Week 10 Lab

Accounting for Heteroskedasticity and Autocorrelation

Micaela Wood
03/10/2022

Today

- Goldfeld-Quant Test
- Breusch-Pagan Test
- White Test
- Accounting for Clustering in Standard Errors

Packages For Today

```
#install.packages(estimatr)
library(tidyverse)
library(estimatr)
library(huxtable)
library(broom)
```

Goldfeld-Quant Test

First, we need to generate some data with heteroskedasticity.

```
n = 100
set.seed(1234)
data = tibble(
  i = c(1:n),
  e1 = rnorm(n, 0, 1),
  e2 = rnorm(n, 0, 3),
  x = runif(n, 0, 10),
  u = ifelse(x ⩽ 5, e1, e2),
  y = 4 + 3*x + u
)
```

Goldfeld-Quant Test

```
head(data)
```

i	e1	e2	x	u	y
1	-1.21	1.24	6.86	1.24	25.8
2	0.277	-1.42	4.17	0.277	16.8
3	1.08	0.198	7.57	0.198	26.9
4	-2.35	-1.51	7.76	-1.51	25.8
5	0.429	-2.48	5.74	-2.48	18.7
6	0.506	0.501	1.89	0.506	10.2

Goldfeld-Quant Test

The steps for the the test are:

1. Order your the observations by x
2. Split the data into two groups of size n
    - G1: The first third
    - G2: The last third
3. Run separate regressions of y on x for G1 and G2
4. Record RSS1 and RSS2
5. Calculate the G-Q test statistic

Step 1

Order your the observations by x

```
ranked_data = data %>% arrange(x)
head(ranked_data)
```

i	e1	e2	x	u	y
73	-0.367	-0.998	0.00612	-0.367	3.65
50	-0.497	1.04	0.336	-0.497	4.51
60	-1.16	-1.17	0.496	-1.16	4.33
27	0.575	-1.89	0.54	0.575	6.2
86	0.698	5.95	0.739	0.698	6.91
98	0.878	-2.9	0.84	0.878	7.4

Step 2

- Split the data into two groups of size n
  - G1: The first third
  - G2: The last third

```
G1 = ranked_data[1:33,]
head(G1)
```

i	e1	e2	x	u	y
73	-0.367	-0.998	0.00612	-0.367	3.65
50	-0.497	1.04	0.336	-0.497	4.51
60	-1.16	-1.17	0.496	-1.16	4.33
27	0.575	-1.89	0.54	0.575	6.2
86	0.698	5.95	0.739	0.698	6.91

Step 2

```
G2 = ranked_data[67:100,]
head(G2)
```

i	e1	e2	x	u	y
56	0.563	4.09	6.81	4.09	28.5
1	-1.21	1.24	6.86	1.24	25.8
88	-0.403	-1.53	7.03	-1.53	23.6
79	0.258	0.705	7.24	0.705	26.4
39	-0.294	2.76	7.36	2.76	28.8
43	-0.855	1.91	7.45	1.91	28.3

Step 3

Run separate regressions of y on x for G1 and G2

```
lmg1 = lm(data = G1, y~x)
lmg2 = lm(data = G2, y~x)
huxreg(tidy(lmg1), tidy(lmg2))
```

	(1)	(2)
(Intercept)	3.807 ***	7.285
	(0.380)	(4.860)
x	3.073 ***	2.600 ***
	(0.153)	(0.579)
N	0	0

*** p < 0.001; ** p < 0.01; * p < 0.05.

Step 4

Record RSS1 and RSS2

```
test_1 ← unname(resid(lmg1))
test_2 ← unname(resid(lmg2))

rss1 = sum(test_1^2)
rss2 = sum(test_2^2)

rss1
```

```
#> [1] 34.4
```

```
rss2
```

```
#> [1] 336.2
```

Step 5

Calculate the G-Q test statistic

$$F_{(n-k,n-k)} = \frac{RSS_2}{RSS_1}$$

```
GQ = rss2/rss1
GQ
```

```
#> [1] 9.771
```

The F statistic is 2.26 < 9.771 so we can **reject** the null hypothesis that $\sigma_1^2 = \sigma_2^2$

This is the outsome we want because we set up our data so that we have heteroskedasticity. **HOORAY!**

