An INTERESTING System

Typhoon Damage Cost Estimation through Artificial Neural Network

Karlos Castillo, Aldwin Del Rosario, Adrian Jarabelo
Department of Electronics and Communications Engineering
Gokongwei College of Engineering, De La Salle University
Manila, Philippines

***Abstract* — The Integral Emergency Response Tool for Identifying Natural disaster Gravity is a system used to estimate the cost of damages brought about by typhoons that sweep the Philippines each year. This program is implemented using artificial neural network where data regarding the characteristics of typhoons, the characteristics of the area to be affected, and cost of damages are used to compute the cost of damages to the said area for better disaster mitigation and emergency response.**

Keywords—artificial neural network; typhoon; cost estimation; disaster mitigation; emergency response; machine learning

I. INTRODUCTION

Annually, the Philippine archipelago is affected by an average of 19 to 20 tropical cyclone systems. Of these, nine to ten make landfall, bringing damage to landlocked provinces as well as battering the coastal towns with strong winds, heavy rains, and storm surges. [1]

The country is prone to such natural disasters due to its location on the equator. The Pacific Ocean waters near the archipelago are exposed more to the sun due to the axial tilt of the Earth and therefore have higher temperatures that are favorable to the formation of typhoons. The ideal water temperature for typhoons to form is 28°C and the temperature of the water around the Philippine archipelago is normally above that temperature [2].

The Philippines is also prone to experience disastrous effects of these typhoons due to the topography of the country. According to World Bank estimates, more than 60 percent of the population lives along coastal regions in the country [2]. The low coastlines provide little protection against the strong winds and storm surges that may occur when the high tide coincides with such strong winds that the water is pushed further into the coast. Additionally, due to poverty in the country, most houses are made of light materials [3] like thin wood boards, galvanized iron roof and even cardboard and scrap. These light material houses provide no shelter against the effects of typhoons. Deforestation also contributes to the gravity of natural disasters. The reduction of the number of trees and plants in the mountains is directly proportional to an increased in severity of floods and landslides due to the lack of roots that would hold down the soil and absorb excess water brought about by the typhoon.

In light of these disastrous effects, it is important that the Philippines increase its disaster mitigation and response system. After experiencing crisis after crisis brought about by typhoons, the Philippines has been steadily improving its disaster mitigation and response action that a zero casualty figure is an achievable goal even if it involves a strong typhoon.

Human lives are not the only ones that are affected when calamity strikes a populated area. The physical infrastructure that is the backbone of civilization is also being damaged by strong winds and heavy rain caused by typhoons. The destruction of such asset prolongs the suffering of the people affected since they could not easily resume their normal lives after the storm has passed.

It is necessary to estimate the cost of damages before a typhoon strikes a locality so that necessary preparations may be easily organized and observed deficiencies may be eliminated or minimized so that at the event when the typhoon strikes the actual cost of damages may be lessened to the point that it is possible to recover in a small amount of time.

This paper seeks to solve this need for a damage cost estimation tool to aid the Filipino people in their annual struggle against typhoons by determining the cost that would be needed to rebuild the lives of the people affected by the typhoon.

II. CONCEPT THEORY

Artificial neural network is an information processing system modeled after the biological neural network present in living organisms such as humans and animals. Interconnected elements act like the neuron cells that compose the brain and the nervous system where data from one element to the other as the system solves a specific problem.

Artificial neural networks learn through example. The user feeds training data to the system for it to learn the process that the user needs it to do. Once the system acquires the skills necessary for it to operate, validation data is fed to it to determine the deviations between the desired and actual output. Once the errors are detected, the system is fine tuned to reduce these deviations to a minimum. Finally, the system is tested in real world applications through the use of test data.

III. METHODOLOGY

The artificial neural network is trained by the researchers to estimate the cost of damages an incoming typhoon will inflict on a locality. Two of the most destructive typhoon characteristics, wind speed and amount of rainfall, and the partition of the area according to use are the input data that is fed to the system. The inputs then pass through the network, where it will gain weights after passing through different connections between neurons. The output of this network will be the cost of damage in the typhoon will cause in the locality.

