1 Results

The most relevant results of the analysis are shown and organized in this chapter. As a first step, the number of states of the model was decided. Then, a model selection was performed for each given dataset in order to find the most appropriate model, and the parameter that have a switching effect in the model. An analysis of residuals was carried out as a means to validate the models. The results are shown in a later section. Next, the results of a non-parametric analysis are presented, and a comparison between the results of Markov switching model analysis and the results of non-parametric analysis are illustrated. The last two sections report the results of a state prediction of the new observations in each dataset, and an evaluation of the predicting function using a simulated data.

1.1 Analysis I: Number of States

To estimate the set of necessary parameters, an *MSwM*[1] package in R was used. More details about the package can be found in **??**.

A complete linear Markov switching autoregressive model in this thesis framework is defined as

$$\begin{aligned} y_t = &\beta_{intercept,S_t} + \beta_{RrcConnectionSetupComplete,S_t} X_{RrcConnectionSetupComplete,t} \\ &+ \beta_{Paging,S_t} X_{Paging,t} + \beta_{X2HandoverRequest,S_t} X_{X2HandoverRequest,t} \\ &+ \beta_{DuProdName,S_t} X_{DuProdName,t} + \beta_{Fdd/Tdd,S_t} X_{Fdd/Tdd,t} \\ &+ \beta_{NumCells,S_t} X_{NumCells,t} + \phi_{1,S_t} y_{t-1} + \varepsilon_{S_t} \end{aligned} \tag{1.1}$$

The estimation was made under the assumptions of two or three states $S_t \in S$, where $S = 1, 2, .., k$ and $k = 2$ or 3. These two numbers come from a hypothesis that the state of the CPU utilization might have two states (*Normal* and *Bad*, *Normal* and *Good*, *Bad* and *Good*) or three states (*Normal*, *Bad*, and *Good*). During the estimation, a normality assumption was also applied to the distribution of residuals.

BICs from fitting the Markov switching autoregressive model are shown in Tab. 1.1.

[1] https://cran.r-project.org/web/packages/MSwM/index.html

Table 1.1: BIC of the model with two and three states. The left column gives the different datasets.

Software release	BIC	
	$k = 2$	$k = 3$
L16A	439.677	417.682
L16B	1,763.507	1,797.259
L17A	1,189.061	1,199.075

1.1.1 Software release L16A

Before performing the Markov switching autoregressive model, a standard linear regression model was fitted to the dataset first. It was found that one coefficient in the dataset of the software release L16A was not defined because of singularity i.e., a perfect correlation between predictor variables. Hence, *DuProdName* variable was dropped from Equation 1.1.

For the software release L16A, the BIC suggests that the three-state Markov switching autoregressive model gives a better fit in comparison to the two-state model. Fig. 1.1 presents that the Markov chain remained in State 1 for an extensive period of time before it switched to State 2. When the chain is in State 2, it stays there only a short time and then quickly moves back to State 1. There are a few switches between these two states in Fig. 1.1. On the other hand, it is visible that there are more switches between states in Fig. 1.2. Note that State 2 in the two-state model seems to be defined as State 1 in the three-state model instead. Moreover, the periods of State 1, which has a rather long duration in the two-state model, now contains several switches between states in the three-state model.

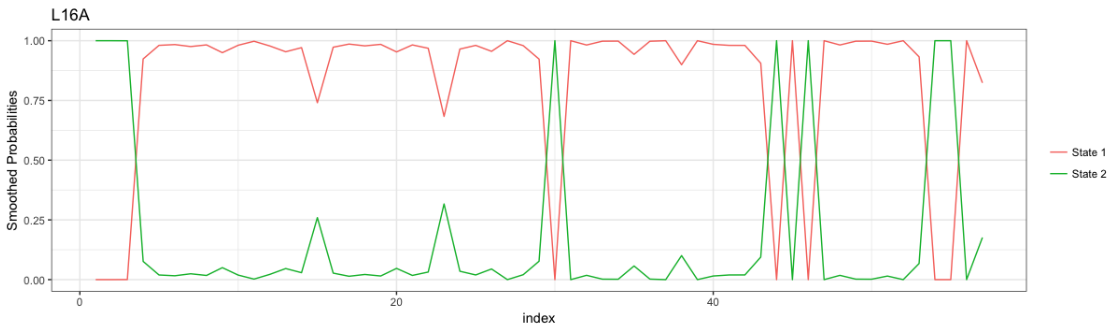

Figure 1.1: The smoothed probabilities of the software release L16A with two-state model

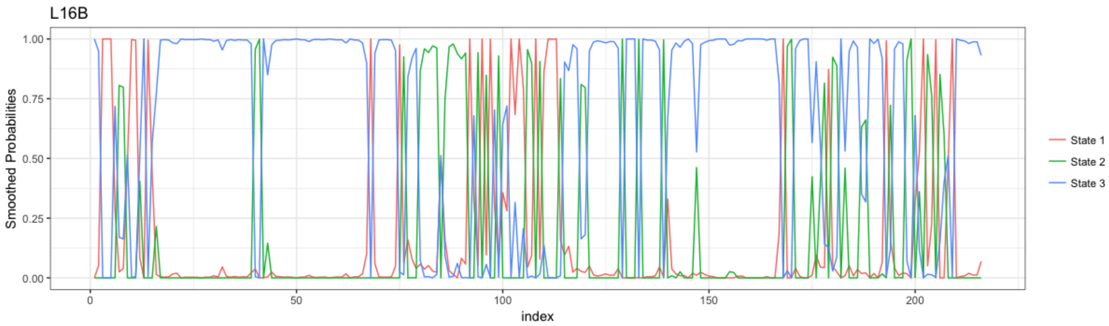

Figure 1.2: The smoothed probabilities of the software release L16A with three-state model

1.1.2 Software release L16B

According to Tab. 1.1, the Markov switching autoregressive models with two states for the remaining two software releases, L16B and L17A, had lower BICs.

