A Deterministic Algorithm for Robust Location and Scatter

Mia Hubert , Peter J. Rousseeuw & Tim Verdonck

A Deterministic Algorithm for Robust Location and Scatter

Mia HUBERT, Peter J. ROUSSEEUW, and Tim VERDONCK

Most algorithms for highly robust estimators of multivariate location and scatter start by drawing a large number of random subsets. For instance, the FASTMCD algorithm of Rousseeuw and Van Driessen starts in this way, and then takes so-called concentration steps to obtain a more accurate approximation to the MCD. The FASTMCD algorithm is affine equivariant but not permutation invariant. In this article, we present a deterministic algorithm, denoted as DetMCD, which does not use random subsets and is even faster. It computes a small number of deterministic initial estimators, followed by concentration steps. DetMCD is permutation invariant and very close to affine equivariant. We compare it to FASTMCD and to the OGK estimator of Maronna and Zamar. We also illustrate it on real and simulated datasets, with applications involving principal component analysis, classification, and time series analysis. Supplemental material (Matlab code of the DetMCD algorithm and the datasets) is available online.

Key Words: Affine equivariance; Covariance; Multivariate; Outliers; Robustness.

1. INTRODUCTION

The minimum covariance determinant (MCD) method (Rousseeuw 1984) is a highly robust estimator of multivariate location and scatter. Given an $n \times p$ data matrix $X = (x_1, \ldots, x_n)^T$ with $x_i = (x_{i1}, \ldots, x_{ip})^T$, its objective is to find h observations (with $\lfloor (n + p + 1)/2 \rfloor \le h \le n$) whose covariance matrix has the lowest determinant. The MCD estimate of location $\hat{\mu}$ is then the average of these h points, and the scatter estimate $\hat{\Sigma}$ is a multiple of their covariance matrix. Consistency and asymptotic normality of the MCD estimator has been shown by Butler, Davies and Jhun (1993) and Cator and Lopuhaä (2010). The MCD has a bounded influence function (Croux and Haesbroeck 1999) and it has the highest possible breakdown value (i.e., 50%) when $h = \lfloor (n + p + 1)/2 \rfloor$ (Lopuhaä and Rousseeuw 1991). In addition to being highly resistant to outliers, the MCD is affine equivariant, that is, the estimates behave properly under affine transformations of the data. To be precise, the estimators $\hat{\mu}$ and $\hat{\Sigma}$ are affine equivariant if for any $n \times p$ dataset X it holds

Mia Hubert is Professor, Department of Mathematics, KU Leuven, Belgium (E-mail: *mia.hubert@wis.kuleuven.be*). Peter J. Rousseeuw is Professor, Department of Mathematics, KU Leuven, Belgium (E-mail: *peter.rousseeuw@wis.kuleuven.be*). Tim Verdonck is Assistant Professor, Department of Mathematics, KU Leuven, Belgium (E-mail: *tim.verdonck@wis.kuleuven.be*).

that

$$\hat{\boldsymbol{\mu}}(XA + \mathbf{1}_n\boldsymbol{v}^T) = \hat{\boldsymbol{\mu}}(X)A + \boldsymbol{v}, \tag{1}$$

$$\hat{\boldsymbol{\Sigma}}(XA + \mathbf{1}_n\boldsymbol{v}^T) = A^T\hat{\boldsymbol{\Sigma}}(X)A, \tag{2}$$

for all nonsingular $p \times p$ matrices A and all $p \times 1$ vectors $\boldsymbol{v}$. The vector $\mathbf{1}_n$ denotes $(1, 1, \ldots, 1)^T$ with n entries. Affine equivariance makes the analysis independent of the measurement scales of the variables, as well as translations and rotations of the data. On the other hand, there are high dimensional situations with many outliers where affine equivariance has to be given up (Alqallaf et al. 2009).

Although the MCD was already introduced in 1984, its practical use only became feasible since the introduction of the computationally efficient FASTMCD algorithm of Rousseeuw and Van Driessen (1999). Since then, the MCD has been applied in various fields such as quality control, medicine, finance, image analysis, and chemistry, see, for example, Hubert, Rousseeuw, and Van Aelst (2008) and Hubert and Debruyne (2010) for references. The MCD is also being used as a basis to develop robust and computationally efficient multivariate techniques, for example, principal component analysis (Croux and Haesbroeck 2000; Hubert, Rousseeuw, and Vanden Branden 2005), factor analysis (Pison et al. 2003), classification (Hubert and Van Driessen 2004), clustering (Hardin and Rocke 2004), and multivariate regression (Rousseeuw et al. 2004). For a review, see Hubert, Rousseeuw, and Van Aelst (2008). The FASTMCD algorithm starts by drawing random subsets of size $p + 1$. It needs to draw many to obtain at least one that is outlier-free. Starting from each subset, several iteration steps are taken, as will be described in the next section. The overall computation time of FASTMCD is thus roughly proportional to the number of initial subsets.

In this article, we will present a deterministic algorithm for robust location and scatter, denoted as DetMCD, which uses the same iteration steps as FASTMCD but does not draw random subsets. Unlike the latter, it is permutation invariant, that is, the result does not depend on the order of the observations in the dataset. It starts from only a few well chosen initial estimates. Typically DetMCD runs even faster than FASTMCD, as will be illustrated later. In Section 2, we give brief descriptions of FASTMCD and the orthogonalized Gnanadesikan–Kettenring (OGK) estimator of Maronna and Zamar (2002), since parts of both are used in Section 3 to construct the new DetMCD algorithm. Section 4 reports on an extensive simulation study, showing that DetMCD is at least as robust as FASTMCD. In Section 5, we show that DetMCD is permutation invariant and close to affine equivariant. Section 6 illustrates the algorithm on several real datasets with applications involving principal component analysis, discriminant analysis, and time series analysis.

2. FASTMCD AND OGK

In this section, we briefly describe the FASTMCD algorithm and the OGK estimator, as our new algorithm DetMCD will use aspects of both. The observations will be denoted as $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$, whereas the columns of our data matrix are denoted by $\boldsymbol{X}_j$ $(j =$

$1, \ldots, p$). For a dataset X with estimated center $\hat{\boldsymbol{\mu}}$ and scatter matrix $\hat{\boldsymbol{\Sigma}}$, the statistical distance of the ith observation $\boldsymbol{x}_i$ will be written as $D(\boldsymbol{x}_i, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}) = \sqrt{(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^T \hat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})}$.

2.1 THE FASTMCD ALGORITHM

A major component of the FASTMCD algorithm is the *concentration step* (C-step), which works as follows. Given initial estimates $\hat{\boldsymbol{\mu}}_{\text{old}}$ for the center and $\hat{\boldsymbol{\Sigma}}_{\text{old}}$ for the scatter matrix,

(1) Compute the distances $d_{\text{old}}(i) = D(\boldsymbol{x}_i, \hat{\boldsymbol{\mu}}_{\text{old}}, \hat{\boldsymbol{\Sigma}}_{\text{old}})$ for $i = 1, \ldots, n$.

(2) Sort these distances, yielding a permutation π for which $d_{\text{old}}(\pi(1)) \leq d_{\text{old}}(\pi(2)) \leq \cdots \leq d_{\text{old}}(\pi(n))$, and set $H = \{\pi(1), \pi(2), \ldots, \pi(h)\}$.

(3) Compute $\hat{\boldsymbol{\mu}}_{\text{new}} = \sum_{i \in H} \boldsymbol{x}_i / h$ and $\hat{\boldsymbol{\Sigma}}_{\text{new}} = \sum_{i \in H} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{\text{new}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{\text{new}})^T / (h - 1)$.

In Theorem 1 of Rousseeuw and Van Driessen (1999) it was proved that $\det(\hat{\boldsymbol{\Sigma}}_{\text{new}}) \leq \det(\hat{\boldsymbol{\Sigma}}_{\text{old}})$, with equality only if $\hat{\boldsymbol{\Sigma}}_{\text{new}} = \hat{\boldsymbol{\Sigma}}_{\text{old}}$. Therefore, if we apply C-steps iteratively, the sequence of determinants obtained in this way must converge in a finite number of steps (because there are only finitely many h-subsets). Since there is no guarantee that the final value of the iteration process is the global minimum of the MCD objective function, an approximate MCD solution is obtained by taking many (by default 500) initial h-subsets $H_1 \subset \{1, 2, \ldots, n\}$, applying C-steps to each, and keeping the solution with the overall lowest determinant.

