Statistically deriving the initial speed from motion of cars data using R

M Shahruz Z Q

Jan 14. 2026

This is a report on how we derived the initial speed of cars from data about their speed and distances using statistical methods. To help us with our goals we wrote a R function that is able to respond with the initial speed of cars by analysing any data set that is similar. We are confident that our result was accurate as we ensured the accuracy of our result by testing the function with known data.

Introduction

It is well known that the speed of a car can be found simply by dividing the distance it covered by the time it took to cover that distance. In other words *speed = distance / time*. With reference to that formula we can also say that $time = distance/speed$. So if we have a data set consisting of the speed of a car over different distances we can easily find the time when that car was moving at a certain speed by dividing the distance when the car was moving at that speed by the value we have for speed. But what if the data does not contain the speed of the car at a certain distance or time and we have to calculate it from the data? For example how do we know its initial speed?

There might be multiple ways to do it. For example we can use the formula that says final speed is equal to the initial speed multiplied by the product of acceleration and time. To put it more simply, $finalspeed(v) = initialspeed(u) + acceleration(a) * time(t)$. The use of this formula can be relatively more complicated even with the help of computer programming because there are more steps involved (first we need to find the time when the car was moving at final speed and a value for acceleration) and the use of this formula can become inefficient to some extent for certain situations but it can also be useful in other situations for example where more than approximate values are needed.

Another and more simpler way to find the initial speed from such data is to use a well known statistical method called linear modelling. This is a report on a statistical analysis where we are given a data set consisting of the speed and distances of a car over fifty different distances and we want to find the initial speed of the car from it (the initial speed of a car is the speed of the car when time is 0 hours or maybe 0 seconds). We will use the R programming language to simplify our statistical analysis. It will be a reuseable R function that will be able to analyse any data set that is similar to our present data (for this statistical analysis we will be using the **cars** data that comes built in with a default R installation).

Details of our method and the supporting code

The method

We will apply the statistical method called linear modelling on our car motion data to find the initial speed. If such a linear model was represented visually it will involve drawing a best fit straight line through a scatter plot of speed against time. A scatter plot of speed against time for our data is shown below:

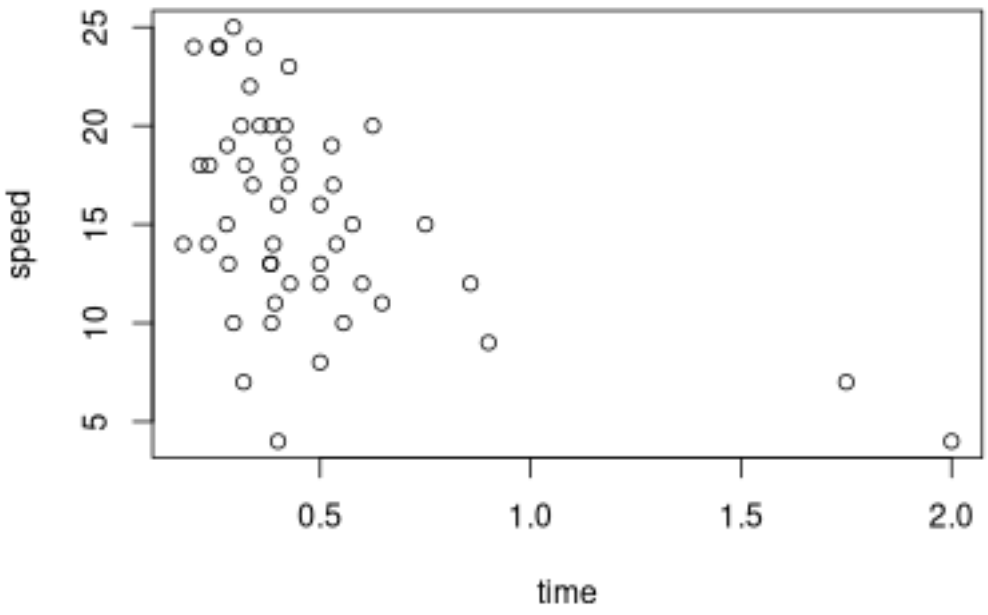

Figure 1: Scatter plot showing speed (km/h) against time (h) for **cars** data

This linear model can then be used to find the speed when the time is 0. For this we use regression of our linear model and we extend the line until it intersects the speed(y)-axis and then note the speed(y)-coordinate where the line intersects it, known as the the **intercept**, because it is at this point that the time(x)-coordinate is 0 and the speed(y)-coordinate is equal to the initial speed.

