Visualization of data

Peter Ralph

13 October – Advanced Biological Statistics

Visualization

Goals

- pattern discovery
- efficient summary of information
- visual/spatial analogy for quantitative patterns

Goals

- pattern discovery
- efficient summary of information
- visual/spatial analogy for quantitative patterns

aim to *maximize information and minimize ink*

paraphrased from Edward Tufte

Considerations

- Is the visual analogy appropriate for the *type* of data?

  counts? quantities? multivariate? relationships?

- Are important *comparisons* clear?

  between groups? differences? time trend?

- Are *units* easily interpretable?

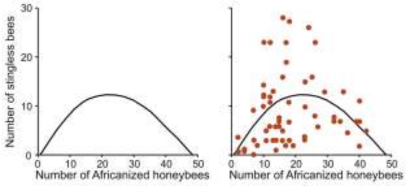
  meters? dollars? percent? relative change? is it isometric?

Principles of effective display

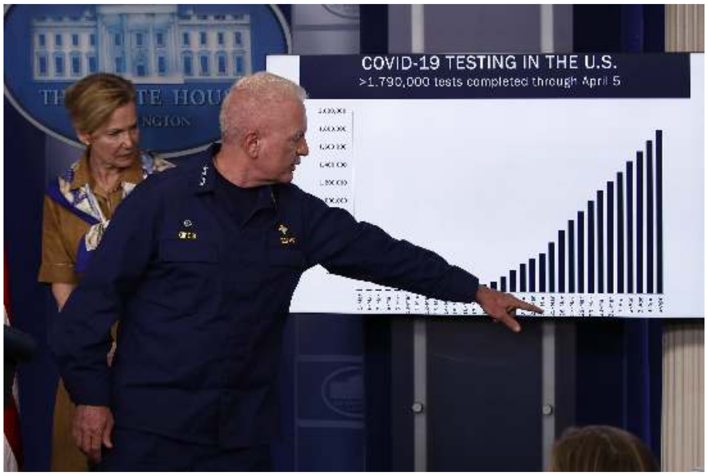
- Show the data
- Encourage the eye to compare differences
- Represent magnitudes honestly and accurately
- Draw graphical elements clearly, minimizing clutter
- Make displays easy to interpret

> *Above all else show the data.*

Tufte 1983



Think about what you want to communicate


COVID-19 TESTING IN THE U.S.
>1,790,000 tests completed through April 5

AP Images: Alex Brandon

Case study:

Distributions of litter sizes by Order, and Family, in the PanTHERIA dataset:

```
source("../Datasets/PanTHERIA/read_pantheria.R")
pantheria <- read_pantheria("../Datasets/PanTHERIA")
# look at most common orders
order_nums <- sort(table(pantheria$Order))
```

```
##
##   Microbiotheria    Tubulidentata       Dermoptera Notoryctemorphia      Proboscidea
##                1                1                2                2                3
##   Perissodactyla       Scandentia        Cingulata  Peramelemorphia  Erinaceomorpha
##               17               20               21               21               24
##        Carnivora         Primates     Soricomorpha       Chiroptera         Rodentia
##              286              376              428             1116             2277
```

note the pipe

```
px <- (pantheria %>% filter(Order %in% big_orders)
       %>% filter(!is.na(LitterSize))
       %>% select(Order, Family, Genus, Species, LitterSize))
for (xn in c("Order", "Family", "Genus")) px[[xn]] <- factor(px[[xn]])
summary(px)
```

```
##            Order                    Family              Genus        Species
##  Artiodactyla :178   Muridae         : 242   Microtus    :  38   Length:2172
##  Carnivora    :209   Cricetidae      : 239   Myotis      :  38   Class :character
##  Chiroptera   :465   Sciuridae       : 158   Crocidura   :  36   Mode  :character
##  Diprotodontia:112   Vespertilionidae: 135   Peromyscus  :  32
##  Primates     :209   Bovidae         : 110   Sorex       :  32
##  Rodentia     :883   Phyllostomidae  : 106   Spermophilus:  31
##  Soricomorpha :116   (Other)         :1182   (Other)     :1965
```

