

# Displaying large data

**Hadley Wickham**

**October 2009**

1. Introduction to the diamonds data
2. Histograms and bar charts
3. More boxplots
4. Scatterplots for large data
5. Some theory

# Diamonds data

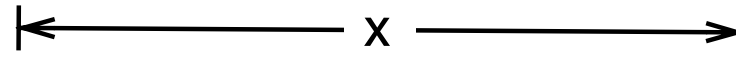
~**54,000** round diamonds from  
<http://www.diamondse.info/>

Carat, colour, clarity, cut

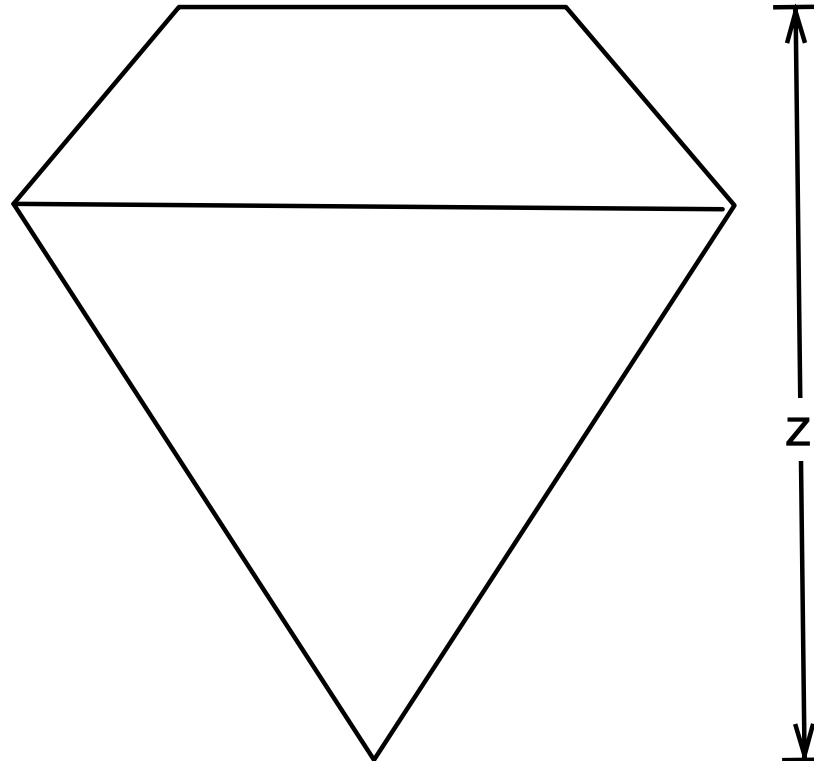
Total depth, table, depth,  
width, height

Price





← table width →



$$\text{depth} = z / \text{diameter}$$
$$\text{table} = \text{table width} / x * 100$$

# Histogram & bar charts

# Histograms and barcharts

Used to display the **distribution** of a variable

Categorical variable → bar chart

Continuous variable → histogram

Always  
experiment with  
the bin width!

# Examples

```
# With only one variable, qplot guesses that  
# you want a bar chart or histogram  
qplot(cut, data = diamonds)
```

```
qplot(carat, data = diamonds)  
qplot(carat, data = diamonds, binwidth = 1)  
qplot(carat, data = diamonds, binwidth = 0.1)  
qplot(carat, data = diamonds, binwidth = 0.01)  
resolution(carat)
```

Common ggplot2  
technique: adding  
together plot  
components

```
last_plot() + xlim(0, 3)
```



```
qplot(table, data = diamonds, binwidth = 1)

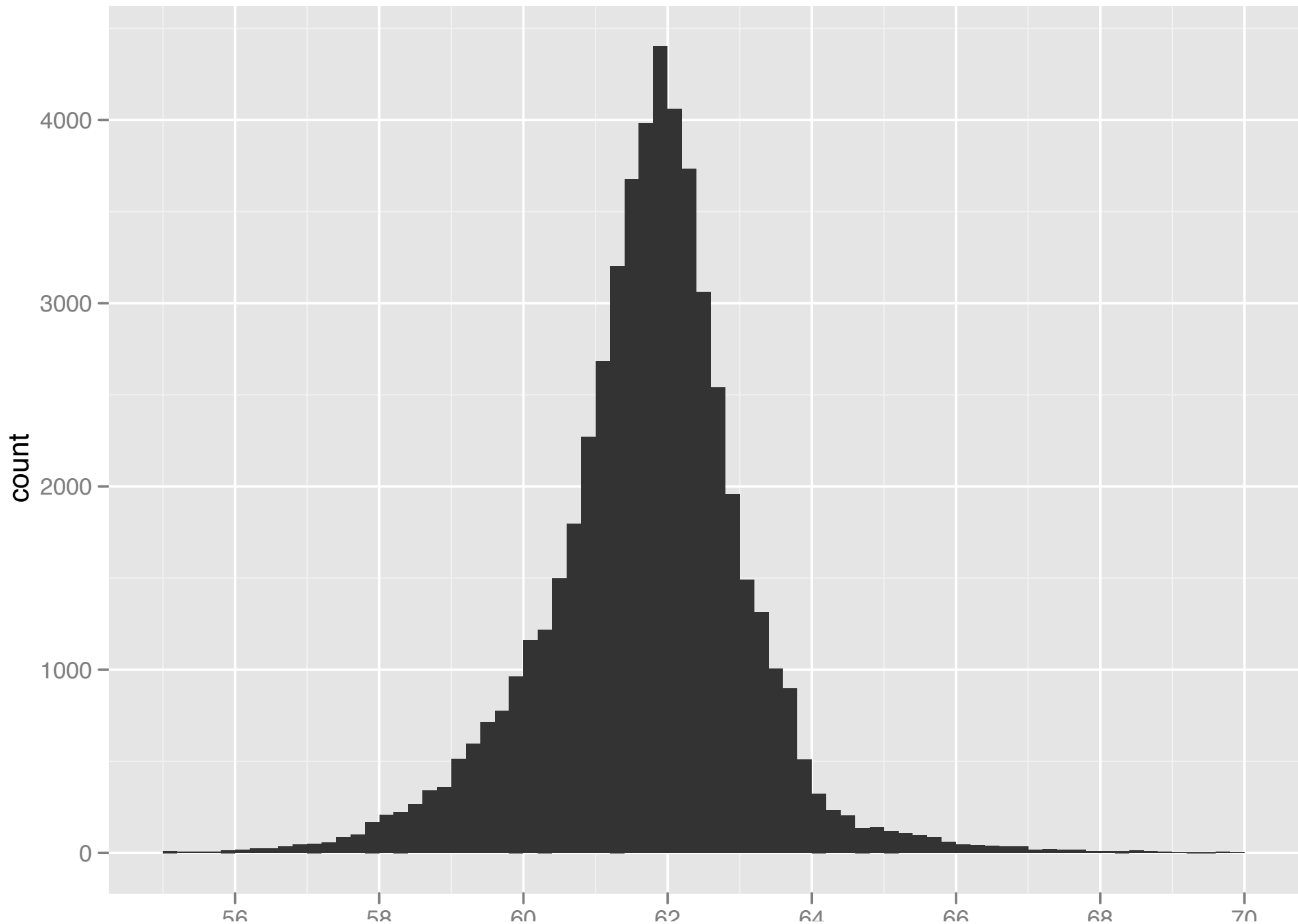
# To zoom in on a plot region use xlim() and ylim()
qplot(table, data = diamonds, binwidth = 1) +
  xlim(50, 70)
qplot(table, data = diamonds, binwidth = 0.1) +
  xlim(50, 70)
qplot(table, data = diamonds, binwidth = 0.1) +
  xlim(50, 70) + ylim(0, 50)

# Note that this type of zooming discards data
outside of the plot regions
# See coord_cartesian() for an alternative
```

