Bayesian Networks with R

Jeremy Williams

Decemeber 13th 2017

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

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


# First: which are the possible values of the nodes (all nodes are boolean):

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

# Specify the CPTs:
node.O<-cptable(~ O, values=c(6,4),levels=tf)
node.T<-cptable(~ T, values=c(9,1), levels=tf)
node.N<-cptable(~ N + T, values=c(9,1,2,8), levels=tf)
node.M<-cptable(~ M + O + T, values=c(7,3,4,6,4,6,2,8), levels=tf)

# Create an intermediate representation of the CPTs:

plist<-compileCPT(list(node.O,node.T,node.N,node.M))
plist
```

```
## CPTspec with probabilities:
##  P( O )
##  P( T )
##  P( N | T )
##  P( M | O T )
```

```
plist$O
```

```
## O
## false  true
##   0.6   0.4
## attr(,"class")
## [1] "parray" "array"
```

```
plist$T
```

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

```
plist$N
```

```
##        T
## N       false true
##   false   0.9  0.2
##   true    0.1  0.8
## attr(,"class")
## [1] "parray" "array"
```

```
plist$M
```

```
## , , T = false
##
##        O
## M       false true
##   false   0.7  0.4
##   true    0.3  0.6
##
## , , T = true
##
##        O
## M       false true
##   false   0.4  0.2
##   true    0.6  0.8
##
## attr(,"class")
## [1] "parray" "array"
```

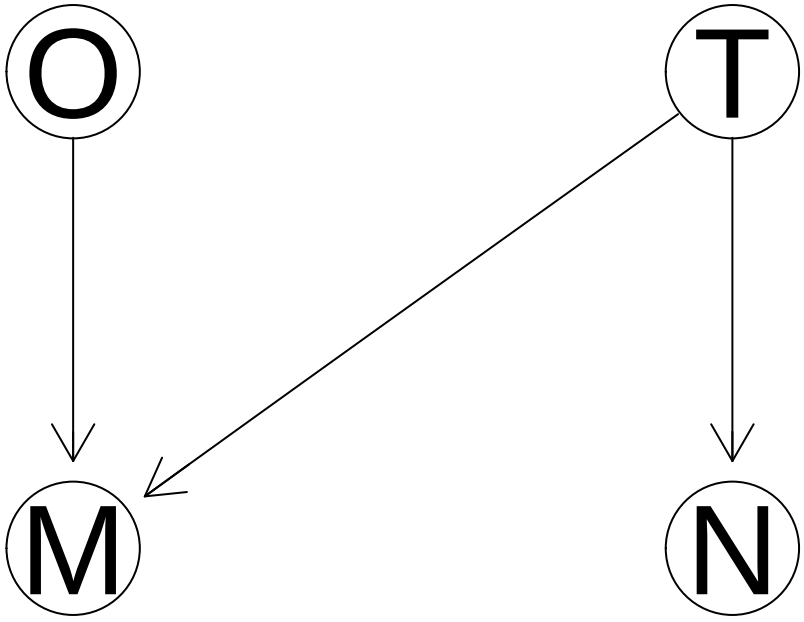
```
# Create a network of name "Norman.net", for instance:

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

```
## Independence network: Compiled: FALSE Propagated: FALSE
##  Nodes : chr [1:4] "O" "T" "N" "M"
```

```
# 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("O","T","N","M"),
           type="marginal")
```

```
## $O
## O
## false  true
##   0.6   0.4
##
## $T
## T
## false  true
##   0.9   0.1
##
## $M
## M
## false  true
## 0.554 0.446
```

```
##
## $N
## N
## false  true
##  0.83  0.17
```

```
# We can also compute the joint probability of some nodes. For
# instance:
#
querygrain(Norman.net,nodes=c("N","M"), type="joint")
```

```
##        N
## M        false   true
##   false 0.4762 0.0778
##   true  0.3538 0.0922
## attr(,"class")
## [1] "parray" "array"
```

```
# We can compute the probability of an event given an evidence.
# If evidence is "N=true", in order to compute the probability of the
# other nodes, first we add the evidence to the network and name the
# new BN Norman.net.2:

Norman.net.2<-setEvidence(Norman.net,nodes=c("N"),
                          states=c("true"))

# The marginal distributions given
# the evidence are:

marg=querygrain(Norman.net.2,nodes
                =c("O","T","M"), type="marginal")

# We can obtain the probability of the evidence used in Norman.net.2:

print(getEvidence(Norman.net.2))
```

```
##   nodes is.hard.evidence hard.state
## 1     N             TRUE       true
```

```
# If the evidence now is: N=true & M=true, we construct a new
# BN named Norman.net.3:
#

Norman.net.3<-setEvidence(Norman.net,nodes=c("N","M"),
                          states=c("true","true"))

# The marginals of the nodes O and T when the evidence is
# N=true & M=true, are:
#
nodos=c("O","T")
marg3=querygrain(Norman.net.3,nodes=nodos, type="marginal")

# The following gives the most probable value for each variable,
# given the evidences N=true & M=true.
# It does not imply that the jointly configuration with these values is
# the most probable!!
```

```
#

prediction=NULL
confidence.level=NULL
Node=NULL
for (i in 1:length(nodos))
{
  prediction[i]<-tf[which.max(marg3[[i]])]
  confidence.level[i]<-max(marg3[[i]])
}
Node<-as.data.frame(cbind(nodos,prediction,confidence.level))
Node
```

```
##   nodos prediction  confidence.level
## 1     O       true 0.511930585683297
## 2     T       true  0.59002169197397
```

```
# The following gives the most probable value for each variable,
# given the evidences N=true & M=true.
# It does not imply that the jointly configuration with these values is
# the most probable!!
#

# Instead, we can obtain the joint probability distribution of nodes O
# and T given the evidences N=true & M=true, and see which is the
# configuration which maximizes the probability:
#

predOT<-querygrain(Norman.net.3,nodes=c("O","T"), type="joint")
predOT
```

```
##        T
## O           false      true
##   false 0.1757050 0.3123644
##   true  0.2342733 0.2776573
```

```
which.max(predOT)
```

```
## [1] 3
```

```
# Therefore, the configuration that maximizes the joint probability
# distribution of O and T, given the evidences N=true & M=true is:
# O=false & T=true, with a confidence level of 0.3123644
```