ON CONVERGENCE OF CONVEX MINORANT ALGORITHMS FOR DISTRIBUTION ESTIMATION UNDER CENSORED DATA

by

Jorge Aragón and David Eberly

Technical Report No. 283
August 1991

Prepared Under the Auspices of
National Science Foundation Grant
DMS 90-02502
Ingram Olkin, Project Director

Department of Statistics
Stanford University
Stanford, California
ON CONVERGENCE OF CONVEX MINORANT ALGORITHMS FOR DISTRIBUTION ESTIMATION UNDER CENSORED DATA

by

Jorge Aragón and David Eberly
Harvard School of Public Health and University of Texas at San Antonio

Technical Report No. 283

Prepared Under the Auspices of
National Science Foundation Grant
DMS 90-02502
Ingram Olkin, Project Director

Department of Statistics
Stanford University
Stanford, California
On Convergence of Convex Minorant Algorithms for
Distribution Estimation under Censored Data

Jorge Aragón¹ and David Eberly²
Harvard School of Public Health and University of Texas at San Antonio

Summary

Proof of local convergence of the convex minorant algorithm to obtain the non-parametric maximum likelihood estimator of a distribution under interval censored observations is given. We also provide a variation of the CM algorithm which yields global convergence.

1 Introduction

Non-parametric maximum likelihood estimator (NPMLE) of a distribution has been discussed in a variety of circumstances. Kaplan and Meier (1958) introduced the product-limit estimator of a survival distribution for right censored data. Turnbull (1974) developed the expectation-maximization (EM) algorithm for doubly-censored data which is an iterative method based on the self-consistency concept introduced by Efron (1967). Groeneboom (1990) presented an alternative method called convex minorant (CM) algorithm for interval censored observations. The CM algorithm gives the NPMLE of the distribution under left, right or interval censoring. Exact observations are not considered. Consistency of the NPMLE has been addressed by Turnbull (1976), Chang and Yang (1987), Groeneboom (1987) and Van de Geer (1990), and the large sample theory by Groeneboom (1990), Chang (1990) and Samuelesen (1989).

The estimation problem for interval censored data as stated by Groeneboom (1990) is the following. Let \((X_i, T_i, U_i), i = 1, \ldots, n\) be a random sample where \(X_i\) is a nonnegative random variable with distribution \(F\) and where \(T_i\) and \(U_i\) are (nonnegative) random variables independent of \(X_i\) such that \(T_i \leq U_i\) with probability one. \(X_i\) is unobservable. We only observe \(T_i, U_i\) and the indicators \(\epsilon_i = \{X_i \leq T_i\}, \gamma_i = \{T_i < X_i \leq U_i\}\). The corresponding loglikelihood is

\[
\phi(F) = \sum_i \left[ \epsilon_i \ln F(T_i) + \gamma_i \ln (F(U_i) - F(T_i)) + (1 - \epsilon_i - \gamma_i) \ln(1 - F(T_i)) \right].
\]

¹Research partially supported by grant NSF DMS90-02502
²Research partially supported by grant NSF DMS90-03037.


Key words and phrases. Censored data, concave programming, convex minorant, nonparametric maximum likelihood, Newton-Jacobi method.
The NPMLE of \( F \) is defined as the (right-continuous) distribution maximizing \( \phi \). Letting \( y_i = F(T_{(i)}) \) where \( T_{(i)} \) is the \( i \)th order statistic of \( \{T_1, ..., T_n, U_1, ..., U_n\} \), the problem of obtaining the NPMLE can be formulated as the problem of maximizing

\[
\phi(y) = \sum_{i \in I_0} \ln(y_i) + \sum_{i \in I_1} \ln(1 - y_i) + \sum_{i, i' \in I_2} \ln(y_i - y_{i'})
\]

over the simplex \( S = \{y \in \mathbb{R}^m : 0 \leq y_1 \leq \cdots \leq y_m \leq 1\} \) where \( \{1, \ldots, m\} = I_0 \cup I_1 \cup I_2 \) is a disjoint union, and where \( i' < i \) is an index dependent on the choice of \( i \). Notice that \( \phi : S \to [-\infty, 0] \).

For example,

\[
\phi(y) = \ln(1 - y_1) + \ln(y_2) + \ln(y_5 - y_3) + \ln(1 - y_4) + \ln(1 - y_8) + \ln(y_7) + \ln(y_9 - y_6)
\]

with \( 0 \leq y_1 \leq \cdots \leq y_9 \leq 1 \). The index sets are \( I_0 = \{2, 7\} \), \( I_1 = \{1, 4, 8\} \), and \( I_2 = \{3, 5, 6, 9\} \).

