TSA HW08 楊雲皓 R09546042

Q1

$Y_t = 5 + 1.1 Y_{t-1} - 0.5 Y_{t-2} + a_t$, where $\sigma_a^2 = 2$.

ⓐ

$\Rightarrow Y_t = 5 + 1.1 Y_{t-1} - 0.5 Y_{t-2} + a_t$

$= 5 + 1.1 (5 + 1.1 Y_{t-2} - 0.5 Y_{t-3} + a_{t-1}) - 0.5 Y_{t-2} + a_t$

$= 10.5 + 0.71 Y_{t-2} + 0.55 Y_{t-3} + a_t + 1.1 a_{t-1}$

$= 10.5 + a_t + \underline{1.1 a_{t-1}} + 0.71 Y_{t-2} + 0.55 Y_{t-3}$

$\therefore \underline{\psi_1 = 1.1}$

ⓑ set when
$\begin{cases} 2005, & t = 0, & Y_0 = 9 \\ 2006, & t = 1, & Y_1 = 11 \\ 2007, & t = 2, & Y_2 = 10 \\ 2008, & t = 3, & \hat{Y}_3 \\ 2009, & t = 4, & \hat{Y}_4 \end{cases}$

$\hat{Y}_3 = 5 + 1.1 Y_2 - 0.5 Y_1 + a_3$
$= 5 + 1.1 \times 10 - 0.5 \times 11 + 0$
$= 10.5$

$\hat{Y}_4 = 5 + 1.1 Y_3 - 0.5 Y_2 + a_4$
$= 5 + 1.1 \times 10.5 - 0.5 \times 10 + 0$
$= 11.55$

$\therefore \underline{\hat{Y}_3 = 10.5,\ \hat{Y}_4 = 11.55}$

ⓒ

95% C.I. on $\hat{Y}_3$ is $[\ \hat{Y}_3 \pm Z_{0.025} \sqrt{V(a_t)}\ ]$

C.I. : $\underline{[\ 10.5 \pm 1.96 \sqrt{2}\ ]}$

ⓓ

$\hat{Y}_4 = 5 + 1.1 Y_3 - 0.5 Y_2 + a_4$

$= 5 + 1.1 \times 12 - 0.5 \times 10 + 0$

$= 13.2$

Import packages

In [1]:
```
import statsmodels.api as sm
from statsmodels import tsa as TSA
from statsmodels.tsa.stattools import acf
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from statsmodels.tsa.statespace.sarimax import SARIMAX
import warnings
warnings.filterwarnings("ignore")
```

Q2.

Recall the dataset “robot” firstly introduced in TSA HW06.

In [2]:
```
df = pd.read_csv(r"C:\Users\TerryYang\Desktop\Github\2021-TSA-Assignment-NTUIIE\H
```

In [3]:
```
# split dataset
cut = 5
df_train = df[:-cut]
df_test = df[-cut:]
```

(a). Use IMA(1, 1) to forecast five values ahead and calculate the 95% confidence intervals.

In [4]:
```python
# IMA(1,1) model
IMA = ARIMA(df_train, order=(0,1,1)).fit()
IMA.summary()
```

Out[4]:

SARIMAX Results

Dep. Variable:	robot	**No. Observations:**	319
Model:	ARIMA(0, 1, 1)	**Log Likelihood**	1448.478
Date:	Tue, 28 Dec 2021	**AIC**	-2892.956
Time:	20:47:33	**BIC**	-2885.431
Sample:	0	**HQIC**	-2889.950
	- 319		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ma.L1	-0.6535	0.045	-14.657	0.000	-0.741	-0.566
sigma2	6.459e-06	4.98e-07	12.976	0.000	5.48e-06	7.43e-06

Ljung-Box (L1) (Q):	2.40	**Jarque-Bera (JB):**	2.02
Prob(Q):	0.12	**Prob(JB):**	0.36
Heteroskedasticity (H):	1.01	**Skew:**	0.08
Prob(H) (two-sided):	0.98	**Kurtosis:**	3.36

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

In [5]:
```python
# 95% confidence intervals
alpha = 0.05
IMA_Prediction = IMA.get_prediction(start=len(df_train), end=len(df)-1)
IMA_Prediction.summary_frame(alpha = alpha)
```

Out[5]:

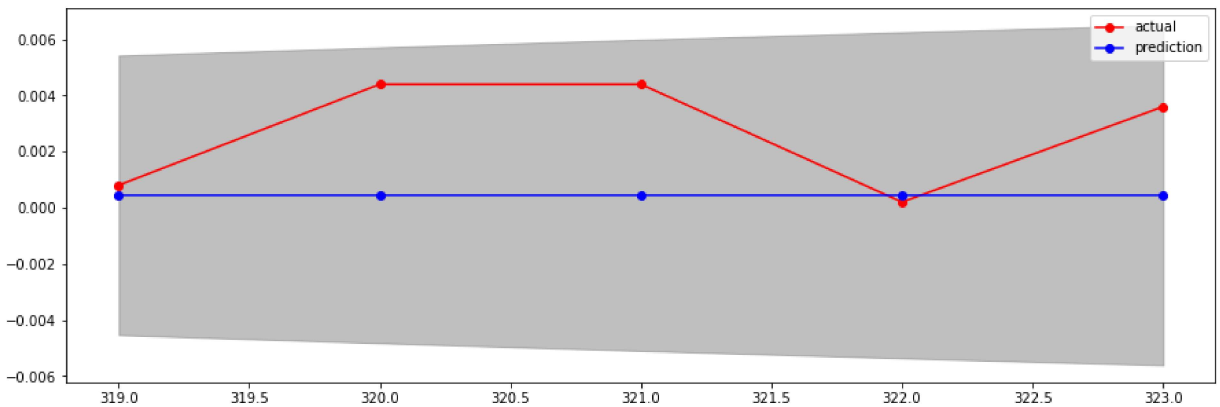
robot	mean	mean_se	mean_ci_lower	mean_ci_upper
319	0.000447	0.002542	-0.004535	0.005428
320	0.000447	0.002690	-0.004825	0.005718
321	0.000447	0.002830	-0.005100	0.005994
322	0.000447	0.002964	-0.005363	0.006256
323	0.000447	0.003092	-0.005614	0.006507

(b). Display the actual values, the five forecasts and the 95% confidence intervals of the five forecasts, all in one graph. What do you observe?

==> the confidence interval widen as t increase, and the actual values fall in the confidence region.

In [6]:
```
# get prediction
Pred_coef = IMA_Prediction.predicted_mean
Pred_coef_itv = IMA_Prediction.conf_int(alpha=alpha)
```

In [7]:
```
fig = plt.figure(figsize = (15,5))
ax = fig.add_subplot()
ax.plot(df_test, color = "red", marker = 'o', label="actual")
ax.plot(Pred_coef, color = "blue", marker = 'o', label="prediction")
ax.fill_between(x = Pred_coef_itv.index, y1 = Pred_coef_itv.iloc[:,0], y2 = Pred_
ax.legend()
plt.show()
```



(c). Use ARMA(1, 1) to forecast five values ahead and calculate the 95% confidence intervals. Compare the results with those in (a), what do you observe?

==> we receive similar result, but a slightly smaller AIC value(-2920 < -2892)

In [8]:
```python
# ARMA(1,1) model
ARIMA = ARIMA(df_train, order=(1,0,1)).fit()
ARIMA.summary()
```

Out[8]:

SARIMAX Results

Dep. Variable:	robot	**No. Observations:**	319
Model:	ARIMA(1, 0, 1)	**Log Likelihood**	1464.198
Date:	Tue, 28 Dec 2021	**AIC**	-2920.397
Time:	20:47:40	**BIC**	-2905.336
Sample:	0	**HQIC**	-2914.382
	- 319		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0013	0.000	3.821	0.000	0.001	0.002
ar.L1	0.8601	0.061	14.199	0.000	0.741	0.979
ma.L1	-0.6615	0.088	-7.492	0.000	-0.835	-0.488
sigma2	6.075e-06	4.63e-07	13.120	0.000	5.17e-06	6.98e-06

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	0.78
Prob(Q):	0.98	**Prob(JB):**	0.68
Heteroskedasticity (H):	1.02	**Skew:**	0.05
Prob(H) (two-sided):	0.92	**Kurtosis:**	3.22

