ROBUST ESTIMATION BY USING LINEAR SEQUENTIAL ALGEBRA

Beatrice Bucciarelli, Dr.
Osservatorio Astronomico di Pino Torinese
I - 10125 Pino Torinese (To)
ITALY

Gianfranco Forlani, Dr. - Luigi Mussio, Prof.
Dipartimento I.I.A.R. - Sezione Rilevamento
Politecnico di Milano
I - 20133 Milano
ITALY
e-mail geopoli@icil64.bitnet

Commission II

ABSTRACT

Most robust estimation techniques are in essence downweighting methods: in an iterative l.s. scheme suspicious observations get low weights according to some specified criterion. Here we focus on the use of a sequential updating of a preliminarly computed solution, as a possible alternative to repeating many times the whole adjustment. The formulae for the updating, in terms of parameters as well as in terms of observations, of the Cholesky factor and the inverse are introduced and their computational load is evaluated; a computer program operating on matrices stored taking sparsity into account is presented. Applications to outliers identification with different methods are discussed for block adjustment and surface reconstruction with spline functions.

KEYWORDS: Robust estimation, sequential algorithms, sparse matrices.

1. INTRODUCTION

In the framework of l.s. computations, it is often required to modify a previously computed solution. Using the standard procedure, this would imply to compute the whole adjustment from the beginning, what can be costly or time consuming. Sequential algorithms offer a more economic and efficient approach: they allow the complete updating of the solution of an equation system, i.e. they provide not only the addition or the removal of an observation equation, but also the elimination or the introduction of unknown parameters.

Sequentially building an equation system is a widespread technique in many areas of scientific computing: this is for instance the case in all dynamic measurement processes, where on-line data acquisition is often required to control in real (or near real time) the process evolution. In regression analysis, when testing for the significance of the parameters involved in determining the observed quantity, the obvious way of modifying the functional model is by using sequential algorithms.

In photogrammetry this approach became interesting with the advent of on-line triangulation, where the possibility of direct data acquisition on the computer and the opportunity of having a quick check and repair of measurement and identification errors strongly suggested the use of such a tool. Many algorithms have been presented and investigated to this aim in the last decade, among which Givens transformations (Blais, 1982; Blais, 1984; Inkila, 1984) are perhaps the most popular, in order to meet the specific requirements of on-line triangulation.

Here we focus on the use of sequential techniques to update a previously computed solution, the objective being to provide an alternative to repeating the whole adjustment of a l.s. equation system, rather than building the system itself. As an example, this might be the case of the design or the adjustment of a (large) geodetic network or photogrammetric block, to be updated with the addition of new observations or points. In the following we will rather concentrate on the problem of outliers identification and removal: we assume therefore the measurement process already completed, and we look for an efficient strategy for outliers identification. The dual operation on the unknowns, i.e. the modification of the parameter set will be put aside.

Most robust estimation techniques are basically downweighting methods, where in an iterative l.s. scheme suspicious observations undergo to a decrease of their role in determining the solution, through the modification of their weights according to some specified criterion. The amount of the weight change is generally determined on the basis of the (standardized) residual of the observation, and may involve from a theoretical point of view all observations; this would exclude the use of sequential techniques, since in this case repeating the whole adjustment is more economic. Following a more practical approach, changes to the weights will be assumed to be significant, only for the observations directly affected by the outliers and for a small other group around (roughly speaking, all the observations closely connected by the funtional model to erroneous ones). This means that, apart from pathologic situations, we can expect that only a small percentage of the weights will change from two successive iterations. In this frame, sequential updating becames again an attractive proposal for outliers removal. A weight change will be obtained by removing from the equation system the same (normal) equation, with weight equal to the given weight change.

2. A REVIEW OF ROBUST ESTIMATION METHODS

Outliers identification and solution methods insensitive to outliers are a main topic in the geodetic and photogrammetric community and many significant results have been established. The fundamental concepts of internal and external reliability introduced by Baarda received a widespread acknowledgement and provide guidelines in block design as well as in outliers identification (Foerstner, 1986). Many testing strategies

have been suggested to improve the efficiency of data snooping and reduce masking effects: some are based on still unidimensional test statistics and look for a satisfactory backward and/or forward elimination procedure (Benciolini et al., 1982; Barbarella, Mussio, 1985); a different approach define multidimensional hypothesis (Kok, 1984). In the last decade also robust estimation procedures became part of the mathematical background of geodetic and photogrammetric community; further achievements are coming out in robust testing. This might lead in the future to a decline of the fortune of the l.s. principle; at present, nevertheless, robust estimation methods heavily rely on l.s. since, as outlined above, their computational scheme is based on iterative l.s. adjustments.

