Stationarity

Strict stationarity

A stochastic process $\{Y_t\}_{t=1}^{\infty}$ is **strictly stationary** if, for any set of subscripts $t_1, t_2, \ldots, t_r$ with any given finite integer r, the **joint distribution** of

$$(Y_{t_1}, \ldots, Y_{t_r})$$

is the same as **the joint distribution** of

$$(Y_{t_1+k}, \ldots, Y_{t_r+k})$$

for any time shift k.

- For instance, the joint distribution of (Y_1, Y_5, Y_8) is the same as the distribution of (Y_{11}, Y_{15}, Y_{18}).
- There is no guarantee that $E[Y_t]$ and/or $E[|Y_t|^2]$ exist.
  - Consider $Z \sim N(0,1)$, $X \sim \chi^2(k)$, and X is independent of Z. Then,

$$Y = \frac{Z}{\sqrt{X/k}} \sim t(k)$$

  - The expected value of Y is well-defined only for $k > 1$, which is 0. If $k = 1$, the mean of Y is undefined.
  - The variance of Y is well-defined only for $k > 2$, which is $k/(k-2)$.

Covariance stationarity

A stochastic process $\{Y_t\}_{t=1}^{\infty}$ is **covariance stationary** if,

- $E[|Y_t|^2] < \infty$ for all t (square-integrable)
- $E[Y_t] = \mu$ and $Var(Y_t) = \sigma^2$ for all t (time-invariant mean and variance)
- $Cov(Y_t, Y_{t-j}) = \gamma_j$ depends on j but not on t (time-invariant covariance)

Accordingly, the j-th autocorrelation is

$$corr(Y_t, Y_{t-j}) = \frac{Cov(Y_t, Y_{t-j})}{\sqrt{Var(Y_t)Var(Y_{t-j})}} = \frac{\gamma_j}{\sigma^2} = \frac{\gamma_j}{\gamma_0}.$$

The **sample autocorrelation function (ACF)** $\hat{\rho}_j$ as a function of j where

$$\hat{\rho}_j = \frac{\hat{\gamma}_j}{\hat{\gamma}_0} \text{ for } |j| < T, \text{ where } \hat{\gamma}_j = \frac{1}{T-j}\sum_{t=1}^{T-j}(Y_t - \bar{Y})(Y_{t+j} - \bar{Y}), \text{ and } \bar{Y} = \frac{1}{T}\sum_{t=1}^{T} Y_t.$$

Note that

- If $E[|Y_t|^2] < \infty$, strict stationarity implies covariance stationarity.
  - If not, then strict stationarity does not imply covariance stationarity.
  - For instance, $\{Y_t\}_{t=1}^{\infty} \overset{IID}{\sim} t(1)$ is strictly stationary by the IID assumption
    - By independence, the joint distribution of $(Y_{t_1}, \ldots, Y_{t_r})$ is the product of marginal distributions of Y_t for $t = t_1, \ldots, t_r$.
    - Similarly, the joint distribution of $(Y_{t_1+k}, \ldots, Y_{t_r+k})$ is the product of marginal distributions of Y_t for $t = t_1 + k, \ldots, t_r + k$.
    - By the assumption of identical distribution, the two joint distributions are identical.
  - However, the mean and variance of Y_t do not exist.
- If $E[Y_t] = \mu$ is time invariant, we can use $\sum_{t=1}^{T} Y_t / T$ to estimate $E[Y_t]$. That's a case of why stationarity is important.

MDS processes

A time series $\{d_t\}_{t=1}^{\infty}$ is a **martingale difference sequence** if $E[d_t|\mathcal{F}_{t-1}] = 0$ for all t.