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

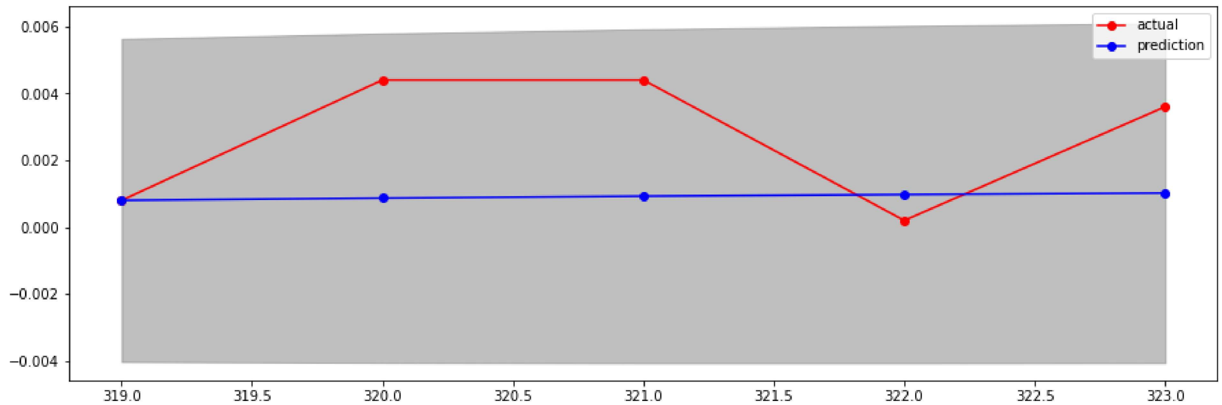
In [9]:
```python
ARIMA_Prediction = ARIMA.get_prediction(start=len(df_train), end=len(df)-1)
ARIMA_Prediction.summary_frame(alpha = alpha)
```

Out[9]:

robot	mean	mean_se	mean_ci_lower	mean_ci_upper
319	0.000801	0.002465	-0.004030	0.005632
320	0.000869	0.002513	-0.004057	0.005794
321	0.000926	0.002548	-0.004068	0.005920
322	0.000976	0.002574	-0.004068	0.006020
323	0.001019	0.002592	-0.004062	0.006100

In [10]:
```python
# get prediction
Pred_coef = ARIMA_Prediction.predicted_mean
Pred_coef_itv = ARIMA_Prediction.conf_int(alpha=alpha)
```

In [11]:
```
fig = plt.figure(figsize = (15,5))
ax = fig.add_subplot()
ax.plot(df_test, color = "red", marker = 'o', label="actual")
ax.plot(Pred_coef, color = "blue", marker = 'o', label="prediction")
ax.fill_between(x = Pred_coef_itv.index, y1 = Pred_coef_itv.iloc[:,0], y2 = Pred_
ax.legend()
plt.show()
```



Q3.

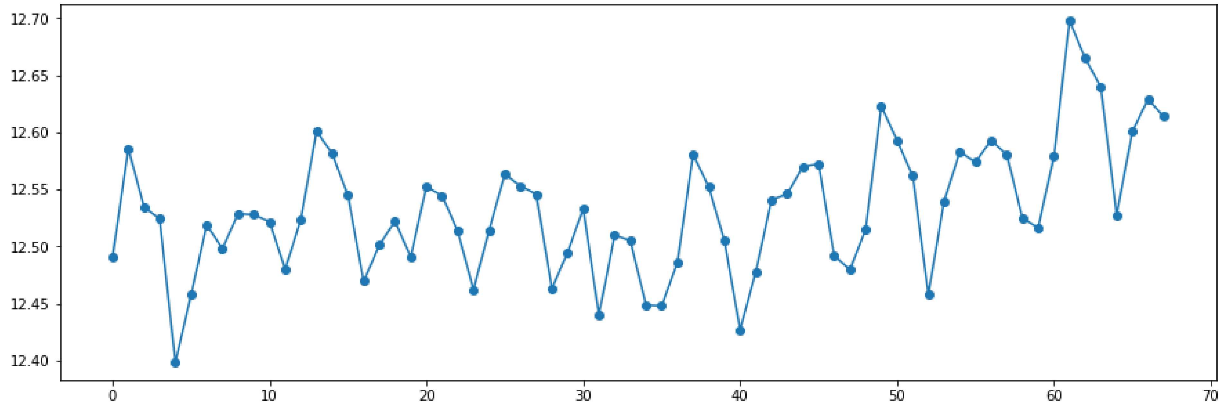
The dataset “boardings” contains the monthly number of passengers who boarded light rail trains and buses in Denver, Colorado, from August 2000 to March 2006.

