**BK21 산업경영공학교육연구단**

정상 프로세스 : 시간에 관계 없이 평균과 분산이 일정한 시계열 데이터

ACF(Autocorrelation Function) : Lag이 1이라는 것은 현 시점과 1 시점 이전의 데이터 값의 차이

모양이 랜덤하면 Stationary 하다고 생각한다.

Non-stationary의 경우 ACF가 천천히 감소한다.

AR 모델 : 전 시점의 데이터와의 관계로 만든 모델

MA 모델 : 전 시점의 에러와의 관계로 만든 모델

ARMA 모델 : 전 시점의 데이터와 에러로 표현한 모델

(AR, MA, ARMA 모델은 데이터가 Stationary여야함)

ARIMA 모델 : 차분을 통해 정상성을 확보할 수 있음

AR(p), I(d), MA(q)를 통해 (p,d,q)로 표현

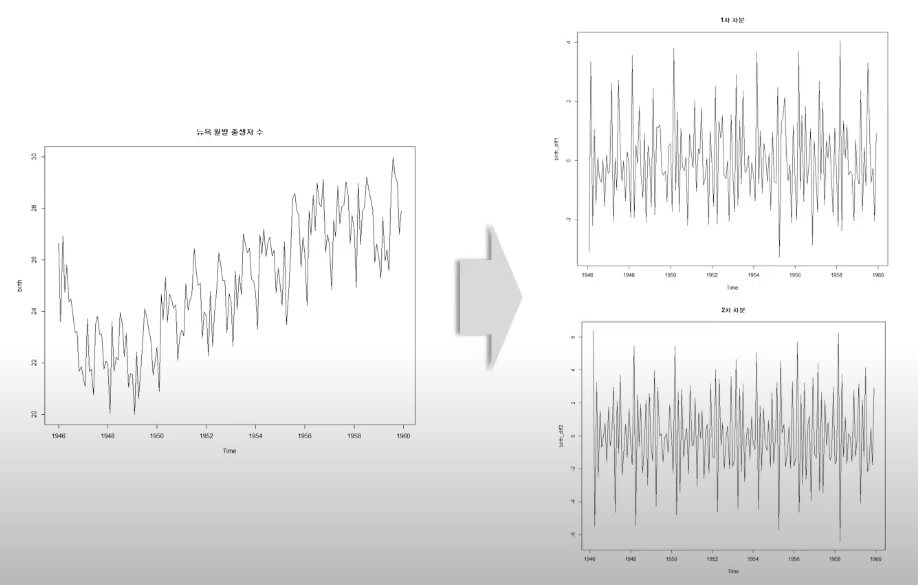
데이터가 정상성을 띤다면 차분할 필요가 없다.

데이터가 단순한 증가/감소 추세가 있다면 1차 차분을 통해 정상성을 확보할 수 있을 것이다.

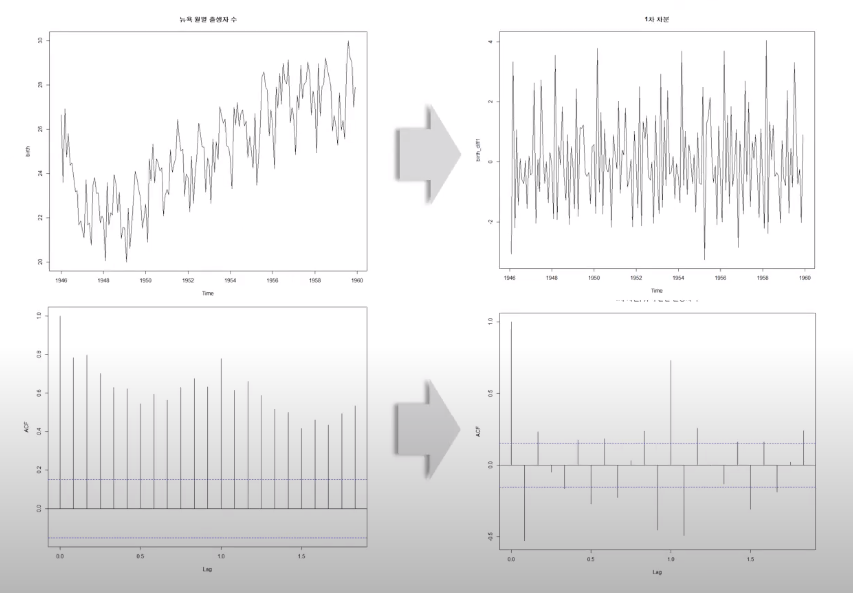
데이터가 등락을 반복하는 복잡한 형태라면 2차 차분까지 필요할 수도 있다.

대부분의 데이터가 2차 차분이면 충분하고, 3차 이상을 통해 정상성을 띠는 데이터는 ARMA 모델이 적합하지 않다.

아래 그림은 데이터를 2차 차분까지 해 본 것. 1차로도 충분하다.



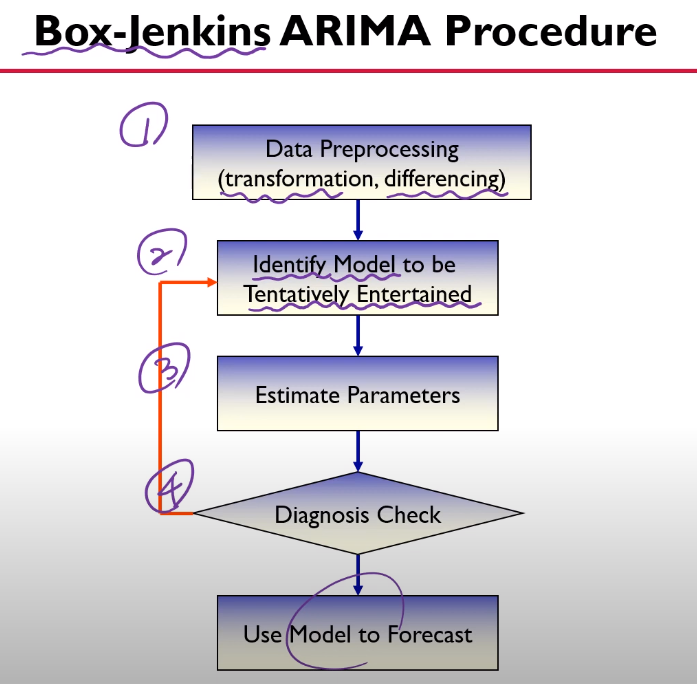
ACF를 보는 법



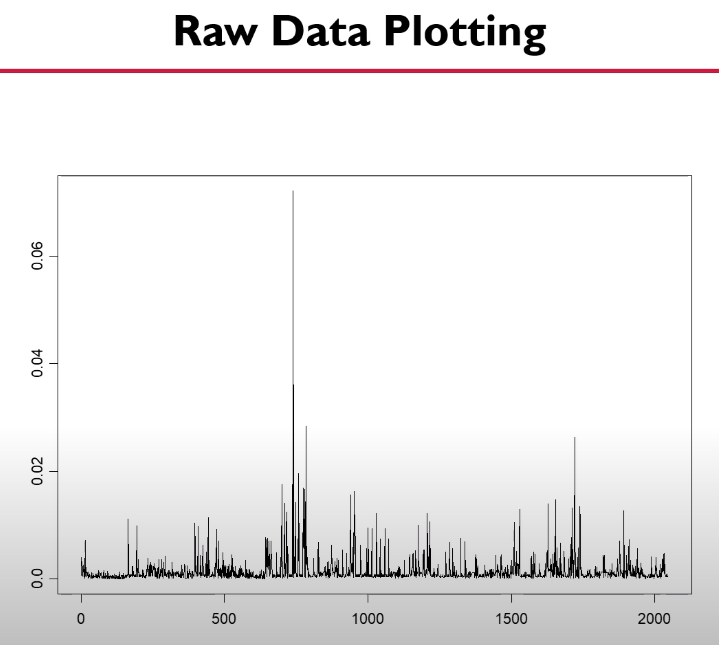
원본 데이터는 천천히 출렁이며 감소하는 추세를 보인다.

1차 차분을 한 데이터는 일정한 패턴이 없다.

