Model Selection For Forecasting Rainfall data in Wonorejo Reservoir using Neural Network

Dwilaksana Abdullah Rasyid, Kinanti Hanugera Gusti, Regita Putri Permata
Statistics Department, Faculty of Mathematics, Computing, and Data Science,
Institut Teknologi Sepuluh Nopember
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia

A. Abstract

The selection of input models in neural networks influence the predictive accuracy. The purpose of this paper is to compare input models in neural network time series using the Surabaya rainfall data in Wonorejo reservoir. The input for Neural Network model from a significant lag on PACF and ARIMA models. From the simulation results it was found that the best FFNN model using PACF input lag with the Rprop+ with logistic activation function. The best DLNN model using the Rprop- algorithm with the logistics activation function. The best model selection in rainfall forecasting with the smallest RMSEP is FFNN model (8,4,1). Then, we optimize best FFNN model by eliminating nonsignificant lag input using stepwise. The result shows that stepwise process is not significantly affect because RMSEP greater than before.

B. Introduction

Neural networks (NN) have found increasing consideration in forecasting theory, leading to successful applications in time series prediction and explanatory forecasting. Despite their theoretical capabilities, NN have not been able to confirm their potential in forecasting competitions against established statistical methods, such as ARIMA or Exponential Smoothing [1]. As NN offer many degrees of freedom in the modelling process, from the selection of activation functions, adequate network topologies of input, hidden and output nodes, learning algorithms etc. their valid and reliable use is often considered as much an art as science. Previous research indicates, that the parsimonious identification of input variables and lags to forecast an unknown data generating process without domain knowledge poses a key problem in model specification [2].

The research by Box and Jenkins shows that identifying the most relevant input variables in the selection of features time series data in the Neural Network model specifications. It becomes particularly important, as complex time series components may include deterministic or stochastic trends and seasonality, interacting in a linear or nonlinear model with pulses, level shifts, structural breaks and different distributions of noise [3]. Although a number of statistical methods have been developed to support the identification of linear dependencies, their use in nonlinear prediction has not been investigated in detail. In other research that compare of two types of neural networks: single- and multiple-hidden-layer networks. The result show that single-hidden-layer networks converge faster to linear target functions compared to multiple-hidden-layer networks [4]. The correct model selection required information criterion to obtain the best prediction results, in this case we use RMSEP [5].

Considering the role of Computational Intelligence to rainfall forecasting, in this paper we use neural networks and time series to deal with the problem of rainfall occurence in a case study scenario. We considered data from Wonorejo, Surabaya in which rainfall. This rainfall data is used because it has the characteristics of two seasons namely the dry season and the rainy season in difference. To predict rainfall in Wonorejo it is not enough just to use statistical methods in detecting rainfall patterns by using Neural Networks to understand the

importance of nonlinear relationship patterns. The purpose of this article is to select the best neural network input model based on statistical concepts such as significant lags in PACF, building inputs from ARIMA modeling.

C. Methods

Time series analysis aims to find the pattern of historical time series data and extrapolate the pattern in the past which is used to forecast the future. Several time series models are given in the following:

1. ARIMA

ARIMA(*p,d,q*) is a nonstasionerity time series model derived from ARMA(*p,q*) such that it requires *d* times differencing to be stationary. The general form of ARIMA (*p,d,q*) is given in the following [6].

$$\phi_p(B)(1-B)^d Y_t = \theta_0 + \theta_q(B)a_t. \tag{1}$$

Multiplicative ARIMA model with s seasonal period is denoted by ARIMA $(p, d, q)(P, D, Q)^s$. The general form of seasonal multiplicative ARIMA model is given as follows.

$$\Phi_P(B^s)\phi_p(B)(1-B)^d(1-B^s)^D\dot{Y}_t = \theta_q(B)\Theta_Q(B^s)a_t, \tag{2}$$

where $\phi_p(B)$ denotes AR operator, $\Phi_P(B^S)$ denotes seasonal AR operator, $\theta_q(B)$ denotes MA operator, $\Theta_Q(B^s)$ denotes seasonal MA operator, and a_t white noise residual with zero mean and constant variance σ_a^2 .

