CalibratedVA

Jacob Fiksel and Abhirup Datta

2019-02-25

Introduction

Many countries cannot conduct full autopsies for the majority of deaths due to either cultural or economic limitations (AbouZahr et al. 2015; Allotey et al. 2015). Rather than perform an invasive autopsy, an alternative method to determine the cause (or "etiology") of death (COD) is to conduct a *verbal autopsy* (VA), where relatives of the deceased individual are asked a set list of questions to obtain information about symptoms experienced by the deceased observed prior to death (Soleman, Chandramohan, and Shibuya 2006). Medical experts can then use these VA responses to determine the COD for an individual.

If we are interested in obtaining causes of death for a large amount of individuals in a region or country, in order to determine optimal resource allocation, we need an efficient way to scale the process of determining the COD from VA data. Several algorithms have recently been proposed to automatically predict COD from VA records. These *computer coded* VA (CCVA) algorithms predict COD using the VA records as input. Examples include Tariff (James, Flaxman, and Murray 2011; Serina et al. 2015), interVA (Byass et al. 2012), and InsilicoVA (McCormick et al. 2016). These algorithms all require some gold standard (GS) information – either a training dataset with both VA symptom data and a gold standard (GS) COD determined for each individual or some symptom-given-cause probability matrix.

If we are interested in obtaining national estimates of cause specific mortality fraction (CSMF), we cantrain one of the above CCVA algorithms with GS COD data from our country of interest, obtain individual predictions from CCVA data for the country of interest, and then aggregate the predictions to the CSMF level. However, there may be limited amounts of GS COD data for a new country. In addition, algorithms trained on GS COD from other countries may not be as accurate.

The CalibratedVA package is to be used for exactly the situation above– we take predictions from one or multiple algorithm(s) trained on a large non-local dataset, and use the local GS COD dataset to improve the CSMF estimates via local calibration.

PHMRC Data

The Population Health Metrics Research Consortium (PHMRC) study contains GS COD data for children, neonates and adults in 4 countries, making it a perfect dataset to demonstrate the merits of `CalibratedVA`. In this vignette, we will use the `openVA` package to obtain PHMRC data and implement the Tariff and InSilicoVA algorithms. We will of course also have to load the `CalibratedVA` package.

```r
library(openVA)
library(CalibratedVA)
```

We will now load in the PHMRC child data and use the `ConvertData.phmrc` function to convert the data into the appropriate structure for Tariff and InSilicoVA.

```r
child.raw <- read.csv(getPHMRC_url("child"))
child.clean <- ConvertData.phmrc(child.raw, phmrc.type = "child")$output
#> The first column is site, assign IDs to each row by default
#> 2064 deaths in input. Format: adult
#> 109 binary symptoms generated
#>
#> Number of Yes        37172
#> Number of No         172134
#> Number of Not known  15670
```

To demonstrate how to implement CalibratedVA, we will first treat Tanzania as our country of interest for obtaining accurate CSMF estimates. We will split the PHMRC data from Tanzania into a hospital (calibration) and population (test) set, and use the rest of the PHMRC data as our gold standard (training) set. We will use a calibration set size of 100.

```
countries <- ifelse(child.raw$site %in% c("Dar", "Pemba"), "Tanzania", "Other")
tanzania.data <- child.clean[countries == "Tanzania",]
train.data <- child.clean[countries == "Other",]
set.seed(851745)
calibration.indices <- sample(nrow(tanzania.data), 100, replace = F)
calibration.data <- tanzania.data[calibration.indices,]
test.data <- tanzania.data[-calibration.indices,]
```

Implementing Tariff and InSilicoVA

In this section, we will use our training data to build prediction models using the Tariff and InSilicoVA algorithms. First we will implement Tariff. Note that the `data` argument is the data for which we want to obtain COD predictions for, which is both the calibration data and our test data (we will use the calibration set for CalibratedVA later in the vignette). When evaluating the performance of these algorithms, we will only do so on the predictions for the test data.

```
set.seed(123)
tariff.train <- codeVA(data = rbind(calibration.data, test.data),
                       data.type = "customize", model = "Tariff",
                       data.train = train.data, causes.train = "Cause")
#>
#> Start re-sampling for significant Tariff cells
#> Calculating ranks
```

