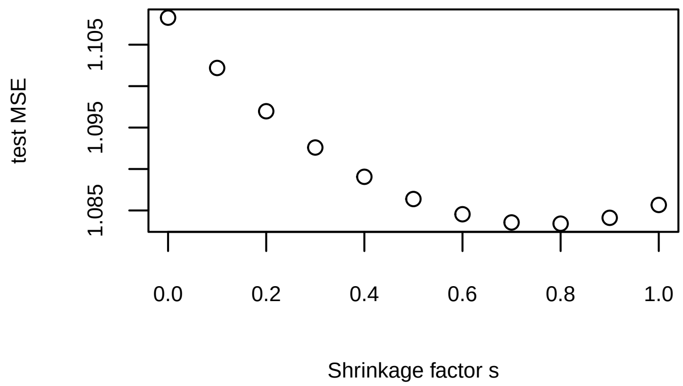
Statistical Learning and Prediction session 1 - Answers to exercises

Exercise 1: Bias can be beneficial

```r
set.seed(1)

## Generate training data
x <- runif(50, min = -3, max = 3)
epsilon <- rnorm(50)
y <- 0.1*x + epsilon
train_dat <- data.frame(x, y)

## Generate test data
x <- runif(1000, min = -3, max = 3)
epsilon <- rnorm(1000)
y <- 0.1*x + epsilon
test_dat <- data.frame(x, y)

## Generate shrinkage values
s <- seq(0, 1, by = .1)

## Get \hat{\beta}
beta_OLS <- coef(lm(y ~ 0 + x, data = train_dat))

## Compute and evaluate predictions using for loop
test_MSE <- numeric(length(s))
for (i in 1:length(s)) {
  y_hat <- test_dat$x * s[i] * beta_OLS
  test_MSE[i] <- mean((y_hat - test_dat$y)^2)
}

## Or with shorter code
test_MSE <- apply(test_dat$x %*% t(beta_OLS*s), 2, \(x) mean((x - test_dat$y)^2))

## Plot result
plot(s, test_MSE, xlab = "Shrinkage factor s", ylab = "test MSE",
     cex.lab = .7, cex.axis = .7)
```

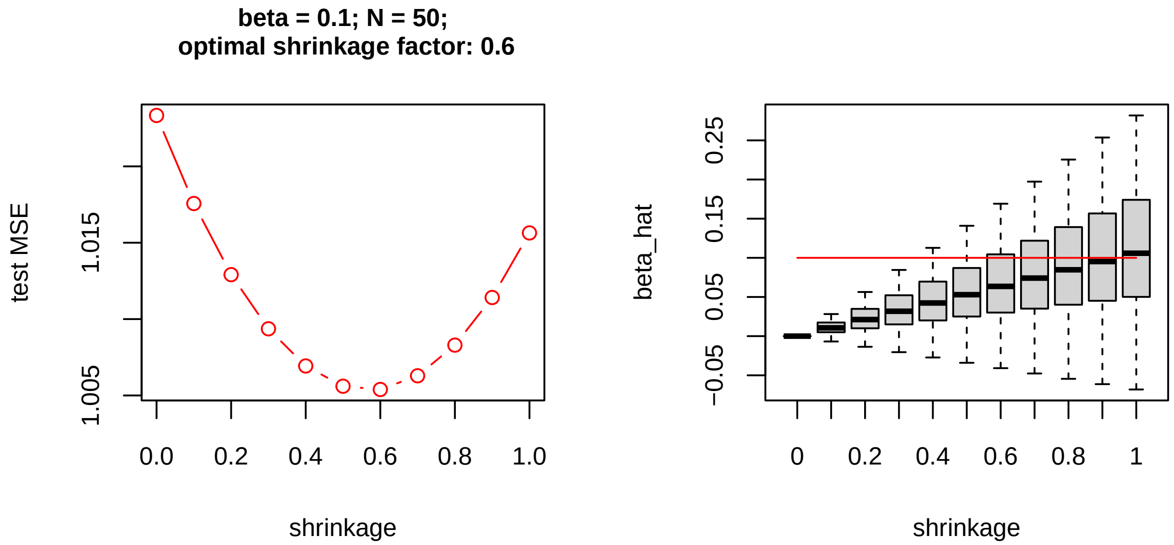


We see the optimal value of the shrinkage factor is less than 1, so the OLS coefficient (which has $s = 1$) is unbiased, but not optimal for prediction.

