Demonstration of the COMBO R Package

Created by Kimberly A. Hochstedler. Contact: kah343@cornell.edu

2022-10-11

In this vignette, we provide a demonstration of the R Package *COMBO* (correcting misclassified binary outcomes). This package provides methods for fitting logistic regression models when the binary outcome is potentially misclassified. Technical details about estimation are not included in this demonstration. For additional information on the methods used in this R Package, please consult "Statistical inference for association studies in the presence of binary outcome misclassification" by Kimberly A. Hochstedler and Martin T. Wells.

Model and Conceptual Framework

Let $Y = j$ denote an observation's true outcome status, taking values $j \in \{1, 2\}$. Suppose we are interested in the relationship between Y and a set of predictors, X, that are correctly measured. This relationship constitutes the *true outcome mechanism*. Let $Y^* = k$ be the observed outcome status, taking values $k \in \{1, 2\}$. Y^* is a potentially misclassified version of Y. Let Z denote a set of predictors related to sensitivity and specificity. The mechanism that generates the observed outcome, Y^*, given the true outcome, Y, is called the *observation mechanism*. **Figure 1** displays the conceptual model. The following equations express the conceptual process mathematically.

$$\text{True outcome mechanism: } \text{logit}\{P(Y = j|X; \beta)\} = \beta_{j0} + \beta_{jX}X$$

$$\text{Observation mechanism: } \text{logit}\{P(Y^* = k|Y = j, Z; \gamma)\} = \gamma_{kj0} + \gamma_{kjZ}Z$$

Simulate data

We begin this demonstration by generating data using the `COMBO_data()` function. The binary outcome data simulated by this scheme is subject to misclassification. The predictor related to the true outcome mechanism is "x" and the predictor related to the observation mechanism is "z".

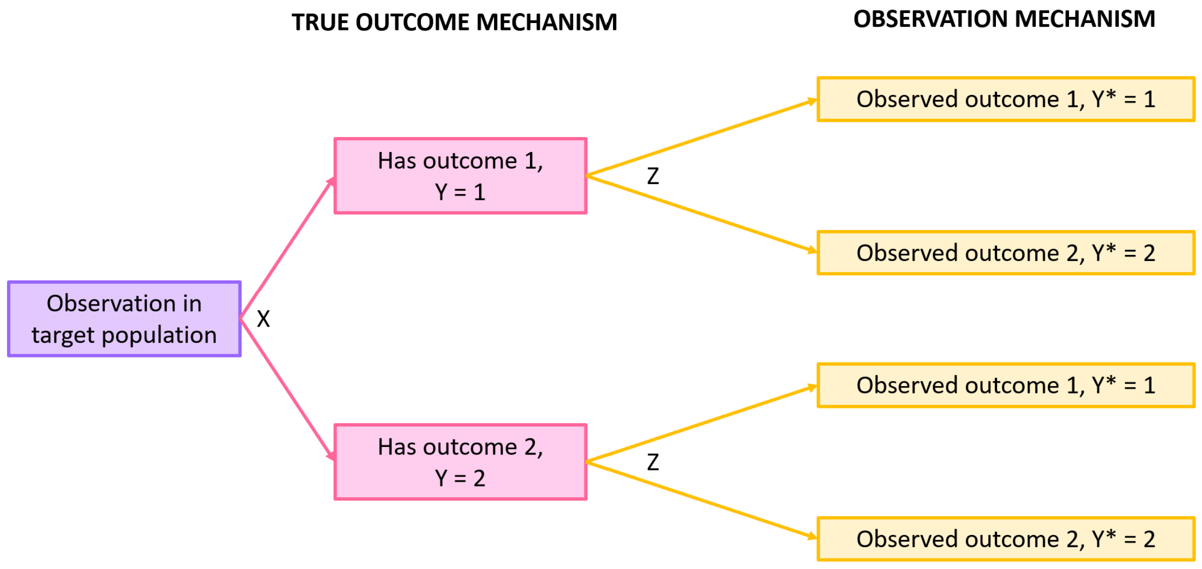

TRUE OUTCOME MECHANISM
OBSERVATION MECHANISM
Observation in target population
X
Has outcome 1, Y = 1
Has outcome 2, Y = 2
Z
Z
Observed outcome 1, Y* = 1
Observed outcome 2, Y* = 2
Observed outcome 1, Y* = 1
Observed outcome 2, Y* = 2

