ESC

ARMA and ARIMA Modeling and Forecasting

박준경, 조민수, 지민구

YONSEI UNIVERSITY

목차 / CONTENTS

1. Time Series Review

- Time Series & Stationarity, White Noise & Random Walk
- Backshift Operator & Differencing and Stationarity, AR & MA & ARMA

2. Model Building Problems & Estimation Methods

- How to Analyze Time Series Model?, ARIMA Model
- Estimation Methods, The Innovations Algorithm,
  Method of Moments, Method of Maximum Likelihood

3. Order Determination

- Information Criterion
- Automatic Order Determination using PYTHON

4. Diagnosis of models

- Significance Test for Estimators
- Residual analysis

5. Forecasting

- In-sample prediction
- Out-of sample forecasting

6. Summary

ESC

01 Time Series Review

Time Series & Stationary

What's **Time Series**?

:an ordered sequence of random variables X_t,

where

t is time index of $t \in T = (\cdots, -n, \cdots, -1, 0, 1, \cdots, n, \cdots)$

t라는 시점에 도달하기 전까지는 X_t의 값을 알 수 없는 **random variable**!

결국 **discrete한 stochastic process**로도 볼 수 있다.

Time series value는 random variable의 realization으로 보면 된다.

Stationary(정상성): 평균, 분산, 공분산 같은 통계적 특징이 시간(시점)에 따라 변하지 않는다는 특징.

Definition 1.4 A time series $\{X_t\}$ is strictly stationary or has strict stationarity if $\{X_1, \cdots, X_n\}$ and $\{X_{1+k}, \cdots, X_{n+k}\}$ possess the same joint distribution for any integer $n \geq 1$ and any integer k.

Definition 1.5 A time series $\{X_t\}$ is weakly stationary or has weak stationarity if (1) $\mathsf{E}(X_t) = \mu$ is a constant and (2) for any time t, $\mathsf{E}(X_t^2) < \infty$ and $\mathrm{Cov}(X_t, X_{t+k}) = \gamma(k)$ is independent of t for each integer k.

White Noise & Random Walk

White Noise(백색 소음 과정): 시간의 흐름에 상관 없이 **완전 랜덤하게 움직이는 모델**. **stationary**한 경우 중 매우 특수한 경우!

Definition 1.8 A time series $\{W_t\}$ is called a white noise series or purely random series if it satisfies the following conditions: (1) for all t, $\mathsf{E}(W_t) = \mu$ is a constant; (2) for all t, $\mathrm{Var}(W_t) = \sigma_w^2$ is a constant; (3) it is uncorrelated at different time points, that is, when $t \neq s$, $\mathrm{Cov}(W_t, W_s) = 0$.

Random Walk: **술취한 사람의 발자국**을 생각해보자.

거나하게 취한 사람은 첫 발자국을 내민 순간 다음 발을 제대로 두기 힘들다.

다음 발자국은 분명 첫 발에서 멀지 않은 곳에 떨어져 있겠지만, 술에 취한 그의 정신상태 때문에 이상한 곳에 찍히게 될 것이다.

즉, 현재 상태(두 번째 발자국 이후)는 이전 상태에 의존하지만, 무작위적인 변화를 가진다.

Definition 1.9 A time series $\{X_t\}$ is called a random walk if it satisfies the following equation

$$X_t = X_{t-1} + W_t \tag{1.15}$$

where $\{W_t\}$ is a white noise and, for all t, W_t and X_{t-1} are uncorrelated.

Backshift Operator & Differencing and Stationarity

Backshift Operator(후향 연산자): bascially **takes the value one step earlier**.

$$BX_t = X_{t-1}$$
$$B^n X_t = B^{n-1}(BX_t) = X_{t-n}$$

Differencing(차분):

Differencing the order d:

$$\nabla^1 X_t = \nabla X_t = (1-B)X_t = X_t - X_{t-1}$$
$$\nabla^d X_t = (1-B)^d X_t$$

Differencing the lag k:

$$\nabla_1 X_t = \nabla X_t = (1-B)X_t = X_t - X_{t-1}$$
$$\nabla_k X_t = (1-B^k)X_t = X_t - X_{t-k}$$

Order: 몇 번 1-B를 적용했는지.

Lag: 몇 번 뒤의 시차로 갔는지.

차분을 하는 이유: **Non-stationary time series**를 **stationary**하게 만들기 위해서.

정상화를 하는 이유: 시계열 분석의 가장 중요한 목적 중 하나는 **미래를 '예측'** 하는 것!

비정상 시계열은 시간이 지남에 따라 평균, 분산, 공분산 구조가 달라져서 **과거 데이터로 미래를 예측하는 것이 의미가 없어짐**.

정상화를 통해 통계적 특징들이 시간에 따라 변하지 않도록 만들어 **과거 데이터로 미래를 설명할 수 있는 확률적 구조 형성**.

AR & MA & ARMA

Motivation:

과거의 패턴이 안정적으로 지속된다면, 시계열 값은 과거의 데이터에 의해 예측 가능하다. → **정상성**만 만족된다면 예측이 수월!

그렇다면 어느 정도 멀리 있는 데이터를 이용해야 할까? 또 멀리 있을수록 예측에 대한 기여도가 떨어질 것이므로 **가중**을 이용하자!

AutoRegressive Model(AR Model)

: 시계열 $\{X_t\}$를 그 이전 시점의 시계열 $\{X_{t-1}, X_{t-2}, \cdots\}$로 회귀시킨 모형.

$$\text{AR Model of order p}$$
$$AR(p) : X_t = \varphi_0 + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t$$
$$\text{where } \{w_t\} \sim^{iid} WN(0, \sigma_\epsilon^2), E[X_s \epsilon_t] = 0 \text{ if } s < t,$$
$$\varphi's \text{ are all real numbers, } \varphi_p \neq 0$$

Moving Average Model(MA model)

$$\text{MA Model of order q}$$
$$\text{MA(q)} : X_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots \theta_q \epsilon_{t-q}$$
$$\text{where } \{\epsilon_t\} \sim^{iid} WN(0, \sigma^2)$$

MA model은 X_t가 과거 시점의 white noise term 들에 대해 선형결합으로 표현되어, 과거 noise에 대한 regression 형태로 생각해볼 수 있다.

→ 결국 MA model은 white noise를 lag에 따라 가중을 주어 합산한 모델이다! θ의 값이 크면 클수록 해당 시차의 noise가 현재 값에 더 큰 영향을 준다.

3.5 ARMA Model

$$\text{ARMA Model of order p and q}$$
$$ARMA(p,q) : X_t = \varphi_0 + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}$$
$$\text{where } \{w_t\} \sim^{iid} WN(0, \sigma_\epsilon^2), E[X_s \epsilon_t] = 0 \text{ if } s < t,$$
$$\varphi's, \theta_k's \text{ are all real numbers, } \varphi_p, \theta_q \neq 0$$

결국 시간 t에서의 값 X_t을 과거의 X_t 값들과(AR) white noise 값들(MA)를 동시에 활용해 설명하는 model.

Table 3.1 Behavior of the ACF and PACF of ARMA models

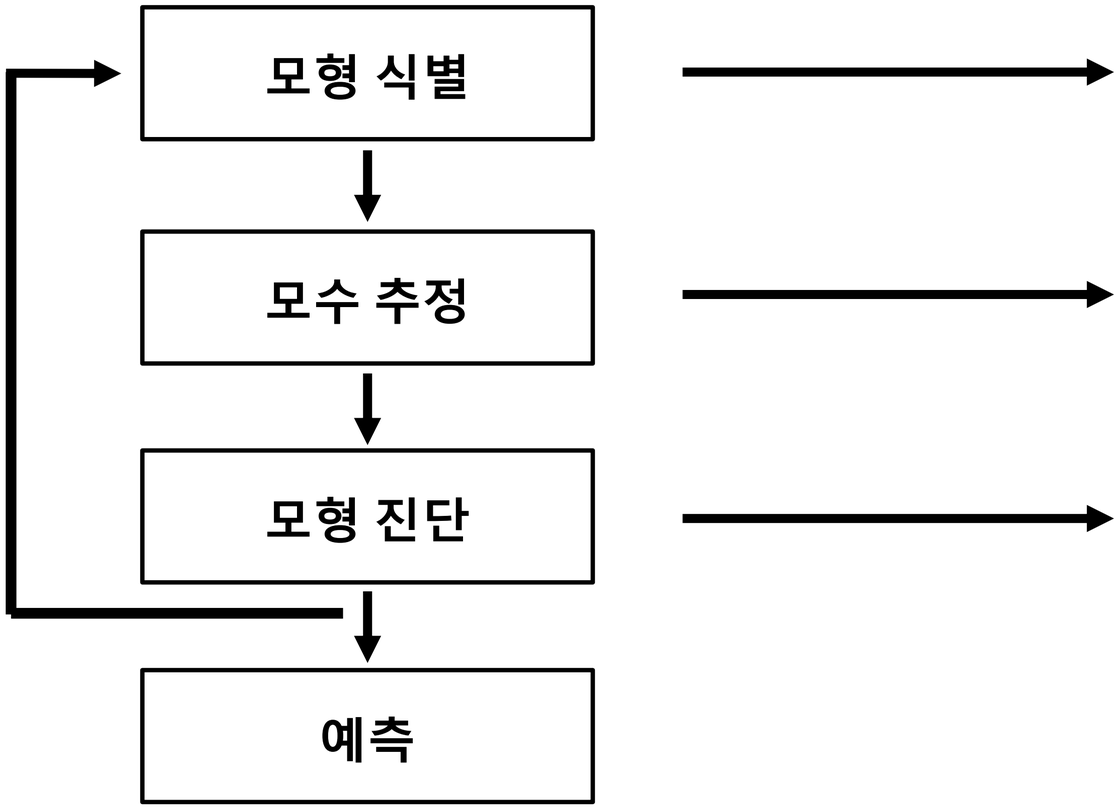
	MA(q)	AR(p)	ARMA(p,q) ($p>0, q>0$)
ACF	Cuts off after lag q	Tails off	Tails off
PACF	Tails off	Cuts off after lag p	Tails off

02

Model Building Problems & Estimation Methods

How to Analyze Times Series Model?

시계열 모형 분석을 위해 **Geroge Box**와 **Gwilyum Jenkins**는 다음과 같은 방법론을 제시했다.



시계열 그래프, **ACF/PACF**를 통한 정상성 판단

→ 모형의 차수(p,q) 결정

Parameter Estimation!

White Noise 여부 등 체크

ESC

ARMA Model

ARMA(p,q) model은 다음과 같은 형태를 띈다.

: $$\mathbf{X_t = \varphi_0 + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}}$$

where $\{\epsilon_t\} \sim \mathrm{WN}(0, \sigma_\epsilon^2)$, $\varphi \neq 0$, $\theta_q \neq 0$, φ_0 is the intercept term(constant).

여기서 식을 조금만 전개하면, 아래와 같이 쓸 수도 있다.

$$\mathbf{X_t - \varphi_1 X_{t-1} - \cdots - \varphi_p X_{t-p} = \varphi_0 + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}}$$

$$\mathbf{\varphi(B) X_t = \varphi_0 + \theta(B) \epsilon_t}$$

where $\varphi(\mathrm{B}) = 1 - \varphi_1 \mathrm{B} - \cdots - \varphi_p \mathrm{B}^p$, $\theta(\mathrm{B}) = 1 + \theta_1 \mathrm{B} + \cdots \theta_q \mathrm{B}^q$

ARMA model의 기본적인 조건은 $\{\mathbf{X_t}\}$라는 시계열이 **정상(stationary) 시계열**이라는 것이다.

그러나, 현실의 데이터에서 처음부터 정상인 시계열은 거의 존재하지 않는다 → **추세**나 **계절성**을 갖는다.

비정상 시계열을 그대로 ARMA model에 fitting 할 순 없으니, 뭔가 조치를 취해야 한다 → **정상화**!

ARIMA Model

추세는 가지지만, **계절성이 존재하지 않는** 시계열 $\{\mathbf{X_t}\}$를 생각해보자.

일반적으로 추세를 제거할 때 사용하는 방법은 당연히 **차분**이다!

차분을 적용해서 새로운 시계열을 하나 만들고 그걸 $\{\mathbf{Y_t}\}$라고 하자.

$\mathbf{Y_t = \nabla^d X_t = (1 - B)^d X_t}$ → $\{\mathbf{X_t}\}$를 d번 차분했다는 뜻! → 이제 $\{\mathbf{Y_t}\}$는 **stationary**하다.

이제 이걸 **ARMA model** 형태로 써보면 아래와 같다.

$\boldsymbol{\varphi(B)Y_t = \varphi_0 + \theta(B)\epsilon_t}$ → 이제 이걸 $\{\mathbf{X_t}\}$에 대해 다시 써보면?

$\boldsymbol{\varphi(B)(1 - B)^d X_t = \varphi_0 + \theta(B)\epsilon_t}$의 형태로 쓸 수 있다!

ARIMA Model

Definition 4.1 (1) Equation (4.2) where $Y_t = (1 - B)^d X_t$ is stationary is called an ARIMA(p, d, q) model. (2) A time series X_t that satisfies Eq. (4.2) is said to be an ARIMA(p, d, q) process or follows an ARIMA(p, d, q) model.

ARIMA = **A**uto**R**egressive **I**ntegrated **M**oving **A**verage

ARIMA model의 특징은 다음과 같다.

1. $\mathbf{d \geq 1}$이면, **ARIMA(p,d,q) Process**는 **non-stationary**하다.

   원래 시계열인 $\{X_t\}$가 비정상이라는 뜻으로 이해!

2. $\mathbf{d = 0}$이면, **ARIMA(p, d, q) Process** → **ARMA(p, q) Process**와 동일하다.

Estimation Methods

시계열 sample $\{\mathbf{X_{1:n}}\}$이 주어졌을 때

ARMA model의 **parameters**를 연구하는 방법은 꽤 오래전부터 연구되어 왔지만,

그 중 일부는 여전히 활발하게 사용되고 있다(**vibrant**).

이제 몇 가지 방법론에 대해 소개할 예정인데, 다음을 가정하자.

$$\mathbf{X_t = \varphi_0 + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}}$$

1. $\boldsymbol{\varphi_0 = 0}$

$\rightarrow \mathbf{X_t = \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}}$

2. $\boldsymbol{\varphi_0 \neq 0}$ $\rightarrow$ Let $\mathbf{Z_t = X_t - \mu}$ where $\boldsymbol{\mu = E[X_t]}$

$\rightarrow \mathbf{Z_t = \varphi_1 Z_{t-1} + \cdots + \varphi_p Z_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}}$

The Innovations Algorithm

결론부터 말하자면, **Innovations Algorithm**은 **최적의 선형 예측**을 **재귀적으로 갱신해나가는 알고리즘**이다.

정상 & 비정상 시계열에서 **second moment가 finite**하다면 사용가능한 방법!**(무슨 소리인지 이해가 안 가는게 정상)**

괜찮아요 따라 오세요~ 우선 아이디어를 먼저 살펴보자.

1. **One-Step-Ahead Linear Prediction**

   어떤 시점 n에서, 과거의 관측지 $\mathbf{X_1}, \mathbf{X_2}, \cdots, \mathbf{X_n}$이 주어졌을 때 다음 시점 $\mathbf{X_{n+1}}$을 예측하려고 한다.

