```
% Proyecto de Grado
% Red Neuronal Evolutiva
% Juan Sebastian Henao Parra
% Version 4.2

%% PREPARA EL ESPACIO DE TRABAJO PARA CALCULOS

%Carpeta
%cd('C:\Users\Sebastian\SkyDrive\Proyecto de Grado\source\source 4.0 (13-03-24)')

%Inicializacion
clear all; close all; clc

%% CREA PARAMETROS QUE SE UTILIZARAN

global datos columnaSerie columnaDesempeno tamanoHorizontes numeroHorizontes

%Definicion de tamaño de los horizontes y el numero horizontes
tamanoHorizontes=100;
numeroHorizontes=10;

%Definicion del periodo a entrenar
periodoEntrenamiento=0.50;

%Definicion de la columna cargada que se usara y el numero de rezagos
columnaSerie=3;
rezagosParteLineal=[1 4 12 24 29];
rezagosParteNoLineal=[1 4 12 24 29];

%Definicion de la topologia
numeroCapasOcultas=1;
numeroNeuronas=4;

%Definicion numero de busquedas de la red neuronal artificial
numeroBusquedas=30;

%Definicion numero de neuroevoluciones
numeroNeuroEvoluciones=1;

%Tipo de funcion de costo
% 1 Para Cuadratica
% 2 Para Absoluta
% 3 Para LINEX
% 4 Para Absoluto Asimetrico
tipoFuncion=1;

%Tipo de normalizacion
% 1 Para MIN y MAX
% 2 Para Media y Desviacion Estandar
tipoNormalizacion=1;

%Algoritmmo de optimizacion
% 1 Para SIMPLEX
% 2 Para BFGS
% 3 Para Steppest Decent
```

```
 56 algoOptim=2;
 57
 58 %Definicion de archivos para los resultados
 59 seccionANN=0:numeroBusquedas:numeroBusquedas*(numeroNeuronas-1);
 60 seccionRezagosANN=0:numeroBusquedas*numeroNeuronas:numeroBusquedas*numeroNeuronas*↙
(size(rezagosParteNoLineal,2)-1);
 61 numeroANNS=numeroBusquedas*numeroNeuronas*size(rezagosParteNoLineal,2);
 62 seccionENN=0:numeroNeuroEvoluciones:numeroNeuroEvoluciones*(numeroNeuronas-1);
 63 seccionRezagosENN=0:numeroNeuroEvoluciones*numeroNeuronas:↙
numeroNeuroEvoluciones*numeroNeuronas*(size(rezagosParteNoLineal,2)-1);
 64 numeroENNS=numeroNeuroEvoluciones*numeroNeuronas*size(rezagosParteNoLineal,2);
 65
 66 %% CREA LOS MONITORES
 67
 68 multiWaitbar( 'Calculando Horizontes ANN', 1/4, 'CancelFcn', @(a,b) disp( ['Cancel↙
',a] ) );
 69 multiWaitbar( 'Calculando ANN', 2/4, 'CancelFcn', @(a,b) disp( ['Cancel ',a] ) );
 70 multiWaitbar( 'Calculando Horizontes ENN', 3/4, 'CancelFcn', @(a,b) disp( ['Cancel↙
',a] ) );
 71 multiWaitbar( 'Calculando ENN', 4/4, 'CancelFcn', @(a,b) disp( ['Cancel ',a] ) );
 72 multiWaitbar( 'Calculando Horizontes ANN', (0/(numeroHorizontes)));
 73 multiWaitbar( 'Calculando ANN', (0/(numeroANNS)));
 74 multiWaitbar( 'Calculando Horizontes ENN', (0/(numeroHorizontes)));
 75 multiWaitbar( 'Calculando ENN', (0/(numeroENNS)));
 76 h(1) = uicontrol↙
('String','Pause','Callback','uiwait','units','Normalized','Position',[0,0, 0.5, 1]);
 77 h(2) = uicontrol↙
('String','Resume','Callback','uiresume','units','Normalized','Position',[0.5,0, 0.5,↙
1]);
 78
 79 %% CARGA ARCHIVOS CON DATOS
 80 datos=xlsread('Datos','COL20');% Carga el archivo y la hoja mencionada
 81
 82 %% RED NEURONAL ARTIFICIAL
 83
 84 for numHorizonte=1:numeroHorizontes
 85
 86     for numRezagosParteNoLineal=1:size(rezagosParteNoLineal,2)
 87         rezagosParteNoLinealTemp=rezagosParteNoLineal(:,1:numRezagosParteNoLineal);
 88
 89         for numNeurona=1:numeroNeuronas
 90
 91             for numBusqueda=1:numeroBusquedas
 92
 93                 % CREA TEMPORALMENTE LAS DIFERENTES MATRICES
 94                 [dentroMuestraNoLineal, fueraMuestraNoLineal,...
