Information design and data visualization
Week 3, Lecture 05

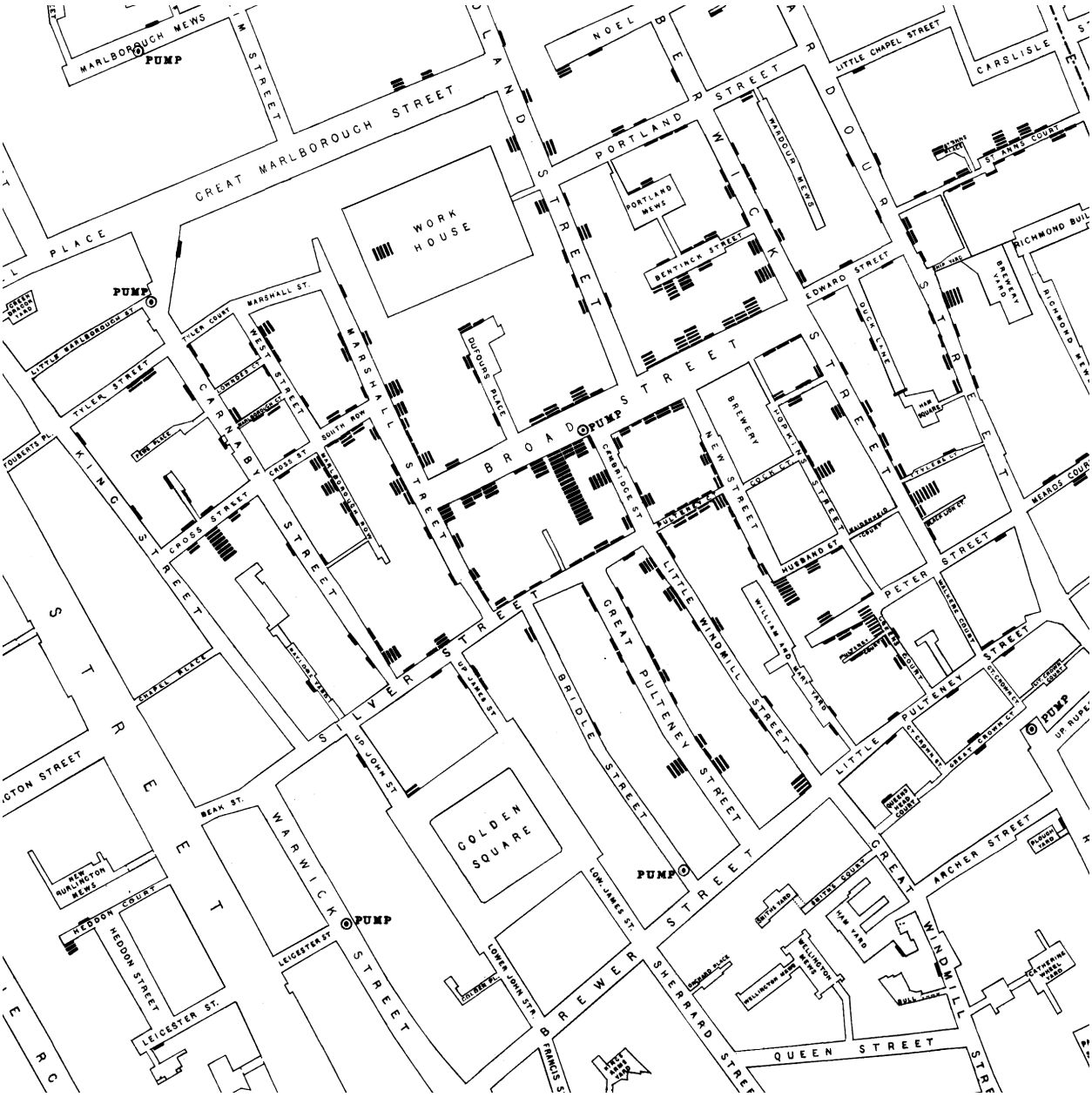
Richard E.W. Berl

Spring 2019

Information design

John Snow did know something... about cholera

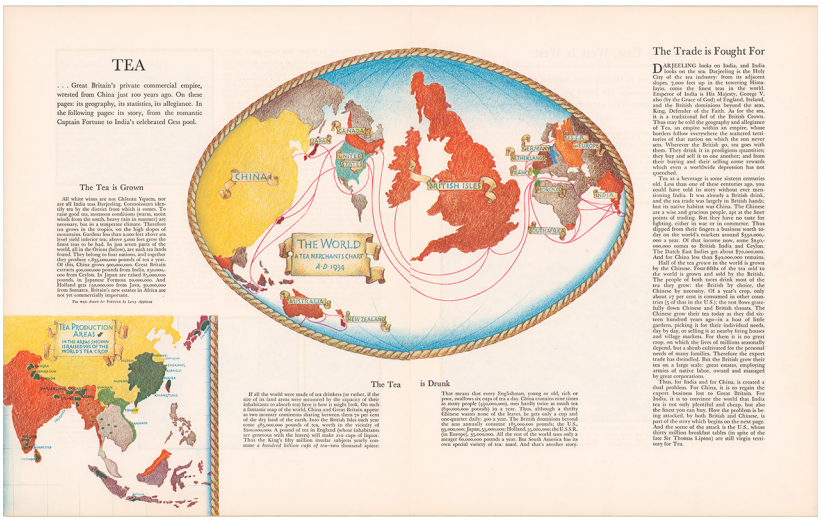
September 1854 cholera outbreak, Soho district, London



Source: John Snow, courtesy of John Mackenzie, University of Delaware

Bad figures are bad...

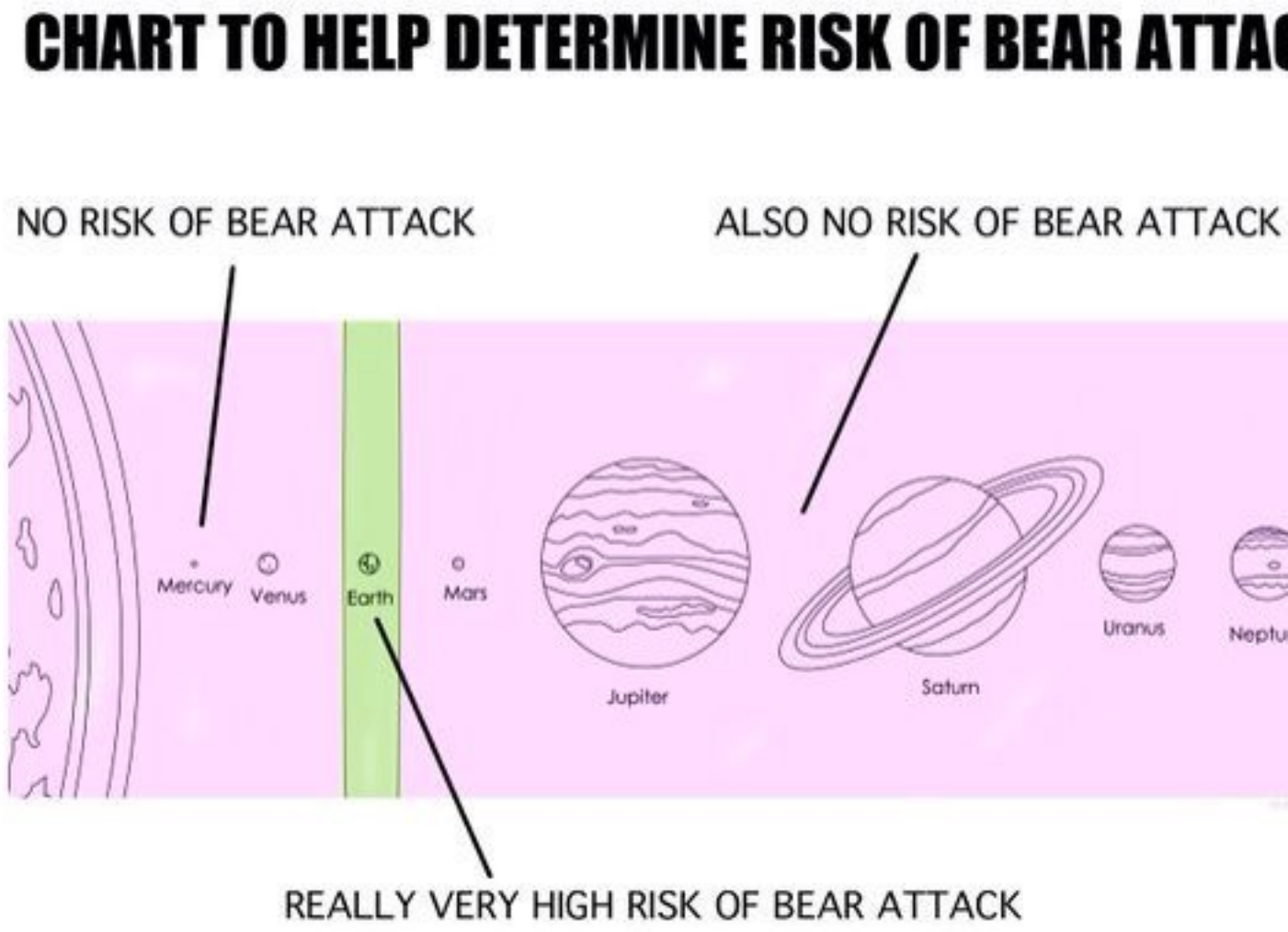
Because of bad taste



Source: "The Tea is Drunk," Fortune Magazine, June 1934, courtesy of Cornell University Library Digital Collections