We will now implement InSilicoVA.

```
set.seed(123)
insilico.train <- codeVA(data = rbind(calibration.data, test.data),
                         data.type = "customize", model = "InSilicoVA",
                         data.train = train.data, causes.train = "Cause",
                         jump.scale = 0.05, Nsim=5000, auto.length = FALSE)
#> Warning in extract.prob(train = train, gs = cause, gstable =
#> causes.table, : 2 symptoms deleted for missing rate over the pre-specified
#> threshold 0.95
#> Warning in insilico.train(data = data, data.type = "customize", data.train
#> = train.data, : 2 symptoms deleted from testing data to match training
#> data: c1_02, c1_04
```

Implementing CalibratedVA with Individual Algorithms

For simplicity, we will restrict ourselves to predicting the distribution for the top 3 COD (from Tanzania), and treat all other COD as "other". Because "Other Defined Causes of Child Deaths" is coded as cause 14, we will not include this cause in the Top 4.

We will extract predictions for the test and calibration set from both algorithms implemented above, and then change any prediction that is not in the top 3 COD to other.

```r
top.cod <- names(sort(table(tanzania.data$Cause), decreasing = TRUE))
top3.cod <- top.cod[top.cod != "14"][1:3]
change.cause <- function(cause) {
    cause <- as.character(cause)
    cause[!(cause %in% top3.cod)] <- "99"
    return(cause)
}
tariff.train.cod <- change.cause(getTopCOD(tariff.train)[,2])
insilico.train.cod <- change.cause(getTopCOD(insilico.train)[,2])
test.changedcod <- change.cause(test.data$Cause)
calibration.changedcod <- change.cause(calibration.data$Cause)
```

Finally, we will separate out the InSilicoVA and Tariff predictions for the test and calibration sets.

```r
tariff.train.cod.test <- tariff.train.cod[-(1:100)]
tariff.train.cod.calib <- tariff.train.cod[1:100]
insilico.train.cod.test <- insilico.train.cod[-(1:100)]
insilico.train.cod.calib <- insilico.train.cod[1:100]
```

We will now initiate hyper-parameter values for CalibratedVA. These can be changed of course, but we have found these values work well. We will run 3 chains, obtaining 50,000 draws for each chains. Note that the order of the causes in the `causes` vector, as this is the order in which CalibratedVA will present the distribution of COD estimates.

```r
causes <- as.character(sort(unique(test.changedcod)))
epsilon <- .001
alpha <- 5
beta <- .5
tau <- .5
tau.vec <- rep(tau, length(causes))
delta <- 1
gamma.init <- 1
ndraws <- 50E3
nchains <- 3
```

We will first run CalibratedVA using the Tariff predictions

```r
set.seed(123)
calibva.seeds <- sample(1e6, nchains, replace = F)
tariff.calibva <- calibva.sampler(test.cod = tariff.train.cod.test,
                                  calib.cod = tariff.train.cod.calib,
                                  calib.truth = calibration.changedcod, causes = causes,
                                  epsilon = epsilon, alpha=alpha, beta=beta,
                                  tau.vec=tau.vec, delta=delta,
                                  gamma.init=gamma.init, ndraws = ndraws,
                                  nchains = nchains,
                                  init.seeds = calibva.seeds)
```

