Introducción al uso de redes bayesianas con R

Daniel Ranchal Parrado

12/02/2022

```
library(bnlearn)
library(gRain)
```

```
## Loading required package: gRbase
```

```
##
## Attaching package: 'gRbase'
```

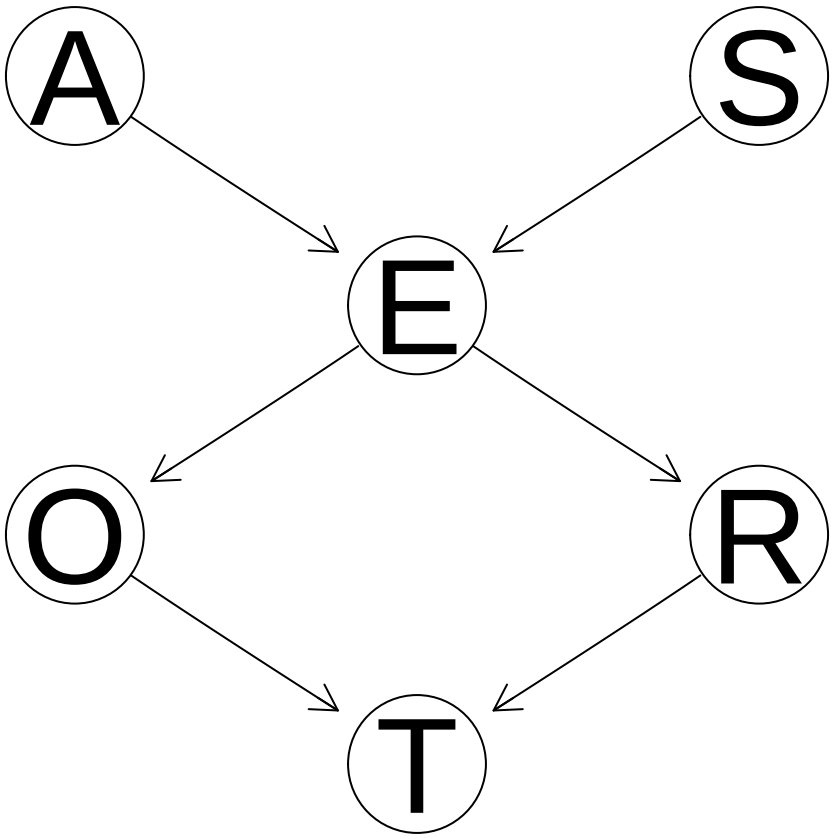
```
## The following objects are masked from 'package:bnlearn':
##
##     ancestors, children, parents
```

Ejercicio 1

Hacer un script en R para construir la red bayesiana (DAG y tablas de probabilidad) usada como ejemplo en esta práctica. El script debe contener también los comandos para dibujar la red bayesiana y para hacer las consultas (d-separación, probabilidad a posteriori) que se han explicado en la sección 7.3 usando el método de inferencia exacto y los dos aproximados.

```
dag <- empty.graph(
    nodes = c("A", "S", "E", "O", "R", "T")
)

modelstring(dag) <- "[A][S][E|A:S][O|E][R|E][T|O:R]"
graphviz.plot(dag)
```




```
A.st <- c("young", "adult", "old")
S.st <- c("M", "F")
E.st <- c("high", "uni")
O.st <- c("emp", "self")
R.st <- c("small", "big")
T.st <- c("car", "train", "other")


A.prob <- array(c(0.30, 0.50, 0.20), dim=3, dimnames = list(A = A.st))
S.prob <- array(c(0.60, 0.40), dim=2, dimnames = list(S = S.st))
O.prob <- array(
    c(
        0.96, 0.04,
        0.92, 0.08
    ),
    dim=c(2,2),
    dimnames = list(O = O.st, E = E.st)
)
R.prob <- array(
    c(
        0.25, 0.75,
        0.20, 0.80
    ),
    dim = c(2,2),
    dimnames = list(R = R.st, E = E.st)
)
```

```
E.prob <- array(
    c(
        0.75, 0.25, 0.72,
        0.28, 0.88, 0.12,
        0.64, 0.36, 0.70,
        0.30, 0.90, 0.10
    ),
    dim=c(2, 3, 2),
    dimnames = list(E = E.st, A = A.st, S = S.st)
)

T.prob <- array(
    c(
        0.48, 0.42,
        0.10, 0.56,
        0.36, 0.08,
        0.58, 0.24,
        0.18, 0.70,
        0.21, 0.09
    ),
    dim=c(3, 2, 2),
    dimnames = list(`T` = T.st, O = O.st, R = R.st)
)

cpt <- list(
    A=A.prob,
    S=S.prob,
    E=E.prob,
    O=O.prob,
    R=R.prob,
    `T`=T.prob
)

bn <- custom.fit(dag, cpt)
bn
```

