Perform Runs Test in Python

Authored by

Mohammed looti

November 7, 2025

The Runs test, formally recognized as the **Wald-Wolfowitz Runs Test**, stands as a crucial non-parametric statistical tool. Its primary function is to rigorously evaluate whether the sequential order of observations within a dataset suggests that the data originated from a truly random process. Unlike tests that examine the distribution or magnitude of data points, the Runs Test focuses exclusively on the qualitative aspect of data order, seeking out underlying structures, systematic biases, or inherent patterns that would fundamentally contradict the core assumption of randomness. This makes it exceptionally valuable in diagnostics where the chronological or sequential arrangement of measurements is paramount.

In various analytical fields, especially when dealing with time series data, rigorous quality control measurements, or any sequence where the precise order of occurrence carries significant meaning, establishing the presence or absence of randomness is a foundational requirement. Non-random behavior often manifests as undesirable characteristics such as strong serial correlation (where adjacent values are dependent), discernible trends (long-term drift), or cyclical behavior (periodic fluctuations). Detecting these deviations early is essential, as their presence invalidates the assumptions of simpler models and necessitates the deployment of more sophisticated, time-aware statistical and machine learning techniques.

This comprehensive tutorial is designed to provide a robust understanding of the theoretical underpinnings of the Runs Test. More importantly, it offers clear, practical, and replicable instructions for its precise implementation. We will utilize the extensive capabilities of the Python programming language, specifically leveraging the specialized statistical functions provided by the **statsmodels** library, ensuring a reliable and academically sound approach to assessing sequential randomness.

Defining and Conceptualizing the Runs Test

At the heart of the Runs Test lies the concept of a "run." A run is precisely defined as a contiguous sequence of identical observations. For the purpose of the single-sample Wald-Wolfowitz test, the raw numerical data must first be transformed into a binary sequence. This transformation is achieved by comparing each observation against a specified central tendency measure, or "cutoff point," typically the sample mean or median. Once dichotomized, the sequence consists solely of two types of outcomes--for example, values 'A' (above the cutoff) and values 'B' (below the cutoff). A run is then simply an uninterrupted stretch of consecutive 'A's or consecutive 'B's.

To illustrate this concept, consider a transformed sequence based on the median cutoff: A A B B A B B B A A. By grouping the consecutive identical outcomes, we can clearly identify five distinct runs: the first run is (A A), followed by the second run (B B), the third run is a single observation (A), the fourth run is (B B B), and finally, the fifth run is (A A). The fundamental premise of this statistical test involves comparing the actual count of runs observed in the real data sequence

against the expected number of runs that would theoretically occur if the data generating process were perfectly random. Deviations from this expected count form the basis for statistical inference.

Significant deviations in the number of observed runs provide critical insight into the nature of non-randomness. If the number of runs is statistically lower than expected, it implies excessive clumping or grouping, suggesting strong positive serial correlation (e.g., A A A A B B B B). This means that high values tend to be followed by high values, and low values by low values. Conversely, if the number of observed runs is significantly higher than expected, it points toward systematic oscillation or strong negative serial correlation (e.g., A B A B A B). This alternating pattern indicates that the data points are rapidly switching sides relative to the cutoff, a clear signature of non-random, potentially cyclical behavior.

Framing the Hypothesis for Randomness Testing

As is standard practice in all formal inferential statistical procedures, the Runs test must be meticulously framed within the context of null and alternative hypotheses. These statements serve to precisely define the assumption being rigorously tested and the specific conclusion that the statistical evidence must support in order to reject that initial assumption. This structured approach ensures that conclusions are based on quantifiable evidence rather than subjective interpretation.

For the single-sample Runs Test, the hypotheses are formally stated as follows:

H0 (Null Hypothesis): The underlying sequence of data values was generated by a truly random process. This hypothesis posits that the order of the observations is independent and that the observed number of runs is consistent with pure chance.
Ha (Alternative Hypothesis): The sequence of data values was *not* produced in a random manner. Rejection of the null hypothesis in favor of the alternative suggests the definite presence of systematic trends, distinct patterns, or pronounced cyclical behavior within the data sequence.

The primary statistical objective is to determine whether the data provides sufficient evidence, typically measured through a calculated p-value, to warrant the rejection of the null hypothesis (H0). If the calculated test statistic indicates that the observed count of runs is highly unlikely under the assumption of perfect randomness, we reject H0 in favor of Ha, thereby concluding definitively that the data exhibits statistically significant non-randomness.

