개요

개해

- 시계열(time series) 데이터는 관측치가 시간적 순서를 가지게 된다. 일정 시점에 조사된 데이터는 횡단(cross-sectional) 자료라 한다.
- KOSPI 주가, △△기업 월별 매출액, 소매물가지수, 실업률, 환율 등이 시계열 자료이다.
- 데이터 표현 : $\{Y_t; t = 1,2,...,T\}$, 횡단자료의 시점 t에서의 관측 데이터는 y_t 기호를 사용한다.
- 시간 t 는 시각을 나타내는 기호로 시, 분, 초, 일, 주, 월, 분기, 년으로 시각 순서대로 정렬

시계열 데이터 분석 목적

- 가장 중요한 목적은 미래 값을 예측 : trend analysis, smoothing, decomposition, ARMA model
- 시스템 시계열 데이터 이해와 특성 파악 : spectrum analysis, intervention analysis, transfer function analysis
- 시계열 데이터의 특성을 파악 : 경향(trend), 주기(cycle), 계절성(seasonality), 불규칙성(irregular) 등
- 데이터의 과거 흔적(정보)을 활용하여 미래 값을 예측하는 것이다. 향후 일주일간 주가 예측, 다음 달 매출액 예측 등

역사

- 17세기 태양의 흑점 자료나 밀 가격 지수 변동을 나타내는 함수로 Sine, Cosine 곡선을 이용하였다.
- Yule(1926)은 ARMA에 대한 개념을 제시하였고 Walker(1937)는 ARMA모형을 제안하였다. 이 기간동안 주기적 변동을 제거하기 위하여 이동 평균법이 제안되었다.
- ARMA 모형에 대한 추정은 Durbin(1960), 그리고 Box & Jenkins(1970)에 의해 이루어졌다. 'Time Series Analysis' 고전 책 발간
- Holt(1957)는 지수 평활법(exponential smoothing)을, Winter(1960)는 계절성(seasonal) 지수 평활법을 제안하였다.
- 미국 Bureau of the Census는 경기지수에 대한 계절 변동 분해방법으로 X-11을 제안하였다(1967년).
- X-11은 이동 평균 개념을 사용하므로 초기 관측치와 마지막 관측치를 사용할 수 없는 문제점을 안고 있다. 해결책으로 1975년 캐나다는 X11-ARMA 방법을 제안하였다. 우리나라는 현재 캐나다 방법을 사용하고 있다.
- Box-Jenkins 모델에서 기인한 시계열 개발의 또 다른 행은 ARCH(AutoRegressive Conditional Heteroscedasticity) 및 GARCH(General) 모델과 같은 비선형 일반화입니다. 이 모델을 사용하면 매개 변수화 및 비상수 분산예측이 가능하다. 따라서 이러한 모델은 금융 시계열에 매우 유용합니다.

time plot 시계열 데이터

Pandas 주가 데이터 불러오기

Yahoo 제공 주가 데이터 가져오기 위한 모듈 설치

```
!pip install finance-datareader
```

이전에 사용하던 이 모듈이 웬일인지 2020.10.01 현재 제대로 설치되지 않아 다음 모듈 활용

```
from pandas_datareader import data
import pandas_datareader.data as stock
import pandas as pd
```

기업 코드 명부 가져오기 및 활용 방법

```
stock_info=pd.read_html('http://kind.krx.co.kr/corpgeneral/corpList.do?method=download&searchType=13', header=0)[0]
stock_info.head(3)
```

	회사명	종목코드	업종	주요제품
0	DSR	155660	1차 비철금속 제조업	합섬섬유로프
1	GS글로벌	1250	상품 종합 도매업	수출입업(시멘트,철강금속,전기전자,섬유,기계화학),상품중개,광업,채석업/하수처리 서...
2	HDC현대산업개발	294870	건물 건설업	외주주택, 자체공사, 일반건축, 토목 등

```
name=input('주식 코드 기업명 ?') #find 종목코드 및 정보
stock_info[stock_info['회사명'].str.contains(name)]
```

```
... 주식 코드 기업명 ?삼성
```

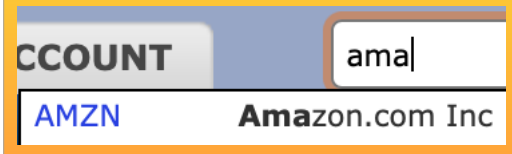
삼성이 들어간 종목은 모두 사용된다.

주식 코드 기업명 ?삼성

	회사명	종목코드	업종	주요제품
30	삼성SDI	6400	일차전지 및 축전지 제조업	칼라브라운관,PDP,평판표시관,모니터,휴대용 디스플레이,폴리머전지(2차전지),초박형...
31	삼성엔지니어링	28050	건축기술, 엔지니어링 및 관련 기술 서비스업	산업설비,건물,구축물,토목시설의 설계,시공,공사감리

관심기업 주가 불러오기

Yahoo 에서 주가를 다운하려면 관심 기업의 주식코드는 반드시 6자리 사용해야 하며 반드시 .KS을 붙여야 한다.
미국 주가는 약어를 사용하면 된다. 물론 .KS 필요없음. http://www.eoddata.com/default.aspx




```
from datetime import datetime
start=datetime(2019,1,1)
end=datetime(2020,10,1)
name='삼성SDI'
df=stock.DataReader('006400.KS','yahoo', start, end)
df.tail(3)
```

^KS11 형식으로 사용하자, 006400.KS 대신

심볼	설명
KS11	KOSPI 지수
KQ11	KOSDAQ 지수
KS50	KOSPI 50 지수
KS100	KOSPI 100
KRX100	KRX 100
KS200	코스피 200

심볼	설명
DJI	다우존스 지수
IXIC	나스닥 지수
US500	S&P 500 지수
VIX	S&P 500 VIX

심볼	설명
JP225	닛케이 225 선물
STOXX50E	Euro Stoxx 50
CSI300	CSI 300 (중국)
HSI	항셍 (홍콩)
FTSE	영국 FTSE
DAX	독일 DAX 30
CAC	프랑스 CAC 40

Date	High	Low	Open	Close	Volume	Adj Close
2020-09-25	417500.0	406000.0	415000.0	412000.0	293217.0	412000.0
2020-09-28	424000.0	413000.0	418000.0	421000.0	229317.0	421000.0
2020-09-29	434000.0	426500.0	433500.0	433500.0	290281.0	433500.0

다른 경제지표 가져오기

```
name1='미국한화환율'
df1=stock.DataReader('USDKRW=X','yahoo', start, end)
name2='한화미국환율'
df2=stock.DataReader('KRWUSD=X','yahoo', start, end)
```

심볼	설명
USD/KRW	달러당 원화 환율
USD/EUR	달러당 유로화 환율
USD/JPY	달러당 엔화 환율
CNY/KRW	위엔화 원화 환율
EUR/USD	유로화 달러 환율
USD/JPY	달러 엔화 환율
JPY/KRW	엔화 원화 환율
AUD/USD	오스트레일리아 달러 환율
EUR/JPY	유로화 엔화 환율
USD/RUB	달러 루블화

심볼	설명
BTC/USD	비트코인 달러 가격
ETH/USD	이더리움 달러 가격
XRP/USD	리플 달러 가격
BCH/USD	비트코인 캐시 달러 가격
EOS/USD	이오스 달러 가격
LTC/USD	라이트 코인 달러 가격
XLM/USD	스텔라 달러 가격

```
[98]  1  df1.tail(3)
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-09-29	1170.699951	1163.099976	1168.729980	1167.859985	0.0	1167.859985
2020-09-30	1164.800049	1160.150024	1163.800049	1163.780029	0.0	1163.780029

```
1  df2.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-09-27	0.000857	0.000851	0.000852	0.000852	0.0	0.000852
2020-09-28	0.000858	0.000854	0.000857	0.000857	0.0	0.000857

다른 모듈 사용하기 : FinanceDataReader

모듈설치

```
!pip install -U finance-datareader
```

```
import FinanceDataReader as fdr
df_krx = fdr.StockListing('KRX')
```

기업 종목정보 및 코드

기업명, 코드 찾기

```
name=input('주식 코드 기업명 ?') #find 종목코드 및 정보
df_krx[df_krx['Name'].str.contains(name)]
```

주식 코드 기업명 ?삼성SDI

	Symbol	Market	Name	Sector
947	006400	KOSPI	삼성SDI	일차전지 및 축전지 제조업
948	006405	KOSPI	삼성SDI 우	NaN

삼성 SDI 기업 주가 데이터 불러오기

```
import datetime as dt
cid='006400' #company id
start_date='2019-01-01'
end_date='2019-12-31'
df=fdr.DataReader(cid,start_date,end_date)
c_name=df_krx[df_krx['Symbol']==cid]['Name'].item()
df.head(3)
```

Date	Open	High	Low	Close	Volume	Change
2019-01-02	222000	224000	208500	210500	345548	-0.038813
2019-01-03	209000	209500	202000	203000	484354	-0.035629

종목 코드는 이전 yahoo 모듈과 동일함

```
fdr.DataReader('KS11',start_date,end_date)
```

```
fdr.DataReader('AMZN',start_date,end_date)
```

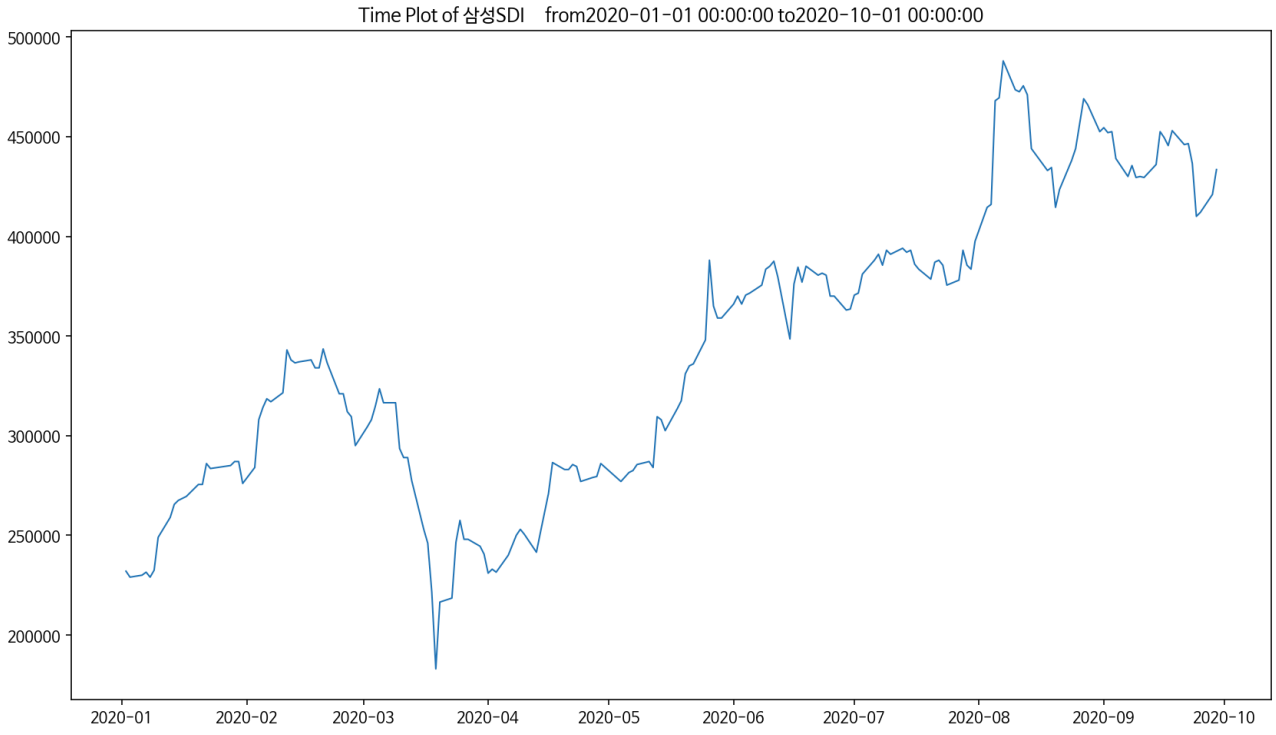
시간도표와 시계열 성분

- 시계열 자료는 시간적 순서를 가지므로 Y축은 값, X축을 시간 순서로 하여 관측값을 연결한 산점도를 그래프를 시간도표(time plot)이라 한다.
- 시간적 순서를 가진 시계열 데이터 의 값의 시간적 변동(변화)를 보기 위한 그래프로 시계열 자료의 구조(4가지 성분 파악)를 파악하는데 도움이 되며 시계열 분석의 시작이다.

삼성 SDI 주가 데이터 시간도표이다.

```
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (14,8)
plt.title('Time Plot of '+str(name)+' from'+str(start)+' to'+str(end))
plt.plot(df['Close'],label='original',linewidth=1)
plt.show()
```

코로나로 인하여 2020년 3월 급격히 하락하였다가 다시 9월까지 급격히 상승하여 2019년 주가보다 높아졌다. 2차전지, 디스플레이의 수요 급증이 주가 상승.을 주도하였다고 할 수 있다.



