

# How can I perform simple linear regression in R step-by-step?

Authored by  
**stats writer**

April 21, 2024

## RECOMMENDED CITATION

stats writer (2024). *How can I perform simple linear regression in R step-by-step?*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=137712>

Simple linear regression is a popular statistical technique used to analyze the relationship between two continuous variables. In order to perform simple linear regression in R, the following step-by-step process can be followed:

#### Step 1: Load the necessary packages

The first step is to load the "stats" package, which contains the necessary functions for performing simple linear regression.

#### Step 2: Import the data

Next, the data must be imported into R using the "read.csv()" or "read.table()" function. Make sure to store the data in a data frame.

#### Step 3: Check for data quality

It is important to check for any missing values or outliers in the data before performing regression analysis. This can be done using functions such as "is.na()" or "boxplot()".

#### Step 4: Visualize the data

It is always helpful to visualize the data using scatter plots or histograms to get a better understanding of the relationship between the variables.

#### Step 5: Fit the regression model

Using the "lm()" function, the regression model can be fitted with the dependent variable as the first argument and the independent variable as the second argument.

#### Step 6: Check the model summary

The summary of the regression model can be viewed using the "summary()" function, which provides information such as the coefficients, p-values, and R-squared value.

#### Step 7: Interpret the results

Based on the model summary, the significance of the relationship between the variables can be determined. The coefficients can also be used to make predictions.

#### Step 8: Visualize the regression line

To visualize the regression line on the scatter plot, the "abline()" function can be used with the intercept and slope values from the regression model.

#### Step 9: Test for assumptions

It is important to test for the assumptions of simple linear regression, such as linearity, normality, and homoscedasticity. This can be done using diagnostic plots or statistical tests.

#### Step 10: Make predictions

Using the "predict()" function, predictions can be made for new data points based on the fitted

regression model.

By following these steps, simple linear regression can be performed in R, providing valuable insights into the relationship between two variables.

## Perform Simple Linear Regression in R (Step-by-Step)

**Simple linear regression is a technique that we can use to understand the relationship between a single explanatory variable and a single response variable.**

**In a nutshell, this technique finds a line that best "fits" the data and takes on the following form:**

$$y = b_0 + b_1x$$

**where:**

**$y$ : The estimated response value**  
 **$b_0$ : The intercept of the regression line**  
 **$b_1$ : The slope of the regression line**

**This equation can help us understand the relationship between the explanatory and response variable, and (assuming it's statistically significant) it can be used to predict the value of a response variable given the value of the explanatory variable.**

**This tutorial provides a step-by-step explanation of how**

## to perform simple linear regression in R.

### Step 1: Load the Data

For this example, we'll create a fake dataset that contains the following two variables for 15 students:

Total hours studied for some exam Exam score

We'll attempt to fit a simple linear regression model using *hours* as the explanatory variable and *exam score* as the response variable.

The following code shows how to create this fake dataset in R:

```
#create dataset
df <- data.frame(hours=c(1, 2, 4, 5, 5, 6, 6, 7, 8, 10, 11,
11, 12, 12, 14),
score=c(64, 66, 76, 73, 74, 81, 83, 82, 80, 88, 84, 82, 91,
93, 89))
```

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

```
head(df)
```

```
hours score
```

```
1 1 64
```

