Derivative-free Gauss-Newton-like Algorithm for Parameter Estimation

Sergei Scherbov
Viktor Golubkov

November 1986
WP-86-63

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

Foreword

The Population Program at IIASA deals with the analysis of consequences of demographic changes. To estimate the consequences one needs to analyze the data from various sources, to develop the models, and to identify their parameter.

This paper by Dr. Scherbov and Dr. Golubkov develops the idea of parameter estimation using the derivative-free nonlinear least-squares algorithm. The algorithm was found efficient and convenient for many applications.

Anatoli Yashin
Deputy Leader
Population Program

Derivative-free Gauss-Newton-like Algorithm for Parameter Estimation

Sergei Scherbov and Viktor Golubkov

Institute for Systems Studies (VNIISI)
Prospect 60 Let Octyabria, 9
117312 Moscow
USSR

An important problem of parameter estimation of different models using statistical data about modeled process very often can be reduced to the least-squares problem. In this paper the derivative-free nonlinear least-squares algorithm, which was found very efficient in the sense of response function calculations, is presented.

Let $\tilde{\psi}_i$ $(i = \overline{1,N})$ be the components of an observed data vector $\tilde{\psi} = (\tilde{\psi}_1, \tilde{\psi}_2, \ldots, \tilde{\psi}_N)^T$, and $\psi_i(q)$ $(i = \overline{1,N})$ be the components of a vector valued response function $\psi(q) = (\psi_1(q), \psi_2(q), \ldots, \psi_N(q))^T$. Let $q = (q_1, q_2, \ldots, q_n)^T$ be the vector of estimated parameters. Then, according to the Generalized Least Squares Method (GLSM), the estimate $\tilde{q} = (\tilde{q}_1, \tilde{q}_2, \ldots, \tilde{q}_n)^T$ could be found as a solution to the following nonlinear programming problem

$$\tilde{q} = \operatorname*{Argmin}_{q} F(q) \quad , \quad F(q) = \langle P_\xi \xi(q), \xi(q) \rangle \tag{1}$$

$$\xi(q) = \tilde{\psi} - \psi(q) \quad , \quad P_\xi^T = P_\xi \geq 0 \quad ,$$

where P_ξ is a given symmetric, positively half-defined square matrix of weights, and $\langle a, b \rangle$ means scalar production.

Gauss-Newton-like algorithms are known to be the most efficient iterative methods for solving problem (1). According to those algorithms the linear approximation of $\xi(q)$ about the current value of parameter vector g is calculated at each iteration, and a linear least-squares problem is solved to obtain a new value of parameter q.

In the presented algorithm, as in many other derivative-free Gauss-Newton-like algorithms [1,2,3,4], the linear approximation of $\xi(q)$ is evaluated according to $n+1$ values of $\xi(q)$ calculated at the previous iterations in order to estimate

the new parameter vector and to pass the updated set of parameters to the next iteration. Gauss-Newton-like algorithms differ from each other generally by the linear approximation of $\xi(q)$ and by use of the information obtained at the previous iterations. Let us dwell upon the most important features that distinguish the presented algorithms among others of the same type.

Let us assume that on the k-th iteration we have $q_1^k, q_2^k, \ldots, q_{n+1}^k$ —estimates of solution computed at the previous iterations and a correspondent set of $\xi_1^k, \xi_2^k, \ldots, \xi_{n+1}^k$, which are necessary for the linear approximation of $\xi(q)$, where $\xi_i^k = \xi(q_i^k)$. The upper index shows the iteration number. Let us assume on the first iteration $F_{n+1}^0 = \min\limits_{1 \le i < n+1} F_i^0$, where $F_i^0 = F(q_i^0)$.