시계열 성분

시계열 데이터, $\{Y_t; t = 1,2,...,T\}$는 다음의 4가지 성분으로 구성되어 있다. 4가지 성분 중 패턴을 갖는 것은 경향, 계절성, 순환이며 나머지 패턴이 없는 부분은 불규칙으로 정의된다. 주기, 계절성은 차분에 의해 제거되며 순환(주기)가 패턴을 대표한다..

성분 4개

경향(Trend) T_t

(장기간 패턴) 시계열 데이터가 증가(감소)하는 경향이 있는지 혹은 안정적인지 알 수 있다. 직선 경향, 이차 경향이 있으며 장기변동 요인에 의해 발생한다.

순환(cycle) C_t

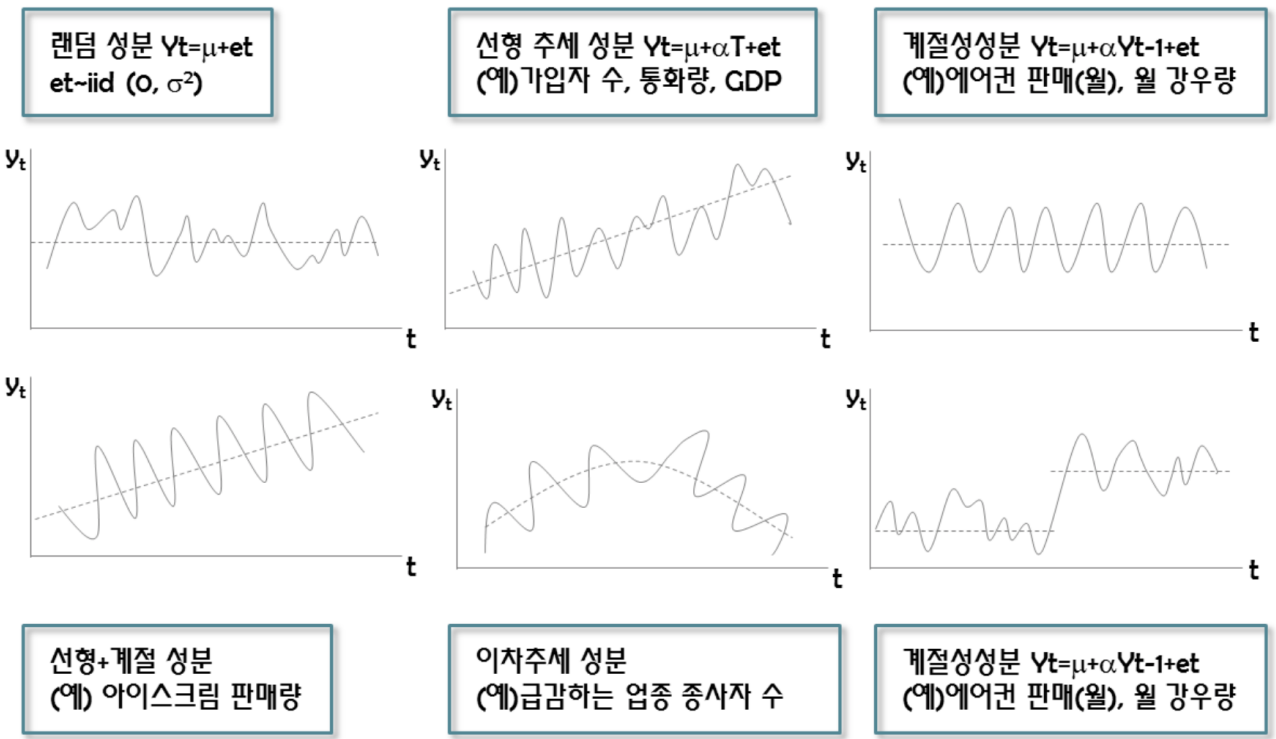
recurring up and down movement around trend levels 일정한 주기(진폭)마다 유사한 변동이 반복된다. sin 커브와 같이 일정 간격으로 높은 곳과 낮은 곳이 반복

계절성(seasonality) S_t

periodic pattern that complete themselves within a specific time period 주별, 월별, 분기별, 년별 유사 패턴이 반복된다.

불규칙성(irregular) I_t

erratic movement with no recognizable pattern 일정한 패턴 없음, 오차항이 여기에 해당함

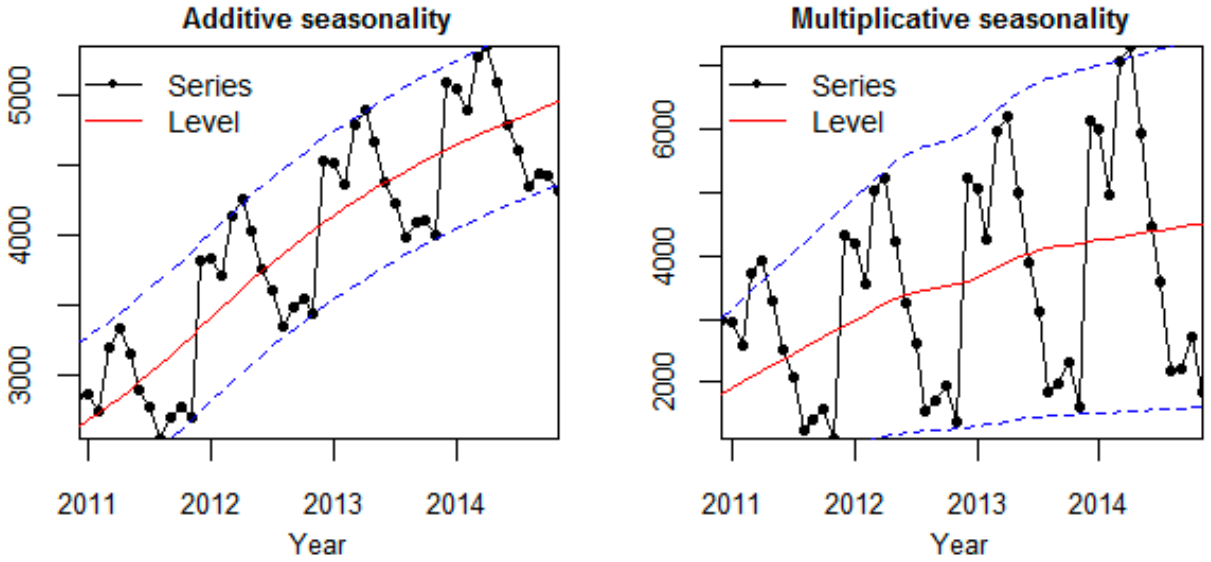


시계열 성분 모형

가법모형 (Additive model) : $Y_t = C_t + S_t + T_t + I_t$

승법모형 (Multiplicative model) : $Y_t = C_t \times S_t \times T_t \times I_t$

승법모형은 계절성분이 증가하는 형태이므로 승법모형을 로그 변환하면 가법모형이 된다.



시계열 자료 분석 방법

회귀분석(계량경제) 방법, BOX-JENKINS 방법, 지수 평활법, 시계열 분해 방법이 있다. 회귀분석 방법과 BOX-JENKINS 방법(ARMA)은 수학적 이론 모형에 의존하고 시간에 따라 변동이 많은(빠른) 시계열 자료에 적용된다. 지수 평활법이나 시계열 분해 방법은 다소 직관적인 방법이며 시간에 따른 변동이 느린 데이터를 분석하는데 사용된다.

과거의 데이터 패턴을 활용하여 미래 값을 예측, 설명변수가 있는 시계열 모형은 econometric 계량경제모형

- frequency domain : Fourier 분석에 기초, spectrum density function
- time domain : 자기상관함수 이용, 관측값들의 시간적 변화 탐색

평활법 : 과거 값의 평균으로 미래 값을 예측하는 방법

- 이동 평균법(moving average): 최근 데이터의 평균을 (혹은 중앙치) 예측치로 사용하는 방법이다. 각 과거치에는 동일한 가중치가 주어지면 과거 패턴 인식이 주목적이다.
- 지수 평활법(exponential smoothing): 현재 가까운 시점에 가장 많은 가중치 주고 멀어질수록 낮은 가중치를 주는 방법이다. 경향이나 계절성 존재여부에 따라 단순지수, 이중지수, 삼중지수, 계절지수 평활법 등이 있다.

ARMA 모형 $Y_t = \mu + \alpha_1 Y_{t-1} + \ldots + \alpha_p Y_{t-p} + \beta_1 e_{t-1} + \ldots + \beta_q e_{t-q} + e_{t-1}$

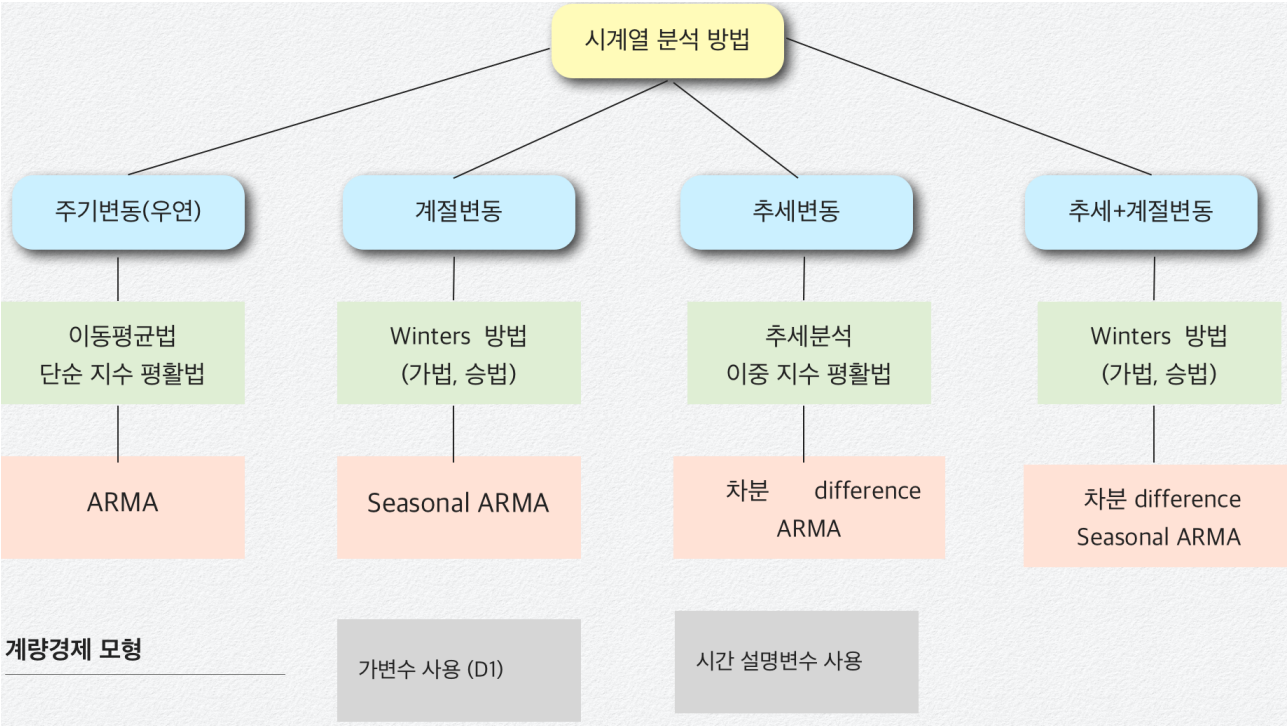
시계열 데이터 $\{Y_t; t = 1,2,...,T\}$에 대한 모형화를 통하여 미래 값을 예측하는 방법이다. 설명변수가 종속변수 자신의 과거 값인 AR(Auto Regressive) 모형 , 설명변수가 오차항의 과거 값인 MA(Moving Average) 모형, 그리고 AR과 MA 모형의 결합인 ARMA 모형이 있다.

회귀모형 : 계량경제 Econometrics Model $Y_t = \mu + \alpha_1 X_{1t} + \alpha_2 X_{2t} + \ldots + \alpha_p X_{pt} + e_{t-1}$

시계열 데이터 $\{Y_t; t = 1,2,...,T\}$ 종속변수로 하고 p개의 (X1, X2, ..., Xp)를 설명하는 회귀분석 모형, 유의한 (영향을 미치는) 설명변수를 찾을 수 있으나 예측에 어려움이 있다. 왜냐하면 설명변수의 예측치도 있어야 하기 때문이다. 그러므로 설명변수의 경우에는 t시점 대신 이전 시점$t-1, t-2, \ldots$을 사용한다.

성분 관련 분석

- 추세분석 : 추세(일차식, 이차식, 로그형태 등) 성분을 파악한다.
- 변동 분해 : 3개 주요 성분 (주기, 계절성, 경향)을 분해하는 방법
- X-11 방법 : 정부통계에서 가장 많이 쓰이는 계절 조정 방법



시계열 프로세스

white noise process

평균이 0이고 분산이 σ^2 인 동일 정규분포로부터 독립적으로(iid) 얻어진 시계열 데이터 $\{Y_t\}$를 백색 잡음(white noise) process라 한다. 데이터의 평균을 μ라 하면 백색잡음 시계열 데이터의 모형은 다음과 같이 쓸 수 있다.

$$Y_t = \mu + e_t,\ e_t \sim N(0,\sigma^2)$$

만약 $y_0 = \mu$라 하면 $y_t = y_0 + e_t + e_{t-1} + \ldots + e_0$가 되는데 이를 random walk process라 한다.