   이 때 선형적인 방법을 통해 예측을 해보면

   $\widehat{\mathbf{X_{n+1}}} = \mathbf{E}[\mathbf{X_{n+1}} \mid \mathbf{X_1}, \cdots, \mathbf{X_n}]$으로 표현할 수 있고, 이를 **one-step (optimal) linear predictor**이라고 한다.

2. **Innovations**

   $\{\mathbf{X_n} - \widehat{\mathbf{X_n}}\}$: **one-step prediction errors**. 사실상 회귀분석의 **잔차(residual)**와 동일.

   이 오차가 **기존 정보(과거의 데이터)와 uncorrelated** 하다면 → **새로운 정보**로 해석 가능!

3. **Recursive**

   매 시점 n에서 이전 시점까지의 예측값과 오차**(innovation)**을 활용해 다음 시점의 **예측 계수**$(\boldsymbol{\theta_{nj}})$**를 갱신**.

The Innovations Algorithm

알고리즘을 수식적으로 살펴보면 다음과 같다.

우선 다음 조건들을 가정하자.

$\{\mathbf{X_t}\}$: time series with **mean zero** and $\mathbf{E}\left[\mathbf{X_t^2}\right] < \infty$, $t \geq 1$

$\kappa(i, j) = E\left[X_i, X_j\right]$: autocovariance function

$\widehat{\mathbf{X_1}} = \mathbf{0}$, $\widehat{\mathbf{X_{j+1}}} = \mathbf{E}\left[\mathbf{X_{j+1}} \mid \mathbf{X_1}, \cdots, \mathbf{X_j}\right], \mathbf{j \geq 2}$: **one-step linear predictor.**

$\mathbf{v_n} = \mathbf{E}\left[(\mathbf{X_{n+1}} - \widehat{\mathbf{X_{n+1}}})^2\right]$: prediction errors(innovation)의 분산(**MSE**) $\rightarrow \mathbf{v_0} = \boldsymbol{\kappa}(\mathbf{1}, \mathbf{1})$

$\left\{\mathbf{X_n} - \widehat{\mathbf{X_{n+1}}}\right\}$: one-step prediction errors = **innovation**

이런 조건들을 만족할 때, 다음 시점의 **linear predictor**을 다음과 같이 재귀적으로 계산할 수 있는 $\boldsymbol{\theta}_{nj}$를 찾을 수 있다.

$$\widehat{\mathbf{X_{n+1}}} = \sum_{j=1}^{n} \boldsymbol{\theta}_{nj}\underbrace{\left(\mathbf{X_{n+1-j}} - \widehat{\mathbf{X_{n+1-j}}}\right)}_{\text{Innovation}} \textbf{ for } \mathbf{n} = 1, 2, \cdots$$

The Innovations Algorithm

$$\widehat{\mathbf{X_{n+1}}} = \sum_{\mathbf{j=1}}^{\mathbf{n}} \boldsymbol{\theta}_{\mathbf{nj}} \underbrace{\left(\mathbf{X_{n+1-j}} - \widehat{\mathbf{X_{n+1-j}}}\right)}_{\textbf{Innovation}} \textbf{ for } \mathbf{n} = \mathbf{1}, \mathbf{2}, \cdots$$

위의 알고리즘을 recursive하게 진행하다 **보면 coefficient $\boldsymbol{\theta}_{\mathbf{nj}}$**와 **MSE인 $\mathbf{v_n}$**을 찾을 수 있다.

→ 이러한 이유 때문에 **prediction** 같아도 **parameter estimation**으로 볼 수 있는 것!

$$\begin{cases} \theta_{n,n-k} = v_k^{-1}\left(\kappa(n+1,k+1) - \sum_{j=0}^{k-1}\theta_{k,k-j}\theta_{n,n-j}v_j\right), \ 0 \le k < n \\ v_n = \kappa(n+1,n+1) - \sum_{j=0}^{n-1}\theta_{n,n-j}^2 v_j . \end{cases}$$

Innovation은 실제로 **ARMA model**에서의 $\boldsymbol{\epsilon_t}$ **term**과 유사한 역할 → **white noise**처럼 각 시점과 **무상관(uncorrelated)**.

$\widehat{\mathbf{X_{n+1}}}$을 과거의 관측지 $\mathbf{X_1}, \mathbf{X_2}, \cdots, \mathbf{X_n}$의 선형결합으로 표현하고, 이 과정에서 $\boldsymbol{\theta}_{\mathbf{nj}}$는 곧 **ARMA의 가중치와 비슷한 역할**.

실제로 $\boldsymbol{\theta_{nj}}$는 ARMA 모델의 **자기상관 구조를 반영**!

또한 **MSE인 $\mathbf{v_n}$**을 통해 **모델이 얼마나 데이터를 잘 설명하는지 측정**할 수 있음.

The Innovations Algorithm

마지막으로 **Innovation**의 **uncorrelation**에 대해 알아 보자.

→ **Innovation**은 **과거 데이터로 이 부분은 도저히 예측이 불가능한 순수한 무언가**. (**Prewhitening**이라고도 불린다.)

1. 직관적인 이해

   Innovation은 MSE를 최소화하는 **optimal linear predictor** → 과거의 데이터에서 **추가적인 정보를 더 얻을 여지가 없다**.

   혁신이라는 말처럼 **과거의 정보를 모두 반영해서 최적의 예측값을 만들었으므로**, 오차는 과거 정보를 반영하지 않는다.

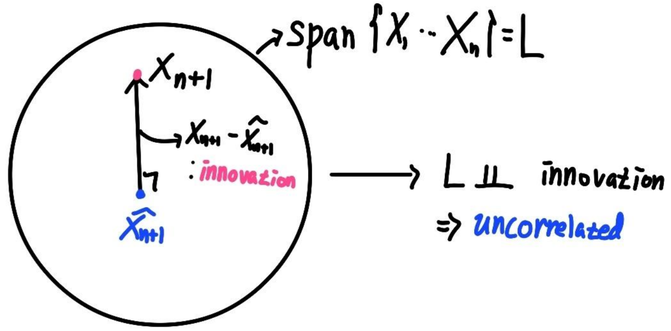
   스펀지가 물을 먹었다 → 얼마나 먹었을까? 정확히 알 수 없음: $\mathbf{X_{n+1}}$

   있는 힘껏 물을 짜내자 → 최적 예측값: $\widehat{\mathbf{X_{n+1}}}$**(과거 데이터)**

   스펀지를 다 짜고 남은 물의 양: **innovation!** = $\mathbf{X_{n+1} - \widehat{X_{n+1}}}$

   근데 이걸 가지고 스펀지에 **원래 물이 얼마만큼 들어있었겠군**~이라고 생각하는 걸 말도 안됨! → **uncorrelated!**

2. 회귀분석의 관점에서 이해



Method of Moments

기본적인 아이디어는 다음과 같다.

모집단의 모수는 절대로 **정확히 알 수 없다**. → **sample로 estimate**하는 것이 우리가 할 수 있는 일!

복잡한 계산 없이 할 수 있다는 장점이 있다.

1. **모집단의 moment**를 **unknown parameter의 함수**로 표현한다.

2. **Sample의 moment**를 구한다.

3. **모집단의 moment = sample의 moment**라고 두고 **parameter을 추정**한다.

Method of Moments

Sample autocovariance와 **sample autocorrelation**은 일반적으로 다음과 같이 정의된다.

$$\hat{\gamma}_k = c_k = \frac{1}{n}\sum_{t=1}^{n-k}\left(X_{t+k} - \bar{X}\right)\left(X_t - \bar{X}\right) \text{ and } \hat{\rho}_k = r_k = \frac{c_k}{c_0}$$

여기서 $\bar{X}$: **sample moment**이면서 **moment estimate of the mean**.

$\widehat{\gamma_k}$: **sample moment**이면서 **moment estimate of the autocovariance**.

$\widehat{\rho_k}$: **function of sample moment**이면서 **moment estimate of the autocorrelation**.

특히 $\widehat{\gamma_0}$는 $\mathrm{Var}(X_t) = \gamma_0$ 의 **estimate**!

이걸 이제 **AR(p) model**에 적용시켜보자.

Method of Moments

AR(p) Model Example)

$$\text{AR Model of order p}$$
$$AR(p): X_t = \varphi_0 + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \epsilon_t$$
$$\text{where } \{w_t\} \sim^{iid} WN(0, \sigma_\epsilon^2), E[X_s \epsilon_t] = 0 \text{ if } s < t,$$
$$\varphi' s \text{ are all real numbers, } \varphi_p \neq 0$$

AR(p) model에서는 몇 번의 식 조작을 통해 **Yule-Walker Equation**을 도출할 수 있었다(1주차).

$$\varphi_1 \rho_{k-1} + \varphi_2 \rho_{k-2} + \cdots + \varphi_p \rho_{k-p} = \rho_k, \ 1 \leq k \leq p$$

$$\sigma_\epsilon^2 = \gamma_0 (1 - \varphi_1 \rho_1 - \varphi_2 \rho_2 - \cdots - \varphi_p \rho_p).$$

이제 **sample data**로부터 $\widehat{\boldsymbol{\rho}_\mathbf{k}} = \boldsymbol{\gamma}_\mathbf{k} = \frac{\widehat{\gamma_k}}{\widehat{\gamma_0}}$를 계산하고 $\boldsymbol{\rho}_\mathbf{k} = \widehat{\boldsymbol{\rho}_\mathbf{k}}$로 두고 풀면

AR(p) model의 파라미터인 $\boldsymbol{\varphi_1, \cdots, \varphi_p}$와 $\boldsymbol{\sigma_\epsilon^2}$를 구할 수 있다.

이 과정을 통해 **Yule-Walker Estimators**를 구할 수 있다.

Method of Moments

방금 식들을 **matrix form**으로 나타내면 아래와 같다.

$$\boldsymbol{\rho}\varphi = \rho \text{ and } \sigma_\epsilon^2 = \gamma_0(1 - \varphi'\rho)$$

where

$$\boldsymbol{\rho} = \begin{pmatrix} \rho_0 & \rho_1 & \cdots & \rho_{p-1} \\ \rho_1 & \rho_0 & \cdots & \rho_{p-2} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_{p-1} & \rho_{p-2} & \cdots & \rho_0 \end{pmatrix}, \ \varphi = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_p \end{pmatrix}, \ \rho = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_p \end{pmatrix}$$

The Yule-Walker estimators are

$$\hat{\varphi} = \hat{\boldsymbol{\rho}}^{-1}\hat{\rho} = \boldsymbol{R}^{-1}r \text{ and } \hat{\sigma}_\epsilon^2 = \hat{\gamma}_0(1 - r'\boldsymbol{R}^{-1}r)$$

where

$$\boldsymbol{R} = \begin{pmatrix} 1 & r_1 & \cdots & r_{p-1} \\ r_1 & 1 & \cdots & r_{p-2} \\ \vdots & \vdots & \vdots & \vdots \\ r_{p-1} & r_{p-2} & \cdots & 1 \end{pmatrix}, \ r = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{pmatrix}.$$

MA(q)나 ARMA(p, q) model에서도 비슷한 방식으로 moment를 이용한 estimation이 가능하다.

그러나 식이 조금 더 복잡해져서 직접 방정식을 풀기보단 **Innovation Algorithm**이나 **Hannan-Rissanen algorithm**을 쓴다.

Method of Maximum Likelihood

시계열 자료가 normal White Nosie를 따른다고 가정하고 MLE를 통해 parameter를 추정하자!

다음을 가정하자.

$\{\mathbf{X_t}\}$가 **평균이 0**이고 **autocovariance function** $\boldsymbol{\kappa(i,j) = E[X_i X_j]}$인 **Gaussian Time Series** 라고 하자.

$\boldsymbol{X_n = (X_1\ X_2 \cdots X_n)'}$, $\boldsymbol{\widehat{X_n} = (\widehat{X_1}\ \widehat{X_2} \cdots \widehat{X_n})'}$, where $\boldsymbol{\widehat{X_1} = 0}$ and $\boldsymbol{\hat{X}_j = E[X_j \mid X_1, \cdots, X_{j-1}], j \geq 2}$

$\boldsymbol{\Gamma_n = E[X_n\ X_n']}$: **autocovariance matrix**

$$\rightarrow \qquad L(\Gamma_n) = (2\pi)^{-\frac{n}{2}} (\det \Gamma_n)^{-\frac{1}{2}} \exp(-\frac{1}{2} X_n' \Gamma_n^{-1} X_n)$$

$$\rightarrow \qquad L(\Gamma_n) = (2\pi)^{-\frac{n}{2}} (v_0 v_1 \cdots v_{n-1})^{-\frac{1}{2}} \exp\left[-\frac{1}{2} \sum_{j=1}^{n} (X_j - \hat{X}_j)^2 / v_{j-1} \right]$$

Method of Maximum Likelihood

여기서 $\{X_t\}$가 **causal하고 invertible**하다는 조건이 추가되면 다음과 같이 **ARMA(p, q) model**을 적을 수 있다.

$$\varphi(B)X_t = \theta(B)\varepsilon_t, \{\varepsilon_t\} \sim \mathrm{WN}(0, \sigma_\epsilon^2)$$

where $\varphi(z) = 1 - \varphi_1 z - \cdots - \varphi_p z^p$, $\theta(z) = 1 + \theta_1 z + \cdots + \theta_q z^q$. Let $\varphi = (\varphi_1\ \varphi_2\ \cdots\ \varphi_p)'$, $\theta = (\theta_1\ \theta_2\ \cdots\ \theta_q)'$ and $\beta = (\varphi, \theta)'$

$$L(\beta, \sigma_\epsilon^2) = (2\pi)^{-\frac{n}{2}}\sigma_\epsilon^{-n}[r_0(\beta)r_1(\beta)\cdots r_{n-1}(\beta)]^{-\frac{1}{2}} \exp\left[-\frac{S(\beta)}{2\sigma_\epsilon^2}\right]$$

$$S(\beta) = \sum_{j=1}^{n}[(X_j - \hat{X}_j(\beta))^2/r_{j-1}(\beta)]$$

and $r_j(\beta) = v_j/\sigma_\epsilon^2$, $\hat{X}_j(\beta) = \hat{X}_j$

이후에는 로그 **씌우고** $\boldsymbol{\beta = (\varphi_1, \cdots, \varphi_p, \theta_1, \cdots, \theta_q, \sigma_\epsilon^2)}$에 대해 **argmax**하면 끝!

대부분 Newton-Raphson 이용.

Gaussian 가정에서는 **MLE가 MVUE에 근접**하고, **large sample**에서는 **asymptotic normality를** 만족.

심지어 시계열이 **정규분포가 아니더라도** $\boldsymbol{\epsilon \sim iid(0, \sigma_\epsilon^2)}$**만 만족**하면 **MLE가 여전히 좋은 추정량**으로 이용됨.

차수가 커지면 β의 차원도 올라가고 계산비용도 늘어남.

→ **Durbin-Levinson**이나 **Innovation Algorithm**으로 계산량을 줄이며 효율적으로 계산.

