Lecture 20: Correlated Errors

Ailin Zhang

2023-07-04

Agenda

- Definition of Correlated Errors (Today)
- Detection of Correlated Errors (Today)
  - Time plot of residuals
  - Runs test
  - Durbin-Watson test
  - Lag plots
  - Autocorrelation function and autocorrelation plot
- Remedies to Correct for Autocorrelated Errors
- Autocorrelation and Seasonality

Correlated Errors

- Recall in MLR Models,

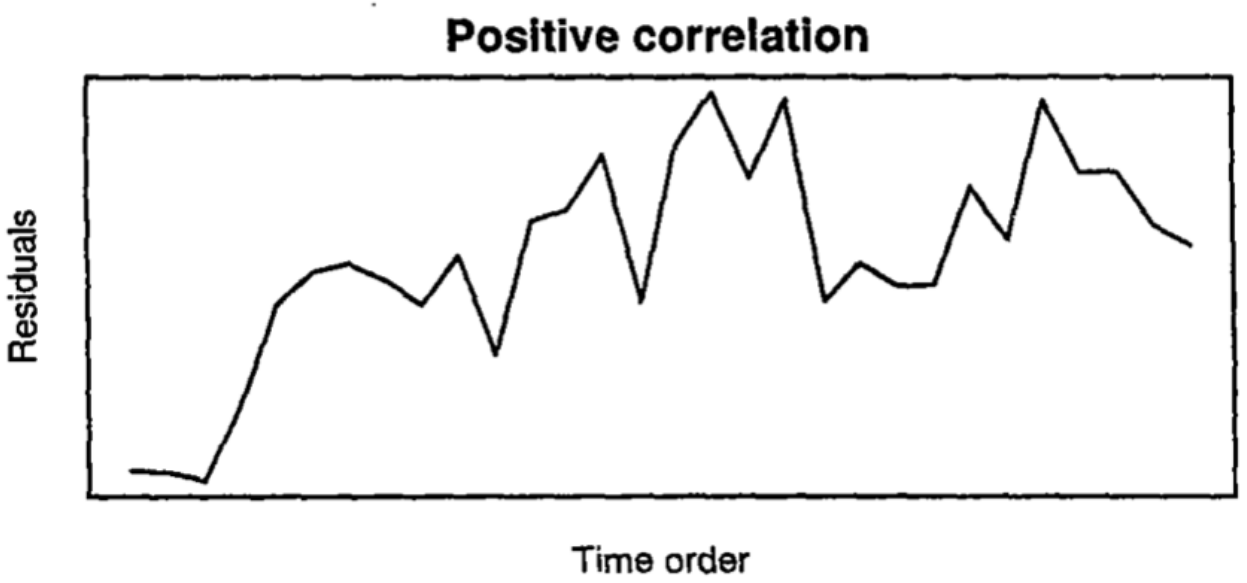
$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i$$

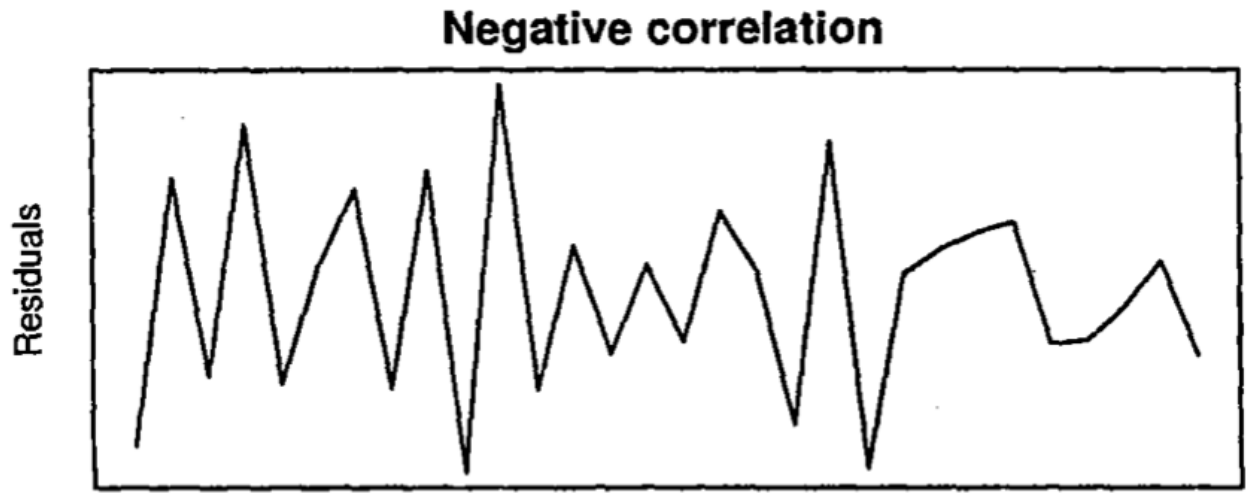
  the errors ϵ_i are assumed to be independent.