The program utilized in this project is Matlab which has a built-in function that provides functions for creating, training, and simulating neural network called the Neural Network toolbox. The code used in this program comes from the examples listed in the documentations for the toolbox online. Some few modifications were done to fit the code into the objectives of the project.

IV. DATASET

The Philippine Atmospheric, Geophysical, and Astronomical Services Administration is the government agency that is tasked to monitor the weather condition throughout the country and therefore keeps track of the typhoons that enter the Philippine Area of Responsibility. They also log the characteristics of the typhoon as they monitor it. However, the researchers opted to use the website Digital Typhoon by the Japanese National Institute of Informatics as the source for the wind speeds of the typhoon since the data is organized in a user-friendly way. Meanwhile, rainfall data was gathered from the Weather Philippines website.

This paper deals about the other casualties that are difficult to evacuate and these are the residential and industrial structures that people utilize everyday such as houses, business establishments, offices, and farms. The data regarding the utilization of areas for agricultural, industrial, residential purposes comes from the National Statistics Office where the utilization of the area is detailed.

Data regarding the cost of damages caused by typhoons come from the situation reports compiled by the National Disaster Risk Reduction and Management Council.

Since there are many factors that contribute to the gravity of the post-calamity situation, the researchers do not expect to achieve a very high accuracy rate.

Parameters:

- X_1 – The first parameter is the wind speed of the typhoon at the time of impact in knots.
- X_2 – The second parameter is the amount of rainfall of the typhoon at the time of impact in millimeters.
- X_3 – The percentage of the area of the locality dedicated to industrial purposes.
- X_4 – The percentage of the area of the locality dedicated to agricultural purposes.
- Y – The estimated cost of damages the typhoon will bring to the locality when it affects the area in millions of Philippine peso.

V. NEURAL NETWORK PROGRAMMING

Creating the neural network involves first feeding the matrix containing the wind speed and amount of rainfall of the typhoon as well as the percentages of land dedicated to industry and agriculture of the area affected. A second matrix containing the respective cost of damages wrought by the typhoon in the respective area is input. Using the function *nnstart*, the Neural Network Wizard was launched to aid the researchers in the development of a neural network.

The matrix inputs for the training data *train_inputs* contains the characteristics of the typhoon and the area affected by it. Another matrix input which contained the cost of damages is called *train_outputs*. These were input by the researchers into the Neural Network Wizard when they were prompted to do so.

The researchers then chose to create 20 nodes since the project can be built with the use of such small number of hidden layer neurons.

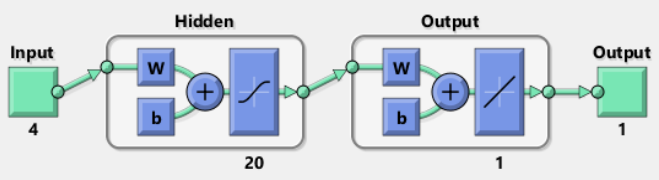

Fig. 1 The Neural Network generated on Matlab. Note that the input and output node numbers are at zero since the network is not yet trained. The four inputs are the typhoon and area characteristics while the one output is the cost of damages.

The data was divided into three parts and the first part will be the training data which is set to be 80% of the data. The second and third parts are the validation and test data with each having 10% of the data. Therefore, the training data will amount to 20 while the validation data and test data will be two each. These numbers were rounded up since there cannot be any fractional parts of the data. The program itself divided the data into parts.

The network training is initialized at the next step where the training algorithm is chosen. The researchers used the Levenberg-Marquardt Method which minimizes the global error function, a multivariate function dependent on the weights of the nodes in the network, by finding the summation of the squares of the nonlinear functions [4].

Mean Square Error meanwhile is the set performance checking algorithm used by the program. This algorithm checks how close vertically the data is to the fitted line. The difference between the two is calculated and then squared to account for negative differences. The smaller the mean square error, the closer the data is to the fitted line [5].