백색잡음 모델은 패턴 인식이 불가능하여 예측이 불가능하다.

stationary process

$F(y_1, y_2, \ldots, y_t) = F(y_{1+k}, y_{2+k}, \ldots, y_{t+t})$이 조건을 만족하는 시계열 데이터를 strongly stationary process(강한 정상성 프로세스)이라 한다. 일정한 기간의 시계열 데이터 결합밀도함수는 동일한 분포를 가진다는 것을 의미한다.

다음 조건을 만족하는 시계열 데이터 는 weakly stationary process(약한 정상성)라 정의한다.

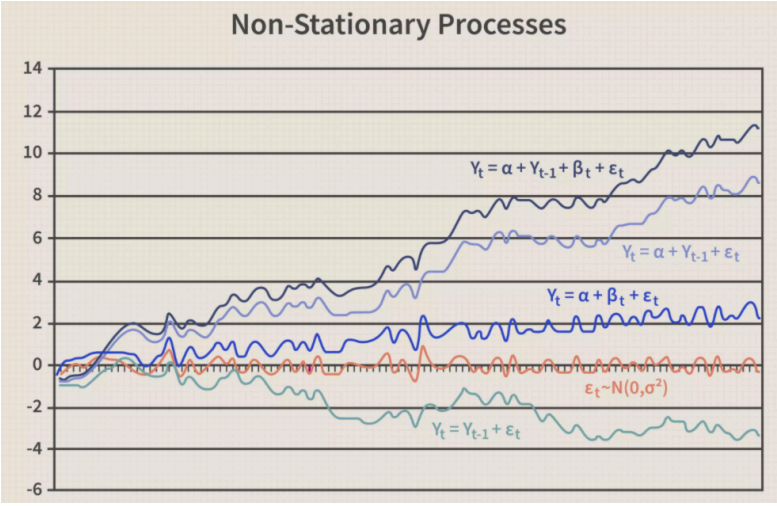
(1)평균이 일정하다. $E(Y_t) = \mu$

(2)분산이 존재하며 일정하다. $V(Y_t) = \sigma^2$

(3)두 시점 사이의 자기 공분산(auto-correlation) $COV(y_{t1}, y_{t2}) = COV(y_{t1+k}, y_{t2+k})$은 시간의 차이에 의존한다.

역한 정상성을 갖는 시계열 데이터는

- (평균, 분산, 공분산) 시간에 따른 변동 없이 일정함
- 시계열 데이터 프로세스가 stationary 해야 모형 적용이 가능함



단위근 unit root 검정

정상성 갖는 시계열 데이터는 평균, 분산의 변동이 없고 시간에 따른 분포의 변동이 없다. 정상성을 갖지 않은 시계열의 변동은 확률적 추세를 갖게되는데 이는 시간이 지나면서 증폭된다.

Non-stationary는 차분이나 변수변환을 통해 해결된다.

Random walk

$Y_t = Y_{t-1} + e_t$ => $Y_t = Y_0 + \sum_{i=1}^{t} e_i$: 오차항에 의해 결정된다.

Random walk with drift

$Y_t = \alpha + Y_{t-1} + e_t$ => $Y_t = t\alpha + Y_0 + \sum_{i=1}^{t} e_i$

Dickey-fuller Test for checking stationarity

시계열 데이터 AR(1) 모형에서 $Y_t = \mu + \phi Y_{t-1} + e_t$

- 귀무가설 : 단위근 모형임 = the time series is non- stationary. $|\phi| = 1$ **<=> 모형 설정 불가**

  $|\phi| = 1$: 단위근 unit root 모형 <=> random walk 모형

- 대립가설 : 단위근 문제가 없다 <=> $|\phi| \neq 1$ <=> 모형설정 가능

  $|\phi| < 1$: 시계열데이터 stationary |

  $|\phi| > 1$: 시계열데이터 explosive => 로그변환을 통해 정상성 프로세스 변환 가능

삼성SDI 주가 단일근 검정

```
from statsmodels.tsa.stattools import adfuller
unit_root=adfuller(df['Close'])
print('단일근 검정통계량=%.2f, 유의확률=%.3f'%(unit_root[0],unit_root[1]))
```

```
단일근 검정통계량=-0.38, 유의확률=0.913
```

유의확률이 91.3%로 유의수준 5%보다 크므로 귀무가설이 채택되어 데이터는 단일근 문제를 갖는다.

```
unit_root=adfuller(df['Close'],regression='ct')
print('추세 단위근 검정통계량=%.2f, 유의확률=%.3f'%(unit_root[0],unit_root[1]))
```

```
추세 단위근 검정통계량=-1.97, 유의확률=0.615
```

Kwiatkowski-Phillips-Schmidt-Shin – KPSS test (trend stationary)

- 귀무가설 : 시계열은 추세 정상성(trend stationary)을 따른다. regression='ct'
- 귀무가설 : 시계열은 상수 정상성(trend stationary)을 따른다. regression='c'
- 대립가설 : 추세 정상성을 따르지 않는다.

```
from statsmodels.tsa.stattools import kpss
kpss(df['Close'], regression='ct')
```

```
(추세_정상,상수_정상) 검정통계량=(0.42, 1.71) 유의확률=(0.010, 0.010)
```

추세 정상성, 상수 정상성도 따르지 않는다.

2020년 코로나로 인하여 주가가 왜곡되었기에 2019년 데이터만으로 단위근 검정결과

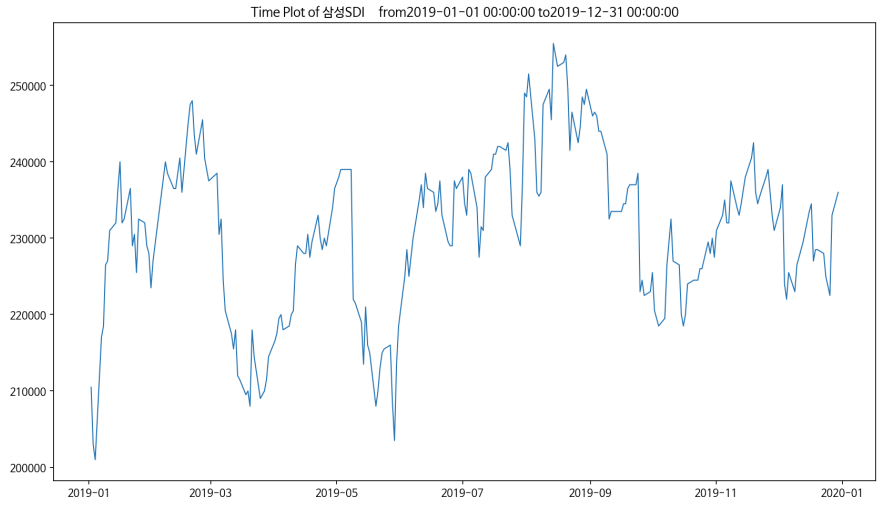
Unit root adfuller 검정결과 : 단위근 문제 없음

```
단위근 검정통계량=-3.55, 유의확률=0.007
```

KSPP 검정결과 : 추세 정상정 있음

```
(추세_정상,상수_정상) 검정통계량=(0.10, 0.24) 유의확률=(0.100, 0.100)
```

추세 정상성이 존재하면 추세를 제거하면 정상성을 갖는 시계열 데이터가 된다.



정상성 만들기

분석대상 시계열데이터가 non-stationary(비정상성) 경우 이를 해결하는 방법은 다음과 같다.

(1) 차분

추세성으로 인한 정상성 위반의 경우 차분으로 해결 가능하다. 추세성이 직선인 경우 1차 차분, 이차식 형태인 경우 2차 차분으로 stationary로 만들 수 있음

- $\nabla Y_t = Y_t - Y_{t-1}$: Back shift 기호 $BY_t = Y_{t-1}$
- pd.diff() : 1차 차분 값 계산, 첫번째 관측치는 결측치 (NaN)

(2) 계절 차분

- 계절성이 있는 시계열가 stationary 하지 않는 경우 계절 차분으로 정상이 되는 시계열 데이터를 Seasional Staionary 하다고 함
- 계절 주기 s 인 경우 : $\nabla_s = Y_t - Y_{t-s} = (1 - B^s)Y_t$

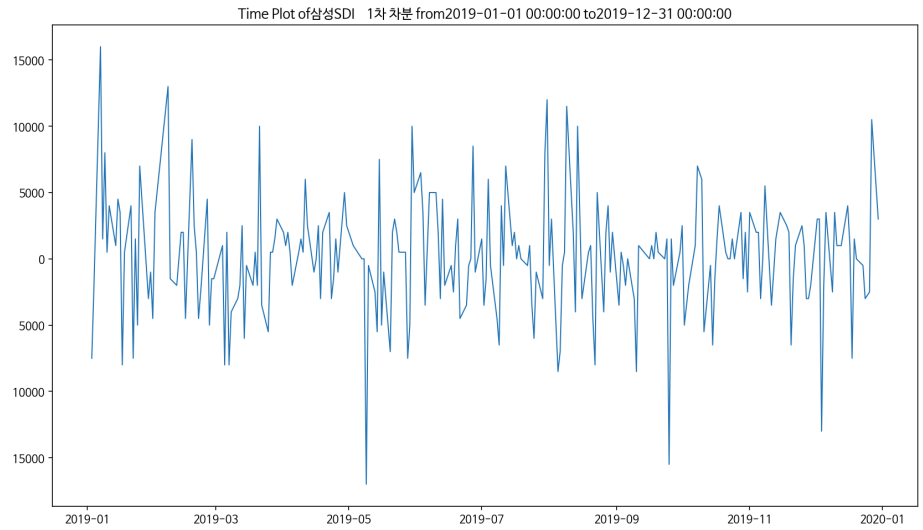
(3) 변수변환

변동 주기의 폭이 증가하거나 감소하는 경우 로그변환, 제곱근 변환으로 안정적 데이터를 만들 수 있음

삼성SDI 주가 단일근 문제 해결 1차 차분 : 2019년 1.1.~2019년 12.31

1차 차분 시 첫 관측치는 결측값이므로 dropna() 함수를 사용하였다.

```
df_diff=df['Close'].diff(1).dropna()
import matplotlib.pyplot as plt
plt.title('Time Plot of'+str(name)+'1차 차분'+' from'+str(start)+' to'+str(end))
plt.plot(df_diff,label='original',linewidth=1)
plt.show()
```



귀무가설이 기각되어 단위근 문제는 해결되었다.

```
from statsmodels.tsa.stattools import adfuller
unit_root_diff=adfuller(df_diff)
print('단위근 검정통계량=%.2f, 유의확률=%.3f'%
(unit_root_diff[0],unit_root_diff[1]))
```

```
단위근 검정통계량=-9.57, 유의확률=0.000
```

이후 분석에서는 삼성SDI 2019년 주가만을 사용하였다.

최적 모형 선택 통계량

Forescasting error 예측오차

$$e_t = Y_t - \hat{Y}_t$$

예측 정확도 척도

Mean Absolute Error (MAE) & Mean Absolute Percentage Error (MAPE)

$$MAE = \frac{1}{T}\sum_{i=1}^{T}|e_t| \qquad MAPE = \frac{1}{T}\sum_{i=1}^{T}|\frac{e_t}{Y_t}| \times 100(\%)$$

Mean Squared Error (MSE) & Root-Mean Squared Error (MSE)

$$MSE = \frac{1}{T}\sum_{i=1}^{T}e_t^2 \qquad RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{T}e_t^2}$$

이동평균법 Moving Average

시계열 데이터는 주기나 불규칙성을 가지고 있으므로 과거의 몇 개 관측치를 평균하여 전반적인 추세를 파악할 수 있는 방법을 이용하여 예측치를 구할 수 있는데 이를 이동평균법(Moving average)이라 한다.

이동 평균법은 과거 몇 개의 과거치의 평균으로 미래 값을 예측하는 방법이다. 지수평활법과는 달리 이동평균법에서는 과거치에 적용되는 가중치는 동일

MA 계산 및 예측

과거 m개의 관측값의 평균으로 이동평균법을 사용한다.

$$MA_t = \frac{\sum_{j=1}^{m} Y_{t-j}}{m} \Rightarrow MA_t = \hat{Y}_t$$

- 중심 이동평균 Centered MA : m의 중앙시점을 사용한다.
- m=3인 경우 일반 이동평균 = $MA_t = \frac{Y_{t-1} + Y_{t-2} + Y_{t-3}}{3}$
- 중심 이동평균=$MA_t = \frac{Y_{t-1} + Y_t + Y_{t+1}}{3}$

주기 m 결정

- 주기의 배수를 활용
- 주가의 경우 : M=5, 20(한달), 60(분기), 120(반년)
- 월별데이터 : M=12, 24, 36, ...

특징

- 계절성, 불규칙성을 제거하여 전반적인 추세 파악 가능 - 직관적 정보 제공
- 주기가 길면 장기 패턴, 짧으면 단기 패턴을 진단할 수 있음 - 주기가 길어질수록 주기(사이클)는 사라지고 직선 형태가 된다.
- 자신의 과거치 (pattern, trace)이용하여 미래 값을 예측한다.
- 자신의 m 개 관측치 평균으로 시계열 자료 {Yt}의 패턴 인식한다.
- 가중치는 1/m으로 동일하다.

