```
clear all
close all

% ------------------------------------------------------------------------------------------------------------------
% Inicializa com um LOOP para contar a quantidade de Treinamentos e Classificaçao a Serem Realizadas;
% ------------------------------------------------------------------------------------------------------------------
QT = 5;                  % Quantidade de Treinamentos que se pretende realizar consecutivamente;
TR = 0;
for TR = 1:QT            % Realiza N vezes (TR) o Treinamento e Classificaçao do Perceptron -> Conforme Variavel QT;

% ------------------------------------------------------------------------------------------------------------------
% X eh "Vetor de Entrada do Padrao" - [limiar(Bias), X1, X2, X3, X4];
% ------------------------------------------------------------------------------------------------------------------

X = [
-1    0.4329     -1.3719    0.7022     -0.8535;
-1    0.3024     0.2286     0.8630     2.7909;
-1    0.1349     -0.6445    1.0530     0.5687;
-1    0.3374     -1.7163    0.3670     -0.6283;
-1    1.1434     -0.0485    0.6637     1.2606;
-1    1.3749     -0.5071    0.4464     1.3009;
-1    0.7221     -0.7587    0.7681     -0.5592;
-1    0.4403     -0.8072    0.5154     -0.3129;
-1    -0.5231    0.3548     0.2538     1.5776;
-1    0.3255     -2.0000    0.7112     -1.1209;
-1    0.5824     1.3915     -0.2291    4.1735;
-1    0.1340     0.6081     0.4450     3.2230;
-1    0.1480     -0.2988    0.4778     0.8649;
-1    0.7359     0.1869     -0.0872    2.3584;
-1    0.7115     -1.1469    0.3394     0.9573;
-1    0.8251     -1.2840    0.8452     1.2382;
-1    0.1569     0.3712     0.8825     1.7633;
-1    0.0033     0.6835     0.5389     2.8249;
-1    0.4243     0.8313     0.2634     3.5855;
-1    1.0490     0.1326     0.9138     1.9792;
-1    1.4276     0.5331     -0.0145    3.7286;
-1    0.5971     1.4865     0.2904     4.6069;
-1    0.8475     2.1479     0.3179     5.8235;
```

```
-1    1.3967     -0.4171    0.6443     1.3927;
-1    0.0044     1.5378     0.6099     4.7755;
-1    0.2201     -0.5668    0.0515     0.7829;
-1    0.6300     -1.2480    0.8591     0.8093;
-1    -0.2479    0.8960     0.0547     1.7381;
-1    -0.3088    -0.0929    0.8659     1.5483;
-1    -0.5180    1.4974     0.5453     2.3993;
-1    0.6833     0.8266     0.0829     2.8864;
-1    0.4353     -1.4066    0.4207     -0.4879;
-1    -0.1069    -3.2329    0.1856     -2.4572;
-1    0.4662     0.6261     0.7304     3.4370;
-1    0.8298     -1.4089    0.3119     1.3235];

% ------------------------------------------------------------------------------------------------------------
% Atribui o tamanho do "Vetor de Entrada do Padrao" para M,N (35,5) que sera utilizada nos LOOPs;
% ------------------------------------------------------------------------------------------------------------

[M,N] = size(X);    %M= Nro.Linhas da Matriz X
                    %N= Nro.Colunas da Matriz X

%M=35;      %Nro. Linhas    (Da Matriz X -> Vetor Entrada)
%N=5;       %Nro. Colunas   (Da Matriz X -> Vetor Entrada)

% ------------------------------------------------------------------------------------------------------------
% D eh "Vetor de Saida Desejada" em relaçao ao "Vetor de Entrada do Padrao"
% [Para -1 os valores de entrada (X1,X2,X3,X4) direcionam-se para "Valvula A" -> Pertencem a Classe 1]
% [Para +1 os valores de entrada (X1,X2,X3,X4) direcionam-se para "Valvula B" -> Pertencem a Classe 2]
% ------------------------------------------------------------------------------------------------------------
D = [
1; -1; -1; -1; 1;
1; 1; 1; -1; 1;
-1; -1; 1; 1; -1;
-1; 1; -1; -1; 1;
1; -1; -1; 1; -1;
1; -1; 1; -1; 1;
1; 1; -1; -1; -1];
```

```
%-------------------------------------------------------------------------------------------------------------
% W sao os "Pesos Sinapticos" que serao multiplicados pelo vetor de entrada - [W1*Limiar + W2*X1, W3*X02 + W4*X3 +
W5*X4];
%-------------------------------------------------------------------------------------------------------------
W = rand(1,5);          % W -> Sao os Pesos Sinapticos utilizado spara "ajustar" a Rede; Sao gerados aleatoriamente;
WAnterior = W;          % WAnterior -> Recebe W para conferencia posterior. Conforme mudanças ocorridas no ajuste dos
Pesos;


%-------------------------------------------------------------------------------------------------------------
% TaxApr (Neta) = E a taxa de Aprendizagem - Utilizado para Ajuste dos Pesos Sinapticos;
% epoca = E a quantidade de interacoes ocorridas durante o treinamento para Ajuste dos Pesos Sinapticos;
% epson = E o Vetor Gradiente utilizado p/ Minimizar o Erro.
