Introduction to openVA

Richard Li

Nov 9, 2018

Bloomberg Philanthropies

DATA FOR HEALTH INITIATIVE

CDC Foundation
Together our impact is greater

Vital Strategies

The Ohio State University
Institute for Population Research

In collaboration with the WHO VA Reference Group

Overview

openVA

Case study: WHO 2012 questionnaire

Case study: WHO 2016 questionnaire

Case study: PHMRC questionnaire

Some special notes on InSilicoVA

Summary

Exercises

Overview

What is the usual ‘pipeline’ of data analysis I

Typically, a data analysis is carried out with the following steps:

1. Scientific question
2. Obtain data
3. Processed data
4. Exploratory data analysis
5. Statistical analysis/modeling/prediction
6. Interpretation of results, write-up and reporting

In the previous lectures, we have examined (1) to (4), in the context of cause-of-death assignments using VA.

In this lecture, we are going to look at (5) in practice with more detail.

Arguably, step 5 is usually the most complicated in the pipeline.

What is the usual ‘pipeline’ of data analysis II

This step also usually seems to be most 'well-developed'.

It is very tempting to find a black-box, apply a single function, and get results.

The previous lecture introduced the mathematical/statistical intuitive behind different algorithms. In this lecture, we will introduce the implementations of these algorithms in `openVA`.

1. What data input is required
2. How to call different methods
3. What parameters you may want to set

Eventually we will wrap everything under the hood, so that routine usage only require one click. But it is important to know how to interactively analyze VA data and check for any inconsistent results from algorithms!

openVA

Using openVA

You should have all installed openVA. Try the following to see if there is any problem

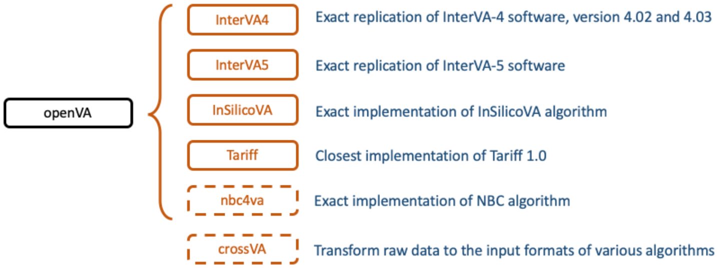
```r
library(openVA)
openVA_status()
```

```
## ------- Attached packages for openVA 1.0.7 -------

## v InSilicoVA 1.2.4
## v InterVA4   1.7.5
## v InterVA5   1.0.2
## v Tariff     1.0.4

## ----- Packages not attached for openVA 1.0.7 -----

## x nbc4va 1.1
```

```r
openVA_update()
```

```
## All required openVA packages up-to-date. Run openVA_status() for a c
```

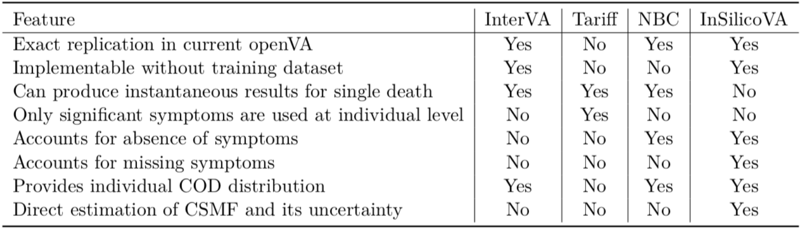
openVA: algorithms and key packages


openVA
InterVA4
Exact replication of InterVA-4 software, version 4.02 and 4.03
InterVA5
Exact replication of InterVA-5 software
InSilicoVA
Exact implementation of InSilicoVA algorithm
Tariff
Closest implementation of Tariff 1.0
nbc4va
Exact implementation of NBC algorithm
crossVA
Transform raw data to the input formats of various algorithms

Currently five R packages are supported:

- nbc4va needs to be installed separately,
- crossVA will be included,
- InterVA4 and InterVA5 are extended to work with customized data (with training datasets).

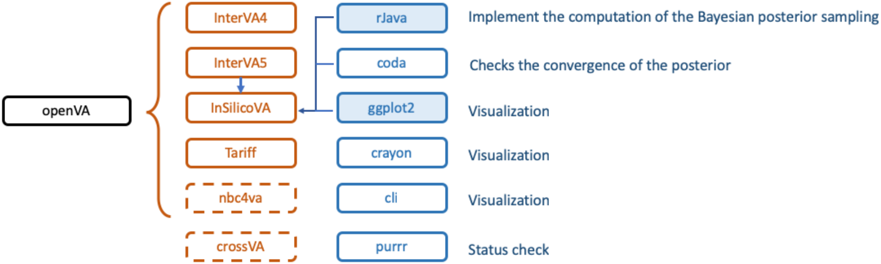
Summary of methods



Feature	InterVA	Tariff	NBC	InSilicoVA
Exact replication in current openVA	Yes	No	Yes	Yes
Implementable without training dataset	Yes	No	No	Yes
Can produce instantaneous results for single death	Yes	Yes	Yes	No
Only significant symptoms are used at individual level	No	Yes	No	No
Accounts for absence of symptoms	No	No	Yes	Yes
Accounts for missing symptoms	No	No	No	Yes
Provides individual COD distribution	Yes	No	Yes	Yes
Direct estimation of CSMF and its uncertainty	No	No	No	Yes

Figure 1: Summary of the main features of the supported algorithms

openVA: additional dependencies



Knowing the dependencies is useful when codes run into errors. The main computing dependency is the `rJava` package. It sometimes require additional configuration in order to install. `ggplot2` is the main visualization tool, but some base R plotting functions are also used.

Where to get help

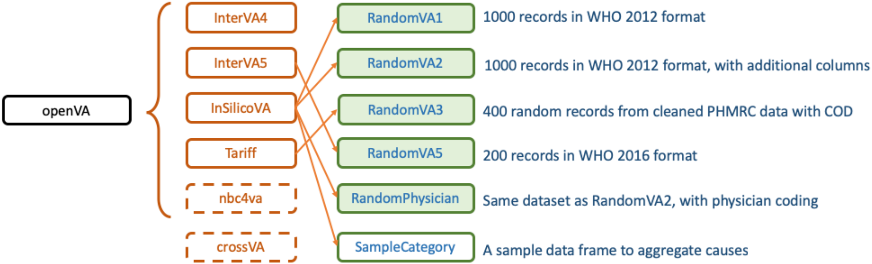
The dependency also means that sometimes when you try to get help, you need to look for a more specific function

For example, at the “See Also” section of `?codeVA`, you can find the function names for different individual algorithm calls.

So if you have questions about the InSilicoVA implementation, you can then go to `?insilico`.

Similarly, if you have questions about plotting function for InSilicoVA models, you may go to `?plotVA` and then be directed to `?plot.insilico`.

openVA: example datasets



openVA
InterVA4
InterVA5
InSilicoVA
Tariff
nbc4va
crossVA
RandomVA1
RandomVA2
RandomVA3
RandomVA5
RandomPhysician
SampleCategory
1000 records in WHO 2012 format
1000 records in WHO 2012 format, with additional columns
400 random records from cleaned PHMRC data with COD
200 records in WHO 2016 format
Same dataset as RandomVA2, with physician coding
A sample data frame to aggregate causes

In this lecture we will use these example datasets to demonstrate model fitting and result summarization.

