# A Ferris-Mangasarian Technique Applied to Linear Least Squares Problems

J. E. Dennis*
Computational and Applied Mathematics
Rice University
Houston TX 77005-1892

Trond Steihaug†
Department of Informatics
University of Bergen
Høyteknologisenteret
N-5020 Bergen Norway

May 1, 1998

**Abstract**

This note specializes to linear least squares problems an approach suggested by Ferris and Mangasarian [4] for solving constrained optimization problems on parallel computers. It will be shown here that this specialization leads to an algorithm which is mathematically equivalent to an acceleration and convergence forcing modification of the block Jacobi iteration applied to the normal equations. The resulting algorithm is a promising way to speed up a parallel multisplitting algorithm of Renaut [9] for linear least squares. Renaut's algorithm is related to a specialization of part of the Ferris and Mangasarian approach.

*Research supported by DOE FG03-93ER25178, CRPC CCR-9120008, AFOSR-F49620-95-1-0210, The Boeing Company, and the REDI Foundation.

†Research supported by The Research Council of Norway and VISTA – a research cooperation between the Norwegian Academy of Science and Den norske stats oljeselskap a.s (Statoil).

# 1 Introduction

This note specializes to linear least squares problems an approach suggested by Ferris and Mangasarian [4] for solving constrained optimization problems on parallel computers. It will be shown here that this specialization leads to an acceleration and convergence forcing mechanism for the block Jacobi iteration applied to the normal equations. We do not form the full normal equations, but our numerical results hint that the condition number of the normal equations affects the number of iterations required for a given problem.

The target problems are assumed to be large. The technique suggested has for each iteration two basic stages both of which involve the solution of smaller linear least squares problems. The first stage is to partition the optimization variables for the problem and then on separate processors to solve the smaller least squares problem in which only those variables in a single partition are allowed to move. The partitioning of the variables is at the discretion of the user, and hence it can be used to select the size of the problem to be solved on each processor. The user may make other considerations in partitioning the variables such as consistent scaling of the partitioned problems.

It is well known that the iteration defined by incrementing each variable by the amount indicated in this first stage and then updating the residual for the next iteration is the block Jacobi iteration applied to the normal equations for the original linear least squares problem. See Björck [1] and the references therein. The classical Jacobi iteration alone may not converge [5, 8].

The second stage of each iteration is to compute a new iterate by a synchronization step that involves solving a least squares problem in a smaller space of surrogate variables identified by the first stage. If the surrogate variables are taken only to be the increments produced by solving the subspace least squares problems, then we will call this the Jacobi method with subspace correction, and it always converges for full rank problems. It turns out that this method has already been considered by Renaut [9]. She calls it Optimal Recombination Least Squares Multisplitting (ORLSMS), and she gives further developments based on the multisplitting method of O'Leary and White [7].

For simplicity, we will restrict ourselves here to the case that at each of the two stages we accurately solve the smaller minimization problems, but this is neither necessary to the theory nor even possible in the general setting considered in [4].

Mangasarian [6] and Ferris and Mangasarian [4] introduced in stage one auxiliary variables, which they called "forget-me-not" variables. The contribution of this paper is to show how to use the "forget-me-not" variables to make some very promising reductions in the number of iterations needed if only the ORLSMS or the Jacobi method with subspace correction is used. Our results hint that perhaps the choice we advocate for least squares may be useful back in the general minimization setting considered in [4].

If the size of the problem and the number of processors indicate, the problem in the surrogate variables can in turn be attacked by the partitioning technique of the first stage, and this can be continued to reduce the dimension of the problem to be solved in the synchronization step until the number of surrogate variables is manageable.

In the next section, we present some preliminaries to set the stage for the new Jacobi-Ferris-Mangasarian algorithm in Section 3. Section 4 presents some promising numerical results, and Section 5 is devoted to a discussion and conclusions concerning this approach.

# 2 Preliminaries

## 2.1 The linear least squares problem

Let $A$ be an $m \times n$ real matrix, $m \geq n, b \in \mathbf{R}^m$. Let $M$ be an $m \times m$ positive definite weighting matrix.

