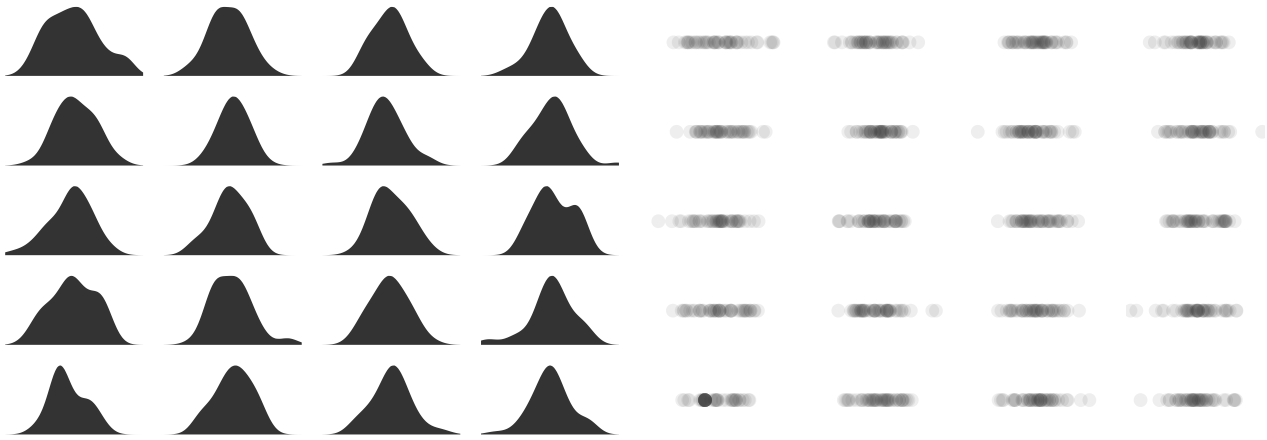
Looks Good To Me: Visualizations As Sanity Checks

Anonymized for Review



(a) (b)

Fig. 1: Example lineups from our evaluation. Both Fig. 1a and 1b show the same univariate datasets. 19 of these charts are "innocent" random samples from a Gaussian. One "guilty" chart is mostly random draws, but with 10 points that all have the same value (an extraneous mode). The oversmoothed density plot makes this abnormality difficult to see (participants were only 35% accurate at picking out the correct density plot). Low opacity scatterplots, however, make the dark black dot of the mode easier to detect (85% accuracy). See §6.1 for the right answer.

Abstract—Famous examples such as Anscombe's Quartet highlight that one of the core benefits of visualizations is allowing people to discover visual patterns that might otherwise be hidden by summary statistics. This visual inspection is particularly important in exploratory data analysis, where analysts can use visualizations such as histograms and 1D scatterplots to identify data quality issues. Yet, these visualizations are driven by parameters such as histogram bin size or mark opacity that have a great deal of impact on the final visual appearance of the chart, but are rarely optimized to make important features visible. In this paper, we show that data flaws have varying impact on the visual features of visualizations, and that the adversarial or merely uncritical setting of design parameters of visualizations can obscure the visual signatures of these flaws. We present the results of a crowd-sourced study that shows that common visualization types can appear to reasonably summarize distributional data while hiding large and important flaws such as missing data and extraneous modes. We make use of these results to propose additional best practices for visualizations of distributions for data quality tasks.

Index Terms—Graphical perception, data quality, univariate visualizations

1 Introduction

The exploration of the distributions of data within particular dimensions in a dataset is a critical step in Exploratory Data Analysis (EDA) [44]. Summary statistics, by themselves, may not capture important aspects of the data [6, 32]. Therefore, for large and complex datasets, visualizations of distributions function as "sanity checks:" these visualizations provide evidence that the underlying data are reasonably free of flaws such as missing values or excessive noise that might have an impact on how they are used in later analysis, and provide an important source of hypotheses for investigating interactions and relationships between multiple variables. These sanity checks also build trust in the underlying data processes. The assumption underlying the sanity check is that data flaws will have characteristic and legible visual signatures in visualizations.

In this paper, we investigate how data quality issues impact the visual appearance of different visualizations of distributional data, and how the design parameters of these visualizations can make the visual signatures of data quality issues more or less prominent. Through simulations and a crowd-sourced study, we show that different data quality issues (such as extraneous modes, missing values, and outliers) have differing levels of detectability across standard visualization techniques (such as histograms, density plots, and scatterplots). Furthermore, this detectability is highly contingent on the design parameters of these visualizations: many visualizations that look "reasonable" can fail to surface important data quality issues to the viewer. We use these results to suggest that designers be mindful of the visual appearance of data flaws in their visualizations, and to avoid common default settings of design parameters that, in general, make these flaws difficult to detect.

While there are many standard visualizations of distributions, these visualizations have different affordances, and are dependent on parameters (such as the number of bins in a histogram, or the bandwidth of the kernel using for smoothing) that have a large impact on their visual appearance. As these visualizations are more often used in the early and more cursory stages of analysis, analysts often do not spend much time editing or adjusting the design of these initial visualizations. This means that the default settings of visualization parameters like histogram bin count or mark opacity in 1D scatterplots are highly influ-

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

ential on what data features are or are not visible to analysts. Employing a metaphor from computer security, we view these parameters as an *attack surface*: through the adversarial setting of these parameters, a malicious or unthinking visualization designer could create a visualization that appears to accurately represent the underlying distribution, but in fact makes important data quality issues difficult or impossible to visually detect.

This paper therefore functions as a sort of vulnerability analysis: we show it is possible to create visualizations that seem "plausible" (in that their design parameters are within normal bounds, and they pass the visual sanity check of revealing nothing untoward in the underlying data) but hide crucial data flaws. We show that no one standard visualization design is robust to all of these sorts of attacks, but that each have different patterns of vulnerability. We conclude with recommendations that will improve the robustness and reliability of visual sanity checks.

2 Related Works

There are many ways in which a dataset can be flawed (see Kim et. al. [28] for a survey. However, there is relatively little work on how these data flaws can appear *visually*. A related line of research is scagnostics [48], which computes structural features of scatterplots, and especially its extension by Dang et al. [14] for the purpose of clustering time series by potentially interesting shared features. The premise behind scagnostics is that an analyst will want to focus on plots showing characteristic features that are associated with important relationships in order to derive interesting findings. For the purpose of sanity checking, we have a similar goal: the analyst should focus on plots that are "odd" in some way, with the assumption that the space of visual oddities is well-correlated with data quality issues that ought to be addressed by either data cleaning, exclusions, or increased skepticism in later stages of analysis.

In this work, we focus on univariate visualizations of distributions. We believe that the creation of visualizations of distributions are a common first step in EDA. These visualizations also tend to be simpler than multivariate visualizations in terms of the number of design parameters, and have been extensively studied by both statisticians and by those in the visualization community. Therefore, while we acknowledge that sanity checks can occur at all scales and steps of the analysis process, we deal mostly with univariate visualizations as a testbed for illustrating that data flaws can have a complex and potentially problematic relationship with visual features.

We build on this line of thinking in three ways in the following sections. Firstly, we review different univariate visualizations with an eye towards how they might be used to sanity check, and specifically how they are designed in practice. Secondly, we make the notion of designing for "oddness" and "sanity checking" more concrete by linking it to a framing of visualization design as an algebraic relationship between data and visualization. Lastly, we discuss how an adversarial framing of the sanity checking problem can encourage designers to think about not just the design of their visualization, but the robustness of their designs to variation and chance.

3 Visualizations of Distributions

While a full review of all techniques for visualizing distributions is outside the scope of this paper, we will discuss a small set of visualization types commonly encountered in EDA tools, focusing on the design parameters that impact their visual design, and how these parameters are set by default.