- An MDS can be non-stationary. For instance, $d_t \sim mds(0, \sigma^2(t))$, where $\sigma^2(t)$ varies across t.
- An MDS with constant variance is covariance stationary.
  - By definition, $E[d_t|\mathcal{F}_{t-1}] = 0$ and $Var(d_t) = \sigma^2$ for all t.
  - By law of iterated expectations, $E[d_t] = E[E[d_t|\mathcal{F}_{t-1}]] = E[0] = 0$ for all t
  - $E[d_t d_{t-j}] = 0$ and hence $Cov(d_t, d_{t-j}) = 0$
    - Let $d_t = Y_t - Y_{t-1}$, where $E[Y_t|\mathcal{F}_{t-1}] = Y_{t-1}$, i.e, $\{Y_t\}$ is a martingale.
    - $E[d_t d_{t-j}|\mathcal{F}_{t-1}] = E[E[d_t|\mathcal{F}_{t-1}]d_{t-j}] = E[0 \cdot d_{t-j}] = 0$.

White noise processes

$\{Y_t\}_{t=1}^{\infty}$ is a white noise process if it has mean zero and no autocorrelations.

- The variance $Var(Y_t)$ may be time-varying

$Y_t \sim WN(0, \sigma^2)$ is an uncorrelated process with mean zero and **constant variance** σ^2.

Moreover,

- $Y_t \sim IID(0, \sigma^2) \Rightarrow Y_t \sim MDS(0, \sigma^2)$.
  - $E[Y_t|\mathcal{F}_{t-1}] = E[Y_t] = 0$ by independence
- $Y_t \sim MDS(0, \sigma^2) \Rightarrow Y_t \sim WN(0, \sigma^2)$

    - $E[Y_tY_{t-j}] = E[E[Y_tY_{t-j}|\mathcal{F}_{t-1}]] = E[E[Y_t|\mathcal{F}_{t-1}]Y_{t-j}] = E[0 \cdot Y_{t-j}] = 0$ and hence $Cov(Y_t, Y_{t-j}) = 0$.

The reverse are not true.

- $Y_t \sim WN(0, \sigma^2) \nRightarrow Y_t \sim MDS(0, \sigma^2)$
    - Example: Suppose that $\epsilon_t \sim IID(0, \sigma^2)$. $Y_t = \epsilon_t + \epsilon_{t-1}\epsilon_{t-2}$ is a $WN(0, \sigma^2 + \sigma^4)$ process but not an $MDS(0, \sigma^2 + \sigma^4)$ process.
- $Y_t \sim MDS(0, \sigma^2) \nRightarrow Y_t \sim IID(0, \sigma^2)$.
    - Example: Consider an ARCH(1) process

$$\begin{aligned} e_t &= \sigma_t \epsilon_t \\ \sigma_t^2 &= \omega + \alpha e_{t-1}^2, \text{ where } \omega \geq 0, 0 < \alpha < 1 \\ \epsilon_t &\sim WN(0,1) \\ \epsilon_t & \text{ independent of } \sigma_t \end{aligned}$$

Note that $E[e_t|e_{t-1}, e_{t-2}, \ldots] = \sigma_t E[\epsilon_t] = \sqrt{\omega + \alpha e_{t-1}^2} E[\epsilon_t] = 0$, and hence $\{e_t\}_{t=1}^{\infty}$ is an MDS. Besides, $Var(e_t) = \omega/(1-\alpha)$. But $Var(e_t|e_{t-1}) = \omega + \alpha e_{t-1}^2$. So $\{e_t\}_{t=1}^{\infty}$ is not IID.

Autoregressive moving-average processes

The ARMA(1,1) process:

$$Y_t - \mu = \phi(Y_{t-1} - \mu) + \epsilon_t + \theta\epsilon_{t-1},$$

where $|\phi| < 1$ and $\epsilon_t \sim WN(0, \sigma_\epsilon^2)$.

- $|\phi| < 1$ ensures ARMA(1,1) is covariance stationary

We have

- $E[Y_t] = \mu$
- $Var(Y_t) = \frac{1+\theta^2+2\phi\theta}{1-\phi^2}\sigma_\epsilon^2$
- $\gamma_j = Cov(Y_t, Y_{t-j}) = \begin{cases} \frac{(1+\phi\theta)(\phi+\theta)\sigma_\epsilon^2}{1-\phi^2}, & |j| = 1 \\ \phi\gamma_{j-1}, & j \geq 2 \\ \gamma_{-j}, & j \leq -2 \end{cases}$
- $\rho_j = corr(Y_t, Y_{t-j}) = \begin{cases} \frac{(1+\phi\theta)(\phi+\theta)}{1+\theta^2+\phi\theta}, & |j| = 1 \\ \phi\rho_{j-1}, & j \geq 2 \\ \rho_{-j}, & j \leq -2 \end{cases}$

