Efficient Use of Differentially Private Binary Trees[*]

James Honaker[†]

October 6, 2019

Abstract

Binary trees can be made differentially private by adding noise to every node and leaf. In such form they allow multifaceted exploration of a variable without revealing any individual information. While a differentially private binary tree can be used and read just like its conventional exact-valued analog, realizing that different combinations of nodes contain overlapping answers to the same information allows us to bring the statistical properties of multiple measurements under measurement error to noisy binary trees to create statistically efficient node estimates. We construct estimators that correctly use all available information in the tree, thus decreasing the error of nodes by up to eighty percent for the same level of privacy protection.

Differentially private binary trees are important summary statistics for a broad variety of uses and algorithms. They are central to the algorithm of Dwork et al (2010) for releasing private streaming data, and used in numerous adaptations of this problem, such as Chan et al (2012), Cao et al (2013), and Thakurta and Smith (2013). A binary tree can be used to compose probability and cumulative densities of variables, range queries, as well as means, medians, modes and variances by Monte Carlo integration. Thus the release of a private binary tree can be a broadly useful privacy preserving means of allowing exploration of a variable, as for example in the statistical release of a non-interactive curator (Dwork and Smith, 2009).

Due to the importance of binary trees, several heuristic adaptations of their use have been studied that bring about some improved accuracy for the same level of privacy guarantee (Hay et al 2010, Xu et al 2012, and relatedly Xiao et al 2010). We show here how to derive an optimally *efficient* use of a private binary tree, in the precise sense of providing minimum variance unbiased estimates. That is, we show how to refine a private tree in a manner that makes full and optimal use of all the information the tree contains. This is accomplished by linking the tree structure to the known statistical properties of multiple measures under measurement error.

1 Problem Statements

Consider a perfect binary tree, in which every node, t_i, is the sum of all leaves below that node, plus a random draw, ϵ_i from some fixed distribution $f(.)$, constructed to guarantee differential privacy. As represented below, the true values are denoted a through h at the leaves, the differentially private value revealed at any node is given to the right, and the notation for index i both numbers the nodes sequentially, and also describes the path from the top to reach that particular node, as a sequence of left (0) and right (1) progressions.

Assume one desires an estimate of $a + b + c + d$. From the tree, the most easily attainable answer is the value of t_{10}; if the tree had no added errors, this is how the tree would normally be

[*] For helpful comments and discussions we thank Marco Gaboardi, Kobbi Nissim, Or Sheffet, and Salil Vadhan. This work was supported by the NSF (CNS-1237235), and the Alfred P. Sloan Foundation.

[†] Senior Research Scientist, Institute for Quantitative Social Science, 1737 Cambridge Street, Cambridge, MA 02138 (jhonaker@iq.harvard.edu, hona.kr)

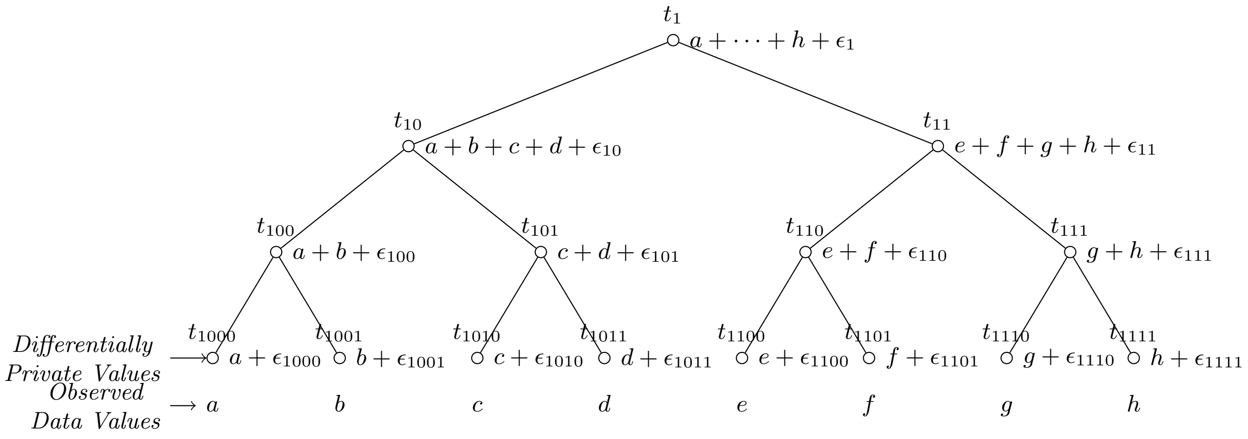


Figure 1: *Example tree, showing leaf values a through h below, node labels above, and differentially private values to the right. Each value includes a draw of ϵ from some fixed distribution to ensure every value is differentially private.*

