Estimates with Regression

Mike Bader

June 24, 2016

Contents

Moving from Mean to Regression

Calculating the univariate mean, as we just did, does not help us much. Our main interest in epidemiology is to find the association between one variable and another. In order to do that, we typically use regression.

Generate Fake Data of Home Values by Metro Size

We will start by using the same variable that we discussed previously, the median price per square foot of housing in metropolitan areas. I suspect that more populous metropolitan areas have higher home prices than less populous metros. This provides a simple model, one with which we are all familiar:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

This represents the linear regression model where y_i represents our outcome variable, price per square foot. Notice the subscript i there that tells us that each unit (metropolitan area in our case) takes on its own value. We estimate the outcome as a function of the intercept, β_0, and the slope of a line, β_1. Notice that neither of these variables includes a subscript, which means that we are indicating that they are the same for all units. The slope of the line predicts the value of y_i given the unit's value of x_i. But, just as the mean was our best guess, now the combination of $\beta_0 + \beta_1$ gives us a more sophisticated best guess – but the actual value between observed y_i and x_i for each unit i will differ. The final term, ϵ_i represents this difference.

We call the β_0 and β_1 the **intercept** and the **slope**, respectively. The intercept represents the value when x_i equals zero and the line crosses, or intercepts, the y-axis. The slope represents the value of the rise over the run and represents the increase in price for each unit change in the x variable. This is likely review, but these terms will become important in a little bit, so I just want to make sure that they are clear.

Write Model of Fake Relationship Between Home Price and Population Size

Now we can move on to specifying a model based on a hypothesized set of parameters. In the case of median price and population size, I anticipate that the percentage change in population size will predict a percentage change in median home value per square foot (this is what economists term elasticity). To model this kind of change, we take the logarithm of both variables (an explanation for why can be found at the bottom of this page). I will create a population described by the process:

$$\ln(y_i) = \beta_0 + \beta_1 \ln(x_i) + \epsilon_i$$

One problem with this model is that β_0 represents the value when the metro population equals zero. That doesn't make a whole lot of sense, so we might want to "center" the population around some meaningful value. This will often be the mean, which makes β_0 the "conditional mean" of price; that is, the price at the mean population size net of the influence of population on price. Let's do something different and write the equation to reflect the difference from a substantively interesting value like, say, the population of the New York metro area (which is about 20.2 million people). I will make a population where the median price per square foot of the metro area increases by half of a percent for every one percent increase in the population of the metro area and the median value equals \$180 at the intercept (the price per square foot of real estate in the New York metro, the natural logarithm of which equals 5.19):

$$\ln(price_i) = 5.19 + 0.5 \times \left[\ln(pop_i) - \ln(20.2 \times 10^6)\right] + \epsilon_i$$

Create Our Fake Population of Metropolitan Areas with Price and Population Size

Now we can input these variables into R to create our population:

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Warning: package 'RCurl' was built under R version 3.2.4

