

# Exploratory Visualization Involving Incremental, Approximate Database Queries and Uncertainty

Danyl Fisher and Steven M. Drucker ■ *Microsoft Research*

A. Christian König ■ *Microsoft XCG*

**A**n important question in large-data analytics is this: How can users interact with incremental visualization—that is, queries that operate on progressively larger samples from a database? To answer that question, we examined how analysts interacted with visualizations employing the standard implementation of error bars over bar charts. This type of visualization’s shortcomings led us to explore alternative visualizations that express probability distributions: ways to picture the results of yet-incomplete computation. Such alternatives let users interact more easily with large datasets.

## The Need for Interactive Approximate Querying

Data is now born digital, generated from simulations, collected from click logs, and gathered from thousands of devices sensing and recording their environments. Computing power hasn’t kept pace with this growth. Often, researchers facing large-data-analysis problems carefully check their code, cross their fingers, and submit their query to the queue. They won’t know if the query was incorrect until the job terminates the next morning.

Processing data queries can take hours, as high-performance clusters churn through massive datasets. This slow throughput means that analysts

must carefully consider their choices—they might only be able to ask a few queries of a large dataset in the course of a study. This contrasts with exploratory visualization, in which analysts expect to try a long series of visualizations, exploring the relationships between dimensions, testing hypotheses, and freely pivoting through the data.

The queries that users issue in exploratory visualization differ fundamentally from those arising in noninteractive batch systems.

In exploratory systems, users will try riskier queries and can modify their subsequent queries with the previous ones in mind. We want, therefore, to support interactive queries that can give users rapid results.

Although tools such as column-oriented databases and massively parallel data engines such as Google’s Dremel can help slice through data more rapidly, there will likely always be situations with more data than computing time. A query that requires processing or examining a stream of images, for example, will typically be slow because it requires significant processing at each instance. A dataset with sparse points of

---

**Large datasets can mean slow queries, for which users must wait. Incremental visualization systems can give faster results at a cost of accuracy. This article asked analysts to use one and report on their results. Their feedback provides suggestions for alternative visualizations to represent a query still in progress.**