The new estimate of parameters q on the k-th iteration q_{new}^k will be

$$q_{new}^k = q_{n+1}^k + \Delta q_H^k \tag{2}$$

$$\Delta Q^k = (\Delta q_1^k, \Delta q_2^k, \ldots, \Delta q_n^k) \ , \ \Delta q_i^k = q_{i+1}^k - q_i^k \ , \ i = \overline{1,n} \ ,$$

where ΔQ^k is a $n \otimes n$ matrix of parameter increments and Δq_H^k is a vector that will be evaluated on the k-th iteration.

The linear approximation $\gamma(q)$ of the vector $\xi(q)$ could be written in a form

$$\gamma^k(q) = \xi_{n+1}^k + (\frac{\Delta\Sigma}{\Delta Q})^k (q - q_{n+1}^k) \tag{3}$$

whose matrix $(\frac{\Delta\Sigma}{\Delta Q})^k$ is derived from $n+1$ previously calculated values q_i^k, ξ_i^k, $(i = \overline{1,n+1})$ to satisfy conditions

$$\xi_i^k = \xi_{n+1}^k + (\frac{\Delta\Sigma}{\Delta Q})^k \Delta q_i^k \ , \ i = \overline{1,n+1} \ .$$

Supposing nonsingularity of matrix ΔQ^k, we obtain

$$(\frac{\Delta\Sigma}{\Delta Q})^k = \Delta\Sigma^k (\Delta Q^k)^{-1} \tag{3'}$$

$$\Delta\Sigma^k = (\Delta\xi_1^k, \ldots, \Delta\xi_n^k) \ , \ \Delta\xi_i^k = \xi_{i+1}^k - \xi_i^k \ , \ i = \overline{1,n} \ .$$

The closer q_i^k are to each other, the higher is the accuracy of approximation (3). Vector Δq_H^k is a solution of

$$\Delta q_H^k = \operatorname{Agrmin}_q \langle P_\xi \gamma^k(q), \gamma^k(q) \rangle$$

and is calculated from

$$\Delta q_H^k = -(A^k)^{-1}B^k \tag{3''}$$

$$A^k = (\frac{\Delta\Sigma}{\Delta Q})^{k\,T} P_\xi (\frac{\Delta\Sigma}{\Delta Q})^k \; ; \; B^k = (\frac{\Delta\Sigma}{\Delta Q})^{k\,T} P_\xi \xi_{n+1}^k \quad .$$

The new $(k+1)$-th estimation q_{n+1}^{k+1} is

$$q_{n+1}^{k+1} = q_{n+1}^k + h^k \Delta q_H^k$$

where h^k is a step length along Δq_H^k, a direction which is obtained from a one-dimensional function minimization $F^k(h) = F(q_{n+1}^k + h\,\Delta q_H^k)$ or under the condition that $F(q_{n+1}^{k+1}) < F(q_{n+1}^k)$. In the present version of the algorithm, a one-dimensional minimization was based on a second-order approximation of $F^k(h)$ and use of the well-known fact that in regular cases h^k, which minimizes $F^k(h)$, approaches to a value close to 1 during the Gauss-Newton algorithm's convergency.

The $(k+1)$-th iteration begins with the calculation of $\xi(q^{k+1})$ and then the $\Delta\Sigma^{k+1}$ and ΔQ^{k+1} matrices are calculated. Before describing these matrix calculations we should mention that during the iterative process the non-singularity of $\overline{\Delta Q}^k$ must be controlled, where

$$\overline{\Delta Q}^k = (\overline{\Delta q}_1^k, \overline{\Delta q}_2^k, \ldots, \overline{\Delta q}_n^k), \; \overline{\Delta q}_i^k = \Delta q_i^k / \,||\Delta q_i^k||, \; i = \overline{1,n} \tag{3'''}$$

and $||\Delta q_i^k||$ is a standard Euclidian norm of the vector Δq_i^k.

That is because ill conditionality of $\overline{\Delta Q}^k$ leads to ill conditionality of ΔQ^k and $\Delta\Sigma^k$, and consequently of A^k, which, in turn, leads to the worsening of the algorithm's convergency.

