HW 11

Jeff Gould

11/12/2020

1

$X(A, S, T, C, B, E, R, D)$, each $X_i \in \{0, 1\}$

a) Show $P(X)$ is completely determined for $\forall X \in S$

Notation: $P(A) \leftrightarrow P(A = i)$, $i \in \{0, 1\}$

$P(A, S, T, C, B, E, R, D) = P(T, C, B, E, R, D|A, S)P(A, S) = P(T, C, B, E, R, D|A, S)P(A)P(S)$

$P(T, C, B, E, R, D) = P(E, R, D|T, C, B)P(T, C, B) = P(E, R, D|T, C, B)P(T)P(C)P(B)$

$P(E, R, D|B) = P(R|E)P(D|E, B)P(E)$

Putting it all together:

$P(A, S, T, C, B, E, R, D) = P(T, C, B, E, R, D|A, S)P(A, S) = P(T, C, B, E, R, D|A, S)P(A)P(S) =$

$P(E, R, D|T, C, B)P(T, C, B|A, S)P(A)P(S) = P(E, R, D|T, C, B)P(T|A)P(C|S)P(B|S)P(A)P(S) =$

$P(R|E)P(D|E, B)P(E|T, C)P(T|A)P(C|S)P(B|S)P(A)P(S)$

And Table 1 provides values for each of the above (here $P(A)$ implies $P(A = 1)$, and $\bar{A} \leftrightarrow A = 0$)

$P(A) = 0.1,\ P(S) = 0.5$

$P(T|A) = 0.05,\ P(T|\bar{A}) = 0.01$

$P(C|S) = 0.10,\ P(C|\bar{S}) = 0.01$

$P(B|S) = 0.60,\ P(B|\bar{S}) = 0.30$

$P(E|C, T) = 1$ if either C, T =1, 0 if both =0

$P(R|E) = 0.98,\ P(R|\bar{E}) = 0.05$

$P(D|E, B) = 0.90,\ P(D|E, \bar{B}) = 0.70,\ P(D|\bar{E}, B) = 0.80,\ P(D|\bar{E}, \bar{B}) = 0.1$

b) Compute $P(R = 1|A = 1, S = 0, D = 1)$

i) Using Metropolis Hastings

MH Ratio $= \frac{P(\omega')R(\omega',\omega)}{P(\omega)R(\omega,\omega')} = \frac{P(\omega')}{P(\omega)}$

$R(\omega', \omega) = R(\omega, \omega')$

$P(\omega') = P(R'|E')P(D = 1|E', B')P(E'|T', C')P(T'|A = 1)P(C'|S = 0)P(B'|S = 0)P(A = 1)P(S = 0)$

$P(\omega) = P(R|E)P(D=1|E,B)P(E|T,C)P(T|A=1)P(C|S=0)P(B|S=0)P(A=1)P(S=0)$

$\frac{P(\omega')}{P(\omega)} = \frac{P(R'|E')P(D=1|E',B')P(E'|T',C')P(T'|A=1)P(C'|S=0)P(B'|S=0)P(A=1)P(S=0)}{P(R|E)P(D=1|E,B)P(E|T,C)P(T|A=1)P(C|S=0)P(B|S=0)P(A=1)P(S=0)} =$