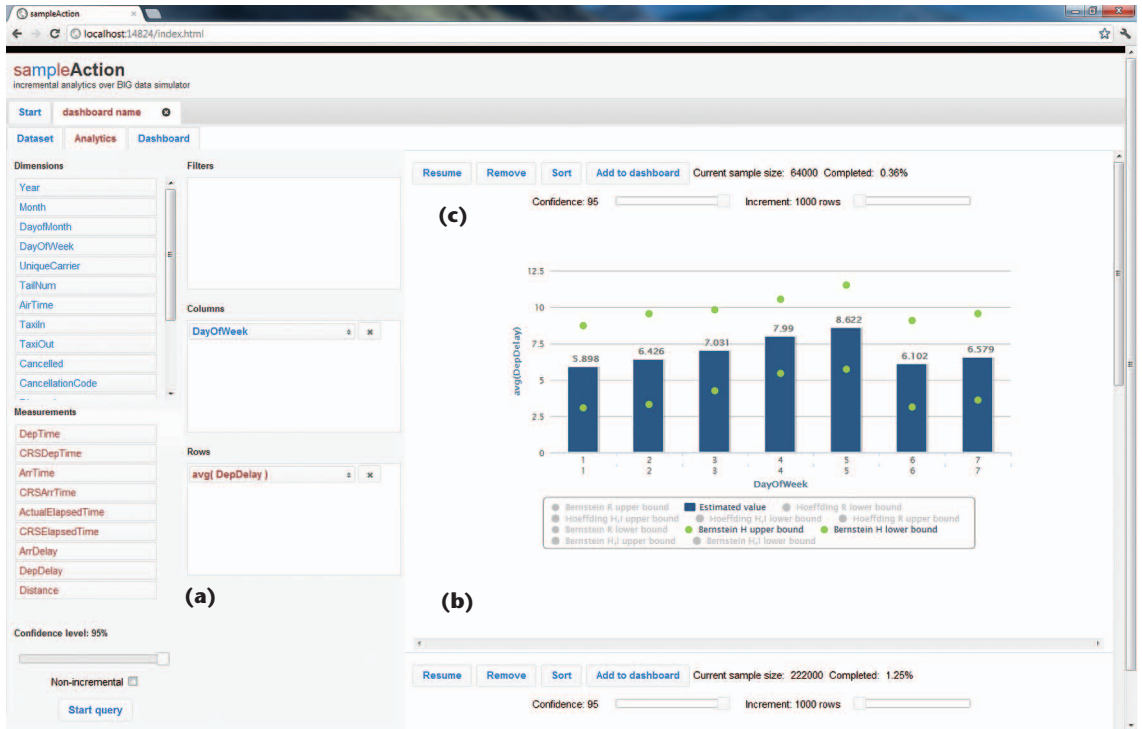


Figure 1. The analytics panel in sampleAction, showing an incremental visualization in progress. (a) The shelves let users select data dimensions. The user is looking at an FAA database of flight delays, examining average departure delay by day of the week. (b) In the visualization, small dots represent the 95 percent probability bounds. (c) The current status is that the visualization is showing estimates based on just 0.32 percent of the dataset. After seeing only a fraction of the dataset, the user can see that flights on Thursday and Friday are likely to be delayed by several minutes more than flights on Saturday and Sunday.

interest (such as searching through query logs) might similarly take much time to process completely.

Online query processing is meant to help enable exploratory data visualization on very large datasets. A series of incremental queries returns partial results obtained by processing progressively larger random samples from the underlying data. This approach is most successful with aggregate functions such as sums and averages. Although you can't compute a precise aggregate without looking at all the data, you can often approximate it with high accuracy by looking at only a small fraction of the data.

We can estimate this approximation's quality. An estimate's bounds are a function of the underlying dataset's variance. They shrink with the square root of the number of values seen to that point. So, we can present probability distributions, rather than fixed values, to users. Users can understand whether the result in their current sample will likely be definitive or whether they should wait longer for further results. This is a novel experience for users: visualization tools usually don't display distributions.

### Incremental Queries against Big Data

To explore how users interact with exploratory vi-

ualization, we developed sampleAction, a prototype system that presents a realistic experience to users.<sup>1</sup> The system incrementally updates samples from a real database, but at far smaller scales than we would expect from a production-class system. We run queries at interactive speeds, allowing us to explore designs. The back end uses a SQL engine with queries modified to return confidence bounds and row counts, in addition to the aggregate values.

A bar chart is a common way to visualize an aggregate query's results. One grouping variable serves as the x-axis; the aggregates' values become y values. For example, a bar chart might show the average time that airline flights are delayed, grouped by day of the week. Using approximate queries, we display an approximate aggregate: a confidence interval in which we expect the value to fall once the computation finishes. We can represent this confidence interval using traditional error bars, showing the range that covers 95 percent of the expected values (see Figure 1b). This is precisely the structure that previous research has used, including the Control project. (See the sidebar for more information on Control.)

The front end of sampleAction is a simplified Tableau-like (<http://tableausoftware.com>) interface that lets users visually construct aggregate

## Exploratory Visualization for Big Data

queries. Users can specify a series of independent dimensions and a measure. The system responds with partial results, displaying a bar chart with confidence bounds. As the analyst waits, the system increases its sample size every second, narrowing the confidence intervals and producing more precise results (see Figure 1). (Throughout this article, we use examples from the US Airline On-Time Performance dataset; <http://explore.data.gov/d/ar4r-an9z>.)

Our initial prototype uses column charts, following the Control project. Error bars show the confidence bounds around the data; the column height shows the estimated value. For example, in Figure 1, an analyst can conclude with 95 percent probability that the true average departure delay on Friday (day 5) is somewhere between 6 and 12 minutes, whereas Saturday is between 3 and 9 minutes. These conclusions are based on 56,000 rows—just 0.32 percent of the full database. An analyst can pause or stop the incremental process at any time. In the current implementation, analysts can also start additional queries while the previous ones are running. All queries will continue to add samples and slowly converge.

We intentionally slowed our simulator to learn more about incremental queries' behavior; it shows updates based on tens of thousands of rows per second. We expect that in a full implementation, we would see updates of millions of rows at a time.

### Interacting with Confidence Intervals

We wanted to understand how users interact with confidence intervals and to understand the weak points; this information would drive our next round of design. In particular, we looked to understand what aspects of the visualization made it difficult for users to interpret their data and what tasks users were trying to accomplish.<sup>1</sup>

### Setting Up the Study

We recruited three data experts from different areas in a large, data-intensive corporation who regularly work with large datasets. Bob's team manages server operations and generates static, visual reports of his system's performance once a day. Allan tracks marketing for online games; his team writes custom code based on a massive database to answer specific customer requests. Sam is a researcher working on social media; he analyzes large corpora of messages, looking for trends in emotion.