2. Feed Forward Neural Network

Feed Forward Neural Network (FFNN) is one of popular nonlinear model which is widely used for time series forecasting. FFNN consists of input layer, hidden layer, and output layer. Each layer contains elements called neurons. Each neuron will receive information only from the neurons in the previous layer [7]. FFNN model for univariate time series data with *p* inputs, *q* hidden neurons, and 1 single output, denoted by FFNN(*p, q*), can be expressed as follows.

$$\hat{Y}_{(t)} = f^o\left[\sum_{j=1}^{q}\left[w_j^o f_j^h\left(\sum_{i=1}^{p} w_{ji}^h X_{i(t)} + b_j^h\right) + b^o\right]\right], \tag{3}$$

where w_{ji}^h are the weights that connect input layer to hidden layer, w_j^o are the weights that connect hidden layer to output layer. $f(.)$ is called activation function. b^o and b_j^h are the biases. $X_{i(t)}$ are the input values and $\hat{Y}_{(t)}$ are the predicted output values. In this study, we compared two algorithm namely Rprop- and Rprop+ and use two activations namely logistics and tanh. Which is given logistics activation function follows.

$$f(x) = \frac{1}{1+e^{-x}} \tag{4}$$

Tangent hyperbolic (tanh) function as the activation function, which is given as follows.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{5}$$

3. Deep Learning Neural Network

Deep Learning Neural Networks (DLNN) is a Feed Forward Neural Networks (FFNN) with the number of hidden layers more than one. In the time series model, the relationship between outputs $\hat{Y}_t$ and inputs $(Y_{t-1}, Y_{t-2}, ..., Y_{t-p},)$ in the DLNN model with two hidden layer is given as follows.

$$\hat{Y}_{(t)} = f^{o}\left[\sum_{i=1}^{s}\alpha_i f_i^{h_2}\left[\sum_{j=1}^{r}\beta_{ij} f_j^{h_1}\left(\sum_{t=1}^{p}\gamma_{it} X_{i(t)} + b_j^{h_1}\right) + b_i^{h_2}\right] + b^{o}\right] + \varepsilon_t \qquad (6)$$

where α_i weight of neuron from second hidden layers to output layer, $b_i^{h_2}$ are biases from neuron second hidden layer. The architectures of FFNN and DLNN are shown in Fig. 1.

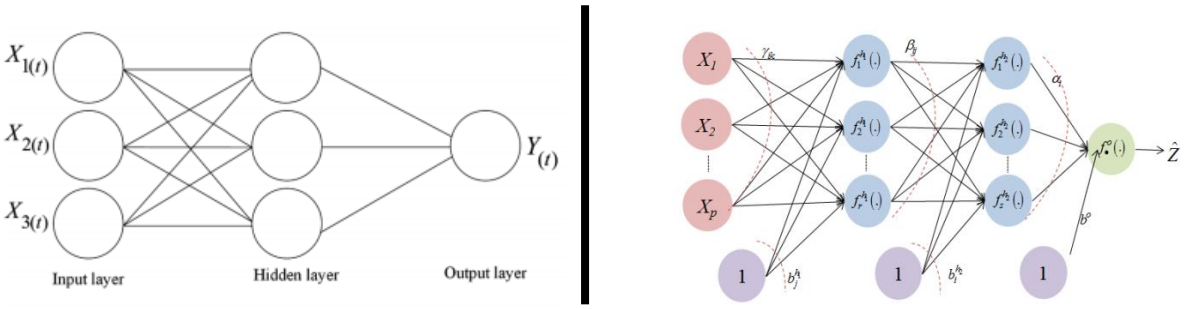

Fig. 1. (Example) the neural architecture of FFNN (left) and DLNN (right)

