From bnlearn to rjags

This is a simulated network from *Bayesian Networks with Examples in R* by Marco Scutari and Jean-Baptiste Denis. The example is about crop loss caused by pests. For modeling crop loss, the network structure is constructed using expert knowledge.

```r
library(bnlearn)
```

```
##
## Attaching package: 'bnlearn'

## The following object is masked from 'package:stats':
##
##     sigma
```

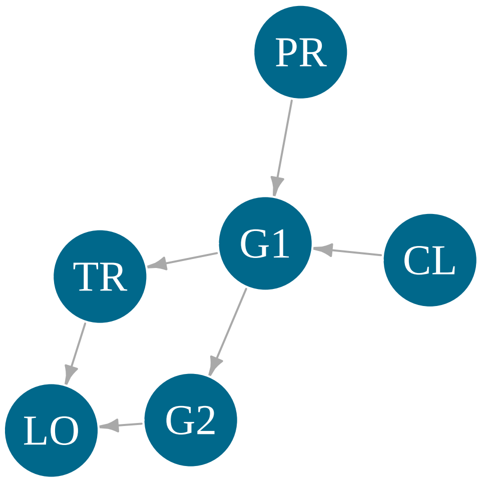
```r
library(rjags)
```

```
## Loading required package: coda

## Linked to JAGS 4.1.0

## Loaded modules: basemod,bugs
```

```r
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:bnlearn':
##
##     compare, degree, path, subgraph

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```r
# Create network structure through conditional probability
# character string.
pest.dag <- model2network("[PR][CL][G1|PR:CL][G2|G1][TR|G1][LO|G2:TR]")

# Plot the pest network.
# Create graph from network arcs.
pest.g <- graph_from_data_frame(pest.dag$arcs)

# Plot graph.
plot(pest.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```



