Exploratory analysis based on openVA output

Richard Li

Nov 12, 2018

In collaboration with the WHO VA Reference Group

Feedback from running algorithms

Interpret openVA outputs

Exploratory analysis of model output

Exercise

Feedback from running algorithms

Some common problems

Selected topics from feedback.

Interpret openVA outputs

Recap

Remember these steps for a typical data analysis:

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.

We are finally at the last step. . . except, it is *not* our last step.

No matter how careful we are in the first five steps, there are usually issues and mistakes:

- There could be lack of fit by the statistical model.
- There could be errors in data cleaning.
- The data may require different model parameters.
- Software may contain bugs.

Prepare output for further analysis

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.

This feedback loop is as important as any other steps.

After models are estimated, the direction of this scientific process is usually reversed for analysts.

Start from the report, can we illustrate what is going on with data and methods?

Recap: simple model fitting

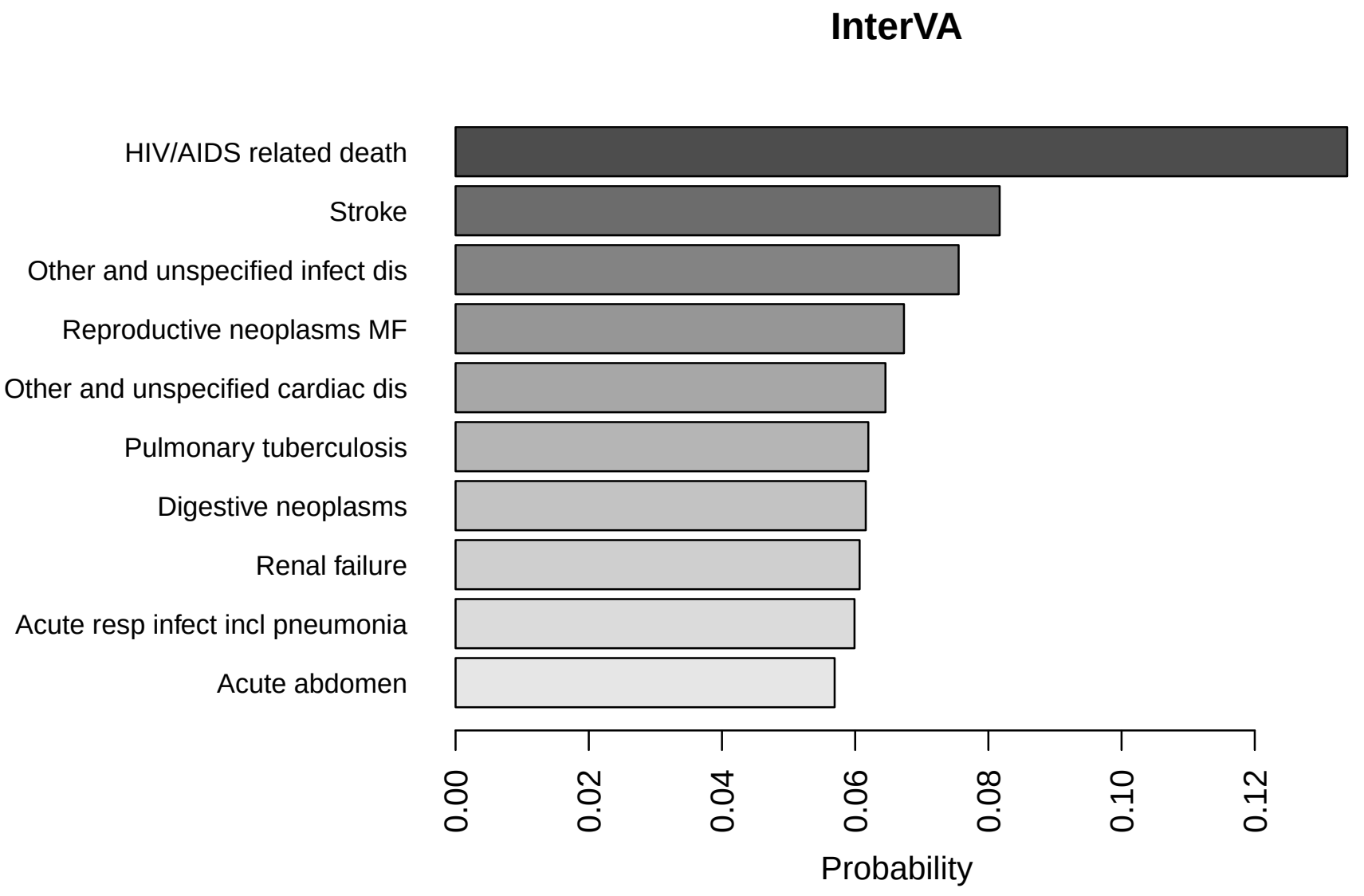
```
library(openVA)
data(RandomVA1)
```

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

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

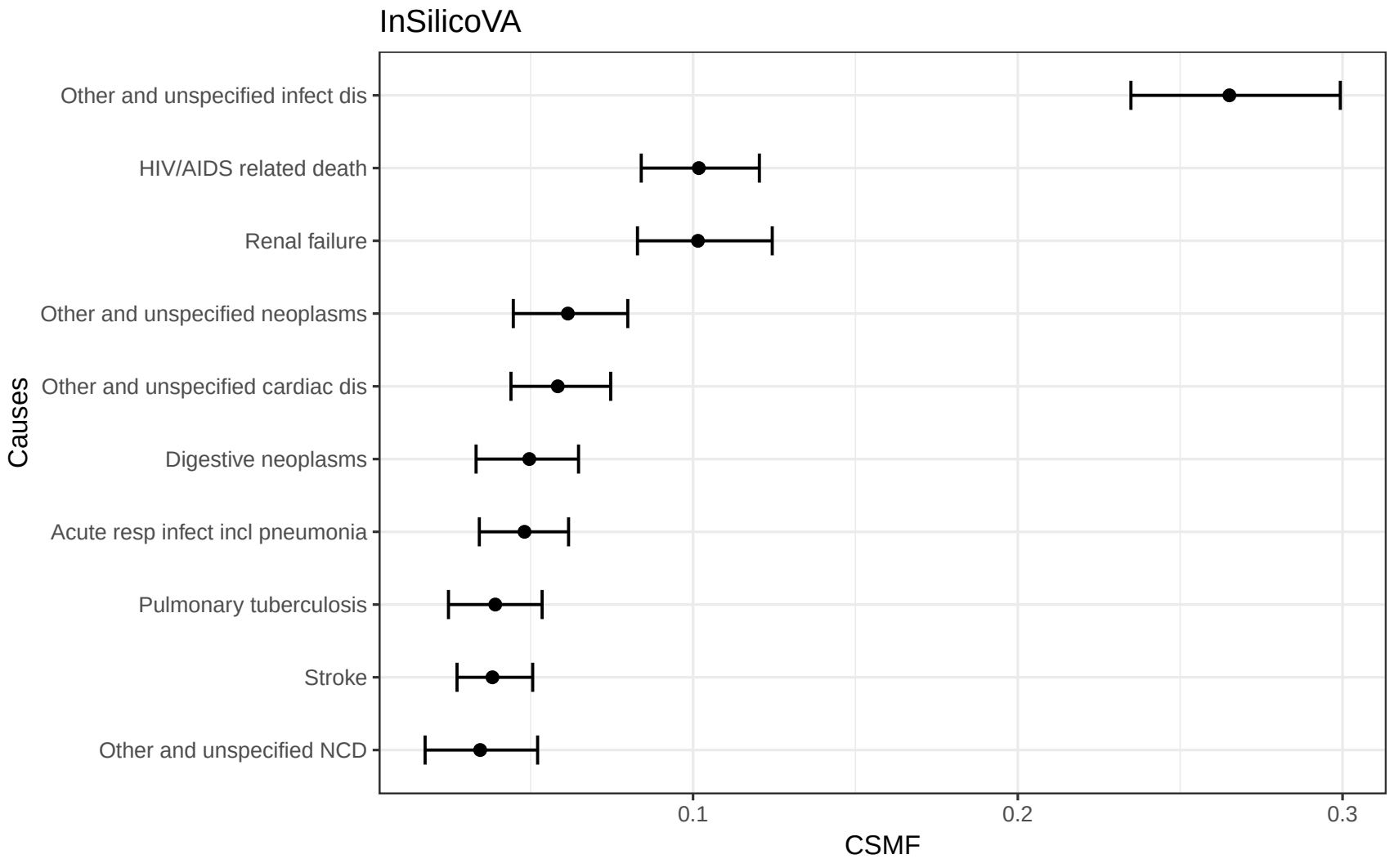
Recap: CSMF comparison

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



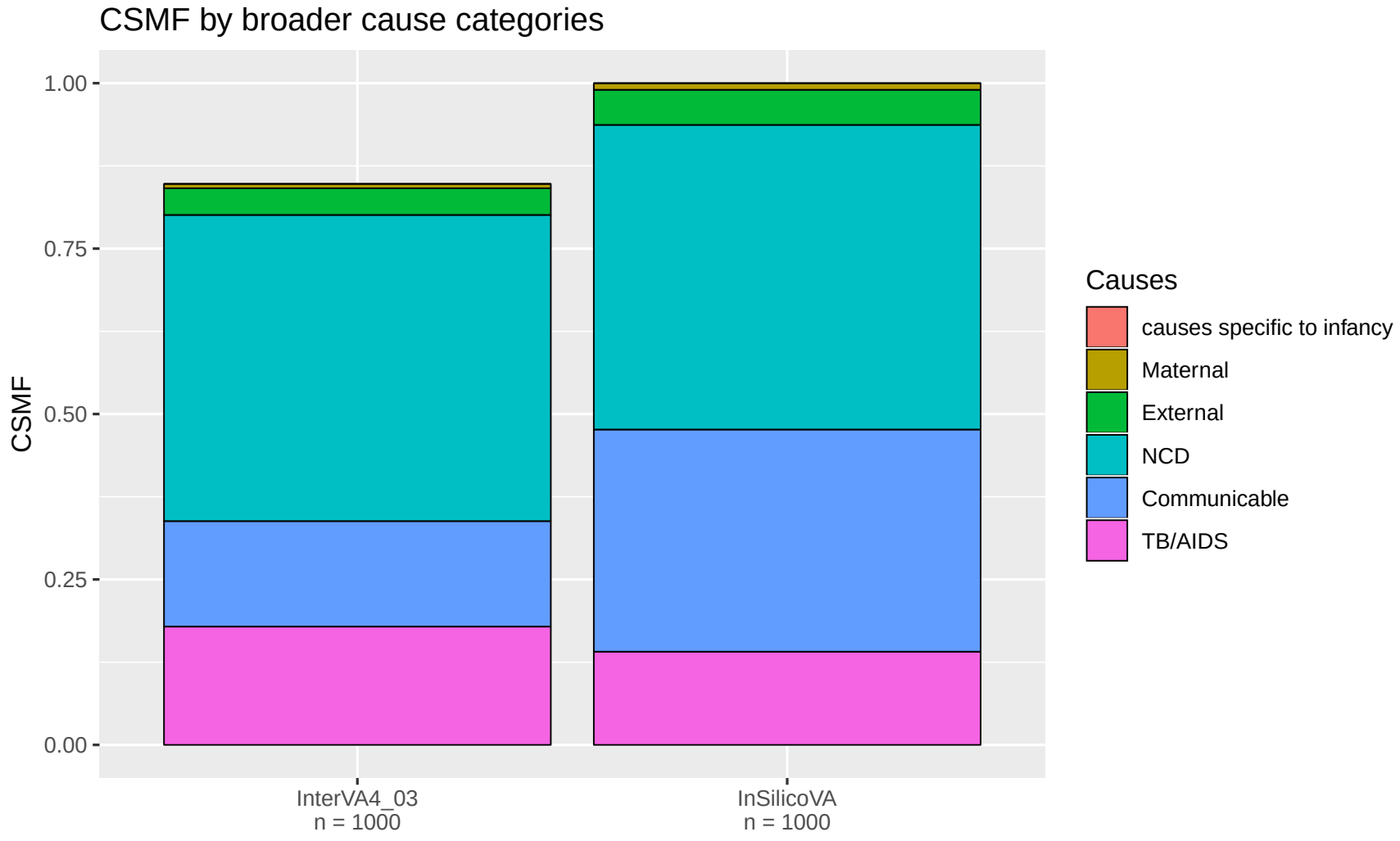
Recap: CSMF comparison

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



Recap: CSMF comparison

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



Next step

The CSMF plots look reasonable in this example dataset.

- Proportions for other broad cause categories are comparable

If we examine the CSMF plots more carefully, we can see

- InSilicoVA assigns more *other and unspecified infectious diseases*
- InSilicoVA assigns more *renal failure*
- InSilicoVA assigns fewer *stoke*
- ...

Can we get a sense of why this is the case?

Exploratory analysis of model output

Overview: what can we check?

Distribution of assigned causes.

- e.g., are any causes being assigned way too often?

Distribution of assigned causes by demographic groups.

- e.g., do any subpopulations get implausible cause assignments?

Relationship of assigned causes and symptoms.

- e.g., are there certain symptoms leading to certain causes to be assigned more/less often than expected?

Distribution of cause assignments by location/time?

- e.g., are there any systematic difference over location/time?

Are any of the differences surprising/unreasonable? Is there a data issue? Is there a model issue?

And even before these: check error and warning messages

Before any more analytical steps we will discuss, it is always a good idea to check any error logs from running the algorithm:

