MALAYA JOURNAL OF MATEMATIK
Malaya J. Mat. **11(01)**(2023), 11–24.
http://doi.org/10.26637/mjm1101/02

A new hybrid algorithm for maximum likelihood estimation in a model of accident frequencies

Issa Cherif GERALDO*[1], Tchilabalo Abozou KPANZOU[2] and Edoh KATCHEKPELE[2]

[1] *Laboratoire d'Analyse, de Modélisations Mathématiques et Applications (LAMMA), Département de Mathématiques, Faculté des Sciences, Université de Lomé, 1 B.P. 1515 Lomé 1, Togo.*
[2] *Laboratoire de Modélisation Mathématique et d'Analyse Statistique Décisionnelle (LaMMASD), Département de Mathématiques, Faculté des Sciences et Techniques, Université de Kara, Kara, Togo.*

*R*eceived 15 July 2022; *A*ccepted 17 October 2022

Abstract. In this paper, we are interested in the numerical computation of the constrained maximum likelihood estimator (MLE) of the parameter vector of a discrete statistical model used in statistics applied to road safety. The parameter vector is divided into two blocks: one block with the parameter of interest and the second block with secondary parameters. The MLE is the solution to a system of non-linear implicit equations difficult to solve in closed-form. To overcome this difficulty, we propose a hybrid algorithm (HA) mixing the use of a one-dimensional Newton-Raphson (NR) algorithm for the first equation of the system and a fixed-point strategy for the remaining equations. Our proposed algorithm involves no matrix inversion but it partially enjoys the quadratic convergence rate of the one-dimensional NR algorithm. We illustrate its performance on simulated data and we compare it to Newton-Raphson (NR) and quasi-Newton algorithms which are two of the most used optimization algorithms. The results suggest that our HA outperforms NR and quasi-Newton algorithms. It is accurate and converges quickly for all the starting values.

AMS Subject Classifications: 62F10, 62F30, 62H10, 62H12, 62P99.

Keywords: Numerical optimization, Constrained optimization, Maximum likelihood, Multinomial distribution, Cyclic algorithm, Hybrid algorithm, Road safety measure.

Contents

*Corresponding author. Email address: **cherifgera@gmail.com** (I. C. GERALDO), **t.kpanzou@univkara.net** (T. A. KPANZOU), **edohkatchekpele@gmail.com** (E. KATCHEKPELE)

1. Introduction

The statistical evaluation of road safety measures (for example, increase or decrease of speed limit, modification of ground markings, installation of roundabouts, etc…) is one of the very important issues in statistics applied to road safety. Assume that a road safety measure has been implemented at s ($s > 0$) geographical sites (hereinafter referred to as *treatment sites*) where crashes are categorized in r ($r > 0$) severity levels. Also assume that each treatment site is paired with a given site (hereinafter called *control site*) with the same characteristics as the treatment site (traffic flow, accident exposure, road conditions, etc…) except that the measure was not implemented. For all $k = 1, \dots, s$, let

$$\boldsymbol{X}_k = (X_{11k}, X_{12k}, \dots, X_{1rk}, X_{21k}, X_{22k}, \dots, X_{2rk})^{\mathsf{T}},$$

where X_{ijk} is the random number of crashes of severity level j ($j = 1, \dots, r$) occurred on treatment site k in period i ($i = 1$ corresponds to the period before implementation of the measure and $i = 2$ corresponds to the period after). Also let

$$\boldsymbol{z}_k = (z_{1k}, \dots, z_{rk})^{\mathsf{T}},$$

where z_{jk} is a (non-random) coefficient equal to the number of crashes of severity level j in the ”after” period divided by the number of crashes of the same severity level in the ”before” period on the control site paired with treatment site k.

Let $\alpha > 0$ be the overall mean effect of the measure (for example, $\alpha = 0.7 < 1$ means that the measure allowed $(1 - \alpha) \times 100\% = 30\%$ reduction in crashes) and $\beta_{jk} \in [0, 1]$ ($j = 1, \dots, r$; $k = 1, \dots, s$) be the probability that a crash occurring on treatment site k has severity level j. Thus, for all $k = 1, \dots, s$, the vector $\boldsymbol{\beta}_k = (\beta_{1k}, \dots, \beta_{rk})^{\mathsf{T}}$ belongs to $\mathbb{S}_{r-1}$, where

$$\mathbb{S}_{r-1} = \left\{ (p_1, \dots, p_r)^{\mathsf{T}} \in [0, 1]^r,\ \sum_{j=1}^{r} p_j = 1 \right\}. \tag{1.1}$$

Let $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathsf{T}})^{\mathsf{T}}$ be the vector composed of the $1 + sr$ parameters, where $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^{\mathsf{T}}, \dots, \boldsymbol{\beta}_s^{\mathsf{T}})^{\mathsf{T}} \in (\mathbb{S}_{r-1})^s$. The question here is how to define the probability distribution of random vectors $\boldsymbol{X}_1, \dots, \boldsymbol{X}_s$ in such a way that it can make it possible to estimate the vector $\boldsymbol{\theta}$.

There exist different models in the literature [28, 33, 34]. For all of them, the computation of the maximum likelihood estimate (MLE) $\hat{\boldsymbol{\theta}}$ of vector $\boldsymbol{\theta}$ requires the numerical maximization a log-likelihood function $L(\boldsymbol{\theta})$. A comprehensive review of modern numerical optimization algorithms is available in reference papers and books such as [1, 4, 11, 18, 19, 21, 26, 35]. The very first algorithm that comes to mind is the Newton-Raphson (NR) algorithm. It starts from an initial guess $\boldsymbol{\theta}^{(0)}$ and iterates according to the scheme

$$\boldsymbol{\theta}^{(m+1)} = \boldsymbol{\theta}^{(m)} - [\nabla^2 L(\boldsymbol{\theta}^{(m)})]^{-1} \nabla L(\boldsymbol{\theta}^{(m)}),$$

where $\nabla^2 L(\boldsymbol{\theta}^{(m)})$ and $\nabla L(\boldsymbol{\theta}^{(m)})$ are respectively the Hessian matrix and the gradient vector evaluated at $\boldsymbol{\theta}^{(m)}$. The convergence of NR algorithm is guaranteed only if the starting guess $\boldsymbol{\theta}^{(0)}$ is close to the true solution $\hat{\boldsymbol{\theta}}$ that is unknown in practice [11]. So, the success of NR algorithm strongly depends on the appropriate choice of $\boldsymbol{\theta}^{(0)}$ in a neighbourhood of the unknown $\hat{\boldsymbol{\theta}}$. In computational terms, NR algorithm can be very costly and heavy for high-dimensional problems or simply impossible to implement because of the numerical inversion of the Hessian matrix at each iteration. When NR algorithm is unsuccessful, alternatives such as quasi-Newton algorithms [35] (which compute approximations of $[\nabla^2 L(\boldsymbol{\theta}^{(m)})]^{-1}$ at each iteration) and Derivative-Free Optimization (DFO) algorithms [2, 23, 37] may be used but they are considered as effective in solving small to mid-size problems. In practice, it is known that no optimization algorithm is perfect and one must find the most suitable algorithm for each problem.

