Calculating Standard Deviation in SAS: A Step-by-Step Guide with Examples

Authored by

Mohammed looti

October 31, 2025

Introduction to Standard Deviation in SAS

The calculation of the Standard Deviation (SD) is a cornerstone of statistical analysis, providing essential insight into the variability of a dataset. A higher SD signifies data points that are widely dispersed from the mean, whereas a lower SD indicates data clustering closely around the central average. Mastery of this calculation is critical for data professionals using SAS, a robust software suite designed for advanced analytics.

In the SAS programming environment, the most efficient and standard way to calculate the SD is through the use of the powerful PROC MEANS procedure. This procedure allows analysts unparalleled flexibility, enabling calculations ranging from simple descriptive statistics for a single variable to complex summaries segmented by categorical groups. This guide outlines the three most common applications of the PROC MEANS procedure to determine the standard deviation in various analytical scenarios.

Before diving into the practical examples, it is important to understand the fundamental structure of the commands. The `std` keyword is essential, instructing PROC MEANS to include the standard deviation in the resulting output table. The method you choose--whether specifying a variable, omitting a variable, or including a `CLASS` statement--dictates the scope and aggregation level of the calculation.

Method 1: Calculating Standard Deviation of One Specific Variable

This method is used when the analyst only needs to assess the variability within a single, designated column. By explicitly listing the variable in the `VAR` statement within the PROC MEANS block, you instruct SAS to focus its calculation solely on that dimension of the dataset.

```
proc means data=my_data std;
var variable1;
run;
```

The code above demonstrates the necessary syntax. The `proc means` statement initiates the procedure, specifying the target dataset (`my_data`) and requesting the standard deviation (`std`). Crucially, the `var` statement identifies the specific variable (`variable1`) for which the statistic should be computed. This approach is highly efficient when dealing with large datasets where only a few key metrics are under examination.

Method 2: Calculating Standard Deviation Across All Numeric Variables

If the goal is to obtain descriptive statistics for all quantitative measures contained within the

dataset, SAS provides an elegant shortcut. By simply omitting the `VAR` statement in the PROC MEANS procedure, you instruct the software to automatically calculate the standard deviation for every column classified as a Numeric Variable.

```
proc means data=my_data std;
run;
```

This streamlined syntax is particularly useful during the initial exploratory data analysis phase (EDA), allowing the user to quickly gauge the spread of all quantifiable metrics simultaneously. It saves time by eliminating the need to manually list dozens of variables, ensuring comprehensive coverage of all numerical data points.

Method 3: Calculating Standard Deviation by Group

Often, data analysts need to understand the variability of a metric not for the entire population, but within distinct subgroups. For instance, calculating the average performance score for "Team A" versus "Team B." In SAS, this is achieved using the `CLASS` statement within the PROC MEANS procedure.

```
proc means data=my_data std;
class grouping_variable;
var values_variable;
run;
```

The inclusion of the `CLASS` statement (e.g., specifying a grouping column like `team` or `region`) instructs SAS to partition the dataset based on the unique values in that classification variable. The subsequent standard deviation calculation specified by the `VAR` statement is then performed independently for each identified group. This results in a summary table that clearly contrasts the variability between subgroups, which is invaluable for comparative analysis.

Demonstration Dataset and Setup

To illustrate these three methods practically, we will utilize a sample dataset containing athletic performance metrics. This dataset, named `my_data`, includes variables for team assignment, points scored, and rebounds collected. The following SAS code creates and displays the data structure we will be analyzing:

```
/*create dataset*/
data my_data;
input team $ points rebounds;
```

```
datalines;
A 23 6
A 31 5
A 33 5
A 18 8
A 20 9
A 25 12
B 18 10
B 20 7
B 17 8
B 14 3
B 14 3
B 15 6
;
run;

/*view dataset*/
proc print data=my_data;
```

As confirmed by the output below, our `my_data` table is successfully loaded into the SAS environment, ready for statistical processing. Note that `team` is a character variable (indicated by the dollar sign in the input statement), while `points` and `rebound` are the Numeric Variables we will analyze for variability.

