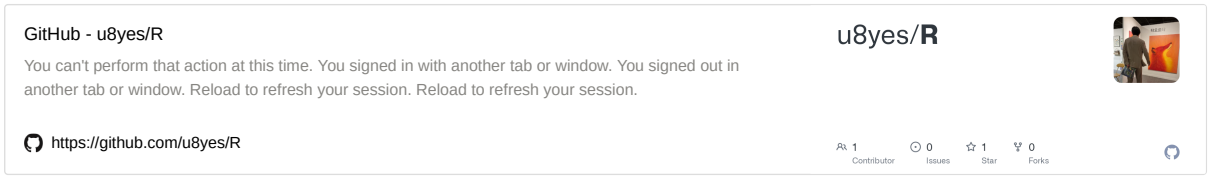
Day34; 20221024

날짜	@2022년 10월 24일
유형	@2022년 10월 24일
태그	

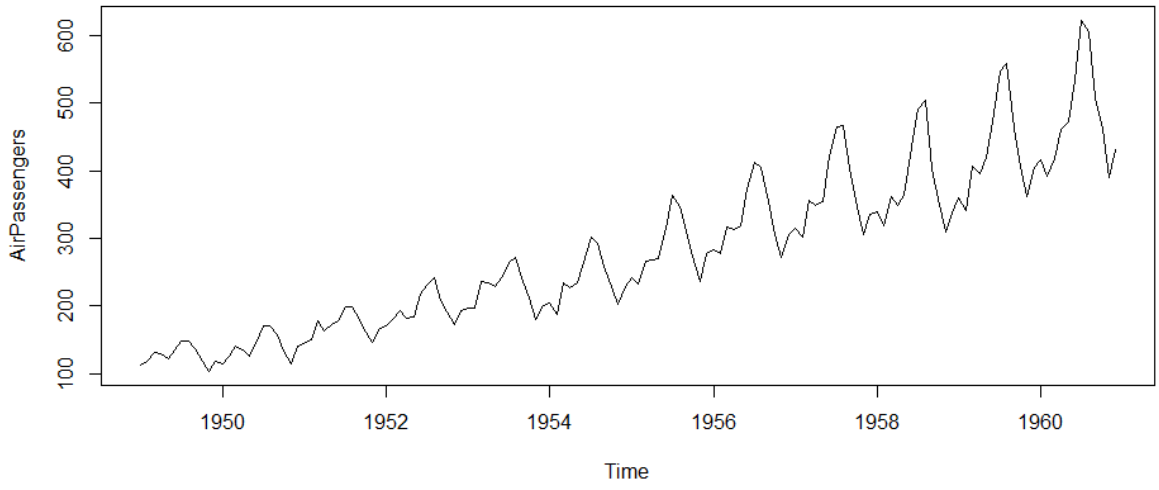
시계열



추세 변동 - 상향 or 하향으로 나아가는

계절 변동 - 1년 이내의 반복적 주기가 있는

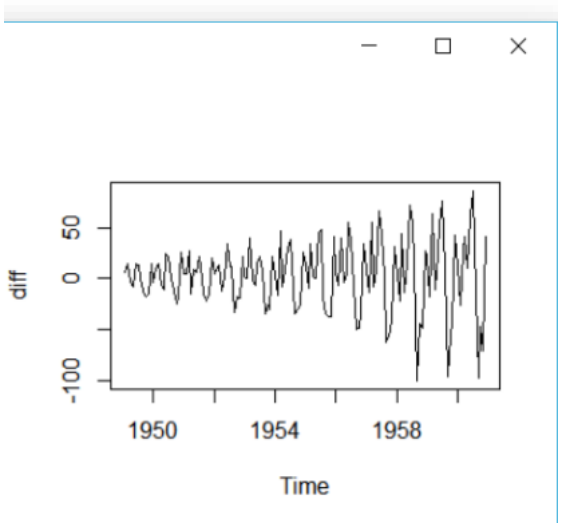
순환 변동 - 2년 ~ 10년 주기에서 일정한 기간이 없는 반복적 요소



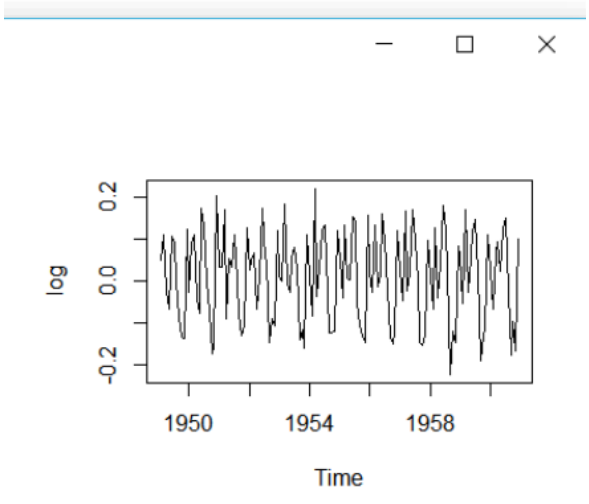

```
## 비정상성 시계열 -> 정상성 시계열

# 단계1: 데이터 셋 가져오기
data("AirPassengers") # 12년(1949~1961년)간 매월 항공기 탑승 승객 수를 기록한 시계열 자료.
str(AirPassengers) # Time-Series [1:144] from 1949 to 1961:

# 단계2:차분(Differencing) 적용-현재 시점에서 이전 시점의 자료를 빼는 연산으로 평균을 정상화하는데 이용 : 평균 정상화.
x11()
ts.plot(AirPassengers) # TimeSeries.plot = ts.plot
```

12년(1949~1961년)간 매월 항공기 탑승 승객 수를 기록한 시계열 자료.