4. Input Selection

Stepwise methods. Forward selection is said to have *fidelity*, in that once an input variable is selected, the selection can not be undone. Step-wise selection is an extension of the forward selection approach, where input variables may also be removed at any subsequent iteration. The formulation of the step-wise approach is aimed at handling redundancy between candidate variables. For example, a variable X_a may be selected initially due to high relevance, but is later found to be inferior to the combination of two other variables, X_b and X_c , which only arises at a subsequent iteration. The initially selected input variable X_a is now redundant, and can be removed in favour of the pair X_b and X_c. A common example of this approach is step-wise regression, which is widely used for the development of linear regression models. In this wrapper approach, linear models are iteratively constructed by adding an input variable to the model, and re-estimating the model coefficients. Input variables are retained based on analysis of the coefficients of the newly developed model. The selection process continues until the model satisfies some optimality criterion, such as the AIC, that is, when *k*+1 input variables are no better than the preceding *k* variables [8]. After we determine the input models affected by used stepwise method, these lag significant obtained were inserted into neural network which was according to the steps that were described previously are shown in Fig. 2.

Fig.2 Diagram of Stepwise

5. Model Selection

We use Root Mean Square Error of Prediction (RMSEP) as the criteria for model selection. The formula of RMSEP is given respectively as follows [6].

$$RMSEP = \sqrt{\frac{\sum_{l=1}^{L}(Y_{n+l} - \hat{Y}_n(l))^2}{L}}. \quad (7)$$

where Y_{n+1} denotes the actual value, $\hat{Y}_n(l)$ denotes the forecast value, and L denotes forecast horizon.

6. Input Variable Selection for Neural Network

The identification of relevant input variables and variable lags aims at capturing the relevant components of the data generating process in a parsimonious form. In time series modeling, it is closely related with identifying the underlying time series components of trend and seasonality and capturing their deterministic behavior in lags of the dependent variable. A simple visual analysis of the time series components frequently fails to reveal the complex interactions of autoregressive and moving average components, multiple overlying and interacting seasonalities and nonlinear patterns. Several methodologies have been developed for input variables selection of the significant lags in forecasting, originating from linear statistics and engineering. However, currently no uniformly accepted approach exists to identify linear or nonlinear input variables [9].

In this study, we use data secunder about Wonorejo rainfall. The dataset contains monthly rainfall period January 1998 to December 2018. We split the data into training and testing sets, where training set contains data from January 1998 to December 2016 and the rest data becomes testing set.

In order to specify a network architecture, we have to choose the relevant input variables and the appropriate number of hidden units, i.e. the complexity of functional form. In this study, we used one hidden layer and tried 1 to 10 neuron units with 10 replications, compared two algorithm namely Rprop- and Rprop+ and use two activations namely logistics and tanh. Earlier studies in Neural Network modeling claim that an analysis of the AR-terms purely from PACF-analysis is sufficient to identify the relevant lags of the time series using differencing of seasonal 12.

D. Results

The average rainfall in the Wonorejo Surabaya during the 1998-2006 period was 213.52 mm. The time series plot shows that rainfall is seasonal in a year and has a tendency to increase from October to March. Time series plot of Wonorejo rainfall are given Fig. 3.

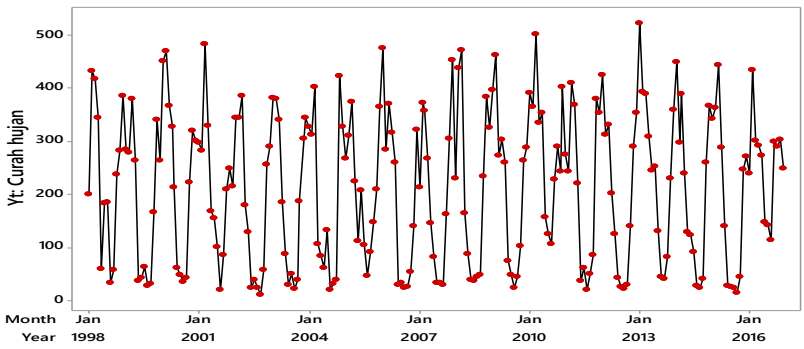

Fig. 3 Time Series plot of Wonorejo rainfall

Figure 4 shows that in January, February, March, April, November and December the average value of rainfall is quite high. Meanwhile, in July, August and September have average rainfall was low, which indicates that there is a dry season and the season from the rainy season to the dry season or vice versa in the month. In 2016 there were differences in rainfall patterns in August and September so that it showed a higher rainfall pattern than other patterns.

