perceptual edge

The DataVis Jitterbug: Let's Improve an Old Dance

Stephen Few, Perceptual Edge
Visual Business Intelligence Newsletter
April/May/June 2017

Graphs sometimes suffer from a problem called *over-plotting*, which is when multiple data-encoding objects (e.g., dots or lines) reside on top of one another such that some are hidden. This is a common problem with large data sets, especially in graphs that encode values as dots, such as scatterplots and strip plots. When over-plotting occurs, information is lost. One of the techniques that was invented many years ago to resolve this problem is called *jittering*. To jitter data is to alter individual values slightly so that those that were equal or nearly equal to one another, and therefore were positioned on top of one another, are separated from one another. Jittering has long been applied as a useful solution to over-plotting in statistical analysis software. Jittering algorithms, however, typically function as a carpet bombing that alters the entire landscape (i.e., all of the values) rather than as a surgical strike that targets only those values that are subject to over-plotting and only to the degree that is necessary to resolve the problem. In this sense, most jittering algorithms can be considered buggy, as in jitterbugs. If jittering were applied more intelligently, we would get better results.

I would like to propose three improvements to jittering algorithms:

- Only alter the position of objects that are hidden behind other objects.
- Only alter those objects to the minimum degree that is necessary to make them visible.
- Reposition those objects in a location that harms the display minimally and supports the use of the data optimally.

I would also like to propose a single guideline for the use of jittering:

- Only use jittering when it is the most effective solution to the problem of over-plotting, given the tasks at hand.

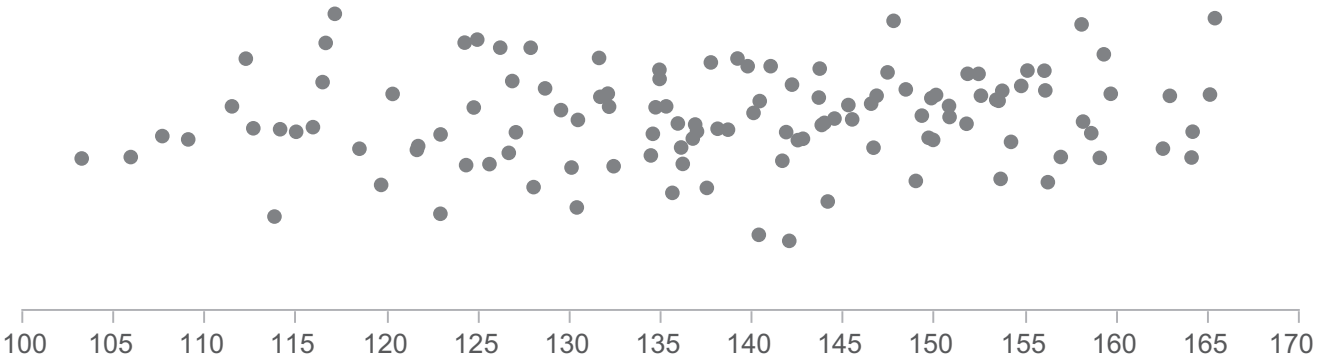
Jittering in Scatterplots vs. Strip Plots

Scatterplots position dots (or data points of other shapes) using the 2-D location of each dot to encode two quantitative variables: one horizontally along the X-axis and the other vertically along the Y-axis. Dots can be located at any position within the plot area. As such, we can allow jittering to reposition dots in any direction relative to their original positions. This provides the jittering algorithm a great deal of flexibility when determining the optimal new position for a dot. Of course, with greater flexibility comes greater complexity in the choices that the algorithm must make, but this is a manageable challenge.

Strip plots, prior to any jittering that might occur, position dots along a single quantitative scale that runs either horizontally or vertically. Dots are positioned along that scale to display multiple values of a single quantitative variable. Strip plots reveal patterns of a quantitative variable's distribution, along with several other graphs that share this purpose, such as histograms and frequency polygons.