Obs	team	points	rebounds
1	A	23	6
2	A	31	5
3	A	33	5
4	A	18	8
5	A	20	9
6	A	25	12
7	B	18	10
8	B	20	7
9	B	17	8
10	B	14	3
11	B	14	3
12	B	15	6

Example 1: Calculating Standard Deviation of One Variable

We begin by applying Method 1 to determine the variability solely for the `points` variable across the entire sample of observations. This calculation will provide a single Standard Deviation value, reflecting the average spread of points scored relative to the overall mean score.

```
proc means data=my_data std;
var points;
run;
```

Executing this SAS code yields a summary table that clearly isolates the requested statistic. The output confirms that the standard deviation of the `points` variable, based on all 12 observations, is **6.2716**. This value serves as our benchmark for overall scoring consistency.

The MEANS Procedure

Analysis Variable : points
Std Dev
6.2716292

A standard deviation of **6.2716** suggests a moderate level of dispersion in the scores. If this value

were significantly lower (e.g., 1 or 2), it would imply that most players scored very close to the average. Conversely, a much higher value would indicate a large variance, suggesting the presence of both extremely low and extremely high scores within the dataset.

Example 2: Calculating Standard Deviation of All Numeric Variables

Next, we employ Method 2 to simultaneously calculate the Standard Deviation for all Numeric Variables in the `my_data` dataset: `points` and `rebound`. By skipping the `VAR` statement, we utilize the efficiency built into the PROC MEANS procedure for broad descriptive analysis.

```
proc means data=my_data std;
run;
```

The resulting output clearly displays the standard deviation for both quantitative measures side-by-side. The standard deviation for `points` is again confirmed at **6.2716**, while the standard deviation for `rebound` is calculated as **2.7247**.

The MEANS Procedure

Variable	Std Dev
points	6.2716292
rebounds	2.7247463

Comparing these two values provides immediate, actionable insight: since the standard deviation for `points` (6.2716) is substantially larger than that for `rebound` (2.7247), we conclude that the point totals are significantly more spread out or volatile than the rebound totals. This suggests a greater consistency in the number of rebounds collected than in the scores achieved across the observed players.

Example 3: Calculating Standard Deviation by Group

Finally, we implement Method 3 to perform a segmented analysis. We want to compare the scoring consistency (variability of `points`) between the two distinct groups defined by the `team` variable. This requires the use of the `CLASS` statement, allowing us to perform subgroup analysis.

```
proc means data=my_data std;
class team;
var points;
run;
```

The output is now grouped by the values found in the `team` column (A and B). This provides two separate standard deviation calculations for the `points` variable, one for each team. The Standard Deviation of points for Team A is **5.9665**, and the standard deviation of points for Team B is **2.4221**.

The MEANS Procedure

Analysis Variable : points		
team	N Obs	Std Dev
A	6	5.9665736
B	6	2.4221203

This group-level analysis reveals a crucial difference in performance consistency. Team B exhibits a much lower standard deviation (2.4221) than Team A (5.9665), indicating that the players on Team B score points much more consistently, clustering tightly around their team's average. Conversely, Team A shows higher volatility in scoring, suggesting a greater mix of high- and low-scoring individual performances.

Summary and Conclusion

The PROC MEANS procedure in SAS offers straightforward and powerful capabilities for calculating the Standard Deviation. Whether performing a simple calculation for a single column, conducting a comprehensive review of all Numeric Variables, or executing a detailed analysis segmented by categorical groups, the syntax remains intuitive and highly adaptable to complex analytical requirements. Understanding these three methods ensures that data professionals can quickly and accurately quantify data dispersion, leading to more robust statistical conclusions.

We have demonstrated that the choice between these methods depends entirely on the analytical objective. If the goal is comparison across different metrics, using the default behavior (omitting the `VAR` statement) is ideal. If the focus is on identifying differences in variability between populations, incorporating the `CLASS` statement is mandatory. These techniques form the foundational steps for any detailed variance analysis performed within the SAS environment.

Additional Resources for SAS Programming

The following tutorials explain how to perform other common statistical and data manipulation tasks in SAS:

How to calculate descriptive statistics using PROC MEANS.

Methods for filtering and subsetting data using the `WHERE` statement.
Techniques for merging and combining multiple datasets using SAS procedures.