

# DATA VISUALIZATION WITH R

Noratiqah Mohd Ariff



# Common Arguments

- For most charts in basic R, common arguments across various functions can be applied.
- We can add the title by using arguments such as `main=` for the header, `sub=` for subtitles, `xlab=` for the labels in x-axis and `ylab=` for labels of the y-axis.
- The font size can be adjusted using the argument `cex` with `cex.main`, `cex.sub`, `cex.axis` and `cex.lab`.
- The colours are changed using the argument `col` with `col.main`, `col.sub`, `col.axis`, `col.lab` and `col` (for the points and lines).
- We can set the range for our graph using the argument `ylim=` and `xlim=`.

# The `plot` Function

- Use for scatter plot and line graph.
- If two vectors are provided in the argument, the first will be the values for the x-axis and the second is the values for the y-axis.
- If only one vector is given as the argument, it will be taken as the values for the y-axis and the scatter plot is against the index for the vector.
- The function could also take the form `plot(y~x, data=...)`
- By default, the plots will be in points. We can change this using the argument `type=`
  - p - points
  - l - line plot
  - b - both but not overlapping
  - o - both and overlapping
  - h - straight line
  - s - step line
  - n - none







# The `plot` Function

- You can change the style of the points using the argument `pch=`

○ 1	△ 2	⊕ 3	× 4	◇ 5
▽ 6	⊠ 7	⋆ 8	⊞ 9	⊕ 10
⊗ 11	⊞ 12	⊗ 13	⊞ 14	■ 15
● 16	▲ 17	◆ 18	● 19	● 20
○ 21	□ 22	◇ 23	△ 24	▽ 25

# The `plot` Function

- The lines can be edited using the argument `lty=`

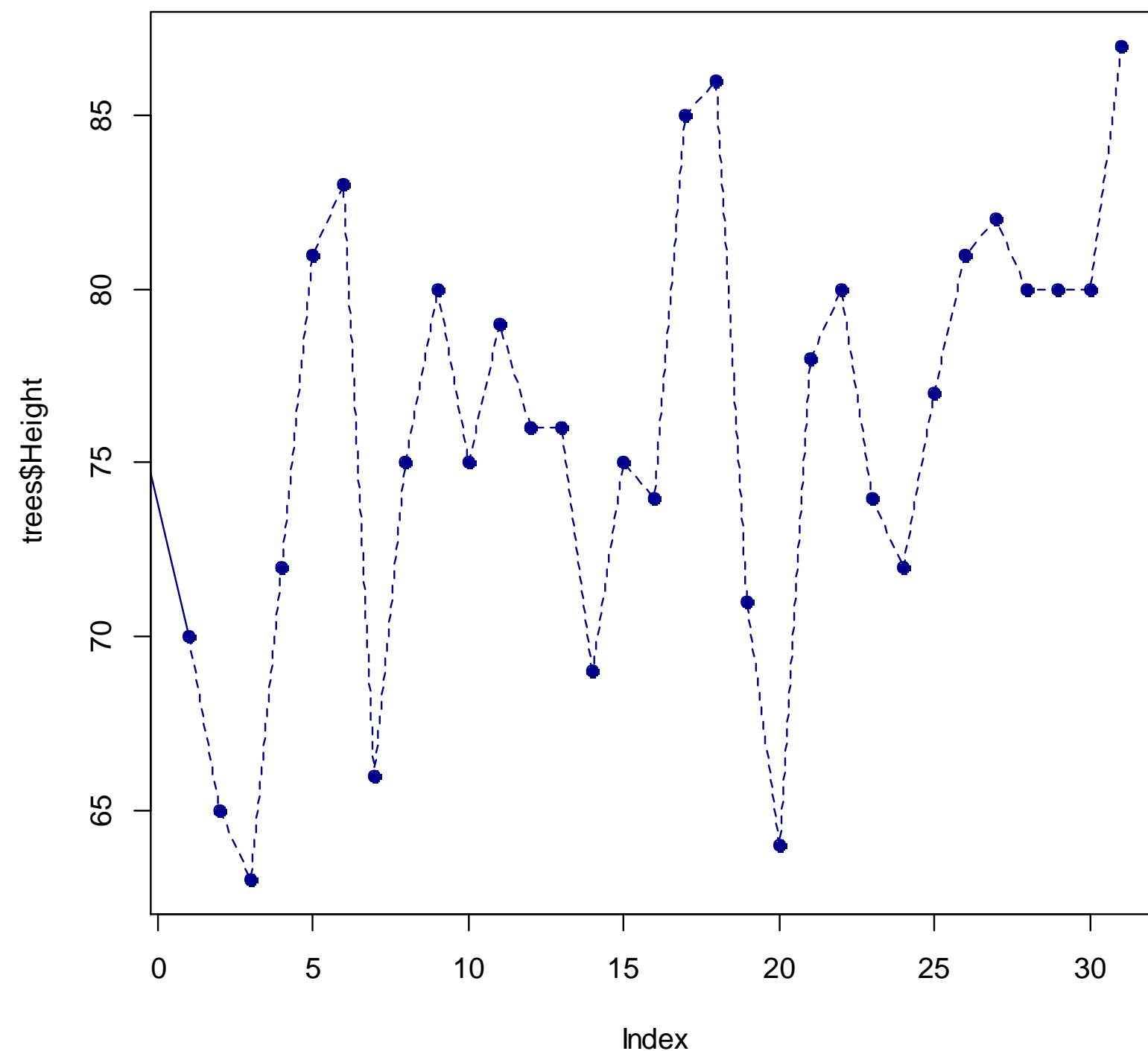
0. 'blank'	
1. 'solid'	
2. 'dashed'	
3. 'dotted'	
4. 'dotdash'	
5. 'longdash'	
6. 'twodash'	

- The colours are changed using the argument `col=`

# The `plot` Function

- Example:

```
plot(trees$Height, type="o",  
     pch=19, lty=2, col="darkblue")
```



