Supplementary Materials for "Computationally efficient likelihood inference in exponential families when the maximum likelihood estimator does not exist"

Daniel J. Eck[1] and Charles J. Geyer[2]

1. Department of Statistics, University of Illinois Urbana-Champaign
2. Department of Statistics, University of Minnesota

January 6, 2020

Contents

1 Introduction

This supplemental article contains proofs and code not contained in the main text. The proofs of Theorems 1-3, Theorem 5, Lemma 1 and Theorems 8-10 in Eck and Geyer [2018] are included here. Theorems 1-3, Theorem 5, and Lemma 1 are properties of generalized affine functions. Theorems 8-10 link cumulant generating function (CGF) convergence on neighborhoods of 0 to convergence of moments of all orders. Theorem 12 and Lemma 2 are intermediate results that are stated and proved in this supplement and are not in the main text. The code for all of the calculations to reproduce our examples is included.

The numbering of displayed equations starts at (32) in this supplement. All references to displayed equations under (32) can be found in the main text.

2 Proofs of the properties of generalized affine functions

We first prove Theorem 2.

Proof. Let $F(E)$ denote the space of all functions $E \to \overline{\mathbb{R}}$ with the topology of pointwise convergence. This makes $F(E) = \overline{\mathbb{R}}^E$, an infinite product. Then $F(E)$ is compact by Tychonoff's theorem. We now show that $G(E)$ is closed in $F(E)$ hence compact.

Let g be any point in the closure of $G(E)$. Then there is a net $\{g_\alpha\}$ in $G(E)$ that converges to g. For any x and y in E such that $g(x) < \infty$ and $g(y) < \infty$ and any $t \in (0, 1)$, write $z = x + t(y - x)$.

Then

$$g_\alpha(z) \leq (1-t)g_\alpha(x) + tg_\alpha(y)$$

whenever the right hand side makes sense (is not $\infty - \infty$), which happens eventually, since $g_\alpha(x)$ and $g_\alpha(y)$ both converge to limits that are not ∞. Hence

$$g(z) \leq \lambda g(x) + (1-\lambda)g(y)$$

and g is convex. By symmetry it is also concave and hence is generalized affine. Thus $G(E)$ contains its closure and is closed.

$F(E)$ is Hausdorff because the product of Hausdorff spaces is Hausdorff. $G(E)$ is Hausdorff because subspaces of Hausdorff spaces are Hausdorff. □

In order to prove Theorem 1, an intermediate Theorem is first stated and its proof is provided.

Theorem 12. *An extended-real-valued function h on a finite-dimensional affine space E is generalized affine if and only if $h^{-1}(\infty)$ and $h^{-1}(-\infty)$ are convex sets, $h^{-1}(\mathbb{R})$ is an affine set, and h is affine on $h^{-1}(\mathbb{R})$.*

Proof. To simplify notation, define

$$A = h^{-1}(\mathbb{R}) \tag{32a}$$

$$B = h^{-1}(\infty) \tag{32b}$$

$$C = h^{-1}(-\infty) \tag{32c}$$

First assume h is generalized affine. Then C is convex because h is convex, and B is convex because h is concave. For any two distinct points $x, y \in A$ and any $s \in \mathbb{R}$, The points x, y, and $z = x + s(y - x)$ lie on a straight line. The convexity and concavity inequalities together imply

$$h\bigl(x + s(y - x)\bigr) = (1-s)h(x) + sh(y).$$

It follows that A is an affine set and h restricted to A is an affine function.

Conversely, assume B and C are convex sets, A is an affine set, and h is affine on A. We must show that h is convex and concave. We just prove convexity because the other proof just the same proof applied to $-h$. So consider two distinct points $x, y \in A \cup C$ and $0 < t < 1$ (the convexity inequality is vacuous when either of x or y is in B). Write $z = x + t(y - x)$.

If x and y are both in A, then A being an affine set implies $z \in A$ and the convexity inequality involving x, y, and z follows from h being affine on A. If x and y are both in C, then C being a convex set implies $z \in C$ and the convexity inequality involving x, y, and z follows from $h(z) = -\infty$.

The only case remaining is $x \in A$ and $y \in C$. In this case, there can be no other point on the line determined by x and y that is in A, because A is an affine set. Hence all the points in this line on one side of x must be in B and all the points on the other side must be in C. Thus $z \in C$, and the convexity inequality involving x, y, and z follows from $h(z) = -\infty$. □

We now provide the proof of Theorem 1.

Proof. Again we use the notation in (32a), (32b), and (32c). First we show that all four cases define generalized affine functions. The first three cases obviously satisfy the conditions of Theorem 12.

In case (d), we just prove convexity because the other proof just the same proof applied to $-h$.

If x and y are both in H, then h being generalized affine on H implies the convexity inequality for x and y and any point between them. If x and y are both in C and not both in H, say $x \notin H$, then any point z between x and y is also not in H, and hence is in C because it is on the same side of H as x is. So $h(z) = -\infty$ implies the convexity inequality involving x, y, and z. That completes the proof that all four cases define generalized affine functions.

So we now show that every generalized affine function falls in one of these four cases. Suppose h is generalized affine, and assume that we are not in case (a), (b), or (c). Then at least one of B and C is nonempty. This implies $A \neq E$, hence, A being an affine set, A^c is dense in E. If $B = \emptyset$, then C is dense in E, hence C being a convex set, $C = E$ and we are in case (c) contrary to assumption. Hence $B \neq \emptyset$. The same proof with B and C swapped implies $C \neq \emptyset$.

Hence B and C are disjoint nonempty convex sets, so by the separating hyperplane theorem [Rockafellar, 1970, Theorem 11.3], there is an affine function g on S such that

$$x \notin B, \qquad \text{when } g(x) < 0 \tag{33a}$$

$$x \notin C, \qquad \text{when } g(x) > 0 \tag{33b}$$

and the hyperplane in question is

$$H = \{\, x \in E : g(x) = 0 \,\}.$$

Again we know A^c is dense in E, hence B is dense in the half space on one side of H, and C is dense in the half space on the other side of H. Now convexity of B and C imply

$$x \in C, \qquad \text{when } g(x) < 0 \tag{34a}$$

$$x \in B, \qquad \text{when } g(x) > 0 \tag{34b}$$

That h is generalized affine on H follows from h being generalized affine on E. Thus we are in case (d). □

We now want to show that $G(E)$ is first countable. In aid of that we first prove a lemma.

Lemma 2. *Every finite-dimensional affine space E is second countable and metrizable. If D is a countable dense set in E, then every point of E is contained in the interior of the convex hull of some finite subset of D. The same is true of any open convex subset O of E: every point of O is contained in the interior of the convex hull of some finite subset of $D \cap O$.*

Proof. The first assertion is trivial. If the dimension of E is d, then the topology of E is defined to make any invertible affine function $E \to \mathbb{R}^d$ a homeomorphism.

The second assertion is just the case $O = E$ of the third assertion.

Assume to get a contradiction that the third assertion is false. Then there is a point $x \in O$ that is disjoint from the convex hull of $(O \cap D) \setminus \{x\}$. It follows that there is a strongly separating hyperplane [Rockafellar, 1970, Corollary 11.4.2], hence an affine function g such that

$$\begin{aligned} &g(x) < 0 \\ &g(y) > 0, \qquad y \in O \cap D \text{ and } y \neq x \end{aligned}$$

But this violates x being in O. □

We can now prove Theorem 3.

Proof. We need to show there is a countable local base at h for any $h \in G(E)$. A set is a neighborhood of h if it has the form

$$\{\, g \in G(E) : g(x) \in O_x, \ x \in F \,\}, \tag{35}$$

where F is a finite subset of E and each O_x is a neighborhood of $h(x)$ in $\overline{\mathbb{R}}$.

We prove first countability by induction on the dimension of E using Theorem 1. For the basis of the induction, if $E = \{0\}$, then $G(E)$ is homeomorphic to $\overline{\mathbb{R}}$, hence actually second countable.

We now show that there is a countable local base at h in each of the four cases of Theorem 1. Fix a countable dense set D in E (there is one by Lemma 2).

There is only one h satisfying case (a), the constant function having the value ∞ everywhere. In this case, a general neighborhood (35) contains a neighborhood of the form

$$W = \{\, g \in G(E) : g(x) > m,\ x \in F \,\},$$

where m can be an integer. Also by Lemma 2 there exists a finite subset V of D that contains F in the interior of its convex hull. Then, by concavity of elements of $G(E)$, the neighborhood

$$W_{m,V} = \{\, g \in G(E) : g(x) > m,\ x \in V \,\}$$

is contained in W. Hence the collection

$$\{\, W_{m,V} : m \in \mathbb{N} \text{ and } V \text{ a finite subset of } D \,\} \tag{36}$$

is a countable local base at h.

