Times Series Assignment 1

12111603 Tan Zhiheng

2023-10-01

1

Essencially, X, Y, Z all obeys $Hypergeometric(5, 6, 15)$ and $Cov(X, Y) = Cov(Y, Z) = Cov(X, Z) = \rho\sigma^2$. Noticed $X + Y + Z = 6$, which is a fixed constant. Thus, we can have:

$$Var(X + Y + Z) = 0 = n\sigma^2 + n(n-1)\rho\sigma^2.$$

Finally, we can obtain $\rho = Cov(X, Y) = Cov(Y, Z) = Cov(X, Z) = -\frac{1}{n-1} = -\frac{1}{2}$.

2

To minimize $L := E(X_{t+h} - f(X_t))^2 = E(X_{t+h} - (a(X_t - \mu) + b))^2 = E(X_{t+h} - (aX_t + \tilde{b}))^2$ where we replace $-a\mu + b$ with $\tilde{b}$

We performance partial derivative on a and b, then let them equal zero:

$\frac{\partial L}{\partial a} = 0$ can derive

$$E(X_t(X_{t+h} - aX_t - b)) = 0$$

which means

$$E(X_t X_{t+h}) - bEX_t - aE(X_t^2) = 0$$

Similarly, $\frac{\partial L}{\partial b} = 0$ can derive

$$E(X_{t+h} - aX_t - b) = 0$$

which means $b = EX_{t+h} - aEX_t$

Since $\rho(h) = \frac{\gamma(h)}{\gamma(0)} = \frac{E(X_t X_{t+h}) - EX_t\, EX_{t+h}}{E(X_t^2) - (EX_t)^2}$, then we bring it into the above equation.

Consequently, we can get $a = \rho(h)$ and $\tilde{b} = \mu(1 - \rho(h))$

Then replace b back from $\tilde{b}$, finally we can conclude the optimal solution to the problem is:

$$a = \rho(h)$$

and

$$b = \mu$$

3

(a)

In a weakly stationary process,

because

$$\rho(h) = \frac{\gamma(h)}{\gamma(0)} \in [-1, 1]$$

and

$$\gamma(0) = \sigma^2 \geq 0$$

So we can state that

$$\gamma(0) \geq |\gamma(h)|, \quad \forall h$$

Here comes another perspective to demonstrate this conclusion.

Because $Cov(\cdot, \cdot)$ satisfies the following definition of inner product, then it can treated as a kind of inner product.

- definite positive
- symmetry
- conjugate linearity

Then by the Cauchy-Schwarz Inequality,

$$|\gamma(h)| = |Cov(X_{t+h}, X_t)| = |< X_{t+h}, X_t >| \leq ||X_{t+h}|| \cdot ||X_t|| = Var(X_t) = \gamma(0)$$

Hence the conclusion has been verified.

(b)

$$\gamma(h) = \gamma(h, 0) = \gamma(t+h, t) = \gamma(t, t+h) = \gamma(-h, 0)$$

where the third equation is supported by the second property of inner product: symmetry.

4

(a)

When μ is known, then

$$\begin{aligned}
E(\tilde{\gamma}'(h)) &= \frac{1}{n-|h|} \sum_{t=1}^{n-|h|} E[(X_t - \mu)(X_{t+|h|} - \mu)] \\
&= \frac{1}{n-|h|} \sum_{t=1}^{n-|h|} Cov(X_{t+|h|}, X_t) \\
&= \frac{1}{n-|h|} \cdot (n-|h|)\gamma(|h|) \\
&= \gamma(|h|) \\
because\ of\ stationary\ process, \\
&= \gamma(h).
\end{aligned}$$

Similarly, we have

$$E(\hat{\gamma}'(h)) = E(\frac{n-|h|}{n}\tilde{\gamma}'(h)) = \frac{n-|h|}{n}\gamma(h).$$

Thus,$\tilde{\gamma}'(h)$ is a unbiased estimator of $\gamma(h)$ and $\hat{\gamma}'(h)$ is NOT a unbiased estimator of $\gamma(h)$.

(b)

$$\begin{aligned}
E(\tilde{\gamma}(h)) &= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} E[(X_t-\bar{X})(X_{t+|h|}-\bar{X})] \\
&= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} E[(X_tX_{t+|h|})-(X_t\bar{X})-(\bar{X}X_{th})+(\bar{X})^2] \\
&= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} [E(X_tX_{t+|h|})-E(X_t\bar{X})-E(\bar{X}X_{th})+E(\bar{X})^2] \\
because\ uncorrelated, \\
&= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} [-\frac{\sigma^2}{n}-\frac{\sigma^2}{n}+VarX] \\
&= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} [-\frac{2\sigma^2}{n}+\frac{\sigma^2}{n}] \\
&= \frac{1}{n-|h|}\sum_{t=1}^{n-|h|} \frac{-\sigma^2}{n} \\
&= -\frac{\sigma^2}{n} \\
because\ of\ stationary\ process, \\
&= -\frac{\gamma(0)}{n}.
\end{aligned}$$

Similarly,

$$
\begin{aligned}
E(\hat{\gamma}(h)) &= \frac{1}{n}\sum_{t=1}^{n-|h|} E[(X_t-\bar{X})(X_{t+|h|}-\bar{X})] \\
&= \frac{1}{n}\sum_{t=1}^{n-|h|} E[(X_t X_{t+|h|}) - (X_t\bar{X}) - (\bar{X}X_{th}) + (\bar{X})^2] \\
&= \frac{1}{n}\sum_{t=1}^{n-|h|} [E(X_t X_{t+|h|}) - E(X_t\bar{X}) - E(\bar{X}X_{th}) + E(\bar{X})^2] \\
&because\ uncorrelated, \\
&= \frac{1}{n}\sum_{t=1}^{n-|h|} [-\frac{\sigma^2}{n} - \frac{\sigma^2}{n} + VarX] \\
&= \frac{1}{n}\sum_{t=1}^{n-|h|} [-\frac{2\sigma^2}{n} + \frac{\sigma^2}{n}] \\
&= \frac{1}{n}\sum_{t=1}^{n-|h|} \frac{-\sigma^2}{n} \\
&= -(1-\frac{|h|}{n})\frac{\sigma^2}{n} \\
&because\ of\ stationary\, process, \\
&= -(1-\frac{|h|}{n})\frac{\gamma(0)}{n}.
\end{aligned}
$$

Consequently, $E(\tilde{\gamma}(h)) = -\frac{\gamma(0)}{n}$ and $E(\hat{\gamma}(h)) = -(1-\frac{|h|}{n})\frac{\gamma(0)}{n}$.

(c)

We want to verify $\sum_{h=-(n-1)}^{(n-1)}\hat{\gamma}(h) = \sum_{h=-(n-1)}^{(n-1)}\frac{1}{n}\sum_{t=1}^{n-|h|}(X_t-\bar{X})(X_{t+|h|}-\bar{X}) = 0$, which is to show

$$
\sum_{h=-(n-1)}^{(n-1)}\sum_{t=1}^{n-|h|}(X_t-\bar{X})(X_{t+|h|}-\bar{X}) = 0
$$

Consider the sampled variance-covariance matrix:

$$
\begin{bmatrix}
(X_1-\bar{X})(X_1-\bar{X}) & (X_1-\bar{X})(X_2-\bar{X}) & \cdots & (X_1-\bar{X})(X_n-\bar{X}) \\
(X_2-\bar{X})(X_1-\bar{X}) & (X_2-\bar{X})(X_2-\bar{X}) & \cdots & (X_2-\bar{X})(X_n-\bar{X}) \\
\vdots & \vdots & \ddots & \vdots \\
(X_n-\bar{X})(X_1-\bar{X}) & (X_n-\bar{X})(X_2-\bar{X}) & \cdots & (X_n-\bar{X})(X_n-\bar{X})
\end{bmatrix}
$$

We can figure out that calculating $\sum_{h=-(n-1)}^{(n-1)}\sum_{t=1}^{n-|h|}(X_t-\bar{X})(X_{t+|h|}-\bar{X}) = 0$ is just to add up all the elements in the matrix above. Surprisingly, the sum of every row is always zero, with proof as follows.

Take the i^{th} row for example.

$$\begin{aligned}\sum(all\ \ elements\ \ in\ \ i^{th}\ \ row) &= (X_i-\bar{X})(\sum_{j=1}^{n}(X_j-\bar{X}))\\ &= (X_i-\bar{X})(\sum_{j=1}^{n}X_j-n\bar{X})\\ &= (X_i-\bar{X})\cdot 0\\ &= 0.\end{aligned}$$

By the arbitrary choice of i, we can conclude that the sum of every row is always zero, thus all the elements in the matrix is added up to zero. Here completes the proof. Therefore, $\sum_{h=-(n-1)}^{(n-1)}\hat{\gamma}(h)=0$ has been verified.

Next, we want to verify $\sum_{h=-(n-1)}^{(n-1)}\hat{\gamma}'(h)\geq 0$

Consider another matrix:

$$\begin{bmatrix}(X_1-\mu)(X_1-\mu) & (X_1-\mu)(X_2-\mu) & \cdots & (X_1-\mu)(X_n-\mu)\\ (X_2-\mu)(X_1-\mu) & (X_2-\mu)(X_2-\mu) & \cdots & (X_2-\mu)(X_n-\mu)\\ \vdots & \vdots & \ddots & \vdots\\ (X_n-\mu)(X_1-\mu) & (X_n-\mu)(X_2-\mu) & \cdots & (X_n-\mu)(X_n-\mu)\end{bmatrix}$$

Similarly, we can figure out that calculating $\sum_{h=-(n-1)}^{(n-1)}\hat{\gamma}'(h)$ is just to add up all the elements in the matrix above together. First, we sum up arbitrary one of the row and see what we find.

Take i^{th} row for example.

$$\begin{aligned}\sum(i^{th}\ \ row) &= (X_i-\mu)(\sum_{j=1}^{n}(X_j-\mu))\\ &= (X_i-\mu)(\sum_{j=1}^{n}X_j-n\mu)\\ &= (X_i-\mu)(n\bar{X}-n\mu)\end{aligned}$$

Then, we add up all rows together:

$$\begin{aligned}\sum(all\ \ elements) &= \sum_{i=1}^{n}\sum(i^{th}\ \ row)\\ &= \sum_{i=1}^{n}(X_i-\mu)(n\bar{X}-n\mu)\\ &= (n\bar{X}-n\mu)(n\bar{X}-n\mu)\\ &= (n\bar{X}-n\mu)^2\\ &\geq 0\end{aligned}$$

So the second proposition has been checked.