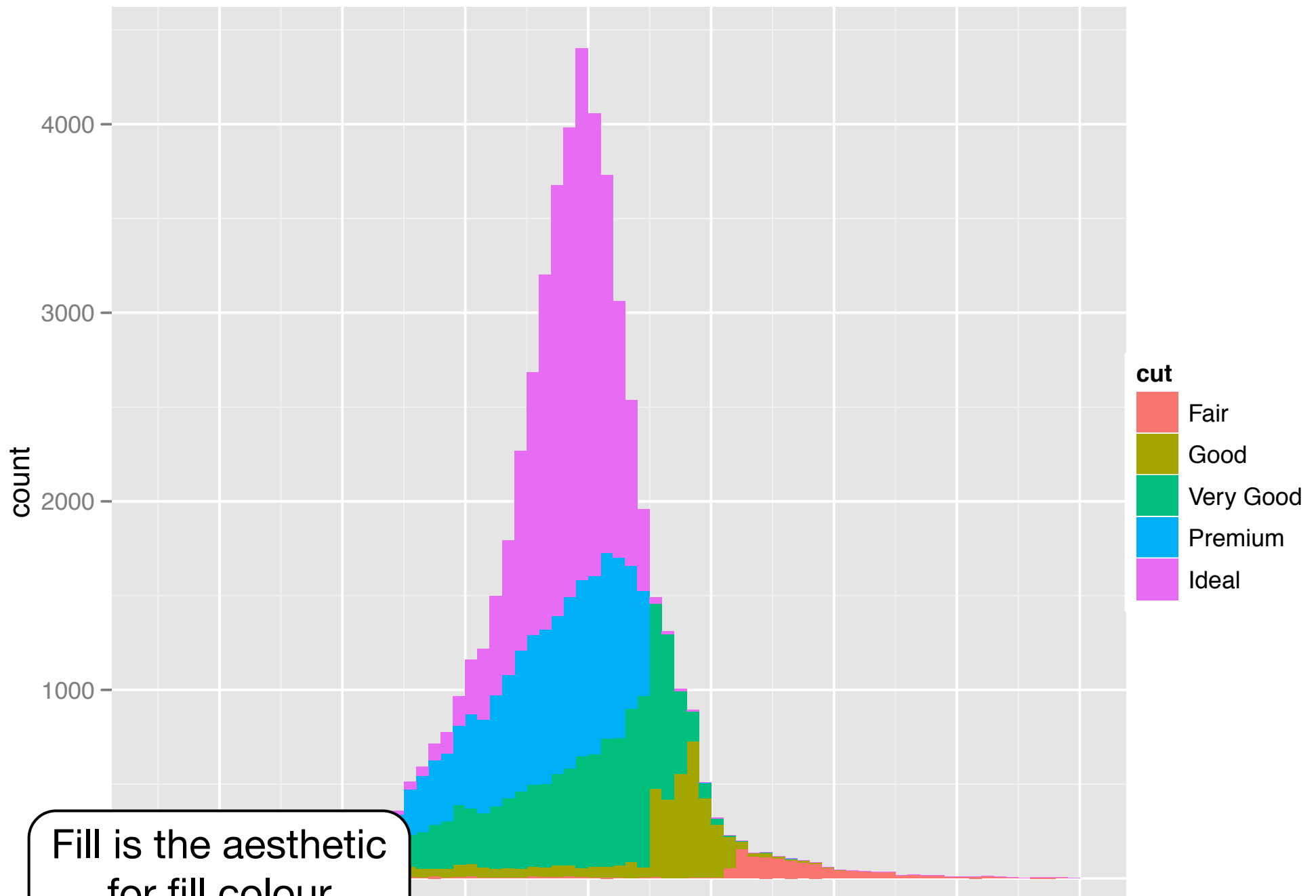
# Additional variables

As with scatterplots can use **aesthetics** or **faceting**. Using aesthetics creates pretty, but ineffective, plots.

The following examples show the difference, when investigation the relationship between cut and depth.

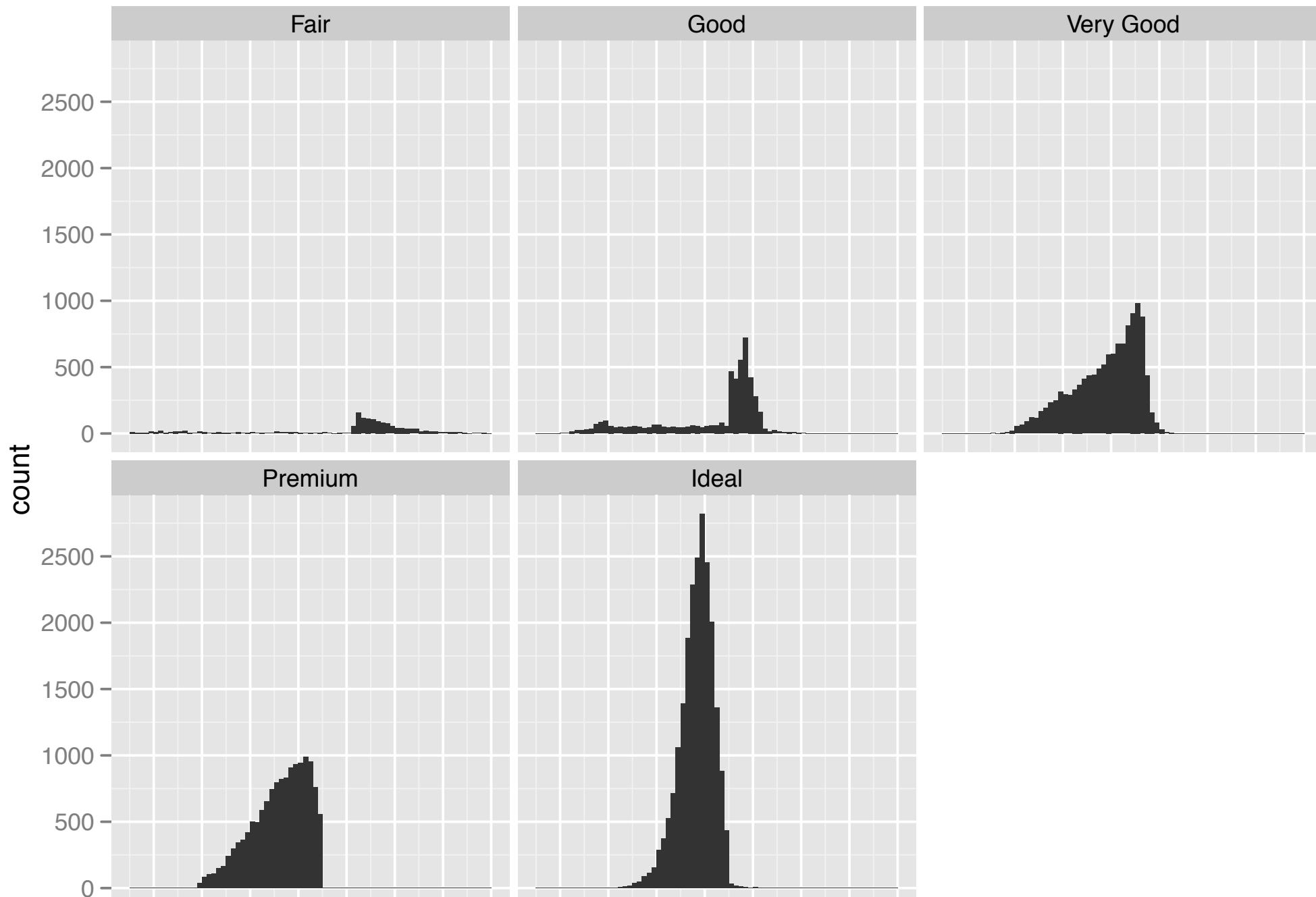


```
qplot(depth, data = diamonds, binwidth = 0.2)
```



Fill is the aesthetic  
for fill colour

```
ggplot(aes(price), data = diamonds, binwidth = 0.2,  
fill = cut) + xlim(55, 70)
```



```
qplot(depth, data = diamonds, binwidth = 0.2) +  
  xlim(55, 70) + facet_wrap(~ cut)
```

# Your turn

Explore the distribution of price.

How does it vary with colour, or cut, and clarity?

# Weighting

```
qplot(cut, data = diamonds, weight = carat)  
qplot(cut, data = diamonds, weight = price)
```

```
# Also useful for pretabulated data  
cuts <- as.data.frame(table(  
  cut = diamonds$cut))  
qplot(cut, weight = Freq, data = cuts)
```