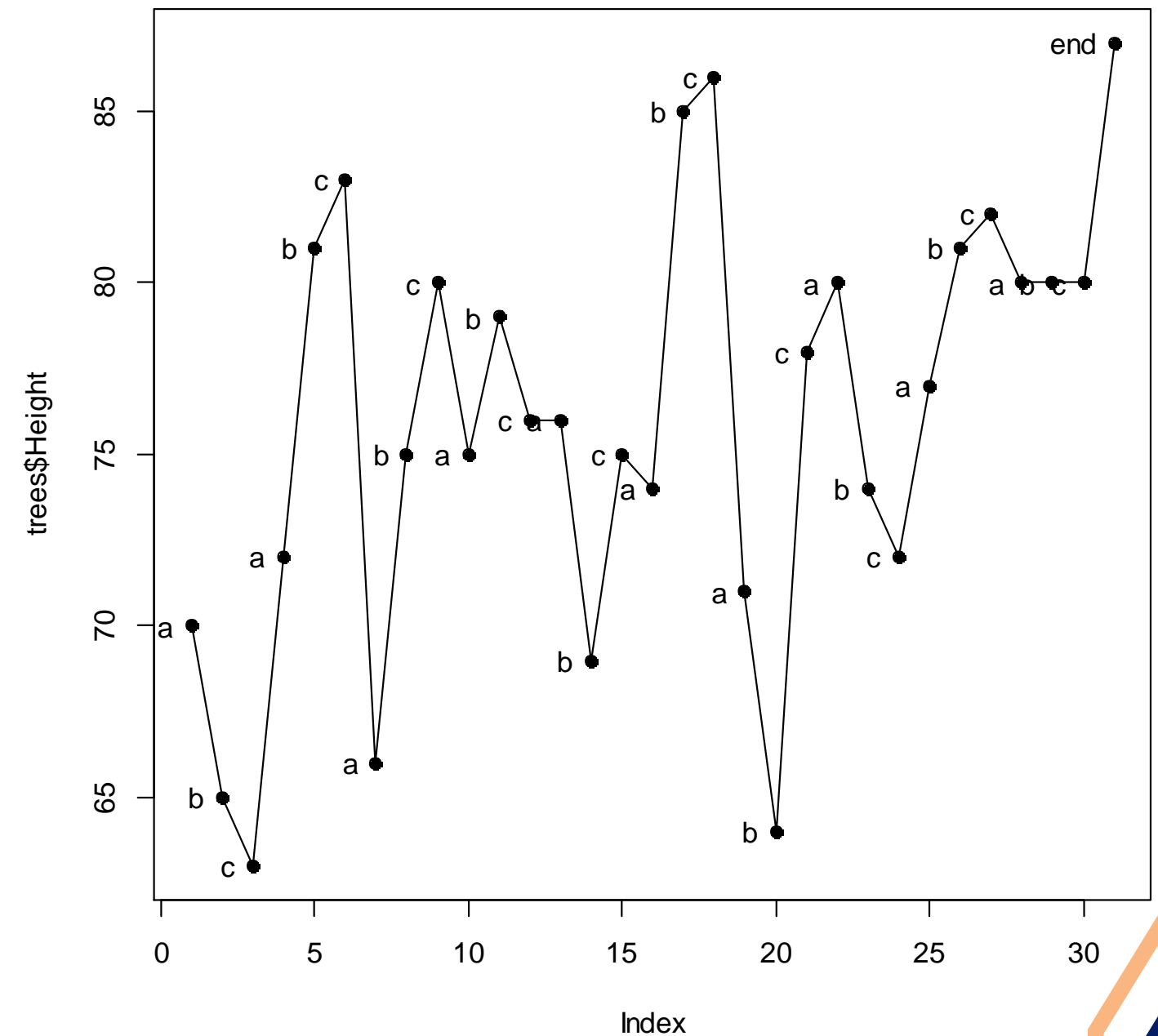


# The `text` Function

- Use `text()` function to add text in a graph.
- The argument `pos=` will set the position of the text in the graph with 1, 2, 3 and 4 indicate below, to the left, above and to the right respectively.

- **Example:**

```
plot(trees$Height, type="o", pch=19)
tplot<-c(rep(c("a", "b", "c"), 10), "end")
text(trees$Height, tplot, pos=2)
```

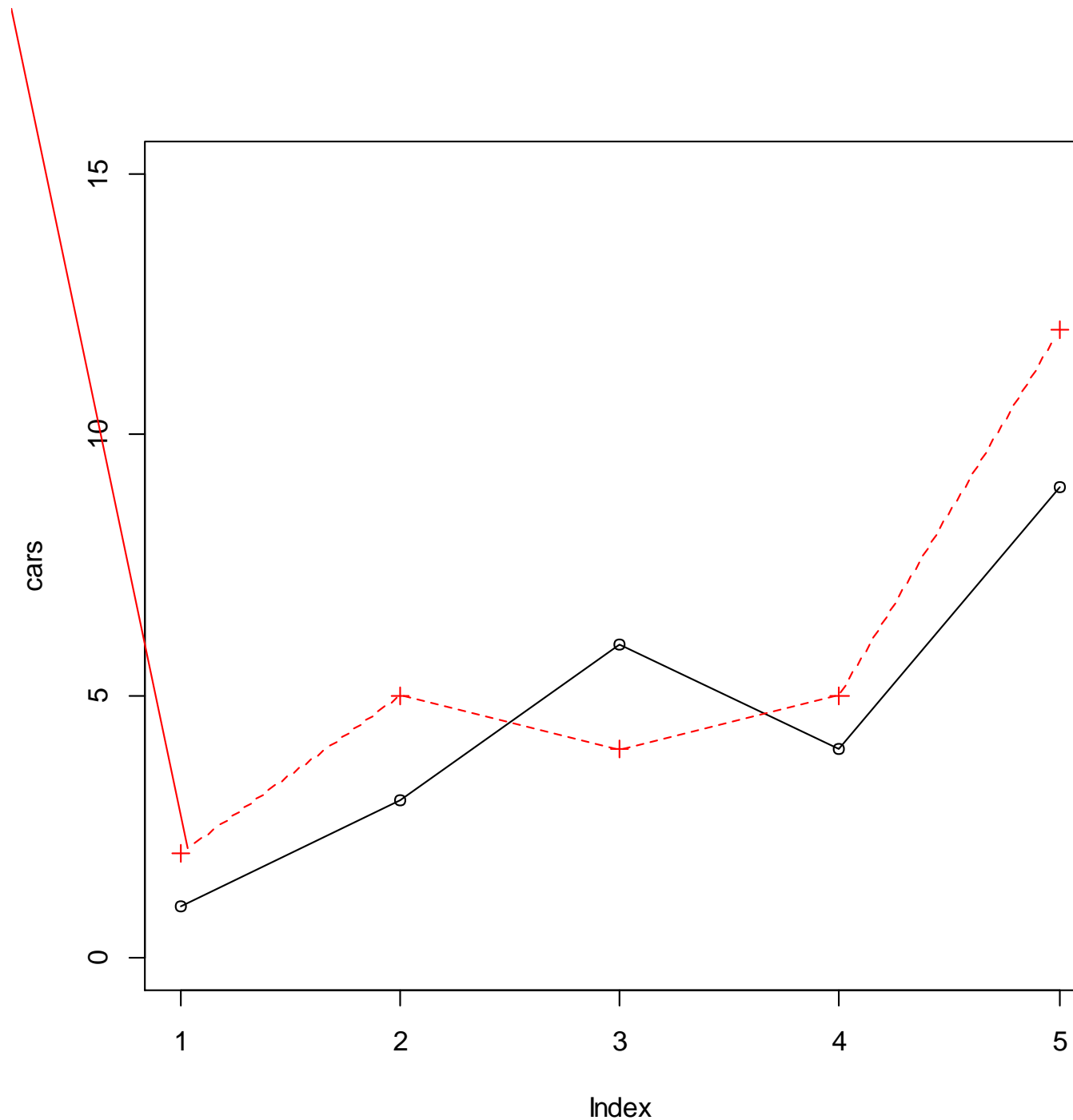


# The `points` and `lines` Functions

- Add points and lines to existing graph.