03

Order Determination

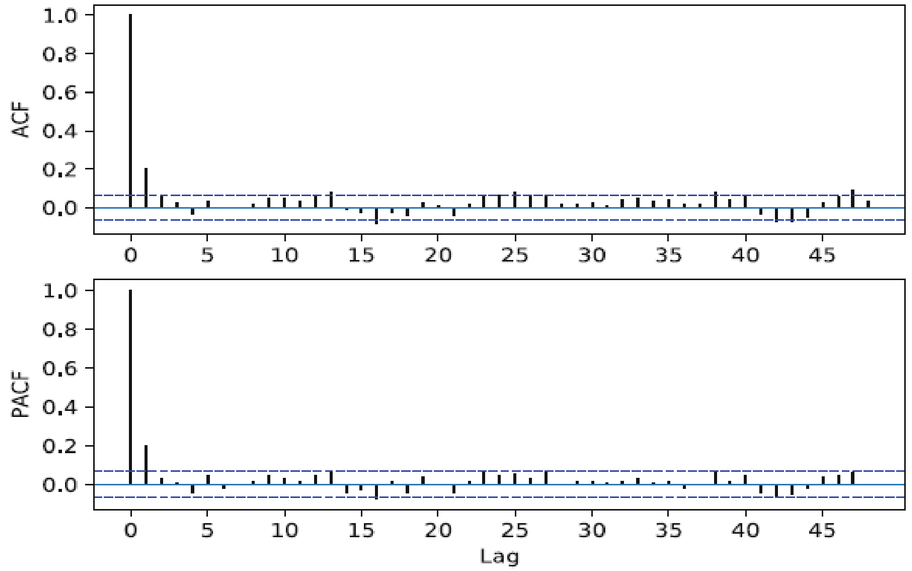
What is Order Determination?

Table 3.1 Behavior of the ACF and PACF of ARMA models

	MA(q)	AR(p)	ARMA(p,q) ($p>0, q>0$)
ACF	Cuts off after lag q	Tails off	Tails off
PACF	Tails off	Cuts off after lag p	Tails off

-> ARMA(p, q) 모델의 차수 (p, q)

선택: **ACF와 PACF의 개형을 이용**



[Example 4.1]

[Example 4.1]

-ACF가 lag 1 이후 절단이니 MA(1) 모형?

-PACF도 lag 1 이후 절단처럼 생겼는데 그럼 AR(1) 모형?

:지수적 감소, 특정시점 이후 절단등을 눈으로만 파악하는 데는 모호한 점이 많음!!

+ 둘 다 지수적 감소라고 판단해버리면 p, q을 결정할 방법이 없음..

-> 추가적인 기준(Information Criterion) 을 도입하여,

모델의 차수(Order)을 결정(Determine)해보자!

세가지 기준: ① AIC ② BIC ③ HQIC

① AIC (Akaike Information Criterion)

$$AIC = -2\ln(L)+2K$$

L: Maximum Likelihood, K: Number of Estimated Parameters

AIC의 값이 작을수록 모델이 더 적합하다고 평가

: -2ln(L)은 모형의 적합도를 의미하는 부분이고,

2k는 파라미터가 클수록 해당 모형에 패널티를 주기 위해 존재한다고 볼 수 있음

-> Likelihood를 최대화하는 동시에 모형은 간단한 것을 고르는 지표!

② BIC (Bayesian Information Criterion)

BIC= -2ln(L)+Kln(N)

L: Maximum Likelihood, K: Number of Estimated Parameters, N: Sample Size

-AIC의 parameter 개수에 의한 **penalty가 2에서 lnN이 됨**

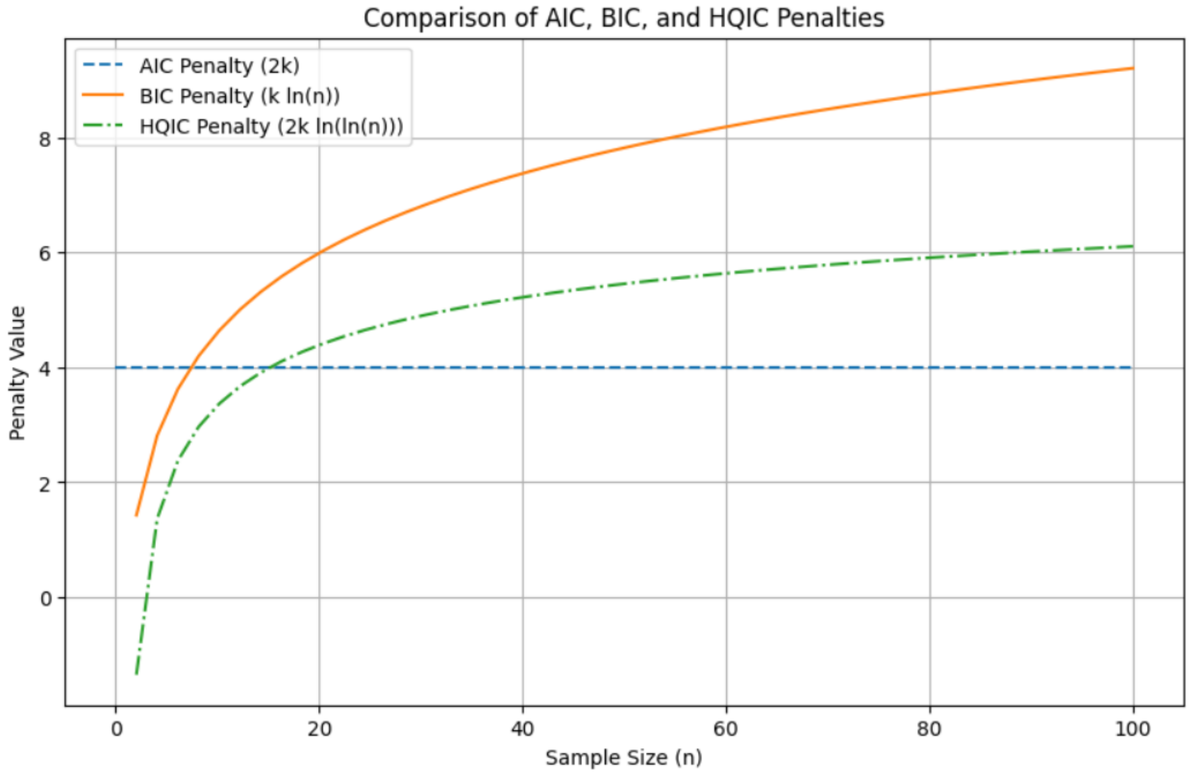
: n이 7보다 큰 경우 BIC는 AIC보다 더 높은 penalty를 부여

	AIC	BIC
공식	**-2ln(L)+2K**	**-2ln(L)+KlnN**
패널티 크기	비교적 작음**(2)**	데이터 크기에 따라 증가**(lnN)**
샘플 크기 **n**이 클때	복잡한 모델을 허용	**ln(N)**이 크므로 더 단순한 모델을 선택
과적합가능성	**AIC**에서 더 높음	

③HQIC (Hannan-Quinn Information Criterion)

HQIC= -2ln(L)+2Kln(ln(N))

L: Maximum Likelihood, K: Number of Estimated Parameters, N: Sample Size



-penalty가 2ln(ln(N))으로, AIC보다는 크지만 BIC보다는 작다고 알려져 있음

: AIC와 BIC의 중간(절충안)이라고 볼 수 있음!

Example: Information Criterion for AR(1) Model

Step 1. AR(1)을 따르는 시계열 데이터 만들기

```
# 샘플 데이터 (X) 생성
np.random.seed(42)
n = 100  # 샘플 크기
phi_true = 0.7  # 가정한 AR(1) 계수
sigma_true = np.sqrt(0.5)  # 가정한 표준편차 (sqrt(0.5))
X = np.zeros(n)
epsilon = np.random.normal(0, sigma_true, n)

for t in range(1, n):
    X[t] = phi_true * X[t-1] + epsilon[t]
```

[복습] AR(1) Model

$$X_t = \phi X_{t-1} + \epsilon_t$$

- ϵ_t 는 white noise로 평균이 0이고 분산이 σ^2인 정규분포를 따름 [$\epsilon_t \sim N(0, \sigma^2)$]

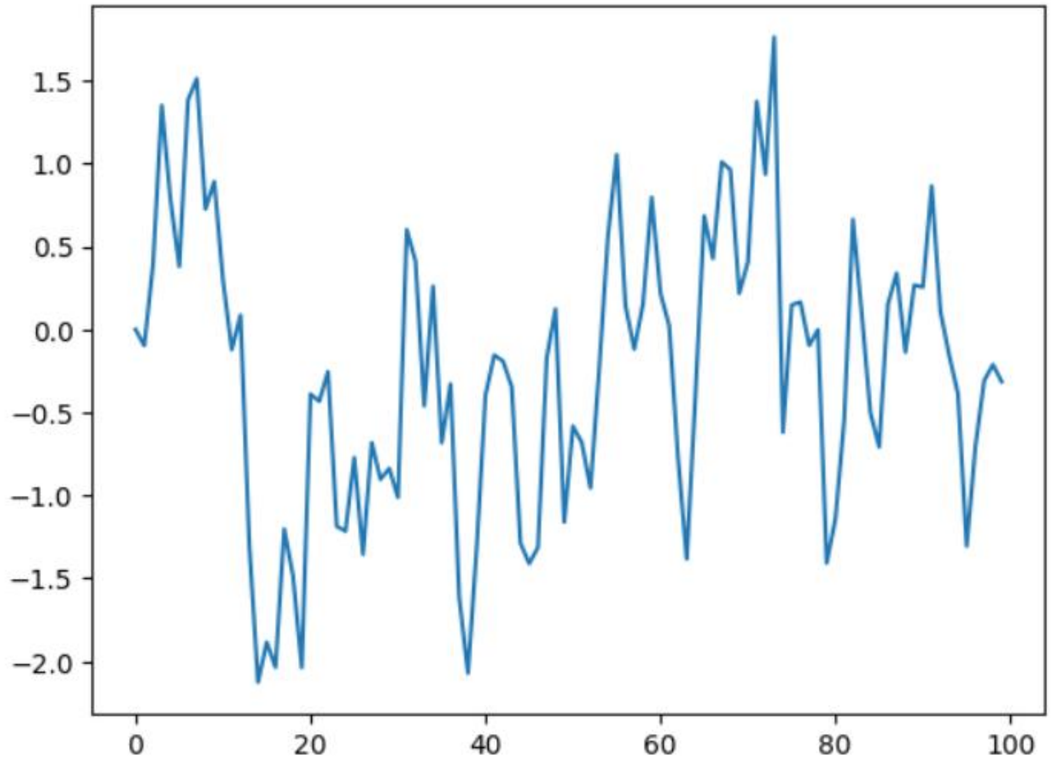
-정상성을 따르려면 $|\varphi| < 1$

-> 데이터 수 100, 회귀계수 0.7, 분산 0.5 가정한 AR(1) 모형 작성코드

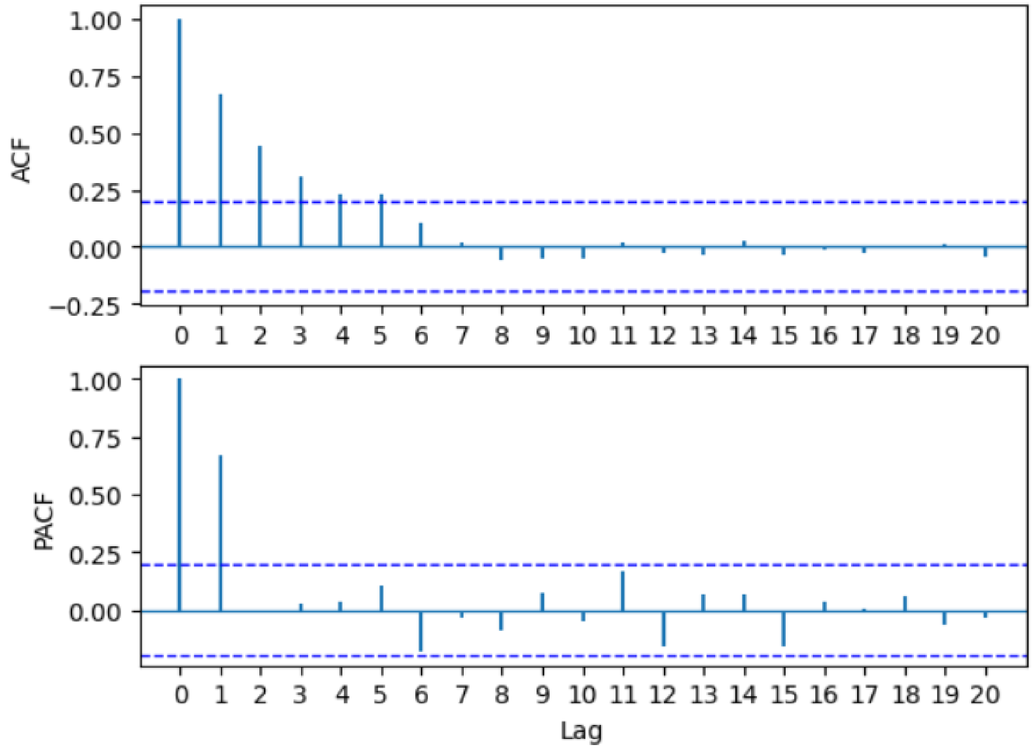
Example: Information Criterion for AR(1) Model

Step 1. AR(1)을 따르는 시계열 데이터 만들기

```
X = pd.Series(X)
X.plot(); plt.show()
```




```
acf_pacf_fig(X, both = True, lag = 20)
plt.show()
```



Example: Information Criterion for AR(1) Model

Step 2. AR(1) Model의 Likelihood 구하기

[복습] Likelihood Function: $L(\theta; \boldsymbol{x}) = \Pi_{i=1}^{n} f(x_i; \theta), \quad \theta \in \Omega,$

-X_1~ X_n 의 joint pdf는 베이즈 정리를 반복적으로 적용하면

$$f(X_n, X_{n-1}, \ldots, X_1) = f(X_n | X_{n-1}, \ldots, X_1) f(X_{n-1}, X_{n-2}, \ldots, X_1)$$

$$= f(X_n | X_{n-1}, \ldots, X_1) f(X_{n-1} | X_{n-2}, \ldots, X_1) f(X_{n-2}, X_{n-3}, \ldots, X_1)$$

$$= f(X_n | X_{n-1}, \ldots, X_1) f(X_{n-1} | X_{n-2}, \ldots, X_1) f(X_{n-2} | X_{n-3}, \ldots, X_1) \ldots f(X_2 | X_1) f(X_1)$$

최종적으로 $f(X_n, X_{n-1}, \ldots, X_1) = f(X_1) \prod_{t=2}^{n} f(X_t | X_{t-1}, \ldots, X_1)$ 이 된다.

Example: Information Criterion for AR(1) Model

Step 2. AR(1) Model의 Likelihood 구하기

-AR(1) model: $X_t = \phi X_{t-1} + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2)$

-조건부 기댓값과 분산 $E(X_t|X_{t-1}, X_{t-2}, \ldots, X_1) = \phi X_{t-1}$

$$V(X_t|X_{t-1}, X_{t-2}, \ldots, X_1) = \sigma^2$$

-오차항이 정규분포를 이루므로 X_t의 조건부 분포도 정규분포를 따른다.