 95                     dentroMuestraLineal, fueraMuestraLineal]=generarInput...
 96                     (rezagosParteNoLinealTemp,rezagosParteLineal,numHorizonte);
 97                 [dentroMuestraNoLinealN, fueraMuestraNoLinealN,...
 98                     dentroMuestraLinealN, fueraMuestraLinealN]↙
=generarInputNormalizado...
 99                     (rezagosParteNoLinealTemp,rezagosParteLineal,numHorizonte,↙
tipoNormalizacion);
100
101                 % GENERA PESOS INICIALES, PHI, BETHA y ALPHA O TRAE LOS DE
```

```
102                 % HORIZONTES ANTERIORES
103                 if numHorizonte==1
104                     [parametros]=generarPesosIniciales(numeroCapasOcultas,...
105                         numNeurona,rezagosParteNoLinealTemp,rezagosParteLineal,...
106                         dentroMuestraNoLinealN,dentroMuestraLinealN);
107                 else
108                     cuboDatosTemp=cubosDatos(:,:,numRezagosParteNoLineal,↙
numHorizonte-1);
109                     [parametros,topologia]=generarPesosRolling...
110                         (cuboDatosTemp,numBusqueda+seccionANN(1,numNeurona));
111                 end
112
113                 fullConected=ones(size(rezagosParteNoLinealTemp,2)↙
+numeroCapasOcultas+1,numNeurona);
114                 llave=generarLlave(rezagosParteLineal,rezagosParteNoLinealTemp,↙
numNeurona);
115
116                 % ENCUENTRA LOS PARAMETROS QUE MINIMIZAN LA FUNCION DE ERROR
117                 %options = optimset('Display','iter','TolFun',1e-8,'PlotFcns',↙
@optimplotfval,'HessUpdate','bfgs','LargeScale','off');
118                 options=optimset↙
('HessUpdate','bfgs','LargeScale','off','MaxFunEvals',1000000,'MaxIter',10000);
119                 [x,fval]=fminunc(@(parametros) funcionCosto(parametros,...
120                     dentroMuestraNoLinealN,dentroMuestraLinealN,tipoFuncion,...
121                     llave,numNeurona),parametros,options);
122
123                 % GUARDA LA INFORMACION EN UN CUBO
124                 [phi,betha,alpha]=separarMatrices(x,llave,numNeurona);
125                 [rmseDM,rmspeDM,maeDM,mapeDM,rmseFM,rmspeFM,maeFM,mapeFM] = ...
126                     obtenerDesempeno(x,dentroMuestraLinealN,↙
dentroMuestraNoLinealN,...
127                     fueraMuestraLinealN,fueraMuestraNoLinealN,fullConected,...
128                     llave,numNeurona);
129                 cubosDatos(numBusqueda+seccionANN(1,numNeurona),:,↙
numRezagosParteNoLineal,...
130                     numHorizonte)=[numBusqueda+seccionANN(1,numNeurona),↙
numBusqueda,numNeurona,...
131                     numRezagosParteNoLineal,numHorizonte,x,{fval},{phi},{betha},↙
{alpha},{fullConected},...