Before thinning, we may want to see what the acceptance rates are for our MH draws of gamma. Acceptance rates that are too high means we will want to choose a higher value of tau, and acceptance rates that are too low means we should choose a smaller value of tau.

```r
gamma_acceptance_rates(tariff.calibva)
#> # A tibble: 12 x 3
#> # Groups:   Chain [?]
#>    Chain Parameter  rate
#>    <int> <fct>     <dbl>
#>  1     1 gamma[1]  0.616
#>  2     1 gamma[2]  0.609
#>  3     1 gamma[3]  0.610
#>  4     1 gamma[4]  0.611
#>  5     2 gamma[1]  0.613
#>  6     2 gamma[2]  0.612
#>  7     2 gamma[3]  0.610
#>  8     2 gamma[4]  0.612
#>  9     3 gamma[1]  0.614
#> 10     3 gamma[2]  0.613
#> 11     3 gamma[3]  0.609
#> 12     3 gamma[4]  0.612
```

We will then use a burn-in of 10,000 for each chain and also thin the chains by a factor of 10

```
tariff.calibva <- window(tariff.calibva, start = 10e3, thin = 10)
```

And now using the InSilicoVA predictions.

```
insilico.calibva <- calibva.sampler(test.cod = insilico.train.cod.test,
                                    calib.cod = insilico.train.cod.calib,
                                    calib.truth = calibration.changedcod, causes = causes,
                                    epsilon = epsilon, alpha=alpha, beta=beta,
                                    tau.vec=tau.vec, delta=delta,
                                    gamma.init=gamma.init, ndraws = ndraws,
                                    nchains = nchains,
                                    init.seeds = calibva.seeds)
insilico.calibva <- window(insilico.calibva, start = 10E3, thin = 10)
```