$\frac{P(R'|E')P(D=1|E',B')P(E'|T',C')P(T'|A=1)P(C'|S=0)P(B'|S=0)}{P(R|E)P(D=1|E,B)P(E|T,C)P(T|A=1)P(C|S=0)P(B|S=0)}$

Use a burn in time of 10,000 iterations, sample every 1,00 iterations, and run for 260,000 total iterations. This will give us 251 samples

```r
### initialize State Space
stateSpace <- matrix(data = 0, ncol = 8, dimnames = list(c(), c("A", "S", "T", "C",
                                                                "B", "E", "R", "D")))
stateSpace[,c("A", "D")] = 1

SShistory <- stateSpace

flipProb = function(SS){

  P_T <- case_when(  ### A = 1, P(T = 1 | A ) = 0.05
    SS[, "T"] == 1 ~ 0.05,
    SS[, "T"] == 0 ~ 1 - 0.05
  )

  P_C <- case_when( ### S = 0, P(C = 1| S = 0) = 0.01
    SS[, "C"] == 1 ~ 0.01,
    SS[, "C"] == 0 ~ 1 - 0.01
  )

  P_B <- case_when( ### S = 0, P(B = 1 | S = 0) = 0.30
    SS[, "B"] == 1 ~ 0.30,
    SS[, "B"] == 0 ~ 1 - 0.30
  )

  P_E <- case_when( ### P(E = 1 | T = 1 OR C = 1) = 1, P(E = 0 | T = 0 AND C = 0) = 1
    SS[, "E"] == 1 & (SS[, "C"] == 1 | SS[, "T"] == 1) ~ 1,
    SS[, "E"] == 0 & SS[, "C"] == 0 & SS[, "T"] == 0 ~ 1,
    TRUE ~ 0
  )

  P_D <- case_when( ### SS[, "D"] == 1 always, doesnt need to be in conditions
    SS[, "B"] == 1 & SS[, "E"] == 1 ~ 0.90,
    SS[, "B"] == 0 & SS[, "E"] == 1 ~ 0.70,
    SS[, "B"] == 1 & SS[, "E"] == 0 ~ 0.80,
    SS[, "B"] == 0 & SS[, "E"] == 0 ~ 0.10
  )

  P_R <- case_when(
    SS[, "R"] == 1 & SS[, "E"] == 1 ~ 0.98,
    SS[, "R"] == 1 & SS[, "E"] == 0 ~ 0.05,
    SS[, "R"] == 0 & SS[, "E"] == 1 ~ 1 - 0.98,
    SS[, "R"] == 0 & SS[, "E"] == 0 ~ 1 - 0.05
  )

  return(prod(P_T, P_C, P_B, P_E, P_D, P_R))
```

```r
}

tictoc::tic()
set.seed(123)
for (i in 1:260000) {

  flip <- sample(x = c( "T", "C", "B", "R"), 1)

  newStateSpace = stateSpace
  newStateSpace[,flip] = (newStateSpace[,flip] + 1) %% 2

  if(newStateSpace[, "T"] == 1 | newStateSpace[,"C"] == 1){
    newStateSpace[, "E"] = 1
  }
  if(newStateSpace[, "T"] == 0 & newStateSpace[,"C"] == 0){
    newStateSpace[, "E"] = 0
  }

  u <- runif(1)

  if(u < min(1, flipProb(newStateSpace) / flipProb(stateSpace))){
    stateSpace <- newStateSpace
  }

  if(i %% 1000 == 0 & i >= 10000){
    #print(i)
    SShistory <- rbind(SShistory, stateSpace)
  }


}
tictoc::toc()
```

```
## 279.298 sec elapsed
```

```r
SShistory <- SShistory[-1, ]

mean(SShistory[,"R"])
```

```
## [1] 0.1912351
```

Using the Metropolis-Hastings Algorithm (and seed =123), we get $P(R = 1|A = 1, S = 0, D = 1) = 0.1912$

ii)

$P(R = 1, A = 1, S = 0, D = 1) = \sum_{t=0}^{1}\sum_{c=0}^{1}\sum_{b=0}^{1}\sum_{e=0}^{1} P(T = t, C = c, B = b, E = e, R = 1, A = 1, S = 0, D = 1) =$

$\sum_{t=0}^{1}\sum_{c=0}^{1}\sum_{b=0}^{1}\sum_{e=0}^{1} P(T = 1|A = 1)P(C = c|S = 0)P(B = b|S = 0)P(E = e|T = t, C = c)|(D = 1|E = e, B = b)P(R = 1|E = e)$

$P(A = 1, S = 0, D = 1) = \sum_{t=0}^{1}\sum_{c=0}^{1}\sum_{b=0}^{1}\sum_{r=0}^{1}\sum_{e=0}^{1} P(T = t, C = c, B = b, E = e, R = r, A = 1, S = 0, D = 1) =$

