BIOS 736: HW4

Thomas Hsiao

Question 1

Setup

We derive the conditional score for logistic regression. Suppose our parameter of interest is $\theta = (\alpha, \beta')'$. Then for response variable Y, under the assumption of classical measurement error model with normal error and known error variance, our model is

$$Y_i|\theta, \mathbf{x} \sim \text{Bern}(\text{expit}(\alpha + \beta'\mathbf{x}))$$

$$\mathbf{W}_i \sim N(\mathbf{X}_i, \Sigma_\epsilon), \epsilon_i \perp \{Y, \mathbf{X}_i\}$$

Using rules of the exponential family, a complete and sufficient statistic for $\mathbf{x}$ is $\Delta = \mathbf{W} + Y\Sigma\beta$.

From pg. 5 in the lecture notes, denote joint pdf h, the conditional score is equal to

$$\Psi_c(Y, \Delta, \theta, \mathbf{x}) = \frac{\partial}{\partial\theta} \log h_{Y,\mathbf{W}} - E\left(\frac{\partial}{\partial\theta} \log h_{Y,\mathbf{W}}|\Delta\right)$$

Since Y and W are independent and $\mathbf{W}$ is not a function of θ, we can further simplify to

$$\Psi_c(Y, \Delta, \theta, \mathbf{x}) = \frac{\partial}{\partial\theta} \log h_Y - E\left(\frac{\partial}{\partial\theta} \log h_Y|\Delta\right)$$

Since the above expression is unbiased, regardless for any value of the true covariates $\mathbf{x}$, we can solve for θ by evaluating the conditional score at $\Psi_c(Y, \Delta, \theta, t(\Delta))$ for some function t.

Derivation

We start by taking the derivative of h_Y with respect to θ.

$$h_Y(y;\theta,\mathbf{x}) = \exp\left\{y(\alpha+\beta'\mathbf{x}) - \log(1+\exp(\alpha+\beta'\mathbf{x}))\right\}$$

$$\frac{\partial}{\partial\theta}\log h_Y = \{y - \text{expit}(\alpha+\beta'\mathbf{x})\}\begin{pmatrix}1\\ \mathbf{x}\end{pmatrix}$$

That takes care of the first part in the conditional score. For the second part, we take the first part and evaluate its conditional expectation with respect to Δ.

$$E\left(\frac{\partial}{\partial\theta}\log h_Y|\Delta\right) = \{E(Y|\Delta) - \text{expit}(\alpha+\beta'\mathbf{x})\}\begin{pmatrix}1\\ \mathbf{x}\end{pmatrix}$$

From pg. 8 in the lecture notes, we know that

$$E(Y|\Delta) = P(Y=1|\Delta) = \text{expit}(\alpha+\beta'\Delta^*)$$

where $\Delta^* = \mathbf{W} + (Y-1/2)\Sigma\beta$. Putting both pieces together for the conditional score we get

$$\Psi_c(Y,\Delta,\theta,\mathbf{x}) = \{Y - \text{expit}(\alpha+\beta'\Delta^*)\}\begin{pmatrix}1\\ \mathbf{x}\end{pmatrix}$$

And replacing $\mathbf{x}$ we get

$$\Psi_c(Y,\Delta,\theta,t(\Delta)) = \{Y - \text{expit}(\alpha+\beta'\Delta^*)\}\begin{pmatrix}1\\ t(\Delta)\end{pmatrix}$$

Question 2

We now implement the conditional score estimator and compare its performance to regression calibration and naive estimators in a simulation experiment. For $t(\Delta)$, we replace with Δ^* so that the conditional score is equivalent to the sufficiency score.