The weighted linear least-squares problem is:

$$\min_{x \in \mathbf{R}^n} \|Ax - b\,\|_M, \quad \text{where} \quad \|y\|_M^2 = y^T My. \tag{1}$$

Let the columns of $A$ be partitioned into $g$ blocks $A = [A_1\ A_2\ \ldots\ A_g]$, where $A_i$ is $m \times n_i$. Further let $x$ be partitioned consistently into blocks $x_1, x_2, \ldots, x_g$. The least squares problem (1) is equivalent to

$$\min_{x \in \mathbf{R}^n} \{\| \sum_{i=1}^{g} A_i x_i - b\|_M : x_i \in \mathbf{R}^{n_i}, i = 1, \ldots, g\}. \tag{2}$$

We want to distribute the variables to the available processors and solve a smaller subproblem on each processor in parallel.

## 2.2 Variable distribution

To solve the weighted linear least squares problem (2) we distribute the variables among the available processors. In this section we will assume that each group is assigned to its own processor.

Let $x^k$ be an approximation to the solution $x^*$ to (1), and partition $x^k$ into $x_1^k, x_2^k, \dots, x_g^k$.

**Parallelization:** $i = 1, 2, \dots, g$

$$\text{Solve for } x_i^{k+1} \in \mathbf{R}^{n_i} : \min_{x_i \in \mathbf{R}^{n_i}} \|A_i x_i - (b - \sum_{1=j\neq i}^{g} A_j x_j^k)\|_M \ . \tag{3}$$

Following the notation and derivation in [3], we introduce the direction $d_i^k = x_i^{k+1} - x_i^k$, and note that successive residuals satisfy

$$r^{k+1} = \sum_{j=1}^{g} A_j x_j^{k+1} - b = r^k + \sum_{j=1}^{g} A_j d_j^k \ .$$

Then the $i$th least squares subproblem (3) is:

$$\text{Solve for } d_i^k \in \mathbf{R}^{n_i} : \min_{d_i \in \mathbf{R}^{n_i}} \{\|A_i d_i + r^k\|_M\}, i = 1, 2, \dots, g, \tag{4}$$

and the $i$th block of the new approximate solution is

$$x_i^{k+1} = x_i^k + d_i^k, i = 1, 2, \dots, g. \tag{5}$$

For $d_i \in \mathbf{R}^{n_i}$ introduce the vector $\bar{d}_i \in \mathbf{R}^n$ which is obtained by starting with a zero vector and placing the nonzero entries of $d_i$ in the positions corresponding to the column indices in $A$ of $A_i$. Define the direction $d^k$:

$$d^k = \sum_{j=1}^{g} \bar{d}_j^k.$$

Then (5) can be written as $x^{k+1} = x^k + d^k$. This is [1] the classical block Jacobi method on the normal equations

$$A^T M A x = A^T M b. \tag{6}$$

Assume that $A$ has full rank. Then the following result says that the block Jacobi method converges if $A^T M A$ is sufficiently 'block diagonally dominant'.

**Theorem 1** *Let $A$ have full rank, and let $C$ be a block diagonal matrix with ith block $A_i^T M A_i$. The corresponding block Jacobi method will converge to $x^*$, a solution of (1) if $2\,C - A^T M A$ is positive definite.*

**Proof:** Corollary 2.1 of [5].

Even when $2\,C - A^T M A$ is not positive definite, we can force convergence by the following two small modifications of the block Jacobi method.

Let $f : \mathbf{R}^n \to \mathbf{R}$ be defined by

$$f(x) = \|Ax - b\|_M^2 = x^T A^T M A x - 2(A^T M b)^T x - b^T M b\ . \tag{7}$$

We can force convergence by introducing a simple linesearch:

$$\sigma_k = \ \text{argmin}_\sigma f(x^k + \sigma d^k)\ . \tag{8}$$

.

**Theorem 2** *Let $A$ have full rank. Given $x^k$, choose*

$$x^{k+1} = x^k + \sigma_k d^k\ ,$$

*where $\sigma_k$ is defined by (8). Then $lim_{k\to\infty} x^k = x^*$ .*

**Proof:** Chapter 6 of [2].

Of course, this linesearch functions as a synchronization step, and so the attractive parallelism in the Jacobi iteration is compromised.

Note that $\sigma_k$ is the easy solution of the 1-dimensional least squares problem to solve for $\sigma_k$ in $\|(Ad^k)\sigma_k + r^k\|_M$. We will introduce a more general linesearch in the next section.

Finally, we end this section with a simple convergence result that follows from the application of a degenerate form of the Ferris-Mangasarian 2nd stage.

