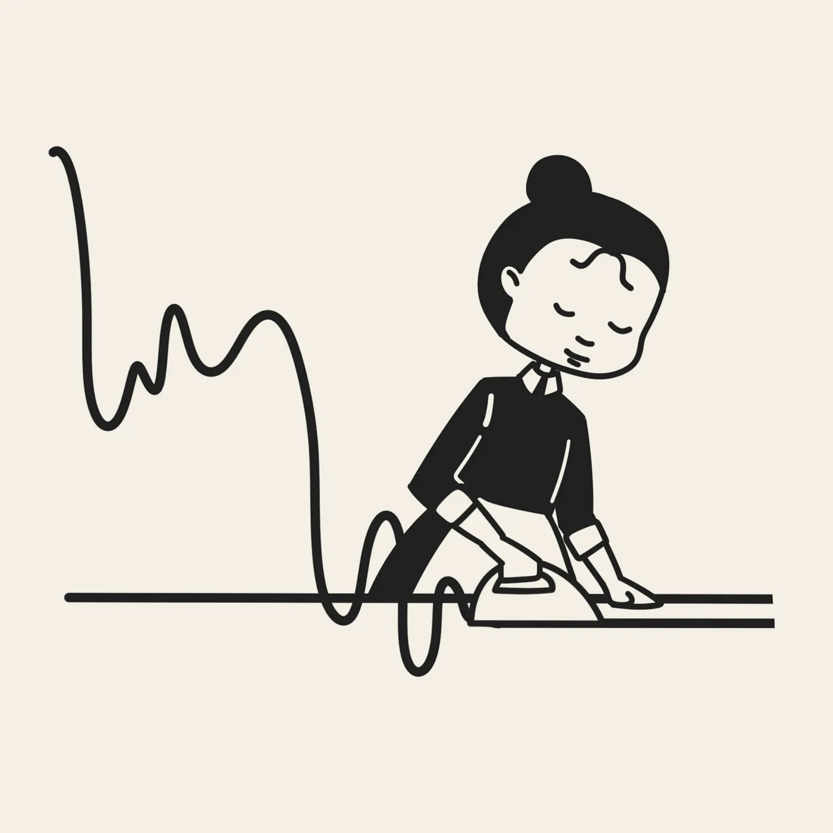
Stationarity in Time Series — A Comprehensive Guide

How to check if a time series is stationary and what you can do if it is non-stationary in Python



Stationarity in time series (Image drawn by the author)

The future is easier to model when it is similar to the present [3]. Stationarity describes the concept that the statistical features of a time series do not change over time. Thus, some time series forecasting models, such as autoregressive models, rely on the stationarity of the time series.

In this article, you will learn:

- what stationarity is,
- why it is important,
- 3 ways to check for stationarity, and
- 3 techniques you can apply when a time series is non-stationary

What Is Stationarity?

Stationarity describes the concept that how a time series is changing will remain the same in the future [3]. In mathematical terms, a time series is stationary when its **statistical properties are independent of time** [3]:

- constant mean,
- constant variance,
- and covariance is independent of time.

This is the definition of **weak-form stationarity**. Another type of stationarity is **strict stationarity**. It implies that samples of identical size have identical distribution [5]. Since strict stationarity is restrictive and rare, this article will only focus on weak-form stationarity.

Why Is Stationarity Important?

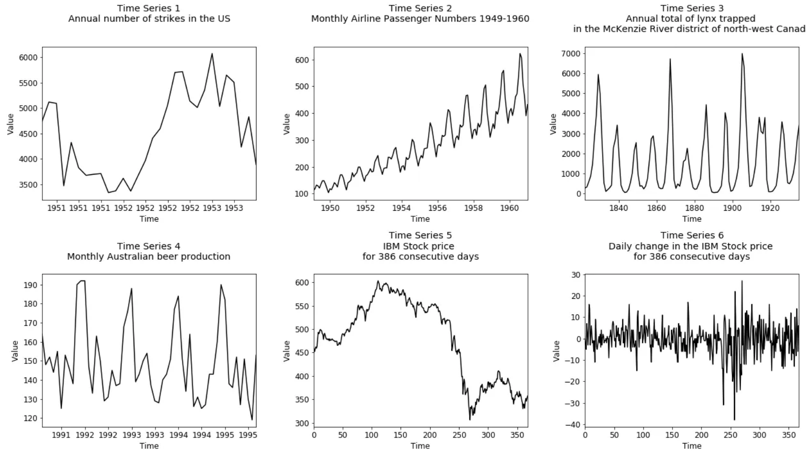
Some time series forecasting models (e.g., autoregressive models) require a stationary time series because they are easier to model due to their constant statistical properties [3]. Thus, you should make your time series stationary if it is not (see What Can You Do When A Time Series Is Not Stationary?).

