Learn How to Sum Every Nth Row in Google Sheets: A Step-by-Step Guide

Authored by

Mohammed looti

October 31, 2025

In the realm of **data analysis** and reporting, a frequent challenge arises when analysts need to calculate sums based not on conditional values, but on the physical position of data points within a spreadsheet. Specifically, summing values from every Nth row in a given range is a powerful capability required for tasks such as summarizing periodic financial statements, aggregating experimental results recorded at regular intervals, or analyzing time-series data. Attempting to manually select and sum these values, particularly within large or dynamic datasets, is highly inefficient, time-consuming, and significantly increases the probability of calculation errors.

Fortunately, **Google Sheets** offers a robust and elegant solution by allowing users to combine several specialized functions into a single, comprehensive formula. This technique dynamically identifies rows based purely on their index position and applies the summation only to those specific rows. Mastering this method is crucial for elevating your spreadsheet proficiency and enhancing your ability to manipulate complex data structures efficiently.

The core formula relies on the synergistic interaction of four fundamental **Google Sheets** functions: **SUMIF**, **ARRAYFORMULA**, **MOD**, and **ROW**. Together, these functions execute a precise sequence of operations: first, generating a normalized index for every row; second, applying the mathematical modulo operation to isolate the Nth rows; and finally, restricting the summation to only those identified rows.

=sumif(ArrayFormula(mod((row(A1:A)-row(A1)+1),3)),0,A1:A)

This powerful formula, shown above, is specifically configured to calculate the sum of every **3rd** value within the range of column A, with the counting sequence initiated from cell **A1**. The key advantage of this approach lies in its adaptability; the structure is designed for easy modification. By adjusting parameters such as the interval (N) and the starting **cell reference**, users can tailor the formula to meet diverse analytical requirements, a flexibility we will thoroughly explore through detailed, practical examples.

To effectively illustrate the versatility and technical power of this formula, we will proceed through a series of practical scenarios. Each example utilizes a consistent column of numerical values in **Google Sheets**, ensuring that you can clearly observe how minor, strategic adjustments to the formula's input parameters yield dramatically different and correct summation results based on the specific positional criteria.

Deconstructing the Core Formula for Nth Row Summation

A deep understanding of the composite formula is essential for successful customization. The general structure is: `=SUMIF(ARRAYFORMULA(MOD((ROW(Range)-ROW(StartCell)+1),N)),0,Range)`. This formula operates by establishing a temporary array that

flags the target rows, which are then used by the **SUMIF** function as the criterion for summation. The entire process hinges on accurately generating and evaluating the row index relative to the starting point.

The initial, innermost component, `(ROW(A1:A)-ROW(A1)+1)`, is responsible for creating a normalized, 1-based index that is isolated from the sheet's absolute row numbers. The function **`ROW(A1:A)`** returns an array of the absolute row numbers (e.g., {1; 2; 3; 4; ...}). By subtracting **`ROW(A1)`** (which equals 1), the index is temporarily shifted to start at zero. Finally, adding `1` restores the array to a clean, 1-based sequential index starting at the designated cell (e.g., {1; 2; 3; 4; ...}), making it mathematically straightforward to count every Nth position from the beginning of the selected range.

Next, the **MOD** function introduces the logic for periodicity. The structure `MOD((normalized_row_index), N)` applies the **modulo operator**, which returns the remainder of a division. When we divide the normalized row index by **N** (the desired interval, such as 3 for every third row), the remainder will consistently be 0 only for those rows that are exact multiples of N. For example, if N=3, the resulting array of remainders will follow the pattern {1, 2, 0, 1, 2, 0, ...}, where every 0 precisely identifies the 3rd, 6th, 9th, and subsequent Nth rows within the defined range.

The wrapper function, **ARRAYFORMULA**, is indispensable for this operation. Its purpose is to force the **`ROW`** and **MOD** functions to process the entire specified range (e.g., `A1:A`) as a series of individual values, rather than just acting upon the first cell. Without **ARRAYFORMULA**, the calculation would collapse and fail to generate the required array of remainders. By enabling array processing, it successfully creates the conditional array consisting of 0s (for target rows) and other remainders (for non-target rows).

