Pattern Analysis and Applications (2020) 23:225–237
https://doi.org/10.1007/s10044-019-00778-3

THEORETICAL ADVANCES

Tracking of multiple quantiles in dynamically varying data streams

Hugo Lewi Hammer[1] · Anis Yazidi[1] · Håvard Rue[2]

Received: 29 June 2018 / Accepted: 3 January 2019 / Published online: 16 January 2019

Abstract

In this paper, we consider the problem of tracking multiple quantiles of dynamically varying data stream distributions. The method is based on making incremental updates of the quantile estimates every time a new sample is received. The method is memory and computationally efficient since it only stores one value for each quantile estimate and only performs one operation per quantile estimate when a new sample is received from the data stream. The estimates are realistic in the sense that the monotone property of quantiles is satisfied in every iteration. Experiments show that the method efficiently tracks multiple quantiles and outperforms state-of-the-art methods.

Keywords Dynamic environment · Incremental algorithm · Multiple quantiles · Quantile tracking

1 Introduction

In this paper, we consider the problem of estimating quantiles when data arrive sequentially (data stream). The problem has been considered for many applications like portfolio risk measurement in the stock market [1, 12], fraud detection [28], signal processing and filtering [24], climate change monitoring [29], SLA violation monitoring [21, 22] and backbone network monitoring [7].

Real-life data streams typically have the following properties:

1. The distribution of data from the data stream changes with time. All sorts of changes may happen like a shift of the distribution, change in the expectation value or the variance or other changes of the shape of the distribution.
2. Data are received with a high intensity over a long period of time.
3. Following a data stream over time, one may expect outliers and some times extreme outliers.

In this paper, we consider the problem of maintaining running estimates of multiple quantiles for data streams with the properties described above (quantile tracking). A natural requirement of the quantile estimates is that the monotone property of quantiles is satisfied, i.e. that the estimate of the, say, 70% quantile always is above the estimate of the, say, 50% quantile.

Efficiently tracking quantiles for data streams with the properties described above (and as shown in Fig. 1) is a challenging task. The most natural is to maintain a sorted list of the data and estimate the quantiles from the sorted list. Such a quantile estimator is not viable for massive data streams as computation time and memory requirement increase with time. In addition, the quantile estimates will not adapt to the dynamic changes of the data stream. Another alternative could be to fit a time series model to the received data and compute quantiles of the forecast distribution, but such approaches are vulnerable to changes in the properties of the data stream, e.g. if the stream changes from a period with slow variations to a period with rapid variations. The method that will be presented in this paper is completely nonparametric and only relies on a single tuning parameter making it robust to changes in the properties of the data stream distribution.

Several algorithms have been proposed to deal with those challenges. Most of the proposed methods fall under the category of what can be called histogram or batch-based methods. The methods are based on efficiently maintaining a histogram estimate of the data stream distribution such that only a small storage footprint is required. See [3, 8, 10, 11, 17] for representative examples and [18] for a recent survey.

✉ Hugo Lewi Hammer
hugo.hammer@oslomet.no

[1] Department of Computer Science, OsloMet – Oslo Metropolitan University, Pilestredet 35, 0166 Oslo, Norway

[2] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

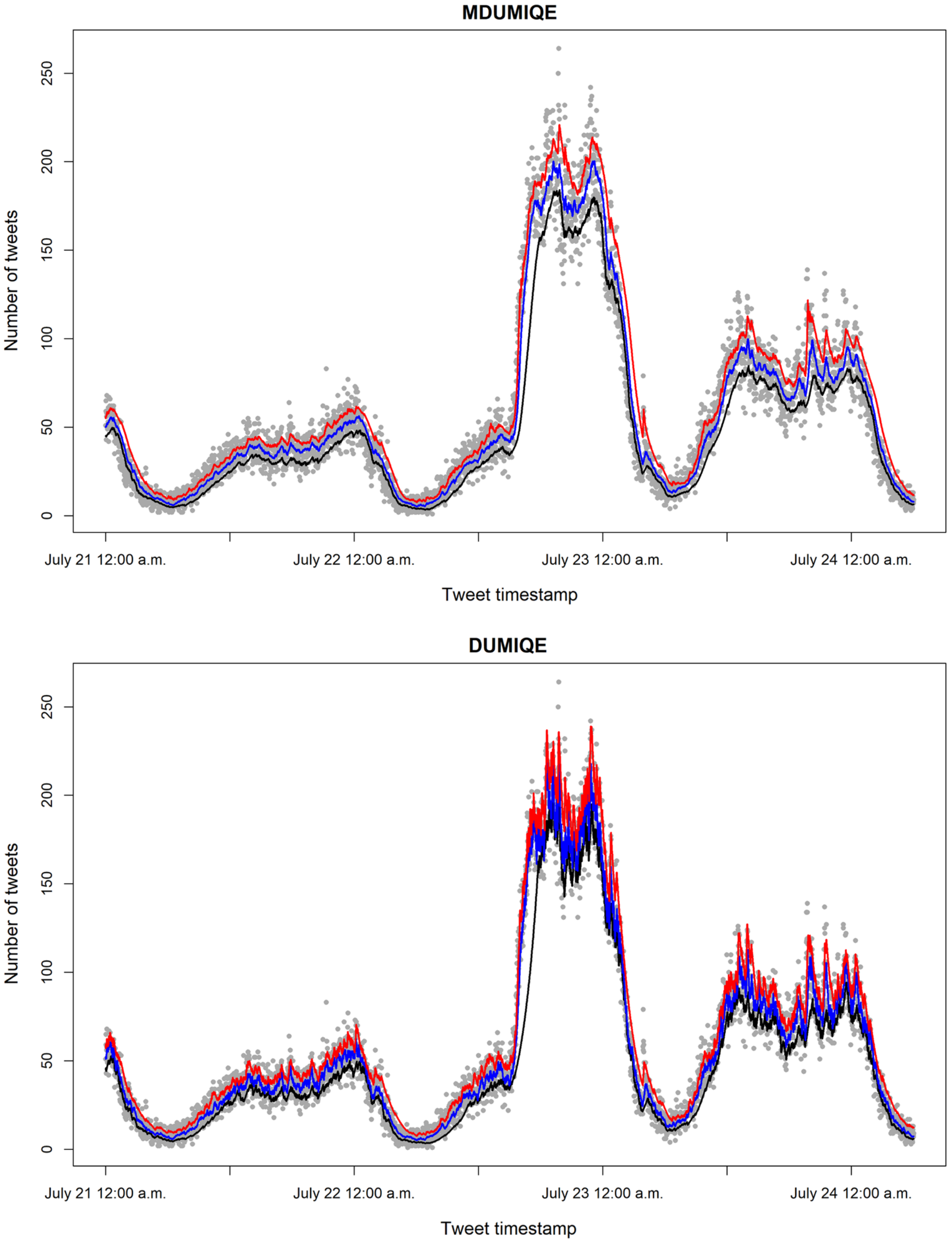

Fig. 1 The upper and lower panel shows quantile tracking for the algorithm suggested in this paper (MDUMIQE) and the DUMIQE algorithm, respectively. The grey circles show the number of tweets posted by Norwegian Twitter users every minute from 21 July 2011 to 24 July 2011. The black, blue and red curves show running estimates of the 20, 50 and 80% quantiles of the distribution of the number of tweets posted (colour figure online)

Another ally of methods is so-called *incremental* update methods. The methods are based on performing small updates of quantile estimates every time a new sample is received from the data stream. The methods only need to store one value for each quantile estimate and therefore are very memory efficient compared to histogram/batch methods. The literature on such quantile estimation methods is sparse. One of the first and prominent examples is the algorithm in [25] by Tierney which is based on the stochastic approximation theory. The method is developed for a static data stream and will not work for dynamically changing data streams. A few modifications of the Tierney method have been suggested that are able to track quantiles of dynamically varying data streams, see e.g. [4, 6]. For more recent methods, we can mention the Frugal methods [19] which run a discrete Markov chain and estimate quantiles of discrete probability distribution. Recently, Hammer and Yazidi proposed the deterministic based multiplicative incremental quantile estimator (DUMIQE) [27] which is a version of the Frugal method that works on continuous sample spaces in addition to an improved version, based on deterministic updates. Other recent methods are the DQTRE and DQTRSE algorithms by Tiwari and Pandey [26]. A nice property of the DUMIQE, DQTRE, DQTRSE and the estimator suggested in this paper is that the update size is automatically adjusted dependent on the scale/range of the data.

This makes the estimators robust to substantial changes in the data stream. The DQTRE and DQTRSE aim to achieve this by estimating the range of the data using peak and valley detectors. However, a disadvantage with these algorithms is that several tuning parameters are required to estimate the range making the algorithms challenging to tune compared to the DUMIQE and the estimator suggested in this paper. Recently in [14, 16], Hammer, Yazidi and Rue presented an incremental estimator that used the values of the received samples directly separating it from *all* incremental estimators previously presented in the literature. The estimator is in fact a generalized exponentially weighed average of previous observations received from the data stream and documents state-of-the-art performance [14, 16].