Figure 1: Conceptual Model

```
library(COMBO)
library(dplyr)

# Set seed.
set.seed(123)

# Set sample size, x and z distribution information.
n <- 1000
x_mu <- 0
x_sigma <- 1
z_shape <- 1

# Set true parameter values.
true_beta <- matrix(c(1, -2), ncol = 1)
true_gamma <- matrix(c(.5, 1, -.5, -1), nrow = 2, byrow = FALSE)

# Generate data.
my_data <- COMBO_data(sample_size = n,
                      x_mu = x_mu, x_sigma = x_sigma,
                      z_shape = z_shape,
                      beta = true_beta, gamma = true_gamma)

# Save list elements as vectors.
Ystar <- my_data[["obs_Y"]]
x_matrix <- my_data[["x"]]
z_matrix <- my_data[["z"]]
```

Effect estimation

We propose estimation methods using the Expectation-Maximization algorithm (EM) and Markov Chain Monte Carlo (MCMC). Each method checks and corrects instances of label switching, as described in Hochstedler and Wells (2022). In the code below, we provide functions for implementing these methods.

```
# Supply starting values for all parameters.
starting_values <- rep(1,6)
beta_start <- matrix(starting_values[1:2], ncol = 1)
gamma_start <- matrix(starting_values[3:6], ncol = 2, nrow = 2, byrow = FALSE)

# Estimate parameters using the EM-Algorithm.
EM_results <- COMBO_EM(Ystar, x_matrix = x_matrix, z_matrix = z_matrix,
                       beta_start = beta_start, gamma_start = gamma_start)

EM_results
```

```
##        Parameter    Estimates          SE Convergence
## 1          beta1   0.94890120  0.27589385        TRUE
## 2          beta2  -2.35565018  0.16518517        TRUE
## 3        gamma11   0.53773267  0.12881774        TRUE
## 4        gamma21   0.94899526  0.25802541        TRUE
## 5        gamma12  -0.03143942  0.17206710        TRUE
## 6        gamma22  -1.42142132  0.21022199        TRUE
## 7    SAMBA_beta1   1.02579269  0.29776853          NA
## 8    SAMBA_beta2  -1.13762209  0.20119894          NA
## 9  SAMBA_gamma11   1.22380033  0.30354598          NA
## 10 SAMBA_gamma21   0.57807251  0.35267049          NA
## 11   PSens_beta1   0.36958220  7.02366053          NA
## 12   PSens_beta2  -0.72231505  2.81551719          NA
## 13 PSens_gamma12  -0.75532783  3.28479253          NA
## 14 PSens_gamma22 -43.39974465 22.14360437          NA
## 15   naive_beta1   0.38315779  0.06811635        TRUE
## 16   naive_beta2  -0.71541393  0.07536077        TRUE
```

```
# Specify parameters for the prior distributions.
unif_lower_beta <- matrix(c(-5, -5, NA, NA), nrow = 2, byrow = TRUE)
unif_upper_beta <- matrix(c(5, 5, NA, NA), nrow = 2, byrow = TRUE)

unif_lower_gamma <- array(data = c(-5, NA, -5, NA, -5, NA, -5, NA),
                          dim = c(2,2,2))
unif_upper_gamma <- array(data = c(5, NA, 5, NA, 5, NA, 5, NA),
                          dim = c(2,2,2))

beta_prior_parameters <- list(lower = unif_lower_beta, upper = unif_upper_beta)
gamma_prior_parameters <- list(lower = unif_lower_gamma, upper = unif_upper_gamma)

# Estimate parameters using MCMC.
MCMC_results <- COMBO_MCMC(Ystar, x = x_matrix, z = z_matrix,
                           prior = "uniform",
                           beta_prior_parameters = beta_prior_parameters,
                           gamma_prior_parameters = gamma_prior_parameters,
                           number_MCMC_chains = 4,
                           MCMC_sample = 2000, burn_in = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 6
##    Total graph size: 35030
##
## Initializing model
##
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 2
##    Total graph size: 12013
##
## Initializing model
```

```
MCMC_results$posterior_means_df
```

