Parallel computation of modified Stahel–Donoho estimators for multivariate outlier detection

WADA, Kazumi
National Statistics Center
Tokyo, JAPAN
kwada@nstac.go.jp

TSUBAKI, Hiroe
The Institute of Statistical Mathematics
Tokyo, JAPAN
tsubaki@ism.ac.jp

***Abstract*—Modified Stahel–Donoho (MSD) estimators are an orthogonally equivariant multivariate outlier detection method with a high breakdown point for all dimensions. An R function of the MSD estimators is created and its performance is confirmed; however, the method suffers from the curse of dimensionality and its implementation is limited to relatively low dimensional datasets. This paper proposes a parallel computing approach to cope with higher dimensionality and presents results for a few datasets to illustrate its use. Code for both the utilized parallelized function and the original single-core function have been placed in a public repository for further evaluation.**

Keywords—multivariate location and scatter; projection pursuit; outlier detection; Mahalanobis distance.

I. Objective

The phrase "big data" is a loosely defined term used to describe datasets so large and complex that they are awkward to work with using standard statistical software [1]. In the field of official statistics, survey and census microdata before aggregation have an usually large volume and are sometimes not easily processed by statistical software. One example is the outlier detection process to make a raw dataset clean for enumeration. Univariate outlier detection methods are commonly used; however, multivariate detection methods are not convenient for most national statistical offices, due to the computational burden. This paper discusses part of the work to explore and evaluate promising multivariate outlier detection methods for official survey enumeration.

A comprehensive research project on the editing and imputation of official statistics named EUREDIT launched in 2000. It was funded by Eurostat and involved professionals of national statistics offices and academics in the same regions. Several modern multivariate outlier detection methods were examined in the project [2]. The results suggest there is no "best" method, in the sense that no method works best in all situations [3]. Reference [4] chose the forward search (BACON-EEM: BEM), the minimization of scale (Fast-MCD), and the nearest neighbor (NNVE) algorithm among those examined in [2] and made a comparison with random number datasets following contaminated normal and long-tailed distributions. The results favored BEM, as in [2], and also suggested that nonparametric methods such as the nearest neighbor algorithm may not be desirable for elliptically distributed data; nevertheless, survey data often fit an elliptical model, after transformation, as necessary.

Statistics Canada introduced the modified Stahel–Donoho (MSD) estimators based on the projection pursuit algorithm for multivariate outlier detection of the Annual Wholesale and Retail Trade Survey (AWRTS) according to [5]. The MSD estimators are a combination of the Stahel–Donoho (SD) estimators [6] [7], and projection pursuit (PP) [8]. The estimators achieve orthogonal equivariance and sufficient robustness owing to their high breakdown point. A few improvements for MSD are also suggested by [2]. Then an R function to implement both the method proposed by [5] and the improved version of [2] was created [9]. We will refer to the former method as the Canada version and the latter method as the EUREDIT version.

The EUREDIT version of the MSD estimators and their algorithm are reviewed in the next section. The major difference with the Canada version is also briefly explained. Then the problem with the EUREDIT version for practical implementations is discussed and a parallelization of the MSD function is proposed to cope with larger datasets in section III. In section IV, we compare MSD to BEM and NNVE using simulation datasets following an asymmetrically contaminated normal distribution and then demonstrate the performance of the parallelized MSD with three different datasets. The problem of tuning the parallelized MSD is also discussed.

II. MSD Estimators

A. Methodology

The SD estimators can be regarded as robust estimators of the mean vector and the covariance matrix. Data points are projected onto randomly generated orthogonal bases. Since multivariate data are translated into one-dimensional data by projection, the problem is reduced to the estimation of one-dimensional location and scale in each projection. Observations of those $|x_i - \text{median}|/\text{MAD}$ exceed a certain threshold are regarded as possible outliers and downweighted, where MAD stands for median absolute deviation and x_i is a one-dimensional projected data point.

Let $\mathbf{X}$ be the $n \times p$ data matrix with n observations $(x_1, \cdots, x_n)$ and p variables. The superscript T designates a

matrix or vector transpose. Let μ and σ^2 be affine equivariant univariate estimators of location and scatter. The outlyingness measure r_i of each observation x_i is given by

$$r_i = \sup_{\|\nu\|=1} \frac{|\nu^{\mathrm{T}} x_i - \mu(\nu^{\mathrm{T}}\mathbf{X}^{\mathrm{T}})|}{\sigma(\nu^{\mathrm{T}}\mathbf{X}^{\mathrm{T}})}.$$

Each r_i is a measure of the maximum standardized one-dimensional deviation from the estimated location μ for all directions in $\mathbb{R}^p$. Then the weights are computed as $w_i = w(r_i)$, where $w:\mathbb{R}^+ \to \mathbb{R}^+$ is a weight function, and the SD estimators are defined as

$$m_{SD} = \frac{\sum_{i=1}^n w_i x_i}{\sum_{i=1}^n w_i} \text{ and } S_{SD} = \frac{\sum_{i=1}^n w_i (x_i - m_{SD})(x_i - m_{SD})^{\mathrm{T}}}{\sum_{i=1}^n w_i}.$$

The finite sample breakdown point was studied when r_i is taken from a random subset of size N of all $\nu \in \mathbb{R}^p$ with $\|\nu\| = 1$ and resulted in the following size N to maintain a high breakdown point and the weight function w [10]:

$$N = \text{truncate}(\exp(2.1328 + 0.8023 \times p)/p), \quad (1)$$

$$w:\mathbb{R}^+ \to \mathbb{R}^+,\ r \mapsto w(r) = \begin{cases} 1 & \text{if } r \le c \\ (c/r)^2 & \text{if } r > c \end{cases} \quad \text{with } c = \sqrt{\chi^2_{p,0.95}}.$$

See [2] and [10] for further details. Table I shows how the size *N* increases along with *p*.

