BP神经网络代码

function [Y,Xf,Af] = myNeuralNetworkFunction(X,Xi,~)

%MYNEURALNETWORKFUNCTION neural network simulation function.

%

% Generated by Neural Network Toolbox function genFunction, 09-Sep-2023 23:06:38.

%

% [Y,Xf,Af] = myNeuralNetworkFunction(X,Xi,~) takes these arguments:

%

% X = 1xTS cell, 1 inputs over TS timesteps

% Each X{1,ts} = 6xQ matrix, input #1 at timestep ts.

%

% Xi = 1x2 cell 1, initial 2 input delay states.

% Each Xi{1,ts} = 6xQ matrix, initial states for input #1.

%

% Ai = 2x0 cell 2, initial 2 layer delay states.

% Each Ai{1,ts} = 10xQ matrix, initial states for layer #1.

% Each Ai{2,ts} = 6xQ matrix, initial states for layer #2.

%

% and returns:

% Y = 1xTS cell of 1 outputs over TS timesteps.

% Each Y{1,ts} = 6xQ matrix, output #1 at timestep ts.

%

% Xf = 1x2 cell 1, final 2 input delay states.

% Each Xf{1,ts} = 6xQ matrix, final states for input #1.

%

% Af = 2x0 cell 2, final 0 layer delay states.

% Each Af{1ts} = 10xQ matrix, final states for layer #1.

% Each Af{2ts} = 6xQ matrix, final states for layer #2.

%

% where Q is number of samples (or series) and TS is the number of timesteps.

%#ok<\*RPMT0>

% ===== NEURAL NETWORK CONSTANTS =====

% Input 1

x1\_step1.xoffset = [0.926;31.298;0.632;0.252;6.066;3.012];

x1\_step1.gain = [0.00675981694415715;0.0016205157291308;0.01078033451378;0.016852181093538;0.0033435590514323;0.00393604710661177];

x1\_step1.ymin = -1;

% Layer 1

b1 = [1.488230975291968905;-1.7010329570372146879;-1.7938416534616841158;0.37163788820108512034;-0.30144517727043834698;0.26527669923702945631;0.40425733743891645311;-3.9730093366299961133;2.3327046783250788842;0.58580362266937435756];

IW1\_1 = [0.28542414215341460615 2.4874620116310497409 0.6023606048580868455 -0.0028533351956805952376 -0.76310349654968478994 -0.88716289381410096127 -0.46814314084097996549 -2.3019714467419460924 0.76823693769103140738 0.59526270454856300418 1.2625088431434590142 0.80173218928845635389;0.5660142640391792046 -3.2137563113730465147 0.13259862954877382757 1.9340149305468243845 -1.002384614529113982 2.9316680257116844999 1.2898653236445651871 -0.38541810736968817874 -1.5136169408722917051 0.43828649654430740235 -1.5388196296996521362 -1.4319882272248649624;-0.090515971591818261688 -3.1513025017640283743 0.50508741670027457182 0.37005878740451370046 0.23491273961490991007 0.4491889464903843221 0.33339995343901274172 1.2067908160840983278 -0.59515833150946961805 -0.16559984829347471069 -0.99352331250338465907 -0.16823531726330659408;-2.5288287157520654702 -2.0813719313255938381 0.19460292941435552994 -0.70687777072485824537 2.6437031934258734545 1.8364004185143163461 -1.6295257795923046018 0.41655513516243075633 -0.046789439579059635688 0.17853729486465583221 1.0625153648936884743 1.2058216000513355404;-0.60623928914250257982 0.19200304899728098951 0.38222986204172643143 -0.19238341459914812259 -1.4049613708774273402 -0.030277995670679333506 -0.49222874388091875097 0.61593542471008155825 0.00077743305439205347834 0.3333400977463704673 -0.49862453595118350069 0.96855327071800179084;0.49489105486446494098 -0.12698632858844721705 -0.32988765530102837076 0.018706288604083590177 0.25446600663871093984 0.48743420164818607176 0.27274106033689948791 -0.61996297358245622267 0.054843794422113172071 -0.10139934737844893964 0.034585878288883271925 -0.051092568302847002348;0.72501754570180165782 -2.595455692315326246 -1.0948408878692632573 -0.042402384787049572645 1.3795142430387059562 3.2667659403663380502 -0.98785169548838225317 0.095283868851211150908 -0.15728128981733027758 1.6849448083091818074 0.79729277207405568895 0.82857840075162603188;2.1384959737608411068 0.20502848101179332874 -0.21379565885234230738 0.0014169415220258024801 -2.457656114234886946 -3.2485172178544505783 2.0333583546521252217 -3.7505489058899894594 -0.11159966902808372835 -0.55708588478547949041 -2.727509042608381673 0.48110016556302925839;0.52489207867900367077 -5.2575161235568037554 -1.6834089579356745769 -3.286200919934943343 -3.1898350493550537976 4.9251461178234503535 0.74258756688095184373 0.96744523550524030853 2.8538662917480150938 -4.0293934533528386055 -4.7938980998908897746 -4.595163265709421907;0.18524319846074199791 -0.25030964891152629459 0.37194513092943365384 -0.20453849634316284556 -0.54635236144405019409 0.55015588938803139385 -0.26203246956053166672 -0.35247066548545147313 0.22234502186401494672 0.08363378654584656402 -0.30547460782554602954 1.1117736095748507186];

% Layer 2

b2 = [-1.2289348616962887384;-1.1202725992630044871;-1.1978539141949626767;-0.86014019707310174212;-1.1896588407068933169;-1.3576961404898955088];