- "Data-to-ink" ratio
- Aesthetics

Because of bad data



Source: @TerribleMaps

- Data quality
- Incorrect analyses
- Misleading or deceptive presentation

Because of bad perception

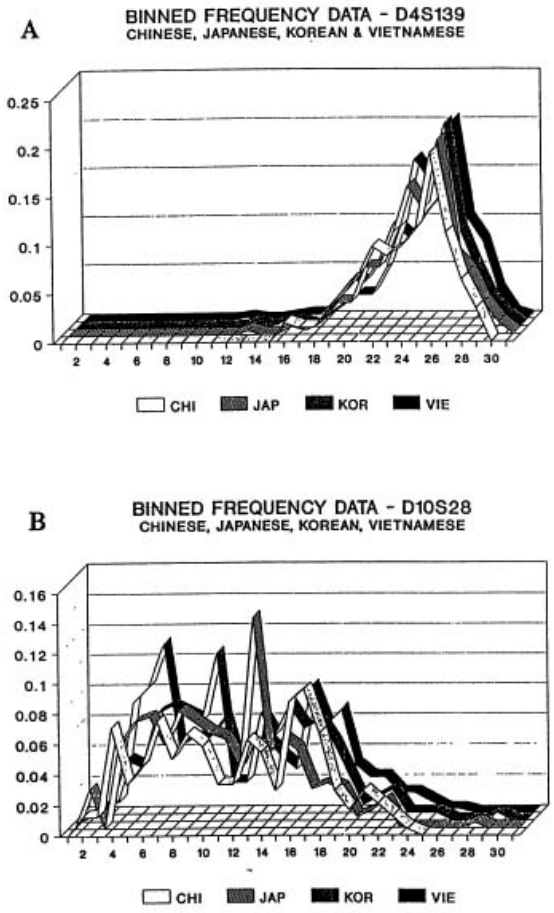
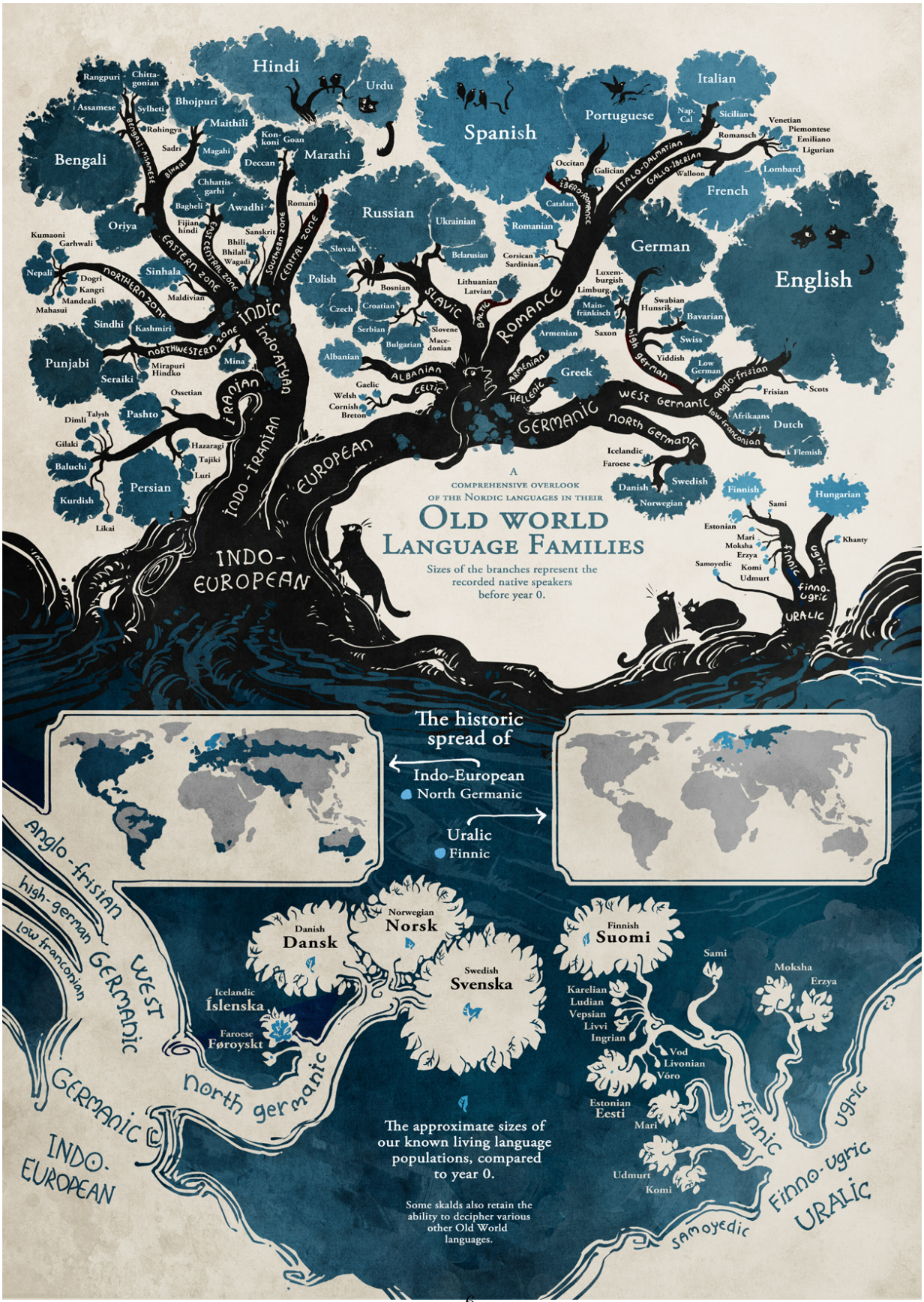


FIG. 4. *Fixed bin distribution (histogram) for two loci and four Asian subpopulations (used with permission from John Hartmann): the boundaries of the* 30 *bins (vertical axis) are determined by the FBI; these bins are not of equal length. Sample sizes (numbers of individuals) for Chinese, Japanese, Korean and Vietnamese are* 103, 125, 93 *and* 215 *for* D4S139 *and* 120, 137, 100 *and* 193 *for* D10S28. *The horizontal axis is the bin number; bins are not of equal length.*

Source: Roeder, K. (1994). DNA fingerprinting: A review of the controversy. Statistical Science 9(2): 222-247, courtesy of Karl Broman, University of Wisconsin-Madison

- Ease of interpretation

Tell a story



Source: Minna Sundberg, Stand Still. Stay Silent #196

base

Useful for quick-and-dirty exploratory visualizations because you don't have to load any additional packages.

lattice

Many examples of potential uses available online; especially good at showing relationships for multiple variables or across multiple plots. However, my opinion is that almost anything `lattice` can do can be done better by `ggplot2` with cleaner code.

ggplot2

```
install.packages("tidyverse")
# OR
install.packages("ggplot2")
```

```
library(ggplot2)
```

A plot made using `ggplot()` has, at minimum, three elements:

- data set
- aesthetics
- geometry

Let's get some data to work with.

Go to the Supplementary Materials for Stoddard et al. (2018) at: https://science.sciencemag.org/content/suppl/2017/06/21/356.6344.1249.DC1

Click "Data S1" to download the data, in Excel format, and put it in your `/data` folder.

```
library(readxl)
```

```
egg = read_xlsx(path="./data/aaj1945_DataS1_Egg_shape_by_species_v2.xlsx",
                      sheet=1)
egg = as.data.frame(egg)
```

```
head(egg)
```

```
##             Order       Family        MVZDatabase             Species
## 1 ACCIPITRIFORMES Accipitridae   Accipiter badius   Accipiter badius
## 2 ACCIPITRIFORMES Accipitridae Accipiter cooperii Accipiter cooperii
## 3 ACCIPITRIFORMES Accipitridae Accipiter gentilis Accipiter gentilis
## 4 ACCIPITRIFORMES Accipitridae    Accipiter nisus    Accipiter nisus
## 5 ACCIPITRIFORMES Accipitridae Accipiter striatus Accipiter striatus
## 6 ACCIPITRIFORMES Accipitridae  Aegypius monachus  Aegypius monachus
##   Asymmetry Ellipticity AvgLength (cm) Number of images Number of eggs
## 1    0.1378   0.3435262         3.8642                1              2
## 2    0.0937   0.2715367         4.9008               27            103
## 3    0.1114   0.3186002         5.9863                7             18
## 4    0.0808   0.2390652         4.0355               13             61
## 5    0.0749   0.2542802         3.8700               15             57
## 6    0.0700   0.3476181         8.9076                1              1
```

```
str(egg)
```

```
## 'data.frame':    1402 obs. of  9 variables:
##  $ Order           : chr  "ACCIPITRIFORMES" "ACCIPITRIFORMES" "ACCIPITRIFORMES" "ACCIPITRIFORMES" ..
##  $ Family          : chr  "Accipitridae" "Accipitridae" "Accipitridae" "Accipitridae" ...
##  $ MVZDatabase     : chr  "Accipiter badius" "Accipiter cooperii" "Accipiter gentilis" "Accipiter nis
##  $ Species         : chr  "Accipiter badius" "Accipiter cooperii" "Accipiter gentilis" "Accipiter nis
##  $ Asymmetry       : num  0.1378 0.0937 0.1114 0.0808 0.0749 ...
##  $ Ellipticity     : num  0.344 0.272 0.319 0.239 0.254 ...
##  $ AvgLength (cm)  : num  3.86 4.9 5.99 4.04 3.87 ...
##  $ Number of images: num  1 27 7 13 15 1 191 1 7 2 ...
##  $ Number of eggs  : num  2 103 18 61 57 1 391 2 17 4 ...
```

Those spaces in the column names may cause problems, so let's fix them.

```
colnames(egg)
```

```
## [1] "Order"            "Family"           "MVZDatabase"
## [4] "Species"          "Asymmetry"        "Ellipticity"
## [7] "AvgLength (cm)"   "Number of images" "Number of eggs"
```

```
colnames(egg)[7:9] = c("AvgLength","NumberOfImages","NumberOfEggs")
```

And check out the help page for the `ggplot()` function before we use it.