Let E_d be the minimal feasible value of $|det\ \overline{\Delta Q}^k|$ when convergency is still normal. Then ΔQ^{k+1} $(k \geq 1)$ should be constructed in a way that

$$|det\ \overline{\Delta Q}^{k+1}| \geq E_d \;\; \text{when} \;\; k \geq 1;\; 0 < E_d < 1 \quad . \tag{4}$$

Let us assume that (4) is true. Then ΔQ^{k+1} building starts with changing in ΔQ^k the column whose number is l_k according to:

$$\Delta q_{l_k}^{k+1} = q_{n+1}^{k+1} - q_{n+1}^k = h^k \Delta q_H^k \;\; , \;\; q_{l_k}^{k+1} = q_{n+1}^k \tag{5}$$

$$\Delta q_i^{k+1} = \Delta q_i^k \;\; , \;\; q_i^{k+1} = q_i^k \;\; , \;\; i \neq l_k, \; i = \overline{1,n}$$

where l_k is defined from

$$l_k = \underset{1 \le i \le n}{\text{Argmin}} \, |S_i^k| \quad . \tag{5'}$$

The values of S_i^k ($i = \overline{1,n}$) are scalar coefficients in the representation of Δq_H

$$\Delta q_H = \sum_{i=1}^{n} S_i^k \Delta q_i^k$$

which may be obtained from (3'), (3''), and (3''').

Such a choice of l_k provides the best conditionality of $\overline{\Delta Q}^{k+1}$. This could be easily derived from

$$|\det \overline{\Delta Q}^{k+1}| = \frac{|S_{l_k}^k|}{||\Delta q_H^k||} |\det \overline{\Delta Q}^k| \quad .$$

From this expression and from (5) it follows that in ΔQ^k the column whose changing provides the best conditionality of $\overline{\Delta Q}^{k+1}$ and is substituted by $h^k \Delta q_H^k$. If $|\det \overline{\Delta Q}^{k+1}| \ge E_d$, then $\Delta \Sigma^{k+1}$ is constructed by changing one column with number l_k according to:

$$\Delta \xi_{l_k}^{k+1} = \xi(q_{n+1}^{k+1}) - \xi(q_{n+1}^k)$$

$$\Delta \xi_i^{k+1} = \Delta \xi_i^k \ , \ i \ne l_k, \ i = \overline{1,n} \quad .$$

One of the distinctions of the presented algorithm from other variants of derivative-free Gauss-Newton-like algorithms is the choice of the column in ΔQ^k that will be substituted by $q_{n+1}^{k+1} - q_{n+1}^k$ after the k-th iteration. In those algorithms the column that was calculated before the other (on iteration with the smallest number) and hence usually corresponds to the values of parameters which differ mostly from their current values is substituted by $q_{n+1}^{k+1} - q_{n+1}^k$, or which is the same as $(h^k \Delta q_H^k)$. Thus the vectors of parameters which contribute to the computation of ΔQ^k are all the time "pulled up" to their latest value. It means that the columns of ΔQ^k are generally calculated with the parameters which are close to their current value.

In the algorithms mentioned above, all the columns of ΔQ will be renewed during n iterations and the approximation correctness (3) will take place. For practically widely-spread objective functions with long valley, their lead surface in parameter space is badly stretched along some directions. It is a well-known fact that descent directions for these functions are early tangent to level surface, and therefore they are very close to the direction in which level surface is stretched.

Thus, while iteration number k increases it can occur that the columns of $\overline{\Delta Q}^k$ and $\Delta\Sigma^k$, and the direction of descent Δq_H^k may be calculated incorrectly due to computational errors.

In the presented algorithm the worsening of conditionality of $\overline{\Delta Q}^k$ occurs essentially rarely than in many other algorithms of the same type. This was proved by numerical experiments. That is because in the $\overline{\Delta Q}^k$ of the algorithm presented here it is not the columns which were calculated earlier are substituted, as in [2,3,4], but rather those which provide the maximum linear independence of the columns in $\overline{\Delta Q}^{k+1}$. This is illustrated by Figures 1 and 2. In Figure 1, substitution of columns in $\Delta\bar{Q}^k$ is shown as it is always done in most of Gauss-Newton-like algorithms. In Figure 2, the same is shown for an algorithm presented here.