- If the data are collected sequentially in time/space, then the errors may be correlated

- Correlated errors can arise when observations have a spatial or temporal order, e.g.,
  - Temporal: In sports, a player may exhibit hot or cold streaks in which he performs above or below expectation for several games.
  - Spatial: In agriculture studies, adjacent plots of land tend to be similar (soil, humidity, sun exposure)

Plots of residuals against time





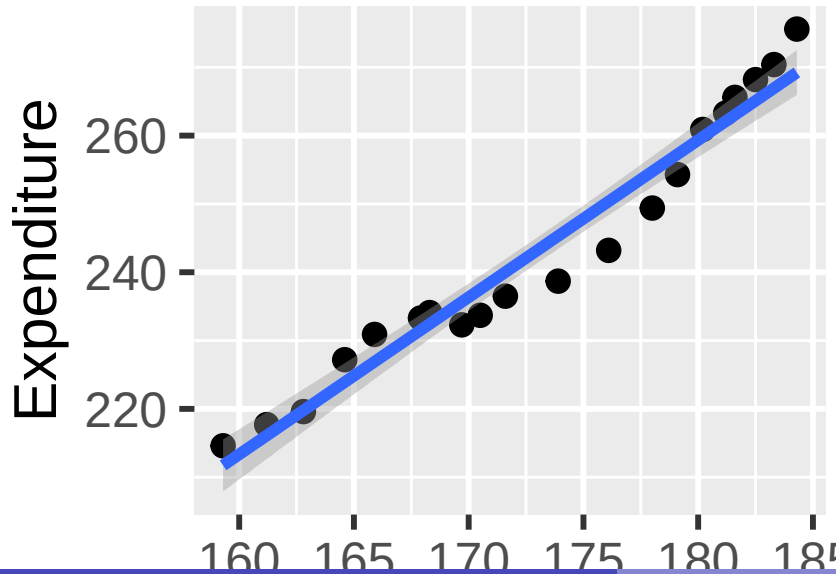
Autocorrelation

When the observations have a natural sequential order, the correlation is referred to as **autocorrelation**.

Example: Quarterly data from 1952 to 1956 on consumer expenditure (Y = Expenditure) and the stock of money (X = Stock), both in millions of current US dollars.

```
stock = read.table("stock.txt", h=T)
library(ggplot2)
ggplot(stock, aes(x=Stock, y=Expenditure)) +
geom_point() + geom_smooth(method='lm')
```

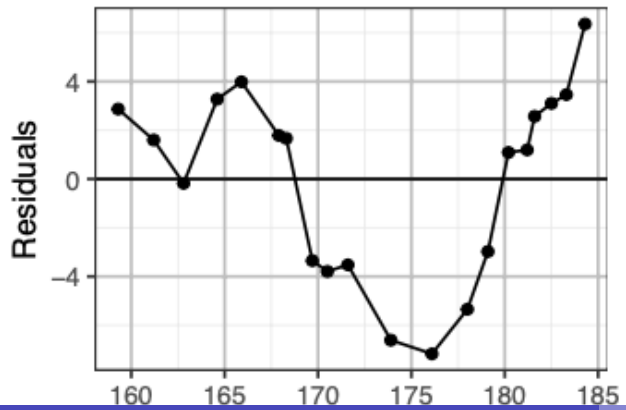
```
## `geom_smooth()` using formula 'y ~ x'
```

