

How does R handle missing values?

Authored by
stats writer

June 30, 2024

RECOMMENDED CITATION

stats writer (2024). *How does R handle missing values?*. PSYCHOLOGICAL SCALES.
Retrieved from <https://scales.arabpsychology.com/?p=161387>

R is a programming language and software environment commonly used for statistical analysis and data visualization. One key feature of R is its ability to handle missing values in datasets. Missing values, also known as NA (Not Available) values, can occur due to various reasons such as human error, data collection issues, or incomplete information.

In order to deal with missing values, R has a built-in mechanism that treats NA values as a separate entity in data analysis. This allows for more accurate and efficient handling of missing values compared to other programming languages.

R provides various functions and tools to handle missing values, such as the "is.na()" function which allows users to identify and flag NA values in their datasets. Additionally, R has functions like "na.omit()" and "na.exclude()" which can be used to remove or exclude NA values from data analysis.

Another approach used by R to handle missing values is through imputation, where missing values are replaced with estimated values based on the available data. R offers different imputation methods, such as mean, median, and regression imputation, which can be chosen based on the type of data and the desired outcome.

Overall, R's robust handling of missing values allows for more accurate and reliable data analysis, making it a popular choice among data scientists and researchers.

How does R handle missing values? | R FAQ

```
Version info: Code for this page was tested in R Under development (unstable)
(2012-02-22 r58461)
```

```
On: 2012-03-28
```

```
With: knitr 0.4
```

Like other statistical software packages, R is capable of handling missing values. However, to those accustomed to working with missing values in other packages, the way in which R handles missing values may require a shift in thinking. On this page, we will present first the basics of how missing values are

represented in R. Next, for those coming from SAS, SPSS, and/or Stata, we will outline some of the differences between missing values in R and missing values elsewhere. Finally, we will introduce some of the tools for working with missing values in R, both in data management and analysis.

Very basics

Missing data in R appears as NA. NA is not a string or a numeric value, but an indicator of missingness. We can create vectors with missing values.

```
x1<-c(1,4,3,NA,7)x2<-c("a","B",NA,"NA")
```

NA is the one of the few non-numbers that we could include in x1 without generating an error (and the other exceptions are letters representing numbers or numeric ideas like infinity). In x2, the third value is missing while the fourth value is the character string "NA". To see which values in each of these vectors R recognizes as missing, we can use the is.na function. It will return a

TRUE/FALSE

vector with as any elements as the vector we provide.

```
is.na(x1)
```

```
## FALSE FALSE FALSE TRUE FALSE
```

```
is.na(x2)
```

```
## FALSE FALSE TRUE FALSE
```

We can see that R distinguishes between the NA and "NA" in x2-NA is seen as a missing value, "NA" is not.

Differences from other packages

```
data test;
```

```
input x y;
```

```
datalines;
```

```
2 .
```

```
3 4
```

```
5 1
```

```
6 0
```

```
;
```

```
data test; set test;
```

```
lowy = (y < 0);
```

```
missy = (y = .);
```

```
run;
```

```
proc print data = test; run;
```

```
Obs x y lowy missy
```

```
1 2 . 1 1
```

```
2 3 4 0 0
```

```
3 5 1 0 0
```

```
4 6 0 0 0
```

We can try the equivalent in R.

```
x1<0
```

```
## FALSE FALSE FALSE NA FALSE
```

```
x1 == NA
```

```
## NA NA NA NA NA
```

Our missing value cannot be compared to 0 and none of

our values can be compared to NA because NA is not assigned a value-it simply is or it isn't.

```
is.na(x1)<-which(x1 == 7)x1
```

```
## 1 4 3 NA NA
```

NA options in R

We have introduced `is.na` as a tool for both finding and creating missing values. It is one of several functions built around NA. Most of the other functions for NA are options for `na.action`.

Just as there are default settings for functions, there are similar underlying defaults for R as a software. You can view these current settings with `options()`. One of these is the "na.action" that describes how missing values should be treated. The possible `na.action` settings within R include:

To see the `na.action` currently in in options, use

`getOption("na.action").`

We can create a data frame with missing values and see how it is treated with each of the above.

```
(g<-as.data.frame(matrix(c(1:5,NA),ncol=2)))
```

```
## V1 V2
```

```
## 1 1 4
```

```
## 2 2 5
```

```
## 3 3 NA
```

```
na.omit(g)
```

```
## V1 V2
```

```
## 1 1 4
```

```
## 2 2 5
```

```
na.exclude(g)
```

```
## V1 V2
```

```
## 1 1 4
```

```
## 2 2 5
```

```
na.fail(g)
```

```
## Error in na.fail.default(g) : missing values in object
```

```
na.pass(g)
```

```
## V1 V2
```

```
## 1 1 4
```

```
## 2 2 5
```

```
## 3 3 NA
```

Missing values in analysis

In some R functions, one of the arguments the user can provide is the `na.action`. For example, if you look at the help for the `lm` command, you can see that `na.action` is one of the listed arguments. By default, it will use the `na.action` specified in the R options. If you wish to use a different `na.action` for the regression, you can indicate the action in the `lm` command.

