uni-venda ggplot2 tutorial

andy south

2016-10-30

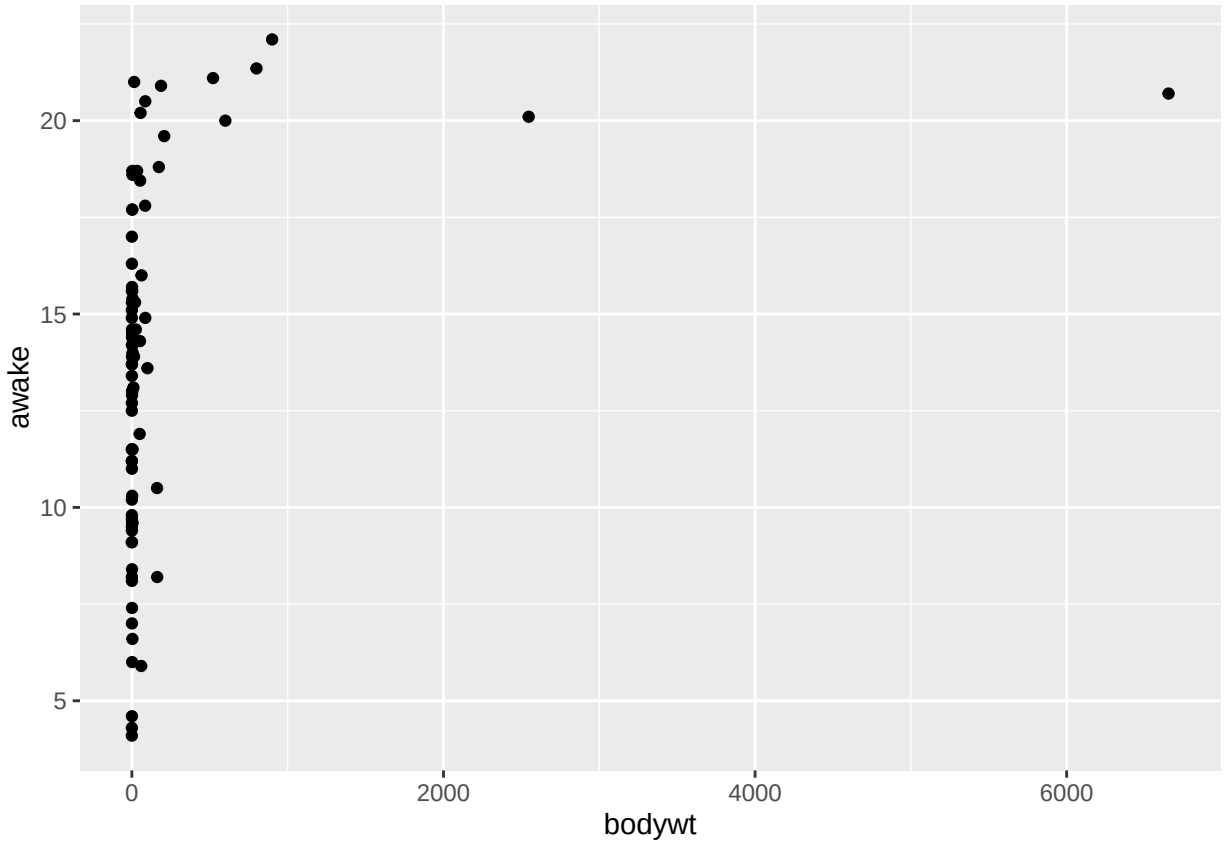
ggplot2 by Hadley Wickham, is a newer alternative to the base graphics available in R. Base graphics are still useful for creating customised plots but we will not look at them here.

A) point plots with ggplot2 first go

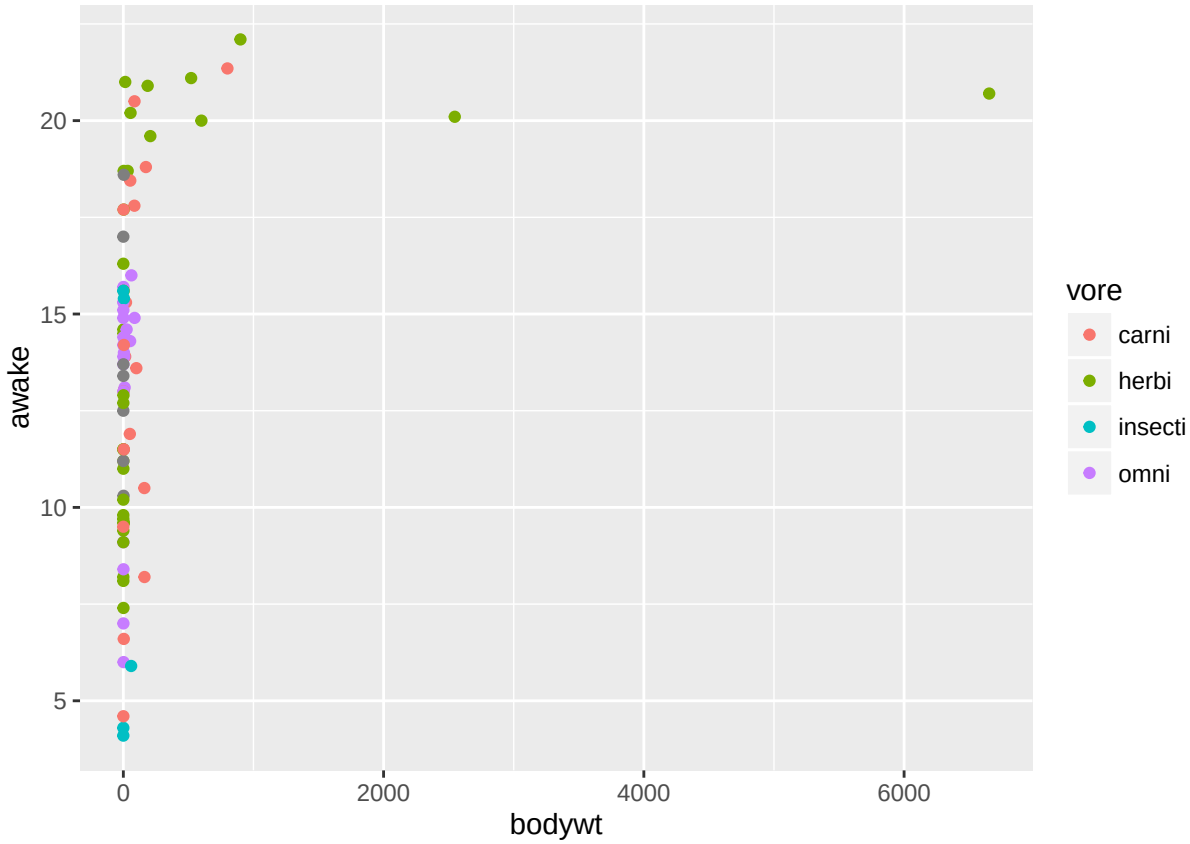
```
# load the package
library("ggplot2")

# load some data on mammal sleep patterns from the package
data(msleep)

# to show the 'str'ucture of the data
str(msleep)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    83 obs. of  11 variables:
##  $ name        : chr  "Cheetah" "Owl monkey" "Mountain beaver" "Greater short-tailed shrew" ...
##  $ genus       : chr  "Acinonyx" "Aotus" "Aplodontia" "Blarina" ...
##  $ vore        : chr  "carni" "omni" "herbi" "omni" ...
##  $ order       : chr  "Carnivora" "Primates" "Rodentia" "Soricomorpha" ...
##  $ conservation: chr  "lc" NA "nt" "lc" ...
##  $ sleep_total : num  12.1 17 14.4 14.9 4 14.4 8.7 7 10.1 3 ...
##  $ sleep_rem   : num  NA 1.8 2.4 2.3 0.7 2.2 1.4 NA 2.9 NA ...
##  $ sleep_cycle : num  NA NA NA 0.133 0.667 ...
##  $ awake       : num  11.9 7 9.6 9.1 20 9.6 15.3 17 13.9 21 ...
##  $ brainwt     : num  NA 0.0155 NA 0.00029 0.423 NA NA NA 0.07 0.0982 ...
##  $ bodywt      : num  50 0.48 1.35 0.019 600 ...
```