To conclude,

$$\sum_{h=-(n-1)}^{(n-1)}\hat{\gamma}(h)=0$$

and

$$\sum_{h=-(n-1)}^{(n-1)} \hat{\gamma}'(h) \geq 0$$

has been proved.

5

Our goal is to verify the following statement:

- $E(X_t) = 0$
- $Var(X_t)$ is a constant independent of t
- uncorrelated
- NOT i.i.d

First, $E(X_t) = 0$:

Because $\{W_t\}$ is independent of $\{W_t\}$, then

$$E(X_t) = E(W_t(1 - W_{t-1})Z_t) = E(W_t(1 - W_{t-1}))E(Z_t)$$

Noticed that $E(Z_t) = 0$,

Consequently, $E(X_t) = 0$.

Secondly, $Var(X_t)$ is a constant independent of t:

$$\begin{aligned}
Var(X_t) &= E(X_t^2) = E(W_t^2(1 - W_{t-1})^2 Z_t^2) \\
&= E(W_t^2(1 - W_{t-1})^2) \cdot E(Z_t^2) \\
Calculate\ that\ E(Z_t^2) = 1, & \\
&= E(W_t^2(1 - W_{t-1})^2) \\
Since\ W_t\ and\ W_{t-1}\ is\ independent, & \\
&= E(W_t^2) \cdot E((1 - W_{t-1})^2) \\
&= E(W_t^2) \cdot E(W_{t-1}^2) \\
&= \frac{1}{4}
\end{aligned}$$

Obviously, variance is a constant independent of time t.

Thirdly, uncorrelated:

$$\begin{aligned}
Cov(X_m, X_n) &= E(X_m X_n) \\
&= E(W_m W_{m-1} Z_m W_n W_{n-1} Z_n) \\
&= E(W_m W_{m-1} W_n W_{n-1})\ E(Z_m)\ E(Z_n) \\
&= 0, \quad \forall m, n \in R.
\end{aligned}$$

Thus, X_i and X_j are uncorrelated.

Fourthly,

P(X_t = 1) = P(W_t = 1, W_{t-1} = 0, Z_t = 1) = ()^3 =

Similarly,

$$P(X_{t-1} = 1) = \frac{1}{8}$$

However,

$$P(X_t = 1, X_{t-1} = 1) = 0,$$

because if $X_t = 1$, then there must have $W_{t-1} = 0$, however, $X_{t-1} = 0$ is certain at this time.

Now, we can see

$$P(X_t = 1, X_{t-1} = 1) \neq P(X_t = 1) \cdot P(X_{t-1} = 1)$$

which means X_t and X_{t-1} are not independent.

Therefore, X_t is a white noise but NOT i.i.d.

6

(a)

Stationary process, because it is a Moving Average(3) with some parameters being zero.

$$E(X_t) = E(W_t) - E(W_{t-3}) = 0$$

$$\begin{aligned}
\gamma(h) &= Cov(X_{t+h}, X_t) = E(X_{t+h}X_t) \\
&= E(W_{t+h} - W_{t+h-3})(W_t - W_{t-3}) \\
&= E(W_{t+hW_t} - W_{t+h}W_{t-3} - W_{t+h-3}W_t + W_{t+h-3}W_{t-3}) \\
&= \begin{cases} 2, & if\ h = 0 \\ -1, & if\ h = 3 \\ -1, & if\ h = -3 \\ 0, & else \end{cases}
\end{aligned}$$

(b)

Stationary process.

$$E(X_t) = E(W_3) = 0$$

$$\gamma(h) = E(W_3^2) = 1$$

(c)

Not a stationary process, because $E(X_t) = t$ is dependent on time t.

(d)

Stationary process.

$$E(X_t) = E(W_t^2) = 1$$

$$\gamma(h) = E(W_t^2 W_{t+h}^2) - E(W_t^2)E(W_{t+h}^2) = E(W_t^2)E(W_{t+h}^2) - 1 = 0$$

(e)

Stationary process, we need to verify its expectation and autocovariance are both independent of time t.

$$E(X_t) = E(W_t W_{t-2}) = E(W_t)E(W_{t-2}) = 0$$

$$\begin{aligned}
\gamma(h) &= E(X_{t+h}E_t) \\
&= E(W_t W_{t+h} W_{t-2} W_{t+h-2}) \\
&= \begin{cases} E(W_t^2)E(W_{t-2}^2) = 1, & if\, h = 0 \\ E(W_t^2)E(W_{t+2})E(W_{t-2}) = 0, & if\, h = 2 \\ E(X_t)E(X_{t-2}^2)E(W_{t-4}) = 0, & if\, h = -2 \end{cases}
\end{aligned}$$

7

(a)

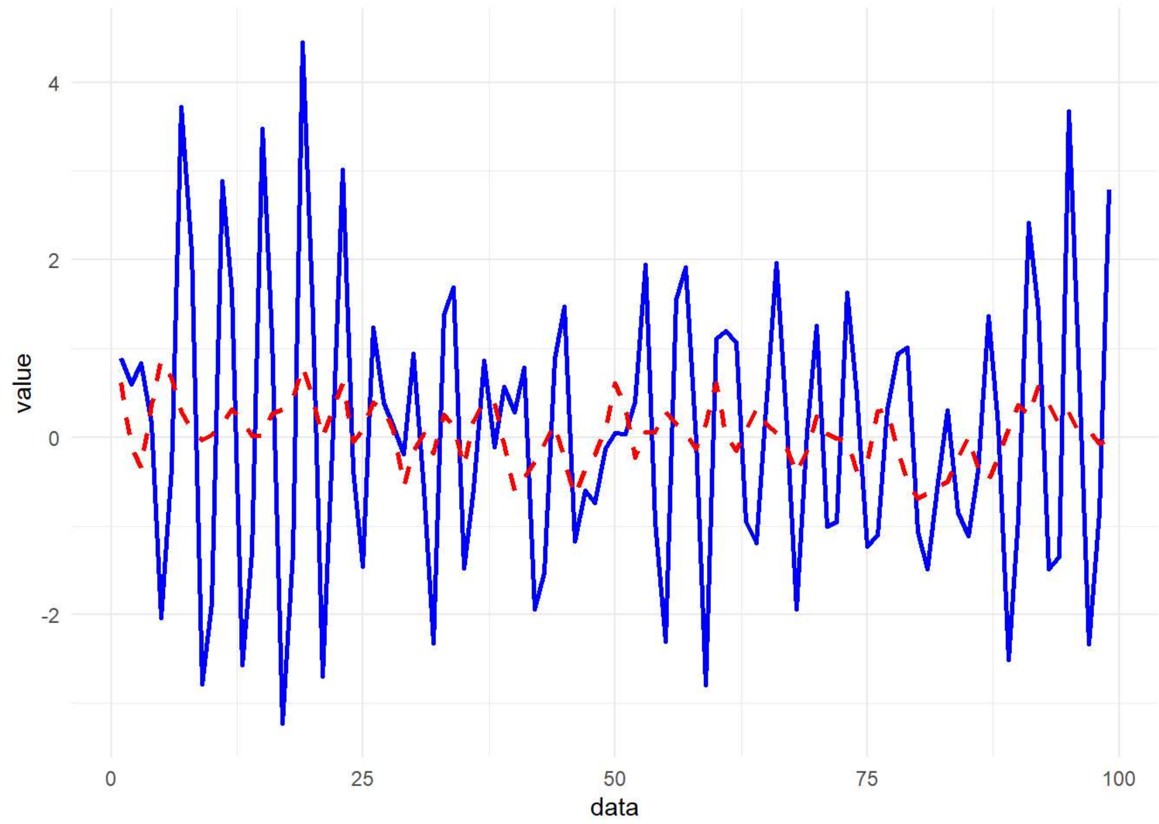
```
set.seed(4)
n <- 105
white_noise <- rnorm(n, mean = 0, sd = 1)

# AR
xt <- numeric(n)
for (t in 3:n) {
  xt[t] <- -0.9 * xt[t - 2] + white_noise[t]
}

# filter
vt <- numeric(n-6)
for (i in 1:(n-6)) {
  vt[i] <- (xt[i+2]+xt[i+3]+xt[i+4]+xt[i+5])/4

}
vt
```

```
##  [1]  0.618401491 -0.112253587 -0.348935769  0.373884270  0.858795176
##  [6]  0.670219924  0.291016101  0.082363195 -0.031853447  0.022871021
## [11]  0.163992829  0.310233183  0.183456835  0.018160429  0.019141306
## [16]  0.261855107  0.313315572  0.446870852  0.793481795  0.436302533
## [21]  0.012324508  0.320851260  0.607827051 -0.050078287  0.072424632
## [26]  0.388934475  0.314205204  0.072800569 -0.540535577 -0.146354482
## [31]  0.041534947 -0.181526590  0.253444246  0.122442692 -0.327286978
## [36]  0.183392932  0.400009438  0.379999573 -0.077581552 -0.600748224
## [41] -0.449845594 -0.277412751 -0.085126760  0.145908115 -0.260958246
## [46] -0.659536340 -0.355374452 -0.197538314  0.086967682  0.603576185
## [51]  0.354262743 -0.228903605  0.060563961  0.053772441  0.279665848
## [56]  0.156774252  0.045591213 -0.133070800  0.145693062  0.606465394
## [61]  0.031543119 -0.154310419  0.069553973  0.335518665  0.149144513
## [66]  0.048784396 -0.127253155 -0.408197828 -0.161816398  0.231798154
## [71]  0.039086014 -0.016055986 -0.054224911 -0.384232560 -0.270419640
## [76]  0.288934749  0.299446233 -0.148457205 -0.512675070 -0.689478667
## [81] -0.634776387 -0.541529885 -0.505743596 -0.239293796  0.004679003
## [86] -0.344977691 -0.481392910 -0.219118323  0.107805537  0.363160375
## [91]  0.256669268  0.569659728  0.367410818  0.155852197  0.280255062
## [96]  0.059277885  0.062928427 -0.069430692 -0.029775175
```

```
library(ggplot2)


data <- data.frame(Time = 1:(n-6), xt = xt[(3:(n-4))], vt = vt[1:(n-6)])


xt_ts <- ts(data$xt, start = 1)
vt_ts <- ts(data$vt, start = 1)


plot <- ggplot(data, aes(x = Time)) +
  geom_line(aes(y = xt), color = "blue", size = 1) +
  geom_line(aes(y = vt), color = "red", size = 1,lty = "dashed") +
  labs(x = "data", y = "value") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
print(plot)
```



