

How to Easily Perform a KPSS Stationarity Test in Python

Authored by
stats writer

December 1, 2025

RECOMMENDED CITATION

stats writer (2025). *How to Easily Perform a KPSS Stationarity Test in Python*.
PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=102938>

The **KPSS Test** (Kwiatkowski-Phillips-Schmidt-Shin test) is an essential diagnostic tool within **time series** analysis, used primarily to assess whether a given data series is **stationary** around a deterministic trend or level. Unlike traditional unit root tests, the KPSS test has a unique null hypothesis that makes it particularly useful for confirming stationarity rather than simply looking for non-stationarity. Understanding the state of stationarity is fundamental, as many sophisticated time series models, such as ARIMA, rely on the assumption that the underlying statistical properties of the data (mean, variance, and autocorrelation) remain constant over time.

Performing the **KPSS Test** efficiently requires specialized statistical software. In the Python ecosystem, the powerful **statsmodels** library provides the necessary functionality. The test takes the time series data as input and calculates a test statistic, which, when compared against critical values or converted into a **p-value**, determines the appropriate conclusion regarding the series' stability. A critical outcome of this procedure is the identification of potential trends or structural breaks, which are crucial precursors to accurate forecasting and data modeling.

The Concept of Stationarity in Data Modeling

A time series is considered **stationary** if its statistical properties do not change over time. This means that the mean and variance are constant, and the autocorrelation structure depends only on the lag, not the specific time point. This concept is vital because non-stationary data often leads to spurious regression results, making causal inferences unreliable. Stationarity is a prerequisite for applying many classical econometric models.

However, real-world data rarely exhibits perfect stationarity. Two common types of non-stationarity are typically addressed: **trend stationarity** and difference stationarity. A series is trend stationary if it fluctuates around a deterministic trend (like a linear line) but is otherwise stationary; removing this trend makes the residuals stationary. The **KPSS Test** is specifically designed to identify if a series is stationary around a level or a deterministic trend, confirming stability once the trend component is accounted for.

KPSS vs. Unit Root Tests: A Crucial Distinction

When assessing stationarity, analysts often choose between two major types of tests: unit root tests (like the Augmented Dickey-Fuller, or ADF, test) and the KPSS test. The fundamental difference lies in their formulation of the **null hypothesis**. The ADF test assumes the series is non-stationary (H_0 : Unit Root exists), meaning we look for evidence strong enough to reject non-stationarity.

Conversely, the **KPSS Test** operates under the assumption that the series is stationary (H_0 : No Unit Root/Trend Stationarity). This makes the KPSS test more stringent for confirming stability. If

the series is found to be non-stationary (H_A is accepted), it implies the non-stationarity is likely due to a persistent structural break or a stochastic trend, often requiring differencing to achieve stationarity. Using both ADF and KPSS tests simultaneously provides a robust mechanism to definitively classify a time series.

Formulating the Hypotheses for the KPSS Test

The mathematical structure of the **KPSS Test** is defined by its core hypotheses, which dictate how the test statistic and resulting **p-value** are interpreted. It is essential to correctly identify the test setup before drawing conclusions. The KPSS test can be used to determine if a **time series** is trend stationary or level stationary, depending on the regression parameter chosen during implementation.

This test uses the following **null and alternative hypothesis**:

H0: The **time series** is trend **stationary**.

HA: The **time series** is *not* trend **stationary**.

The decision rule is based on the **p-value** (P) or the test statistic (TS). If the **p-value** of the test is less than some predetermined significance level (e.g., $\alpha = .05$), then we reject the **null hypothesis** (H_0) and conclude that the time series is non-stationary. Otherwise, if $P \geq \alpha$, we fail to reject H_0 , suggesting the series is stable (trend stationary or level stationary).

Implementation Prerequisites: Utilizing the statsmodels Library

To execute the KPSS Test in Python, we rely on the extensive statistical capabilities provided by the **statsmodels** library. This package is the industry standard for econometric and statistical modeling in Python. Specifically, the function needed is **sm.tsa.stattools.kpss()**, which is designed to handle the nuances of stationarity testing.

Before running the test, it is good practice to visualize the data using tools like Matplotlib and ensure data handling is managed by libraries such as NumPy. The **kpss()** function accepts several parameters, most importantly the time series array and the **'regression'** parameter, which specifies whether to test for stationarity around a mean ('c' for constant) or around a linear trend ('ct' for constant plus trend). In the examples below, we will use the **'ct'** option, testing for trend stationarity.

Example 1: Applying the KPSS Test to Stationary Data

This first example demonstrates the application of the **KPSS Test** to synthetic data that is intentionally generated to be **stationary**. We begin by importing necessary libraries (NumPy for

data generation and Matplotlib for visualization) and creating a reproducible dataset consisting of 100 observations drawn from a standard normal distribution.