Finally, the **SUMIF** function provides the final aggregation step, structured as `SUMIF(range, criterion, sum_range)`. In this context, the `range` input is the array of remainders generated by the **ARRAYFORMULA**/**MOD** operation; the `criterion` is fixed as `0`, serving as the definitive flag for Nth rows; and the `sum_range` is the original data range, `A1:A`. **SUMIF** intelligently matches the `0` criteria against the remainder array and sums only the corresponding values in the data column, thereby executing the summation of every Nth row precisely.

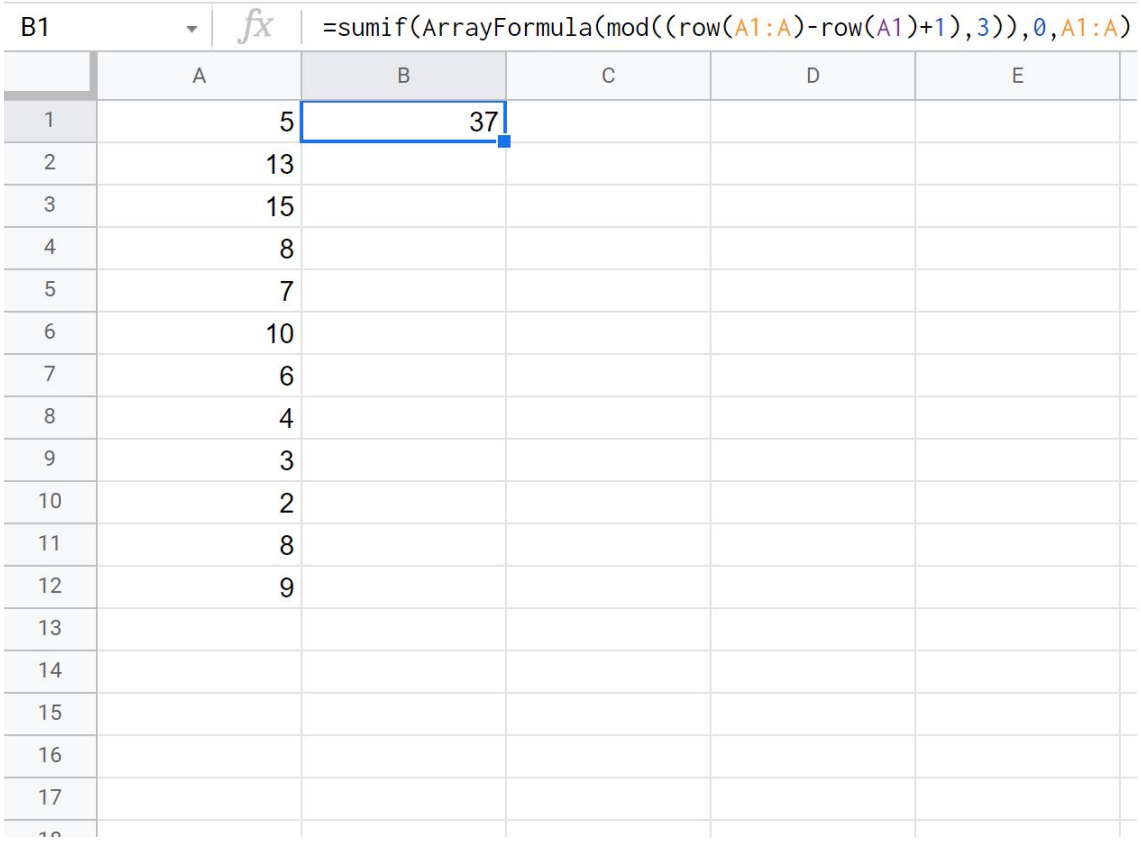
Example 1: Summing Every Third Row from the Initial Cell

We begin with the most common application of this technique: calculating the sum of values located in every third row of a dataset, starting the count directly from the very first cell of the data range. This scenario is frequently encountered when data is sampled or logged with a consistent skip pattern and requires aggregation that respects that interval. The formula employed here demonstrates the fundamental setup for interval-based summation.

