Chapter 9

Nick Lauerman

Graphs are included in a seperate code block, when needed, to allow caching of data within the markdown enviroment allowing to reduced time to execute and complile the page.

Change in section 9.2.2 leaving the second subset to reflect whatis in the books code verse what is in the description and using all2011.csv supplied by author

setup

```
library(plotrix)
library(ggplot2)
library(lattice)
library(car)
library(plyr)
```

Section 9.2

Section 9.2.2

```
data2011 <- read.csv("E:/Baseball/data/book/all2011.csv",
                     header = FALSE)
fields <- read.csv("E:/Baseball/data/book/fields.csv")
names(data2011) <- fields[ ,"Header"]


data2011$HALF_INNING <- with(data2011,
                             paste(GAME_ID, INN_CT, BAT_HOME_ID))
data2011$RUNS.SCORED  <-  with(data2011,
                                (BAT_DEST_ID >3) +
                                     (RUN1_DEST_ID > 3) +
                                     (RUN2_DEST_ID > 3) +
                                     (RUN3_DEST_ID > 3))

get.state <- function(runner1, runner2, runner3, out){
     runners <- paste0(runner1, runner2, runner3)
     paste(runners, out)
}

RUNNER1 <- ifelse(as.character(data2011[, "BASE1_RUN_ID"]) == "", 0, 1)
RUNNER2 <- ifelse(as.character(data2011[, "BASE2_RUN_ID"]) == "", 0, 1)
RUNNER3 <- ifelse(as.character(data2011[, "BASE3_RUN_ID"]) == "", 0, 1)
```

```r
data2011$STATE <- get.state(RUNNER1, RUNNER2, RUNNER3,  data2011$OUTS_CT)

NRUNNER1 <- with(data2011,
                 as.numeric(RUN1_DEST_ID ==1 |
                                 BAT_DEST_ID == 1))
NRUNNER2 <- with(data2011,
                 as.numeric(RUN1_DEST_ID == 2 |
                                 RUN1_DEST_ID == 2 |
                                 BAT_DEST_ID == 2))
NRUNNER3 <- with(data2011,
                 as.numeric(RUN1_DEST_ID == 3 |
                                 RUN2_DEST_ID == 3 |
                                 RUN3_DEST_ID == 3 |
                                 BAT_DEST_ID == 3))
NOUTS <- with(data2011, OUTS_CT + EVENT_OUTS_CT)
data2011$NEW.STATE <- get.state(NRUNNER1, NRUNNER2, NRUNNER3, NOUTS)

data2011 <- subset(data2011, (STATE != NEW.STATE) | (RUNS.SCORED > 0))

require(plyr)
data.outs <- ddply(data2011, .(HALF_INNING),
                   summarize,
                   Outs.Inning = sum(EVENT_OUTS_CT))
data2011 <- merge(data2011, data.outs)
data2011C <- subset(data2011, Outs.Inning == 3)
data2011C <- subset(data2011, BAT_EVENT_FL == TRUE)

require(car)
data2011C$NEW.STATE <- recode(data2011C$NEW.STATE,
                              "c('000 3', '100 3','010 3',
                              '001 3','110 3','101 3','011 3',
                              '111 3') = '3' ")
```

```r
table(data2011C$Outs.Inning)
```

```
##
##      0      1      2      3
##     79    380    522 184264
```

Section 9.2.3

```r
T.matrix <- with(data2011C,
                 table(STATE, NEW.STATE))
P.matrix <- prop.table(T.matrix, 1)

P.matrix <- rbind(P.matrix, c(rep(0, 24), 1))

P1 <- round(P.matrix["000 0", ], 3)
data.frame(Prob=P1[P1 > 0])
```

```
##         Prob
```

```
## 000 0 0.027
## 000 1 0.677
## 001 0 0.006
## 010 0 0.050
## 100 0 0.240
```

```
P2 <- round(P.matrix["010 2",], 3)
data.frame(Prob = P2[P2 > 0])
```

```
##        Prob
## 000 2 0.020
## 001 2 0.006
## 010 2 0.055
## 100 2 0.245
## 101 2 0.034
## 3     0.640
```