Table 1: Bias for alpha for conditional score

N	SIGMA2	CS	Naive	RC	True
100	0.5	0.10878	-0.00272	-0.02451	0.00526
100	1.0	-0.08762	0.01372	-0.02256	0.01978
300	0.5	0.01114	0.00606	-0.01708	0.01505
300	1.0	-0.12766	-0.00036	-0.03449	0.00293

Table 2: Standard deviation for alpha for conditional score

N	SIGMA2	CS	Naive	RC	True
100	0.5	1.83264	0.21904	0.22528	0.23244
100	1.0	3.29391	0.22770	0.24445	0.23911
300	0.5	1.18618	0.12543	0.13203	0.13072
300	1.0	1.55089	0.12404	0.13319	0.12848

Table 3: Bias for beta for conditional score

N	SIGMA2	CS	Naive	RC	True
100	0.5	1.68448	-0.37470	-0.11821	0.04054
100	1.0	4.35570	-0.54114	-0.15661	0.04538
300	0.5	1.15071	-0.37703	-0.13659	0.03224
300	1.0	2.99847	-0.54329	-0.19078	0.02740

Table 4: SD for beta for conditional score

N	SIGMA2	CS	Naive	RC	True
100	0.5	7.14238	0.20096	0.32128	0.31391
100	1.0	9.78065	0.16993	0.38864	0.31940
300	0.5	6.25948	0.10945	0.19022	0.17150
300	1.0	8.18888	0.08515	0.20286	0.15935

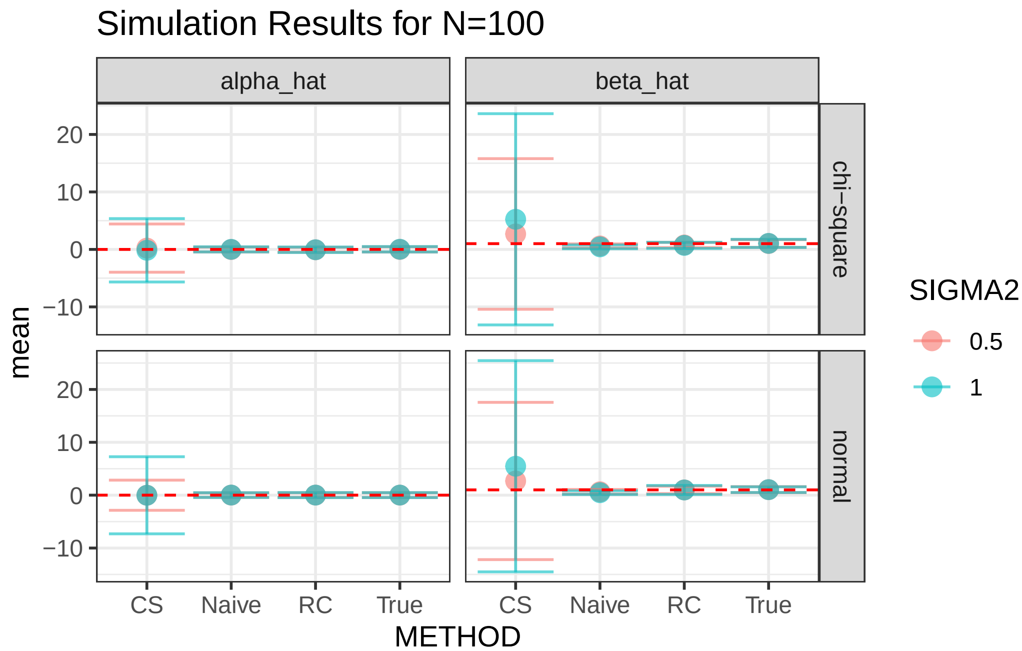

Figure 1: Simulation results for n=100

The CS method is influenced by some outliers in the estimates of both alpha and beta. This causes the mean to be shifted, and the confidence intervals to be much wider compared to the Naive or RC methods. We can get a better idea of the extent of the root-finding problem by looking at a histogram of estimates for both parameters for the $n = 300$.