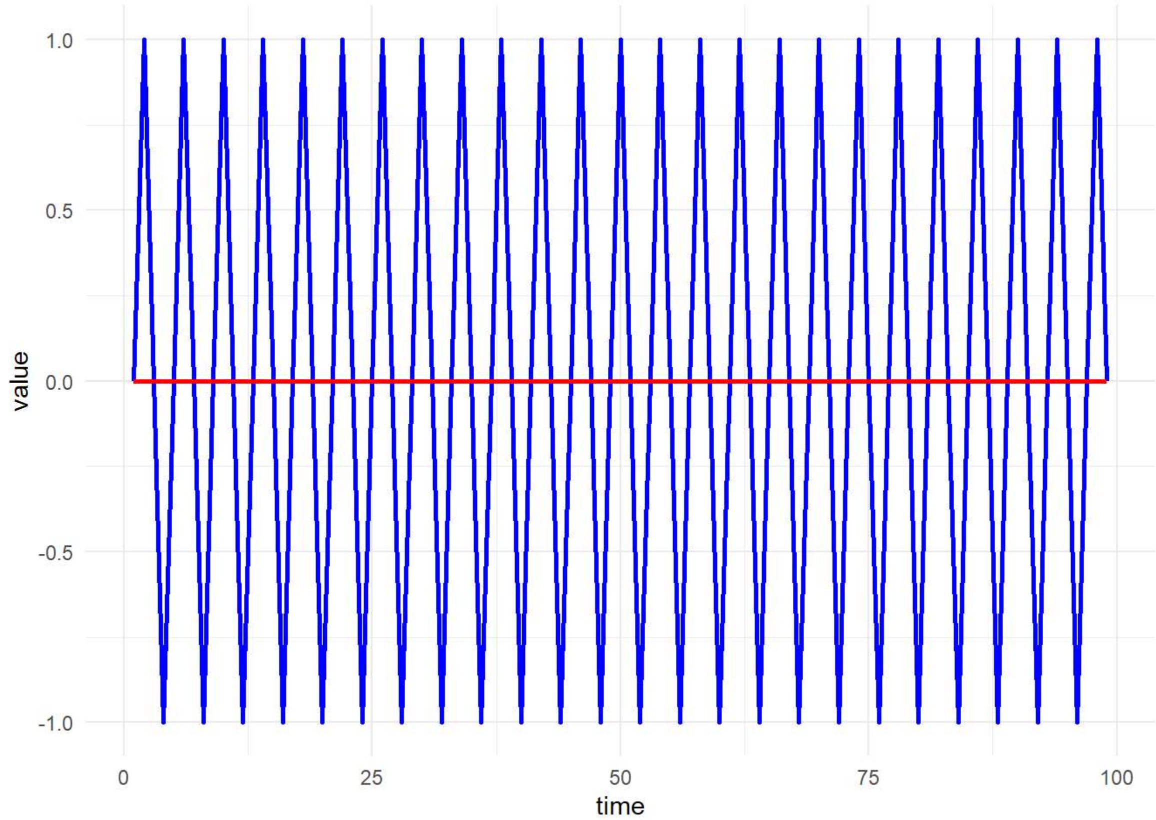
Comment: every x_t keeps -90% performance of x_{t-2} and also some noise is added. After being filtered, the the data is aggregated because it takes average among its previous points. So the variance becomes smaller and the line becomes more smooth.

(b)

```
n <- 105
white_noise <- rnorm(n, mean = 0, sd = 1)

# AR
xt <- numeric(n)
for (t in 3:n) {
  xt[t] <- cos(2*pi*t/4)
}

# filter
vt <- numeric(n-6)
for (i in 1:(n-6)) {
  vt[i] <- (xt[i+2]+xt[i+3]+xt[i+4]+xt[i+5])/4

}
vt
```

```
##  [1]  0.000000e+00 -5.164190e-17  0.000000e+00  2.674591e-17  0.000000e+00
##  [6] -5.014503e-16 -4.996004e-16 -8.557133e-16 -8.604228e-16 -9.235031e-16
## [11] -8.881784e-16 -8.499942e-16 -8.604228e-16 -9.292222e-16 -8.881784e-16
## [16] -8.442750e-16 -8.326673e-16 -9.071858e-16 -8.881784e-16 -8.385558e-16
## [21] -8.604228e-16 -9.129050e-16 -8.881784e-16 -8.328367e-16 -8.604228e-16
## [26] -9.186242e-16 -9.436896e-16 -8.826287e-16 -8.604228e-16 -9.243433e-16
## [31] -9.436896e-16 -8.769095e-16 -8.604228e-16 -9.300625e-16 -9.436896e-16
## [36]  9.051665e-16  8.881784e-16  8.405752e-16  8.604228e-16 -8.654712e-16
## [41] -8.604228e-16  8.348560e-16  8.604228e-16  2.692962e-15  2.720046e-15
## [46]  8.846480e-16  8.604228e-16 -8.540328e-16 -8.604228e-16  8.789288e-16
## [51]  8.604228e-16  2.704400e-15  2.720046e-15  8.732097e-16  8.326673e-16
## [56] -8.425945e-16 -8.604228e-16  8.674905e-16  8.604228e-16  2.688083e-15
## [61]  2.720046e-15  8.617713e-16  8.604228e-16 -8.311562e-16 -8.604228e-16
## [66]  8.560522e-16  8.604228e-16  2.699521e-15  2.664535e-15  8.503330e-16
## [71]  8.604228e-16 -8.752290e-16 -8.604228e-16  8.446138e-16  8.326673e-16
## [76]  2.683204e-15  2.664535e-15  4.391608e-15  4.413137e-15  2.688923e-15
## [81]  2.664535e-15 -2.719538e-15 -2.664535e-15 -4.410785e-15 -4.413137e-15
## [86] -2.697502e-15 -2.664535e-15  2.700361e-15  2.720046e-15  4.429962e-15
## [91]  4.413137e-15  2.706081e-15  2.720046e-15  4.424243e-15  4.385381e-15
## [96]  2.684044e-15  2.720046e-15 -2.686904e-15 -2.664535e-15
```

```
library(ggplot2)

data <- data.frame(Time = 1:(n-6), xt = xt[(3:(n-4))], vt = vt[1:(n-6)])


xt_ts <- ts(data$xt, start = 1)
vt_ts <- ts(data$vt, start = 1)


plot <- ggplot(data, aes(x = Time)) +
  geom_line(aes(y = xt), color = "blue", size = 1) +
  geom_line(aes(y = vt), color = "red", size = 1,) +
  labs(x = "time", y = "value") +
  theme_minimal()

print(plot)
```

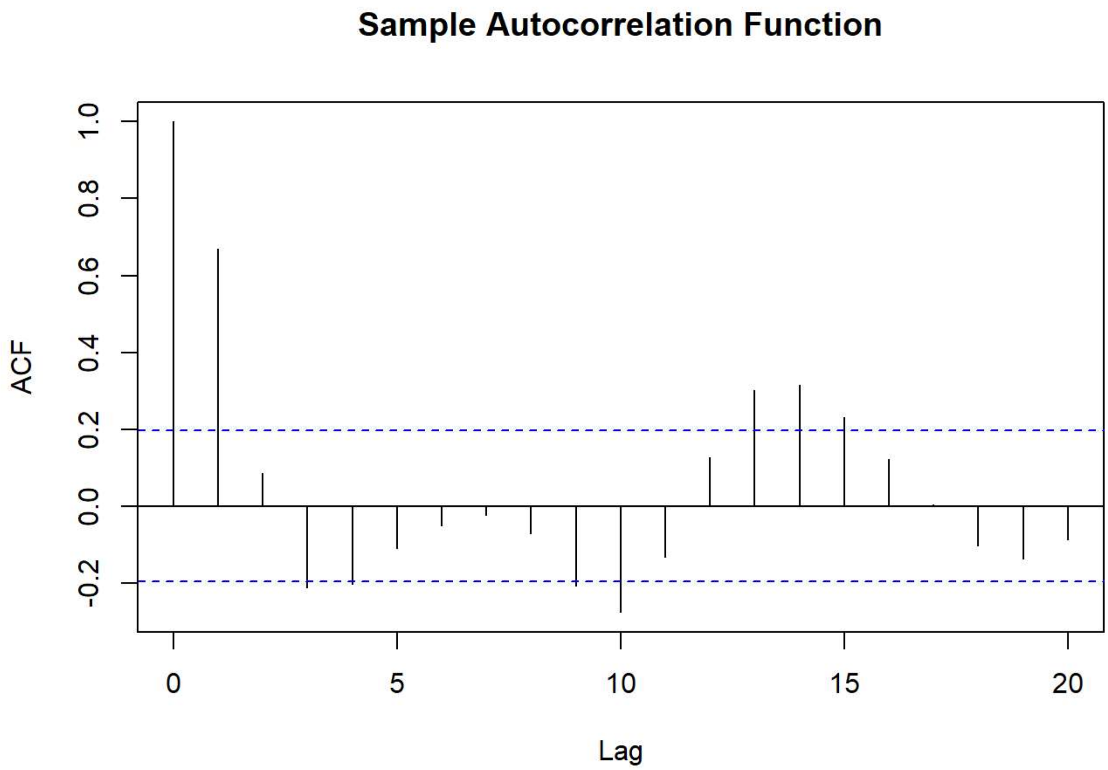


8

```r
set.seed(0)  # Set seed for reproducibility
n <- 100  # Number of observations

# Generate random values from a standard normal distribution
Wt <- rnorm(n+2)
Xt <- rep(0,n)

# Generate time series values

for(i in 1:n){
  Xt[i] <- Wt[i] + 2*Wt[i+1] + Wt[i+2]

}

# Xt <- Wt[-c(1, n)] + 2 * Wt + Wt[-c(1, n-1)]

# Compute sample autocorrelation
acf_values <- acf(Xt, plot = FALSE)
acf_values
```

```
##
## Autocorrelations of series 'Xt', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.669  0.086 -0.213 -0.203 -0.109 -0.051 -0.024 -0.070 -0.207 -0.276
##     11     12     13     14     15     16     17     18     19     20
## -0.132  0.127  0.301  0.316  0.232  0.123  0.005 -0.103 -0.137 -0.086
```

```r
# Plot the sample autocorrelation function
plot(acf_values, main = "Sample Autocorrelation Function")
```



$$E(X_t) = 0$$

$$\begin{aligned}\gamma(h) &= Cov(X_{t+h}, X_t) = E(X_{t+h}X_t) \\ &= E((W_{t+h-1} + 2W_t + W_{t+h+1})(W_{t-1} + 2W_t + W_{t+1})) \\ &= \begin{cases} 6, & if\ h = 0 \\ 4, & if\ h = 1 \\ 4, & if\ h = -1 \\ 1, & if\ h = 2 \\ 1, & if\ h = -2 \\ 0, & else \end{cases}\end{aligned}$$

which implies:

$\rho(h) = \frac{\gamma(h)}{\gamma(0)} =$

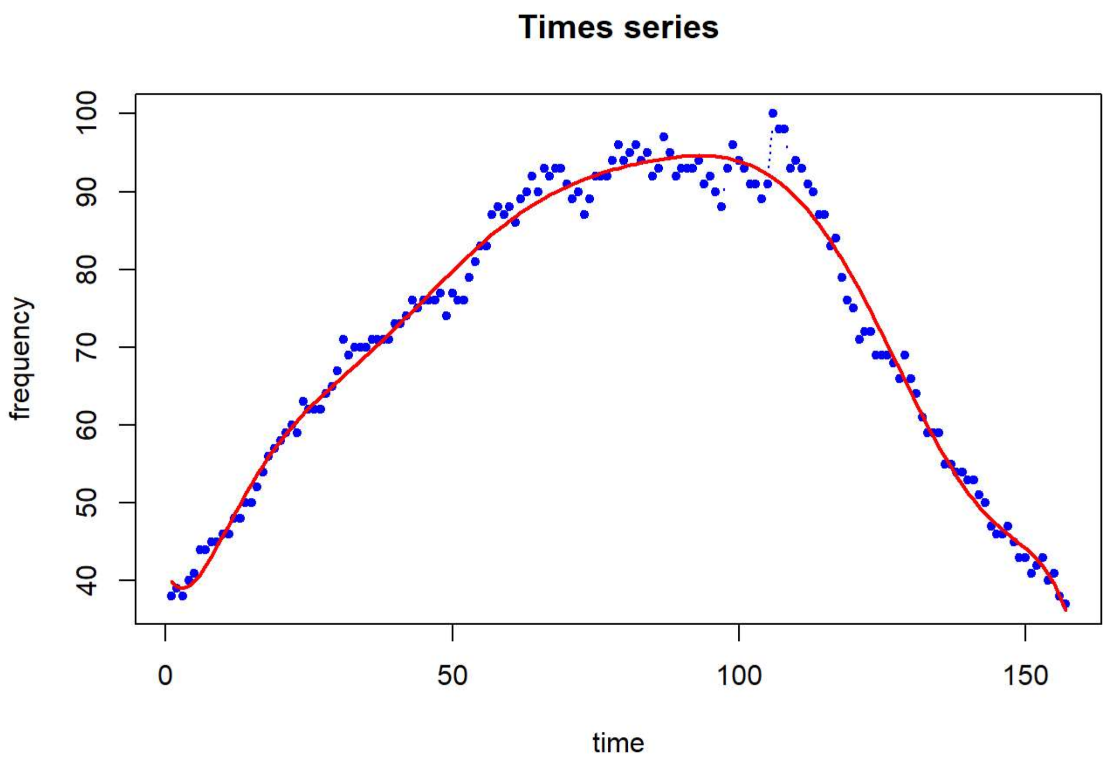
$$\begin{cases} 1, & if\ h = 0 \\ 2/3, & if\ h = 1 \\ 2/3, & if\ h = -1 \\ 1/6, & if\ h = 2 \\ 1/6, & if\ h = -2 \\ 0, & else \end{cases}$$

Conspicuously, the figure is consistent with the theoretical result.

9

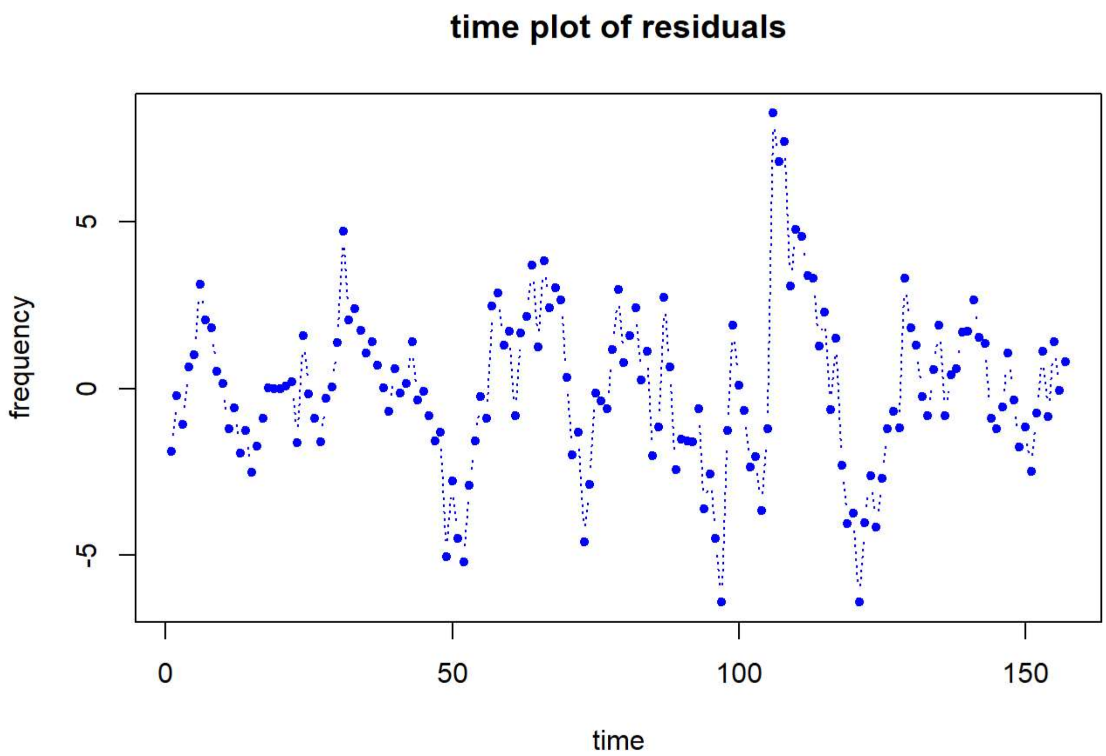
(a)

```r
data = read.csv("C:/Users/Lenovo/Desktop/yahoo(1).csv")

y <- data$X
y <- y[-c(1,2)]
x <- 1:157
df <- data.frame(y,x)

degree <- 10

model <- lm(y ~ poly(x, degree), data = df)


x_seq <- seq(min(df$x), max(df$x), length.out = 157)

# predict y
y_pred <- predict(model, newdata = data.frame(x = x_seq))

plot(df$y~df$x,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "Times series",pch = 20,lty = 3)


  lines(x_seq,y_pred,type = "l",col = "red",lty = 1,lwd = 2)
```



We use tenth-order polynomial to estimate the data points, the effect of estimation is not bad visually.

```
# time plot and ACF

e <- rep(0,157)
y <- as.numeric(y)
y2 <- as.numeric(y_pred)
e <- y-y2


# time plot of residuals

plot(e~x,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "time plot of residuals",pch = 20,lty = 3)
```

