BIOS736: HW3

Thomas Hsiao

Question 1

We distinguish the observed design matrix W and the random vector $\mathbf{W}$. The same convention holds for X and $\mathbf{X}$.

Recall from the notes that the RC estimator can be written as a function of the naive estimator as follows

$$\hat{\alpha}_{RC} = \hat{\alpha}_{NV} - \hat{\beta}_{RC}\{\hat{\mu}_X - \hat{\Sigma}_X(\hat{\Sigma}_X + \hat{\Sigma}_\epsilon)^{-1}(\hat{\mu}_X + \hat{\mu}_\epsilon)\} \tag{1}$$

$$\hat{\beta}_{RC} = \hat{\Sigma}_X^{-1}(\hat{\Sigma}_X + \hat{\Sigma}_\epsilon)\hat{\beta}_{NV} \tag{2}$$

Then we can first find what the naive estimator for the coefficients converge to in probability. We break up the naive estimator into two parts

$$\hat{\beta}_{NV} = (W'W)^{-1}W'Y \tag{3}$$

Part 1: showing consistency

First part

$$n^{-1}\begin{pmatrix}1_n & W\end{pmatrix}'\begin{pmatrix}1_n & W\end{pmatrix} = n^{-1}\begin{pmatrix}n & 1_n'W \\ W'1_n & W'W\end{pmatrix} \to \begin{pmatrix}1 & \mu_W' \\ \mu_W & \Sigma_W + \mu_W^{\otimes 2}\end{pmatrix}$$

Second part

$$n^{-1}\begin{pmatrix}1_n & W\end{pmatrix}'Y = n^{-1}\begin{pmatrix}1_n & W\end{pmatrix}'\left[\begin{pmatrix}1_n & X\end{pmatrix}\begin{pmatrix}\alpha \\ \beta\end{pmatrix} + e\right] =$$

$$n^{-1}\begin{pmatrix}1_n & X+\epsilon\end{pmatrix}'(\alpha_n + X\beta + e) = n^{-1}\begin{pmatrix}1_n'(\alpha_n + X\beta + e)\\ (X+\epsilon)'(\alpha_n + X\beta + e)\end{pmatrix} =$$

$$n^{-1}\begin{pmatrix}1_n'\alpha_n + 1_n'X\beta + 1_n'e\\ X'\alpha_n + X'X\beta + X'e + \epsilon'\alpha_n + \epsilon'X\beta + \epsilon'e\end{pmatrix} \to \begin{pmatrix}\alpha + \mu_X'\beta\\ \alpha(\mu_X + \mu_\epsilon) + (\Sigma_X + \mu_X^{\otimes 2})\beta\end{pmatrix}$$

Putting the pieces together, we have that the naive estimator converges to

$$\begin{pmatrix}1 & \mu_W'\\ \mu_W & \Sigma_W + \mu_W^{\otimes 2}\end{pmatrix}^{-1}\begin{pmatrix}\alpha + \mu_X'\beta\\ \alpha(\mu_X + \mu_\epsilon) + (\Sigma_X + \mu_X^{\otimes 2})\beta\end{pmatrix}$$

Using the formula for the inverse of 2×2 block matrix, we have

$$\begin{pmatrix}1 + \mu_W'\Sigma_W^{-1}\mu_W & -\mu_W'\Sigma_W^{-1}\\ -\Sigma_W^{-1}\mu_W & \Sigma_W^{-1}\end{pmatrix}\begin{pmatrix}\alpha + \mu_X'\beta\\ \alpha(\mu_X + \mu_\epsilon) + (\Sigma_X + \mu_X^{\otimes 2})\beta\end{pmatrix}$$

With some algebraic manipulation, we arrive at the following convergence result for the naive estimator.

$$\hat{\beta}_{NV} \to (\Sigma_X + \Sigma_\epsilon)^{-1}\Sigma_X\beta \tag{4}$$

$$\hat{\alpha}_{NV} \to \alpha + \beta\{\mu_X - \Sigma_X(\Sigma_X + \Sigma_\epsilon)^{-1}(\mu_X + \mu_\epsilon)\} \tag{5}$$

Combining the results of 1 and 2 with the above, using WLLN and CMT, we arrive at the consistency of the RC estimator for both α and β, without enforcing any distributional assumptions on $\mathbf{X}$ and ϵ.

Part 2: assumptions on the moments

If we do not know beforehand or cannot estimate μ_ϵ, then we cannot estimate α since 1 requires $\hat{\mu}_\epsilon$. However, $\hat{\mu}_\epsilon$ is not necessary for estimating β. Based on 2, we only require an estimate for Σ_ϵ.