Many of the deficiencies we will bring up can be ameliorated through the interactive or data-driven adjustment of design parameters. While we acknowledge that these post-hoc "tweaks" of the visual design by the user can be powerful, we believe that they do not solve the issues related to visualizations as sanity checks. For one, interactive setting of parameters for sanity checking requires that the analysts take time to manually explore the design space for each data dimension independently, which is intensive in terms of both time and the analyst's attention. The second is that often these defaults can produce "plausible" visualizations that nevertheless hide important issues. That is, the analyst may not know that there are features that would be revealed through interaction, and so have no incentive to alter the visualization.

At the very least, we maintain that the default settings of designs parameters have an outsized influence on how data dimensions are spot-checked, and so consider default parameters across several common types of univariate visualizations in common visual analytics systems.

3.1 Histograms

Histograms are widely used in data prep and summarization contexts. They place data into discrete bins (as in Fig. 7). The main parameter in their design is therefore the size and location of bins. If there are too few bins, then spatial modes or trends may be lost. If there are too many bins, then each bin may be so sparse or noisy as to prevent the overall shape of the distribution from being visible.

The default bin size of histograms is often based on "rules of thumb" which use properties of the data to estimate a bin size that is assumed to be reasonable. These rules of thumb are based on statistical assumptions that may or may not hold for real datasets. For instance, Sturges' rule for histogram bins is derived from an assumption that the distribution to be estimated is a unimodal Gaussian [40]. This results in over-smoothed histograms when the data are not normal (for instance, if the data are bimodal, or have long-tailed outliers.

While data analysis systems may *support* more complex methods, by *default*, these simplistic rules of thumb are used. E.g., Sturges' rule is the default histogram binning method in R [3] and D3 [1], and a modified form of the Freedman-Diaconis rule [18] is the default histogram binning method in Tableau [4]. Lunzner & McNamara [30] explored "stretchy" histograms that allow the user to interactively compare histogram binning parameters, but acknowledge that very few standard data analysis tools fluidly support interactive re-binning with immediate visual feedback.

3.2 Density Plots

Density plots use an underlying Kernel Density Estimate (KDE) to create a smooth curve based on the observed data (as in Fig. 1a). The type and bandwidth of these kernels has a large impact on the resulting plot. As with histograms, there is a tradeoff between choosing a bandwidth which is too large (and so over-smooths the KDE) or too small (and so under-smooths the KDE) [39].

Likewise, the default setting of these parameters is frequently based on rules of thumb. Silverman's rule for determining the bandwidth of a kernel for KDE [41] is a common one, and is the default in R [2]. While more sophisticated data-driven methods for selecting these parameters exist (such as Sheather & Jones's pilot derivative-based method for selecting bandwidths [42]), these methods often require complex calculations that may not scale to large numbers of points, or require sampling or other approximation techniques that may not be stable across runs (see Jones et al. [25] or Park & Marron [35] for a survey). Silverman's rule, as with Sturges' rule, assumes that the underlying sampling distribution is a unimodal Gaussian, and so can oversmooth data where these assumptions are violated. With the exception of 2D density heat maps (which typically rely on KDE with adjustable bandwidths), we are not aware of any common EDA system that affords the interactive setting of kernel bandwidths with immediate visual feedback.

3.3 Scatter and Strip Plots

1D Scatterplots and strip plots encode each sample of a distribution as a dot or a line, respectively (as in Fig. 1b). While we focus on scatterplots in this paper, our guidelines apply to most visualizations of distributions where one sample directly corresponds with one mark.

Overplotting in scatterplots can obscure distributional data. This is especially true for 1D scatterplots: due the curse of dimensionality, overplotting is more likely to occur in univariate samples of a distribution than in bivariate samples of the same distribution. One technique to reduce overplotting is to jitter the marks, as in a beeswarm plot [17]). Adding jitter increases the vertical space taken up by the plot, and might be impractical if there are a large number of points to be plotted.

Altering the opacity of marks is a common technique to reduce the effects of overplotting. However, just as with histogram binning and KDE, this opacity must be chosen with respect to the structure of the data: too little opacity, and overplotting makes spatial modes and the shape of the distribution invisible. Too much opacity, and outliers and smaller structures are impossible to clearly distinguish. Recent approaches to optimize mark opacity in scatterplots rely on screen space image quality metrics [31, 33] and have not seen wide adoption in common visualization systems. The default opacity of marks in EDA systems we examined is either fully opaque (as it is in R and Tableau), or a constant opacity value (for instance, in the Vega-Lite [38], the default opacity of non-aggregated marks is 0.7 [5]). Unlike our previous visualization examples, common EDA systems like Tableau, Excel, and PowerBI *do* allow the interactive manipulation of scatter plot opacity through sliders with immediate visual feedback. However, in contrast, to our knowledge, no common visual analytics system provides data-driven procedures for optimizing scatterplot opacity, or provides data-driven defaults.

Another option to address overplotting is to reduce the size of marks, either by reducing the radius of a circle in a scatterplot, or by replacing closed shapes with open ones. In Tableau and VegaLite [38], for instance, the default scatterplot mark is an open, rather than filled, circle. However, as with opacity, the default *size* of the marks in scatterplots is usually a constant, rather than data-driven, value. There is also an interplay between size and opacity: the perceptual discriminability of the lightness of marks (and so, implicitly, the density) becomes poorer as the marks become smaller [43].

3.4 Other Univariate Visualizations

There are many other visualizations of distributions and sampled data. One tactic is to avoid presenting the samples directly, but instead aggregate the data into a small set of summary statistics. For instance, a dot for the mean and error bars for variance. As an example, boxplots can communicate quartile information, but can also be extended to communicate more complex summary statistics [37]. However, for many of the same reasons that summary statistics alone are not sufficient as sanity checks, visualizations of these summary statistics by themselves also fail to identify key classes of data quality issues. The correct interpretation of these summary statistics can also require statistical information that is outside of the analyst's expertise: error bars in particularly may often be misinterpreted [7, 10].

Of particular interest to us are "hybrid" or "ensemble" visualizations of distributions: these classes of visualization either combine layout and plotting techniques from multiple designs into one novel design, or literally juxtapose or superimpose multiple chart types together, respectively.

As an example of hybrid charts, Wilkinson's version of the dot plot [47] combines the scatterplot with the histogram by placing scatterplot marks into discrete columns, stacking marks when there would otherwise be overplotting. This method replaces the parameter of the histogram bin with that of the mark size: changes in mark size can cause large shifts in where columns are laid out. As with jittered scatterplots such as beeswarm plots [17], Wilkinson dot plots also take up more space depending on the mark size and modes of the distribution.

As examples of ensemble charts, Violin plots [21] combine a density plot with a box plot, and Bean plots [26] combine a density plot with a strip plot. These combinations are meant to supplement the deficiencies of their component visualizations, but are somewhat ad hoc.: it is not clear what sorts of ensembles best support different sorts of sanity checks. Designers must also independently set parameters for each of the component visualizations.

4 Algebraic Visualization

Kindlmann & Scheidegger's framework of algebraic visualization design (AVD) suggests two principles for creating and evaluating visualizations [29]. First, visualizations should be assessed by reference to potential *transformations of the data*, which are denoted in AVD by αs. Every function α that transforms the input induces a function that transforms the actual image produced in the visualization (denoted in AVD by ωs). By considering different ways to transform the input, designers can investigate how the visualization responds differently. In AVD, an α that does not change the visualization output (that is, whose corresponding ω is an identity) is known as a *confuser*: the change in the data does not produce a corresponding change in the visualization. Because such αs encode two different ways in which the data could have been that are indistinguishable from the perspective of the generated output, these confusers tend to signify potential problems in the visualization. Crucially in the AVD framework, αs that are confusers for some visual mappings are *not* confusers for other visual mappings. Second, visualizations should be *invariant to equivalent representations of the data*. If a visualization produces different input because of the order in which elements are stored in a list, then the switch from one representation to another is a *hallucinator*: a way to effectively "trick" a visualization to produce different outputs from what effectively is the same input.

