IcETRAN 2015

Proceedings of papers

International Conference on Electrical, Electronic and Computing Engineering

Silver Lake (Srebrno jezero), Serbia, June 8 – 11, 2015

ISBN 978-86-80509-71-6

Data Centrality Computation: Implementation and Complexity Calculation

Milica Bogićević, Milan Merkle

Abstract—**Multivariate medians can be considered as data centrality, i.e. as a data set that is placed in the center of the data cloud. Its usage is very important in distribution-free methods. Data centrality computation is very demanding even for low dimension datasets. In this paper we are presenting an algorithm and its implementation for data centrality calculation. The algorithm complexity is $O(pn^2 + n^2 \log n)$ where n is the data set size and p is the number of dimensions. Experiments show that algorithm is much faster than other implemented algorithms and it can accept thousands of multidimensional observations, since the other algorithms are tested with many two-dimensional observations or with a couple of hundreds multidimensional observations.**

Keywords—**Data centrality; Multivariate medians; Jensen's inequality; Partial order; Complexity computation**

I. Introduction

Every data analysis requires good understanding of how the data is spread over the data cloud. Sometimes it's not feasible to determine data distribution, so in that cases distribution-free methods represent the only way to describe the data.

The typical parameters used for representing the data are mean and variance, and sometimes mean are good enough to describe data centrality, but median is more robust since it's less sensitive to outliers and heavy-tailed distributions. Robustness is one of the main medians' properties. In order to illustrate how important it is, this simple example would be sufficient: it is enough to place one outlier to change the mean, but to change the median in one dimension up to one half of the data can be changed without affecting the median.

Median computation in one dimension only requires to sort the input data. If we choose quick sort algorithm it requires $O(n \log n)$ complexity. To obtain the median itself, we need to pick the element in the middle of the sorted data set and it requires $O(1)$ complexity, which means that complexity of median calculation in one dimension yields $O(n \log n)$ complexity.

In multidimensional cases, when the data set is multidimensional, the complexity is changed and it requires more resources to be obtained, so the algorithm complexity grows.

There are different methods and algorithms for calculating multidimensional median. One of the most popular methods is Tukey depth or half-space depth introduced by John Tukey [1].

In this paper we are presenting an algorithm based on Jensens' inequality in context of multivariate medians [2].

Section II deals with elementary definitions and notions of half-space depth.

Section III describes the algorithm propsed in [2].

In section IV we present algorithm implementation and its output.

Section V compares other algorithms, their complexities and performances, for those that are implemented, with the algorithm we're presenting.

II. Half-space depth definitions and notions

For multivariate data Tukey's half-space depth is one of the most popular depth functions in the literature. The Tukey median, i.e. the multivariate median associated with the half-space depth, is also a well-known measure of center for multivariate data.

There are several definitions of half-space depth and we consider some of them.

In p dimensions, the half-space location depth of a point θ relative to a data set is denoted as $ldepth(\theta; X_n)$. It is defined as smallest number of observation in any closed half-space with boundary through θ.

In the univariate setting ($p = 1$) , this definition becomes:

$$ldepth_1(\theta; X_n) = \min(\#\{x_i \le \theta\}, \#\{x_i \ge \theta\}) . \qquad (1)$$

The multivariate $ldepth$ can be seen as the smallest univariate $ldepth$ of θ relative to any projection of the data set onto a direction u , since

$$\begin{aligned} ldepth(\theta; X_n) &= \min\nolimits_{\|u\|=1} ldepth_1(u'\theta; u'X_n) \\ &= \min\nolimits_{\|u\|=1} \#\{i; u'x_i \le u'\theta\} \end{aligned} \qquad (2)$$

In words, $ldepth(\theta; X_n)$ says how deep θ lies in the data

Milica Bogićević – Faculty of Electrical Engineering, University of Belgrade, Bulevar Kralja Aleksandra 73, 11020 Belgrade, Serbia (e-mail: antomripmuk@ yahoo.com).

Milan Merkle – Faculty of Electrical Engineering, University of Belgrade, Bulevar Kralja Aleksandra 73, 11020 Belgrade, Serbia (e-mail: emerkle@etf.bg.ac.rs).

cloud [4][3].