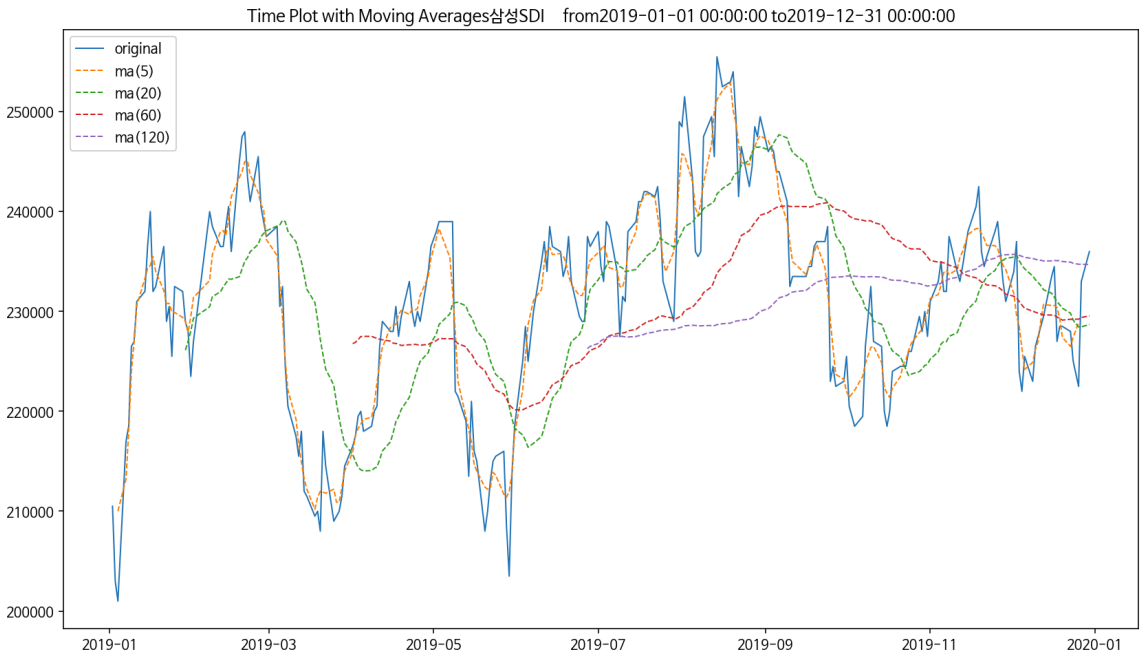
이동평균법 문제점

- 과거치에 대한 가중치가 동일하다. - 시간이 먼 관측값와 가까운 값의 영향을 동일하게 간주하므로 왜곡 가능성이 높음
- 시계열 데이터에 추세과 계절성이 없다면 문제가 없으나, 존재한다면
- 차기 1기만 예측이 가능함 - 이전 추세분석

주기 5, 20, 60, 120으로 이동평균을 계산하여 시간도표를 그렸다. center=True 사용하여 주기 5의 경우 중심 이동평균 계산하였음

```
import pandas as pd
df_ma5=df['Close'].rolling(window=5,center=True).mean()
df_ma20=df['Close'].rolling(window=20).mean()
df_ma60=df['Close'].rolling(window=60).mean()
df_ma120=df['Close'].rolling(window=120).mean()
df_cross=pd.concat([df_ma5,df_ma20,df_ma60,df_ma120], axis=1)
df_cross.columns=['ma5','ma20','ma60','ma120']
df_cross['ma5_diff']=df_cross['ma5'].diff()
df_cross['ma20_diff']=df_cross['ma20'].diff()
df_cross['ma60_diff']=df_cross['ma60'].diff()
df_cross['ma120_diff']=df_cross['ma120'].diff()
df_cross0=df_cross.iloc[:,[0,1,4,5]].dropna() #short term
df_cross00=df_cross.iloc[:,[1,2,5,6]].dropna() #mid term
df_cross000=df_cross.iloc[:,[2,3,6,7]].dropna() #long term

import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["figure.figsize"] = (14,8)
plt.title('Time Plot with Moving Averages'+str(name)+' from'+str(start)+' to'+str(end))
plt.plot(df['Close'],label='original',linewidth=1)
plt.plot(df_ma5,label='ma(5)',linestyle='dashed',linewidth=1)
plt.plot(df_ma20,label='ma(20)',linestyle='dashed',linewidth=1)
plt.plot(df_ma60,label='ma(60)',linestyle='dashed',linewidth=1)
plt.plot(df_ma120,label='ma(120)',linestyle='dashed',linewidth=1)
plt.legend(loc=2)
plt.show()
```



지수평활법 exponential smoothing

모든 관측치에 동일한 가중치를 부여하는 이동평균법은 최근 관측치나 오래된 관측치나 동일한 가중치를 사용하므로 정보를 동일하게 이용하는 단점이 있고 과거 추세 패턴을 많이 반영되는 단점이 있음

최근 관측치에 높은 가중치를 부여하고 멀어질수록 지수적으로 가중치 값 감소시키는 방법인 지수평활법은 미래 예측에 사용

단순지수평활법 Simple(single) ES : 주기(순환)만 있는 경우

시계열 데이터 $\{Y_t\}$에 추세(추세)과 계절성이 없는 경우 사용한다.

평활값

$$S_t = \alpha Y_{t-1} + \alpha(1-\alpha)Y_{t-2} + \alpha(1-\alpha)^2 Y_{t-3} + \ldots + S_0$$

- $\alpha = 0$: 평균 모형, $\alpha = 1$: Unit root (단일근) 모형
- 평활값(예측값) : $S_t = \alpha Y_t + (1-\alpha)S_{t-1}$
- 예측치 : $\hat{Y}_{t+1} = S_t$ - $(t+1)$ 미래 1차만 예측 가능함

초기값 설정

$S_0 = \frac{\sum_i^T}{T}$, $T = 6,\ n/2$ 사용

가중치 설정

- 가중치 α 가 클수록 최근 값에 영향이 큼 Brown : 0.05~0.3
- 주기(m)와 가중치(α) 관계 : $m = \frac{2-\alpha}{\alpha}$
- Montgomery & Johnson : 1-0.8^(1/추세 기울기) $1 - 0.8^{1/trend\ slope}$

삼성 SDI 단순지수평활 예측

```
from statsmodels.tsa.api import ExponentialSmoothing, SimpleExpSmoothing, Holt
ssm1=SimpleExpSmoothing(df['Close']).fit(smoothing_level=0.05,optimized=False)
ssm2=SimpleExpSmoothing(df['Close']).fit(smoothing_level=0.3,optimized=False)
ssm0=SimpleExpSmoothing(df['Close']).fit()
print('optimal alpha',ssm0.model.params['smoothing_level'])
```

1차 추세(trend)가 있어$\alpha = 1$로에 가까운 0.98이 최적 가중치로 산출되었다. 이는 바로 전 관측치, 즉 어제 주가가 오늘 주가에 영향을 거의 모든 영향을 미친다. 적절한 최적 가중치는 0.3 이하가 되면 단순 평활법이 적절함

```
optimal alpha 0.9816012616630881
```
=> 단순지수평활법 적절하지 않음

[추정값, 예측값]

추정값은 단순평활값이고 단순지수평활법에 의해서는 다음 1차기만 가능하여 미래 5개 예측값 모두 동일하다.

```
ssm0.fittedvalues
```

```
2019-01-08    216706.343925
                      ...
2019-12-23    228499.538818
2019-12-24    228009.190884
2019-12-26    225055.365316
2019-12-27    222547.015498
2019-12-30    232807.678273
```

```
ssm0.forecast(5)
```

```
245    235941.265308
246    235941.265308
247    235941.265308
248    235941.265308
249    235941.265308
```

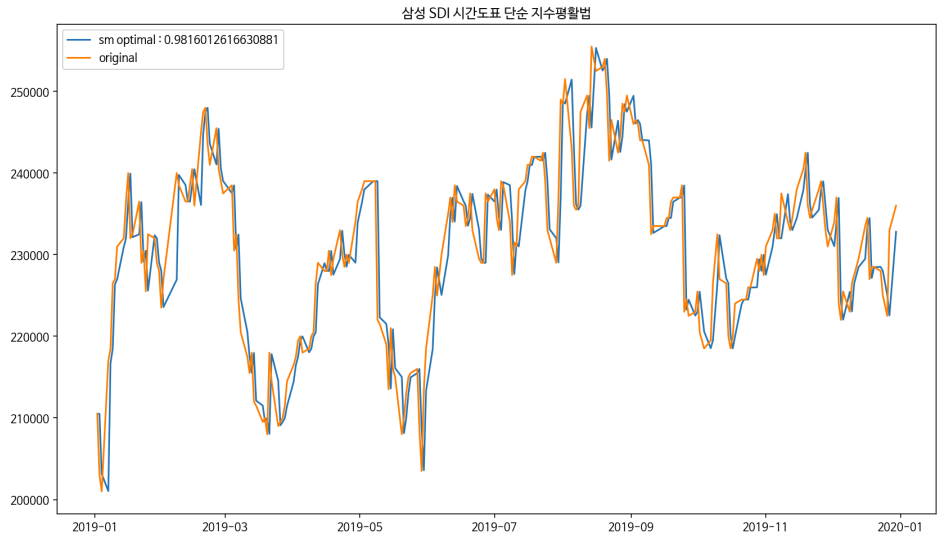
[예측 정확도 RMSE]

```
import numpy as np
sm_rmse=np.sqrt(sum((df['Close']-ssm0.fittedvalues)**2)/(df['Close'].shape[0]))
sm_rmse
```

```
4388.2045689069655
```

[삼성 SDI 주가 시간도표 및 지수평활 시간도표]

```
import matplotlib.pyplot as plt
plt.title('삼성 SDI 시간도표 단순 지수평활법(2019년 주가)')
plt.plot(ssm0.fittedvalues,label=('sm optimal :
'+str(ssm0.model.params['smoothing_level'])))
plt.plot(df['Close'],label='original')
plt.legend(loc=2)
plt.show()
```



삼성 SDI 추세선(단위근 문제) 제거 후 단순지수 평활 예측

차분한 시계열은 최적 α=0이므로 평균모형이다. 백색잡음 모형

```
ssm_diff=SimpleExpSmoothing(df_diff).fit()
print('optimal alpha for 1차 차분',ssm_diff.model.params['smoothing_level'])
```

```
optimal alpha for 1차 차분 0.0
```

Holt Method

시간 추세가 직선형태를 가진 시계열 데이터 예측에 적합한 평활법

이전까지는 가중치가 하나이었으나(이를 1모수 이중지수평활법) Holts는 2모수(α, β) 이중지수 평활법을 제안하였다.

예측 방정식 미래 h기 이후

$$\hat{y_{t+h}} = l_t + hb_t$$

- Level equation : $l_t = \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1})$, α=level smoothing 모수
- Trend equation :$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$ β=trend smoothing 모수

가중치

- 1모수 추정량 : 0.03~0.16 (Brown, 1962)

장기 예측의 경우 지수평활법은 적절하지 않음 - Damped Trend Method

Damped trend 방법

Holt의 선형 방법에 의한 예측값은 hb_t를 지니고 있어 미래로 갈수록 지속적인 추세 (증가 또는 감소)를 모함하게 된다. 이로 인하여 특히 더 긴 예측 기간에 대해 과도하게 예측되는 경향이 있음을 나타낸다. 하여, Gardner & McKenzie(1985) 는 향후 언젠가 추세를 평평한 선으로 "감쇠"시키는 매개 변수를 도입했다. 감쇠된 추세를 포함하는 방법은 매우 성공적인 것으로 입증되었다.

$$\hat{y_{t+h}} = l_t + (\phi + \phi^2 + \ldots + \phi^h)b_t$$

- Level equation : $l_t = \alpha y_t + (1-\alpha)(l_{t-1} + \phi b_{t-1})$, α=level smoothing 모수
- Trend equation :$b_t = \beta(l_t - l_{t-1}) + (1-\beta)\phi b_{t-1}$ β=trend smoothing 모수

삼성 SDI 2019년 주가 Holt 방법

```
from statsmodels.tsa.api import ExponentialSmoothing,
SimpleExpSmoothing, Holt
holt_fit1=Holt(df['Close']).fit(smoothing_level=0.8,smoothing_slo
pe=0.3) #holt additive model
holt_fcast1=fit1.forecast(5).rename("Holt's linear trend")
holt_fit2=Holt(df['Close'],exponential=True).fit(smoothing_level=
0.8,smoothing_slope=0.3) #holt exponential model
holt_fcast2=fit2.forecast(5).rename("Exponential trend")
holt_fit3=Holt(df['Close'],damped=True).fit(smoothing_level=0.8,s
moothing_slope=0.3) #holt damped trend
holt_fcast3 = fit3.forecast(5).rename("Additive damped trend")
```

최적 ϕ 보기

```
holt_fit3.model.params['damping_slope']
```

```
0.3209132740135329
```

