Week 9 Module 4 – Scatterplots

CSCI E-5a: Programming in R

Let's clear the global computing environment:

```
rm( list = ls() )
```

Module Overview and Learning Objectives

Hello! And welcome to Module 5: Scatter Plots.

In this module we'll learn about a data visualization method called a *scatter plot*.

- In Section 1, we'll discuss the basic concept behind a scatter plot, and see how to construct this graph in R and add a grid to it.
- In Section 2, we'll explore the concept of a "least-squares regression line", and learn how to draw a regression line on a scatterplot.
- In Section 3, we'll learn more about linear model objects, and see how to select the regression coefficients from such objects.

When you've completed this module, you should be able to:

- Construct a scatter plot to display the relationship between two numeric variables.
- Add a grid to the scatter plot.
- Add a least-squares regression line to the graph.
- Select the slope and y-intercept regression coefficients from a linear model object.

There are three new built-in R functions in this module:

- `grid()`
- `lm()`
- `abline()`

Section 1: The Scatter Plot

> **Main Idea:** *We can use a scatter plot to visualize the relationship between two variables*

In this section, we'll discuss the basic concept behind a scatter plot, and see how to construct this graph in R and add a grid to it.

Scatter plots enable us to visualize the relationship between two variables.

In order to use a scatter plot, the two variables must be *coordinated*: that means that the corresponding entries have to be related.

The data in the `cars` data frame is coordinated: it consists of a speed measurement and a stopping distance, and corresponding values of the two variables are related, because they both come from the same experimental replication.

To visualize the relationship between the `speed` and `dist` variables, we then plot a point for each pair of corresponding values:

- The x-coordinate of the point will be the value of the `speed` variable.
- The y-coordinate of the point will be the value of the `dist` variable.

