Hw3 Monte Carlo

Jimmy Hickey

10/22/2020

1

f

i

```
library("DynTxRegime")
```

```
## Loading required package: modelObj
```

```
# generate data
logistic_func = function(x){
  return( exp(x) / (1 + exp(x)) )
}
n = 1000
px = 0.5

X1 = rbinom(n, 1 , px)


A1 = rbinom(n, 1, logistic_func(0.3 - 0.5 * X1))


X2 = rnorm(n,
           1 + 0.5*X1 + -0.75 *A1 + 0.25 * X1 * A1,
           sqrt(2))

A2 = rbinom(n, 1, logistic_func(0 + 0.05 * X1 + 0.1 * A1 + -1 *X1 * A1 + -0.1 * X2))

Y = rnorm(n,
          3 + 0 + 0.1 * A1 + 0.5 * X1 * A1 + 0.5 * X2 + 1 * X2^2 + A2*(-1 + 0.75 * X2 + 0.5* A1),
          sqrt(10))
```

ii

```r
QLearn_wrapper = function(data, d2Main, d2Cont, d1Main, d1Cont){
  q2 = qLearn(moMain =d2Main,
              moCont = d2Cont,
              data = data,
              response = data$Y,
              txName = "A2",
              verbose = FALSE)

  q1 = qLearn(moMain = d1Main,
              moCont = d1Cont,
              data = data,
              response = q2,
              txName = "A1",
              verbose = FALSE)

  beta1 = coef(q1)$outcome$Combined
  beta2 = coef(q2)$outcome$Combined
  val = estimator(q1)

  return(list("val" = val, "beta1" = beta1, "beta2" = beta2))
}

#thank you Ye for the apply tricks!
MC_calcs = function(est_mat, true_val){
  B = dim(est_mat)[1]
  n = length(true_val)

  mean = apply(est_mat, 2, mean)

  bias = mean - true_val

  relative_bias = bias / true_val

  stddev = apply(est_mat, 2, sd)

  mse = apply( (est_mat - matrix(rep(true_val, B), nrow=B, byrow = TRUE))^2, 2, sum ) /B

  return(list("mean" = mean,
              "bias" = bias,
              "relative_bias" = relative_bias,
              "standard dev" = stddev,
              "mse" = mse))
}


B = 1000
beta2 = c(3,0,0.1,0.5, 0.5, 1, -1, 0.75, 0.5)
beta1 = c(6.8098, 1.6787, -1.2372, 0.4085)
v = 7.64915

beta2_ii = matrix(NA, nrow = B, ncol =length(beta2))
colnames(beta2_ii) = c("Intercept", "X1", "A1", "X2", "I(X2^2)", "A2", "X1:A1", "X2:A2", "A1:A2")
beta2_iii = matrix(NA, nrow = B, ncol =length(beta2) - 1)
```

```r
colnames(beta2_iii) = c("Intercept", "X1", "A1", "X2", "A2", "X1:A1", "X2:A2", "A1:A2")
beta2_iv = matrix(NA, nrow = B, ncol =length(beta2) - 2)
colnames(beta2_iv) = c("Intercept", "X1", "A1", "X2", "A2", "X1:A1", "X2:A2")


beta1_ii = matrix(NA, nrow = B, ncol =length(beta1))
colnames(beta1_ii) = c("Intercept", "X1", "A1", "X1:A1")
beta1_iii = matrix(NA, nrow = B, ncol =length(beta1))
colnames(beta1_iii) = c("Intercept", "X1", "A1", "X1:A1")
beta1_iv = matrix(NA, nrow = B, ncol =length(beta1))
colnames(beta1_iv) = c("Intercept", "X1", "A1", "X1:A1")


vhat_ii = matrix(NA, nrow = B, ncol = 1)
vhat_iii = matrix(NA, nrow = B, ncol = 1)
vhat_iv = matrix(NA, nrow = B, ncol = 1)

 # decision 2 models

moMain_ii2 = buildModelObj(model = ~ X1 + A1 + X1:A1 + X2 + I(X2^2),
                 solver.method = "lm",
                 predict.method = "predict.lm")

moCont_ii2 = buildModelObj(model = ~ A1 + X2,
               solver.method = "lm",
               predict.method = "predict.lm")


moMain_iii2 = buildModelObj(model = ~ X1 + A1 + X1:A1 + X2,
                 solver.method = "lm",
                 predict.method = "predict.lm")

moCont_iii2 = buildModelObj(model = ~ A1 + X2,
                 solver.method = "lm",
                 predict.method = "predict.lm")


moMain_iv2 = buildModelObj(model = ~ X1 + A1 + X1:A1 + X2,
                 solver.method = "lm",
                 predict.method = "predict.lm")

moCont_iv2 = buildModelObj(model = ~ A1,
                 solver.method = "lm",
                 predict.method = "predict.lm")

# decision 1 models

moMain_1 = buildModelObj(model = ~ X1,
                 solver.method = "lm",
                 predict.method = "predict.lm")

moCont_1 = buildModelObj(model = ~ X1,
               solver.method = "lm",
```

```r
                predict.method = "predict.lm")


for(i in 1:B){
  n = 1000

  px = 0.5

  X1 = rbinom(n, 1 , px)


  A1 = rbinom(n, 1, logistic_func(0.3 - 0.5 * X1))


  X2 = rnorm(n,
             1 + 0.5*X1 + -0.75 *A1 + 0.25 * X1 * A1,
             sqrt(2))

  A2 = rbinom(n, 1, logistic_func(0 + 0.05 * X1 + 0.1 * A1 + -1 *X1 * A1 + -0.1 * X2))

  Y = rnorm(n,
            3 + 0 + 0.1 * A1 + 0.5 * X1 * A1 + 0.5 * X2 + 1 * X2^2 + A2*(-1 + 0.75 * X2 + 0.5* A1),
            sqrt(10))

  data = data.frame(cbind(X1, A1, X2, A2, Y))



  qii = QLearn_wrapper(data, moMain_ii2, moCont_ii2, moMain_1, moCont_1)
  beta1_ii[i,] = qii$beta1
  beta2_ii[i,] = qii$beta2
  vhat_ii[i,] = qii$val

  qiii = QLearn_wrapper(data, moMain_iii2, moCont_iii2, moMain_1, moCont_1)
  beta1_iii[i,] = qiii$beta1
  beta2_iii[i,] = qiii$beta2
  vhat_iii[i,] = qiii$val

  qiv = QLearn_wrapper(data, moMain_iv2, moCont_iv2, moMain_1, moCont_1)
  beta1_iv[i,] = qiv$beta1
  beta2_iv[i,] = qiv$beta2
  vhat_iv[i,] = qiv$val
}
```