- WHO 2012
- WHO 2016
- PHMRC (long format)

In the homework, you will work with some other more realistic datasets

Data preparation: WHO 2012 and 2016 data input

You should have already seen the default input formats for InterVA/InSilicoVA

- WHO 2012: ID and 245 symptoms
- WHO 2016: ID and 353 symptoms

The {yes, no, missing} are coded by

- WHO 2012: {"Y", empty string, "."}
- WHO 2016: {"y", "n", "-"}

When using openVA, both upper and lower cases are acceptable.

Missing can be coded by either '.' or '-' for WHO 2016 input.

But you need to be careful with NA in the dataset, since different people may view NA as either no or missing.

WHO 2012 and 2016 example dataset

We will start with the example datasets corresponding to WHO 2012 and WHO 2016 questionnaires. The formats below are what is expected by the InterVA software.

```r
data(RandomVA1)
RandomVA1[1:3, 1:9]
```

```
##   ID elder midage adult child under5 infant
## 1 d1     Y                                 
## 2 d2     Y                                 
## 3 d3            Y                          
##   neonate male
## 1            Y
## 2             
## 3            Y
```

```r
data(RandomVA5)
RandomVA5[1:3, 1:9]
```

```
##   ID i004a i004b i019a i019b i022a i022b i022c
## 1 d1     .     .     y     .     y     .     .
## 2 d2     .     .     .     y     y     .     .
## 3 d3     .     .     y     .     .     y     .
```

Data preparation: PHMRC data (long form)

At the end of this lecture, we will use PHMRC adult data to illustrate fitting all the methods on the same input data.

```
PHMRC_first1000 <- read.csv(getPHMRC_url("adult"),
    nrows = 1000)
head(PHMRC_first1000[, 1:9])
```

```
##     site module gs_code34
## 1 Mexico  Adult       K71
## 2     AP  Adult       G40
## 3     AP  Adult       J12
## 4 Mexico  Adult       J33
## 5     UP  Adult       I21
## 6     UP  Adult       X09
##                      gs_text34 va34 gs_code46
## 1                    Cirrhosis    6       K71
## 2                     Epilepsy   12       G40
## 3                    Pneumonia   26       J12
## 4                         COPD    8       J33
## 5 Acute Myocardial Infarction   17       I21
## 6                        Fires   15       X09
##                      gs_text46 va46 gs_code55
## 1                    Cirrhosis    8       K71
```

More data cleaning

And in case you may have data prepared with a wrong encoding scheme, you can use the ConvertData function to quickly correct it.

```
badData <- data.frame(id = c("d1", "d2"), symptom1 = c("Yes",
    "N"), symptom2 = c("Dk", "Y"), symptom3 = c("1",
    "0"))
badData
```

```
##   id symptom1 symptom2 symptom3
## 1 d1      Yes       Dk        1
## 2 d2        N        Y        0
```

```
cleanData <- ConvertData(badData, yesLabel = c("Yes",
    "Y", "1"), noLabel = c("N", "0"), missLabel = c("Dk"))
cleanData
```

```
##   id symptom1 symptom2 symptom3
## 1 d1        Y        .        Y
## 2 d2                 Y
```

Case study: WHO 2012 questionnaire

Fitting InterVA4 to the model

There are three parameters that are most significant in fitting InterVA-4 model to WHO 2012 data

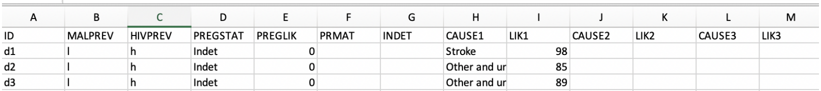
- `version`: can be 4.02 or 4.03. InterVA-4.03 fixes bugs in the previous release. We provide exact replications of both versions.
- `HIV` and `Malaria`: can be set to h (high), l (low), or v (very low). These changes the 'prior' prevalence of some causes.

```
fit_inter <- codeVA(data = RandomVA1, data.type = "WHO2012",
    model = "InterVA", version = "4.03", HIV = "h",
    Malaria = "l")
```

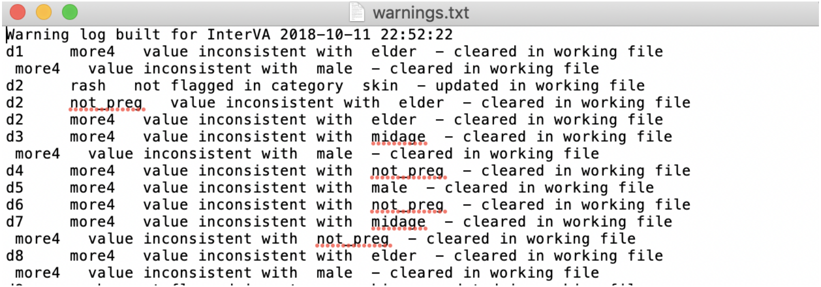
InterVA-4 style output can also be automatically saved to file by specifying a few additional arguments

```
fit_inter <- codeVA(data = RandomVA1, data.type = "WHO2012",
    model = "InterVA", version = "4.03", HIV = "h",
    Malaria = "l", write = TRUE, directory = "InterVA_ouput",
    filename = "Nov9testrun", output = "classic")
```

Fitting InterVA4 to the model



A	B	C	D	E	F	G	H	I	J	K	L	M
ID	MALPREV	HIVPREV	PREGSTAT	PREGLIK	PRMAT	INDET	CAUSE1	LIK1	CAUSE2	LIK2	CAUSE3	LIK3
d1	l	h	Indet	0			Stroke	98				
d2	l	h	Indet	0			Other and ur	85				
d3	l	h	Indet	0			Other and ur	89				



```
Warning log built for InterVA 2018-10-11 22:52:22
d1      more4   value inconsistent with  elder  - cleared in working file
 more4   value inconsistent with  male  - cleared in working file
d2      rash   not flagged in category  skin  - updated in working file
d2      not_preg   value inconsistent with  elder  - cleared in working file
d2      more4   value inconsistent with  elder  - cleared in working file
d3      more4   value inconsistent with  midage  - cleared in working file
 more4   value inconsistent with  male  - cleared in working file
d4      more4   value inconsistent with  not_preg  - cleared in working file
d5      more4   value inconsistent with  male  - cleared in working file
d6      more4   value inconsistent with  not_preg  - cleared in working file
d7      more4   value inconsistent with  midage  - cleared in working file
 more4   value inconsistent with  not_preg  - cleared in working file
d8      more4   value inconsistent with  elder  - cleared in working file
 more4   value inconsistent with  male  - cleared in working file
```

Fitting InterVA4 to the model

```
summary(fit_inter)
```

```
## InterVA-4 fitted on 1000 deaths
## CSMF calculated using reported causes by InterVA-4 only
## The remaining probabilities are assigned to 'Undetermined'
##
## Top 5 CSMFs:
##  cause                            likelihood
##  Undetermined                     0.1522
##  HIV/AIDS related death           0.1234
##  Stroke                           0.0731
##  Other and unspecified infect dis 0.0620
##  Reproductive neoplasms MF        0.0583
```

Fitting InSilicoVA to the model

To fit InSilicoVA on the same dataset, we still use `codeVA` function The main arguments to InSilicoVA is the `Nsim`, the number of iterations of posterior draws.