Breusch-Pagan Test

The steps for this test are:

1. Regress y on an intercept and our variables
2. Record residuals $\hat{u}_i$
3. Regress $\hat{u}_i^2$ on an intercept and the same variables
4. Record R^2
5. Test Hypothesis $H_0 : \alpha_1 = \alpha_2 = \ldots = \alpha_k = 0$

Step 1

Regress y on an intercept and our variables

```
bplm = lm(data = data, y ~ x)
huxreg(tidy(bplm))
```

	(1)
(Intercept)	3.864 ***
	(0.576)
x	3.004 ***
	(0.096)
N	0

*** p < 0.001; ** p < 0.01; * p < 0.05.

Step 2

Record residuals $\hat{u}_i$

```
res = unname(resid(bplm))
data_bp = cbind(data, res)
head(data_bp)
```

i	e1	e2	x	u	y	res
1	-1.21	1.24	6.86	1.24	25.8	1.35
2	0.277	-1.42	4.17	0.277	16.8	0.398
3	1.08	0.198	7.57	0.198	26.9	0.305
4	-2.35	-1.51	7.76	-1.51	25.8	-1.4
5	0.429	-2.48	5.74	-2.48	18.7	-2.36
6	0.506	0.501	1.89	0.506	10.2	0.635

Step 3

Regress $\hat{u}_i$ on an intercept and the same variables

```
data_bp = data_bp %>% mutate(
  res_sq = res^2
)
res_lm = lm(data = data_bp, res_sq ~ x)
huxreg(res_lm)
```

	(1)
(Intercept)	-1.481
	(2.657)
x	1.503 ***
	(0.441)
N	100

Step 4

Record R^2

```
Rsq = summary(res_lm)$adj.r.squared
Rsq
```

```
#> [1] 0.09696
```

Step 5

Test Hypothesis $H_0 : \alpha_1 = 0$

```
LM = n*Rsq
LM
```

```
#> [1] 9.696
```

The χ^2 statistic for 5% power is 3.84 < 9.696 so we again **reject** the null hypothesis

White Test

The steps for the White test are:

1. Regress y on our variables and save the residuals.
2. Regress residuals squared on variables, their squares, and interactions
3. Record the R^2
4. Calculate the test statistics for $H_0 : \alpha_p = 0$ for all $p \neq 0$

Step 1

Regress y on our variables and save the residuals.

```
lm_white = lm(data = data, y ~ x)
res_w = unname(resid(lm_white))
white_df = cbind(data, res_w)
head(white_df)
```

i	e1	e2	x	u	y	res_w
1	-1.21	1.24	6.86	1.24	25.8	1.35
2	0.277	-1.42	4.17	0.277	16.8	0.398
3	1.08	0.198	7.57	0.198	26.9	0.305
4	-2.35	-1.51	7.76	-1.51	25.8	-1.4
5	0.429	-2.48	5.74	-2.48	18.7	-2.36

Step 2

Regress residuals on variables, their squares, and interactions

```
white_df = white_df %>% mutate(res_w_sq = res_w^2)
lm_res_w = lm(data = white_df, res_w_sq ~ x + I(x^2))
```

Step 2

```
huxreg(tidy(lm_res_w))
```

	(1)
(Intercept)	-1.366
	(4.095)
x	1.441
	(1.752)
I(x^2)	0.006
	(0.166)
N	0

*** p < 0.001; ** p < 0.01; * p < 0.05.

Step 3

Record the R^2

```
rsq_w = summary(lm_res_w)$adj.r.squared
```

Step 4

Calculate the test statistics for $H_0 : \alpha_p = 0$ for all $p \neq 0$

```
LM_w = rsq_w *n
LM_w
```

```
#> [1] 8.766
```

The χ^2_2 statistic for 5% power is 5.991 < 8.766 so we **reject** the null hypothesis.