Given a dynamically changing data stream, two main problems are considered in the literature, namely to (1) dynamically update estimates of quantiles of all data received from the stream so far or (2) estimate quantiles of the current distribution of the data stream (tracking). Despite the importance of efficient tracking of statistical properties, the tracking problem (2) has been far less studied in the literature than problem (1). Incremental methods are well suited to address the tracking problem (2), while histogram and batch methods mainly have been used to address problem (1). Histogram and batch-based methods are not well suited for the tracking problem (2), and incremental methods typically are the only viable lightweight alternatives [5].

A disadvantage with the incremental methods referred to above is that they are constructed to track only *a single* quantile of the data stream. Of course, one could run such methods for several quantile probabilities, but for such methods, the quantile estimates usually will be unrealistic since the monotone property of quantiles will be violated. The problem with monotone property violation will be reduced if the incremental update size is reduced, but this is not a viable alternative for dynamically changing data streams. Using too small update steps, the incremental methods will not be able to track the dynamic changes of the data stream. In other words, a good quantile tracking algorithm must on one hand be able to efficiently track the quantiles of the data stream and at the same time satisfy the monotone property of quantiles in every iteration.

The only methods we have found in the literature that attempt to satisfy this is that of Cao et al. in [5] and by Hammer and Yazidi in [13]. The method by Cao et al. is based on first running an incremental update of each quantile estimate and secondly computing a monotonically increasing approximation of the cumulative distribution of the data stream distribution. Finally, the quantile estimates are computed from the approximate cumulative distribution. A disadvantage of the method is that the data from the data stream will be used directly making it sensitive to outliers (recall property three of real-life data streams). The method by Hammer and Yazidi in [13] is an immature version of the algorithm presented in this paper. A disadvantage of both of these methods is that we do not have any guarantee that the resulting quantile estimates converge to the true quantiles.

In this paper, we suggest a novel incremental method to track multiple quantiles that handles all the challenges with real-life data streams as described above. The method adapts efficiently to dynamically changing data streams (property 1 of real-life data streams) and only needs one operation and to store one value per quantile estimate per iteration making it extremely computational and memory efficient (property 2). The method does not use the values of the data streams directly, but only if the values are above or below the current estimates. This makes the method robust to outliers (property 3). The method is constructed such that quantile estimates satisfy the monotone property of quantiles in every iteration. A theoretical proof will be given showing that the quantile estimates converge to the true quantiles. It is hard, or maybe even impossible, to prove such convergence for the alternative methods in [5, 13].

As a first demonstration of the suggested algorithm, we look at the problem of detecting and characterizing real-world events based on tweets. A popular approach is to monitor the number of posted tweets [2], although also more advanced approaches have been explored [9, 15, 23]. Figure 1 shows an example to illustrate the use of quantile tracking on Twitter data.

The upper and lower panel shows quantile tracking for the algorithm suggested in this paper and the DUMIQE, respectively. The grey circles show the number of tweets posted by Norwegian Twitter users every minute in the time period before and after the Oslo bombing and Utøya massacre in Norway 22 July 2011. The terror attack was initiated by a bomb going off in Oslo at July 22 3:25 p.m, and as expected, we see a rapid increase in the number of posted tweets after that time. The black, blue and red curves show the tracking of the 20, 50 and 80% quantiles of the distribution of the number of tweets posted using the method that will be presented in this paper. Comparing the two methods, we observe that the suggested algorithm better represents the *simultaneous* estimates of the three quantiles in each iteration. The estimates are more smooth. The DUMIQE quantile estimates even violate the monotone property of quantiles in several iterations. This is further demonstrated in the paper, for example, in Fig. 2.

2 Estimation of multiple quantiles

Let X_n denote a stochastic variable for the possible outcomes from a data stream at time n and let x_n denote a random sample (realization) of X_n. We assume that X_n is distributed according to some distribution $f_n(x)$ that varies dynamically with time n. Further, let $Q_n(q)$ denote the quantile associated with probability q, i.e $P(X_n \leq Q_n(q)) = F_{X_n}(Q_n(q)) = q$.

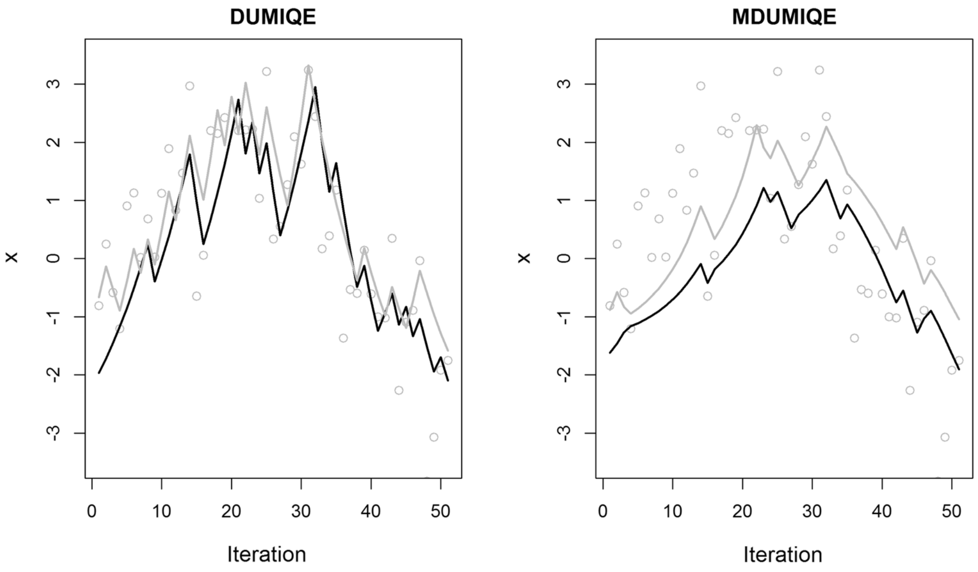

Fig. 2 Estimation processes using DUMIQE and MDUMIQE. The grey dots show samples from the data stream distribution, while the black and the grey curves show estimates of the 0.4 and the 0.6 quantiles of the data, respectively

In this paper, we focus on simultaneously estimating the quantiles for K different probabilities $q_1, q_2, \dots, q_K$ at each time step (tracking) for a data stream where the distribution of the samples varies dynamically with time. We assume an increasing order of the target quantiles to be tracked, i.e. $q_1 < q_2 < \cdots < q_K$. The straightforward approach to estimate the quantiles would be to run K online quantiles estimators in parallel and in isolation, one for each probability. Using the deterministic based multiplicative incremental quantile estimator (DUMIQE) approach from [27], the update equations become

$$\begin{aligned}
&\widehat{Q}_{n+1}(q_k) \leftarrow (1 + \lambda q_k)\widehat{Q}_n(q_k) && \text{if } \widehat{Q}_n(q_k) < x_n \\
&\widehat{Q}_{n+1}(q_k) \leftarrow (1 - \lambda(1 - q_k))\widehat{Q}_n(q_k) && \text{if } \widehat{Q}_n(q_k) \geq x_n
\end{aligned} \tag{1}$$

for $k = 1, 2, \dots, K$.

The scheme is presented in detail in Algorithm 1. We see that the algorithm requires to only store *a single* value for each quantile estimate, namely the estimate itself. Further, we see that the algorithm is computationally efficient requiring only *a single* update per quantile estimate per iteration. Finally, we see that the method is robust to outliers since the observations x_n are not used directly, but only if it is above or below the current quantile estimate.

```
Algorithm 1 The DUMIQE algorithm.
Input:
$x_1, x_2, x_3, \dots$ // Data stream
$\lambda$
$q_k,\ k = 1, 2, \dots, K$
$\widehat{Q}_0(q_k),\ k = 1, 2, \dots, K$
Method:
for $n \in 1, 2, \dots$ do
    for $k \in 1, 2, \dots, K$ do
        if $\widehat{Q}_n(q_k) < x_n$ then
            $\widehat{Q}_{n+1}(q_k) \leftarrow (1 + \lambda q_k)\widehat{Q}_n(q_k)$
        else
            $\widehat{Q}_{n+1}(q_k) \leftarrow (1 - \lambda(1 - q_k))\widehat{Q}_n(q_k)$
        end if
    end for
end for
```

We assume for now that $\widehat{Q}_n(q_k) > 0 \ \forall\ k, n$. Generalization such that $\widehat{Q}_n(q_k)$ can take any positive or negative value will be explained in remark 1 below. Unfortunately, using (1) results in unrealistic estimates as the monotone property of quantiles, as given by the constraint $\widehat{Q}_n(q_1) \leq \widehat{Q}_n(q_2) \leq \cdots \leq \widehat{Q}_n(q_K)$, is most likely violated in some iterations. For the DUMIQE, this can be explained as follows (see [5] for an example of another method). Assume at time n that the monotone property is satisfied and that the sample x_n admits a value between $\widehat{Q}_n(q_k)$ and $\widehat{Q}_n(q_{k+1})$, i.e

$$\widehat{Q}_n(q_1) \leq \cdots \leq \widehat{Q}_n(q_k) < x_n < \widehat{Q}_n(q_{k+1}) \leq \cdots \leq \widehat{Q}_n(q_K) \tag{2}$$

Then according to (1), the estimates are updated as follows

$$\begin{aligned}
&\widehat{Q}_{n+1}(q_j) \leftarrow (1 + \lambda q_j)\widehat{Q}_n(q_j) && \text{for } j = 1, 2, \dots, k \\
&\widehat{Q}_{n+1}(q_j) \leftarrow (1 - \lambda(1 - q_j))\widehat{Q}_n(q_j) && \text{for } j = k+1, \dots, K
\end{aligned} \tag{3}$$

which means that the estimates are increased for the quantiles with an estimate below x_n and decreased for the estimates above x_n. Consequently, the monotone property may get violated. In the next section, we will present a novel update scheme that satisfies the monotone property of quantiles while converging both theoretically and experimentally to the true quantiles.

