ggplot2 for beginners

Maria Novosolov

1 December, 2014

For this tutorial we will use the data of reproductive traits in lizards on different islands (found in the website)

First thing is to set the working directory to your working directory, read the dataset and see that the names line up with the data we want to use

```
# read the package
library(ggplot2)
# set the working directory
setwd("~/Dropbox/Rworkshop/2014/Databases")
# read the data into R
data<-read.csv("island_type_final2.csv",header=T)
# check the names
names(data)
```

```
##  [1] "species"        "what"           "family"         "insular"
##  [5] "Archipelago"    "largest_island" "area"           "type"
##  [9] "age"            "iso"            "lat"            "mass"
## [13] "clutch"         "brood"          "hatchling"      "productivity"
```

Now we want to start learning the basics of ggplot2 There are two types of plots you can do in ggplot2:

- qplot
- ggplot

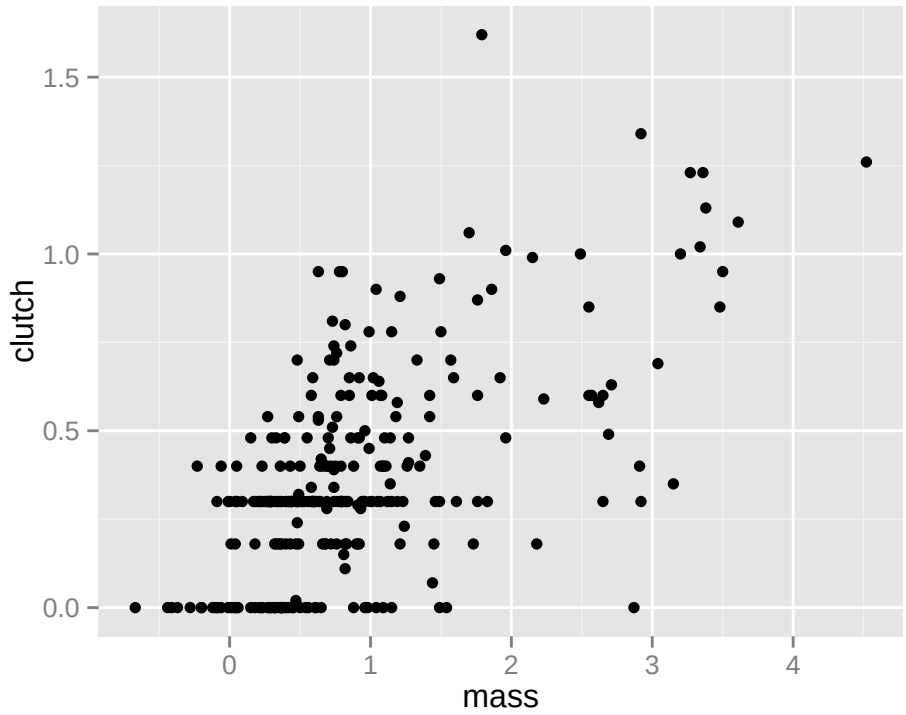
We'll start with the qplot. This is basically the same as the 'plot' function in R but it uses the nicer graphics of ggplot

The general code for a basic qplot

```
qplot(x,y,data=data)
```

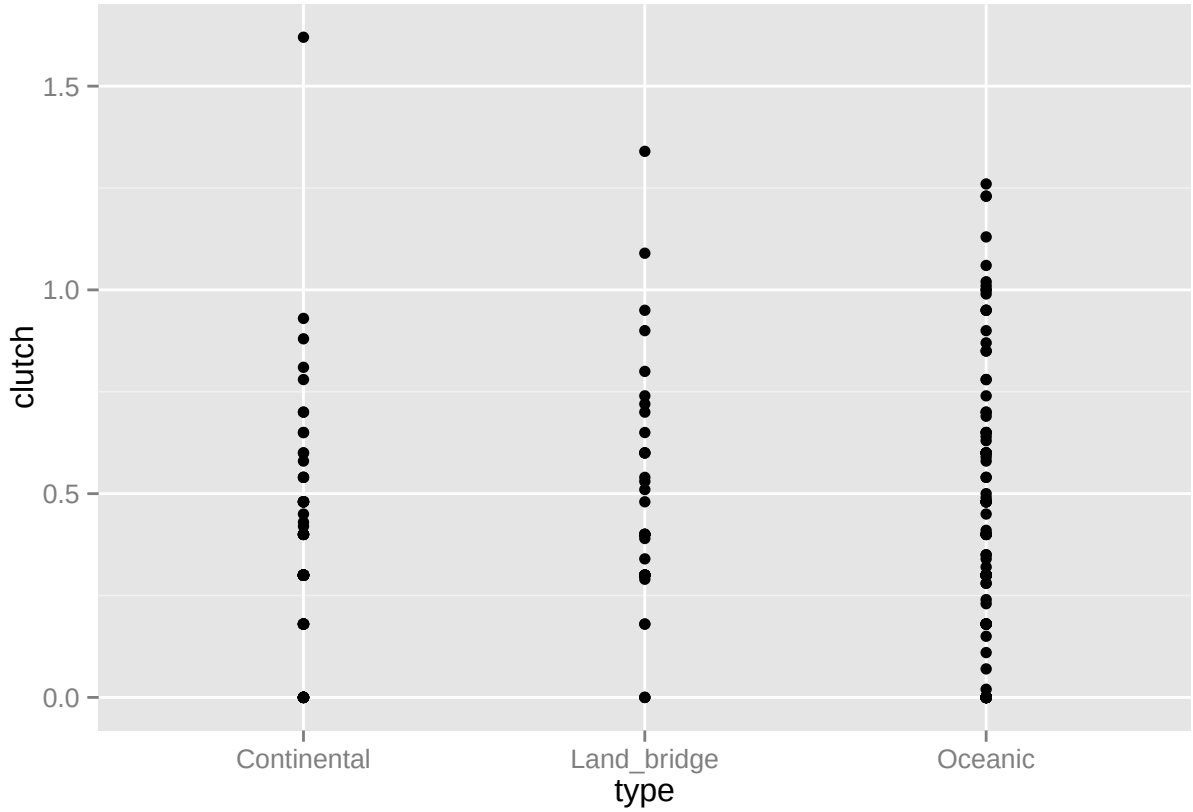
Lets try to see the correlation between clutch size and body mass

```
qplot(mass,clutch,data=data)
```



we can also use a categorical variable as x

```
qplot(type,clutch,data=data)
```



 # the default option is scatterplot so if we want to have a different plot we have to specify which

ggplot function is an easier solution to manipulate the plot to look like we want Lets see what it knows

ggplot2 in action

Let's start with the basic. Remember that all the plot have to have:

1. X
2. Y
3. data
4. Geom

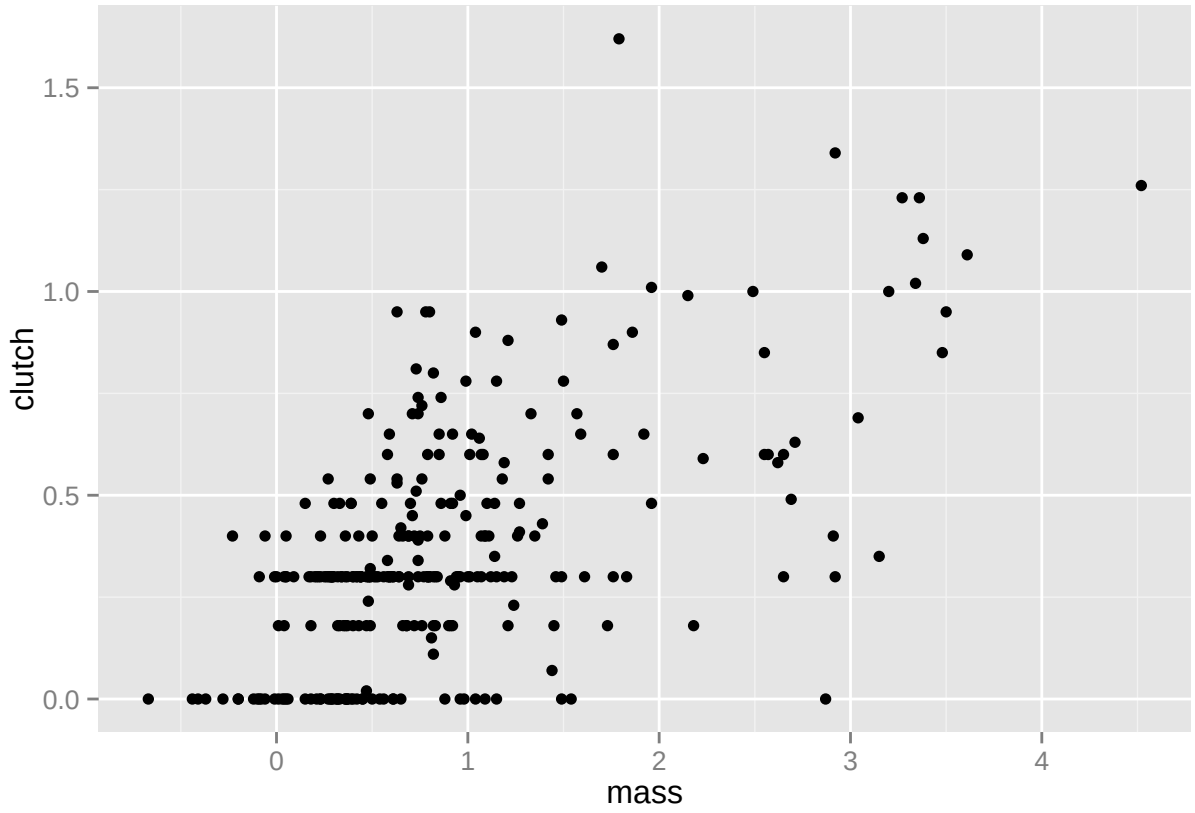
the general code and the basic things you mist have to create a plot is

`ggplot(your.data,aes(x,y))+geom_your.geom()`

This basic code works for different types of geoms though in some geoms you'll need to specify more things before you can run the plot

Let's plot the same plot as our first *qplot* using *ggplot*

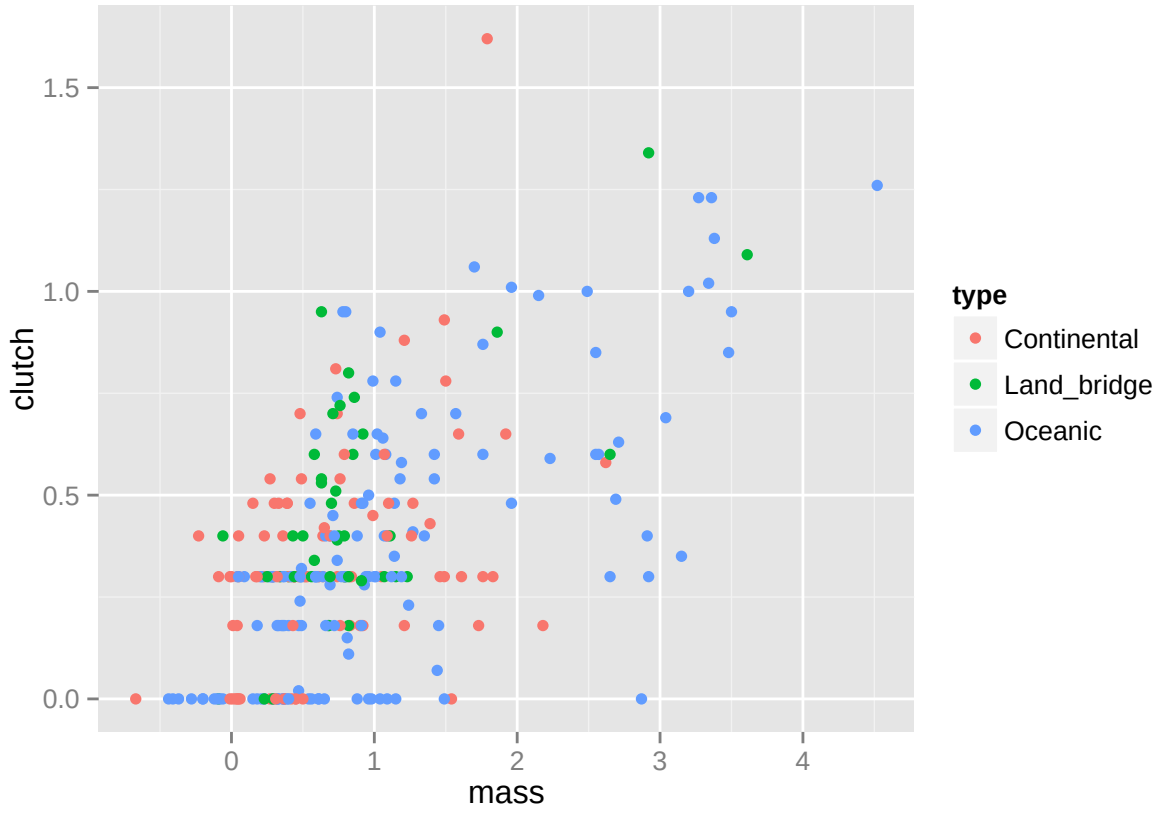
```
ggplot(data, aes(mass, clutch)) + geom_point()
```




```
# geom point tells it to plot a scatter plot
```

Now let's add some color to the scatterplot, let's give different colors to different types of islands we will do this by adding aesthetics to the geom

```
ggplot(data, aes(mass, clutch)) +
  geom_point(aes(color=type))
```




```
# aes will let us change colors, shapes and fill
```

 # important to remember that fill will be the inner color and color will represent the outline of the shape or the bar

Now let's add some shape - each type will get a different shape this can be done by either adding aesthetics to the geom or to the ggplot itself

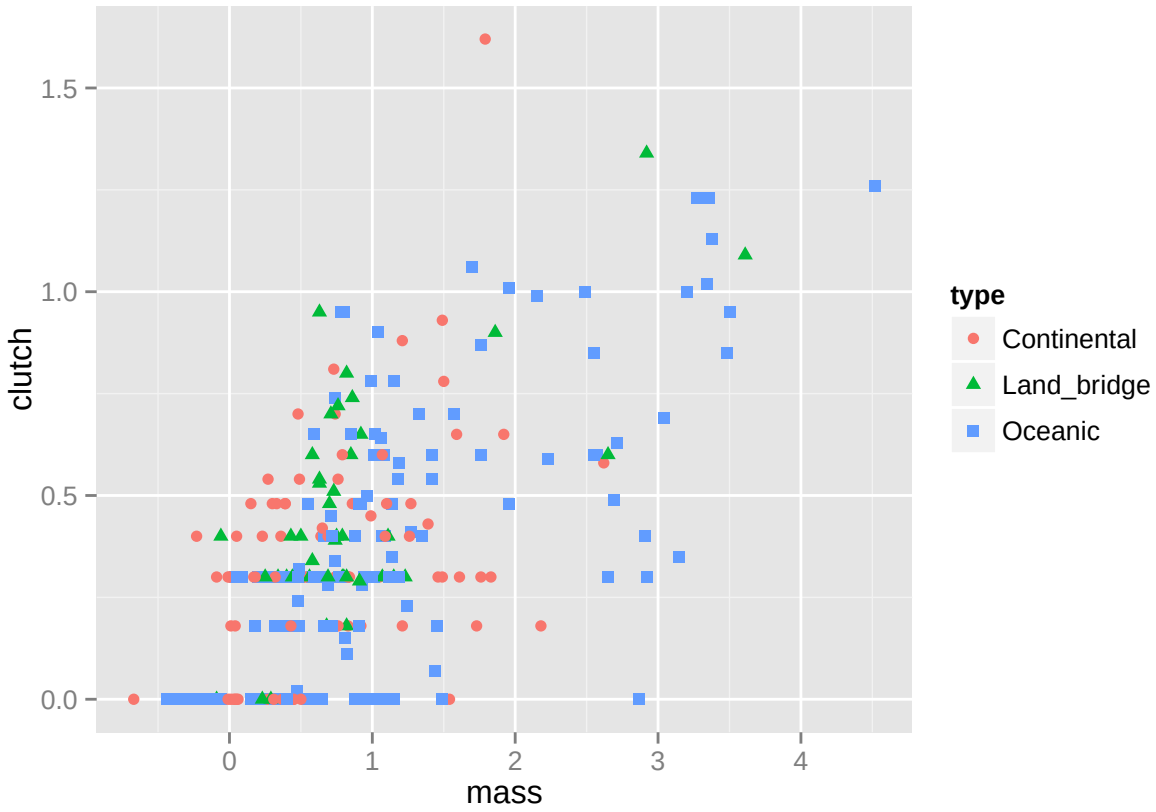
```
ggplot(data, aes(mass, clutch)) + geom_point(aes(color=type,shape=type))
```

