

Approximation of Distributions via T-Digests

Dr. Alexander Weiß

ALGORITHMS DATA CHALLENGES 

Two algorithms walks into bar... part I

Zalando

March 5, 2015

Improved Formula



Health King 康帝®

Digest

(Caffeine-Free)

Herb Tea™

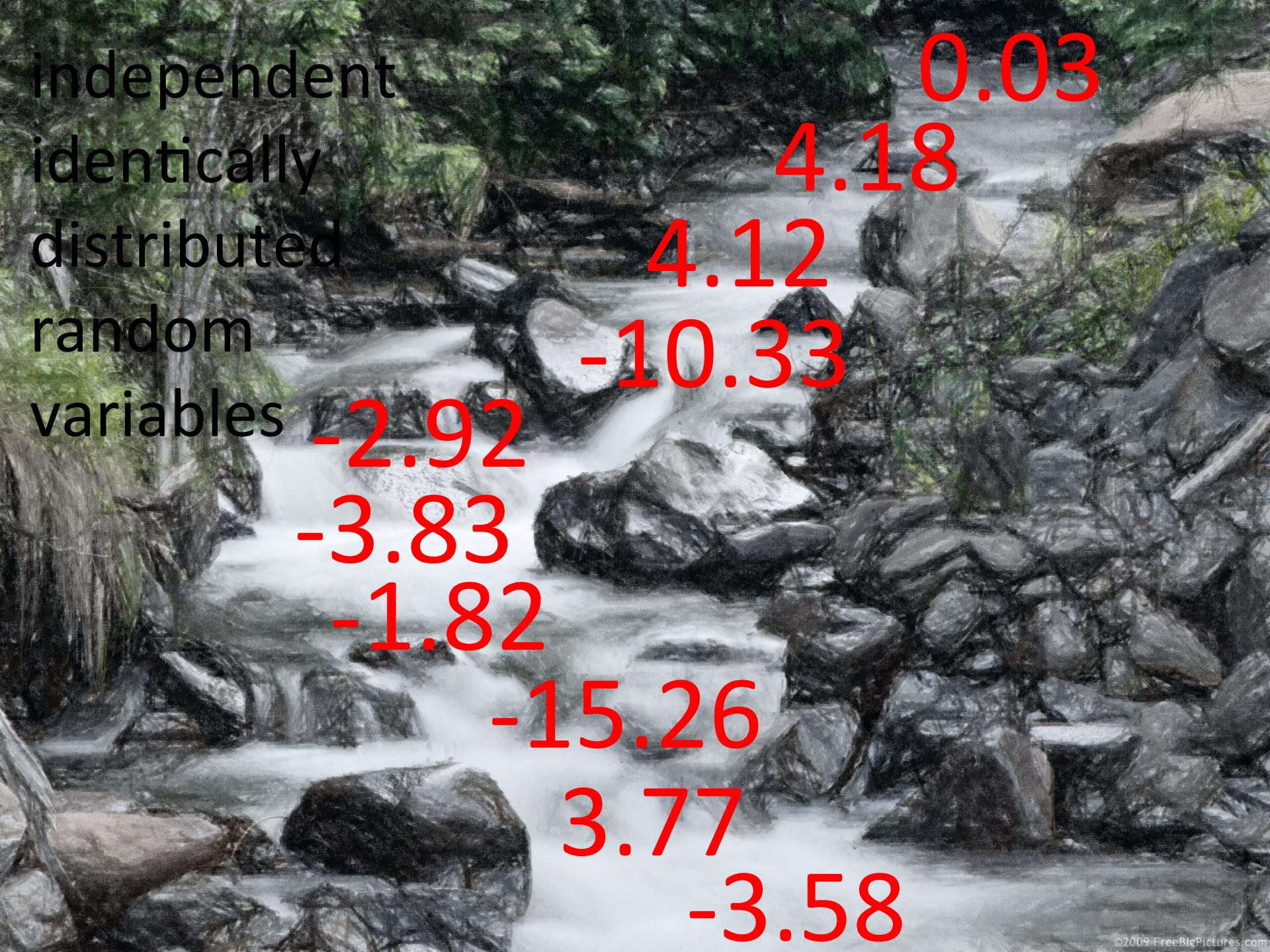
助消化茶

Help
digestion,
clear liver
heat &
increase
acidity

100% NATURAL

20 Tea bags

Net Wt: 1.26 oz (36 g)

A black and white photograph of a rocky stream flowing through a forest. The water is turbulent, creating white foam as it flows over and around large, dark rocks. The banks of the stream are covered with dense green foliage and trees. The overall scene is natural and somewhat rugged.

