Information design and data visualization
Week 3, Lecture 06

Richard E.W. Berl

Spring 2019

ggplot2

```
library(ggplot2)
```

Load the egg data as a CSV.

```
egg = read.csv("./data/aaj1945_DataS1_Egg_shape_by_species_v2.csv", header=T,
               stringsAsFactors=F)
colnames(egg)[7:9] = c("AvgLength","NumberOfImages","NumberOfEggs")
egg = egg[-c(1401,1402),]
```

```
str(egg)
```

```
## 'data.frame':    1400 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 nisus
##  $ Species       : chr  "Accipiter badius" "Accipiter cooperii" "Accipiter gentilis" "Accipiter nisus
##  $ 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     : num  3.86 4.9 5.99 4.04 3.87 ...
##  $ NumberOfImages: int  1 27 7 13 15 1 191 1 7 2 ...
##  $ NumberOfEggs  : int  2 103 18 61 57 1 391 2 17 4 ...
```

Subset the full data frame to Order Galliformes, chickens and other fowl.

```
eggGalli = egg[egg$Order == "GALLIFORMES",]
```

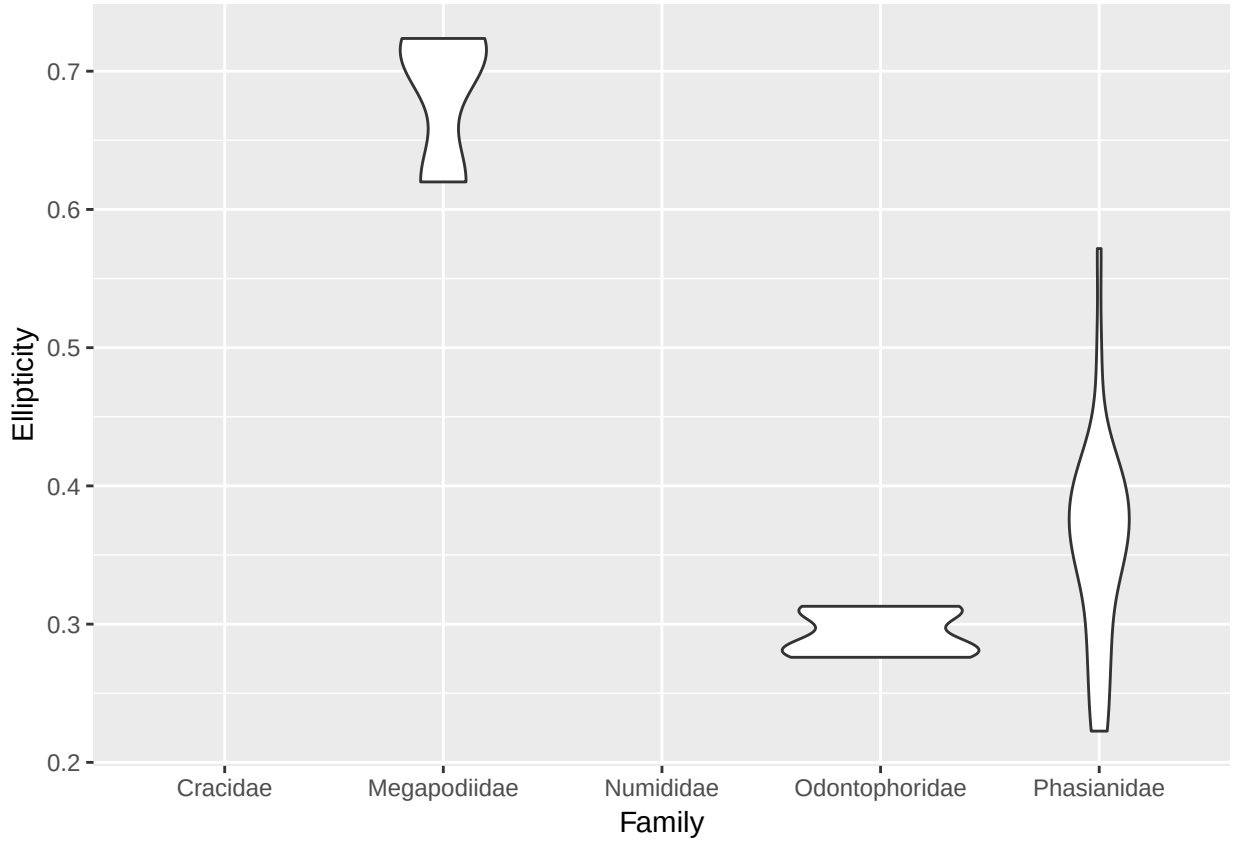
```
head(eggGalli)
```

```
##           Order         Family            MVZDatabase
## 392 GALLIFORMES       Cracidae         Ortalis vetula
## 393 GALLIFORMES   Megapodiidae    Macrocephalon maleo
## 394 GALLIFORMES   Megapodiidae   Megapodius freycinet
## 395 GALLIFORMES   Megapodiidae Megapodius pritchardii
## 396 GALLIFORMES      Numididae       Numida meleagris
## 397 GALLIFORMES Odontophoridae Callipepla californica
##                     Species Asymmetry Ellipticity AvgLength NumberOfImages
## 392         Ortalis vetula    0.1178      0.4590    6.0847             10
## 393    Macrocephalon maleo    0.0365      0.7237    8.3400              1
## 394   Megapodius freycinet    0.0551      0.6199    8.9621              1
## 395 Megapodius pritchardii    0.0221      0.7072    7.9284              1
## 396       Numida meleagris    0.2260      0.3197    5.2572              1
## 397 Callipepla californica    0.2364      0.2850    3.0381             55
```

```
##     NumberOfEggs
## 392           31
## 393            1
## 394            2
## 395            3
## 396           10
## 397          700
```

Violin plot

```
ggplot(data=eggGalli, mapping=aes(x=Family, y=Ellipticity)) +
  geom_violin()
```



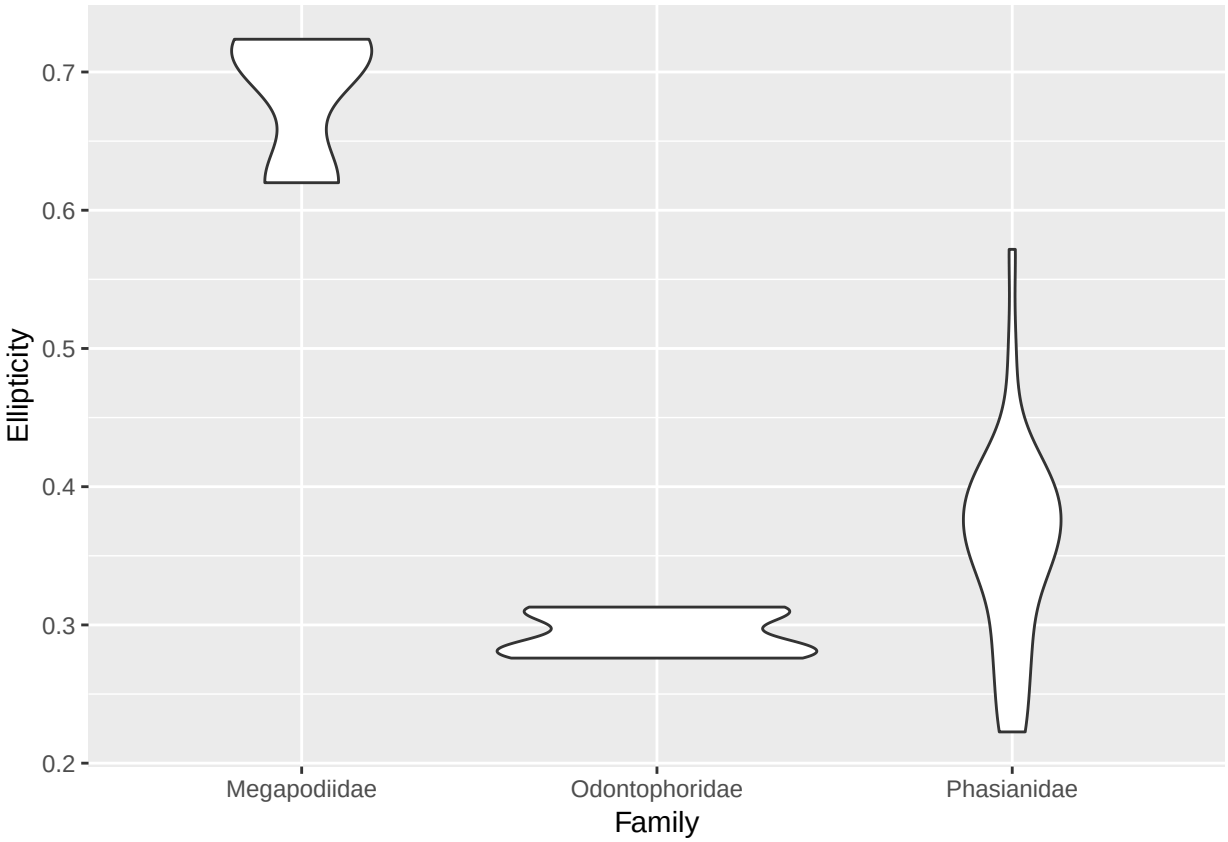

```
table(eggGalli$Family)
```

```
##
##       Cracidae   Megapodiidae      Numididae Odontophoridae    Phasianidae
##              1              3              1              7             36
```

Cracidae and Numididae don't show distributions because they each only have 1 observation.

We can subset the plot to not show them. Alternatively, we could add a `geom_point()` layer to show them as single points.

```
ggplot(data=eggGalli[eggGalli$Family != "Cracidae" &
                       eggGalli$Family != "Numididae",],
       mapping=aes(x=Family, y=Ellipticity)) +
  geom_violin()
```



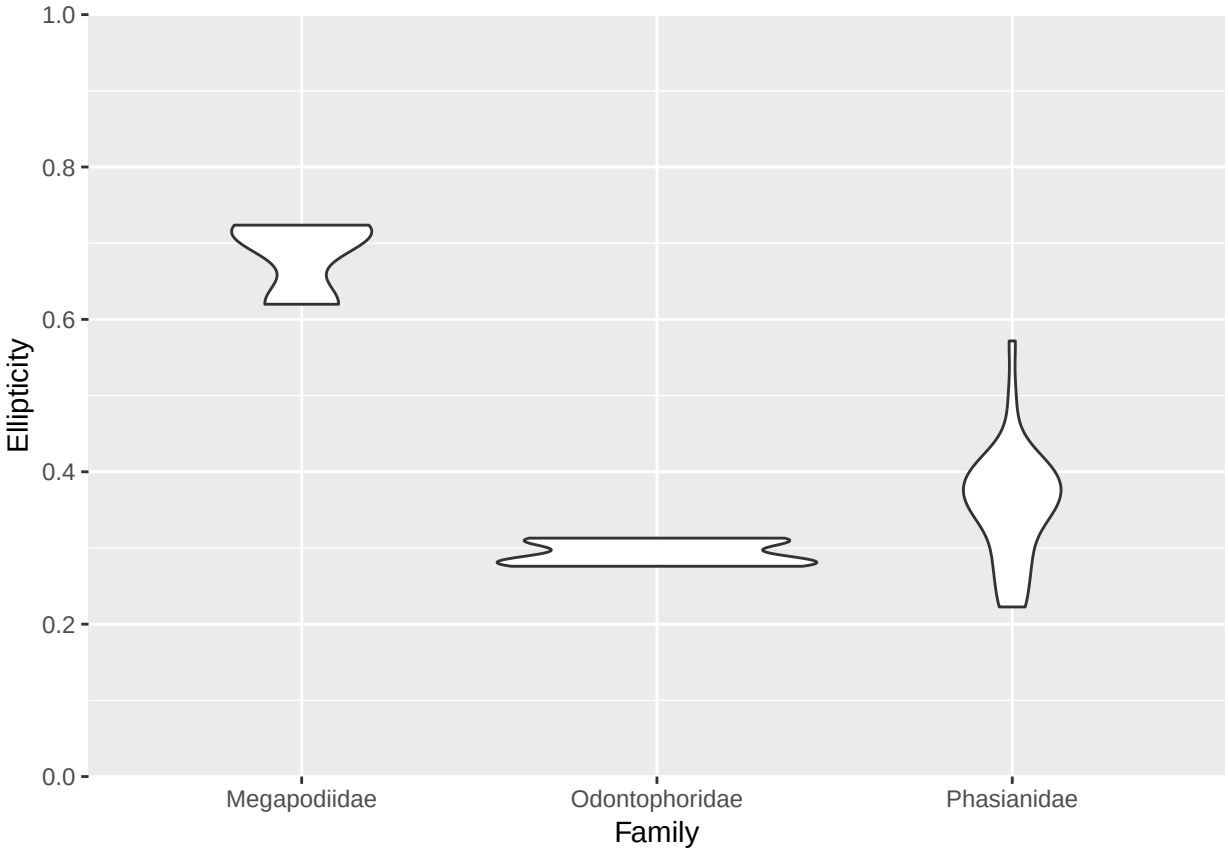
Our axis doesn't start at zero, which can be misleading.

```
range(egg$Ellipticity)
```

```
## [1] 0.0967 0.7237
```

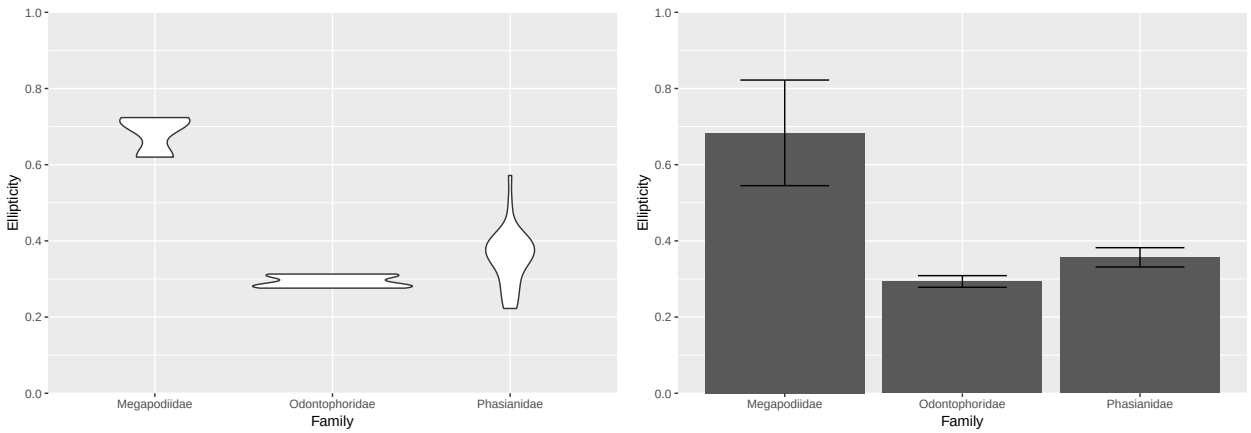
Looks like the range of possible values is probably 0 to 1. Let's change our axis, and add the `expand` argument to get rid of the extra space beyond the axis limits.

```
ggplot(data=eggGalli[eggGalli$Family != "Cracidae" &
                       eggGalli$Family != "Numididae",],
       mapping=aes(x=Family, y=Ellipticity)) +
  geom_violin() +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0))
```