During functioning of the presented algorithm it could happen that some of the columns of ΔQ^k were not substituted for a long period. That means that in this case these columns were calculated with parameters that were far from their current values. In this case linear approximation (3) may not be valid, and descent direction Δq_H^k will be calculated incorrectly although condition (4) is satisfied. Getting rid of the algorithm operates in such a way that during a given number of iterations all the columns of ΔQ^k, and hence $\Delta\Sigma^k$ are renewed. This is achieved in the following way.

Let us assume we are at k's iteration and Δq_H^k has already been calculated. Let us denote n_i^k $(i = \overline{1,n})$ as the number of substitutions of the i-th column of ΔQ^k during k iterations, and I^k is the set of the column's numbers

$$I^k = \begin{cases} I_{N_g}^k, & \text{if } I_{N_g}^k \neq 0 \\ I_0, & \text{if } I_{N_g}^k \neq 0 \end{cases},$$

where

$$I_{N_g}^k = \{i : n_i^k \leq n_*^k - N_g,\ 1 \leq i \leq n\};\ n_i^0 = 0\ ,\ i = \overline{1,n}$$

$$I_0 = \{1,2,\ldots,n\}$$

$$n_*^k = \max_{1 \leq i \leq n} n_i^k\ ,\ k = 0,1,2,\ldots\ .$$

N_g—algorithm's control parameter which accepts integer values and satisfies the condition $N_g \geq 1$ ($N_g = 0$ corresponds to the case without "pulling up" the columns because in this case $I^k \equiv I^0$). Then the column with number l_k that will be substi-

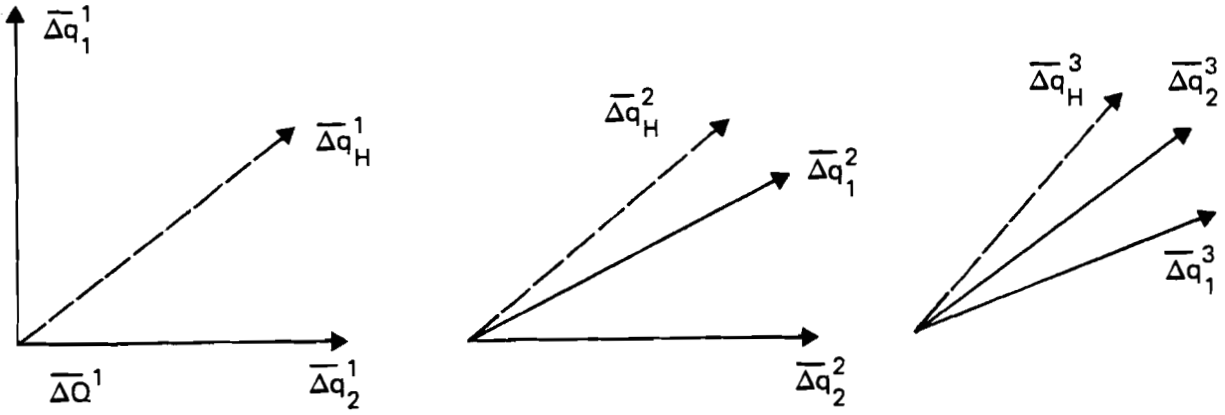

Figure 1.

