Forecasting - Assignment 2

Nisa Rachmatika (s3570512)

Part I - Introduction

This report is written based on two datasets: first, dataset about amount of horizontal solar radiation reaching the ground at a particular location over the globe between January 1960 and December 2014. Second, quarterly Residential Property Price Index (PPI) in Melbourne and quarterly population change over previous quarter in Victoria between September 2003 and December 2016.

With forecasting method, this report tried to answer two objectives:

First, to give best 2 years ahead forecasts in terms of MASE for the solar radiation series by using the time series regression methods (distributed lag models (dLagM package)), dynamic linear models (dynlm package), and exponential smoothing and corresponding state space model.

Second, to analyze correlation between Residential PPI in Melbourne and population change in Victoria: whether the correlation between these two series is spurious or not.

Part II - Discussions and Results

Task 1

Time Series Plot Exploration

The data is being converted to time series object first.

```
solars<-read_csv("data1.csv")
s<- ts(solars$solar,start = c(1960,1),frequency = 12)
ppt = ts(solars$ppt,start = c(1960,1),frequency = 12)

solarppt = ts(solars[,1:2],start = c(1960,1),frequency = 12)
plot( solarppt, ylab="Solar Radiation", xlab = "Year", main = "Time series plot of
s olar radiation and precipitation series", type="l", yax.flip=T)
```

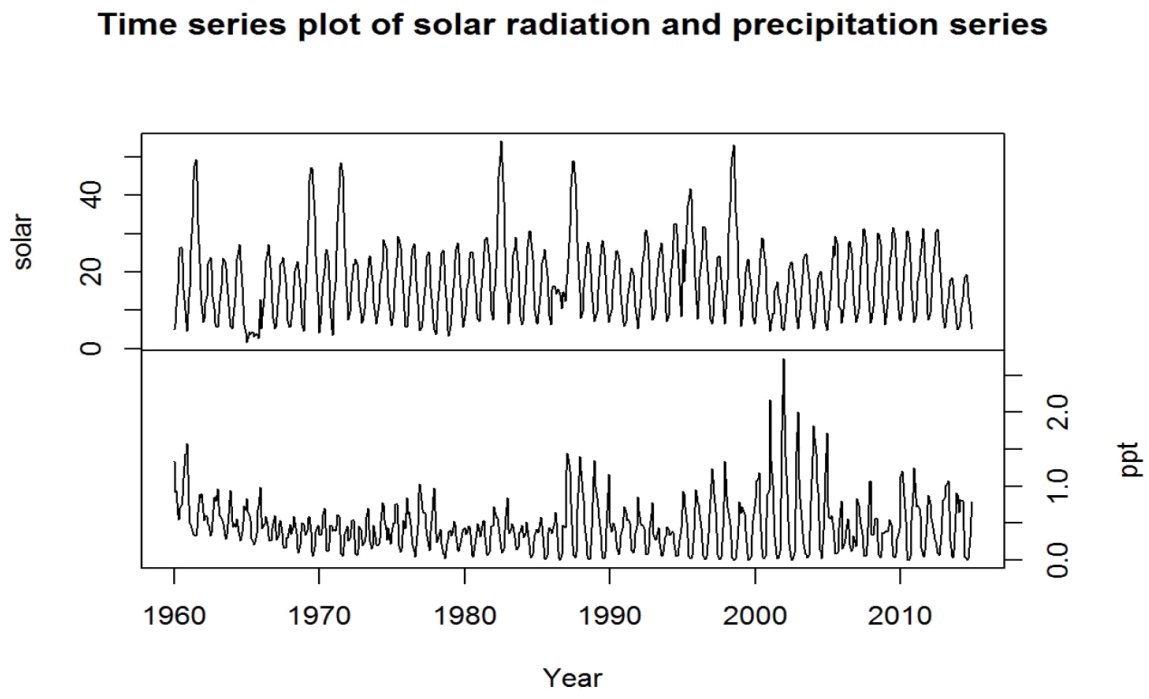


Figure1: Solar Radiation & Precipitation Series

There are no obvious trend visible in both solar radiation and precipitation series, since the series are bouncing around mean level. Also, the intervention is not clearly visible. However, the seasonal pattern and changing variance is appear.
Now the correlation of both series will be examined.

```
cor(solarppt)
```

```
##            solar        ppt
## solar  1.0000000 -0.4540277
## ppt   -0.4540277  1.0000000
```

Solar radiation and precipitation data is negatively correlated, means as precipitation value decreases, solar radiation value will also decreases. However, this correlation is only moderate.

The next steps is modelling using Distributed Lag Models (DLM), dynamic linear models, and exponential smoothing and corresponding state space models.

1. Distributed Lag Models (DLM)

1.a. Normal Distributed Lag Models

The first thing needs to be determined in normal DLM modelling is to specify number of lags. Here, because the correlation is moderate, we will start with moderate number of lags. We can also choose the appropriate lag by observing the goodness of fit in Adjusted R-squared value below:

```
ppt<-as.vector(ppt)
solar<-as.vector(s)
for (i in 2:12){
  lags<-finiteDLMauto(ppt, solar, q.min = 1, q.max = i,
                      model.type = c("dlm"), error.type = c("AIC"), trace = TRUE)
}
lags
```

```
##      q    MASE      AIC      BIC R.Adj.Sq Ljung-Box
## 12  12 1.55160 4578.787 4645.895  0.30769         0
## 11  11 1.56213 4590.961 4653.617  0.30199         0
## 10  10 1.57800 4602.658 4660.858  0.29615         0
## 9    9 1.59312 4615.084 4668.827  0.28882         0
## 8    8 1.60481 4625.986 4675.267  0.28251         0
## 7    7 1.60704 4632.716 4677.532  0.28096         0
## 6    6 1.60753 4637.489 4677.837  0.28214         0
## 5    5 1.61385 4644.622 4680.499  0.28072         0
## 4    4 1.64636 4663.600 4695.003  0.26577         0
## 3    3 1.66270 4688.551 4715.478  0.24326         0
## 2    2 1.67597 4712.649 4735.095  0.22173         0
## 1    1 1.68846 4728.713 4746.676  0.21036         0
```

Here we can see that from the lag 1, adjusted R-squared values keep increasing, means the model keep fitting the data better.However, after lag 5, the increase rate becomes slower. Hence, we chose lag 5 as starting point.

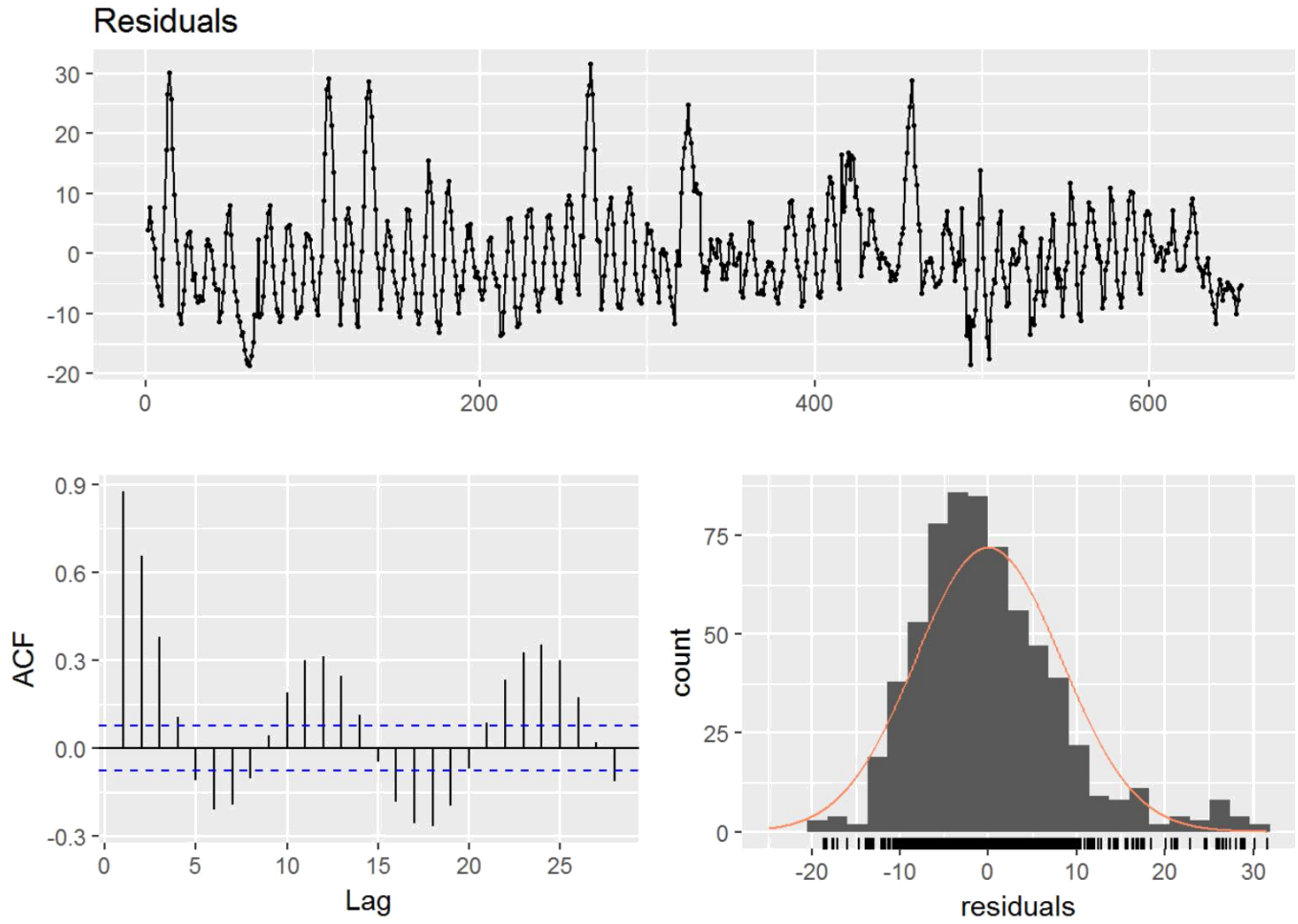
DLM model with 5 lags

```
m1 = dlm(x = as.vector(ppt) , y = as.vector(solar), q = 5 , show.summary = TRUE)
```

```
 ##
 ## Call:
 ## lm(formula = y.t ~ ., data = design)
##
 ## Residuals:
 ##     Min       1Q  Median      3Q     Max
 ## -18.587   -5.811  -1.331   4.306  31.606
 ##
 ## Coefficients:
 ##             Estimate Std. Error t value Pr(>|t|)
 ## (Intercept)  18.0351     0.8413  21.437  < 2e-16 ***
 ## x.t         -10.5535     1.7350 -6.083 2.02e-09 ***
 ## x.1           0.5363     2.5297   0.212 0.832166
 ## x.2           0.4982     2.5838   0.193 0.847167
 ## x.3           1.5183     2.5766   0.589 0.555878
 ## x.4           0.8421     2.5171   0.335 0.738073
 ## x.5           6.6487     1.7228   3.859 0.000125 ***
 ## ---
 ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
 ## Residual standard error: 8.329 on 648 degrees of freedom
 ## Multiple R-squared:  0.2873, Adjusted R-squared:  0.2807
 ## F-statistic: 43.54 on 6 and 648 DF,  p-value: < 2.2e-16
 ##
 ## AIC and BIC values for the model:
 ##        AIC      BIC
 ## 1 4644.622 4680.499
```

Here, even though p-value indicates the model is significant, most of the coefficient values are insignificants. Also, the adjusted R-squared is very low. Hence, the model result is assumed not that good, and we can check the residuals to prove this assumption.

```
checkresiduals(m1$model)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 10
##
 ## data:  object
 ## LM test = 590.75, df = 10, p-value < 2.2e-16
```

Figure2: Residuals Plot of Model 1

As assumed before, this model have significant residuals, confirmed by BG-test result. From the residual plots, there are visible trend and changing variance, also there are significant correlations in ACF plots. The histogram also not normally distributed.

All of the test indicates the model is not good enough in fitting the data. Hence, we will try another lag numbers. Here I proposed to decrease and increase 1 step of the lag.

```
m2 = dlm(x = as.vector(ppt) , y = as.vector(solar), q = 4 , show.summary = TRUE)
```

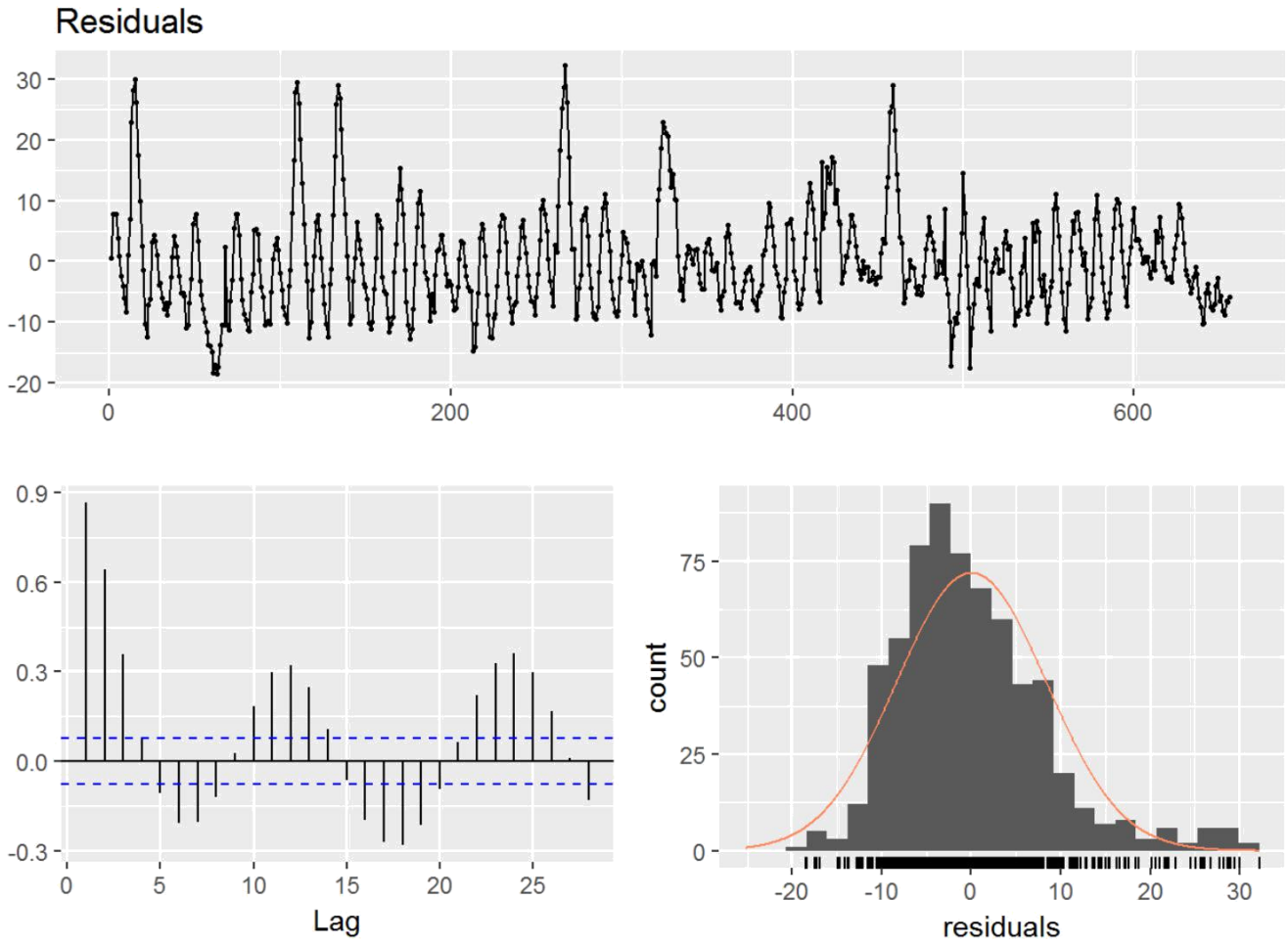
```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min       1Q  Median      3Q     Max
## -18.418   -5.743  -1.444   4.398  32.225
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.3727     0.7723  25.086  < 2e-16 ***
## x.t         -10.8482     1.7492 -6.202 9.93e-10 ***
## x.1           0.3550     2.5524   0.139    0.889
## x.2          -0.2807     2.5946  -0.108    0.914
## x.3          -0.4812     2.5406  -0.189    0.850
## x.4           7.9026     1.7380   4.547 6.49e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.411 on 650 degrees of freedom
## Multiple R-squared:  0.2714, Adjusted R-squared:  0.2658
## F-statistic: 48.42 on 5 and 650 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##      AIC      BIC
## 1 4663.6 4695.003
```

```
m3 = dlm(x = as.vector(ppt) , y = as.vector(solar), q = 6 , show.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min       1Q  Median      3Q     Max
## -19.136   -5.796  -1.202   4.354  31.403
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.4381     0.9244  18.865 < 2e-16 ***
## x.t         -10.3695     1.7430 -5.949 4.42e-09 ***
## x.1           0.4397     2.5294   0.174   0.8621
## x.2           0.6780     2.5840   0.262   0.7931
## x.3           1.7948     2.5913   0.693   0.4888
## x.4           1.8136     2.5767   0.704   0.4818
## x.5           3.5003     2.5159   1.391   0.1646
## x.6           2.8893     1.7274   1.673   0.0949 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.322 on 646 degrees of freedom
## Multiple R-squared:  0.2898, Adjusted R-squared:  0.2821
## F-statistic: 37.66 on 7 and 646 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4637.489 4677.837
```

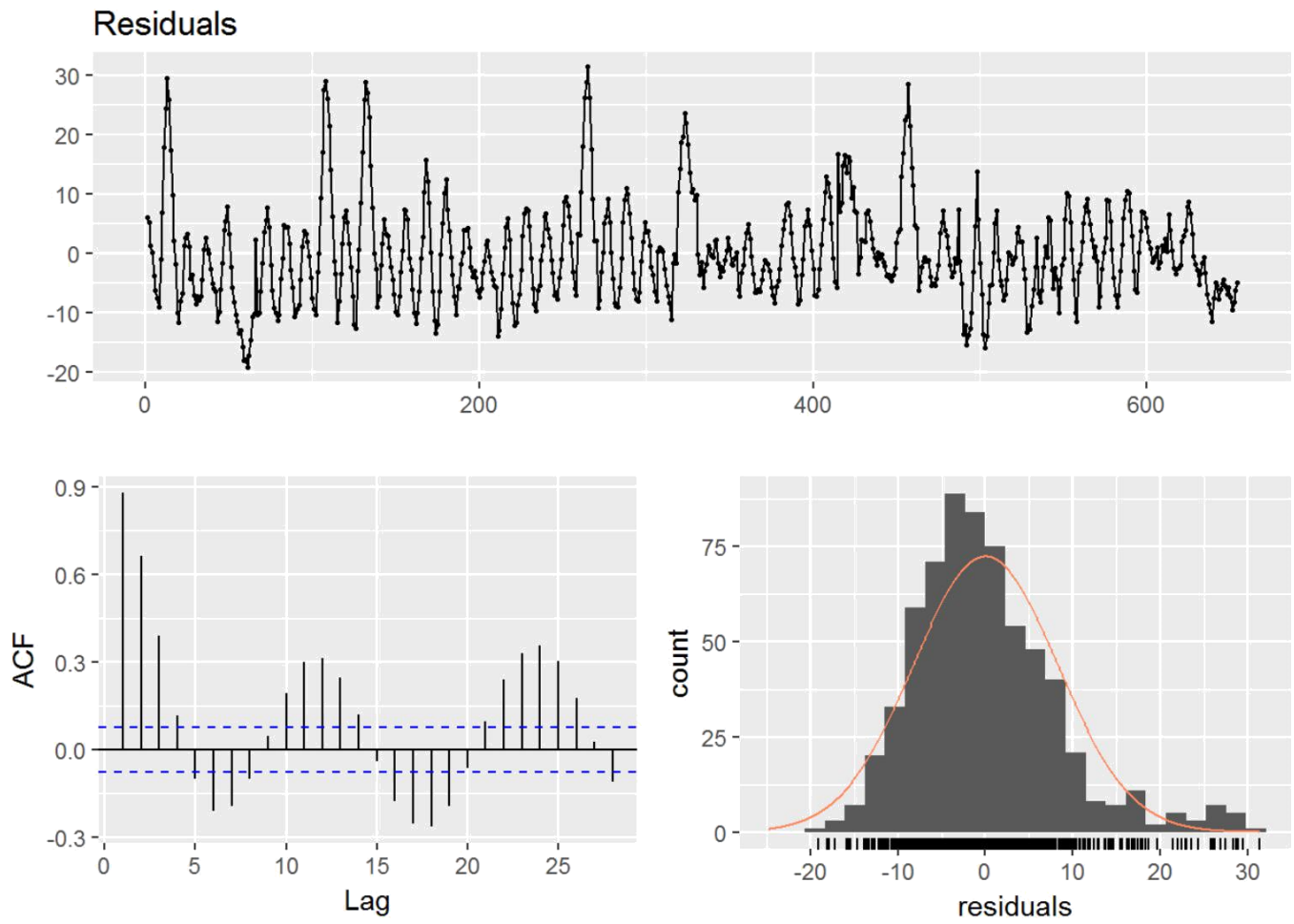
Similar with the first model above, even though p-value indicates model2 and model3 are significant, most of the coefficient values are insignificants. However, even though the adjusted R-squared is still low, the value is slightly increase in model3, with lag 6. Because the test result is similar with model1, here we assumed that the residual results will not have a big difference also.

```
checkresiduals(m2$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  object
## LM test = 586.43, df = 10, p-value < 2.2e-16
```

```
checkresiduals(m3$model)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 11
##
 ## data:  object
 ## LM test = 592.05, df = 11, p-value < 2.2e-16
```

Figure3: Residuals Plot of Model 2 and 3

As predicted, these models have significant residuals, confirmed by BG-test results. Also, the trend and changing variance is still visible, and there are significant correlations in ACF plots. Furthermore, the histogram is still not normally distributed.