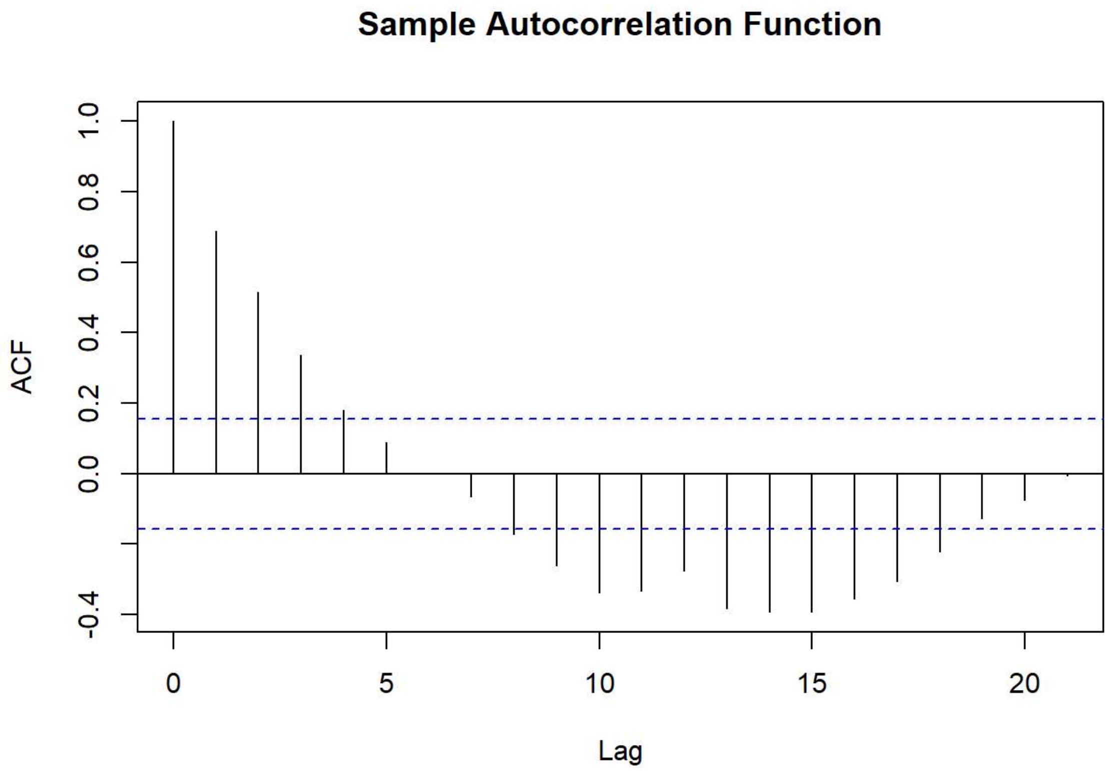


From the time plot of residuals, we can maintain that the mean of residuals is approximately zero, which means the trend pattern has been estimated by this model. However,on the other hand, the variance of residuals is a little bit large, thus the accuracy and effect of this model need to be advanced.

```
# Compute sample autocorrelation
acf_values <- acf(e, plot = FALSE)
acf_values
```

```
##
## Autocorrelations of series 'e', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.689  0.514  0.335  0.181  0.088  0.001 -0.065 -0.171 -0.260 -0.339
##     11     12     13     14     15     16     17     18     19     20     21
## -0.332 -0.275 -0.383 -0.392 -0.393 -0.355 -0.306 -0.220 -0.127 -0.075 -0.005
```

```
# Plot the sample autocorrelation function
plot(acf_values, main = "Sample Autocorrelation Function")
```



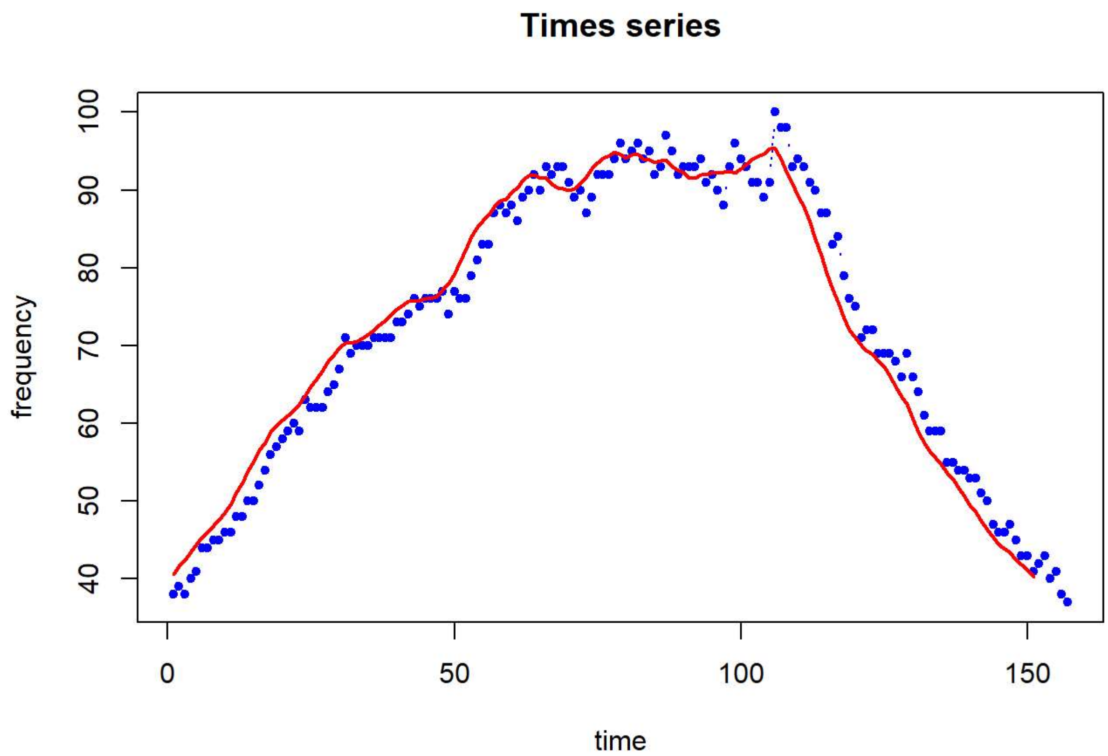
According to the correlogram, ACF does NOT stay quite near the x-axis, which means the residuals can NOT be a white noise sequence.

(b)

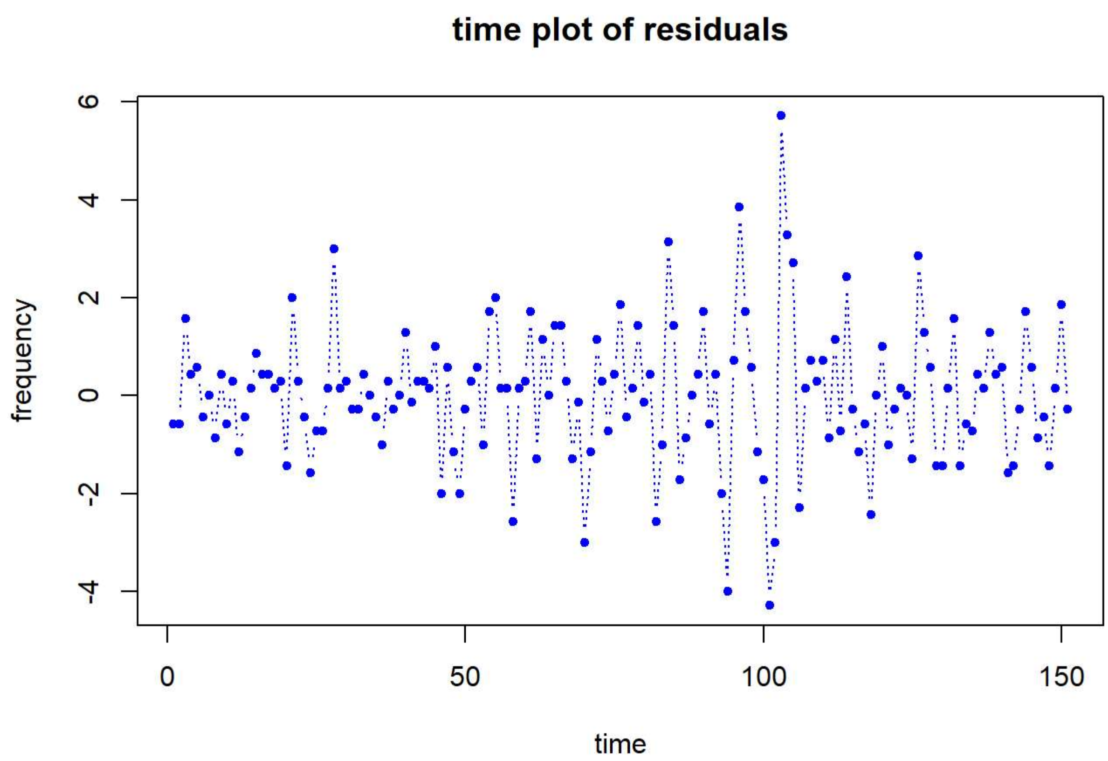
```
q <- 3
```

We set the parameter $q = 3$ because if it is too small, then the variance will become large whereas the bias is relatively small; on the other hand, if it is too large, then the bias will become large though the variance tends to be small.

```
total = 157-2*q
x_alter <- 1:total
result <- rep(0,total)

for (k in 1:total ) {
  result[k] <- sum(y[k:(k+2*q)])/(1+2*q)

}


plot(df$y~df$x,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "Times series",pch = 20,lty = 3)

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)
```




```
error <- y[(q+1):(157-q)] - result


# time plot of residuals

plot(error~x_alter,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "time plot of residuals",pch = 20,lty = 3)
```

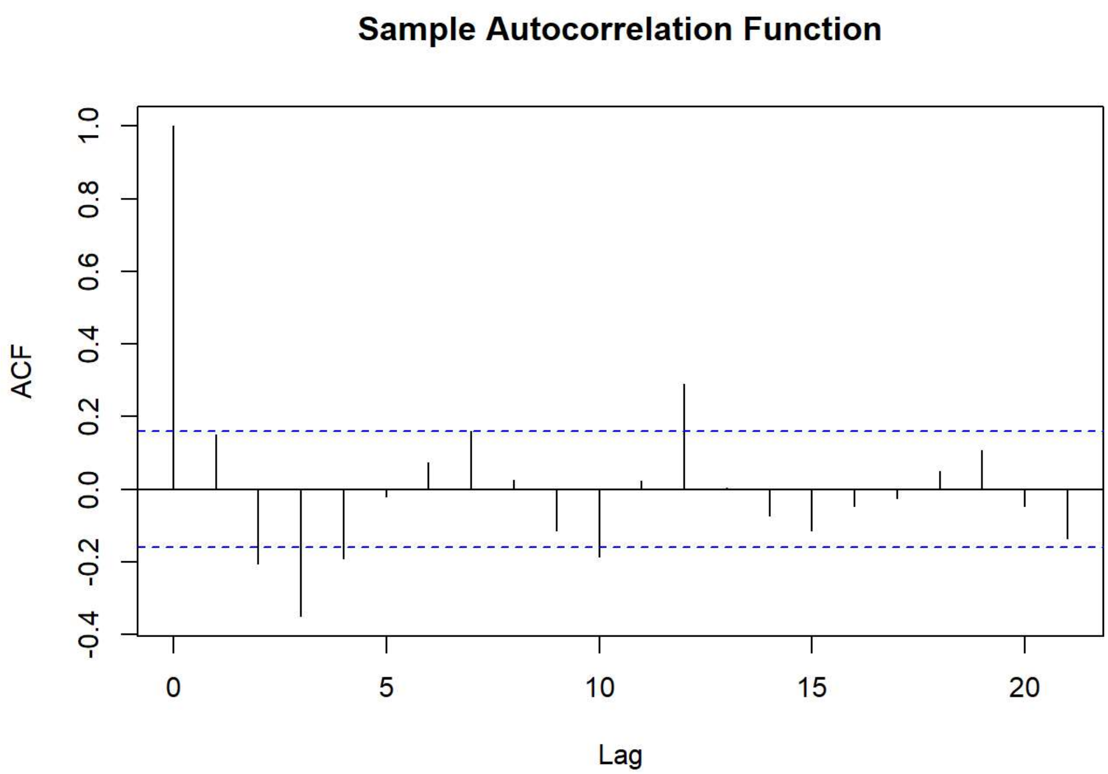


From the time plot of residuals, we can state that the mean of the residuals tends to be zero, so the trend pattern of data has been estimated. The variance, on the other hand, is smaller than the previous model but is still a little large, which means the accuracy and effect is advanced relative to the previous model but can be enhanced further.

```
acf_values <- acf(error, plot = FALSE)
acf_values
```

```
##
## Autocorrelations of series 'error', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.149 -0.204 -0.350 -0.190 -0.021  0.073  0.161  0.026 -0.115 -0.187
##     11     12     13     14     15     16     17     18     19     20     21
##  0.022  0.289  0.004 -0.074 -0.114 -0.046 -0.024  0.050  0.107 -0.047 -0.136
```

```
# Plot the sample autocorrelation function
plot(acf_values, main = "Sample Autocorrelation Function")
```