In general, \( \phi \) is concave on \( S \) and has a unique maximum which occurs at some \( y^* \in \partial S \) (boundary of \( S \)). The convex minorant algorithm is an iteration scheme to construct \( y^* \). Although the algorithm may be implemented directly using \( \phi \) and \( S \), when viewed in a reduced space it is a Newton–Jacobi 1-step method with an adjustment at each iteration to guarantee that the new iterate is back in the reduced space. We briefly describe the convex minorant algorithm in section 2. Reductions to a smaller space are discussed in section 3. The convergence properties of the convex minorant algorithm are analyzed in section 4. In general there is local convergence, but not global convergence. In fact, for many examples, the set of initial values for which there is convergence is very small. Section 5 describes a variation of the convex minorant algorithm using damping, which yields global convergence.

## 2 Convex Minorant Algorithm

Let \( S \subset \mathbb{R}^m \) and \( \phi(y) \) be defined as in (1). We use the notation \( \phi_i = \partial \phi / \partial y_i \) and \( \phi_{ij} = \partial^2 \phi / \partial y_i \partial y_j \). Define

\[
(2) \quad A_i(y) = \sum_{j=1}^{i} [\phi_j(y) - y_j \phi_{jj}(y)], \quad Q_i(y) = -\sum_{j=1}^{i} \phi_{jj}(y), \quad \text{and} \quad \Delta_{i,l}(y) = \frac{A_i(y) - A_l(y)}{Q_i(y) - Q_l(y)}
\]

for \( i = 0, \ldots, m \) and for \( l < i \). The convex minorant algorithm is as follows. Select an initial guess \( y^{(0)} \in S \). For each iterate \( y^{(k)} \), \( k \geq 0 \), set \( i_0 = 0 \) and construct the set of indices \( i_0 < i_1 < \cdots < i_l = m \) such that

\[
(3) \quad \Delta_{i_j,i_{j-1}}(y^{(k)}) = \min \left\{ \Delta_{r,i_{j-1}}(y^{(k)}) \right\}_{r=i_{j-1}+1}^{m}
\]
for \( j = 1, \ldots, \ell = \ell(k) \). Set

\[ y_i^{(k+1)} = \Delta_{i,i-1} (y^{(k)}), \quad i_{j-1} < i \leq i_j \]

for \( j = 1, \ldots, \ell \). Note that if \( 0 \leq \Delta_{1,0} \leq \Delta_{2,1} \leq \cdots \leq \Delta_{m-1,m-1} \leq 1 \), then \( \ell = m, i_j = j \), and

\[ y_i^{(k+1)} = \Delta_{i,i-1} (y^{(k)}) = y_i^{(k)} - \phi_i (y^{(k)}) / \phi_{ii} (y^{(k)}) \]

for all \( i \). But in general, since \( y^* \in \partial S \), we will have \( \ell < m \) and quite a few blocks of equal variables. The components of \( y^{(k+1)} \) can be seen as the slopes of the greatest CM of the cumulative sum diagram given by the points \( P_i = (Q_i(y^{(k)}), A_i(y^{(k)})) \), \( i = 0, 1, \ldots, m \) where \( P_0 = (0,0) \) (See Groeneboom, 1990).

3 Redductions

We may reduce the number of independent variables in \( \phi \). The first type of reduction, called a primary reduction, can be made a priori. The second type of reduction, called a secondary reduction, may or may not be made dependent on the structure of \( \phi \). For theoretical disucussions, we will make secondary reductions, but for numerical implementation, identifying secondary reductions a priori appears to be intractable. The implementation will be able to make the necessary reductions. In fact, if one implements the convex minorant algorithm as stated in the last section, it will make both primary and secondary reductions itself.

**EXAMPLE 1.** This example illustrates primary reductions. The reduced space will still be a simplex, and the reduced function is strictly concave. Consider \( S \subset \mathbb{R}^6 \) and

\[ \phi(y) = \ln(1 - y_1) + \ln(y_2) + \ln(y_3 - y_5) + \ln(1 - y_4) + \ln(1 - y_6) + \ln(y_7) + \ln(y_9 - y_6). \]

To maximize \( \phi \), set \( y_2, y_5, y_7, \) and \( y_9 \) as large as possible, and set \( y_1, y_3, y_4, y_6, \) and \( y_8 \) as small as possible. Thus, we can make the reduction: \( 0 = y_1, z_1 := y_2 = y_3 = y_4, z_2 := y_5 = y_6, z_3 := y_7 = y_8, \) and \( 1 = y_9 \). The reduced space is \( T = \{ z \in \mathbb{R}^3 : 0 \leq z_1 \leq z_2 \leq z_3 \leq 1 \} \) and the reduced function is \( \theta : T \to [-\infty, 0] \) given by

\[ \theta (z) = \ln(z_1) + \ln(1 - z_1) + \ln(1 - z_2) + \ln(z_3) + \ln(1 - z_3) + \ln(z_2 - z_1). \]