To execute this specific task, the core formula is deployed with the 'N' value within the **MOD** function explicitly set to `3`. This configuration instructs the formula to identify and target every third row. Crucially, the range `A1:A` is used consistently throughout the formula, establishing cell **A1** as the absolute starting point for both the indexing and the summation process, continuing down the entirety of column A.

=sumif(ArrayFormula(mod((row(A1:A)-row(A1)+1),3)),0,A1:A)

The accompanying visual aid below illustrates the practical application of this formula within a **Google Sheets** environment. The screenshot clearly displays the formula input in a designated result cell (preferably outside the data column to prevent any calculation conflicts) and the subsequent calculated total. This demonstration provides a transparent view of the formula in action, highlighting which data points are being included in the aggregated sum based on the 'every third row' criterion.



Following the application of the formula, the resulting sum of every third row, beginning at **A1**, is determined to be **37**. This total is a direct consequence of the formula systematically identifying the rows where the normalized row index divided by 3 results in a remainder of zero, and aggregating only the values within those specific cells.

To ensure the absolute accuracy and reliability of our calculation, a crucial manual verification step must be performed. By identifying the rows corresponding to the 3rd, 6th, 9th, 12th, and so on (relative to the start of the range) and manually totaling their respective values, we should confirm the calculated result. The relevant values are found in A3 (15), A6 (10), A9 (3), and A12 (9). Summing these individual values (15 + 10 + 3 + 9) unequivocally confirms a total of **37**, validating the formula's precision.

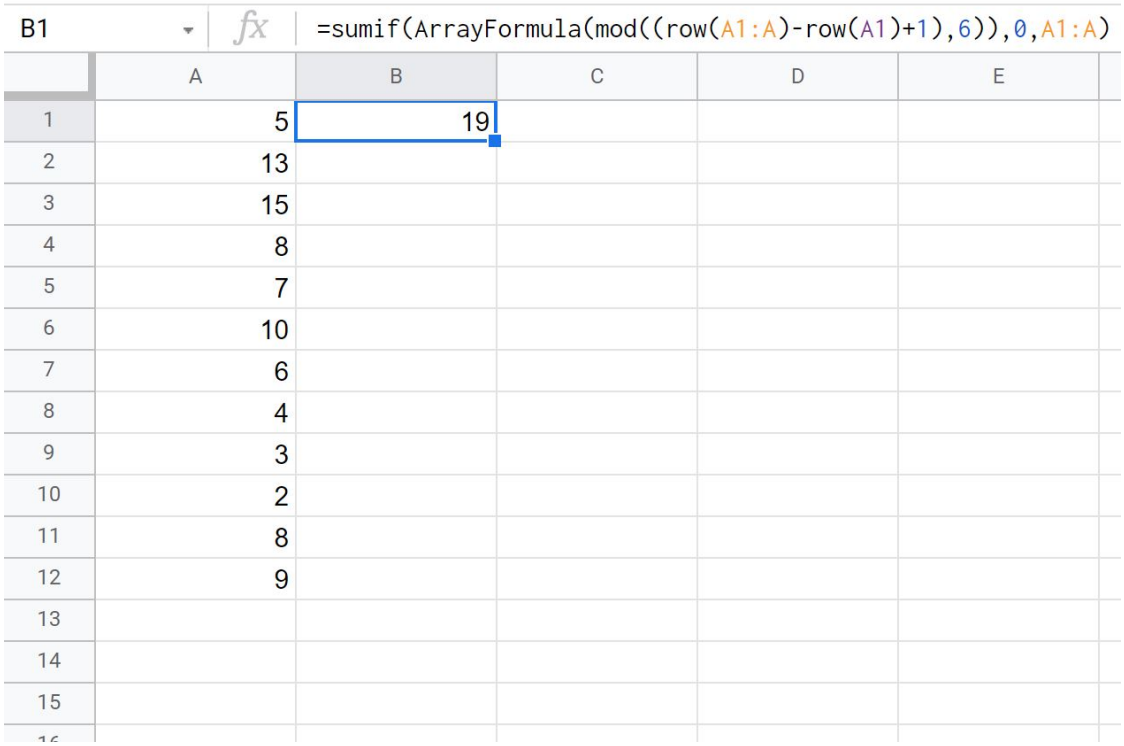
Example 2: Adapting the Formula for Different Nth Intervals

The flexibility of the core formula allows for straightforward adaptation to different periodicities. Extending beyond the previous example, we can easily adjust the formula to sum values based on a broader interval, such as every sixth row. This capability is vital for managing long-term data series where aggregation over larger, less frequent batches is necessary, showcasing the formula's scalability across various data resolutions.