Correlation in Standard errors

- Sometimes we may be worried that there are unobservables that are similar among people in different groups
    - People in different states
    - People in different universities
    - People who choose to get a vaccine or not
    - Many other cases

Clustering

Let's make a Dataframe where there are groups that have different error terms.

```
n = 100
set.seed(1234)
data2 = tibble(
  i = c(1:n),
  e1 = rnorm(n, 0, 1),
  e2 = rnorm(n, 2, 1),
  v = rnorm(n, 0, 3),
  state = ifelse(i ⩽ 50, 'Oregon', 'Washington'),
  u = ifelse(state == 'Oregon', e1, e2),
  x = runif(n, 1, 10),
  y = 2 + 3*x + u + v
)
```

Clustering

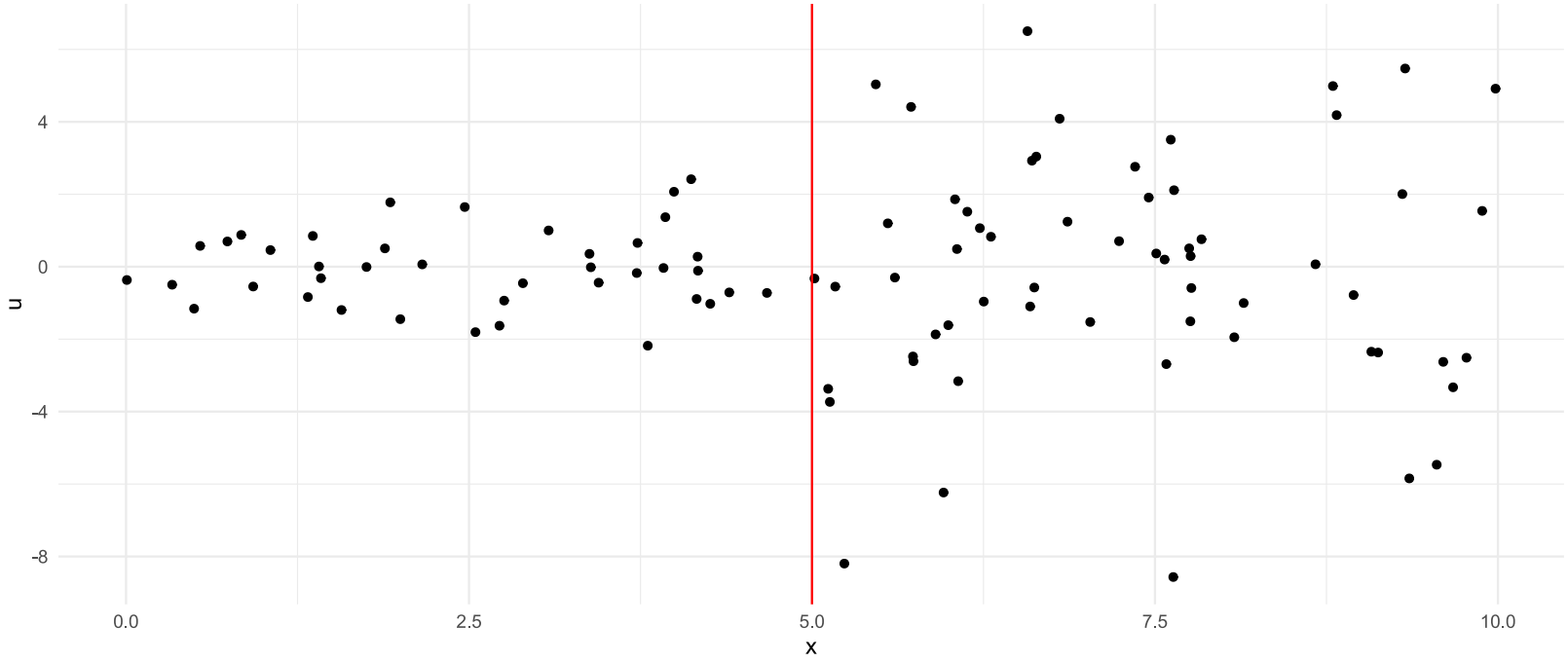
```
head(data2)
```

i	e1	e2	v	state	u	x	y
1	-1.21	2.41	1.46	Oregon	-1.21	3.53	12.8
2	0.277	1.53	2.09	Oregon	0.277	2.57	12.1
3	1.08	2.07	0.557	Oregon	1.08	2.53	11.2
4	-2.35	1.5	2.1	Oregon	-2.35	6.04	19.9
5	0.429	1.17	0.935	Oregon	0.429	4.86	17.9
6	0.506	2.17	2.28	Oregon	0.506	4.5	18.3