**EXAMPLE 2.** This example illustrates secondary reductions. Suppose that after primary reductions the reduced simplex is \( T \subset \mathbb{R}^2 \) and the reduced function is

\[ \theta(z) = a_1 \ln(z_1) + b_1 \ln(1 - z_1) + a_2 \ln(z_2) + b_2 \ln(1 - z_2) \]
where \( a_1, a_2, b_1 \) and \( b_2 \) are positive. Notice that \( \theta(x) \) is defined on a larger set, \( T_{\text{max}} = [0, 1]^2 \). A maximal domain for \( \theta \) larger than \( T \) generally occurs whenever linking terms are absent. In our example, there is no linking term \( \ln(2 - 1) \). The solution to \( \theta'(z) = 0 \) is \( z^* = (a_1/(a_1 + b_1), a_2/(a_2 + b_2)) \). If \( a_1/(a_1 + b_1) \leq a_2/(a_2 + b_2) \) [or equivalently, \( a_1/a_2 \leq b_1/b_2 \)], then \( z^* \) is the point at which \( \theta \) attains its maximum in \( T \). However, if \( a_1/a_2 > b_1/b_2 \), then the maximum of \( \theta \) must occur on \( \partial T \), that is, on the boundary where \( z_1 = z_2 \). We can reduce \( T \) to the lower dimensional simplex \( R = [0, 1] \) and we can reduce \( \theta \) to \( \psi(x) = (a_1 + a_2) \ln(x) + (b_1 + b_2) \ln(1 - x) \) for \( x \in R \). In this setting, \( \psi \) is strictly concave, and the maximum of \( \psi \) occurs at an interior point. The maximum occurs when \( \psi'(x) = 0 \), at \( x^* = (a_1 + b_1)/(a_1 + b_1 + a_2 + b_2) \).

In general, one can make primary (and if necessary, secondary) reductions to a smaller dimensional simplex \( R = \{ x \in \mathbb{R}^r : 0 \leq x_1 \leq \cdots \leq x_r \leq 1 \} \) and \( \phi \) to a strictly concave function \( \psi : R \to [-\infty, 0] \) given by

\[
\psi(x) = \sum_{i=1}^{r} a_i \ln(x_i) + \sum_{i=1}^{r} b_i \ln(1 - x_i) + \sum_{i=2}^{r-1} \sum_{j=1}^{i-1} c_{ij} \ln(x_i - x_j)
\]

where \( a_i, b_i, \) and \( c_{ij} \) are nonnegative integers with \( a_1 > 0 \) and \( b_r > 0 \). If all the necessary reductions have been made, then the maximum of \( \psi \) occurs at \( x^* \in \mathbb{R}^r \) (the interior of \( R \)) and \( \psi'(x^*) = 0 \). Note that the maximal domain of \( \psi \), call it \( R_{\text{max}} \), may be a strictly larger set than \( R \), and \( \psi(\partial R_{\text{max}}) = -\infty \).

The convex minorant can be implemented in the primary reduced setting (5). In (2), (3), and (4), replace \( S \) by \( T \), \( \phi \) by \( \psi \), and \( y \) by \( x \). As in the nonreduced case, if the \( \Delta \) terms are ordered, then \( x_i^{(k+1)} = x_i^{(k)} - \psi_i(x^{(k)})/\psi_{ii}(x^{(k)}) \) for all \( i \). The probability of obtaining blocks of equal variables is much smaller in the reduced setting if \( x^* \) is an interior point of \( R \). However, if \( x^* \in \partial R \), then we might obtain some small blocks. Since identifying secondary reductions \( \text{a priori} \) is intractable, the implementation of the algorithm will have to make these reductions on the fly.

Our later analysis requires looking at the derivatives of \( \psi(x) \). The first partial derivatives are

\[
\psi_i = \frac{a_i}{x_i} + \frac{b_i}{x_i - 1} + \sum_{j=1}^{i-1} \frac{c_{ij}}{x_i - x_j} + \sum_{j=i+1}^{r} \frac{c_{ji}}{x_i - x_j}
\]

and the second partial derivatives are

\[
\psi_{ii} = -\frac{a_i}{x_i^2} - \frac{b_i}{(x_i - 1)^2} - \sum_{j=1}^{i-1} \frac{c_{ij}}{(x_i - x_j)^2} - \sum_{j=i+1}^{r} \frac{c_{ji}}{(x_i - x_j)^2},
\]

and

\[
\psi_{ij} = \frac{c_{ij}}{(x_i - x_j)^2} \quad \text{for} \; j < i, \; \psi_{ji} = \psi_{ij}.
\]
As pointed out earlier, it is difficult to identify secondary reductions a priori, but for theoretical discussions we will make these reductions (if necessary). The next example indicates that (5) can be decomposed into a sum of functions, each containing terms from a subset of the variables that are maximally linked.

