ggplot2: The basics

Adam Wespiser

November 4, 2014

This course includes both example code, and exercises for you to apply the concepts yourself. Before starting, please make sure the following libraries are loaded. If you cannot load a library, you probably don't have it, and should use the

```
install.packages()
```

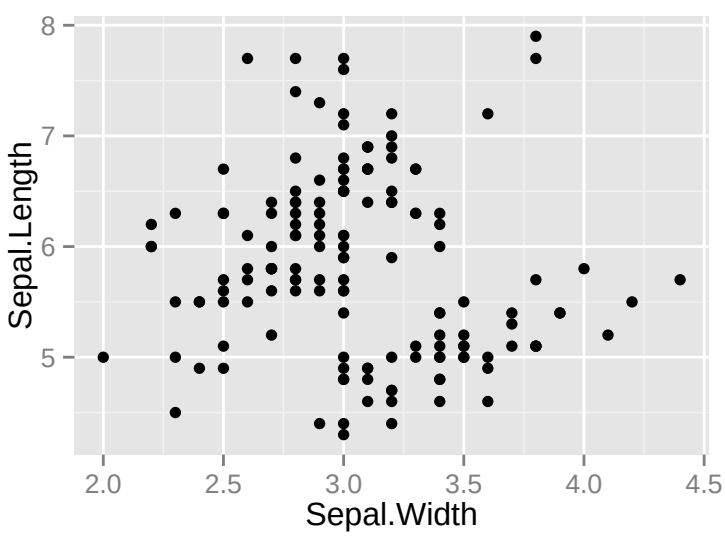
function to load them.

```
source("http://bib.umassmed.edu/~wespisea/rCourse/adamGgplot2/answers.R")
library(ggplot2)
library(plyr)
library(grid)
library(reshape2)
library(gridExtra)
library(MASS)
library(HistData)
```

Load the datasets...

Scatter Plots

```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length)) + geom_point()
```



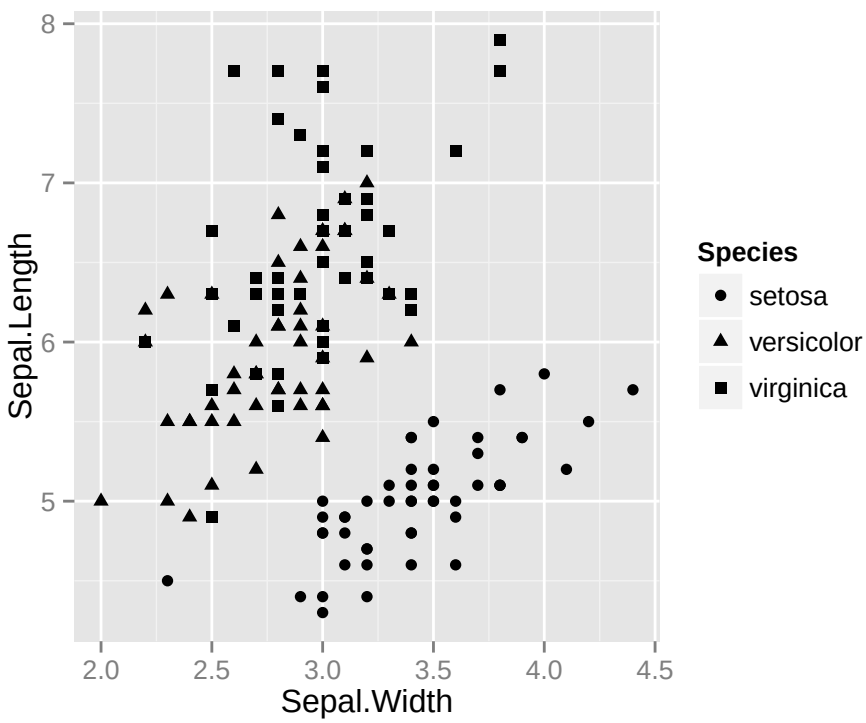
Question 1

Use the women dataset to plot average heights and weights for american women.
First, check the contents of the dataset using `str(women)`.
Check the answer by `print(answer1)`.

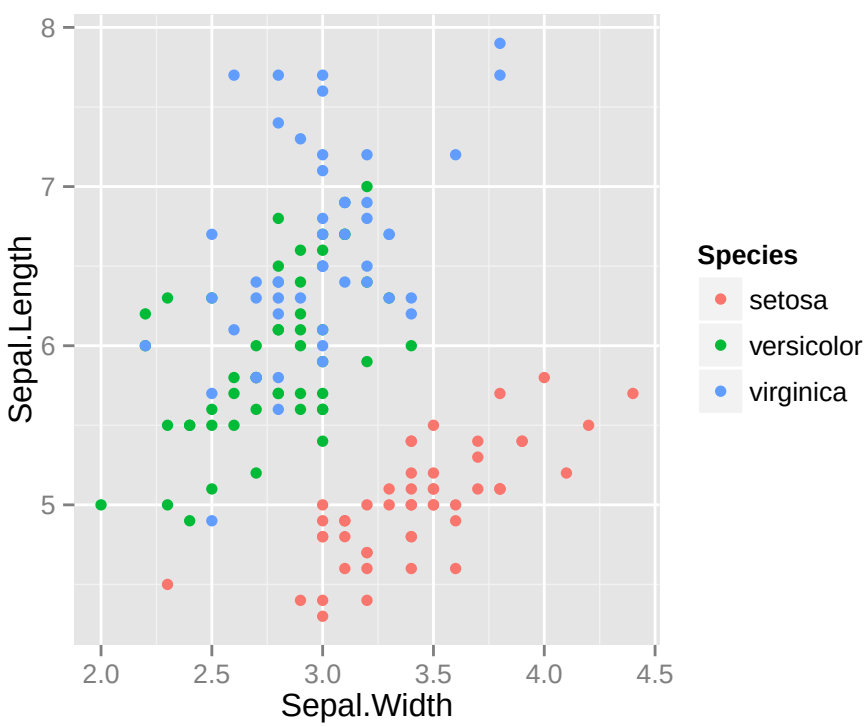
Aesthetic Mappings

we can map: color, shape to data.frames or arbitrary values

```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length,shape=Species)) + geom_point()
```




```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length,color=Species)) + geom_point()
```



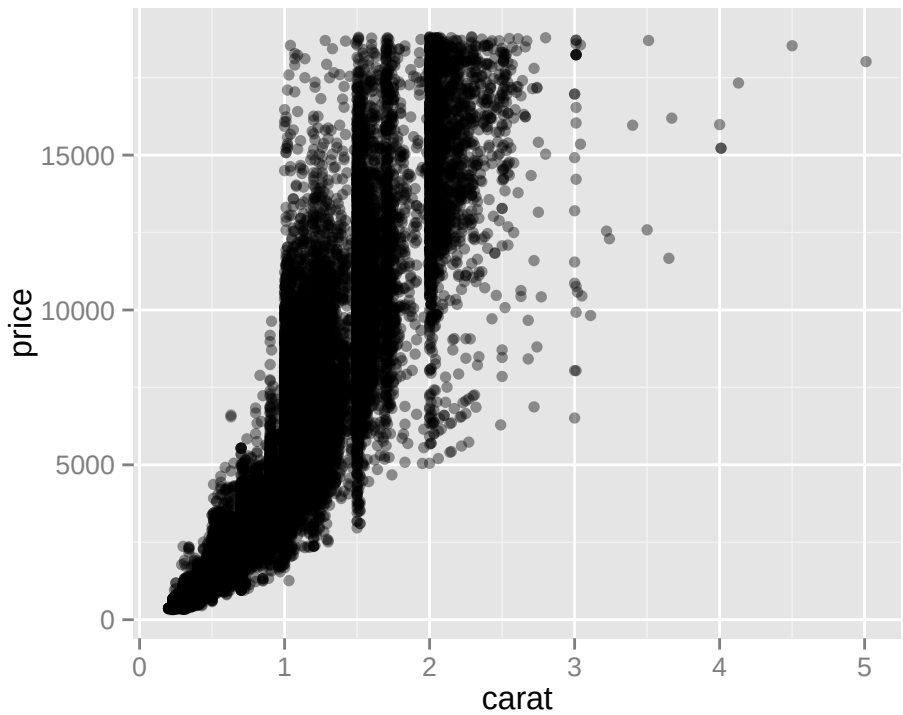
Question 2

Using the weight dataset, what is the distribution of age vs. weight?
By coloring the points, is low birth weight associated with either variable?
Check the answer by `print(answer2)`.

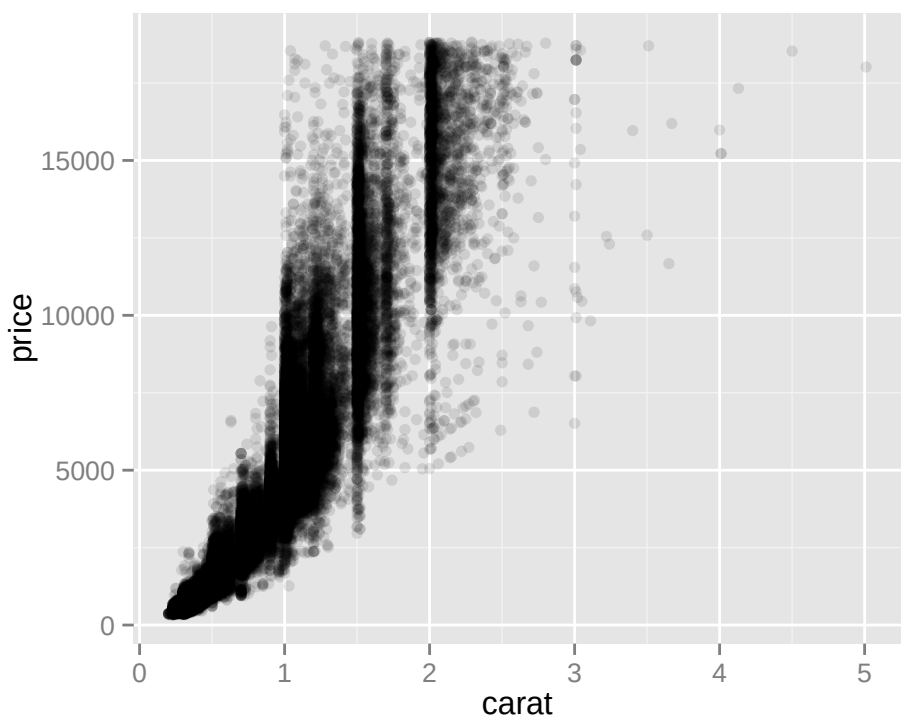
```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length)) + geom_point(color="red",size=4)
```

Set aesthetics to fixed values:

```
ggplot(diamonds,aes(x=carat,y=price))+ geom_point(alpha=I(0.4))
```




```
ggplot(diamonds,aes(x=carat,y=price))+ geom_point(alpha=I(0.1))
```



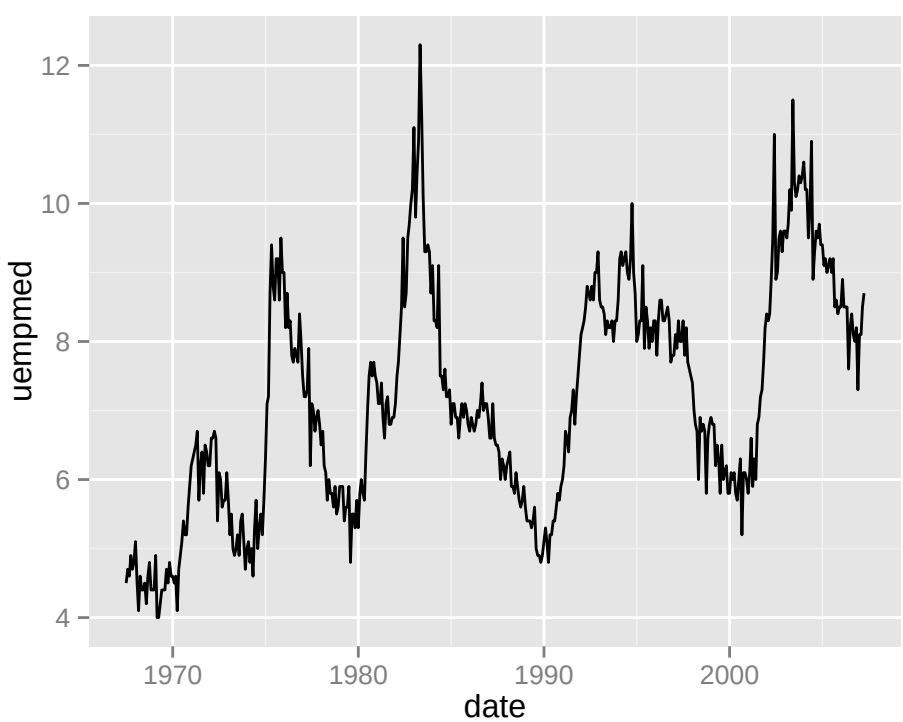

```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Species)) +
  geom_point(size=4) +
  geom_point(size=2,color="grey")
```

Layered geoms

Line Plot

note, the aesthetic mappings work the same

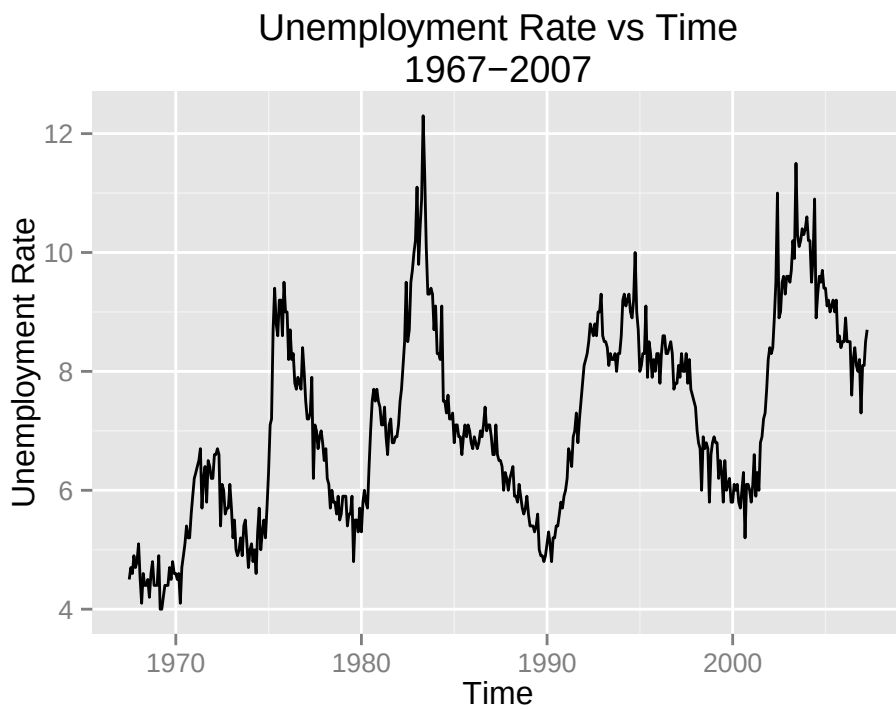
```
ggplot(economics,aes(x=date,y=uempmed))+geom_line()
```