- Example:

```
points(trucks, pch=2, col=2)  
lines(trucks, lty=2, col=2)
```



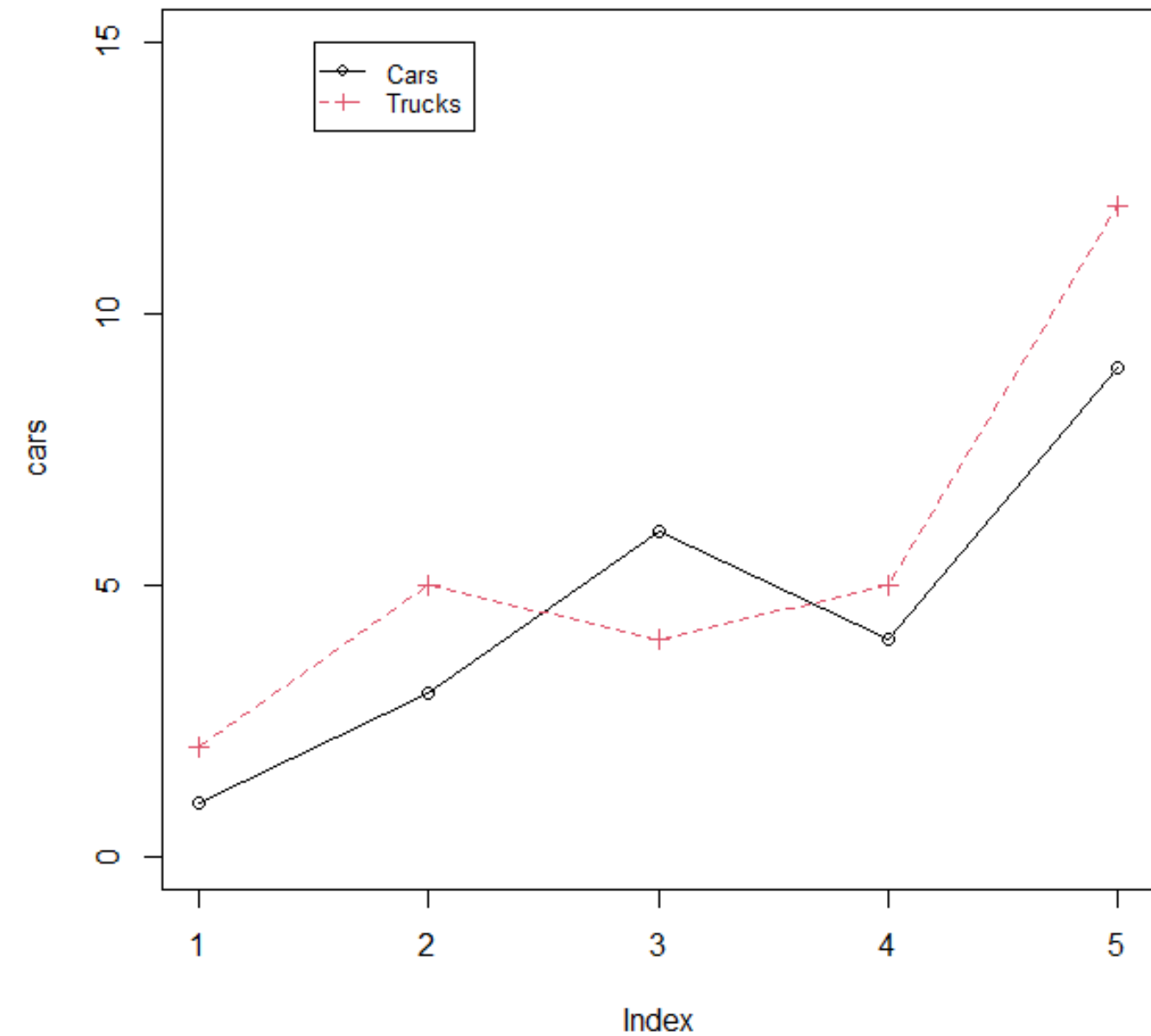


# The legend Function

- Use `legend()` function to add legend for the graph.

- Example:

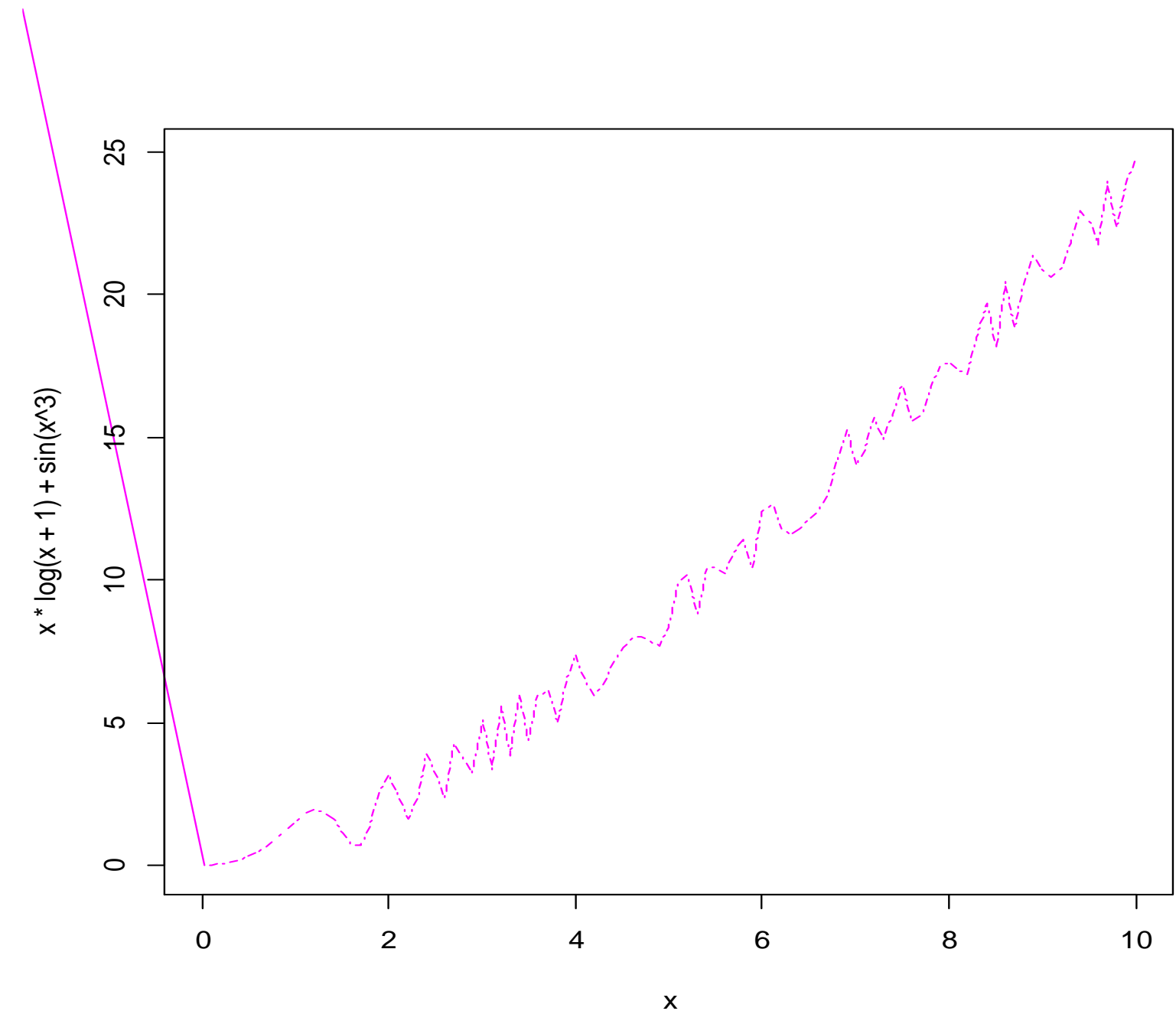
```
legend(1.5, 15, cex=0.8, lty=1:2,  
col=c(1, 2), pch=c(1, 3), c("Cars",  
"Trucks"))
```