In [12]:
```
df = pd.read_csv(r"C:\Users\TerryYang\Desktop\Github\2021-TSA-Assignment-NTUIIE\H
```

(a) Plot the time series and tell your observation if there exists seasonality and if the series is stationary.

==> within this time window, we can conclude that the series is stationary, and having a obivous trend.

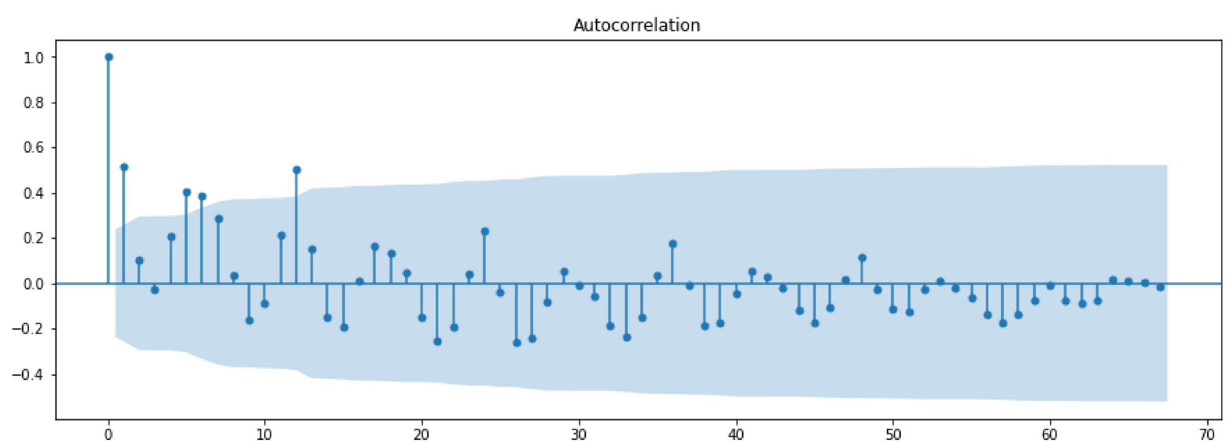
```
In [13]: fig = plt.figure(figsize = (15,5))
         plt.plot(df, marker = 'o')
         plt.show()
```



(b) Plot the sample ACF and see what are the significant lags?

==> at 0, 1, 5, 6, and 12, we have the significant lags.

```
In [14]: fig = plt.figure(figsize=(15,5))
         ax = fig.add_subplot()
         fig = sm.graphics.tsa.plot_acf(df, lags=len(df)-1, ax=ax)
```



(c) Fit the data with ARMA(0, 3) × (1,0)12, evaluate if the estimated coefficients {θ 1, θ 2, θ 3,ϕ 12} are significant. Hint: you need to check the associated standard errors "s.e." to the estimated coefficients to know if the coefficients are significant, via hypothesis testing.