The proof for case (b) is similar. In case (c) we are considering an affine function h on E. In this case, a general neighborhood (35) contains a neighborhood of the form

$$W = \{\, g \in G(E) : h(x) - \tfrac{1}{m} < g(x) < h(x) + \tfrac{1}{m},\ x \in F \,\},$$

where F is a finite subset of E and m is a positive integer.

Again use Lemma 2 to choose a finite set V containing F in the interior of its convex hull. Then, by convexity and concavity of elements of $G(E)$, the neighborhood

$$W_{m,V} = \{\, g \in G(E) : h(x) - \tfrac{1}{m} < g(x) < h(x) + \tfrac{1}{m},\ x \in V \,\}.$$

is contained in W because any $y \in F$ can be written as a convex combination of the elements of V

$$y = \sum_{x \in V} p_x x,$$

where the p_x are nonnegative and sum to one, so $g \in W_{m,n}$ implies

$$g(y) \le \sum_{x \in V} p_x g(x) < \left(\sum_{x \in V} p_x h(x) \right) + \frac{1}{m} = h(y) + \frac{1}{m}$$

by the convexity inequality, and the same with the inequalities reversed and $1/m$ replaced by $-1/m$ by the concavity inequality. Hence the collection (36) with $W_{m,V}$ as defined in this part is a countable local base at h.

In case (d) we are considering a generalized affine function h that is neither affine nor constant. Then, as the proof of Theorem 1 shows, there is a hyperplane H that is the boundary of $h^{-1}(\infty)$ and $h^{-1}(-\infty)$. The induction hypothesis is that $G(H)$ is first countable, that is, there is a countable family $\mathcal{U}$ of neighborhoods of h in $G(E)$ such that

$$\{\, U \cap H : U \in \mathcal{U} \,\}$$

is a countable local base for $G(H)$ at the restriction of h to H.

Again consider a general neighborhood of h (35); call it W. Let $g|H$ denote the restriction of $g \in G(E)$ to H. For any subset Q of $G(E)$ let $Q|H$ be defined by

$$Q|H = \{\, q|H : q \in Q \,\}.$$

Then the induction hypothesis is that there exists a $U \in \mathcal{U}$ such that $U|H$ is contained in $W|H$.

Also adopt the notation (32a), (32b), and (32c) used in the proofs of Theorems 12 and 1. By Lemma 2 choose a set V_B in $D \cap (B \setminus H)$ that contains $F \cap (B \setminus H)$ in the interior of its convex hull, and choose a set V_C in $D \cap (C \setminus H)$ that contains $F \cap (C \setminus H)$ in the interior of its convex hull, Then, by convexity and concavity of elements of $G(E)$, the neighborhood

$$W_{m,U,V_B,V_C} = \{\, g \in U : h(x) \geq m,\ x \in V_B \text{ and } h(x) \leq -m,\ x \in V_C \,\} \tag{37}$$

is contained in W. To see this, first consider $x \in F \cap H$ (if there are any). Any g in (37) has $g(x) \in O_x$ because of $U|H \subset W|H$. Next consider $x \in F \cap B$ (if there are any). Any g in (37) has $g(x) \in O_x$ because of concavity of g assures $g(x) \geq m$, and we chose m so that $(m, \infty) \subset O_x$. Last consider $x \in F \cap C$ (if there are any). Any g in (37) has $g(x) \in O_x$ because of convexity of g assures $g(x) \leq -m$, and we chose m so that $(-\infty, -m) \subset O_x$.

Hence the collection

$$\{\, W_{m,U,V \cap (B \setminus H), V \cap (C \setminus H)} : m \in \mathbb{N} \text{ and } U \in \mathcal{U} \text{ and } V \text{ a finite subset of } D \,\}$$

is a countable local base at h.

We forgot the case where E is empty. Then $G(E)$ is a one-point space whose only element is the empty function (that has no argument-value pairs). It is trivially first countable. □

We now prove Theorem 5.

Proof. First, assume h satisfies the conditions of Theorem 1 of Eck and Geyer [2018] on E. We then show that h satisfies the conditions of Theorem 5 of Eck and Geyer [2018] by induction on the dimension of E. The induction hypothesis, H(p), is that the conclusions of Theorem 1 imply that the conclusions of Theorem 5 hold when $\dim(E) = p$. We now show that H(0) holds. In this setting, $E = \{0\}$. Therefore our result holds with $j = 0$ and h is constant on E. The basis of the induction holds.

Let $\dim(E) = p + 1$. We now show that H(p) implies that H($p + 1$) holds. In the event that h is characterized by case (a) or (b) of Theorem 1 then our result holds with $j = 0$. If case (c) of Theorem 1 characterizes h then there is an affine function f_1 defined by $f_1(x) = \langle x, \eta_1 \rangle - \delta_1$, $x \in E$, such that $h(x) = +\infty$ for x such that $f_1(x) > 0$, $h(x) = -\infty$ for x such that $f_1(x) < 0$, and h is generalized affine on the hyperplane $H_1 = \{x : f_1(x) = 0\}$. The hyperplane H_1 is p-dimensional affine subspace of E. Now, for some arbitrary $\zeta_1 \in H_1$, define

$$\begin{aligned} V_1 &= \{x - \zeta_1 : x \in H_1\} \\ &= \{y \in E : \langle y, \eta_1 \rangle = \delta_1 - \langle \zeta_1, \eta_1 \rangle\} \\ &= \{y \in E : \langle y, \eta_1 \rangle = 0\} \end{aligned}$$

where the last equality follows from $\zeta_1 \in H_1$. The space V_1 is a p-dimensional vector subspace of E since every affine space containing the origin is a vector subspace [Rockafellar, 1970, Theorem 1.1] and because every translate of an affine space is another affine space [Rockafellar, 1970, pp. 4]. Let

$$h_1(y) = h(y + \zeta_1), \qquad y \in V_1. \tag{38}$$

The function h_1 is convex since the composition of a convex function with an affine function is convex. To see this, let $0 < \lambda < 1$, pick $y_1, y_2 \in V_1$ and observe that

$$\begin{aligned} h_1(\lambda y_1 + (1 - \lambda) y_2) &= h(\lambda y_1 + (1 - \lambda) y_2 + \zeta_1) \\ &\leq \lambda h(y_1 + \zeta_1) + (1 - \lambda) h(y_2 + \zeta_1) \\ &= \lambda h_1(y_1) + (1 - \lambda) h_1(y_2). \end{aligned}$$

A similar argument shows that h_1 is concave. Therefore h_1 is generalized affine. From our induction hypothesis, the conclusions of Theorem 1 imply that our result holds for the generalized affine function h_1. These conditions are that there exist finite sequences of vectors $\tilde{\eta}_2$, …, $\tilde{\eta}_j$ being a linearly independent subset of V_1^*, the dual space of V_1, and scalars $\tilde{\delta}_2$, …, $\tilde{\delta}_j$ such that h_1 has the following form. Define $\widetilde{H}_1 = V_1$ and, inductively, for integers i such that $2 < i \leq j$

$$\begin{aligned} \widetilde{H}_i &= \{\, x \in \widetilde{H}_{i-1} : \langle x, \tilde{\eta}_i \rangle = \tilde{\delta}_i \,\} \\ \widetilde{C}_i^+ &= \{\, x \in \widetilde{H}_{i-1} : \langle x, \tilde{\eta}_i \rangle > \tilde{\delta}_i \,\} \\ \widetilde{C}_i^- &= \{\, x \in \widetilde{H}_{i-1} : \langle x, \tilde{\eta}_i \rangle < \tilde{\delta}_i \,\} \end{aligned} \tag{39}$$

all of these sets (if any) being nonempty. Then $h_1(x) = +\infty$ whenever $x \in \widetilde{C}_i^+$ for any i, $h_1(x) = -\infty$ whenever $x \in \widetilde{C}_i^-$ for any i, and h_1 is either affine or constant on $\widetilde{H}_j$, where $+\infty$ and $-\infty$ are allowed for constant values.