# Scatterplots

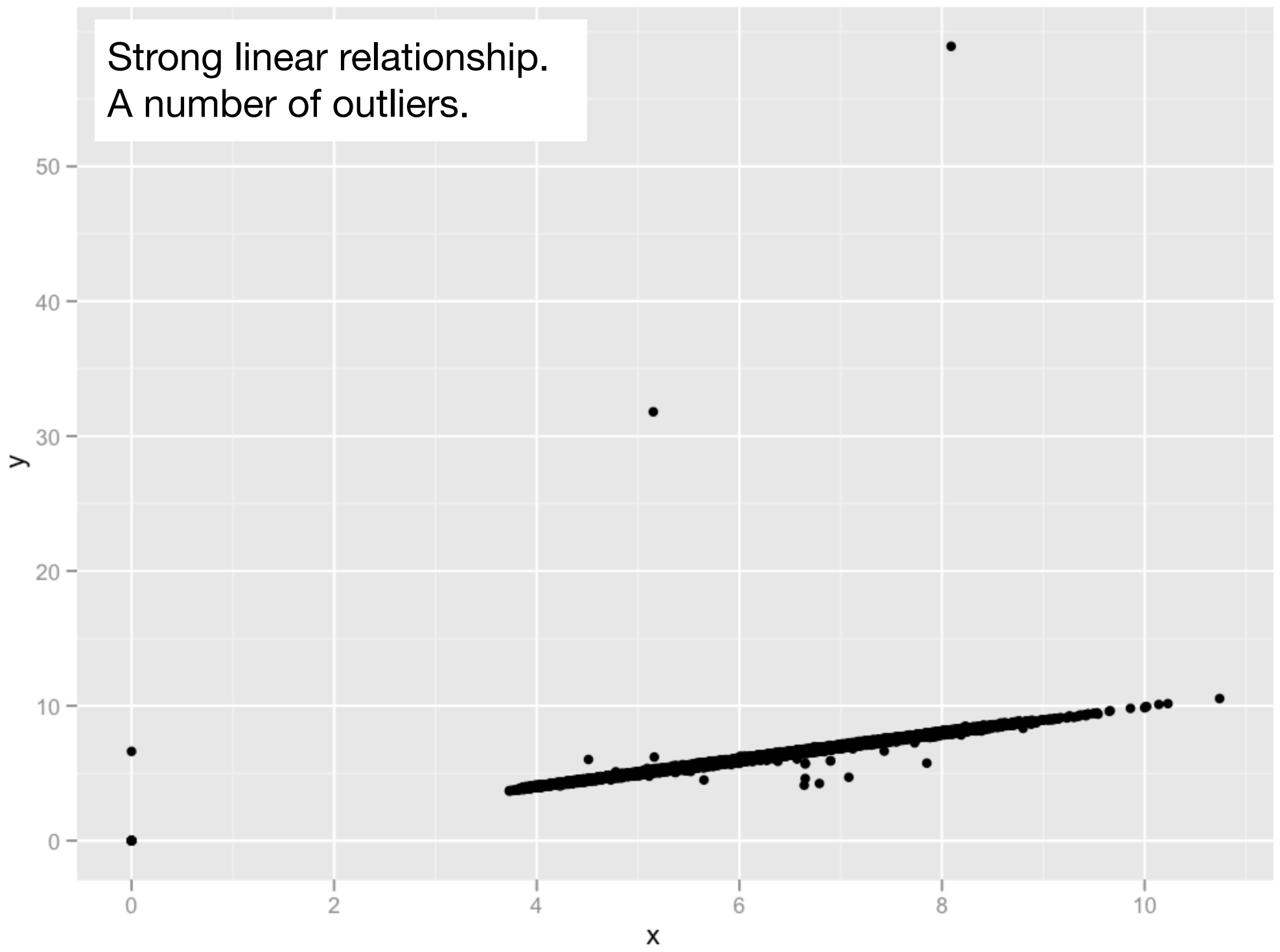


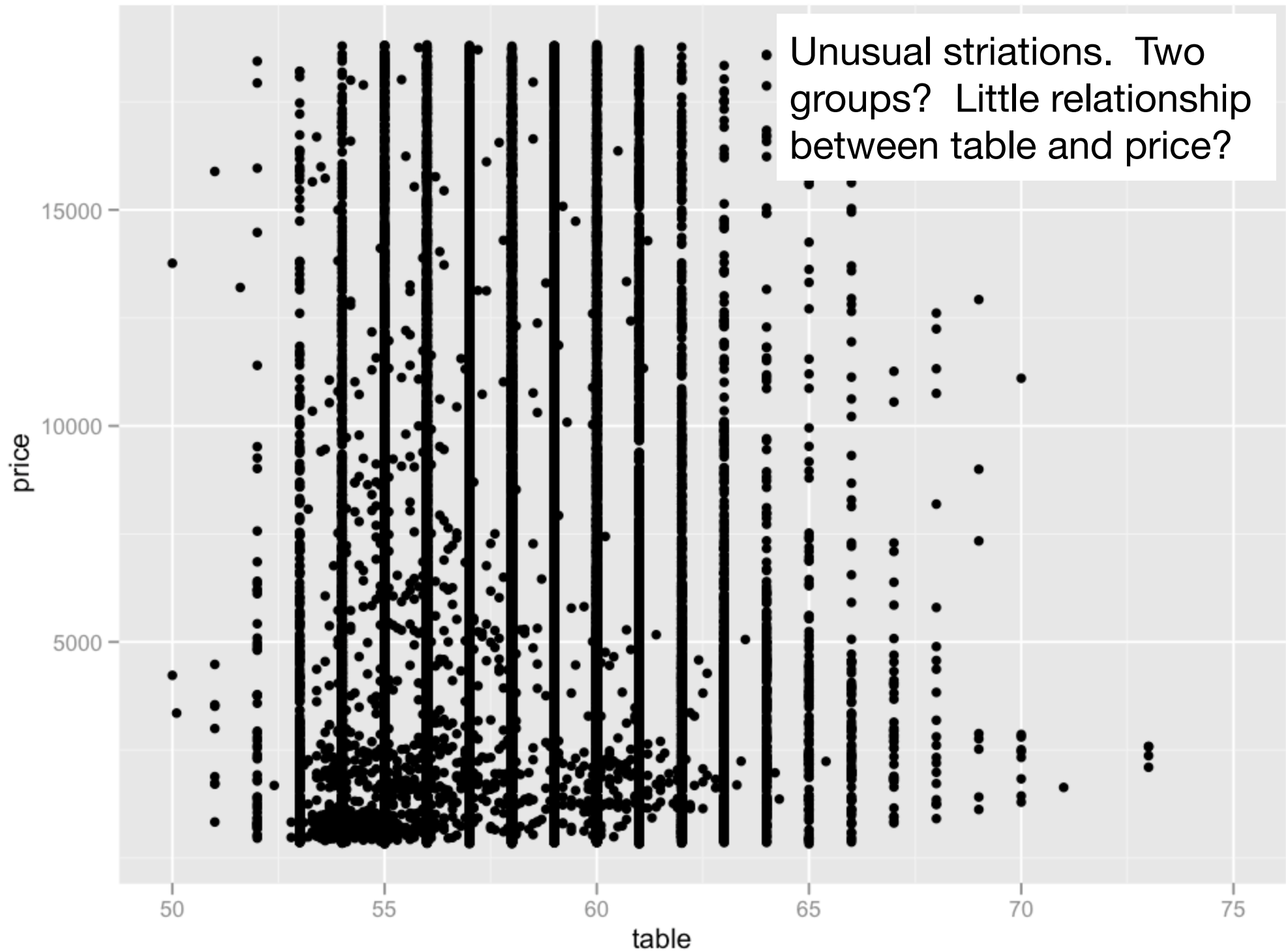
# Revision:

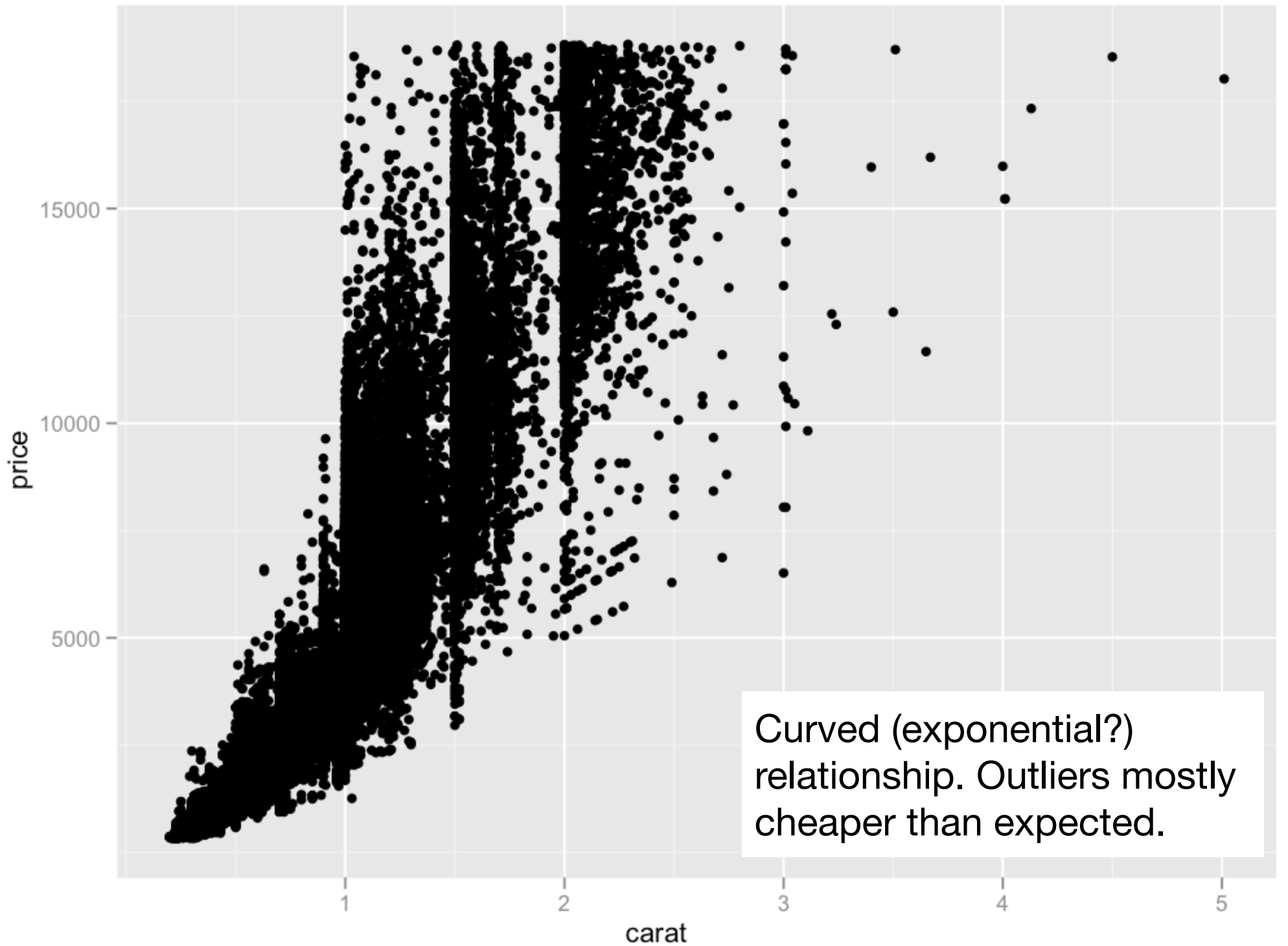
## Interpreting a scatterplot

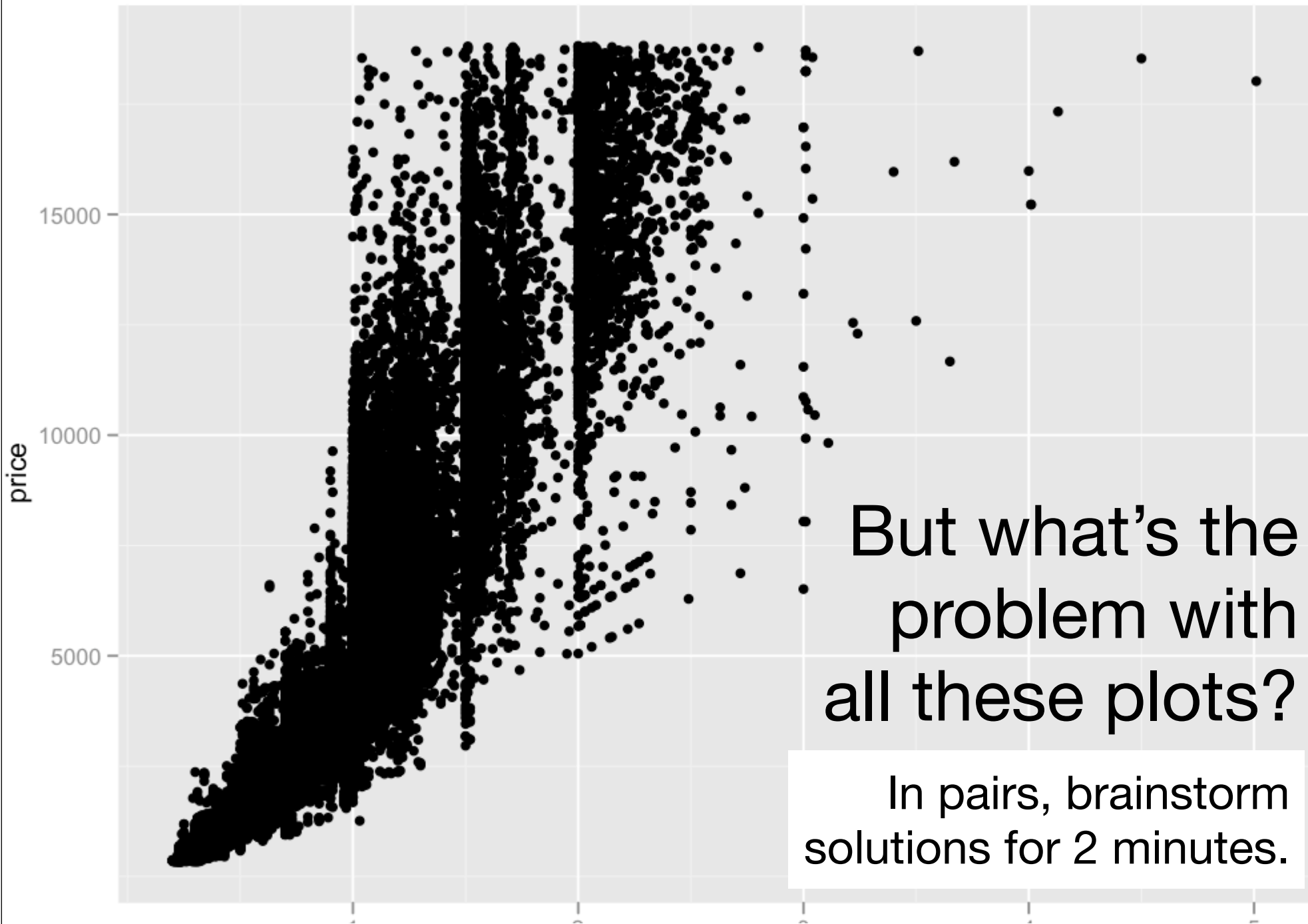
- Big patterns
- Small patterns
- Deviations from the pattern
- Strange patterns

Strong linear relationship.  
A number of outliers.





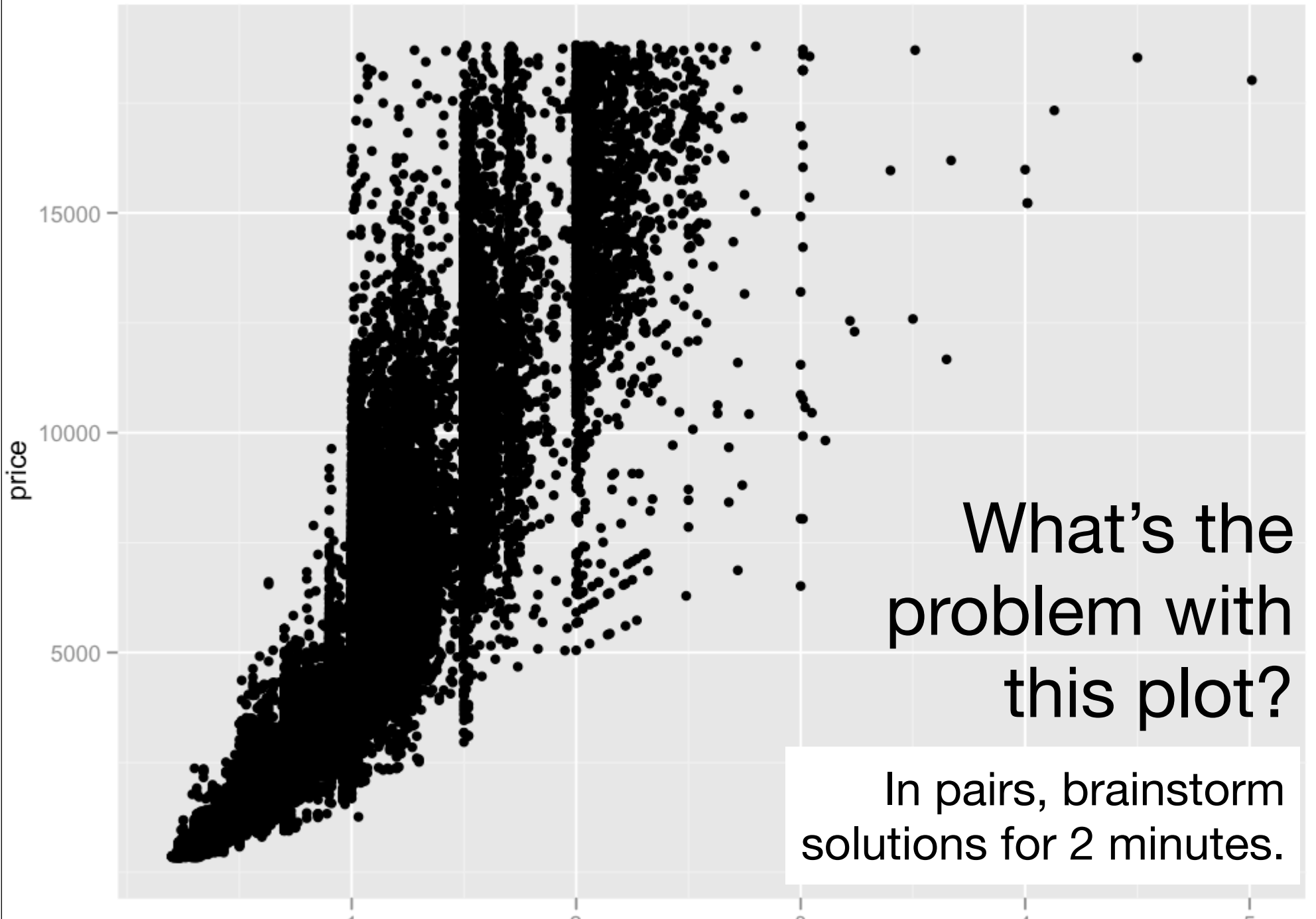




But what's the  
problem with  
all these plots?

In pairs, brainstorm  
solutions for 2 minutes.

```
qplot(carat, price, data = diamonds)
```



What's the  
problem with  
this plot?

In pairs, brainstorm  
solutions for 2 minutes.

```
qplot(carat, price, data = diamonds)
```

# Ideas

If  $x$  discrete, use boxplots.

Use semi-transparent points.

Divide into bins and count number of points in each bin (2d histogram).

Display statistical summary.