In [15]:
```
SARIMA=sm.tsa.statespace.SARIMAX(endog=df,order=(0,0,3),seasonal_order=(1,0,0,12)
print(SARIMA.summary())
```

```
                                     SARIMAX Results
================================================================================
============
Dep. Variable:                     log_boardings   No. Observations:
68
Model:             SARIMAX(0, 0, 3)x(1, 0, [], 12)   Log Likelihood
118.602
Date:                           Tue, 28 Dec 2021   AIC
-225.205
Time:                                   20:47:54   BIC
-211.888
Sample:                                        0   HQIC
-219.928
                                            - 68
Covariance Type:                             opg
===============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------
intercept     10.3806      1.785      5.816      0.000       6.883      13.879
ma.L1          0.5596      0.155      3.619      0.000       0.256       0.863
ma.L2          0.2859      0.187      1.531      0.126      -0.080       0.652
ma.L3         -0.1668      0.141     -1.186      0.236      -0.442       0.109
ar.S.L12       0.1718      0.142      1.207      0.228      -0.107       0.451
sigma2         0.0018      0.000      4.514      0.000       0.001       0.003
================================================================================
====
Ljung-Box (L1) (Q):                   0.87   Jarque-Bera (JB):
1.70
Prob(Q):                              0.35   Prob(JB):
0.43
Heteroskedasticity (H):               1.42   Skew:                           -
0.32
Prob(H) (two-sided):                  0.41   Kurtosis:
3.43
================================================================================
====

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

Q4.

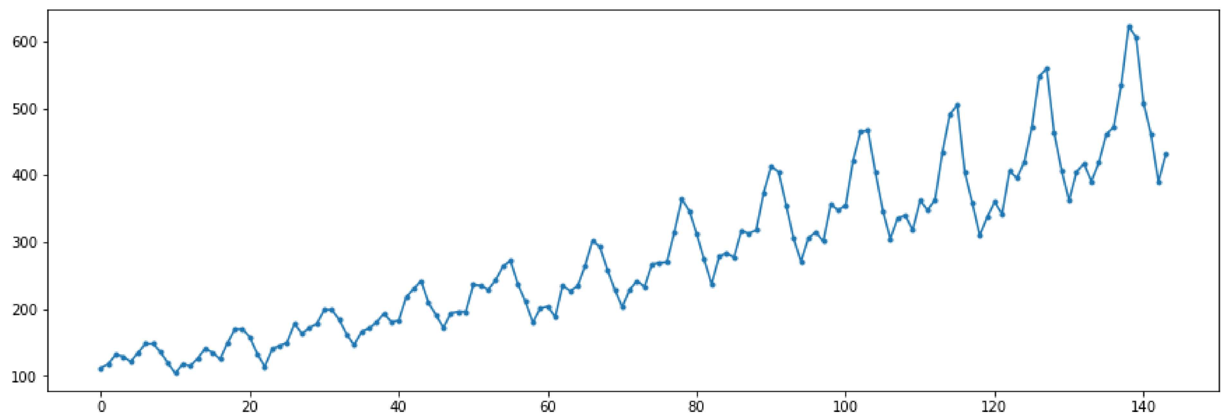
The monthly airline passengers, first investigated by Box and Jenkins in 1976, is considered as the classic time series dataset (see “TSA HW08.airpass.csv”).

In [16]:
```
df = pd.read_csv(r"C:\Users\TerryYang\Desktop\Github\2021-TSA-Assignment-NTUIIE\H
```

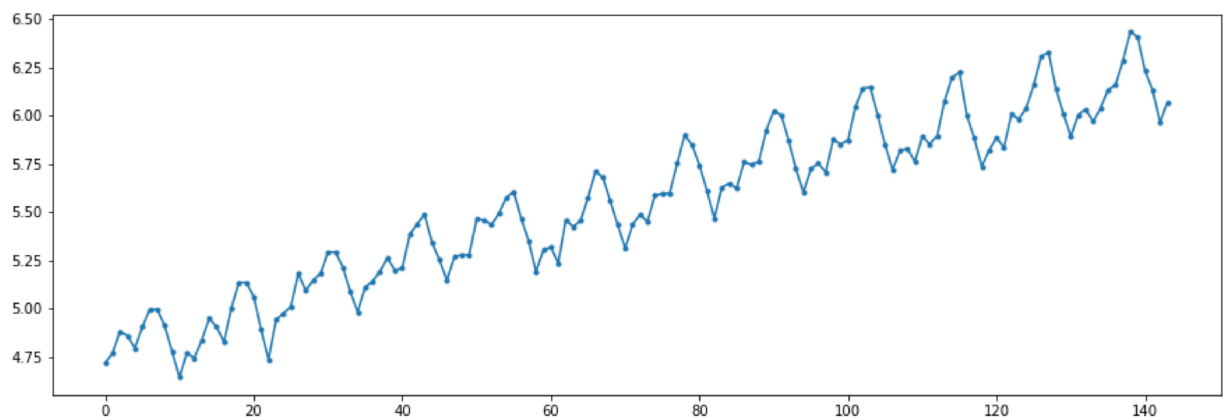
(a) Plot the time series in its original scale and the log-transformed scale. Do you think making the log-transformation is appropriate?

==> after log-trasformation, the series maintains its trending, and shrink the scale of the original series. Therefore, the transformation is appropriate

In [17]:
```
fig = plt.figure(figsize = (15,5))
plt.plot(df, marker = 'o', markersize = 3)
plt.show()
```



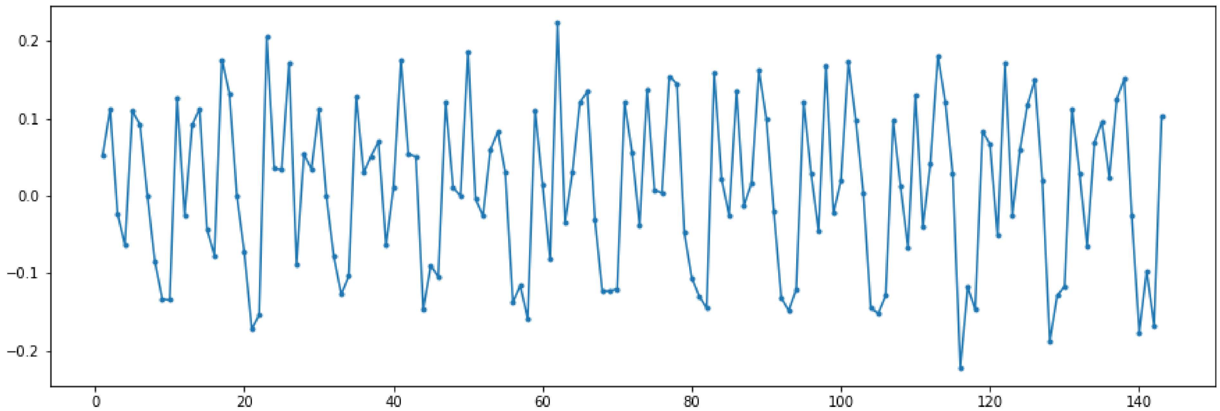
In [18]:
```
fig = plt.figure(figsize = (15,5))
plt.plot(np.log(df), marker = 'o', markersize = 3)
plt.show()
```



(b) Make the first-order difference over the “log-transformed” data. What do you observe?