independent
identically
distributed
random
variables

0.03
4.18
4.12
-10.33
-2.92
-3.83
-1.82
-15.26
3.77
-3.58

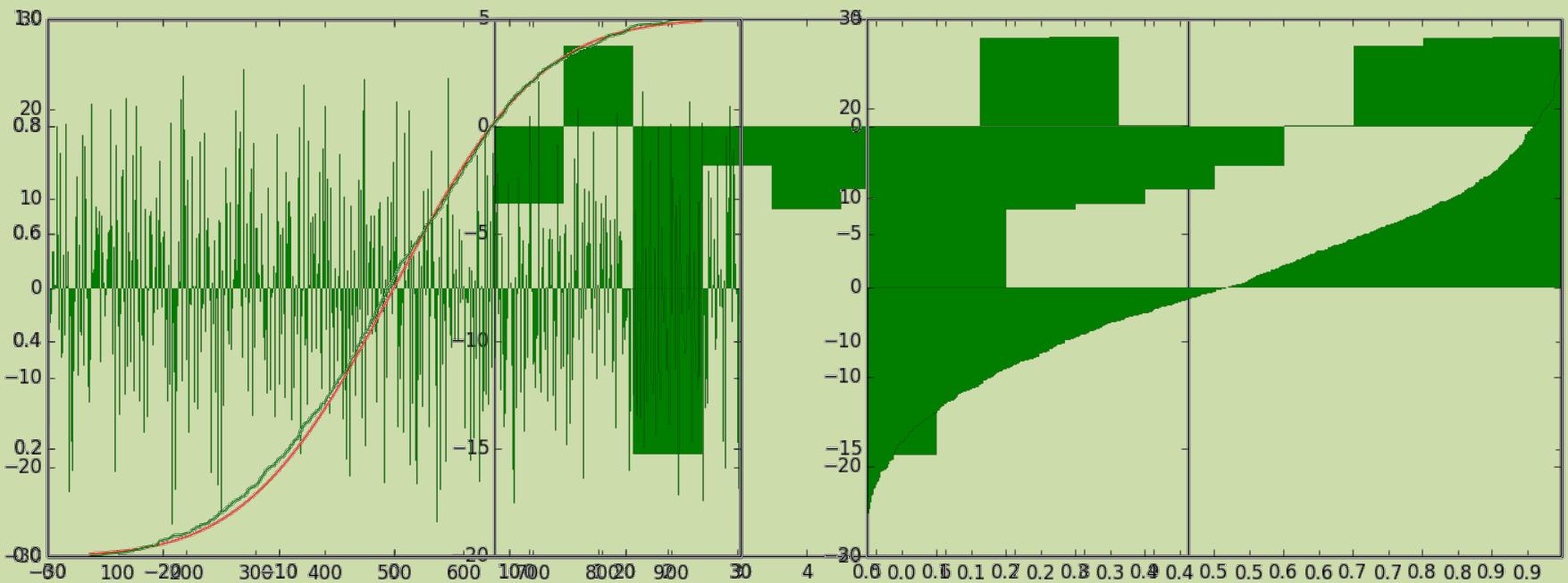
What do I want to know?



Moments? Quantiles? Distribution!