"Robustness signifies insensitivity against small deviations from assumptions" (Huber, 1977): it is looked for an estimator being perhaps less efficient when all model hypothesis are satisfied, but which is still capable, to the contrary, of identifying the kernel of consistent observations. Among model assumptions violations, the more understood is perhaps the shape of the true underlying distribution deviating slightly from the assumed (usually the gaussian distribution). According to (Hampel, 1986), "robust statistics are the statistics of the approximate parametric models"; this means robust estimators are derived under a distributional model more flexible than the maximun likelihood estimators: more precisely they provide an infinite dimensional neighbourhood of a specified parametric model. Contaminations of the basic distribution are then explicitely accounted of. The estimation procedure is designed to provide a screening among the observations, taking a priori into account that not all of them should be given the same role in determining the solution. This does not happen to l.s. estimates, where all observations equally contribute, on the basis of their a priori variance, to the solution.

In its original formulation (Huber, 1964) introduced the so called M-estimators, a generalization of maximum likelihood type, where the probability density function F is defined as the convex combination of the given probability distribution G of the observations and of an unknown contaminating distribution H:

$$F(x-t) = (1-e)\, G(x-t) + e\, H(x-t) \qquad (1)$$

For small e, this allows to represent explicitely an approximate parametric model. An M-estimator T_n of a location parameter t is defined as the solution of the minimum problem

$$\sum R\ (x_i - T_n) = \min \qquad (2)$$

or by the implicit equation

$$\sum H\ (x_i - T_n) = 0 \qquad (3)$$

where $R(x,t)$ is an arbitrary function and $H(x,t)$ its derivative with respect to t.
Defining a weight function W with coefficients

$$w_i = \frac{H\ (x_i - T_n)}{(x_i - T_n)} \qquad (4)$$

equation (3) is rewritten explicitely with respect to T_n as a weighted mean of the observations:

$$T_n = \frac{\sum w_i\ x_i}{\sum w_i} \qquad (5)$$

M-estimator are then characterized by these three functions.
Further insight in the theoretical foundations of estimation methods was provided by Hampel, through the definition of concepts like the influence function I.F. and the breakdown point. The former describes the effect on the estimate of a small contamination of the distribution: this can be evaluated as the differential influence on the estimate of an observation of value x, when sample size goes to infinity. If the I.F. is bounded, outliers cannot completely ruin the estimates. The latter represent the percentage of contaminating data which can be tolerated by the estimator, before it deviates. For M-estimators, it can be shown that the influence function is proportional to the derivative of the objective function.

Applying these concepts to l.s. criterion, we found that the I.F. is a straight line (the weight function is a constant) so there is no limit to the influence of contaminating observations.
To the contrary, all robust methods show a bounded I.F. (the weight function goes to zero to infinity or even is zero after some threshold value): this means that the contribution of an observation to the determination of the estimate decreases (and may become null) with the degree of consistency of the observation with the bulk of the data.

In Huber estimator a threshold c is introduced, to separate the observations according to the magnitude of their standard deviation from the location parameter: the I.F. is the same as the l.s. for observations smaller than c and became a constant for larger values. For $|x| > c$, this leads to weights vanishing as $c/|x|$. It can be shown that the corresponding distribution function of the observations is normal in [-c;c] and exponential outside.

Hampel estimator is a "three part redescending estimator", with three constans c1, c2, c3 defining four regions: the I.F. coincides with Huber estimator in the first two, then goes linearly to zero at c3 and is zero for larger values: c3 defines a finite rejection point. Since I.F. is descending, the objective function is not any more convex: this means the uniqueness of the solution cannot be guaranteed.

3. OUTLIERS IDENTIFICATION: ROBUST METHODS OR L.S. TESTING PROCEDURES ?

As already mentioned, the computation of the solution with such (and similar) estimators can be performed in an iterative l.s. scheme, according to the weight function specified by the method. Also other robust methods, independently derived in the framework of geodetic sciences like the Danish method (Juhl, 1984; Eeg, 1986), use a fully corresponding procedure, since the basic idea is to weaken observations with high residuals. To reduce the computational load, only the solution vector is computed at each iteration, since it is not strictly necessary to compute the covariance matrix of the residuals. The weight change is determined on the basis of the magnitude of the residuals. Iterative methods are the best choice for the solution, taking into account their characteristics: they allow the sparsity of the normal system to be fully exploited in programming algorithms, since they preserve sparseness, and offer fast convergence rates. Because of its characteristics, the conjugate gradient method has got the largest popularity.

On the other hand, outliers removal is just a preliminary step for the adjustment: confidence interval and error ellipses of the estimated parameter are of great interest. It is therefore necessary to invert the normal matrix, by first computing the Cholesky factor T and then solving the matrix equation:

$$T^t\ C^{-1} = T^{t\,(-1)} \qquad (6)$$

at the end of the elimination process. Because of the fill-in of the envelope of T, sparsity is not preserved, and a different array structure is to be used for the inversion.

When using testing strategies based on data snooping

approach, standardization of the residuals and covariance matrices are necessary at each iteration step of the procedure, if masking effects are to be minimized. These considerations give support to the use of sequential updating of the solution, since this would be a more economic way of facing the amount of computations required. As already outlined above, in downweighting methods the magnitude of the weight change is determined on the basis of the residual of the observation. Empirically choosing a threshold, changes to the weights will be assumed to be significant only for larger residuals, under the assumption that only those affected by the outliers (roughly speaking, all the observations closely connected by the funtional model to the erroneous ones) are greater than the threshold. Only a small percentage of the weights should change from two successive iterations. In this frame, sequential updating becames again attracting for use in outliers removal also with robust downweighting methods. A weight change will be obtained by removing from the equation system the same (normal) equation, with weight equal to the given weight change.

4. THE SEQUENTIAL UPDATING OF A L.S. SOLUTION

4.1 **Adding or removing observations: updating the Cholesky factor T and the solution vector**

Applying the Householder orthogonal reflection matrix Q to a m-vector v results in a vector u=Qv whose elements are zeroed from the p-th on, (1<p<m) (Lawson and Hanson, 1974); a sequence of n such reflections, applied to the column of the design matrix A, leads to a (m,n) matrix U having non-zero elements only in the upper triangle; this takes about m n^2 operations. The Cholesky factor T and the upper triangle in U are identical within row signs:

$$T^t T = C = A^t A = U^t Q^t Q U = U^t U \qquad (7)$$

Given the Cholesky factor T and the corresponding known term d, adding a new row vector a (a new observation equation) to the normal system is equivalent to consider the extended matrix $U^* = [U^t \mid a^t]^t$ and apply n Householder transformations to that matrix, in order to zero all coefficients in the last row. Since only one element is to be zeroed in each column, the tranformation becames in fact a Givens rotation.
The operation count then drops from the order of m n^2 operations (or from about n^3 operations solving with Cholesky factorization the augmented system), to the order of n^2. It can be seen also that the j-th tranformation modifies only the elements of the last row and of the j-th row, both from the j-th column on. Only if the corresponding elements of the row are both zero in the original matrix they will be so in the updated. This means also that the profile is subject to changes, as soon as new connections are generated outside the original. Writing the transformation in terms of the elements t_{ij} of the Cholesky factor requires a working array w of lenght n, containing the elements of the row to add or remove. Let T* be the updated matrix and $w^{(i)}$ the vector w at the i-th step; we have:

$$t^*_{ii} = (t^2_{ii} +/- (w^{(i)}_i)^2)^{1/2}$$
$$w^{(i+1)}_i = 0 \qquad (8)$$

while for the off-diagonal elements we obtain:

$$t^*_{ij} = (t_{ii} t_{ij} +/- w^{(i)}_i w^{(i)}_j) / t^*_{ii}$$
$$w^{(i+1)}_j = (w^{(i)}_j t_{ii} - w^{(i)}_i t_{ij}) / t^*_{ii} \qquad (9)$$

There is no basic difference between adding and removing an equation, since all modifications to the existing elements merely consist of the addition of some terms representing the contribution of the equation to be introduced or deleted. When an observation is to be removed, the corresponding equation will be therefore added with all signs of the correction terms changed, so that the effect of its introduction is cancealed.
Once T* has been computed, the solution vector follows by the forward-backward substitution. All in all the amount of operations required is therefore in the order of n^2.

4.2 **Adding or removing observations: updating the inverse**

In order to update the inverse, the following linear algebra theorem is used: given the square nonsingular matrices Q, S and matrices R, T so that the product

$$Q +/- R S T$$

can be defined and the result is nonsingular, and if matrix

$$S^{-1} +/- T Q^{-1} R$$

is also nonsingular, then the following equality holds:

$$(Q +/- R S T)^{-1} =$$
$$= Q^{-1} -/+ Q^{-1} R (S^{-1} +/- T Q^{-1} R)^{-1} T Q^{-1} \qquad (10)$$

Let now substitute the normal matrix C for the matrix Q, the scalar p (the weight of the observation to be introduced) for S and the transpose a^t of the row vector of coefficients for R, the row coefficient vector for T. The expression then becames:

$$(C +/- a^t p a)^{-1} =$$
$$= C^{-1} -/+ C^{-1} a^t (p^{-1} +/- a C^{-1} a^t)^{-1} a C \qquad (11)$$

The procedure does not require to compute any inverse matrix explicitely, using the Cholesky factor T instead of the inverse C^{-1}: only an additional working array e of dimension n is used. The first step is the backward-forward substitution

$$T^t T e = a^t \qquad (12)$$

yielding the unknown e; then the scalar product

$$a e = f$$

is computed, followed by the matrix product

$$e g^{-1} e^t = H \qquad (13)$$

where the scalar g is defined as:

$$g = p^{-1} +/- f$$

The updated inverse C^{*-1} is then obtained as

$$C^{*-1} = C^{-1} + H$$

Since the major cost of each updating is due to (12) and (13), the operation count is again in the order of n^2.

4.3 **Adding or removing unknown parameters: updating the Cholesky factor and the solution vector**

Modifying the parameter set is equivalent to add or remove a column to the design matrix (and a row and a column to the normal matrix). Introducing a new unknown is straightforward if the new parameter is stored in the last position in the normal matrix: there is no need for any modification of the previuos Cholesky factor. In fact, since T can be computed also column by column, from left to right, this is nothing but the factorization of the last column of the normal matrix. Removing an unknown, to the contrary, imply changes in all elements of the triangle whose row and column number are both larger than those

of the unknown to eliminate: again, this is clearly understood by looking at factorization formulae:

$$t_{ii} = \sqrt{c_{ii} - \sum_{k=1}^{j-1} t_{ki}^2} \qquad (14)$$

$$t_{ij} = \left(c_{ij} - \sum_{k=1}^{j-1} t_{ki} t_{kj} \right) / t_{jj} \qquad j < i \qquad (15)$$

In order to modify the elements of the above mentioned triangle, it is sufficient to take into account that, since the matrix element t_{ij} results from a sum of products, whose factors are the corresponding elements of columns i and j, the contribution due to the unknown k ($k<i<j$) is easily identified. To compute the modified element we merely subtract this term to the original; to this aim, only the nonzero part of the row corresponding to the parameter to remove is used. Let w be again a working array of size n, containing at step 1 the h-th row to be removed:

$$\begin{aligned} t^*_{ii} &= (t^2_{ii} +/- (w^{(i-h)}_i)^2)^{1/2} \\ w^{(i-h)}_i &= 0 \end{aligned} \qquad (16)$$

while for the off-diagonal elements we obtain:

$$\begin{aligned} t^*_{ij} &= (t_{ii} t_{ij} +/- w^{(i-h)}_i w^{(i-h)}_j) / t^*_{ii} \\ w^{(i-h)} &= (w^{(i-h)}_j t_{ii} - w^{(i-h)}_i t_{ij}) / t^*_{ii} \end{aligned} \qquad (17)$$

The corrections apply for $i = h+1, n$ and $j = h+1, n$; it can be noticed that (16) and (17) are in fact fully equivalent to (8) and (9). The number of elements to be actually updated, therefore, depends on the storage position of the parameter in the matrix: it can be as large as $n(n+1)/2$ when unknown 1 is removed or just zero when unknown n is cancealed. On average, considering that each updating takes three operations (an operation = a multiply and an add), we can obtain again an order of n^2 operations.