2.1 Multiple quantile DUMIQE

When updating $\widehat{Q}_n(q_k)$, we ensure that the value of λ is such that $\widehat{Q}_n(q_k)$ never cross the "neighbours" $\widehat{Q}_n(q_{k-1})$ and $\widehat{Q}_n(q_{k+1})$. Assume at time n that the monotone property is satisfied and that the sample x_n gets a value between $\widehat{Q}_n(q_k)$ and $\widehat{Q}_n(q_{k+1})$ as given by (2). We now use a λ (denoted $\widetilde{\lambda}_k$ below) such that the distance between $\widehat{Q}_{n+1}(q_k)$ and

$\widehat{Q}_{n+1}(q_{k+1})$ is equal to some *portion*, α, of the distance from the previous iteration, i.e.

$$\widehat{Q}_{n+1}(q_{k+1}) - \widehat{Q}_{n+1}(q_k) = \alpha\Big(\widehat{Q}_n(q_{k+1}) - \widehat{Q}_n(q_k)\Big)$$

$$\Big(1 - \widetilde{\lambda}_k(1 - q_{k+1})\Big)\widehat{Q}_n(q_{k+1}) - \Big(1 + \widetilde{\lambda}_k q_k\Big)\widehat{Q}_n(q_k) \qquad (4)$$
$$= \alpha\Big(\widehat{Q}_n(q_{k+1}) - \widehat{Q}_n(q_k)\Big)$$

By solving (4) with respect to $\widetilde{\lambda}_k$, we obtain

$$\widetilde{\lambda}_k = (1 - \alpha)\frac{\widehat{Q}_n(q_{k+1}) - \widehat{Q}_n(q_k)}{(1 - q_{k+1})\widehat{Q}_n(q_{k+1}) + q_k\widehat{Q}_n(q_k)} \qquad (5)$$

To avoid crossing, we must ensure that $\widehat{Q}_n(q_k)$ stays above the estimate below, $\widehat{Q}_n(q_{k-1})$ as well. Thus, a sufficient criterion to guarantee that $\widehat{Q}_n(q_k)$ stays between $\widehat{Q}_n(q_{k-1})$ and $\widehat{Q}_n(q_{k+1})$ is to use the minimum of $\widetilde{\lambda}_k$ computed from $\widehat{Q}_n(q_k)$ and $\widehat{Q}_n(q_{k+1})$ and computed from $\widehat{Q}_n(q_k)$ and $\widehat{Q}_n(q_{k-1})$. This gives the following

$$\widetilde{\lambda}_k = (1 - \alpha)\min\left\{\frac{\widehat{Q}_n(q_k) - \widehat{Q}_n(q_{k-1})}{(1 - q_k)\widehat{Q}_n(q_k) + q_{k-1}\widehat{Q}_n(q_{k-1})}, \frac{\widehat{Q}_n(q_{k+1}) - \widehat{Q}_n(q_k)}{(1 - q_{k+1})\widehat{Q}_n(q_{k+1}) + q_k\widehat{Q}_n(q_k)}\right\} \qquad (6)$$
$$= (1 - \alpha)H\Big(\widehat{Q}_n(q_k);\widehat{Q}_n(q_{k-1}), \widehat{Q}_n(q_{k+1})\Big)$$

By using $\lambda = \widetilde{\lambda}_k$ from (6) in (1) when updating the estimates $\widehat{Q}_n(q_k), k = 1, 2, \dots, K$, the monotone property will be satisfied for all the quantile estimates. Of course, the lowest quantile estimate $\widehat{Q}_n(q_1)$ only needs to satisfy the monotone property against $\widehat{Q}_n(q_2)$, and therefore, H becomes

$$H\Big(\widehat{Q}_n(q_1), \widehat{Q}_n(q_2)\Big) = \frac{\widehat{Q}_n(q_2) - \widehat{Q}_n(q_1)}{(1 - q_2)\widehat{Q}_n(q_2) + q_1\widehat{Q}_n(q_1)} \qquad (7)$$

and similarly for the highest quantile estimate

$$H\Big(\widehat{Q}_n(q_{K-1}), \widehat{Q}_n(q_K)\Big) = \frac{\widehat{Q}_n(q_K) - \widehat{Q}_n(q_{K-1})}{(1 - q_K)\widehat{Q}_n(q_K) + q_{K-1}\widehat{Q}_n(q_{K-1})} \qquad (8)$$

Substituting $\widetilde{\lambda}_k$ in (6) for λ in (1) and defining $\beta = 1 - \alpha$, we obtain the following update rules

$$\begin{aligned}
&\widehat{Q}_{n+1}(q_k) \leftarrow \Big(1 + \beta H\Big(\widehat{Q}_n(q_k);\widehat{Q}_n(q_{k-1}), \widehat{Q}_n(q_{k+1})\Big)q_k\Big)\widehat{Q}_n(q_k) && \text{if } \widehat{Q}_n(q_k) < x_n\\
&\widehat{Q}_{n+1}(q_k) \leftarrow \Big(1 - \beta H\Big(\widehat{Q}_n(q_k);\widehat{Q}_n(q_{k-1}), \widehat{Q}_n(q_{k+1})\Big)(1 - q_k)\Big)\widehat{Q}_n(q_k) && \text{if } \widehat{Q}_n(q_k) \ge x_n
\end{aligned} \qquad (9)$$

for $k = 2, \dots, K - 1$. For $k = 1$ and $k = K$, it results in

$$\begin{aligned}
&\widehat{Q}_{n+1}(q_1) \leftarrow \Big(1 + \beta H\Big(\widehat{Q}_n(q_1), \widehat{Q}_n(q_2)\Big)q_1\Big)\widehat{Q}_n(q_1) && \text{if } \widehat{Q}_n(q_1) < x_n\\
&\widehat{Q}_{n+1}(q_1) \leftarrow \Big(1 - \beta H\Big(\widehat{Q}_n(q_1), \widehat{Q}_n(q_2)\Big)(1 - q_1)\Big)\widehat{Q}_n(q_1) && \text{if } \widehat{Q}_n(q_1) \ge x_n
\end{aligned} \qquad (10)$$

and

$$\begin{aligned}
&\widehat{Q}_{n+1}(q_K) \leftarrow \Big(1 + \beta H\Big(\widehat{Q}_n(q_{K-1}), \widehat{Q}_n(q_K)\Big)q_K\Big)\widehat{Q}_n(q_K) && \text{if } \widehat{Q}_n(q_K) < x_n\\
&\widehat{Q}_{n+1}(q_K) \leftarrow \Big(1 - \beta H\Big(\widehat{Q}_n(q_{K-1}), \widehat{Q}_n(q_K)\Big)(1 - q_K)\Big)\widehat{Q}_n(q_K) && \text{if } \widehat{Q}_n(q_K) \ge x_n
\end{aligned} \qquad (11)$$

The parameter $\beta \in [0, 1)$ controls the size of the update when a new sample arrives, and the H-functions ensure that the monotone property will be satisfied in every iteration. Please note that since the $H-$ functions depend of k, the increment lengths vary with k.

In the rest of the paper, we will refer to the method in Algorithm 2 as MDUMIQE which is an abbreviation for multiple DUMIQE. Please see Algorithm 2 for a detailed representation of the algorithm. The structure of the algorithm is quite simple and intuitive. First in lines 3–9, the update size λ is computed to ensure no monotone violations. In lines 10 to 14, the quantiles estimates are updated using the ordinary DUMIQE with the λ computed in lines 3–9. Similar to the original DUMIQE, the MDUMIQE requires to only store *a single* value for each quantile estimate, namely the estimate itself. Further, only two operations are necessary per quantile estimate in every iteration, namely to adjust the step size and update the quantile estimate. In other words, the MDUMIQE algorithm is extremely memory and computationally efficient. Finally, we observe that, similar to the DUMIQE, MDUMIQE does not use the observations x_n directly and thus is very robust to outliers. Please see [15] for a further demonstration of the robustness of the DUMIQE and MDUMIQE.