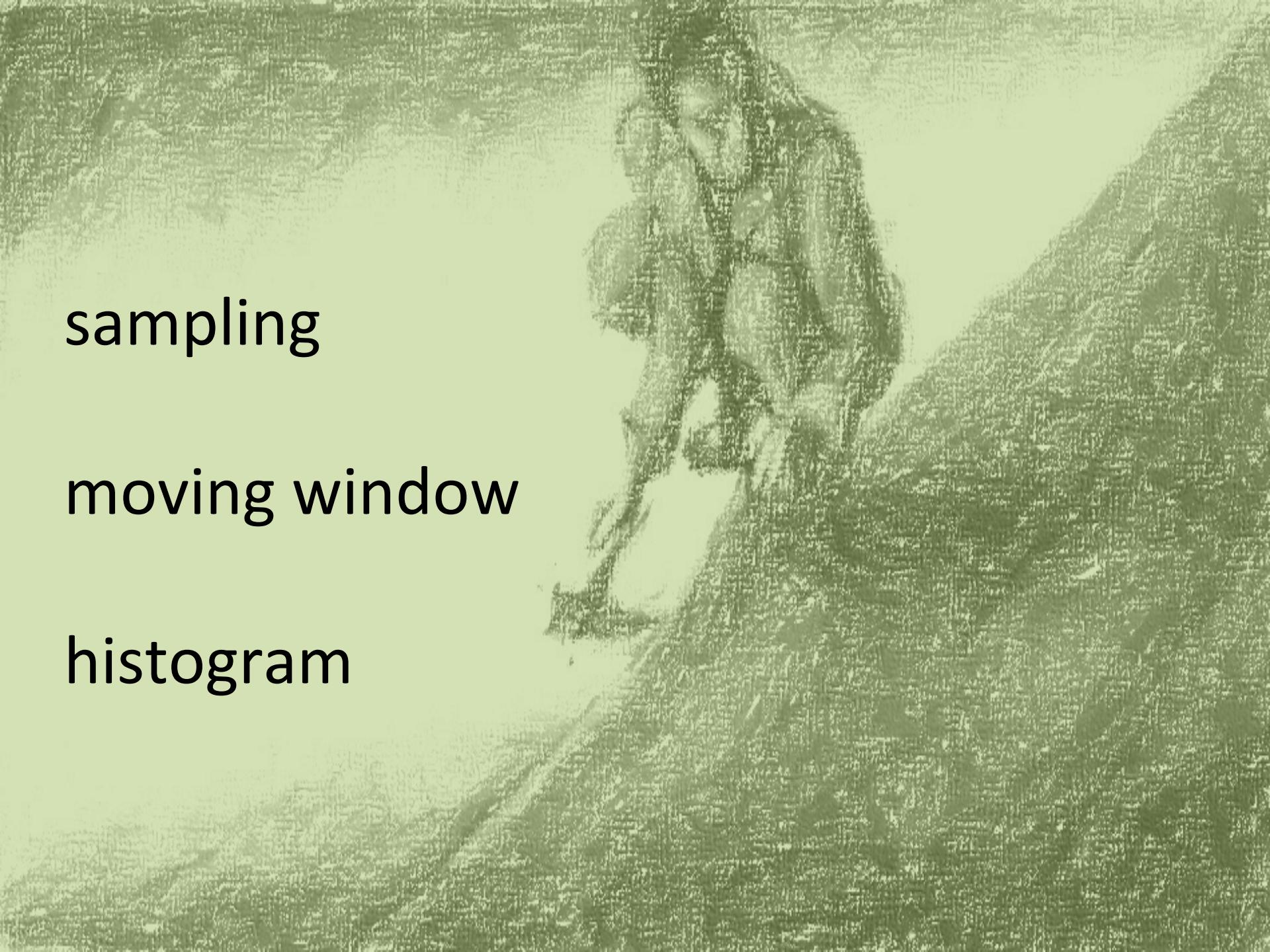
Lossless approach

-3.58, 3.77, -15.26, -1.82, -3.83, -2.92, -10.33, 4.12, 4.18, 0.03, ...





KEEP
CALM
AND
USE
COMPRESSION



sampling

moving window

histogram

T-Digests is ...

a mixture: online clustering / histogram

applicable for real number streams

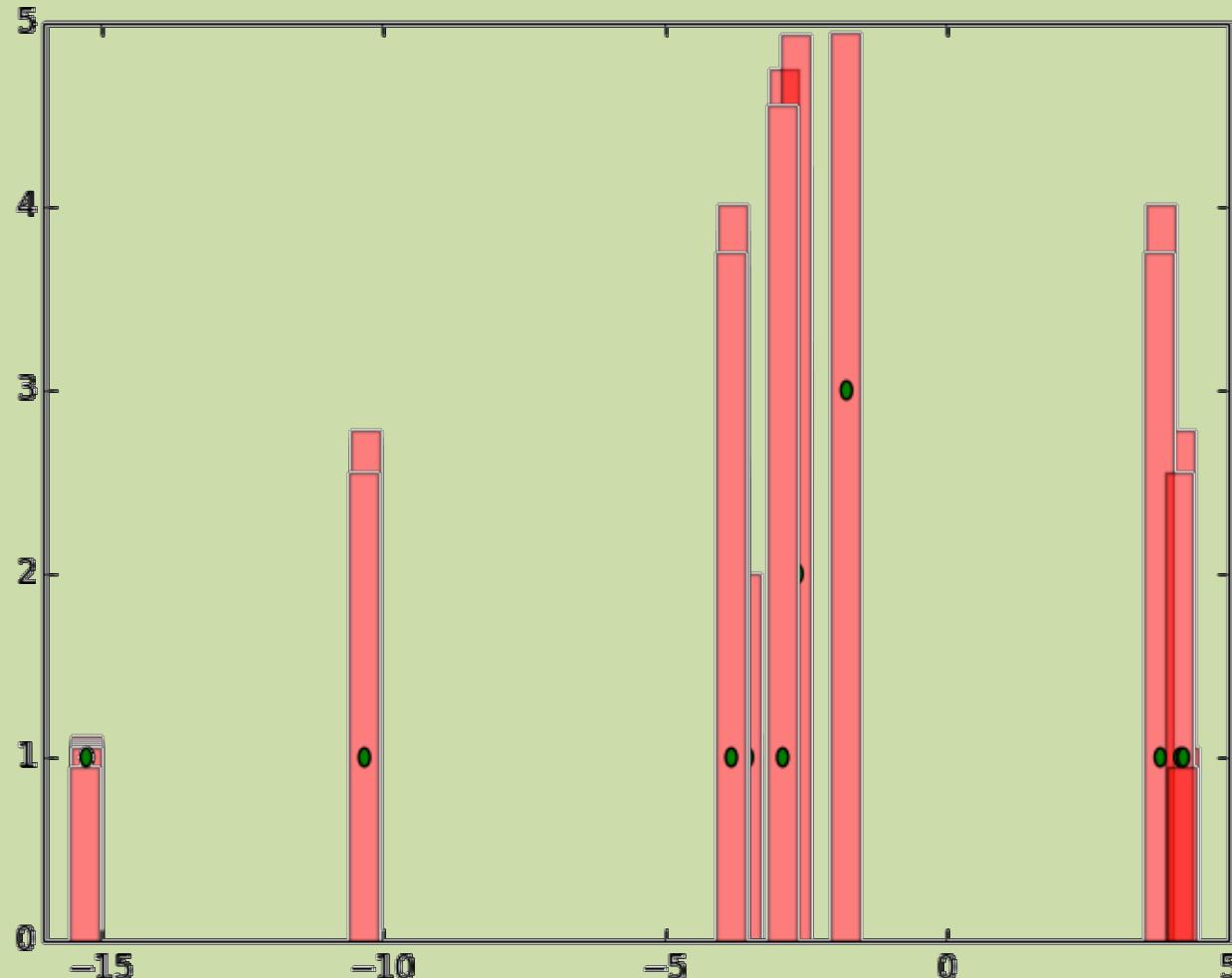
especially accurate for extreme quantiles