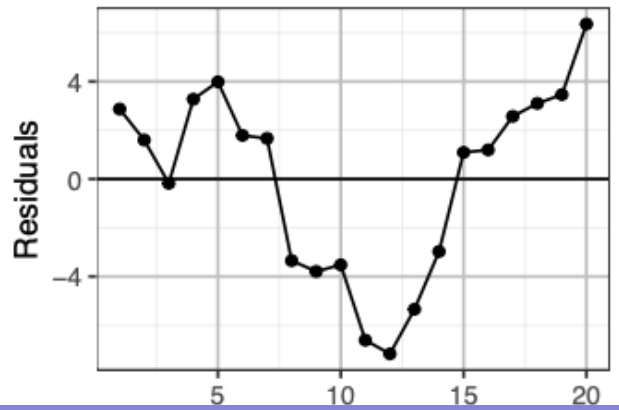


Residual Plot

Since the observations are ordered in time, we should also plot the residuals against time (index).

```
lm1 = lm(Expenditure ~ Stock, data=stock)
ggplot(stock, aes(x=Stock, y=lm1$res)) +
  geom_point() + geom_line() +
  ylab("Residuals")+ geom_hline(yintercept=0)
ggplot(stock, aes(x=1:20, y=lm1$res)) +
  geom_point() + geom_line() + xlab("Index") +
  ylab("Residuals")+ geom_hline(yintercept=0)
```

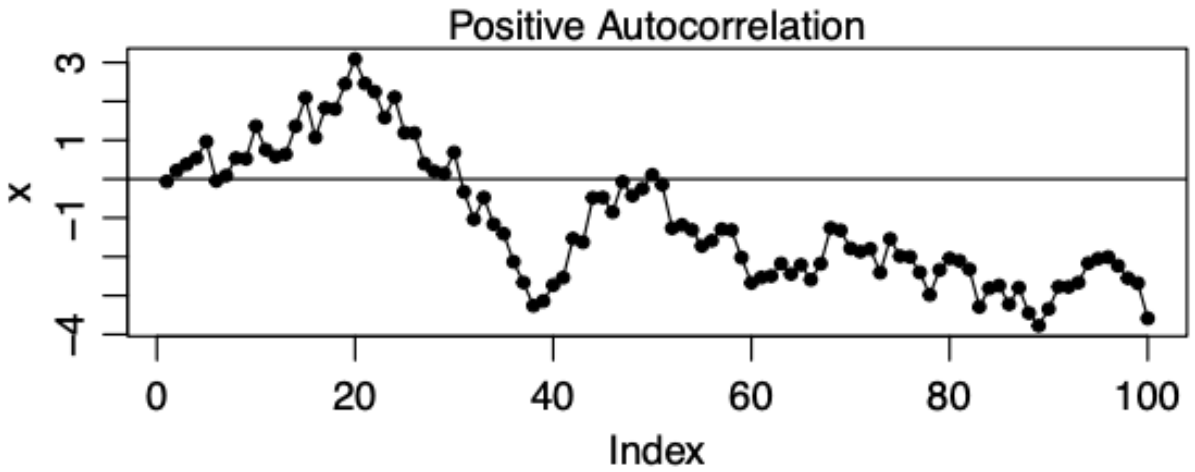




Interpreting Time Plot/Index Plot of Residuals

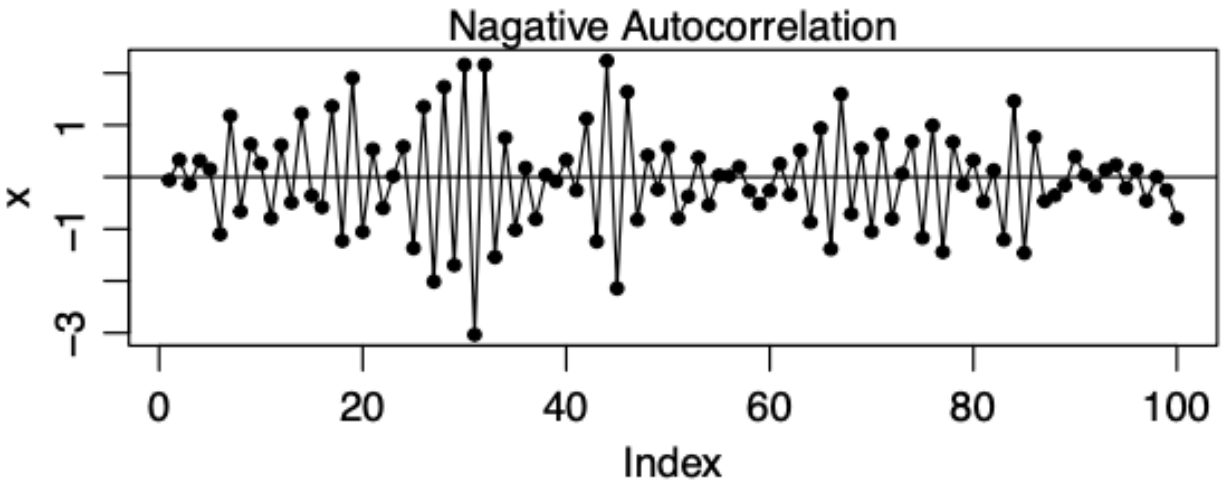
A time plot or an index plot of the residuals is a plot of residuals v.s. the (time) order they are recorded. Points in a time-plot are connected by a line.