-따라서, 조건부분포의 pdf는 아래와 같다.

$$f(X_t|X_{t-1}, X_{t-2}, \ldots, X_1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(X_t - \phi X_{t-1})^2}{2\sigma^2}\right)$$

Example: Information Criterion for AR(1) Model

Step 2. AR(1) Model의 Likelihood 구하기

-AR(1) model: $X_t = \phi X_{t-1} + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2)$

-초기분포 $f(X_t)$를 구해보자. X_t 는 정규분포를 따르며, 평균은 0이고 분산은

$Var(X_t) = Var(\varphi X_{t-1} + \varepsilon_t)$

$= \varphi^2 Var(X_{t-1}) + 2\varphi Cov(X_{t-1}, \varepsilon_t) + Var(\varepsilon_t) = \varphi^2 Var(X_{t-1}) + 0 + \sigma^2$

정상시계열이므로 $Var(X_t) \cong Var(X_{t-1})$ 임을 이용하면 $Var(X_t) = \frac{\sigma^2}{1-\varphi^2}$ [정상성: 분산은 t에 의존하지 않음]

따라서, $X_1 \sim N\left(0, \frac{\sigma^2}{1-\phi^2}\right)$ 이므로 pdf는

$$f(X_1) = \frac{1}{\sqrt{2\pi\sigma^2/(1-\phi^2)}} \exp\left(-\frac{X_1^2}{2\sigma^2/(1-\phi^2)}\right)$$

Example: Information Criterion for AR(1) Model

Step 2. AR(1) Model의 Likelihood 구하기

- $f(X_1) = \frac{1}{\sqrt{2\pi\sigma^2/(1-\phi^2)}} \exp\left(-\frac{X_1^2}{2\sigma^2/(1-\phi^2)}\right)$ $\quad f(X_t|X_{t-1}, X_{t-2}, \ldots, X_1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(X_t - \phi X_{t-1})^2}{2\sigma^2}\right)$

- $f(X_n, X_{n-1}, \ldots, X_1) = f(X_1) \prod_{t=2}^{n} f(X_t|X_{t-1}, \ldots, X_1)$ 에 대입하면

$$L(\phi, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2/(1-\phi^2)}} \exp\left(-\frac{X_1^2}{2\sigma^2/(1-\phi^2)}\right) \times \prod_{t=2}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(X_t - \phi X_{t-1})^2}{2\sigma^2}\right)$$

- $\log L(\phi, \sigma^2) = -\frac{1}{2}\log\left(2\pi\sigma^2/(1-\phi^2)\right) - \frac{X_1^2}{2\sigma^2/(1-\phi^2)} - \frac{(n-1)}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{t=2}^{n}(X_t - \phi X_{t-1})^2$

Example: Information Criterion for AR(1) Model

Step 3. Log-Likelihood Function Python으로 구현하기

$$\log L(\phi, \sigma^2) = -\frac{1}{2}\log\left(2\pi\sigma^2/(1-\phi^2)\right) - \frac{X_1^2}{2\sigma^2/(1-\phi^2)} - \frac{(n-1)}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{t=2}^{n}(X_t - \phi X_{t-1})^2$$

```python
# AR(1) 모델의 로그 우도 함수 도출
def log_likelihood_ar1(X, phi, sigma2):
    n = len(X)

    # 초기 분포의 로그 우도 (정상성을 가정한 경우)
    log_L1 = -0.5 * np.log(2 * np.pi * sigma2 / (1 - phi**2)) - (X[0]**2 / (2 * sigma2 / (1 - phi**2)))

    # 조건부 분포의 로그 우도 합산
    residuals = X[1:] - phi * X[:-1]  # Xt - phi * Xt-1
    SSE = np.sum(residuals**2)  # 잔차 제곱합

    log_L2 = - (n-1)/2 * np.log(2 * np.pi * sigma2) - (1 / (2 * sigma2)) * SSE

    # 총 로그 우도 값
    log_L = log_L1 + log_L2

    return log_L
```

Example: Information Criterion for AR(1) Model

Step 4. 모수의 추정치를 이용하여 log-likelihood 구하면 끝

```
# MLE로 추정한 파라미터 (phi_hat, sigma2_hat) 사용
phi_hat = np.sum(X[1:] * X[:-1]) / np.sum(X[:-1]**2)  # phi 추정
sigma2_hat = np.var(X[1:] - phi_hat * X[:-1], ddof=1)  # sigma^2 추정 (불편분산 사용)

# 로그 우도 계산
log_L = log_likelihood_ar1(X, phi_hat, sigma2_hat)

# AIC 및 BIC 계산
k = 2  # 추정된 파라미터 개수 (phi, sigma2)
AIC = -2 * log_L + 2 * k
BIC = -2 * log_L + k * np.log(n)
HQIC= -2 * log_L + 2 * k * np.log(np.log(n))
# 결과 출력
print("AIC:", AIC)
print("BIC:", BIC)
print("HQIC:", HQIC)
```

```
AIC: 199.87926359973983
BIC: 205.08960397171 6
HQIC: 201.98798210297144
```

Automatic Order Determination(Python)

-가장 단순한 AR(1) 모형 조차도 Likelihood 구하는 데 한참 걸린다!

-AIC, BIC, HQIC의 정보 기준을 종합하여 ARMA 모델의 **최적 차수 (p, q)를 알아서 찾아주는 함수가 두가지 존재!**

① statsmodels 라이브러리의 sm.tsa.arma_order_select_ic 함수

```
sm.tsa.arma_order_select_ic (data, max_ar, max_ma, ic, trend)
```

-max_ar, max_ma에는 AR, MA의 최대 차수를 지정해줌 (기본값: 4)

-ic는 최적 차수를 찾는데 사용할 정보 기준으로, ic=['aic', 'bic'] 같이 사용하면 된다.

-trend는 ARMA 모형에서 상수항의 포함 여부로, trend='c'면 상수 포함, trend='n'이면 상수 미포함이다.

Automatic Order Determination(Python)

②PythonTsa 패키지의 choose_arma 함수

choose_arma(data, max_p, max_q, ctrl)

-max_p, max_q에는 AR, MA의 최대 차수를 지정해줌 (기본값: p=6, q=7)

-ctrl은 제어 계수로, 1.0<=ctrl<=1.1 범위에서 설정함 (일반적으로 1.05 이내가 적당함)

-이는 모델이 stationary 및 invertible을 유지할 수 있도록 제어하는 역할을 함.

: ARMA 모형의 정상성 또는 가역 만족조건으로 특성방정식의 모든 근이 1보다 커야 한다는 것이 있었는데,

근이 1에 거의 가까우면 정상성 또는 가역을 만족한다고 보기 어려울 수 있음.

-> **특성방정식의 근이 ctrl 값보다 작아지면** 정상 또는 가역을 만족하지 않을 수 있다고 보고 다른 모델을 채택함!

Automatic Order Determination: Example 4.2

Simulating and Building an ARMA(2, 2) Model

-ARMA(2, 2) 모델을 따르는 가상 시계열 데이터를 생성하고, 이 데이터를 통해 최적의 ARMA 모델을 선택해보자!

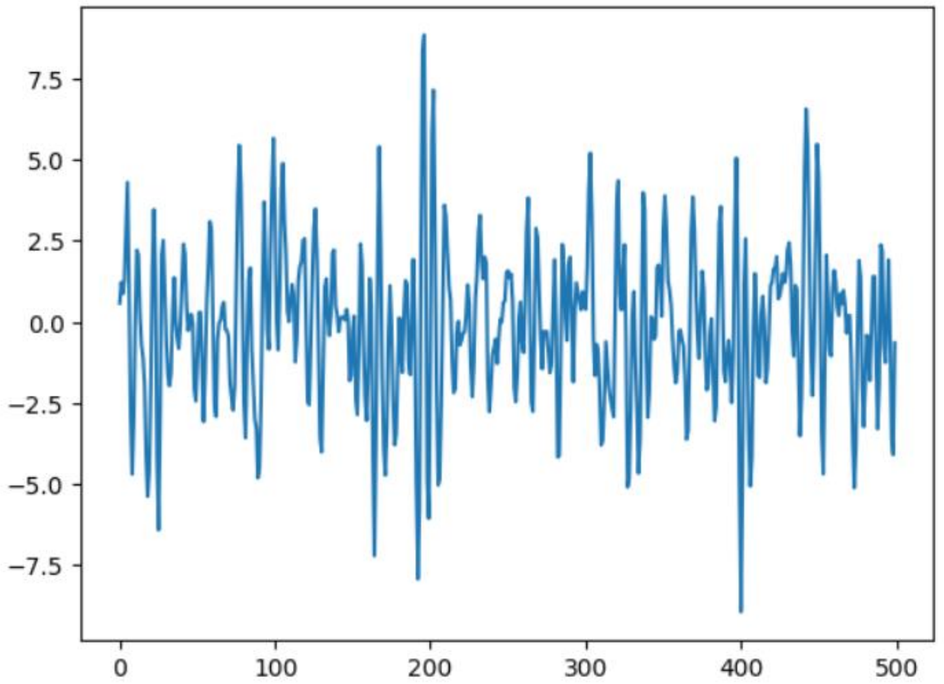
```
# Step 1: 라이브러리 불러오기
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.tsa.arima_process import arma_generate_sample
from PythonTsa.plot_acf_pacf import acf_pacf_fig
from PythonTsa.Selecting_arma import choose_arma
from statsmodels.tsa.arima.model import ARIMA
from PythonTsa.LjungBoxtest import plot_LB_pvalue
from scipy import stats
```

Automatic Order Determination: Example 4.2

Simulating and Building an ARMA(2, 2) Model

```
# Step 2: ARMA(2, 2) 데이터 생성

# AR, MA 계수 정의
ar = np.array([1, -0.8, 0.6])  # AR(2) 부분
ma = np.array([1, 0.7, 0.4])   # MA(2) 부분

# 랜덤 시드 고정
np.random.seed(12357)

# 500개의 샘플 데이터 생성
y = arma_generate_sample(ar=ar, ma=ma, nsample=500)
y = pd.Series(y, name="y")
```

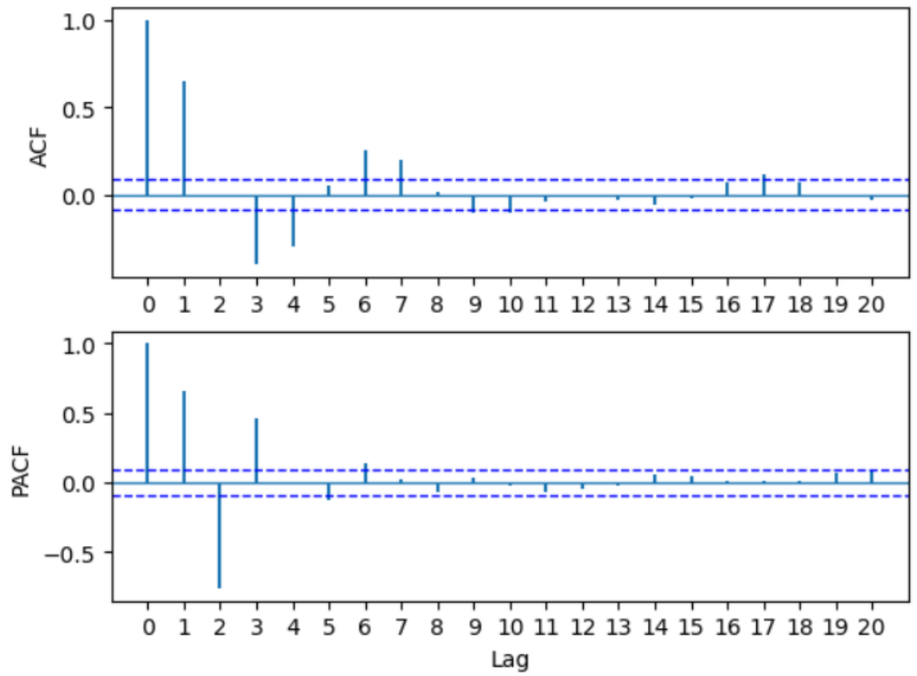
```
# 시계열 데이터 플롯
y.plot(); plt.show()
```



Automatic Order Determination: Example 4.2

Simulating and Building an ARMA(2, 2) Model

```
# Step 3: ACF/PACF 분석
acf_pacf_fig(y, both=True, lag=20)
plt.show()
```



-ACF: 지수적 감소? lag1 이후 절단?

-PACF: 지수적 감소? lag3 이후 절단?

애매하니 Information Criterion을 이용하자!

Automatic Order Determination: Example 4.2

Simulating and Building an ARMA(2, 2) Model

```
# Step 4: Order Determination by sm.tsa.arma_order_select_ic
inf = sm.tsa.arma_order_select_ic(y, max_ar=6, max_ma=7, ic=['aic', 'bic', 'hqic'], trend='n')

print("AIC 기준 최적 차수:", inf.aic_min_order)
print("BIC 기준 최적 차수:", inf.bic_min_order)
print("HQIC 기준 최적 차수:", inf.hqic_min_order)
```

```
AIC 기준 최적 차수: (4, 5)
BIC 기준 최적 차수: (2, 2)
HQIC 기준 최적 차수: (2, 2)
```

참고) AIC에서는 확실히 parameter가 많은 모델을 최적 차수로 제시하는 모습!

Automatic Order Determination: Example 4.2

Simulating and Building an ARMA(2, 2) Model

```
# Step 5: Order Determination by choose_arma
from PythonTsa.Selecting_arma2 import choose_arma2
choose_arma2(y, max_p=6, max_q=7, ctrl=1.02)
```

-choose_arma는 (0, 0)~(6, 7) 사이의 모든 AIC/BIC/HQIC 값을 알아서 구해준다!