Now we will check multicollinearity of these 3 models:

```
vif(m1$model)
```

```
##      x.t      x.1      x.2      x.3      x.4      x.5
## 3.550236 7.538781 7.865358 7.841270 7.484421
```

```
3.528685 vif(m2$model)
```

```
 ##      x.t      x.1      x.2      x.3      x.4
 ## 3.538721 7.527187 7.799334 7.494641 3.531005
```

```
vif(m3$model)
```

```
## x.t x.1 x.2 x.3 x.4 x.5 x.6 ## 3.586454 7.548252 7.877271
7.921428 7.836035 7.469526 3.544996
```

Interestingly, all of the models show that all of the value is under 10, means the effect of multicollinearity is low. Because all of 3 models seems gives almost similar performance, we will pick the best model from these DLM approaches based on their AIC, BIC and MASE values.

```
aic.models = AIC(m1$model, m2$model, m3$model)
sortScore(aic.models, score="aic")
```

```
##          df      AIC
## m3$model  9 4637.489
## m1$model  8 4644.622
## m2$model  7 4663.600
```

```
## $df
## [1] 9 8 7
##
## $AIC
## [1] 4637.489 4644.622 4663.600
##
## $call
## sortScore.default(x = aic.models, score = "aic")
##
## attr(,"row.names")
## [1] "m3$model" "m1$model" "m2$model"
## attr(,"class")
## [1] "sortScore" "dLagM"
```

```
bic.models = BIC(m1$model, m2$model, m3$model)
sortScore(bic.models, score="bic")
```

```
##          df      BIC
## m3$model  9 4677.837
## m1$model  8 4680.499
## m2$model  7 4695.003
```

```
## $df
## [1] 9 8 7
##
## $BIC
## [1] 4677.837 4680.499 4695.003
##
## $call
## sortScore.default(x = bic.models, score = "bic")
##
## attr(,"row.names")
## [1] "m3$model" "m1$model" "m2$model"
## attr(,"class")
## [1] "sortScore" "dLagM"
```

```
MASE(m1,m2,m3)
```

```
##      n     MASE
## m1 655 1.613848
## m2 656 1.646357
## m3 654 1.607532
```

All of these 3 tests gives same conclusions, that model “m3” is the best model with the lowest AIC, BIC, and MASE Hence, in DLM approach, lag 6 seems will be a better fit.

1.b. Polynomial Distributed Lags

Even though DLMs models above indicates the effect of multicollinearity is low, we still will try using Polynomial Distributed Lags to see whether the effect of multicollinearity can be reduced. First, we will apply second order polynomial with lag order determined below:

```
for (i in 2:12){
poly<-finiteDLMauto(ppt, solar, q.min = 1, q.max = i, k.order =
2, model.type = c("poly"), error.type = c("AIC"), trace = TRUE) }

poly
```

```
##     q - k     MASE      AIC      BIC R.Adj.Sq Ljung-Box
## 12 12 - 2 1.56304 4567.969 4590.339   0.30871         0
## 11 11 - 2 1.57046 4577.156 4599.533   0.30727         0
## 10 10 - 2 1.59255 4591.904 4614.289   0.29924         0
## 9   9  - 2 1.60880 4607.861 4630.253  0.28915         0
## 8   8  - 2 1.61903 4620.470 4642.870  0.28205         0
## 7   7  - 2 1.61909 4627.974 4650.382  0.28073         0
## 6   6  - 2 1.61999 4634.526 4656.942  0.28104         0
## 5   5  - 2 1.63194 4645.250 4667.673  0.27675         0
## 4   4  - 2 1.65329 4664.741 4687.171  0.26226         0
## 3   3  - 2 1.66635 4689.018 4711.457  0.24157         0
## 2   2  - 2 1.67597 4712.649 4735.095  0.22173         0
## 1   1  - 2 1.68846 4728.713 4746.676  0.21036         0
```

As explained in the previous step, the Adjusted R-Squared values are increasing slowly at lag 6. Hence, I will chose 2nd order polynomial with lag 6 as a starter.

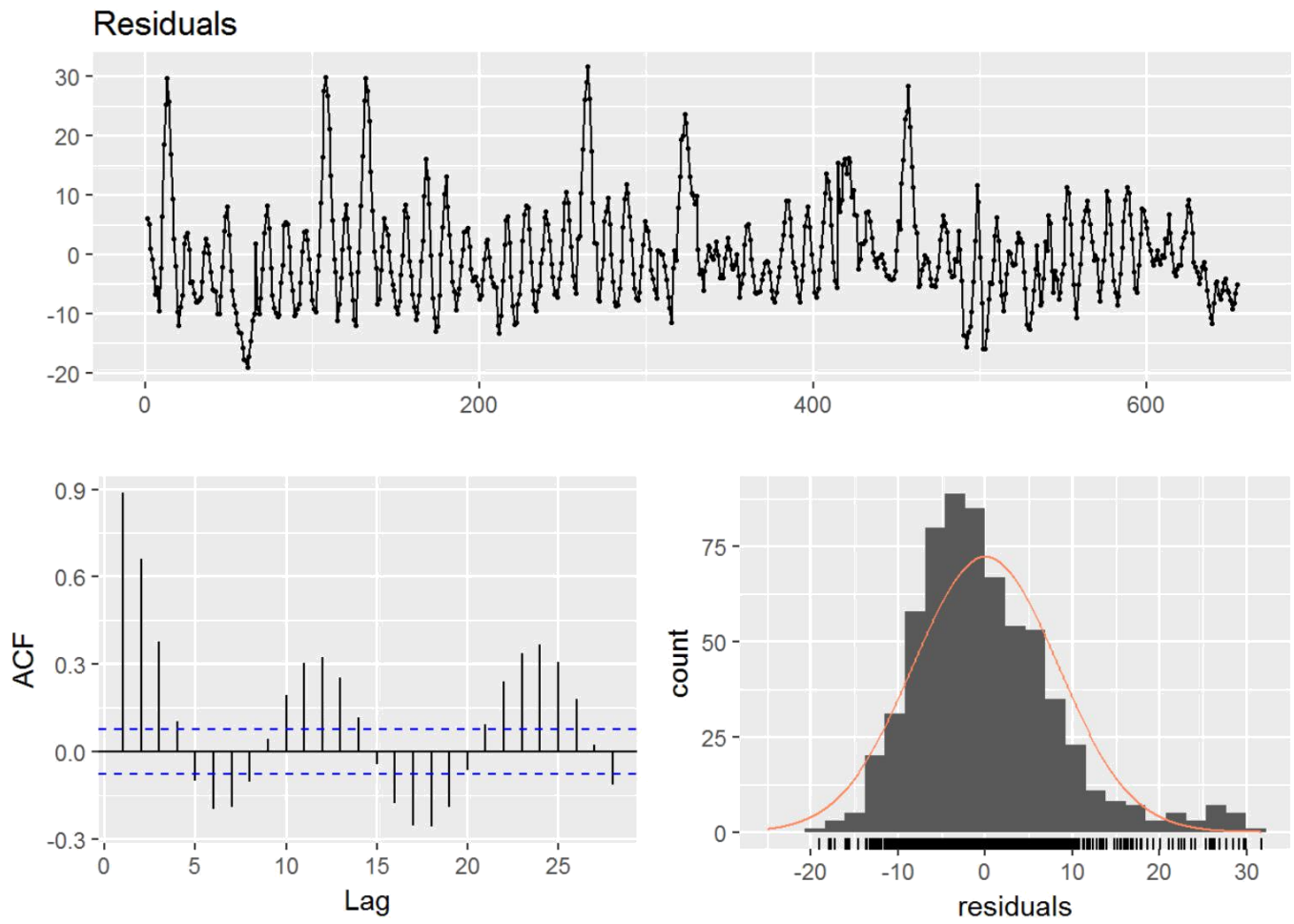
```
m4 = polyDlm(x=ppt, y=solar, q=6, k=2, show.beta = TRUE , show.summary = TRUE)
```

```
 ##
 ## Call:
 ## lm(formula = y.t ~ ., data = z)
##
 ## Residuals:
 ##     Min       1Q  Median      3Q     Max
 ## -18.994   -5.799  -1.272   4.440  31.675
 ##
 ## Coefficients:
 ##              Estimate Std. Error t value Pr(>|t|)
 ## (Intercept)  17.2263     0.9050  19.034  < 2e-16 ***
 ## z.t0          -7.2534     0.7906  -9.175  < 2e-16 ***
 ## z.t1           4.8900     0.6769   7.224 1.41e-12 ***
 ## z.t2          -0.5573     0.1115 -5.000 7.38e-07 ***
 ## ---
 ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
 ## Residual standard error: 8.329 on 650 degrees of freedom
 ## Multiple R-squared:  0.2843, Adjusted R-squared:  0.281
 ## F-statistic: 86.09 on 3 and 650 DF,  p-value: < 2.2e-16
 ##
 ## Estimates and t-tests for beta coefficients:
 ##        Estimate Std. Error t value  P(>|t|)
 ## beta.0    -7.250      0.791  -9.17 5.96e-19
 ## beta.1    -2.920      0.333  -8.78 1.45e-17
 ## beta.2     0.298      0.317    0.94 3.48e-01
 ## beta.3     2.400      0.383    6.28 6.30e-10
 ## beta.4     3.390      0.316   10.70 8.35e-25
 ## beta.5     3.270      0.332    9.84 2.27e-21
 ## beta.6     2.030      0.790    2.56 1.06e-02
```

By putting second order polynomial, all of beta coefficients are significant, also p-value is significant. However, the adjusted R-squared value is still small, and it's even smaller than model3 in DLM approach.

Now we will check the residuals, VIF, and MASE of this model.

```
checkresiduals(m4$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  object
## LM test = 580.01, df = 10, p-value < 2.2e-16
```

```
vif(m4$model)
```

```
##      z.t0      z.t1      z.t2
##  12.79995 126.60080 77.48449
```

```
MASE(m4$model)
```

```
##                MASE
## m4$model 1.619987
```

Figure4: Residuals plot of Model 4

BG-test indicates significant residuals result. The inference is still not further from the last 3 models: the trend and changing variance is visible, and the histogram is not normally distributed. However, all of the VIF value is higher than 10, and MASE is higher than the best model in DLM model before. Hence, this polynomial model is not good enough to beat the best model in DLM approach before. Here, I did not increase the polynomial order to avoid overfitting. Also, I did not change the lag order since last DLM approach indicates model with lag 6 is performs best.

Hence, I only fit 1 model for polynomial distributed lags approach, with no better result than DLM approach.
Therefore, model “m3” is the best fit model this far.

1.c. Koyck Transformation

Because so far the DLM approach perform better, we will make this approach better by using Koyck transformation to deal with the nature of DLM (that nonlinear in terms of its parameters).

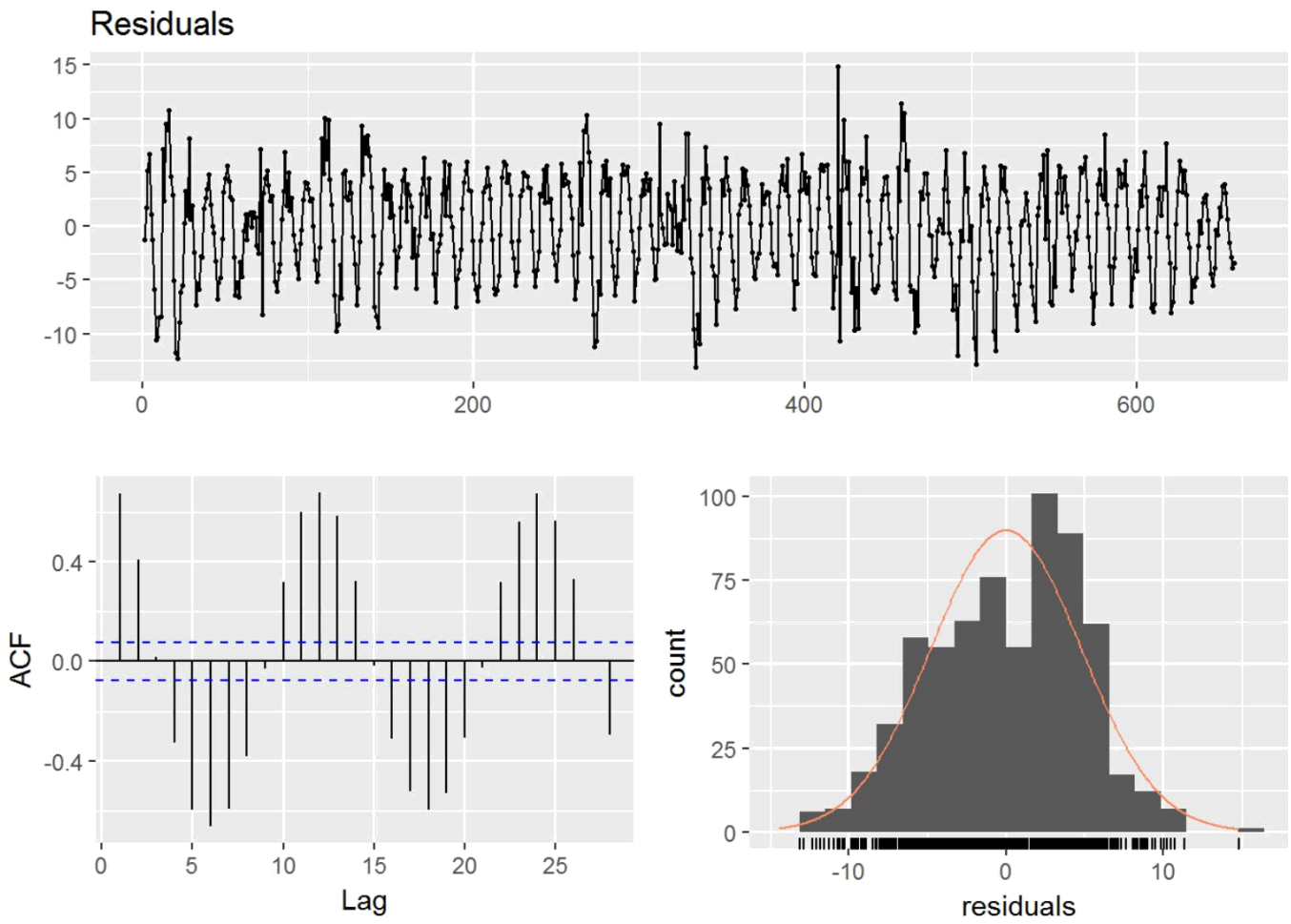
```
m5 = koyckDlm(x = ppt , y = solar , show.summary = TRUE)
```

```
 ##
 ## Call:
 ## ivreg(formula = y.t ~ Y.t_1 + X.t | Y.t_1 + X.t_1)
##
 ## Residuals:
 ##      Min        1Q  Median        3Q      Max
 ## -13.0926  -3.5961  0.3176    3.6103  14.8399
 ##
 ## Coefficients:
 ##              Estimate Std. Error t value Pr(>|t|)
 ## (Intercept) -2.23925     0.76549  -2.925  0.00356 **
 ## Y.t_1        0.98546     0.02424  40.650  < 2e-16 ***
 ## X.t          5.34684     0.84383   6.336 4.37e-10 ***
 ## ---
 ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
 ## Residual standard error: 4.814 on 656 degrees of freedom
 ## Multiple R-Squared: 0.7598,  Adjusted R-squared: 0.7591
 ## Wald test:  1104 on 2 and 656 DF,  p-value: < 2.2e-16
 ##
 ## alpha beta phi ## Geometric coefficients: -154.0203
 5.346844 0.9854613
```

Here, all of the coefficients are significant, and we have significant p-value also. The adjusted R-squared also increase significantly. The original parameter values are beta=5.35 and phi=0.98. So the weights will decline quickly at the rate of 0.98.

Now we will check the residuals, VIF, and MASE value

```
checkresiduals(m5$model)
```




```
bgtest(m5$model)
```

```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 1
##
 ## data:  m5$model
 ## LM test = 387.66, df = 1, p-value < 2.2e-16
```

```
vif(m5$model)
```

```
##    Y.t_1      X.t
## 1.605001
```

```
1.605001 MASE(m5)
```

```
##        MASE
## m5 1.032483
```

Figure5: Residuals plot of Model 5

The BG-test still indicates significant residuals in this model. However, the trend and variance now are not clearly visible. The histogram also not rightly skewed as before, however there are still some correlations in ACF plot.

VIF value is very low, indicates the multicollinearity effect is low. Also, the MASE value decrease

significantly. Hence, compared to the best model before (“m3” model from DLM approach), “m5” model with Koyck approach perform better as becomes the best fit so far.