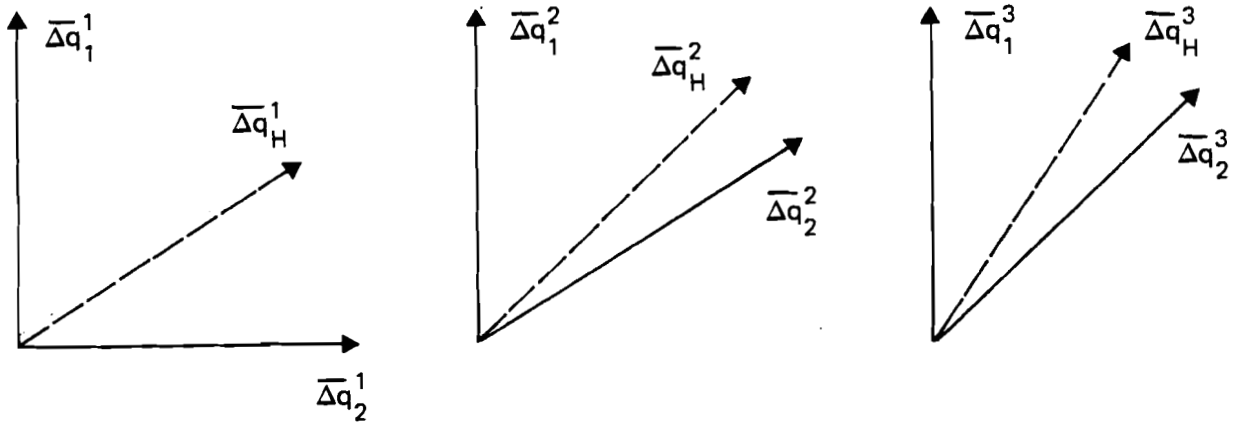

Figure 2.

tuted by $h^k \Delta g_H^k$ instead of (5) and (5') we shall find from:

$$l_k = \underset{i \in I^k}{\text{Argmax}} \, |S_i^k|$$

$$n_i^{k+1} = n_i^k, \; i \neq l_k \; , \; i = \overline{1,n}$$

$$n_{l_k}^{k+1} = n_{l_k}^k + 1$$

If after substitution of l_k column $|det \overline{\Delta Q}^{k+1}| \geq E_d$, then we pass to the $(k+1)$-th iteration as already described.

Otherwise the ortogonalization procedure is specified in the algorithm. (This is also the important feature of the algorithm.) According to this procedure each column Δq_i^{k+1}; $i \in I^k \setminus l_k$ of ΔQ^{k+1} is tested for substitution by Δg_i^{k+1} which is an ortogonal complement to the subspace spanned upon the other columns of $\overline{\Delta Q}^{k+1}$ and for each case the determinant of thus modified matrix is calculated. Vector Δg_i^{k+1} is calculated by:

$$\Delta g_i^{k+1} = ||\Delta g_i^{k+1}|| \, sgn < \Delta q^{k+1}, \overline{\Delta g}_i^{k+1} > \overline{\Delta g}^{k+1}$$

$$||\Delta g_i^{k+1}|| = \alpha \min_{1 \leq j \leq n} (|a_{ij}^{k+1}| / |\overline{\Delta g}_{ij}^{k+1}|)$$

$$a_{ij}^{k+1} = \begin{cases} \Delta q_{l_{kj}}^{k+1}, & \text{when } |\Delta q_{l_{kj}}^{k+1}| \geq \varepsilon_j \\ \beta b^{k+1} / (1 + b^{k+1}) & \text{when } |\Delta q_{l_{kj}}^{k+1}| < \varepsilon_j \end{cases}$$

$$b^{k+1} = \max_{1 \leq j \leq n} (|\Delta q_{l_{kj}}^{k+1}| / \varepsilon_j)$$

where α and β are positive control parameters of algorithm; $\varepsilon_j (j = \overline{1,n})$ is a vector of precisions for parameter estimation (iterative process stops when simultaneously $|q_{n+1,j}^{k+1} - q_{n+1,j}^k| \leq \varepsilon_j$, and $|\Delta q_{H_j}^k| \leq \varepsilon_j$; $j = \overline{1,n}$); $\overline{\Delta g}_i^{k+1}$ is a unit vector which is ortogonal to the other columns Δq_r^{k+1} $(r \neq i, \; r = \overline{1,n})$; and j is the vector's component number.