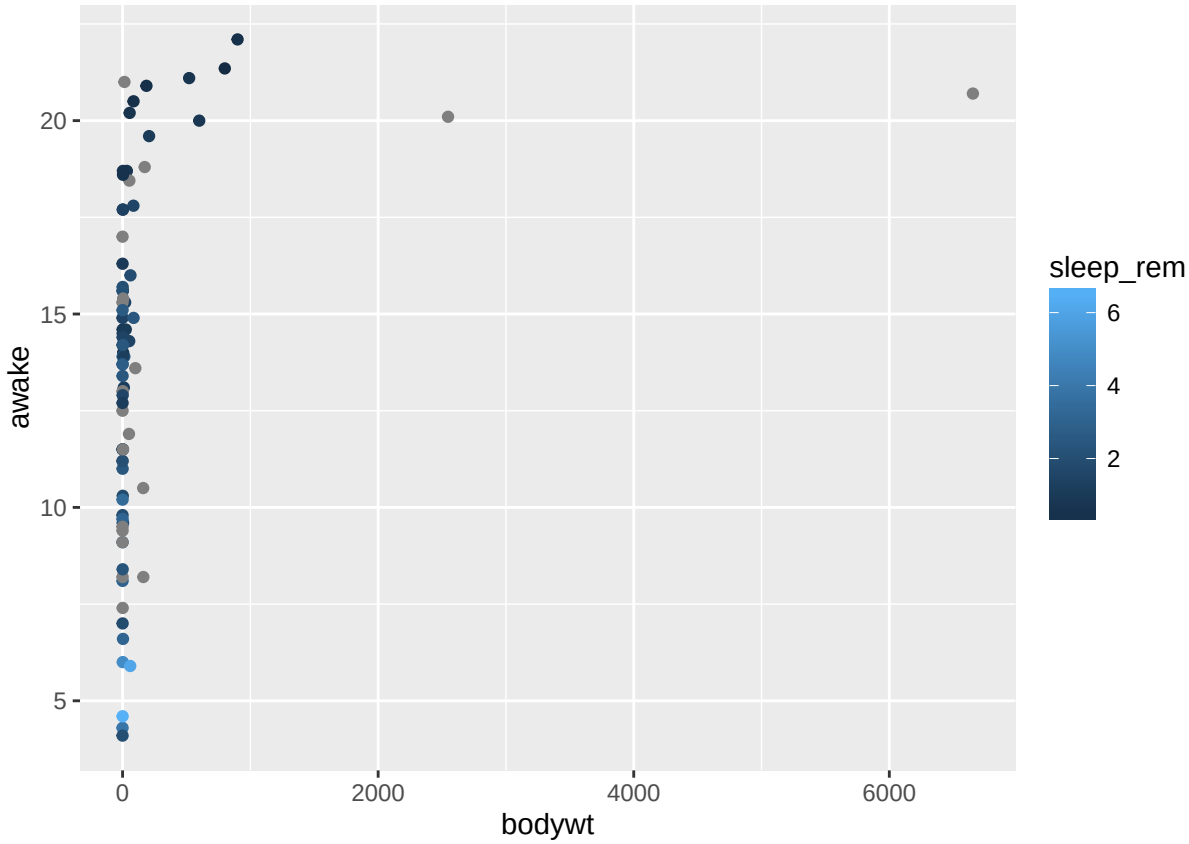
```
# first plot
ggplot(data = msleep, aes(x = bodywt, y = awake)) +
    geom_point()
```