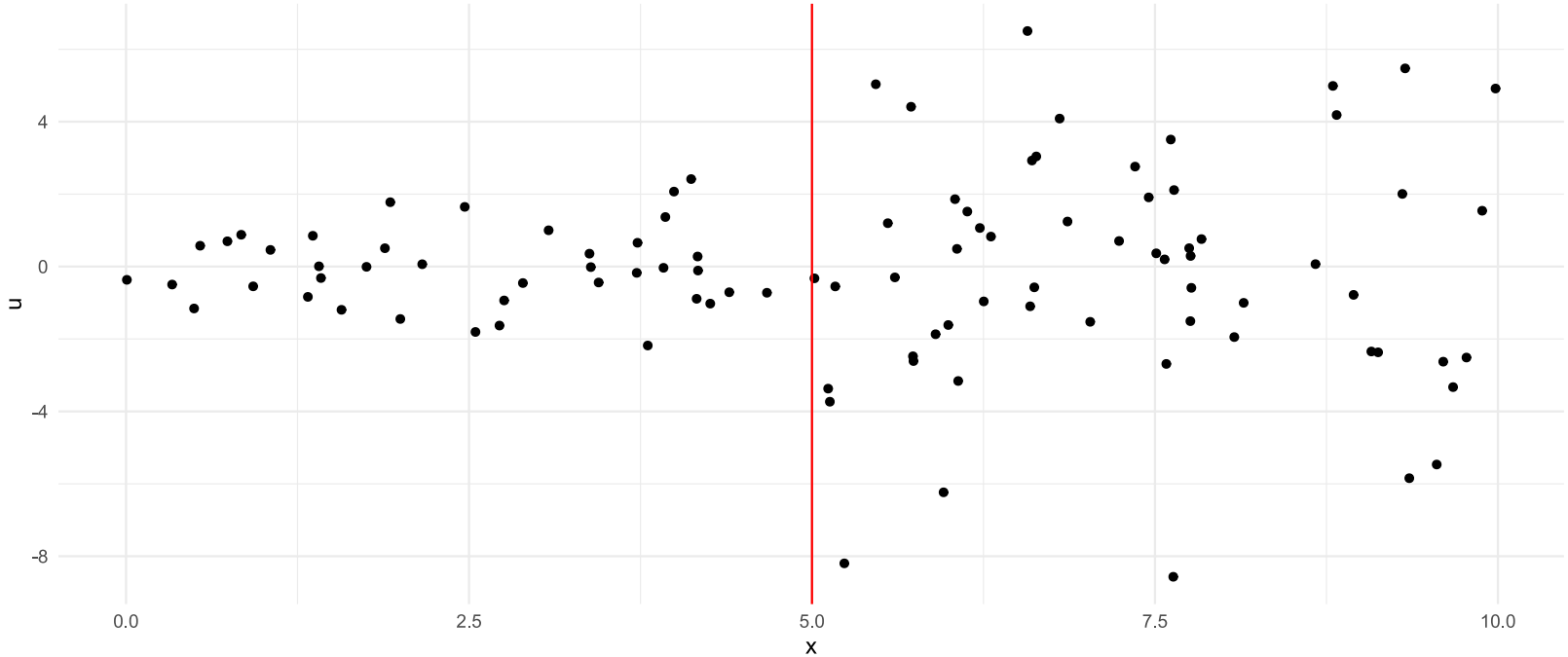
Clustering

Now let's see how the errors compare to each other by graphing the errors and x terms from the first data set and the second

```
ggplot(data = data, aes(x = x, y = u))+
  geom_point() + theme_minimal() + geom_vline(xintercept = 5, color = 'red')
```