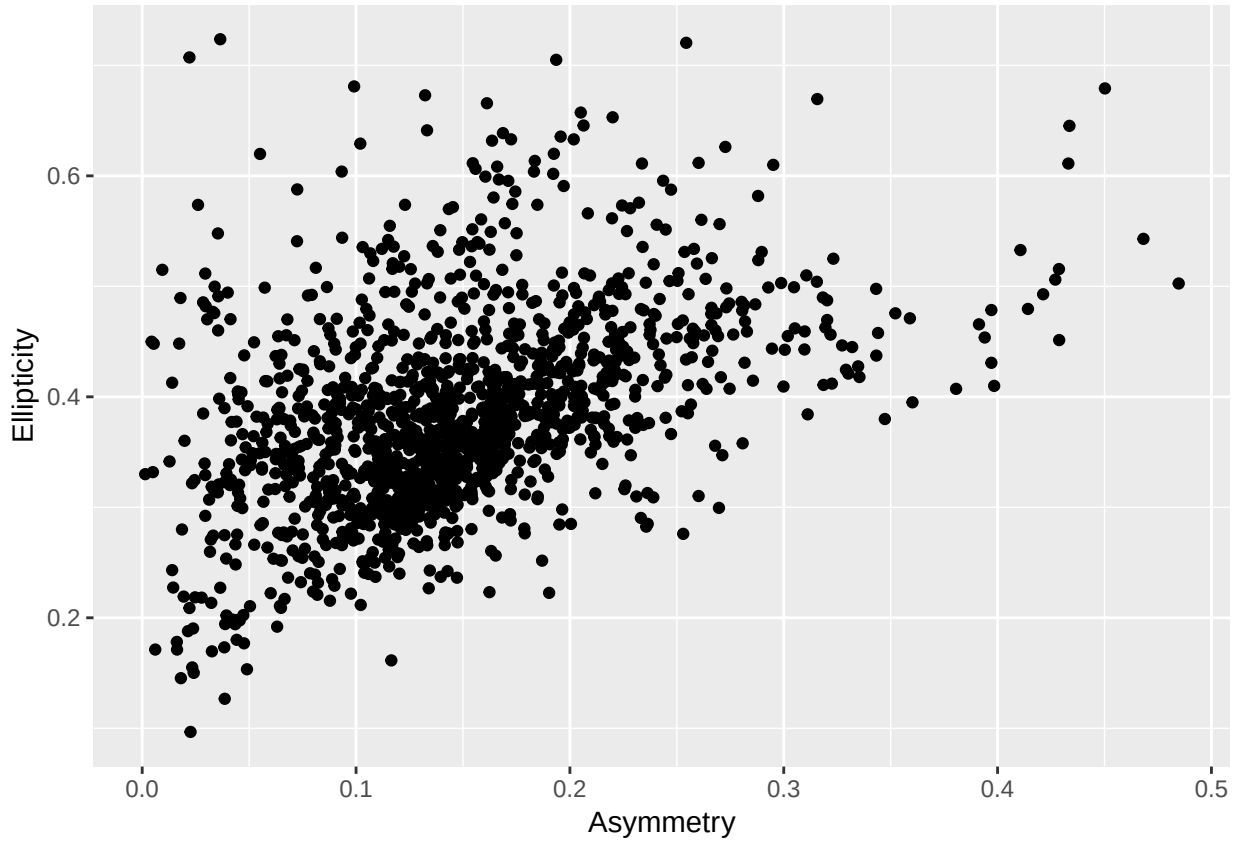
```
?ggplot
```

Not very helpful... A better resource is the tidyverse page for `ggplot2`: https://ggplot2.tidyverse.org/

Now we can make our first plot with the three elements needed: data, aesthetic mappings, and a geometry.

```
ggplot(data=egg, mapping=aes(x=Asymmetry, y=Ellipticity)) +
  geom_point()
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



The function warns us that we have some missing values. Let's find them and remove them.

```
egg[is.na(egg$Asymmetry),]
```

```
##      Order Family MVZDatabase Species Asymmetry Ellipticity AvgLength
## 1401  <NA>   <NA>        <NA>    <NA>        NA          NA        NA
## 1402  <NA>   <NA>        <NA>    <NA>        NA          NA        NA
##      NumberOfImages NumberOfEggs
## 1401             NA           NA
## 1402             NA           NA
```

```
egg[is.na(egg$Ellipticity),]
```

```
##      Order Family MVZDatabase Species Asymmetry Ellipticity AvgLength
## 1401  <NA>   <NA>        <NA>    <NA>        NA          NA        NA
## 1402  <NA>   <NA>        <NA>    <NA>        NA          NA        NA
##      NumberOfImages NumberOfEggs
## 1401             NA           NA
## 1402             NA           NA
```

```
tail(egg)
```

```
##              Order     Family               MVZDatabase
## 1397  TINAMIFORMES  Tinamidae      Rhynchotus rufescens
## 1398 TROGONIFORMES Trogonidae       Euptilotis neoxenus
## 1399 TROGONIFORMES Trogonidae Harpactes erythrocephalus
## 1400 TROGONIFORMES Trogonidae            Trogon elegans
## 1401          <NA>       <NA>                      <NA>
## 1402          <NA>       <NA>                      <NA>
```

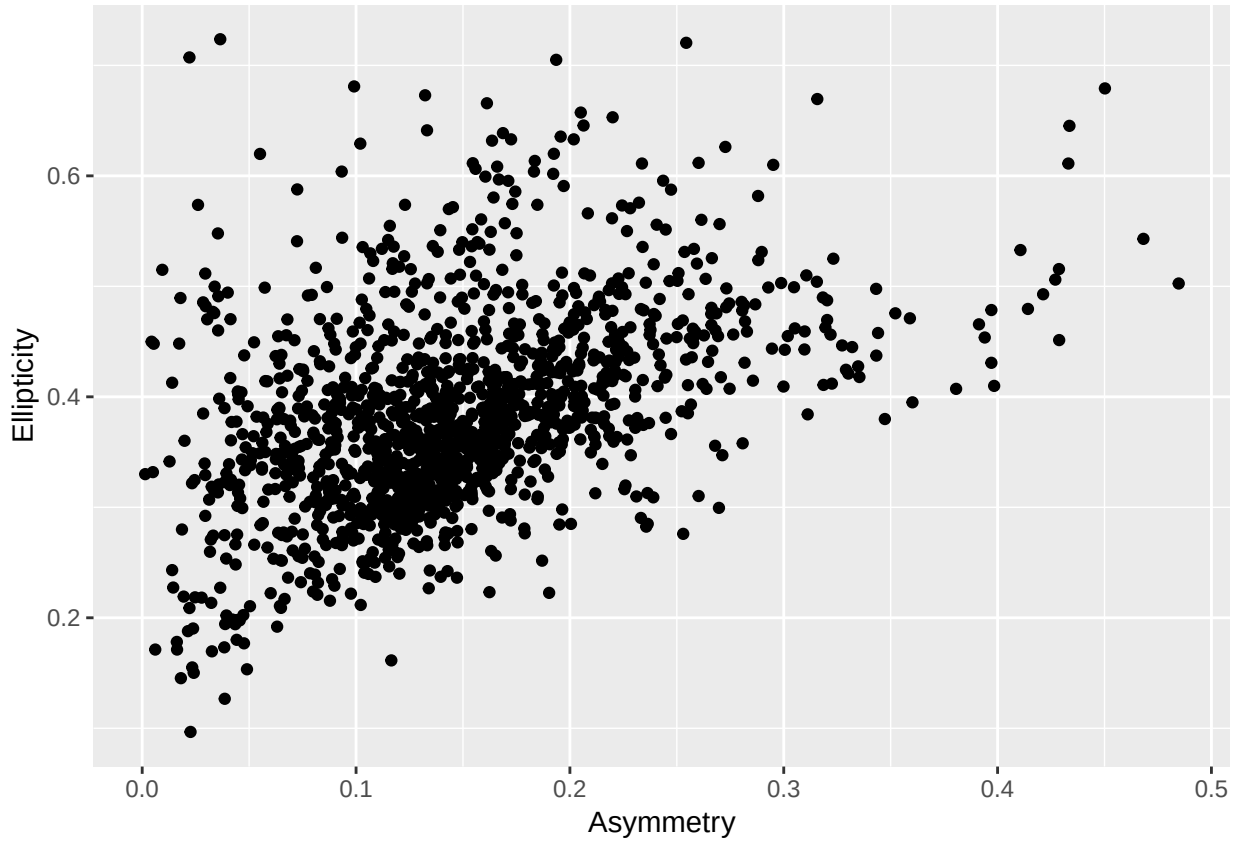
```
##                         Species Asymmetry Ellipticity AvgLength
## 1397      Rhynchotus rufescens    0.0484   0.3542979    5.5407
## 1398       Euptilotis neoxenus    0.1472   0.2362926    3.0608
## 1399 Harpactes erythrocephalus    0.0490   0.1533892    2.7930
## 1400            Trogon elegans    0.2357   0.2824952    2.8055
## 1401                      <NA>        NA          NA        NA
## 1402                      <NA>        NA          NA        NA
##      NumberOfImages NumberOfEggs
## 1397              1            1
## 1398              1            1
## 1399              1            3
## 1400              1            4
## 1401             NA           NA
## 1402             NA           NA
```

Sometimes Excel spreadsheets can have empty rows at the bottom that get interpreted as data.

