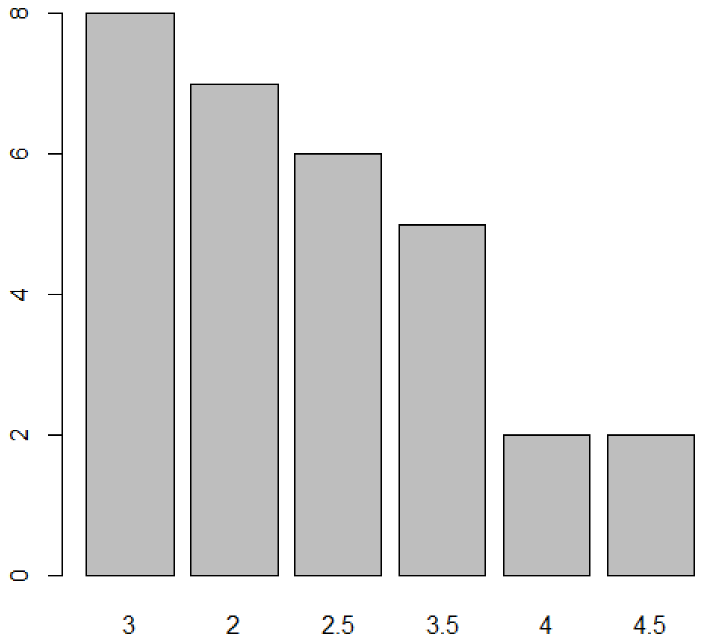
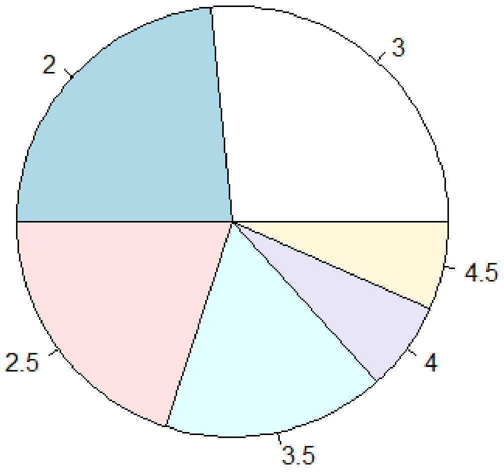
R Lesson 7

Higher Level Graphics
For categorical variables:

Bar Plot

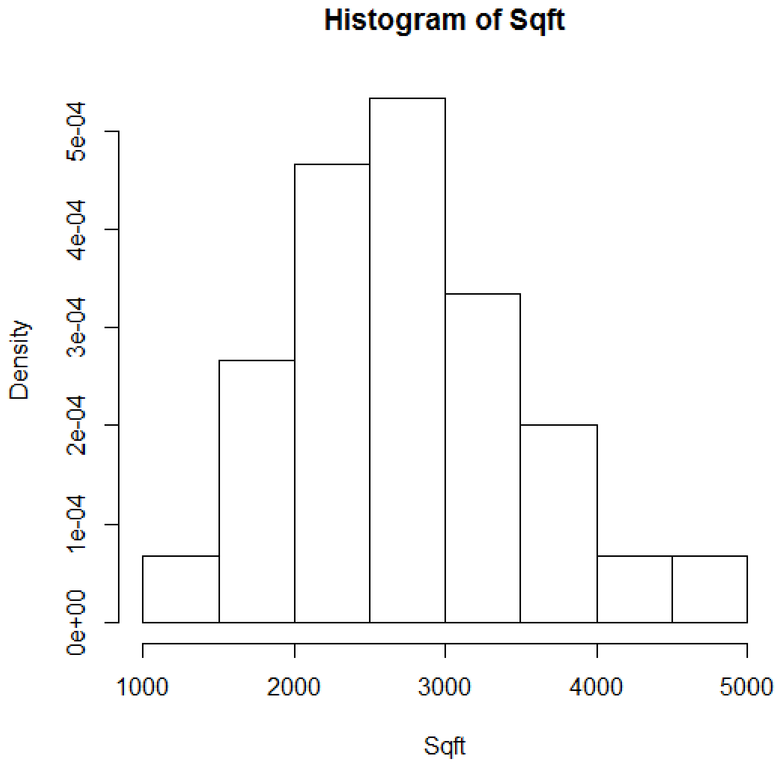


Pie Chart

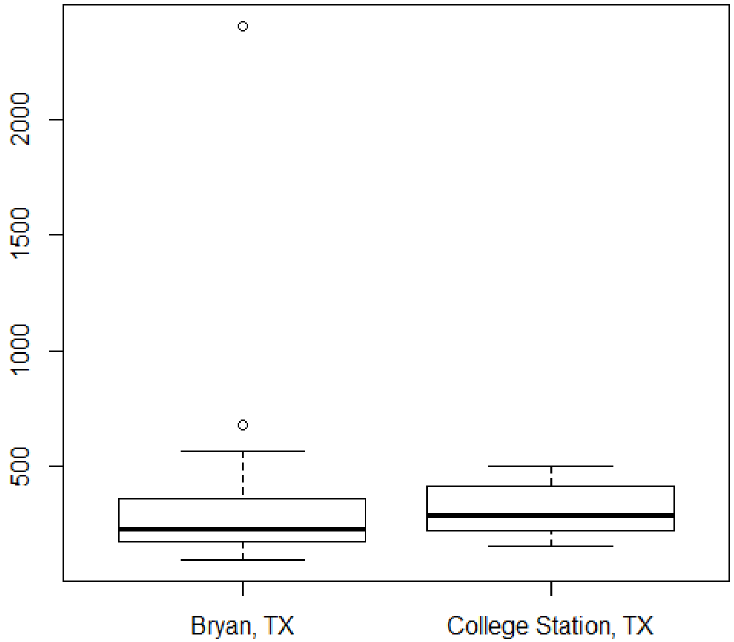


Higher Level Graphics
For continuous variables:

Histogram



Boxplot



Bar Plot – General Form

```
barplot(height, width = 1, space = NULL,
names.arg = NULL, beside = FALSE,
horiz = FALSE)
```

- *height:* vector or matrix describing the bars
- *width:* width of bars
- *space:* amount of space between bars
- *names.arg:* vector of names below bars
- *beside:* controls bar stacking
- *horiz:* orientation of bars

Function for Bar Height

- uses the cross-classifying factors to build a contingency table of the counts at each combination of factor levels
  - Frequency distribution table
- General form

```
table(…)
```

  - …: one or more objects which can be interpreted as factors

Bar Plot Example

- barplot(sort(table(Baths),decreasing=TRUE))

Compute Cross-tab Statistics

- Apply a function to each group of values given by a unique combination of the levels of certain factors.
- General form

  tapply(*X, index, function*)

  - *X* – vector of values being analyzed
  - *index* – classification (grouping) values
  - *function* – name of function to be applied
  - tapply(bcs$Price, bcs[,3:4], mean)

Pie Chart – General Form

```
pie(x, labels = names(x), clockwise = FALSE,
init.angle = if(clockwise) 90 else 0)
```

  - ▪ *x:* vector of non-negative numerical quantities
  - ▪ *labels:* names for slices
  - ▪ *clockwise:* specifies order slices are drawn
  - ▪ *init.angle:* specifies starting angle for slices

- Pie charts are not an ideal method of displaying information.

Pie Chart Example

- pie(sort(table(Baths),decreasing=TRUE))

Histogram – General Form

hist(*x*, *breaks* = "Sturges", *freq* = NULL)

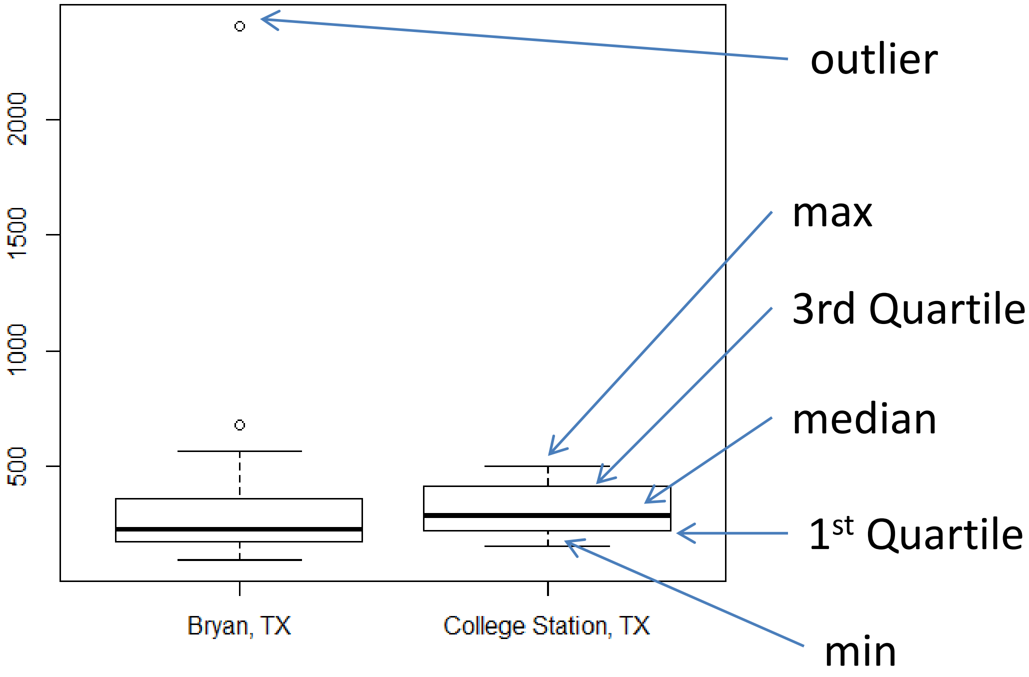
- *x:* vector of values for which histogram is desired
- *breaks:*
  - ✓ a vector giving the breakpoints between histogram cells (only way to force)
  - a character string naming an algorithm to compute the number of cells (Sturges, Scott, Freedman-Diaconis/FD)
  - a single number giving the number of cells for the histogram
  - a function to compute the number of cells
- *freq:* specifies type of y values
  - TRUE - counts
  - FALSE - probability densities