In Fig. 1.3, the Markov chain has several periods where it switches back and forth between two states of the software release L16B. The durations of the chain being in State 2 is longer than the durations of the chain staying in State 1. Although the chain temporarily stays in State 1, it remains in this state for a few moments in the middle of the time period (observation 91-99 and 101-114) before returning to State 2. Apparently, there are more switches between states in the three-state model, especially in the beginning, middle, and at the end of the period. Fig. 1.4 shows that the chain remains in State 3 over a considerable period as shown throughout observation 15-39, 42-67, and 140-170.

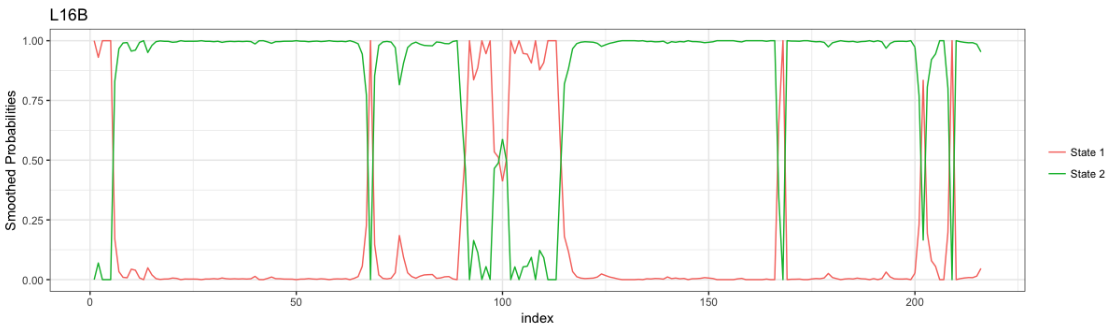

Figure 1.3: The smoothed probabilities of the software release L16B with two-state model

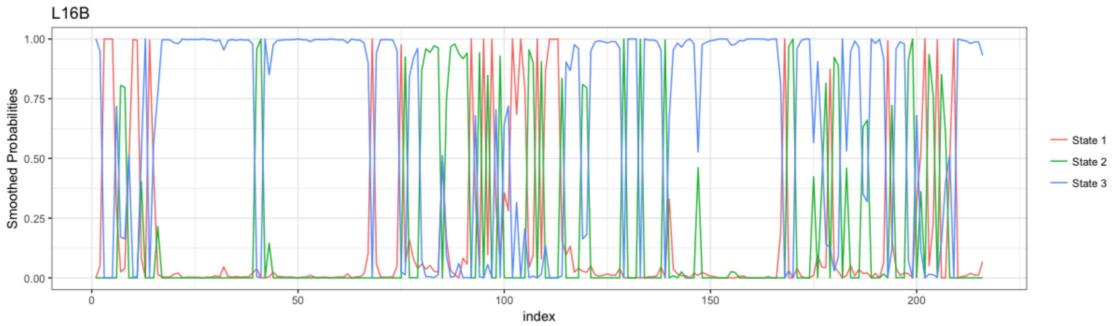

Figure 1.4: The smoothed probabilities of the software release L16B with three-state model

1.1.3 Software release L17A

There are a number of switches between states in the two-state model of the software release L17A. In Fig. 1.5, when the Markov chain is in State 1, it continues to stay in its state for a while before leaving to State 2. Furthermore, the chain has a fairly short duration of staying in State 2. After the chain visits State 2, it instantly switches back to State 1. Fig. 1.6 presents the chain which has many switches between State 1 and State 2 in the first half of the time period. The chain for the three-state model also stays in State 2 significantly long from observation 104 to 129, which is the end of the time series.

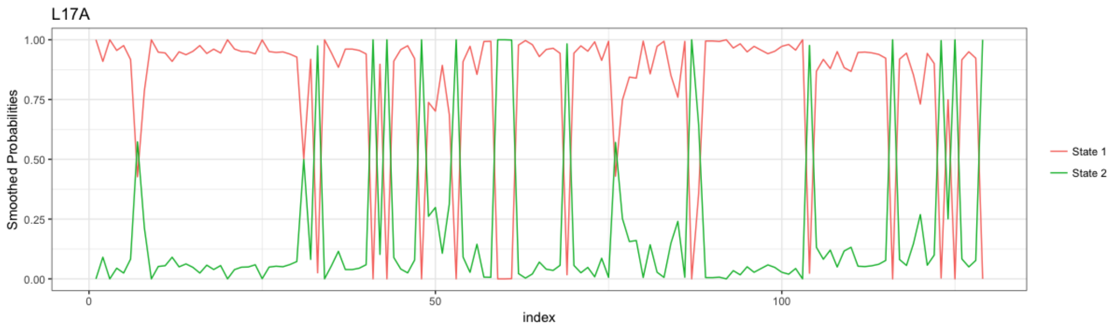

Figure 1.5: The smoothed probabilities of the software release L17A with three-state model