Algorithm 2 The MDUMIQE Algorithm.

```
Input:
$x_1, x_2, x_3, \dots$ // Data stream
$\beta$
$q_k,\ k = 1, 2, \dots, K$
$\widehat{Q}_0(q_k),\ k = 1, 2, \dots, K$
Method:
for $n \in 1, 2, \dots$ do
    for $k \in 1, 2, \dots, K$ do
        if $k == 1$ then
            $\lambda \leftarrow \beta \frac{\widehat{Q}_n(q_2) - \widehat{Q}_n(q_1)}{(1-q_2)\widehat{Q}_n(q_2) + q_1\widehat{Q}_n(q_1)}$
        else if $k == K$ then
            $\lambda \leftarrow \beta \frac{\widehat{Q}_n(q_K) - \widehat{Q}_n(q_{K-1})}{(1-q_K)\widehat{Q}_n(q_K) + q_{K-1}\widehat{Q}_n(q_{K-1})}$
        else
            $\lambda \leftarrow \beta \min\left\{ \frac{\widehat{Q}_n(q_k) - \widehat{Q}_n(q_{k-1})}{(1-q_k)\widehat{Q}_n(q_k) + q_{k-1}\widehat{Q}_n(q_{k-1})}, \frac{\widehat{Q}_n(q_{k+1}) - \widehat{Q}_n(q_k)}{(1-q_{k+1})\widehat{Q}_n(q_{k+1}) + q_k\widehat{Q}_n(q_k)} \right\}$
        end if
        if $\widehat{Q}_n(q_k) < x_n$ then
            $\widehat{Q}_{n+1}(q_k) \leftarrow (1 + \lambda q_k)\widehat{Q}_n(q_k)$
        else
            $\widehat{Q}_{n+1}(q_k) \leftarrow (1 - \lambda(1 - q_k))\widehat{Q}_n(q_k)$
        end if
    end for
end for
```

Now we will present a theorem that catalogues the properties of the estimators $\widehat{Q}_n(q_k),\ k = 1, 2, \dots, K$ given in (9)–(11) for a stationary data stream, i.e. $X_n = X \sim f(x),\ n = 1, 2, \dots$. We assume that all the estimators $\widehat{Q}_n(q_k) > 0,\ k = 1, 2, \dots, K$ and the true quantiles $Q(q_k) > 0,\ k = 1, 2, \dots, K$. A sufficient condition to obtain $Q(q_k) > 0,\ k = 1, 2, \dots, K$ is that the random variable X only takes positive values.

Theorem 1 *Let* $Q(q_k) = {F_X}^{-1}(q_k),\ k = 1, 2, \dots, K$ *be the true quantiles to be estimated and suppose that* $Q(q_k) > 0,\ k = 1, 2, \dots, K$. *In addition, we suppose that* $\widehat{Q}_1(q_k) > 0,\ k = 1, 2, \dots, K$. *Applying the updating rules* (9)– (11), *we obtain*

$$\lim_{n\beta \to \infty, \beta \to 0} \widehat{Q}_n(q_k) = Q(q_k), \quad k = 1, 2, \dots, K$$

The proof of the theorem can be found in "Appendix 1". Although the quantile estimators $\widehat{Q}_n(q_k),\ k = 1, 2, \dots, K$ given in (9) to (11) are mainly designed to estimate quantiles for dynamic environments, it is an important requirement of the estimators that they converge to the true quantiles for static data streams as given by Theorem 1.

We end this section with a few remarks.

Remark 1 A potential challenge with multiplicative update schemes, as given by (1) and by (9)–(11), is that if we start with a quantile estimate above zero, $\widehat{Q}_0(q_k) > 0$, the estimates will stay above zero. Similarly, if we start with a quantile estimate below zero, it will stay below zero. A simple solution is to estimate the quantiles of a transformation of the data $h(X_n)$ where $h(\cdot)$ is a monotonically increasing function and $h(x) > 0\,\forall x$. A natural alternative is $h(x) = \exp(x)$.

Remark 2 We see that the updating rules (9)–(11) only update based on $\widehat{Q}_n(q_k) < x_n$ or $\widehat{Q}_n(q_k) > x_n$ and not the value of x_n. This means that the algorithm is very robust against outliers, which is important for real-life data streams.

Remark 3 The strategy of adjusting the value λ in order to avoid the monotone violation of quantiles, as described in Sect. 2.1, can also be used for the additive alternative to DUMIQE in (1) given by

$$\begin{array}{ll} \widehat{Q}_{n+1}(q_k) \leftarrow \widehat{Q}_n(q_k) + \lambda q_k & \text{if } \widehat{Q}_n(q_k) < x_n \\ \widehat{Q}_{n+1}(q_k) \leftarrow \widehat{Q}_n(q_k) - \lambda(1 - q_k) & \text{if } \widehat{Q}_n(q_k) \geq x_n \end{array} \tag{12}$$

In our experiments, MDUMIQE outperformed this additive alternative and the experimental results thus limit to MDUMIQE.

3 Experiments

In this section, we evaluate the performance of the estimators presented in this paper. We compare against the only alternative multiple quantile tracking algorithm we are aware of, namely the method of Cao et al. [5]. It would be interesting to evaluate the performance of different methods for real-life data, but this is challenging to do in a systematic way for dynamical data streams as the ground truth generally is missing. Before proceeding to systematic experiments based on synthetic data, we just recall Fig. 1 showing that MDUMIQE can be used to efficiently track quantiles of challenging real-life data streams.

We look at two different cases where we assume that the data are outcomes from a normal distribution or from a χ^2 distribution. For the normal distribution case, we assume that the expectation of the distribution varies with time

$$\mu_n = a \sin\left(\frac{2\pi}{T} n\right), \quad n = 1, 2, 3, \dots$$

which is the sinus function with period T. Further, we assume that the standard deviation of the distribution does not vary with time and is equal to one. For the χ^2 distribution case, we assume that the number of degrees of freedom varies with time as follows:

$$\nu_n = a \sin\left(\frac{2\pi}{T} n\right) + b, \quad n = 1, 2, 3, \dots$$

where $b > a$ such that $\nu_n > 0$ for all n. In the experiments below, we used $a = 2$ and $b = 6$.

Figure 2 shows a small section of the estimation processes using DUMIQE and MDUMIQUE. The grey dots show the samples from the data stream and are the same in both panels. The data are generated from the normal distribution above with $T = 75$. The grey and the black curves show estimates of the 0.4 and the 0.6 quantiles of the data, respectively. We see that using DUMIQE, the monotone property is violated in several iterations, while MDUMIQE satisfies the monotone property and at the same time is able to track the quantiles efficiently.

We now turn to performing a thorough analysis of how well the proposed method in Sect. 2 estimates the quantiles of data streams. We considered two different periods, namely $T = 800$ (rapid variation) and $T = 8000$ (slow variation), i.e. in total four different data streams. In addition, for each of the four data streams, we estimated quantiles around the median and in the tail of the distribution. We estimated three or nine quantiles representing cases where the distance between the quantiles is either large or small, respectively. Obviously, if the quantiles are close to each other, the monotone property will be violated more frequently, making the estimation problem more difficult. In more detail, for the different cases we estimated the following quantiles:

- For the normal distribution and the quantiles around the median, we estimated the quantiles related to the following probabilities $q_k = \Phi(-0.8 + 0.2(k-1)),\ \ k = 1, 2, \ldots, 9$. For the case with three quantiles, we only used $k = 1, 5$ and 9. $\Phi(\cdot)$ refers to the cumulative distribution function of the standard normal distribution. Recall that in dynamically changing data streams, as in these experiments, the value of a quantile related to a specific probability varies with time.
- For the normal distribution and the quantiles in the tail of the distribution, we use $q_k = \Phi(0.8 + 0.2(k-1)),\ \ k = 1, 2, \ldots, 9$. For the case with three quantiles, we only used $k = 1, 5$ and 9.
- For the χ^2 distribution and the quantiles around the median, we estimated the quantiles related to the following probabilities $q_k = F(4.2 + 0.3(k-1); \nu = 6),\ \ k = 1, 2, \ldots, 9$ where $F(\cdot; \nu)$ refers to the cumulative distribution function of the χ^2 distribution with ν degrees of freedom. For the case with three quantiles, we only used $k = 1, 5$ and 9.
- Finally, for the χ^2 distribution and the quantiles in the tail of the distribution, we estimated the quantiles related to the following probabilities $q_k = F(12 + 0.4(k-1); \nu = 6),\ \ k = 1, 2, \ldots, 9$. For the case with three quantiles, we only used $k = 1, 5$ and 9.

The probabilities related to quantiles in the median and in tail of the distribution are centred around the probabilities 0.5 and 0.95, respectively. When estimating nine quantiles, the choices above resulted in a monotone property violation at about every third iteration using a typical value of $\lambda = 0.05$ in (1). Similarly when estimating three quantiles, we got a monotone property violation at about every eleventh iteration.