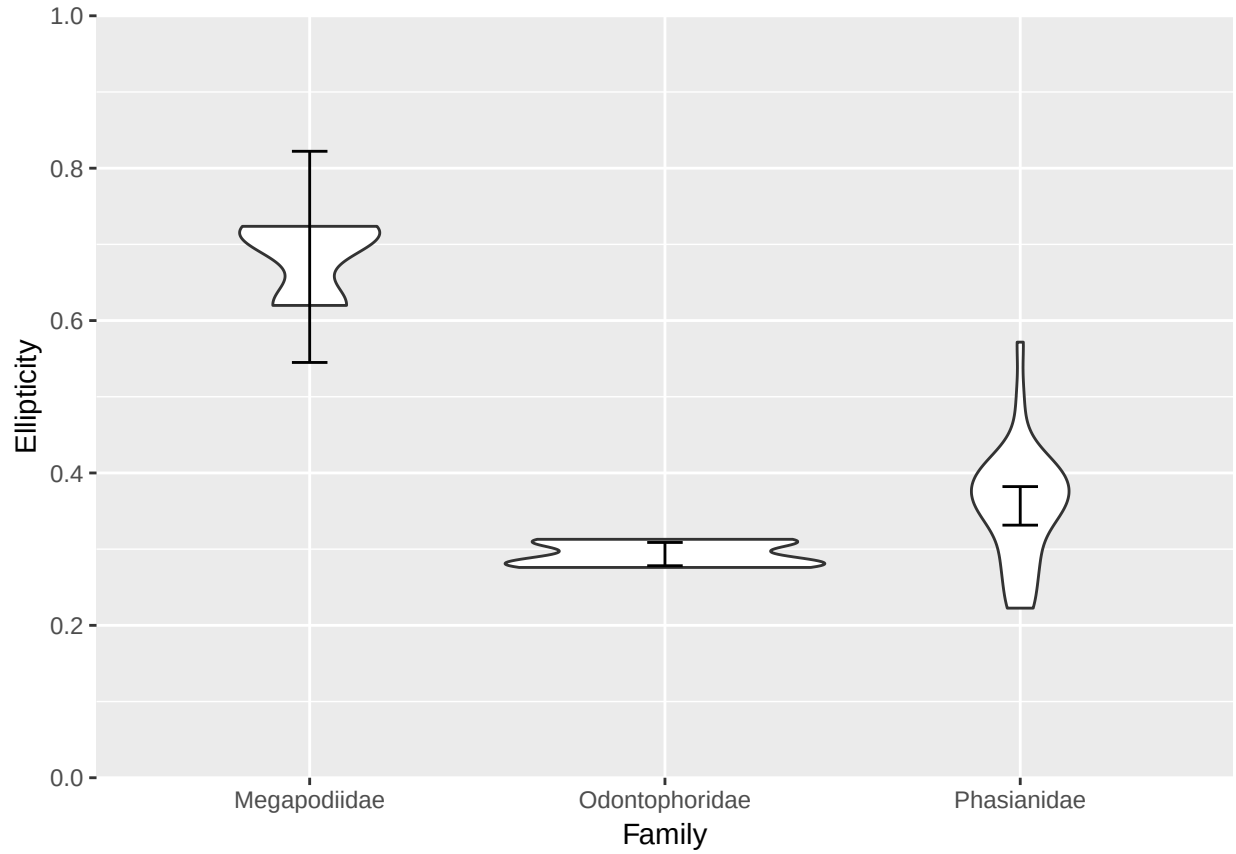
Compare this to a bar plot of the same data:

```
ggplot(data=eggGalli[eggGalli$Family != "Cracidae" &
                       eggGalli$Family != "Numididae",],
       mapping=aes(x=Family, y=Ellipticity)) +
  geom_violin() +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0))

ggplot(data=eggGalli[eggGalli$Family != "Cracidae" &
                       eggGalli$Family != "Numididae",],
       mapping=aes(x=Family, y=Ellipticity)) +
  geom_bar(stat="summary", fun.y="mean") +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.5) +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0))
```



If we want, we can take the best of both and lay the confidence intervals on top of the violin plot:

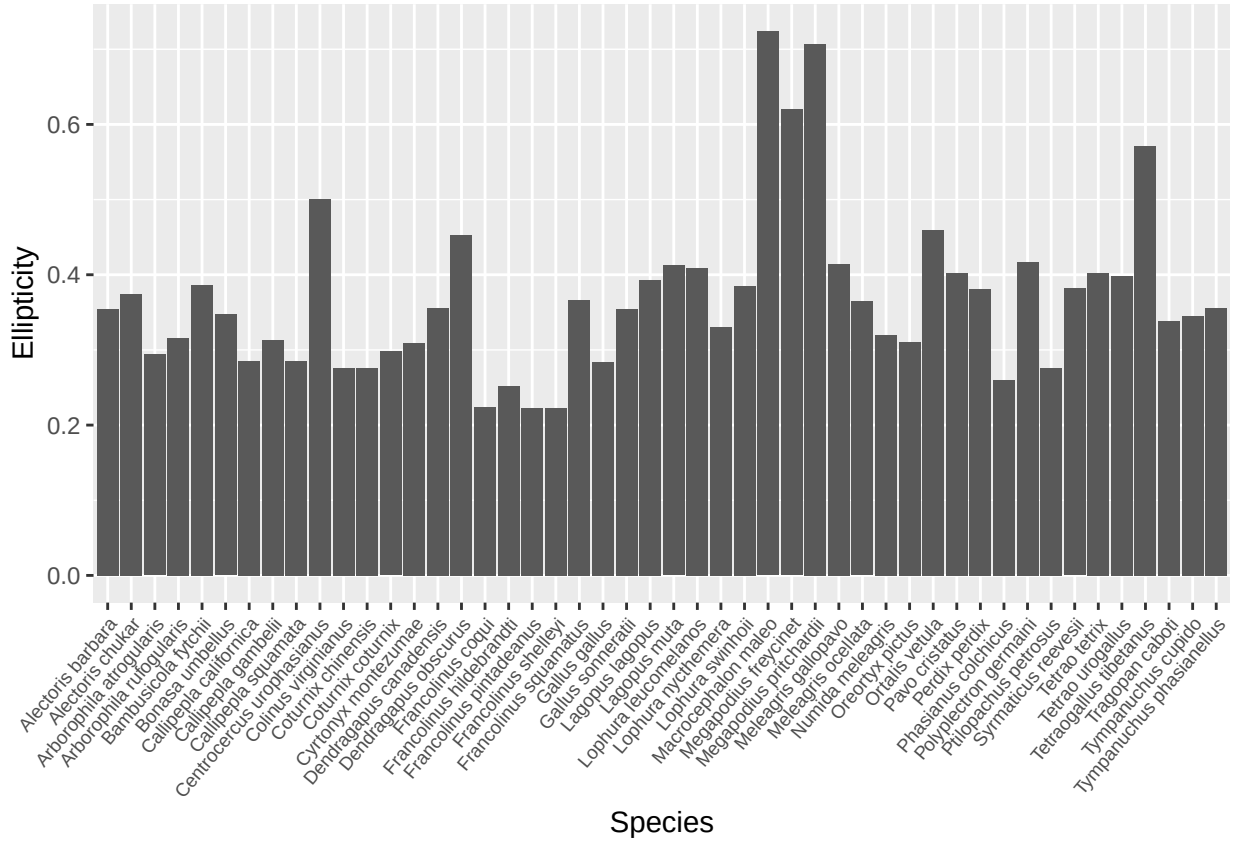
```
ggplot(data=eggGalli[eggGalli$Family != "Cracidae" &
                     eggGalli$Family != "Numididae",],
       mapping=aes(x=Family, y=Ellipticity)) +
  geom_violin() +
  geom_errorbar(stat="summary", fun.data="mean_cl_normal",
                width=0.1) +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0))
```



Facets

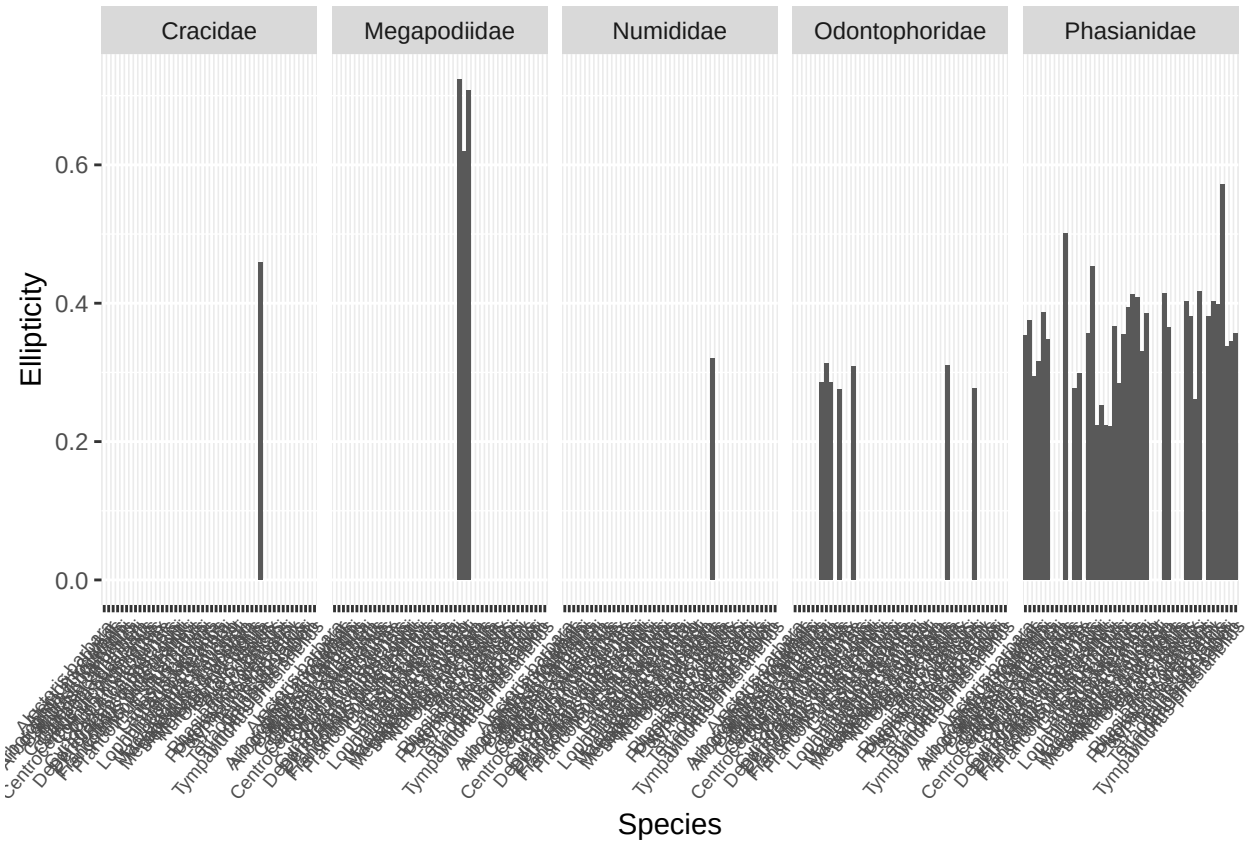
Plot all ellipticity values by species.

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



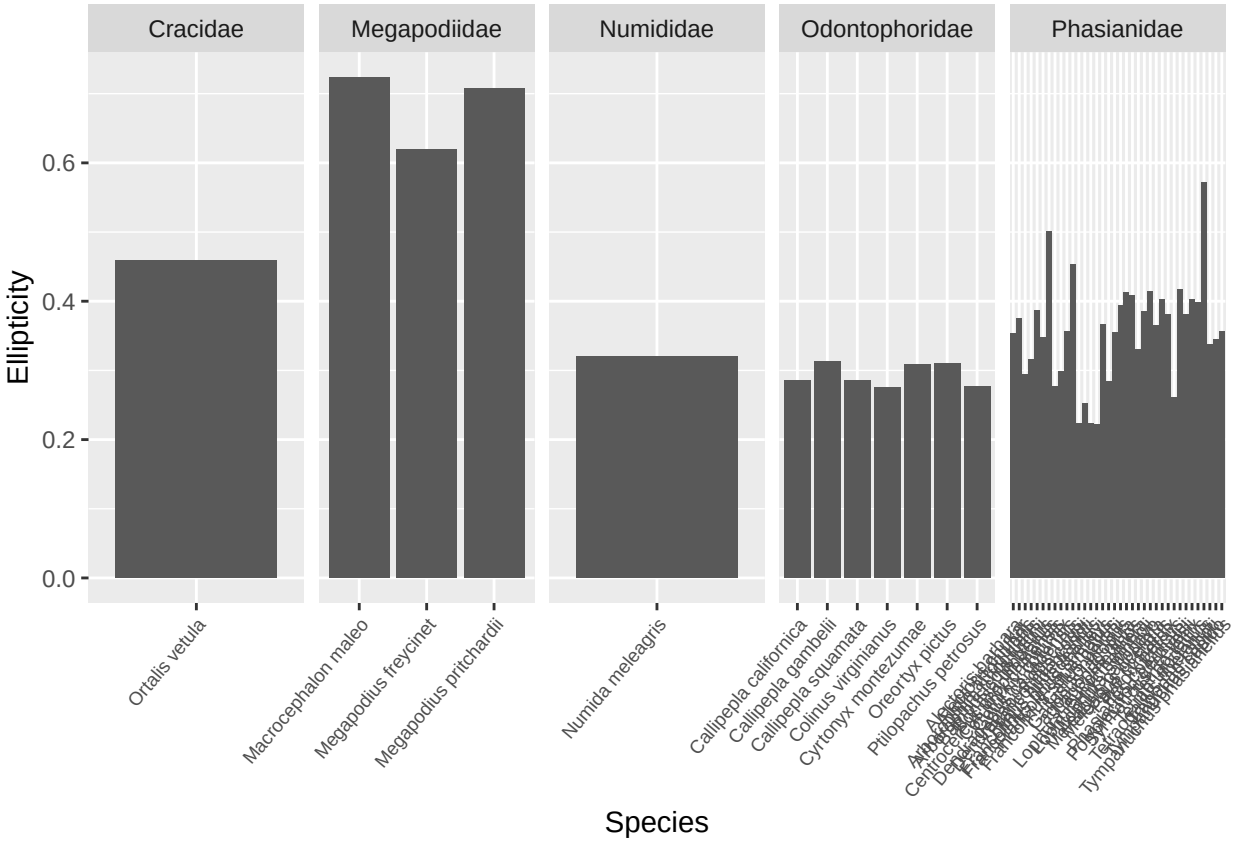
Add a facet to group them by family.

