Lecture 3: Data Visualization

BIOF 339

September 26, 2016

Data Visualization in R

One of R's strengths is data visualization.

R can create static as well as interactive graphs with a rich set of user-contributed packages

- Static graph systems
  - Base R graphics
  - `lattice` (Sarkar, et al)
  - `ggplot2` (Wickham)
- Dynamic graphs
  - `rCharts`
  - `leaflet`

Data visualization in R

I'm making the decision to use `ggplot2` for my graphics

- Makes pretty good formatting choices out of the box
- Is declarative (tell it what you want) without getting caught up in minutae
- Strongly leverages data frames (good practice)
- There are good templates if you want to change the look

Introduction to ggplot2

Introduction to ggplot2

If you haven't installed it yet:

```
install.packages('ggplot2',
                 repos="http://watson.nci.nih.gov/cran_mirror/")
```

then

```
library(ggplot2)
```

Introduction to ggplot2

The `ggplot2` package is a very flexible and (to me) intuitive way of visualizing data. It is based on the concept of layering elements on a canvas.

You need:

- A `data.frame` object
- *Aesthetic mappings* (aes) to say what data is used for what purpose in the viz
    - x- and y-direction
    - shapes, colors, lines
- A *geometry object* (geom) to say what to draw
    - You can "layer" geoms on each other to build plots

Introduction to ggplot2

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg)) + geom_point()
```

- A `data.frame` object: mtcars
- Aesthetic mapping: x-axis: wt y-axis: mpg
- Geometry:
    - geom_point: draw points

Introduction to ggplot2

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg)) + geom_point()+ geom_smooth()
```

- A `data.frame` object: mtcars
- Aesthetic mapping: x-axis: wt y-axis: mpg
- Geometry:
  - geom_point: draw points
  - geom_smooth: Add a layer which draws a best-fitting line

Introduction to ggplot2

We will use the two data sets introduced last week:

```
data_spine <- read.csv('http://www.araastat.com/BIOF339_PracticalR/
                       Lectures/lecture2_data/Dataset_spine.csv')

data_brca <- read.csv('http://www.araastat.com/BIOF339_PracticalR/
                      Lectures/lecture2_data/
                      clinical_data_breast_cancer_modified.csv')
```

Plotting one variable

Histograms

```
ggplot(data_brca, aes(x = Age.at.Initial.Pathologic.Diagnosis)) +
  geom_histogram()
```

```
#   `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Histograms

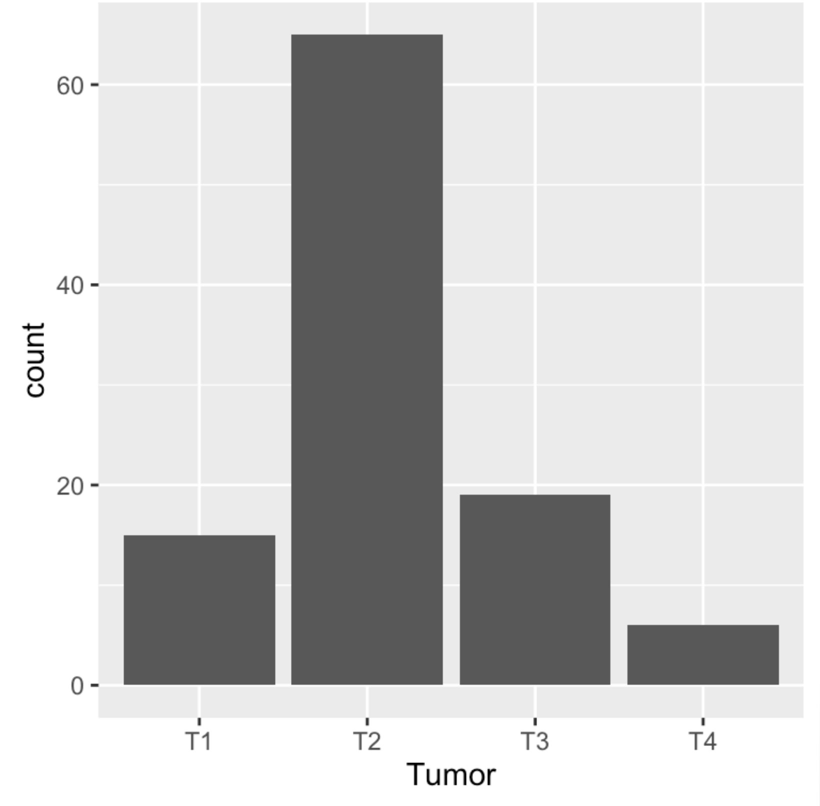
```
ggplot(data_brca, aes(x = Age.at.Initial.Pathologic.Diagnosis)) +
  geom_histogram(binwidth=4)
```

Density plot