If (4) is true then substitution of corresponding column by ortogonal column is performed and the iteration process continues. If during examination of all the columns Δq_i^{k+1}, $i \in I^k \setminus l_k$, condition (4) could not be satisfied, the substitution that provides a maximum value of $|\det \overline{\Delta Q}^{k+1}|$ is made. Then the procedure is repeated but those columns of ΔQ^{k+1} are substituted by ortogonal columns which have not been substituted yet.

If all Δq_i^{k+1}, $i \in I^k \setminus l_k$ are substituted but condition (4) is still not satisfied, then it is allowed to examine for substitution the columns whose numbers initially were not included in I^k besides columns with number l_k. The condition (4) will be satisfied not more than for $n-1$ substitutions of columns of ΔQ^{k+1} by ortogonal as all columns in ΔQ^{k+1} are ortogonal. During ortogonalization after substitution of the i-th column in ΔQ^{k+1} by the corresponding ortogonal one Δg_i^{k+1}, the value $\xi(q^k + \Delta g_i^{k+1})$ is calculated and the i-th column of $\Delta\Sigma^{k+1}$ is substituted by $\xi(q^k + \Delta g_i^{k+1}) - \xi(q^k)$ and $n_i^{k+1} = n_i^k + 1$. It should be mentioned that while testing (4) during ortogonalization after substituting column by the corresponding ortogonal one, determinant $\det(\overline{\Delta Q}^{k+1})$ is calculated recursively. This reduces the computational efforts.

The $n+1$ initial values of parameters q required for the algorithm are generated from starting value q_{n+1}^0. For $i = \overline{1,n}$, q_i^0 is computed from q_{n+1}^0 by changing its i-th component by non-zero steps h_i. The corresponding set of ξ_i^0 $(i = \overline{1,n})$ is also calculated. The convergency of the algorithm presented here was analytically proved.

The comparative numerical analysis of this algorithm with several well-known and efficient derivative-free algorithms for minimization the sum of square was made. The total number of function evaluations to find minimum with a given precision was considered as efficiency criteria of algorithms.

This criteria is computer-independent and characterizes real computational expenditures in cases when $\xi(q)$ evaluation demands much more efforts than algorithm needs itself. In this paper we present some results of comparison of discussed algorithm with two derivative-free Gauss-Newton-like algorithms.

First is Powell's method for least-square problems. Its code was taken from the Harwell Subroutine Library (HSL). Its code is VA02A. The second method is a compromise between three different algorithms for minimizing a sum of squares, namely Newton-Raphson, Steepest Descent, and Marquardt. Its code is VA05A, also from HSL.

The last two were compared by many authors with different other algorithms and were discovered as being very efficient. Comparison of the discussed algorithm with VA02A and VA05A was performed on standard test problems found in the literature.

Four test problems were considered.

1. $F(q) = 100(q_1^2 - q_2) + (1 - q_1)^2$

   $\tilde{q} = (1,1)^T,\ F(\tilde{q}) = 0$

2. $F(q) = \sum_{i=1}^{10} [e^{-q_1 x_i} - e^{q_2 x_i} - q_3(e^{-x_i} - e^{-10x_i})]^2$

   $x_i = i/10$

   $\tilde{q} = (1,10,0.1)^T$ or $(10,1,-1)^T$ or $(q_0,q_r,0)^T$

   $F(\tilde{q}) = 0$

3. $F(q) = (10000q_1q_2 - 1)^2 + (e^{-q_1} + e^{-q_2} - 1.0001)^2$

   $\tilde{q} = (0.00001098,9.106)^T$ or $(9106,0.00001098)^T$

   $F(\tilde{q}) = 0$

4. $F(q) = (q_1 + 10q_2)^2 + 5(q_3 - q_4)^2 + (q_4 - 2q_3)^4 + 10(q_1 - q_4)^4$

   $\tilde{q} = (0,0,0,0)^T,\ F(\tilde{q}) = 0$

Here $\tilde{q}$ is a vector of parameters where minimum occurs. Minimization of functions 1-4 begins from different starting values. The obtained results are presented in Table 1. The algorithm described in this paper is denoted MMGN.