시계열 분석의 순서는 전처리(변환, 차분), 임시 모델링, 파라미터 조정, 성능 평가

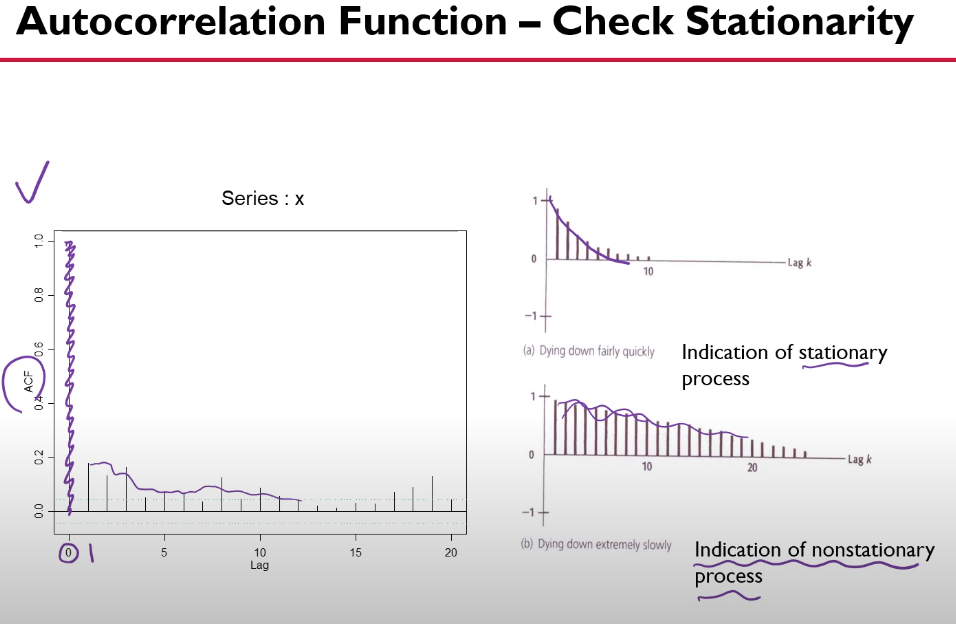


예시



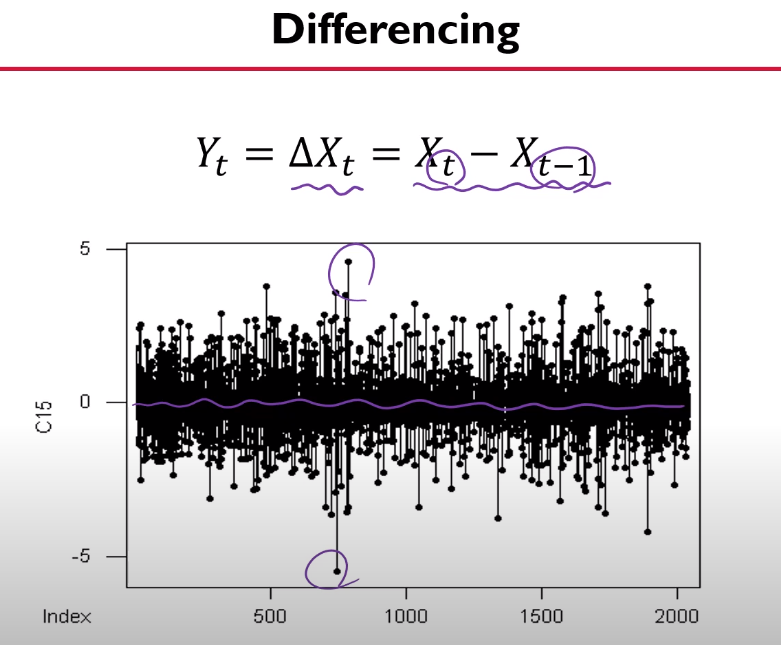
원본 데이터가 정상성이 있는지 없는지 애매하다.

눈으로 보기에 가늠이 되지 않을 때에는 ACF를 그려본다.

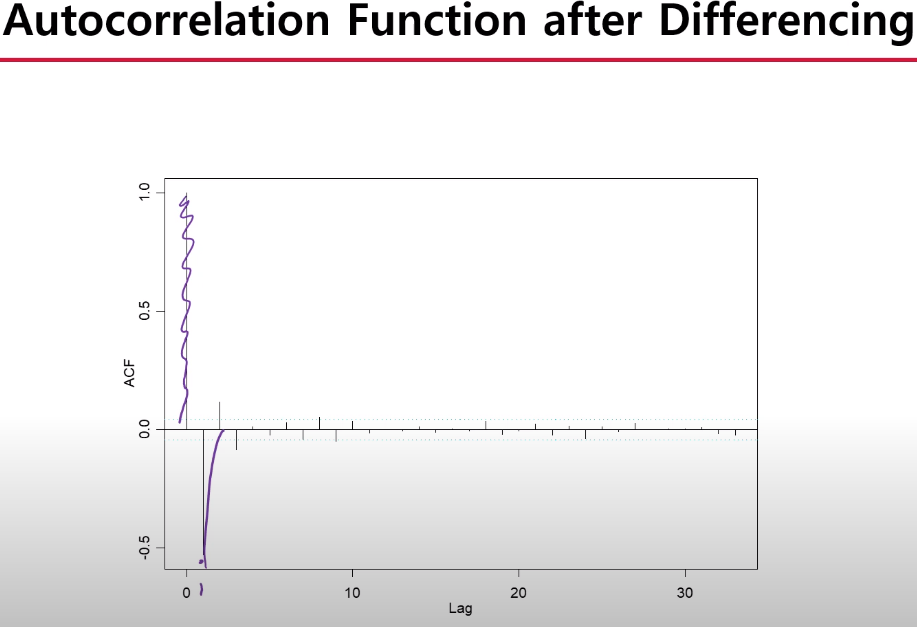


ACF의 Lag이 0일 때에는 자기 자신과의 관계이기 때문에 0은 빼고 봐야한다.

위 그래프는 천천히 떨어지는 모양이다. Stationary로 바꿔주기 위해 차분을 한다.



1차 차분을 해 보니 그래프가 이렇게 나온다. Stationary해 보이지만 확인을 위해 ACF를 그려본다.

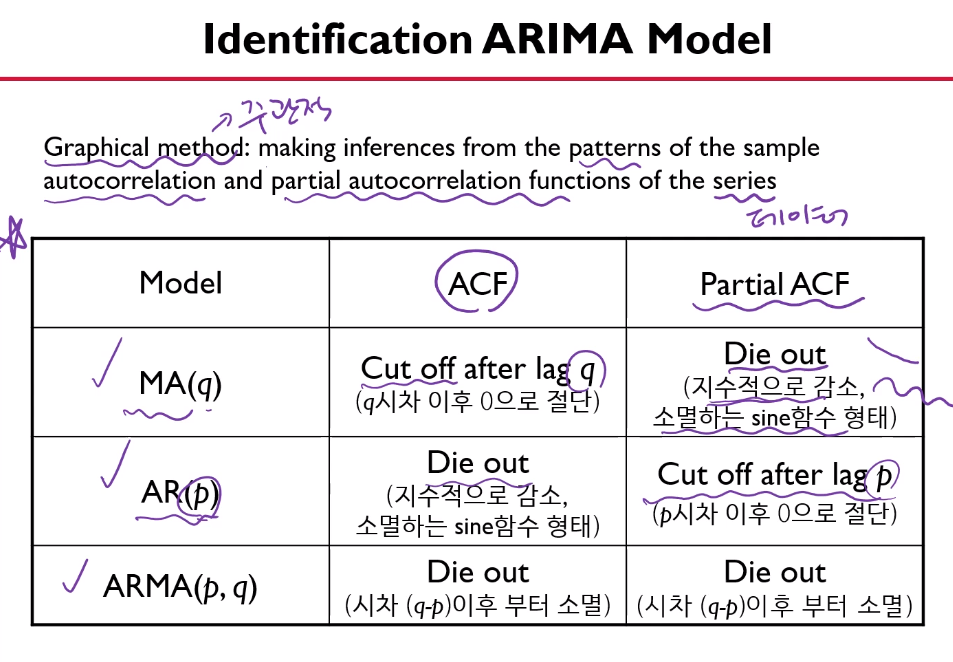


Stationary 해진 것 같이 보인다.

다음 단계로 시범적인 모델을 만들어본다.

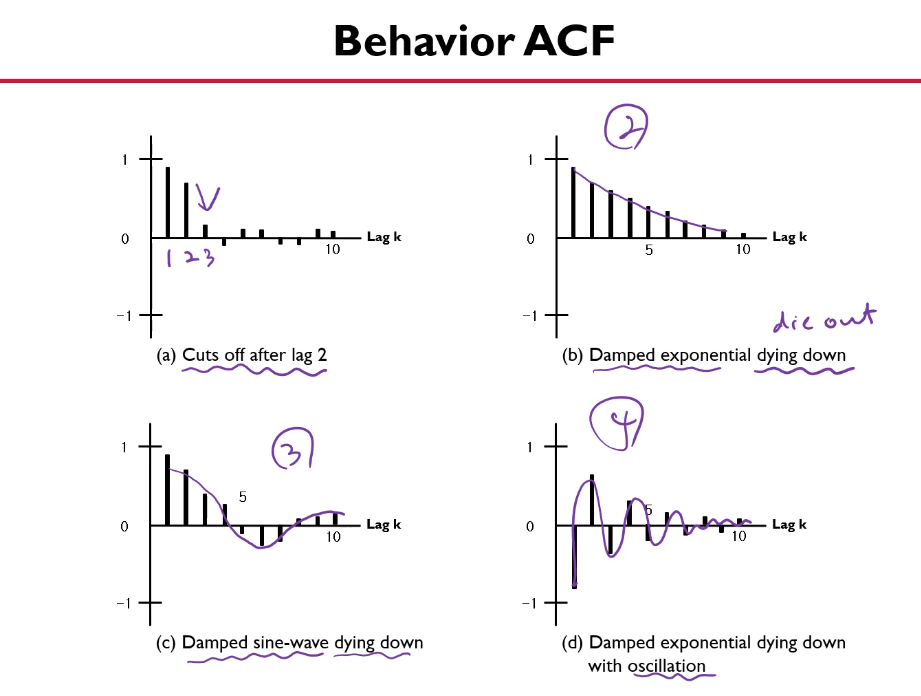
AR, MA, ARMA 중 무엇으로 할 것인지, 그리고 p, q는 얼마로 할 것인지 정한다.

