RODIAN: Robustified Median
Seong Hun Lee and Javier Civera

Abstract—We propose a robust method for averaging numbers contaminated by a large proportion of outliers. Our method, dubbed RODIAN, is inspired by the key idea of MINPRAN [1]: We assume that the outliers are uniformly distributed within the range of the data and we search for the region that is least likely to contain outliers only. The median of the data within this region is then taken as RODIAN. Our approach can accurately estimate the true mean of data with more than 50% outliers and runs in time $O(n \log n)$. Unlike other robust techniques, it is completely deterministic and does not rely on a known inlier error bound. Our extensive evaluation shows that RODIAN is much more robust than the median and the least-median-of-squares. This result also holds in the case of non-uniform outlier distributions.

Index Terms—Averaging, median, measure of central tendency, robust statistics, outlier-resistant method.

I. INTRODUCTION

AVERAGING means finding the most representative value of a given set of data points. For one-dimensional numbers, it can be the arithmetic mean, the median or other measures of central tendency. All these measures have different properties, one of which is the robustness to outliers. This is an important property to consider because outliers can severely degrade the averaging accuracy if not handled properly. Robust averaging is a useful technique in a wide variety of domains, including pattern recognition [2], [3], [4], image processing [5], [6], [7], 3D computer vision [8], [9], [10], biomedical engineering [11], [12], [13], economics/econometrics [14], [15], [16], information science [17], [18], [19], environmental studies [20], [21], geochemistry [22], [23], forensic science [24], psychology [25] and database research [26].

If the given data set contains a small number of outliers, it may be sufficient to use the median, as, contrary to the mean, it is robust up to a certain outlier ratio [27]. The median can be considered as a specific case of the alpha-trimmed mean with $\alpha = 0.5$. The alpha-trimmed mean [28] of $n$ numbers is defined as the arithmetic mean after truncating the largest $\frac{\alpha}{2}n$ and the smallest $\frac{\alpha}{2}n$ elements. This method assumes that the outliers are likely to be located at the high and low ends of the sorted data. As a result, it fails when a large number of outliers are located mostly on one side of the long tails.

To handle such cases, a more elaborate method should be used. One popular example is the maximum likelihood-type estimator (M-estimator) [29]. Paired with iteratively reweighted least squares (IRLS) [30], it can effectively down-weight the influence of outliers. However, M-estimators, such as the Huber function, often require a control parameter to be carefully tuned to the inlier error distribution. Also, their robustness strongly depends on the initial seed, and accurate initialization in the presence of many outliers is already a non-trivial problem in and of itself.

Another popular robust estimation method is RANSAC [31]. It involves random sampling, but it can be made deterministic for the 1D averaging problem if we simply pick every number as a sample once. While this method can handle a very large number of outliers, it incurs a computational cost of $O(n^2)$ and requires the prior knowledge of the inlier error bound.

The least-median-of-squares (LMedS) [32], on the other hand, does not require any prior knowledge. For the 1D averaging problem, the LMedS can be obtained by finding the data point that yields the smallest median deviation from the rest. This would involve $O(n^2 \log n)$ computations. Like the median, the LMedS has a breakdown point of 50%.

Another method that does not rely on a known inlier error bound is MINPRAN [1]. This method is more robust than the LMedS, as it can handle more than 50% outliers. However, it is slower than the LMedS and has a random nature.

In this work, we propose RODIAN, a novel robust measure of central tendency. Our method is inspired by the core idea of MINPRAN [1]: We assume that the outliers follow a uniform distribution and find the median in the bounded region that is least likely to contain outliers only. Unlike MINPRAN, however, our method is deterministic and runs in time $O(n \log n)$.

Also, unlike RANSAC and Huber-like cost functions [29], no parameter tuning is needed to account for different inlier distributions. Our experiments show that RODIAN can handle more than 50% outliers, outperforming the median and the LMedS [32] in terms of robustness. We release our code at https://seonghun-lee.github.io.

The remainder of this paper is organized as follows: We detail our method in Section II and present the evaluation results in Section III. In Section IV, we discuss the limitation of our work. Finally, conclusions are drawn in Section V.

II. METHOD

A. Main Idea

Suppose that we are given a set of numbers. Each number is either an inlier or an outlier, but we do not know which is which. Assuming that the inliers are scattered around a certain number $\mu$, how can we estimate $\mu$ from this noisy, outlier-contaminated data? Our approach is to find the most densely populated region in the data and take the median value in that region. Now the question is how to determine this region.