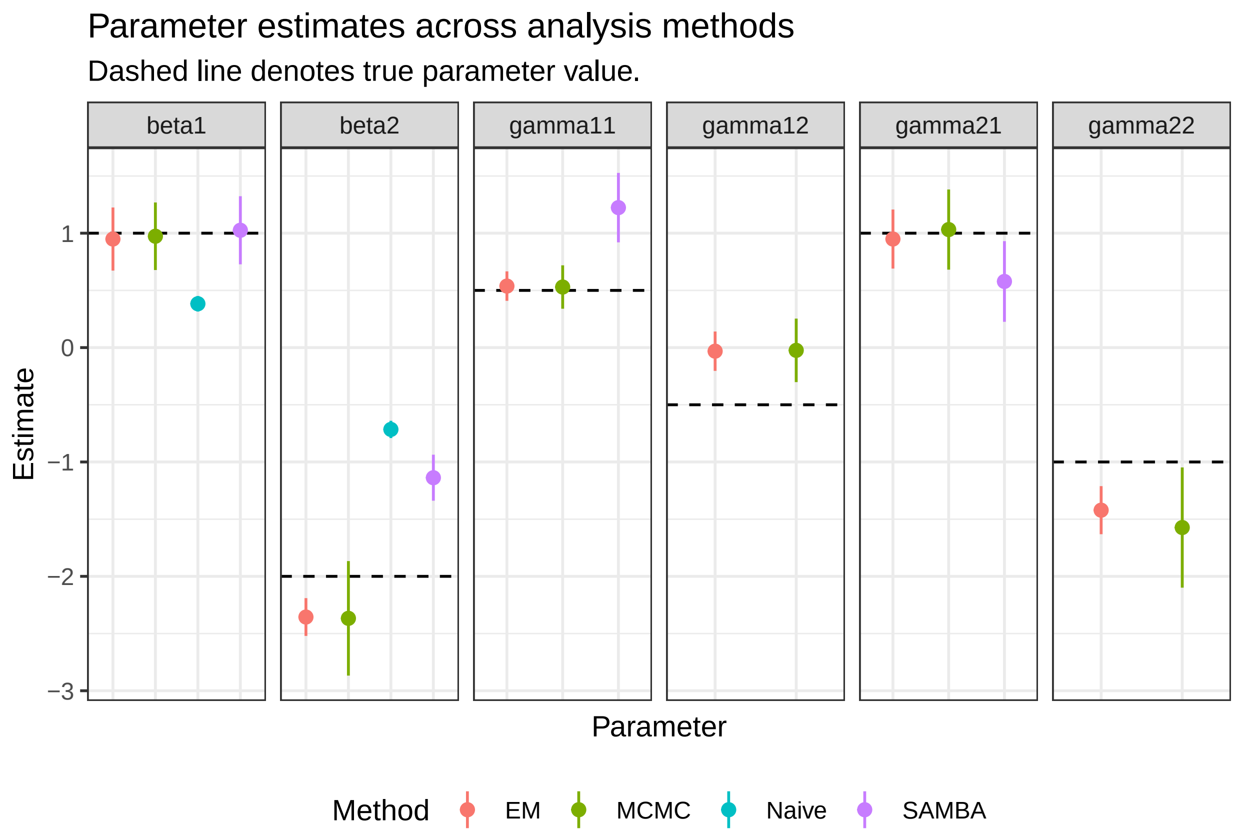
```
## # A tibble: 6 x 3
##   parameter    posterior_mean posterior_median
##   <fct>                 <dbl>            <dbl>
## 1 beta[1,1]            0.973            0.955
## 2 beta[1,2]           -2.37            -2.31
## 3 gamma[1,1,1]         0.529            0.530
## 4 gamma[1,2,1]        -0.0242          -0.0259
## 5 gamma[1,1,2]         1.03             0.990
## 6 gamma[1,2,2]        -1.57            -1.51
```

```
MCMC_results$naive_posterior_means_df
```

```
## # A tibble: 2 x 3
##   parameter       posterior_mean posterior_median
##   <chr>                    <dbl>            <dbl>
## 1 naive_beta[1,1]          0.384            0.384
## 2 naive_beta[1,2]         -0.718           -0.718
```

