POWER SYSTEM CONGESTION PREDICTION USING NEURAL NETWORK ALGORITHMS

A Thesis By

Joshua M. Newell

Bachelor of Science, Wichita State University, 2011

Submitted to the Department of Electrical Engineering and Computer Science
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

May 2018

POWER SYSTEM CONGESTION PREDICTION USING NEURAL NETWORK ALGORITHMS

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science, with a major in Electrical Engineering.

____

Visvakumar Aravinthan, Committee Chair

____

Krishna Krishnan, Committee Member

____

Abu Abuduzzaman, Committee Member

DEDICATION

To my wife, parents and sons for their guidance, support, love, and patience. Without all of them, I would not have been able to complete such a project.

ACKNOWLEDGEMENT

I would like to show my appreciation for my advisor Dr. Visvakumar Aravinthan for his support and guidance through my period as a graduate student. He has been actively helpful in pushing me towards my goals and always making sure that I was moving in the right direction. I enjoyed working with you through my research and experimenting.

I would like to express my gratitude to Krishna Krishnan and Abu Abuduzzaman for being committee members on my Thesis, thank you for being part of this.

I cannot fully explain how much support and encouragement my wife, Brianna Newell has given to me throughout my work on my master’s degree. None of this would have been possible without her help.

.

ABSTRACT

As the number of electronic devices increases, there has become an increase of congestion on supporting power lines in a power system. This congestion, can increase the cost of power transmission due to the necessity of building new lines and can increase the cost of power in peak congestion time, depending on geographical location and transmission lines that are feeding the nearest hub. This rising power congestion also causes damage to power lines if they run too close to their limitations frequently.

This thesis is exploring some of the available algorithms and tools that can be used to predict the congestion of a power system, which could then be coupled with a power system and monitored real time. This would enable the system management to make better decisions on future development/improvements to the system. The analysis methods presented are based upon a modified 24 bus IEEE Reliability Test System, and a pre-defined system load curve to simulate a real-time system. This load model is based upon the setup in the appendix of IEEE (RTS) [14].

This research led to finding a neural network algorithm approach to train an algorithm using past data and then used simulated real-time input to predict the congestion. The idea was for this prediction to be applied to a more dynamic system to see how well it would be able to find where the congestion is.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
DCOPF	DC Optimal Power Flow
FA	Firefly Algorithm
FACTS	Flexible Alternating Current Transmission System
GRBF	Growing Radial Basis Function
IPFC	Interline Power Flow Controller
LUF	Line Utilization Factor
NCP	Nodal Congestion Price
PSAT	Power System Analysis Toolbox
RTS	Reliability Test System

CHAPTER 1

INTRODUCTION

Congestion has become a big problem in power systems, due to many factors including the increasing number of electronic devices that are being supplied power by these systems. Power operators are always working to make the systems more efficient and to not have to keep extra generators up and running when not in use, using up resources. Power operators will only schedule the number of generators required with a few on reserve and no more. This becomes a problem when there are unexpected spikes in power consumption.

1.1 Managing Congestion

There have been many methods tested to find out how to alleviate congestion altogether or to make it more manageable. For example; one research team attempted to alleviate congestion by rescheduling the generation of a power system based on a Firefly Algorithm [1]. Another group used Flexible Alternating Current Transmission System devices that were strategically placed based on locations found using an algorithm to increase the stability of a system and in-tern decrease the congestion of a system. These along with some others will be covered more in depth in the document review section.

1.2 Predicting Congestion

There is a need to predict when and where the congestion will likely happen in a system. This information will enable power system operators to have a more heightened sense of how the system will behave. This sort of predictor can be based upon characteristics of past and currently collected data such as: day of year, month, time of day, generator location, line location, generator power output, and line capacity. With this information, power loads on the lines in a system can be predicted within a certain degree of accuracy, therefore enabling better decisions to be made regarding where to install new generators, new distribution lines, and even where to install various renewable energy sources such as wind generators or solar panels.

If there is a means to detect congestion and possibly even predict congestion, it could help engineers make a more educated guess on how to modify the power system to prevent future congestion and possible outages due to congestion. For example, if a specific area of the power system is congested on a frequent basis, there could be the need to install a new power line along the existing to take some load off the congested line, or to replace it with a line that can carry a higher capacity, or even to install a new power generation source near the load to disperse the load more evenly. Being able to predict the congestion further into the future could help for planning future developments in a power system.

1.3 Power system basics

A power system is made up of three primary items: generators, loads, and lines. The power is transferred from the generator to the load via the power lines. As the loads increase, the utilization of the lines increases. Lines have a specific limit on the power that can be transferred based upon thickness, type of material, and what type of conductor is used. When this limit threshold is exceeded, the power line is then considered “congested” and then other nearby power lines must be accommodated to supply the power to the loads without overloading the power lines and causing damage to the system. Over time, power lines get worn out when congested frequently, and if they end up going beyond the transfer limits, catastrophic failures can occur leading to blackouts. The power system by design needs to have an ample number of generators to supply the demand with a surplus, and must have the transmission infrastructure to hold the predicted power capacity to be transferred between the generation and the load.

1.4 Causes of Congestion

Congestion can occur in any system, and pretty much at any time. Depending on how the system was designed there may be places in which there is more generation or load on one side of a system. If there is more load on one side with nowhere else to get the generation then through one line, there will be congestion on that line. If the transmission line cannot carry enough to supply the load, there will be a blackout. Essentially a lack of coordination between the generation and transmission will cause congestion, which can be due to abrupt changes in the load or equipment failure. [7]

1.5 Calculation of Congestion

There are many methods that can be used to gauge how congested a power system is. Line Utilization Factor is an index that can used to estimate the congestion of a single line. It is based upon the MVA rating of the line between two buses which can be found with the following equations that were used in the work by [5] A. Mishra et al.

$$LUF_{ij} = \frac{MVA_{ij}}{MVA_{ij}^{max}} \tag{1.1}$$

With

$$MVA(Apparent\ Power) = \sqrt{MW(Real\ Power)^2 + MVAR(Reactive\ Power)^2} \tag{1.2}$$

and

$$MVAR(Reactive\ Power) = MW(Real\ Power)x\tan(\cos^{-1}(\emptyset)) \tag{1.3}$$

1.6 Contributions made by this thesis

Through the work in the thesis, a means to predict congestion in a power system using Artificial Neural Network has been taken into consideration. The simulation set that was generated programmatically was based upon some basic profiles. These can be found below.

For the power system, a modified version of the IEEE 24 bus (RTS) system was used.

With this data, some batch testing of samples from a whole year of generated data was used to test the capabilities and possible accuracy of an (ANN) to predict congestion using training. Given multiple training algorithms, the most accurate and efficient training was found. A means to generate simulation data using Matlab given some base parameters when power grid data is not available for testing against is proposed as well. From this testing, a specific algorithm was found that produces the best results given the test data.