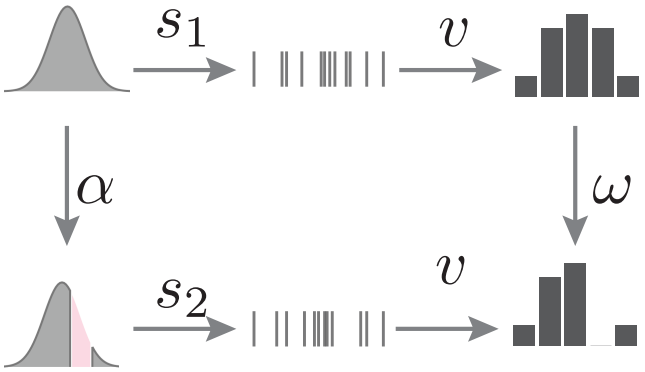


Fig. 2: The commutative diagram in this figure provides a way to evaluate visualizations of population samples. If a change α occurs in the population distribution (for instance, the introduction of a flaw such as a region of missing values), we expect to see a proportionally large change in the visualization ω. However, the data to be encoded is often a sample of this unknown distribution: this sampling s can also introduce visual changes in the resulting mapping from data to visualization v.

Samples as (inaccurate) representations of a population In this paper, we build on Kindlmann & Scheidegger's observation in AVD that the *sampling process* we use to obtain finite samples from a population can be thought of as a representation mapping. As we illustrate in Figure 2, similarly to AVD, we model the possibility of getting different samples from a population by different sampling mappings, in this case s_1 and s_2 (but there are clearly many other possible mappings). If α is a nontrivial data transformation which commutes with an identity mapping ω, then v is discarding information from the data that is necessary to disambiguate the effect of α. On the other hand, if α is an identity mapping while ω is not, then our visualization is simply showing differences between the different samples rather than a feature of the population itself. In this paper, we encode this perspective on data transformations by thinking of the data flaws in §6 as different potential αs, and then creating many visualizations without those flaws but with different sample mappings. If participants are unable to spot the odd visualization, then one of the two failure modes described above has occurred: either a significant alteration in the data failed to produce a large enough change in the visualization to make it stand out (a confuser), or the visual differences caused by non-significant sampling variation was sufficient to produce visualizations which stand out despite having no data alterations (a hallucinator).

Visualizations have forced choices that are representation-independent In addition, the work we present here highlights an additional complication in applying AVD to the design and evaluation of visualization designs. AVD was developed with the goal of reducing the number of arbitrary decisions in the course of designing a visual-

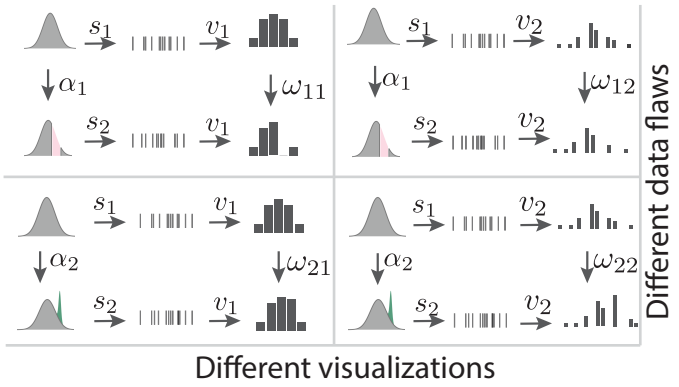


Fig. 3: Different data flaws interact with different visualization mappings differently based on the setting of design parameters, producing a large number of potential ωs. A visualization and parameter setting that is adept at depicting a certain class of data flaws unambiguously might be a confuser for another type of flaw.

ization (or, conversely, to point out arbitrary decisions in potentially sub-optimal designs). The main discussion in AVD focuses on forced choices and arbitrary features of input representations being the cause of confusers and hallucinators. Notably, AVD does not handle the setting where a visualization mapping itself requires a forced choice that can be made *independently of the data*. These are typically design parameters: the transparency of dots in a dot plot, the number of bins in a histogram, or the bandwidth of a density plot. One of the central observations of the work in this paper is that different αs require different settings for these parameters. In other words, we present evidence that appropriately setting parameters requires not only appropriate algorithms that take the data being given, but an understanding of the desired transformations that the viewer should be able to visually distinguish.

If we model these different parameter settings as different visualization mappings (analogous to how we handle different samples from a population), then, as we illustrate in Figure 3, we have not only a large number of ω mappings to assess, but we ought to consider the consequences of an *adversarial* setting; are there visualization mappings in which it is possible to *hide* specific flaws in the data-generation process by making particularly bad parameter choices?

5 Adversarial Visualizations

Huff's formative book "How To Lie With Statistics" [23] explores many ways in which statistics can be manipulated to convey misleading impression of the data. More recently, Correll & Heer [12] propose the metaphor of "Black Hat Visualization:" that the designer of a visualization can act as a "man-in-the-middle" attacker between a dataset, and the analyst who wishes to understand and communicate using the dataset. In an adversarial setting, the designer can create visualizations which seem to accurately convey the data, but in fact hide or obscure data features. In this paper, we are especially interested in the case where a visualization can seem to plausibly depict a distribution, but, in fact, hides important distributional features, and so whose αs are confusers, in AVD terms. It is our contention that the uncritical (or intentionally malicious) setting of design parameters can reliably generate visualizations of this adversarial form.

In order to explore the attack surface of design parameters in visualizations of distributions, we created a web-based tool for automatically creating confusers via the adversarial setting of design parameters. That is, given an initial set of samples, we introduce one or more αs (data flaws such as noise or mean shifts or gaps). We then exhaustively search through a space of "plausible" design parameters assuming a data range of $\{0,1\}$: from 3 to 50 histogram bins, KDE bandwidths of 0.1 to 0.25, and scatter plot mark radii of 10 to 25 pixels crossed with mark opacities of 0.2 to 1.0. We then report the parameter settings that result in the smallest number of different pixels between the visualization of the original and the visualization of the flawed distribution (an imperfect proxy for ω, $\hat{\omega}$). The smaller this difference, the worse the confuser, and so the more successful the "attack."

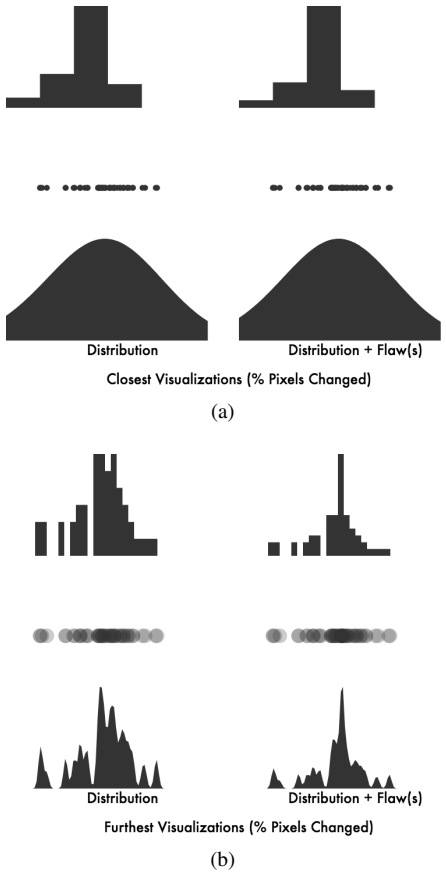


Fig. 4: A synthetic adversarial visualization. We added a "flaw" (in this case, 10 sample points all with the same value) to a dataset. The visualizations on the left column are the original distribution. Those on the right are the flawed distribution. By setting design parameters adversarially (as in Fig. 4a), we can hide this data flaw. Conversely, we can set the parameters to attempt to highlight the differences (Fig. 4b).