Another definition concerns the data distribution, i.e. given a probability distribution P defined in a multidimensional space X, a depth function tries to order data in X from the center of P to the outer of P. In other words, if data is moved towards the center of data cloud, then its depth increases and if the data is moved towards the outside, then its depth decreases.

In the one dimensional case, the points are ordered by the function:

$$x \to D_1(x,P) := \min\{P(-\infty,x], P[x,\infty)\}. \quad (3)$$

If $x \in \mathbb{R}^p$ then the half-space depth of point x with respect to P, $D_T(x,P)$, is the minimal probability that can be attained in the closest half-space that contains x [6][11], i.e.

$$D_T(x;P) = \inf_H\{P(H) : H \textit{ a closed halfspace in } \mathbb{R}^d : x \in H\}. \quad (4)$$

The following definition considers half-space depth as a given set P of n points in $\mathbb{R}^d$ [7]. The half-space depth of a point $q \in \mathbb{R}^d$ is defined as:

$$\min\{|P \cap \gamma| : \textit{over all halfspaces } \gamma \textit{ containing } q\}. \quad (5)$$

For every of those definitions the same properties can be applied [3]:

- Affine invariance – depth of a point $x \in \mathbb{R}^d$ should not be dependent on underlying coordinate system.
- Vanishing at infinity – the depth of point x should approach to zero as its norm approaches infinity, i.e. $D(x,P) \to 0, \|x\| \to 0$.
- Maximality at center – for a symmetric distribution the maximum value of depth should be attained at its center.
- Monotonicity relative to the deepest point – as a point $x \in \mathbb{R}^d$ moves from a deepest point along some fixed ray, its depth should decrease monotonically.

The term “center” at third point is used to denote point of symmetry – we can say that a random vector X is *half-space symmetric* around θ if $P(X \in H) \geq \frac{1}{2}$ for every closed half-space H containing θ.

In addition to these properties, one of the most the most cited is *breakdown property* [3]. In Section I we mentioned advantage of the median relative to the mean value. That advantage describes in short breakdown property. The breakdown value is a measure of the robustness of an estimator against outlying observations. It indicates the smallest fraction of incorrect observation in the sample that causes the estimator to “break down”, or to take on values that are arbitrarily bad or meaningless. The higher the breakdown point of an estimator, the more robust it is. Half-space median has a breakdown point of at least $\frac{1}{p+1}$ in dimension p and the breakdown point can be as high as $\frac{1}{3}$. In contrast, various estimators that reject apparent outliers and afterwards calculate the mean of the remaining observation have breakdown point not larger than $\left\lceil \frac{n}{2} \right\rceil$ in dimension p.

Since the half-space median is represented as the deepest point, i.e. the point with the maximum half-space depth, half-space median is not unique point. The set of points of maximal depth is guaranteed to be a closed, bounded convex set and thanks to those three properties, half-space contains between $\left\lceil \frac{n}{p+1} \right\rceil$ and $\left\lfloor \frac{n}{2} \right\rfloor$ points [3] as consequence of Helly’s theorem. The maximal guaranteed depth in general is $\frac{n}{p+1}$. However, if we define depths as in (6), but over subfamily of halfspaces that are defined by tangents planes to a given convex cone, then the median property is preserved, that is, the maximal depth is at least $\frac{n}{2}$ [2] regardless of the shape of data cloud. The resulted median set in this case is not affine invariant in general. It is shown in [6] that for some symmetric distributions of data points the standard Tukey median has depth of $\frac{n}{2}$.

III. Algorithm for Finding Multivariate Medians Based on Intersectional Balls

Algorithm implemented in this paper arises from theorems proven in [2]. We will walk through the theorems that are the most important for this paper.

Theorem 3.1. Let $\preccurlyeq$ be a partial order in $\overline{\mathbb{R}^p}$ such that the following conditions hold:

- Any interval $[\boldsymbol{a}, \boldsymbol{b}]$ is topologically closed, and for $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{\boldsymbol{p}}$ (i.e. with finite coordinates), the interval $[\boldsymbol{a}, \boldsymbol{b}]$ is a compact set.
- For any ball $B \subset \mathbb{R}^p$, there exist $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{\boldsymbol{p}}$ such that $B \subset [\boldsymbol{a}, \boldsymbol{b}]$.
- For any set S which is bounded from above with a finite point, there exists a finite $\sup S$. For any set S which is bounded from below with a finite point, there exists a finite $\inf S$.