# The curve Function

- Plot curves by inserting the equation of a curve in the `curve()` function.
- Similar to the `plot()` function, you can change the colour, font, and so on.
- Example:

```
curve(x*log(x+1)+sin(x^3), from=0,  
to=10,col="magenta",lty=4)
```

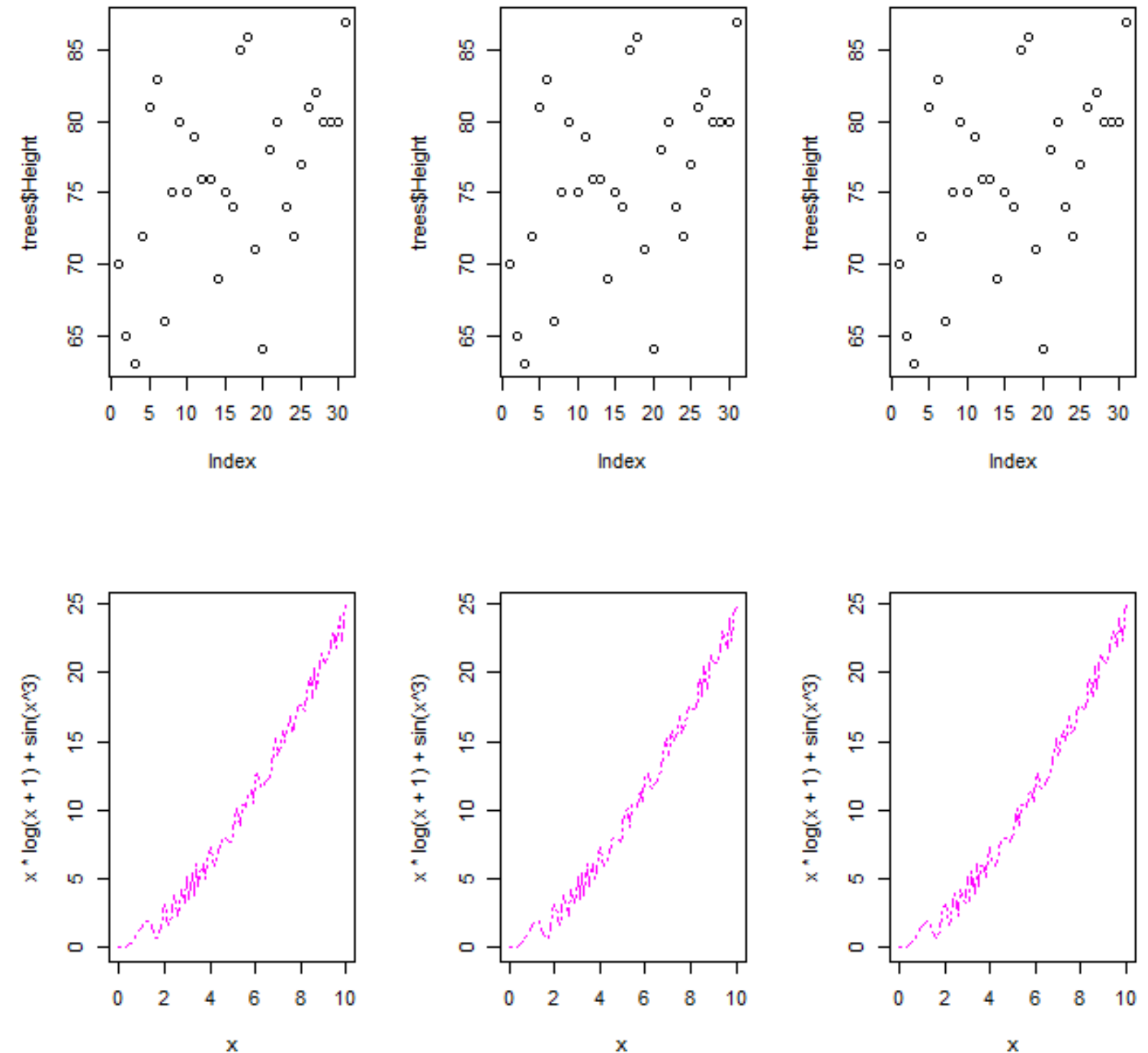


# The `par` Function

- Use `par()` function to plot multiple graphs in one window.