1This is made possible because, unlike MINPRAN [1], we do not use random sampling and we fix the number of inlier bounds we evaluate (by fixing the number of histograms, as will be explained in Section II).
One simple heuristic approach is to build a histogram and find the tallest bin. Then, the edges of this bin correspond to the upper and lower bounds of the densest region. This is a reasonable approach, but there is one problem: The histogram can be constructed in many different ways. If we constrain the lower edge of the first bin to be the minimum value and the upper edge of the last bin to be the maximum value, then we can obtain multiple histograms by varying the number of bins. So, which histogram is the right one to use?

Our answer to this question is that we choose the histogram with the bin that is least likely to have occurred by chance. For any histogram, each of its bins has its associated probability of randomness. For example, if all bins have the exact same height, we can deduce that the numbers are uniformly distributed, and thus random in this sense. By the same token, if one of the bins is significantly taller than the others, then it is unlikely that it occurred due to the randomness. In other words, this very tall bin has a low probability of randomness.

Essentially, what we propose is to build multiple histograms with the different numbers of bins, evaluate the probability of randomness associated with the tallest bin of each histogram, and choose the one that yields the smallest probability of randomness. This is because the minimum probability of randomness implies the maximum probability of containing mostly inliers. This process is illustrated in Fig. 1.

The remaining question is how exactly we compute this probability of randomness. Basically, we adopt a similar idea proposed by Stewart [1] and compute the probability in a binomial distribution, assuming that the random outliers are uniformly distributed across the entire range. Note that if a trial has a probability of success \( p \), the probability of obtaining \( k \) successes from \( n \) trials is given by

\[
P(k) = \binom{n}{k} p^k (1 - p)^{n-k}.
\]  

(1)

In our problem, \( p \) is the probability of a random outlier falling inside the tallest bin, \( k \) is the frequency of this bin, and \( n \) is the size of the data. Since an outlier can fall inside any other bins with an equal probability, we have \( p = 1/b \) where \( b \) is the number of bins. Therefore, the probability of randomness associated with the bin containing \( k \) numbers is given by

\[
P(k) = \binom{n}{k} \left( \frac{1}{b} \right)^k \left( 1 - \frac{1}{b} \right)^{n-k}.
\]  

(2)

In summary, we vary \( b \), find \( k \) by building a histogram, compute \( P(k) \), and repeat this process until we find the value of \( b \) that leads to the smallest \( P(k) \). In the next section, we discuss several strategies we came up with to improve the efficiency of the algorithm.

We note that, in contrast to our approach, MINPRAN [1] uses the probability that at least \( k \) data points fall within the inlier region, which is technically different from (2). Empirically, we found that using this probability instead of (2) leads to similar results, but at a slower speed.

B. Implementation Details

1. How many histograms do we consider?:
According to the pigeonhole principle, if we set the number of bins to \( n - 1 \), at least one bin will contain more than one number. Therefore, one could find the theoretically optimal number of bins, \( b^* \), by varying \( b \) from 1 to \( n - 1 \), searching for the minimum \( P(k) \) in Eq. (2). However, in our experiment described in Section III, we found that setting \( b > 20 \) hardly makes any difference in the final accuracy. We also empirically found that there is no need to try all integers between 1 and 20, as similar results could be obtained faster with \( b \in \{2, 3, 4, 5, 7, 9, 11, 14, 17, 20\} \).

2. How to accelerate the histogram building process:
Building multiple histograms one by one can take a long time. For efficiency, we precompute a table that matches the bin edges and the bin indices of all the histograms. This process is explained Fig. 2. In order to reuse this table on any data, it must be agnostic of the input. To this end, we normalize the input data such that its range becomes [0, 1]. This way, all edges get predetermined values between 0 and 1.
C. Summary

1) Precompute a table, as described in Fig. 2, with \( A = 0, B = 0.5, C = 1 \), etc. For the number of bins, we use \( b \in \{2, 3, 4, 5, 7, 9, 11, 14, 17, 20\} \).

2) Sort and normalize the input such that its range is between 0 and 1, i.e.,
   \[
   x_i \leftarrow \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \quad \text{for } i = 1, 2, \cdots, n.
   \]

3) For each data point, use the precomputed table to find the corresponding bin index in each histogram.

4) For each histogram, find the frequency of the tallest bin and the associated probability of randomness (Eq. (2)).

5) Find the histogram that leads to the smallest probability.

6) In that histogram, find the median of the numbers that fall inside the tallest bin.

7) Unnormalize this median. The final value is RODIAN.

In Step 4, we discard a histogram if multiple bins have the same maximum frequency. If, by any chance, all histograms are discarded, we simply take the median of the original input.

Time analysis: The time complexity of Step 2 and 6 is \( O(n \log n) \). The other steps run in either \( O(1) \) or \( O(n) \). Hence, the total time complexity is \( O(n \log n) \).