3가지 방법 RMSE 계산 : damped Holt 방법이 최적

```
import numpy as np
holt_rmse1=np.sqrt(sum((df['Close']-holt_fit1.fittedvalues)**2)/
(df['Close'].shape[0]))
holt_rmse2=np.sqrt(sum((df['Close']-holt_fit2.fittedvalues)**2)/
(df['Close'].shape[0]))
holt_rmse3=np.sqrt(sum((df['Close']-holt_fit3.fittedvalues)**2)/
(df['Close'].shape[0]))
holt_rmse1,holt_rmse2,holt_rmse3
```

```
(4797.796804622143, 4804.350550676734, 4420.862210845488)
```

damped Holt 방법에 의한 향후 5일 주가 예측결과

```
holt_fcast3
```

```
245    235531.166256
246    235687.846760
247    235743.935568
248    235764.014354
249    235771.202199
```

damped Holt 방법에 의한 향후 5일 주가 예측값 실제값 비교

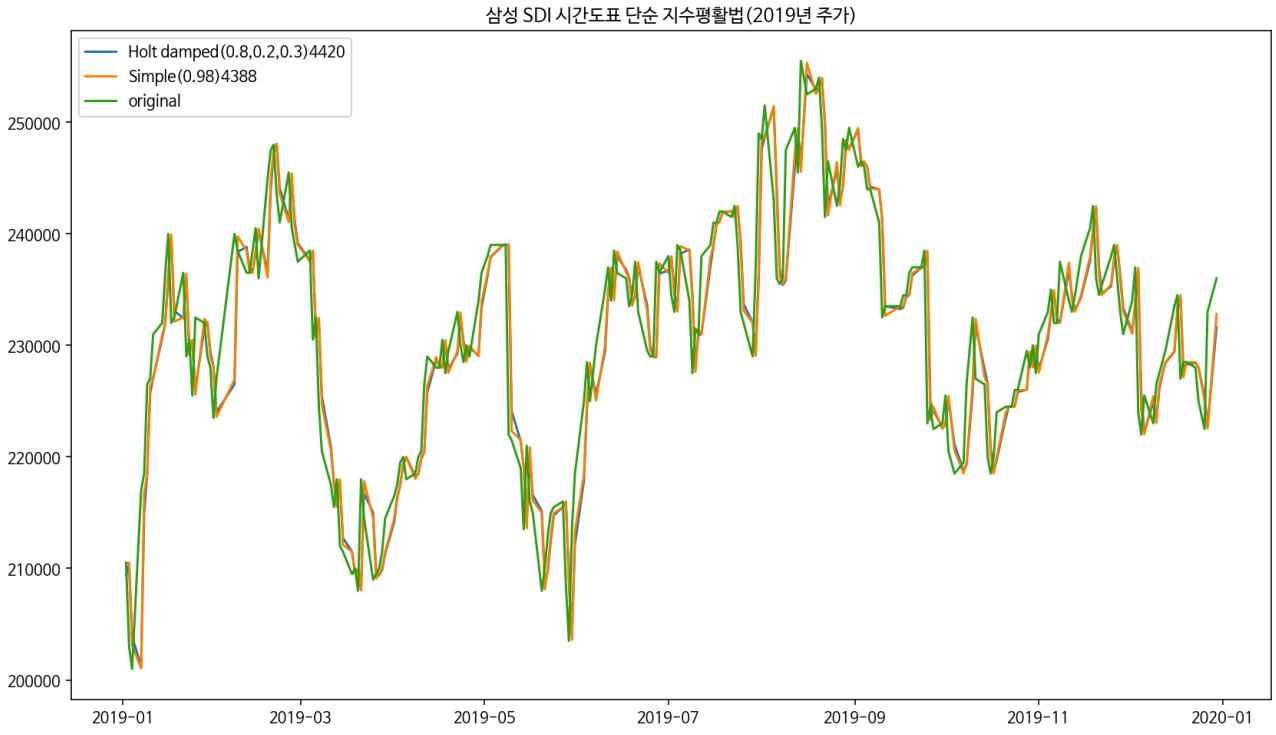
```
start=datetime(2020,1,1)
end=datetime(2020,1,10)
name='삼성SDI '
stock.DataReader('006400.KS','
yahoo', start, end)
```

Date	High	Low	Open	Close
2020-01-02	237500	231500	237000	232000
2020-01-03	235500	228500	235000	229000
2020-01-06	230500	226000	227000	230000
2020-01-07	233500	229000	230000	231500
2020-01-08	231000	227000	229000	229000

단순지수평활법, Holt 방법 비교

```
import matplotlib.pyplot as plt
import numpy as np
plt.title('삼성 SDI 시간도표 단순 지수평활법(2019년 주가)')
plt.plot(holt_fit3.fittedvalues,label=('Holt
damped(0.8,0.2,0.3)'+str(np.int(holt_rmse3))))
plt.plot(ssm0.fittedvalues,label=('Simple(0.98)'+str(np.int(sm_rm
se))))
plt.plot(df['Close'],label='original')
plt.legend(loc=2)
plt.show()
```

RMSE 개념으로는 단순지수 평활법이 조금 예측력이 높으나 단순지수 평활법은 내일 주가만 예측 가능하나 Holt 방법은 어느 정도 단기 예측은 가능하다.



Holt-Winters 계절지수평활법

이전 설명한 지수평활법은 계절성분이 없는 경우 사용된다. 그러므로 계절성이 있는 시계열 데이터에는 적합하지 않다. 강우량, 월별 수출량, 여행 승객 수 등은 계절성을 가지고 있다.

- Winters 모형에는 가법모형과 승법모형이 있고, 변동분해법과 관련이 있음
- Winters 방법은 각 성분을 평활법으로 추정하고 이를 이용하여 시계열 값을 예측한다.

1개의 예측모형과 3개의 평활모형으로 구성되어 있음 : level 수준 l_t, 추세 기울기 b_t, 그리고 계절성분 S_t 구성되어 있고 3개의 모수(α, β, γ)가 존재한다.

상수 h는 미래 예측 차수, m 주기, $k = \dfrac{h-1}{m}$라 하자.

Holt-Winters' additive method 가법모형

- 계절성이 존재하고 시계열 변동 폭이 시간의 흐름에 따라 변동이 없는 경우 사용

$$\hat{y_{t+h}}|t = l_t + hb_t + s_{t+h-m(k+1)}$$

$$\begin{aligned}
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},
\end{aligned}$$

Holt-Winters' multiplicative method 승법모형

- 시계열 변동 폭이 시간의 흐름에 따라 커지는 없는 경우 사용

$$\hat{y_{t+h}}|t = (l_t + hb_t)s_{t+h-m(k+1)}$$

$$\begin{aligned}
\ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma\frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}$$

2가지 방법 적용 : damped H-Winters 가법모형 & 승법모형

```
from statsmodels.tsa.api import Holt
hw_fit1=ExponentialSmoothing(df['Close'], seasonal_periods=5,
trend='add', seasonal='add', damped=True).fit(use_boxcox=True)
hw_fit2=ExponentialSmoothing(df['Close'], seasonal_periods=5,
trend='add', seasonal='mul', damped=True).fit(use_boxcox=True)
```

주가 데이터이므로 주기 5를 사용하였다(사실 주식시장 휴일을 고려하여야 한다).

RMSE 두 방법 예측비교

```
import numpy as np
hw_rmse1=np.sqrt(sum((df['Close']-hw_fit1.fittedvalues)**2)/
(df['Close'].shape[0]))
hw_rmse2=np.sqrt(sum((df['Close']-hw_fit2.fittedvalues)**2)/
(df['Close'].shape[0]))
hw_rmse1,hw_rmse2
```

```
(4955.271499930792, 4719.353655247651)
```

damped Holt 방법에 의한 향후 5일 주가 예측결과

```
hw_fit2.forecast(5).rename("Holt_winters_Mul")
```

```
245    237236.879032
246    237832.629466
247    233616.061469
248    235996.857442
249    235225.020165
```

Date	High	Low	Open	Close
2020-01-02	237500	231500	237000	232000
2020-01-03	235500	228500	235000	229000
2020-01-06	230500	226000	227000	230000
2020-01-07	233500	229000	230000	231500
2020-01-08	231000	227000	229000	229000
2020-01-09	233500	230000	230500	232500
2020-01-10	249500	233000	233000	249000

단순지수평활법, Holt 방법, Holt-winters 방법 비교

```
import matplotlib.pyplot as plt
import numpy as np
plt.title('삼성 SDI 시간도표 단순 지수평활법(2019년 주가)')
plt.plot(hw_fit2.fittedvalues,label=('Holt
-wintwer'+str(np.int(hw_rmse2))))
plt.plot(holt_fit3.fittedvalues,label=('Holt
damped(0.8,0.2,0.3)'+str(np.int(holt_rmse3))))
plt.plot(ssm0.fittedvalues,label=('Simple(0.98)'+str(np.int(sm_rm
se))))
plt.plot(df['Close'],label='original')
plt.legend(loc=2)
plt.show()
```

주가는 계절성을 갖지 않으므로 holt winters 방법이 가장 예측력이 낮다.

ARMA 모형

개요

George Box, Gwilym Jenkins 제안한 시계열 모형

ARIMA(Auto-Regressive Integrated Moving-Average) 모형은 시계열 데이터 {Yt}의 과거치(previous observation) {Yt-1,Yt-2,...}가 설명변수인 AR 모형 - 이전 관측치들의 가중치가 동일하면 이동평균법, 최근부터 멀어질수록 지수적으로 감소하는 가중치는 지수평활법이라 함

과거 관측치가 설명하지 못하는 부분에 해당되는 오차항(et-1,et-2,...)들이 설명변수인 MA, 차분을 나타내는 integrate 의 합성어이다.

$ARIMA(p,d,q) : Y_t = \mu + \alpha_1 Y_{t-1} + \ldots + \alpha_p Y_{t-p} + \beta_1 e_{t-1} + \ldots + +\beta_q e_{t-q} + e_{t-1}$ 모형

- p : 자기회귀 auto-regressive 차수를 의미한다. 시계열 데이터의 이전 관측값에 의해 설명되는 패턴
- q : 이동평균 moving average 차수를 나타낸다. 이전 관측값이 설명하지 못하는 패턴에 의한 설명 부분
- 차수 $p,\ q$는 일반적으로 최대 3까지만 활용된다.
- d는 차분 차수를 의미한다. $d = 1$이면 Y_t 대신 $B^{d=1}Y_t$ 시계열 데이터에 $ARMA(p,q)$가 적용된다.
- 그러므로 계절성이 발견되거나 존재한다고 판단되면 주기에 의해 차분한 후 모형을 추정한다.

ARMA 모형 추정 과정

(1) 데이터 사전 진단

A. 시간도표 진단 - 시각적 판단 (경향, 계절성 여부, Stationaity-정상성=분산 일정)

B. 원 시계열 데이터$\{Y_t\}$ 백색잡음 검정 - 모형 적합 가능여부 진단

C. $\{Y_t\}$ 단위근 문제 진단

D. $\{Y_t\}$ 등분산 검정

(2) 모형 식별 : Correlation Function $(p,\ q)$ 차수 진단 및 계절성 진단

A. ACF Auto Correlation Function

B. PACF Partial Auto Correlation Function

(3) 모형 추정

계절성이 있는 경우 주기만큼 차분한 시계열 데이터에 MLE 방법으로 ARMA(p, q) 회귀계수를 추정한다.

(4) 모형진단

A. 회귀계수의 유의성 진단

B. 잔차 진단 - white-noise / no auto correlation

(5) 예측모형 적용

최종 예측모형을 적용하여 미래 값을 예측한다.

자기상관함수 ACF 및 부분자기상관함수 PACF

Auto correlation function 자기상관함수

정의 $\rho(j)$

시차 j인 시계열 데이터의 상관계수 $\rho(j) = \dfrac{COV(Y_t, Y_{t-j})}{V(Y_t)}$

성질

- $j = 0$: Y_t와 Y_t의 상관계수이므로 1이다.
- $j = 1$: Y_t와 Y_{t-1}의 상관계수로 1차 자기상관계수이다.
- $\rho(j) = \rho(-j)$

백색잡음 $Y_t = e_t$ ACF

$Y_{t-j} = e_{t-j}$이므로 $\rho(j) = \dfrac{COV(e_t, e_{t-j})}{V(e_t)} = 0$ (오차항은 서로 독립이므로 상관계수는 0이다)

$Y_t = \mu + \beta e_{t-1} + e_t \sim MA(1)$ ACF

오차항은 서로 독립이므로 $\rho(1) = \dfrac{COV(Y_t, Y_{t-1})}{V(Y_t)} = 0$ 같은 이유로 $\rho(j) = \dfrac{COV(Y_t, Y_{t-j})}{V(Y_t)} = 0$

즉 모든 $MA(q)$의 자기상관함수 값은 0이다.