Question 3

plot the number of Typhoid Fever deaths per year using the epi dataset. If you are stuck, take a look at `cat(hint3)` and `cat(hint3.1)` for the data transforms.

```
ggplot(economics,aes(x=date,y=uempmed))+ geom_line() +
  xlab("Time") +
  ylab("Unemployment Rate") +
  ggtitle("Unemployment Rate vs Time\n1967-2007")
```



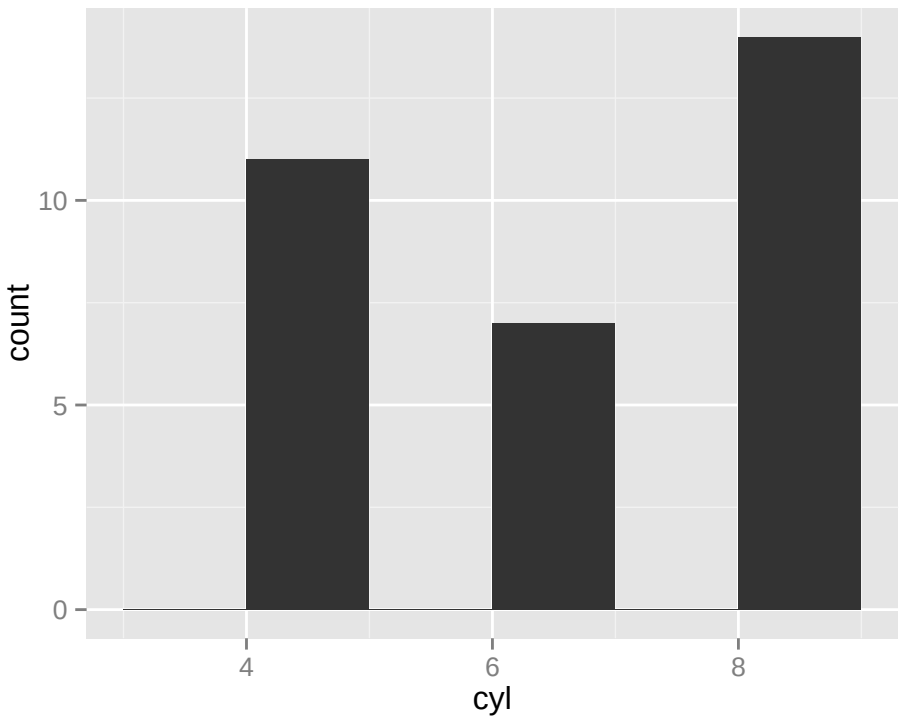
Add labels and a title:

Question 4

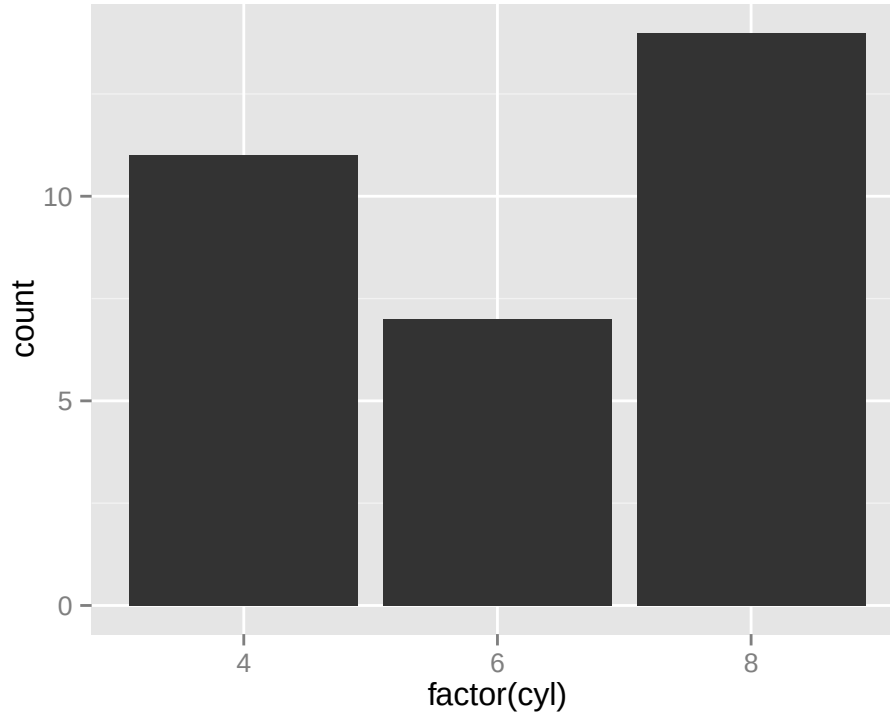
Add labels and a title to the previous plot.

Bar Plot

```
ggplot(mtcars,aes(x=cyl))+geom_bar(binwidth=1)
```

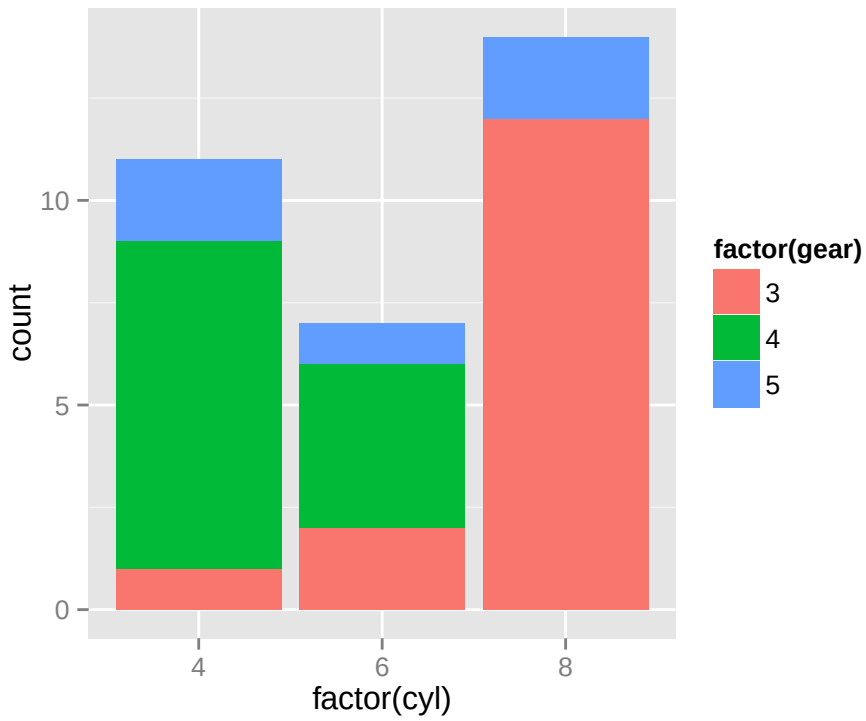



```
ggplot(mtcars,aes(x=factor(cyl)))+geom_bar()
```



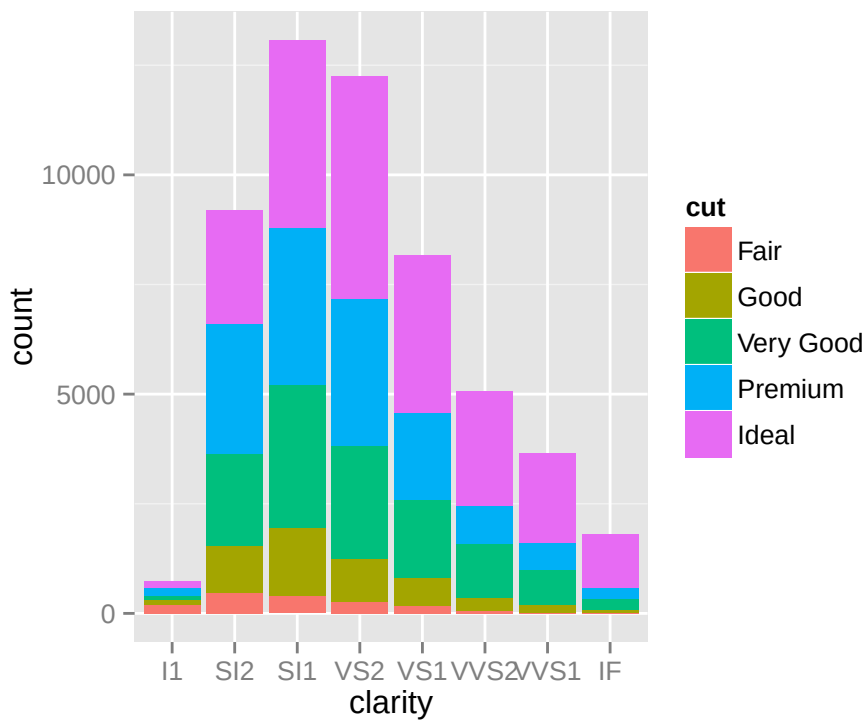
use factor instead of numerical datatypes

```
ggplot(mtcars,aes(x=factor(cyl),fill=factor(gear)))+geom_bar()
```



color the bars by type

```
ggplot(diamonds, aes(clarity, fill=cut)) + geom_bar()
```



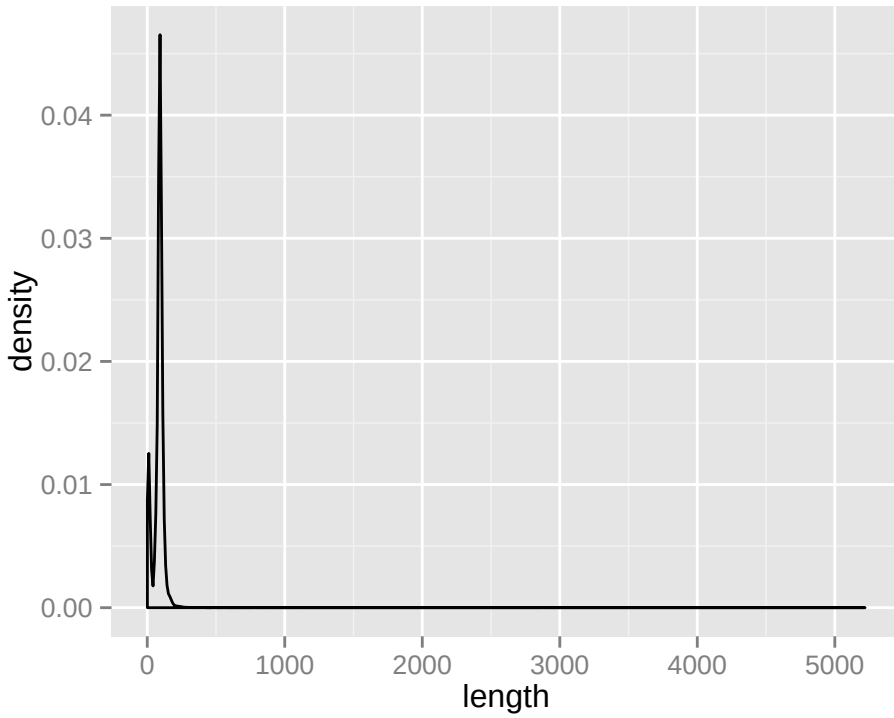
another, better example:

Question 5

What is the distribution of cases of typhoid per week? What about per year?
see `cat(answer5)` and `cat(answer5.1)`

Density Plot

```
ggplot(movies, aes(length))+geom_density()
```

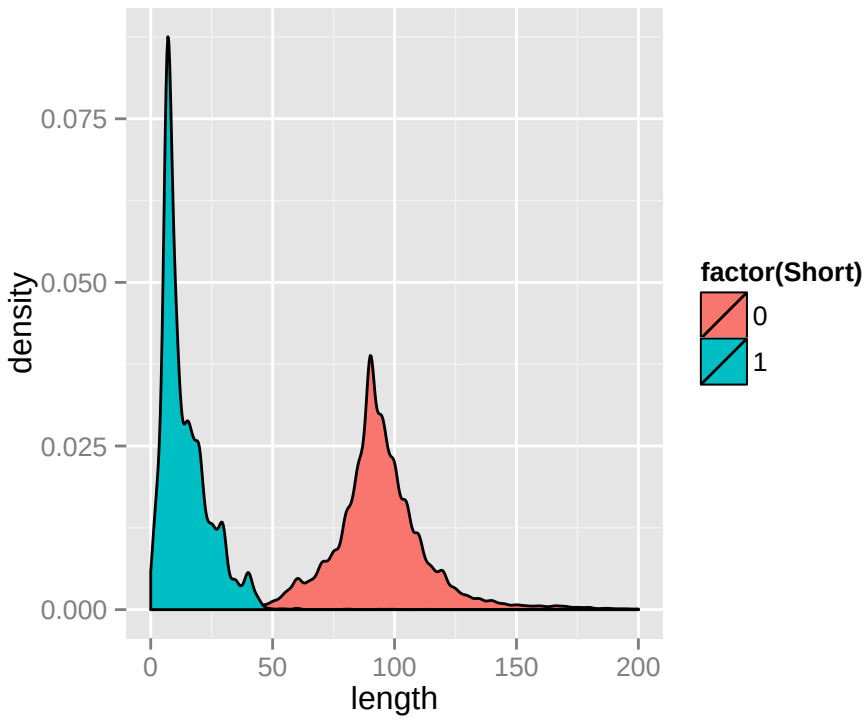


Adjust x and y limits we need to "zoom in" to get a resonable view create distributions of short vs. non-short movies:

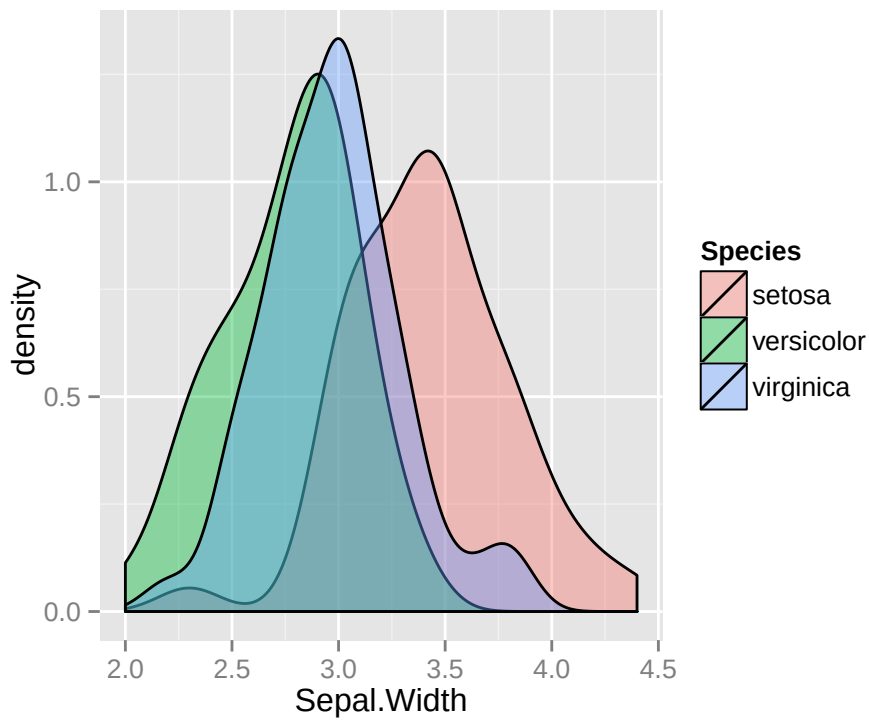
```
ggplot(movies, aes(x=length,fill=factor(Short)))+geom_density() + xlim(0,200)
```

```
## Warning: Removed 249 rows containing non-finite values (stat_density).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```




```
ggplot(iris, aes(x=Sepal.Width,fill=Species))+geom_density(alpha=I(0.4))
```



Set transperancy of distro

Question 6a

High glycosaminoglycans (GAGs) urine levels can progressively damage tissue, and high levels are indictitive of inherited disease.Using the GAGurine dataset

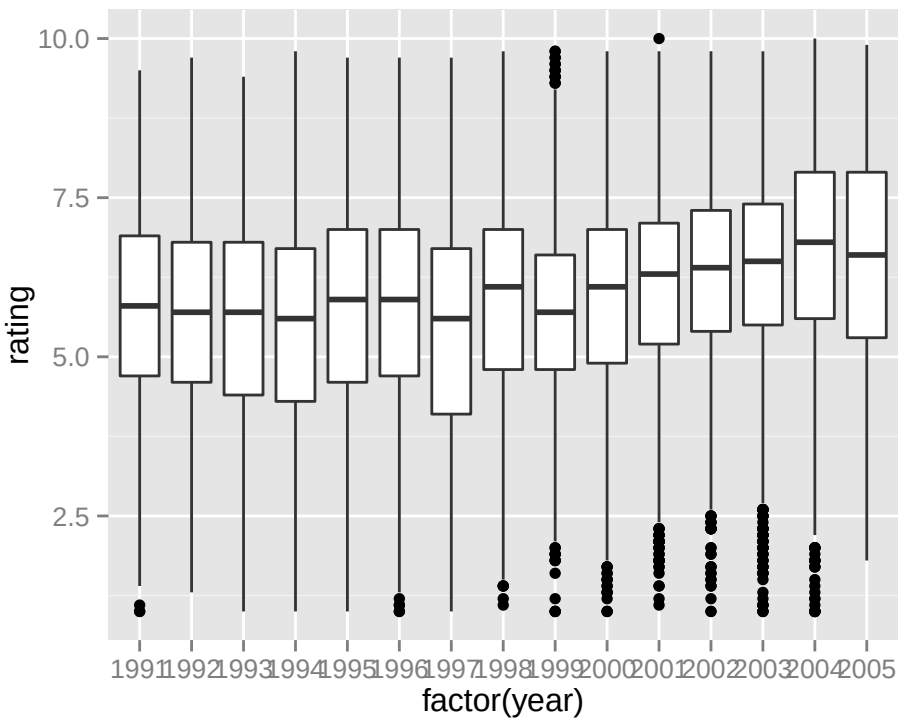
from the MASS package, what is the distribution of GAG assay values?

Question 6b

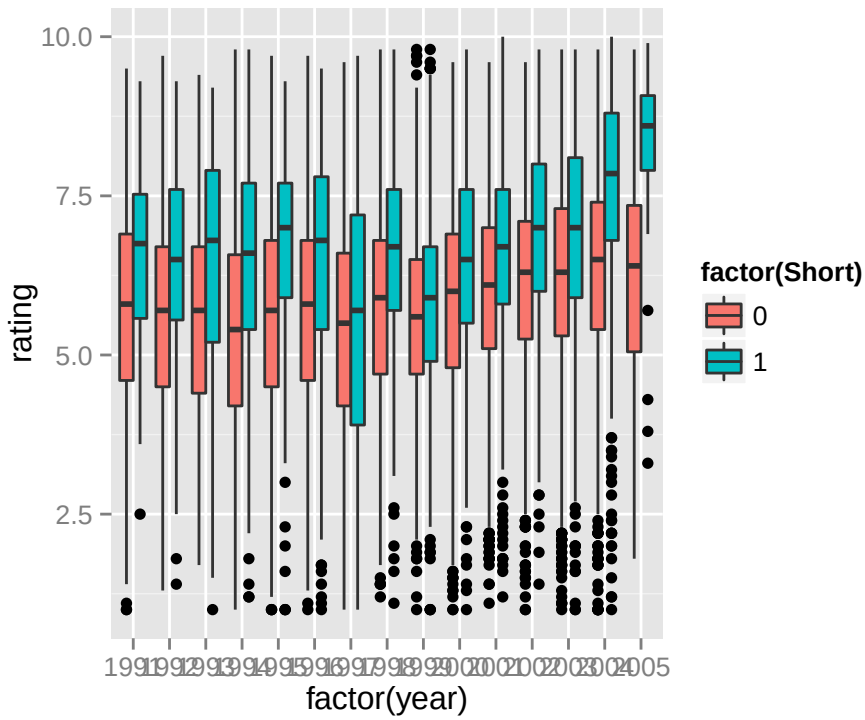
What's the difference in GAG distribution between children under 2 and all other children?
see `cat(hint6b)` for data transform

Boxplot

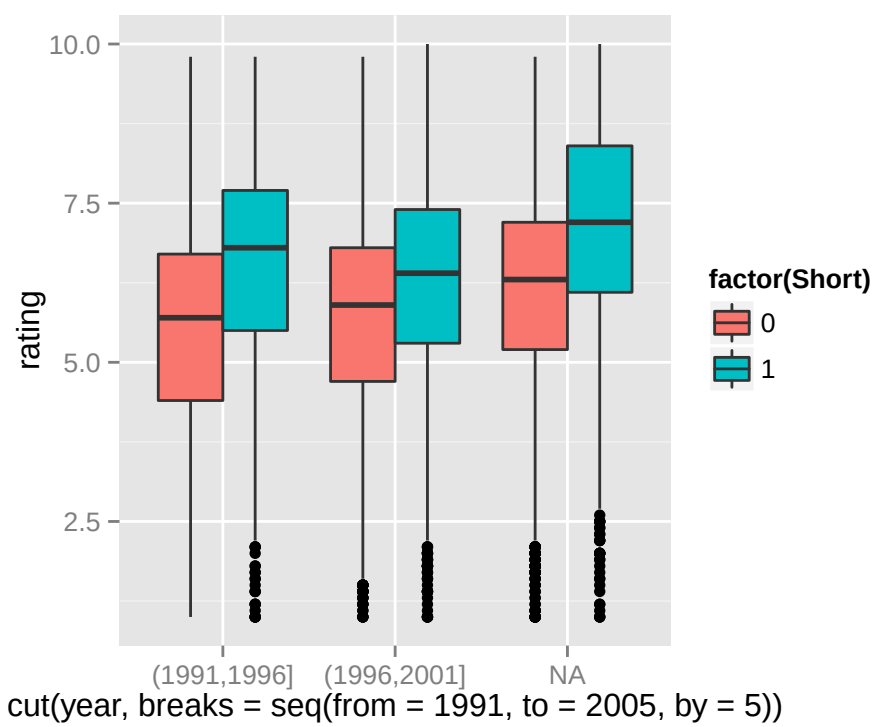
```
ggplot(movies[movies$year > 1990,],aes(x=factor(year),y=rating)) +
  geom_boxplot()
```




```
ggplot(movies[movies$year > 1990,],aes(x=factor(year),y=rating,fill=factor(Short))) +
  geom_boxplot()
```




```
ggplot(movies[movies$year > 1990,],
aes(x=cut(year,breaks=seq(from=1991,to=2005,by=5)),y=rating,fill=factor(Short))) +
geom_boxplot()
```



Question 7

What is the distribution of GAG assay levels for each age?
see `cat(hint7)` for data transform

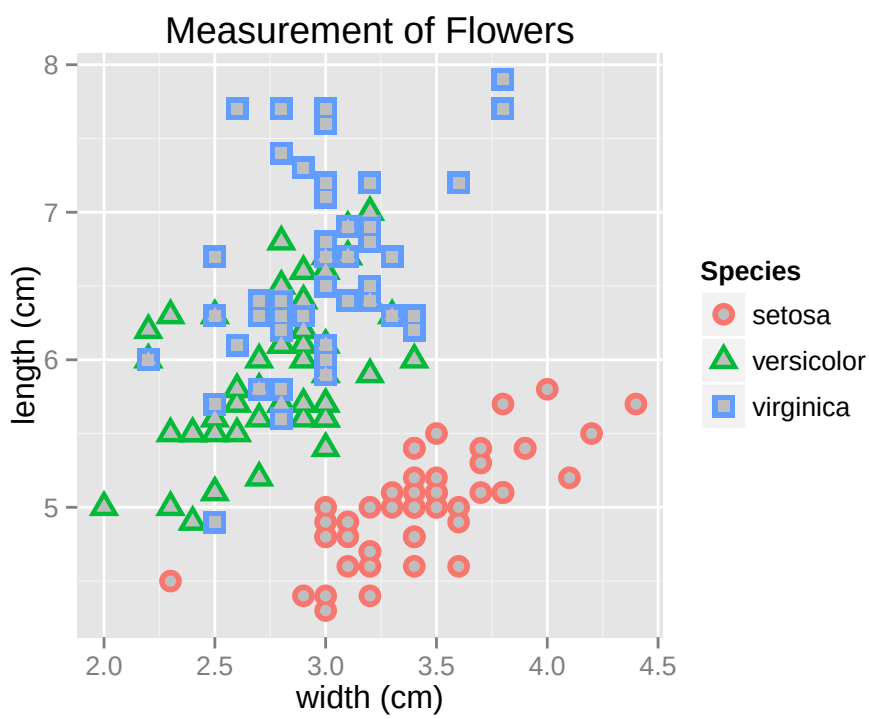
Themes

Adjust the background color, tick marks, etc

```
iris.plot = ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Species)) +
  geom_point(size=4) +
  geom_point(size=2,color="grey")+
  ggtitle("Measurement of Flowers")+
  xlab("width (cm)")+
  ylab("length (cm)")
```

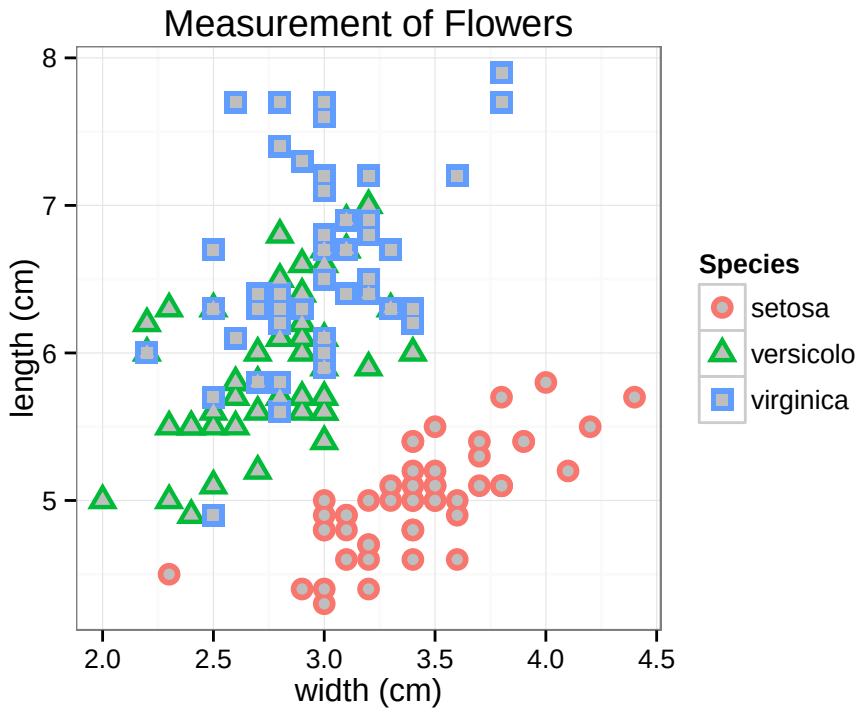
store what we have so far to a variable

```
iris.plot
```



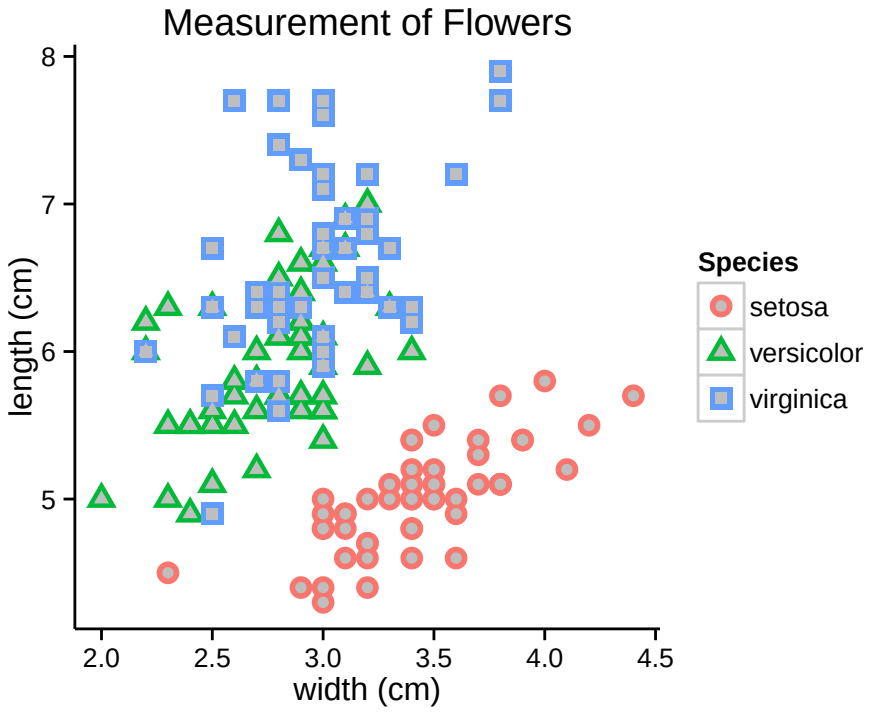
plot stored variable