- I typically start with $10,000$ iterations, which can take one to several minutes to run depending on the size of the dataset.

```r
fit_ins <- codeVA(RandomVA1, data.type = "WHO2012",
    model = "InSilicoVA", Nsim = 10000, auto.length = FALSE)
```

The stochastic nature of the sampling-based approach adopted by InSilicoVA makes it flexible with nice characterization of uncertainties.
But it also means the algorithm may need to be tunned with more care.

Fitting InSilicoVA to the model

First, the convergence depends on how long the algorithm is run

- `Nsim`: The total number of iterations to run the algorithm.
- `auto.length`: Whether or not to automatic double the number of iterations at the end if convergence test fails.

Second, the convergence also depends on how many proposed new parameters are accepted.

- This is directly reflected in `jump.scale` and the 'acceptance rate' printed to the screen when running InSilicoVA.
- If `jump.scale` is too large, at each iteration, the algorithm 'tries' more wild guesses, leading to many of such guesses rejected. This can waste many iterations of sampling.
- If `jump.scale` is too small, at each iteration, the algorithm makes new guesses that are very similar to current values. This may prevent the algorithm to explore the right range of parameters.
- Ideally, we want to 'tune' the algorithm so that the acceptance rate is neither too large or too small. 20% to 25% is usually recommended.
- In practice, typically as long as it is not very small (<5%) or very large (>50%), we have found InSilicoVA to be mostly robust, at least for causes with higher prevalence.

Fitting InSilicoVA to the model

If you are interested, you can try the following:

Increased `jump.scale` and decreased acceptance probability.

```
fit <- codeVA(RandomVA1, data.type = "WHO2012", model = "InSilicoVA",
    Nsim = 10000, auto.length = FALSE, jump.scale = 0.2)
```

Let the algorithm decide whether to keep running after the specified number of iterations.

```
fit <- codeVA(RandomVA1, data.type = "WHO2012", model = "InSilicoVA",
    Nsim = 10000, auto.length = TRUE)
```

Fitting InSilicoVA to the model

Similar to before, summary of the results can be obtained by

```
summary(fit_ins)
```

```
## InSilicoVA Call:
## 1000 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
## Other and unspecified infect dis  0.2652
## HIV/AIDS related death            0.1018
## Renal failure                     0.1015
## Other and unspecified neoplasms   0.0615
## Other and unspecified cardiac dis 0.0583
## Digestive neoplasms               0.0496
## Acute resp infect incl pneumonia  0.0481
## Pulmonary tuberculosis            0.0391
## Stroke                            0.0382
## Other and unspecified NCD         0.0344
##                                   Std.Error
## Other and unspecified infect dis     0.0165
## HIV/AIDS related death               0.0095
```

Obtain CSMF summary of the fitted models

Once we fit the model, we may want to obtain summary statistics of the estimated model.

```
csmf_inter <- getCSMF(fit_inter)
head(csmf_inter)
```

```
##            Sepsis (non-obstetric)
##                       0.004667518
## Acute resp infect incl pneumonia
##                       0.051667517
##            HIV/AIDS related death
##                       0.123415869
##               Diarrhoeal diseases
##                       0.008841005
##                           Malaria
##                       0.013246687
##                           Measles
##                       0.000000000
```

```
csmf_ins <- getCSMF(fit_ins)
head(csmf_ins)
```

```
##                                          Mean
## Sepsis (non-obstetric)           2.384478e-04
## Acute resp infect incl pneumonia 4.809903e-02
## HIV/AIDS related death           1.017784e-01
```

Obtain individual summary

We may also look more closely into some individuals

```r
summary(fit_inter, id = "d1")
```

```
## InterVA-4 fitted top 5 causes for death ID: d1
##
##  Cause                                Likelihood
##  Stroke                               0.9805
##  Renal failure                        0.0078
##  Digestive neoplasms                  0.0039
##  Other and unspecified neoplasms      0.0025
##  Other and unspecified external CoD 0.0010
```

```r
summary(fit_ins, id = "d1")
```

```
## InSilicoVA fitted top  causes for death ID: d1
## Credible intervals shown: %
##                                           Mean Lower
## Renal failure                        0.5784170    NA
## Other and unspecified neoplasms      0.2103543    NA
## Other and unspecified infect dis     0.1518149    NA
## Other and unspecified NCD            0.0305403    NA
## Stroke                               0.0170459    NA
## Tetanus                              0.0068023    NA
## Other and unspecified cardiac dis 0.0021873    NA
## Oral neoplasms                       0.0014231    NA
```

Obtain individual summary

As suggested in the warning message, for InSilicoVA, uncertainties associated with individual probabilities are not calculated by default to save computation time.

```
fit_ins <- updateIndiv(fit_ins, CI = 0.95)
summary(fit_ins, id = "d1")
```

```
## InSilicoVA fitted top  causes for death ID: d1
## Credible intervals shown: 95%
##                                         Mean
## Renal failure                     0.5784170
## Other and unspecified neoplasms   0.2103543
## Other and unspecified infect dis  0.1518149
## Other and unspecified NCD         0.0305403
## Stroke                            0.0170459
## Tetanus                           0.0068023
## Other and unspecified cardiac dis 0.0021873
## Oral neoplasms                    0.0014231
## Digestive neoplasms               0.0010339
## Respiratory neoplasms             0.0001723
##                                        Lower
## Renal failure                     0.5073320
```

Obtain most likely cause of death assignments

```
cod_inter <- getTopCOD(fit_inter)
head(cod_inter)
```

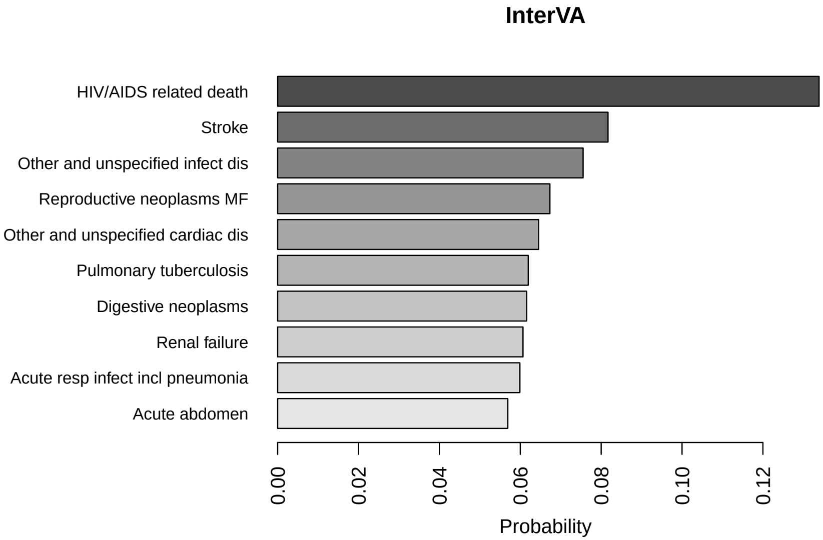
```
##   ID                             cause
## 1 d1                            Stroke
## 2 d2 Other and unspecified cardiac dis
## 3 d3 Other and unspecified cardiac dis
## 4 d4            HIV/AIDS related death
## 5 d5            Pulmonary tuberculosis
## 6 d6            HIV/AIDS related death
```

```
cod_ins <- getTopCOD(fit_ins)
head(cod_ins)
```

```
##   ID                             cause
## 1 d1                     Renal failure
## 2 d2  Other and unspecified infect dis
## 3 d3 Other and unspecified cardiac dis
## 4 d4            HIV/AIDS related death
## 5 d5            Pulmonary tuberculosis
## 6 d6                     Renal failure
```