```
egg = egg[-c(1401,1402),]
tail(egg)
```

```
##              Order     Family               MVZDatabase
## 1395  TINAMIFORMES  Tinamidae          Eudromia elegans
## 1396  TINAMIFORMES  Tinamidae    Nothoprocta perdicaria
## 1397  TINAMIFORMES  Tinamidae      Rhynchotus rufescens
## 1398 TROGONIFORMES Trogonidae       Euptilotis neoxenus
## 1399 TROGONIFORMES Trogonidae Harpactes erythrocephalus
## 1400 TROGONIFORMES Trogonidae            Trogon elegans
##                         Species Asymmetry Ellipticity AvgLength
## 1395          Eudromia elegans    0.0418   0.3772397    5.2926
## 1396    Nothoprocta perdicaria    0.0535   0.3820363    4.3292
## 1397      Rhynchotus rufescens    0.0484   0.3542979    5.5407
## 1398       Euptilotis neoxenus    0.1472   0.2362926    3.0608
## 1399 Harpactes erythrocephalus    0.0490   0.1533892    2.7930
## 1400            Trogon elegans    0.2357   0.2824952    2.8055
##      NumberOfImages NumberOfEggs
## 1395              3           11
## 1396              1            3
## 1397              1            1
## 1398              1            1
## 1399              1            3
## 1400              1            4
```

```
ggplot(data=egg, mapping=aes(x=Asymmetry, y=Ellipticity)) +
  geom_point()
```

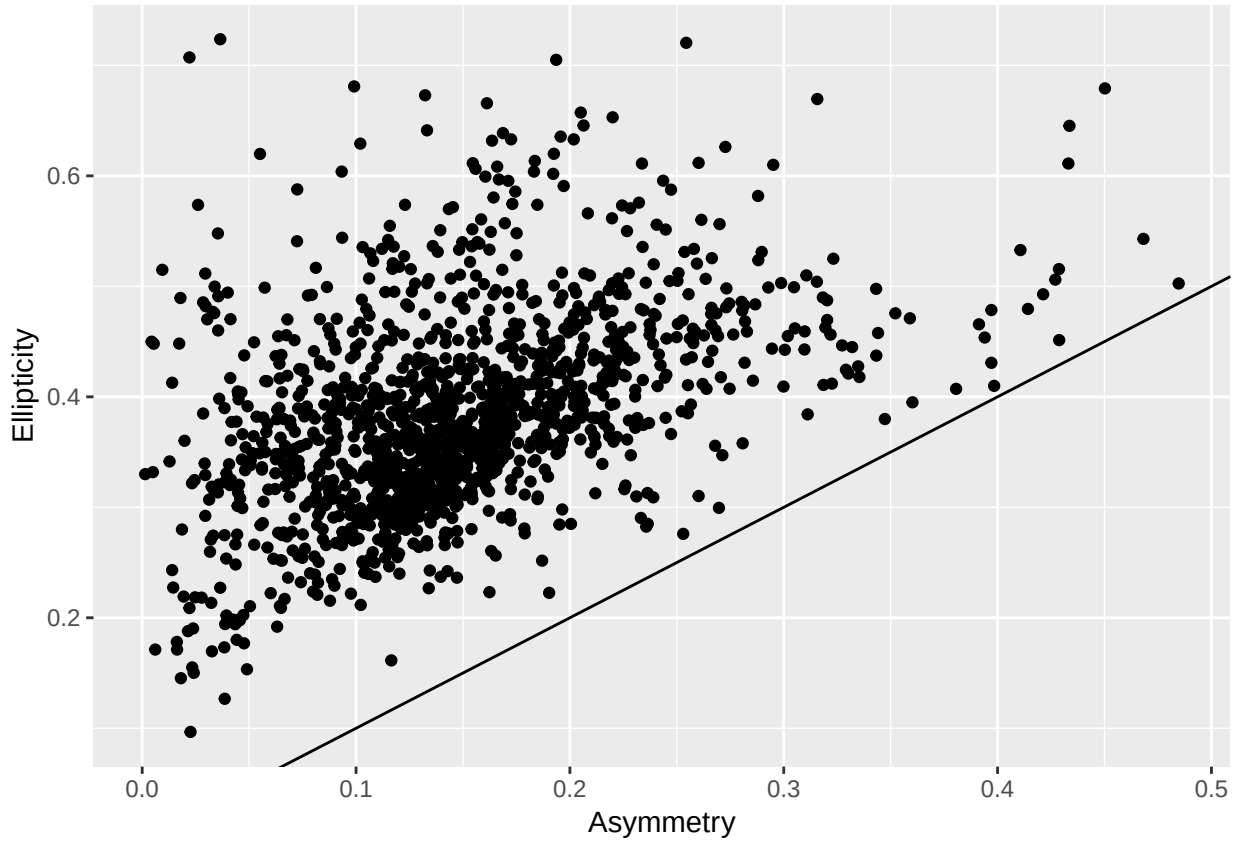


The `ggplot2` reference page on the tidyverse website has a list of `geom`s and other layers you can apply: https://ggplot2.tidyverse.org/reference/

One of the first ones is `geom_abline()`, which is one way to add a reference line to our plot.

```
?geom_abline
```

```
ggplot(data=egg, mapping=aes(x=Asymmetry, y=Ellipticity)) +
  geom_point() +
  geom_abline(slope=1, intercept=0)
```



We see that every point is above the 1:1 line; that is, an egg's ellipticity is always greater than its asymmetry.

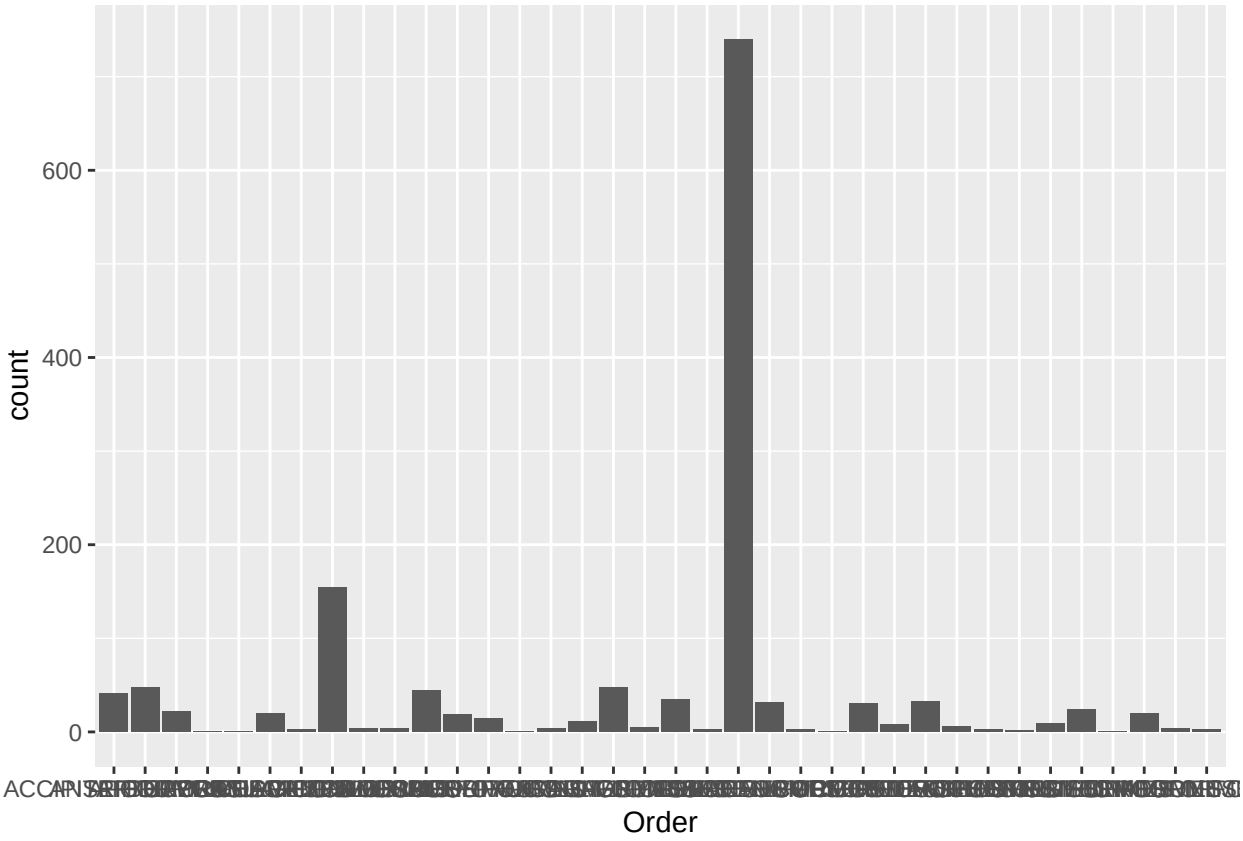
Let's try a bar chart now. We need a categorical X value.

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar()
```

```
## Error: stat_count() must not be used with a y aesthetic.
```

Well, it seems like it wants us to remove the Y variable, so let's try that.

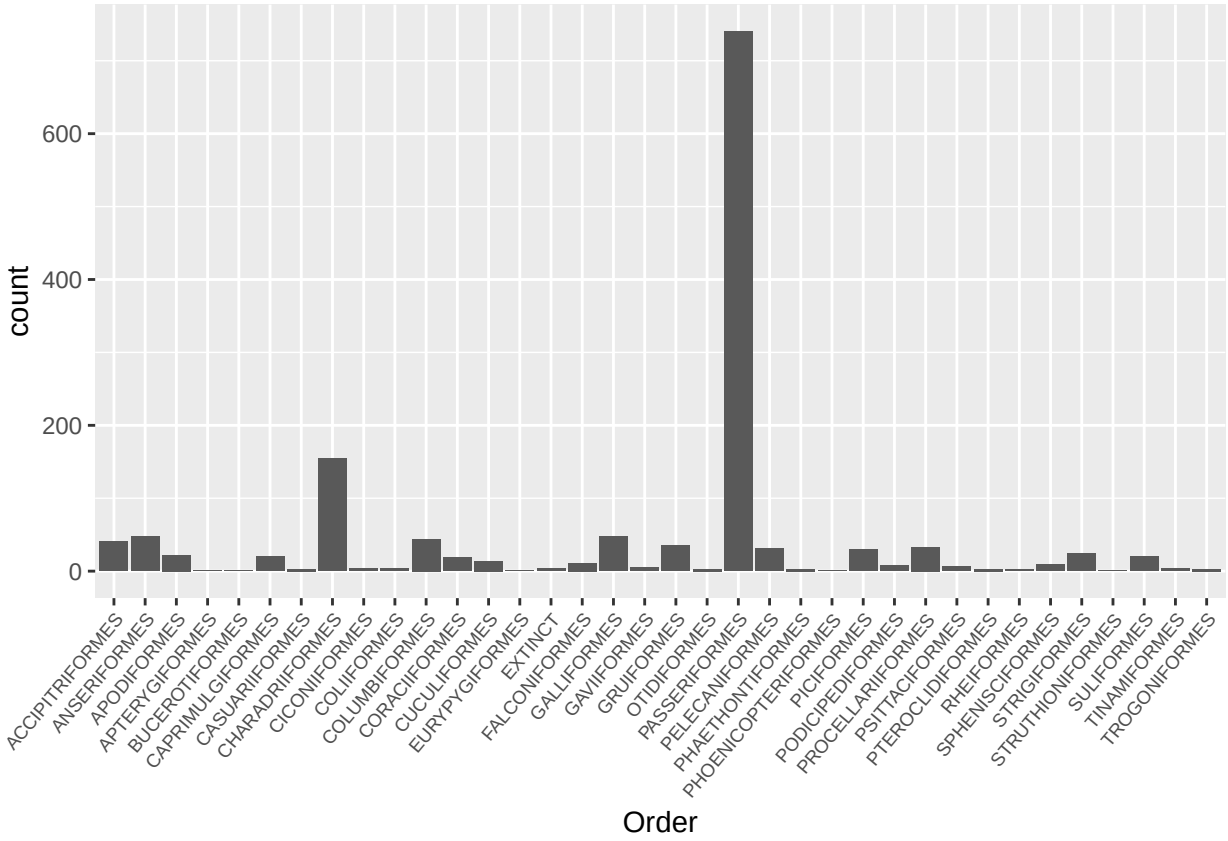
```
ggplot(data=egg, mapping=aes(x=Order)) +
  geom_bar()
```