To measure estimation error, we use the average of the root-mean-square error (RMSE) for each quantile. In order to get a good overview of the performance of the algorithms, we measure the estimation error for a large set of different values of the tuning parameters λ and β.

The results for the normal and χ^2 distribution cases when estimating three quantiles are shown in Figs. 3 and 4, respectively. We proceed to discussing the normal distribution cases. For all the four cases, we observe that MDUMIQE outperforms DUMIQE for the optimal values of β and λ which are chosen (results in the smallest RMSE). We also see that estimation performance of the MDUMIQE is less sensitive to the choice of β than DUMIQE on the choice of λ. This is a crucial remark since for real-life applications we do not know the optimal values of β and λ that yield the best results. Hence, not only are we able to satisfy the monotone property of quantiles, we also improve estimation precision compared to DUMIQE. For the χ^2 distribution cases, we see that MDUMIQE and DUMIQE perform about equally well except that DUMIQE performs better when $T = 800$ and when estimating quantiles in the tail of the distribution. The reason will be explained below.

The results for the normal and χ^2 distribution cases when estimating nine quantiles are shown in Figs. 5 and 6, respectively. We start by discussing the normal distribution cases. We see that for $T = 800$, the DUMIQE performs better than the MDUMIQE, especially when estimating in the tail of the distribution. On the other hand, for $T = 8000$ and estimating the median, the MDUMIQE outperforms DUMIQE. The explanation for why DUMIQE performs better when $T = 800$ is that for such a rapidly changing data stream, large updates of the quantile estimates must be used to track the true quantiles. MDUMIQE is required to satisfy the monotone property which sets a limitation on how far MDUMIQE can update the estimates in each iteration. More specifically, for MDUMIQE, the estimate $\widehat{Q}_n(q_k)$ must always be between $\widehat{Q}_n(q_{k-1})$ and $\widehat{Q}_n(q_{k+1})$, recall (6). Whenever the difference between $\widehat{Q}_n(q_{k-1})$ and $\widehat{Q}_n(q_{k+1})$ is small, we can only do small updates of $\widehat{Q}_n(q_k)$. For the χ^2 distribution, the two methods perform about equally well for $T = 8000$ and estimating the median, and for the other cases, DUMIQE performs better than MDUMIQE. An appealing property of the MDUMIQE approach (in addition to satisfying the monotone property) is that the estimation performance is less sensitive to the choice of β than DUMIQE is on the choice of λ. Using $\beta = 0.5$, we get satisfactory results in all the cases. Such a “universal” λ does not exist for DUMIQE.

For comparison purposes, we tested the multiple quantile estimation method in [5] as well for the estimation tasks described above. This is the only viable method we have found in the literature for estimating multiple quantiles in

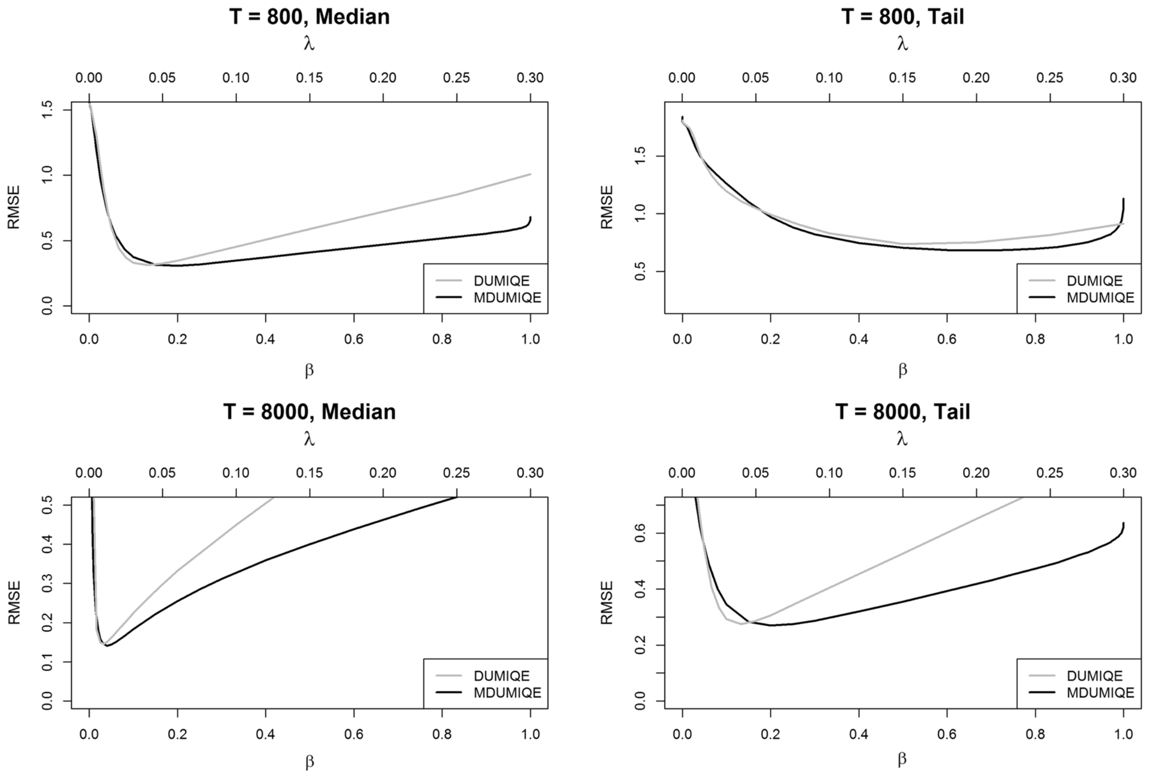

Fig. 3 Estimation error for data from the normal distribution when estimating three quantiles

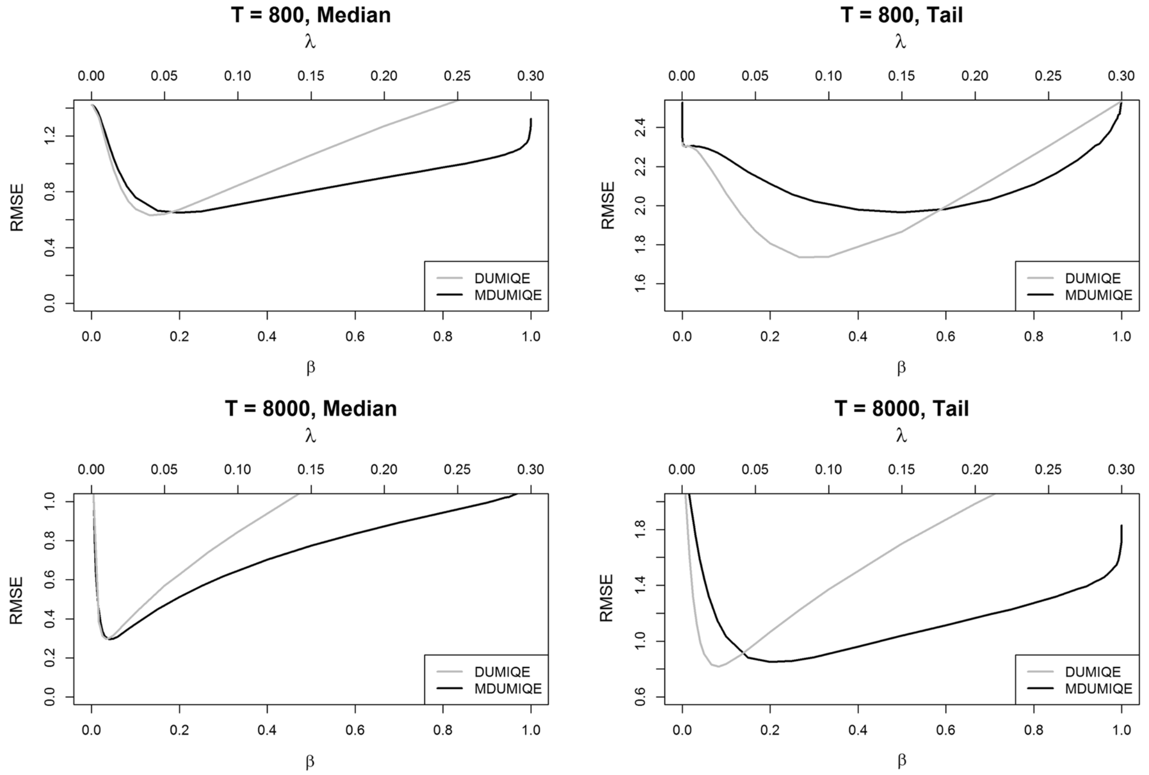

Fig. 4 Estimation error for data from the χ^2 distribution when estimating three quantiles