-최종적으로 ARMA(2, 2) 모델을 선택

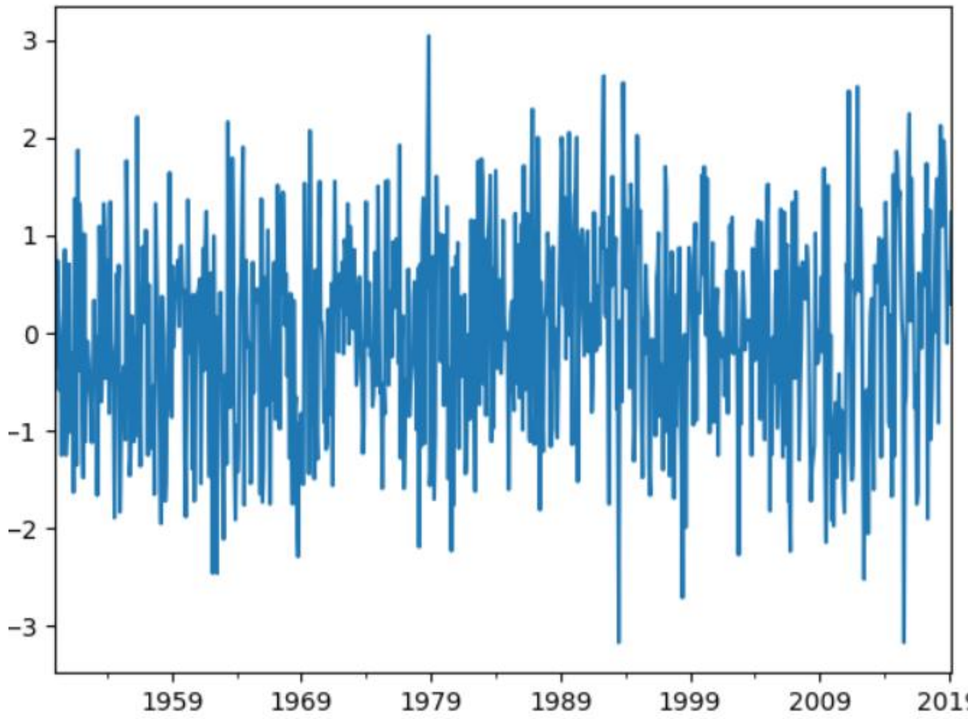
참고) choose_arma 함수가 불안정하게 작동하는 경우가 많아, 개선된 ARMA 차수를 선택해주는 choose_arma2를 이용하는 것이 권장됨.

```
AIC:
          0        1        2        3        4        5        6        7
0       NaN  1859.36  1619.67  1529.46  1531.44  1491.31  1478.01  1479.48
1   2030.95  1728.36  1582.53      NaN  1510.55  1486.01  1479.85  1474.95
2   1594.04  1511.55  1456.82  1457.63  1459.59  1460.75  1462.64  1462.78
3   1475.13  1477.12  1457.68  1459.22  1461.24  1461.91  1463.80  1464.76
4   1477.11  1475.17  1459.67      NaN  1462.48      NaN  1465.90      NaN
5   1469.46  1467.28  1460.79  1462.30  1464.28  1466.13      NaN  1469.67
6   1462.48  1464.14  1462.64  1464.29  1466.12  1467.73      NaN  1466.56
AIC minimum is 1456.82
(p, q)= (array([2]), array([2]))
BIC:
          0        1        2        3        4        5        6        7
0       NaN  1872.01  1636.53  1550.54  1556.73  1520.81  1511.73  1517.41
1   2043.59  1745.22  1603.60      NaN  1540.05  1519.72  1517.78  1517.10
2   1610.90  1532.63  1482.11  1487.13  1493.31  1498.68  1504.79  1509.14
3   1496.20  1502.41  1487.18  1492.94  1499.17  1504.06  1510.16  1515.33
4   1502.40  1504.67  1493.39      NaN  1504.63      NaN  1516.47      NaN
5   1498.96  1501.00  1498.73  1504.44  1510.64  1516.71      NaN  1528.68
6   1496.19  1502.08  1504.78  1510.65  1516.70  1522.52      NaN  1529.78
BIC minimum is 1482.11
(p, q)= (array([2]), array([2]))
HQIC:
          0        1        2        3        4        5        6        7
0       NaN  1864.32  1626.28  1537.73  1541.36  1502.88  1491.24  1494.37
1   2035.91  1734.97  1590.80      NaN  1522.13  1499.24  1494.73  1491.49
2   1600.66  1519.82  1466.75  1469.21  1472.82  1475.63  1479.18  1480.97
3   1483.40  1487.04  1469.25  1472.45  1476.12  1478.45  1481.99  1484.60
4   1487.04  1486.75  1472.91      NaN  1479.02      NaN  1485.74      NaN
5   1481.03  1480.51  1475.68  1478.83  1482.48  1485.98      NaN  1492.83
6   1475.71  1479.03  1479.18  1482.48  1485.97  1489.23      NaN  1491.36
HQIC minimum is 1466.75
(p, q)= (array([2]), array([2]))
```

Order Determination: Example 4.1

북대서양 진동 지수(NAO Index) 시계열데이터의 ARMA 모델 구축

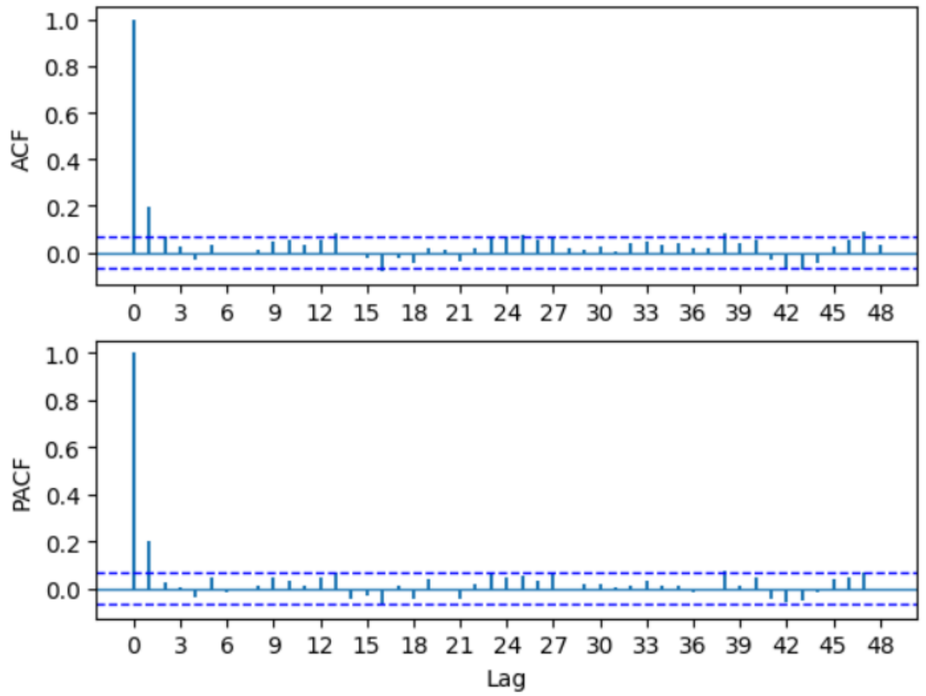
```
# Step 1: 시계열 데이터 불러오기
# 데이터 불러오기
nao = pd.read_csv('nao.csv', header=0)

# 날짜 인덱스 설정
timeindex = pd.date_range('1950-01', periods=len(nao), freq='M')
nao.index = timeindex
naots = nao['index']

# 시계열 데이터 플롯
naots.plot()
plt.show()
```



Order Determination: Example 4.1

북대서양 진동 지수(NAO Index) 시계열데이터의 ARMA 모델 구축

```
# Step 2: ACF/PACF 분석
acf_pacf_fig(naots, both=True, lag=48)
plt.show()
```



-ACF: lag1 이후 절단?

-PACF: 이것도 lag1 이후 절단?

AR(1)이나 MA(1) 둘 중 하나일 거 같은데

애매하니 Information Criterion을 이용하자!

Order Determination: Example 4.1

북대서양 진동 지수(NAO Index) 시계열데이터의 ARMA 모델 구축

```
# AR(1) 모델 적합: 상수(절편)가 유의미하지 않으므로 상수=0으로 fit
ar1 = ARIMA(naots, order=(1,0,0), trend='n').fit()
print(ar1.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                  index   No. Observations:                  831
Model:                 ARIMA(1, 0, 0)   Log Likelihood               -1176.011
Date:                Tue, 11 Mar 2025   AIC                           2356.022
Time:                        01:21:22   BIC                           2365.467
Sample:                    01-31-1950   HQIC                          2359.644
                         - 03-31-2019
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.1996      0.033      6.042      0.000       0.135       0.264
sigma2         0.9924      0.054     18.510      0.000       0.887       1.098
===================================================================================
Ljung-Box (L1) (Q):                   0.03   Jarque-Bera (JB):                 5.97
Prob(Q):                              0.87   Prob(JB):                         0.05
Heteroskedasticity (H):               1.01   Skew:                            -0.12
Prob(H) (two-sided):                  0.91   Kurtosis:                         2.65
===================================================================================
```

```
# MA(1) 과 비교
ma1 = ARIMA(naots, order=(0,0,1), trend='n').fit()

# AIC, BIC, HQIC 비교
print("AR(1) AIC:", ar1.aic, " BIC:", ar1.bic, " HQIC:", ar1.hqic)
print("MA(1) AIC:", ma1.aic, " BIC:", ma1.bic, " HQIC:", ma1.hqic)
```

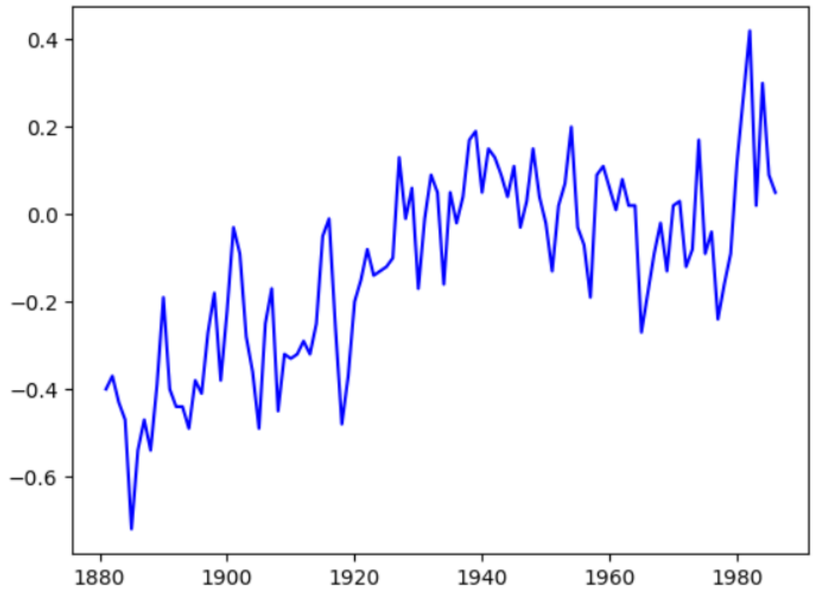
```
AR(1) AIC: 2356.0217979990257  BIC: 2365.4670575887367  HQIC: 2359.643715665035
MA(1) AIC: 2358.745612936791  BIC: 2368.190872526502  HQIC: 2362.3675306028003
```

-후보모형 AR(1)과 MA(1) 모형 중 AR(1) 쪽이 AIC, BIC, HQIC 모두 작으니 AR(1)을 채택!

Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

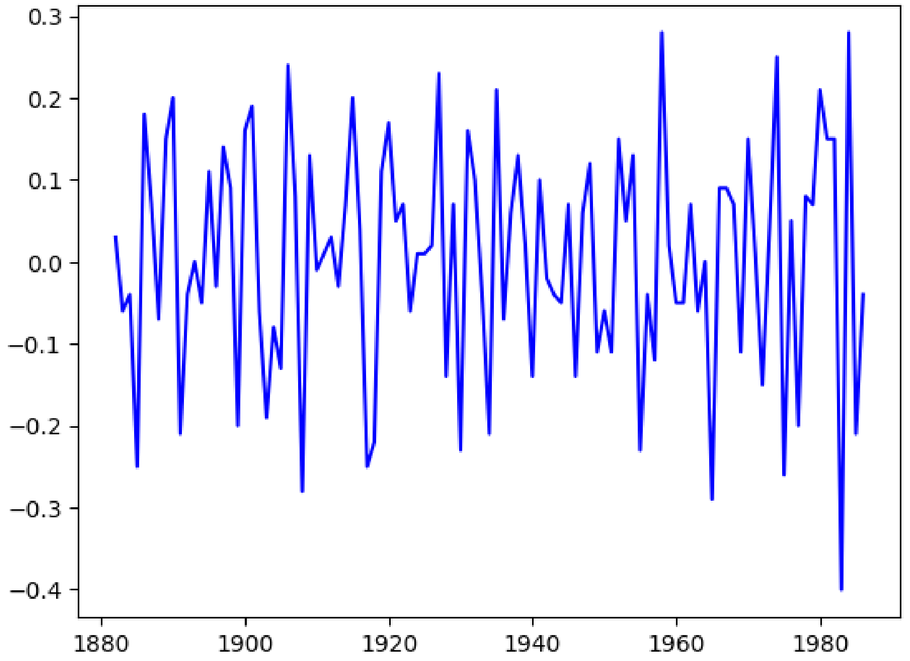
```
# Step 1: 시계열 데이터 불러오기

# 데이터 불러오기
tep = pd.read_csv('Global mean surface air temp changes 1880-1985.csv', header=None)

# 날짜 인덱스 설정
dates = pd.date_range('1880-12', periods=len(tep), freq='A-DEC')
tep.index = dates
tepts = pd.Series(tep[0], name='tep')

# 원본 데이터 시각화
plt.plot(tepts, color='b')
plt.show()
```



Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

```
# Step 2: 추세가 있으므로 1차 차분 후 시각화, 정상성 확인
# 1차 차분 수행
dtepts = tepts.diff(1).dropna()
dtepts.name = 'dtep'

# 차분 후 데이터 시각화
plt.plot(dtepts, color='b')
plt.show()
```



Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

```
# KPSS 테스트 수행 (정상성 확인)
sm.tsa.kpss(dtepts, regression='c', nlags='auto')
```

```
(0.08259101825264815,
 0.1,
 12,
 {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739})
```

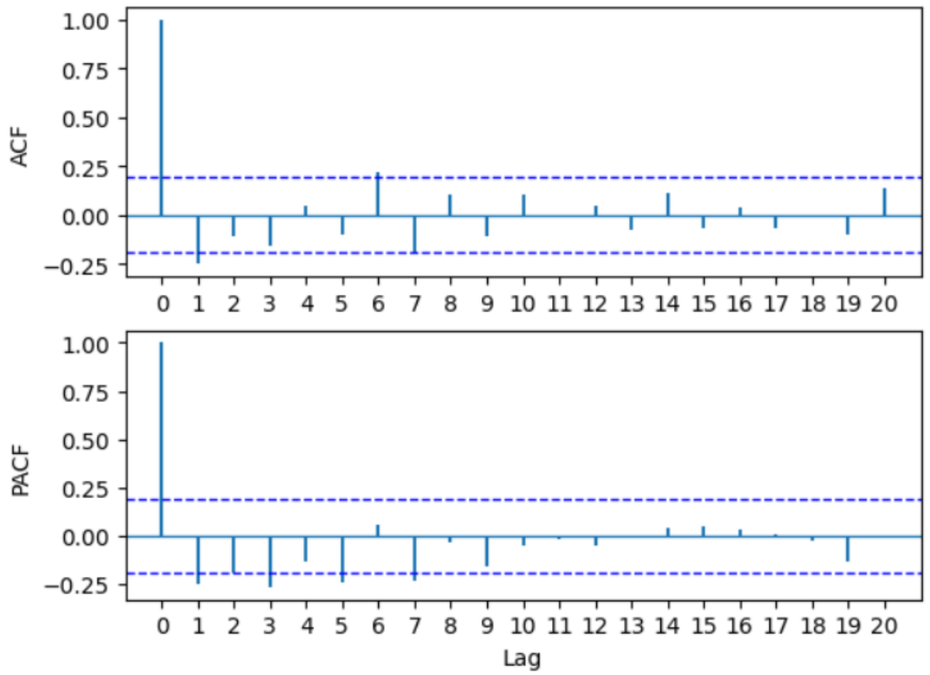
-KPSS 통계량 0.0826이 모든 임계값(10%, 5%, 2.5%, 1%)보다 작으므로 p-value>0.1

: 귀무가설을 기각하지 못하여 데이터가 stationary하다고 본다!

Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

```
# Step 3: ACF/PACF 분석
acf_pacf_fig(dtepts, both=True, lag=20)
plt.show()
```



-ACF: lag1 이후 절단? 지수적 감소?

-PACF: 지수적 감소? lag 몇 이후 절단?

무슨 모형인지 감이 안잡히므로

Automatic Information Criterion을 이용하자!

Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

```
# Step 4: Auto Order Determination
choose_arma2(dtepts, max_p=7, max_q=7, ctrl=1.03)
```

```
AIC:
          0       1       2       3       4       5       6       7
0       NaN -122.23 -129.96 -129.74 -128.09 -126.09 -129.74     NaN
1 -113.35 -130.71     NaN -135.56 -133.00 -131.23 -131.66 -129.61
2 -115.28 -129.25 -129.44 -126.29 -131.66 -129.50 -129.69 -127.34
3 -120.97 -128.65 -126.81 -128.47     NaN -128.00 -127.54 -125.34
4 -121.02 -126.65     NaN     NaN -126.33 -127.15 -126.29     NaN
5 -126.98 -125.61 -127.71 -127.71     NaN -126.08 -123.59 -121.18
6 -125.47 -124.49 -129.32 -127.76     NaN -125.30 -121.58     NaN
7 -131.25 -129.42 -129.24 -127.33 -124.07 -124.34 -121.61     NaN
AIC minimum is -135.56
(p, q)= (array([1]), array([3]))
BIC:
          0       1       2       3       4       5       6       7
0       NaN -114.27 -119.34 -116.47 -112.16 -107.51 -108.50     NaN
1 -105.39 -120.10     NaN -119.63 -114.42 -110.00 -107.77 -103.07
2 -104.67 -115.98 -113.52 -107.71 -110.43 -105.62 -103.15  -98.15
3 -107.70 -112.73 -108.23 -107.23     NaN -101.46  -98.34  -93.49
4 -105.10 -108.08     NaN     NaN  -99.79  -97.95  -94.44     NaN
5 -108.40 -104.38 -103.82 -101.17     NaN  -94.24  -89.09  -84.03
6 -104.24 -100.61 -102.78  -98.57     NaN  -90.80  -84.42     NaN
7 -107.37 -102.88 -100.05  -95.48  -89.57  -87.18  -81.80     NaN
BIC minimum is -120.1
(p, q)= (array([1]), array([1]))
HQIC:
          0       1       2       3       4       5       6       7
0       NaN -119.01 -125.65 -124.36 -121.64 -118.56 -121.13     NaN
1 -110.12 -126.41     NaN -129.11 -125.47 -122.62 -121.98 -118.85
2 -110.98 -123.87 -122.99 -118.76 -123.06 -119.82 -118.93 -115.51
3 -115.59 -122.20 -119.28 -119.86     NaN -117.24 -115.71 -112.44
4 -114.57 -119.13     NaN     NaN -115.57 -115.32 -113.38     NaN
5 -119.45 -117.00 -118.03 -116.96     NaN -113.18 -109.61 -106.13
6 -116.87 -114.81 -118.56 -115.93     NaN -111.32 -106.52     NaN
7 -121.57 -118.67 -117.41 -114.42 -110.09 -109.28 -105.48     NaN
HQIC minimum is -129.11
(p, q)= (array([1]), array([3]))
```

-AIC, HQIC는 ARIMA(1,1,3) 이 최적의 모형

-BIC는 ARIMA(1,1,1)이 최적의 모형

: parameter의 수에 대해 큰 penalty를

BIC가 부여함을 다시한번 확인해볼 수 있음!

Order Determination: Example 4.3

연간 평균 지표면 기온 변화(GMSATC) 시계열데이터의 ARMA 모델 구축

```
inf = sm.tsa.arma_order_select_ic(dtepts, max_ar=7, max_ma=7, ic=['aic', 'bic', 'hqic'], trend='c')

print("AIC 기준 최적 차수:", inf.aic_min_order)
print("BIC 기준 최적 차수:", inf.bic_min_order)
print("HQIC 기준 최적 차수:", inf.hqic_min_order)
```

```
AIC 기준 최적 차수: (1, 3)
BIC 기준 최적 차수: (1, 1)
HQIC 기준 최적 차수: (1, 3)
```

-choose_arma와 같은 결과가 도출됨

-이정도면 두 경우 다 해봐야하고, 교재 본문에서는 (1, 1)로 모형을 설정하고 진단 및 예측을 진행함.

04 Diagnosis of models

Diagnosis of models

앞서 우리는 시계열 모델의 차수를 결정하고, 적절한 추정 방법을 통해 모수(estimator)를 구하는 과정에 대해 살펴보았다. 하지만 모델을 설정하고 추정했다고 해서 바로 사용할 수 있는 것은 아니다. 모델이 주어진 데이터에 적절하게 적합(fitting)되었는지 검증하는 과정이 필수적이다.

이를 위해 크게 두 가지 검정을 수행해야 한다.

1.모수의 유의성 검정(Significance Test for Estimators)

1.추정된 모수가 통계적으로 유의미한지를 검정해야 한다.

2.일반적으로 t-검정이나 z-검정을 사용하여 각 모수가 0이라는 귀무가설을 기각할 수 있는지 확인한다.

2.잔차 분석(Residual Analysis)

1.모델이 데이터를 잘 설명하고 있는지 확인하기 위해 잔차가 백색소음(white noise) 특성을 가지는지 검토해야 한다.

2.즉, 잔차가 독립적이고 등분산성을 가지며, 자기상관이 존재하지 않아야 한다.

3.이를 위해 Ljung-Box 검정, ACF(자기상관함수) 플롯, 정규성 검정 등을 활용할 수 있다.

Significance Test for Estimators

Example 4.1) 앞서 우리는 example 4.1의 시계열 데이터가 AR(1) model을 따른다는 것을 알아봤다.

이제 그에 대한 가설 검정을 해볼 차례!

```
#ARMA 모델 추정 및 결과 출력
ar1=ARMA(naots, order=(1,0)).fit(trend='c', disp=False)
print(ar1.summary())
```

```
const         0.0040     0.043     0.092     0.927    -0.081     0.089
ar.L1.index   0.1996     0.034     5.867     0.000     0.133     0.266
```

Const(상수항)의 p-value는 0.927로 매우 크므로, 유의하지 않음을 알 수 있음.

```
#모수 유의성 해석
ar1=ARMA(naots, order=(1,0)).fit(trend='nc', disp=False)
print(ar1.summary())
```

trend='nc'옵션을 통해 상수항을 제거한 모델을 적합한다.

새로운 모델에서 다시 p-value값을 확인하여 모수의 유의성을 평가할 수 있음.

Residual analysis

Estimated model이 유의미하기 위해서는 residual이 white noise의 특성을 띄어야 한다.

➔잔차에 유의미한 상관이 남아 있다면, 모델이 설명하지 못한 패턴이 존재한다는 뜻이므로!

모델을 수정하거나 추가적인 변수를 고려해야 함.

잔차가 백색소음의 특성을 가지는지 확인하는 대표적인 방법 4가지는 다음과 같다.

1. ACF plotting
2. Ljung-Box 검정(Portmanteau Test)
3. QQ plotting or Normality Test
4. ACF Plotting of Squared Residuals

Residual analysis

1. **ACF Plotting: R**esidual의 ACF가 White noise와 유사한 형태여야 함

잔차의 자기상관을 시각적으로 확인하는 방법

만약 ACF의 유의미한 lag값에서 높은 자기상관이 나타나면, 모델이 해당 정보를 제대로 반영하지 못했다는 의미

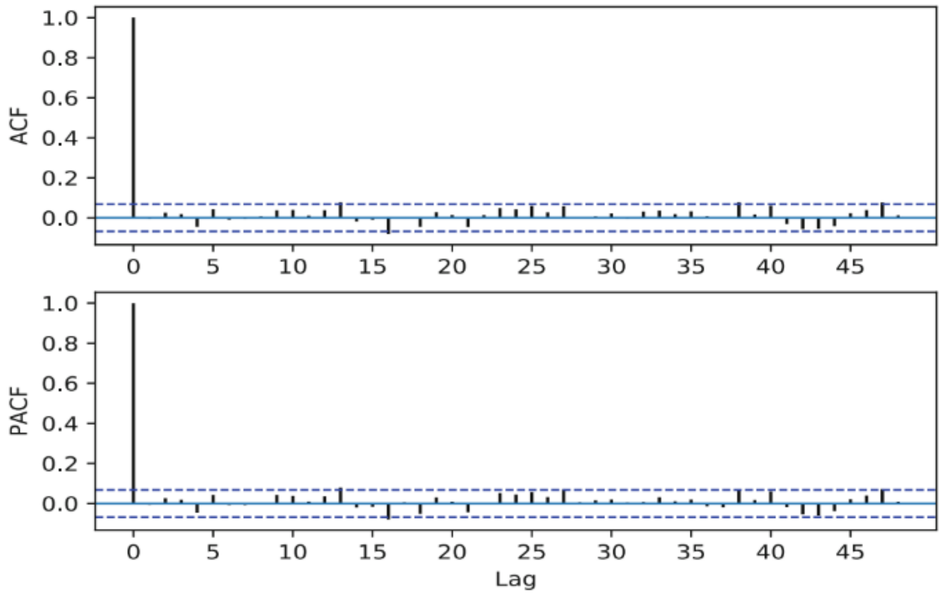


Fig. 4.3 ACF and PACF plots of the NAO index AR(1) model residuals

-AR(1) model의 ACF와 PACF 모두 모든 시차에서
0에 가까운 값을 나타내어 white noise형태임을 앎.

Residual analysis

2. Ljung-Box 검정(Portmanteau Test): ACF플롯이 시각적인 도구라면, Ljung-Box 검정은 이를 정량적으로 평가하는 방법

H_0: 잔차에 자기상관이 없다

$$Q_{LB}(m) = n(n+2)\sum_{k=1}^{m}\frac{r_k^2}{n-k} \sim \chi^2(m) \text{ under } H_0$$

n = 표본개수

r_k = 시차 k에서 sample ACF

m = 검정할 최대 시차

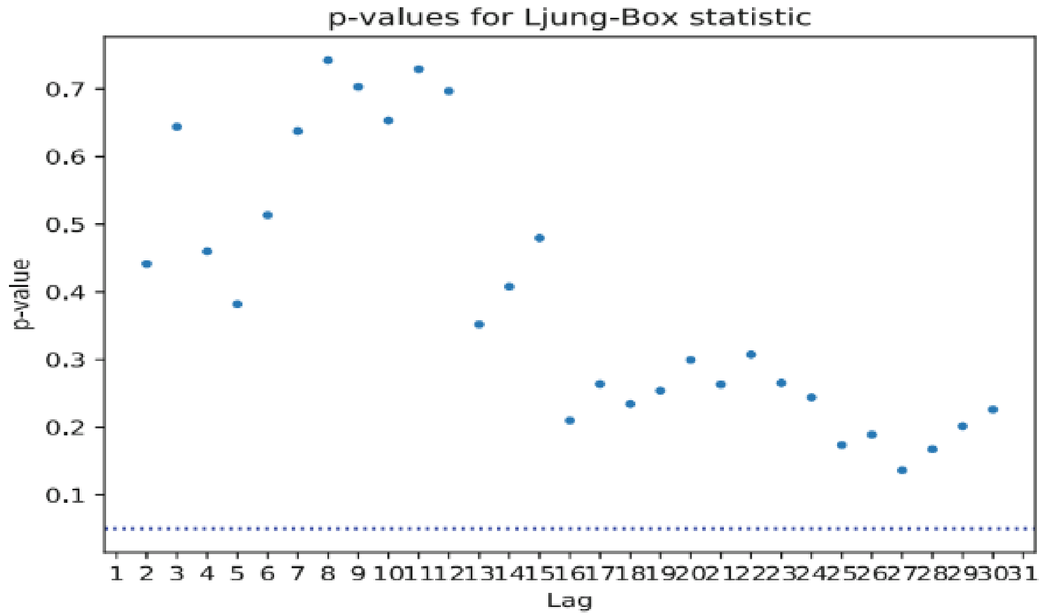


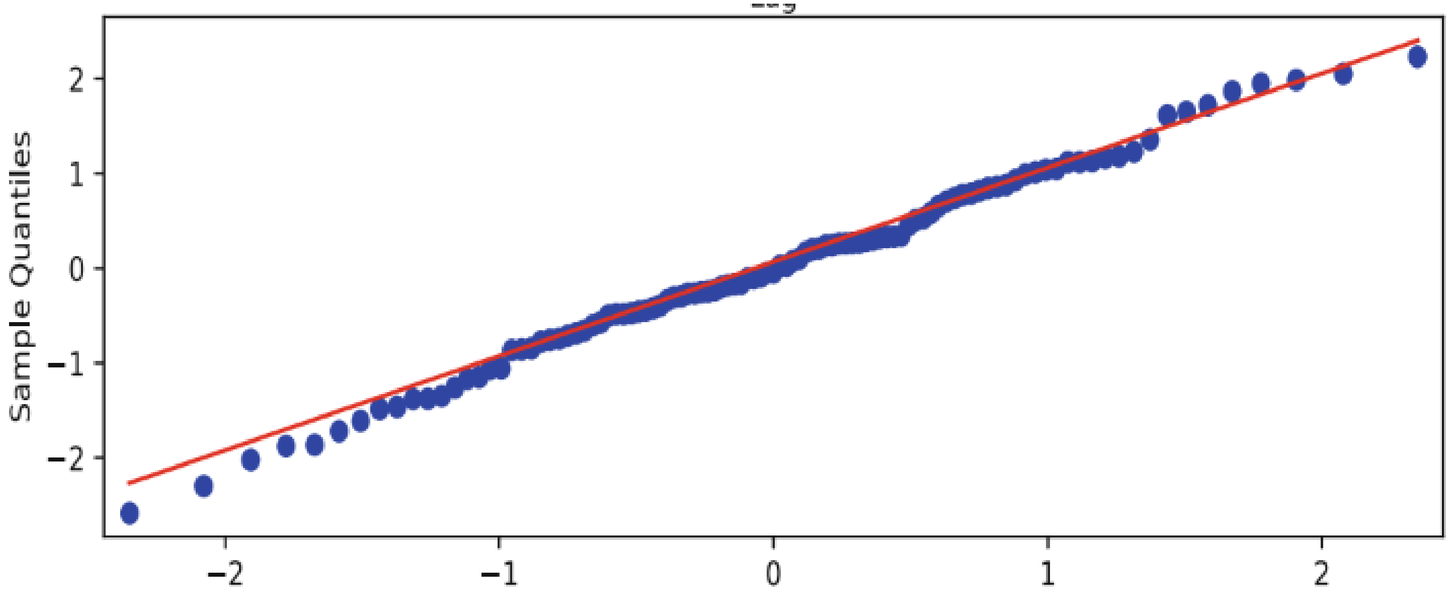
Fig. 4.4 *p*-values for Ljung-Box statistic of the NAO index AR(1) model

-다양한 시차 범위에서 p-value가 크게 유지되므로, 어느 시차에서도 통계적으로 유의미한 잔차 자기 상관이 나타나지 않는다.