```
iris.plot + theme_bw()
```



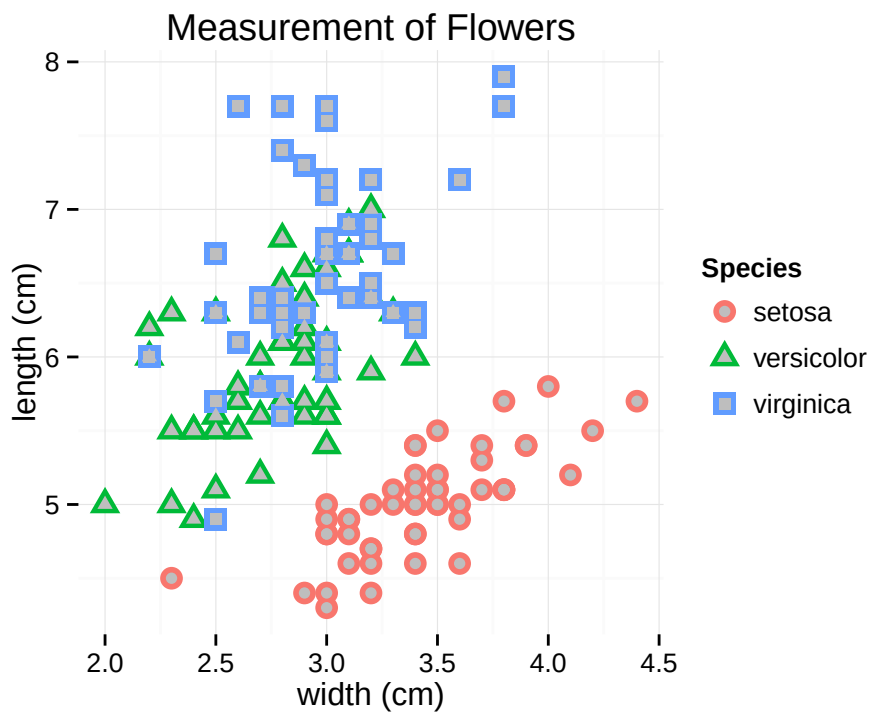
Change the background to black and white

```
iris.plot + theme_classic()
```

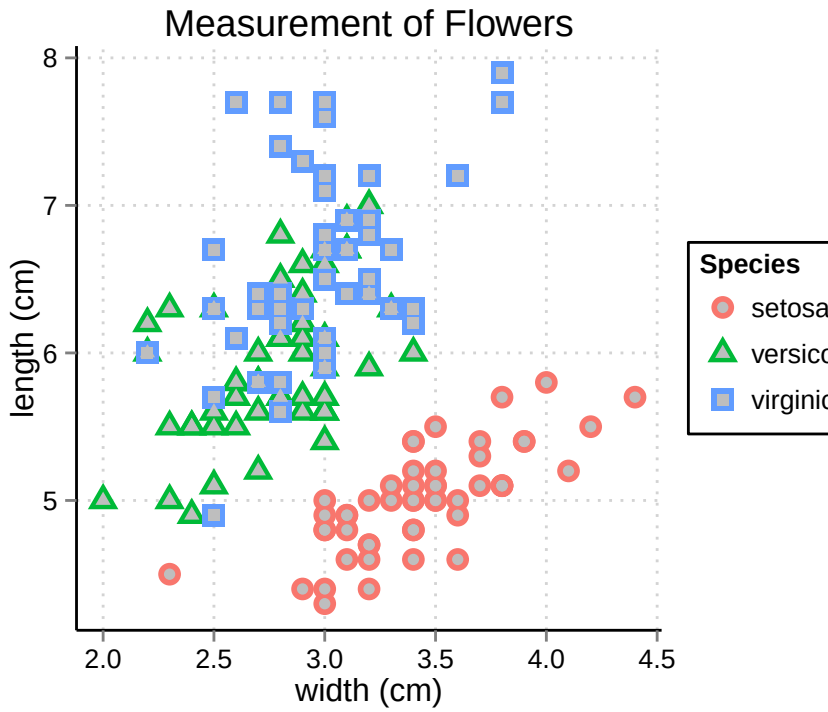


Other themes to consider:

```
iris.plot + theme_minimal()
```




```
iris.plot +
  theme(plot.background = element_rect(fill = "white"),
        panel.background = element_rect(fill = "white"),
        legend.key = element_rect(fill = "white"),
        legend.background = element_rect(colour = "black"),
        panel.grid.minor = element_blank(),
        panel.grid.major = element_line(colour = "lightgrey", linetype = "dotted"),
        axis.line = element_line(),
        axis.text = element_text(colour = "black"))
```



Customize the theme and specify features. . .

See http://docs.ggplot2.org/current/theme.html for more options

Question 8

Make the ugliest plot you can. Bonus points for discordant color schemes.

Bonus Material

go through at your own pace, or use as a reference

Bonus material

Legend Positioning

what options do we have for the position of the legend? examples borrowed from: http://www.r-bloggers.com/ggplot2-cheatsheet-for-scatterplots/

```
g1 <- ggplot(mtcars, aes(x = hp, y = mpg)) + geom_point(aes(color=factor(vs)))
```

```
g2 <- g1 + theme(legend.position=c(1,1),legend.justification=c(1,1))
```

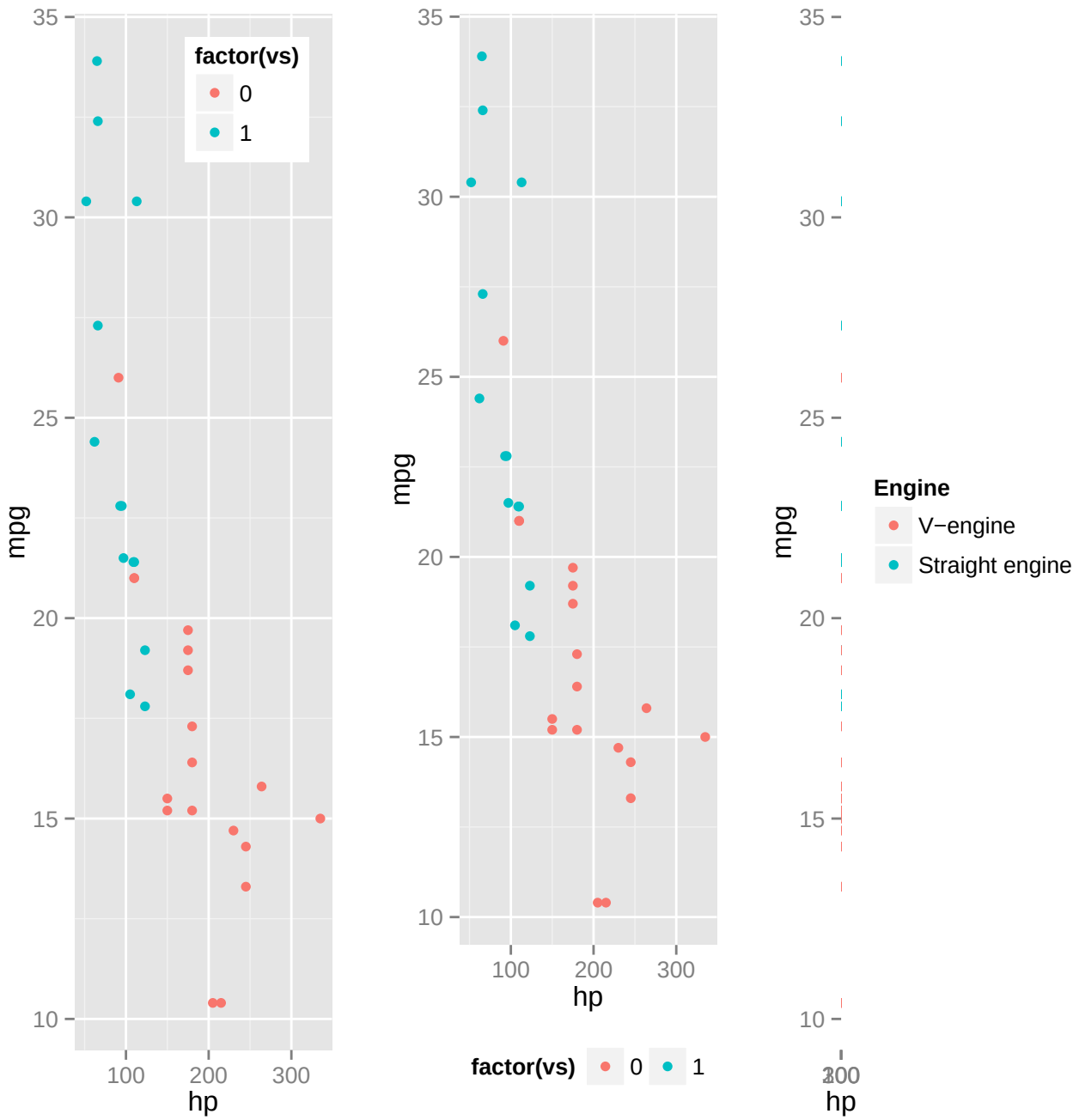
move legend to the inside

```
g3 <- g1 + theme(legend.position = "bottom")
```

move legend to the bottom

```
g4 <- g1 + scale_color_discrete(name ="Engine",
                                labels=c("V-engine", "Straight engine"))
```

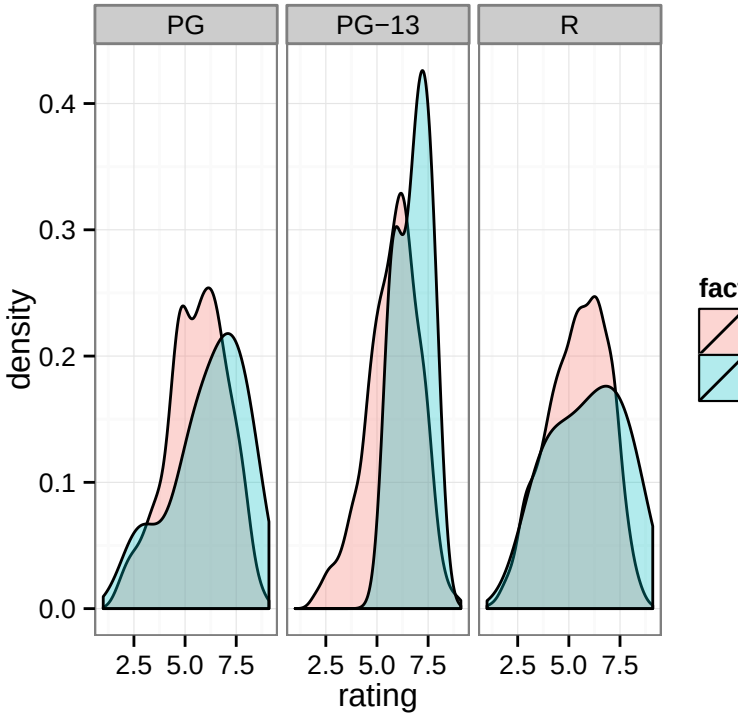
```
grid.arrange(g2, g3, g4, nrow=1)
```



change labels

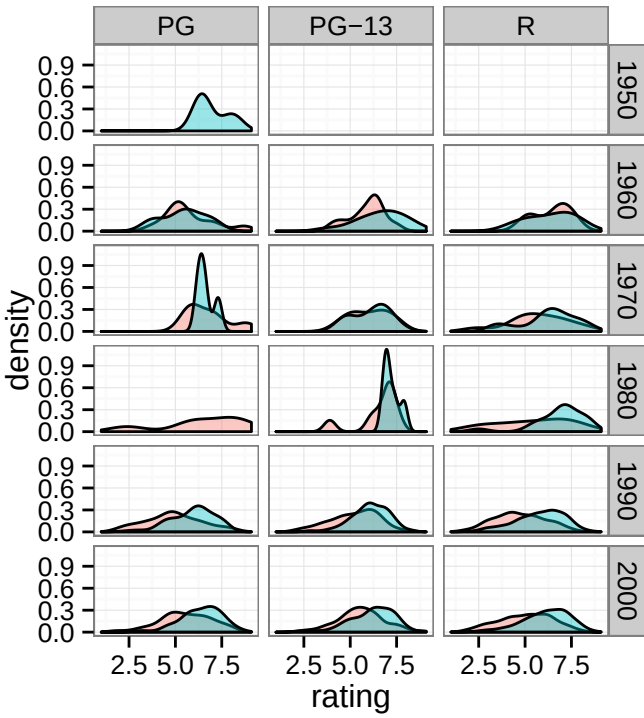
Facets

```
mov.df = movies[as.character(movies$mpaa) %in% c("PG","R","PG-13"),]
mov.df$mpaa = mov.df$mpaa[,drop=TRUE]
ggplot(mov.df,aes(x=rating,fill=factor(Short))) +
  geom_density(alpha=I(0.3)) + theme_bw() +
  facet_wrap(~mpaa)
```



Let's just look at PG, R, and PG-13 rated movies. . .

