NMA package practical

Introduction

In this practical we repeat the analysis from the previous practical but using the `NMA` package.

Install `NMA` package

Obtain `NMA` from GitHub using the following.

```
if (!require(NMA)) remotes::install_github("ICON-in-R/NMA")
```

Data preparation

Firstly, load the study data in to the workspace from the `NMA-tutorial` project.

```
load(here::here("practicals","BUGS","smoke.Rdata"))
```

The original data are in an R `list` format and look like this

```
smoke.list
```

```
$r
     [,1] [,2] [,3] [,4]
 [1,]   79   77   NA   NA
 [2,]   18   21   NA   NA
 [3,]    8   19   NA   NA
 [4,]   75   NA  363   NA
 [5,]    2   NA    9   NA
 [6,]   58   NA  237   NA
 [7,]    0   NA    9   NA
 [8,]    3   NA   31   NA
 [9,]    1   NA   26   NA
[10,]    6   NA   17   NA
[11,]   64   NA  107   NA
[12,]    5   NA    8   NA
[13,]   20   NA   34   NA
[14,]   95   NA  143   NA
[15,]   15   NA   36   NA
[16,]   78   NA   73   NA
[17,]   69   NA   54   NA
[18,]    9   NA   23   10
[19,]    0   NA   NA    9
[20,]   NA   20   16   NA
[21,]   NA   11   12   29
[22,]   NA    7   NA   32
[23,]   NA   NA   12   20
[24,]   NA   NA    9    3

$n
      [,1] [,2] [,3] [,4]
 [1,]  702  694   NA   NA
 [2,]  671  535   NA   NA
 [3,]  116  149   NA   NA
 [4,]  731   NA  714   NA
 [5,]  106   NA  205   NA
 [6,]  549   NA 1561   NA
 [7,]   33   NA   48   NA
 [8,]  100   NA   98   NA
 [9,]   31   NA   95   NA
[10,]   39   NA   77   NA
[11,]  642   NA  761   NA
[12,]   62   NA   90   NA
[13,]  234   NA  237   NA
[14,] 1107   NA 1031   NA
[15,]  187   NA  504   NA
[16,]  584   NA  675   NA
[17,] 1177   NA  888   NA
[18,]  140   NA  140  138
[19,]   20   NA   NA   20
[20,]   NA   49   43   NA
[21,]   NA   78   85  170
[22,]   NA   66   NA  127
[23,]   NA   NA   76   74
```

```
[24,]   NA   NA   55   26

$t
      t1 t2 t3
 [1,]  1  2 NA
 [2,]  1  2 NA
 [3,]  1  2 NA
 [4,]  1  3 NA
 [5,]  1  3 NA
 [6,]  1  3 NA
 [7,]  1  3 NA
 [8,]  1  3 NA
 [9,]  1  3 NA
[10,]  1  3 NA
[11,]  1  3 NA
[12,]  1  3 NA
[13,]  1  3 NA
[14,]  1  3 NA
[15,]  1  3 NA
[16,]  1  3 NA
[17,]  1  3 NA
[18,]  1  3  4
[19,]  1  4 NA
[20,]  2  3 NA
[21,]  2  3  4
[22,]  2  4 NA
[23,]  3  4 NA
[24,]  3  4 NA

$na
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 3 2 2 2

$NS
[1] 24

$NT
[1] 4
```

In many cases, the data you are provided with is not in the correct format to plug into the code for the NMA, and this is the case here.

We need to rearrange this data in to the format that the `NMA` function requires before we can do the analysis. Generally, learning how to *munge* or *wrangle* is worth the time and effort.

We want a single array with a column for “study”, “treatment”, “n” and “r”. There are several ways to achieve this but we will make use of the `reshape2` package to *melt* the data in to the shape we want. That is, we change the shape of the data from a wide array to a long array so that we will only have one column with values in.

Specifically, the following code converts the original data to a long format, renames the columns and then removed rows which don’t have any values in.

We do this for both number of counts (r) and sample size (n).

```r
library(dplyr)
library(reshape2)

r_data <- melt(smoke.list$r) |>                # rearrange to 'long' format
  `names<-`(c("study", "treatment", "r")) |>  # rename columns
  na.omit()

n_data <- melt(smoke.list$n) |>
  `names<-`(c("study", "treatment", "n")) |>
  na.omit()

head(r_data)
```

```
  study treatment  r
1     1         1 79
2     2         1 18
3     3         1  8
4     4         1 75
5     5         1  2
6     6         1 58
```

The last thing to do is combine these two arrays into a single object. We can simply append one to another or a more elegant and robust way is to *join* them. This will ensure that the correct rows are match up between arrays. The command to do this in base R is `merge()`. We just rearrange the columns in to a nice order at the end.

```r
data <- merge(r_data, n_data,
              by = c("study", "treatment")) |>  # which columns to match
  arrange(study)                                # ascending order

head(data)
```

```
  study treatment  r   n
1     1         1 79 702
2     1         2 77 694
3     2         1 18 671
4     2         2 21 535
5     3         1  8 116
6     3         2 19 149
```

Analysis

The workflow is to first create and NMA object separately to actually doing the fitting. This then means that we can perform modified fits but we don't have to redo any of the preparatory work.

We define the BUGS specific parameter values in the same way as for the first practical and also some extra values to specify whether fixed or random effects model.

```r
library(NMA)

bugs_params <-
  list(
    PROG = "openBugs",  # which version of BUGS to use to run the MCMC
    N.BURNIN = 10,#00,  # number of steps to throw away
    N.SIMS = 150,#0,    # total number of simulations
    N.CHAINS = 2,       # number of chains
    N.THIN = 1,         # thinning rate
    PAUSE = TRUE)

RANDOM <- FALSE               # is this a random effects model?
REFTX <- "X"                  # reference treatment
data_type <- "bin_data"       # which type of data to use
label_name <- "label_name"

nma_model <-
  new_NMA(binData = data,
          bugs_params = bugs_params,
          is_random = RANDOM,
          data_type = data_type,
          refTx = REFTX,
          effectParam = "d",  # which parameters to 'monitor' i.e. record
          label = "",
          endpoint = "")
```

```r
nma_model
```

```
$dat
$dat$inits
function() {
    list(
      beta = c(NA, rnorm(nTx - 1, 0, 2)),
      sd = 0.1,
      alpha = rnorm(nStudies),
      d = c(NA, rnorm(nTx - 1, 0, 2)),  ##TODO: can we remove duplication?
      mu = rnorm(nStudies),
      baseLod = 0) %>%
      .[param_names]  # filter redundant
  }
<bytecode: 0x000002a85c21bbc0>
<environment: 0x000002a85c21d2d8>

$dat$binData
   study treatment   r    n
1      1         1  79  702
2      1         2  77  694
3      2         1  18  671
4      2         2  21  535
5      3         1   8  116
6      3         2  19  149
7      4         1  75  731
8      4         3 363  714
9      5         1   2  106
10     5         3   9  205
11     6         1  58  549
12     6         3 237 1561
13     7         1   0   33
14     7         3   9   48
15     8         1   3  100
16     8         3  31   98
17     9         1   1   31
18     9         3  26   95
19    10         1   6   39
20    10         3  17   77
21    11         1  64  642
22    11         3 107  761
```

```
23    12          1   5   62
24    12          3   8   90
25    13          1  20  234
26    13          3  34  237
27    14          1  95 1107
28    14          3 143 1031
29    15          1  15  187
30    15          3  36  504
31    16          1  78  584
32    16          3  73  675
33    17          1  69 1177
34    17          3  54  888
35    18          1   9  140
36    18          3  23  140
37    18          4  10  138
38    19          1   0   20
39    19          4   9   20
40    20          2  20   49
41    20          3  16   43
42    21          2  11   78
43    21          3  12   85
44    21          4  29  170
45    22          2   7   66
46    22          4  32  127
47    23          3  12   76
48    23          4  20   74
49    24          3   9   55
50    24          4   3   26

$dat$bugsData
$dat$bugsData$mu_beta
[1] 0

$dat$bugsData$prec_beta
[1] 1e-06

$dat$bugsData$mu_alpha
[1] 0

$dat$bugsData$prec_alpha
[1] 1e-06

$dat$bugsData$t
```

```
      [,1] [,2] [,3]
 [1,]    1    2   NA
 [2,]    1    2   NA
 [3,]    1    2   NA
 [4,]    1    3   NA
 [5,]    1    3   NA
 [6,]    1    3   NA
 [7,]    1    3   NA
 [8,]    1    3   NA
 [9,]    1    3   NA
[10,]    1    3   NA
[11,]    1    3   NA
[12,]    1    3   NA
[13,]    1    3   NA
[14,]    1    3   NA
[15,]    1    3   NA
[16,]    1    3   NA
[17,]    1    3   NA
[18,]    1    4   NA
[19,]    2    3   NA
[20,]    2    4   NA
[21,]    3    4   NA
[22,]    3    4   NA
[23,]    1    3    4
[24,]    2    3    4

$dat$bugsData$r
      [,1] [,2] [,3]
 [1,]   79   77   NA
 [2,]   18   21   NA
 [3,]    8   19   NA
 [4,]   75  363   NA
 [5,]    2    9   NA
 [6,]   58  237   NA
 [7,]    0    9   NA
 [8,]    3   31   NA
 [9,]    1   26   NA
[10,]    6   17   NA
[11,]   64  107   NA
[12,]    5    8   NA
[13,]   20   34   NA
[14,]   95  143   NA
[15,]   15   36   NA
```

```
[16,]   78   73   NA
[17,]   69   54   NA
[18,]    0    9   NA
[19,]   20   16   NA
[20,]    7   32   NA
[21,]   12   20   NA
[22,]    9    3   NA
[23,]    9   23   10
[24,]   11   12   29

$dat$bugsData$n
      [,1] [,2] [,3]
 [1,]  702  694   NA
 [2,]  671  535   NA
 [3,]  116  149   NA
 [4,]  731  714   NA
 [5,]  106  205   NA
 [6,]  549 1561   NA
 [7,]   33   48   NA
 [8,]  100   98   NA
 [9,]   31   95   NA
[10,]   39   77   NA
[11,]  642  761   NA
[12,]   62   90   NA
[13,]  234  237   NA
[14,] 1107 1031   NA
[15,]  187  504   NA
[16,]  584  675   NA
[17,] 1177  888   NA
[18,]   20   20   NA
[19,]   49   43   NA
[20,]   66  127   NA
[21,]   76   74   NA
[22,]   55   26   NA
[23,]  140  140  138
[24,]   78   85  170

$dat$bugsData$nt
[1] 4

$dat$bugsData$na
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3
```

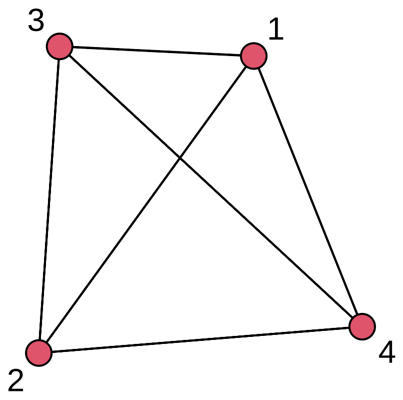
```
$dat$bugsData$ns
[1] 24

$dat$bugsData$baseR
 [1] 79 18  8 75  2 58  0  3  1  6 64  5 20 95 15 78 69  0  9 NA

$dat$bugsData$baseN
 [1]  702  671  116  731  106  549   33  100   31   39  642   62  234 1107  187
[16]  584 1177   20  140    1

$dat$bugsData$nBase
[1] 20

$dat$bugsData$baseTx
[1] 1

$dat$bugsData$refTx
[1] 1


$dat$txList
[1] 1 2 3 4


$data_type
[1] "bin_data"

$bugs_params
$bugs_params$PROG
[1] "openBugs"

$bugs_params$N.BURNIN
[1] 10

$bugs_params$N.SIMS
[1] 150

$bugs_params$N.CHAINS
[1] 2

$bugs_params$N.THIN
[1] 1
```

```
$bugs_params$PAUSE
[1] TRUE

$bugs_params$run_bugs
[1] TRUE


$bugs_fn
function(...)
        R2OpenBUGS::bugs(...)
<bytecode: 0x000002a85c44d480>
<environment: 0x000002a85c45b3d8>

$is_random
[1] FALSE

$refTx
[1] "X"

$effectParam
[1] "d"

$modelParams
   bin_data
"totresdev"

$label
[1] ""

$endpoint
[1] ""

attr(,"class")
[1] "nma"
attr(,"CALL")
attr(,"CALL")$binData
data

attr(,"CALL")$bugs_params
bugs_params

attr(,"CALL")$is_random
RANDOM
```

```
attr(,"CALL")$data_type
data_type

attr(,"CALL")$refTx
REFTX

attr(,"CALL")$effectParam
[1] "d"

attr(,"CALL")$label
[1] ""

attr(,"CALL")$endpoint
[1] ""
```

We can view the network diagram.

```
library(sna)
plotNetwork(nma_model)
```



The `NMA` object can simply be passed to the `NMA_run()` function to do the analysis using

BUGS.

```r
nma_res <- NMA_run(nma_model, save = FALSE)
```

```
====== RUNNING BUGS MODEL
```

```r
nma_res
```

```
Inference for Bugs model at "C:/Users/n8tha/AppData/Local/R/win-library/4.2/NMA/FE_bin.txt",
 2 chains, each with 160 iterations (first 10 discarded)
 n.sims = 300 iterations saved
           mean  sd  2.5%   25%   50%   75% 97.5% Rhat n.eff
d[2]        0.2 0.1   0.0   0.1   0.2   0.3   0.5  1.0   110
d[3]        0.8 0.1   0.6   0.7   0.8   0.8   0.9  1.0   300
d[4]        0.8 0.2   0.4   0.7   0.8   0.9   1.1  1.1    56
totresdev 267.3 7.6 255.7 261.8 266.2 272.4 282.7  1.0   240
deviance  681.4 7.6 669.3 676.1 680.3 687.1 696.5  1.0   300

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = Dbar-Dhat)
pD = 28.0 and DIC = 709.4
DIC is an estimate of expected predictive error (lower deviance is better).
```

- `totresdev` is the total residual deviance