All three of them had expressed interest in visualizing data but hadn't been able to explore their data. Bob and Allan created noninteractive visualization tools based on static sets of queries. Sam

Exploratory data analysis is a way of learning about the characteristics of a dataset by looking at its various distributions and values. The statistician John Tukey promoted the idea,<sup>1</sup> which has inspired tools for rapidly visualizing data.

But the interactive nature of data exploration falls down with large datasets. We're by no means the first to suggest that exploratory visualization can be part of a large-data process. In the Control project, Joseph Hellerstein and his colleagues argued that incremental queries can help users quickly get satisfying results to long queries by returning approximate rather than precise answers.<sup>2</sup> Control emphasized the database implications of building systems that support iterative queries. It left open the question of how users interact with these incremental computations.

Currently, we don't know of back-end systems that implement interactive approximate queries. The closest, the commercial product Infobright,<sup>3</sup> generates a single-pass approximate value but doesn't support estimates that converge over time. Generally, the research community continues to work on ways to make incremental queries more efficient and approachable.

### References

1. J.W. Tukey, *Exploratory Data Analysis*, Addison-Wesley, 1977.
2. J. Hellerstein et al., "Interactive Data Analysis with Control," *Computer*, vol. 32, no. 8, 1999, pp. 51–59.
3. D. Slezak and M. Kowalski, "Towards Approximate SQL: Infobright's Approach," *Proc. 7th Int'l Conf. Rough Sets and Current Trends in Computing (RSCTC 10)*, Springer, 2010, pp. 630–639.

had tried to visualize his data but became overwhelmed by its scale.

We aimed to have these participants interact with data that they hadn't previously visualized but with which they were fairly familiar. To prime them for the study, we asked them to consider the sorts of questions they would ask of these datasets. Although we expected them to diverge from their usual habits, we wanted to start with familiar information, to evaluate the interface's effects rather than the learning curve on the data. Each participant provided us with approximately a million rows of sample data.

### Results

We trained each participant on the interface and then let him explore it on his own. We found that each participant ran a series of queries. Although the participants would terminate some queries rapidly, they would let others run for a time before making a decision. For example, Bob quickly realized that his data had an error when he saw a column he didn't expect. He found that the data had several error code values buried in it. To follow

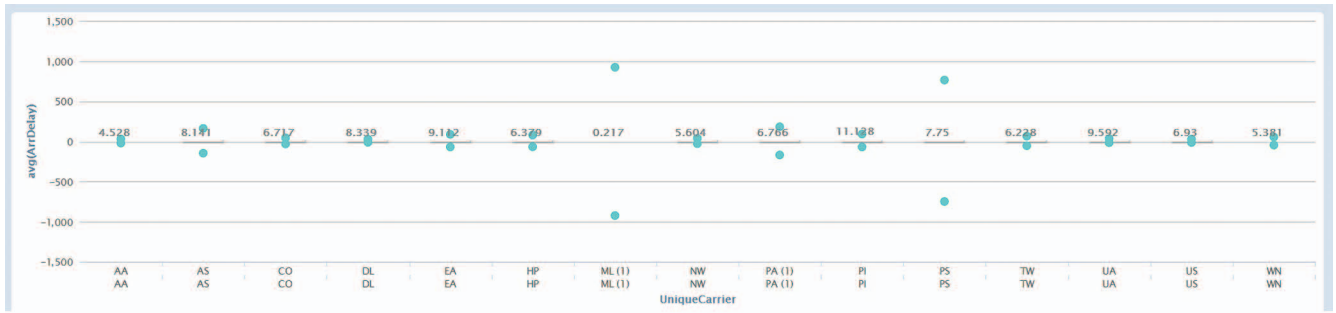


Figure 2. The average delay of different airlines across the dataset. Confidence intervals for a few airlines are wide, whereas most airlines have narrow confidence intervals at this stage. The bars are all but invisible owing to the huge intervals.

his line of investigation, he needed to repeatedly add filters to the query to remove the error codes.

Bob recognized that some of the logging his team was doing was wasted. One error was being triggered only after a different code had been triggered; the two had identical counts and occurred under the same circumstances. After discussing this problem with the team, he had a better sense of how to adjust its data-logging procedures to reduce redundancy.

Allan wanted to explore questions of the average age of players, categorized by country and game. He was surprised to find that some combinations of games and countries had startlingly different average ages. To explore this mystery further, he started a long series of queries, looking at possible factors that could correlate with age.

Sam had been working with a list of keywords of interest and correlating them with the frequency of other data. He noticed that one keyword was far more frequent than the others; when he removed that keyword from the set, other distributions on his data changed significantly. He realized he would have to readjust his experiment to account for this.