In this paper, we build a hybrid algorithm (HA) which combines a one-dimensional NR method for estimating α and a fixed-point strategy for estimating $\boldsymbol{\beta}$. Our proposed HA alternates between computing α from $\boldsymbol{\beta}$ using the one-dimensional NR algorithm and computing $\boldsymbol{\beta}$ from α using the fixed-point based strategy. Starting from an initial guess $\boldsymbol{\theta}^{(0)}$, our proposed hybrid algorithm for estimating $\boldsymbol{\theta}$ updates the successive iterates in the following manner: at step $m+1$, $\alpha^{(m+1)}$ is updated from $\boldsymbol{\beta}^{(m)}$ afterwards $\boldsymbol{\beta}^{(m+1)}$ is updated from $\alpha^{(m+1)}$ and $\boldsymbol{\beta}^{(m)}$. Our proposed algorithm is thus a cyclic algorithm (because it cycles through the components of $\boldsymbol{\theta}$ updating one from the other rather than updating the whole parameter vector at once) and it may be easily implemented.

The remainder of this paper is organized as follows. In Section 2, we describe the statistical model and give the likelihood function. In Section 3, we present our new hybrid algorithm for estimating the parameter vector $\boldsymbol{\theta}$. In Section 4, we present the results of the numerical study of the proposed algorithm and the results of its comparison with NR and quasi-Newton algorithms which are two of the most used optimization algorithms. The paper ends with Section 5 dedicated to the conclusion and some discussions.

2. Statistical model

As mentioned earlier, there exist different models in the literature [28, 33, 34]. The model described in [33] has been the subject of several research works in the particular case $s = 1$ [9, 10, 27, 29, 32] and in the general case $s \geqslant 1$ [6, 30, 31]. These results can be adapted to the model of [28] which is a reparametrization of the one in [33]. The models in [33] and [34] have been compared in the case $s = 1$ by [8] who demonstrated theoretical results on the measure of divergence between the two models and showed through real data that both models are very competitive.

In this paper, we are interested in the model of [34] which assumes that for all $k = 1, \dots, s$,

$$\boldsymbol{X}_k \rightsquigarrow \mathcal{M}(n_k; \boldsymbol{\pi}_k(\boldsymbol{\theta}|\boldsymbol{z}_k)), \tag{2.1}$$

where $\mathcal{M}$ denotes the multinomial distribution, n_k is the total number of crashes recorded on treatment site k,

$$\boldsymbol{\pi}_k(\boldsymbol{\theta}|\boldsymbol{z}_k) = (\pi_{11k}(\boldsymbol{\theta}|\boldsymbol{z}_k), \dots, \pi_{1rk}(\boldsymbol{\theta}|\boldsymbol{z}_k), \pi_{21k}(\boldsymbol{\theta}|\boldsymbol{z}_k), \dots, \pi_{2rk}(\boldsymbol{\theta}|\boldsymbol{z}_k))^{\mathsf{T}} \tag{2.2}$$

$$\pi_{1jk}(\boldsymbol{\theta}|\boldsymbol{z}_k) = \frac{\beta_{jk}}{1 + \alpha\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle}, \quad j = 1, \dots, r, \tag{2.3}$$

$$\pi_{2jk}(\boldsymbol{\theta}|\boldsymbol{z}_k) = \frac{\alpha\beta_{jk}\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle}{1 + \alpha\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle}, \quad j = 1, \dots, r, \tag{2.4}$$

and $\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle = \sum_{j'=1}^{r} z_{j'k}\beta_{j'k}$. The log-likelihood (see [34, p. 1275]) of observed data $\boldsymbol{x}_k = (x_{11k}, \dots, x_{1rk}, x_{21k}, \dots, x_{2rk})$ such as $\sum_{i=1}^{2}\sum_{j=1}^{r} x_{ijk} = n_k$, $k = 1, \dots, s$, is given to one additive constant by:

$$L(\boldsymbol{\theta}) = \sum_{k=1}^{s}\sum_{j=1}^{r}\left\{x_{\bullet jk}\log\beta_{jk} + x_{2jk}\log\alpha - x_{\bullet jk}\log\Big(1 + \alpha\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle\Big) + x_{2jk}\log\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k\rangle\right\}, \tag{2.5}$$

where $x_{\bullet jk} = x_{1jk} + x_{2jk}$. The maximum likelihood estimation problem is the following constrained maximization problem:

$$\begin{cases} \text{maximize } L(\boldsymbol{\theta}) & \text{(2.6a)} \\ \text{subject to} & \\ \alpha > 0 \quad \text{and} \quad \beta_{jk} > 0, \quad j = 1, \dots, r; \quad k = 1, \dots, s, & \text{(2.6b)} \\ \displaystyle\sum_{j=1}^{r} \beta_{jk} = 1, \quad k = 1, \dots, s. & \text{(2.6c)} \end{cases}$$

The maximum likelihood estimate (MLE) of $\boldsymbol{\theta}$ is denoted $\hat{\boldsymbol{\theta}} = (\hat{\alpha}, \hat{\boldsymbol{\beta}}^{\mathsf{T}})^{\mathsf{T}}$, where $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_1^{\mathsf{T}}, \dots, \hat{\boldsymbol{\beta}}_s^{\mathsf{T}})^{\mathsf{T}}$ and for all $k = 1, \dots, s$, $\hat{\boldsymbol{\beta}}_k = (\hat{\beta}_{1k}, \dots, \hat{\beta}_{rk})^{\mathsf{T}}$.

In the case $s = 1$, [7] has obtained the closed-form expression of the MLE. In the next section, we build a hybrid algorithm for computing the MLE $\hat{\boldsymbol{\theta}}$ in the case $s \geqslant 1$.

3. A hybrid algorithm for computing the MLE of the vector parameter

Let us begin with the following result.