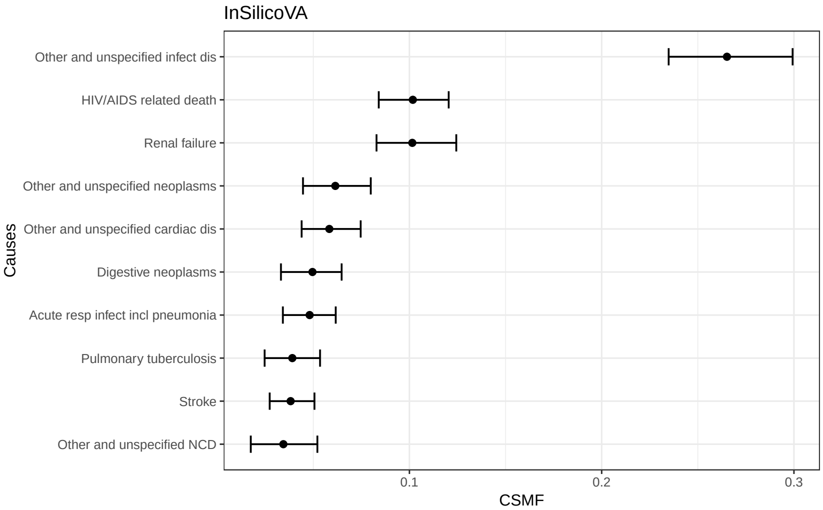
Visualization: CSMF

```
plotVA(fit_inter, title = "InterVA")
```



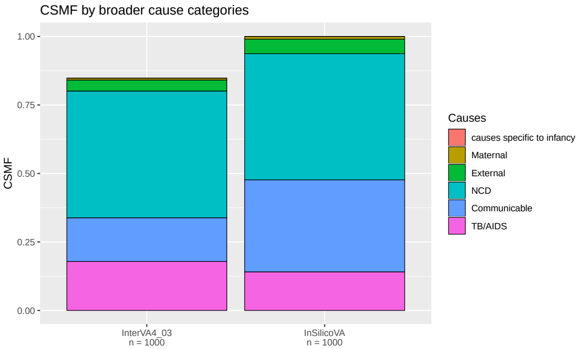
Visualization: CSMF

```
plotVA(fit_ins, title = "InSilicoVA", bw = TRUE)
```



More visualization: stack plot of CSMF

```
compare <- list(InterVA4_03 = fit_inter, InSilicoVA = fit_ins)
stackplotVA(compare, sample.size.print = TRUE, xlab = "",
    angle = 0)
```



Why CSMF from InterVA does not sum to 1?

More visualization: stack plot of CSMF

You can customize how to group the causes for your own needs

```
data(SampleCategory)
SampleCategory[1:3, ]
```

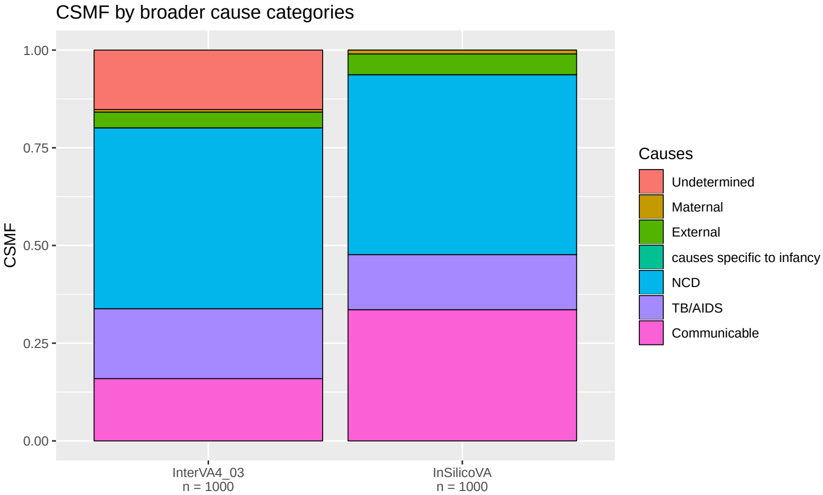
```
##                             InterVA    Physician
## 1           Sepsis (non-obstetric) Communicable
## 2 Acute resp infect incl pneumonia Communicable
## 3           HIV/AIDS related death      TB/AIDS
```

```
grouping <- SampleCategory
grouping[, 1] <- as.character(grouping[, 1])
grouping <- rbind(grouping, c("Undetermined", "Undetermined"))
tail(grouping)
```

```
##                               InterVA
## 56                  Obstructed labour
## 57           Pregnancy-related sepsis
## 58               Anaemia of pregnancy
## 59                    Ruptured uterus
## 60 Other and unspecified maternal CoD
## 61                       Undetermined
##       Physician
```

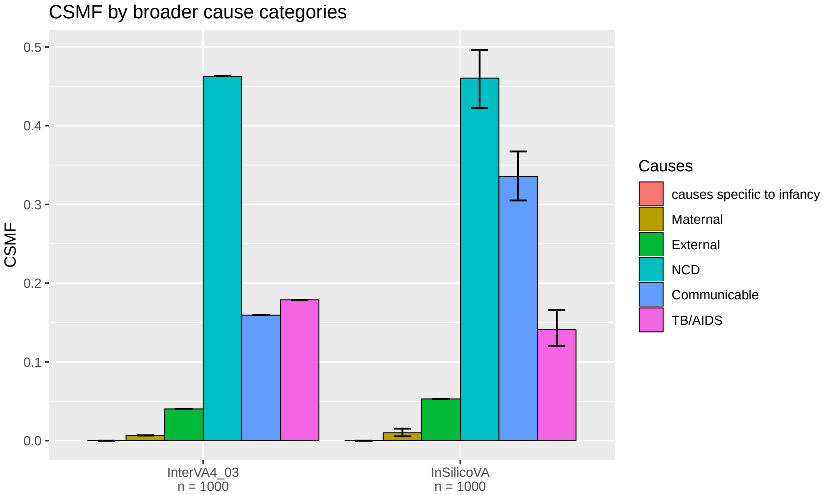
More visualization: stack plot of CSMF

```
stackplotVA(compare, sample.size.print = TRUE, xlab = "",
    angle = 0, grouping = grouping)
```



More visualization: stack plot of CSMF

```r
stackplotVA(compare, sample.size.print = TRUE, xlab = "",
    angle = 0, type = "dodge")
```