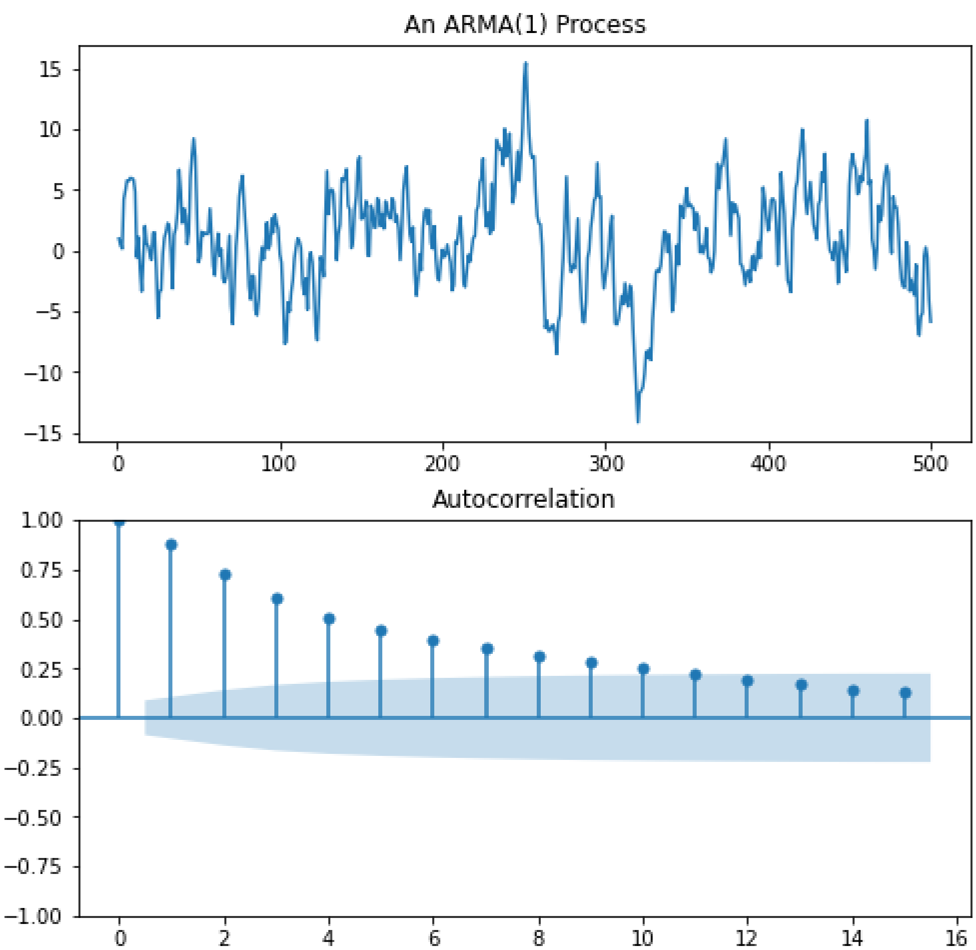
Consider $Y_t - 1 = 0.8(Y_{t-1} - 1) + \epsilon_t + 0.2\epsilon_{t-1}$, where $\epsilon_t \sim WN(0, 4)$.

In [1]:

```
import numpy as np
from statsmodels.tsa.arima_process import arma_generate_sample

np.random.seed(2022)
phi = -0.8 # note the minus sign despite the true phi = 8
theta = 0.2
ar = np.r_[1, phi] # add zero-lag
ma = np.r_[1, theta] # add zero-lag
y = 1 + arma_generate_sample(ar, ma, scale = 2, nsample = 500)
```

In [2]:

```
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf


fig, ax = plt.subplots(nrows = 2, ncols = 1,figsize=(8,8))
ax[0].plot(np.arange(1, 501, 1), y) # first subgraph
ax[0].title.set_text('An ARMA(1) Process') # add the title

plot_acf(y, lags = 15, ax = ax[1]) # second subgraph
plt.show()
```



Ljung-Box test

The Ljung-Box test is used for testing H_0: the correlations in the population from which the sample is taken are 0. The test statistic is given by

$$Q = T(T+2)\sum_{j=1}^{h}\frac{\hat{\rho}_j^2}{T-j} \sim \chi^2(h) \text{ asymptotically},$$

where T is the sample size, $\hat{\rho}_j$ is the sample autocorrelation at lag j, and h is the number of lags being tested. We reject H_0 if the p-value is less than significance level α.