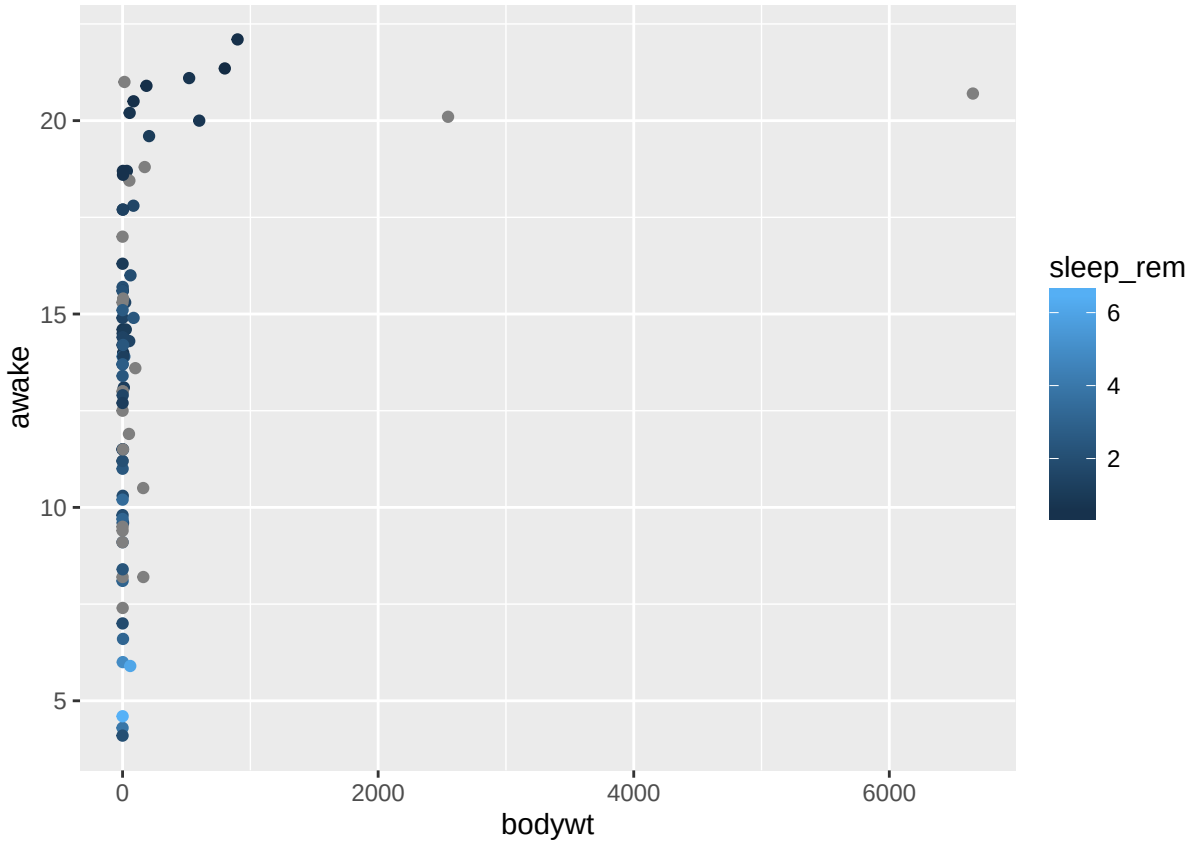

```
# aes (for aesthetic) sets up links between the data and aspects of the plot
# + adds layers to the plot, in this case points
# geom* stands for the geometric objects you can add to a plot
# bodywt and awake are columns within the data

# add an extra column to the aes
ggplot(data = msleep, aes(x = bodywt, y = awake, col = vore)) +
    geom_point()
```



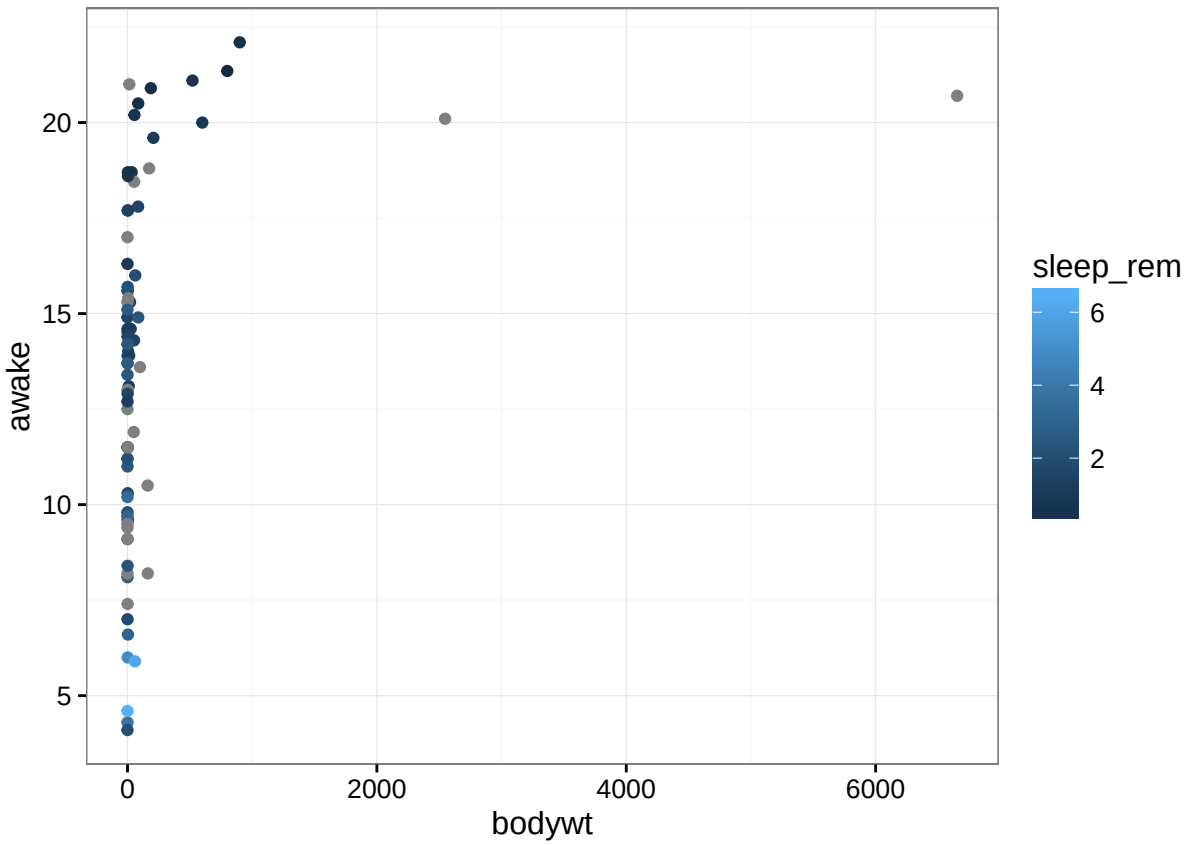

```
# if you change col= to a column that is numeric it changes the legend
ggplot(data = msleep, aes(x = bodywt, y = awake, col = sleep_rem)) +
    geom_point()
```




```
# you can store the plot as an object (in this case called p) which can make modifying it easier
p <- ggplot(data = msleep, aes(x = bodywt, y = awake, col = sleep_rem))
p <- p + geom_point()
# type the variable name to display the plot
p
```




```
# use themes to change the overall appearance of plots
# note that when typing this in RStudio it will show you the options once you've typed th...
p + theme_bw()
```