- Example:

```
par(mfrow=c(2,3))
```

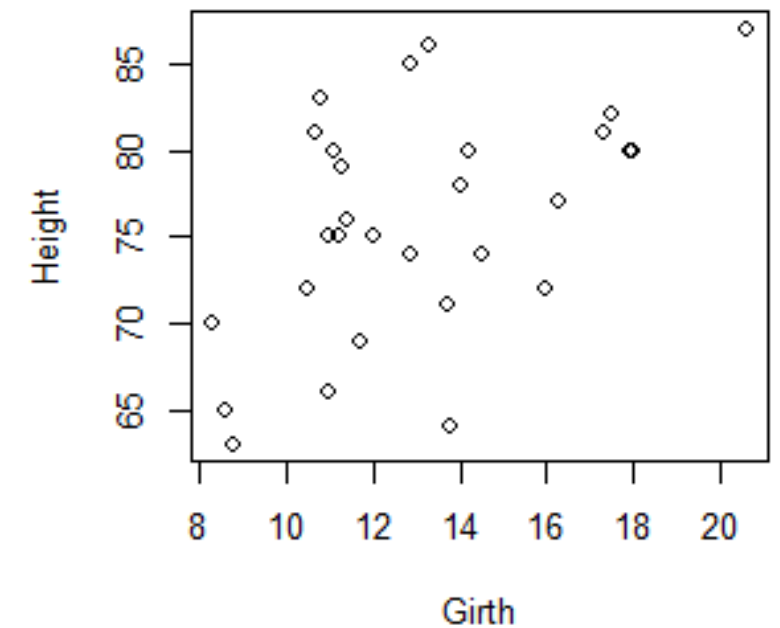
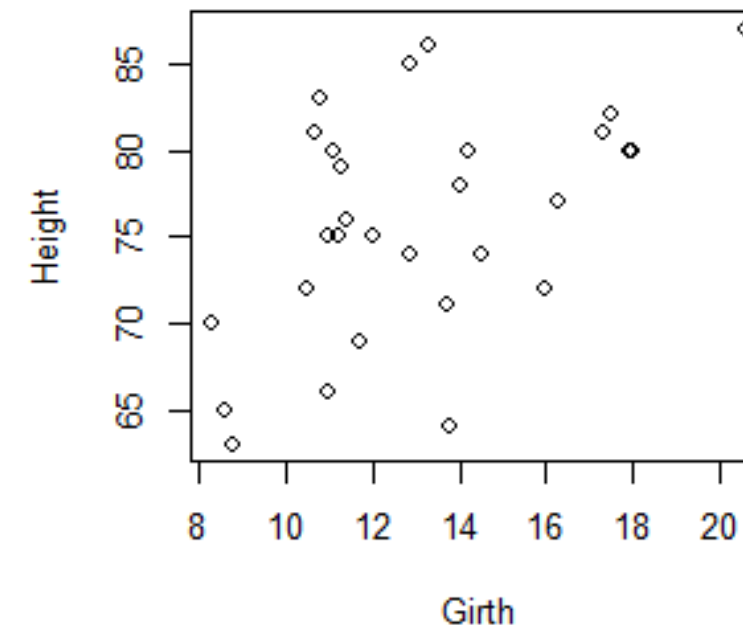
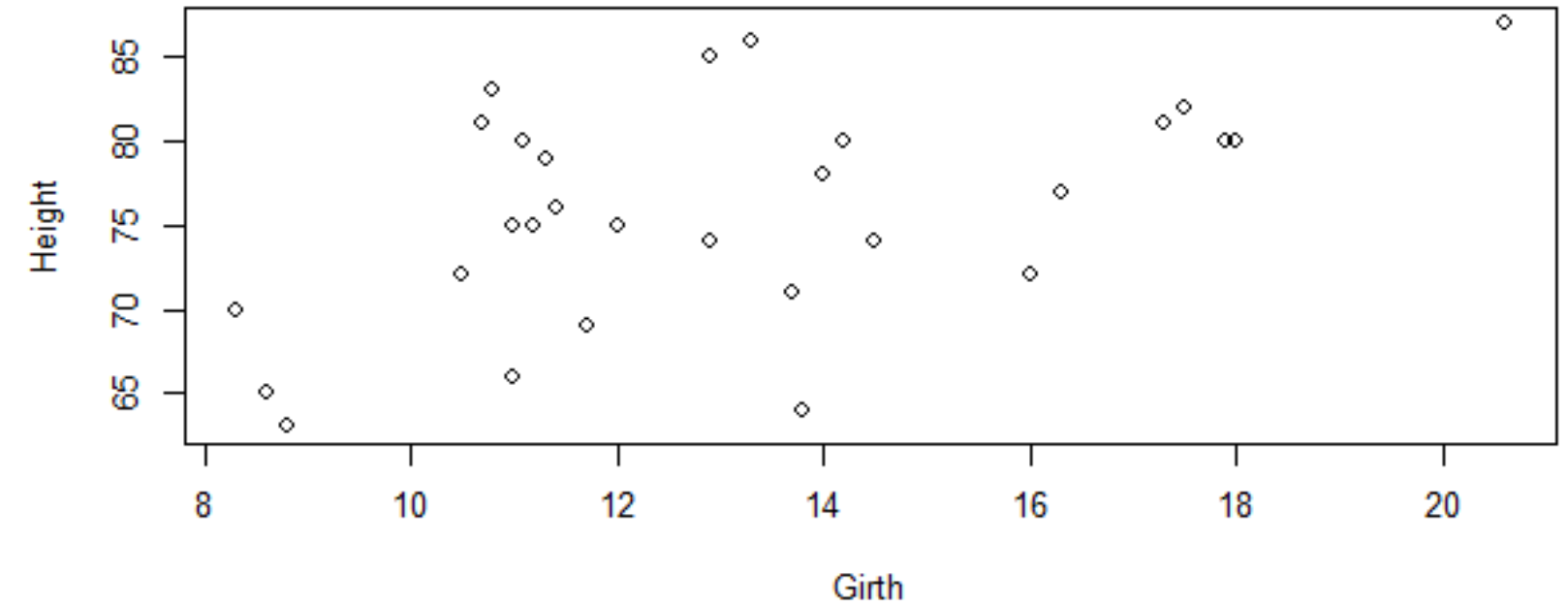


# The layout Function

- Use `layout()` function to plot multiple graphs in one window.

- Example:

```
layout(matrix(c(1,1,2,3),nrow=2,  
byrow=T))
```



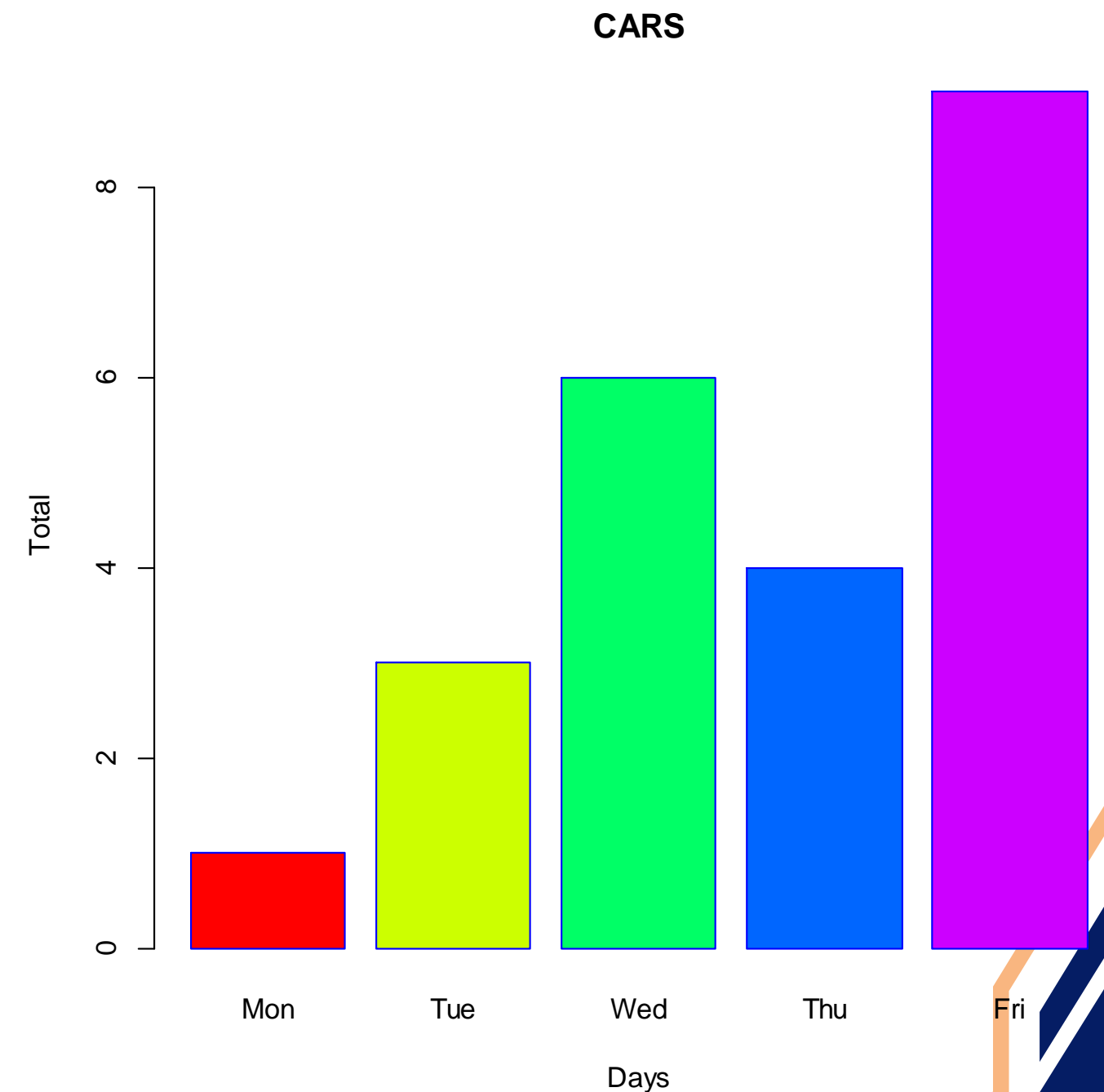
# The `barplot` Function

- You can add titles and labels to the plot like the function `plot()`.
- We can add limits for the x and y axes to the barplot.
- We can add bars to existing barplots by using argument `add=T` and adjusting the space for the plots using argument `space=`
- You can add names for each box in the plot using `names.arg=`
- If we define a variable name for the barplot, it will give us the mid points for the bars.
- The argument `legend=` in the `barplot()` function will provide legend for the bars.