How Do You Test For Stationarity?

You can test a time series for stationarity in two ways:

1. Intuitive approach: Visual assessment
2. Statistical approach: Unit root test

For this section, we will recreate a few examples Hyndman and Athanasopoulos [3] used to explain the visual assessment of stationarity and extend their usage also to explain testing for stationarity with unit root testing. The data is taken from the related `fma` R-package [1].

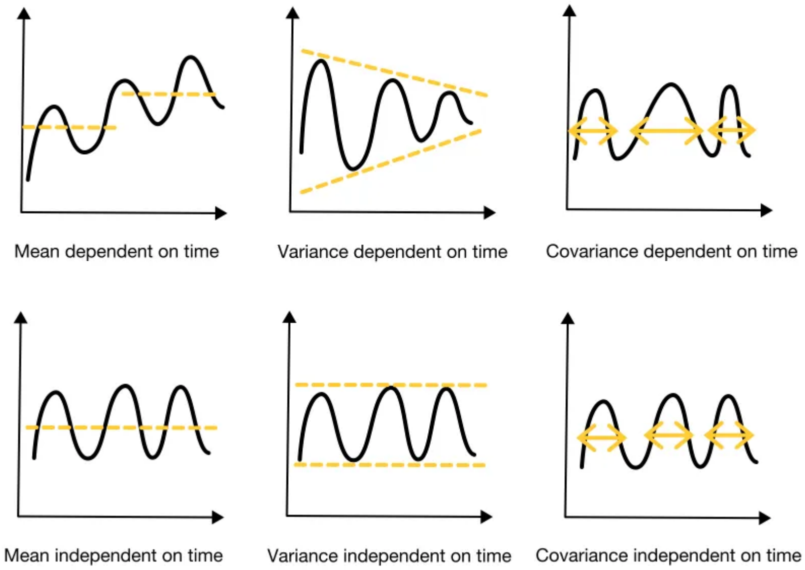


Example time series to showcase testing for stationarity (Image by the author, heavily inspired by Hyndman and Athanasopoulos [3])

How to visually assess stationarity

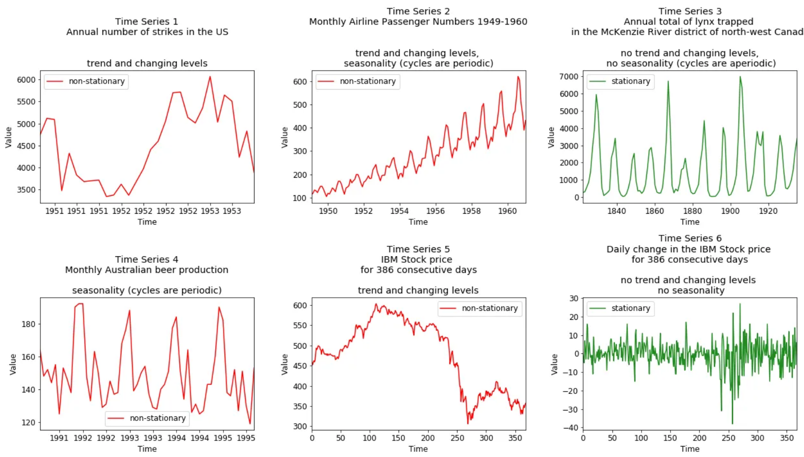
You can visually assess the stationarity of a time series by mentally dividing the time series in half and comparing the mean, amplitude, and cycle length from the first half to the second half of the time series.

- **constant mean** —The mean value of the first half of the time series should be similar to that of the second half.
- **constant variance** —The amplitude of the first half of the time series should be similar to that of the second half.
- **covariance is independent of time** — The cycle length in the first half of the time series should be similar to that in the second half. The cycles should be independent on time (e.g., not weekly or monthly, etc.).


