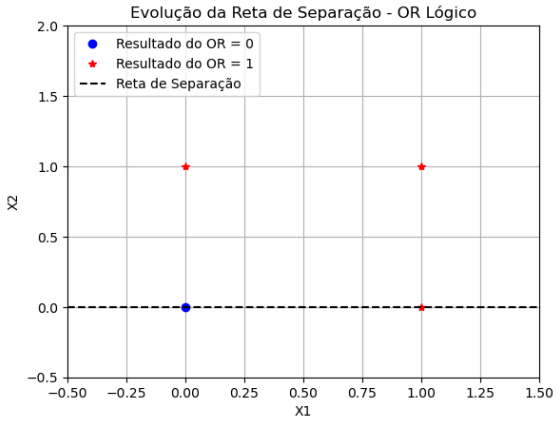

Figure 7: Evolução do Treinamento - Porta OR

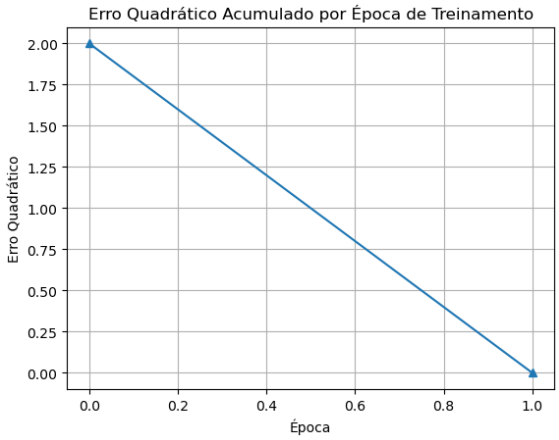

Figure 8: Erro Quadrático Acumulado por Época - Porta OR

- Altere o exemplo para que o neurônio seja treinado para portas lógicas NAND e NOR. Apresente a evolução do erro no treinamento. Comente os resultados obtidos.

Para a porta NAND, tem-se o sucesso do aprendizado após cerca de 26 épocas. As oscilações no erro durante o treinamento são normais devido à natureza aleatória na ordem de apresentação dos padrões e aos ajustes dos pesos, porém o erro final zero confirma o aprendizado correto de acordo com a lógica da porta.

Para porta NOR, o Perceptron conseguiu encontrar uma solução muito rapidamente em apenas 2 épocas. Esse comportamento pode variar ligeiramente com diferentes inicializações aleatórias dos pesos ou com a ordem de apresentação dos dados, mas normalmente a convergência é rápida. O gráfico mostra uma con-

vergência perfeita e eficiente, o que é um ótimo resultado para esse tipo de problema.

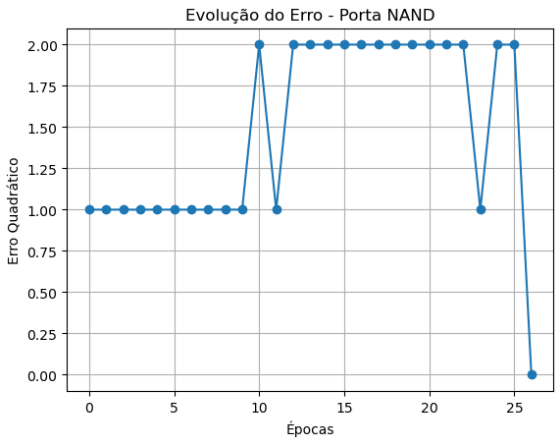

Figure 9: Evolução do Erro no Treinamento - Porta NAND

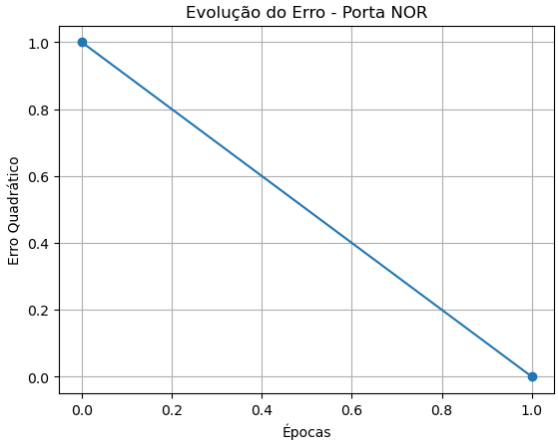

Figure 10: Evolução do Erro no Treinamento - Porta NOR

- Altere o exemplo para que o neurônio seja treinado para implementar uma porta lógica XOR. Apresente a evolução do erro no treinamento. Comente os resultados obtidos.