$\sum_{t=0}^{1}\sum_{c=0}^{1}\sum_{b=0}^{1}\sum_{e=0}^{1}\sum_{r=0}^{1} P(T=1|A=1)P(C=c|S=0)P(B=b|S=0)P(E=e|T=t,C=c)|(D=1|E=e,B=b)P(R=r|E=e)$

```
cumProb <- 0

for (t in c(0,1)) {
  for(C in c(0,1)){
    for(b in c(0,1)){
      for(e in c(0,1)){
        for(r in c(0,1)){
          P_T <- case_when(  ### A = 1, P(T = 1 | A ) = 0.05
            t == 1 ~ 0.05,
            t == 0 ~ 1 - 0.05
          )

          P_C <- case_when( ### S = 0, P(C = 1| S = 0) = 0.01
            C == 1 ~ 0.01,
            C == 0 ~ 1 - 0.01
          )

          P_B <- case_when( ### S = 0, P(B = 1 | S = 0) = 0.30
            b == 1 ~ 0.30,
            b == 0 ~ 1 - 0.30
          )

          P_E <- case_when( ### P(E = 1 | T = 1 or C = 1) = 1, P(E = 0 | T = 0 AND C = 0) = 1
            e == 1 & (C == 1 | t == 1) ~ 1,
            e == 0 & C == 0 & t == 0 ~ 1,
            TRUE ~ 0
          )

          P_D <- case_when( ### SS[, "D"] == 1 always, doesnt need to be in conditionals
            b == 1 & e == 1 ~ 0.90,
            b == 0 & e == 1 ~ 0.70,
            b == 1 & e == 0 ~ 0.80,
            b == 0 & e == 0 ~ 0.10
          )

          P_R <- case_when(
            r == 1 & e == 1 ~ 0.98,
            r == 1 & e == 0 ~ 0.05,
            r == 0 & e == 1 ~ 1 - 0.98,
            r == 0 & e == 0 ~ 1 - 0.05
          )

          cumProb <- cumProb + P_T * P_C * P_B * P_E * P_D * P_R
        }
      }
    }
  }
}

cumProb2 <- 0
```

```r
for (t in c(0,1)) {
  for(C in c(0,1)){
    for(b in c(0,1)){
      for(e in c(0,1)){

          P_T <- case_when(  ### A = 1, P(T = 1 | A ) = 0.05
            t == 1 ~ 0.05,
            t == 0 ~ 1 - 0.05
          )

          P_C <- case_when( ### S = 0, P(C = 1| S = 0) = 0.01
            C == 1 ~ 0.01,
            C == 0 ~ 1 - 0.01
          )

          P_B <- case_when( ### S = 0, P(B = 1 | S = 0) = 0.30
            b == 1 ~ 0.30,
            b == 0 ~ 1 - 0.30
          )

          P_E <- case_when( ### P(E = 1 | T = 1 or C = 1) = 1, P(E = 0 | T = 0 AND C = 0) = 1
            e == 1 & (C == 1 | t == 1) ~ 1,
            e == 0 & C == 0 & t == 0 ~ 1,
            TRUE ~ 0
          )

          P_D <- case_when( ### SS[, "D"] == 1 always, doesnt need to be in conditionals
            b == 1 & e == 1 ~ 0.90,
            b == 0 & e == 1 ~ 0.70,
            b == 1 & e == 0 ~ 0.80,
            b == 0 & e == 0 ~ 0.10
          )

          P_R <- case_when(
            e == 1 ~ 0.98,
            e == 0 ~ 0.05
          )

          cumProb2 <- cumProb2 + P_T * P_C * P_B * P_E * P_D * P_R

      }
    }
  }
}
cumProb2 / cumProb
```

```
## [1] 0.1748745
```

And analytically, we get $P(R = 1|A = 1, S = 0, D = 1) = 0.1749$, decently close to what we got with the Metropolis-Hastings algorithm.

Let's improve by running the MH algorithm in 100 samples. Here we will use a burn in of 5,000, sample every 500, and only run for 54,500 iterations, giving a sample of 100 for each run

```r
set.seed(2)
MHAlgo <- function(iterations, burnin = 5000, sampleEvery = 500){

### initialize State Space
stateSpace <- matrix(data = 0, ncol = 8, dimnames = list(c(), c("A", "S", "T", "C",
                                                                "B", "E", "R", "D")))
stateSpace[,c("A", "D")] = 1

SShistory <- stateSpace


for (i in 1:iterations) {

  flip <- sample(x = c( "T", "C", "B", "R"), 1)

  newStateSpace = stateSpace
  newStateSpace[,flip] = (newStateSpace[,flip] + 1) %% 2

  if(newStateSpace[, "T"] == 1 | newStateSpace[,"C"] == 1){
    newStateSpace[, "E"] = 1
  }
  if(newStateSpace[, "T"] == 0 & newStateSpace[,"C"] == 0){
    newStateSpace[, "E"] = 0
  }

  u <- runif(1)

  if(u < min(1, flipProb(newStateSpace) / flipProb(stateSpace))){
    stateSpace <- newStateSpace
  }

  if(i %% sampleEvery == 0 & i >= burnin){
    SShistory <- rbind(SShistory, stateSpace)
  }


}

SShistory <- SShistory[-1, ]

return(mean(SShistory[,"R"]))
}


library(parallel)


cl <- makeCluster(detectCores())
clusterEvalQ(cl, library(dplyr))
```

```
## [[1]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
```

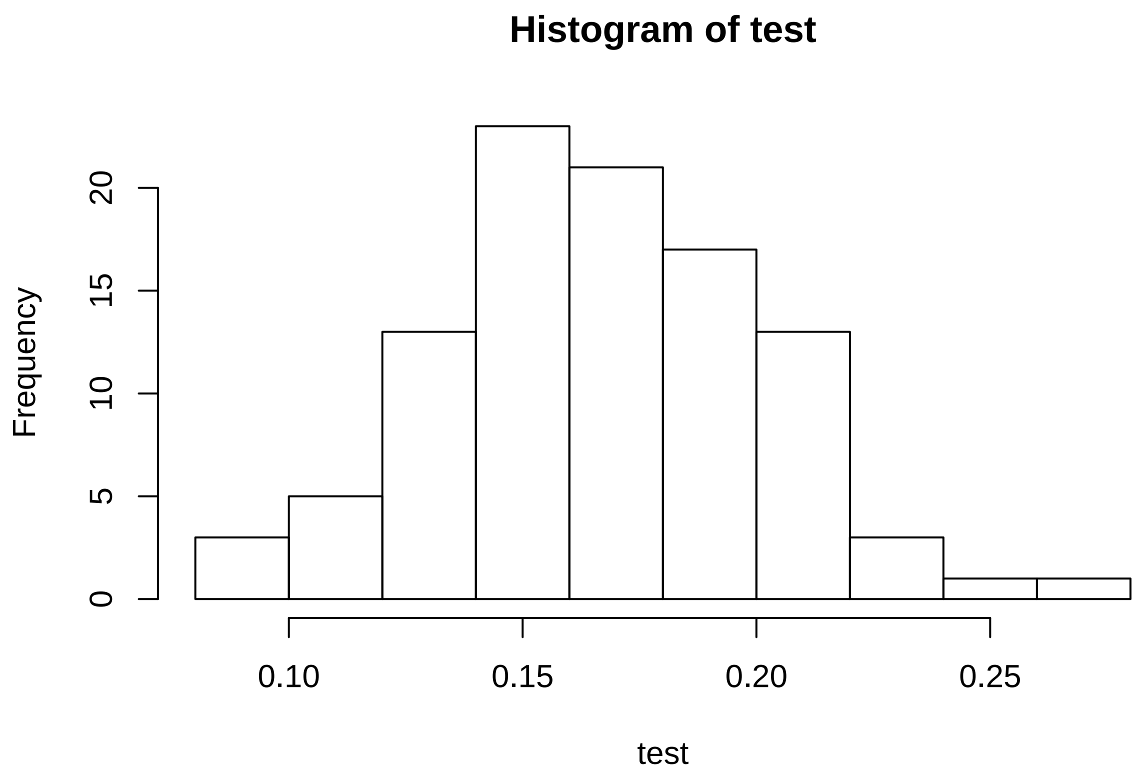
```
##
## [[2]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[3]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[4]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[5]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[6]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[7]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[8]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[9]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[10]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[11]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[12]]
## [1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
```

```r
clusterExport(cl, "flipProb")
tictoc::tic()
test <- parSapply(cl, rep(54500, 100), MHAlgo)
tictoc::toc()
```

```
## 1357.389 sec elapsed
```

```r
stopCluster(cl)

hist(test)
```




```r
mean(test)
```

```
## [1] 0.1713
```

```r
median(test)
```

```
## [1] 0.17
```

Using this method, we get much closer to the analytical solution. We have a mean for $P(R = 1|A = 1, S = 0, D = 1)$ of 0.1713

Of course, this is a much more computationally intensive way of doing this, taking about 25-30 minutes to complete

2

i)

Let V_9^+ be the set of nodes from 9 and extending to 14 and 15.

Let V_9^- bet the set of nodes from 6 to 10 to 16

We know then that $\beta_9(i) = P(V_9^+|X_9 = i)$

Then $\beta_6(j) = P(V_6^+|X_s = j) = \sum_{i=1}^m \sum_{i'}^m P(V_9^+, V_9^-, X_9 = i, X + 1- = i|X_6 = j) =$

$\sum_{i=1}^m \sum_{i'=1}^m P(X_9 = i|X_6 = j) \cdot P(X_{10} = i'|X_6 = j) \cdot P(V_9^+|X_9 = i) \cdot P(V_1 0^+|X_1 0 = i) =$

$\sum_{i=1}^m \sum_{i'=1}^m P(X_9 = i|X_6 = j) P(X_1 0 = i'|X_6 = j) = \beta_9(i)\beta_{9'}(i')$

ii)

We know $\alpha_6(j) = P(X_6 = j, V_6^-)$

Want to find $\alpha_9(i) = P(X_9, V_9^-) = P(X_9 = i, V_6^-, V_{10}^+) = \sum_{j=1}^m P(X_9 = i, X_6 = j, V_6^-, V_{10}^+) =$

$\sum_{j=1}^m P(X_6 = j, V_6^-) P(X_9 = i, V_{10}^+|X_6 = j, V_6^-)$

We know that $\alpha_6(j) = P(X_6 = j, V_6^-)$, so sub in

$= \sum_{j=1}^m \alpha_6(j) \sum_{j'=1}^m P(X_9 = i, X_{10} = j', V_{1o}^+|X_6 = j) =$

$\sum_{j=1}^m \sum_{j'=1}^m \alpha_6(j) P(X_9 = i|X_6 = j) P(X_{10} = j'|X_6 = j) P(V_{10}^+|X_{10} = j') =$

$$\sum_{j=1}^m \sum_{j'=1}^m \alpha_6(j) P(X_9 = i|X_6 = j) P(X_{10} = j'|X_6 = j) \beta_{10}(j')$$