Control
and Cybernetics

VOL. 18 (1989) No. 2

Gauss-Newton versus Newton-Raphson algorithm for the recursive generalized least-squares identification

by

ROMAN WEINFELD
FRYDERYK Z. UNTON

Systems Research Institute
Polish Academy of Sciences
Newelska 6
01-447 Warsaw, POLAND

An exact on-line scheme for the Gauss-Newton iterative algorithm for the recursive generalized least-squares identification is proposed. Although in some sense close to the Newton-Raphson algorithm [5], it proves to behave considerably better in a number of cases.

1. Introduction

In recursive identification at each instant a new estimate of true parameters is required. In practice however, it is important that memory space and computation time does not increase with time. This makes it necessary to condense the data in certain way and imposes important restrictions upon algorithms. Computational difficulties caused that either the Newton-Raphson (N-R) algorithm for the recursive generalized least-squares (RGLS) identification considered in [2] and the Gauss-Newton (G-N) algorithm proposed in [3] and applied to the said problem use robust approximation of gradient and/or Hessian.

In [5] a new variant of N-R algorithm for the RGLS identification was proposed. Instead of the said approximations it uses exact recursive formulas for gradient and Hessian thus leading to the improvement of identification accuracy.

In this paper an exact recursive version of G-N algorithm is proposed. Numerical tests performed show that this algorithm in many cases gives better estimates than N-R algorithm.

2. RGLS Identification

Consider the discrete time asymptotically stable system of the form

$$A^*(q^{-1})\, y(t) = B^*(q^{-1})\, u(t) + \frac{1}{C^*(q^{-1})} e(t) \qquad t = 1, 2,... \quad (1a)$$

where

$$A^*(q^{-1}) = 1 + a_1^* q^{-1} + a_{na}^* q^{-na} \quad (1b)$$

$$B^*(q^{-1}) = b_1^* q^{-1} + ... + b_{nb}^* q^{-nb} \quad (1c)$$

$$C^*(q^{-1}) = 1 + c_1^* q^{-1} + ... + c_{nc}^* q^{-nc} \quad (1d)$$

and q^{-1} is backward shift operator, i.e. $q^{-1} y(t) = y(t-1)$, $y(t)$ is a scalar output, $u(t)$ — scalar input, $e(t)$ is a zero mean white noise independent with $u(t)$.

The problem is to estimate the vector of true parameters

$$\theta^* = [a_1^*,...,a_{na}^*, b_1^*,..., b_{nb}^*, c_1^*,...,c_{nc}^*]^T$$

using the observations $u(t)$, $y(t)$, $(t = 1, 2,...)$.

The off-line GLS algorithm which was introduced by Clarke [1] can be interpreted as the minimization with respect to θ of the following function [4]

$$v(t, \theta) = \frac{1}{2} \sum_{k=1}^{t} \varepsilon(k, \theta)^2 \quad (2a)$$

and

$$\varepsilon(k, \theta) = A(q^{-1})\, C(q^{-1})\, y(k) - B(q^{-1})\, C(q^{-1})\, u(k) \quad (2b)$$

The structure of polynomials $A(q^{-1})$, $B(q^{-1})$ and $C(q^{-1})$ is the same as their counterparts in (1) and

$$\theta = [a_1,..., a_{na}, b_1,..., b_{nb}, c_1,..., c_{nc}]^T$$

is the vector of their coefficients.

The N-R algorithm can be interpreted as follows. Let $\theta(t-1)$ be the estimate at time $t-1$. By means of a Taylor expansion of $V(t, \theta)$ around $\theta(t-1)$, $V(t, \theta)$ is approximated by

$$V(t,\theta) = V(t,\theta(t-1)) + [V'_\theta(t,\theta(t-1))]^{\mathrm{T}}[\theta-\theta(t-1)] +$$

$$+ \frac{1}{2}[\theta-\theta(t-1)]^{\mathrm{T}} V''_\theta(t,\theta(t-1))[\theta-\theta(t-1)] \quad (3)$$

($V'_\theta(t,\cdot)$ as well as all the gradients in the paper are column vectors).

As a result of minimization of expression (3) with respect to θ the new estimate is obtained

$$\theta(t) = \theta(t-1) - [V''_\theta(t,\theta(t-1))]^{-1} V'_\theta(t,\theta(t-1)) \quad (4)$$

Formula (4) defines the general N-R algorithm. It can be put into recursive form in a way explained in [5], or using the approximating scheme given in [2].