%-------------------------------------------------------------------------------------------------------------

DifeDU = 0;             % SomaEQM -> Primeira parte do processo para encontrar a Funcao Quadratica (EQM) - Utilizada
para Calculo do EQM;
P = 0;                  % P -> Quantidade de Padroes de Treinamento (Qtde.Linhas de X) - Utilizada para Calculo do EQM;
U = 0;                  % U -> Saida do Combinador Linear - Resultado da SOMA da MULTIPLICACAO dos valores do Padrao de
treinamento (X) pelos "Pesos Sinapticos" (W) gerados aleatoriamente;
EQM = 0;                % Erro Quadratico Medio -> Diferenca entre U (Saida do Combinador Linear) e D (Saida Desejada);
TaxApr = 0.0025;        % N(Eta) - Taxa de Aprendizagem -> Utilizada para "rearranjar" (para achar os melhores
resultados) dos pesos sinapticos durante a fase de treinamento.

EQMAnter = 0;           % Erro Quadratico Anterior -> Utilizado como FLAG, atribuiçao e passagem de valores no LOOP;
EQMAtual = 0;           % Erro Quadratico Atual -> Utilizado como FLAG, atribuiçao e passagem de valores no LOOP;

EpsPreci = 10e-6;       % E(Epsilono) - Valor de Precisao -> Utilizado para estar mais proximo possivel do W "Otimo"
(Nao dando "passos" muito distantes).
epoca = 0;              % Contagem dos LOOPINGs Realizados -> Mostra a quantidade de "calculos" (iteraçoes) realizados
durante o processo de treinamento da RN.


%-------------------------------------------------------------------------------------------------------------
% Exibe Cabeçalho para demonstrar/visualizar os resultados do Treinamneto e das Atividades realizadas (para
conferencia!)...
%-------------------------------------------------------------------------------------------------------------
fprintf('\n----------------------------\n')
```

```
fprintf('REDE NEURAL - ADALINE')
fprintf('\n-----------------------------\n')

fprintf('4 Entradas (X1,X2,X3,X4) \n')
fprintf('1 Neuronio \n')
fprintf('1 Saida \n')

fprintf('\n\n---------- Resultado Treinamento da RN - Adaline ----------\n\n')
fprintf('Onde:\n')
fprintf('(D)-> Valor Desejado \n')
fprintf('(Y)-> Valor Saida \n\n')
fprintf('Limiar     X1        X2        X3        X4        (D)    (Y) \n')



%-------------------------------------------------------------------------------------------------------
% INICIO
% PARTE 1   = Primeiramente Treina-se a Rede Realizando o Seguinte Procedimento antes de Iniciar o LOOP:
%-------------------------------------------------------------------------------------------------------


%-------------------------------------------------------------------------------------------------------
% Comeca o Treinamento procurando o "Combinador Linear" (U) de cada linha de entrada da matriz (Padrao de Treinamento)
% Depois, verificando o resultado deste (Cada Padrao) com a "Saida Desejada" (D),
% Acumula seu resultado em uma variavel (DifeDU) para calcular depois o EQM;...
