DATA TO INSIGHT: AN INTRODUCTION TO DATA ANALYSIS

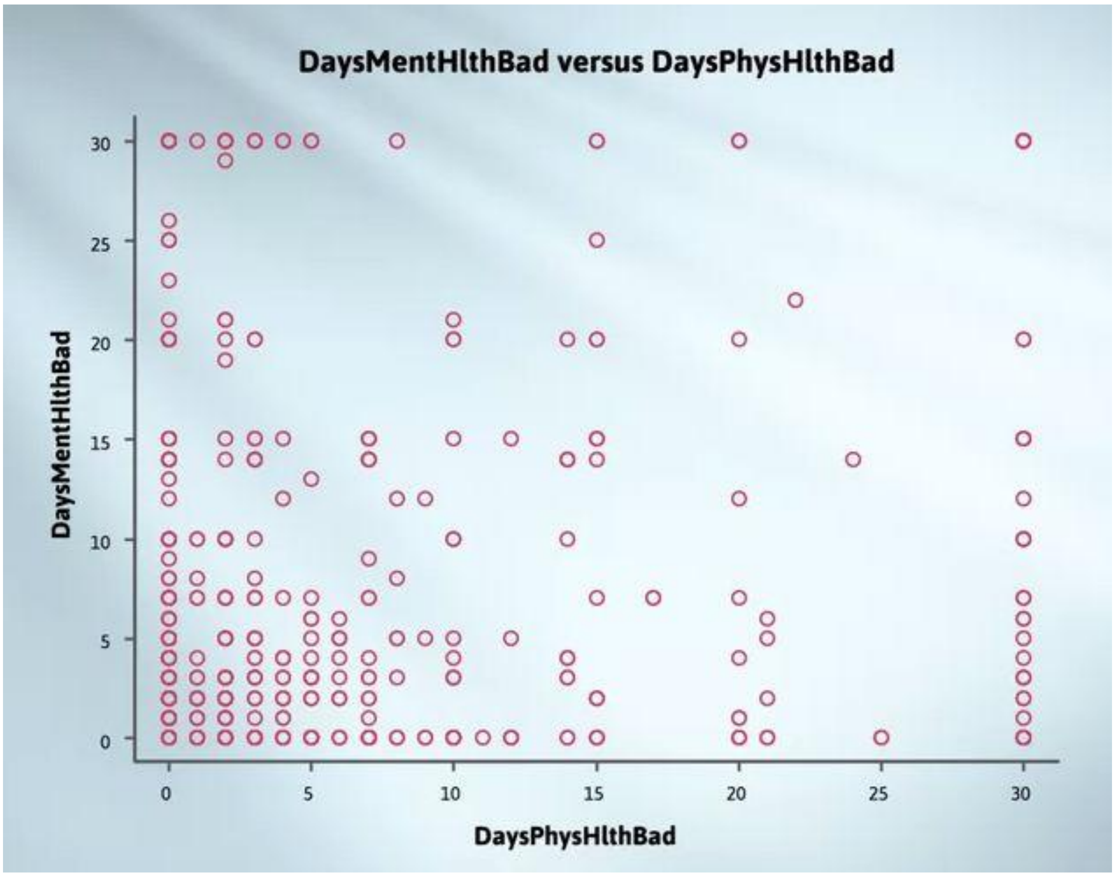
THE UNIVERSITY OF AUCKLAND

WEEK 4

4.6 OVERCOMING PERCEPTUAL PROBLEMS by Chris Wild

Welcome back. In this video, we'll move on to problems that limit our ability to see data structures properly and ways of overcoming those problems. The main problems we address are caused by overprinting, or having a large data set, or both.

We'll start with overprinting, which is just as its name suggests. I'm using our NHANES 1000 file and plotting the average number of days in a month where the respondents' mental health was bad versus the average number of days where their physical health was bad.

