

How to Calculate Percentiles in R: A Step-by-Step Guide

Authored by
stats writer

March 4, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Calculate Percentiles in R: A Step-by-Step Guide*.

PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=133834>

In order to easily calculate percentiles in R, one can use the "quantile" function, which takes in a numeric vector and a probability value as inputs. This function returns the value at the specified percentile. For example, to calculate the 75th percentile of a dataset named "data", one can use the code "quantile(data, 0.75)". This will return the value at which 75% of the data falls below. Other examples of calculating percentiles in R include finding the median, which is the 50th percentile, and the quartiles, which are the 25th and 75th percentiles. By using the "quantile" function, one can easily and accurately calculate any percentile needed in their analysis.

Easily Calculate Percentiles in R (With Examples)

The n th percentile of a dataset is the value that cuts off the first n percent of the data values when all of the values are sorted from least to greatest.

For example, the 90th percentile of a dataset is the value that cuts off the bottom 90% of the data values from the top 10% of data values.

One of the most commonly used percentiles is the 50th percentile, which represents the median value of a dataset: this is the value at which 50% of all data values fall below.

Percentiles can be used to answer questions such as:

What score does a student need to earn on a particular test to be in the top 10% of scores? To answer this, we would find the 90th percentile of all scores, which is the value that separates the bottom 90% of values from the

top 10%.

What heights encompass the middle 50% of heights for students at a particular school? To answer this, we would find the 75th percentile of heights and 25th percentile of heights, which are the two values that determine the upper and lower bounds for the middle 50% of heights.

How to Calculate Percentiles in R

We can easily calculate percentiles in R using the `quantile()` function, which uses the following syntax:

```
quantile(x, probs = seq(0, 1, 0.25))
```

x: a numeric vector whose percentiles we wish to find

probs: a numeric vector of probabilities in that represent the percentiles we wish to find

Finding Percentiles of a Vector

The following code illustrates how to find various percentiles for a given vector in R:

```
#create vector of 100 random values uniformly distributed between 0 and 500
```

```
data <- runif(100, 0, 500)
```

```
#Find the quartiles (25th, 50th, and 75th percentiles) of  
the vector
```

```
quantile(data, probs = c(.25, .5, .75))
```

```
# 25% 50% 75%
```

```
# 97.78961 225.07593 356.47943
```

```
#Find the deciles (10th, 20th, 30th, ..., 90th percentiles)  
of the vector
```

```
quantile(data, probs = seq(.1, .9, by = .1))
```

```
# 10% 20% 30% 40% 50% 60% 70% 80%
```

```
# 45.92510 87.16659 129.49574 178.27989 225.07593  
300.79690 337.84393 386.36108
```

```
# 90%
```

```
#423.28070
```

```
#Find the 37th, 53rd, and 87th percentiles
```

```
quantile(data, probs = c(.37, .53, .87))
```

```
# 37% 53% 87%
```

```
#159.9561 239.8420 418.4787
```

Finding Percentiles of a Data Frame Column

To illustrate how to find the percentiles of a specific data frame column, we'll use the built-in dataset *iris*:

```
#view first six rows of iris dataset
```

```
head(iris)
```

```
Sepal.Length Sepal.Width Petal.Length Petal.Width  
Species
```

```
1 5.1 3.5 1.4 0.2 setosa
```

```
2 4.9 3.0 1.4 0.2 setosa
```

```
3 4.7 3.2 1.3 0.2 setosa
```

```
4 4.6 3.1 1.5 0.2 setosa
```

```
5 5.0 3.6 1.4 0.2 setosa
```

```
6 5.4 3.9 1.7 0.4 setosa
```

The following code shows how to find the 90th percentile value for the column *Sepal.Length*:

```
quantile(iris$Sepal.Length, probs = 0.9)
```

```
#90%
```

```
#6.9
```

Finding Percentiles of Several Data Frame Columns

We can also find percentiles for several columns at once using the `apply()` function:

```
#define columns we want to find percentiles for  
small_iris<- iris
```

```
#use apply() function to find 90th percentile for every  
column  
apply(small_iris, 2, function(x) quantile(x, probs  
= .9))
```

```
#Sepal.Length Sepal.Width Petal.Length Petal.Width  
# 6.90 3.61 5.80 2.20
```

Finding Percentiles by Group

We can also find percentiles by group in R using the `group_by()` function from the `dplyr` library.