parallelizable

t-Digests step by step

-3.58, 3.77, -15.26, -1.82, -3.83, -2.92, -10.33, 4.12, 4.18, 0.03

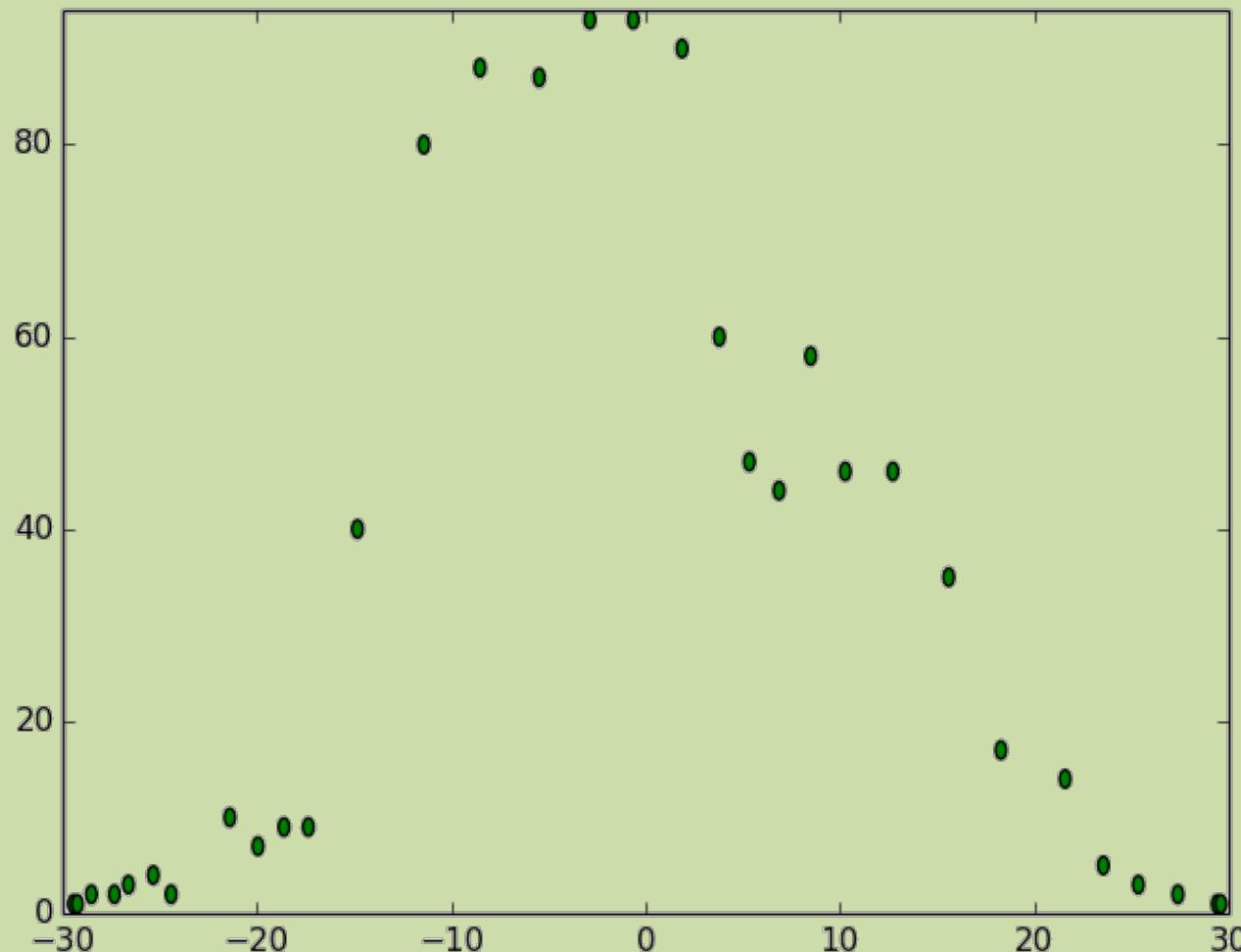
(-15.26, 5.12, 6.51, 0.63, 3.01, 1.63, 0.08, 0.38, 0.32, 0.29, 0.21, 0.19, 0.23, 0.12, 0.27, 0.31, 0.17, 0.11, 0.12, 0.41, 0.24, 0.18, 1)