**Theorem 3** *Let $A$ have full rank. Given $x^k$, choose*

$$x^{k+1} = \ argmin\{f(x^k + d^k), f(x^k + \bar{d}_i^k), i = 1, \ldots, g\} \ , \tag{9}$$

*then $lim_{k\to\infty} x^k = x^*$, where $x^*$ is the unique solution of (1).*

**Proof:** Since $A$ is full rank, $f$ is strongly convex and the result follows from Theorem 2.3 of [4] or Theorem 5 of the next section.

# 3 The Ferris-Mangasarian Correction Step

In the last section, we saw that the Jacobi iteration converges when the block cross terms in the coefficient matrix $A^T MA$ are weak enough to be neglected. This section will introduce simple techniques for incorporating the influence of these cross terms into the iteration. Unfortunately, these all will take the form of a synchronization step and so parallelism will be compromised.

Mangasarian [6] and Ferris and Mangasarian [4] introduced a synchronization step in which the step $\Delta x^k = x^{k+1} - x^k$ is chosen by approximately minimizing $f(x^k + \Delta x)$ in the subspace spanned by $\bar{d}_i$ $i = 1, \ldots, g$. In the following, we call this a subspace-correction step.

Thus, the block Jacobi iteration can be seen as choosing adaptively a single surrogate variable $d_i^k$ to represent the subspace spanned by the $i$th block of variables $x_i^k$ in the correction step. The subspace-correction iteration step is then chosen to be the step that provides approximately the most decrease from $x^k$ for $f$ in the space of Jacobi-surrogate variables. This sort of dimensional reduction is common in engineering design through so-called surrogate variable or reduced basis techniques. The difference here is that the surrogate variables are being chosen adaptively by the Jacobi iteration rather than to be chosen *a priori* by engineering judgement.

## 3.1 Supplementary variables

The subspaces spanned by the column blocks $A_i$ can be supplemented by what Ferris and Mangasarian call "forget-me-not" variables. For our setting, a more appropriate name would be "look-ahead" variables. Thus, we will use the more neutral designation "supplementary variables".

Beginning with a single full space vector, the procedures of the previous section are used to obtain supplementary variables to expand each subspace. Unfortunately, this requires us to introduce still more complicated notation, which we will give now. Then we will discuss strategies for choosing supplementary variables that we have found to be so advantageous as to justify the added fuss.

Let $I_i$ be the $n_i \times n_i$ identity matrix and let $\bar{I}_i$ be the $n \times n_i$ matrix formed from columns of the $n \times n$ identity matrix so that $A\bar{I}_i = A_i$. Let the supplementary vector $p \in \mathbf{R}^n$ be partitioned accordingly and define the $n \times (g-1+n_i)$ matrix $P_i$

$$P_i = \left[\bar{p}_1 \cdots \bar{p}_{i-1} \ \bar{I}_i \ \ \bar{p}_{i+1} \cdots \bar{p}_g\right] \ . \tag{10}$$

For $\tilde{n}_i = n_i + g - 1$, define the $m \times \tilde{n}_i$ matrix

$$\widetilde{A}_i = AP_i = \left[A\bar{p}_1 \cdots A\bar{p}_{i-1} \ A_i \ A\bar{p}_{i+1} \cdots A\bar{p}_g\right]. \tag{11}$$

For a given supplementary vector $p^k \in \mathbf{R}^n$ the $g$ subproblems (3) are replaced by

$$\text{Solve for } \widetilde{d}_i^k \in \mathbf{R}^{\tilde{n}_i} : \min_{\widetilde{d}_i}\{\|\widetilde{A}_i^k \widetilde{d}_i + r^k\|_M\} \tag{12}$$

where $\widetilde{A}_i^k$ is defined in (11) for the given vector $p^k$. The step $d^k \in \mathbf{R}^n$ is

$$d^k = \sum_{i=1}^{g} P_i^k \widetilde{d}_i^k \tag{13}$$

Of course, the inclusion of $p$ in the algorithmic mix raises the question of how to choose an ideal $p$ for the iteration. That question turns out to have a simple answer, which we give in the following theorem and then follow with some algorithmic modifications aimed at approximating the ideal $p$.

**Theorem 4** *Let $x^k \in \mathbf{R}^n$ be arbitrary, and set $p^k = e^k = x^* - x^k$. Then, each $P_i\widetilde{d}_i^k = e^k$, and $x^* = x^k + \frac{1}{g}d^k$.*

**Proof**: To simplify notation, we will consider the case $i = 1$. Let $v \equiv ({e_1^k}^T, 1, 1, \cdots, 1)^T \in \mathbf{R}^{\tilde{n}_1}$. First we will show that $\widetilde{d}_1^k = v$ solves (12).

Notice that

$$\widetilde{A}_1^k = AP_1 = [A_1 \ A_2 e_2^k \cdots A_g e_g^k] \ ,$$

and $P_1 v = e^k$. Thus,

$$\widetilde{A}_1^k v + r^k = AP_1 v + r^k = Ae^k + r^k = Ax^* - b \ ,$$

and if $\widetilde{d}_1^k$ is any other solution, then it must give the same residual. Thus, by the unicity of $x^*$,

$$P_1 \widetilde{d}_1^k = P_1 v = e^k$$

is unique.

So, if we could choose $p^k = x^* - x^k \equiv e^k$, then *each* $P_i \widetilde{d}_i^k$ would be $e^k$. Of course, if we knew $e^k$, we would be finished, but this points to taking $p^k$ to be our best estimate of $e^k$. The best way we have thought to do this at a particular iteration is by taking $p^k = x^k - x^{k-1}$, and even this crude approximation to $e^k$ leads to a significant reduction in iterations.

However, a more elaborate scheme is reasonable because if $p^k$ does not depend on $k$, and if the subproblems are solved using a Cholesky factorization of the $n_i \times n_i$ matrix $\widetilde{A}_i^T M \widetilde{A}_i$, then the Cholesky factors are saved and solving the subproblems require only a back substitution (forward and backward substitution).

This suggests that we might profitably exploit the linear algebra savings to try a predictor/corrector scheme defined by keeping $p = p^{k-1}$ fixed for several *predictor* iterations to obtain say $x^{\text{pred}}$ without having to redo any factorizations. The sole purpose of these predictor iterations is to obtain a better approximation $x^{\text{pred}} - x^k \approx e^k$ to use as $p^k$ in a *corrector* iteration to obtain $x^{k+1}$. We will give numerical results supporting this procedure.

## 3.2 The complete algorithm

At this point, we have obtained the full set of supplementary variables from the block Jacobi subproblems supplemented by the projections of $p$. To finish specializing the Ferris-Mangasarian technique to linear least squares, we will explain the subspace-correction step, and then we will give the complete algorithm.

For a given $d \in \mathbf{R}^n$, define the $n \times g$ matrix

$$D = \left[\bar{d}_1 \cdots \bar{d}_g\right]. \tag{14}$$

Consider the $m \times g$ matrix

$$\widehat{A} = AD = \left[A\bar{d}_1 \cdots A\bar{d}_g\right].$$

Then the columns of $\widehat{A}$ are the full set of surrogate variables. We solve the least squares problem in this set of variables to get the subspace corrected step, which is given by

$$\text{Solve for } s^k \in \mathbf{R}^g : \min \|\widehat{A}s^k + r^k\|_M \ . \tag{15}$$

We use the vector $d^k$ defined by (13), and the new iterate is $x^{k+1} = x^k + D^k s^k$ where $D^k$ is defined in (14).

Before we give the Ferris-Mangasarian convergence theorem for the more general nonlinear optimization algorithm, we pause to sum up all the specializations we have suggested in the following

**Algorithm**: Jacobi-Ferris-Mangasarian

Subdivide $A$ into $g$ blocks.
Choose $x^0$.
Compute $r^0 = Ax^0 - b$.
**for** $k = 0$ **step** 1 **until** *convergence* **do**
  Choose vector $p^k$. *This may involve several predictor iterations*
  **for** $i = 1, \ldots, g$ **in parallel**
    Compute $P_i^k$ in (10).
    Let $\widetilde{A}_i^k = AP_i^k$
    Solve for $\widetilde{d}_i^k : \min\{\|\widetilde{A}_i^k \widetilde{d}_i^k + r^k\|_M\}$.
  Compute $d^k = \sum_{i=1}^g P_i^k \widetilde{d}_i^k$.
  Compute $D^k$ in (14) and $\widehat{A}^k = AD^k$.
  Solve for $s^k : \min \|\widehat{A}^k s^k + r^k\|_M$.
  $x^{k+1} = x^k + D^k s^k$.
  $r^{k+1} = r^k + \widehat{A}^k s^k$.
  Check for *convergence*.