In [3]:

```python
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.stats.diagnostic import acorr_ljungbox

res = ARIMA(y, order=(1, 0, 1)).fit() # fit the ARMA(1, 1) model
print(res.summary())

print('=============== if you want to extract the values ===============')
print('The estimated coefficients are ', res.params) # estimated coefficients
print('The standard errors for the coefficients are ', res.bse) # standard error

Q = acorr_ljungbox(res.resid, lags=[10], return_df=True) # test if there are no
print('=================================================================')
print('The Ljung-Box test statistic and p-value are \n', Q)

print('=============== if you want to extract the values ===============')
print('The test statistic is %.6f' % Q.lb_stat.values[0])
print('The p-value is %.6f' % Q.lb_pvalue.values[0])
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  500
Model:                 ARIMA(1, 0, 1)   Log Likelihood               -1055.078
Date:                Fri, 04 Mar 2022   AIC                           2118.156
Time:                        22:23:23   BIC                           2135.014
Sample:                             0   HQIC                          2124.771
                                - 500
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.0166      0.645      1.576      0.115      -0.248       2.281
ar.L1          0.8276      0.029     28.376      0.000       0.770       0.885
ma.L1          0.2389      0.049      4.846      0.000       0.142       0.336
sigma2         3.9720      0.258     15.399      0.000       3.466       4.478
================================================================================
===
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
0.82
Prob(Q):                              0.98   Prob(JB):
0.66
Heteroskedasticity (H):               1.00   Skew:                             -
0.09
Prob(H) (two-sided):                  0.97   Kurtosis:
2.93
================================================================================
===

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-s
tep).
```

```
=============== if you want to extract the values ===============
The estimated coefficients are  [1.01659652 0.82761202 0.23889735 3.97203293]
The standard errors for the coefficients are  [0.64505658 0.02916641 0.0492956
0.25794199]
=================================================================
The Ljung-Box test statistic and p-value are
      lb_stat  lb_pvalue
10  4.359196   0.929692
=============== if you want to extract the values ===============
The test statistic is 4.359196
The p-value is 0.929692
```

If we do not impose the specification of ARMA model, we can use `auto_arima()` to find the optimal lags.

In [4]:

```python
# pip install pmdarima
import pmdarima as pm

ARMA_Est = pm.auto_arima(y, start_p=1, start_q=1, # initial guesses for p and q
                      d=0, # non-seasonal difference order
                      max_p=10, max_q=10, # maximum p and q
                      stepwise=False)
print(ARMA_Est.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  500
Model:               SARIMAX(1, 0, 1)   Log Likelihood               -1055.078
Date:                Fri, 04 Mar 2022   AIC                           2118.156
Time:                        22:23:27   BIC                           2135.014
Sample:                             0   HQIC                          2124.771
                                - 500
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.1753      0.118      1.489      0.137      -0.056       0.406
ar.L1          0.8276      0.029     28.376      0.000       0.770       0.885
ma.L1          0.2389      0.049      4.846      0.000       0.142       0.336
sigma2         3.9720      0.258     15.399      0.000       3.466       4.478
================================================================================
===
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
0.82
Prob(Q):                              0.98   Prob(JB):
0.66
Heteroskedasticity (H):               1.00   Skew:                           -
0.09
Prob(H) (two-sided):                  0.97   Kurtosis:
2.93
================================================================================
===

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-s
tep).
```

Here is a way to extract values like estimated coefficients and/or standard errors.