Theorem 3.1. *For all $k = 1, \dots, s$, let*

$$x_{1\bullet k} = \sum_{j=1}^{r} x_{1jk}, \quad x_{2\bullet k} = \sum_{j=1}^{r} x_{2jk} \quad \textit{and} \quad x_{1\bullet\bullet} = \sum_{k=1}^{s} x_{1\bullet k}.$$

Then, the MLE $\hat{\boldsymbol{\theta}}$ is solution to the following system of non-linear equations:

$$\begin{cases} \displaystyle\sum_{k=1}^{s} \frac{n_k}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} - x_{1\bullet\bullet} = 0 & \text{(3.1a)} \\[2ex] \displaystyle x_{\bullet jk} - \frac{n_k \hat{\beta}_{jk}(\hat{\alpha} z_{jk}+1)}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} - \frac{x_{2\bullet k}\hat{\beta}_{jk}(\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle - z_{jk})}{\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} = 0, \\ \qquad\qquad j = 1, \dots, r, \quad k = 1, \dots, s. & \text{(3.1b)} \end{cases}$$

Proof. In [34, p. 1276], it is proved that $\hat{\boldsymbol{\theta}}$ is solution to the following system of non-linear equations:

$$\begin{cases} \displaystyle\sum_{k=1}^{s}\sum_{j=1}^{r} \frac{x_{2jk} - \hat{\alpha} x_{1jk}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} = 0 & \text{(3.2a)} \\[2ex] \displaystyle x_{\bullet jk} - \frac{n_k \hat{\beta}_{jk}(\hat{\alpha} z_{jk}+1)}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} - \frac{x_{2\bullet k}\hat{\beta}_{jk}(\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle - z_{jk})}{\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} = 0, \\ \qquad\qquad j = 1, \dots, r, \quad k = 1, \dots, s. & \text{(3.2b)} \end{cases}$$

The non-obvious part of the proof consists in proving Equation (3.1a) using Equation (3.2a). Passing the summation over index j to the numerator, we obtain Equation (3.2a) under the equivalent form

$$\sum_{k=1}^{s} \frac{x_{2\bullet k} - \hat{\alpha} x_{1\bullet k}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} = 0.$$

Since $x_{2\bullet k} = n_k - x_{1\bullet k}$, we also have

$$\sum_{k=1}^{s} \frac{n_k - x_{1\bullet k} - \hat{\alpha} x_{1\bullet k}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} = 0$$

which is equivalent to

$$\sum_{k=1}^{s} \frac{n_k}{1+\hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k\rangle} - \sum_{k=1}^{s} x_{1\bullet k} = 0.$$

The proof is thus completed. ■

In the case $s = 1$, [7] obtained the exact expression of $\hat{\boldsymbol{\theta}}$. But in the general case $s \geqslant 1$, non-linear Equations (3.1a) and (3.1b) cannot be solved in closed-form. There are implicit relations between parameters $\hat{\alpha}$ and $\hat{\beta}_{jk}$'s which can make it possible to compute $\hat{\alpha}$ from the $\hat{\beta}_{jk}$'s and conversely, compute the $\hat{\beta}_{jk}$'s from $\hat{\alpha}$. So, we propose the following strategy for a fast and efficient computation of $\hat{\boldsymbol{\theta}}$: for a given estimate $\boldsymbol{\theta}^{(m)} = (\alpha^{(m)}, (\boldsymbol{\beta}^{(m)})^\mathsf{T})^\mathsf{T}$ obtained after m iterations,

(a) use Equation (3.1a) to compute $\alpha^{(m+1)}$ from $\boldsymbol{\beta}^{(m)}$ and

(b) use Equation (3.1b) to compute the $\beta_{jk}^{(m+1)}$'s (the components of $\boldsymbol{\beta}^{(m+1)}$) from $\alpha^{(m+1)}$ and $\boldsymbol{\beta}^{(m)}$.

We propose a one-dimensional Newton-Raphson (NR) algorithm to perform Step (a) and a fixed point strategy for Step (b).

3.1. Computation of $\alpha^{(m+1)}$ via a NR algorithm

The following lemma states that for all estimate $\boldsymbol{\beta}^{(m)}$ of $\boldsymbol{\beta}$, Equation (3.1a) (considered as an equation of unknown α) has a unique solution $\alpha^{(m+1)}$.

Lemma 3.2. *Let*

$$x_{2\bullet\bullet} = \sum_{k=1}^{s} x_{2\bullet k}$$

and let $\boldsymbol{\beta}^{(m)}$ be the estimate of $\boldsymbol{\beta}$ after m iterations. Let us rewrite Equation (3.1a) *under the form $\Psi_{\boldsymbol{\beta}^{(m)}}(u) = 0$ where $\Psi_{\boldsymbol{\beta}^{(m)}}$ is the function from $[0, +\infty[$ to $]-x_{1\bullet\bullet}, x_{2\bullet\bullet}]$ defined by:*

$$\Psi_{\boldsymbol{\beta}^{(m)}}(u) = -x_{1\bullet\bullet} + \sum_{k=1}^{s} \frac{n_k}{1 + u\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)}\rangle}. \tag{3.3}$$

Then, Equation $\Psi_{\boldsymbol{\beta}^{(m)}}(u) = 0$ has a unique solution $\alpha^{(m+1)}$.

Proof. The function $\Psi_{\boldsymbol{\beta}^{(m)}}$ is differentiable and its derivative $\Psi'_{\boldsymbol{\beta}^{(m)}}$, defined on $[0, +\infty[$ by

$$\Psi'_{\boldsymbol{\beta}^{(m)}}(u) = -\sum_{k=1}^{s} \frac{n_k \langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)}\rangle}{(1 + u\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)}\rangle)^2},$$

is strictly negative. Thus, $\Psi_{\boldsymbol{\beta}^{(m)}}$ is a continuous and strictly decreasing function. Therefore,

$$\Psi_{\boldsymbol{\beta}^{(m)}}\Big([0, +\infty[\Big) = \Big]\lim_{u \to +\infty} \Psi_{\boldsymbol{\beta}^{(m)}}(u), \Psi_{\boldsymbol{\beta}^{(m)}}(0)\Big],$$

where

$$\Psi_{\boldsymbol{\beta}^{(m)}}(0) = -x_{1\bullet\bullet} + \sum_{k=1}^{s} n_k = \sum_{k=1}^{s}(-x_{1\bullet k} + n_k) = \sum_{k=1}^{s} x_{2\bullet k} = x_{2\bullet\bullet} > 0$$

and

$$\lim_{u \to +\infty} \Psi_{\boldsymbol{\beta}^{(m)}}(u) = -x_{1\bullet\bullet} < 0.$$

We can conclude that $\Psi_{\boldsymbol{\beta}^{(m)}}$ is bijective from $[0, +\infty[$ to $]-x_{1\bullet\bullet}, x_{2\bullet\bullet}]$ and, since $-x_{1\bullet\bullet} < 0 < x_{2\bullet\bullet}$, the equation $\Psi_{\boldsymbol{\beta}^{(m)}}(u) = 0$ has a unique solution. ■