Case study: WHO 2016 questionnaire

Fitting InterVA5

In general, fitting InterVA-5 and InSilicoVA with the WHO 2016 data is very similar to before.
The computation overhead is slightly more than before, due to the current implementation of data checking procedure.
A faster data checking procedure will be updated in the near future.
Stay tunned!

```
data(RandomVA5)
fit_inter2016 <- codeVA(RandomVA5, data.type = "WHO2016",
    model = "InterVA", version = "5.0", HIV = "h",
    Malaria = "l", write = FALSE)
```

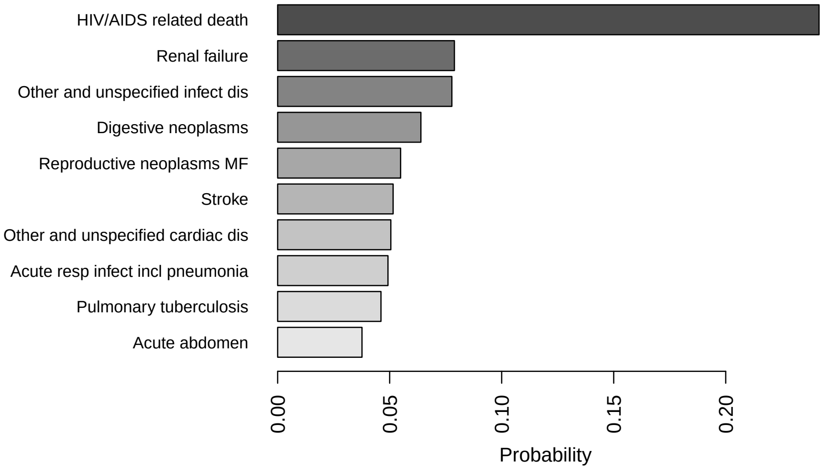
Obtain CSMF summary

```
summary(fit_inter2016)
```

```
## InterVA5 fitted on 200 deaths
## CSMF calculated using reported causes by InterVA5 only
## The remaining probabilities are assigned to 'Undetermined'
##
## Top 5 CSMFs:
##  cause                              likelihood
##  HIV/AIDS related death             0.2294
##  Undetermined                       0.1261
##  Other and unspecified infect dis 0.0714
##  Renal failure                      0.0706
##  Digestive neoplasms                0.0558
##
## Top 5 Circumstance of Mortality Category:
##  cause      likelihood
##  Knowledge  0.365
##  Culture    0.240
##  Multiple   0.215
##  Inevitable 0.110
##  Resources  0.035
```

Visualize CSMF

```
plotVA(fit_inter2016)
```



Fitting InSilicoVA

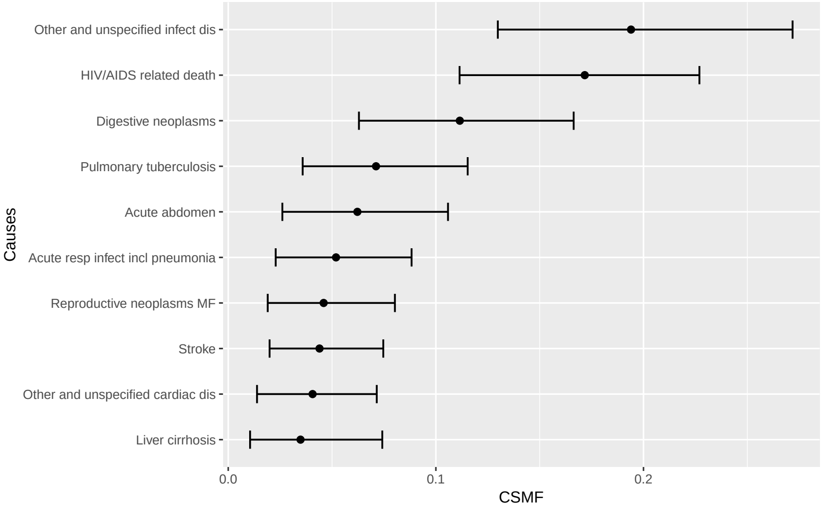
```r
fit_ins2016 <- codeVA(RandomVA5, data.type = "WHO2016",
    model = "InSilicoVA", Nsim = 10000, auto.length = FALSE,
    jump.scale = 0.2)
```

```r
summary(fit_ins2016)
```

```
## InSilicoVA Call:
## 200 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
## Other and unspecified infect dis  0.1940
## HIV/AIDS related death            0.1717
## Digestive neoplasms               0.1115
## Pulmonary tuberculosis            0.0712
## Acute abdomen                     0.0622
## Acute resp infect incl pneumonia  0.0519
## Reproductive neoplasms MF         0.0460
## Stroke                            0.0440
## Other and unspecified cardiac dis 0.0406
## Liver cirrhosis                   0.0348
```

Visualize CSMF

```
plotVA(fit_ins2016)
```



Case study: PHMRC questionnaire

PHMRC gold-standard data

In this section, we will bring back the other two algorithms and look at methods based on training data.

As you have already known, there is no magic different between training data or physician provided conditional probabilities. When reflected in the algorithm, they are just different forms of symptom-cause-information (SCI). Thus both InterVA and InSilicoVA can be extended to this case.

Since currently there is no other good source of training data, we use the PHMRC gold standard dataset as an illustration. We use Mexico city as the test set and the rest of the data as training set.

```r
PHMRC_all <- read.csv(getPHMRC_url("adult"))
Mexico <- which(PHMRC_all$site == "Mexico")
test <- PHMRC_all[Mexico, ]
train <- PHMRC_all[-Mexico, ]
dim(test)
```

```
## [1] 1586  946
```

```r
dim(train)
```

```
## [1] 6255  946
```

Fitting InterVA and InSilicoVA

The arguments is similar to before, but now we need to specify training and testing data, as well as the column name for the cause of death labels in the training data.

For both InterVA and InSilicoVA, we also have an option, covert.type, to calculate the 'rankings' of conditional probability from the training data.

```
fit_phmrc_inter <- codeVA(data = test, data.type = "PHMRC",
                         model = "InterVA",
                         data.train = train, causes.train = "gs_text34",
                         phmrc.type = "adult")
fit_phmrc_ins <- codeVA(data = test, data.type = "PHMRC",
                          model = "InSilicoVA",
                          data.train = train, causes.train = "gs_text34",
                          phmrc.type = "adult",
                          jump.scale = 0.05, convert.type = "fixed",
                          Nsim=10000, auto.length = FALSE)
```

Fitting Tariff and NBC

```
fit_phmrc_tar <- codeVA(data = test, data.type = "PHMRC",
    model = "Tariff", data.train = train, causes.train = "gs_text34",
    phmrc.type = "adult")
```

```
fit_phmrc_nbc <- codeVA(data = test, data.type = "PHMRC",
    model = "NBC", data.train = train, causes.train = "gs_text34",
    phmrc.type = "adult")
```

Two things to notice about Tariff and NBC:

- Again, Tariff implementation is not exact, as we do not have enough information. But an R version of Tariff might be coming in the near future.
- Both methods consider missing symptom as absent. All missing symptoms are first transformed into absent when calling the function.

