BCest_simulation

Simulation Process

We simulate covariates X_1 independently from normal distribution with known mean and variance to generate the covariate matrix X,

$$X = \begin{bmatrix} 1 & x_{11} \\ 1 & x_{12} \\ \vdots & \\ 1 & x_{1n} \end{bmatrix}$$

To generate the response matrix Y, we use some true underlying model $Y = g(X) + \epsilon$. For example, we set $g(X)$ to be a linear regression model $g(X) = X\beta$, where X is the simulated covariate matrix, β is given, and ϵ is also simulated.

```
library(dplyr)
library(caret)
library(parallel)
```

We want to split our covariates and responses randomly into training, testing, and validation data sets in proportion 6:2:2, so we first sample data randomly into 10 sub-groups. In the `creat_testIndex(sample_size)` function, it takes the sample size as the parameter, and randomly samples sample size into 10 sub-groups.

```
create_testIndex <- function(sample_size){
  Index_list <- list()
  group_idx <- sample(cut(seq(1,sample_size),breaks=10,label=FALSE))
  for (i in 1:10){
    Index_list[[i]] <- which(group_idx == i)
  }

  return(Index_list)
}
```

In the `simulation(sample_size, num_of_cov, beta_list, Index_list, use_seed=12345)` function, it takes in 5 parameters: sample size, number of covariates, a list of given betas, an index list created from `creat_testIndex(sample_size)`, and a seed number for simulation purpose.

```
simulation <- function(sample_size, num_of_cov, beta_list, Index_list, use_seed=12345
){

  set.seed(use_seed)
```

We create a list of covariates `covariate_list`, where the first element in the list is a vector of 1s where the length is the same as the sample size. The second element of `covariate_list` is X_1, where we simulate from $\mathcal{N}(0, 1)$ (assuming we have standardized covariates) and the length of X_1 is equal to the sample size.

```
  covariate_list <- list()
  covariate_list[[1]] <- rep(1, sample_size)
  for (i in c(2:(num_of_cov+1))){
    covariate_list[[i]] <- rnorm(sample_size, mean = 0, sd = 1) ## Xs
  }
```

We partition the simulated covariate matrix X into testing, training, validation in propotion 6:2:2 using the `Index_list` created from `creat_testIndex(sample_size)`.

```
  covariate_training <- list()
  covariate_testing <- list()
  covariate_validation <- list()
  for (i in 1:num_of_cov){
    covariate_training[[i]] <- covariate_list[[i+1]][unlist(Index_list[1:6])]
    covariate_testing[[i]] <- covariate_list[[i+1]][unlist(Index_list[7:8])]
    covariate_validation[[i]] <- covariate_list[[i+1]][unlist(Index_list[9:10])]
  }
```

We simulate a vector of errors from $\mathcal{N}(0, 1)$ with the length equal to the sample size.

```
  e <- rnorm(sample_size, mean = 0, sd = 1)
```

We simulate Y from the the underlying true model $Y = g(X) + \epsilon$ and $g(X) = X\beta$. In this case we only have one covariate X_1, so Y comes from $Y = \beta_0 + X_1\beta_1 + \epsilon$.

```
  y <- Map('*', beta_list, covariate_list) %>% Reduce('+',.) + e

  ## if Y has a quatratic term
  #y <- Map('*', beta_list[1:(length(beta_list) - 1)], covariate_list[1:(length(beta_
list) - 1)]) %>%
  #     Reduce('+',.) + beta_list[[length(beta_list)]] * covariate_list[[length(beta_
list)]]^2 + e
```

We then partition the simulated true Y values into testing, training, validation sets in propotion 6:2:2 using the `Index_list` created from `creat_testIndex(sample_size)`.

```
  ## partition simulated true Y into testing, training, validation in propotion 6:2:2
  y_training <- y[unlist(Index_list[1:6])]
  y_testing <- y[unlist(Index_list[7:8])]
  y_validation <- y[unlist(Index_list[9:10])]
```