==> after transfomation, original series seems to became stationary, and the trending is disappear

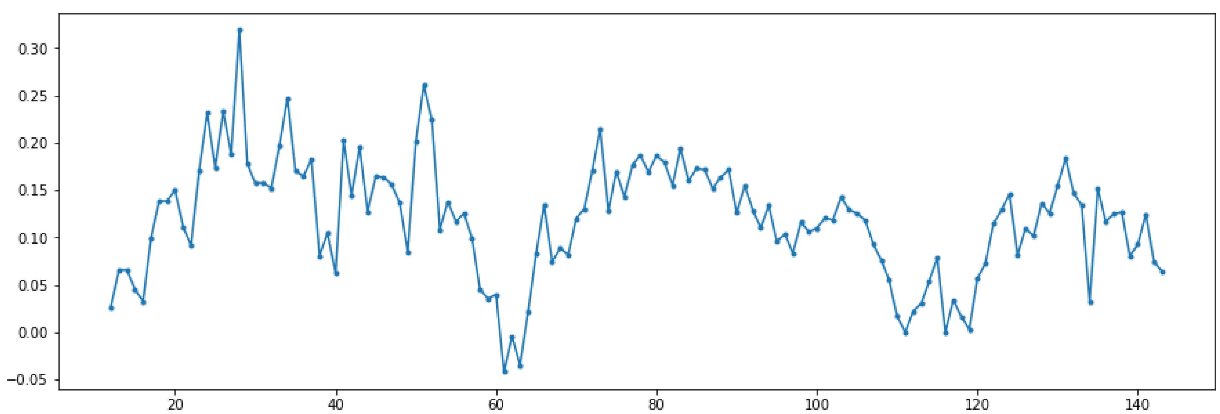
In [19]:
```python
fig = plt.figure(figsize = (15,5))
plt.plot(np.log(df).diff(periods = 1), marker = 'o', markersize = 3)
plt.show()
```



(c) Make a seasonal difference of the resulted series in (b), what do you observe?

==> after transformation, seasonality disappear.

In [20]:
```python
fig = plt.figure(figsize = (15,5))
plt.plot(np.log(df).diff(periods = 12), marker = 'o', markersize = 3)
plt.show()
```

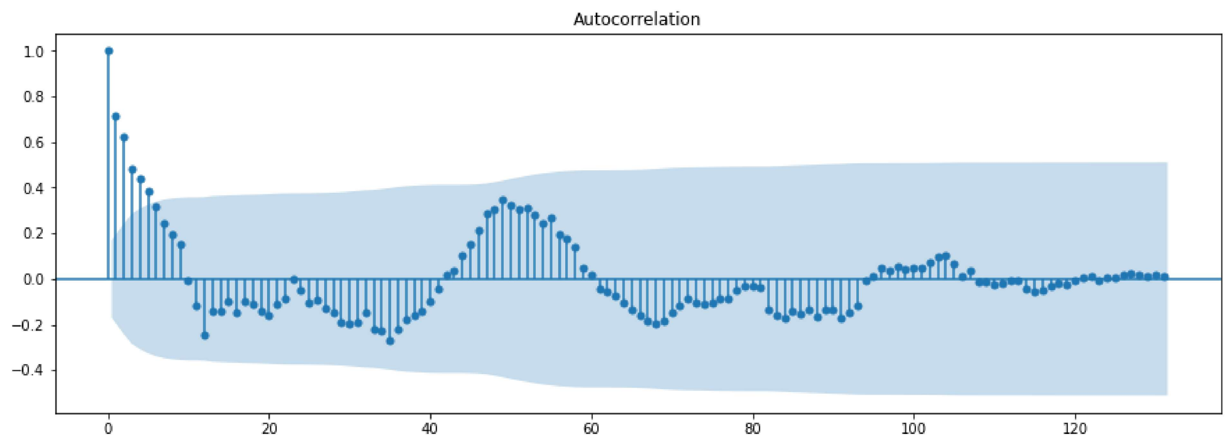


(d) Plot the sample ACF of the resulted series in (c), explain what you see.

==> after lag 5, ACF falls in control area.

In [21]:
```
# generate log series
log_series = np.log(df).diff(periods = 12)
log_series = log_series.dropna()
```

In [22]:
```
fig = plt.figure(figsize=(15,5))
ax = fig.add_subplot()
fig = sm.graphics.tsa.plot_acf(log_series, lags=len(log_series)-1, fft=False, ax
```



(e) Fit an ARIMA(0,1,1) × (0,1,1)12 model to the log-transformed series. Diagnose the residuals of this model, including the sample ACF and the normality test.

In [23]:
```
SARIMA=sm.tsa.statespace.SARIMAX(endog=np.log(df),order=(0,0,3),seasonal_order=(1
SARIMA.summary()
```

Out[23]:

SARIMAX Results

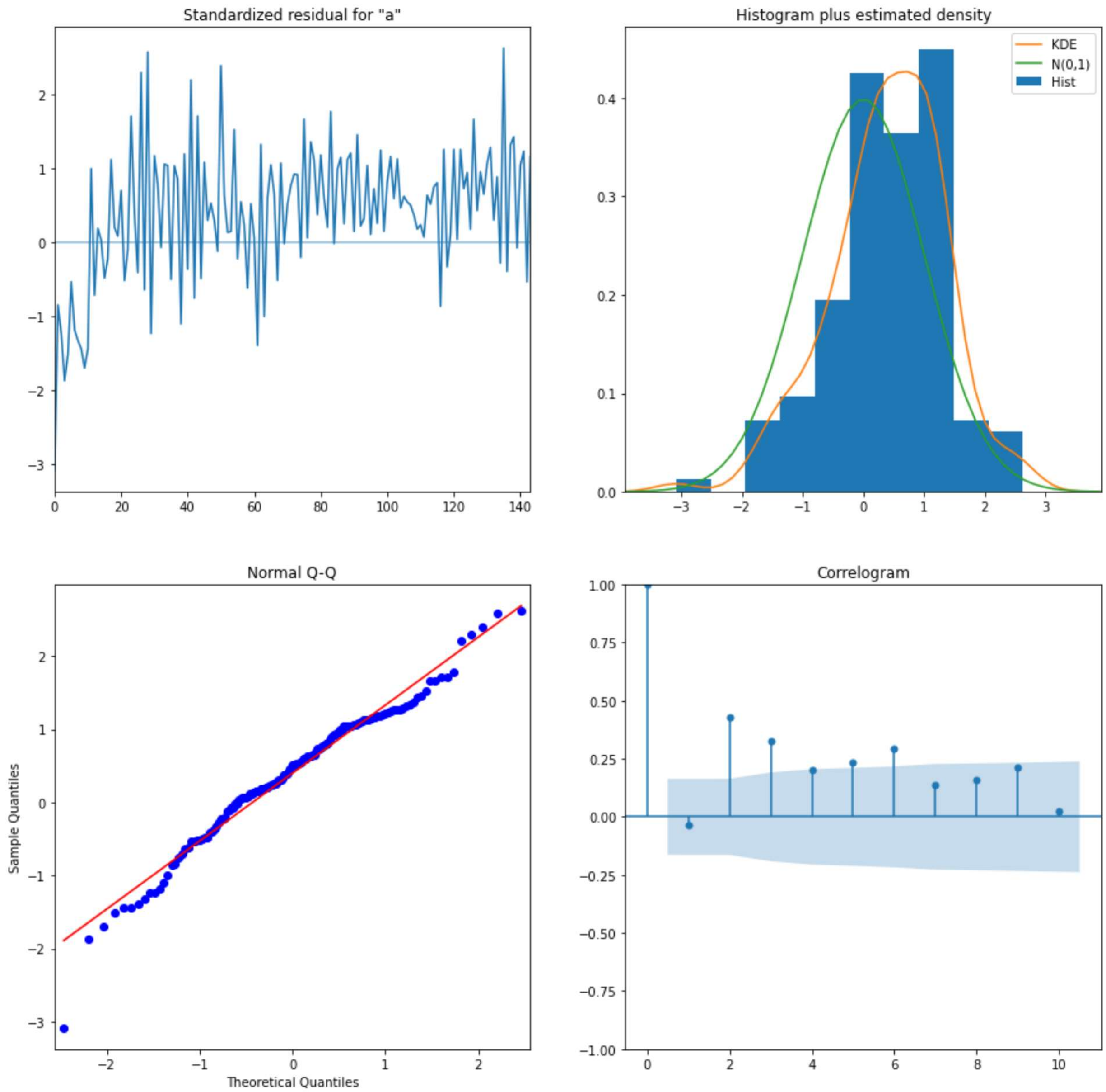
Dep. Variable:	airpass	**No. Observations:**	144