Mean dependent on time
Variance dependent on time
Covariance dependent on time
Mean independent on time
Variance independent on time
Covariance independent on time

How to test for stationarity visually (Image by the author)

For our examples, the assessment result is visualized below:



Visual assessment of stationarity on example time series (Image by the author, heavily inspired by Hyndman and Athanasopoulos [3])

How to statistically assess stationarity — a detour on unit root tests

A **unit root** is a stochastic trend called a "random walk with drift". Since randomness can't be predicted, that means:

- Unit root present: not stationary (unpredictable)
- Unit root absent: stationary

To test for stationarity with a unit root test, you will state your initial assumption in the form of two competing hypotheses [6]:

- **Null hypothesis (H0)** — e.g., the time series is stationary (no unit root present)
- **Alternative hypothesis (H1)** — e.g., the time series is not stationary (unit root present)

Then you will assess whether to reject or not to reject the null hypothesis based on two approaches:

- **The p-value approach:**
  If the p-value > 0.05, fail to reject the null hypothesis.
  If the p-value ≤ 0.05, reject the null hypothesis.
- **The critical value approach:**
  If the test statistic is less extreme than the critical value, fail to reject the null hypothesis.
  If the test statistic is more extreme than the critical value, reject the null hypothesis.
  The critical value approach should be used when the p-value is close to significant (e.g., around 0.05) [8].

There are several unit root tests you can use to check for stationarity. This article will focus on the most popular ones:

- Augmented Dickey-Fuller test [2]
- Kwiatkowski-Phillips-Schmidt-Shin test [4].

How to test for stationarity with Augmented Dickey-Fuller test

The hypotheses for the Augmented Dickey-Fuller (ADF) test are [2]:

1. **Null hypothesis (H0):** The time series is **not stationary** because there is a unit root (if p-value > 0.05)
2. **Alternative hypothesis (H1):** The time series is **stationary** because there is no unit root (if p-value ≤ 0.05)

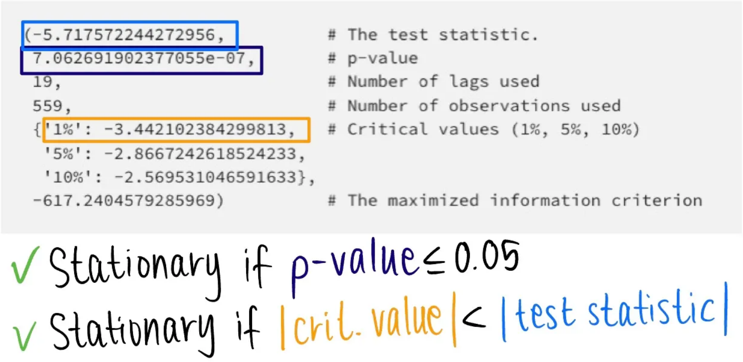
In Python, we can use the `adfuller` method from the `statsmodels.tsa.stattools` library [8].

```
from statsmodels.tsa.stattools import adfuller
```

```
result = adfuller(df["example"].values)
```

The time series is stationary if we can reject the null hypothesis of the ADF test:

- If the p-value (`result[1]`) ≤ 0.05
- If the test statistic (`result[0]`) is more extreme than the critical value (`result[4]["1%"]` , `result[4]["5%"]` , and `result[4]["10%"]`)



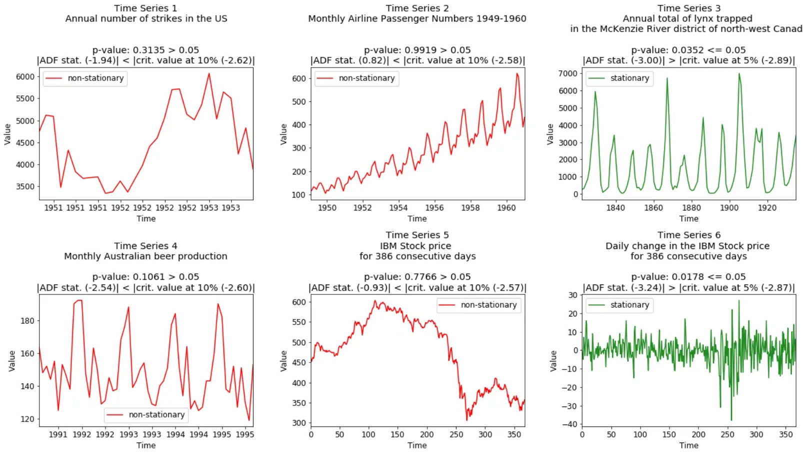
```
(-5.717572244272956,            # The test statistic.
 7.062691902377055e-07,         # p-value
 19,                            # Number of lags used
 559,                           # Number of observations used
 {'1%': -3.442102384299813,     # Critical values (1%, 5%, 10%)
  '5%': -2.8667242618524233,
  '10%': -2.569531046591633},
 -617.2404579285969)            # The maximized information criterion
```