In the training data set, we can fit any model to learn the relationship between the response and covariates. For example, we fit a random forest machine learning model in the training data set, denoted as $f(x)$.

```
  ## fit a machine learning model (random forest -- 'method = ranger') using traing d
ata Y ~ X, and get predictions for testing data sets
  training_data <- data.frame(Y = y_training, X = matrix(unlist(covariate_training),
nrow = num_of_cov, byrow = TRUE) %>% t())

  model <- train(Y ~., data = training_data,
             method = 'rf',
             importance = TRUE)
```

Then we can obtain the predicted Y value in the testing data set using $f(x)$ by $\hat{Y}_{test} = f(X_{test})$, and the predicted Y value in the validation data set by $\hat{Y}_{val} = f(X_{val})$.

```
  ## calculate expected y in the testing dataset using model (random forrest)
  testing_covariates <- data.frame(X = matrix(unlist(covariate_testing), nrow = num_o
f_cov, by = TRUE) %>% t())
  y_hat_testing <- predict(model, testing_covariates)

  validation_covariates <- data.frame(X = matrix(unlist(covariate_validation), nrow =
num_of_cov, by = TRUE) %>% t())
  y_hat_validation <- predict(model, validation_covariates)
```

[PLOTS]

```
## plot y_testing and y_hat_testing in 2D plot (explore joint distribution of y_testi
ng and y_hat_testing)
## joint distribution of y_testing and y_hat_testing is approximately multivariate(bi
variate) normal
colors <- densCols(y_testing, y_hat_testing, bandwidth = 2)
data <- data.frame(y_testing, y_hat_testing)
p_yhat_y <- ggplot(data, aes(x = y_testing, y = y_hat_testing)) +
            geom_point(color = colors, size = 2) +
            geom_abline(linetype="dashed",color="red",size=1) +
            labs(y="Predicted Y in testing set (LM)", x = "True Y in testing set")
print(p_yhat_y)
```

The true underlying model $Y = g(X) + \epsilon$ is unknown to us, but we want to fit a linear regression model, $Y = X\beta + \epsilon^*$, on true value Y and covariate matrix X. We can then obtain the OLS estimator $\hat{\beta}$ as,

$$\hat{\beta} = (X^T X)^{-1} X^T Y. \tag{1}$$

However, we are unable to calculate the above unbiased OLS estimator $\hat{\beta}_{val}$ in the validation data set, because the true values Y_{val} in the validation data set are unrevealed to us. Thus, we use the known predicted values $\widehat{Y}_{val}$ instead to write the regression model as $\widehat{Y}_{val} = X_{val}\beta_{est} + \epsilon^*$. We can then obtain the OLS estimators $\hat{\beta}^*$ as,

$$\hat{\beta}_{est} = (X_{val}^T X_{val})^{-1} X_{val}^T \widehat{Y}_{val}. \tag{2}$$

```
  X_val_matrix <- cbind(rep(1,nrow(validation_covariates)),validation_covariates) %>%
as.matrix()
  X_test_matrix <- cbind(rep(1,nrow(testing_covariates)),testing_covariates) %>% as.m
atrix()

  beta_hat_est <- solve(t(X_val_matrix) %*% X_val_matrix) %*% t(X_val_matrix) %*% y_h
at_validation
```

We assume that the predicted value $\widehat{Y}$, where $\widehat{Y} = f(X)$, follows a normal distribution centered around the true value Y. So, we propose that the mean of $\widehat{Y}$ is a linear function of Y. We write out the distribution of $\widehat{Y}$ as,

$$\widehat{Y} \sim \mathcal{N}(\gamma_0 + \gamma_1 Y, \sigma^2_{\widehat{Y}}). \tag{3}$$

Here, we also propose that the mean of true value Y follows a linear regression model $X\beta$, and variance of true value Y is denoted as σ^2_Y. Then we attempt to find the unknown parameter values of γ_0, γ_1, and $\sigma^2_{\widehat{Y}}$ that maximize the likelihood function of equation (3). The MLE of parameters can be written as,

$$\hat{\gamma}_0 = \overline{\widehat{Y}} - \hat{\gamma}_1 \overline{Y} \tag{4}$$

$$\hat{\gamma}_1 = \frac{\sum_{i=1}^n (Y_i - \overline{Y})(\widehat{Y}_i - \overline{\widehat{Y}})}{\sum_{i=1}^n (Y_i - \overline{Y})^2} \tag{5}$$

$$\hat{\sigma}^2_{\widehat{Y}} = \frac{\sum_{i=1}^n (\widehat{Y}_i - \hat{\gamma}_0 - \hat{\gamma}_1 Y_i)^2}{n} \tag{6}$$