$Y_t = \mu + \alpha Y_{t-1} + e_t \sim AR(1)$ ACF

$AR(1)$ Invertibility 가역성 : 만약 $|\alpha| < 1$이면 정상성을 갖는 $MA(\infty)$로 변환 가능하다.
가능하다.

$$\rho(1) = \frac{COV(Y_t, Y_{t-1})}{V(Y_t)} = \frac{\alpha\sigma^2}{\sigma^2} = \alpha$$

$$\rho(2) = \frac{COV(Y_t, Y_{t-2})}{V(Y_t)} = 0$$

같은 방법으로 $\rho(j) = 0,\ for\ j \geq 2$

$Y_t = \mu + \alpha_1 Y_{t-1} + \alpha_2 Y_{t-2} + e_t \sim AR(2)$ ACF

$$\rho(1) = \frac{\alpha_1}{1-\alpha_2}$$

$$\rho(2) = \alpha_1\rho(1) + \alpha_2 = \frac{\alpha_1^2 - (1-\alpha_2)\alpha_2}{1-\alpha 2}$$

$\rho(j) = \alpha_1\rho(j-1) + \alpha_2\rho(j-2)$ => 지수적으로 감소한다. Exponentially decay

$Y_t = \mu + \alpha_1 Y_{t-1} + \ldots + \alpha_p Y_{t-p} + e_t \sim AR(p)$ ACF

주기 p차까지는 acf 값이 지수적으로 감소하고 p주기 지난 $(p+1)$부터는 0에 근사한다.

($AR(p)$가역성)
$1 - \alpha_1 M + -\alpha_2 M^2 - \ldots - \alpha_p M^p = 0$의 근의 절대값이 1보다 클 경우 이 데이터는 $MA(\infty)$ 변환

결론적으로 $MA(q)$ 모든 $\rho(j)$는 0이다. $AR(p)$의 ACF는 차수 p까지 지수적으로 감소하고 그 이후에는 0으로 떨어진다 drop out.

Partial Auto correlation function 부분자기상관함수

정의 $\rho(j)$

(X,Y)의 상관관계를 관심변수 Z 효과를 제외한 후 구한 것을 부분상관관계라 함. (X<-Z) 회귀분석의 잔차와 (Y<-Z) 회귀분석의 잔차와의 상관계수를 의미한다.

[시계열자료 적용] $\phi(k) = corr(Y_t, Y_{t-k} \mid Y_{t-k+1}, Y_{t-k+2}, \ldots, Y_{t-1})$

종속변수 Y_t<- 설명변수 $Y_{t+k-1}, Y_{t+k-2}, \ldots, Y_{t-1}$), 종속변수 Y_{t-k}<- 설명변수 $Y_{t+k-1}, Y_{t+k-2}, \ldots, Y_{t-1}$)
각각의 잔차의 자기상관계수가 부분자기상관함수이다.

$Y_t = \mu + \beta e_{t-1} + e_t \sim MA(1)$ PACF

$MA(1)$ Invertibility 가역성 : 만약 $|\beta| < 1$이면 정상성을 갖는 $AR(\infty)$로 변환 가능하다.

$$\phi(1) = \frac{\beta_1}{1-\beta_1^2}$$

$$\phi(j) = 0,\ for\ j \geq 2$$

$Y_t = \mu + \beta_1 e_{t-1} + \beta_2 e_{t-2} + e_t \sim MA(2)$ PACF

$$\phi(1) = \frac{\beta_1 + \beta_1\beta_2}{1+\beta_1^2+\beta_2^2},\ \phi(2) = \frac{\beta_2}{1+\beta_1^2+\beta_2^2}$$

$$\phi(j) = 0,\ for\ j \geq 3$$

$Y_t = \mu + \beta_1 e_{t-1} + \ldots + \beta_q e_{t-q} + e_t \sim MA(q)$ PACF

PACF는 주기 q까지는 지수적으로 감소하고 $q+1$부터는 0에 근사한다.

($MA(q)$가역성)
$1 - \beta_1 M + -\beta_2 M^2 - \ldots - \beta_q M^p = 0$의 근의 절대값이 1보다 클 경우 이 데이터는 $AR(\infty)$ 변환 가능하다.

$Y_t = \mu + \alpha Y_{t-1} + e_t \sim AR(1)$ PACF

$$\phi(1) = \alpha$$

$$\phi(j) = 0,\ after\ j = 2$$

$Y_t = \mu + \alpha_1 Y_{t-1} + \alpha_2 Y_{t-2} + e_t \sim AR(2)$ PACF

$$\phi(1) = \alpha_1,\ \phi(2) = \alpha_2$$

$$\phi(j) = 0,\ after\ j = 3$$

$Y_t = \mu + \alpha_1 Y_{t-1} + \ldots + \alpha_p Y_{t-p} + e_t \sim AR(p)$ PACF

$$\phi(j) = \alpha_k\ \ for\ j \leq p$$

$$\phi(j) = 0,\ after\ j = p$$

결론적으로 $AR(p)$ p차 이후 모든 PCAF $\phi(j)$는 0이고(drop out) 그 전에는 주기에 해당되는 회귀계수 α_j이다. $MA(q)$의 PACF는 차수 q까지 지수적으로 감소하고 그 이후에는 0으로 떨어진다.

ACF & PACF 활용 모형진단 요약표

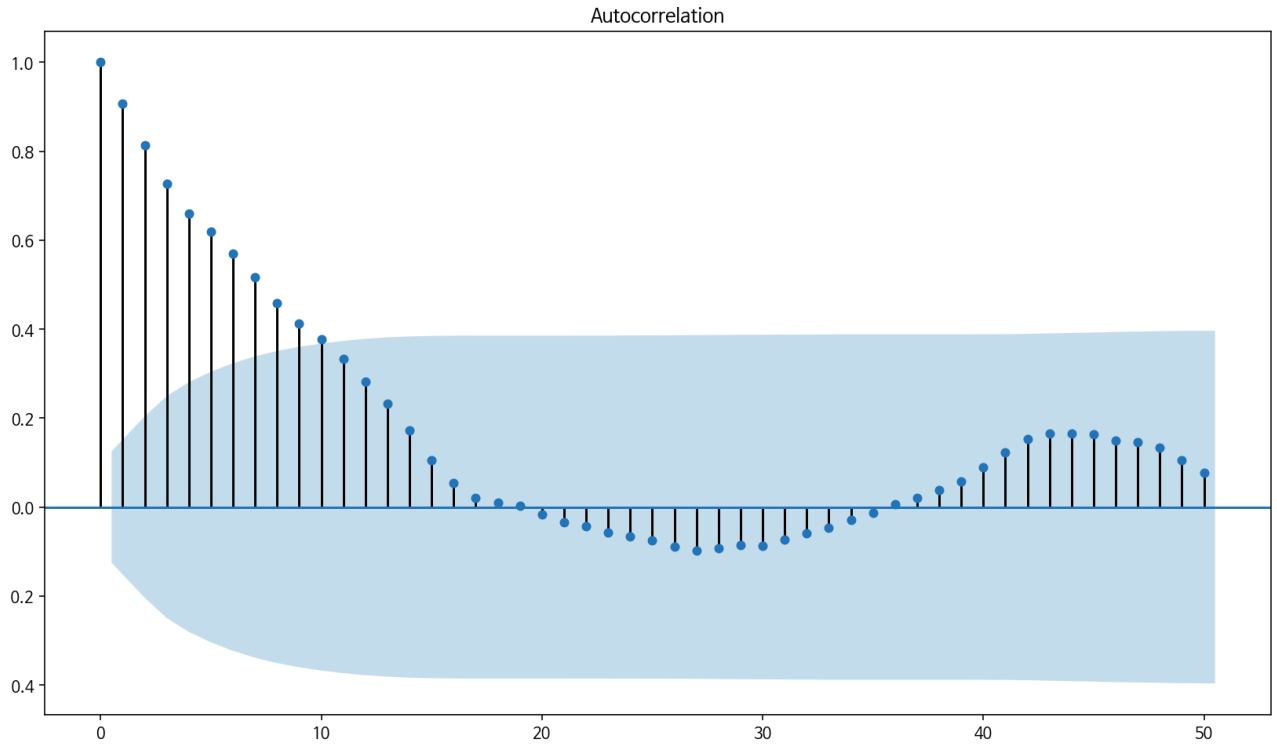
모형	ACF	PACF
AR(p)	Exponentially decay	Drop off after lag p
MA(q)	Drop off after lag q	Exponentially decay
ARMA(p, q)	Exponentially decay	Exponentially decay

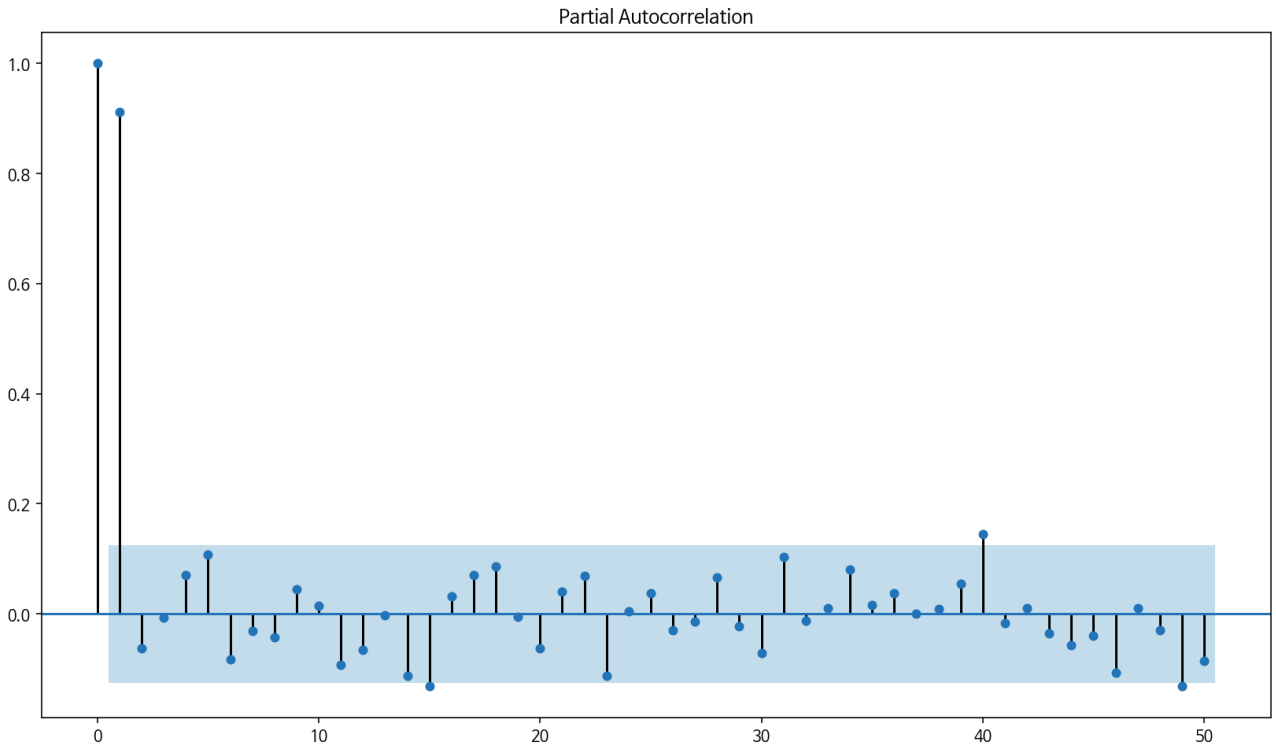
삼성 SDI 2019년 주가 ACF, PACF 구하기

```
from matplotlib import pyplot
from statsmodels.graphics.tsaplots import plot_acf
plot_acf(df['Close'],lags=50)
pyplot.show()
```

```
from statsmodels.graphics.tsaplots import plot_pacf
plot_pacf(df['Close'],lags=50)
pyplot.show()
```

ACF 지수적으로 감소한다, PCAF 1차 주기 이후 0이므로 AR(1)을 따른다.





삼성 SDI 주가 데이터에 AR(1) 모형을 적합한 결과 $\hat{Y}_t = 23000 + 0.9179 Y_{t-1}$으로 유의하나 Y_{t-1}의 회귀계수가 1인 단위근 문제가 발생한다.

```
from statsmodels.tsa.arima_model import ARIMA
model=ARIMA(df['Close'],order=(1,0,0))
model_fit=model.fit()
print(model_fit.summary())
```

```
==============================================================
                 coef    std err          z      P>|z|
--------------------------------------------------------------
:onst         2.301e+05   3212.488     71.616      0.000
ır.L1.Close      0.9179      0.025     36.146      0.000
                                    Roots
==============================================================
```

```
model_fit.forecast(5) #예측값, 추정오차, 95% 신뢰구간
예측값만 출력 : model_fit.forecast(5)[0]
```

```
(array([235512.55783525, 235065.15616194, 234654.50589046, 234277.58811048,
        233931.6318973 ]),
 array([4310.15964527, 5850.49117897, 6885.73796222, 7649.92376765,
        8238.89120841]),
 array([[227064.8001629 , 243960.3155076 ],
        [223598.40415929, 246531.90816459],
        [221158.70747753, 248150.3043034 ],
        [219284.01304141, 249271.16317955],
        [217783.70185627, 250079.56193832]]))
```

삼성 SDI 2019년 주가 ARMA 예측모형

(1) 사전진단

백색잡음 white noise

귀무가설 : 분석대상 시계열 데이터는 백색 잡음이다. $Y_t = e_t$ <=> 모형인식 불가능

대립가설 : 백색 잡음이 아니다. <=> 패턴이 존재한다. <=> ARMA 모형 인식이 가능하다.