1.d. Autoregressive Distributed Lag Model

As explained before, Koyck approach gives the best fit model so far, but it still suffer from significant residuals. Hence, another approach called Autoregressive DLM is being done. Here, we will experiment with ARDL model with p=q=1,2,3.

```
 m6 = ardlDlm(x = ppt, y = solar, p = 1, q = 1)
 m7 = ardlDlm(x = ppt, y = solar, p = 1, q = 2)
 m8 = ardlDlm(x = ppt, y = solar, p = 1, q = 3)
 m9 = ardlDlm(x = ppt, y = solar, p = 2, q = 1)
m10 = ardlDlm(x = ppt, y = solar, p = 2, q = 2)
m11 = ardlDlm(x = ppt, y = solar, p = 2, q = 3)
m12 = ardlDlm(x = ppt, y = solar, p = 3, q = 1)
m13 = ardlDlm(x = ppt, y = solar, p = 3, q = 2)
m14 = ardlDlm(x = ppt, y = solar, p = 3, q = 3)
```

All of the Autoregressive model gives a high adjusted R-squared value. However, “m11” (with p=2 and q=3) and “m14” (with p=3 and q=3) gives the highest adjusted R-squared value. Now we will check the MASE value of these models

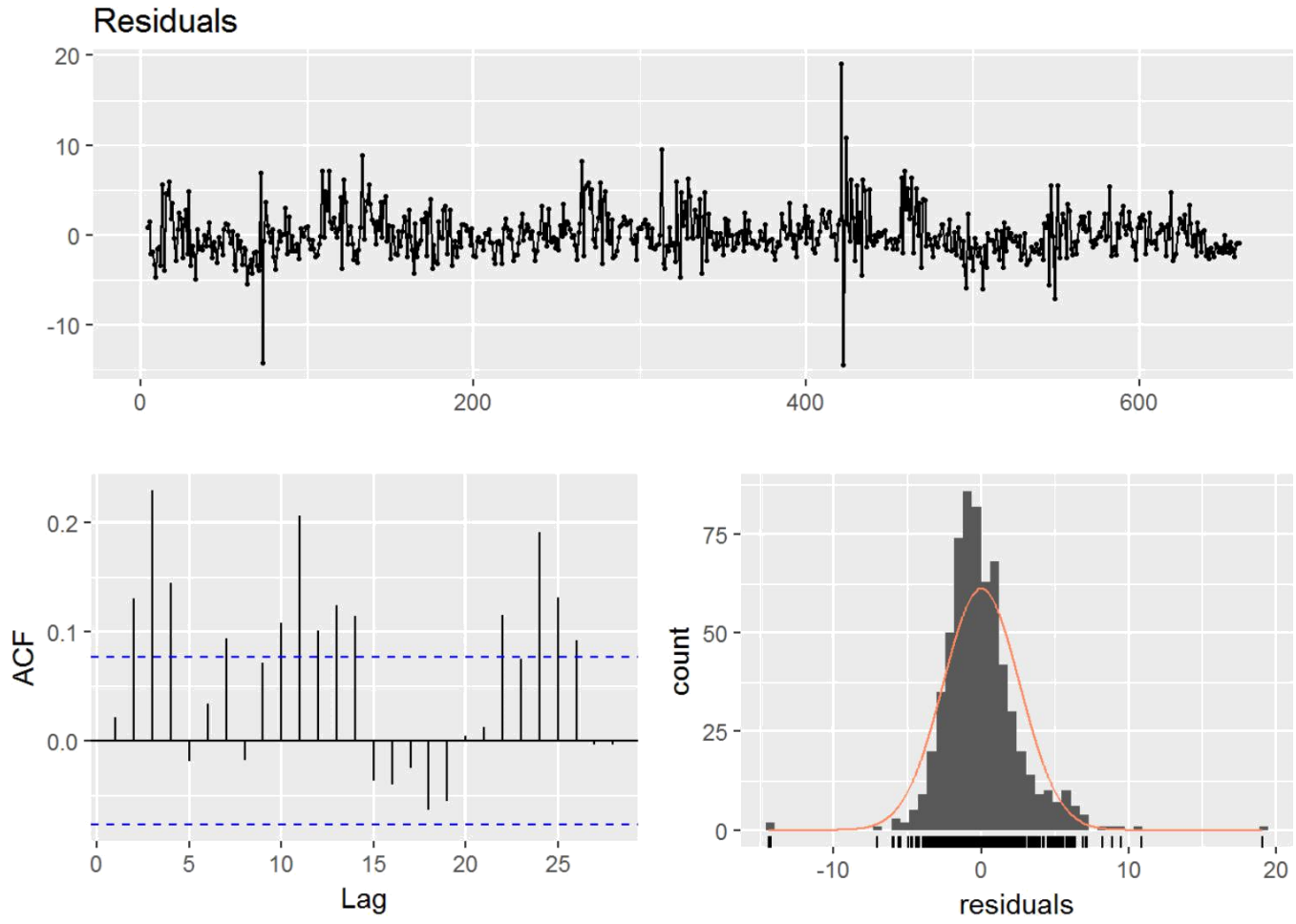
```
mase <- MASE(m6,m7,m8,m9,m10,m11,m12,m13,m14)
mase
```

```
##       n      MASE
## m6  659 0.8392434
## m7  658 0.4971918
## m8  657 0.4740063
## m9  658 0.7834855
## m10 658 0.4951319
## m11 657 0.4738939
## m12 657 0.7572489
## m13 657 0.4955334
## m14 657 0.4737144
```

Here,“m14” gives the lowest MASE value (0.4737), lower than previous best model (“m5” with Koyck DLM approach) with MASE value of 1.03.

we will check the residuals and VIF value for this model.

```
checkresiduals(m14$model)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 11
##
 ## data:  object
 ## LM test = 131.77, df = 11, p-value < 2.2e-16
```

```
vif(m14$model)
```

```
##       X.t L(X.t, 1) L(X.t, 2) L(X.t, 3) L(y.t, 1) L(y.t, 2) L(y.t, 3)
##  3.777687  7.531419  7.514934  3.885577 12.548670 34.621642 12.222617
```

Figure6: Residuals plot of Model 14

Here, half of VIF value are under 10, and half of them are above 10. Hence, the multi-collinearity effect on this model is moderate. However, this model still have significant residuals, proven by BG-test and the residuals graph: significant correlation on ACF plot and changing variance in time series plot.

Hence, we can conclude that all of the models still capture significant residuals. However, because model “m14” with Autoregressive DLM approach gives the lowest MASE amongst all of the models, the forecast for the next 2 years will be based on this model. Here, the forecast data is inserted manually from “data.x.csv” dataset, due to technical issue.

```
fc = ardlDlmForecast(model = m14, x = c(0.189009998,0.697262522,0.595213491,0.487388526,0.261677017,0.808606651,0.94186202,0.905636325,1.059964682,0.341438784,0.525805322,0.602471062,0.109860632,0.781464707,0.69685501,0.502413906,0.649385609,0.745960773,0.663047123,0.533770112,0.61542621,0.54606508,0.142673325,0.013650407), h=24)$forecast
fc
```

```
##  [1]  6.857344  8.845041 12.016105  7.852801 11.098041 13.833209  7.236146
##  [8] 14.523800 16.293669  6.765202 18.585099 16.734391  4.580927 23.319762
## [15] 15.817759  1.895151 30.964447 12.859888 -1.902011 42.745119  5.709499
## [22] -6.126360 60.030120 -9.114401
```

```
{plot(s, type="o", xaxt="n",xlim=c(1970,2035),ylim=c(0,100), ylab = "Solar Radiation", xlab = "Time", main="Solar Radiation Forecast")
lines(ts(fc, start = 2015),col="Red",type="o")}
```

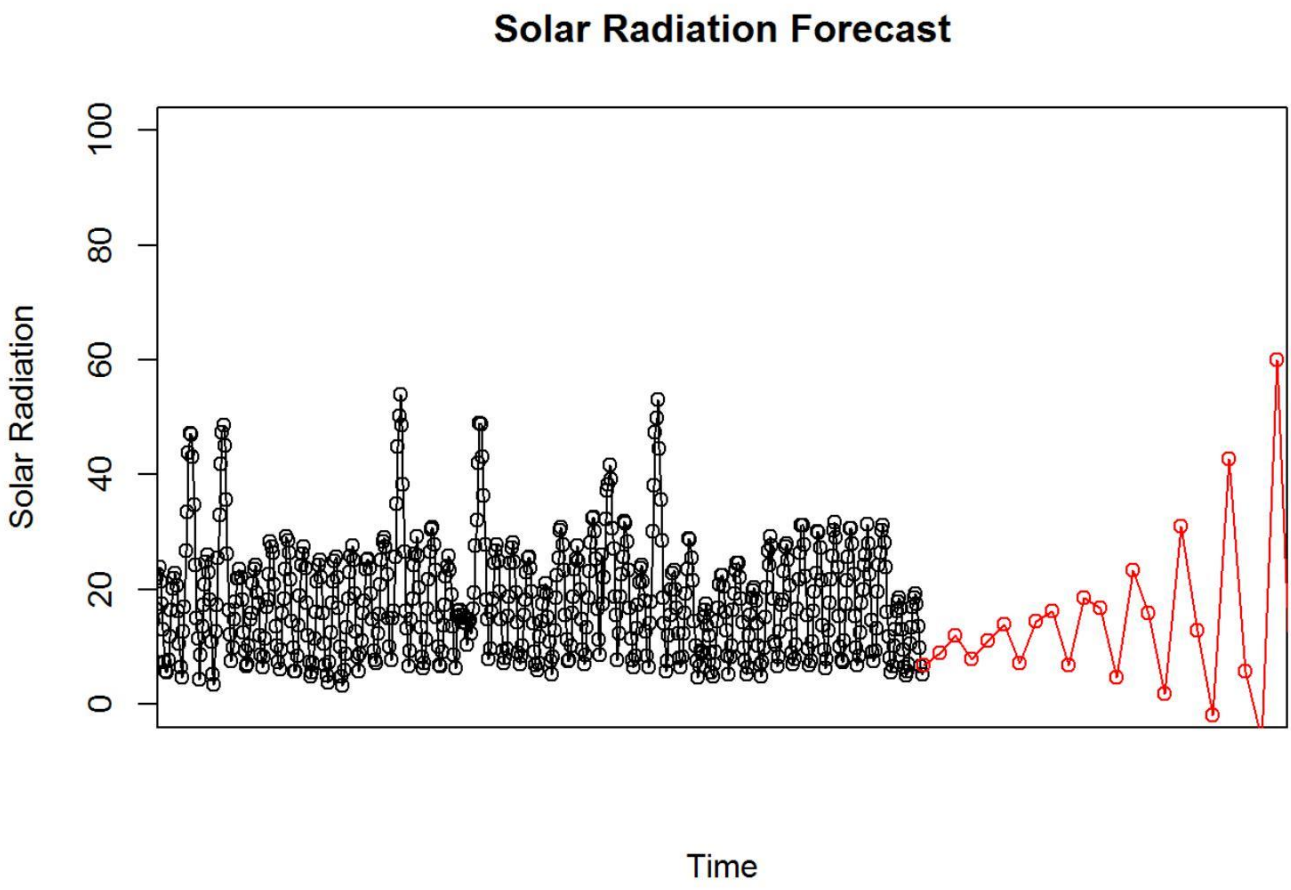


Figure7: Solar Radiation Forecast 1970-1935

2. Dynamic Linear Models

In previous DLM models, all of the models have significant residuals, means they cannot capture all the trends, correlations, and seasonal pattern in the data. In this part, Dynamic Linear Models is used to capture all of those components. There are two events that can be skeptically looked as an intervention. However, the changing in mean level most obvious in year 1965, that happened for a whole year. Because the intervention is not immediate, also the shift in the mean level is not permanent, I propose to use pulse

function for this intervention. Also, even though the trend is not clearly visible in solar radiation time series plot, we will include all of the component with lag 1 at the beginning.

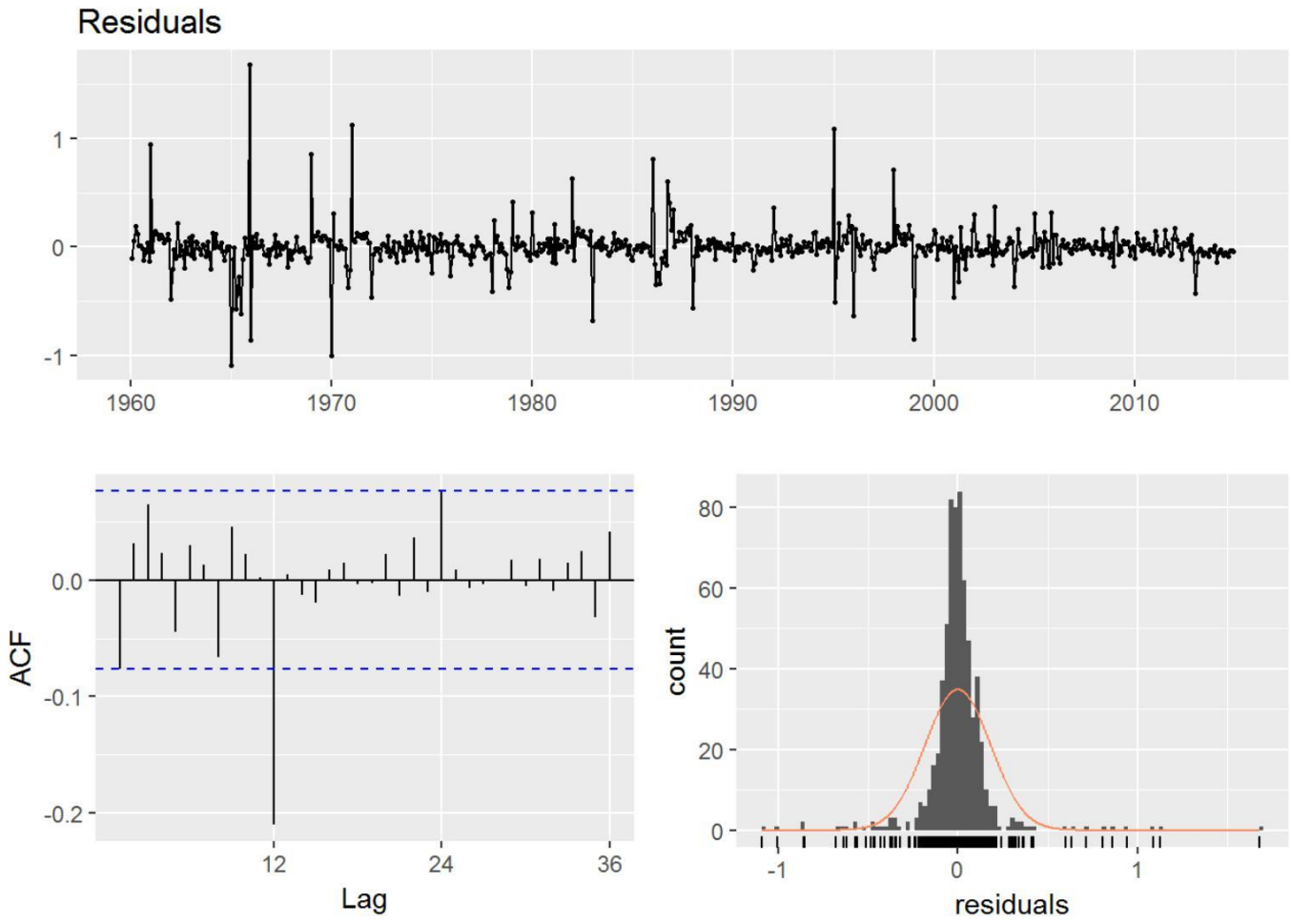
```
#Here, Y.t is treated as log of solar radiation series.
Y.ta = log(s)
X.t=ppt
T=59
P.t=1*(seq(s)==T)
P.t.1=Lag(P.t,+1)
```

```
ma<- dynlm(Y.ta ~ X.t+ L(Y.ta , k = 1 ) + P.t.1 + P.t +trend(Y.ta) + season(Y.ta))
summary(ma)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(2), End = 2014(12)
##
## Call:
## dynlm(formula = Y.ta ~ X.t + L(Y.ta, k = 1) + P.t.1 + P.t + trend(Y.ta) +
##     season(Y.ta))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.09014 -0.05142 -0.00033  0.05518  1.68124
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.3754582  0.0518986   7.234 1.34e-12 ***
## X.t              -0.0444865  0.0268560  -1.656  0.09811 .
## L(Y.ta, k = 1)    0.8527820  0.0203855  41.833  < 2e-16 ***
## P.t.1            -0.1926463  0.1876041  -1.027  0.30487
## P.t              -0.2074913  0.1874880  -1.107  0.26884
## trend(Y.ta)       0.0002006  0.0004599   0.436  0.66281
## season(Y.ta)Feb   0.2519886  0.0356046   7.077 3.86e-12 ***
## season(Y.ta)Mar   0.4054211  0.0363889  11.141  < 2e-16 ***
## season(Y.ta)Apr   0.2563600  0.0390487   6.565 1.07e-10 ***
## season(Y.ta)May   0.2972955  0.0411956   7.217 1.51e-12 ***
## season(Y.ta)Jun   0.2042402  0.0445541   4.584 5.48e-06 ***
## season(Y.ta)Jul   0.1336343  0.0466718   2.863  0.00433 **
## season(Y.ta)Aug   0.0041204  0.0471696   0.087  0.93042
## season(Y.ta)Sep -0.1282170  0.0454115  -2.823  0.00490 **
## season(Y.ta)Oct -0.2695124  0.0415218  -6.491 1.71e-10 ***
## season(Y.ta)Nov -0.3777438  0.0383271  -9.856  < 2e-16 ***
## season(Y.ta)Dec -0.2924553  0.0362548  -8.067 3.55e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1854 on 642 degrees of freedom
## Multiple R-squared:  0.9059, Adjusted R-squared:  0.9036
## F-statistic: 386.4 on 16 and 642 DF,  p-value: < 2.2e-16
```

Here we get high adjusted R-squared value (0.9036), however this need to be treated skeptically since this high value means the model can be overfitting. We will see the residuals to see if the model can capture all seasonal and trend components.

```
checkresiduals(ma)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 55.957, df = 24, p-value = 0.0002323
```

Figure 8: Residuals plot of Model "ma"

The histogram indicates the series is normally distributed, however, there are visible changing variance in time series plot, also 1 significant lag in ACF plot. Also, BG-test stated this residuals is significant. Hence, we need to take a look at another model to get a better residual result.

Interestingly, in this model, even though the trend is expected to be not significant (because there is no visible trend in the plot), all of pulse step also not significant. Next, we will take out

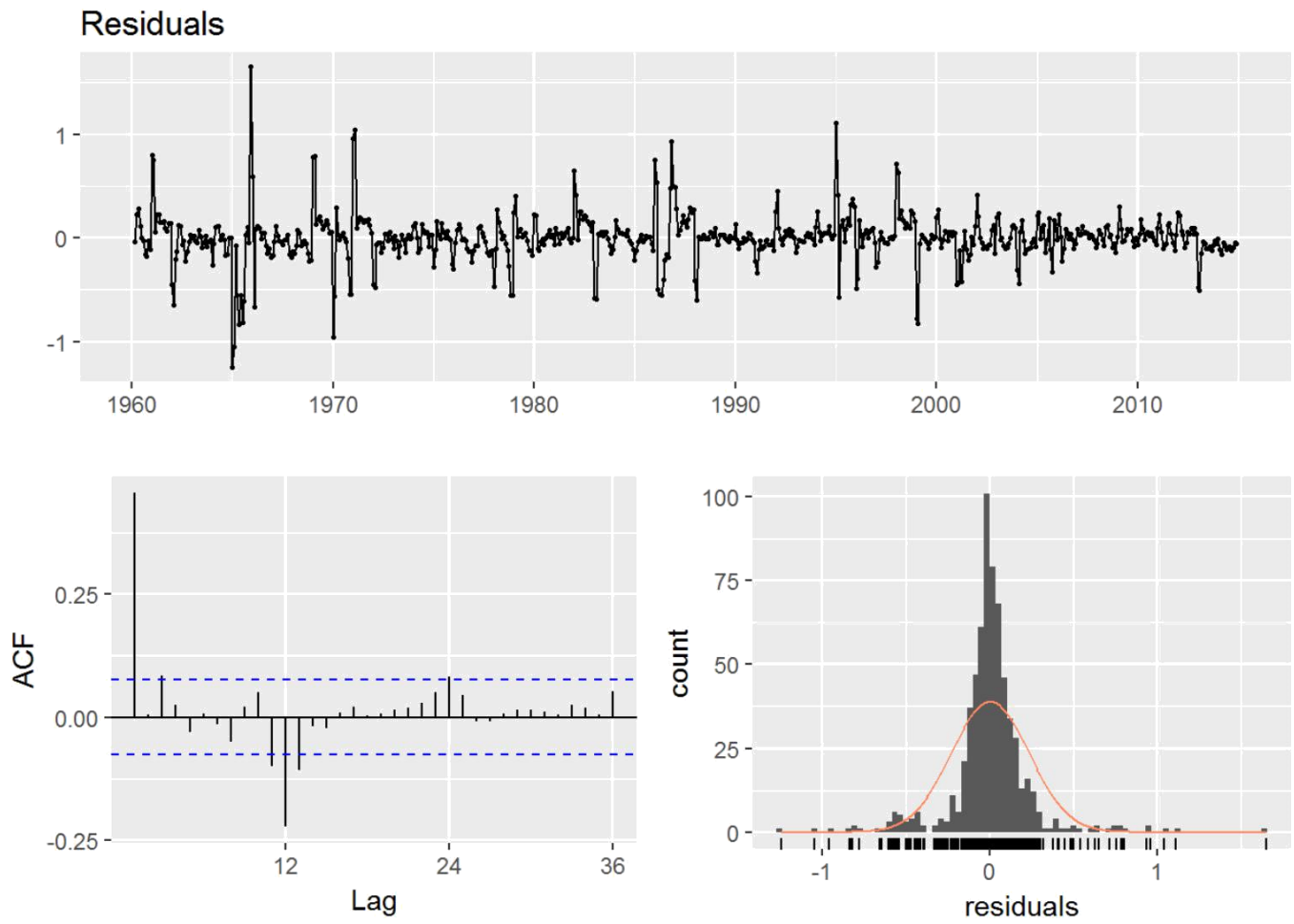
the trend but not the pulse, to see if the model behave better. Here, the new model is built with second lag.

```
mb<- dynlm(Y.ta ~ X.t+ L(Y.ta , k = 2) + P.t.1 + P.t + season(Y.ta))
summary(mb)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(3), End = 2014(12)
##
## Call:
## dynlm(formula = Y.ta ~ X.t + L(Y.ta, k = 2) + P.t.1 + P.t + season(Y.ta))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.24332 -0.07118 -0.00010  0.08004  1.65109
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.39020    0.07043   5.541 4.40e-08 ***
## X.t              -0.07078    0.03406  -2.078   0.0381 *
## L(Y.ta, k = 2)    0.75074    0.02572  29.187  < 2e-16 ***
## P.t.1            -0.37970    0.23774  -1.597   0.1107
## P.t              -0.24329    0.23770  -1.024   0.3064
## season(Y.ta)Feb   0.51380    0.04598  11.174  < 2e-16 ***
## season(Y.ta)Mar   0.88107    0.04562  19.313  < 2e-16 ***
## season(Y.ta)Apr   0.85623    0.04535  18.881  < 2e-16 ***
## season(Y.ta)May   0.75792    0.04753  15.944  < 2e-16 ***
## season(Y.ta)Jun   0.69313    0.05083  13.635  < 2e-16 ***
## season(Y.ta)Jul   0.54276    0.05448   9.962  < 2e-16 ***
## season(Y.ta)Aug   0.35484    0.05601   6.335 4.45e-10 ***
## season(Y.ta)Sep   0.11430    0.05553   2.058   0.0400 *
## season(Y.ta)Oct -0.13266    0.05212  -2.545   0.0111 *
## season(Y.ta)Nov -0.36142    0.04934  -7.325 7.20e-13 ***
## season(Y.ta)Dec -0.36394    0.04671  -7.792 2.67e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2355 on 642 degrees of freedom
## Multiple R-squared:  0.8478, Adjusted R-squared:  0.8443
## F-statistic: 238.4 on 15 and 642 DF,  p-value: < 2.2e-16
```