Since parameters γ_0, γ_1, β are unknown, we approximate them by using the MLE estimators of $\hat{\gamma}_{0test}$, $\hat{\gamma}_{1test}$, and the OLS estimator of $\hat{\beta}_{test}$ calculated in the testing data set,

$$\hat{\gamma}_{1test} = \frac{\sum_{i=1}^{n}(Y_{test_i} - \overline{Y}_{test})(\widehat{Y}_{test_i} - \overline{\widehat{Y}}_{test})}{\sum_{i=1}^{n}(Y_{test_i} - \overline{Y}_{test})^2} \tag{7}$$

```
  Y_test_mu <- mean(y_testing)
  Y_hat_test_mu <- mean(y_hat_testing)

  gamma1_mle <- sum((y_testing - Y_test_mu) * (y_hat_testing - Y_hat_test_mu)) /
                sum((y_testing - Y_test_mu)^2)
```

$$\hat{\gamma}_{0test} = \overline{\widehat{Y}}_{test} - \hat{\gamma}_1 \overline{Y}_{test} \tag{8}$$

```
  gamma0_mle <- Y_hat_test_mu - gamma1_mle * Y_test_mu
```

$$\hat{\beta}_{test} = (X_{test}^T X_{test})^{-1} X_{test}^T Y_{test} \tag{9}$$

```
  beta_hat_test <- solve(t(X_test_matrix) %*% X_test_matrix) %*% t(X_test_matrix) %*%
y_testing
```

[PLOTS]

```
  ## plot y_hat_validation and est_hat_validation in 2D plot
  est_hat_validation <- gamma0_mle + gamma1_mle * y_validation #X_val_matrix %*% beta
_hat_test
  colors <- densCols(y_hat_validation, est_hat_validation, bandwidth = 2)
  data <- data.frame(y_hat_validation, est_hat_validation)
  p_yhat_y <- ggplot(data, aes(x = y_hat_validation, y = est_hat_validation)) +
              geom_point(color = colors, size = 2) +
              geom_abline(linetype="dashed",color="red",size=1) +
              labs(y="estimated Y_hat in validation set (MLE)", x = "Y_hat in validat
ion set")
  print(p_yhat_y)
```

The bias of the OLS estimator $\hat{\beta}$ from equation (1) relative to model assumption β is 0,

$$Bias_{\beta}(\hat{\beta}) = \mathbf{E}_{Y|X}(\hat{\beta}) - \beta = 0 \tag{10}$$

In this case, the bias of $\hat{\beta}_{est}$ from equation (2) relative to β must not equal to 0 due to errors introduced by the prediction model $\widehat{Y}_{val} = f(X_{val})$. Therefore, if we estimate such bias using the existing data in the testing data set, and correct this bias in $\hat{\beta}_{est}$ to better estimate the model assumption β in the validation data set. Let us write the bias of $\hat{\beta}_{est}$ relative to β,

$$Bias_\beta(\hat{\beta}_{est}) = \mathbf{E}_{\widehat{Y}|X}(\hat{\beta}_{est}) - \beta \tag{11}$$

In this case, we further use $\widehat{Y}_{test}$ and X covariate matrix in the testing data set to estimate $\mathbf{E}_{\widehat{Y}|X}$, and Y_{test} and X covariate matrix in the testing data set to estimate β. Thus, we estimate the bias of $\hat{\beta}_{est}$ relative to β to be

$$Bias_\beta(\hat{\beta}_{est}) \approx (X_{test}^T X_{test})^{-1} X_{test}^T \widehat{Y}_{test} - \hat{\beta}_{test} \tag{12}$$

```
#Bias1 <- (solve(t(X_val_matrix) %*% X_val_matrix) %*% t(X_val_matrix)) %*%
#        (gamma0_mle + gamma1_mle * X_val_matrix %*% beta_hat_test) - beta_hat_test
beta_hat_test_yhat <- solve(t(X_test_matrix) %*% X_test_matrix) %*% t(X_test_matrix
) %*% y_hat_testing
Bias2 <-  beta_hat_test_yhat -  beta_hat_test
```