Ljung-Box and Box-Pierce statistic differ in their scaling of the autocorrelation function. Ljung-Box test is has better finite-sample properties.

```
import statsmodels.api as sm
sm.stats.acorr_ljungbox(df['Close'],lags=[20],boxpierce=True)
```

- 앞의 값은 검정통계량, 뒤의 값은 유의확률
- Ljung-Box 유의확률 <0.001, Box-Pierce 유의확률 <0.001 모두 귀무가설을 기각하여 백색잡음 아니다,

```
(array([1066.37015835]),
 array([2.69265243e-213]),
 array([1037.8035301]),
 array([3.36832312e-207]))
```

단위근 unit root 검정

귀무가설 : 단위근을 갖는다. 단위근 unit root 모형 <=> random walk 모형

대립가설 : 단위근 문제가 없다 <=> 모형설정 가능

```
from statsmodels.tsa.stattools import adfuller
unit_root=adfuller(df['Close'])
print('단위근 검정통계량=%.2f, 유의확률=%.3f'%(unit_root[0],unit_root[1]))
```

검정통계량=-3.55, 유의확률<0.001
귀무가설이 기각되어 단위근 문제 없음

```
단위근 검정통계량=-3.55, 유의확률=0.007
```

차분 필요성 검정

```
! pip install pmdarima
```

```
from pmdarima.arima import ADFTest
adf_test=ADFTest(alpha=0.05)
adf_test.should_diff(df['Close'])
```

차분이 필요하다.

```
(0.1483061887993583, True)
```

상수, 추세 안정성

귀무가설 : 추세, 상수 정상성을 갖는다.

```
from statsmodels.tsa.stattools import kpss
kpss_ct=kpss(df['Close'],regression='ct')
kpss_c=kpss(df['Close'], regression='c')
print('(추세_정상,상수_정상) 검정통계량=(%.2f, %.2f) 유의확률=(%.3f, %.3f)'%(kpss_ct[0],kpss_c[0],kpss_ct[1],kpss_c[1]))
```

```
(추세_정상,상수_정상) 검정통계량=(0.10, 0.24) 유의확률=(0.100, 0.100)
```

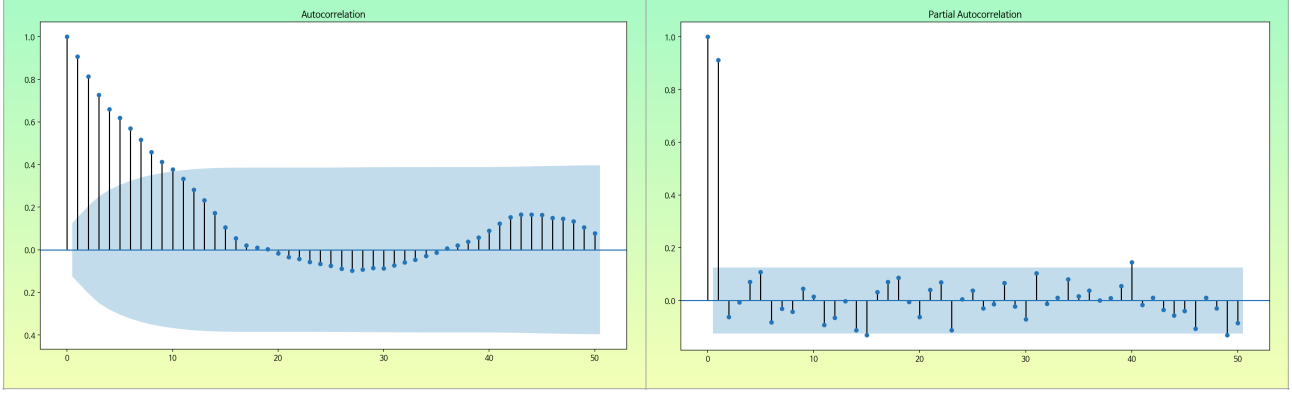
귀무가설이 채택되어 정상성을 갖는 시계열이다.

(2) 모형 진단 ACF, PACF 활용

ACF는 지수적으로 감소하고 PACF는 1차 시점에서 피크가 있고 그 이후에는 0으로 떨어지므로 $AR(1)$이 가장 적합함, 그러므로 차분이 필요해 보인다.

```
from matplotlib import pyplot
from statsmodels.graphics.tsaplots import plot_acf
plot_acf(df['Close'],lags=50)
pyplot.show()
```

```
from statsmodels.graphics.tsaplots import plot_pacf
plot_pacf(df['Close'],lags=50)
pyplot.show()
```



(3) 모형 예측

```
from statsmodels.tsa.arima_model import ARIMA
model=ARIMA(df['Close'],order=(1,0,0))
model_fit=model.fit()
print(model_fit.summary())
```

```
==============================================================
                 coef    std err          z      P>|z|
--------------------------------------------------------------
const       2.301e+05   3212.488     71.616      0.000
ar.L1.Close     0.9179      0.025     36.146      0.000
                                    Roots
==============================================================
```

$\hat{Y}_t = 23000 + 0.9179 Y_{t-1}$

(4) 모형 적합성 진단

모형 계수 유의성 : $AR(1)$의 회귀계수 유의확률은 <0.0001이므로 매우 유의함 => 통과

잔차 백색잡음 : 모형 적합성 결여

```
import statsmodels.api as sm
sm.stats.acorr_ljungbox(model_fit.fittedvalues,lags=[20],boxpierce=True)
```

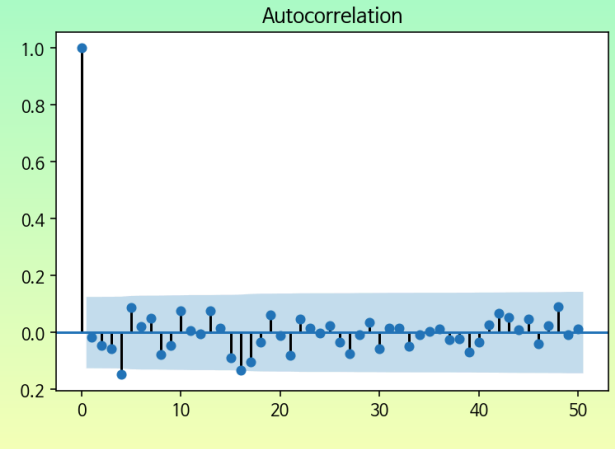
잔차가 백색잡음을 따르지 않는다. 이것은 잔차에는 인식 가능한 패턴이 있다는 것이므로 모형이 적합하지 않다는 것이다.
사전 진단에서 차분이 필요하다는 사실 기억

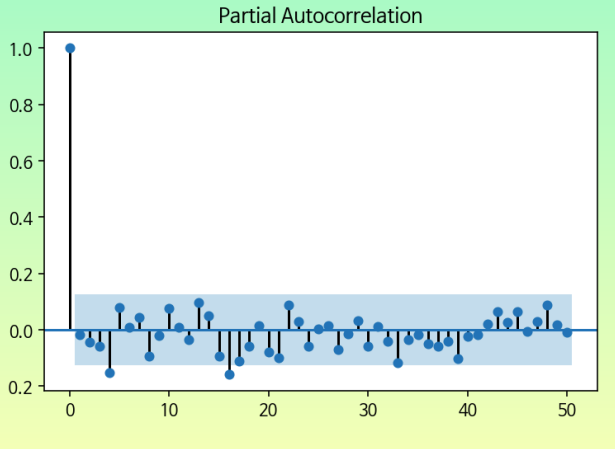
```
(array([1071.25181215]),
 array([2.44321156e-214]),
 array([1042.54898903]),
 array([3.27150533e-208]))
```

차분한 시계열 모형 진단 백색잡음임 - 진단불가

```
df_diff=df['Close'].diff(1).dropna()
from statsmodels.graphics.tsaplots import plot_acf
plot_acf(df_diff,lags=50)
pyplot.show()
```

```
from statsmodels.graphics.tsaplots import plot_pacf
plot_pacf(df_diff,lags=50)
pyplot.show()
```





차분 시계열 백색잡음 검정 : 결과 백색잡음

```
import statsmodels.api as sm
sm.stats.acorr_ljungbox(df_diff,lags=[20])
```

```
(array([25.39624977]), array([0.18668985]))
```

(3-*/4-*) 모형 추정 by auto ARIMA 함수 :

AIC가 가장 적은 최적 ARMA 모형을 찾는다.

```
!pip install pyramid.arima
from pyramid.arima import auto_arima
fit_auto=auto_arima(df['Close'],error_action='ignore')
print(fit_auto.summary())
```

```
                           Statespace Model Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  245
Model:               SARIMAX(3, 1, 2)   Log Likelihood               -2383.116
Date:                Sat, 03 Oct 2020   AIC                           4780.232
Time:                        11:25:06   BIC                           4804.712
Sample:                             0   HQIC                          4790.091
                                  - 245
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     36.1925    139.807      0.259      0.796    -237.824     310.209
ar.L1         -0.0528      0.324     -0.163      0.871      -0.688       0.582
ar.L2          0.5600      0.189      2.960      0.003       0.189       0.931
ar.L3          0.0876      0.046      1.899      0.058      -0.003       0.178
ma.L1          0.0215      0.328      0.065      0.948      -0.622       0.665
ma.L2         -0.5758      0.196     -2.930      0.003      -0.961      -0.191
sigma2      1.814e+07      0.002   7.84e+09      0.000    1.81e+07    1.81e+07
==============================================================================
```

회귀계수 일부가 유의하지 않고 잔차도 백색잡음이 아님

```
import statsmodels.api as sm
fit_auto.fittedvalues=fit_auto.predict_in_sample(df['Close'])
sm.stats.acorr_ljungbox(fit_auto.fittedvalues,lags=[20])
```

```
(array([165.55662336]), array([6.31848907e-25]))
```

(5) 예측 활용

RMSE 계산

auto는 1차 차분을 하여 첫 예측치(89원으로 예측)는 관측치와 편차가 큼, 하여 첫 관측치의 값을 예측값으로 하였음

```
fit_auto.fittedvalues[0]=df['Close'][0]
```

```
import numpy as np
arma_rmse=np.sqrt(sum((df['Close']-model_fit.fittedvalues)**2)/
(df['Close'].shape[0]))
auto_rmse=np.sqrt(sum((df['Close']-fit_auto.fittedvalues)**2)/
(df['Close'].shape[0]))
arma_rmse,auto_rmse
```

```
(4460.255038675654, 4212.302909677389)
```

자동 ARMA 적합 모형 $ARMA(3,1,2)$가 더 적합

예측값 향후 5일

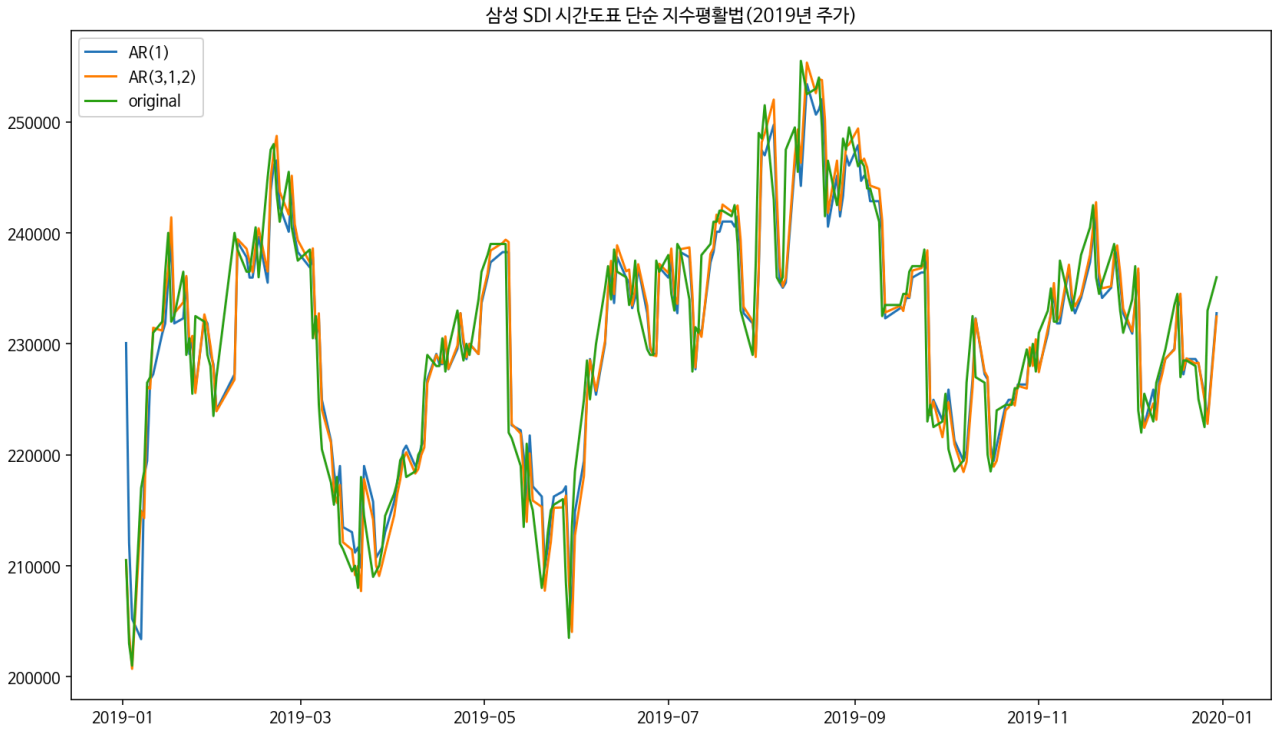
```
model_fit.forecast(5)[0], fit_auto.predict(n_periods=5)
```

```
(array([235512.55783525, 235065.15616194, 234654.50589046, 234277.58811048,
        233931.6318973 ]),
 array([235735.48303953, 236348.54317153, 236467.01449094, 236817.08214149,
        236954.82623665]))
```

관측값과 추정값 시간도표

Auto ARMA 방법에 의해 추정되는 값은 배열로 되어 있고 행 인덱스는 날짜가 아니라 숫자 0~ 으로 시작하므로 원 데이터와 함께 시간도표를 그리기 위하여 행 인덱스를 원 데이터와 동일하게 하였다. 첫 2 코드

```
fit_auto.fittedvalues=pd.DataFrame(fit_auto.fittedvalues)
fit_auto.fittedvalues.set_index(df.index,inplace=True)
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (14,8)
plt.title('삼성 SDI 시간도표 단순 지수평활법(2019년 주가)')
plt.plot(model_fit.fittedvalues,label=('AR(1)'))
plt.plot(fit_auto.fittedvalues,label=('AR(3,1,2)'))
plt.plot(df['Close'],label='original')
plt.legend(loc=2)
plt.show()
```



연습문제

대한민국 1980년 1월부터 2019년 12월 월별 출생아 수에 대하여 다음을 실행하시오.