Summary of results

Again, you can use `getTopCOD` and `getIndivProb` functions as before.
In this particular truth, we may also calculate CSMF accuracy for each methods, since we know the true causes in the testing data.

```r
csmf.tariff <- getCSMF(fit_phmrc_tar)
csmf.interva <- getCSMF(fit_phmrc_inter)
csmf.nbc <- getCSMF(fit_phmrc_nbc)
csmf.insilico <- getCSMF(fit_phmrc_ins)
csmf.all <- cbind(Tariff = csmf.tariff,
                  InterVA = csmf.interva[1:34],
                  NBC = csmf.nbc,
                  InSilicoVA = csmf.insilico[, "Mean"])
```

```r
csmf.true <- table(test$gs_text34)
csmf.true <- csmf.true[names(csmf.tariff)]
csmf.true <- as.numeric(csmf.true/sum(csmf.true))
```

CSMF accuracy

```r
getCSMF_accuracy(csmf.tariff, csmf.true)
```

```
## [1] 0.6128625
```

```r
getCSMF_accuracy(csmf.interva, csmf.true, "Undetermined")
```

```
## [1] 0.6930848
```

```r
getCSMF_accuracy(csmf.nbc, csmf.true)
```

```
## [1] 0.7225725
```

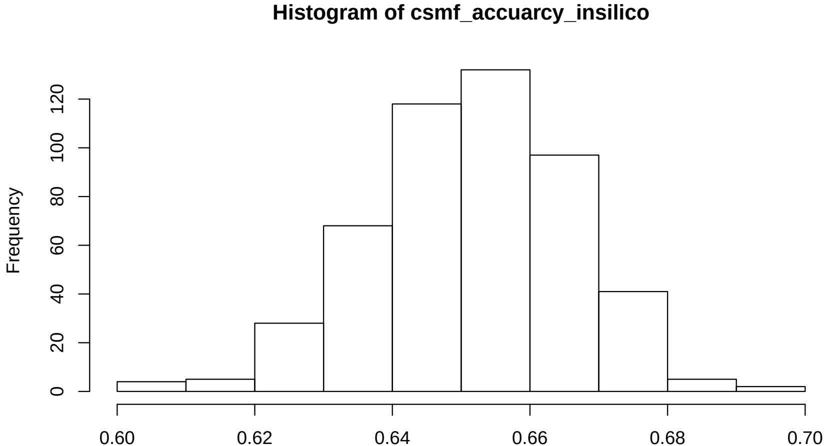
```r
getCSMF_accuracy(csmf.insilico[, "Mean"], csmf.true)
```

```
## [1] 0.6537247
```

CSMF accuracy

And as a reminder, we can also calculate the uncertainties around the metrics for InSilicoVA as well.

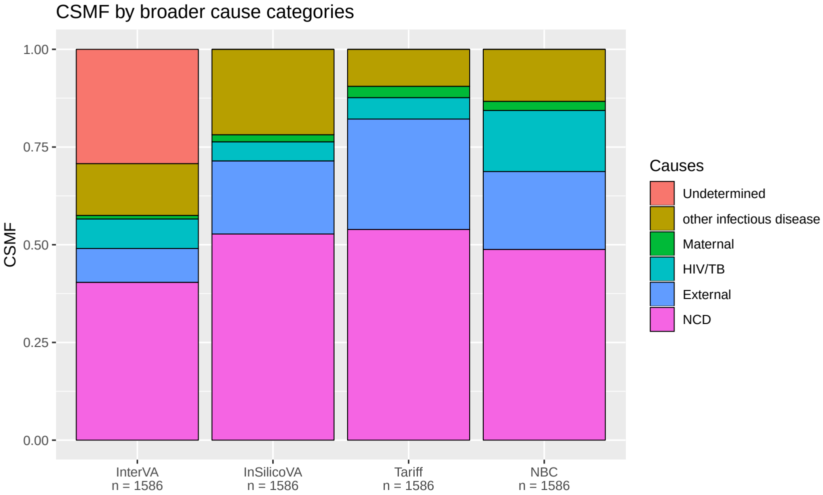
```
csmf_accuarcy_insilico <- getCSMF_accuracy(fit_phmrc_ins,
    csmf.true)
hist(csmf_accuarcy_insilico)
```



Comparison

```
cod.phmrc <- c(unique(as.character(train[, "gs_text34"])),
               "Undetermined")
group.phmrc <- c(rep("NCD", 3), "External", "NCD", "HIV/TB",
                "NCD", "Maternal", "External", "NCD",
                "External", "NCD", "External", rep("NCD", 2),
                "other infectious disease", "NCD", "External",
                "NCD", "HIV/TB", "other infectious disease",
                rep("NCD", 3), "External", rep("NCD", 2),
                rep("External", 2), rep("NCD", 4),
                "External", "Undetermined")
grouping2 <- cbind(cod.phmrc, group.phmrc)
compare <- list(InterVA = fit_phmrc_inter,
                InSilicoVA = fit_phmrc_ins,
                Tariff = fit_phmrc_tar,
                NBC = fit_phmrc_nbc)
stackplotVA(compare, sample.size.print = TRUE,
            xlab = "", angle = 0,
            grouping = grouping2)
```

Comparison


CSMF by broader cause categories
CSMF
1.00
0.75
0.50
0.25
0.00
InterVA
n = 1586
InSilicoVA
n = 1586
Tariff
n = 1586
NBC
n = 1586
Causes
Undetermined
other infectious disease
Maternal
HIV/TB
External
NCD

Some special notes on InSilicoVA

InSilicoVA and CSMF

This is a very brief discussion of some of the more technical questions about InSilicoVA.

Perhaps the most conceptually different aspect of InSilicoVA, compared to the other algorithms is how CSMF is viewed.

Let us first look at how CSMF is calculated in other algorithms

- InterVA: take **up to top three** causes and aggregate their probabilities.
- Tariff: count the **number of assigned causes**, and calculate their fractions.
- NBC: take the average of the full **individual probabilities**.

The common theme is the CSMF can be directly derived from individual results. However, InSilicoVA parameterize CSMF as a separate set of parameters to be learned from the data.

Population estimates

To see what it means, consider two datasets, both randomly sampled from a large population:

- a small dataset with 100 observations
- a large dataset by exactly repeating the small dataset 10 times.

For any deterministic algorithm, the distribution of causes in both datasets should match exactly.

But knowing there are more data may change what we believe about the unknown population: we may be more certain about our estimators.

Essentially this is the idea behind the InSilicoVA logic: our observations are samples from a larger population, and CSMF measures the distribution of causes in that population.

Aggregated CSMF and Subpopulations

What this implies is that if you take the individual probabilities of a subset of the data, say, all female deaths.

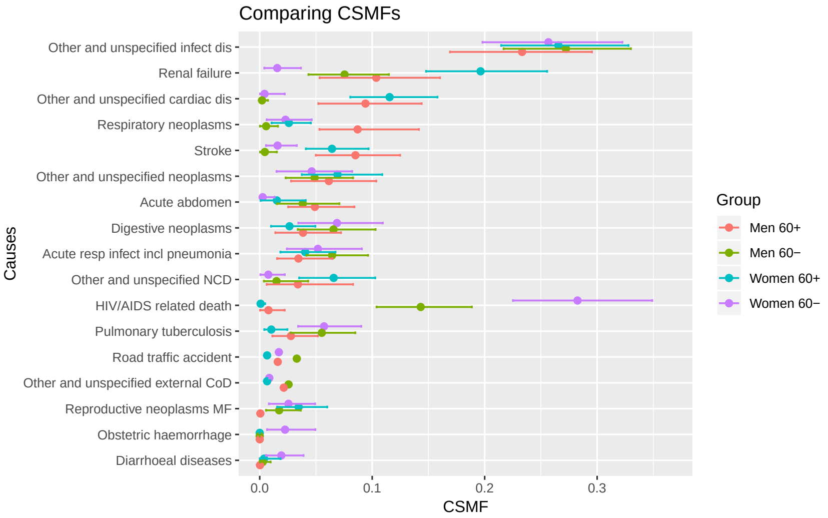
And you aggregate them to get a female-specific CSMF. The uncertainty is reduced, since we ignored the uncertainty due to the fact that the observations being a sample of the population.