A1) Exercise

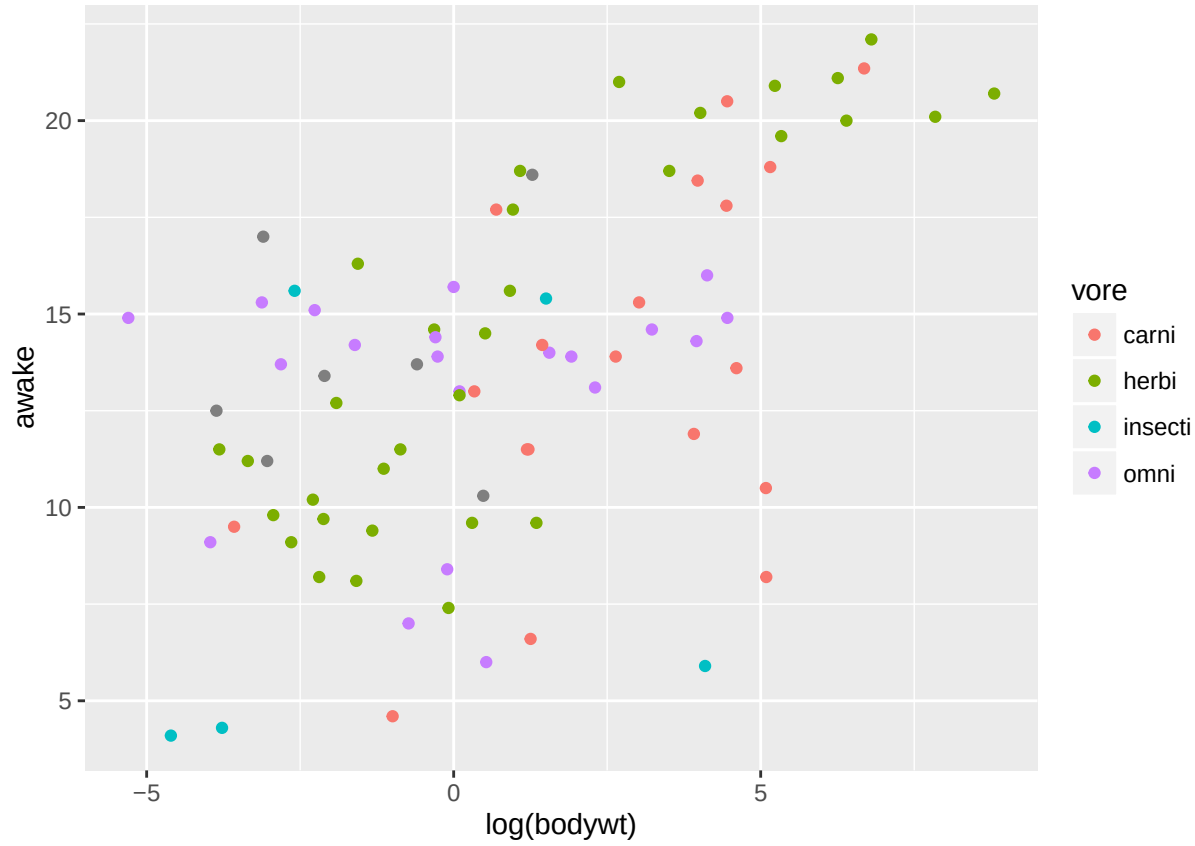
Have a quick look at the online ggplot2 documentation here : http://docs.ggplot2.org/current/ Particularly look at geom_point and scroll down to the examples.

From within R you can get similar help but without the helpful plots by : `?geom_point`

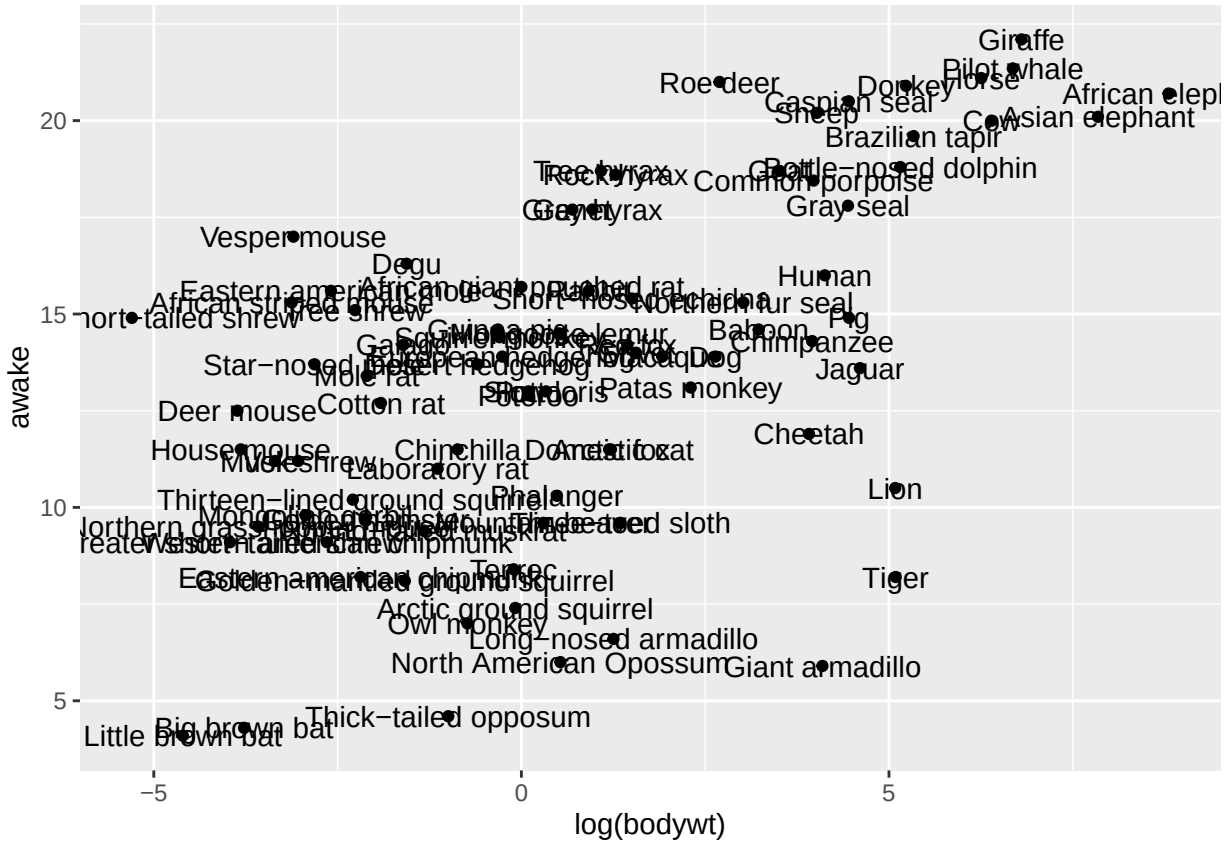
Try modifying a point plot with the msleep data.

