MSIS 5223: Tutorial 3 – Descriptive Statistics in R

If you have Crawley's book *The R Book* you should read chapters 5 and 8.

Instructions

The purpose of this tutorial is to help you become familiar with using R. Due to its complexity this assignment will focus on performing descriptive statistics. This is an important step in familiarizing yourself with your data so you can better determine what type of analysis to perform later on. For this example, use the ozone.data.txt file.

To follow along with the examples used in this document, use the R script file *Data Mining - Data Derivation, Selection, ODBC.*

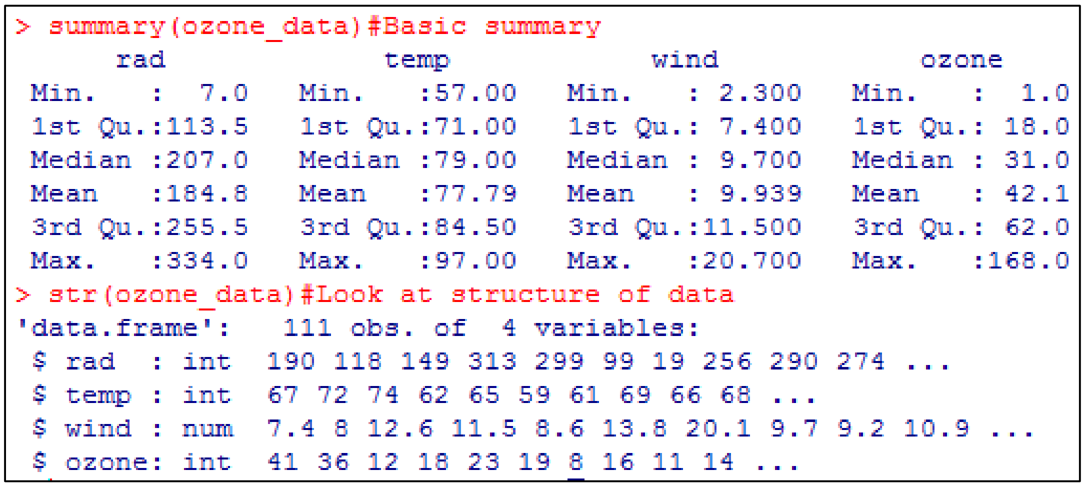
1. Summarizing the Data

After obtaining any data, the first thing you should do is familiarize yourself with it. This allows you to make a more informed decision as to the kinds of statistical models you can build with your data. Often, this merely entails looking at the mean, median, spread, shape, and data types within your data.

There are many ways to assess the basic descriptive information of the data. R comes with some built-in functions that are adequate:

- summary(*data*)
- str(*data*)

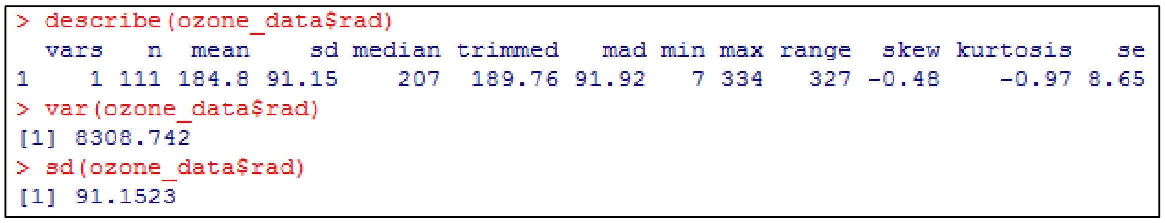
For example, if I was using the ozone.data.txt file and wanted to look at the basic descriptive information, I would type the following and receive the following output:



```
> summary(ozone_data)#Basic summary
      rad              temp             wind              ozone
 Min.   :  7.0   Min.   :57.00   Min.   : 2.300   Min.   :  1.0
 1st Qu.:113.5   1st Qu.:71.00   1st Qu.: 7.400   1st Qu.: 18.0
 Median :207.0   Median :79.00   Median : 9.700   Median : 31.0
 Mean   :184.8   Mean   :77.79   Mean   : 9.939   Mean   : 42.1
 3rd Qu.:255.5   3rd Qu.:84.50   3rd Qu.:11.500   3rd Qu.: 62.0
 Max.   :334.0   Max.   :97.00   Max.   :20.700   Max.   :168.0
> str(ozone_data)#Look at structure of data
'data.frame':   111 obs. of  4 variables:
 $ rad  : int  190 118 149 313 299 99 19 256 290 274 ...
 $ temp : int  67 72 74 62 65 59 61 69 66 68 ...
 $ wind : num  7.4 8 12.6 11.5 8.6 13.8 20.1 9.7 9.2 10.9 ...
 $ ozone: int  41 36 12 18 23 19 8 16 11 14 ...
```