Not every value of the random seed will yield this result, so we repeat the experiment with 100 replications:

```r
beta <- 0.1
n <- 50
n_reps <- 100
shrinkage <- seq(0, 1, by = 0.1)
mse <- beta_hats <- matrix(0, nrow = n_reps, ncol = length(shrinkage))
colnames(mse) <- colnames(beta_hats) <- shrinkage
set.seed(1234)
for (i in 1:n_reps) {
  ## generate training data
  x <- runif(n, min = -3, max = 3)
  y <- beta*x + rnorm(n)
  ## fit OLS and get parameter estimates
  b_ols <- coef(lm(y ~ 0 + x))
  ## generate test data:
  xtest <- runif(1000, min = -3, max = 3)
  ytest <- beta*xtest + rnorm(1000)
  ## apply shrinkage and obtain predictions
  for (s in 1:length(shrinkage)) {
    ## generate predictions for test observations
    ypred <- xtest * shrinkage[s] * b_ols
    mse[i, s] <- mean((ytest - ypred)^2)
    beta_hats[i, s] <- shrinkage[s] * b_ols
  }
}
par(mfrow = c(1, 2))
min_id <- which(colMeans(mse) == min(colMeans(mse)))
## Plot MSE versus shrinkage
plot(x = shrinkage, y = colMeans(mse), type = 'b',
     col = "red", xlab = "shrinkage", ylab = "test MSE",
     main = paste0("beta = ", beta, "; N = ", n,
                   ";\n optimal shrinkage factor: ",
```

```
                    shrinkage[min_id]), cex.main = .8,
     cex.lab = .8, cex.axis = .8)
## Plot distributions of beta estimates (add line for true value)
boxplot(beta_hats, xlab = "shrinkage", ylab = "beta_hat", outline = FALSE,
        cex.lab = .8, cex.axis = .8)
lines(c(1, 11), c(beta, beta), col = "red")
```

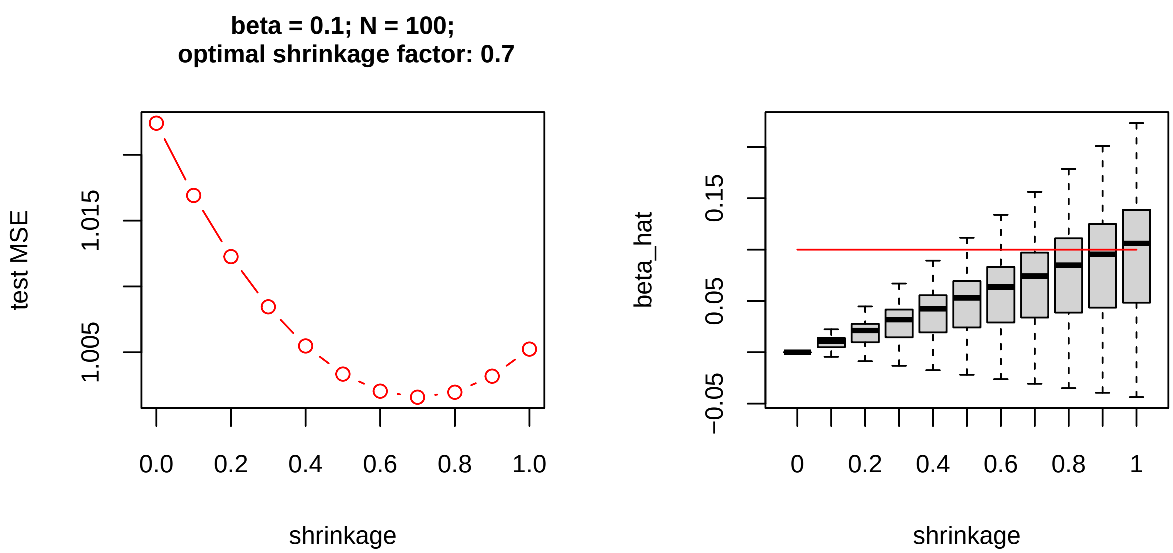


With shrinkage, we see that the estimated coefficient $\hat{\beta}$ becomes biased downwards (the red line in the middle plot indicates the true value β), but the variance of the estimate also gets (much) smaller. A shrinkage factor of 0.6 was optimal. The red line in the right plot indicates the irreducible error.

What if we increase training sample size?

```
n <- 100
set.seed(1234)
for (i in 1:n_reps) {
  ## generate training data
  x <- runif(n, min = -3, max = 3)
  y <- beta*x + rnorm(n)
  ## fit OLS and get parameter estimates
  fit <- lm(y ~ 0 + x)
  b_ols <- coef(fit)
  ## generate test data
  xtest <- runif(1000, min = -3, max = 3)
  ytest <- beta*xtest + rnorm(1000)
  ## apply shrinkage and obtain predictions
  for (s in 1:length(shrinkage)) {
    ## generate predictions for test observations
    ypred <- xtest * shrinkage[s] * b_ols
    mse[i, s] <- mean((ytest - ypred)^2)
    beta_hats[i, s] <- shrinkage[s] * b_ols
  }
}
par(mfrow = c(1, 2))
```

```
min_id <- which(colMeans(mse) == min(colMeans(mse)))
## Plot MSE versus shrinkage
plot(x = shrinkage, y = colMeans(mse), type = 'b',
     col = "red", xlab = "shrinkage", ylab = "test MSE",
     main = paste0("beta = ", beta, "; N = ", n,
                   ";\n optimal shrinkage factor: ",
                   shrinkage[min_id]),
     cex.lab = .8, cex.axis = .8, cex.main = .8)
## Plot distributions of beta estimates (add line for true value)
boxplot(beta_hats, xlab = "shrinkage", ylab = "beta_hat", outline = FALSE,
        cex.lab = .8, cex.axis = .8)
lines(c(1, 11), c(beta, beta), col = "red")
```