To construct an initial subset H_1, a random $(p + 1)$-subset J is drawn and $\hat{\boldsymbol{\mu}}_0 = \sum_{i \in J} \boldsymbol{x}_i / (p + 1)$ and $\hat{\boldsymbol{\Sigma}}_0 = \sum_{i \in J} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_0)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_0)^T / p$ are computed. (If $\hat{\boldsymbol{\Sigma}}_0$ is singular, random points are added to J until it becomes nonsingular.) Next, we apply the C-step to $(\hat{\boldsymbol{\mu}}_0, \hat{\boldsymbol{\Sigma}}_0)$ yielding $(\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\Sigma}}_1)$, and so on. Since each C-step involves the calculation of a covariance matrix, its inverse, and the corresponding distances, we do not want to use too many. Therefore, the FASTMCD algorithm only applies two C-steps to each initial subset, and only on the 10 subsets with lowest determinant further C-steps are taken until convergence. The raw FASTMCD estimates, $\hat{\boldsymbol{\mu}}_{\text{RAWMCD}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{RAWMCD}}$, then correspond to the empirical mean and covariance matrix of the h-subset with the lowest determinant. To increase the statistical efficiency while retaining high robustness, weighted estimators $\hat{\boldsymbol{\mu}}_{\text{FASTMCD}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{FASTMCD}}$ are computed as a weighted mean and covariance matrix with weights $w_i = 1$ if $D(\boldsymbol{x}_i, \hat{\boldsymbol{\mu}}_{\text{RAWMCD}}, \hat{\boldsymbol{\Sigma}}_{\text{RAWMCD}}) \leq \sqrt{\chi^2_{p, 0.975}}$ and 0 otherwise. Here, $\chi^2_{p, \alpha}$ is the α-quantile of the χ^2_p distribution.

Implementations of the FASTMCD algorithm are available in the package S-PLUS (as the built in function `cov.mcd`), in R (as part of the packages `rrcov`, `robust`, and `robustbase`), in SAS/IML Version $\geq$ 7, and in SAS Version $\geq$ 9 (in `PROC ROBUSTREG`). The FASTMCD is also part of LIBRA, a Matlab LIBrary for Robust Analysis (Verboven and Hubert 2005) as the function `mcdcov`. Moreover, it is available in the PLS_Toolbox of Eigenvector Research (Wise et al. 2006) used in chemometrics.

2.2 THE OGK ESTIMATOR

Maronna and Zamar (2002) presented a general method to obtain positive definite and approximately affine equivariant robust scatter matrices starting from any pairwise robust scatter matrix. This method was applied to the robust covariance estimate of Gnanadesikan and Kettenring (1972). The resulting multivariate location and scatter estimates are called OGK estimates and are calculated as follows:

(1) Let $m(.)$ and $s(.)$ be robust univariate estimators of location and scale.

(2) Construct $z_i = D^{-1}x_i$ for $i = 1, \ldots, n$ with $D = \mathrm{diag}(s(X_1), \ldots, s(X_p))$.

(3) Compute the "correlation matrix" U of the variables of $Z = (Z_1, \ldots, Z_p)$, given by $u_{jk} = 1/4(s(Z_j + Z_k)^2 - s(Z_j - Z_k)^2)$.

(4) Compute the matrix E of eigenvectors of U and

(a) project the data on these eigenvectors, that is, $V = ZE$;

(b) compute "robust variances" of $V = (V_1, \ldots, V_p)$, that is, $\mathbf{L} = \mathrm{diag}(s^2(V_1), \ldots, s^2(V_p))$;

(c) set the $p \times 1$ vector $\hat{\mu}(Z) = Em$, where $m = (m(V_1), \ldots, m(V_p))^T$, and compute the positive definite matrix $\hat{\Sigma}(Z) = ELE^T$.

(5) Transform back to X, that is, $\hat{\mu}_{\mathrm{RAWOGK}} = D\hat{\mu}(Z)$ and $\hat{\Sigma}_{\mathrm{RAWOGK}} = D\hat{\Sigma}(Z)D^T$.

In the OGK algorithm, $m(.)$ is a weighted mean and $s(.)$ is the τ-scale of Yohai and Zamar (1988). Step 2 makes the estimate scale equivariant, whereas the following steps are a kind of principal components that replace the eigenvalues of U (which may be negative) by robust variances. As in the FASTMCD algorithm, the estimate is improved by a weighting step, where the cutoff value in the weight function is now taken as $c = \chi^2_{p,0.9}\mathrm{med}(d_1, \ldots, d_n)/\chi^2_{p,0.5}$ with $d_i = D(x_i, \hat{\mu}_{\mathrm{RAWOGK}}, \hat{\Sigma}_{\mathrm{RAWOGK}})$. The weighted estimates are denoted as $\hat{\mu}_{\mathrm{OGK}}$ and $\hat{\Sigma}_{\mathrm{OGK}}$.

3. DETERMINISTIC MCD ALGORITHM

3.1 GENERAL PROCEDURE

In this section, we present an alternative algorithm. First, we standardize each variable X_j by subtracting its median and dividing by the Q_n scale estimator of Rousseeuw and Croux (1993). This standardization makes the algorithm location and scale equivariant, that is, (1) and (2) hold for any nonsingular diagonal matrix A. (We also looked into centering by the spatial median, but based on speed considerations and simulation results we stayed with the coordinatewise median.) The standardized dataset is denoted by the $n \times p$ matrix Z with rows z_i^T ($i = 1, \ldots, n$) and columns Z_j ($j = 1, \ldots, p$).

Next, we construct six initial estimates $\hat{\mu}_k(Z)$ and $\hat{\Sigma}_k(Z)$ ($k = 1, \ldots, 6$) for the center and scatter of Z. Each estimator computes a preliminary estimate S_k of the covariance or correlation matrix of Z. They will be described in Section 3.2. As these S_k may have very

inaccurate eigenvalues, we apply the following steps to each. Note that the first two steps are similar to Steps 4(a) and 4(b) of the OGK algorithm:

(1) Compute the matrix E of eigenvectors of S_k and put $V = ZE$.

(2) Estimate the covariance of Z by $\hat{\Sigma}_k(Z) = ELE^T$, where $L = \text{diag}(Q_n^2(V_1), \ldots, Q_n^2(V_p))$.

(3) To estimate the center of Z, we sphere the data, apply the coordinatewise median, and transform it back, that is, $\hat{\mu}_k(Z) = \hat{\Sigma}_k^{1/2}(\text{med}(Z\hat{\Sigma}_k^{-1/2}))$.

For all six estimates $(\hat{\mu}_k(Z), \hat{\Sigma}_k(Z))$, we then compute the statistical distances

$$d_{ik} = D(z_i, \hat{\mu}_k(Z), \hat{\Sigma}_k(Z)). \tag{3}$$

For each initial estimate k, we take the $h_0 = \lceil n/2 \rceil$ observations with smallest d_{ik} and we compute the statistical distances (denoted as d_{ik}^*) based on these h_0 observations. Then, we select for all six estimates the h observations x_i with smallest d_{ik}^* and apply C-steps until convergence. The solution with the smallest determinant we call the raw DetMCD. Then, we apply a weighting step as in the FASTMCD algorithm, yielding the final DetMCD.

3.2 INITIAL SCATTER ESTIMATES

(1) The first initial scatter estimate is obtained by computing the hyperbolic tangent (sigmoid) of each column of Z, that is, $\tilde{Z}_j = \tanh(Z_j)$ for $j = 1, \ldots, p$. This bounded function reduces the effect of large coordinatewise outliers. Computing the classical correlation matrix of $\tilde{Z}$ yields $S_1 = \text{corr}(\tilde{Z})$.

(2) Now let R_j be the ranks of the column Z_j, and put $S_2 = \text{corr}(R)$. This is the Spearman correlation matrix of Z.

(3) For S_3, we compute normal scores from the ranks R_j, namely, $T_j = \Phi^{-1}((R_j - 1/3)/(n + 1/3))$, where $\Phi(.)$ is the normal cumulative distribution function, and set $S_3 = \text{corr}(T)$.

(4) The fourth scatter estimate is based on the *spatial sign* covariance matrix (Visuri, Koivunen, and Oja 2000). Define $k_i = z_i/\|z_i\|$ for all i and let $S_4 = (1/n)\sum_{i=1}^n k_i k_i^T$. (Note that this is not the usual spatial sign covariance matrix because the z_i were centered by the coordinatewise median instead of the spatial median to save computation time.)

(5) For S_5, we take the first step of the BACON algorithm (Billor, Hadi, and Velleman 2000). Consider the $\lceil n/2 \rceil$ standardized observations z_i with smallest norm, and compute their mean and covariance matrix. (Note that the BACON algorithm starts with fewer observations.)

(6) The sixth scatter estimate is the raw OGK estimator. For $m(.)$ and $s(.)$, we used the median and Q_n for reasons of simplicity (no choice of tuning parameters) and to be consistent with the other components of DetMCD.

Note that to improve speed, for $n \geq 1000$ we replace the Q_n estimator with the τ-scale of Yohai and Zamar (1988) throughout the algorithm.

4. SIMULATION STUDY

In this section, we compare the new DetMCD algorithm with FASTMCD on artificial data. All simulations were run in MATLAB R2009b (The MathWorks, Natick, MA). We wrote new code for DetMCD, whereas FASTMCD was computed with the `mcdcov` function in the Matlab library LIBRA (Verboven and Hubert 2005).