None of the three participants were accustomed to interactive queries. Each of them stumbled down blind alleys, made mistakes, and ended up issuing many queries (at least a dozen). This led to them exploring data that, in other circumstances, might have taken them weeks to explore—or, more likely, that they wouldn't have explored at all.

**User tasks.** Although the explorations differed considerably, all three participants had common building-block operations that they performed repeatedly. They often compared bars to check which one was highest. For example, Allan wanted to know which country had the oldest average players of a game. All three looked for outliers whose values differed dramatically from that of the others.

The participants often wanted to compare two bar charts. For example, Scott wanted to compare the distribution of posts having a given word to

the posts not having that word; Allan wanted to compare age distributions for two games. Because sampleAction doesn't directly support multiple series, the participants approximated this in several ways, including creating multiple queries at once.

**Obstacles to success.** Although the participants were able to complete their queries, they still ran into issues with the visualization. The most important issues related to scale and perception of error bars. Generally, early during the computation, confidence intervals can be broad—sometimes orders of magnitude wider than the estimated values' sizes. The participants sometimes faced a screen of large intervals with comparatively small values.

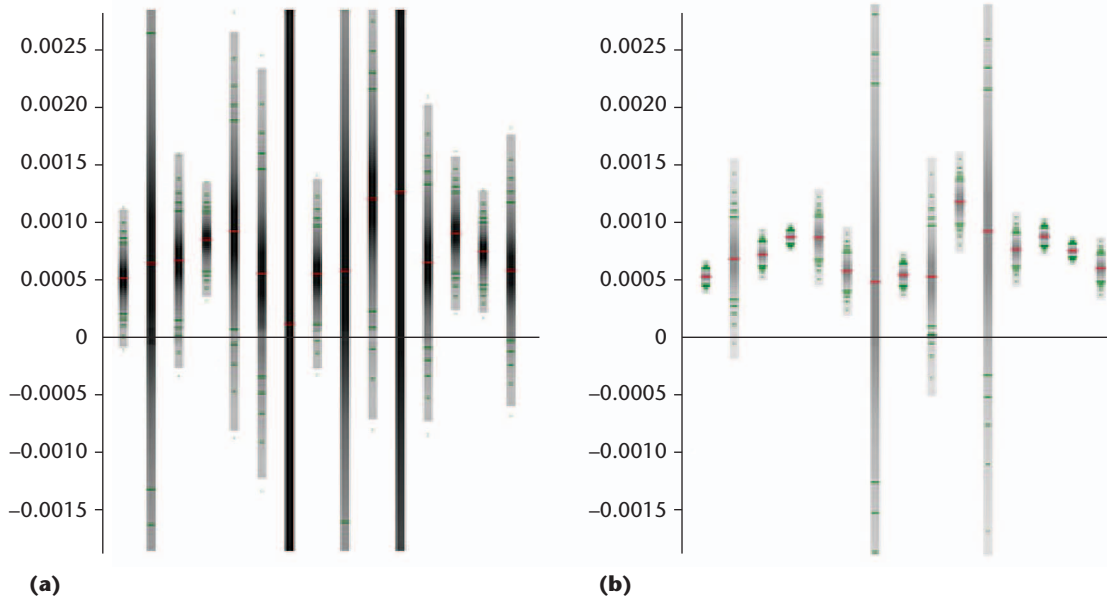
In Figure 2, for example, one airline, ML, has a confidence interval ranging from approximately -700 to +700. The other airlines have much tighter constraints and therefore display much smaller ranges. Worst, this image's most visually salient aspect is that the points with the least certainty are distinct from all the other points. Preferably, the data with the most-converged results would be the easiest to read.

The participants sometimes turned off the out-of-scale error bar display to track current estimates, but then would lose track of the range. Were they looking at a bar that had mostly converged (such as AA), had partially converged (such as AS), or was still very wide (such as PS)?

For all three participants, then, traditional error bars were a poor match for their tasks. For incremental visualization to be a realistic opportunity, we must find a visualization that can be adapted to the special constraints of uncertain results and animating data.