When jittering is applied to a strip plot, it can potentially be done in one of two ways: either the values may be altered, which repositions dots along the same dimension as the scale (i.e., either horizontally or vertically),

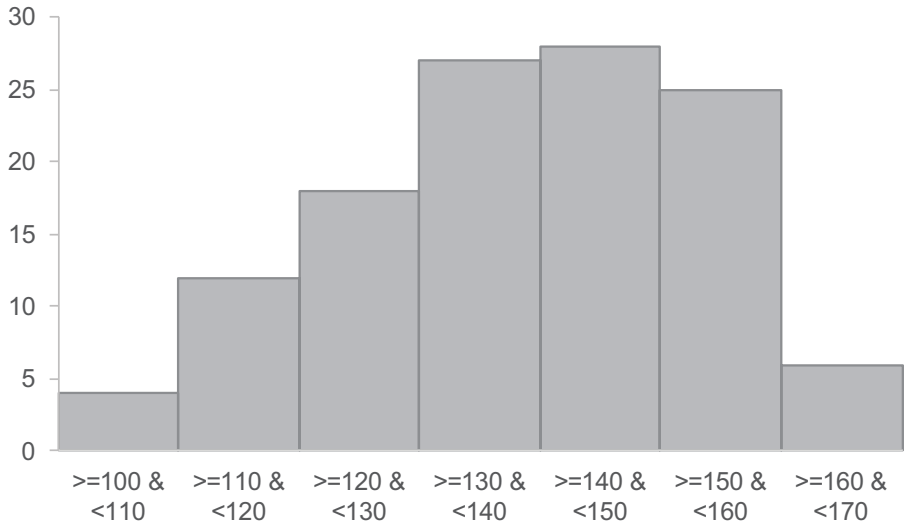
or their positions may be altered along the other dimension. Jittering a strip plot in the first manner, however, usually cannot eliminate over-plotting well because of insufficient space along the quantitative scale. Due to this frequent limitation, strip plots dots are usually jittered along the other dimension, such as in the example below, which was created using a popular visual analytics tool.



This particular example appears extreme in that the dots were repositioned to a much greater degree than was necessary to resolve the over-plotting in many instances, but not enough in others. This is typical of many jittered strip plots that I've seen. While jittering in this manner and to this degree certainly makes the individual dots visible, it does so in a way that introduces a new problem: a vertical arrangement of dots that doesn't convey meaning. As such, it exhibits patterns in the vertical positions of the dots that are completely arbitrary, yet they catch our eyes nonetheless. This arrangement also makes the overall pattern of distribution more difficult to see. Jittering in this manner and to this degree is usually done to make the individual values not only easy to see but also easy to select, such as by hovering over them to access their precise values. Ideally, when we display a distribution as a strip plot, however, we want to see the individual values while still maintaining as clear a representation of the distribution's shape as possible, which invites an improved form of jittering.

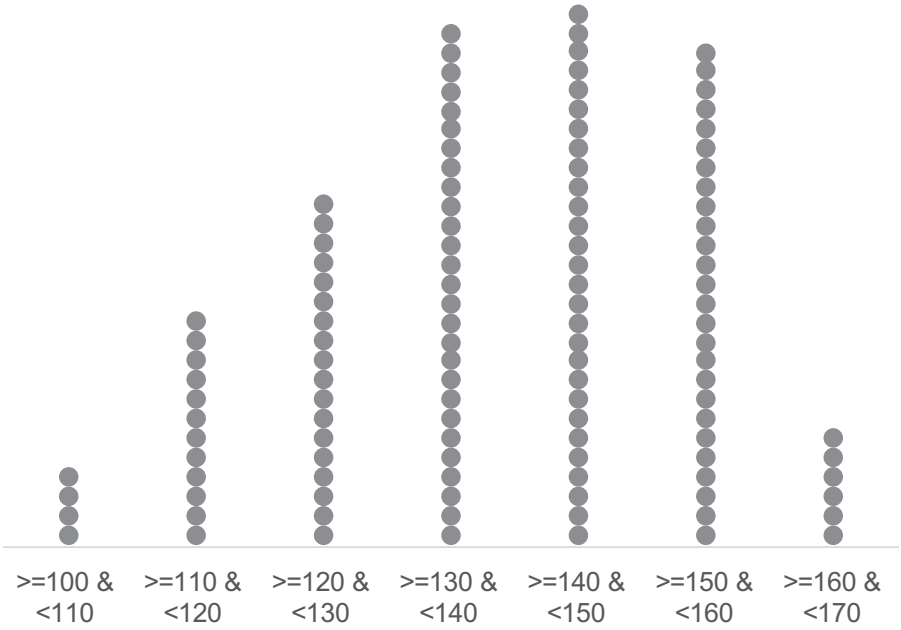
I'd like to propose a jittering approach that repositions dots along the dimension that is perpendicular to the scale, but does so in a way that clearly reveals the pattern of distribution. Before showing what I have in mind, however, it will help to consider another form of distribution display that includes a dot for each value: the *dot histogram*.