The MSD estimators proposed by [5] use the SD estimators as the initial robust mean vector and covariance matrix. These are used for the following principal component analysis of PP, which regards the principal components as "interesting" directions to find outliers. Then the Mahalanobis distance is computed from the final mean vector and covariance matrix to detect outliers.

B. Algorithm

The following algorithm is the EUREDIT version, i.e., the algorithm based on [5]. It also incorporates a few improvements of [2].

1) Let N be the number of orthogonal basis required as described in the previous subsection. Repeat the following three steps N times.

a) Obtain the necessary amount of random numbers and create p-dimensional unit vectors.

b) Make the unit vectors an orthogonal basis $\nu_j,\ j = 1, \cdots, p$. *Project the data onto* ν_j. *Compute residuals* $r_{ij}^{(1)}$ *and trimmed residuals* $\tilde{r}_{ij}^{(1)}$ *as follows*:

$$r_{ij}^{(1)} = \frac{|\nu_j^{\mathrm{T}} x_i - \text{med}(\nu_j^{\mathrm{T}} x)|}{\text{mad}(\nu_j^{\mathrm{T}} x)/0.674},$$

$$\tilde{r}_{ij}^{(1)} = \begin{cases} r_{ij}^{(1)} & \text{if } 0 \le r_{ij}^{(1)} \le c \\ c^2 / r_{ij}^{(1)} & \text{if } c \le r_{ij}^{(1)} \end{cases} \quad \left(c = \sqrt{\chi^2_{p,0.95}}\right).$$

*Please note that "*med*" means median and "*mad*" means median absolute deviation.*

c) Compute a set of weights $w_i^{(1)}$:

$$w_i^{(1)} = \prod_{j=1}^p \frac{\tilde{r}_{ij}^{(1)}}{r_{ij}^{(1)}} = \prod_{j=1}^p w_{ij}^{(1)}. \quad (2)$$

2) After iterating N times, there are N sets of weights $w_i^{(1)}$. *Choose one smallest weight for each data point to configure a set of the initial weights.*

3) Robust principal component analysis of the initial mean vector and the covariance matrix.

a) Compute the initial mean vector $\hat{u}_1$ *and the covariance matrix* $\hat{V}_1$ *as follows*:

$$\begin{aligned} &\hat{u}_1 = \textstyle\sum_i^n w_i^{(1)} x_i / \sum_i^n w_i^{(1)}, \\ &\hat{V}_1 = \textstyle\sum_{i=1}^n (x_i - \hat{u}_1)(x_i - \hat{u}_1)^{\mathrm{T}} (w_i^{(1)})^2 / \sum_{i=1}^n (w_i^{(1)})^2. \end{aligned} \quad (3)$$

b) Let $\hat{b}_1, \cdots, \hat{b}_p$ *be the eigenvectors of* $\hat{V}_1$. *Project the data onto the eigenvectors* $r_{ij}^{(2)}$ *and compute* $\tilde{r}_{ij}^{(2)}$ *as follows*:

$$r_{ij}^{(2)} = \frac{|b_j^{\mathrm{T}} x_i - \text{med}(b_j^{\mathrm{T}} x)|}{\text{mad}(b_j^{\mathrm{T}} x)/0.674},$$

$$\tilde{r}_{ij}^{(2)} = \begin{cases} r_{ij}^{(2)} & \text{if } 0 \le r_{ij}^{(2)} \le c \\ c^2 / r_{ij}^{(2)} & \text{if } c \le r_{ij}^{(2)} \end{cases} \quad \left(c = \sqrt{\chi^2_{p,0.95}}\right).$$

c) Compute the secondary weights $w_i^{(2)}$ *from* $r_{ij}^{(2)}$ *and* $\tilde{r}_{ij}^{(2)}$ *according to (1).*

d) Compare the initial and the secondary weights and choose the smaller weight for each data point to configure a set of the final weight w_i.

4) Compute the final mean vector $\hat{u}_2$ *and the covariance matrix* $\hat{V}_2$ *according to (2).*

5) Compute robust Mahalanobis squared distance $D(x_i)^2$ *of each data point as follows using* $\hat{u}_2$ *and* $\hat{V}_2$:

$$D(x_i)^2 = (x_i - \hat{u}_2)^{\mathrm{T}} \hat{V}_2^{-1} (x_i - \hat{u}_2).$$

6) Mahalanobis squared distance follows the F distribution with p and n-p degrees of freedom. Compute the test statistic F_i *as follows*:

$$F_i = \frac{(n-p)n}{(n^2-1)p} D(x_i)^2.$$

Any data point with greater than the corresponding F value for the 99.9th percentile is recognized as an outlier. The percentile figure is based on [5].

III. Implementation and Test Environment

An R function of the MSD estimators is developed [9] to compare the Canada and EUREDIT versions by Monte-Carlo simulation with the asymmetrically contaminated datasets based on [11] following the normal and long-tailed distributions. Both versions performed well; however, the EUREDIT version was superior owing to the increased number of orthogonal bases regarding more difficult datasets for outlier detection method evaluation, such as the Bushfire dataset [12].

A. Original MSD Function

Among the few differences between the Canada and EUREDIT versions, the most influential one is the number of orthogonal bases per dimension to obtain the initial SD estimators. The number increases exponentially in the EUREDIT version, as shown in Table I, whereas the Canada version always needs 10 bases per dimension. In compensation for better performance in outlier detection, the large number of bases may lead to running out of memory and/or an excessively long time to compute, and so the EUREDIT version is computationally more burdensome than the Canada version.

