# Is there a function in Python to calculate Standardized Residuals?

Authored by

**stats writer**

December 14, 2025

The short answer is no; there is no single, built-in function within the core Python standard library designed specifically for calculating Standardized Residuals. However, this statistical necessity is easily addressed using specialized third-party libraries.

Powerful packages designed for statistical modeling and analysis, such as statsmodels and SciPy, provide comprehensive methods to derive standardized residuals as part of their robust regression analysis outputs. Unlike simple residuals, which only require the difference between observed and predicted values, standardized residuals require additional calculations involving statistical influence metrics, which are handled efficiently by these specialized functions and classes.

Understanding the role of standardized residuals is essential for proper model diagnostics, especially when trying to identify influential data points or potential violations of regression assumptions. This guide will walk through the process of generating these crucial statistics using the industry-standard statsmodels library.

## Understanding Residuals in Regression Analysis

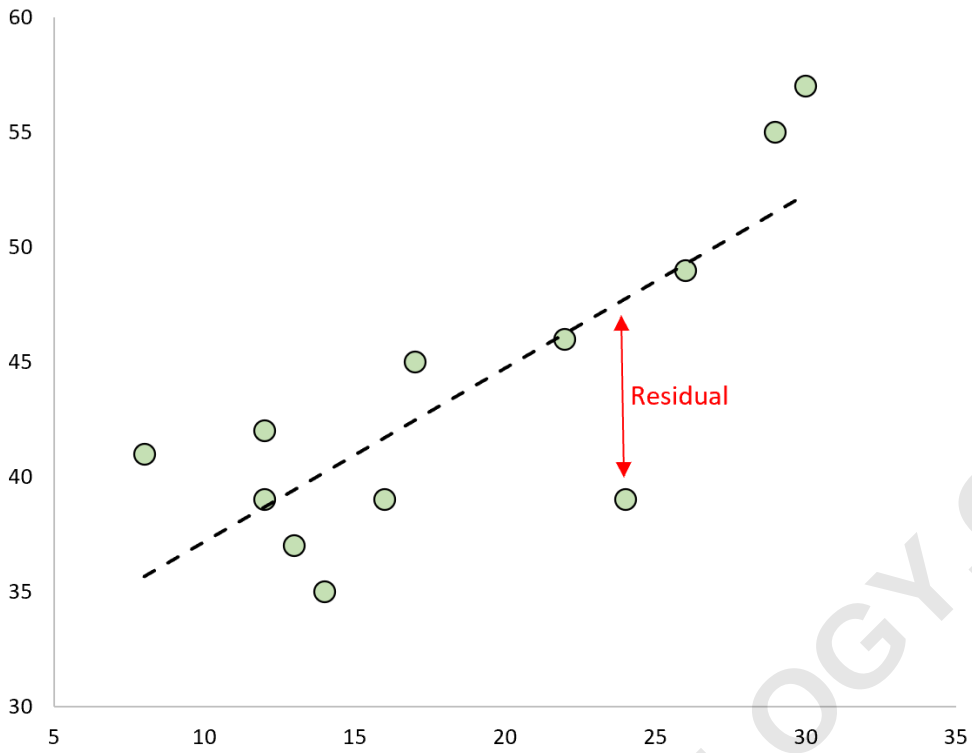
Before diving into standardization, it is crucial to establish a firm understanding of what a simple residual represents within a regression model. A **residual** is fundamentally the difference between an observed value and the value predicted by the fitted regression line for that observation.

Mathematically, the calculation is straightforward:

**Residual (e) = Observed value (y) - Predicted value (?)**

In essence, the residual measures the error in prediction for a specific data point. If the observed value falls exactly on the regression line, the residual is zero. A large positive residual means the model significantly underpredicted the actual outcome, while a large negative residual means the model overpredicted the outcome.

When visualized on a scatterplot where the regression line is overlaid on the data points, the residual for any observation is the vertical distance between that observation and the fitted regression line:






Analyzing simple residuals is the first step in diagnosing model performance. However, because residuals are measured in the original units of the dependent variable (Y), their magnitude can be difficult to compare across different datasets or when the scale of Y is very large, leading us to the need for standardization.

