Figure construction for t-digest

Ted Dunning

Summary. This short paper describes how to recreate the figures in the larger article

1. Introduction

The figures in the t-digest paper are generated using a variety of Java test cases, R scripts and hand-drawn figures. To regenerate these figures generally requires that you run the Java test cases to produce data that is reduced by R and then plotted in pdf form.

This document describes how to replicate all of the figures. The figures are included at the end of this document for reference together with their original captions.

Note that all figures must be moved to the `docs/t-digest-paper/figures` directory to be available when formatting the paper.

2. Figure 1

Figure 1 includes `k-q-plot.pdf` which is produced by `./docs/t-digest-paper/k-q-plot.r` without using any external data.

3. Figure 2

Figure 2 is based on `linear-interpolation.pdf` which is produced by the R script `linear-interpolation.r` found in `./docs/t-digest-paper/`. No external data is read by this script, but the sizes of the clusters on line 46 were taken from an actual t-digest.

4. Figures 3, 4 and 5

Figures 3, 4 and 5 were all hand-drawn in Omnigraffle. The source for all of them is in `quantile-figures.graffle`. Figure 3 is found on canvas 5, Figure 4 is on canvas 4 and Figure 5 is on canvas 6. To recreate the figures themselves, use File > Export to bring up a dialog that will write all artboards to separate PDF files. The LaTeXsource code assumes that these figures will be in the directory `figures` and that directory has to be set during the export process.

5. Figure 6

Figure 6 illustrates error rates for different scale functions and values of q. Production of the two graphs in this figure requires a series of steps to first generate the data in question and then to process and plot the data. Note that this figure will vary a fair bit each time you generate new data. This is due to the fact that the number of iterations used to generate the data is pretty small in order to keep the time required for the test to a reasonable level.

The data is generated by `AccuracyTest.testAccuracyVersusCompression()` in the `quality` module. The data is written to the directory `quality/tests` with a pathname composed using the "experiment" and the git hash of the current version of the code. This structure allows you (and me) to generate the same figure for different versions of the code.

To generate this figure, you need to do the following steps:

(a) In the `quality` sub-directory, run `AccuracyTest.testAccuracyVersusCompression`. This will produce a lot of measurements into `quality/tests` and should take a few minutes. Note that running *all* of the tests in `quality` will take tens of minutes.

(b) look at the most recently generated files in `quality/tests` to find out the current git hash. As an example, here is an extract of what I just saw when doing this

```
$ cd quality
$ ls -rt tests
    ...
accuracy-tree-digest-76e0eb8670.csv
accuracy-sizes-tree-digest-76e0eb8670.csv
accuracy-cdf-tree-digest-76e0eb8670.csv
accuracy-sizes-digest-a09d38d1d1.csv
accuracy-cdf-digest-a09d38d1d1.csv
accuracy-digest-a09d38d1d1.csv
```

From this, I know that my current tag is `a09d38d1d1`.

(c) in R, read this data and plot it. To read the data, we specify an experiment (digest, in this case) and git hash which we learned in the previous step.

```
$ cd quality
$ R

R version 3.5.2 (2018-12-20) -- "Eggshell Igloo"
Copyright (C) 2018 The R Foundation for Statistical Computing
     ...
Type 'q()' to quit R.
```

```
> source('accuracy.r')
> read.data('digest', 'a09d38d1d1')
> draw.relative.error.fig()
foo MergingDigest-K_1-weight-alternating-twoLevel UNIFORM unsorted 100 NA
624 10
foo MergingDigest-K_2-weight-alternating-twoLevel UNIFORM unsorted 100 NA
650 10
foo MergingDigest-K_3-weight-alternating-twoLevel UNIFORM unsorted 100 NA
624 10
foo MergingDigest-K_1-weight-alternating-twoLevel UNIFORM unsorted 100 NA
624 10
foo MergingDigest-K_2-weight-alternating-twoLevel UNIFORM unsorted 100 NA
650 10
foo MergingDigest-K_3-weight-alternating-twoLevel UNIFORM unsorted 100 NA
624 10
null device
          1
> q()
Save workspace image? [y/n/c]: n
$
```

This should generate `relative-error.pdf` which can by copied to `docs/t-digest-paper` to include it in the paper.

6. Figure 7

Figure 7 is based on `cluster-spread.pdf` which is, in turn, produced using data generated by `AccuracyTest#testBucketFill()` in a similar manner as above. The underlying data is not versioned so the necessary R code is relatively simple:

```
> source('accuracy.r')
> read.slow.data()
> draw.bucket.spread()
```

The data for this figure is about 1.5GB in size so just reading the data for this diagram takes quite a while. Plotting the figure isn't quick, but it is much faster than reading the data.