TABLE I. MEMORY USED FOR ORTHOGONAL BASES

Number of variables (p)	Number of bases (M)	Number of elements for bases	Megabytes (MB) [a]
2	21	84	1
3	31	279	2
4	52	832	7
5	93	2,325	19
8	5,172	331,008	265
10	25,740	2,574,000	2,059
15	94,774	21,324,150	170,593
20	3,925,749	1,570,299,600	12,562,397

a. Calculated as one element of bases equals 8 bytes.

BEM is fairly fast compared to Fast-MCD [2]; however, in contrast, MSD becomes much slower than either of these methods as the dimension p increases. Since its computational time is the most serious problem of MSD in its practical use for survey enumeration, the priority is placed on speed in developing the R function. The MSD function processes N orthogonal bases in Step 1 simultaneously without looping N times in order to make the most of the R ability to operate on vectors, matrices, and arrays. In return for the maximization of the processing speed, the function requires a large array of size $N \times p \times p$ for the bases in Step 1-a, and then the projection residuals $r_{ij}^{(1)}$, trimmed residuals $\tilde{r}_{ij}^{(1)}$, and the corresponding weights $w_{ij}^{(1)}$ required in Steps 1-b and c each have sizes of $N \times n \times p$. The function restricts the number of variables (and observations) processed, since R keeps all the data in RAM. Since all the arrays of data described above result in a set of weights $w_i^{(1)}$ of the $n \times p$ matrix at the end of Step 1-c, Step 1 is the memory bottleneck.

R works on a single core even on a PC with a multi-core processor. Therefore, the original MSD function works on a single core, too, and we consider parallelizing Step 1 to extend its capacity while sacrificing speed as little as possible.

B. Parallelization

The CRAN packages "*foreach*" and "*doParallel*" are used, and "*doParallel*" depends on "*iterators*" and "*parallel*". Table II shows the specifications of the PCs used and the software versions. The package "*doParallel*" uses "*snow*" functionality on Windows and it enables execution on a cluster [13].

In the parallelized function, we divided an array of bases into smaller pieces and processed them separated in the different cores of a PC. Only the smallest set of weights $w_i^{(1)}$ in each child process is returned to the master process. The divided pieces have to be within some size limit, since the parallelized processes share the same memory. On the other hand, making the pieces smaller increases the number of loops and requires more computational time. The new parallelized MSD function has a parameter "dv" to set the maximum number of elements processed together on the same core. Finding a suitable setting of *dv* is also treated in this section.

TABLE II. PC SPECIFICATIONS AND SOFTWARE VERSIONS

Product name		EPSON Pro4700
CPU		Intel® Core™ i5
	Max clock speed	3.33 GHz
	Number of cores	4
RAM		4.000 GB
OS installed		Windows 7 Professional (32 bit)
Available memory on R		1.535 GB
R		ver. 3.0.0
CRAN Packages	***Foreach***	ver. 1.4.0
	doParallel	ver. 1.0.1
	Iterators	ver. 1.0.6
	Parallel	ver. 3.0.0

IV. Evaluation and Tuning

The numerical calculation conducted by the parallelized function is the same as that of the single-core function. However, please note that the outcome of the functions executed over time with a given dataset may not necessarily be the same unless the same random seed is specified, since the initial SD estimators are computed based on projection onto orthogonal bases made from pseudo-random numbers.

A. Evaluation with Asymmetrically Contaminated Datasets

The simulation datasets used here consist of random variables following a multivariate normal distribution with asymmetric contamination. This contamination model is of the form $(1-\alpha)N_p(0,\mathbf{R})+\alpha N_p(\delta e_1,\lambda\mathbf{I})$ and is particularly difficult to analyze for many outlier detection procedures [11]. The first and second terms represent normal data and outliers, respectively. In this formula, α is the fraction of contamination (the rate of outliers); p is the number of variables; $\mathbf{R}$ is the correlation matrix in which all the correlations between variables have the same value r; δ stands for the distance between normal data and outliers; e_1 is the first unit vector; and λ is the variance of the outliers.

The actual values of the parameters used and the results of the comparison with BEM and NNVE are shown in Table III. The MSD function tends to have fewer false positives and its accuracy improves along with correlation between variables. The MSD function also showed better performance than BEM for outliers with variances larger than that of the normal data; however, the MSD function may not be suitable for datasets with any outlier clusters of smaller variance. Further simulation is necessary with long-tailed datasets, since [4] showed that BEM does not have good performance with such datasets.

As the number of variables increases, there is no obvious sign that the performance of the MSD function drops; therefore, the choice of N according to (1) proposed by [10] seems sufficient.