This gives us a histogram, which is not what we wanted, but let's work with it. The first issue is that we can't read the tick labels on the X axis (in `ggplot` terms, the axis label here is "Order" and the tick labels are the labels for the levels on that axis).

Let's rotate them and make them a bit smaller so we can see them.

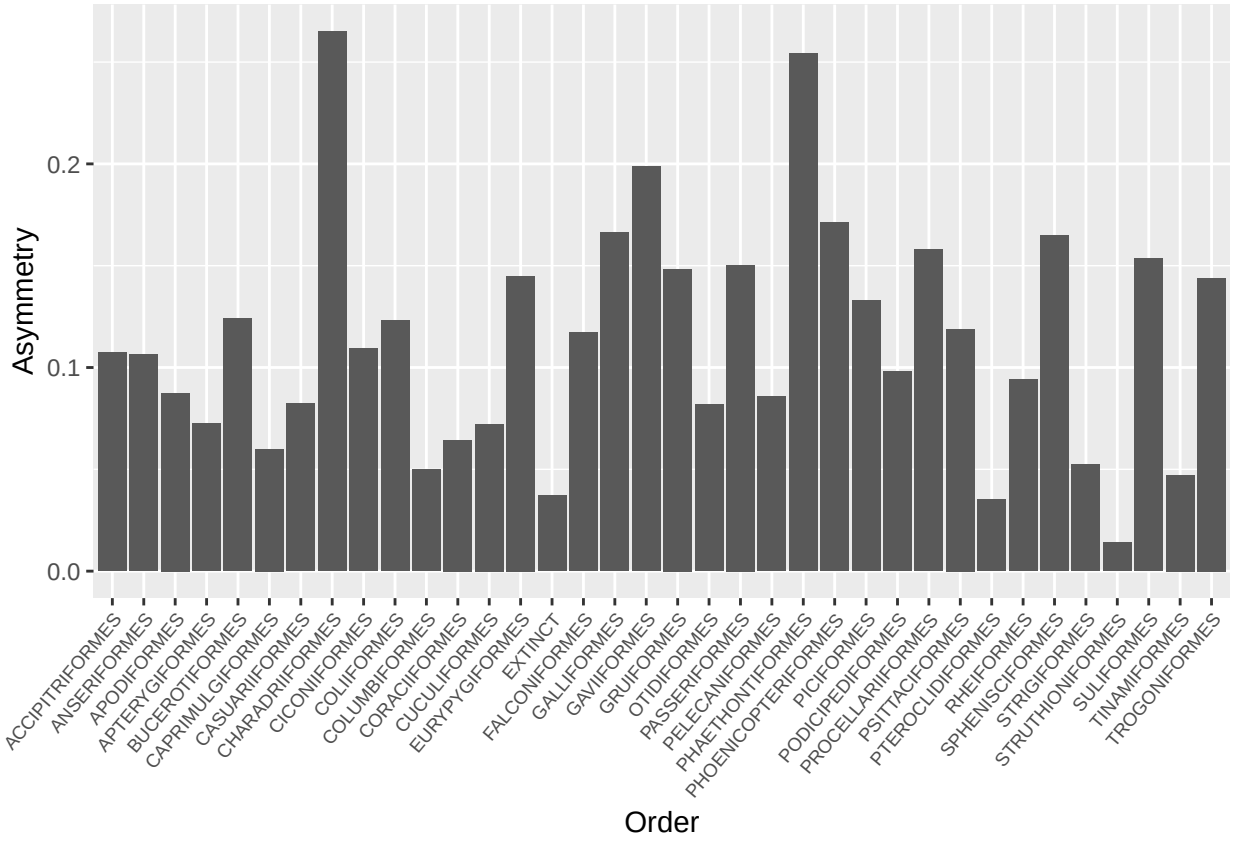
```
ggplot(data=egg, mapping=aes(x=Order)) +
  geom_bar() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Looks better, for a histogram. But we wanted to see the `Asymmetry` value for each order.

Let's tell `ggplot` that we want to see the mean values. For this we need to change the `stat` argument of `geom_bar()` from `count` (its default, which produces a histogram) to `summary`, and specify that the function we want to use to calculate our Y axis values is `mean()`.

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar(stat="summary", fun.y="mean") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Looks good. But for bar plots it's always good to have error bars. We'll need to use `geom_errorbar()`.

```
?geom_errorbar
```

What if we want to use confidence intervals for those bars? How do we calculate them? Well, one way is:

```
?mean_cl_normal
```

But for this, it shows we need to load the `Hmisc` package.

```
install.packages("Hmisc")
```

```
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

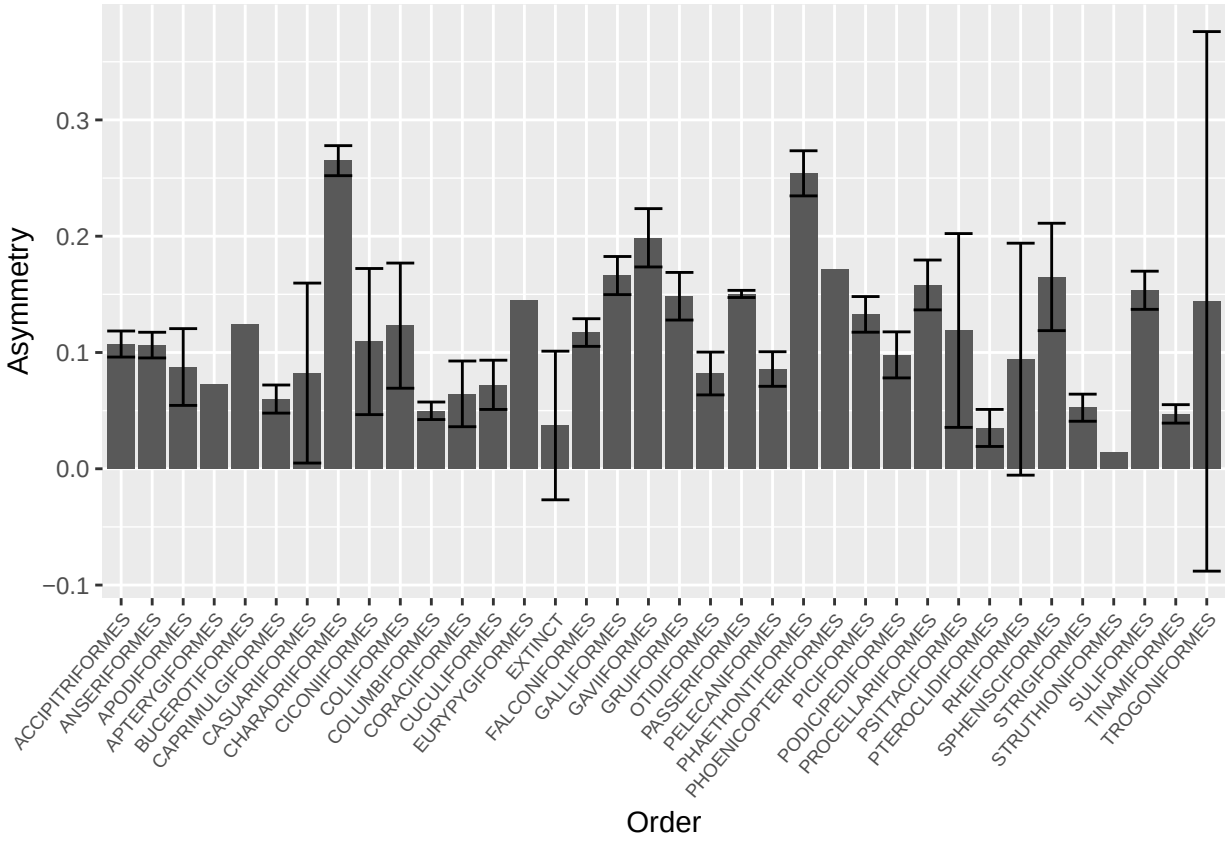
```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```



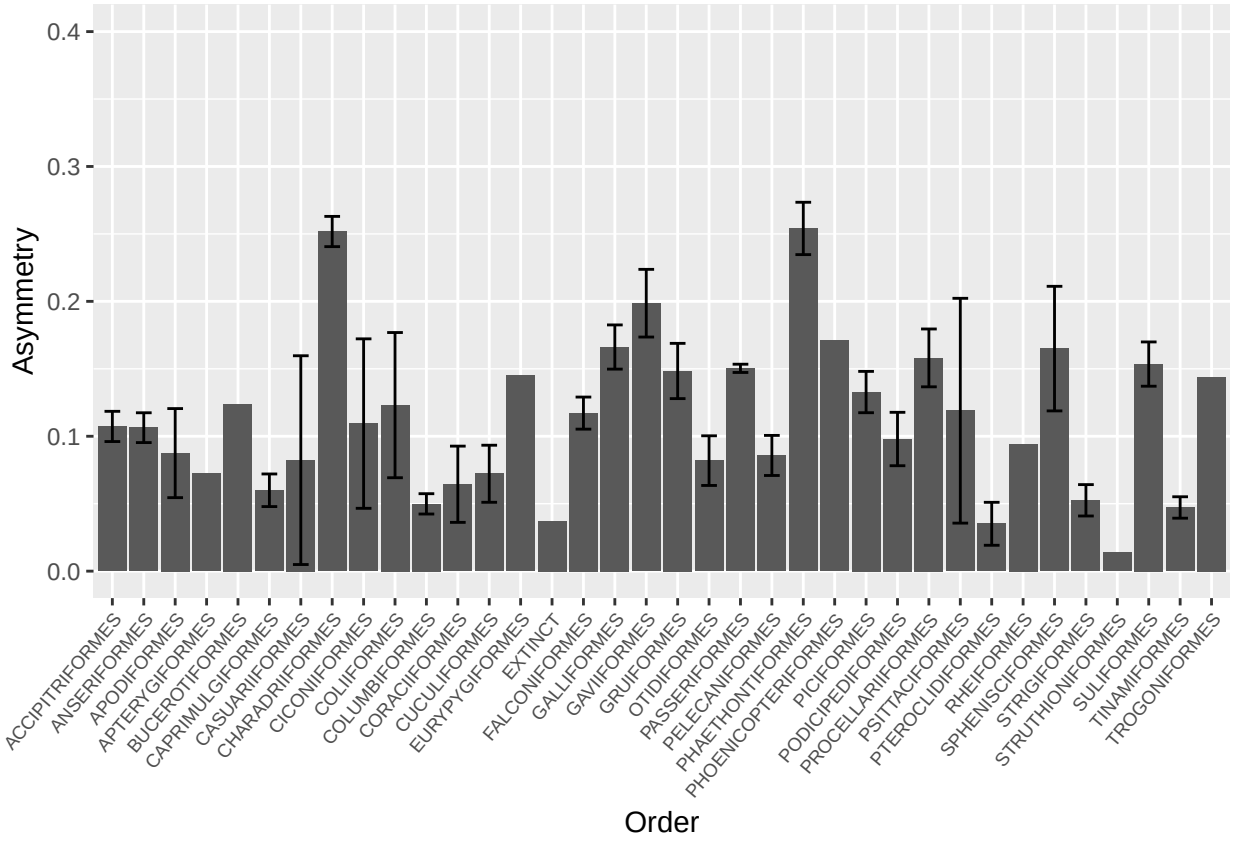
This put our axes out of whack. Let's fix them, and reduce the width of the error bars.

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  ylim(c(0,0.4)) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 11 rows containing non-finite values (stat_summary).
```

```
## Warning: Removed 11 rows containing non-finite values (stat_summary).
```

```
## Warning: Removed 8 rows containing missing values (geom_errorbar).
```