Pay particular attention to the second function that examines the structure of the data. You can see that, in addition to the first few data points for each variable, it provides the data type as well. Three of the variables (*rad*, *temp*, and *ozone*) are integers while the fourth one, *wind*, is numeric (i.e. has decimals). Another important piece of information is the sample size, or the number of observations. For this data set, the sample size is 111.

The library *psych* also provides many additional functions that can help with assessing your data. An important one is the function describe(*y*). First, run the code library(psych) to load it into memory. Second, run the describe(y) function:



```
> describe(ozone_data$rad)
  vars   n  mean    sd median trimmed   mad min max range  skew kurtosis   se
1    1 111 184.8 91.15    207  189.76 91.92   7 334   327 -0.48    -0.97 8.65
> var(ozone_data$rad)
[1] 8308.742
> sd(ozone_data$rad)
[1] 91.1523
```

This function provides the sample size *n*, mean, standard deviation, median, skew, kurtosis, etc. You can compare this with the function summary(*y*) and see the results are the same.

After finding the mean, median, minimum, and maximum you should determine the variance and standard deviation. If you recall, these values provide an idea as to the spread of the data about the mean. In other words, how the data spread out to the left and right of the mean. You can use the function var(*x*) and sd(*x*) (where *x* is your variable) to calculate the variance and standard deviation, respectively. See the figure above for an example of the usage.

2. Using Plots

Often, numbers by themselves are not intuitive. Human brains are designed to interpret visual objects more readily than numerical data. Thus, it is important to create basic plots to assess your data in addition to looking at numbers. This includes simple scatter plots, box plots, or histograms.

One of the most basic plots is an Index Plot. This takes the data as it is ordered in the file and presents them on an x-y plane where the y-value is each data point’s value and x is the position, in order, the data point is. For example, say you have the variable *shoe_size* for 5 people as shown:

Name	Shoe Size
Grace McKenzie	6
Sankar Reddy	7.5
Peter Zhang	11
David Wilkinson	12
Amy Johnson	8.5

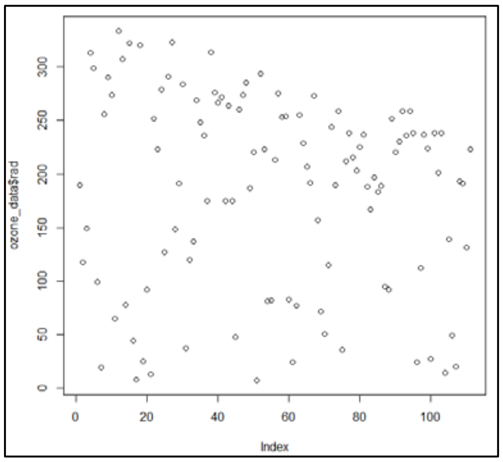
The Index Plot would create a plot giving the following x-value to each of the individuals in the data:

- Grace: 1
- Sankar: 2
- Peter: 3
- David: 4
- Amy: 5

As you can see, the x-values are merely just an ordering of the data as it appears in the data file. If Grace had been listed third and Peter first, then their assigned index values would have been swapped so that Grace was given 3 and Peter 1. What this means is that the index value provides very little information and has no meaning beyond the ordering of your data. Keep in mind, sometimes data is time-ordered, such as transaction-based data; this would indicate the ordering is time-based and sequential.