## Loading required package: bitops
```

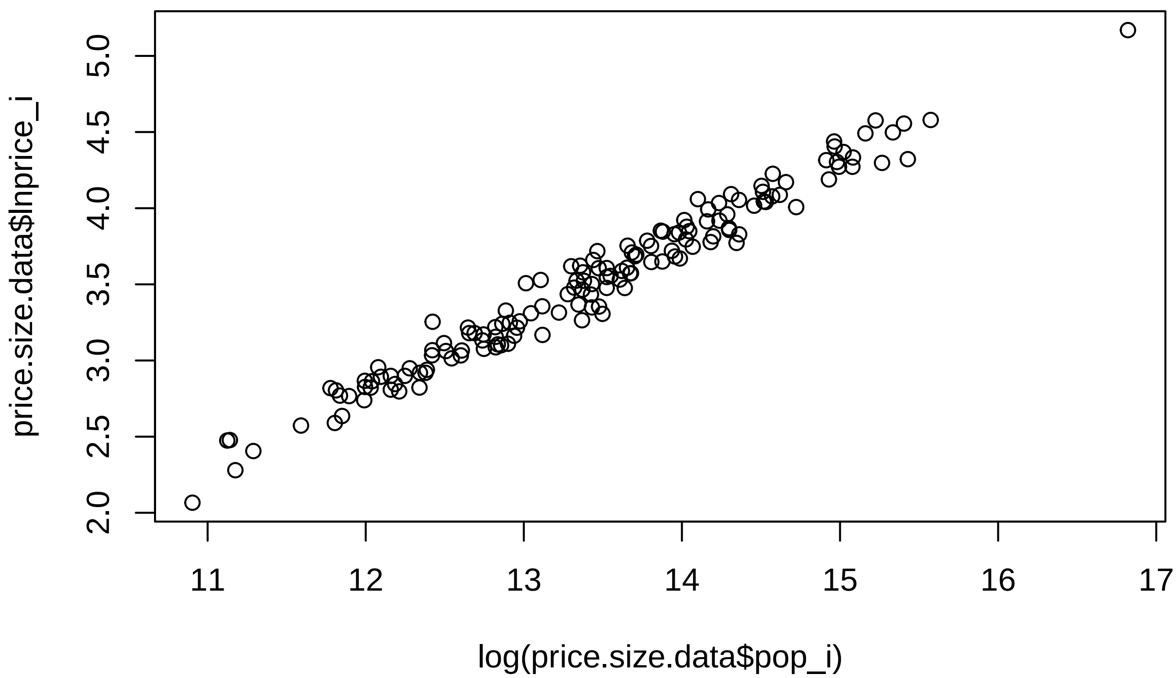
And let's look at our data:

```
price.size.data[c(1:5,146:150),]
```

```
##       i  beta_0 beta_1     pop_i         e_i   price_i lnprice_i
## 1     1 5.192957    0.5  54389.92 -0.16786858  7.896809  2.066459
## 2     2 5.192957    0.5  67796.92  0.13009091 11.876803  2.474587
## 3     3 5.192957    0.5  68921.73  0.12494688 11.913480  2.477671
## 4     4 5.192957    0.5  71356.22 -0.09040120  9.773542  2.279679
```

```
## 5     5 5.192957    0.5    79956.12 -0.02155498  11.083106  2.405422
## 146 146 5.192957    0.5  4566841.82  0.04809957  89.803690  4.497626
## 147 147 5.192957    0.5  4898104.83  0.07117862  95.175100  4.555718
## 148 148 5.192957    0.5  5019288.79 -0.17482851  75.333970  4.321931
## 149 149 5.192957    0.5  5800768.54  0.01024960  97.452039  4.579360
## 150 150 5.192957    0.5 20200000.00 -0.02367323 175.788861  5.169284
```

```
plot(log(price.size.data$pop_i),price.size.data$lnprice_i)
```



Analyze Fake Relationship Between Home Price and Population Size

Look again at the data printed above. Remember, we know the real values of β_0 and β_1 because we played god and make them that way. If we just collected these data (from a population, we are not dealing with sampling), all we would be able to observe would be:

```
price.size.data[c(1:5,146:150),c('i','price_i','pop_i')]
```

```
##       i    price_i       pop_i
## 1     1   7.896809    54389.92
## 2     2  11.876803    67796.92
## 3     3  11.913480    68921.73
## 4     4   9.773542    71356.22
## 5     5  11.083106    79956.12
## 146 146  89.803690  4566841.82
## 147 147  95.175100  4898104.83
## 148 148  75.333970  5019288.79
## 149 149  97.452039  5800768.54
## 150 150 175.788861 20200000.00
```

In order to estimate what process generated these data, we analyze the data based on a regression model that estimates the value of the parameter based on observed data. We would then specify the parameters we want to estimate and then go about estimating them. Recall that this is the same thing that we did with the mean, its just that we now have a slightly more complicated estimate. We would specify our model:

$$\ln(price_i) = \beta_0 + \beta_1 \times \left[\ln(pop_i) - \ln(20.2 \times 10^6)\right] + \epsilon_i$$

then estimate that model and show the line of best fit on the scatter plot to check our work:

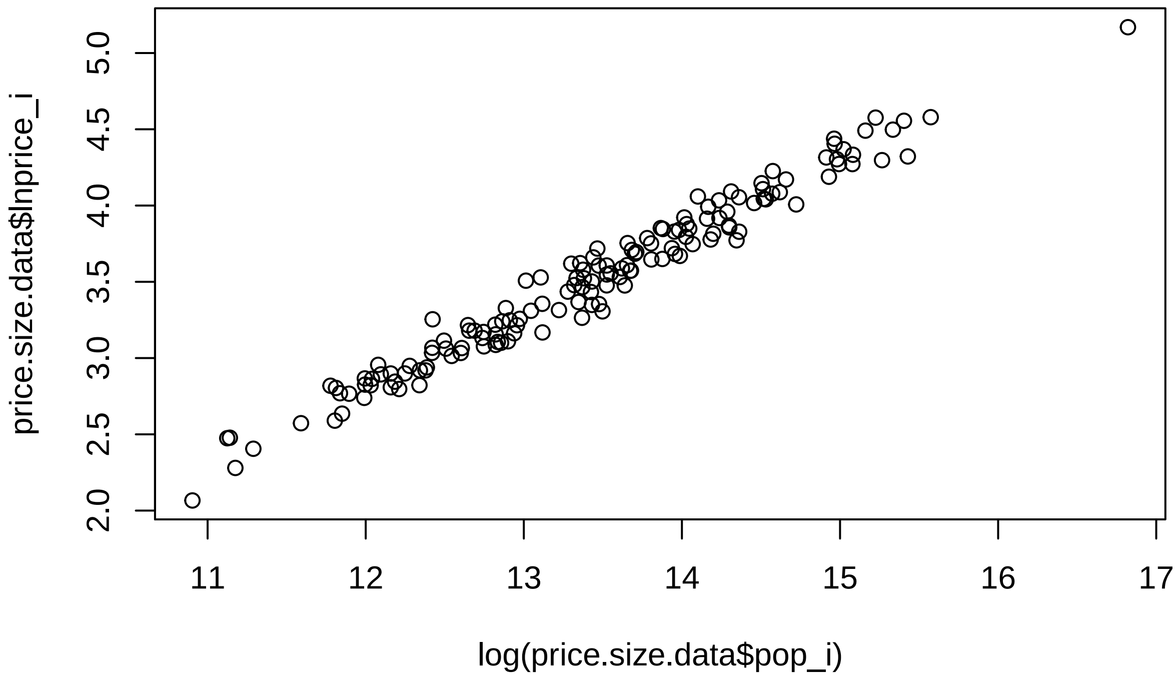
```r
price.size.model <- lm(lnprice_i ~ I(log(pop_i) - log(20.2e6)))
summary(price.size.model)
```

```
##
## Call:
## lm(formula = lnprice_i ~ I(log(pop_i) - log(20200000)))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.239211 -0.067736 -0.005219  0.061202  0.250035
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   5.219949   0.026243  198.91   <2e-16 ***
## I(log(pop_i) - log(20200000)) 0.503779   0.007366   68.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09708 on 148 degrees of freedom
## Multiple R-squared:  0.9693, Adjusted R-squared:  0.9691
## F-statistic:  4678 on 1 and 148 DF,  p-value: < 2.2e-16
```

Et voila! We get estimates very similar to the actual population parameters that we specified in the model. Let's interpret what this means. The intercept, our estimate of β_0, equals 5.22. Since we centered this at the population of the largest city (x=0 when the population equals 20.2 million), this means that we estimated the median home value to be $e^{5.22} = 184.93$ USD in the largest city (our fake New York). The slope, our estimate of β_1, equals 0.5. This represents the elasticity of price by metro size: for every one percent increase in the size of the metro population, we expect the median home value per square foot to increase 0.5 percent.

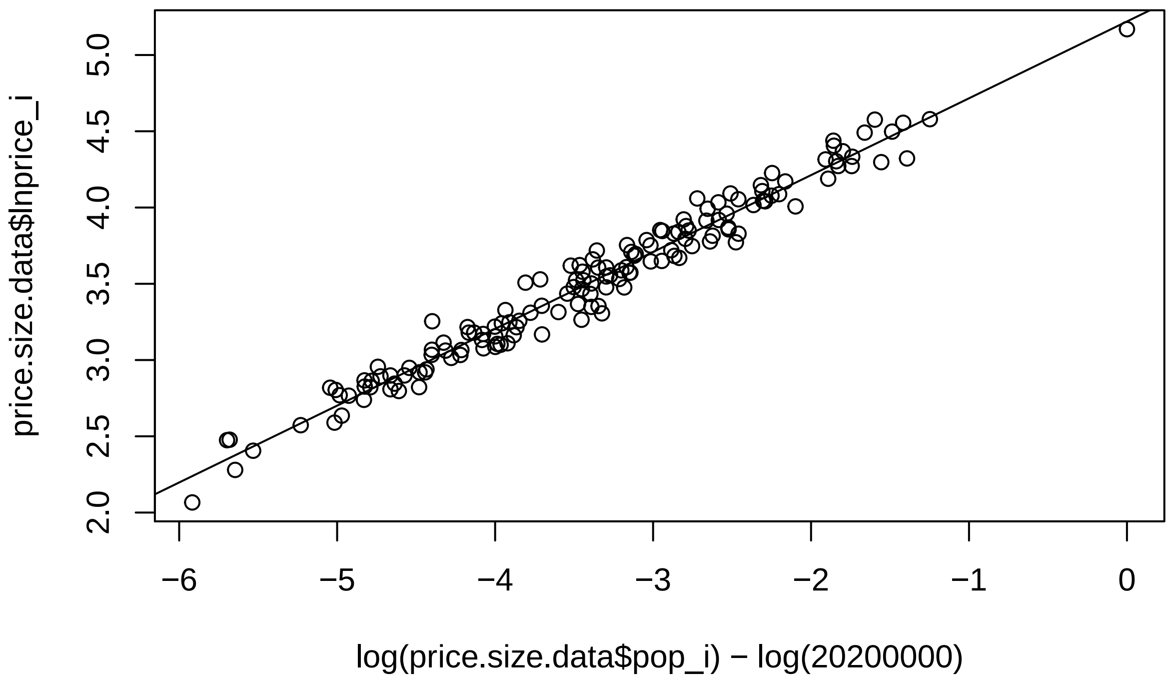
Now let's plot the data and the regression line that we just fit:

```r
plot(log(price.size.data$pop_i),price.size.data$lnprice_i)
abline(price.size.model$coefficients)
```



Uh oh! There is no line! Did we make a mistake? No: remember that we set the intercept in our model to equal the (logged) population of the New York metropolitan area. We need to shift our x-intercept to reflect the model that we ran:

```r
plot(log(price.size.data$pop_i)-log(20.2e6),price.size.data$lnprice_i)
abline(price.size.model$coefficients)
```



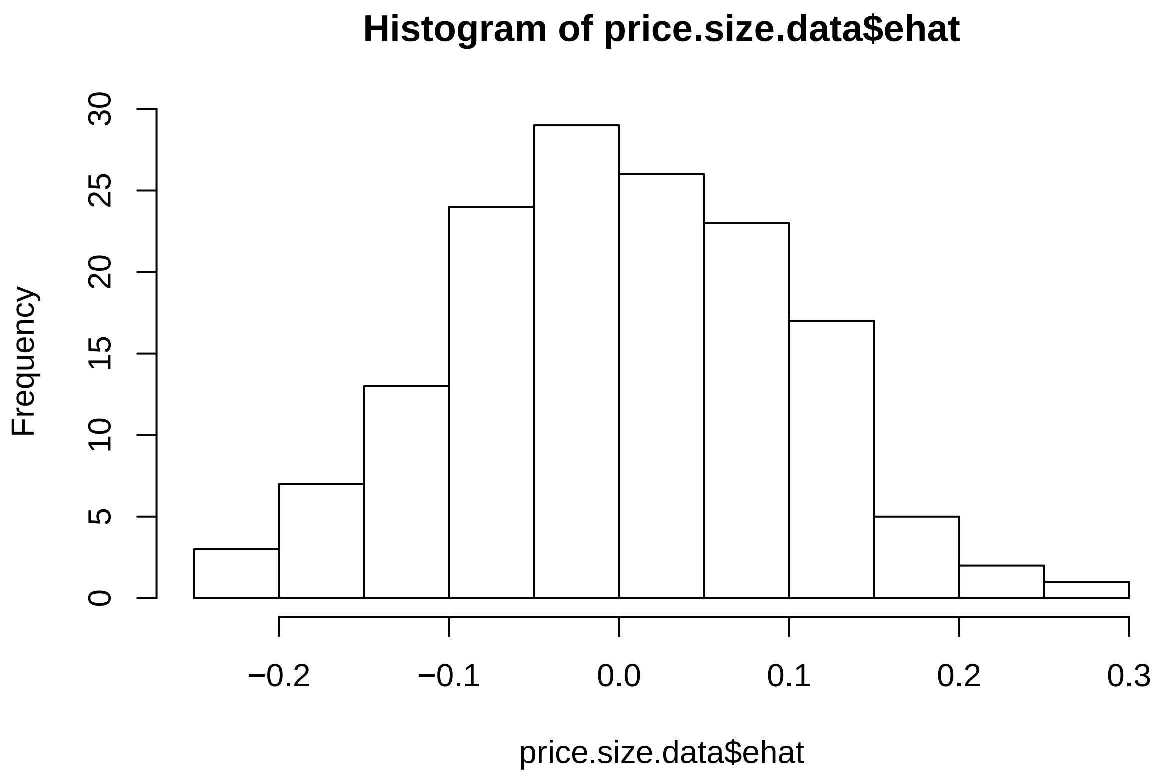
Now we can find the error that exists from the model. Let's add columns representing our estimates of β_0 & β_1 to our dataset (represented by the columns `b0` and `b1`) and then calculate our error from the model (`ehat`, which we get by solving for e_i in the regression equation) :