III. RESULTS

We compare RODIAN with three other methods:

1) Median,

2) Least-median-of-squares (LMedS) [32], estimated as the data point with the smallest median (squared) distance to the rest, i.e.,
   \[
   \text{LMedS} = \arg \min \med (x_i - x_j)^2,
   \]

3) Median of the tallest bin of a fixed histogram, obtained in the following way: (i) Build a histogram with a fixed number of bins, (ii) Collect all the numbers that fall inside the tallest bin, (iii) Compute their median.

We present the results on synthetic data with two different outlier distributions: a uniform distribution (Fig. 3) and a mixture of a uniform and a Gaussian distribution (Fig. 4). In both cases, RODIAN outperforms the rest in terms of robustness. Especially, Fig. 4 shows that even though we assumed a uniform outlier distribution in our derivation of RODIAN, it can still handle non-uniform outliers relatively well if \( \sigma_{\text{outlier}} \) is larger than \( \sigma_{\text{inlier}} \).

Table I compares the accuracy of RODIAN and the fixed-histogram approach for low to moderate outlier ratios. It again demonstrates the advantage of using RODIAN over a fixed histogram. In Fig. 5, we plot the mean computation times of the median, LMedS and RODIAN. We observe that RODIAN is much more scalable than the LMedS.

IV. LIMITATION

The main limitation of RODIAN is that its accuracy slightly drops when there are too few outliers (see Table I). This happens because the densest region of the inlier distribution is not always well aligned with the location of the true mean.

While this is certainly not a favorable property, the average error increase is relatively small (around 10% of the standard deviation of the inliers in Table I). We believe that this is a tolerable level in outlier-prone situations, which is the main domain we target in this work.

One potential solution is to detect when the data is outlier-free and switch to a traditional median. If the type of the inlier distribution is known (e.g., Gaussian), one can use a statistical test to check if the whole data follow the inlier distribution (e.g., normality test [33], [34]). In this work, however, we aim to make our method generalizable to any types of inlier distribution as long as it is unimodal. Discerning outlier-free data in such a general scenario is left for future work.

V. CONCLUSION

In this work, we presented RODIAN, a novel method for averaging outlier-contaminated numbers. It consists of two main steps: (1) determine a bounded region in the range that would contain mostly inliers, and (2) find the median within that region. The key idea of the first step is to assume a uniform outlier distribution and search for the region that is least likely to have occurred due to outliers. Unlike MINPRAN [1], where a similar idea was used, our method is deterministic and runs in time \( O(n \log n) \). Unlike RANSAC [31] and Huber-like loss functions [29], we do not need to tune a control parameter to adapt to different inlier error distributions. Finally, unlike the median and the LMedS [32], RODIAN can handle more than 50% outliers. An extensive evaluation demonstrates its excellent robustness, versatility and scalability.
1.34 0.38 0.30 0.42 0.36 0.30
0.26 0.28 0.30 0.32 0.36
0.42 0.28 0.30 0.38 1.34

---

**Table I: Using Fixed Histograms vs. RODIAN**

| Outliers | 5 bins | 10 bins | 20 bins | 30 bins | 50 bins | RODIAN |
|----------|--------|---------|---------|---------|---------|--------|
| 0%       | 0.54   | 0.64    | 0.69    | 0.74    | 0.80    | 0.42   |
| 10%      | 0.36   | 0.50    | 0.69    | 0.61    | 0.61    | 0.26   |
| 20%      | 0.39   | 0.51    | 0.69    | 0.63    | 0.62    | 0.28   |
| 30%      | 0.43   | 0.54    | 0.70    | 0.64    | 0.63    | 0.30   |
| 40%      | 0.48   | 0.57    | 0.71    | 0.67    | 0.66    | 0.32   |
| 50%      | 0.56   | 0.61    | 0.72    | 0.68    | 0.68    | 0.36   |

We generate 100 numbers within a range [0, 100] and average them using either a fixed histogram or RODIAN. This is repeated 10000 times. RODIAN produces the smallest mean error.

---

*Inliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ with $0 < \mu_1 < 100$.

---

*Inliers and outliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ with $0 < \mu_1, \mu_2 < 100$. (Note: This dataset is different from that of Fig. 4.)

---

We generate 100 numbers within a range [0, 100] and average them using either a fixed histogram or RODIAN. This is repeated 10000 times. RODIAN produces the smallest mean error.

---

*Inliers and outliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ with $0 < \mu_1, \mu_2 < 100$. (Note: This dataset is different from that of Fig. 4.)

---

*Inliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ with $0 < \mu_1 < 100$.

---

*Inliers and outliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ with $0 < \mu_1, \mu_2 < 100$. (Note: This dataset is different from that of Fig. 4.)

---

*Inliers follow $\mathcal{N}(\mu_1, \sigma_1^2)$ with $0 < \mu_1 < 100$.
