ECS256 - Homework II

Olga Prilepova, Christopher Patton, Alexander Rumbaugh,
John Chen, Thomas Provan

February 11, 2014

Problem 1.a

First, we'll derive π_i. The definition of the tree searching markov model leads to the following set of balance equations for the long-run state probabilities:

$$\pi_i = \pi_{i-1} q_{i-1} = \pi_0 \prod_{j=0}^{i-1} q_j \quad \text{for } i \geq 1, \text{ and}$$

$$\pi_0 = \sum_{i=1}^{\infty} \pi_i (1 - q_i) \quad \text{for } i = 0.$$

This definition for π_0 is a bit unweildy. We can also think of this quantity as one over the expected recurrance time, as in eq. (10.63) in the book:

$$\begin{aligned}
\pi_0 &= \frac{1}{E(T_{0,0})} \\
E(T_{0,0}) &= 1 + \sum_{k \neq 0} p_{0,k} E(T_{k,0}) \\
&= 1 + p_{0,1} E(T_{1,0}) \\
&= 1 + p_{0,1}(1 + \sum_{k \neq 0} p_{1,k} E(T_{k,0})) \\
&= 1 + p_{0,1}(1 + p_{1,2} E(T_{2,0})) \\
&= 1 + p_{0,1}(1 + p_{1,2}(1 + \sum_{k \neq 0} p_{2,k} E(T_{k,0}))) \\
&= 1 + p_{0,1}(1 + p_{1,2}(1 + p_{2,3} E(T_{3,0})))
\end{aligned}$$

and so on. This unravels into a familiar closed form:

$$\begin{aligned}
E(T_{0,0}) &= 1 + q_0(1 + q_1(1 + q_2(1 + \dots) \dots))) \\
&= 1 + q_0 + q_0 q_1 + q_0 q_1 q_2 + \dots \\
&= 1 + \sum_{i=0}^{\infty} [\prod_{j=0}^{i-1} q_j]
\end{aligned}$$

If the model is positive recurrent, then there exists some value R such that

$$R = \sum_{i=0}^{\infty} [\prod_{j=0}^{i-1} q_j] < \infty.$$

Thus,

$$\pi_i = \frac{\prod_{j=0}^{i-1} q_j}{1+R} \quad \text{for } i \geq 0.$$

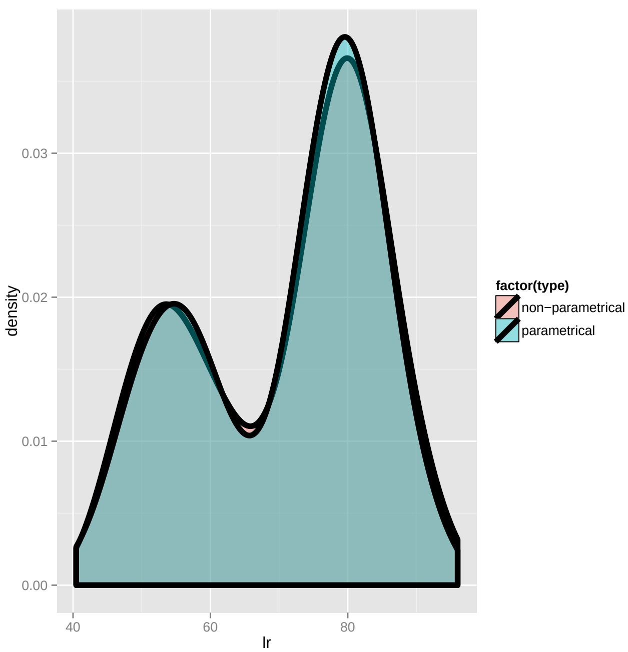
Next, $E(T_{i,0})$ follows a similar pattern.

$$\begin{aligned} E(T_{i,0}) &= 1 + \sum_{k \neq 0} p_{i,k} E(T_{k,0}) \\ &= 1 + p_{i,i+1} E(T_{j+1,0}) \\ &= 1 + q_i + q_i q_{i+1} + q_i q_{i+1} q_{i+2} + \dots \\ &= 1 + \sum_{j=i}^{\infty} \Big[\prod_{k=0}^{j-1} q_k \Big]. \end{aligned}$$

Problem 4

4.a-b

TODO: The picture needs better descriptors .



Appendix

Problem 4

```r
#install.packages("ggplot2")
#install.packages("mixtools")
library(mixtools)
library(ggplot2)

#4.a ###############################
p<-ggplot(data.frame(faithful))
p+geom_density(aes(x=faithful$waiting))

#4.b ###############################
simulateFromDist <- function(n,p1,m1,s1,m2,s2){
        k1 <- p1*n #proportion of type 1
        k2 <- n-k1+1 #proportion of type 2
        x1 <- rnorm(k1, mean=m1, sd=s1)
        x2 <- rnorm(k2, mean=m2, sd=s2)
        c(x1,x2) #order of events doesn't matter for histogram, so simply concatenate
}

####from mixtools simulation
mixout<-normalmixEM(faithful$waiting,lambda=0.5,mu=c(55,80),sigma=10,k=2)
str(mixout)
# $ lambda    : num [1:2] 0.361 0.639
# $ mu        : num [1:2] 54.6 80.1
# $ sigma     : num [1:2] 5.87 5.87

sim_waiting<-simulateFromDist(length(faithful$waiting),0.361,54.6,5.87,80.1,5.87)

data <- rbind( data.frame(type="non-parametrical", lr=faithful$waiting), data.frame(type="parametr
m <- ggplot(data, aes(x=lr))
m <- m + geom_density(aes(fill=factor(type)), size=2, alpha=.4)
m
```