✓ Stationary if p-value ≤ 0.05

✓ Stationary if |crit. value| < |test statistic|

Test for stationarity with ADF test from stattools Python library (Image by the author)

Below are the results from the ADF test for the sample dataset:



Unit root (ADF test) assessment of stationarity on example time series (Image by the author)

How to test for stationarity with Kwiatkowski-Phillips-Schmidt-Shin test

The hypotheses for the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test are [4]:

1. **Null hypothesis (H0):** The time series is **stationary** because there is no unit root (if p-value > 0.05)
2. **Alternative hypothesis (H1):** The time series is **not stationary** because there is a unit root (if p-value ≤ 0.05)

The more positive this statistic, the more likely we are to reject the null hypothesis (we have a non-stationary time series).

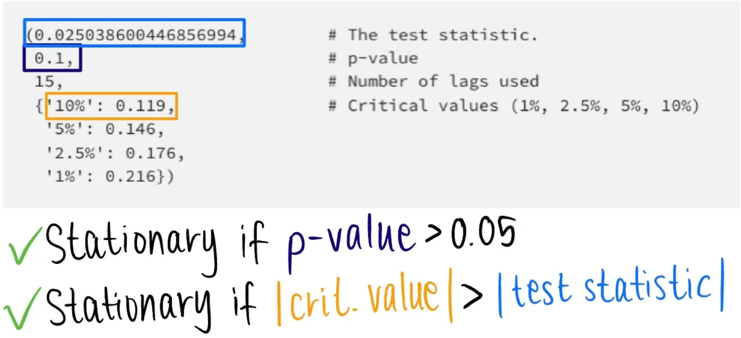
In Python, we can use the `kpss` method from the `statsmodels.tsa.stattools` library [9]. We must use the argument `regression = 'ct'` to specify that the test's null hypothesis is that the data is trend stationary. [9]

```
from statsmodels.tsa.stattools import kpss

result = kpss(df["example"].values,
              regression = "ct")
```

The time series is stationary if we fail to reject the null hypothesis of the KPSS test:

- If the p-value (`result[1]`) > 0.05
- If the test statistic (`result[0]`) is less extreme than the critical value (`result[3]["1%"]` , `result[3]["2.5%"]` , `result[3]["5%"]` , and `result[3]["10%"]`)



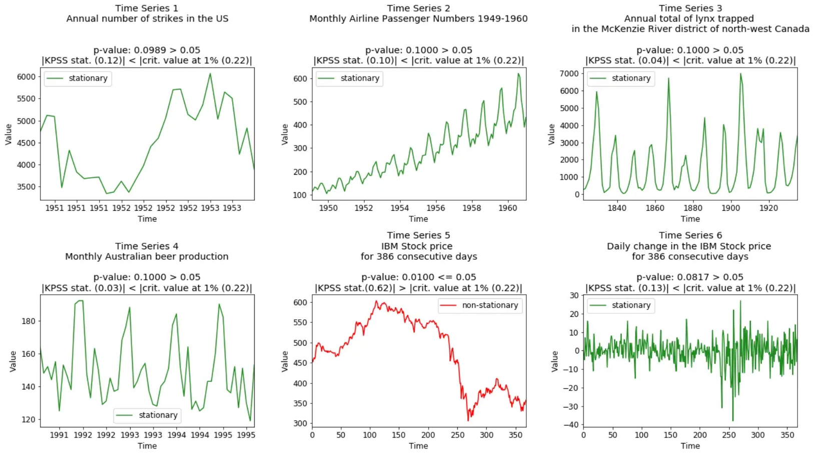
```
(0.025038600446856994,        # The test statistic.
 0.1,                         # p-value
 15,                          # Number of lags used
 {'10%': 0.119,               # Critical values (1%, 2.5%, 5%, 10%)
  '5%': 0.146,
  '2.5%': 0.176,
  '1%': 0.216})
```

✓Stationary if p-value > 0.05

✓Stationary if |crit. value| > |test statistic|

Test for stationarity with KPSS test from stattools Python library (Image by the author)

Below are the results from the KPSS test for the sample dataset:



Unit root (KPSS test) assessment of stationarity on example time series (Image by the author)

What Can You Do When A Time Series Is Not Stationary?