열 : 평균 정상화 : 차분



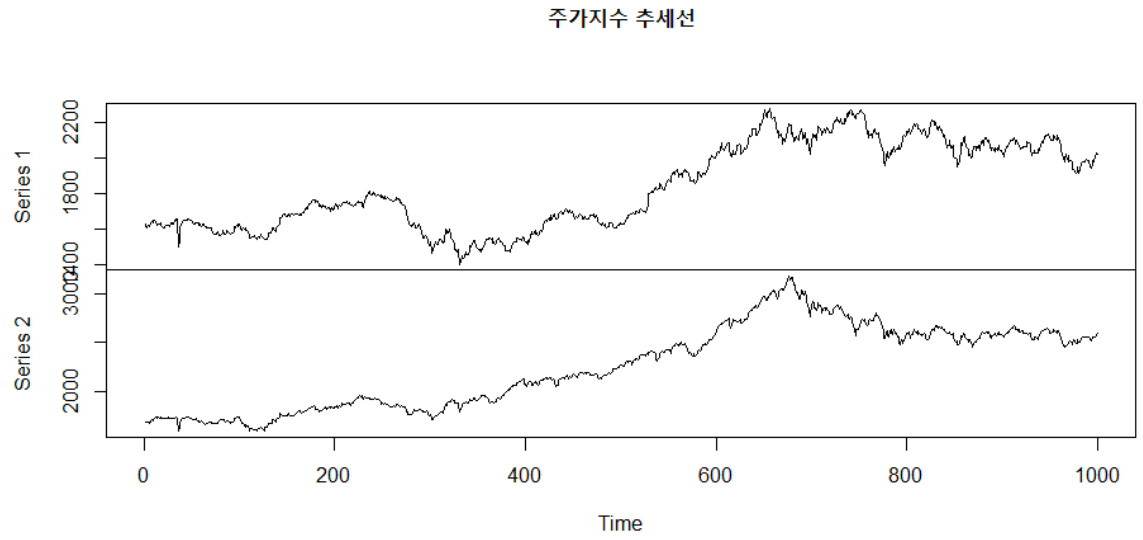
열 : 분산 정상화 : 로그 -> 차분



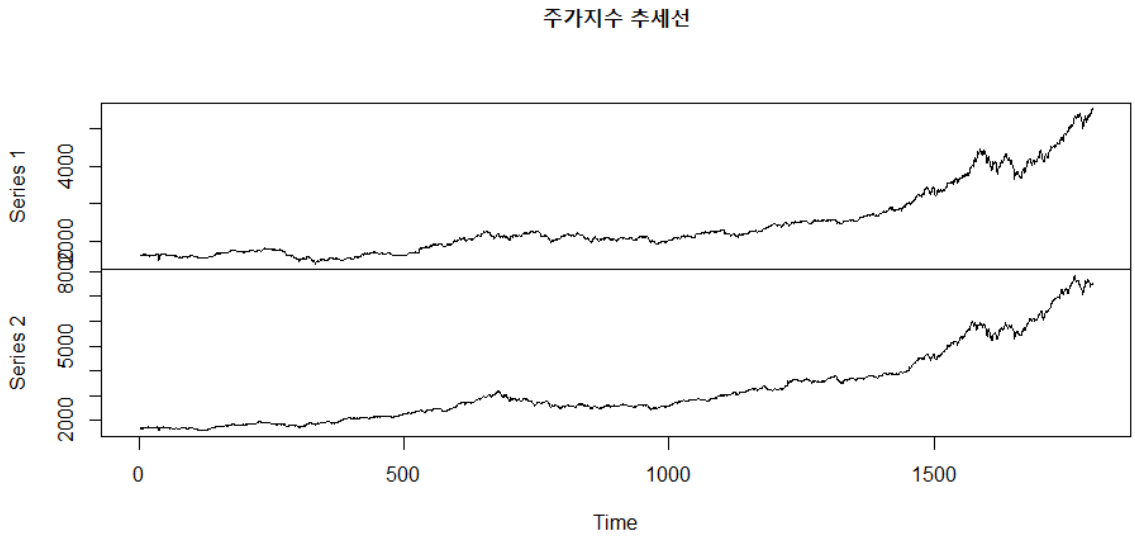

```
par(mfrow=c(1,2))
log <- diff(AirPassengers) # 차분 수행
```

```
plot(log) # 평균 정상화
```

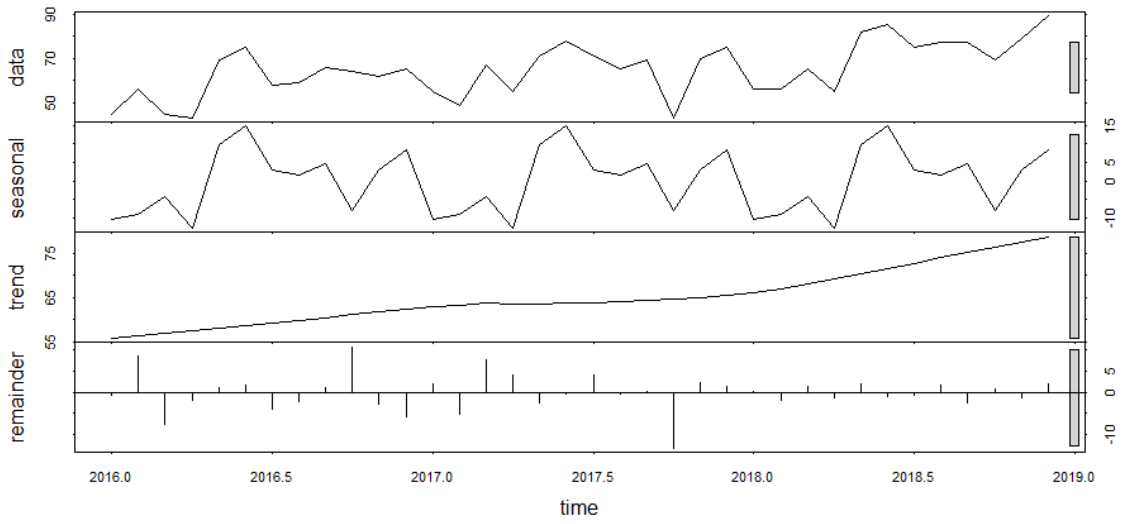
```
log <- diff(log(AirPassengers)) # 로그+차분 수행
plot(log) # 분산 정상화
```




```
# 다중 시계열 자료 시각화

# 단계1 : 데이터 셋 가져오기
data("EuStockMarkets") # 유럽(1991~1998년)의 주요 주식의 주가지수 일일 마감 가격.

head(EuStockMarkets) # DAX(독일) SMI(스위스) CAC(프랑스) FTSE(영국)
str(EuStockMarkets) # Time-Series [1:1860, 1:4]

# 단계2 : 데이터프레임으로 변환
EuStock <- data.frame(EuStockMarkets)
EuStock
head(EuStock)

# 단계3 : 단일 시계열 데이터 추세선
X11()
plot(EuStock$DAX[1:1000], type = "l", col="red") # 선 그래프 시각화

# 단계4 : 다중 시계열 데이터 추세선
plot.ts(cbind(EuStock$DAX[1:1000], EuStock$SMI[1:1000]), main="주가지수 추세선")
```




```
# 단계4 : 다중 시계열 데이터 추세선
plot.ts(cbind(EuStock$DAX[1:1800], EuStock$SMI[1:1800]), main="주가지수 추세선")
```




```
# 단계4 : 시계열 분해- stl():계절요소, 추세, 잔차 모두 제공.
plot(stl(tsdata, "periodic")) # periodic : 주기
```

decompose

미국식[ˌdi:kəmˈpouz] 영국식[ˌdi:kəmˈpəuz]

동사

1 (자연스런 화학 작용에 의해) 분해[부패]되다 (=decay, rot)

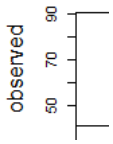
a decomposing corpse

부패되고 있는 시체

2 (더 작은 부분들로) 분해하다

영어사전 다른 뜻 1

관찰치 데이터 = 기본 데이터



observe

미국식[əbˈzɜ:rv] 영국식[əbˈzɜ:v]

동사

1 ...을 보다[(보고) 알다/목격하다]

Have you **observed** any changes lately?

최근에 무슨 변화가 보였나요?

2 관찰[관측/주시]하다 (=monitor)

I felt he was observing everything I did.

나는 그가 내가 하는 모든 것을 주시하고 있다는 것을 느꼈다.

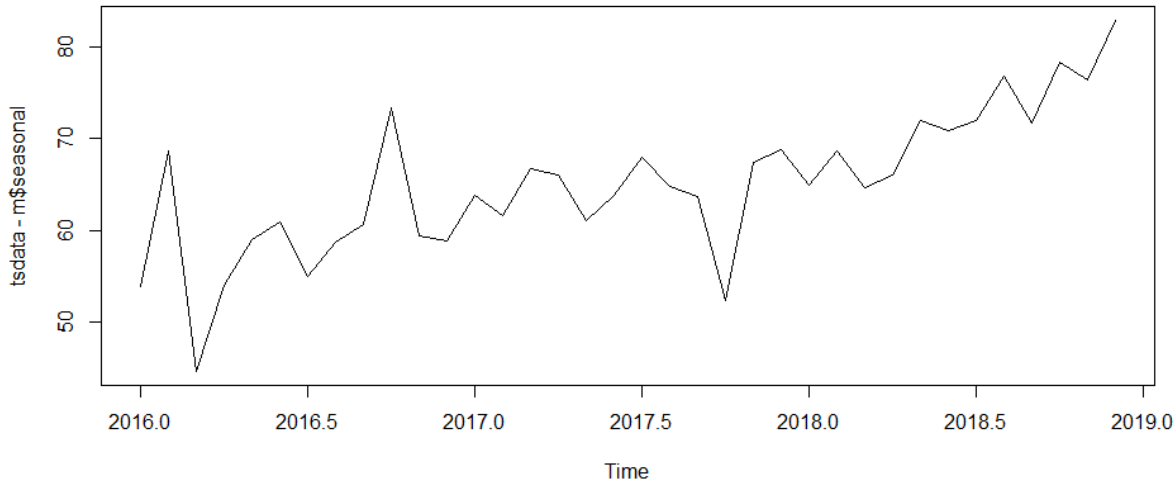
3 (발언·논평·의견을) 말하다 (=comment)

She **observed** that it was getting late.

그녀가 시간이 늦어지고 있다고 말했다.