```r
price.size.data <- mutate(price.size.data,
                          b0 = price.size.model$coefficients[1]
                         ,b1 = price.size.model$coefficients[2]
                         ,ehat = lnprice_i - (b0 + b1*(log(pop_i)-log(20.2e6)))
                         )
round(price.size.data[c(1:5,146:150),],3)
```

```
##       i beta_0 beta_1       pop_i    e_i price_i lnprice_i   b0    b1
## 1     1  5.193    0.5    54389.92 -0.168   7.897     2.066 5.22 0.504
## 2     2  5.193    0.5    67796.92  0.130  11.877     2.475 5.22 0.504
## 3     3  5.193    0.5    68921.73  0.125  11.913     2.478 5.22 0.504
## 4     4  5.193    0.5    71356.22 -0.090   9.774     2.280 5.22 0.504
## 5     5  5.193    0.5    79956.12 -0.022  11.083     2.405 5.22 0.504
## 146 146  5.193    0.5  4566841.83  0.048  89.804     4.498 5.22 0.504
## 147 147  5.193    0.5  4898104.83  0.071  95.175     4.556 5.22 0.504
## 148 148  5.193    0.5  5019288.79 -0.175  75.334     4.322 5.22 0.504
## 149 149  5.193    0.5  5800768.54  0.010  97.452     4.579 5.22 0.504
## 150 150  5.193    0.5 20200000.00 -0.024 175.789     5.169 5.22 0.504
##       ehat
## 1   -0.172
## 2    0.125
## 3    0.119
## 4   -0.096
## 5   -0.028
## 146  0.027
## 147  0.050
## 148 -0.197
## 149 -0.012
## 150 -0.051
```

If you look at the value of `ehat` in the data, it equals the distance from the observed point (circle) on the plot to the line. A positive value means that the model underestimated the (logged) price and a negative value means that the model overestimated the (logged) price. We can then look at the errors:

```r
hist(price.size.data$ehat,breaks=10)
```




```
price.size.sd <- sd(price.size.data$ehat)
price.size.sd
```

```
## [1] 0.09675226
```

```
pct.1sd <- sum(
    price.size.data$ehat >= -price.size.sd
    & price.size.data$ehat <= price.size.sd)/N
pct.1sd
```

```
## [1] 0.6666667
```

The standard deviation of the errors approximately equals the parameter, and the errors are approximately normally distributed with 67% within ± 1s.d. The standard deviation means that we expect that 67% of metropolitan-level estimates of prices will fall within 9.7% of the price expected from the model.

Introducing Real Home Price and Metro Population Data

Now we turn from modeling our fake data that we generated in our own little sandbox of a world, to analyzing real-world data. We will use the same data that we used in the previous section. To analyze the data, however, we also need to append metro population data to the dataset.

Gather Data from Zillow and American Community Survey

The first issue is that we need to connect the Zillow region IDs to the MSA codes used in Census data. Fortunately Zillow publishes this crosswalk. We first read this data and merge it to our data from April 2016 from Zillow. Next, we get ACS data from a file that I downloaded from Social Explorer and available on my website. Then, we merge those two files together.