**Algorithm**: Predictor iterations

Let $\widetilde{A}_i = AP_i^{k-1}$ and $P_i = P_i^{k-1}$ for $i = 1, 2, \ldots, g$.
Let $z^0 = x^k - x^{k-1},\ v^0 = r^k + Az^0$.
**for** $j = 0, 1, \ldots, l-1$ **do**
  **for** $i = 1, \ \ldots, g$ **in parallel**
    Solve for $\widetilde{d}_i^j : \min\{\|\widetilde{A}_i \widetilde{d}_i^j + v^j\|_M\}$.
  Compute $d^j = \sum_{i=1}^{g} P_i \widetilde{d}_i^j$.
  Compute $D^j$ and $\widehat{A}^j = AD^j$.
  Solve for $s^j : \min \|\widehat{A}^j s^j + v^j\|_M$.
  $z^{j+1} = z^j + D^j s^j$.
  $v^{j+1} = v^j + \widehat{A}^j s^j$.
Let $p^k = z^l$

The following result follows from Ferris and Mangasarian [4] Theorem 2.3 by noticing that if $A$ has full rank then the function $f$ in (7) is strongly convex.

**Theorem 5** *Assume that $\{p^k\}$ is bounded independent of $k$. If $A$ has full rank, then $\lim_{k\to\infty} x^k = x^*$, where $x^*$ is the unique solution of (1).*

# 4 Numerical results

The convergence of the methods will be illustrated on a class of randomly generated least squares problems (1). The $m \times n$ coefficient matrix $A = QD + \varepsilon R$ where $Q$ is $m \times n$ with orthonormal columns, $D$ is a $n \times n$ diagonal matrix and $R$ is a $m \times n$ matrix. The elements in $R$ and on the diagonal of $D$ are randomly distributed. For small values of $\varepsilon$ the matrix $A^T A$ is diagonally dominated. The elements in the $m$ vector $b$ in the least squares problem (1) are either random (for 'non-zero' residual case problems) or the vector is chosen to be $b = Ac$ where $c$ is a random $n$ vector for 'zero residual' problems. The weight matrix $M$ is the identity matrix.

All tests are run on a SPARC with Sun-4 floating-point using Matlab.

In Table 1, variations of the Ferris and Mangasarian technique are compared to the Gauss-Seidel iteration and the Jacobi iteration with the subspace-corrected step on the normal equations (6). For this problem $D \equiv 0$, $\varepsilon = 1$, and the elements in $R$ are uniformly distributed in [-1,1]. Note that this problem gets more diagonally dominant as $m$ increases. For the particular case reported here $m = 280$, $n = 256$. The condition number (the square of the ratio of largest and smallest singular values of $A$) is 1403.5. The block Jacobi method does not converge without a linesearch or subspace correction for the reported values of $g$.

The stopping criterion is that the $\ell_2$ difference between the exact and approximate solutions is not more than $10^{-6}$.

In all numerical experiments the columns of the matrix are partioned into $g$ groups based on the natural order: the 1st $n/g$ columns form the first group, the 2nd $n/g$ columns form the next group, etc. The starting point is $x^0 = 0$.

| g | Gauss-Seidel | Jacobi with subspace correction | Jacobi-Ferris-Mangasarian | | | |
|---|---|---|---|---|---|---|
| | | | p=1 | F&M | p=Ds | Pred |
| Zero residual: | | | | | | |
| 4 | 1891 | 5532 | 4919 | 4499 | 1835 | 676 |
| 8 | 2222 | 7157 | 11250 | 13411 | 1073 | 478 |
| 32 | 2486 | 6812 | >20000 | >20000 | 322 | 2955 |
| Non zero residual: | | | | | | |
| 4 | 1881 | 5223 | 5473 | 4275 | 1843 | 561 |
| 8 | 2244 | 6681 | 12249 | 12808 | 950 | 447 |
| 32 | 2522 | 6947 | >20000 | >20000 | 395 | 3000 |

Table 1: Number of iterations

The columns in Table 1 give the number of iterations to achieve the desired accuracy. The Guass-Seidel method is applied to the normal equations without forming the normal equations (see for example [1, 3]). For the Jacobi method we use the subspace corrected step (15) to guarantee convergence.

The column marked "p=1" are the iterations using the supplementary variables defined by $p^k = (1, \ldots, 1)^T \in \mathbf{R}^n$.

Ferris and Mangasarian [4] suggest using the vector $p^k$ to be

$$
\begin{aligned}
[p_i^k]_j &= \frac{1}{[(\nabla f(x^k + e_i) - \nabla f(x^k))_i]_j} \\
&= \frac{1}{[A_i^T M A_i e_i]_j} \text{ for } j = 1, 2, \ldots, n_i, \; i = 1, 2, \ldots, g
\end{aligned} \tag{16}
$$

where $e_i \in \mathbf{R}^{n_i}$ is a vector with all ones and $[\,\cdot\,]_j$ denotes the $j$th component of a $n_i$ vector. Note that this $p^k$ does not depend on the iteration index $k$. The column marked "F&M" are the iterations using the supplementary variables defined by (16). All tests indicate very little difference between the Ferris and Mangasarian choice of $p^k$ and $p^k = (1, \ldots, 1)^T$.