ii

```r
print(MC_calcs(beta1_ii[,3:4], beta1[3:4]))
```

```
## $mean
##         A1      X1:A1
## -1.2557246  0.4307131
```

```
##
## $bias
##          A1       X1:A1
## -0.01852461  0.02221306
##
## $relative_bias
##         A1      X1:A1
## 0.01497301 0.05437714
##
## $`standard dev`
##        A1     X1:A1
## 0.5082360 0.7697022
##
## $mse
##        A1     X1:A1
## 0.2583887 0.5923425
```

```r
print(MC_calcs(beta2_ii[,c(6, 8, 9)], beta2[7:9]))
```

```
## $mean
##         A2      X2:A2      A1:A2
## -1.0006641  0.5038972  0.7536706
##
## $bias
##           A2        X2:A2        A1:A2
## -0.000664114 -0.246102819  0.253670607
##
## $relative_bias
##           A2        X2:A2        A1:A2
##  0.000664114 -0.328137093  0.507341214
##
## $`standard dev`
##        A2     X2:A2     A1:A2
## 0.3417970 0.4313789 0.1444231
##
## $mse
##         A2      X2:A2      A1:A2
## 0.11670882 0.24646831 0.08518596
```

```r
print(MC_calcs(vhat_ii, v))
```

```
## $mean
## [1] 7.684473
##
## $bias
## [1] 0.03532324
##
## $relative_bias
## [1] 0.00461793
##
## $`standard dev`
## [1] 0.3044966
##
```

```
## $mse
## [1] 0.09387322
```

