R part II

Data Analysis with R

Dr. Irene Vrbik

University of British Columbia Okanagan

irene.vrbik@ubc.ca

Vector

A vector is an indexed list of data of any type.

Create vectors using a colon or `seq()` (R's version of range)

```
> 1:10
[1]  1  2  3  4  5  6  7  8  9 10
> seq(5, 1, by = -0.5 )
[1] 5.0 4.5 4.0 3.5 3.0 2.5 2.0 1.5 1.0
```

Create an empty vector with `c()` or fill it by specifying elements.

```
> c()
NULL
> c(4, 3, 5, 'a', 'd')
[1] "4" "3" "5" "a" "d"
```

Vector

Access elements in a vector using [ ]

```
> myVector = c(4, 3, 5, 'a', 'd')
> myVector[i]  #Returns the ith element of myVector
> myVector[1]
[1] "4"
```

Vectors

We could also name vectors (similar to Python's dictionaries)

```
> names(myVector) <- c("first", "second", "third",
"fourth","fifth")
> myVector["first"]
first
  "4"
```

Vectors in R

Question

Which (if any) of the following statements are true?

1. Vectors in R are indexed from 0. ✗
2. `1:10` creates a vector of ten numbers. ✓
3. A vector may have data value of different types. ✗
4. If `data <- 1:5` then `data[2] + data[3] = 3`. ✗

Matrices

A matrix is a structure of rows and columns where each data value is the same data type. All rows must have the same length. All columns must have the same length.

Create a matrix from the vector `x` using `matrix()`.

```
> matrix(x, nrow=5, ncol=3, byrow=False)
# Starts at [1,1] and fills the column first before going to
# next column.
# Need to only specify ncol or nrow.
```

Access elements using `[row, col]`. Leaving one of them blank returns the whole row or column.

```
> myMatrix[i, j]  #Returns ith row and jth column.
```

Matrices and Vectors

Append a vector to a matrix as a row using `rbind()`

```
> myMatrix = rbind(myMatrix, vec)
```

Append a vector to a matrix as a column using `cbind()`

```
> myMatrix = cbind(myMatrix, vec)
```

Matrices and Vectors

We could also name the columns are rows of a matrix:

```
> mat1 <- matrix(1:12, nrow=3)
> colnames(mat1) <- c("col1", "col2", "col3", "col4")
> rownames(mat1) <- c("row1", "row2", "row3")
> mat1["row1",] # same as mat[1,]
col1 col2 col3 col4
   1    4    7   10
> mat1$col3 # won't work (can only use $ on data.frames)
Error in mat1$col3 : $ operator is invalid for atomic vectors
```

Lists

A list is an ordered collection of objects of any type.

Create a list using `list()`. Specify names of elements by using `name =` inside the brackets.

```
> myList = list(x=1:4, y=c('a', 'b'))
#Creates a list with two elements x and y
```

Access elements using the double square brackets.

```
> myList[[2]]   # Returns the 2nd item of list (y)
> myList[['x']] #Returns item with the name x.
> myList$x #Returns item with the name x (no quotes)
```

Lists and Matrices in R

Question

Which (if any) of the following statements are true?

1. Data values in a list may be of different types. ✓
2. In a matrix, the number of rows and columns must be the same. ✗
3. Given matrix `m`, `m[2]` would return an error. ✗
4. Given matrix `m`, `m[,2]` would return all data in column 2. ✓

Question

Create a list called `grades` and add the following elements

1. Name (first name and last name),
2. Student number,
3. Assignment grades (multiple entries), and
4. Midterm grade.

Data Frame

A data frame is similar to a matrix but the columns can have different data types. Note these data frames are quite common and have uniform length of rows and columns.

To create a data frame by using `data.frame()`, specify names of variables within the brackets:

```
> myDF = data.frame(x = c(1:3), y = (2:4))
```

To convert a matrix into a `data.frame` using `as.data.frame()`.

```
> myDF = as.data.frame(myMatrix)
```

Accessing Data in Data Frames

Access elements using `[row, col]` or `$variable_name`.

```
> myDF[i, j]            #ith row and jth column
> myDF$x                #returns the column labelled x
```

Add new columns called `vec` into the data frame use in `$`.

```
> myDF$new_col = vec    #Adds vec as new_col
```

Factors

Factors are used for qualitative groups/categories (i.e. male/female). Use `as.factor()` to turn a vector or `data.frame` column into a factor.

```
mFyFactor = as.factor(x)
myDF$x = as.factor(myDF$x)
```

Access elements using `[ ]`:

```
myFactor[i]    #Returns ith element
```

Can use `class()` or `str()` to gain information about the type and/or structure of your variable/data. `str()` gives me detail.