Hm. What happened?

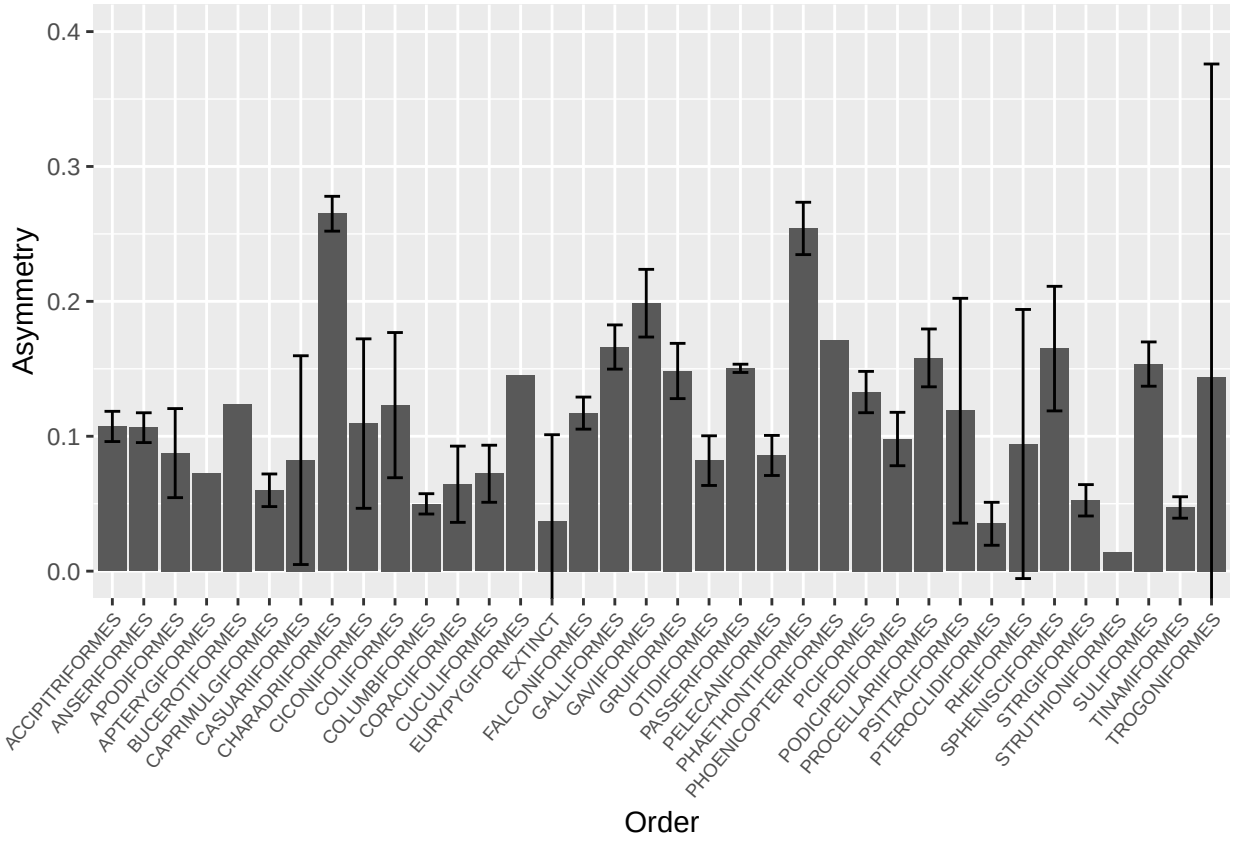
Instead of just "zooming in" on the area we wanted, it also dropped the confidence intervals that went outside that range.

```
?ylim
```

```
?coord_cartesian
```

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```

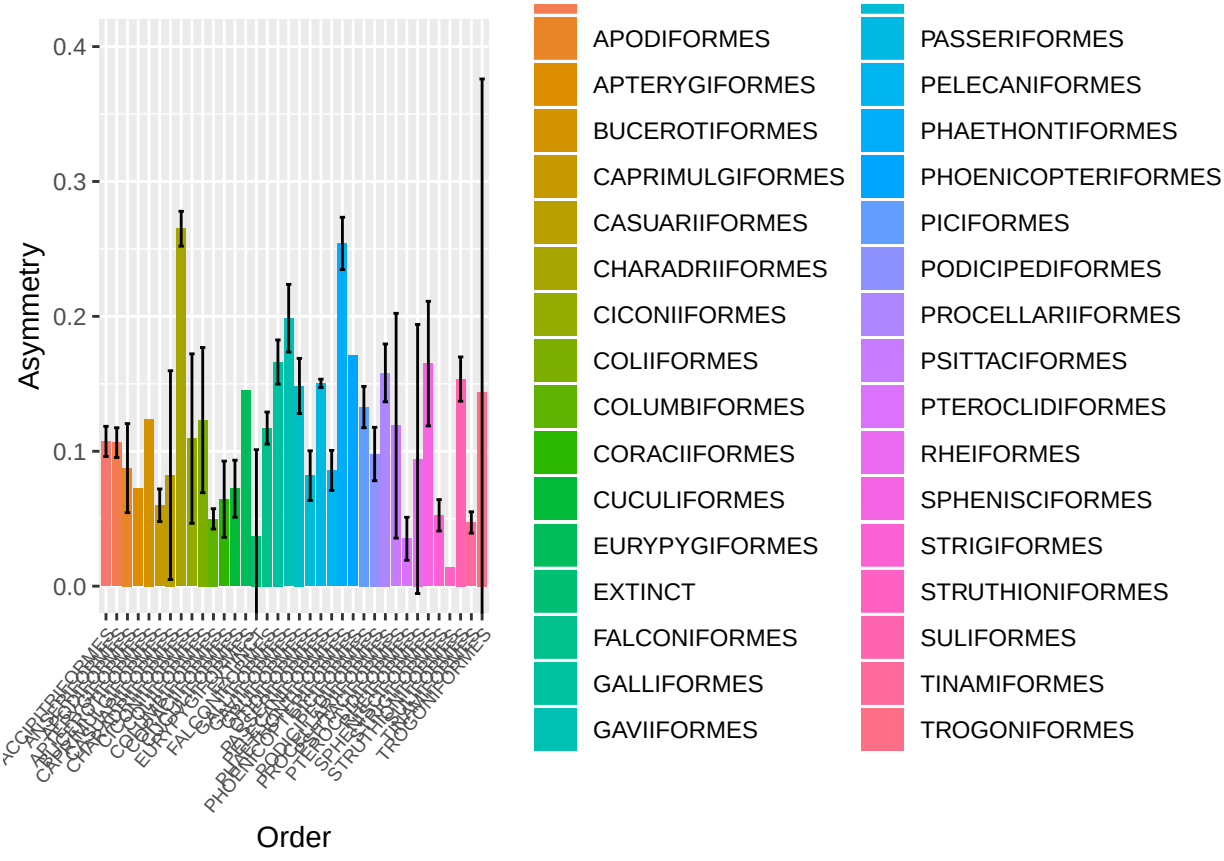


Good. We could do more to try to bring the lower intervals up to zero, but we can leave that for now (it would be easier to do using the `summarySE()` method shown later).