### Alternative Visualizations

In response to these challenges, we considered alternative designs that can help users compare distributions. Any alternative to a bar chart with error bars should continue to be effective even when distributions are on substantially different scales. We first explored representations that would both



**Figure 3.** Density strips (a) with black at the midpoint with a linear decrease and (b) modulated so that only the high-density levels are black. Subtle green tick marks represent percentiles at 10 percent intervals. The particular estimator of uncertainty used in this example has a large central portion that’s equally probable; consequently, the green bands are clustered at the edges.

carry out the required tasks and help clarify the nature of the uncertainty. On the basis of these issues, we established four basic design principles, which we can use to evaluate visualizations:

- *Reduce to a bar chart.* As a confidence interval’s size shrinks toward zero, the visualization should show an unambiguous (and familiar) single point.
- *Allow zooming.* If a bar doesn’t fit onscreen, users should understand whether the region they’re looking at is probable or improbable.
- *Allow comparison.* If users look at two bars side by side, they should be able to estimate which bar is more likely to have a smaller or larger value.
- *Map to animation well.* As the confidence interval converges, the visualization should change smoothly. A bar chart with error bars scores poorly on these criteria. Error bars are difficult to compare, except in extreme cases when the confidence intervals don’t overlap. A zoomed-in error bar can look like a vertical line without features and therefore isn’t safe at scale.

Inspired by a recent review,<sup>2</sup> we considered several representations with the probability distribution as a cumulative density function. Many contemporary types of visualizations are meant for distributions across a real sample and thus show individual values. Unlike traditional box plots, which are based on individual data points, these distributions result from a prediction function. Other researchers have also examined these

problems, suggesting a suite of possible uncertainty visualizations that modify bar, pie, and line charts with fuzzy uncertain zones as well as error bars.<sup>3</sup> One of the authors, Danyel Fisher, has described some of the strategies for large-data and uncertainty visualization.<sup>4</sup>

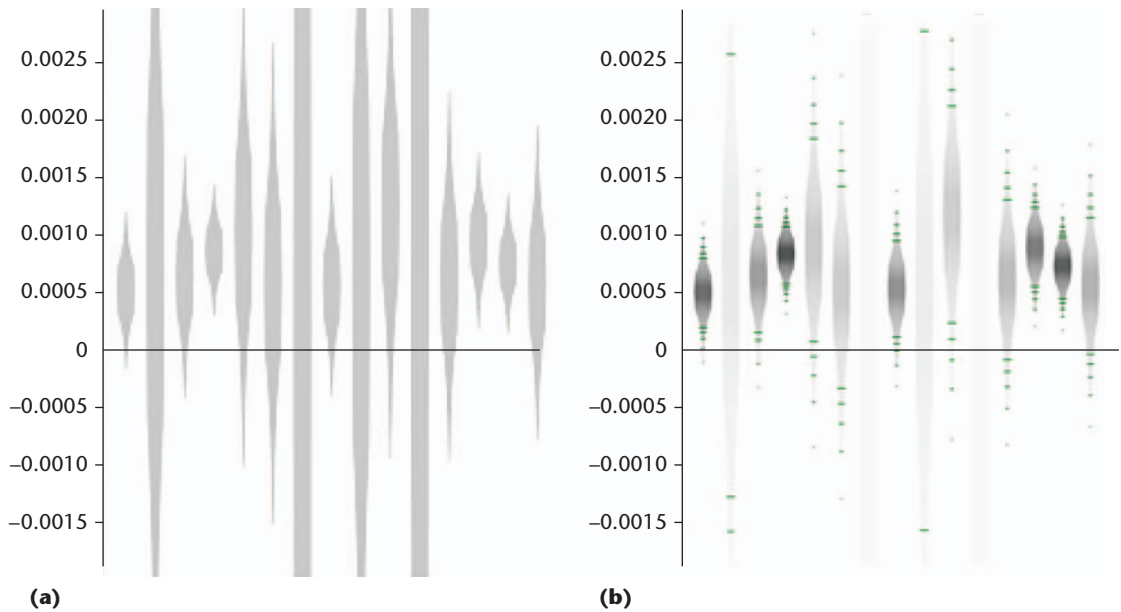
Computing a cumulative density function is a generalization of traditional error bars. Traditional error bars, such as the 90 percent range of Figure 1, mean that there’s a 90 percent chance that the final value will fall in this range and a 10 percent chance that it won’t. If we extend that value, we can compute the probability that the true value will be outside this point. We choose to look at the outside probability—the 10 percent, rather than the 90—to ensure that the function is at its highest value at the center.

We consider two particularly promising techniques for illustrating these distributions: *density strips* and *modified box-percentile plots*.

### Density Strips

One natural way to visualize uncertainty might be by mapping darkness to certainty—and, indeed, several different researchers have used gradients to represent uncertainty.<sup>5,6</sup> In our rendering, we set the highest probability value to be black, and allowed the bars to continue indefinitely outward (see Figure 3). Bars that are converged will have a small dark area, whereas bars that haven’t converged will have a much larger dark area.