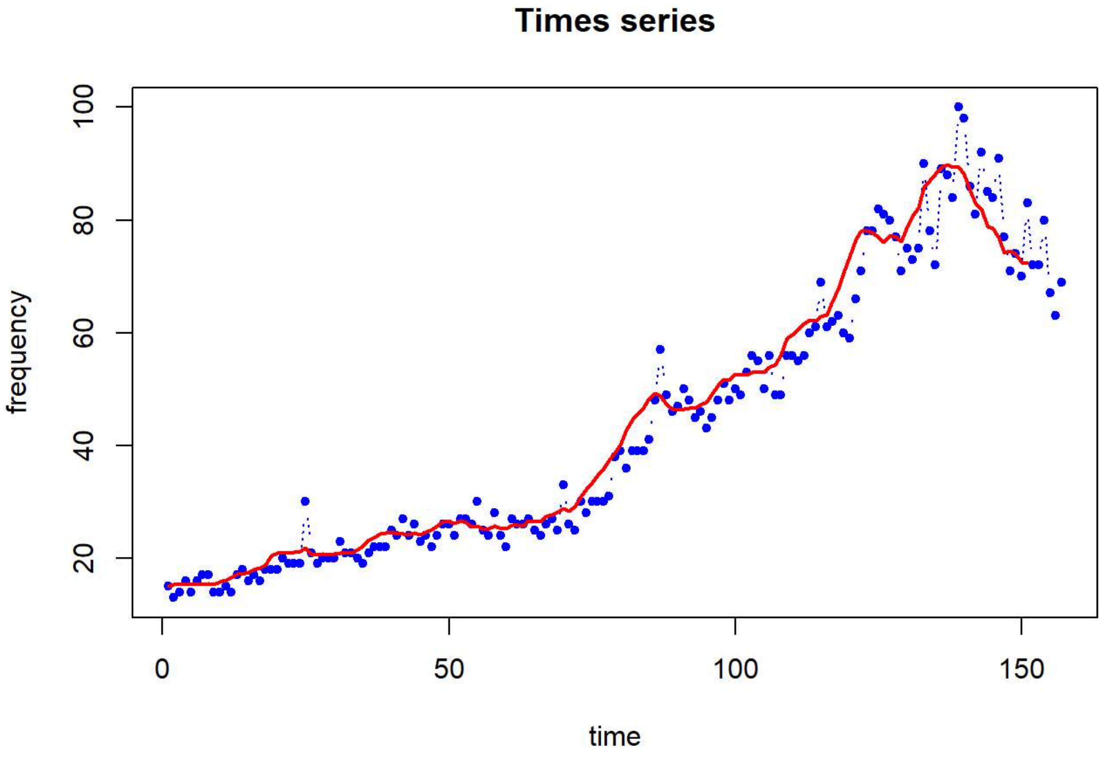


According to the correlogram, ACF is quite close to zero among nearly all the point (except at h = 0) ,we can find that the residuals can be a white noise sequence approximately. The effect is nice.

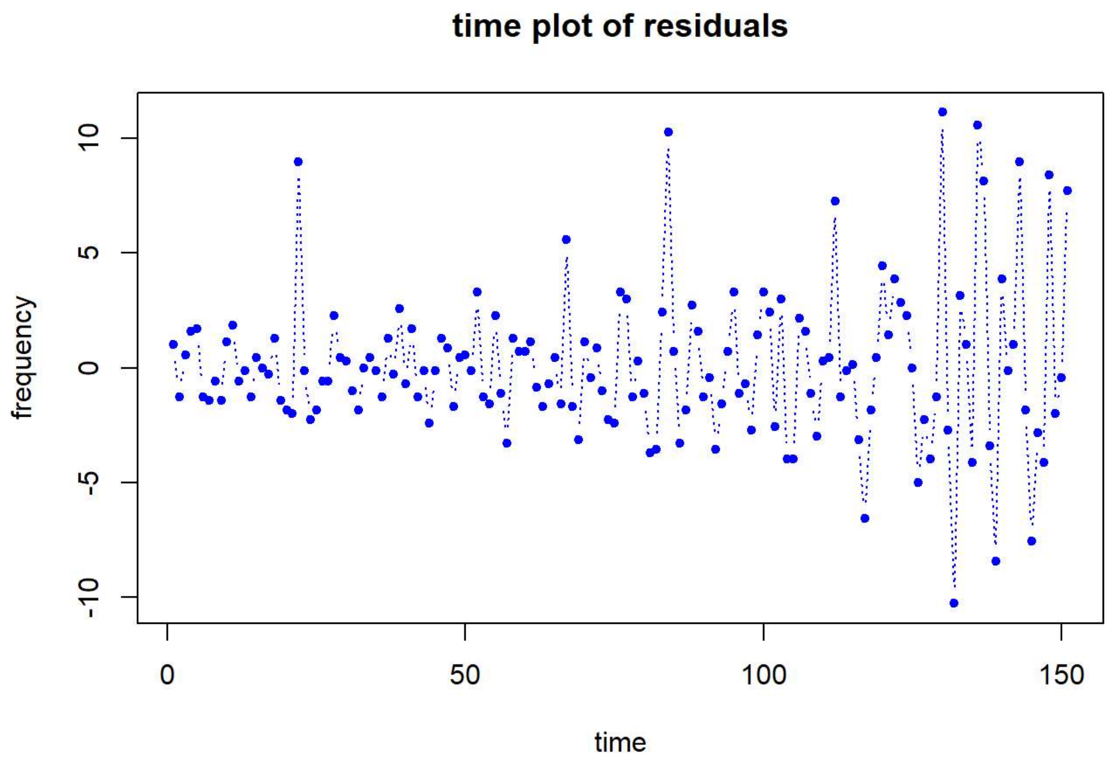
10

(a)

```
data <- read.csv("C:/Users/Lenovo/Desktop/chipotle(1).csv")
y <- data$X
y <- y[-c(1,2)]
x <- 1:157
x <- as.numeric(x)
y <- as.numeric(y)


q <- 3
total = 157-2*q
x_alter <- 1:total
result <- rep(0,total)

for (k in 1:total ) {
  result[k] <- sum(y[k:(k+2*q)])/(1+2*q)

}


plot(y~x,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "Times series",pch = 20,lty = 3)

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)
```




```
error <- y[(q+1):(157-q)] - result

plot(error~x_alter,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "time plot of residuals",pch = 20,lty = 3)
```

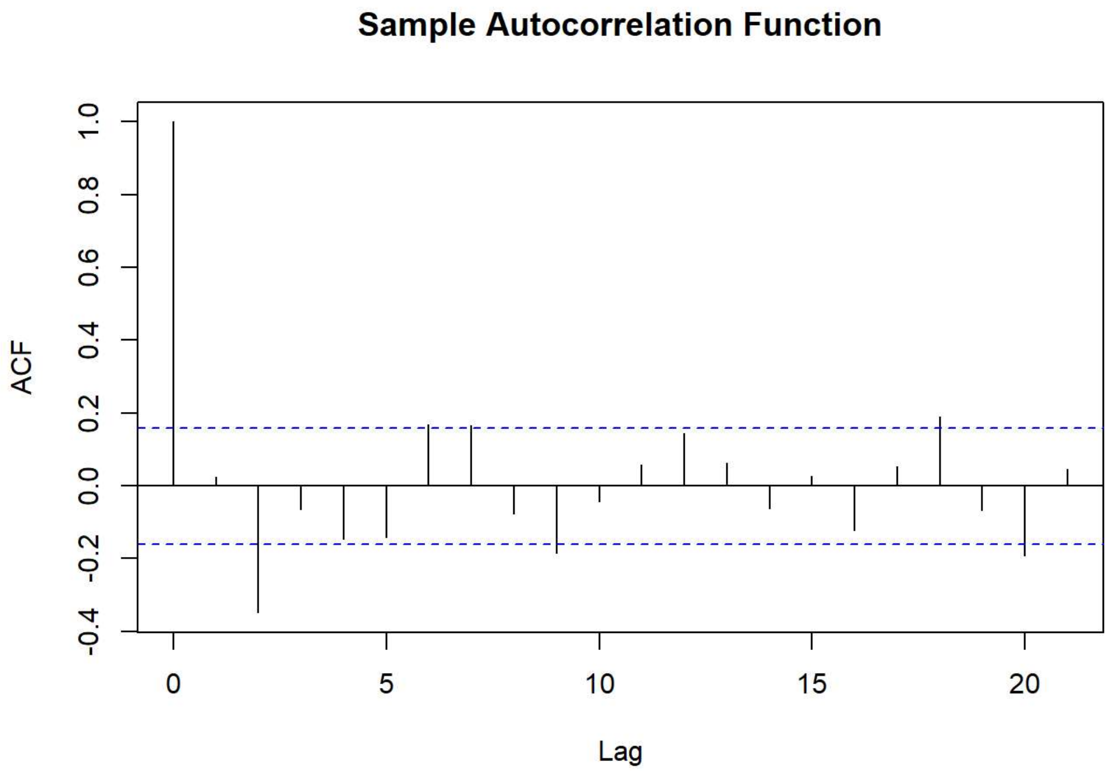


From the time plot of residuals, we can figure out that the mean of residuals is approximately zero, which means the estimation keeps the trend pattern of the data. Also the variance is relatively acceptable, which implies the accuracy and the effect of the model is trustable.

```
acf_values <- acf(error, plot = FALSE)
acf_values
```

```
##
## Autocorrelations of series 'error', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.024 -0.348 -0.065 -0.146 -0.142  0.168  0.165 -0.077 -0.185 -0.044
##     11     12     13     14     15     16     17     18     19     20     21
##  0.057  0.145  0.062 -0.061  0.027 -0.121  0.053  0.189 -0.066 -0.193  0.046
```