Question

Which (if any) of the following are true?

1. Matrices must have the same number of rows as columns. ✗
2. Vectors must contain only one data type. ✓
3. ~~A factor can contain only characters.~~ *While we can use numbers to represent the factors (eg. 0 = male, 1 = female) they will be treated as characters in R.*
4. A data frame's columns can be of varying length. ✗

Subsets

Subsetting is used to extract data with particular values.

Syntax in R

```
subset(data, condition)
# if we want to check two conditions:
subset(data, condition1 & condition2)
```

Example in R

```
cars_bc = subset(cars, prov == 'BC')
```

Question

1. Create a data frame `mydata` with the following column names/data:
   1.1 `id` – numbers one to 5.
   1.2 `location` – "BC", "BC", "AB", "MB", "BC".
   1.3 `value` – 10, 20, 30, 40, 50.
   1.4 Make location a factor.
2. Add one more column to your data frame that is a factor
   2.1 `success` – "Y", "N", 'N", 'N", "Y".
3. Display only the data from BC and value ≥ 20.

Visualizing Data in R

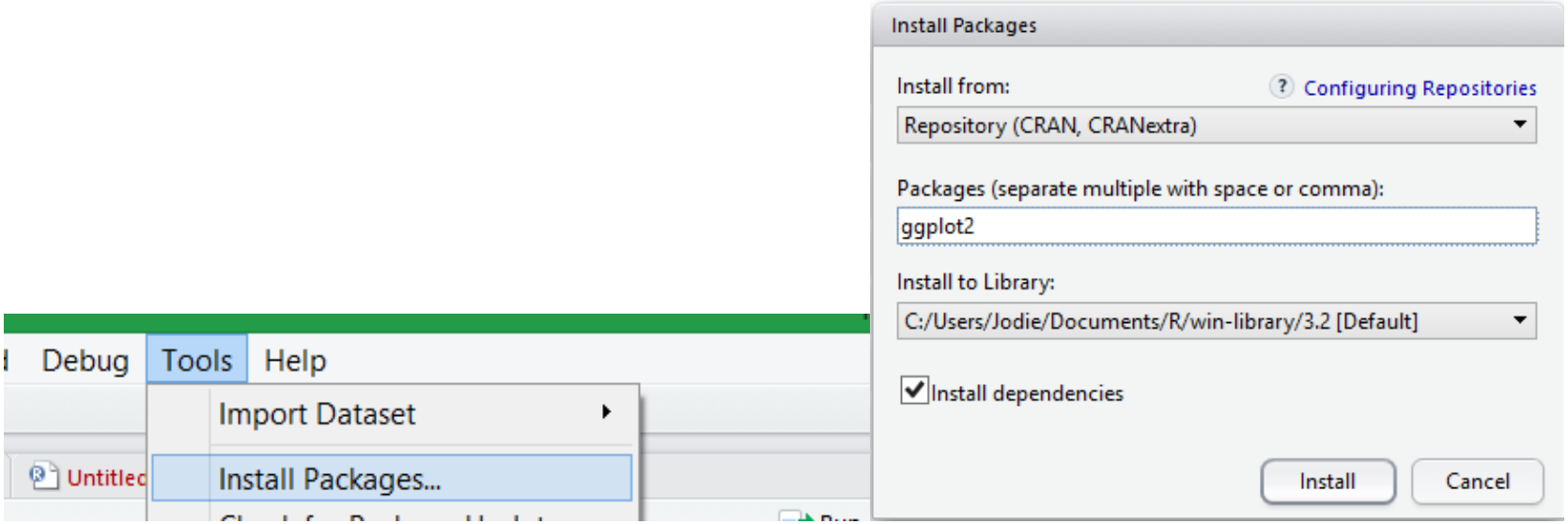
R supports several graphing libraries to produce graphs for qualitative and quantitative data including bar charts, histograms, and box plots.

We will use the package `ggplot2`. gg stands for Grammar of Graphics.

See the ggplot cheatsheet

Visualizing Data in R

To install `Tools` → `Install Packages...` Then input `ggplot2`.



Or (and this would be my preferred method) type the command:

```
install.packages("ggplot2")
```

To load that library into your working session type:

```
load("ggplot2") # quotes not necessary
```

Graphs for Qualitative Data: Frequency Tables

Frequency tables summarize the number of observations in each group.