Javascript code for creating your own adversarial visualizations using the p5.js framework is available at `URL-REDACTED`. Echoing the prior literature, we found that scatterplots with high opacity could successfully "cover up" many data flaws, provided the marks were large enough, and the samples dense enough. Similarly, small numbers of histogram bins, and wide bandwidths, can result in plots that are virtually identical despite large changes in the underlying distributions.

Fig. 4 shows an example of one such attack. In this case, our original dataset was 50 samples from a Gaussian. We then introduced a mode of 10 additional points with the exact same value within the IQR of the samples. By choosing the design parameters (in this case, the number of bins in a histogram, size and opacity of points in a scatterplot, and the bandwidth of a KDE) adversarially, we can almost entirely hide the spike (Fig. 4a). This process of creating this mode is therefore a confuser: for histograms, the spike occurs in the modal bin, causing the other bins to renormalize their heights but otherwise keep a similar shape. For scatterplots, small points with maximum opacity hide the new mode among other, sparser sample points. For the density plot, the overlarge bandwidth smooths the mode into the rest of the points.

Pixel difference is not always a reasonable proxy for human judgments of similarity in visualization, which focuses on larger-scale visual structures [34]. We present these examples to show that the visual signature of data flaws can be quite subtle (or can be made to be subtle) while still producing visualizations which appear reasonable. Our perceptual

study attempts to address these factors directly, while also measuring how robust different visualizations are to theses sorts of attacks.

6 Evaluation

For visualizations to function as sanity checks, data flaws must be reliably detectable in these visualizations. This in turn means that the visual signatures associated with these flaws must be prominent enough to be visible (we should avoid confusers), and robust enough to not be confused with the visual properties of visualizations of data without these flaws (we should avoid hallucinators). We therefore had the following research questions:

1. How good are the general audience at detecting data flaws in standard visualizations of distributions?
2. Do certain visualizations result in more reliable detectability (and so better sanity checks) than others?
3. How sensitive are these visualizations to the design parameters used in their construction?

To answer theses questions, We performed a crowdsourced experiment to evaluate how detectable different data flaws are amongst different visualizations, and how robust this detectability is amongst different design parameter settings. Our objective was not to fully map out the space of detectability among features, visualizations, and parameters of interest, but to provide preliminary empirical support that data features can have characteristically different visual impacts on different visualization types, and that the design parameters of these visualizations, even within reasonable ranges, can make these visual signatures more or less prominent.

Experimental materials, including data tables and stimuli generation code, are available at `URL-REDACTED`.

6.1 Methods

Since our focus was on whether analysts could robustly detect data flaws in charts, we adopted the "Visual Lineups" protocol. Hofmann et al. [22], as well as Vanderplas et al. [45], have used this protocol to determine the varying signal-detection power of different graphical designs; we use it for a similar purpose here. The task, as laid out by Wickham et al. [46], is to create a "police lineup" of n visualizations. 1 of these visualizations contains the actual data "culprit" (e.g., the dataset with the signal present). The other $n-1$ visualizations contain "innocent" data generated under some null distribution (in our case, random draws from Gaussian. The analyst must then try to identify the culprit. For instance, in Fig. 1, the chart on the bottom left of the lineup is "guilty." Under the null hypothesis, the probability of selecting the culprit by chance (the false positive rate) is $\frac{1}{n}$. These lineups can therefore be used to assess a measure somewhat like statistical power. While other methods for assessing the perceived similarity between visualizations exist (e.g., Demilarp et al. [15] and Correll & Gleicher [11] both solicit subjective Likert ratings of visual similarity), the lineup protocol seems to be a reasonably proxy for sanity checking in EDA, as it entails a visual signal detection task in the context of sampling error, as per our formalization in Fig. 2.

Each trial of our experiment was a different lineup, with 19 charts generated via random sampling from a Gaussian, and 1 chart generated from Gaussian draws, but with the addition of some flaw. We used 3 different flaw types (Spike, Gap, or Outlier, described below) with 3 different flaw magnitudes (10, 15, or 20 flawed sample points), across 3 different chart types (1D scatterplot, histogram, or density plot), and 3 different parameters settings (3 different levels of mark opacity, bin count, or kernel size for our 3 chart types, described below), for a resulting $3\times3\times3\times3$ within subjects factorial design, for a total of 81 stimuli.

For training and engagement check purposes, we also included at least one example of each combination of visualization and flaw type (so 9 additional stimuli total), with a flaw size of 25 abnormal points, and with parameters we believed to be favorable for detection. These trials were excluded from analysis.

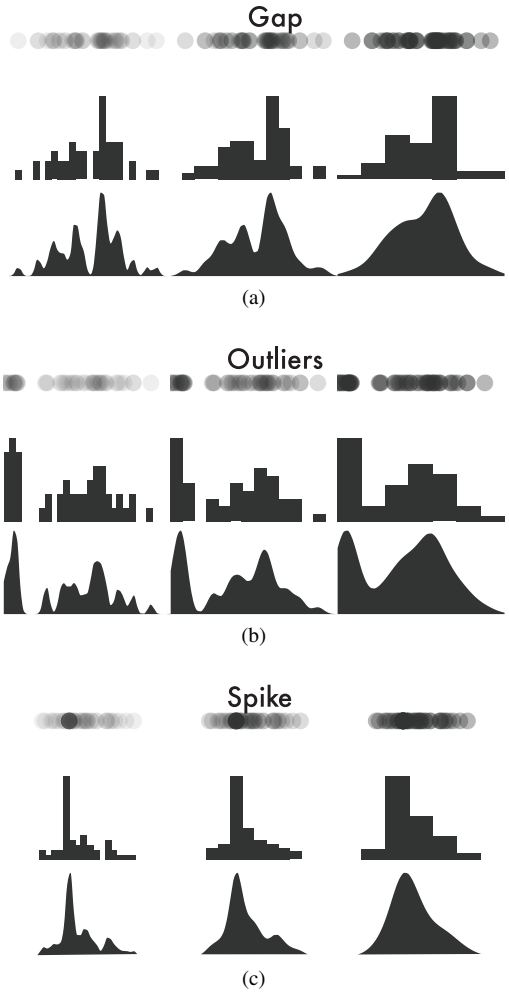


Fig. 5: Examples of our data flaws we tested, and their visual signatures across scatterplots, histograms, and density plots. From left to right are examples of the settings of our design parameters: increasing mark opacity, decreasing the number of bins, and increasing kernel bandwidth, respectively.

Each visualization consisted of 50 samples from the a Gaussian distribution with a mean of 0.5 and variance 0.15. Since Gaussians have infinite support, we clamped extreme samples to the interval $\{0,1\}$. We generated the resulting visualizations as 150×100 pixel svg images using D3 [8].

We generated "flawed" graphs of $50-n$ samples and n abnormal points via the following strategies (Fig. 6):

- **Gaps**: We randomly sampled $50+n$ points from the null distribution, and randomly chose a value uniformly from $[Q_1,Q_3]$. We removed the n closest points from this location. This results in an irregularly sized region of missing values somewhere in the middle of the distribution.
- **Outliers**: We randomly sampled $50-n$ points from the null distribution. We measured the post-hoc quartiles of these samples. We then placed n points randomly in either the interval $[0,Q_1-1.5{*}\text{IQR}]$ or $[Q_3+1.5{*}\text{IQR},1]$, whichever was further from the original sample mean. This results in a "clump" of extreme values on one end of the distribution.
- **Spikes**: We randomly sampled $50-n$ points from the null distribution. We measured the post-hoc quartiles of the samples, and randomly chose a value uniformly from $[Q_1,Q_3]$. We added

Fig. 6: A representation of the αs that we tested in our study, representing three classes of potential data quality issues. Fig. 5 shows how these flaws might appear across the visualizations we tested in our study.

n points with *exactly* this value to the sample. This results in a "spiky" mode somewhere in the middle of the distribution.

