Exercises: Introduction to openVA

Richard Li

Nov 9, 2018

Exercise 1

1. *Using the dataset* va16Data1_hw.csv*, fit InterVA5 and InSilicoVA. Make a log and explain your choice of parameters. If you are not sure about what parameters to choose, you can experiment with a few of them and explain the differences in the fitted model.*

   - We first load the dataset into `R`. Notice you need to set the appropriate working directory or the directory to the data file. Below my codes operate under the folder structure that looks like:
     - exercise_folder
       - codes *(where my working directory is)*
       - data *(where data file is stored)*

     ```
     library(openVA)
     data <- read.csv("../data/va16Data1_hw.csv")
     dim(data)
     ```

     ```
     ## [1] 120 354
     ```

     Note the ".." takes us one level above the working directory and then into the "data" folder.

   - We can do a quick summary check of the data to see everything seem to be in order (results not shown here).

     ```
     summary(data)
     ```

   - In practice, you would most likely need to check the data and choose the parameters based on context. Here we assume we are happy with the data now, and that we know both levels of HIV and Malaria are low in the region. Now we first fit InterVA5 to the data. This step creates a new sub-folder in my `data`. Notice if you want to store the results somewhere else, you need to change the `directory` argument.

     ```
     fit_inter <- codeVA(data = data, data.type = "WHO2016", model = "InterVA", version = "5.0",
         HIV = "l", Malaria = "l", directory = "../data/InterVA_output")
     ```

   - Now, let us try InSilicoVA. There are many options, so it is likely we need to perform the analysis a few times to find the best set of parameters. We will start with the default.

     ```
     fit_ins <- codeVA(data = data, data.type = "WHO2016", model = "InSilicoVA", Nsim = 5000)
     ```

   - Let us first see how it looks. We eventually ran the chain $20,000$ iterations, because the algorithm automatically doubled the lengths twice.

     ```
     summary(fit_ins)
     ```

     ```
     ## InSilicoVA Call:
     ## 100 death processed
     ## 20000 iterations performed, with first 10000 iterations discarded
     ##  1000 iterations saved after thinning
     ## Fitted with re-estimated conditional probability level table
     ## Data consistency check performed as in InterVA4
     ##
     ```

```
## Top 10 CSMFs:
##                                    Mean Std.Error  Lower Median  Upper
## Fresh stillbirth                 0.1277    0.0336 0.0701 0.1248 0.2018
## Respiratory neoplasms            0.0826    0.0288 0.0380 0.0788 0.1480
## Acute resp infect incl pneumonia 0.0787    0.0278 0.0310 0.0767 0.1424
## Other and unspecified neoplasms  0.0728    0.0288 0.0298 0.0678 0.1368
## Prematurity                      0.0723    0.0258 0.0269 0.0703 0.1269
## Diarrhoeal diseases              0.0668    0.0242 0.0258 0.0648 0.1199
## Stroke                           0.0593    0.0257 0.0187 0.0557 0.1175
## Meningitis and encephalitis      0.0425    0.0227 0.0110 0.0377 0.0977
## Road traffic accident            0.0367    0.0000 0.0367 0.0367 0.0367
## Assault                          0.0339    0.0000 0.0339 0.0339 0.0339
```

- Also, at the end of the run, it still reports convergence not achieved for all causes of death with probability greater than 0.02. This warning means there are some values in the CSMF estimates that are not stable enough. However, it does not necessarily mean the results are unreliable, since the very tiny probabilities can and usually do 'bounce' a lot. But in practice, it does not matter much as long as they are close to 0. Notice the default criterion also uses a more strict criterion. In fact, if you check the convergence for causes with probability greater than 0.06, they all passed the test.

```
csmf.diag(fit_ins, conv.csmf = 0.06)
```

```
##
##                                  Stationarity start     p-value
##                                  test         iteration
## Fresh stillbirth                 passed         1       0.8471
## Respiratory neoplasms            passed         1       0.1576
## Acute resp infect incl pneumonia passed         1       0.4433
## Other and unspecified neoplasms  passed       101       0.2199
## Prematurity                      passed         1       0.0935
## Diarrhoeal diseases              passed         1       0.0567
##
##                                  Halfwidth Mean   Halfwidth
##                                  test
## Fresh stillbirth                 passed    0.1277 0.00458
## Respiratory neoplasms            passed    0.0826 0.00519
## Acute resp infect incl pneumonia passed    0.0787 0.00512
## Other and unspecified neoplasms  passed    0.0710 0.00553
## Prematurity                      passed    0.0723 0.00477
## Diarrhoeal diseases              passed    0.0668 0.00477
```