```
mov.df$decade = mov.df$year %/% 10 * 10
mov50plus.df = mov.df[mov.df$decade >= 1950,]
# facet_grid( y ~ x )
ggplot(mov50plus.df, aes(x=rating,fill=factor(Drama))) +
  geom_density(alpha=I(0.4))+ theme_bw()+
  facet_grid(decade ~ mpaa )
```

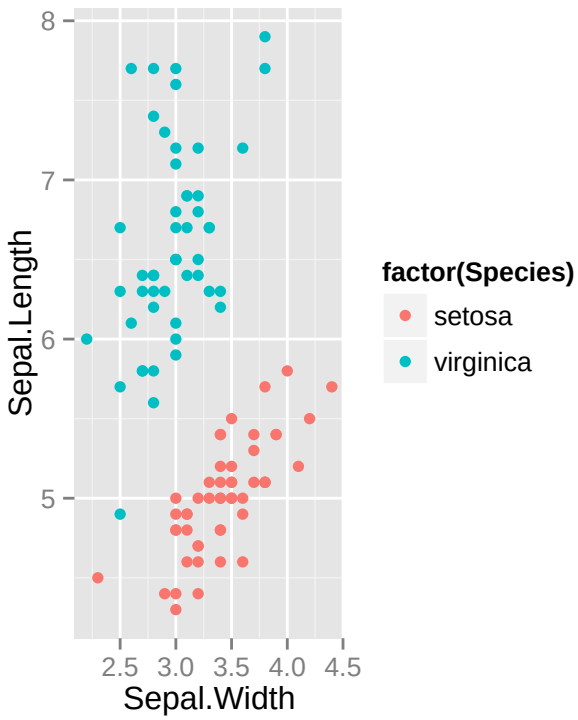


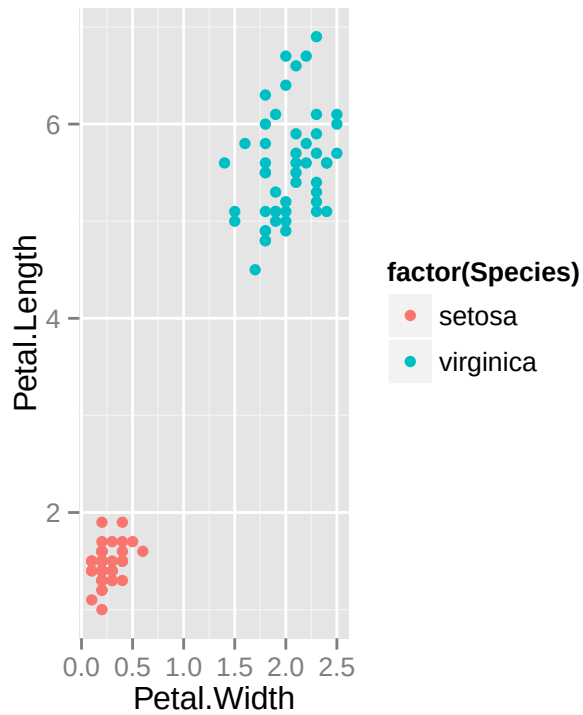
seperate movies by decade, rating: view rating distribution

Grids

Set up two different plots of iris data for two species, and view them side by side.

```
irisSmall.df <- subset(iris, Species != "versicolor")

p1 <- ggplot(irisSmall.df, aes(x= Sepal.Width, y = Sepal.Length)) + geom_point(aes(color=factor(Species
p2 <- ggplot(irisSmall.df, aes(x = Petal.Width, y = Petal.Length)) + geom_point(aes(color=factor(Specie

#' all plots in one row
grid.arrange(p1,p2,nrow=1)
```

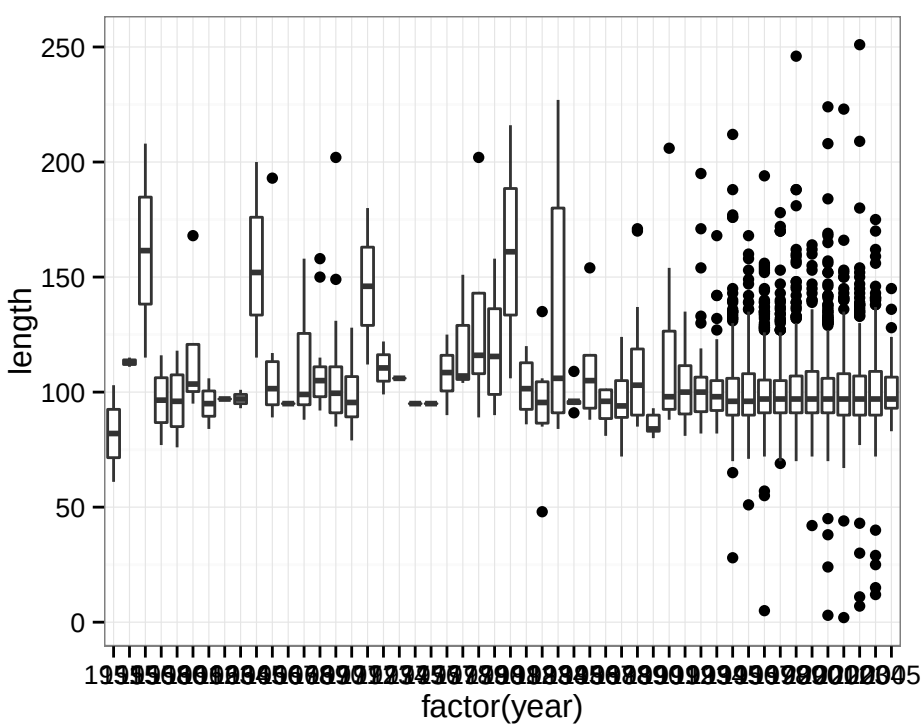




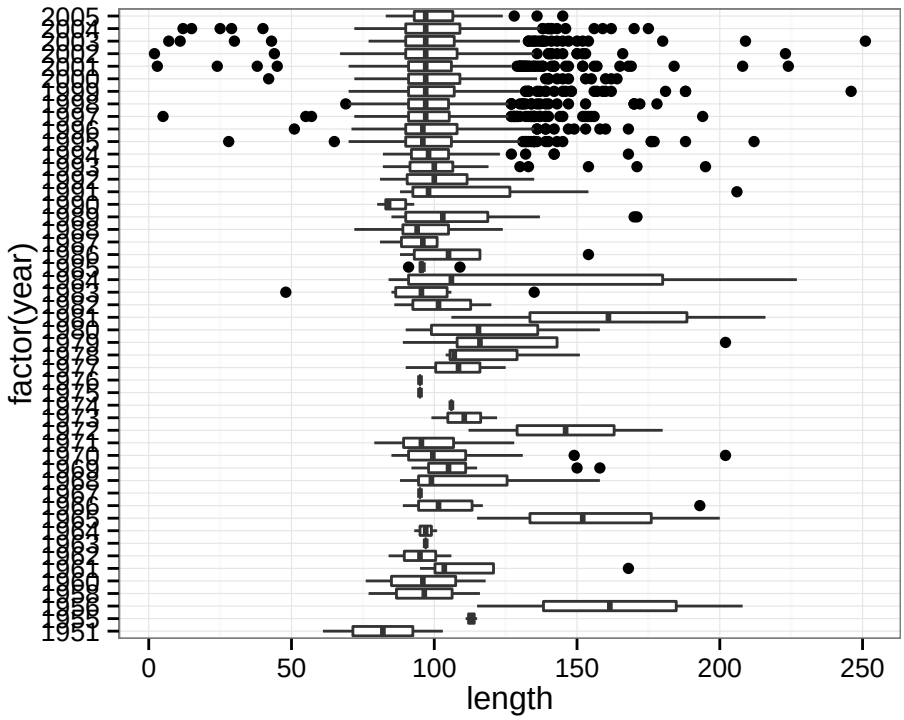
Coordinates

coord_flip flip the coords, easier to read x labels...

```
ggplot(mov50plus.df,aes(x=factor(year),y=length))+geom_boxplot()+theme_bw()
```



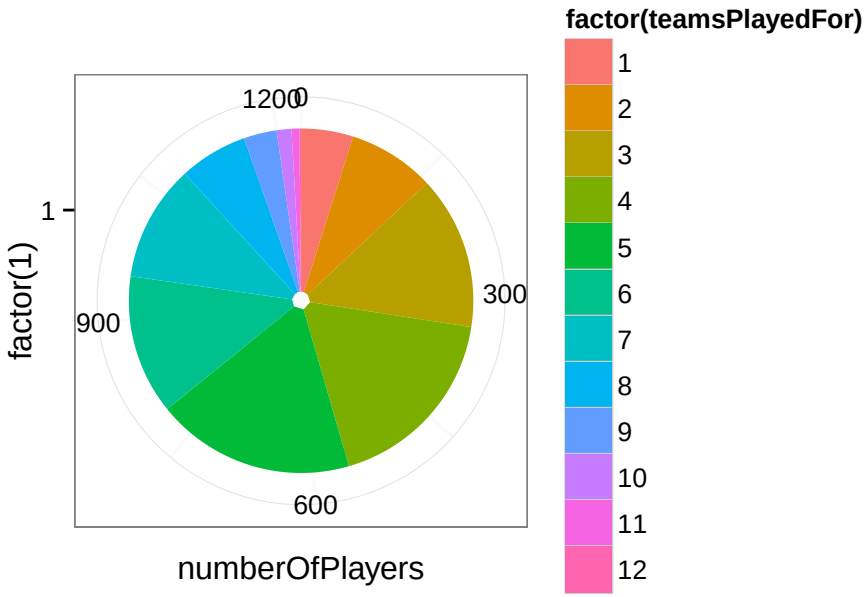
```
ggplot(mov50plus.df,aes(x=factor(year),y=length))+geom_boxplot()+theme_bw() + coord_flip()
```



coord_polar

How many teams do baseball players play for?

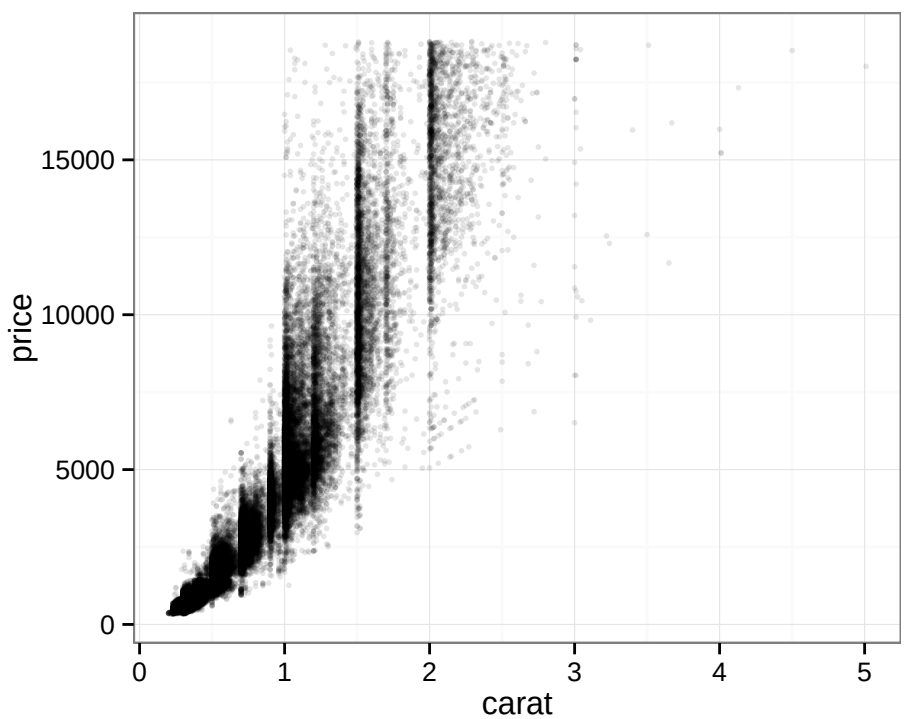
```
playerTeam.df = ddply(baseball,.(id),function(df)length(unique(df$team)))
teamsPerPlayer.df = as.data.frame(table(playerTeam.df$V1))