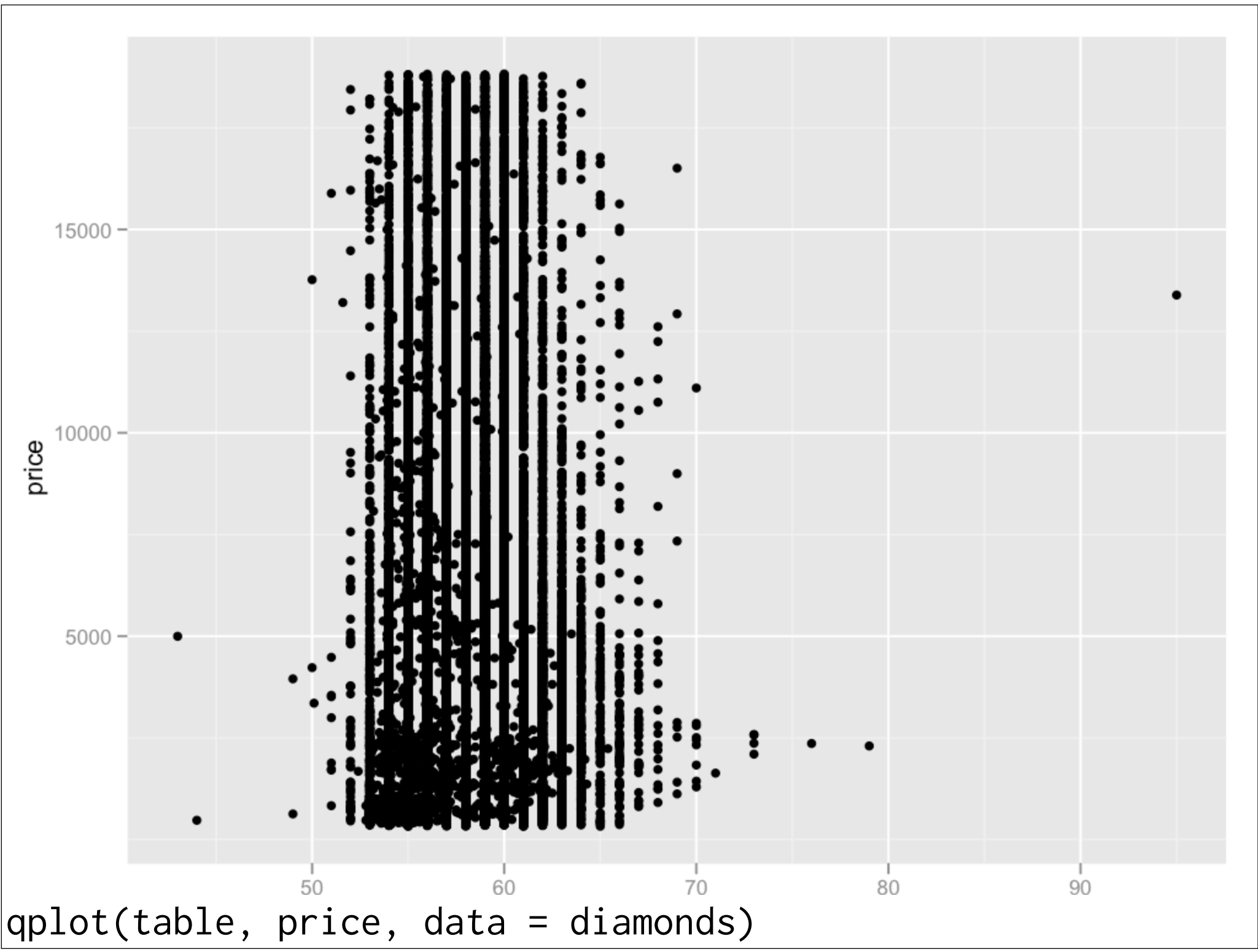
# Box and whisker plots



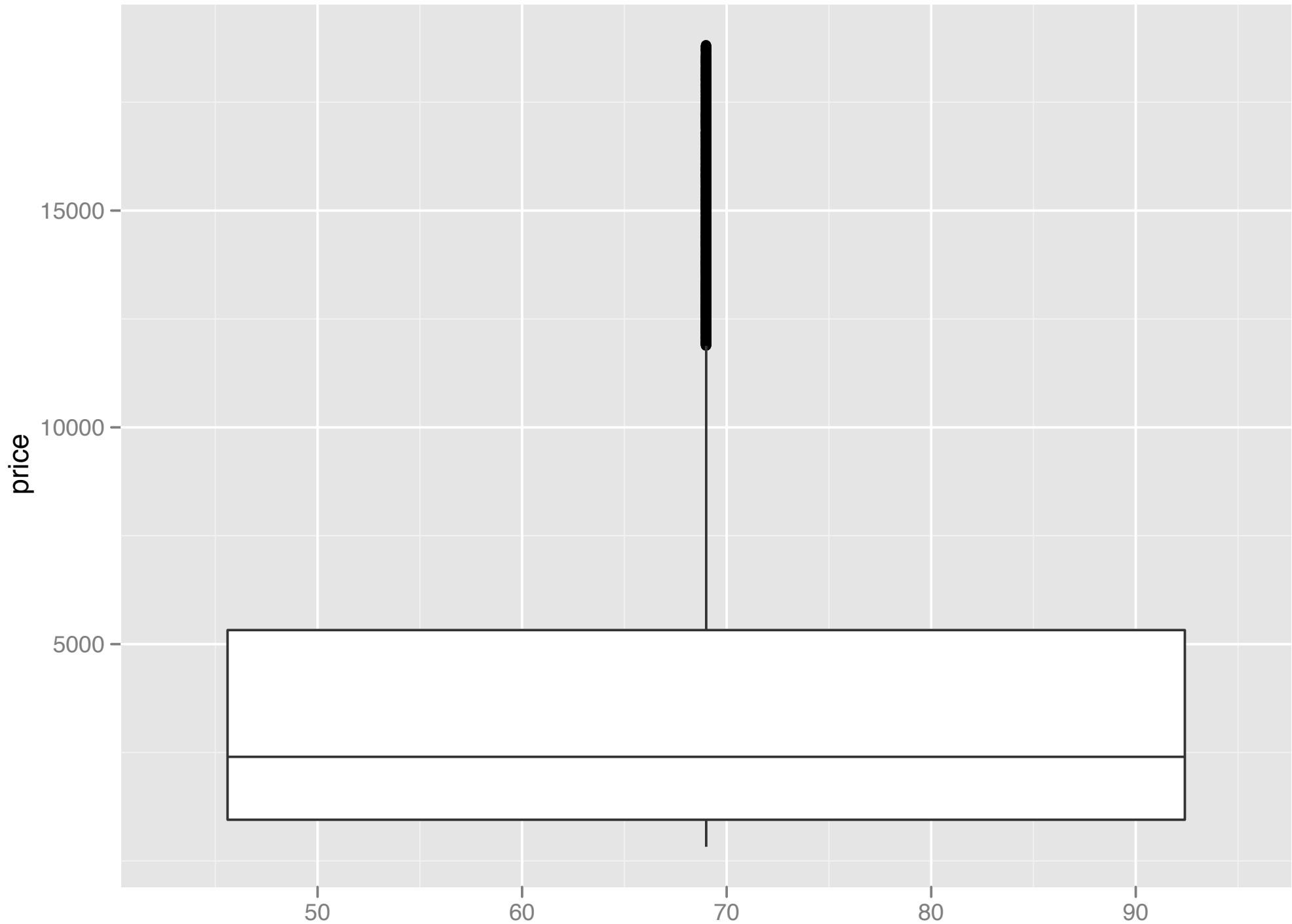
# Boxplots

Less information than a histogram, but take up much less space.

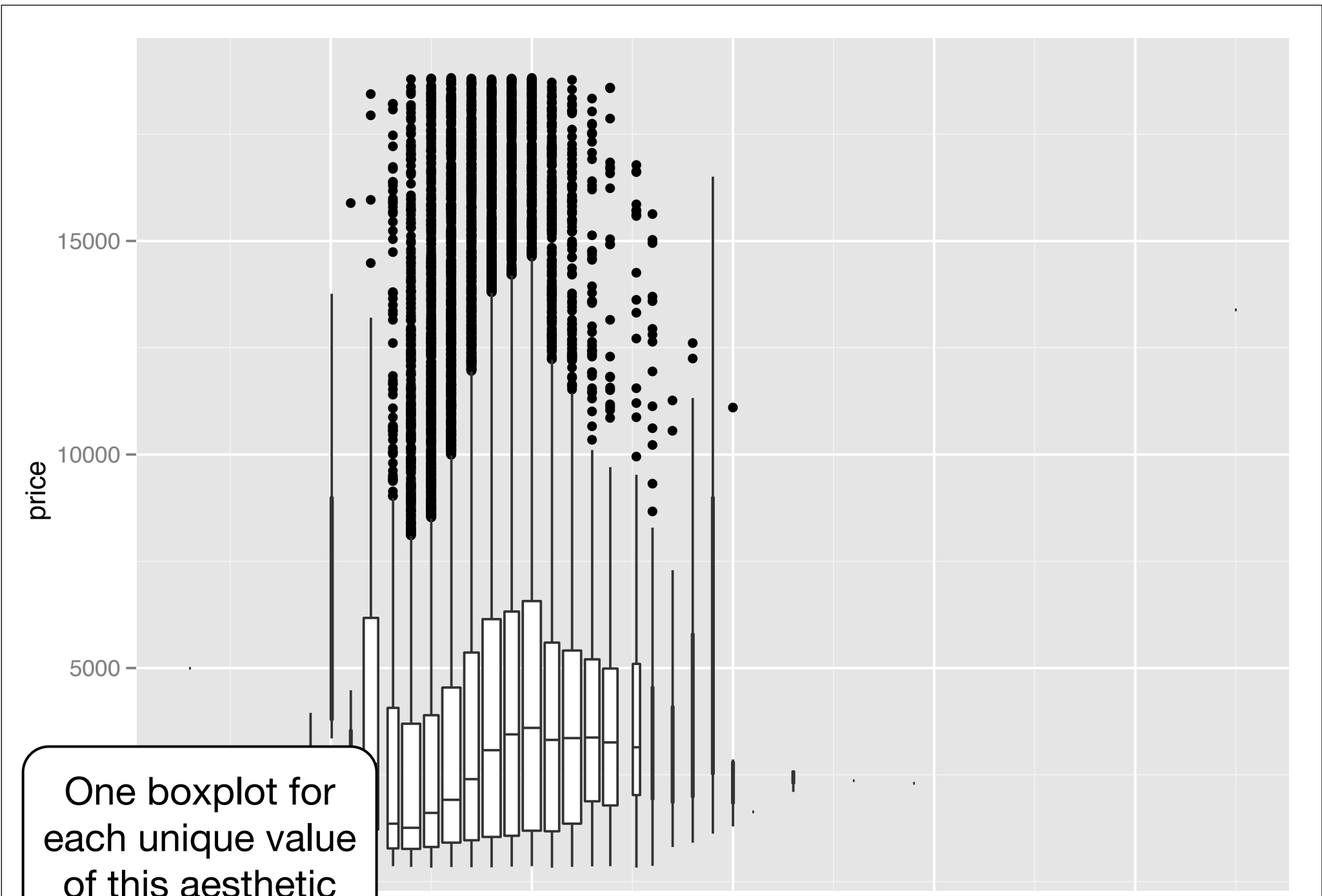
Already seen them used with discrete x values. Can also use with continuous x values, by specifying how we want the data **grouped**.



```
qplot(table, price, data = diamonds)
```