B) point plots, modifying axes, label points, divide into subplots

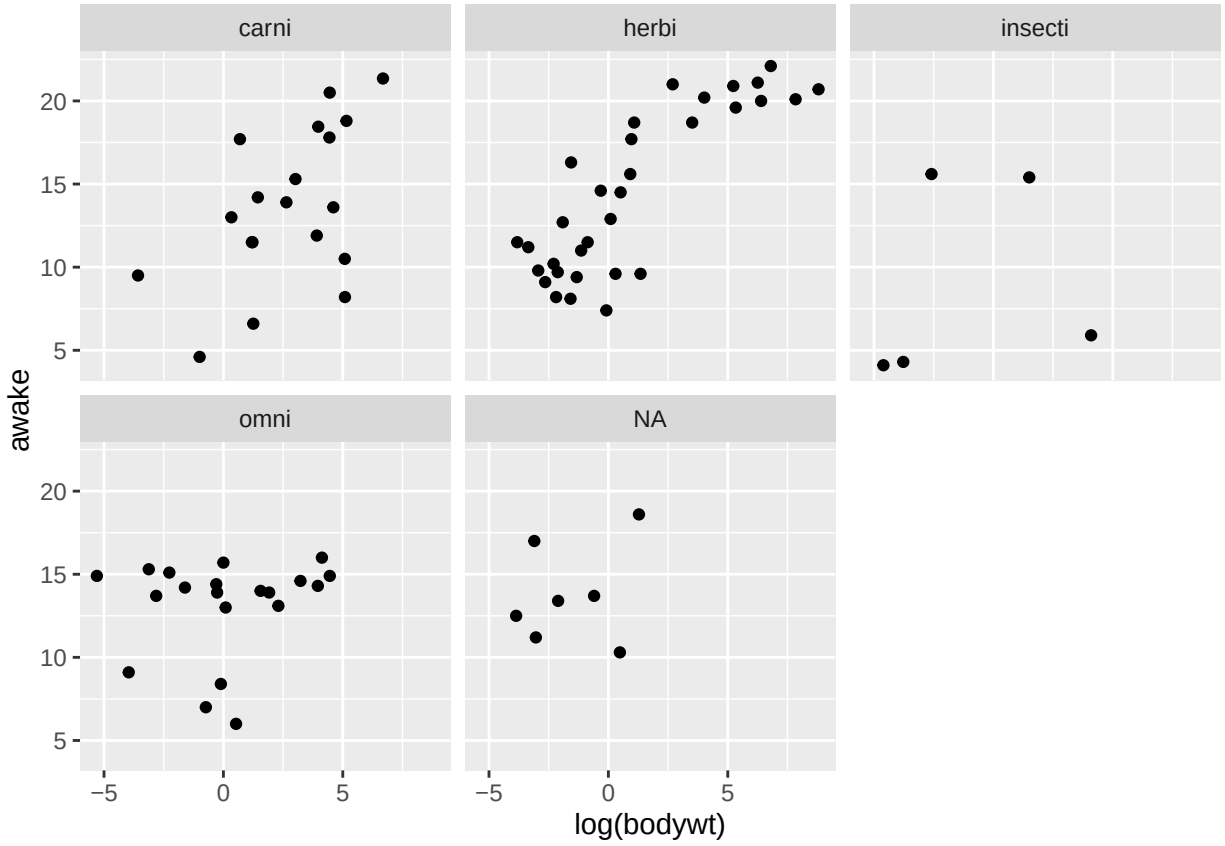
```
# we can log bodywt to space the points out more
p <- ggplot(data = msleep, aes(x = log(bodywt), y = awake, col = vore)) +
     geom_point()
p
```




```
# label the points for data exploration
p <- ggplot(data = msleep, aes(x = log(bodywt), y = awake, label = name)) +
     geom_point() +
     geom_text()
p
```




```
# divide into subplots using facet_wrap()
p <- ggplot(data = msleep, aes(x = log(bodywt), y = awake, label = name)) +
    geom_point() +
    facet_wrap(~vore)
p
```