OR

```
ggplot(data, aes(mass, clutch,shape=type)) + geom_point(aes(color=type))
```

Let's see the output of one of them

```
ggplot(data, aes(mass, clutch,shape=type)) +
  geom_point(aes(color=type))
```



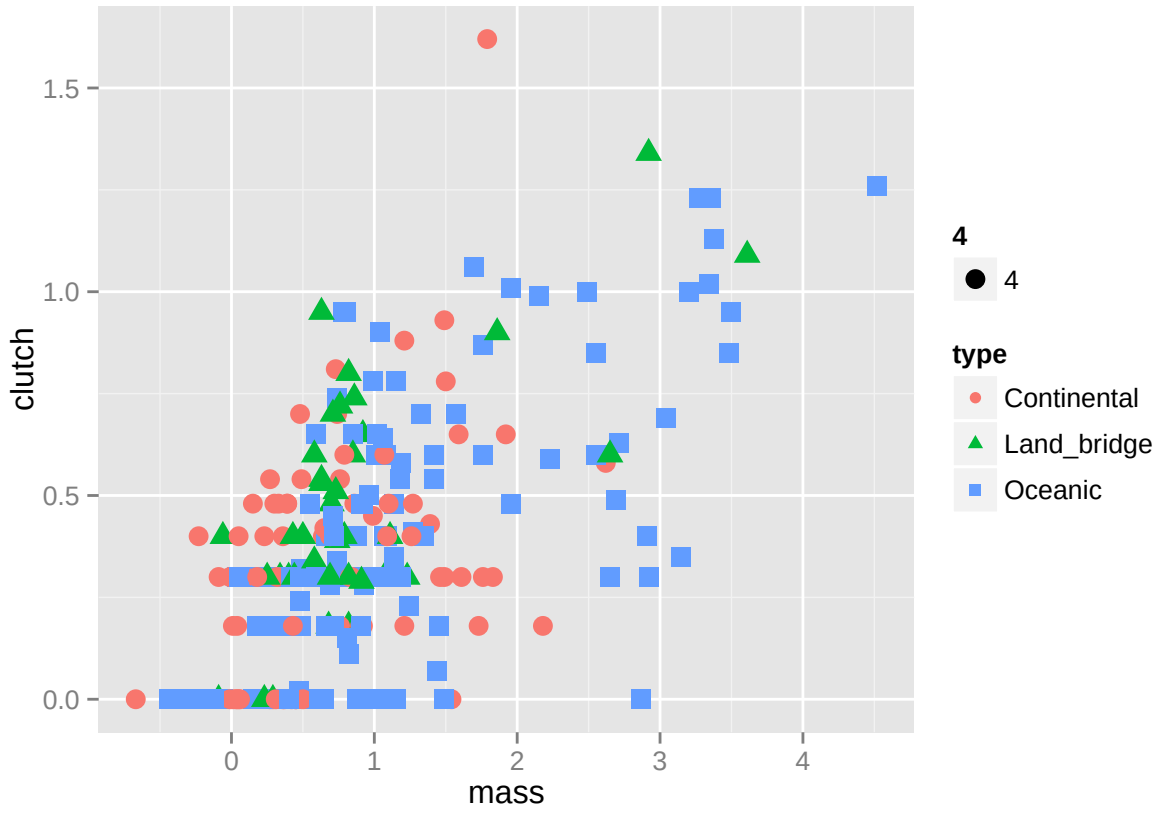
If the shape is too small. We can change the size by adding size to the aesthetics in the ggplot

General code

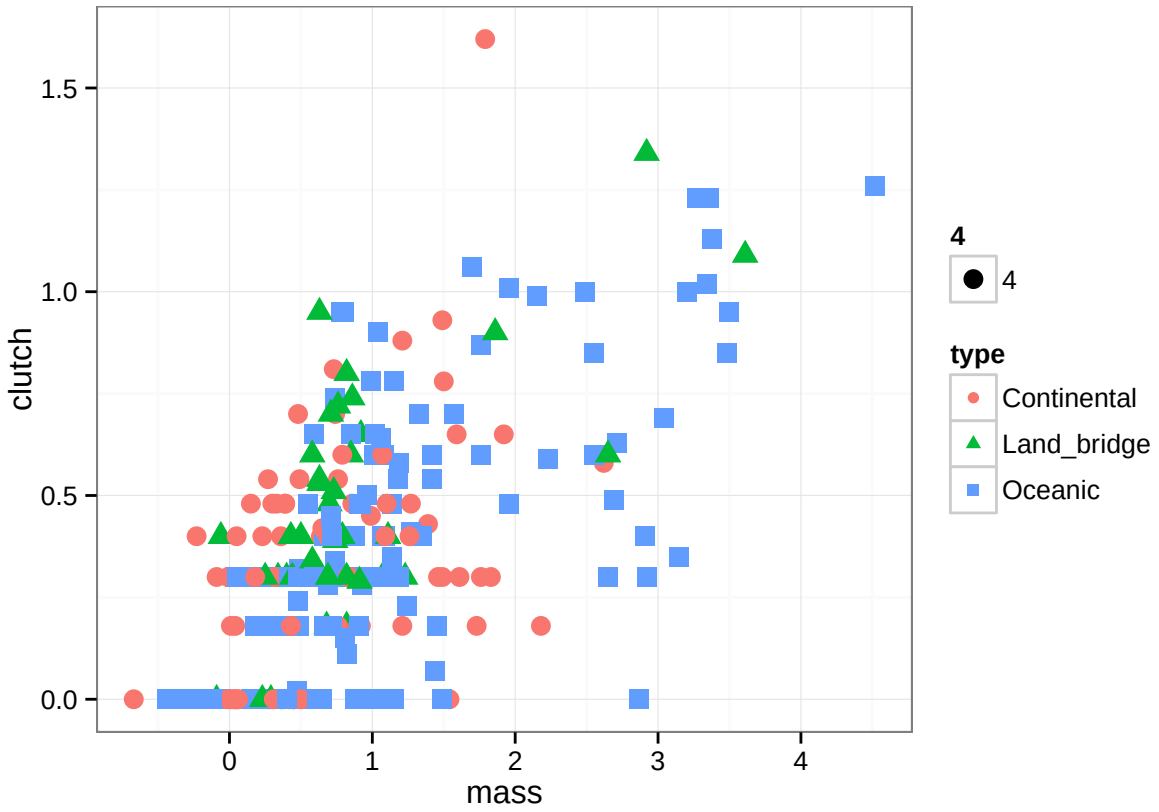
```
ggplot(your.data, aes(x_variable, y_variable,size=a.number.for.the.size)) +geom_point(aes(color=categor
```

If we put it to test with our data

```
p <- ggplot(data, aes(mass, clutch)) +
  geom_point(aes(color=type,shape=type,size=4))
p
```




```
# I also change the background to white using the theme function
p+theme_bw()
```



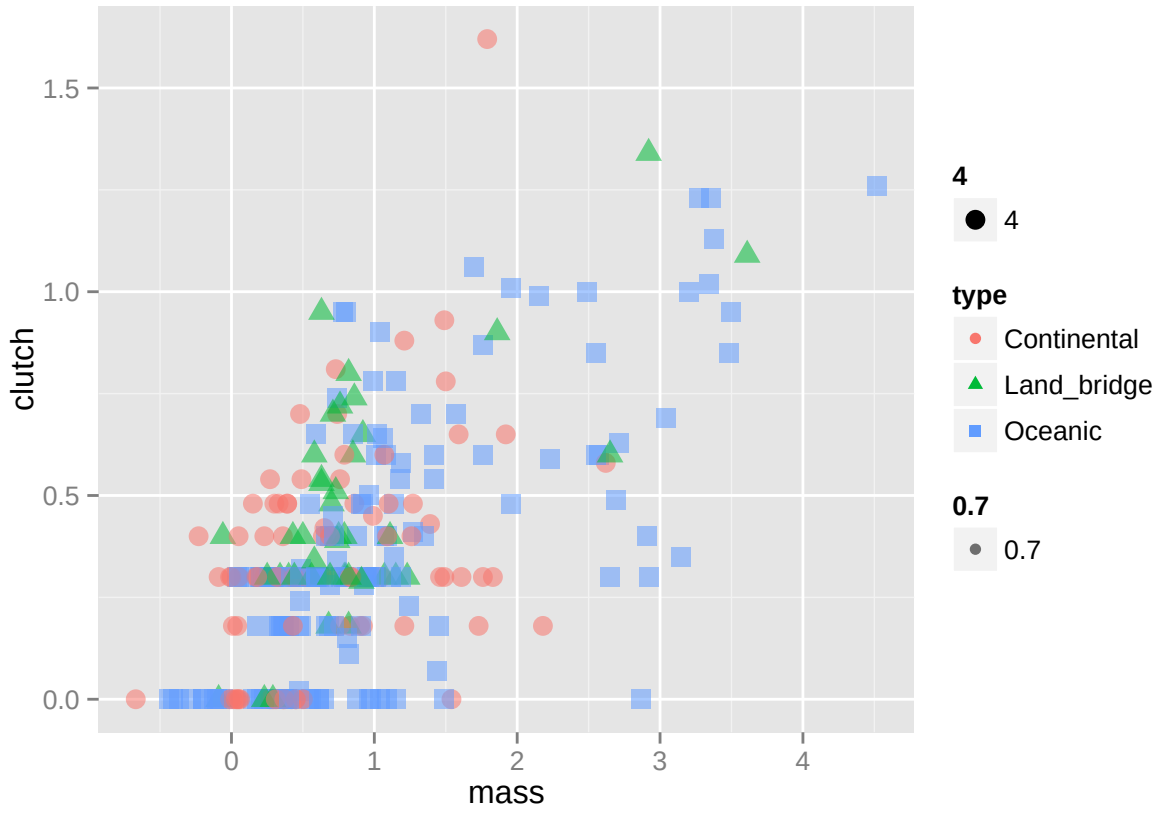
 # notice that I save the plot in a variable 'p' to reuse it in an easier way

If you have too many point and you want them all to be more seen you can control the transparency of the dots this you do with the 'alpha' which should be in the aesthetics of the ggplot and can be any value between 0 and 1

```
ggplot(your.data, aes(x_variable, y_variable,size=z_variable_continuous,alpha=number.between.0.and.1))
+ geom_point(aes(color=categorical_variable))+ scale_size_area(max_size=the.maximum.size.you.want)
```

In our example: Lets see the mass vs. the latitude with clutch size as the size of the dot

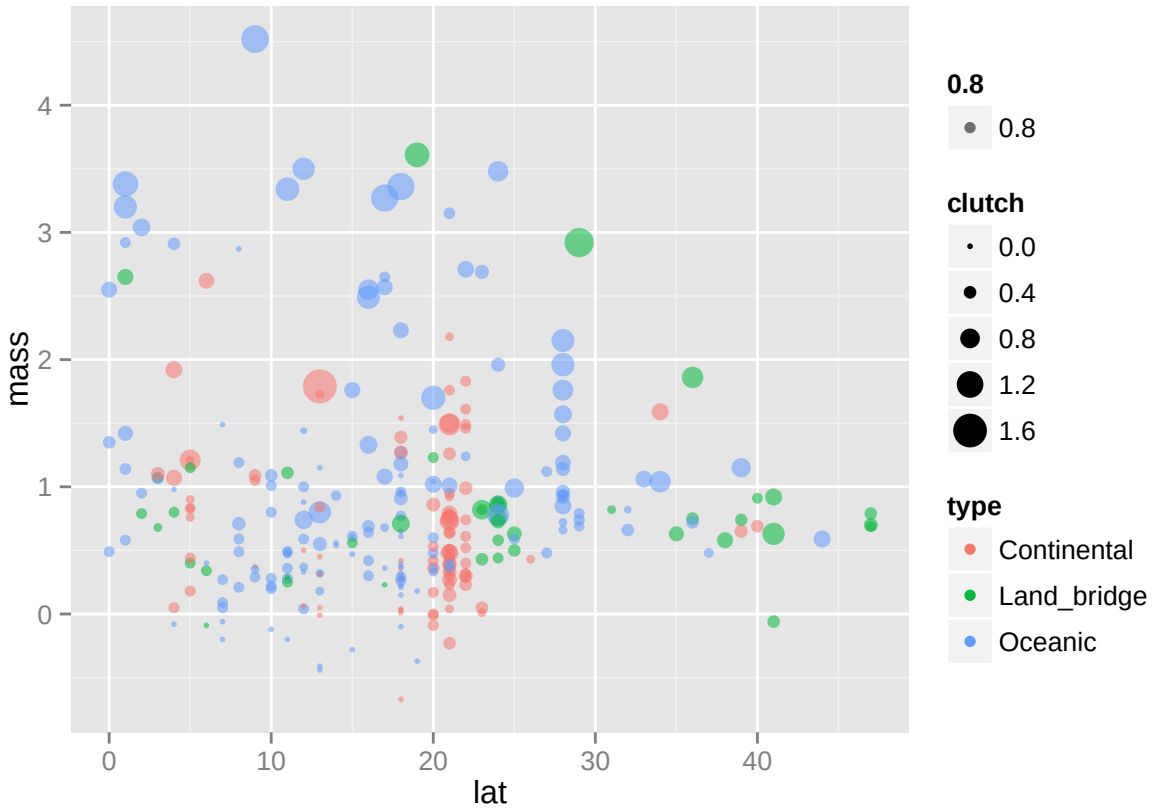
```
p<-ggplot(data, aes(mass, clutch)) +
  geom_point(aes(color=type,shape=type,size=4,alpha = 0.7))
p
```



Scales

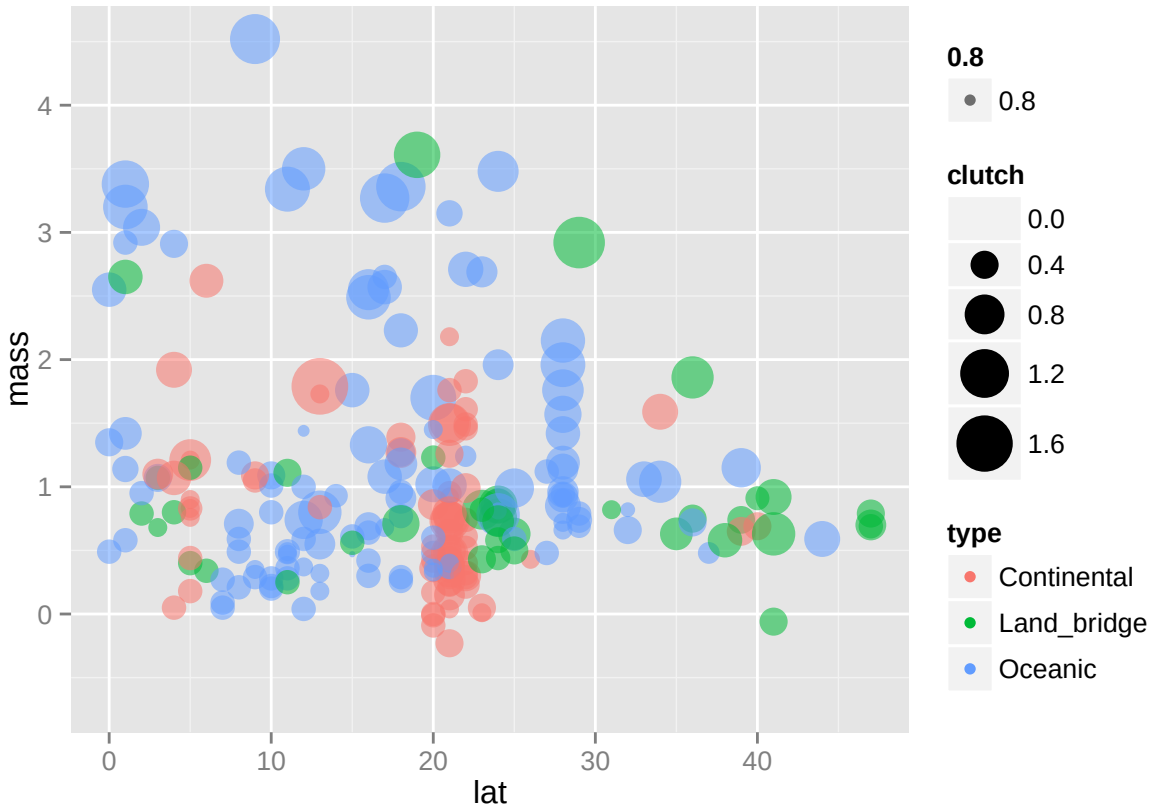
the size can also symbolize the number of a measurement for each dot. for example: we want to see the mass vs latitude while looking at the clutch size for each mass variable for this we will specify that the size is the clutch size

```
p <- ggplot(data, aes(lat, mass,size=clutch,alpha=0.8)) +
  geom_point(aes(color=type))
p
```