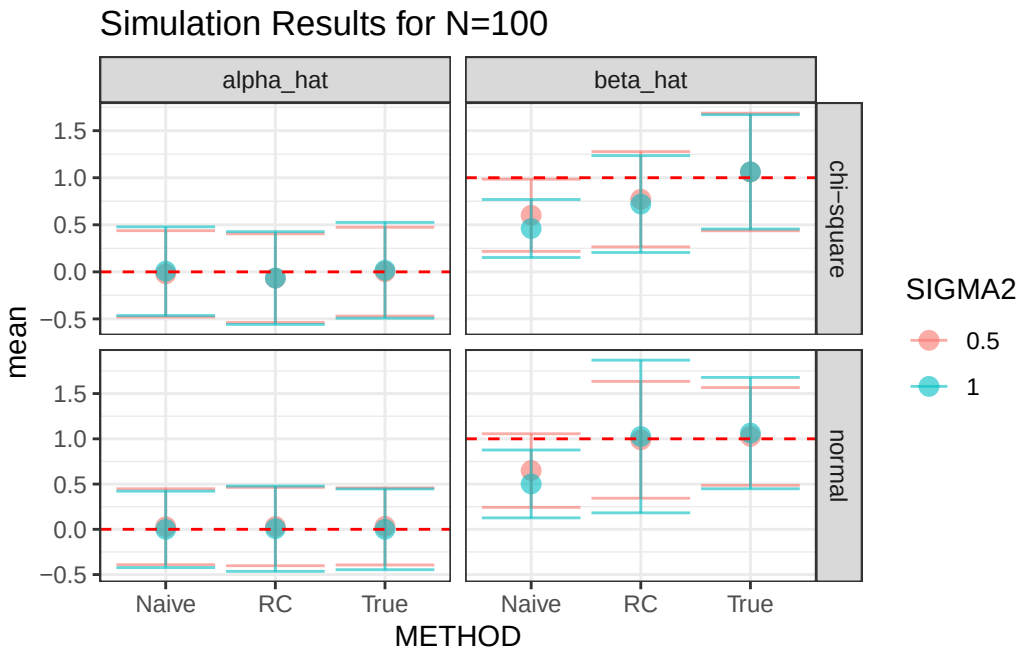

Figure 1: Simulation results for n=100

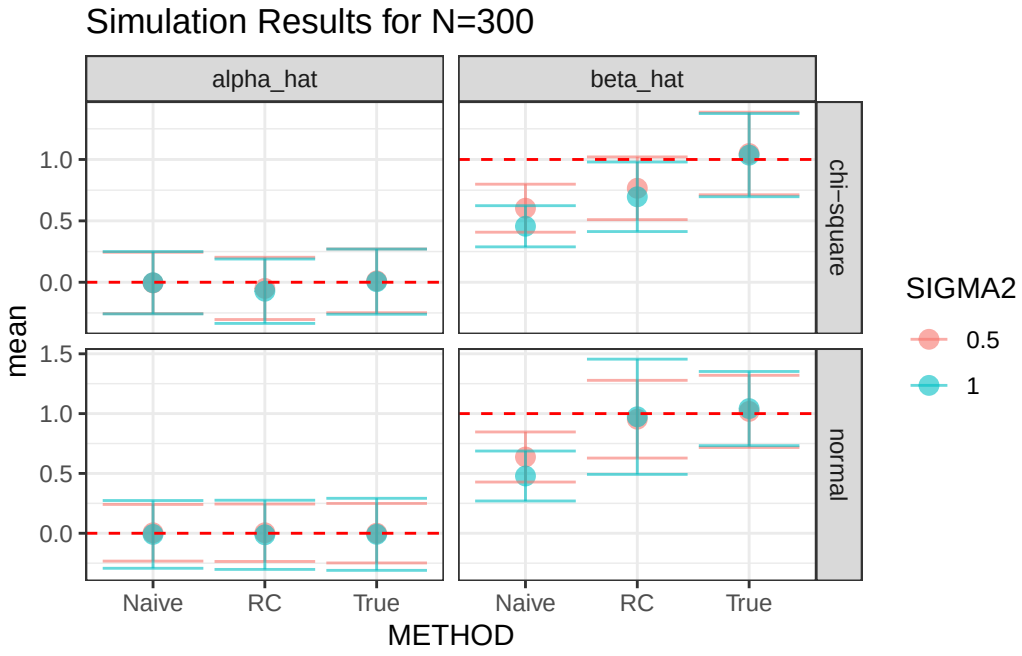

Figure 2: Simulation results for n=300

Question 2

We start off with observing $\hat{\alpha}$. Across all sample sizes, distribution scenarios for X and σ_ϵ^2 values, all point estimates and standard errors are similar with no significant change between them. As expected, standard errors are smaller across the board when $n = 300$ compared to when $n = 100$. For $\sigma_\epsilon^2 = 1$, the standard errors are a bit larger but not by much. Interestingly, in both the n=100 and n=300 cases, the RC estimator is slightly underestimated compared to the true and naive estimators under the assumption of a χ^2 distributed X. This slight bias disappears when X is normal.