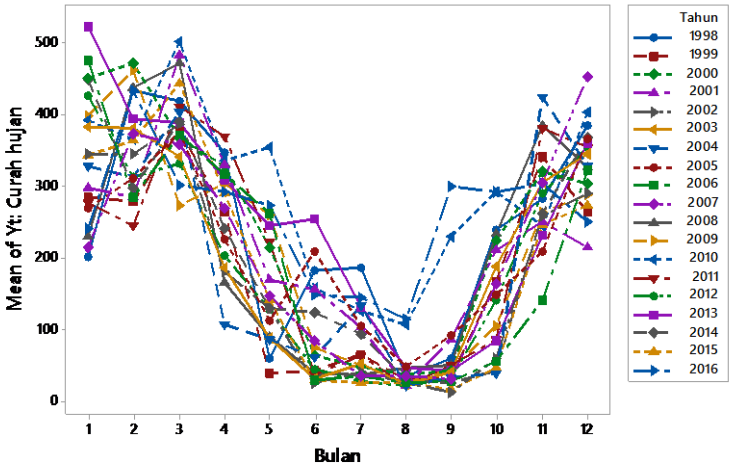

Fig. 4 Line plots of Wonorejo rainfall

ARIMA Modeling. ARIMA modeling using the Box-Jenkins procedure consists of model identification, parameter estimation, diagnostic checking, and forecasting data. ARIMA models along with forecast accuracy are shown in Table 1.

Table 1. Forecast evaluation of ARIMA

ARIMA models	White Noise	RMSEP
$(0,0,1)(0,1,1)^{12}$	Ya	65.552
$([1,3,5],0,0)(2,1,0)^{12}$	Ya	68.632

From the modelling results in Table 1, the first ARIMA models is more accurate than the second ARIMA models it is ARIMA$(0,0,1)(0,1,1)^{12}$ using RMSEP evaluation.

Feed Forward Neural Network Models. We create a NN architecture model for each neurons of hidden layer, algorithm combination, and activation combination. First using high significant lag of PACF as inputs variables of $\{y_{t-1}, y_{t-3}, y_{t-5}, y_{t-12}, y_{t-13}, y_{t-15}, y_{t-17}, y_{t-24}\}$.

Table 2. Forecast evaluation of FFNN using standardized preprocessing

	Rprop-		Rprop+	
Neuron	test tanh	Test log	test tanh	Test log
1	85.644	85.733	85.697	85.689
2	75.953	75.858	100.358	75.577
3	90.809	92.706	117.529	67.238
4	95.274	58.882	78.562	74.887
5	127.755	87.463	132.249	147.302
6	148.567	87.884	147.779	130.245
7	101.211	116.546	147.047	96.924
8	83.6935	163.920	114.704	95.592
9	218.357	167.571	87.2509	137.986
10	179.264	124.907	188.453	153.474

Table 3. Forecast evaluation of FFNN using normalized preprocessing

	Rprop-		Rprop+	
Neuron	**test tanh**	**Test log**	**test tanh**	**Test log**
1	81.961	81.933	82.730	81.882
2	72.221	72.434	72.643	73.087
3	74.229	56.237	62.611	69.224
4	80.236	91.173	107.078	51.961
5	72.424	95.602	68.821	85.179
6	130.252	66.680	79.918	97.061
7	147.347	95.671	142.745	94.956
8	124.509	643.671	799.918	82.737
9	127.059	128.918	1396.57	97.498
10	138.653	98.194	99.694	98.048