As we can see time data is crucial for deriving the initial speed from a linear model of speed against time but the data we are dealing with does not directly provide time data. So for our analysis we have to do the following:

- We are only provided with speed and distances so we will first have to find the time data for them.
- Then we will build our linear model of speed against time and get its intercept value.
- We can simplify this process by writing a function in the R programming language which will quickly respond with the initial speed value for our data. It will also be able to respond with the initial speed value for any data that is similar.
- This reusability is an advantage for us because we will be able to provide it another data set with known initial speed value and test whether our function returns the right output. And that will be the last step of our statistical analysis where we will test our function because it will aslo ensure the accuracy of the result of our actual **cars** data.

The code

In addition to having a text editor (or text editors) that is optimized for performance it will also help if before we start writing our function we first create an appropriate directory structure similar to the one shown below:

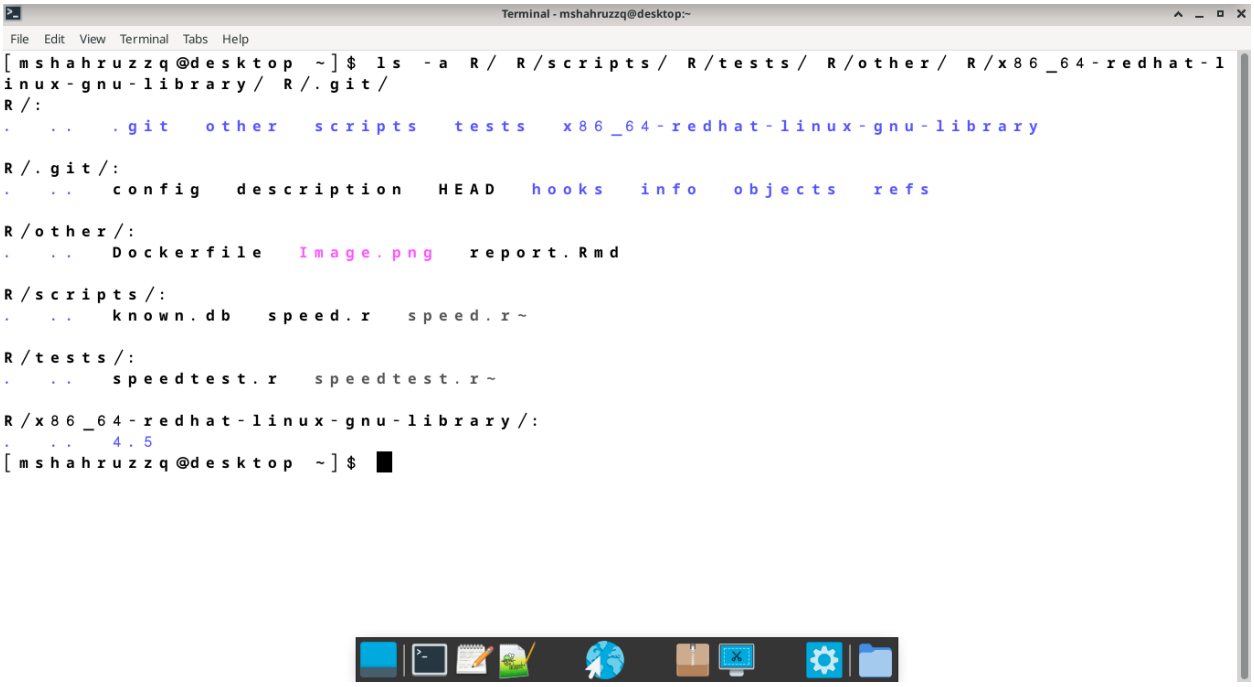

Figure 2: Screenshot showing a directory structure for performing statistical analysis with R