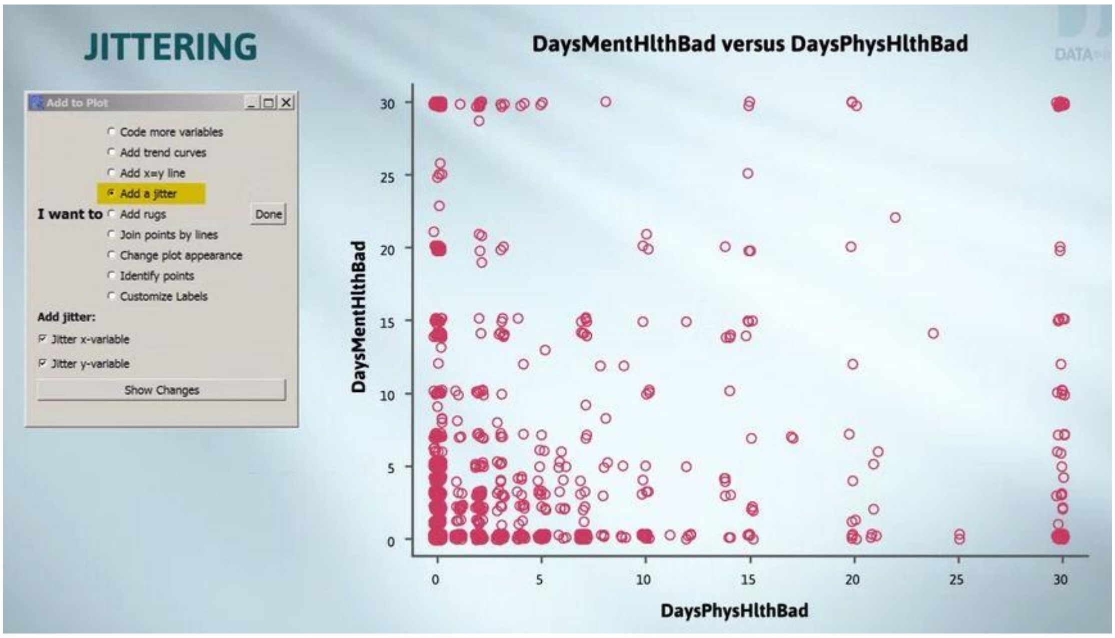


The data seems to be spread all over the plotting window. You'd guess that the median number of bad physical health days might be about 7, say, and similarly for bad mental health days. But there are 747 people represented here. (The rest were missing.) We can only see a small fraction of the 747 points because large numbers of people gave the same combination of values.

When we plot two different people with the same set of values, the second point we plot sits directly on top of the first point we plotted. This is called overprinting. When there's exact overprinting, we can't see how many points there are sitting at any given position. It always looks like there is just one.

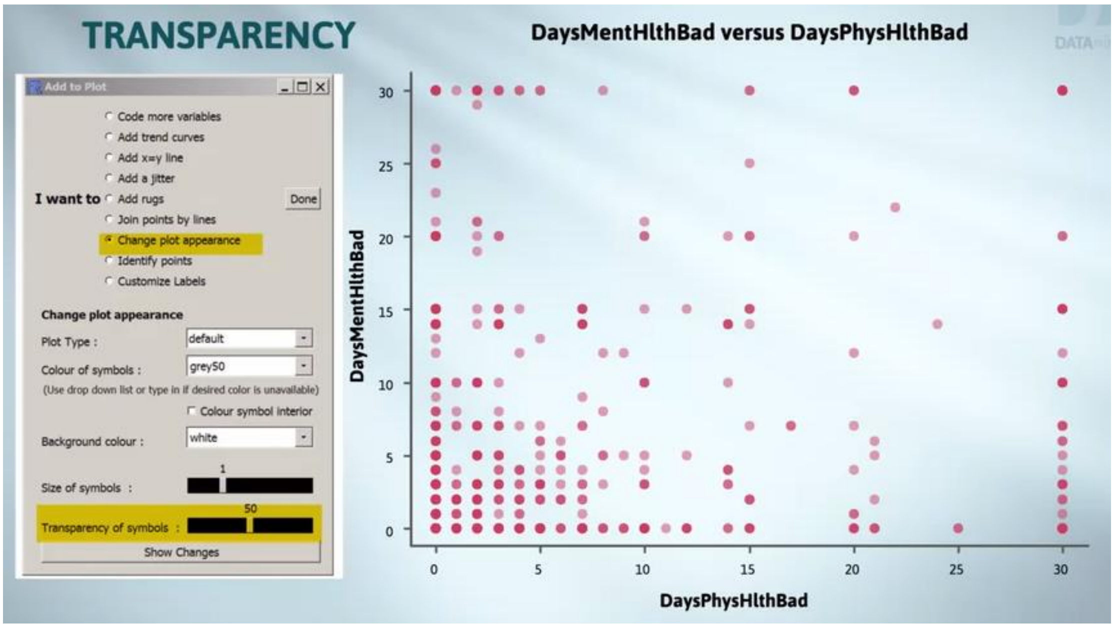
The worst case here is the zero-zero position, zero bad mental health days and 0 bad physical health days. In fact, there are 302 people sitting in the zero-zero position. That's 40% of them, but you'd never know it.