Table 4. Forecast evaluation of FFNN using adjusted normalized preprocessing

	Rprop-		Rprop+	
Neuron	**test tanh**	**Test log**	**test tanh**	**Test log**
1	83.883	83.881	83.994	83.854
2	75.957	73.765	76.043	76.337
3	74.653	90.767	74.629	124.27
4	73.800	77.655	115.977	74.319
5	90.457	99.334	98.921	64.196
6	105.844	97.168	104.941	81.509
7	149.521	75.194	222.455	120.726
8	106.686	111.952	154.455	106.033
9	96.391	163.017	141.934	96.548
10	309.068	118.408	109.435	232.803

From the table, the formation of a Neural Network model using 10 combination neurons shows that the two NN algorithms, the logistic activation function gives a smaller RMSEP value than the tanh activation function. Other than that, the smallest RMSEP is on neuron 4 using the logistic activation function and the Rprop + algorithm with the best preprocessing data using normalized.

Input using ARIMA model. From the modeling of ARIMA, we use input model from ARIMA([1,3,5],0,0)(2,1,0)^{12}. ARIMA model selection using subset because the nonseasonal AR model and seasonal AR model with differencing 12. Input variable that we used is $\{y_{t-1}, y_{t-3}, y_{t-5}, y_{t-12}, y_{t-13}, y_{t-15}, y_{t-24}, y_{t-25}, y_{t-29}, y_{t-36}\}$ and use the normalized preprocessing.

Table 5 Forecast evaluation of FFNN input ARIMA model

	Rprop-		Rprop+	
Neuron	**test tanh**	**Test log**	**test tanh**	**Test log**
1	66.884	66.869	67.263	66.990
2	89.193	87.735	88.743	87.293
3	88.007	86.462	70.827	57.057
4	72.414	98.901	86.933	96.239
5	105.877	91.439	78.270	94.123
6	102.805	67.842	101.297	121.321
7	74.113	75.623	110.471	393.740
8	86.111	133.759	84.130	107.857
9	96.272	163.290	373.684	168.343

Rprop-			Rprop+	
Neuron	**test tanh**	**Test log**	**test tanh**	**Test log**
10	515.559	99.662	650.098	98.330

Table 5 shows that ARIMA input with 3 neuron and logistic activation function from Rprop- algorithm has accuration value 57,057.

Deep Learning Neural Network Models. DLNN model input variables significant lag from PACF $\{y_{t-1}, y_{t-3}, y_{t-5}, y_{t-12}, y_{t-13}, y_{t-15}, y_{t-17}, y_{t-24}\}$. The algorithm that is tested on the DLNN method is Rprop + and Rprop- without using replication. The DLNN architectures that will be applied are two hidden layers with1 untul 3 neurons in each layers. We didnt use replication because there is no effect of the value in the addition of nodes in the DLNN layer.

Table 6. Forecast evaluation of DLNN-PACF using standardized preprocessing

Neuron	RPROP+		RPROP-	
	Tanh	logistic	Tanh	logistic
	RMSEP	RMSEP	RMSEP	RMSEP
1-1	67.251	67.076	67.245	67.096
1-2	66.021	69.988	66.142	67.244
2-1	63.402	62.548	67.255	66.966
2-2	67.285	66.952	64.752	77.899
3-1	81.865	91.923		
3-2	74.233	101.670		

Table 7. Forecast evaluation of DLNN-PACF using normalized preprocessing

Neuron	RPROP+		RPROP-	
	Tanh	logistic	Tanh	logistic
	RMSEP	RMSEP	RMSEP	RMSEP
1-1	63.717	63.816	63.165	63.723
1-2	75.864	75.666	75.143	77.301
2-1	75.469	64.850	78.878	60.537
2-2	64.145	64.877	62.778	63.775
3-1	74.664	75.870	74.736	70.666
3-2	72.510	70.259	63.631	58.744

Table 8. Forecast evaluation of DLNN-PACF using adjusted normalized preprocessing

Neuron	RPROP+		RPROP-	
	Tanh	logistic	Tanh	logistic
	RMSEP	RMSEP	RMSEP	RMSEP
1-1	65.316	65.310	65.364	64.983
1-2	67.452	77.425	82.532	61.499
2-1	66.839	68.093	81.649	66.918
2-2	64.916	64.956	64.808	64.873
3-1	77.120	69.032	62.702	64.051
3-2	75.879	65.669	64.942	65.733

Based on Table 6, Table 7, and Table 8, the best DLNN model results are obtained when using the Rprop-algorithm. However, when the standardized data as input selection, the

convergence model is not achieved when the first neuron is more than two. Probably, it is because the uncontrolled weight and bias on it neuron make too many iterations needed to reach convergence. The convergence iteration criteria which has been declare at the beginning were not enough to make the model on neuron be convergence, so it must be modified again, moreover the more neurons the older time needed for converging model.