Below is an example of an Index Plot using the variable *radiation* from the ozone.data.txt file by typing the following:

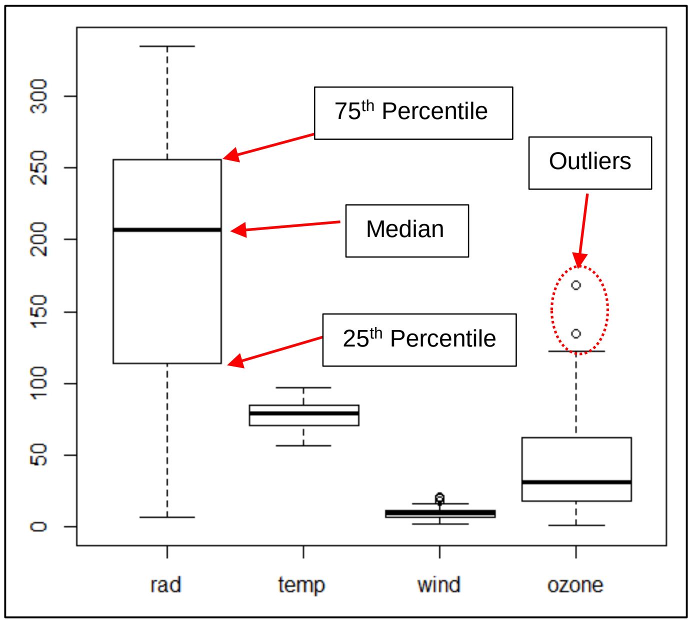
```
plot(ozone_data$rad)
```



To create such a plot, use the function plot(*x*) where *x* is the variable. It should be noted that several variations exist using this function:

- plot(*x*): creates an Index Plot of *x*
- plot(*x, y*): if *x* is continuous, this provides a scatterplot; if *x* is a categorical (i.e. factor) variable, then it provides a box-and-whisker plot

Another type of plot that you should use is the boxplot. While you can use the function plot(*x, y*) to obtain the boxplot for categorical data, a different function is needed for continuous data. Using the ozone data as an example, typing in boxplot(*ozone_data*) results in a boxplot for each variable:

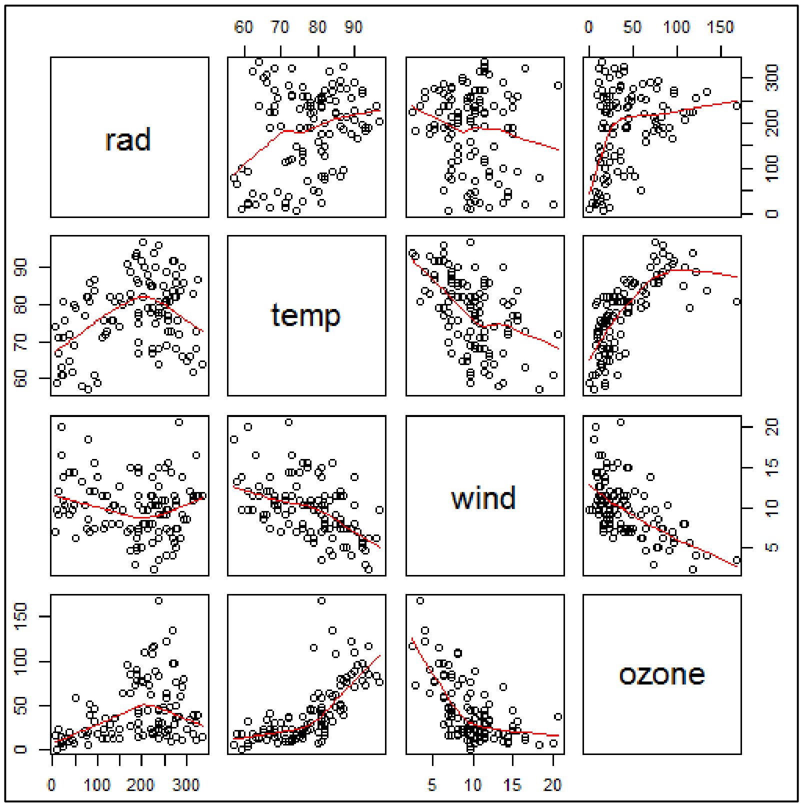


The bold, horizontal line in the middle of each box is the median value for each variable. The top of the box is the 75th percentile while the bottom of the box is the 25th percentile. The dashed lines are the whiskers, leading from the 25th or 75th percentiles to the minimum or maximum data point, respectively. Sometimes, however, that horizontal line does not refer to the minimum or maximum value; they can represent 1.5 times the interquartile range of the data (approximately 2 standard deviations).