## The Necessity of Standardized Residuals

While simple residuals tell us the magnitude of the error, they do not inherently tell us if that error is statistically significant or unusual relative to the overall spread of the data. This is where **standardized residuals** become indispensable for robust statistical analysis.

Standardized residuals normalize the error by dividing the raw residual by an estimate of its standard deviation. This transformation makes the residuals unitless and allows them to be compared directly to a standard normal distribution (Z-scores). The primary goal of using standardized residuals is to identify observations that are poorly fitted by the model--often referred to as outliers.

Standardization is particularly important because the residuals are not expected to have the same variance across all observations. Observations with high leverage (i.e., data points whose predictor values are far from the mean of the predictors) tend to have smaller residual variances. Standardizing corrects for this non-constant variance, providing a more reliable metric for outlier detection.

## Mathematical Foundation of Standardized Residuals

The calculation for a standardized residual is more complex than a simple Z-score because it accounts for the unique influence and variance of each observation. The standard formula for the internally standardized residual (often interchangeably referred to as the standardized residual in statistical software outputs) is:

**ri = ei / s(ei) = ei / RSE√1-hii**

Where the components are defined as:

**ei:** This represents the ith raw residual (Observed Y minus Predicted Y).
**RSE:** This is the **Residual Standard Error** (or Root Mean Square Error) of the overall regression model, which serves as an overall estimate of the standard deviation of the error term.
**hii**: This critical term represents the **leverage** of the ith observation. Leverage measures how far an observation's predictor value(s) deviate from the mean of the predictor values, influencing the variance of the residual.

The denominator, RSE√1-hii, is essentially the estimated standard deviation of the ith residual. By dividing the residual by its own standard deviation, we effectively measure how many standard deviations away from the regression line that observation lies, thus standardizing the error across all data points.

In applied statistics, a common rule of thumb is to consider any standardized residual with an **absolute value greater than 3** to be an extreme value or a potential outlier that warrants further investigation. Such extreme values may indicate errors in data entry, a deficiency in the model specification, or a truly exceptional observation.

## Step 1: Preparing the Data in Python

The first step in calculating standardized residuals involves setting up the data structure suitable for statistical modeling. We utilize the **pandas** library, the fundamental tool for data manipulation and analysis in Python, to create a DataFrame containing our independent and dependent variables.

We begin by importing the necessary library and defining a small sample dataset. This dataset includes a predictor variable 'x' and a response variable 'y'.

**import pandas as pd**

#create dataset
df = pd.DataFrame({'x': ,

'y': })

The use of a pandas DataFrame ensures that the data is structured correctly with labeled columns, which is essential for integration with the statsmodels library in the subsequent steps. This structure facilitates clear definition of the response and predictor variables necessary for fitting the regression line.

## Step 2: Fitting the Ordinary Least Squares (OLS) Model

Once the data is prepared, we proceed to fit the Ordinary Least Squares (OLS) regression model. We use the powerful API submodule from the statsmodels library, which is designed specifically for deep statistical inference and diagnostics, making it the ideal choice for calculating advanced metrics like standardized residuals.

We first separate the response variable (y) and the explanatory variable (x). A crucial step in fitting a standard linear model using the OLS method in statsmodels is adding a constant term to the predictor variables. This constant represents the intercept of the regression line and ensures that the model includes a bias term.

**import statsmodels.api as sm**

#define response variable
y = df

#define explanatory variable
x = df

#add constant to predictor variables (for the intercept)
x = sm.add_constant(x)

#fit linear regression model using OLS
model = sm.OLS(y, x).fit()

The `model` object now holds the results of the fitted regression, including coefficients, standard errors, and, importantly for our purpose, all the necessary information to calculate diagnostic statistics such as residuals and leverage values.

## Step 3: Calculating Standardized Residuals using statsmodels

The statsmodels library encapsulates the complex calculations required for diagnostic statistics within its fitted model object. To access the standardized residuals, we first need to instantiate the

influence statistics module associated with our fitted OLS model.