Another common way of approximating $V(t,\theta)$ is to exploit the idea of replacing $\varepsilon(k,\theta)$ in (2a) by its Taylor expansion around $\theta(t-1)$

$$\varepsilon(k,\theta) \cong \varepsilon(k,\theta(t-1)) - [\psi(k,\theta(t-1))]^{\mathrm{T}}[\theta-\theta(t-1)]$$

where $\psi(k,\theta) \equiv -\varepsilon'_\theta(k,\theta)$.

This results in the following approximation

$$V(t,\theta) = \frac{1}{2}\sum_{k=1}^{t}[\varepsilon(k,\theta(t-1)) - [\psi(k,\theta(t-1))]^{T}[\theta-\theta(t-1)]]^2 \quad (5)$$

The last expression is a function quadratic in θ. It is minimized by

$$\theta(t) = \theta(t-1) + \left[\sum_{k=1}^{t}\psi(k,\theta(t-1))\,\psi(k,\theta(t-))^{T}\right]^{-1} \cdot$$

$$\cdot \sum_{k=1}^{t}\psi(k,\theta(t-1))\,\varepsilon(k,\theta(t-1)) \quad (6)$$

The above algorithm is called Gauss-Newton (G-N) algorithm for nonlinear regression. In the next section it will be shown how $\theta(t)$ can be recursively calculated.

Notice that most often the following version of G-N algorithm is proposed (i.e. [3])

$$\theta(t) = \theta(t-1) + \left[\sum_{k=1}^{t}\psi(k,\theta(t-1))\,\psi(k,\theta(t-))^{T}\right]^{-1} \cdot$$

$$\psi(k,\theta(t-1))\,\varepsilon(k,\theta(t-1)) \quad (7)$$

Recursive version of (7) can be realized using the well known matrix inversion lemma. It turns out that the moderate difference in going from (7) to (6) makes important improvement in the algorithm's behaviour.

3. G-N algorithm

To establish the main result let us introduce the following notation

$$\varphi(t) = [-y(t-1),\ldots, -y(t-n_a-n_c),\ u(t-1),\ldots,\ u(t-n_b-n_c)]^{\mathrm{T}} \tag{8a}$$

$$R(t) = R(t-1) + \varphi(t)\varphi(t)^{\mathrm{T}} \qquad R(0) = 0 \tag{8b}$$

$$r(t) = r(t-1) + \varphi(t)y(t) \qquad r(0) = 0 \tag{8c}$$

$$A(\theta) = \begin{array}{c} \begin{matrix} \quad n_a + n_c \quad & \quad n_b + n_c \quad \end{matrix} \\ \left[\begin{array}{c|c} \begin{matrix} 1\ c_1 \cdots c_{n_c} \\ \vdots \\ 1\ c_1 \cdots c_{n_c} \end{matrix} & 0 \\ \hline 0 & \begin{matrix} 1\ c_1 \cdots c_{n_c} \\ \vdots \\ 1\ c_1 \cdots c_{n_c} \end{matrix} \\ \hline \begin{matrix} 1\ a_1 \cdots a_{n_a} \\ \vdots \\ 1\ a_1 \cdots a_{n_a} \end{matrix} & \begin{matrix} 0\ b_1 \cdots b_{n_b} \\ \vdots \\ 0\ b_1 \cdots b_{n_b} \end{matrix} \end{array}\right] \end{array} \begin{matrix} \\ n_a \\ n_b \\ n_c \end{matrix}$$

$$\hat{a}(\theta) = \underset{n_a+n_b+2n_c}{[c_1,\ldots, c_{n_c}, 0,\ldots, 0]^{\mathrm{T}}} \tag{8e}$$

$$\hat{b}(\theta) = \underset{n_a+n_b+n_c}{[a_1,\ldots, a_{n_a}, b_1,\ldots, b_{n_b}, 0,\ldots, 0]^{T}} \tag{8f}$$

$$x(\theta) = A^{T}(\theta)\hat{b}(\theta) + \hat{a}(\theta) \tag{8g}$$

Then the theorem holds:

THEOREM. *Let θ be any fixed vector. Then the componets of the algorithm (6)*

$$g(t, \theta) = \sum_{k=1}^{t} \psi(k, \theta(t-1))\, \varepsilon(k, \theta(t-1))$$

and

$$H(t, \theta) = \sum_{k=1}^{t} \psi(k, \theta(t-1))\, \psi(k, \theta(t-1))^T$$

can be recursively computed using the following formulas:

$$g(t, \theta) = A(\theta)\,(r(t) - R(t)\,x(\theta)) \tag{9}$$

$$H(t, \theta) = A(\theta)\,R(t)\,A^T(\theta) \tag{10}$$

Proof. The function $\varepsilon(k, \theta)$ can be rewritten as

$$\varepsilon(k, \theta) = T(x(\theta))$$

where

$$T(x) = y(k) - \varphi^T(k-1)\,x$$

and $x(\theta)$ given by (8g) is the vector of coefficients of $A(q^{-1})\,C(q^{-1})$ and $B(q^{-1})\,C(q^{-1})$.

The gradient $\psi(k, \theta)$ of $-\varepsilon(k, \theta)$ is given by

$$\psi(k, \theta) = A(\theta) \operatorname*{grad}_{x} T(x) = A(\theta)\,\varphi(k)$$

where $A(\theta)$ is the transposed Jacobian matrix of the vector function $x(\theta)$ (8g) with respect to θ. It is easy to see that it is given by (8d).
Now

$$H(t, \theta) = \sum_{k=1}^{t} \psi(k, \theta(t))\, \psi(k, \theta(t))^T = A(\theta)\,R(t)\,A^T(\theta)$$

and

$$g(t, \theta) = \sum_{k=1}^{t} A(\theta)\,\varphi(k)\,(y(k) - \varphi^T(k)\,x(\theta)) = A(\theta)\,(r(t) - R(t)\,x(\theta))$$

This ends the proof. ■

This is clear that the formulas (8), (9) and (10) make it possible to evaluate the next estimate in a recursive manner. The resulting one step of the algorithm

$$H(t, \theta(t-1))\,\Delta(t) = g(t, \theta(t-1)) \tag{11a}$$

$$\theta(t) = \theta(t-1) + \Delta(t) \tag{11b}$$

is in fact equivalent to the appropriate step of G-N stationary algorithm. Since the minimized function changes slightly in each instant both Hessian and gradient depend explicitly on t. This also constitutes the main difference between the stationary on-line G-N algorithm and the one described in the paper.

Note that there is only a slight difference between the final form of the G-N (11) and the N-R [5] algorithms, but as it will be shown in the next section this difference has an important influence on the transient behaviours of the algorithms.

4. Numerical example

To compare the behaviours of G-N and N-R algorithms we have repeatedly tested them.

The system simulated has the form (1) where

$$B^*(q^{-1}) = 1.0q^{-1} + 0.5q^{-2}$$

$$C^*(q^{-1}) = 1 - 1.0q^{-1} + 0.2q^{-2}$$

and $u(t) \sim N(0,1)$, $e(t) \sim N(0,\sigma^2)$, $\sigma = 0.4$

To improve the convergence in initial iterations the algorithms are started up in the following way: for the first K instants the estimates of $A^*(q^{-1})$ and $B^*(q^{-1})$ are calculated via the recursive least-squares algorithm while the estimates of $C^*(q^{-1})$ are assumed to be zero. $R(t)$ and $r(t)$ are of course constantly updated. In addition

$$A^*(q^{-1}) = 1 - 1.5q^{-1} + 0.7q^{-2}, \quad \text{No} = 70 \text{ for example 1}$$

$$A^*(q^{-1}) = 1 - 1.5q^{-1} + 0.7q^{-2}, \quad \text{No} = 60 \text{ for example 2}$$

$$A^*(q^{-1}) = 1 + 1.5q^{-1} + 0.7q^{-2}, \quad \text{No} = 60 \text{ for example 3}$$

Several such tests were conducted. The typical results are given in Figs. 1, 2, 3.