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family)) +
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



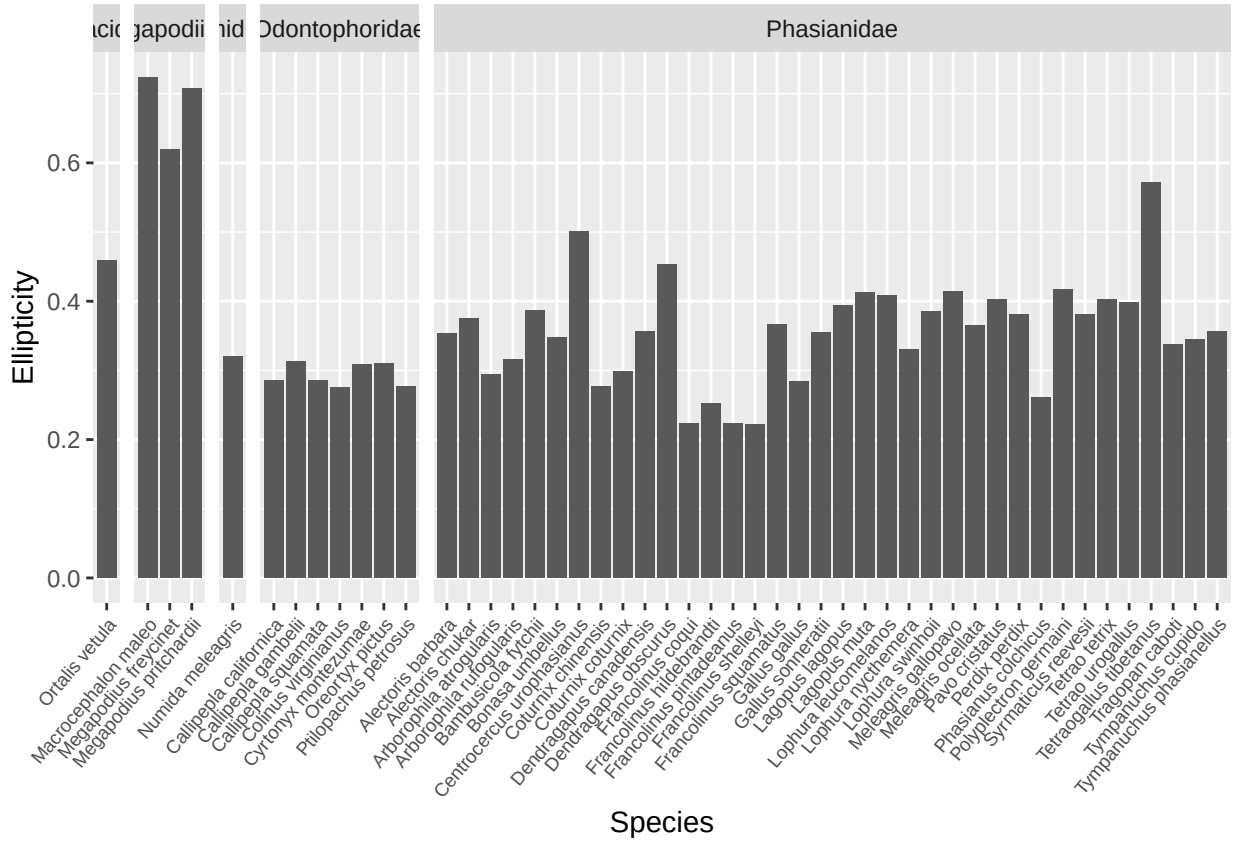
. . . But all of our values appear under each facet. Not what we want. So we add `scales="free"` to allow scales to vary across rows and columns.

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free") +
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Still not quite right, as some bars are now wider than others and may seem like they're communicating some other information. So we add `space="free"` to allow panel sizes to vary.

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



And we've now made a new problem in that our facet labels don't fit anymore. So let's use the `recode()` function from `dplyr` to change those values.

```
unique(eggGalli$Family)
```

```
## [1] "Cracidae"       "Megapodiidae"   "Numididae"      "Odontophoridae"
## [5] "Phasianidae"
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

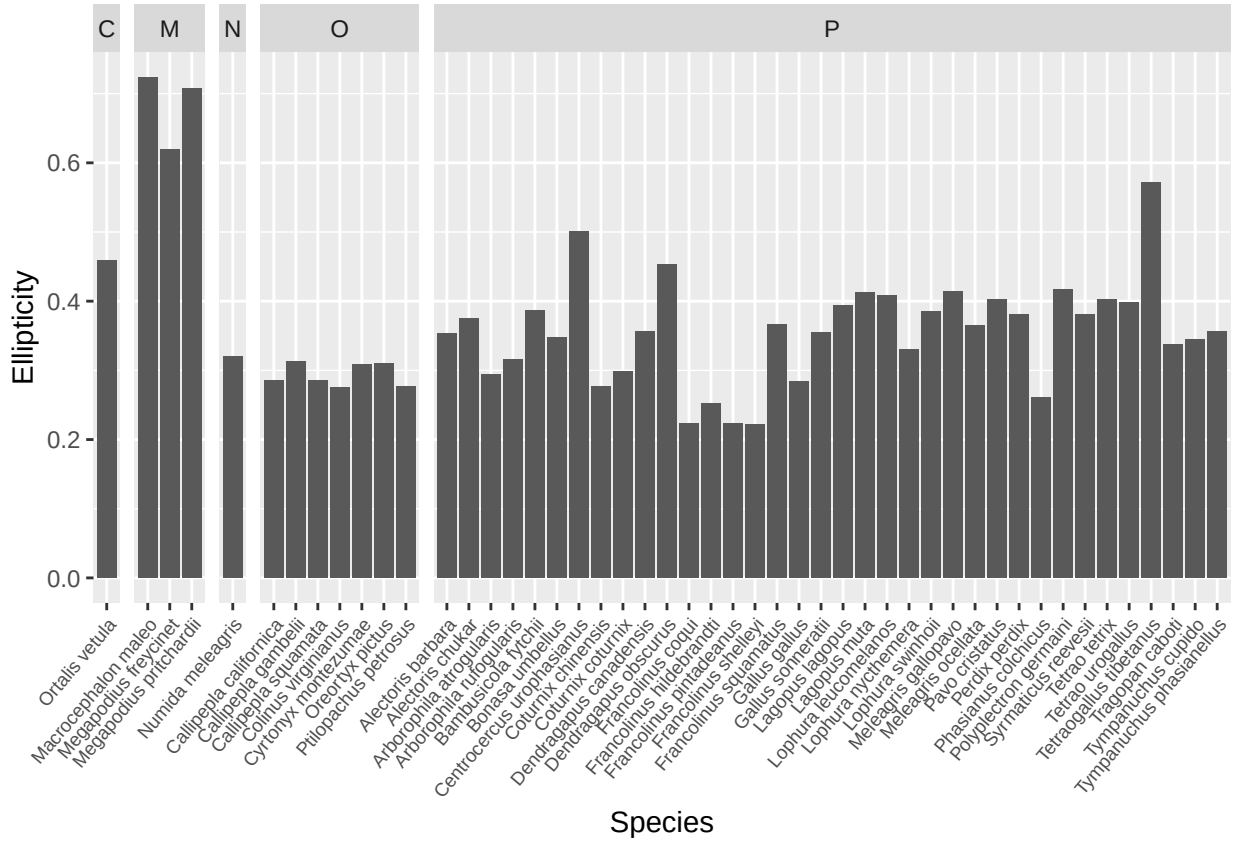
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
eggGalli$Family = recode(eggGalli$Family,
                         Cracidae = "C",
                         Megapodiidae = "M",
                         Numididae = "N",
                         Odontophoridae = "O",
                         Phasianidae = "P")
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
```

```
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```

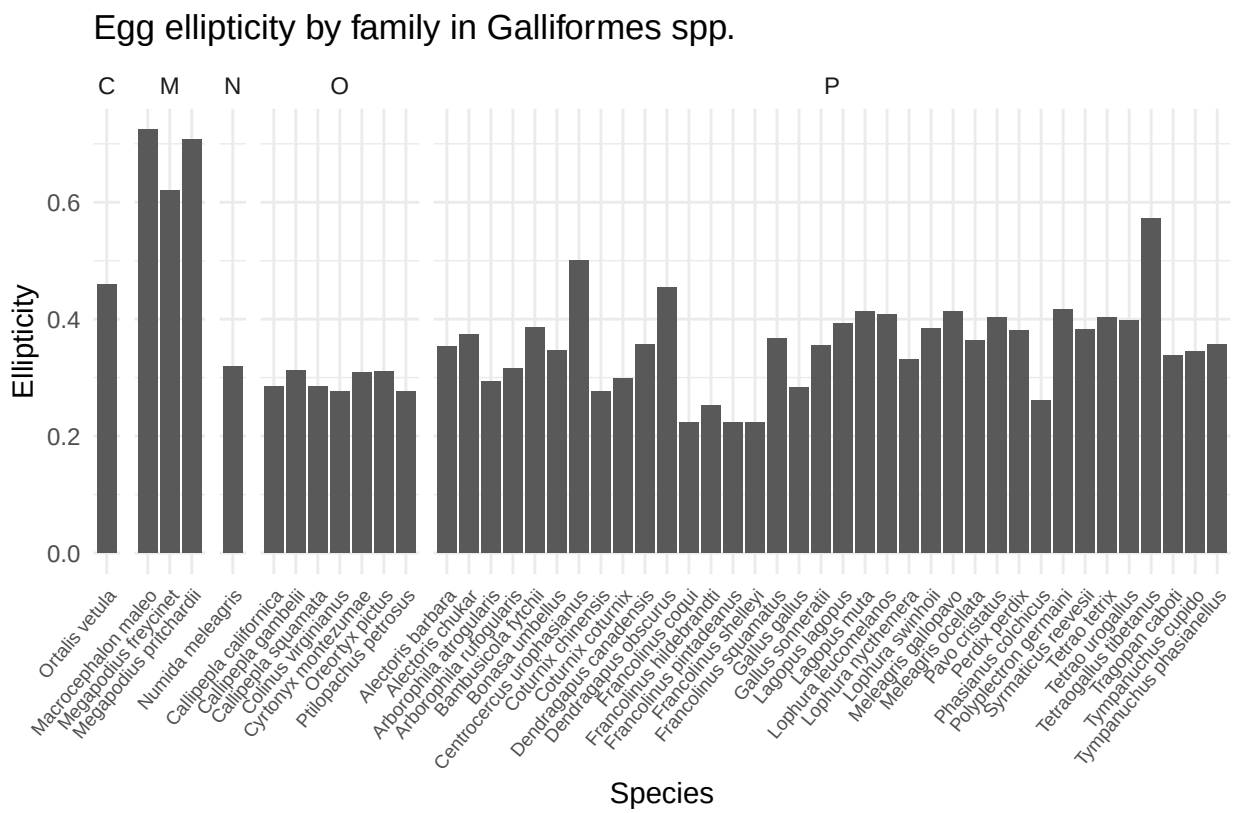


Themes

`theme_gray()` is the default

`theme_minimal()`

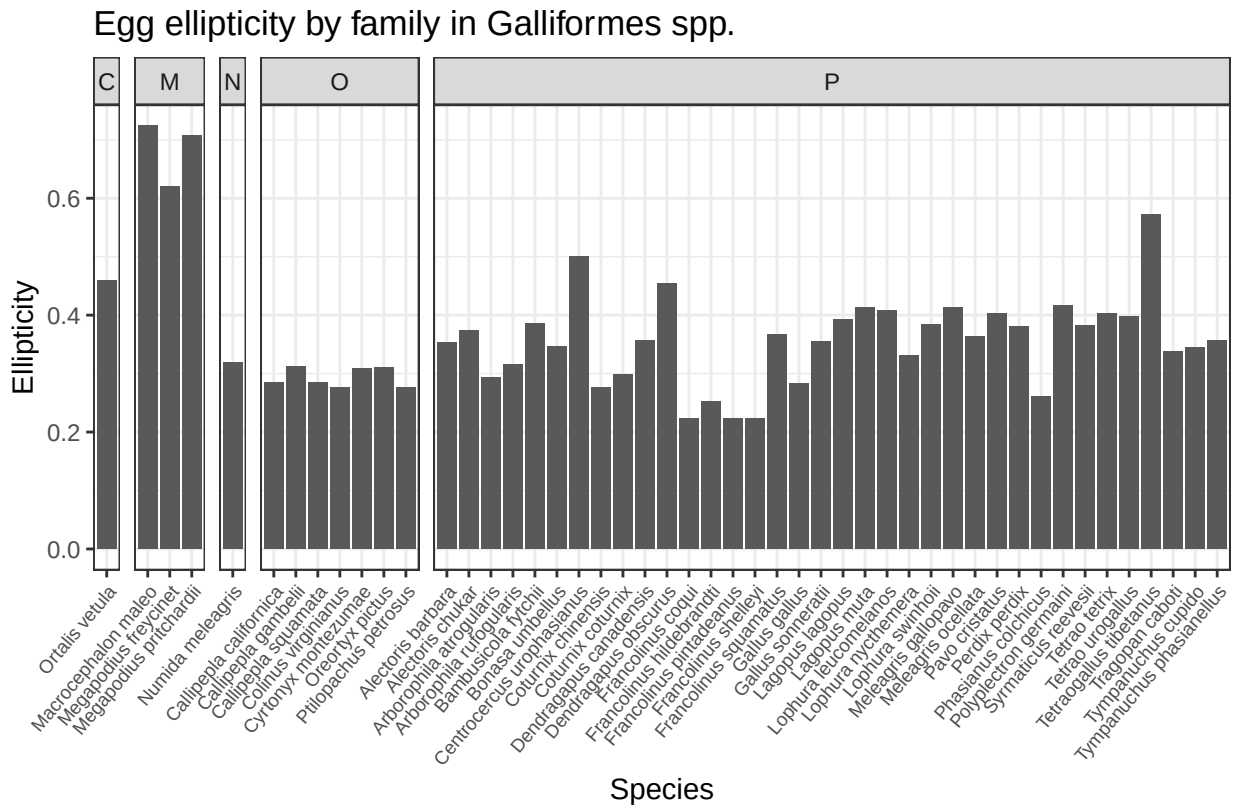
```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Data from Stoddard et al. (2017)

```
theme_bw()
```