- A **smooth** time-plot is a sign of **positive** autocorrelation, since a smooth time plot means successive residuals are close together

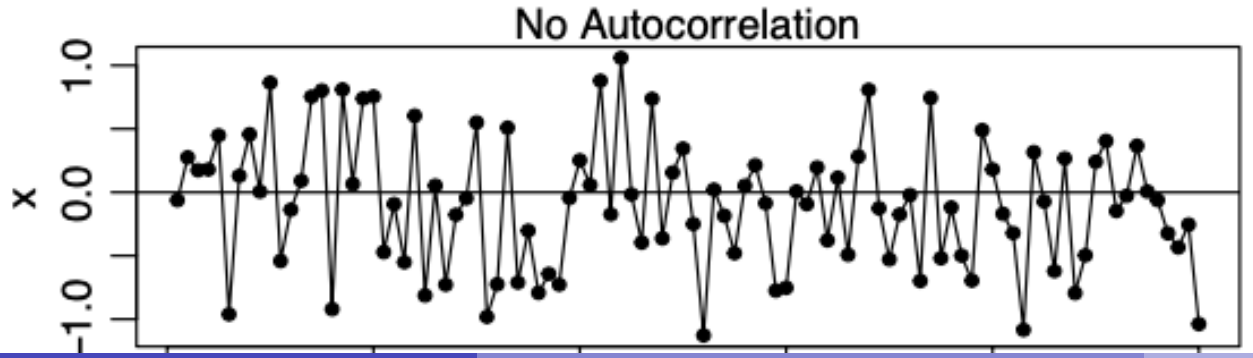


Interpreting Time Plot/Index Plot of Residuals

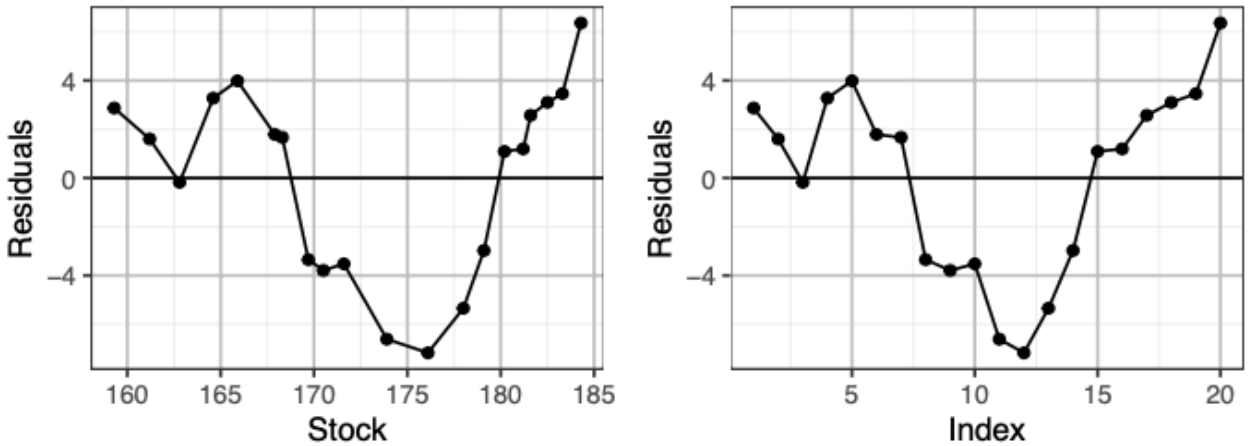
- Time plot of data w/ **negative** autocorrelation tend to alternate regularly between positive and negative values.