Raw numbers

```
px$LitterSize
```

```
##    [1]  0.98  4.50  3.74  5.72  4.98  1.22  1.00  1.22  1.01  1.02  1.02  1.02  1.02
##   [35]  2.00  2.31  2.00  2.00  1.93  1.04  1.94  3.00  2.19  1.94  2.36  2.62  3.00
##   [69]  1.00  7.29  6.12  1.00  1.00  1.00  1.00  1.00  2.37  4.60  3.33  3.96  1.01
##  [103]  1.02  3.09  1.09  1.00  1.00  0.99  1.00  0.99  2.91  3.68  4.00  3.45  5.28
##  [137]  3.00  2.99  3.89  3.00  2.44  4.04  3.99  3.49  1.22  1.50  4.86  1.00  1.00
##  [171]  1.02  1.00  1.00  1.00  1.00  1.34  1.41  1.38  1.11  1.00  1.01  1.01  1.00
##  [205]  4.99  1.18  4.34  3.88  6.24  4.99  4.99  3.46  4.66  3.67  1.00  0.99  1.00
##  [239]  2.50  1.00  1.00  1.00  1.50  1.22  1.22  1.00  1.00  1.39  4.14  3.00  1.73
##  [273]  1.01  4.37  0.97  2.50  1.87  5.83  3.42  1.94  1.00  0.99  0.98  0.99  0.98
##  [307]  0.98  0.99  1.00  0.98  1.94  0.98  3.60  2.95  2.47  1.50  2.14  1.34  3.45
##  [341]  2.00  0.98  2.26  2.04  0.98  0.98  0.99  0.99  2.00  2.31  0.99  0.98  1.00
##  [375]  1.00  4.50  3.00  0.97  1.94  0.98  1.00  0.99  1.00  2.34  1.94  1.94  1.94
##  [409]  2.00  2.00  3.00  4.34  2.50  1.01  1.00  4.30  2.15  2.30  0.98  0.99  1.00
##  [443]  3.36  2.91  3.11  4.77  2.39  2.37  1.94  2.14  2.95  3.78  2.59  2.67  2.68
##  [477]  4.18  3.24  3.00  1.94  0.99  1.22  1.08  2.14  1.00  1.22  0.96  3.74  7.50
##  [511]  1.89  2.53  3.14  3.09  1.68  4.00  5.94  3.40  3.92  4.06  2.74  4.00  4.50
##  [545]  4.99  2.84  1.54  1.94  1.90  0.96  1.01  1.50  1.00  0.98  1.00  1.10  2.46
##  [579]  0.99  1.60  1.50  4.00  4.86  3.16  4.65  3.30  1.29  3.00  3.13  1.12  4.00
##  [613]  3.88  3.24  2.41  4.00  1.00  0.98  2.91  2.00  2.91  3.49  1.00  1.73  1.41
##  [647]  0.98  1.00  0.99  1.00  0.98  1.00  0.98  1.00  1.39  3.88  3.12  1.00  2.10
##  [681]  1.96  2.14  2.97  2.68  0.99  5.11  1.00  1.02  1.00  4.00  4.18  3.00  4.61
```

stem-and-leaf “plot”

```
stem(px$LitterSize)
```

```
##
##   The decimal point is at the |
##
##    0 | 99
##    1 | 00000000000000000000000000000000000000000000000000000000000000000000+1036
##    2 | 00000000000000000000000000000000000000000000000000000000000000000000+266
##    3 | 00000000000000000000000000000000000000000000000000000000000000000000+232
##    4 | 00000000000000000000000000000000000000000001111111112222222222222223+105
##    5 | 00000000000000000000000000000111111222222222222233333333333333444444+35
##    6 | 00000000000111122223333444455555555666777788899999
##    7 | 001111235555567888899
##    8 | 000011155557789
##    9 | 0000249
##   10 | 0
##   11 | 23
```

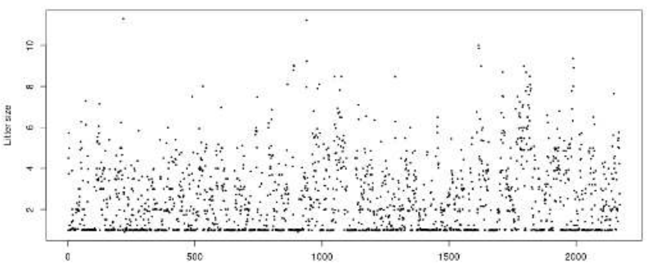
five(-ish) number summary

```
summary(px$LitterSize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.900   1.000   1.935   2.426   3.450  11.300
```

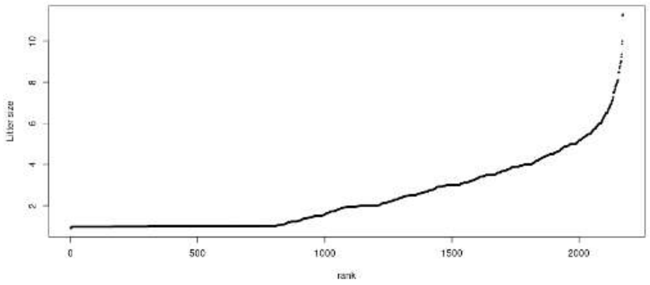
Points

```
plot(px$LitterSize, xlab='', ylab='Litter size', pch=20, cex=0.5)
```