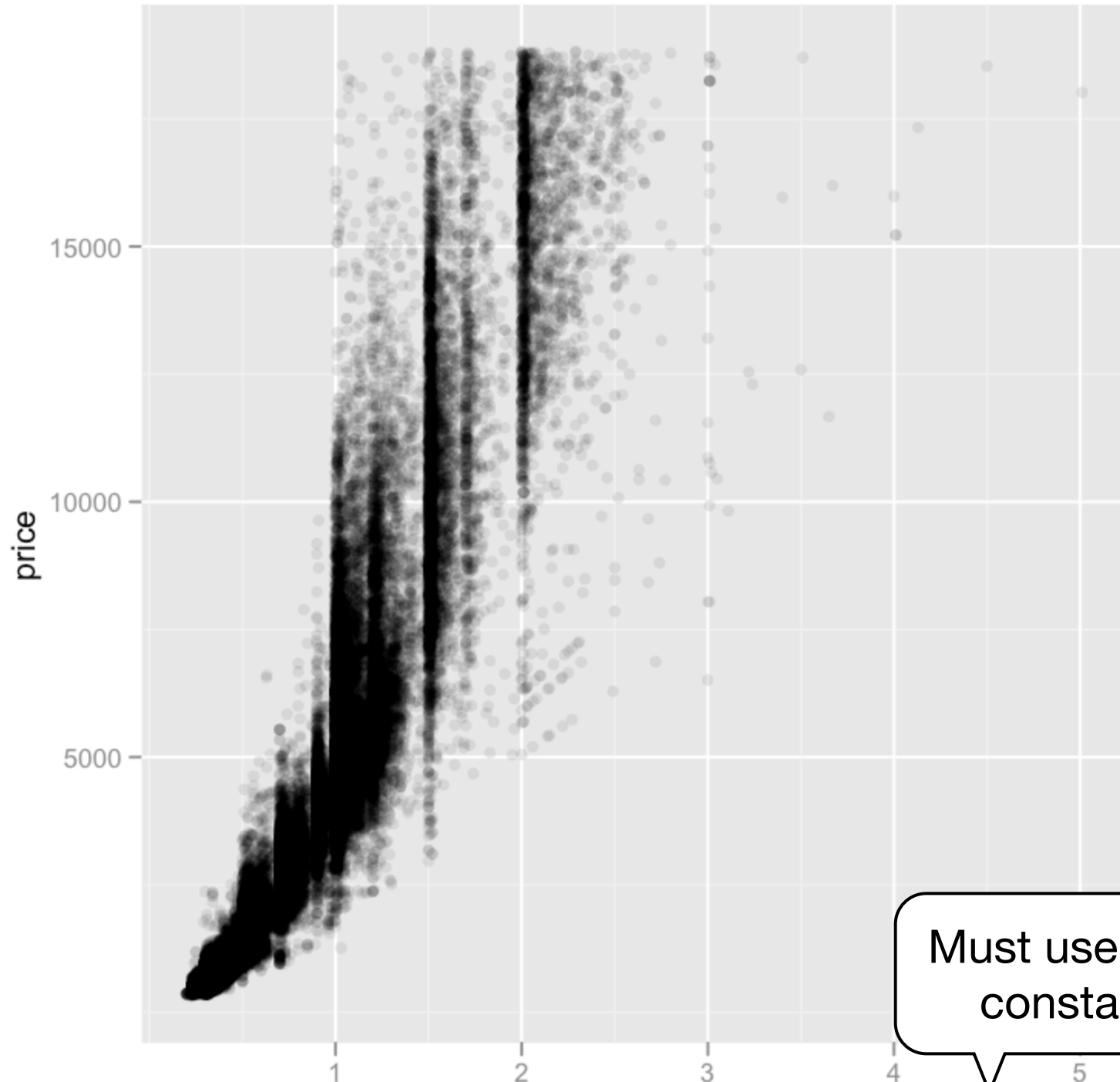
```
qplot(table, price, data = diamonds, geom = "boxplot")
```



One boxplot for each unique value of this aesthetic

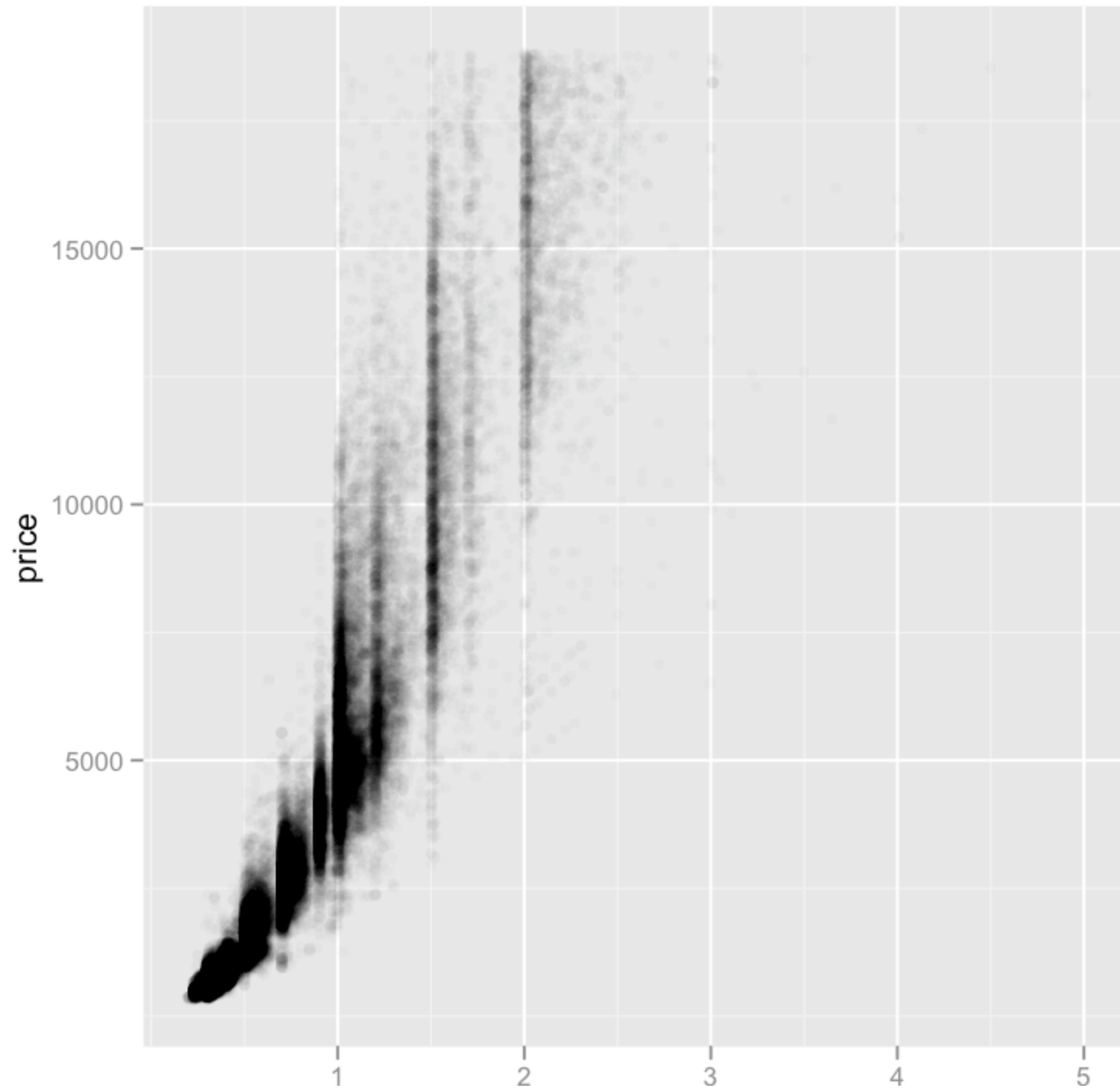
```
ggplot(diamonds, aes(price, price, data = diamonds, geom = "boxplot", group = round(table)))
```