- If no autocorrelation, the time plot has more up-and-downs.

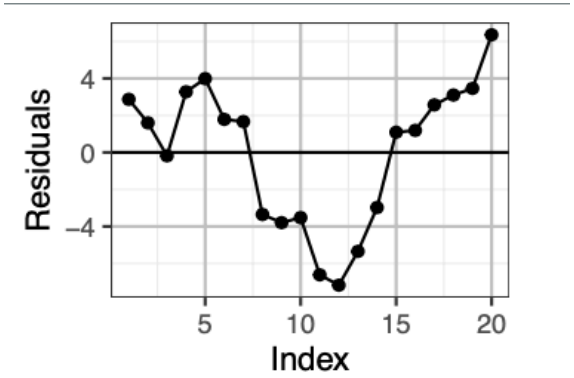


For the Stock Data, what is the sign of autocorrelation?



- **Positive Autocorrelation!**

Runs Test



- The runs test (Bradley, 1968) can be used to decide if a data set is from a random process.
- The first step in the runs test is to count the number of runs in the data sequence.
- A group of successive positive residuals or negative residuals are called as a run.
- For the above data there are 5 separate runs:

$$+\ +\ -\ +\ +\ +\ +\ -\ -\ -\ -\ -\ -\ +\ +\ +\ +\ +\ +$$

Distribution of Runs

Assuming independence, with n_1 positive and n_2 negative residuals, the expected number μ and variance σ^2 of runs are

- $\mu = \dfrac{2n_1n_2}{n_1 + n_2} + 1$
- $\sigma^2 = \dfrac{2n_1n_2(2n_1n_2 - n_1 - n_2)}{(n_1 + n_2)^2(n_1 + n_2 - 1)}$

In the above example, $n_1 = 12$. $n_2 = 8$, therefore $\mu = 10.6$, $\sigma = 2.08$

Z-test of runs

- Normal Approximation:

  If n_1 and n_2 are large (say ≥ 10), number of Runs is approx. $\sim N(\mu, \sigma^2)$

- then we can use an approximate z-statistic

$$z = \frac{\text{Number of Runs} - \mu}{\sigma} \sim \text{approx. } N(0,1)$$

- For our example,

$$z = \frac{5 - 10.6}{2.0845} \approx -2.6865$$

  The two-sided P-value is `2*pnorm(-2.686)` =`0.0072`

  If $\alpha = 0.05$, shall you conclude that the error is correlated?

Run test in R

The command `runs.test()` in the `tseries` library can perform the runs test. You need to first install the `tseries`.

```
library(tseries)
runs.test(factor(lm1$res > 0)) # two-sided by default
```

```
##
##  Runs Test
##
## data:  factor(lm1$res > 0)
## Standard Normal = -2.6865, p-value = 0.007221
## alternative hypothesis: two.sided
```

For testing **positive** autocorrelation, use `alternative = "less"` as positive autocorrelation leads to fewer runs.

```
runs.test(factor(lm1$res > 0), alternative = "less")
```

```
##
##  Runs Test
##
## data:  factor(lm1$res > 0)
## Standard Normal = -2.6865, p-value = 0.003611
## alternative hypothesis: less
```

For testing negative autocorrelation, use alternative = "greater" as negative autocorrelation leads to more runs.

```
runs.test(factor(lm1$res > 0), alternative = "greater")
```

```
##
##  Runs Test
##
## data:  factor(lm1$res > 0)
## Standard Normal = -2.6865, p-value = 0.9964
## alternative hypothesis: greater
```

Pros and Cons of the Runs Test

- Pros: Simple, intuitive (Can be generalized to test randomness)
- Cons:
  - It ignores the magnitude of the residuals e_i.
  - Treatment when residuals $=0$

Durbin-Watson Test

- Proceeds from the assumption that successive errors are correlated:

$$\epsilon_t = \rho\epsilon_{t-1} + \omega_t, \qquad |\rho| < 1$$

- Note: In Time Series analysis, this is called a first-order autoregressive model, abbreviated AR(1), or first-order autocorrelation.
- The actual autocorrelation structure may be more complex (e.g. AR(2), AR(3), etc.) In this case, the first-order structure is a simple approximation.
- Durbin-Watson statistic