**EXAMPLE 3.** Consider the function

$$
\phi(y) = \ln(y_1) + \ln(y_4 - y_2) + \ln(1 - y_9) + \ln(y_5) + \ln(1 - y_6) + \ln(y_8) + \ln(1 - y_9) \\
+ \ln(y_{10}) + \ln(y_{11} - y_7) + \ln(1 - y_{12})
$$

where 0 ≤ y_1 ≤ ⋯ ≤ y_{12} ≤ 1. Primary reductions yield x_1 := y_1 = y_2 = y_3, x_2 := y_4 = y_5 = y_6 = y_7, x_3 := y_8 = y_9, and x_4 := y_{10} = y_{11} = y_{12}. The reduced function is

$$
\psi(x) = [\ln(x_1) + \ln(1 - x_1) + \ln(x_2) + \ln(1 - x_2) + \ln(x_4) + \ln(1 - x_4) \\
+ \ln(x_2 - x_1) + \ln(x_4 - x_2)] + [\ln(x_3) + \ln(1 - x_3)]
$$

where 0 ≤ x_1 ≤ x_2 ≤ x_3 ≤ x_4 ≤ 1. The maximal domain is

$$
R_{\text{max}} = \{x \in \mathbb{R}^4 : 0 \leq x_1 \leq x_2 \leq x_4, 0 \leq x_3 \leq 1\}
$$

and the maximally linked variables are \{x_1, x_2, x_4\} and \{x_3\}. If \psi'(x) = 0 has solution x^* such that x_3^* is not in the correct ordering with the other components, then we should make a secondary reduction. However, for this example, the solution to \psi'(x) = 0 is x^* = ((3 - \sqrt{3})/6, 1/2, 1/2, (3 + \sqrt{3})/6) ∈ R^4, so no secondary reduction is required. Notice that x_3^* = 1/2 is the solution to the maximally linked block \ln(x_3) + \ln(1 - x_3).

With the proper permutation of the variables of \psi, the second derivative matrix can be written as a block diagonal matrix. Each block corresponds to a maximally linked set of variables, so the block is an irreducible matrix. In example 3,

$$
P^T \psi''(x) P = \begin{bmatrix}
\psi_{11} & \psi_{12} & \psi_{14} & 0 \\
\psi_{12} & \psi_{22} & \psi_{24} & 0 \\
\psi_{14} & \psi_{24} & \psi_{44} & 0 \\
0 & 0 & 0 & \psi_{33}
\end{bmatrix}
$$

and the 3 × 3 and 1 × 1 blocks are irreducible. The permutation matrix P corresponds to the permutation (1 2 3 4) → (1 2 4 3).
4 Convergence

We now consider convergence properties of the convex minorant algorithm in the reduced setting (5). The convex minorant algorithm is a special case of the m-step Newton-Jacobi method

$$x^{(k+1)} = x^{(k)} - \left[ \sum_{i=0}^{m-1} H(x^{(k)})^i \right] B(x^{(k)})^{-1} F(x^{(k)})$$

where $F'(x) = B(x) - C(x)$ is some splitting of $F'$ and $H(x) = B(x)^{-1}C(x)$. For our case, $F(x) = \psi(x)$, $F'(x) = \psi'(x)$, $B(x) = \text{Diag}(\psi_H)$, and $m = 1$.

The results we use to show local convergence are found in Ortega (1970), Result 10.3.1, pg. 321. Lemma 1 contains stronger hypotheses than that of the referenced result, but they suffice for our case. In the statement of the lemma the following notation is used. The set $\mathbb{R}^{r \times r}$ denotes the set of $r \times r$ matrices with real entries. The real number $\rho(M)$ is the spectral radius of matrix $M$. The root-convergence of the iteration scheme is defined by

$$R_1(x^*) = \sup \left\{ \limsup_{k \to \infty} ||x^{(k)} - x^*||^{1/k} \right\}$$

where the supremum is taken over all sequences generated by the iteration scheme. $R_1$ is independent of norm.

**Lemma 1** Suppose $F : D \subseteq \mathbb{R}^r \to \mathbb{R}^r$ is continuously differentiable, $F(x^*) = 0$, $B : D \to \mathbb{R}^{r \times r}$ is continuous and nonsingular at $x^*$, and $\rho(H(x^*)) < 1$. For any $m \geq 1$, $x^*$ is a point of attraction of (6) and $R_1(x^*) = \rho(H(x^*))^m < 1$.