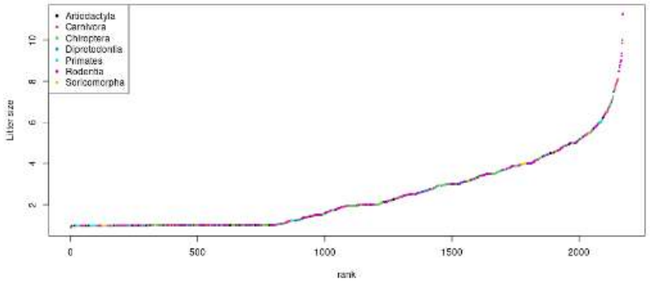
Points, sorted

```
plot(sort(px$LitterSize), xlab='rank', ylab='Litter size', pch=20, cex=0.5)
```



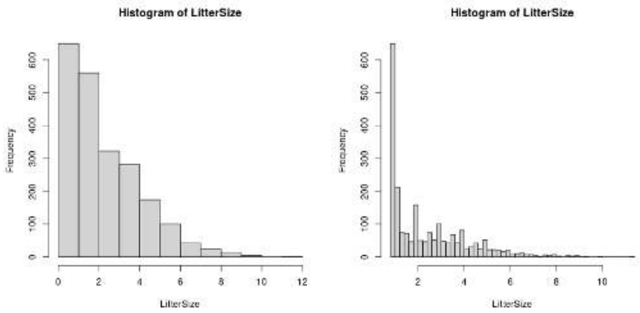
Points, sorted and colored

```
plot(sort(px$LitterSize), col=px$Order, xlab='rank', ylab='Litter size', pch=20,
        cex=0.5)
legend("topleft", pch=20, col=1:nlevels(px$Order), legend=levels(px$Order))
```



Histogram

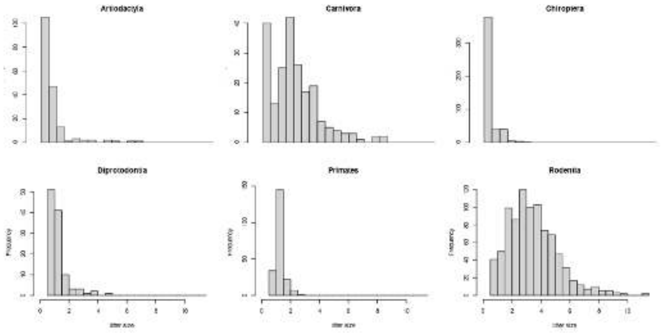
```
layout(t(1:2))
with(px, hist(LitterSize))
with(px, hist(LitterSize, breaks=40))
```

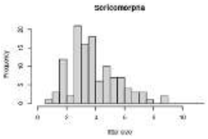


Histograms

```
layout(matrix(1:6, ncol=3, byrow=TRUE), heights=c(1, 1.2))
opar <- par(mar=c(1, 3, 1, 1)+.1)
xh <- hist(px$LitterSize, plot=FALSE, breaks=30)
for (k in 1:nlevels(px$Order)) {
    ord <- levels(px$Order)[k]
    if (k == 4) par(opar)
    with(subset(px, Order == ord),
        hist(LitterSize, xlim=c(0, max(px$LitterSize)),
             breaks=xh$breaks, main=ord,
             xaxt=if (k > 3) 's' else 'n',
             xlab=if (k > 3) 'litter size' else '') )
}
```

Histograms



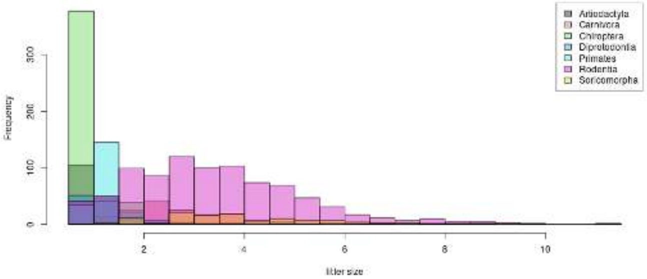


Overlaid histograms

```r
overlay_hist <- function (x, f, breaks=30, ...) {
    xh <- hist(x, breaks=breaks, plot=FALSE)
    ymax <- do.call(max, with(px, lapply(tapply(LitterSize, Order, hist, plot=FALSE),
        "[[", "counts")))
    for (k in 1:nlevels(f)) {
        hist(x[f==levels(f)[k]], breaks=xh$breaks, ...,
             add=(k>1), col=adjustcolor(k, 0.4), ylim=c(0, ymax))
    }
    legend("topright", fill=adjustcolor(1:nlevels(f), 0.4),
          legend=levels(f))
}
```