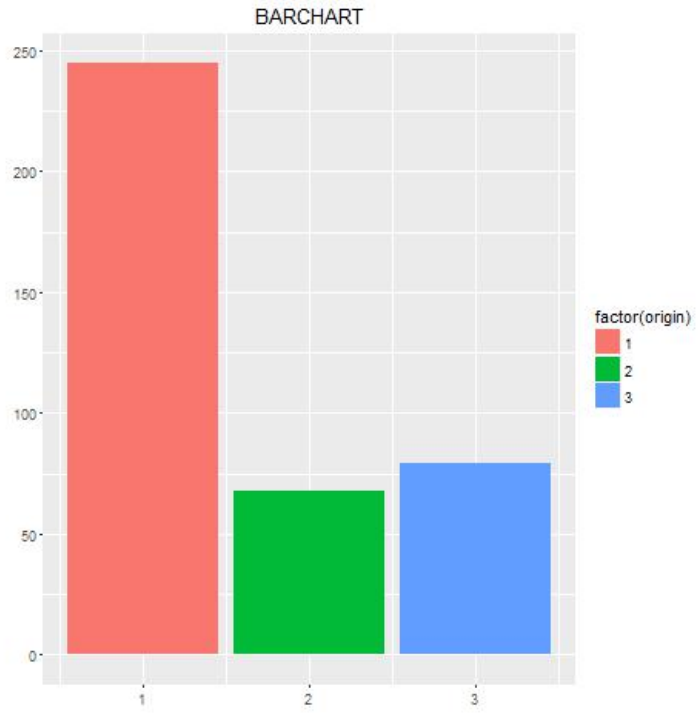
Frequency Table

```
> table(Auto$origin)
  1   2   3
245  68  79
```

Graphs for Qualitative Data: Bar Charts

Bar charts have each group along the x-axis and a vertical bar with the height representing the number of observations of each group.

Using the dataset `Auto` in the ISLR package.



Frequency Table

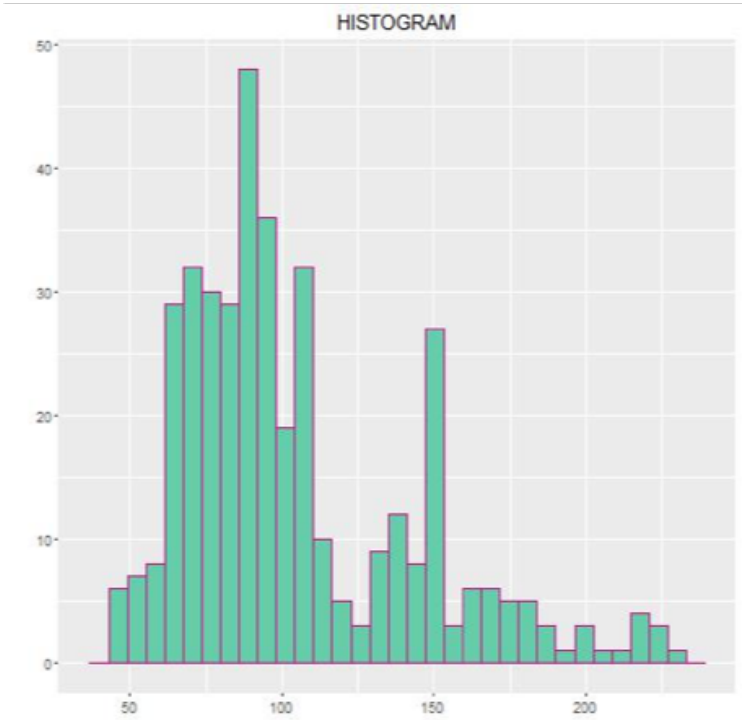
```
ggplot(Auto, aes(x=origin))
+ geom_bar(aes(fill=factor(origin)))
+ xlab("") + ylab("") + ggtitle("BARCHART")
```

Graphs for Quantitative Data: Histogram

A histogram is similar to a bar chart, but the x-axis is divided into bins.

The variable of interest is on the x-axis and the y-axis represents count of observations within each bin.

Visualizes the data distribution.