Now we propose a new bias-corrected estimator $\hat{\beta}_{BCest}$ to better estimate the true β using predicted values $\widehat{Y}_{val}$, where $\hat{\beta}_{BCest}$ is an estimator that corrects the bias of $\hat{\beta}_{est}$ relative to β in equation (12).

$$\hat{\beta}_{BCest} = \hat{\beta}_{est} - Bias_\beta(\hat{\beta}_{est}) \tag{13}$$

```
beta_hat_BCest <- beta_hat_est - Bias2
```

Let us calculate the mean of the bias-corrected estimator $\hat{\beta}_{BCest}$ as,

$$\begin{aligned}
&\mathbf{E}_{\widehat{Y}|X}(\hat{\beta}_{BCest}) \\
&= \mathbf{E}_{\widehat{Y}|X}\left(\hat{\beta}_{est} - Bias_\beta(\hat{\beta}_{est})\right) \\
&= \mathbf{E}_{\widehat{Y}|X}(\hat{\beta}_{est}) - \mathbf{E}_{\widehat{Y}|X}(\mathbf{E}_{\widehat{Y}|X}(\hat{\beta}_{est} - \beta)) \\
&\approx \hat{\beta}_{test}
\end{aligned} \tag{14}$$

Now we want to estimate the variance of the bias-corrected estimator $\hat{\beta}_{BCest}$. In order to do so, let us calculate the marginal variance of $\widehat{Y}$ first. Recall from equation (3), we propose that $\widehat{Y} \sim \mathcal{N}(\gamma_0 + \gamma_1 Y, \sigma^2_{\widehat{Y}})$. Then, we use the MLE estimators from equation (4-6) to calculate the mariginal variance of $\widehat{Y}$,

$$\begin{aligned}
\mathbf{Var}_{\widehat{Y}}(\widehat{Y}) &= \mathbf{E}_Y\left(\mathbf{Var}_{\widehat{Y}|Y}(\widehat{Y} \mid Y)\right) + \mathbf{Var}_Y\left(\mathbf{E}_{\widehat{Y}|Y}(\widehat{Y} \mid Y)\right) \\
&= \sigma^2_{\widehat{Y}} + \mathbf{Var}_Y(\gamma_0 + \gamma_1 Y) \\
&= \sigma^2_{\widehat{Y}} + \gamma_1^2 \mathbf{Var}_Y(Y) \\
&= \sigma^2_{\widehat{Y}} + \gamma_1^2 \sigma^2_Y
\end{aligned} \tag{15}$$

Then we approximate parameters γ_1 using the MLE estimators of $\hat{\gamma}_{1test}$ from equation (7).

Let us calculate the mean of the bias-corrected estimator $\hat{\beta}_{BCest}$ as,

$$\begin{aligned}
\mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{BCest}) &= \mathbf{Var}_{\widehat{Y}|X}\Big(\hat{\beta}_{est} - Bias_{\beta}(\hat{\beta}_{est})\Big) \\
&= \mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{est}) \\
&= \mathbf{Var}_{\widehat{Y}|X}\Big((X_{val}^T X_{val})^{-1} X_{val}^T \widehat{Y}_{val}\Big) \\
&= (X_{val}^T X_{val})^{-1} X_{val}^T \cdot \mathbf{Var}_{\widehat{Y}|X}(\widehat{Y}_{val}) \cdot X_{val}(X_{val}^T X_{val})^{-1} \\
&= (X_{val}^T X_{val})^{-1} X_{val}^T \cdot (\sigma_{\widehat{Y}}^2 + \gamma_1^2 \sigma_Y^2) \cdot X_{val}(X_{val}^T X_{val})^{-1} \\
&\approx (X_{val}^T X_{val})^{-1} (\hat{\sigma}^2_{\widehat{Y}_{test}} + \hat{\gamma}_{1test}^2 \cdot \sigma_{Y_{test}}^2)
\end{aligned} \tag{16}$$

```
  sigma_Y_hat_test <- (sum((y_hat_testing - gamma0_mle - gamma1_mle * y_testing)^2) /
(length(y_testing)-1))
  sigma_Y_test <- sum((y_testing - X_test_matrix %*% beta_hat_test)^2) / (length(y_te
sting)-1)
  var_BCest <- solve(t(X_val_matrix) %*% X_val_matrix) * (sigma_Y_hat_test + gamma1_m
le^2 * sigma_Y_test)
```