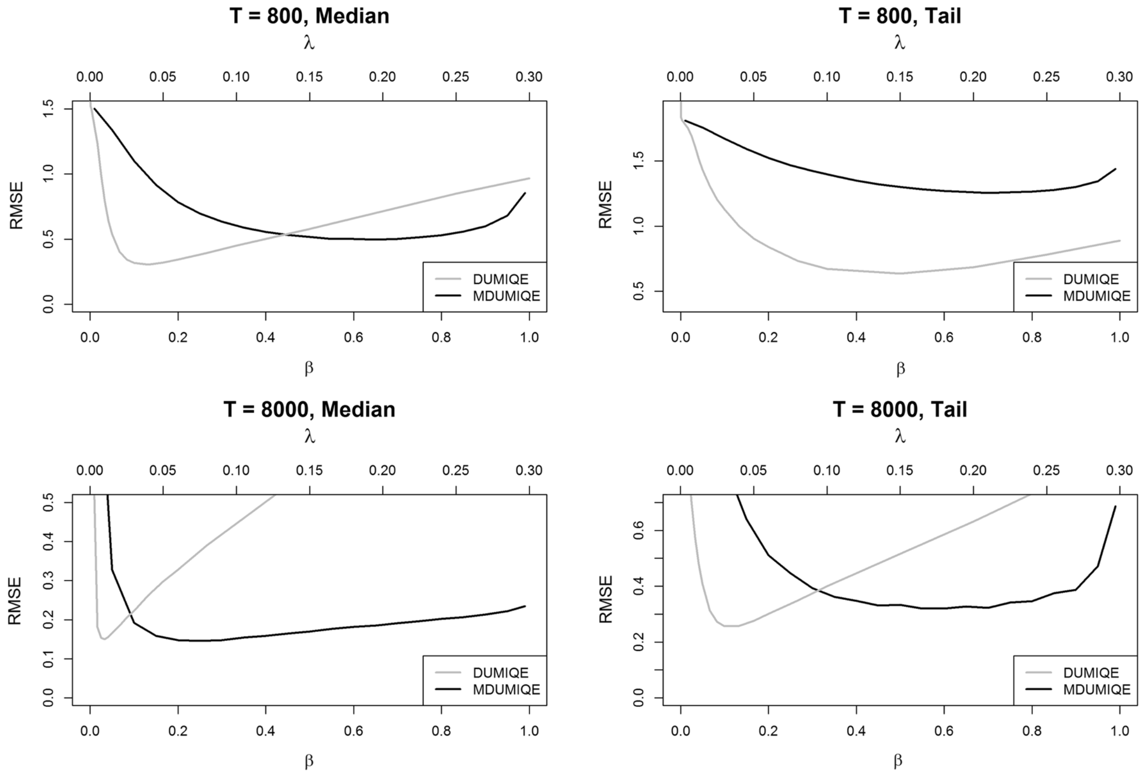

Fig. 5 Estimation error for data from the normal distribution when estimating nine quantiles

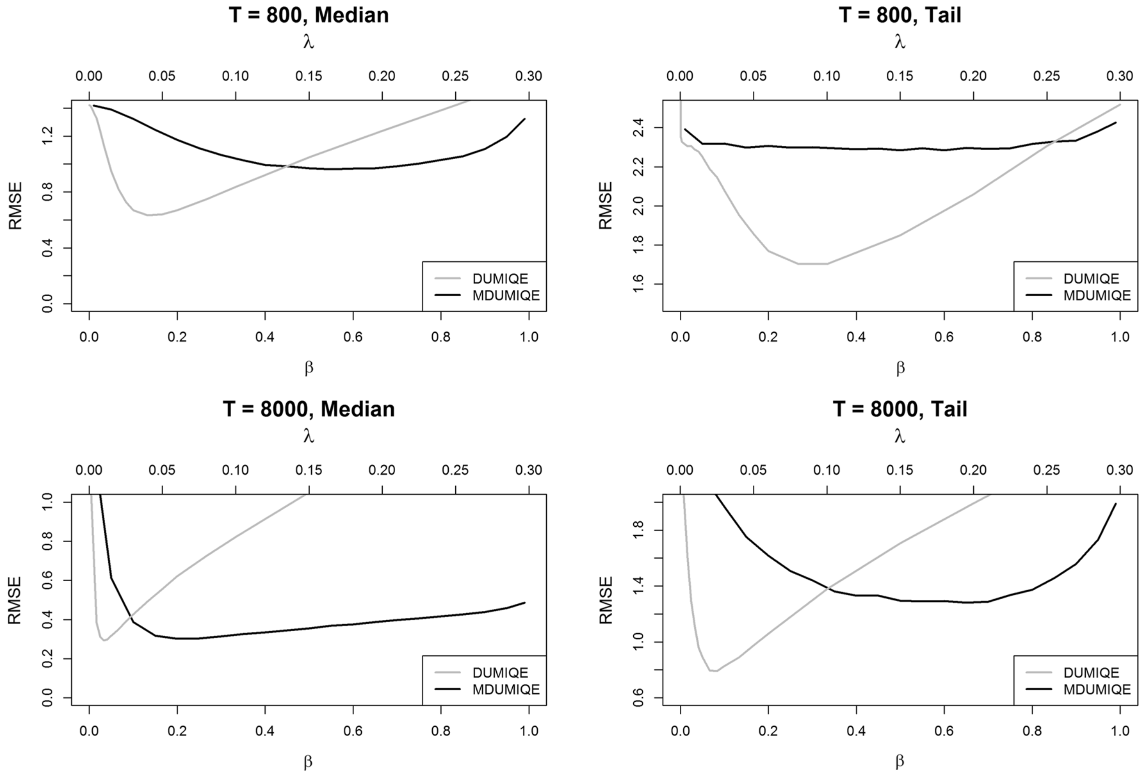

Fig. 6 Estimation error for data from the χ^2 distribution when estimating nine quantiles

dynamically changing data streams. The method has two tuning parameters, a weight parameter similar to λ and β for the methods in this paper, and a parameter that controls the width of intervals to estimate the distribution of the data stream around a quantile. To achieve the best possible results, we ran the method for a large set of values for the two parameters. The best estimation results (smallest RMSE) are shown in Tables 1 and 2 for the cases with three and nine quantiles, respectively. Comparing these results with the results in Figs. 3, 4, 5 and 6, we see that MDUMIQE clearly outperforms Cao et al. [5].

4 Closing remarks

In this paper, we present a novel algorithm for keeping online estimates of multiple quantiles in a dynamically changing data stream (tracking). The algorithm is an extension of the efficient DUMIQE algorithm from [27], developed to avoid monotone violations of the quantile estimates. A theoretical proof is given that it ensures the convergence of each quantile estimate to its true quantile.

The experimental results in Sect. 3 show that the suggested algorithm performs very well. For most of the experiments, the method performs equally well or better than DUMIQE that does not satisfy the monotone property of quantiles. The suggested algorithm is well suited to track multiple quantiles for real-life data streams as it adapts efficiently to dynamic changes in the data streams, is very computationally and memory efficient and is robust to outliers.

Another advantage of the suggested algorithm is that the estimation performance is less sensitive to the choice of the tuning parameter compared to DUMIQE. Choosing a $\beta = 0.5$ performed well in all the experiments. This is a crucial property for real-life data streams since the distribution of data streams may vary slowly in some time periods and more rapidly in others (see Fig. 1 for an example). Using the DUMIQE, one must choose a tuning parameter that performs well either where the data stream varies slowly or rapidly. Since the performance of the algorithm suggested in this paper is less sensitive to the choice of the tuning parameter, it will perform well both when the data stream varies slowly and rapidly. In Fig. 1, we see that the algorithm performs well both when the data stream varies slowly and rapidly. In addition, we saw that the algorithm outperformed the state-of-the-art method of Cao et al. [5] with a clear margin.

The suggested algorithm experiences some reduction in performance for rapidly changing data streams. For such data streams, large updates of the quantile estimates are necessary to track the true quantiles efficiently and the requirement of satisfying the monotone property of quantiles sets a limit on how large increments that are possible, and thus reduces the performance of the algorithm. An interesting challenge for future research is to develop an incremental quantile estimation algorithm that performs better in rapidly changing dynamically changing data streams when many quantiles need to be estimated.

Appendix: Proof of Theorem 1

We will first present a theorem due to Norman [20] that will be used to prove Theorem 1. Norman [20] studied distance "diminishing models". The convergence of $\widehat{Q}_n(q_k)$ to $Q(q_k)$ is a consequence of this theorem.