For a given $\boldsymbol{\beta}^{(m)}$, computing $\alpha^{(m+1)}$ (the unique solution to Equation $\Psi_{\boldsymbol{\beta}^{(m)}}(u) = 0$) in closed-form is complicated when $s > 1$. Thus, a numerical root finding algorithm is required. There certainly exist different algorithms for this purpose (see for example [12, Chapter 3]). Here, we propose the following one-dimensional Newton-Raphson (NR) algorithm:

$$u^{(\ell+1)} = u^{(\ell)} - \frac{\Psi_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}{\Psi'_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}, \qquad \ell = 1, 2, \ldots \tag{3.4}$$

where the starting guess $u^{(0)}$ is positive. The main advantage of NR algorithm is that it converges quadratically to the solution (the number of correct significant digits doubles with each iteration) if the starting guess $u^{(0)}$ is close to the unknown solution. And there lies the problem: the choice of $u^{(0)}$. Fortunately, we prove hereafter that, if we set $u^{(0)} = 0$, then, the convergence of NR iterations (3.4) is guaranteed.

Lemma 3.3. *Let $\boldsymbol{\beta}^{(m)}$ be the estimate of $\boldsymbol{\beta}$ after m iterations. If $u^{(0)} = 0$, then, the NR iterations*

$$u^{(\ell+1)} = u^{(\ell)} - \frac{\Psi_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}{\Psi'_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}, \qquad \ell = 1, 2, \dots \tag{3.5}$$

converge to the next iterate $\alpha^{(m+1)}$ such that $\Psi_{\boldsymbol{\beta}^{(m)}}(\alpha^{(m+1)}) = 0$.

Proof. The proof uses a result from [3, Section 4.5] which states that if $f : [a, b] \to \mathbb{R}$ is a function twice differentiable such that $f(a)f(b) < 0$, and $f'(u)$ and $f''(u)$ are non-zero and preserve signs over $[a, b]$, then, proceeding from an initial approximation $u^{(0)} \in [a, b]$ such that $f(u^{(0)})f''(u^{(0)}) > 0$, the NR sequence $(u^{(\ell)})$ defined by

$$u^{(\ell+1)} = u^{(\ell)} - \frac{f(u^{(\ell)})}{f'(u^{(\ell)})}, \qquad \ell = 1, 2, \dots,$$

converges to the unique root of f in $[a, b]$.

Let $u_M = \alpha^{(m+1)} + 1$. As $\Psi_{\boldsymbol{\beta}^{(m)}}$ is a strictly decreasing function and $\Psi_{\boldsymbol{\beta}^{(m)}}(\alpha^{(m+1)}) = 0$, we have $\Psi_{\boldsymbol{\beta}^{(m)}}(0) \times \Psi_{\boldsymbol{\beta}^{(m)}}(u_M) < 0$. Moreover, the function $\Psi_{\boldsymbol{\beta}^{(m)}}$ is twice differentiable and, for all $u \in [0, +\infty[$,

$$\Psi'_{\boldsymbol{\beta}^{(m)}}(u) = -\sum_{k=1}^{s} \frac{n_k \langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle}{(1 + u\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle)^2} < 0 \tag{3.6}$$

and

$$\Psi''_{\boldsymbol{\beta}^{(m)}}(u) = \sum_{k=1}^{s} \frac{2n_k (\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle)^2}{(1 + u\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle)^3} > 0. \tag{3.7}$$

We also have $u^{(0)} = 0 \in [0, u_M]$ and $\Psi_{\boldsymbol{\beta}^{(m)}}(0) \times \Psi''_{\boldsymbol{\beta}^{(m)}}(0) > 0$. We can conclude that the sequence $(u^{(\ell)})$ defined by (3.4) converges to the desired value $\alpha^{(m+1)}$. ■

Remark 3.4. *Actually, since $\Psi''_{\boldsymbol{\beta}^{(m)}}(u) > 0$ for all $u > 0$, any starting guess $u^{(0)}$ such that $\Psi_{\boldsymbol{\beta}^{(m)}}(\alpha^{(0)}) > 0$ could be suitable to guarantee convergence of the iterative scheme* (3.4). *But the search of such an $u^{(0)}$ seems doomed because when $u \neq 0$, the denominators contained in the expression of $\Psi_{\boldsymbol{\beta}^{(m)}}(u)$ do not vanish any more.*

3.2. A fixed-point scheme for computing $\beta^{(m+1)}$

From Equation (3.1b), we have for all $k = 1, \dots, s$ and $j = 1, \dots, r$,

$$\hat{\beta}_{jk} \left(\frac{n_k(\hat{\alpha} z_{jk} + 1)}{1 + \hat{\alpha}\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k \rangle} + x_{2\bullet k} - \frac{x_{2\bullet k} z_{jk}}{\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k \rangle} \right) = x_{\bullet jk}. \tag{3.8}$$

In this case too, it is complicated to get $\hat{\beta}_{jk}$ in closed-form because it also appears in the denominator as a component of the weighted sum $\langle \boldsymbol{z}_k, \hat{\boldsymbol{\beta}}_k \rangle$. For all $k = 1, \dots, s$ and $j = 1, \dots, r$, we propose to compute $\beta_{jk}^{(m+1)}$ using the following fixed-point scheme:

$$\beta_{jk}^{(m+1)} = x_{\bullet jk} \Bigg/ \left(\frac{n_k(\alpha^{(m+1)} z_{jk} + 1)}{1 + \alpha^{(m+1)}\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle} + x_{2\bullet k} - \frac{x_{2\bullet k} z_{jk}}{\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle} \right). \tag{3.9}$$

3.3. The hybrid algorithm (HA)

Our proposed hybrid algorithm is Algorithm 3.5. It starts from $\boldsymbol{\theta}^{(0)} = (\alpha^{(0)}, (\boldsymbol{\beta}^{(0)})^\intercal)^\intercal$, where $\alpha^{(0)} > 0$ is randomly set and $\boldsymbol{\beta}^{(0)} = ((\boldsymbol{\beta}_1^{(0)})^\intercal, \ldots, (\boldsymbol{\beta}_s^{(0)})^\intercal)^\intercal$ is also randomly set such that for all $k = 1, \ldots, s$, $\boldsymbol{\beta}_k^{(0)} = (\beta_{1k}^{(0)}, \ldots, \beta_{rk}^{(0)})^\intercal \in \mathbb{S}_{r-1}$. At the $(m+1)-$iteration, the update $\alpha^{(m+1)}$ is computed from $\boldsymbol{\beta}^{(m)}$ using the one-dimensional NR iterations (3.4), afterwards $\boldsymbol{\beta}^{(m+1)}$ is updated from $\alpha^{(m+1)}$ and $\boldsymbol{\beta}^{(m)}$ using Equation (3.9) and so on. This process is repeated until a convergence criterion is satisfied.