(1) 시간도표를 그리고 해석하시오.

(2) 이동평균법 추정하시오.

(3) 지수평활법 중 최적 방법으로 추정하고 20년 출생자수를 예측하시오.

(4) 최적 ARMA 추정 모형을 설정하고 20년 출생자수를 예측하시오.

(5) 자동 ARMA 추정 모형을 설정하고 20년 출생자수를 예측하시오.

(6) (3)~(5) 방법 추정값과 원 데이터, 4개의 시간도표를 함께 그리시오. (RMSE 출력)

성분 분해 Decomposition

개념

시계열데이터 패턴을 체계적, 비체계적 성분으로 구별하여 예측에 활용

체계적 Systematic: 성분 중 일관성, 재발현이 예측 가능, 모형 가능 - 주기, 동향, 계절성

비체계적 Non-Systematic: 무규칙성

4개 성분

- Cycle(주기, 순환): 사인함수(sine function)의 형태(주기)를 가지며 시간의 흐름에 따라 증감이 반복되나 주기는 불규칙함. 주기의 평균(중앙) 값을 level (총평균)이라 함.
- Trend(추세): 시간의 경과에 따른 증가, 감소하는 등의 일정한 추세를 갖는 변동을 의미함, 일차, 이차, 다항 추세를 가며 추세
- Seasonality(계절성): 강수량, 기온, 실업율 등 연, 월, 분기 등의 일정한 주기가 규칙적으로 반복되는 주기를 갖는 성분
- Noise(무규칙성): 시간과 관계없이 랜덤한 원인에 의해 나타나는 변동으로 예측할 수 없고, 관심의 대상 성분도 아님, 백색잡음

평활법 차이

평활법은 시계열을 구성하는 각 성분의 형태나 존재여부에 관계없이 불규칙성분을 제거하여 미래 값을 예측하는 방법이나 분해법은 시계열 구성 성분을 구분한 후 이를 활용하여 미래 값을 예측함

분해법의 또 다른 목적은 계절조정(seasonal adjustment)이다. 경제지표에 대한 장기변동 예측의 경우 계절적 성분을 제외한 추세만을 예측하는 경우가 많다.

모형 종류

가법 모형 $Y_t = C_t + T_t + S_t + I_t$

- 계절성분의 진폭이 시계열 수준에 관계없이 일정할 때 적용

승법 모형 - $Y_t = C_t \times T_t \times S_t \times I_t$

- 계절성분의 진폭이 시계열 수준에 따라 변동할 때 사용
- 승법모형을 로그변환하면 가법모형이 된다. 즉, 계절변동의 진폭의 변동은 이분산 문제와 동일함을 알 수 있다.

분해방법

추세모형에 의한 분해

단계 1 : 원 시계열데이터 Y_t 을 추세모형 $Y_t = b_0 + \sum_i^k b_i t^i + e_t$ 적합한 후 추세성분을 추정한다.

$\hat{T}_t = \hat{b}_0 + \sum_i^k \hat{b}_i t^i$ - 추세성분 추정치

단계 2 : $(Y_t - \hat{T}_t)$ 계절모형을 적용하여 $Y_t - \hat{T}_t = \sum_i^m A_i sin(\frac{2\pi i}{s} t + \phi_i) + e_t$ 적합한 후 계절성분을 추정한다. $\hat{S}_t = \sum_i^m \hat{A}_i sin(\frac{2\pi i}{s} t + \hat{\phi}_i)$

단계 3 : $Y_t - \hat{T}_t - \hat{S}_t$ 을 불규칙성분으로 추정한다.

위의 단계별 예측은 각 성분이 서로 독립이라는 가정 하에 얻는다. 만약 독립이 아니라면

$Y_t = b_0 + \sum_i^k b_i t^i + \sum_i^m A_i sin(\frac{2\pi i}{s} t + \phi_i) + e_t$ 동시 추정한다.

만약 승법모형이라면 $Y_t = C_t * T_t * S_t * I_t$ => $\hat{T}_t$ 추세성분 추정

- 잔차 시계열 $\frac{Y_t}{\hat{T}_t}$에 $\hat{S}_t$ 계절성분 추정
- 불규칙 성분 $\hat{I}_t = \frac{Y_t}{\hat{T}_t \hat{S}_t}$ 추정

이동평균법에 의한 분해 : 가법모형

단계 1 : 원 시계열 $\{ Y_t \}$ 계절성분 주기 s에 적절한 항의 이동평균을 적용하여 계절성분과 불규칙성분에 제거된 추세_순환 성분 $T_t \hat{+} C_t$ 을 얻는다. $M_t = \frac{1}{s}(Y_{t-s+1} + Y_{t-s+2} + \ldots + Y_t)$: 이동평균

단계 2 : $Y_t - (T_t \hat{+} C_t) = S_t \hat{+} I_t$: 계절_불규칙 성분을 추정한다.

단계 3 : $S_t \hat{+} I_t$ 성분추정에 계절성분의 주기 s와 일치하지 않는 개수 항의 이동평균을 적용시켜 계절성분 { $\hat{S}_t$ } 추정한다.

단계 4 : 추세_순환 성분 $T_t \hat{+} C_t$에 추세다항식 모형을 적용하여 추세성분 { $\hat{T}_t$ } 추정한다.

단계 5: 추세_순환 성분 $T_t \hat{+} C_t$에서 단계4의 추세성분 { $\hat{T}_t$ }을 빼면 순환성분 {$\hat{C}_t$} 추정한다.

단계 6 : 최종 불규칙 성분은 { $Y_t - \hat{T}_t - \hat{C}_t - \hat{S}_t$ }을 추정한다.

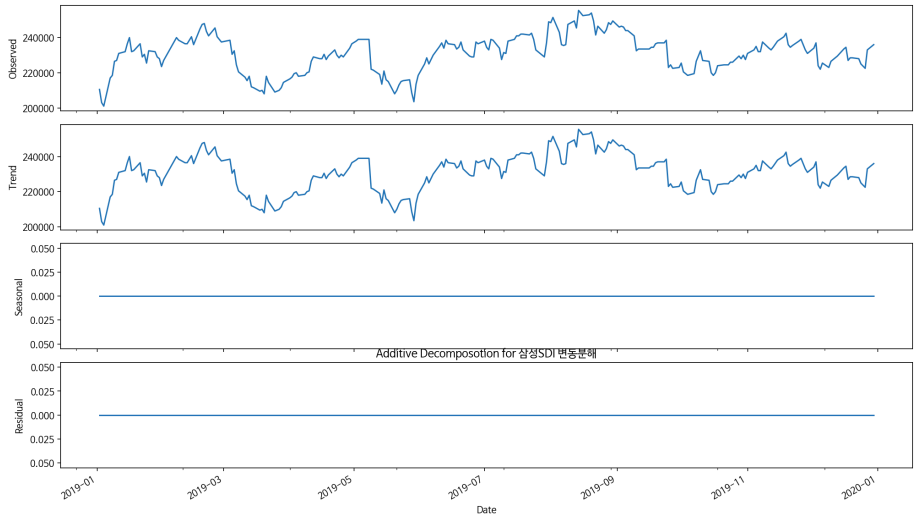
계정조정에 사용되는 이동평균법

- 대칭 (2d+1) 항 이동평균 : $MA_t(2d+1) = \sum_{j=-d}^{d} \frac{Y_{t+j}}{2d+1}$
- 비대칭 (2d) 이동평균 : $MA_t(2d+1) = \sum_{j=-d+1}^{d} \frac{Y_{t+j}}{2d}$
- 대칭(2d+1) 가중이동평균 : $MA_t(2d+1) = \sum_{j=-d}^{d} w_j Y_{t+j'} \sum w_j = 1$

삼성 SDI 주가 변동분해

```
from statsmodels.tsa.seasonal import seasonal_decompose
result=seasonal_decompose(df['Close'],model='additive',freq=1)
result.plot()
plt.title('Additive Decomposotion for 삼성SDI 변동분해')
plt.show()
```

가법모형은 multiplicative 사용하고 주기 설정은 freq=m을 설정한다.



추세분석 trend analysis

개념

- 시계열 데이터 함수 표현 : $Y_t = f(t) + e_t$ (패턴함수 + 오차)
- 대립가설 : $f(t)$는 시간 t에 의해 결정되는 함수로 추세함수
- 추세$f(t)$가 시간 t에 의존하지 않으면 추세가 없음 - 귀무가설 : $f(t)$는 상수

시간 순서를 갖는 시계열 데이터의 특성을 고려하여 시간의 함수로 표현하는 방법으로 다항추세모형(polynomial trend model)이 가장 널리 사용

$Y_t = b_0 + b_1 t + b_2 t^2 + \ldots + b_q t^q$, q차 다항식

- $q = 0$: 상수평균모형 constant mean model
- $q = 1$: 선형추세모형 linear trend model
- $q = 2$: 이차추세모형 quardratic trend model
- 계절추세모형 seasonal trend model : 동일 패턴이 일정한 주기를 가지고 반복됨

추세모형

- 단순 선형 Simple Linear : $f(t) = a + b * T$
- 제곱근 root : $f(t) = a + b * \sqrt{T}$
- 역함수 : $f(t) = a + b * (1/T)$
- 이차 Qurdratic : $f(t) = a + b * T + c * T^2$
- 로그모형 : $f(t) = a + b * ln(T)$
- 지수 exponential : $f(t) = a + b * exp(T)$
- 역지수 Negative exponential : $f(t) = a + b * exp(-T)$
- 성장 Growth : $f(t) = a + b^T$
- S-곡선 : $f(t) = (10^a)/(a + b * c^T)$

상수평균모형 $Y_t = b_0 + e_t$

- 불규칙 성분만 있음
- 추정치 $\hat{b}_0 = \bar{Y} = \sum \frac{Y_t}{T}$: 시계열 데이터 총평균
- L기 이후 예측치 : $\hat{Y}_{t+L} = \hat{b}_0$ | 예측오차 : $\hat{e}_{t+L} = b_0 + e_{t+L} - \bar{Y}$
- 예측오차 분산 추정 : $\hat{V}(\hat{e}_{t+L}) = (1 + 1/n)s^2$, $s^2 = \sum \frac{(Y_t - \bar{Y})^2}{(T-1)}$

직선, 이차 추세성분 모형 , $Y_t = b_0 + b_1 t + e_t$ $Y_t = b_0 + b_1 t + b_2 t^2 + e_t$

- 회귀모형 OLS 추정

$$X = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ \cdots & \\ 1 & T \end{pmatrix}, \underline{b} = \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} \qquad X = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ \cdots & & \\ 1 & T & T^2 \end{pmatrix}, \underline{b} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix}$$

계절성분만 있는 모형

- $Y_t = b_0 + b_1 sin(\frac{2\pi t}{s}) + b_2 cos(\frac{2\pi t}{s}) + e_t$, s : 주기 period
- $Y_t = b_0 + \sum_i^m A_i sin(\frac{2\pi t}{s} t + \phi_i) + e_t$, A_i : 진동폭, ϕ_i : Phase shift, $\frac{2\pi i}{s}$: 2π 단위시간 동안 관

  측되는 i번째 주기항의 빈도, $\frac{s}{i}$: i-번째 주기항의 주기, m은 최대 s/2를 넘지 않음, 일반적으로 m=s/2 사용

추세 & 계절성분 있는 모형

$$Y_t = b_0 + b_1 t + b_2 sin(\frac{2\pi t}{s}) + b_3 cos(\frac{2\pi t}{s}) + e_t$$

비선형 추세모형

- Gompertz 모형 $E(Y_t) = Kexp(-b_0 exp(-b_1 t))$
- Von Bertalantiff 곡선 $E(Y_t) = [1 - b_0 exp(-b_1 t)]^3$
- Logistic 곡선 $E(Y_t) = \frac{K}{[1 + exp(b_0 + b_1 t)]}$