# The barplot Function

- **Example:**

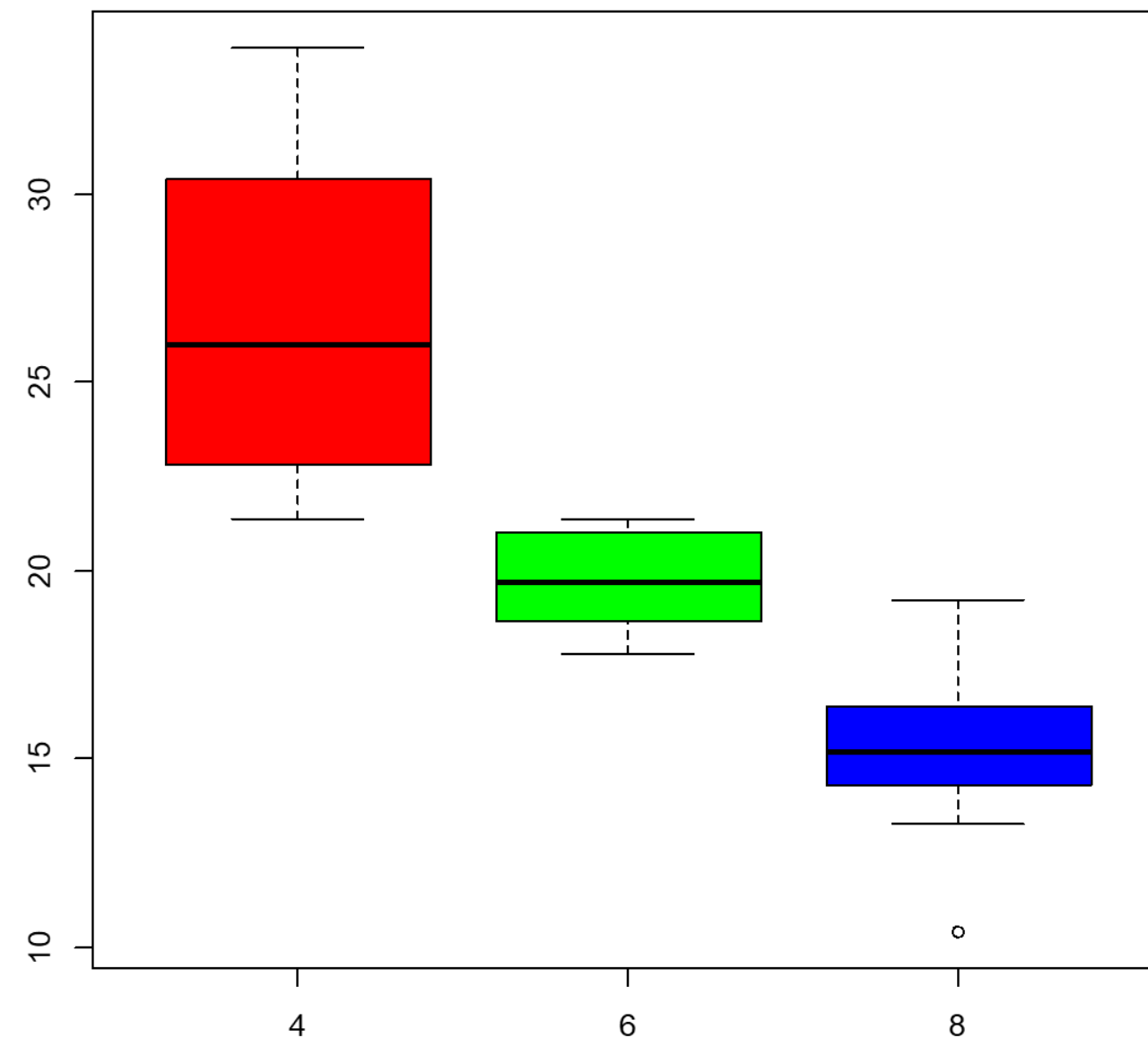
```
barplot(cars, main="CARS", xlab="Days",  
        ylab="Total", border="blue",  
        names.arg=c("Mon", "Tue", "Wed", "Thu", "Fri"),  
        col=rainbow(5))
```



# The `boxplot` Function

- By default, the boxplots is in the vertical direction. You can change its direction by using argument `horizontal=T`.
- You can use it on continuous data as well.
- Example:

```
boxplot(mtcars$mpg~mtcars$cyl,  
col=c("red", "blue", "green"))
```