So how can we get around this? There are two main methods, jittering and transparency. First we'll look at jittering.



Add jitter:
☑ Jitter x-variable
☑ Jitter y-variable

With jittering, we add a little bit of random horizontal or vertical displacement to each point to break them apart so we can see where multiple points are sitting. This helps a lot. But this example is so extreme, we still see no indication that 40% of the data is sitting at zero-zero in the bottom left-hand corner.



TRANSPARENCY

Add to Plot

I want to
- ○ Code more variables
- ○ Add trend curves
- ○ Add x=y line
- ○ Add a jitter
- ○ Add rugs
- ○ Join points by lines
- ● Change plot appearance
- ○ Identify points
- ○ Customize Labels

Done

Change plot appearance

Plot Type : default

Colour of symbols : grey50

(Use drop down list or type in if desired color is unavailable)

□ Colour symbol interior

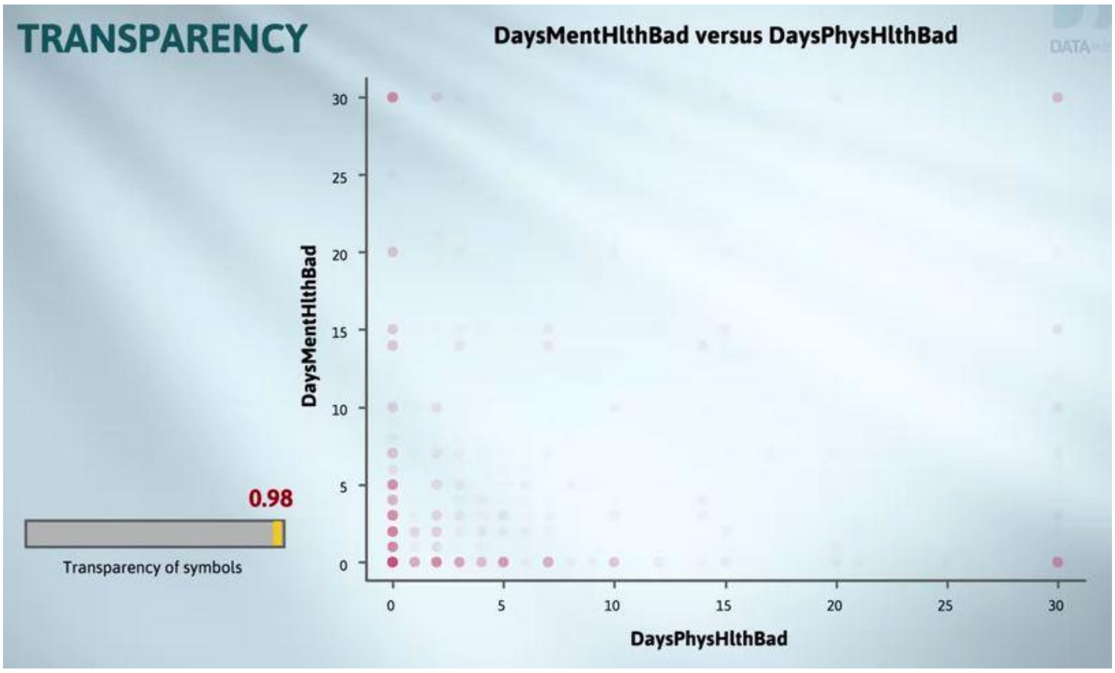
Background colour : white

Size of symbols : 1

Transparency of symbols : 50

Show Changes

We'll now switch to transparency. We're using discs that are semi-transparent. This way, as the number of discs sitting on top of one another gets bigger, the colour gets darker. The faintest points you can see here represent a single observation. We can vary the degree of transparency.