The parameters for the visualizations were generated based on scalar multiples of the observed defaults from prior work, based on the idealized sample (which would have $\bar{x} = 0.5$ and $S_x = 0.15$:

- **Scatterplot**: We believed that the 0.7 default opacity of Vega-Lite (and certainly the 1.0 default of other VA tools) would result in limited dynamic range in opacity in our scatterplots. In piloting, opacities of 0.7 with 50 points produced nearly solid black scatterplots, and so we used opacities of $\{0.35, 0.175, 0.0875\}$ in order to avoid floor effects. While the size of marks is also a parameter that can obscure data flaws (and, indeed, it is one of the attack surfaces in §5), we kept a relatively large constant mark radius of 10 pixels both to limit the amount of tested factors, and to create situations in which opacity would result in large visual differences between plots.

- **Histogram**: Sturges' rule would provide 7 histogram bins. We believed that this would result in too few bins, and so created we histograms with $\{7, 14, 28\}$ bins.

- **Density Plot**: We believed that the bandwidth of 0.07 provided by Silverman's rule would result in oversmoothing, and so used bandwidths of $\{0.07, 0.035, 0.0175\}$.

Fig. 5 shows our data flaws, and how they might appear across the different design parameters of our visualizations.

There are several significant differences between this task and real-world sanity checks. The first is that, in real-world sanity checks, the analyst is unlikely to have access to multiple draws from the same sample, but, rather, one unique dataset. The lineups here complicate flaw detection (in that there are many candidates to examine for flaws), but also might assist in discarding false positives (in that the participant is exposed to the sampling variability first-hand, and so may be better able to distinguish between sampling error and other sources of error). The second is that, in the instructions, the participants were given only a very simple prior about the shape of the distribution: "most of the charts will have the most amount of data in the middle of the chart, and gradually fewer and fewer data points the further from the center of the chart." Beyond this instruction, we did not give any context or fictional framing of our data, as these narrative framings can complicate crowdsourced studies [16]. Real-world analysts are likely to have stronger conceptual models about their data, including knowledge about data format (ages are non-negative integers that are rarely triple digits, for instance), and perhaps even a strong prior about the data distribution (word frequency in texts generally follows Zipf's law, for instance). Lastly, for each trial, the participants were informed of the precise type of flaw they were meant to detect. The full space of dirty data features that an analyst might care about is immense, and a real-world sanity check would entail a simultaneous search across this entire space, rather than a directed search for one specific type of abnormality. While we acknowledge these ecological differences, we believe that the task captures important aspects of sanity checking as a signal detection task.

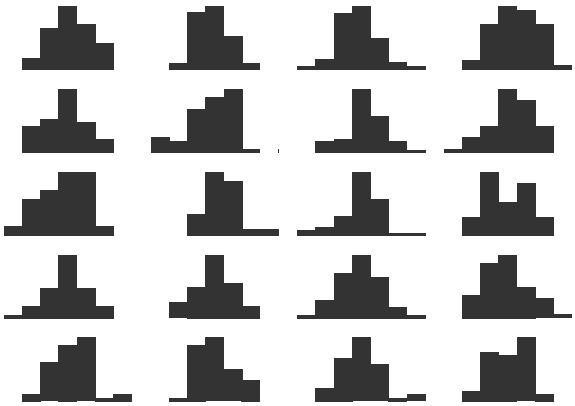


Fig. 7: An example lineup. 19 charts are innocent random samples from a Gaussian, one contains a gap where 15 contiguous points have been removed. When there are only a few histogram bins, these bins are unlikely to match exactly with the gap; it can be hard to distinguish true gaps (which will appear as shorter than expected bars) from variation due to sampling error: participant accuracy at gap-detection for histograms with 7 bins was 7%, close to chance. See §6.3.4 for the correct answer.

6.2 Hypotheses

Based on our testing concerning the visual properties of visualizations of distributions, we had the following major hypotheses:

- **As the number of flawed points increases, accuracy will increase**. Our model for this task was signal detection. As such, we expected that the strength of the signal (in terms of the number of abnormal points) would result in higher detectability.

- **Different visualizations would be better suited for different detection tasks.** That is, we expected an interaction effect between participant accuracy and visualization type. Furthermore, we did not expect any one visualization to dominate, but that different visualizations would outperform the others for different classes of flaws.

- **Existing default parameters are too conservative.** That is, participants would have higher accuracy for lower opacity scatterplots, larger numbers of histogram bins, and smaller bandwidths than suggested by VA defaults.

6.3 Results

We report our effect sizes using bootstrapped confidence intervals of 90% trimmed means, as per Cleveland & McGill [9]. Trimmed means violate sampling assumptions for standard null-hypothesis tests, however, and so those tests (such as ANOVAs) are reported based on standard means.

6.3.1 Participants

We recruited our participants using `Prolific.ac`. Prolific is a crowdworking platform focused on deploying online studies. Results from Prolific are comparable to those of other crowdwork plaforms [36] such as Amazon's Mechanical Turk. Turk, in turn, has results that are comparable to in-person laboratory studies for graphical perception tasks [20]. We limited our participants to those between 18-65 years of age, and with normal or corrected to normal vision. Based on internal piloting, we rewarded participants with $4, for a target compensation rate of $8/hour. Participants, on average, completed the task in 22 minutes ($\sigma_t = 12$ min.), for a post-hoc rate of $10.91/hour.

We recruited 32 participants, 21 men, 11 women, $\mu_{age} = 28.5$, $\sigma_{age} = 8.2$. 26 had at least some college education, of which a further 7 had graduate degrees. We solicited the participants' self-reported familiarity with charts and graphs using a 5-point Likert scale, with 1 being least familiar, and 5 being most familiar. The average familiarity

was 2.62, with no participants reporting a familiarity of 5. 1 of our participants did not answer any of the training stimuli correctly, and so their responses were excluded from analysis.

6.3.2 Signal Detection

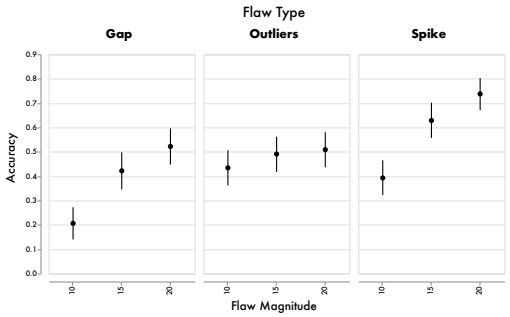


Fig. 8: Accuracy at identifying which visualization contained a specific data flaw, given the size of that flaw in turns of abnormal points (out of 50 total points). Confidence intervals represent 95% bootstrapped C.I.s of the trimmed mean.

Chance at this task was $1/20 = 0.05\%$. Across all conditions, participant accuracy was 48.3% (45.9%, 50.9%), significantly higher than chance. The average response time from signaling that the participants were ready, to confirming their selected choice, was 8.5s. Not all flaws were equally detectable, however. A Factorial ANOVA of accuracy as a function of flaw type and flaw size (in terms of the number of abnormal points), and their interaction, found a significant effect of flaw type on accuracy $F(1,2490) = 23$, $p < 0.001$, as well flaw size ($F(2,2490) = 69$, $p < 0.001$). Their interaction was also a significant effect ($F(2,2490) = 8.2$, $p < 0.001$).