1.7 Layout of the Thesis

Chapter 1 will be covering the basics about power systems, what congestion is, and how congestion can be calculated. Chapter 2 is a brief summary of some research articles regarding congestion and the means to prevent or predict power system congestion. Chapter 3 is covering information relating to some algorithms used for congestion prediction and information about (ANN) s. Chapter 4 will go over calculations made in the forming of an artificial neural network, and the application of it to the proposed thesis as well as some neural network tool information. Chapter 5 will be presenting the Matlab code that was implemented to produce a simulation data set, and the purpose of it. Chapter 6 will finally be a conclusion going over the results of the works in this thesis, and what could be done in the future to advance the topic of research.

CHAPTER 2

LITERATURE REVIEW

The purpose of this chapter is to go into some different methods that have been applied to power systems to solve issues regarding power system congestion, whether it be prevention in general, dealing with congestion as it surfaces, or by changing the anatomy of the system. Lots of work has been done regarding congestion mitigation, but not a whole lot regarding Congestion Prediction.

2.1 Congestion Management

One method of dealing with congestion is done by modifying power system components with the result either minimizing the amount of congestion, or possibly alleviating it before it causes any problems.

Gope et al. [2] proposed a method to find the generators that would be best fit for output rescheduling using the Generator Sensitivity Factor to help manage congestion, and also went on to use the (FA) to determine the optimal cost that would be involved with these generators that were selected using the (GSF). Their study concluded that (FA) was an improvement upon the older methods. This is beneficial, but it doesn't really help much with the topic of prevention of congestion.

```
Firefly Algorithm
    Objective function f(x), x = (x1, . . ., xd)T
    Initialize a population of fireflies xi(i = 1, 2, . . ., n)
    Define light absorption coefficient _
    while (t < MaxGeneration)
    for i = 1: n all n fireflies
        for j = 1: i all n fireflies
        Brightness Ii at xi is determined by f(xi)
           if (Ij > Ii)
             Move firefly i towards j in all d dimensions
           end if
           Attractiveness changes with distance r via exp
           [-γ r^2]
           Update brightness and evaluate new solutions
end for j
    end for i
    Rank the fireflies and find the current best
    end while
    Post process results and visualization
```

Figure 1. Pseudo code of firefly algorithm [2]

Medeiros Junior et al. [3] presented a means to create a system that would employ two Corrective Switching methods (Relief Function and Loading Characteristics), with the goal to prevent multiple overloads of lines in a system. This results in a system that only requires low computations and can easily be applied to current energy management systems.

Retnamony and Raglend [4] proposed a means to relieve congestion using a cost-free method called PSAT to find the best locations to place certain (FACTS) devices in a system using Eigen value analysis. The stability and system oscillations were compared before and after the addition of the devices. The results were found that out of the (FACTS) devices tested, Unified Power Flow Controller (UPFC) was more effective to increase the stability of the system than the Thyristor-controlled series compensator (TCSC) and Static Synchronous series compensator (SSSC).

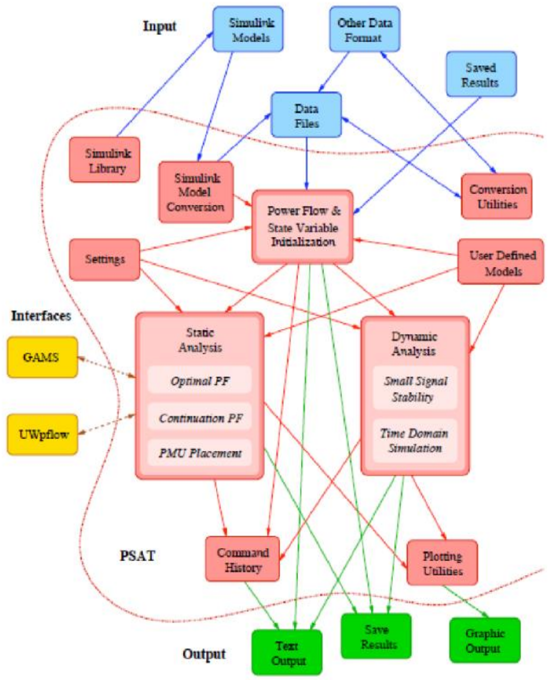

Figure 2. PSAT Synoptic Scheme [4]

Mishra and Kumar [5] proposed calculation of the disparity line utilization factor (DLUF) in order to find optimal placement of an Interline Power Flow Controller (IPFC). With optimal placement of this controller, there is to be expected an improvement of the congestion. The DLUF is an index used for determining the congestion of transmission lines. This was found to be useful to base the calculation of congestion used in this thesis upon. The addition of an IPFC could be something that could be added when there is an area of congestion predicted, in order to mitigate possible congestion.

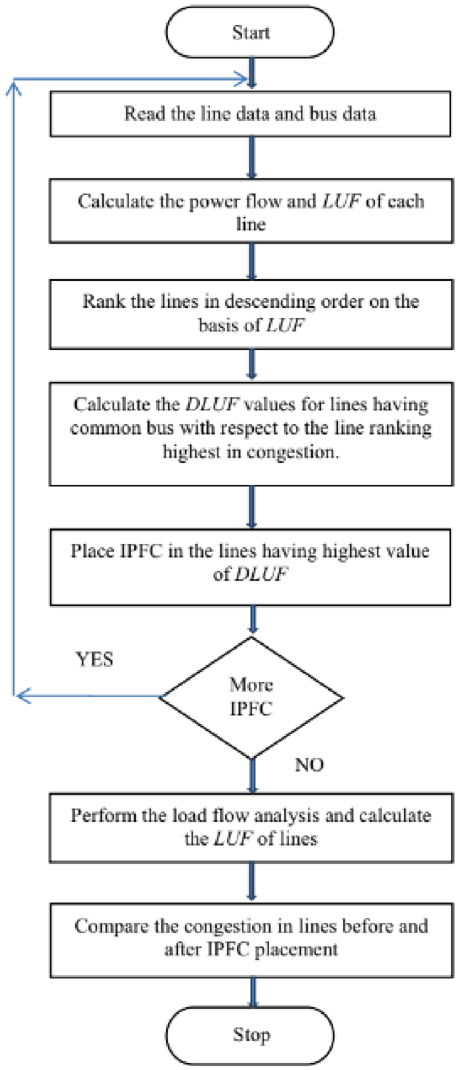

Figure 3. Flow chart for placement of IPFC [5]

Sivakumar and Devaraj [6] proposed a method to manage congestion by means of rescheduling specific generators. Along with this rescheduling they used (GA) to find what the minimum cost of rescheduling would be based upon which generators are selected based on the Generation sensitivity factor (GSF) of each generator. They found that their proposed method would be more effective in larger systems due to the small number of generators available for rescheduling in the system that was tested.