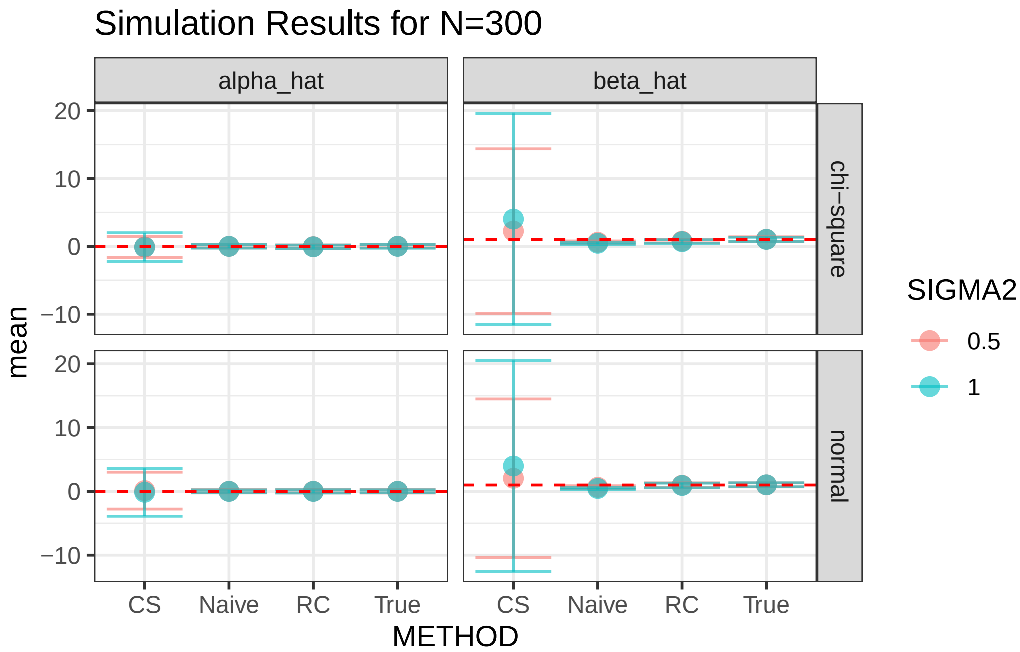

Figure 2: Simulation results for n=300

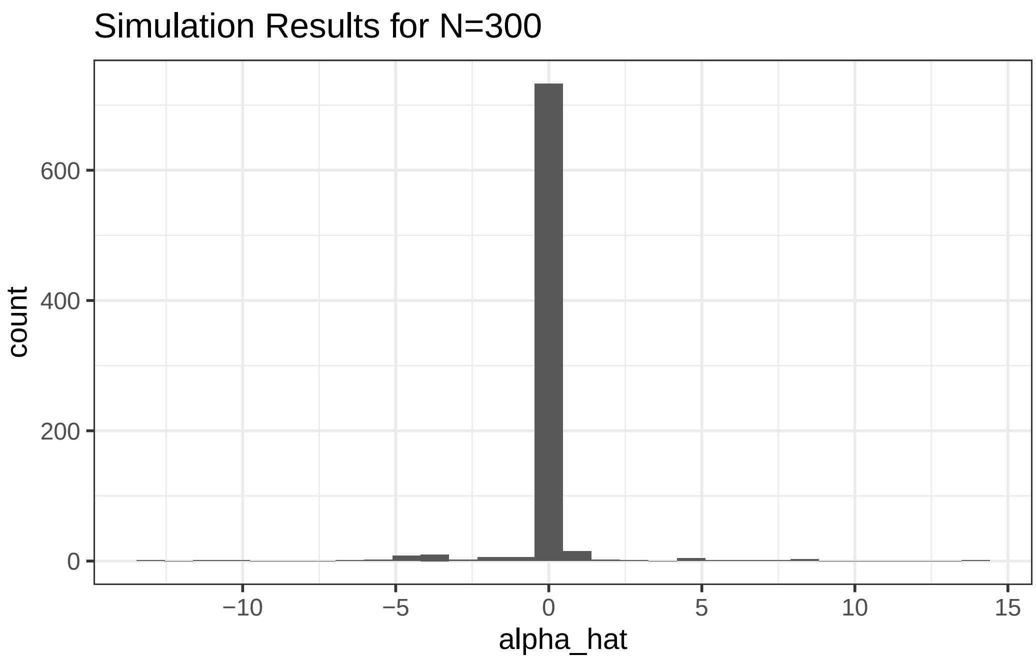

Figure 3: Simulation results for n=300

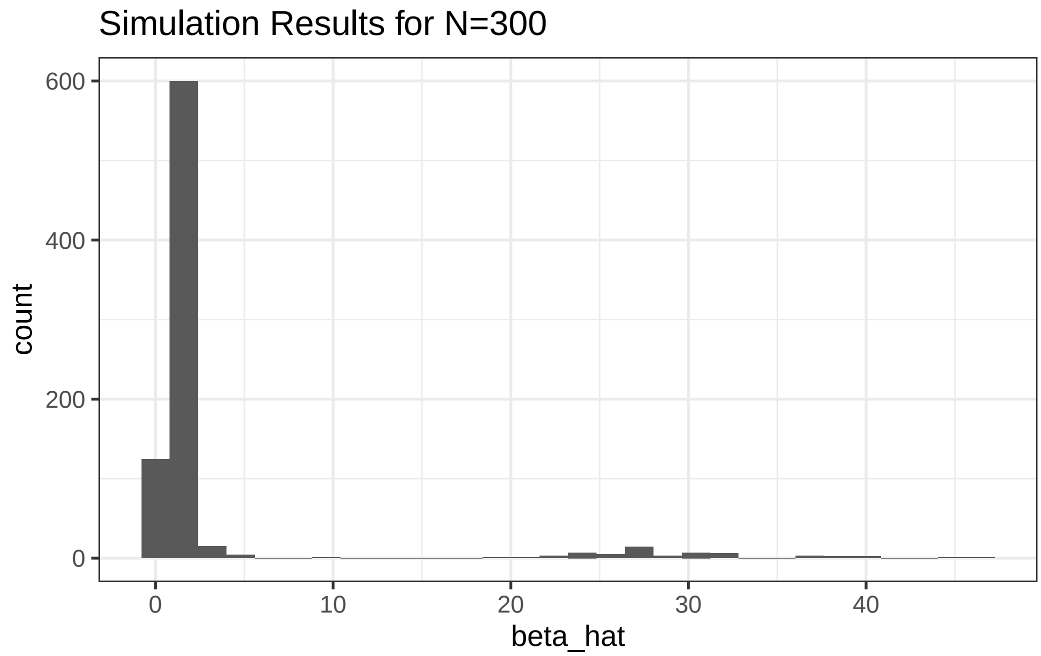

Figure 4: Simulation results for n=300

Code Appendix

```
library(data.table)
library(ggplot2)
inv_logit <- function(x) {
  exp(x) / (1 + exp(x))
}

# Score function. We return the Euc norm of the gradient for optimization purposes.
cscore <- function(theta, Y, W, sigma2_eps){
  alpha <- theta[1]
  beta  <- theta[2]

  css <- W + (Y-1/2)*sigma2_eps*beta
  tmp <- Y - inv_logit(alpha + beta*css)

  score <- sqrt( sum(c(sum(tmp), sum(tmp*css) )^2) )
  return(score)
}
```

```r
MLE <- function(Y, X){
  fit <- glm(Y ~ X, family = binomial)
  est <- fit$coefficients
  names(est) <- c("alpha", "beta")
  return(fit$coefficients)
}

RC <- function(Y, W, sigma2_eps){
  mu_X  <- mean(W)
  var_W <- var(W)
  var_X <- var_W - sigma2_eps

  Xhat <- mu_X + var_X * var_W^(-1) * (W - mu_X)

  return(MLE(Y, Xhat))
}

# Conditional score method
CS <- function(Y, W, sigma2_eps){
  fit <- optim(c(0,0), fn=cscore, Y=Y, W=W,sigma2_eps=sigma2_eps)
  return(fit$par)
}

sim_data <- function(N, alpha, beta, sigma2_eps, X_model = "normal"){
  # Simulate True Covariates
  if(X_model == "normal"){
    X <- rnorm(N)
  } else if(X_model == "chi-square"){
    X <- ( rchisq(N, 1) - 1 / (sqrt(2)) )
  }

  # Simulate ME
  eps <- rnorm(N, sd = sqrt(sigma2_eps))

  # Define surrogate
  W <- X + eps

  # Linear predictor
  eta <- alpha + beta*X

  # Simulate Y
```

```r
  Y <- sapply(inv_logit(eta), function(p){rbinom(1, 1, p)})