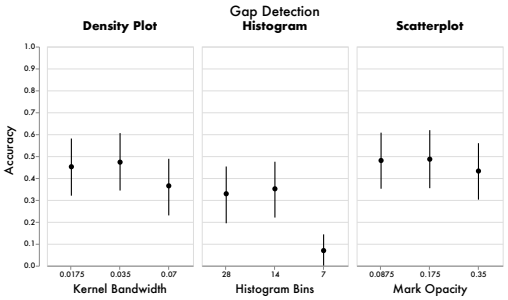
In general, spikes were the easiest to detect, with an accuracy of 58.8% (55.2%, 61.9%), followed by outliers at 47.8% (43.6%, 52.2%), with gaps being the hardest to detect at 38.3% (34.0%, 42.5%) accuracy. A Tukey's HSD showed that not all parameter levels were significantly different from each other. In particular, the number of outliers did not significantly affect their detectability. Fig. 8 shows these results in more detail. In general, these results partially supported our first hypothesis: **Participants were better able to detect flaws as these flaws were larger**, with the exception of outliers.

6.3.3 Visualization Performance

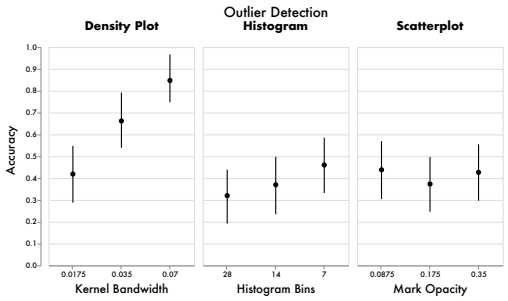
Not all visualizations were equally useful for detecting flaws. A Factorial ANOVA of accuracy as a function of flaw type, visualization type, and their interaction, found a significant affect of visualization on accuracy ($F(2,2454) = 17$, $p < 0.001$). There was also a significant interaction effect ($F(4,2454) = 5.8$, $p < 0.001$).

Across all conditions, density plots were the most accurate, 56.8% (52.5%, 61.1%), followed by scatterplots, 48.0% (43.8%, 52.2%), with histograms being the least accurate, 40.2% (36.0%, 44.4%). A post-hoc Tukey's HSD confirmed that all visualizations were significantly different from each other.

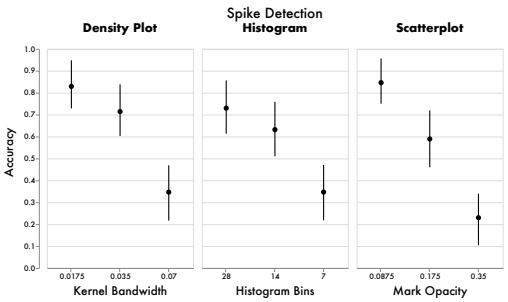
Despite this ranking, our results partially support our second hypothesis: **no single visualization was significantly better for detecting every type of data quality issue**. Density charts performed better than the other visualizations for detecting outliers (64.4% (56.9%, 71.4%) vs. 39.8% (34.8%, 44.8%) for the other designs) and for detecting spikes (63.3% (56.0%, 70.6%) vs. 56.5% (51.5%, 61.6%) for the other designs), while scatterplots were best for finding gaps (46.7% (39.4%, 53.8%) vs. 34.1% (29.0%, 39.1%) for the other designs). A post-hoc Tukey's HSD confirmed that density charts were significantly better than other charts for outlier detection, and that histograms were significantly worse than other charts for detecting gaps. All other charts were comparable.



(a) Gap Detection by Visualization



(b) Outlier Detection by Visualization



(c) Spike Detection by Visualization

Fig. 9: Performance at our flaw detection tasks across visualizations and visualizations parameters. Each rightmost column across visualization types represents a default value from Silverman's Rule, Sturges' Rule, and $\frac{1}{2}$ VegaLite defaults, respectively. With the exception of the outlier detection task, deviating from these defaults resulted in better performance. Confidence intervals represent 95% bootstrapped C.I.s of the trimmed mean.

6.3.4 Parameter Settings

Accuracy at detecting flaws was highly dependent on the settings of the design parameters of the visualizations. In the most extreme case, doubling the amount of bins in a histogram from the 7 recommended by Sturges' rule to 14 resulted in a 5-fold increase in accuracy at detecting gaps (from 7%, near chance, to 35%). Fig. 7 shows an example of this extreme case. In that figure, the guilty visualization is the right-most column, third from the top.

Our results partially support our third hypothesis: **default parameter settings are often too conservative for sanity checking**: for two of the tasks, the default parameter settings underperformed the more liberal settings (lower scatter plot mark opacity, more histogram bins, and smaller KDE bandwidths). A Factorial ANOVA of accuracy as a function of flaw type, the default nature of a parameter setting, and their interaction, found a significant interaction between default settings and flaw type ($F(2,2490) = 28$, $p < 0.001$). A post-hoc Tukey's HSD found that, for each flaw, the more liberal parameter settings outperformed the default settings, except for outlier detection, where the default parameters significantly *overperformed* the non-defaults. An

analysis of this result shows that it is mostly due to density plots: since the outliers were sufficiently far from the mode to not be "smoothed away" by large bandwidth kernels, the increased kernel bandwidth had the effect of making the outliers more prominent.

An analysis of our per-visualization results show that the aggregate group differences were driven by several particularly beneficial or harmful parameter settings, in keeping with our adversarial results from §5: small bandwidths KDEs make spikes especially prominent, as the large number of overlapping points in a small region create a large visual spike that is not smoothed into the neighboring points. Histograms with small numbers of bins make the detection of spikes and gaps difficult, as the spike causes renormalization (as in Fig. 4), and the bin containing the gap will contain many neighboring values, resulting in a subtle visual change rather than a sudden drop-off in density. Lastly, high-opacity scatterplots make internal modes indistinguishable from overplotted regions. Fig. 9 shows these results, pivoting on detection type, task, and parameter setting.

7 Discussion

Our results identify some potential concerns with the use of visualizations as sanity checks. Overall, it appears that the detection of data flaws from visualizations is by no means reliable, or neatly disentangled from differences caused by sampling error: even spikes, the flaw with the highest overall detection rate, were correctly identified only 58% of the time, despite the fact that the average spike was 20% of the total points in the sample. Real world datasets, with more subtle data flaws, larger sample sizes, and more complex sources of error, should induce even more skepticism about the ability of standard visualizations to reliably surface data flaws.

Our results also suggest that there is no single visualization design, or parameter setting, that will make all potential data quality issues equally visible. While density plots were the most robust visualization we tested (across all flaws, they were either the visualization with the highest performance, or not significantly different from the visualization with the highest performance), density plots also showed wide variability in their performance based on their bandwidth. Disadvantageous or adversarial setting of design parameters can erase the performance benefits of any particular visualization design.

Lastly, our results suggest that, in many cases, more liberal settings of things like histogram bin size and mark opacity will result in better performance. In the following section, we speculate on the visual impact of these features across each visualization type we tested, and their implications for designing for sanity checking.

7.1 Density Plots

We were in general surprised by the relatively high performance of density plots across conditions, as the specifics of their design is based on a concept, KDE, that requires a reasonable amount of statistical background to interpret. Their performance here indicates that they may be justified for inclusion among the standard arsenal of distribution visualization tools in visual analytics systems, despite their sensitivity to their underlying parameters.

As discussed in prior work, there appears to be a significant cost for oversmoothing in density plots with respect to detecting gaps and modes. Oversmoothed modes are "blended" into the regular distribution. Oversmoothed gaps make the case of *no* data ambiguous from the case *less* data, especially if they occur in regions with large amounts of data on either side of the gap. Our results show a monotonic decrease in performance for these tasks as bandwidths get larger.

However, we observed a positive benefit for large bandwidths in outlier detection. On reflection, outliers are large numbers of points in sparse regions: a large bandwidth would make these outliers "wider" and more visually prominent. Large bandwidths also help disambiguate the small number of extreme points that would occur due to sampling error (which would be smoothed into the larger shape of the distribution) from a cluster of systematically outlying points (which stick out).