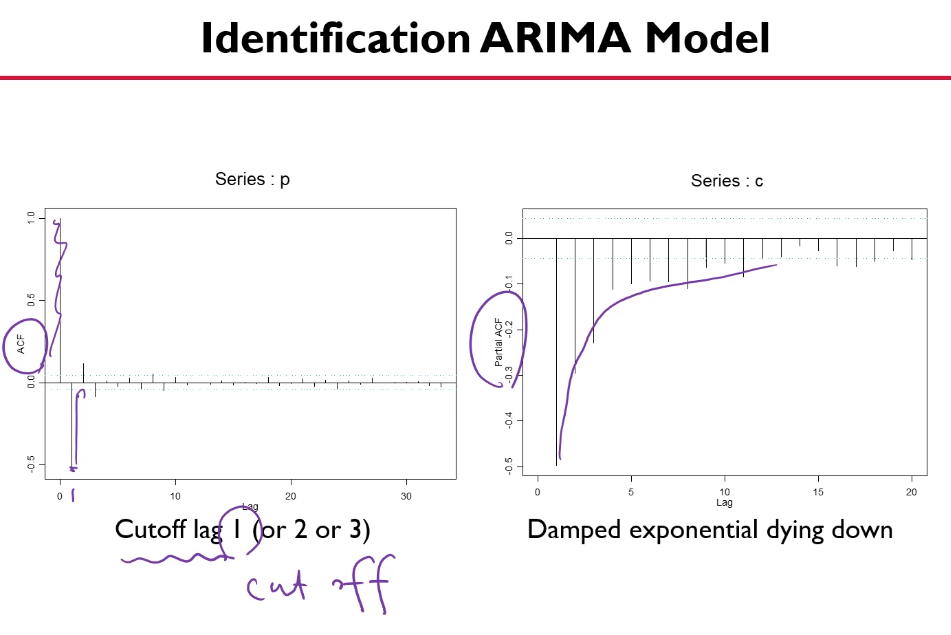
ACF, PACF의 패턴을 보고 결정한다.



ACF, PACF 그래프를 보고 절단면을 찾는 것이 상당히 주관적이다. (처음에 적당히 정해보고, 나중에 함수 사용해서 p, q를 바꿔가며 다 돌려본다.)

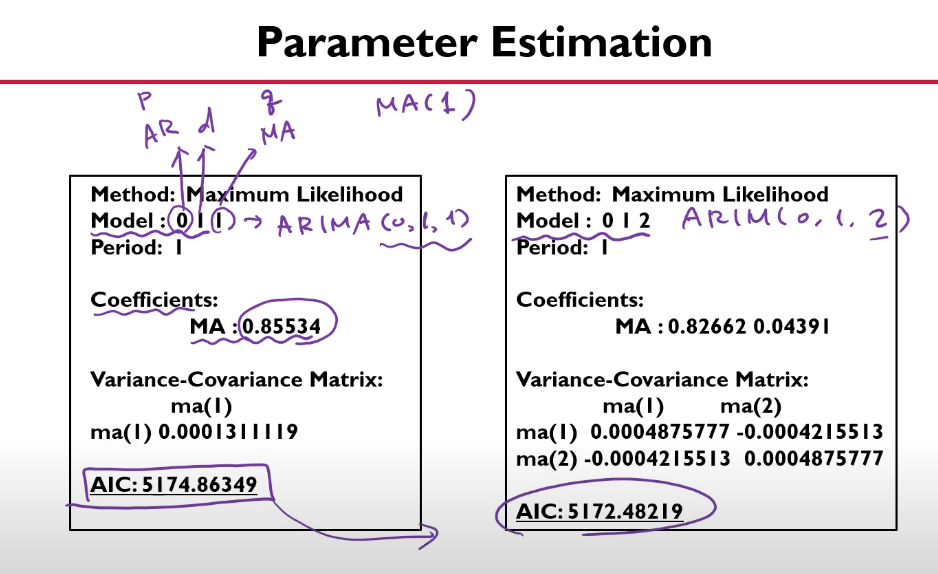
아래의 (b)~(d)의 그래프는 방식은 다르나 서서히 감소(dying down)하는 것으로 해석되고, (a)는 Lag 2 이후 확 감소하는 절단면을 보인다.



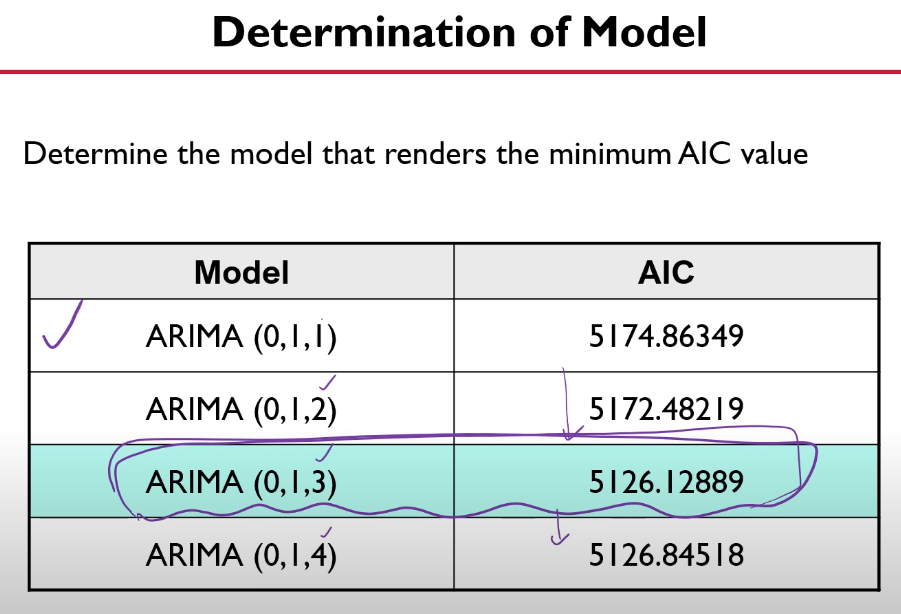


1차 차분한 데이터로 ACF, PACF 그래프를 그려보니 ACF Lag 1 이후 절단면이 보이고, PACF는 dying down하는 것으로 보여진다.

그래서 일단 임시 모델로 MA모델을 선택하고, ACF Lag 1 이후 절단면이므로 MA(1) 모델을 선택한다. 위 그래프는 1차 차분한 데이터를 사용한 것이므로 d=1 이다.

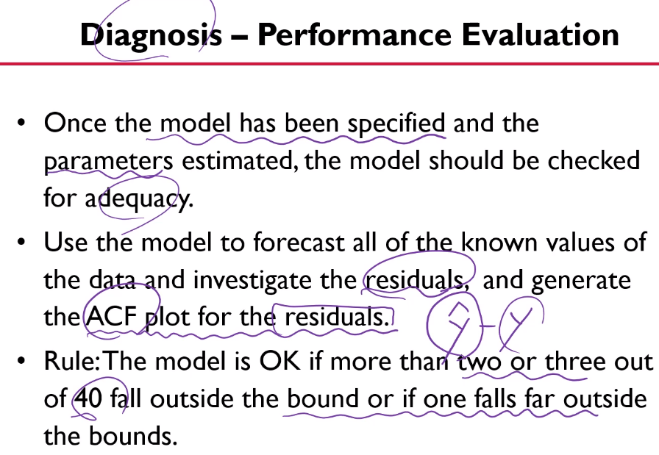


첫 번째 모델로 ARIMA(0,1,1) 모델을 만들어 보았다. 그리고 확정 모델은 아니므로 그 주변 모델로 ARIMA(0,1,2) 모델을 만들었다. 주변 모델을 더 만들어 봤다(아래)



일단은 ARIMA(0,1,3)이 AIC 5126.12889로 가장 좋다.

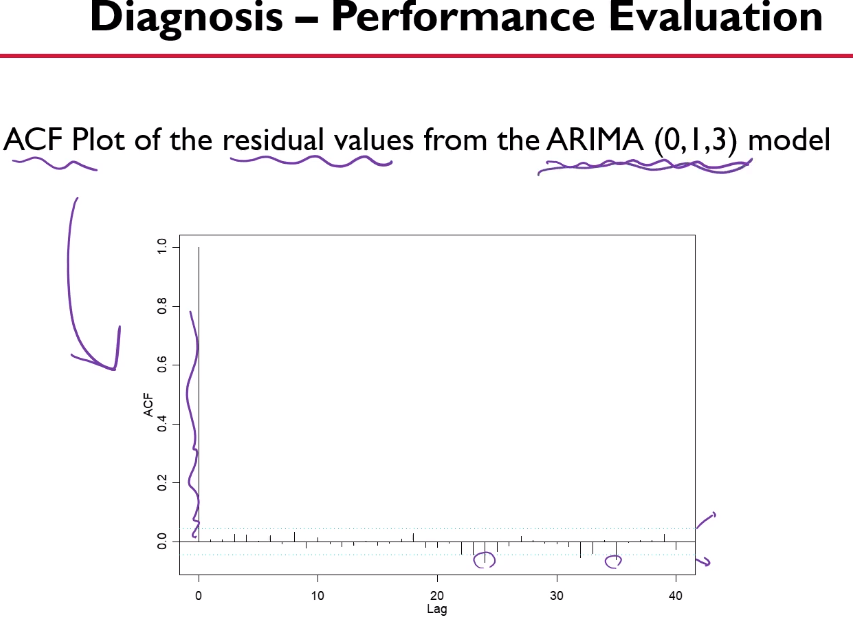
이 모델을 Evaluation 한다.