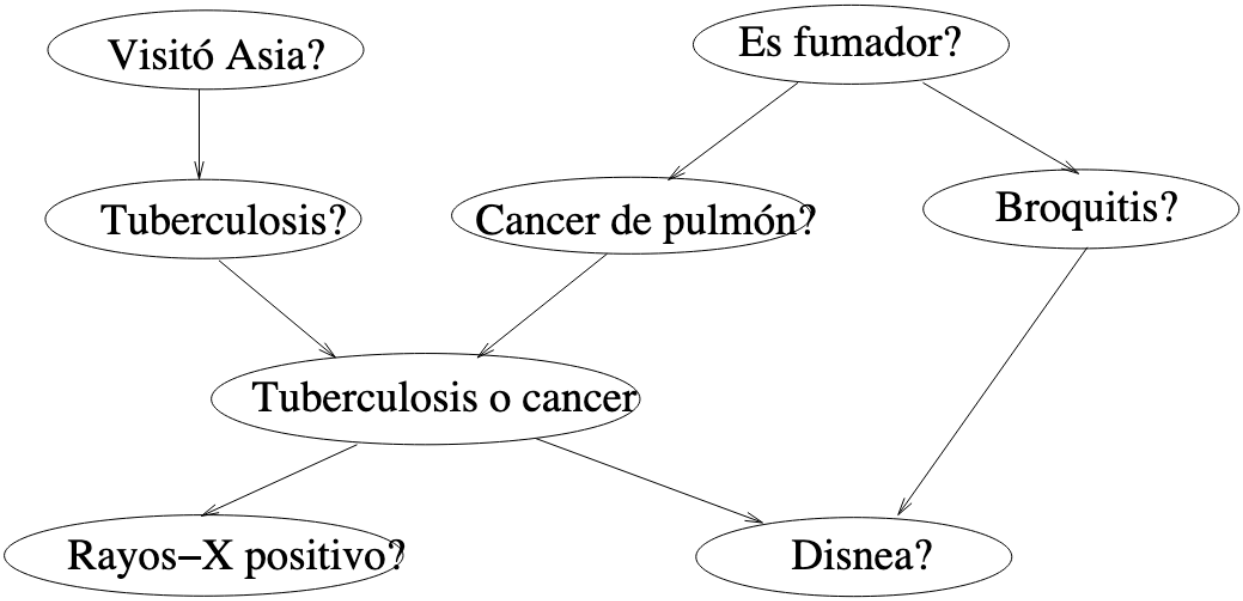
```
##
##   Bayesian network parameters
##
##   Parameters of node A (multinomial distribution)
##
## Conditional probability table:
##  A
## young adult   old
##   0.3   0.5   0.2
##
##   Parameters of node S (multinomial distribution)
##
## Conditional probability table:
##  S
##   M   F
## 0.6 0.4
##
##   Parameters of node E (multinomial distribution)
```

```
##
## Conditional probability table:
##
## , , S = M
##
##       A
## E      young adult  old
##   high  0.75  0.72 0.88
##   uni   0.25  0.28 0.12
##
## , , S = F
##
##       A
## E      young adult  old
##   high  0.64  0.70 0.90
##   uni   0.36  0.30 0.10
##
##
##   Parameters of node O (multinomial distribution)
##
## Conditional probability table:
##
##       E
## O      high  uni
##   emp  0.96 0.92
##   self 0.04 0.08
##
##   Parameters of node R (multinomial distribution)
##
## Conditional probability table:
##
##        E
## R       high  uni
##   small 0.25 0.20
##   big   0.75 0.80
##
##   Parameters of node T (multinomial distribution)
##
## Conditional probability table:
##
## , , R = small
##
##        O
## T        emp self
##   car   0.48 0.56
##   train 0.42 0.36
##   other 0.10 0.08
##
## , , R = big
##
##        O
## T        emp self
##   car   0.58 0.70
##   train 0.24 0.21
```

```
##   other 0.18 0.09
```

Ejercicio 2

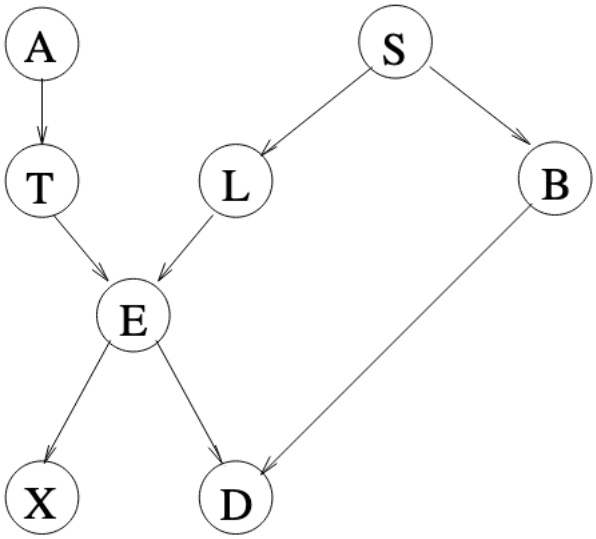
Construye un script en R para construir la red Asia (Dag y tablas de probabilidad)



Usaremos los siguientes nombres para las variables:

- Variable Visitó Asia: A
- Variable Es fumador: S
- Variable Tuberculosis: T
- Variable Cáncer de pulmón: L
- Variable Tuberculosis o cáncer de pulmón: E
- Variable Bronquitis: B
- Variable Rayos X positivo: X
- Variable Disnea: D

Con estos nombres, la red bayesiana quedaría de la siguiente forma:

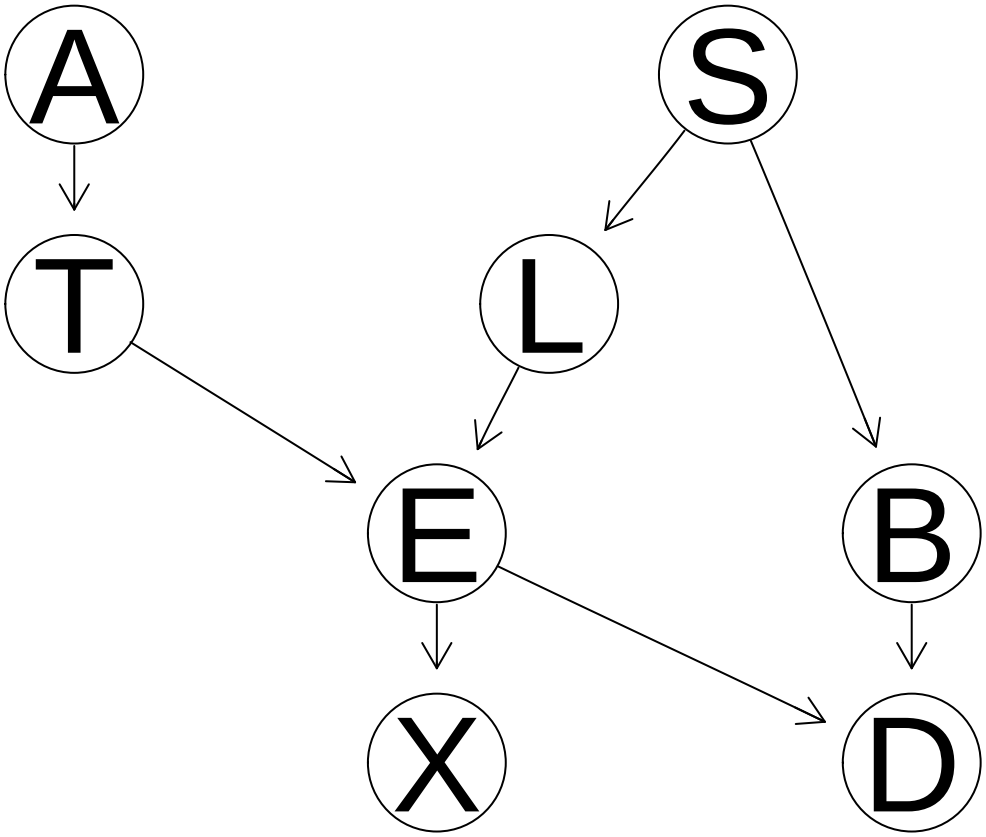


Todas las variables tienen dos estados: yes y no. Los valores de probabilidad son los siguientes:

- P(A=yes) = 0.1
- P(T=yes|A=yes) = 0.05
- P(T=yes|A=no) = 0.01
- P(S=yes) = 0.5
- P(L=yes|S=yes) = 0.1
- P(L=yes|S=no) = 0.01
- P(B=yes|S=yes) = 0.6
- P(B=yes|S=no) = 0.3
- P(E=yes|L=yes, T=yes) = 1
- P(E=yes|L=yes, T=no) = 1
- P(E=yes|L=no, T=yes) = 1
- P(E=yes|L=no, T=no) = 0
- P(X=yes|E=yes) = 0.98
- P(X=yes|E=no) = 0.05
- P(D=yes|E=yes, B=yes) = 0.9
- P(E=yes|E=yes, B=no) = 0.7
- P(E=yes|E=no, B=yes) = 0.8
- P(E=yes|E=no, B=no) = 0.1

```
dag2 <- empty.graph(
  nodes = c("A", "S", "T", "L", "E", "B", "X", "D")
)

modelstring(dag2) <- "[A][S][T|A][L|S][B|S][E|T:L][X|E][D|E:B]"
graphviz.plot(dag2)
```



```r
state <- c("yes", "no")

A.probability <- array(
  c(0.1, 0.9),
  dim = 2,
  dimnames = list(A = state)
)

S.probability <- array(
  c(0.5, 0.5),
  dim = 2,
  dimnames = list(S = state)
)

T.probability <- array(
  c(
    0.05, 0.95,
    0.01, 0.99
  ),
  dim = c(2, 2),
  dimnames = list(`T` = state, A = state)
)

L.probability <- array(
  c(
    0.1, 0.9,
    0.01, 0.99
  ),
  dim = c(2, 2),
  dimnames = list(L = state, S = state)
)

B.probability <- array(
  c(
    0.6, 0.4,
    0.3, 0.7
  ),
  dim = c(2, 2),
  dimnames = list(B = state, S = state)
)

E.probability <- array(
  c(
    1, 0,
    1, 0,
    1, 0,
    0, 1
  ),
  dim = c(2, 2, 2),
  dimnames = list(E = state, L = state, `T` = state)
)

X.probability <- array(
```

```
  c(
    0.98, 0.02,
    0.05, 0.95
  ),
  dim = c(2, 2),
  dimnames = list(X = state, E = state)
)

D.probability <- array(
  c(
    0.9, 0.1,
    0.8, 0.2,
    0.7, 0.3,
    0.1, 0.9
  ),
  dim = c(2, 2, 2),
  dimnames = list(D = state, E = state, B = state)
)

cpt2 <- list(
    A=A.probability,
    S=S.probability,
    `T`=T.probability,
    L=L.probability,
    B=B.probability,
    E=E.probability,
    X=X.probability,
    D=D.probability
)

bn2 <- custom.fit(dag2, cpt2)
junction2 <- compile(as.grain(bn2))
bn2
```

```
##
##   Bayesian network parameters
##
##   Parameters of node A (multinomial distribution)
##
## Conditional probability table:
##  A
## yes  no
## 0.1 0.9
##
##   Parameters of node S (multinomial distribution)
##
## Conditional probability table:
##  S
## yes  no
## 0.5 0.5
##
##   Parameters of node T (multinomial distribution)
##
## Conditional probability table:
```

```
##
##      A
## T      yes   no
##   yes 0.05 0.01
##   no  0.95 0.99
##
##   Parameters of node L (multinomial distribution)
##
## Conditional probability table:
##
##      S
## L      yes   no
##   yes 0.10 0.01
##   no  0.90 0.99
##
##   Parameters of node E (multinomial distribution)
##
## Conditional probability table:
##
## , , L = yes
##
##      T
## E     yes no
##   yes   1  1
##   no    0  0
##
## , , L = no
##
##      T
## E     yes no
##   yes   1  0
##   no    0  1
##
##
##   Parameters of node B (multinomial distribution)
##
## Conditional probability table:
##
##      S
## B     yes  no
##   yes 0.6 0.3
##   no  0.4 0.7
##
##   Parameters of node X (multinomial distribution)
##
## Conditional probability table:
##
##      E
## X      yes   no
##   yes 0.98 0.05
##   no  0.02 0.95
##
##   Parameters of node D (multinomial distribution)
##
```

```
## Conditional probability table:
##
## , , B = yes
##
##      E
## D     yes  no
##   yes 0.9 0.8
##   no  0.1 0.2
##
## , , B = no
##
##      E
## D     yes  no
##   yes 0.7 0.1
##   no  0.3 0.9
```

El script debe también hacer consultas para obtener lo siguiente:

1. **Tablas de probabilidad marginal de padecer tuberculosis, de padecer cáncer de pulmón y de padecer bronquitis. Usa el método exacto y los dos aproximados.**

```
exercise1.exact_method <- querygrain(
  junction2,
  nodes = c("T", "L", "B"),
  type = "marginal"
)

exercise1.t.approximated.ls <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = TRUE,
  method = "ls"
)
exercise1.l.approximated.ls <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = TRUE,
  method = "ls"
)
exercise1.b.approximated.ls <- cpquery(
  bn2,
  event = (B == "yes"),
  evidence = TRUE,
  method = "ls"
)

exercise1.approximated_method.ls <- list(
  `T` = array(
    c(exercise1.t.approximated.ls, abs(exercise1.t.approximated.ls - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise1.l.approximated.ls, abs(exercise1.l.approximated.ls - 1)),
    dim = 2,
```

```
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise1.b.approximated.ls, abs(exercise1.b.approximated.ls - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)

exercise1.t.approximated.lw <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = TRUE,
  method = "lw"
)
exercise1.l.approximated.lw <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = TRUE,
  method = "lw"
)
exercise1.b.approximated.lw <- cpquery(
  bn2,
  event = (B == "yes"),
  evidence = TRUE,
  method = "lw"
)

exercise1.approximated_method.lw <- list(
  `T` = array(
    c(exercise1.t.approximated.lw, abs(exercise1.t.approximated.lw - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise1.l.approximated.lw, abs(exercise1.l.approximated.lw - 1)),
    dim = 2,
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise1.b.approximated.lw, abs(exercise1.b.approximated.lw - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)

exercise1.exact_method
```

```
## $T
## T
##   yes    no
## 0.014 0.986
##
## $L
```

```
## L
##   yes    no
## 0.055 0.945
##
## $B
## B
##  yes   no
## 0.45 0.55
```

```
exercise1.approximated_method.ls
```

```
## $T
## T
##    yes     no
## 0.0128 0.9872
##
## $L
## L
##    yes     no
## 0.0548 0.9452
##
## $B
## B
##    yes     no
## 0.4494 0.5506
```

```
exercise1.approximated_method.lw
```

```
## $T
## T
##   yes    no
## 0.017 0.983
##
## $L
## L
##   yes    no
## 0.055 0.945
##
## $B
## B
##    yes     no
## 0.4654 0.5346
```

2. **Tablas de probabilidad marginal a posteriori de padecer cada una de las tres enfermedades anteriores dado que se sabe que el paciente visitó Asia. Usa el método exacto y los dos aproximados.**

```
exercise2.evidence <- setEvidence(
  junction2,
  nodes = "A",
  states = "yes"
)
exercise2.exact_method <- querygrain(
  exercise2.evidence,
  nodes = c("T", "L", "B"),
  type = "marginal"
```

```
)

exercise2.t.approximated.ls <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = (A == "yes"),
  method = "ls"
)
exercise2.l.approximated.ls <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = (A == "yes"),
  method = "ls"
)
exercise2.b.approximated.ls <- cpquery(
  bn2,
  event = (B == "yes"),
  evidence = (A == "yes"),
  method = "ls"
)

exercise2.approximated_method.ls <- list(
  `T` = array(
    c(exercise2.t.approximated.ls, abs(exercise2.t.approximated.ls - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise2.l.approximated.ls, abs(exercise2.l.approximated.ls - 1)),
    dim = 2,
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise2.b.approximated.ls, abs(exercise2.b.approximated.ls - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)

exercise2.t.approximated.lw <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = list(A = "yes"),
  method = "lw"
)
exercise2.l.approximated.lw <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = list(A = "yes"),
  method = "lw"
)
exercise2.b.approximated.lw <- cpquery(
  bn2,
```

```
  event = (B == "yes"),
  evidence = list(A = "yes"),
  method = "lw"
)

exercise2.approximated_method.lw <- list(
  `T` = array(
    c(exercise2.t.approximated.lw, abs(exercise2.t.approximated.lw - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise2.l.approximated.lw, abs(exercise2.l.approximated.lw - 1)),
    dim = 2,
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise2.b.approximated.lw, abs(exercise2.b.approximated.lw - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)

exercise2.exact_method
```

```
## $T
## T
##  yes   no
## 0.05 0.95
##
## $L
## L
##   yes    no
## 0.055 0.945
##
## $B
## B
##  yes   no
## 0.45 0.55
```

```
exercise2.approximated_method.ls
```

```
## $T
## T
##        yes         no
## 0.03891051 0.96108949
##
## $L
## L
##        yes         no
## 0.05394191 0.94605809
##
## $B
## B
```

```
##       yes        no
## 0.4501992 0.5498008
```

```
exercise2.approximated_method.lw
```

```
## $T
## T
##   yes    no
## 0.048 0.952
##
## $L
## L
##    yes     no
## 0.0576 0.9424
##
## $B
## B
##    yes     no
## 0.4542 0.5458
```

3. **Tablas de probabilidad marginal a posteriori de padecer cada una de las tres enfermedades anteriores dado que se sabe que el paciente visitó Asia, y tiene disnea. Usa el método exacto y los dos aproximados**

```
exercise3.evidence <- setEvidence(
  junction2,
  nodes = c("A", "D"),
  states = c("yes", "yes")
)
exercise3.exact_method <- querygrain(
  exercise3.evidence,
  nodes = c("T", "L", "B"),
  type = "marginal"
)

exercise3.t.approximated.ls <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = (A == "yes") & (D == "yes"),
  method = "ls"
)
exercise3.l.approximated.ls <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = (A == "yes") & (D == "yes"),
  method = "ls"
)
exercise3.b.approximated.ls <- cpquery(
  bn2,
  event = (B == "yes"),
  evidence = (A == "yes") & (D == "yes"),
  method = "ls"
)

exercise3.approximated_method.ls <- list(
  `T` = array(
```

```
    c(exercise3.t.approximated.ls, abs(exercise3.t.approximated.ls - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise3.l.approximated.ls, abs(exercise3.l.approximated.ls - 1)),
    dim = 2,
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise3.b.approximated.ls, abs(exercise3.b.approximated.ls - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)

exercise3.t.approximated.lw <- cpquery(
  bn2,
  event = (`T` == "yes"),
  evidence = list(A = "yes", D = "yes"),
  method = "lw"
)
exercise3.l.approximated.lw <- cpquery(
  bn2,
  event = (L == "yes"),
  evidence = list(A = "yes", D = "yes"),
  method = "lw"
)
exercise3.b.approximated.lw <- cpquery(
  bn2,
  event = (B == "yes"),
  evidence = list(A = "yes", D = "yes"),
  method = "lw"
)

exercise3.approximated_method.lw <- list(
  `T` = array(
    c(exercise3.t.approximated.lw, abs(exercise3.t.approximated.lw - 1)),
    dim = 2,
    dimnames = list(`T` = state)
  ),
  L = array(
    c(exercise3.l.approximated.lw, abs(exercise3.l.approximated.lw - 1)),
    dim = 2,
    dimnames = list(L = state)
  ),
  B = array(
    c(exercise3.b.approximated.lw, abs(exercise3.b.approximated.lw - 1)),
    dim = 2,
    dimnames = list(B = state)
  )
)
```

```
exercise3.exact_method
```

```
## $T
## T
##       yes        no
## 0.08775096 0.91224904
##
## $L
## L
##       yes        no
## 0.09952515 0.90047485
##
## $B
## B
##       yes        no
## 0.8114021 0.1885979
```

```
exercise3.approximated_method.ls
```

```
## $T
## T
##       yes        no
## 0.0776699 0.9223301
##
## $L
## L
##       yes        no
## 0.1141553 0.8858447
##
## $B
## B
##       yes        no
## 0.7783251 0.2216749
```

```
exercise3.approximated_method.lw
```

```
## $T
## T
##       yes        no
## 0.09523599 0.90476401
##
## $L
## L
##       yes        no
## 0.1018841 0.8981159
##
## $B
## B
##       yes        no
## 0.8110626 0.1889374
```