TABLE III. SIMULATION RESULTS OF NNVE, BEM AND MSD

SD of outliers $\sqrt{\lambda}$	Distance δ	Variables p	Outliers α	Correlation between variables 0									Correlation between variables 0.8								
				NNVE			BEM			MSD			NNVE			BEM			MSD		
				False positives	Leakage	Total rate	False positives	Leakage	Total rate	False positives	Leakage	Total rate	False positives	Leakage	Total rate	False positives	Leakage	Total rate	False positives	Leakage	Total rate
–	0	10	0	34.00%	–	66%	1.00%	–	99%	0.00%	–	100%	12.00%	–	88%	0.00%	–	100%	0.00%	–	100%
				25.00%	–	75%	0.00%	–	100%	0.00%	–	100%	10.00%	–	90%	0.00%	–	100%	0.00%	–	100%
		20	0	50.00%	–	50%	0.00%	–	100%	0.00%	–	100%	9.00%	–	91%	0.00%	–	100%	0.00%	–	100%
				8.00%	–	92%	0.00%	–	100%	0.00%	–	100%	9.00%	–	91%	0.00%	–	100%	0.00%	–	100%
0.1	10	10	0.1	5.56%	0.00%	95%	0.00%	0.00%	100%	0.00%	0.00%	100%	7.78%	0.00%	93%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	100.00%	80%	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	100.00%	100.00%	0%	1.43%	0.00%	99%	24.29%	100.00%	53%	14.29%	100.00%	60%	0.00%	0.00%	100%	1.43%	0.00%	99%
			0.4	100.00%	100.00%	0%	10.00%	0.00%	94%	100.00%	100.00%	0%	25.00%	100.00%	45%	1.67%	0.00%	99%	18.33%	100.00%	49%
			0.5	100.00%	100.00%	0%	100.00%	96.00%	2%	100.00%	100.00%	0%	42.00%	100.00%	29%	100.00%	96.00%	2%	100.00%	100.00%	0%
		20	0.1	22.22%	0.00%	80%	1.11%	0.00%	99%	0.00%	100.00%	90%	10.00%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	100.00%	80%	10.00%	100.00%	72%	0.00%	0.00%	100%	0.00%	100.00%	80%
			0.3	100.00%	100.00%	0%	1.43%	0.00%	99%	12.86%	100.00%	61%	12.86%	100.00%	61%	0.00%	0.00%	100%	0.00%	100.00%	70%
			0.4	100.00%	100.00%	0%	0.00%	0.00%	100%	98.33%	100.00%	1%	50.00%	100.00%	30%	0.00%	100.00%	60%	38.33%	100.00%	37%
			0.5	100.00%	100.00%	0%	80.00%	100.00%	10%	100.00%	100.00%	0%	16.00%	100.00%	42%	80.00%	100.00%	10%	100.00%	100.00%	0%
	100	10	0.1	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.11%	0.00%	99%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	0.00%	100%	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	100.00%	100.00%	0%	1.43%	0.00%	99%	1.43%	0.00%	99%	8.57%	100.00%	64%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	100.00%	100.00%	0%	10.00%	0.00%	94%	13.33%	0.00%	92%	26.67%	100.00%	44%	1.67%	0.00%	99%	5.00%	0.00%	97%
			0.5	100.00%	100.00%	0%	100.00%	100.00%	0%	100.00%	100.00%	0%	40.00%	100.00%	30%	100.00%	100.00%	0%	100.00%	100.00%	0%
		20	0.1	1.11%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%	3.33%	0.00%	97%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	0.00%	100%	5.00%	100.00%	76%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	100.00%	100.00%	0%	1.43%	0.00%	99%	0.00%	0.00%	100%	12.86%	100.00%	61%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	100.00%	100.00%	0%	0.00%	0.00%	100%	0.00%	0.00%	100%	26.67%	100.00%	44%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.5	100.00%	98.00%	1%	80.00%	100.00%	10%	100.00%	100.00%	0%	22.00%	100.00%	39%	80.00%	100.00%	10%	100.00%	100.00%	0%
1	10	10	0.1	4.44%	0.00%	96%	0.00%	0.00%	100%	0.00%	0.00%	100%	7.78%	0.00%	93%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	5.00%	20.00%	92%	0.00%	0.00%	100%	0.00%	65.00%	87%	7.50%	0.00%	94%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	20.00%	3.33%	85%	1.43%	0.00%	99%	0.00%	100.00%	70%	12.86%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	31.67%	42.50%	64%	10.00%	0.00%	94%	0.00%	100.00%	60%	16.67%	0.00%	90%	1.67%	0.00%	99%	0.00%	60.00%	76%
			0.5	28.00%	88.00%	42%	0.00%	100.00%	50%	0.00%	100.00%	50%	8.00%	78.00%	57%	0.00%	92.00%	54%	0.00%	94.00%	53%
		20	0.1	7.78%	0.00%	93%	1.11%	0.00%	99%	0.00%	100.00%	90%	8.89%	0.00%	92%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	28.75%	5.00%	76%	0.00%	0.00%	100%	0.00%	100.00%	80%	10.00%	0.00%	92%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	1.43%	76.67%	76%	1.43%	0.00%	99%	0.00%	100.00%	70%	12.86%	0.00%	91%	0.00%	0.00%	100%	0.00%	60.00%	82%
			0.4	8.33%	85.00%	61%	0.00%	0.00%	100%	0.00%	100.00%	60%	15.00%	0.00%	91%	0.00%	5.00%	98%	0.00%	97.50%	61%
			0.5	2.00%	90.00%	54%	0.00%	100.00%	50%	0.00%	100.00%	50%	6.00%	76.00%	59%	0.00%	82.00%	59%	0.00%	100.00%	50%
	100	10	0.1	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.11%	0.00%	99%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	13.75%	10.00%	87%	0.00%	0.00%	100%	0.00%	0.00%	100%	7.50%	0.00%	94%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	30.00%	6.67%	77%	1.43%	0.00%	99%	0.00%	0.00%	100%	12.86%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	100.00%	60%	10.00%	0.00%	94%	0.00%	0.00%	100%	15.00%	0.00%	91%	1.67%	0.00%	99%	1.67%	0.00%	99%
			0.5	50.00%	62.00%	44%	0.00%	100.00%	50%	0.00%	100.00%	50%	10.00%	40.00%	75%	0.00%	100.00%	50%	0.00%	100.00%	50%
		20	0.1	1.11%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.11%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	0.00%	95.00%	81%	0.00%	0.00%	100%	0.00%	0.00%	100%	11.25%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	34.29%	6.67%	74%	1.43%	0.00%	99%	0.00%	0.00%	100%	12.86%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	100.00%	60%	0.00%	0.00%	100%	0.00%	0.00%	100%	15.00%	0.00%	91%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.5	16.00%	86.00%	49%	0.00%	100.00%	50%	0.00%	100.00%	50%	6.00%	86.00%	54%	0.00%	92.00%	54%	0.00%	100.00%	50%
5	10	10	0.1	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	7.78%	0.00%	93%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.25%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	0.00%	0.00%	100%	1.43%	0.00%	99%	0.00%	0.00%	100%	4.29%	0.00%	97%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	0.00%	100%	10.00%	0.00%	94%	0.00%	2.50%	99%	5.00%	0.00%	97%	1.67%	0.00%	99%	0.00%	0.00%	100%
			0.5	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	6.00%	97%	10.00%	0.00%	95%	0.00%	0.00%	100%	0.00%	0.00%	100%
		20	0.1	1.11%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%	4.44%	0.00%	96%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	5.00%	0.00%	96%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	0.00%	0.00%	100%	1.43%	0.00%	99%	0.00%	0.00%	100%	1.43%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	3.33%	0.00%	98%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.5	0.00%	0.00%	100%	0.00%	20.00%	90%	0.00%	2.00%	99%	2.00%	0.00%	99%	0.00%	20.00%	90%	0.00%	0.00%	100%
	100	10	0.1	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	2.22%	0.00%	98%	1.11%	0.00%	99%	1.11%	0.00%	99%
			0.2	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.25%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	0.00%	0.00%	100%	1.43%	0.00%	99%	0.00%	0.00%	100%	4.29%	0.00%	97%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	0.00%	100%	10.00%	0.00%	94%	0.00%	0.00%	100%	5.00%	0.00%	97%	1.67%	0.00%	99%	0.00%	0.00%	100%
			0.5	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	12.00%	94%	10.00%	0.00%	95%	0.00%	0.00%	100%	0.00%	0.00%	100%
		20	0.1	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	1.11%	0.00%	99%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.2	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	2.50%	0.00%	98%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.3	0.00%	0.00%	100%	1.43%	0.00%	99%	0.00%	0.00%	100%	5.71%	0.00%	96%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.4	0.00%	0.00%	100%	0.00%	0.00%	100%	0.00%	0.00%	100%	3.33%	0.00%	98%	0.00%	0.00%	100%	0.00%	0.00%	100%
			0.5	0.00%	0.00%	100%	0.00%	20.00%	90%	0.00%	22.00%	89%	6.00%	0.00%	97%	0.00%	20.00%	90%	0.00%	0.00%	100%