read. However, the sum of the two nodes below, $(t_{100} + t_{101})$, is also an estimate of this quantity, as is the larger sum $(t_{1000} + t_{1001} + t_{1010} + t_{1011})$. We expect these latter estimates to be more noisy than t_{10}, but they have informational value. They are analogous to repeated measurements of the same quantity, with different levels of measurement error. Some correct weighting over these three estimates is more *efficient*, that is, makes more use of the available information, than simply using the value of the node at t_{10}. The optimal weighting can be determined by considering the relative amounts of measurement error in the sums, as they contain differing numbers of random draws. A first question is, *how do we derive a correct weighting across different estimates of the same quantity from different information in the tree.*

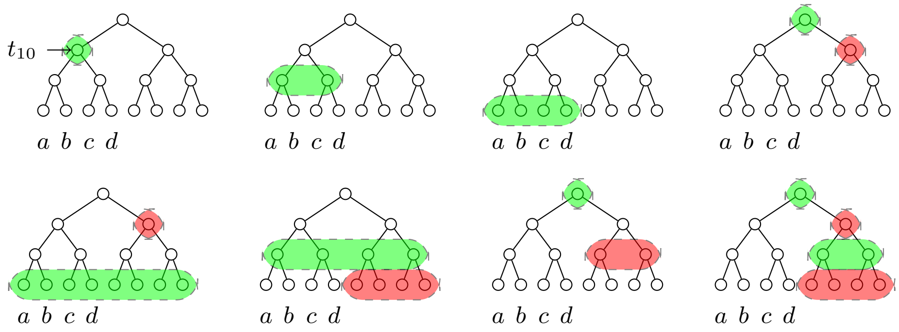


Figure 2: *Eight of the possible ways of using the nodes on the tree to estimate $a + b + c + d$, the sum of the left four leaves. Nodes in green are added, and nodes in red subtracted to create an estimate.*

However, there are yet more possible estimates from this tree, that use different information entirely. The difference, $(t_1 - t_{11})$ is another estimate of $a+b+c+d$, as is $(t_1 - t_{110} - t_{111})$ or even $(t_1 - t_{1100} - t_{1101} - t_{1110} - t_{1111})$. Various other estimates, composed by simple sums, or sums and differences, are shown graphically in figure 2. However, these three estimates are now correlated in their errors, as they all contain ϵ_1, the error of the top node, so simple weighting is not efficient. The second question to answer is, *how do we construct a set of estimates to weight together, that we know efficiently uses all the available information in all the nodes (both below and not below) the node of interest.*

We first show how to weight together alternate answers of the same quantity of interest from the tree. Then we show how to construct a set of answers that make full use of all the information available in the tree. Below we consider the distribution of error terms to be Gaussian for ease of exposition and space reasons, but all the findings can be translated to equivalent results in the case of Laplace noise.

2 Efficiency Under Multiple Mismeasurements

Consider multiple measurements $m_1, \dots, m_n$, of the same quantity, y, each with their own known measurement standard error, $\sigma_1, \dots, \sigma_n$. An estimator is a function for mapping from a set of data to an estimate of a desired quantity of interest. An *efficient* estimator for y gives estimates that are minimum variance and unbiased. Efficient estimates have the lowest squared error from the truth, among all estimates that are unbiased. Efficiency is one of many common *desiderata* of statistical estimators, and conventionally means that maximal use of the available information in the data is being used to create the estimate of the question of interest; it is a broadly desirable property although in some contexts or problems it might be traded for other desired properties. We first show three properties of efficient estimators of multiple measurements that we will use throughout.