```
xwlk.url <- 'http://files.zillowstatic.com/research/public/CountyCrossWalk_Zillow.csv'
xwlk <- read.csv(xwlk.url,header=TRUE)[,c('CBSAName','MetroRegionID_Zillow', 'CBSACode')]
xwlk <- xwlk[!duplicated(xwlk),]

zillow.acs <- merge(zillow.apr16[,c('RegionID','RegionName','X2016.04')],xwlk,
                    by.x='RegionID',by.y='MetroRegionID_Zillow')

f.url <- 'https://raw.githubusercontent.com/mikebader/teaching-growth-curve-workshop/master/Data/R11198
acs <- read.csv(f.url,header=T)[,c("Geo_FIPS","Geo_NAME","Geo_DIVISION","SE_T001_001","SE_T057_001")]

zillow.acs <- merge(zillow.acs,acs,by.x='CBSACode',by.y='Geo_FIPS')
zillow.acs <- mutate(zillow.acs,
                 price_i = X2016.04
                ,pop_i = SE_T001_001
                ,lnprice_i = log(X2016.04)
                ,lnpop_i = log(pop_i)
                ,i = 1:150
                )
zillow.acs[c(1:5,146:150),c('i','price_i','pop_i','lnprice_i','lnpop_i')]
```

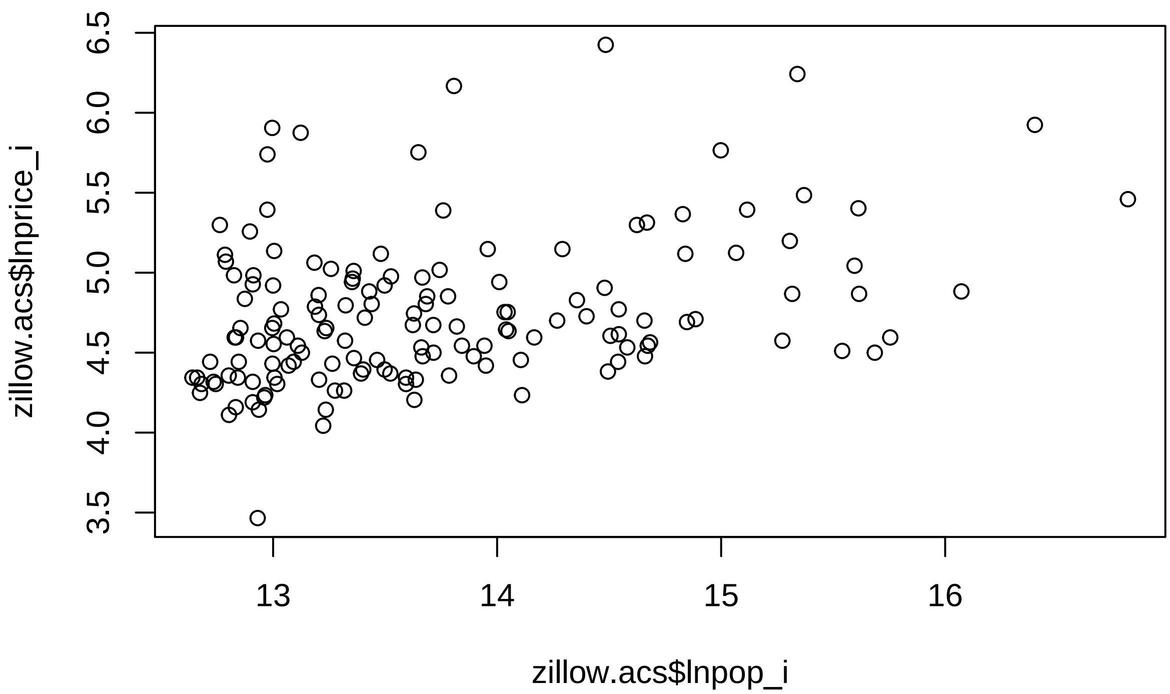
```
##       i price_i   pop_i lnprice_i  lnpop_i
## 1     1      86  703825  4.454347 13.46429
## 2     2     128  880167  4.852030 13.68787
## 3     3      90  905213  4.499810 13.71593
## 4     4     115  829835  4.744932 13.62898
## 5     5     192  398892  5.257495 12.89645
## 146 146     222 6032744  5.402677 15.61271
## 147 147      79  655015  4.369448 13.39241
## 148 148     151  930473  5.017280 13.74345
## 149 149     105  440755  4.653960 12.99624
## 150 150      57  553263  4.043051 13.22359
```

Analyze Zillow and ACS Data

If you look back a few sections, you will see that these data mimic those that we created. Now we are in a position to analyze the data. Recall that both median home prices per square feet and population size tend to be exponentially distributed and we would like to estimate the elasticity of the model. Hence, I took the log of both price and population.

Let's look at a plot of the data:

```r
plot(zillow.acs$lnpop_i,zillow.acs$lnprice_i)
```



A lot more dispersed than our old data, but still trending positive. Now we analyze the combined data with the model of elasticities centering the model so that the intercept equals the estimated home value in New York City:

```r
real.price.model <- lm(lnprice_i ~ I(lnpop_i - log(20.2e6)),data=zillow.acs)
summary(real.price.model)
```

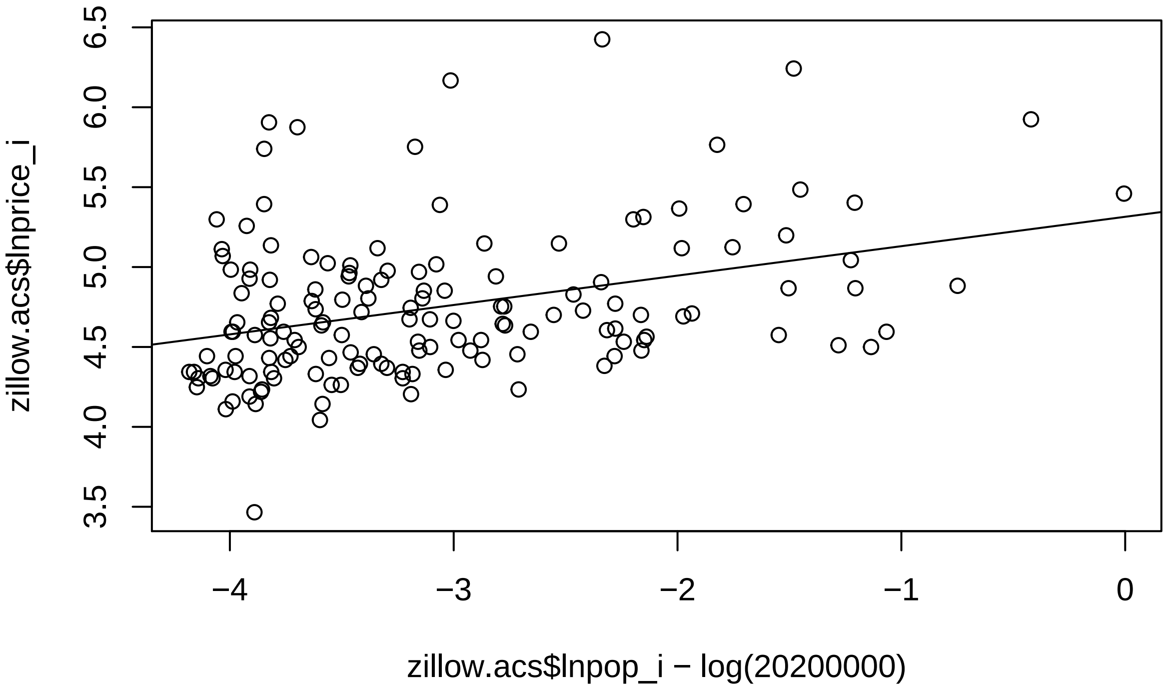
```
##
## Call:
## lm(formula = lnprice_i ~ I(lnpop_i - log(20200000)), data = zillow.acs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13322 -0.28926 -0.09753  0.23092  1.54010
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 5.31453    0.12794  41.539  < 2e-16 ***
## I(lnpop_i - log(20200000))  0.18394    0.03992   4.608 8.72e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4342 on 148 degrees of freedom
## Multiple R-squared:  0.1254, Adjusted R-squared:  0.1195
## F-statistic: 21.23 on 1 and 148 DF,  p-value: 8.723e-06
```

```r
real.coef <- real.price.model$coefficients
```

Interpret Zillow and ACS data

This model tells us that the median price per square foot of homes in the New York metro in April 2016 was $e^{5.31} = \$203$. We would expect a metro area with a one percent larger population than another to have home values 0.18% higher. Let's re-plot the data with this estimation line included:

```r
plot(zillow.acs$lnpop_i-log(20.2e6),zillow.acs$lnprice_i)
abline(real.coef)
```

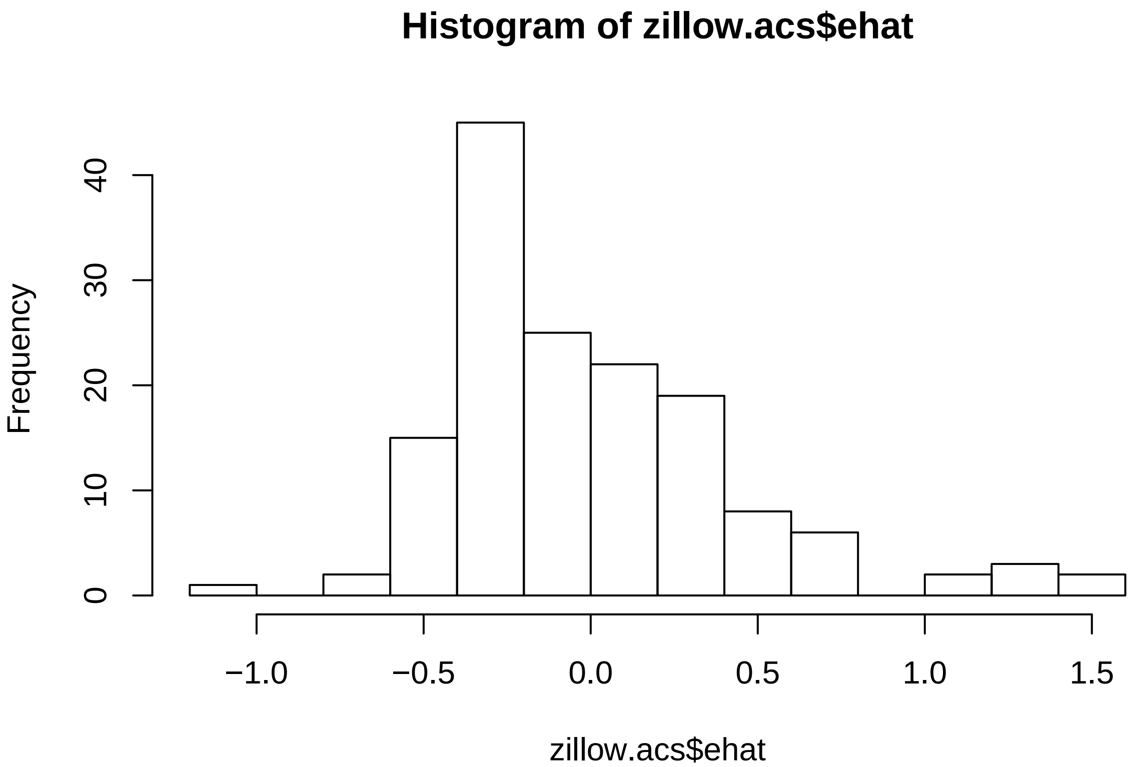


Now we can also calculate the errors off of the trend line, the variation unique to each metropolitan area:

```r
zillow.acs <- mutate(zillow.acs
                , b0 = real.coef[1]
                , b1 = real.coef[2]
                , ehat = lnprice_i - (b0 + b1*(lnpop_i-log(20.2e6)))
)
zillow.acs[c(1:5,146:150),c('i','lnprice_i','lnpop_i','b0','b1','ehat')]
```

```
##       i lnprice_i  lnpop_i       b0        b1        ehat
## 1     1  4.454347 13.46429 5.314528 0.1839413 -0.24270640
## 2     2  4.852030 13.68787 5.314528 0.1839413  0.11385061
## 3     3  4.499810 13.71593 5.314528 0.1839413 -0.24353112
## 4     4  4.744932 13.62898 5.314528 0.1839413  0.01758382
## 5     5  5.257495 12.89645 5.314528 0.1839413  0.66489075
## 146 146  5.402677 15.61271 5.314528 0.1839413  0.31043906
## 147 147  4.369448 13.39241 5.314528 0.1839413 -0.31438568
## 148 148  5.017280 13.74345 5.314528 0.1839413  0.26887647
## 149 149  4.653960 12.99624 5.314528 0.1839413  0.04299867
## 150 150  4.043051 13.22359 5.314528 0.1839413 -0.60972843
```

```r
hist(zillow.acs$ehat,breaks=12)
```




```r
real.price.sd <- sd(zillow.acs$ehat)
pct.1sd <- sum(
    zillow.acs$ehat >= -real.price.sd
    & zillow.acs$ehat <= real.price.sd)/N
pct.1sd
```

```
## [1] 0.7933333
```

We can see that, once again, our model does not fit the data as well as it could. The errors are not centered on zero as they should be and they are skewed right. This could mean that we are missing important variables in the model or could indicate that we need to find a better transformation to make the relationship between price and size more linear.