Model:	SARIMAX(0, 0, 3)x(1, 0, [], 12)	**Log Likelihood**	175.497
Date:	Tue, 28 Dec 2021	**AIC**	-338.994
Time:	20:48:09	**BIC**	-321.175
Sample:	0	**HQIC**	-331.753
	- 144		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	0.6260	0.192	3.260	0.001	0.250	1.002
ma.L1	3.4528	0.947	3.644	0.000	1.596	5.310
ma.L2	4.7947	1.646	2.913	0.004	1.569	8.021
ma.L3	4.0586	1.620	2.505	0.012	0.883	7.234
ar.S.L12	0.8862	0.036	24.630	0.000	0.816	0.957
sigma2	0.0003	0.000	1.271	0.204	-0.000	0.001

Ljung-Box (L1) (Q):	0.19	**Jarque-Bera (JB):**	10.25
Prob(Q):	0.66	**Prob(JB):**	0.01
Heteroskedasticity (H):	0.58	**Skew:**	-0.50
Prob(H) (two-sided):	0.06	**Kurtosis:**	3.84

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

In [24]:
```
SARIMA.plot_diagnostics(figsize = (15,15))
plt.show()
```



(f) Make forecasts for “two” years based on the model in (e). The

confidence intervals shall be included.

In [25]:
```
SARIMA_pre = SARIMA.get_prediction(start=len(np.log(df)), end=len(np.log(df))+24)
SARIMA_pre.predicted_mean

Pred_coef = SARIMA_pre.predicted_mean
Pred_coef_itv = SARIMA_pre.conf_int(alpha=alpha)

fig = plt.figure(figsize = (15,5))
ax = fig.add_subplot()
ax.plot(np.log(df), color = "red", marker = 'o', label="history")
ax.plot(SARIMA_pre.predicted_mean, color = "blue", marker = 'o', label="predictio
ax.fill_between(x = Pred_coef_itv.index, y1 = Pred_coef_itv.iloc[:,0], y2 = Pred_
ax.legend()
plt.show()
```