If we think it is too small we can change the size of the points with scale_size_area

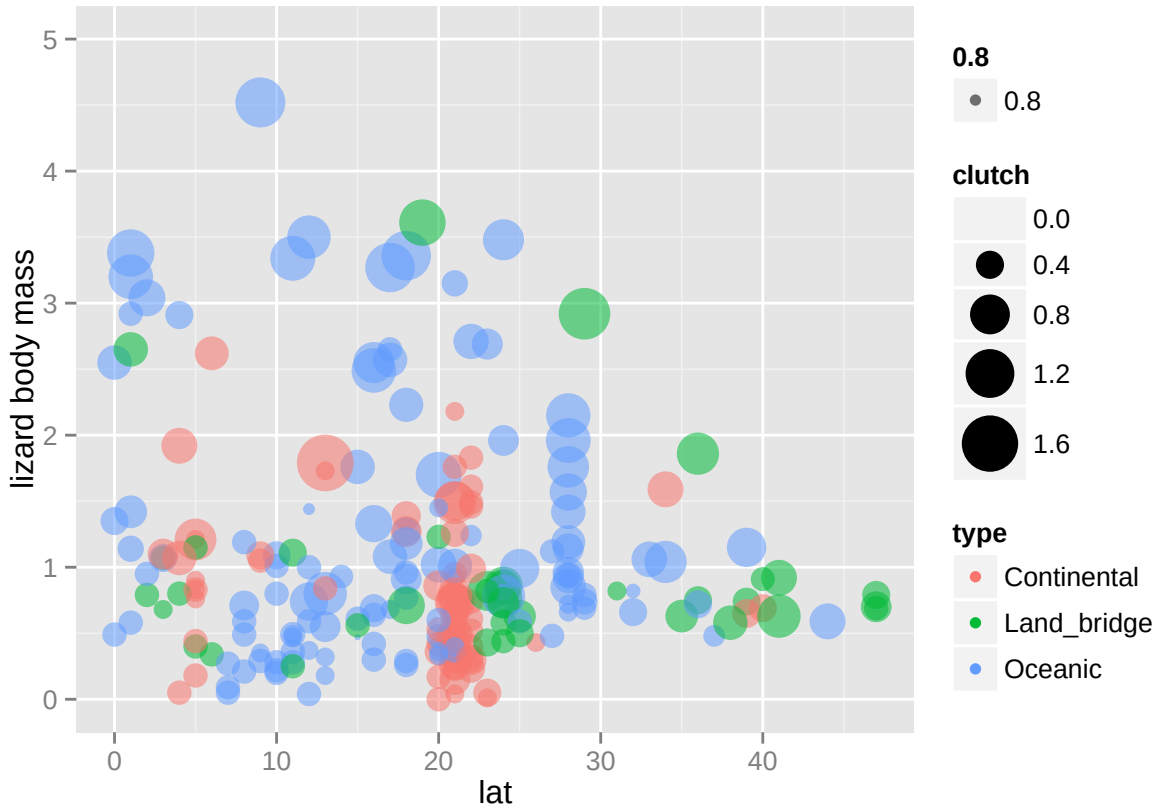
```
p <- ggplot(data, aes(lat, mass,size=clutch,alpha=0.8)) +
  geom_point(aes(color=type))+scale_size_area(max_size=10)
p
```



It is possible to change the default axis by using scale_x/y Using this you can change the names, the range, to specify the axis as a date or as time to change the name and the range of the y axis you use scale_y_continuous

In this case we are changing the name (writing the name between "name") and the range (using the limit function)

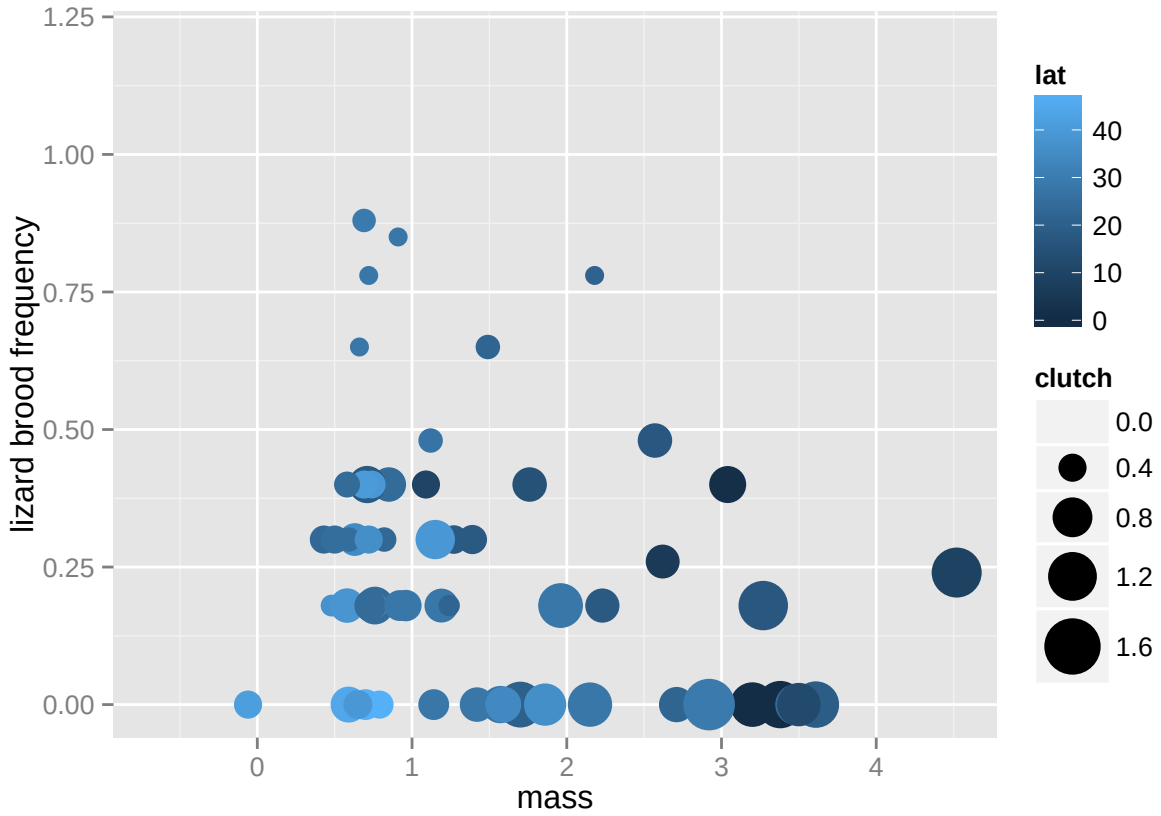
```
p<-ggplot(data, aes(lat, mass,size=clutch,alpha=0.8)) +
  geom_point(aes(color=type))+
  scale_size_area(max_size=10)+
  scale_y_continuous("lizard body mass",limit=c(0,5))
p
```



We can also scale the color. If you are interested to make the color continuous according to a specific Variable than you just enter the name of the variable. if you want to set the color manually you just use scale_color_manual and set the colors you want you can choose a set of colors or a specific color for each category there is a big variation of colors that you can choose. Take a look at the website http://sape.inf.usi.ch/quick-reference/ggplot2/colour

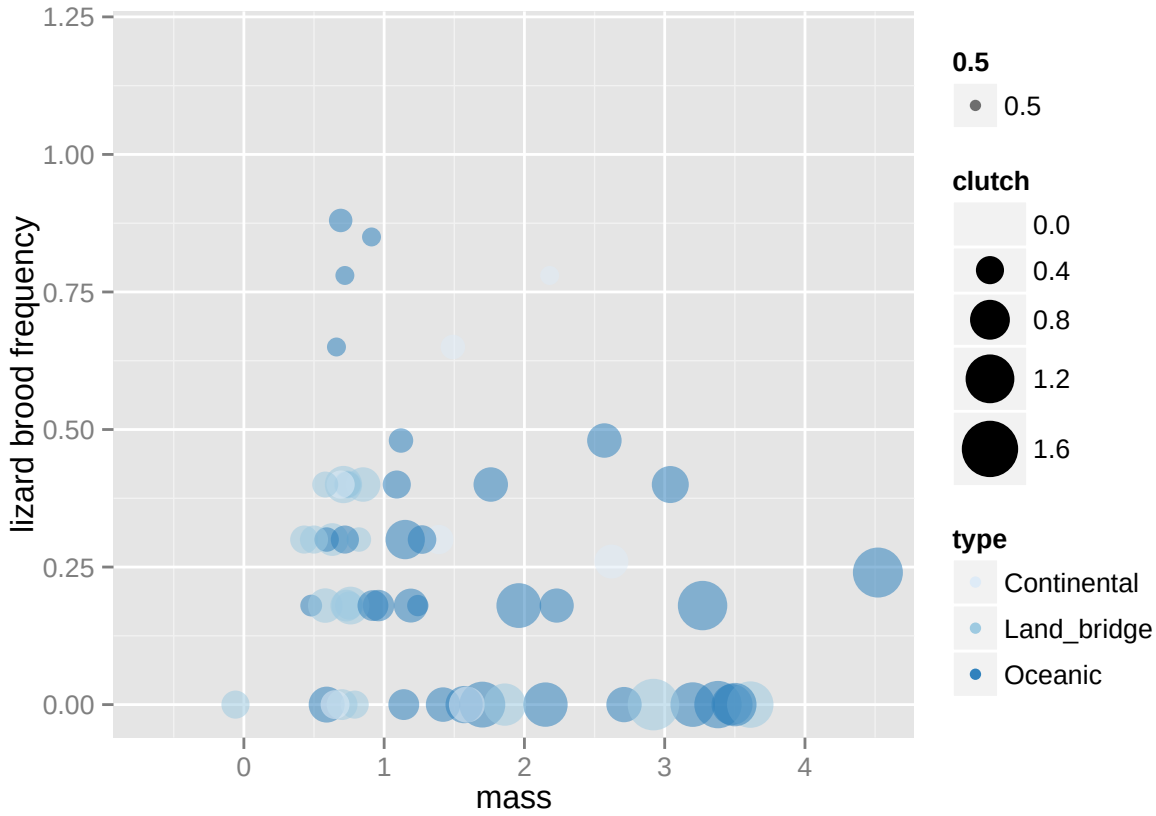
we can use a continuous variable as color. This will create a continuous colors

```
p<-ggplot(data, aes(mass, brood,size=clutch,color=lat)) +
  geom_point(aes())+
  scale_size_area(max_size=10)+
  scale_y_continuous("lizard brood frequency",limit=c(0,1.2))
p
```



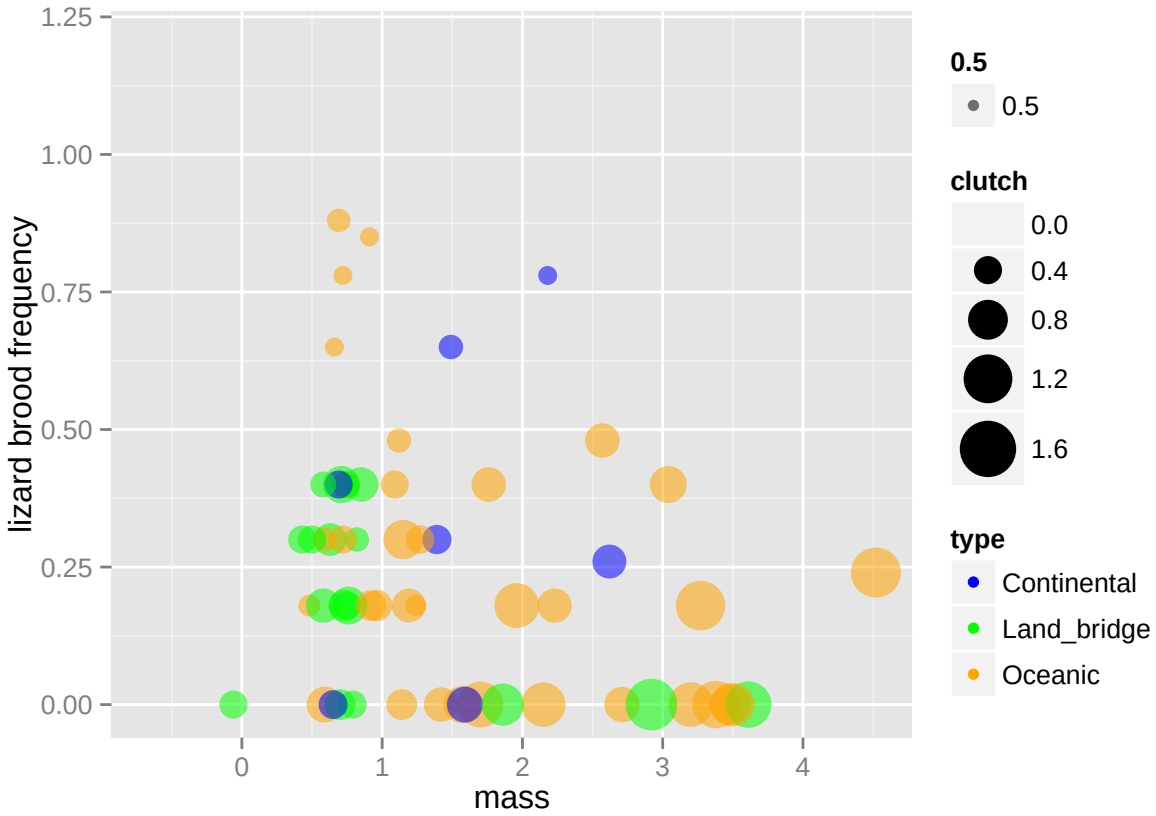
but if we have a categorical data that we want to become with continuous colors we can use the scale_colour_brewer

```
p<-ggplot(data, aes(mass, brood,size=clutch,color=type,alpha=0.5)) +
  geom_point(aes())
p<-p+scale_size_area(max_size=10)
p<-p+scale_y_continuous("lizard brood frequency",limit=c(0,1.2))+
  scale_colour_brewer()
p
```



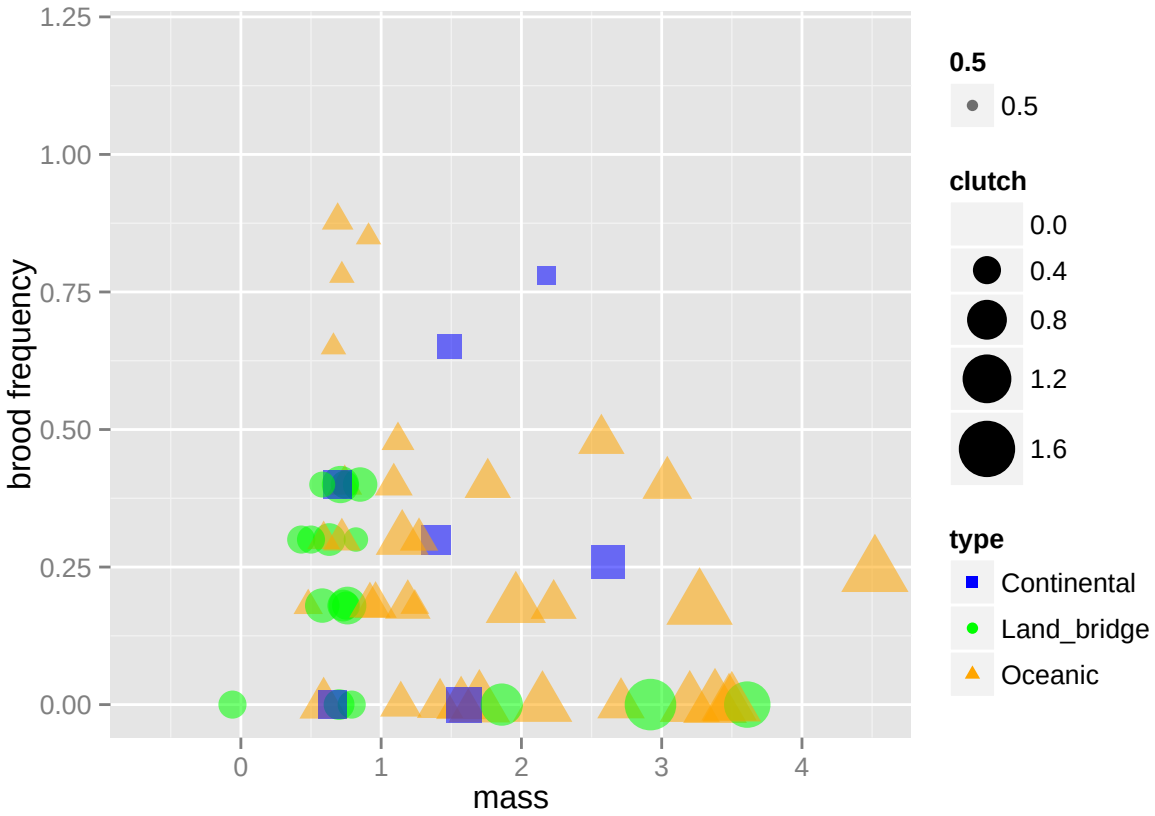
OR we can set the colors for each category manually

```
p<-ggplot(data, aes(mass, brood,size=clutch,color=type,alpha=0.5)) +
  geom_point(aes())
p<-p+scale_size_area(max_size=10)+
  scale_y_continuous("lizard brood frequency",limit=c(0,1.2))+
  scale_colour_manual(values=c("blue","green","orange"))
p
```