Histogram Example

- hist(Sqft, freq=FALSE, breaks=brv)

Boxplot

- 5 number summary



outlier

max

3rd Quartile

median

1st Quartile

min

Boxplot – General Forms

```
boxplot(formula, range = 1.5, notch = FALSE)
boxplot(x, …)
```

- *formula:* such as y ~ grp (y as a function of grp)
- *x:* vector or list of vectors (like data frame columns)
- *range:* position of whiskers
    - positive value n: limit to n times interquartile range
    - 0: include all values
- *notch:* draw notch on each side of boxes

Boxplot Example

- boxplot(Price ~ Location, range=0)

More Graphics Parameters

- Some can only be set with par()
- mfrow, mfcol: multiple plots per image
  - *par(mfrow=c(nr, nc))*
  - *mfrow vs. mfcol controls the order graphs appear*

More Graphics Parameters

- mar, oma, mai, omi: margin controls

  par(*type*=c(*bottom, left, top, right*))

  - *type:*
    - *mar/mai:* margins for the specific plot area
    - *oma/omi:* overall (page) margins
    - – parameters ending with i use inches as units
    - – parameters not ending in i use lines as units
    - *bottom, left, top, right:* number of lines or inches

  Example: par(omi=c(.5, .75, 1, .75))

Enhancing Graphs – Optional Parameters

- Add a title to the graph
  - *main=*
    main = "Real Estate Prices"
    main = paste("Histogram of" , xname)
- Scale ranges of x or y axes
  - *xlim =, ylim =* (numeric vectors of length 2)
    ylim=c(0,10)

Enhancing Graphs – Optional Parameters

- Change the axis labels
  - *xlab =, ylab =*

    ylab = “Square Footage”
- Specify a vector of character strings under plotted groups
  - *names =*

    names = c(“Bryan”, “Coll. Sta.”)

Enhancing Graphs – Optional Parameters

- Control Alignment of Text
  - *adj=*
  - 0 left justifies
  - default of 0.5 centers text
  - 1 right justifies

  adj=0

Adding Objects to Existing Graphs

- Add generic X-Y plotting to an existing graph
- Same general form as plot() function
- add more points - like plot(*x,y,type=*“p”)

```
points(x ,y, col="blue")
```

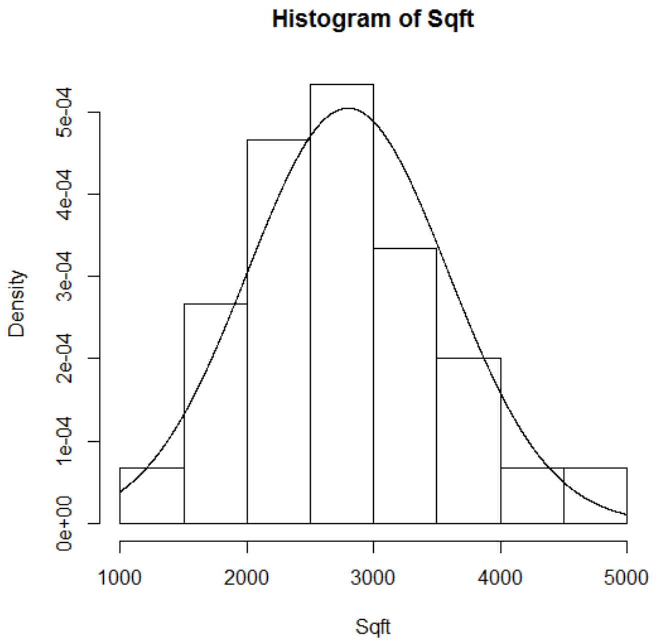
- add more points connected by lines - like plot(*x,y,type=*“l”)

```
lines(x, y, col="blue")
```