Sandhiya et al. [7] proposed a method to find the best location to place (FACTS) devices using a Simulated Annealing (SA) algorithm. Matlab and Simulink were used in order to simulate the system with and without the (UPFC) device to verify that the application of a (FACTS) device helped the congestion of the system.

Bae et al. [11] proposed a new pricing method that would balance the supply and demand of power and that would maximize the surplus of power that is generated. Their research was tested on the IEEE39 bus system. The means to balance the power loads required a means of managing congestion, due to various reasons such as the large influx of renewables that are being used in power systems where there is supply that is distant from the demand. A means to get the power to the demand, while at the same time keeping line congestion low is one of the problems they were aiming to solve.

Bin et al. [12] proposed a new method of improving cost control in real time systems. The way this was to be done was by implementing congestion factor while working with both generators and loads at the same time. They explain transmission congestion rate and how it is usually used to measure the congestion. Traditionally, the congestion control model is based on what the price is at a given node.

Pandey et al [13] proposed a GRBF (Growing Radial Basis Function) Neural Network methodology to predict NCP (Nodal Congestion Price), with the goal to use the predicted information to manage congestion. Unsupervised learning vector quantization (VQ) clustering method was used in order to extract features from the neural network.

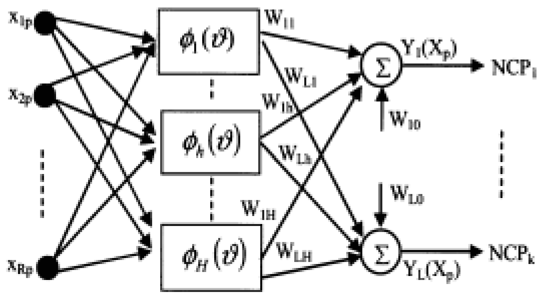

Figure 4. GRBF neural network Architecture [13]

2.2 Congestion Indications

Chakrubarti et al. [10] proposed examining singular values obtained from a decomposition of the basis matrix of a system using a Linear Programming (LP) solution and found that they were not practical indicators of congestion levels. It was found that the smallest singular value was good for explaining why the price was high during a period of congestion.

Trudnowski et al. [15] proposed a means to develop a short term load predictor using Kalman estimators and a by hour forecast. Kalman model parameters are determined by matching the frequency response of the estimator to the load residuals [15]. The system used in their study was from the Bonneville Power Administration.

CHAPTER 3

TRAINING ALGORITHMS

In the Matlab analysis, two main training algorithms were tested, to find which one would result in the best accuracy in testing against the data set. To train and test the ANN, the Neural Time Series Tool (ntstool) was used using the Nonlinear input-output solution. This was selected because of the nature of the data that is being used, essentially the method is to predict future values of output based upon past values of input. There are a few options you have when setting up the training of the datasets. In this testing, the defaults for the percentages of the data to use for validation and testing were left as the default. The next options that had to be setup are related to the architecture of the ANN; how many neurons, and time delays.

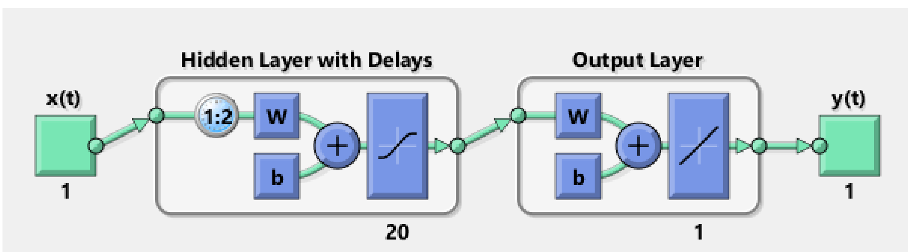

Figure 5. Layout of ANN from Matlab

Two main algorithms were analyzed in this thesis, Levenberg-Marquart and Scaled Conjugate Gradient. They both take a long period of time to run through a whole year of training data. The Scaled Conjugate Gradient has an advantage that it requires less system memory than the Levenberg-Marquart algorithm. There may be better ways to optimize this, an average computer was used to run the algorithms, and a more advanced system may improve speed performance. Bayesian Regularization was another optional algorithm, but it was found to be inefficient with the nature of the data that was used in this research therefore it was ruled out as a candidate early in the research. Below is an example of a training result output, Performance curve, and Autocorrelation plot of a training session.

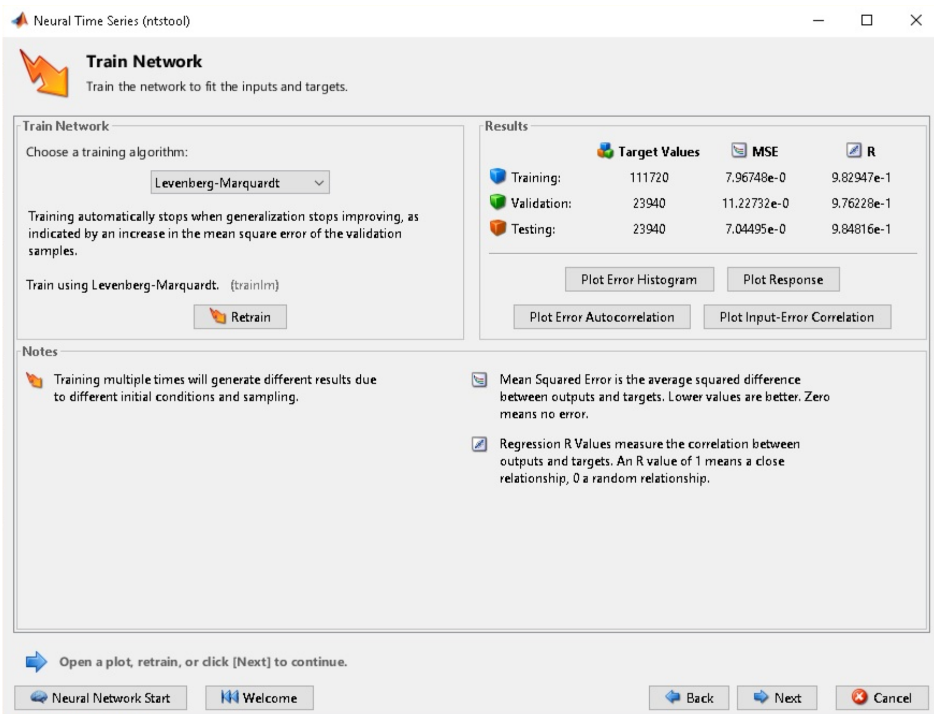

Figure 6. Training results within the ntstool (25,10,2)