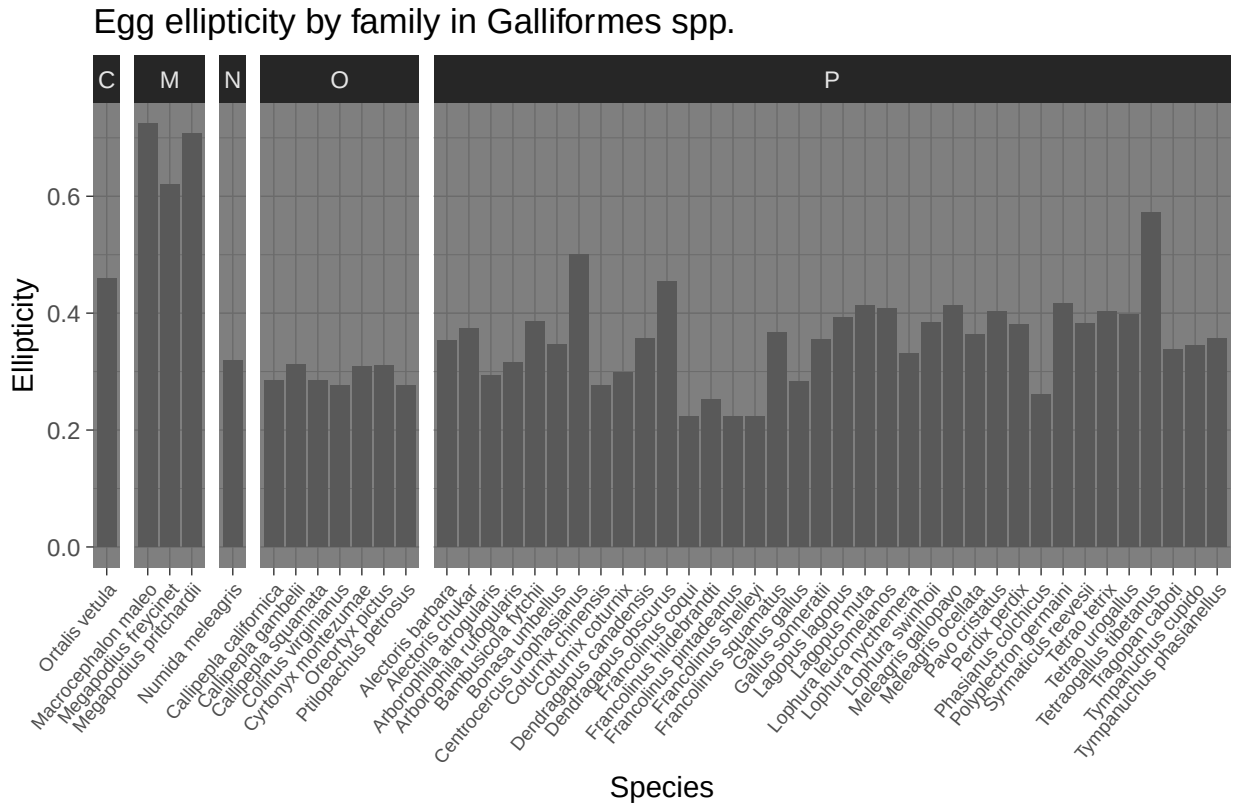
```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_bw() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Data from Stoddard et al. (2017)

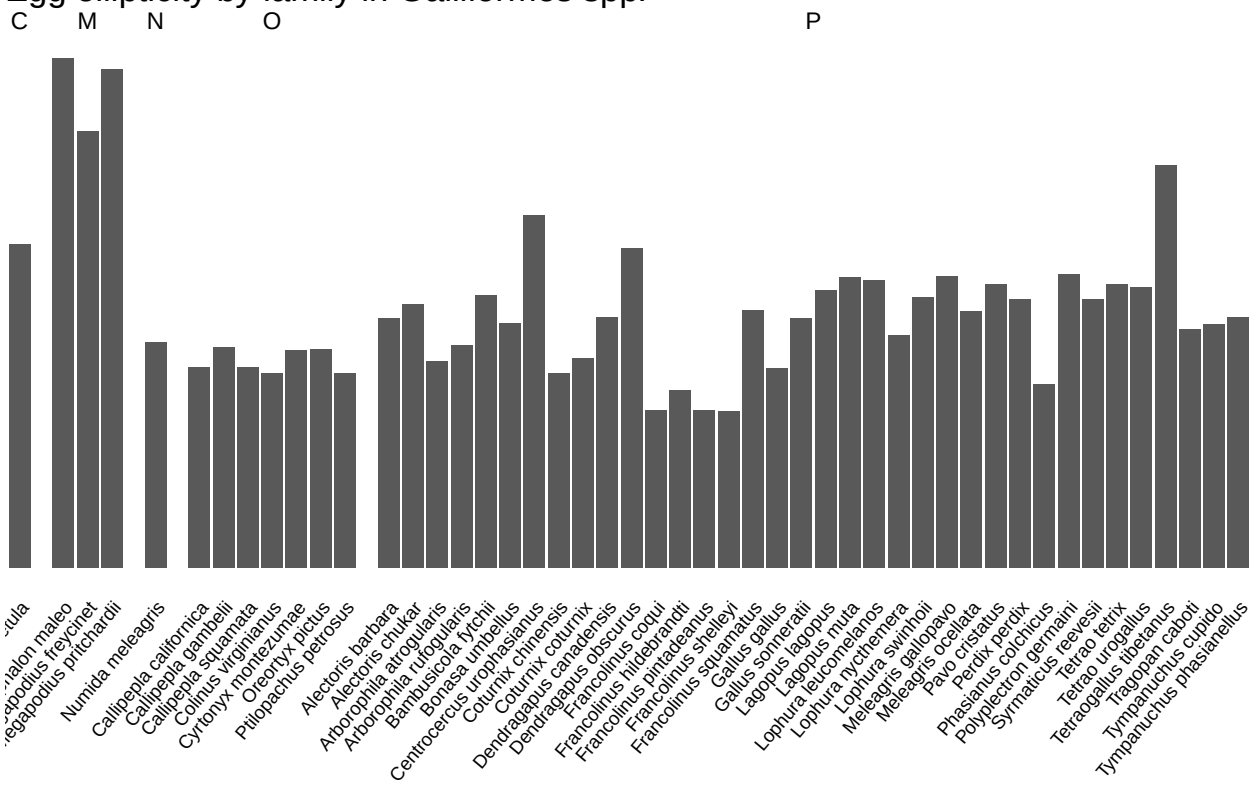
```
theme_dark()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_dark() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```




```
theme_void()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_void() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Others listed here: https://ggplot2.tidyverse.org/reference/ggtheme.html

Some themes are also available in external packages, such as `ggthemes`.

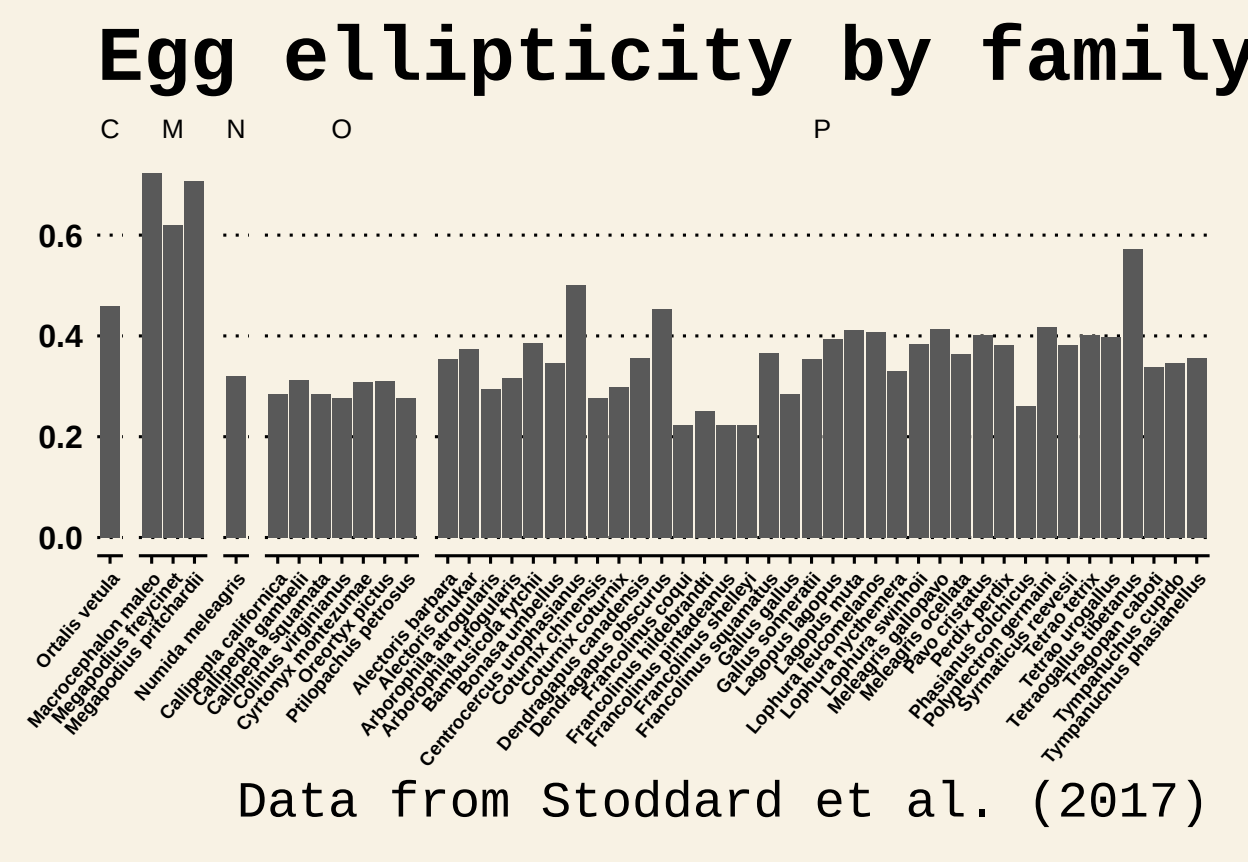
```
install.packages("ggthemes")
```

```
library(ggthemes)
```

```
## Warning: package 'ggthemes' was built under R version 3.5.3
```

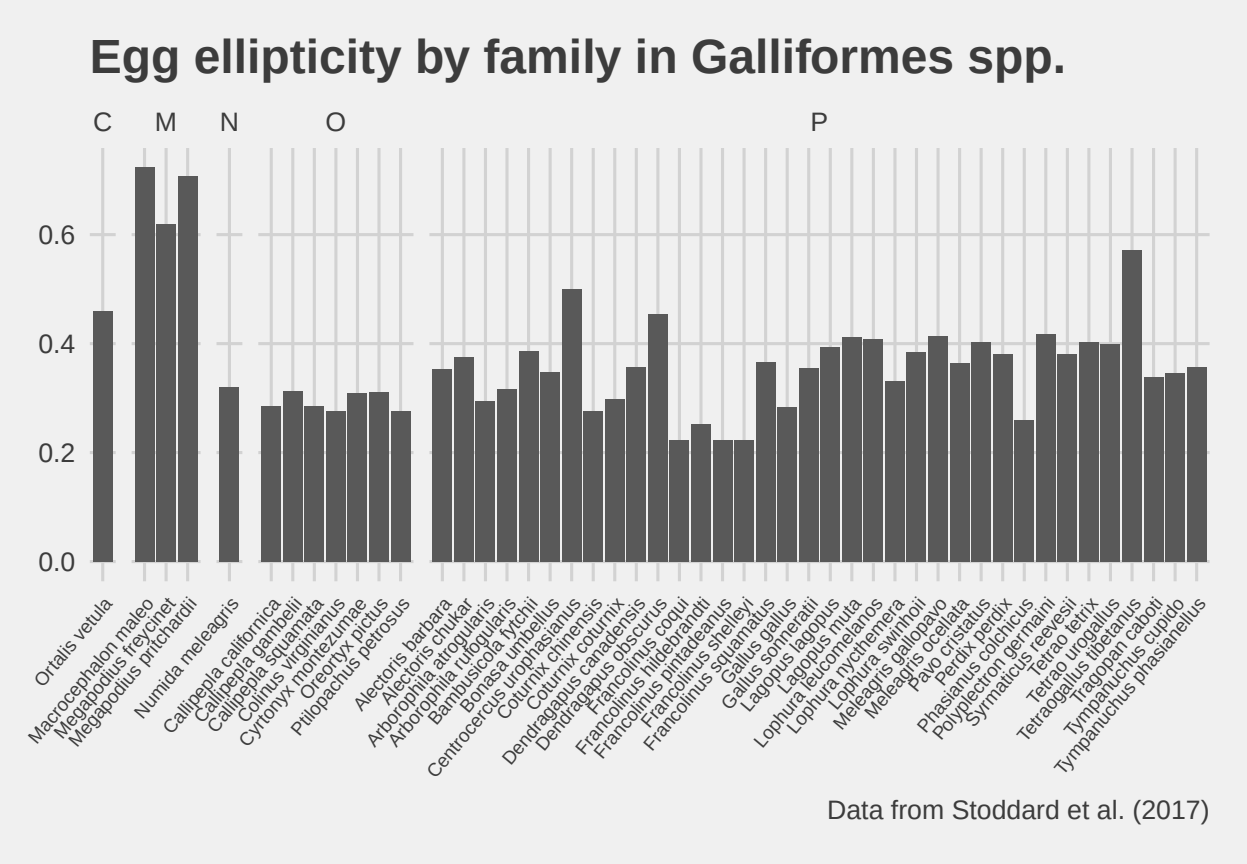
```
ggthemes::theme_wsj()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_wsj() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