Remark 1. *For a set of independent measurements, M, among all linear functions $\hat{y}(M) = cM$, the efficient estimator $\hat{y}^*$ weights measurements inverse to their error variance.*

If we construct an estimator $\hat{y}$ that uses the available measures in the form $\hat{y} = c_1 m_1 + \cdots + c_n m_n = \sum c_i m_i = \sum c_i (y + \epsilon_i)$, the bias is given by:

$$\text{bias}(\hat{y}) = \text{E}[y - \hat{y}] = y - y \sum c_i - \sum c_1 \text{E}[\epsilon_i]. \tag{1}$$

If each ϵ_i comes from some distribution that has mean zero, then the right term in 1 disappears and zero bias thus implies $\sum c_i = 1$. The variance can be derived as:

$$\begin{aligned}\text{var}(\hat{y}) &= \text{E}[(\bar{\hat{y}} - \hat{y})^2] = \text{E}[(\bar{y} + \sum c_i \text{E}(\epsilon_i) - \sum c_i (y + \epsilon_i))^2] = \text{E}[(\bar{y} - y \sum c_i - \sum c_i \epsilon_i)^2] = \text{E}[(\bar{y} - y - \sum c_i \epsilon_i)^2] \\ &= \text{E}[(\bar{y} - y)^2 - \sum (\bar{y} - y) c_i \epsilon_i + \sum c_i \epsilon_i \sum c_j \epsilon_j] \end{aligned} \tag{2}$$

If ϵ_i, $\epsilon_{j \neq i}$ and y are uncorrelated, then many cross-products have expectation zero and this reduces to:

$$\text{var}(\hat{y}) = \text{E}[(\bar{y} - y)^2 + \sum c_i^2 \epsilon_i^2] = var(y) + \sum c_i^2 \sigma_i^2 \tag{3}$$

Minimization of the variance with respect to c, under the constraint $\sum c_i = 1$, is a straightforward quadratic programming problem giving:

$$c_i = \frac{\sigma_i^{-2}}{\sum_{n=1}^{N} \sigma_n^{-2}} \tag{4}$$

Thus the optimal efficient weights of multiple measures are inversely proportional to the variances of the measurement errors. □

Remark 2. *The standard error of this efficient linear estimator function has two convenient simplified forms:*

$$\sigma(\hat{y}^*) = \sqrt{\sum c_i^2 \sigma_i^2} = \sqrt{\frac{\sum(\sigma_i^{-4} \sigma_i^2)}{(\sum \sigma_i^{-2})^2}} = \boxed{\left(\sum \sigma_i^{-2}\right)^{-1/2}} = \sigma_j \sqrt{\frac{\sigma_j^{-2}}{\sum \sigma_i^{-2}}} = \boxed{\sigma_j \sqrt{c_j},\ \forall j} \tag{5}$$

Remark 3. *The efficient linear estimator function is generalized associative, i.e.,* $\hat{y}^*(M) = \hat{y}^*(\hat{y}^*(M_I), M_{-I})$.

For a sequence of measurements $M = (m_1, \ldots, m_N)$, partition the measurements into two mutually exclusive and exhaustive sets M_I and M_{-I}. Consider the case of $m_j \in M_I$. Let c_j be the weight of the m_j measurement for $\hat{y}^*(M)$ and c'_j the weight from the estimate using the subset $\hat{y}^*(M_I)$, and σ_* the resulting standard error of some efficient estimator. Then these are:

$$c_j = \frac{\sigma_j^{-2}}{\sum_{k=1}^{N} \sigma_k^{-2}}, \quad \sigma_* = \sigma(\hat{y}^*(M)) = \frac{1}{\sqrt{\sum_{k=1}^{N} \sigma_k^{-2}}} \quad \Bigg| \quad c'_j = \frac{\sigma_j^{-2}}{\sum_{k \in I} \sigma_k^{-2}}, \quad \sigma'_* = \sigma_\sigma(\hat{y}^*(M_I)) = \frac{1}{\sqrt{\sum_{k \in I} \sigma_k^{-2}}} \tag{6}$$