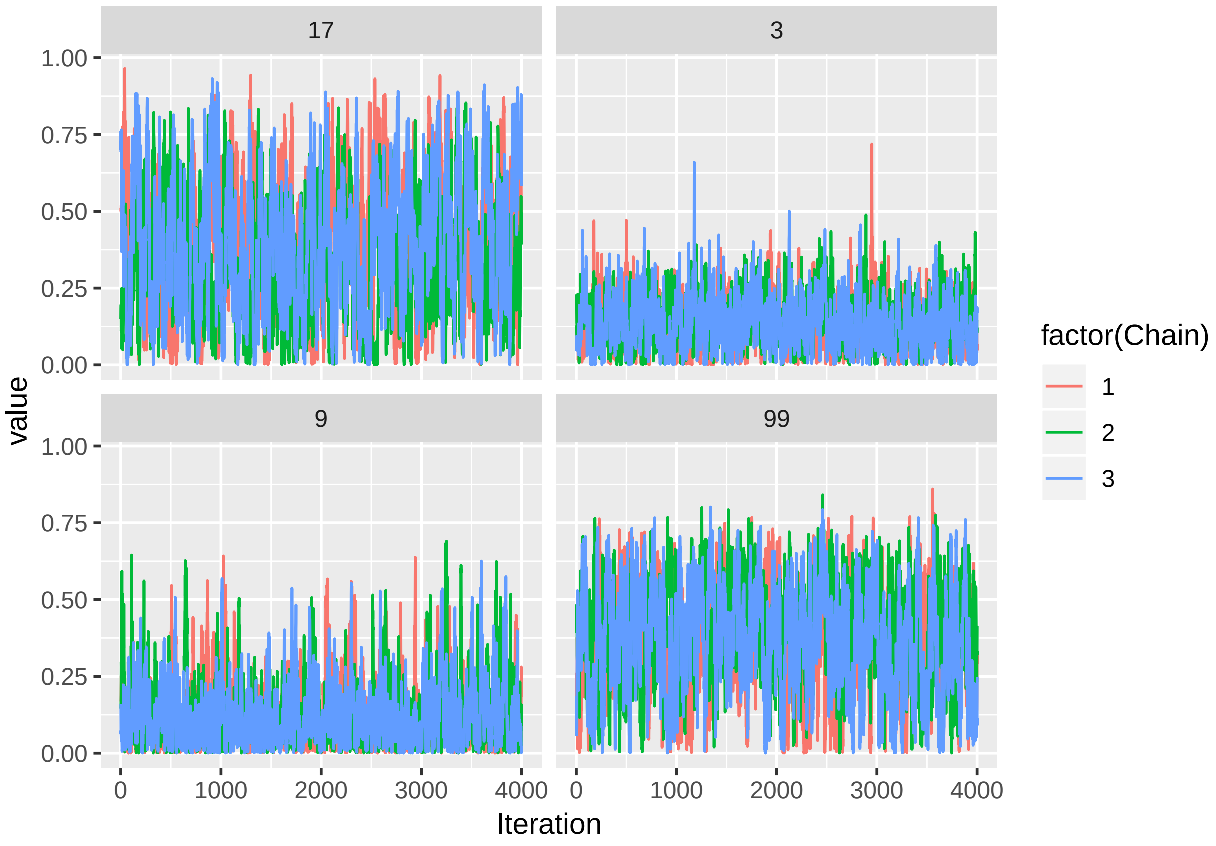
Obtaining CalibratedVA output

The `calibva.sampler` output is a `mcmc.list` object of length `nchains`. Each element in this list is a `mcmc` object, which contains a `ndraw` x `nparam` matrix, each each row representing the posterior draw. We are most interested in the posterior draws for the CSMF parameters, which we can extract using `calibvaCSMFPosteriorSamples` function. This function utilizes the `ggs` function from the `ggmcmc` package. We will demonstrate how to obtain CalibratedVA output using the output from Tariff.

```
library(tidyverse)
library(ggmcmc)
tariff.calibva.csmf.samples <- calibvaCSMFPosteriorSamples(tariff.calibva, causes = causes)
```

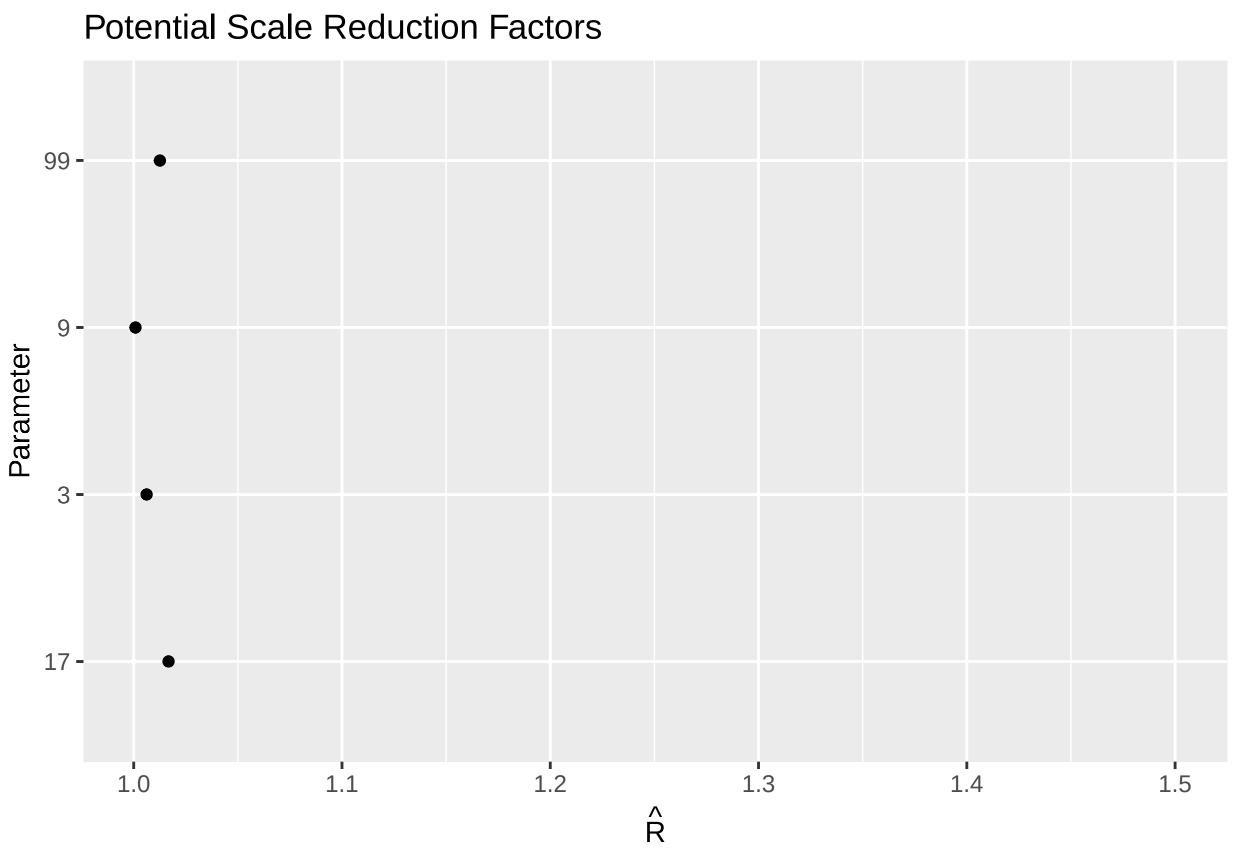
We can easily make a trace plot for each of the parameters:

```
ggplot(tariff.calibva.csmf.samples, aes(x = Iteration, y = value, color = factor(Chain))) +
  geom_line() +
  facet_wrap(~cause)
```