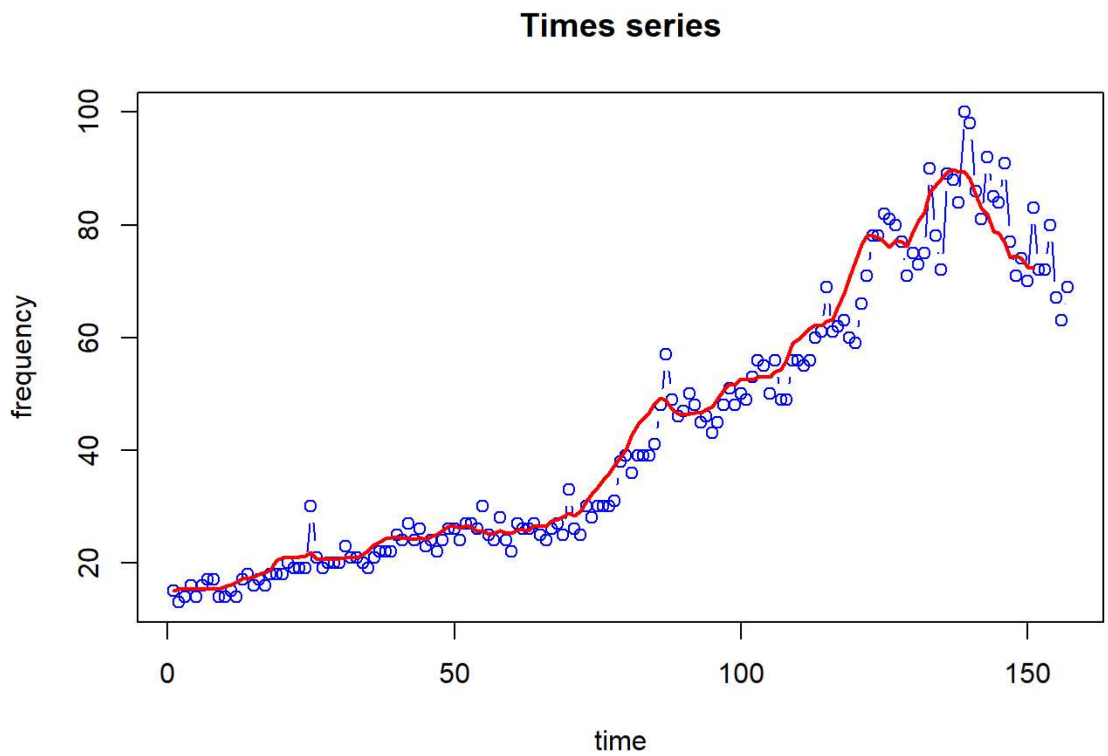
```
# Plot the sample autocorrelation function
plot(acf_values, main = "Sample Autocorrelation Function")
```



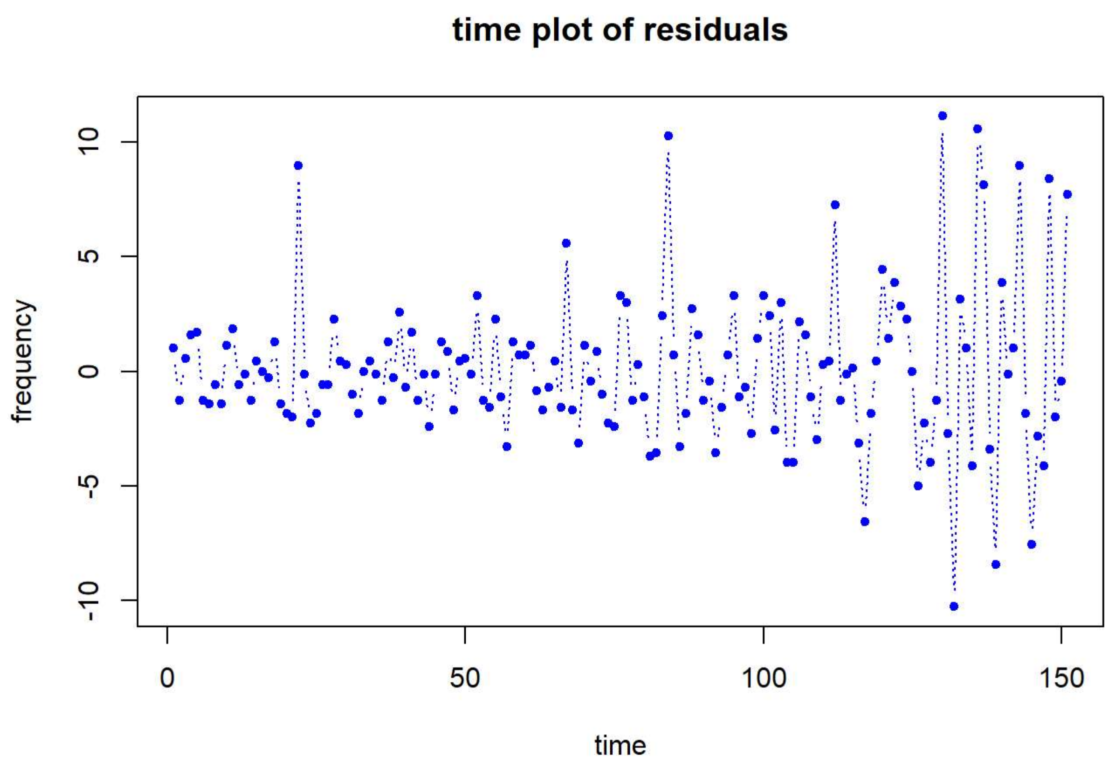
We can see from the autocorrelation figure that $\gamma(h)$ of most of the points except 0 is quite close to zero, which means the residuals can be considered as a white noise sequence.

(b)

```
q <- 3
total = 157-2*q
x_alter <- 1:total
result <- rep(0,total)

for (k in 1:total ) {
  result[k] <- sum(y[k:(k+2*q)])/(1+2*q)

}

plot(y~x,type = "b",xlab = "time", ylab = "frequency",
     col = "blue",main = "Times series")

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)
```



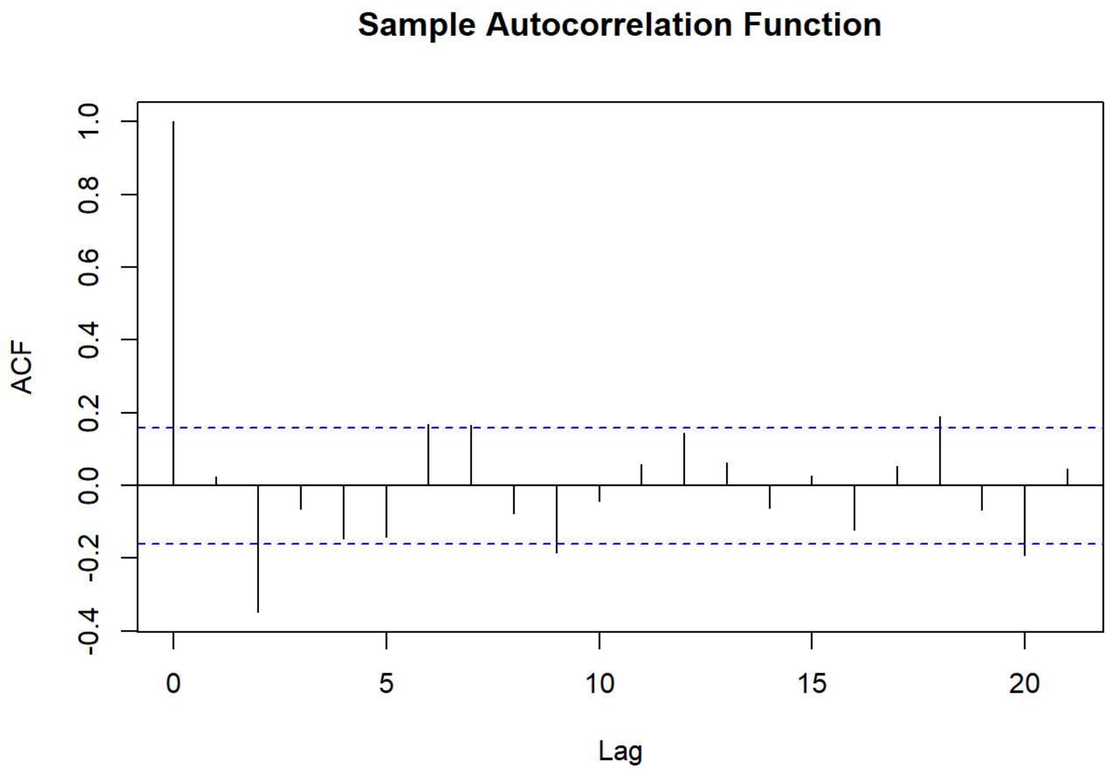

```
error <- y[(q+1):(157-q)] - result

plot(error~x_alter,type = "b",xlab = "time",ylab = "frequency",
     col = "blue",main = "time plot of residuals",pch = 20,lty = 3)
```




```
acf_values <- acf(error, plot = FALSE)
acf_values
```

```
##
## Autocorrelations of series 'error', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.024 -0.348 -0.065 -0.146 -0.142  0.168  0.165 -0.077 -0.185 -0.044
##     11     12     13     14     15     16     17     18     19     20     21
##  0.057  0.145  0.062 -0.061  0.027 -0.121  0.053  0.189 -0.066 -0.193  0.046
```

```
# Plot the sample autocorrelation function
plot(acf_values, main = "Sample Autocorrelation Function")
```




```
average <- mean(error)
std <- sqrt(var(error))
z <- y
count_2sigma <- 0
count_3sigma <- 0

for(m in 1:(total-1)){
  if((error[m] - average)^2 > (2*std)^2 ){
    z[q+m] <- result[m]
  }
  if((error[m] - average)^2 > (2*std)^2 ){
    count_2sigma <- count_2sigma + 1
  }
  if((error[m] - average)^2 > (3*std)^2 ){
    count_3sigma <- count_3sigma + 1
  }

}
z - y
```

```
##   [1]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##   [7]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [13]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [19]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [25]  -9.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [31]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [37]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [43]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [49]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [55]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [61]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [67]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [73]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [79]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [85]   0.000000   0.000000 -10.285714   0.000000   0.000000   0.000000
##  [91]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
##  [97]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
## [103]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
## [109]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
## [115]  -7.285714   0.000000   0.000000   0.000000   0.000000   6.571429
## [121]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
## [127]   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
## [133] -11.142857   0.000000  10.285714   0.000000   0.000000   0.000000
## [139] -10.571429  -8.142857   0.000000   8.428571   0.000000   0.000000
## [145]   0.000000  -9.000000   0.000000   7.571429   0.000000   0.000000
## [151]  -8.428571   0.000000   0.000000   0.000000   0.000000   0.000000
## [157]   0.000000
```

```
count_2sigma
```

```
## [1] 12
```

```
count_3sigma
```

```
## [1] 4
```