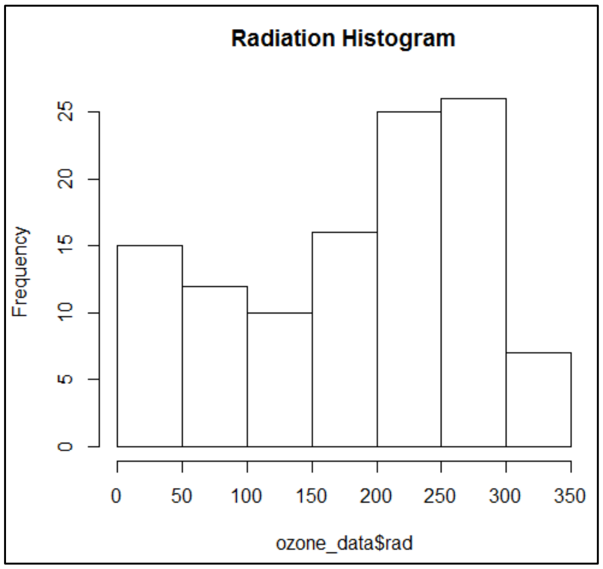
For both *wind* and *ozone* there are data points above the horizontal bar. These are considered outliers in the data. These are data points that are either greater than 1.5 times the interquartile range or lower than 1.5 times the interquartile range.

If your data set contains more than one or two variables, it may take too much time to create individual scatter plots. One function that allows for assessing the relationship of all variables at the same time is pairs(*data*) where *data* is your dataframe object (i.e. your data file). This creates a plot of all variables in your data. Using the ozone data as an example, the function can be used as follows with the resulting plot:

pairs(*ozone_data*, panel=panel.smotth)



The argument "panel=panel.smooth" adds a non-parametric smoother. The target variable is in the row and the explanatory variable is in the column. For example, in the bottom-left corner of the plot *ozone* is on the y-axis and *rad* is on the x-axis. Look at the plot where *ozone* is on the y-axis and *wind* is on the x-axis; there is a strong relationship between the two variables because the data points tightly follow the red line in the plot. It also appears that the relationship is non-linear due to the curvature seen.



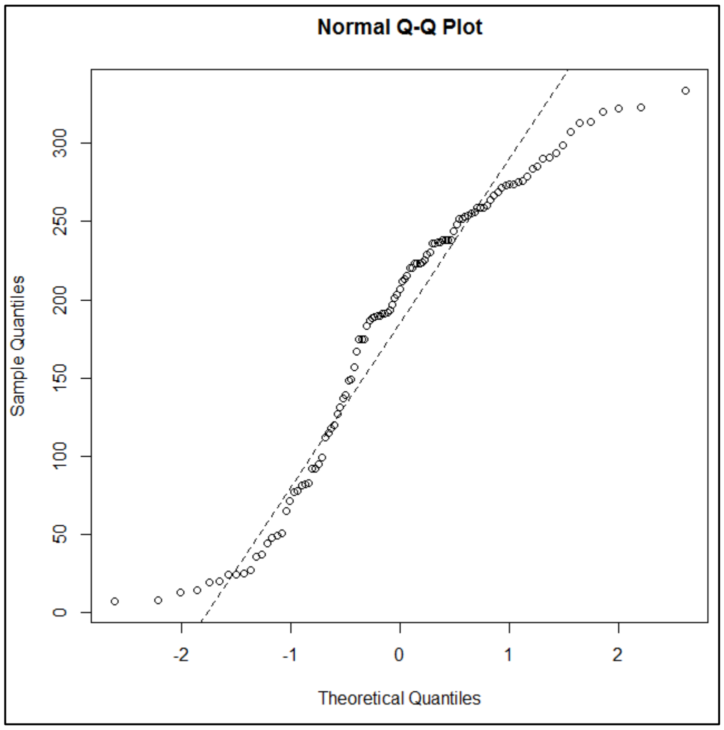
Histograms are excellent for viewing the spread of your data. A histogram provides the frequency in which certain data points appear. This also lets you eyeball the skewness and kurtosis of your data. The histogram on the previous page uses the following code for the variable *rad*:

```
hist(ozone_data$rad, main="Radiation Histogram")
```

3. Assessing Normality

Testing for normality is an important first step of familiarizing yourself with your data. In many ways, this will help determine what kinds of analysis you should perform. If non-normality is an issue, you may need to use a non-parametric test. Some statistical techniques, such as linear regression, require the data to exhibit a normal distribution.