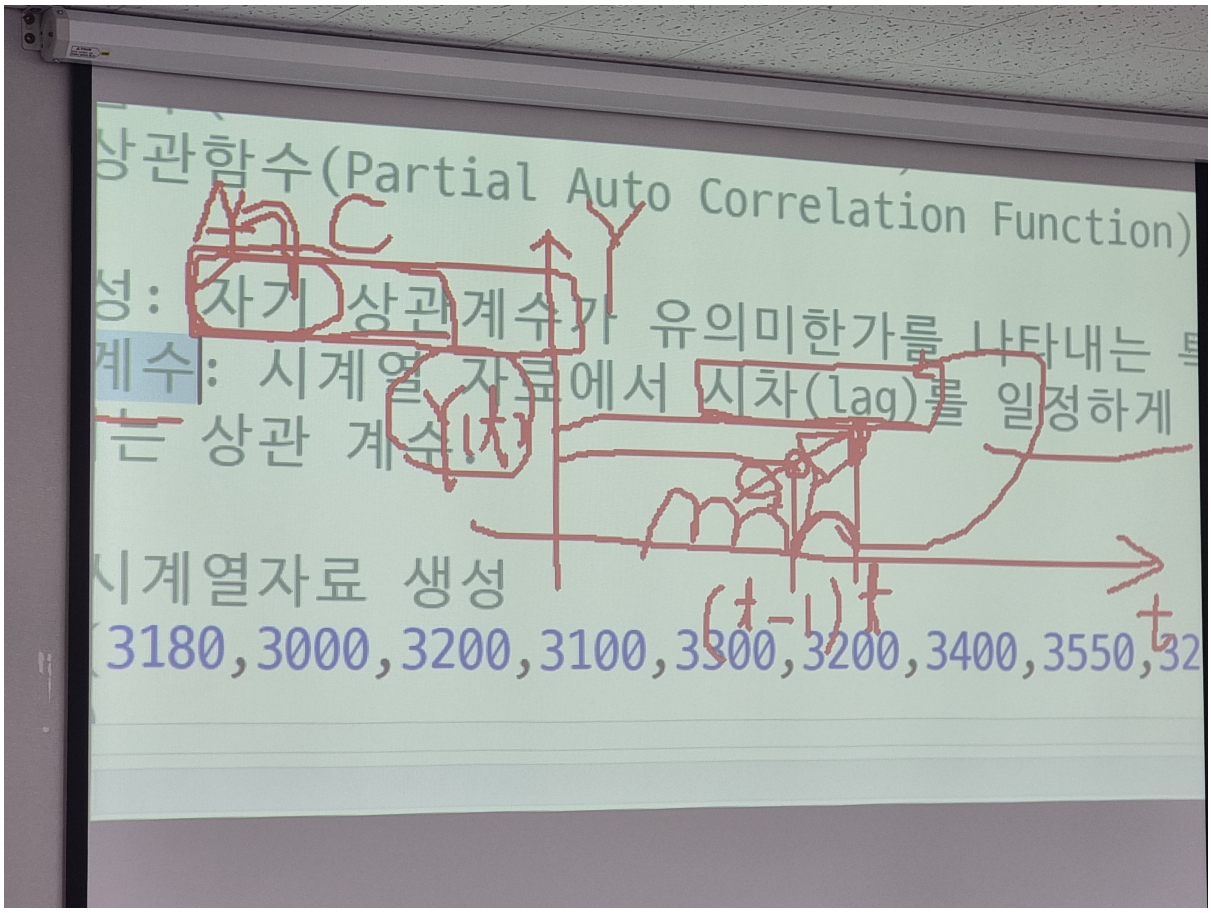
영어사전 다른 뜻 1

계절요인만 빼고 나머지를 보여주는 데이터



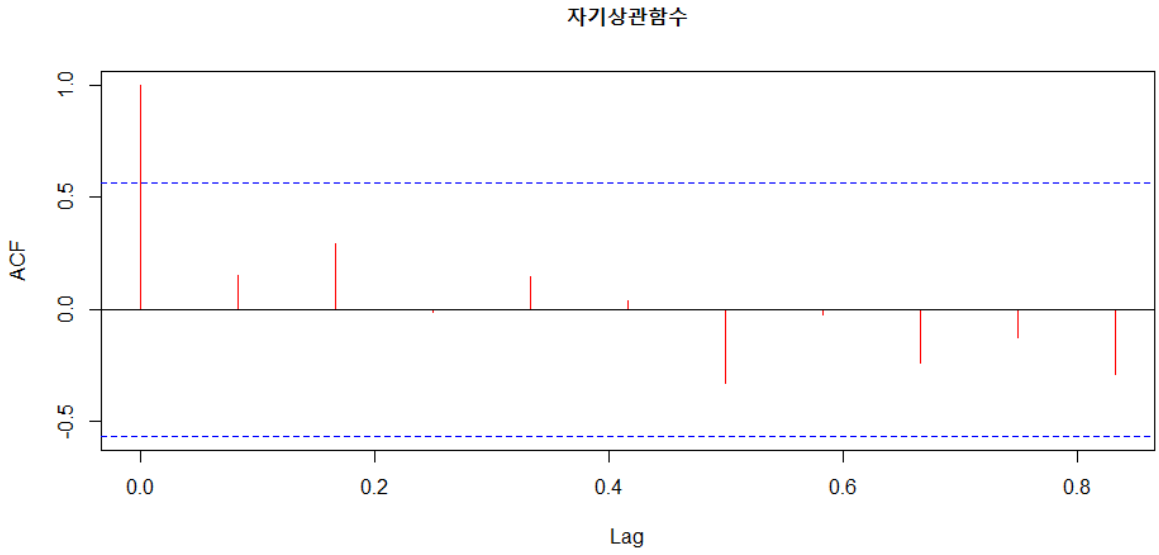
```
plot(tsdata - m$seasonal) # 계절요인을 제거한 그래프.
```

자기 상관계수



시차가 있으면 그 시차(lag)끼리 값을 상관계수로 알아봄.

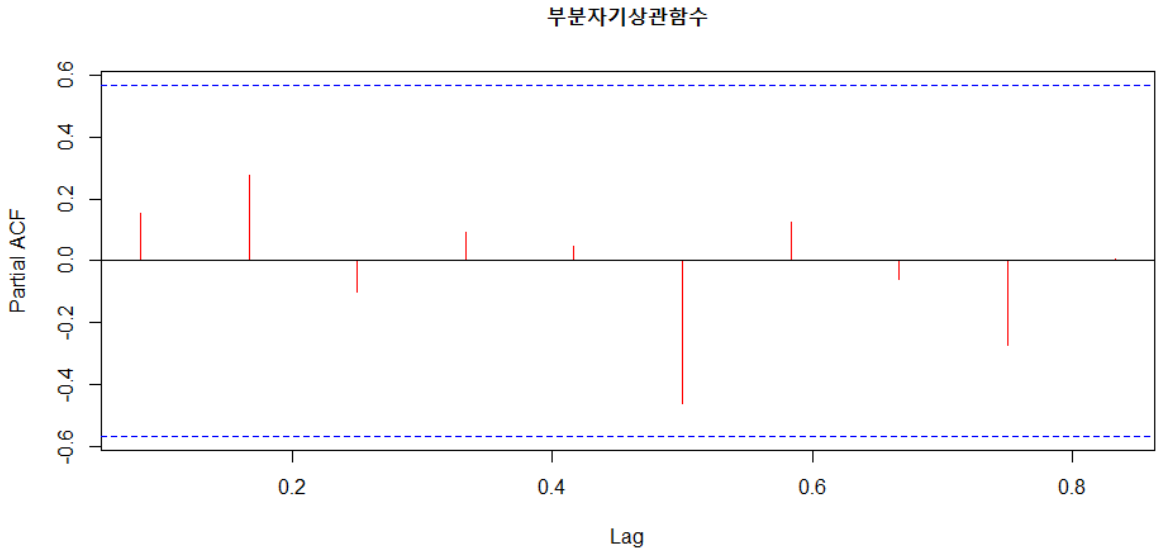
처음에 1은 나 자신이기 때문에 1이라서 아무 의미없다.




```
###################################
# 자기상관함수/ 부분자기상관함수
###################################
# 자기상관함수(Auto Correlation Function) # ACF
# 부분자기상관함수(Partial Auto Correlation Function) # PACF

# 자기상관성: 자기 상관계수가 유의미한가를 나타내는 특성.
# 자기상관계수: 시계열 자료에서 시차(lag)를 일정하게 주는 경우 얻어지는 상관 계수.
# 시차(lag) - 일정 주기를 가짐

# 단계1 : 시계열자료 생성
input <- c(3180,3000,3200,3100,3300,3200,3400,3550,3200,3400,3300,3700) # 의미가 없다 # 데이터가 드러나게 할 수 있게 부각시키기 위해 임의로 만든 숫자
length(input) # 12

tsdata <- ts(input, start = c(2015, 2), frequency = 12) # Time Series # 2015년 2월 ~ 2016년 1월까지의 의미를 가지는 데이터셋, 주기는 12
tsdata

# 단계2 : 자기상관함수 시각화
x11()
acf(na.omit(tsdata), main="자기상관함수", col="red")
```