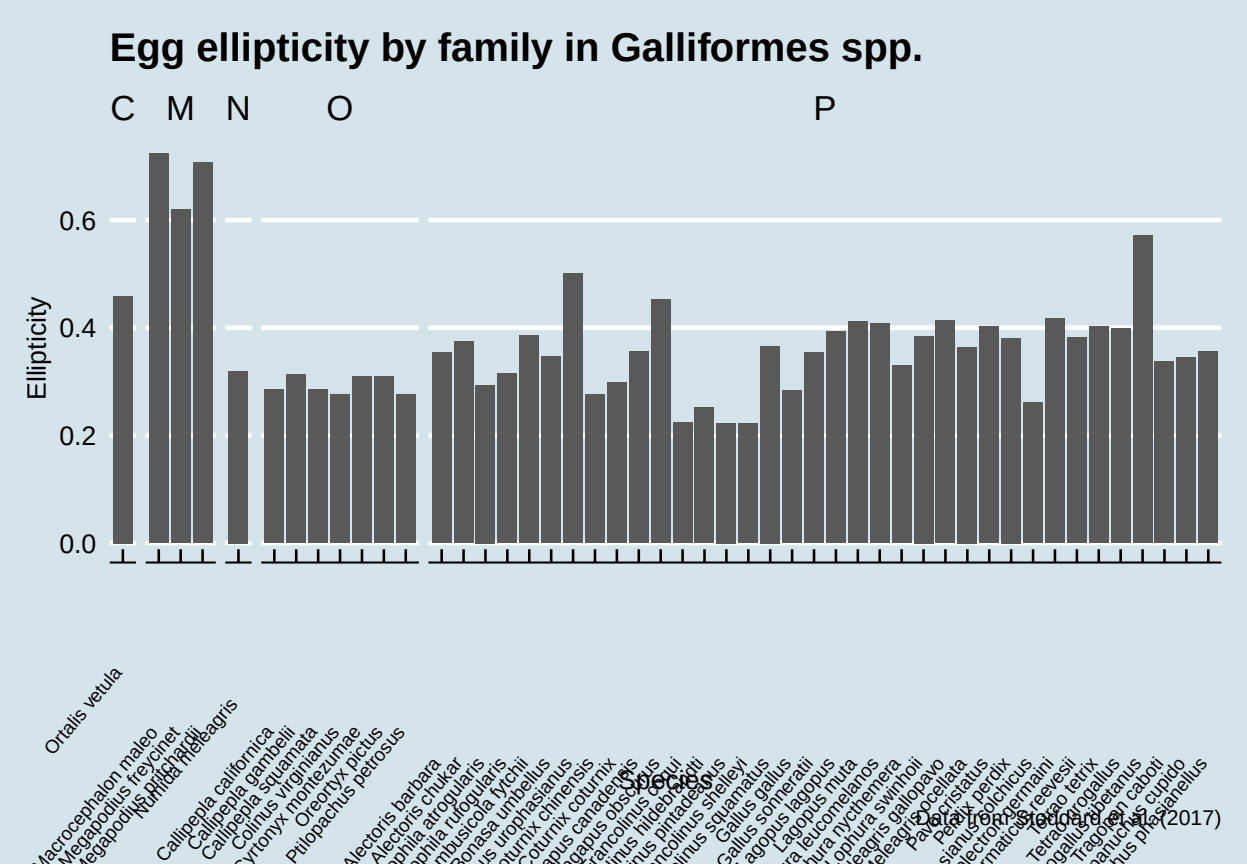

```
ggthemes::theme_fivethirtyeight()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_fivethirtyeight() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



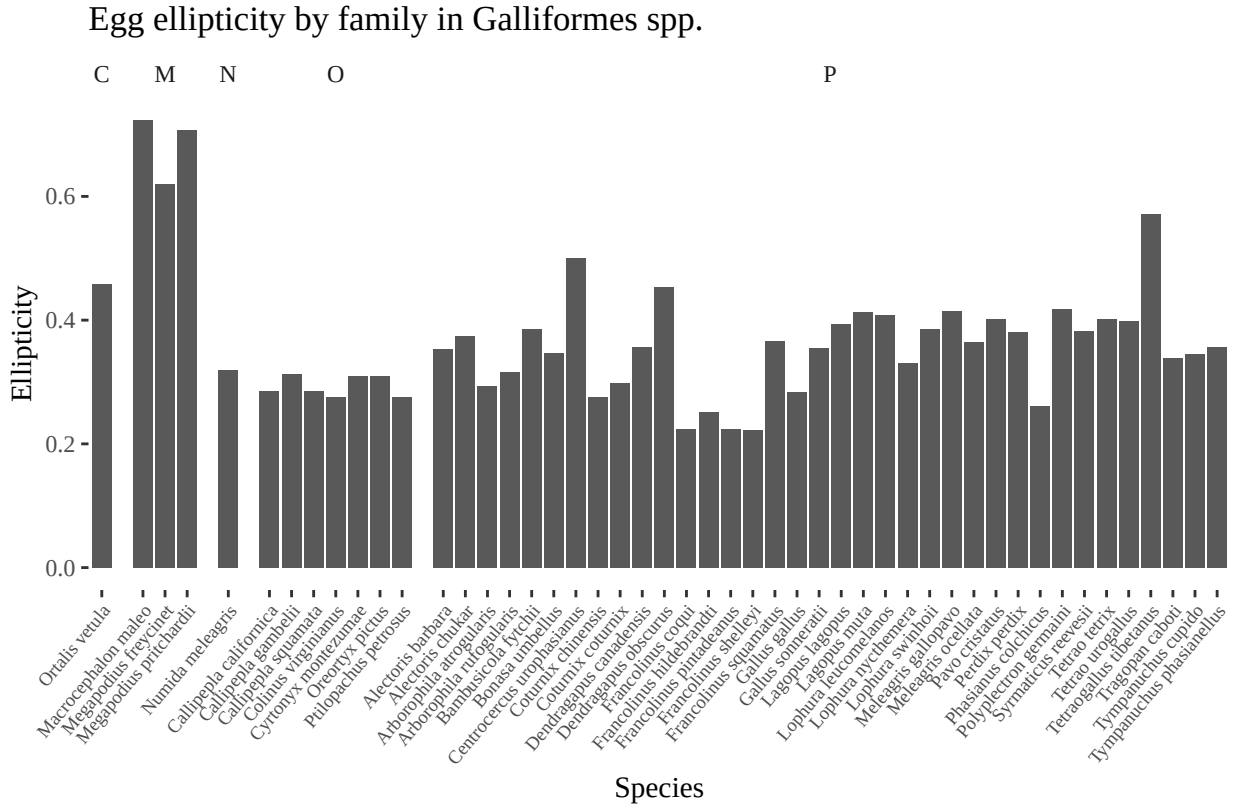

```
ggthemes::theme_economist()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_economist() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```




```
ggthemes::theme_tufte()
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_tufte() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7))
```



Data from Stoddard et al. (2017)

Or, for fun, `xkcd`... (like the webcomic)

```
install.packages("xkcd")
install.packages("extrafont")
```

```
library(xkcd)
```

```
## Loading required package: extrafont
```

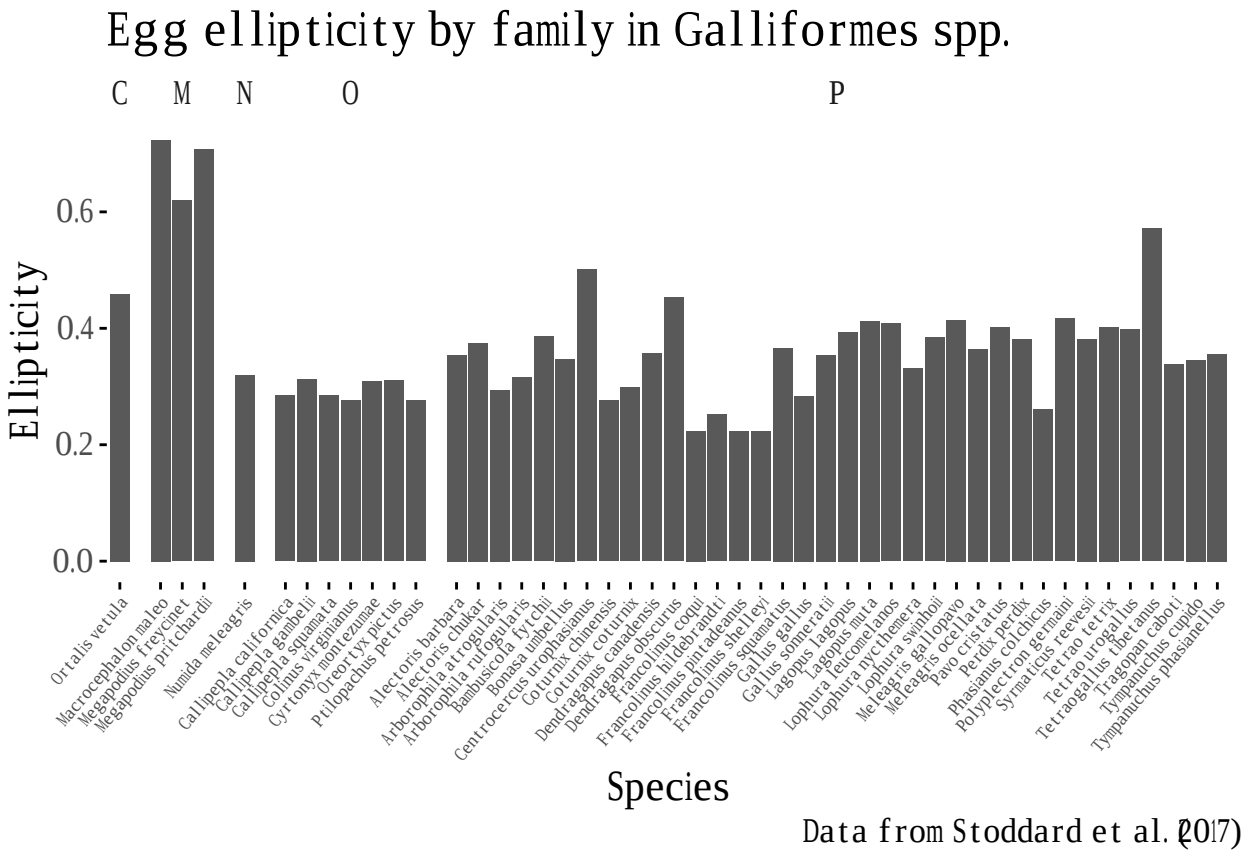
```
## Registering fonts with R
```

```
download.file("http://simonsoftware.se/other/xkcd.ttf",
              dest="./resources/xkcd.ttf", mode="wb")
font_import(paths="./resources/", pattern="[X/x]kcd", prompt=FALSE)
loadfonts(device="win")  # For Mac: loadfonts()

# Also have to put the font in your system fonts directory
```

```
ggplot(data=eggGalli, mapping=aes(x=Species, y=Ellipticity)) +
  facet_grid(cols=vars(Family), scales="free", space="free") +
  geom_bar(stat="identity") +
  labs(title="Egg ellipticity by family in Galliformes spp.",
       caption="Data from Stoddard et al. (2017)") +
  theme_xkcd() +
  theme(axis.text.x=element_text(angle=50, hjust=1, size=7),
        text=element_text(family="xkcd"))
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x28

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font width unknown for character 0x29

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font width unknown for character 0x28

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font width unknown for character 0x29
```



There are additional functions to make "fuzzy" rectangles and lines, but they're very particular about the input so you'd have to reformat your data:

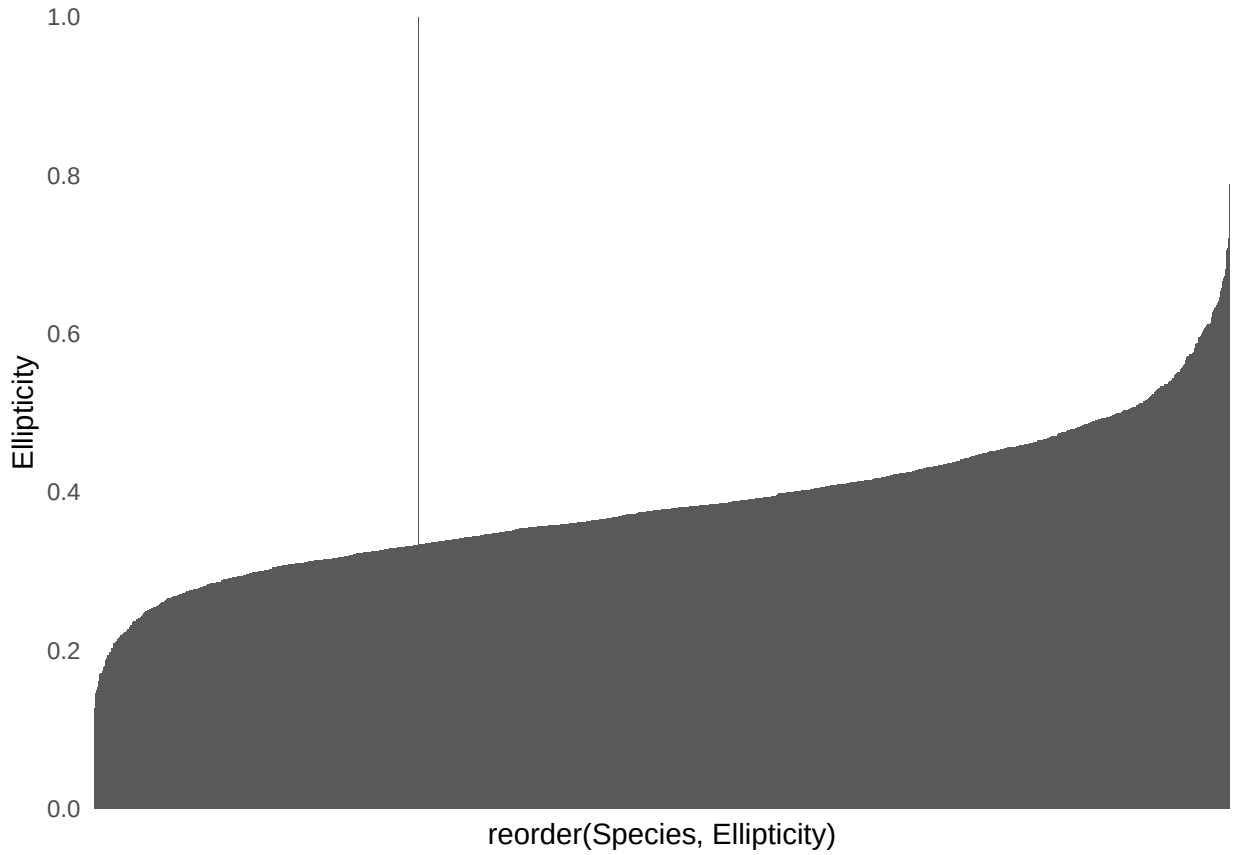
```
?xkcdrect
?xkcdline
```

Color scales

Arrange all species in the original data frame by their ellipticity values (and remove the grid and tick marks):

```
ggplot(data=egg, mapping=aes(x=reorder(Species, Ellipticity),
                             y=Ellipticity)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0)) +
  theme_minimal() +
  theme(axis.text.x=element_blank(),
        panel.grid.major=element_blank(),
        panel.grid.minor=element_blank())
```

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



We're seeing a warning about missing values.