t-Digests step by step

32 centroids

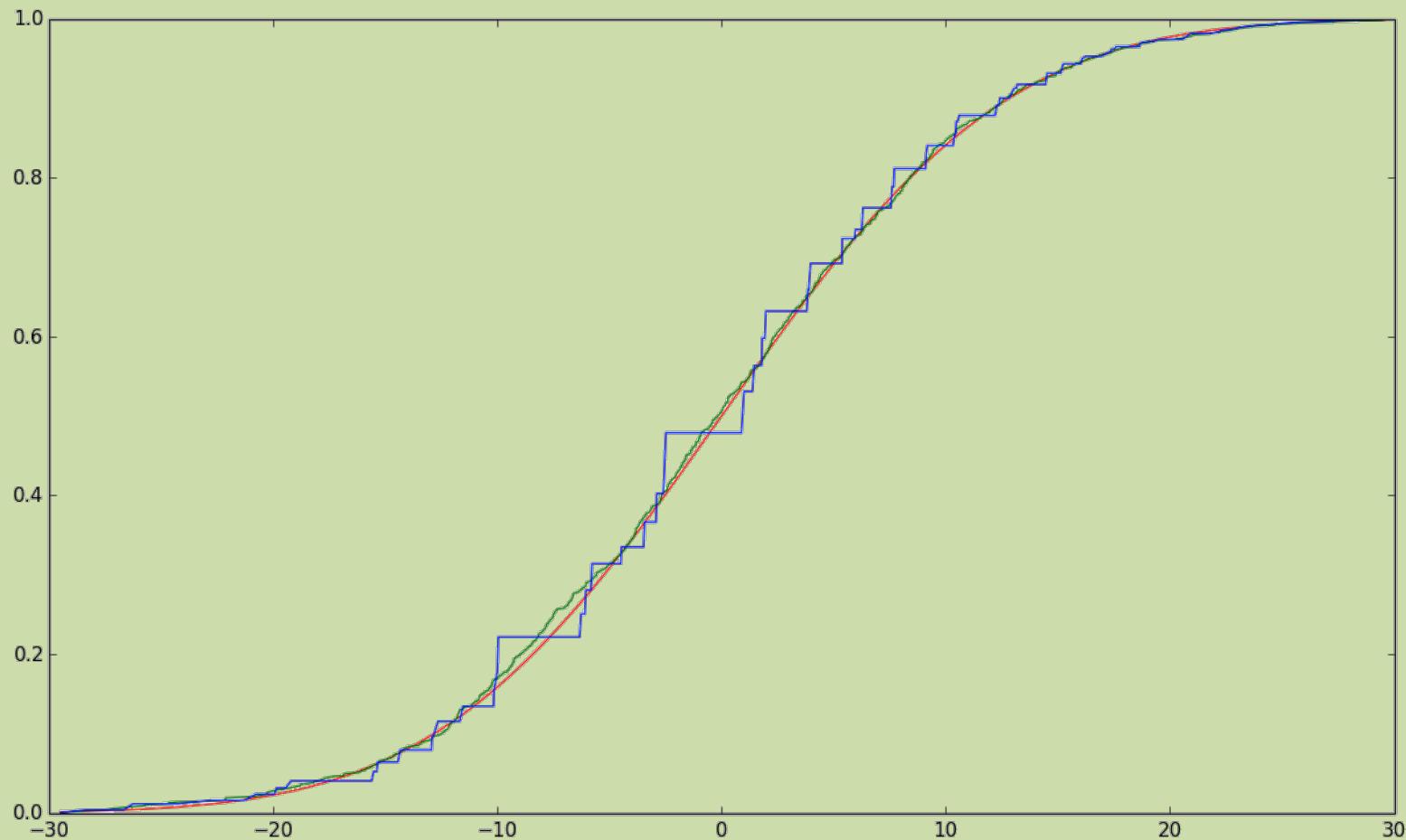
-3.58, 3.77, -15.26, -1.82, -3.83, -2.92, -10.33, 4.12, 4.18, 0.03, ...



t-Digests step by step

32 centroids

-3.58, 3.77, -15.26, -1.82, -3.83, -2.92, -10.33, 4.12, 4.18, 0.03, ...



Capacity of a centroid c

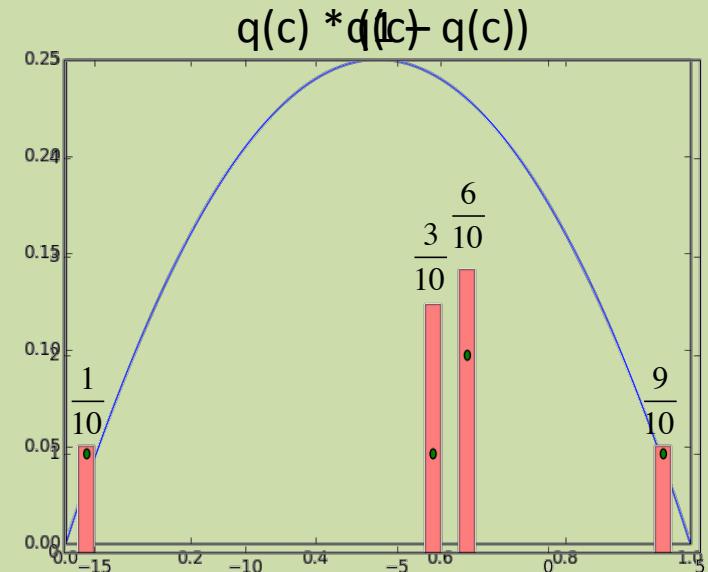
$$4 * n * \delta * q(c) * (1 - q(c))$$

free parameter
number of sample quantile of centroid

Quantile of centroid c

$$q(c) = \frac{\frac{1}{2}c.count + \sum_{d:d.mean < c.mean} d.count}{\sum_d d.count}$$

