# Quantile Regression

*Bill Foote*

*February 10, 2018*

## What is it?

When we think "regression" we usually think of linear regression where we estimate parameters based on the mean of the dependent variable. This mean is conditional on the various levels of the independent variables. what we are doing is explaining the variability of the dependent variable around its arithmetic mean. But we all know that this will only work if that mean is truly indicative of the central value of the dependent variable. What if it isn't? What if the distribution of the dependent variable has lots of skewness, and thick (or very thin) tails?

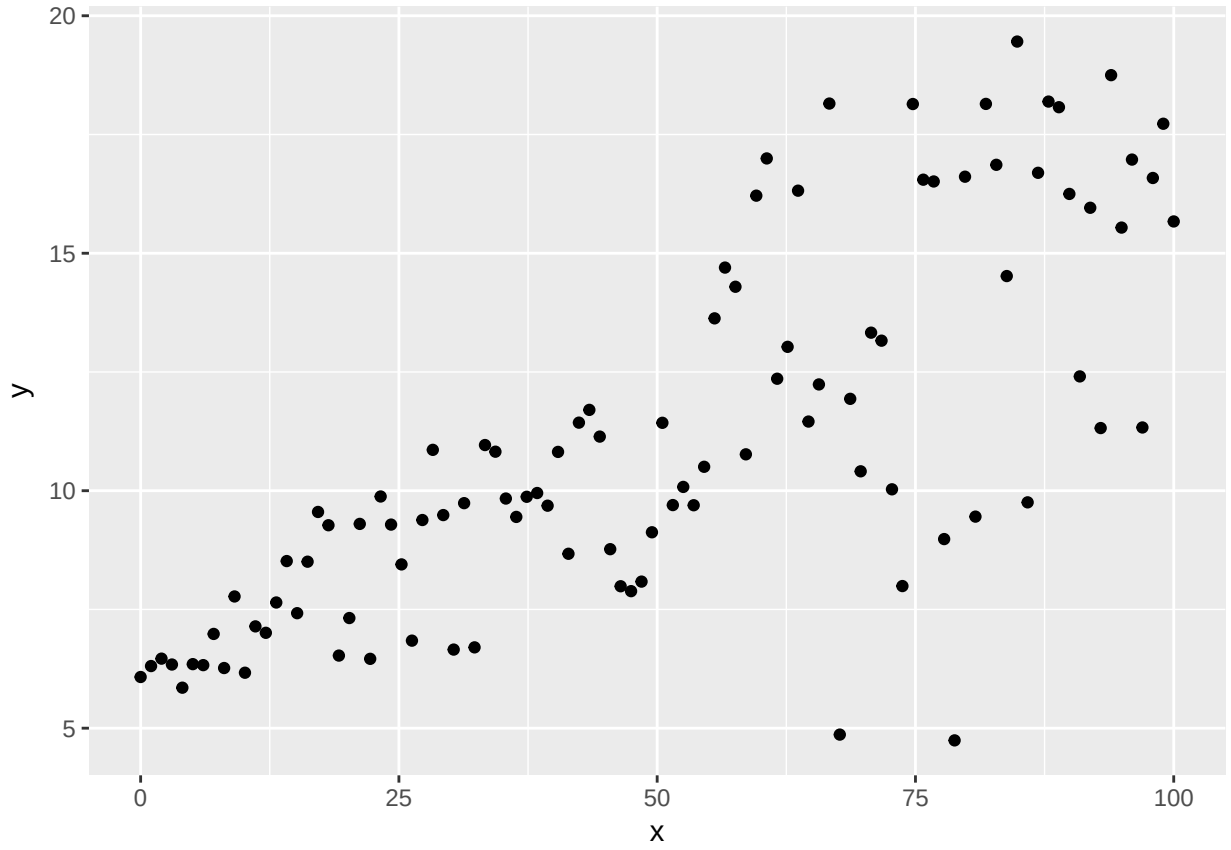
What if we do not have to use the mean? What if we can use other measures of position in the distribution of the dependent variable? We can and we will. These other positions are called quantiles. They measure the fraction of observations of the dependent variable less than the quantile position. Even more, we can measure the deviations of a variable around the quantile conditional on a meaningful list of independent explanatory variables.

But we don't have to always estimate the conditional mean. We could estimate the median, or the 0.25 quantile, or the 0.90 quantile. That's where quantile regression comes in. The math under the hood is a little different, but the interpretation is basically the same. In the end we have regression coefficients that estimate an independent variable's effect on a specified quantile of our dependent variable.