Two simple tests exist for assessing normality: 1) Quantile-quantile plot and 2) Shapiro-Wilk test. The quantile-quantile plot, or QQ plot, is a more subjective assessment that relies on the statistician's eye. On the next page is a QQ plot. The straight, dashed line represents a normal distribution; the circles represent the data points of your variable. Notice the slight S-shape of the data. The tail ends appear above and slightly below the left and right ends of the normal distribution, respectively.



It should be noted that this plot is for a single variable. If you have several, say 12, then you would have to create 12 separate QQ plots. To perform this function in R, use the following two functions:

```
qqnorm(y)
qqline(y, lty=2)
```

Within each function is the argument *y*, which is the variable you are assessing. For example, if you have a variable *annual_sales* you would type the following:

```
qqnorm(annual_sales)
qqline(annual_sales, lty=2)
```

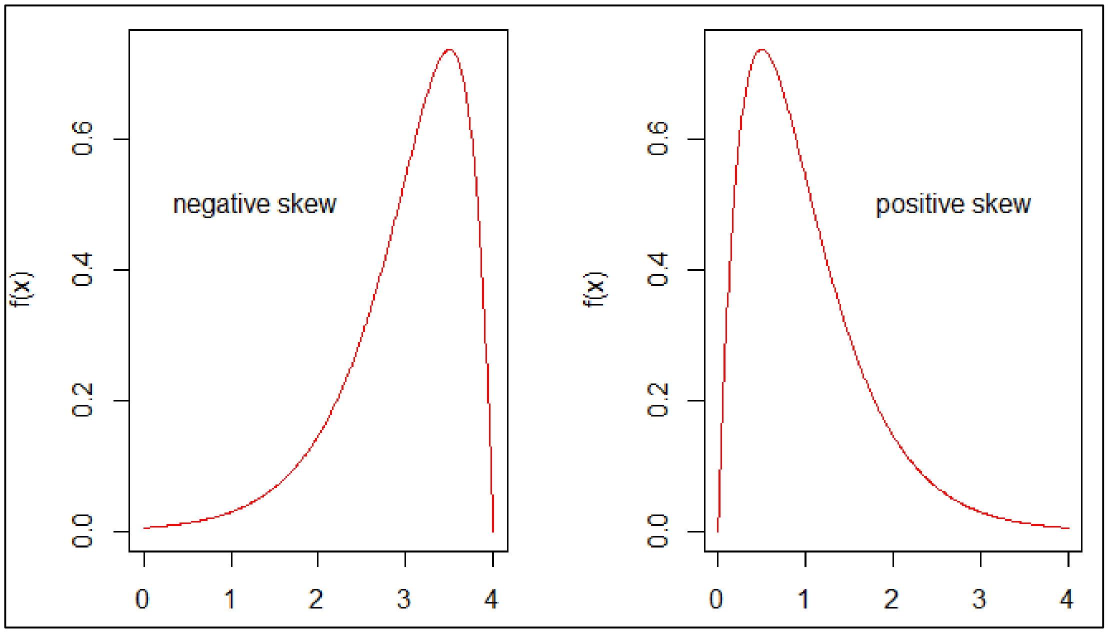
In addition to the QQ plot, you can use the Shapiro-Wilk test. This is considered a more objective assessment and provides a p-value. Normality results in a non-significant result of the test. Importantly, the significance depends on the alpha level you choose. I always take a more conservative approach and require an alpha of 0.05. The function for the Shapiro-Wilk test is written as follows:

```
shapiro.test(y)
```

4. Skewness and Kurtosis

After assessing the normality of your data, you need to understand the distribution in more detail. This requires you to look at the skewness and the kurtosis. While a histogram can help you to visually assess the skewness, it is better to use more objective criteria.

Skewness deals with the extent to which the end tails of the data are drawn out. Michael Crawley provides an excellent illustration in his book of what this may look like (page 286 1st edition, page 350 2nd edition), which I have included here. Note, that a negative skew is an indication of a skew to the left (see the tail end?) and a positive skew indicates a skew to the right. The positive and negative is in reference to the high point of that hump.