- The acceptance rate if about 0.66. It is not necessarily bad, but since it is not very costly to run this analysis, let us try to make the proposal of new CSMF values 'jump' a little bit more, so that we can explore more space. Also, let us tune down the convergence test by only checking causes with probability greater than 0.5. If you forget what is the default values of some arguments, type `?insilico`.

```
fit_ins <- codeVA(data = data, data.type = "WHO2016", model = "InSilicoVA", Nsim = 5000,
    jump.scale = 0.25, conv.csmf = 0.05, directory = "../data/InSilicoVA_output",
    warning.write = TRUE)
```

- Now this time we have acceptance ratio close to 0.26, and the algorithm automatically stopped after $10,000$ iterations. Let us take another look at the result summaries. Compare with the first try, you can see the results change only slightly.

```
summary(fit_ins)
```

```
## InSilicoVA Call:
## 100 death processed
## 10000 iterations performed, with first 5000 iterations discarded
##  500 iterations saved after thinning
## Fitted with re-estimated conditional probability level table
## Data consistency check performed as in InterVA4
##
## Top 10 CSMFs:
##                                    Mean Std.Error  Lower Median  Upper
## Fresh stillbirth                 0.1230    0.0345 0.0640 0.1204 0.1934
## Respiratory neoplasms            0.0791    0.0268 0.0377 0.0751 0.1430
## Acute resp infect incl pneumonia 0.0756    0.0269 0.0310 0.0736 0.1348
## Other and unspecified neoplasms  0.0706    0.0292 0.0268 0.0663 0.1356
## Diarrhoeal diseases              0.0658    0.0252 0.0274 0.0628 0.1291
## Prematurity                      0.0617    0.0238 0.0233 0.0596 0.1134
## Stroke                           0.0572    0.0252 0.0184 0.0550 0.1152
## Meningitis and encephalitis      0.0383    0.0199 0.0060 0.0354 0.0782
## Road traffic accident            0.0367    0.0000 0.0367 0.0367 0.0367
## Birth asphyxia                   0.0344    0.0191 0.0070 0.0307 0.0820
```

2. Using both InterVA5 and InSilicoVA, obtain the CSMF estimates, individual probabilities of all causes, and most likely causes for each individual.

   - We extract CSMF estimates, individual probabilities and most likely CODs here. You should explore these results by yourselves.

```
csmf_int <- getCSMF(fit_inter)
csmf_ins <- getCSMF(fit_ins)
prob_int <- getIndivProb(fit_inter)
prob_ins <- getIndivProb(fit_ins)
assign_int <- getTopCOD(fit_inter)
assign_ins <- getTopCOD(fit_ins)
```

3. Find out how many records dropped out of the analysis.

   - You can find this in the error files, or you can also see from the extracted output.

```
dim(data)[1]
```

```
## [1] 120
```

```
dim(prob_int)[1]
```

```
## [1] 100
```

```
dim(prob_ins)[1]
```

```
## [1] 100
```

4. Using both algorithms, compare the number of deaths attributed to each cause with the CSMF fraction. Do they differ proportionally? Can you explain why they are not the same.

   - First let us find the number of assigned causes under each algorithm

```
tab_int <- table(assign_int[, 2])
tab_ins <- table(assign_ins[, 2])
```

- To avoid misalign numbers for different causes. Let us also define an order for the causes, so that we do not confuse ourselves later, and redo the table command. Only this time, with a little trick that gives you the results in order and with 0 counts. Also the reason I use the InterVA CSMF to extract causes instead of InSilicoVA, is that there is an additional cause 'undetermined' from InterVA, i.e., there are 62 causes rather than 61.

