Running a simple MCMC example

Jack Kuipers, March 6, 2015

For this example we will assume that the score of any DAG is simply proportional to its number of edges. The size of the DAGs is also restricted to $n = 3$

```
n <- 3
```

First we load the `R` files needed to run the MCMC codes

```
source("./edgerevandstructure/structurefns.R")
source("./edgerevandstructure/structureMCMC.R")
source("./edgerevandstructure/newedgerevfns.R")
source("./edgerevandstructure/newedgerevmove.R")
source("./orderandpartition/orderMCMC.R")
source("./orderandpartition/orderfns.R")
source("./orderandpartition/partitionMCMC.R")
source("./orderandpartition/partitionmoves.R")
source("./orderandpartition/partitionfns.R")
source("./orderandpartition/samplefns.R")
source("./scoring/combinations.R")
source("./scoring/scorefns.R")
source("./scoring/scoretables.R")
# load a simple score proportional to the number of edges in
# the DAG
source("./scoring/numedgescore.R")
```

Then we build the score table of all parent sets

```
maxparents <- 2  # Maximum number of parents to allow
# Fill up a matrix with possible parents
parenttable <- listpossibleparents(maxparents, c(1:n))
tablelength <- nrow(parenttable[[1]])  # size of the table
# Now need to score them!
scoretable <- scorepossibleparents(parenttable, n)
```

and fix the number of iterations for the various MCMC schemes, recording the outcome at each step

```
iterations <- 100  # number of iterations in the chain
stepsave <- 1  # how often to record the DAG
```

We will also perform all the runs with the same seed

```
seednumber <- 101  # seed number
```

1 Structure MCMC

For structure MCMC we include standard edge reversal by setting

```
revallowed <- 1  # allow standard edge reversals
```

and exclude the new edge reversal move by giving the `moveprobs` vector a single element

```
moveprobs <- c(1)  # having length 1 disallows the new edge reversal move
```

Starting with a DAG with no edges

```
startDAG <- matrix(numeric(n * n), nrow = n)  # starting DAG is empty say
```

we can then run structure MCMC as follows

```
set.seed(seednumber)  # set the seed
example <- structureMCMC(n, startDAG, iterations, stepsave, maxparents,
    parenttable, scoretable, revallowed, moveprobs)  # run the MCMC code
```

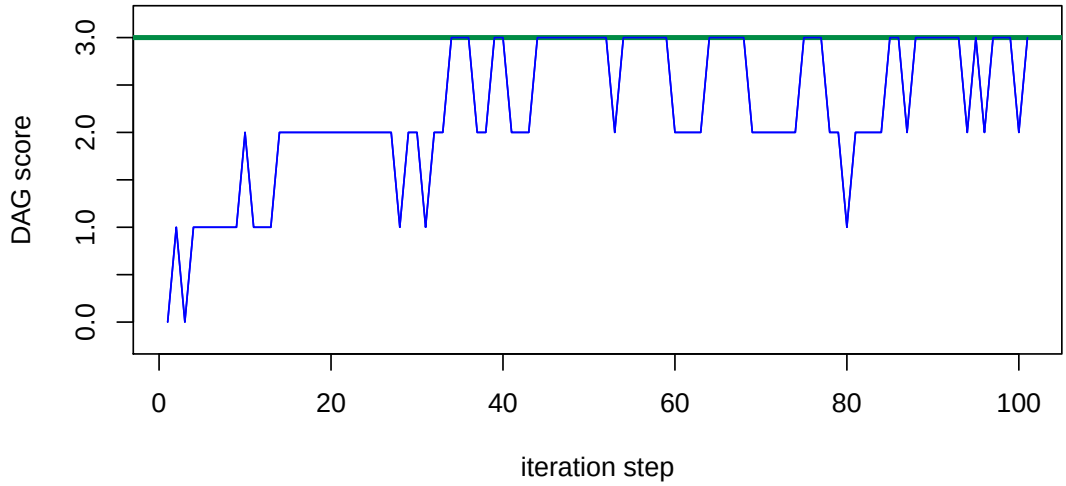
The result of the function is a list containing the sampled DAGs (stored as incidence matrices) as the first element and their scores as the second. We can extract the scores and an example of the DAG with the highest score, represened as its incidence matrix

```
DAGscores <- unlist(example[[2]])
maxscore <- max(DAGscores)
maxDAG <- example[[1]][which(DAGscores == maxscore)][[1]]
maxDAG

     [,1] [,2] [,3]
[1,]    0    1    0
[2,]    0    0    0
[3,]    1    1    0
```

The trace plot of the run instead is

```
nparts <- length(example[[2]])
plot(1:nparts, DAGscores, type = "l", ylab = "DAG score", xlab = "iteration step",
    main = "", col = "blue", ylim = c(maxscore - 3.2, maxscore +
        0.2))
abline(h = maxscore, col = "springgreen4", lwd = 3)  # the maximal score line
lines(1:nparts, DAGscores, type = "l", col = "blue")
```



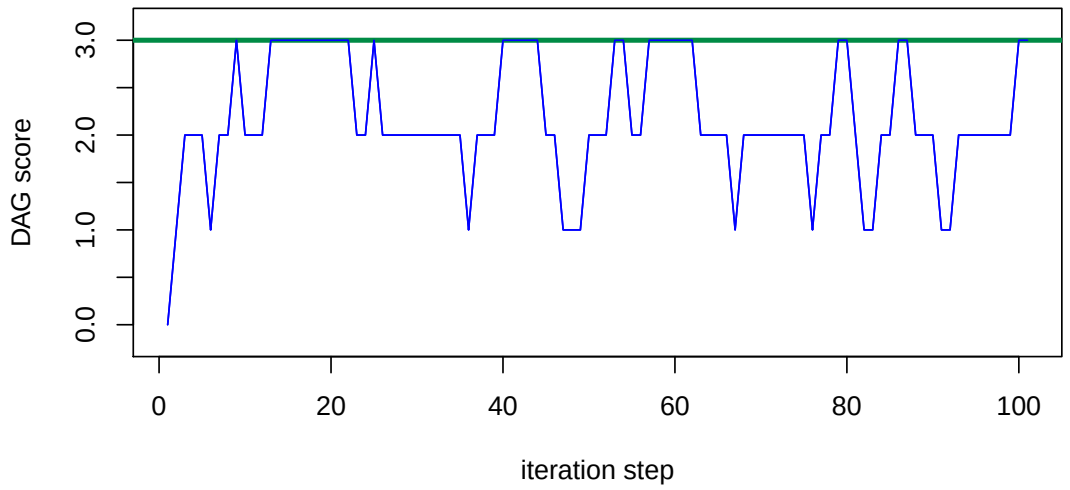
2 New edge reversal move

We can include the new edge reversal move, by simply making the vector `moveprobs` have two elements, with the second corresponding to the probability of picking the new edge reversal move.

```
moveprobs <- c(0.93, 0.07)  # having length 1 disallows the new edge reversal move
set.seed(seednumber)  # set the seed
example <- structureMCMC(n, startDAG, iterations, stepsave, maxparents,
    parenttable, scoretable, revallowed, moveprobs)  # run the MCMC code
```

The trace plot is now

```
DAGscores <- unlist(example[[2]])
maxscore <- max(DAGscores)
nparts <- length(example[[2]])
plot(1:nparts, DAGscores, type = "l", ylab = "DAG score", xlab = "iteration step",
    main = "", col = "blue", ylim = c(maxscore - 3.2, maxscore +
        0.2))
abline(h = maxscore, col = "springgreen4", lwd = 3)  # the maximal score line
lines(1:nparts, DAGscores, type = "l", col = "blue")
```



3 Order MCMC

For Order MCMC we now have three elements in the `moveprobs` vector. The first is the probability of swapping any two elements in the order, the second the probability of only swapping adjacent nodes and the last the probability to stay still (to enforce aperiodicity of the chain)

```
moveprobs <- c(0.99, 0, 0.01)
```

We also need to choose a starting order, for example the identity permutation

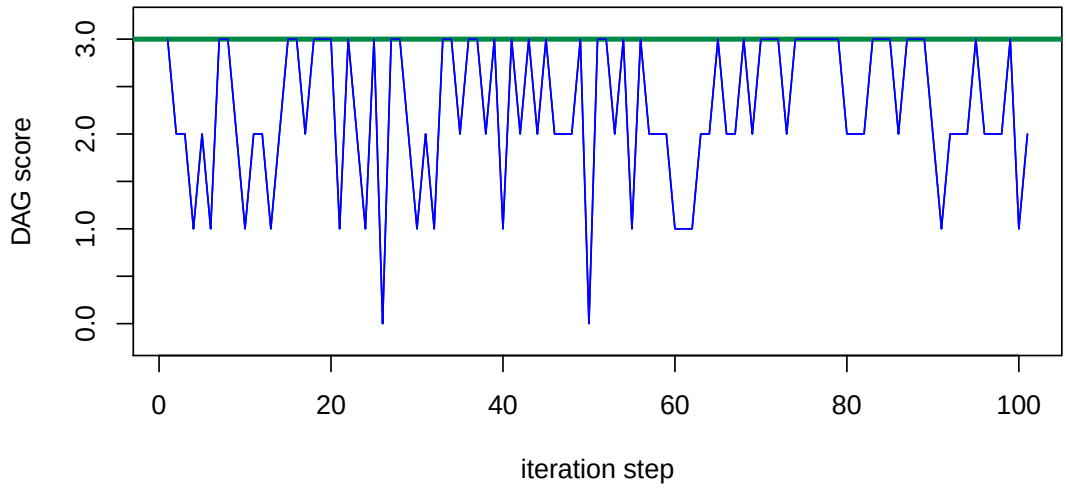
```
startorder <- c(1:n)  # starting order
```

and then we can run the MCMC code

```
set.seed(seednumber)  # set the seed
example <- orderMCMC(n, startorder, iterations, stepsave, parenttable,
    scoretable, moveprobs)
```

The output now contains four components, a sampled DAG and its score along with the score of the entire order and the sampled order itself (stored as a permutation vector). We can again plot the outcome

```
DAGscores <- unlist(example[[2]])
maxscore <- max(DAGscores)
nparts <- length(example[[2]])
plot(1:nparts, DAGscores, type = "l", ylab = "DAG score", xlab = "iteration step",
    main = "", col = "blue", ylim = c(maxscore - 3.2, maxscore +
        0.2))
abline(h = maxscore, col = "springgreen4", lwd = 3)  # the maximal score line
lines(1:nparts, DAGscores, type = "l", col = "blue")
```



4 Partition MCMC

For Partiton MCMC we have five elements in the `moveprobs` vector. The first is the probability of swapping two nodes from different partititon elements, while the second for swapping nodes from adjacent partition elements. The third corresponds to joining and splitting partition elements with the fourth being the probability of moving a node from one element to another. The last is again the probability of not moving

```
moveprobs <- c(0.4, 0, 0, 0.59, 0.01)
```

For the starting partition we can choose the DAG with no edges

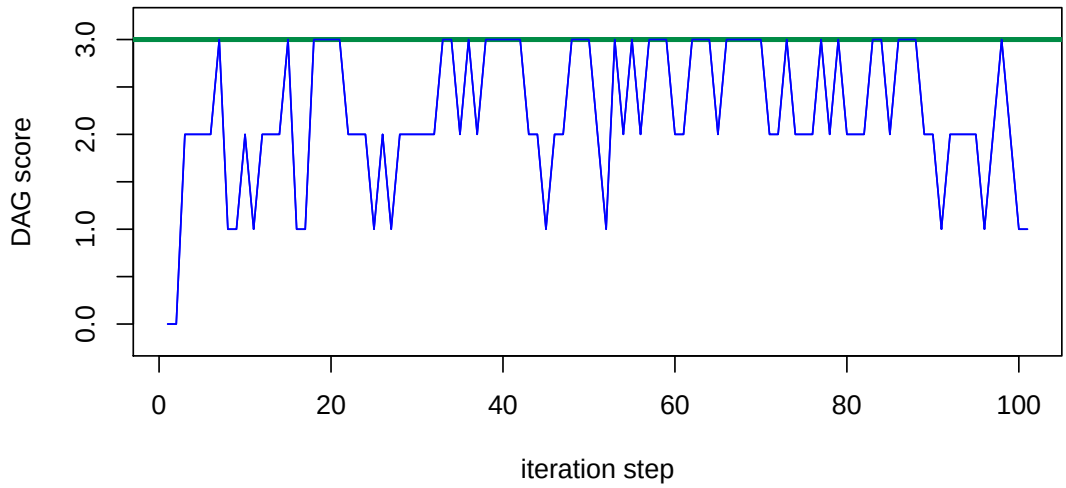
```
startpermutation <- c(1:n)  # pick a starting permutation
startpartition <- c(n)  # and a starting partition - c(n) gives the empty DAG
```

and then run the MCMC code

```
set.seed(seednumber)  # set the seed
example <- partitionMCMC(n, startpermutation, startpartition, iterations,
    stepsave, parenttable, scoretable, moveprobs)
```

The output now contains a five components: a sampled DAG and its score, the score of the entire partition and the sampled permutation and partition. The trace plot is

```
DAGscores <- unlist(example[[2]])
maxscore <- max(DAGscores)
nparts <- length(example[[2]])
plot(1:nparts, DAGscores, type = "l", ylab = "DAG score", xlab = "iteration step",
    main = "", col = "blue", ylim = c(maxscore - 3.2, maxscore +
        0.2))
abline(h = maxscore, col = "springgreen4", lwd = 3)  # the maximal score line
lines(1:nparts, DAGscores, type = "l", col = "blue")
```



5 Partition MCMC with edge reversal

Finally we can include the new edge reversal move on an underlying Partition MCMC chain by including a sixth element in the `moveprobs` vector, corresponding to the probability of picking the edge reversal move

```
moveprobs <- c(0.37, 0, 0, 0.55, 0.01, 0.07)
set.seed(seednumber)  # set the seed
example <- partitionMCMC(n, startpermutation, startpartition, iterations,
    stepsave, parenttable, scoretable, moveprobs)
```

with trace plot

```
DAGscores <- unlist(example[[2]])
maxscore <- max(DAGscores)
nparts <- length(example[[2]])
plot(1:nparts, DAGscores, type = "l", ylab = "DAG score", xlab = "iteration step",
    main = "", col = "blue", ylim = c(maxscore - 3.2, maxscore +
        0.2))
abline(h = maxscore, col = "springgreen4", lwd = 3)  # the maximal score line
lines(1:nparts, DAGscores, type = "l", col = "blue")
```