Lesson 01
Introduction to R
Vectors

Outline

- Vectors in R
- Vector operations
  - Algebraic operations
  - Logical operations
- Vector elements
  - Accessing
  - Naming
- Factor vectors
- Missing data
- Using R functions
  - Pipes
  - Documentation

Vectors in R

- Vector – a collection of elements of same data type
  - The second best (after packages) feature of R
  - R is a **vectorized** language - all the operations are applied to the entire vector (instead of looping through its elements)
  - Similar to one-dimensional arrays although they are not thought as having a dimension
    - Row and columns are associated with matrices discussed later on
  - Most vectors are of numerical or character data type
- Vectors are created with a `c` (combine) function
  - Open **`Intro2_R_Vectors.r`**

```
# Create vector of interest rates on 10 loans
rates <- c(0.07, 0.075, 0.07, 0.065, 0.077, 0.0625,
0.065, 0.0775, 0.0575, 0.0575)
```

Vector Operations - Algebraic

- Algebraic operations on a single vector

```
# Multiply vector by 100 to get rates as
percentages
rates * 100
# Add 1% to each of the 10 rates
rates + 0.01
```

- Algebraic operations on multiple (two) vectors

```
rate_inc <- 1:10
rate_inc <- rate_inc/100
rates + rate_inc
```

- Subtraction, division, power, … all work the same way

Vector Operations - Miscellaneous

- Determine vector length

```
length(rates)
length(rates + rate_inc)
```

- Careful when vectors are NOT of the same size

```
rate_inc <- c(0, 0.01, 0.02)
rates + rate_inc
```

  - Smaller vector gets “recycled”, i.e. it gets repeated until matched with the longer vector
  - `c(0, 0.01, 0.02)` becomes
  - `0, 0.01, 0.02,0, 0.01, 0.02,0, 0.01, 0.02, 0`

Vector Operations - Logical

- Logical operations on a single vector

```
# See which loans are under 7%
rates < 0.07
```

- Logical operations on multiple (two) vectors

```
# Confirm all the loans experienced rate increase
rate_inc <- rep(1,10)
new_rates <- rates + rate_inc/100
new_rates > rates
all(new_rates > rates)
```

Vector Operations - Character

- Logical operations on multiple (two) vectors

```
# Confirm some loans experienced rate increase
rate_inc <- c(0, 0.01)
new_rates <- rates + rate_inc
new_rates > rates
any(new_rates > rates)
```

- An example of a character vector

```
loanType <- c("Mortg", "Mortg", "Mortg", "Car",
"Car", "Mortg", "Other", "Car", "Mortg", "Mortg")
```

  – Determine the length of each vector element

```
nchar(loanType)
```

Vector Elements

- Accessing individual element of a vector
  - Use square brackets []

```
# Accessing first 5 elements
loanType[1:5]
# Accessing just elements 1 and 5
loanType[c(1,5)]
```

- Works as long as argument is a legit vector

```
# It can get weird, accessing just mortgage loan rates
mortgType <- loanType == "Mortg"
rates[mortgType]
mean(rates[mortgType])
```

- Using one vector to name another

```
names(rates) <- loanType
rates
```

Factor Vectors

- Factor is essentially a categorical variable
  - Important in analytics when analyzing data
    - How many loans of different types do we have?
    - What is the average mortgage rate?
    - What is the average car payment?

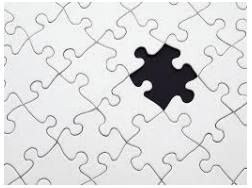
```
loanTypeFactor <- as.factor(loanType)
loanTypeFactor
```

  - Notice the absence of “” and the 3 unique levels
- Each level is represented with a unique integer

```
as.numeric(loanTypeFactor)
```

  - For nominal factors (like gender) the order does not matter
    - Appears to be alphabetical (so Female would be 1, Male 2)
  - For ordinal factors (like education) the order does matter

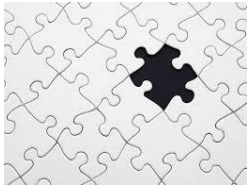
Missing Data in R



- Two types of missing data in R
  - NA: not available
  - NULL: the absence of value
- NA or Not Available represents truly missing data
  - May or may not become available later
  - `NA` becomes a part of the vector instead

```
rates <- c(0.07, 0.075, 0.07, 0.065, 0.077, 0.0625,
NA, 0.0775, 0.0575, 0.0575)
rates
is.na(rates)
```

Missing Data in R



- NA or Not Available (cont.)

```
loanType <- c("Mortg", "Mortg", "Mortg", "Car", "Car",
"Mortg", NA, "Car", "Mortg", "Mortg")
loanType
is.na(loanType)
```

- NULL represents the absence of value
  - Comes from DB terminology
  - Used when the absence of value is the only truly legitimate "value" (the value will never become "available")
  - Future value (FV) or balloon payment is typically optional

```
fv_balloon <- c(5000, NULL, 10000, 15000)
fv_balloon
is.null(fv_balloon)
```

Using Pipes in R

- New way of calling functions in R

```
# Traditional vs. piped function calls
mean(rates, na.rm = TRUE)
rates %>% mean(na.rm = TRUE)
```

- Facilitates more efficient nesting of calls

```
# Nesting of functions is simpler and more efficient
sum(is.na(rates))
rates %>% is.na %>% sum
```

- Will be used later on in class

Function Documentation

- My first attempt is always to Google it
  - "R documentation pmt function" returns
  - From **`optiRum`** package discussed before
    - https://www.rdocumentation.org/packages/optiRum/versions/0.40.1/topics/PMT
  - From **`finCal`** package that could have been used before
    - https://www.rdocumentation.org/packages/FinCal/versions/0.6.3/topics/pmt
- If you know the function name
  - Use `?` followed immediately by the name of the function
  - R documentation under **Help** tab in the bottom-right pane

Summary

- Defined a vector
  - Collection of elements of the same type
- Reviewed some basic vector operations
  - Operations act on the entire vector
  - Algebraic and logical operations
- Showed how to access individual and groups of vector elements
- Discussed the importance of factor vectors
- Compared two types of missing values
- Showed how to get help on R functions