In `InSilicoVA`, there is an option to specify subpopulation, which estimates separate CSMF vector for the specified groups.

```r
data(RandomVA2)
fit_sub_ins <- codeVA(RandomVA2, model = "InSilicoVA",
    subpop = list("sex", "age"), indiv.CI = 0.95, Nsim = 10000,
    auto.length = FALSE)
summary(fit_sub_ins)
```

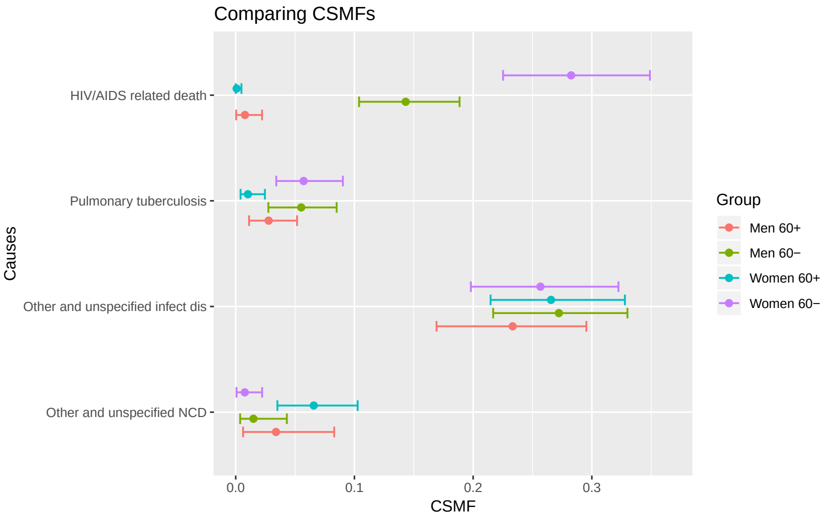
Subpopulation

```
plotVA(fit_sub_ins, type = "compare", title = "Comparing CSMFs")
```



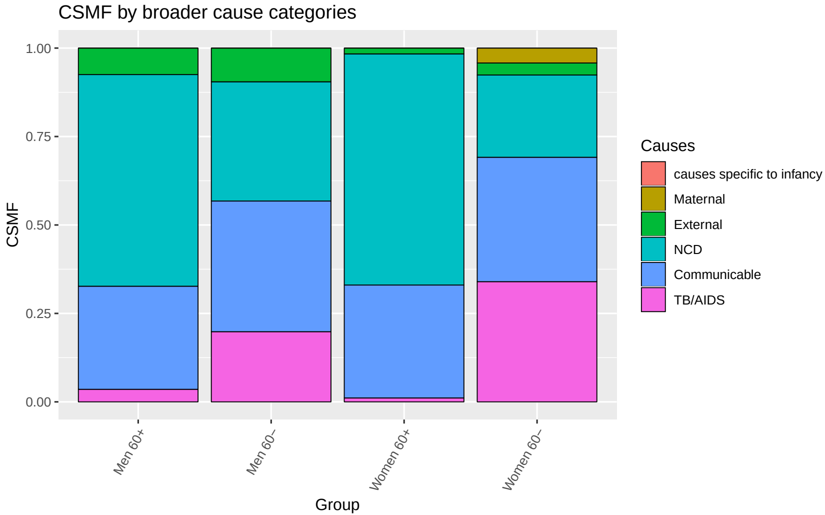
Subpopulation

```
plotVA(fit_sub_ins, type = "compare", title = "Comparing CSMFs",
    causelist = c("HIV/AIDS related death", "Pulmonary tuberculosis",
        "Other and unspecified infect dis", "Other and unspecified NCD"
```



Subpopulation

```
stackplot(fit_sub_ins)
```



Aggregated CSMF

However, although subpopulation specification is recommended, in practice, sometimes the sample size within each subpopulation may be too small.

We can still get the aggregated CSMF directly from InSilicoVA fitted in the standard way, but we should know that the uncertainty is likely underestimated.

```r
fit_sub0_ins <- codeVA(RandomVA2, model = "InSilicoVA",
    Nsim = 10000, auto.length = FALSE)
agg.by.sex.age <- get.indiv(data = RandomVA2, fit_sub0_ins,
    CI = 0.95, is.aggregate = TRUE, by = list("sex",
        "age"))
```

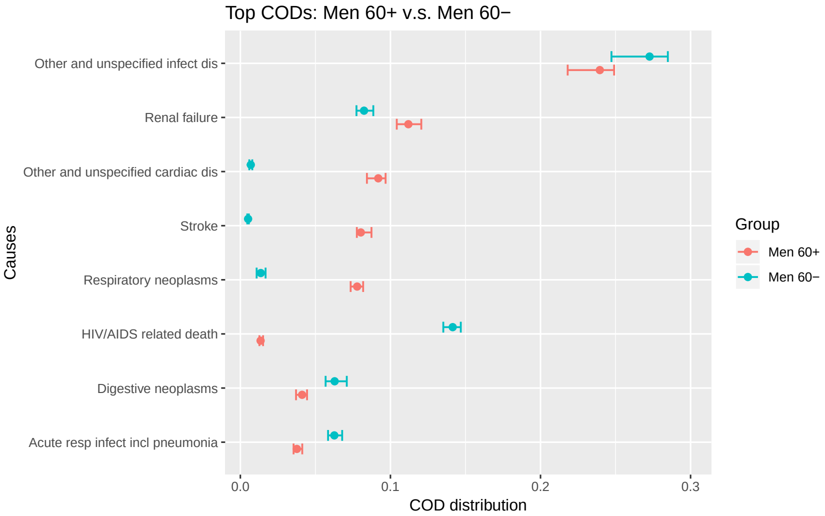
Aggregated CSMF

```r
head(agg.by.sex.age$mean)
```

```
##                                       Men 60+
## Sepsis (non-obstetric)           4.708987e-04
## Acute resp infect incl pneumonia 3.766775e-02
## HIV/AIDS related death           1.348041e-02
## Diarrhoeal diseases              2.205966e-04
## Malaria                          6.708219e-04
## Measles                          1.318893e-10
##                                       Men 60-
## Sepsis (non-obstetric)           8.272100e-04
## Acute resp infect incl pneumonia 6.260964e-02
## HIV/AIDS related death           1.414926e-01
## Diarrhoeal diseases              3.894277e-03
## Malaria                          3.605837e-04
## Measles                          2.324138e-09
##                                     Women 60+
## Sepsis (non-obstetric)           1.410778e-04
## Acute resp infect incl pneumonia 4.458290e-02
## HIV/AIDS related death           2.528444e-03
## Diarrhoeal diseases              3.499877e-03
## Malaria                          6.497071e-04
```

Aggregated CSMF

```
indivplot(agg.by.sex.age, which.plot = list("Men 60+",
    "Men 60-"), top = 5, title = "Top CODs: Men 60+ v.s. Men 60-")
```



Impossible causes

An additional topic recently come to our attention is the determination of impossible causes. Basically we want to enforce some screening rule so that given certain symptoms, some causes of death should not be considered possible.