It remains to show that the conditions of Theorem 5 of Eck and Geyer [2018] hold with respect to h. The vectors $\tilde{\eta}_i$, $i = 2, ..., j$ can be extended to form a set of vectors η_i, $i = 2, ..., j$ in E^* by the Hahn-Banach Theorem [Rudin, 1991, Theorem 3.6]. The vectors η_i, $i = 2, ..., j$, form a linearly independent subset of E^*. To see this, let $\sum_{k=2}^{j} a_k \eta_k = 0$ on E for scalars a_k, $k = 2, ..., j$. Then $\sum_{k=2}^{j} a_k \eta_k = 0$ on V_1 which implies that $a_k = 0$ for $k = 2, ..., j$ by the definition of linearly independent. Let $H_0 = E$, and, for $i = 2, ..., j$, define

$$\begin{aligned} H_i &= \{\, x \in H_{i-1} : \langle x, \eta_i \rangle = \delta_i \,\} \\ C_i^+ &= \{\, x \in H_{i-1} : \langle x, \eta_i \rangle > \delta_i \,\} \\ C_i^- &= \{\, x \in H_{i-1} : \langle x, \eta_i \rangle < \delta_i \,\} \end{aligned} \tag{40}$$

where $\delta_i = \tilde{\delta}_i - \langle \zeta_1, \eta_i \rangle$ for $i = 2, ..., j$ and $\widetilde{H}_i = H_i + \zeta_1$ as a result. We see that $h(x) = h_1(x - \zeta_1) = +\infty$ whenever $\langle x + \zeta_1, \eta_i \rangle > \tilde{\delta}_i$. Therefore $h(x) = +\infty$ for all $x \in C_i^+$ for any i. The same derivation shows that $h(x) = -\infty$ whenever $x \in C_i^-$ for any i. The generalized affine function h is either affine or constant on H_j, where $+\infty$ and $-\infty$ are allowed for constant values since the composition of an affine function with an affine function is affine.

We now show that the vectors $\eta_1, ..., \eta_j$ are linearly independent. Assume that $\sum_{k=1}^{j} a_k \eta_k = 0$ on E for scalars a_k, $k = 1, ..., j$. This assumption implies that $\sum_{k=1}^{j} a_k \tilde{\eta} = 0$ on V_1^* where $\tilde{\eta}_1$ is the restriction of η_1 to V_1. Thus $\tilde{\eta}_1$ is an element of V_1^* and $\tilde{\eta}_1 = 0$ on V_1 since $\langle y, \tilde{\eta}_1 \rangle = \langle y, \eta_1 \rangle = 0$ on V_1. Therefore $\sum_{k=2}^{j} a_k \tilde{\eta}_k = 0$ where $a_k = 0$ for $k = 2, ..., j$ from what has already been shown. In the event that $a_1 = 0$, we can conclude that $\eta_1, ..., \eta_j$ are linearly independent. Now consider $a_1 \neq 0$. In this case, $\sum_{k=1}^{j} a_k \eta_k = 0$ implies that $\eta_1 = \sum_{k=2}^{j} b_k \eta_k$ where $b_k = -a_k / a_1$. This states that $\sum_{k=2}^{j} b_k \tilde{\eta}_k = 0$ on V_1. Therefore, $b_k = 0$ for all $k = 2, ..., j$ which implies that η_1 is the zero vector, which is a contradiction. Thus $a_1 = 0$ and we can conclude that $\eta_1, ..., \eta_j$ are linearly independent. This completes one direction of the proof.

Now assume that h satisfies the conclusions of Theorem 5 of Eck and Geyer [2018] and show that these conclusions imply that Theorem 1 of Eck and Geyer [2018] holds by induction on j. The induction hypothesis, H(j), is that the conclusions of Theorem 5 imply that the conclusions of Theorem 1 hold for sequences of length j. For the basis of the induction let $j = 0$. We now show that H(0) holds. The generalized affine function h is either affine or constant on E where $+\infty$ and $-\infty$ are allowed for constant values. This characterization of h is the same as cases (a) of (b) of Theorem 1. The basis of the induction holds.

We now show that H(j) implies that H($j+1$) holds. When the length of sequences is $j+1$, there exist vectors $\eta_1, ..., \eta_{j+1}$ and scalars $\delta_1, ..., \delta_{j+1}$ such that h has the following form. Define $H_0 = E$ and, inductively, for integers i, $0 < i \leq j+1$, such that the sets in (40) are all nonempty. Then $h(x) = +\infty$ whenever $x \in C_i^+$ for any i, $h(x) = -\infty$ whenever $x \in C_i^-$ for any i, and h is either affine or constant on H_{j+1}, where $+\infty$ and $-\infty$ are allowed for constant values. From the definition of the sets H_1, C_1^+, and C_1^-, there is an affine function f_1 defined by $f_1(x) = \langle x, \eta_1 \rangle - \delta_1$, $x \in E$, such that $h(x) = +\infty$ for all $x \in E$ such that $f_1(x) > 0$ and $h(x) = -\infty$ for all $x \in E$ such that $f_1(x) < 0$. This is equivalent to the case (c) characterization of h in Theorem 1, provided we show that the restriction of h to H_1 is a generalized affine function.

Define $V_1 = H_1 - \zeta_1$ for some arbitrary $\zeta_1 \in H_1$. Let $\dim(E) = p$. The space V_1 is a $(p-1)$-dimensional vector subspace of E. Define h_1 as in (38). Let $\tilde{\eta}_i$ be the restriction of η_i to V_1 so that $\tilde{\eta}_i$ is an element of V_1^* for $1 < i \leq j+1$. Now let $\widetilde{H}_1 = V_1$ and, for $1 < i \leq j+1$, we can define the sets as in (39) where $\tilde{\delta}_i = \delta_i - \langle \zeta_1, \tilde{\eta}_i \rangle$. We see that $h_1(x) = h(x + \zeta_1) = +\infty$ whenever $\langle x + \zeta_1, \eta_i \rangle > \tilde{\delta}_i$. Therefore $h_1(x) = +\infty$ for all $x \in \widetilde{C}_i^+$ for any i. The same derivation shows that $h_1(x) = -\infty$ whenever $x \in \widetilde{C}_i^-$ for any i. The generalized affine function h_1 is either affine or constant on H_{j+1}, where $+\infty$ and $-\infty$ are allowed for constant values. Therefore h_1 meets the conditions of Theorem 5 with sequences of length j. From H(j), we know that the conclusions of Theorem 1 hold with respect to h_1. This completes the proof. □

We now prove Lemma 1 using the characterization of generalized affine functions on finite-dimensional vector spaces given by Theorem 5.

Proof. First suppose that h_n converges to h. The assumption that h is finite at at least one point guarantees that h is affine on H_j from Theorem 5. For all $y \in H_j$ we can write $h(y) = \langle y, \theta^* \rangle + a$ where $\langle y, \theta^* \rangle =$

$\sum_{i=j+1}^{p} d_i\langle y, \eta_i\rangle$ and $s, d_i \in \mathbb{R}$. The convergence $h_n \to h$ implies that $b_{i,n} \to d_i$, $i = j+1, ..., p$ where the set of $b_{i,n}$s is empty when $j = p$ and that $a_n \to a$ as $n \to \infty$. Thus conclusions (c) and (d) hold. To show that conclusions (a) and (b) hold we will suppose that $j > 0$, because these conclusions are vacuous when $j = 0$. Both cases (a) and (b) will be shown by induction with the hypothesis H(m) that $b_{(j-m),n} \to +\infty$ and $b_{(j-m+1),n}/b_{(j-m),n} \to 0$ as $n \to \infty$ for $0 \leq m \leq j-1$. We now show that the basis of this induction holds. Pick $y \in C_j^+$ and observe that

$$h_n(y) = a_n + b_{j,n}\left(\langle y, \eta_j\rangle - \delta_j\right) + \sum_{k=j+1}^{p} b_{k,n}\langle y, \eta_k\rangle \to +\infty.$$

since $h(y) = +\infty$ and $h_n \to h$ pointwise. From this, we see that $b_{j,n} \to +\infty$ as $n \to \infty$ and $b_{j+1,n}/b_{j,n} \to 0$ as $n \to \infty$ from part (c). Therefore H(0) holds. It is now shown that H(m) implies that H($m+1$) holds. There exists a basis $y_1, ..., y_p$ in E^{**}, the dual space of E^*, such that $\langle y_i, \eta_k\rangle = 0$ when $i \neq k$ and $\langle y_i, \eta_k\rangle = 1$ when $i = k$. The set of vectors $y_1, ..., y_p$ is a basis of E since $E = E^{**}$. Arbitrarily choose a $y \in H_{j-m-1}$ such that $y = \sum_{i=1}^{j-m-1} \delta_i y_i + c_1 y_{j-m}$ where $c_1 > \delta_{j-m}$. At this choice of y we see that $h(y) = +\infty$ and