B. *Examples with Various Datasets*

The results of three different datasets are shown here in order to illustrate the performance of the parallelized MSD function.

The first example is the Bushfire dataset to locate bushfire scars. It has 5 variables and 38 observations, and is widely used to evaluate multivariate outlier detection methods (e.g., [10]). The results are shown as Fig. 1. Outliers detected are shown in red. The number of outliers detected is 12 or 13, which depends on the random seed.

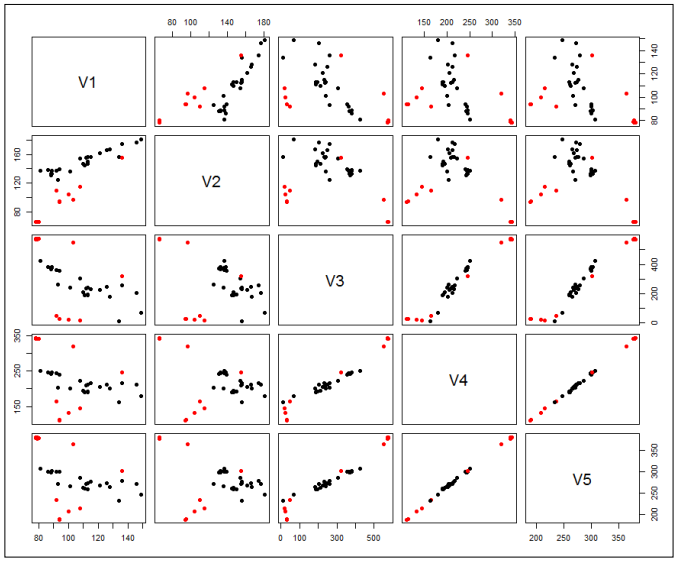

Fig. 1. Bushfire Data

The next example is from the simulation datasets used in subsection A. The scatterplot matrix is shown as Fig. 2. The dataset has 100 observations, 20 variables with no correlation, contamination α=0.1, distance δ=10, and variance of outliers λ=5. The parallelized function worked with *dv*=100,000, which means 250 sets of orthogonal bases were processed simultaneously and the correct outliers were detected. The computational time slightly exceeds 2 hours. The single-core function can process at most 11 variables with 100 observations.

The last example is Synthetic Microdata, provided by the National Statistics Center (NSTAC) of Japan. Since the use of microdata of official statistical surveys is restricted in order to protect respondents' privacy, the dataset is not of the real data, but rather is generated from random numbers; however, the data has similar distributions and relations between variables to the real survey data. Synthetic Microdata is based on the descriptive statistics on the 2004 National Survey of Family Income and Expenditure, which concerns workers' households having at least two persons. We extracted 10 numerical variables, removed observations less than or equal to zero, and carried out logarithmic transformation so that the data cloud would have an elliptical shape. The dataset contains 20,336 observations. The results are shown as Fig. 3 and the fundamental statistics of both before and after the detection are listed in Table IV (below Fig. 3). As shown in the figure, the function effectively removes the tail of the distribution. The single-core function can process at most 5 variables with 20,336 observations.