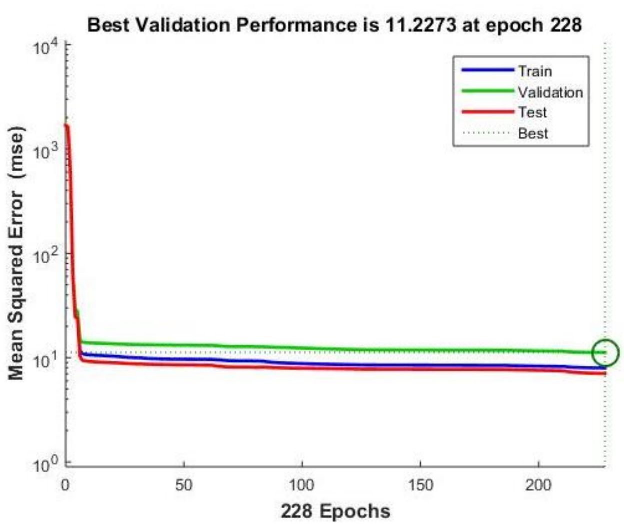

Figure 7. Performance results (25,10,2)

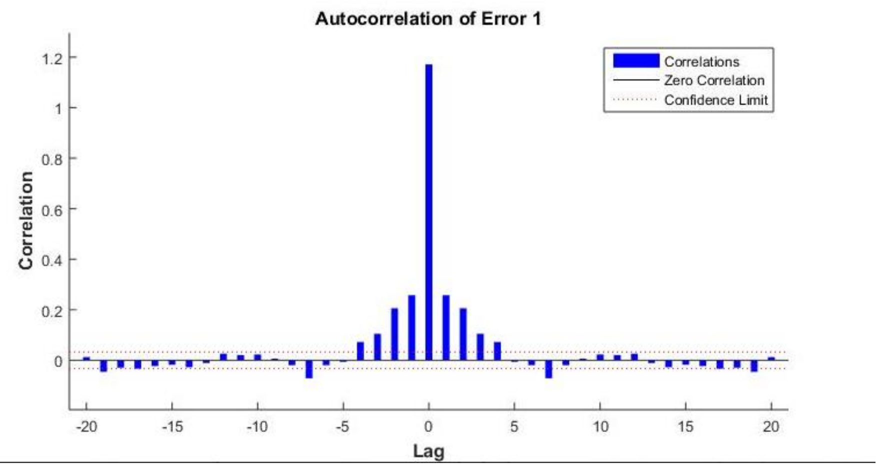

Figure 8. Autocorrelation of Error (25,10,2)

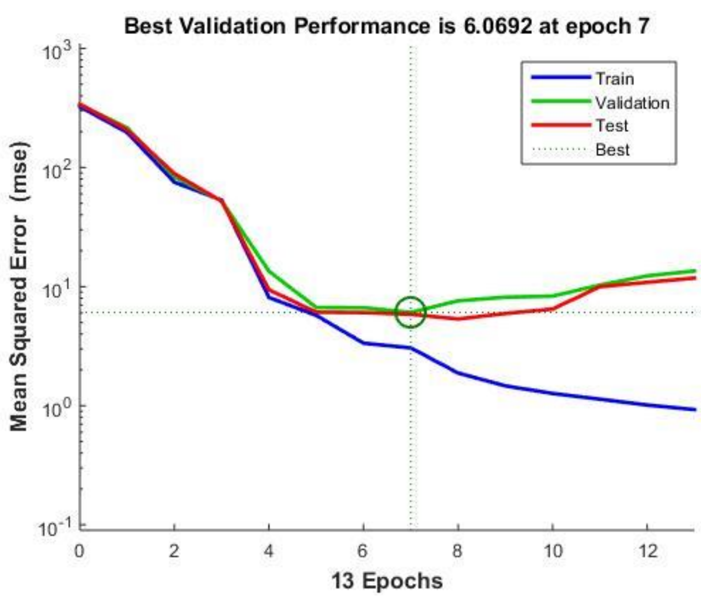

Figure 9. Performance Results (20,10,2)

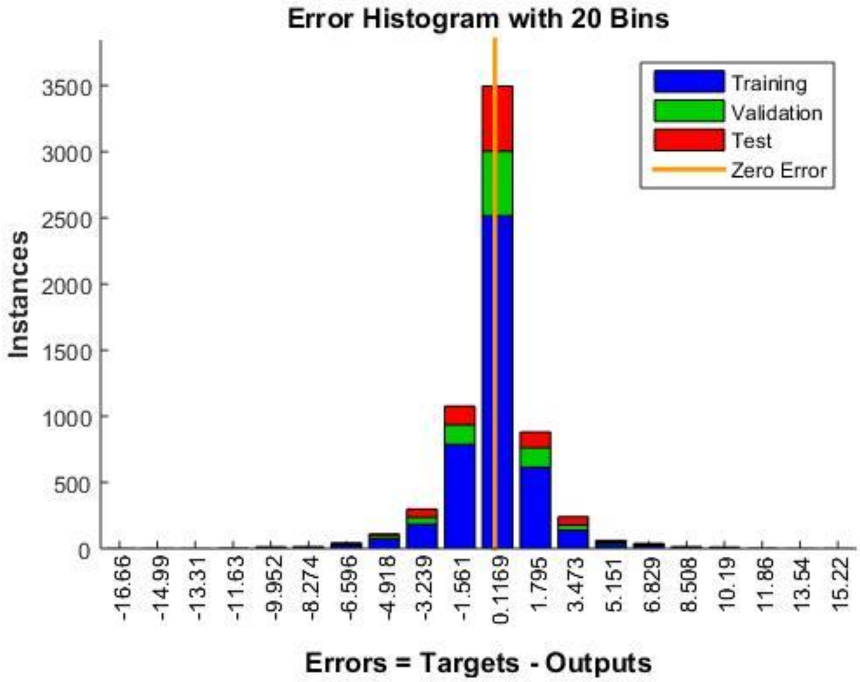

Figure 10. Error Histogram (20,10,2)