Section 9.2.4

```
count.runners.outs <- function(s){
     sum(as.numeric(strsplit(s,"")[[1]]),
         na.rm = TRUE)
}

runners.outs <- sapply(dimnames(T.matrix)[[1]],
                      count.runners.outs)[-25]
R <- outer(runners.outs + 1,
           runners.outs,
           FUN = "-")
dimnames(R)[[1]] <- dimnames(T.matrix)[[1]][-25]
dimnames(R)[[2]] <- dimnames(T.matrix)[[1]][-25]

R <- cbind(R, rep(0,24))

simulate.half.inning <- function(P, R, start = 1){
     s <- start
     path <- NULL
     runs <- 0
     while(s < 25){
          s.new <- sample(1:25, 1, prob = P[s, ])
          path <- c(path, s.new)
          runs <- runs + R[s, s.new]
          s  <- s.new
     }
     runs
}

RUNS <- replicate(10000,
                  simulate.half.inning(T.matrix, R))
table(RUNS)
```

```
## RUNS
```

```
##    0    1    2    3    4    5    6    7    8    9   11
## 6631 1948  813  322  163   81   26    9    4    2    1
```

```
sum(RUNS[RUNS >= 5]) / 10000
```

```
## [1] 0.0685
```

```
mean(RUNS)
```

```
## [1] 0.5877
```

```
RUNS.J <- function(j){
     mean(replicate(10000,
                    simulate.half.inning(T.matrix, R, j)))
}
Runs.Expectancy <- sapply(1:24, RUNS.J)
Runs.Expectancy <- t(round(matrix(Runs.Expectancy, 3, 8), 2))
dimnames(Runs.Expectancy)[[2]] <- c("0 Outs", "1 Outs","2 Outs")
dimnames(Runs.Expectancy)[[1]] <- c("000","001","010","011","100",
                                    "101","110","111")
Runs.Expectancy
```

```
##     0 Outs 1 Outs 2 Outs
## 000   0.59   0.32   0.10
## 001   1.53   1.01   0.33
## 010   1.45   1.14   0.44
## 011   2.44   1.86   0.69
## 100   1.08   0.63   0.22
## 101   1.99   1.29   0.46
## 110   1.89   1.27   0.42
## 111   2.75   1.87   0.69
```

```
Runs <- matrix(c(0.47, 0.25, 0.10, 1.45, 0.94, 0.32,
                 1.06, 0.65, 0.31, 1.93, 1.34, 0.54,
                 0.84, 0.50, 0.22, 1.75, 1.15, 0.49,
                 1.41, 0.87, 0.42, 2.17, 1.47, 0.76),
               8, 3,  byrow = TRUE)
Runs - Runs.Expectancy
```

```
##     0 Outs 1 Outs 2 Outs
## 000  -0.12  -0.07   0.00
## 001  -0.08  -0.07  -0.01
## 010  -0.39  -0.49  -0.13
## 011  -0.51  -0.52  -0.15
## 100  -0.24  -0.13   0.00
## 101  -0.24  -0.14   0.03
## 110  -0.48  -0.40   0.00
## 111  -0.58  -0.40   0.07
```

Section 9.2.5

```
P.matrix.3 <- P.matrix %*% P.matrix %*% P.matrix

sorted.P <- sort(round(P.matrix.3["000 0", ], 3),
                 decreasing = TRUE)
head(data.frame(Prob = sorted.P))
```

```
##          Prob
## 3       0.369
## 100 2 0.240
## 000 2 0.085
## 110 1 0.057
## 100 1 0.053
## 010 2 0.048
```

```
Q <- P.matrix[-25, -25]
N <- solve(diag(rep(1, 24)) - Q)

N.0000 <- round(N["000 0", ], 2)
head(data.frame(N = N.0000))
```

```
##            N
## 000 0 1.04
## 000 1 0.76
## 000 2 0.63
## 001 0 0.01
## 001 1 0.04
## 001 2 0.06
```

```
sum(N.0000)
```

```
## [1] 4.28
```

```
Length <- round(t(N %*% rep(1, 24)), 2)
data.frame(L= Length[1, 1:8])
```

```
##            L
## 000 0 4.28
## 000 1 2.88
## 000 2 1.47
## 001 0 4.37
## 001 1 2.97
## 001 2 1.51
## 010 0 4.30
## 010 1 2.93
```