  return(list(Y=Y, X=X, W=W))
}
SIM      <- 1:200
N      <- c(100, 300)
SIGMA2 <- c(.5, 1)
XDIST  <- c("normal", "chi-square")
METHOD <- c("True", "Naive", "RC", "CS")
alpha_hat <- 0; beta_hat <- 0
results <- data.table::CJ(SIM, N, SIGMA2, XDIST, METHOD, alpha_hat, beta_hat)

# alpha <- 0; beta <- 1
# n <- 100
# sigma2 <- .4
# xdist <- "normal"
# i <- 1

for(n in N){
  for(sigma2 in SIGMA2){
    for(xdist in XDIST){
      for(i in SIM){
        data <- sim_data(n, alpha, beta, sigma2, xdist)
        Y <- data$Y
        X <- data$X
        W <- data$W
        for(method in METHOD){
          if(method == "True"){
            est <- MLE(Y, X)
          }
          if(method == "Naive"){
            est <- MLE(Y, W)
          }
          if(method == "RC"){
            est <- RC(Y, W, sigma2)
          }
          if(method == "CS"){
            est <- CS(Y, W, sigma2)
          }

          results[SIM == i &
```

```
                    N == n &
                    SIGMA2 == sigma2 &
                    XDIST == xdist &
                    METHOD == method,
                  `:=`(alpha_hat = est[1],
        }
      }
    }
  }
}

# fwrite(results, "HW4_simulation_results.csv")
results <- fread("HW4_simulation_results.csv")
summary <- melt(results, id.vars = c("N", "SIGMA2", "XDIST", "METHOD"),
     measure.vars=c("alpha_hat", "beta_hat"),
     variable.name = "parameter", value.name = "est")

final <- summary[, .(mean = mean(est),
            se = var(est)^.5), .(N, SIGMA2, XDIST, METHOD, parameter)]
final[, `:=`(lower = mean - 1.96*se, upper = mean+1.96*se)]