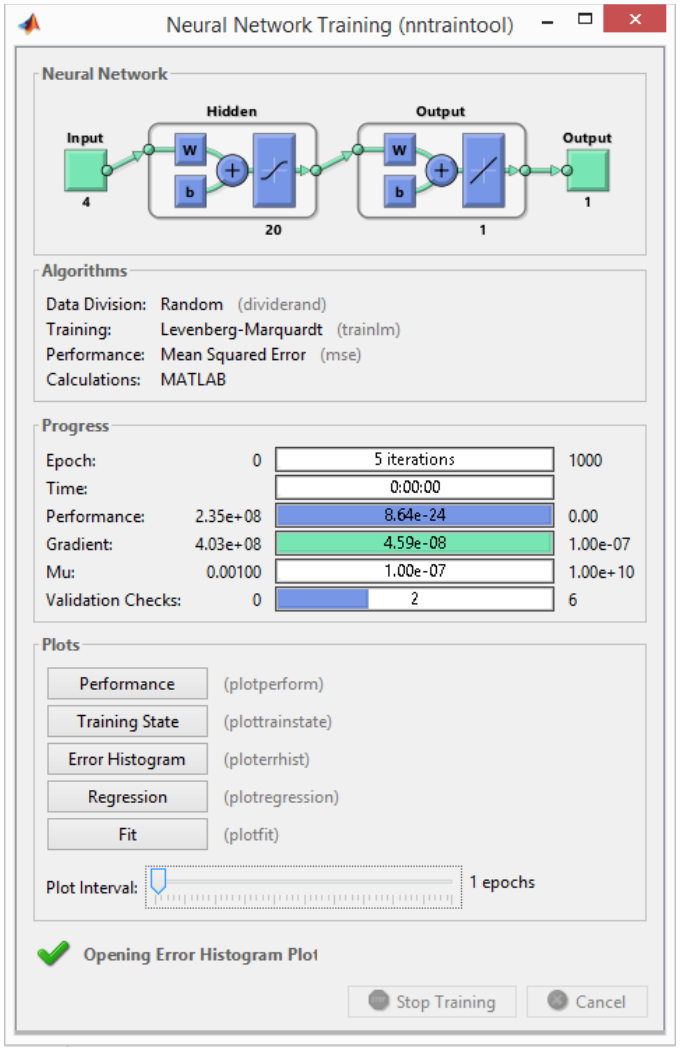

Fig. 2 The Neural Network Training window from Matlab. At this state the training is complete and the plots may now be seen.

After some time, the neural network is now trained and the different graphs regarding the performance of the network may be extracted from the program.

VI. ANALYSIS OF RESULTS

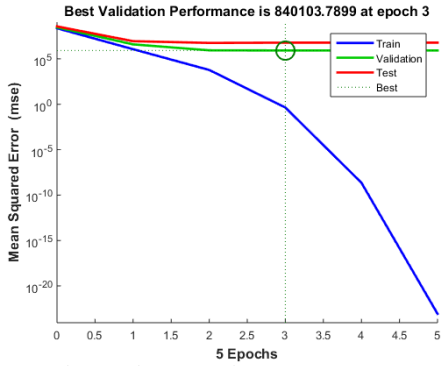

Fig. 3. The Plot of the Performance of the trained neural network. Plotted on the graph are lines for the training data, validation data, and test data.

The above graph (Figure 3) shows the performance of the trained neural network. The blue line represents the training data, the red line the test data, and the green line the validation data. The blue line decreases in a larger rate than the other two lines due to it being composed of more data. The two other graphs are similar and this means that no over fitting occurred in the process. The best validation performance is indicated by the intersection of the two dotted lines and emphasized by the circle. The rest of the graph at the right side of the vertical line that crosses this point has errors that are beyond the acceptable limit that they are no longer considered.

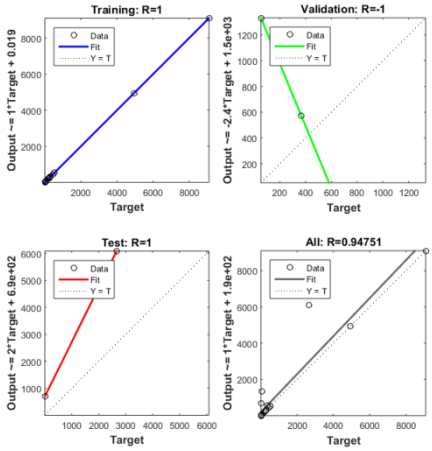

Fig.4. The Regression plots for the trained neural network.