$$\begin{aligned}
h_n(y) &= a_n + \sum_{i=1}^{j-m+1} b_{i,n}\left(\langle y, \eta_i\rangle - \delta_i\right) \\
&= a_n + b_{(j-m),n}\left(\langle y, \eta_{j-m}\rangle - \delta_{j-m}\right) \\
&\to +\infty
\end{aligned}$$

as $n \to \infty$. Therefore $b_{(j-m),n} \to +\infty$ as $n \to \infty$. Now arbitrarily choose $y = \sum_{i=1}^{j-m-1} \delta_i y_i + c_1 y_{j-m} + c_2 y_{j-m+1}$ where c_1 is defined as before and $c_2 < \delta_{j-m+1}$. At this choice of y we see that $h(y) = +\infty$ and

$$\begin{aligned}
h_n(y) &= a_n + \sum_{i=1}^{j-m+1} b_{i,n}\left(\langle y, \eta_i\rangle - \delta_i\right) \\
&= a_n + b_{(j-m),n}\left(\langle y, \eta_{j-m}\rangle - \delta_{j-m}\right. \\
&\qquad \left. + \frac{b_{(j-m+1),n}}{b_{(j-m),n}}\left(\langle y, \eta_{j-m+1}\rangle - \delta_{j-m+1}\right)\right) \qquad (41) \\
&= a_n + b_{(j-m),n}\left(c_1 - \delta_{j-m} - \frac{b_{(j-m+1),n}}{b_{(j-m),n}}\left(c_2 - \delta_{j-m+1}\right)\right) \\
&\to +\infty
\end{aligned}$$

as $n \to \infty$. It follows from (41) that

$$\left(c_1 - \delta_{j-m} - \frac{b_{(j-m+1),n}}{b_{(j-m),n}}\left(c_2 - \delta_{j-m+1}\right)\right) \geq 0$$

for sufficiently large n. This implies that

$$\frac{b_{(j-m+1),n}}{b_{(j-m),n}} \leq \frac{c_1 - \delta_{j-m}}{\delta_{j-m-1} - c_2}$$

for sufficiently large n. From the arbitrariness of the constants c_1 and c_2 and (41), we can conclude that $b_{(j-m+1),n}/b_{(j-m),n} \to 0$ as $n \to \infty$. Therefore H($m+1$) holds and this completes one direction of the proof.

We now assume that conditions (a) through (d) and the h_n takes the form in (13). Let $\lim_{n\to\infty} \sum_{i=j+1}^{p} b_{i,n}\eta_i = \theta^*$ and $\lim_{n\to\infty} a_n = a$. Cases (a) through (d) then imply that

$$h_n(y) \to \begin{cases} -\infty, & y \in C_i^- \\ \langle y, \theta^*\rangle + a, & y \in H_j \\ +\infty, & y \in C_i^+ \end{cases} \qquad (42)$$

for all $i = 1, ..., j$ where the right hand side of (42) of Eck and Geyer [2018] is a generalized affine function in its Theorem 5 representation. This completes the proof. □

3 Proofs of MGF and moment convergence results

We first prove Theorem 8.

Proof. Suppose φ_X is an MGF, hence finite on a neighborhood W of zero. Fix $t \in E^*$. Then by (17) $\varphi_{\langle X,t\rangle}(s)$ is finite whenever $st \in W$. Continuity of scalar multiplication means there exists an $\varepsilon > 0$ such that $st \in W$ whenever $|s| < \varepsilon$. That proves one direction.

Conversely, suppose $\varphi_{\langle X,t\rangle}$ is an MGF for each $t \in E^*$. Suppose $v_1, \ldots, v_d$ is a basis for E and $w_1, \ldots, w_d$ is the dual basis for E^* that satisfies (18). Then there exists $\varepsilon > 0$ such that $\varphi_{\langle X,w_i\rangle}$ is finite on $[-\varepsilon, \varepsilon]$ for each i.

We can write each $t \in E^*$ as a linear combination of basis vectors

$$t = \sum_{i=1}^d a_i w_i,$$

where the a_i are scalars that are unique [Halmos, 1974, Theorem 1 of Section 15]. Applying (18) we get

$$\langle v_j, t\rangle = a_j,$$

so

$$t = \sum_{i=1}^d \langle v_i, t\rangle w_i,$$

and

$$\langle X, t\rangle = \sum_{i=1}^d \langle v_i, t\rangle \langle X, w_i\rangle.$$

Suppose

$$|\langle v_i, t\rangle| \le \varepsilon, \qquad i = 1, \ldots, d$$

(the set of all such t is a neighborhood of 0 in E^*). Let sign denote the sign function, which takes values -1, 0, and $+1$ as its argument is negative, zero, or positive, and write

$$s_i = \operatorname{sign}(\langle v_i, t\rangle), \qquad i = 1, \ldots, d.$$

Then we can write $\langle X, t\rangle$ as a convex combination

$$\langle X, t\rangle = \sum_{i=1}^d \frac{\langle v_i, t\rangle}{s_i \varepsilon} \cdot s_i \varepsilon \langle X, w_i\rangle + \left(1 - \sum_{i=1}^d \frac{\langle v_i, t\rangle}{s_i \varepsilon}\right) \cdot \langle X, 0\rangle.$$

So, by convexity of the exponential function,

$$\varphi_X(t) \le \sum_{i=1}^d \frac{\langle v_i, t\rangle}{s_i \varepsilon} \varphi_{\langle X,w_i\rangle}(s_i \varepsilon) + \left(1 - \sum_{i=1}^d \frac{\langle v_i, t\rangle}{s_i \varepsilon}\right) < \infty.$$

That proves the other direction. □

We now prove Theorem 9.

Proof. The one-dimensional case of this theorem is proved in Billingsley [2012]. We only need to show the general case follows by Cramér-Wold. It follows from the assumption that $\varphi_{\langle X_n,t\rangle}$ converges on a neighborhood W of zero for each $t \in E^*$. Then (19) follows from the one-dimensional case of this theorem and the Cramér-Wold theorem. And this implies

$$\langle X_n, t\rangle \xrightarrow{d} \langle X, t\rangle, \qquad t \in E^*.$$

By the one-dimensional case of this theorem, this implies $\langle X, t\rangle$ has an MGF for each t, and then Theorem 8 implies X has an MGF φ_X. By the one-dimensional case of this theorem, $\varphi_{\langle X_n,t\rangle}$ converges pointwise to $\varphi_{\langle X,t\rangle}$. So by (17), φ_{X_n} converges pointwise to φ_X. □

We now prove Theorem 10.

Proof. From Theorem 9, we have that $\langle X_n, t_i\rangle \xrightarrow{d} \langle X, , t_i\rangle$. Continuity of the exponential function implies that $e^{\langle X_n, t_i\rangle} \xrightarrow{d} e^{\langle X, t_i\rangle}$. Now, pick an $\varepsilon > 0$ such that both $\varepsilon \sum_{i=1}^k t_i \in W$ and $\varepsilon \sum_{i=1}^k u_i \in W$ where $u_1 = -t_1$ and $u_i = t_i$ for all $i > 1$. This construction gives

$$e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle} \xrightarrow{d} e^{\langle X, \varepsilon \sum_{i=1}^k t_i\rangle} \tag{43}$$

and

$$\mathrm{E}\left(e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle}\right) \xrightarrow{d} \mathrm{E}\left(e^{\langle X, \varepsilon \sum_{i=1}^k t_i\rangle}\right). \tag{44}$$

Equations (43) and (44) imply that $e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle}$ is uniformly integrable by [Billingsley, 1999, Theorem 3.6]. A similar argument shows that $e^{\langle X_n, \varepsilon \sum_{i=1}^k u_i\rangle}$ is uniformly integrable. We now bound $|\varepsilon^k \prod_{i=1}^k \langle X_n, t_i\rangle|$ to show uniform integrability of $\prod_{i=1}^k \langle X_n, t_i\rangle$. Define

$$A_n = \{X_n : \prod_{i=1}^k \langle X_n, t_i\rangle \geq 0\}.$$

and let I_A be the indicator function. We have,

$$\begin{aligned}\varepsilon^k \prod_{i=1}^k \langle X_n, t_i\rangle &\leq \prod_{i=1}^k \langle X_n, \varepsilon t_i\rangle I_{A_n} \\ &\leq e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle} I_{A_n} \\ &\leq e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle}\end{aligned}$$

and

$$\begin{aligned}-\varepsilon^k \prod_{i=1}^k \langle X_n, t_i\rangle &= \prod_{i=1}^k \langle X_n, \varepsilon u_i\rangle \\ &\leq \prod_{i=1}^k \langle X_n, \varepsilon u_i\rangle I_{A_n^c} \\ &\leq e^{\langle X_n, \varepsilon \sum_{i=1}^k u_i\rangle} I_{A_n^c} \\ &\leq e^{\langle X_n, \varepsilon \sum_{i=1}^k u_i\rangle}.\end{aligned}$$

Therefore

$$|\varepsilon^k \prod_{i=1}^k \langle X_n, t_i\rangle| \leq e^{\langle X_n, \varepsilon \sum_{i=1}^k t_i\rangle} + e^{\langle X_n, \varepsilon \sum_{i=1}^k u_i\rangle}$$

The sum of uniformly integrable is uniformly integrable. This implies that $|\varepsilon^k \prod_{i=1}^k \langle X_n, t_i\rangle|$ is uniformly integrable. Scaling of uniformly integrable is also uniformly integrable, which implies $\prod_{i=1}^k \langle X_n, t_i\rangle$ is uniformly integrable. Our result follows from [Billingsley, 1999, Theorem 3.5] and this completes the proof. □

4 Counterexample

This section provides a counterexample to the non-theorem which is Theorem 6 with its conditions removed (that is, the assertion that cumulant generating function convergence always occurs). It shows that some conditions like those the theorem requires are needed.