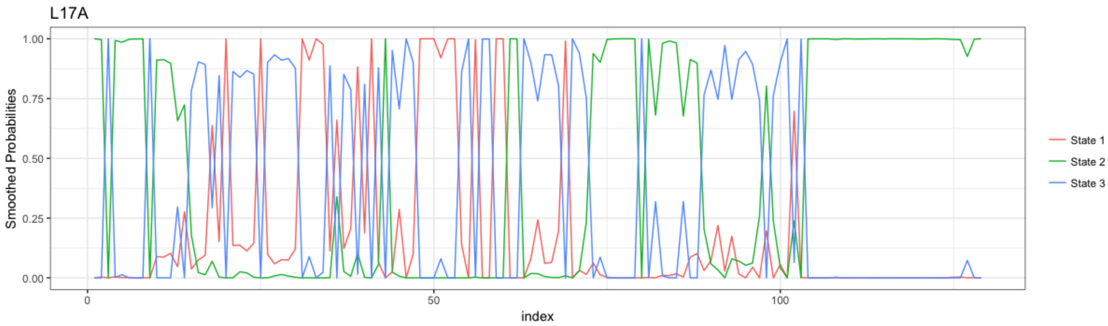

Figure 1.6: The smoothed probabilities of the software release L17A with three-state model

After examining the outputs from the models along with the plots, the three-state models for each software release were further analyzed in the thesis. More details are provided in **??**.

1.2 Analysis II: Number of Switching coefficients

The fitted Markov switching autoregressive models in sec. 1.1 were performed by assuming that every parameter in the model had switching effects i.e., coefficients can have different values in different periods. However, in practice, each coefficient can have either a switching or non-switching effect. Therefore, Markov switching autoregressive models were applied to each dataset again but with a hypothesis that the variables considered as a test environment are possible to have non-switching effects. In this section, the structure of all the models from all three datasets are reported in the table. The best model is selected for each dataset and its state specification is presented in the plots. Further discussion and details about these chosen models are provided in **??**. It should be noted that these three chosen models will later be used throughout this thesis and the model outputs are shown in **??**.

1.2.1 Software release L16A

For the dataset of the software release L16A, *DuProdName* was not included in the model fitting as explained previously. Only two variables of the test environment were left to try whether they could have non-switching effects or not. The result is shown in Tab. 1.2. The second model has the highest BIC and even higher than the model with all switching coefficients. The first model, where both *Fdd/Tdd* and *NumCells* have switching effects, was selected to be used with this dataset.

Fig. 1.7 indicates the CPU utilization of the software release L16A and also shows the periods of the derived state from the model. From the plot, State 2 clearly has the

Table 1.2: List of the model structure of the software release L16A along with its BIC. The line in bold indicates the selected model.

Model	Switching effect		BIC
	Fdd/Tdd	NumCells	
1	**N**	**N**	**413.408**
2	N	Y	438.371
3	Y	N	401.232
	Y	Y	417.682

longest duration to remain in its own state. When the chain moves to either State 1 or State 3, it immediately switches to the other states. However, the duration that the chain stays in State 1 is longer in the beginning and almost at the end of the period. Another characteristic that could be observed is that when State 2 happens to have more chance to switch to State 3 rather than switch to State 1.

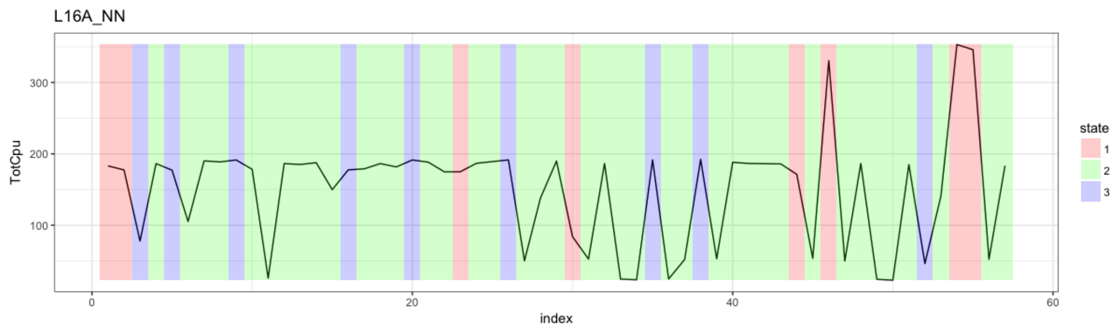

Figure 1.7: The CPU utilization of the software release L16A showing the periods where the observation is in the specific state.
Model 1: Fdd/Tdd and Numcells are non-switching coefficients.

1.2.2 Software release L16B

For the software release L16B, Tab. 1.3 presents the results of fitting the model with different combinations of switching coefficients. Models 5 and 7 have higher BICs than the model which have switching effects in all coefficients. The second model, where *DuProdName* and *Fdd/Tdd* are non-switching coefficients, has the smallest BIC. The chosen model for this dataset is the model which has only *DuProdName* as a non-switching coefficient or model 4.

