How to Create a Lag Function in Google Sheets: A Step-by-Step Guide

Authored by

stats writer

January 15, 2026

To create a **Lag Function** in Google Sheets, you can utilize powerful built-in functions like `OFFSET` or a combination of `INDEX` and `ROW`. These methods allow you to specify the range of cells you want to shift by a certain number of rows or columns. This dynamic referencing capability is essential for retrieving the value from the previous or next row, thereby creating the desired lag effect.

By using the OFFSET function, you can precisely control the shift value, enabling effective analysis of time series data or tracking sequential changes within a dataset. The fundamental principle is to dynamically reference cells based on their position relative to the current cell, allowing for streamlined calculation of period-over-period differences and historical comparisons.

There does not exist a dedicated **LAG** function in Google Sheets to calculate lagged values directly, but you can achieve this necessary analytical capability efficiently by using the **OFFSET** function instead. The `OFFSET` function is the standard and most intuitive method for simulating lag functionality within the spreadsheet environment.

The following comprehensive examples detail how to structure and implement the **OFFSET** function to calculate lagged values in practice, covering both simple sequential data and more complex grouped data scenarios.

Introduction to Lagged Analysis in Google Sheets

Analyzing sequential or temporal data often requires comparing a current observation with a prior one. This technique, known as lagged values analysis, is fundamental in finance, statistical modeling, and operational research. Although dedicated functions like `LAG` are common in database systems or specialized statistical software, the powerful spreadsheet environment of Google Sheets requires a slightly different approach. Specifically, to effectively implement a lag function, we utilize core built-in functions such as `OFFSET` or a combination of `INDEX` and `ROW`.

The core concept behind creating a lag effect is simple: we need a formula that can dynamically look up a cell's value located a specified number of rows (or columns) away from the current cell. This is crucial when analyzing sequential events, determining dependencies between consecutive periods, or preparing data for advanced statistical analysis. Achieving this requires meticulous attention to relative cell referencing, ensuring that the formula shifts correctly as it is applied down a column.

Mastering this technique allows users to transform raw data into actionable insights, particularly when dealing with time series data. Whether you are tracking daily stock prices, monitoring sales trends across weeks, or analyzing sequential experimental results, the ability to effortlessly reference the previous period's data point is invaluable for calculating rates of change, comparing

performance, and identifying patterns that span across multiple time steps.

The Power and Syntax of the OFFSET Function

The OFFSET function is arguably the most straightforward and powerful method for calculating lagged values in Google Sheets. It works by returning a reference to a range that is a specified number of rows and columns away from an initial reference cell. This function is essential for creating dynamic ranges and, in our specific use case, for looking backward (or forward) in sequential data.

The syntax for `OFFSET` is structured as follows: `OFFSET(cell_reference, offset_rows, offset_columns, , )`. For calculating simple lagged values, we focus primarily on the first three arguments:

`cell_reference`: The starting cell (usually the current row's data point).

`offset_rows`: The number of rows to move. Use a negative number (e.g., `-1`) for a lag (looking backward).

`offset_columns`: The number of columns to move. Use `0` to stay within the same column.

By setting the row offset to a negative value corresponding to the desired lag period and the column offset to zero, we precisely isolate the value from the preceding time step. This relative referencing behavior ensures that the derived column, which contains the lagged values, accurately reflects the data from the preceding period for every row.

Example 1: Calculating Simple Lagged Values in Google Sheets

Suppose we have the following dataset in Google Sheets that shows the total sales made by some store in 10 consecutive days. We aim to create a "Lag Sales" column that reflects the sales from the previous day.

	A	B	C	D
1	**Day**	**Sales**		
2	1	13		
3	2	17		
4	3	24		
5	4	29		
6	5	34		
7	6	17		
8	7	15		
9	8	12		
10	9	10		
11	10	22		
12				
13				
14				
15				

In this scenario, the sales data is in column B, starting at cell B3. We will calculate the one-period lag (n=1) starting in cell C3. We need a formula that instructs the sheet to look up one row from the current position.