First, let's create some fake data in Python to work with:

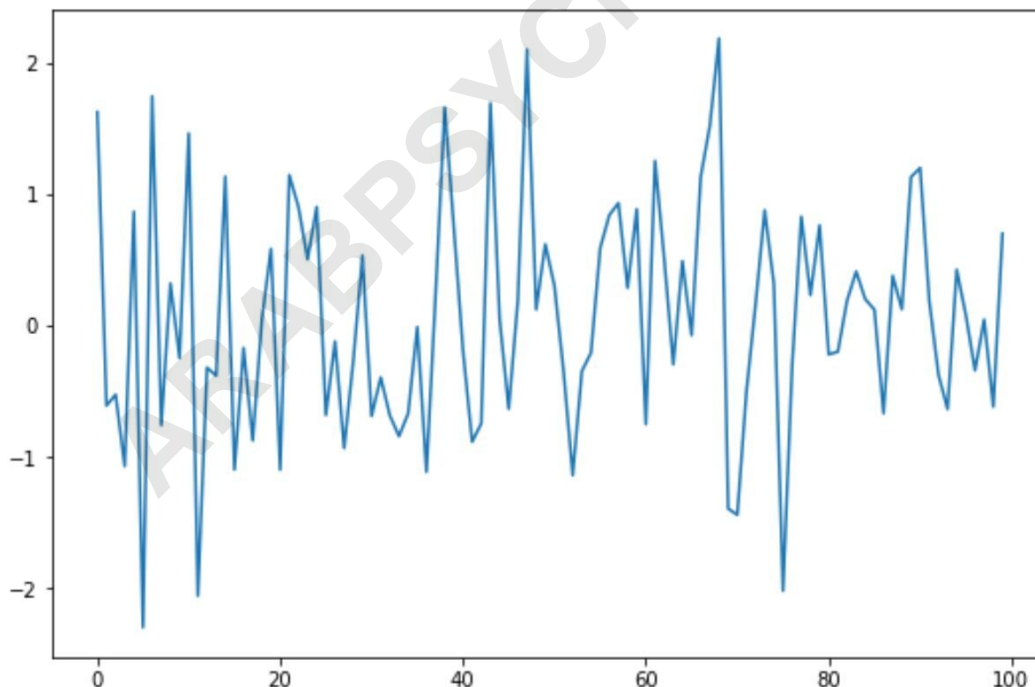
```
import numpy as np
import matplotlib.pyplot as plt

#make this example reproducible
np.random.seed(1)

#create time series data
data = np.random.normal(size=100)

#create line plot of time series data
plt.plot(data)
```

The resulting plot visually confirms the **stationary** nature of the data, showing fluctuations around a constant mean (zero) without any discernible upward or downward trend. This visual assessment provides initial confirmation for the statistical test we are about to run.



We can now use the **kpss()** function from the **statsmodels** package to formally perform a **KPSS Test** on this time series data. We specify `regression='ct'` to test for trend stationarity.

```
import statsmodels.api as sm
```

```
#perform KPSS test
```

```
sm.tsa.stattools.kpss(data, regression='ct')
```

```
(0.0477617848370993,
```

```
0.1,
```

```
1,
```

```
{'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216})
```

InterpolationWarning: The test statistic is outside of the range of p-values available in the look-up table. The actual p-value is greater than the p-value returned.

The output provides four key elements that must be analyzed for a proper statistical conclusion. Here's how to interpret the output tuple:

The **KPSS test** statistic: **0.04776**

The **p-value**: **0.1**

The truncation lag parameter: **1** (Used in the calculation of the long-run variance estimate.)

The critical values at **10%**, **5%**, **2.5%**, and **1%** (Used for comparison against the test statistic if the p-value is unavailable or unreliable.)

The calculated **p-value** is **0.1**. Since this value is significantly greater than our common significance level of $\alpha = 0.05$, we consequently fail to reject the **null hypothesis** of the KPSS test. This critical result confirms that the time series is indeed trend stationary, aligning with our visual inspection and understanding of how the data was generated.

Note 1: The **kpss()** function often returns a maximum **p-value** of 0.1, even if the actual statistical probability is higher. The accompanying warning indicates that the test statistic is so small that it falls outside the interpolation range of the look-up tables used by **statsmodels**, meaning the evidence supporting the null hypothesis is very strong.

Example 2: Applying the KPSS Test to Non-Stationary Data

In contrast to the first example, we now analyze a dataset that exhibits clear characteristics of **non-stationarity**, specifically a strong upward trend. This stochastic trend means the mean of the series is clearly increasing over time, violating the basic assumption of stationarity. We again use NumPy and Matplotlib to set up and visualize this non-stationary **time series**.