4.1 SIMULATION STUDY FOR SMALL AND MODERATE DATASETS

In this section, we study different small and moderate datasets, namely, A: $n = 100$ and $p = 2$, B: $n = 100$ and $p = 5$, C: $n = 200$ and $p = 10$, D: $n = 400$ and $p = 40$, and E: $n = 600$ and $p = 60$. The simulation was similar to the setup of Maronna and Zamar (2002). Because the DetMCD estimates are not fully affine equivariant, we need to generate correlated data. These are obtained by first generating uncorrelated normal data $\boldsymbol{y}_i \sim N_p(\boldsymbol{0}, \mathbf{I})$ and applying an affine transformation $\boldsymbol{x}_i = \boldsymbol{G}\boldsymbol{y}_i$ to them, where $\boldsymbol{G}$ is the matrix with $G_{jj} = 1$ and $G_{jk} = \rho$ for $j \neq k$. If there is no contamination ($\varepsilon = 0$), X has covariance matrix $\boldsymbol{G}^2$, and the squared multiple correlation ρ_{mult}^2 (which is the R^2 obtained by regressing any coordinate of X on all of the others) can be calculated as a function of ρ. In the simulations, we have taken ρ such that $\rho_{\text{mult}} = 0.75$, which is a rather collinear situation. Different contamination levels are considered, namely, $\varepsilon = 0\%$, 10%, 20%, 30%, and 40% and we always set the number of observations whose covariance determinant will be minimized to $h = \lfloor (n + p + 1)/2 \rfloor$ (this yields the maximal breakdown value) in both DetMCD and FASTMCD.

For both algorithms, we compute the raw and the weighted location vectors $\hat{\boldsymbol{\mu}}_{\text{raw}}(X)$ and $\hat{\boldsymbol{\mu}}(X)$, and the raw and the weighted scatter matrices $\hat{\boldsymbol{\Sigma}}_{\text{raw}}(X)$ and $\hat{\boldsymbol{\Sigma}}(X)$. The corresponding estimators for the dataset Y are obtained by transforming back to $\hat{\boldsymbol{\mu}}(Y) = \boldsymbol{G}^{-1}\hat{\boldsymbol{\mu}}(X)$ and $\hat{\boldsymbol{\Sigma}}(Y) = \boldsymbol{G}^{-1}\hat{\boldsymbol{\Sigma}}(X)\boldsymbol{G}^{-1}$. The following performance measures were considered:

(1) The objective function of the raw scatter estimator, $\text{OBJ} = \det \hat{\boldsymbol{\Sigma}}_{\text{raw}}(Y)$.

(2) An error measure of the location estimator, given by $e_\mu = ||\hat{\boldsymbol{\mu}}(Y)||^2$.

(3) An error measure of the scatter estimate, defined as the logarithm of its condition number: $e_\Sigma = \log_{10}(\text{cond}(\hat{\boldsymbol{\Sigma}}(Y)))$.

(4) The computation time t (in sec).

Each of these performance measures should be as close to zero as possible.

Outliers were generated in $\boldsymbol{y}$-space, and the same affine transformation $\boldsymbol{G}$ was applied to them. We considered three types of contamination: point contamination, cluster contamination, and radial contamination. In all cases $\boldsymbol{y}_i \sim N_p(\boldsymbol{0}, \mathbf{I})$ for $i = 1, \ldots, n - m$, where $m = \lfloor n\varepsilon \rfloor$ and ε is the percentage of contamination. *Point contamination* was obtained as in Maronna and Zamar (2002) by generating $\boldsymbol{y}_i \sim N_p(r\boldsymbol{a}_0\sqrt{p}, \delta^2\mathbf{I})$ for
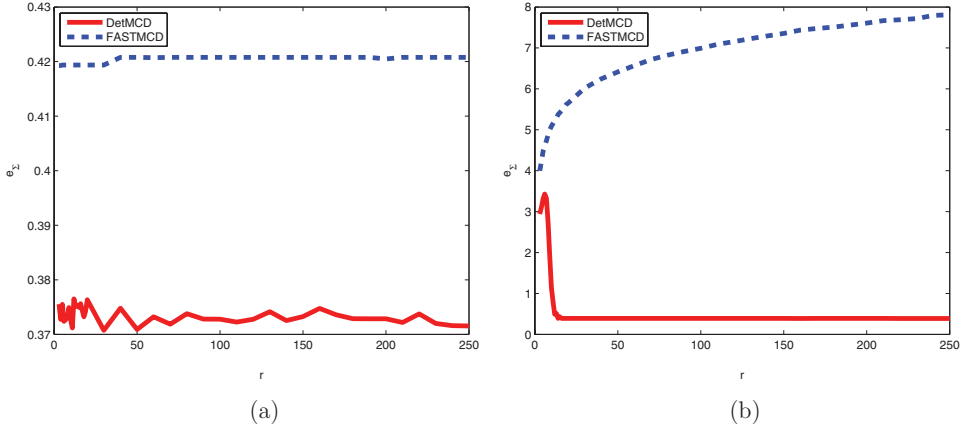
Figure 1.   The error of the scatter estimate for different values of r when $n = 100$ and $p = 5$ for (a) 10% and (b) 40% of point contamination. The online version of this figure is in color.

$i = n - m + 1, \ldots, n$ with $\delta = 0.001$, where $\boldsymbol{a}_0$ is a unit vector generated orthogonal to $(1, 1, \ldots, 1)^T$. *Cluster contamination* was generated in the same way as point contamination, but now the contamination is constructed using the same covariance matrix as the original data, hence $\delta = 1$. For *radial contamination*, many observations were generated from the distribution $N_p(\mathbf{0}, 5\mathbf{I})$ and as radial outliers we took the first m observations whose statistical distance exceeded the cutoff value $\sqrt{\chi^2_{p,0.8}}$.

The value of r, which determines the distance between the outliers and the main center (for point and cluster contamination) was varied over a wide range from $r_{\min}$ to 250, where the lower bound for r was defined as $r_{\min} = \lceil (1.2\sqrt{\chi^2_{p,1-\alpha}} + \sqrt{\delta\chi^2_{p,1-\alpha}})/\sqrt{p} \rceil$ to avoid overlap between the outliers and the regular observations.

To select the worst value of r, for each r and for each data size, contamination configuration, and contamination level, we generated 100 datasets and then computed the average e_Σ for both DetMCD and FASTMCD. We then plotted this average error measure against r, yielding 50 figures. As many showed similar behavior, we only report a few of them. In Figure 1, the results are plotted for data setting B ($n = 100$ and $p = 5$) with 10% and 40% of outliers. For a small percentage of contamination, we get approximately the same results (as the scale on the vertical axis is very small here), whereas for 40% contamination DetMCD clearly outperforms FASTMCD. Figure 2 shows the results for data setting C ($n = 200$ and $p = 10$) and 40% of point contamination and for data setting E ($n = 600$ and $p = 60$) with 10% contamination. In both situations, we see that the results for DetMCD are significantly better than those of FASTMCD. In Figure 3, the results are shown for data dimension D ($n = 400$ and $p = 40$) with 10% and 40% cluster contamination. Again, we see that for a small percentage of outliers, both methods give similar results, but DetMCD is clearly more robust when the data contain 40% of outlying observations.

We also provide a more detailed numerical output for the simulation settings under study. Each entry in the tables is now the average over 1000 runs of the performance measure in question. As many conclusions were similar, we only report the results for 0%, 10%, and 40% of contamination.
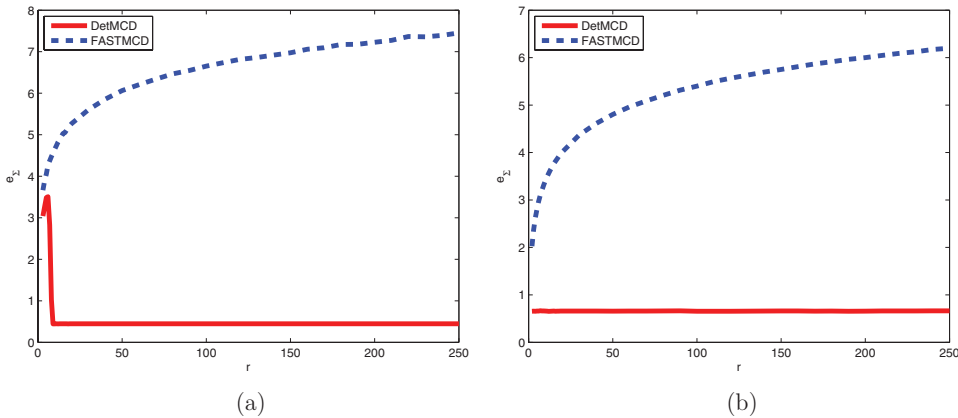
Figure 2. The error of the scatter estimate for different values of r when (a) $n = 200$, $p = 10$, and $\varepsilon = 40\%$ and (b) $n = 600$, $p = 60$, and $\varepsilon = 10\%$ (point contamination). The online version of this figure is in color.

Table 1 shows the simulation results for clean data (without contamination) for the different data sizes. We see that both methods perform similarly. In small to moderate dimensions (settings A, B, and C), FASTMCD on average attains a slightly smaller objective function than DetMCD, whereas DetMCD does better in higher dimensions (settings D and E). The error measures of location and scatter are comparable. Moreover, we see that DetMCD is much faster than FASTMCD, especially in lower dimensions. We note that in our settings roughly $t_{\text{DetMCD}} \approx t_{\text{FASTMCD}} * p/100$.

Tables 2 and 3 report the results for 10% and 40% of contamination. In the columns of point and cluster contamination, we first report the results for DetMCD's worst value of r, and then (after the slash) the results for FASTMCD's worst r.

When the amount of contamination is limited (Table 2) and the dimension is small to moderate, we do not see much difference between DetMCD and FASTMCD. The most
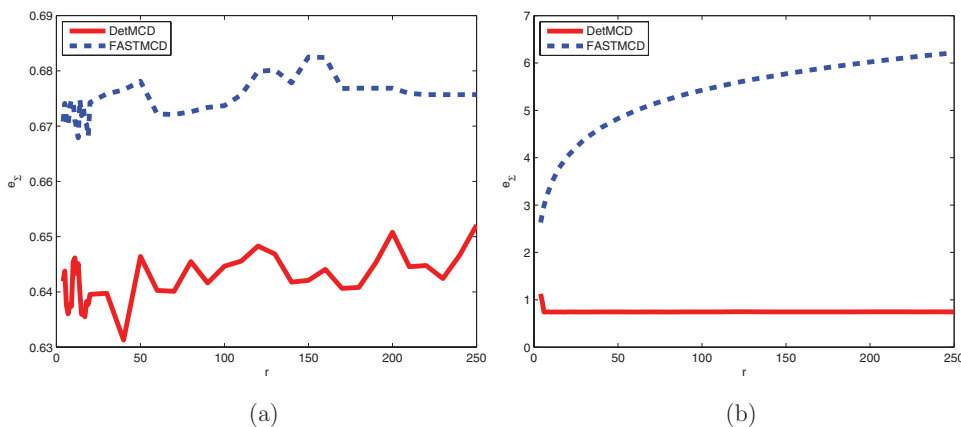

Figure 3. The error of the scatter estimate for different values of r when $n = 400$, $p = 40$ for (a) 10% and (b) 40% of cluster contamination. The online version of this figure is in color.

Table 1. Simulation results for clean data

	A		B		C		D		E	
	DetMCD	FASTMCD	DetMCD	FASTMCD	DetMCD	FASTMCD	DetMCD	FASTMCD	DetMCD	FASTMCD
OBJ	0.088	0.086	0.031	0.030	0.009	0.009	1e−5	1e−5	4.35e−7	8.68e−7
e_μ	0.028	0.031	0.065	0.073	0.060	0.063	0.124	0.132	0.1250	0.1285
e_Σ	0.175	0.202	0.390	0.460	0.393	0.418	0.636	0.668	0.6424	0.6576
t	0.019	0.498	0.029	0.581	0.096	0.868	1.775	4.349	5.7487	8.7541

notable difference shows up with point contamination in high dimensions. Here, DetMCD is much better able to withstand the effect of the contamination. This is in line with Figure 2(b).

When there is a large amount of contamination (Table 3), both algorithms show more bias in several situations. However, DetMCD is almost never doing worse than FASTMCD. Also here, we see large differences at point contamination in high dimensions. Also, cluster contamination is handled better by DetMCD, as we already observed in Figure 3(b).

The computation time does not go up with the amount of contamination, and was always lower for DetMCD. We conclude that DetMCD is a more robust and faster alternative to FASTMCD.

For DetMCD, Figure 4 shows how often on average each initial subset led (after convergence) to the smallest value of the objective function. Figure 4(a) and 4(b) shows this for

Table 2. Simulation results for data with 10% of contamination

		Point		Cluster		Radial	
		DetMCD	FASTMCD	DetMCD	FASTMCD	DetMCD	FASTMCD
A	OBJ	0.120 / 0.120	0.117 / 0.117	0.119 / 0.120	0.117 / 0.117	0.119	0.117
	e_μ	0.027 / 0.028	0.028 / 0.029	0.027 / 0.027	0.028 / 0.028	0.027	0.029
	e_Σ	0.156 / 0.158	0.171 / 0.172	0.157 / 0.157	0.171 / 0.171	0.161	0.177
	t	0.018 / 0.019	0.483 / 0.482	0.018 / 0.018	0.482 / 0.482	0.018	0.496
B	OBJ	0.047 / 0.047	0.045 / 0.045	0.047 / 0.047	0.045 / 0.045	0.047	0.045
	e_μ	0.068 / 0.068	0.074 / 0.074	0.068 / 0.068	0.074 / 0.074	0.067	0.074
	e_Σ	0.383 / 0.383	0.425 / 0.425	0.382 / 0.383	0.426 / 0.426	0.379	0.425
	t	0.028 / 0.028	0.556 / 0.555	0.028 / 0.028	0.557 / 0.557	0.028	0.579
C	OBJ	0.014 / 0.015	0.014 / 0.013	0.015 / 0.015	0.014 / 0.014	0.015	0.014
	e_μ	0.064 / 0.063	0.065 / 0.855	0.063 / 0.064	0.065 / 0.065	0.063	0.066
	e_Σ	0.399 / 0.398	0.415 / 1.037	0.398 / 0.398	0.415 / 0.415	0.397	0.414
	t	0.092 / 0.092	0.823 / 0.825	0.093 / 0.093	0.828 / 0.828	0.092	0.861
D	OBJ	3e−05 / 3e−05	5e−05 / 3e−05	4e−05 / 4e−05	5e−05 / 5e−05	4e−05	5e−05
	e_μ	0.131 / 0.130	0.135 / 175	0.131 / 0.130	0.135 / 0.135	0.129	0.136
	e_Σ	0.651 / 0.650	0.672 / 4.639	0.651 / 0.651	0.672/ 0.673	0.645	0.670
	t	1.694 / 1.710	4.395 / 4.305	1.715 / 1.717	4.362 / 4.344	1.739	4.336
E	OBJ	1e−06 / 2e−06	5e−10 / 6e−07	1e−06 / 1e−06	2e−06 / 2e−06	1e−06	2e−06
	e_μ	0.288 / 0.134	51.5 / 65317	0.134 / 0.134	0.134 / 0.134	0.135	0.136
	e_Σ	0.666 / 0.661	3.098 / 6.201	0.660 / 0.660	0.663 / 0.663	0.660	0.669
	t	5.527 / 5.527	8.530 / 8.769	5.649 / 5.644	8.773 / 8.758	5.703	8.617

Table 3. Simulation results for data with 40% of contamination

		Point		Cluster		Radial	
		DetMCD	FASTMCD	DetMCD	FASTMCD	DetMCD	FASTMCD
A	OBJ	0.018 / 0.436	0.010 / 0.165	0.436 / 0.436	0.433 / 0.433	0.435	0.433
	e_μ	13.79 / 0.033	15.24 / 272.0	0.033 / 0.033	0.033 / 0.033	0.095	0.091
	e_Σ	2.615 / 0.144	2.870 / 4.102	0.144 / 0.144	0.144 / 0.144	0.352	0.361
	t	0.019 / 0.017	0.483 / 0.483	0.017 / 0.017	0.482 / 0.482	0.016	0.495
B	OBJ	1e−04 / 0.313	3e−05 / 0.053	0.371 / 0.312	0.309 / 0.309	0.313	0.309
	e_μ	79.0 / 0.084	96.8 / 2e+05	1.206 / 0.084	0.134 / 0.085	0.086	0.086
	e_Σ	3.46 / 0.391	4.58 / 7.84	0.465 / 0.391	0.395 / 0.392	0.398	0.400
	t	0.027 / 0.027	0.550 / 0.553	0.030 / 0.027	0.553 / 0.554	0.027	0.577
C	OBJ	3e−04 / 0.168	4e−09 / 6e−06	0.168/ 0.168	110 / 1404	0.168	0.166
	e_μ	160 / 0.084	187 / 3+05	0.084 / 0.084	7111 / 90886	0.084	0.084
	e_Σ	3.58 / 0.441	4.20 / 7.43	0.441 / 0.441	4.089 / 5.127	0.440	0.442
	t	0.088 / 0.088	0.804 / 0.809	0.093 / 0.093	0.824 / 0.830	0.089	0.850
D	OBJ	5e−33 / 0.004	2e−32/ 1e−29	0.004 / 0.004	0.003/ 12.2	0.004	0.004
	e_μ	766 / 0.171	760 / 1e+06	15.7 / 0.171	99.76 / 4e+05	0.172	0.174
	e_Σ	4.57 / 0.734	5.06 / 8.13	1.03 / 0.733	2.62 / 6.21	0.735	0.737
	t	1.64 / 1.64	4.00 / 4.18	1.76 / 1.78	4.34 / 4.33	1.72	4.23
E	OBJ	5−49 / 5e−04	6e−49 / 8e−46	1e−04 / 4e−04	1e−04 / 0.819	4e−04	4e−04
	e_μ	1152 / 0.172	1142 / 2e+06	75.4 / 0.172	84.7 / 6e+05	0.171	0.171
	e_Σ	4.72 / 0.744	4.88 / 8.14	2.43 / 0.742	2.53 / 6.37	0.739	0.740
	t	5.33 / 5.32	7.13 / 7.39	5.91 / 5.77	8.70 / 8.76	5.59	8.43

10% and 40% of *point* contamination, whereas Figure 4(c) and 4(d) corresponds to 10% and 40% of *cluster* contamination. The bars at each initial subset correspond to different sample sizes (from small to large). We do not include the figures for the uncontaminated case as they were very similar to Figure 4(a) and 4(c). Also, for radial contamination, we obtained similar figures; hence, they are not included here.

We see that all six initial scatter estimates often attained the smallest objective function, which motivated us to keep them all in our algorithm. At higher dimensions, we observe that initial estimate 3 (based on the normal scores) is selected more often. With a high amount of point contamination, the fifth initial estimate (based on the BACON algorithm) is not often selected. This is because that estimate is based on a subset of size $\lceil n/2 \rceil$ which is more likely to contain contaminated cases. We also note that in high dimensions with cluster contamination, the initial estimate 6 (OGK) becomes more important. For the same configurations, we also concluded that typically three or four C-steps were needed on average to reach convergence. This explains why the computation time of DetMCD does not depend much on the contamination. On average, the DetMCD algorithm used around 21 C-steps in all, compared to over 1000 in FASTMCD.

4.2   SIMULATION STUDY FOR LARGE DATASETS

In this section, we consider simulated Gaussian data with 40% cluster contamination as in Section 4.1 with $r = 100$ for various sample sizes n ranging from 1000 to 40,000, and we concentrate on e_Σ and the required computation time (in seconds). In Figure 5, we
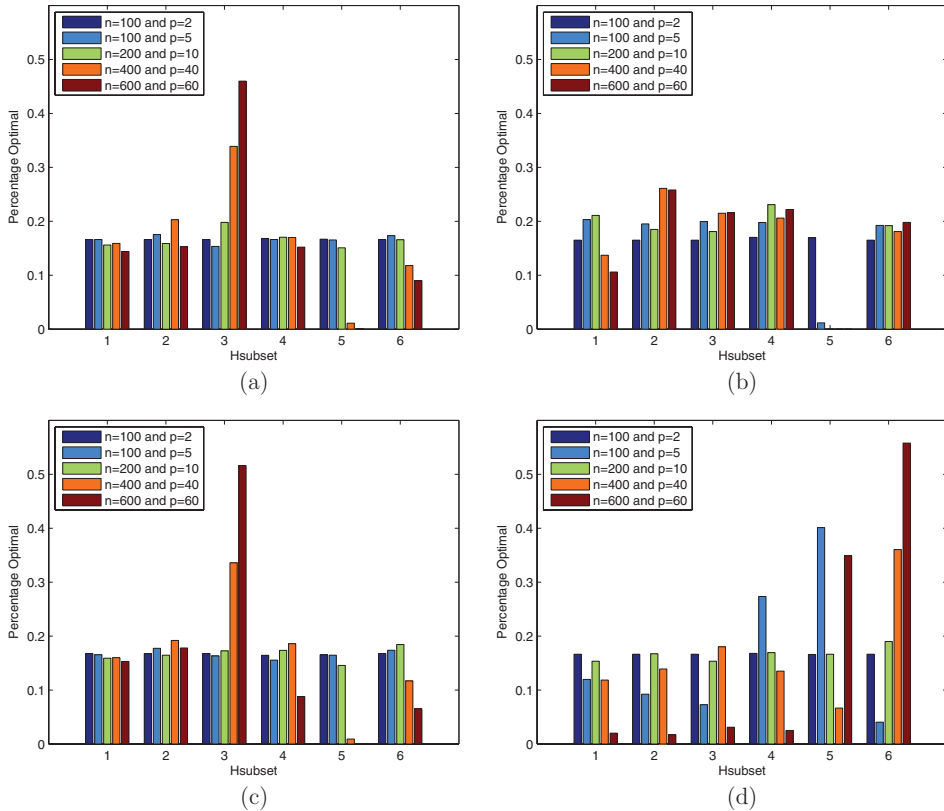
Figure 4. Relative frequency that each of the initial subsets of DetMCD led to the best objective function, for (a) 10% and (b) 40% of point contamination and for (c) 10% and (d) 40% of cluster contamination.

show the averages on two simulation runs for $p = 10$ and $p = 60$, respectively. By default, FASTMCD starts by drawing 500 random $(p + 1)$-subsets. The results for this setting are denoted as FASTMCD(500). From Figure 5(a), we see that for $p = 10$ FASTMCD(500) is slower than DetMCD up to $n = 10,000$, but it takes almost constant computing time for larger n. On the other hand, the DetMCD computation time is linear in n. Figure 5(b) displays the estimation error and clearly shows that DetMCD yields much more robust estimates. To increase the robustness of FASTMCD, more random initial subsets should be taken. If we consider 5000 initial subsets, we indeed obtain a performance similar to DetMCD, but the computation time increases considerably.

In much higher dimensions ($p = 60$), we see from Figure 5(c) that FASTMCD(500) is faster than DetMCD. But again this comes at the cost of robustness, as observed in Figure 5(d). If we now increase the number of initial subsets to 100,000, FASTMCD still performs badly, whereas its computation time has increased to roughly 1600 sec (not plotted in Figure 5(c) for clarity). This behavior can be explained by the fact that we would need at least 1.5×10^{14} initial subsets to achieve a 99% probability of drawing at least one outlier-free initial subset.
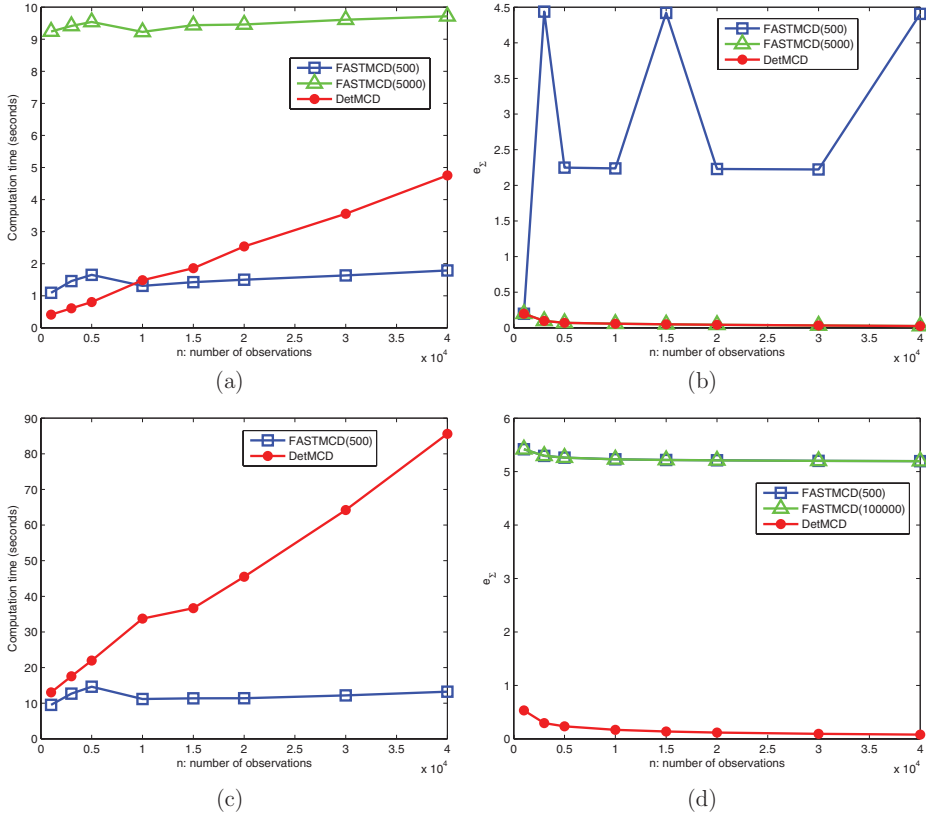
Figure 5.   Computation time for (a) $p = 10$ and (c) $p = 60$; e_Σ for (b) $p = 10$ and (d) $p = 60$. The online version of this figure is in color.

We conclude that DetMCD is more robust than FASTMCD while often requiring less computation time. Note that the larger computation time of DetMCD with respect to FAST-MCD(500) is due on the one hand to the use of pairwise scale estimates in the OGK estimator, and on the other hand to the computation and ordering of n distances in each C-step. In FASTMCD, the latter problem is solved by partitioning the data from $n \geq 600$ on, such that most C-steps are applied to subsets of the data. As this mechanism involves random splitting of the dataset, we did not implement this in DetMCD, as then our algorithm would no longer be deterministic. A hybrid approach in which such partitioning is only done for large sample sizes could however be useful in practice.

5. PROPERTIES OF DetMCD

5.1 AFFINE EQUIVARIANCE

DetMCD is not fully affine equivariant any more due to the construction of the initial estimates. We will measure its deviation from affine equivariance as it was done in Maronna and Zamar (2002) for the OGK. Since DetMCD is clearly location equivariant, we can drop v from (1) and (2) and only consider nonsingular matrices A. We

Table 4. Deviation from affine equivariance for the weighted estimators on simulated data with 0% (clean) and 10% of contamination

		Clean		Point		Cluster		Radial	
		DetMCD	OGK	DetMCD	OGK	DetMCD	OGK	DetMCD	OGK
A	d_μ	0.0184	0.0231	0.0131	0.0501	0.0139	0.0434	0.0120	0.0246
	d_Σ	1.0768	1.0811	1.0504	1.1642	1.0538	1.1537	1.0813	1.1021
B	d_μ	0.0564	0.0751	0.0535	0.0848	0.0518	0.0824	0.0333	0.0802
	d_Σ	1.3183	1.4183	1.3204	1.5754	1.2593	1.5630	1.1620	1.5078
C	d_μ	0.0491	0.0842	0.0585	0.1068	0.0479	0.0992	0.0385	0.0813
	d_Σ	1.2549	1.5313	1.2795	1.6804	1.2424	1.6579	1.1979	1.5707
D	d_μ	0.0836	0.1439	0.1213	0.1808	0.1089	0.1641	0.0973	0.1503
	d_Σ	1.5741	1.9683	1.8206	2.1407	1.7734	2.1126	1.6857	2.0224
E	d_μ	0.0679	0.1460	0.1129	0.1758	0.1154	0.1636	0.1052	0.1619
	d_Σ	1.4430	1.9556	1.8250	2.1331	1.8035	2.0965	1.7365	1.9814

generate such a $p \times p$ matrix A as the product of a random orthogonal matrix and a diagonal matrix diag($u_1, \ldots, u_p$), where the u_i are independent and uniformly distributed on $(0, 1)$. Let $X_A = \{Ax_1, \ldots, Ax_n\}$. We then compare the original estimates $\hat{\mu}_X = \hat{\mu}(X)$ and $\hat{\Sigma}_X = \hat{\Sigma}(X)$ with $\hat{\mu}_A = A^{-1}\hat{\mu}(X_A)$ and $\hat{\Sigma}_A = A^{-1}\hat{\Sigma}(X_A)(A^{-1})^T$. Maronna and Zamar (2002) measured the deviation from equivariance by $d_\mu = ||\hat{\mu}_A - \hat{\mu}_X||$ and $d_\Sigma = \text{cond}(\hat{\Sigma}_X^{-1/2}\hat{\Sigma}_A\hat{\Sigma}_X^{-1/2})$. Note that affine equivariant estimators satisfy $d_\mu = 0$ and $d_\Sigma = 1$. We considered datasets from the simulation in Section 4.1. For 20 such datasets, we generated 50 matrices A. Tables 4 and 5 report d_μ and d_Σ for 0%, 10%, and 40% contamination. For point and cluster contamination, we chose $r = 100$ throughout. We see that DetMCD was closer to affine equivariance than OGK in each case. Since Maronna and Zamar (2002) concluded that the OGK's deviation from affine equivariance was small enough not to be concerned about, this holds even more so for DetMCD.

Table 5. Deviation from affine equivariance for the weighted estimators on simulated data with 40% of contamination

		Point		Cluster		Radial	
		DetMCD	OGK	DetMCD	OGK	DetMCD	OGK
A	d_μ	0.0005	0.0214	0.0008	0.0271	0.0182	0.0641
	d_Σ	1.0017	1.0615	1.0027	1.0619	1.0785	1.2980
B	d_μ	0.0000	0.1355	0.0063	0.1403	0.0149	0.0647
	d_Σ	1.0000	1.4762	1.0217	1.4849	1.0482	1.6777
C	d_μ	0.0000	0.1539	0.0091	0.1108	0.0037	0.0513
	d_Σ	1.0000	1.6508	1.0532	1.5339	1.0272	1.3933
D	d_μ	0.0516	0.3081	0.0465	0.2234	0.0527	0.0938
	d_Σ	1.4211	2.6982	1.3662	2.2761	1.13763	1.7873
E	d_μ	0.0661	0.3075	0.0537	0.2298	0.0408	0.0925
	d_Σ	1.5652	2.6326	1.5150	2.2692	1.3334	1.8719

5.2 PERMUTATION INVARIANCE

Another property we are interested in is permutation invariance. An estimator $T(.)$ is said to be permutation invariant if $T(PX) = T(X)$ for any dataset X and any permutation matrix P. A permutation matrix is a square matrix that has a single entry 1 in each row and each column, and zeroes elsewhere. Therefore, PX simply permutes the rows of X. Note that FASTMCD is not permutation invariant because the initial subsets (generated by a pseudorandom number generator with a fixed seed) will have the same case numbers but correspond to different observations. By contrast, all ingredients of DetMCD are permutation invariant. Analogous to the previous section, the deviation from permutation invariance can be measured by $d_\mu = ||\hat{\mu}(PX) - \hat{\mu}(X)||$ and $d_\Sigma = \text{cond}(\hat{\Sigma}(X)^{-1/2}\hat{\Sigma}(PX)\hat{\Sigma}(X)^{-1/2})$. We study the permutation invariance on the *ionospheric data*, preprocessed as in Maronna and Zamar (2002), which has $n = 225$ observations and $p = 31$ variables. For FASTMCD, the average $\overline{d}_\mu = 0.0410$ and $\overline{d}_\Sigma = 12.4131$ over 1000 matrices P confirm that this algorithm is not permutation invariant. However, OGK and DetMCD are both yielding the optimal values $\overline{d}_\mu = 0$ and $\overline{d}_\Sigma = 1$.

5.3 DIFFERENT VALUES OF h

We already noted that DetMCD is faster than FASTMCD when applying the algorithm once for a fixed value of h. As the number of outliers should be below $n - h$, it is commonly advised to set $h \approx 0.5n$ when many outliers could occur, and $h \approx 0.75n$ otherwise. Alternatively, one could compute the MCD for a whole range of h-values, and see whether at some h there is an important change in the objective function or the estimates. This is related to the forward search of Atkinson, Riani, and Cerioli (2004). With DetMCD, it becomes very easy to compute the MCD for several h-values: since the initial estimates do not depend on h, we only need to store the resulting ordered distances (3), yielding the initial h-subset for any h. We will illustrate this in Section 6.

6. REAL EXAMPLES

In this section, we apply FASTMCD and DetMCD to various multivariate techniques, such as principal component analysis (PCA), classification, and time series analysis.

6.1 PRINCIPAL COMPONENT ANALYSIS

As a first example, we consider $p = 6$ measurements of $n = 100$ forged Swiss bank notes, from Flury and Riedwyl (1988). As shown in Pison and Van Aelst (2004), Salibian-Barrera, Van Aelst, and Willems (2006), and Willems, Joe, and Zamar (2009), this dataset contains several outlying observations and highly correlated variables. Therefore, it is appropriate to analyze the data with a robust PCA method. Following Croux and Haesbroeck (2000), we use the eigenvectors of the MCD scatter estimate as the robust principal components. The first three principal components are retained, because together they explain 92% of the variance. Missing values were randomly added in the data for a total of 5%. We then applied the methodology of Serneels and Verdonck (2008) to perform a robust PCA method on incomplete data. Figure 6 shows the resulting outlier maps based on FASTMCD
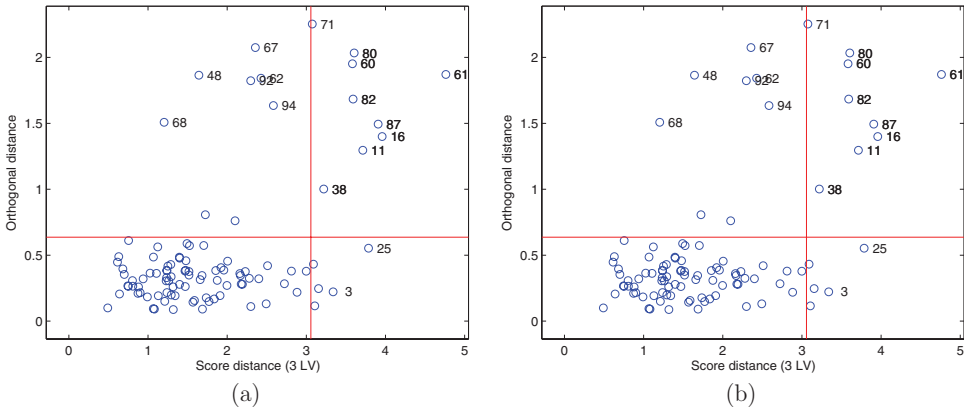
Figure 6. Outlier map of the Swiss bank notes data using robust PCA that can handle outliers with (a) DetMCD and (b) FASTMCD. The online version of this figure is in color.

and DetMCD using $h = 75$. On the horizontal axis, they have the robust distance of the observation in the three-dimensional PCA subspace. The vertical axis shows the orthogonal distance of the observation to the PCA subspace. Such an outlier map allows to classify observations into regular cases, good PCA leverage points, orthogonal outliers, and bad PCA leverage points (Hubert, Rousseeuw, and Vanden Branden 2005). We see that the outlier maps are very similar, and that the same observations are flagged as outlying. The optimal h-subsets obtained with both algorithms had $h - 2$ points in common. The main difference lies in the computation times: FASTMCD took 197 sec whereas DetMCD only needed 10 sec.

6.2 CLASSIFICATION

The fruit dataset is high dimensional and contains spectra of three different cultivars (with sizes 490, 106, and 500) of a type of cantaloupe, and was previously analyzed by Hubert and Van Driessen (2004). All spectra were measured at 256 wavelengths, hence the dataset contains 1096 observations and 256 variables. First, we performed a robust PCA using the ROBPCA method of Hubert, Rousseeuw, and Vanden Branden (2005). ROBPCA mainly consists of two steps. In the first step, a robust subspace is constructed based on the Stahel–Donoho outlyingness (see, e.g., Debruyne and Hubert 2009). Next, robust eigenvectors and eigenvalues are found within this subspace by applying the MCD to the projected observations. We consider the original ROBPCA method that uses FASTMCD in the second stage of the algorithm, and a modified ROBPCA that applies DetMCD. From the scree plot, we decided to retain two principal components. Again, FASTMCD and DetMCD gave identical results. We note a big group of outliers in the outlier map in Figure 7(a), which corresponds to a change in the instrument's illumination system.

Next, we applied the robust quadratic discriminant rule RQDR (Hubert and Van Driessen 2004) to the robust two-dimensional PCA scores for values of h from 550 to 1095. The RQDR method first runs the MCD estimator on each of the groups. A new datum is then assigned to the group for which it attains the largest discriminant score. Also, membership
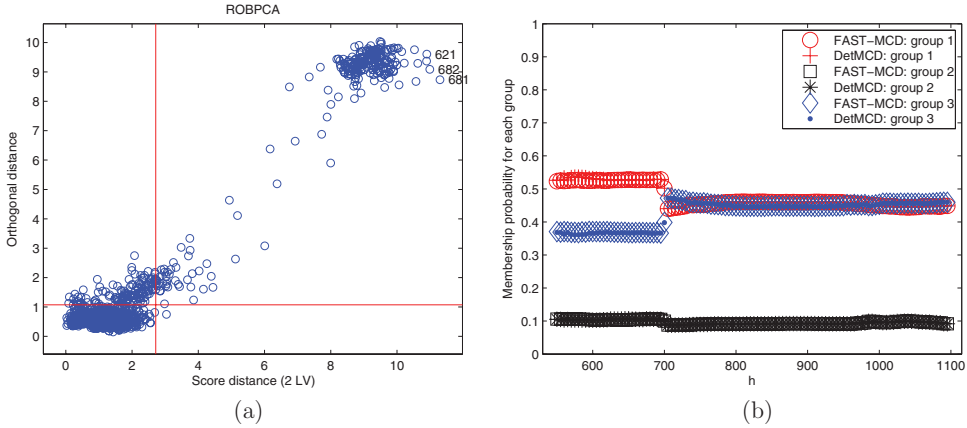
Figure 7. Fruit data (a) outlier map using ROBPCA; (b) membership probabilities of each group for different values of h. The online version of this figure is in color.

probabilities for each group are estimated, as the proportion of regular observations in each group. Figure 7(b) shows these membership probabilities obtained with both methods as a function of h. Also, here FASTMCD and DetMCD give almost the same results. We see that the membership probabilities change significantly at $h = 700$, hence there are a substantial number of outliers present. Therefore, h should be taken sufficiently small to obtain robust results. The entire analysis took 241 sec when using FASTMCD, whereas it only needed 44 sec with DetMCD. The computation time only went down by a factor of six here because the analyses have several parts in common, such as the computation of the discriminant scores.

6.3 MULTIVARIATE TIME SERIES

Our last example illustrates an econometric application of MCD. Recently Croux, Gelper, and Mahieu (2010) proposed a robust version of multivariate exponential smoothing of multivariate time series, based on the MCD. Exponential smoothing is a widely used technique to forecast time series and is defined in a recursive way. Let $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T$ be a multivariate time series and assume that the (robust) smoothed values of $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}$ are already computed (denoted $\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_{t-1}$) up to time point $t - 1$, then the next robust smoothed value is given by $\hat{\boldsymbol{y}}_t = \boldsymbol{\Lambda} \boldsymbol{y}_t^* + (\boldsymbol{I} - \boldsymbol{\Lambda}) \hat{\boldsymbol{y}}_{t-1}$, where $\boldsymbol{\Lambda}$ is the smoothing matrix and $\boldsymbol{y}_t^*$ is the cleaned version of the p-dimensional vector $\boldsymbol{y}_t$. The forecast for $\boldsymbol{y}_{T+1}$ that can be made at time T is given by

$$\hat{\boldsymbol{y}}_{T+1|T} = \hat{\boldsymbol{y}}_T = \boldsymbol{\Lambda} \sum_{k=0}^{T-1} (\boldsymbol{I} - \boldsymbol{\Lambda})^k \boldsymbol{y}_{T-k}.$$

This multivariate cleaned series is calculated as follows:

$$\boldsymbol{y}_t^* = \frac{\psi\left(\sqrt{\boldsymbol{r}^T \hat{\boldsymbol{\Sigma}}_t^{-1} \boldsymbol{r}_t}\right)}{\sqrt{\boldsymbol{r}^T \hat{\boldsymbol{\Sigma}}_t^{-1} \boldsymbol{r}_t}} \boldsymbol{r}_t + \hat{\boldsymbol{y}}_{t|t-1},$$

where $\boldsymbol{r}_t = \boldsymbol{y}_t - \hat{\boldsymbol{y}}_{t|t-1}$ denotes the one-step-ahead forecast error, ψ is the Huber ψ-function with clipping constant $\sqrt{\chi^2_{p,0.95}}$, and $\hat{\boldsymbol{\Sigma}}_t$ is an estimated covariance matrix of the one-step-ahead forecast error at time t (for more details, see Croux, Gelper, and Mahieu 2010). The robust version uses the MCD in two different stages of the algorithm. First, the starting values are obtained by the MCD-based robust multivariate regression of Rousseeuw et al. (2004). Second, the MCD is used as a loss function to choose the smoothing matrix $\boldsymbol{\Lambda}$. This selection of the smoothing matrix uses a data-driven approach and hence, the MCD is applied in an iterative manner (during a certain training period). Croux, Gelper, and Mahieu (2010) have illustrated their method on the housing dataset from Diebold (2001), a real bivariate time series of monthly data. By using a startup period of length 10 and the complete series as training sample, the following smoothing matrix $\boldsymbol{\Lambda}$ was obtained:

$$\boldsymbol{\Lambda} = \begin{pmatrix} 0.68 & 0.04 \\ 0.04 & 0.62 \end{pmatrix}.$$

When redoing this example with DetMCD instead of FASTMCD, the exact same smoothing matrix was found. With FASTMCD, this analysis took 1 hr and 52 min, whereas with DetMCD it only took 22 min. This speedup will become even more important when considering higher dimensional time series.

7. CONCLUSIONS AND OUTLOOK

DetMCD is a new algorithm which is typically more robust than FASTMCD and needs even less time. It starts from a few easily computed h-subsets, and then takes concentration steps until convergence. The DetMCD algorithm is deterministic in that it does not use any random subsets. It is permutation invariant and close to affine equivariant, and allows to run the analysis for many values of h without much additional computation. We illustrated DetMCD in the contexts of PCA, regression, classification, and time series analysis.

Also, many other methods that directly or indirectly rely on the MCD (e.g., through its robust distances) may benefit from the DetMCD approach, such as robust canonical correlation (Croux and Dehon 2002), robust regression with continuous and categorical regressors (Hubert and Rousseeuw 1996), robust errors-in-variables regression (Fekri and Ruiz-Gazen 2004), and robust calibration (Hubert and Verboven 2003; Hubert and Vanden Branden 2003). In particular, online applications or procedures that require the MCD to be computed many times, such as genetic algorithms (Wiegand, Pell, and Comas 2009), will become more efficient. The cross-validation techniques of Engelen and Hubert (2005) and Hubert and Engelen (2007) may benefit from the fact that DetMCD is easily updated when an observation is added or removed. Following Copt and Victoria-Feser (2004) and Serneels and Verdonck (2008), we will also investigate whether DetMCD can be extended to the missing data framework.

The DetMCD algorithm is available in Matlab as part of LIBRA (Verboven and Hubert 2005) and can be downloaded from *http://wis.kuleuven.be/stat/robust/*. An R implementation is also available from this website.

The random sampling mechanism is currently used for many other high-breakdown robust estimators. Our deterministic approach could improve on those algorithms as well.

In particular, we intend to study a deterministic algorithm for S-estimators and τ-estimators, for which algorithms in the spirit of FASTMCD were developed recently (Salibian-Barrera and Yohai 2006; Salibian-Barrera, Willems, and Zamar 2008). We will also work on a deterministic algorithm for LTS regression, which is typically computed with the FASTLTS algorithm (Rousseeuw and Van Driessen 2006).