Let μ be a probability measure on $\mathbb{R}^p$ and let $\mathcal{J}$ be a family of intervals with respect to a partial order $\preccurlyeq$, with the property that:

$$\mu(J) > \tfrac{1}{2}, for\ each\ J \in \mathcal{J}.$$

Then the intersection off all intervals from $\mathcal{J}$ is a non-empty compact interval.

Furthermore, in [2] is shown that according to the Theorem 3.1., definition of the median is induced by partial order $\preccurlyeq$:

$$\{Med\ \mu\}_{\preccurlyeq} := \bigcap_{J=[\boldsymbol{a},\boldsymbol{b}]:\mu(J)>\frac{1}{2}} J\,.$$

Theorem 3.2. (Jensen's inequality for multivariate medians): Let f be a lower semi-continuous and quasi-convex function on $\mathbb{R}^p$, and let μ be an arbitrary probability measure on Borel sets of $\mathbb{R}^p$. Suppose that the depth function with respect to halfspaces reaches its maximum α_m on the set $C(\mu)$ (Tukey median set). Then for every $m \in C(\mu)$,

$$f(m) \leq Q_{1-\alpha_m},$$

where $Q_{1-\alpha_m}$ is the largest quantile of order $1 - \alpha_m$ for μ_f.

In other words, the following statement is proven in [2]: *In any multidimensional data set $\mathcal{S} \in \mathbb{R}^p$, Tukey median set can be obtained as an intersection of all convex sets $S \subset \mathcal{S}$ that has center at $x \in \mathcal{S}$ and that contains $\frac{2}{3}n + 1$ data points from $\mathcal{S}$.*

In the next section we will give some implementation details.

IV. Algorithm Implementation and the Output

Algorithm implementation has two major steps:

1. Construct multi-spheres. This step has complexity $O\left(n * \frac{n}{2} * p + n * (n + n\log n)\right) \sim O(pn^2 + n^2 \log n)$.
2. Intersect all multi-spheres from 1. in order to get a multivariate median. The complexity in this phase is $O(n^2)$.

The algorithm complexity is $O(pn^2 + n^2\log n + n^2) \sim O(pn^2 + n^2 \log n)$, where p is the number of dimensions and n is the number of points in the data set. Complexity is quadratic for any number of dimensions and it grows linearly when the number of dimensions increases.

All implementations and optimizations are programmed in JAVA programing language. The project contains different data distribution generators that we are using in order to verify algorithm credibility and it contains a web application for data visualization that we have described in [5]. All graphs are plotted in R.

On Figure 1. and Figure 2. are shown how n and p affect on execution times, with respect to the calculated complexity.

Figure 3. shows dependency between number of dimensions and number of points in the median dataset.

Measurements are repeated 10 times for $p \in \{1, \ldots, 10\}$ and for $n \in \begin{Bmatrix} 40, 80, 160, 320, 640, 1280, 2560, 3000, 3500, \\ 4000, 4500, 5000, 5500, 6000, 6500, 7000 \end{Bmatrix}$.

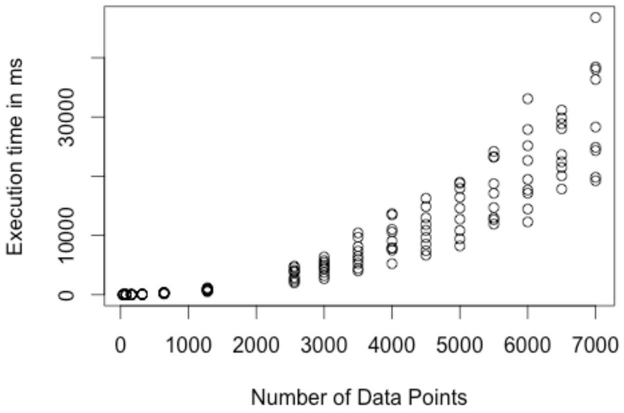

Fig. 1. When number of points increases execution time grows exponentially.