First, let's create some fake data in Python to work with:

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
#make this example reproducible
```

```
np.random.seed(1)
```

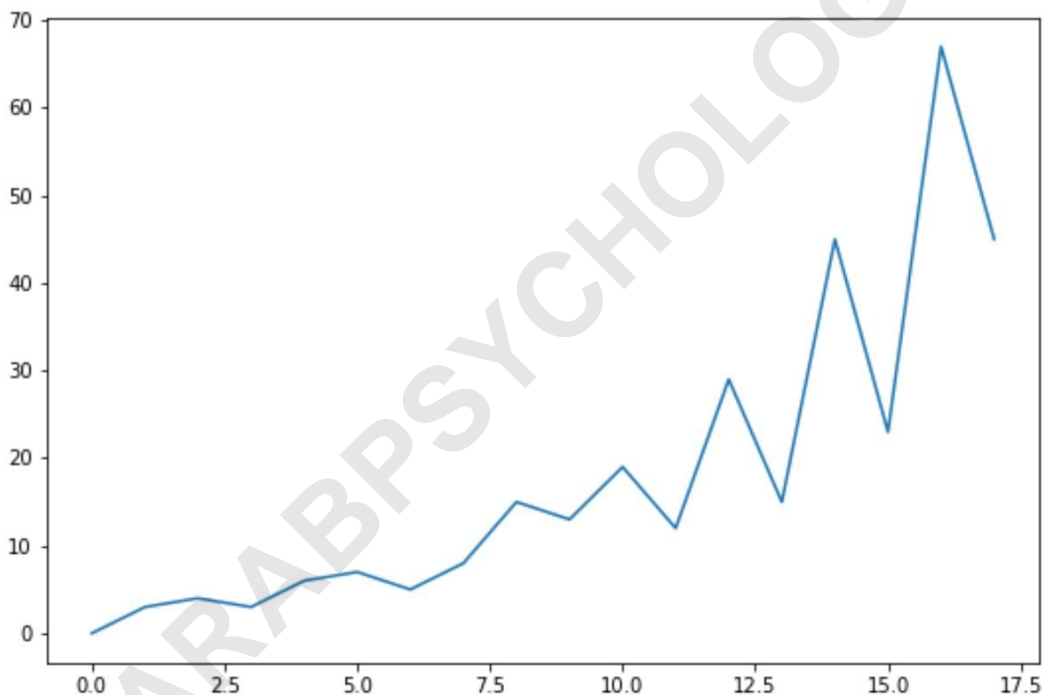
```
#create time series data
```

```
data =np.array()
```

```
#create line plot of time series data
```

```
plt.plot(data)
```

The resulting line plot clearly shows an increasing average level over the sequence of observations, which visually flags the data as non-stationary. This strong visual evidence suggests that the **KPSS Test** should lead to the rejection of the **null hypothesis** of stationarity.



Once again, we use the **kpss()** function from the **statsmodels** package to perform the statistical evaluation, retaining `regression='ct'` to test whether the series is stationary around a deterministic trend.

```
import statsmodels.api as sm
```

```
#perform KPSS test
```

```
sm.tsa.stattools.kpss(data, regression='ct')
```

```
(0.15096358910843685,  
0.04586367574296928,  
3,  
{'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216})
```

Here's how to interpret the output of the non-stationary case:

The KPSS test statistic: **0.1509**

The **p-value**: **0.0458**

The truncation lag parameter: **3**

The critical values at **10%**, **5%**, **2.5%**, and **1%**

The resulting **p-value** is **0.0458**. Since this value is less than the typical significance level of 0.05, we have sufficient evidence to reject the **null hypothesis** of the KPSS test. This statistical rejection strongly indicates that the time series is *not* trend **stationary**, confirming the visual trend we observed.

Conclusion: Importance of Stationarity Testing

The ability to accurately determine whether a **time series** is stationary is paramount for successful statistical modeling and forecasting. The **KPSS Test** provides a powerful method, particularly due to its **null hypothesis** structure, which allows analysts to confidently assert stationarity when the evidence supports it. By integrating the **kpss()** function from **statsmodels** into a Python workflow, we can efficiently perform this diagnosis and prepare our data for advanced analysis techniques like ARIMA or GARCH modeling.

Remember that the choice of regression parameter ('c' vs. 'ct') is essential and determines whether you are testing for stationarity around a fixed mean or around a deterministic trend.

Note: You can find the complete documentation for the **kpss()** function from the **statsmodels** package [here](#).

Further Resources for Time Series Analysis

The following tutorials provide additional information on how to work with time series data in Python, deepening your expertise in data preparation and modeling: