BAYESIAN NETWORKS 3rd Exercise Block 2

Jeremy Williams

January 16th, 2018

```
#install.packages("bnlearn")
#install.packages("gRain")
#source("http://bioconductor.org/biocLite.R")
#biocLite("Rgraphviz")
#biocLite("RBGL")
#install.packages("Rcpp")

suppressMessages(suppressWarnings(library("bnlearn")))
suppressMessages(suppressWarnings(library("gRain")))
suppressMessages(suppressWarnings(library("gRbase")))
suppressMessages(suppressWarnings(library("Rgraphviz")))
suppressMessages(suppressWarnings(library("RBGL")))
suppressMessages(suppressWarnings(library("Rcpp")))


#### Model 1  KIT A


tf<-c("false","true")

# Specify the CPTs:
node.E<-cptable(~ E, values=c(2,8),levels=tf)
node.B<-cptable(~ B , values=c(9,1), levels=tf)
node.S<-cptable(~ S + E , values=c(9.9,0.1,0.3,9.7), levels=tf)
node.L<-cptable(~ S + B, values=c(1,0,0.72,0.28), levels=tf)
node.S2<-cptable(~ S2 + E , values=c(9.9,0.1,0.3,9.7), levels=tf)
node.L2<-cptable(~ S2 + B, values=c(1,0,0.72,0.28), levels=tf)

plist<-compileCPT(list(node.E,node.B,node.S,node.L,node.S2,node.L2))
plist
```

```
## CPTspec with probabilities:
##  P( E )
##  P( B )
##  P( S | E )
##  P( S | B )
##  P( S2 | E )
##  P( S2 | B )
```

```
plist$E
```

```
## E
## false  true
##   0.2   0.8
## attr(,"class")
## [1] "parray" "array"
```

```
plist$B
```

```
## B
## false  true
##   0.9   0.1
## attr(,"class")
## [1] "parray" "array"
```

```
plist$S
```

```
##        E
## S       false true
##   false  0.99 0.03
##   true   0.01 0.97
## attr(,"class")
## [1] "parray" "array"
```

```
plist$L
```

```
## NULL
```

```
plist$S2
```

```
##        E
## S2      false true
##   false  0.99 0.03
##   true   0.01 0.97
## attr(,"class")
## [1] "parray" "array"
```

```
plist$L2
```

```
## NULL
```

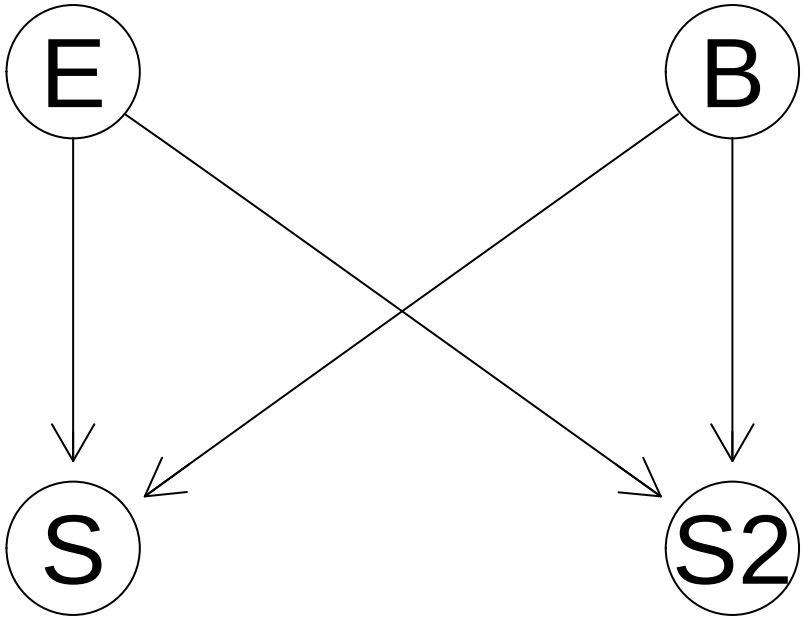
```
# Create a network of name "Norman.net", for instance:

Norman.net<-grain(plist)
summary(Norman.net)
```

```
## Independence network: Compiled: FALSE Propagated: FALSE
##  Nodes : chr [1:6] "E" "B" "S" "S" "S2" "S2"
```

```
# The graph:

plot1=plot(Norman.net)
```




```
plot1
```