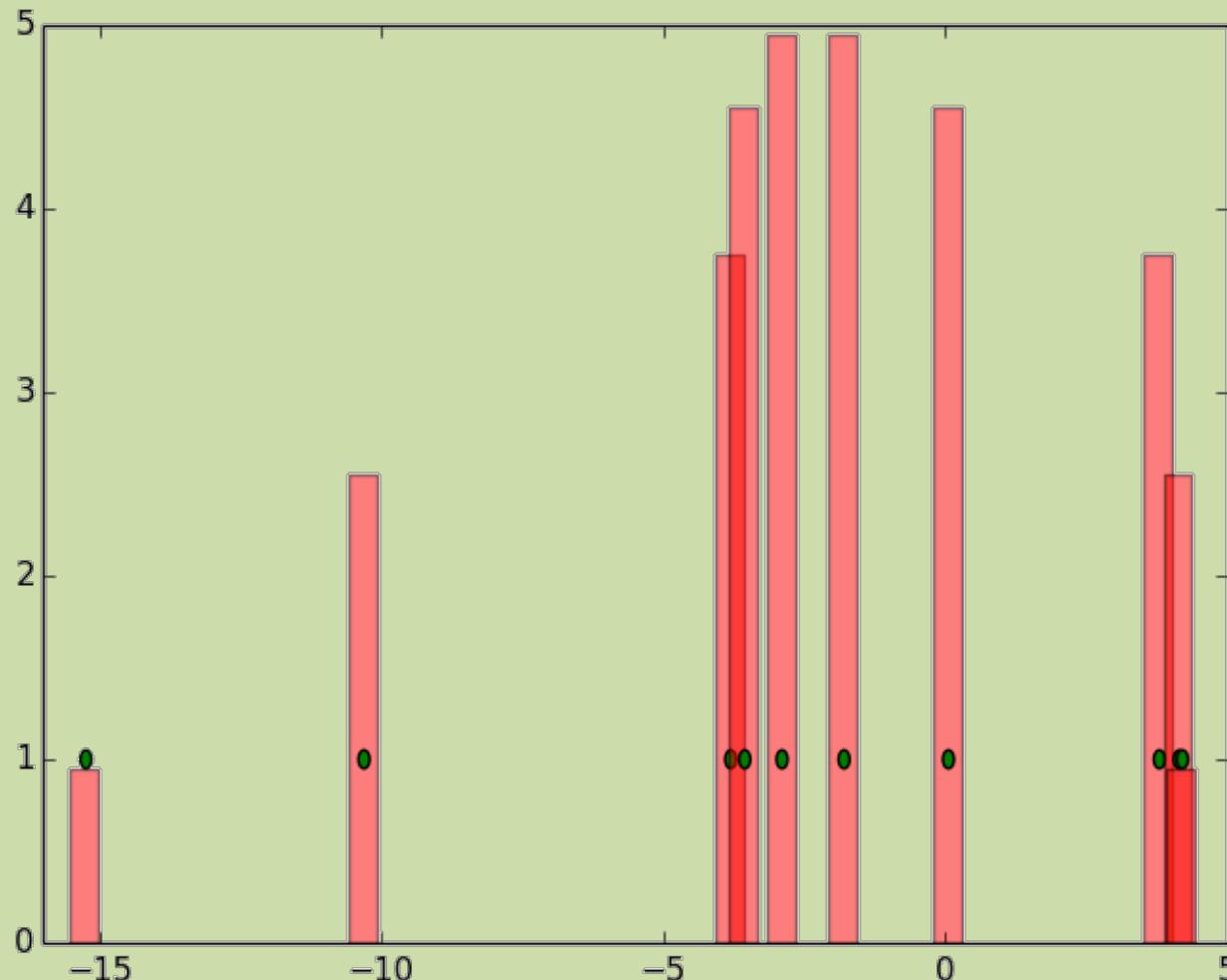
$$\Rightarrow q(c) \in (0,1)$$



t-Digests with ordered input

-15.26, -10.33, -3.83, -3.58, -2.92, -1.82, 0.03, 3.77, 4.12, 4.18

(-15.26, 1), (-10.33, 1), (-3.83, 1), (-3.58, 1), (-2.92, 1),
(-1.82, 1), (0.03, 1), (3.77, 1), (4.12, 1), (4.18, 1)



t-Digests with ordered input: shuffle!