Regular histograms display the shape of a distribution by dividing the quantitative range into equally sized bins (a.k.a., intervals) and then using bars to represent the number of values that reside within each bin. The following example is based on the same data set as the strip plots above.

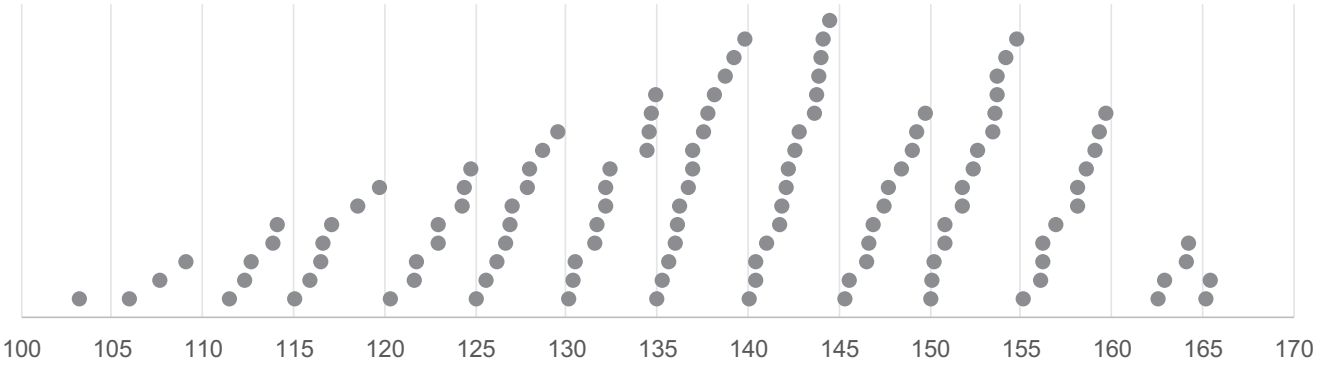


Histograms and other distribution graphs that use binning do a good job of revealing the overall shapes of distributions, but they don't identify the individual values that fall within each bin. For example, in the first bin,

which contains values greater than or equal to 100 and less than 110, the histogram reveals that there are four values, yet we don't know what they are. This is ordinarily fine, because usually when we examine distributions we care more about their overall shapes than the individual values, but it is not fine when the individual values must be seen. A dot histogram works like a regular histogram, except that, rather than using bars to display the number of values in each bin, it uses individual dots, one for each value.



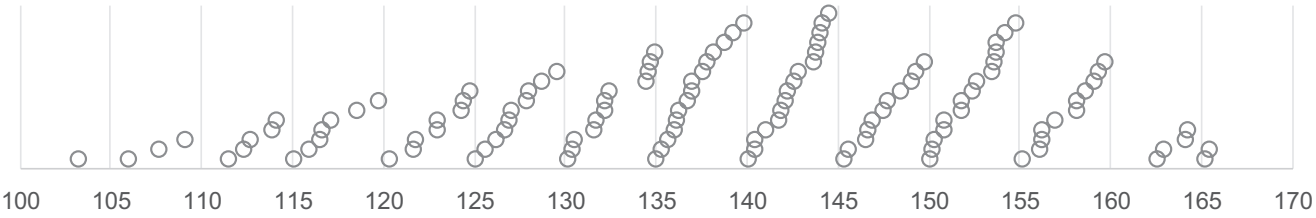
A dot histogram provides no advantage over a regular histogram except that, when it is viewed on a computer screen, it can be set up to allow hovering over individual dots to access their precise values, but this benefit is minor. Wouldn't it be useful if the dots could graphically depict their actual values, all visible at once? The new form of jittering that I have in mind does this. Here's an example, based once again on the same data set as the examples above, with smaller bins.