Unfortunately, the dark areas overwhelm the visual area—the thick dark bars are more visible



**Figure 4. Box-percentile plots. (a) A plot drawn in full black has a visual salience that’s overwhelming. (b) Adding the color modulation from Figure 3 to the same shape produces a visualization suggesting a value in both width and saturation.**

than the nearly converged values. So, we add one more rule to our collection: *have appropriate visual salience*. Values that have a narrow confidence bound should be at least as easy to read as values that have a broader confidence bound.

Christopher Jackson suggested scaling the shading to the largest density across all the strips.<sup>6</sup> However, that wouldn’t apply to our animated visualization. Users would expect it to grow darker as the dataset converges. Instead, we chose a threshold we labeled “black.” We use the black level as a function of the bars’ estimate values.

#### **Modified Box-Percentile Plots**

We can also map confidence to bar width, perhaps in conjunction with gradient visualization. We again define the base thickness at the most probable point. At the 95 percent confidence level, we draw a line 5 percent as thick as the thickness at the center (see Figure 4). This visualization can be useful even when bar color serves a different purpose, and, to some users, could be more evocative of the notion of likelihood. This visualization is a variant of the box-percentile plot.<sup>7</sup>

Again, we find that the solid bars’ visual salience is overwhelming; this scheme spends more ink on uncertain bars than on certain ones. We adapt this visualization to add the color schemes of density plots (see Figure 4b) to get a visualization suggesting a value in both width and saturation.

#### **Annotating the Visualization**

A typical confidence interval suggests the breadth of ranges that it can display by placing one tick

at the center and additional caps at the interval’s ends. Unfortunately, if these three markers aren’t visible, users might not be able to tell where on the distribution their current value is. We add ribs to the gradients to show confidence intervals at 10 percent increments. These ribs help readers stay oriented in the bar.

All these variations reduce to a visualization much like a bar chart: a small thick area at the estimated value. All of them will work in an interface that includes zooming. (It’s perfectly acceptable for a bar to be taller than the screen.) Moreover, they all change smoothly as data shrinks, so they all allow smooth animation.

Figure 5 illustrates a dataset incrementally converging. At the start, all the bars are diaphanous and indistinct. As the visualization progresses, the bars look more solid. Bars with little data remain wide, whereas bars with precise estimates become tight and dark. The user can easily tell which data is currently interpretable; unconverged bars are still visible, but not overwhelming.

We aim to provide a visualization that lets users make decisions as incremental data streams in. So, it would be valuable for the tool to let them compare distributions. In general, the question to address is, how likely is it that this bar will converge on a greater value than that bar?

#### **A Direct-Comparison Tool**

If we relax the constraint on the results looking like a bar chart, an interesting possibility opens up for users to directly compare two distributions. The fundamental technique is to look at the relative

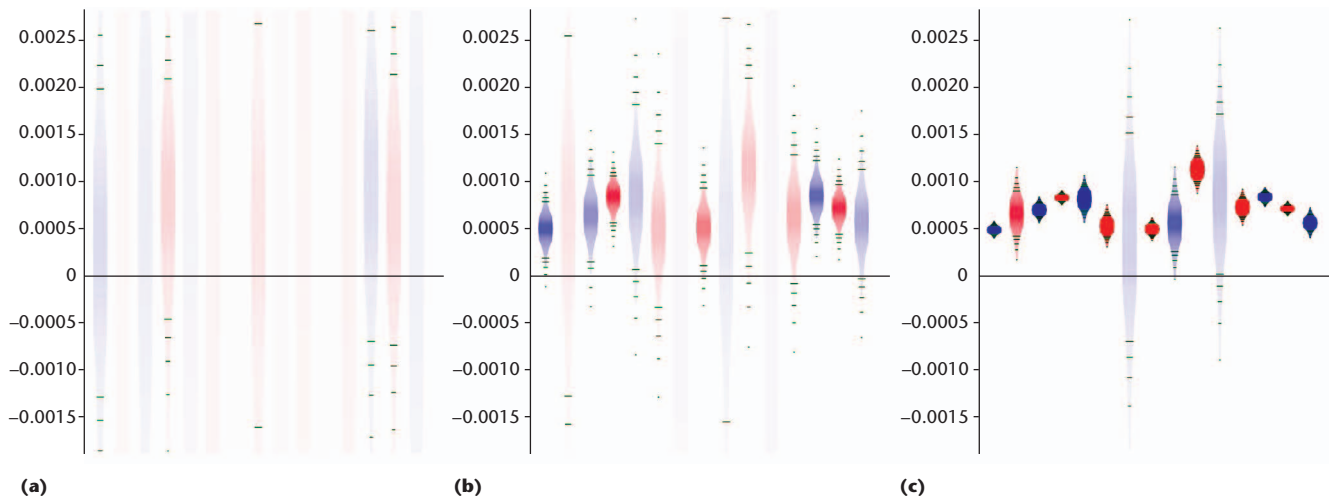


Figure 5. A tinted, shaped representation of the same query at three time intervals: (a) the very beginning, when all estimates are uncertain; (b) when several estimates have largely converged; and (c) when most estimates have converged.