The regression plots show how the outputs of the network are close to the target values. It can be seen that the output lie almost comfortably along a straight line that they all have a good fit. On the graph in the lower right however, there are notable data outliers that cannot be covered by the line. Three of the four graphs indicate and R close to one and this means that there is an exact linear relationship between the targets and the network output. The upper right graph has an R of -1 and is most likely an error.

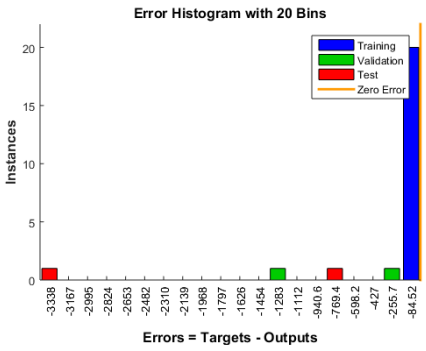

Fig.5. The Error Histogram for the neural network.

This is the histogram of the network. The blue bar shows the training data while the red bar shows the test data. The green bar shows the validation data. The orange line indicates the location of where the zero error is. The red bar on the extreme left shows great errors between the targets and outputs while the blue bar has small errors since it is close to the zero errors line.

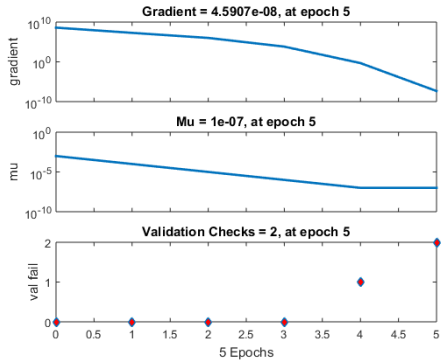

Fig.6. The Training State graphs for the network.

The training state displays the training statistics for the network. Gradient is the value of the back propagation gradient in each iteration and is therefore expected to decrease since regularization reduces the theta of the function by the gradient for system optimization. The gradient value indicates that the function has reached the local minimum. The validation fails meanwhile indicate the increase of the mean square error. The graph above indicates that there are only a few fails in this network. Mu is a factor present in the Levenberg-Marquardt method and in this network it decreases.

VII. CONCLUSION AND RECOMMENDATIONS

The Integral Emergency Response Tool for Identifying Natural disaster Gravity system is a tool designed to assist in typhoon disaster mitigation and response by estimating the cost of damages a typhoon would bring to a populated area given the characteristics of both the typhoon and the affected area. The needed parameters were fed to an artificial neural network and the estimated cost of damages was obtained as an output.

Data from previous typhoons obtained from the various government agencies responsible were used as training data, validation data, and testing data.

The neural network was trained successfully with a high accuracy. There are however significant errors that may be attributed to the small amount of data used to train this network.

The researchers would like to recommend increasing the capability of this system through the use of more parameters such as the strength of the structures in the area concerned and the parameters of the surrounding waters so that the actual estimated cost of damages would be as close as possible to reality. Also, this system is adaptable to other calamities such as earthquakes, volcanic eruptions, and tsunamis with only minimal modification required. Also, the researchers would like to recommend the increase of the scope of this system to also accommodate natural disasters that may occur in unison with a typhoon so that this system can truly live up to its name as a system to estimate the cost of damages of natural disasters in general and not only of typhoons.

ACKNOWLEDGMENT

The authors would like to thank Engr. Melvin Cabatuan for the support he has given through his lectures and discussions in the Machine Intelligence (CPELEC1) course and for his frank yet needed remarks that inspired the researchers to strive for a better result. The authors would also like to thank their fellow student, Mr. Ryan Joshua Liwag, for inspiring the title of this project, as well as their other classmates in the CPELEC1 class for their invaluable support to the researchers in technical as well as in emotional matters.

REFERENCES

[1] E. De Vera, “20 cyclones visit PH annually,” Manila Bulletin, 7 November 2013. [Online]. Available: http://www.mb.com.ph/20-cyclones-visit-ph-annually/. [Accessed 16 November 2015].