iii

```r
print(MC_calcs(beta1_iii[,3:4], beta1[3:4]))
```

```
## $mean
##         A1      X1:A1
## -1.2856410  0.4412414
##
## $bias
##          A1       X1:A1
## -0.04844097  0.03274145
##
## $relative_bias
##         A1      X1:A1
## 0.03915371 0.08015042
##
## $`standard dev`
##        A1     X1:A1
## 0.5121488 0.7546151
##
## $mse
##        A1     X1:A1
## 0.2643807 0.5699465
```

```r
print(MC_calcs(beta2_iii[,c(5, 7, 8)], beta2[7:9]))
```

```
## $mean
##         A2      X2:A2      A1:A2
## -0.5493899  0.4526623  0.2750561
##
## $bias
##         A2      X2:A2      A1:A2
##  0.4506101 -0.2973377 -0.2249439
##
## $relative_bias
##         A2      X2:A2      A1:A2
## -0.4506101 -0.3964503 -0.4498878
##
## $`standard dev`
##        A2     X2:A2     A1:A2
## 0.5345972 0.5646487 0.3300251
##
## $mse
##        A2     X2:A2     A1:A2
## 0.4885579 0.4069190 0.1594074
```

```r
print(MC_calcs(vhat_iii, v))
```

```
## $mean
## [1] 7.491432
##
## $bias
## [1] -0.1577179
##
## $relative_bias
## [1] -0.02061901
##
## $`standard dev`
## [1] 0.3209546
##
## $mse
## [1] 0.1277838
```

```r
print("ARE beta1 iii")
```

```
## [1] "ARE beta1 iii"
```

```r
print(MC_calcs(beta1_ii[,3:4], beta1[3:4])$mse/MC_calcs(beta1_iii[,3:4], beta1[3:4])$mse)
```

```
##        A1     X1:A1
## 0.9773359 1.0392948
```

```r
print("ARE beta2 iii")
```

```
## [1] "ARE beta2 iii"
```

```r
print(MC_calcs(beta2_ii[,c(6, 8, 9)], beta2[7:9])$mse/MC_calcs(beta2_iii[,c(5, 7, 8)], beta2[7:9])$mse)
```

```
##        A2     X2:A2     A1:A2
## 0.2388843 0.6056938 0.5343916
```

```r
print("ARE vhat iii")
```

```
## [1] "ARE vhat iii"
```

```r
print(MC_calcs(vhat_ii, v)$mse/MC_calcs(vhat_iii, v)$mse)
```

```
## [1] 0.7346254
```

iii

```r
print(MC_calcs(beta1_iv[,3:4], beta1[3:4]))
```

```
## $mean
##         A1      X1:A1
## -1.2861249  0.4037255
##
## $bias
##           A1         X1:A1
## -0.048924909 -0.004774485
##
## $relative_bias
##          A1       X1:A1
##  0.03954487 -0.01168785
##
## $`standard dev`
##        A1     X1:A1
## 0.5222917 0.7436609
##
## $mse
##        A1     X1:A1
## 0.2749095 0.5525012
```

```r
print(MC_calcs(beta2_iv[,c(5, 7)], beta2[7:8]))
```

```
## $mean
##         A2      X2:A2
## -0.1983923  0.2443510
##
## $bias
##         A2      X2:A2
##  0.8016077 -0.5056490
##
## $relative_bias
##         A2      X2:A2
## -0.8016077 -0.6741986
##
## $`standard dev`
##        A2     X2:A2
## 0.3945042 0.5572222
##
## $mse
##        A2     X2:A2
## 0.7980529 0.5658670
```

```r
print(MC_calcs(vhat_iv, v))
```

```
## $mean
## [1] 7.371793
##
## $bias
## [1] -0.2773571
```

```
##
## $relative_bias
## [1] -0.03625985
##
## $`standard dev`
## [1] 0.3092088
##
## $mse
## [1] 0.1724414
```

```r
print("ARE beta1 iv")
```

```
## [1] "ARE beta1 iv"
```

```r
print(MC_calcs(beta1_ii[,3:4], beta1[3:4])$mse/MC_calcs(beta1_iv[,3:4], beta1[3:4])$mse)
```

```
##        A1     X1:A1
## 0.9399046 1.0721107
```

```r
print("ARE beta2 iv")
```

```
## [1] "ARE beta2 iv"
```

```r
print(MC_calcs(beta2_ii[,c(6, 8)], beta2[7:8])$mse/MC_calcs(beta2_iv[,c(5, 7)], beta2[7:8])$mse)
```

```
##        A2     X2:A2
## 0.1462420 0.4355587
```

```r
print("ARE vhat iv")
```

```
## [1] "ARE vhat iv"
```

```r
print(MC_calcs(vhat_ii, v)$mse/MC_calcs(vhat_iv, v)$mse)
```

```
## [1] 0.5443774
```

g

You can see by these AREs that we lose significant efficiency for even a slight mis-estimation. The ARE of the value decreases as the model gets further from the the truth. Notice that the AREs of the β_1 terms are relatively close to one. This is because they were correctly specified.