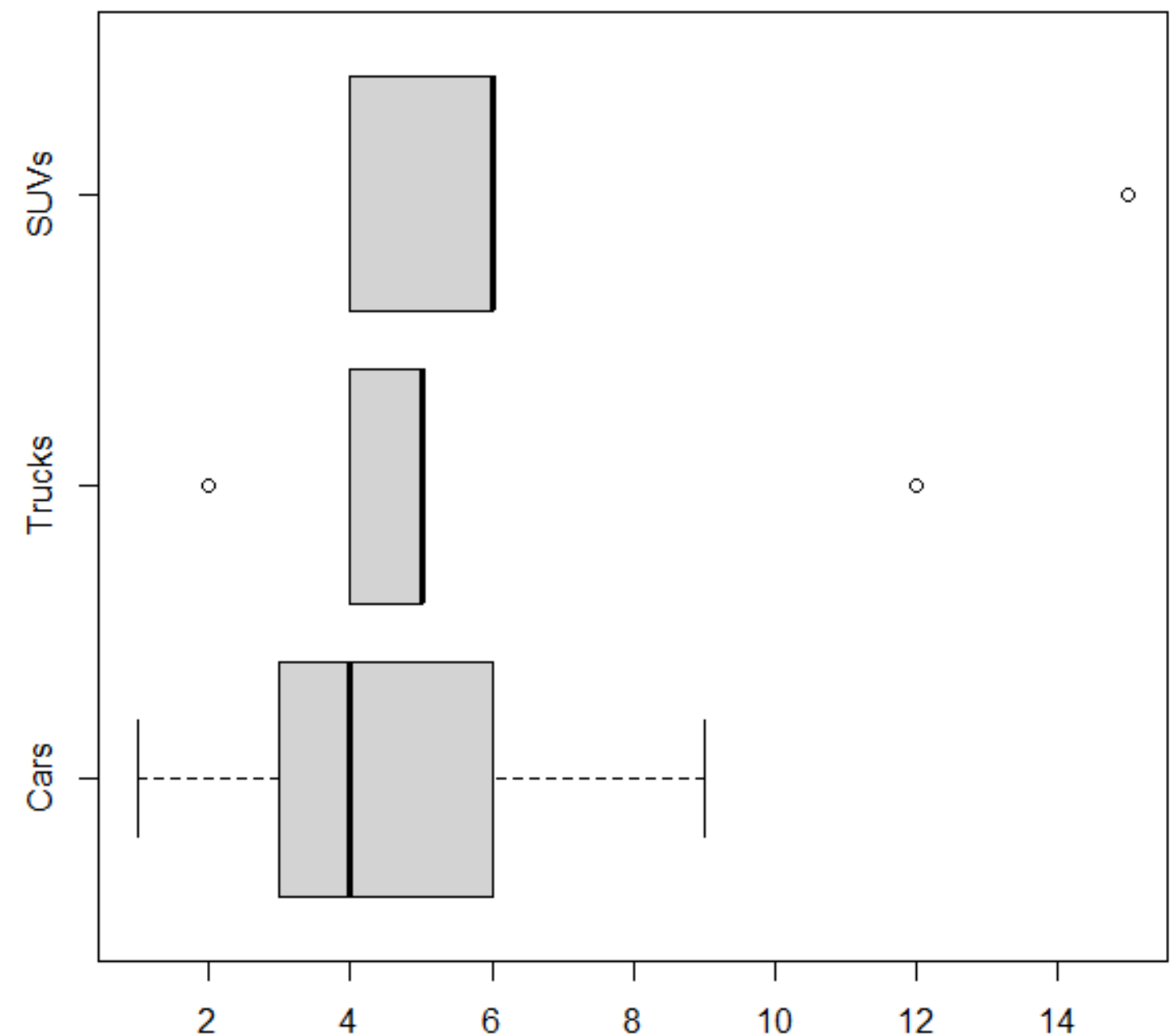




# The `boxplot` Function

- **Example:**

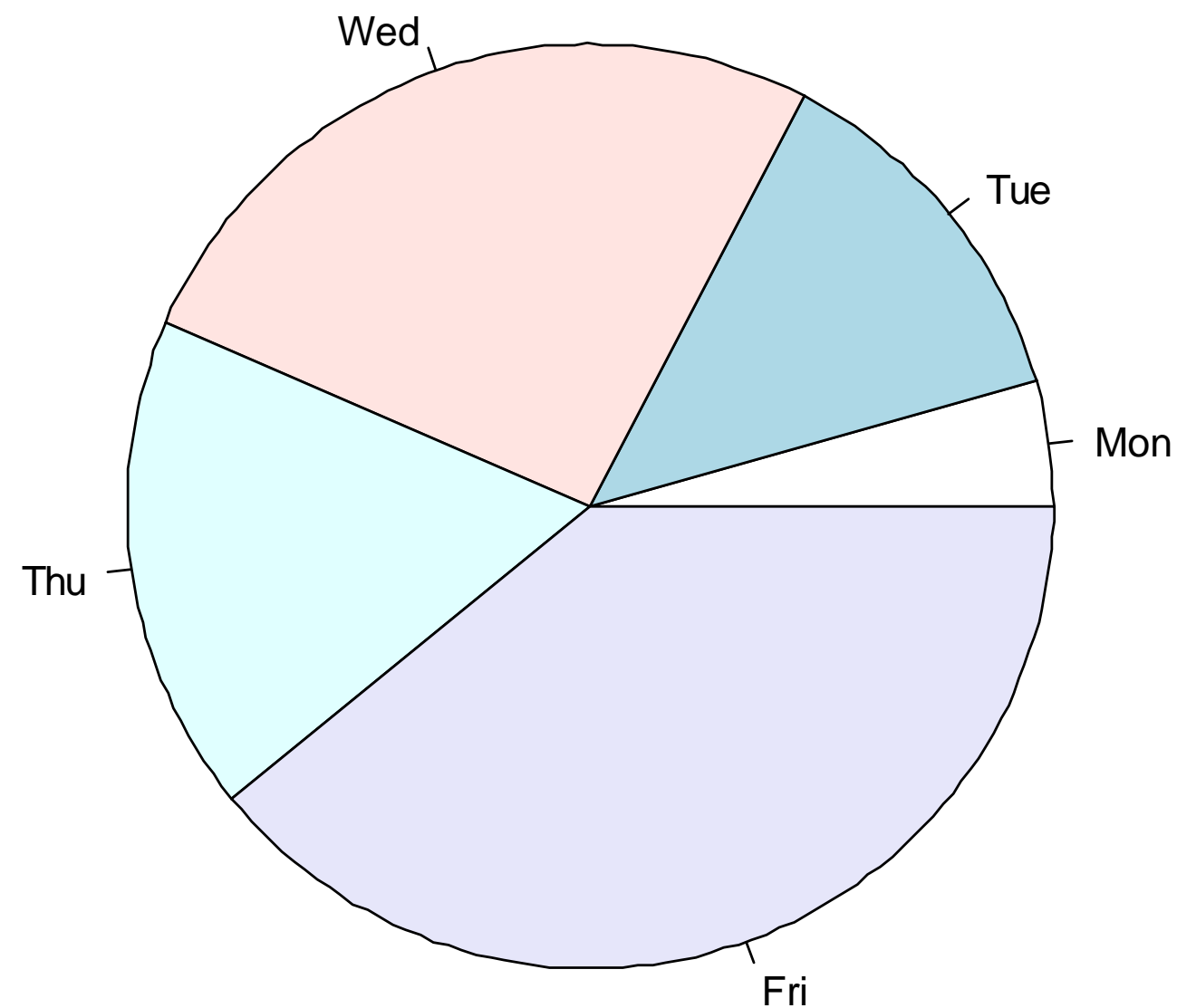
```
boxplot(cars, trucks,  
        suvs, names=c("Cars", "Trucks", "SUVs"),  
        horizontal=T)
```