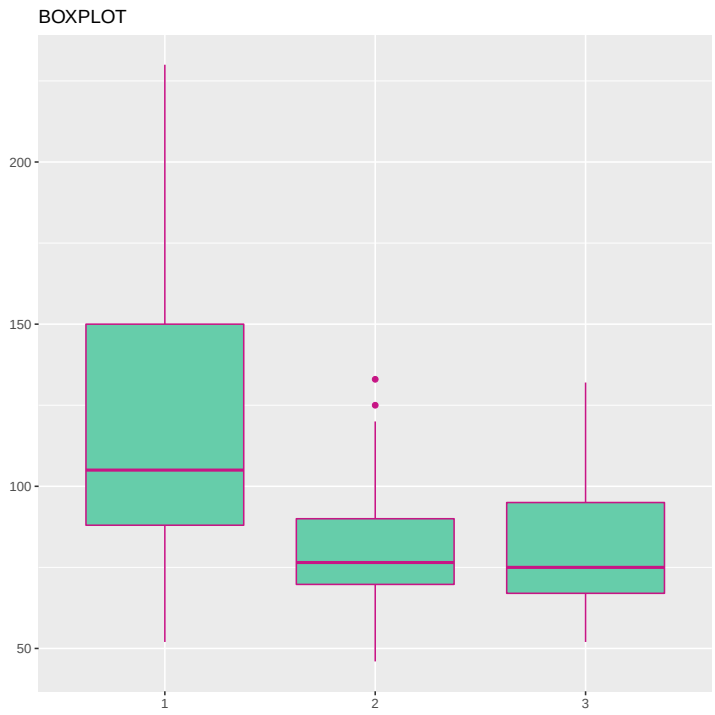
Histogram Example

```
ggplot(Auto, aes(x=horsepower))
+ geom_histogram(color='mediumvioletred', fill='mediumaquarmarine')
+ xlab("") + ylab("") + ggtitle("HISTOGRAM")
```

Graphs for Quantitative Data: Boxplot

A boxplot is a visualization of the five number summary.

1. Groups along the x-axis.
2. Data values along the y-axis.
3. Lowest and highest points are the min and max of the data respectively.
4. Bottom of box is Q1 and top is Q3.
5. Median is represented as the bar inside the box.
6. Single points represent outliers.



Boxplot Example Code

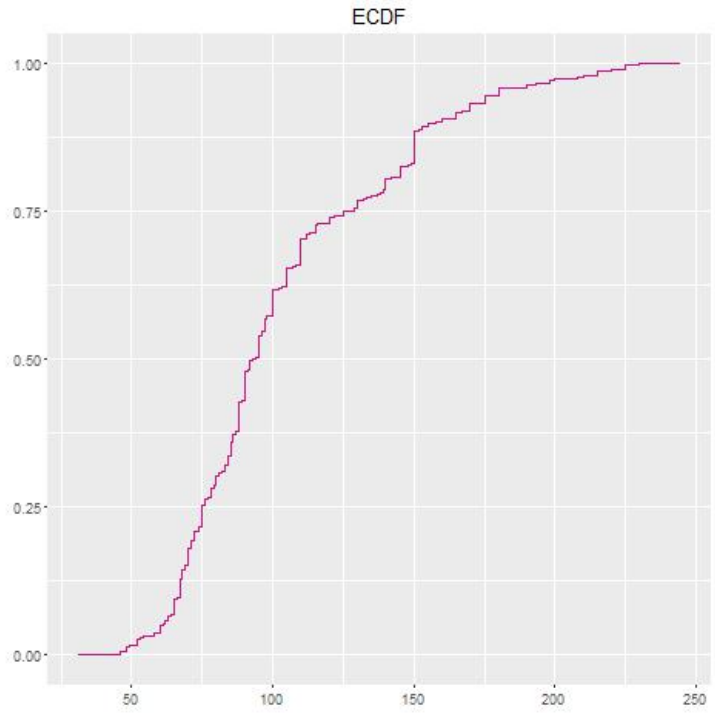
Boxplot Example

```
ggplot(Auto, aes(x=factor(origin), y=horsepower))
+ geom_boxplot(color='mediumvioletred', fill='mediumaquamarine')
+ xlab("") + ylab("") + ggtitle("BOXPLOT")
```

Graphs for Quantitative Data: ECDF

An empirical cumulative distribution function (ECDF) plot shows values along the x-axis and quantiles along the y-axis.

Each data point is plotted along with its corresponding quantile.



Boxplot Example

```
ggplot(Auto, aes(x=horsepower))
+ stat_ecdf(color='mediumvioletred')
+ xlab("") + ylab("") + ggtitle("ECDF")
```

Question

Which (if any) of the following are true?

1. Bar charts and histograms will work for the same variables. ✗
2. Boxplots show a five number summary. ✓
3. Variables type does not matter, any graph can be used. ✗
4. Histograms can give an idea of the distribution of a variable. ✓