Many switches between states can easily be seen in Fig. 1.8. However, the state which has the longest duration remaining in its own state is State 3. There are three durations where the chain stays in State 3 for a long time. Another noticeable

Table 1.3: List of the model structure of the software release L16B along with its BIC. The line in bold indicates the selected model.

Model	Switching effect			BIC
	DuProdName	Fdd/Tdd	NumCells	
1	N	N	N	1,787.528
2	N	N	Y	1,704.393
3	N	Y	N	1,784.384
4	**N**	**Y**	**Y**	**1,776.102**
5	Y	N	N	1,806.385
6	Y	N	Y	1,725.865
7	Y	Y	N	1,804.487

behavior from this switching mechanism is that there are several switches between State 1 and State 2 in the beginning, middle, and at the end of the time period.

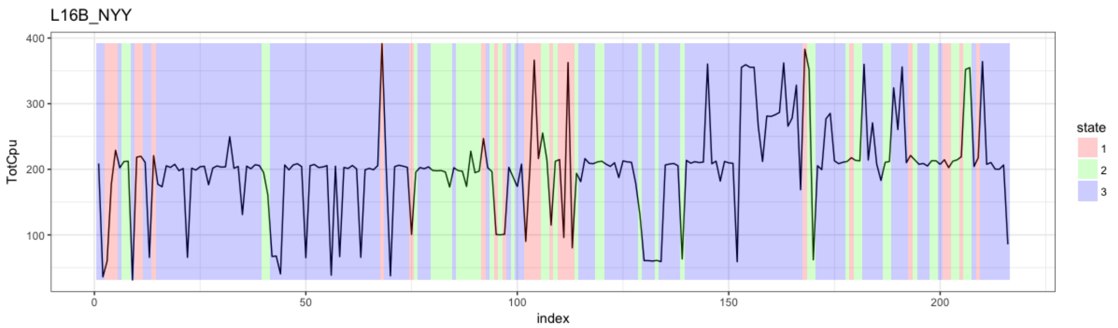

Figure 1.8: The CPU utilization of the software release L16B showing the periods where the observation is in the specific state.
Model 4: DuProdName is non-switching coefficient.

1.2.3 Software relase L17A

Tab. 1.4 presents model structure of the software release L17A. There is only model 2 which has higher BIC than the model with all switching coefficients. The least BIC is from the first model that all three variables in the test environment have non-switching effects. This model was also chosen to be further used for this dataset.

Several switches between three states occur in the beginning of the time series as shown in Fig. 1.9. Around the end of the time series period, State 3 appears to have a longer duration and fewer switches to State 1. State 2 seems to be the only state which has a fairly short duration for the chain to stay in the state. The plot also indicates that State 2 tends to switch to State 1 more often than to switch to State 3.

Table 1.4: List of the model structure of the software release L17A along with its BIC. The line in bold indicates the selected model.

Model	Switching effect			BIC
	DuProdName	Fdd/Tdd	NumCells	
1	**N**	**N**	**N**	**1,140.474**
2	N	N	Y	1,204.280
3	N	Y	N	1,152.740
4	N	Y	Y	1,184.643
5	Y	N	N	1,146.000
6	Y	N	Y	1,189.236
7	Y	Y	N	1,157.311

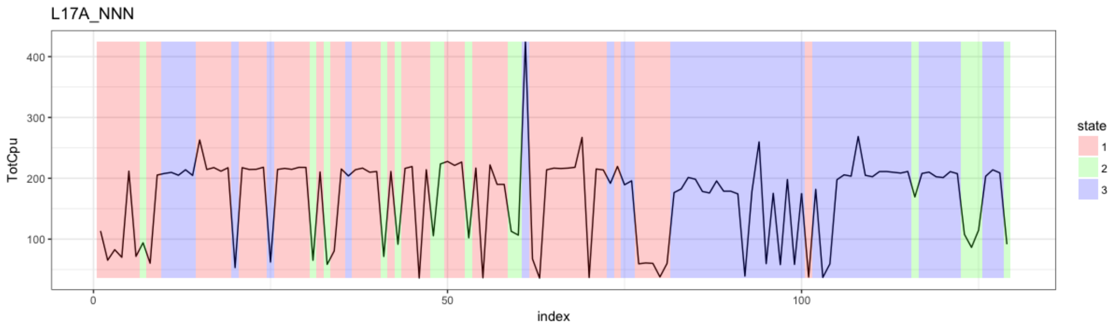

Figure 1.9: The CPU utilization of the software release L17A showing the periods where the observation is in the specific state.
Model 1: DuProdName, Fdd/Tdd, and NumCells are non-switching coefficients.

1.3 Residual analysis

Pooled residuals of the selected Markov switching autoregressive model from sec. 1.2 were analyzed to see how well the model fitted an assumption of a normal distribution. A Quantile-Quantile (Q-Q) plot is an effective tool for assessing normality. Moreover, an Autocorrelation function (ACF) and a Partial Autocorrelation Function (PACF) of residuals are a useful technique to check on the independence of noise terms in the model. The Q-Q plot and the ACF/PACF plot play a significant role in the residual diagnostics. These plots of each dataset are shown below.

In Fig. 1.10, the pooled residuals appeared to fall in a straight line with some deviations in its tails. There was an evidence of autocorrelation in the residuals of this model, which can be seen in both ACF and PACF plot, at lag 8.