4. **Tabla de probabilidad conjunta de padecer cáncer de pulmón y bronquitis. Indica cual es la probabilidad de la configuración más probable para las variables de estas dos enfermadades.**

```
exercise4.LxB <- querygrain(
  junction2,
```

```
  nodes = c("L", "B"),
  type = "joint"
)
exercise4.LxB
```

```
##      B
## L        yes     no
##   yes 0.0315 0.0235
##   no  0.4185 0.5265
```

La probabilidad de la configuración más probable es 0.5265, es decir, que no se padezca cáncer de pulmón ni bronquitis.

5. **Tabla de probabilidad conjunta de padecer cáncer de pulmón y bronquitis dado que el paciente no visitó Asia y el paciente es fumador. Indica cual es la probabilidad de la configuración más probable para las variables de estas dos enfermadades cuando el paciente no visitó Asia y el paciente es fumador.**

```
exercice5.evidence <- setEvidence(
  junction2,
  nodes = c("A", "S"),
  states = c("no", "yes")
)

exercise5.LxB <- querygrain(
  exercice5.evidence,
  nodes = c("L", "B"),
  type = "joint"
)
exercise5.LxB
```

```
##      B
## L      yes   no
##   yes 0.06 0.04
##   no  0.54 0.36
```

La probabilidad de la configuración más probable es 0.54, es decir, que no se padezca cáncer de pulmón pero si se padece de bronquitis.

6. **Si no se conoce el valor de ninguna variable, ¿qué comando(s) podemos usar para saber si visitar Asia influye en tener cáncer de pulmón o no?**

Se tiene que utilizar el comando dsep para ver si las dos variables son independientes. También se puede comprobar calculando la probabilidad marginal de padecer cáncer de pulmón dado el visitar asia y no visitarlo.

```
dsep(bn2, x = "A", y = "L")
```

```
## [1] TRUE
```

```
exercise6.evidence <- setEvidence(
  junction2,
  nodes = "A",
  states = "yes"
)
querygrain(exercise6.evidence, nodes = "L")
```

```
## $L
## L
##   yes    no
```

```
## 0.055 0.945
```

```
querygrain(junction2, nodes = "L")
```

```
## $L
## L
##   yes    no
## 0.055 0.945
```

Como están d-separados, visitar Asia no influye en tener cáncer de pulmón. Además, como las probabilidades marginales son iguales, se deduce que visitar asia no influye en padecer cáncer del pulmón.

7. **Sabiendo que el paciente es fumador, ¿qué comando(s) podemos usar para saber si dar positivo en una prueba de rayos X, puede influir en tener bronquitis?**

También se tiene que utilizar el comando dsep, para ver si las dos variables son independientes y en este caso el nodo de ser fumador se especifica como evidencia. También se puede comprobar calculando la probabilidad marginal de padecer bronquitis dado el ser fumador y dar positivo en la prueba de rayos x con respecto a ser fumador y dar negativo en la prueba.

```
dsep(bn2, x = "X", y = "B", z = "S")
```

```
## [1] TRUE
```

```
exercise7.evidence.1 <- setEvidence(
  junction2,
  nodes = c("S", "X"),
  states = c("yes", "yes")
)
querygrain(exercise7.evidence.1, nodes = "B")
```

```
## $B
## B
## yes  no
## 0.6 0.4
```

```
exercise7.evidence.2 <- setEvidence(
  junction2,
  nodes = "S",
  states = "yes"
)
querygrain(exercise7.evidence.2, nodes = "B")
```

```
## $B
## B
## yes  no
## 0.6 0.4
```

Como están d-separados conociendo el valor de S (ser fumador), dar positivo en una prueba de rayo X no influye en tener bronquitis. Además, como las probabilidades marginales son iguales, se deduce que dado ser fumador, dar positivo en una prueba de rayos X no influye en padecer bronquitis.

8. **La variable Tuberculosis o cáncer tiene asociada una distribución de probabilidad condicional que es una función determinista de los padres, concretamente una puerta or. Sabiendo que el paciente tiene tuberculosis, obtén las tablas de probabilidad marginal de padecer cáncer de pulmón y las tablas de probabilidad marginal a posteriori de padecer cáncer de pulmón si además se sabe que el paciente da positivo en la prueba de rayos X. ¿Puedes comentar si ha influido en la probabilidad de padecer cáncer de pulmón, el conocer que la prueba de rayos X ha sido positiva?**

```r
exercise8.evidence.1 <- setEvidence(
  junction2,
  nodes = "T",
  states = "yes"
)
querygrain(exercise8.evidence.1, nodes = "L")
```

```
## $L
## L
##   yes    no
## 0.055 0.945
```

```r
exercise8.evidence.2 <- setEvidence(
  junction2,
  nodes = c("T", "X"),
  states = c("yes", "yes")
)
querygrain(exercise8.evidence.2, nodes = "L")
```

```
## $L
## L
##   yes    no
## 0.055 0.945
```

Como se puede observar, las probabilidades marginales no cambian, por lo que se puede afirmar que el conocer que la prueba de rayos X haya sido positiva no influye en la probabilidad de padecer cáncer de pulmón. Esto se debe principalmente al nodo E, que es un nodo determinista que depende de los nodos T y L. Como se da el valor de T, E "bloquea" el camino entre X y L, y por lo tanto, el nodo X no influye en el nodo L.