REMEMBER: the number of colors you provide have to be as the number of categories you have In the same way you can scale the shape of your data points. this works only for categorical variables. you can either give R to choose which shapes to use or you can do it manually with manual size scaling you can also use Unicode shapes (just google Unicode shapes) make sure always to add what you are scaling in the ggplot line

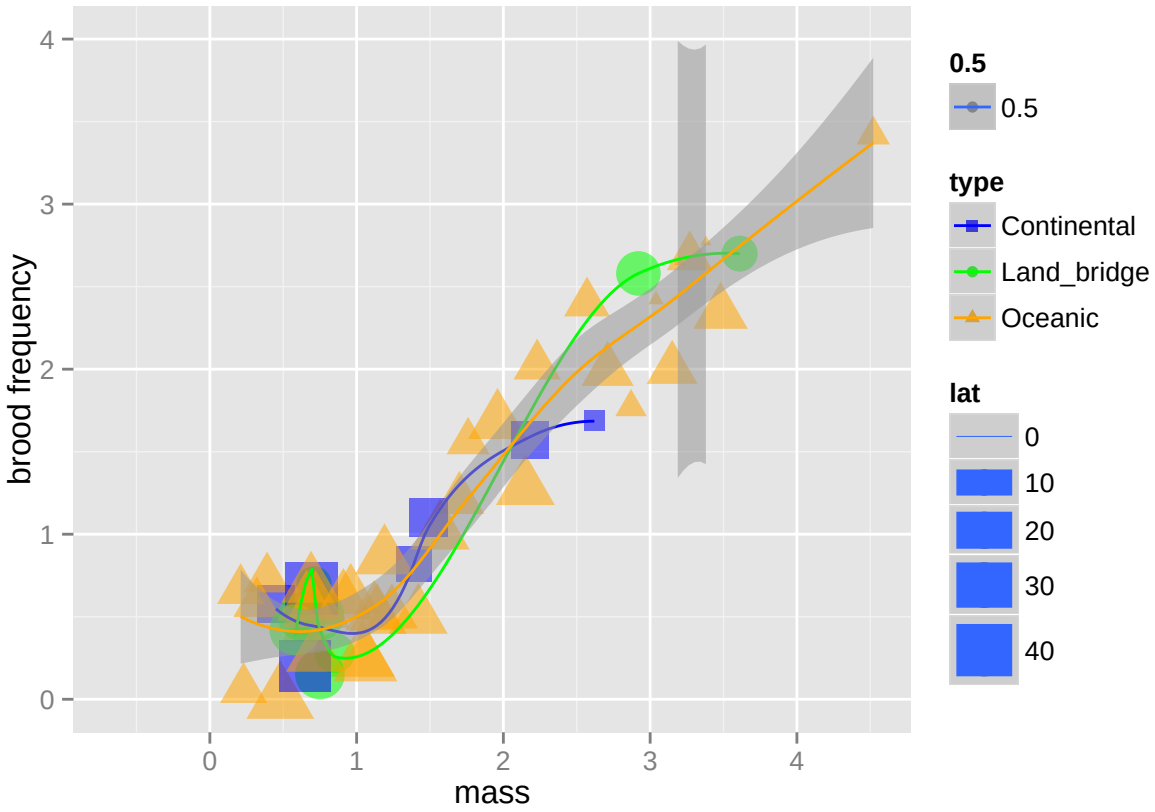
```
ggplot(data, aes(mass, brood,shape=type,size=clutch,color=type,alpha=0.5)) +
  geom_point(aes())+
  scale_size_area(max_size=10)+
  scale_y_continuous("brood frequency",limit=c(0,1.2))+
  scale_colour_manual(values=c("blue","green","orange"))+
  scale_shape_manual(values=c(15,16,17))
```



to the scatterplot we can add the area of error by adding "statistics" adding "smooth" statistics will add an error area to the plot

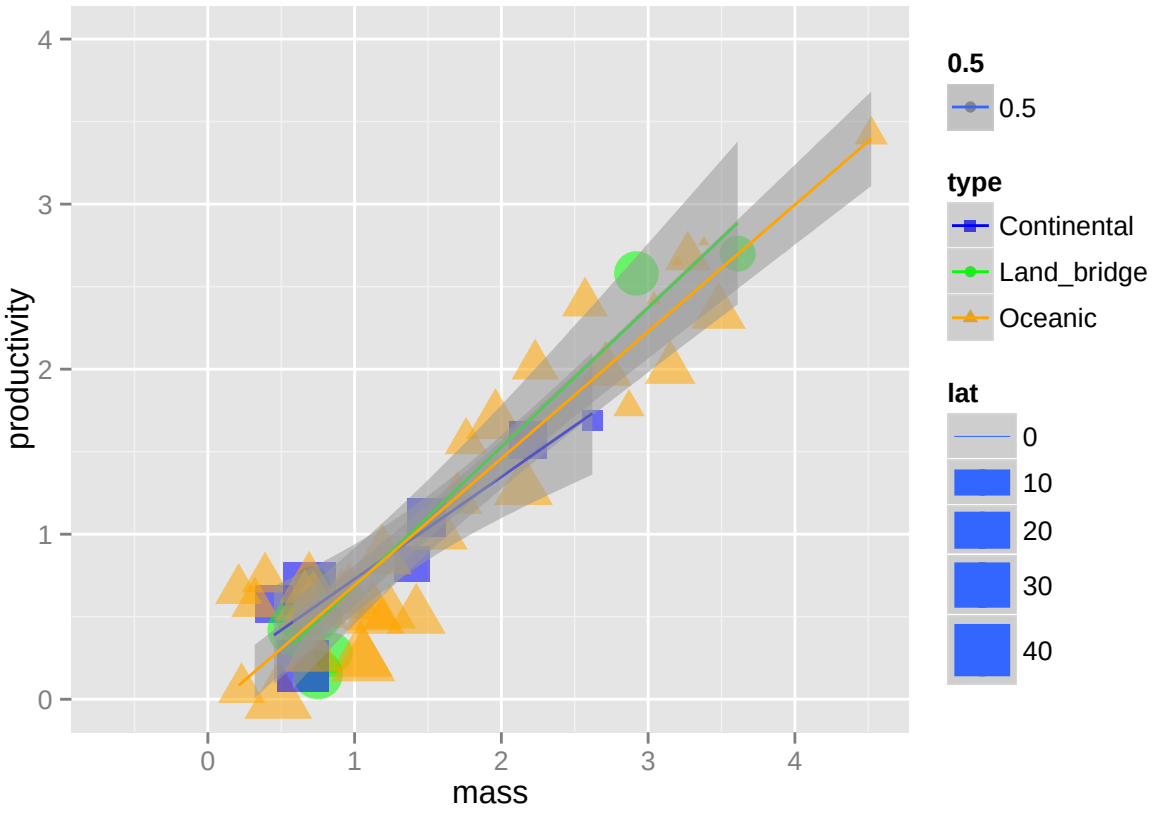
```
p<-ggplot(data, aes(mass, productivity,shape=type,size=lat,color=type,alpha=0.5)) +
  geom_point(aes())
p<-p+scale_size_area(max_size=10)+
  scale_y_continuous("brood frequency",limit=c(0,4))+
  scale_colour_manual(values=c("blue","green","orange"))+
  scale_shape_manual(values=c(15,16,17))
p<-p+stat_smooth()
p
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method = x' to cl
```



if you want it to be a straight line rather than a "connecting the dots" line you need to specify with method="lm"

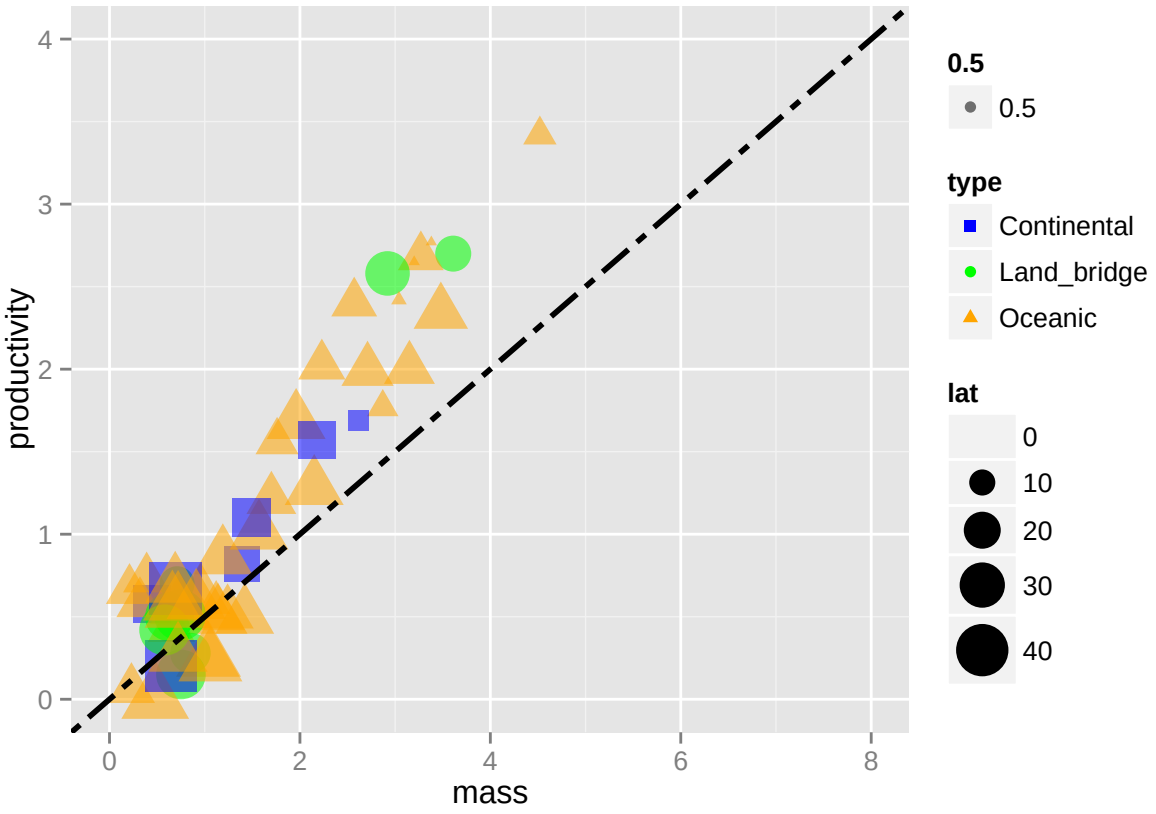
```
p<-ggplot(data, aes(mass, productivity,shape=type,size=lat,color=type,alpha=0.5)) +
  geom_point(aes())
p<-p+scale_size_area(max_size=10)+
  scale_y_continuous("productivity",limit=c(0,4))+
  scale_colour_manual(values=c("blue","green","orange"))+
  scale_shape_manual(values=c(15,16,17))
p<-p+stat_smooth(method="lm") #uses the full range of the plot
p
```



Geoms

We can change different geoms. Each one has a bit of different things that you can change but the idea stays the same. We can use a geom to specify a regression line that will fit our model (with a specific intercept and slope)

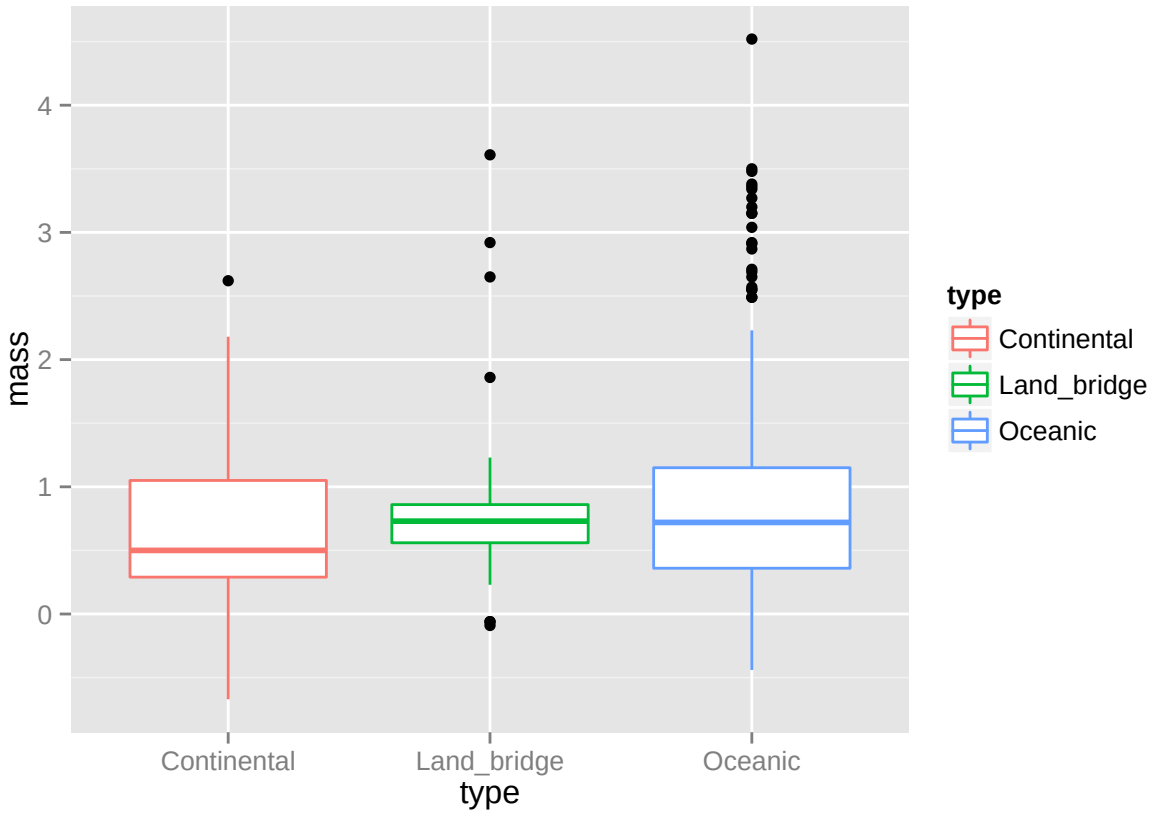
```
ggplot(data, aes(mass, productivity,shape=type,size=lat,color=type,alpha=0.5)) +
  geom_point(aes())+
  scale_size_area(max_size=10)+
  scale_y_continuous("productivity",limit=c(0,4))+
  scale_x_continuous(limit=c(0,8))+
  scale_colour_manual(values=c("blue","green","orange"))+
  scale_shape_manual(values=c(15,16,17))+
  geom_abline(intercept = (0),slope=0.5,linetype =6, size = 1)
```