colnames(teamsPerPlayer.df) = c("teamsPlayedFor","numberOfPlayers")
teamsPerPlayer.df$teamsPlayedFor = as.numeric(teamsPerPlayer.df$teamsPlayedFor)
ggplot(teamsPerPlayer.df,aes(x=factor(1),y=numberOfPlayers,fill=factor(teamsPlayedFor))) +
  geom_bar(stat="identity") +
  theme_bw() +
  coord_polar(theta="y")
```



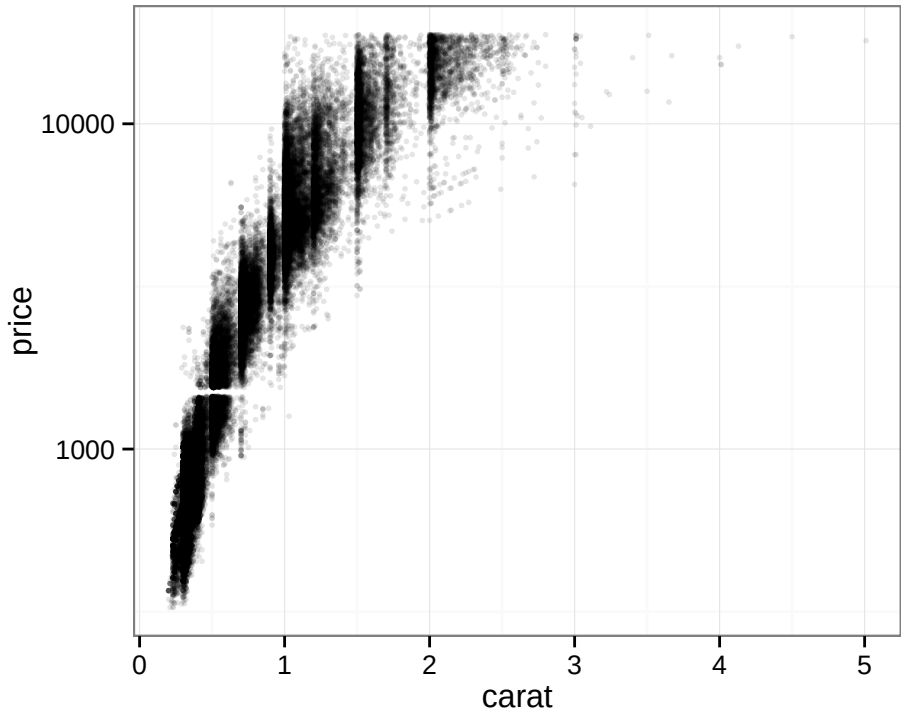
coord_trans

apply a transform to your coordinates, simpler than applying it to the data...

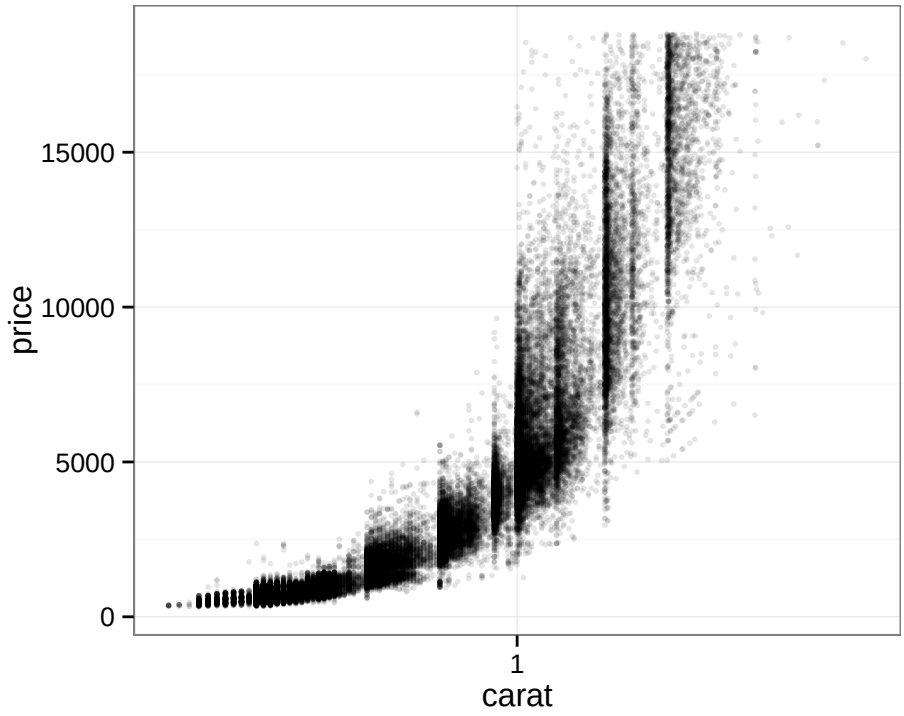
```
ct.plot = ggplot(diamonds, aes(carat,price))+geom_point(alpha=I(0.1),size=1)+
  theme_bw()
ct.plot
```



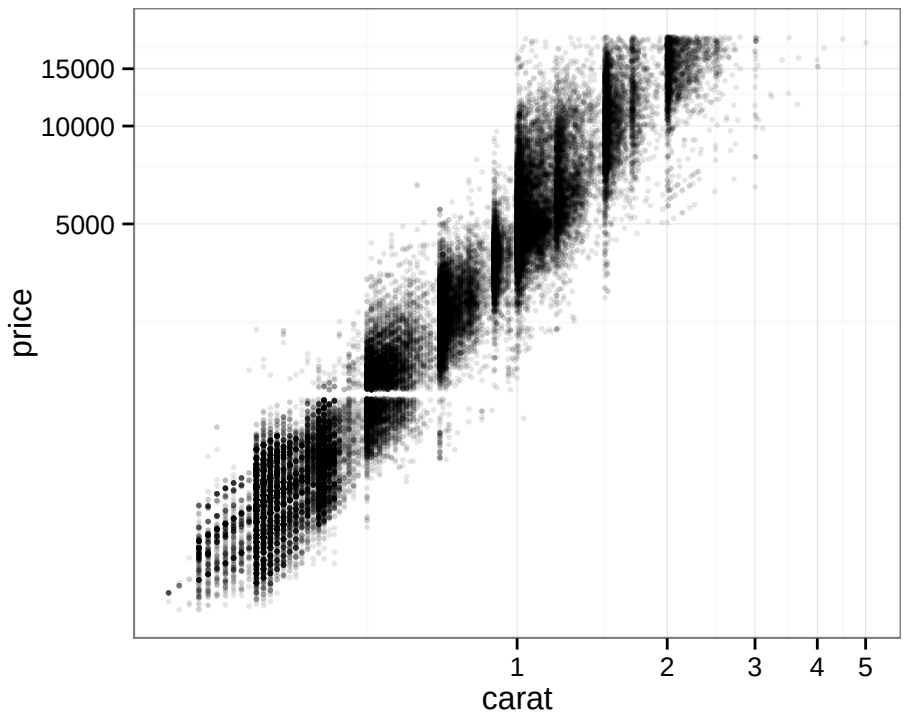
```
ct.plot + scale_y_log10()
```



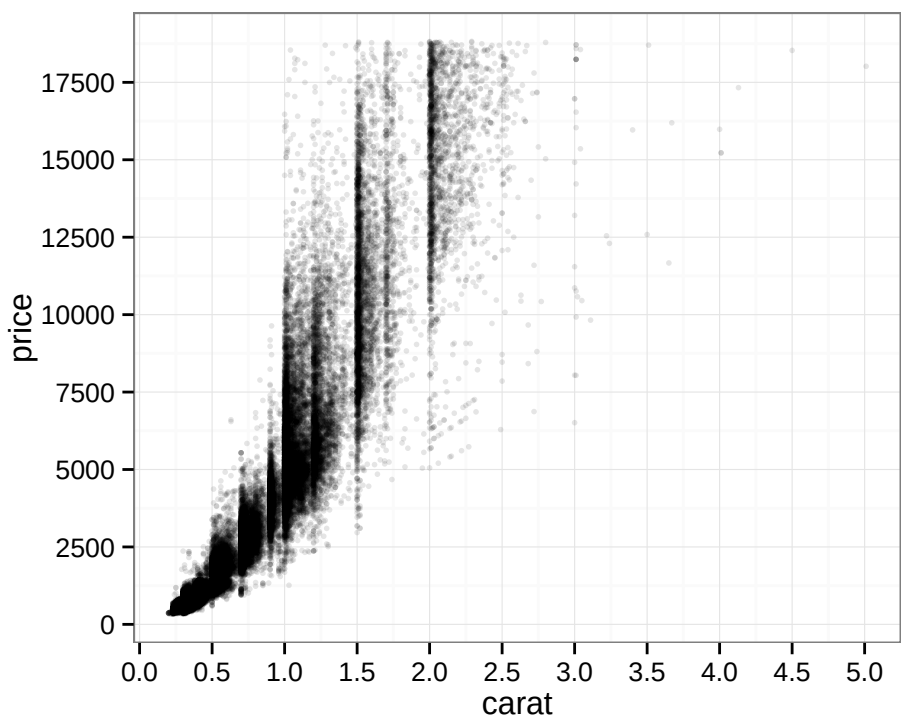

```
ct.plot + scale_x_log10()
```




```
ct.plot + coord_trans(x="log10",y="log10")
```




```
#'# change the axis ticks w/ labels
ct.plot + scale_y_continuous(breaks= seq(0,20000,by=2500))+
  scale_x_continuous(breaks= seq(0,5,by=0.5))
```

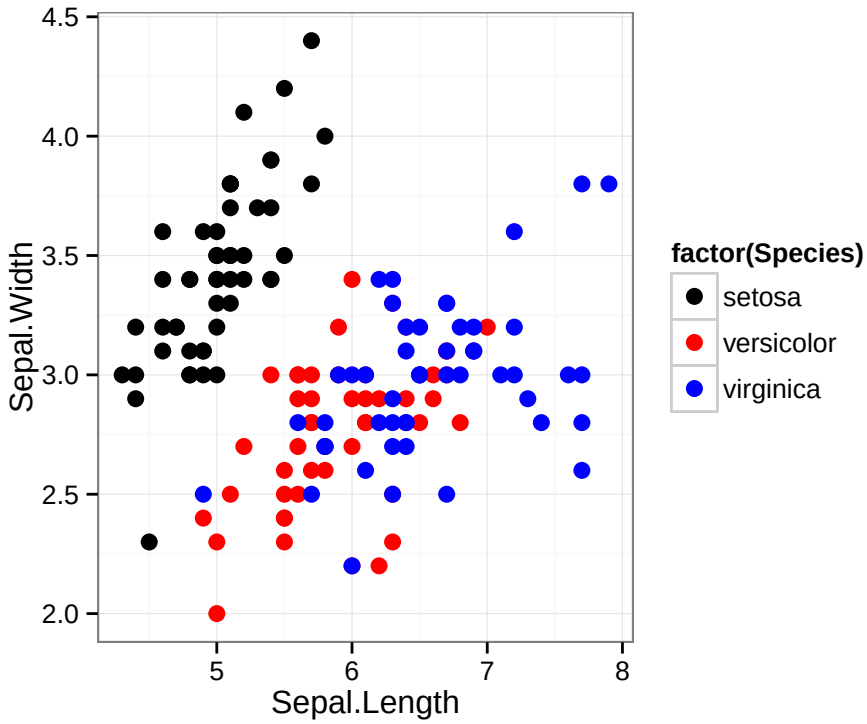


Color/Fill

note: different scales are used for discrete and continuous data scale_color_manual for discrete data

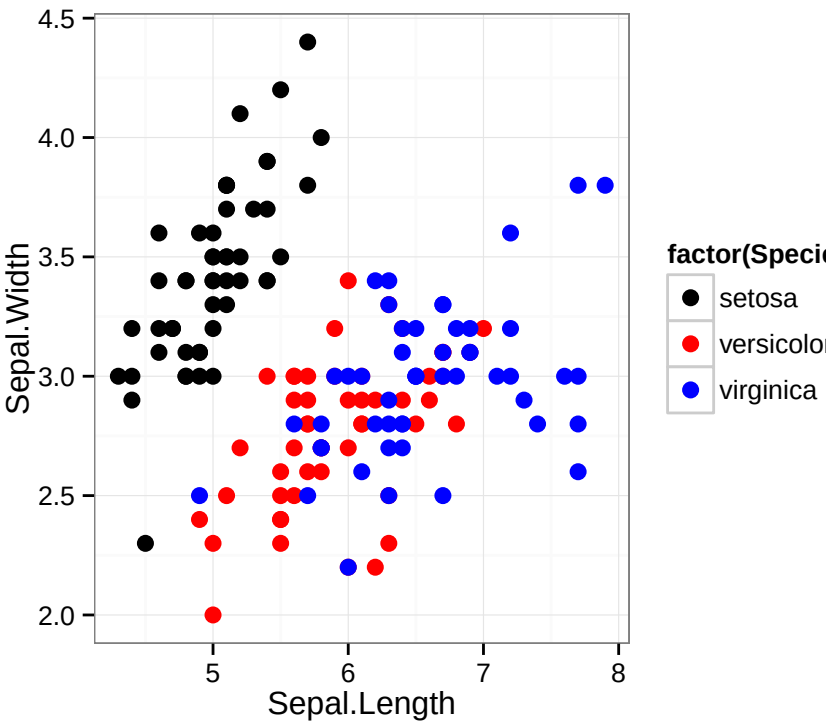
```
scm.plot = ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width,color=factor(Species))) + geom_point(size=3)+
  theme_bw()
```

```
scm.plot + scale_color_manual(values=c("black","red","blue"))
```



automatic assignment

```
scm.plot + scale_color_manual(values=c("setosa"= "black","versicolor"= "red","virginica"= "blue"))
```

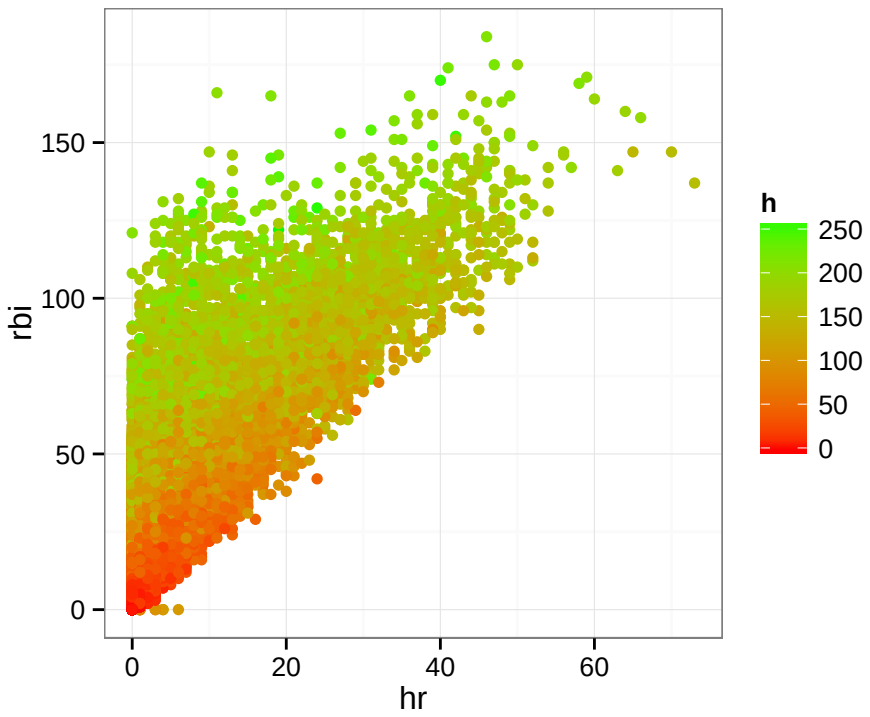


specify colors and assignment to factor values

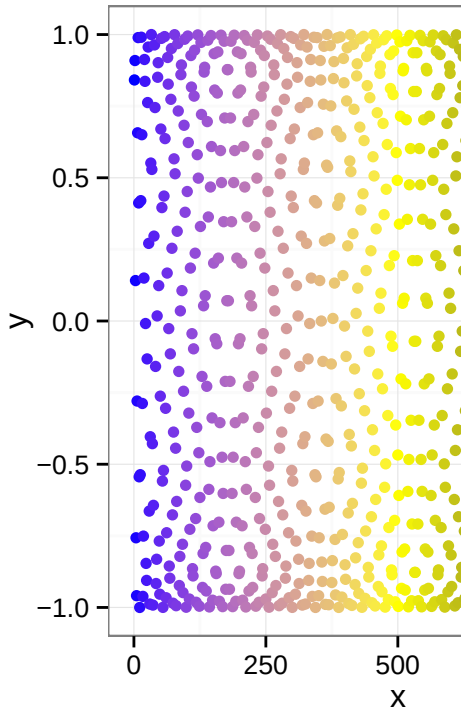
```
ggplot(baseball,aes(x=hr,y=rbi,color=h))+ geom_point()+
  theme_bw() +
  scale_colour_gradient(low="red",high="green")
```

scale_colour_gradient for continuous data

```
## Warning: Removed 12 rows containing missing values (geom_point).
```



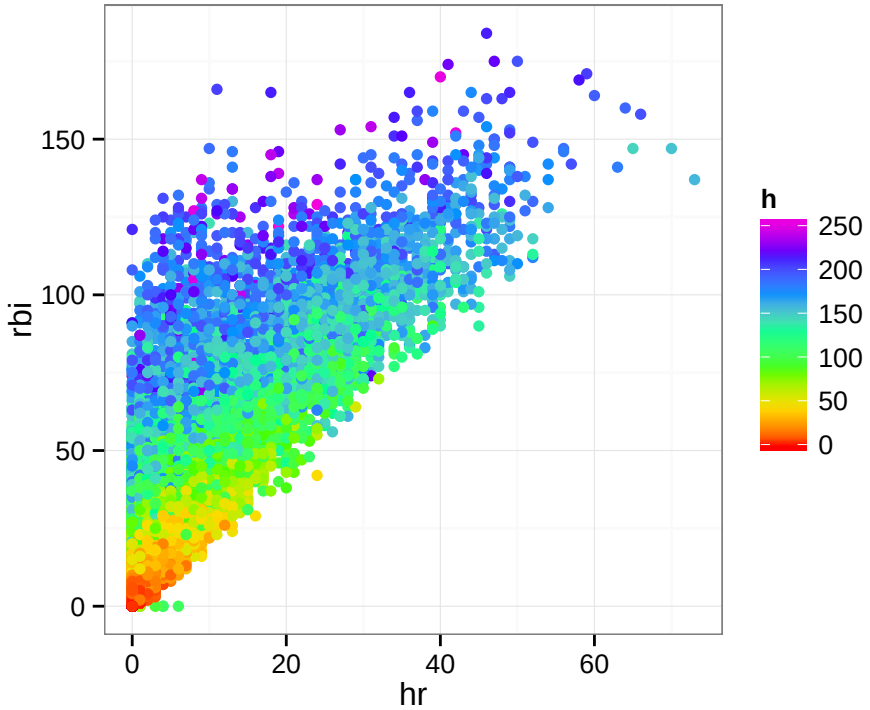

```
sin.df <-data.frame(x=1:1000,y=sin(1:1000),rr=1:1000 - (2*pi)*(1:1000 %/% 2*pi))
ggplot(sin.df,aes(x,y,color=rr))+geom_point(size=2)+theme_bw()+
  scale_colour_gradientn(colours=c("black","yellow","blue"))
```



scale_colour_gradientn provides smoothing over multiple color values. . .