To successfully sum the values in every sixth row, starting from cell **A1**, the only required modification is a simple change to the 'N' parameter within the **MOD** function. We replace the previous value of `3` with `6`. All other structural components, including the range references (`A1:A`), remain identical, affirming that we are still analyzing the entire data column beginning at its first element, but with a different frequency.

=sumif(ArrayFormula(mod((row(A1:A)-row(A1)+1),6)),0,A1:A)

The subsequent screenshot visually demonstrates the effective implementation of this revised formula in **Google Sheets**. The image provides clear evidence of the formula's entry and the subsequent resulting sum, offering immediate visual confirmation that the calculation has successfully executed using the new, less frequent interval (N=6).



Upon processing the data with the formula adjusted for every sixth row, the calculated total sum is found to be **19**. This result logically reflects the aggregation of values from a much sparser set of data points compared to the three-row interval used in Example 1, demonstrating the direct impact of changing the 'N' parameter.

For verification purposes, we identify the values located in every sixth row, starting from the range beginning at A1. The relevant entries correspond to the 6th, 12th, 18th, and subsequent normalized rows within the provided data column. Specifically, the values included in the sum are: 10 (from A6) and 9 (from A12). Summing these two identified values (10 + 9) yields a total of **19**. This manual check firmly establishes the formula's accuracy when successfully configured for a different Nth interval.

Example 3: Modifying the Starting Point for Nth Row Summation

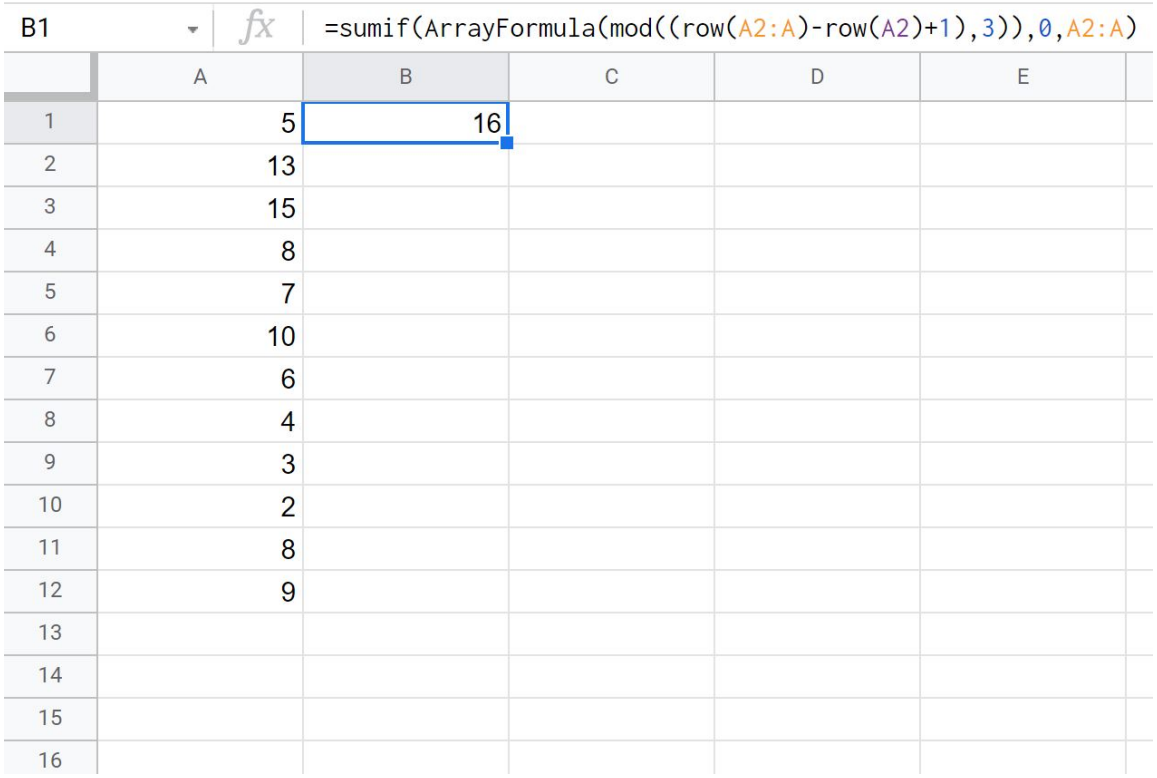
One of the most valuable features of this complex formula is its capacity to define an arbitrary starting point for the summation, rather than being restricted to the absolute first row (A1). This flexibility is critical when dealing with raw data that includes header rows, metadata, or introductory sections that must be excluded from the periodic analysis. In this example, we demonstrate how to sum every third row, but specifically initiating the count from the second row (cell **A2**) of column A.