Gray on gray is boring. Let's add some fill colors (even though, in this case, they don't add anything meaningful to the plot).

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry, fill=Order)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

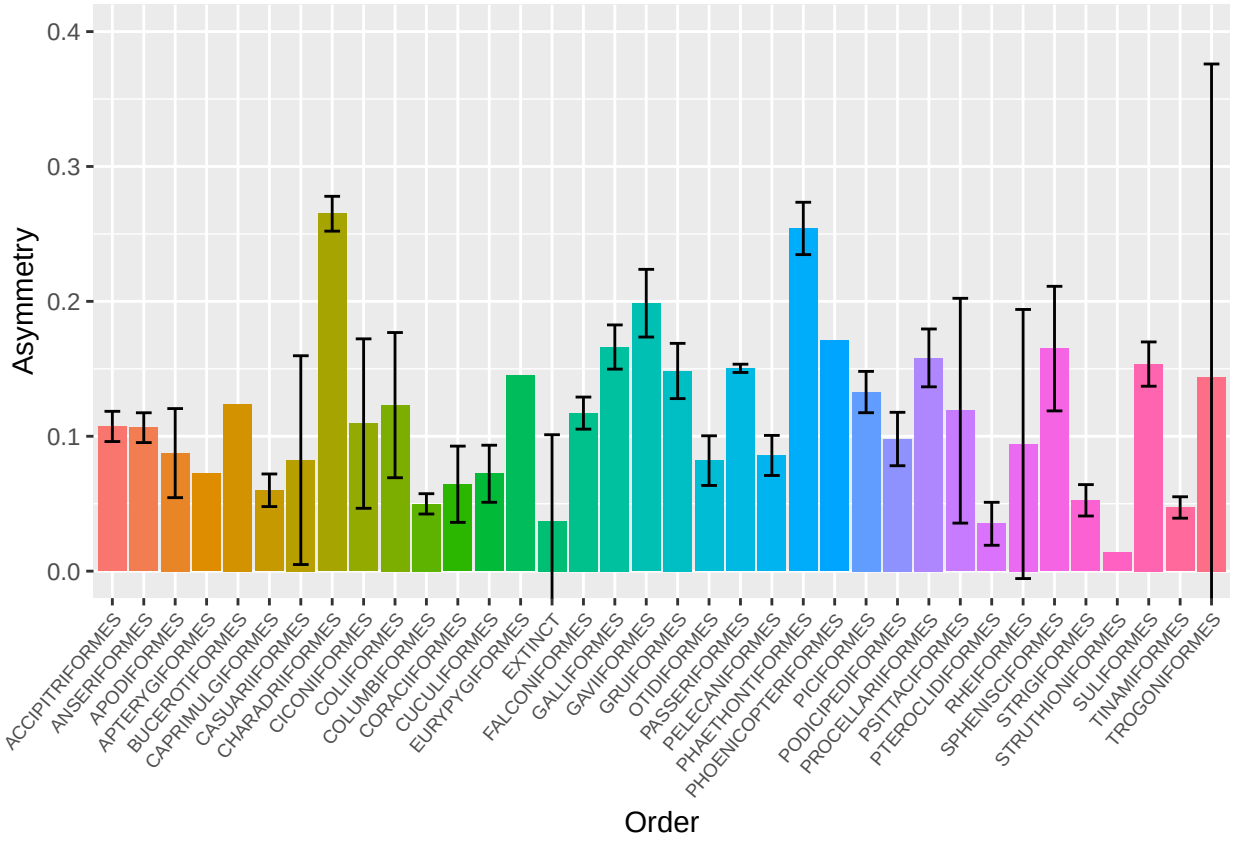
```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```



The legend takes up half the plot. Let's remove it.

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry, fill=Order)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  guides(fill=FALSE) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```

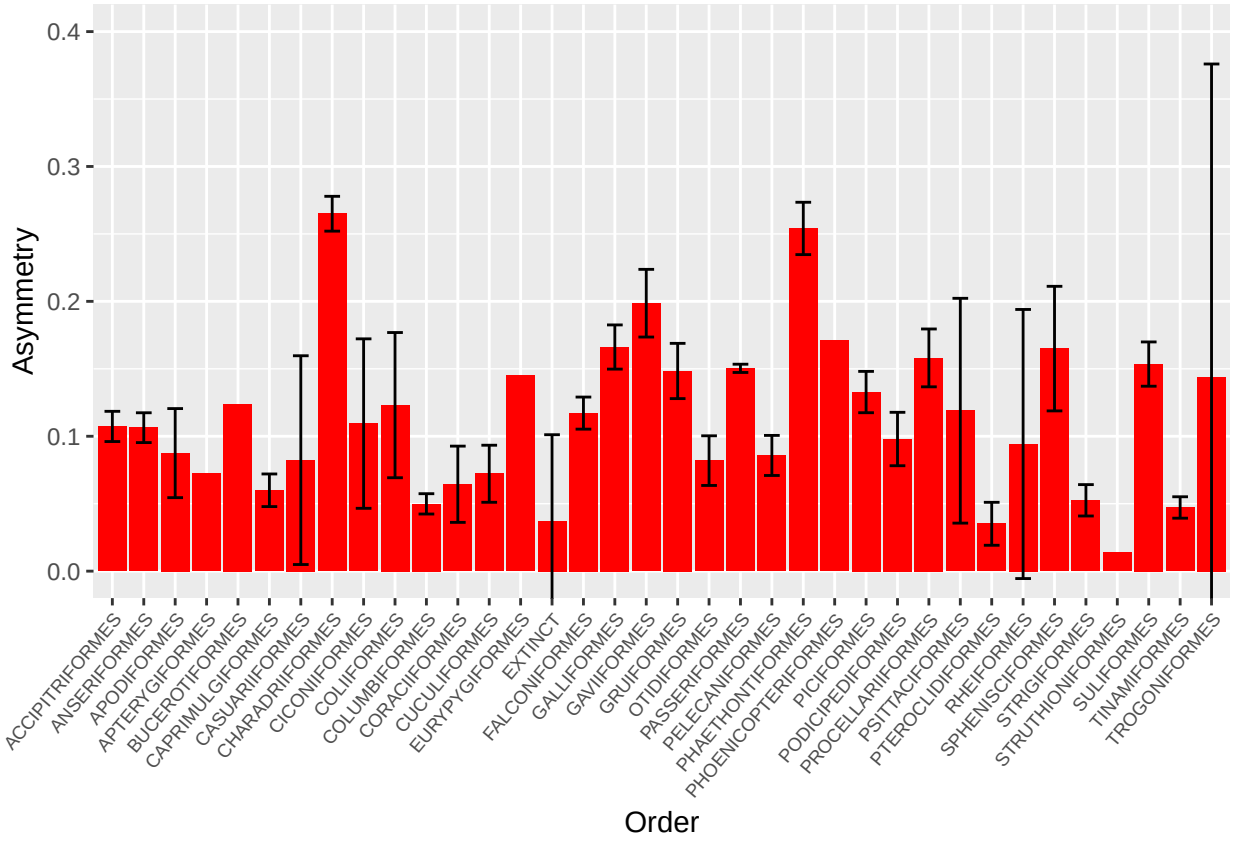


If we want to specify a single color, we can move our `fill` argument to the `geom_bar()` function and specify what color we want (either by name or hexadecimal value, e.g. "#FF0000").

You can find a list of colors defined in R here: http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry)) +
  geom_bar(stat="summary", fun.y="mean", fill="red") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  guides(fill=FALSE) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```

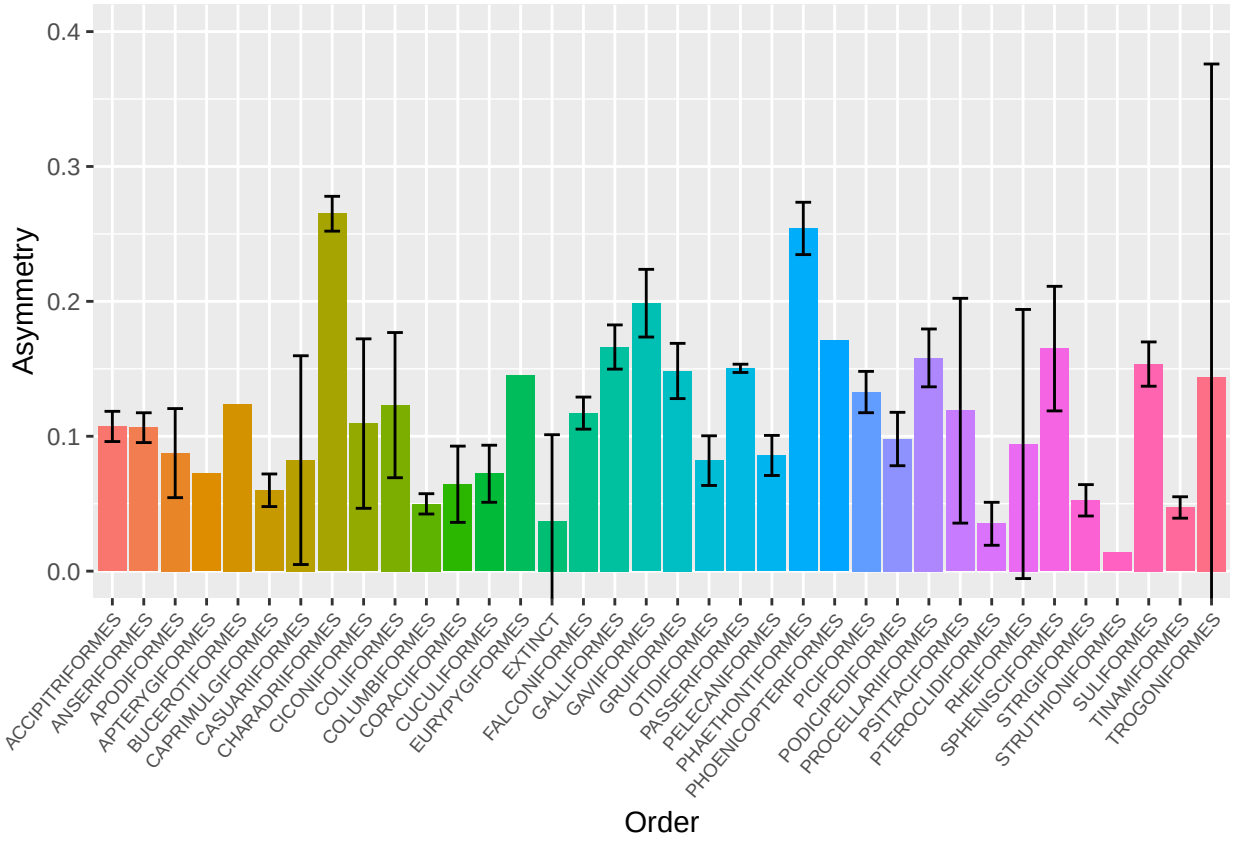


As with everything in R, there are multiple ways to do the same thing.

We can use the `stat_summary()` function instead of the `geom` functions, and specify the `geom` in the arguments:

```
ggplot(data=egg, mapping=aes(x=Order, y=Asymmetry, fill=Order)) +
  stat_summary(geom="bar", fun.y=mean) +
  stat_summary(geom="errorbar", fun.data=mean_cl_normal,
               width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  guides(fill=FALSE) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```



Another, more useful way is to use the `summarySE()` function to make a new data frame with the information we need.

If I know a function that I want to use but can't remember what package it's in, I can search from it (as long as I have that package installed).

```
??summarySE  # Will only search packages that you have installed
```

It's in `Rmisc`, from the creator of the R Companion website.

```
install.packages("Rmisc")
```

```
library(Rmisc)
```

```
## Loading required package: plyr
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize
```

Now we can read the help file for the function.

```
?summarySE
```

And use it to make a new data frame:

(Note: Pay attention to the arguments. Variable names need to be given as characters, and the `groupvars` argument wants a vector.)

```
eggAsymByOrder = summarySE(data=egg,
                           measurevar="Asymmetry",
                           groupvars=c("Order"))
```

```
## Warning in qt(conf.interval/2 + 0.5, datac$N - 1): NaNs produced
```

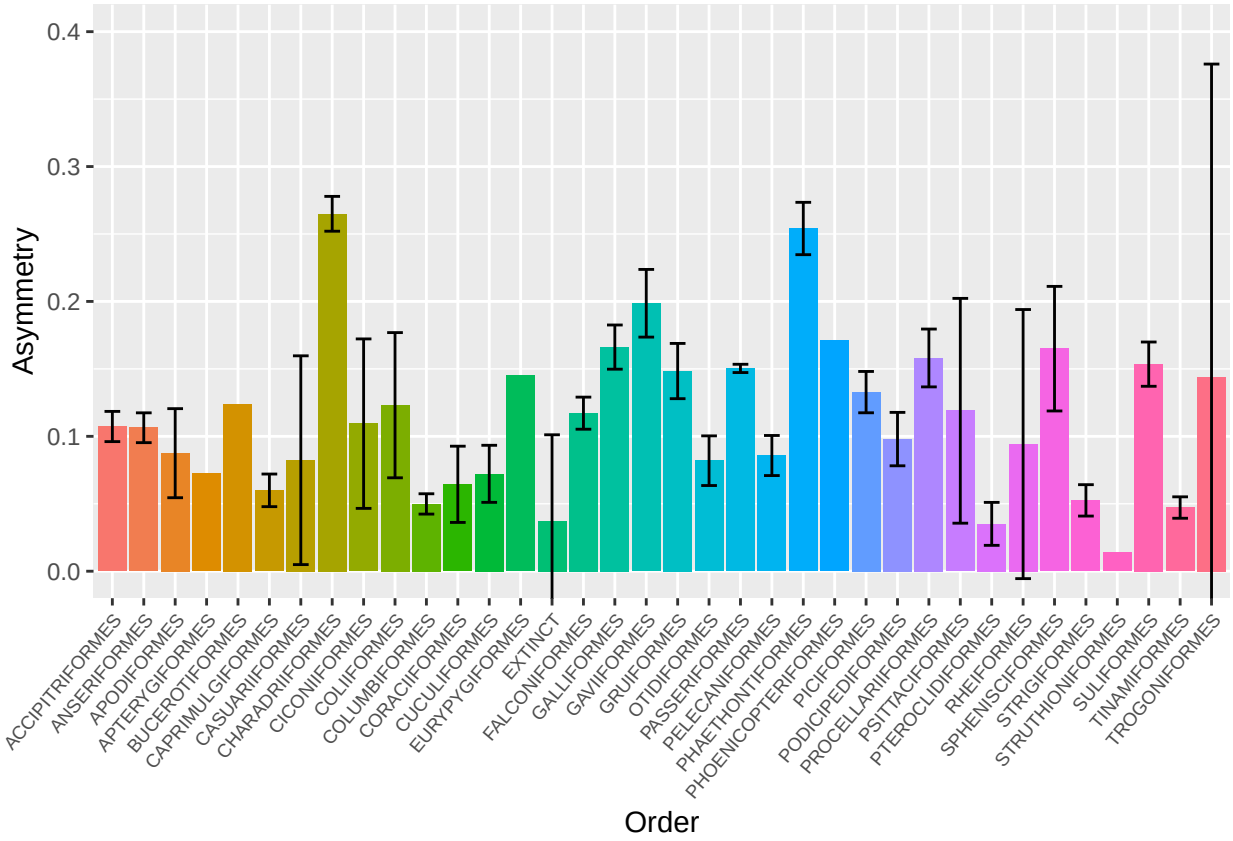
```
eggAsymByOrder
```

```
##                   Order   N  Asymmetry          sd          se          ci
## 1      ACCIPITRIFORMES  41 0.10728049 0.035478530 0.005540816 0.011198406
## 2         ANSERIFORMES  48 0.10635625 0.038055373 0.005492820 0.011050129
## 3          APODIFORMES  22 0.08750455 0.074462579 0.015875475 0.033014858
## 4       APTERYGIFORMES   1 0.07250000          NA          NA         NaN
## 5       BUCEROTIFORMES   1 0.12400000          NA          NA         NaN
## 6     CAPRIMULGIFORMES  20 0.05995500 0.025815366 0.005772491 0.012081963
## 7       CASUARIIFORMES   3 0.08230000 0.031153009 0.017986198 0.077388366
## 8      CHARADRIIFORMES 154 0.26496948 0.081065790 0.006532467 0.012905480
## 9        CICONIIFORMES   4 0.10940000 0.039479699 0.019739850 0.062821012
## 10         COLIIFORMES   4 0.12307500 0.033823697 0.016911849 0.053821050
## 11       COLUMBIFORMES  44 0.04987955 0.024793300 0.003737731 0.007537852
## 12       CORACIIFORMES  19 0.06442632 0.058624870 0.013449468 0.028256283
## 13        CUCULIFORMES  14 0.07220000 0.036647300 0.009794403 0.021159521
## 14      EURYPYGIFORMES   1 0.14500000          NA          NA         NaN
## 15             EXTINCT   4 0.03722500 0.040169174 0.020084587 0.063918119
## 16       FALCONIFORMES  11 0.11716364 0.017703122 0.005337692 0.011893120
## 17         GALLIFORMES  48 0.16615625 0.056425141 0.008144268 0.016384153
## 18         GAVIIFORMES   5 0.19864000 0.020200322 0.009033859 0.025082012
## 19          GRUIFORMES  35 0.14838571 0.059621774 0.010077919 0.020480796
## 20         OTIDIFORMES   3 0.08193333 0.007409678 0.004277980 0.018406661
## 21       PASSERIFORMES 740 0.15035392 0.042467665 0.001561142 0.003064802
## 22      PELECANIFORMES  31 0.08577097 0.040492791 0.007272720 0.014852876
## 23    PHAETHONTIFORMES   3 0.25406667 0.007811743 0.004510112 0.019405446
## 24 PHOENICOPTERIFORMES   1 0.17120000          NA          NA         NaN
## 25          PICIFORMES  30 0.13277667 0.041035751 0.007492069 0.015323001
## 26    PODICIPEDIFORMES   8 0.09795000 0.023687369 0.008374749 0.019803136
## 27   PROCELLARIIFORMES  33 0.15808788 0.060472038 0.010526831 0.021442452
## 28      PSITTACIFORMES   6 0.11893333 0.079404576 0.032416782 0.083329992
## 29    PTEROCLIDIFORMES   3 0.03510000 0.006409368 0.003700450 0.015921753
## 30          RHEIFORMES   2 0.09425000 0.011101576 0.007850000 0.099743707
## 31     SPHENISCIFORMES   9 0.16497778 0.060067790 0.020022597 0.046172191
## 32        STRIGIFORMES  24 0.05252083 0.027568413 0.005627379 0.011641120
## 33    STRUTHIONIFORMES   1 0.01400000          NA          NA         NaN
## 34          SULIFORMES  20 0.15350500 0.035058905 0.007839409 0.016408073
## 35        TINAMIFORMES   4 0.04720000 0.004989990 0.002494995 0.007940188
## 36       TROGONIFORMES   3 0.14396667 0.093391988 0.053919889 0.231998558
```

We can see some of these are missing values (`NA` or `NaN`, "not a number") because those orders only had one observation (see the `N` column) and you can't get a confidence interval from one value.

```
ggplot(data=eggAsymByOrder, mapping=aes(x=Order, y=Asymmetry, fill=Order)) +
  geom_bar(stat="identity") +
  geom_errorbar(aes(ymin=Asymmetry-ci, ymax=Asymmetry+ci),
                width=0.5) +
  coord_cartesian(ylim=c(0,0.4)) +
  guides(fill=FALSE) +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

```
## Warning: Removed 5 rows containing missing values (geom_errorbar).
```



(pdf / Rmd)