In [5]:

```
import pandas as pd
Est_html = ARMA_Est.summary().tables[1].as_html()
df = pd.read_html(Est_html, header=0, index_col=0)[0]
coef = df.iloc[:, 0]
se = df.iloc[:, 1]
print(coef)
print(se)
```

```
intercept    0.1753
ar.L1        0.8276
ma.L1        0.2389
sigma2       3.9720
Name: coef, dtype: float64
intercept    0.118
ar.L1        0.029
ma.L1        0.049
sigma2       0.258
Name: std err, dtype: float64
```

Augmented Dickey-Fuller test

$$
\begin{aligned}
Y_t &= \mu + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \epsilon_t \\
&= \mu + \rho Y_{t-1} + \beta_1 \Delta Y_{t-1} + \beta_{p-1} \Delta Y_{t-p+1} + \epsilon_t
\end{aligned}
$$

and the unit root test has null hypothesis $H_0 : \rho = 1$.

AIC or BIC is usually used for determining the number of lags p.

Under H_0,

$$
ADF = \frac{\hat{\rho} - 1}{s(\hat{\rho})} \rightarrow_d \frac{\int_0^1 U(r) dW(r)}{\left(\int_0^1 U(r)^2 dr\right)^{1/2}}
$$

We reject H_0 if the test statistic is less than the critical value at the significance level α (three forms: without intercept, with intercept, with trend). Check Table 1 in Prof. Liao's slides for DF testing critical values.

In [6]:

```
from statsmodels.tsa.stattools import adfuller

# augmented Dickey-Fuller test
adfuller(y)
# The test statistic, approximate p-value, the number of lags used.,
# The number of observations used for the ADF regression
# Critical values for the test statistic at the 1 %, 5 %, and 10 % levels
# The maximized information criterion if autolag is not None
```

Out[6]:

```
(-6.580412707132463,
 7.539024565676394e-09,
 1,
 498,
 {'1%': -3.4435494520411605,
  '5%': -2.8673612117611267,
```

```
  '10%': -2.5698704830567247},
 2036.5487950738197)
```

We reject H_0 at a significance level of 1% since the test statistic is ~~greater~~ smaller than -3.4435.

KPSS Stationary Test

$$Y_t = \beta_0 + S_t + e_t, \text{ where } S_t = S_{t-1} + \epsilon_t,$$

where $\epsilon_t \sim MDS(0, \sigma_\epsilon^2)$.

$H_0 : \sigma_\epsilon^2 = 0$ (Y_t is stationary) against $H_1 : \sigma_\epsilon^2 > 0$ (Y_t is a unit root process).

Intuitively, under H_0, $Y_t = \beta_0 + c + e_t$, where $c = S_t = S_{t-1} = \cdots$ It is a stationary process.

Under H_1, $Y_t - \beta_0 - e_t = Y_{t-1} - \beta_0 - e_{t-1} + \epsilon_t \Rightarrow Y_t = Y_{t-1} + e_t - e_{t-1} + \epsilon_t$. It is a unit root process.

Under H_0,

$$KPSS_1 = n^{-2}\hat{\omega}^{-2}\sum_{t=1}^{n}\left(\sum_{j=1}^{t}\hat{e}_j\right)^2 \rightarrow_d \int_0^1 V(r)^2 dr, \text{ where } V(r) = W(r) - rW(1) \text{ is the Bro}$$

We reject H_0 if the test statistic is **greater than** the critical value at the significance level α.

Also, we can allow for a linear time trend, and construct stationarity test statistic similarly.

$$Y_t = \beta_0 + \beta_1 t + S_t + e_t, \text{ where } S_t = S_{t-1} + \epsilon_t.$$

Check Table 2 in Prof. Liao's slides for KPSS testing critical values.

In [7]:

```python
from statsmodels.tsa.stattools import kpss
import warnings
warnings.filterwarnings("ignore") # suppress the warnings
kpss(y) # The KPSS test statistic, the p-value of the test, the truncation lag p
# The critical values at 10%, 5%, 2.5% and 1%
```

Out[7]:

```
(0.08479755597333012,
 0.1,
 12,
 {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739})
```

We cannot reject the stationarity at a significance level of 10% since the test statistic is less than 0.347.