```
ggplot(data_brca, aes(x = Age.at.Initial.Pathologic.Diagnosis)) +
  geom_density()
```

Bar plot

```
ggplot(data_brca, aes(x = Tumor))+geom_bar()
```



Two continuous variables

Scatter plots

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope)) +
  geom_point()
```

Scatter plot with a smooth line

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope))+
  geom_point() +
  geom_smooth()
```

Scatter plot with a smooth straight line

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope)) +
  geom_point()+
  geom_smooth(method='lm')
```

Line plot (for time series)

```
library(forecast)
d <- data.frame(x = 1:length(gas), y = gas) # Australian monthly gas production
ggplot(d, aes(x, y)) + geom_line()
```

Continuous variable with discrete variable

Boxplots

```
ggplot(data_spine, aes(x = Class.attribute, y = Sacral.slope))+
  geom_boxplot()
```

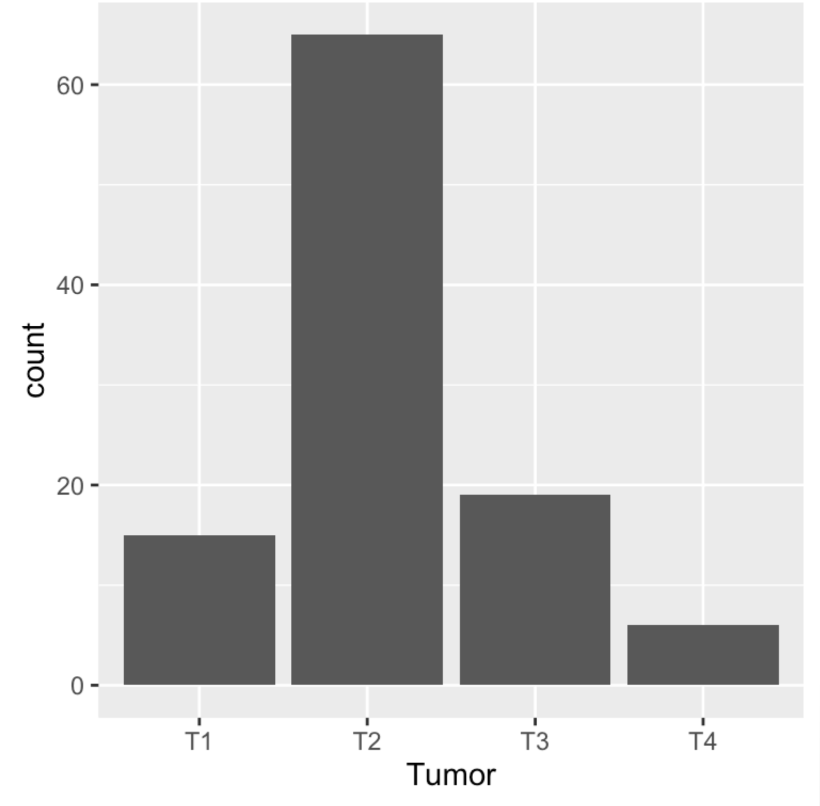
Violin plots

```
ggplot(data_spine, aes(x = Class.attribute, y = Sacral.slope)) +
  geom_violin()
```

Flipping axes

Vertical bars

```
ggplot(data_brca, aes(x = Tumor))+geom_bar()
```



Horizontal bars

```
ggplot(data_brca, aes(x = Tumor))+geom_bar()+
  coord_flip()
```

Resources

Online resources

- The ggplot website has many resources to help create visualizations
- There are a lot of blogs showing many capabilities of ggplot2
- StackOverflow is the place for Q & A.