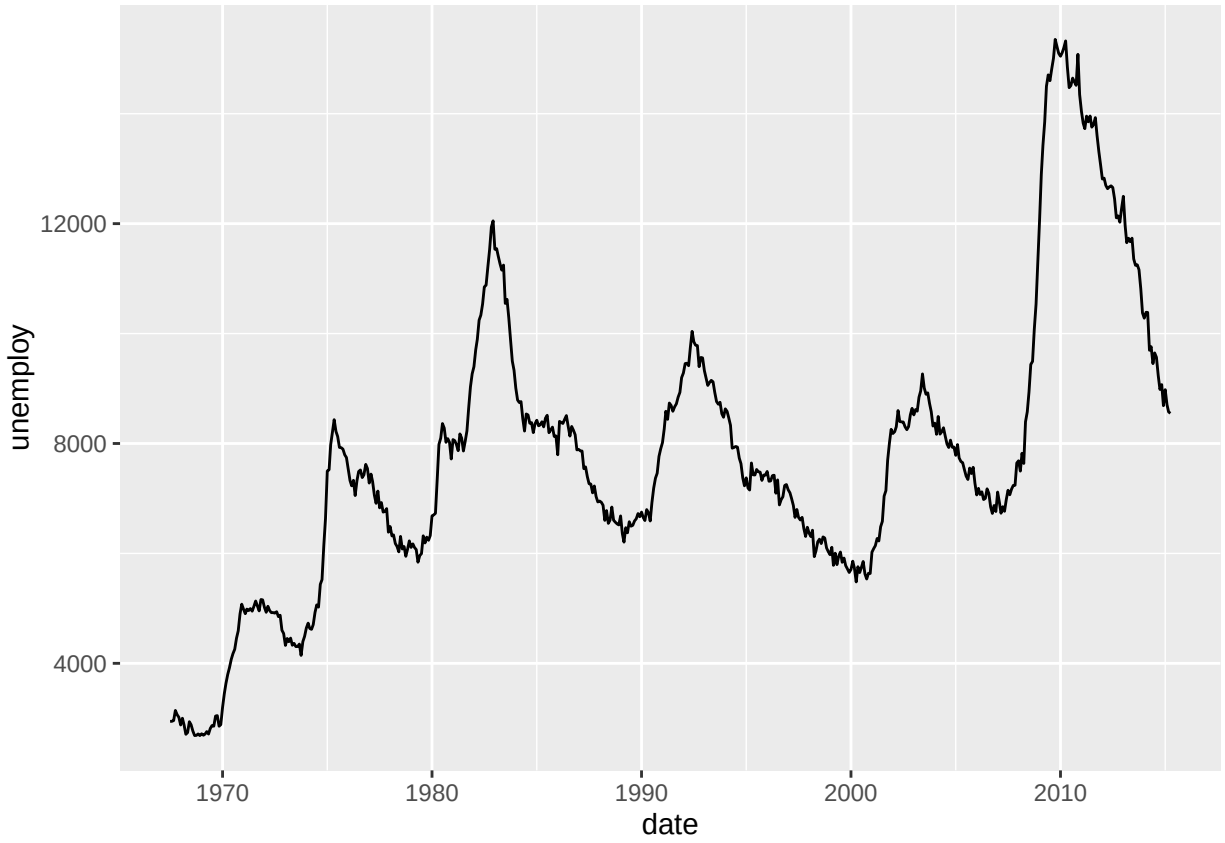


C) line plots with ggplot2

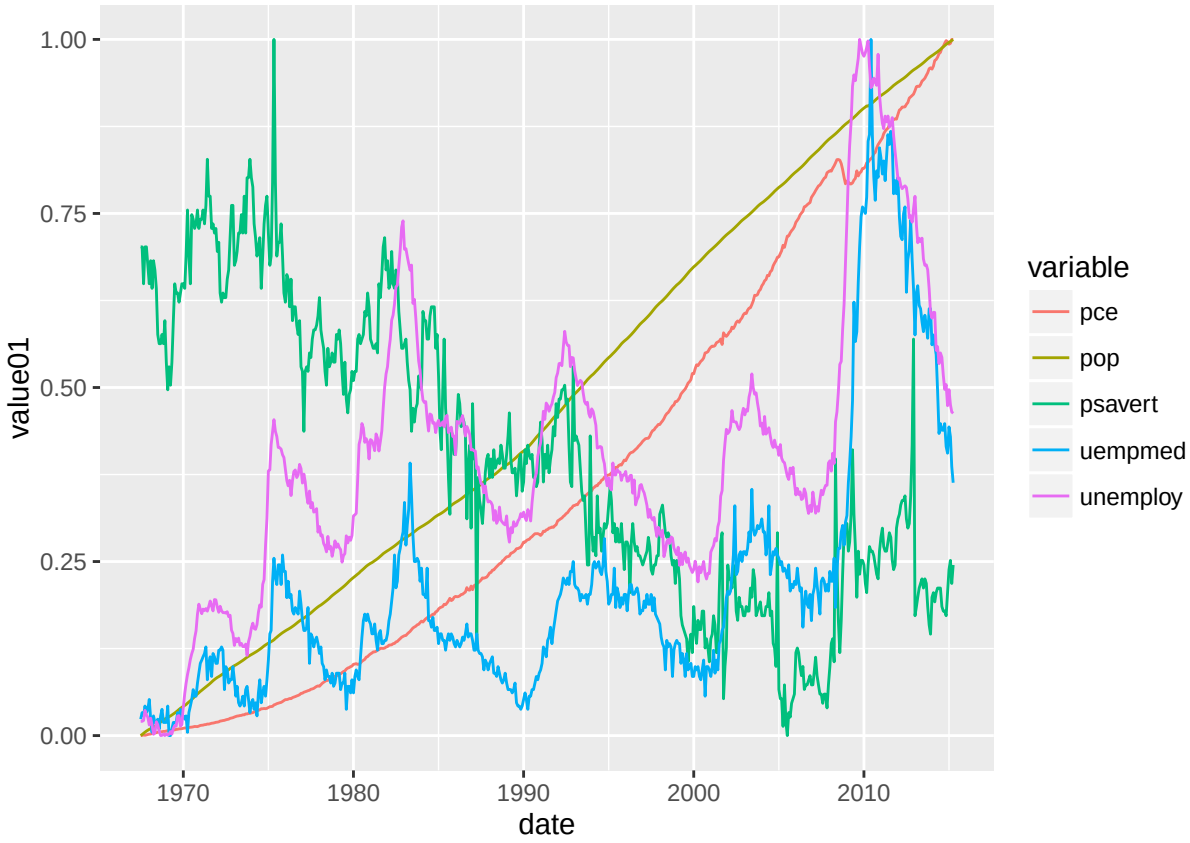
```
#load some time-series data from ggplot2
data(economics)

# to show the 'str'ucture of the data
str(economics)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    574 obs. of  6 variables:
##  $ date    : Date, format: "1967-07-01" "1967-08-01" ...
##  $ pce     : num  507 510 516 513 518 ...
##  $ pop     : int  198712 198911 199113 199311 199498 199657 199808 199920 200056 200208 ...
##  $ psavert : num  12.5 12.5 11.7 12.5 12.5 12.1 11.7 12.2 11.6 12.2 ...
##  $ uempmed : num  4.5 4.7 4.6 4.9 4.7 4.8 5.1 4.5 4.1 4.6 ...
##  $ unemploy: int  2944 2945 2958 3143 3066 3018 2878 3001 2877 2709 ...
```

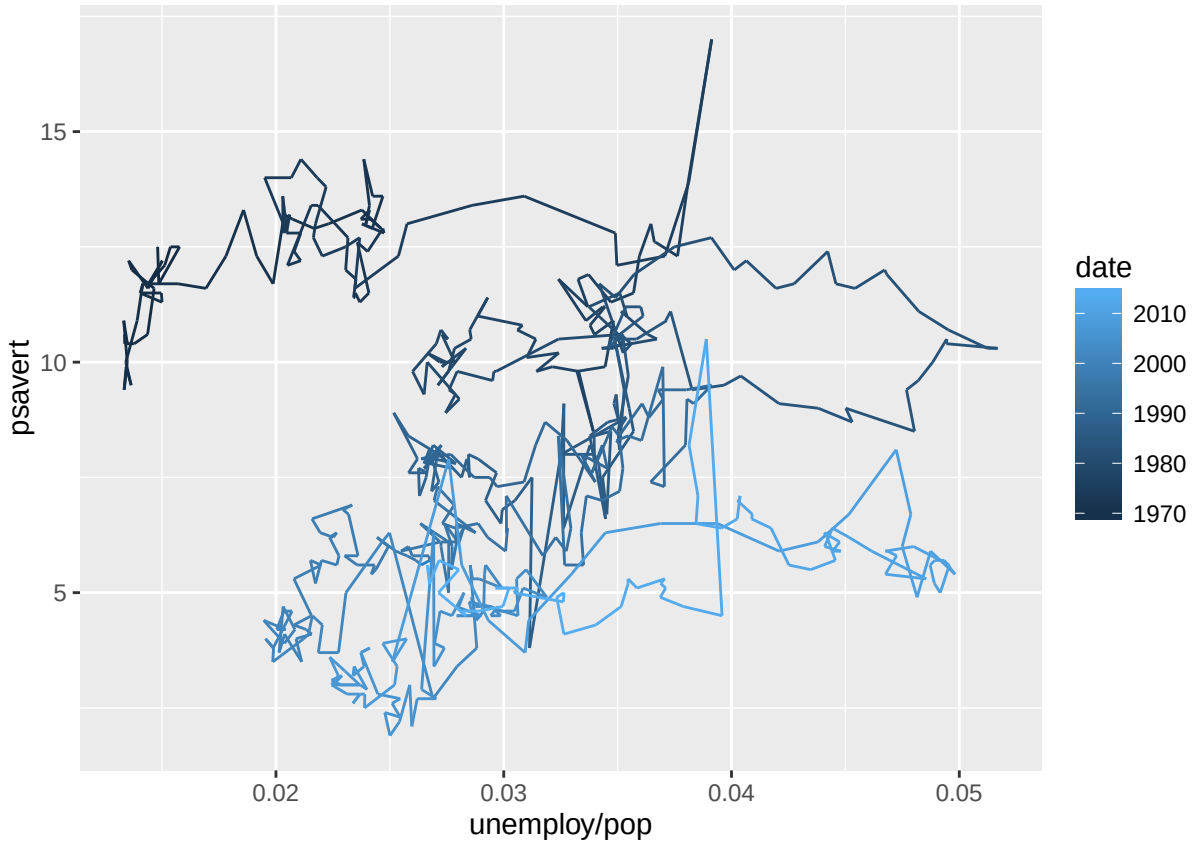
```
# geom_line()
ggplot(economics, aes(date, unemploy)) + geom_line()
```




```
#note that above aes(date, unemploy) assumes that these are to set x & y respectively
# this : aes(x=date, y=unemploy) would give an identical result

# another dataset economics_long is in 'long' format
# instead of having one column for each attribute (e.g. pop and unemploy)
# it has a column (variable) which has a repeated factor values for each date
# this allows us to create a plot with multiple lines

data(economics_long)

ggplot(economics_long, aes(date, value01, colour = variable)) +
  geom_line()
```




```
#note that for each variable the value01 column has been scaled between 0 & 1 to fit on same scale


# geom_path can be used to look at the relationship of 2 variables over time
# unemployment and savings rate
p <- ggplot(economics, aes(unemploy/pop, psavert))
p + geom_path(aes(colour = date))
```




```
# note how the unemployment rate is calculated by unemploy/pop within the aes call
# you can do other calculations between columns in there
```