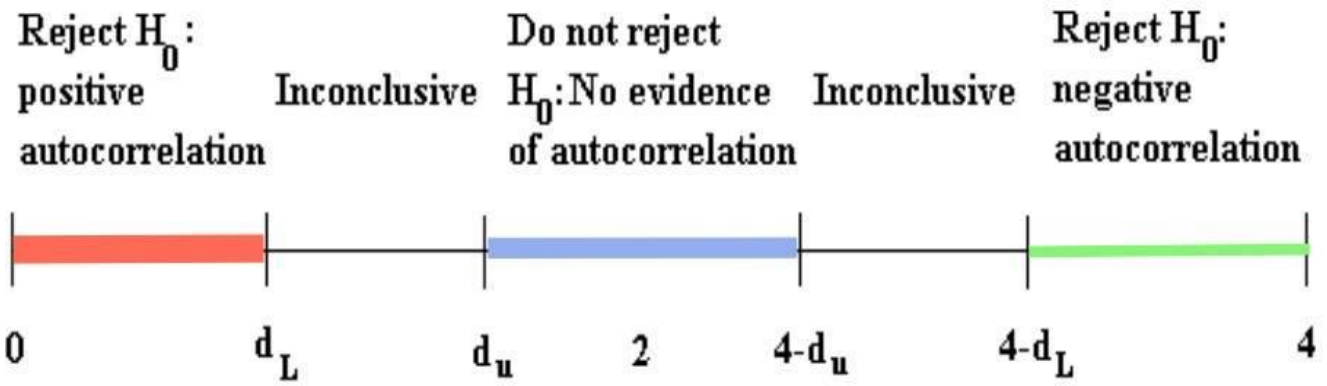
$$d = \frac{\sum_{t=2}^{n}(e_t - e_{t-1})^2}{\sum_{t=1}^{n} e_t^2}$$

Properties of Durbin-Watson Statistic (d)

- $d \approx 2(1 - \hat{\rho})$, where $\hat{rho}$ estimates the autocorrelation ρ by

$$\hat{\rho} = \frac{\sum_{t=2}^{n} e_t e_{t-1}}{\sum_{t=1}^{n} e_t^2}$$

- d is a test statistic for testing: $H_0 : \rho = 0$ v.s $H_A : \rho > 0$
- The null hypothesis indicates that successive residuals are not correlated.
- Under the H_0 of no autocorrelation, d should be close to 2.



Durbin-Watson Test in R

In R, `durbinWatsonTest()` in `library(car)` can produce an approximate P-value (by simulation) for the DW test.

```
library(car)
durbinWatsonTest(lm1, alt="positive")
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7506122     0.3282113       0
##  Alternative hypothesis: rho > 0
```

```
durbinWatsonTest(lm1, alt="negative")
```

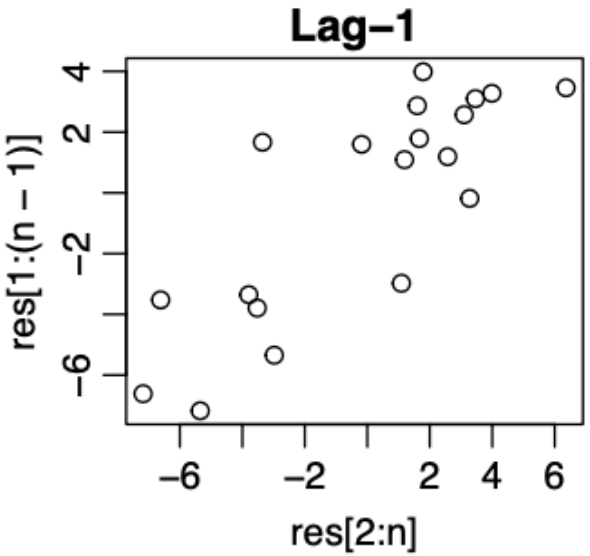
```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7506122     0.3282113       1
##  Alternative hypothesis: rho < 0
```