4.4 Adding or removing unknown parameters: updating the normal inverse

To derive the expression of the updated inverse, let first consider a partitioning in four submatrices of a positive definite matrix, in such a way that the last diagonal element s is one of the diagonal blocks and the first n-1 elements of last row r^t and column r are the off-diagonal blocks, C being the rest:

$$\begin{bmatrix} C & r \\ r^t & s \end{bmatrix} \qquad (18)$$

The corresponding partitioning of the inverse

$$\begin{bmatrix} G & r \\ r^t & s \end{bmatrix} \qquad (19)$$

expressed as a function of s, r and C is obtained by exploiting the identity:

$$\begin{bmatrix} C & r \\ r^t & s \end{bmatrix} \begin{bmatrix} G & r \\ r^t & s \end{bmatrix} \qquad (20)$$

We obtain for r, s and G the following expressions:

$$r = - (C^{-1} - r r^t / s)^{-1} \; r/s \qquad (21)$$

$$s = (1 - r^t r) / s \qquad (22)$$

$$G = C^{-1} + (I + r r^t (C^{-1} - r r^t / s)^{-1} / s) \qquad (23)$$

Applying theorem (10), (23) breaks off in a series of elementary products: the product $C^{-1}r = e$ is not computed explicitely, but rather a backward-forward substitution using T gives the vector e. Then the scalar product:

$$r^t e = f$$

is computed and the matrix products:

$$e\, e^t = G$$

$$e\, f\, e^t / s\,(s-f) = H$$

follow. From the last two equation we obtain:

$$G = C^{-1} + G + H \qquad (24)$$

Again from (10) applied to (21) we get:

$$r = - e/s - e\, f/s(s-f)$$

When an unknown parameter is removed from the model, expression (24) is to be used, this time solved with respect to C^{-1}:

$$C^{-1} = G - G - H$$

Note that there is no actual dependence on C^{-1} in G and H, since all terms involving the inverse can be computed by first updating the factor T and then solving once more two triangular systems.

In order to introduce the solution formulae we used a partition where the modified parameter is the last. This is not actually a limitation, since we can always think of pre- and post-multiply C^{-1} by a permutation matrix, so that the candidate parameter becames the last. This is just a symbolic operation, since in a computer program we merely skip over in all computation the element of row and column to remove.

5. THE PROGRAM SEQUALGE

As outlined above, in our approach outliers identification cannot be seen as a separate step of the adjustment procedure: this lead us to integrate the sequential updating just described into the existing software library of our Department.
The stress has been put on two applications which are of main interest at D.I.I.A.R. since the last decade: the adjustment of geodetic network and photogrammetric data (Forlani, Mussio, 1986) as well as surface reconstruction by finite elements interpolation and l.s. collocation filtering (Crippa, Mussio, 1987). To take the sparsity structure of normal matrices in geodesy and photogrammetry into consideration, the adjustment program CALGE uses Cholesky factorization and minimum profile algorithm (Gibbs et al., 1976) to reduce storage requirements; in the design matrix only non-zero elements are stored, while the whole envelope of the normal matrix is stored in an array, overwritten with the Cholesky factor T and the partial inverse. Similar considerations apply to the interpolation program SPLINE-P, which is available also with a conjugate gradient version.
Within this background, SEQUALGE program has been written, capable of all updating of a l.s. problem depicted above.
Actually, three different versions of SEQUALGE exist, dealing with full, banded and sparse matrices, to integrate with the corresponding linear algebra library routine. The algorithms for the updating apply equally well to full and sparse matrices. Notice only that, as in l.s. adjustment programs, the inverse is only updated (and computed) within the profile, in the sparse storage version. In terms of

storage, requirements are larger than those demanded by solving the system in a single step: we need both T and C^{-1}. In fact this is not strictly necessary, since there is no need for these matrices to be in core memory at the same time: nevertheless, loading and unloading from disk would reduce sharply program performances.