Implementing a custom starting point requires precise modification of two critical formula parameters. First, the data range referenced by both the **ROW** function and the **SUMIF** function's

`sum_range` must be updated from `A1:A` to `A2:A`. Second, the normalization factor **`ROW(A1)`** must be replaced by **`ROW(A2)`**. This ensures that the normalized index correctly starts at 1 precisely at cell **A2**, treating it as the first item in the new calculation sequence.

=sumif(ArrayFormula(mod((row(A2:A)-row(A2)+1),3)),0,A2:A)

The following screenshot serves as a comprehensive visual reference, demonstrating how this expertly modified formula is entered and executed within **Google Sheets**. The image clearly verifies that the calculation successfully ignores the first row (A1) and correctly begins the summation process from cell **A2**, proceeding to sum every third subsequent row from that new anchor point.



With the formula precisely adjusted to start its calculation from cell **A2** and sum every third row thereafter (N=3), the computed total is found to be **16**. This result powerfully illustrates how altering the starting reference point fundamentally changes which data values are encompassed in the aggregated summation, providing granular control over the analysis scope.

To manually verify this result, we must initiate our count at cell **A2** and select every third row relative to that starting point. Thus, A2 is considered the 1st row, A5 is the 4th row (the 3rd normalized row), A8 is the 7th row (the 6th normalized row), and so on. Based on this adjusted calculation sequence, the values selected are: 8 (from A2), 6 (from A5), and 2 (from A8). Summing

these specific values (8 + 6 + 2) yields a total of **16**. This rigorous manual check confirms the formula's accuracy and robustness in managing custom starting rows.

Conclusion: Mastering Advanced Row Operations in Google Sheets

Mastering the advanced technique of summing every Nth row represents a significant leap forward in your proficiency with **Google Sheets**, enabling highly specific **data analysis** and sophisticated reporting. By strategically combining the power of **SUMIF**, **ARRAYFORMULA**, **MOD**, and **ROW** functions, you gain granular and dynamic control over data aggregation based purely on positional criteria. This method is far superior to manual selection, offering substantial time savings and dramatically reducing the likelihood of critical calculation errors in complex datasets.

The detailed examples provided herein clearly demonstrated the formula's versatility, showcasing how targeted adjustments to the 'N' value (the interval) or the starting **cell reference** allow you to precisely define the scope and frequency of your summation. Whether your objective is to aggregate data sampled at specific intervals or to initiate your analysis from a precise point within a large spreadsheet, this robust composite formula provides a clean, automated, and highly efficient solution.

We strongly encourage users to apply and experiment with these formulas using their own datasets. Developing an intuitive understanding of the underlying mathematical logic--specifically how the **modulo operator** interacts with the row indexing--will empower you to adapt this powerful technique to an extensive range of sophisticated data manipulation challenges, further solidifying your expertise in advanced spreadsheet operations.

Additional Resources

To further deepen your comprehension of **Google Sheets** functionality and explore related data manipulation tasks, we recommend consulting the following tutorials and documentation. These resources cover a wide array of topics that complement the advanced skills discussed in this article, offering valuable further insights into efficient spreadsheet management and operation.