%-------------------------------------------------------------------------------------------------------

P = M;
for L = 1:M                    % PARA L de 1 a M -> M Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(Linhas)-> M=35;
        U = 0;                     % U e a Saida do Combinador Linear -> Inicia valendo 0 para receber o resultado do
procedimento abaixo;
        K = 0;                     % K e a Coluna da Matriz -> Inicia o LOOP com valor 0 para começar desde a primeira
coluna;


        for K = 1:N                % PARA K de 1 a N -> N Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(colunas)-> N=5;
```

```
            U = U + X(L,K) * W(K);      % Faz multiplicacao e atribui a U ->
                                        % U = Saida do Combinador Linear -> Vai acumular o Resultado da Multiplicacao
(Linha x Coluna) de X e W -->
                                        %   Ele vai ser o resultado do seguinte procedimento:
                                        %       O Elemento na posicao L,M da matriz X (Vetor de Entrada) sera
                                        %       multiplicado pelo valor relativo a ele (na posicao L,M) da
                                        %       matriz W (Vetor de Pesos Sinapticos) atribuidos, no inicio,
aleatoriamente;
        end
        DifeDU = DifeDU + (D(L) - U)^2;           % Diferença entre: O Resultado da Somatoria do "Combinador Linear"(U)
dos padroes MENOS a "Saida Desejada" (D) Elevando ao quadrado;

end

EQM = (1/P)* DifeDU;             % EQM (Erro Quadratico) -> 1/P [P = Nr.Padroes (Cada Linha da Matriz de Entrada)]
MULTIPLICADO pela Diferença entre o OBTIDO(U) e o DESEJADO(D) -> DIFEDU (d(k)-u(k)^2);

EQMAtual = EQM;                 % FLAG;...
EQMAnter = 10^5;                % FLAG;...


%-------------------------------------------------------------------------------------------------------------
% Comeca a Montar o Grafico para exibir, posteriormente, o resultado do Treinamento da Rede;
%-------------------------------------------------------------------------------------------------------------
hold on;
grid on;
title('Treinamento da Rede Neural Adaline');
xlabel('EQM');
ylabel('Epoca');

%-------------------------------------------------------------------------------------------------------------
% ... Depois Realiza o Treinamento alterando o W conforme a TaxApr;
%-------------------------------------------------------------------------------------------------------------

while (abs(EQMAtual - EQMAnter)) > EpsPreci     % Inicia o Treinamento com "EQMAtual" e "EQMAnter" forcando a entrada
no LOOPING;
    EQMAnter = EQMAtual;                        % Depois EQMAnter vai receber o Valor de EQM Atual para posterior
comparacao (Quando voltar ao inicio deste LOOP);
```

```
    DifeDU=0;                   % DifeDU e zerado para receber novamente (e cada instante) o valor da Diferença de Cada
Padrao de Treinamento (Linha de X);

    for L = 1:M
        U = 0;                  % U e a Saida do Combinador Linear -> Inicia valendo 0 para receber o resultado do
procedimento abaixo;
        K = 0;                  % K e a Coluna da Matriz -> Inicia o LOOP com valor 0 para começar desde a primeira
coluna;



        for K = 1:N             % PARA K de 1 a N -> N Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(Qtde de Elementos por Padrao de Treinamento. Ou seja, o Nr.colunas)-> N=5;
            U = U + X(L,K) * W(K);      % Faz multiplicacao e atribui a U ->
                                        % U = Saida do Combinador Linear -> Vai acumular o Resultado da Multiplicacao
(Linha x Coluna) de X e W -->
                                        %   Ele vai ser o resultado do seguinte procedimento:
                                        %       O Elemento na posicao L,M da matriz X (Vetor de Entrada) sera
                                        %       multiplicado pelo valor relativo a ele (na posicao L,M) da
                                        %       matriz W (Vetor de Pesos Sinapticos) atribuidos, no inicio,
aleatoriamente;
        end

        for K = 1:N             % PARA K de 1 a N -> N Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(Qtde de Elementos por Padrao de Treinamento. Ou seja, o Nr.colunas)-> N=5;
            W = W + (TaxApr * ((D(L) - U) * X(L,1:5)));      % Realiza o Ajuste dos Pesos Sinapticos Conforme o Valor
do U
        end
        DifeDU = DifeDU + ((D(L) - U)^2);          % Diferença entre: O Resultado da Somatoria do "Combinador
Linear"(U) dos padroes MENOS a "Saida Desejada" (D) Elevando ao quadrado;