SUPPLEMENTAL MATERIALS

Matlab code for DetMCD algorithm: Matlab code to run the DetMCD algorithm has been proposed in this article. Note that this requires the Matlab library for Robust Analysis LIBRA, which can be downloaded freely from *http://wis.kuleuven.be/ stat/robust/LIBRA.html* (.m file).

Datasets: Matlab file that contains all the datasets used in this article (.mat file).

ACKNOWLEDGMENTS

REFERENCES

Alqallaf, F., Van Aelst, S., Yohai, V. J., and Zamar, R. H. (2009), "Propagation of Outliers in Multivariate Data," *The Annals of Statistics*, 37, 311–331. [619]

Atkinson, A., Riani, M., and Cerioli, A. (2004), *Exploring Multivariate Data With the Forward Search*, New York, NY: Springer-Verlag. [631]

Billor, N., Hadi, A., and Velleman, P. (2000), "Bacon: Blocked Adaptive Computationally Efficient Outlier Nominators," *Computational Statistics and Data Analysis*, 34 (3), 279–298. [622]

Butler, R., Davies, P., and Jhun, M. (1993), "Asymptotics for the Minimum Covariance Determinant Estimator," *The Annals of Statistics*, 21, 1385–1400. [618]

Cator, E., and Lopuhaä, H. (2010), "Asymptotic Expansion of the Minimum Covariance Determinant Estimators," *Journal of Multivariate Analysis*, 101, 2372–2388. [618]

Copt, S., and Victoria-Feser, M.-P. (2004), "Fast Algorithms for Computing High Breakdown Covariance Matrices With Missing Data," in *Theory and Applications of Recent Robust Methods*, eds. M. Hubert, G. Pison, A. Struyf, and S. Van Aelst, Basel: Birkhäuser, Statistics for Industry and Technology. [634]

Croux, C., and Dehon, C. (2002), "Analyse canonique basée sur des estimateurs robustes de la matrice de covariance [Canonical Correlation Based on Robust Estimators of the Covariance Matrix]," *La Revue de Statistique Appliquée*, 2, 5–26. [634]

Croux, C., Gelper, S., and Mahieu, K. (2010), "Robust Exponential Smoothing of Multivariate Time Series," *Computational Statistics and Data Analysis*, 54, 2999–3006. [633]

Croux, C., and Haesbroeck, G. (1999), "Influence Function and Efficiency of the Minimum Covariance Determinant Scatter Matrix Estimator," *Journal of Multivariate Analysis*, 71, 161–190. [618]

——— (2000), "Principal Components Analysis Based on Robust Estimators of the Covariance or Correlation Matrix: Influence Functions and Efficiencies," *Biometrika*, 87, 603–618. [619,631]

Debruyne, M., and Hubert, M. (2009), "The Influence Function of the Stahel–Donoho Covariance Estimator of Smallest Outlyingness," *Statistics and Probability Letters*, 79, 275–282. [632]

Diebold, F. (2001), *Elements of Forecasting*, Cincinnati, OH: South-Western. [634]

Engelen, S., and Hubert, M. (2005), "Fast Model Selection for Robust Calibration," *Analytica Chemica Acta*, 544, 219–228. [634]

Fekri, M., and Ruiz-Gazen, A. (2004), "Robust Weighted Orthogonal Regression in the Errors-in-Variables Model," *Journal of Multivariate Analysis*, 88, 89–108. [634]

Flury, B. and Riedwyl, H. (1988), *Multivariate Statistics: A Practical Approach*, London: Chapman and Hall. [631]

Gnanadesikan, R., and Kettenring, J. (1972), "Robust Estimates, Residuals, and Outlier Detection With Multiresponse Data," *Biometrics*, 28, 81–124. [621]

Hardin, J., and Rocke, D. (2004), "Outlier Detection in the Multiple Cluster Setting Using the Minimum Covariance Determinant Estimator," *Computational Statistics and Data Analysis*, 44, 625–638. [619]

Hubert, M., and Debruyne, M. (2010), "Minimum Covariance Determinant," *Wiley Interdisciplinary Reviews: Computational Statistics*, 2, 36–43. [619]

Hubert, M., and Engelen, S. (2007), "Fast Cross-Validation for High-Breakdown Resampling Algorithms for PCA," *Computational Statistics and Data Analysis*, 51, 5013–5024. [634]

Hubert, M., and Rousseeuw, P. (1996), "Robust Regression With Both Continuous and Binary Regressors," *Journal of Statistical Planning and Inference*, 57, 153–163. [634]

Hubert, M., Rousseeuw, P., and Van Aelst, S. (2008), "High Breakdown Robust Multivariate Methods," *Statistical Science*, 23, 92–119. [619]

Hubert, M., Rousseeuw, P., and Vanden Branden, K. (2005), "ROBPCA: A New Approach to Robust Principal Components Analysis," *Technometrics*, 47, 64–79. [619,632]

Hubert, M., and Van Driessen, K. (2004), "Fast and Robust Discriminant Analysis," *Computational Statistics and Data Analysis*, 45, 301–320. [619,632]

Hubert, M., and Vanden Branden, K. (2003), "Robust Methods for Partial Least Squares Regression," *Journal of Chemometrics*, 17, 537–549. [634]

Hubert, M., and Verboven, S. (2003), "A Robust PCR Method for High-Dimensional Regressors," *Journal of Chemometrics*, 17, 438–452. [634]

Lopuhaä, H., and Rousseeuw, P. (1991), "Breakdown Points of Affine Equivariant Estimators of Multivariate Location and Covariance Matrices," *The Annals of Statistics*, 19, 229–248. [618]

Maronna, R., and Zamar, R. (2002), "Robust Estimates of Location and Dispersion for High-Dimensional Data Sets," *Technometrics*, 44, 307–317. [619,621,623,629,631]

Pison, G., Rousseeuw, P., Filzmoser, P., and Croux, C. (2003), "Robust Factor Analysis," *Journal of Multivariate Analysis*, 84, 145–172. [619]

Pison, G., and Van Aelst, S. (2004), "Diagnostic Plots for Robust Multivariate Methods," *Journal of Computational and Graphical Statistics*, 13, 310–329. [631]

Rousseeuw, P. (1984), "Least Median of Squares Regression," *Journal of the American Statistical Association*, 79, 871–880. [618]

Rousseeuw, P., and Croux, C. (1993), "Alternatives to the Median Absolute Deviation," *Journal of the American Statistical Association*, 88, 1273–1283. [621]

Rousseeuw, P., Van Aelst, S., Van Driessen, K., and Agulló, J. (2004), "Robust Multivariate Regression," *Technometrics*, 46, 293–305. [619,634]

Rousseeuw, P., and Van Driessen, K. (1999), "A Fast Algorithm for the Minimum Covariance Determinant Estimator," *Technometrics*, 41, 212–223. [619,620]

——— (2006), "Computing LTS Regression for Large Data Sets," *Data Mining and Knowledge Discovery*, 12, 29–45. [635]

Salibian-Barrera, M., Van Aelst, S., and Willems, G. (2006), "PCA Based on Multivariate MM-Estimators With Fast and Robust Bootstrap," *Journal of the American Statistical Association*, 101, 1198–1211. [631]

Salibian-Barrera, M., Willems, G., and Zamar, R. (2008), "The Fast-τ Estimator for Regression," *Journal of Computational and Graphical Statistics*, 17, 659–682. [635]

Salibian-Barrera, M., and Yohai, V. J. (2006), "A Fast Algorithm for S-Regression Estimates," *Journal of Computational and Graphical Statistics*, 15, 414–427. [635]

Serneels, S., and Verdonck, T. (2008), "Principal Component Analysis for Data Containing Outliers and Missing Elements," *Computational Statistics and Data Analysis*, 52, 1712–1727. [631,634]

Verboven, S., and Hubert, M. (2005), "LIBRA: A Matlab Library for Robust Analysis," *Chemometrics and Intelligent Laboratory Systems*, 75, 127–136. [620,623,634]

Visuri, S., Koivunen, V., and Oja, H. (2000), "Sign and Rank Covariance Matrices," *Journal of Statistical Planning and Inference*, 91, 557–575. [622]

Wiegand, P., Pell, R., and Comas, E. (2009), "Simultaneous Variable Selection and Outlier Detection Using a Robust Genetic Algorithm," *Chemometrics and Intelligent Laboratory Systems*, 98, 108–114. [634]

Willems, G., Joe, H., and Zamar, R. (2009), "Diagnosing Multivariate Outliers Detected by Robust Estimators," *Journal of Computational and Graphical Statistics*, 18 (1), 73–91. [631]

Wise, B., Gallagher, N., Bro, R., Shaver, J., Windig, W., and Koch, R. (2006), *PLS_Toolbox 4.0 for Use With MATLAB*, Software, Eigenvector Research, Inc. *http://software.eigenvector.com/* [620]

Yohai, V., and Zamar, R. (1988), "High Breakdown Point Estimates of Regression by Means of the Minimization of an Efficient Scale," *Journal of the American Statistical Association*, 83, 406–413. [621,623]