What follows is the R code we will use for our statistical analysis:

```
library(DBI)
library(RSQLite)
library(dplyr)

con <- dbConnect(RSQLite::SQLite(), dbname = "/home/mshahruzzq/R/scripts/known.db")
```

```
cknown <- dbGetQuery(con, "SELECT * FROM cknown;")
dbDisconnect(con)

ispeed <- function(cdata = cknown) {
    cdtime <- cdata |> mutate(time = dist / speed)
    cdtlmodel <- lm(speed ~ time, data = cdtime)
    cdtlmspeed <- coef(cdtlmodel)[("(Intercept)")]
    cdtgradient <- ((cdtime$speed)[49] - (cdtime$speed)[0]) / ((cdtime$dist)[49] - (cdtime$dist)[0])
    cdtlmsunit <- paste(cdtlmspeed, "km/h")
    if(isTRUE(cdtgradient == 1)) {
        cdtlmsunit

    }
    else {
        print(cdtlmsunit)
    }

}
```

Here is a break down of the code in some detail:

1. First we load required libraries **DBI**, **RSQLite** and **dplyr** (data cleaning will not be necessary for this statistical code otherwise we would be loading **tidyr** also).
2. Next we connect to a database that contains a data set of speed and distances with a known initial speed value. The data set is contained in a table which we query and store directly into a data frame and then we disconnect the database connection.
3. Finally we write our function that takes one parameter and that one parameter can be an entire data set such as the **cars** data that we want to analyse. If we do not pass any parameter when we call our function it will use the default data set which we previously stored into a data frame from a database query. Once called the function generates the time data, extracts the intercept value from a linear model of speed against time and returns a response letting us know the speed when time is 0.

There would still be is one last step in our statistical analysis. Since the code is able to process any data set that is similar to our present data set (**cars**) it is not only possible but also important that we ensure the accuracy of the results that are returned by testing the function in it with a second data set. Here is some additional R code that we might want to use to test our function:

```
library(testthat)

source("/home/mshahruzzq/R/scripts/speed.r")

test_that("Function ispeed(data) returns 25.5 km/h if called without any parameter", {
    expect_equal(ispeed(), "25.5 km/h")
})
```

This is how the test code works:

1. We load the **testthat** library.
2. We load the source file that contains the code for our statistical function so that we are able to test it.
3. We call the test functions in the library to test whether our target statistical function returns the right output for a second data set for which the initial speed is known. As we wrote our target function in a way that it uses a second data set (queried from a database) by default if the function is called without any data parameter and is supposed to return a known value for initial speed, the test functions simply call the target function without a parameter and compares the output to the known value. If the test passes we will know that our function is responding with accurate initial speed values for all data sets, including the initial speed we get for the **cars** data.

Output and the results of the analysis

Finally we will pass our data to the function we have written:

```
source("/home/mshahruzzq/R/scripts/speed.r")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
ispeed(cars)
```

```
## [1] "9.65479076979405 km/h"
```

To ensure that the output is accurate we will also run our test code:

```
testthat::test_file("/home/mshahruzzq/R/tests/speedtest.r")
```

```
##
## Attaching package: 'testthat'
```

```
## The following object is masked from 'package:dplyr':
##
##     matches
```

```
##
## == Testing speedtest.r ======================================================
## [ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ][1] "25.5 km/h"
## [ FAIL 0 | WARN 0 | SKIP 0 | PASS 1 ] Done!
```

Conclusion

The purpose of this report was to document our attempt to statistically find the initial speed from a data set consisting of a number of speed and distance values. To support our analysis we wrote a R function and processed our data. As we were able to obtain a value for the initial speed and our function was tested to ensure that it produces accurate responses, it can be confirmed that our analysis was completed successfully.