Ejercicio 3

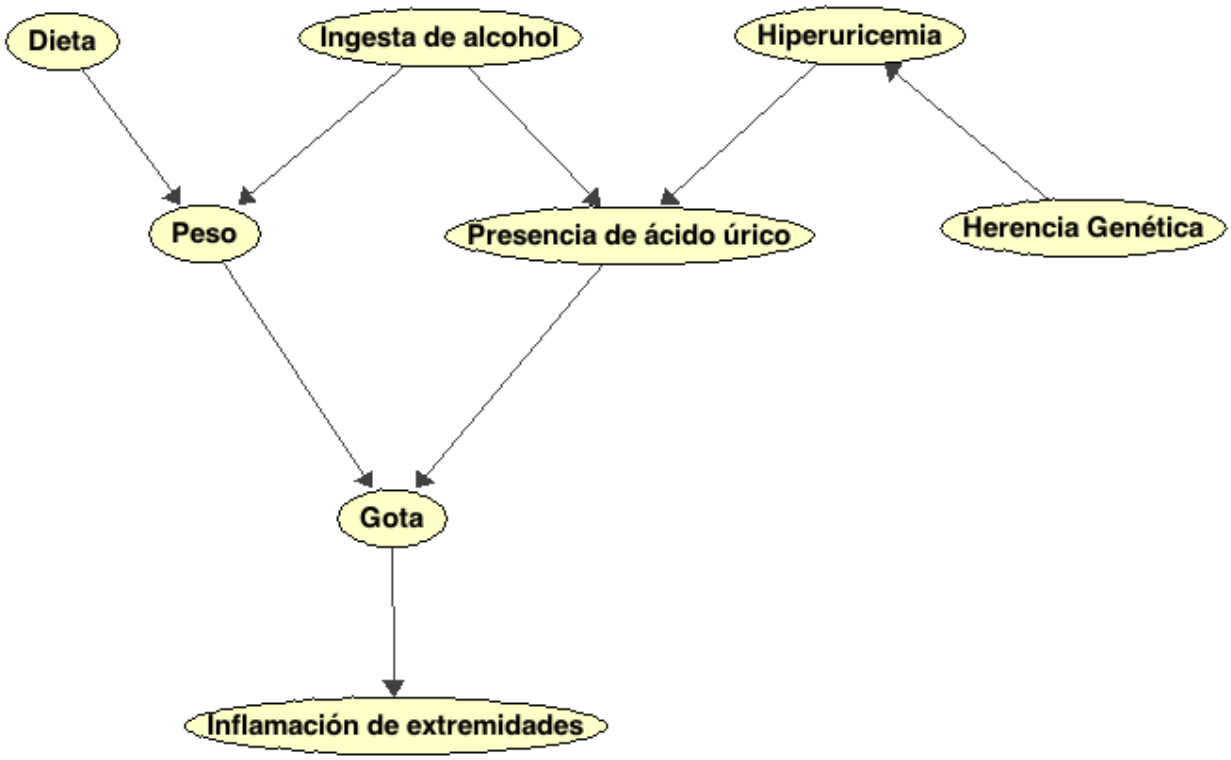
Como trabajo opcional de esta parte de la asignatura, se debe entregar un script en R para construir una red bayesiana de tamaño pequeño (mínimo 7 nodos). Podría entregarse la misma red que se haga para el trabajo a entregar a Manuel Gómez. Debe entregarse también un pequeño informe con la descripción del problema, y la descripción de la red (parte cualitativa y cuantitativa).

Se ha considerado la enfermedad de la gota para la construcción de la red bayesiana. Las causas de la gota (G) se debe principalmente a la alta presencia de ácido úrico (AU) y a un peso (P) elevado. La alta presencia de ácido úrico está relacionado con la ingesta de alcohol (A) pero también a sufrir hiperuricemia (H), que se puede heredar genéticamente (HG). Mientras que el peso (P) está relacionado con la dieta (D) que tenga el individuo y al alcohol (A) que se consuma. Por otro lado, tener gota (G) supone tener inflamación en diferentes extremidades (I).

Los estados de cada variable son los siguientes:

- Dieta (D): buena y mala
- Ingesta de alcohol (A): mucho, poco y nada
- Herencia Genética (HG): Sí y no
- Hiperuricemia (H): Sí y no
- Peso (P): Sobrepeso y saludable
- Presencia de ácido úrico (AU): mucho y poco
- Gota (G): Sí y no
- Inflamación de extremidades (I): Sí y no

A partir de esta descripción, se ha definido el grafo dirigido Acíclico que se muestra en la siguiente figura



Como se puede observar, las variables sin padres y por lo tanto independientes respecto a las demás son Dieta, Ingesta de Alcohol y Herencia genética. El nodo de herencia genética influye sobre la hiperuricemia, ya que esta se puede heredar genéticamente. Por otro lado, el nodo de la hiperuricemia e ingesta de alcohol influyen sobre la presencia de ácido úrico en el organismo. A su vez, el nodo de ingesta de alcohol y dieta influyen sobre el nodo peso. Después, los nodos peso y presencia de ácido úrico influyen en el nodo Gota. Finalmente, el nodo gota influye sobre el nodo Inflamación de extremidades. Los nodos que se encuentran por encima del nodo Gota son causas mientras que el nodo que se encuentra justamente después del nodo gota es un síntoma. A continuación se muestran las distribuciones de probabilidad para cada nodo.

