Bimodal Covariates and the Estimated MCCE

Brenton Kenkel — PSCI 8357
February 11, 2016

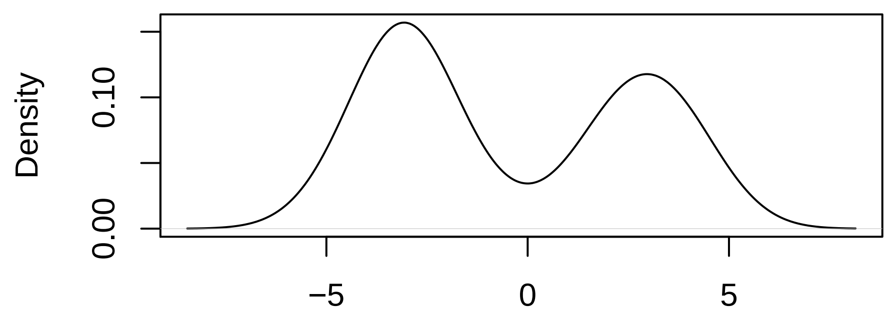
In class today, I mentioned that the estimated standard error of the MCCE for a quadratic relationship tends to be smallest for values close to the sample mean. This raised the question of whether this holds true with a bimodal covariate—if there's relatively little data near the sample mean, is the standard error of the estimated MCCE lowest there? This note will run a little simulation to find out.

First we need to know how to simulate a bimodal covariate. Assume there is a "left" distribution with mean -3 and a "right" distribution with mean 3, and each observation has a 50-50 chance of being chosen from the left or the right.

```
n_obs <- 100
x_left <- rnorm(n_obs, mean = -3)
x_right <- rnorm(n_obs, mean = 3)
is_left <- rbinom(n_obs, size = 1, prob = 0.5)
x <- is_left * x_left + (1 - is_left) * x_right
```

Let's check that the resulting distribution is indeed bimodal.

```
plot(density(x), xlab = "", main = "")
```

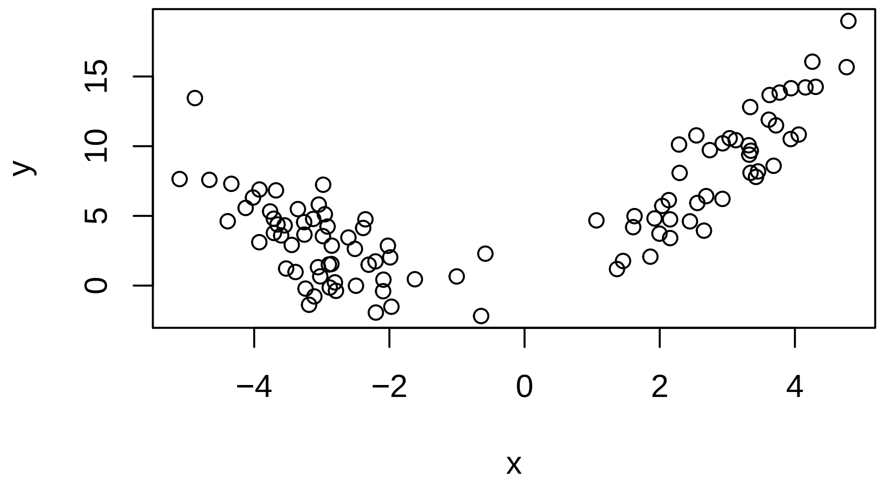


OK, so now let's generate a response variable according to a quadratic model:

$$Y = 1 + \beta_1 X + \beta_2 X^2 + \epsilon.$$

As in class, we'll assume that the expected response is greater at the extremes (e.g., people with extreme viewpoints are more likely to vote). We'll also assume a slight positive trend overall (e.g., conservatives are more likely to vote, all else equal).

```r
y <- 1 + x + 0.5 * x^2 + rnorm(n_obs, sd = 2)
plot(x, y)
```



We can fit a quadratic model and use it to estimate the pointwise marginal changes in conditional expectation,

$$\text{MCCE}(X) = \beta_1 + 2\beta_2 X.$$

```r
fit_quadratic <- lm(y ~ x + I(x^2))
summary(fit_quadratic)
```