## An example

Here is a motivating "toy" example. Below we generate data with non-constant variance then plot the data using the ggplot2 package:

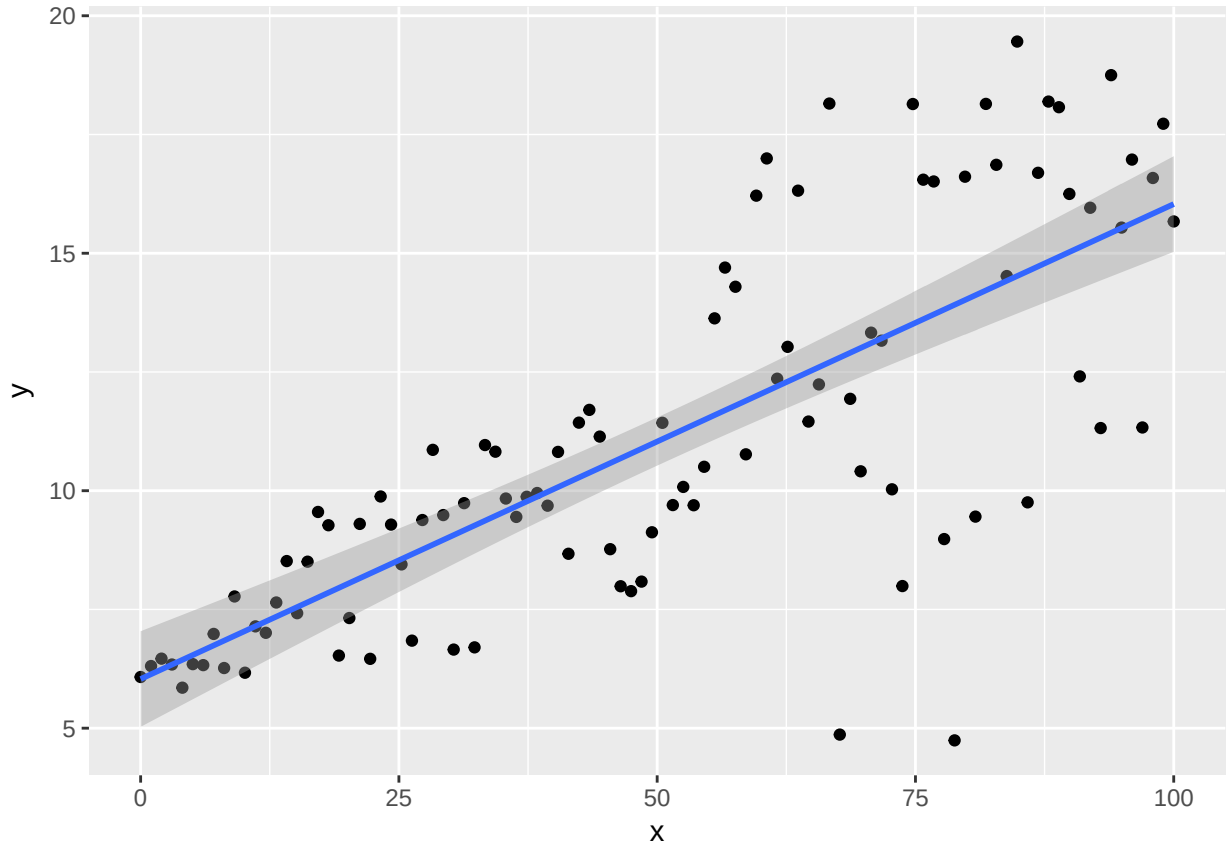
```
# generate data with non-constant variance
#
x <- seq(0,100,length.out = 100)        # independent variable
sig <- 0.1 + 0.05*x                     # non-constant variance
b_0 <- 6                                # true intercept
b_1 <- 0.1                              # true slope
set.seed(1016)                             # make the next line reproducible
e <- rnorm(100,mean = 0, sd = sig)      # normal random error with non-constant variance
y <- b_0 + b_1*x + e                    # dependent variable
dat <- data.frame(x,y)
library(ggplot2)
ggplot(dat, aes(x,y)) + geom_point()
```




What do we observe from this scatter plot? We see the relationship of the dependent variable `y` gets more and more dispersed in its relationship (conditional) with the independent variable `x`, a well-known condition called *heteroskedasticity*. This condition fundamentally violates even the weaker of assumptions around the ordinary least squares (OLS) regression model.