In this example, pests damage the crop in two ways: when the first generation (G1) awakes at the end of winter, and when the second generation (G2) is born at a later time. G2 is influenced by G1. G1 is influenced by the preceding crop in the plot (PR) and the climate conditions (CL) during winter. Crop loss (LO) is primarily influenced by G2, which is larger than G1 and tends to eat the reproductive organs of the plants. Whether or not the farmer treats the crop for pests (TR) also influences LO.

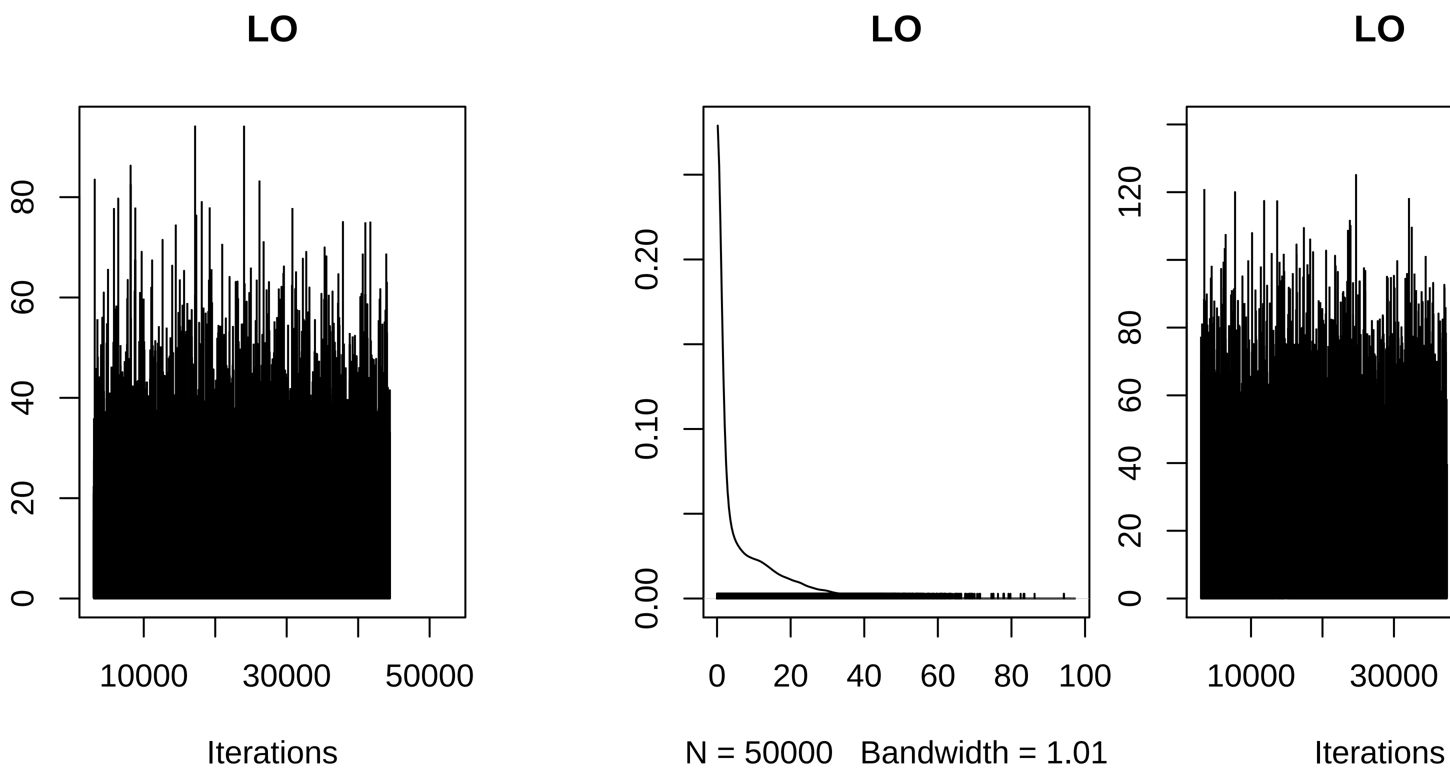
All variables have their own distribution:

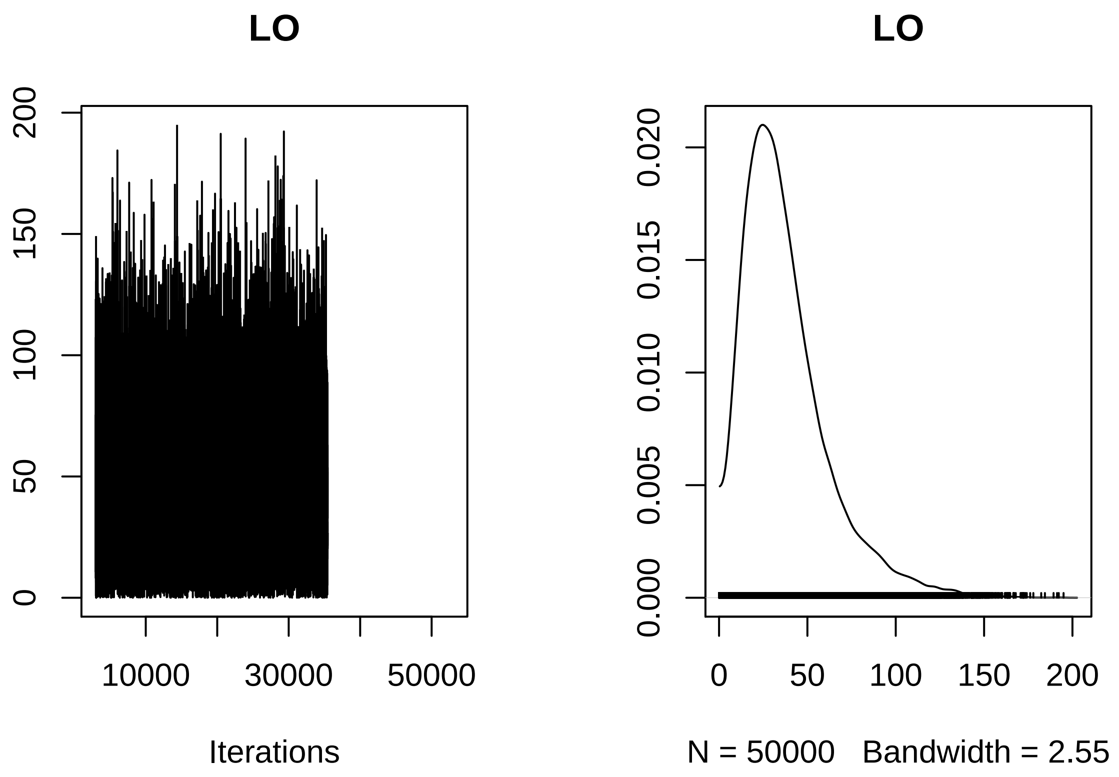
PR ~ $Mu(1, p = (0.7, 0.2, 0.1))$
CL ~ $Beta(3, 1)$
G1 | PR = p, CL = c ~ $Pois(c \times g(p))$
G2 | G1 = g_1 ~ $Pois(10 \times G1)$
TR | G1 = g_1 ~ $Ber(logit^{-1}\left[\frac{g_1-5}{2.5}\right])$
LO | G2 = g_2, TR = t ~ $nc\chi^2\left(1, \left[g_2 \times \left\{1 - \frac{2t}{3}\right\}\right]\right)$

The following code shows parameter learning using rjags and the graph of $P(LO|G2, TR)$ with PR = 1, 2, and 3:

```
library(rjags)
###################################################
### code chunk number 12: chapter3.rnw:416-422
###################################################
dat0 <- list(p.PR = c(0.7, 0.2, 0.1),
             a.CL = 3, b.CL = 1,
             g.G1 = c(1, 3, 10),
             k.G2 = 10,
             m.TR = 5, s.TR = 2.5,
             r.LO = 1/3, d.LO = 1)


###################################################
### code chunk number 13: chapter3.rnw:474-475
###################################################
set.seed(123)

# SET WORKING DIRECTORY CONTAINING rjags FILE HERE:
setwd("~/Documents/GitHub/brassicaDroughtBN/jags")

###################################################
### code chunk number 14: pest.PR123
```

```
####################################################
exp.loss  <- rep(NA, 3)
names(exp.loss) <- paste("PR=", 1:3, sep = "")
qua.loss <- exp.loss
for (PR in 1:3) {
  dat1 <- dat0
  dat1$PR <- PR
  mopest <- jags.model(file = "inclu.pest.jam", data = dat1,
                       quiet = TRUE)
  update(mopest, 3000)
  sipest <-
    coda.samples(model = mopest, variable.names = "LO",
                 n.iter  =  50000)
  summa <- summary(sipest)
  exp.loss[PR] <- summa$statistics["Mean"]
  qua.loss[PR] <- summa$quantiles["75%"]
  plot(sipest[[1]][, "LO"], main = "LO")
}#FOR
```





```
#plot(sipest[[1]][, "LO"], main = "LO")
round(c(exp.loss, MEAN = mean(exp.loss)), 1)
```

```
## PR=1 PR=2 PR=3 MEAN
##  7.5 18.2 37.5 21.1
```

We can see that LO changes given different values of PR.

Structure learning

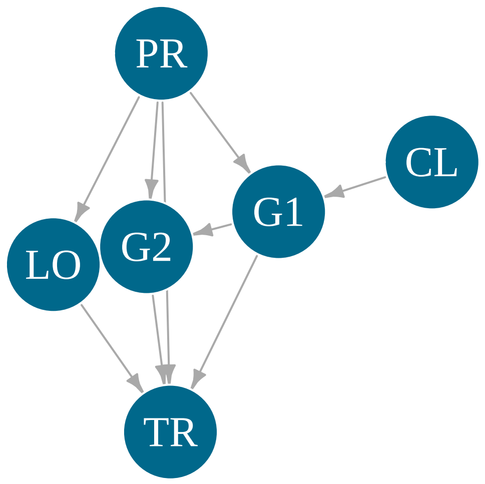
For my network, I don't have a structure and I don't have distributions for my nodes. But I can simulate data and learn the structure as I will for Lina's droughted data.

```
#### Simulate data from p.76
rm(list=ls())
set.seed(567)

# Simulate 50,000 samples from each node's distribution.
nbs <- 50000
PR <- sample(1:3, nbs, prob = c(.7, .2, .1), replace = T)
g <- function(pr)c(1, 3, 10)[pr]
CL <- rbeta(nbs, 3, 1)
G1 <- rpois(nbs, CL * g(PR))
G2 <- rpois(nbs, G1 * 10)
il <- function(x){
  exp((x - 5) / 2.5)/(1 + exp((x-5) / 2.5))
}
TR <- rbinom(nbs, 1, il(G1))
x.lo <- G2 * (1 - (1 - 1/3) * TR)
```

```r
LO <- rchisq(nbs, 1, ncp = x.lo)

# Combine data into crop dataframe.
crop <- data.frame(CL = CL, G1 = G1, G2 = G2, TR = TR, LO = LO, PR = PR)
```

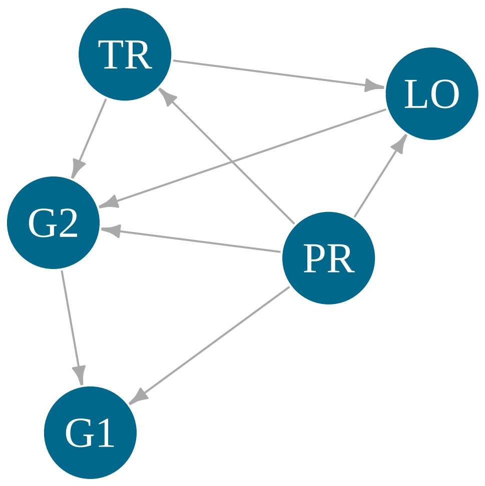
For structure learning, I'll show a couple learning algorithms to show you how they compare.