C1) Exercise

Try using facet_wrap with the economics_long line plot to get subplots for each variable.

...

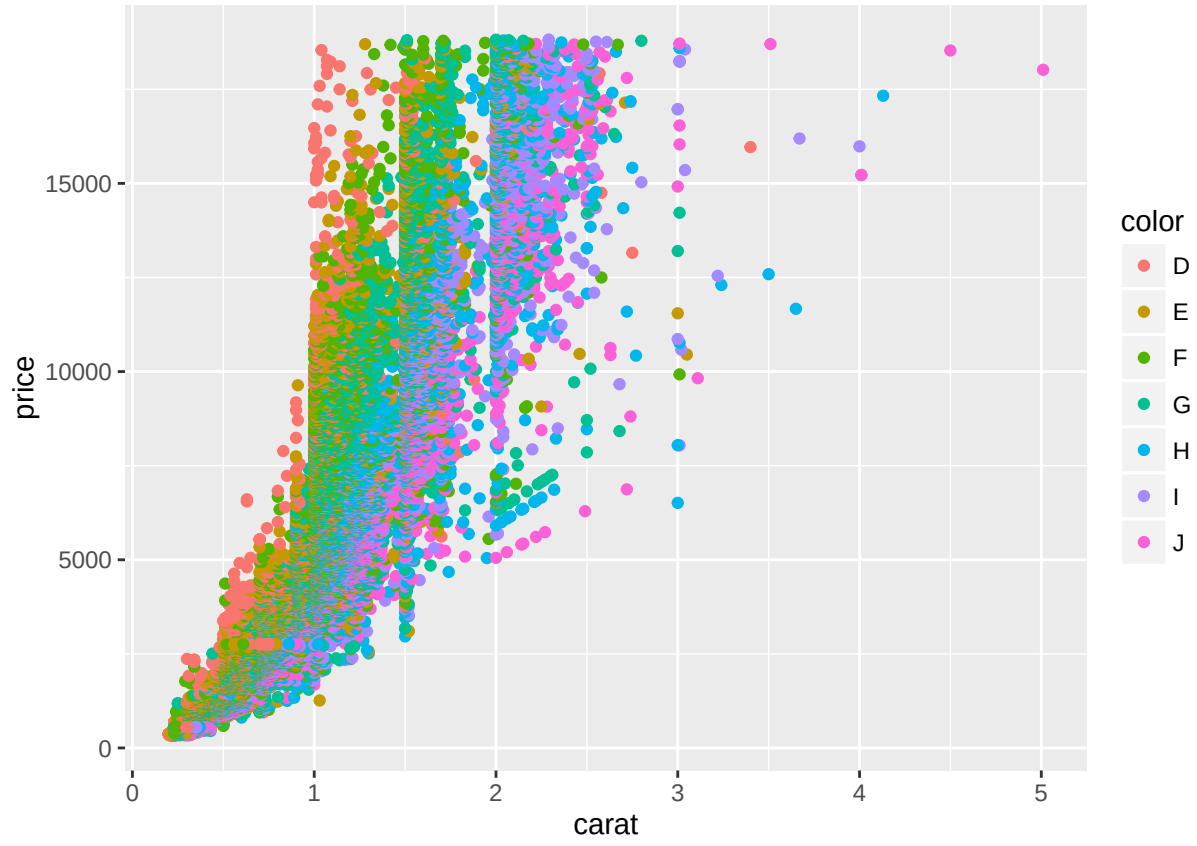
D) bigger (but still small!) data, histograms etc.

```
# the diamonds dataset has > 50K diamonds (rows) in it

data(diamonds)
str(diamonds)
```