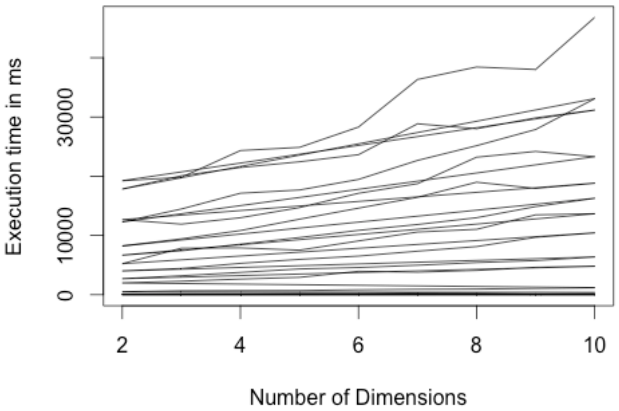

Fig. 2. When number of dimensions increases execution time grows linearly.

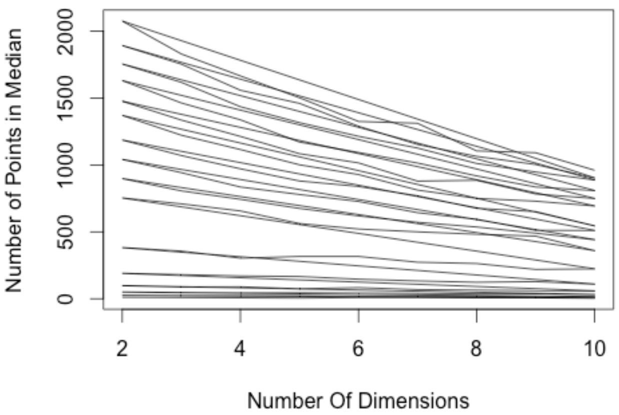

Fig. 3. When number of dimensions increases execution number of points in the median dataset linearly decreases.

Finally, Figure 4. shows the median constructed from 1000 points in 2 dimensions. Distribution is bivariate normal distribution and covariance matrix is identity matrix.

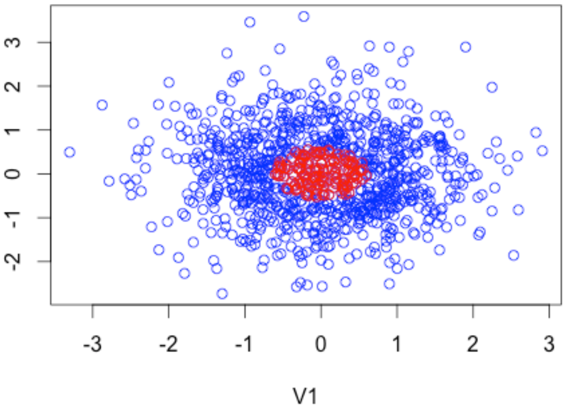

Fig. 4. Red points from the figure represents median. Blue dots represent the rest of the distribution.

Figure 5. shows multivariate normal distribution constructed from 1000 points.

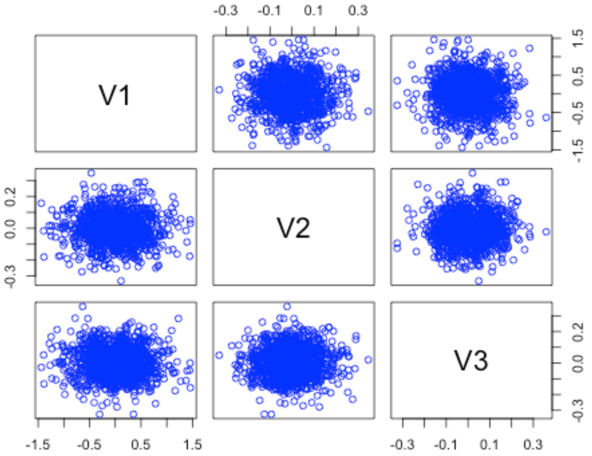

Fig. 5. Three-dimensional normal distribution.