Moving on to $\hat{\beta}$, when X is normal, the RC and true estimators are unbiased. However the standard error for the RC estimator is greater than that of the true, and is exacerbated as σ_ϵ^2 increases. The naive estimator underestimated β considerably (mean of 0.649 for a true value of 1 when $\sigma_\epsilon^2 = 0.5$), and this bias also worsened with increasing σ_ϵ^2.

When X is χ^2 distributed, the story changes a little bit. As expected, the naive estimator underestimates β again, but so does the RC. However, the RC is still closer to the truth than the naive, evidence of its usefulness even when it is not exact. The bias in the RC increases with increasing σ_ϵ^2.

Question 3

For the SIMEX procedure, we set $B = 1000$ and $\lambda \in \{0, 0.1, 0.2, ..., 2\}$. We use the working extrapolant function (quadratic in λ) from pg. 6 of the SIMEX lecture notes.

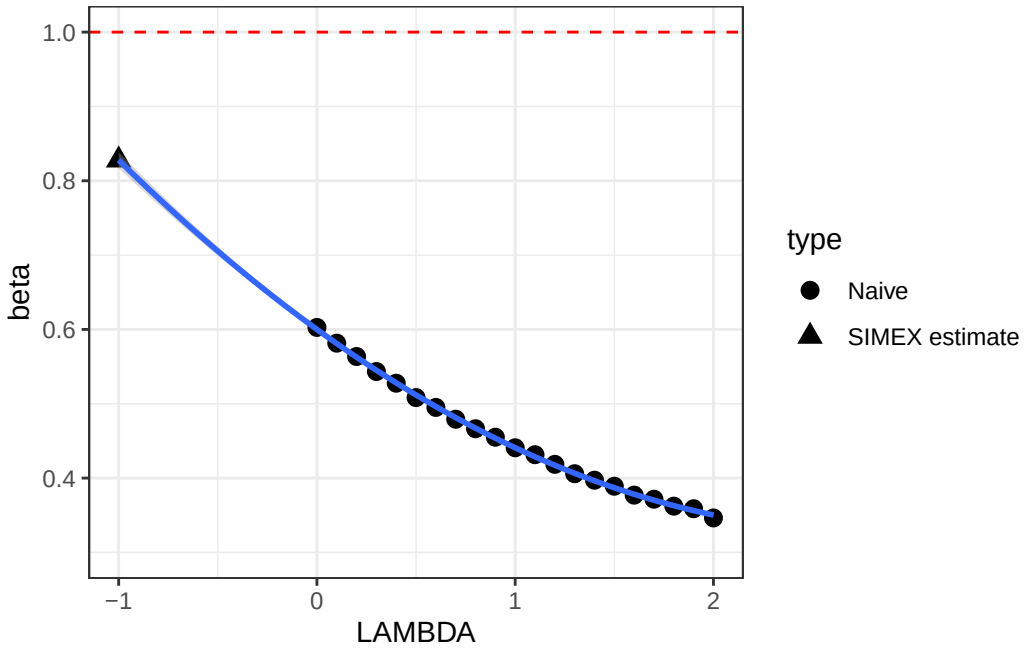

Figure 3: SIMEX plot for beta

The SIMEX estimator for α is -0.0455 and for β is 0.827.