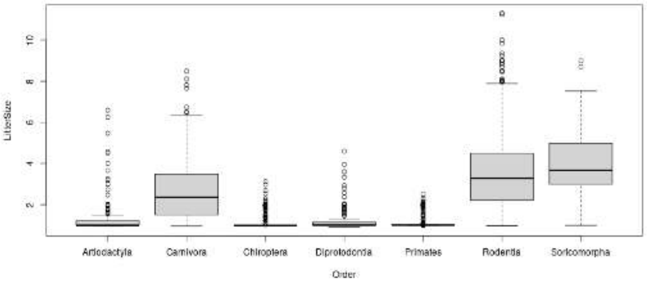
Overlaid histograms

```r
with(px, overlay_hist(LitterSize, Order, xlab='litter size', main=''))
```



boxplots

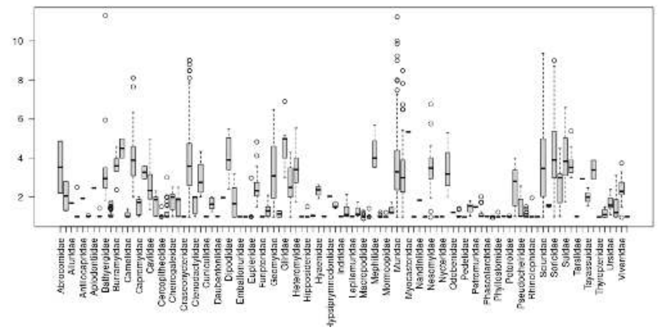
```r
with(px, boxplot(LitterSize ~ Order))
```



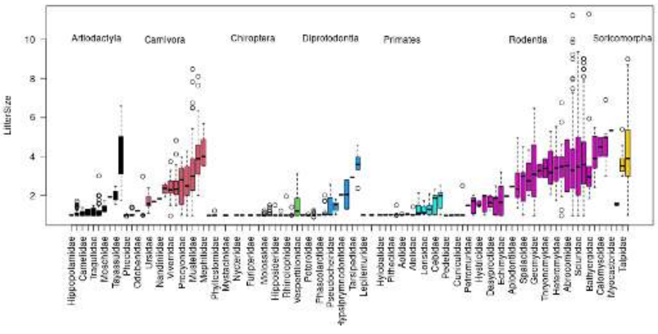
introduced by Mary Eleanor Spear

Many boxes

```r
par(mar=c(9, 3, 1, 1) + 0.1)
with(px, boxplot(LitterSize ~ Family, las=2, xlab=''))
```



```r
par(mar=c(9, 4, 1, 1) + 0.1)
famsize <- aggregate(LitterSize ~ Order + Family, data=px, mean)
famorder <- rank(with(famsize, LitterSize + 100 * as.numeric(Order)))
with(px, boxplot(LitterSize ~ Family, las=2, xlab='',
                 col=as.numeric(famsize$Order),
                 at=famorder))
text(x=tapply(famorder, famsize$Order, mean),
     y=10, label=levels(famsize$Order))
```



Your turn

Challenge: visualize `LitterSize` by `TeatNumber`, using a boxplot.

The Grammar of Graphics

or, “`gg`”

- introduced by Leland Wilkinson
- adopted by Hadley Wickham in the `ggplot` library
- thinks of plots as *objects*
- see this chapter of *R for Data Science*

Ingredients of a visualization

Ingredients of a visualization

- data

Ingredients of a visualization

- data
- coordinate axes

Ingredients of a visualization

- data
- coordinate axes
- a geometric representation of numbers

Ingredients of a visualization

- data
- coordinate axes
- a geometric representation of numbers
- a mapping from (summaries of) variables to properties of the geoms

Ingredients of a visualization

- data
- coordinate axes
- a geometric representation of numbers
- a mapping from (summaries of) variables to properties of the geoms
- maybe more plots

basic template

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

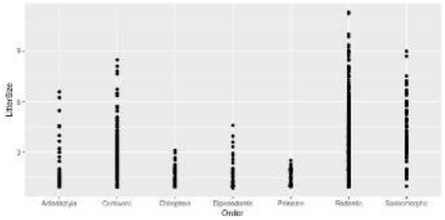
more options

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(
     mapping = aes(<MAPPINGS>),
     stat = <STAT>,
     position = <POSITION>
  ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

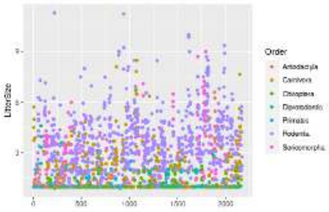
Reference: the ggplot2 book.