TRANSPARENCY

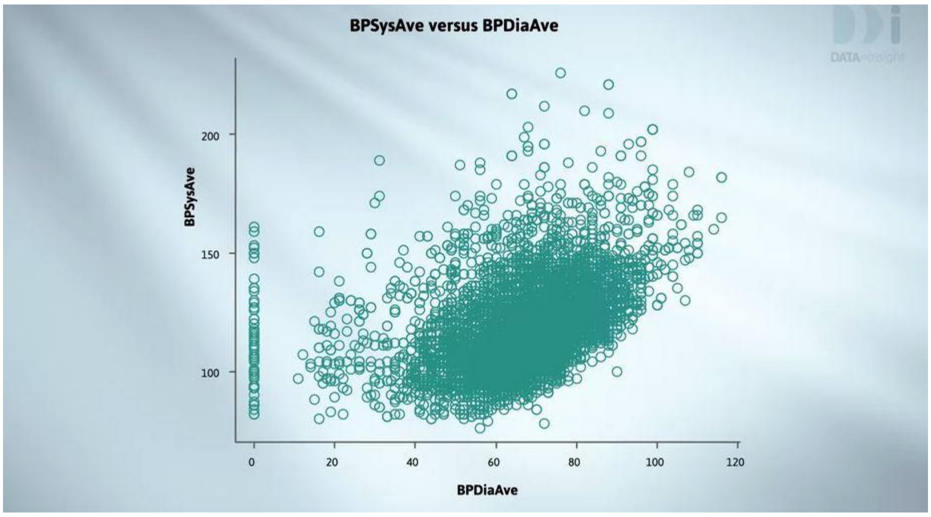
0.98

Transparency of symbols

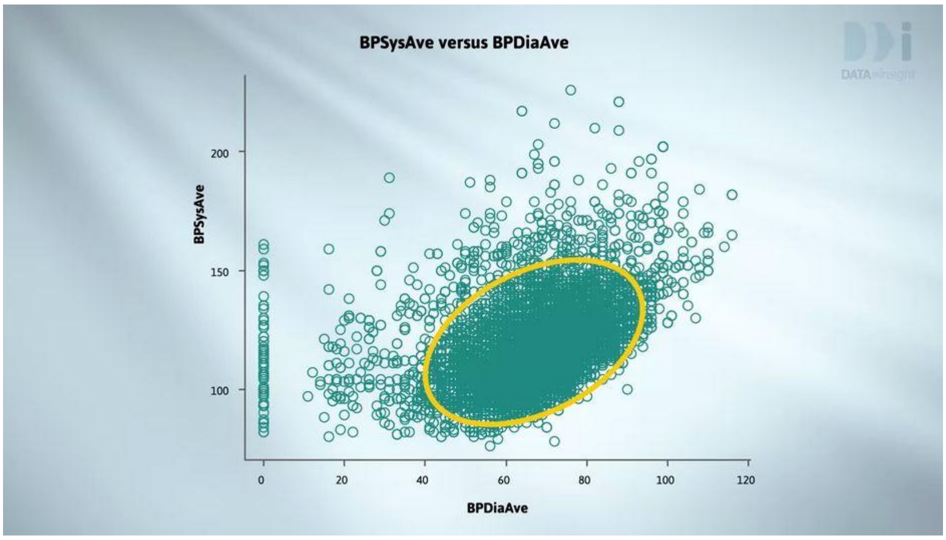
Because this example is so extreme(40% of the observations sitting on a single position) by making the transparency extreme enough to show that there are more points at zero-zero than anywhere else, I have almost lost sight of the positions that represent one point.

This isn't a problem because we don't just look at a single image. We keep varying the transparency setting until we have a good idea of what's going on. Any single image can mislead us.

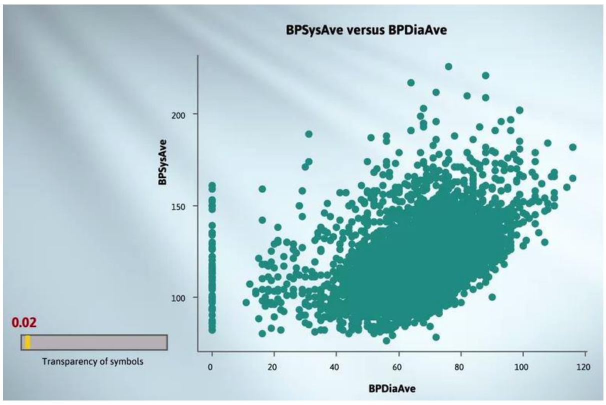
We now move on to problems with large data sets where transparency can again be invaluable. When the doctors take your blood pressure, they give you two values. They'll say something like, "Your blood pressure is 120 over 70". The first number is called the systolic blood pressure, and the second is called the diastolic blood pressure.