Our errors are normally distributed, but the variance depends on `x`. OLS regression in this scenario is of limited value. It is true that the estimated mean of `y` conditional on `x` is unbiased and as good an estimate of the mean as we could hope to get, but it doesn't tell us much about the relationship between `x` and `y`, especially as `x` gets larger. Let's build a plot of the confidence interval for predicted mean values of `y` using just OLS. The geom_smooth() function regresses y on x, plots the fitted line and adds a confidence interval.

```
# another layer
#
p <- ggplot(dat, aes(x,y)) + geom_point() + geom_smooth(method="lm")
p
```



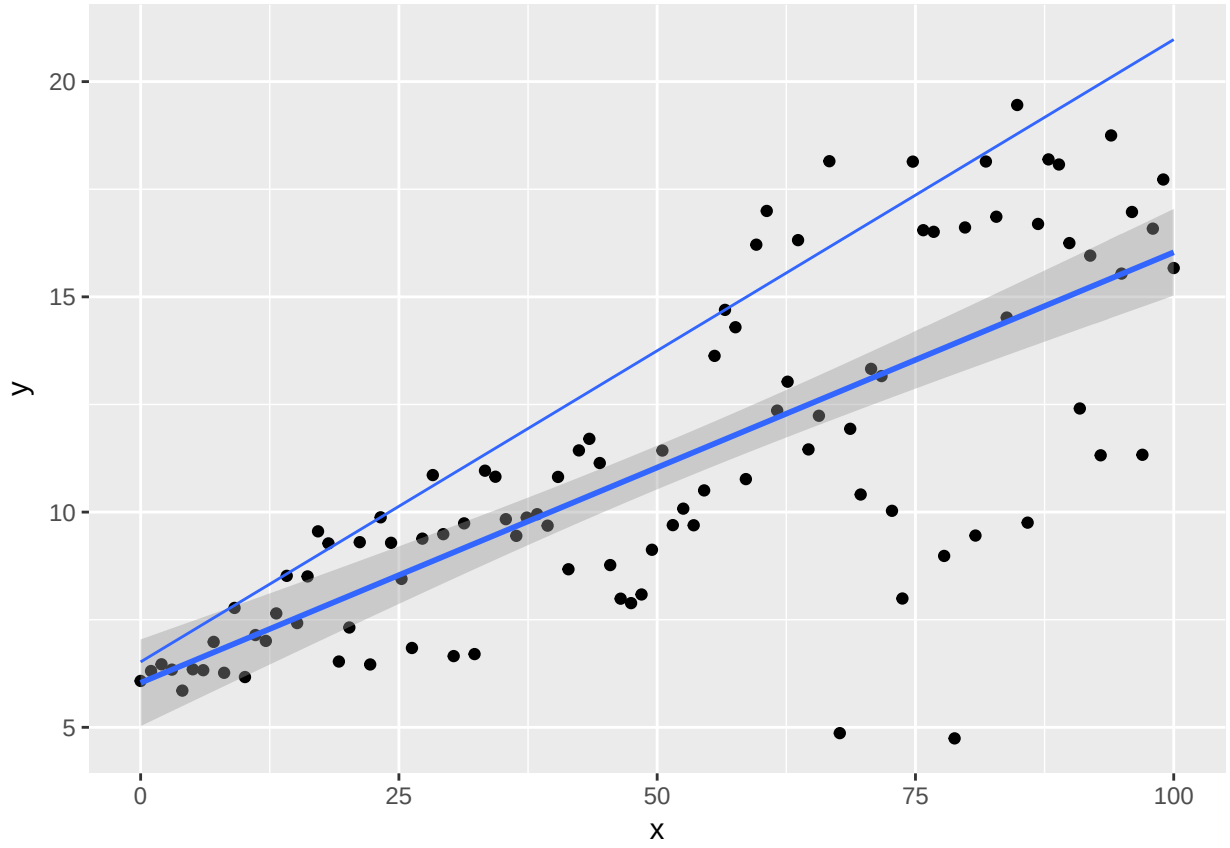

we immediately are taken aback by the incredibly small confidence interval relative to all of the rest of the scatter dots of data! But small `x` does seem to predict `y` fairly well. What about mid and higher levels of the relationship between `y` and `x`? There's much more at work here.