```r
# Convert all variables to numeric.
crop <- as.data.frame(sapply(crop, as.numeric))

# Discretize the data appropriately.
cropDisc <- discretize(crop, method = "interval",
                       breaks = c(5, 5, 5, 2, 5, 3))

# Use tabu algorithm to find the highest network score.
cropBN <- tabu(cropDisc, score = "bde", iss = 5, tabu = 50)

# Plot network with highest network score.
# Create graph from network arcs.
tabu.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabu.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```

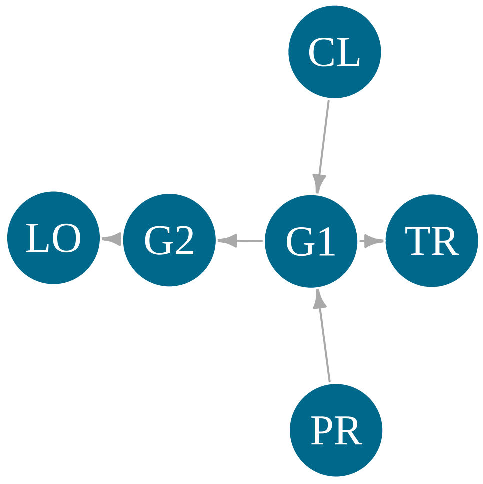


The tabu algorithm has many redundant relationships and doesn't find the true network. Hybrid learning algorithms can help off-set some of the weaknesses of score or structure based algorithms used on their own. Here, I use the tabu/si.hiton.pc hybrid algorithm to find the network structure.

```r
# Use hybrid tabu/si.hiton.pc algorithm to find the highest network score.
cropBN <- rsmax2(cropDisc, restrict = "si.hiton.pc", test = "x2",
                 maximize = "tabu", score = "bde",
                 maximize.args = list(iss = 5))

# Plot network with highest network score.
# Create graph from network arcs.
tabuHiton.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabuHiton.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```



```r
# Use hybrid tabu/aracne algorithm to find the highest network score.
cropBN <-   rsmax2(cropDisc, restrict = "aracne", maximize = "tabu",
        score = "bde", maximize.args = list(iss = 5))

# Plot network with highest network score.
# Create graph from network arcs.
tabuAracne.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabuAracne.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```



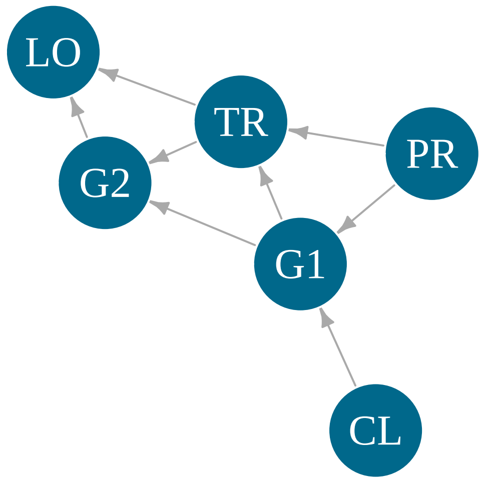
The tabu-aracne hybrid algorithm comes the closest to learning the correct network structure. As you can see, it's still missing one relationship (TR -> LO).