Residual analysis

3. QQ Plotting or normality testing: 잔차가 정규분포를 따르는지 확인하는 과정

QQ플롯을 통해 잔차가 정규성 가정을 만족하는지 시각적으로 평가할 수 있음.



-파란 점들이 붉은 기준선에 가깝게 놓여져 있음.

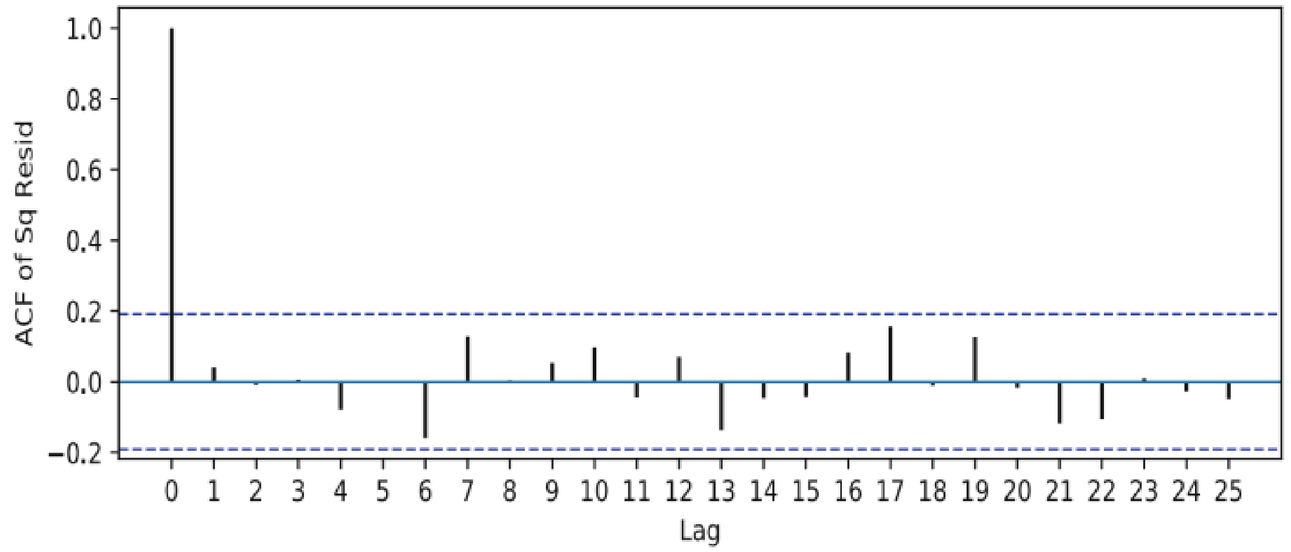
잔차가 정규성에 크게 벗어나지 않는다고 판단할 수 있다.

Residual analysis

4. ACF plotting of the squared residuals: 잔차의 분산이 일정한지(등분산성)를 검토하기 위한 방법

만약 제곱된 잔차에서 유의미한 자기상관이 존재한다면, 이는 모델이 시간에 따른 분산의 변화를 설명하지 못함을 의미함.(즉, ARCH 효과 가능성)

이 경우, ARCH/GARCH 모델과 같은 조건부 이분산 모델을 고려해야 함.(자세한 내용은 이후에 다룸)



-파란 점선은 대략 95% 신뢰구간을 나타냄. 대부분의 ACF 막대가 신뢰구간 내에 존재하므로, 제곱 잔차들 간에 유의미한 자기상관이 없다고 판단할 수 있음

Residual analysis

잔차 분석 과정을 보다 손쉽게 수행하기 위해 pythontsa패키지에서 잔차 분석을 위한 함수를 제공한다.

1. plot_LB_pvalue(x, noestimatedcoef, nolags)

    x: 분석할 잔차 데이터

    noestimatedcoef: 모델에서 추정된 모수의 개수(ARMA(p,q)모델이면 p+q)

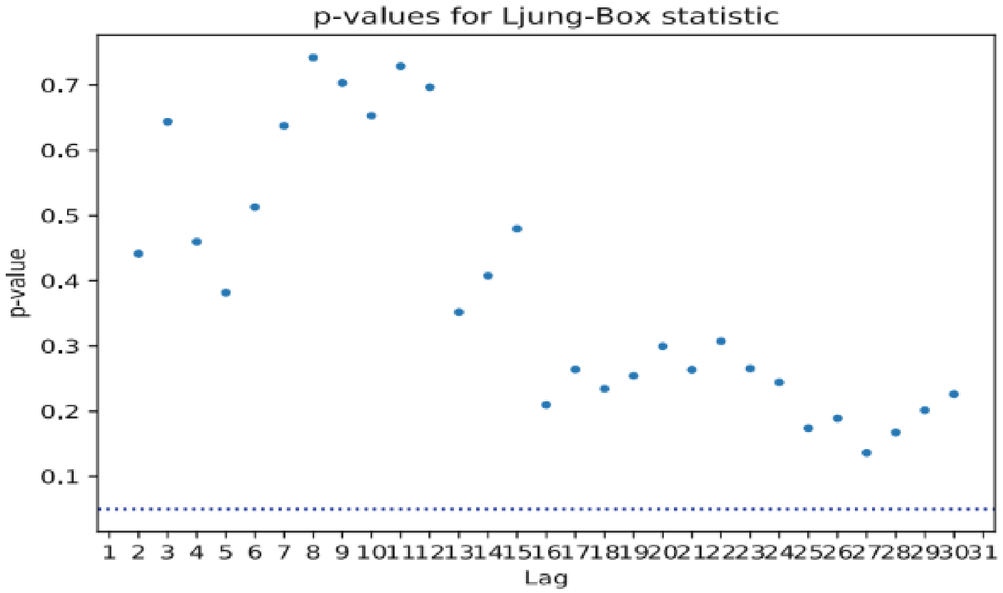
    nolags: Ljung-Box 검정 시 고려하는 최대 시차(Lag) 수

    이 함수는 Ljung-Box 통계량의 p-value를 다양한 시차에 대해 시각화해 준다

Residual analysis

Example 4.1

```
#AR(1)모델 적합
ar1 = ARMA(naots, order=(1,0)).fit(trend='nc', disp=False)
#잔차 추출
resid1 = ar1.resid
#plot_LB_pvalue() 함수 호출
acf_pacf_fig(resid1, both=True, lag=48)
plt.show()
plot_LB_pvalue(resid1, noestimatedcoef=1, nolags=30)
plt.show() #여러 시차에 대한 Ljung-Box검정 결과의 p-value를 그려준다
```



-시차에 따른 잔차의 자기상관을 시각적으로 간단하게 확인할 수 있다!

Residual analysis

2. plot_ResidDiag(x, noesimatedcoef, nolags, lag)

   Ljung-Box test p-value, QQ plot, 잔차 ACF, 제곱 잔차 ACF 등을 한 번에 확인할 수 있는 함수

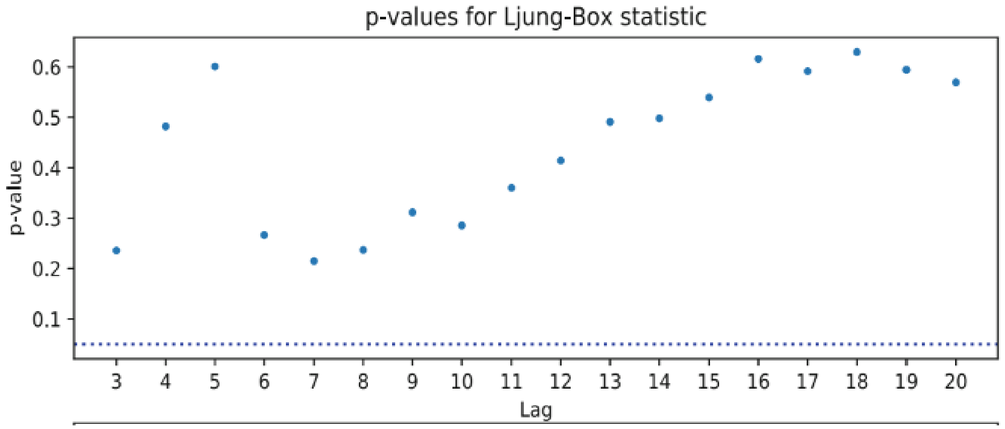
   lag인자는 ACF계산 시 고려할 최대 시차 수를 지정함.

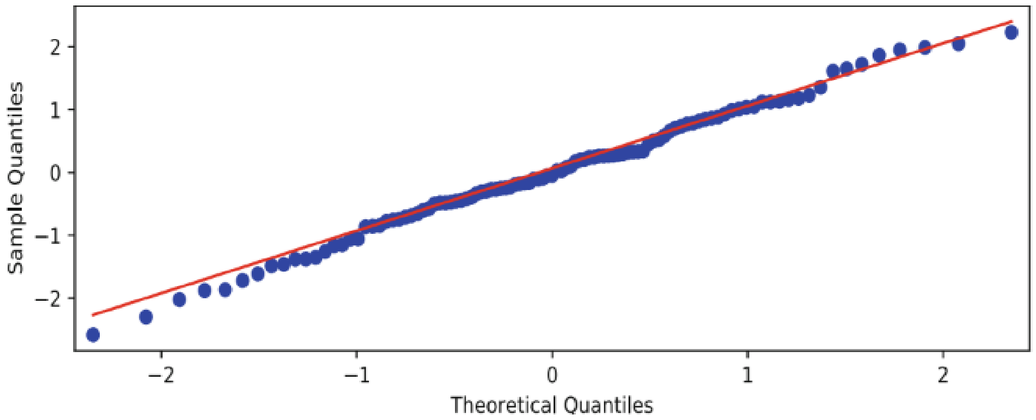
Example 4.3

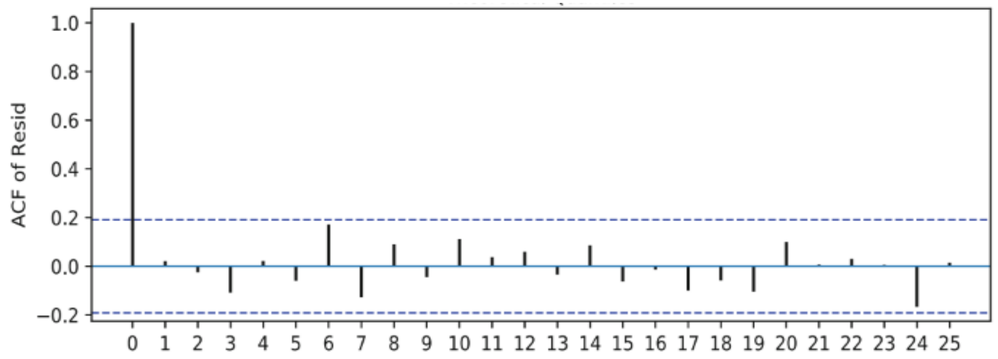
```
# ARMA(1,1) 모델 적합
arma11 = ARMA(detpts, order=(1,1)).fit(trend='c', disp=False)
# 잔차 저장
resid11 = arma11.resid
# 잔차 진단을 위한 plot_ResidDiag() 실행
plot_ResidDiag(resid11, noestimatedcoef=2, nolags=20, lag=25)
plt.show()
```

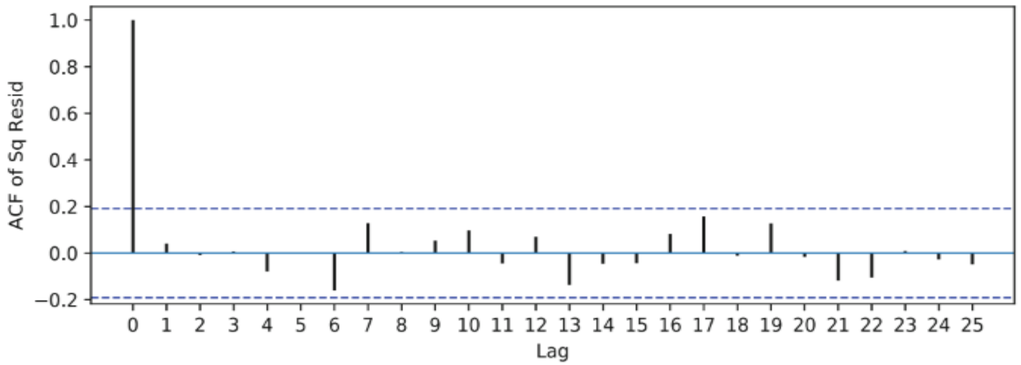
Residual analysis

다음과 같이 잔차에 대한 여러 검정 결과를 한눈에 시각화 할 수 있다!









05 Forecasting

Forecasting

시계열 분석의 주요 목표 중 하나는 미래 값을 예측(forecasting)하는 것이다

이를 위해 모형 식별, 모수 추정, 모형 지단의 단계를 거친 후, 적절한 예측 기법을 활용하여 미래의 값을 추정하게 된다.

예측은 크게 두 가지 유형으로 나뉜다.

1. **in-sample prediction:** 주어진 데이터 t_0시점까지의 값을 이용하여 예측값을 산출한 후, 실제 관측값과 비교하여 모델의 성능을 평가하는 과정.

   예측 시점 t_0에 실제 관측값이 존재하므로, 예측값이 fitted values 혹은 in-sample predictors라고 불린다.

   ->모델이 과거 데이터를 얼마나 잘 설명하는지(적합도)를 평가하는데 사용되며, 직접적인 미래 예측이 아닌 모델 성능 검증의

   용도로 사용됨.(회귀 분석에서 R^2값을 계산하는 과정과 유사!).

Forecasting

2. Out-of-sample forecasting(일반적인 예측): 현재 시점 n까지의 데이터를 이용하여, 미래 시점 n+h(h-step ahead forecast)에 대한 값을 예측하는 경우. 즉, 현재까지 알려진 데이터 X_n을 활용하여 미래 값 X_{n+h}를 예측하는 과정.

n을 예측의 기준점(forecast origin)이라고 하며, 예측 시점 n+h로 진행될수록 불확실성이 증가한다.

In-sample Prediction vs. Out-of-sample Forecasting 비교

예측 유형	설명	목적	예측 대상
In-sample Prediction	기존 데이터(과거 시점)에서 예측값을 계산하여 실제값과 비교	모델 적합도 평가	과거 데이터(학습 데이터)
Out-of-sample Forecasting	현재까지의 데이터를 기반으로 미래 값을 예측	실제 예측 수행	미래 데이터

Out-of-sample forecasting

We denote the forecast of $X_{n+h} = \hat{X}_n(h)$

예측오차: $e_n(h) = X_{n+h} - \hat{X}_n(h)$, 실제 값과 예측값의 차이로 정의

예측 오차를 최소화하기 위해 **MSE**가 가장 많이 사용된다.

$$\mathsf{E}[e_n(h)]^2 = \mathsf{E}[X_{n+h} - \hat{X}_n(h)]^2 = \min_g \left\{ \mathsf{E}[X_{n+h} - g(X_{1;n})]^2 \right\}$$

where $g(X_{1;n})$ is any measurable function of the observations $\{X_{1:n}\}$.