## Even in ggplot2

Just like we used `geom_smooth()` to get the OLS line (through the `lm` method), we can use `geom_quantile()' to build a similar line that estimates intercept (b_0) and slope (b_1) of a line that runs not through the arithmetic mean ofybut through a quantile ofy‘` instead.

Let's try the 1 in 10 quantile of 90/%. We will look at how `x` explains deviation of `y` around the `0.90` quantile of `y` instead of the mean of `y`.

```
p <- p + geom_quantile(quantiles = 0.90)
p
```




Here a lot of the relationship between `y` and `x` is captured at the higher end of the `y` distribution.

## Interpretations

Let's use the `quantreg` package to gain further insight and inference into our toy model. The variable `taus` captures a range of quantiles in steps of `0.10`. The `rq` function is the quantile regression replacement for the OLS `lm` function. We display results using `summary()` with the `boot` option to calculate standard errors `se`.

```
taus <- 1:9/10
qreg_fit <- rq(y ~ x, data=dat, tau = taus)
summary(qreg_fit, se = "boot")
```

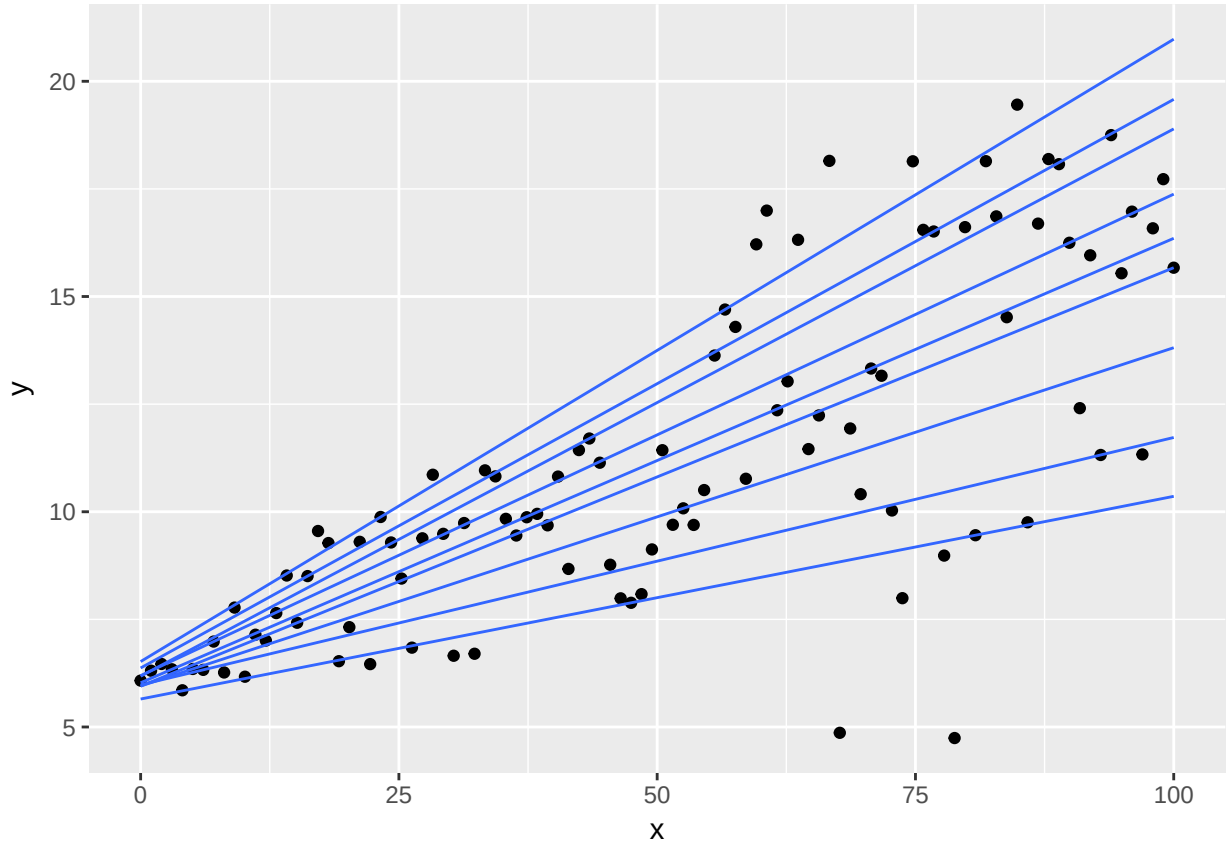
```
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.1
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  5.64945  0.26726   21.13836  0.00000
## x            0.04709  0.00985    4.78175  0.00001
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.2
##
```

```
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  5.97913  0.31078   19.23887  0.00000
## x            0.05746  0.01186    4.84317  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.3
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  5.95148  0.30191   19.71255  0.00000
## x            0.07859  0.01428    5.50356  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.4
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  5.94874  0.23736   25.06228  0.00000
## x            0.09721  0.00955   10.18309  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.5
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  6.02747  0.19581   30.78212  0.00000
## x            0.10324  0.00666   15.50713  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.6
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  6.19363  0.18924   32.72813  0.00000
## x            0.11185  0.00828   13.51283  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.7
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  6.17927  0.26595   23.23510  0.00000
## x            0.12714  0.00993   12.80231  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.8
##
```

```
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  6.37052  0.24443   26.06316  0.00000
## x            0.13209  0.00647   20.42792  0.00000
##
## Call: rq(formula = y ~ x, tau = taus, data = dat)
##
## tau: [1] 0.9
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  6.51972  0.28157   23.15507  0.00000
## x            0.14458  0.01135   12.73487  0.00000
```