We use the built-in method `get_influence()` on the fitted model. This method returns an object that contains various influence and diagnostic metrics. From this influence object, we can extract the standardized residuals using the attribute `resid_studentized_internal`.

```
#create instance of influence object
influence = model.get_influence()

#obtain standardized residuals (internally studentized residuals)
standardized_residuals = influence.resid_studentized_internal

#display standardized residuals
print(standardized_residuals)
```

It is worth noting that statsmodels refers to these as "internally studentized residuals." For most practical purposes in introductory diagnostics, internally studentized residuals are treated synonymously with standardized residuals, as both adjust the residual based on the estimate of its variance (which includes the leverage term). They are a robust measure of unusual deviation from the fitted line.

## Step 4: Interpreting and Identifying Outliers

The output provides a list of standardized residual values corresponding to each observation in the dataset. Since these values follow a distribution that approximates the standard normal distribution, their magnitude directly indicates how unusual the observation is relative to the model's overall fit. We apply the standard statistical threshold for identifying extreme outliers.

As established earlier, an observation is typically flagged as an outlier if the absolute value of its standardized residual is greater than 3. Reviewing the results from our example calculation:

The highest absolute value observed is approximately 2.117 (corresponding to the ninth observation). Since this value is significantly less than 3, we can conclude that, based on the standardized residual analysis, none of the observations in this particular dataset appear to be statistical outliers requiring specific removal or model adjustment.

If an absolute value exceeding 3 were found, it would be crucial to investigate that specific data point. Potential actions could include checking for data entry errors, assessing if the observation belongs to a subpopulation not captured by the model, or determining if it possesses high influence metrics that necessitate a more sophisticated modeling approach (such as weighted least squares or robust regression).

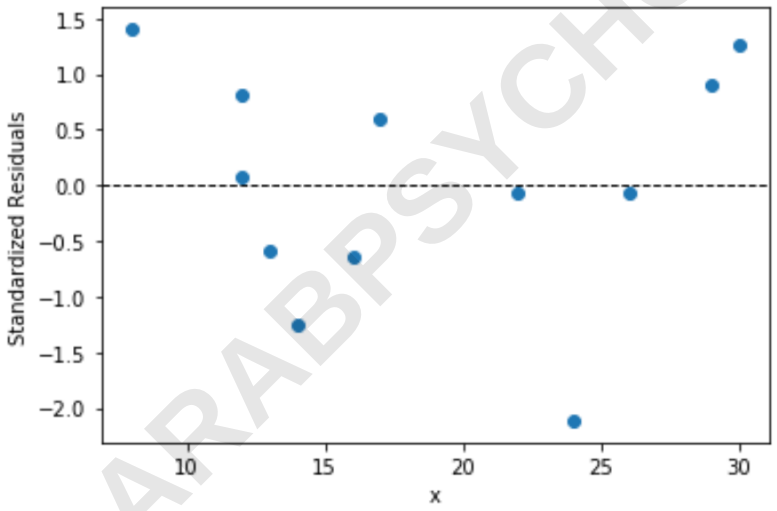
## Step 5: Visualizing Standardized Residuals

While numerical checks are essential, visualizing the standardized residuals offers deeper insight into the model's adherence to key assumptions, particularly the assumption of homoscedasticity (constant variance of errors). We use the matplotlib library to generate a scatterplot of the predictor variable ('x') against the calculated standardized residuals.

**import matplotlib.pyplot as plt**

```
plt.scatter(df.x, standardized_residuals)
plt.xlabel('x')
plt.ylabel('Standardized Residuals')
plt.axhline(y=0, color='black', linestyle='--', linewidth=1)
plt.show()
```

The resulting plot displays the spread of errors across the range of the predictor variable. The horizontal dashed line at Y=0 represents the ideal scenario where all predictions are perfectly accurate.






In a healthy regression model, the standardized residuals should show no discernible pattern when plotted against the predictor variable. They should be scattered randomly above and below the zero line, forming a roughly horizontal band. If the plot exhibits patterns--such as a funnel shape (indicating increasing variance as X increases, known as heteroscedasticity) or a curve--it suggests that the fundamental assumptions of the regression model are violated, potentially requiring data transformation or the use of a non-linear model.

In our example visualization, the points appear randomly distributed around the zero line,

confirming that the assumptions related to the random distribution of errors are reasonably met for this simple dataset.

Further Reading: What Are Standardized Residuals?