4.1 Model

Suppose we have a two-dimensional exponential family with generating measure λ concentrated on the set

$$S = \{(0,0), (0,1)\} \cup \{\, (1,n) : n \in \mathbb{N}\,\},$$

where $\mathbb{N}$ is the set of natural numbers 0, 1, 2, ... And suppose λ takes values

$$\lambda(x) = \frac{1}{x_2!}, \qquad x \in S.$$

The Laplace transform of λ is the function of θ given by

$$1 + e^{\theta_2} + e^{\theta_1} \sum_{x_2=0}^{\infty} \frac{e^{x_2\theta_2}}{x_2!} = 1 + e^{\theta_2} + e^{\theta_1} e^{e^{\theta_2}}$$

and the cumulant function (log Laplace transform) is

$$c(\theta) = \log\left[1 + e^{\theta_2} + e^{\theta_1 + e^{\theta_2}}\right] \tag{45}$$

4.2 Maximum Likelihood

Suppose the observed value of the canonical statistic is $x = (0,1)$.

From Chapter 2 of Geyer [1990] we know that we can find the MLE in the completion of the family by taking limits first in the direction $\eta_1 = (-1,0)$ (which is a direction of recession) and second in the direction $\eta_2 = (0,1)$ (which is a direction of recession for the limiting conditional model resulting from the first limit). Thus the MLE in the completion is the completely degenerate distribution concentrated at the observed data. The Theorem 5 (in the main article) characterization of the corresponding generalized affine function evaluated at data x, $h(x)$, yields set $C_1^- = \{(1,n) : n \in \mathbb{N}\}$ and thus $D_1 = \{(1,n), n\mathbb{N},\ n \geq 1\}$. Clearly $\lambda(D_1) > 0$, and we have

$$\sup_{\theta\in\Theta}\sup_{y\in D_1} e^{\langle y,\theta\rangle - c_{D_1}(\theta)} \geq \sup_{y\in\mathbb{N}} e^{\langle (1,y),(0,1)\rangle - c_{D_1}((0,1))} = \infty.$$

Therefore the bound condition of Theorem 6 in the main article is violated. We now show that CGF convergence along a likelihood maximizing sequence fails for t in a neighborhood of 0.

4.3 Log Likelihood

The log likelihood is

$$\begin{aligned} l(\theta) &= x_1\theta_1 + x_2\theta_2 - c(\theta) \\ &= \theta_2 - c(\theta) \\ &= -\log\left[e^{-\theta_2} + 1 + e^{\theta_1 - \theta_2 + e^{\theta_2}}\right] \end{aligned}$$

4.4 Likelihood Maximizing Sequences

Because the MLE in the completion is completely degenerate and because $\lambda(x) = 1$, the log likelihood must go to $\log(1) = 0$ along any likelihood maximizing sequence.

We know from Lemma 1 in the main article that any likelihood maximizing sequence θ_n must have

(i) $\theta_{1,n} \to -\infty$,

(ii) $\theta_{2,n} \to +\infty$,

(iii) $|\theta_{2,n}/\theta_{1,n}| \to 0$,

but now we see that, in this example, it must also have

(iv) $\theta_{1,n} - \theta_{2,n} + e^{\theta_{2,n}} \to -\infty$.

Thus we see that Lemma 1 doesn't tell us everything about likelihood maximizing sequences (it may do under the conditions of Brown).

4.5 Cumulant Generating Function Convergence

The cumulant generating function for canonical parameter value θ is

$$k_\theta(t) = c(\theta + t) - c(\theta).$$

Thus along a likelihood maximizing sequence we have

$$\begin{aligned} k_{\theta_n}(t) &= \log \left[\frac{1 + e^{\theta_2 + t_2} + e^{\theta_1 + t_1 + e^{\theta_2 + t_2}}}{1 + e^{\theta_2} + e^{\theta_1 + e^{\theta_2}}} \right] \\ &= \log \left[\frac{e^{-\theta_2} + e^{t_2} + e^{\theta_1 - \theta_2 + t_1 + e^{\theta_2 + t_2}}}{e^{-\theta_2} + 1 + e^{\theta_1 - \theta_2 + e^{\theta_2}}} \right] \end{aligned}$$

We know the denominator of the fraction converges to one along any likelihood maximizing sequence. The cumulant generating function of the distribution concentrated at x is the log of

$$e^{0 \cdot t_1 + 1 \cdot t_2}$$

so

$$k_{\text{limit}}(t) = t_2$$

Thus we see that to get the correct limit we need a different condition

(v) $\theta_{1,n} - \theta_{2,n} + e^{\theta_{2,n} + t_2} \to -\infty$.

Since (i) through (iv) do not imply (v) unless $t_2 \le 0$, we cannot guarantee cumulant generating function convergence on a neighborhood of zero.

Suppose, for concreteness

$$\theta_n = (-n, \log(n)) \tag{46}$$

so the sequence in (v) becomes

$$-n - \log(n) + ne^{t_2}$$

Hence condition (v) is not satisfied unless $t_2 \le 0$, but conditions (i) through (iv) are satisfied.

4.6 Nonconvergence of First Moments

First moments (of the canonical statistic) are given by differentiating the cumulant function (45)

$$\nabla c(\theta) = \begin{pmatrix} \frac{e^{\theta_1 + e^{\theta_2}}}{1 + e^{\theta_2} + e^{\theta_1 + e^{\theta_2}}} \\ \frac{e^{\theta_2} + e^{\theta_1 + e^{\theta_2} + \theta_2}}{1 + e^{\theta_2} + e^{\theta_1 + e^{\theta_2}}} \end{pmatrix}$$

The first moment of the LCM, which is concentrated at x is just x. So the necessary and sufficient condition for convergence of first moments to the first moments of the LCM is

$$\begin{aligned} \frac{e^{\theta_{1,n} + e^{\theta_{2,n}}}}{1 + e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}}}} &\to 0 \\ \frac{e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}} + \theta_{2,n}}}{1 + e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}}}} &\to 1 \end{aligned}$$

For the specific likelihood maximizing sequence (46) we have

$$\begin{aligned} \frac{e^{\theta_{1,n} + e^{\theta_{2,n}}}}{1 + e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}}}} &= \frac{e^{-n+n}}{1 + n + e^{-n+n}} \\ &= \frac{1}{2 + n} \\ \frac{e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}} + \theta_{2,n}}}{1 + e^{\theta_{2,n}} + e^{\theta_{1,n} + e^{\theta_{2,n}}}} &= \frac{n + e^{-n+n+\log(n)}}{1 + n + e^{-n+n}} \\ &= \frac{2n}{2 + n} \end{aligned}$$

The first converges to 0 as it must for CGF convergence. The second converges to 2, but it must converge to 1 for CGF convergence. So we do not get convergence of first moments for this model and this likelihood maximizing sequence, hence cannot have CGF convergence.

4.7 Nonconvergence of Second Moments

Non-convergence of first moments already makes CGF convergence impossible, but since our main interest in CGF convergence is convergence of second moments, which are components of the Fisher information matrix, we compute them too.

For c given by (45) and θ_n given by (46)

$$\nabla^2 c(\theta_n) = \frac{1}{(2+n)^2} \begin{pmatrix} 1+n & n^2 \\ n^2 & n(4+n^2) \end{pmatrix} \to \begin{pmatrix} 0 & 1 \\ 1 & \infty \end{pmatrix}$$

The variance-covariance matrix for the LCM is the zero matrix (the variance-covariance matrix of a completely degenerate distribution). Hence we do not get convergence of Fisher information for this example.