Group-wise descriptives and visualizations

Grouping

- It is common to look at statistics within subgroups of the data
- The idea is to see if secondary variables affect your primary outcome or relationship

Introducing the `dplyr` package

`dplyr` is the most lucid package for manipulating and analyzing data organized in a data frame.

- It has a `group_by` function which creates a *grouped data frame*

```
library(dplyr)
grouped_data_spine = group_by( data_spine, Class.attribute)
```

Note that you have to group using a discrete valued variable (factor, character, integer)

Grouped summaries

```
summarize(grouped_data_spine,
          mean(Pelvic.incidence),
          sd(Pelvic.incidence),
          min(Pelvic.incidence),
          max(Pelvic.incidence))
```

Class.attribute	mean(Pelvic.incidence)	sd(Pelvic.incidence)	min(Pelvic.incidence)	max(Pelvic.in
Abnormal	64.69256	17.66213	26.14792	12
Normal	51.68524	12.36816	30.74194	8

Grouped summaries

```
summarize(grouped_data_spine,
          Mean = mean(Pelvic.incidence),
          SD = sd(Pelvic.incidence),
          Min = min(Pelvic.incidence),
          Max = max(Pelvic.incidence))
```

Class.attribute	Mean	SD	Min	Max
Abnormal	64.69256	17.66213	26.14792	129.83404
Normal	51.68524	12.36816	30.74194	89.83468

Grouped summaries

```
summarize_all(grouped_data_spine, mean)
```

```
#   # A tibble: 2 × 13
#     Class.attribute Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle
#              <fctr>            <dbl>       <dbl>                 <dbl>
#   1        Abnormal         64.69256    19.79111              55.92537
#   2          Normal         51.68524    12.82141              43.54260
#   # ... with 9 more variables: Sacral.slope <dbl>, Pelvic.radius <dbl>,
#   #   Degree.spondylolisthesis <dbl>, Pelvic.slope <dbl>, Direct.tilt <dbl>,
#   #   Thoracic.slope <dbl>, Cervical.tilt <dbl>, Sacrum.angle <dbl>,
#   #   Scoliosis.slope <dbl>
```

Grouped visualization

Density plot

```
ggplot(data_spine, aes(x = Sacral.slope, group = Class.attribute, color=Class.att
  geom_density()
```

Scatter plot

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope,
                       group = Class.attribute, color = Class.attribute))+
  geom_point()
```

Scatter plot with lines

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope,
                       group = Class.attribute, color=Class.attribute))+
  geom_point()+
  geom_smooth(method='lm')
```

Scatter plot with lines

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope))+
  geom_point()+
  geom_smooth(aes(color = Class.attribute), method='lm')
```

Facetting

Facetting

Facetted graphs are a panel of graphs, each of which corresponds to a particular subgroup of the data.

Facetted scatter plot

```
ggplot(data_spine, aes(x = Lumbar.lordosis.angle, y = Sacral.slope))+
  geom_point()+
  facet_wrap( ~ Class.attribute, nrow=1)
```

Manhattan plot

Manhattan plot

```
library(qqman)
data(gwasResults)
head(gwasResults)
```

```
#     SNP CHR BP         P
#   1 rs1   1  1 0.9148060
#   2 rs2   1  2 0.9370754
#   3 rs3   1  3 0.2861395
#   4 rs4   1  4 0.8304476
#   5 rs5   1  5 0.6417455
#   6 rs6   1  6 0.5190959
```

```
gwasResults = transform(gwasResults, x_position = 1:nrow(gwasResults))
```

Manhattan plot

```
ggplot(gwasResults, aes(x = x_position, y = -log(P, base=10)))+
  geom_point(size=0.2)
```

Manhattan plot

```
ggplot(gwasResults, aes(x = x_position, y = -log(P, base=10),
                        group=CHR, color=CHR))+
  geom_point(size=0.2)
```

Manhattan plot