See how many cases are dropped out of the analysis and why do they drop out.

See if there are too many/few warnings about symptom check.

If you already identified some deaths with implausible cause assignments, see if something happens in data checking from the warnings log. For example, some key symptoms may not be flagged in the original input, but changed during data checking steps.

Plausibility check I: distribution of assigned causes

The easiest way to check the results is perhaps the `summary()` function.

```
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 Std.Error
## Other and unspecified infect dis 0.2652    0.0165
## HIV/AIDS related death           0.1018    0.0095
## Renal failure                    0.1015    0.0110
## Other and unspecified neoplasms  0.0615    0.0090
## Other and unspecified cardiac dis 0.0583   0.0076
## Digestive neoplasms              0.0496    0.0076
## Acute resp infect incl pneumonia 0.0481    0.0072
## Pulmonary tuberculosis           0.0391    0.0069
## Stroke                           0.0382    0.0061
```

Plausibility check I: distribution of assigned causes

```
top_ins <- getTopCOD(fit_ins)
table(top_ins[, 2])
```

```
##
##              Abortion-related death
##                                   1
##      Accid drowning and submersion
##                                   1
##                       Acute abdomen
##                                  27
##   Acute resp infect incl pneumonia
##                                  51
##                             Assault
##                                   9
##                              Asthma
##                                   9
##                     Breast neoplasms
##                                   1
##  Chronic obstructive pulmonary dis
##                                   2
##                    Diabetes mellitus
##                                   6
```

Tabulate assigned causes

```
counts <- table(top_ins[, 2])
sort(counts, decreasing = TRUE)
```

```
##
##   Other and unspecified infect dis
##                                267
##                       Renal failure
##                                112
##              HIV/AIDS related death
##                                101
##    Other and unspecified neoplasms
##                                 60
##  Other and unspecified cardiac dis
##                                 58
##                Digestive neoplasms
##                                 52
##   Acute resp infect incl pneumonia
##                                 51
##                             Stroke
##                                 40
##              Pulmonary tuberculosis
##                                 38
```

Algorithm-provided CSMF estimates

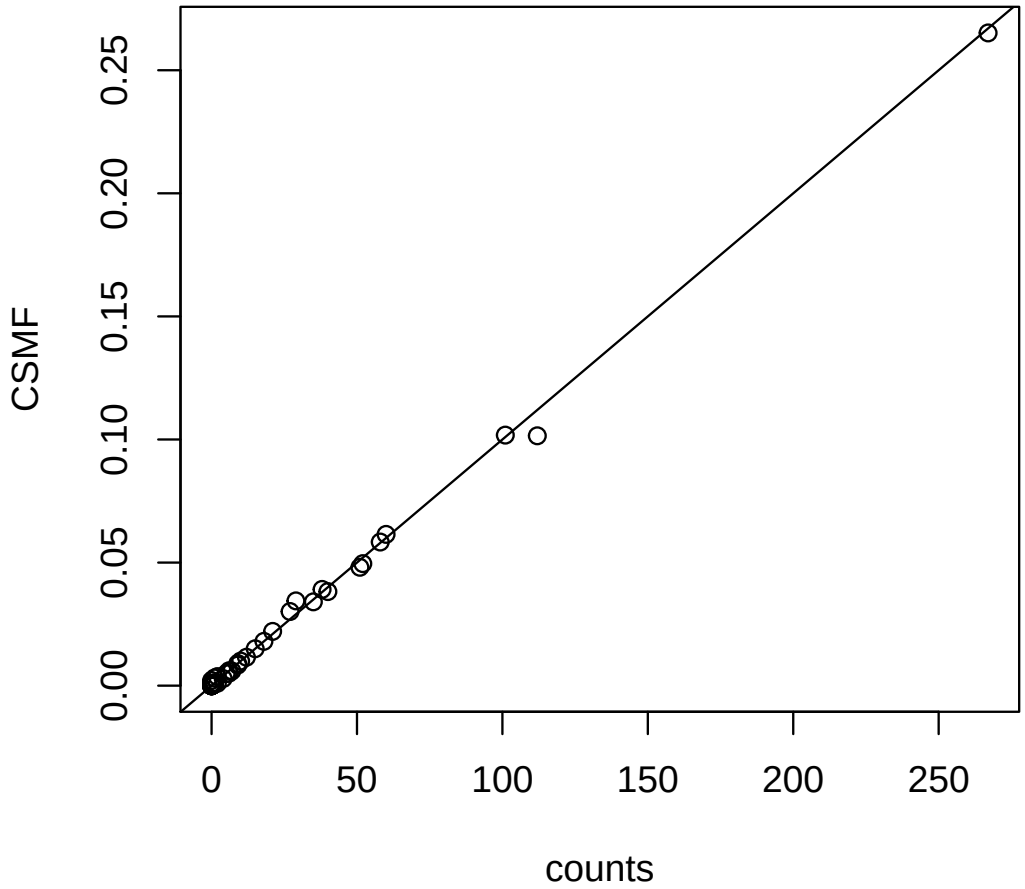
Sometimes, data manipulation can be tedious/frustrating, can you see what gos wrong with the code below?

```
tab_ins <- table(top_ins[, 2])
csmf_ins <- getCSMF(fit_ins)[, "Mean"]
plot(tab_ins, csmf_ins)
```