Figure 6. shows median constructed from Figure 5. All points lie between -0.2 and 0.2. That interval is exactly the center considered in Figure 5.

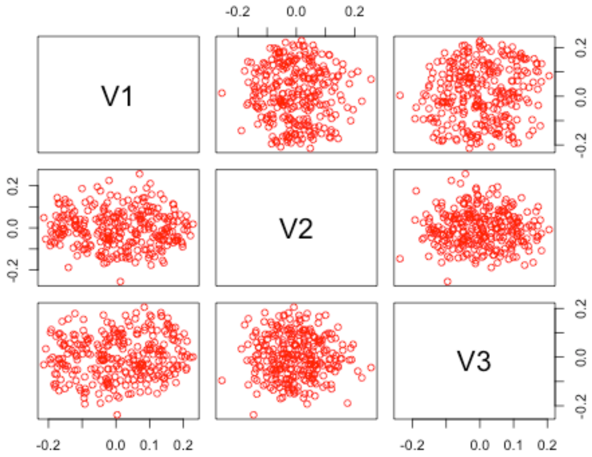

Fig. 6. Median found based on distribution represented on Figure 5.

V. Performances and Comparisions

The main performed tests are based on data generated from multivariate normal distribution. One of the tests calculates the median based on data given in [8]. Namely, it's daily simple returns of IBM stock from 1970 January 1st to 2008 December 25th. The data sample contains 9845 points and 5 dimensions.

There are a couple suggested algorithms in the last two decades. One of the first, HALFMED, was [7]. Its complexity is also quadratic, $O(n^2 \log^2 n)$, but it construct Tukey median only in two dimensions. In the sake of execution time comparison, HALFMED takes 187.86 seconds to find a median in 500 points, while algorithm proposed in this paper takes 0.246 seconds.

The second, one of the most popular, is DEEPLOC [4]. It calculates Tukey median in any dimension. Its complexity is $O(kmn \log n + kpn + mp^3 + mpn)$, where k is the number of steps taken by the program and m is the number of directions, i.e. vectors constructed by the program. Measurements in [4] go up to 1000 points and 5 dimensions and it takes 136 seconds while algorithm proposed in this paper takes 0.678 seconds.

Chan in [9] proposed an algorithm with complexity $O(n \log n)$ for $p < 3$ and $O(n^{p-1})$ for $p \geq 3$. This complexity is better for dimensions $1, 2, 3$, but for dimension 4 its complexity is worse. The algorithm from [9] is not implemented.

In [8] are proposed two algorithms implemented a couple months ago for $p \geq 3$. The first one has complexity $O(n^{p-1} \log n)$ and the second one has complexity $O(\frac{1}{2^{p-1}} n^{p-1} \log n)$. This algorithm can be better for dimension 3, but for dimension 4 it's not. For 2560 data points in 6 dimensions it takes 75.0221 seconds, while algorithm proposed in this paper takes 3.946 seconds.

Figure 7. and Figure 8. Shows data distribution and its median for IBM stock.

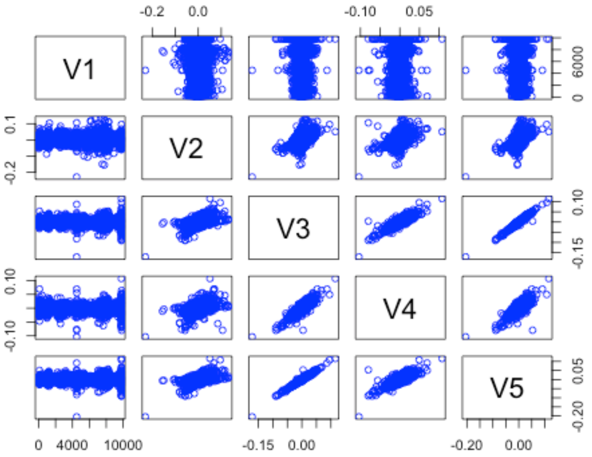

Fig. 7. IBM stock, data distribution.