Fig. 1.11 presents points that formed a straight line in the middle of the plot, but curved off at both ends. This is a characteristic of a heavy-tailed distribution. The data has more extreme values than it should be if the data truly comes from a

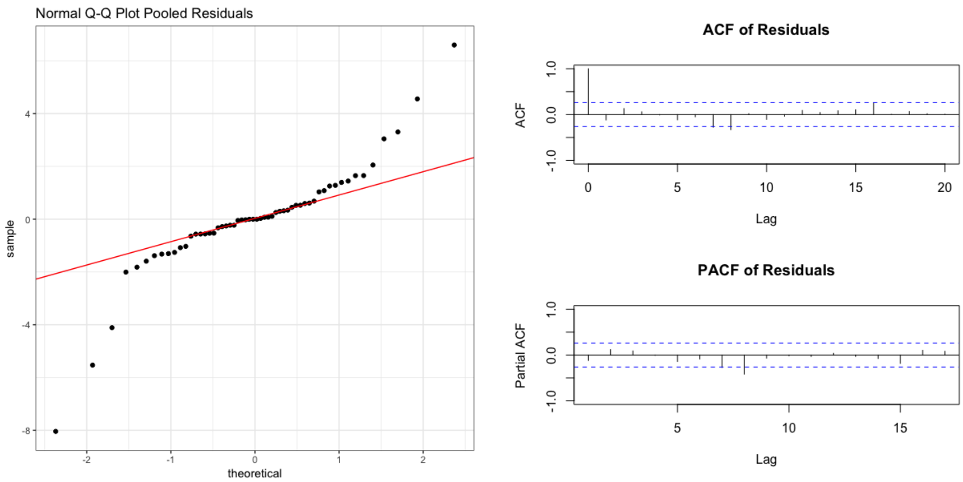

Figure 1.10: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L16A

normal distribution. In addition, both the ACF and PACF plot show that there was a small amount of autocorrelation remaining in the residuals. The statistically significant correlation of this model were at lags 6 and 10. The significant at lag 4 both in the ACF and PACF plot was slightly higher than two standard errors.

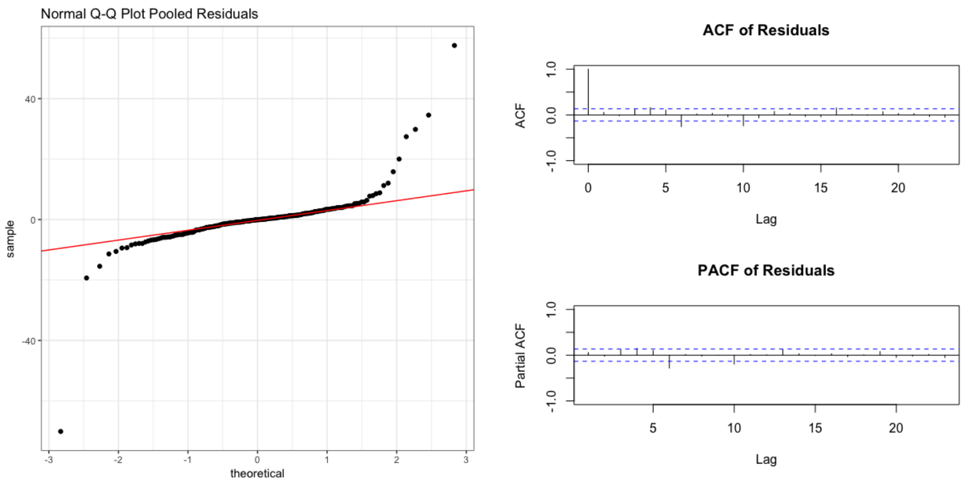

Figure 1.11: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L16B

A Q-Q plot in Fig. 1.12 suggests that a distribution of the pooled residuals may have a tail thicker than that of a normal distribution. It is visible that there were many extreme positive and negative residuals in the plot. Furthermore, the ACF plot of pooled residuals were significant for the first two lags, whereas the PACF plot was significant only at lag 2.

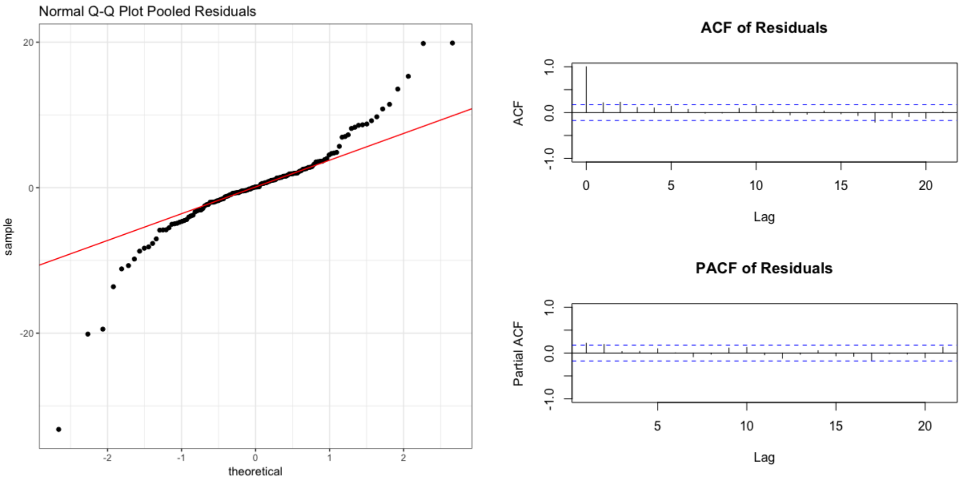

Figure 1.12: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L17A

1.4 Non-parametric analysis

An E-divisive method was applied to all three datasets. The method reported one cluster for the dataset of the software release L16A. There were five clusters found in both datasets of the software release L16B and L17A. Tab. 1.5 shows places in the time series data where the E-divisive algorithm was able to detect the significant changes.