파란선 안을 집중해서 focus 시켜줌

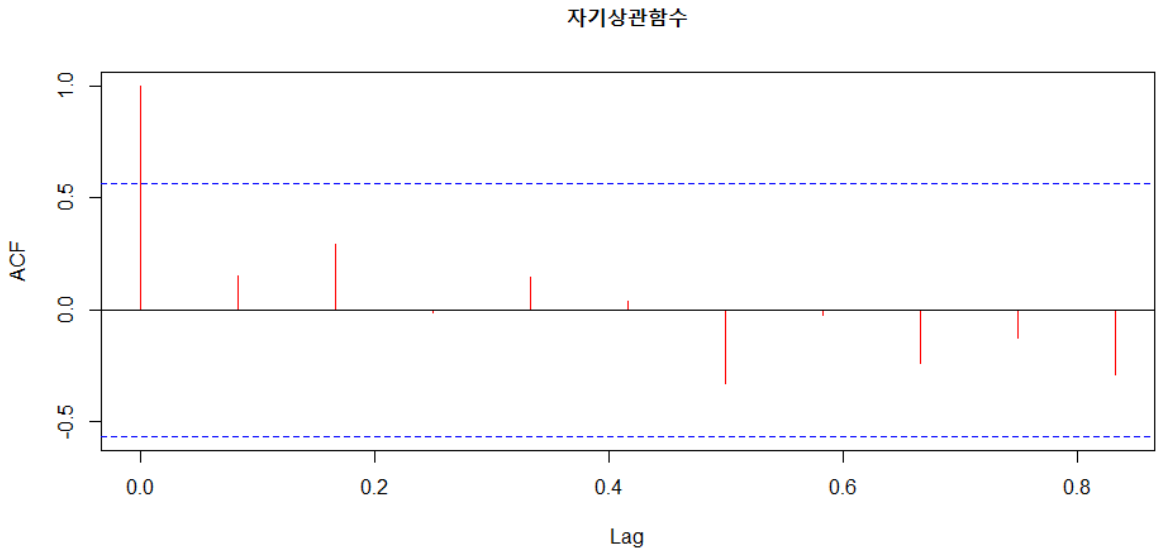
파란선 안에 들어오면 무작위성을 가진 데이터셋이다.



결과는 동일하지만 조금씩 음수, 양수 위치가 다르다.

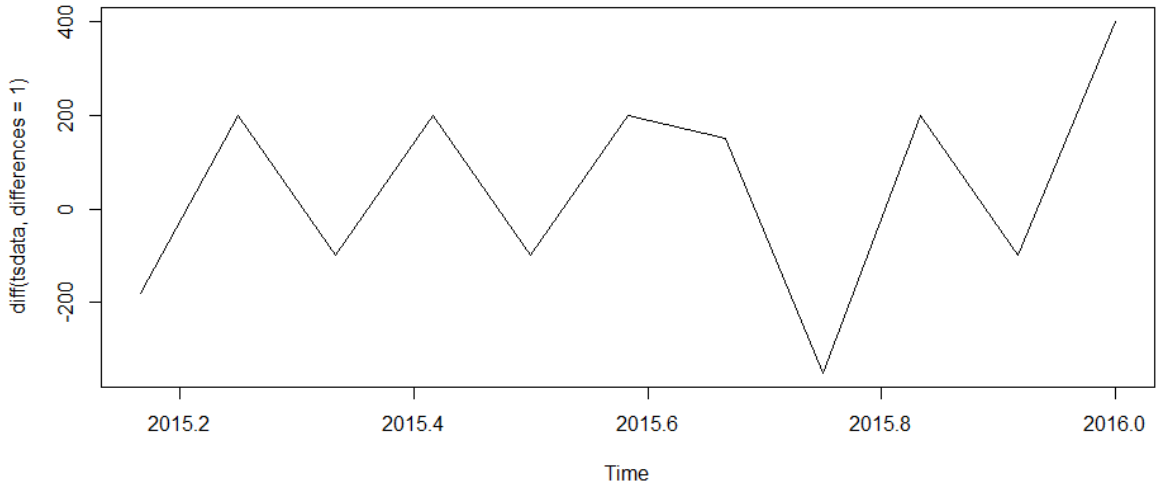
유의미성은 시간에 대한 의존성이 없다 = random = 무작위성의 의미를 띈다라고 해석.

다음의 표는 선형적인 특성을 가지고 있다.

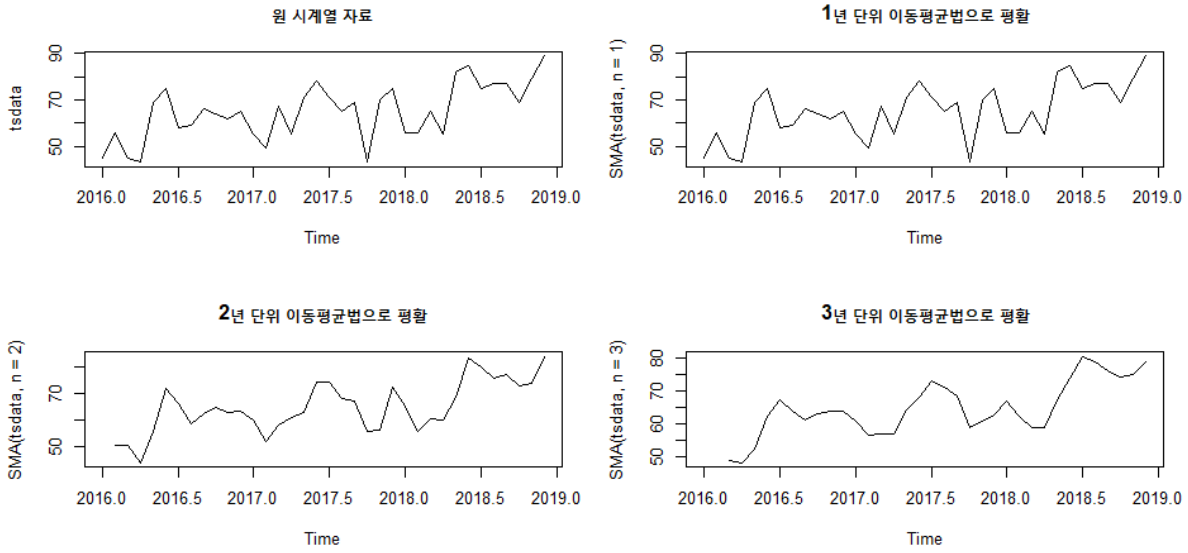



```
###################################
# 추세 패턴 찾기 시각화
###################################
# 추세패턴:시계열 자료가 증가 또는 감소하는 경향이 있는지 알아보고, 증가나 감소의 경향이 선형인지 비선형인지를 찾는 과정.

## 시계열 자료의 추세 패턴 찾기 시각화

# 단계1 : 시계열 자료 생성
input <- c(3180,3000,3200,3100,3300,3200,3400,3550,3200,3400,3300,3700)

# Time Series
tsdata <- ts(input, start = c(2015, 2), frequency = 12)

# 단계2 : 추세선 시각
plot(tsdata, type="l", col="red")
```

```
# 단계3 : 자기상관 함수 시각화
acf(na.omit(tsdata), main="자기상관함수", col="red")
```




```
# 단계4 : 차분 시각화
plot(diff(tsdata, differences=1))
# 1 주기를 가지고 출력 # 평균 0을 기준으로 낮아지고 올라감
```