The intercepts and slopes change with the quantiles. We can depict these changes in this plot.

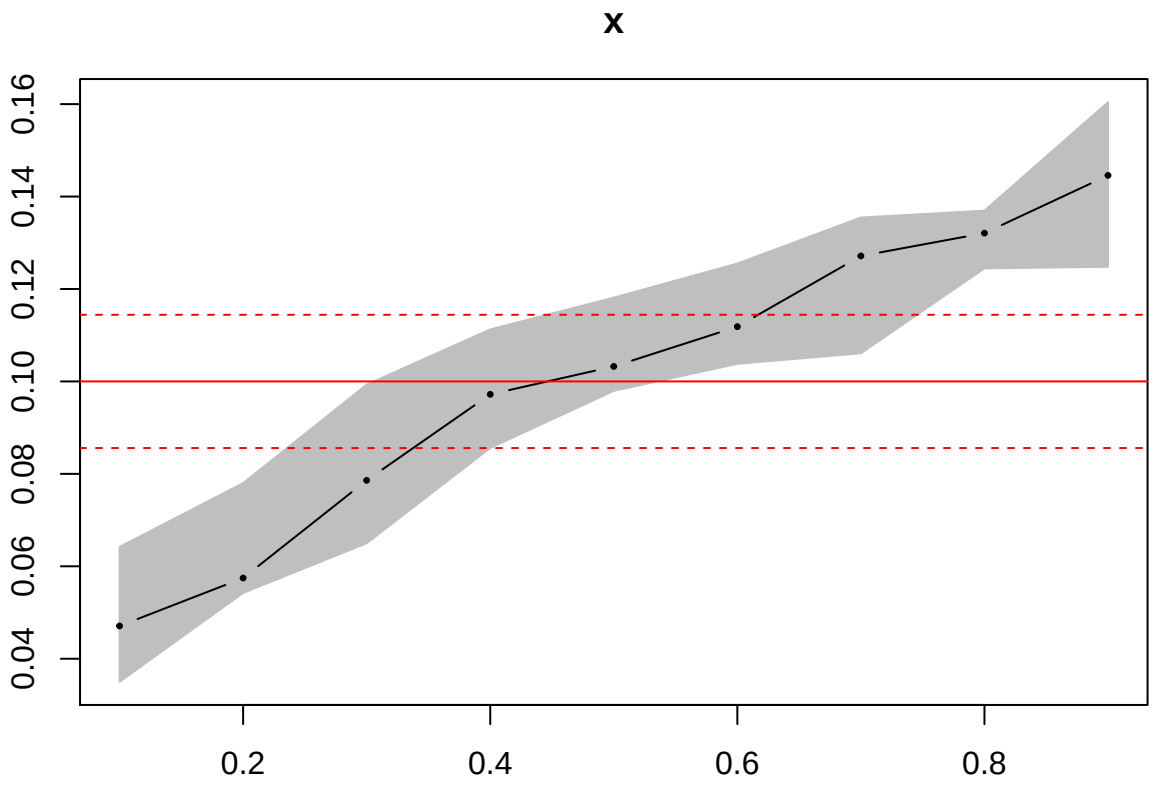
```
p <- ggplot(dat, aes(x,y)) + geom_point() + geom_quantile(quantiles = taus)
p
```




We now have a distribution of intercepts and slopes that more completely describe the relationship between `y` and `x`. The `t-stats` and `p-values` indicate a rejection of the null hypotheses that `b_0 = 0` or that `b_1 = 0`.