Let c''_j be the weight resulting when the estimate on the subset is added to the excluded data, that is from $\hat{y}^*(\hat{y}^*(M_I), M_{-I})$. The weight given to the first term is $\sigma_*^{-2}/(\sigma_*^{-2} + \sum_{k \notin I} \sigma_k^{-2})$ therefore the total weight on the m_j measurement is:

$$c''_j = \frac{\sigma_*^{-2} c'_j}{\sigma_*^{-2} + \sum_{k \notin I} \sigma_k^{-2}} = \frac{\frac{\sigma_j^{-2} \sum_{k \in I} \sigma_k^{-2}}{\sum_{k \in I} \sigma_k^{-2}}}{\sum_{k \in I} \sigma_k^{-2} + \sum_{k \notin I} \sigma_k^{-2}} = \frac{\sigma_j^{-2}}{\sum_{k \in I} \sigma_k^{-2} + \sum_{k \notin I} \sigma_k^{-2}} = \frac{\sigma_j^{-2}}{\sum_{k=1}^{N} \sigma_k^{-2}} = c_j \tag{7}$$

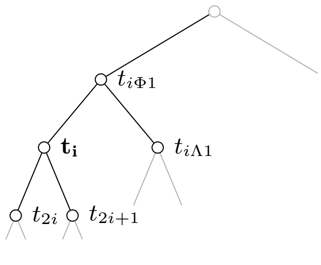
It is even simpler to show this holds for $c''_j : j \in M_{-I}$, as we just change the numerator in 7. Since the weighting on every component m_j is the same under the two estimators, $\hat{y}^*(M) = \hat{y}^*(\hat{y}^*(M_I), M_{-I})$. □

The first remark shows the optimal efficient weights of multiple measures are inversely proportional to the variances of the measurement errors. We will use this to construct estimators for combining tree nodes and other quantities. The second remark gives us two convenient simplifications for the standard error of this estimator. The third remark shows that an efficient estimator for a vector of information, can be constructed incrementally by adding new information, appropriately weighted, into a previously computed estimate. The order in which we group and combine measurements does not influence the resulting estimator. We will use this to construct iterative definitions of some quantities that act as rolling estimators sequentially adding new nodes.

3 Efficient Estimators

3.1 Notation

Let t_i denote a node, where index i sequentially labels the nodes, at depth $d_i = \lfloor \log_2(i) \rfloor$ in a tree of depth D. Given the binary nature, t_{2i} and t_{2i+1} will be the nodes directly below t_i. Let $t_{i\Phi 1}$ represent the node directly above t_i, and $t_{i\Lambda 1}$ represent the adjacent node reached by changing the path only at $t_{i\Phi 1}$. These can be expressed as $i\Phi 1 = \lfloor i/2 \rfloor$ and piecewise as $i\Lambda 1 = \{i{+}1,\ i \text{ even};\ i{-}1,\ i \text{ odd}.$ If i is expressed base 2, then $i\Lambda 1$ is i with the last bit flipped, and $i\Phi 1$ is i bitwise shifted to the right.



3.2 Estimation from below

Following remark 1, denote t_i^- as the optimal estimate of the query at node t_i using only the information at node i or below, as constructed (along with its standard error) as:

$$t_i^- = \sum_{j=0}^{D-d_i} w_j q_j; \quad \sigma(t_i^-) = \Big(\sum_{j=0}^{D-d_i} s_j^{-2}\Big)^{-1/2} \text{ where: } q_j = \sum_{k=0}^{2^j-1} t_{2^j i+k}; \quad w_j = \frac{s_j^{-2}}{\sum_{k=0}^{D-d_i} s_k^{-2}}; \quad s_j = \Big(\sum_{k=0}^{2^j-1} \sigma^2_{2^j i+k}\Big)^{1/2}. \tag{8}$$

Here the q's represent each possible estimate created by summing all the observations below t_i at some particular depth of the tree, the s's compute the inverse variances of these sums, and t_i^- the efficient weighted average of these. From this definition, t_i^- and $\sigma(t_i^-)$ can be calculated for any node i.

In the case of constant masking error, where $\sigma_i = \sigma,\ \forall i$, then $s_j^{-2} = 1/(2^{d_j-d_i}\sigma^2)$ and the sum of all the inverse variances goes to $2/\sigma^2$ as the tree gets taller, and $\sigma(t_i^-)$ converges to $\sigma/\sqrt{2}$.