Algorithm 3.5.
Input: $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_s, \epsilon_1 > 0$ *and* $\epsilon_2 > 0$.
Output: MLE $\hat{\boldsymbol{\theta}}$.

```
1. (a) Set $m = 0$
   (b) Initialize $\boldsymbol{\theta}^{(0)} = (\alpha^{(0)}, (\boldsymbol{\beta}_1^{(0)})^\intercal, \ldots, (\boldsymbol{\beta}_s^{(0)})^\intercal)^\intercal$ randomly such that $\alpha^{(0)} > 0$, and for all $k = 1, \ldots, s$,
       $\boldsymbol{\beta}_k^{(0)} = (\beta_{1k}^{(0)}, \ldots, \beta_{rk}^{(0)})^\intercal \in \mathbb{S}_{r-1}$.
2. (a) Update $\alpha^{(m+1)}$ as follows:
         i. Set $\ell = 0$
        ii. Set $u^{(\ell)} = 0$
       iii. Set
              $u^{(\ell+1)} = u^{(\ell)} - \dfrac{\Psi_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}{\Psi'_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)})}$
        iv. Set $\ell \leftarrow \ell + 1$
         v. If $|\Psi_{\boldsymbol{\beta}^{(m)}}(u^{(\ell)}) - \Psi_{\boldsymbol{\beta}^{(m)}}(u^{(\ell-1)})| \geqslant \epsilon_1$, go back to Step (ii).
        vi. Set $\alpha^{(m+1)} = u^{(\ell)}$.
   (b) For all $k = 1, \ldots, s$, compute $\boldsymbol{\beta}_k^{(m+1)} = (\beta_{1k}^{(m+1)}, \ldots, \beta_{rk}^{(m+1)})^\intercal$ where, for all $j = 1, \ldots, r$,
         $\beta_{jk}^{(m+1)} = x_{\bullet jk} \Bigg/ \left( \dfrac{n_k(\alpha^{(m+1)} z_{jk} + 1)}{1 + \alpha^{(m+1)} \langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle} + x_{2\bullet k} - \dfrac{x_{2\bullet k} z_{jk}}{\langle \boldsymbol{z}_k, \boldsymbol{\beta}_k^{(m)} \rangle} \right)$
   (c) Update $\boldsymbol{\beta}^{(m+1)} = ((\boldsymbol{\beta}_1^{(m+1)})^\intercal, \ldots, (\boldsymbol{\beta}_s^{(m+1)})^\intercal)^\intercal$.
   (d) Set $\boldsymbol{\theta}^{(m+1)} = (\alpha^{(m+1)}, (\boldsymbol{\beta}^{(m+1)})^\intercal)^\intercal$
   (e) Set $m \leftarrow m + 1$.
3. If $|L(\boldsymbol{\theta}^{(m)}) - L(\boldsymbol{\theta}^{(m-1)})| \geqslant \epsilon_2$, go back to Step 2.
4. Set $\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}^{(m)}$.
```

4. Simulation study

In this section, we compare our proposed HA to some of the best optimization algorithms available in R software [36] in terms of accuracy, robustness (the ability of each algorithm to perform well regardless of the starting guess) and computation time. The selected algorithms are the Newton-Raphson (NR) algorithm and the quasi-Newton BFGS algorithm (named after its authors Broyden, Fletcher, Goldfarb and Shanno) [35, Section 6.1]. The NR and BFGS algorithms are implemented using respectively the R packages **nleqslv** [13] and **alabama** [38]. Accuracy is measured with the Mean Squared Error (MSE)

$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}|\boldsymbol{\theta}^0) = \frac{1}{1+sr} \|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0\|^2,$$

where $\boldsymbol{\theta}^0 = (\alpha^0, (\boldsymbol{\beta}_1^0)^\top, \ldots, (\boldsymbol{\beta}_s^0)^\top)^\top$ is the true value and $\|\cdot\|$ represents the usual Euclidean norm. Robustness is measured by comparing the convergence proportions (the ratio of the number of times each algorithm converged to the total number of replications) and the numbers of iterations for different starting guesses.

4.1. Data generation

The data are generated using Formulas (2.1), (2.2), (2.3) and (2.4) where the true value $\boldsymbol{\theta}^0 = (\alpha^0, (\boldsymbol{\beta}_1^0)^\top, \ldots, (\boldsymbol{\beta}_s^0)^\top)^\top$ of $\boldsymbol{\theta}$ is presented under five scenarios described below:

- **Scenario 1**: $s = 2$, $r = 2$,
$$\alpha^0 = 0.8, \qquad \boldsymbol{\beta}_1^0 = (0.85, 0.15)^\top, \qquad \boldsymbol{\beta}_2^0 = (0.40, 0.60)^\top.$$
- **Scenario 2**: $s = 5$, $r = 3$,
$$\begin{aligned} \alpha^0 = 1,\ \boldsymbol{\beta}_1^0 &= (0.80, 0.15, 0.05)^\top, \\ \boldsymbol{\beta}_2^0 &= (0.10, 0.30, 0.60)^\top, \\ \boldsymbol{\beta}_3^0 &= (0.35, 0.30, 0.35)^\top, \\ \boldsymbol{\beta}_4^0 &= (0.70, 0.20, 0.10)^\top, \\ \boldsymbol{\beta}_5^0 &= (0.30, 0.40, 0.30)^\top. \end{aligned}$$
- **Scenario 3**: $s = 10$, $r = 5$,
$$\begin{aligned} \alpha^0 = 1,\ \boldsymbol{\beta}_k^0 &= (0.40, 0.10, 0.05, 0.25, 0.20)^\top, \quad k \in \{1, 3, 5, 9\}, \\ \boldsymbol{\beta}_k^0 &= (0.30, 0.15, 0.10, 0.25, 0.20)^\top, \quad k \in \{2, 4, 7\}, \\ \boldsymbol{\beta}_k^0 &= (\underbrace{0.20, \ldots, 0.20}_{5})^\top, \quad i \in \{6, 8, 10\}. \end{aligned}$$
- **Scenario 4**: $s = 10$, $r = 10$,
$$\begin{aligned} \alpha^0 = 1.2,\ \boldsymbol{\beta}_k^0 &= (0.40, 0.10, 0.05, \underbrace{0.10}_{2}, \underbrace{0.05}_{5})^\top, \quad k \in \{1, 5, 7, 10\}; \\ \boldsymbol{\beta}_k^0 &= (\underbrace{0.10}_{3}, \underbrace{0.05}_{2}, 0.10, 0.25, \underbrace{0.05}_{2}, 0.15)^\top, \quad k \in \{2, 3, 6\}; \\ \boldsymbol{\beta}_k^0 &= (\underbrace{0.10, \ldots, 0.10}_{10})^\top, \quad k \in \{4, 8, 9\}. \end{aligned}$$
- **Scenario 5**: $s = 20$, $r = 10$,
$$\begin{aligned} \alpha^0 = 1.2,\ \boldsymbol{\beta}_k^0 &= (0.40, 0.10, 0.05, \underbrace{0.10}_{2}, \underbrace{0.05}_{5})^\top, \quad k \in \{1, 5, 7, 10, 11, 15, 17, 20\}, \\ \boldsymbol{\beta}_k^0 &= (\underbrace{0.10}_{3}, \underbrace{0.05}_{2}, 0.10, 0.25, \underbrace{0.05}_{2}, 0.15)^\top, \quad k \in \{2, 3, 6, 12, 13, 16\}, \\ \boldsymbol{\beta}_k^0 &= (\underbrace{0.10, \ldots, 0.10}_{10})^\top, \quad k \in \{4, 8, 9, 14, 18, 19\}. \end{aligned}$$