The use of Rprop- algorithm will produce a good DLNN when using normalized data as preprocessing data and logictic activation function. Normalized values fall between 0 and 1. This is the required rescaling method for scale-dependent variables if the output layer uses the sigmoid activation function. The correction option specifies a small number ε that is applied as a correction to the rescaling formula; this correction ensures that all rescaled dependent variable values will be within the range of the activation function.

From Table 3 we have best FFNN (8,4,1) with smallest RMSEP which shown in Figure...

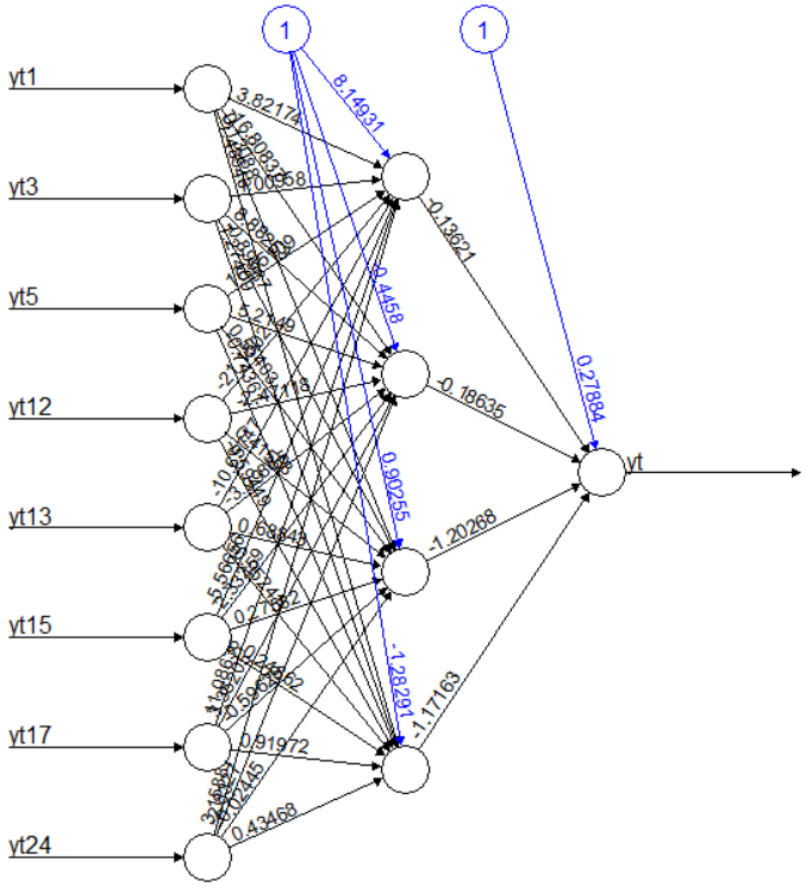

Fig. 5 Architecture of FFNN (8,4,1)

From the Neural Network iput simulation, we found that FFNN method with PACF input using Rprop+ on neuron 4 has the best accuracy. Then we use stepwise on PACF input to get a significant lag.

The result of stepwise method shows that significant lag input from PACF are $\{y_{t-1}, y_{t-3}, y_{t-5}, y_{t-12}, y_{t-15}\}$ and RMSEP value obtained by 59,759.