Therefore, while we recommend that designers consider using density plots in wider applications, we suggest that they be mindful of the data quality flaws they expect their analysts to encounter, and consider that rules of thumb such as Silverman's rule may be too conservative to show much in the way of the internal structure of distributions.

7.2 Histograms

We were also surprised by the relatively poor performance of histograms; in no condition were they significantly better than other visualization types for detecting flaws, and in one case (gap detection), they were significantly *worse* than their competitors. Histograms are a standard tool for visualizing distributions, and are the default per-field visualization in commercial data prep tools such as Trifacta's Wrangler and Tableau's Project Maestro. Histograms do have some advantages in terms of the scalability of their design (affording the visualiztion of arbitrary numbers of data points in a constrained visual space with no potential for overplotting), but our results indicate that they, in themselves, may only be weak evidence for the presence or absence of a flaw in the data.

Expanding on existing results, we show that there are severe performance costs to underestimating the number of histogram bins. If there are too few bins, then missing data can be small enough to be drowned out by dense neighbors. As with density plots, this underestimation also causes an ambiguity between *no* data and *less* data, a signal that can be confused with variability due to sampling error. Similarly, histograms are often normalized with respect to their maximum observed density. Therefore, extraneous modes that are too close to existing modes can just force a renormalization of all the other bins, while leaving the modal bin unchanged. This "renormalization bias" can obscure important patterns in both univariate and spatial visualizations [13]. Large bins, however, can group all outliers into a single bin of locally high density. These large outlying bins are more prominent than if outliers cross multiples bins. They also reduce some of the ambiguity in whether there are enough extreme points to indicate a data quality issue, or if they are artifacts of sampling error.

In general, we do not recommend the use of Struges' rule for creating sanity checking histograms. Other simple rules of thumbs, such as the Freedman-Diaconis rule [18], are more liberal (it would generate 13 bins for our idealized distribution, instead of the 7 of Sturges' rule). Even so, with extremely liberal bin settings, other visualization types, which do not discretely aggregate data, are likely to expose a wider class of data flaws than histograms.

7.3 Scatterplots

We expected strong performance from scatterplots which, under reasonable settings, have direct and unambiguous visual signatures for each one of our data flaws. For instance, modes appear as solid circles surrounded by less opaque neighbors, and gaps appear as continuous regions with no marks whatsover (see Fig. 5). Overplotting was an issue for the detection of spikes, as there were many candidate regions with maximum opacity. Otherwise, scatterplots were robust to opacity issues.

Unlike histogram binning, which has been studied for decades, work on automatically setting the opacity of points in a scatterplot is relatively recent [31, 33]. As such, there are few heuristic rules for setting mark opacity. Our results, while coarse, do not suggest any particular local maximum of performance: most opacities, so long as they are low enough to reduce the effects of overplotting, will still afford many forms of sanity checking. Not just the opacity, but the size of the marks, and the density of the data, will contribute to the severity of overplotting.

The relative robustness of scatterplots could be due to the fact that, for feasibility reasons, our list of viable parameters for mark opacity began relatively low (half of VegaLite's default mark opacity of 0.7). We piloted with different maximums (including 0.7 and 1.0), but encountered floor effects, as most of the scatterplots were very close to solid black lines of overplotted marks. Therefore, we still recommend that designers be mindful of scatterplot opacity, and generally more liberal in setting defaults than those of alternative analytics tools. Our results indicate that, unlike the other visualization types where there were performance benefits on both extremes of the parameter scale, data flaws are visible even with very low mark opacity.

When employing scatterplots for sanity checking purposes, we recommend that designers, by default, employ some method of ameliorating overplotting, either in terms of mark opacity, jittering of points, or reduction of mark size. Constant opacity values, that do not take into consideration the number of modality of points, may not be sufficient measures to produce usable scatterplots by default.

8 Limitations & Future Directions

8.1 Limitations

We explored a narrow and coarse range of design parameters in our lineup task. It is not within the scope of this paper to generate full models of the impact of design parameters on flaw detection tasks. As mentioned in our exploration of univariate visualizations, there are many other techniques for automatically setting parameters such as histogram bin size or KDE bandwidth, compared to the rather simplistic rules of thumb we use as our defaults here. Our goal with this work was not to derive new rules, but to examine how these parameters are set in practice, and to explore what impact these parameter settings have on the task of sanity checking data. It is our contention that analysts may not be aware of the existence of more complex parameter-setting rules, or may not feel the need to adjust the initial univariate plots. It is our hope that this work prompts the designers of future visual analytics systems to think more carefully about how these defaults are set, and how the system can encourage user interaction with these initial univariate plots.

We also limited the visualizations we tested to a small class of well-known exemplars. Just as one example KDE, by generating a smooth convex curve of data density, can borrow any number of visualization techniques that have been evaluated in the context of either probability distributions [24] or time series visualization [19]. Again, we were focused on sanity checking visualizations as they occur in common visual analytics systems. More complex visualizations (such as Summary Box Plots [37]) may directly encode features that are relevant to data quality. These methods may introduce costs in terms of visual complexity or training time, however. Designers should weigh these costs (and the false positive and false negative costs of detecting potential data quality issues) when determining whether or not to support more complex initial views of distributions.

Our experimental task, by forcing participants to make a choice, does not afford the precise discrimination between false positives (they selected a chart with no flaw, but a visual distinction that is a mere result of sampling error) and false negatives (they could not find *any* chart that appeared flawed, and so guessed randomly). Therefore, our experimental results do not distinguish between confusers and hallucinators. We intend to disambiguate these scenarios in future work. Speculatively, for data sanity checking tasks specifically, the false negative might be a more critical case (we don't want to inadvertently use dirty data). EDA tools may therefore value the absence of confusers more than the absence of hallucinators.

Lastly, we used naïve participants with no particular stakes in, or strong semantic connections with, the data. Statistical expertise, visual literacy, or strong priors from the analytical domain, could greatly improve performance at sanity checking. It remains an open research question to quantify how semantic information impacts performance at lower-level graphical perception tasks, and whether higher stakes in decision-making encourages fundamentally different patterns of information-seeking and verification.

8.2 Potential Solutions & Future Work

One solution to the issues raised in this paper is to integrate automatic anomaly detection with visualization directly. Tools like Profiler [27] use approaches from data mining to suggest potential data anomalies, and then use visualization to allow the analyst to diagnose their source, or assess their severity. We recommend a further integration of anomaly detection and visualization: anomaly detection might function as the visual analytics equivalent of "compiler warnings:" by automatically detecting anomalies that are not readily visible in particular visualization types (such as gaps in histograms), a system could prompt interaction or other mixed-initiative solutions (such as a focus+context view inside a particularly irregular histogram bin). This visualizations+warnings design could afford the speed and fluidity of sanity checking when the data quality issues are readily apparent, but encourage the analyst to slow down and interact when these issues are less prominent. Likewise, recommender systems such as Voyager [49] could use these automatic methods in order to provide explicit guidance to analysts to more closely examine problematic fields in a dataset. We intend to explore ways that automated anomaly detection methods can better integrate with visual analytics, and how to communicate potential data quality issues to analysts, especially as these issue occur further on in the analysis pipeline.

Another class of solutions is to employ hybrid or ensemble visualizations of distributions as the default view for data prep contexts. Bean plots [26] (with low opacity interior strips) represent a potential ensemble design with components that, in our study, afforded detectability across the full range of the data flaws we explored. Likewise, hybrid visualizations such as Wilkinson dot plots [47] or beeswarm plots [17], when the dataset is small enough, could afford some of the benefits of both scatterplots (in that individual points can be visible) and density plots (in that the overall shape of the distribution can be visible). We intend to explore this space in more detail, and empirically test our supposition that these sorts of encodings can provide the additive benefits of their components.