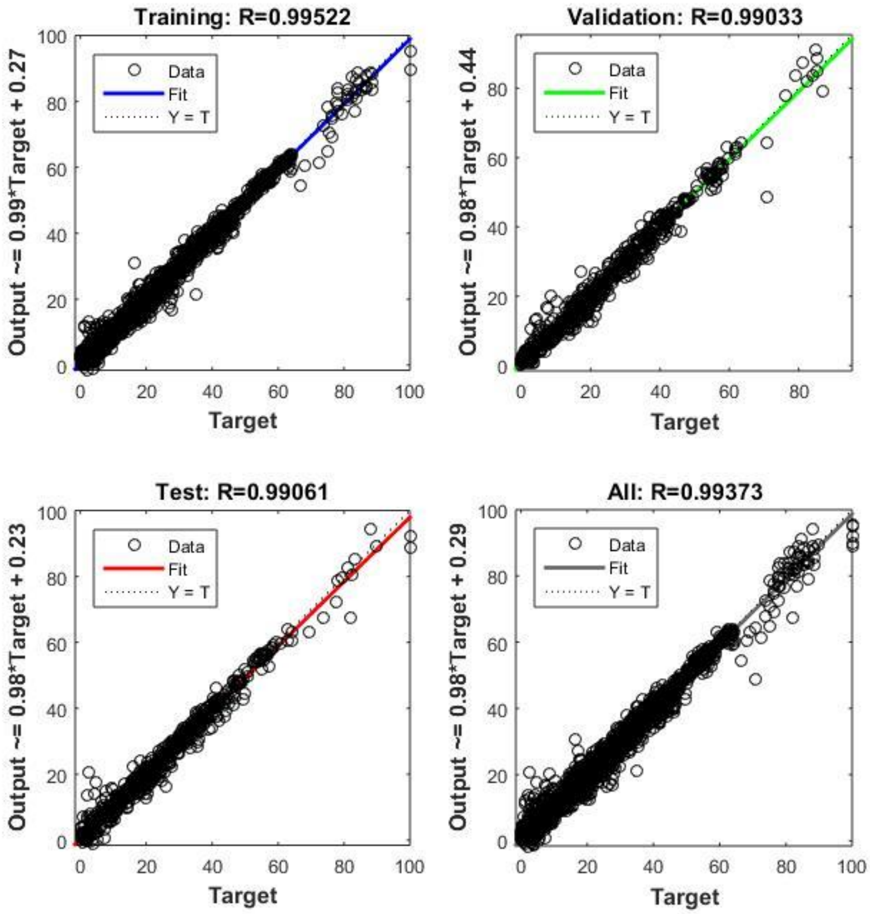

Figure 11. Training Regression (20,10,2)

CHAPTER 4

SIMULATION, ANALYSIS, AND RESULTS

Through this project it was learned that it is very difficult to obtain real time data from a power system, due to the data being protected by the power companies. That lead to only one option, generation of power grid data based upon some load curves, and the IEEE 24 bus system layout.

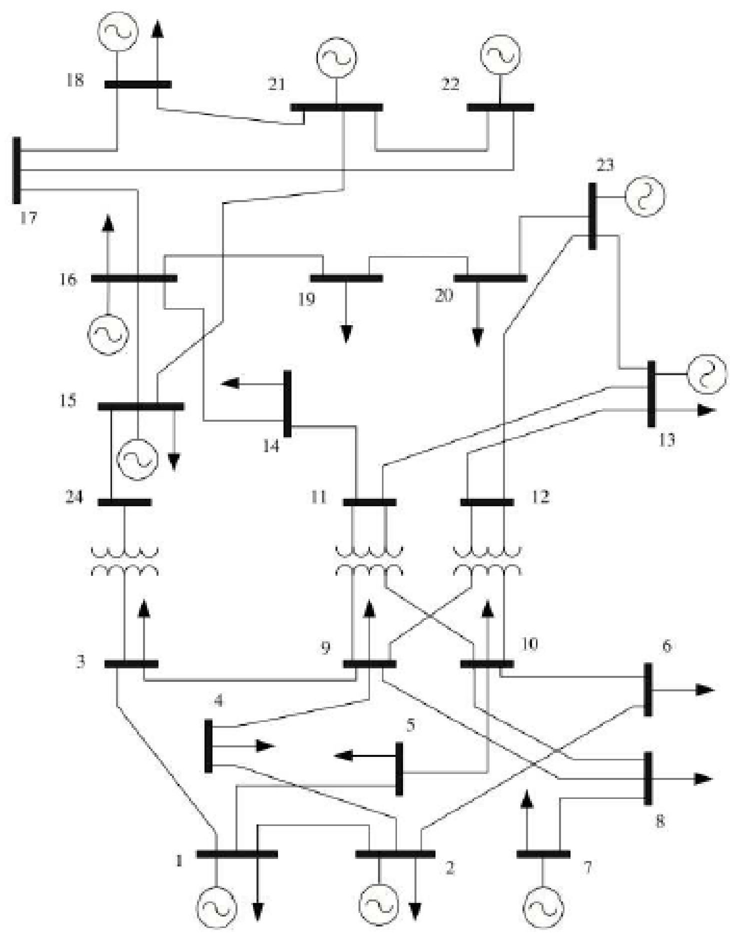

Figure 12. As seen in [1] 24-bus power system IEEE One Area RTS-96

At first, it was thought that Powerworld might be able to provide the necessary capabilities to generate some output data that could be analyzed. Lots of research was done to find out how to use the tool and whether or not it would fit the needs or being able to be modified easily and output simulation data that would be used for training algorithms. Given the circumstances, based upon limitations of the tool and some licensing issues, it was found to not meet the needs of the project.

Using the Matpower plugin tool in Matlab, which enabled some variables related to the buses, generators, branches, and costs to be modified using a base IEEE 24 bus system. The input data was derived from the following tables.

TABLE 1

LOAD PROFILE [1]

Hour	System Demand (MW)	Hour	System Demand (MW)
1	1598252	13	2266.178
2	1502.834	14	2266.178
3	1431.270	15	2218.469
4	1407.416	16	2218.469
5	1407.416	17	2361.596
6	1431.270	18	2385.450
7	2051.487	19	2385.450
8	2266.178	20	2290.032

TABLE 1 (continued)

Hour	System Demand (MW)	Hour	System Demand (MW)
9	2266.178	21	2170.760
10	2290.032	22	1979.924
11	2290.032	23	1741.379
12	2266.178	24	1502.834

TABLE 2

HOURLY PEAK LOAD IN PERCENT OF DAILY PEAK [14]

	Winter 1-8 & 44-52		Summer 18-30		Spring/Fall 9-17 & 31-43	
Hour	Weekday	Weekend	Weekday	Weekend	Weekday	Weekend
12-1 am	67	78	64	74	63	75
1-2	63	72	60	70	62	73
2-3	60	68	58	66	60	69
3-4	59	66	56	65	58	66
4-5	59	64	56	64	59	65
5-6	60	65	58	62	65	65
6-7	74	66	64	62	72	68
7-8	86	70	76	66	85	74

TABLE 2 (continued)

	Winter 1-8 & 44-52		Summer 18-30		Spring/Fall 9-17 & 31-43	
Hour	Weekday	Weekend	Weekday	Hour	Weekday	Weekend
8-9	95	80	87	81	95	83
9-10	96	88	95	86	99	89
10-11	96	90	99	91	100	92
11-Noon	95	91	100	93	99	94
Noon – 1pm	95	90	99	93	93	91
1-2	95	88	100	92	92	90
2-3	93	87	100	91	90	90
3-4	94	87	97	91	88	86
4-5	99	91	96	92	90	85
5-6	100	100	96	94	92	88
6-7	100	99	93	95	96	92
7-8	96	97	92	95	98	100
8-9	91	94	92	100	96	97
9-10	83	92	93	93	90	95
10-11	73	87	87	88	80	90
11-12	63	81	72	80	70	85

Given the input data, a DCOPF could then be run against the parameters to calculate the power flow of the power system. With the output from the DCOPF, the line utilization could be calculated as the ratio of the MVA of each line to the line total limits. Running this calculation in a loop, along with changing variables based on season, month, week, day of year, and adding in random generator and line failures enabled a simulated run of the power system. This is what made it possible to generate simulation data.