7. Figure 8

Figure 8 uses the data in `quality/tests` similarly to other figures. It can be drawn using this R code:

```
> read.data('digest', 'a09d38d1d1')
> draw.accuracy.fig()
```

8. Figure 9

Figure 9 is produced by the script `quality/merge.r` which processes data produced by the test `com.tdunning.tdigest.quality.AccuracyTest#merges`. All that is required to produce the diagram is run the test and then run the R script, both with a working directory of `quality/merge`. The diagram is stored in `merge.pdf` which must be moved to the `docs/t-digest-paper/figures` manually.

9. Figure 10

Figure 10 is produced by the `quality/comparison.r` script. This script consumes data produced by the `com.tdunning.tdigest.quality.ComparisonTest#compareToQDigest` unit test. Again, running the unit test and then the R script in the `quality` directory will produce the `qd-sizes.pdf` figure which must then be manually moved to the `docs/t-digest-paper/figures` directory.

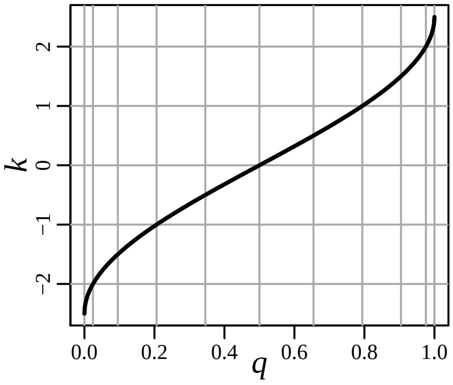

Fig. 1. The scale function translates the quantile q to the scale factor k in order to give variable size steps in q. Limiting cluster sizes allows better accuracy near $q = 0$ or $q = 1$.

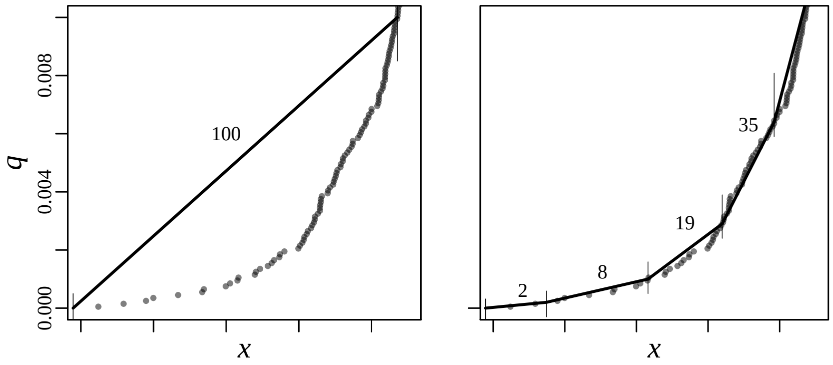

Fig. 2. The left panel shows linear interpolation of the cumulative distribution function near $q = 0$ with 100 equal sized bins applied to $10,000$ data points sampled from an exponential distribution. The right panel shows the same interpolation with variable size bins as given by a strongly ordered t-digest with $\delta = 100$. The numbers above the data points represent the number of points in each bin.

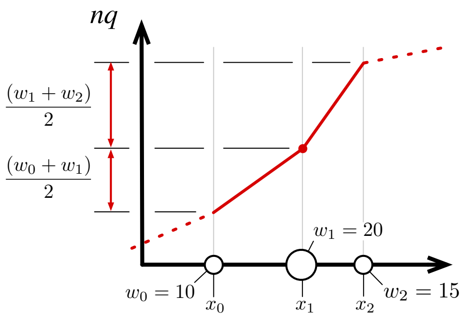

Fig. 3. Interpolation of the empirical cumulative distribution function between centroids of clusters with more than one sample is done by assuming half of the points for each centroid are to the left of the centroid and half are to the right.

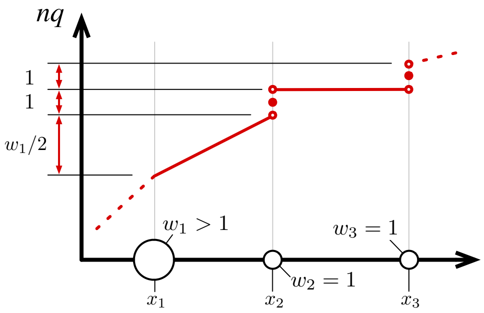

Fig. 4. Clusters with a single sample are handled specially. Adjacent to a normal cluster, as is the case between x_1 and x_2, interpolation is done assuming $w_1/2$ samples occur between x_1 and x_2, and the single sample at x_2. That single sample causes the cumulative distribution to step to the mid-point of the individual sample at x_2. Between singleton clusters x_2 and x_3, the cumulative distribution is given a constant value until it steps again at x_3.