In the latest version, we introduced three different rules for WHO 2016 inputs.

- **subset**: For any death, we exclude any causes with Pr(symptom present | cause) = 0 or 1 for symptoms directly related to age and gender.
- **InterVA-like**: For any death, we exclude any causes with Pr(symptom present | cause) = 0 for any symptom (after negating a small number of symptoms where absence if considered the significant response).
- **all** For any death, we exclude any causes with Pr(symptom present | cause) = 0 or 1 for any symptom.

Impossible causes

List of symptoms for ‘subset’ option:
Was he male?
Was she female?
Was s(he) aged 65 years or more at death?
Was s(he) aged 50 to 64 years at death?
Was s(he) aged 15 to 49 years at death?
Was s(he) aged 5-14 years at death?
Was s(he) aged 1 to 4 years at death?
Was s(he) aged 1 to 11 months at death?
Was s(he) aged $<$ 1 month (28 days) at death?
Was s(he) a live baby who died within 24 hours of birth?
Was s(he) a baby who died between 24 and 48 hours of birth?
Was s(he) a baby who died more than 48 hours from birth, but within the first week?
Was s(he) a baby who died after the first week, but within the first month?
Was she a woman aged 12-19 years at death?
Was she a woman aged 20-34 years at death?
Was she a woman aged 35 to 49 years at death?

If the baby didn’t show any sign of life, was it born dead?

I’m happy to share the full list of check rules if anyone is interested!

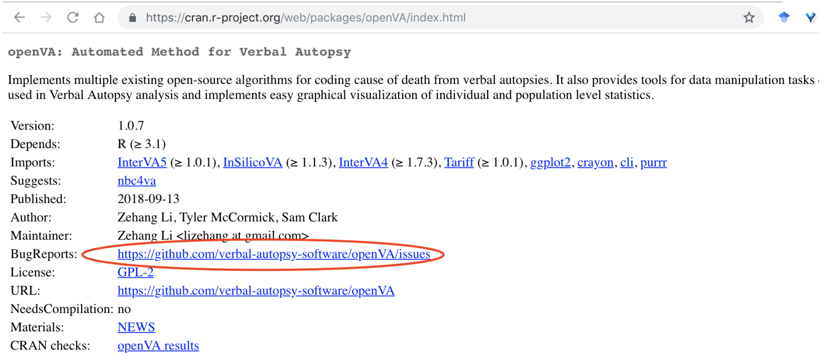
Summary

What happens if you run into errors and cannot fix?

It is natural bugs and unhandled exceptions may exist in the packages.

You can report bugs/issues from the link on CRAN package page.

For example, if there is a problem with openVA that you would like to report:



https://cran.r-project.org/web/packages/openVA/index.html

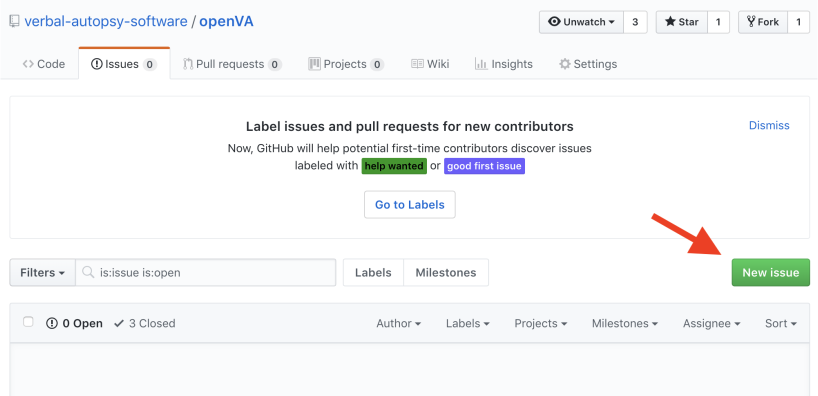
openVA: Automated Method for Verbal Autopsy

Implements multiple existing open-source algorithms for coding cause of death from verbal autopsies. It also provides tools for data manipulation tasks used in Verbal Autopsy analysis and implements easy graphical visualization of individual and population level statistics.

Version:	1.0.7
Depends:	R (≥ 3.1)
Imports:	InterVA5 (≥ 1.0.1), InSilicoVA (≥ 1.1.3), InterVA4 (≥ 1.7.3), Tariff (≥ 1.0.1), ggplot2, crayon, cli, purrr
Suggests:	nbc4va
Published:	2018-09-13
Author:	Zehang Li, Tyler McCormick, Sam Clark
Maintainer:	Zehang Li <lizehang at gmail.com>
BugReports:	https://github.com/verbal-autopsy-software/openVA/issues
License:	GPL-2
URL:	https://github.com/verbal-autopsy-software/openVA
NeedsCompilation:	no
Materials:	NEWS
CRAN checks:	openVA results

What happens if you run into errors and cannot fix?

It will take you to our github repository. Same can be done for other specific packages as well.



Label issues and pull requests for new contributors

Now, GitHub will help potential first-time contributors discover issues labeled with help wanted or good first issue

Summary of this lecture

1. Structure of `openVA` and where to get help.
2. Different input/output between algorithms.
3. Algorithms in action: `codeVA`, `plotVA`, etc.
4. Basic result extractions and visualization.
5. For the next lecture: What do we do after running the algorithms and seeing the results?

Exercises

Model fitting exercise I

Exercise 1: WHO 2016 data

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.
2. Using both InterVA5 and InSilicoVA, obtain the CSMF estimates, individual probabilities of all causes, and most likely causes for each individual.
3. Find out how many records dropped out of the analysis.
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.

Model fitting exercise II

Exercise 2: PHMRC data

1. Download the PHMRC data within R
2. Randomly select 1000 deaths for testing, and 1000 deaths for training.
3. Fit InterVA, InSilicoVA, NBC, and Tariff to the selected data.
4. Compare the estimated CSMF using four methods.
5. If the comparison is not visually clear, aggregate the causes to a higher level *(You may decide how to group the causes that makes sense. It does not have to be the same grouping as in the slides)*, and visualize the comparison of the estimated CSMF at the higher level.
6. For each algorithm, calculate CSMF accuracy, and the proportion of deaths assigned the correct cause.
7. Explore what other tasks you find helpful in examining the results.

Model fitting exercise: feedback

1. Please let us know if you run into errors and cannot fix. Make sure when you do, let us know (1) what dataset you use; (2) what codes you have run; (3) at which step the error occurs and what are the error messages.
2. Please let us know if there is anything tasks you want to do with data or results that are not covered in this lecture.

If you have any feedback/questions about algorithms and model fitting, send an email to *lizehang@gmail.com.*

We will discuss findings on Monday.