Boxplot

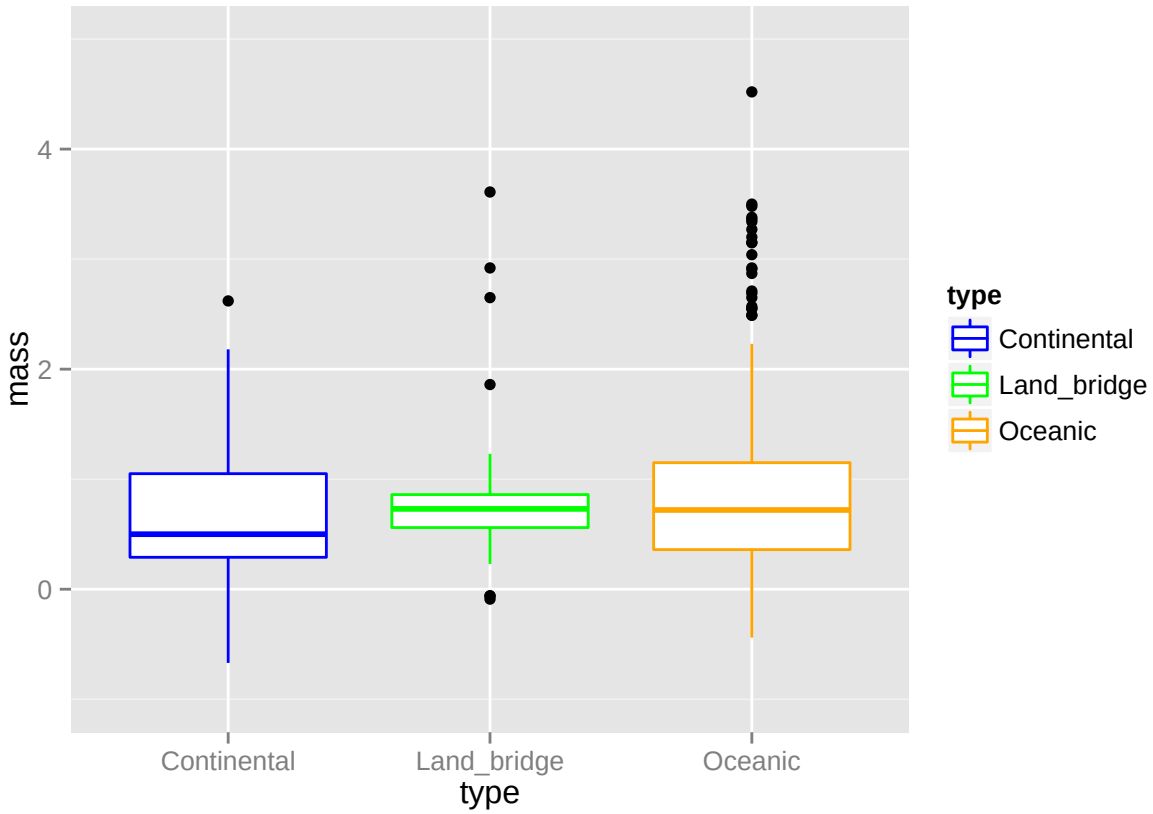
we can use the general boxplot and just say to R to make different colors to each species and change transparency

```
ggplot(data, aes(type, mass,color=type)) +
  geom_boxplot(aes())
```



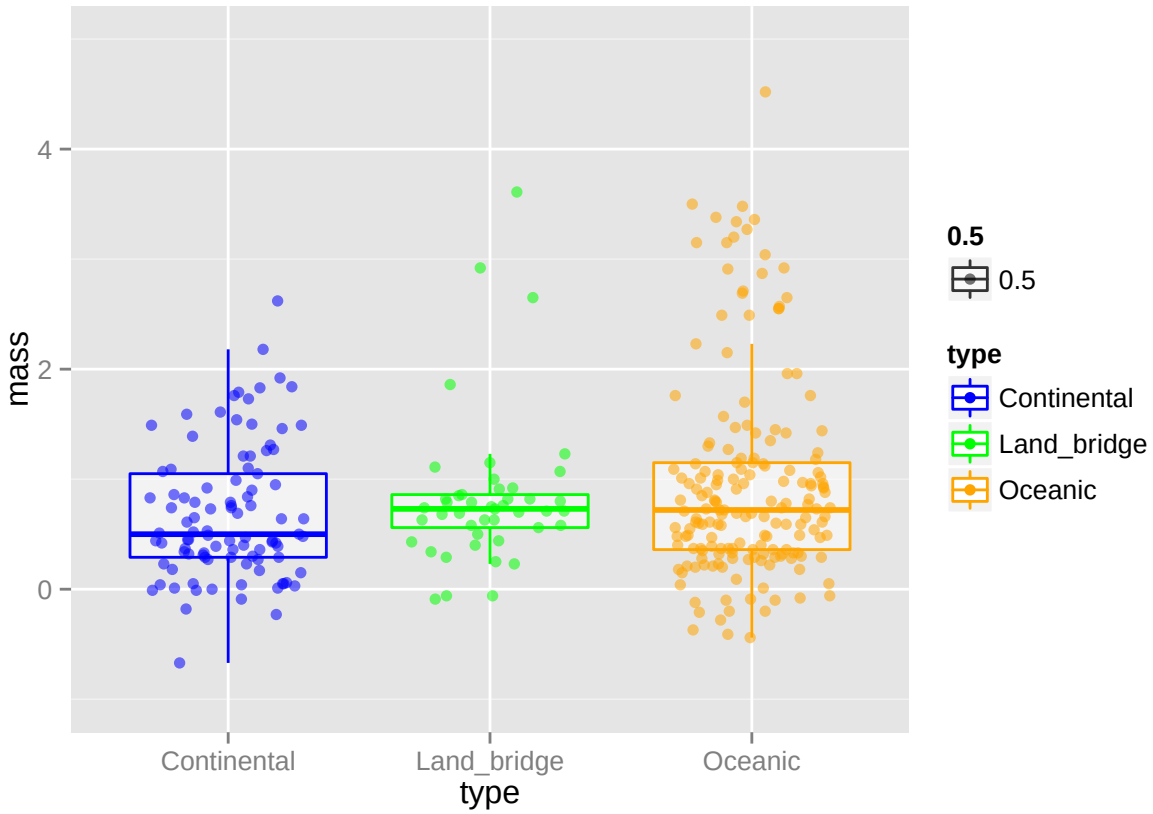
But we can also do the scaling that we used before as scaling the y axis or the color

```
ggplot(data, aes(type, mass,color=type)) +
  geom_boxplot(aes())+
  scale_y_continuous("mass",limit=c(-1,5))+
  scale_colour_manual(values=c("blue","green","orange"))
```



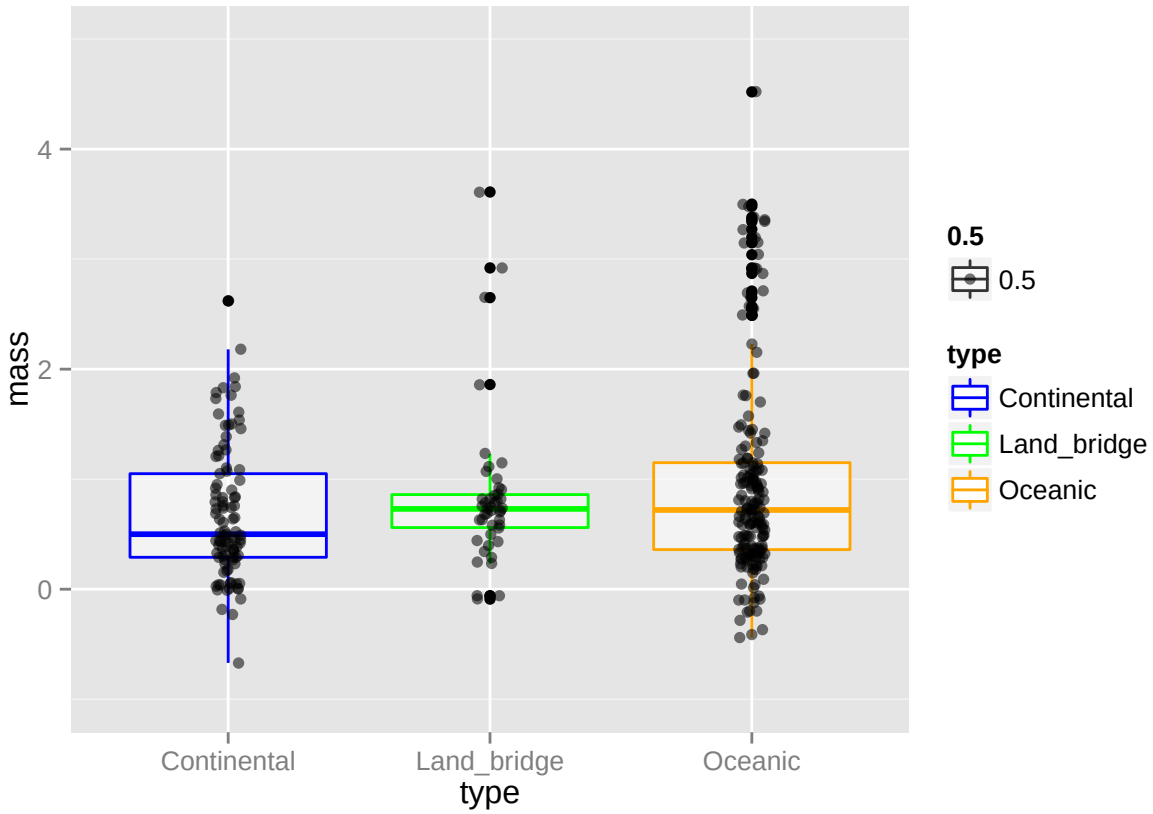
if we want to see the distribution of data point for each box we can use the jitter geom

```
ggplot(data, aes(type, mass,color=type,alpha=0.5)) +
  geom_boxplot(outlier.shape = NA)+    #remove outliers for jitter
  geom_jitter(position=position_jitter(width=0.3, height=0))+
  scale_y_continuous("mass",limit=c(-1,5))+
  scale_colour_manual(values=c("blue","green","orange"))
```



we can change the position of the points by adding position to the jitter

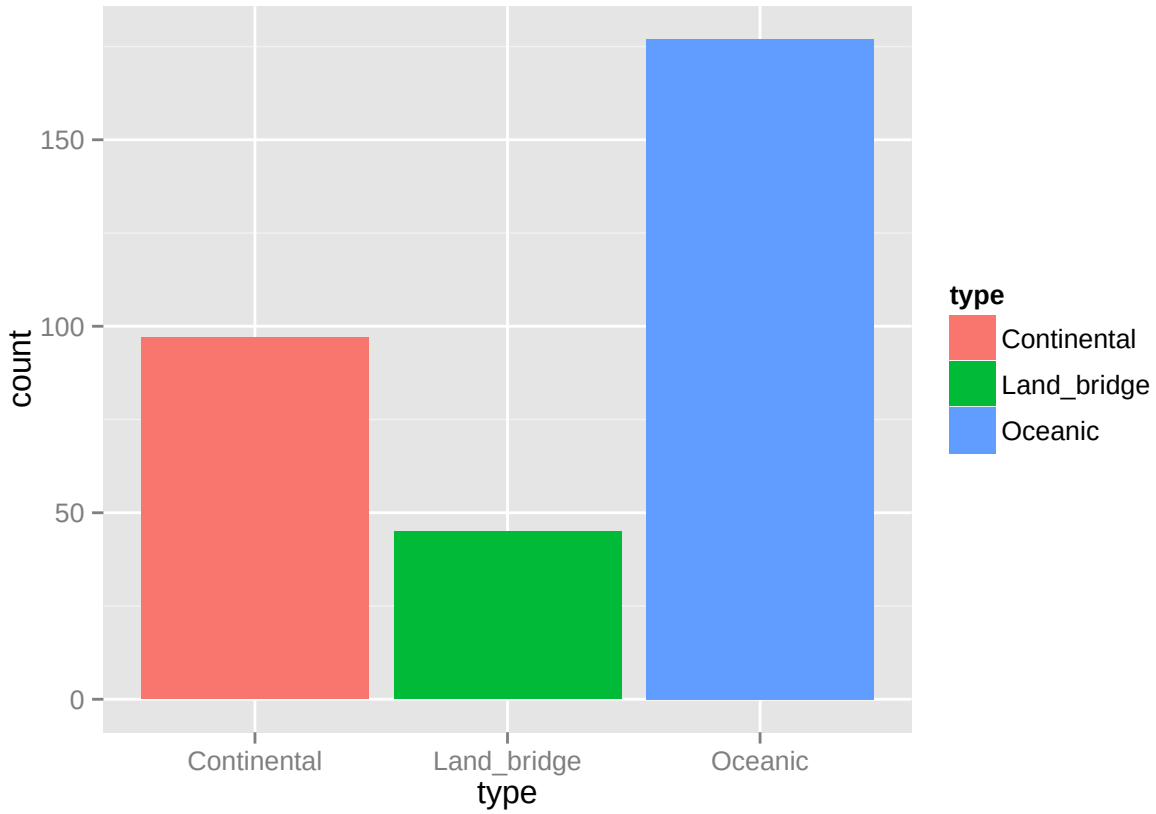
```
ggplot(data, aes(type, mass,color=type,alpha=0.5)) +
  geom_boxplot(aes())+
  geom_jitter(color="black",position=position_jitter(width=0.05))+ #putting the "black" here makes sure
  scale_y_continuous("mass",limit=c(-1,5))+
  scale_colour_manual(values=c("blue","green","orange"))
```



Barplot with error bars

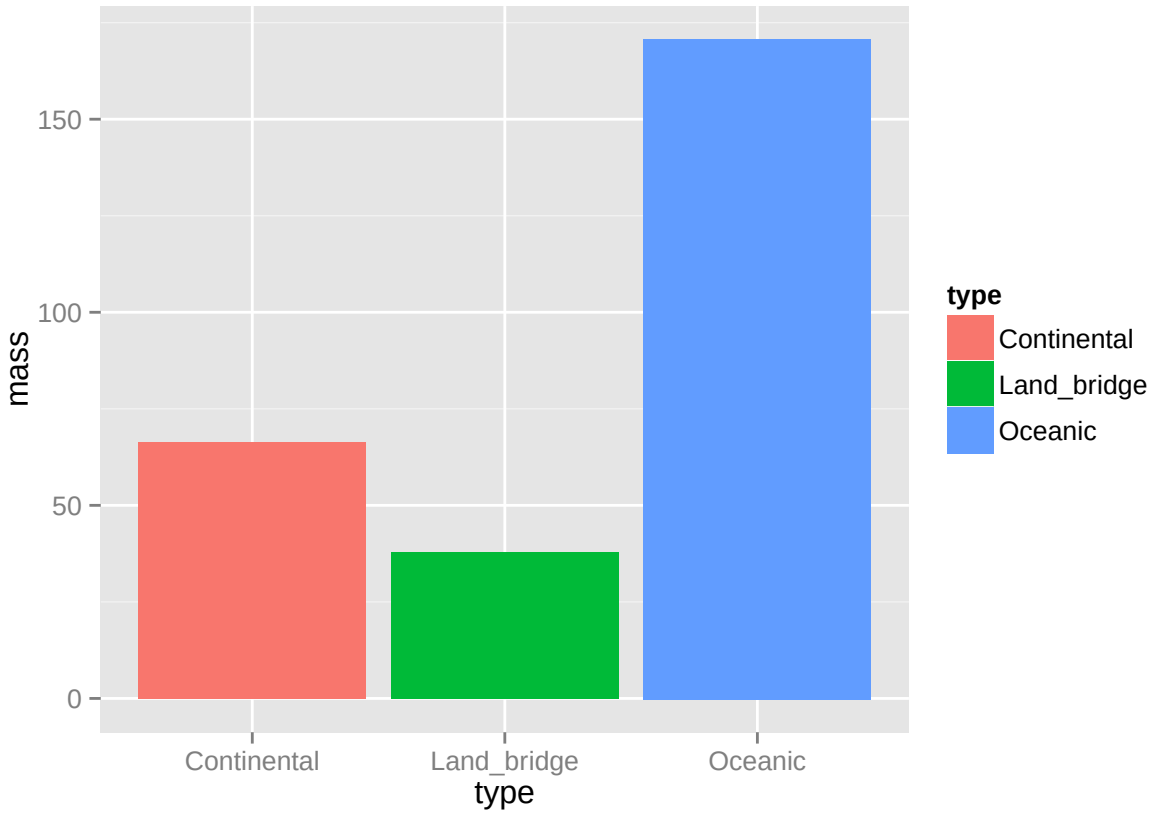
The default of bar plot is to create a bin in the y axis showing the count showing the number of observations for each category in the x variable

```
ggplot(data, aes(type,fill=type)) + # fill allows us to change the color for each type category
  geom_bar()
```



If you are interested for it to show the count of a specific continuous variable for each category you need to specify it by using stat=“identity”

```
ggplot(data, aes(type,mass,fill=type)) +
  geom_bar(stat = "identity")
```



If you want to see the mean of a variable you first need to calculate the mean and the SE using the ddply function and the packages plotrix or sciplot

```
library(plyr)
library(plotrix)
library(sciplot)
```

lets calculate the mean mass for each type of island and the SE

using 'plotrix'

```
barplot.data<-ddply(data,.(type),summarize,mean.mass=round(mean(mass),3),se.mass=round(std.error(mass),3
```