A better way to compare them. Notice the several tricks to make life easier.

```
csmf_ins <- getCSMF(fit_ins)[, "Mean"]
names <- names(csmf_ins)
tab_ins <- table(c(names, as.character(top_ins[, 2]))) -
    1
tab_ins <- tab_ins[names]
plot(as.numeric(tab_ins), csmf_ins, xlab = "counts", ylab = "CSMF")
abline(c(0, 1/sum(tab_ins)))
```

Compare CSMF estimates with the counts of top causes



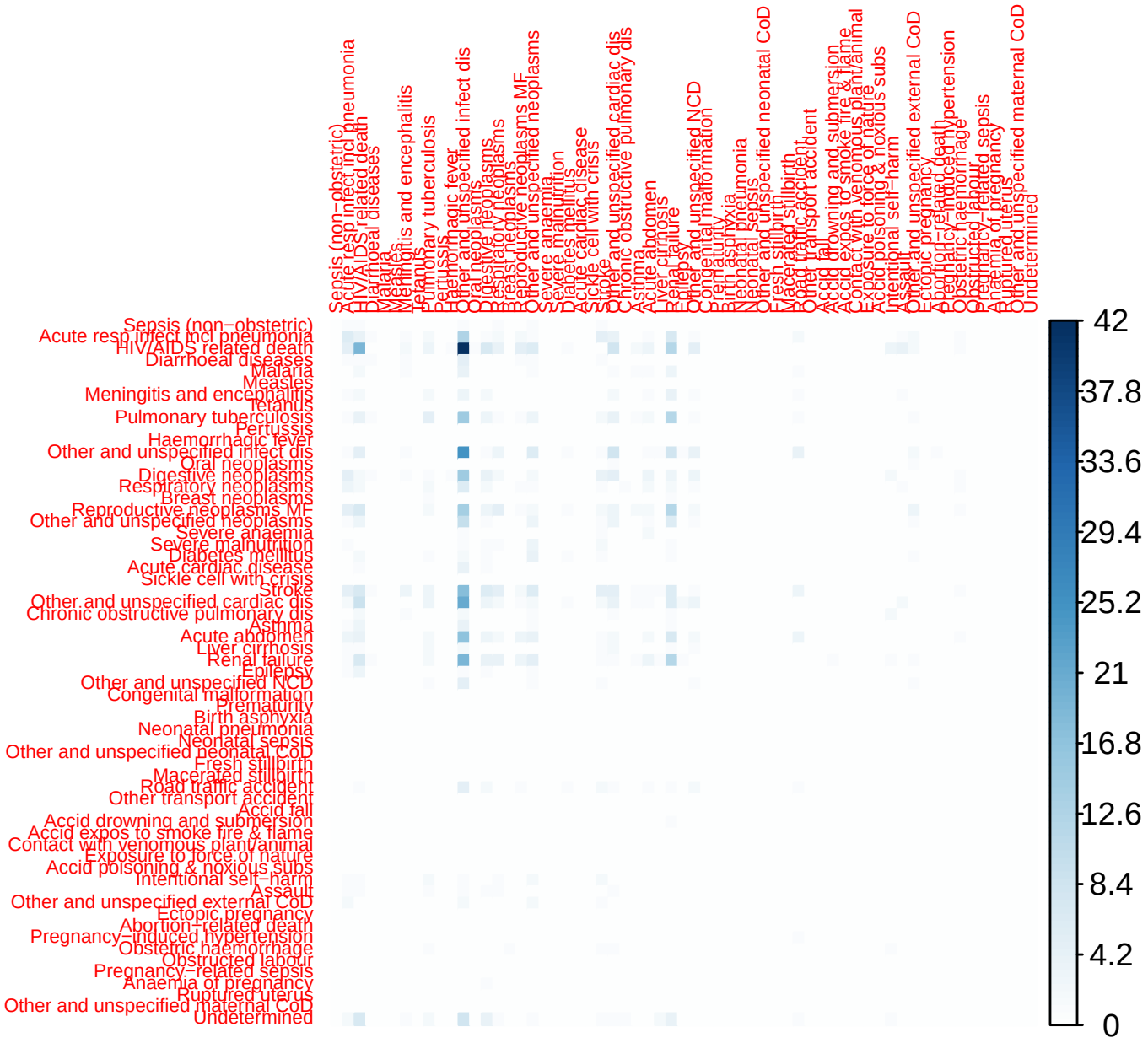
So it seems that the estimated CSMF and top cause assignment counts nicely align with each other.
When sample size is smaller and/or some CSMF has large uncertainty, the two may differ.
We can do the same thing with other algorithms too, as you have

Compare assigned causes from multiple algorithms

Now we can also compare the assigned causes from both algorithms in more detail.
We visualize the confusion matrix of assignments (Rows are the counts from InterVA, since the vector appears first in `table()` function).

```
# install.packages(corrplot)
library(corrplot)
top_inter <- getTopCOD(fit_inter)
names <- c(names, "Undetermined")
cod_inter <- factor(top_inter[, 2], levels = names)
cod_ins <- factor(top_ins[, 2], levels = names)
cross <- table(cod_inter, cod_ins)
corrplot(cross, method = "color", is.corr = FALSE, tl.cex = 0.5)
```

Visual comparison



About the confusion matrix

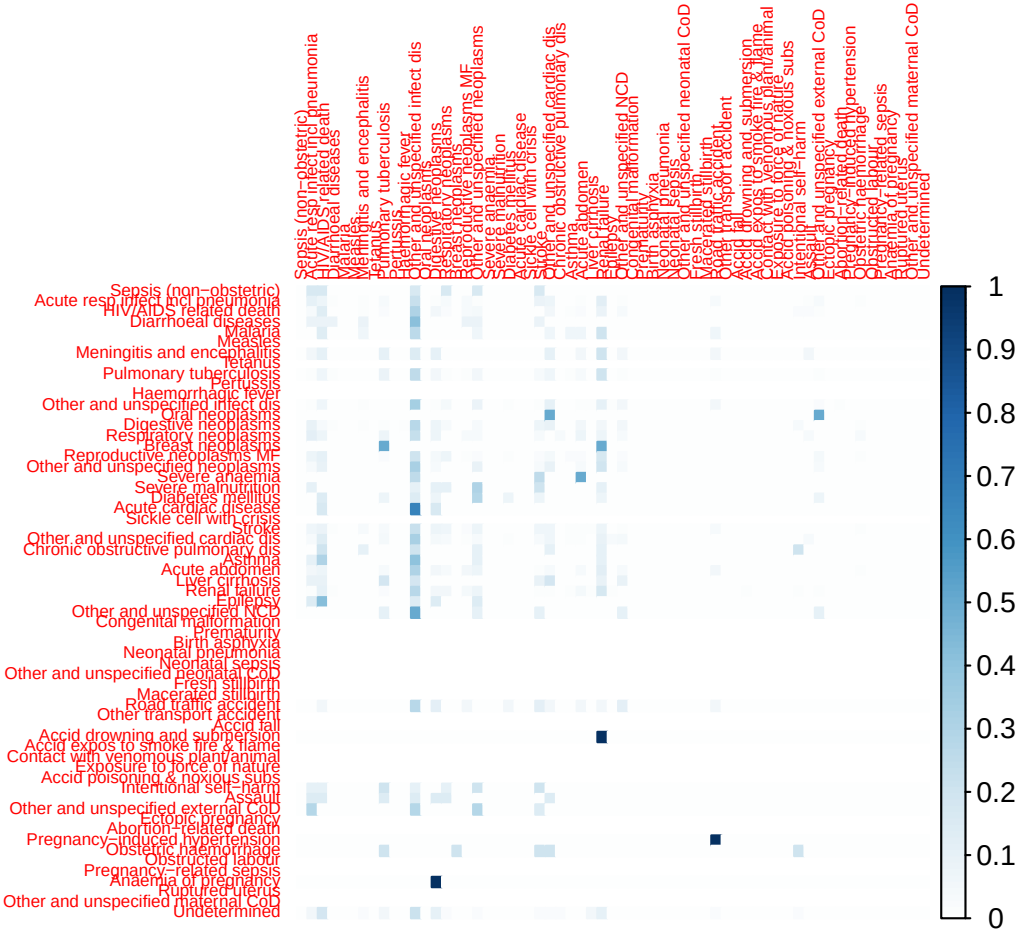
The confusion matrix shows that the agreement between the two algorithms are very low in terms of the most likely assignment.
Notice this can happen even if CSMF estimates are similar.
A few other observations from the comparison:

- Among the agreed assignment, HIV/AIDS, Unspecified infectious diseases are the two main causes.
- InSilicoVA assigns more 'Unspecified infectious diseases', while among those cases InterVA assigns many to HIV/AIDS, Pulmonary tuberculosis, Unspecified cardiovascular diseases, Stroke, ...
- ...