Two files are output, one that represents the input values, and another one that represents the output values, which are the % usage of each line. The time in which this value is above a certain threshold (85%) is when it is referred to in this project as being 'congested'. The main goal is this thesis is to train an algorithm that can predict where congestion will occur in order to allow for changes to be made to the system in order to mitigate congestion by making modifications to the power grid/power system. This data was then put into a format (.csv) that could then be input into the ANN training algorithm.

Following is a chart of the results. Within the chart are the listing of runs that were done using a trained network based upon the generated dataset. The number of input weeks, number of neurons, number of weeks predicted in the future and the accuracy are listed for each run. (my) is referring to the Levenberg-Marquart Algorithm and (scg) is referring to the Scaled Conjugate Gradient Algorithm. Overall, it looked to be the runs with a higher input week number along with the predicted output resulted in higher accuracies. Also, the runs with the SCG algorithm resulted in definitely lower accuracies than the Levenberg-Marquart Algorithm.

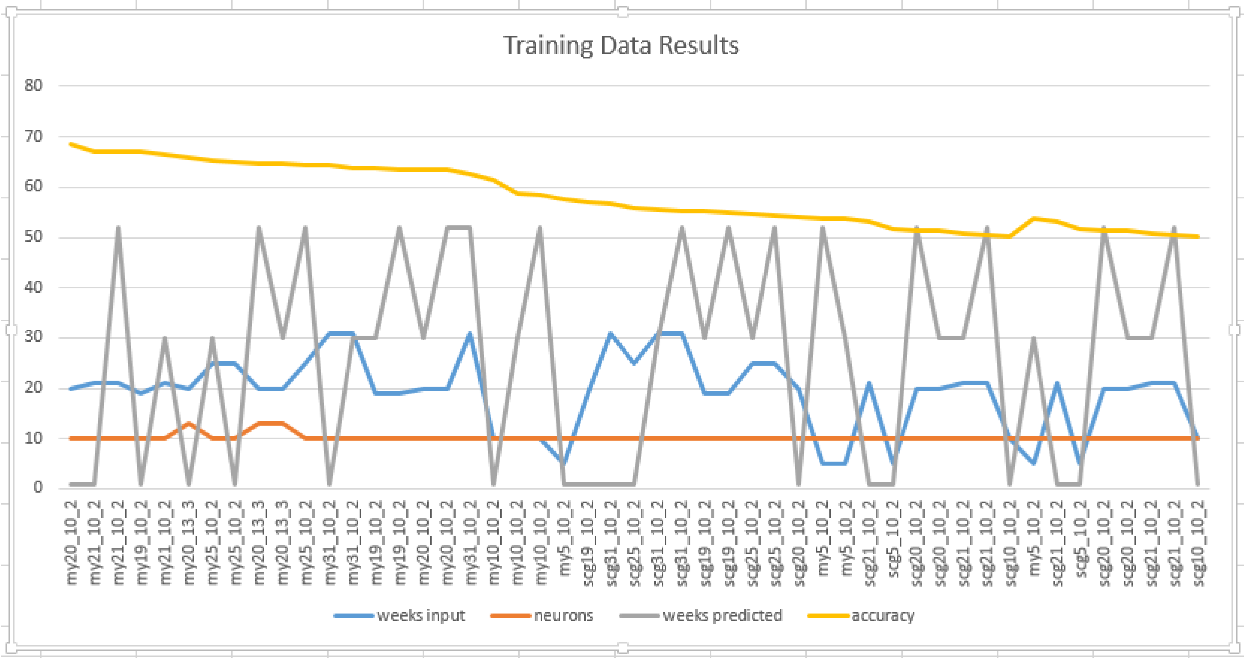

Figure 13. Training Data Results

CHAPTER 5

CONCLUSION

Although there have been some struggles along the way regarding a means to acquire some power system data to use for analysis, and after some trial and error with simulating and generating sample data it was found that the ANN algorithm with the best accuracy was the Levenburg Marquart Algorithm. Based upon the research done, and the small number of articles relating to predicting congestion, it has been found that this is an area that is in need of more development. The idea of a tool that can use historical data to predict future congestion help power grid engineers or planners to make better decisions about what changes to make to the power grid. This could in effect, ensure the high reliability in systems in order to prevent outages and other problems from occurring in the first place.

CHAPTER 6

FUTURE RESEARCH

Through the research it was found that it is hard to find articles relating to this topic, due to lack of research done. There needs to be some more research done in other aspects regarding prediction of congestion regarding what values would make the most optimal prediction system, whether it be load values, current values, line limits, and generator output. There was not enough time to go over every single combination to find the most optimal, so the testing ended up being aimed upon the line limits, bus loads, and peak load. Future research could include splitting the data up into smaller subsets of different combinations of system output data in order to find what would be the best to use for prediction.

Optimization of this method and testing with real system data would also be necessary in order to get a higher accuracy. With generated data, there is only so close to an actual system that you can get due to the limitations on being able to simulate the randomness of failures and outages, and some of the other phenomenon that are found in a real system. Programmatically, it is possible to randomize some failures and outages, but it still doesn’t compare to how a real system behaves, due to all of the phenomena that can affect a system. Also, something that future research would benefit from is a more powerful computing system, to run the training sessions. The system used was only capable of doing up to 13 neurons in the ANN before overloading the available memory. Increasing the neurons could possibly increase the accuracy of the training algorithms due to having more features to compare with the data being tested against it.

REFERENCES

REFERENCES

[1] http://pierrepinson.com/31761/Projects/Project2/IEEE-RTS-24.pdf [cited 11th July 2017]

[2] S. Gope, A. Goswami, et al. “Generator Rescheduling for Congestion Management using Firefly Algorithm” *in International Conference on Energy Systems and Applications (ICESA),* Nov. 2015, pp. 41-44.