2 2 66

3 4 76

4 5 73

5 5 74

6 6 81

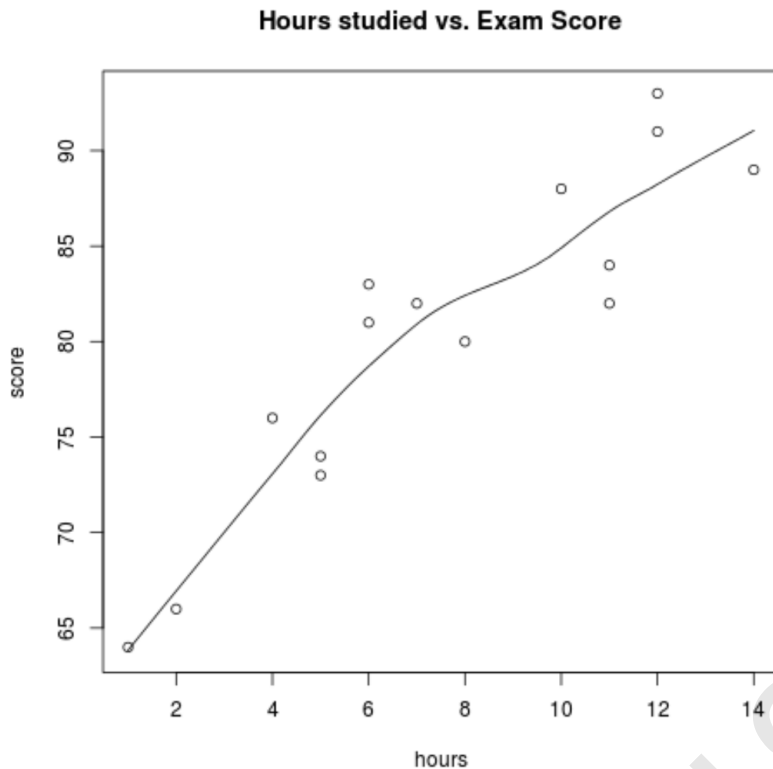
**#attach dataset to make it more convenient to work with  
attach(df)**

**Step 2: Visualize the Data**

**Before we fit a simple linear regression model, we should first visualize the data to gain an understanding of it.**

**First, we want to make sure that the relationship between *hours* and *score* is roughly linear, since that is a massive underlying assumption of simple linear regression. We can create a simple scatterplot to view the relationship between the two variables:**

**scatter.smooth(hours, score, main='Hours studied vs. Exam Score')**

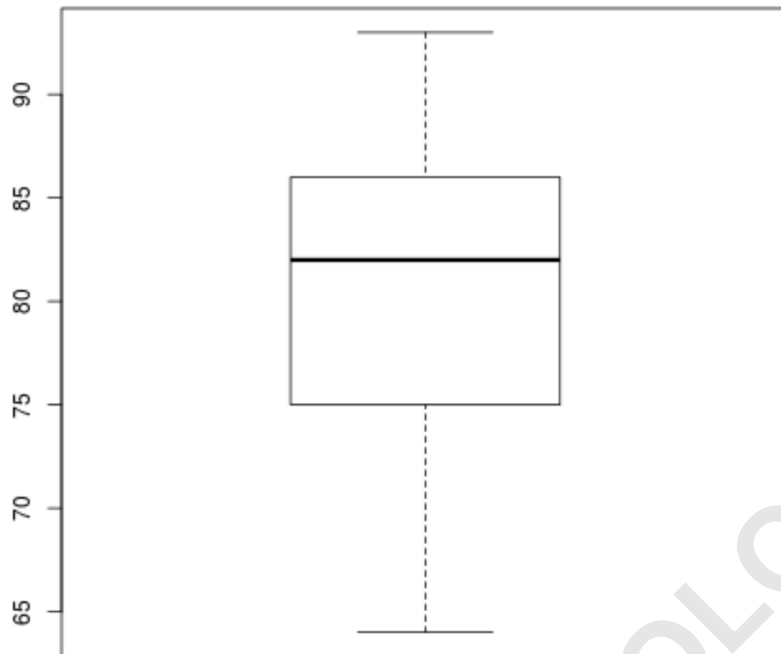


From the plot we can see that the relationship does appear to be linear. As *hours* increases, *score* tends to increase as well in a linear fashion.

Next, we can create a boxplot to visualize the distribution of exam scores and check for outliers. By default, R defines an observation to be an outlier if it is 1.5 times the interquartile range greater than the third quartile (Q3) or 1.5 times the interquartile range less than the first quartile (Q1).