A variação do erro quadrático entre 1.0 e 2.0 mostra que o Perceptron está tentando ajustar os pesos, porém não consegue encontrar uma combinação que classifique corretamente todos os padrões da porta XOR. Esse comportamento é típico quando o modelo está tentando "forçar" uma solução para algo que não é linearmente separável.

A partir da época 30, o erro quadrático salta para 4.0, que é o erro máximo possível com 4 padrões, demonstrando que, em algumas épocas, o Perceptron classificou todos os padrões de forma incorreta. Por fim, a estabilização em um erro alto confirma que o modelo não é capaz de aprender a função XOR.

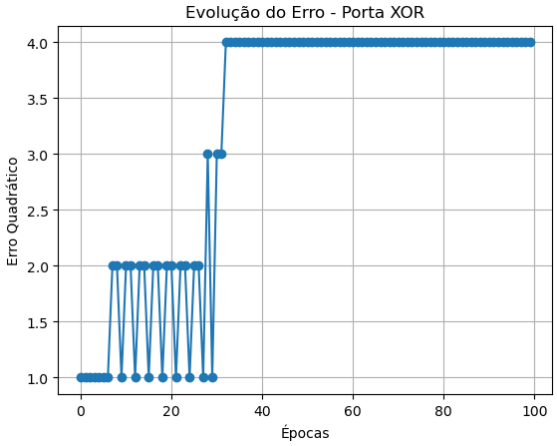

Figure 11: Erro Quadrático Acumulado por Época - Porta XOR

- Altere o exemplo para treinar portas AND e OR de três entradas. Apresente a evolução do erro no treinamento. Comente os resultados obtidos.

Para a porta AND de três entradas, o erro quadrático começa em 3.0, que é o número de erros nos primeiros ciclos. Pode-se ver que durante o treinamento o erro oscila bastante, indicando que a ordem de apresentação dos padrões de forma aleatória afetou a aprendizagem e o modelo está ajustando os pesos tentando encontrar uma reta para separação correta dos padrões. Porém, após a metade da época 17, o erro atinge zero, indicando que o algoritma conseguiu aprender a função AND com 3 entradas.

Para a porta OR de 3 entradas, como a função também é linearmente separável, então o Perceptron simples consegue aprender. Também se tem as oscilações naturais devido ao treinamento com padrões embaralhados e ajustes sucessivos dos pesos. Porém, após a época 25, tem-se erro 0, indicando que o modelo convergiu com sucesso.

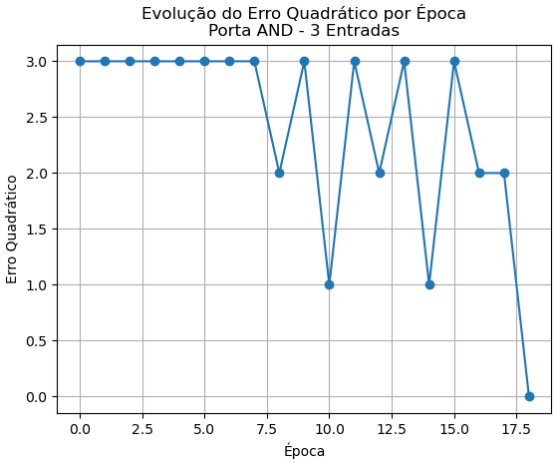

Figure 12: Erro Quadrático Acumulado por Época - Porta AND

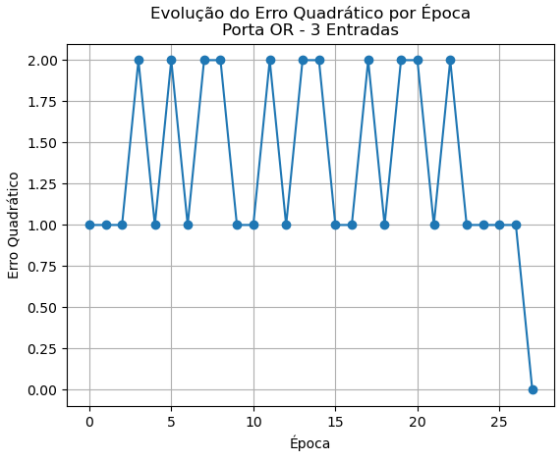

Figure 13: Erro Quadrático Acumulado por Época - Porta OR