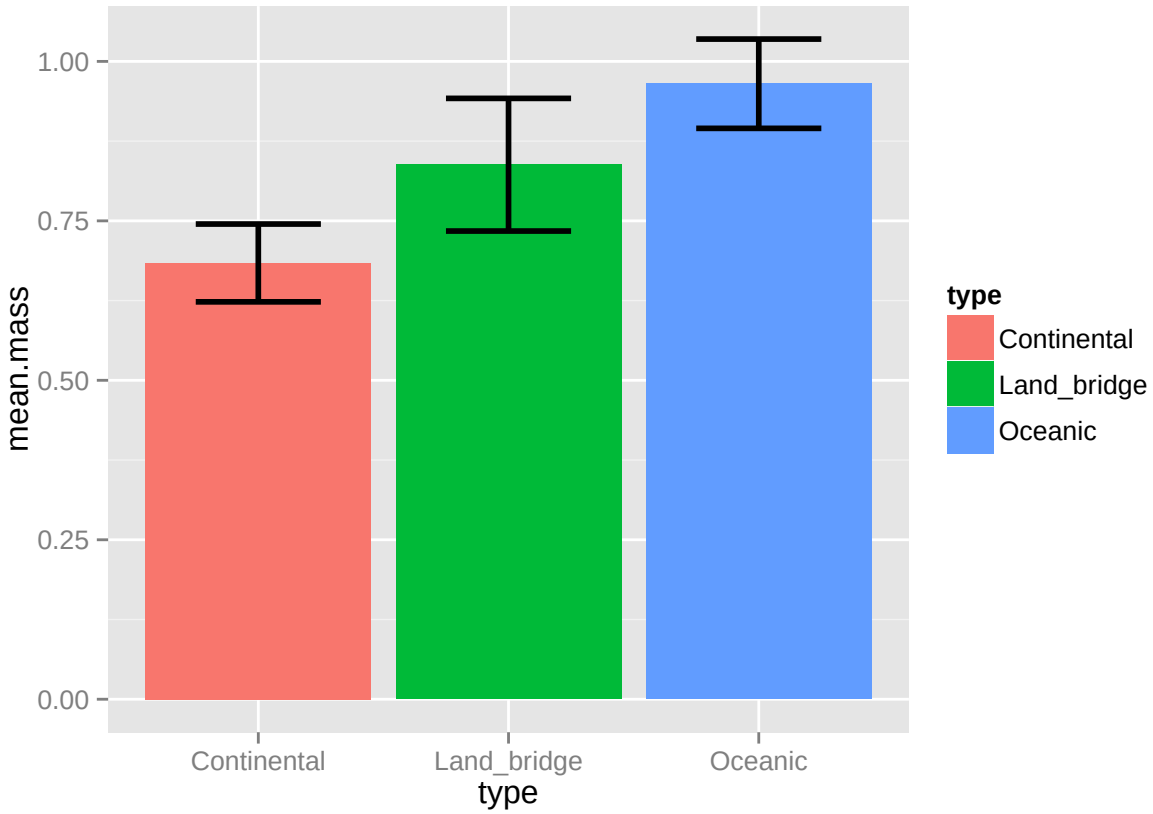
using 'sciplot'

```
barplot.data<-ddply(data,.(type),summarize,mean.mass=round(mean(mass),3),se.mass=round(se(mass),3))
```

 # there is no different between the packages in this manner so it's up to you which one you want to use

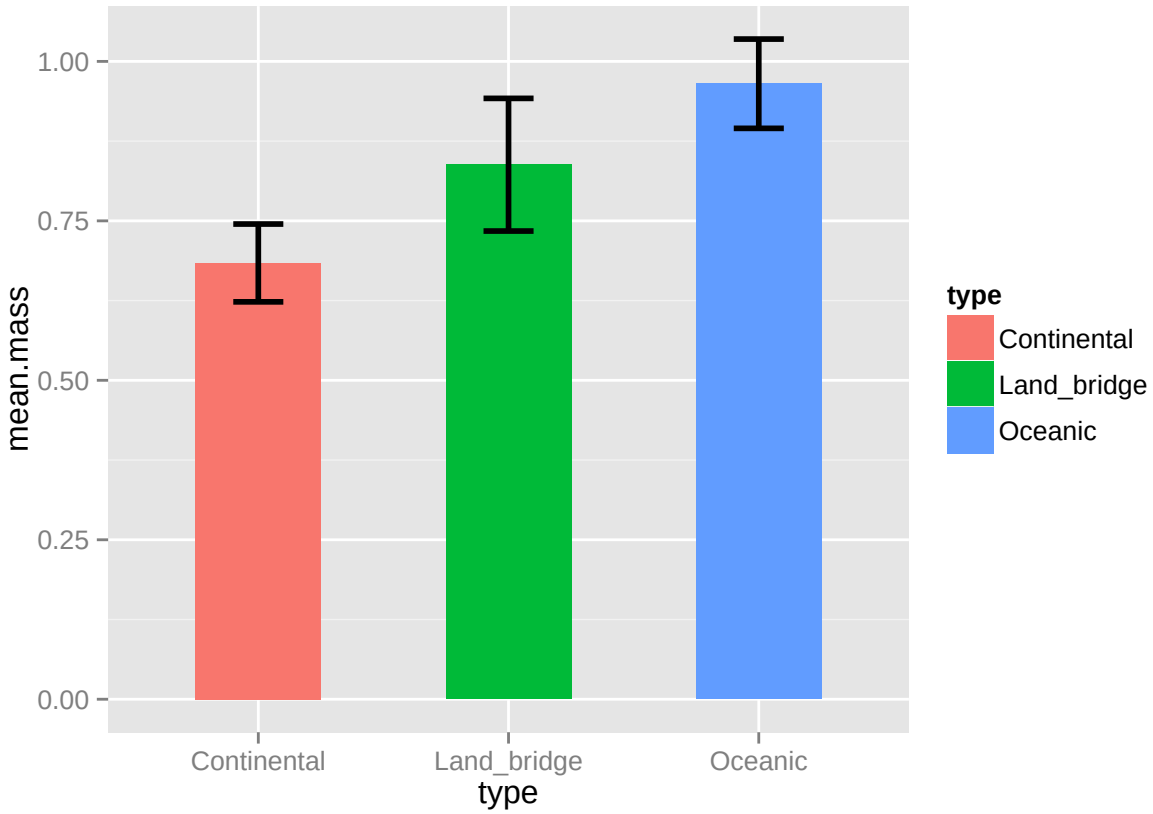
Now lets plot the barplot with the error bars

```
ggplot(barplot.data,aes(type,mean.mass,fill=type))+
  geom_bar(stat='identity')+ # sets the mean to be the statistics for the plot
  geom_errorbar(aes(ymin=mean.mass-se.mass,ymax=mean.mass+se.mass), width=0.5,size=1) # this geom bar c
```



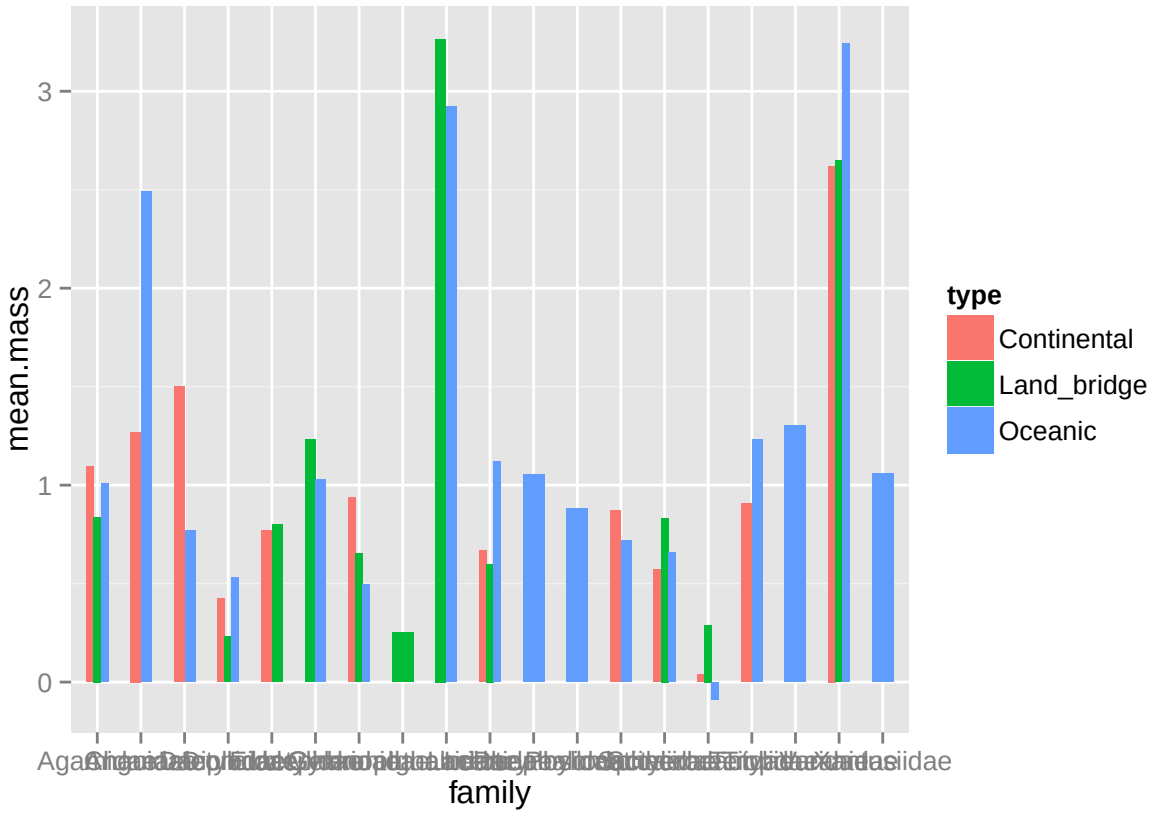
You can change the width of the bars by adding the "width" function to the geom

```
ggplot(barplot.data,aes(type,mean.mass,fill=type))+
  geom_bar(stat='identity',width=.5)+ # sets the mean to be the statistics for the plot
  geom_errorbar(aes(ymin=mean.mass-se.mass,ymax=mean.mass+se.mass), width=0.2,size=1)
```



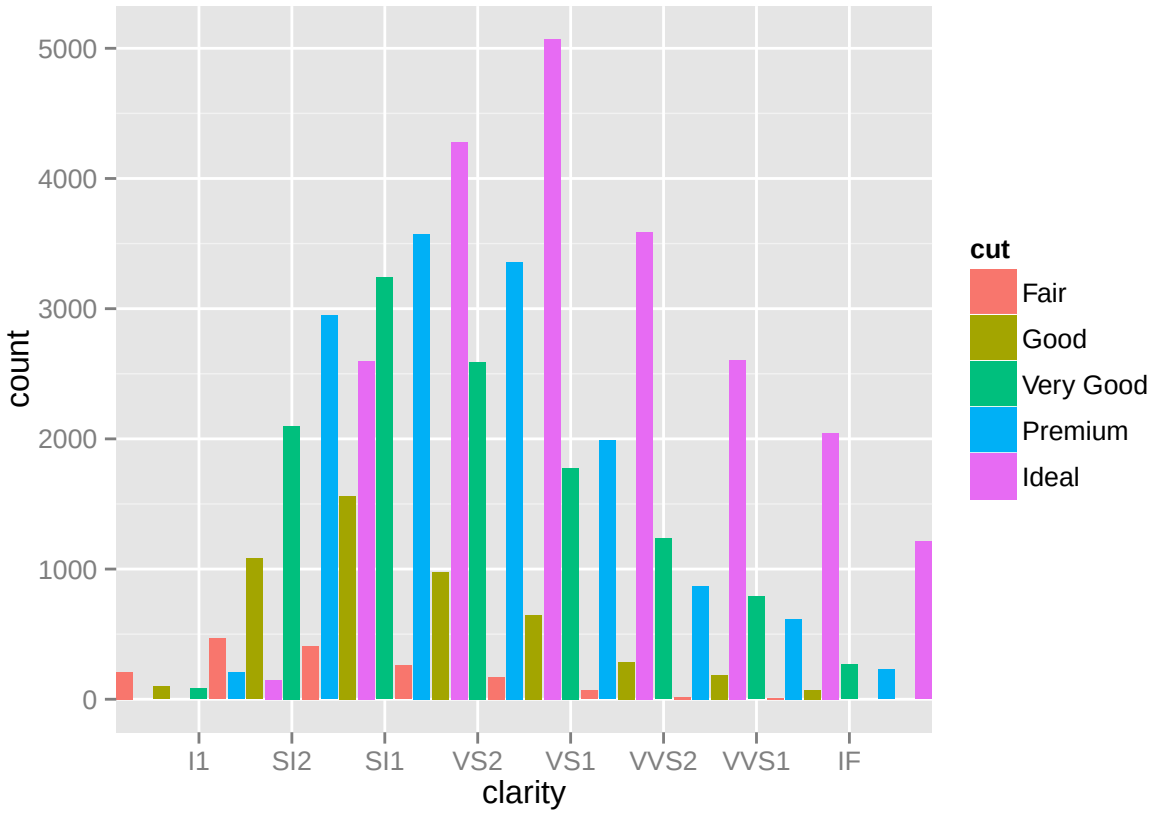
if you want several bars side by side coz you have two categorical variables you want to plot you can use the position="dodge" look up an example in http://docs.ggplot2.org/current/geom_bar.html It is possible to draw error lines on the bars using geom_errorbar first you find the SE of you Y variable

```
barplot.data2<-ddply(data,.(type,family),summarize,mean.mass=round(mean(mass),3),se.mass=round(std.erro
ggplot(barplot.data2, aes(family,mean.mass,fill=type,na.rm=T)) +
  geom_bar(stat = "identity",width=.5,position=position_dodge(width=0.5))
```



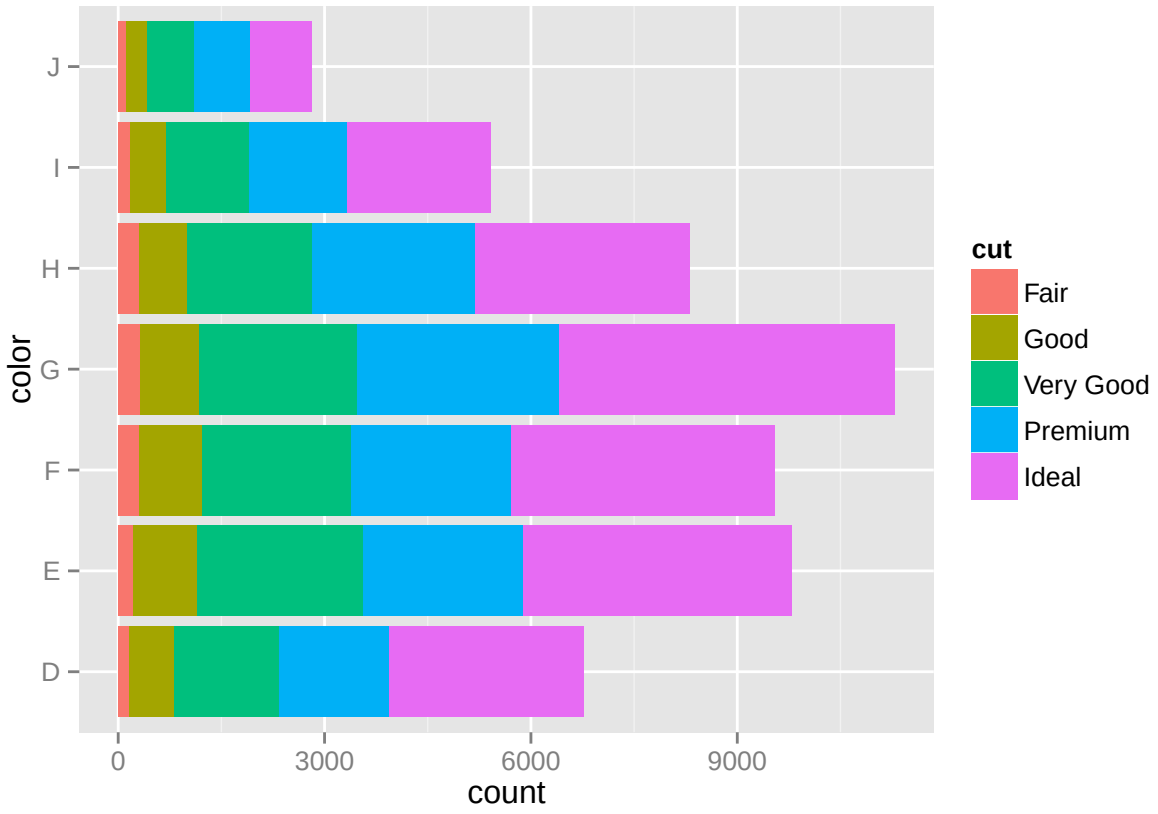
here's an example with some more good looking data from R

```
ggplot(diamonds, aes(clarity, fill=cut)) + geom_bar(position=position_dodge(width=2))
```