Table 1.5: The locations of the statistically significant change points from applying the E-divisive algorithm in each dataset

Software release	Change-point location
L16A	-
L16B	130, 135, 153, 170
L17A	9, 77, 82, 105

The CPU utilization of the software release L16A, L16B and L17A along with its estimated change points in the time series are plotted and shown in Fig. 1.13, Fig. 1.14 and Fig. 1.15, respectively. It can be seen that the E-divisive method could not identify any changes in the dataset of the software release L16A.

Four change points were identified from the method for the software release L16B and L17A. In Fig. 1.14, the estimated change points for the dataset of the software release L16B were likely to occur around the same period of time, which were almost at the end of the time series data. The estimated points from the method were approximately at peaks and negative peaks.

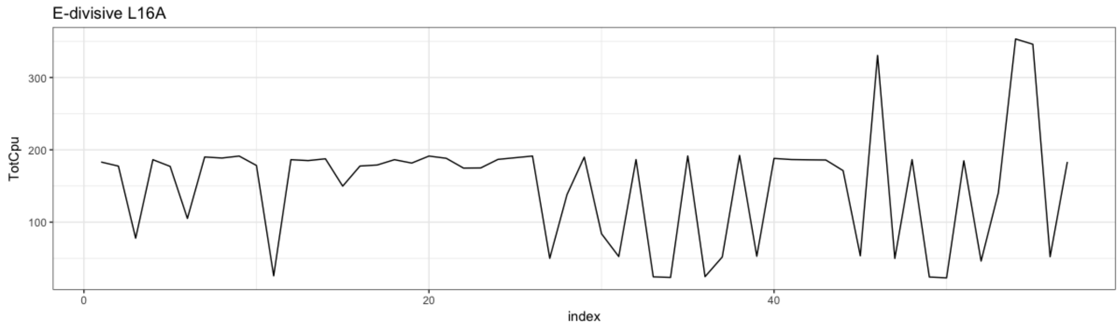

Figure 1.13: The CPU utilization of the software release L16A

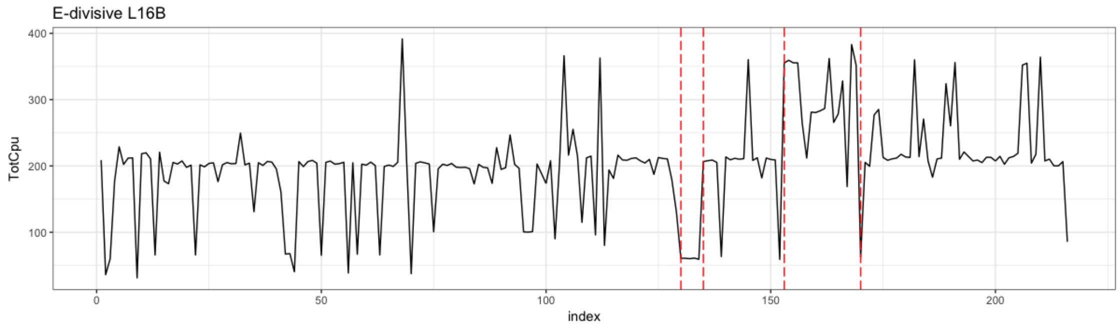

Figure 1.14: The CPU utilization of the software release L16B. The red dashed vertical lines indicate the locations of estimated change points.

On the contrary, the result of the dataset of the software release L17A, which is shown in Fig. 1.15, had the estimated change points rather spread out. The E-divisive method discovered changes when the CPU utilization was about to drop or increase its value.

1.4.1 Comparison between the Markov switching autoregressive model and the E-divisive

One noticeable thing is that the E-divisive method was able to identify the changes of the data less than the Markov switching autoregressive model. As mentioned previously, no estimated change points was discovered when applying the E-divisive algorithm to the dataset of the software release L16A. Thus, a comparison between two methods could not be made for this dataset.