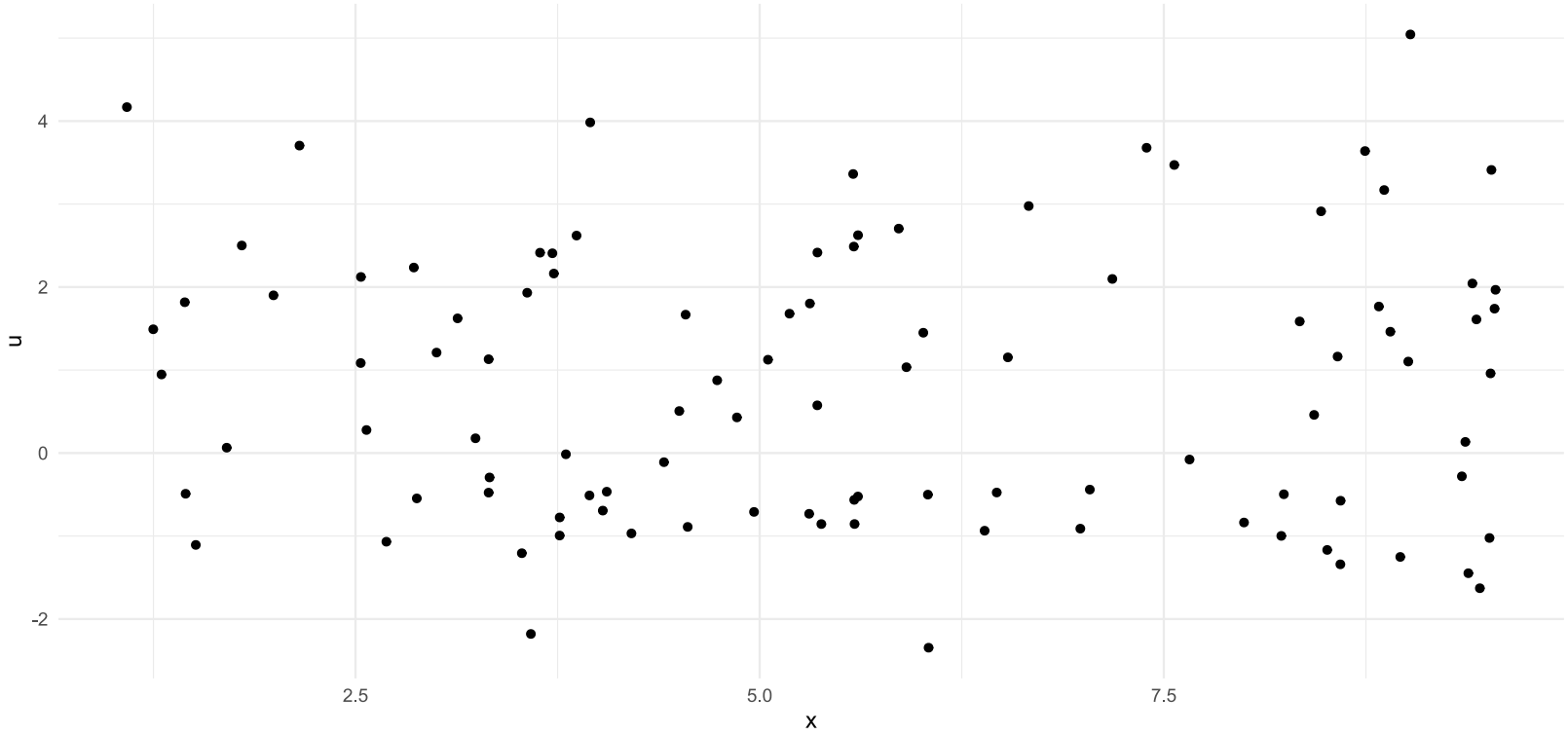
Clustering



Here we can see that the variance in the standard errors gets larger once x gets big enough. So we have heteroskedasticity.