C. *Tuning*

The parallelized function takes over 9 seconds for Synthetic Data of 5 variables, although the single-core function takes only about 6 seconds to process the same dataset. This is because the parallelized code needs to call packages, carrying out iteration, and conduct inter-core communication, which the single-core codes does not need to do; however, the efficiency is expected to increase along with a number of variables *p*. The parallelized function takes about 55 seconds with 8 variables and 150 seconds with 10 variables. Various settings of *dv* are also tested to find an appropriate size of bases processed simultaneously for the case of Synthetic Microdata.

The results of the turning are shown in Tables V, VI, and VII. The parameter *dv* determines the maximum number of elements for orthogonal bases, the number of bases processed simultaneously, and the total number of iterations. Any inappropriate setting of *dv* causes an out-of-memory problem and a better setting improves processing speed.

Although the tuning tests executed here are limited to a PC with multi-core processes as the test environment, we expect that the parallelized code could be easily extended to a cluster environment, considering the characteristics of the package "*doParallel*" [14].

TABLE V. DATASET WITH 5 VARIABLES

dv[c]	Total iterations[d]	Chunk size[e]	System Time		
			User	*System*	*Elapsed*
300-500	8	12	0.24	0.01	6.81
600-8000	4	24	0.26	0.02	7.69

c. Maximum number of elements processed simultaneously on one core.

d. Total number of iterations on all cores.

e. Number of bases processed simultaneously on one core.

TABLE VI. DATASET WITH 8 VARIABLES

dv	Total iterations	Chunk size	System Time		
			User	*System*	*Elapsed*
300	164	4	0.58	0.07	38.16
400	108	6	0.43	0.06	36.34
500	96	7	0.33	0.03	36.63
600	72	9	0.39	0.05	35.73
700	68	10	0.39	0.04	36.58
800	56	12	0.45	0.06	35.68
900	48	14	0.35	0.03	38.08
1000	44	15	0.39	0.03	38.33
2000	24	27	0.34	0.03	43.54
3000	16	41	0.26	0.00	47.51
5000	12	54	0.39	0.06	80.47
8000	8	81	0.35	0.16	118.89

TABLE VII. DATASET WITH 10 VARIABLES

dv	Total iterations	Chunk size	System Time		
			User	*System*	*Elapsed*
300	860	3	1.66	0.44	164.38
400	644	4	1.40	0.27	157.34
500	516	5	1.20	0.31	152.21
600	432	6	0.74	0.20	147.34
700	368	7	0.82	0.15	145.53
800	324	8	0.81	0.17	143.35
900	288	9	0.71	0.21	144.32
1000	260	10	0.55	0.13	140.23
1200	216	12	0.67	0.12	144.17
1500	172	15	0.50	0.06	153.33
2000	132	20	0.47	0.07	168.95
3000	88	30	0.53	0.13	204.21
5000	52	50	0.78	0.25	527.03
8000	36	72	N.A.	N.A.	N.A.

V. Conclusion

In this paper, we presented a parallel algorithm for the MSD estimators for multivariate outlier detection, along with a few examples. The parallelized R function is able to cope with a bigger dataset and is applicable not only for the practical use of official survey statistics enumeration but also for general statistical modeling of big multivariate data in business or scientific research.

All the R packages used and R itself are distributed by the Comprehensive R Archive Network (CRAN: http://cran.r-project.org), including the NNVE function "cov.nnve" of library "covRobust". The parallelized R function code developed and tested in this paper has been uploaded to a public repository, https://github.com/kazwd2008/MSD.parallel/. The original single-core function published in [9] is also made available at https://github.com/kazwd2008/MSD/. The BEM function for R has been ported by Masato Okamoto of the Ministry of Internal Affairs and Communications from the original code for S-plus by Cedric Béguin published in [2]. The details of the modifications have been documented and made available at https://github.com/kazwd2008/BEM/.