You can apply different transformations to a non-stationary time series to try to make it stationary:

- Differencing
- Detrending by model fitting
- Log transformation

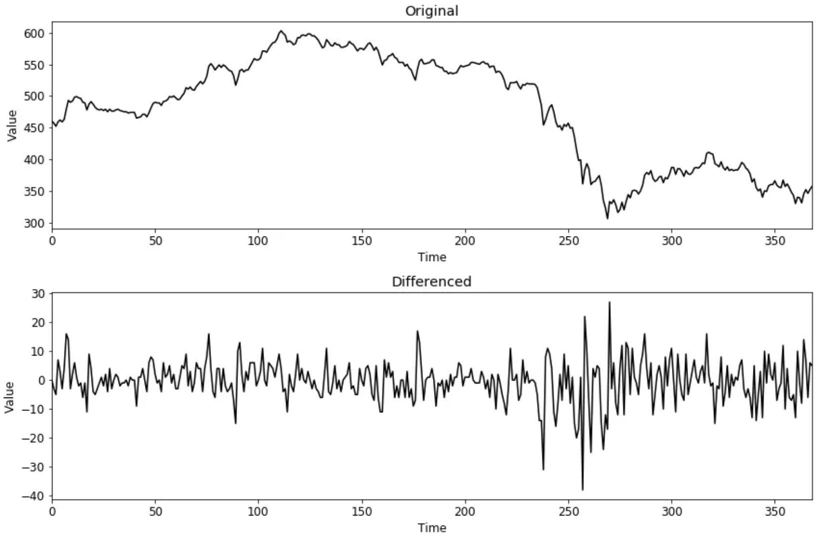
Because there are several stationarity types, we can combine the ADF and KPSS tests to determine what transformations to make [7]:

- If the ADF test result is stationary and the KPSS test result is non-stationary, the time series is **difference stationary** — Apply differencing to time series and check for stationarity again [7].
- If the ADF test result is non-stationary and the KPSS test result is stationary, the time series is **trend stationary** — Detrend time series and check for stationarity again [7].

Differencing

Differencing calculates the difference between two consecutive observations. It stabilizes the mean of a time series and thus reduces the trend [3].

```
df["example_diff"] = df["example"].diff()
```


Original

Differenced

Differenced time series (Image by the author)

If you want to expand your knowledge on differencing, you should have a look at fractional differencing.

Detrending by model fitting

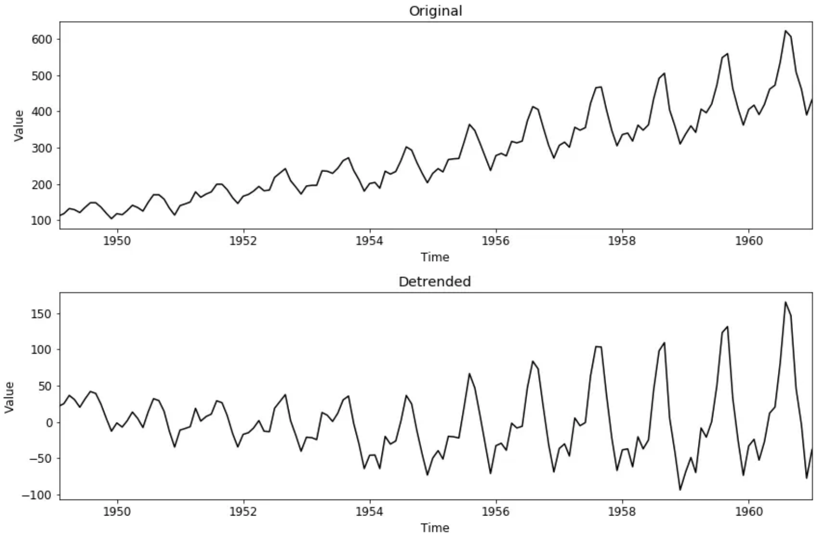
Another way to remove the trend from a non-stationary time series is to fit a simple model (e.g., linear regression) to the data and then to model the residuals from that fit.

```
from sklearn.linear_model import LinearRegression

# Fit model (e.g., linear model)
X = [i for i in range(0, len(airpass_df))]
X = numpy.reshape(X, (len(X), 1))
y = df["example"].values
model = LinearRegression()
model.fit(X, y)

# Calculate trend
trend = model.predict(X)

# Detrend
df["example_detrend"] = df["example"].values - trend
```