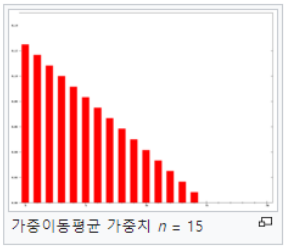


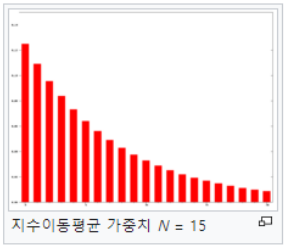

```
###################################
# 평활법(Smoothing Method)
###################################
```

```
# - 수학/통계적 방법의 분석이 아닌 시각화를 통한 직관적 방법의 데이터 분석 방법.
# - 단기 예측. 1개(일변량)

# - 시계열 자료의 체계적인 자료의 흐름을 파악하기 위해서 과거 자료의 불규칙적인 변동을 제거하는 방법.

# - 이동 평균(Moving Average) : 시계열 자료를 대상으로 일정한 기간의 자료를 평균으로 계산하고, 이동 시킨 추세를 파악하여 추세를 예측하는 분석 기법.

# 단계1: 시계열  자료 생성
data <- c(45,56,45,43,69,75,58,59,66,64,62,65,
          55,49,67,55,71,78,71,65,69,43,70,75,
          56,56,65,55,82,85,75,77,77,69,79,89)
length(data) # 36

tsdata <- ts(data, start = c(2016, 1), frequency = 12)

tsdata

# 단계2 : 평활 관련 패키지 설치
install.packages("TTR")
library(TTR)

# 단계3 : 이동평균법으로 평활 및 시각화
par(mfrow=c(2,2))
plot(tsdata, main="원 시계열 자료") # 시계열 자료 시각화
plot(SMA(tsdata, n=1), main="1년 단위 이동평균법으로 평활")
plot(SMA(tsdata, n=2), main="2년 단위 이동평균법으로 평활")
plot(SMA(tsdata, n=3), main="3년 단위 이동평균법으로 평활")
par(mfrow=c(1,1))
```

지수이동평균(EMA) - 가까운 것에 더 가중치를 줘서 평균을 계산함.

오른쪽 그래프는 가장 최근 데이텀 포인트에 대한 가장 높은 가중치에서 0으로 가중치가 어떻게 감소하는지 보여준다. 다음 지수 이동 평균의 가중치와 비교할 수 있다.


가중이동평균 가중치 n = 15


지수이동평균 가중치 N = 15

지수이동평균 [편집]

지수이동평균(Exponential Moving Average)[3] 또는 **지수가중이동평균**(Exponentially Weighted Moving Average)은 지수적으로 감소하는 가중치를 적용하는 1차 무한 임펄스 응답 필터다. 이 경우 오래된 데이터에 대한 가중치는 기하 급수적으로 감소하지만 0이 되지는 않는다. 오른쪽 그래프는 가중치 감소의 예를 보여준다.

급수 Y에 대한 가중이동평균은 재귀적으로 계산할 수 있다.

$$S_t = \begin{cases} Y_1, & t = 1 \\ \alpha \cdot Y_t + (1-\alpha) \cdot S_{t-1}, & t > 1 \end{cases}$$

- 계수 α는 0과 1 사이의 평활상수로 가중치 감소 정도를 나타냅니다. α 가 클수록 오래된 관측치가 더 빨리 감소된다.
- Y_t는 기간 t 에서의 값이다.
- S_t는 임의의 기간 t 에서의 지수이동평균 값이다.

단순이동평균과 지수이동평균의 관계 [편집]

응용 분야에 따라 권장되는 값은 있지만 선택해야 할 "허용된" α값은 없다. 일반적으로 사용되는 α값은 $2/(N+1)$ 다. 이는 $\alpha_{\mathrm{EMA}} = 2/(N_{\mathrm{SMA}}+1)$ 일 때 단순이동평균과 지수이동평균의 가중치가 같은 값을 갖기 때문이다.

1. 정상성을 가진 시계열 모형
   - ✓ 자기회귀모형(AR), 이동평균모형(MA), 자기회귀이동평균모형(ARMA)
2. 비정상성을 가진 시계열 모형(차수 적용)
   - ✓ 자기회귀누적이동평균모형(ARIMA)
   - ✓ 형식) ARIMA(p, d, q) : 3개의 인수
   - ✓ p : AR모형 차수, d : 차분 차수, q : MA모형 차수

- 정상성 - 자기회귀모형(AR), 이동평균모형(MA), 자기회귀이동평균모형(ARMA)
- 비정상성 - 자기회귀누적이동평균모형(ARIMA)
  형식) ARIMA(p, d, q) : 3개의 인수
  p : AR모형 차수, d : 차분 차수, q : MA모형 차수

비정상성 - 추세, 순환, 계절, 불규칙...

- 시계열자료 특성분석

시각화해서 정상성인지 아닌지 확인을 해봄

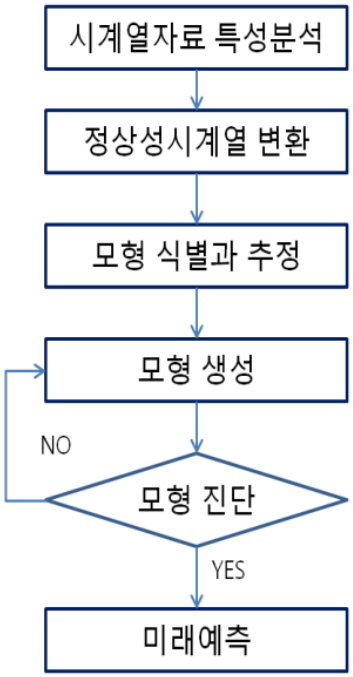
- 정상성시계열로 변환

차분을 했을 때 정상성을 나타내지 않으면 다시 한번 차분하는 것이다.

● 시계열 분석 절차

- ARIMA 모형을 이용

[단계1] 시계열자료 특성분석(정상성/비정상성)
[단계2] 정상성시계열 변환
[단계3] 모형 식별과 추정
[단계4] 모형 생성
[단계5] 모형 진단(모형 타당성 검정)
[단계6] 미래 예측(업무 적용)



모형 식별과 추정 - ARIMA 모델

ARIMA 모델을 통해서 모형을 생성할 수 있다.

그래서 타당한 모델인지 진단을 할 수 있다.

(6개월 ~ 2년 정도의 데이터를 예측해보는 것이 포인트.)

---

coefficient

[kɔefisjɑ̃]

남성형 명사

1 비율, 퍼센트,정도,요인
coefficient d'erreur
오차 비율

2 호봉(號俸), 봉급[급여]계수
cadre qui est au **coefficient** 450
450호봉의 간부

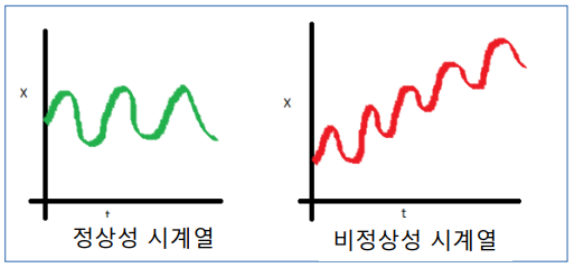
3 [수학] 계수
coefficient de proportionnalité
비례 계수