132                     {rezagosParteLineal},{rezagosParteNoLinealTemp},rmseDM,...
133                     rmspeDM,maeDM,mapeDM,rmseFM,rmspeFM,maeFM,mapeFM];
134
135                 % GENERA UNA VENTANA PARA MONITOREAR EL AVANCE DEL PROGRAMA
136                 abort = multiWaitbar( 'Calculando ANN', (numBusqueda+seccionANN(1,↙
numNeurona)+seccionRezagosANN(1,numRezagosParteNoLineal))/(numeroANNS));
137                 if abort
138                     break
139                 else
140                     pause( 1 )
141                 end
142
143             end
144         end
145
146     end
```

```
147
148     abort = multiWaitbar( 'Calculando Horizontes ANN', (numHorizonte/↙
(numeroHorizontes)));
149     if abort
150         break
151     else
152         pause( 1 )
153     end
154
155 end
156
157 %% RED NEURONAL EVOLUTIVA
158
159 for numHorizonte=1:numeroHorizontes
160
161     for numRezagosParteNoLineal=1:size(rezagosParteNoLineal,2)
162         rezagosParteNoLinealTemp=rezagosParteNoLineal(:,1:numRezagosParteNoLineal);
163
164         for numNeurona=1:numeroNeuronas
165
166             for numNeuroEvolucion=1:numeroNeuroEvoluciones
167
168                 % CREA TEMPORALMENTE LAS DIFERENTES MATRICES
169                 [dentroMuestraNoLineal, fueraMuestraNoLineal,↙
dentroMuestraLineal,...
170                     fueraMuestraLineal]=generarInput(rezagosParteNoLinealTemp,...
171                     rezagosParteLineal,numHorizonte);
172                 [dentroMuestraNoLinealN, fueraMuestraNoLinealN,↙
dentroMuestraLinealN,...
173                     fueraMuestraLinealN]=generarInputNormalizado↙
(rezagosParteNoLinealTemp,...
174                     rezagosParteLineal,numHorizonte,tipoNormalizacion);
175
176                 % GENERA PESOS INICIALES, PHI, BETHA y ALPHA O TRAE LOS DE
177                 % HORIZONTES ANTERIORES
178                 if numHorizonte==1
179                     [parametrosIniciales]=generarPesosIniciales↙
(numeroCapasOcultas,...
180                         numNeurona,rezagosParteNoLinealTemp,rezagosParteLineal,...
181                         dentroMuestraNoLinealN,dentroMuestraLinealN);
182                     [topologiaInicial]=unirTopologia(ones(size↙
(rezagosParteNoLinealTemp,2)+...
183                     numeroCapasOcultas+1,numNeurona));
184                 else
185                     cuboDatosTemp=cubosDatosNeuroEvolucion(:,:,↙
numRezagosParteNoLineal,numHorizonte-1);
186                     [parametrosIniciales,topologiaInicial]=generarPesosRolling...
187                         (cuboDatosTemp,numNeuroEvolucion+seccionENN(1,numNeurona))
188                 end
189
190                 fullConected=ones(size(rezagosParteNoLinealTemp,2)+...
191                     numeroCapasOcultas+1,numNeurona);
192                 llave=generarLlave(rezagosParteLineal,rezagosParteNoLinealTemp,...
193                     numNeurona);
194
```

```
195                 % NEUROEVOLUCION
196                 [parametrosMejorTopologia,mejorTopologia,costoMejorTopologia,...
197                     ultimaGeneracion,cuboDesempeno]=algoritmoEvolutivo↙
(parametrosIniciales,...
198                     topologiaInicial,fullConected,numeroCapasOcultas,↙
rezagosParteNoLinealTemp,...
199                     rezagosParteLineal,dentroMuestraNoLinealN,↙
dentroMuestraLinealN,...
200                     fueraMuestraNoLinealN,fueraMuestraLinealN,↙
dentroMuestraNoLineal,...