# Alpha blending

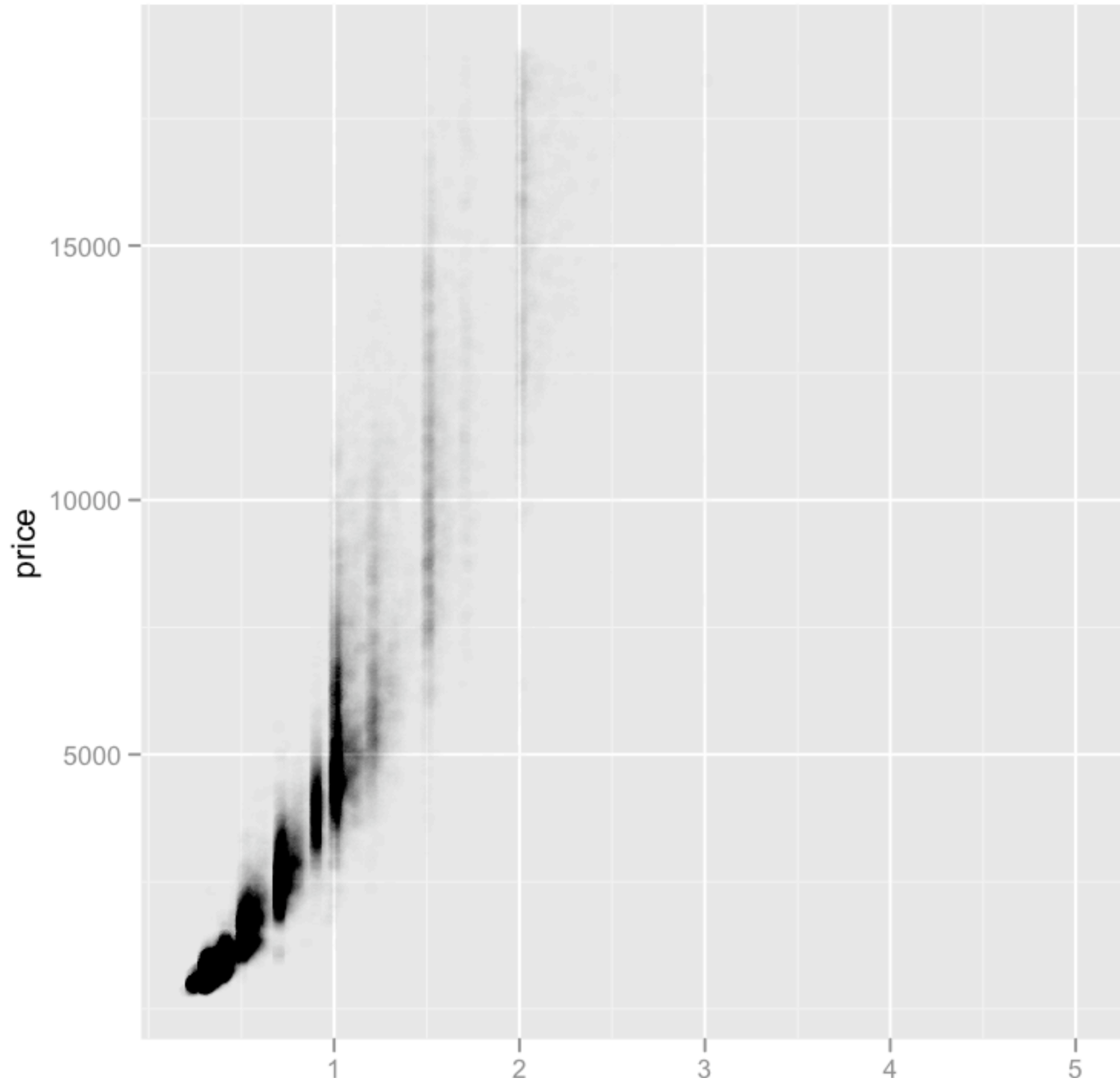


Must use I() for constants

```
qplot(carat, price, data = diamonds, alpha = I(1/10))
```



```
qplot(carat, price, data = diamonds, alpha = I(1/50))
```



```
qplot(carat, price, data = diamonds, alpha = I(1/250))
```



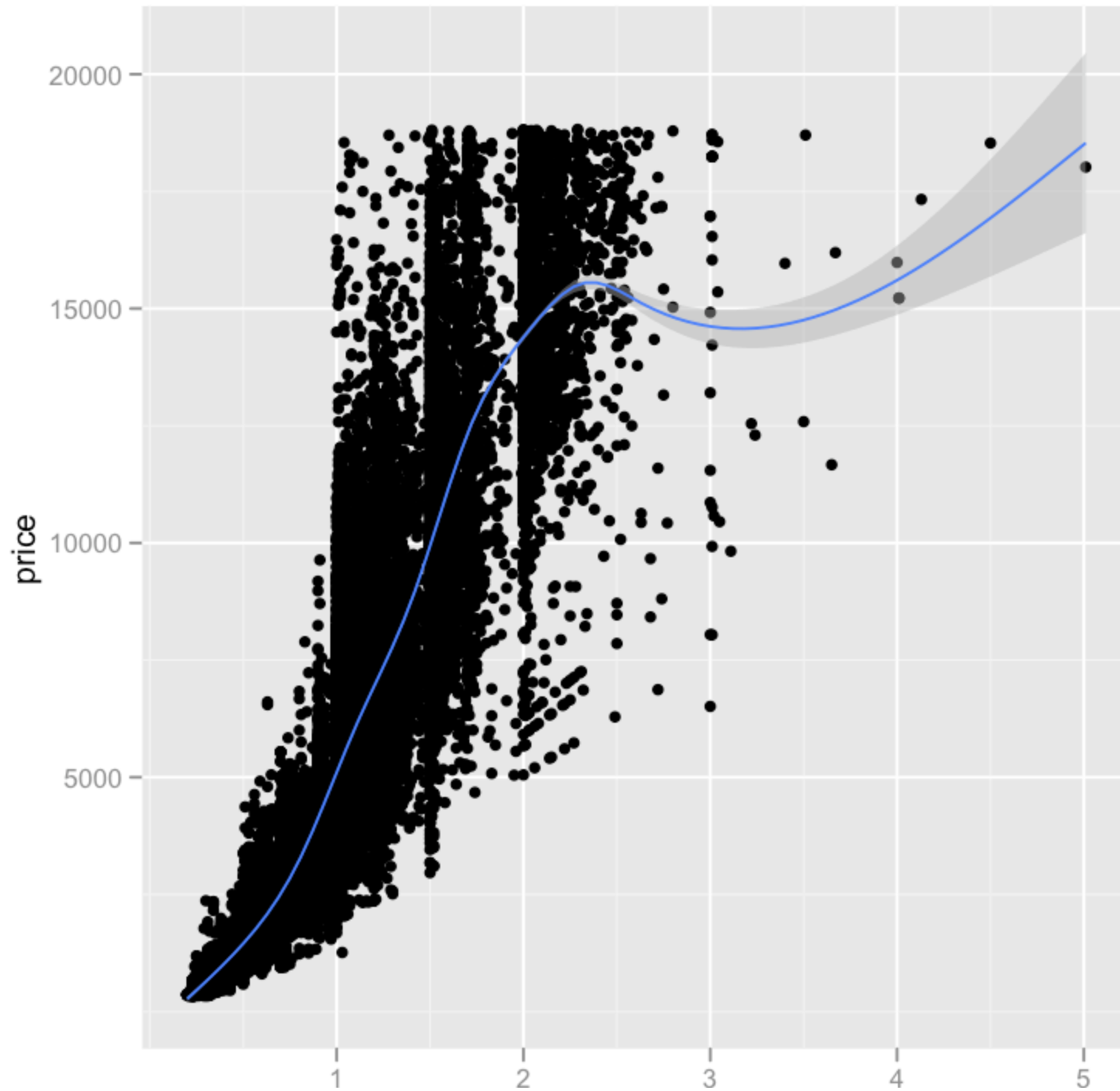
```
qplot(carat, price, data = diamonds,  
      colour = I(alpha("black", 1/255)))
```

```
qplot(carat, price, data = diamonds, geom = "bin2d")  
qplot(carat, price, data = diamonds, geom = "bin2d",  
      bins = 100)
```

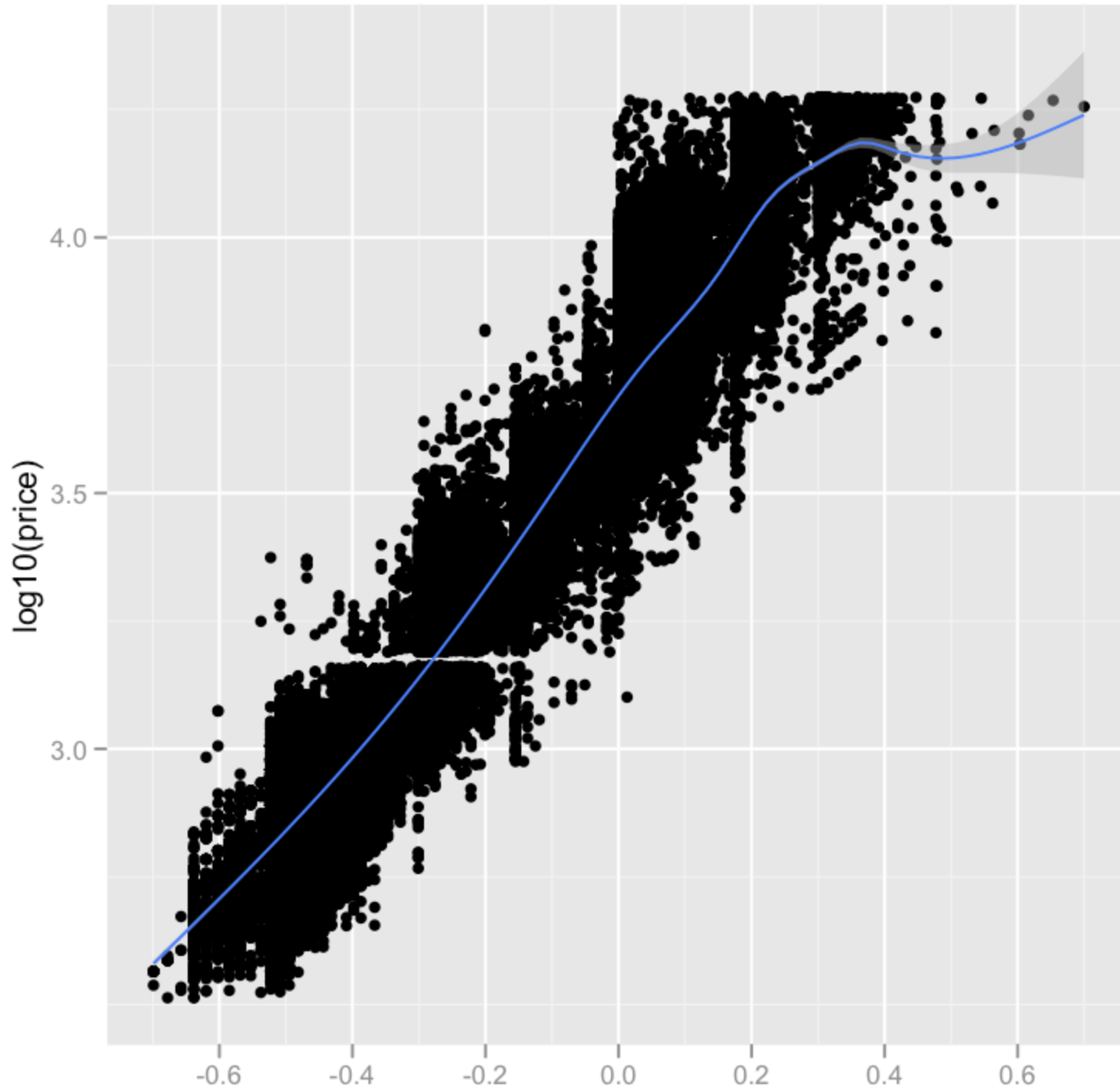
```
qplot(carat, price, data = diamonds, geom = "hex")
```

```
qplot(carat, price, data = diamonds) + geom_smooth()
```