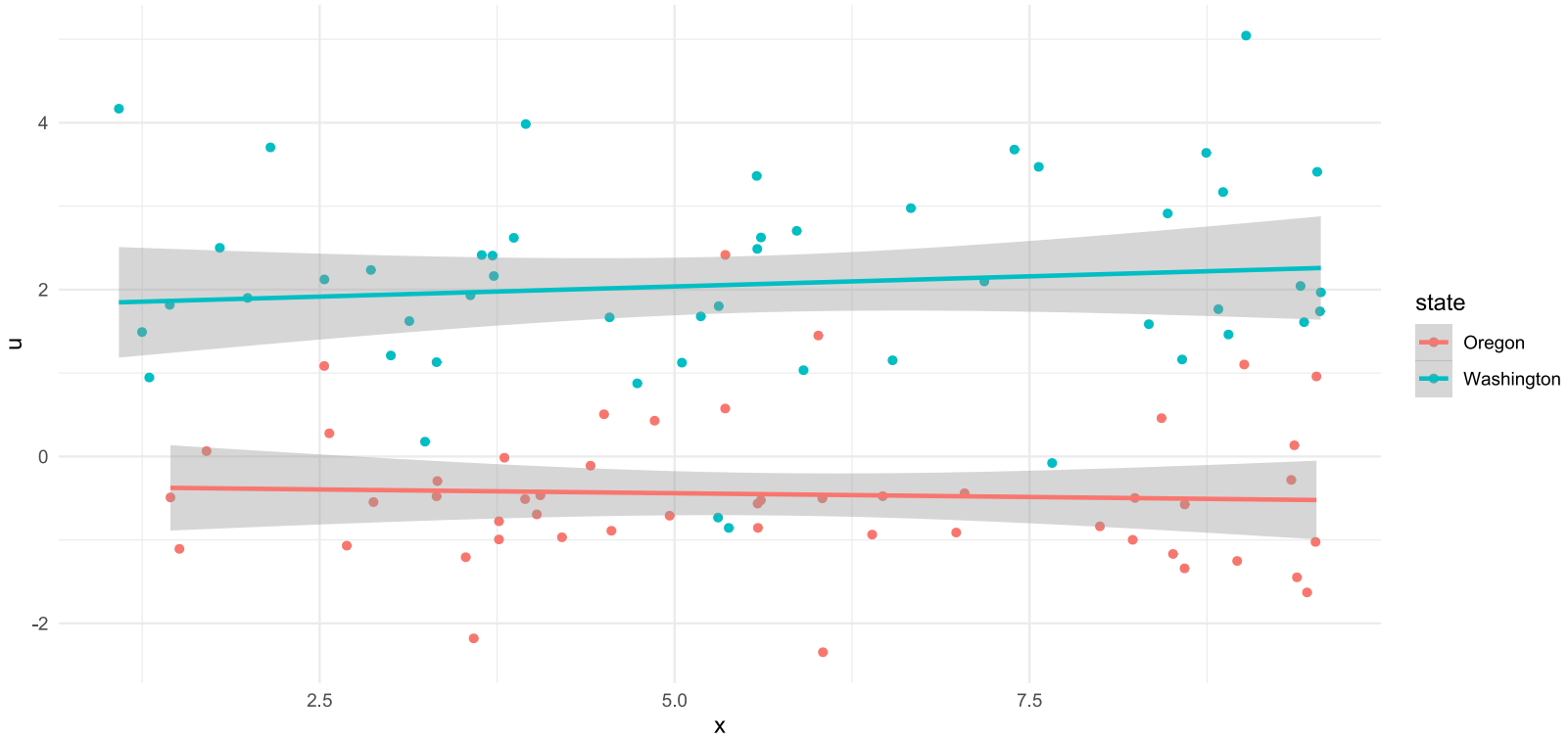
Clustering

```
ggplot(data = data2, aes(x = x, y = u))+
  geom_point() + theme_minimal()
```