If an observation is an outlier, a tiny circle will appear in the boxplot:

## boxplot(score)



**There are no tiny circles in the boxplot, which means there are no outliers in our dataset.**

### Step 3: Perform Simple Linear Regression

**Once we've confirmed that the relationship between our variables is linear and that there are no outliers present, we can proceed to fit a simple linear regression model using *hours* as the explanatory variable and *score* as the response variable:**

**#fit simple linear regression model**

```
model <- lm(score~hours)
```

```
#view model summary
```

```
summary(model)
```

**Call:**

```
lm(formula = score ~ hours)
```

**Residuals:**

```
Min 1Q Median 3Q Max
```

```
-5.140 -3.219 -1.193 2.816 5.772
```

**Coefficients:**

```
Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept) 65.334 2.106 31.023 1.41e-13 ***
```

```
hours 1.982 0.248 7.995 2.25e-06 ***
```

```
---
```

```
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

**Residual standard error: 3.641 on 13 degrees of freedom**

**Multiple R-squared: 0.831, Adjusted R-squared: 0.818**

**F-statistic: 63.91 on 1 and 13 DF, p-value: 2.253e-06**

**From the model summary we can see that the fitted**

regression equation is:

$$\text{Score} = 65.334 + 1.982 * (\text{hours})$$

This means that each additional hour studied is associated with an average increase in exam score of 1.982 points. And the intercept value of 65.334 tells us the average expected exam score for a student who studies zero hours.

We can also use this equation to find the expected exam score based on the number of hours that a student studies. For example, a student who studies for 10 hours is expected to receive an exam score of 85.15:

$$\text{Score} = 65.334 + 1.982 * (10) = 85.15$$

Here is how to interpret the rest of the model summary:

**Pr(>|t|):** This is the p-value associated with the model coefficients. Since the p-value for *hours* (2.25e-06) is significantly less than .05, we can say that there is a statistically significant association between *hours* and *score*. **Multiple R-squared:** This number tells us the percentage of the variation in the exam scores can be explained by the number of hours studied. In general,

the larger the R-squared value of a regression model the better the explanatory variables are able to predict the value of the response variable. In this case, 83.1% of the variation in scores can be explained hours studied. Residual standard error: This is the average distance that the observed values fall from the regression line. The lower this value, the more closely a regression line is able to match the observed data. In this case, the average observed exam score falls 3.641 points away from the score predicted by the regression line. F-statistic & p-value: The F-statistic (63.91) and the corresponding p-value (2.253e-06) tell us the overall significance of the regression model, i.e. whether explanatory variables in the model are useful for explaining the variation in the response variable. Since the p-value in this example is less than .05, our model is statistically significant and *hours* is deemed to be useful for explaining the variation in *score*.

#### Step 4: Create Residual Plots

After we've fit the simple linear regression model to the data, the last step is to create residual plots.

One of the key assumptions of linear regression is that

the residuals of a regression model are roughly normally distributed and are homoscedastic at each level of the explanatory variable. If these assumptions are violated, then the results of our regression model could be misleading or unreliable.

To verify that these assumptions are met, we can create the following residual plots:

**Residual vs. fitted values plot:** This plot is useful for confirming homoscedasticity. The x-axis displays the fitted values and the y-axis displays the residuals. As long as the residuals appear to be randomly and evenly distributed throughout the chart around the value zero, we can assume that homoscedasticity is not violated:

```
#define residuals
```

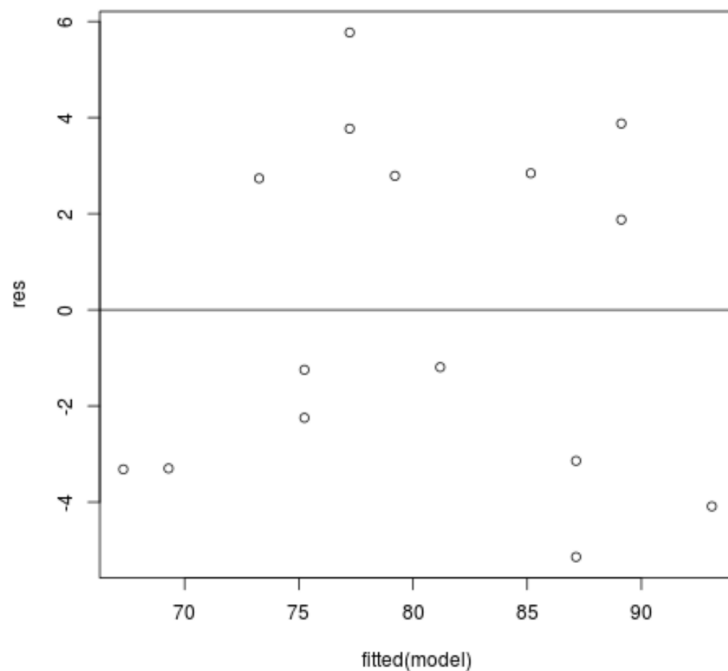
```
res <- resid(model)
```

```
#produce residual vs. fitted plot
```

```
plot(fitted(model), res)
```

```
#add a horizontal line at 0
```

```
abline(0,0)
```



The residuals appear to be randomly scattered around zero and don't exhibit any noticeable patterns, so this assumption is met.

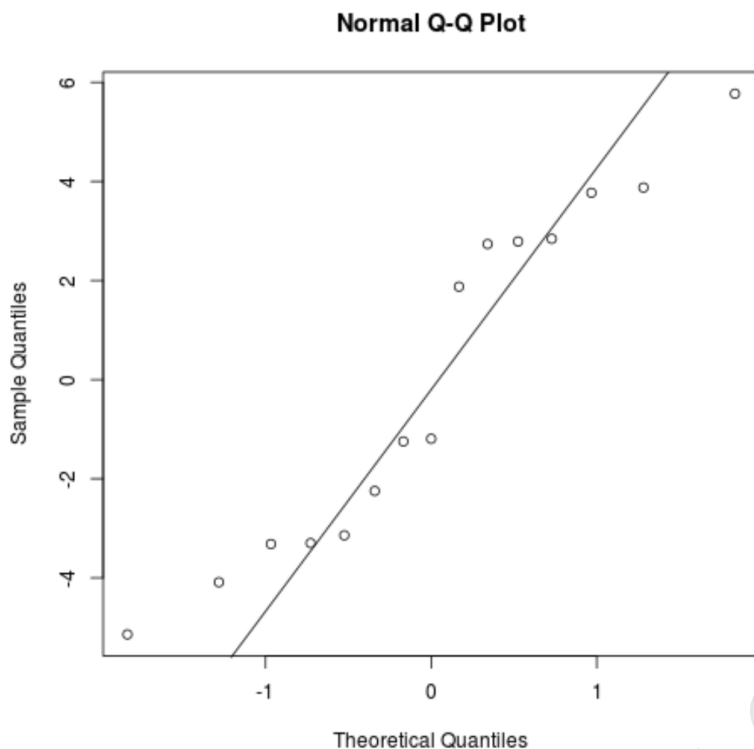
**Q-Q plot:** This plot is useful for determining if the residuals follow a normal distribution. If the data values in the plot fall along a roughly straight line at a 45-degree angle, then the data is normally distributed:

**#create Q-Q plot for residuals**

**qqnorm(res)**

**#add a straight diagonal line to the plot**

**qqline(res)**



The residuals stray from the 45-degree line a bit, but not enough to cause serious concern. We can assume that the normality assumption is met.

Since the residuals are normally distributed and homoscedastic, we've verified that the assumptions of the simple linear regression model are met. Thus, the output from our model is reliable.

***The complete R code used in this tutorial can be found [here](#).***