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

```
##       Order         Family            MVZDatabase Species Asymmetry
## 377 EXTINCT Aepyornithidae          Aepyornis sp.    <NA>    0.0044
## 378 EXTINCT     Columbidae Ectopistes migratorius    <NA>    0.0872
## 379 EXTINCT  Dinornithidae           Dinornis sp.    <NA>    0.0050
## 380 EXTINCT       Rallidae        Porzana palmeri    <NA>    0.0523
##     Ellipticity AvgLength NumberOfImages NumberOfEggs
## 377      0.4499   23.8700              2            2
## 378      0.3391    3.9290              5            6
## 379      0.3318   14.4235              1            1
## 380      0.4494    2.9990              1            1
```

There are some extinct species without a `Species` value. To get rid of the warning, we can fix it for now by using the `MVZDatabase` values, which are good enough:

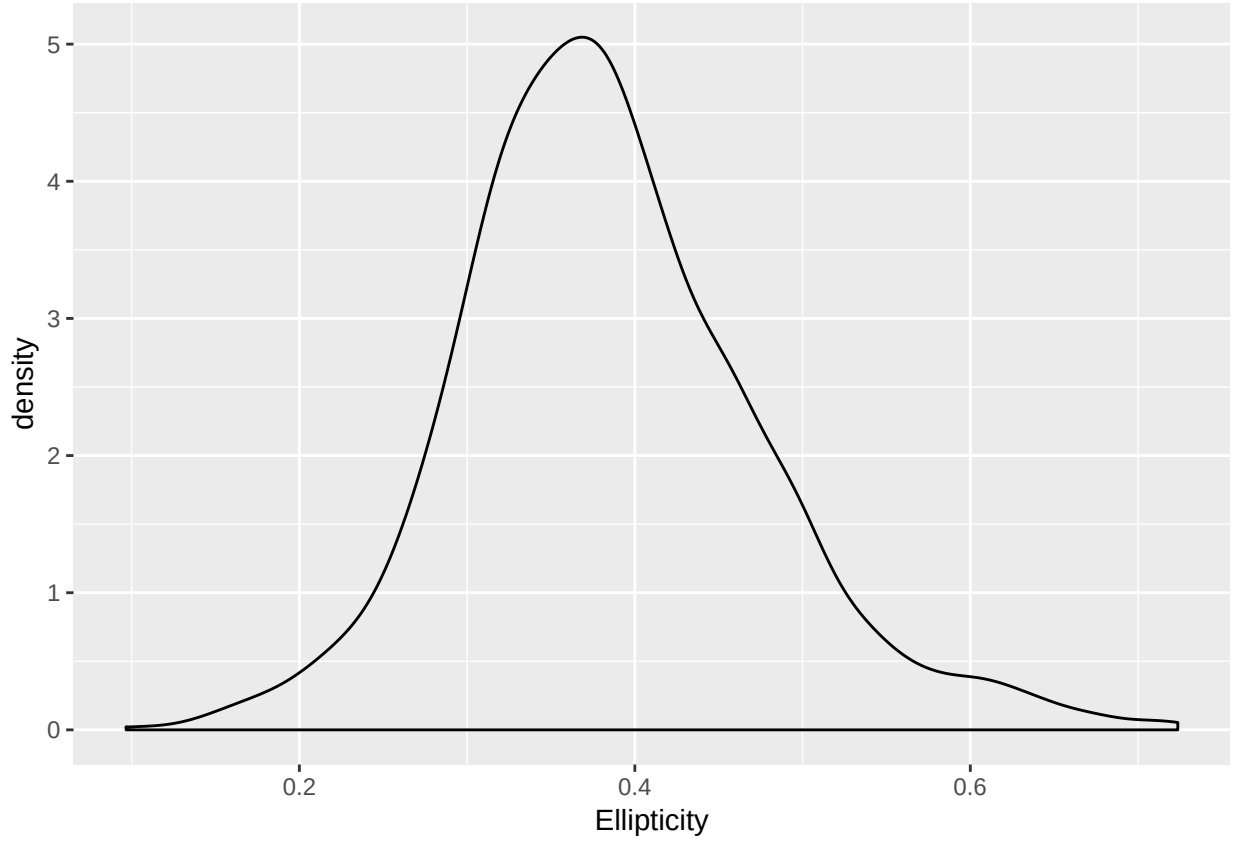
```
egg$Species[is.na(egg$Species)] = egg$MVZDatabase[is.na(egg$Species)]
```

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

```
## [1] Order          Family         MVZDatabase    Species
## [5] Asymmetry      Ellipticity    AvgLength      NumberOfImages
## [9] NumberOfEggs
## <0 rows> (or 0-length row.names)
```

We also saw a beautiful sigmoidal curve for the ellipticity values when we lined them all up. Why?

```
ggplot(data=egg, mapping=aes(x=Ellipticity)) +
  geom_density()
```



Because the distribution of values is really close to a "normal" Gaussian curve.

Now let's subset the data for Order Cuculiformes, the cuckoos.

```
eggCuckoo = egg[egg$Order == "CUCULIFORMES",]
eggCuckoo$Species
```

```
##  [1] "Centropus bengalensis"        "Centropus senegalensis"
##  [3] "Centropus superciliosus"      "Centropus viridis"
##  [5] "Coccyzus americanus"          "Coccyzus erythropthalmus"
##  [7] "Coccyzus minor"               "Crotophaga ani"
##  [9] "Crotophaga sulcirostris"      "Cuculus pallidus"
## [11] "Geococcyx californianus"      "Geococcyx velox"
## [13] "Guira guira"                  "Phaenicophaeus superciliosus"
```

We also want to take these `Species` values and split them to make a `Genus` variable too (since we have Order, Family, and Species, but not Genus).

We do this by using `strsplit()` and a fancy `apply()`-family function:

```
strsplit(eggCuckoo$Species, " ")
```

```
## [[1]]
## [1] "Centropus"   "bengalensis"
##
## [[2]]
## [1] "Centropus"    "senegalensis"
```

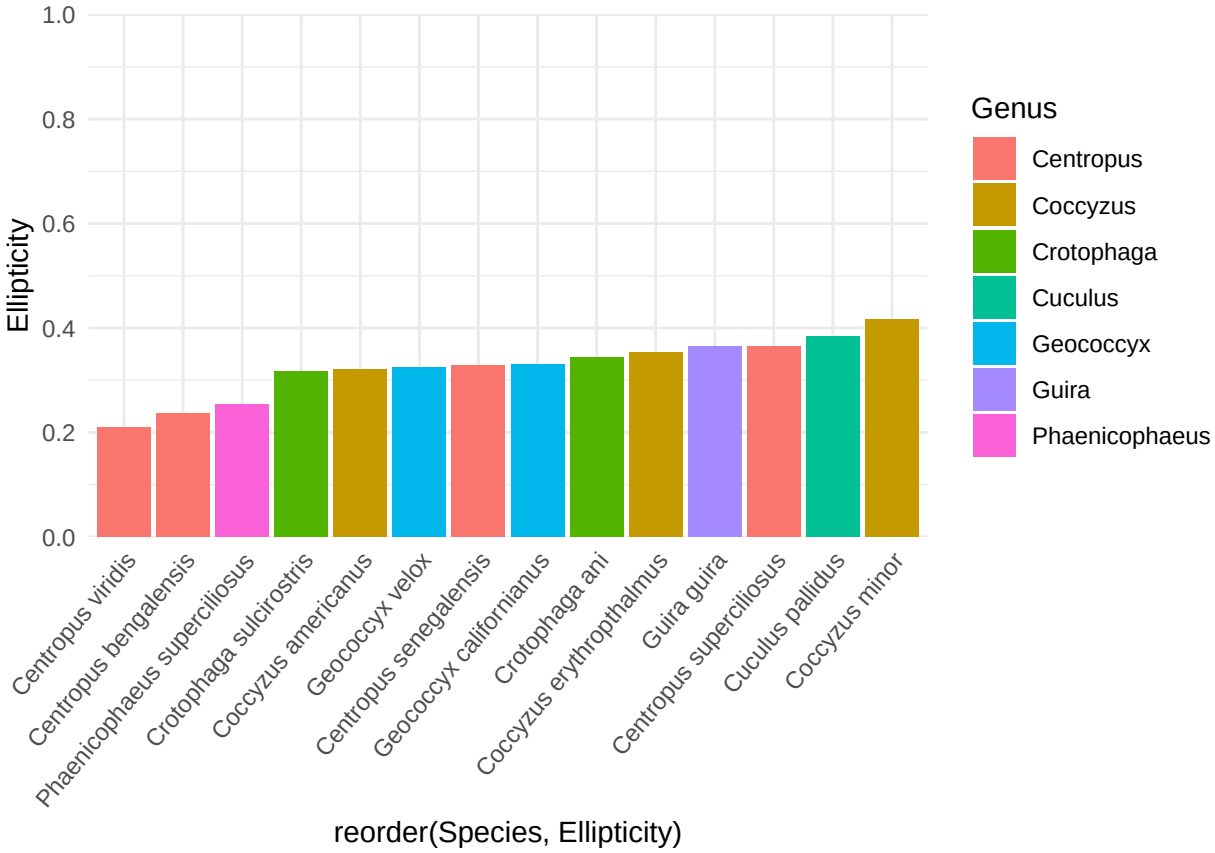
```
##
## [[3]]
## [1] "Centropus"     "superciliosus"
##
## [[4]]
## [1] "Centropus" "viridis"
##
## [[5]]
## [1] "Coccyzus"   "americanus"
##
## [[6]]
## [1] "Coccyzus"        "erythropthalmus"
##
## [[7]]
## [1] "Coccyzus" "minor"
##
## [[8]]
## [1] "Crotophaga" "ani"
##
## [[9]]
## [1] "Crotophaga"   "sulcirostris"
##
## [[10]]
## [1] "Cuculus"  "pallidus"
##
## [[11]]
## [1] "Geococcyx"     "californianus"
##
## [[12]]
## [1] "Geococcyx" "velox"
##
## [[13]]
## [1] "Guira" "guira"
##
## [[14]]
## [1] "Phaenicophaeus" "superciliosus"
```

```
eggCuckoo$Genus = sapply(strsplit(eggCuckoo$Species, " "), "[", 1)
unique(eggCuckoo$Genus)
```

```
## [1] "Centropus"      "Coccyzus"       "Crotophaga"     "Cuculus"
## [5] "Geococcyx"      "Guira"          "Phaenicophaeus"
```

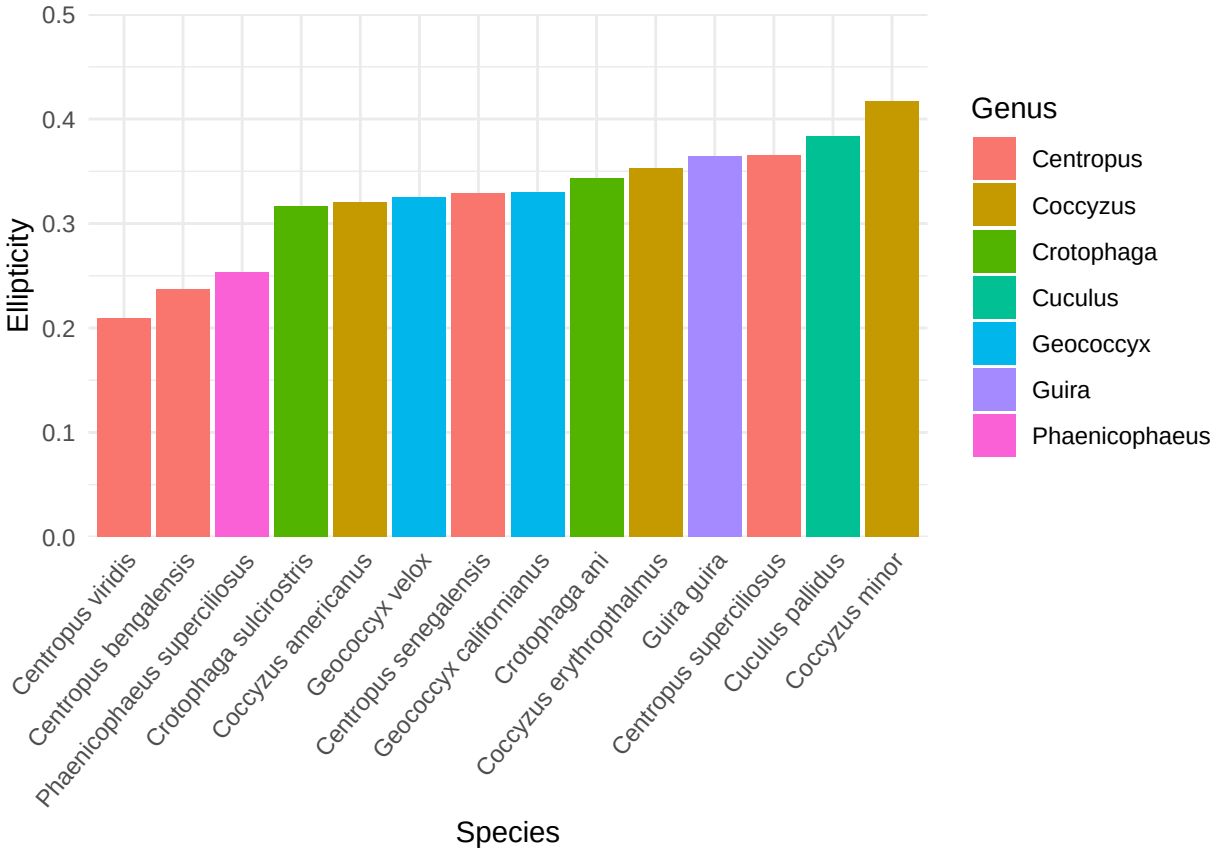
Now let's plot the cuckoo species, ordered by ellipticity values, with their fill colors according to genus.