Sometimes it also makes sense to compare the 'normalized' version of the confusion matrix, i.e., what percentage of HIV/AIDS deaths identified by InterVA get assigned into unspecified infectious disease by InSilicoVA?
Especially helpful if you are comparing algorithm assigned causes to something more reliable, e.g., ground truth or consensus physician codings, etc.

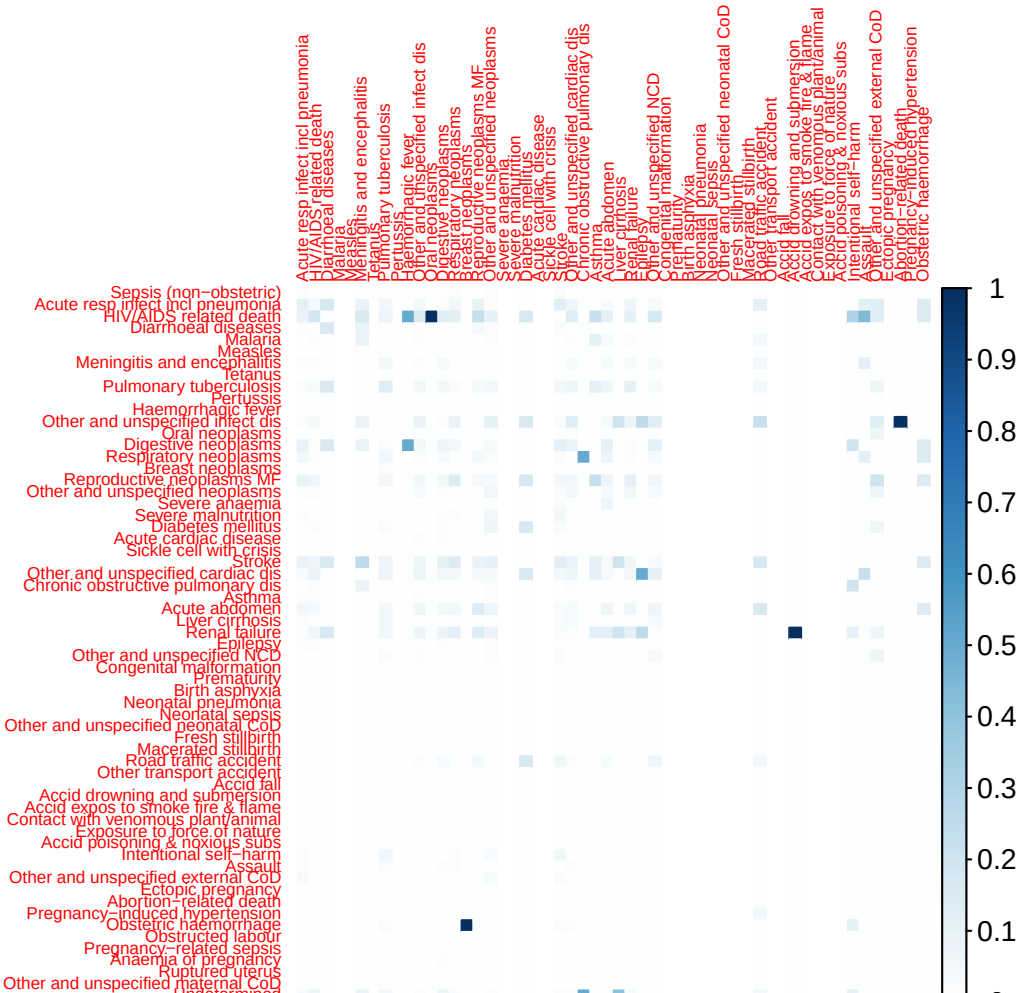
Normalize by the counts from InterVA

```
cross <- table(cod_inter, cod_ins)
for (i in 1:dim(cross)[1]) cross[i, ] <- cross[i, ]/sum(cross[i,
    ])
corrplot(cross, method = "color", is.corr = FALSE, tl.cex = 0.5,
    na.label = NA)
```



Normalize by the counts from InSilicoVA

```
cross <- table(cod_inter, cod_ins)
for (i in 1:dim(cross)[2]) cross[, i] <- cross[, i]/sum(cross[,
    i])
corrplot(cross, method = "color", is.corr = FALSE, tl.cex = 0.5,
    na.label = NA)
```



Plausibility check II: Distribution of assigned causes within demographic groups

In order to get a better sense of individual level results, we first find probability distribution of causes of death for each individual.
Notice this can also be obtained using NBC, but not Tariff. (*Why?*)

```
indiv_inter <- getIndivProb(fit_inter)[, -c(1:3)]
dim(indiv_inter)
```

```
## [1] 1000   60
```

```
indiv_ins <- getIndivProb(fit_ins)
dim(indiv_ins)
```

```
## [1] 1000   60
```

Merge output and input data

Now we can merge the data with the output using the common identifier.

```
combined_inter <- merge(RandomVA1, top_inter, by.x = "ID")
combined_inter <- merge(combined_inter, indiv_inter, by.x = "ID",
    by.y = "row.names")
combined_ins <- merge(RandomVA1, top_ins, by.x = "ID")
combined_ins <- merge(combined_ins, indiv_ins, by.x = "ID",
    by.y = "row.names")
```

Causes assigned for a subpopulation

For example, if we are interested in understanding the assigned causes of death for elder people,
we can subset the data to find where the symptom `elder` is yes.
Of course, in practice, you may have the raw data with more information on age, so that you can choose to define your subset using your own variable, after merging that variable into the data frame.

```
colnames(combined_inter)
sub <- which(toupper(combined_inter$elder) == "Y")
count_sub <- table(combined_inter[sub, "cause"])
sort(count_sub, decreasing = TRUE)
```

CSMF for a subpopulation

And as we have emphasized before, if you have a small dataset (or if you are investigating a small subset), looking at only the assigned causes may be misleading.

So let us also aggregate the individual probabilities and obtain something roughly as a sub-population version of CSMF.

```
# match(names, colnames(combined_inter))
prob_sub <- apply(combined_inter[sub, 248:307], 2, mean)
sort(prob_sub, decreasing = TRUE)
```

You may find repeating these steps for many different subpopulations can be tedious. . .
To make the exploration easier, sometimes it is worth building a small tool yourselves.

A slightly more systematic way of doing this

```
quicksummary <- function(data, symps, symps.value){
    for(i in 1:length(symps)){
        data <- data[toupper(data[, symps[i]]) ==
                     toupper(symps.value[i]), ]
        if(length(data) == 0) stop("No such combination!")
    }
    count <- table(data[, "cause"])
    count.sort <- sort(count, decreasing = TRUE)
    prob <- apply(data[, 248:307], 2, mean)
    prob.sort <- sort(prob, decreasing=TRUE)
    message(paste("Number of observations:", dim(data)[1]))
    message("Top 5 Assigned causes")
    tmp <- as.matrix(count.sort[1:5], ncol=1)
    colnames(tmp) <- "Freq"
    print(tmp)
    message("Top 5 sub-population CSMF")
    tmp <- as.matrix(prob.sort[1:5], ncol=1)
    colnames(tmp) <- "Prob"
    print(tmp)
    out <- list(count=count, count.sort=count.sort,
                prob=prob, prob.sort=prob.sort)
    return(out) }
```

Using the small function

```
out <- quicksummary(data = combined_inter, symps = "elder",
    symps.value = "y")
```

```
## Number of observations: 438

## Top 5 Assigned causes

##                                      Freq
## Stroke                                 63
## Other and unspecified cardiac dis      52
## Renal failure                          44
## Reproductive neoplasms MF              35
## Acute abdomen                          30

## Top 5 sub-population CSMF

##                                         Prob
## Stroke                              0.14401276
## Other and unspecified cardiac dis   0.11766053
## Renal failure                       0.09237247
## Reproductive neoplasms MF           0.07593892
## Other and unspecified infect dis    0.07133300
```

Filtering multiple variables

```
out <- quicksummary(data = combined_inter, symps = c("adult",
    "male"), symps.value = c("y", "y"))
```

```
## Number of observations: 214

## Top 5 Assigned causes

##                                    Freq
## HIV/AIDS related death               42
## Pulmonary tuberculosis               18
## Digestive neoplasms                  17
## Other and unspecified infect dis     17
## Renal failure                        17

## Top 5 sub-population CSMF

##                                        Prob
## HIV/AIDS related death              0.20368145
## Digestive neoplasms                 0.08481058
## Pulmonary tuberculosis              0.08330267
## Other and unspecified infect dis    0.07923790
## Acute abdomen                       0.06517372
```

Customization

Similarly you can check different combination of variables, and `combined_ins` too.

```
out <- quicksummary(data = combined_inter, symps = c("adult",
    "male"), symps.value = c("y", ""))
out <- quicksummary(data = combined_ins, symps = c("adult",
    "male"), symps.value = c("y", ""))
```

With a little bit more coding, you can build similar functions more useful for your own tasks.
And if you like your function, feel free to share with us!