프랑스어사전 다른 뜻 1

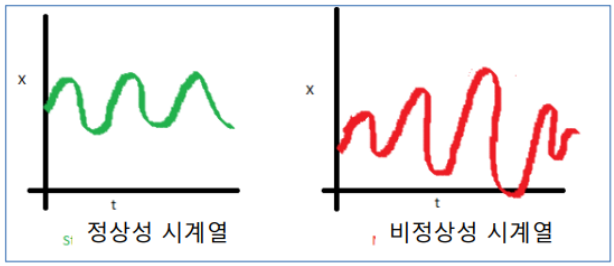
정상성 VS 비정상성

1단계. 시계열 자료 특성 분석 : 비정상성과 정상성 시계열 확인

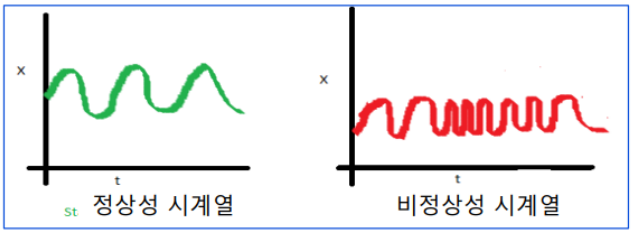
1. 시간의 추이와 관계 없이 평균이 불변



2. 시간의 추이와 관계 없이 분산이 불변



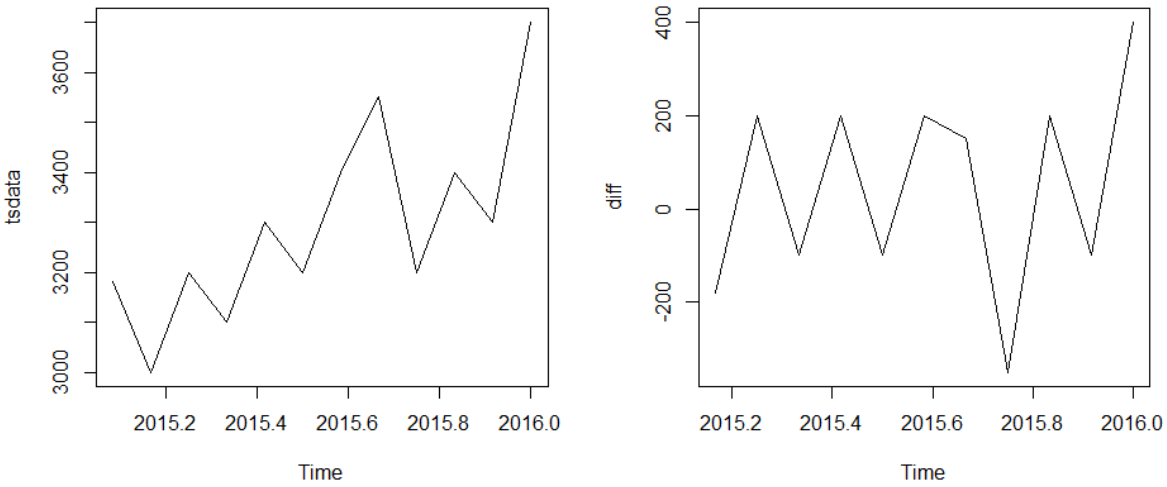
3. 두 시점 간의 자기상관(공분산)이 기준시점과 무관



1. 비정상성이 상향성이던지
2. 비정상성이 하향성이던지
3. 비정상성이 주기성이 다르다던지

왼쪽은 시계열자료 특성분석하기 위해 차분 이전에 시각화.

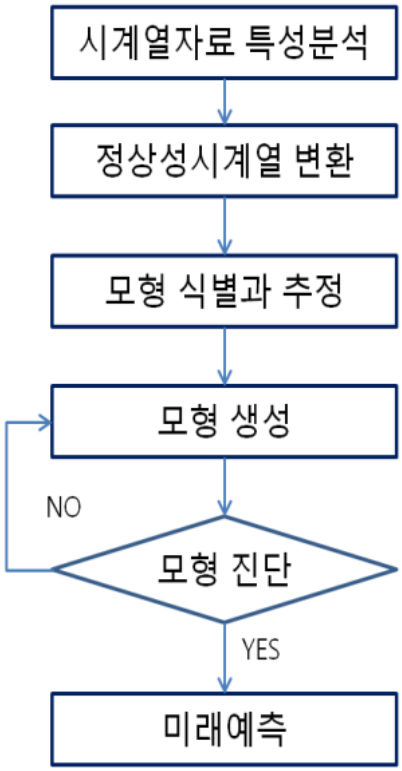
오른쪽이 한번 차분이 들어간 상태




```
# 단계2:정상성시계열 변환
par(mfrow=c(1,2))
ts.plot(tsdata)
diff <- diff(tsdata) # 차분
plot(diff)
```

회귀 - 데이터가 많아질수록 평균값으로 수렴 또는 회귀한다. (갈톤 - 예: 부모, 자식간의 키 차이 분석)

모델 = 모형



arima 모델의 값을 출력하면 ARIMA(1,1,0)가 가장 적합하다고 피드백 되고 그걸 해석하는 방법만 알면 됨.

```
# 단계3: 모형 식별과 추정
install.packages('forecast')
library(forecast)

arima <- auto.arima(tsdata) # 시계열 데이터 이용. # ARIMA 모형을 통해
arima
# ARIMA(1,1,0) - 1번 차분한 결과가 정상성시계열의 AR(1) 모형으로 식별.
```

2. 비정상성을 가진 시계열 모형(차수 적용)

- ✓ 자기회귀누적이동평균모형(ARIMA)
- ✓ 형식) ARIMA(p, d, q) : 3개의 인수
- ✓ p : AR모형 차수, d : 차분 차수, q : MA모형 차수

```
> plot(diff)
> library(forecast)
> arima <- auto.arima(tsdata) # 시계열 데이터 이용. # ARIMA 모형을 통해
> arima
Series: tsdata
ARIMA(1,1,0)

Coefficients:
          ar1
      -0.6891
s.e.   0.2451

sigma^2 = 31644:  log likelihood = -72.4
AIC=148.8   AICc=150.3   BIC=149.59
```

3단계. 모형 식별과 추정 : auto.arima()함수 이용

- ✓ auto.arima 함수 : ARIMA 모형의 초적화된 파라미터 제공
- ✓ ARIMA : 비정상성을 가진 시계열 자료를 모형 생성

  형식) ARIMA(p, d, q) : 3개 파라미터

  - ✓ p : AR차수, d : 차분차수, q : MA 차수
  - ✓ auto.arima()함수 : 모형과 차수 제공

자기회귀모형(AR)
이동평균모형(MA)
자기회귀이동평균모형(ARMA)

[ARIMA(p,d,q) 모형 → 정상성 시계열 모형 식별]

d=0이면, ARMA(p, q)모형이며, 정상성을 만족한다.

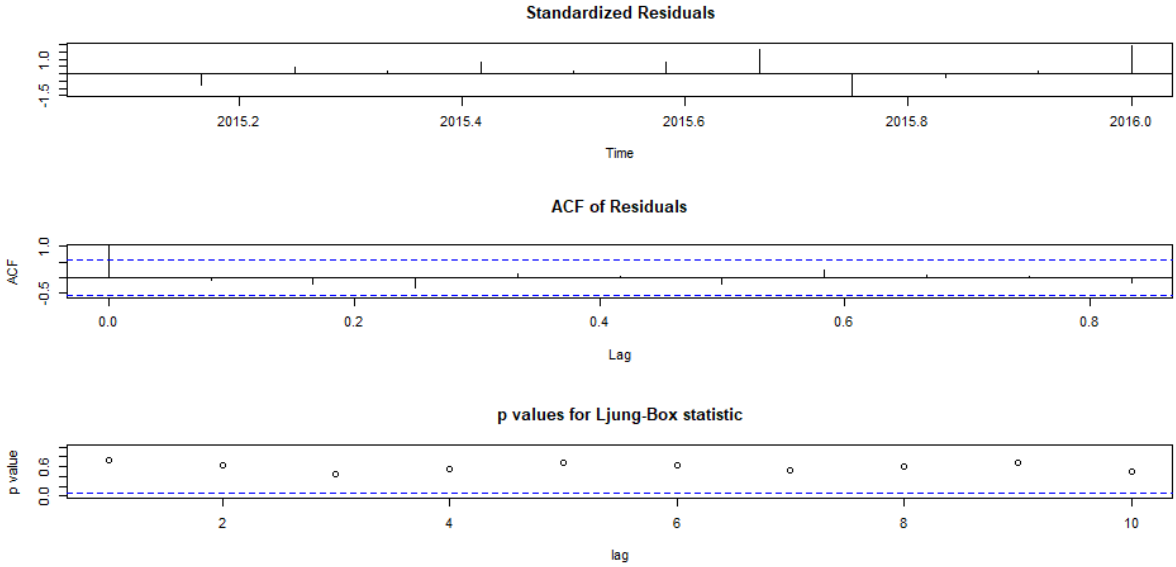
q=0이면 IAR(p, d)모형이며, d번 차분하면 AR(p) 모형을 따른다.

p=0이면 IMA(d, q)모형이며, d번 차분하면 MA(q) 모형을 따른다.