The following code illustrates how to find the 90th percentile of *Sepal.Length* for each of the three species in the iris dataset:

```
#load dplyr library  
library(dplyr)
```

#find 90th percentile of *Sepal.Length* for each of the three species

```
iris %>%
```

```
group_by(Species) %>%
```

```
summarise(percent90 = quantile(Sepal.Length, probs = .9))
```

```
# A tibble: 3 x 2
```

```
# Species percent90
```

```
#
```

```
#1 setosa 5.41
```

```
#2 versicolor 6.7
```

```
#3 virginica 7.61
```

The following code illustrates how to find the 90th percentile for all of the variables by Species:

```
iris %>%
```

```
group_by(Species) %>%
```

```
summarise(percent90_SL = quantile(Sepal.Length, probs = .9),
```

```
percent90_SW = quantile(Sepal.Width, probs = .9),
```

```
percent90_PL = quantile(Petal.Length, probs = .9),
```

```
percent90_PW = quantile(Petal.Width, probs = .9))
```

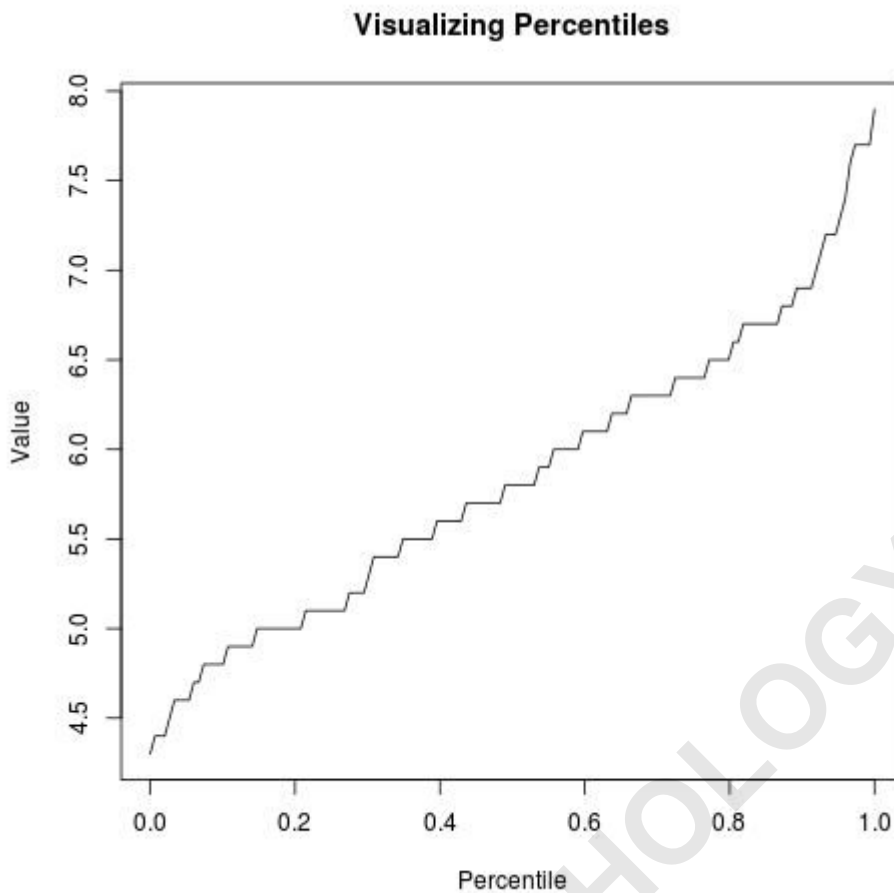
```
# A tibble: 3 x 5
# Species percent90_SL percent90_SW percent90_PL
percent90_PW
#
#1 setosa 5.41 3.9 1.7 0.4
#2 versicolor 6.7 3.11 4.8 1.51
#3 virginica 7.61 3.31 6.31 2.4
```

Visualizing Percentiles

There is no built-in function to visualize the percentiles of a dataset in R, but we can create a plot to visualize the percentiles relatively easily.

The following code illustrates how to create a plot of the percentiles for the data values of *Sepal.Length* from the *iris* dataset:

```
n = length(iris$Sepal.Length)
plot((1:n - 1)/(n - 1), sort(iris$Sepal.Length), type="l",
main = "Visualizing Percentiles",
xlab = "Percentile",
ylab = "Value")
```



A Guide to apply(), lapply(), sapply(), and tapply() in R
Create New Variables in R with mutate() and
case when()