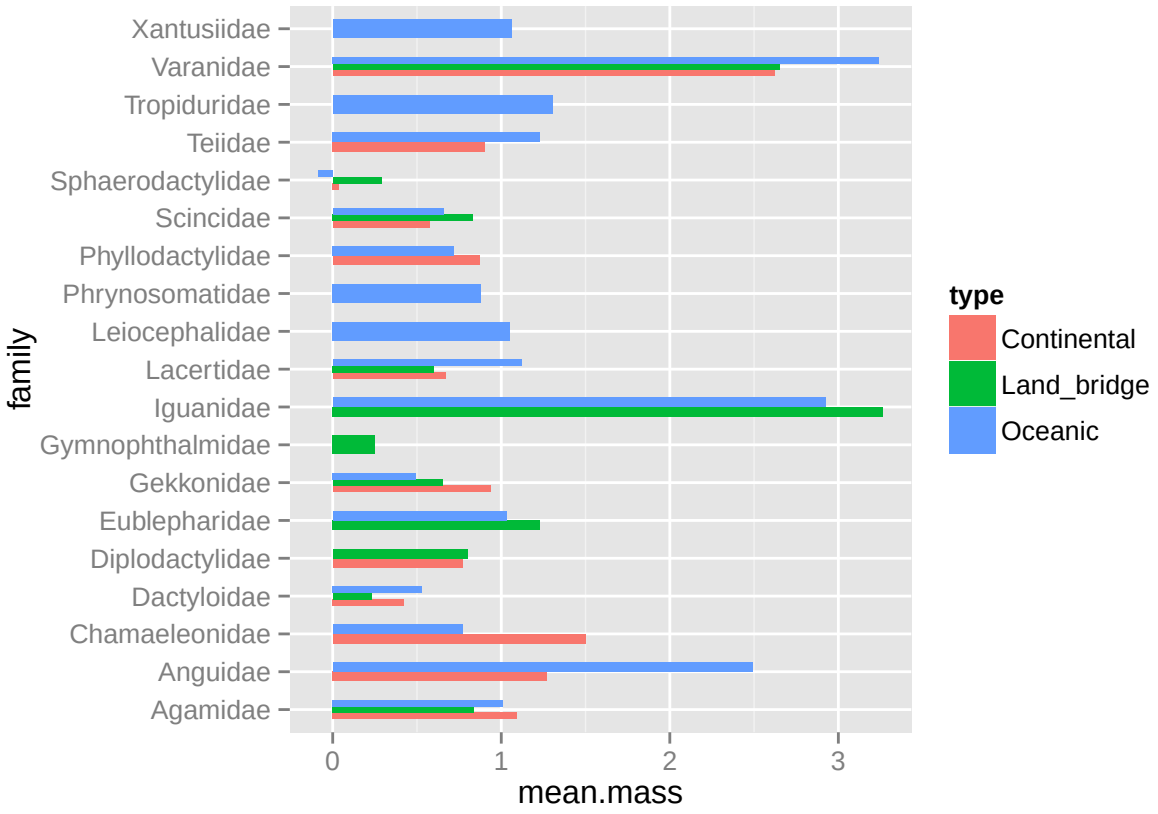
If we want to put the bars on the side we need to add coord_flip

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



and now with our data

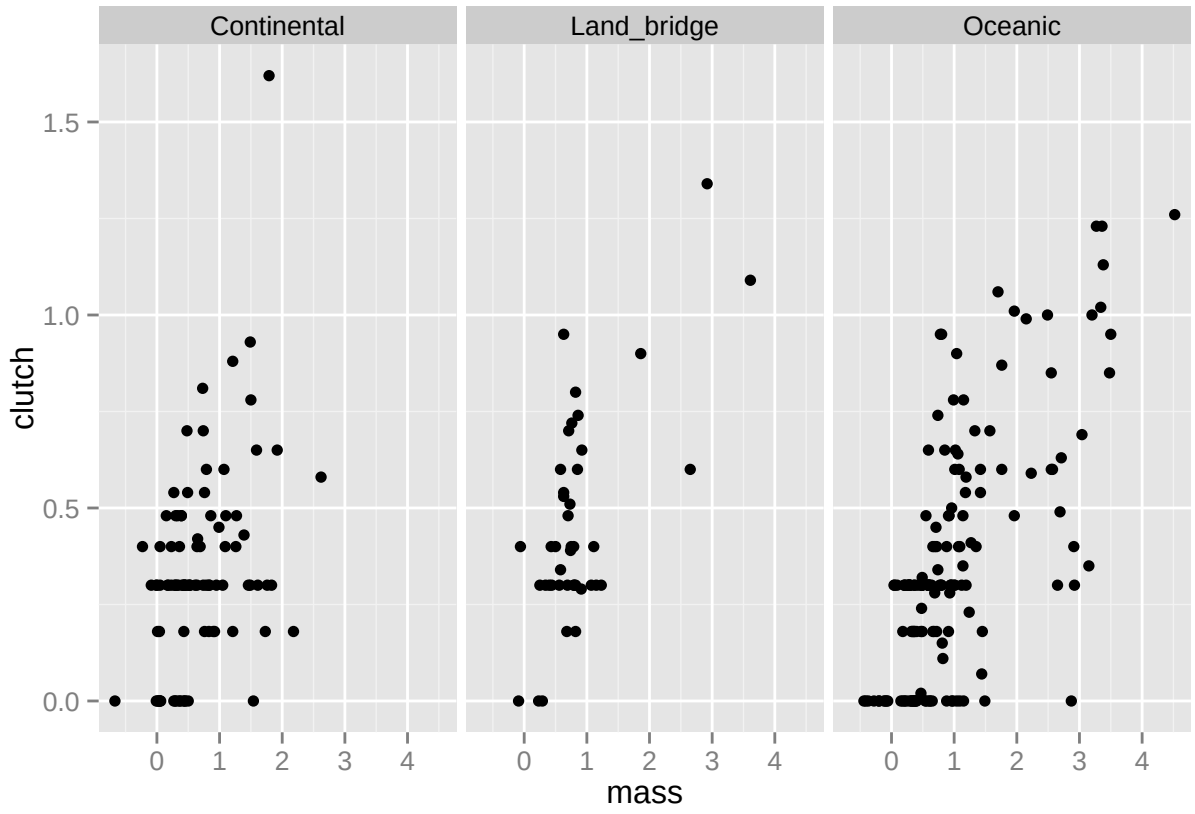
```
ggplot(barplot.data2, aes(family,mean.mass,fill=type,na.rm=T)) +
  geom_bar(stat = "identity",width=.5,position=position_dodge(width=0.5))+coord_flip()
```



Faceting

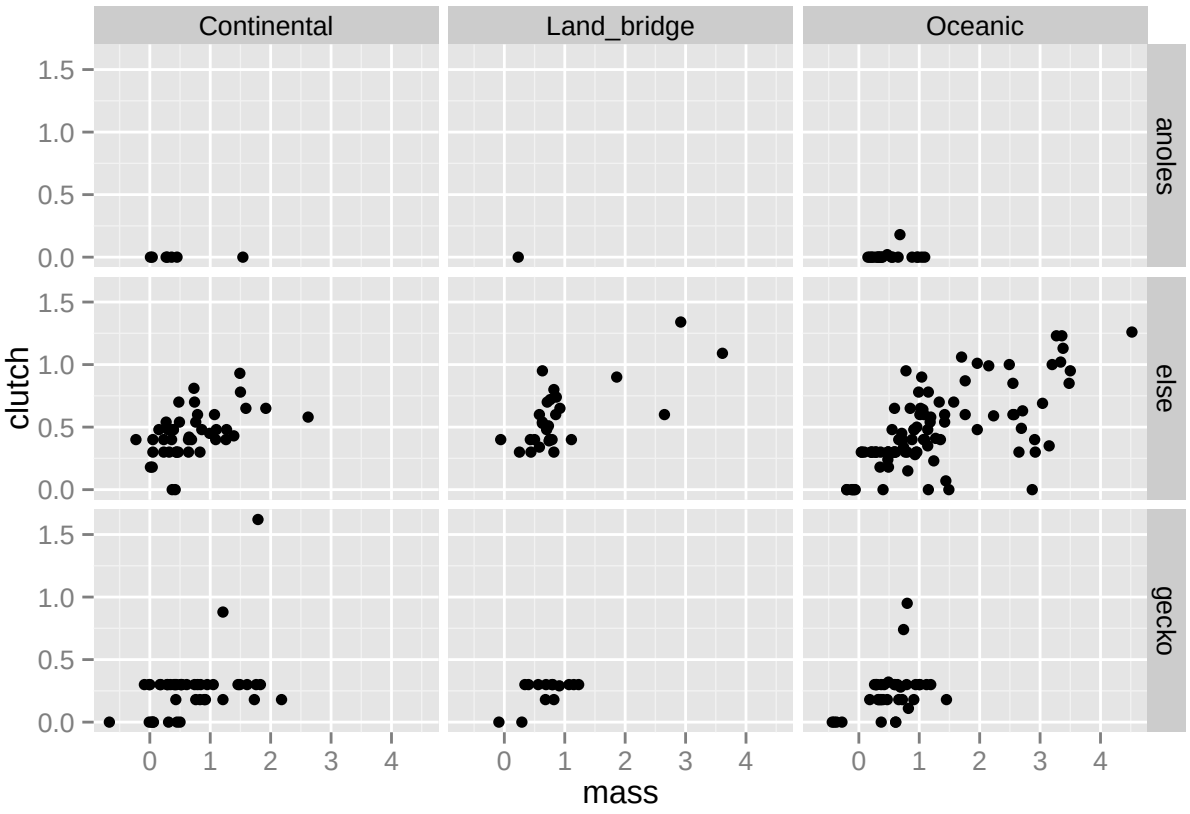
We can lay out multiple plots on a page using the faceting option. This also allows us to split the data into subsets and plot the subsets into diffeent panels.

```
p<-ggplot(data,aes(mass,clutch))+
  geom_point()
p+facet_grid(.~type)
```



Another example

```
p<-ggplot(data,aes(mass,clutch))+
  geom_point()
p+facet_grid(what~type)
```

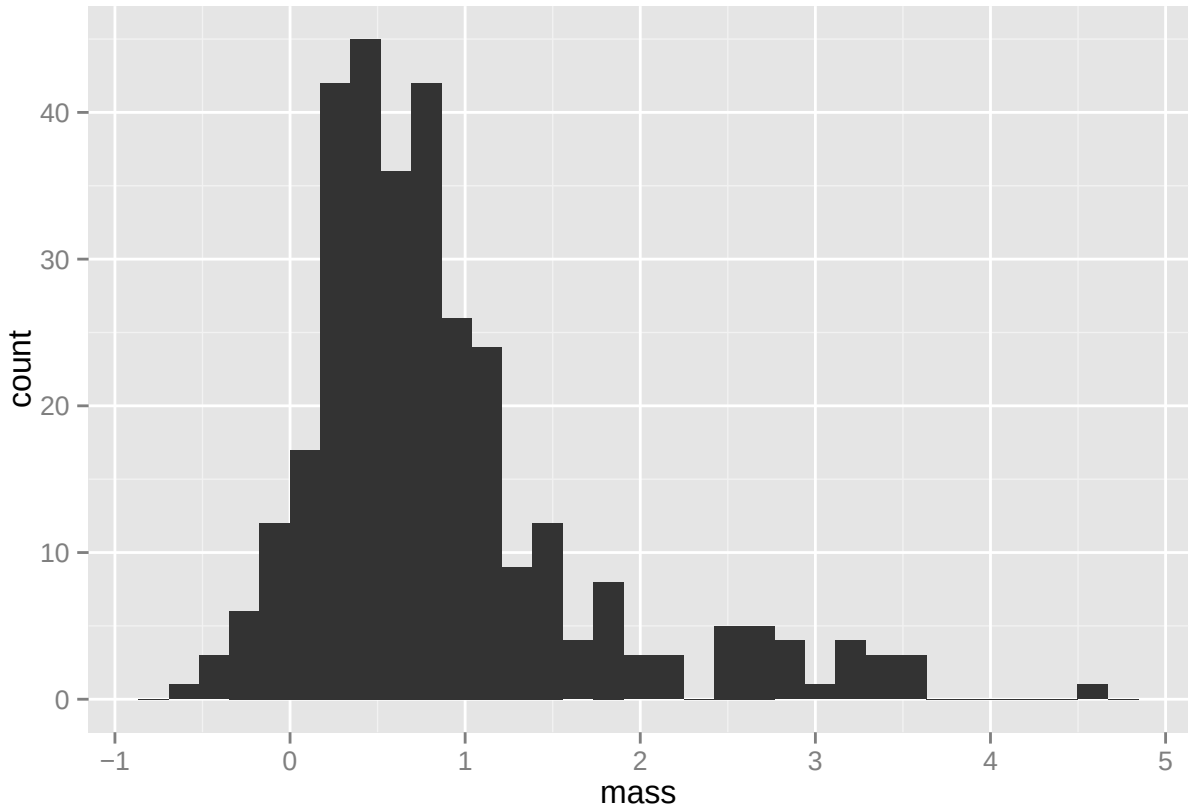


Histograms

It is very easy to do histograms in R. Perhaps one of the main reasons to start working with ggplot this you can do using the geom_histogram

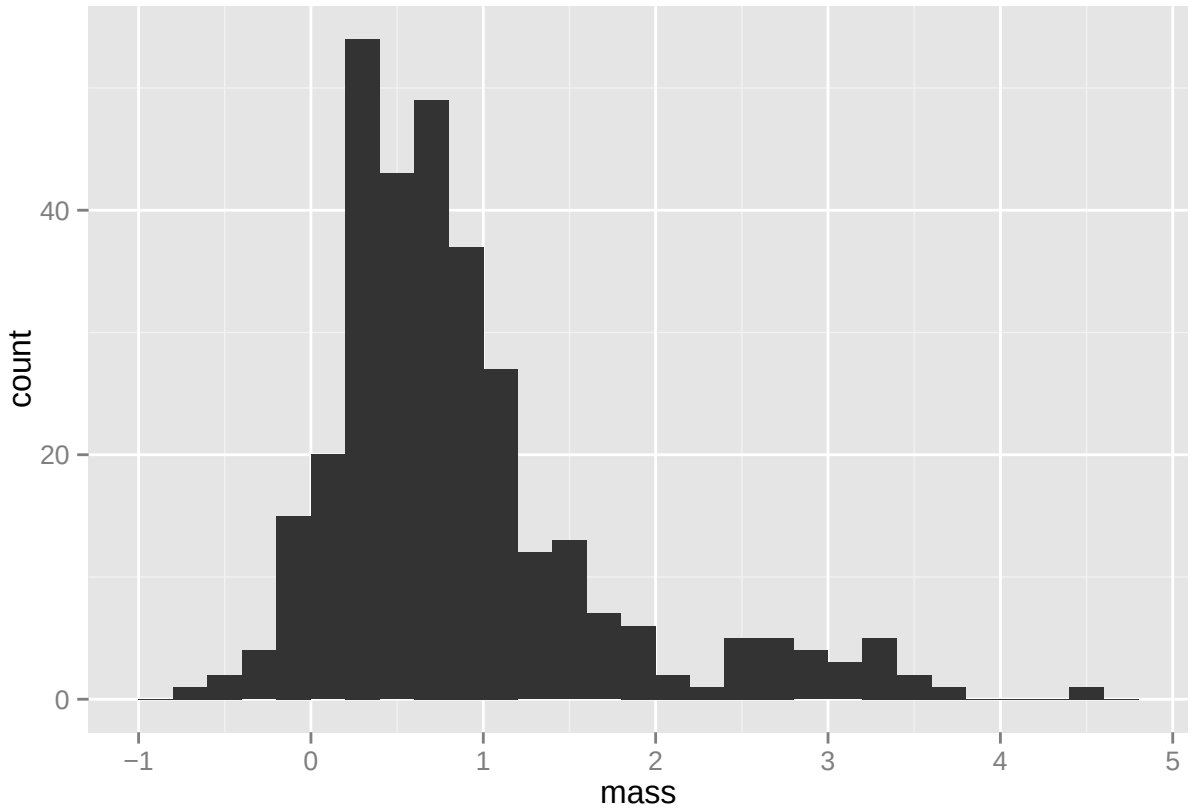
```
ggplot(data, aes(mass)) +
  geom_histogram()
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



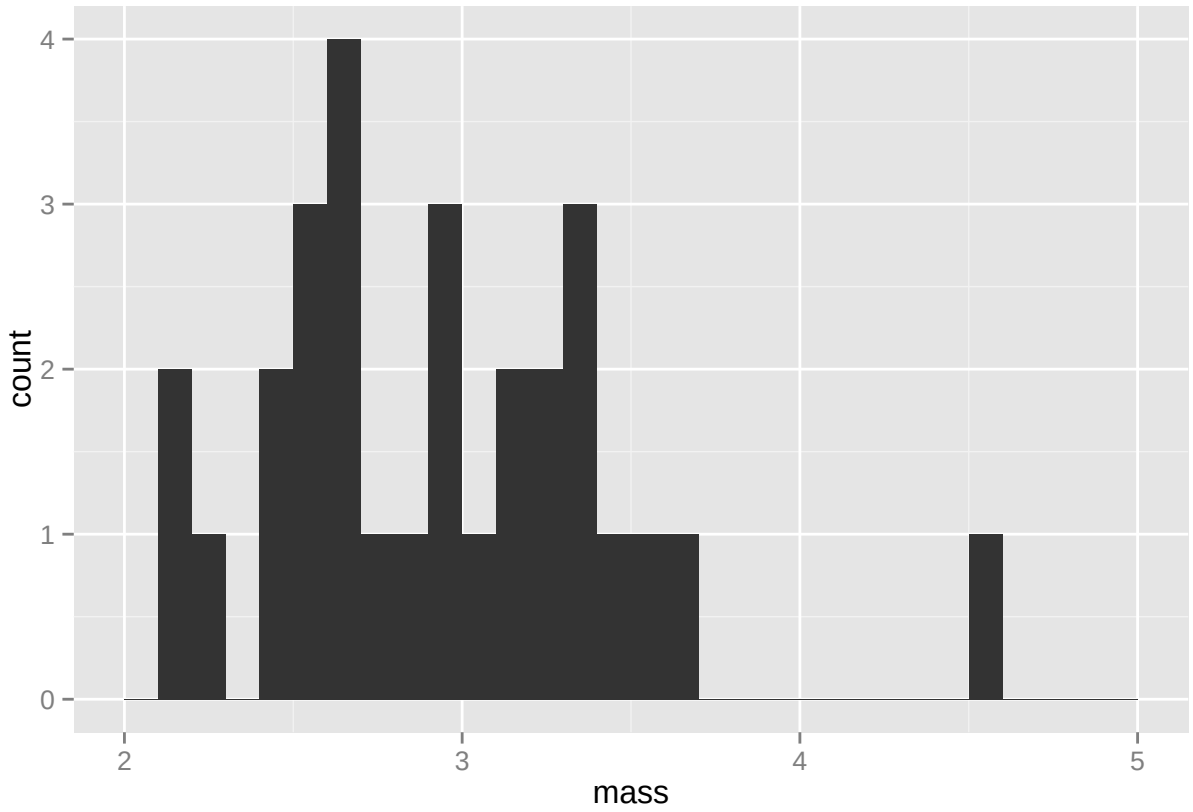
you can change the bin width but here you use the geom_bar