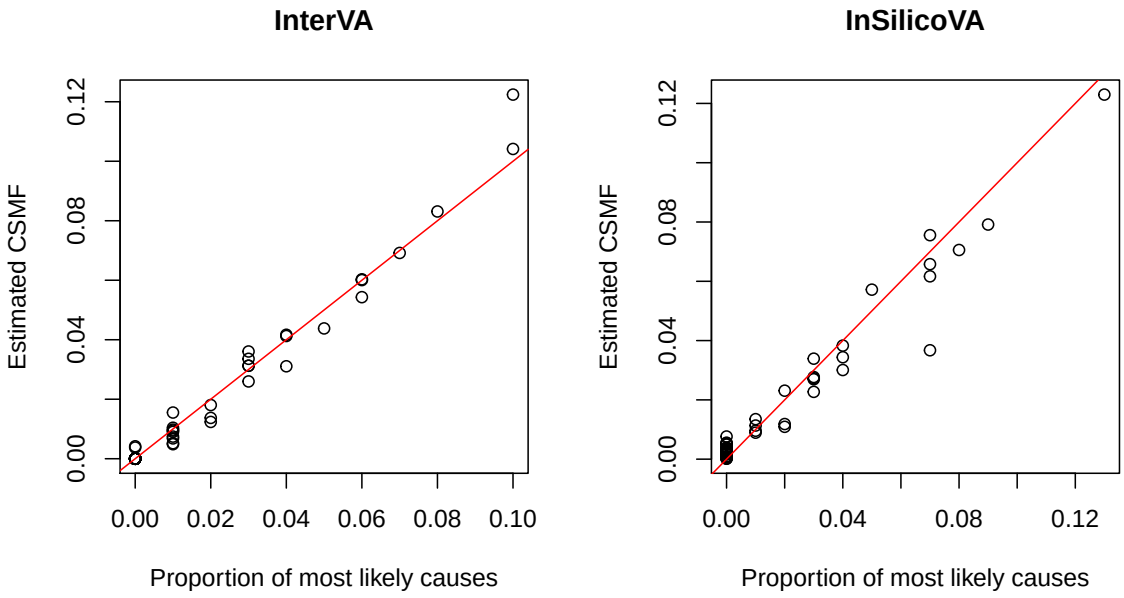
```
causes <- names(csmf_int)
tab_int <- table(c(causes, as.character(assign_int[, 2]))) - 1
tab_int <- tab_int[causes]
tab_ins <- table(c(causes, as.character(assign_ins[, 2]))) - 1
tab_ins <- tab_ins[causes]
```

- Putting them side by side, you can compare the counts

```
tabs <- cbind(tab_int, tab_ins)
```

- Back to the question, we first normalize the counts into fractions, and compare with the CSMF estimates. The visual comparison is as below. Since InSilicoVA does not assign 'undetermined' causes, we remove the last number from \texttt{tab_int}.

```
tab_int <- tab_int / sum(tab_int)
tab_ins <- tab_ins / sum(tab_ins)
par(mfrow = c(1,2))
plot(as.numeric(tab_int), csmf_int, main="InterVA",
    xlab = "Proportion of most likely causes", ylab = "Estimated CSMF")
abline(c(0, 1), col = "red")
plot(as.numeric(tab_ins)[-62], csmf_ins[,1], main="InSilicoVA",
    xlab = "Proportion of most likely causes", ylab = "Estimated CSMF")
    abline(c(0, 1), col = "red")
```



- As you can see, the estimated CSMF and the proportion of most likely causes are similar, but not the same. This is because the CSMF are calculated in a different way depending on algorithms, and they are typically not calculated directly as the fraction of most likely cause assignments. As a recap here:
  - InterVA calculates CSMF using the likelihood of up to top 3 causes for each death, and small likelihood values are cut off into the 'undetermined' category.

- NBC calculates CSMF using the likelihood of all causes for each death.
- InSilicoVA calculates CSMF as a separate parameter in the algorithm, which on expectation usually equal the aggregation of the likelihood of all causes for each death, but with wider uncertainty bounds.
- Tariff is the only algorithm that calculates CSMF as the proportion of most likely causes. But SmartVA-Analyze has some additional heuristics in post-processing that may lead to different results.

Exercise 2

Here we only include the codes to download the PHMRC data within R, and randomly select 1000 deaths for testing, and 1000 deaths for training. You can choose a 'seed' different from below (12345). It makes sure that the random samples you get are fixed the next time you run the codes. (*But notice that sometimes you may not want those random numbers to be fixed.*)

```
PHMRC_all <- read.csv(getPHMRC_url("adult"))
set.seed(12345)
samples <- sample(1:dim(PHMRC_all)[1], 2000)
test <- PHMRC_all[samples[1:1000], ]
train <- PHMRC_all[samples[1001:2000], ]
```

The rest of the tasks can be done by following the lecture slides.