- Use lines(x, y, type=‘b’) to add points and lines

Business Scenario

- Add the normal distribution line to the histogram of Real Estate Square Footage



The Normal Distribution

- Bell curve
- 68 – 95 – 99.7% Rule
  - *68% of observations within 1 std. dev. of mean*
  - *95% of observations within 2 std. dev. of mean*
  - *99.7% of observations within 3 std. dev. of mean*

Normal Distribution Functions

- dnorm – densities for the normal distribution
- rnorm – pseudo-random generation
- General Forms:

dnorm(*x*, *mean* = 0, *sd* = 1)

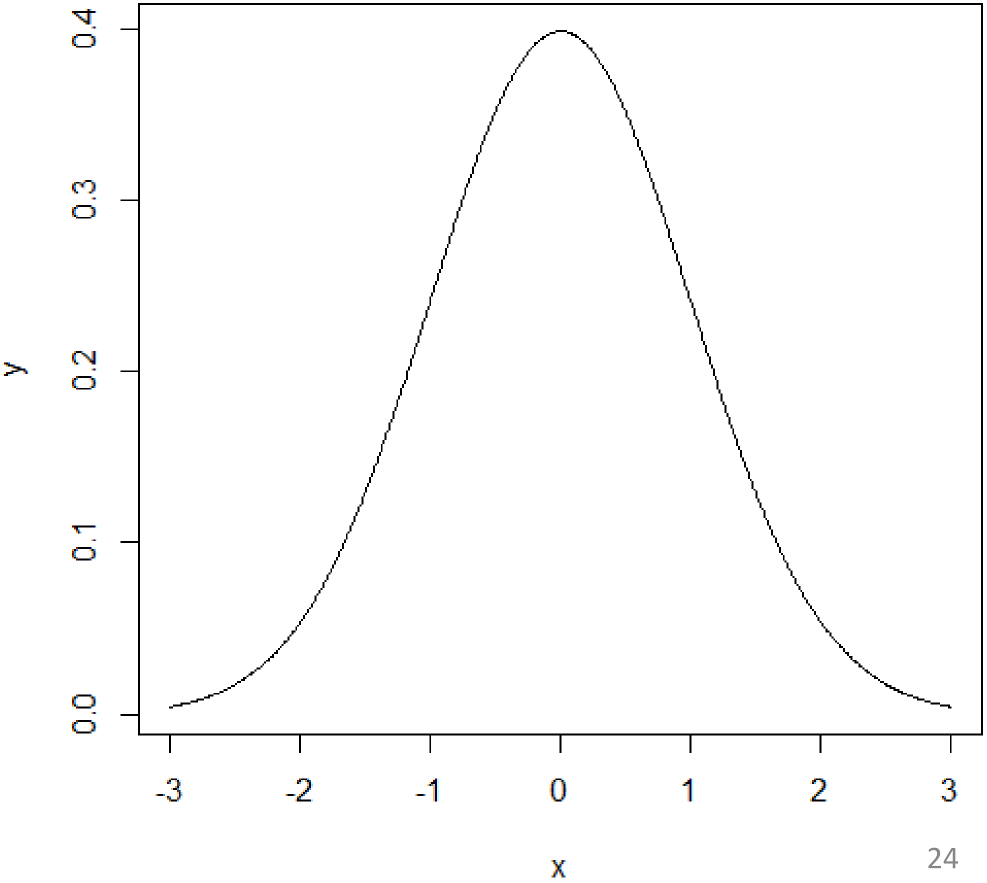
  - *x:* vector of quantiles
  - *mean:* vector of means
  - *sd:* vector of standard deviations

rnorm(*n*, *mean* = 0, *sd* = 1)

  - *n:* number of observations

Normal Distribution Example

-
  ```
  x <- seq(-3,3,0.01)
  y <- dnorm(x)
  plot(x,y,type="l")
  ```
- Use lines to add to add to existing



Adding Objects to Existing Graphs

- Fill a polygon defined by the x and y values
  - plots line of x and y coordinates
  - draws line from last x,y point back to first x,y point
  - fills in enclosed shape (if color is specified)
- General form:

```
polygon(x, y, …)
```

- *x:* vector of horizontal values
- *y:* vector of vertical values
- …*:* arguments such as graphical parameters

Polygon Example

```
x <- seq(-3,3,0.01)
y <- dnorm(x)
polygon(x, y, col=“blue")
```