1) 모델을 만들었으면 적절한지 판단해야한다.

2) 모델로 예측을 해 보고, 실제 값과의 차이(residuals)로 ACF plot을 그려본다.

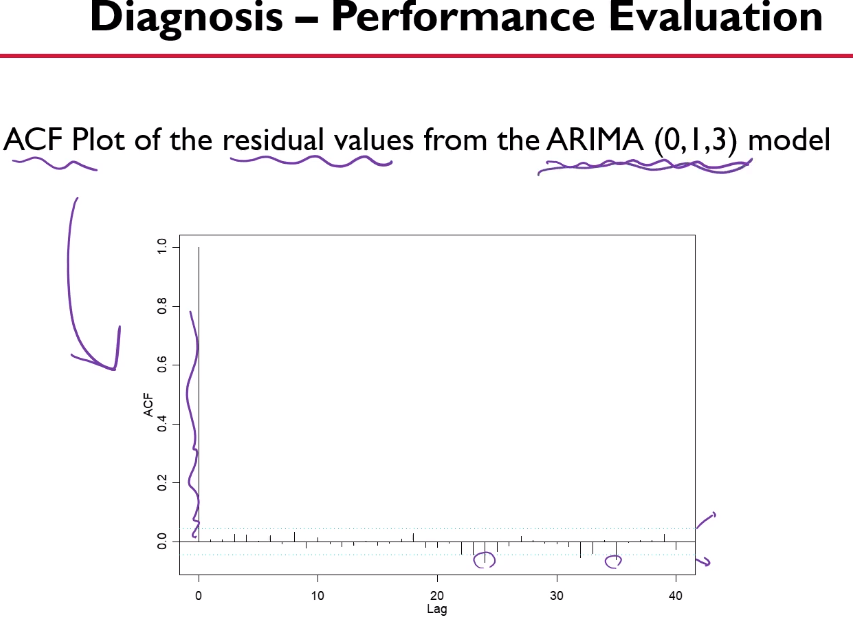
3) 범위 내에 대부분의 값이 있고, 40개 중에 2~3개 정도만 나간다면 괜찮은 모델이다.

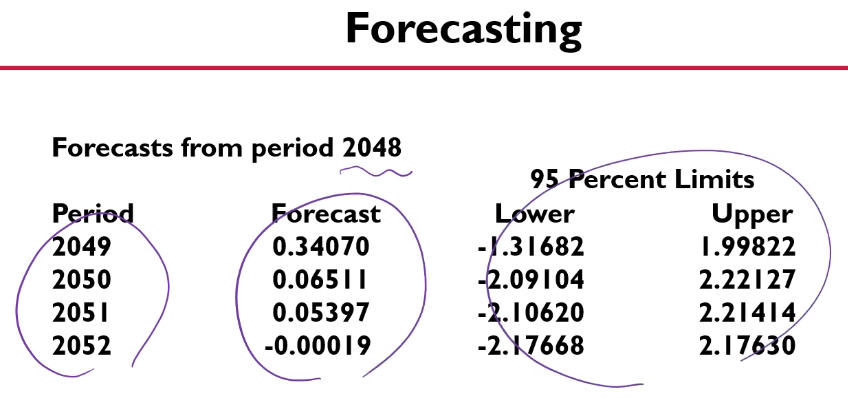


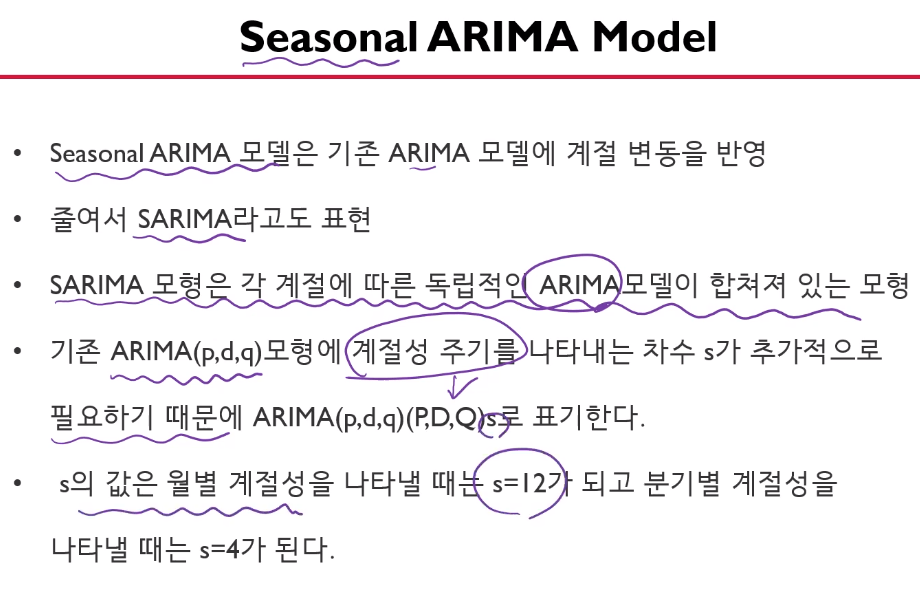
ARIMA(0,1,3) 모델의 residual 값으로 ACF plot을 그렸더니 두 개 값만 범위를 벗어났으므로, 괜찮은 모델이라고 판단한다.

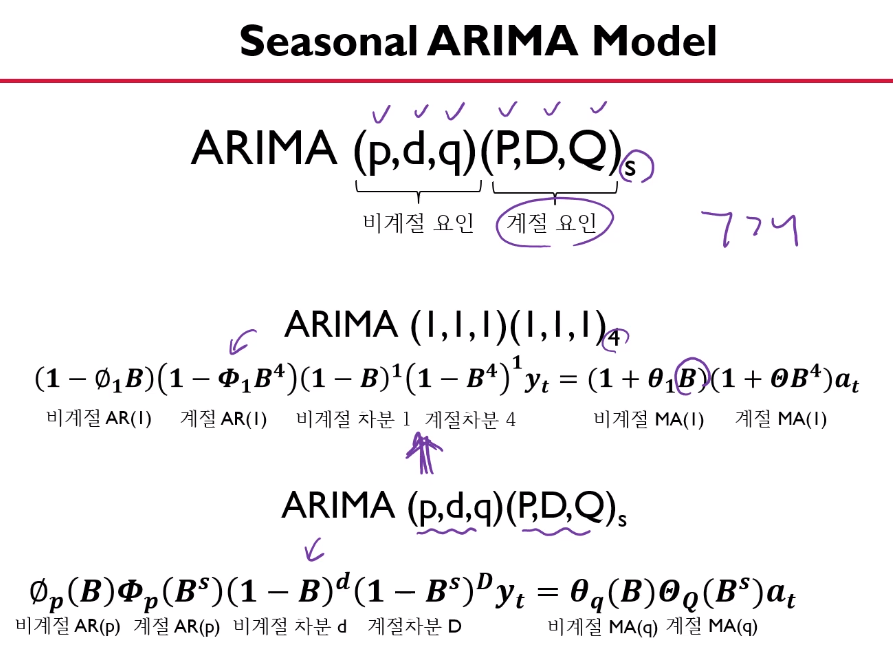
= 최종모델로 결정

이후 Forecasting 하면 된다.



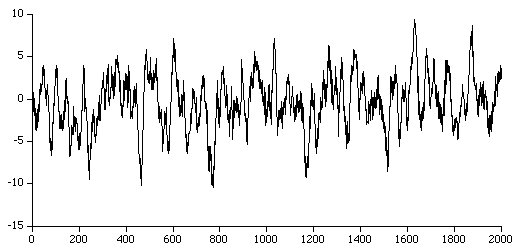






**3. 대표적인 시계열(Time Series)  데이터의 특성과 모형 소개**  
많은 연구자들이 시계열 정보를 크게 2가지로 쪼개어 규칙성을 가지는 패턴과 불규칙한 패턴의 결합이라고 여겨왔다. 따라서 시계열 모형은 전통적으로 이 두가지를 나누어서 개발되어 왔는데, 규칙성을 만드는 패턴을 또한 두가지로 쪼개어 이전의 결과와 이후의 결과 사이에서 발생하는 자기상관성(Autocorrelativeness)과 이전에 생긴 불규칙한 사건이 이후의 결과에 편향성을 초래하는 이동평균(Moving Average)현상으로 구분하고 있다. 한편, 많은 시계열 모형들이 불규칙한 패턴을 White Noise라고 칭하고 평균이 0이며 일정한 분산을 지닌 정규분포에서 추출된 임의의 수치라고 규정하고 있는데, 이러한 정규분포 가정은 모델의 해석을 전반적으로 편리하게 만들기 때문에 대부분의 통계 및 수리분석에서 채택되고 있다.  
  