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,1,0.2),
                     limits=c(0,1),
                     expand=c(0,0)) +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



Let's readjust the Y axis and fix the X axis label.

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



Now we want to customize the colors instead of using the default R palette.

```
length(unique(eggCuckoo$Genus))
```

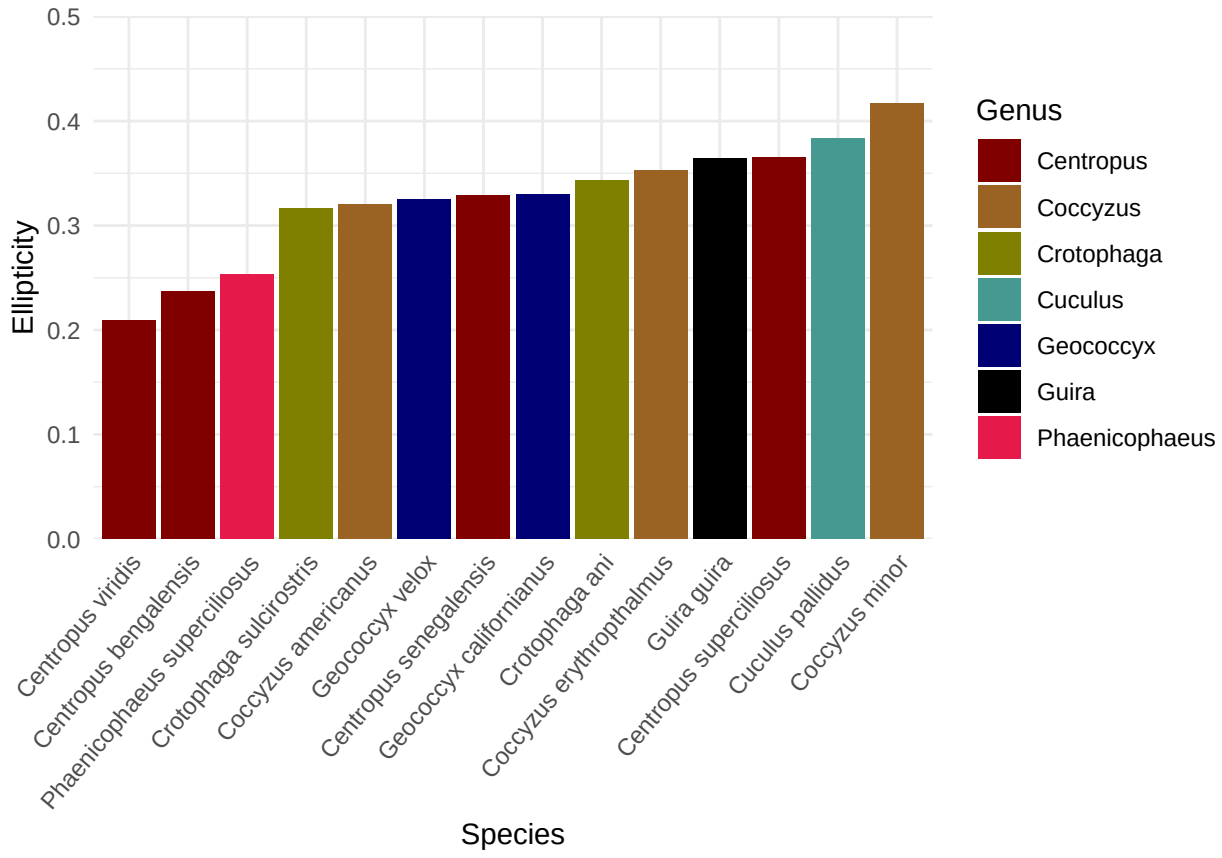
```
## [1] 7
```

We will need 7 different color values, and those values need to be for discrete or categorical data, not continuous (`Ellipticity`, for instance, would need a continuous scale).

Manual

The first option is to assign color values manually, either by name or by hexadecimal value.

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=c("#800000","#9A6324","#808000","#469990",
                             "#000075","#000000","#e6194B")) +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



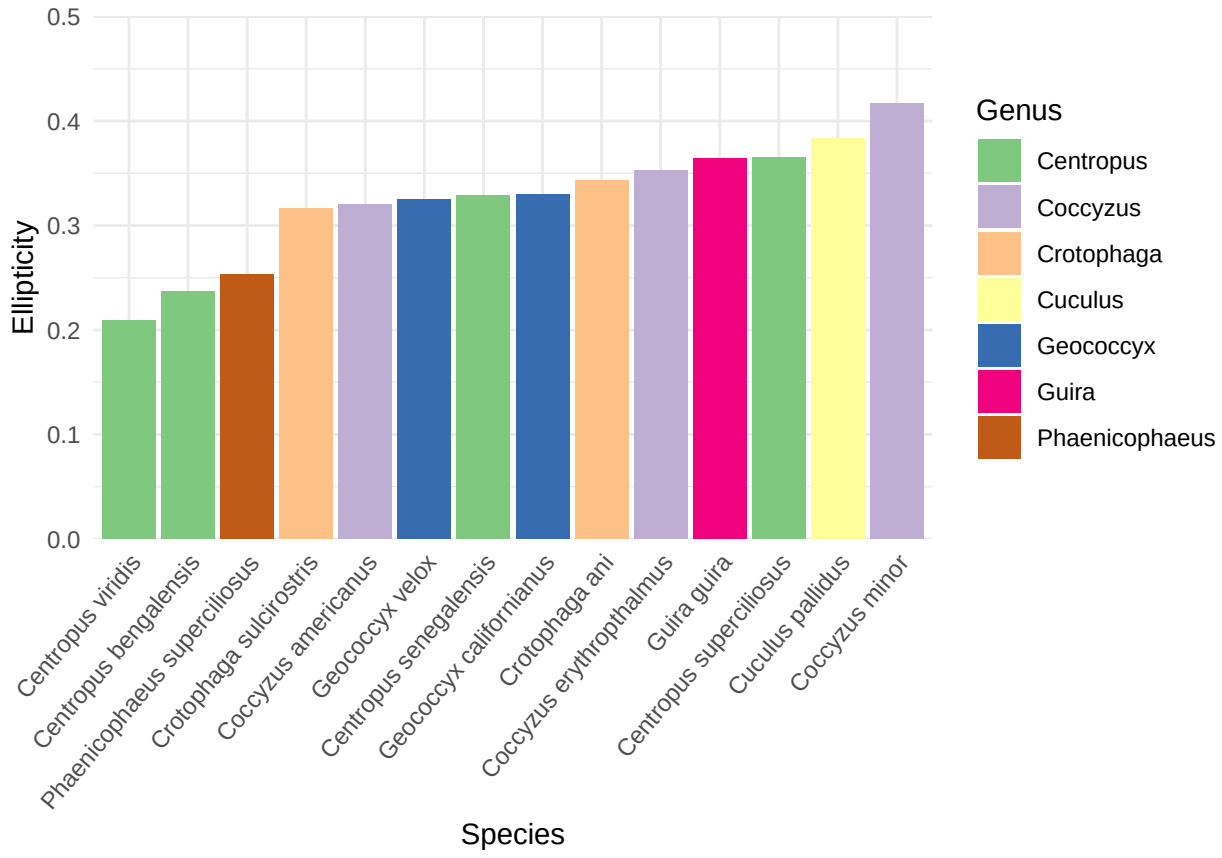
Source: Sasha Trubetskoy

ColorBrewer

Or we can use the `brewer` scales that are included in `ggplot2` and can be viewed online at: http://colorbrewer2.org/

Here, we just specify the type of scale (qualitative) and the palette to be used (here, "Accent"). Other choices are shown in `?scale_fill_brewer`.

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_brewer(type="qual", palette="Accent") +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



Source: ColorBrewer 2.0

Color-blind friendly

But none of the palettes we've used so far are color-blind friendly, and may not be discernable in print or grayscale (after being photocopied), either. There are better palettes out there that are.

`dichromat`

```
install.packages("dichromat")
```

```
library(dichromat)
```

```
## Warning: package 'dichromat' was built under R version 3.5.2
```

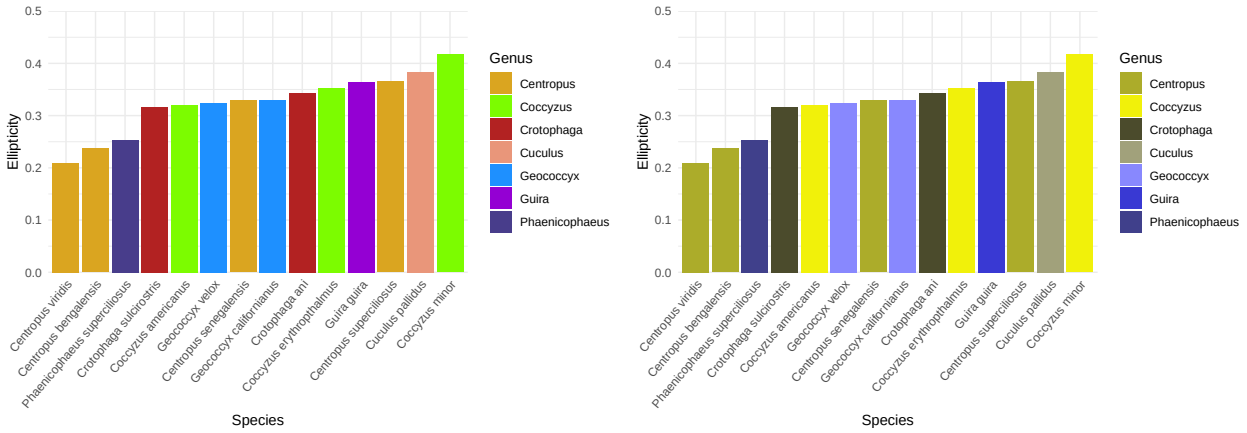
The `dichromat()` function will take a list of colors (again, named or hexadecimal values) and convert them into a comparable color-blind friendly palette for a specific type of color-blindness (here: protanopia, the kind I have). When in doubt, you can use deuteranopia, the most common form of dichromacy.

```
dichromat(c("goldenrod","lawngreen","firebrick",
            "darksalmon","dodgerblue","darkviolet",
            "darkslateblue"),
          type="protan")
```

```
## [1] "#ACAC2A" "#F1F10A" "#4B4B2C" "#A1A17B" "#8888FE" "#3939D3" "#40408B"
```

Compare the original colors (on the left) with the ones substituted by `dichromat()` (on the right):

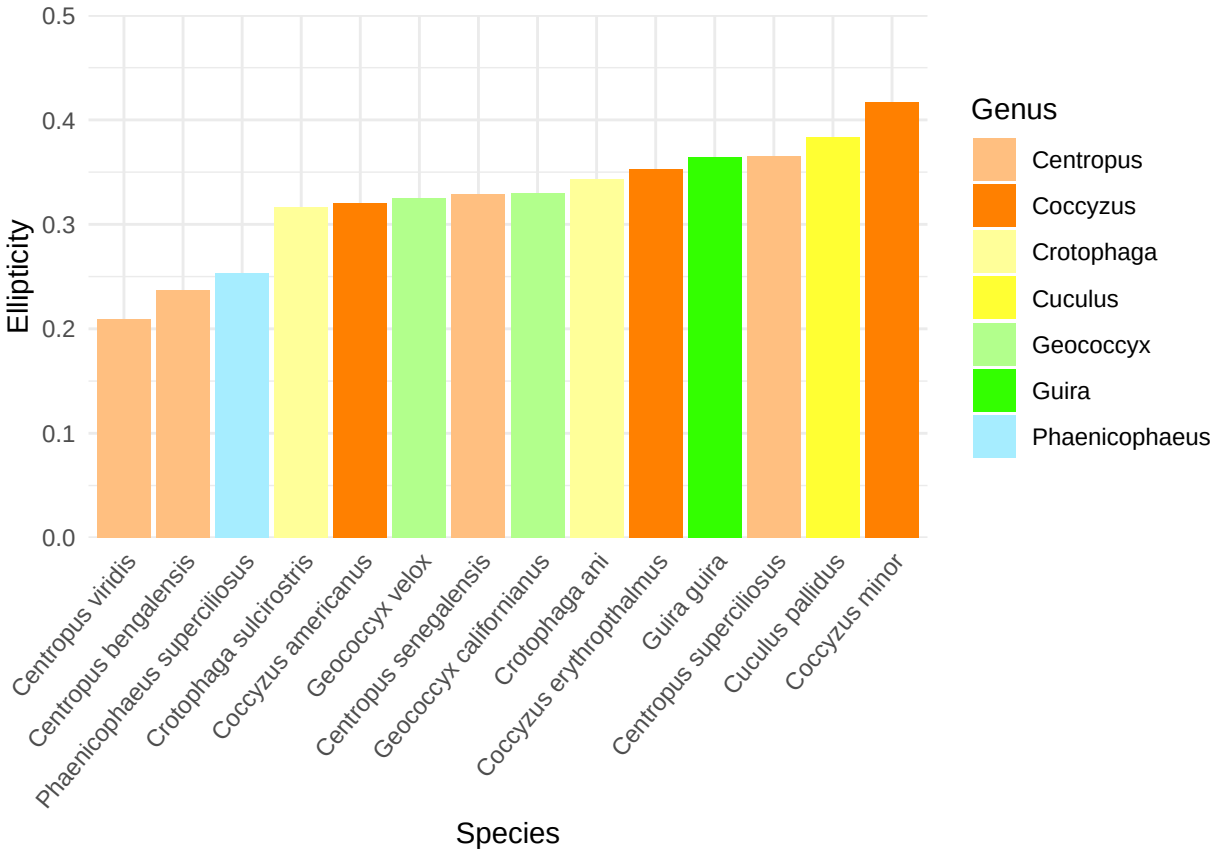
```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=c("goldenrod","lawngreen","firebrick",
                             "darksalmon","dodgerblue","darkviolet",
                             "darkslateblue")) +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))

ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=dichromat(c("goldenrod","lawngreen","firebrick",
                                       "darksalmon","dodgerblue","darkviolet",
                                       "darkslateblue"),
                                     "protan")) +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



`dichromat` also comes with some predefined color schemes in the `colorschemes` object. Here, we use `colorschemes$Categorical.12` because we're using categorical data, and subset the first 7 elements for our 7 levels:

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=dichromat::colorschemes$Categorical.12[1:7]) +
```

```
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



I find this one doesn't work great for me personally.

viridis