Let's recall some of the values in the `cars()` dataset:

```
head( cars )
```

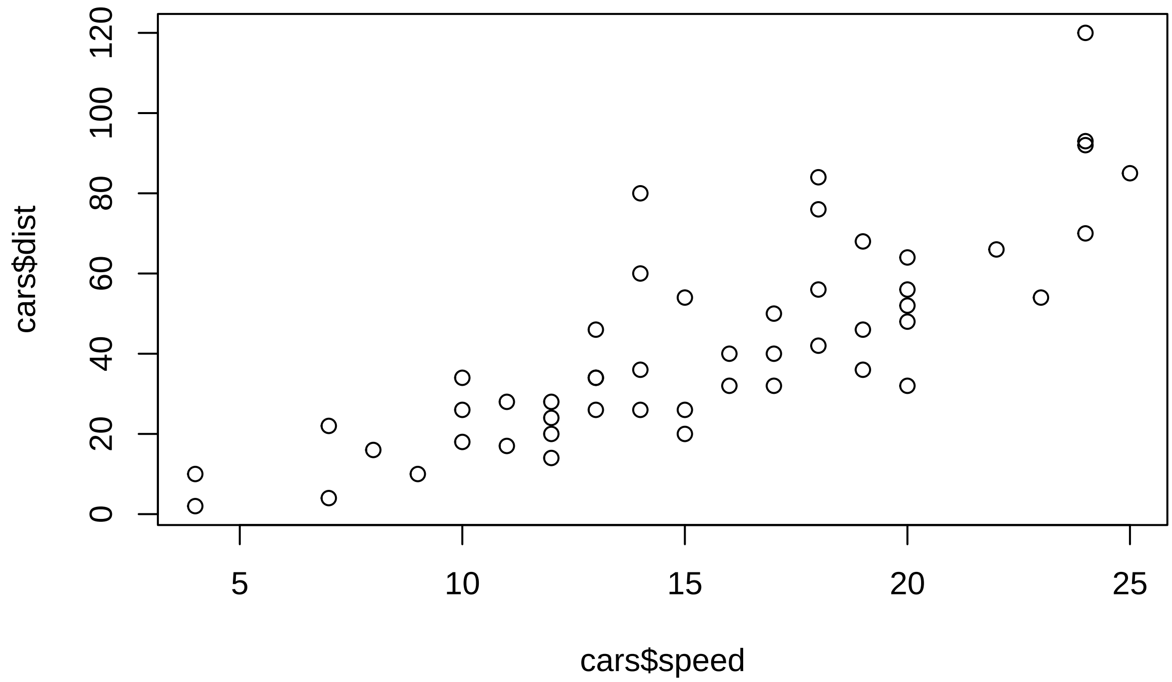
```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

To construct a scatter plot, we use the `plot()` function.

If you go back to the very beginning of the course, the first plot that we made was a scatter plot with one point, using the `plot()` function.

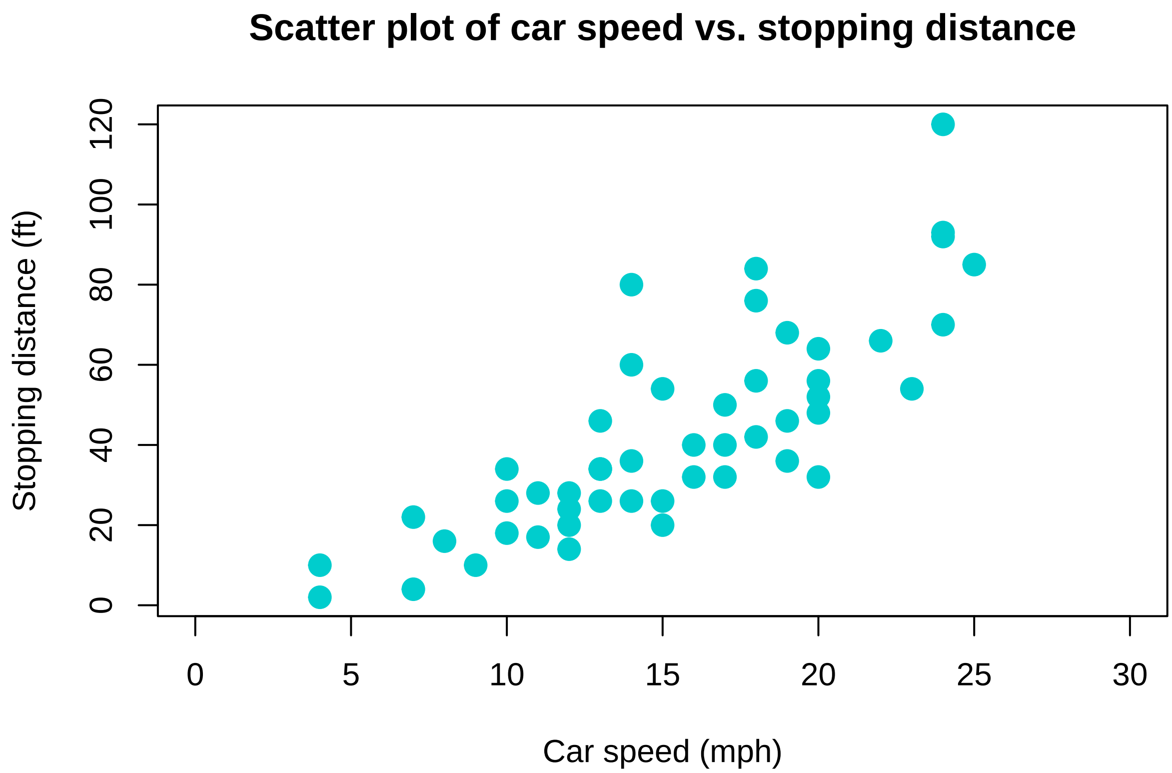
We'll start by making the simplest possible scatter plot:

```
plot(
    x = cars$speed,
    y = cars$dist
)
```



Let's make this look a little nicer:

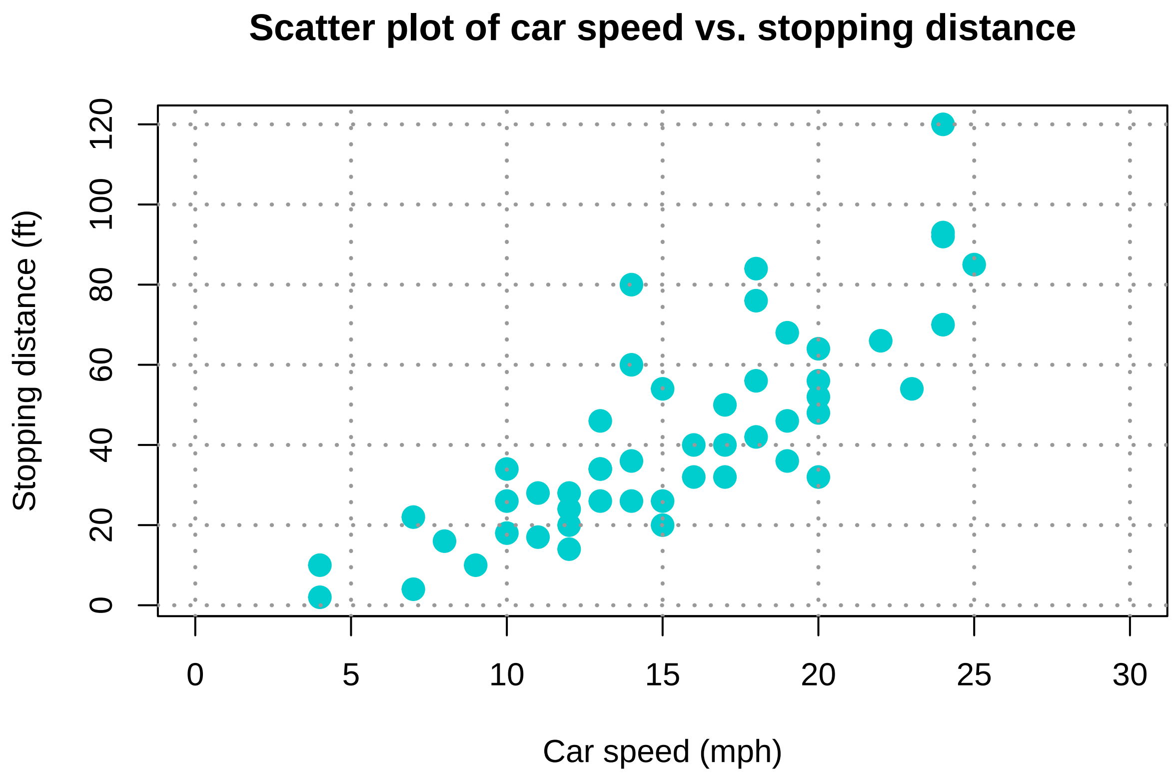
```
plot(
    x = cars$speed,
    y = cars$dist,
    xlim = c(0, 30),
    main = "Scatter plot of car speed vs. stopping distance",
    xlab = "Car speed (mph)",
    ylab = "Stopping distance (ft)",
    pch = 19,
    cex = 1.5,
    col = "cyan3"
)
```



We can add a grid to our scatter plot:

```
plot(
    cars$speed,
    cars$dist,
    xlim = c(0, 30),
    main = "Scatter plot of car speed vs. stopping distance",
    xlab = "Car speed (mph)",
    ylab = "Stopping distance (ft)",
    pch = 19,
    cex = 1.5,
    col = "cyan3"
)


grid(
    col = "gray60",
    lwd = 2
)
```



So that's how to construct a scattplot to visualize the relationship between two numeric variables.

Now let's see how to construct a least-squares regression line through our the points of the scatter plot.

Section 2: Adding a Least-Squares Regression Line

> **Main Idea:** *We can add a least-squares regression line to a scatter plot*

In this section, we'll explore the concept of a "least-squares regression line", and learn how to draw a regression line on a scatter plot.

A popular statistical method for visualizing the relationship between two numeric variables such as the `dist` and `speed` variables in the `cars` data frame is to construct what is called a "regression line", or sometimes a "least-squares regression line".

We won't get into the statistical theory of what this means, or how to calculate it, but the basic idea is to draw the "best-fitting" line through our two-dimensional display of points in the scatter plot.

To draw this "best-fitting least-squares regression line", we first have to run what's called a "linear model", and then we can use the output of this procedure to draw the regression line.

To do this, we construct what's called a "linear model", and to do this we use the `lm()` function:

```
cars.regression.model <-
    lm( cars$dist ~ cars$speed )
```

The `lm()` function has a nice feature: since we're almost always working with data in data frame, we can specify the data frame using the `data` option, and then just refer to the variables using the column names, without the $ notation:

```
cars.regression.model <-
    lm(
      formula = dist ~ speed,
      data = cars
    )
```

Notice the curious notation for the argument of the `lm()` function.

This expression is called a *formula*, and it indicates that we want to determine the least-squares line where the `speed` data is represented by the x-coordinate and the `dist` values are represented by the y-coordinate.

Notice that ~ character in the formula – it's called a "tilde", and you should make sure that you know how to type it on your keyboard.

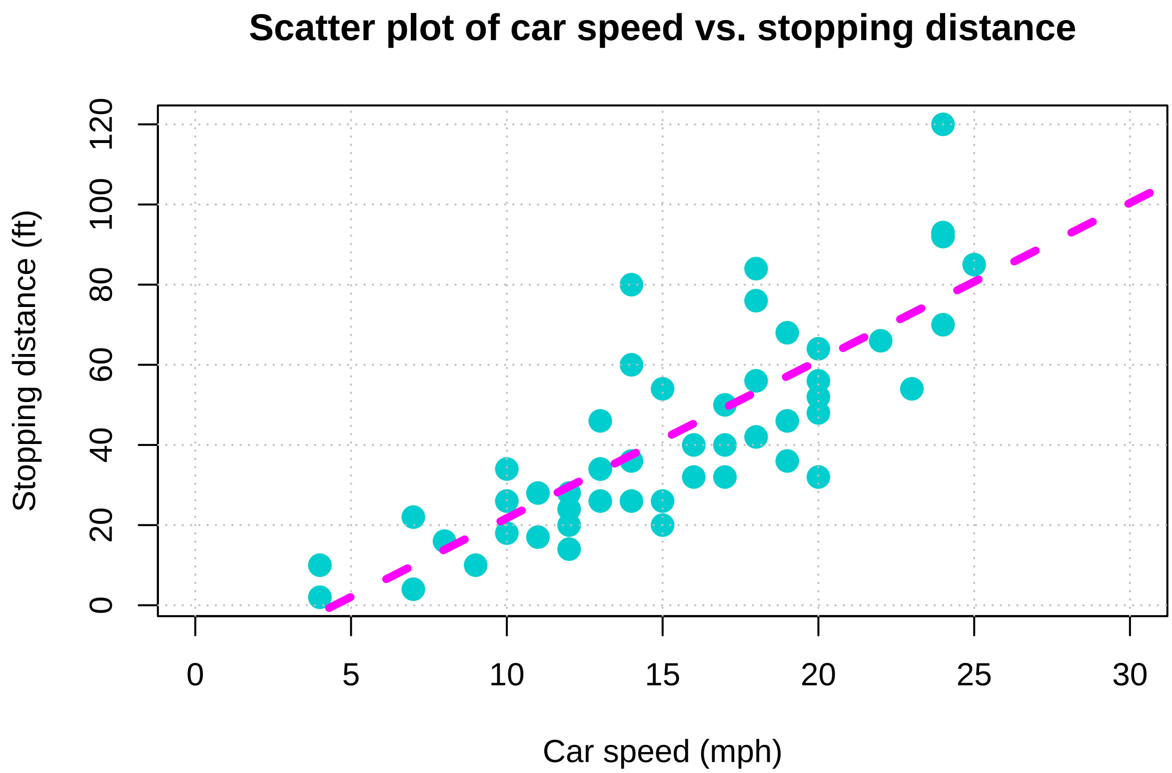
To add a regression line to your scatter plot, we use the `abline()` function with the result of the `lm()` function:

```
plot(
    cars$speed,
    cars$dist,
    xlim = c(0, 30),
    main = "Scatter plot of car speed vs. stopping distance",
    xlab = "Car speed (mph)",
    ylab = "Stopping distance (ft)",
    pch = 19,
    cex = 1.5,
    col = "cyan3"
)

grid( col = "gray75" )

cars.regression.model <-
    lm( cars$dist ~ cars$speed )

abline(
    cars.regression.model,
    col = "magenta",
    lwd = 4,
    lty = "dashed"
)
```



So that's how to add a least-squares regression line to a scatter plot.

Now let's see how to extract the regression coefficients from the linear model.

Section 3: Regression Coefficients

> **Main Idea:** *We can determine the coefficients of the linear model*

In this section, we'll learn more about linear model objects, and see how to select the regression coefficients from such objects.

When we just want to draw the least-squares regression line for a scatter plot, we don't actually have to worry about the details.

We just run the `lm()` function, and then pass the result to the `abline()` function, and the line gets drawn.

In particular, the regression line has a slope and a y-intercept, but we never had to deal with the actual values.

Instead, all of that information was bundled up in the complex object returned by the `lm()` function, and then `abline()` unpacked and drew the line for us.

Sometimes however you *do* want to know the actual numerical values of the slope and the y-intercept.

How can we extract that information from the linear model object returned by the `lm()` function?

The output of the `lm()` function is a complicated object, and it has its own special class:

```r
class( cars.regression.model )
```

```
## [1] "lm"
```

```r
is.list( cars.regression.model )
```

```
## [1] TRUE
```

If you want to see a report of the regression analysis of `cars.regression.model`, you can use the `summary()` function:

```r
summary( cars.regression.model )
```

```
##
## Call:
## lm(formula = cars$dist ~ cars$speed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## cars$speed    3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

That's nice, and if you take a statistics class I hope it's useful for you.

But we want to actually extract the values for the slope and y-intercept, and then store them in variables.

To do this, we first select the vector of coefficients from the linear model object.

The linear model object is a named list, so we can just use our standard list indexing method:

```r
regression.coefficients <-
    cars.regression.model$coefficients

regression.coefficients
```

```
## (Intercept)  cars$speed
##  -17.579095    3.932409
```

Do you see what that is?

It's just a named numeric vector.

The y-intercept always has the name `(Intercept)`, regardless of the variables that you used to construct the regression:

```
y.intercept <-
    regression.coefficients[ "(Intercept)" ]

cat(
    "Y-intercept of least-squares regression line:",
    formatC(
        y.intercept,
        format = "f",
        digits = 5
    )
)
```

```
## Y-intercept of least-squares regression line: -17.57909
```

The name of the slope coefficient *will* be different for different regression models, but it always be the name of the variable on the right-hand side of the formala that you used to construct the linear model.

For our example, we used the variable `cars$speed`, and so that's name of the slope coefficient:

```
slope <-
    regression.coefficients[ "cars$speed" ]

cat(
    "Slope of least-squares regression line:",
    formatC(
        slope,
        format = "f",
        digits = 5
    )
)
```

```
## Slope of least-squares regression line: 3.93241
```

So that's how to obtain the regression coefficients from a linear model object.

Now let's review what we've learned in this module.

Module Review

In this module we learned about a data visualization method called a *scatter plot*.

- In Section 1, we discussed the basic concept behind a scatter plot, and saw how to construct this graph in R and add a grid to it.
- In Section 2, we explored the concept of a "least-squares regression line", and learned how to draw a regression line on a scatter plot.
- In Section 3, we learned more about linear model objects, and saw how to select the regression coefficients from such objects.

Now that you've completed this module, you should be able to:

- Construct a scatter plot to display the relationship between two numeric variables.
- Add a grid to the scatter plot.
- Add a least-squares regression line to the graph.
- Select the slope and y-intercept regression coefficients from a linear model object.

There were three new built-in R functions in this module:

- `grid()`
- `lm()`
- `abline()`

Allright, that's it for Module 5: Scatter Plots, and in fact that's the end of the material for Unit 9: Data Frames.

Now let's move on to Unit 10: Managing Data.

See you there!