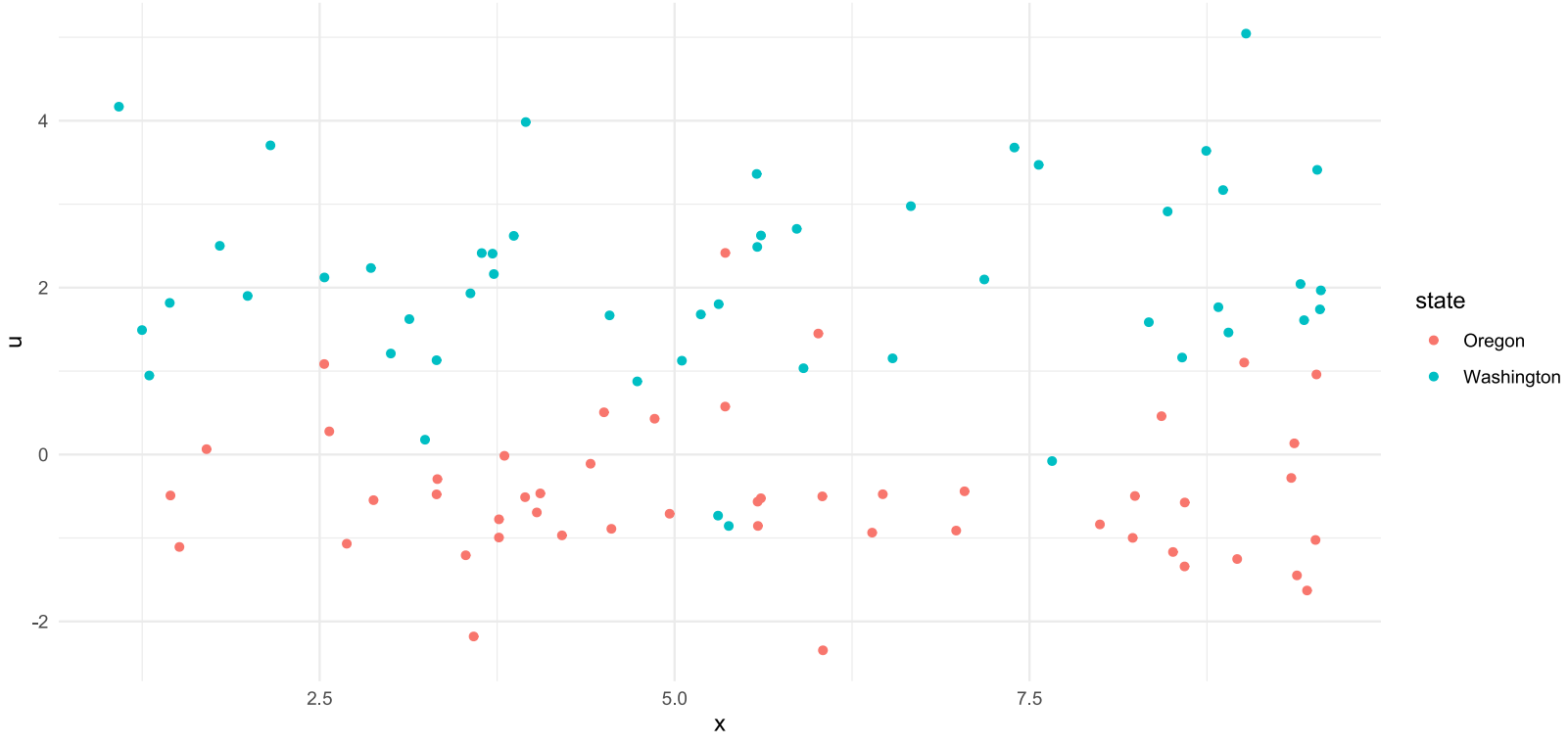


Clustering

```
ggplot(data = data2, aes(x = x, y = u, color = state))+
  geom_point() + theme_minimal() + geom_smooth(method = 'lm')
```



Clustering



Here the variance in the standard errors is the same variance across different sizes of x, but it is different for each state. This is correlation of standard errors. The standard errors of one person in Oregon is correlated with others.

Clustering

Let's see what happens when we run a regression to see what happens when we don't account for this grouping.

```
lm1 = lm(data = data2, y ~ x )
```

Clustering

```
huxreg(tidy(lm1))
```

	(1)
(Intercept)	2.473 **
	(0.823)
x	3.142 ***
	(0.134)
N	0

*** p < 0.001; ** p < 0.01; * p < 0.05.

Clustering

Now let's use the `estimatr` package to run a regression that can account for these clustered standard errors.

```
#install.packages(estimatr)
#library(estimatr)

lm2 = lm_robust(data = data2, y ~ x,
                clusters = state)
#huxreg(tidy(lm1), lm2)
```

Clustering

	(1)	(2)
(Intercept)	2.473 **	2.473
	(0.823)	(0.294)
x	3.142 ***	3.142 *
	(0.134)	(0.144)
N	0	100
R2		0.849
*** p < 0.001; ** p < 0.01; * p < 0.05.		

By allowing for clustering in our standard errors we get more accurate standard errors and are less likely to make error in our inference. **YAY!**