3.2.1 Example

In our estimate of $a+b+c+d$ from figure 1, using t_{10} and all the nodes below it, if every node has equal σ then:

$$t_{10}^- = \frac{t_{10}(1) + (t_{100} + t_{101})(1/2) + (t_{1000} + t_{1001} + t_{1010} + t_{1011})(1/4)}{7/4} \tag{9}$$

In this example, t_{10}, the most direct estimate, only gets 4/7, or approximately 0.57, of the total weight.

3.2.2 Iterative definition

Following remark 3, since any t_i^- already contains all the information from nodes below it, all the node estimates can be iteratively computed from the bottom of the tree to the top, using the new information in some node, and the efficient estimates of the two nodes directly below it, as:

$$\text{For } d_i < D: \quad t_i^- = w\, t_i + (1-w)(t_{2i}^- + t_{2i+1}^-); \quad w = \frac{\sigma_i^{-2}}{\sigma_i^{-2} + (\sigma(t_{2i}^-)^2) + \sigma(t_{2i+1}^-)^2)^{-1}}; \quad \sigma(t_i^-) = \sigma_i\sqrt{w} \tag{10}$$

with the terminal nodes of the tree defined simply as $t_i^- = t_i$ and $\sigma(t_i^-) = \sigma_i$. This is equivalent to equation 8 but much simpler to compute if every node in the tree is to be efficiently estimated.

3.3 Estimation from above

Intuitively, any node can be computed as the difference between the node above it in the tree, $t_{i\Phi 1}$, and the node adjacent, $t_{i\Lambda 1}$. For example, an estimate of t_{10} is $t_1 - t_{11}$. In general, an estimate of t_i is $t_{i\Phi 1} - t_{i\Lambda 1}$. However, neither of these terms are themselves informationally efficient, nor are we using the information in t_i. This leads to the following iterative definition. Denote t_i^+ as the optimal estimate of the query at node t_i using no information below node i, as:

$$\text{For } i > 1: \quad t_i^+ = w\, t_i + (1-w)(t_{i\Phi 1}^+ - t_{i\Lambda 1}^-); \quad w = \frac{\sigma_i^{-2}}{\sigma_i^{-2} + \big(\sigma(t_{i\Phi 1}^+)^2 + \sigma(t_{i\Lambda 1}^-)^2\big)^{-1}}; \quad \sigma(t_i^+) = \sigma_i\sqrt{w} \tag{11}$$

with t_1^+ defined as t_1, and $\sigma(t_1^+)$ defined as σ_1 when N is private information, and 0 when N is known. Intuitively, the right-most terms are the most efficient estimates of the node above and the node horizontally adjacent. Appropriately weighted with the node itself, this gives the best estimate of the node, without using any information below the node. Calculation of t_i^+ is only possible after calculation of the same quantity in nodes above this in the tree, thus this can be computed iteratively from the top down. For constant σ for each node, in the limit of the best case, when t_i is close to the top of a tall tree, this new standard error is $\sigma/\sqrt{3}$. Thus in this best case, there is slightly more information in t_i^+ than t_i^-.

3.3.1 Example (check this)

In the running example of t_{10}, in a tree of depth four, with constant σ, and known N, we have:

$$t^+_{10} = w\, t_{10} + (1-w)\Big(t_1 - \frac{t_{11}(1) + (t_{110}+t_{111})(1/2) + (t_{1100}+t_{1101}+t_{1110}+t_{1111})(1/4)}{7/4}\Big)$$

$$w = \frac{\sigma^{-2}}{\sigma^{-2} + c_i^- \sigma^{-2}} = \frac{1}{1+c_i^-} \qquad \mathrm{se}(t^+_{10}) = \frac{\sigma}{\sqrt{1+c_i^-}} = \frac{\sigma}{\sqrt{11/4}} \tag{12}$$

In this example case, for known N, the standard error of $\sigma/\sqrt{11/4}$ is close to the best possible case of $\sigma/\sqrt{3}$.