```
install.packages("viridis")
```

```
library(viridis)
```

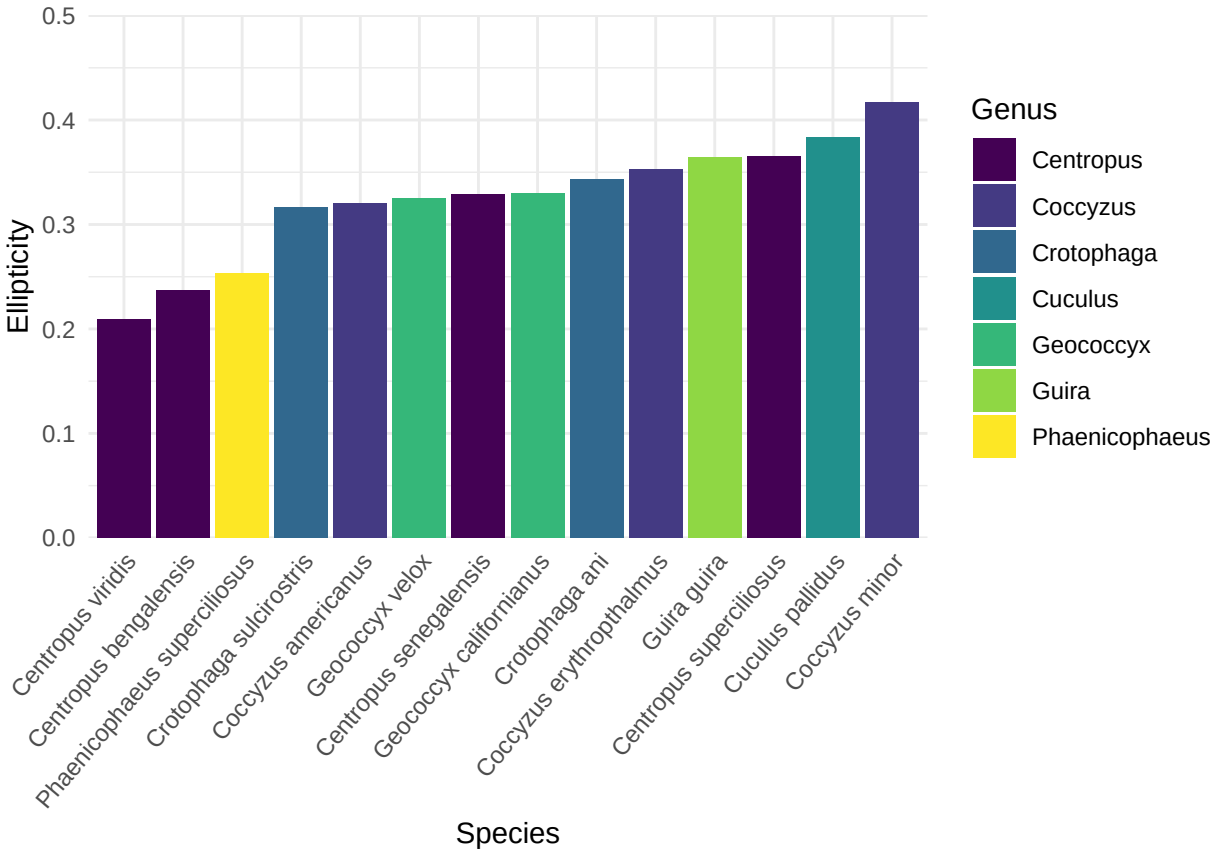
```
## Warning: package 'viridis' was built under R version 3.5.3
```

```
## Loading required package: viridisLite
```

The `viridis` package has a number of color-blind friendly palettes, listed under their own functions that end with `_d` (for discrete) or `_c` (for continuous). The default is `"viridis"`:

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_viridis_d() +
  labs(x="Species") +
```

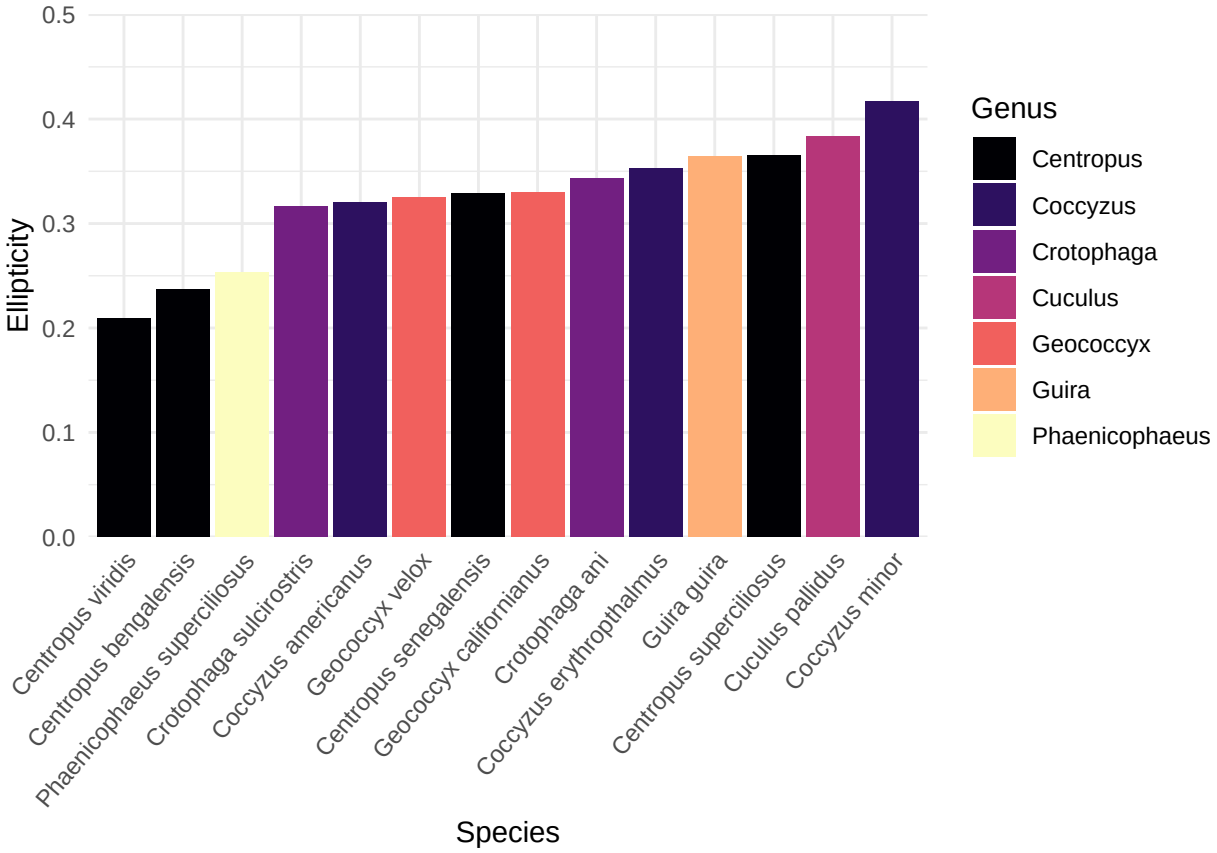
```
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



Still not great for me, with all the dark blues/purples.

But there are others, such as `"magma"`:

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_viridis_d(option="magma") +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



Again, not my preference, but it's there and could work.

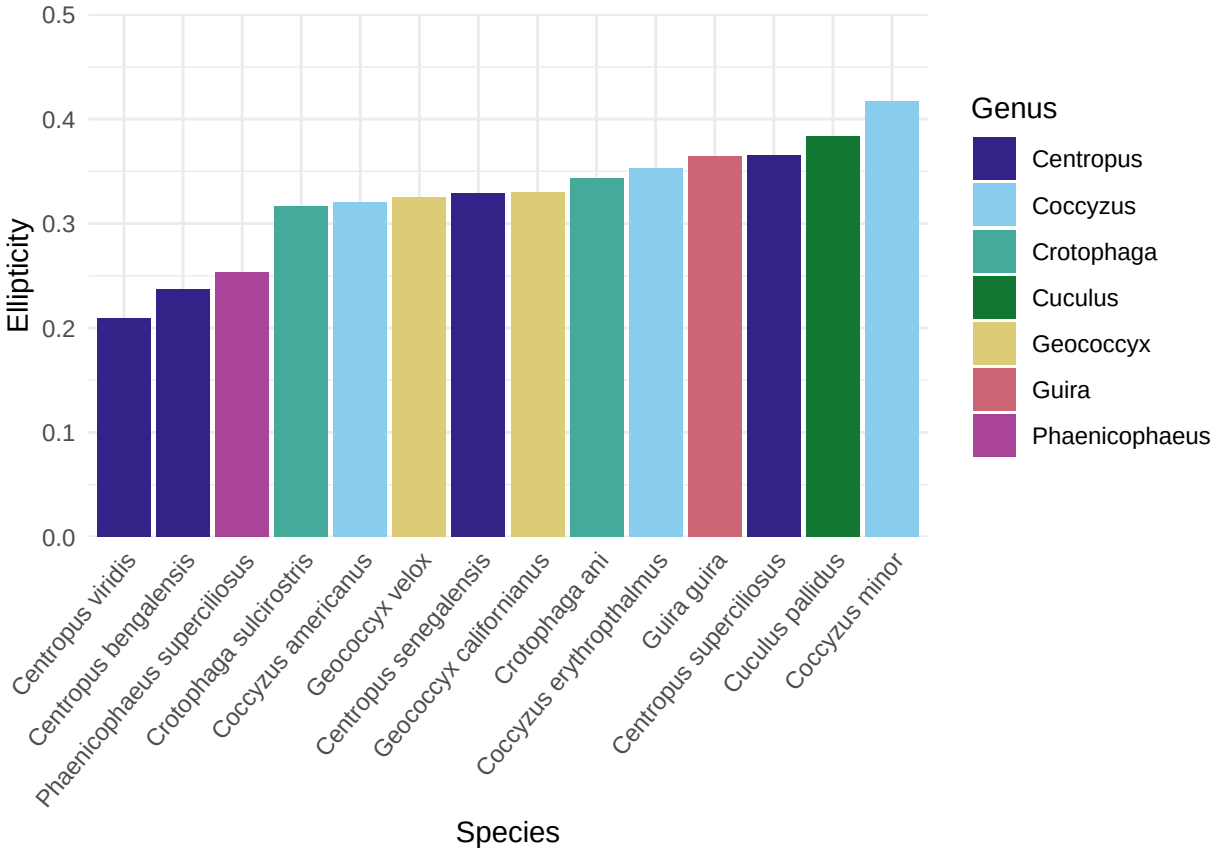
Paul Tol

Finally, my absolute favorite:

Tol, P. (2018). Colour schemes. SRON Netherlands Institute for Space Research. Available: https://personal.sron.nl/~pault/data/colourschemes.pdf

The older version of his qualitative color scheme (which I actually like better than the new one) is available in the `ggthemes` package, which we already have loaded. Weirdly, you have to specify the number of levels in parentheses *after* the function, rather than as an argument in the function.

```
ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                   y=Ellipticity,
                                   fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=ggthemes::ptol_pal()(7)) +
  labs(x="Species") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```



A variety of other color schemes are available in the document linked above which, in addition to being color-blind friendly, are also distinct in grayscale.

Saving plots

`ggplot()`

First, assign your plot to an object:

```
plotCuckoo = ggplot(data=eggCuckoo, mapping=aes(x=reorder(Species, Ellipticity),
                                y=Ellipticity,
                                fill=Genus)) +
  geom_bar(stat="identity") +
  scale_y_continuous(breaks=seq(0,0.5,0.1),
                     limits=c(0,0.5),
                     expand=c(0,0)) +
  scale_fill_manual(values=ggthemes::ptol_pal()(7)) +
  labs(title="Egg ellipticity in Cuculiformes spp.", x="Species",
       caption="Data from Stoddard et al. (2017)") +
  theme_minimal() +
  theme(axis.text.x=element_text(angle=50, hjust=1))
```

Then give it as an argument to the `ggsave()` function:

JPG

```
ggsave("./figs/plotCuckoo.jpg", plotCuckoo, width=8, height=6, units="in",
       dpi=300)
```

PDF

```
ggsave("./figs/plotCuckoo.pdf", plotCuckoo, width=8, height=6, units="in")
```

base

These functions let you save any kind of graphic, including `ggplots`. You have to use the graphic function first to open a "graphic device" (where the plot is sent), then run the code to produce the plot, then close the connection to the device with `dev.off()`.

JPG

```
jpeg("./figs/plotCuckoo2.jpg", width=8, height=6, units="in",
     res=300)
plotCuckoo
dev.off()
```

PDF

```
pdf("./figs/plotCuckoo2.pdf", width=8, height=6)
plotCuckoo
dev.off()
```

(pdf / Rmd)