We use 2σ principle to judge if a data is a regular point, which means a point is irregular if the residual at which falls outside $[\mu - 2\sigma, \mu + 2\sigma]$ (where μ stands for the mean of all the residuals and σ is the standard error of all the residuals).

If one point is judged as irregular, then we replace it with the average of $(2q + 1)$ points nearby to reduce its irregularity. (Here q is the parameter we set previously.)

From the last table above, we can see explicitly where and how much we change the data to smooth them. Moreover, there are 12 residual points greater than 2σ and 4 residual points greater than 3σ.

Method 1: replacing the irregular points by their averages

nearby.

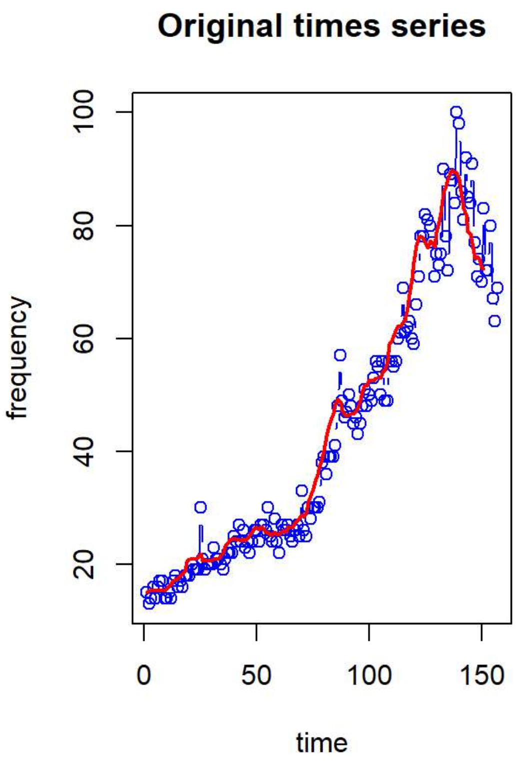
```
 par(mfrow = c(1,2))



error <- z[(q+1):(157-q)] - result


average <- mean(error)
std <- sqrt(var(error))

count_2sigma <- 0
count_3sigma <- 0

for(m in 1:(total-1)){

  if((error[m] - average)^2 > (2*std)^2 ){
    count_2sigma <- count_2sigma + 1
  }
  if((error[m] - average)^2 > (3*std)^2 ){
    count_3sigma <- count_3sigma + 1
  }

}

count_2sigma
```

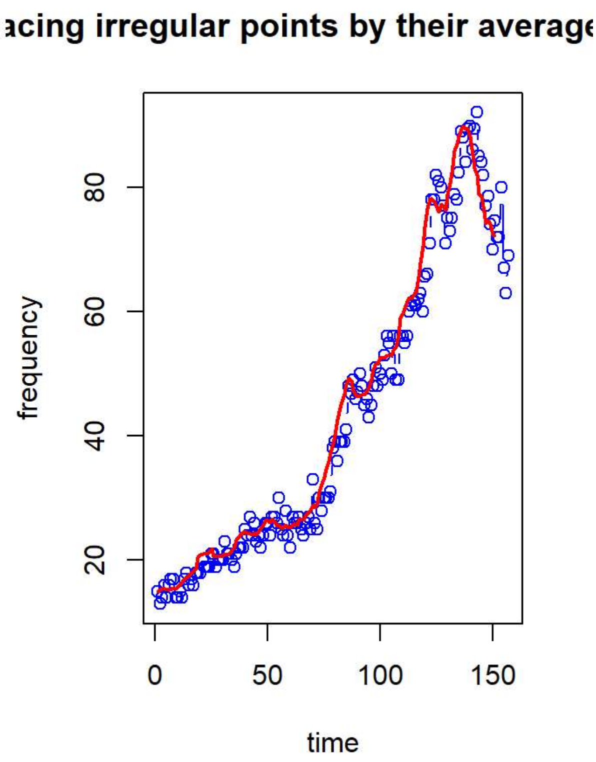
```
## [1] 3
```

```
count_3sigma
```

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

```
plot(y~x,type = "b",xlab = "time", ylab = "frequency",
     col = "blue",main = "Original times series")

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)

plot(z~x,type = "b",xlab = "time", ylab = "frequency",
     col = "blue",main = "Replacing irregular points by their averages nearby")

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)
```





After replacing irregular points by their averages nearby, we can figure out that the data is smoother than before and literally, there are only 3 residual points greater than 2σ and 0 residual points greater than 3σ.

Method 2: taking log transformation.

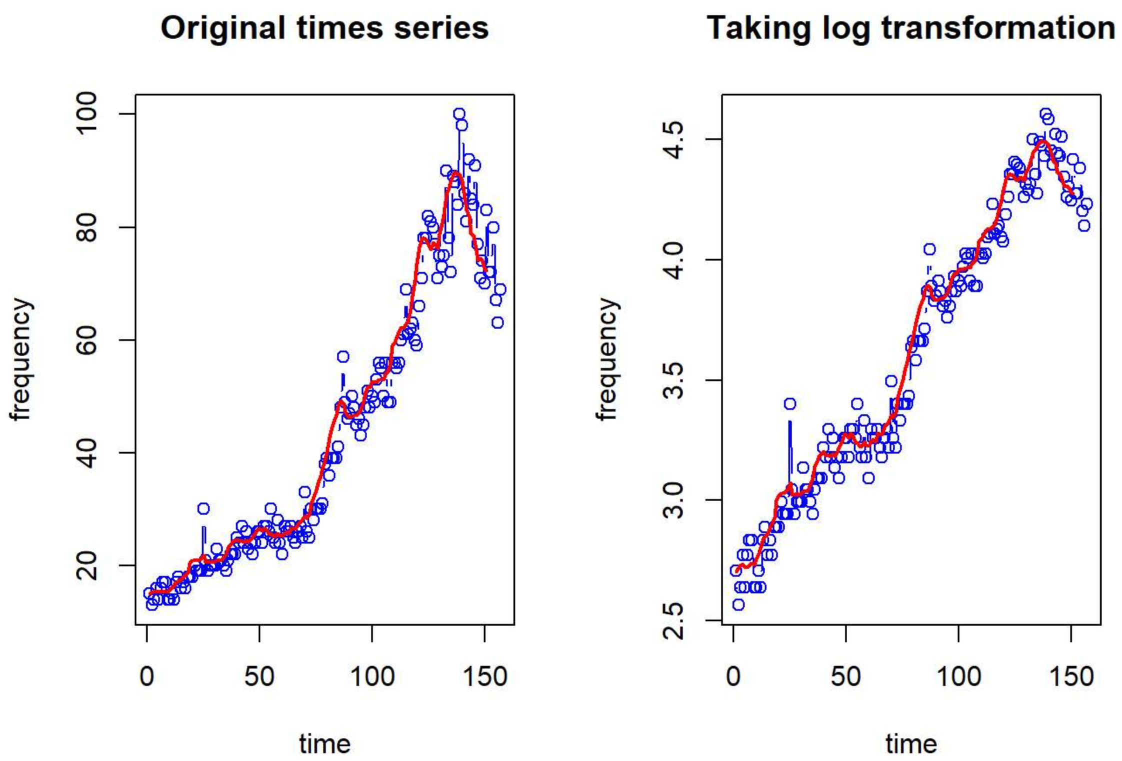
```
z2 <- log(y)

q <- 3
total = 157-2*q
x_alter <- 1:total
result2 <- rep(0,total)

for (k in 1:total ) {
  result2[k] <- sum(z2[k:(k+2*q)])/(1+2*q)

}




error <- z2[(q+1):(157-q)] - result2


average <- mean(error)
std <- sqrt(var(error))

count_2sigma <- 0
count_3sigma <- 0

for(m in 1:(total-1)){

  if((error[m] - average)^2 > (2*std)^2 ){
    count_2sigma <- count_2sigma + 1
  }
  if((error[m] - average)^2 > (3*std)^2 ){
    count_3sigma <- count_3sigma + 1
  }

}

count_2sigma
```

```
## [1] 3
```

```
count_3sigma
```

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

```
 par(mfrow = c(1,2))

plot(y~x,type = "b",xlab = "time", ylab = "frequency",
     col = "blue",main = "Original times series")

lines(x_alter,result,type = "l",col = "red",lty = 1,lwd = 2)

plot(z2~x,type = "b",xlab = "time", ylab = "frequency",
     col = "blue",main = "Taking log transformation")

lines(x_alter,result2,type = "l",col = "red",lty = 1,lwd = 2)
```



After taking log transformation, we can figure out that the data is smoother than before and literally, there are only 3 residual points greater than 2σ and 1 residual points greater than 3σ.

To conclude, we can smooth the data by replacing the irregular points with their averages in their neighbor or taking log transformation.