Points

```
ggplot(px, mapping=aes(x=Order, y=LitterSize)) + geom_point()
```

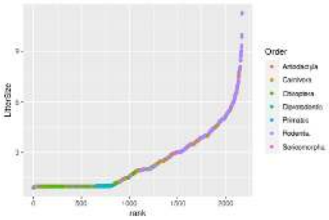


```
(ggplot(px,
        mapping=aes(x=1:nrow(px),
                    y=LitterSize,
                    col=Order))
 + xlab("")
 + geom_point())
```



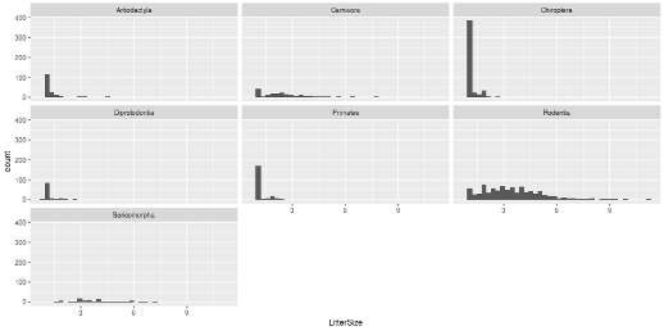
```
(ggplot(px,
        mapping=aes(x=rank(LitterSize,

        ties.method='first'),
                    y=LitterSize,
                    col=Order))
 + xlab("rank")
 + geom_point())
```



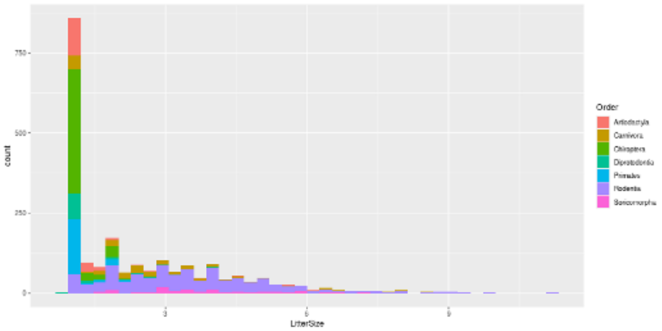
Histogram

```
ggplot(px, mapping=aes(LitterSize)) + geom_histogram(bins=40) + facet_wrap(~ Order)
```



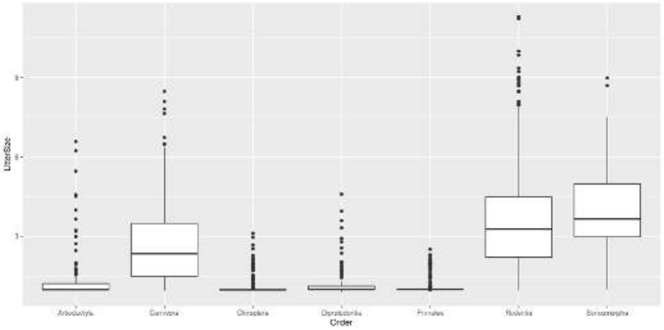
Histogram, stacked

```
ggplot(px, mapping=aes(LitterSize, fill=Order)) + geom_histogram(bins=40)
```



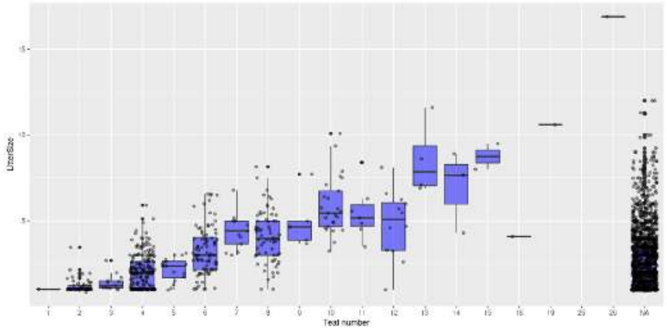
Boxplot

```
ggplot(px, mapping=aes(y=LitterSize, x=Order)) + geom_boxplot()
```



Your turn, again

Challenge: make this plot.



The cheatsheet might be helpful.

// reveal.js plugins