Plausibility check III: symptoms and assigned causes

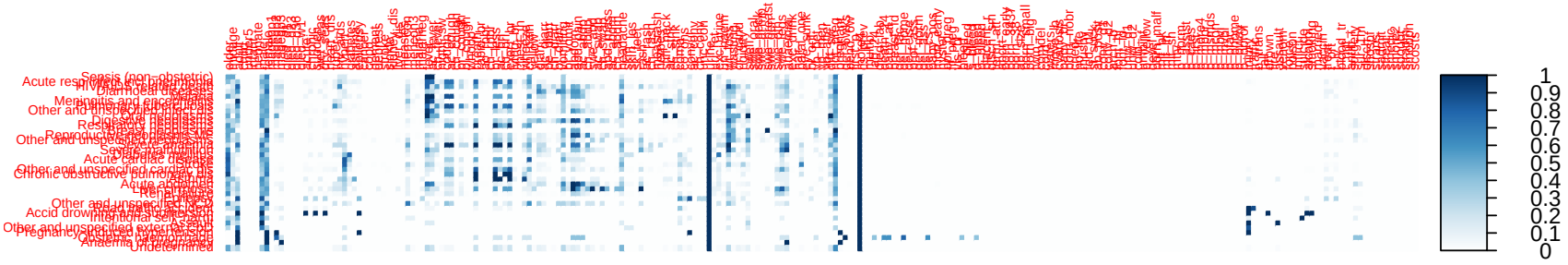
Besides subsetting your data, it is also usually helpful to calculate the conditional probability of observing a symptom given an assigned cause of death.

```
cond <- matrix(NA, 61, 245)
for (i in 1:61) {
    this.cause <- which(combined_inter$cause == names[i])
    nc <- length(this.cause)
    if (nc > 0) {
        for (j in 1:245) {
            cond[i, j] <- sum(combined_inter[this.cause,
                j + 1] == "Y")/nc
        }
    }
}
colnames(cond) <- colnames(combined_inter)[2:246]
rownames(cond) <- names
cond <- cond[!is.na(cond[, 1]), ]
```

Heatmap: symptoms and assigned causes

For easier visualization later, we have removed the causes not appeared in the assigned list from InterVA.
In practice, it is a good practice to first check those never-assigned causes and see if they are plausible.

```
corrplot(cond, method = "color", is.corr = FALSE, tl.cex = 0.5)
```

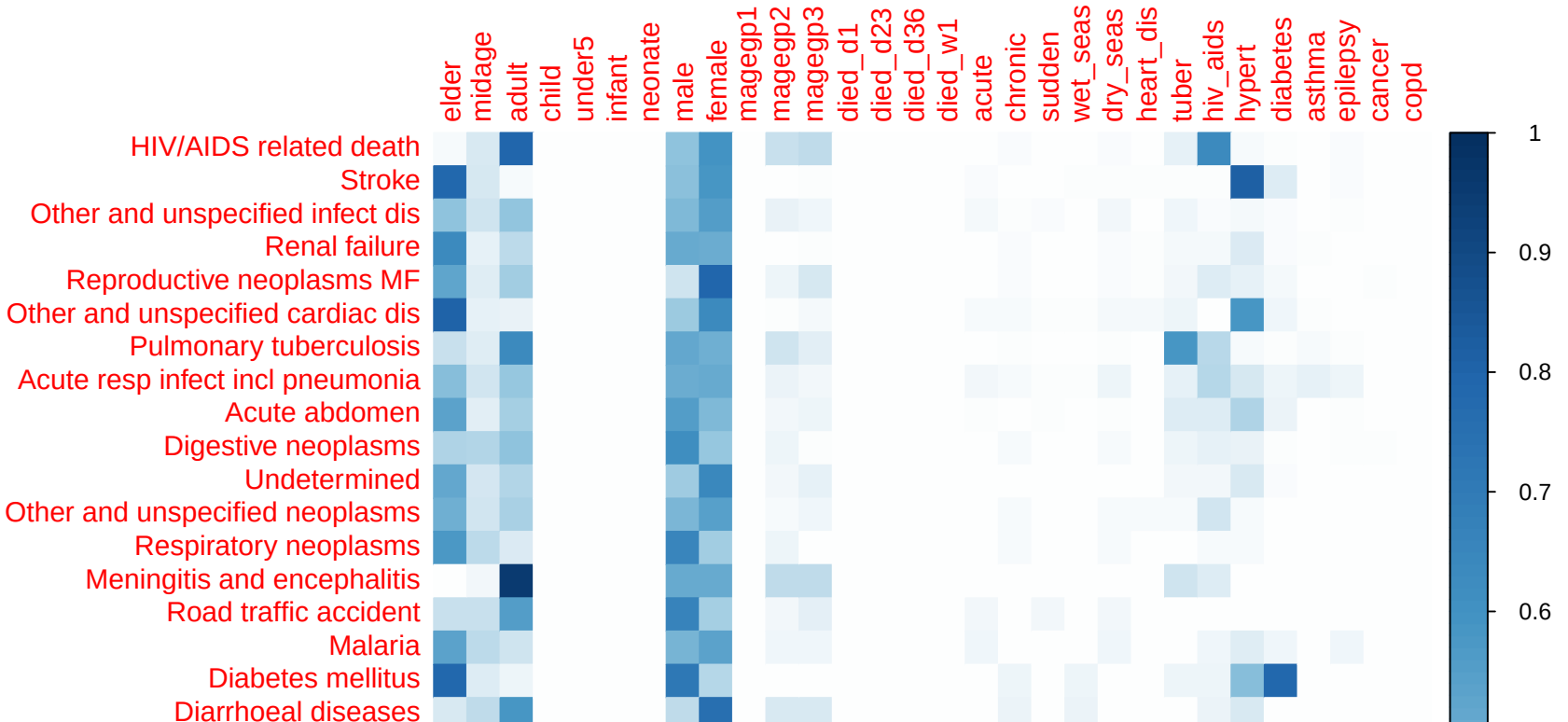


With a large number of symptoms and causes, this is very difficult to read off directly.
We may alternatively look at a subset of symptoms/causes

Heatmap: symptoms and assigned causes

We can also sort the order of causes by prevalence.

```
tab_inter <- table(c(names, as.character(top_inter[, 2]))) -
    1
tab_inter <- tab_inter[names]
tab_inter <- tab_inter[tab_inter > 0]
order <- order(tab_inter, decreasing = T)
corrplot(cond[order, 1:30], method = "color", is.corr = FALSE,
    tl.cex = 1)
```



Symptoms and assigned causes

Or we can order the conditional probabilities directly

```
library(reshape2)
condtab <- melt(cond)
condtab <- condtab[order(condtab[, 3], decreasing = T),
    ]
head(condtab)
```

```
##                                    Var1  Var2 value
## 101  Accid drowning and submersion adult     1
## 105 Pregnancy-induced hypertension adult     1
## 106          Obstetric haemorrhage adult     1
## 107           Anaemia of pregnancy adult     1
## 281  Accid drowning and submersion  male     1
## 283                        Assault  male     1
```

Symptoms and assigned causes

Or we can look at only such probabilities within larger causes

```
condtab <- merge(condtab, as.matrix(tab_inter), by.x = "Var1",
    by.y = "row.names")
condtab <- condtab[order(condtab[, 3], decreasing = T),
    ]
head(condtab[condtab$V1 > 100, ])
```

```
##                          Var1     Var2     value  V1
## 3205 HIV/AIDS related death    more4 1.0000000 136
## 3320 HIV/AIDS related death    urine 1.0000000 136
## 3401 HIV/AIDS related death  wt_loss 0.8308824 136
## 3282 HIV/AIDS related death    fever 0.8088235 136
## 3288 HIV/AIDS related death    adult 0.7941176 136
## 3419 HIV/AIDS related death hiv_aids 0.6397059 136
```