Our work has concrete implications not just for designers using standard visualizations in visual analytics contexts, but also for those designers creating and evaluating novel visualization techniques. We encourage designers not only to test the robustness of their design parameters in terms of the quality of their visualizations in normal scenarios, but to specifically consider whether data quality concerns are easily discoverable or detectable across the relevant parameter spaces. Ideally, designers would build their new techniques *defensively*, such that important data quality concerns are difficult to ignore, even across a wide (or perhaps adversarial) range of parameter settings.

8.3 Conclusion

In this work, we examine the capabilities of visualizations to act as sanity checks: simple visualizations that are meant to be used to rapidly confirm that a particular dimension of a dataset is relatively free from flaws. The sanity checking process may be brief, and the analyst may be discouraged from interacting with or otherwise altering the design of these preliminary visualizations. In this setting, we have shown that there is a wide class of visualizations that appear to plausibly summarize data, but make flaws in the data difficult to detect.

Acknowledgments

Omitted for review.

References

[1] D3: histogram.js. https://github.com/d3/d3-array/blob/master/src/histogram.js.

[2] R: bandwidth. https://stat.ethz.ch/R-manual/R-devel/library/stats/html/bandwidth.html.

[3] R: hist. http://stat.ethz.ch/R-manual/R-devel/library/graphics/html/hist.html.

[4] Tableau help: Create bins from a continuous measure. https://onlinehelp.tableau.com/current/pro/desktop/en-us/calculations_bins.html.

[5] Vega-lite: init.ts. https://github.com/vega/vega-lite/blob/cebdffb20517947bd6dadb040ffe80f00d5bfcb2/src/compile/mark/init.ts.

[6] F. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973.

[7] S. Belia, F. Fidler, J. Williams, and G. Cumming. Researchers misunderstand confidence intervals and standard error bars. *Psychological methods*, 10(4):389, 2005.

[8] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[9] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American statistical association*, 79(387):531–554, 1984.

[10] M. Correll and M. Gleicher. Error bars considered harmful: Exploring alternate encodings for mean and error. *IEEE transactions on visualization and computer graphics*, 20(12):2142–2151, 2014.

[11] M. Correll and M. Gleicher. The semantics of sketch: Flexibility in visual query systems for time series data. In *Visual Analytics Science and Technology (VAST), 2016 IEEE Conference on*, pp. 131–140. IEEE, 2016.

[12] M. Correll and J. Heer. Black hat visualization. In *Workshop on Dealing with Cognitive Biases in Visualisations (DECISIVe), IEEE VIS*, 2017.

[13] M. Correll and J. Heer. Surprise! bayesian weighting for de-biasing thematic maps. *IEEE transactions on visualization and computer graphics*, 23(1):651–660, 2017.

[14] T. N. Dang, A. Anand, and L. Wilkinson. Timeseer: Scagnostics for high-dimensional time series. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):470–483, 2013.

[15] Ç. Demiralp, M. S. Bernstein, and J. Heer. Learning perceptual kernels for visualization design. *IEEE transactions on visualization and computer graphics*, 20(12):1933–1942, 2014.

[16] E. Dimara, A. Bezerianos, and P. Dragicevic. Narratives in crowdsourced evaluation of visualizations: A double-edged sword? In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 5475–5484. ACM, 2017.

[17] A. Eklund. Beeswarm: the bee swarm plot, an alternative to stripchart. *R package version 0.1*, 5, 2012.

[18] D. Freedman and P. Diaconis. On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476, 1981.

[19] J. Fuchs, F. Fischer, F. Mansmann, E. Bertini, and P. Isenberg. Evaluation of alternative glyph designs for time series data in a small multiple setting. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 3237–3246. ACM, 2013.

[20] J. Heer and M. Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 203–212. ACM, 2010.

[21] J. L. Hintze and R. D. Nelson. Violin plots: a box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998.

[22] H. Hofmann, L. Follett, M. Majumder, and D. Cook. Graphical tests for power comparison of competing designs. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2441–2448, 2012.

[23] D. Huff. *How to lie with statistics*. WW Norton & Company, 2010.

[24] H. Ibrekk and M. G. Morgan. Graphical communication of uncertain quantities to nontechnical people. *Risk analysis*, 7(4):519–529, 1987.

[25] M. C. Jones, J. S. Marron, and S. J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433):401–407, 1996.

[26] P. Kampstra. Beanplot: A boxplot alternative for visual comparison of distributions. 2008.

[27] S. Kandel, R. Parikh, A. Paepcke, J. M. Hellerstein, and J. Heer. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 547–554. ACM, 2012.

[28] W. Kim, B.-J. Choi, E.-K. Hong, S.-K. Kim, and D. Lee. A taxonomy of dirty data. *Data mining and knowledge discovery*, 7(1):81–99, 2003.

[29] G. Kindlmann and C. Scheidegger. An algebraic process for visualization design. *IEEE transactions on visualization and computer graphics*, 20(12):2181–2190, 2014. doi: 10.1109/TVCG.2014.2346325

[30] A. Lunzer and A. McNamara. It aint necessarily so: Checking charts for robustness. *IEEE VisWeek Poster Proceedings*.

[31] J. Matejka, F. Anderson, and G. Fitzmaurice. Dynamic opacity optimization for scatter plots. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 2707–2710. ACM, 2015.

[32] J. Matejka and G. Fitzmaurice. Same stats, different graphs: generating datasets with varied appearance and identical statistics through simulated annealing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 1290–1294. ACM, 2017.

[33] L. Micallef, G. Palmas, A. Oulasvirta, and T. Weinkauf. Towards perceptual optimization of the visual design of scatterplots. *IEEE transactions on visualization and computer graphics*, 23(6):1588–1599, 2017.

[34] A. V. Pandey, J. Krause, C. Felix, J. Boy, and E. Bertini. Towards understanding human similarity perception in the analysis of large sets of scatter plots. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 3659–3669. ACM, 2016.

[35] B. U. Park and J. S. Marron. Comparison of data-driven bandwidth selectors. *Journal of the American Statistical Association*, 85(409):66–72, 1990.

[36] E. Peer, L. Brandimarte, S. Samat, and A. Acquisti. Beyond the turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, 70:153–163, 2017.

[37] K. Potter, J. Kniss, R. Riesenfeld, and C. R. Johnson. Visualizing summary statistics and uncertainty. In *Computer Graphics Forum*, vol. 29, pp. 823–832. Wiley Online Library, 2010.

[38] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, 2017.

[39] D. W. Scott. Kernel density estimators. *Multivariate Density Estimation: Theory, Practice, and Visualization*, pp. 125–193, 2008.

[40] D. W. Scott. Sturges' rule. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):303–306, 2009.

[41] S. J. Sheather. Density estimation. *Statistical science*, pp. 588–597, 2004.

[42] S. J. Sheather and M. C. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 683–690, 1991.

[43] M. Stone, D. A. Szafir, and V. Setlur. An engineering model for color difference as a function of size. In *Color and Imaging Conference*, vol. 2014, pp. 253–258. Society for Imaging Science and Technology, 2014.

[44] J. W. Tukey. *Exploratory data analysis*, vol. 2. Reading, Mass., 1977.

[45] S. VanderPlas and H. Hofmann. Spatial reasoning and data displays. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):459–468, 2016.

[46] H. Wickham, D. Cook, H. Hofmann, and A. Buja. Graphical inference for infovis. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):973–979, 2010.

[47] L. Wilkinson. Dot plots. *The American Statistician*, 53(3):276–281, 1999.

[48] L. Wilkinson, A. Anand, and R. Grossman. Graph-theoretic scagnostics. 2005. doi: 10.1109/INFVIS.2005.1532142

[49] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE transactions on visualization and computer graphics*, 22(1):649–658, 2016.