Plotting effect estimates

Figure 2 shows the parameter estimates (+/- one standard deviation) for different analysis methods: EM, MCMC, SAMBA (an R package that estimates a binary outcome misclassification model, assuming perfect specificity), and a "naive" logistic regression of $Y^*|X$.



Estimating sensitivity and specificity

For each analysis method, we may use the estimated γ parameters to compute estimates of sensitivity and specificity as a function of the covariate, z. Here, we compute these values under the EM algorithm estimates, MCMC estimates, and using the generated data.

```r
# Create matrix of gamma parameter estimates from the EM algorithm.
EM_gamma <- matrix(EM_results$Estimates[3:6], ncol = 2, byrow = FALSE)

# Compute misclassification probabilities.
EM_misclassification_prob <- misclassification_prob(EM_gamma,
                                                    matrix(z_matrix, ncol = 1))
# Find the average sensitivity and specificity.
EM_sensitivity_df <- EM_misclassification_prob %>%
  filter(Y == 1) %>% filter(Ystar == 1)
EM_sensitivity <- mean(EM_sensitivity_df$Probability)

EM_specificity_df <- EM_misclassification_prob %>%
  filter(Y == 2) %>% filter(Ystar == 2)
EM_specificity <- mean(EM_specificity_df$Probability)
```

```r
# Create matrix of gamma parameter estimates from MCMC.
MCMC_gamma <- matrix(MCMC_results$posterior_means_df$posterior_mean[3:6],
                     ncol = 2, byrow = TRUE)
```

```r
# Compute misclassification probabilities.
MCMC_misclassification_prob <- misclassification_prob(MCMC_gamma,
                                                      matrix(z_matrix, ncol = 1))

# Find the average sensitivity and specificity
MCMC_sensitivity_df <- MCMC_misclassification_prob %>%
  filter(Y == 1) %>% filter(Ystar == 1)
MCMC_sensitivity <- mean(MCMC_sensitivity_df$Probability)

MCMC_specificity_df <- MCMC_misclassification_prob %>%
  filter(Y == 2) %>% filter(Ystar == 2)
MCMC_specificity <- mean(MCMC_specificity_df$Probability)
```

```r
# Use the generated data to compute the actual sensitivity and specificity rate.
data_classification_table <- table(my_data[["obs_Y"]], my_data[["true_Y"]])

true_sensitivity <- prop.table(data_classification_table, 2)[1,1]

true_specificity <- prop.table(data_classification_table, 2)[2,2]
```

	Sensitivity, $P(Y^* = 1 \mid Y = 1)$	Specificity, $P(Y^* = 2 \mid Y = 2)$
Data	0.779	0.802
EM	0.781	0.743
MCMC	0.787	0.755

Table 1 shows the actual sensitivity and specificity values for the data, in addition to the average sensitivity and specificity estimates computed from EM-Algorithm and MCMC parameter estimates and the covariate z.

References

Hochstedler, K.A. and Wells, M.T. "Statistical inference for association studies in the presence of binary outcome misclassification", (2022). In preparation.