$P(D)$	$buena$	$mala$
	0.8	0.2

Table 1: Distribución de probabilidad del nodo Dieta

$P(A)$	$mucho$	$poco$	$nada$
	0.2	0.6	0.2

Table 2: Distribución de probabilidad del nodo Ingesta de alcohol

$P(HG)$	Si	No
	0.2	0.8

Table 3: Distribución de probabilidad del nodo Herencia Genética

$P(H\|HG)$	Si	No
Si	0.95	0.05
No	0.1	0.9

Table 4: Distribución de probabilidad del nodo Hiperuricemia

$P(I\|G)$	Si	No
Si	0.9	0.1
No	0.2	0.8

Table 5: Distribución de probabilidad del nodo Inflamación de extremidades

$P(P\|D, A)$	$Sobrepeso$	$Saludable$
$buena, mucho$	0.3	0.7
$buena, poco$	0.1	0.9
$buena, nada$	0.05	0.95
$mala, mucho$	0.95	0.05
$mala, poco$	0.7	0.3
$mala, nada$	0.6	0.4

Table 6: Distribución de probabilidad del nodo Peso

$P(AU\|H, A)$	$mucho$	$poco$
$si, mucho$	0.98	0.02
$si, poco$	0.9	0.1
$si, nada$	0.85	0.15
$no, mucho$	0.6	0.4
$no, poco$	0.2	0.8
$no, nada$	0.02	0.98

Table 7: Distribución de probabilidad del nodo Presencia de ácido úrico

$P(G\|AU, P)$	Si	No
$mucho, sobrepeso$	0.98	0.02
$mucho, saludable$	0.95	0.05
$poco, sobrepeso$	0.3	0.7
$poco, saludable$	0.2	0.8

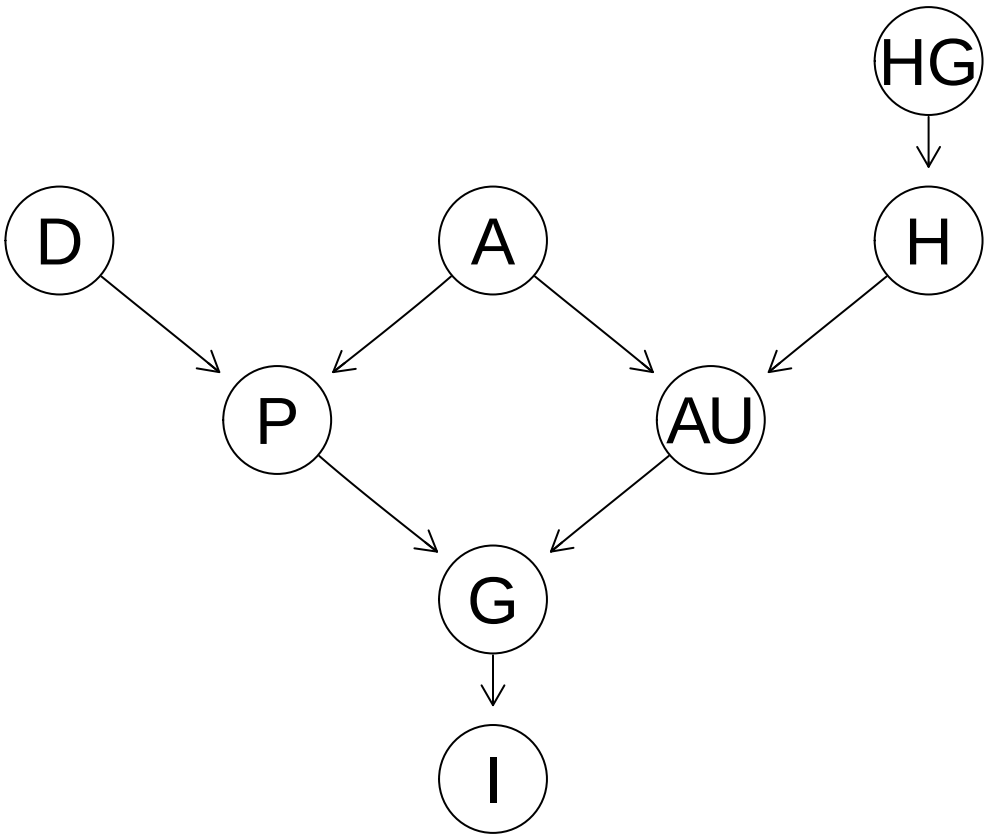
Table 8: Distribución de probabilidad del nodo Gota

El script debe hacer lo siguiente:

- **Construir el DAG de la red bayesiana**

```
dag3 <- empty.graph(
  nodes = c("D", "A", "HG", "H", "P", "AU", "G", "I")
)

modelstring(dag3) <- "[D][A][HG][H|HG][P|D:A][AU|A:H][G|P:AU][I|G]"
graphviz.plot(dag3)
```