The number of iterations for the algorithm that uses supplementary variables defined by choosing $p^k = x^k - x^{k-1} = D^k s^k$ is in the column marked "p=Ds". In the column marked "Pred" the supplementary variables are determined using one predictor step to compute the new $p^k$.

All tests indicate very small variations between zero and non-zero residual cases. This small difference indicates that the governing condition number for the methods is the condition number of the normal equation (6).

For $p^k = x^k - x^{k-1}$ the number of iterations does not increase with the number of groups $g$ on most problems. However, for the method that uses one predictor step, we see that the number of iterations in some cases increases with the number of groups. This is investigated further in Table 2. Here we have chosen $\varepsilon = 1$, the diagonal elements in $D$ are uniformly distributed in [1,2] and the elements of $R$ are in the interval [0,1]. Further $m = 26$ and $n = 24$.

A predictor iteration has the same cost in terms of arithmetic operations as one iteration of the algorithm with $p^k$ independent of the iteration index. If the cost of computing the QR factorizations of the smaller systems is neglected, then using $l$ predictor iterations has the same cost as $l+1$ iterations using "p=DS". If we consider $g = 8$ in Table 2 we see that it is more efficient to use 2 or 3 predictor iterations than use $p^k = x^k - x^{k-1}$. If we use $l = 2$ predictor iterations and compare with the results in Table 1 for "Pred" ($l = 1$) the number of iterations decreases from 2955 to 306 for the zero residual case and $g = 32$. For the non zero residual case the number of iterations decreased from 3000 to 444 when $l = 2$.

| g | p=Ds | Predictor iterations $l$ | | | |
|---|---|---|---|---|---|
| | | $l=1$ | $l=2$ | $l=3$ | $l=4$ |
| 2 | 897 | 381 | 77 | 80 | 53 |
| 4 | 2424 | 879 | 185 | 397 | 95 |
| 6 | 1488 | 343 | 176 | 450 | 135 |
| 8 | 1421 | 478 | 257 | 567 | 95 |
| 12 | 336 | 2603 | 217 | 313 | 104 |

Table 2: Number of iterations and predictor iterations

# 5 Conclusions

This paper raises more questions than it answers, and we hope to pursue some of these questions soon. We believe that we have found a valuable choice of the "forget-me-not" variables of Ferris and Mangasarian, and the behavior of the predictor/corrector fairly cries out for an adaptive way to decide when how many predictor iterations to do. At this point, we can only say that more groups means more predictor iterations.

It would be also interesting to know whether our choice would be useful in the general nonlinear optimization case. We suspect it would.

These questions will have to wait in order that we can make the deadline to have our paper considered for the issue to honor Olvi Mangasarian on his 65th birthday. We join all of Olvi's friends, not just contributers to this volume, in wishing Olvi and Claire many more happy and healthy years.

# References

[1] Å. Björck, *Numerical methods for least squares problems*, SIAM, 1996.

[2] J. E. Dennis, Jr., Robert B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice-Hall, 1983.

[3] J. E. Dennis, Jr., and T. Steihaug, *On the successive projections approach to least squares problems*, SIAM J. Numer. Anal. 23 (1986) 717-733

[4] M. C. Ferris and O. L. Mangasarian, *Parallel variable distribution*, SIAM J. Optimization 4(1994) 815-832.

[5] H. B. Keller, *On the solution of singular and semidefinite linear systems by iteration*, SIAM J. Numer.Anal. 2(1965) 281-290.

[6] O. L. Mangasarian, *Parallel gradient distribution in unconstrained optimization*, SIAM J. Control and Optimization 33(1995)1916-1925.

[7] D. P. O'Leary and R. E. White, *Multisplitting of matrices and parallel solutions of linear systems*, SIAM J. Algebraic Discrete Methods, 6(1985) 630-640.

[8] J. M. Ortega, *Introduction to parallel and vector solution of linear systems*, Plenum Press, 1988.

[9] R. A. Renaut, *A parallel multisplitting solution of the least squares problem*, Numerical Linear Algebra with Applications, 4(1997) 1-21.