[2] D. Vergano, “5 Reasons the Philippines Is So Disaster Prone,” National Geographic, 11 November 2013. [Online]. Available: http://news.nationalgeographic.com/news/2013/11/131111-philippines-dangers-haiyan-yolanda-death-toll-rises/. [Accessed 16 November 2015].

[3] S. Brown, “The Philippines Is the Most Storm-Exposed Country on Earth,” Time, 11 November 2013. [Online]. Available http://world.time.com/2013/11/11/the-philippines-is-the-most-storm-exposed-country-on-earth/. [Accessed 16 November 2015].

[4] M. Lourakis, “A Brief Description of the Levenberg-Marquardt Algorithm Implmented by levmar,” Foundation for Research and Technology – Hellas (FORTH), 11 February 2005. [Online]. Available http://users.ics.forth.gr/~lourakis/levmar/levmar.pdf. [Accessed 16 November 2015].

[5] “What are Mean Squared Error and Root Mean Squared Error?,” Vernier, n. d. [Online]. Available http://www.vernier.com/til/1014/. [Accessed 16 November 2015].

Appendix A
Dataset

Year	Typhoon name	Affected Area	Wind Speed (knots)	Amount of Rainfall (mm)	% Industrial Area	% Agricultural Area	Cost of Damages (millions of Php)
2014	Kajiki/Basyang	Surigao del Norte	35	300	22	57	258.3914023
	Rammasun/Glenda	Albay	80	300	25	51	9100
		Batangas	70	200	27	53	4949
	Kalmaegi/Luis	Cagayan	45	250	22	62	12.31946215
		Ilocos Norte	45	250	23	60	0.606529
	Fung-wong/Mario	Cagayan	65	200	22	62	365.63847
	Hagupit/Ruby	Eastern Samar	90	250	19	59	301.162911
		Samar	90	250	18	61	262.083045
		Batangas	45	112	27	53	17.611663
	Jangmi/Seniang	Surigao del Sur	35	175	18	57	31.90596729
		Bohol	40	200	20	58	52.504807
		Negros Occidental	35	200	21	61	0.48904
2013	Rumbia/Gorio	Eastern Samar	35	250	19	59	31.220989
		Sorsogon	35	300	18	57	44.7821
		Batangas	35	250	27	53	11.7451
	Cimaron/Isang	Cagayan	35	200	22	62	36.2821
	Utor/Labuyo	Aurora	95	350	17	58	430.90221
		La Union	75	350	24	53	14.11207
	Nari/Santi	Aurora	70	400	17	58	59.283222
		Zambales	65	320	19	66	3.845216
	Krosa/Vinta	Cagayan	70	52	22	62	182.93908
		Ilocos Norte	70	52	23	60	235.0319105
	Haiyan/Yolanda	Leyte	110	52	20	57	2665.705225
		Iloilo	90	52	22	64	537.8568

Table 1. The data input for the neural network. The program was the one who separated the data into the test data, validation data, and training data therefore the researchers could not identify which is which.