Theorem 2 *Let $x(t)$ be a stationary Markov process dependent on a constant parameter $\theta \in [0, 1]$. Each $x(t) \in I$, where I is a subset of the real line. Let $\delta x(t) = x(t+1) - x(t)$. The following are assumed to hold:*

1. *I is compact*
2. $E[\delta x(t)|x(t) = y] = \theta w(y) + O(\theta^2)$
3. $Var[\delta x(t)|x(t) = y] = \theta^2 s(y) + O(\theta^2)$
4. $E[\delta x(t)^3|x(t) = y] = O(\theta^3)$ *where* $sup_{y \in I} \frac{O(\theta^k)}{\theta^k} < \infty$ *for* $k = 2, 3$ *and* $sup_{y \in I} \frac{o(\theta^2)}{\theta^2} \to 0$ as $\theta \to 0$
5. *$w(y)$ has a Lipschitz derivative in I*

Table 1 RMSE estimation error using the method in Cao et al. [5] when estimating three quantiles

	$T = 800$, median	$T = 800$, tail	$T = 8000$, median	$T = 8000$, tail
Normal distribution	0.835	1.00	0.223	0.570
χ^2 distribution	1.512	3.93	1.00	3.75

Table 2 RMSE estimation error using the method in Cao et al. [5] when estimating nine quantiles

	$T = 800$, median	$T = 800$, tail	$T = 8000$, median	$T = 8000$, tail
Normal distribution	0.312	0.630	0.259	0.370
χ^2 distribution	0.79	2.40	0.445	1.611

6. *$s(y)$ is Lipschitz I.*

If Assumptions 1 to 6 above hold, $w(y)$ has a unique root y^* in I and $\left.\frac{dw}{dy}\right|_{y=y^*} \leq 0$ then

1. *$var[\delta x(t)|x(0)=x] = O(\theta)$ uniformly for all $x \in I$ and $t \geq 0$. For any $x \in I$, the differential equation $\frac{dy(\tau)}{d\tau} = w(y(t))$ has a unique solution $y(\tau) = y(\tau, x)$ with $y(0) = x$ and $E[\delta x(t)|x(0)=x] = y(t\theta) + O(\theta)$ uniformly for all $x \in I$* and *$t \geq 0$.*
2. *$\frac{x(t)-y(t\theta)}{\sqrt{\theta}}$ has a normal distribution with zero mean and finite variance as $\theta \to 0$ and $t\theta \to \infty$.*

Having presented Theorem 2, we are now ready to prove Theorem 1 by resorting to Theorem 2. This is the main result of this paper. We will prove the convergence of $\hat{Q}_n(q_k)$ for $k = 2, 3, \dots, K-1$ below. The proof for $\hat{Q}_n(q_1)$ and $\hat{Q}_n(q_K)$ can be done in the same manner and are not shown in this paper for the sake of brevity.

Proof We now start by showing that the Markov process based on the updating rules (9)–(11) satisfies the assumptions 1 to 6 in Theorem 2.

$$\begin{aligned}
&E\left(\delta\hat{Q}_n(q_k) \,\middle|\, \hat{Q}_n(q_k)\right) \\
&\quad = E\left(\delta\hat{Q}_n(q_k) \,\middle|\, \hat{Q}_n(q_k) \geq X\right) P\left(\hat{Q}_n(q_k) \geq X\right) \\
&\qquad + E\left(\delta\hat{Q}_n(q_k) \mid \hat{Q}_n(q_k) < X\right) P\left(\hat{Q}_n(q_k) < X\right) \\
&\quad = \beta H\left(\hat{Q}_n(q_k); \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) q_k \hat{Q}_n(q_k)\left(1 - F_X\left(\hat{Q}_n(q_k)\right)\right) \\
&\qquad - \beta H\left(\hat{Q}_n(q_k); \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right)(1-q_k)\hat{Q}_n(q_k) F_X\left(\hat{Q}_n(q_k)\right) \\
&\quad = \beta H\left(\hat{Q}_n(q_k); \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \hat{Q}_n(q_k)\left(q_k - F_X\left(\hat{Q}_n(q_k)\right)\right)
\end{aligned} \tag{13}$$

We now let $\theta = \beta$, $y = \hat{Q}_n(q_k)$ and $w\left(\hat{Q}_n(q_k)\right)$ be equal to "everything" in (13) except β. It is easy to see that assumption 2 in Theorem 2 is satisfied. Next, we turn to assumption 5 which requires that $w\left(\hat{Q}_n(q_k)\right)$ has a Lipschitz derivative with respect to $\hat{Q}_n(q_k)$. Unfortunately it is not obvious that this is satisfied since H has a discontinuous derivative with respect to $\hat{Q}_n(q_k)$ due to the min-function in (6). To show that both assumptions 2 and 5 are satisfied, we need to perform a subtle modification of (13) as follows. A typical example of H as a function of $\hat{Q}_n(q_k)$ is shown as the black curve in Fig. 7. We define a function H^* that is equal to H except for an the interval $[Q^* - \delta, Q^* + \delta]$ (see Fig. 7). In the interval $[Q^* - \delta, Q^* + \delta]$, H^* is a function that is smaller then H (to satisfy the monotone property) and has a Lipschitz derivative. This requires that H^* satisfy the following

$$\begin{aligned}
&H^*\left(Q^* - \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\quad = H\left(Q^* - \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\frac{dH^*}{d\hat{Q}_n(q_k)}\left(Q^* - \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\quad = \frac{dH}{d\hat{Q}_n(q_k)}\left(Q^* - \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&H^*\left(Q^* + \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\quad = H\left(Q^* + \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\frac{dH^*}{d\hat{Q}_n(q_k)}\left(Q^* + \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right) \\
&\quad = \frac{dH}{d\hat{Q}_n(q_k)}\left(Q^* + \delta; \hat{Q}_n(q_{k-1}), \hat{Q}_n(q_{k+1})\right)
\end{aligned}$$

i.e. that the function value and the derivative must be equal for H and H^* in $Q^* - \delta$ and $Q^* + \delta$. It is straightforward to satisfy these criteria, e.g. by fitting a polynomial. H^* is illustrated as the grey curve in Fig. 7 (and is equal to H outside the interval). By reducing the value of δ, H^* will be more and more similar to H. In other words, there exists always a δ such that

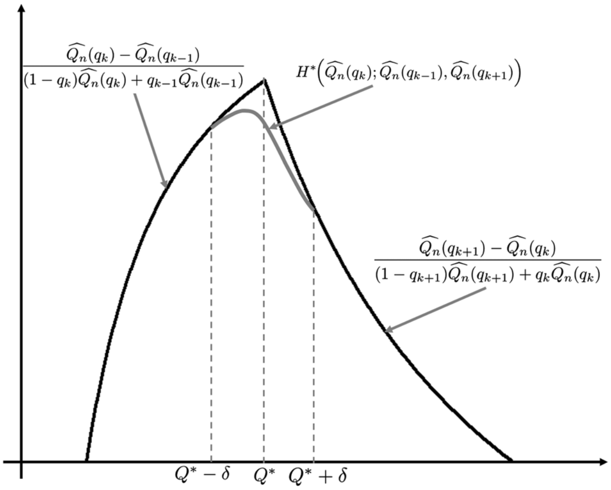

Fig. 7 The black and the grey curves show the functions H and H^*, respectively

$$\left|H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)-H^*\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\right|<\beta\quad\forall\,\hat{Q}_n(q_k)\in\left[\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right]$$

which means that we can write

$$\begin{aligned}&H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\\&\quad=H^*\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)+O(\beta)\end{aligned}\tag{14}$$

Substituting (14) into (13) we obtain

$$\begin{aligned}&\beta H^*\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\hat{Q}_n(q_k)\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)\right)\\&\quad+\beta\hat{Q}_n(q_k)\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)\right)O(\beta)\\&=\beta H^*\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\hat{Q}_n(q_k)\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)\right)+O(\beta^2)\end{aligned}$$

Since H^* has a Lipschitz derivative, we see that with

$$w\left(\hat{Q}_n(q_k)\right)=H^*\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\hat{Q}_n(q_k)\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)\right)\tag{15}$$

both assumptions 2 and 5 are satisfied.

Next we turn to assumption 3.

$$\begin{aligned}&E\left(\delta\hat{Q}_n(q_k)^2\,\middle|\,\hat{Q}_n(q_k)\right)\\&\quad=E\left(\delta\hat{Q}_n(q_k)^2\,\middle|\,\hat{Q}_n(q_k)\geq X\right)P\left(\hat{Q}_n(q_k)\geq X\right)\\&\qquad+E\left(\delta\hat{Q}_n(q_k)^2\,\middle|\,\hat{Q}_n(q_k)<X\right)P\left(\hat{Q}_n(q_k)<X\right)\\&\quad=\beta^2\left(H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)q_k\hat{Q}_n(q_k)\right)^2\left(1-F_X\left(\hat{Q}_n(q_k)\right)\right)\\&\qquad-\beta^2\left(H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)(1-q_k)\hat{Q}_n(q_k)\right)^2F_X\left(\hat{Q}_n(q_k)\right)\end{aligned}$$

$$\begin{aligned}&\mathrm{Var}\left(\delta\hat{Q}_n(q_k)\,\middle|\,\hat{Q}_n(q_k)\right)=E\left(\delta\hat{Q}_n(q_k)^2\,\middle|\,\hat{Q}_n(q_k)\right)-E\left(\delta\hat{Q}_n(q_k)\,\middle|\,\hat{Q}_n(q_k)\right)^2\\&\quad=\beta^2\left(H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)q_k\hat{Q}_n(q_k)\right)^2\left(1-F_X\left(\hat{Q}_n(q_k)\right)\right)\\&\qquad-\beta^2\left(H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)(1-q_k)\hat{Q}_n(q_k)\right)^2F_X\left(\hat{Q}_n(q_k)\right)\\&\qquad+\beta^2\left(H\left(\hat{Q}_n(q_k);\hat{Q}_n(q_{k-1}),\hat{Q}_n(q_{k+1})\right)\hat{Q}_n(q_k)\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)\right)\right)^2\end{aligned}\tag{16}$$

We see that assumption 3 is satisfied with $s\left(\hat{Q}_n(q_k)\right)$ equal to everything in (16) except β^2. Since H is Lipschitz (has a bounded derivative), $s\left(\hat{Q}_n(q_k)\right)$ is Lipschitz and assumption 6 is also satisfied. Assumption 4 can now be proved in the same manner.