[3] M. F. Medeiros Junior, A. V. Costa Oliveira and A. R. Costa Oliveira, "Employment of Corrective Switching in Congestion Management for Eliminating Multiple Overloads in Power Systems," *in IEEE Latin America Transactions*, vol. 14, no. 1, pp. 179-187, Jan. 2016.

[4] R. Retnamony and I. J. Raglend, "Congestion Management is to enhance the transient stability in a deregulated power system using FACTS devices," *2015 International Conference Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kumaracoil, 2015, pp. 744-752.

[5] A. Mishra and G. V. N. Kumar, "Congestion management of power system with interline power flow controller using disparity line utilization factor and multi-objective differential evolution," in *CSEE Journal of Power and Energy Systems*, vol. 1, no. 3, pp. 76-85, Sept. 2015.

[6] S. Sivakumar and D. Devaraj, "Congestion management in deregulated power system by rescheduling of generators using genetic algorithm," *2014 International Conference on Power Signals Control and Computations (EPSCICON)*, Thrissur, 2014, pp. 1-5.

[7] J. Sandhiya, C. Nayanatara and J. Baskaran, "Optimal location of UPFC for congestion relief in power systems using Simulated annealing algorithm," *2016 International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC)*, Chennai, 2016, pp. 648-658.

[8] R. Guguloth and T. K. S. Kumar, "LMP calculation and OPF based congestion management in deregulated power systems," *2016 ELEKTRO*, Strbske Pleso, 2016, pp. 299-304.

[9] S. Deb, S. Gope and A. K. Goswami, "Real power rescheduling for congestion management with integration of wind farm using firefly algorithm," *2015 International Conference on Energy, Power and Environment: Towards Sustainable Growth (ICEPE)*, Shillong, 2015, pp. 1-6.

[10] B. B. Chakrabarti, D. Goodwin and N. K. C. Nair, "Power System Congestion – In Search of an Index from LP Basis Matrices," *2007 IEEE Power Engineering Society General Meeting*, Tampa, FL, 2007, pp. 1-8.

[11] H. Bae, T. Tsuji, T. Oyama and K. Uchida, "Supply and demand balance control of power system with renewable energy integration by introducing congestion management," *2015 IEEE Eindhoven PowerTech*, Eindhoven, 2015, pp. 1-6.

[12] L. Bin, J. Nan, L. Ting and J. Yuanwei, "Transmission congestion control research in power system based on immune genetic algorithm," *Proceedings of the 31st Chinese Control Conference*, Hefei, 2012, pp. 7487-7492.

[13] S. N. Pandey, S. Tapaswi and L. Srivastava, "Price prediction based congestion management using growing RBF neural network," *2008 Annual IEEE India Conference*, Kanpur, 2008, pp. 482-487.

[14] https://link.springer.com/content/pdf/bbm%3A978-1-4613-1689-3%2F1.pdf – Appendix 1 IEEE Reliability Test System [Cited 11th July 2017]

[15] D. J. Trudnowski, W. L. McReynolds and J. M. Johnson, "Real-time very short-term load prediction for power-system automatic generation control," in *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 254-260, Mar 2001.

APPENDIXES

APPENDIX A

MATLAB CODE FOR GENERATION OF SIMULATION DATA

```
define_constants;
for weekl=1:52
num_weeks = weekl;
items_per_week = 168;
A = zeros(num_weeks*items_per_week,97);
B = zeros(num_weeks*items_per_week,38);
Z = zeros(num_weeks*items_per_week,34);
max_load = 2900;
%Randomly Pick 50 hours of the year for a generator to fail
yearly_generator_failures = sort(round(8760*rand(50,1)));
yearly_line_failure = sort(round(8760*rand(25,1)));
day_text = {'Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'};
daily_peak_load_percent = [86 93 91 89 87 70 68 ];
day_of_year = 1;
%Calculation to get total MW load per day
daily_peak_in_MW = daily_peak_load_percent/100 * max_load;
%Percent of daily peak at each hour
%For weeks 1-8 and 44-52 of year
winter_week_hourly_load = [67 63 60 59 59 60 74 86 95 96 96 95 95 95 93 94 99 100 100 96 91
83 73 63];
winter_weekend_hourly_load = [78 72 68 66 64 65 66 70 80 88 90 91 90 88 87 87 91 100 99 97
94 92 87 81];
%For weeks 18-30 of year
summer_week_hourly_load = [64 60 58 56 56 58 64 76 87 95 99 100 99 100 100 97 96 96 93 92
92 93 87 72];
summer_weekend_hourly_load = [74 70 66 65 64 62 62 66 81 86 91 93 93 92 91 91 92 94 95 95
100 93 88 80];
%For weeks 9-17 and 31-43 of year
spring_week_hourly_load = [63 62 60 58 59 65 72 85 95 99 100 99 93 92 90 88 90 92 96 98 96
90 80 70];
spring_weekend_hourly_load = [75 74 69 66 65 65 68 74 83 89 92 94 91 90 90 86 85 88 92 100
97 95 90 85];
```

```
bus_locations = [1 2 3 4 5 6 7 8 9 10 13 14 15 16 18 19 20];
bus_location_percent = [3.8 3.4 6.3 2.6 2.5 4.8 2.4 6.0 6.1 2.8 9.3 6.8 11.1 5.5 15.7 6.4 4.5];
gen_locations = [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32 33];
hour_of_year = 1;
season = 0;
for week=1:num_weeks
  week;
  if(week == 1 || week == 2 || week ==3 || week ==4 || week ==5 || week ==6 || week ==7 || week
==8 || week ==
44||week==45||week==46||week==47||week==48||week==49||week==50||week==51||week==52)
    week_hourly_percent = winter_week_hourly_load;
    weekend_hourly_percent = winter_weekend_hourly_load;
    season_text = 'Winter';
    season = 1;
  end
  if(week == 18 || week == 19 || week == 20 || week == 21 || week == 22 || week == 23 || week ==
24 || week == 25 || week == 26 || week == 27 || week == 28 || week == 29 || week == 30)
    week_hourly_percent = summer_week_hourly_load;
    weekend_hourly_percent = summer_week_hourly_load;
    season_text = 'Summer';
    season = 2;
  end
  if(week == 9 || week ==10 || week ==11 || week ==12|| week ==13 || week ==14 || week ==15 ||
week ==16 || week ==17 || week ==31 || week ==32 || week ==33 || week ==34|| week ==35 ||
week ==36 || week ==37 || week ==38 || week ==39 || week ==40 || week ==41 || week ==42 ||
week ==43)
    week_hourly_percent = spring_week_hourly_load;
    weekend_hourly_percent = spring_week_hourly_load;
    season_text = 'Spring';
    season = 3;
  end
  for day=1:7
    day_text{day};
    if(day == 1 || day ==2 || day==3 ||day==4|| day==5)
      day_peak = daily_peak_in_MW(day);
      day_hourly = (week_hourly_percent/100)*day_peak;
    end
```

```
if(day == 6 || day == 7)
  day_peak = daily_peak_in_MW(day);
  day_hourly = (weekend_hourly_percent/100)*day_peak;
end
for hour=1:24
  mpc = loadcase('case24_ieee_rts');
  A(hour_of_year,5) = day_peak;
  A(hour_of_year,3) = day;
  A(hour_of_year,8) = hour_of_year;
  A(hour_of_year,4) = season;
  A(hour_of_year,6) = week;
  A(hour_of_year,7) = day_of_year;
  season_text;
  week;
  day_text{day};
  hour;
  total_hourly_load = 0;
  bus_loads = [];
  for bus_loc=1:length(bus_locations)
    bus_loads(bus_loc) = (bus_location_percent(bus_loc)/100)*day_hourly(hour);
    mpc.bus(bus_loc,PD) = bus_loads(bus_loc);
    bus_load_start = 8;
    A(hour_of_year,(bus_load_start+bus_loc)) = bus_loads(bus_loc);
    total_hourly_load = total_hourly_load +bus_loads(bus_loc);
  end
  total_hourly_load;
  the_hour = ismember(hour_of_year,yearly_generator_failures);
  if(the_hour)
    generator_num = round(33*rand(1,1));
    if(generator_num == 0)
      generator_num = round(33*rand(1,1));
    end
    A(hour_of_year,2) = generator_num;
    mpc.gen(generator_num, 9)= 0;
  end
  the_line_hour = ismember(hour_of_year,yearly_line_failure);
```

```
if(the_line_hour)
  line_num = round(37*rand(1,1));
  if(line_num == 0)
    line_num = round(37*rand(1,1));
  end
  A(hour_of_year,1) = line_num;
  mpc.branch(line_num,11) = 0;
end
for gen_loc=1:length(gen_locations)
  gen_output_start = 25;
  A(hour_of_year,(gen_output_start+gen_loc)) = mpc.gen(gen_loc, 9);
  Z(hour_of_year,(gen_loc)) = mpc.gen(gen_loc,9);
end
result = rundcopf(mpc);
%38 lines
for z=1:38
  %Limit
  start = 59;
  A(hour_of_year,(start+z)) = mpc.branch(z,6);
end
line_mva_limits = mpc.branch (:,6);
line_MW_from = result.branch (:,PF);
line_MVAR_from = result.branch (:,QF);
mva_run = sqrt(line_MW_from.^2+line_MVAR_from.^2);
for z=1:length(mva_run)
  percent_usage(z) = (mva_run(z)/line_mva_limits(z))*100;
  percent_start = 98;
  ones_start = 0;
  if(percent_usage(z) > 84.9)
     %percent_ones (z) = 1;
     percent_ones (z) = percent_usage(z);
  else
     %percent_ones(z) = 0;
     percent_ones(z) = percent_usage(z);
  end
  B(hour_of_year,(ones_start+z)) = percent_ones(z);
```

```
      end
      percent_usage;
      hour_of_year
      hour_of_year=hour_of_year+1;
      clc
    end
    day_of_year =day_of_year+1;
  end
end
outputstring = sprintf('out_%d.csv',num_weeks);
inputstring = sprintf('in_%d.csv',num_weeks);
csvwrite(inputstring,A)
csvwrite(outputstring,B)
weekl
pause(5);
end
```