Here NFE is the number of function calculations; $\log(F)$ is the logarithm of convergency error: $\log(F) = \log F(q^{k_f})$ where k_f is the last iteration number. The symbol * means that the algorithm failed to find the solution.

From Table 1 it can be seen that our algorithm in all cases except one needs less function evaluations than the other two to converge with the same precision. Only in one case for function 1 and starting vector (-1.2,1) algorithm VA05A performed less function evaluations than MMGN. But, for example, for function 3 VA05A failed once, And VA02A failed all the times while MMGN succeeded. If we compare all presented results then VA05A and VA02A spend, on the average, 2.2 times more of function evaluations to find the solution than MMGN. (The cases where VA05A and VA02A failed are not taken into account.)

Table 1.

Method	NFE	log(F)	NFE	log(F)	NFE	log(F)	NFE	log(F)
Function 1	(-1.2,1)		(0,0)		(10,10)		(-1,-1)	
MMGN	43	$-\infty$	23	$-\infty$	13	$-\infty$	21	$-\infty$
VA05A	36	$-\infty$	25	$-\infty$	37	$-\infty$	26	$-\infty$
VA02A	80	-11	51	-12	48	-13	58	-13
Function 2	(0,20,20)		(0,20,10)		(0,20,0)		(0,10,10)	
MMGN	17	-15	18	$-\infty$	18	-15	13	-15
VA05A	48	-14	39	-14	34	-18	22	-15
VA02A	36	-14	37	-13	39	-12	31	-14
Function 3	(0,1)		(-1,1)		(0,-1)		(0,0)	
MMGN	35	-14	73	$-\infty$	119	-14	72	$-\infty$
VA05A	123	-10	143	-11	*	*	244	$-\infty$
VA02A	*	*	*	*	*	*	*	*
Function 4	(10,10,10,-10)		(10,10,10,10)					
MMGN	25	-15	35	-15				
VA05A	57	-15	57	-15				
VA02A	63	-15	63	-15				

Thus MMGN was found more efficient than algorithms for HSL library. This gives reason to expect that it will be more efficient than many of the algoritms that were compared with the latest two and were found less efficient. Besides as it follows from [6] the MMGN algorithm was more efficient in many cases for demographic data fitting in comparison with the algorithms from the IMSL library. In conclusion it should be noted that the presented algorithm was widely used for data processing and for parameter estimation in demographic, socio-economic, ecological and other models. Big practical experience dealing with this algorithm proved its reliability and efficiency.

REFERENCES

[1] Gelovani, V., V. Golubkov, and S. Scherbov (1979) *Parameter Estimation by Derivative-Free Analog of Gauss-Newton Method*, Vol. 8 (in Russian). Moscow: VNIISI.

[2] Ralston, M. and R. Jenrich (1978) Dud, a derivative-free algorithm for non-linear least squares. *Tecnometrics* Vol. 20, No. 1.

[3] Peckham, G. (1970) A new method for minimizing a sum of squares without calculating gradients. *Computer Journal*, No. 7.

[4] Vereskov, A. N. Levin, and V. Fedorov (1980) Regularised derivative-free least squares algorithm. *Issues of Cybernetics*, No. 71 (in Russian).

[5] Hoppez, M.J. (Ed.) (1978) *Harwell Subroutine Library, VKEA Report*. AERE-R-9185.

[6] Rogers, A. and F. Planck (1983) *A General Program for Estimating Parametrized Model Schedules of Fertility, Mortality, Migration, and Marital and Labor Force Status Transitions*. Working Paper WP-83-102. Laxenburg, Austria: International Institute for Applied Systems Analysis.