(-15.26, 1), (-10.33, 1), (-3.83, 1), (-3.58, 1), (-2.92, 1),
(-1.82, 1), (0.03, 1), (3.77, 1), (4.12, 1), (4.18, 1)

10 centroids



(-3.58, 1), (3.77, 1), (-15.26, 1), (-1.82, 1), (-3.83, 1),
(-2.92, 1), (-10.33, 1), (4.12, 1) , (4.18, 1), (0.03, 1)

8 centroids

shuffling typically reduces number of centroids by 20% - 40% without loss of accuracy

taking a centroid list as input could cause weights $\neq 1$

Algorithm 1: Construction of a t -Digest

Input: Ordered set of weighted centroids $C = \{\}$, sequence of real-valued, weighted points $X = \{(x_1, w_1), \dots, (x_N, w_N)\}$, and accuracy tolerance δ

Output: final set $C = [c_1 \dots c_m]$ of weighted centroids

1 **for** $(x_n, w_n) \in X$:

2 $z = \min |c_i.\text{mean} - x|$;

3 $S = \{c_i : |c_i.\text{mean} - x| = z \wedge c_i.\text{count} + 1 \leq 4n\delta q(c_i)(1 - q(c_i))\}$;

4 **while** $S \neq \{\}$

5 Sample $c_j \sim \text{Uniform}(S)$;

6

7 $c_j.\text{count} \leftarrow c_j.\text{count} + 1$

8 $c_j.\text{mean} \leftarrow c_j.\text{mean} + (x_n - c_j.\text{mean})/c_j.\text{count}$;

9 $w_n \leftarrow w_n - 1$;

10

11 **if** $w_n > 0$:

12 $C \leftarrow C + (x_n, w_n)$;

13 **if** $|C| > K/\delta$:

14 $C \leftarrow \text{cluster}(\text{permute}(C))$;

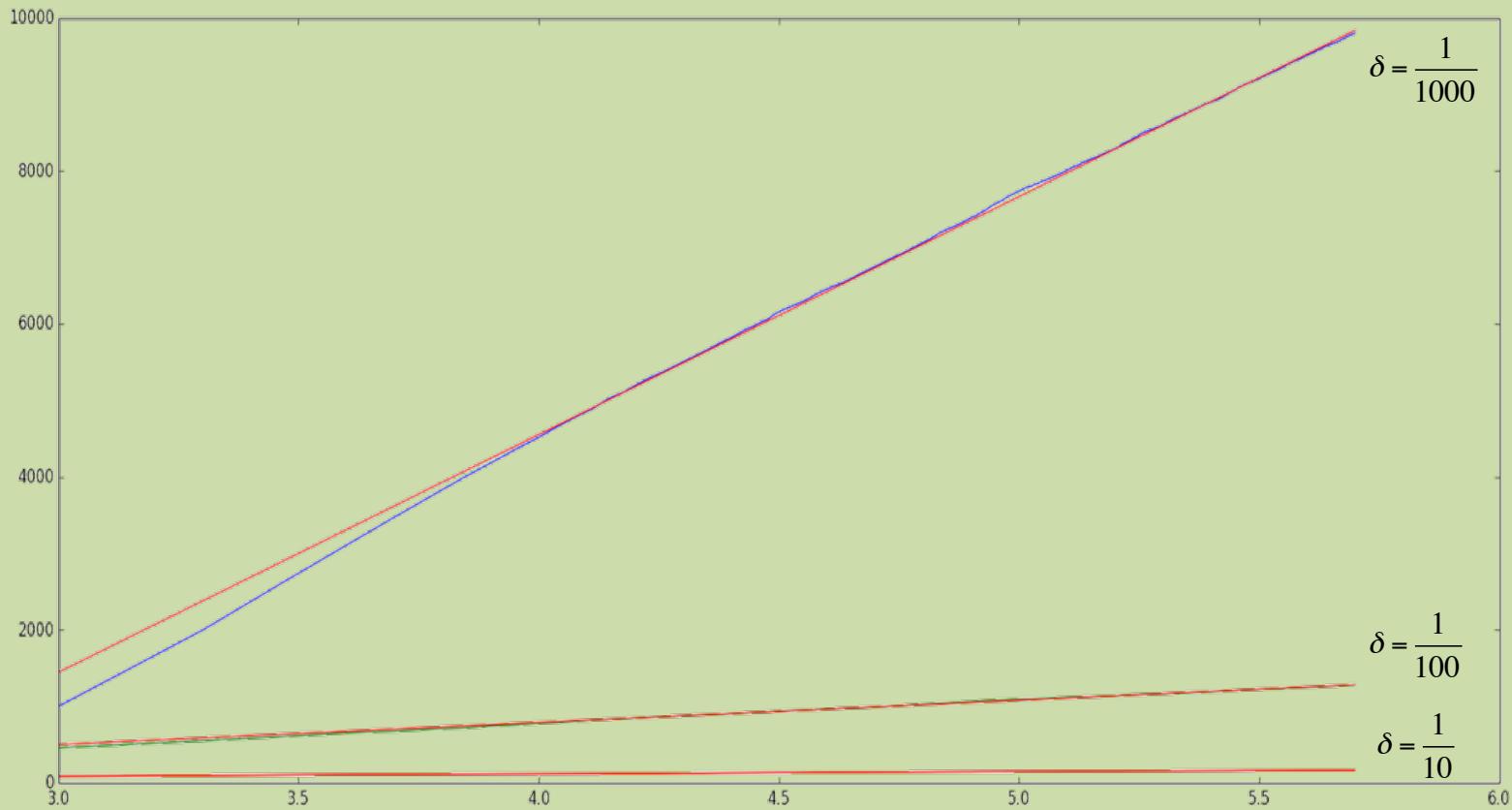
15 $C \leftarrow \text{cluster}(\text{permute}(C))$;