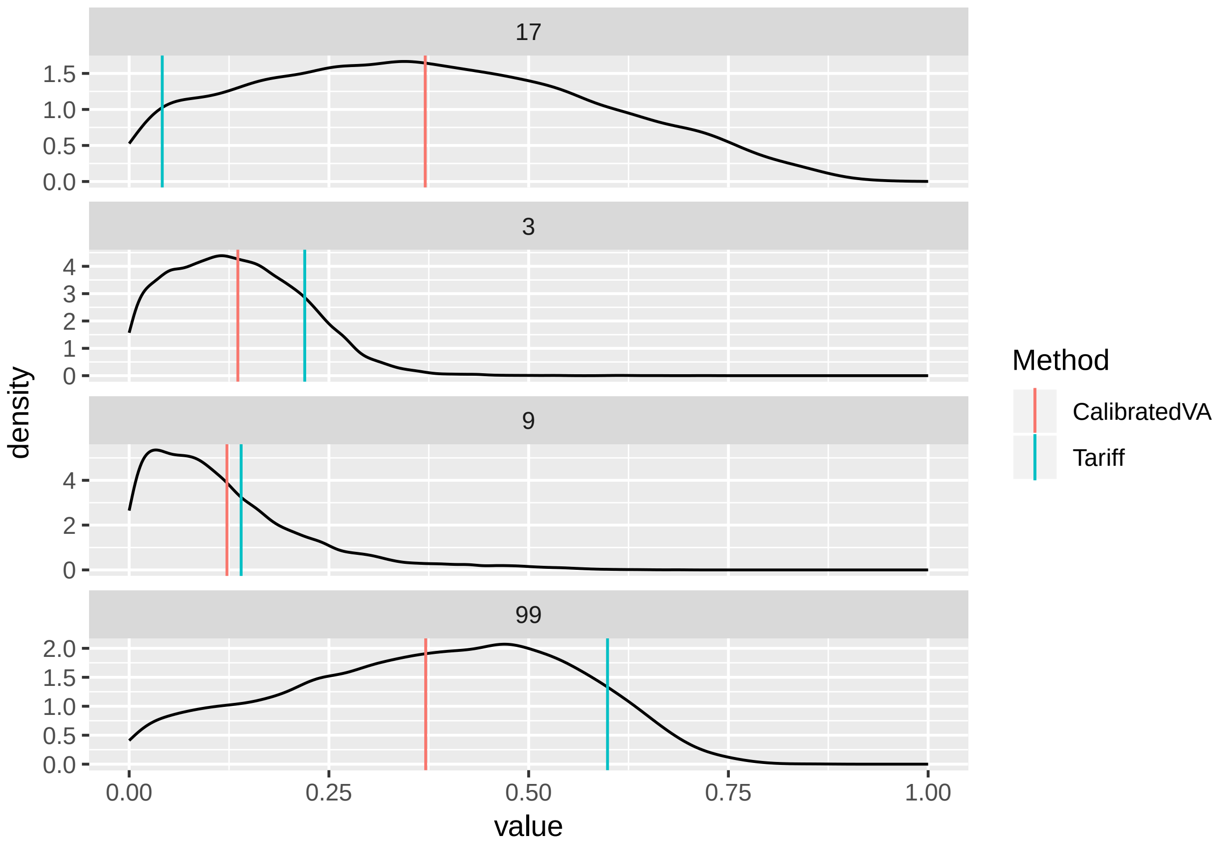
We can plot the Gelman-Rubin Rhat statistics for the CSMF parameters.

```
tariff.calibva.rhat.df <- rename(tariff.calibva.csmf.samples, Parameter = cause)
ggs_Rhat(tariff.calibva.rhat.df)
```