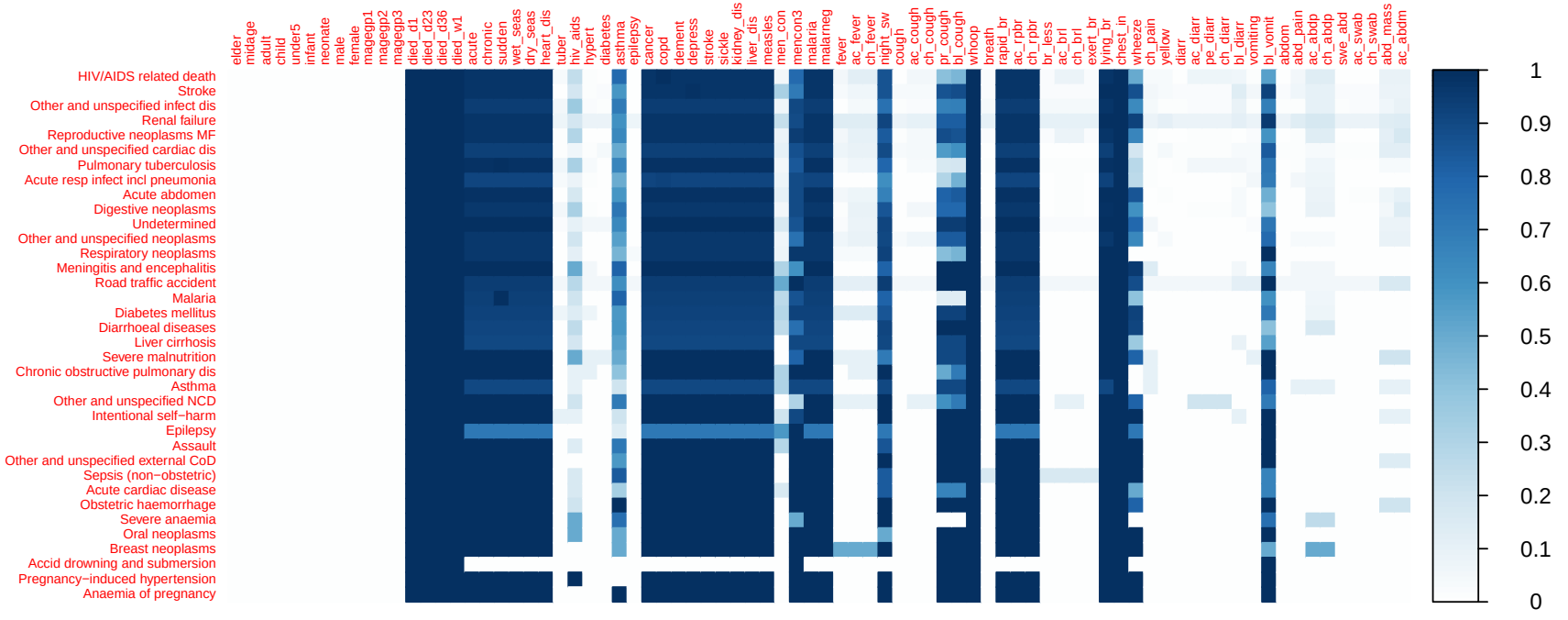
Patterns of missing data

Similarly, we can compare the patterns of missing data as well.

```
cond <- matrix(NA, 61, 245)
for (i in 1:61) {
    this.cause <- which(combined_inter$cause == names[i])
    nc <- length(this.cause)
    if (nc > 0) {
        for (j in 1:245) {
            cond[i, j] <- sum(combined_inter[this.cause,
                j + 1] == ".")/nc
        }
    }
}
colnames(cond) <- colnames(combined_inter)[2:246]
rownames(cond) <- names
cond <- cond[!is.na(cond[, 1]), ]
```

Patterns of missing data

```
corrplot(cond[order, 1:80], method = "color", is.corr = FALSE,
    tl.cex = 0.5)
```



More about tabulated data

Sometimes, you may also have many subgroups, and the high-level differences/relationships of these subgroups are of interest.
For example, let us take a look of a such summary table.

```
data <- read.csv("../data/cod-tabulate.csv")
head(data)
```

```
##    module                  cause34      age  sex counts
## 1 neonate          Birth asphyxia 0-7 days Male     29
## 2 neonate Congenital malformation 0-7 days Male     20
## 3 neonate       Meningitis/Sepsis 0-7 days Male     19
## 4 neonate                Pneumonia 0-7 days Male     21
## 5 neonate         Preterm Delivery 0-7 days Male     19
## 6 neonate               Stillbirth 0-7 days Male      3
```

```
tail(data)
```

```
##     module       cause34 age    sex counts
## 468  adult     Pneumonia 80+ Female     18
## 469  adult Renal Failure 80+ Female     16
## 470  adult  Road Traffic 80+ Female      3
## 471  adult        Stroke 80+ Female    207
## 472  adult            TB 80+ Female      6
```

Further aggregation

```
data.sex <- aggregate(counts ~ cause34 + sex, data = data,
    FUN = sum)
head(data.sex)
```

```
##                    cause34    sex counts
## 1           Birth asphyxia Female     14
## 2 Bite of Venomous Animal Female      3
## 3            Breast Cancer Female     66
## 4                  Cancers Female     11
## 5 Cardiovascular Diseases Female     17
## 6          Cervical Cancer Female     70
```

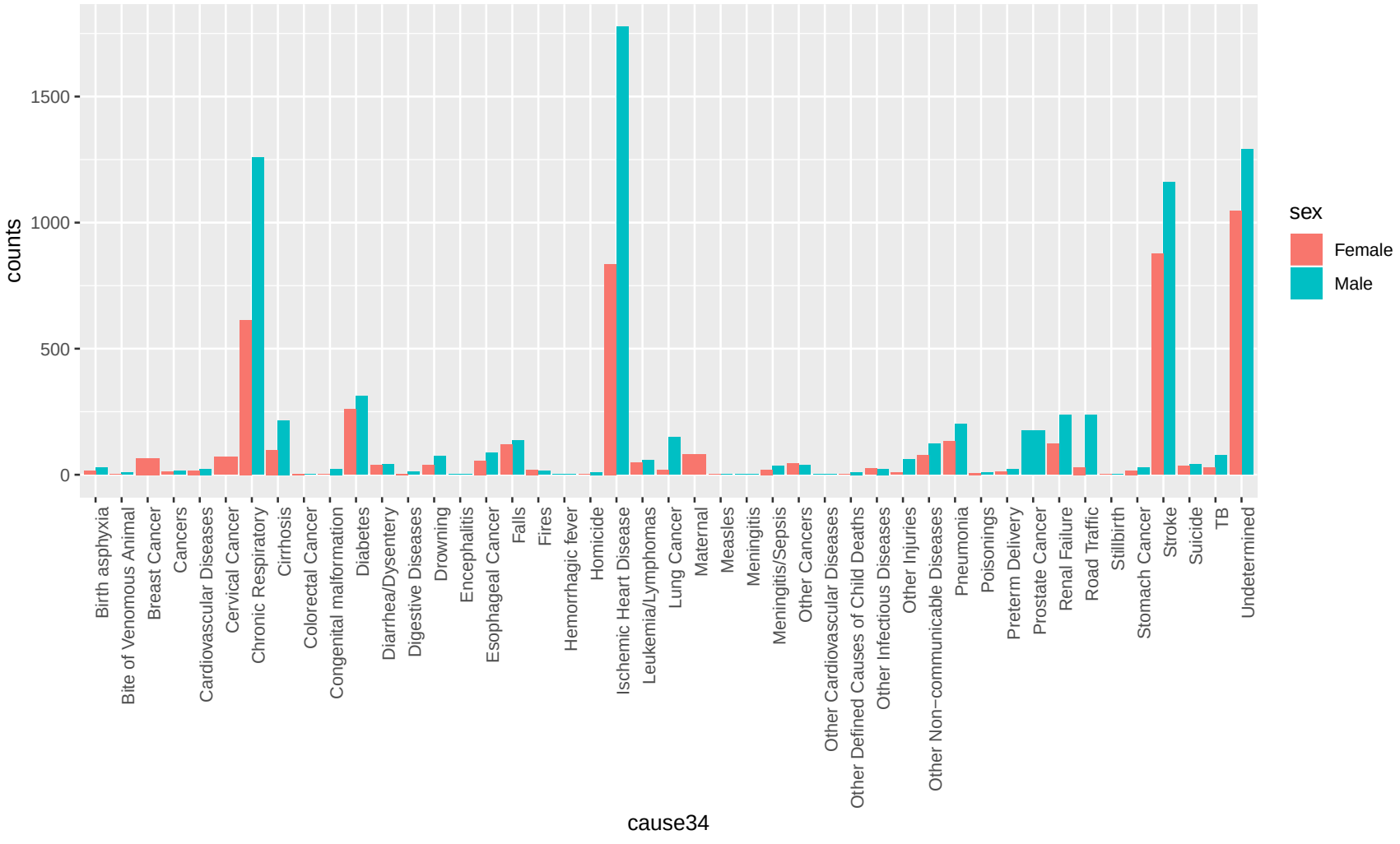
Visualization tabulated data

There are other ways to do this, but the codes are much more tedious.
In the next few slides, I use the visualization package called 'ggplot2'.

- It lets you write much less codes to make flexible plots
- BUT, the syntax is also more obscure, so steeper learning curves!
- *Writing fancy visualization codes is not out main goal here*, so feel free to explore tables in your own ways.