LW2\_1 = [-0.46314752930722691282 0.11768441795638540093 -0.50488025680516634974 -0.18500246839973125135 -1.0988398257958476023 -1.4395188455498564295 0.70330893850044640114 0.13394789365179041174 0.9075558070419430523 1.2237254442310392388;-0.50423228470656888955 0.21595476323492082682 -0.83792319596064146481 -0.066037102834457808709 -1.0367034642945160972 -1.9301552909185895057 0.37239261485963170983 0.14887922132303693656 0.38352513628975820614 1.1560123423292103517;-0.22290589760799101038 0.21009927089209087803 -0.47698437554417227879 -0.10140423166646114761 -1.5348431004820717316 -2.5030550171438208373 0.49001827142835219231 0.22191138579947308762 0.76220483339965128255 1.6724933835803674764;-0.030981310606274767988 0.38412038479796978274 -0.51123337844904070959 0.039342044606983775368 -0.61974289476526811526 -1.5479441128527848814 0.75056281812173342161 0.22233658078484591636 0.71372249233986384542 0.40660959915528754882;-0.31298054038680156452 0.13649256852006605434 -0.50691482468452564447 0.056164135170275034104 -1.0861388633601822828 -1.4260845260870971263 0.64377883634825905368 0.18647447442718678712 0.86679062095262071619 0.85579974145559256637;-0.47770091806733766537 0.22695251073382063933 -0.70807048357169066755 0.034474003919214589453 -0.83822922138450728369 -1.2864816923006745508 0.59746903828310993223 0.17333718001974310341 0.8328457636068541392 1.231318925418159127];

% Output 1

y1\_step1.ymin = -1;

y1\_step1.gain = [0.00675981694415715;0.0016205157291308;0.01078033451378;0.016852181093538;0.0033435590514323;0.00393604710661177];

y1\_step1.xoffset = [0.926;31.298;0.632;0.252;6.066;3.012];

% ===== SIMULATION ========

% Format Input Arguments

isCellX = iscell(X);

if ~isCellX

X = {X};

end

if (nargin < 2), error('Initial input states Xi argument needed.'); end

% Dimensions

TS = size(X,2); % timesteps

if ~isempty(X)

Q = size(X{1},2); % samples/series

elseif ~isempty(Xi)

Q = size(Xi{1},2);

else

Q = 0;

end

% Input 1 Delay States

Xd1 = cell(1,3);

for ts=1:2

Xd1{ts} = mapminmax\_apply(Xi{1,ts},x1\_step1);

end

% Allocate Outputs

Y = cell(1,TS);

% Time loop

for ts=1:TS

% Rotating delay state position

xdts = mod(ts+1,3)+1;

% Input 1

Xd1{xdts} = mapminmax\_apply(X{1,ts},x1\_step1);

% Layer 1

tapdelay1 = cat(1,Xd1{mod(xdts-[1 2]-1,3)+1});

a1 = tansig\_apply(repmat(b1,1,Q) + IW1\_1\*tapdelay1);

% Layer 2

a2 = repmat(b2,1,Q) + LW2\_1\*a1;

% Output 1

Y{1,ts} = mapminmax\_reverse(a2,y1\_step1);

end

% Final Delay States

finalxts = TS+(1: 2);

xits = finalxts(finalxts<=2);

xts = finalxts(finalxts>2)-2;

Xf = [Xi(:,xits) X(:,xts)];

Af = cell(2,0);

% Format Output Arguments

if ~isCellX

Y = cell2mat(Y);

end

end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function

function y = mapminmax\_apply(x,settings)

y = bsxfun(@minus,x,settings.xoffset);

y = bsxfun(@times,y,settings.gain);

y = bsxfun(@plus,y,settings.ymin);

end

% Sigmoid Symmetric Transfer Function

function a = tansig\_apply(n,~)

a = 2 ./ (1 + exp(-2\*n)) - 1;

end

% Map Minimum and Maximum Output Reverse-Processing Function

function x = mapminmax\_reverse(y,settings)

x = bsxfun(@minus,y,settings.ymin);

x = bsxfun(@rdivide,x,settings.gain);

x = bsxfun(@plus,x,settings.xoffset);

end

stata回归代码

import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\水生根茎类求加权成本加成定价.xlsx", sheet("Sheet1") firstrow

regress 成本加成定价 销量千克

estat hettest,rhs iid

clear

import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\花菜类求加权成本加成定价.xlsx", sheet("Sheet1") firstrow

. regress 成本加成定价 销量千克

estat hettest,rhs iid

regress 成本加成定价 销量千克,robust

clear

. import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\花叶类求加权成本加成定价.xlsx", sheet("Sheet1") firstrow

. regress 成本加成定价 销量千克

estat hettest,rhs iid

import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\辣椒类求加权成本加成定价.xlsx", sheet("Sheet1") firstrow clear

. regress 成本加成定价 销量千克

estat hettest,rhs iid

. import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\茄类求加权成本加成定价.xlsx", sheet("Sheet1") firstrow clear

. regress 成本加成定价 销量千克

estat hettest,rhs iid

import excel "D:\HuaweiMoveData\Users\volcano\Desktop\问题二成本加成预处理数据\

> 问题二\食用菌求加权成本加成定价.xlsx", sheet("Sheet1") firstrow clear

. regress 成本加成定价 销量千克

estat hettest,rhs iid

twoway (scatter 成本加成定价 销量千克 ) (lfit 成本加成定价 销量千克 )