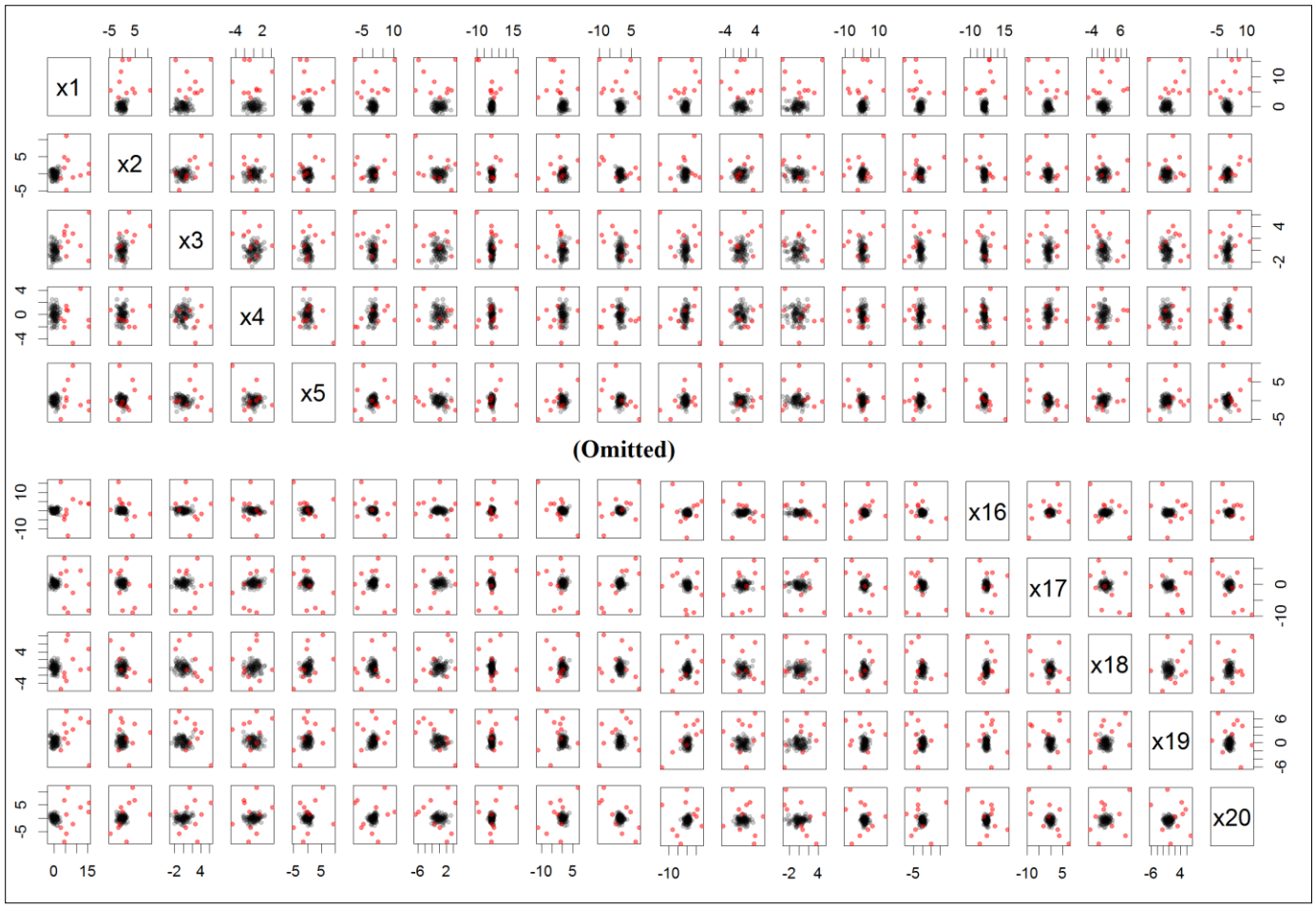

Fig. 2. Outliers in Simulation Data

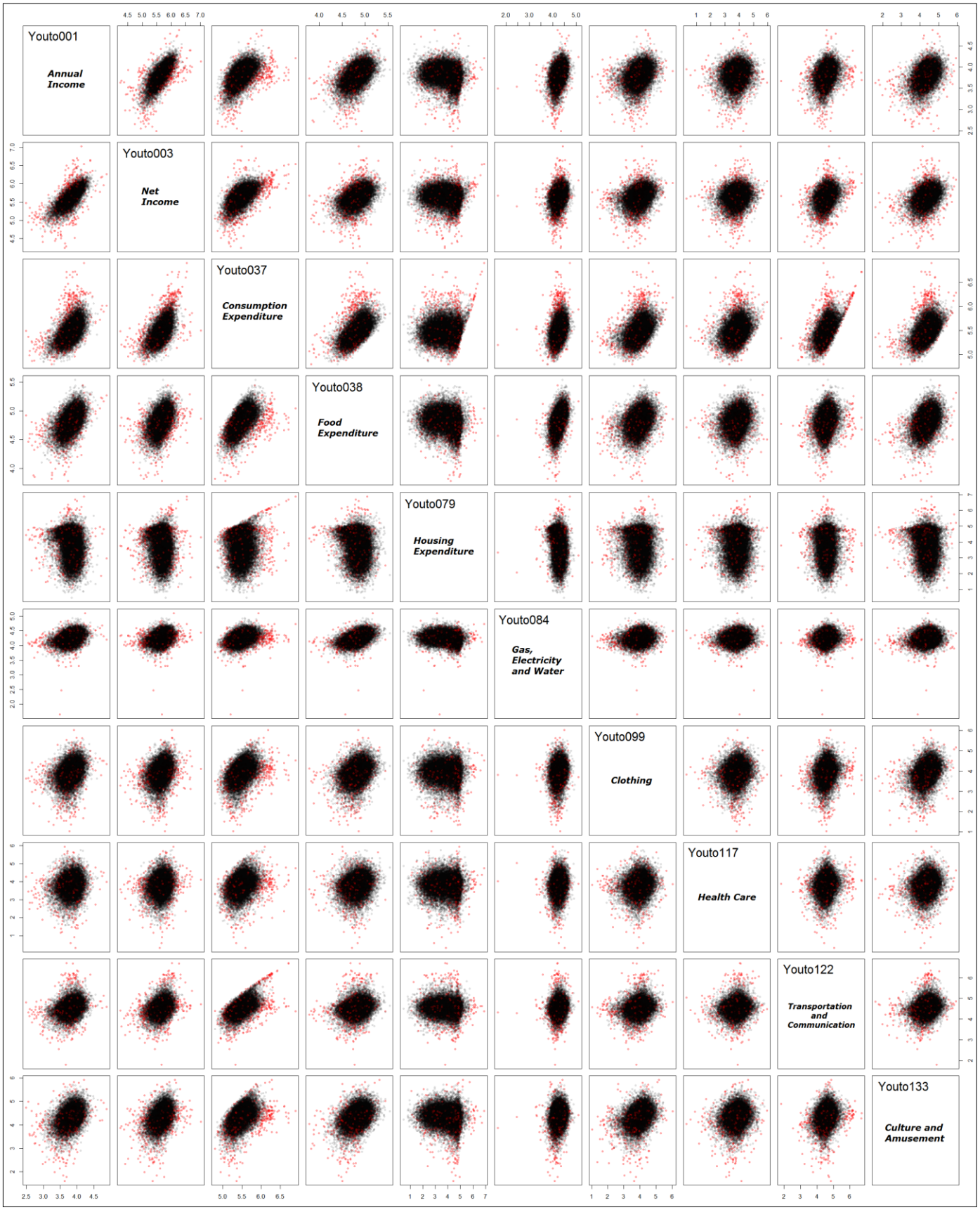

Fig. 3. Outliers in Synthetic Microdata

TABLE IV. DEFFERENCE OF FUNDAMENTAL STATISTICS AFTER OUTLIER DETECTION [SYNTHETIC MICRODATA, 10 VARIABLES]

Statistics	Annual Income		Net Income		Consumption Expenditure		Food Expenditure		Housing Expenditure	
	Original	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*
Min.	2.490	2.885	4.253	4.735	4.794	4.796	3.776	4.084	0.479	0.479
Q1	3.683	3.685	5.529	5.531	5.353	5.354	4.717	4.719	3.177	3.179
Median	3.818	3.819	5.656	5.656	5.467	5.466	4.834	4.835	3.945	3.938
Mean	3.807	3.809	5.646	5.647	5.477	5.473	4.822	4.825	3.815	3.812
Q3	3.947	3.947	5.778	5.776	5.587	5.583	4.938	4.939	4.568	4.562
Max.	4.879	4.615	7.024	6.431	6.894	6.285	5.539	5.539	6.875	5.877
SD[f]	0.2068	0.2067	0.1896	0.1723	0.8697	0.1677	0.4454	0.4776	0.3212	0.3640
Statistics	**Gas, Electricity and Water**		**Clothing**		**Health Care**		**Transportation and Communication**		**Culture and Amusement**	
	Original	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*	*Original*	*Cleaned*
Min.	1.651	3.503	1.039	1.938	0.321	1.568	1.807	3.102	1.597	2.635
Q1	4.137	4.138	3.697	3.702	3.552	3.556	4.367	4.370	4.126	4.132
Median	4.245	4.246	3.970	3.972	3.855	3.858	4.554	4.555	4.357	4.359
Mean	4.245	4.247	3.943	3.949	3.819	3.823	4.549	4.549	4.341	4.347
Q3	4.355	4.355	4.228	4.229	4.117	4.116	4.735	4.734	4.576	4.577
Max.	5.100	4.937	6.033	5.553	5.955	5.766	6.701	6.030	5.920	5.792
SD	0.2004	0.1962	0.1793	0.1688	0.8637	0.1637	0.4320	0.4659	0.3057	0.3516

f. SD stands for standard deviation.

References

[1] C. Snijders, U. Matzat, and U.-D. Reips, "'Big Data': big gaps of knowledge in the field of Internet science," International Journal of Internet Science, vol. 7, pp. 1-5, 2012.

[2] C. Béguin and B. Hulliger, "Robust multivariate outlier detection and imputation with incomplete survey data", EUREDIT Deliverable D4/5.2.1/2 Part C, 2003.

