R Lesson 8

Adding Straight Lines (4 methods)

- General forms:
- Slope/Intercept line as in y=b(x)+a

abline(*a*, *b*)

- A horizontal line at the value y

abline(*h*=y)

- A vertical line at the value x

abline(*v*=x)

- Fit line from linear model of control x and response y

abline(*lm(y~x)*)

Linear Models

- General Form:

  lm(*formula*, *data, na.action*)

  - *formula:* symbolic description of model like response~control
  - *data:* optional data frame or environment containing variables
  - *na.action:* function indicating how to handle Nas (default is na.omit)

Linear Model Examples

- Assign a model object of price by square feet
  tm<-lm(Price~Sqft, bcs)

- Show a summary of the model results
  summary(tm)

Adding Legends to Graphs

- General form:
  legend(*xy.coords, legend, pch, col*)
  - *xy.coords:* specify x, y coordinates of upper left corner or use locator(1) function to click on position
  - *legend:* vector with legend text
  - *pch:* plotting characters
  - *col:* colors of plot objects
- Example
  legend(locator(1),c("treatment", "control"),pch=c(1,1),col=c(2,4))

Adding Text to Graphs

- text – puts text inside plot
- mtext – write text into the margin of the plot
- General forms:

text(*x*, *y*, *text*, *cex*=NA, *adj*=NA)

mtext(*text*, *side*=3 , *cex*=NA, *adj*=NA)

  - *x, y:* coordinates for text (centered by default)
  - *side:* 1=bottom, 2=left, 3=top, 4=right side of plot
  - *cex:* optional font size override
  - *adj:* 0=left or bottom, 1= right or top alignment

Mathematical Annotation in R

- plotmath
  - *not a function but a collection of features*
  - *imbed inside text functions and options*
  - ?plotmath
  - demo(plotmath)
  - example(plotmath)

Mathematical Annotation in R

expression(…)

- ✓ no variables get evaluated
- ✓ add plain text inside expression with paste()

- Examples

```
text(4, 7, expression(bar(x) == sum(frac(x[i], n), i==1, n)))
xlab = expression(paste("Phase Angle ", phi))
```

Mathematical Annotation in R

bquote(*expr1 .(expr2)*)

- *expr1:* plotmath expression – no evaluation
- *.(expr2) :* variables inside **.( )** are evaluated
- NOTE: use **.( )** as often as you like inside bquote

- Example

```
MU <- 8.25
ylab=bquote(mu== .(MU))
```

Conditional Data Assignment

- Business Scenario:
- Write a single statement that specifies blue for Bryan and red for College Station.
- Use **ifelse** to process multiple elements
- Returns object of same shape as conditional expression

Conditional Data Assignment

- General form:

  ifelse(*cond*, *Texpr*, *Fexpr*)

  - *cond:* Expression that produces TRUE/FALSE value(s)
  - *Texpr:* value expression when condition is TRUE
  - *Fexpr:* value expression when condition is FALSE

- Example:

  colv <- ifelse(Location=="Bryan, TX“, "blue“, “red”)

Conditional Code Execution

- Business Scenario: Execute a section of code based on a person's credit score threshold
- Basic control-flow constructs:
  - if(cond){…}
  - if(cond){…}else{…}
  - for(i in seq){…}
  - while(cond){…}
- ??control

Conditional Code Execution

- General form:

  if (*cond*) {*Texpr*} else {*Fexpr*}

  - *cond:* Expression that produces TRUE/FALSE value
  - *Texpr:* action when condition is TRUE
  - else: optional reserved word preceding Fexpr
  - *Fexpr: action when condition is FALSE*
- {} optional for simple statement on one line
- Must evaluate to a length-one logical vector
- NOTE: Combined conditions can use && and ||.

Conditional Code Execution

- Example:

```
i <- 6
if(i > 6){
cat("i is bigger than 6.\n")
}else{
cat("i is 6 or less.\n")
}
```

ifelse vs. if

- same basic principle but different
- NOT fully interchangeable
- ifelse – multiple values like vector or matrix
  - Use to assign values
- if – ONLY one value like a scalar
  - Use for flow control

Repetitive Code Execution (Loops)

- For Loop – repeat process for specified iterations
- General form:

  for(*i* in *seq*){…}

  - *i:* name of variable to store loop position
  - *seq:* any vector of values to control repetitions
  - {…} R expression to be repeated
- R loops are evaluated at the start of the loop

For Loop

- Example:

```
y <- 1:10
for(i in y){
    if(i > 6){
        cat("i is bigger than 6.\n")
    }else{
        cat("i is 6 or less.\n")
    }
}
```

While Loop

- General Form

  while(*cond*){…}

  - *cond:* Expression that produces TRUE/FALSE value
  - {…} R expression to be repeated (must contain code to manage repetitions)

While Loop

- Example:

```
i <- 1
while(i <= 10){
    if(i > 6){
        cat("i is bigger than 6.\n")
    }else{
        cat("i is 6 or less.\n")
    }
    i <-i+1 #crucial to prevent infinite loop
}
```

Conditional Processing

- Which indices are TRUE?

  which(*x*)

  - *x:* logical vector or array. NAs treated as FALSE.
- Example:

  Bryan <- which(Location=="Bryan, TX")