In applying this result to our case, $D = \mathbb{R}^r$, $F(x) = \psi'(x)$ which is continuously differentiable on $\mathbb{R}^r$, and $x^*$ is the unique solution to $\psi'(x) = 0$ in $\mathbb{R}^r$. Clearly $B(x) = \text{Diag}(\psi_H(x))$ is continuous and nonsingular everywhere on $\mathbb{R}$ since $\psi_H(x) < 0$. We only need to show that $\rho(H(x^*)) < 1$. In fact we will show that $\rho(H(x)) < 1$ for all $x \in \mathbb{R}^r$. To do so requires some results from the theory of nonnegative matrices [Horn (1985), Theorem 8.1.22, pg. 492]:

**Lemma 2** Let $A = [a_{ij}] \in \mathbb{R}^{r \times r}$ be a nonnegative matrix with spectral radius $\rho(A)$. The spectral radius must be an eigenvalue and $$\min \sum_j a_{ij} \leq \rho(A) \leq \max \sum_j a_{ij}.$$ and [Horn (1985), Corollary 6.2.27, pg. 363]:

6
**Lemma 3** If $A = [a_{ij}] \in \mathbb{R}^{n \times n}$ is irreducible and $|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$ for all $i$, but with at least one strict inequality, then $A$ is invertible.

The next theorem is the local convergence result for the convex minorant algorithm (2,3,4).

**Theorem 4** Let $H(x) = B^{-1}(x)C(x)$ where $B = \text{Diag}(\psi_{ii}(x))$ and $C(x) = B(x) - \psi''(x)$. For any $x \in \mathbb{R}^n$, $\rho(H(x)) < 1$. Consequently, the convex minorant algorithm converges as long as the initial guess is sufficiently close to $x^*$.

**Proof.** The following arguments are pointwise for $x \in \mathbb{R}^n$, so we suppress $x$ in the notation. First note that $-\psi_{ii} = q_i + \sum_{j \neq i} \psi_{ij}$ where $q_i = a_i/x_i^2 + b_i/(x_i - 1)^2$. Therefore, the $i$th row sum of $H$ is

$$\frac{\sum_{j \neq i} \psi_{ij}}{q_i + \sum_{j \neq i} \psi_{ij}} \in [0, 1]$$

for any $x \in \mathbb{R}^n$. Since $H$ is a nonnegative matrix, Lemma 2 implies $\rho(H) \leq 1$.

Suppose that $\rho(H) = 1$; then 1 is an eigenvalue for $H$ and the matrix $I - H = B^{-1}(B - C) = B^{-1}\psi''(x)$ is singular. As indicated in the last section, there is a permutation of the independent variables represented by a permutation matrix $P$ such that $P^T \psi''(x)P = \text{Diag}(M_k)$ is a block diagonal matrix whose blocks are irreducible.

Since $\psi''(x)$ is singular, some block $M_k$ must be singular. For the sake of the argument, suppose that $M_1 = [m_{ij}]$ is the singular block and that the maximally linked variables are $x_1, \ldots, x_t$. For each $i$,

$$|m_{ii}| = |\psi_{ii}| = q_i + \sum_{j \neq i} \psi_{ij} \geq \sum_{j \neq i} \psi_{ij} = \sum_{j \neq i} |m_{ij}|.$$  

Moreover, since $q_i > 0$ (in the reduction $a_1 > 0$), strict inequality occurs at least once. By Lemma 3, $M_k$ is invertible, a contradiction. The same argument works for any block, since the reductions force at least one of $q_i > 0$.

It must be that $\rho(H) < 1$. By Lemma 1, the solution to $\psi'(x^*) = 0$ is a point of attraction for $x$ in a sufficiently small neighborhood of $x^*$. □

We now give some examples that demonstrate the lack of global convergence. In fact, these examples show that a suitable initial value may be difficult to select. All the problems that can arise to prevent convergence of the iterates are illustrated in the case $m = 2$. In this case, the simplex is $\mathcal{R} = \{(x_1, x_2) : 0 \leq x_1 \leq x_2 \leq 1\}$.
the general reduced function is

$$
\psi(x_1, x_2) = a_1 \ln(x_1) + b_1 \ln(1 - x_1) + a_2 \ln(x_2) + b_2 \ln(1 - x_2) + c \ln(x_2 - x_1),
$$

with $a_1 > 0, b_2 > 0, a_2 > 0$ and $c > 0$. Define $s_1 = a_1 + b_1, s_2 = a_2 + b_2, p_1 = (s_1 + c)(s_1 + s_2 + c), q_1 = (s_1 + c)(2a_1 + c) + cs_2 + a_1 s_2 + a_2 s_1, r_1 = a_1 (a_1 + a_2 + c), p_2 = (s_2 + c)(s_1 + s_2 + c), q_2 = (s_2 + c)(2a_2 + c) + cs_1 + a_1 s_2 + a_2 s_1,$

and $r_2 = a_2 (a_1 + a_2 + c)$. Solving for $\psi'(x) = 0$ algebraically, solutions $(x_1, x_2)$ satisfy

$$(x_1, x_2) = \left( \frac{q_1 \pm \sqrt{q_1^2 - 4p_1 r_1}}{2p_1}, \frac{q_2 \mp \sqrt{q_2^2 - 4p_2 r_2}}{2p_2} \right).$$