A critical decision in setting up the test involves selecting the appropriate cutoff point for data dichotomization. While the **mean** of the dataset is often used as the default threshold, the **median** is frequently the preferred choice, particularly when the underlying data distribution is known or suspected to be skewed. Using the median ensures that the sequence is partitioned into two groups (above and below the median) of approximately equal size. This property is highly desirable and often requisite for maintaining the statistical validity of non-parametric tests like the Wald-

Wolfowitz Runs Test, as it minimizes the impact of outliers.

Implementing the Runs Test using Python's statsmodels Library

To execute the Runs Test efficiently and reliably within the Python environment, data scientists rely heavily on the powerful statsmodels library, which provides comprehensive tools for statistical modeling and hypothesis testing. The specific function tailored for the single-sample Runs Test is `runstest_1samp()`. A thorough understanding of its syntax and parameters is crucial for accurate and reproducible statistical application.

The function signature is structured concisely, offering flexibility in how the cutoff is defined and whether corrections are applied:

```
runstest_1samp(x, cutoff='mean', correction=True)
```

The functionality of the test is governed by its three primary parameters:

x: This required parameter mandates the input of the numerical data sequence. It must be provided as an array, list, or similar iterable structure containing the values intended for the test of sequential randomness.
cutoff: This parameter defines the critical threshold utilized to dichotomize the data into 'runs.' The default setting is `'mean'`, which employs the arithmetic mean of the dataset. Alternatively, specifying `'median'` instructs the function to use the precise median value of the data as the dividing point, often improving robustness against distributional asymmetry.
correction: This Boolean parameter controls the application of a continuity correction (subtracting 0.5) to the calculated test statistic. For practical purposes, especially when dealing with smaller sample sizes (conventionally N < 50), statistical theory strongly advises applying this continuity adjustment to enhance the accuracy of approximating the discrete distribution of the number of runs with the continuous normal distribution. By default, this is set to `True`, applying the correction automatically. Setting it to `False` disables the adjustment.

The execution of the `runstest_1samp()` function yields a highly informative tuple. This output contains two essential statistical measures: the calculated z-test statistic and its corresponding two-sided p-value. These metrics are the foundational elements required for formally conducting the hypothesis test and drawing conclusions regarding the randomness of the input sequence.

Practical Case Study: Applying the Test to Sensor Data

To solidify the understanding of the implementation process, let us examine a typical scenario: A data engineer is tasked with monitoring a sequence of ten consecutive readings obtained from an industrial sensor. The primary objective is to confirm that these sensor measurements are purely

random, assuring that there is no underlying systematic drift, mechanical pattern, or measurement bias affecting the output. We will apply the Runs Test to this short sample sequence, employing the default cutoff point (the mean) for initial diagnosis, and meticulously demonstrating the necessary steps in Python.

The initial requirement is to import the specific function from the **statsmodels** library structure. Following the import, we define our sample dataset. It is essential to strictly adhere to standard Python conventions for package imports and variable assignment before executing the statistical function.

The following, precisely structured code block illustrates the exact steps taken to define the data and perform the Runs Test using the `runstest_1samp()` function:

from statsmodels.sandbox.stats.runs import runstest_1samp

#create dataset
data =

#Perform Runs test
runstest_1samp(data, correction=False)

(-0.6708203932499369, 0.5023349543605021)

In this specific computational example, the `correction` parameter was explicitly set to `False`. This deliberate choice is sometimes made in analytical environments for instructional clarity or to ensure direct alignment of results with specific legacy statistical software packages, such as certain implementations within R, which may omit the standard continuity adjustment. However, it is crucial to reiterate that for robust professional analysis, particularly with small samples like N=10, the application of the continuity correction (i.e., setting `correction=True`, the default) is generally recommended practice to improve the Z-approximation accuracy.

Upon execution, the function returns the tuple containing the two crucial statistical values: `(-0.6708203932499369, 0.5023349543605021)`. The first element, approximately -0.67082, represents the calculated Z-statistic, which quantifies the deviation of the observed number of runs from the expected number. The second element, approximately 0.50233, is the resulting two-sided p-value, which dictates the final decision regarding the null hypothesis.