With second lag of Y.t, the adjusted R-square (0.8443) becomes lower. Now we will see the residuals.

```
checkresiduals(mb)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 272.35, df = 24, p-value < 2.2e-16
```

Figure9: Residuals plot of Model "mb"

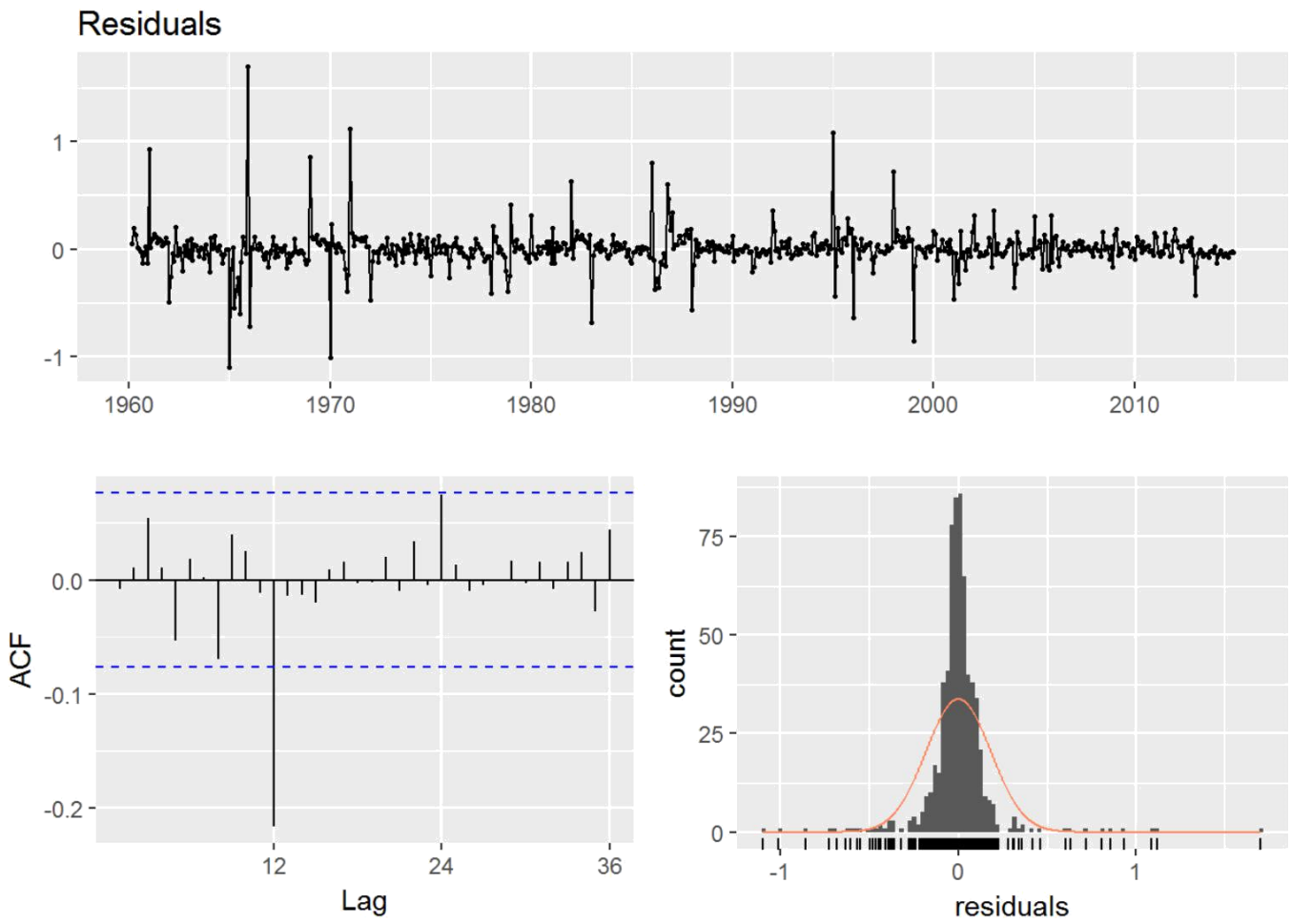
After putting the second lag, the residuals is getting worse. The changing variance in time series plot and autocorrelation in ACF plot become more and more. Also, BG-test stated this residuals is still significant. Hence, another model is proposed by combining the first and second lag together.

```
mc<- dynlm(Y.ta ~ X.t+ L(Y.ta, k = 1) + P.t.1 + P.t + L(Y.ta , k = 2)+season(Y.ta)
)
summary(mc)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(3), End = 2014(12)
##
## Call:
## dynlm(formula = Y.ta ~ X.t + L(Y.ta, k = 1) + P.t.1 + P.t + L(Y.ta,
##     k = 2) + season(Y.ta))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.09916 -0.04873  0.00022  0.05392  1.70114
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.33676    0.05539   6.080 2.06e-09 ***
## X.t              -0.04426    0.02678  -1.653  0.09892 .
## L(Y.ta, k = 1)    0.78544    0.03930  19.986  < 2e-16 ***
## P.t.1            -0.20755    0.18695  -1.110  0.26732
## P.t              -0.21124    0.18673  -1.131  0.25835
## L(Y.ta, k = 2)    0.07889    0.03922   2.012  0.04469 *
## season(Y.ta)Feb   0.28013    0.03796   7.379 4.96e-13 ***
## season(Y.ta)Mar   0.44743    0.04189  10.680  < 2e-16 ***
## season(Y.ta)Apr   0.30211    0.04514   6.693 4.79e-11 ***
## season(Y.ta)May   0.32406    0.04319   7.503 2.09e-13 ***
## season(Y.ta)Jun   0.22963    0.04618   4.973 8.48e-07 ***
## season(Y.ta)Jul   0.14864    0.04712   3.154  0.00168 **
## season(Y.ta)Aug   0.01245    0.04721   0.264  0.79205
## season(Y.ta)Sep -0.12913    0.04529  -2.851  0.00450 **
## season(Y.ta)Oct -0.27736    0.04158  -6.671 5.49e-11 ***
## season(Y.ta)Nov -0.39109    0.03879 -10.083  < 2e-16 ***
## season(Y.ta)Dec -0.30628    0.03680  -8.322 5.21e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.185 on 641 degrees of freedom
## Multiple R-squared:  0.9062, Adjusted R-squared:  0.9039
## F-statistic: 387.2 on 16 and 641 DF,  p-value: < 2.2e-16
```

Interestingly, by combining first and second lag, now X.t becomes significant. Adjusted R-squared value also becomes higher than previous model. Now we will check the residuals.

```
checkresiduals(mc)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 60.148, df = 24, p-value = 6.087e-05
```

Figure10: Residuals plot of Model "mc"

However, the residuals still significant in this model, also changing variance and one significant lag at lag 12 is appear. It looks like the pulse coefficients are still not significant so far, and the model is not have much progress, since the residuals is still significant. Hence, I propose to take out the pulse coefficients and not take a log in solar radiation.

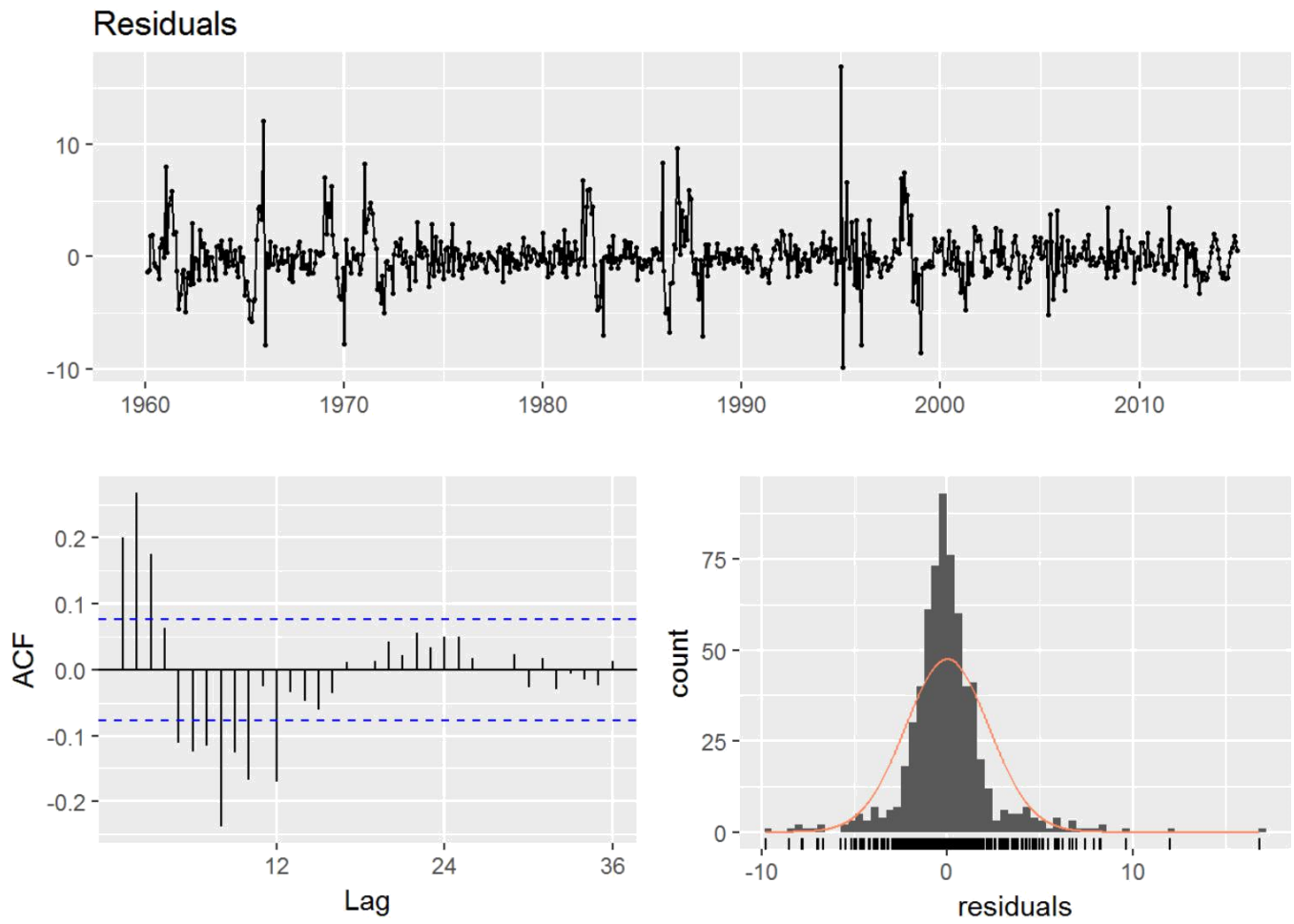
```
Y.t=s
X.t=ppt
```

```
m16 = dynlm(Y.t ~ X.t+ L(Y.t , k = 1 ) + season(Y.t))
summary(m16)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(2), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(Y.t, k = 1) + season(Y.t))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7834 -1.0052 -0.0964  0.8024 16.8758
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.36184    0.40650   3.350 0.000855 ***
## X.t             -0.23642    0.33142  -0.713 0.475877
## L(Y.t, k = 1)    0.93192    0.01423  65.471  < 2e-16 ***
## season(Y.t)Feb   1.90792    0.44040   4.332 1.71e-05 ***
## season(Y.t)Mar   4.86290    0.44315  10.973  < 2e-16 ***
## season(Y.t)Apr   3.76530    0.45742   8.232 1.02e-15 ***
## season(Y.t)May   5.10305    0.47866  10.661  < 2e-16 ***
## season(Y.t)Jun   3.27202    0.52284   6.258 7.10e-10 ***
## season(Y.t)Jul   1.50367    0.55526   2.708 0.006947 **
## season(Y.t)Aug -2.291170.56467  -4.058 5.57e-05 ***
## season(Y.t)Sep -4.814950.53621  -8.980  < 2e-16 ***
## season(Y.t)Oct -5.861480.48261 -12.145  < 2e-16 ***
## season(Y.t)Nov -5.105090.45124 -11.313  < 2e-16 ***
## season(Y.t)Dec -2.808580.44199  -6.354 3.96e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.294 on 645 degrees of freedom
## Multiple R-squared:  0.9464, Adjusted R-squared:  0.9453
## F-statistic: 875.3 on 13 and 645 DF,  p-value: < 2.2e-16
```

Here, X.t becomes insignificant again, however another coefficients are significant. The adjusted R-squared value is also still high.

```
checkresiduals(m16)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 134.42, df = 24, p-value < 2.2e-16
```

Figure 11: Residuals plot of Model 16

The residuals of this model is similar with previous models: the residuals are still significant, with changing variance in time series plot, and autocorrelation getting worse in ACF plot. Hence, we will fit another model with lag 2.

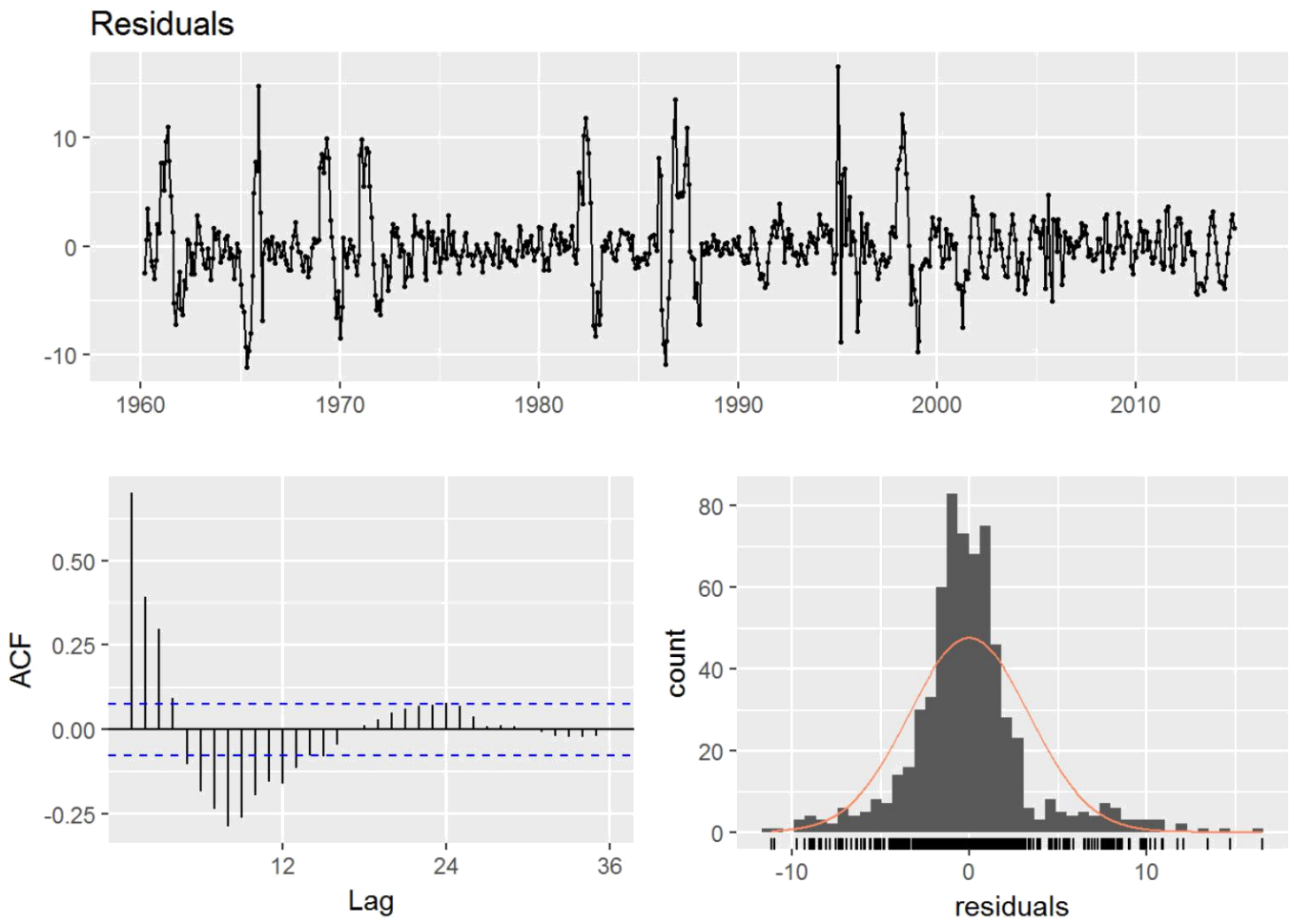
```
m17 = dynlm(Y.t ~ X.t+ L(Y.t , k = 2 ) + season(Y.t))
summary(m17)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(3), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(Y.t, k = 2) + season(Y.t))
##
## Residuals:
##      Min       1Q    Median       3Q       Max
## -11.1450  -1.5157   -0.2312  1.2198   16.5579
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.1491     0.6241   0.239  0.81129
## X.t              -0.3113     0.4963  -0.627  0.53074
## L(Y.t, k = 2)     0.8407     0.0213  39.468  < 2e-16 ***
## season(Y.t)Feb    4.5109     0.6639   6.794 2.48e-11 ***
## season(Y.t)Mar    9.2528     0.6608  14.002  < 2e-16 ***
## season(Y.t)Apr   10.9856     0.6605  16.633  < 2e-16 ***
## season(Y.t)May   11.4445     0.6793  16.848  < 2e-16 ***
## season(Y.t)Jun   10.9755     0.7235  15.170  < 2e-16 ***
## season(Y.t)Jul    7.6781     0.7808   9.834  < 2e-16 ***
## season(Y.t)Aug    2.3423     0.8110   2.888  0.00401 **
## season(Y.t)Sep   -3.6809     0.8085  -4.552 6.34e-06 ***
## season(Y.t)Oct   -7.1554     0.7482  -9.563  < 2e-16 ***
## season(Y.t)Nov   -7.5696     0.6970 -10.860  < 2e-16 ***
## season(Y.t)Dec   -4.7652     0.6677  -7.137 2.59e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.433 on 644 degrees of freedom
## Multiple R-squared:  0.8799, Adjusted R-squared:  0.8774
## F-statistic: 362.8 on 13 and 644 DF,  p-value: < 2.2e-16
```

Figure 12: Residuals plot of Model 17

Now the p-value is reduced. The residuals check is below:

```
checkresiduals(m17)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 421.5, df = 24, p-value < 2.2e-16
```

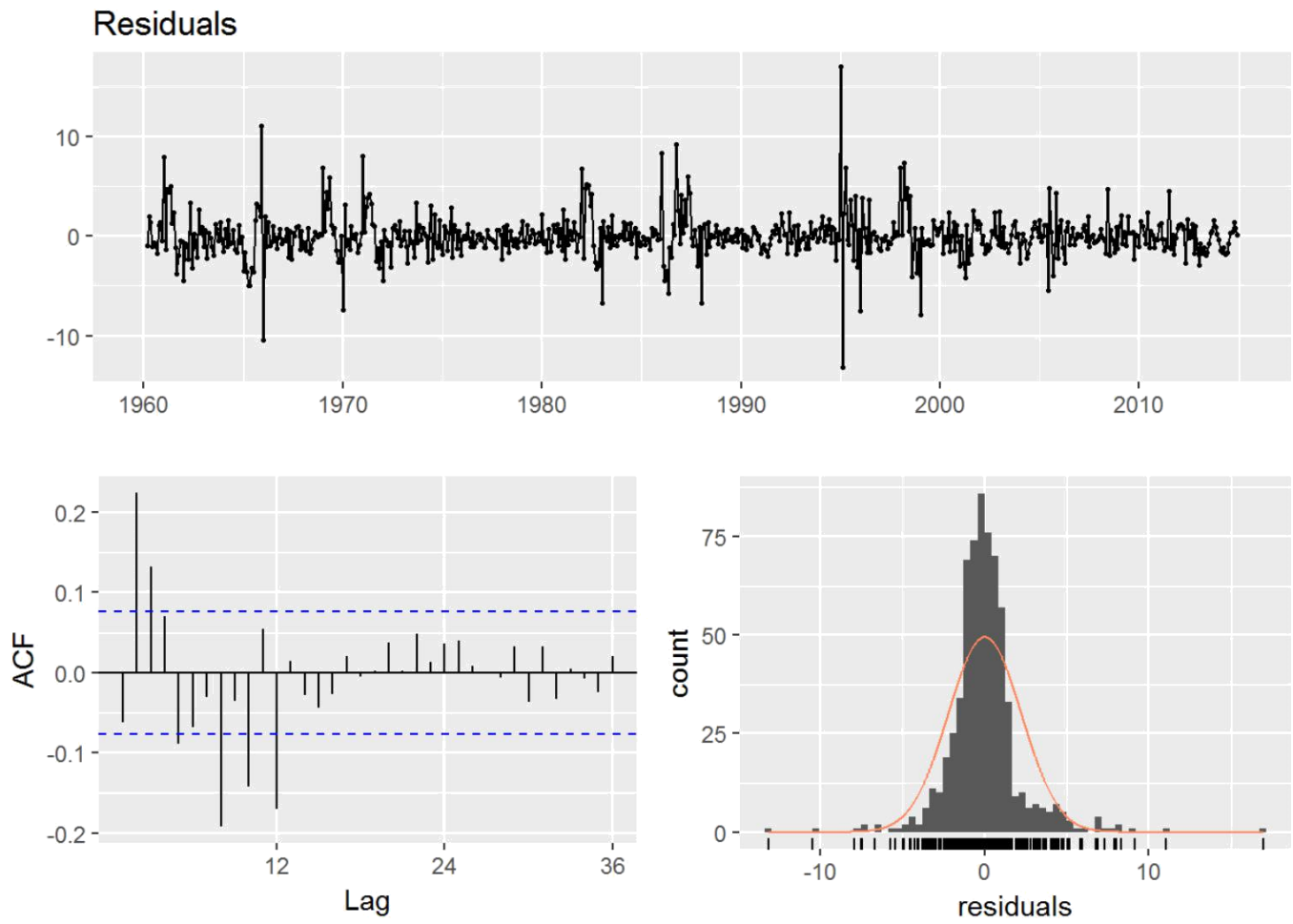
The similar inference of this model is still happened: the residuals still have changing variance in time series plot, and autocorrelation in ACF plot. Hence, we will fit another model combining lag 1 and lag 2.

```
m18 = dynlm(Y.t ~ X.t+ L(Y.t , k = 1 ) + L(Y.t , k = 2 ) + season(Y.t))
summary(m18)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(3), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(Y.t, k = 1) + L(Y.t, k = 2) + season(Y.t))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.1353  -0.9959  -0.0915   0.8252  17.0059
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.97079    0.41260   4.777 2.21e-06 ***
## X.t            -0.24527    0.32442  -0.756  0.44990
## L(Y.t, k = 1)   1.13172    0.03849  29.400  < 2e-16 ***
## L(Y.t, k = 2)  -0.21447    0.03850  -5.571 3.72e-08 ***
## season(Y.t)Feb  1.29901    0.44752   2.903  0.00383 **
## season(Y.t)Mar  3.86196    0.46925   8.230 1.04e-15 ***
## season(Y.t)Apr  2.24539    0.52418   4.284 2.12e-05 ***
## season(Y.t)May  3.95247    0.51194   7.721 4.45e-14 ***
## season(Y.t)Jun  1.95757    0.56368   3.473  0.00055 ***
## season(Y.t)Jul  0.68494    0.56305   1.216  0.22425
## season(Y.t)Aug -2.681540.55699  -4.814 1.84e-06 ***
## season(Y.t)Sep -4.420790.52910  -8.355 4.02e-16 ***
## season(Y.t)Oct -5.039140.49435 -10.194  < 2e-16 ***
## season(Y.t)Nov -4.206870.46974  -8.956  < 2e-16 ***
## season(Y.t)Dec -2.221250.44494  -4.992 7.70e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.244 on 643 degrees of freedom
## Multiple R-squared:  0.9488, Adjusted R-squared:  0.9476
## F-statistic: 850.2 on 14 and 643 DF,  p-value: < 2.2e-16
```

Now the adjusted R-values are higher again, but we need to check the residuals to see this model's performance.

```
checkresiduals(m18)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 109.78, df = 24, p-value = 6.162e-13
```

Figure 13: Residuals plot of Model 18

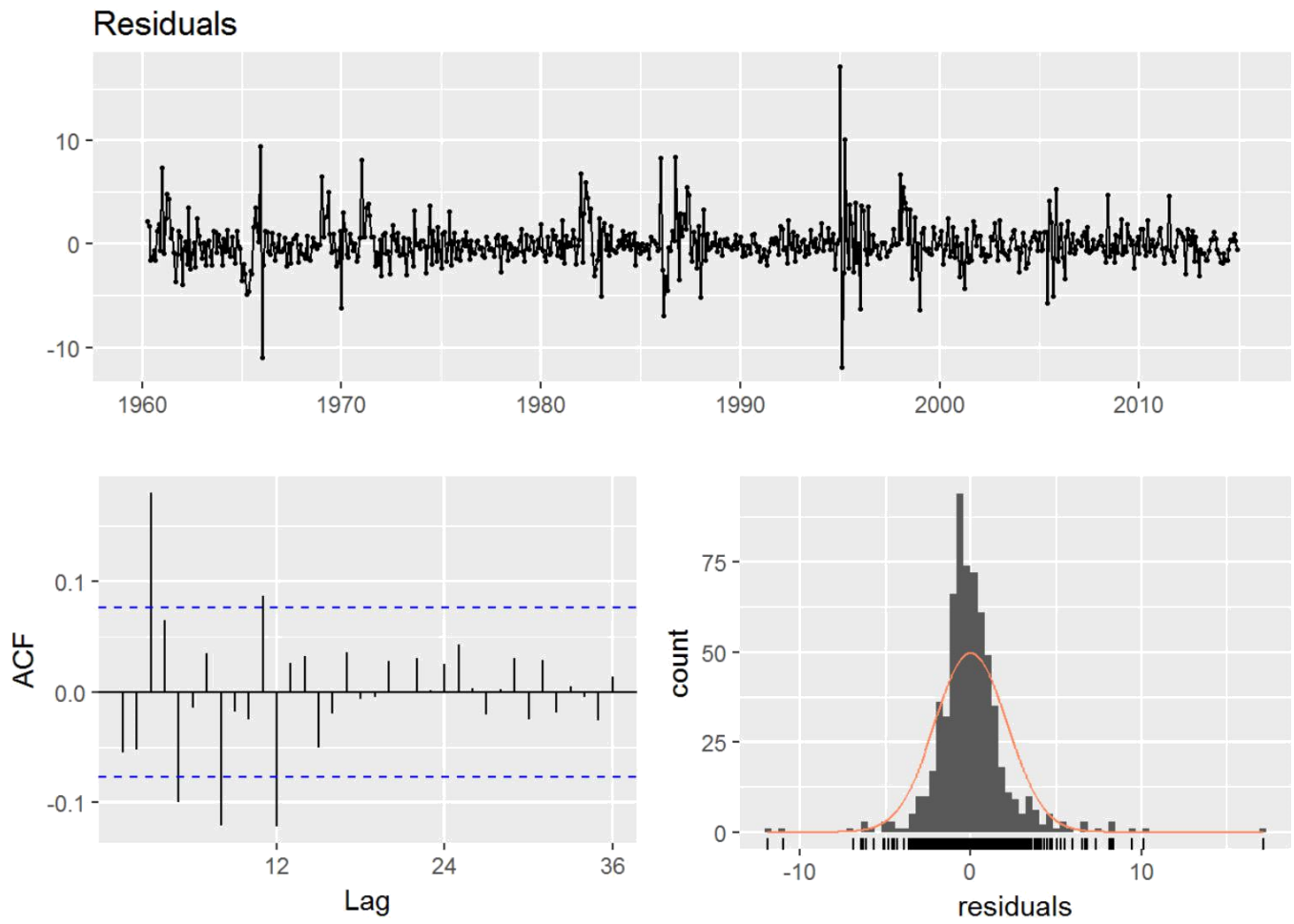
The inference is still similar with previous models. However, by adding both lags, the autocorrelation in ACF is reduced. Hence, we will add another lag to see if the model performs better.

```
m19=dynlm(Y.t ~ X.t + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+season(Y.
t))
summary(m19)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(4), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(Y.t, k = 1) + L(Y.t, k = 2) + L(Y.t,
##     k = 3) + season(Y.t))
##
## Residuals:
##      Min       1Q    Median       3Q      Max
## -11.8910  -0.9667  -0.1398   0.8091  17.1517
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.54381    0.44657   7.936 9.37e-15 ***
## X.t             -0.33681    0.31173  -1.080   0.2803
## L(Y.t, k = 1)    1.06952    0.03780  28.293  < 2e-16 ***
## L(Y.t, k = 2)    0.11225    0.05653   1.986   0.0475 *
## L(Y.t, k = 3)   -0.28856    0.03784  -7.626 8.78e-14 ***
## season(Y.t)Feb   0.66860    0.43705   1.530   0.1266
## season(Y.t)Mar   2.51605    0.48637   5.173 3.08e-07 ***
## season(Y.t)Apr   0.57076    0.54877   1.040   0.2987
## season(Y.t)May   1.50248    0.58710   2.559   0.0107 *
## season(Y.t)Jun   0.07092    0.59503   0.119   0.9052
## season(Y.t)Jul -1.54179     0.61454  -2.509   0.0124 *
## season(Y.t)Aug -4.349710.57777  -7.528 1.75e-13 ***
## season(Y.t)Sep -5.741080.53646 -10.702  < 2e-16 ***
## season(Y.t)Oct -5.436850.47694 -11.399  < 2e-16 ***
## season(Y.t)Nov -4.084680.45070  -9.063  < 2e-16 ***
## season(Y.t)Dec -1.942020.42823  -4.535 6.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.151 on 641 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.9519
## F-statistic: 866.3 on 15 and 641 DF,  p-value: < 2.2e-16
```

Here the adjusted R-value is keep increasing, however, the number of insignificant coefficients also increase. Now we will check the residuals:

```
checkresiduals(m19)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 58.344, df = 24, p-value = 0.000109
```

Figure 14: Residuals plot of Model 19

Here, even though the changing variance still visible, the number of correlation in ACF plot keeps reducing.

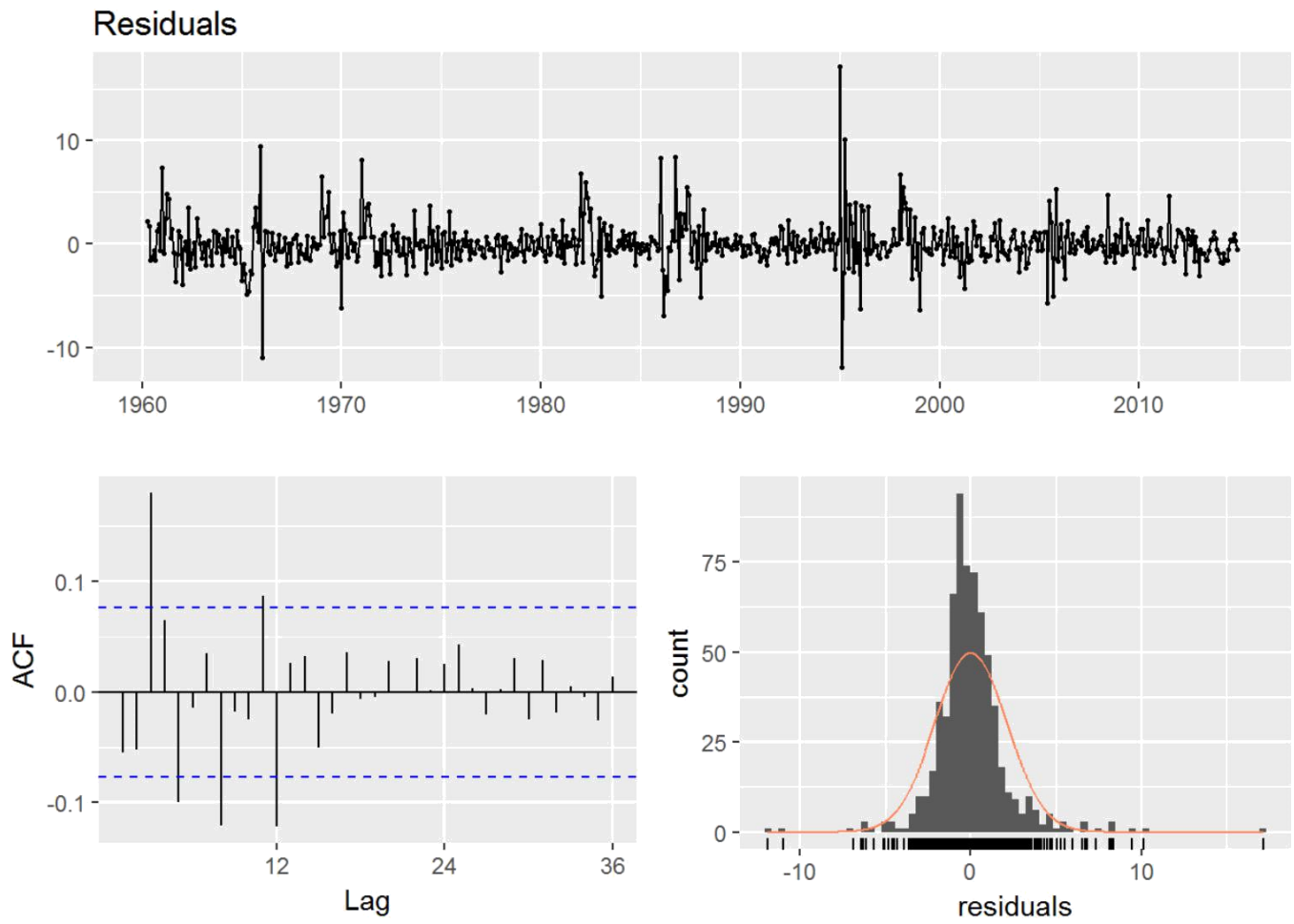
Now, we will add independent (X.t) variable, and see if we can make the model better.

```
m20=dynlm(Y.t ~ X.t + L(X.t , k = 1 ) + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+season(Y.t))
summary(m20)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(4), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(X.t, k = 1) + L(Y.t, k = 1) + L(Y.t,
##     k = 2) + L(Y.t, k = 3) + season(Y.t))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.8910  -0.9667  -0.1398   0.8091 17.1517
##
## Coefficients: (1 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.54381    0.44657   7.936 9.37e-15 ***
## X.t             -0.33681    0.31173  -1.080   0.2803
## L(X.t, k = 1)         NA         NA      NA       NA
## L(Y.t, k = 1)    1.06952    0.03780  28.293  < 2e-16 ***
## L(Y.t, k = 2)    0.11225    0.05653   1.986   0.0475 *
## L(Y.t, k = 3)   -0.28856    0.03784  -7.626 8.78e-14 ***
## season(Y.t)Feb   0.66860    0.43705   1.530   0.1266
## season(Y.t)Mar   2.51605    0.48637   5.173 3.08e-07 ***
## season(Y.t)Apr   0.57076    0.54877   1.040   0.2987
## season(Y.t)May   1.50248    0.58710   2.559   0.0107 *
## season(Y.t)Jun   0.07092    0.59503   0.119   0.9052
## season(Y.t)Jul -1.54179    0.61454  -2.509   0.0124 *
## season(Y.t)Aug -4.349710.57777  -7.528 1.75e-13 ***
## season(Y.t)Sep -5.741080.53646 -10.702  < 2e-16 ***
## season(Y.t)Oct -5.436850.47694 -11.399  < 2e-16 ***
## season(Y.t)Nov -4.084680.45070  -9.063  < 2e-16 ***
## season(Y.t)Dec -1.942020.42823  -4.535 6.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.151 on 641 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.9519
## F-statistic: 866.3 on 15 and 641 DF,  p-value: < 2.2e-16
```

Again, the number of significant correlations is keep decreasing. Now we will see the residuals:

```
checkresiduals(m20)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 58.344, df = 24, p-value = 0.000109
```

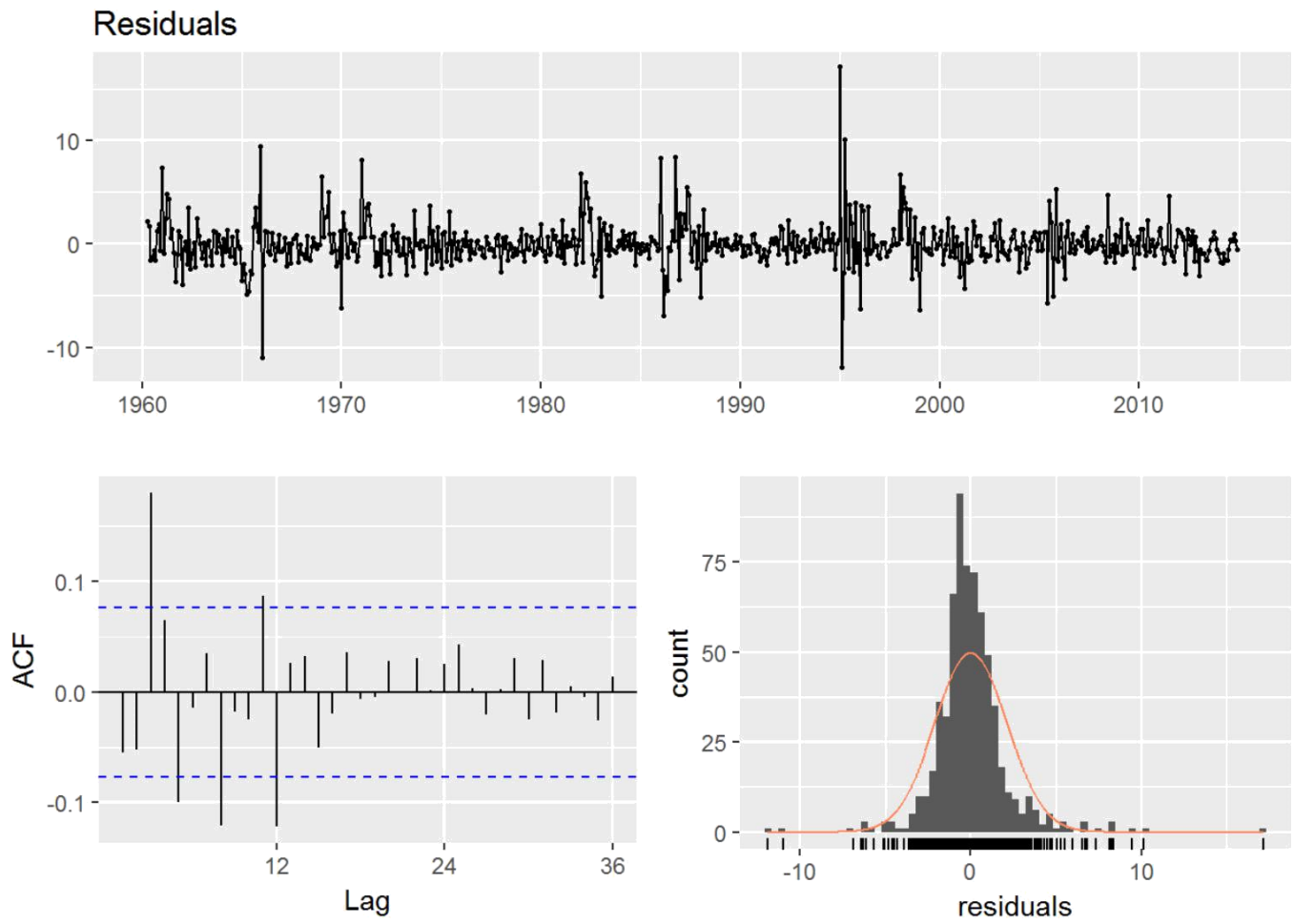
Figure 15: Residuals plot of Model 20

Here even though the residuals still significant, but the number of autocorrelation in ACF plot is keep decreasing. Now we will try to use lag 2 in X variable to find a better result.

```
m21=dynlm(Y.t ~ X.t + L(X.t , k = 2 ) + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+season(Y.t))
summary(m20)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(4), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(X.t, k = 1) + L(Y.t, k = 1) + L(Y.t,
##     k = 2) + L(Y.t, k = 3) + season(Y.t))
##
## Residuals:
##      Min       1Q    Median        3Q       Max
## -11.8910  -0.9667  -0.1398   0.8091 17.1517
##
## Coefficients: (1 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.54381    0.44657   7.936 9.37e-15 ***
## X.t             -0.33681    0.31173  -1.080   0.2803
## L(X.t, k = 1)         NA         NA      NA       NA
## L(Y.t, k = 1)    1.06952    0.03780  28.293  < 2e-16 ***
## L(Y.t, k = 2)    0.11225    0.05653   1.986   0.0475 *
## L(Y.t, k = 3)   -0.28856    0.03784  -7.626 8.78e-14 ***
## season(Y.t)Feb   0.66860    0.43705   1.530   0.1266
## season(Y.t)Mar   2.51605    0.48637   5.173 3.08e-07 ***
## season(Y.t)Apr   0.57076    0.54877   1.040   0.2987
## season(Y.t)May   1.50248    0.58710   2.559   0.0107 *
## season(Y.t)Jun   0.07092    0.59503   0.119   0.9052
## season(Y.t)Jul -1.54179    0.61454  -2.509   0.0124 *
## season(Y.t)Aug -4.349710.57777  -7.528 1.75e-13 ***
## season(Y.t)Sep -5.741080.53646 -10.702  < 2e-16 ***
## season(Y.t)Oct -5.436850.47694 -11.399  < 2e-16 ***
## season(Y.t)Nov -4.084680.45070  -9.063  < 2e-16 ***
## season(Y.t)Dec -1.942020.42823  -4.535 6.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.151 on 641 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.9519
## F-statistic: 866.3 on 15 and 641 DF,  p-value: < 2.2e-16
```

```
checkresiduals(m21)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 58.344, df = 24, p-value = 0.000109
```