We can plot the posterior densities, along with the posterior mean and the uncalibrated CSMF estimate. We will use built-in helper functions to CalibratedVA to extract these values.

```r
tariff.calibva.csmf.mean <- calibvaCSMFPosteriorSummary(tariff.calibva.csmf.samples)
tariff.csmf.test <- getRawCSMF(tariff.train.cod.test, causes = causes)
ggplot(tariff.calibva.csmf.samples, aes(x = value)) +
  stat_density(geom="line", position = "identity") +
  facet_wrap(~cause, ncol = 1, scales = "free_y") +
  geom_vline(data = tariff.calibva.csmf.mean, aes(xintercept = mean, colour = "CalibratedVA")) +
  geom_vline(data = tariff.csmf.test, aes(xintercept = csmf, colour = "Tariff")) +
  scale_color_discrete(name = "Method") +
  xlim(0, 1)
```



We can see that there is considerable uncertainty in our parameter estimates. This may be due to considerable uncertainty in estimating the misclassification matrix M. We can use built-in functions to look at the posterior mean and variances for each entry of M.

```
tariff.posterior.M <- mMatrixPosteriorSummary(tariff.calibva, causes = causes)
tariff.posterior.M$m_posterior_mean
#>              17          3          9        99
#> 17 0.0720718708 0.25428540 0.08756359 0.5860791
#> 3  0.0759907923 0.67763271 0.03755846 0.2088180
#> 9  0.0003188452 0.16242842 0.42379968 0.4134530
#> 99 0.0222436423 0.04908028 0.17063418 0.7580419
tariff.posterior.M$m_posterior_var
#>              17           3           9          99
#> 17 1.190089e-03 0.003775397 0.001440370 0.005215856
#> 3  2.191296e-03 0.009722714 0.001308573 0.007731396
#> 9  1.477819e-05 0.006911286 0.018113896 0.016698249
#> 99 3.515094e-04 0.001076714 0.002520838 0.003503211
```

We can see that we estimate there to be substantial misclassficiation for individuals with true COD 17 and 9. We can compare the posterior mean of the M matrix, which is shrunken towards the identity matrix, to the empirical estimate of M.

```
T.emp <-rawMisclassificationMatrix(calib.cod = insilico.train.cod.calib,
                                   calib.truth = calibration.changedcod,
                                   causes = causes)
M.emp <- normalizedMisclassificationMatrix(T.emp)
M.emp
#>            17          3          9        99
#> 17 0.2051282 0.3076923 0.02564103 0.4615385
#> 3  0.1764706 0.2352941 0.05882353 0.5294118
#> 9  0.0000000 0.2307692 0.38461538 0.3846154
#> 99 0.1290323 0.1935484 0.09677419 0.5806452
```