These scenarios allow to have low and large values of the number of parameters $(1 + sr)$. The values of the number of parameters for these scenarios are given in Table 1.

For each scenario, we set $n_1 = \cdots = n_s = n$, where n had two different values: a low value ($n = 50$) and a great value ($n = 5000$). In order to explore a plethora of starting guesses in the parameter space, we have considered a random initialization scheme for setting the starting guess $\boldsymbol{\theta}^{(0)} = (\alpha^{(0)}, (\boldsymbol{\beta}^{(0)})^\top)^\top$. The parameter $\alpha^{(0)}$ is randomly generated from an uniform distribution and each $\boldsymbol{\beta}_k^{(0)} = (\beta_{1k}^{(0)}, \ldots, \beta_{rk}^{(0)})^\top$ is randomly generated as $\boldsymbol{\beta}_k^{(0)} = \boldsymbol{U}_k / \sum_{j=1}^{r} u_{jk}$ where $\boldsymbol{U}_k = (u_{1k}, \ldots, u_{rk})^\top$ is a $r-$dimensional vector whose components are randomly generated from a uniform distribution $\mathcal{U}[0.05; 0.95]$.

Table 1: Number of parameters for the different scenarios

Scenario	1	2	3	4	5
s	2	5	10	10	20
r	2	3	5	10	10
Number of parameters $(1 + sr)$	5	16	51	101	201

Table 2: Results for Scenario 1 ($s = 2$ and $r = 2$). Values in brackets are standard deviations.

		TRUTH	HA	NR	BFGS
$n = 50$	α	0.800	0.817 (0.172)	0.817 (0.172)	0.811 (0.170)
	β_{11}	0.850	0.848 (0.049)	0.848 (0.049)	0.848 (0.048)
	β_{21}	0.150	0.152 (0.049)	0.152 (0.049)	0.152 (0.048)
	β_{12}	0.400	0.400 (0.071)	0.400 (0.071)	0.399 (0.071)
	β_{22}	0.600	0.600 (0.071)	0.600 (0.071)	0.601 (0.071)
	Convergence proportion (%)	-	100	99.6	77.1
	Iterations	-	7.1 (1.6)	5.7 (1.5)	13 (0)
	CPU time (secs)	-	0.004	0.002	0.100
	Time ratio	-	1	0.6	27.7
	Log-likelihood	-	-121.37	-121.37	-121.37
	MSE	-	9e-03	8.9e-03	8.7e-03
$n = 5000$	α	0.800	0.800 (0.016)	0.800 (0.016)	0.800 (0.016)
	β_{11}	0.850	0.850 (0.005)	0.850 (0.005)	0.850 (0.005)
	β_{21}	0.150	0.150 (0.005)	0.150 (0.005)	0.150 (0.005)
	β_{12}	0.400	0.400 (0.007)	0.400 (0.007)	0.400 (0.007)
	β_{22}	0.600	0.600 (0.007)	0.600 (0.007)	0.600 (0.007)
	Convergence proportion (%)	-	100	99.8	77.7
	Iterations	-	6.5 (1.1)	6 (1.5)	16 (0)
	CPU time (secs)	-	0.004	0.003	0.121
	Time ratio	-	1	0.8	33.3
	Log-likelihood	-	-12245.17	-12245.17	-12245.17
	MSE	-	8.1e-05	8.1e-05	8.2e-05

Table 3: Results for Scenario 2 ($s = 5$ and $r = 3$). Values in brackets are standard deviations.

		HA	NR	BFGS
$n = 50$	Convergence proportion (%)	100	99.9	51.6
	Iterations	9.5 (1.8)	8 (2.8)	14 (0.2)
	CPU time (secs)	0.007	0.017	0.248
	Time ratio	1	2.5	37.1
	Log-likelihood	-390.89	-390.89	-390.89
	MSE	4.3e-03	4.3e-03	4.2e-03
$n = 5000$	Convergence proportion (%)	100	99.8	56.1
	Iterations	8.1 (1.1)	8.3 (2.9)	18 (4)
	CPU time (secs)	0.005	0.013	0.405
	Time ratio	1	2.9	89.4
	Log-likelihood	-39171.57	-39171.57	-40218.56
	MSE	4.4e-05	4.4e-05	1.8e-02

Table 4: Results for Scenario 3 ($s = 10$ and $r = 5$). Values in brackets are standard deviations.

		HA	NR	BFGS
$n = 50$	Convergence proportion (%)	100	99.8	21.4
	Iterations	10.8 (1.8)	10.6 (3.2)	13.9 (0.3)
	CPU time (secs)	0.006	0.070	0.321
	Time ratio	1	11.9	54.1
	Log-likelihood	-1083.73	-1083.73	-1083.73
	MSE	3e-03	3e-03	3e-03
$n = 5000$	Convergence proportion (%)	100	99.9	24
	Iterations	8.6 (0.9)	10.5 (2.8)	16.7 (4.4)
	CPU time (secs)	0.005	0.062	0.452
	Time ratio	1	13.4	98.3
	Log-likelihood	-109133.14	-109133.14	-111075.03
	MSE	3.2e-05	3.2e-05	5.4e-03

Table 5: Results for Scenario 4 ($s = 10$ and $r = 10$). Values in brackets are standard deviations.