```
scgn.plot = ggplot(baseball,aes(x=hr,y=rbi,color=h))+ geom_point()+ theme_bw()
scgn.plot + scale_color_gradientn(colours=rainbow(7))
```

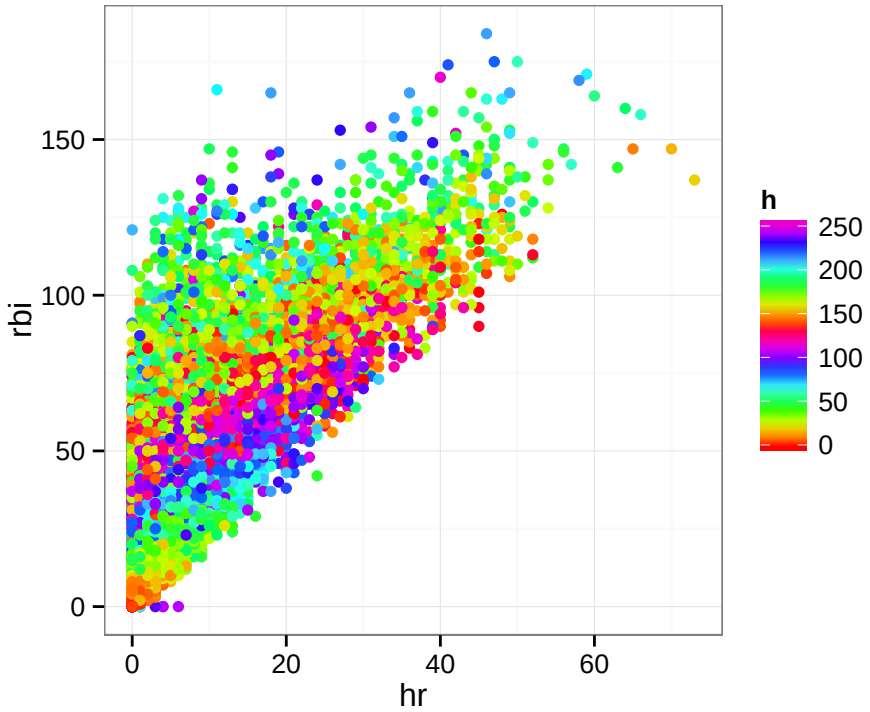
```
## Warning: Removed 12 rows containing missing values (geom_point).
```




```
scgn.plot + scale_color_gradientn(colours=rep(rainbow(10),2))
```

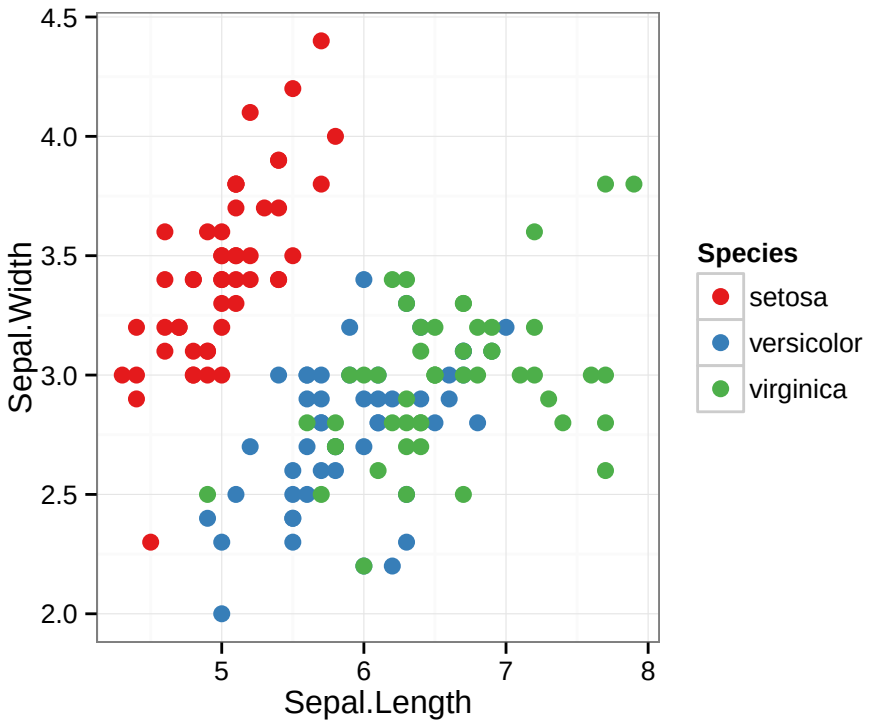
you can cycle through the colors for better resolution

```
## Warning: Removed 12 rows containing missing values (geom_point).
```



scale_colour_brewer, offers a set of nice palettes for discrete coloring:

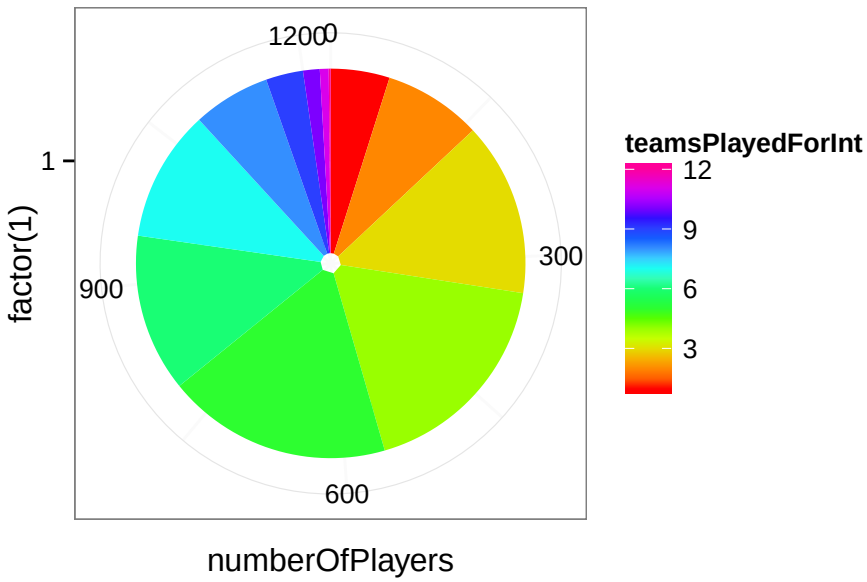
```
ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width,color=Species)) +
  geom_point(size=3) +
  scale_colour_brewer(palette="Set1")+theme_bw()
```



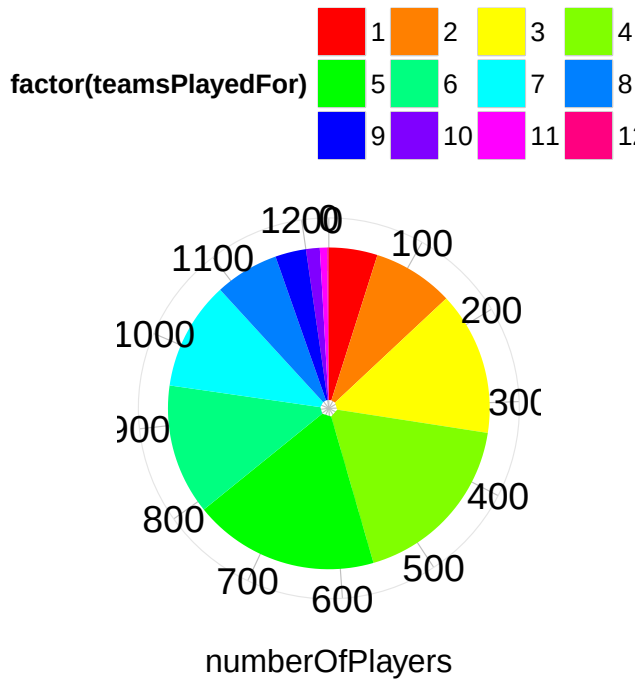
scale_fill_*

works the same as the scale_colour_*, and is used when the geom is "filled" instead of "colored" (note this change from factor to integer is improper)

```
teamsPerPlayer.df$teamsPlayedForInt = 1:12

ggplot(teamsPerPlayer.df,aes(x=factor(1),y=numberOfPlayers,fill=teamsPlayedForInt)) +
  geom_bar(stat="identity") +
  theme_bw() +
  coord_polar(theta="y")+
  scale_fill_gradientn(colours=rainbow(10))
```




```
ggplot(teamsPerPlayer.df,aes(x=factor(1),y=numberOfPlayers,fill=factor(teamsPlayedFor))) +
  geom_bar(stat="identity") +
  theme_bw() +
  coord_polar(theta="y")+
  scale_fill_manual(values=rainbow(12),guide= guide_legend(ncol = 4,byrow=TRUE))+
  theme(legend.position = "top",
        panel.grid.minor.y = element_line(colour = "grey"),
        panel.grid.major.y = element_line(colour = "grey"),
        axis.text.x = element_text(size=rel(1.5)),
        axis.text.y = element_blank(),
        panel.border = element_blank(),
        axis.ticks.y = element_blank()) +
  scale_y_discrete(limits= seq(0,1228,by=100))+
  xlab("")
```



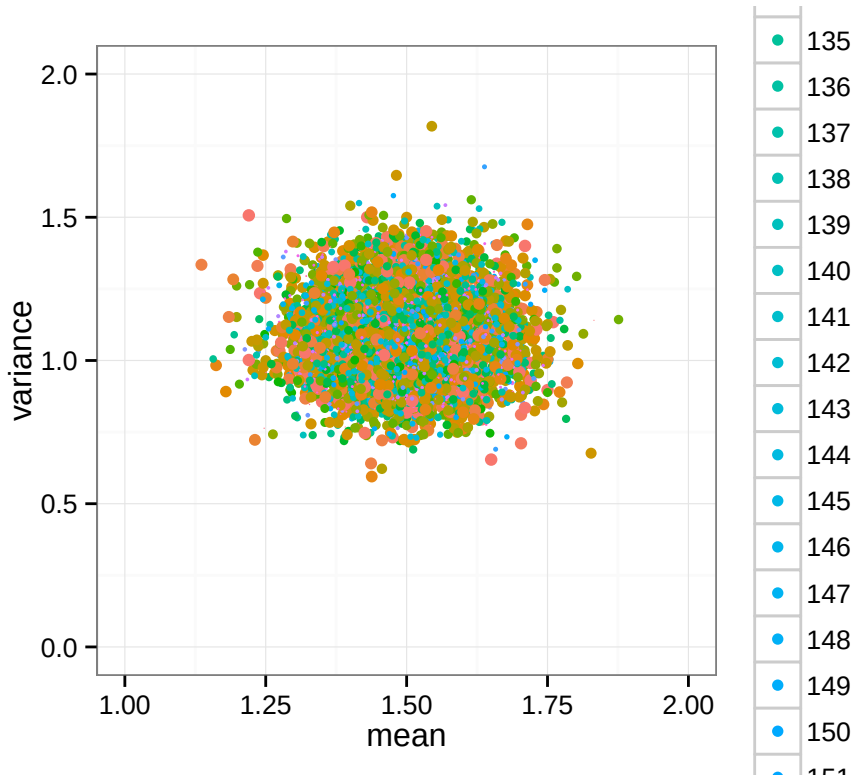
we can try again using discrete values to get proper labels. . .

Advanced Examples

ex. 1

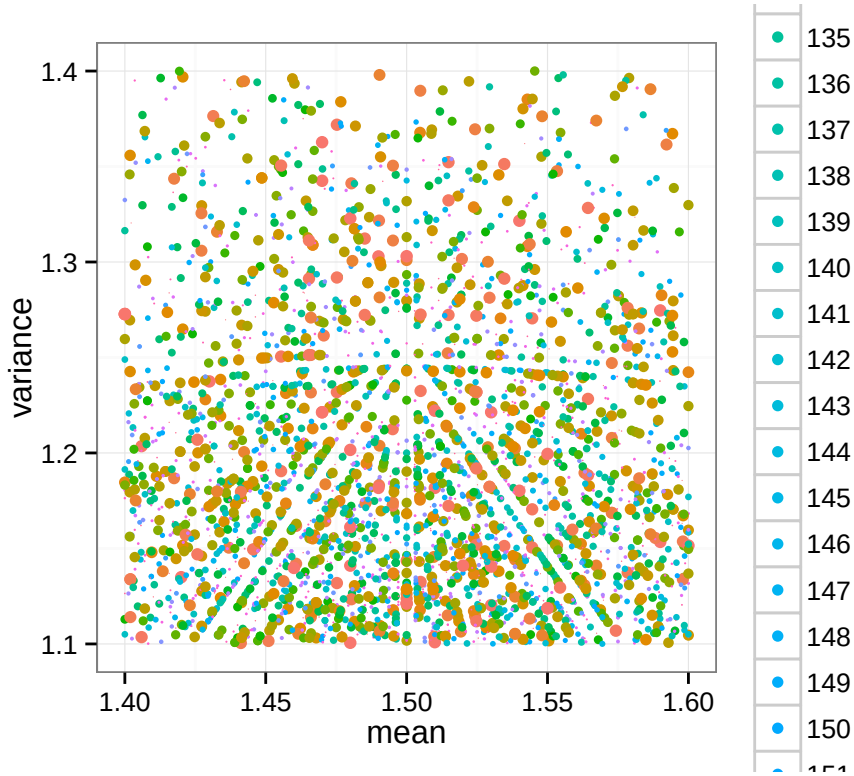
objective: look at the variance and mean of samples of discrete data approach: create a function that initiates the dataset, then returns a function whose arguments set the plot window