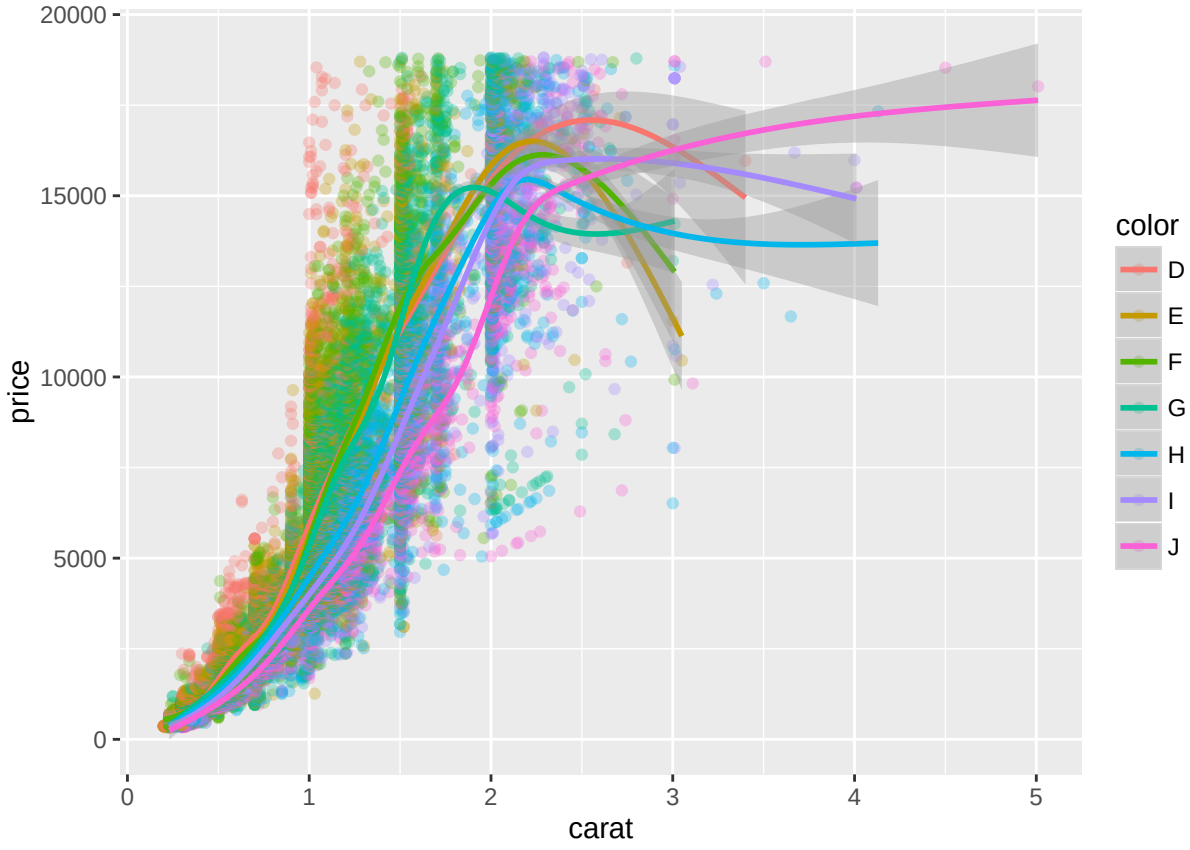
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    53940 obs. of  10 variables:
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ depth  : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num  55 61 65 58 58 57 57 55 61 61 ...
```

```
##  $ price  : int  326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

```
# a scatter plot is not very informative
p <- ggplot(diamonds, aes(x = carat, y = price, col = color))
p + geom_point()
```

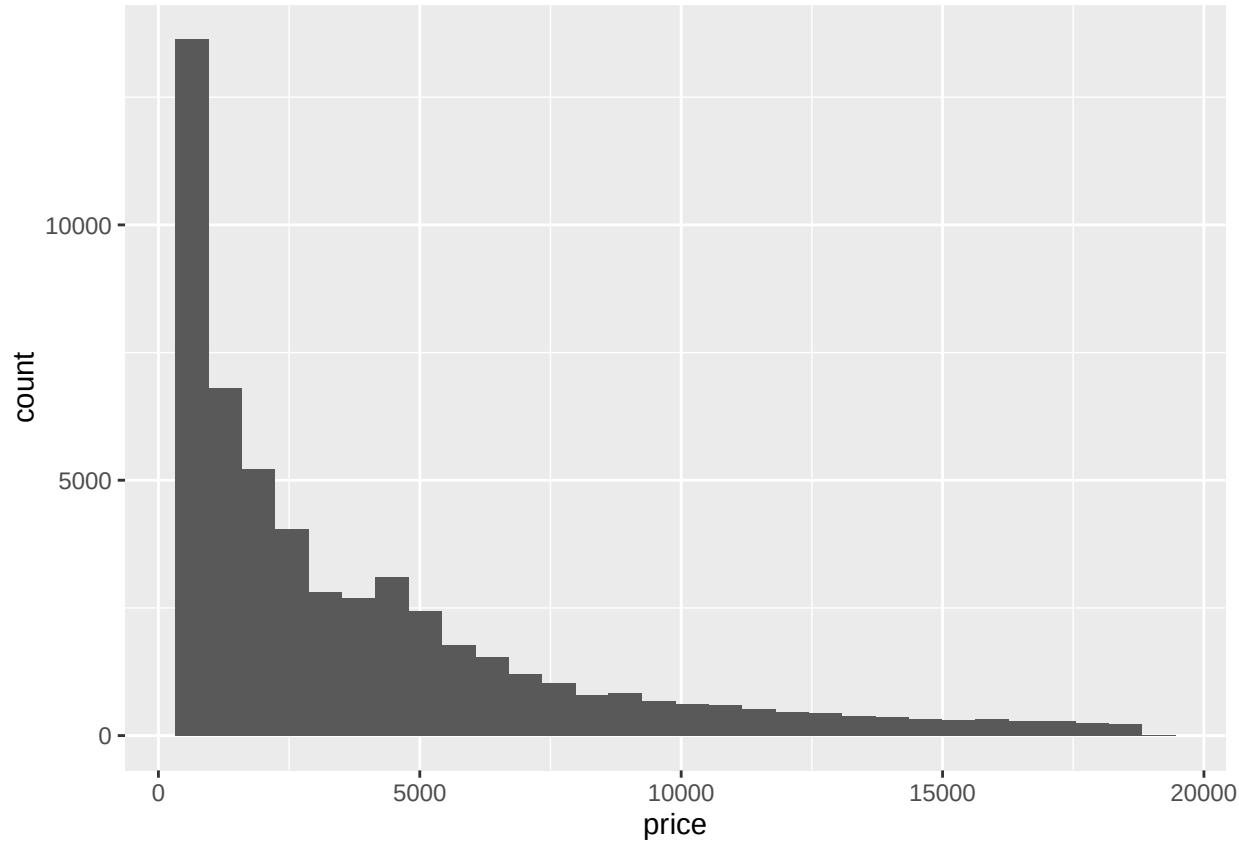



```
# setting the alpha (transparency) level to < 1 can improve things but not much
p <- ggplot(diamonds, aes(x = carat, y = price, col = color))
p <- p + geom_point(alpha=0.3)

# geom_smooth adds a smoothed conditional mean which can help reveal patterns
p + geom_smooth()
```




```
# geom_histogram can be used to look at distributions
p <- ggplot(data=diamonds) + geom_histogram(aes(x=price))
p
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# most of the diamonds are cheap ! (relatively)
```

D1) Exercise

Use any of your ggplot skills to find out something interesting about diamonds . . .

. . .

Acknowledgements :

Thankyou to all the package developers on whose work this tutorial is based.