익히 알려진 대표적인 시계열 모델로는 AR모형, MA모형, ARMA모형, ARIMA모형이 있으며, 거의 대부분 시중에 존재하는 모형들은 이러한 기본 모형들의 특수해 또는 개량/변형한 것들에 해당된다. 이 대표모델의 시계열 움직임은 또한 정규분포 성질을 가지기 때문에 해석에서도 여러모로 편리하다.

**A. 자기상관(Autocorrelation) - AR모형**  
자기상관이란 어떠한 Random Variable에 대해서 이전 의 값이 이후의 값에 영향을 미치고 있는 상황을 이야기한다. 예를 들면 이전에 값이 크면 이후에는 낮은 값이 나온다거나 하는 경향 따위를 말한다. 이러한 자기 상관은 바로 이전의 결과의 영향을 받을 수도 있지만, 드물게는 Delay가 발생하기도 한다. 우리가 실생활에서 가장 많이 발견하는 평균으로 돌아가려는 경향을 지닌 것들이다. 예를 들면 용수철을 원래 길이보다 길게 잡아당기면 이후는 반드시 용수철은 줄어드는 방향으로 반발하는 움직임을 보여 궁극적으로 원래 길이로 돌아가려고 한다.

[](https://m.blog.naver.com/bluefish850/220749045909)

많은 금융정보나 시계열 정보들이 자기 상관의 특징을 잘 가지고 있다. 이 데이터는 평균이 0라는 점 외에도 이전에 양수가 나오면 이후에는 음수가 나올 것이다(반대도 가능)는 일정한 패턴도 예상하게 만든다.

자기상관성을 시계열 모형으로 구성한 것을 AR 모형이라고 부르는데, 가장 간단한 형태가 바로 직전 데이터가 다음 데이터에 영향을 준다고 가정한 AR(1)모형이다.  
  
             **X(t)={a\*X(t-1)+c}+u\*e(t)**  
  
좀 쉽게 서술하자면, 시점 t에서 얻게 될 X(t)의 평균값은 시점 t-1에서 얻었던 X(t-1)의 값에 a를 곱하고 c를 더한 것과 같다는 뜻이다. 여기서 e(t)항은 white noise라고 부르며 평균이 0이고 분산이 1인 정규분포에서 도출된 random한 값이다. 즉, X(t) 값은 평균이 a\*X(t-1)+c이며 분산이 u인 정규 분포에서 도출되는 임의의 값이라는 뜻이다.  
  
만약 더 이전의 시점(P시점)을 모델에 넣고자 하면 AR(P) 모형이 되는 셈이다.

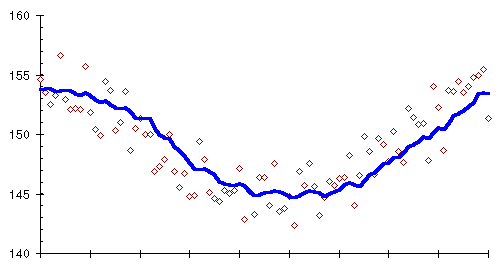
[https://mblogthumb-phinf.pstatic.net/20160609_250/bluefish850_14654603110167avVW_PNG/864bb5b2b097012272d52180553e16d4139d2d90.png?type=w800](https://m.blog.naver.com/bluefish850/220749045909)

AR(P)모형의 일반식

Seasonality 가 존재할 경우, 예측값 t-1 보다 더 큰 시차값을 사용 할 떄도 있지만, 일반적으로 trend 와 seasonality 가 존재 하지 않는 시계열 데이터분석시 사용한다.

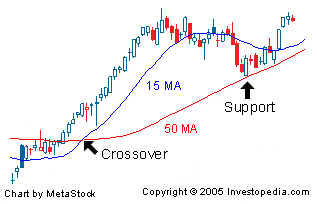
필요조건: univariate time series data, seasonality X, trend X, stationarity

**B. 이동평균(Moving Average) - MA모형**  
시간이 지날수록 어떠한 Random Variable의 평균값이 지속적으로 증가하거나 감소하는 경향이 생길 수 있다. 예를 들어 봄에서 여름이 될 수록 일반적으로 가계 전기 수요량은 증가하는 경향이 있고 여름에서 겨울로 갈 수록 감소하는 경향이 있다. 우리는 특별히 전기세가 크게 부담되지 않는다면 전월의 전기 사용량이 다음 월 전기 사용량에 상관을 주지는 않을 것이라고 가정할 수 있고, 아무래도 전기 사용량이 얼마가 될 것인지는 확실히 말하기 어렵기 때문에 이러한 경우 평균이동이 있는 시계열 데이터가 될 것이라고 생각해봄직하다.

[](https://m.blog.naver.com/bluefish850/220749045909)

데이터의 평균값 자체가 시간에 따라 변화하는 경향이 Moving Average이다.

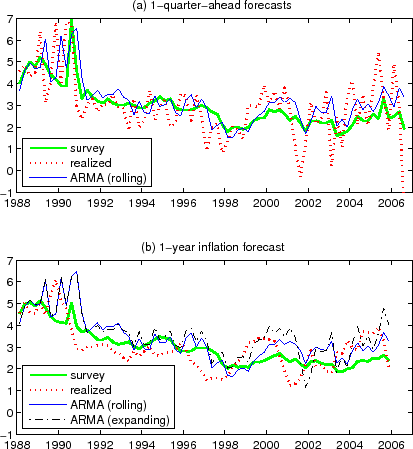
이동평균을 시계열 모형으로 구성한 것을 MA 모형이라고 부르는데, 가장 간단한 형태가 바로 직전 데이터가 다음 데이터에 영향을 준다고 가정한 MA(1)모형이다.  
  
             **X(t)={a\*e(t-1)+c}+u\*e(t)**  
  
좀 쉽게 서술하자면, 시점 t에서 얻게 될 X(t)의 평균값은 시점 t-1에서 발생한 error e(t-1)의 값에 a를 곱하고 c를 더한 것과 같다는 뜻이다. 여기서 즉, X(t) 값은 평균이 a\*e(t-1)이며 분산이 u인 정규 분포에서 도출되는 임의의 값이라는 뜻이다. AR(1)모형과 가장 결정적인 차이는 이전에 발생한 error가 중요한 것이지 결과적으로 X(t-1)이 무엇이엇는지는 중요하지 않다는 점이다.  
  
만약 더 이전의 시점(P시점)을 모델에 넣고자 하면 MA(P) 모형이 되는 셈이다.

[](https://m.blog.naver.com/bluefish850/220749045909)

증권가에서 기술적 분석을 할 때 가장 많이 쓰는 것이 MA모형이기도 하다. 예를 들어 최근 50일 평균값보다 최근15일 이동평균값이 커지면 주가가 치솟는다, 즉 골든크로스가 발생한다 같은 접근 말이다.

필요조건: univariate time series data, seasonality X, trend X, stationarity

**C. ARMA 모형**  
ARMA모형은 Autoregressive Moving Average라는 뜻으로,  AR모형과 MA모형을 합친 것으로, 연구 기관에서 가장 선호되는 모델이기도 하다. (통상 ARMA(2,2) 정도의 모델이면 충분히 양질의 데이터를 얻는다.)  
  
가장 단순한 형태인 ARMA(1,1)모형은 아래와 같다.  
  
             **X(t)={a\*X(t-1)}+{b\*e(t-1)}+c+u\*e(t)**  
  
필자가 가장 사랑하는  ARMA(2,2)모형은 아래와 같다.  
  
             **X(t)={a1\*X(t-1)+a2\*X(t-2)}+{b1\*e(t-1)+b2\*e(t-2)}+c+u\*e(t)**  
  
따로 ARMA(p,q)모형을 표시하지 않더라도 어떤 형태인지 쉽게 짐작이 되리라 생각한다.

[](https://m.blog.naver.com/bluefish850/220749045909)

여론조사결과(X)와 실제 발생량의 차이(e)가 가지는 패턴을 이용해서 보다 나은 추정을 할 수 있지 않을까 하는 시도를 해볼 수 있다. 데이터를 보면 여론조사결과 자체보다 ARMA모델의 내삽 결과가 확실히 더 나은 결과를 보이긴 한다. 허나 이 또한 예측력이 그렇게 강해 보이진 않아서 외삽은 시도조차 할 필요가 없어 보인다.

**D. ARIMA 모형**  
ARIMA모형은 Autoregressive Integrated Moving Average라는 뜻으로,  ARMA모형이 과거의 데이터들을 사용하는 것에 반해 ARIMA 모형은 이것을 넘어서서 과거의 데이터가 지니고 있던 '추세(Momentum)'까지 반영하게 된다. 즉, Correlation 뿐 아니라 Cointegration까지 고려한 모델이다.