```
makePlotFunction <-function(size,fun,mean,range,rangeLower=2){
  df<-as.data.frame(t(as.matrix(
    sapply(floor(runif(size)*(range))+ rangeLower,
           function(x){vec<-sapply(fun(x,mean),floor);
                       c(mean(vec),var(vec),length(vec))}))))#end of data.frame
  colnames(df)<-c("mean","variance","n");
  function(pMin,pMax,xVec,yVec){
    df$pSize = as.numeric(df$n)
    ggplot(df,aes(x=mean,y=variance,color=factor(n),size=pSize))+
      geom_point()+
      scale_size(range = c(pMin,pMax))+
      theme_bw()+
      xlim(xVec)+
      ylim(yVec)}}

# set the function
f1= makePlotFunction(size=10000,fun=rnorm,mean=2,range=80,rangeLower=100)
f1(xVec=c(1,2),yVec=c(0,2),pMin=2.2,pMax=0.2)
```

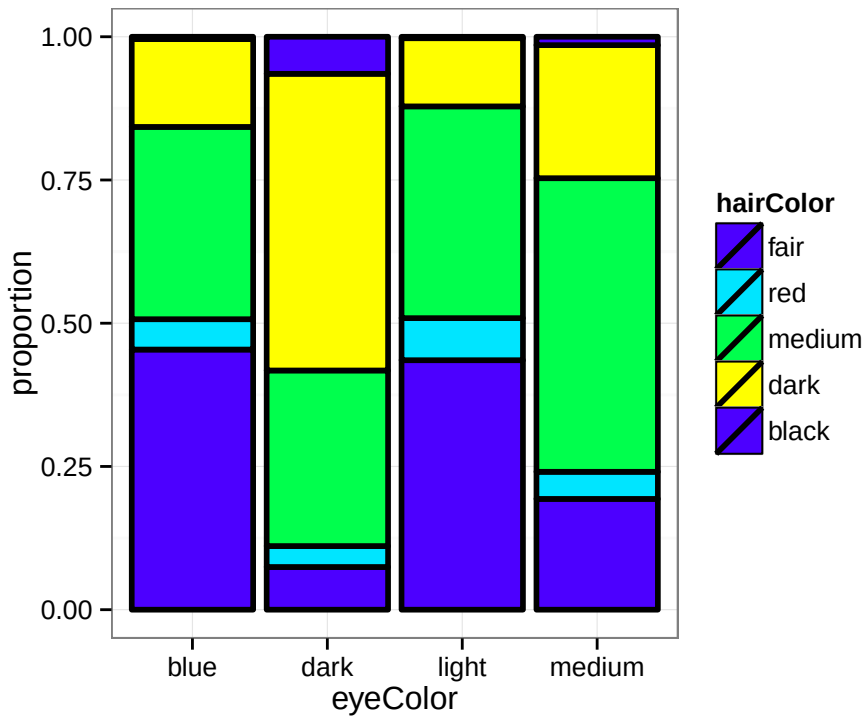



```
f1(xVec=c(1.4,1.6),yVec=c(1.1,1.4),pMin=2.2,pMax=0.2)
```

```
## Warning: Removed 6939 rows containing missing values (geom_point).
```



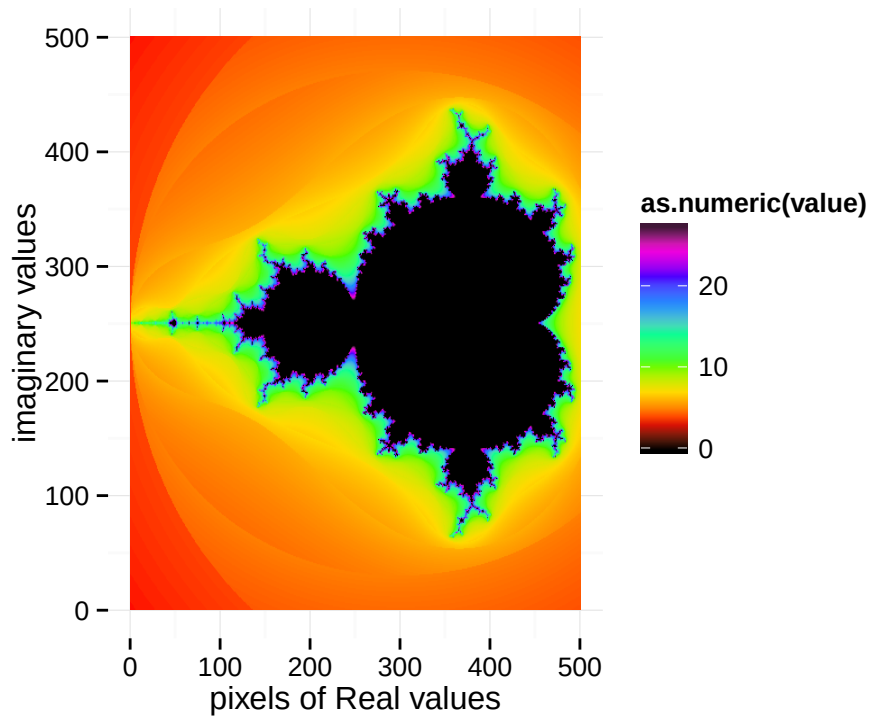
```r
# Mosiac Plot
library(MASS)
data(caith)

# melt function :: wide data.frame -> long data.frame
caith.melt= melt(
  cbind(as.data.frame(caith),
        row=rownames(caith)),
  id.var=c("row"))

#' rename the columns
colnames(caith.melt) = c("eyeColor","hairColor","proportion")

ggplot(caith.melt ,aes(x=eyeColor,y=proportion,fill=hairColor,color=hairColor)) +
  geom_bar(position="fill",stat="identity",size=1) +
  scale_fill_manual(values=c(topo.colors(4),topo.colors(1))) +
  theme_bw() +
  scale_color_manual(values=rep("black",6))
```



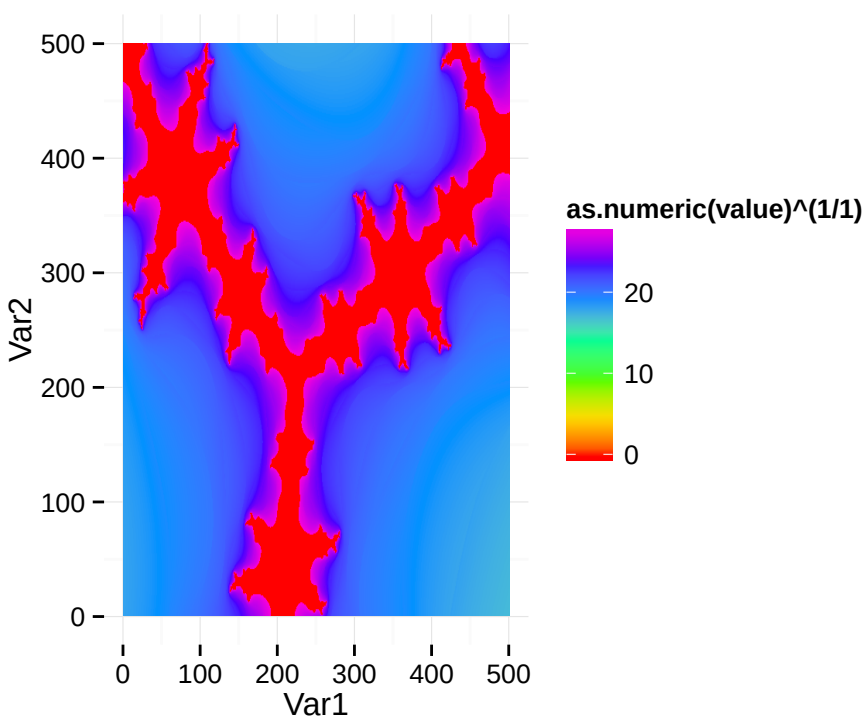
Mandelbrot set

```r
normIter3 = function(n,z,P=2,N=200){n - log( (log(abs(z),base=N)),base=P) }
iterate.until.escape <- function(z, c, trans, cond, max=50, response=dwell) {
  #we iterate all active points in the same array operation,
  #and keeping track of which points are still iterating.
  active <- seq_along(z)
  dwell <- z
  dwell[] <- 0
```

```
  for (i in 1:max) {
    z[active] <- trans(z[active], c[active]);
    survived <- cond(z[active])
    dwell[active[!survived]] <- normIter3(i, z[active[!survived]])
    #dwell[active[!survived]] <- i
    active <- active[survived]
    if (length(active) == 0) break
  }
  eval(substitute(response))
}


getArrayZoom <- function(re.vec=c(-2,1),im.vec=c(-1.5,1.5),max=25,len=500){
  re = seq(re.vec[1], re.vec[2], len=len)
  im = seq(im.vec[1], im.vec[2], len=len)
  c <- outer(re, im, function(x,y) complex(real=x, imaginary=y))
  x <- iterate.until.escape(array(0, dim(c)), c,
                            function(z,c)z^2+c,
                            function(z)abs(z) <= 2,
                            max=max) # was 100
  melt(x)
}

xRel = c(-2,0.5)
yImg = c(-1.5,1.5)

ggplot(getArrayZoom(re.vec=xRel,
                    im.vec=yImg),
       aes(Var1,Var2,fill=as.numeric(value))) +
  geom_raster() +
  scale_fill_gradientn(colours=c("black",(rainbow(7)),na.value="black")) +
  theme_minimal()+
  xlab("pixels of Real values")+
  ylab("imaginary values")
```



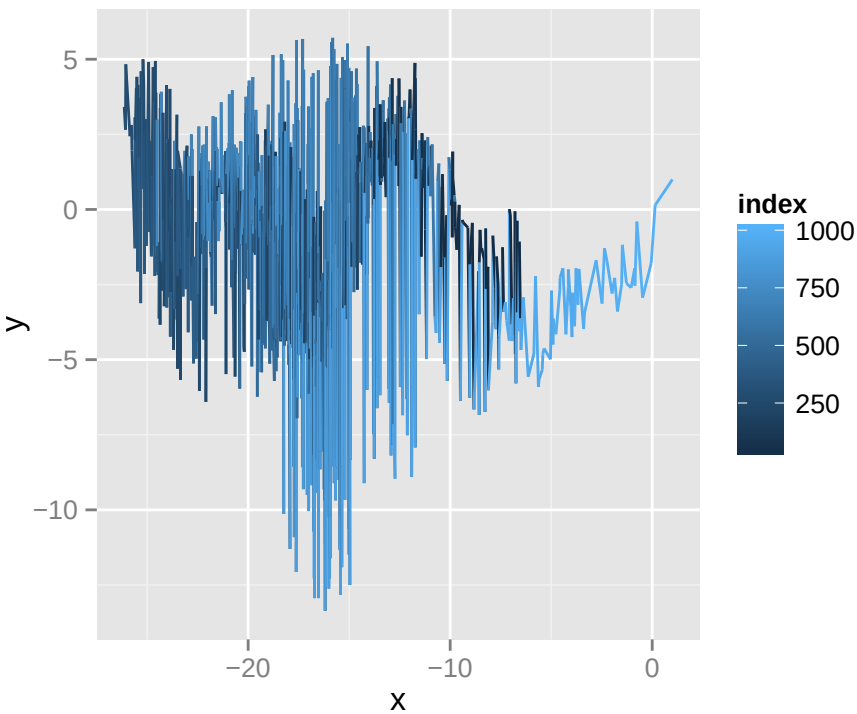

```
ggplot(getArrayZoom(re.vec=c(-0.11,-.09),im.vec=c(0.95,0.965)), aes(Var1,Var2,fill=as.numeric(value)^(1/
  geom_raster() +
  scale_fill_gradientn(colours=rainbow(7),na.value="black") +
  theme_minimal()
```



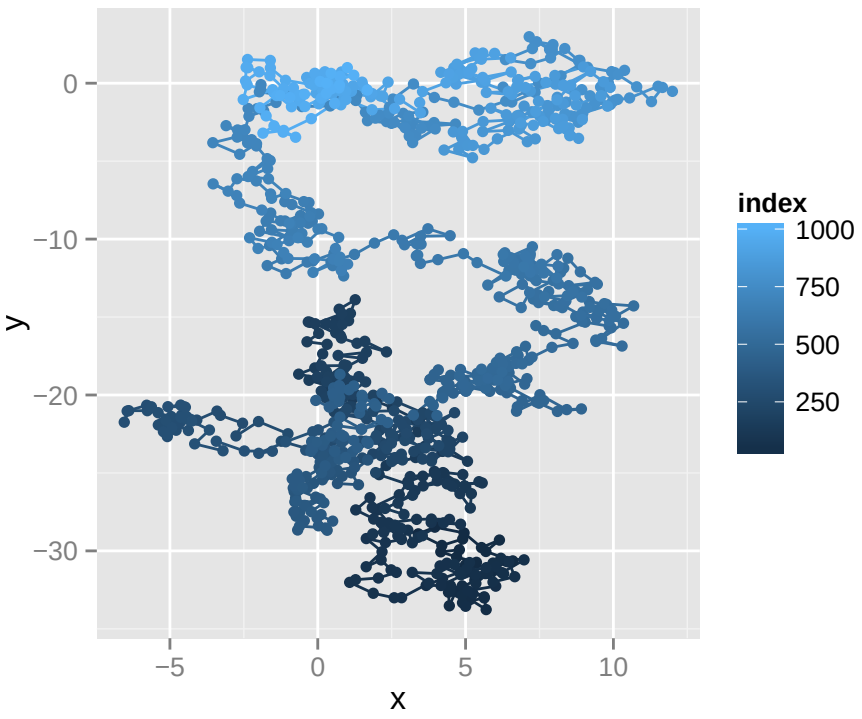
Path

instead of a line, we can follow each subsequent point to the next...

```
# create a random walk:
steps = 500
#' set our reduce function, so we don't have to use loops...
reduceFn = function(x,y)c(x[1]+runif(1)*2 - 1,x)
#' create a function that returns a data.frame with a random walk of "steps" steps
randomWalk  = function(steps)data.frame(x= Reduce(f=reduceFn,1:steps,init=c(1)),
                                        y=  Reduce(f=reduceFn,1:steps,init=c(1)),
                                        index=1:(steps + 1))

#' if you use a line plot, notice that the x values are ordered...
ggplot(randomWalk(1000),aes(x,y,color=index))+geom_line()
```




```
ggplot(randomWalk(1000),aes(x,y,color=index))+geom_path() + geom_point()
```


0
−10
−20
−30
y
−5
0
5
10
x
index
1000
750
500
250