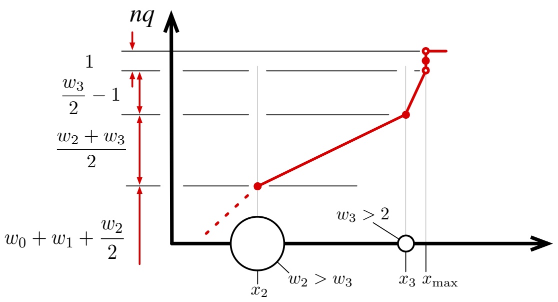

Fig. 5. For the last clusters, if the cluster weight is greater than 2, we can use the fact that a singleton must occur at $x_{\min}$ or $x_{\max}$ to improve interpolation accuracy. The first cluster is handled by reversing these steps.

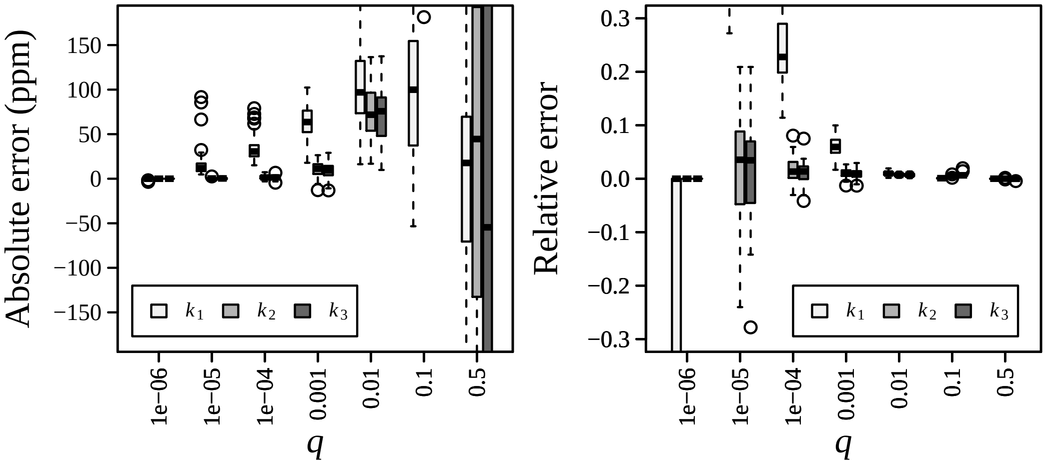

Fig. 6. The left panel shows absolute error in parts per million of estimations of quantile with $\delta = 100$ for scale functions k_1, k_2 and k_3. Errors are computed by taking 10^6 samples from a uniform distribution and comparing estimates from a t-digest against exact quantiles computed from the original samples. This was repeated 50 times to get a sense of variability. The left panel shows absolute error in parts per million and the right panel shows relative error.

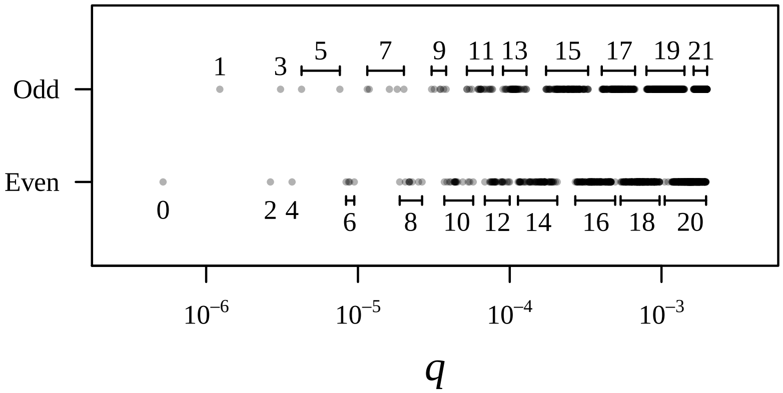

Fig. 7. The first few clusters in a typical t-digest show that while the digest is not strictly ordered, only adjacent clusters overlap. This plot shows roughly the first 20 clusters of the digest with even-numbered clusters below and odd-numbered ones above. Note how odd clusters overlap with adjacent even clusters, but never with adjacent odd clusters. This corresponds to weak ordering with $\Delta = 1$