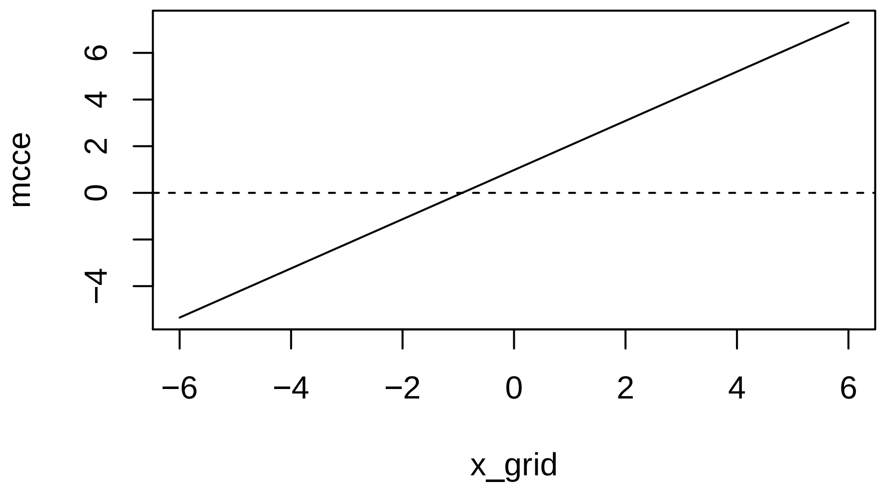
```
##
## Call:
## lm(formula = y ~ x + I(x^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.396 -1.858  0.153  1.477  4.905
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.7792     0.4163    1.87    0.064
```

```
## x              0.9781     0.0659   14.85   <2e-16
## I(x^2)         0.5272     0.0371   14.20   <2e-16
##
## Residual standard error: 2.04 on 97 degrees of freedom
## Multiple R-squared:  0.804,  Adjusted R-squared:   0.8
## F-statistic:  199 on 2 and 97 DF,  p-value: <2e-16
```

To estimate the MCCE function, we'll calculate the estimated MCCE for each point in a grid of potential values of X.

```
x_grid <- seq(-6, 6, length.out = 100)
beta_1 <- coef(fit_quadratic)["x"]
beta_2 <- coef(fit_quadratic)["I(x^2)"]
mcce <- beta_1 + 2 * beta_2 * x_grid

plot(x_grid, mcce, type = "l")
abline(h = 0, lty = 2)
```



At this point we could use `vcov(fit_quadratic)` to estimate the standard error of the MCCEs. But we're not interested in the estimated standard errors—we want the *true* standard errors. In other words, is the estimated MCCE more variable at the extremes (where more of the data lies) or near the mean of X (where data is scarce)? To do this, we'll simulate the whole process repeatedly.

```
library("foreach")
```

```r
n_obs <- 100
x_grid <- seq(-6, 6, length.out = 100)

sim_results <- foreach (i = 1:1000, .combine = "rbind") %do% {
    ## Simulate data
    x_left <- rnorm(n_obs, mean = -3)
    x_right <- rnorm(n_obs, mean = 3)
    is_left <- rbinom(n_obs, size = 1, prob = 0.5)
    x <- is_left * x_left + (1 - is_left) * x_right
    y <- 1 + x + 0.5 * x^2 + rnorm(n_obs, sd = 2)

    ## Fit regression
    fit_quadratic <- lm(y ~ x + I(x^2))

    ## Extract coefficients and calculate MCCE for each element
    ## in the grid
    beta_1 <- coef(fit_quadratic)["x"]
    beta_2 <- coef(fit_quadratic)["I(x^2)"]
    mcce <- beta_1 + 2 * beta_2 * x_grid

    ## Return pointwise MCCEs
    mcce
}
```

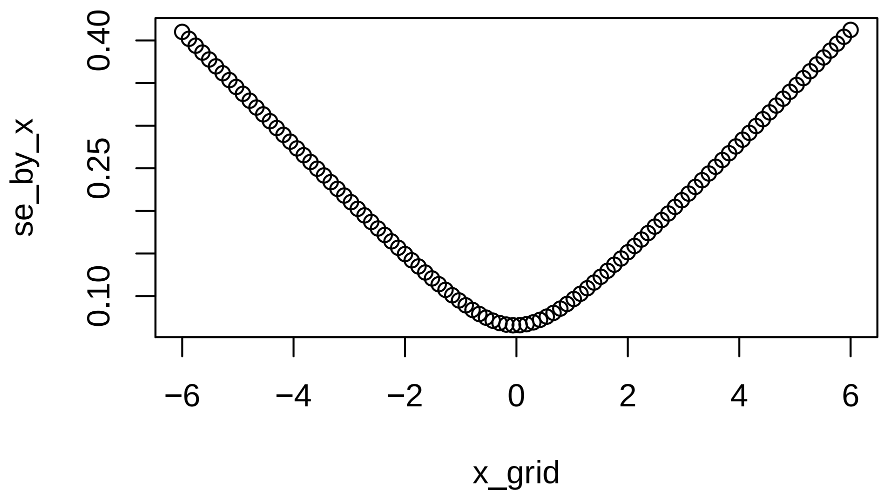
Let's take a look at the simulation results.

```r
dim(sim_results)
```

```
## [1] 1000  100
```

Each column is a value of `x_grid`, our grid of potential values of X. Each row is the estimated set of MCCEs from one iteration of the simulation. So to get the standard error corresponding to each value of X, we will take the standard deviation of each column of `sim_results`. We use `apply()` to apply the same function to each column of a matrix.

```r
se_by_x <- apply(sim_results,
                 MARGIN = 2,  # 1 for rows, 2 for columns
                 FUN = sd)
plot(x_grid, se_by_x)
```



So we see that the standard errors—the true ones, not just the estimated ones—are lowest near the mean of X, even though there is relatively little data there.