**ARIMA의 경우 시계열의 비정상성(Non-stationary)을 설명하기 위해 관측치간의 차분(Diffrance)을 사용한다는 차이점이 있다.**

**현실에 존재하는 시계열자료는 불안정(Non-stationary)한 경우가 많다. 그런데 AR(p), MA(q) 모형이나, 이 둘을 합한 ARMA(p, q)모형으로는 이러한 불안정성을 설명할 수가 없다.**

**따라서 모형 그 자체에 이러한 비정상성을 제거하는 과정을 포함한것이 ARIMA모형이며 ARIMA(p, d, q)로 표현한다. ARMIA(p, d, q)는 AR, MA, ARMA를 모두 표현할 수 있다.**

Cointegration은 Correlation보다 설명하기가 더 어려운데, 가장 단순하게 설명하자고 하면 Correlation은 서로간에 선형관계를 설명하는 것이라면 Cointegration은 추세관계를 설명한다, 즉 cointegration인 시점이 고려되지 않으면 성립하지 않기 때문에 시계열 데이터에만 쓰이는 개념이다.  
  
선형관계  
두 변수 X-Y간에 correlation이 0보다 크면 X가 큰 값이 나올 때 Y값도 큰 값을 가진다.  
두 변수 X-Y간에 correlation이 0보다 작으면 X가 큰 값이 나올 때 Y값은 작은 갑을 가진다.  
  
추세관계  
두 변수 X-Y간에 cointegration이 0보다 크면 X의 값이 이전 값보다 증가하면 Y값도 증가한다.  
두 변수 X-Y간에 cointegration이 0보다 작으면 X의 값이 이전 값보다 증가하면 Y값은 감소한다.  
  
**<추가적인 Example>**  
만약 Correlation이 0보다 작고 cointegration은 0보다 큰 관계라면,  
X가 큰 값이며 증가하는 추세에 있는 경우 Y는 현재 작은 값이나 빠르게 증가하는 추세로 반응하게 된다.  
  
만약 Correlation이 0보다 크고 cointegration은 0보다 작은 관계라면,  
X가 큰 값이며 증가하는 추세에 있는 경우 Y는 현재 큰 값이나 빠르게 감소하는 추세로 반응하게 된다.  
  
Correlation과 Cointegration의 개념이 다소 혼란스럽겠지만, 간단히 현재의 관계와 추세의 관계라는 요소로 분리해서 생각하다 보면 아주 어렵게 이해되진 않으리라 생각한다. 어쨋든, ARIMA 모형은 추세 또한 고려할 수 있기 때문에 momentum을 중요하게 보는 분석가들에게 아주 유용하다.  
  
**Note:**ARIMA 모델은 자기 자신의 추세만 고려한다.white noise의 추세는 고려하지 않는다. 올바른 모델의 white noise는 추세가 존재하면 안되기 때문이다. 즉, Autoregressive는 자기 자신의 correlation을 말하는 것이고 Integrated 모델은 자기 자신의 cointegration을 말하는 것이다.  
  
  
결론적으로, 가장 단순한 형태인 ARIMA(1,1,1)모형은 아래와 같다.  
  
             **a\*{X(t)-X(t-1)}={b\*X(t-1)}+{c\*e(t-1)}+d+u\*e(t)**  
  
즉 X의 추세는 이전 X의 값과 이전의 white noise의 결과의 영향을 받게 된다는 뜻이다.  
식을 조금 다르게 전개하면 아래와 같은 형태가 된다.  
  
             **X(t)=[X(t-1)+{b\*X(t-1)}+{c\*e(t-1)}+d+u\*e(t)]/a**  
  
그런데 이 모형은 자세히 보면 ARMA(1,1)과 다를 바가 없다. 그래서 ARIMA(1,1,1)은 사실상 쓰이지 않는다. 그러니 더 복잡한 모델인 ARIMA(1,2,1)를 소개하도록 하겠다.  
  
             **a\*[{X(t)-X(t-1)}-{X(t-1)-X(t-2)}]={b\*X(t-1)}+{c\*e(t-1)}+d+u\*e(t)**  
  
이는 X를 2차 미분한 값에 대한 모델이다. 추세의 추세...라는 표현은 좀 이상하고, 데이터가 확실하게 모멘텀 성향을 지닌다고 가정했을 때 모멘텀의 변화를 모델링한 것이라고 보는 것이 맞겠다. 즉 이 모델은 모멘텀의 변화는 X의 이전 값과 이전에 발생한 white noise에 의해 결정된다는 것을 표현한 것이다.  
  
마찬가지로 X(t)에 대해서만 전개하면 아래와 같은 수식이 되며, 이는 사실 ARMA(2,1)과 매우 유사하다.  
  
             **X(t)=(2+b/a)\*X(t-1)+X(t-2)+(c/a)\*e(t-1)+(d/a)+(u/a)\*e(t)**  
  
수식을 보면서 조금 답답함을 느꼈을 것이고, ARMA모델과 많이 비슷하다는 생각도 했을 것이다. 많은 학자들도 비슷하게 생각하기 때문에 ARIMA모델을 선호하지 않고 ARMA모델만 택하는 경우도 많으며, 필자 또한 ARIMA모델을 선호하지 않는다. 특히 추세의 일관성이나 유의미성이 크지 않은 데이터의 경우 ARIMA 모형은 ARMA모형보다 모델의 타당성이 떨어지기도 하다는 점에서 ARIMA 모델을 도입하는 경우가 그리 흔하게 발견되지는 않는 것 같다.  
  
이번 편에서 수식이 자주 등장해서 독자들이 흥미를 잃지 않았을까 고민이 되지만 다음 편에서는 보다 실용적인 내용인, '우수한 모델을 발굴하는 방법'을 직관적인 그림들을 통해 설명하도록 하겠다.

필요조건: univariate time series data, seasonality X

**정상성(Stationarity) 확보**

정상성을 갖는 시계열 데이터는 시점마다 유사하게 동작하는 특징이 있어 안정된 분포를 갖는다.

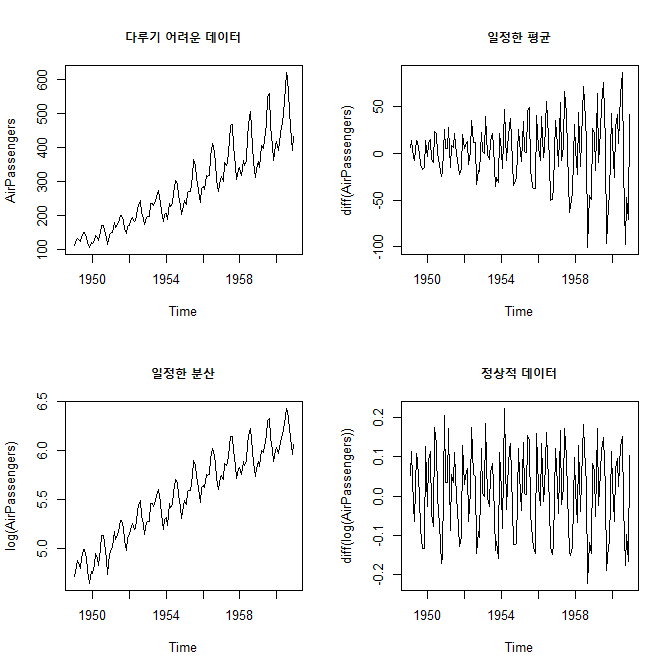
데이터가 정상성을 가진다는 것은 평균과 분산이 안정되어 있어서 분석하기 쉽다는 의미가 된다. 데이터가 정상성을 가지지 않으면 분석이 어렵기 때문에 정상성을 갖도록 만드는 전처리를 하게 된다. 보통 평균이 일정하지 않으면 차분을 취하고, 분산이 일정하지 않으면 변환을 취한다.

약정상성은 시간에 따라 일정한 평균, 분산, 공분산을 갖는 확률과정으로 정의된다. 즉, 시계열 데이터 $Y\_1 , Y\_2 , $이 다음을 세가지 조건을 충족하면 된다.

* 시계열 YtYt 모든 시점 tt에 대해 평균이 μμ로 같다.
* 시계열 YtYt 모든 시점 tt에 대해 분산이 σ2σ2로 같다.
* 시계열 YtYt, YsYs 모든 hh에 대해 공분산이 |t−s|=h|t−s|=h로 같다; 예를 들어, 6단위 만큼 시점차이가 나는 경우 Cov(Y1,Y7)=Cov(Y11,Y17)Cov(Y1,Y7)=Cov(Y11,Y17).

차분(differencing)은 시계열의 수준에서 나타나는 변화를 제거하여 시계열의 평균 변화를 일정하게 만드는데 도움이 될 수 있습니다. 결과적으로 추세나 계절성이 제거(또는 감소)됩니다.

다음 네가지 그래프를 보자.



**다루기 어려운 데이터** : 안 그래도 복잡한 등락을 반복하는 모양을 보이는데 갈수록 값이 커지는 경향이 있을뿐만 아니라 그 정도도 심해지고 있다. 이후의 동향을 짐작하는 것은 어렵지 않지만 수식적으로 깔끔하게 나타내기엔 몹시 어려워 보인다.

**일정한 평균** : 고정된 0을 중심으로 점점 퍼져나가는 모양이기 때문에 어렵지는 않지만 그 폭이 넓어지는 것이 문제다.

**일정한 분산** : 각각의 패턴은 일정한 모양을 갖추었지만 시간에 따라 값 자체가 증가하는 경향을 설명할 수 있어야한다.

**정상적 데이터** : 평균과 분산이 일정하므로, 반복되는 등락만 잘 설명하면 된다.

### 백색잡음(White Noise)

백색잡음(White Noise)은 백색잡음은 고정된 평균, 분산을 갖고 시계열 상관은 없는 시계열 데이터

# **서론**

시계열 분석(Time series analysis)이란,  
독립변수(Independent variable)를 이용하여 종속변수(Dependent variable)를 예측하는 일반적인 기계학습 방법론에 대하여 시간을 독립변수로 사용한다는 특징이 있다. 독립변수로 시간을 사용하는 특성때문에 분석에 있어서 일반적인 방법론들과는 다른 몇가지 고려가 필요하다.