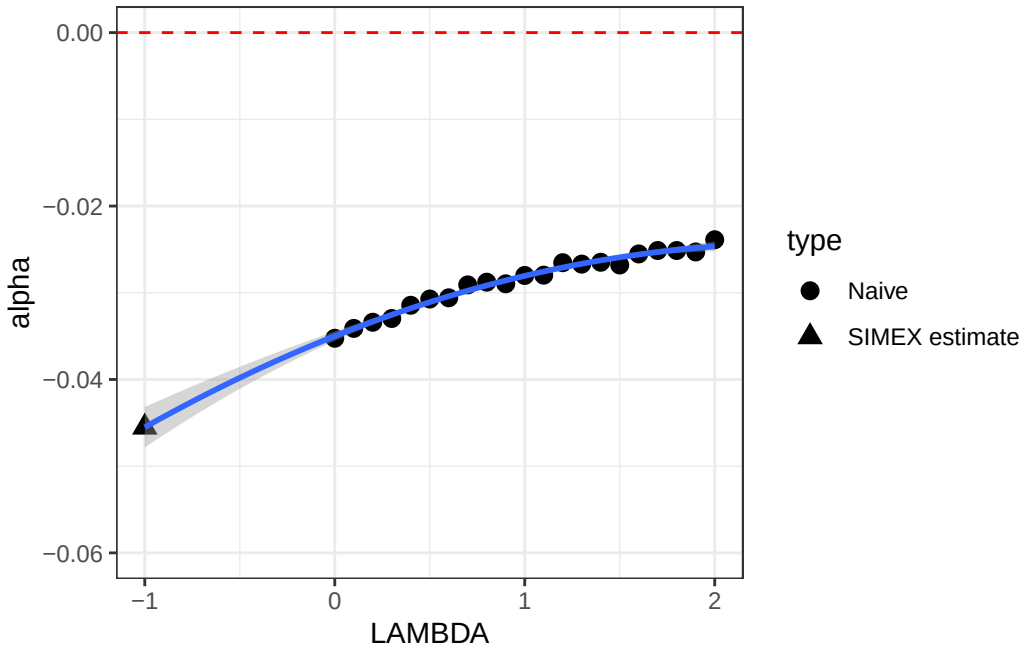

Figure 4: SIMEX plot for alpha

Code Appendix

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

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
```

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

  return(MLE(Y, Xhat))
}

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
  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")
alpha_hat <- 0; beta_hat <- 0
results <- data.table::CJ(SIM, N, SIGMA2, XDIST, METHOD, alpha_hat, beta_hat)

alpha <- 0; beta <- 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)
          }

          results[SIM == i &
                    N == n &
                    SIGMA2 == sigma2 &
                    XDIST == xdist &
                    METHOD == method,
                  `:=`(alpha_hat = est[1],
        }
      }
    }
  }
}
results <- fread("HW3_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)]
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")
set.seed(1)
# Simulate dataset
data <- sim_data(N=300, 0, 1, 0.5, "normal")
Y <- data$Y
W <- data$W

# SIMEX parameters
B      <- 1000
LAMBDA <- seq(0, 2, .1)

results <- CJ(SIM=1:B, LAMBDA=lambda, alpha=0, beta=0)

# Loop through each SIMEX index
for(lambda in LAMBDA){
  message(lambda)
  for(i in 1:B){
    U <- rnorm(300, mean=0, sd=sqrt(.5))
    W_lambda <- W + sqrt(lambda) * U
    fit <- MLE(Y, W_lambda)
    results[SIM==i & LAMBDA == lambda, `:=`(alpha=fit[1], beta=fit[2])]
  }
}

fwrite(results, "HW3_SIMEX_results.csv")
results <- fread("HW3_SIMEX_results.csv")

summary <- results[, .(alpha = mean(alpha), beta = mean(beta)), .(LAMBDA)]
summary[, LAMBDA2 := LAMBDA^2]

extr.beta <- lm(beta ~ LAMBDA + LAMBDA2, data=summary)
```

```
extr.alpha <- lm(alpha ~ LAMBDA + LAMBDA2, data=summary)

SIMEX_beta <- predict(extr.beta, data.frame(LAMBDA=-1, LAMBDA2=1))
SIMEX_alpha <- predict(extr.alpha, data.frame(LAMBDA=-1, LAMBDA2=1))

summary <- rbind(summary, data.table(LAMBDA=-1, LAMBDA2=1, alpha=SIMEX_alpha,
                                     beta=SIMEX_beta), fill=TRUE)
summary[, type := ifelse(LAMBDA==-1, "SIMEX estimate", "Naive")]
ggplot(summary, aes(LAMBDA, beta)) +
  geom_point(aes(shape=type), size=3)+
  coord_cartesian(xlim=c(-1, 2), ylim=c(.3,1)) +
  geom_hline(yintercept=0, col="red", linetype = "dashed") +
  theme_bw() +
  geom_smooth(data=summary[-22], method="lm", formula = y ~ x + I(x^2),
              fullrange=TRUE) +
  geom_hline(yintercept = 1, linetype="dashed", col="red")
ggplot(summary, aes(LAMBDA, alpha)) +
  geom_point(aes(shape=type), size=3)+
  coord_cartesian(xlim=c(-1, 2), ylim=c(-.06,0)) +
  geom_hline(yintercept=0, col="red", linetype = "dashed") +
  theme_bw() +
  geom_smooth(data=summary[-22], method="lm", formula = y ~ x + I(x^2), fullrange=TRUE)
```