Finally, if we were only interested in point estimates of p and M, we can use MAP estimation, with the SQUAREM package

```
tariff.map <- calibva.map(test.cod = tariff.train.cod.test,
                          calib.cod = tariff.train.cod.calib,
                          calib.truth = calibration.changedcod, causes = causes,
                          epsilon = epsilon, alpha=alpha, beta=beta,
                          delta=delta)
tariff.map$p
#> [1] 0.18810950 0.19513478 0.04030788 0.57644784
tariff.map$M
#>              [,1]       [,2]         [,3]      [,4]
#> [1,] 0.1783247899 0.23478980 0.0705616412 0.5163238
#> [2,] 0.0391125706 0.80542915 0.0004960006 0.1549623
#> [3,] 0.0009784711 0.09320667 0.6272629492 0.2785519
#> [4,] 0.0003209710 0.02542264 0.1760326645 0.7982237
```

Implementing Ensemble CalibratedVA

The previous section clearly reveals that CalibratedVA can use the output from any CCVA algorithm. Several CCVA algorithms have been implemented in publicly available software; in any particular analysis, the optimal CCVA is unknown. For example, in the above implentation, if our population (test) set had not already been labeled with the GS COD, we would not know whether it would be best to implement CalibratedVA with Tariff or InSilicoVA. In the code below, we show how to implement the Independent CalibratedVA Independence Ensemble Model, which uses the predictions from multiple VA-based algorithms. We use output from the Tariff and InSilicoVA models built in the previous section.

```
test.cod.mat <- matrix(c(tariff.train.cod.test,  insilico.train.cod.test), ncol = 2)
calib.cod.mat <- matrix(c(tariff.train.cod.calib,  insilico.train.cod.calib), ncol = 2)
ensemble.calibva <- calibva.ensemble.lite.sampler(test.cod.mat = test.cod.mat,
                                                  calib.cod.mat = calib.cod.mat,
                                                  calib.truth = calibration.changedcod,
                                                  causes = causes,
                                                  epsilon = epsilon, alpha=alpha, beta=beta,
                                                  tau.vec=tau.vec, delta=delta,
                                                  gamma.init=gamma.init, ndraws = ndraws,
                                                  nchains = nchains,
                                                  init.seeds = calibva.seeds)
ensemble.calibva <- window(ensemble.calibva, start = 10E3, thin = 10)
```

Comparing algorithm accuracy

A common metric for comparing CSMF predictions is CSMF Accuracy, which is defined as

$$CSMF_{acc} = 1 - \frac{\sum_{i=1}^{C}(CSMF_i - CSMF_i^{(true)})}{2(1 - \min\{CSMF^{(true)}\})}$$

where $CSMF_i$ and $CSMF_i^{(true)}$ are the estimated and true percentages, respectively, of the population for which the GS COD is cause i, $i = 1, \ldots, C$.

To obtain estimates of the CSMFs, we will use the marginal posterior means from the posterior draws. We will now obtain these CSMF estimates, using the causes defined in the code block where we set the CalibratedVA hyperparameter values.

```
tariff.calibva.csmf.samples <- calibvaCSMFPosteriorSamples(tariff.calibva, causes = causes)
tariff.calibva.csmf.mean <- calibvaCSMFPosteriorSummary(tariff.calibva.csmf.samples)

insilico.calibva.csmf.samples <- calibvaCSMFPosteriorSamples(insilico.calibva, causes = causes)
insilico.calibva.csmf.mean <- calibvaCSMFPosteriorSummary(insilico.calibva.csmf.samples)

ensemble.calibva.csmf.samples <- calibvaCSMFPosteriorSamples(ensemble.calibva, causes = causes)
ensemble.calibva.csmf.mean <- calibvaCSMFPosteriorSummary(ensemble.calibva.csmf.samples)

tariff.train.csmf <- getRawCSMF(tariff.train.cod.test, causes)
insilico.train.csmf <- getRawCSMF(insilico.train.cod.test, causes)
```