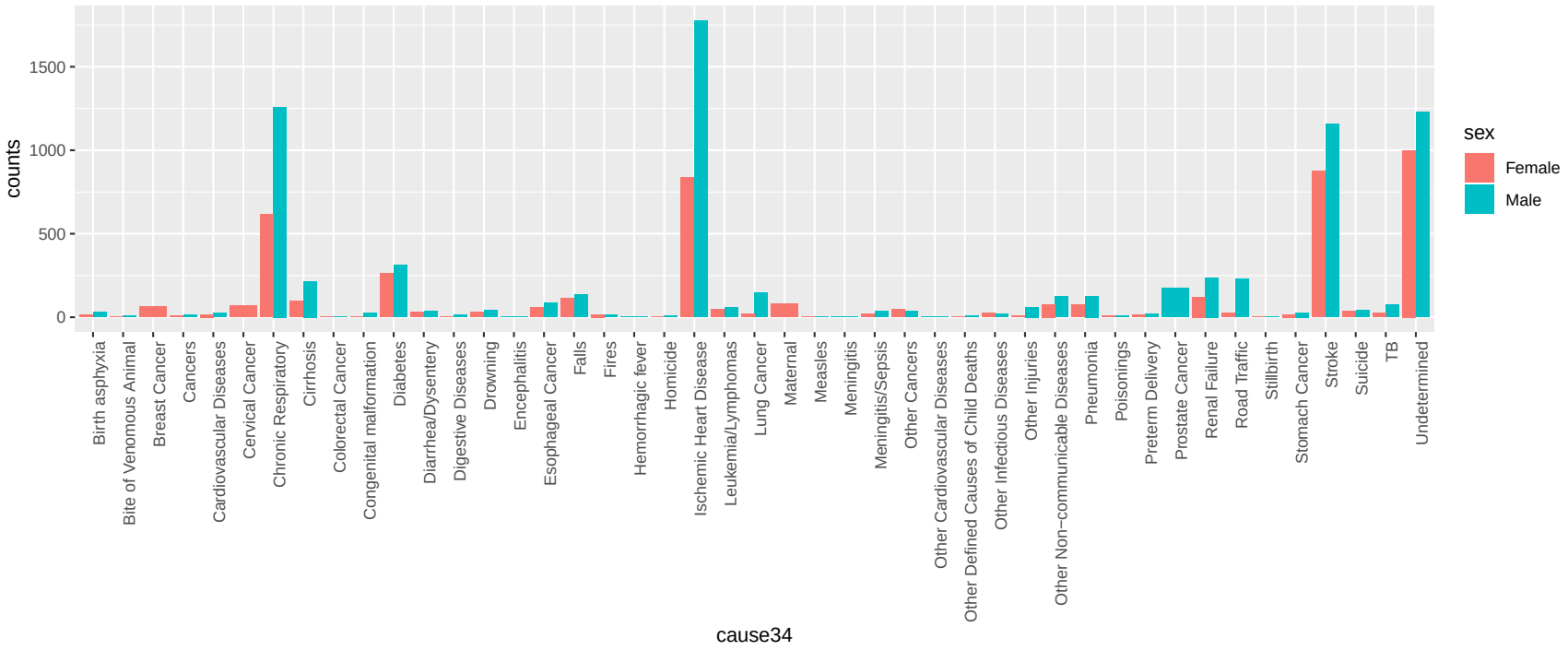
Barplot of counts by sex

```
ggplot(data.sex, aes(x=cause34, y=counts, fill=sex)) + geom_bar(stat="i
```



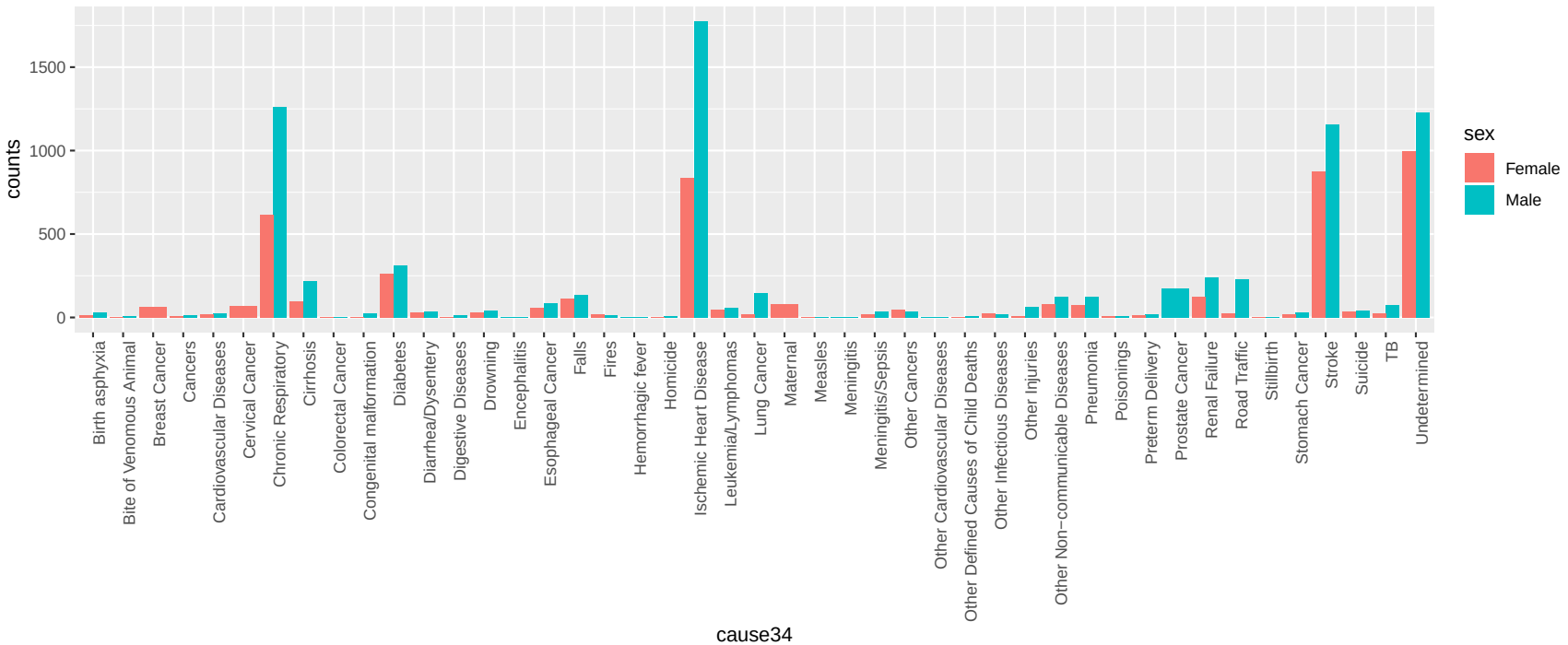
Barplot of counts by sex: adult

```
data.sex.module <- aggregate(counts~cause34+sex+module,
    data = data, FUN = sum)
ggplot(subset(data.sex.module, module="adult"),
    aes(x=cause34, y=counts, fill=sex)) +
    geom_bar(stat="identity", position=position_dodge()) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



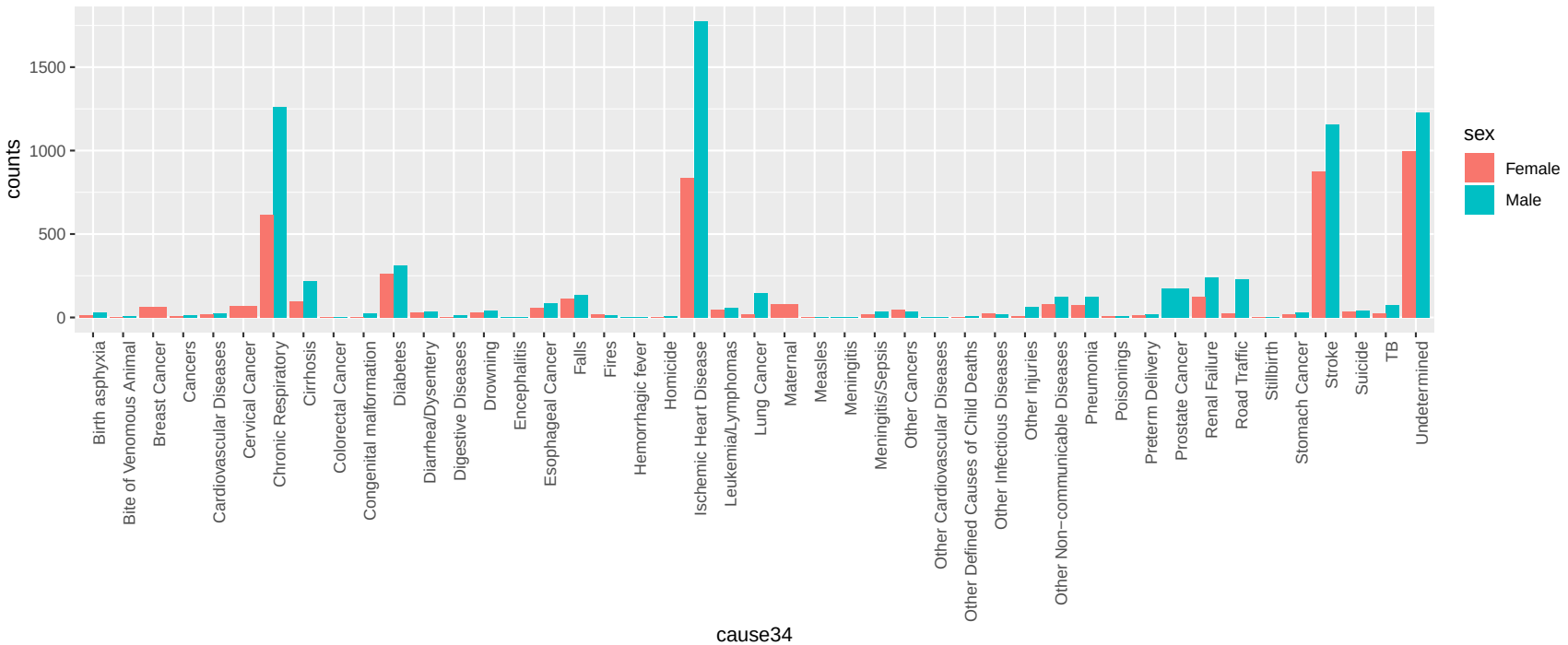
Barplot of counts by sex: child

```
ggplot(subset(data.sex.module, module="child"),
    aes(x=cause34, y=counts, fill=sex))+
    geom_bar(stat="identity", position=position_dodge())+
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



