Darmstadt University of Technology
Institute of Telecommunications
Signal Processing Group
Prof. Dr.-Ing. A. M. Zoubir
Merckstraße 25,
64283 Darmstadt, Germany

Bachelor Thesis
Bachelorarbeit

On the robust estimation of scale

Robuste Schätzung von Skalierung

by / von
Christian Debes
09.11.2004

Supervisor / Betreuer : Prof. Dr. Abdelhak M. Zoubir
Dr. Chris L. Brown
Dr. Ramon F. Brcic

Contents

List of Figures

List of Tables

Abstract (English)

In this work several topics of robust estimation will be covered. First a new score function for Huber's M-Estimator for scale will be presented and its performance evaluated, whereby the quality of the estimate is treated as well as its computational complexity. Furthermore, the effect of a scale estimator on the location estimate in the scenario of joint estimation of location and scale is explored.
Finally the problem of robust estimation of scale is extended to the robust estimation of covariance matrices, where we will present a scheme, based on the idea of Visuri's sample sign covariance matrix. This can find application when estimating the number of sources or their directions of arrival in wireless communications.

Abstract (Deutsch)

In dieser Arbeit werden mehrere Bereiche der robusten Schätzverfahren abgedeckt. Zuerst wird eine neue Score-Funktion für Huber's M-Estimator präsentiert, um die Skalierung einer Zufallsvariablen zu schätzen. Hierbei werden Simulationen durchgeführt, die sowohl die Qualität, als auch die Effizienz betreffen. Weiterhin erhält das Szenario der gemeinsamen Schätzung eines Lageparameters und einer Skalierung Beachtung. Insbesondere wird beobachtet, wie sich ein Fehler in der Schätzung der Skalierung auf den Lageparameter auswirkt.
Schliesslich wird das Problem der robusten Schätzverfahren für Skalierungen hin zur robusten Schätzung von Kovarianz Matrizen erweitert. Dabei wird ein Schema präsentiert, das auf Visuri's Sample Sign Covariance Matrix basiert und Anwendung in der Schätzung der Nutzeranzahl und der eintreffenden Winkel der Signale in der Mobilkommunikation findet.

Chapter 1

Introduction

In many engineering applications, observations or measurements are corrupted by additive noise which is assumed to be Gaussian distributed. Classical approaches in detection and estimation of signals are based on this assumption. In numerous scenarios, like in wireless communication or underwater acoustics, the noise is better described as heavy-tailed or impulsive. The reasons for this impulsiveness could be electromagnetic interference, man-made noise or marine life in sonar. However, even if the noise is known to be non-Gaussian, classical estimators need to be based on some assumption, like the K-distribution in radar. One question remains: What if the assumptions on the noise distribution whether Gaussian or non-Gaussian, are not fulfilled? How does the estimator react if the distribution of the noise deviates from the assumption or changes with time?
It has been shown in many applications that classical estimators are non-robust, presenting poor performance, even if the deviations from the assumptions are small. This leads, for example, to an increasing bit error rate in digital communications [4,5] or to inaccurate estimates of the number of sources or their directions of arrival in subspace estimation [3]. All this shows that robust estimation is crucial to ensure good performance is maintained, even if assumptions on the noise model are violated.

Besides the quality of an estimate, the computational complexity is an important issue. It is known that highly robust estimators like Huber's M-Estimators [1] are computational expensive. This trade-off between speed and accuracy has to be managed separately for each scenario.

Chapter 2

Robust Estimation Of Scale

Scale estimation is an important issue for many applications. Pure scale estimation problems are very rare but they are used as a first step in estimation of a location parameter and - extended to a multidimensional scenario - in estimation of covariance matrices. Consider the following observation model

$$y_n = \sigma u_n, \qquad n = 1, ..., N \tag{2.1}$$

where u_n are independent, identically distributed (i.i.d.) random variables, σ is the noise scale and N is the number of samples. Assuming the observation distribution to be Gaussian, estimation of scale by using the sample standard deviation is the best choice. But what if the distribution of the observation deviates from the nominal model in the sense that it is contaminated, e.g. it contains outliers? In section 2.1 we will evaluate and compare the performance of different scale estimators, when the observation is modeled as a two-term Gaussian mixture:

$$u \sim (1-\epsilon)\mathcal{N}(0,1) + \epsilon\mathcal{N}(0,\kappa), \qquad 0 \leq \epsilon \leq 1, \kappa > 0 \tag{2.2}$$

Herein $\mathcal{N}(0,1)$ is the nominal standard Gaussian component, which appears with probability $(1-\epsilon)$ and $\mathcal{N}(0,\kappa)$ is the contamination component, appearing with probability ϵ. Usually, it is assumed, that $\kappa >> 1$ and $\epsilon << 1$, which leads to the interpretation of outliers, buried in Gaussian noise. In the following we will use these conventions:

- $\bar{y}$ is the empirical mean of y: $\bar{y} = \frac{1}{N}\sum_{n=1}^{N} y_n$.
- $\hat{\sigma}(y;c)$ is a scale estimator, whereby $\hat{\sigma}(a \cdot y;c) = a \cdot \hat{\sigma}(y;c)$ and c is a constant, making the estimator unbiased for the nominal Gaussian distribution $\mathcal{N}(0,1)$
- The mean square error will be used as a measure of quality: $mse(\hat{\sigma}) = \frac{1}{N}\sum_{n=1}^{N}(\hat{\sigma}_n - \sigma)^2$
- Scalar quantities are denoted by regular letters, while bold letters are used for vectors.

While section 2.1 only deals with the quality of the measure, complexity and computation-time issues will be treated in section 2.2.

2.1 Scale Estimators

For simplicity in the simulations, we will assume a unit scale nominal component ($\sigma = 1$), $\kappa \in \{25, 100\}$, $N \in \{50, 100, 1000\}$ while varying $0 \leq \epsilon \leq 0.1$ Simulation results are obtained by averaging over 1000 Monte Carlo-simulations. Note, that for practical reasons, when presenting the different scale estimators, we restrict ourselves to the sample scale estimates.

2.1.1 The sample standard deviation

The sample standard deviation leads to an estimate for σ, based on the well-known formula

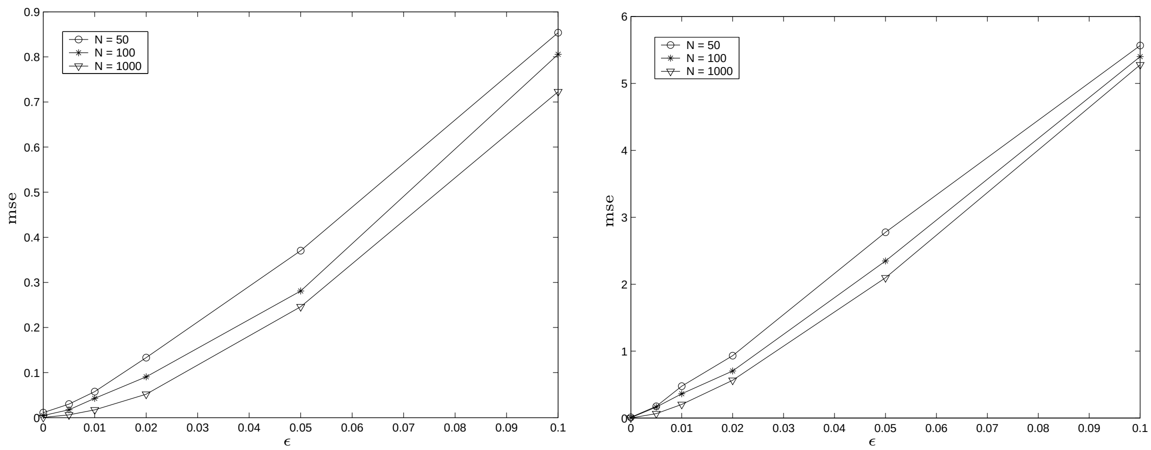
$$\hat{\sigma} = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} (y_n - \bar{y})^2} \tag{2.3}$$

It is obvious, that even one single outlier y_n can dominate the scale estimate, making the sample standard deviation a nonrobust scale estimator. We will demonstrate this with a small numerical example. Assume the noise u_n to be distributed as $\mathcal{N}(0, 1)$, having $N = 1000$ samples. As the first column of table 2.1 shows, the sample standard deviation provides a good estimate of the true scale ($\sigma = 1$). We will now include one contaminating sample u_c. Even if the degree of contamination is very low (0.001), the sample standard deviation reacts with an error of 18% for $u_c = 10$ and an error of 80% for $u_c = 50$ (Results are obtained by averaging over 1000 Monte Carlo simulations).

	No contamination	Contamination, $u_c = 10$	Contamination, $u_c = 50$
Average $\hat{\sigma}$	1.00	1.18	1.80

Table 2.1: Numerical evaluation of scale estimation, using the sample standard deviation

Figure 2.1 now shows the performance of the sample standard deviation when the observation is modeled as a two-term Gaussian mixture. We can not only observe a relative high error when increasing ϵ, similarly, the variance of the outliers, κ has a significant influence on the performance of the standard deviation. All this leads to unreliable scale estimates when Gaussianity is not fulfilled.



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.1: Performance of the sample standard deviation

2.1.2 The α-trimmed standard deviation

The α-trimmed standard deviation [9] is an intuitive way for scale estimation in the presence of outliers. A proportion $0 < \alpha < 0.5$ from the largest and smallest samples in y_n is discarded, making the α-trimmed standard deviation robust against outliers. Assuming $y_{(n)}$ to be the sorted version of y_n, i.e. $y_{(n)} \leq y_{(n+1)} \ \forall \ n = 1, ..., N$, the α-trimmed standard deviation is defined as

$$\hat{\sigma} = c \cdot \sqrt{\frac{1}{N - 2N_0 - 1} \sum_{n=N_0+1}^{N-N_0} (y_{(n)} - \bar{y})^2} \tag{2.4}$$

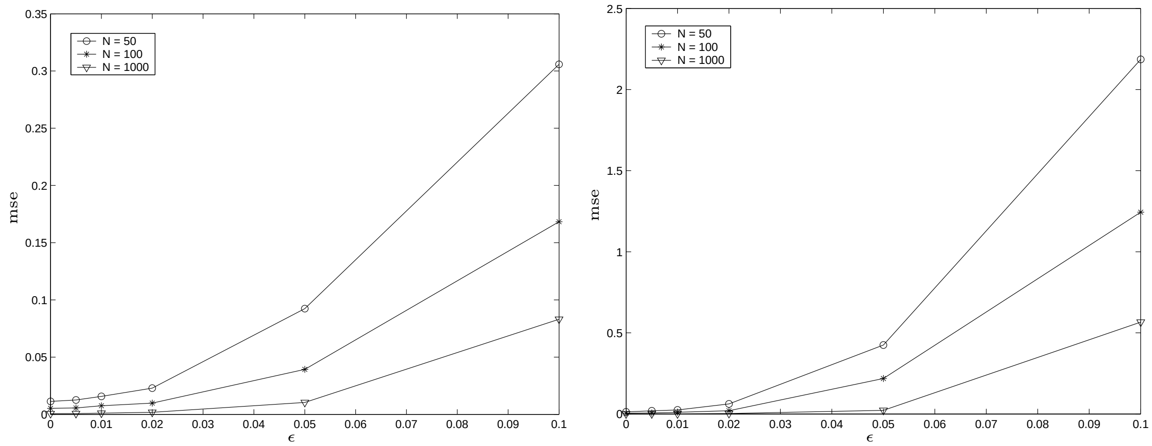
whereby $N_0 = \lceil \alpha N \rceil$. It can be seen in figures 2.2, 2.3 and 2.4, that the α-trimmed standard deviation provides insensitivity to small deviations from the Gaussian model. However, the value for α has to be chosen properly. A too large α will discard uncontaminated data, while a too small α will decrease robustness.

2.1.3 The interquartile range

The interquartile range is the difference between the first and the third quartile, representing the range of the middle 50% of the data.

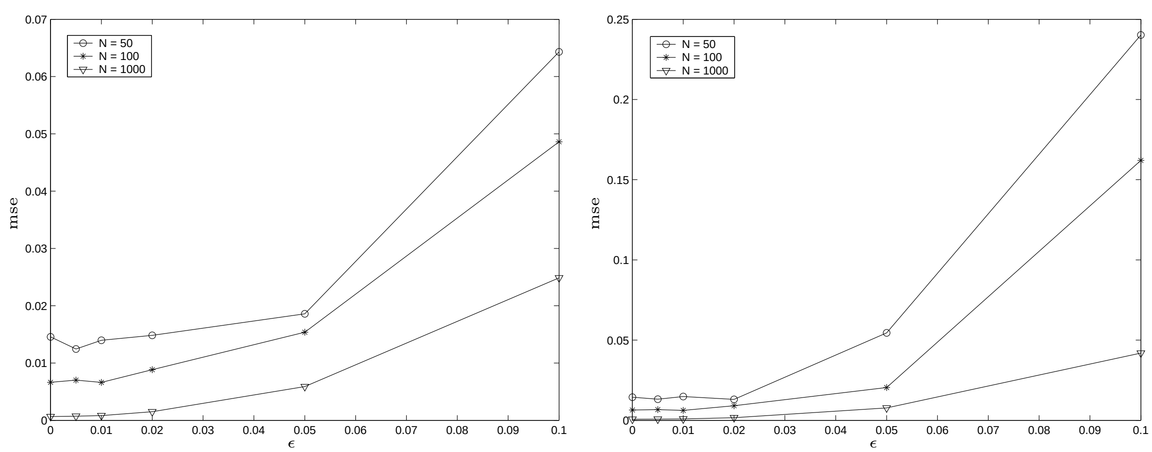
$$\hat{\sigma} = c \cdot \left(y_{(\frac{3N}{4})} - y_{(\frac{N}{4})} \right) \tag{2.5}$$

Figure 2.5 shows robustness, provided by the interquartile range. As the interquartile range only uses the first and the third quartile of the data for estimating the scale its performance is practically independent of κ and we can achieve good results, even if ϵ is high.



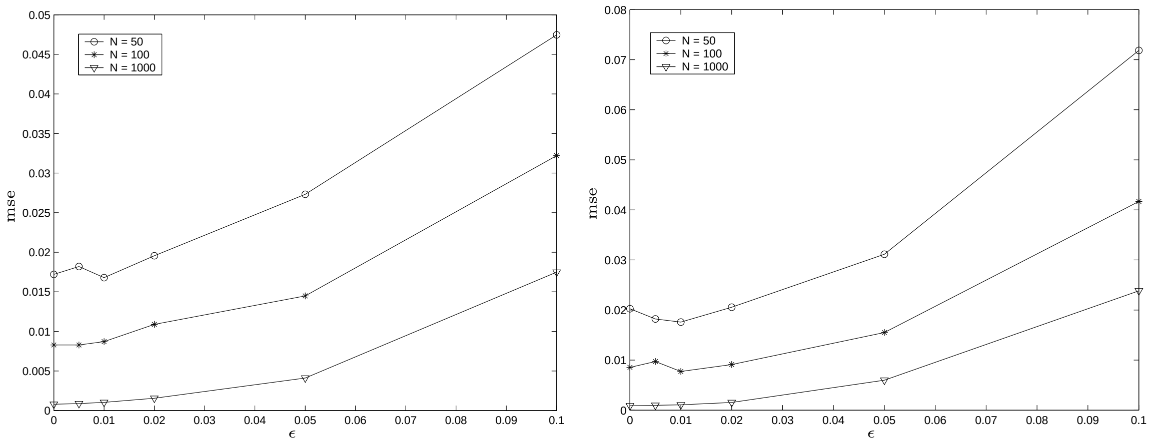
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.2: Performance of the α-trimmed standard deviation, $\alpha = 0.02$



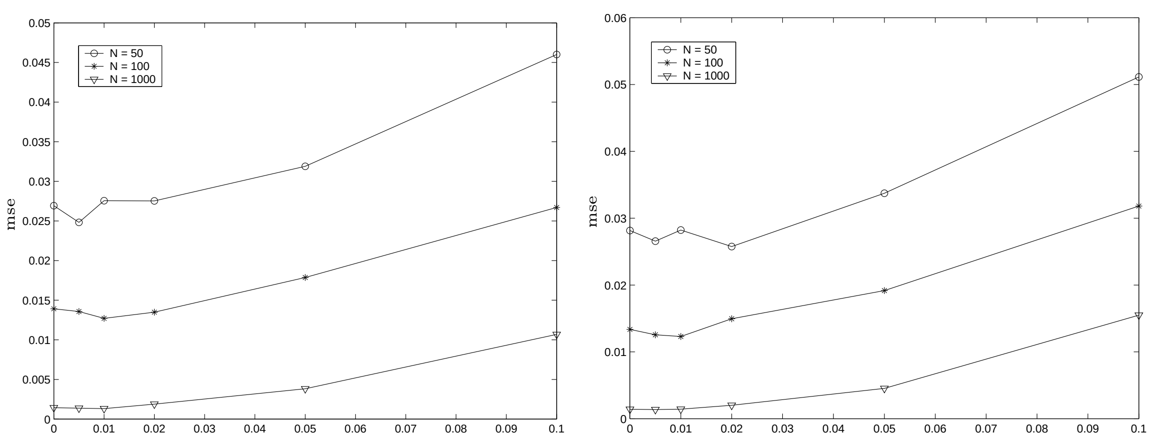
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.3: Performance of the α-trimmed standard deviation, $\alpha = 0.05$



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.4: Performance of the α-trimmed standard deviation, $\alpha = 0.1$



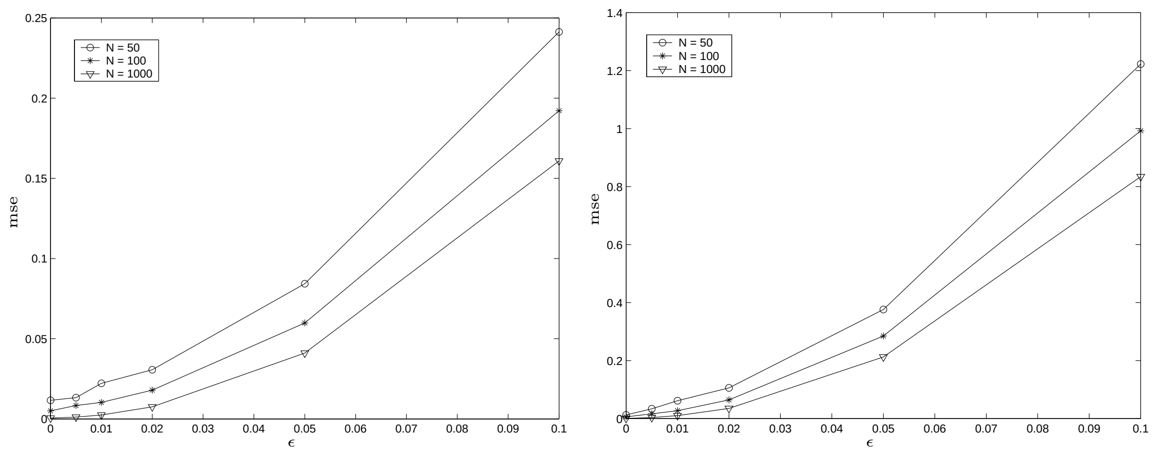
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.5: Performance of the interquartile range

2.1.4 The mean absolute deviation

The mean absolute deviation is defined as follows:

$$\hat{\sigma} = c \cdot \frac{1}{N} \sum_{n=1}^{N} |y_n - \bar{y}| \tag{2.6}$$



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.6: Performance of the mean absolute deviation

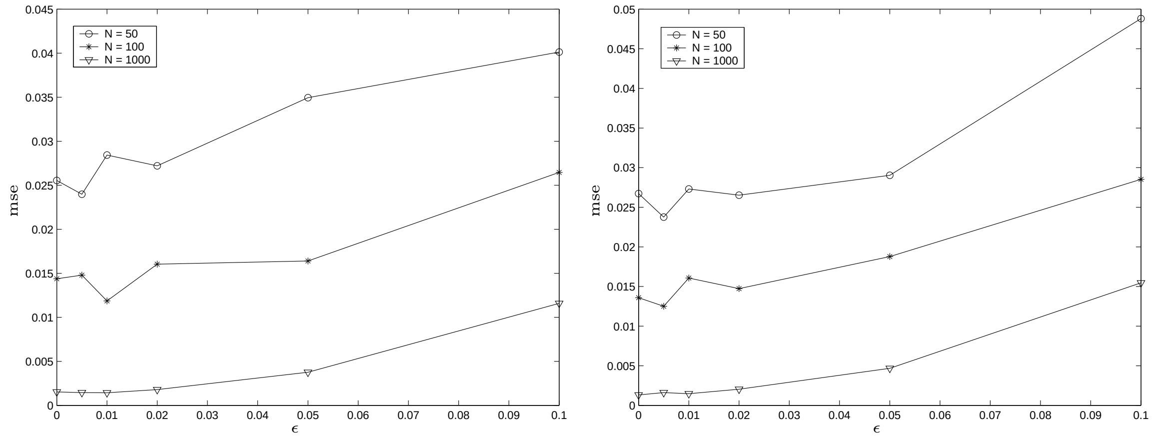
As one can see in the above pictures, the mean absolute deviation is less sensitive to outliers as the sample standard deviation, but still gives poor estimates, being highly dependent on κ.

2.1.5 The median absolute deviation

As the median is insensitive to outliers, the median absolute deviation is an often applied scale estimator:

$$\hat{\sigma} = c \cdot median(|y_n - median(y_n)|) \tag{2.7}$$

The simulations are showing the good performance of the median absolute deviation in the presence of outliers. It has a low mean square error and, as the median itself is unaffected by the actual size of an outlier, no significant difference between $\kappa = 25$ and $\kappa = 100$ can be seen (see Figure 2.7).



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.7: Performance of the median absolute deviation

2.1.6 The static M-Estimator

Clipped quadratic score function

Huber [1] recommended the use of the M-Estimator for scale, defined as the solution of the implicit equation

$$\sum_{n=1}^{N} \chi(\frac{y_n}{\hat{\sigma}}) = 0 \tag{2.8}$$

where $\chi(x)$ is the score function. It is shown in [1] that the minimax-solution for Gaussian mixture models is the clipped quadratic (cq) score function, defined as

$$\chi_{cq}(x; x_1) = \begin{cases} x^2 - \beta & : \quad |x| \leq x_1 \\ {x_1}^2 - \beta & : \quad |x| > x_1 \end{cases} \tag{2.9}$$

where x_1 is the clipping point, according to

$$2\int_0^{x_1} \phi(x)dx + \frac{2x_1\phi(x_1)}{{x_1}^2 - 1} = \frac{1}{1-\epsilon}. \tag{2.10}$$

with $\phi(x)$ being the standard Gaussian pdf.
Note, that in equation (2.9), the offset β is chosen as

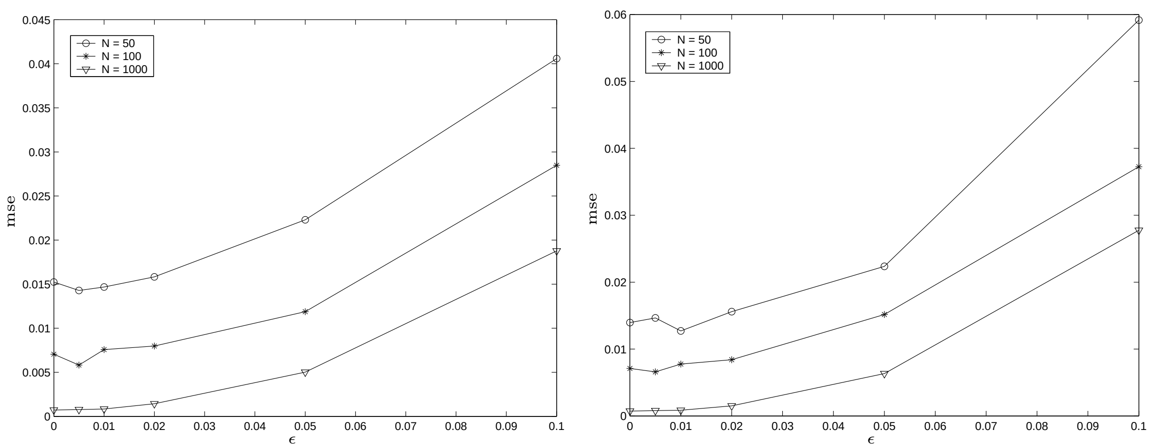
$$\beta = 1 - 2x_1\phi(x_1) + 2(1 - {x_1}^2)\left(\int_{-\infty}^{x_1} \phi(x)dx - 1\right), \tag{2.11}$$

such that the M-Estimator is unbiased for a nominal Gaussian distribution.
Due to equation (2.10), the best value for x_1 can only be achieved when ϵ, the degree of contamination, is known. Table 2.2 shows some solutions of equation (2.10), using different values for ϵ. (A fixed background nominal distribution with $\sigma_N = 1$ is assumed here.)

ϵ	x_1	ϵ	x_1
0	∞	0.026	2.0
0.004	2.5	0.05	1.81
0.005	2.46	0.10	1.62
0.01	2.27	0.15	1.5
0.02	2.07	0.20	1.41

Table 2.2: Calculation of some values of the clipping point x_1

The following figures 2.8, 2.9 and 2.10 now show the behavior of some M-Estimators for scale with clipped quadratic (cq) score functions in the presence of outliers.



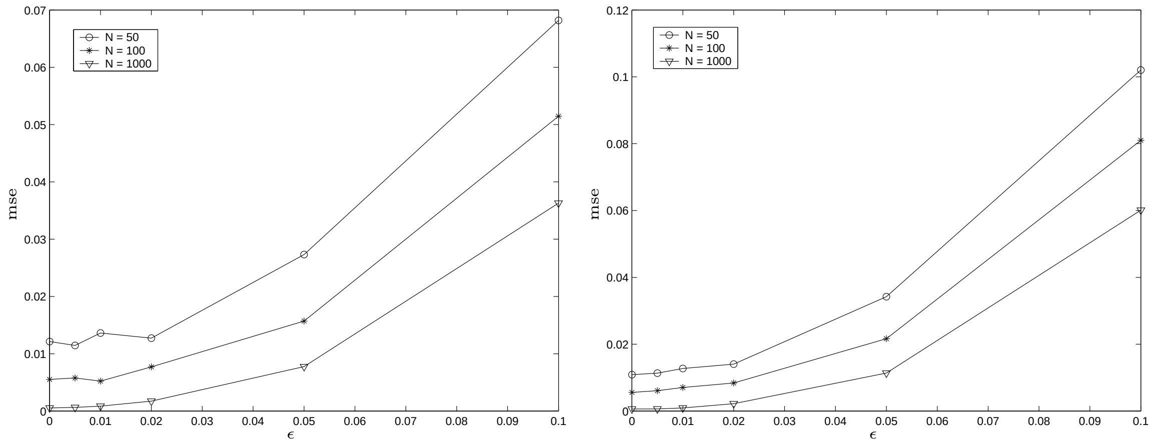
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.8: Performance of the static M-Estimator (cq), $x_1 = 1.5$

It is easy to see, that the performance of the static M-Estimator with clipped quadratic score functions is highly dependent on the choice of the clipping point x_1. If ϵ is small, we can achieve best results by choosing a high value x_1 (e.g. $\epsilon = 0.005$, $\Rightarrow x_1 = 2.46$). When ϵ increases, we need to cut off more of the data, as it is more contaminated (e.g. $\epsilon = 0.1$, $\Rightarrow x_1 = 1.5$).

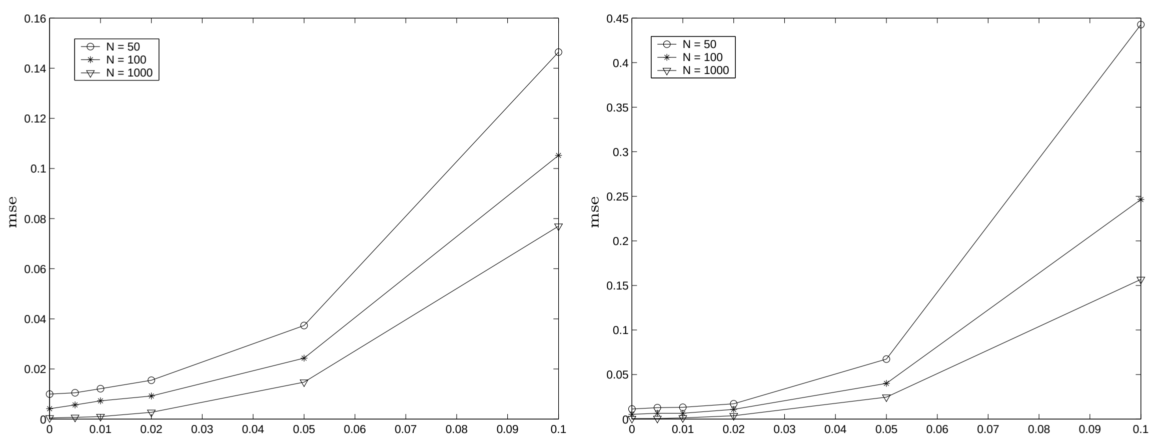
Smoothed Huber score function

The choice of clipped quadratic score-functions in the previous section is somehow unnatural. In practice, one cannot simply define an exact point, which distinguishes between the background noise and the contaminating noise. This is why we will now propose a new score function, which approximates the clipped quadratic one, but offers a smoother transition in the region of the clipping point x_1. We start with the



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.9: Performance of the static M-Estimator (cq), $x_1 = 2.0$



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.10: Performance of the static M-Estimator (cq), $x_1 = 2.5$

following function:

$$\chi_{sh}(x) = a \cdot e^{-\frac{x^2}{b}} + c \tag{2.12}$$

Ignoring the bias correcting term β for the time being, we want $\chi_{sh}(x)$ to approximate $\chi_{cq}(x) = min(x^2, x_1^2)$. Fulfilling the requirements

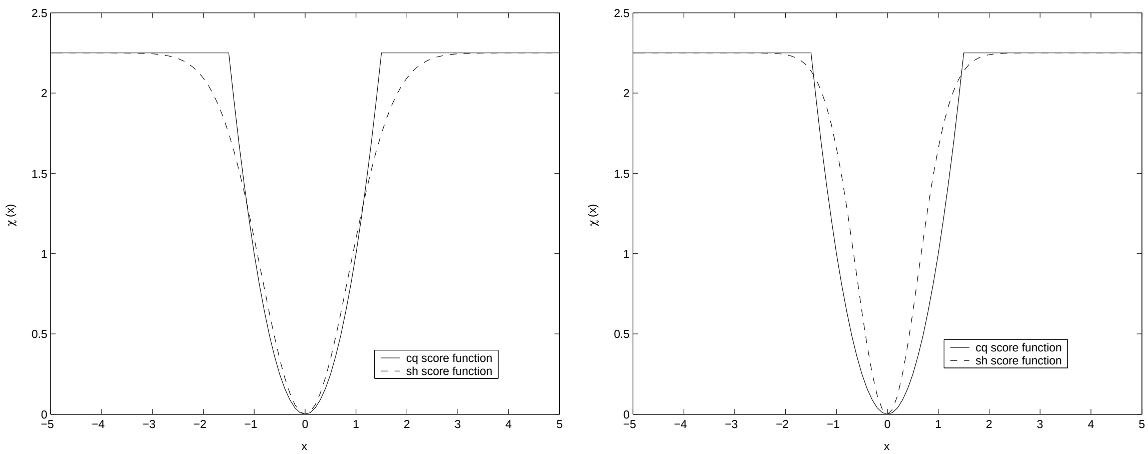
$$\chi_{sh}(0) = 0 \tag{2.13}$$

$$\lim_{x \to \pm\infty} \chi_{sh}(x) = x_1^2 \tag{2.14}$$

leads us to

$$\chi_{sh}(x) = -x_1^2 e^{-\frac{x^2}{b}} + x_1^2, \tag{2.15}$$

whereby the remaining width parameter b can be chosen by the user.



(a) $b = x_1$ (b) $b = \frac{x_1}{2}$

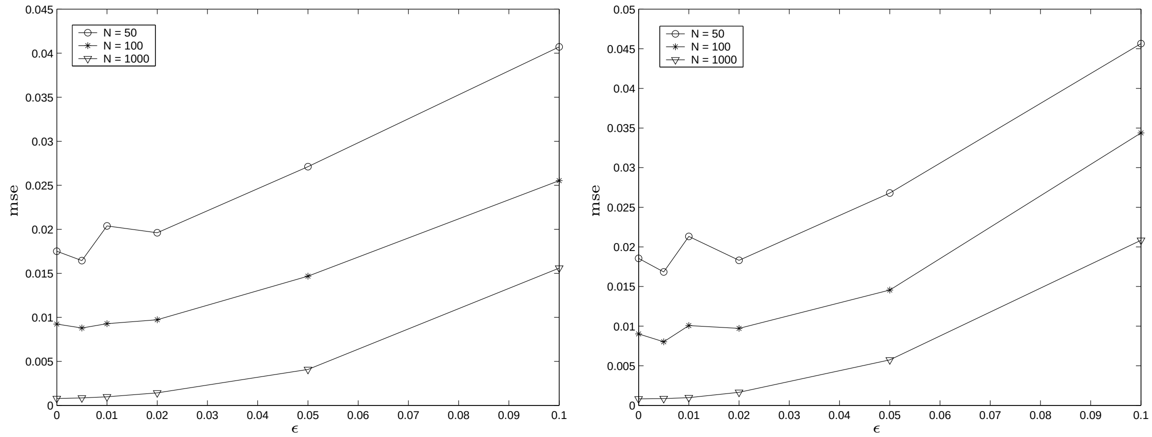
Figure 2.11: Plots of the clipped quadratic and the smoothed Huber score functions

Figure 2.11 shows the plots of the clipped quadratic and the smoothed Huber score functions with $x_1 = 1.5$ and no bias correcting term considered.
As the M-Estimator with the smoothed Huber score function as presented in (2.15) is generally biased for the nominal Gaussian distribution, we have to introduce a correction factor δ, such that solving (2.8) with

$$\chi_{sh}(x) = -x_1^2 e^{-\frac{x^2}{b}} + x_1^2 - \delta \tag{2.16}$$

will produce an unbiased scale estimator $\hat{\sigma}$ for $\mathcal{N}(0, 1)$. Considering the functional form of a M-Estimator for scale as discussed in [1]

$$\int_{-\infty}^{\infty} \chi\left(\frac{x}{\sigma}\right) f(x)dx = 0 \tag{2.17}$$



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.12: Performance of the static M-Estimator (sh), $x_1 = 1.5$

where $f(x)$ is the assumed underlying probability density function, we will now determine the value for δ, such that our estimator is unbiased for $\mathcal{N}$ (0,1).

$$\int_{-\infty}^{\infty} \chi_g(\frac{x}{1})\varphi(x) = 0 \tag{2.18}$$

$$\int_{-\infty}^{\infty} (-x_1^2 e^{-\frac{x^2}{b}} + x_1^2 - \delta)\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\, dx = 0 \tag{2.19}$$

$$(x_1^2 - \delta)\underbrace{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\, dx}_{=1} - x_1^2 \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2\frac{b}{b+2}}}\, dx = 0 \tag{2.20}$$

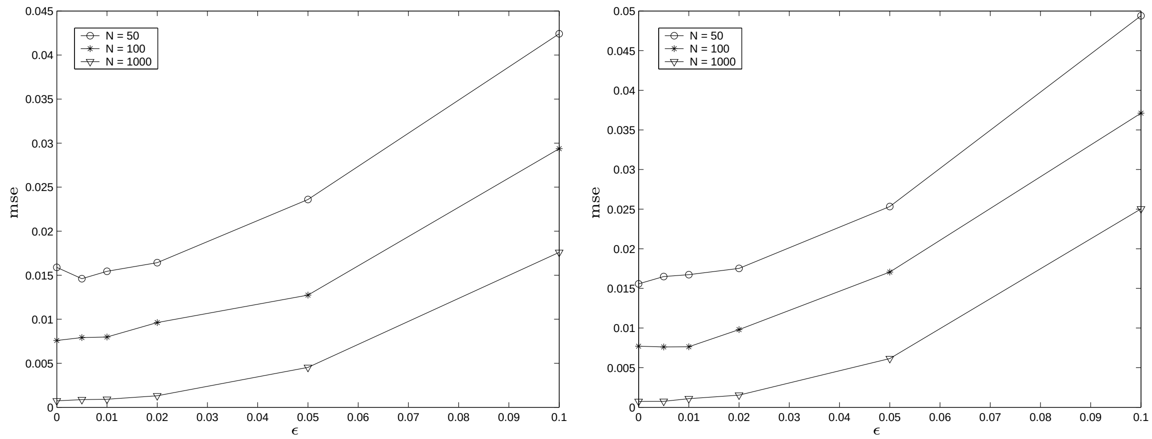
$$x_1^2 - \delta - x_1^2\sqrt{\frac{b}{b+2}}\underbrace{\int_{-\infty}^{\infty} \frac{1}{\sqrt{\frac{b}{b+2}}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2\frac{b}{b+2}}}\, dx}_{=1} = 0 \tag{2.21}$$

$$\Rightarrow \delta = x_1^2\left(1 - \sqrt{\frac{b}{b+2}}\right) \tag{2.22}$$

Simulation results have shown, that $b = \frac{x_1}{2}$ is a good choice, which reduces equation (2.22) to

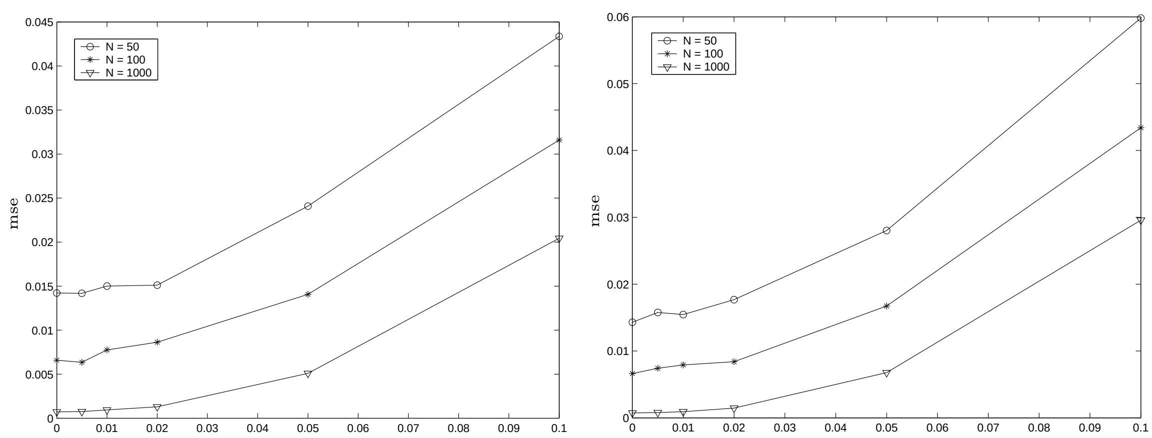
$$\delta = x_1^2\left(1 - \sqrt{\frac{x_1}{x_1+4}}\right) \tag{2.23}$$

The following plots show the performance of the static M-Estimator with a smoothed Huber score function, for $x_1 = 1.5$, 2.0 and 2.5 and $b = \frac{x_1}{2}$.



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.13: Performance of the static M-Estimator (sh), $x_1 = 2.0$



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.14: Performance of the static M-Estimator (sh), $x_1 = 2.5$

κ	N	ϵ	cq	sh		κ	N	ϵ	cq	sh
25	50	0	15	17		100	50	0	14	18
		0.01	15	20				0.01	13	21
		0.05	23	27				0.05	22	27
		0.1	41	41				0.1	59	46
25	100	0	7	9.2		100	100	0	7	9
		0.01	7.6	9.3				0.01	8	10
		0.05	12	15				0.05	15	15
		0.1	29	26				0.1	37	34
25	1000	0	0.7	0.8		100	1000	0	0.7	0.8
		0.01	0.8	1				0.01	0.9	1
		0.05	5	4				0.05	6.3	5.7
		0.1	19	16				0.1	28	21

Table 2.3: Comparing the clipped quadratic with the smoothed Huber score function for the static M-Estimator MSE$\cdot 10^3$, $x_1 = 1.5$

The tables 2.3, 2.4 and 2.5 compare the mean square error of the M-Estimator with a clipped quadratic score function and the one with a smoothed Huber score function for $N \in \{50, 100, 1000\}$, $\epsilon \in \{0, 0.01, 0.05, 0.1\}$, $\kappa \in \{25, 100\}$ and $x_1 = 1.5$, 2.0 and 2.5. Note that the cq-column shows the mean square error ($\cdot$ 1000) of the M-Estimator with a clipped quadratic score function, while the sh-column corresponds to the mean square error ($\cdot$ 1000) of the M-Estimator with the smoothed Huber score function.

The results of the simulations with smoothed Huber score functions are encouraging. We can achieve an excellent performance, almost independent of the parameter x_1, which makes the estimator more robust, compared to the M-Estimator with clipped quadratic score functions. To show this, let us consider Table 2.4, which compares the two M-Estimators for $x_1 = 2.0$. If N is small (e.g. $N = 50$), we can observe in the first row, that our proposed smoothed Huber score function provides a higher mean square error than the clipped quadratic score function for small ϵ. As ϵ increases, the smoothed Huber score function shows a better performance. This trade-off gets smaller and finally vanishes for $N = 100$ and $N = 1000$, where the smoothed Huber score function gives better results for all ϵ.
Another big advantage of the smoothed Huber score function is the reduced complexity: Solving equation 2.8, for example by applying the Newton-Raphson method, takes less time to converge if $\chi(x)$ is not a piecewise defined function, as it the case for the clipped quadratic score function.

κ	N	ϵ	cq	sh		κ	N	ϵ	cq	sh
25	50	0	13	16		100	50	0	11	15
		0.01	15	15				0.01	12	17
		0.05	28	23				0.05	33	24
		0.1	68	42				0.1	102	49
25	100	0	7.3	7.5		100	100	0	8	8
		0.01	7.3	7.6				0.01	8.6	8.2
		0.05	13	12				0.05	21	16
		0.1	52	29				0.1	77	37
25	1000	0	0.9	0.8		100	1000	0	0.9	1
		0.01	1.1	0.9				0.01	1.3	1.1
		0.05	7.3	4.5				0.05	11	5
		0.1	37	18				0.1	59	25

Table 2.4: Comparing the clipped quadratic with the smoothed Huber score function for the static M-Estimator MSE$\cdot 10^3$, $x_1 = 2.0$

κ	N	ϵ	cq	sh		κ	N	ϵ	cq	sh
25	50	0	11	14		100	50	0	16	15
		0.01	13	15				0.01	18	16
		0.05	39	24				0.05	59	28
		0.1	147	43				0.1	448	60
25	100	0	6	7		100	100	0	10	7
		0.01	9	8				0.01	12	8
		0.05	23	13				0.05	40	16
		0.1	105	32				0.1	245	43
25	1000	0	1	0.9		100	1000	0	0.9	1
		0.01	1.8	1				0.01	1.3	1.2
		0.05	14	5				0.05	27	6
		0.1	78	20				0.1	153	30

Table 2.5: Comparing the clipped quadratic with the smoothed Huber score function for the static M-Estimator MSE$\cdot 10^3$, $x_1 = 2.5$

2.1.7 The adaptive M-Estimator

Clipped quadratic score function

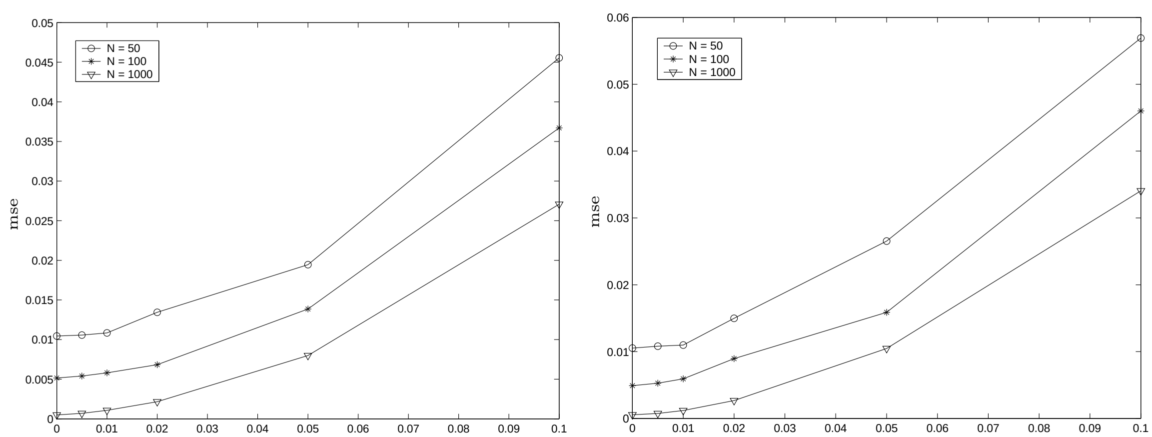
The adaptive M-Estimator, proposed in [2] attempts to address the problem that a minimax-solution can only be optimal when the right clipping point is used. This can be a problem in practice, as the values for ϵ and κ may change in time or are unknown. The adaptive M-Estimator, as the name suggests, adapts itself to changes in the distribution and models the score function $\chi(x)$ as a sum of weighted basis functions, which may be just a collection of clipped quadratic functions, with different values for x_1 or other score functions, optimized for distributions, deviating from the Gaussian mixture model, defined in (2.2). The score function $\chi(x)$ can then be obtained by

$$\chi(x) = \sum_{q=1}^{Q} a_q g_q(x) = \mathbf{a}^{\mathbf{T}}\mathbf{g}(x) \tag{2.24}$$

with $\mathbf{g}(x) = [g_1(x), g_2(x), ..., g_q(x)]^T$ being the vector of the q chosen bases functions, weighted by $\mathbf{a} = [a_1, a_2, ..., a_q]^T$. The vector of weightings $\mathbf{a}$ can be calculated by

$$\mathbf{a} = \left(\sum_{n=1}^{N} \mathbf{g}(\hat{x}_n)\mathbf{g}^{\mathbf{T}}(\hat{x}_n)\right)^{-1} \sum_{n=1}^{N} \hat{x}_n \dot{\mathbf{g}}(\hat{x}_n) \tag{2.25}$$

with $\dot{\mathbf{g}} = [\dot{g}_1, \dot{g}_2, ..., \dot{g}_q]^T$ denoting the vector of the derivatives of $\mathbf{g}$ and $x_n = \frac{y_n}{\sigma_i}$ is the scaled observations with σ_i being the scale estimate in the ith step of convergence. By solving (2.8) with $\chi(x)$ defined as in (2.24) one can compute a next scale estimate $\hat{\sigma}_{i+1}$ until the estimate reaches convergence. For details, see [2]. For the simulations shown here, $\chi(x)$ is modeled as a collection of three clipped quadratic functions, defined by (2.9), with $x_1 = 1.5$, 2.0 and 2.5.

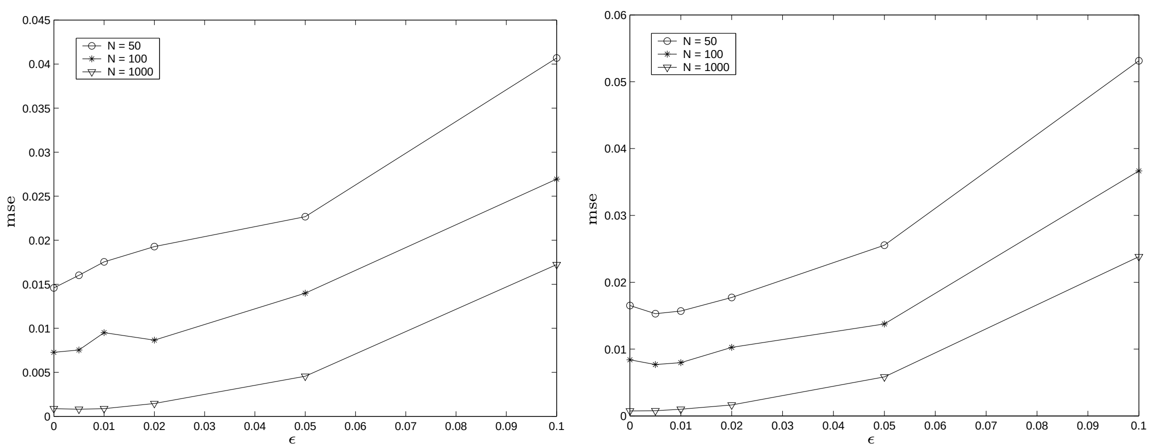


(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.15: Performance of the adaptive M-Estimator (cq)

Smoothed Huber score function

As shown in section 2.1.6, the smoothed Huber score functions exhibit better performance for high values of ϵ and reduced computation time when compared to the clipped quadratic score functions. Now we will use this model for adaptive M-Estimators and design our basis functions as presented in equation (2.15). To compare our approach with the adaptive M-Estimator with clipped quadratic functions, as seen in the previous section, we will again choose three basis-functions, each of them modeled as a smoothed Huber score function with parameters $x_1 = 1.5$, 2.0 and 2.5.



(a) $\kappa = 25$ (b) $\kappa = 100$

Figure 2.16: Performance of the adaptive M-Estimator (sh)

Table 2.6 compares the performance of the adaptive M-Estimator with clipped quadratic score functions (cq) and the one with smoothed Huber score functions (sh) by showing the respective mean square error ($\cdot$ 1000).

As one can see, the gain in performance for the adaptive M-Estimators is not as big as for the static M-Estimators. This is most probably due to the following fact: The adaptive M-Estimator tries to create the best combination of different score functions and, by doing so, it reduces the dependence of the parameter x_1 for the clipped quadratic score functions. Applying the same scheme to the smoothed Huber score functions won't change much, because, as seen in section 2.1.6, they show much less dependence on x_1.

The main advantage of the adaptive M-Estimator is the possibility to use collections of basis-functions reflecting the level of knowledge available about the noise distribution. This ensures quality, even if the Gaussian mixture model does not fit reality. This shows itself through the results that point out that the adaptive M-Estimator is usually not the best solution for a specific underlying distribution, but below the proposed estimators in this chapter it is the most reliable one.

κ	N	ϵ	cq	sh	κ	N	ϵ	cq	sh
25	50	0	10	15	100	50	0	11	16
		0.01	11	17			0.01	11	16
		0.05	19	22			0.05	27	25
		0.1	46	41			0.1	57	53
25	100	0	5	7	100	100	0	5	8
		0.01	6	9			0.01	6	8
		0.05	14	14			0.05	16	14
		0.1	37	27			0.1	46	36
25	1000	0	0.5	0.9	100	1000	0	0.5	0.7
		0.01	1	0.9			0.01	1.2	1
		0.05	8	4.5			0.05	10.5	5.8
		0.1	27	17.2			0.1	34	24

Table 2.6: Comparing the clipped quadratic with the smoothed Huber score function for the adaptive M-Estimator, MSE$\cdot 10^3$

2.2 Review about complexity

Until now, we have only looked at the mean square error of the different scale estimators. In practice it is often the case that it is not only the statistical properties of the estimate, but also the calculation-time which is of interest. This is especially important as the adaptive M-Estimators have a far higher computational complexity than other scale estimators mentioned so far. To quantify the "quality" of an estimator not only for a specific κ, N or ϵ, we define $q(T_n)$ as being the overall measure of quality of an estimator $\hat{\sigma}$:

$$q(\hat{\sigma}) = \sum_{\epsilon} \sum_{\kappa} mse(\hat{\sigma}; \epsilon, \kappa) \tag{2.26}$$

whereby $mse_{\hat{\sigma}}(T_n, \epsilon, \kappa)$ is the mean square error of the scale estimate $\hat{\sigma}$ for fixed ϵ, κ, estimator T_n and sample size $N = 1000$. As before $\epsilon \in \{0, 0.005, 0.01, 0.02, 0.05, 0.1\}$ and $\kappa \in \{25, 100\}$.
We define the computation time $comp(\hat{\sigma})$ of an estimator T_n as

$$comp(\hat{\sigma}) = \frac{cputime(\hat{\sigma})}{cputime(\hat{\sigma}_{std})} \tag{2.27}$$

whereby $cputime(\hat{\sigma})$ is the computation time of the respective estimator $\hat{\sigma}$, calculated by the Matlab-function "cputime" and $\hat{\sigma}_{std}$ being the sample standard deviation. Figure 2.18 finally shows the performance of the estimators represented graphically by a $(q(\hat{\sigma}), comp(\hat{\sigma}))$-pair.
Note that these measures are very rough. The quality of an estimator and its computation time are dependent on the particular parameters chosen and on the implementation of the estimator. $q(\hat{\sigma})$ and $comp(\hat{\sigma})$ are chosen to give an overall indicator

No.	Estimator	$q(\hat{\sigma})$	$comp(\hat{\sigma})$
1	Static M-Estimator (cq), $x_1 = 1.5$	0.15	3.81
2	Adaptive M-Estimator (sh)	0.16	23.64
3	Static M-Estimator (sh), $x_1 = 2.0$	0.16	2.71
4	Adaptive M-Estimator (cq)	0.16	37.07
5	Static M-Estimator (sh) $x_1 = 1.5$	0.16	2.62
6	Static M-Estimator (sh) $x_1 = 2.5$	0.17	2.64
7	α-trimmed, $\alpha = 0.10$	0.18	1.11
8	Interquartile range	0.20	1.38
9	Median absolute deviation	0.21	1.2
10	Static M-Estimator (cq), $x_1 = 2.0$	0.22	4.37
11	α-trimmed, $\alpha = 0.05$	0.31	1.11
12	Static M-Estimator (cq), $x_1 = 2.5$	0.47	5.22
13	Mean absolute deviation	1.68	1.21
14	α-trimmed, $\alpha = 0.02$	1.74	1.11
15	Standard deviation	10.2	1

Table 2.7: Calculation time vs. mean square error of all scale estimator

of quality and computation time, which then can be compared to other estimators. The actual values of these measures hereby are not as important as their relationship to each other.
Considering Table 2.7, we can observe the following:

- There is a trade-off between speed and accuracy. Reliable estimators with excellent robust behavior, like the adaptive M-Estimator, are quite complex, while fast estimators as the sample standard deviation are presenting a poor performance in the presence of outliers.
- As already observed in section 2.1.6, the M-Estimators with smoothed Huber score functions (sh) are nearly independent of the clipping point x_1.

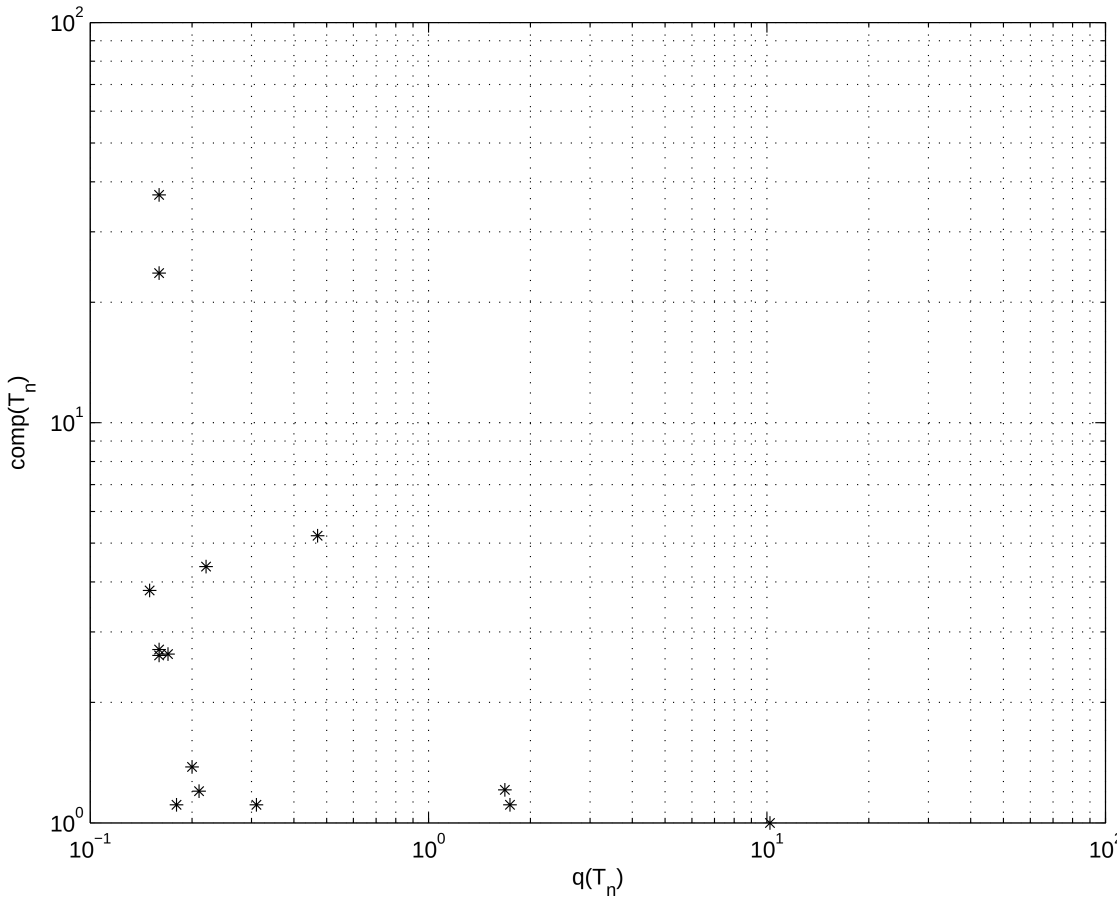


Figure 2.17: Calculation time vs. mean square error of all scale estimator

Chapter 3

Joint Estimation of Location and Scale

In the last chapter, we implicitly assumed scale estimation to be a self-standing process, independent from its environment. However, in practice, scale estimation is almost never the last step in our calculation and it often interacts with other estimators and procedures. We might not be interested in the scale parameter σ explicitly, but we will use it in the process of estimating the parameter of interest, which in the most simple case could be a location parameter.
This chapter deals with joint estimation of location and scale. In section 3.1 we will give a short overview of the signal model and the M-Estimator for location which will be used in this chapter. Section 3.2 will then introduce virtual estimators which are used to study the effect which an error in the scale estimate will have on the location estimate. In section 3.3 the scale estimators introduced in chapter 2 are implemented in the scenario of joint estimation of location and scale, while section 3.4 gives a conclusion of the work done in this chapter.

3.1 Signal model

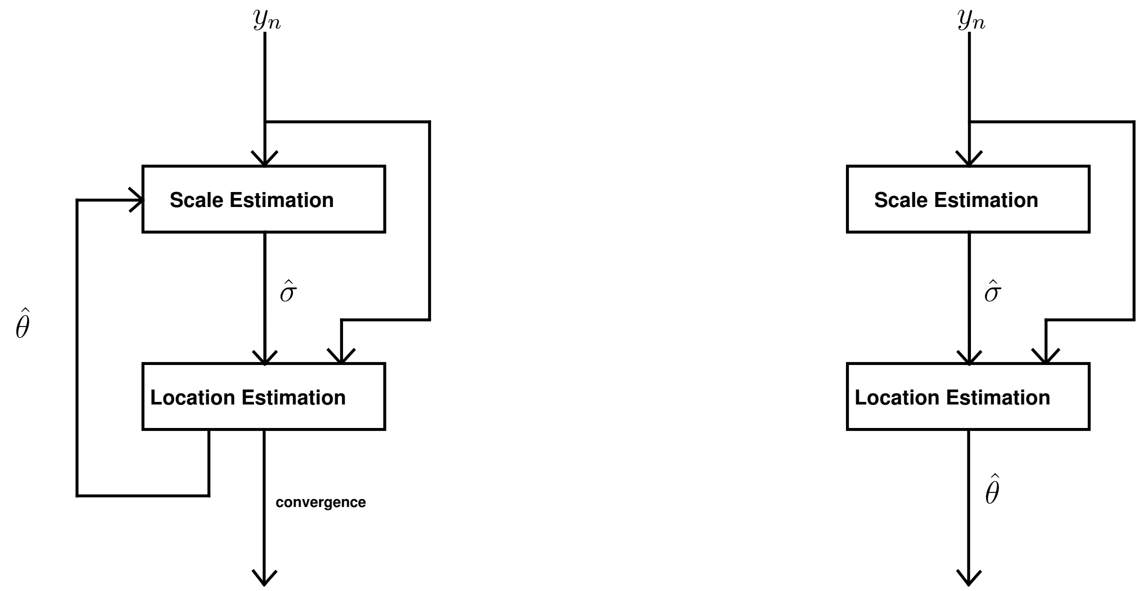
The signal model for this chapter is a parametrized signal, embedded in noise:

$$y_n = s_n(\theta) + \sigma u_n \tag{3.1}$$

where u_n is distributed as follows:

$$u_n \sim (1-\epsilon)\mathcal{N}(0,1) + \epsilon\mathcal{N}(0,\kappa). \tag{3.2}$$

The joint estimation of the location parameter θ and scale parameter σ could be performed within the framework of a scheme such as that, shown in figure 3.1(a).



(a) General Scheme (b) Simplified Scheme

Figure 3.1: Scheme of the joint estimation of location and scale

First, a scale estimation of the data is done, leading to $\hat{\sigma}_0$. This scale estimate is then used by a location estimator to produce a first estimate of the location parameter, $\hat{\theta}_0$. The location estimate is then passed back to the scale estimator. The joint estimation of location and scale is completed, when convergence is reached, e.g. $|\hat{\theta}_{i+1} - \hat{\theta}_i| < \delta$ for a fixed $\delta > 0$.

As a location estimator, we will use the M-Estimator for location, proposed by Huber [1]. It works in a similar manner to the M-Estimator for scale, introduced in chapter 2. The implicit equation, which is derived from the maximum likelihood-estimator is as follows:

$$\sum_{n=1}^{N} \psi\left(\frac{y_n - s_n(\theta)}{\sigma}\right) \frac{d}{d\theta} s_n(\theta) = 0 \tag{3.3}$$

where $s_n(\theta)$ is the parameterized signal. In our scenario, we consider $s_n(\theta) = \theta$ (a DC-value, embedded in noise), which simplifies equation (3.3) to

$$\sum_{n=1}^{N} \psi\left(\frac{y_n - \theta}{\sigma}\right) = 0. \tag{3.4}$$

$\psi(x)$ is Huber's score function for location (minimax solution for Gaussian mixture models), denoted by

$$\psi(x) = \begin{cases} \frac{x}{\sigma^2} & : \quad |x| \le x_1 \\ x_1 \cdot sgn(x) & : \quad |x| > x_1. \end{cases} \tag{3.5}$$

where x_1 is the clipping point, dependent on σ and ϵ, due to

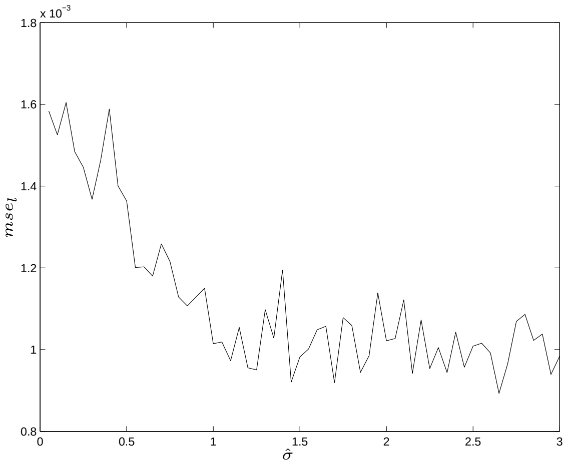
$$\frac{2\phi(x_1\sigma)}{x_1\sigma} - 2\Phi(-k\sigma) = \frac{\epsilon}{1-\epsilon} \tag{3.6}$$

with $\phi(x)$ and $\Phi(x)$ denoting the standard Gaussian pdf and cdf respectively.

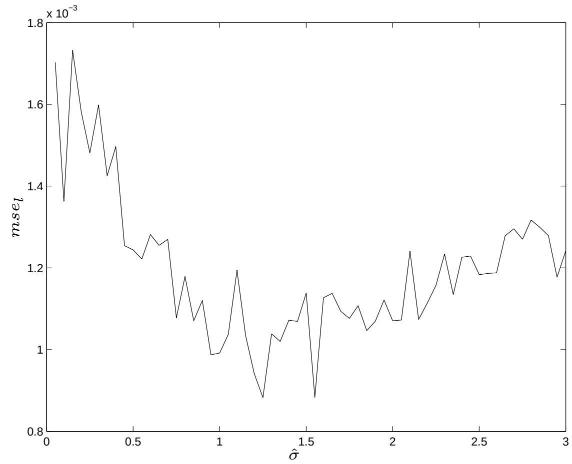
Having a scale estimate $\hat{\sigma}$, equation (3.6) is still not solvable without the knowledge of ϵ (assuming minimax optimality). To get rid of this, we will fix the value for ϵ to 0.05. Equation (3.6) can then be solved by using $\hat{\sigma}$.

3.2 Virtual estimators of scale

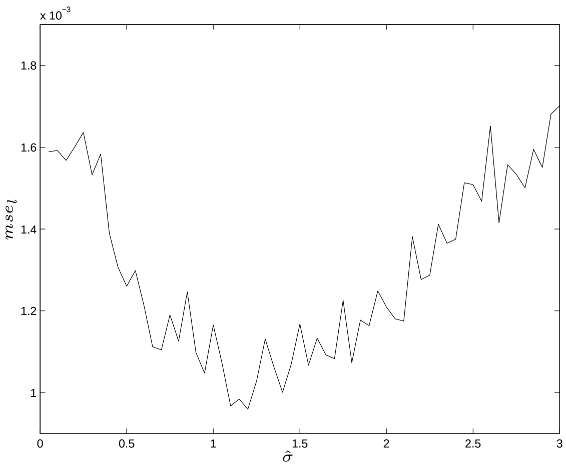
The aim in this section is to explore how a scale estimate, or more precisely an error in the scale estimate, will affect on the location estimate. Before using "real" scale estimators, we study this effect with virtual scale estimators, which are predefined values used as being estimates $\hat{\sigma}$ of the real scale σ.
We will therefore change the scheme of joint estimation of scale and location to the simplified one, which can be seen in Figure 3.1(b). Here the feedback of the location estimator is unnecessary and, hence, is omitted.
Simulations are now done by the following scheme: In the signal model 3.1, we set $\sigma = 1$ with $N = 1000$ samples, using the static M-Estimator for location as presented in section 3.1. The noise is modeled as a two term Gaussian mixture (equation (3.2)) with $\kappa = 100$ and $\epsilon \in \{0, 0.005, 0.01, 0.02, 0.05, 0.1\}$. We now vary the scale estimate and observe the effect on the mean square error of the location estimate, mse_l. Figure 3.2 shows the plots for different degrees of contamination and $\hat{\sigma} \in (0, 3]$. Results have been averaged over 500 Monte Carlo simulations.
These plots are quite intuitive: Best results are obtained when $\hat{\sigma}$ is close to $\sigma = 1$. Deviations from the true scale σ will increase the mean square error of the location estimate. The most interesting observation in these plots is the following: The mean square error is not changing much in the range $0.75 < \hat{\sigma} < 1.25$, which practically means, that we can accept an error of $\pm 25\%$ in the scale estimate without visibly increasing the mean square error of the location estimate. Note also that the error increases faster for big values of ϵ.



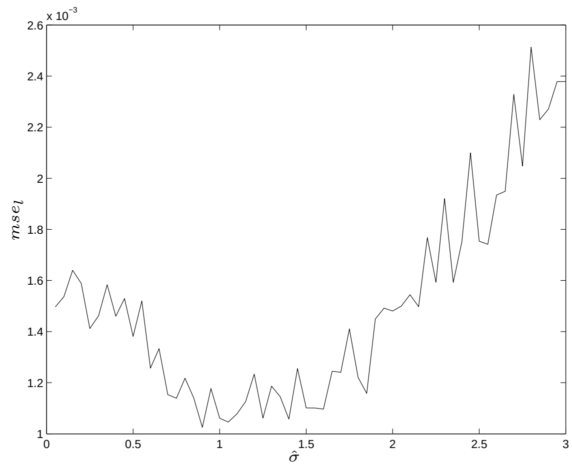
(a) $\epsilon = 0$



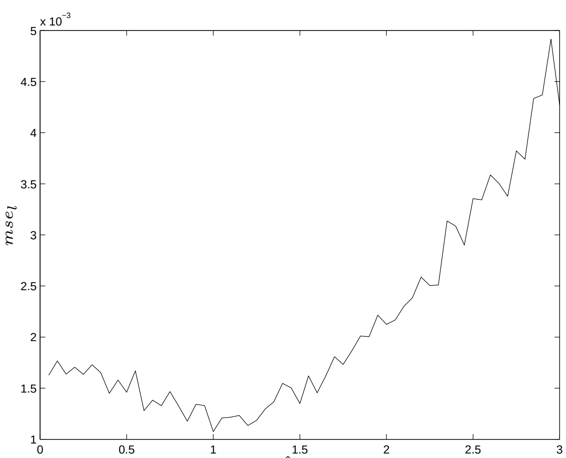
(b) $\epsilon = 0.005$



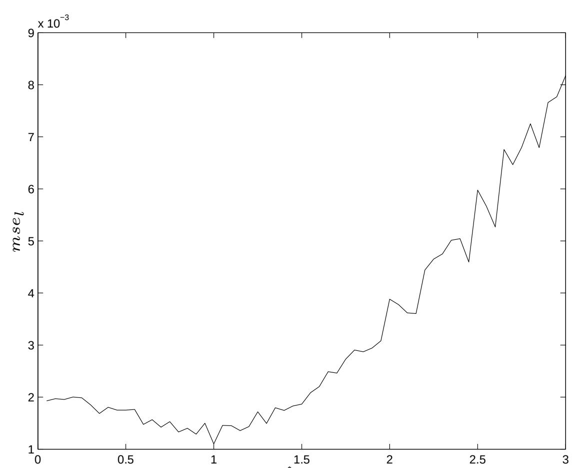
(c) $\epsilon = 0.01$



(d) $\epsilon = 0.02$



(e) $\epsilon = 0.05$



(f) $\epsilon = 0.01$

Figure 3.2: mse_l as a function of $\hat{\sigma}$

3.3 Estimators of scale

Having seen the effects of the virtual scale estimators on the location estimate, we will now explore, how the “real” scale estimators, introduced in section 2.1 behave in the scenario of joint estimation of location and scale. Like in the previous section, our signal model will be

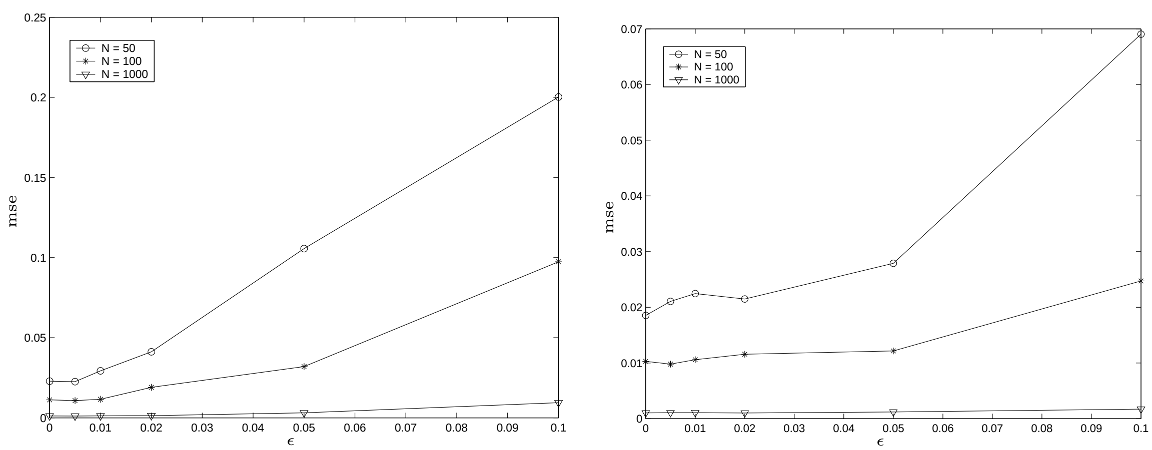
$$y_n = \theta + \sigma u_n \tag{3.7}$$

with θ being a dc-value, buried in scaled noise u_n, which is distributed as

$$u_n \sim (1-\epsilon)\mathcal{N}(0,1) + \epsilon\mathcal{N}(0,\kappa) \tag{3.8}$$

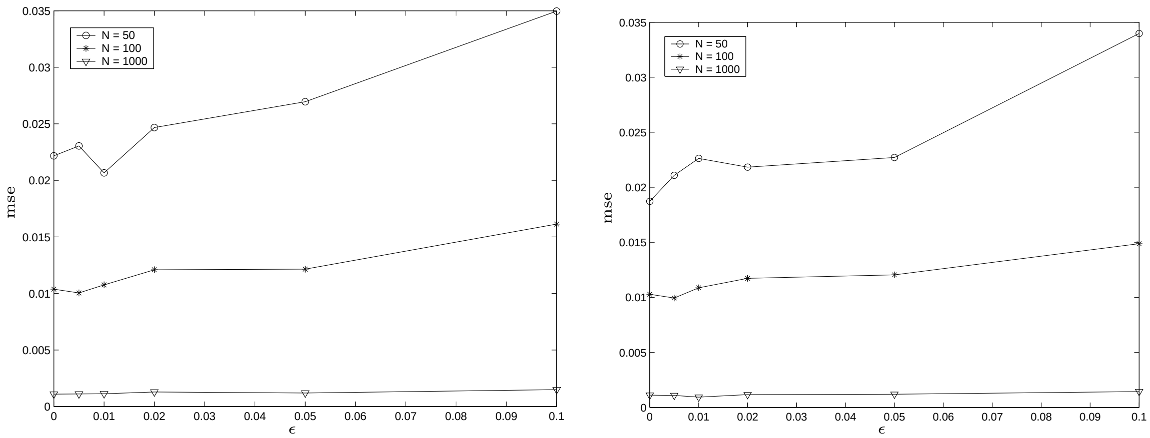
Again, we will use the M-Estimator for location, presented in section 3.1.
On the following pages, the simulation results with $\epsilon \in \{0, 0.005, 0.01, 0.02, 0.05, 0.1\}$, $\kappa \in \{25, 100\}$, $N \in \{50, 100, 1000\}$, $\sigma = 1$ and $\theta = 1$ are shown. The mean square error will be used as an index of quality. For space reasons, we restrict ourselves to only some representative scale estimators. Simulation results for the remaining scale estimators can be found in the Appendix.

3.3.1 Simulations



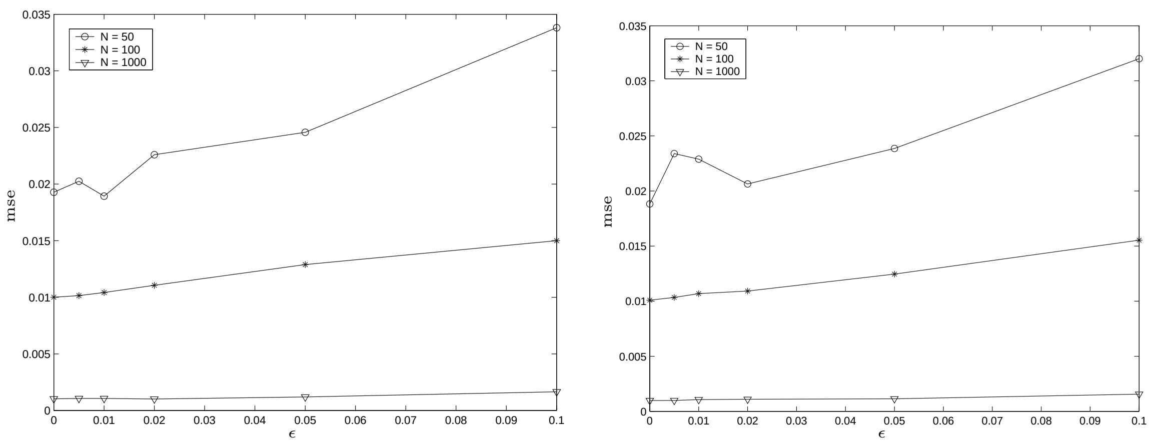
(a) Standard deviation (b) α-trimmed standard deviation, $\alpha = 0.05$

Figure 3.3: Performance of the sample and the α-trimmed standard deviation



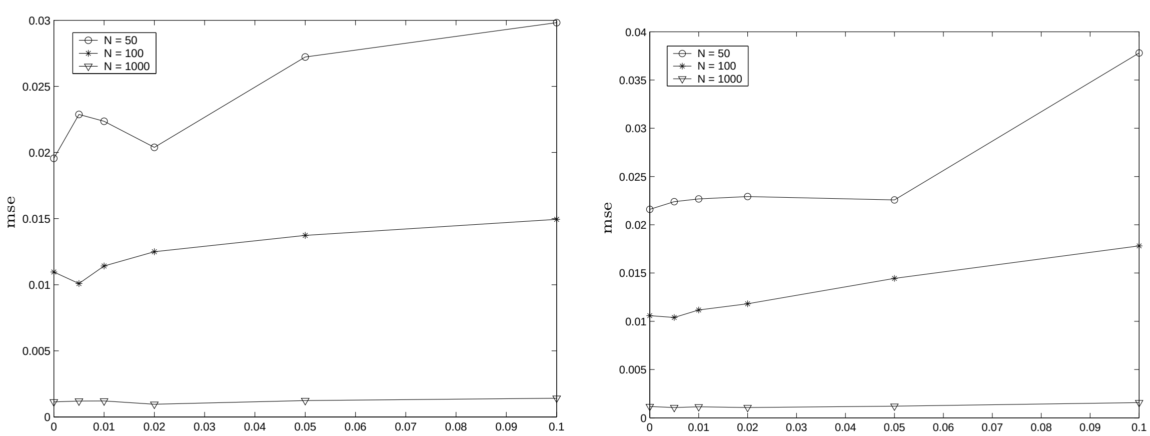
(a) Median absolute deviation (b) Interquartile range

Figure 3.4: Performance of the median absolute deviation and the interquartile range



(a) cq (b) sh

Figure 3.5: Performance of the static M-Estimator, cq and sh, $x_1 = 1.5$



(a) Cq (b) $\kappa = 100$

Figure 3.6: Performance of the adaptive M-Estimator, cq and sh

3.4 Conclusion

As one can see, there is not much difference in the effects, different scale estimator have on the location estimate. The following figures 3.7, 3.8 and 3.9 summarize this effect for three different scale estimators: The static M-Estimator, the adaptive M-Estimator and the median absolute deviation. It is easy to see, that it would not make much sense to choose a complex estimator like the adaptive M-Estimator for scale. When we are using a robust location estimator, like Huber's M-Estimator, to estimate a DC-value, robust estimation of scale becomes less important. A fast estimator like the median absolute deviation would be sufficient, independent of N and ϵ.

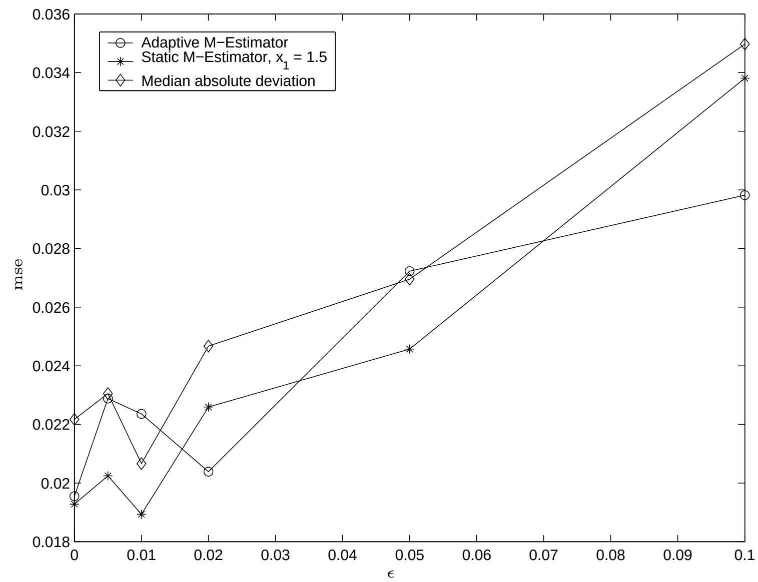


Figure 3.7: Final comparison, $N = 50$

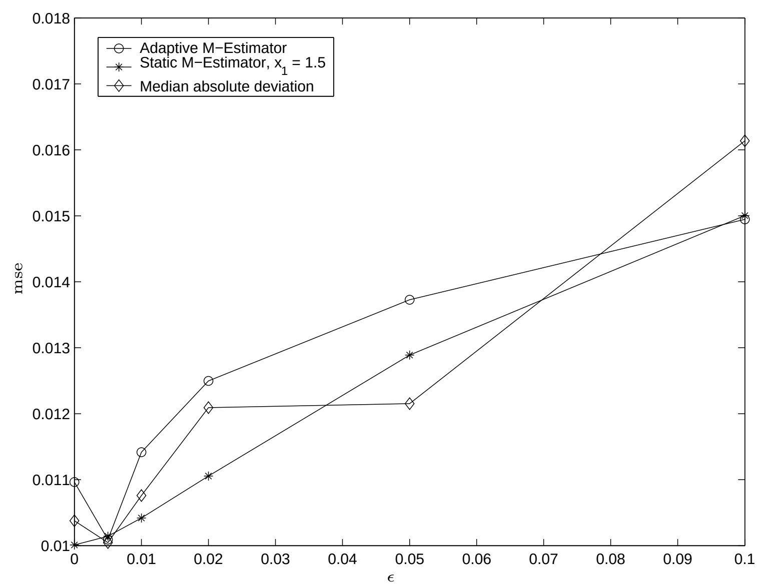


Figure 3.8: Final comparison, $N = 100$

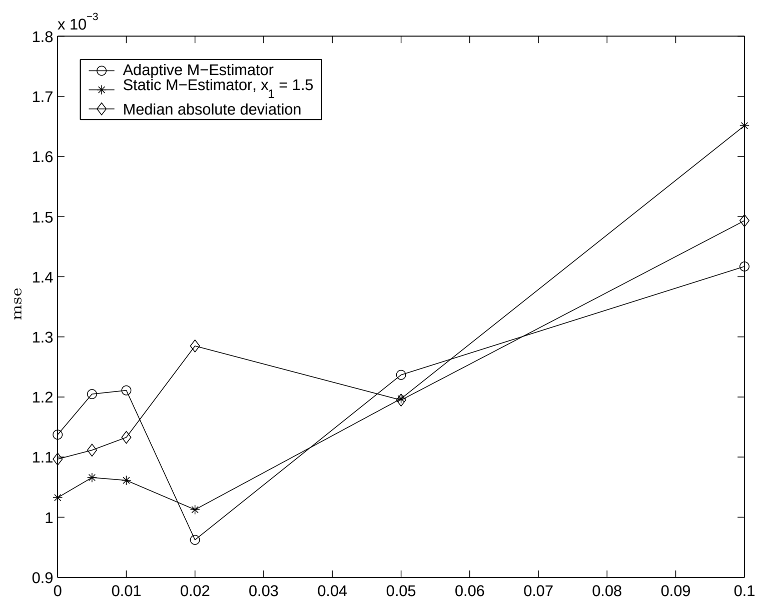


Figure 3.9: Final comparison, $N = 1000$

Chapter 4

Robust Covariance Estimation

Estimation of covariance matrices is an important issue in multiuser detection, as it is the first step in estimating the number of sources or their directions of arrival (DOA). It has been shown [3] that common approaches like the MUSIC-algorithm for DOA-Estimation and the MDL-principle for estimating the number of sources are quite sensitive to deviations from an assumption of Gaussian sources. Especially in a wireless communication scenario sources are better described as impulsive, deviating more or less from Gaussianity due to electromagnetic interference or natural phenomenon. This motivates a robust estimation of covariance matrices.
In section 4.1 we will introduce the signal model and some methods for estimating covariance matrices. Some representative simulation results are found in section 4.2. In section 4.3 a new approach in estimating covariance matrices is discussed, based on the idea of Visuri, Oja and Koivunen [3] to use the marginal variances of whitened observations as estimates of the eigenvalues of the covariance matrix. Simulation results for this are found in section 4.4, while section 4.5 gives a conclusion of the work done in this chapter.

4.1 Covariance Estimators

4.1.1 Signal model

Consider the following signal model

$$\mathbf{x}(n) = A\mathbf{u}(n) \tag{4.1}$$

where $n = 1, 2, ..., N$, $\mathbf{x}(n) = [x_1(n), x_2(n), ..., x_M(n)]^T$ is the snapshot matrix, $\mathbf{u}(n) = [u_1(n), u_2(n), ...u_P(n)]^T$ is a vector of independent and identically distributed random variables and A is the $M \times P$ mixing matrix. This corresponds to a scenario with P users and M sensors. The true underlying covariance matrix can then be determined

as $C = E[\mathbf{x}\mathbf{x}^\mathbf{T}] = AA^T$.
Note that we will restrict ourselves to real valued random variables only.

4.1.2 Elementwise estimation of covariance matrices

One possibility to estimate the covariance matrix is to estimate each single matrix element. Huber [1] suggested to use the following operation:

$$\hat{C}(j,k) = \frac{\hat{\sigma}^2(x_j + x_k) - \hat{\sigma}^2(x_j - x_k)}{\hat{\sigma}^2(x_j + y_k) + \hat{\sigma}^2(x_j - x_k)}\hat{\sigma}(x_j)\hat{\sigma}(x_k) \tag{4.2}$$

with $\hat{\sigma}(\cdot)$ being a robust scale estimator as presented in section 2.1 and x_j denoting the jth row of the snapshot matrix $\mathbf{x} = [x(1), x(2), ..., x(N)]$.

4.1.3 Sign covariance matrix

Another method has been proposed by [3], in which the sample spatial sign covariance matrix (SCM) is used as an estimator for C. The algorithm is as follows:
We first apply the spatial sign function $\mathbf{S}(\mathbf{x})$, a generalization of the univariate sign function to each row x_i of the snapshot matrix $\mathbf{x}$:

$$\mathbf{S}(\mathbf{x_i}) = \begin{cases} \frac{\mathbf{x_i}}{||\mathbf{x_i}||} & : \ \mathbf{x_i} \neq \mathbf{0} \\ \mathbf{0} & : \ \mathbf{x_i} = \mathbf{0} \end{cases} \tag{4.3}$$

where $||\mathbf{x_i}|| = (\mathbf{x_i^T}\mathbf{x_i})^{\frac{1}{2}}$.
After that, the SCM can be computed as follows:

$$\hat{C}_{SCM} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{S}(\mathbf{x_i})\mathbf{S^T}(\mathbf{x_i}) \tag{4.4}$$

4.1.4 FLOM based estimation of covariance matrices

The last proposed method for covariance estimation in this chapter is based on fractional lower order moments (FLOM). FLOMs are used to estimate the covariation of α-stable processes [6], where second-order moments (for $\alpha \neq 2$) do not exist. With $0 < p < \alpha$, the covariation matrix can be estimated by applying

$$\hat{C}_{FLOM}(j,k;p) = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x_j}(\mathbf{n})|\mathbf{x_k}(\mathbf{n})|^{p-2}\mathbf{x_k^T}(n) \tag{4.5}$$

FLOMs were motivated for α-stable distributions, but we use them for the Gaussian mixture model because of their excellent behavior in the presence of outliers (see

[7]). Since $\hat{C}_{FLOM}$ is generally not symmetric, it is averaged with its transpose, as proposed in [7]:

$$\hat{C} = \frac{1}{2}(\hat{C}_{FLOM} + \hat{C}_{FLOM}^T) \tag{4.6}$$

4.1.5 Metrics

In the scenario of estimating a scale parameter σ, as in chapter 2, the mean square error is a natural, and expressive metric to quantify the quality of an estimator. Therefore, one way to measure the quality of a covariance matrix estimate can be found by the Frobenius norm.

$$L_F(\hat{C}) = \sum_{i=1}^{N}\sum_{j=1}^{N}(\hat{C}(i,j) - C(i,j))^2 \tag{4.7}$$

In addition to the Frobenius norm, we will use the relative mean square error of the eigenvalues as another metric:

$$L_E(\hat{C}) = \sum_{i=1}^{N}\left(\frac{\lambda_{i,\hat{C}} - \lambda_{i,C}}{\lambda_{i,C}}\right)^2 \tag{4.8}$$

This last metric is especially interesting, when estimating the number of sources by applying the MDL-principle, where only the eigenvalues of the covariance matrix are used.

4.2 Simulations

This section follows closely the work done in [8]. We will therefore restrict ourselves to some representative simulation results, referring to [8] for more details.
Considering the signal model in equation (4.1), we will set $P = M = 4$ and $N = 100$ (4 sources and 4 sensors, 100 samples) and model $u(n)$ as follows:

$$u(n) \sim (1-\epsilon)\mathcal{N}(0,1) + \epsilon\mathcal{N}(0,100) \tag{4.9}$$

with $\epsilon \in \{0, 0.005, 0.01, 0.02, 0.05, 0.1\}$. As candidates for the mixing matrix A, we choose the following ones:

$$\tilde{A}_1(i,k) = \begin{pmatrix} 1 & 0.8^1 & 0.8^2 & 0.8^3 \\ 0.8^1 & 1 & 0.8^1 & 0.8^2 \\ 0.8^2 & 0.8^1 & 1 & 0.8^1 \\ 0.8^3 & 0.8^2 & 0.8^1 & 1 \end{pmatrix} \tag{4.10}$$

$$\tilde{A}_2(i,k) = \begin{pmatrix} 1 & 0.2^1 & 0.2^2 & 0.2^3 \\ 0.2^1 & 1 & 0.2^1 & 0.2^2 \\ 0.2^2 & 0.2^1 & 1 & 0.2^1 \\ 0.2^3 & 0.2^2 & 0.2^1 & 1 \end{pmatrix} \tag{4.11}$$

$$\tilde{A}_3(i,k) = \begin{pmatrix} 1 & 0.25 & 0.25 & 0.25 \\ 0.25 & 1 & 0.25 & 0.25 \\ 0.25 & 0.25 & 1 & 0.25 \\ 0.25 & 0.25 & 0.25 & 1 \end{pmatrix} \tag{4.12}$$

$$\tilde{A}_4(i,k) = \begin{pmatrix} 1 & -0.75 & 0.5 & -0.25 \\ -0.75 & 1 & -0.75 & 0.5 \\ 0.5 & -0.75 & 1 & -0.75 \\ -0.25 & 0.5 & -0.75 & 1 \end{pmatrix} \tag{4.13}$$

which then have been standardized due to

$$A(i,k) = \frac{\tilde{A}(i,k)}{\sqrt{\sum_{j=1}^{P} \tilde{A}^2(i,j)}}. \tag{4.14}$$

Simulation results are found by averaging over 500 Monte Carlo simulations.

Tables 4.1 and 4.2 show the performance of different estimators, using the Frobenius norm for the A_1 and the A_2 mixing matrices:
We can observe the following:

- The sample covariance matrix soon breaks down when ϵ increases.
- The SCM is insensitive to ϵ but provides a constant high error, even for the Gaussian case.
- Like in the pure scale estimation problem, the smoothed Huber score function shows much better performance for higher values of ϵ at the price of a higher mean square error in the nominal case.
- The errors for the A_1 mixing matrix (which simulates a higher dependence between the sources) are much higher, which could cause troubles especially for the SCM, where the mean square error is increased by approximately 77% compared to the A_2 mixing matrix.

	ϵ					
Estimator	0	0.005	0.01	0.02	0.05	0.1
Sample	0.44	1.95	3.97	7.11	19.0	38.5
MAD	0.76	0.79	0.85	1.00	1.88	4.87
Static M-Est. 1.5 cq	0.56	0.58	0.66	0.96	2.75	8.52
Static M-Est. 1.5 sh	0.61	0.65	0.70	0.98	2.30	6.29
Static M-Est. 2.0 cq	0.47	0.56	0.75	1.21	4.27	16.1
Static M-Est. 2.0 sh	0.56	0.65	0.72	0.98	2.45	7.20
Static M-Est. 2.5 cq	0.46	0.57	0.90	1.78	7.32	24.1
Static M-Est. 2.5 sh	0.55	0.64	0.70	1.01	2.71	8.15
Adaptive M-Est. cq	0.49	0.54	0.75	1.02	2.84	8.17
Adaptive M-Est. sh	0.56	0.62	0.74	0.96	2.28	6.07
FLOM 1.0	0.78	0.57	0.44	0.47	1.68	3.93
FLOM 1.5	0.61	0.65	1.03	2.25	6.16	12.7
FLOM 1.8	0.47	1.31	2.29	4.79	12.4	24.6
SCM	3.03	3.03	3.02	3.02	3.01	3.00

Table 4.1: Frobenius norm for the A_1 mixing matrix

	ϵ					
Estimator	0	0.005	0.01	0.02	0.05	0.1
Sample	0.45	1.79	3.05	5.54	12.4	23.6
MAD	0.71	0.73	0.73	0.78	0.99	1.54
Static M-Est. 1.5 cq	0.52	0.53	0.57	0.62	0.97	1.98
Static M-Est. 1.5 sh	0.57	0.59	0.61	0.67	0.96	1.75
Static M-Est. 2.0 cq	0.46	0.49	0.53	0.64	1.18	2.91
Static M-Est. 2.0 sh	0.54	0.56	0.59	0.63	1.00	1.90
Static M-Est. 2.5 cq	0.45	0.49	0.54	0.71	1.68	4.85
Static M-Est. 2.5 sh	0.53	0.53	0.56	0.63	1.03	2.05
Adaptive M-Est. cq	0.45	0.49	0.52	0.64	1.07	2.03
Adaptive M-Est. sh	0.45	0.49	0.52	0.64	1.07	2.03
FLOM 1.0	0.56	0.52	0.51	0.58	1.17	2.39
FLOM 1.5	0.47	0.63	0.91	1.54	3.45	6.86
FLOM 1.8	0.42	1.16	1.85	3.36	7.54	14.1
SCM	1.71	1.70	1.70	1.70	1.70	1.70

Table 4.2: Frobenius norm for the A_2 mixing matrix

The following tables 4.3 and 4.4 are showing the performance of the estimators for the A_1 and A_2 mixing matrices when using the relative mean square error of the eigenvalues as a metric:

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.08	2.01	6.32	17.4	105	409
MAD	1.28	1.18	1.44	1.60	2.36	8.92
Static M-Est. 1.5 cq	0.18	0.19	0.26	0.48	2.88	23.2
Static M-Est. 1.5 sh	0.28	0.31	0.37	0.52	2.09	13.5
Static M-Est. 2.0 cq	0.11	0.15	0.28	0.71	6.68	75.5
Static M-Est. 2.0 sh	0.22	0.26	0.33	0.51	2.41	16.7
Static M-Est. 2.5 cq	0.09	0.16	0.37	1.39	18.6	158
Static M-Est. 2.5 sh	0.19	0.22	0.26	0.51	3.01	22.4
Adaptive M-Est. cq	0.10	0.16	0.26	0.51	3.06	22.2
Adaptive M-Est. sh	0.23	0.28	0.31	0.50	2.16	12.4
FLOM 1.0	0.68	0.75	0.71	0.75	1.21	2.62
FLOM 1.5	0.18	0.18	0.27	0.78	4.62	20.2
FLOM 1.8	0.10	0.47	1.25	4.55	28.7	117
SCM	2.64	2.66	2.51	2.40	2.01	1.50

Table 4.3: Relative MSE of the eigenvalues for the A_1 mixing matrix

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.08	2.18	6.02	18.7	97.4	376
MAD	0.22	0.22	0.23	0.24	0.43	1.27
Static M-Est. 1.5 cq	0.10	0.11	0.12	0.16	0.50	2.52
Static M-Est. 1.5 sh	0.13	0.13	0.14	0.18	0.47	1.82
Static M-Est. 2.0 cq	0.08	0.09	0.12	0.19	0.83	5.56
Static M-Est. 2.0 sh	0.11	0.12	0.14	0.16	0.51	2.27
Static M-Est. 2.5 cq	0.08	0.09	0.13	0.26	1.92	15.7
Static M-Est. 2.5 sh	0.11	0.10	0.12	0.16	0.58	2.61
Adaptive M-Est. cq	0.08	0.09	0.11	0.18	0.64	2.61
Adaptive M-Est. sh	0.11	0.12	0.13	0.17	0.45	1.93
FLOM 1.0	0.24	0.22	0.23	0.29	0.77	2.52
FLOM 1.5	0.15	0.17	0.36	1.11	5.89	24.3
FLOM 1.8	0.09	0.77	1.82	6.31	33.6	123
SCM	2.16	2.16	2.16	2.16	2.17	2.17

Table 4.4: Relative MSE of the eigenvalues for the A_2 mixing matrix

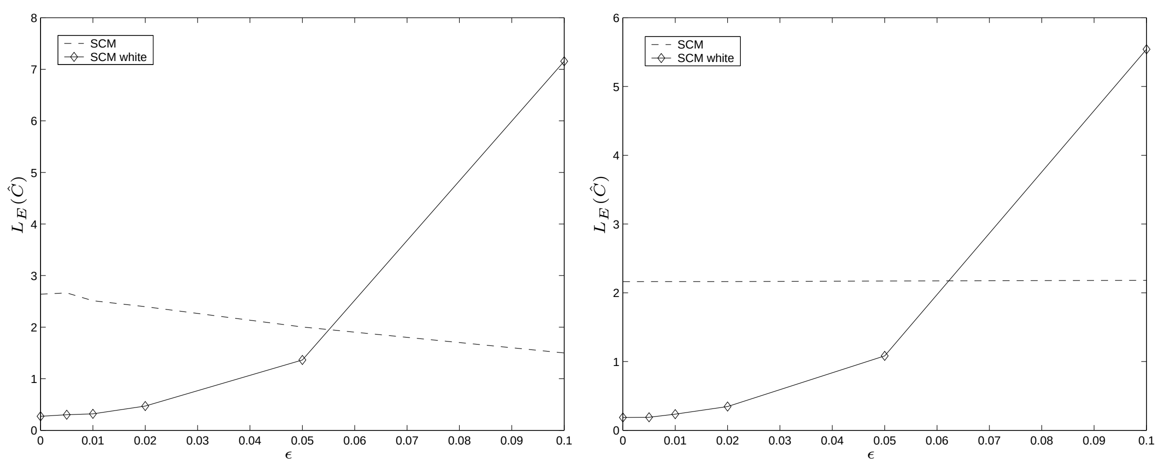
The results are quite similar: We can observe a constant error of the SCM-method, practically independent of ϵ. Again, the errors are higher when using the A_1 mixing matrix.

4.3 Whitening scheme

In [3] it was shown that the performance of the sample sign covariance matrix (SCM) can be increased for small sample sizes by applying the following whitening scheme:

1. Calculate the SCM as presented in section 4.1.3
2. Let $\hat{\mathbf{\Gamma}}$ be the matrix of eigenvectors of the SCM.
3. Calculate the transformed observations $\tilde{\mathbf{x}} = \hat{\mathbf{\Gamma}}^{\mathbf{T}}\mathbf{x}$
4. Let the robust marginal variances (using the median absolute deviation) of $\tilde{\mathbf{x}}$ be estimates for the eigenvalues λ_i, $i = 1, ..., M$.

The effect of this scheme on the estimation of the eigenvalues λ_i of the covariance matrix can be seen in the following figure 4.1.



(a) A_1 mixing matrix (b) A_2 mixing matrix

Figure 4.1: The effect of the whitened SCM

As we can see the performance changed a lot. Instead of having a practically constant (and high) error for all ϵ we now observe a much better performance for small degrees of contamination ($\epsilon \leq 0.06$), while losing performance at a higher degree of contamination.

We remark that the scheme of calculating the whitened SCM (Step 2 to 4)

just uses a given estimated covariance matrix (the SCM), but it is independent of the SCM-method itself. This motivates the application of this whitening scheme to all of the other covariance matrix estimators and observe the effect of it.
Tables 4.5 and 4.6 now show this effect on the relative mean square error of the eigenvalues for the A_1 and A_2 mixing matrix and should be compared with tables 4.3 and 4.4.
Let us first examine the whitening effect on the A_1 mixing matrix which simulates a relative high dependence. The results are encouraging, showing a highly robust behavior for every single estimator. Again we slightly increase the error for the nominal case, which is compensated by a much better behavior for $\epsilon \geq 0.01$.
Now examining the A_2 mixing matrix which simulates a smaller dependence shows a different behavior: There is a gain in performance for the sample covariance matrix and consequently for the FLOMs of higher order. For most of the other estimators and scenarios the performance is decreasing.
An interesting observation is the following: When applying the whitening scheme the differences between the estimators get small. Especially when comparing the different rows of table 4.6 one can see that the difference between the different estimators is vanishing. It seems as if the median absolute deviation as an estimator for the marginal variances is dominating the procedure.
However it seems as if the whitening scheme generally is only useful for scenarios with high dependence between the sources. We will therefore apply the scheme to the A_3 (low dependence) and A_4 (high dependence) mixing matrix and observe if this will strengthen our theory. Table 4.7 and 4.8 are comparing the results for the A_3, Table 4.9 and 4.10 for the A_4 mixing matrix.

Again we can observe the same behavior in performance: We lose a lot of performance by applying the whitening scheme in the scenario with the A_3 mixing matrix, especially for the M-Estimators (static and adaptive). In the scenario with the A_4 mixing matrix there is a gain in performance for almost all estimators and degrees of contamination. In some cases even the performance in the nominal case is improved here.

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.23	0.34	0.48	0.81	1.84	7.32
MAD	0.72	0.79	0.93	1.11	2.59	9.35
Static M-Est. 1.5 cq	0.23	0.24	0.29	0.43	1.41	7.29
Static M-Est. 1.5 sh	0.26	0.28	0.32	0.47	1.52	7.10
Static M-Est. 2.0 cq	0.22	0.23	0.26	0.40	1.56	8.04
Static M-Est. 2.0 sh	0.24	0.28	0.33	0.46	1.48	7.41
Static M-Est. 2.5 cq	0.21	0.23	0.27	0.40	1.77	8.17
Static M-Est. 2.5 sh	0.22	0.26	0.29	0.44	1.38	7.49
Adaptive M-Est. cq	0.22	0.22	0.28	0.41	1.44	7.36
Adaptive M-Est. sh	0.23	0.27	0.32	0.46	1.43	6.91
FLOM 1.0	0.40	0.44	0.48	0.70	1.88	9.06
FLOM 1.5	0.21	0.24	0.28	0.42	1.50	7.08
FLOM 1.8	0.22	0.28	0.37	0.55	1.76	7.20
SCM	0.27	0.30	0.32	0.47	1.36	7.15

Table 4.5: Relative MSE of the eigenvalues for the A_1 mixing matrix (whitened)

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.19	0.22	0.27	0.36	1.07	5.04
MAD	0.19	0.20	0.22	0.33	1.09	5.20
Static M-Est. 1.5 cq	0.19	0.20	0.21	0.32	1.05	5.78
Static M-Est. 1.5 g	0.18	0.20	0.22	0.33	1.06	5.19
Static M-Est. 2.0 cq	0.19	0.20	0.21	0.29	1.14	5.72
Static M-Est. 2.0 g	0.18	0.20	0.21	0.30	1.07	5.76
Static M-Est. 2.5 cq	0.19	0.19	0.22	0.30	1.03	4.95
Static M-Est. 2.5 g	0.18	0.19	0.23	0.32	1.07	5.27
Adaptive M-Est. cq	0.18	0.21	0.23	0.33	1.04	5.20
Adaptive M-Est. g	0.19	0.20	0.22	0.35	1.02	5.34
FLOM 1.0	0.19	0.19	0.22	0.33	1.10	5.95
FLOM 1.5	0.19	0.21	0.23	0.32	1.12	6.17
FLOM 1.8	0.19	0.21	0.24	0.38	1.10	5.37
SCM	0.19	0.19	0.23	0.34	1.08	5.54

Table 4.6: Relative MSE of the eigenvalues for the A_2 mixing matrix (whitened)

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.14	2.70	8.44	24.3	119	445
MAD	0.41	0.43	0.45	0.53	1.06	3.14
Static M-Est. 1.5 cq	0.19	0.21	0.25	0.34	1.04	5.95
Static M-Est. 1.5 sh	0.25	0.27	0.29	0.38	0.95	4.06
Static M-Est. 2.0 cq	0.15	0.19	0.22	0.40	1.81	13.9
Static M-Est. 2.0 sh	0.21	0.24	0.27	0.37	1.00	4.67
Static M-Est. 2.5 cq	0.15	0.18	0.25	0.56	4.03	40.0
Static M-Est. 2.5 sh	0.20	0.22	0.24	0.36	1.13	5.40
Adaptive M-Est. cq	0.15	0.18	0.23	0.38	1.22	5.80
Adaptive M-Est. sh	0.23	0.24	0.37	0.36	0.97	4.12
FLOM 1.0	0.28	0.29	0.29	0.34	0.73	2.22
FLOM 1.5	0.19	0.25	0.36	1.12	6.43	24.5
FLOM 1.8	0.14	0.80	2.16	7.09	35.4	134
SCM	1.92	1.92	1.93	1.93	1.95	1.97

Table 4.7: Relative MSE of the eigenvalues for the A_3 mixing matrix

Estimator	ϵ 0	0.005	0.01	0.02	0.05	0.1
Sample	0.28	0.50	0.60	0.83	1.64	5.61
MAD	0.27	0.29	0.34	0.45	1.27	5.92
Static M-Est. 1.5 cq	0.30	0.32	0.36	0.46	1.21	5.67
Static M-Est. 1.5 sh	0.29	0.29	0.35	0.45	1.24	5.68
Static M-Est. 2.0 cq	0.27	0.31	0.34	0.48	1.20	5.86
Static M-Est. 2.0 sh	0.28	0.31	0.36	0.47	1.22	6.11
Static M-Est. 2.5 cq	0.27	0.30	0.31	0.45	1.16	5.59
Static M-Est. 2.5 sh	0.30	0.32	0.36	0.46	1.32	5.69
Adaptive M-Est. cq	0.26	0.30	0.34	0.44	1.21	5.62
Adaptive M-Est. sh	0.30	0.31	0.35	0.43	1.31	5.35
FLOM 1.0	0.26	0.28	0.31	0.41	1.15	5.20
FLOM 1.5	0.27	0.30	0.34	0.43	1.24	5.53
FLOM 1.8	0.27	0.35	0.42	0.52	1.34	5.55
SCM	0.30	0.31	0.35	0.46	1.29	5.65

Table 4.8: Relative MSE of the eigenvalues for the A_3 mixing matrix (whitened)

Estimator	ϵ					
	0	0.005	0.01	0.02	0.05	0.1
Sample	0.08	2.29	8.40	22.5	115	415
MAD	4.81	4.51	4.91	4.35	8.47	18.1
Static M-Est. 1.5 cq	0.23	0.25	0.35	0.59	3.65	39.5
Static M-Est. 1.5 sh	0.45	0.46	0.49	0.69	2.67	16.9
Static M-Est. 2.0 cq	0.13	0.19	0.34	0.92	10.5	141
Static M-Est. 2.0 sh	0.30	0.32	0.38	0.71	3.12	24.2
Static M-Est. 2.5 cq	0.09	0.23	0.50	1.93	34.4	258
Static M-Est. 2.5 sh	0.23	0.26	0.35	0.62	3.67	34.5
Adaptive M-Est. cq	0.12	0.22	0.35	0.74	3.96	39.4
Adaptive M-Est. sh	0.33	0.37	0.45	0.66	2.94	18.4
FLOM 1.0	0.62	0.67	0.73	0.79	1.50	3.55
FLOM 1.5	0.19	0.18	0.30	0.75	4.84	20.2
FLOM 1.8	0.10	0.42	1.29	4.77	28.6	114
SCM	1.32	1.32	1.25	1.23	1.15	1.05

Table 4.9: Relative MSE of the eigenvalues for the A_4 mixing matrix

Estimator	ϵ					
	0	0.005	0.01	0.02	0.05	0.1
Sample	0.21	0.41	0.51	0.84	1.92	7.55
MAD	0.71	0.82	0.85	1.11	2.62	10.3
Static M-Est. 1.5 cq	0.23	0.26	0.33	0.45	1.49	8.47
Static M-Est. 1.5 sh	0.29	0.31	0.37	0.54	1.69	7.8
Static M-Est. 2.0 cq	0.22	0.24	0.26	0.41	1.85	9.7
Static M-Est. 2.0 sh	0.26	0.29	0.33	0.48	1.67	7.52
Static M-Est. 2.5 cq	0.21	0.24	0.31	0.49	2.08	9.17
Static M-Est. 2.5 sh	0.23	0.27	0.33	0.44	1.60	7.96
Adaptive M-Est. cq	0.22	0.24	0.30	0.47	1.45	8.55
Adaptive M-Est. sh	0.27	0.31	0.35	0.49	1.53	7.79
FLOM 1.0	0.33	0.42	0.45	0.68	1.88	9.21
FLOM 1.5	0.21	0.24	0.28	0.43	1.35	6.76
FLOM 1.8	0.19	0.27	0.39	0.55	1.60	7.52
SCM	0.25	0.28	0.32	0.43	1.37	6.60

Table 4.10: Relative MSE of the eigenvalues for the A_4 mixing matrix (whitened)

4.4 Conclusion

In this section we have seen the performance of different estimators for covariance matrices. Encouraging results are found for the FLOMs, which are very fast and reliable estimators, suffering by a loss of performance for the Gaussian case, the M-Estimators with a smoothed Huber score function as a faster alternative to the clipped quadratic score function and the adaptive M-Estimators which are offering the same advantages as in the case of pure scale estimation.
We have seen that the whitening scheme in section 4.3 can increase performance, dependent on the mixing matrix A. It could be used as an extension to the FLOMs where the loss in performance for scenarios with lower interference is small or even as an extension for the sample- and the median absolute deviation covariance matrix where we can gain much performance with small effort.

Chapter 5

Conclusion

The work done in this thesis can be divided into three parts:

- A new score function - the smoothed Huber score function - for Huber's M-Estimator for scale has been presented. We have shown that by using this score function instead of the classical clipped quadratic score function one can decrease the mean square error for higher contaminations while at the same time decreasing the computational complexity.
- In the scenario of joint estimation of location and scale we observed the influence of a scale-parameter on a DC location-parameter. As a conclusion it can be said that assuming a robust estimator for location - such as Huber's M-Estimator - the difference in the error when using different (robust) scale estimators is very small. In this case a fast robust estimator such as the median absolute deviation would be sufficient.
- Finally we applied a whitening scheme in the scenario of estimating covariance matrices. We have seen that this whitening scheme is a useful and computationally inexpensive extension for the FLOM based estimation of covariance matrices or the element-by-element estimation of covariance matrices using the sample standard deviation or the median absolute deviation.

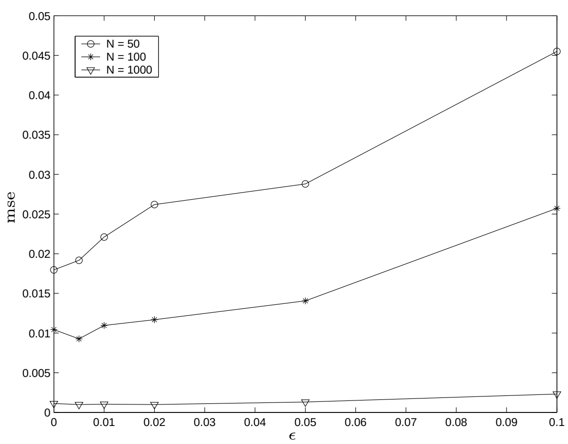
Future work based on this thesis could be done in the following areas:

- Exploring the performance of adaptive M-Estimators of scale whereby the collection of basis-functions consists of clipped quadratic score function with different values for the clipping point x_1 and smoothed Huber score functions with different values for x_1 and the width parameter b.
- Changing the scenario of joint estimation of location and scale with a DC location-parameter to a more complex one (e.g. a sinusoidal signal) and observing if the conclusion in chapter 3.4 still holds.

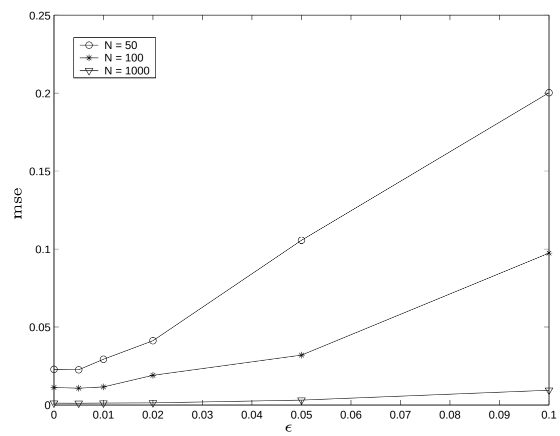
- Applying the different estimators of covariance matrices (with and without the whitening scheme) to methods like MDL or MUSIC and observing the mean square error of the respective estimates (the number of sources or their directions of arrival).

Appendix A

Joint estimation of location and scale

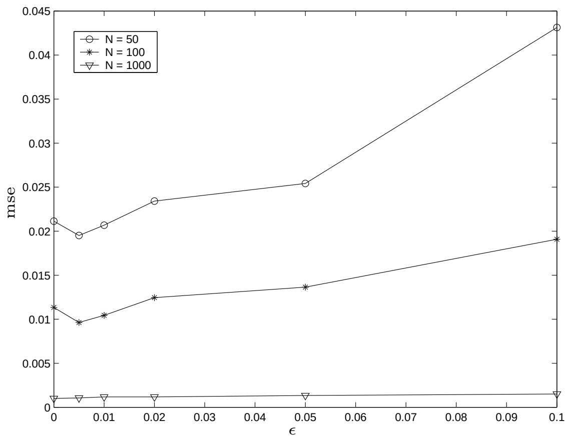


(a) $\kappa = 25$

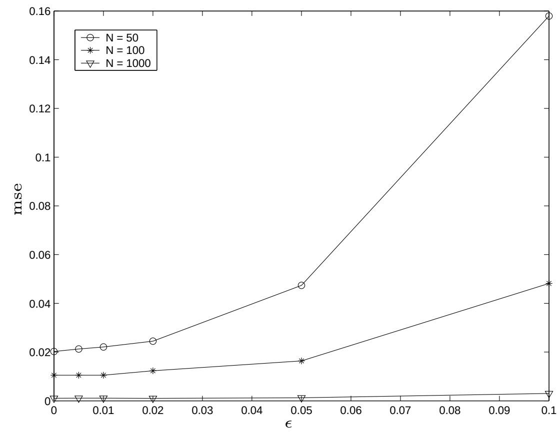


(b) $\kappa = 100$

Figure A.1: Performance of the sample standard deviation

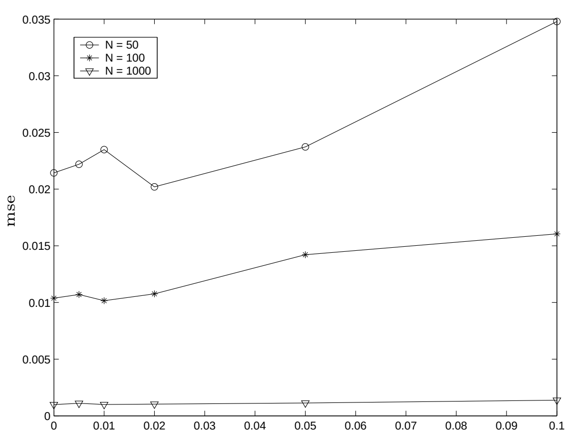


(a) $\kappa = 25$

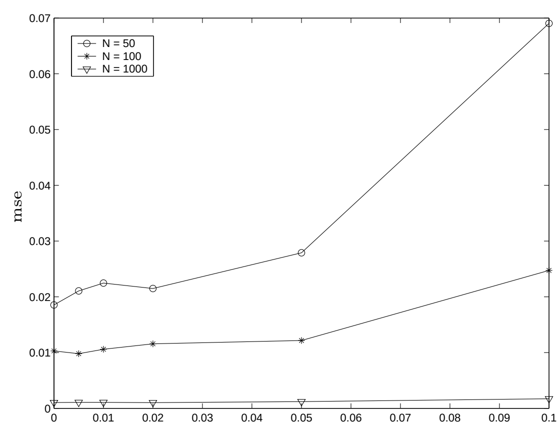


(b) $\kappa = 100$

Figure A.2: Performance of the α-trimmed standard deviation, $\alpha = 0.02$

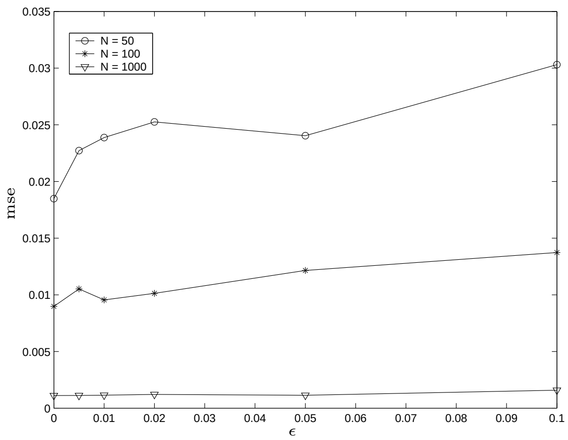


(a) $\kappa = 25$

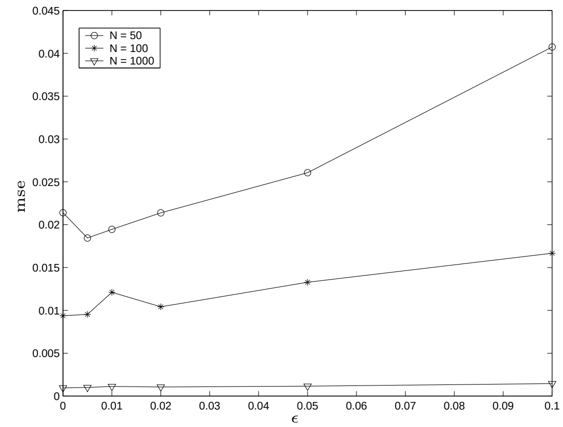


(b) $\kappa = 100$

Figure A.3: Performance of the α-trimmed standard deviation, $\alpha = 0.05$

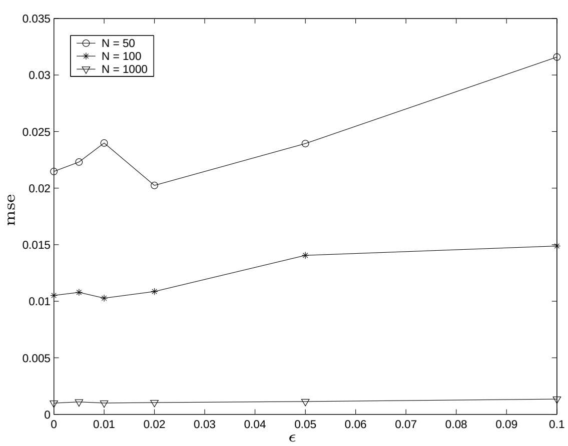


(a) $\kappa = 25$

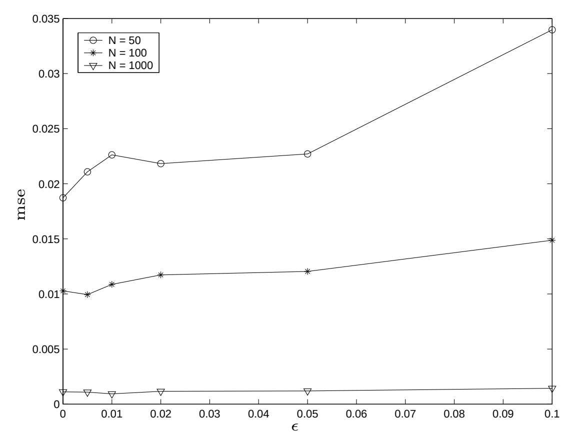


(b) $\kappa = 100$

Figure A.4: Performance of the α-trimmed standard deviation, $\alpha = 0.02$

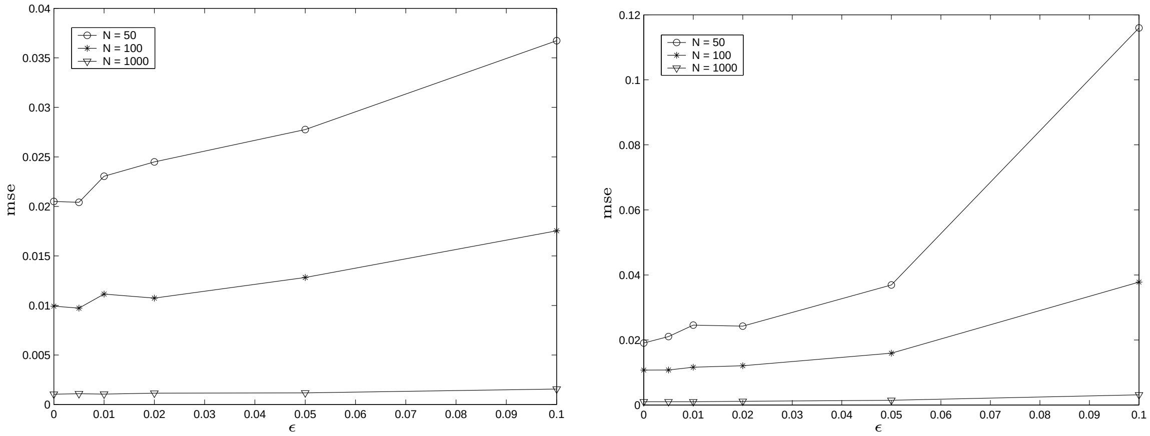


(a) $\kappa = 25$



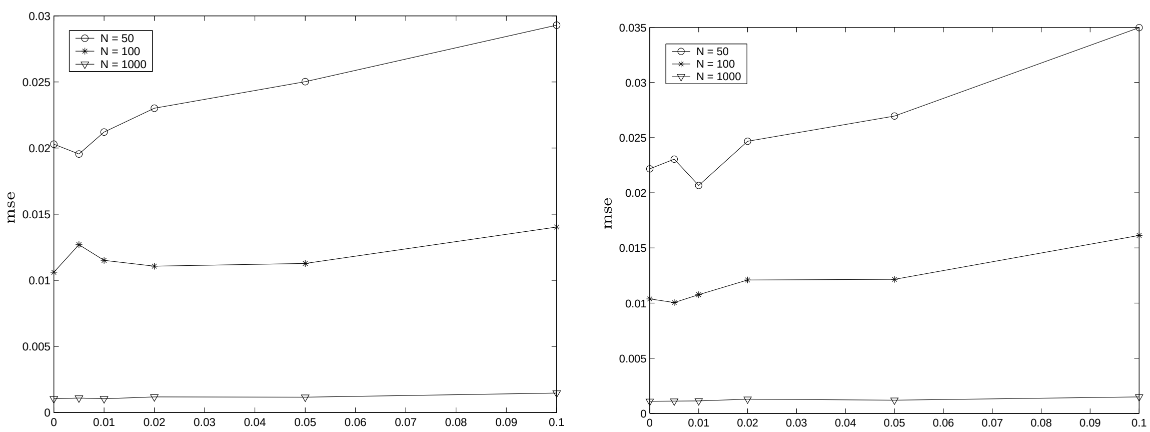
(b) $\kappa = 100$

Figure A.5: Performance of the interquartile range



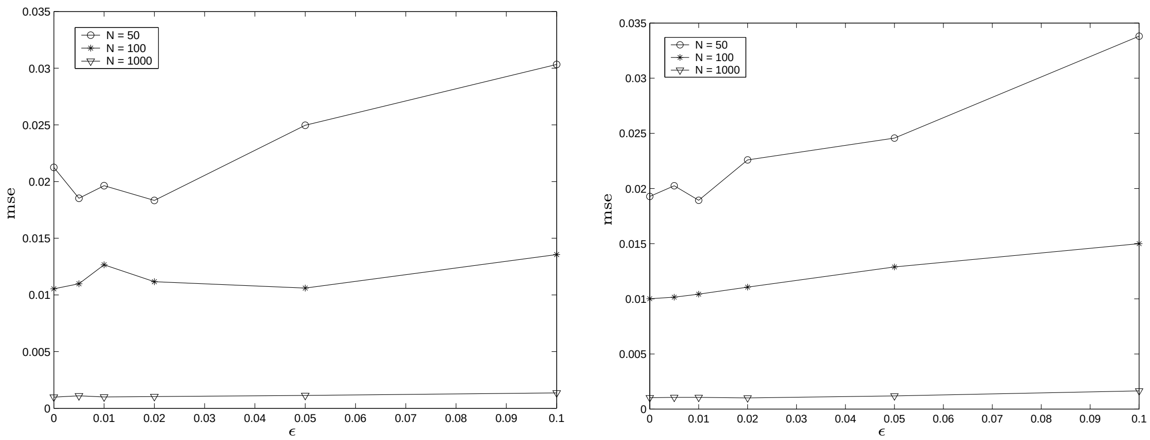
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.6: Performance of the mean absolute deviation



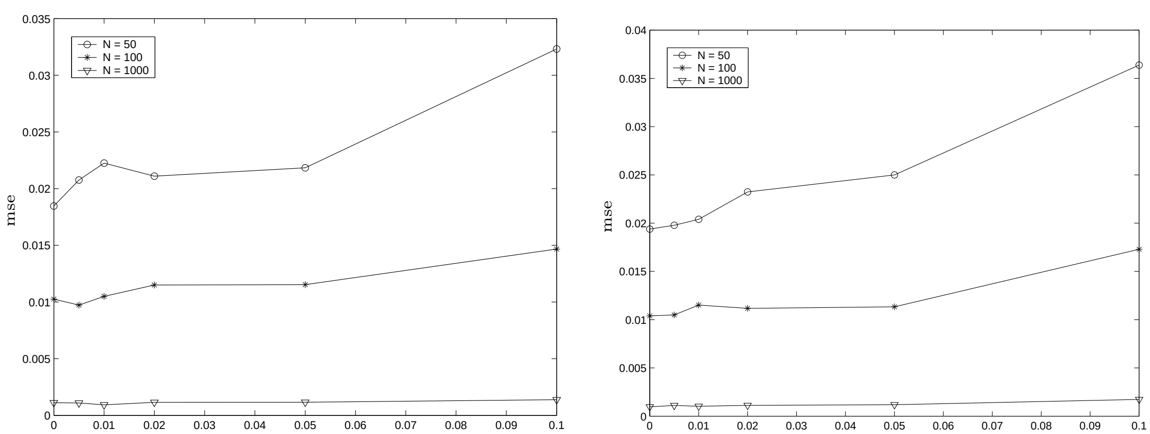
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.7: Performance of the median absolute deviation



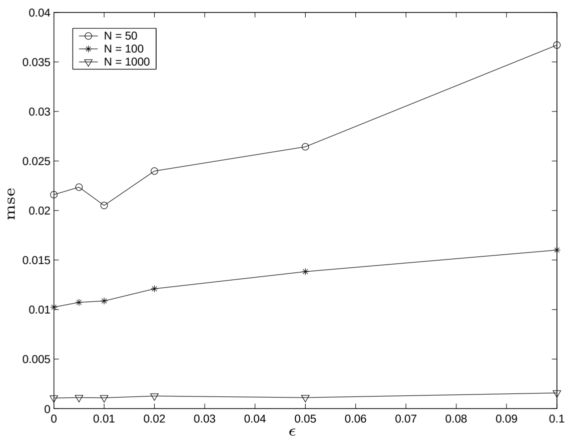
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.8: Performance of the static M-Estimator (cq), $x_1 = 1.5$

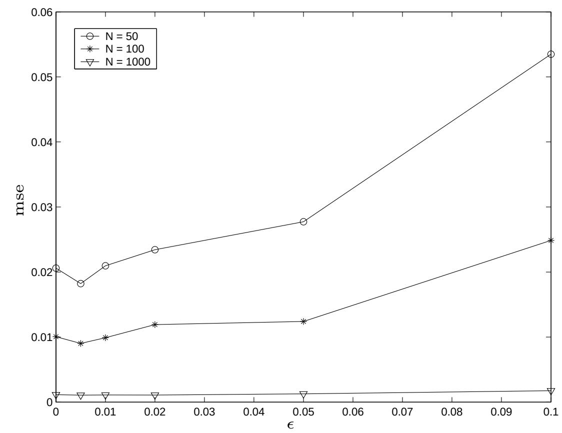


(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.9: Performance of the static M-Estimator (cq), $x_1 = 2.0$

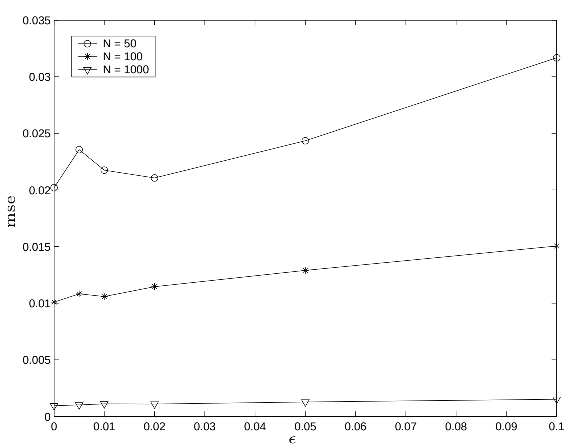


(a) $\kappa = 25$

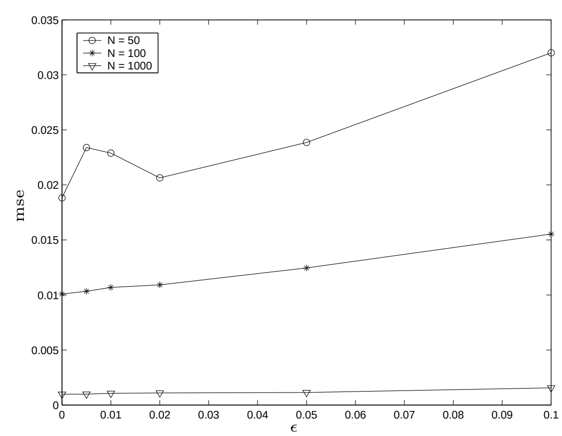


(b) $\kappa = 100$

Figure A.10: Performance of the static M-Estimator (cq), $x_1 = 2.5$

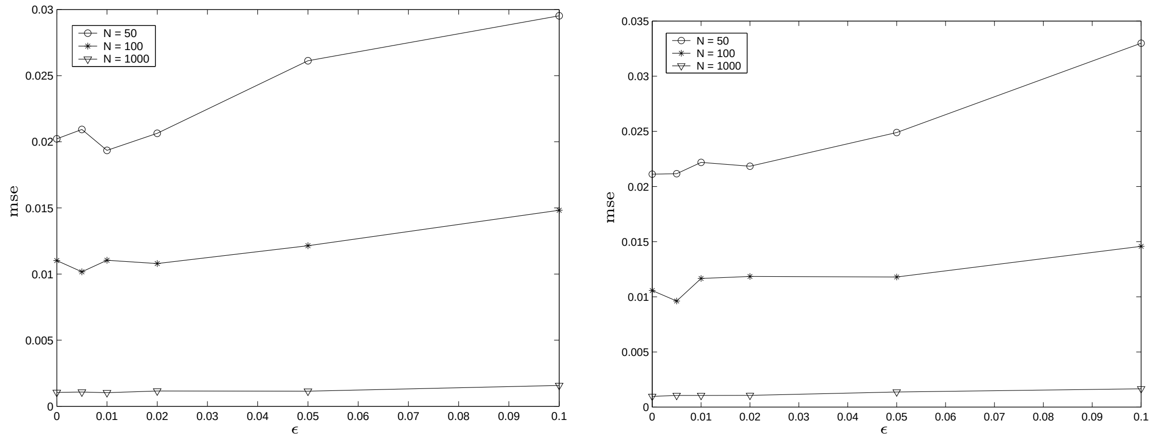


(a) $\kappa = 25$



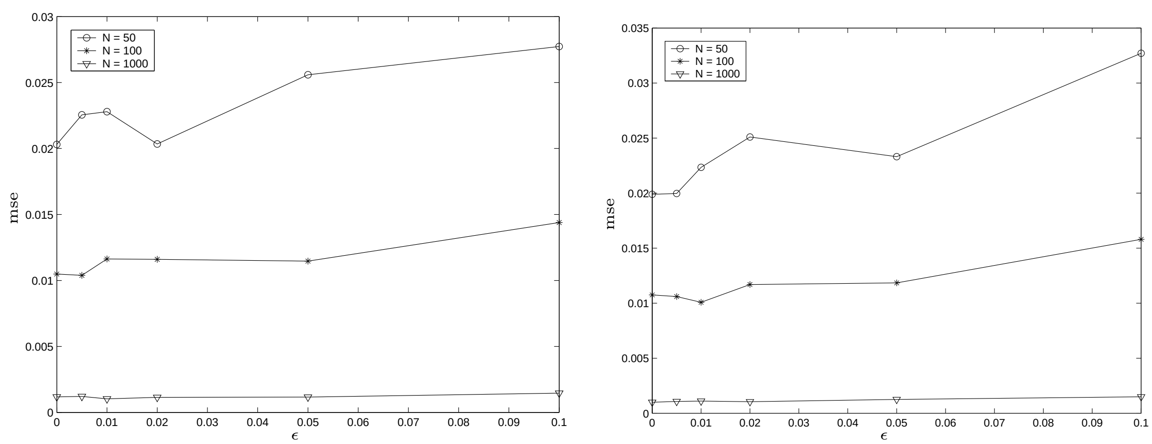
(b) $\kappa = 100$

Figure A.11: Performance of the static M-Estimator (sh), $x_1 = 1.5$



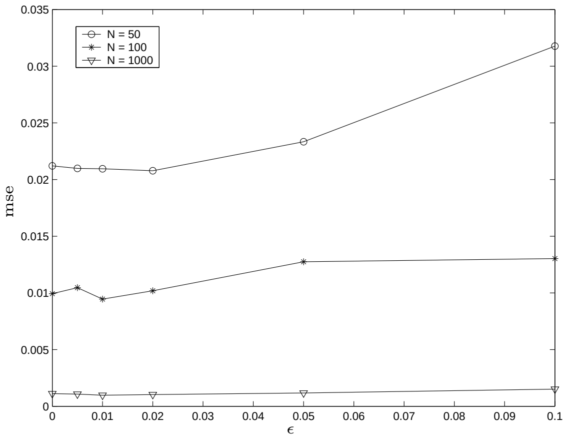
(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.12: Performance of the static M-Estimator (sh), $x_1 = 2.0$

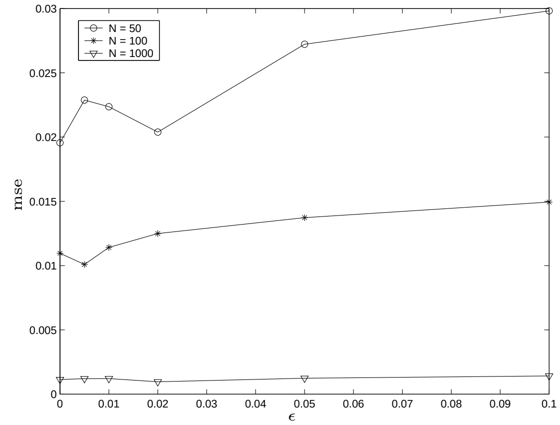


(a) $\kappa = 25$ (b) $\kappa = 100$

Figure A.13: Performance of the static M-Estimator (sh), $x_1 = 2.5$

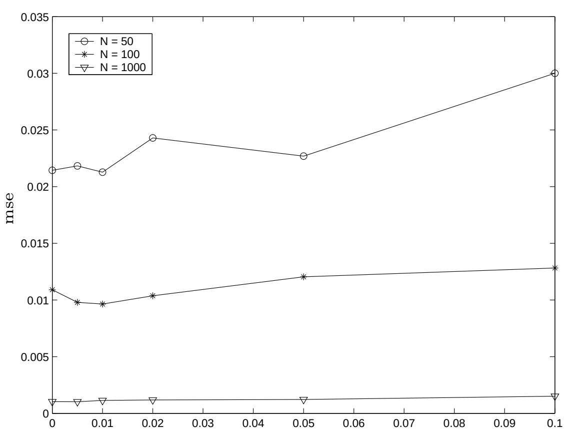


(a) $\kappa = 25$

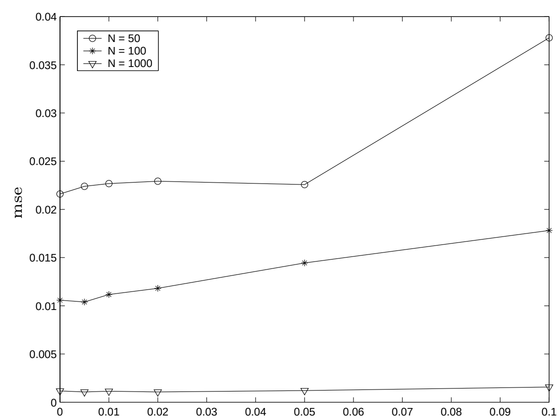


(b) $\kappa = 100$

Figure A.14: Performance of the adaptive M-Estimator (cq)



(a) $\kappa = 25$



(b) $\kappa = 100$

Figure A.15: Performance of the adaptive M-Estimator (sh)

References

[1] P.Huber, *Robust Statistics*, John Wiley, 1981

[2] R.F.Brcic, C.L.Brown and A.M.Zoubir, "An adaptive robust estimator for scale in cotaminated distributions", in *Proceedings of ICASSP 2004*, Montreal, Canada, May 2004

[3] S.Visuri, H.Oja and V.Koivunen, "Subspace-based direction-of-arrival estimation using nonparametric statistics", *IEEE Trans. on Signal Processing*, vol. 49, no. 9, pp. 2060-73, Sept. 2001

[4] X.Wang and H.Poor, "Robust multiuser detection in non-gaussian channels", *IEEE Transactions on Signal Processing*, vol.47, no.2, pp 289-304, February 1999

[5] H.Johansson, R.F.Brcich and A.M.Zoubir, "Robust multiuser detection in unknown noise channels" in *Proceedings of GLOBECOM'02*, volume 2, pages 1078-82, Taipei, Taiwan, November 2002

[6] M.Shao and C.L. Nikias, "Signal processing with fractional lower order moments: Stable processes and their applications", *Proc. IEEE*, vol. 81, no. 7, pp 986-1010, July 1993

[7] P.Tsakalides and C.Nikias, "The robust covariation-based music (roc-music) algorithm for bearing estimation in impulsive environments", *IEEE Trans. on Signal Processing*, vol. 44, no. 7, pp 1623-33, July 1996

[8] C.L.Brown, R.F.Brcich and C.Debes, "Adaptive M-Estimators for robust covariance estimation", ICASSP 2005 (under review), March 2005

[9] R. G. Staudte and S. J. Sheather, *Robust Estimation and Testing*, John Wiley, 1990