Original

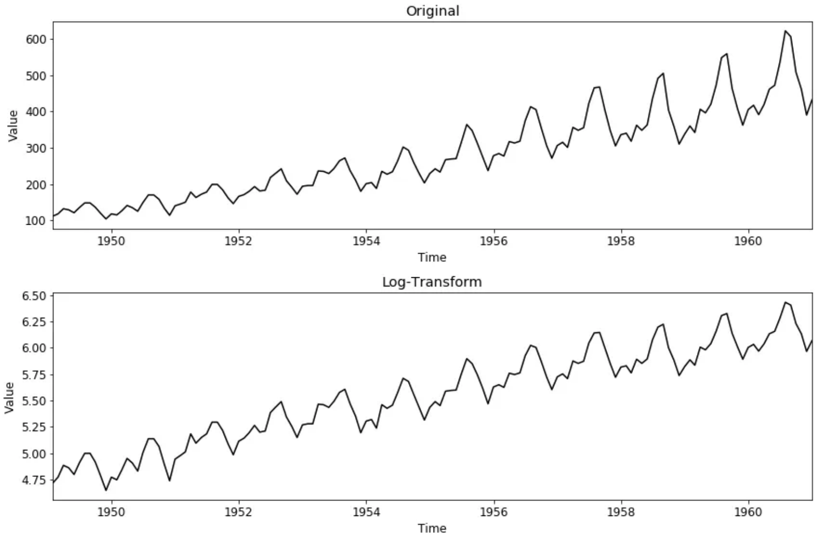
Detrended

Detrend time series by model fitting (Image by the author)

Log transformation

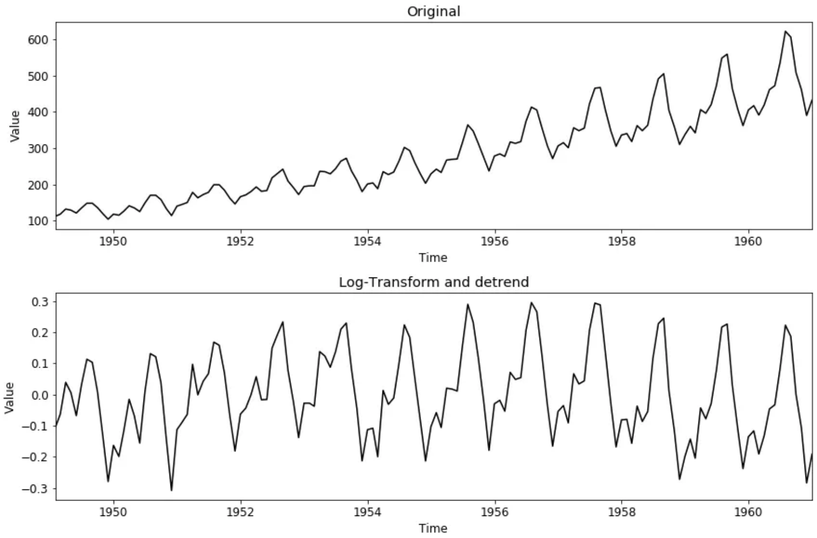
Log transformation stabilizes the variance of a time series [8].

```
df["example_diff"] = np.log(df["example"].value)
```



Log-transformed time series (Image by the author)

As you can see, both the detrending with model fitting as well as the log transform alone did not make our example time series stationary. You can also combine different techniques to make a time series stationary:



Detrend log-transformed time series by model fitting (Image by the author)

Summary

In time series forecasting, a time series, which has **constant statistical properties (mean, variance, and covariance)** and thus is independent of time, is described as stationary.

Because of the constant statistical characteristics, a stationary time series is easier to model than a non-stationary time series. Thus, a lot of time series forecasting models **assume stationarity**.

Stationarity can be checked either **by visual assessment or by a statistical approach**. The statistical approach checks for a unit root, an indicator of non-stationarity. The two most popular unit root tests are ADF and KPSS. Both are available in the Python `stattools` library [8,9].

If a time series is non-stationary, you can try to **make it stationary** by differencing, log transforming, or removing the trend.