16 **return** C

**special case:
input weights are always 1**

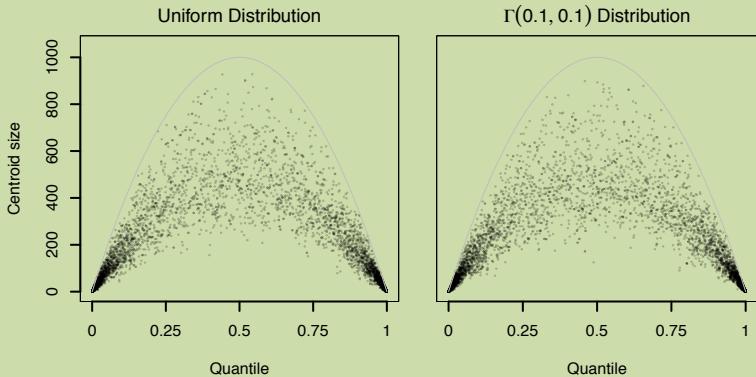
Number of centroids

- 500K samples
- x axis has exponential scale (-> straight line has logarithmic growth)

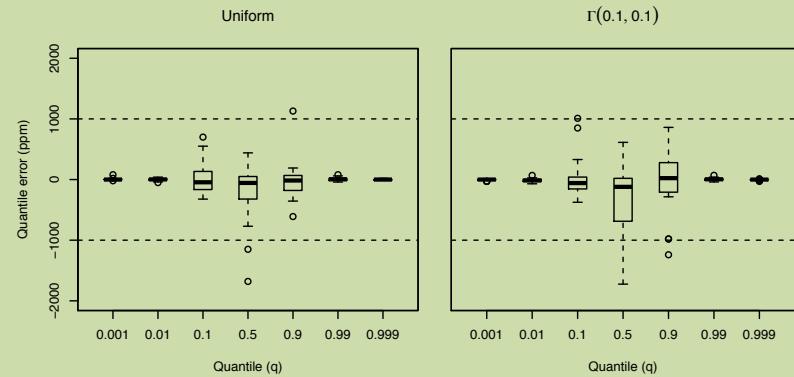


Estimation quality for quantiles

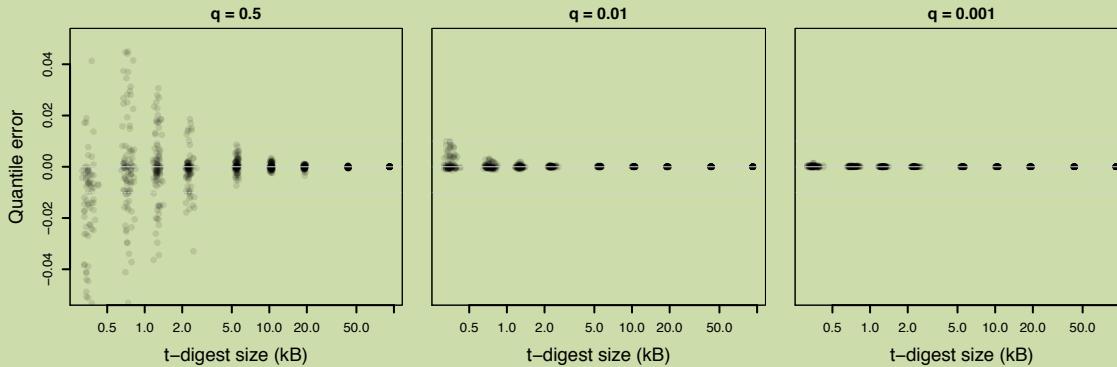
size of centroid



absolute quantile error



absolute quantile error



taken from <https://github.com/tdunning/t-digest/blob/master/docs/t-digest-paper/histo.pdf>

Ressources

Ted Dunning's GitHub rep: github.com/tdunning/t-digest

- research paper
- JAVA implementation

Integrated in

- Mahout
- ElasticSearch

Python implementation

- Trademob's working on open source package

A large, stylized word cloud composed of the words "When?", "Where?", "Who?", "What?", and "Why?" in various sizes and colors (red, green, yellow). The words are arranged in a dense, overlapping pattern against a light green background. The word "When?" is the largest and most prominent, appearing in red at the top left. Other words like "Where?", "Who?", "What?", and "Why?" are also repeated throughout the design.

Who? What? Questions & Answers