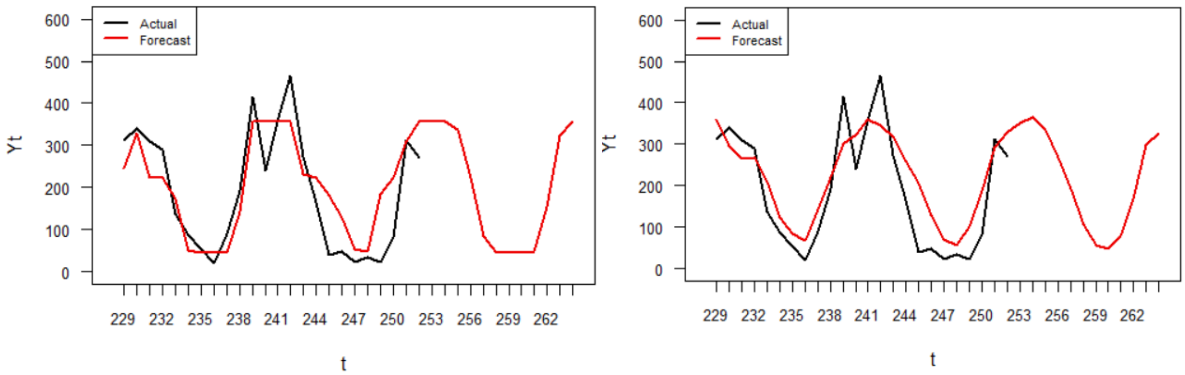

Fig.5 Rainfall forecast from (a) Lag PACF input model (b) Stepwise methods input models

From the fig. 5 we can show that rainfall from lag PACF input is more fitted than using stepwise optimize input to Neural Network model.

E. Discussion

The result show that FFNN (single hidden layer) not only fast but also more accurate than DLNN (multiple hidden layers) to estimate only in time series data as explained in Nakama (2011). The developement using stepwise to optimize the lag input for Neural Network. The stepwise process give us a glimps that it could make the process to estimate FFNN faster that without stepwise.

F. Conclussion

Based on the result, at the preprocessing step, it is found that normalized is better as an NN input. The best activation function is logistics with the Rprop+ algorithm. Analyze rainfall data in Wonorejo Reservoir using FFNN is better than ARIMA and DLNN method because it has the minimum RMSEP. Furthermore, the FFNN method was tried using stepwise for selecting the input variable so it is obtained 5 variables from 8 variables were inputted. By using the same architecture, it can be concluded that the input variable selection process using stepwise is not significantly difference in the forecasting results. The number of input lags greatly influences the accuracy of predictions, so stepwise steps on input lags are carried out which results in RMSEP values greater than before, but can shorten the time for the estimation process.

G. References

[1] S. Makridakis and M. Hibon, "The M3-Competition: results, conclusions and implications," *International Journal of Forecasting,* vol. 16, pp. 451-476, 2000.

[2] S. F. Crone and N. Kourentzes, "Input variable selection for time series prediction with neural networks- an evaluation of visual, autocorrelation and spectral analysis for varying seasonality," United Kingdom, 2007.

[3] G. E. P. Box and G. M. Jenkins, "Time series analysis: forecasting and control," San Francisco, 1970.

[4] T. Nakama, "Comparisons of Single- and Multiple-Hidden-Layer Neural Networks," *Procceedings of the 8th International Conference on Advances in neural network,* pp. 270-279, 2011.

[5] U. Anders and O. Korn, "Model selection in neural networks," *Neural Networks,* vol. 12, pp. 309-323, 1999.

[6] W. S. Wei, Time Series Analysis: Univariate and Multivariate Methods, Second Edition, United States: Pearson Education, Inc, 2006.

[7] E. K. Chong and S. H. Zak, An Introduction To Optimization, Canada: John Wiley & Sons, 2001.

[8] R. May, G. Dandy and H. Maier, " Review of Input Variable Selection Methods for Artificial Neural Networks," in *Artificial Neural Networks - Methodological Advances and Biomedical Applications*, Shanghai, China, Intech, 2011, p. 29.

[9] G. Zhang, B. E. Patuwo and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting,* vol. 14, no. 1, pp. 35-62, 1998.