3.4 Fully efficient estimation

The estimates, t_i^- and t_i^+ overlap informationally, as they both contain t_i. However, for any node, t_i, the estimates t_i^-, $t^-_{i\Lambda1}$, and $t^+_{i\Phi1}$ strictly partition the dataset. That is, every node contributes to exactly one of these estimates. Therefore they have independent errors, and make full use of the available information in the tree. As an example the sets of nodes that construct these for t_{100} are shown for a tree in figure 3. These estimates can be weighted together for an optimal estimate, t_i^*, as:

$$t_i^* = w\, t_i^- + (1-w)\,(t^+_{i\Phi1} - t^-_{i\Lambda1})$$

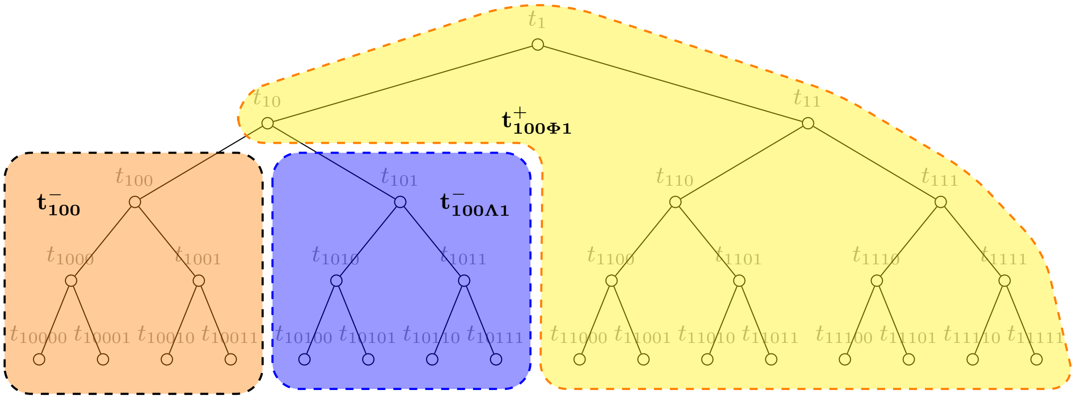
$$w = \frac{\sigma(t_i^-)^{-2}}{\sigma(t_i^-)^{-2} + \big(\sigma(t^+_{i\Phi1})^2 + \sigma(t^-_{i\Lambda1})^2\big)^{-1}}$$

$$\sigma(t_i^*) = \sigma(t_i^-)\sqrt{w} \tag{13}$$

In the constant σ, tall tree cases described previously, where $\sigma(t_i^-) = \sigma(t^-_{i\Lambda1}) = \sigma/\sqrt{2}$ and $\sigma(t^+_{i\Phi1}) = \sigma/\sqrt{3}$ this leads to $\sigma(t_i^*) = \sigma(\sqrt{2}/7) \approx .202\,\sigma$ or variance $\sigma^2(t_i^*) = \sigma^2 49/2$. This is about an eighty percent reduction in error, or a reduction in variance by a factor of 25, for nodes near the top of tall trees.

4 Simulation for Nodes

To demonstrate the estimators, we simulated ten thousand perfect binary trees of depth 5. Each of the true values of the 16 leaves was drawn from a Poisson distribution with mean 10. Every node in the tree was then computed, and then masking errors drawn from a mean zero Gaussian with standard deviation 3; in expectation then, each node has squared error of 9. For each tree



t_1
t_{10}
t_{11}
$\mathbf{t}^+_{100\Phi1}$
$\mathbf{t}^-_{100}$
$\mathbf{t}^-_{100\Lambda1}$
t_{100}
t_{101}
t_{110}
t_{111}
t_{1000}
t_{1001}
t_{1010}
t_{1011}
t_{1100}
t_{1101}
t_{1110}
t_{1111}
t_{10000} t_{10001} t_{10010} t_{10011} t_{10100} t_{10101} t_{10110} t_{10111} t_{11000} t_{11001} t_{11010} t_{11011} t_{11100} t_{11101} t_{11110} t_{11111}

Figure 3: *The sets of nodes that contribute to* t^-_{100}, $t^-_{100\Lambda1}$ *and* $t^+_{100\Phi1}$. *Together, they cover all nodes, without any overlap, and can be weighted to form* t^*_{100}, *the fully efficient estimate.*

we computed the estimates from above, below and the combined efficient estimates, and calculated the squared error of every node. We assume N is known.