- **Crear las tablas de probabilidad**

```
optional.d.state <- c("buena", "mala")
optional.a.state <- c("mucho", "poco", "nada")
optional.hg.state <- c("si", "no")
optional.h.state <- c("si", "no")
optional.p.state <- c("sobrepeso", "saludable")
optional.au.state <- c("mucho", "poco")
optional.g.state <- c("si", "no")
optional.i.state <- c("si", "no")

optional.d.probability <- array(
  c(0.8, 0.2),
  dim = 2,
  dimnames = list(D = optional.d.state)
)

optional.a.probability <- array(
  c(0.2, 0.6, 0.2),
  dim = 3,
  dimnames = list(A = optional.a.state)
)

optional.hg.probability <- array(
  c(0.2, 0.8),
  dim = 2,
  dimnames = list(HG = optional.hg.state)
```

```
)

optional.h.probability <- array(
  c(
    0.95, 0.05,
    0.1, 0.9
  ),
  dim = c(2, 2),
  dimnames = list(H = optional.h.state, HG = optional.hg.state)
)

optional.p.probability <- array(
  c(
    0.3, 0.7,
    0.95, 0.05,
    0.1, 0.9,
    0.7, 0.3,
    0.05, 0.95,
    0.6, 0.4
  ),
  dim = c(2, 2, 3),
  dimnames = list(P = optional.p.state, D = optional.d.state, A = optional.a.state)
)

optional.au.probability <- array(
  c(
    0.98, 0.02,
    0.6, 0.4,
    0.9, 0.1,
    0.2, 0.8,
    0.85, 0.15,
    0.02, 0.98
  ),
  dim = c(2, 2, 3),
  dimnames = list(AU = optional.au.state, H = optional.h.state, A = optional.a.state)
)

optional.g.probability <- array(
  c(
    0.98, 0.02,
    0.3, 0.7,
    0.95, 0.05,
    0.2, 0.8
  ),
  dim = c(2, 2, 2),
  dimnames = list(G = optional.g.state, AU = optional.au.state, P = optional.p.state)
)

optional.i.probability <- array(
  c(
    0.9, 0.1,
    0.2, 0.8
  ),
```

```
  dim = c(2, 2),
  dimnames = list(I = optional.i.state, G = optional.g.state)
)

cpt3 <- list(
    D=optional.d.probability,
    A=optional.a.probability,
    HG=optional.hg.probability,
    H=optional.h.probability,
    P=optional.p.probability,
    AU=optional.au.probability,
    G=optional.g.probability,
    I=optional.i.probability
)

bn3 <- custom.fit(dag3, cpt3)
junction3 <- compile(as.grain(bn3))
bn3
```

```
##
##   Bayesian network parameters
##
##   Parameters of node D (multinomial distribution)
##
## Conditional probability table:
##  D
## buena  mala
##   0.8   0.2
##
##   Parameters of node A (multinomial distribution)
##
## Conditional probability table:
##  A
## mucho  poco  nada
##   0.2   0.6   0.2
##
##   Parameters of node HG (multinomial distribution)
##
## Conditional probability table:
##  HG
##  si  no
## 0.2 0.8
##
##   Parameters of node H (multinomial distribution)
##
## Conditional probability table:
##
##     HG
## H      si   no
##   si 0.95 0.10
##   no 0.05 0.90
##
##   Parameters of node P (multinomial distribution)
##
```

```
## Conditional probability table:
##
## , , A = mucho
##
##            D
## P           buena mala
##   sobrepeso  0.30 0.95
##   saludable  0.70 0.05
##
## , , A = poco
##
##            D
## P           buena mala
##   sobrepeso  0.10 0.70
##   saludable  0.90 0.30
##
## , , A = nada
##
##            D
## P           buena mala
##   sobrepeso  0.05 0.60
##   saludable  0.95 0.40
##
##
##   Parameters of node AU (multinomial distribution)
##
## Conditional probability table:
##
## , , H = si
##
##        A
## AU      mucho poco nada
##   mucho  0.98 0.90 0.85
##   poco   0.02 0.10 0.15
##
## , , H = no
##
##        A
## AU      mucho poco nada
##   mucho  0.60 0.20 0.02
##   poco   0.40 0.80 0.98
##
##
##   Parameters of node G (multinomial distribution)
##
## Conditional probability table:
##
## , , AU = mucho
##
##     P
## G    sobrepeso saludable
##   si      0.98      0.95
##   no      0.02      0.05
##
```

```
## , , AU = poco
##
##     P
## G    sobrepeso saludable
##   si      0.30      0.20
##   no      0.70      0.80
##
##
##   Parameters of node I (multinomial distribution)
##
## Conditional probability table:
##
##     G
## I     si  no
##   si 0.9 0.2
##   no 0.1 0.8
```

- **Hacer al menos tres consultas de la probabilidad a posteriori dada cierta evidencia, de alguna variable, usando el método exacto y los dos métodos aproximados.**

```
optional.evidence <- setEvidence(
  junction3,
  nodes = c("A", "P"),
  states = c("mucho", "sobrepeso")
)

optional.exact_method <- querygrain(
  optional.evidence,
  nodes = c("AU", "G", "I"),
  type = "marginal"
)

optional.au.approximated.ls <- cpquery(
  bn3,
  event = (AU == "mucho"),
  evidence = (A == "mucho") & (P == "sobrepeso"),
  method = "ls"
)
optional.g.approximated.ls <- cpquery(
  bn3,
  event = (G == "si"),
  evidence = (A == "mucho") & (P == "sobrepeso"),
  method = "ls"
)
optional.i.approximated.ls <- cpquery(
  bn3,
  event = (I == "si"),
  evidence = (A == "mucho") & (P == "sobrepeso"),
  method = "ls"
)

optional.approximated_method.ls <- list(
  AU = array(
    c(optional.au.approximated.ls, abs(optional.au.approximated.ls - 1)),
    dim = 2,
```

```
    dimnames = list(AU = optional.au.state)
  ),
  G = array(
    c(optional.g.approximated.ls, abs(optional.g.approximated.ls - 1)),
    dim = 2,
    dimnames = list(G = optional.g.state)
  ),
  I = array(
    c(optional.i.approximated.ls, abs(optional.i.approximated.ls - 1)),
    dim = 2,
    dimnames = list(I = optional.i.state)
  )
)

optional.au.approximated.lw <- cpquery(
  bn3,
  event = (AU == "mucho"),
  evidence = list(A = "mucho", P = "sobrepeso"),
  method = "lw"
)
optional.g.approximated.lw <- cpquery(
  bn3,
  event = (G == "si"),
  evidence = list(A = "mucho", P = "sobrepeso"),
  method = "lw"
)
optional.i.approximated.lw <- cpquery(
  bn3,
  event = (I == "si"),
  evidence = list(A = "mucho", P = "sobrepeso"),
  method = "lw"
)

optional.approximated_method.lw <- list(
  AU = array(
    c(optional.au.approximated.lw, abs(optional.au.approximated.lw - 1)),
    dim = 2,
    dimnames = list(AU = optional.au.state)
  ),
  G = array(
    c(optional.g.approximated.lw, abs(optional.g.approximated.lw - 1)),
    dim = 2,
    dimnames = list(G = optional.g.state)
  ),
  I = array(
    c(optional.i.approximated.lw, abs(optional.i.approximated.lw - 1)),
    dim = 2,
    dimnames = list(I = optional.i.state)
  )
)

optional.exact_method
```

```
## $AU
```

```
## AU
##  mucho   poco
## 0.7026 0.2974
##
## $G
## G
##       si       no
## 0.777768 0.222232
##
## $I
## I
##        si        no
## 0.7444376 0.2555624
```

```
optional.approximated_method.ls
```

```
## $AU
## AU
##     mucho      poco
## 0.7096774 0.2903226
##
## $G
## G
##        si        no
## 0.7511111 0.2488889
##
## $I
## I
##        si        no
## 0.7556561 0.2443439
```

```
optional.approximated_method.lw
```

```
## $AU
## AU
##     mucho      poco
## 0.7049535 0.2950465
##
## $G
## G
##        si        no
## 0.7739455 0.2260545
##
## $I
## I
##      si      no
## 0.73101 0.26899
```

- **Comprobar la d-separación en unos tres casos**

```
dsep(bn3, x = "A", y = "HG")
```

```
## [1] TRUE
```

```
dsep(bn3, x = "A", y = "HG", z = "AU")
```

```
## [1] FALSE
```

```r
dsep(bn3, x = "I", y = "A", z="G")
```

```
## [1] TRUE
```