5 R

- The version of R used to make this document is 3.6.1.
- The version of the `knitr` package used to make this document is 1.24.
- The version of the `glmdr` package used to make this document is 0.1.
- The version of the `rcdd` package used to make this document is 1.2.2.
- The version of the `numDeriv` package used to make this document is 2016.8.1.1.
- The version of the `alabama` package used to make this document is 2015.3.1.
- The version of the `Matrix` package used to make this document is 1.2.17.

Load these packages.

```r
library(glmdr)
library(rcdd)

## If you want correct answers, use rational arithmetic.
## See the Warnings sections added to help pages for
##    functions that do computational geometry.

library(numDeriv)
library(alabama)
library(Matrix)
```

Set random number generator seeds. We only use randomness in tests. This assures the tests always come out the same.

```r
set.seed(42)
```

Figure out some stuff about the machine (only works on Linux).

```
if (Sys.info()["sysname"] == "Linux") {
    foo <- scan("/proc/cpuinfo", what = character(0), sep = "\n")
    bar <- grep("^model name", foo, value = TRUE)
    bar <- unique(bar)
    baz <- sub("^model name\\t: ", "", bar)
    cat("computer name:", system("hostname", intern = TRUE), "\n")
    cat("computer model:", baz, "\n")
}
```

Clean R global environment.

```
rm(list = ls())
```

6 Complete separation example of Agresti

6.1 Data

Agresti [2013, Section 6.5.1] introduces the notion of complete separation with the following simple logistic regression example.

```
x <- seq(10, 90, 10)
x <- x[x != 50]
x

## [1] 10 20 30 40 60 70 80 90

y <- as.numeric(x > 50)
y

## [1] 0 0 0 0 1 1 1 1
```

These data are included in the `glmdr` package.

```
data(complete)
all.equal(complete, as.data.frame(cbind(x,y)))

## [1] TRUE
```

6.2 The MLE in the LCM

We fit these data using R function `glmdr` in the `glmdr` R package [Geyer and Eck, 2016].

```
gout <- glmdr(y ~ x, family = "binomial", data = complete)
summary(gout)

##
## MLE exists in Barndorff-Nielsen completion
## it is completely degenerate
## the MLE says the response actually observed is the only
## possible value that could ever be observed
```

In this example the LCM is completely degenerate and has no identifiable parameters.

6.2.1 Linearity

The function `glmdr` determines which data points belong to the support of the LCM. We already know that the support of the LCM is empty.

```
gout$linearity

## [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

The support of the LCM (the linearity) are the data points with responses that are not conditioned to be their observed value.

6.3 One-sided confidence intervals for mean value parameters

We now provide one-sided confidence intervals for mean value parameters whose MLE is on the boundary. We calculate these intervals using a new method not previously published, but whose concept is found in Geyer [2009] in the penultimate paragraph of Section 3.16.2 and further discussed in Sections 3.6.1–3.6.3 of Geyer [2016]. Sections 6.3.1 and 6.3.3 contain a description of our method in the context of this example.

R function `inference` in R package `glmdr` computes these one-sided confidence intervals for mean value parameters.

```
system.time(mus.CI <- inference(gout))

##    user  system elapsed
##   3.589   0.006   3.642

mus.CI

##   intercept  x y     lower     upper
## 1         1 10 0 0.0000000 0.2852500
## 2         1 20 0 0.0000000 0.3940359
## 3         1 30 0 0.0000000 0.5708292
## 4         1 40 0 0.0000000 0.9499881
## 5         1 60 1 0.0500257 1.0000000
## 6         1 70 1 0.4291708 1.0000000
## 7         1 80 1 0.6059641 1.0000000
## 8         1 90 1 0.7147500 1.0000000
```

Note that for some components of the mean value parameter vector the lower or upper bound of our confidence interval is close to the quick and dirty limit (Section 6.3.2 below). In particular, for $x = 40$ the upper bound is close to 0.95 and for $x = 60$ the lower bound is close to 0.05. But for other components of the response vector there are much more restrictive bounds.

Now make a plot of these intervals.