Now we'll look at what happens when you separate the crop dataframe to create two networks: **Treatment** and **No Treatment**. We'll start by looking at **Treatment** values only.

```r
# Subset treatment only.
cropTR1 <- cropDisc[crop$TR == 1, ]

# Use hybrid tabu/aracne algorithm to find the highest network score.
cropBN <-   rsmax2(cropTR1, restrict = "aracne", maximize = "tabu",
         score = "bde", maximize.args = list(iss = 5))
```

```
## Warning in check.data(x, allowed.types = c(discrete.data.types,
## continuous.data.types)): variable TR has levels that are not observed in
## the data.

## Warning in check.data(x, allowed.types = c(discrete.data.types,
## continuous.data.types)): variable LO has levels that are not observed in
## the data.

## Warning in check.data(x): variable TR has levels that are not observed in
## the data.

## Warning in check.data(x): variable LO has levels that are not observed in
## the data.
```

```r
# Plot network with highest network score.
# Create graph from network arcs.
tabuAracne.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabuAracne.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```

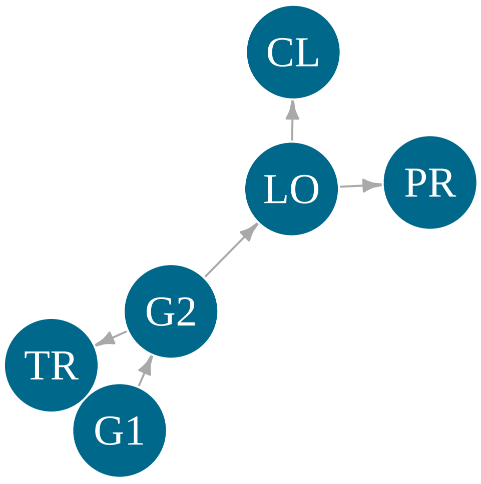


Notice that subsetting **Treatment** only takes out levels of LO. Although this structure is close to the correct structure (in fact, it does have TR -> LO), it also has redundant relationships. Let's look at **No Treatment** now.

```
# Subset no treatment only.
cropTR0 <- cropDisc[crop$TR == 0, ]

# Use hybrid tabu/aracne algorithm to find the highest network score.
cropBN <-   rsmax2(cropTR0, restrict = "aracne", maximize = "tabu",
         score = "bde", maximize.args = list(iss = 5))
```

```
## Warning in check.data(x, allowed.types = c(discrete.data.types,
## continuous.data.types)): variable G1 has levels that are not observed in
## the data.

## Warning in check.data(x, allowed.types = c(discrete.data.types,
## continuous.data.types)): variable TR has levels that are not observed in
## the data.

## Warning in check.data(x): variable G1 has levels that are not observed in
## the data.

## Warning in check.data(x): variable TR has levels that are not observed in
## the data.
```

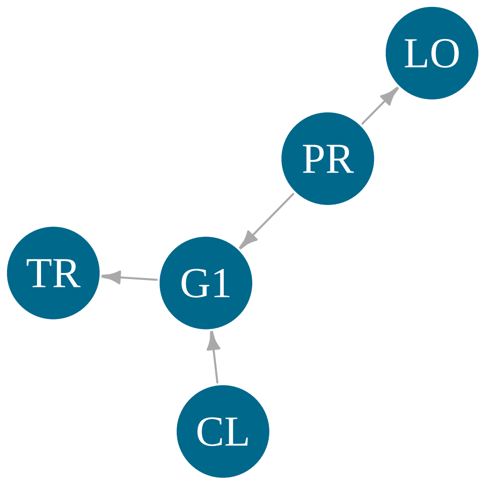
```
# Plot network with highest network score.
# Create graph from network arcs.
tabuAracne.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabuAracne.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```



Notice that subsetting **No Treatment** takes out levels of G1 necessary to get close to the correct structure. Some of the relationships (like G1 -> G2 -> LO) are correct, but others are not. When subsetting based on treatment, we were able to see TR -> LO with the **Treatment** group, but we saw many incorrect relationships with the **No Treatment** group. It is apparent that the levels of G1 removed from the data when subsetting the **No Treatment** group was more important to learning the correct structure than the levels of LO lost in the **Treatment** group.

Without knowing the correct structure beforehand, reconciling the differences between these two networks is not straightforward. Additionally, the relationships between **Treatment** and **No Treatment** *shouldn't really* change, only their probability distributions should. Let's look at how the probability distributions change with TR = 0 and TR = 1.

```r
# Subset no treatment only.
cropG2gone <- cropDisc[, -3]

# Use hybrid tabu/aracne algorithm to find the highest network score.
cropBN <-   rsmax2(cropG2gone, restrict = "aracne", maximize = "tabu",
         score = "bde", maximize.args = list(iss = 5))

# Plot network with highest network score.
# Create graph from network arcs.
tabuAracne.g <- graph_from_data_frame(cropBN$arcs)

# Plot graph.
plot(tabuAracne.g, vertex.size = 50,
     vertex.label.cex = 1.3,
     vertex.color = "deepskyblue4",
     edge.arrow.size = 0.5,
     vertex.frame.color = "white",
     vertex.label.color = "white")
```



Conditional Probability Distributions.