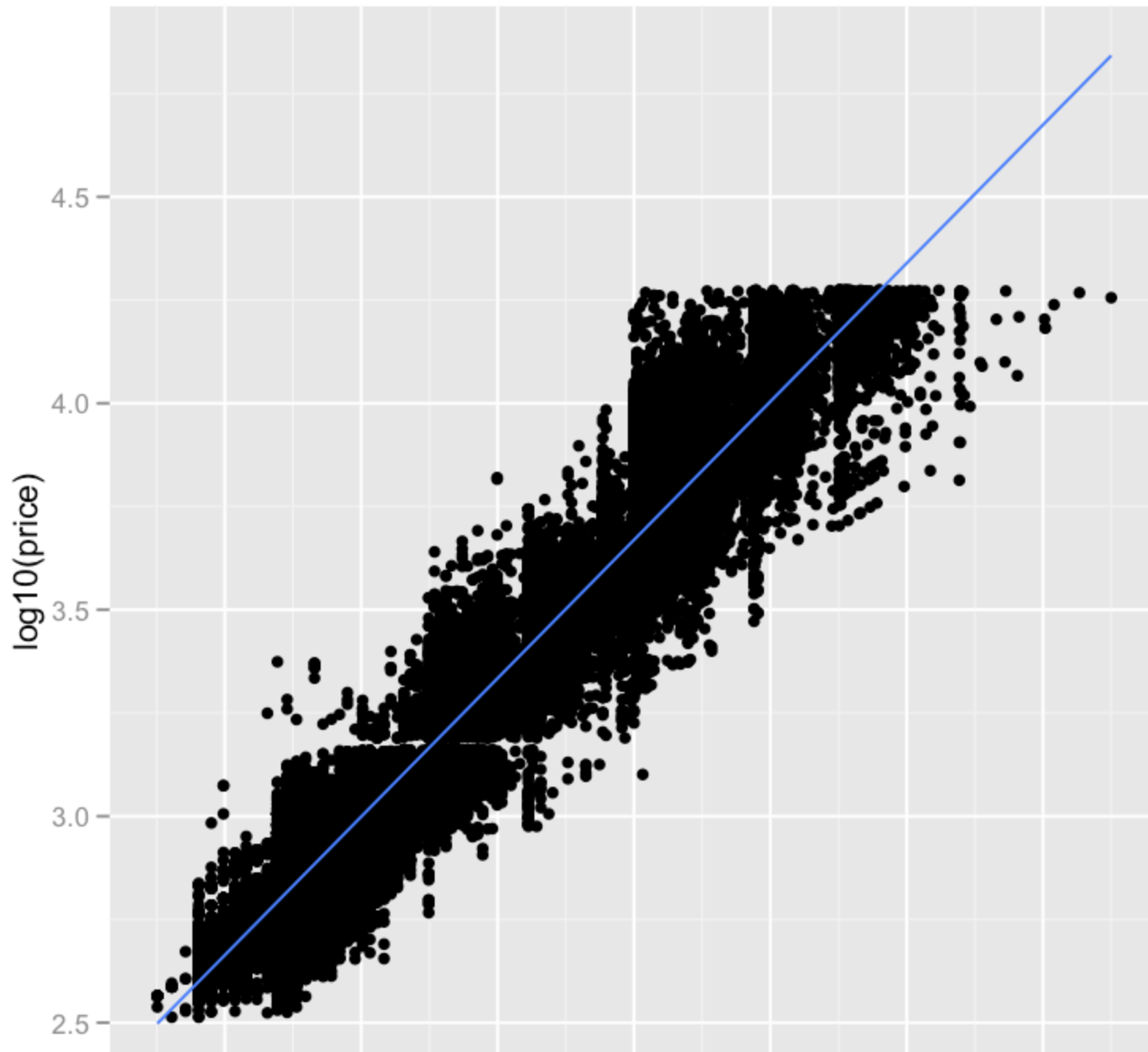
# Statistical summary



```
qplot(carat, price, data = diamonds) + geom_smooth()
```



```
qplot(log10(carat), log10(price), data = diamonds) + geom_smooth()
```



```
qplot(log10(carat), log10(price), data = diamonds) +  
  geom_smooth(method = "lm")
```

**2d bins**

```
# Very basic cleaning
```

```
diamonds$x[diamonds$x == 0] <- NA
```

```
diamonds$y[diamonds$y == 0] <- NA
```

```
diamonds$y[diamonds$y > 12] <- NA
```

```
qplot(x, y, data = diamonds)
```

```
qplot(x, y, data = diamonds, geom = "bin2d")
```

```
qplot(x, y, data = diamonds, geom = "hex")
```

```
qplot(x, y, data = diamonds, geom = "bin2d", bins = 100)
```

```
qplot(x, y, data = diamonds, geom = "hex", bins = 100)
```

```
# Zoom in
```

```
qplot(x, y, data = diamonds, geom = "bin2d", bins = 100) +  
  xlim(4,7) + ylim(4,7)
```

```
qplot(x, y, data = diamonds, geom = "bin2d", bins = 100) +  
  xlim(4,5) + ylim(4,5)
```

```
qplot(x, x / y, data = diamonds,  
      geom = "bin2d")  
qplot(x, log(x / y), data = diamonds,  
      geom = "bin2d")
```

```
clean <- subset(diamonds, abs(log(x / y)) < 0.1)
```

```
qplot(x, log(x / y), data = clean, geom = "bin2d")  
qplot(x, log(x / y), data = clean, geom = "bin2d",  
      bins = 80)
```

What would be a good name for  $\log(x / y)$ ? What other variable might you create to go with it?

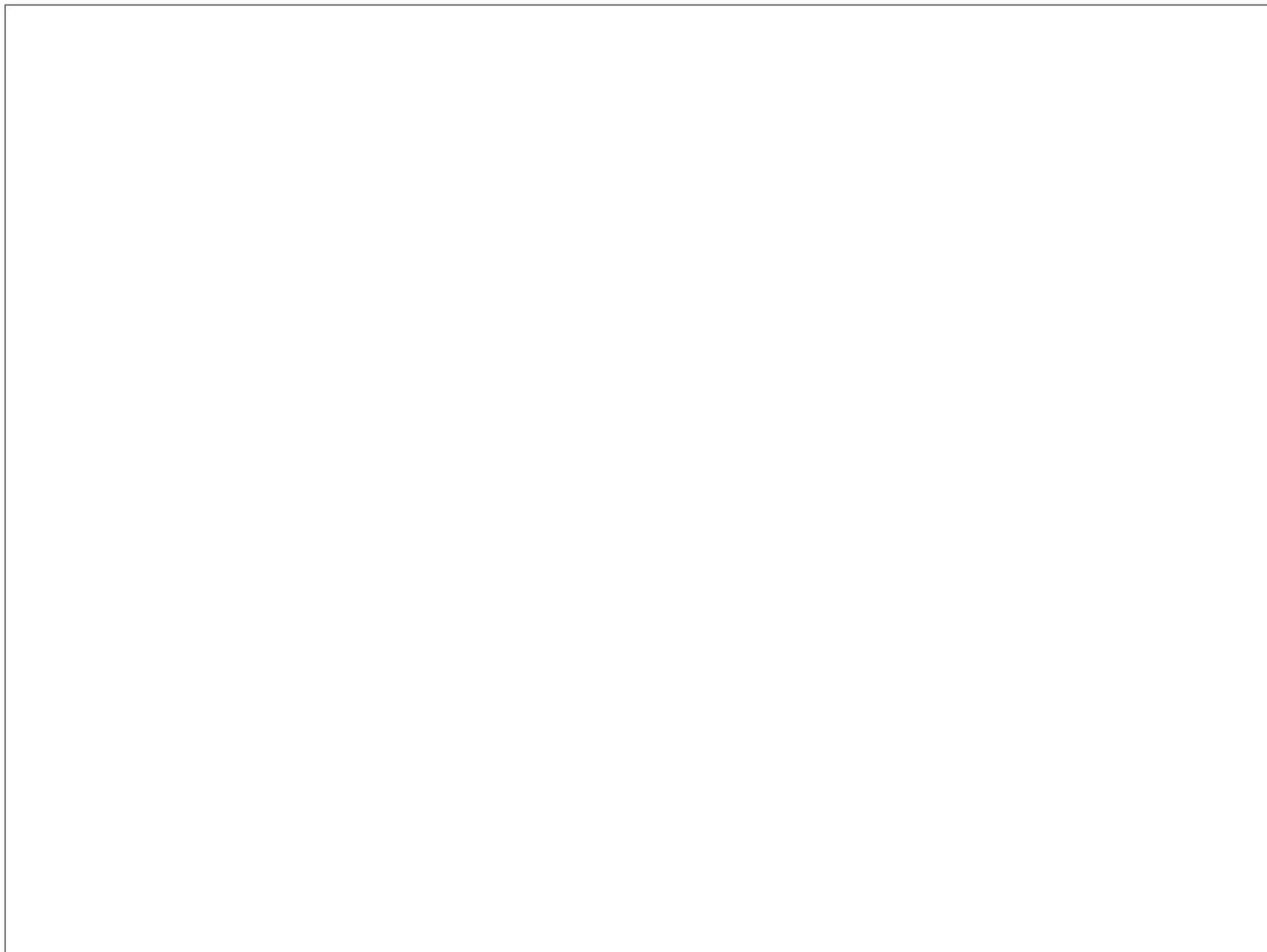


# Your turn

Continue to explore the relationship between  $x$ ,  $y$ ,  $z$  and carat. Create new variables as necessary.

You might also want to do more cleaning.

Some good ideas here: <http://www.diamondhelpers.com/fivesteps/4-certified-diamonds.shtml>



This work is licensed under the Creative Commons Attribution-Noncommercial 3.0 United States License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/3.0/us/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.