```
## [1] "A graph with 4 nodes."
```

```
# We can compute the marginal probability

# of each variable

# These probabilities are EXACT!!

querygrain(Norman.net,nodes=c("E","B","S","L","S2","L2"),type="marginal")
```

```
## $E
## E
##      false       true
## 0.96124675 0.03875325
##
## $B
## B
##      false       true
## 0.91184196 0.08815804
##
## $S
## S
##      false       true
## 0.96698651 0.03301349
##
```

```
## $S2
## S2
##      false       true
## 0.96698651 0.03301349
```

```
querygrain(Norman.net,nodes=c("S","S2"), type="joint")
```

```
##        S2
## S              false        true
##   false 0.964364055 0.002622456
##   true  0.002622456 0.030391032
## attr(,"class")
## [1] "parray" "array"
```

```
#Question 1

Norman.net.1<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("false","false"))

Norman.net.1
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:6] "E" "B" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE      false
## 2    S2             TRUE      false
##   pEvidence: 0.187265
```

```
predOT<-querygrain(Norman.net.1,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##       false        true
## 0.996340348 0.003659652
```

```
#Question 2
Norman.net.2<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("true","true"))

Norman.net.2
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:6] "E" "B" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE       true
## 2    S2             TRUE       true
##   pEvidence: 0.005901
```

```
predOT<-querygrain(Norman.net.2,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##        false         true
## 0.0000265696 0.9999734304
```

```
#Question 3
```

```
Norman.net.3<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("true","false"))

Norman.net.3
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:6] "E" "B" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE       true
## 2    S2             TRUE      false
##   pEvidence: 0.000509
```

```
predOT<-querygrain(Norman.net.3,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##     false      true
## 0.0783848 0.9216152
```

```
#### Model 2  KIT B



tf<-c("false","true")

# Specify the CPTs:
node.E<-cptable(~ E, values=c(2,8),levels=tf)
node.B<-cptable(~ B , values=c(9,1), levels=tf)
node.B2<-cptable(~ B2 , values=c(9,1), levels=tf)
node.S<-cptable(~ S + E , values=c(9.9,0.1,0.3,9.7), levels=tf)
node.L<-cptable(~ S + B, values=c(1,0,0.72,0.28), levels=tf)
node.S2<-cptable(~ S2 + E , values=c(9.9,0.1,0.3,9.7), levels=tf)
node.L2<-cptable(~ S2 + B2, values=c(1,0,0.72,0.28), levels=tf)

plist<-compileCPT(list(node.E,node.B,node.B2,node.S,node.L,node.S2,node.L2))
plist
```

```
## CPTspec with probabilities:
##  P( E )
##  P( B )
##  P( B2 )
##  P( S | E )
##  P( S | B )
##  P( S2 | E )
##  P( S2 | B2 )
```

```
plist$E
```

```
## E
## false  true
##   0.2   0.8
## attr(,"class")
## [1] "parray" "array"
```

```
plist$B
```

```
## B
## false  true
##   0.9   0.1
## attr(,"class")
## [1] "parray" "array"
```

```
plist$B2
```

```