```
ggplot(data, aes(mass)) +
  geom_bar(binwidth = 0.2,position="dodge")
```



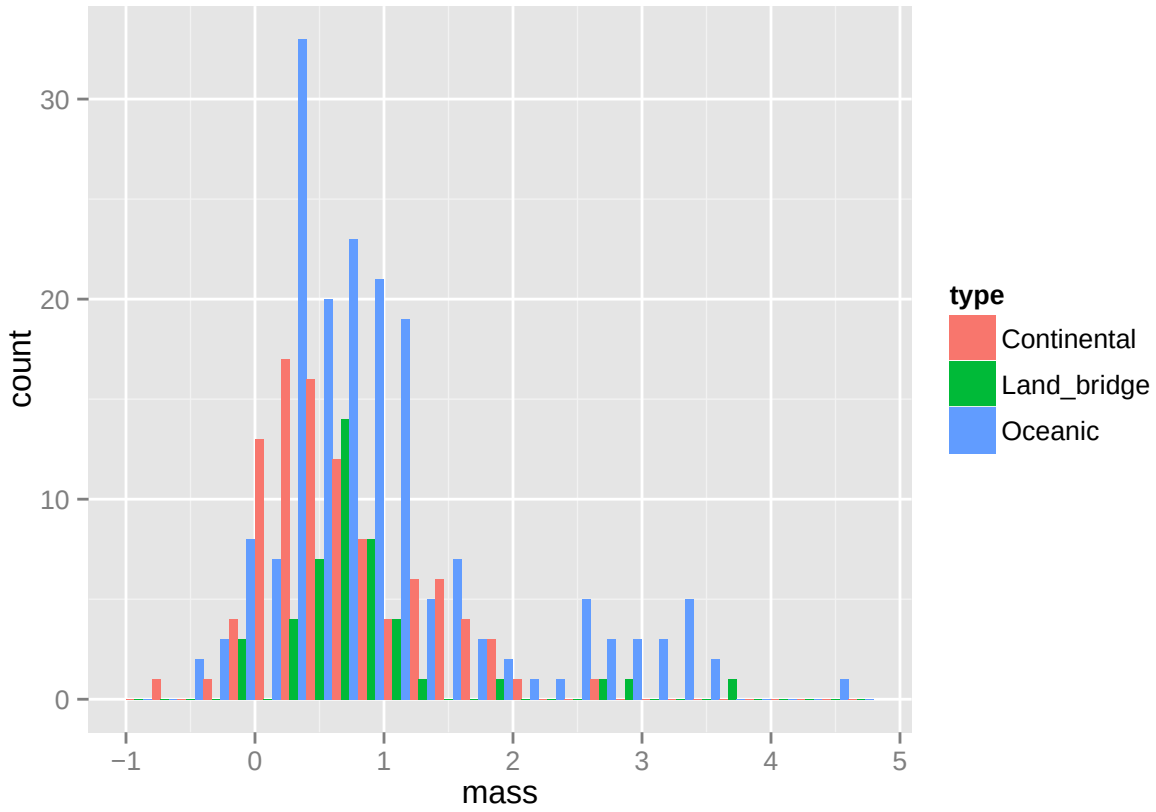
you can also define the limits and the brakes

```
ggplot(data, aes(mass)) +
  geom_bar(breaks=seq(2,5, by=0.1))
```



you can also look at all the categories in your categorical variable by adding fill

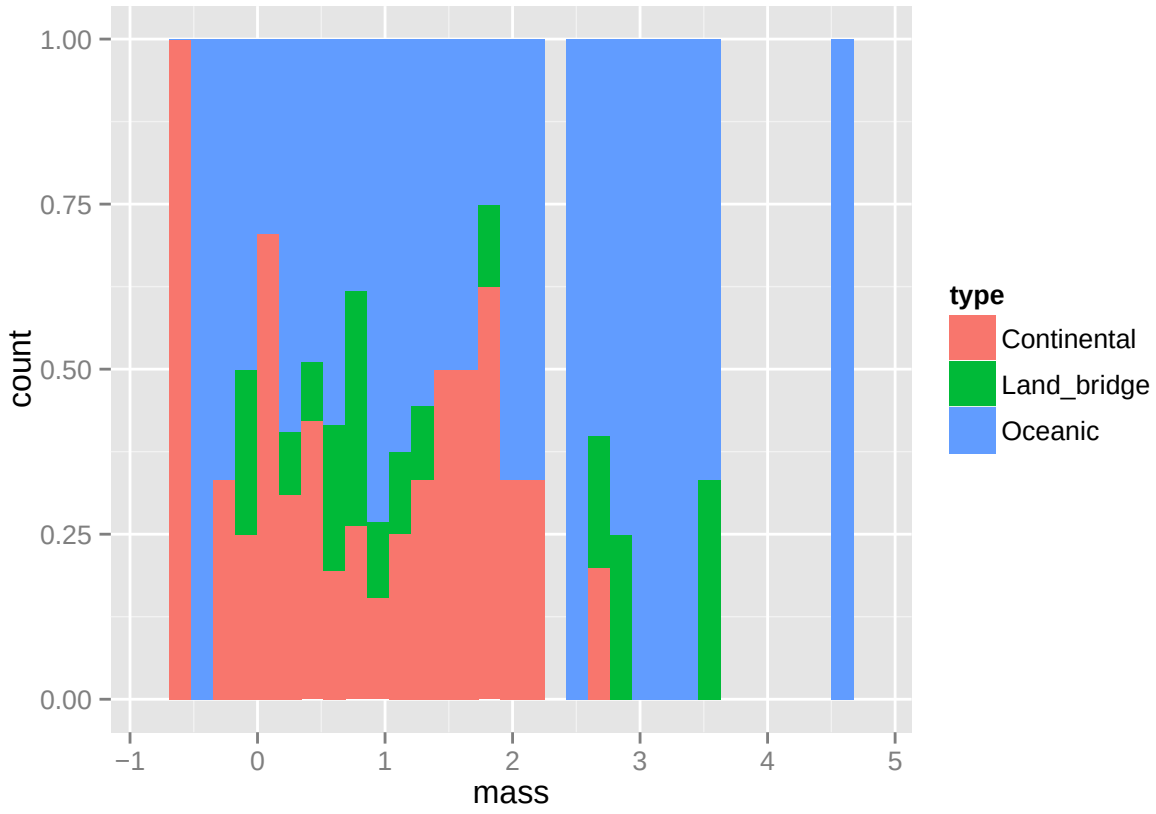
```
ggplot(data, aes(mass,fill=type)) +
  geom_bar(binwidth = 0.2,position="dodge")
```



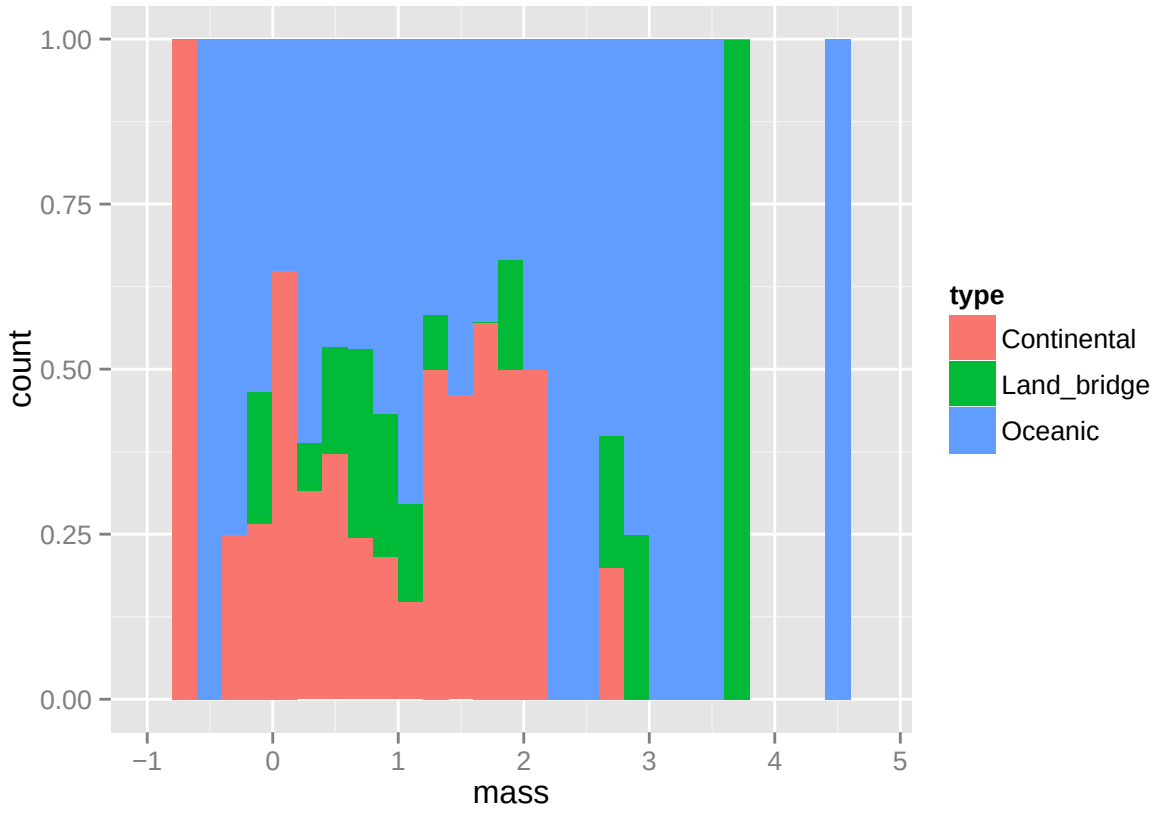
you can also look at the relative percentage that each category takes of a specific value. This you can do by using putting fill instead on dodge in the position function

```
ggplot(data, aes(mass,fill=type)) +
  geom_histogram(position="fill")
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```




```
ggplot(data, aes(mass,fill=type)) +
  geom_bar(binwidth = 0.2,position="fill")
```



for more information on the different possibilities with ggplot check out http://docs.ggplot2.org/0.9.3.1/index.html

Bonus plots

you can do plots for che squer. These kind of plots are called mosaic plots

lets do the chi square test that Shai showed in his lessons. Will see the difference in different lizard groups on different types of islands

first we'll create a matrix with the data. the cast function is enough to make our data ready for chi test but the mosaic plot needs a matrix so we will apply the function as.matrix to the cast function to make it as a function

```
library(reshape)
```

```
##
## Attaching package: 'reshape'
##
## The following objects are masked from 'package:plyr':
##
##     rename, round_any
```

```
chitest<-as.matrix(cast(data,type~what))
```

```
## Using productivity as value column.  Use the value argument to cast to override this choice
## Aggregation requires fun.aggregate: length used as default
```

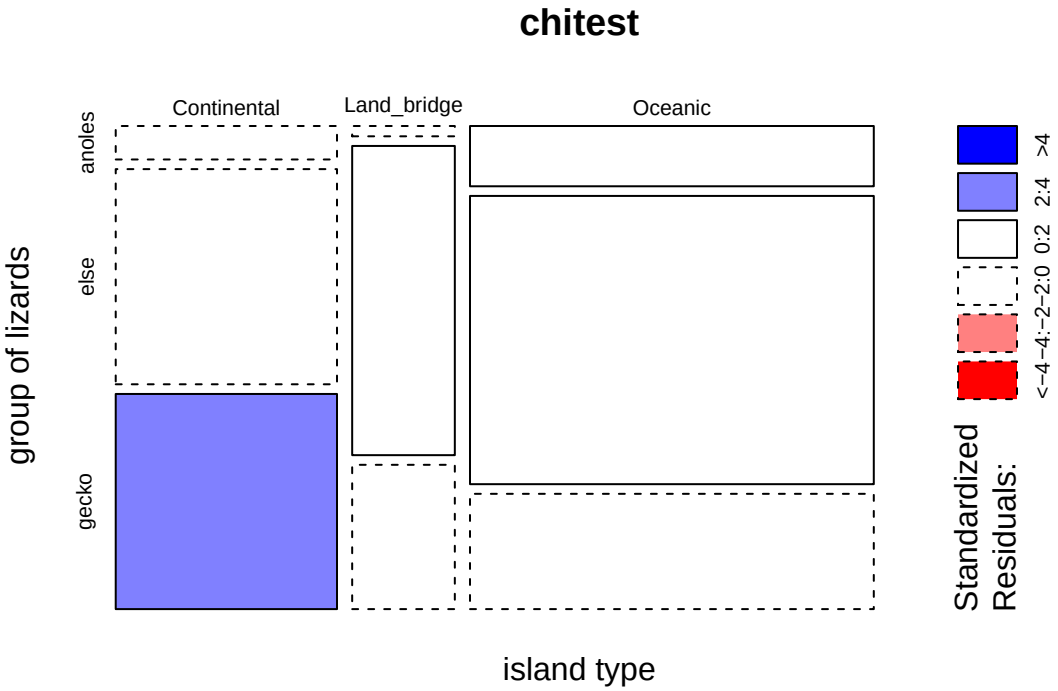
now we'll create a chi square test to see whether there is significance between our groups stor it to be able to use it in the future

```
model<-chisq.test(chitest)
model
```

```
##
##  Pearson's Chi-squared test
##
## data:  chitest
## X-squared = 17.5638, df = 4, p-value = 0.001501
```

now do a mosaic plot on our matrix using the function mosaic plot you can look into the help file to look which variables you can add to manipulate the plot to look as you need it to look

```
mosaicplot(chitest, shade=TRUE, legend=TRUE,xlab="island type",ylab="group of lizards")
```



the chi square test gives you a results that says there is a significant difference between the groups in your data and what is expected by chance. However, it doesn't say which of the groups is different. it's enough that one group is different from what is expected by chance and the whole test will be significant. To know which group effects the significance we need to do a post hoc test (something you know from ANOVA). There is no official test like that for chi square but great statistical minds say that you can look at the residuals of the observed from the expected.

If its larger than 2 it's significantly more than expected by chance, if it's smaller than -2 it's significantly less than expected by chance. In our example you can see that there is more geckos on continental islands than is expected by chance (colored in blue).

This is where we end our lesson on ggplot2 and some other plots. You are welcome to check out ggplot2 website for more information on how to specify your plots http://docs.ggplot2.org/current/. Also google is a good friend in this case