Fig. 1.16 presents results of switches between states from the Markov switching autoregressive model and change point locations from the E-divisive method for the software release L16B. There was one location where the E-divisive method detected a change but the Markov switching autoregressive model did not show any switches. As can be seen, the E-divisive method appeared to detect changes when

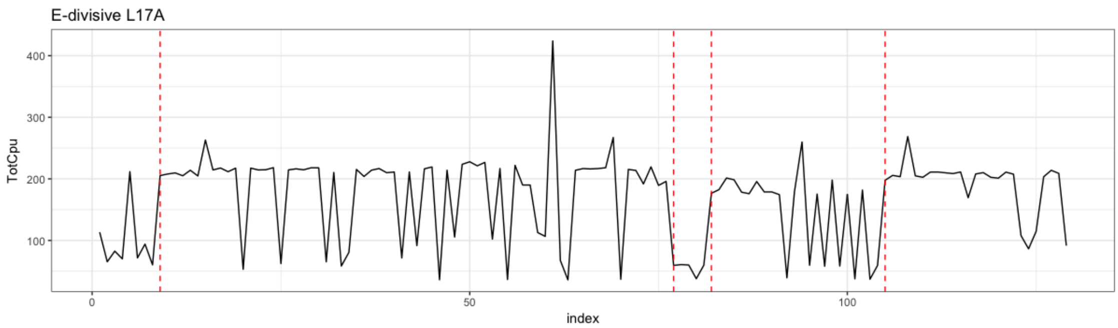

Figure 1.15: The CPU utilization of the software release L17A. The red dashed vertical lines indicate the locations of estimated change points.

State 2 switches to State 3. At observation 130, the E-divisive method discovered a switch in the state at the same time as Markov switching autoregressive model did. However, the E-divisive method identified switches after and before the Markov switching autoregressive model did at observations 135 and 170, respectively.

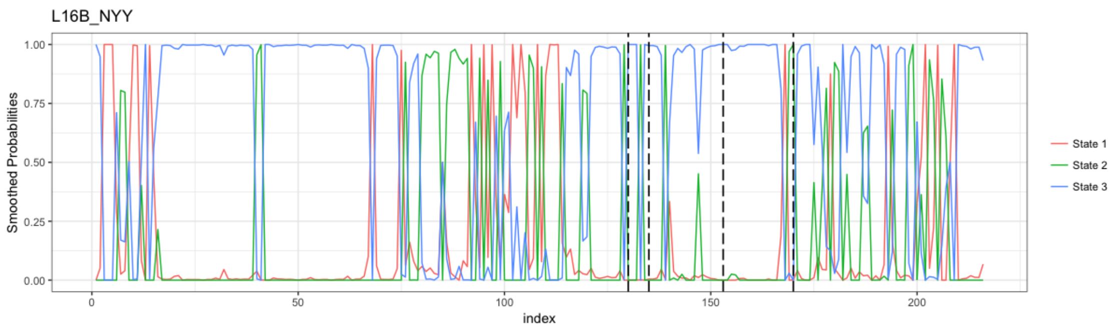

Figure 1.16: The combined results of the Markov switching autoregressive model and the E-divisive method for the software release 16B

In Fig. 1.17, the E-divisive method reported a change at observation 105 whereas no switch between states could be found from the result of the Markov switching autoregressive model. It is visible that the E-divisive method was able to detect a switch from State 1 to State 3, and also a switch from State 3 to State 1. Both two methods could detect the changes at the same period of time at observations 77 and 82. The E-divisive method detected a change at observation 9 which was before there was a switch in the state from the result of the Markov switching autoregressive model.

1.5 Predicting the state

Now, an implemented of state prediction function was applied to the test set in order to find the most probable state for new observations. For the software release L16A,

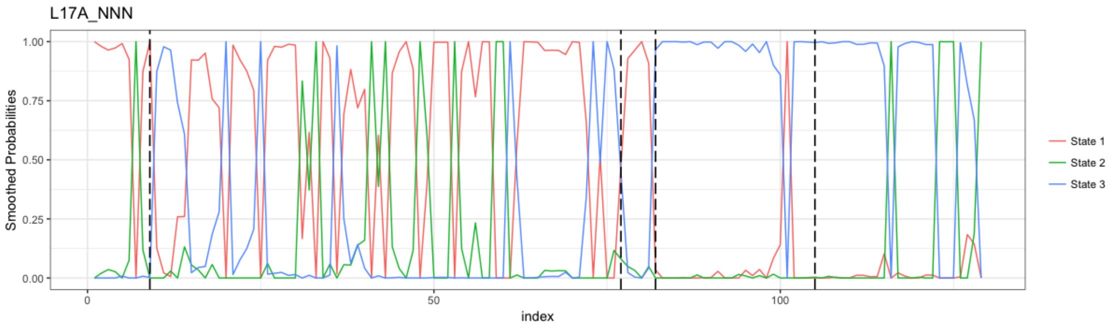

Figure 1.17: The combined results of the Markov switching autoregressive model and the E-divisive method for the software release 17A

there are 7 observations in a test set. In Fig. 1.18, only two states, which were State 1 and State 2, were assigned for these observations. The first three observations were in State 2. Afterwards, observation tended to switch back and forth between states until the end. It is noticed that the last observation did not belong to any state. After applying a predict function to the test set, the function was unable to predict the most likely state for the last observation of the test set.