## B2
## false  true
##   0.9   0.1
## attr(,"class")
## [1] "parray" "array"
```

```
plist$S
```

```
##        E
## S       false true
##   false  0.99 0.03
##   true   0.01 0.97
## attr(,"class")
## [1] "parray" "array"
```

```
plist$L
```

```
## NULL
```

```
plist$S2
```

```
##        E
## S2      false true
##   false  0.99 0.03
##   true   0.01 0.97
## attr(,"class")
## [1] "parray" "array"
```

```
plist$L2
```

```
## NULL
```

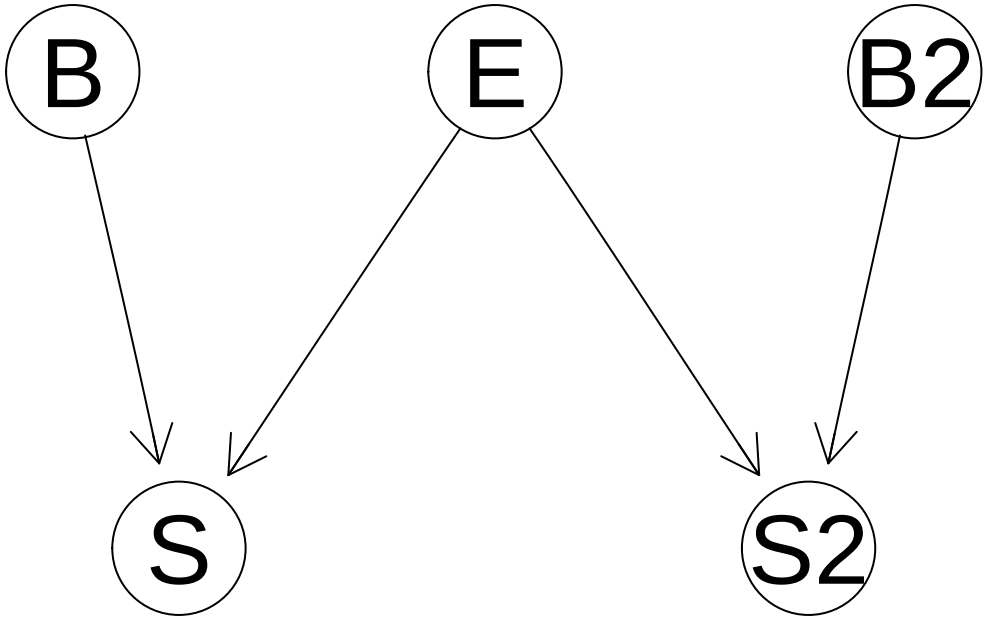
```
# Create a network of name "Norman.net", for instance:

Norman.net<-grain(plist)
summary(Norman.net)
```

```
## Independence network: Compiled: FALSE Propagated: FALSE
##  Nodes : chr [1:7] "E" "B" "B2" "S" "S" "S2" "S2"
```

```
# The graph:

plot1=plot(Norman.net)
```




```
plot1
```

```
## [1] "A graph with 5 nodes."
```

```
# We can compute the marginal probability

# of each variable

# These probabilities are EXACT!!

querygrain(Norman.net,nodes=c("E","B","B2","S","L","S2","L2"),type="marginal")
```

```
## $E
## E
##      false       true
## 0.98649101 0.01350899
##
## $B
## B
##      false       true
## 0.91962815 0.08037185
##
## $S
## S
##       false        true
## 0.993198406 0.006801594
##
```

```
## $B2
## B2
##      false       true
## 0.91962815 0.08037185
##
## $S2
## S2
##       false        true
## 0.993198406 0.006801594
```

```
querygrain(Norman.net,nodes=c("S","S2"), type="joint")
```

```
##        S2
## S             false        true
##   false 0.989538540 0.003659866
##   true  0.003659866 0.003141728
## attr(,"class")
## [1] "parray" "array"
```

```
#Question 1

Norman.net.1<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("false","false"))

Norman.net.1
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:7] "E" "B" "B2" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE      false
## 2    S2             TRUE      false
##   pEvidence: 0.185877
```

```
predOT<-querygrain(Norman.net.1,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##       false        true
## 0.996340348 0.003659652
```

```
#Question 2
Norman.net.2<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("true","true"))

Norman.net.2
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:7] "E" "B" "B2" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE       true
## 2    S2             TRUE       true
##   pEvidence: 0.000590
```

```
predOT<-querygrain(Norman.net.2,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##        false         true
## 0.0000265696 0.9999734304
```

```
#Question 3

Norman.net.3<-setEvidence(Norman.net,nodes=c("S","S2"),
                          states=c("true","false"))

Norman.net.3
```

```
## Independence network: Compiled: TRUE Propagated: TRUE
##   Nodes: chr [1:7] "E" "B" "B2" "S" "S" "S2" "S2"
##   Evidence:
##   nodes is.hard.evidence hard.state
## 1     S             TRUE       true
## 2    S2             TRUE      false
##   pEvidence: 0.000687
```

```
predOT<-querygrain(Norman.net.3,nodes=c("L","S","E","S2","L2"), type="joint")
predOT
```

```
## E
##     false      true
## 0.0783848 0.9216152
```