결과가 나오면 무조건 'I'를 앞에 붙여줌.

```
# 단계4: 모형 생성(모델을 찾음)
model <- arima(tsdata, order=c(1,1,0)) # order에 ARIMA 결과를 그대로 써주면 됨.
model
```

```
> model <- arima(tsdata, order=c(1,1,0)) # order에 ARIMA 결과를 그대로 써주면 됨.
> model

Call:
arima(x = tsdata, order = c(1, 1, 0))

Coefficients:
          ar1
      -0.6891
s.e.   0.2451

sigma^2 estimated as 28767:  log likelihood = -72.4,  aic = 148.8
```

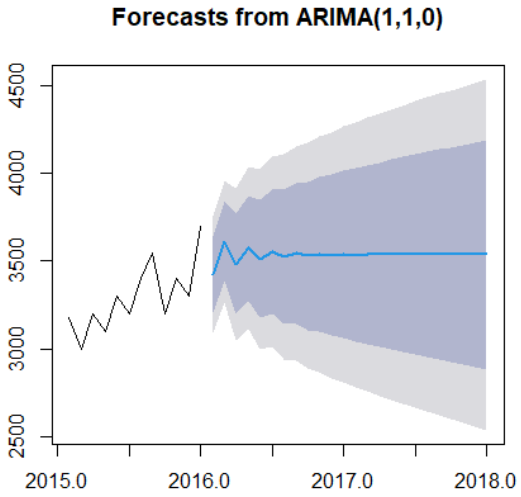


파란선 넘어가면 자기상관성이 떨어진다. 파란선 안에 들어와야 적합하다.

```
# 단계5: 모형 진단(모형 타당성 검정)
# (1) 자기상관함수(AutoCorrelationFunction)에 의한 모형 진단 # 일정 간격을 찾아가는 것.
tsdiag(model) # TimeSeriesDialog # p-value값이 0 이상으로 분포하면 정상값이다.

# (2) Box-Ljung에 의한 잔차항 모형 진단
Box.test(model$residuals, lag = 1, type = "Ljung")
# p-value = 0.7252 > 0.05 : 결론) 모형이 통계적으로 적절하다.
```

p-value값이 0.05가 넘으면 모형이 통계적으로 유의미하다.

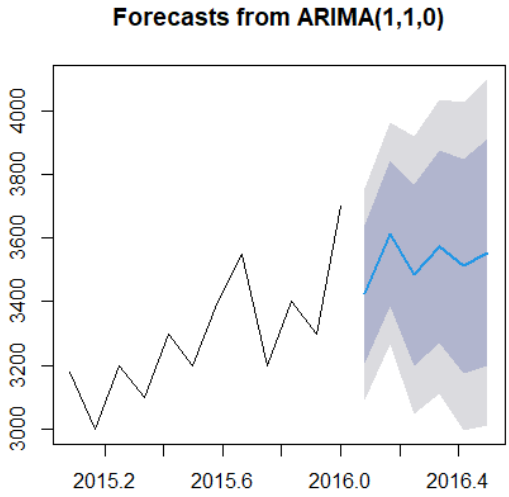
향후 2년치를 예측함.




```
> fore <- forecast(model) # 향후 2년 예측
> fore
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Feb 2016       3424.367 3207.007 3641.727 3091.944 3756.791
Mar 2016       3614.301 3386.677 3841.925 3266.180 3962.421
Apr 2016       3483.421 3198.847 3767.995 3048.203 3918.639
May 2016       3573.608 3272.084 3875.131 3112.467 4034.748
Jun 2016       3511.462 3175.275 3847.649 2997.308 4025.615
Jul 2016       3554.286 3199.003 3909.568 3010.928 4097.643
Aug 2016       3524.776 3143.569 3905.984 2941.770 4107.783
Sep 2016       3545.111 3144.813 3945.408 2932.908 4157.313
Oct 2016       3531.099 3109.224 3952.974 2885.897 4176.301
Nov 2016       3540.754 3100.585 3980.923 2867.574 4213.934
Dec 2016       3534.101 3074.901 3993.300 2831.816 4236.385
Jan 2017       3538.685 3062.192 4015.179 2809.951 4267.420
Feb 2017       3535.526 3041.695 4029.357 2780.277 4290.775
Mar 2017       3537.703 3027.557 4047.849 2757.502 4317.904
Apr 2017       3536.203 3009.958 4062.448 2731.381 4341.025
May 2017       3537.237 2995.565 4078.908 2708.822 4365.651
Jun 2017       3536.524 2979.724 4093.325 2684.972 4388.077
Jul 2017       3537.015 2965.573 4108.457 2663.070 4410.960
Aug 2017       3536.677 2950.901 4122.453 2640.809 4432.545
Sep 2017       3536.910 2937.181 4136.639 2619.704 4454.116
Oct 2017       3536.749 2923.359 4150.140 2598.650 4474.849
Nov 2017       3536.860 2910.124 4163.596 2578.350 4495.371
Dec 2017       3536.784 2896.968 4176.600 2558.270 4515.298
Jan 2018       3536.836 2884.211 4189.462 2538.732 4534.941
```

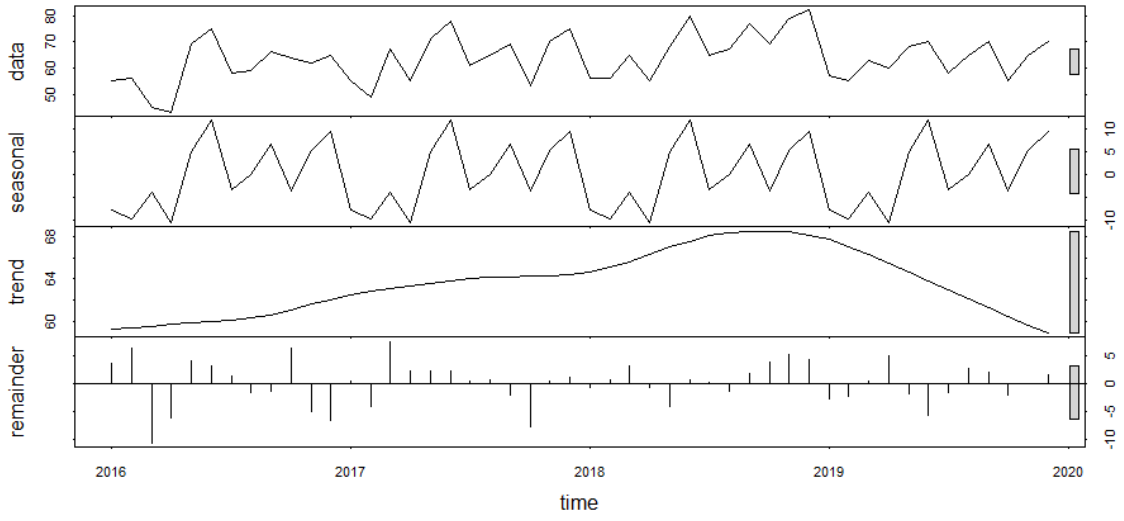
```
# 단계6 : 미래 예측(업무 적용)
fore <- forecast(model) # 향후 2년 예측
fore
x11()
par(mfrow=c(1,2))
plot(fore) # 향후 24개월 예측치 시각화

model2 <- forecast(model, h = 6) # 향후 6개월 예측치 시각화
plot(model2)
```

6개월치를 예측해서 보여줌

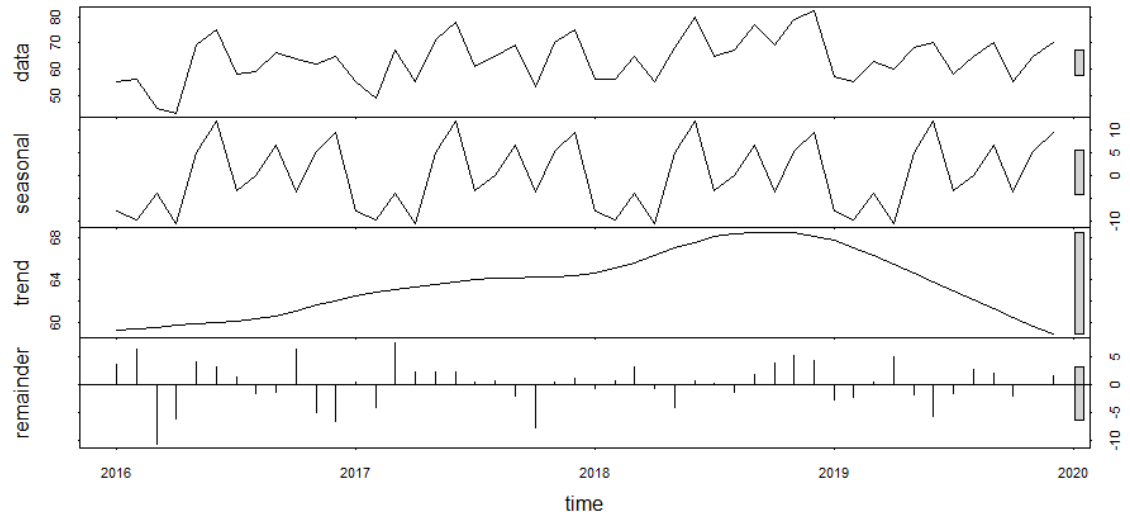



```
model2 <- forecast(model, h = 6) # 향후 6개월 예측치 시각화
plot(model2)
```