```
bounds.lower.p <- mus.CI$lower
bounds.upper.p <- mus.CI$upper
par(mar = c(4, 4, 0, 0) + 0.1)
plot(x, y, axes = FALSE, type = "n",
    xlab = expression(x), ylab = expression(mu(x)))
segments(x, bounds.lower.p, x, bounds.upper.p, lwd = 2)
box()
axis(side = 1)
axis(side = 2)
points(x, y, pch = 21, bg = "white")
```

Our Figure 1 is Figure 2 in Eck and Geyer (submitted). It agrees with Figure 9 in [Geyer, 2016], except R function `inference` only predicts at the observed predictor values, whereas the calculations in [Geyer, 2016] predict for all predictor values. Also [Geyer, 2016] used a method that only works for two-parameter models.

6.3.1 Theory for logistic regression

Let β denote the vector of submodel canonical parameters (what R calls "coefficients" in the GLM case). Let $l(\beta)$ denote the log likelihood. Let $\hat{\beta}$ denote an MLE in the LCM. Since the LCM in this example is the degenerate model with no identifiable parameters, every vector is an MLE. We take the zero vector to be $\hat{\beta}$. Let I denote the index set of the components of the response vector on which we condition the original model (OM) to get the limiting conditional model (LCM). See Geyer [2009, Section 3.4] for definitions. In this example I is the whole index vector for the model. In the examples in Sections 8 and 9 below I will not be the whole index vector. Let Y_I and y_I denote the corresponding components of the response vector considered as a random vector and as an observed value, respectively. Then endpoints for a $100(1-\alpha)\%$ confidence interval for a scalar parameter $g(\beta)$ are

$$\min_{\substack{\gamma \in \Gamma_{\text{lim}} \\ \text{pr}_{\hat{\beta}+\gamma}(Y_I = y_I) \geq \alpha}} g(\hat{\beta}+\gamma) \qquad \text{and} \qquad \max_{\substack{\gamma \in \Gamma_{\text{lim}} \\ \text{pr}_{\hat{\beta}+\gamma}(Y_I = y_I) \geq \alpha}} g(\hat{\beta}+\gamma) \tag{47}$$

where Γ_{lim} is the constancy space of the LCM [Geyer, 2009, Section 3.16.2]. In this example Γ_{lim} is the whole parameter space. In the examples in Sections 8 and 9 it won't be. In these expressions pr denotes probability with respect to the OM not the LCM.

The formula (47) is valid when one of the endpoints is at the end of the range of the parameter $g(\beta)$. Otherwise we can use conventional two-sided intervals.

In this example, we will be doing confidence intervals for mean value parameters for components of the response vector for which the MLE in the LCM is on the boundary, either zero or one. Since we always know whether the observed data is at the upper or lower or lower end of its range, we always know we only have to calculate the other end of the confidence interval.

Let $p = \text{logit}^{-1}(\theta)$ denote the mean value parameter vector (here logit^{-1} operates componentwise). Then the probabilities in (47) are

$$\text{pr}_\beta(Y_I = y_I) = \prod_{i \in I} p_i^{y_i}(1-p_i)^{n_i - y_i}$$

where the n_i are the binomial sample sizes. In this example we have $n_i = 1$ for all i, but in Section 8 below we have $n_i = 2$ for all i.

We could take the confidence interval problem to be

$$\begin{array}{ll} \text{maximize} & p_k \\ \text{subject to} & \displaystyle\prod_{i \in I} p_i^{y_i}(1-p_i)^{n_i - y_i} \geq \alpha \end{array} \tag{48}$$

where p is taken to be the function of γ described above. And this can be done for any $k \in I$.

But the problem will be more computationally stable if we state it as

$$\begin{array}{ll} \text{maximize} & \theta_k \\ \text{subject to} & \displaystyle\sum_{i \in I} \bigl[y_i \log(p_i) + (n_i - y_i)\log(1-p_i)\bigr] \geq \log(\alpha) \end{array} \tag{49}$$

Since $\theta_k = \text{logit}(p_k)$ is a monotone transformation and log is a monotone transformation, the two problems are equivalent (a solution for one is also a solution for the other).

We maximize canonical rather than mean value parameters to avoid extreme inexactness of computer arithmetic in calculating mean value parameters near zero and one. We take logs in the constraint for the same reasons we take logs of likelihoods.

Because optimizers expect to optimize over $\mathbb{R}^q$ for some q, let N be a matrix whose columns are a basis for Γ_{lim}. In this example Γ_{lim} is the whole parameter space so N can be the identity matrix. In other

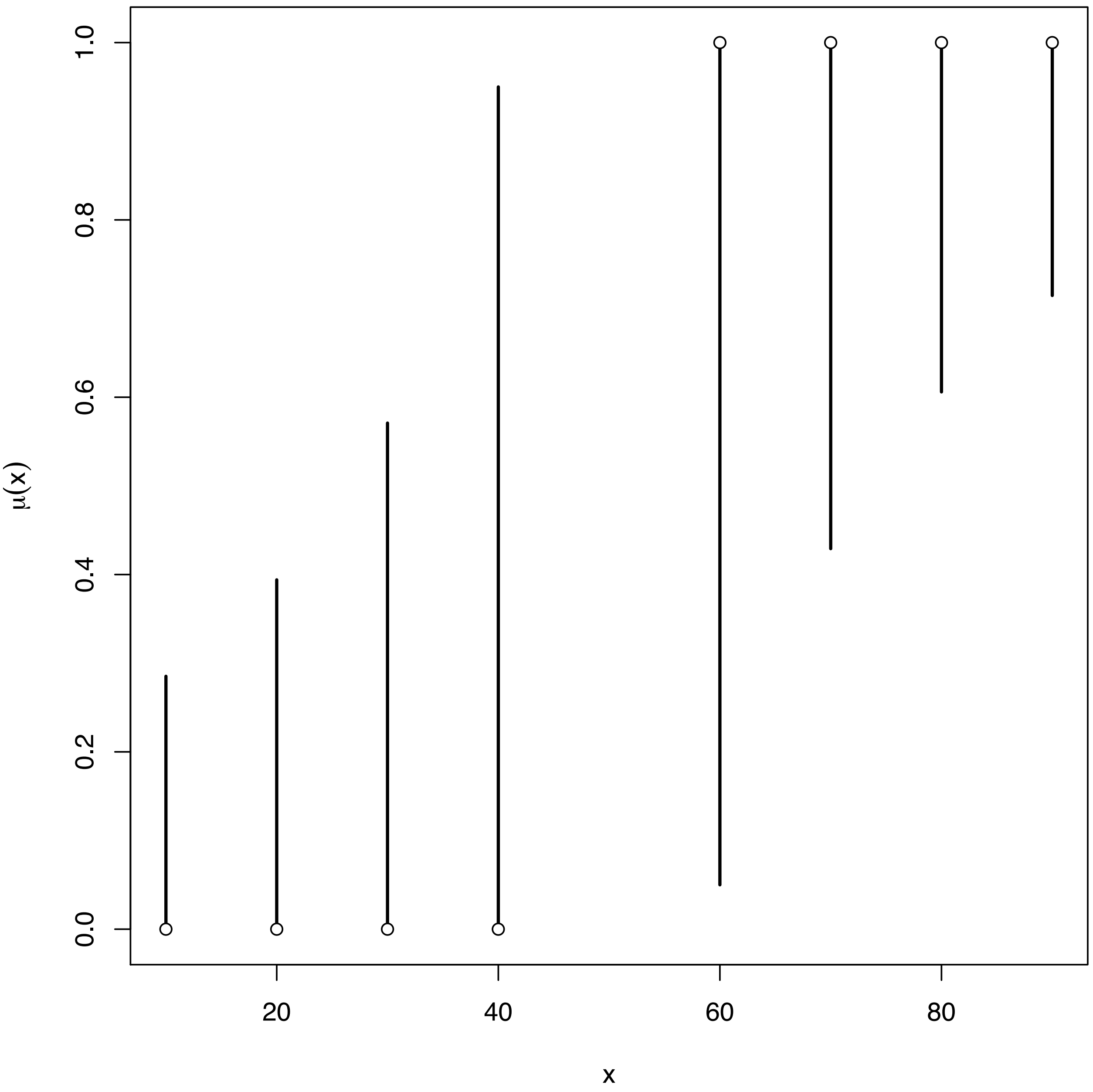

Figure 1: One-sided 95% confidence intervals for mean value parameters. Bars are the intervals. Vertical axis is the probability of observing response value one when the predictor value is x. Solid dots are the observed data.

problems we take it to be a matrix whose columns are null eigenvectors of the Fisher information matrix. Then every $\gamma \in \Gamma_{\text{lim}}$ can be written as $\gamma = N\xi$ for some $\xi \in \mathbb{R}^q$, where q is the column dimension of N and the dimension of Γ_{lim}.

To an optimizer (the `inference` function in the `glmdr` package will use the R function `auglag` in CRAN package `alabama`) problem (49) has the abstract form

$$\begin{aligned} \text{minimize} \quad & f(\xi) \\ \text{subject to} \quad & g(\xi) \geq 0 \end{aligned} \tag{50}$$

and the optimization works better if derivatives of f and g are provided. Because R function `auglag` only does minimization, the objective function must be the negation of what we have in (49). That is

$$\begin{aligned} f(\xi) &= -\theta_k \\ \frac{\partial f(\xi)}{\partial \xi_j} &= -o_{kj} \\ g(\xi) &= \sum_{i \in I} \bigl[y_i \log(p_i) + (n_i - y_i) \log(1 - p_i) \bigr] - \log(\alpha) \\ \frac{\partial g(\xi)}{\partial \xi_j} &= \sum_{i \in I} (y_i - n_i p_i) o_{ij} \end{aligned}$$

where o_{ij} are the components of $O = MN$.

6.3.2 Quick and dirty intervals

As a sanity check and as a quick and dirty conservative (perhaps very conservative) confidence interval, we note that since all the p_i are between zero and one we must have

$$\begin{aligned} p_k^{n_k} &\geq \alpha, \qquad y_k = n_k \\ (1 - p_k)^{n_k} &\geq \alpha, \qquad y_k = 0 \end{aligned}$$

or

$$\begin{aligned} \alpha^{1/n_k} \leq p_k &\leq 1, \qquad y_k = n_k \\ 0 \leq p_k &\leq 1 - \alpha^{1/n_k}, \qquad y_k = 0 \end{aligned}$$

For $\alpha = 0.05$ and $n_k = 1$ we have

$$\begin{aligned} \alpha^{1/n_k} &= 0.05 \\ 1 - \alpha^{1/n_k} &= 0.95 \end{aligned}$$

In this example, no upper bound for a one-sided 95% confidence interval for the mean value parameter for a cell for which the MLE in the LCM is zero can be larger than than 0.95 and no lower bound for the analogous confidence interval for which the MLE in the LCM is one can be smaller than 0.05.

6.3.3 Careful coding for logistic regression

The math of logistic regression is very tricky for the computer. Unless arranged very carefully, the computer may overflow or underflow causing loss of all significant figures.

First there is the map from canonical to mean value parameters

$$p = \operatorname{logit}^{-1}(\theta)$$

where this inverse logit function operates componentwise

$$\begin{aligned} p_i &= \frac{e^{\theta_i}}{1 + e^{\theta_i}} = \frac{1}{1 + e^{-\theta_i}} \\ 1 - p_i &= \frac{1}{1 + e^{\theta_i}} = \frac{e^{-\theta_i}}{1 + e^{-\theta_i}} \end{aligned}$$

for all i.

We should always choose one of these formulas for which we know we can have neither overflow, nor catastrophic cancellation. We always calculate $1 - p_i$ using the second line, we never calculate p_i and subtract from one because this results in catastrophic cancellation when p_i is near one. If θ_i is large positive, we choose a formula that has $e^{-\theta_i}$ in it, as that cannot overflow. If θ_i is large negative, we choose a formula that has e^{θ_i} in it, as that cannot overflow. If θ_i is not large, it doesn't matter which we choose.

We also never use the log function to take logarithms as this can cause horrible inaccuracy when the argument is near one. R has a function `log1p` that calculates $\log(1 + x)$ accurately for small values of x. We want to use that.

$$\begin{aligned} \log(p_i) &= \theta_i - \log(1 + e^{\theta_i}) &&= -\log(1 + e^{-\theta_i}) \\ \log(1 - p_i) &= -\log(1 + e^{\theta_i}) &&= -\theta_i - \log(1 + e^{-\theta_i}) \end{aligned}$$

so we calculate

$$\begin{aligned} \log(p_i) &= \theta_i - \operatorname{log1p}(e^{\theta_i}) &&= -\operatorname{log1p}(e^{-\theta_i}) \\ \log(1 - p_i) &= -\operatorname{log1p}(e^{\theta_i}) &&= -\theta_i - \operatorname{log1p}(e^{-\theta_i}) \end{aligned}$$

With this care, we have a hope of getting approximately correct answers out of the computer.

6.4 Support of the submodel canonical statistic

In this section we duplicate Figure 2 of [Geyer, 2016], which is Figure 1 in the main article [Eck and Geyer, 2018]. The methods of this section take computer time proportional to the size of the sample space. Hence they can only be used on toy problems and are useless for practical applications. They do help in understanding the Barndorff-Nielsen completion.

For GLM the (submodel) canonical statistic is $M^T Y$, where M is the model matrix and y is the response vector [Geyer, 2009, Section 3.9]. There are 2^n possible values where n is the dimension of the response vector because each component of y can be either zero or one. The following code makes all of those vectors.