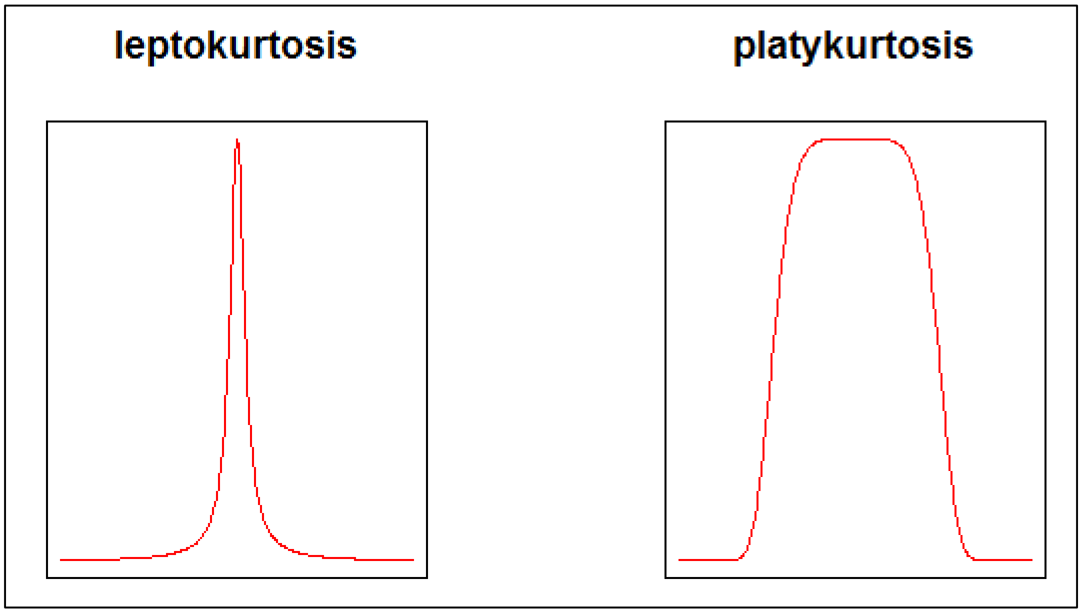
To calculate the skewness value for a single variable, simply type the following command in R using the psych package:

```
skew(x)
```

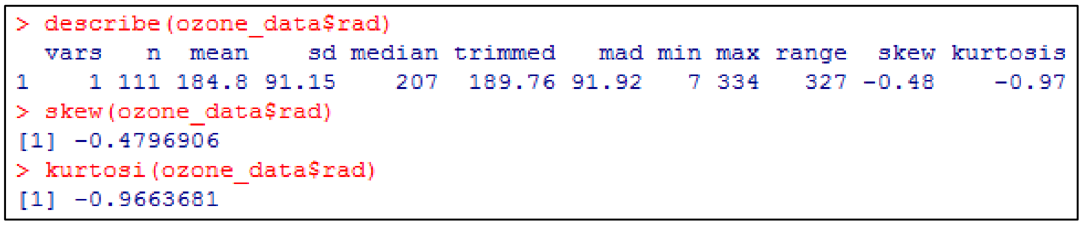
This is a univariate test. This results in a value that is negative (left skew), 0 (no skew), or positive (right skew). What a skew value does not indicate is how significant the skew value is. In other words, is it really worth worrying about? If it is, then a transformation, such as a square-root transformation, may be needed. Note, this value is the same as that given by the function describe(*x*).

If you are interested in performing a multivariate test, that is a test that takes into consideration all of your variables at the same time, then use the function mardia(*data*) where *data* is your data file or set of variables.

Kurtosis is another measure of normality that deals with the peak of a distribution. Two main types of kurtosis are encountered: leptokurtosis, which is very pointy, and platykurtosis, which is flat. See the images below:



The pysch package comes with a function to calculate the kurtosis value. Simply type in kurtosi(*x*) (yes, the "s" is intentionally left off of the end) and a value is provided. The function describe(*x*) also provides an estimate of kurtosis as shown below:



```
> describe(ozone_data$rad)
  vars   n  mean    sd median trimmed   mad min max range  skew kurtosis
1    1 111 184.8 91.15    207  189.76 91.92   7 334   327 -0.48    -0.97
> skew(ozone_data$rad)
[1] -0.4796906
> kurtosi(ozone_data$rad)
[1] -0.9663681
```