본 포스트에서는 시계열 분석(혹은 예측)에 있어서 가장 널리 사용되는 모델중 하나인 ARIMA에 대해 알아보고 Python을 통해 구현해본다.  
본 포스트에서는

* ARIMA의 간단한 개념
* ARIMA 모델의 파라미터 선정 방법
* 실제 데이터와 ARIMA 모델을 이용한 미래 예측

등 과 관련된 내용을 다룬다.

# **ARIMA(Autoregressive Integrated Moving Average)**

ARIMA는 Autoregressive Integrated Moving Average의 약자로, Autoregressive는 자기회귀모형을 의미하고, Moving Average는 이동평균모형을 의미한다.

즉, ARIMA는 자기회귀와 이동평균을 둘 다 고려하는 모형인데, ARMA와 ARIMA의 차이점은 ARIMA의 경우 시계열의 비정상성(Non-stationary)을 설명하기 위해 관측치간의 차분(Diffrance)을 사용한다는 차이점이 있다.  
ARMA와 ARIMA 외에도 ARIMAX 등의 방법도 있는데, 이는 본 포스트에서 살펴보지 않는다.

* **AR**: 자기회귀(Autoregression). 이전 관측값의 오차항이 이후 관측값에 영향을 주는 모형이다. 아래 식은 제일 기본적인 AR(1) 식으로, theta는 자기상관계수, epsilon은 white noise이다. Time lag은 1이 될수도 있고 그 이상이 될 수도 있다.  
  eq_ar1
* **I**: Intgrated. 누적을 의미하는 것으로, 차분을 이용하는 시계열모형들에 붙이는 표현이라고 생각하면 편하다.
* **MA**: 이동평균(Moving Average). 관측값이 이전의 연속적인 오차항의 영향을 받는다는 모형이다. 아래 식은 가장 기본적인 MA(1) 모형을 나타낸 식으로, beta는 이동평균계수, epsilon은 t시점의 오차항이다.  
  eq_ma1

현실에 존재하는 시계열자료는 불안정(Non-stationary)한 경우가 많다. 그런데 AR(p), MA(q) 모형이나, 이 둘을 합한 ARMA(p, q)모형으로는 이러한 불안정성을 설명할 수가 없다.  
따라서 모형 그 자체에 이러한 비정상성을 제거하는 과정을 포함한것이 ARIMA모형이며 **ARIMA(p, d, q)**로 표현한다.  
eq_arima

* eq_arima_mu
* eq_arima_ar
* eq_arima_ma
* eq_arima_d0
* eq_arima_d1
* eq_arima_d2

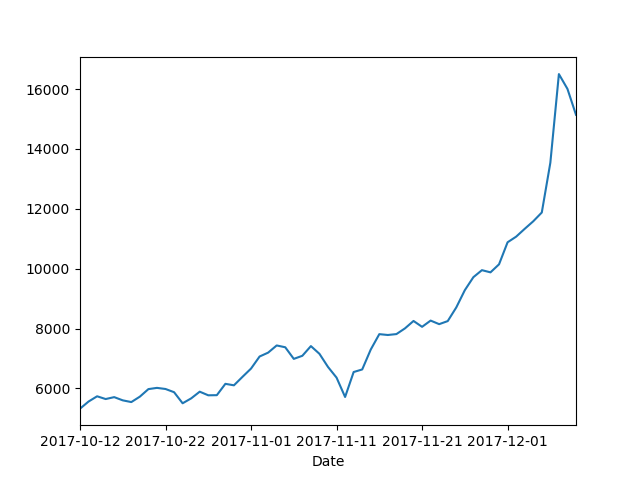
위와 같은 특징에 따라 ARMIA(p, d, q)는 AR, MA, ARMA를 모두 표현할 수 있다.

* AR(p) = ARIMA(p, 0, 0)
* MA(q) = ARIMA(0, 0, q)
* ARMA(p, q) = ARIMA(p, 0, q)

# **데이터**

본 포스트에서 ARIMA를 이용한 예측에 사용할 데이터는 [Blockchain Luxembourg S.A](https://blockchain.info/ko/charts/market-price?timespan=60days)에서 Export한 최근 60일간의 비트코인 시세와 관련된 자료이다.  
CSV파일로 Export했으며, Column name만 수동으로 추가해 주었다.

| **Date** | **Price** |
| --- | --- |
| 2017-10-12 | 5325.130683333333 |
| 2017-10-13 | 5563.806566666666 |
| 2017-10-14 | 5739.438733333333 |
| … | … |

미화(USD)기준 가격 변동임에도, 며칠전 일련의 사태(비트코인 플래티넘 등)로 폭등했던 흔적이 남아있다.  


import pandas as pd

**series** = pd.read\_csv('market-price.csv', header=0, index\_col=0, squeeze=True)

series.plot()

# **ARIMA 모수 설정**

ARIMA의 모수는 크게 3가지가 있다. **AR모형의 Lag을 의미하는 p, MA모형의 Lag을 의미하는 q, 차분(Diffrence)횟수를 의미하는 d** 가 그것이다. 보통은 p, d, q의 순서로 쓴다.  
통상적으로 p + q < 2, p \* q = 0 인 값들을 많이 사용한다.

여기서 p \* q = 0 이라 하면, 두 값중 하나는 0이라는 이야기이다. **ARIMA는 AR모형과 MA모형을 하나로 합쳤다면서 둘 중 하나의 모수가 0인건 또 무슨소리?** 라고 할지 모르겠지만, 실제로 대부분의 시계열 자료에서는 하나의 경향만을 강하게 띄기 때문에, 이렇게 사용하는것이 더 잘 맞는다고 한다.

그렇다면, p와 d, q는 어떻게 정해야 할까? Rules of thumb이긴 하지만 ACF plot와 PACF plot을 통해 AR 및 MA의 모수를 추정할 수 있다.

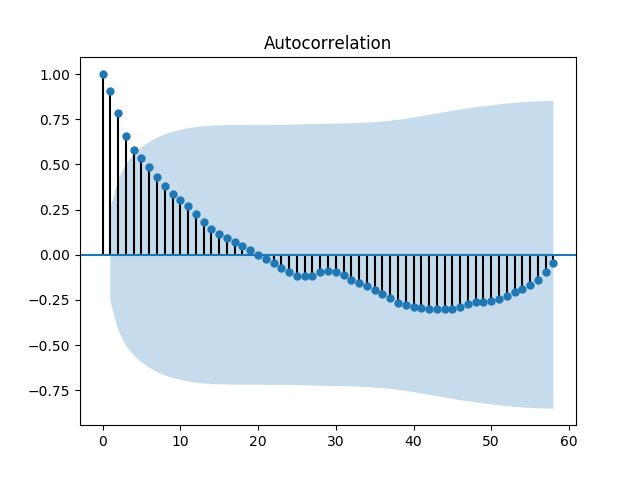
* ACF(Autocorrelation function) : Lag에 따른 관측치들 사이의 관련성을 측정하는 함수  
  eq_acf
* PACF(Partial autocorrelation function) : k 이외의 모든 다른 시점 관측치의 영향력을 배제하고eq_yt와 eq_ytk 두 관측치의 관련성을 측정하는 함수  
  eq_pacf

시계열 데이터가 AR의 특성을 띄는 경우, ACF는 천천히 감소하고 PACF는 처음 시차를 제외하고 급격히 감소한다.  
반대로, MA의 특성을 띄는 경우 ACF는 급격히 감소하고 PACF는 천천히 감소한다.  
급격히 감소하는 시차를 각 AR과 MA 모형의 모수(p, q)로 사용할 수 있다. 또한 데이터를 차분하여 ACF 및 PACF 계산함으로써 적절한 차분횟수까지 구할 수 있다 ([Robert Nau @Duke university](http://people.duke.edu/~rnau/411home.htm)).

본 포스트에서는 python package인 statsmodels를 사용하여 ACF 및 PACF를 계산했으며, 비트코인 가격 자료의 ACF 및 PACF는 다음과 같다.

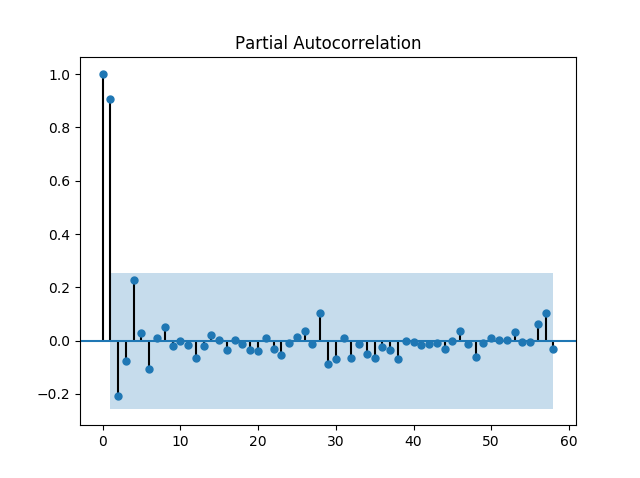
import matplotlib.pyplot as plt

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

plot\_acf(series)

plot\_pacf(series)

plt.show()



ACF를 보면 20의 Time lag을 기준으로 자기상관이 양에서 음으로 변동한다. 또한 PACF는 1의 Time lag에서 약 0.9를 보이고 이후에 급격히 감소한다. 따라서 p=0, q=1이 적당하다고 추측할 수 있다.