APPENDIX B

RESULTS TABLE

Type	#weeks trained	neurons	time delays	Size	#Weeks Predicted	Tolerance	Accuracy
my20_10_2	20	10	2	17	1	+/-1	68.452
my21_10_2	21	10	2	17	1	+/-1	67.215
my21_10_2	21	10	2	17	52	+/-1	67.134
my19_10_2	19	10	2	17	1	+/-1	67.043
my21_10_2	21	10	2	17	30	+/-1	66.547
my20_13_3	20	13	3	17	1	+/-1	66.024
my25_10_2	25	10	2	17	30	+/-1	65.196
my25_10_2	25	10	2	17	1	+/-1	65.132
my20_13_3	20	13	3	17	52	+/-1	64.755
my20_13_3	20	13	3	17	30	+/-1	64.692
my25_10_2	25	10	2	17	52	+/-1	64.494
my31_10_2	31	10	2	17	1	+/-1	64.254
my31_10_2	31	10	2	17	30	+/-1	63.794
my19_10_2	19	10	2	17	30	+/-1	63.789
my19_10_2	19	10	2	17	52	+/-1	63.644
my20_10_2	20	10	2	17	30	+/-1	63.547
my20_10_2	20	10	2	17	52	+/-1	63.469
my31_10_2	31	10	2	17	52	+/-1	62.657
my10_10_2	10	10	2	17	1	+/-1	61.497
my10_10_2	10	10	2	17	30	+/-1	58.865
my10_10_2	10	10	2	17	52	+/-1	58.519
my5_10_2	5	10	2	17	1	+/-1	57.55
scg19_10_2	19	10	2	17	1	+/-1	57.08
scg31_10_2	31	10	2	17	1	+/-1	56.814
scg25_10_2	25	10	2	17	1	+/-1	55.655
scg31_10_2	31	10	2	17	30	+/-1	55.572
scg31_10_2	31	10	2	17	52	+/-1	55.296
scg19_10_2	19	10	2	17	30	+/-1	55.143
scg19_10_2	19	10	2	17	52	+/-1	54.893

Type	#weeks trained	neurons	time delays	Size	#Weeks Predicted	Tolerance	Accuracy
scg25_10_2	25	10	2	17	30	+/-1	54.645
scg25_10_2	25	10	2	17	52	+/-1	54.363
scg20_10_2	20	10	2	17	1	+/-1	53.947
my5_10_2	5	10	2	17	52	+/-1	53.833
my5_10_2	5	10	2	17	30	+/-1	53.767
scg21_10_2	21	10	2	17	1	+/-1	53.117
scg5_10_2	5	10	2	17	1	+/-1	51.754
scg20_10_2	20	10	2	17	52	+/-1	51.36
scg20_10_2	20	10	2	17	30	+/-1	51.242
scg21_10_2	21	10	2	17	30	+/-1	50.881
scg21_10_2	21	10	2	17	52	+/-1	50.41
scg10_10_2	10	10	2	17	1	+/-1	50.251
scg5_10_2	5	10	2	17	30	+/-1	50.218
scg5_10_2	5	10	2	17	52	+/-1	50.036
scg10_10_2	10	10	2	17	30	+/-1	48.04
scg10_10_2	10	10	2	17	52	+/-1	47.986