```
ggplot(gwasResults, aes(x = x_position, y = -log(P, base=10),
                        group=factor(CHR), color=factor(CHR)))+
  geom_point(size=0.2)
```

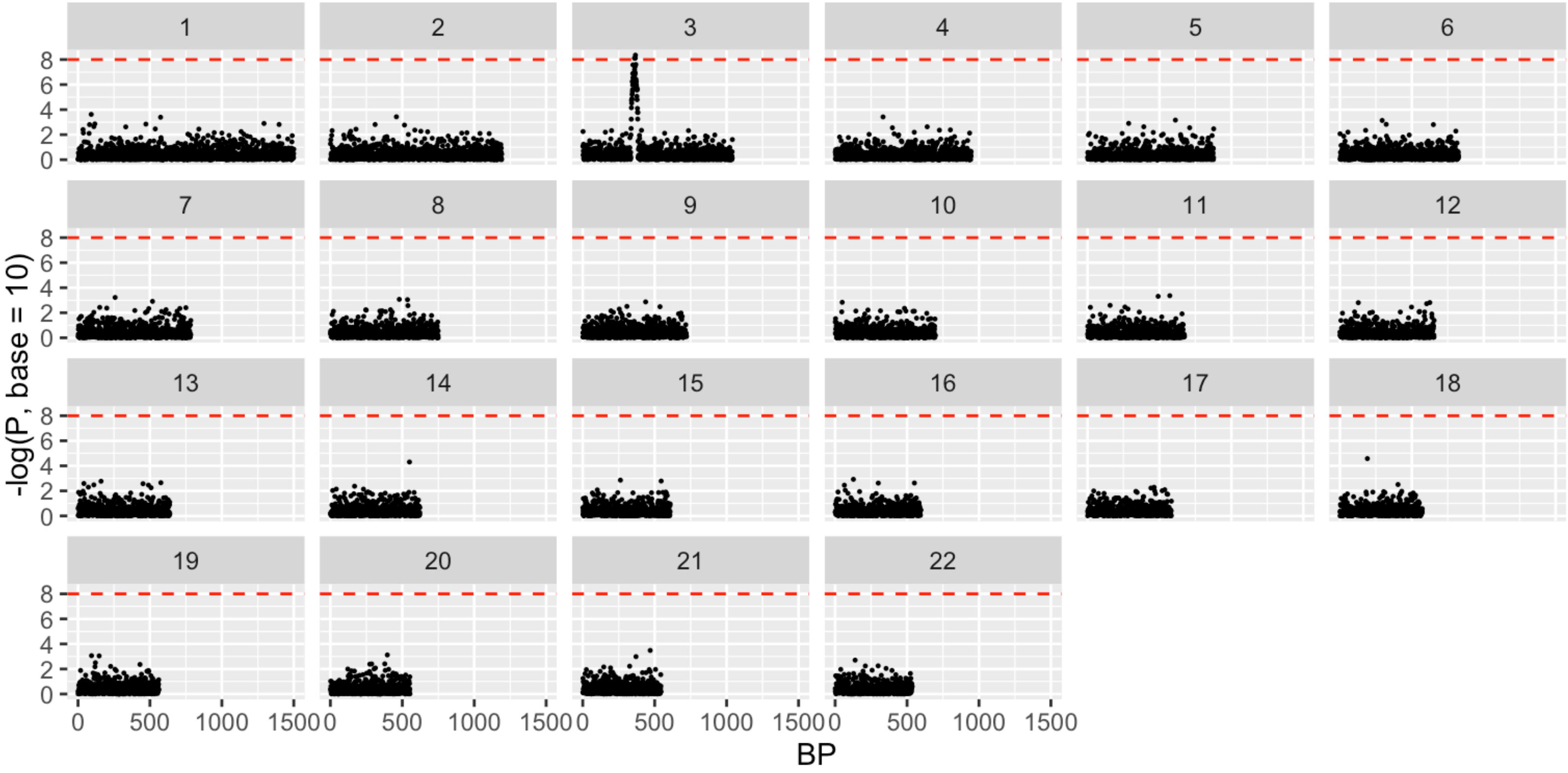
Manhattan plot

```
ggplot(gwasResults, aes(x = x_position, y = -log(P, base=10),
                        group=factor(CHR), color=factor(CHR)))+
  geom_point(size=0.2)+
  geom_hline(yintercept = 8, color='red', linetype=2)
```

Manhattan plot, exploded

```
ggplot(gwasResults, aes(x = BP, y = -log(P, base=10)))+
  geom_point(size=0.2)+
  facet_wrap(~ CHR, nrow=4)+
  geom_hline(yintercept = 8, color='red', linetype=2)
```

Manhattan plot, exploded


-log(P, base = 10)
BP