probabilities of values. So, when computation starts, we might see that the airline UA has a 70 percent likelihood of having the highest value, whereas AA has only a 30 percent likelihood. As more data comes in, we might see this distribution change until the probability of one value being the largest converges to almost 100 percent. Such approaches can be extended to handle not just total ordering but also the difference’s magnitude (for example, “What’s the probability that  $x$  is 10 units larger than  $y$ ?”).

Our technique for finding this is inspired by techniques developed for experimental ranking and selection.<sup>8</sup> Those techniques take into account both the underlying distributions’ observed variance and the aggregate values’ magnitude.

Under some simplifying assumptions, we compute these direct comparisons via a convolution of the underlying value-probability distributions. We illustrate this for the comparison of two aggregate values  $V_1$  and  $V_2$ .  $D_1$  and  $D_2$  denote the corresponding cumulative probability distributions, which we define as follows. Let  $D_1(x)$  be the smallest value  $t$  such that the probability that  $V_1$  is larger than  $t$  is greater than or equal to  $x$ . (We define  $D_2(x)$  similarly.) So, when  $x$  is 0.025,  $D_1$  returns the lower bound of the 95 percent confidence interval; when  $x$  is 0.975,  $D_1$  returns the upper bound.

We then compute a discrete approximation of the convolution of  $D_1$  and  $D_2$  to quickly estimate the probability that  $V_1 > V_2$ . We do this by sampling  $D_1$  and  $D_2$  at periodic intervals and measuring for which fraction of all point pairs it holds that  $D_1 > D_2$ . Essentially, we’re approximating the probability with which one distribution will return a larger value than the other across all combinations of discrete intervals. Figure 6 illustrates this convolution technique.

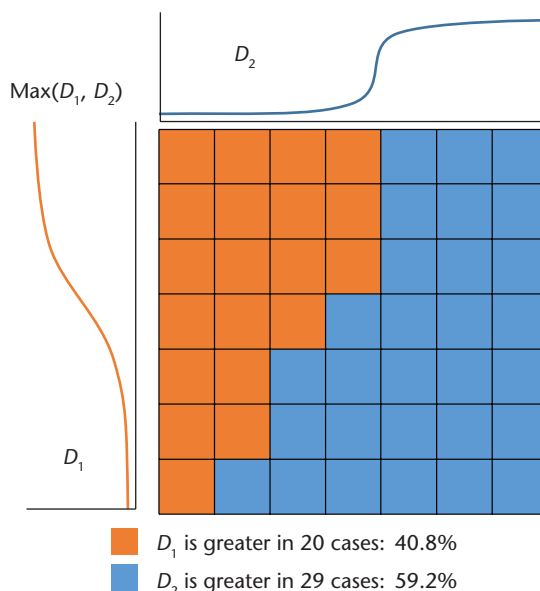
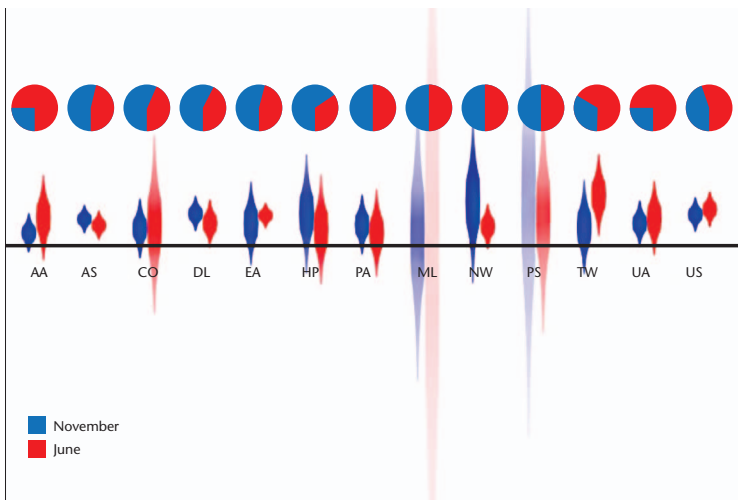


Figure 6. Convolution of distributions  $D_1$  and  $D_2$ . The distributions are illustrated for their value by probability. For probability ranges across the two distributions, values at which  $D_1$  is greater than  $D_2$  are red; those for which  $D_2$  is greater than  $D_1$  are blue. Of the 49 cells,  $D_2$  is greater than  $D_1$  in 29 of them. Thus, we would see that there is a 59.2% chance that the final, converged value of  $D_2$  will be greater than  $D_1$ .