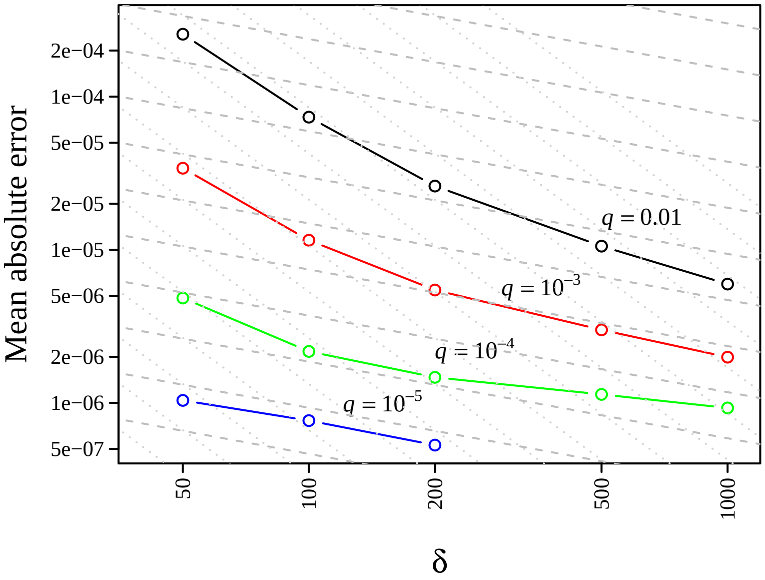

Fig. 8. The scaling of quantile estimation for various values of compression factor δ and q. The general pattern is that absolute error scales like $1/\delta^2$ for small values of δ and like $1/\sqrt{\delta}$ for values larger than some cutoff. The cutoff is higher for larger values of q (roughly 500 for $q = 0.01$ but only 100 for $q = 10^{-5}$). The grey dashed lines provide a reference for $1/\sqrt{\delta}$ scaling, the dotted grey lines show $1/\delta^2$. At $q = 10^{-5}$ error goes to zero for values $\delta > 200$ and cannot be shown on a log scale. The same happens for $q = 10^{-6}$ except error is zero for all values of δ. The data shown here are for k_2, but the results for k_3 are nearly indistinguishable.

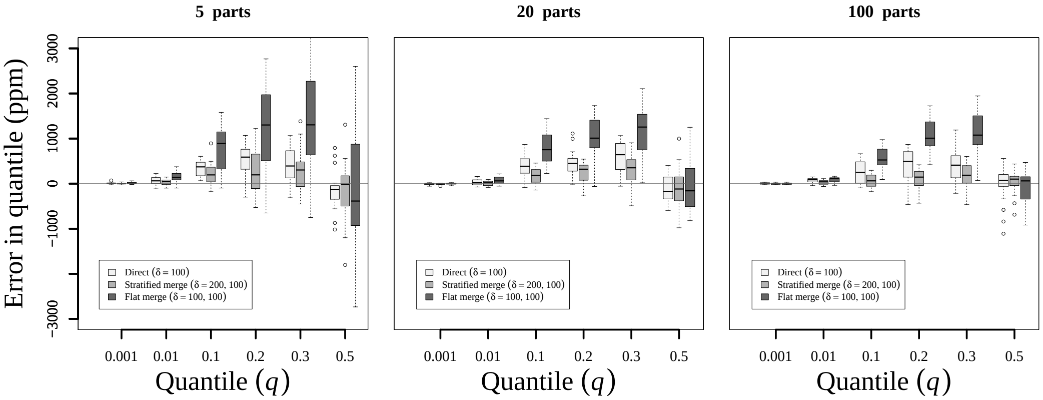

Fig. 9. Accuracy of a t-digest accumulated directly is nearly the same as when the digest is computed by combining digests from 5, 20 or 100 equal sized paritions of the data. All panels were computed by 20 repetitions of aggregating 1,000,000 values.

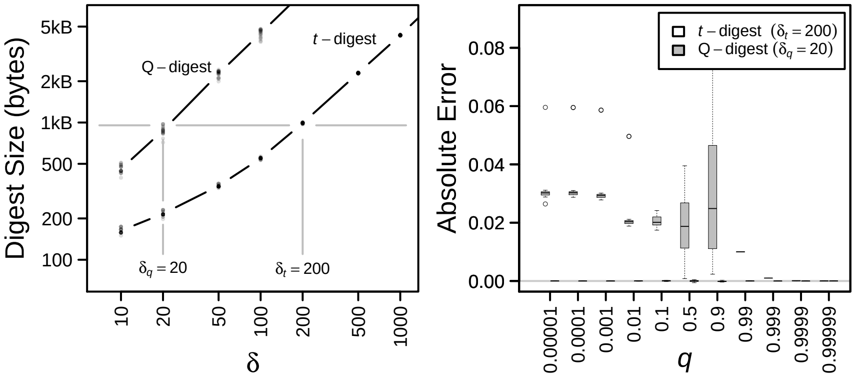

Fig. 10. The left panel shows the size of a serialized Q-digest and t-digest versus compression parameter δ for 100,000 uniformly distributed samples. Trials were repeated 20 times. The left panel shows how compression parameters δ_q and δ_t are chosen to control digest size to approximately 1k byte. The right panel shows absolute error for various values of q. With this vertical scale, t-digest errors are only barely distinguishable from zero near the median.