Each time an equation or an unknown parameter is removed, both the design matrix and the factor T are modified, to save storage. When adding unknowns, they are always put in the right margin of the normal matrix, since this is straightforward. Introducing new equations, on the other hand, may change the profile: then a preliminary check is necessary in the profile version of the program; if it is the case the envelope is enlarged. This is also the reason why, for geodetic and photogrammetric applications, SEQUALGE has been designed primarily to update equation systems, rather than building them sequentially from the beginning. The latter approach would make reordering very inefficient, since each time an unknown is introduced, a new reordering is necessary. The argument does not apply of course to SPLINE program, where reordering is not used.

At the current stage, SEQUALGE has been integrated as a subroutine module in the two mentioned adjustment programs. The outliers identification procedure allows the use of Huber and Hampel estimators, as well as the data snooping.

6. EVALUATION OF PROGRAM AND ALGORITHMS PERFORMANCES - EXAMPLES

To compare the performances of different robust methods in gross errors identification and to evaluate program performance a series of adjustment has been executed on two photogrammetric blocks and a DTM.

All estimators available to the program were used in the computations. Observations were completely simulated in the photogrammetric examples, while for the DTM the original measurements were used. Moderate and large gross errors were introduced in different combinations and percentages in data set, to have a picture of the sensitivity of the robust methods used.

6.1 **Program performance**

Describing sequential algorithms we already pointed out the amount of savings in terms of floating point operations and the increased cost of storage. When focussing the analysis on the use of programs taking sparsity into account, some additional remarks are necessary.

The basic assumption in suggesting the use of such methods in the context of outliers location is that the number of weights subject to changes in an iteration should be small, compare to the number of unknowns. This fact may not occurs when a large set of data is acquired, e.g. automatically. However in this case a recommandable statistical treatment implies always the preprocessing of small subsets of data. If this is not the case, repeating each time the whole adjustment would be more economic. Let be:

m the number of observation equations;
n the number of unknowns;
h the bandwidth of the normal matrix;
k the average number of weight changes in an iteration;
l the number of iterations in the identification procedure.

It is important to evaluate the number of operations required to complete the process when dealing with arrays storing the whole normal matrix rather than only its profile. Working only within the profile, the number of operation required to solve a l.s. problem is in the order of $n\ h^2$. The corresponding single step (a weight change) of a sequential update will then involve n h operations. If we compare now the ratios between full and sequential solution in case of full and sparse storage, we see that, while for full matrices the sequential solution is always reasonably competitive (especially when using data snooping), dealing with sparse storage, the dependence of the complete solution switches from the number of unknonws to the bandwidth: in this case only for small percentages of weight changes sequential updating will be still more efficient.

To try to establish a balance point between the two approaches, let assume the bandwidth to be proportional to the square root of the number of unknowns and the relative redundancy to be about 2. If we assume 1% of weights to change at each iteration, we have the balance for n close to 2500; with 2% the balance is reached much earlier, at about 600 unknowns.

In order to empirically confirm this analysis of program performance, a service version of CALGE and SPLINE-P has been set up, which, once observations weights are modified, restart the complete system solution. The same results are then obtained by using sequential updating and by building a new normal system at each iteration. A comparison has been done between the CPU time spent by SEQUALGE in the iterative sequential updating of the weights and CPU time required to the service version for the equivalent adjustment.

6.2 **Test 1: joint adjustment**

The photogrammetric blocks used were adjusted together with their control network on ground. Both examples use simulated observations, in order to be able to monitor the behaviour of the estimators; the observation scheme in both cases is a realistic one, since we used an existing block and net configuration.

The first block consists of 13 photograps divided in three strips, with 65% forward overlap and 20% sidelap. The control net is made of a triangulation net and of a spirit levelling loop. The number of observations is 846, the unknown parameters are 549.

The second consists of four strip, with photograps taken with 65% forward overlap and 50% sidelap, for a total of 15 models. The control network includes four connected levelling loops and a two chains of traverses, strongly connected. The number of observations is 1548, the unknowns are 745.

In both examples the observations were first given normally distributed errors and then small outliers (two or three times the minimum undetectable error of the observation) were introduced, in the control network as well as in the block.