가능한 모든 예측함수 g($X_{1:n}$) 중에서 MSE를 최소화하는 함수를 선택할 때, $\hat{X}_n(h)$ 를 Best Linear Predictor라고 부른다.

Out-of-sample forecasting

$$\mathsf{E}[e_n(h)]^2 = \mathsf{E}[X_{n+h} - \hat{X}_n(h)]^2 = \min_g \left\{ \mathsf{E}[X_{n+h} - g(X_{1;n})]^2 \right\}$$

가능한 모든 예측함수 g(*)중에서 MSE를 최소화하는 함수를 사용 $\hat{X}_n(h) = \mathbb{E}[X_{n+h} \mid X_{1:n}]$ 이 이 된다!

Proof)

$$\mathbb{E}[(X_{n+h} - g(X_{1:n}))^2]$$

$$= \mathbb{E}[(X_{n+h} - \mathbb{E}[X_{n+h} \mid X_{1:n}] + \mathbb{E}[X_{n+h} \mid X_{1:n}] - g(X_{1:n}))^2]$$

$$\mathbb{E}[(X_{n+h} - g(X_{1:n}))^2] = \mathbb{E}[(X_{n+h} - \mathbb{E}[X_{n+h} \mid X_{1:n}])^2] + \mathbb{E}[(\mathbb{E}[X_{n+h} \mid X_{1:n}] - g(X_{1:n}))^2].$$

$$\mathbb{E}[X_{n+h} - \mathbb{E}[X_{n+h} \mid X_{1:n}] \mid X_{1:n}] = 0.$$

One-step ahead predictor

ARMA(p,q)에서는 innovation 알고리즘을 통해 과거 관측값과 오차를 재귀적으로 이용하여 예측치를 구한다.

$$\hat{X}_{n+1} = \begin{cases} 0, & n = 0 \\ \sum_{j=1}^{n} \theta_{nj}(X_{n+1-j} - \hat{X}_{n+1-j}), & 1 \le n < \max(p,q) \\ \sum_{i=1}^{p} \varphi_i X_{n+1-i} + \sum_{j=1}^{q} \theta_{nj}(X_{n+1-j} - \hat{X}_{n+1-j}), & n \ge \max(p,q) \end{cases}$$

여기서 θ_{nj}값들은 innovation 알고리즘을 통해 구함

X_{n+1-h}값들은 과거 관측값을 사용

One-step ahead predictor

식이 왜 저렇게 나왔나?

기본 아이디어: 과거 시점의 관측값 X_{n+1-h}, 과거 예측 오차 X_{n+1-j} - $\hat{X}_{n+1-j}$ 이 두가지 정보를 조합하여 예측을 수행한다.

ARMA(p, q) 모델은 다음과 같이 정의됨:

$$X_n = \sum_{i=1}^{p} \phi_i X_{n-i} + \sum_{j=1}^{q} \theta_j \epsilon_{n-j} + \epsilon_n$$

$\hat{X}_{n+1}$를 구할 때, 우리는 현재까지 관측된 값 $X_{1:n}$을 기반으로 예측해야 함.

미래의 백색소음 ϵ_n은 알 수 없으므로 $E(\epsilon_n | X_{1:n})=0$을 사용함.

One-step ahead predictor

Case 1: n=0(초기 상태)

$\hat{X}_{n+1}$ = 0 초기 상태 에서는 과거 데이터가 없으므로 예측값은 0으로 설정됨.

Case2: 1 ≤ n ≤ max(p,q)

$$\hat{X}_{n+1} = \sum_{j=1}^{p} \theta_{n,j}(X_{n+1-j} - \hat{X}_{n+1-j})$$

현재 시점까지의 데이터를 사용하여 예측 오차와 가중치를 적용하여 예측 수행

Case3: n ≥ max(p,q)

$$\hat{X}_{n+1} = \sum_{i=1}^{p} \phi_i X_{n+1-i} + \sum_{j=1}^{q} \theta_{n,j}(X_{n+1-j} - \hat{X}_{n+1-j})$$

AR항: 과거 관측값 X_{n+1-i}에 계수 φ_i를 곱하여 선형 결합

MA항: 과거 예측 오차(X_{n+1-j} - $\hat{X}_{n+1-j}$)를 사용하여 보정

과거 값의 선형 조합 + 이전 예측 오차(innovation)보정의 형태를 가짐

H-step ahead predictor

H-step ahead predictor is

$$\hat{X}_n(h) = \sum_{i=1}^{p} \varphi_i \hat{X}_n(h-i) + \sum_{j=h}^{q} \theta_{n+h-1,j}(X_{n+h-j} - \hat{X}_{n+h-j}),$$

1-step ahead predictor를 반복하여 여러 스텝 앞까지 예측을 수행

예측 시점이 멀어질수록(h가 커질수록) 예측 오차가 누적되어 신뢰도가 낮아질 수 있음.

따라서 장기 예측에서는 모델의 불확실성을 고려해 예측구간도 함께 계산하는 것이 필요.

$\hat{X}_n(h-i)$는 과거 예측값이고

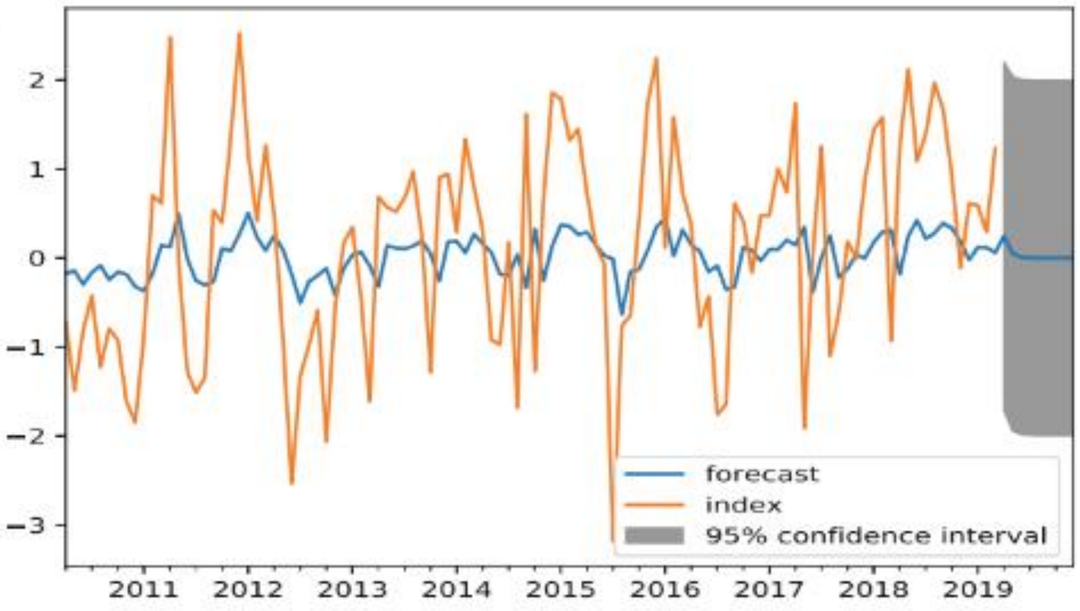
$(X_{n+h-j} - \hat{X}_{n+h-j})$는 한 스텝씩 예측 오차를 누적하는 구조이다.

식은 매우 어렵지만, 라이브러리 내부적으로 공식이 구현되어 있기에 우리는 model.predict(start,end)를 호출하기만 하면 위 식을 재귀적으로 계산 가능!

Forecasting-Example 4.1

Example 4.1:ARMA(1,0) 모델을 활용한 시계열 예측

```
#데이터 로드
nao = pd.read_csv('nao.csv', header=0)
nao.index = pd.date_range(start='1950-01', periods=len(nao), freq='M')
#ARMA(1,0) 모델 적합(Fitting)
model = ARMA(nao['index'], order=(1,0)).fit(trend='nc', disp=False)
#예측(Forecasting) 수행
pred = model.predict(start='2010-04-30', end='2019-12')
#예측값과 실제값 비교
pred_frame = pd.concat([nao['2010-04-30':], pred], axis=1)
pred_frame.plot()
plt.show()
```



ARMA(1,0)모델을 사용하여 2010~2019년 데이터를 예측

그래프를 보면, 예측값이 실제 데이터와 그다지 유사하지 않다는 것을 알 수 있음.

회색 신뢰구간은 예측의 불확실성을 나타낸다.

Forecasting-Example 4.2

Example 4.2: 예시에서는 ARMA(2,2) 모델을 가정하여 시뮬레이션된 시계열 데이터(길이 500)를 생성한 뒤,

실제로는 모델 차수를 모른다는 가정 하에 choose_arma() 등의 함수를 사용하여 (p,q) 를 추정해본다.

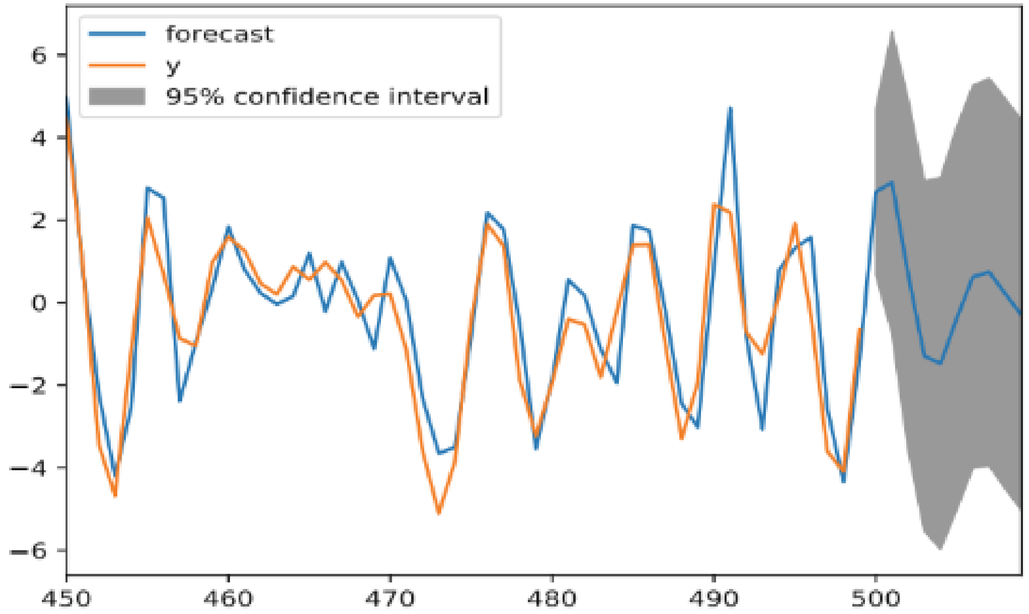
최종적으로 (p,q) = (2,2) 로 모델을 적합한 뒤, 잔차 검정 및 예측 을 수행한다.

```
#(1) 모델 적합
arma22 = ARMA(y, order=(2,2)).fit(trend='nc')
print(arma22.summary())
#(2)잔차 분석(백색소음, 정규성 등)
resid22 = arma22.resid
acf_pacf_fig(resid22, both=True, lag=20)
plt.show()
plot_LB_pvalue(resid22, noestimatedcoef=4, nolags=26)
plt.show()
stats.normaltest(resid22)
#(3) 예측(out-of-sample + in-sample)
arma22.plot_predict(start=450, end=509)
plt.show()
#(4) 예측 결과를 기존 데이터와 병합 후 시각화
predframe = pd.concat([y[450:], pred_mean, pred_conf.iloc[-10:]], axis=1)
predframe.plot()
plt.show()
```

arma22.plot_predict(start=450, end=509):

시계열 데이터가 0부터 499까지 있으므로,
450~499 구간은 in-sample 예측, 500~509
구간은 out-of sample 예측이 됨.

Forecasting-Example 4.2



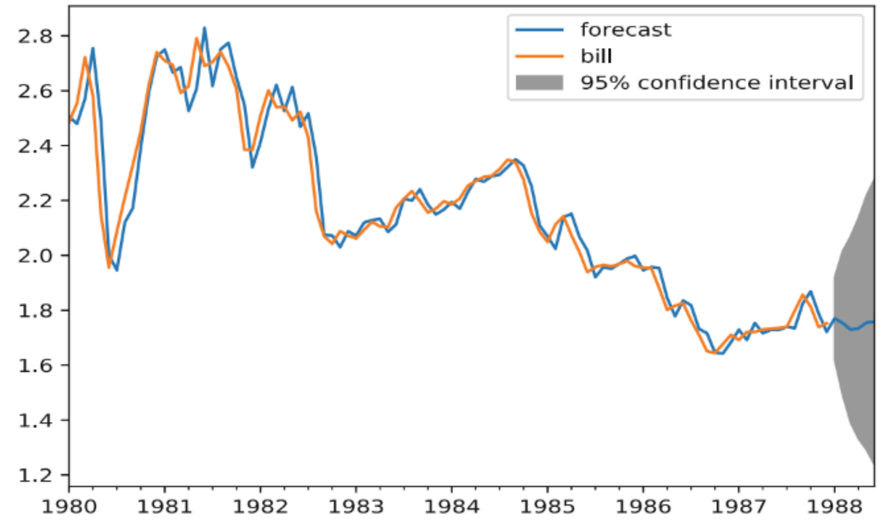
회색 영역(95% CI)가 예측 구간이 늘어날수록 예측 불확실성이 증가하면서 넓어지는 것을 볼 수 있다.

모델이 시뮬레이션으로 만들어진 원래 ARMA(2,2)과정을 잘 복원했음을 알 수 있다!

Forecasting-Example 4.4

Example 4.4: ARIMA(6,1,0) 모델을 활용한 시계열 예측

```
# ARIMA(6,1,0) 모델 적합
arima610 = ARIMA(ly, order=(6,1,0)).fit(trend='nc')
print(arima610.summary())
# 미래 값 예측
fo = arima610.predict(start='1980-01', end='1988-06', typ='levels')
print(fo)
fo.plot(); plt.show()
# 예측값과 신뢰구간 계산
pred = arima610.get_prediction(start='1980-01', end='1988-06')
predicts = pred.predicted_mean
predconf = pred.conf_int()
# 실제값과 예측값 비교 데이터 생성
predframe = pd.concat([ly['1980-01':], predicts, predconf['1988-06':]], axis=1)
# 최종 그래프 출력
predframe.plot(); plt.show()
```



ARIMA(6,1,0)은 비정상 시계열을 1차 차분하여 정상화 후 모델링한다.

typ='levels'옵션을 사용하여 차분된 값을 예측한 뒤, 자동으로 누적화 하여 원래 시계열 스케일로 복원한다.

06

Conclusion + Summary

Conclusion / Summary

1 . Modeling building problems & estimation methods:

ARMA model의 parameters를 구하는 다양한 방법들을 알아봄

2 . Order determination:

model의 차수를 어떻게 결정할 것인가?

3 . Diagnosis of models

모델이 주어진 데이터에 적절하게 적합(fitting)되었는지 검증하는 과정

4 . Forecasting

미래의 값을 예측하는 방법에 대해 소개

THANK YOU