final[, truth := ifelse(parameter == "alpha_hat", 0, 1)]

final[, SIGMA2 := as.factor(SIGMA2)]
final[, N := as.factor(N)]

alpha_table <- results[, .(bias = mean(alpha_hat), sd = sd(alpha_hat)), .(N, SIGMA2, METHO
beta_table <- results[, .(bias = mean(beta_hat - 1), sd = sd(beta_hat)), .(N, SIGMA2, METH
knitr::kable(dcast(alpha_table, N + SIGMA2 ~ METHOD, value.var = c("bias")), digits=5, cap
knitr::kable(dcast(alpha_table, N + SIGMA2 ~ METHOD, value.var = c("sd")), digits=5, capti
knitr::kable(dcast(beta_table, N + SIGMA2 ~ METHOD, value.var = c("bias")), digits=5, capt
knitr::kable(dcast(beta_table, N + SIGMA2 ~ METHOD, value.var = c("sd")), digits=5, captio
ggplot(final[N==100], aes(METHOD, mean, col=SIGMA2)) +
  geom_point(alpha=0.6, size=3) +
  geom_errorbar(aes(ymin=lower, ymax=upper), alpha=0.6, size=0.5) +
  facet_grid( XDIST ~ parameter, scales = "free") +
  geom_hline(aes(yintercept=truth), linetype = "dashed", col="red") +
  theme_bw() +
  ggtitle("Simulation Results for N=100")
ggplot(final[N==300], aes(METHOD, mean, col=SIGMA2)) +
  geom_point(alpha=0.6, size=3) +
  geom_errorbar(aes(ymin=lower, ymax=upper), alpha=0.6, size=0.5) +
```

```
  facet_grid( XDIST ~ parameter, scales = "free") +
  geom_hline(aes(yintercept=truth), linetype = "dashed", col="red") +
  theme_bw() +
   ggtitle("Simulation Results for N=300")
ggplot(results[N==300 & METHOD == "CS"], aes(alpha_hat)) +
  geom_histogram() +
  theme_bw() +
   ggtitle("Simulation Results for N=300")
ggplot(results[N==300 & METHOD == "CS"], aes(beta_hat)) +
  geom_histogram() +
  theme_bw() +
   ggtitle("Simulation Results for N=300")
```