        if U >= 0                % Verifica SE o Resultado de U obtido no processamento acima pertence a alguma
Classe (Vai conferir posteriormente c/ o Padrao Desejado mencionado na matriz D); Especifica e Atribui 1/-1 a Y(L) (que
sera comparado com valor da linha da Matriz do Vetor Desejado) conforme o resultado obtido no processamneto acima;
            Y(L) = 1;               % Neste caso o U ou Y(L) Pertence a Valvula B (Y(L) vai ser comparado com o
Vetor Desejado D na mesma posicao deste);
            else
```

```
            Y(L) = -1;                  % Neste caso o U ou Y(L) Pertence a Valvula A (Y(L) vai ser comparado com o
Vetor Desejado D na mesma posicao deste);
        end
    end

    EQM = (1/P)* DifeDU;            % EQM (Erro Quadratico) -> 1/P (P=Nr.Padroes) MULTIPLICADO pela Diferença entre o
OBTIDO(U) e o DESEJADO(D) -> DIFEDU (d(k)-u(k)^2);
    EQMAtual = EQM;                 % FLAG para entrar no LOOP;

    epoca = epoca + 1;

    plot(EQM,epoca,'k+');

end


%if TR<=2;
%    print -f1;      % Imprime a Figura.1 para os 2 primeiros treinamentos;
%end


for L = 1:M
    for K = 1:N
        fprintf('%7.4f   ', X(L,K))
    end
    fprintf('%4d   %4d\n', D(L), Y(L))
end

fprintf('\n                [------------------ PESOS -------------------]')
fprintf('\n                W0        W1        W2        W3        W4')
fprintf('\nIniciais   =')
fprintf('%10.4f',WAnterior)
fprintf('\nAjustados  =')
fprintf('%10.4f',W)
fprintf('\n\nNo.Epocas  =    %d', epoca)
fprintf('\nNo.Padroes =    %d', P)
fprintf('\nValor EQM  =')
fprintf('%10.4f',EQM)
```

```
fprintf('\n\nNro.Treinamento ->  %d', TR)

fprintf('\n\n\nPressione ENTER para executar a CLASSIFICACAO...')

pause;

%-------------------------------------------------------------------------------------------------------------
% PARTE 2   = Executa-se a Classificacao com os Pesos Obtidos no Treinamento da Rede
%-------------------------------------------------------------------------------------------------------------

X = [
-1    0.9694     0.6909     0.4334     3.4965;
-1    0.5427     1.3832     0.6390     4.0352;
-1    0.6081     -0.9196    0.5925     0.1016;
-1    -0.1618    0.4694     0.2030     3.0117;
-1    0.1870     -0.2578    0.6124     1.7749;
-1    0.4891     -0.5276    0.4378     0.6439;
-1    0.3777     2.0149     0.7423     3.3932;
-1    1.1498     -0.4067    0.2469     1.5866;
-1    0.9325     1.0950     1.0359     3.3591;
-1    0.5060     1.3317     0.9222     3.7174;
-1    0.0497     -2.0656    0.6124     -0.6585;
-1    0.4004     3.5369     0.9766     5.3532;
-1    -0.1874    1.3343     0.5374     3.2189;
-1    0.5060     1.3317     0.9222     3.7174;
-1    1.6375     -0.7911    0.7537     0.5515];

[M,N] = size(X);


for L = 1:M                    % PARA L de 1 a M -> M Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(Linhas)-> M=35;
        U = 0;                     % U e a Saida do Combinador Linear -> Inicia valendo 0 para receber o resultado do
procedimento abaixo;
        K = 0;                     % K e a Coluna da Matriz -> Inicia o LOOP com valor 0 para começar desde a primeira
coluna;
```

```
        for K = 1:N                 % PARA K de 1 a N -> N Declarado anteriomente [M:N] p/tamanho Vetor de Entrada
(colunas)-> N=5;
            U = U + X(L,K) * W(K);      % Faz multiplicacao e atribui a U ->
                                        % U = Saida do Combinador Linear -> Vai acumular o Resultado da Multiplicacao
(Linha x Coluna) de X e W -->
                                        %   Ele vai ser o resultado do seguinte procedimento:
                                        %       O Elemento na posicao L,M da matriz X (Vetor de Entrada) sera
                                        %       multiplicado pelo valor relativo a ele (na posicao L,M) da
                                        %       matriz W (Vetor de Pesos Sinapticos) atribuidos, no inicio,
aleatoriamente;
        end

        if U >= 0
            Y(L) = 1;                   % Valvula B
        else
            Y(L) = -1;                  % Valvula A
        end

end

fprintf('\n\n\n---------- Resultado da Classificacao da RN - Adaline ----------\n\n')
fprintf(' X0        X1        X2        X3        X4         (Y)    Valvula\n')
for L = 1:M
    for K = 1:N
        fprintf('%7.4f   ', X(L,K))
    end
    if Y(L) == -1
        fprintf('%4d   ->   A\n', Y(L))
    else
        fprintf('%4d   ->   B\n', Y(L))
    end
end

fprintf('\n\n\nPressione ENTER para executar novo TREINAMENTO...')


pause;
end             % END para a quantidade de Treinamento -> QT;
```