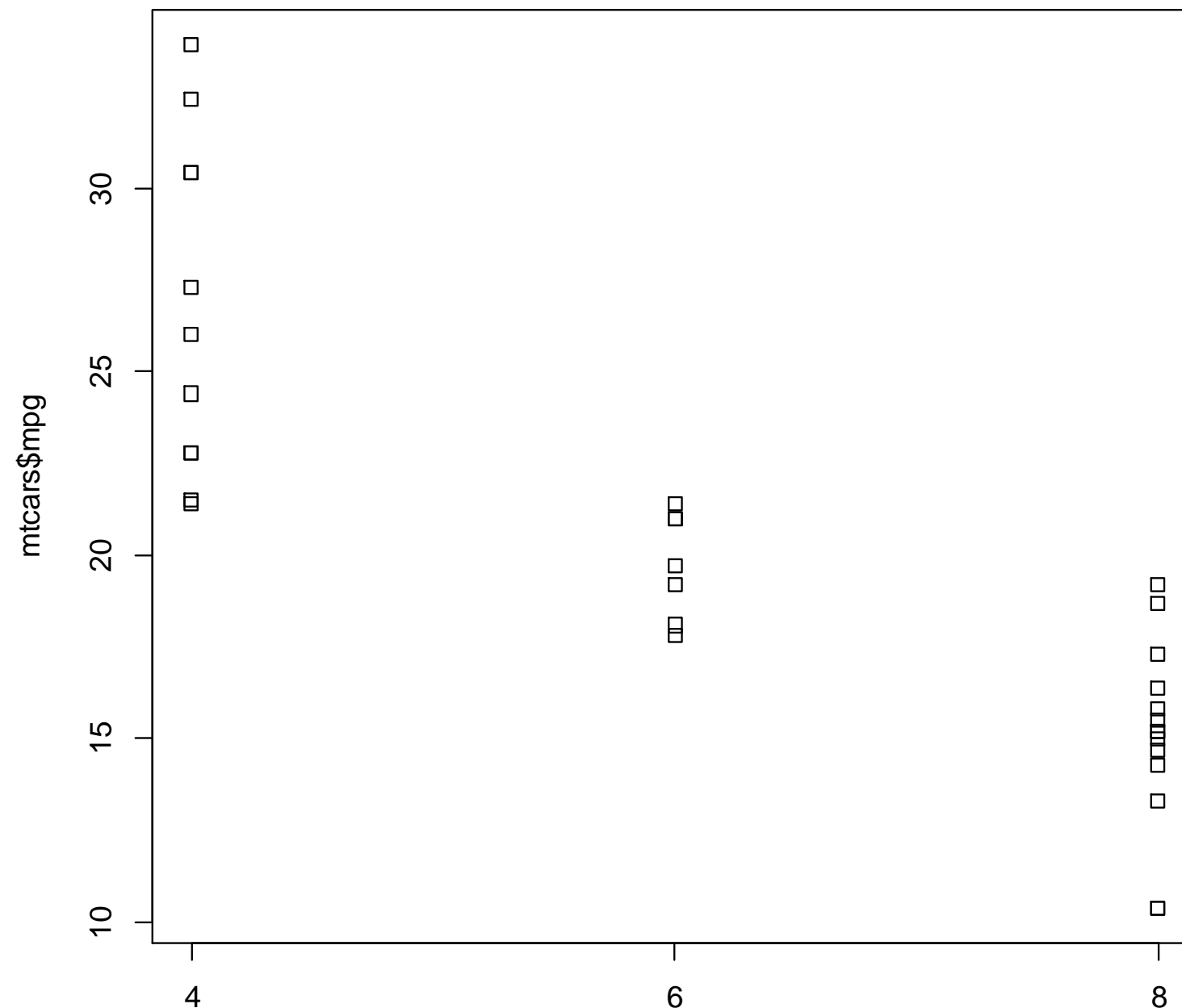
# The `pie` Function

- For two-dimensional pie chart.
- Example:

```
pie(cars, labels=c("Mon", "Tue", "Wed", "Thu", "Fri"))
```



# The stripchart Function




- By default, the stripcharts is in the horizontal direction. You can change its direction by using argument `vertical=T`.
- You can use it on continuous data as well.
- Example:  

```
stripchart(mtcars$mpg~mtcars$cyl,  
vertical=T)
```



# Example

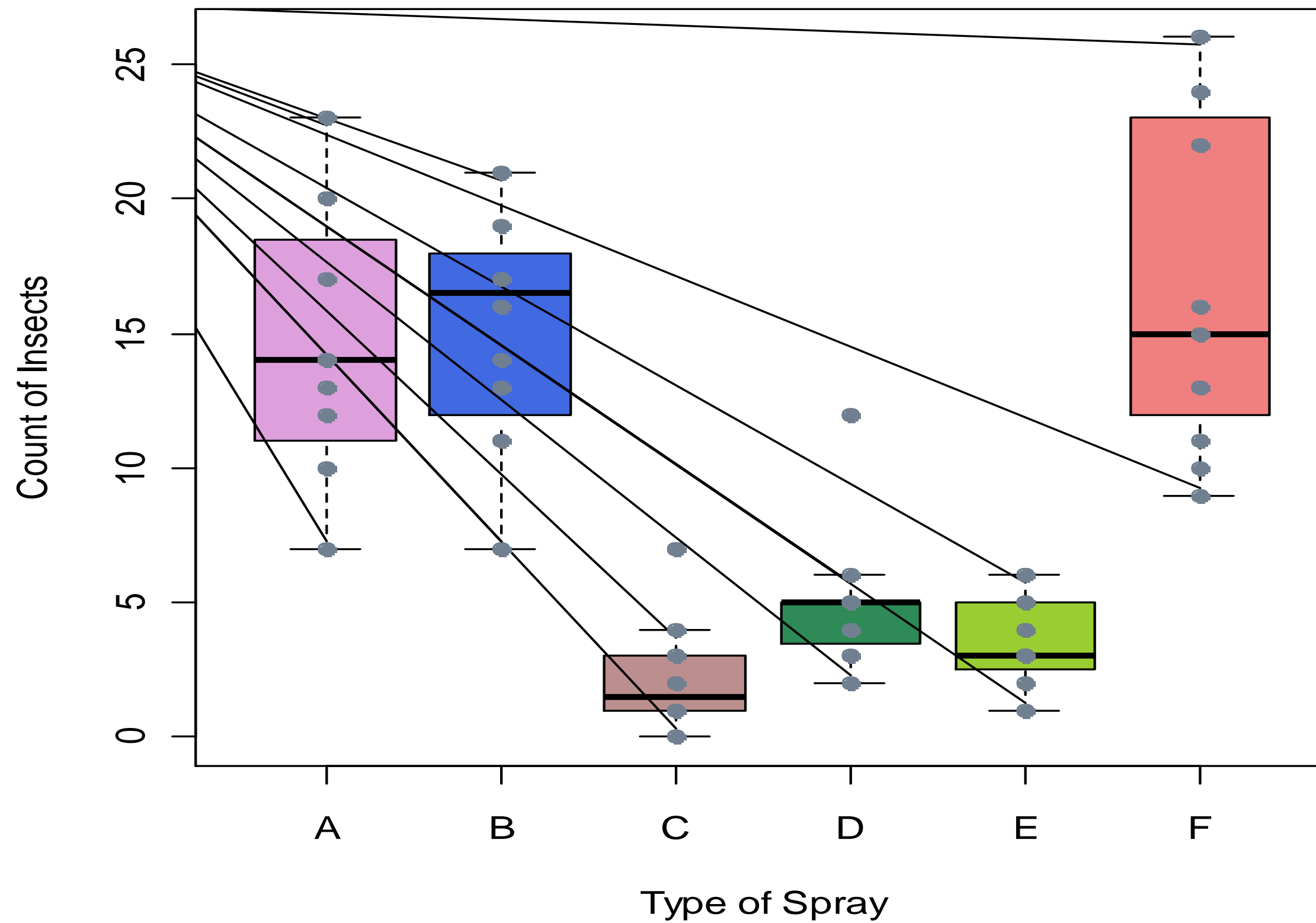


- Use data insect spray (InsectSprays) from available datasets in R.
  - Create boxplots for each type of spray with the colours for the boxplot A, B, C, D, E and F are plum, royalblue, rosybrown, seagreen, yellowgreen and lightcoral respectively.
  - Add a stripchart to each boxplot with point style equal to 19 and the colour slategray.
  - Give labels to the plot.
- 

# Example

```
data (InsectSprays)
attach (InsectSprays)
boxplot (count~spray,
col=c ("plum", "royalblue", "rosybrown", "seagreen",
"yellowgreen", "lightcoral"),
xlab="Type of Spray", ylab="Count of Insects")
stripchart (count~spray, add=T, vertical=T, pch=19,
col="slategray")
```

# Example



# THANK YOU