Interpreting Results and Determining Statistical Significance

The process of interpreting the output from the Runs test hinges on comparing the calculated p-value against a pre-established significance level, conventionally symbolized by the Greek letter

alpha ($alpha$). In most scientific and engineering contexts, the standard threshold for determining statistical significance is $alpha = 0.05$ (or 5%). This threshold represents the maximum acceptable probability of incorrectly rejecting the null hypothesis when it is actually true (Type I Error).

The statistical decision rule is applied in a precise and unambiguous manner:

If the p-value computed by the test is strictly less than the chosen $alpha$ (e.g., $p < 0.05$), we possess sufficient statistical evidence to **reject the null hypothesis** (H_0). This outcome definitively indicates that the observed sequence exhibits statistically significant non-randomness.
If the p-value is greater than or equal to $alpha$ ($p geq 0.05$), we **fail to reject the null hypothesis** (H_0). This conclusion suggests that there is insufficient statistical evidence in the data to conclude that the sequence deviates significantly from a purely random process.

Applying this rule to our sensor data case study, we find that the calculated z-test statistic is approximately **-0.67082**, and the corresponding p-value is **0.50233**. Since 0.50233 is substantially larger than the standard significance level of $alpha = 0.05$, the definitive conclusion is to **fail to reject the null hypothesis**. This result provides strong statistical affirmation that the sequence of sensor readings was produced in a manner entirely consistent with a random process, suggesting no issues of serial dependence or systematic pattern in the sequence's order.

The sign of the Z-statistic provides ancillary diagnostic information, even when the result is not significant. A negative Z-statistic, as seen in our example (-0.67082), implies that the observed number of runs was slightly fewer than what was statistically expected under perfect randomness, subtly hinting at a minor degree of data clumping. Conversely, a positive Z-statistic would suggest an observed number of runs higher than expected, indicating a slight tendency toward alternation. However, because our p-value (0.50233) is very high, neither the minor clumping nor alternation observed is considered statistically meaningful or significant.

Limitations and Consideration of Alternative Randomness Tests

While the Runs test remains an invaluable, accessible diagnostic tool for the initial detection of basic patterns of non-randomness, especially concerning serial dependence, practitioners must be cognizant of its inherent limitations. Fundamentally, the test is only sensitive to the order of observations relative to a single, chosen cutoff point (mean or median); it is not designed to test for specific complex distributional properties, nor does it provide a quantitative measure of the strength of the correlation, unlike methods based on autocorrelation functions.

Furthermore, the mathematical foundation of the test relies on the assumption that the dichotomized data is strictly binary (only 'A' or 'B'). If the raw dataset contains numerous values that are exactly equal to the chosen cutoff point, the core assumptions underpinning the test's

theoretical calculation can be violated, potentially leading to compromised accuracy in the final results. The reliability of the standard Z-approximation, which allows us to use the normal distribution for inference, also diminishes notably when the sample sizes are very small, even with the application of the recommended continuity correction.

When analyzing specialized datasets, such as complex financial data, geophysical measurements, or long-term environmental time series, alternative and often more powerful tests may offer a more nuanced and detailed perspective on the independence and randomness of the data sequence:

Ljung-Box Test: This is a cornerstone test in time series analysis, specifically designed to check for the overall lack of fit in a model. It rigorously tests whether the autocorrelations of the standardized residuals, or the raw data itself, are statistically different from zero for a specified number of lags, providing a robust measure of whether the data is independently and identically distributed (IID).
Turning Point Test: Conceptually similar to the Runs Test, this non-parametric method counts the number of times the sequence changes direction--that is, switching from an increasing trend to a decreasing trend, or vice versa. It provides an alternative measure focused purely on the directional order of successive observations.
Difference Sign Test: This simple yet informative test systematically examines the signs of the differences between consecutive observations in the sequence. By focusing on whether $X_{i} - X_{i-1}$ is positive or negative, it offers an efficient and straightforward initial check for the presence of persistent trends or noticeable cyclical patterns within the data stream.

Ultimately, selecting the most appropriate randomness test must be guided by the specific type of non-randomness suspected and the overarching analytical goals of the data project. For a swift, general assessment of simple serial correlation and sequential randomness, the Python implementation of the Runs Test available through the **statsmodels** library remains an extremely accessible, effective, and crucial initial diagnostic tool in any data scientist's toolkit.