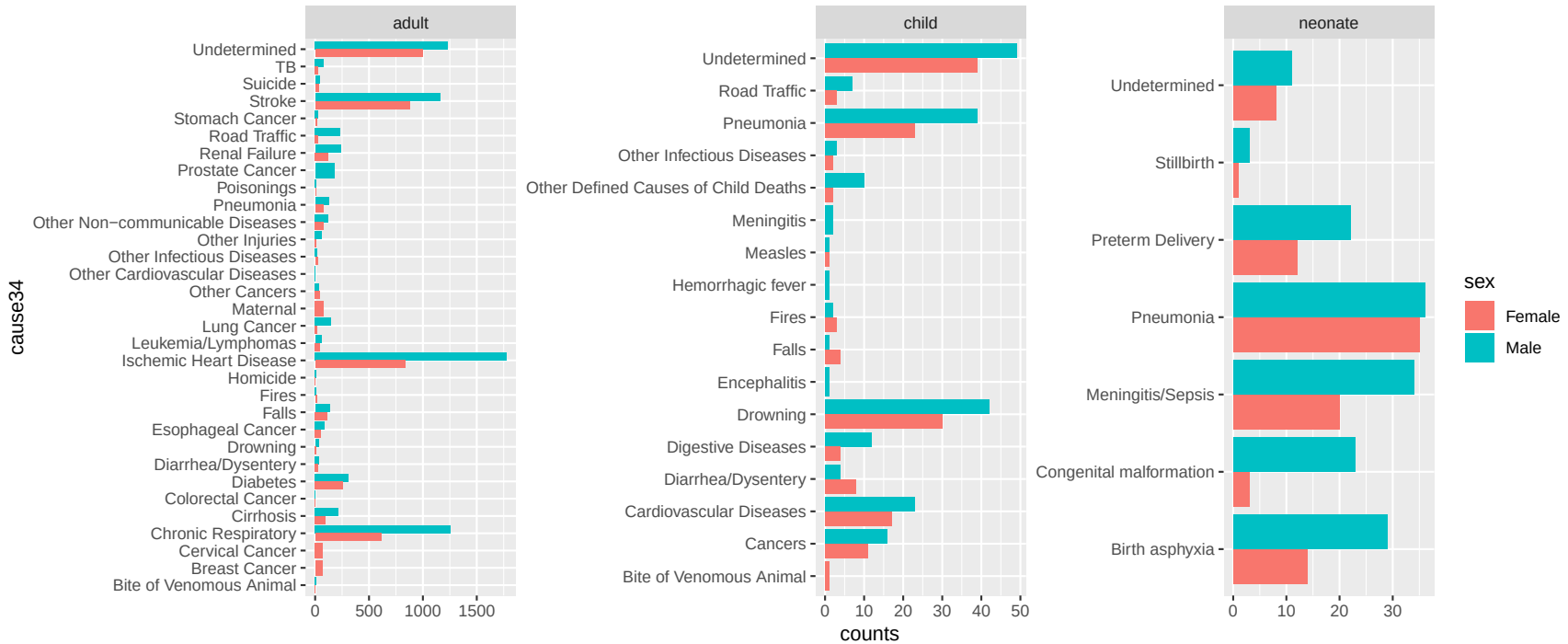
Barplot of counts by sex: neonate

```
ggplot(subset(data.sex.module, module="neonate"),
    aes(x=cause34, y=counts, fill=sex))+
    geom_bar(stat="identity", position=position_dodge())+
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



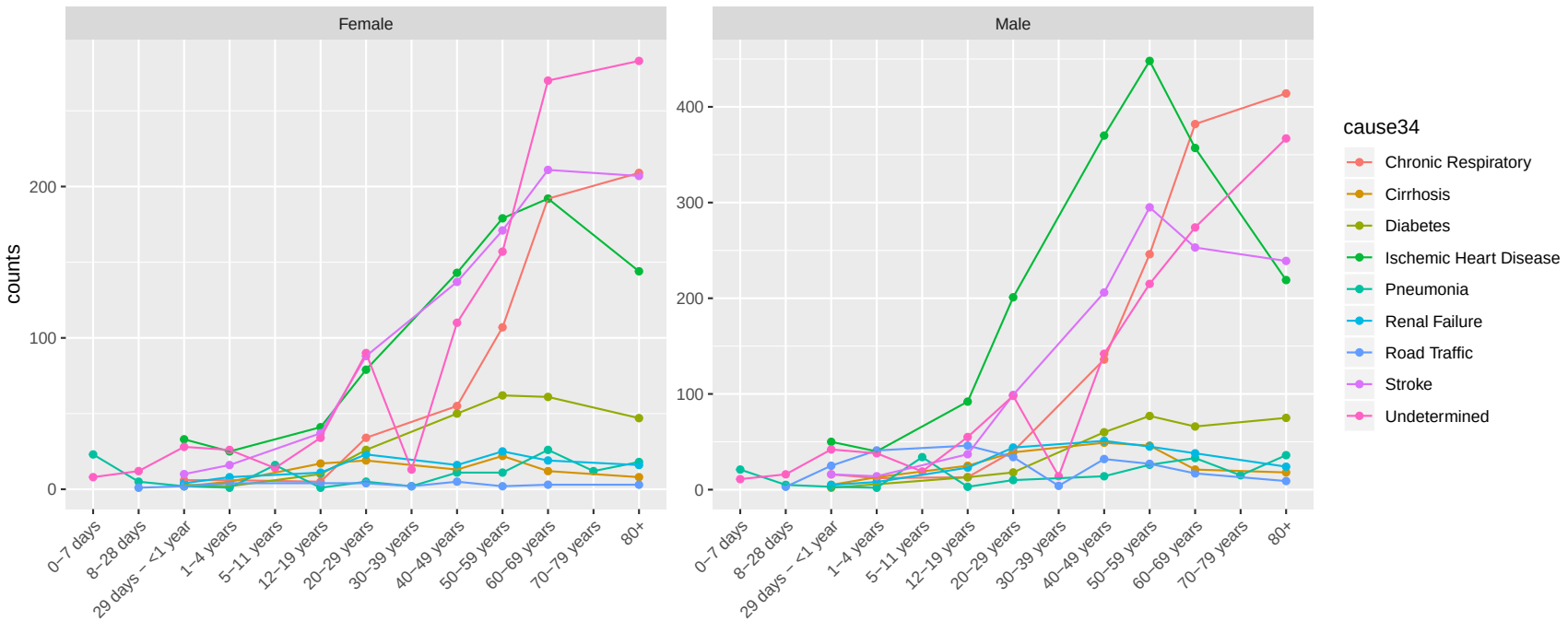
Barplot of counts by sex and module

```
ggplot(data.sex.module,
     aes(x=cause34, y=counts, fill=sex))+
    geom_bar(stat="identity", position=position_dodge())+
    facet_wrap(~module, scale = "free") + coord_flip()
```



Visualize trends

```
agegrps <- c("0-7 days", "8-28 days", "29 days - <1 year",
    "1-4 years", "5-11 years", "12-19 years", "20-29 years",
    "30-39 years", "40-49 years", "50-59 years",
    "60-69 years", "70-79 years", "80+")
levels(data$age) <- agegrps
topcauses <- data.sex$cause34[which(data.sex$counts > 200)]
ggplot(subset(data, cause34 %in% topcauses),
    aes(x=age, y=counts, color= cause34, group=cause34))+
    geom_point() + geom_line() + facet_wrap(~sex, scale = "free")+
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Spatial/temporal pattern of causes

When you have data collected from different locations and time period, it is usually useful to also explore the output across space/time.

You can perform all the checks for each location/period separately first.

Then putting them together may also reveal useful information

- Are there any trends in time/space?
- Does the observed trend make sense in practice? Are there any places/periods that seem strange?
- Have the algorithms been carried out properly for them?
- Are those data processed different from the rest, or contaminated?
- ...

See exercise for an example

Wrap up

It is important to check model fitting results

There are a few things usually helpful for sanity checks,

- Distribution of causes, overall and by subpopulations.
- Relationship between causes and symptoms, and missing variables.
- Trends in time/space
- ...

This is not meant to be an exhaustive list! You may discover other things in practice.

It is also not easy to find answers why certain causes have high/low prevalence, but doing such exercises may help identify problems in algorithms, model fitting, data processing and collection.

Finally, you may usually have more data besides input for `openVA`. So think how you can use them too!

Exercise

Analyzing multiple datasets

1. In the data folder, there are three datasets *va16Data1_hw.csv*, *va16Data2_hw.csv*, and *va16Data3_hw.csv*. Suppose they each represent the VA data collected in the same area during three consecutive years.
2. Fitting both InSilicoVA and InterVA on the three datasets separately, obtain CSMF distribution and cause of death counts for each year and each algorithm.
3. Check if the results make sense to you. Imagine if you are the analyst responsible for processing these dataset, is there anything strange going on?
4. Make some visual comparison of time trends over the three years. You can decide your own quantity of interest to show.

Exploring SCI

1. Identify up to three major causes of death assigned by each algorithm in these datasets.
2. Identify the symptoms with the highest associations, both positive and negative, with the deaths assigned to these causes. *(Hint: In order to do this, you can start with identifying the deaths and tabulate their symptoms.)*
3. Can you visually and verbally explain why these causes have higher prevalence in these datasets to your colleagues? If you identified any unreasonable phenomenon going on in the previous exercise, can you make some suggestions how to deal with them?