In a second series of simulations, bigger outliers were used.

6.3 **Test 2: surface reconstruction**

The DTM used in this example is the Noiretable area, the same used as test area from OEEPE some years ago. It consist of profiles almost equally spaced with an interval of about 20 m, with irregular sampling within each profile; the amount of observations is close to 6600. Bicubic splines were introduced every 100 m on a regular grid, this choice resulting in a normal system of about 1300 unknowns. The sigma naught of the spline interpolation before introducing the errors was about 2 m; then a number of gross errors of constant size (15 m) were introduced, roughly equally spaced, amounting to 2% of the observations. Then the error location procedure was performed.

6.4 **Analysis of the results**

From the relatively limited amount of simulations performed we could not get a clear picture of differences in the behaviour of the robust methods.

In the two photogrammetric blocks, with small outliers we had ambigous results: the largest errors were identified, while some of the smallest were not; furthermore, we noticed still masking effects, since also "correct" observations were labeled as outliers. It is also difficult to highlight differences among the estimators (at a first look, one could say rather they behaved quite the same way),

since wrong or missing identifications occurred to all of them. With large outliers on the contrary, the results are very satisfactory, since all errors were found; again, no apparent difference among the methods could be noticed.
In DTM interpolation, with true observations, results are apparently better, because all the outliers introduced were found, by all methods. Some other observations were labeled as erroneous, but in this case we cannot say whether masking occurred or not.

REFERENCES

Barbarella, M., Mussio, L., 1985. A strategy for a robust identification of outliers in geodetic sciences. Statistics and Decisions, Supplement Issue n. 2, Oldenbourg Verlag, Muenchen.

Benciolini, B., Mussio, L., Sansò, F., 1982. An approach to gross error detection more conservative than Baarda snooping. In: Int. Arch. Photogram. Remote Sensing., Helsinki, Vol. 24, Part 3.

Blais, J.A.R., 1982. Recursive least-squares estimation using Givens Tranformations. In: Int. Arch. Photogram. Remote Sensing, Helsinki, Vol. 24, Part 3.

Blais, J.A.R., 1984. Optimization of least-squares computations in on-line photogrammetry. In: Int. Arch. Photogram. Remote Sensing, Rio de Janeiro, Vol. 25, Part 3.

Crippa, B., Mussio, L., 1987, The new ITM system of programs MODEL for digital modelling. In: Proc. of Int. Symp. Progress in Terrain Modelling, Jacobi, O., Frederiksen, P. (Eds), Kobenhavn.

Eeg, J., 1986. On the adjustment of observations in presence of blunders. Geodetic Institut Technical Report n. 1, Kobenhavn.

Foerstner, W., 1986. Reliability, error detection and selfcalibration. In: Int. Arch. Photogram. Remote Sensing., Rovaniemi, Vol. 26, Part 3.

Forlani, G., Mussio, L., 1986. Test on joint adjustment of geodetic and photogrammetric data. In: Int. Arch. Photogram. Remote Sensing., Rovaniemi, Vol. 26, Part 3.

Gibbs, N. E., Poole, W. G., Stockmeyer, 1976. An algorithm for reducing the bandwidth and profile of a sparse matrix. SIAM, Numerical analysis, Vol. 13, n. 2.

Hampel, F.R., Ronchetti, E., Rousseeuw, P., Stahel, W., 1986. Robust statistics. J Wiley & Sons, New York.

Huber, P.J. 1964. Robust estimation of a location parameter. Annals of Mathematical Statistics, 35.

Huber, P.J., 1977. Robust Statistical Procedures. Society for Industrial and Applied Mathematics (SIAM) (Ed), Arrowsmith Ldt., Bristol.

Kok, J., 1984. On data snooping and multiple outlier testing. NOAA Technical Report NOS NGS 30, Rockville, Maryland.

Inkila, K., 1984. Recursive least squares estimation by updating Cholesky factorization. In: Int. Arch. Photogram. Remote Sensing, Rio de Janeiro, Vol. 25, Part 3.

Lawson, G.L., Hanson, R.J., 1974. Solving least squares problems. Prentice Hall, Englewood Cliffs, New Jersey.