We want to calculate t statistics of $\hat{\beta}$ estimators to test null hypothesis $H_0 : \beta = \beta_{true}$. First, we need to calcuate β_{true} using $g(X)$ in the validation data set. In this simulation process, we set $g(X)$ to be a linear model $g(X) = X\beta$. Recall that $Y = g(X) + \epsilon$. So,

$$\begin{aligned}
\beta_{true} &= \mathbf{E}\Big[(X_{val}^T X_{val})^{-1} X_{val}^T Y | g(X_{val})\Big] \\
&= (X_{val}^T X_{val})^{-1} X_{val}^T \mathbf{E}\Big[Y | g(X_{val})\Big] \\
&= (X_{val}^T X_{val})^{-1} X_{val}^T g(X_{val})
\end{aligned} \tag{17}$$

```
  g_xval <- Map('*', beta_list, c(rep(1,length(covariate_validation[[1]])) %>% list()
,covariate_validation)) %>%             Reduce('+',.)
  #g_xval <- beta_list[[1]] + beta_list[[2]] * covariate_validation[[1]]^2
  beta_true <- solve(t(X_val_matrix) %*% X_val_matrix) %*% t(X_val_matrix) %*% g_xval
```

Null Hypothesis: $H_0 : \beta = \beta_{true}$ a. calcuate t statistics for $t_{\hat{\beta}_{est}}$ using standard error from only fitting the linear model:

$$
\begin{aligned}
t_{1\hat{\beta}_{est}(se_{\hat{\beta}_{est}})} &= \frac{\hat{\beta}_{est} - \beta_{true}}{se_{\hat{\beta}_{est}}} \\
&= \frac{\hat{\beta}_{est} - \beta_{true}}{\sqrt{(X_{val}^T X_{val})^{-1} (\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})^T (\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})/df}}
\end{aligned} \tag{18}
$$

```
## predicted Y_hat_val with lm standard error
  t_b_est_lmse <- (beta_hat_est - beta_true) / sqrt(diag(solve(t(X_val_matrix) %*% X_
val_matrix) * sum((y_hat_validation - X_val_matrix %*% beta_hat_est)^2)/(length(y_val
idation - 1))))
```

b. calcuate t statistics for $t_{\hat{\beta}_{est}}$ using bias-corrected standard error of $\hat{\beta}_{BCest}$:

$$
\begin{aligned}
t_{1\hat{\beta}_{est}(se_{\hat{\beta}_{BCest}})} &= \frac{\hat{\beta}_{est} - \beta_{true}}{se_{\hat{\beta}_{BCest}}} \\
&= \frac{\hat{\beta}_{est} - \beta_{true}}{\sqrt{\mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{BCest})}}
\end{aligned} \tag{19}
$$

```
## predicted Y_hat_val with bias corrected standard error
  t_b_est_BCse <- (beta_hat_est - beta_true) / sqrt(diag(var_BCest))
```

c. calcuate t statistics for $t_{\hat{\beta}_{BCest}}$ using standard error of only fitting the linear model:

$$
\begin{aligned}
t_{1\hat{\beta}_{BCest}(se_{\hat{\beta}_{est}})} &= \frac{\hat{\beta}_{BCest} - \beta_{true}}{se_{\hat{\beta}_{est}}} \\
&= \frac{\hat{\beta}_{BCest} - \beta_{true}}{\sqrt{(X_{val}^T X_{val})^{-1} (\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})^T (\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})/df}}
\end{aligned} \tag{20}
$$

```
 ## bias corrected beta with lm standard error
  t_b_BCest_lmse <- (beta_hat_est - Bias2 - beta_true) / sqrt(diag(solve(t(X_val_matr
ix) %*% X_val_matrix) * sum((y_hat_validation - X_val_matrix %*% beta_hat_est)^2)/(le
ngth(y_validation - 1))))
```