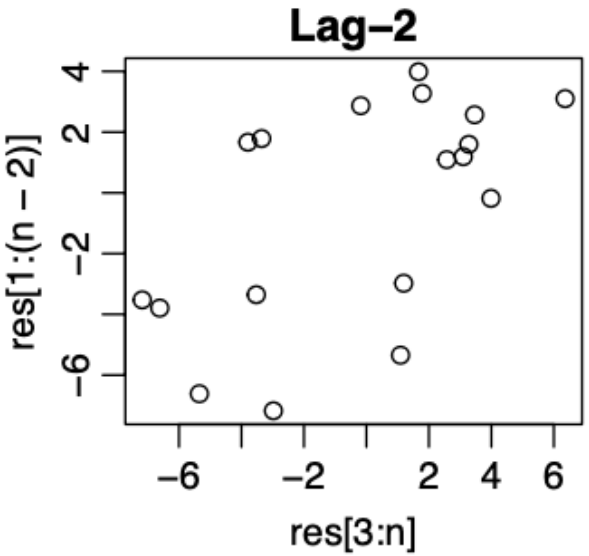
```
durbinWatsonTest(lm1) # two-sided by default
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7506122     0.3282113       0
##  Alternative hypothesis: rho != 0
```

Plotting Residuals Against Lag-k Residuals

```
res = lm1$res
n = length(res)
plot(res[2:n], res[1:(n-1)], main="Lag-1")
plot(res[3:n], res[1:(n-2)], main="Lag-2")
```

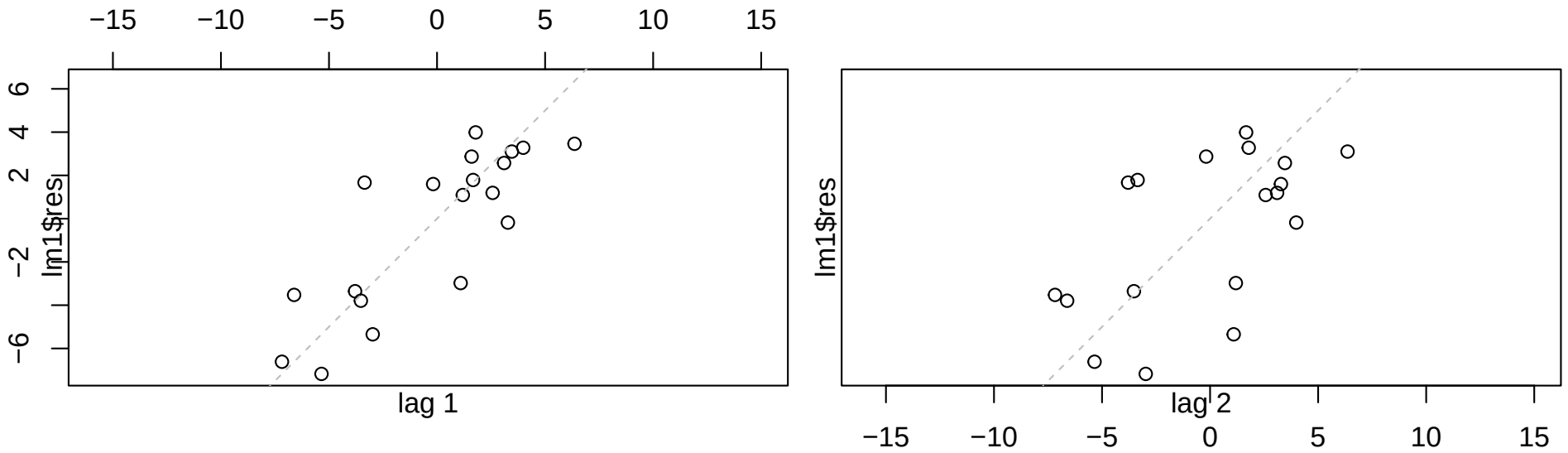




Lag Plots in R

The `lag.plot()` command can produce lag plots.

```
lag.plot(lm1$res, lags=2, layout=c(1,2), do.lines=FALSE)
```



- `lags = k` would produce lag-1 to lag-k plots
- `layout = c(1,2)` arranges the plots in 1 row and 2 columns.

Autocorrelation

The R command `acf()` (autocorrelation function) in R can calculate lag-k autocorrelation.

```
acf(lm1$res, lag.max =5, plot=FALSE)
```

```
##
## Autocorrelations of series 'lm1$res', by lag
##
##      0      1      2      3      4      5
##  1.000  0.751  0.521  0.297 -0.007 -0.220
```

Autocorrelation Function and the Plot

In time-series analysis, one often plot the lag-k autocorrelations against k to examine the autocorrelation structure of a variable. The `acf()` command can produce such autocorrelation plot.

```
acf(lm1$res)
```