Appendix B
Neural Network Code generated by Matlab

```
function [Y,Xf,Af] = INTERESTING_nn(X,~,~)
% INTERESTING_nn neural network simulation function.
%
% Generated by Neural Network Toolbox function genFunction, 03-Dec-2015 21:29:36.
%
% CPELEC1 Project
% Karlos Castillo, Aldwin del Rosario, Adrian Jarabelo
% INTegral Emergency RESponse Tool for Identifying Natural disaster Gravity
% (Typhoon Damage Cost Estimation through Artificial Neural Network)
%
% [Y] = INTERESTING_nn (X,~,~) takes these arguments:
%
%   X = 1xTS cell, 1 inputs over TS timsteps
```

```
%   Each X{1,ts} = 4xQ matrix, input #1 at timestep ts.
%
% and returns:
%   Y = 1xTS cell of 1 outputs over TS timesteps.
%   Each Y{1,ts} = 1xQ matrix, output #1 at timestep ts.
%
% 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 = [35;52;17;51];
x1_step1_gain = [0.0266666666666667;0.00574712643678161;0.2;0.133333333333333];
x1_step1_ymin = -1;

% Layer 1
b1 = [2.9532883512583625;2.6419993694383046;-2.2187938403392402;-2.0424042157661093;-
1.700207848019845;1.1709114301252741;1.2810559213900794;0.66157171575050655;0.53339982975947564;
-0.4779257933332543;0.077071996547465244;0.26962029558460415;-
0.85218740740327803;1.1840746034642839;1.4050431288570004;1.7734188258314538;-
1.8541472419152634;-2.0495033404396534;-2.7037319188867013;-2.7971389230424677];
IW1_1 = [-0.88945433111196859 0.28443636097420671 -2.3241065392309563 1.5912141390861185;-
2.4288139492203347 -0.31065264684490856 -1.0629395104600627
1.2756783228898694;1.1149253100394707 1.0751367362242055 2.4544341694459071 -
0.91654767171707308;2.2917891764475562 0.045262425222588561 1.6452651843138053 -
0.98455690680873398;1.134306518132064 2.3546220029185645 0.86687043179997614 -
1.0605531810914146;-1.5742713240467625 -1.4428936225872222 1.9338108918114254
1.1947348129729514;-0.94356886267216811 -0.61473121257384722 2.6394856886454345 -
0.3037139339906394;-1.9100477570461623 0.75308685253745333 -2.0395366084654545 -
0.42807051793946832;-1.8533285155915196 -2.0671987593912684 1.0291503905077732 -
0.37576251069465783;-2.205458713317241 0.60586719207211259 0.006561410797317058
1.9309100768056044;1.7164882947279101 -1.5894125500003744 -1.3343038001085532 -
1.0773024381278993;1.3403375166110989 0.43196143452402247 1.9517587572730732
1.6798363397224663;-0.83401224314951239 1.4872936351534023 0.038739487375355854 -
2.4333728892094637;2.0444983979721676 -0.11030215342861478 1.8325788195119159 -
0.96898698439983921;1.3162944991974286 -0.31708155292459311 -2.2593888080512277
1.3279732410967813;0.72639414018256532 -2.2792741358603101 -1.7426598965138942 -
0.61353196143673594;-0.6606391117409447 -2.7577295786262992 0.87165321195167322
0.68956411278226992;-1.0347661764334866 1.3648889891152962 -2.5880351104801194 -
0.31700238604333258;-1.6357661322786399 1.214792746600259 0.55069606979425578 -
2.00802371871282;-1.8483015881964029 -1.9342741184291334 -1.3745064855060649
0.84073696941376153];

% Layer 2
b2 = 0.27566221120896356;
LW2_1 = [-0.24318358790968109 -0.46193989613478897 -0.056582301944425602 0.68836207576894359
0.72844779026344508 0.35907824785951503 -0.56656351202379174 -0.44244993984150471
0.17153120270533831 0.64397317573333435 0.39301802977179784 -0.5133009456438411 -
0.25396640564497314 0.015480877915486701 -0.090332721534104113 0.42578252967334257 -
0.21184071333794591 -0.37153618233685887 0.21170182509803129 -0.44421845635146412];

% Output 1
y1_step1_ymin = -1;
y1_step1_gain = 0.000219792031548913;
y1_step1_xoffset = 0.48904;

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

% Format Input Arguments
isCellX = iscell(X);
if ~isCellX, X = {X}; end;

% Dimensions
```

```
TS = size(X,2); % timesteps
if ~isempty(X)
    Q = size(X{1},2); % samples/series
else
    Q = 0;
end

% Allocate Outputs
Y = cell(1,TS);

% Time loop
for ts=1:TS

    % Input 1
    Xp1 = mapminmax_apply(X{1,ts},x1_step1_gain,x1_step1_xoffset,x1_step1_ymin);

    % Layer 1
    a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*Xp1);

    % Layer 2
    a2 = repmat(b2,1,Q) + LW2_1*a1;

    % Output 1
    Y{1,ts} = mapminmax_reverse(a2,y1_step1_gain,y1_step1_xoffset,y1_step1_ymin);
end

% Final Delay States
Xf = cell(1,0);
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_gain,settings_xoffset,settings_ymin)
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_gain,settings_xoffset,settings_ymin)
x = bsxfun(@minus,y,settings_ymin);
x = bsxfun(@rdivide,x,settings_gain);
x = bsxfun(@plus,x,settings_xoffset);
end
```