		HA	NR	BFGS
$n = 50$	Convergence proportion (%)	100	97.1	20.1
	Iterations	10.1 (1.5)	14.2 (4.9)	13.9 (0.3)
	CPU time (secs)	0.008	0.162	0.635
	Time ratio	1	19.5	76.4
	Log-likelihood	-1406.16	-1406.16	-1406.16
	MSE	1.9e-03	1.9e-03	1.9e-03
$n = 5000$	Convergence proportion (%)	100	93.4	19.6
	Iterations	8.4 (0.7)	15 (5.3)	17.8 (2)
	CPU time (secs)	0.005	0.181	1.012
	Time ratio	1	34.4	193
	Log-likelihood	-138384.13	-138384.13	-138673.98
	MSE	1.8e-05	1.8e-05	9.6e-04

Table 6: Results for Scenario 5 ($s = 20$ and $r = 10$). Values in brackets are standard deviations.

		HA	NR	BFGS
$n = 50$	Convergence proportion (%)	100	95.2	3.5
	Iterations	11 (1.6)	16.6 (5.1)	14 (0)
	CPU time (secs)	0.008	0.644	2.079
	Time ratio	1	80.1	258.6
	Log-likelihood	-2816.68	-2816.68	-2816.68
	MSE	1.7e-03	1.7e-03	1.6e-03
$n = 5000$	Convergence proportion (%)	100	88	5
	Iterations	8.8 (0.7)	17.9 (5.7)	11 (8)
	CPU time (secs)	0.006	0.732	2.254
	Time ratio	1	123.9	381.6
	Log-likelihood	-276918.57	-276918.57	-282868.47
	MSE	1.7e-05	1.7e-05	8.6e-03

4.2. Results

Tables 2 to 6 present the average estimates for 1000 replicates for each scenario and each value of n. In these tables, CPU (central process unit) times are given in seconds and time ratios are calculated as the ratio between the mean CPU time of a given algorithm and the mean CPU time of the HA. Thus, the time ratio of the HA is always 1. To save space, the estimate $\hat{\boldsymbol{\theta}}$ has been included only for Scenario 1 (Table 2).

Ideally, any iterative algorithm should converge to a solution close to the true value in a relatively reasonable computation time. By analysing Tables 2 to 6, we see that the HA has a convergence proportion of 100%. The NR algorithm has a convergence proportion between 88% and 99.6% so for this model, the convergence proportion of the NR algorithm does not reach 100% even for a small number of parameters. The convergence proportion of the BFGS algorithm decreases drastically (from 77.7% down to 3.5%) when the number of parameters increases. Concerning precision, the general trend observed is the decrease of the MSE when the sample size n increases. But the MSE of BFGS algorithm remain relatively high when n goes from 50 to 5000.

For computation times, it is noticed that the computation time of HA is stable (between 0.004 and 0.008 seconds) despite the increase in the number of parameters to be estimated. This is not the case for NR and BFGS algorithms. The analysis of the CPU time ratios shows that the computation times required by NR and BFGS algorithms increase with the number of parameters. Our proposed hybrid algorithm is up to 123 times faster than NR and up to 381 times faster than BFGS.

As far as the number of iterations is concerned, the HA has a low average number of iterations. The BFGS and NR algorithms look much more sensitive to the starting guesses. The average number of iterations of NR algorithm and its standard deviation increase with the number of parameters while the number of iterations of BFGS algorithm remains very high.

Overall, our hybrid algorithm outperforms NR and BFGS algorithms in terms of convergence proportion, convergence speed, computation time and accuracy.

5. Discussion and conclusion

For numerical optimization in general, and computation of maximum likelihood estimates (MLE) specifically, Newton-Raphson (NR) algorithm is the very first to be considered because of its fast convergence when the starting guess is close to the unknown solution. But because it requires matrix inversion at each iteration, it becomes tricky in computation terms for high-dimensional problems. Alternatives such as the quasi-Newton BFGS algorithm are usually considered as good remedies when NR is unsuccessful.

In this paper, we presented a new hybrid algorithm (HA) for estimating the parameter vector $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathsf{T}})^{\mathsf{T}}$ of a statistical model used in road safety. The parameter $\alpha > 0$ which is also the parameter of interest represents the mean effect (in the multiplicative sense) of a road safety measure and the vector $\boldsymbol{\beta}$ is a vector of sr probabilities where s is the number of sites where the measure has been applied and r is the number of accidents severity levels. Our HA mixes a one-dimensional NR approach for computing the parameter α and a fixed-point strategy for computing $\boldsymbol{\beta}$. It cycles through the parameter vector updating α from $\boldsymbol{\beta}$ and $\boldsymbol{\beta}$ from both α and $\boldsymbol{\beta}$ until a convergence criterion is satisfied. It thus partially enjoys (for the estimation of α) the fast convergence property of NR while avoiding its defects (it requires no matrix inversion and the starting value is automated in order to guarantee convergence). Since it cycles through the parameters, our HA is also a cyclic algorithm and, therefore, as claimed by [16], it enjoys overall fast convergence because the log-likelihood function is always driven in the right direction. The numerical studies performed in this paper suggest that our HA outperforms NR and BFGS algorithms in terms of convergence proportion, convergence speed, computation time and accuracy. They also suggest that the HA is globally convergent (it converges to the MLE for all starting guesses). A future work may be devoted to the study of the theoretical properties of our proposed hybrid algorithm such as the global convergence (the convergence to the MLE whatever the starting point $\boldsymbol{\theta}^{(0)}$).

In this paper, we have not mentioned MM (Minorization-Maximization) algorithms [16, 20, 22] which have become very popular over the years. Strictly speaking, the MM method is not a simple box in which we insert the function $L(\theta)$ while waiting for the output but it is rather a principle of construction of an optimization algorithm for a specific function. For example, the MM principle for building an algorithm for maximization of a log-likelihood $L(\boldsymbol{\theta})$ consists in constructing a minorizing function $g(\theta)$ for $L(\boldsymbol{\theta})$ such that the maximization of $g(\theta)$ is equivalent to that of $L(\theta)$ and afterwards construct an algorithm for maximizing $g(\theta)$ (see, for example, [16, 20, 22] for the definition of a minorizing function). The design of an MM algorithm is often not easy and for each $L(\boldsymbol{\theta})$ it is necessary to build an adequate MM algorithm. This explains why the design of an MM algorithm is often done by model [5, 14, 15, 17, 24, 39]. In this line of thought, in [25], the authors constructed an MM algorithm for estimating the parameters of the road safety model of [33]. However this algorithm cannot be used for the model [34] considered in the present paper. In a future work, it would be interesting to build an MM algorithm for the model of [34] and compare it with our hybrid algorithm.