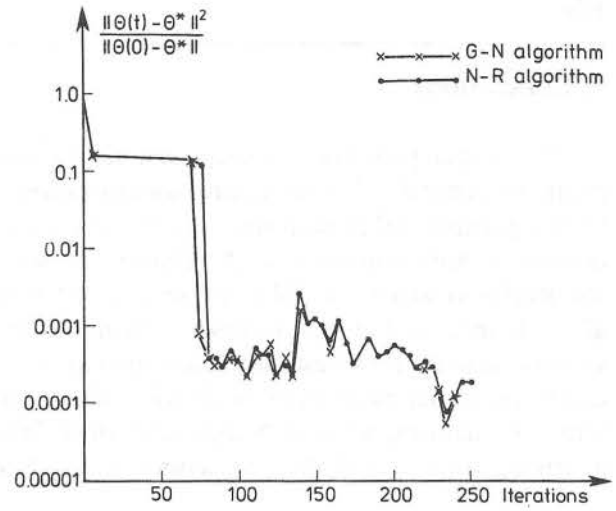

Fig. 1

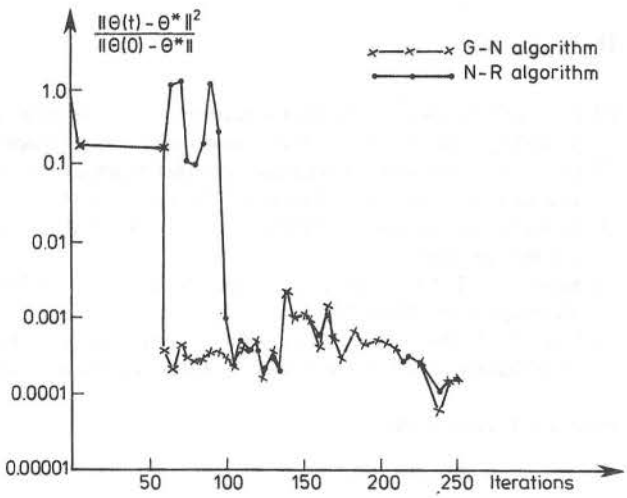

Fig. 2

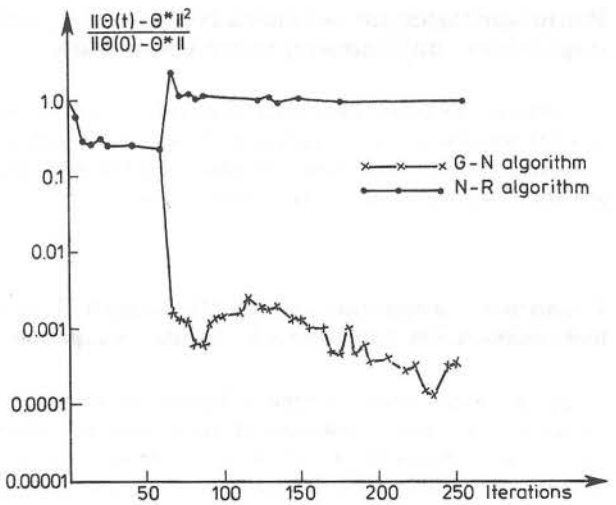

Fig. 3

5. Conclusions

This paper presents an exact version of Gauss-Newton algorithm for the recursive generalized least-squares identification. As compared to the existing G-N algorithms [3] at each step it evaluates gradient and Hessian of (5) exactly instead of approximating it. A relationship with the exact version of Newton-Raphson algorithm [5] is presented and their numerical compariosn is given. It appears that in most cases G-N algorithm is better than N-R being less vulnerable to the initial estimates. In some cases N-R algorithm doesn't converge due to the initial oscillations, while G-N algorithm doesn't converge due to the initial oscillations, while G-N algorithm does. When both algorithms converge a striking similarity of their behaviour is observed.

References

[1] CLARKE D. W. Generalized least squares estimation of parameters of a dynamic model. Ist IFAC Symposium on Identification in Automatic Control Systems. Prague.

[2] GERTLER J., BANYASZ C. A recursive (on line) maximum likelihood identification method. *IEEE Transaction on Automatic Control,* 1974, pp. 816-820.

[3] LJUNG L., SÖDERSTRÖM T. Theory and Practice of Recursive Identification. The MIT Press, Cambridge, 1983.

[4] SÖDERSTRÖM T. Convergence properties of the generalized least squares identification method. *Automatica,* **10** (1974), 617-626.

[5] UNTON F. Z., WEINFELD R. Newton-Raphson algorithm for the recursive generalized least-squares identification. *IEEE Transactions on Automatic Control,* **AC-31** (1986) 7.

Received, December 1988.

Porównanie algorytmów Gaussa-Newtona i Newtona-Raphsona dla rekurencyjnej uogólnionej najmniejszej sumy kwadratów

Przedstawiono dokładną wersję algorytmu Gaussa-Newtona dla rekurencyjnej identyfikacji w sensie uogólnionej najmniejszej sumy kwadratów. Przedstawiony algorytm jest w jakimś sensie podobny do algorytmu Newtona-Raphsona zaproponowanego niedawno przez autorów, jednak wykazuje znacznie lepsze własności numeryczne.

Сравнение алгоритмов Гаусса-Ньютона и Ньютона-Рафсона для рекуррентной обобщенной наименьшей суммы квадратов

Представлен точный алгоритм Гаусса—Ньютона для рекуррентоной идентификации в смысле обобщенной наименьшей суммы квадратов. Представленный алгоритм в каком-то смысле аналогичен алгоритму Ньютона-Рафсона недавно предложенному авторами, однако отличается значительно лучшими вычислительными свойствами.