For different choices of coefficients, we selected points $(i/100, j/100) \in R$ for $j = 1, \ldots, 99$ and $i < j$ as the initial guess for the iterations. We kept track of those points for which the iterates converged to $x^*$. Sometimes the iterates did not converge in that eventually they exited $R$. We flagged those initial points depending on if the exiting iterate $(x_1, x_2)$ satisfied $x_1 < 0$ (type 1), $x_1 > x_2$ (type 2), or $x_2 > 1$ (type3). The stopping criterion was $\max\{\psi_1(x^{(k)}), \psi_2(x^{(k)})\} \leq 10^{-8}$.

EXAMPLE 4. Let $c = 8$ and $a_1 = a_2 = b_1 = b_2 = 1$. The iterates converged for 2464 initial values. There were 1300 type 1 points, 0 type 2 points, and 1087 type 3 points. For $c = 10$ and $a_1 = a_2 = b_1 = b_2 = 1$, the iterates converged for only 7 initial values! There were 3353 type 1 points, 0 type 2 points, and 1291 type 3 points. Clearly the probability of randomly selecting a good initial guess can be quite small.

Let $a_1 = b_2 = 10, a_2 = b_1 = c = 1$. The iterates converged for 2067 initial values. There were 0 type 1 points, 2784 type 2 points, and 0 type 3 points.

Let $b_1 = 9, a_1 = a_2 = b_2 = c = 1$. The iterates converged for 3379 initial values. There were 1472 type 1 points, 0 type 2 points, and 0 type 3 points. In fact, there appears to be a sequence of disjoint sets $\{E_k\}_{k=1}^\infty$ such that if $x^{(0)} \in E_k$ is an initial guess, then $x^{(p)} \in E_{k-p}$ for $p = 1, \ldots, k-1$, and $x^{(k)}$ is outside of the simplex. Thus, there are initial values such that the iterates exit the simplex in an arbitrarily large number of steps.

5 Damped Convex Minorant Algorithm

The examples of the last section demonstrate that one could have difficulty choosing an initial guess so that the convex minorant algorithm converges. The problem with convergence of iterates is that the steplength
is too large for those examples. We suggest a modified algorithm which guarantees global convergence for any \( z^{(0)} \in R^n \). Define the damped convex minorant algorithm by

\[
x^{(k+1)} = x^{(k)} - \alpha_k B(x^{(k)})^{-1} \psi'(x^{(k)}), \quad k \geq 0
\]

where \( B(x) = \text{Diag}(\psi''(x)) \). The steplengths \( \alpha_k \in (0, 1] \) can be constructed by a few methods to be mentioned later, and will have the property \( \alpha_k \to 1 \) as \( k \to \infty \). For \( x^{(k)} \) sufficiently close to \( z^* \), the algorithm will essentially be the convex minorant algorithm. The variable steplengths are small initially to guarantee that the iterates do not exit \( R \).

For global convergence, we use Result 14.4.2 in Ortega (1970), pg. 502. In the statement of the lemma the following notation is used. Given a function \( g(x) : D \subset R^n \to R \), the set \( L(g(x)) = \{ y \in D : g(y) \leq g(x) \} \) is the connected component of the level set of \( g(x) \) which contains \( x \) itself.

**THEOREM 5** Assume that \( g : D \subset R^n \to R \) is continuously differentiable on the open set \( D \), that there is an \( z^{(0)} \in D \) such that \( L := L(g(z^{(0)})) \) is compact, and that \( g \) has a unique critical point \( z^* \in L \). Suppose that \( A(x) \in R^{n \times n} \) is positive definite for all \( x \in L \). Then the iterates

\[
x^{(k+1)} = x^{(k)} - \alpha_k A(x^{(k)})^{-1} g'(x^{(k)}), \quad k \geq 0
\]

converge to \( z^* \) for any sequence of \( \alpha_k \) such that \( \{ z^{(k)} \} \subset L \) and \( \lim_{k \to \infty} g'(z^{(k)})^T p^{(k)}/\|p^{(k)}\| = 0 \) where \( p^{(k)} = A(x^{(k)})^{-1} g'(x^{(k)}) \).