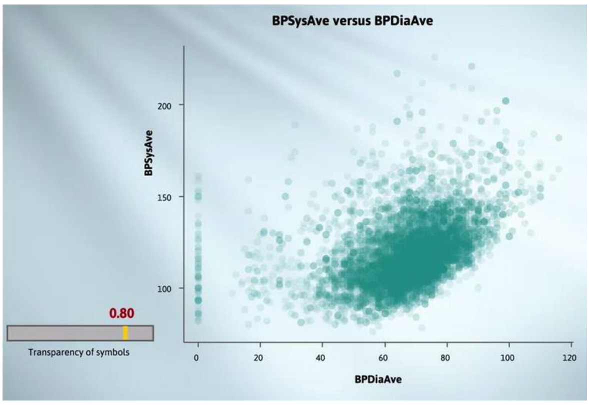


This is a plot of systolic versus diastolic blood pressures. The systolic blood pressure is running vertically and diastolic horizontally. They're plotted for our full NHANES data set, all 10,000 observations. The dominant visual features are a column of values on the left and a large rugby-ball-shaped splodge in the lower central region with a scattering of points around it.

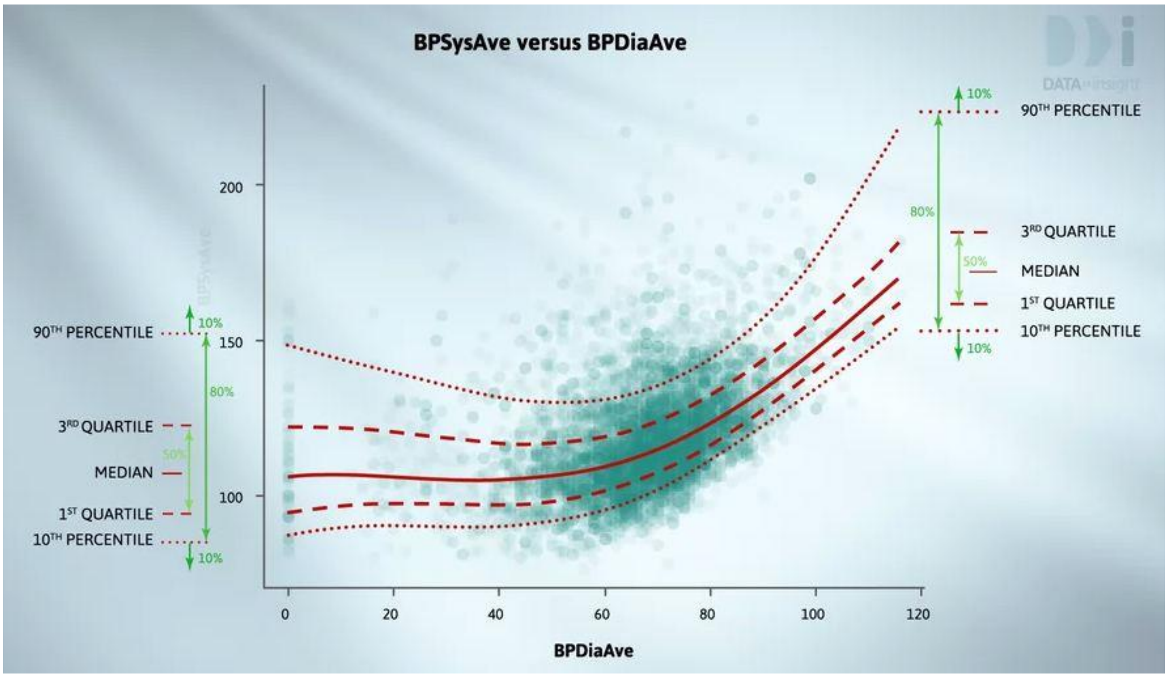


We can't see any detail inside the ball, and the values scattered around it look hugely more common than they really are. Transparency is a great tool for correcting these visual miscues.