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

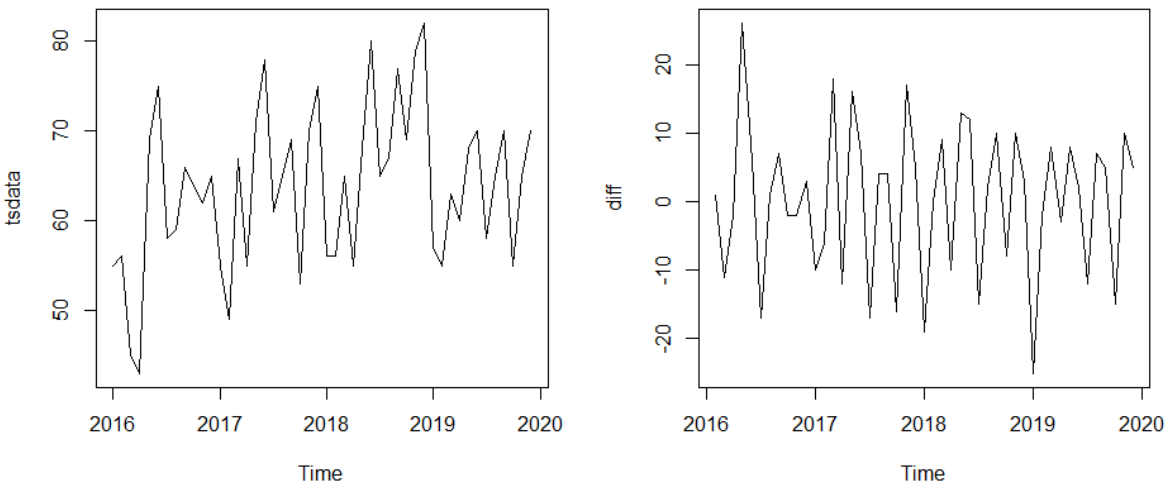



```
# (3) 시계열요소분해 시각화
ts_feature <- stl(tsdata, s.window="periodic") # or Decompose함수를 사용
plot(ts_feature)
```

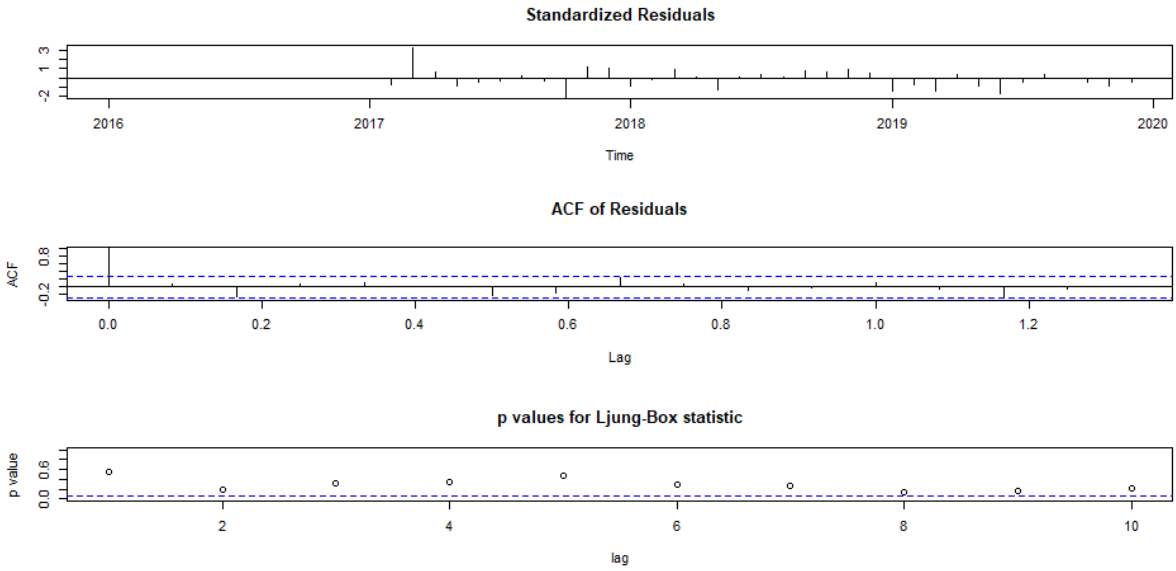



```
## 정상성시계열의 계절형 # 계절형은 주기적이다. 데이터가 짧건 길건 상관없이 주기성을 가지고 있다.

# 단계1 : 시계열자료 특성분석

# (1) 데이터 준비 # 추위에 기반하는 특징을 가지는 주기를 가지고 있다. 반드시 똑같은 개념은 아니다.
data <- c(55,56,45,43,69,75,58,59,66,64,62,65,
          55,49,67,55,71,78,61,65,69,53,70,75,
          56,56,65,55,68,80,65,67,77,69,79,82,
          57,55,63,60,68,70,58,65,70,55,65,70)
length(data)# 48

# (2) 시계열자료 생성
tsdata <- ts(data, start=c(2016, 1),frequency=12)

#tsdata <- AirPassengers # 실제 data 적용.
tsdata
head(tsdata)
tail(tsdata)

# (3) 시계열요소분해 시각화
ts_feature <- stl(tsdata, s.window="periodic") # or Decompose함수를 사용
plot(ts_feature)
```

오른쪽이 차분한 결과



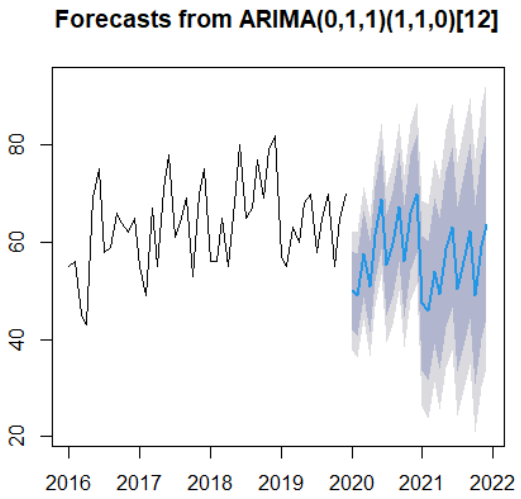
```
# 단계2 : 정상성시계열 변환
par(mfrow=c(1,2))
ts.plot(tsdata)
diff <- diff(tsdata)
plot(diff) # 차분 시각화
```

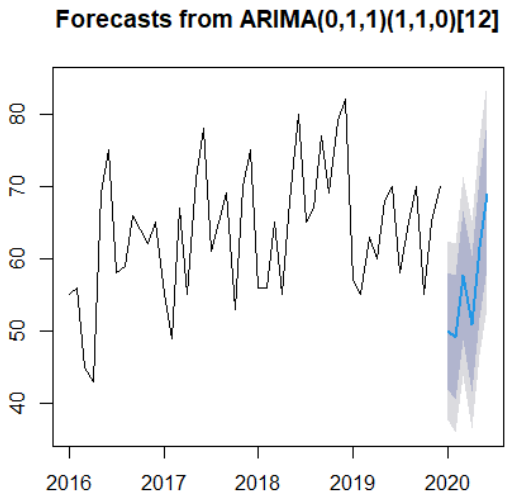



```
# 단계4 : 모형 생성
model <- arima(tsdata, c(0,1,1), seasonal = list(order = c(1,1,0)))
#model <- arima(tsdata, c(2,1,1), seasonal = list(order = c(0,1,0)))
model

# 단계5 : 모형 진단(모형 타당성 검정)
# (1) 자기상관함수에 의한 모형 진단
tsdiag(model)
```

(2)Box-Ljung에 의한 잔차항 모형 진단

```
> Box.test(model$residuals, lag=1, type = "Ljung") # p-value = 0.5618 / p-value = 0.9879

  Box-Ljung test

data:  model$residuals
X-squared = 0.33656, df = 1, p-value = 0.5618
```






```
# 단계6 : 미래 예측
par(mfrow=c(1,2))
fore <- forecast(model, h=24) # 2년 예측
plot(fore)
fore2 <- forecast(model, h=6) # 6개월 예측
plot(fore2)
```

딥러닝의 시작은 신경망으로부터 나옴.

텐서플로우 라이브러리은 파이썬으로 릴리즈 됐음.

2018년에는 파이썬이 R을 역전하기 시작.