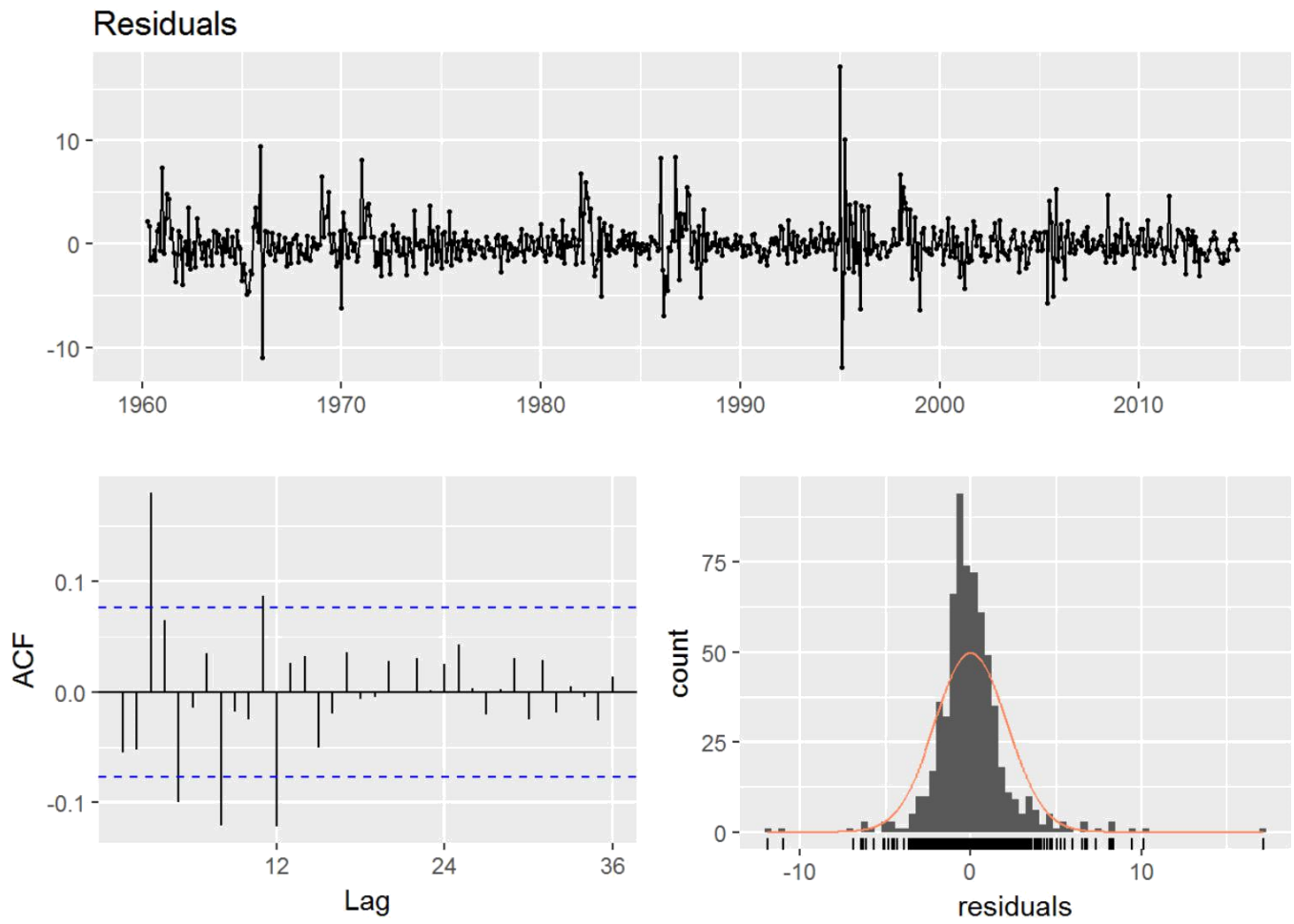
Figure 16: Residuals plot of Model 21

Here the number of autocorrelation in ACF plot keeps decreasing. Hence, we will add first and second lag in X variable.

```
m22=dynlm(Y.t ~ X.t + L(X.t , k = 1 )+ L(X.t , k = 2 ) + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+season(Y.t))
summary(m22)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(4), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(X.t, k = 1) + L(X.t, k = 2) + L(Y.t,
##     k = 1) + L(Y.t, k = 2) + L(Y.t, k = 3) + season(Y.t))
##
## Residuals:
##      Min       1Q    Median        3Q       Max
## -11.8910  -0.9667   -0.1398    0.8091  17.1517
##
## Coefficients: (2 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.54381    0.44657   7.936 9.37e-15 ***
## X.t             -0.33681    0.31173  -1.080   0.2803
## L(X.t, k = 1)         NA         NA      NA       NA
## L(X.t, k = 2)         NA         NA      NA       NA
## L(Y.t, k = 1)    1.06952    0.03780  28.293  < 2e-16 ***
## L(Y.t, k = 2)    0.11225    0.05653   1.986   0.0475 *
## L(Y.t, k = 3)   -0.28856    0.03784  -7.626 8.78e-14 ***
## season(Y.t)Feb   0.66860    0.43705   1.530   0.1266
## season(Y.t)Mar   2.51605    0.48637   5.173 3.08e-07 ***
## season(Y.t)Apr   0.57076    0.54877   1.040   0.2987
## season(Y.t)May   1.50248    0.58710   2.559   0.0107 *
## season(Y.t)Jun   0.07092    0.59503   0.119   0.9052
## season(Y.t)Jul -1.54179    0.61454  -2.509   0.0124 *
## season(Y.t)Aug -4.349710.57777  -7.528 1.75e-13 ***
## season(Y.t)Sep -5.741080.53646 -10.702  < 2e-16 ***
## season(Y.t)Oct -5.436850.47694 -11.399  < 2e-16 ***
## season(Y.t)Nov -4.084680.45070  -9.063  < 2e-16 ***
## season(Y.t)Dec -1.942020.42823  -4.535 6.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.151 on 641 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.9519
## F-statistic: 866.3 on 15 and 641 DF,  p-value: < 2.2e-16
```

```
checkresiduals(m22)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 58.344, df = 24, p-value = 0.000109
```

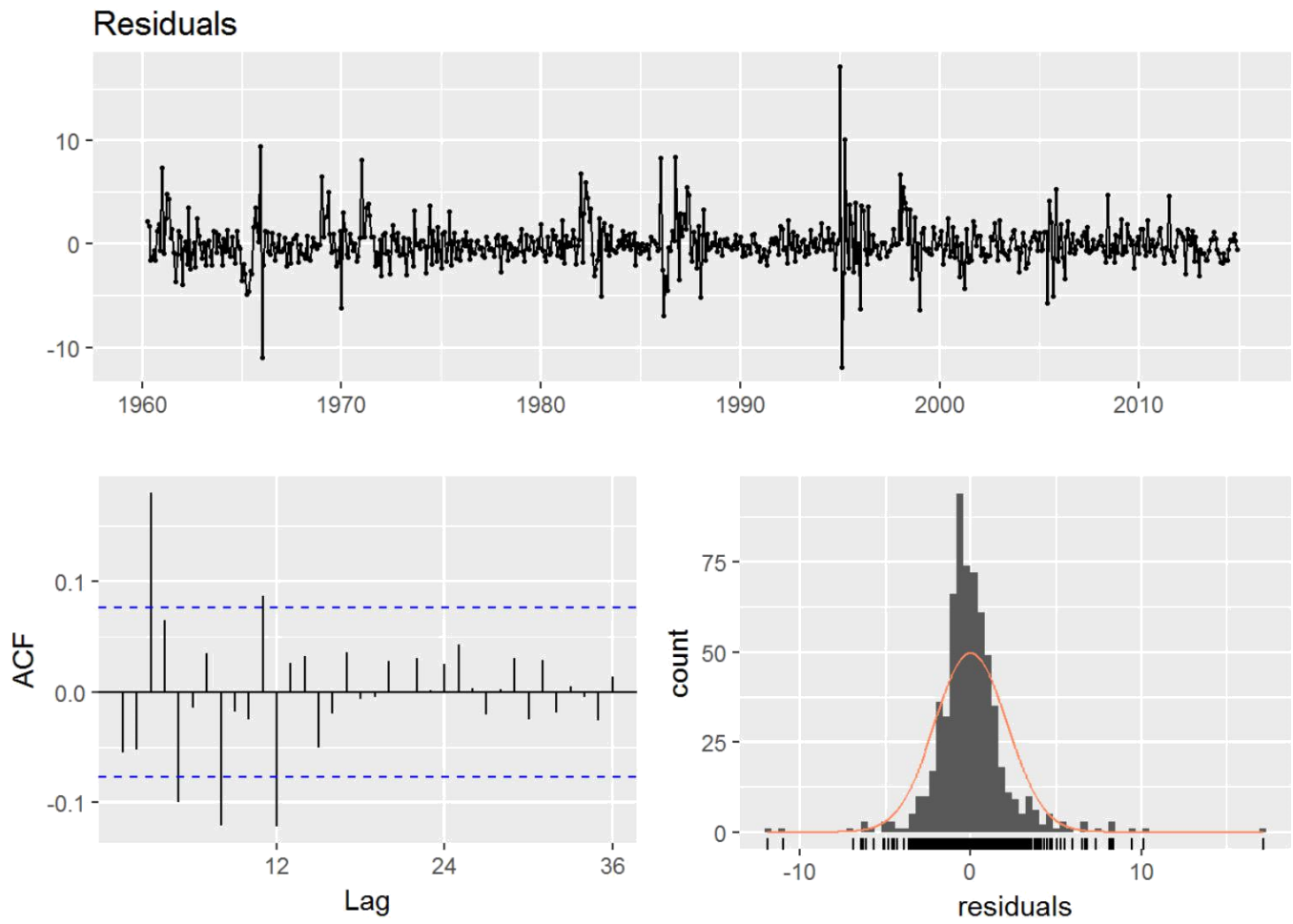
Figure17: Residuals plot of Model 22

The model perform better in term of residuals (and the p-value keep getting higher), hence I propose to add another lag in X variable.

```
m23=dynlm(Y.t ~ X.t + L(X.t , k = 1 )+ L(X.t , k = 2 )+ L(X.t , k = 3) + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+season(Y.t))
summary(m23)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(4), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(X.t, k = 1) + L(X.t, k = 2) + L(X.t,
##     k = 3) + L(Y.t, k = 1) + L(Y.t, k = 2) + L(Y.t, k = 3) +
##     season(Y.t))
##
## Residuals:
##      Min       1Q    Median        3Q       Max
## -11.8910  -0.9667  -0.1398   0.8091  17.1517
##
## Coefficients: (3 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.54381    0.44657   7.936 9.37e-15 ***
## X.t            -0.33681    0.31173  -1.080   0.2803
## L(X.t, k = 1)         NA         NA      NA       NA
## L(X.t, k = 2)         NA         NA      NA       NA
## L(X.t, k = 3)         NA         NA      NA       NA
## L(Y.t, k = 1)   1.06952    0.03780  28.293  < 2e-16 ***
## L(Y.t, k = 2)   0.11225    0.05653   1.986   0.0475 *
## L(Y.t, k = 3)  -0.28856    0.03784  -7.626 8.78e-14 ***
## season(Y.t)Feb  0.66860    0.43705   1.530   0.1266
## season(Y.t)Mar  2.51605    0.48637   5.173 3.08e-07 ***
## season(Y.t)Apr  0.57076    0.54877   1.040   0.2987
## season(Y.t)May  1.50248    0.58710   2.559   0.0107 *
## season(Y.t)Jun  0.07092    0.59503   0.119   0.9052
## season(Y.t)Jul -1.54179    0.61454  -2.509   0.0124 *
## season(Y.t)Aug -4.349710.57777  -7.528 1.75e-13 ***
## season(Y.t)Sep -5.741080.53646 -10.702  < 2e-16 ***
## season(Y.t)Oct -5.436850.47694 -11.399  < 2e-16 ***
## season(Y.t)Nov -4.084680.45070  -9.063  < 2e-16 ***
## season(Y.t)Dec -1.942020.42823  -4.535 6.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.151 on 641 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.9519
## F-statistic: 866.3 on 15 and 641 DF,  p-value: < 2.2e-16
```

```
checkresiduals(m23)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 58.344, df = 24, p-value = 0.000109
```

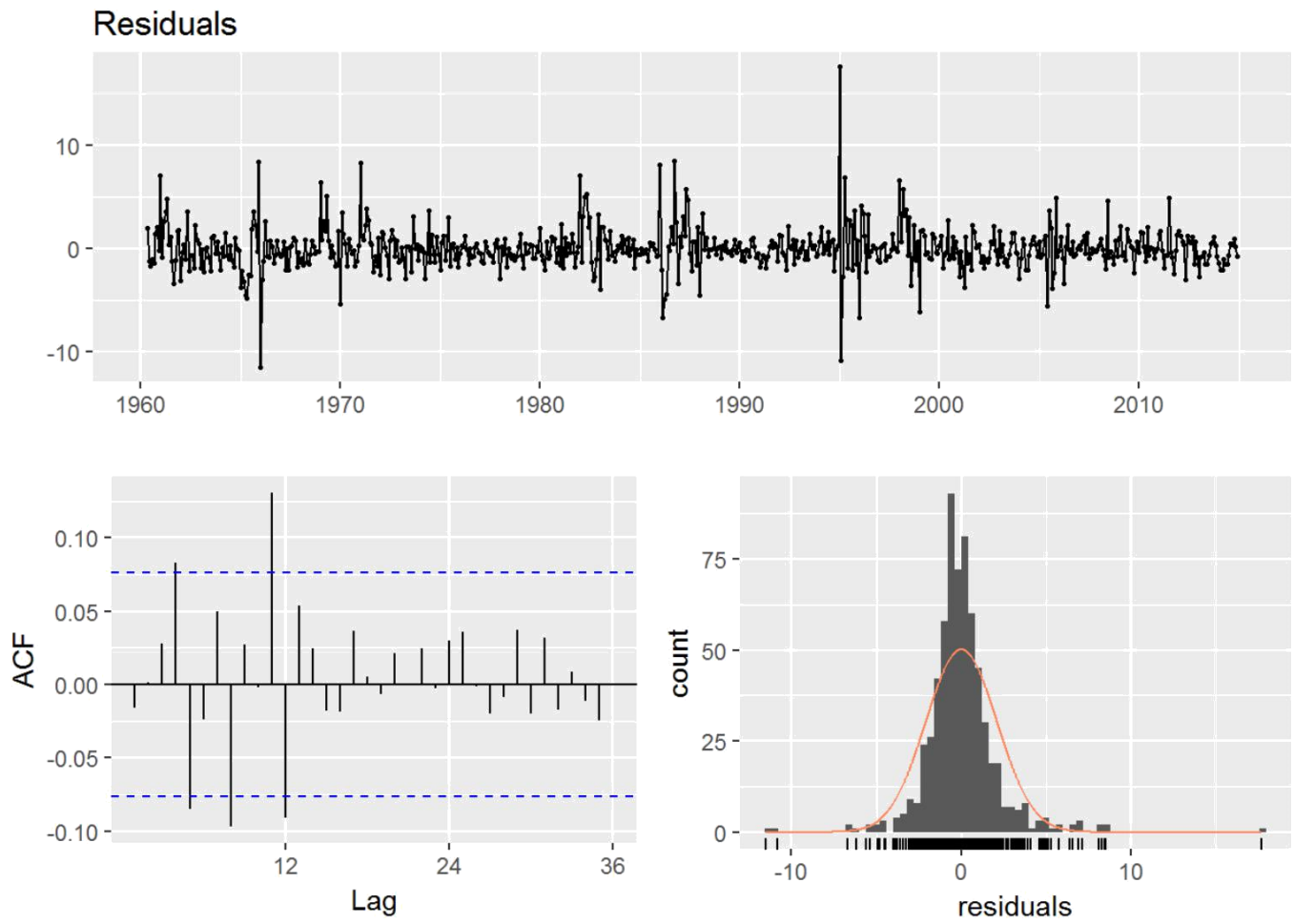
Figure18: Residuals plot of Model 23

Since the residuals result getting better, and autocorrelation in ACF plot getting fewer, another lag in dependent variable is added.

```
m24=dynlm(Y.t ~ X.t+L(X.t , k = 1 )+ L(X.t , k = 2 )+ L(X.t , k = 3) + L(Y.t , k = 1 ) + L(Y.t , k = 2 )+ L(Y.t , k = 3 )+L(Y.t , k = 4)+season(Y.t))
summary(m24)
```

```
##
## Time series regression with "ts" data:
## Start = 1960(5), End = 2014(12)
##
## Call:
## dynlm(formula = Y.t ~ X.t + L(X.t, k = 1) + L(X.t, k = 2) + L(X.t,
##     k = 3) + L(Y.t, k = 1) + L(Y.t, k = 2) + L(Y.t, k = 3) +
##     L(Y.t, k = 4) + season(Y.t))
##
## Residuals:
##      Min        1Q    Median        3Q     Max
## -11.5168   -0.9660  -0.1436   0.7828 17.6796
##
## Coefficients: (3 not defined because of singularities)
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.77882    0.50531   9.457  < 2e-16 ***
## X.t             -0.43285    0.30683  -1.411   0.1588
## L(X.t, k = 1)         NA         NA      NA       NA
## L(X.t, k = 2)         NA         NA      NA       NA
## L(X.t, k = 3)         NA         NA      NA       NA
## L(Y.t, k = 1)    1.01519    0.03879  26.174  < 2e-16 ***
## L(Y.t, k = 2)    0.13412    0.05570   2.408   0.0163 *
## L(Y.t, k = 3)   -0.08499    0.05569  -1.526   0.1275
## L(Y.t, k = 4)   -0.19129    0.03888  -4.920 1.10e-06 ***
## season(Y.t)Feb   0.59199    0.42960   1.378   0.1687
## season(Y.t)Mar   2.09149    0.48533   4.309 1.90e-05 ***
## season(Y.t)Apr -0.24124     0.56295  -0.429   0.6684
## season(Y.t)May   0.42047    0.61651   0.682   0.4955
## season(Y.t)Jun -1.44450     0.65958  -2.190   0.0289 *
## season(Y.t)Jul -2.791300.65395  -4.268 2.27e-05 ***
## season(Y.t)Aug -5.884250.64646  -9.102  < 2e-16 ***
## season(Y.t)Sep -7.076860.59197 -11.955  < 2e-16 ***
## season(Y.t)Oct -6.608480.52512 -12.585  < 2e-16 ***
## season(Y.t)Nov -4.677160.45876 -10.195  < 2e-16 ***
## season(Y.t)Dec -2.141750.42263  -5.068 5.28e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.113 on 639 degrees of freedom
## Multiple R-squared:  0.9548, Adjusted R-squared:  0.9536
## F-statistic: 843.2 on 16 and 639 DF,  p-value: < 2.2e-16
```

```
checkresiduals(m24)
```




```
 ##
 ##  Breusch-Godfrey test for serial correlation of order up to 24
##
 ## data:  object
 ## LM test = 36.782, df = 24, p-value = 0.046
```

Figure19: Residuals plot of Model 24

Now finally BG-test result indicates the residuals in not significant, even though there is still some correlation and changing variance. Hence, we can conclude that “m24” model, with the third lag in X variable and fourth lag in Y variable is the best fit for Dynamic Linear Model approach in terms of residuals.