References

[1] J. F. Bonnans, J. C. Gilbert, C. Lemaréchal and C. Sagastizábal, *Numerical Optimization – Theoretical and Practical Aspects*, Springer Verlag, Berlin, (2006).

[2] A. R. Conn, K. Scheinberg and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Book Series on Optimization. SIAM, Philadelphia, (2009).

[3] B. P. Demidovich and I. A. Maron, *Computational Mathematics*. MIR Publishers, Moscow, (1981).

[4] J. E. Dennis Jr and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, SIAM, Philadelphia, (1996).

[5] J. Ding, G. L. Tian and K. C. Yuen, A new MM algorithm for constrained estimation in the proportional hazards model, *Comput. Statist. Data Anal.*, **84**(2015), 135–151.

[6] I. C. Geraldo, An Automated Profile-Likelihood-Based Algorithm for Fast Computation of the Maximum Likelihood Estimate in a Statistical Model for Crash Data, *J. Appl. Math.*, **2022**(2022), Article ID 6974166, 11 p.

[7] I. C. Geraldo, On the maximum likelihood estimator for a discrete multivariate crash frequencies model, *Afr. Stat.*, **15(2)**(2020), 2335–2348.

[8] I. C. Geraldo and T. A. Kpanzou, Theoretical and empirical comparison of two discrete statistical models of crash frequencies, *Malaya J. Mat.*, **10(04)**(2022), 322–335.

[9] I. C. Geraldo, A. N'Guessan and K. E. Gneyou, A note on the strong consistency of a constrained maximum likelihood estimator used in crash data modeling, *C. R. Math. Acad. Sci. Paris*, **353(12)**(2015), 1147–1152.

[10] I. C. Geraldo, A. N'Guessan and K. E. Gneyou, Global convergence and ascent property of a cyclic algorithm used for statistical analysis of crash data, *Afr. Stat.*, **13(2)**(2018), 1631–1643.

[11] I. Griva, S. G. Nash, and A. Sofer, *Linear and Nonlinear Optimization*, Society for Industrial and Applied Mathematics, Philadelphia, 2nd edition, (2009).

[12] R. K. Gupta, *Numerical methods: Fundamentals and applications*, Cambridge University Press, New Delhi, (2019).

[13] B. Hasselman, *nleqslv: Solve Systems of Nonlinear Equations*, URL `https://CRAN.R-project.org/package=nleqslv`, R package version 3.3.2., (2018).

[14] D. R. Hunter, MM algorithms for generalized Bradley-Terry models, *Ann. Statist.*, **32(1)**(2004), 384–406.

[15] D. R. Hunter and R. Li, Variable Selection using MM Algorithms, *Ann. Statist.*, **33(4)**(2005), 1617–1642.

[16] D. R. Hunter and K. Lange, A tutorial on MM algorithms, *Amer. Statist.*, **58(1)**(2004), 30–37.

[17] J. James, MM algorithm for general mixed multinomial logit models, *J. Appl. Econometrics*, **32(4)**(2017), 841–857.

[18] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for optimization*, The MIT Press, Cambridge (Massachusetts), (2019).

[19] K. Lange, *Optimization*, Springer, New York, 2nd edition, (2013).

[20] K. Lange, *MM optimization algorithms*, SIAM, Philadelphia, (2016).

[21] K. Lange, E. C. Chi, and H. Zhou, A brief survey of modern optimization for statisticians, *Int. Stat. Rev.*, **82(1)**(2014), 46–70.

[22] K. Lange, D. R. Hunter and I. Yang, Optimization transfer using surrogate objective functions, *J. Comput. Graph. Statist.*, **9(1)**(2000), 1–20.

[23] J. Larson, M. Menickelly, and S. M. Wild, Derivative-free optimization methods, *Acta Numer.*, **28**(2019), 287–404.

[24] S. Lee and J. Z. Huang, A coordinate descent MM algorithm for fast computation of sparse logistic PCA, *Comput. Statist. Data Anal.*, **62**(2013), 26–38.

[25] A. Mkhadri, A. N'Guessan and B. Hafidi, An MM algorithm for constrained estimation in a road safety measure modeling, *Comm. Statist. Simulation Comput.*, **39(5)**(2010), 1057–1071.

[26] J. F. Monahan, *Numerical Methods of Statistics*, Cambridge University Press, 2nd edition, (2011).

[27] A. N'Guessan, Analytical existence of solutions to a system of nonlinear equations with application, *J. Comput. Appl. Math.*, **234(1)**(2010), 297–304.

[28] A. N'Guessan and F. Bellavance, A confidence interval estimation problem using Schur complement approach, *C. R. Math. Rep. Acad. Sci. Canada*, **27(3)**(2005), 84–91.

[29] A. N'Guessan and I. C. Geraldo, A cyclic algorithm for maximum likelihood estimation using Schur complement, *Numer. Linear Algebra Appl.*, **22(6)**(2015), 1161–1179.

[30] A. N'Guessan and C. Langrand, A covariance components estimation procedure when modelling a road safety measure in terms of linear constraints, *Statistics*, **39(4)**(2005), 303–314.

[31] A. N'Guessan and C. Langrand, A Schur complement approach for computing subcovariance matrices arising in a road safety measure modelling. *J. Comput. Appl. Math.*, **177(2)**(2005), 331–345.

[32] A. N'Guessan and M. Truffier, Impact d'un aménagement de sécurité routière sur la gravité des accidents de la route, *J. SFdS*, **49**(2008), 23–41.

[33] A. N'Guessan, A. Essai and C. Langrand, Estimation multidimensionnelle des contrôles et de l'effet moyen d'une mesure de sécurité routière, *Revue de statistique appliquée*, **49(2)**(2001), 85–102.

[34] A. N'Guessan, A. Essai and M. N'Zi, An estimation method of the average effect and the different accident risks when modelling a road safety measure: A simulation study, *Comput. Statist. Data Anal.*, **51(2)**(2006), 1260–1277.

[35] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, second edition, (2006).

[36] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, URL `https://www.R-project.org/`, (2022).

[37] L. M. Rios and N. V. Sahinidis, Derivative-free optimization: a review of algorithms and comparison of software implementations, *J. Global Optim.*, **56(3)**(2013), 1247–1293.

[38] R. Varadhan, *alabama: Constrained Nonlinear Optimization*, URL `https://CRAN.R-project.org/package=alabama`, R package version 2015.3-1, (2015).

[39] H. Zhou and K. Lange, MM algorithms for some discrete multivariate distributions, *J. Comput. Graph. Statist.*, **19(3)**(2010), 645–665.