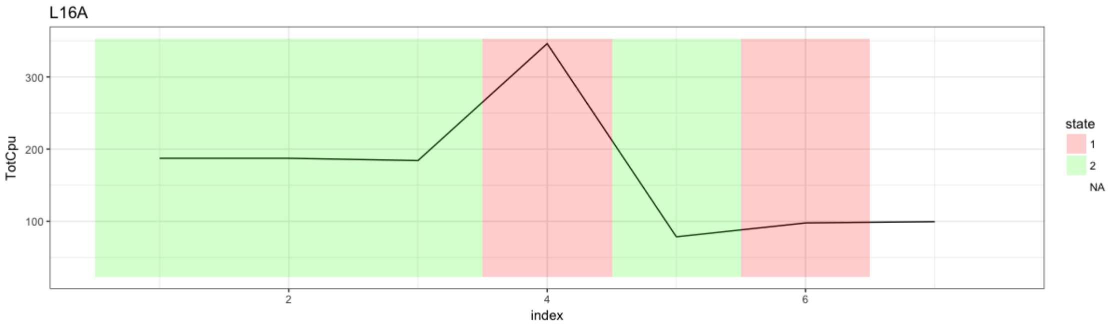

Figure 1.18: The predicted state of the test set in the software release L16A

In total, there are 25 observations in a test set of the software release L16B. The result after applying the predict function to the test set is shown in Fig. 1.19. Observation 15 was the only observation which was in State 2. Many switches between State 1 and State 2 can be seen from the plot. In addition, observation appeared to stay in State 1 only a short time before moving to State 3, except for the first five observations.

Fifteen observations is in a test set of the software release L17A. Fig. 1.20 presents a considerably long period for staying in State 2, which was from observation 10 to the end of the time series data. There were several switches between states happening in the plot. As can be seen, observation between 4 and 7 swapped between states fairly quick. Observation visited the particular state for one time and then moved to the other states.

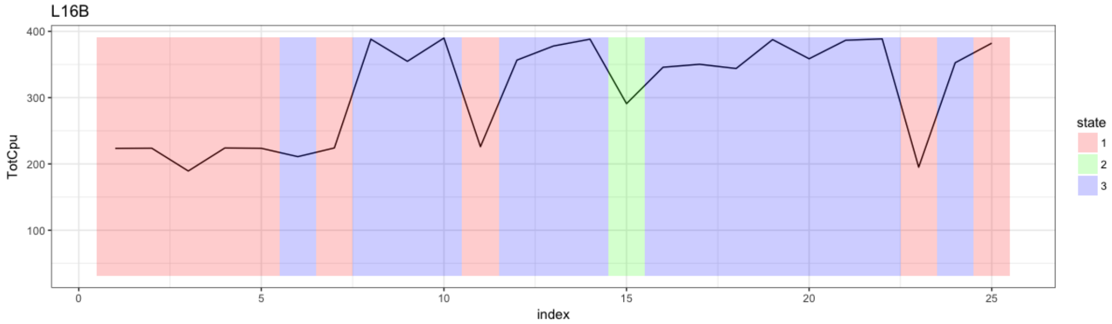

Figure 1.19: The predicted state of the test set in the software release L16B

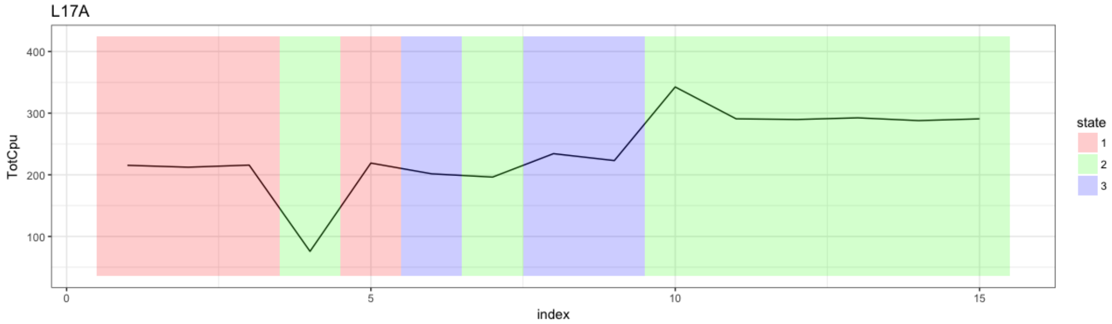

Figure 1.20: The predicted state of the test set in the software release L17A

1.6 Assessing state prediction using a simulation technique

Markov switching autoregressive model was fitted with eighty percents of the observations from a simulated data, and the remaining was used as a test set to evaluate a performance of the model. The result of the model performance is shown in Tab. 1.6. There were two observations from the *Bad* state which were wrongly classified to the Normal state. Moreover, another two observations from the *Good* state were classified to the *Normal* state. The overall accuracy of the model was 0.96, and the misclassification rate was 0.04. One can see that the model was able to correctly predict the observations which had *Bad* and *Good* state.

Tab. 1.7 presents a confusion matrix for a test set from a simulated data. One can see that the model able to correctly predict the observations which had *Bad* state. On the contrary, the model did not perform well in predicting observation which had *Good* state. Nine observations were classified to *Bad* state while another five observations were classified to *Normal* state. Moreover, there were six observations from the *Normal* state which were wrongly classified to the *Good* state. The overall accuracy of the model and the misclassification rate was 0.8 and 0.2, respectively.

Table 1.6: Confusion matrix after applying the Markov switching autoregressive model to fit with the test set

		Predicted state		
		Bad	Normal	Good
Actual state	Bad	58	2	0
	Normal	0	30	0
	Good	0	2	8

Table 1.7: Confusion matrix after applying the Markov switching autoregressive model to fit with the test set from the second simulated data

		Predicted state		
		Bad	Normal	Good
Actual state	Bad	35	0	0
	Normal	0	29	6
	Good	9	5	16