The `quantreg` package includes a plot method to visualize the change in quantile coefficients along with their confidence intervals. We use the parm argument to indicate we only want to see the slope (or intercept) coefficients.

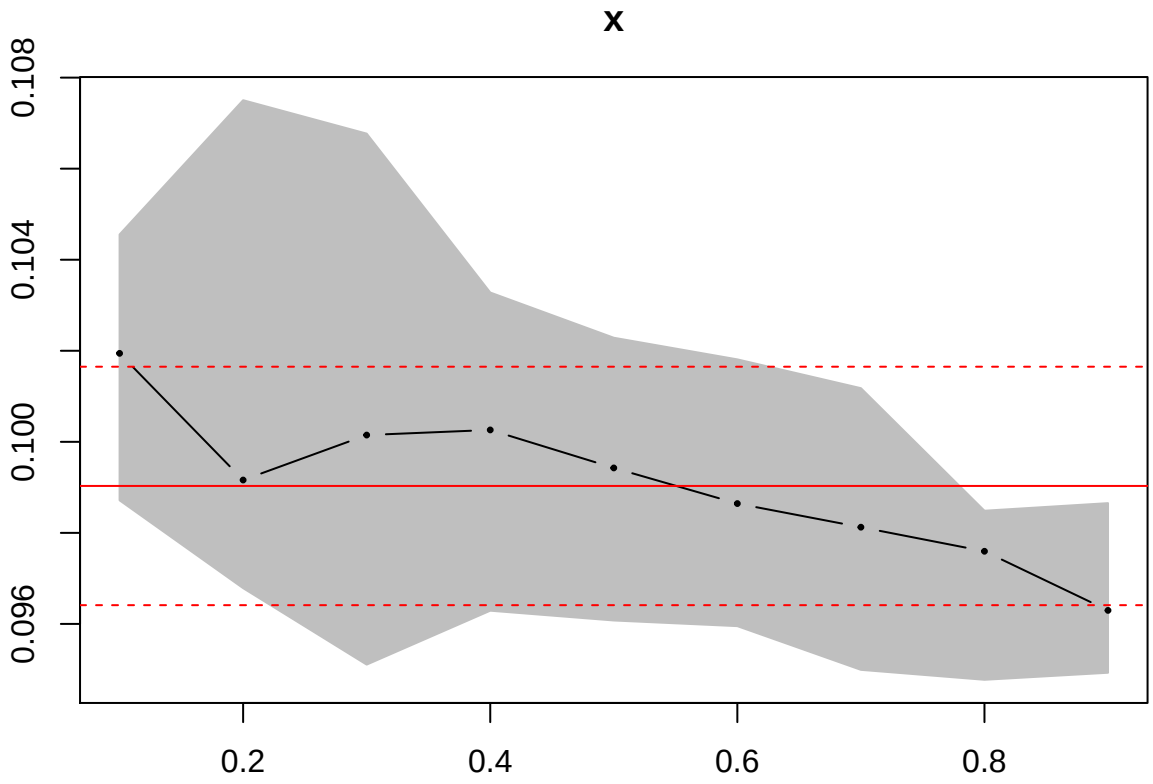
```
plot(summary(qreg_fit), parm="x")
```




Each dot is the slope coefficient for the quantile indicated on the `x` axis with a line connecting them. The red lines are the least squares estimate and its confidence interval. The lower and upper quantiles well exceed the OLS estimate.

Let's compare this whole situation of non-constant variance errors with data that has both normal errors and constant variance. Then let's run quantile regression.

```
x <- seq(0,100,length.out = 100)
b_0 <- 6
b_1 <- 0.1
set.seed(1016)
e <- rnorm(100,mean = 0, sd = 0.5) # constant variance sd^2 = 0.5^2
y <- b_0 + b_1*x + e
dat_normal <- data.frame(x, y)
qreg_fit_normal <- rq(y ~ x, data = dat_normal, tau = taus)
plot(summary(qreg_fit_normal), parm = "x")
```




The fit looks good for and OLS version of the "truth." All of the quantile slopes are within the OLS confidence interval of the OLS slope. Nice.

## References

Koenker, Roger (2005), Quantile Regression (Econometric Society Monographs), Cambridge University Press.