적절한 차분 차수의 계산을 위해 우선 1차 차분을 하고, ACF 및 PACF를 다시 계산한다.

import matplotlib.pyplot as plt

import pandas as pd

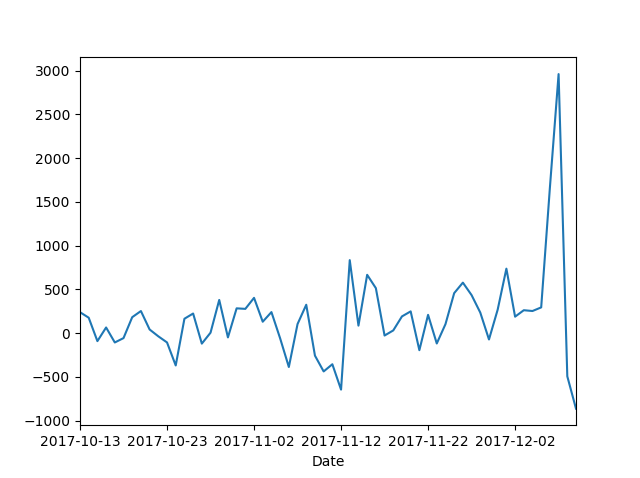
from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

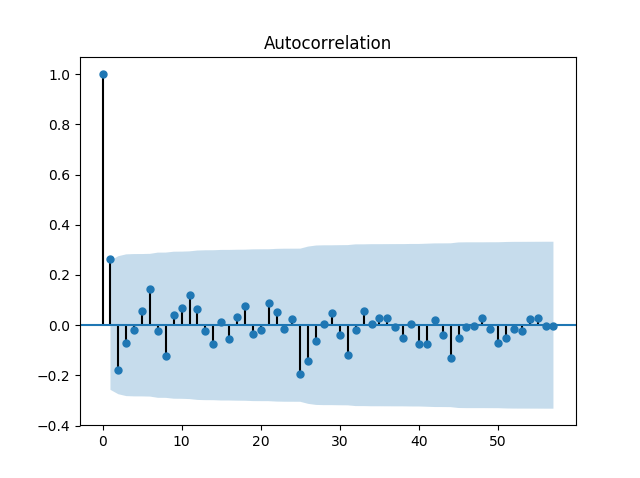
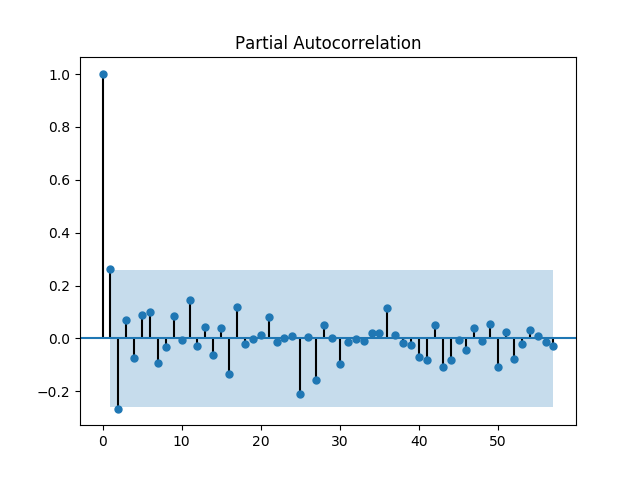
diff\_1=series.diff(periods=1).iloc[1:]

diff\_1.plot()

plot\_acf(diff\_1)

plot\_pacf(diff\_1)

plt.show()

차분 결과를 보니, 12월 초에 있었던 급등락이 더 도드라져보인다…2500만원에 물리신분들께 애도..

아무튼, 차분이후의 ACF와 PACF를 보니, 시계열이 정상상태(Stationary)를 보이는것으로 생각되므로, 1차차분 만으로 충분할것같다.  
따라서 본 데이터에는 ARIMA(0,1,1)을 사용하기로 한다.

# **모형구축**

ARIMA(0,1,1)을 이용하여 모형의 Parameter를 추정하고, 결과를 확인한다.

from statsmodels.tsa.arima\_model import ARIMA

model = ARIMA(series, order=(0,1,1))

model\_fit = model.fit(trend='c',full\_output=True, disp=1)

print(model\_fit.summary())

ARIMA Model Results

==============================================================================

Dep. Variable: D.Price No. Observations: 58

Model: ARIMA(0, 1, 1) Log Likelihood -442.285

Method: css-mle S.D. of innovations 495.214

Date: Wed, 13 Dec 2017 AIC 890.570

Time: 13:10:49 BIC 896.751

Sample: 10-13-2017 HQIC 892.977

- 12-09-2017

===================================================================================

coef std err z P>|z| [0.025 0.975]

-----------------------------------------------------------------------------------

const 166.9630 91.874 1.817 0.075 -13.107 347.033

ma.L1.D.Price 0.4200 0.131 3.199 0.002 0.163 0.677

Roots

=============================================================================

Real Imaginary Modulus Frequency

-----------------------------------------------------------------------------

MA.1 -2.3811 +0.0000j 2.3811 0.5000

-----------------------------------------------------------------------------

‘P > z’ 값이 일반적으로 학습의 적정성을 위해 확인되는 t-test값이다. 즉, p value 0.05수준에서 보면 MA(1)의 계수는 유효하고, 모형의 Constant는 유효하지 않다…(슬픔)

따라서, 위 코드에서 model.fit()의 파라미터중 trend=’c’가 아니라 ‘nc’로 설정해주어야 하는게 옳다.

from statsmodels.tsa.arima\_model import ARIMA

model = ARIMA(series, order=(0,1,1))

model\_fit = model.fit(trend='nc',full\_output=True, disp=1)

print(model\_fit.summary())

ARIMA Model Results

==============================================================================

Dep. Variable: D.Price No. Observations: 58

Model: ARIMA(0, 1, 1) Log Likelihood -443.856

Method: css-mle S.D. of innovations 508.665

Date: Wed, 13 Dec 2017 AIC 891.712

Time: 14:08:16 BIC 895.833

Sample: 10-13-2017 HQIC 893.318

- 12-09-2017

===================================================================================

coef std err z P>|z| [0.025 0.975]

-----------------------------------------------------------------------------------

ma.L1.D.Price 0.4514 0.123 3.657 0.001 0.209 0.693

Roots

=============================================================================

Real Imaginary Modulus Frequency

-----------------------------------------------------------------------------

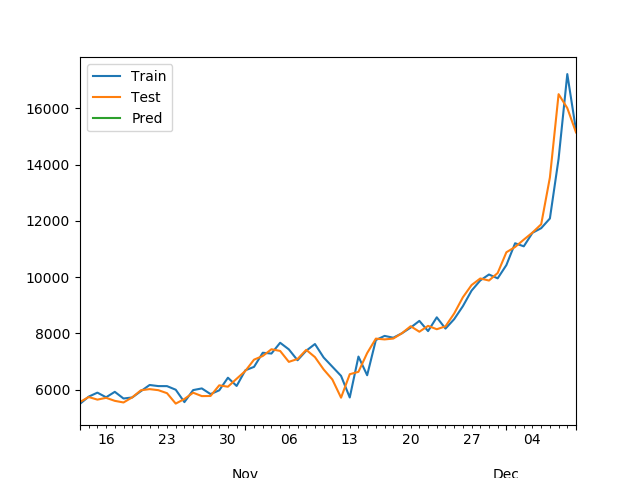
MA.1 -2.2152 +0.0000j 2.2152 0.5000

-----------------------------------------------------------------------------

# 예측

constraint가 없는 모형으로 fitting하고 나니, MA(1)의 t-test값이 0.001로 더 좋아졌다.  
이제, 모형을 통해 예측된 값을 보자.

model\_fit.plot\_predict()

  
fitting이 잘 된것으로 보인다.  
또한, 앞으로의 값을 예측하기 위해서는 forecast method를 사용할 수 있다. 코드에서 steps는 예측할 개수를 의미한다.

fore = model\_fit.forecast(steps=1)

print(fore)

위 코드의 결과는 아래와 같은데, 순서대로 예측값, stderr, upper bound, lower bound 이다.  
(array([ 15061.16108756]), array([ 508.66521867]), array([[ 14064.19557878], [ 13303.94590548]]))  
즉, 가장 즁요한 예측값은 15061달러인데, 학습에 쓰인 데이터가 12월 9일까지이므로, ARIMA모형은 12월 10일의 비트코인 가격을 15,061달러로 예측하였다.

참고로, 같은 사이트에서 획득한 12월 10일의 실제 비트코인 가격은 아래와 같다.  