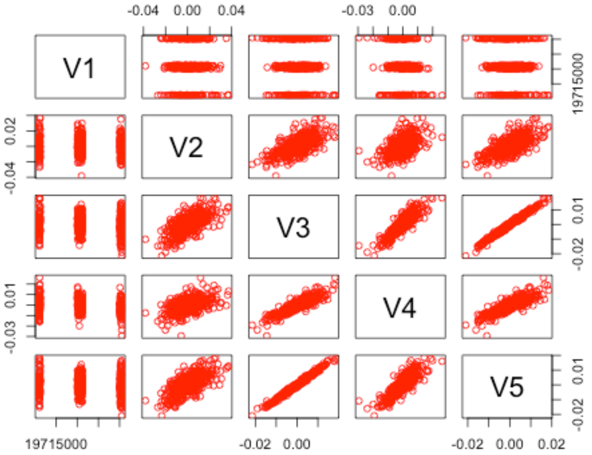

Fig. 8. IBM stock, median.

All test in his paper are tested up do 10 dimensions and 7000 data points and in the worst case it takes 46.795 seconds.

In Table I are presented computation times for higher number of data points and higher dimensions.

TABLE I
COMPUTATION TIMES (IN SECONDS)

	4	5	6	7	8	9	10
640	0.200	0.241	0.243	0.309	0.329	0.368	0.396
1280	0.695	0.704	0.830	0.913	0.946	0.106	0.117
2560	2.638	2.931	3.952	3.808	4.104	4.620	4.811
3000	3.764	4.345	4.590	4.973	5.430	5.758	6.373
3500	5.294	5.963	6.520	7.349	8.123	9.694	10.432
4000	7.901	8.223	9.077	10.580	11.037	13.480	13.676
4500	8.523	9.649	10.861	11.851	12.995	14.872	16.275
5000	10.831	12.747	14.611	16.445	18.991	19.123	19.325
5500	12.994	14.701	17.121	18.473	23.210	24.190	23.296
6000	17.129	17.676	19.462	22.667	25.153	27.888	33.086
6500	21.476	22.439	23.636	28.850	28.018	29.848	31.169
7000	24.333	24.873	28.297	36.330	38.435	38.018	46.795

VI. SUMMARY

Described algorithm is a very fast way to compute the multivariate median. It can calculate multivariate median in many dimensions in decent time, but in this paper we tested algorithm up to dimension 10. Some examples with a lower number of points and dimensions are escaped since it takes a few milliseconds.

Further optimization will take a few steps. The first one will be algorithm parallelization, so it will be even faster. Algorithm parallelization will not decrease complexity. The second phase showed in Section IV has complexity that grows linearly with increasing number of dimensions, but with some optimized data structures we are currently working on, we can achieve that number of dimensions doesn't affect on the complexity at all, so the complexity will remain the same for any number of dimensions. We are planning to integrate algorithm in web application described in [5], as well.

References

[1] J. W. Tukey (1974), "Mathematics and Picturing Data", *International Congress of Mathematicians*, Vancouver

[2] M. Merkle (2010), "Jensen's inequality for multivariate medians", J. Math. Anal. Appl., 370 258-269

[3] D. L. Donoho and M. Gasko (1992), "Breakdown properties of location estimates based on halfspace depth and projected outlyingness". *Ann. Statist.* **20** 1803-1827

[4] A. Struyf and P. J. Rousseeuw (1999), "High-dimensional computation of the deepest location," *Comp. Statist. & Data Anal.*, **34** 415-426

[5] M. Bogićević, M. Merkle (2014) "Multivariate Medians and Halfspace Depth: Algorithms and Implementation", IcETRAN conference

[6] S. Dutta, A. K. Ghosh and P. Chaudhuri (2011), "Some intriguing properties of Tukey's half-space depth," *Bernoulli* **17** 1420-1434

[7] P. J. Rousseeuw and I. Ruts (1998), "Constructing the Bivariate Tukey Median," *Statist. Sinica* **8** 827-839

[8] Xiaohui Lui (2014), "Fast Implementation of the Tukey depth"

[9] T. M. Chan, "An Optimal Randomized Algorithm for Maximum Tukey Depth", *NSERC Research Grant*

[10] S. Dutta, A. K. Ghosh and P. Chaudhuri (2011), "Some intriguing properties of Tukey's half-space depth," *Bernoulli* **17** 1420-143