For our application, we choose \( g(x) = -\psi(x) \), \( D = R_n^{\max} \), and \( A(x) = \text{Diag}(-\psi''(x)) \). The iteration scheme is then given by (7). Since \( g \) is strictly convex, the level sets are convex, and so the set \( L \) in the statement of the lemma is always compact for any \( z^{(0)} \in R_n^{\max} \). There are a number of methods for computing steplengths to guarantee that \( \{ z^{(k)} \} \subset L \), for example the Goldstein–Armijo algorithm [Ortega (1970)]. A method to compute \( \alpha_k \) which is easy to implement is the following. Choose \( \alpha_k = \max\{ (1/2)^m : m \geq 0, \, x^{(k)} - (1/2)^m B(x^{(k)})^{-1} \psi'(x^{(k)}) \in L(g(x^{(k)})) \} \)

and set \( z^{(k+1)} = x^{(k)} - \alpha_k B(x^{(k)})^{-1} \psi'(x^{(k)}) \).

We applied this algorithm to the examples in Example 4. In all cases we had global convergence. The implementation of the algorithm using the C language is given in the appendix.
Acknowledgement. Our interest in this topic originated from the 1990 Summer course taught by Piet Groeneboom at Stanford University. We thank him for his valuable comments and for providing us with the original version of his algorithm written in Pascal which is equivalent to routine cm_iterate() in the Appendix. He has communicated to us recently that the material of the course which includes interval censoring and an updated version of the convex minorant algorithm (different than the modified algorithm in this paper) will appear in a Technical Report in the Statistics Department in Stanford University. It will also appear in a book with Jon Wellner, published by Birkhauser in the series DMV lectures notes, tentatively titled "Statistics in Large Parameter Spaces."

6 Appendix

/**********************************************************
 * Damped Convex Minorant Algorithm
 **********************************************************/

#include <math.h>       /* log, fabs */
#include <malloc.h>      /* calloc, free */
#include <stdio.h>       /* FILE, fopen, stderr, fprintf, fgets, scanf, NULL,
                         fclose */
#include <stdlib.h>      /* exit, qsort, abs */
#include <string.h>      /* strtok */
#include <values.h>      /* MAXINT, MAXDOUBLE, MINDDOUBLE */

typedef double real;   /* could use "float" or other numeric type */

real toler;            /* tolerance for stopping condition */
int max_iter;          /* maximum iterations to run algorithm */
int xcount;            /* number of variables, indices = 1..xcount */
real *x;               /* x[0]..x[xcount+1], x[0]=0, x[xcount+1]=1 */
int *link;             /* array to keep track of linked variables */
typedef struct { /* represents \( A_i, Q_i \) in cumulative sum diagram */
    real gg, vv;
} POINT;

POINT *pt;

/**
 * void error (char *message)
 *
 * fprintf(stderr,"%s\n",message);
 * exit(1);
 */

/**
 * real psi (real x[])--/ function to maximize */
 *
 * int i;
 * real sum = 0.0;
 *
 * for (i = 1; i <= xcount; i++)
 *     if ( link[i] == 0 )
 *         sum += log(x[i]);
 *     else if ( link[i] == xcount+1 )
 *         sum += log(1-x[i]);
 *     else if ( i > link[i] )
 *         sum += log(x[i]-x[link[i]]);
 *
 * return sum;
 */

/**
 * real psi_p (int i, real x[])--/ first partial derivatives of psi */
 *
 * if ( link[i] == 0 )
return 1/x[i];
else if ( link[i] == xcount+1 )
    return 1/(x[i]-1);
else
    return 1/(x[i]-x[link[i]]);
}

real psi_pp (int i, real x[]) /* second partial derivatives of psi */
{
    if ( link[i] == 0 )
        return -1/x[i]/x[i];
    else if ( link[i] == xcount+1 )
        return -1/(x[i]-1)/(x[i]-1);
    else
        return -1/(x[i]-x[link[i]])/(x[i]-x[link[i]]);
}

int within_tolerance (real x[], real fp[], real toler)
{
    /* stopping condition for the algorithm */
    /* test for small gradient components in reduced space */

    int i, j;
    real grad;

    for (i = 1; i <= xcount; ) {
        grad = fp[i];
        for (j = i+1; j <= xcount; j++)
            if ( x[j] == x[i] )
                grad += fp[j];
            else
                break;
        if ( fabs(grad) > toler )
return 0;
    
i = j;
}

return 1;
}
                        /********************************************************/

void cm_iterate (real x[], real y[], real fp[], real fpp[], real step)
{
    /* the convex minorant algorithm (without modification) */

    int i, j, min_index;
    real slope, min_slope;

    /* construct cumulative sum diagram points */
    pt[0].gg = 0;
    pt[0].vv = 0;
    for (i = 1; i <= xcount; i++) {
        pt[i].gg = pt[i-1].gg - fpp[i];
        pt[i].vv = pt[i-1].vv - fpp[i]*x[i] + step*f[i];
    }