Two common options with `lm` are the default, `na.omit` and `na.exclude` which does not use the missing values,

but maintains their position for the residuals and fitted values.

```
## use the famous anscombe data and set a few to NA
anscombe<-within(anscombe,{y1<-NA})anscombe#
view
```

```
## x1 x2 x3 x4 y1 y2 y3 y4
## 1 10 10 10 8 NA 9.14 7.46 6.58
## 2 8 8 8 8 NA 8.14 6.77 5.76
## 3 13 13 13 8 NA 8.74 12.74 7.71
## 4 9 9 9 8 8.81 8.77 7.11 8.84
## 5 11 11 11 8 8.33 9.26 7.81 8.47
## 6 14 14 14 8 9.96 8.10 8.84 7.04
## 7 6 6 6 8 7.24 6.13 6.08 5.25
## 8 4 4 4 19 4.26 3.10 5.39 12.50
## 9 12 12 12 8 10.84 9.13 8.15 5.56
## 10 7 7 7 8 4.82 7.26 6.42 7.91
## 11 5 5 5 8 5.68 4.74 5.73 6.89
```

```
model.omit<-
lm(y2~y1,data=anscombe,na.action=na.omit)model.exclude<-lm(y2~y1,data=anscombe,na.action=na.exclude)##
compare effects on residualsresid(model.omit)
```

```
## 4 5 6 7 8 9 10 11
```

```
## 0.727 1.575 -0.799 -0.743 -1.553 -0.425 2.190 -0.971
```

```
resid(model.exclude)
```

```
## 1 2 3 4 5 6 7 8 9 10
```

```
## NA NA NA 0.727 1.575 -0.799 -0.743 -1.553 -0.425  
2.190
```

```
## 11
```

```
## -0.971
```

```
## compare effects on fitted (predicted)  
valuesfitted(model.omit)
```

```
## 4 5 6 7 8 9 10 11
```

```
## 8.04 7.69 8.90 6.87 4.65 9.55 5.07 5.71
```

```
fitted(model.exclude)
```

```
## 1 2 3 4 5 6 7 8 9 10 11
```

```
## NA NA NA 8.04 7.69 8.90 6.87 4.65 9.55 5.07 5.71
```

Using `na.exclude` pads the residuals and fitted values with NAs where there were missing values. Other functions do not use the `na.action`, but instead have a different argument (with some default) for how they will

handle missing values. For example, the `mean` command will, by default, return `NA` if there are any `NA`s in the passed object.

```
mean(x1)
```

```
## NA
```

If you wish to calculate the mean of the non-missing values in the passed object, you can indicate this in the `na.rm` argument (which is, by default, set to `FALSE`).

```
mean(x1,na.rm=TRUE)
```

```
## 2.67
```

Two common commands used in data management and exploration are `summary` and `table`. The `summary` command (when used with numeric vectors) returns the number of `NA`s in a vector, but the `table` command ignores `NA`s by default.

```
summary(x1)
```

```
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
```

```
## 1.00 2.00 3.00 2.67 3.50 4.00 2
```

```
table(x1)
```

```
## x1
```

```
## 1 3 4
```

```
## 1 1 1
```

To see NA among the table output, you can indicate "ifany" or "always" in the useNA argument. The first will show NA in the output only if there is some missing data in the object. The second will include NA in the output regardless.

```
table(x1,useNA="ifany")
```

```
## x1
```

```
## 1 3 4
```

```
## 1 1 1 2
```

```
table(1:3,useNA="always")
```

```
##
```

```
## 1 2 3
```

```
## 1 1 1 0
```

Sorting data containing missing values in R is again different from other packages because NA cannot be compared to other values. By default, `sort` removes any NA values and can therefore change the length of a vector.

```
(x1s<-sort(x1))
```

```
## 1 3 4
```

```
length(x1s)
```

```
## 3
```

The user can specify if NA should be last or first in a sorted order by indicating TRUE or FALSE for the `na.last` argument.

```
sort(x1,na.last=TRUE)
```

1 3 4 NA NA

No matter the goal of your R code, it is wise to both investigate

missing values in your data and use the help files for all functions you use.

You should be either aware of and comfortable with the default treatments of

missing values or specifying the treatment of missing values you want for

your analysis.

ARABPSYCHOLOGY.COM