Next we will see whether VIF test confirm this result.

```
MASE.dynlm(ma,mb,mc,m16,m17,m18,m19,m20,m21,m22,m23,m24)
```

```
##       n      MASE
## ma  659 0.3614931
## mb  658 0.5150954
## mc  658 0.3590523
## m16 659 0.3739879
## m17 658 0.5866566
## m18 658 0.3632657
## m19 657 0.3520468
## m20 657 0.3520468
## m21 657 0.3520468
## m22 657 0.3520468
## m23 657 0.3520468
## m24 656 0.3483075
```

As expected, “m24” has the lowest MASE. Hence, this model is chosen as the best fit fot Dynamic Linear Model approach. Now, we will forecast the model.

```
Y.t=s
q = 24
n = nrow(m24$model)
s.rad = array(NA , (n + q))
X.t.1 = Lag(X.t,+1)
X.t.2 = Lag(X.t,+2)
X.t.3 = Lag(X.t,+3)
s.rad[1:n] = Y.t[4:length(Y.t)]

for (i in 1:q){
  months = array(0,11)
  months[(i-1)%%12] = 1
  print(months)
  data.new = c(1,X.t[n],X.t.1[n], X.t.2[n], X.t.3[n],s.rad[n-1+i],s.rad[n-2+i],s.rad[n-3+i],s.rad[n-4+i],months)

        s.rad[n+i] = as.vector(m24$coefficients) %*% data.new
}
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0
##  [1] 1 0 0 0 0 0 0 0 0 0 0
##  [1] 0 1 0 0 0 0 0 0 0 0 0
##  [1] 0 0 1 0 0 0 0 0 0 0 0
##  [1] 0 0 0 1 0 0 0 0 0 0 0
##  [1] 0 0 0 0 1 0 0 0 0 0 0
##  [1] 0 0 0 0 0 1 0 0 0 0 0
##  [1] 0 0 0 0 0 0 1 0 0 0 0
##  [1] 0 0 0 0 0 0 0 1 0 0 0
##  [1] 0 0 0 0 0 0 0 0 1 0 0
##  [1] 0 0 0 0 0 0 0 0 0 1 0
##  [1] 0 0 0 0 0 0 0 0 0 0 1
##  [1] 0 0 0 0 0 0 0 0 0 0 0
##  [1] 1 0 0 0 0 0 0 0 0 0 0
##  [1] 0 1 0 0 0 0 0 0 0 0 0
##  [1] 0 0 1 0 0 0 0 0 0 0 0
##  [1] 0 0 0 1 0 0 0 0 0 0 0
##  [1] 0 0 0 0 1 0 0 0 0 0 0
##  [1] 0 0 0 0 0 1 0 0 0 0 0
##  [1] 0 0 0 0 0 0 1 0 0 0 0
##  [1] 0 0 0 0 0 0 0 1 0 0 0
##  [1] 0 0 0 0 0 0 0 0 1 0 0
##  [1] 0 0 0 0 0 0 0 0 0 1 0
##  [1] 0 0 0 0 0 0 0 0 0 0 1
```

```
result<-s.rad[(n+1):(n+q)]
result
```

```
##  [1] 8.387430 11.214959 16.809071 21.026096 26.182939 29.094241 29.909339
26.826285
##   [9] 21.397853 15.286977 10.498671  8.325364  9.226827 11.900397 17.359061
21.290421
##  [17] 26.306230 29.076997 29.780595 26.632149 21.161283 15.034912 10.252084
8.098383
```

```
{plot(Y.t,xlim=c(1960,2016),main = "Time Series Plot of Forecasting using Dynamic
L inear Model Approach")
lines(ts(result,start=c(2015,1),frequency = 12),col="red")}
```

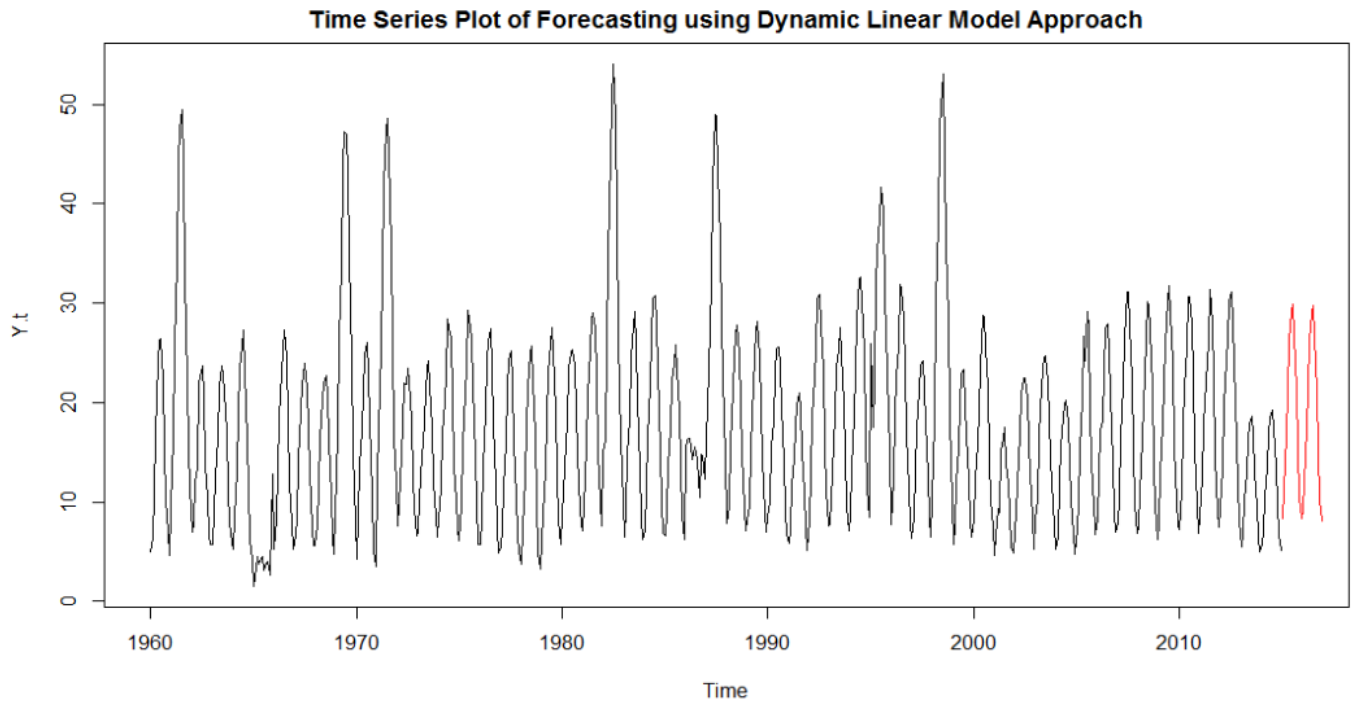
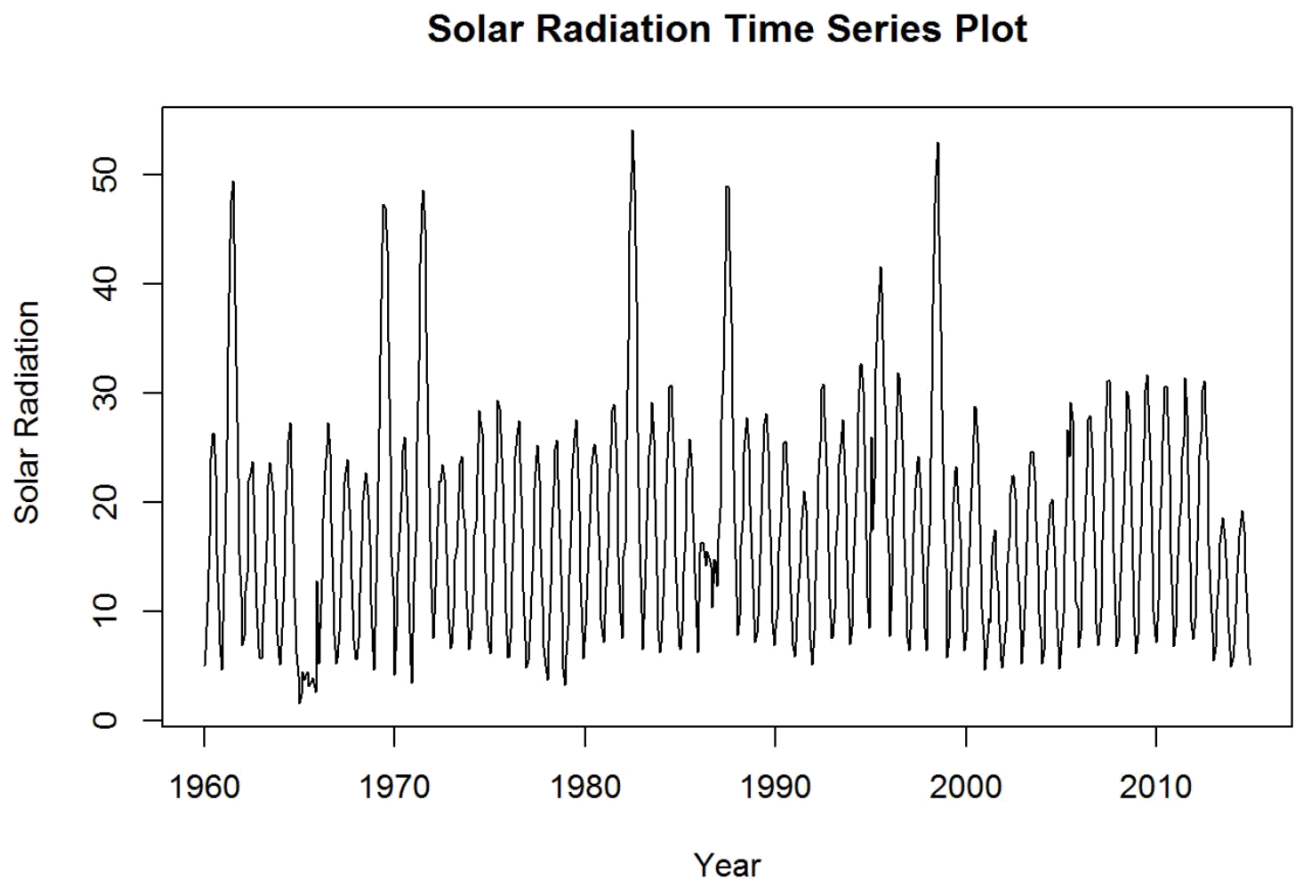


Figure 20: Forecasting result of Model 24

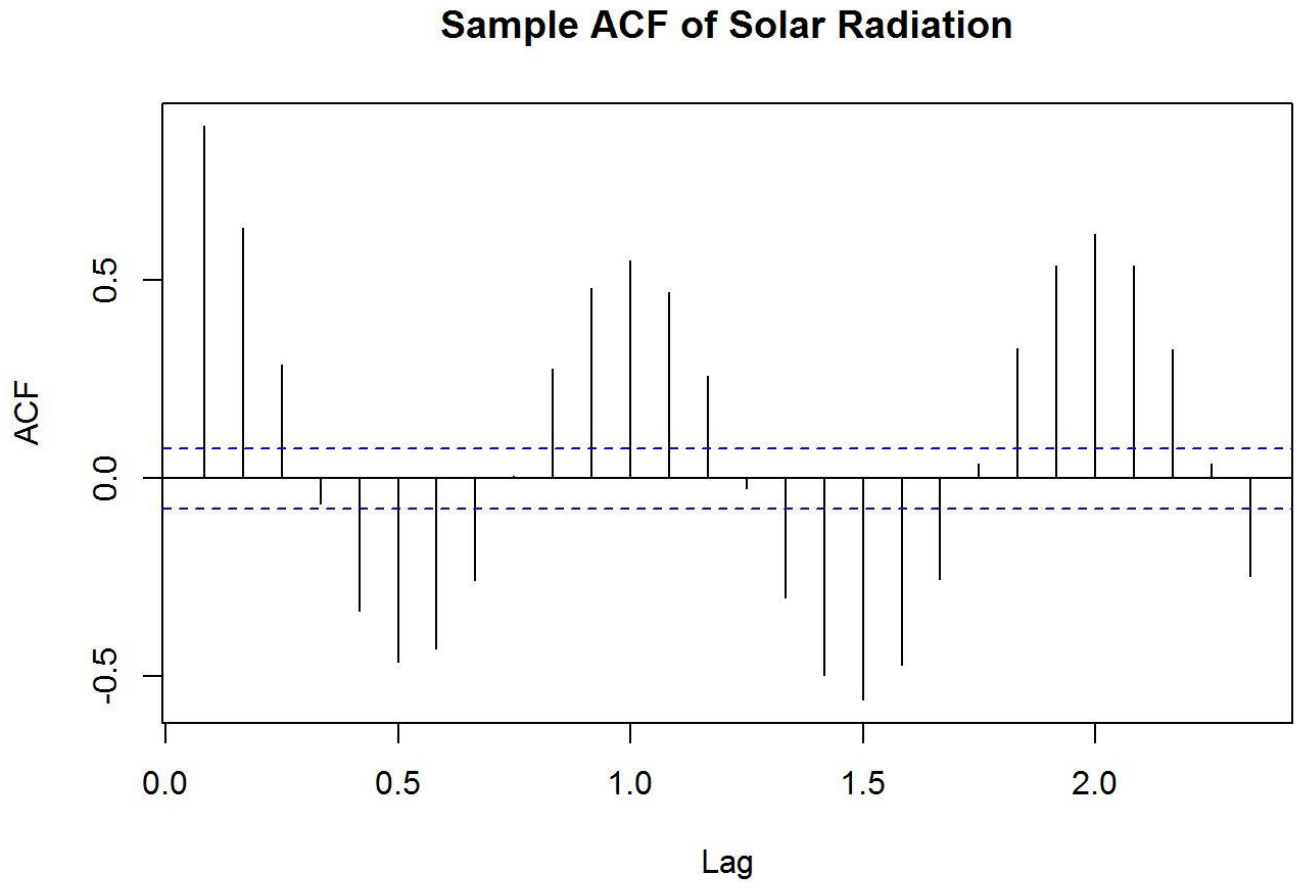
3. Exponential Smoothing and State Space Model

To explore the model further, we will fit different trend and seasonality patterns by only use solar radiation series.
First, we will examine the series, ACF, and PACF plots.

```
plot(s , type="l", xlab= "Year", ylab=" Solar Radiation", main="Solar Radiation
Time Series Plot")
```




```
acf(s, main = "Sample ACF of Solar Radiation")
```



```
pacf(s, main = "Sample ACF of Solar Radiation")
```