201                     dentroMuestraLineal,fueraMuestraNoLineal,fueraMuestraLineal,↙
tipoFuncion,numNeurona);
202
203                 %GUARDAR LA INFORMACION EN UN CUBO
204                 [phi,betha,alpha]=separarMatrices(parametrosMejorTopologia,...
205                     llave,numNeurona);
206                 [rmseDM,rmspeDM,maeDM,mapeDM,rmseFM,rmspeFM,maeFM,mapeFM] =...
207                     obtenerDesempeno(parametrosMejorTopologia,↙
dentroMuestraLinealN,...
208                     dentroMuestraNoLinealN,fueraMuestraLinealN,↙
fueraMuestraNoLinealN,...
209                     mejorTopologia,llave,numNeurona);
210                 cubosDatosNeuroEvolucion(numNeuroEvolucion+seccionENN(1,↙
numNeurona),:,numRezagosParteNoLineal,...
211                     numHorizonte)=[numNeuroEvolucion+seccionENN(1,numNeurona),↙
numNeuroEvolucion,numNeurona,...
212                     numRezagosParteNoLineal,numHorizonte,↙
parametrosMejorTopologia,...
213                     {costoMejorTopologia},{phi},{betha},{alpha},↙
{mejorTopologia},...
214                     {rezagosParteLineal},{rezagosParteNoLinealTemp},...
215                     rmseDM,rmspeDM,maeDM,mapeDM,rmseFM,rmspeFM,maeFM,mapeFM];
216
217                 % GENERA UNA VENTANA PARA MONITOREAR EL AVANCE DEL PROGRAMA
218                 abort = multiWaitbar( 'Calculando ENN',↙
(numNeuroEvolucion+seccionENN(1,numNeurona)+seccionRezagosENN(1,↙
numRezagosParteNoLineal))/(numeroENNS));
219                 if abort
220                     % Here we would normally ask the user if they're sure
221                     break
222                 else
223                     pause( 1 )
224                 end
225
226             end
227
228         end
229
230     end
231
232     abort = multiWaitbar( 'Calculando Horizontes ENN', (numHorizonte/↙
(numeroHorizontes)));
233     if abort
234         break
235     else
```

```
236         pause( 1 )
237     end
238
239 end
240
241 multiWaitbar( 'CloseAll' );
242
243 %% OBTIENE LAS MEJORES REDES Y LA MEJOR RED
244 %Las columnas de desempeño pueden ser 18-19-20-21
245 numHorizonteEval=10;
246 columnaDesempeno=20;
247
248 for numHorizonteEval=1:10
249 [mejorANN,matrizDatos] = obtenerMejorRed(size(rezagosParteNoLineal,2),cubosDatos↙
(:,:,:,numHorizonteEval));
250 mejorANNHorizonte(numHorizonteEval,:) = mejorANN
251 [mejorENN,matrizDatosNeuroEvolucion] = obtenerMejorRed(size(rezagosParteNoLineal,↙
2),cubosDatosNeuroEvolucion(:,:,:,numHorizonteEval));
252 mejorENNHorizonte(numHorizonteEval,:) = mejorENN
253 end
254
255 %[mejorANN,matrizDatos] = obtenerMejorRed(size(rezagosParteNoLineal,2),cubosDatos↙
(:,:,:,numHorizonteEval));
256 %[mejorENN,matrizDatosNeuroEvolucion] = obtenerMejorRed(size(rezagosParteNoLineal,↙
2),cubosDatosNeuroEvolucion(:,:,:,numHorizonteEval));
257
258 %% GUARDA LOS DATOS EN UN ARCHIVO
259 formatOut = 'mm-dd-yy';
260 datestr(now,formatOut);
261 filenameNN = sprintf('Datos guardados con %d neuronas el', numeroNeuronas);
262 filenameDD = sprintf(datestr(now,formatOut));
263 filename = strcat(filenameNN,'_', filenameDD);
264 save(filename);
265
266 %% GENERA GRAFICAS
267 generarGraficas(mejorANN,mejorENN,rezagosParteNoLineal,rezagosParteLineal,↙
numHorizonte)
268
```