After this set of samples is complete, we can compare the number of cases in which each value is greater than the other. We can represent this sum as a pie chart showing the chances that one distribution will produce a higher value than the other. Users can decide what level of certainty is sufficient for them to move ahead with their calculations.


We can extend this equation to find the maximum value of arbitrarily many distributions. We



**Figure 7. Comparing flight delays by airline for November and June. Miniature pie charts show the probability of a flight delay. Although pairs that haven't yet converged, such as ML, are still difficult to predict, the estimator can make a better guess for converged pairs.**

do this simply by looking for the maximum value of each possible combination of the multiple distributions (although at substantial computational cost).

We can present the output of this computation as a miniature pie chart. Such charts show the chances that a given bar will end up being highest. In Figure 7, the small pies are part of a clustered bar chart, comparing columns. In other systems, users might manually choose pairs of columns to compare.

**W**e invite the community to explore ways to maintain interactive speeds—and thus exploratory techniques—even as data moves to larger computation. Although this article focuses on incremental and approximate computation and visualization, other strategies and architectures for ensuring queries that produce rapid results are possible. The age of big data shouldn't bring us back to the techniques of punch-card computing. Rather, as data moves to the cloud and the cluster, we must search for ways to ensure that exploratory visualization's powerful techniques follow it. 


**References**

1. D. Fisher et al., "Trust Me, I'm Partially Right: Incremental Visualization Lets Analysts Explore Large Datasets Faster," to be published in *Proc. ACM Conf. Human Factors in Computing Systems (CHI 2012)*, ACM, 2012.
2. H. Wickham and L. Stryjewski, "40 Years of Boxplots," *Am. Statistician*, preprint, 2011; <http://vita.had.co.nz/papers/boxplots.html>.
3. C. Olston and J. Mackinlay, "Visualizing Data with Bounded Uncertainty," *Proc. IEEE Symp. Information Visualization*, IEEE, 2002, pp. 37–40.
4. D. Fisher, "Incremental, Approximate Database Queries and Uncertainty for Exploratory Visualization," *Proc. IEEE Symp. Large-Scale Data Analysis and Visualization (LDAV 11)*, IEEE, 2011, pp. 73–80.
5. A. Streit, B. Pham, and R. Brown, "A Spreadsheet Approach to Facilitate Visualization of Uncertainty in Information," *IEEE Trans. Visualization and Computer Graphics*, vol. 14, no. 1, 2008, pp. 61–72.
6. C.H. Jackson, "Statistical Computing and Graphics: Displaying Uncertainty with Shading," *Am. Statistician*, vol. 62, no. 4, 2008; doi:10.1198/000313008X370843.
7. W.W. Esty and J.D. Banfield, "The Box-Percentile Plot," *J. Statistical Software*, vol. 8, no. 17, 2003, pp. 1–14.
8. S.-H. Kim and B.L. Nelson, "Selecting the Best System: Theory and Methods," *Proc. 2003 Winter Simulation Conf.*, vol. 1, IEEE, 2003, pp. 101–112.

**Danyel Fisher** is a researcher at Microsoft Research. His research interests include new ways to explore and interact with data. Fisher has a PhD in information and computer science from the University of California, Irvine. He's a member of IEEE and a senior member of ACM. Contact him at [danyelf@microsoft.com](mailto:danyelf@microsoft.com).

**Steven M. Drucker** is a principal researcher and the manager of the VUE group at Microsoft Research and an affiliate professor in the University of Washington's computer science and engineering department. His research interests focus on human-computer interaction for dealing with large amounts of information. Drucker has a PhD from the Media Lab at the Massachusetts Institute of Technology. He's a member of IEEE and a senior member of ACM. Contact him at [sdrucker@microsoft.com](mailto:sdrucker@microsoft.com).

**A. Christian König** is a researcher in Microsoft's Extreme Computing Group. His research interests include data management and exploration, database management, algorithms for large-data analytics, and information retrieval. König has a PhD in computer science from Saarland University. Contact him at [chrisko@microsoft.com](mailto:chrisko@microsoft.com).

 Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.