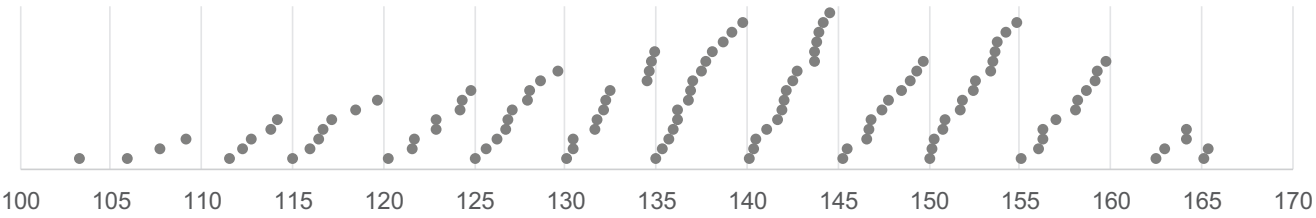
Even though the scale displays a continuous range of values, the dots have actually been grouped into bins in histogram fashion. In this particular case, each bin spans a quantity of five (greater or equal to 100 and less than 105, greater than or equal to 105 and less than 110, etc.) corresponding to the values that are labeled along the axis. Rather than aligning the dots in each bin, the dots have been arranged in order from the lowest value to the highest and aligned with their precise values along the scale. The curved alignment of the dots is meaningful, for it graphically displays the distribution of values within each interval, based on their positions. Although this looks odd at first glance, it takes only a minute to understand and learn how to read. With this design, in addition to seeing each value, we have a histogram-like view of the distribution's shape, and can also spot useful details, such as the two minor gaps that appear in the range between 130 and 135, as well as the gap in the lower half of the 160 to 165 range.

Because this form of display enables a strip plot to perform like a histogram while still retaining the individual values, I'm tempted to call it a *stripogram*. Cute or sexy names get more notice and have a better chance of sticking, but unfortunately they also belie the serious and useful nature of well-designed graphs. So, if we need a name for this other than a strip plot that's more usefully jittered, I propose that we call it a *wheat plot*, given the appearance of rows of wheat bending in the wind.

In the example above, the dots were jittered just enough to completely eliminate overlapping, but this isn't always an option. If there isn't enough space, by removing the fill color from the dots, they can overlap to some degree and still all be seen, illustrated below.

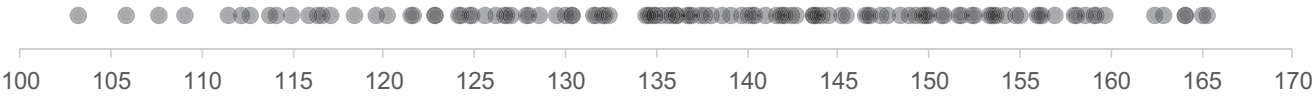


As an alternative when space is limited, the dots can often be made smaller, which might eliminate overlapping entirely, illustrated below.



Wheat plots, just like regular strip plots, can be arranged vertically or horizontally, which work equally well.

Bear in mind that, as with all graphs, wheat plots have limitations. For instance, it's not hard to imagine a limit to the number of dots that you can display without over-plotting in the space that's available. When this limit is reached, rather than jittering, a regular strip plot can be used with transparency applied to address over-plotting to some degree.



Obviously, the use of transparency does not allow you to see individual values when over-plotting occurs, but it does provide a sense of the distribution's shape while still seeing the patterns that we usually use strip plots to spot. I usually restrict my use of strip plots to relatively small data sets. When I use them to view large data sets, I do so primarily to look for the existence of small gaps in the distribution where no values reside, or to see small ranges where there are fewer or more values than I would expect to find, which could signal something meaningful about those ranges.

I invite those of you who have the ability to jitter strip plots in this improved manner to give it a try. I also invite software vendors to implement this form of jittering into their tools. More effective displays produce greater understanding and better decisions. Even minor improvements make a difference.

Discuss this Article

Share your thoughts about this article by visiting The DataVis Jitterbug thread in our discussion forum.

About the Author

Stephen Few has worked for over 30 years as an IT innovator, consultant, and teacher. Today, as Principal of the consultancy Perceptual Edge, Stephen focuses on data visualization for analyzing and communicating quantitative business information. He provides training and consulting services, writes the quarterly *Visual Business Intelligence Newsletter*, and speaks frequently at conferences. He is the author of four books: *Show Me the Numbers: Designing Tables and Graphs to Enlighten*, Second Edition, *Information Dashboard Design: Displaying Data for at-a-Glance Monitoring*, Second Edition, *Now You See It: Simple Visualization Techniques for Quantitative Analysis*, and *Signal: Understanding What Matters in a World of Noise*. You can learn more about Stephen's work and access an entire library of articles at www.perceptualedge.com. Between articles, you can read Stephen's thoughts on the industry in his blog.