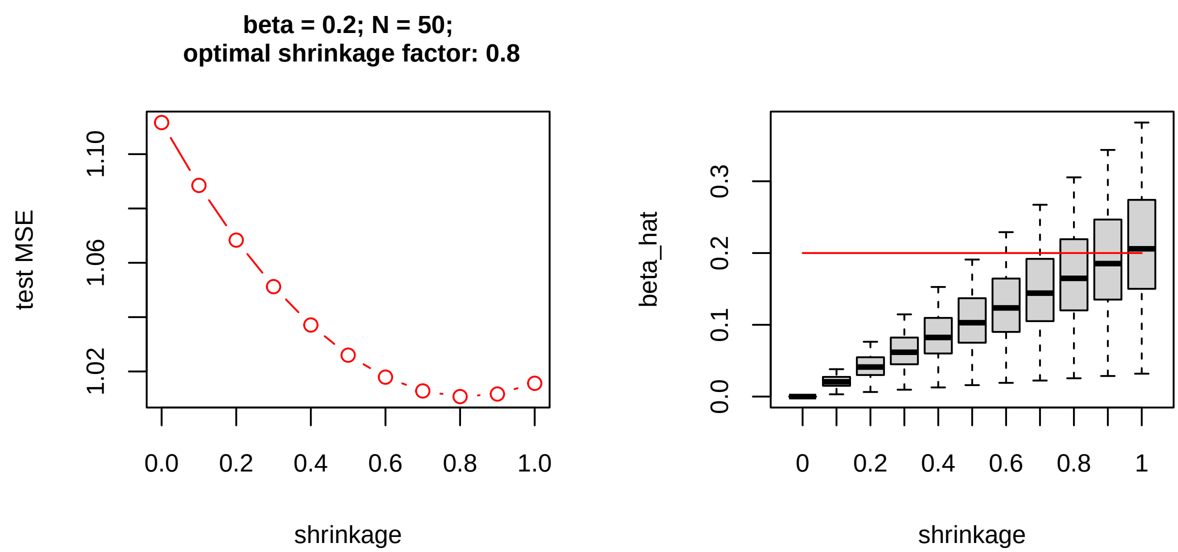


With larger sample size (i.e., more information in the sample), shrinkage is still beneficial. However, with larger sample size, variance of $\hat{\beta}$ is smaller, so we need less shrinkage (bias) to optimize prediction error. Now, a shrinkage factor of 0.7 was optimal.

What if we increase the effect of X?

```
beta <- 0.2
n <- 50
set.seed(1234)
for (i in 1:n_reps) {
  ## generate training data
  x <- runif(n, min = -3, max = 3)
  y <- beta*x + rnorm(n)
  ## fit OLS and get parameter estimates
  fit <- lm(y ~ 0 + x)
  b_ols <- coef(fit)
  ## generate test data
  xtest <- runif(1000, min = -3, max = 3)
  ytest <- beta*xtest + rnorm(1000)
  ## apply shrinkage and obtain predictions
  for (s in 1:length(shrinkage)) {
```

```r
    ## generate predictions for test observations
    ypred <- xtest * shrinkage[s] * b_ols
    mse[i, s] <- mean((ytest - ypred)^2)
    beta_hats[i, s] <- shrinkage[s] * b_ols
  }
}
par(mfrow = c(1, 2))
min_id <- which(colMeans(mse) == min(colMeans(mse)))
## Plot MSE versus shrinkage factor
plot(x = shrinkage, y = colMeans(mse), type = 'b',
     col = "red", xlab = "shrinkage", ylab = "test MSE",
     main = paste0("beta = ", beta, "; N = ", n,
                   ";\n optimal shrinkage factor: ",
                   shrinkage[min_id]),
     cex.main = .8, cex.lab = .8, cex.axis = .8)
## Plot distributions of beta estimates (add line for true value)
boxplot(beta_hats, xlab = "shrinkage", ylab = "beta_hat", outline = FALSE,
        cex.lab = .8, cex.axis = .8)
lines(c(1, 11), c(beta, beta), col = "red")
```



With larger effect size (i.e., larger β, so higher signal-to-noise ratio), shrinkage is still beneficial. Note that the variance of $\hat{\beta}$ does not change as a function of effect size. However, the shrinkage factor c has a stronger effect on larger coefficients, so we need less shrinkage (bias) to optimize prediction error. A shrinkage factor of 0.8 was optimal.

Conclusion: Shrinkage is beneficial for prediction. With higher signal-to-noise ratio and/or larger training sample size (i.e., there is more information in the training data), a lower amount of shrinkage is optimal.