We can use the following formula to calculate the lagged sales values in a new column:

=OFFSET(B3, -1, 0)

We can type this formula into cell **C3** and drag it down to every remaining cell in column C:

C3 =OFFSET(B3, -1, 0)

	A	B	C	D
1	**Day**	**Sales**	**Lag Sales**	
2	1	13		
3	2	17	13	
4	3	24	17	
5	4	29	24	
6	5	34	29	
7	6	17	34	
8	7	15	17	
9	8	12	15	
10	9	10	12	
11	10	22	10	
12				
13				
14				
15				

The "Lag Sales" column successfully shows the sales for a lag of n=1. For example, on day 2, the store made **17** sales. The lagged value of sales on day 2 (which corresponds to the sales made on day 1) is correctly shown as **13** sales.

Note: To calculate the lagged value for a different number of previous periods (e.g., a two-day lag), simply change the **-1** in the **OFFSET** formula to the corresponding negative number (e.g., `-2`).

Example 2: Calculating Lagged Values by Group in Google Sheets

When dealing with complex datasets that contain multiple groups (such as sales across different stores or regions), it is essential that the lag calculation resets at the start of each new group. If this reset is not performed, the first data point of a new group would incorrectly reference the last data point of the previous group.

Suppose we have the following dataset in Google Sheets that shows the total sales made by two different stores (Store A and Store B) during 5 days each:

	A	B	C	D
1	**Store**	**Sales**		
2	A	13		
3	A	17		
4	A	24		
5	A	29		
6	A	34		
7	B	17		
8	B	15		
9	B	12		
10	B	10		
11	B	22		
12				
13				
14				
15				

To ensure the lag calculation respects the group boundaries defined in column A, we must incorporate an `IF` condition. This condition checks if the current row belongs to the same store as the preceding row before applying the OFFSET function.

We can use the following formula, which combines conditional logic with dynamic referencing, to calculate the lagged sales values by store in a new column:

=IF(A3=A2, OFFSET(B3, -1, 0), "")

C3 =IF(A3=A2, OFFSET(B3, -1, 0), "")

	A	B	C	D
1	**Store**	**Sales**	**Lag Sales**	
2	A	13		
3	A	17	13	
4	A	24	17	
5	A	29	24	
6	A	34	29	
7	B	17		
8	B	15	17	
9	B	12	15	
10	B	10	12	
11	B	22	10	
12				
13				
14				

This function first checks if the store value in the current row (e.g., A3) is equal to the store value in the previous row (A2). This logical test is crucial for segmented time series data.

If the stores match (the condition is TRUE), then it returns the lagged sales value using the `OFFSET` function. If the stores do not match (the condition is FALSE, indicating a new group has started), then it returns a blank (`""`).

For example, in row 3, the sales value was **17**. Since the store value in row 2 ('Store A') is equal to the store value in row 3 ('Store A'), the lagged value of sales is calculated as **13**. However, when the formula is applied to row 8, the value for the store ('Store B') does not match the value for the store in row 7 ('Store A'), so a blank value is returned instead of an incorrect lagged sales value.

Alternative: Using INDEX and ROW for Non-Volatile Lag

While the OFFSET function is easy to understand, an alternative method using `INDEX` and `ROW` is preferred by some users because it is non-volatile, meaning it only recalculates when its input arguments change, potentially leading to better performance in very large sheets.

The non-volatile lag formula structure for a one-period lag is: `=INDEX(B:B, ROW() - 1)`.

This formula works by setting the sales column (B:B) as the range and dynamically calculating the row number that is one period behind the current row (`ROW() - 1`). This method is highly efficient

and equally effective at calculating lagged values.

Summary of Lag Calculation Methods

To finalize your understanding of lag implementation in Google Sheets, here is a summary of the recommended approaches:

Simple Lag: Use `=OFFSET(Current_Cell, -N, 0)`, where N is the lag period (e.g., 1).

Grouped Lag: Use `=IF(Current_Group = Previous_Group, OFFSET(Current_Cell, -N, 0), "")`.

Non-Volatile Lag: Use `=INDEX function(Data_Column, ROW() - N)`.

Other Common Tasks

The following tutorials explain how to perform other common tasks in Google Sheets that often accompany time series analysis:

Tutorial 1: Calculating Moving Averages
Tutorial 2: Performing Rolling Sums
Tutorial 3: Handling Missing Data in Sequences