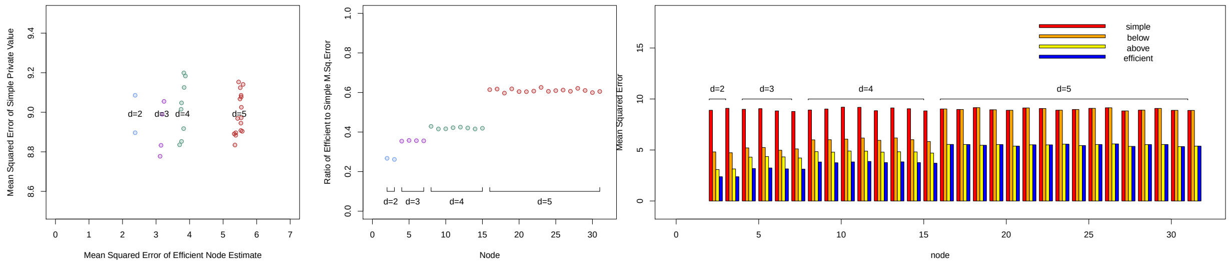


Figure 4: *The left graph shows mean squared error for each node from the fully efficient estimates compared to the simple private values (and the center graph their ratios). On the right, the distribution for simple Gaussian errors (red) is plotted next to the* above *(orange),* below *(yellow) and* fully efficient *(blue) estimators for every node. The most dramatically reduced errors are for nodes highest in the tree.*

The left of figure 4 plots the mean squared error across all simulation of the efficient estimates against the mean squared error of the original Gaussian private values of the nodes. The errors for the efficient estimator cluster by depth, with nodes higher up the tree containing less error. Care should taken reading the axes; the difference in error is dramatic enough that the scaling on the y-axis does not even exist in the range of the x-axis. The center figure shows the ratio of the squared error in the efficient estimates to the original private nodes. We see that the high nodes have about one quarter the squared error, and even the terminal leaves have about a one-third reduction. The right of figure 4 shows this in slightly more nuance. Here we can see the squared error in turn for the private nodes, the below and above estimators, and finally for the efficient estimator. We can see that while the below estimator is more intuitive, the above estimator always has slightly less error. In the terminal nodes, we see a tie between the below estimator and the simple private node values, because there is no information below the node to exploit; similarly, there is a tie between the above estimator and the efficient estimator, because at these nodes the definition is exactly the same. Of course the efficient estimator performs equal or better to all other results across all nodes.

References

Jianneng Cao, Qian Xiao, Gabriel Ghinita, Ninghui Li Elisa Bertino, Kian-Lee Tan. 2013. "Efficient and Accurate Strategies for Differentially-Private Sliding Window Queries" *EDBT/ICDT'13*

Cynthia Dwork and Adam Smith. 2009. "Differential Privacy for Statistics: What we Know and What we Want to Learn." *Journal of Privacy and Confidentiality* 1(2): pp. 135-154.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. 2010. "Differential privacy under continual observation." In Proceedings of the 42nd ACM symposium on Theory of Computing.

T-H. Hubert Chan, Mingfei Li, Elaine Shi, and Wenchang Xu. 2012. "Differentially Private Continual Monitoring of Heavy Hitters from Distributed Streams" *PETS'12* Proceedings of the 12th International Conference on Privacy Enhancing Technologies.

Michael Hay, Vibhor Rastogi, Gerome Miklau, Dan Suciu. 2010. "Boosting the Accuracy of Differentially Private Histograms Through Consistency." *Proceedings of the VLDB Endowment* 3(1)

Abhradeep Thakurta and Adam Smith. 2013 "(Nearly) Optimal Algorithms for Private Online Learning in Full-information and Bandit Settings" *NIPS 2013*.

Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Ge Yu. 2012. "Differentially Private Histogram Publications." *ICDE'12* Proceedings of the 2012 IEEE 28th International Conference on Data Engineering. pp. 32-43.

X. Xiao, G. Wang, and J. Gehrke. 2010 "Differential privacy via wavelet transforms," *ICDE 2010*, pp. 225–236.