    /* construct left derivatives of convex minorant */
    for (i = 0; i < xcount; ) {
        min_slope = MAXDOUBLE;
        for (j = i+1; j <= xcount; j++) {
            slope = (pt[j].vv-pt[i].vv)/(pt[j].gg-pt[i].gg);
            if (slope <= min_slope) {
                min_slope = slope;
                min_index = j;
            }
        }
        for (j = i+1; j <= min_index; j++)
if (min_slope <= 0)
    y[j] = 0;
else if (min_slope >= 1)
    y[j] = 1;
else
    y[j] = min_slope;
    i = min_index;
}

/* output: convex minorant iterate y[] */
}

int valid_iterate (real x[], int link[])
{
    /* The convex minorant algorithm may produce an iterate on the boundary of the simplex at which psi is -infinity. This routine checks to make sure psi is defined at this point. If it is, the function returns TRUE, otherwise if psi is undefined at the iterate, the function returns FALSE. */

    int i, j, start, final;

    /* check 0 boundary */
    for (i = 1; i <= xcount; i++) {
        if (x[i] > 0)
            break;
        if (i > link[i]) /* psi(x) = -infinity */
            return 0;
    }
    start = i;

    /* check 1 boundary */
    for (j = xcount; j >= 1; j--) {
if (i > x[j])
    break;

if (j < link[j]) /* psi(x) = -infinity */
    return 0;

final = j;

/* check middle boundaries */
while (start < final) {
    for (i = start+1; i <= final; i++) {
        if (x[i] > x[start])
            break;
        for (j = start; j < i; j++)
            if (link[i] == j) /* psi(x) = -infinity */
                return 0;
    }
    start = i;
}

return 1;

int damped_cm_algorithm (real x[]) /* IN = guess, OUT = maximum point */
{
    int iter, i;
    real old_level, new_level, step;
    real *y, *fp, *fpp;

    y = (real *) calloc(xcount+2,sizeof(real));
    fp = (real *) calloc(xcount+2,sizeof(real));
    fpp = (real *) calloc(xcount+2,sizeof(real));
    if (!y || !fp || !fpp )
        error("out of memory");
old_level = psi(x); /* keep track of current level set */
for (iter = 0; iter < max_iter; iter++) {
    for (i = 1; i <= xcount; i++) {
        fp[i] = psi_p(i,x);
        fpp[i] = psi_pp(i,x);
    }
    if ( within_tolerance(x,fp, toler) )
        break;
    for (step = 1.0; step > MINDCUBE; step /= 2.0) {
        cm_iterate(x,y,fp,fpp,step);
        if ( valid_iterate(y,link) ) {
            new_level = psi(y);
            if ( new_level >= old_level ) { /* new iterate is closer */
                old_level = new_level;
                for (i = 1; i <= xcount; i++)
                    x[i] = y[i];
                break;
            }
        }
    }
    free((char *) fpp);
    free((char *) fp);
    free((char *) y);
    return iter;
}

#define TEST_DRIVER
#ifdef TEST_DRIVER

void main ()
{

    int i, j, count;

    /* initialize parameters */
    max_iter = 256;
    toler = 1e-04;

    /* example: \( \log(x_1) + \log(x_4-x_2) + \log(1-x_3) + \log(1-x_5) \) */

    xcount = 5;
    x = (real *) calloc(xcount+2,sizeof(real));
    if ( !x )
        error("out of memory");

    /* initial guess, \( x[0] = 0 \) and \( x[xcount+1] = 1 \) always */
    x[0] = 0;
    x[1] = 0.1;
    x[2] = 0.2;
    x[3] = 0.3;
    x[4] = 0.4;
    x[5] = 0.5;
    x[xcount+1] = 1;

    /* Indicate linked variables: */
    link = 0 for terms \( \log(x[i]) \)
    link = xcount+1 for terms \( \log(1-x[i]) \)
    i = \text{link}[j], j = \text{link}[i] for terms \( \log(x[i]-x[j]) \) */

    link = (int *) calloc(xcount+2,sizeof(int));
    if ( !link )

17
error("out of memory");
link[1] = 0;
link[2] = 4;
link[3] = xcount+1;
link[4] = 2;
link[5] = xcount+1;

/ * allocate space for cumulative sum points */
pt = (POINT *) calloc(xcount+2,sizeof(POINT));
if ( !pt )
    error("out of memory");

/ * run the algorithm */
printf("iterations = %d\n", damped_cm_algorithm(x));
for (i = 1; i <= xcount; i++)
    printf("x[%d] = %lf\n", i, x[i]);

free((char *) pt);
free((char *) link);
free((char *) x);
}
#endif

References.


Addresses.

Jorge Aragón
Harvard School of Public Health
Biostatistics Department
Boston, Massachusetts 02115

David Eberly
University of Texas at San Antonio
Division of Mathematics, Computer Sciences and Statistics
San Antonio, Texas 78249