d. calcuate t statistics for $t_{\hat{\beta}_{BCest}}$ using bias-corrected standard error of $\hat{\beta}_{BCest}$

$$
\begin{aligned}
t_{1\hat{\beta}_{BCest}(se_{\hat{\beta}_{BCest}})} &= \frac{\hat{\beta}_{BCest} - \beta_{true}}{se_{\hat{\beta}_{BCest}}} \\
&= \frac{\hat{\beta}_{BCest} - \beta_{true}}{\sqrt{\mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{BCest})}}
\end{aligned} \tag{21}
$$

```
## bias corrected beta with bias corrected standard error
t_b_BCest_BCse <- (beta_hat_est - Bias2 - beta_true) / sqrt(diag(var_BCest))
```

e. calculate t statistics for $t_{\hat{\beta}_{val}}$ using true Y_{val} values in the validation data set. Note that we do not know Y_{val} in real life. We can only calculate this statistics using simulation data with known data generation mechanism.

$$
\begin{aligned}
t_{1\hat{\beta}_{val}(se_{\hat{\beta}_{val}})} &= \frac{\hat{\beta}_{val} - \beta_{true}}{se_{\hat{\beta}_{val}}} \\
&= \frac{\hat{\beta}_{val} - \beta_{true}}{\sqrt{(X_{val}^T X_{val})^{-1}(Y - X_{val}\hat{\beta}_{val})^T(Y - X_{val}\hat{\beta}_{val})/df}}
\end{aligned} \tag{22}
$$

```
  beta_val <- solve(t(X_val_matrix) %*% X_val_matrix) %*% t(X_val_matrix) %*% y_valid
ation
  ## lm using known Y_val
  t_b_val <- (beta_val - beta_true) / sqrt(diag(solve(t(X_val_matrix) %*% X_val_matri
x) * sum((y_validation - X_val_matrix %*% beta_val)^2)/(length(y_validation - 1))))
```

We also want to calculate t statistics of $\hat{\beta}$ estimators to test null hypothesis $H_0 : \beta = 0$.

Null Hypothesis: $H_0 : \beta = 0$ a. calcuate t statistics for $t_{\hat{\beta}_{est}}$ using standard error from only fitting the linear model:

$$
\begin{aligned}
t_{2\hat{\beta}_{est}(se_{\hat{\beta}_{est}})} &= \frac{\hat{\beta}_{est} - 0}{se_{\hat{\beta}_{est}}} \\
&= \frac{\hat{\beta}_{est}}{\sqrt{(X_{val}^T X_{val})^{-1}(\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})^T(\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})/df}}
\end{aligned} \tag{23}
$$

```
  t_0_est_lmse <- beta_hat_est / sqrt(diag(solve(t(X_val_matrix) %*% X_val_matrix) *
sum((y_hat_validation - X_val_matrix %*% beta_hat_est)^2)/(length(y_validation - 1)))
)
```

b. calcuate t statistics for $t_{\hat{\beta}_{est}}$ using bias-corrected standard error of $\hat{\beta}_{BCest}$:

$$\begin{aligned} t_{2\hat{\beta}_{est}(se_{\hat{\beta}_{BCest}})} &= \frac{\hat{\beta}_{est} - 0}{se_{\hat{\beta}_{BCest}}} \\ &= \frac{\hat{\beta}_{est}}{\sqrt{\mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{BCest})}} \end{aligned} \tag{24}$$

```
t_0_est_BCse <- beta_hat_est / sqrt(diag(var_BCest))
```

c. calcuate t statistics for $t_{\hat{\beta}_{BCest}}$ using standard error of only fitting the linear model:

$$\begin{aligned} t_{2\hat{\beta}_{BCest}(se_{\hat{\beta}_{est}})} &= \frac{\hat{\beta}_{BCest} - 0}{se_{\hat{\beta}_{est}}} \\ &= \frac{\hat{\beta}_{BCest}}{\sqrt{(X_{val}^T X_{val})^{-1}(\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})^T(\widehat{Y}_{val} - X_{val}\hat{\beta}_{est})/df}} \end{aligned} \tag{25}$$

```
t_0_BCest_lmse <- (beta_hat_est - Bias2) / sqrt(diag(solve(t(X_val_matrix) %*% X_val_
matrix) * sum((y_hat_validation - X_val_matrix %*% beta_hat_est)^2)/(length(y_validat
ion - 1))))
```

d. calcuate t statistics for $t_{\hat{\beta}_{BCest}}$ using bias-corrected standard error of $\hat{\beta}_{BCest}$

$$\begin{aligned} t_{2\hat{\beta}_{BCest}(se_{\hat{\beta}_{BCest}})} &= \frac{\hat{\beta}_{BCest} - 0}{se_{\hat{\beta}_{BCest}}} \\ &= \frac{\hat{\beta}_{BCest}}{\sqrt{\mathbf{Var}_{\widehat{Y}|X}(\hat{\beta}_{BCest})}} \end{aligned} \tag{26}$$

```
t_0_BCest_BCse <- (beta_hat_est - Bias2) / sqrt(diag(var_BCest))
```

e. calculate t statistics for $t_{\hat{\beta}_{val}}$ using true Y_{val} values in the validation data set. Note that we do not know Y_{val} in real life. We can only calculate this statistics using simulation data with known data generation mechanism.

$$
\begin{aligned}
t_{2\hat{\beta}_{val}(se_{\hat{\beta}_{val}})} &= \frac{\hat{\beta}_{val} - 0}{se_{\hat{\beta}_{val}}} \\
&= \frac{\hat{\beta}_{val}}{\sqrt{(X_{val}^T X_{val})^{-1}(Y - X_{val}\hat{\beta}_{val})^T(Y - X_{val}\hat{\beta}_{val})/df}}
\end{aligned} \tag{27}
$$

```
  #### t statistics for testing beta = 0 (in the summary table when fitting a linear model)
  t_0_val <- beta_val / sqrt(diag(solve(t(X_val_matrix) %*% X_val_matrix) * sum((y_validation - X_val_matrix %*% beta_val)^2)/(length(y_validation - 1))))
```

Let us also compare the root-mean-square error(RMSE) of the bias-corrected improved estimators $\hat{\beta}_{BCest}$ to the RMSE of the estimators $\hat{\beta}_{est}$,

$$
RMSE(\hat{\beta}_{BCest}) = \sqrt{\mathbf{E}\left((\hat{\beta}_{BCest} - \beta_{true})^2\right)} \tag{28}
$$

```
  rmse_beta_hat_est <- sqrt(mean((beta_hat_est[-1] - beta_true[-1])^2))
  rmse_beta_hat_BCest <- sqrt(mean((beta_hat_BCest[-1] - beta_true[-1])^2))
```

```
 # function returns a list of estimators and t statistics
  result <- list(Bias2, beta_hat_est, beta_hat_test, beta_val, beta_hat_test_yhat, beta_true,
                 rmse_beta_hat_est, rmse_beta_hat_BCest,
                 t_b_val, t_b_est_lmse, t_b_BCest_lmse, t_b_est_BCse, t_b_BCest_BCse,
                 t_0_val, t_0_est_lmse, t_0_BCest_lmse, t_0_est_BCse, t_0_BCest_BCse)
  return(result)
}
```

```
# Run the simulation
num_of_cov = 1
sample_size = 2000

for (beta2 in seq(1,40,1)){
  beta_list <- list(1,beta2)

  set.seed(2018)
  Index_list <- create_testIndex(sample_size)

  result <- mclapply(c(1:100), simulation, sample_size = sample_size, num_of_cov = nu
m_of_cov, beta_list = beta_list, Index_list = Index_list,mc.cores = 20)
  save(result, file = paste0("BC_app2_nonlinear_rf_beta",beta2,".rda"))

}

q(save = "no")
```