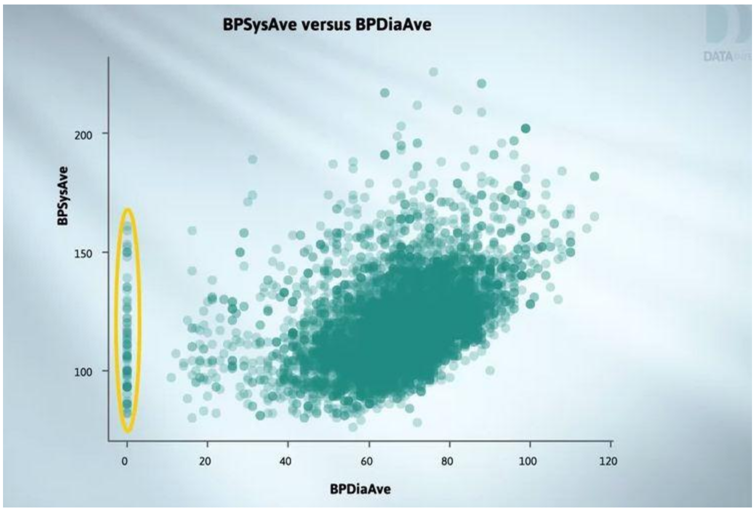


Here, as we increase the transparency, we can see more and more clearly into where the bulk of the data is, as opposed to where the unusual and comparatively rare values are.



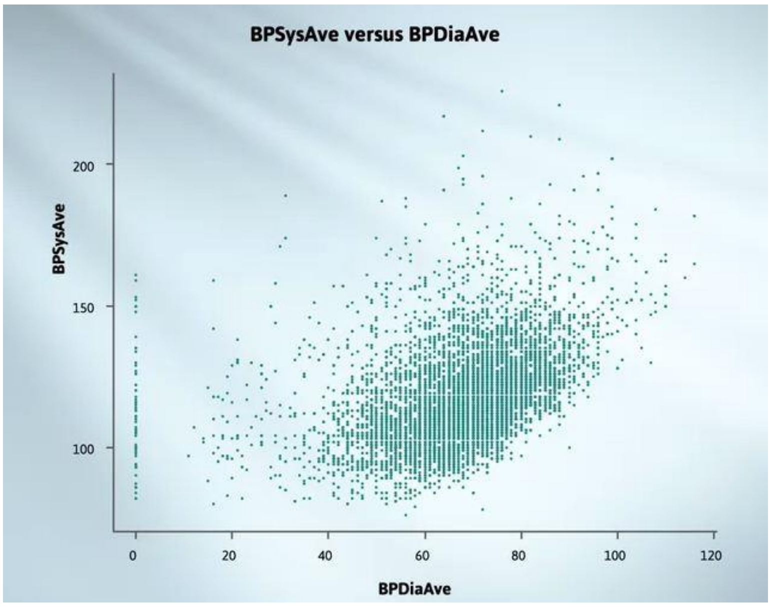
In this plot, we've added running quantiles to the graph. The curve labelled 50% is a smoothed trend line that gives us an idea what the median systolic blood pressure is for a given diastolic blood pressure. The dashed 25% and 75% curves tell us where the quartiles are. And outside them, we have the dotted 10% and 90% curves. So we expect that at any setting, 90% of the people are falling below the upper dotted curve.

What's most surprising to me is how low on the plot the 90th percentile curve comes. If I had to draw something to see if we're at the top 10% from the bottom 90%, I would have wanted to draw it quite a lot higher. So we really do need aids like this to see properly into scatterplots for large data sets.



Let's return now to the column of values above zero. It really is possible to repeatedly give diastolic blood pressure readings of zero and not be dead. You can Google to find out more.

In this next version, we've made the plotting points very small. This helps us to see gaps between data points, usually caused by rounding. The blood pressures here are all rounded to whole numbers.



To summarise, using no transparency emphasises what's happening on the edges of the data, while high transparency lets us look at the comparative density of the observations in the bulk of the data. Using very small plotting symbols is good for seeing discreteness or separateness.

Scatterplots with many thousands of individual points can be quite slow to draw, especially if you're using transparency. That's why iNZight defaults to plots that look like this when the sample size exceeds about 2,000 because they're much faster to draw. But you can always force it to produce standard scatterplots.

That brings us to the end of this video.