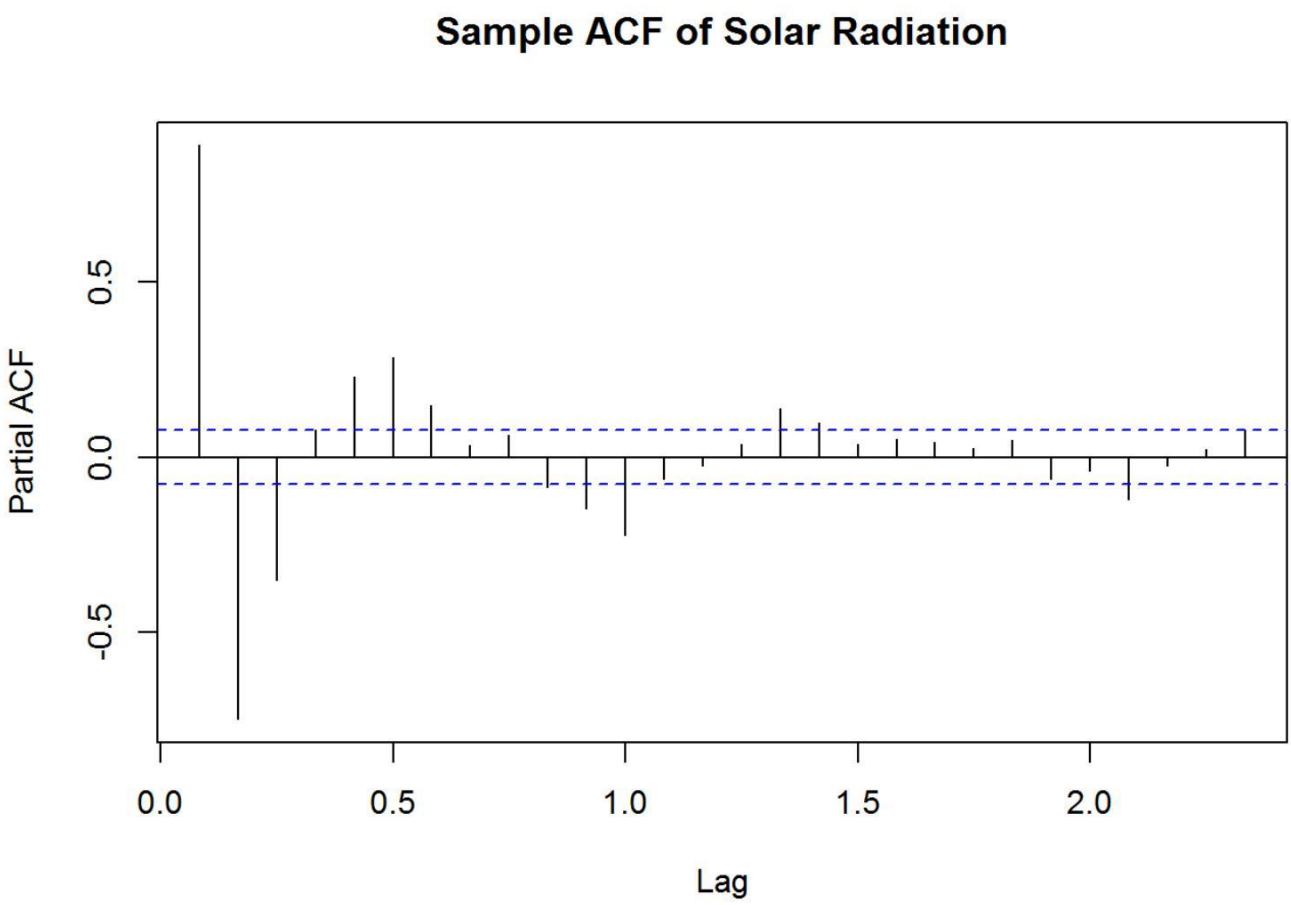


Figure 21: Time Series, ACF, and PACF Plot of Solar Radiation

The time series plot indicates seasonal pattern and changing variance, but no visible trends. It is confirmed by ACF and PACF plot, especialy the sample ACF plot that clearly show the seasonal part. Based on this observation, we will just fit Holt-Winter's models to the series, since it is best for seasonal part.

```
m25<-hw(s,seasonal = "additive")
summary(m25)
```

```
 ##
 ## Forecast method: Holt-Winters' additive method
##
 ## Model Information:
 ## Holt-Winters' additive method
 ##
 ## Call:
 ##  hw(y = s, seasonal = "additive")
##
 ##   Smoothing parameters:
 ##     alpha = 0.9968
 ##     beta  = 0.0079
 ##     gamma = 0.0027
 ##
 ##   Initial states:
 ##     l = 12.813
```

```
##     b = 0.4276
##     s=-10.6349 -7.3748 -2.6593 2.7233 7.775 11.0058
##            9.8199 6.1144 1.8544 -1.8065 -7.0856 -9.7316
##
##   sigma:  2.3699
##
##      AIC     AICc      BIC
## 5457.817 5458.770 5534.185
##
## Error measures:

## ME RMSE MAE MPE MAPE MASE ## Training set -0.08375221 2.369864 1.547273 -
1.615444 12.99165 0.2541887 ## ACF1

## Training set 0.163735
##
## Forecasts:
##           Point Forecast      Lo 80     Hi 80       Lo 95    Hi 95
## Jan 2015        5.899303   2.862201  8.936406   1.2544557 10.54415
## Feb 2015        8.536199   4.213959 12.858438   1.9259038 15.14649
## Mar 2015       13.828280   8.509665 19.146895   5.6941602 21.96240
## Apr 2015       17.502130  11.334239 23.670021   8.0691544 26.93511
## May 201521.822830  14.898340 28.747319  11.2327369 32.41292
## Jun 201525.314433  17.698277 32.930589  13.6665276 36.96234
## Jul 201526.552786  18.293496 34.812075  13.9212921 39.18428
## Aug 2015       23.394989  14.530464 32.259514   9.8378675 36.95211
## Sep  2015      18.270816   8.831599 27.710033   3.8347798 32.70685
## Oct  2015      12.811417   2.822722 22.800112  -2.4649740 28.08781
## Nov 20158.147760  -2.369208 18.664727  -7.9365542 24.23207
## Dec 20155.037795  -5.991806 16.067396 -11.8305235 21.90611
## Jan 20165.789632  -5.734380 17.313644 -11.8348239 23.41409
## Feb 20168.426527  -3.578240 20.431294  -9.9331799 26.78623
## Mar 2016       13.718608   1.245117 26.192099 -5.3579498 32.79517
## Apr 2016       17.392458   4.460922 30.323995 -2.3846202 37.16954
## May 2016       21.713158   8.333114 35.093201   1.2501467 42.17617
## Jun 2016       25.204761  11.384778 39.024744   4.0689210 46.34060
## Jul 2016       26.443114  12.190926 40.695302   4.6462733 48.23995
## Aug 2016       23.285317   8.607936 37.962698   0.8381999 45.73243
## Sep 2016       18.161144   3.064951 33.257337  -4.9264910 41.24878
## Oct 201612.701745  -2.807433 28.210923 -11.0174965 36.42099
## Nov 20168.038088  -7.878738 23.954914 -16.3045970 32.38077
## Dec 20164.928123 -11.393259 21.249506 -20.0332775 29.88952
```

```
m26<-hw(s, seasonal = "multiplicative")
summary(m26)
```

```
##      AIC     AICc      BIC
## 6420.503 6421.456 6496.871
##
## Error measures:
```

```
## ME RMSE MAE MPE MAPE MASE ## Training set -0.1060967 2.062279 1.255284 -
2.17078 10.01439 0.2062203 ## ACF1
```

```
m27<-hw(s,seasonal = "additive", damped=TRUE)
summary(m27)
```

```
##      AIC     AICc      BIC
## 5423.009 5424.076 5503.869
##
## Error measures:

## ME RMSE MAE MPE MAPE MASE ## Training set -0.0003458727 2.304693 1.479661 -
1.293116 12.22459 0.2430814 ## ACF1
```

```
m28<-hw(s, seasonal = "multiplicative", exponential=TRUE)
summary(m28)
```

```
##      AIC     AICc      BIC
## 6733.816 6734.769 6810.184
##
## Error measures:

## ME RMSE MAE MPE MAPE MASE ## Training set 0.2010306 2.222834 1.403495 -
0.8975766 11.4617 0.2305686 ## ACF1
##
```

```
m29<-hw(s,seasonal = "multiplicative", damped=TRUE)
summary(m29)
```

```
##      AIC     AICc      BIC
## 6327.138 6328.205 6407.999
##
## Error measures:

## ME RMSE MAE MPE MAPE MASE ## Training set -0.03547783 2.039583 1.240267 -
2.200423 10.02395 0.2037532 ## ACF1
```

It turns out “m29” model with multiplicative seasonal and damped trend has the lowest MASE (0.2037).

Now we will take a look at state space model and see whether MASE score can be improved. Here we will set all the trends as none, since there are no obvious trend in the series. Hence, only 3 models can be fitted with state space model.

```r
m30<-ets(s, model="ANA")
summary(m30)
```

```
## ETS(A,N,A)
##
## Call:
##  ets(y = s, model = "ANA")
##
##   Smoothing parameters:
##     alpha = 0.9752
##     gamma = 0.0248
##
##   Initial states:
##     l = 22.4659
##     s=-10.2239 -8.9188 -3.0532 2.3506 8.2098 10.8414
##            10.4077 7.5173 1.4119 -2.4583 -6.8829 -9.2016
##
##   sigma:  2.3788
##
##      AIC     AICc      BIC
## 5458.759 5459.505 5526.143
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.01014996 2.378754 1.557999 -1.880712 13.08487 0.2559508
##                   ACF1
## Training set 0.185752
```

```r
m31<-ets(s, model="MNA")
summary(m31)
```

```
## ETS(M,N,A)
##
## Call:
##  ets(y = s, model = "MNA")
##
##   Smoothing parameters:
##     alpha = 0.4777
##     gamma = 1e-04
##
##   Initial states:
##     l = 21.5697
##     s=-10.1753 -7.1745 -4.0165 0.0827 7.1147 7.8517
##            12.2277 6.0807 2.1198 -0.5072 -6.0681 -7.5357
##
##   sigma:  0.334
##
##      AIC     AICc      BIC
## 6496.630 6497.376 6564.014
##
## Training set error measures:
##                       ME    RMSE      MAE        MPE     MAPE      MASE
## Training set -0.02316152 3.6531 2.621824 -6.455377 20.94911 0.4307179
##                    ACF1
## Training set 0.4615006
```

```
m32<-ets(s, model="MNM")
summary(m32)
```

```
## ETS(M,N,M)
##
## Call:
##  ets(y = s, model = "MNM")
##
##   Smoothing parameters:
##     alpha = 0.7065
##     gamma = 0.0804
##
##   Initial states:
##     l = 21.5335
##     s=0.8906 0.3179 0.6025 0.9817 1.2849 1.4813
##            1.5419 1.375 1.1558 0.9043 0.6746 0.7896
##
##   sigma:  0.2323
##
##      AIC     AICc      BIC
## 5988.832 5989.577 6056.215
##
## Training set error measures:
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.2126627  3.195065 2.043712 -5.568916 17.95583 0.3357446
##                    ACF1
## Training set 0.2896291
```

```
models=c("ANA","MNA","MNM")
fit.AICc=array(NA,3)
levels=array(NA, dim=c(3,1))
expand=expand.grid(models)
for (i in 1:3){
  fit.AICc[i]=ets(s, model=toString(expand[i,1]))$aicc
  levels[i,1]=toString(expand[i,1])
}
results=data.frame(levels,fit.AICc)
colnames(results)=c("Model", "AICc")
results
```

```
##   Model     AICc
## 1   ANA 5459.505
## 2   MNA 6497.376
## 3   MNM 5989.577
```

Here, all of the models agree that “m30” (Additive, None, Additive) state space model has the lowest MASE(0.2560) and AICc value (5459.505). However, the value is still lower than previous model (“m29” model with MASE value of 0.2037). Hence, model “m29” will be used for forecasting.

```
{plot(m29, type= "l", ylab="Solar Radiation",
     xlab="Year", fcol="red", plot.conf=FALSE)
  lines(fitted(m29), col="blue")}
```

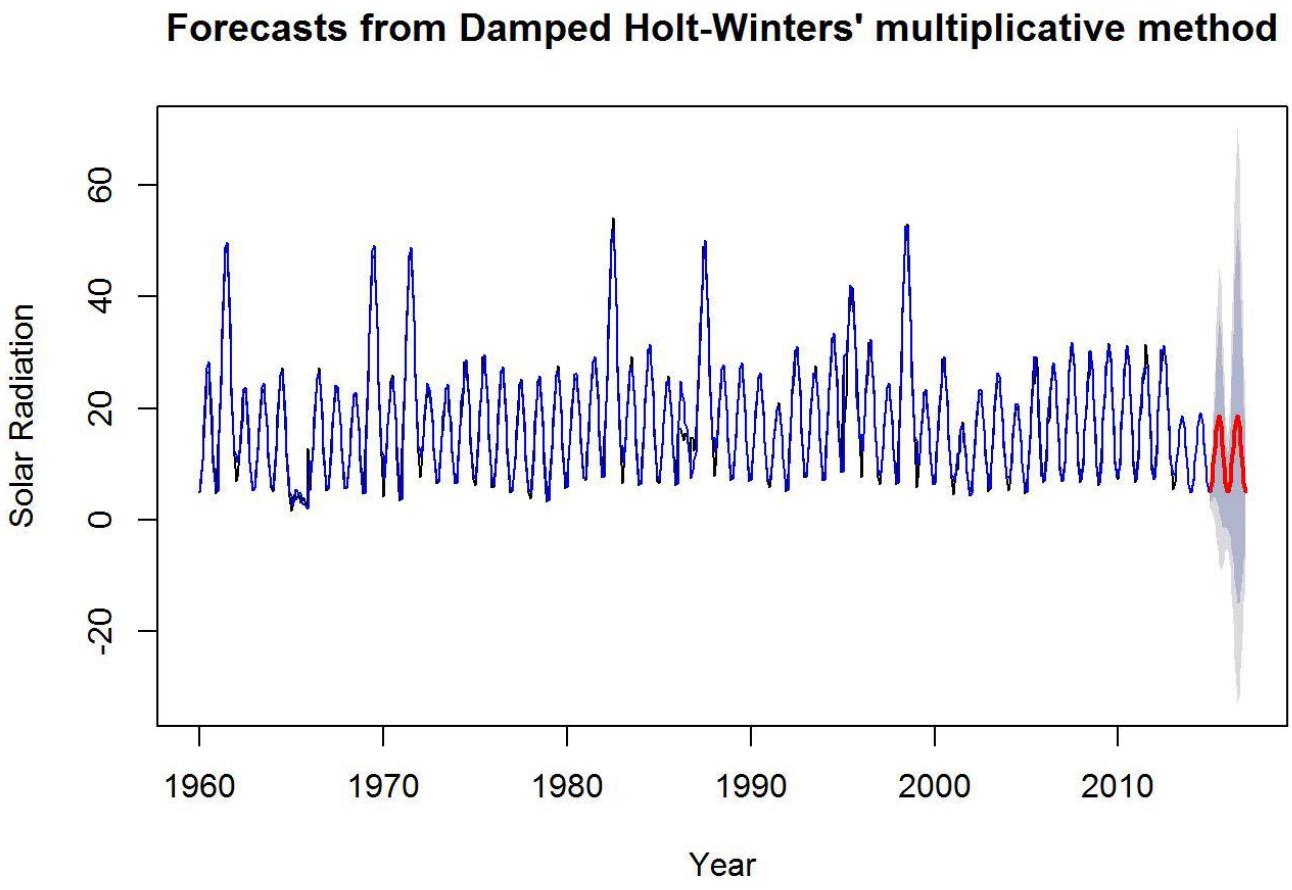


Figure 22: Time Series Plot of Forecasting using Exponential Smoothing and State Space Model

Task 2

Now we will examine whether there are spurious correlation between quarterly Residential Property Price Index (PPI) in Melbourne and quarterly population change over previous quarter in Victoria between September 2003 and December 2016. The time series plot is presented below:

```
price.data<-read_csv("data2.csv")
```

```
## Parsed with column specification:
## cols(
##   Quarter = col_character(),
##   price = col_double(),
##   change = col_integer()
## )
```

```
price = ts(price.data$price,start = c(2003,3),frequency = 4)
change = ts(price.data$change, start =c(2003,3),frequency = 4)
pricedata.ts = ts(price.data[,2:3],start = c(2003,3),frequency = 4)

plot(pricedata.ts, xlab = "Year", main = "Time series plot of Property Price and Po
pulation Change in Victoria", type="l", yax.flip=T)
```

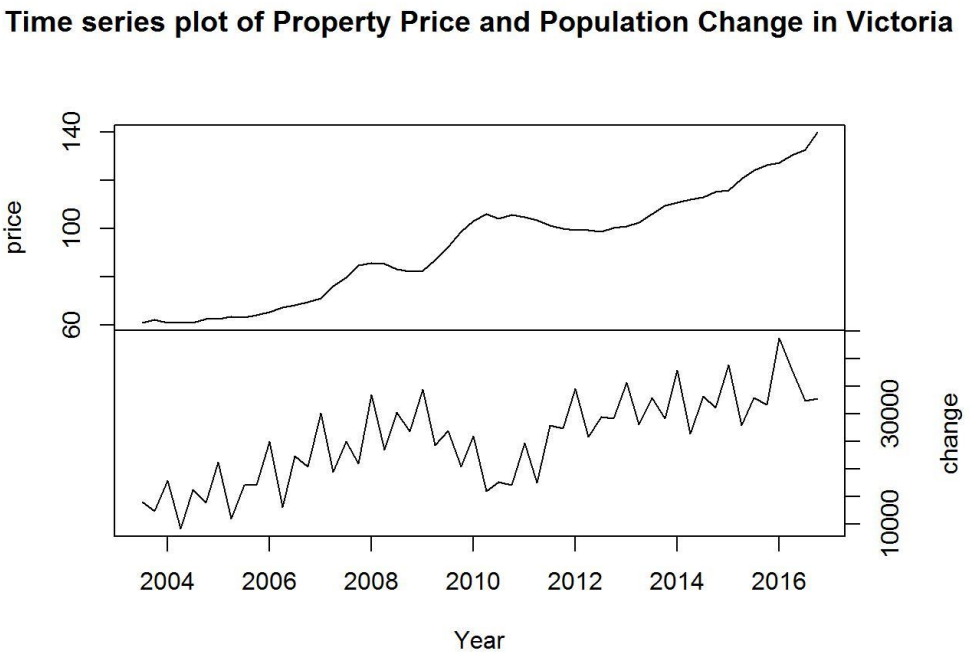


Figure 23: Time Series Plot of property Price and Population Change in Victoria

Here we can see upward trend for both series, shows the possibility of correlation between residential PPI and population change. It also proven by CCF plot below, that shows nearly all of the cross-correlations are significantly different from zero.

```
ccf(as.vector(pricedata.ts [,1]), as.vector(pricedata.ts[, 2]),ylab='CCF', main =
"S ample CCF between Property Price and Population Change in Victoria")
```

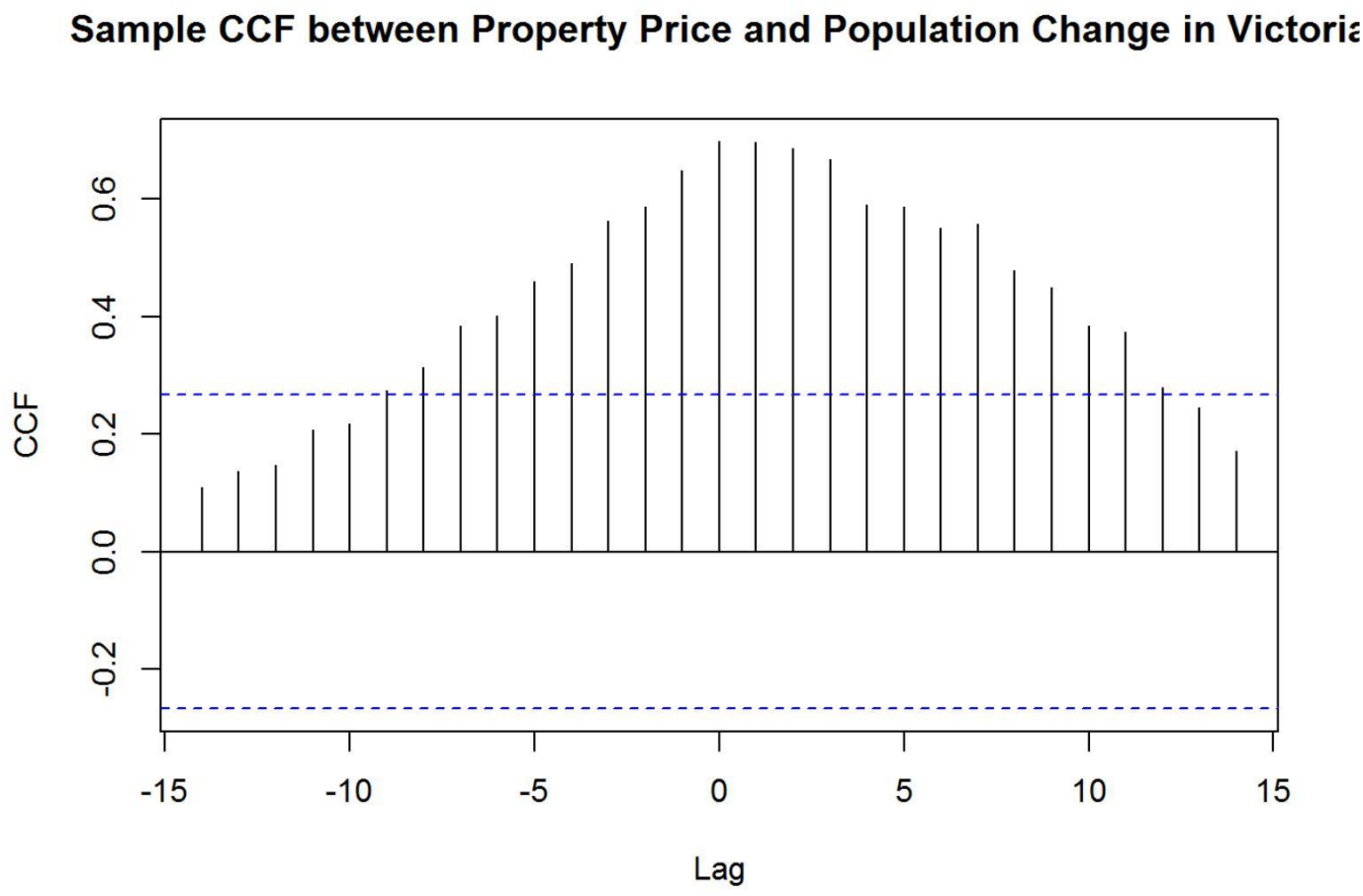


Figure 24: CCF Plot of Property Price and Population Change in Victoria

However, we will examine if this correlation is real or spurious, first by using CCF after first difference of both series.

```
ccf(as.vector(diff(pricedata.ts[,1])),as.vector(diff(pricedata.ts[ ,2])),ylab='CCF
', main = "Sample CCF after first difference between Property Price and Population
Change in Victoria")
```

Figure 25: CCF Plot of Property Price and Population Change in Victoria after First Difference

Now the autocorrelation is significantly reduced. From this plot, the assumption that the correlation is real becomes stronger. However, to be able to clearly see the correlation, we will do pre-whitening and see the sample CCF after pre-whitening.

```
price.white=ts.intersect(diff(diff(price,4)),diff(diff(log(change),4)))
prewhitened = TSA::prewhiten(as.vector(price.white[,1]),as.vector(price.white[,2])
,ylab='CCF', main="Sample CFF after prewhitening")
```

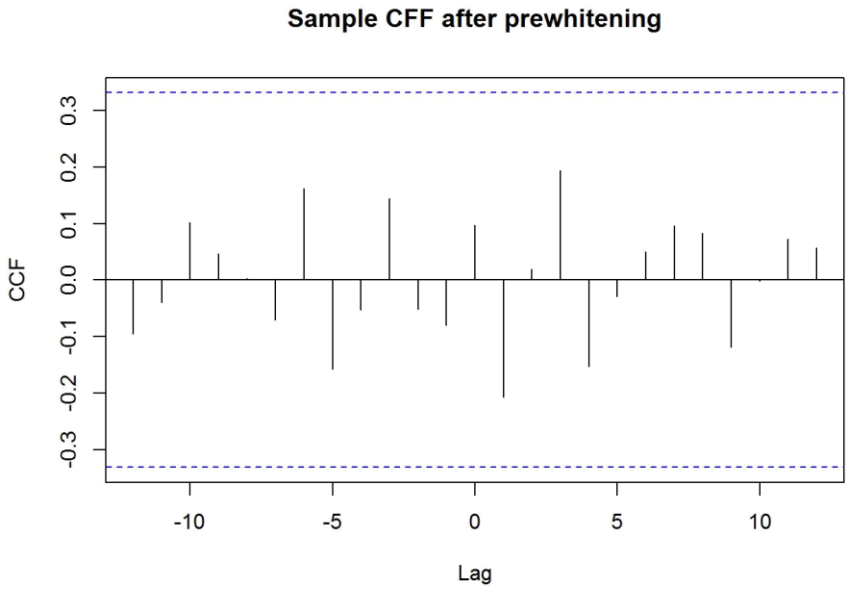


Figure 26: CCF Plot of Property Price and Population Change in Victoria after Pre-Whitening

Here, after pre-whitening, the CCF plot becomes white noise. Means, all the correlations in the previous plot is false alarm, and quarterly Residential Property Price Index (PPI) in Melbourne and quarterly population change over previous quarter in Victoria between September 2003 and December 2016 is not correlated.