[3] G. Barcaroli, "The EUREDIT project: activities and results," Rivista di statistica ufficiale, issue 2, pp. 101-135, 2002.

[4] K. Wada, "Comparison of multivariate outlier detection methods," Proceedings of the 2004 Japanese Joint Statistical Meeting, pp. 95-96, 2004 (in Japanese).

[5] S. Franklin and M. Brodeur, "A practical application of a robust multivariate outlier detection method," Proceedings of the Survey Research Methods Section, the American Statistical Association, pp. 186-191, 1997.

[6] W. A. Stahel, "Breakdown of covariance estimators," Research Report 31, Fachgruppe für Statistik, E.T.H. Zürich, 1981.

[7] D. L. Donoho, "Breakdown properties of multivariate location estimators", Ph.D. qualifying paper, Harvard University, 1982.

[8] Z. Patak, "Robust principal component analysis via projection pursuit," M. Sc. Thesis, University of British Columbia, Canada, 1990.

[9] K. Wada, "Detection of multivariate outliers: modified Stahel-Donoho estimators," Research Memoir of Official Statistics, Statistical Research and Training Institute, no. 67, pp. 89-157, 2010, available at http://www.stat.go.jp/training/2kenkyu/pdf/ihou/67/wada1.pdf (in Japanese).

[10] R. A. Maronna and V. J. Yohai, "The behavior of the Stahel-Donoho robust multivariate estimator," Journal of the American Statistical Association, vol. 90, pp. 330-341, 1995.

[11] D. Peña and F. J. Prieto, "Multivariate outlier detection and robust covariance matrix estimation," Technometrics, vol. 43, pp. 286-300, 2001.

[12] N. A. Campbell, "Bushfire mapping using NOAA AVHRR data," Technical report, CSIRO, 1989.

[13] S. Weston and R. Calaway, "Getting started with doParallel and foreach," 2010, available at http://cran.r-project.org/web/packages/doParallel/vignettes/gettingstartedParallel.pdf.

[14] Revolution Analytics. "doParallel: Foreach parallel adaptor for the parallel package," R package version 1.0.1. 2012, available at http ://CRAN.R-project.org/package=doParallel.