Finally, we will obtain and display the CSMF accuracy for these 5 methods.

```
methods <- c("tariff_calibva",
             "insilico_calibva",
             "ensemble_calibva",
             "tariff_train",
             "insilico_train")
ptrue <- sapply(causes, function(c) mean(change.cause(tanzania.data$Cause) == c))
csmf.acc.df <- data.frame(csmf.acc = c(getCSMF_accuracy(tariff.calibva.csmf.mean$mean, ptrue),
                                       getCSMF_accuracy(insilico.calibva.csmf.mean$mean, ptrue),
                                       getCSMF_accuracy(ensemble.calibva.csmf.mean$mean, ptrue),
                                       getCSMF_accuracy(tariff.train.csmf$csmf, ptrue),
                                       getCSMF_accuracy(insilico.train.csmf$csmf, ptrue)),
                         method = methods)
csmf.acc.df
#>    csmf.acc           method
#> 1 0.9520732   tariff_calibva
#> 2 0.8541614 insilico_calibva
#> 3 0.9005139 ensemble_calibva
#> 4 0.6524886     tariff_train
#> 5 0.7045056   insilico_train
```

We see two interesting observations. First, all three CalibratedVA methods substantially outperform Tariff and InSilicoVA in terms of CSMF accuracy. Second, The CSMF accuracy of the ensemble method of CalibratedVA falls right in the middle of the CSMF accuracies of Tariff and CalibratedVA. This indicates that it is a useful method to use, as it is always unknown whether Tariff or InSilicoVA will be the best performing algorithm.

References

AbouZahr, Carla, Don De Savigny, Lene Mikkelsen, Philip W Setel, Rafael Lozano, Erin Nichols, Francis Notzon, and Alan D Lopez. 2015. "Civil Registration and Vital Statistics: Progress in the Data Revolution for Counting and Accountability." *The Lancet* 386 (10001). Elsevier: 1373–85.

Allotey, Pascale A, Daniel D Reidpath, Natalie C Evans, Nirmala Devarajan, Kanason Rajagobal, Ruhaida Bachok, Kridaraan Komahan, and SEACO Team. 2015. "Let's Talk About Death: Data Collection for Verbal Autopsies in a Demographic and Health Surveillance Site in Malaysia." *Global Health Action* 8 (1). Taylor & Francis: 28219.

Byass, Peter, Daniel Chandramohan, Samuel J Clark, Lucia D'ambruoso, Edward Fottrell, Wendy J Graham, Abraham J Herbst, et al. 2012. "Strengthening Standardised Interpretation of Verbal Autopsy Data: The New Interva-4 Tool." *Global Health Action* 5 (1). Taylor & Francis: 19281.

James, Spencer L, Abraham D Flaxman, and Christopher JL Murray. 2011. "Performance of the Tariff Method: Validation of a Simple Additive Algorithm for Analysis of Verbal Autopsies." *Population Health Metrics* 9 (1). BioMed Central: 31.

McCormick, Tyler H, Zehang Richard Li, Clara Calvert, Amelia C Crampin, Kathleen Kahn, and Samuel J Clark. 2016. "Probabilistic Cause-of-Death Assignment Using Verbal Autopsies." *Journal of the American Statistical Association* 111 (515). Taylor & Francis: 1036–49.

Serina, Peter, Ian Riley, Andrea Stewart, Spencer L James, Abraham D Flaxman, Rafael Lozano, Bernardo Hernandez, et al. 2015. "Improving Performance of the Tariff Method for Assigning Causes of Death to Verbal Autopsies." *BMC Medicine* 13 (1). BioMed Central: 291.

Soleman, Nadia, Daniel Chandramohan, and Kenji Shibuya. 2006. "Verbal Autopsy: Current Practices and Challenges." *Bulletin of the World Health Organization* 84. SciELO Public Health: 239–45.