We will use the results of Norman to prove the convergence. It is easy to see that $w\left(\hat{Q}_n(q_k)\right)$ in (15) admits two roots $\hat{Q}_n(q_k)=F_X{}^{-1}(q_k)=Q(q_k)$ and $\hat{Q}_n(q_k)=0$. By introducing an arbitrarily small lower bound $Q_{\min}>0$ on estimate $\hat{Q}_n(q_k)$, we can avoid the $\hat{Q}_n(q_k)=0$. This is easily implemented by modifying the update rules and adding $Q_{\min}$ to the right term of Eqs. (9)–(11). Therefore, the unique root becomes $\hat{Q}_n(q_k)=F_X{}^{-1}(q_k)=Q(q_k)$.

We now differentiate to get

$$\begin{aligned}&\frac{\mathrm{d}w\left(\hat{Q}_n(q_k)\right)}{\mathrm{d}\hat{Q}_n(q_k)}=\left(\frac{\mathrm{d}H^*}{\mathrm{d}\hat{Q}_n(q_k)}+H^*\right)\\&\left(q_k-F_X\left(\hat{Q}_n(q_k)\right)-H^*\hat{Q}_n(q_k)f\left(\hat{Q}_n(q_k)\right)\right)\end{aligned}$$

We substitute the unique root $Q(q_k)$ for $\hat{Q}_n(q_k)$ and get

$$\begin{aligned}&\left.\frac{\mathrm{d}w\left(\hat{Q}_n(q_k)\right)}{\mathrm{d}\hat{Q}_n(q_k)}\right|_{\hat{Q}_n(q_k)=Q(q_k)}\\&\quad=\left(\frac{\mathrm{d}H^*}{\mathrm{d}\hat{Q}_n(q_k)}+H^*\right)(q_k-F_X\left(Q(q_k)\right)-H^*Q(q_k)f\left(Q(q_k)\right))\\&\quad=0-H^*Q(q_k)f_X(Q(q_k))<0.\end{aligned}$$

This gives

$$\lim_{n\beta\to\infty,\beta\to0}E\left(\hat{Q}_n(q_k)\right)=Q(q_k)+O(\beta)$$

and

$$\mathrm{Var}\left(\hat{Q}_n(q_k)\right)=O(\beta)$$

Consequently

$$\lim_{n\beta\to\infty,\beta\to0}\hat{Q}_n(q_k)=Q(q_k)$$

□

References

1. Abbasi B, Guillen M (2013) Bootstrap control charts in monitoring value at risk in insurance. Expert Syst Appl 40(15):6125–6135
2. Alkhamees N, Fasli M (2016) Event detection from social network streams using frequent pattern mining with dynamic support values. In: 2016 IEEE international conference on big data (big data). IEEE, pp 1670–1679

3. Arasu A, Manku GS (2004) Approximate counts and quantiles over sliding windows. In: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems. ACM, pp 286–296
4. Cao J, Li L, Chen A, Bu T (2010) Tracking quantiles of network data streams with dynamic operations. In: INFOCOM, 2010 proceedings IEEE. IEEE, pp 1–5
5. Cao J, Li LE, Chen A, Bu T (2009) Incremental tracking of multiple quantiles for network monitoring in cellular networks. In: Proceedings of the 1st ACM workshop on mobile internet through cellular networks. ACM, pp 7–12
6. Chen F, Lambert D, Pinheiro JC (2000) Incremental quantile estimation for massive tracking. In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 516–522
7. Choi B-Y, Moon S, Cruz R, Zhang Z-L, Diot C (2007) Quantile sampling for practical delay monitoring in internet backbone networks. Comput Netw 51(10):2701–2716
8. Datar M, Gionis A, Indyk P, Motwani R (2002) Maintaining stream statistics over sliding windows. SIAM J Comput 31(6):1794–1813
9. Fang Y, Zhang H, Ye Y, Li X (2014) Detecting hot topics from twitter: a multiview approach. J Inf Sci 40(5):578–593
10. Gilbert AC, Guha S, Indyk P, Kotidis Y, Muthukrishnan S, Strauss MJ (2002) Fast, small-space algorithms for approximate histogram maintenance. In: Proceedings of the thiry-fourth annual ACM symposium on theory of computing. ACM, pp 389–398
11. Gilbert AC, Kotidis Y, Muthukrishnan S, Strauss MJ (2002) How to summarize the universe: dynamic maintenance of quantiles. In: Proceedings of the 28th Kotidisinternational conference on very large data bases. VLDB Endowment, pp 454–465
12. Gilli M et al (2006) An application of extreme value theory for measuring financial risk. Comput Econ 27(2–3):207–228
13. Hammer HL, Yazidi A (2017) Incremental quantiles estimators for tracking multiple quantiles. In: International conference on industrial, engineering and other applications of applied intelligent systems. Springer, pp 202–210
14. Hammer HL, Yazidi A (2018) A novel incremental quantile estimator using the magnitude of the observations. In: 2018 26th Mediterranean conference on control and automation (MED). IEEE, pp 290–295
15. Hammer HL, Yazidi A, John OB (2018) On the classification of dynamical data streams using novel anti-Bayesian techniques. Pattern Recognit 76:108–124
16. Hammer HL, Yazidi A, Rue H (2018) A new quantile tracking algorithm using a generalized exponentially weighted average of observations. Appl Intell. https://doi.org/10.1007/s10489-018-1335-7
17. Lin X, Lu H, Xu J, Yu JX (2004) Continuously maintaining quantile summaries of the most recent n elements over a data stream. In: Proceedings of the 20th international conference on data engineering, 2004. IEEE, pp 362–373
18. Luo G, Wang L, Yi K, Cormode G (2016) Quantiles over data streams: experimental comparisons, new analyses, and further improvements. VLDB J 25(4):1–24
19. Ma Q, Muthukrishnan S, Sandler M (2013) Frugal streaming for estimating quantiles. In: Brodnik A, López-Ortiz A, Raman V, Viola A (eds) Space-efficient data structures, streams, and algorithms. Springer, Berlin, pp 77–96
20. Norman MF (1972) Markov processes and learning models, vol 84. Academic Press, New York
21. Sommers J, Barford P, Duffield N, Ron A (2007) Accurate and efficient SLA compliance monitoring. In: ACM SIGCOMM computer communication review, vol 37. ACM, pp 109–120
22. Sommers J, Barford P, Duffield N, Ron A (2010) Multiobjective monitoring for SLA compliance. IEEE/ACM Trans Netw (TON) 18(2):652–665
23. Song G, Ye Y, Zhang H, Xu X, Lau RYK, Liu F (2016) Dynamic clustering forest: an ensemble framework to efficiently classify textual data stream with concept drift. Inf Sci 357:125–143
24. Stahl V, Fischer A, Bippus R (2000) Quantile based noise estimation for spectral subtraction and wiener filtering. In: 2000 IEEE international conference on acoustics, speech, and signal processing, 2000. ICASSP'00, vol 3. IEEE, pp 1875–1878
25. Tierney L (1983) A space-efficient recursive procedure for estimating a quantile of an unknown distribution. SIAM J Sci Stat Comput 4(4):706–711
26. Tiwari N, Pandey PC (2018) A technique with low memory and computational requirements for dynamic tracking of quantiles. J Signal Process Syst. https://doi.org/10.1007/s11265-017-1327-6
27. Yazidi A, Hammer HL (2017) Multiplicative update methods for incremental quantile estimation. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2017.2779140
28. Zhang L, Guan Y (2008) Detecting click fraud in pay-per-click streams of online advertising networks. In: The 28th international conference on distributed computing systems, 2008. ICDCS'08. IEEE, pp 77–84
29. Zhang X, Alexander L, Hegerl GC, Jones P, Tank AK, Peterson TC, Trewin B, Zwiers FW (2011) Indices for monitoring changes in extremes based on daily temperature and precipitation data. Wiley Interdiscip Rev Clim Change 2(6):851–870