```r
yy <- NULL
n <- length(y)
for (i in 1:n) {
    j <- 2^(i - 1)
    k <- 2^n / j / 2
    yy <- cbind(rep(rep(0:1, each = j), times = k), yy)
}
```

But there are not so many distinct values of the submodel canonical statistic.

```r
m <- cbind(1, x)
mtyy <- t(m) %*% t(yy)
t1 <- mtyy[1, ]
t2 <- mtyy[2, ]
t1.obs <- sum(y)
t2.obs <- sum(x * y)
```

Figure 2 shows these possible values of the submodel canonical statistic. As already stated, it is Figure 1 in the main article [Eck and Geyer, 2018].

6.5 Linearity by computational geometry

For comparison of computer times and to see that our new methods give correct results, we redo some of our analysis above using the methods of [Geyer, 2009]. In this section we find the linearity of the tangent cone [Geyer, 2009, Sections 3.6 through 3.12].

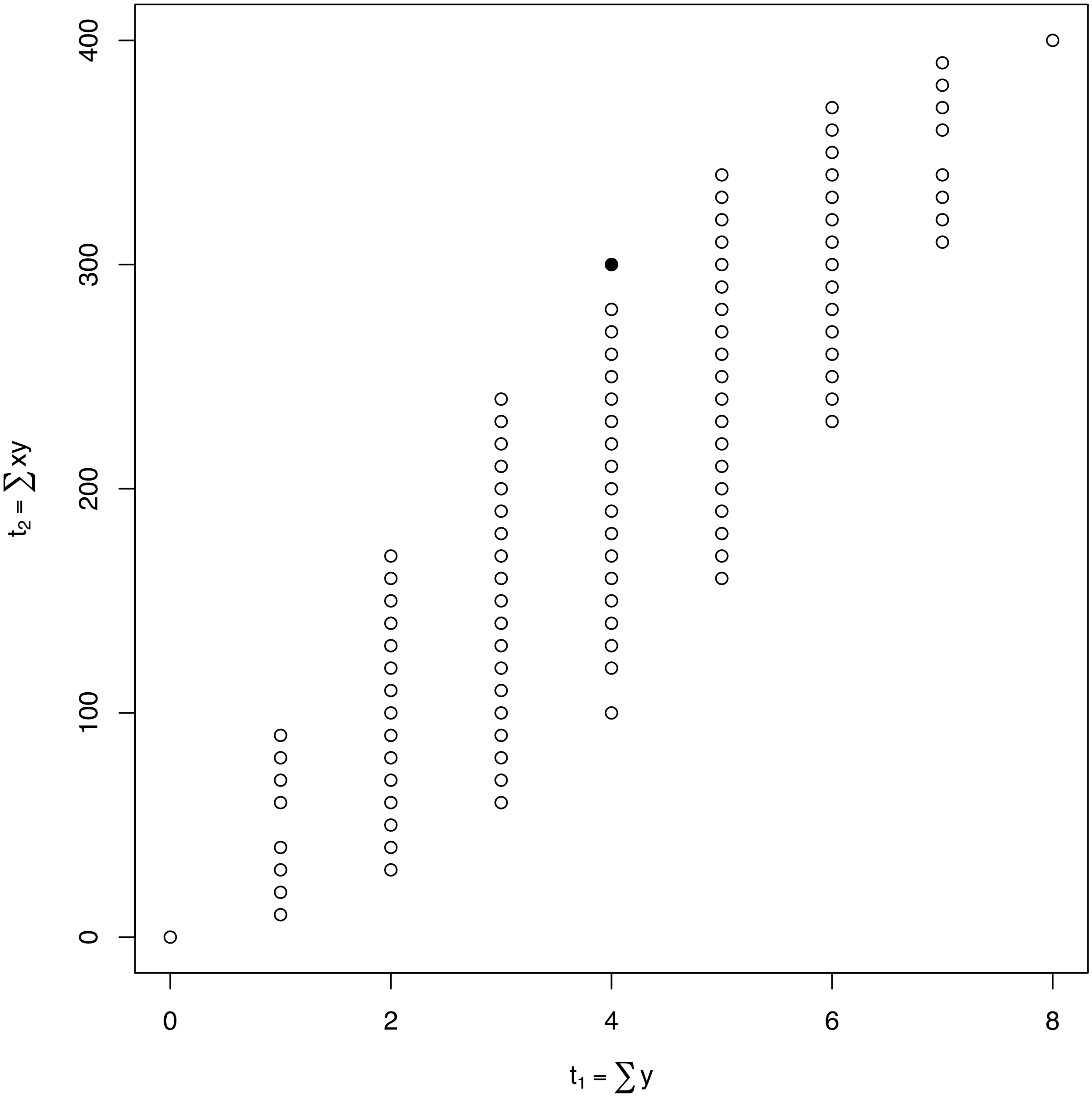

Figure 2: Possible values of the submodel canonical statistic vector $M^T y$ for these data. Solid dot is the observed value of the submodel canonical statistic vector.

The computer code in this section can be found in a technical report [Geyer, 2008, Section 3.12] cited in [Geyer, 2009] and also in the lecture notes [Geyer, 2016].

```
## calling glm to:
## 1) get model matrix and
## 2) illustrate that it outputs a warning message when fit to this data
out <- glm(y ~ x, family = "binomial", data = complete, x = TRUE)

## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred

tanv <- modmat <- out$x
tanv[y == 1, ] <- (-tanv[y == 1, ])
vrep <- makeV(rays = tanv)
system.time(lout <- linearity(d2q(vrep), rep = "V"))

##    user  system elapsed
##   0.012   0.000   0.012

lout

## integer(0)
```

R object `lout` is the set of indices of components of the response vector that do not have a degenerate distribution in the LCM. In this example it has length zero indicating that the LCM is completely degenerate. This agrees with our analysis in Section 6.2 above.

Unlike the analysis using our new methods in Section 6.2 above, the analysis in this section using R package `rcdd` is guaranteed to be correct — as valid as any mathematical proof — because the functions in that package can use infinite precision rational arithmetic (R function `linearity` is doing so in the code chunk above).

Although the analysis in this section takes a trivial amount of computer time on this toy problem, it does not scale. It takes days of computer time on the example in Section 11 below. Our new methods do scale.

6.6 Generic direction of recession

The main theoretical tool of Geyer [2009] is the notion of a *generic direction of recession* (GDOR) [Geyer, 2009, Sections 3.3 through 3.13]. But our new methods of calculation do not need to refer to it. (We only need to get the correct linearity using eigenvalues and eigenvectors of the Fisher information matrix.)

The code chunk below comes from the technical report [Geyer, 2008, Section 4.1] and also from the lecture notes [Geyer, 2016].

```
p <- ncol(tanv)
hrep <- cbind(0, 0, -tanv, -1)
hrep <- rbind(hrep, c(0, 1, rep(0, p), -1))
objv <- c(rep(0, p), 1)
pout <- lpcdd(d2q(hrep), d2q(objv), minimize = FALSE)
names(pout)

## [1] "solution.type"   "primal.solution" "dual.solution"
## [4] "optimal.value"

pout$solution.type

## [1] "Optimal"

gdor <- pout$primal.solution[1:p]
gdor
```

```
## [1] "-5"   "1/10"

pout$optimal.value

## [1] "1"
```

The code chunk above is not general. It assumes the linearity is trivial, as in the particular example we are working on. Other examples below will have more general code. This agrees with the calculation in [Geyer, 2016, Section 3.3].

The fact that a GDOR exists shows that our calculation of the linearity was correct (no matter how it was done). That a GDOR exists is shown by `pout$solution.type` being `"Optimal"` by `pout$optimal.value` being strictly positive.

Clean R global environment.

```
rm(list = ls())
```

7 Complete separation example of Geyer

This is the example in Section 2.2 of [Geyer, 2009]. Its behavior is very similar to that of the preceding example. The only difference is that this does quadratic logistic regression instead of linear logistic regression.

7.1 Data

Data

```
x <- 1:30
y <- c(rep(0, 12), rep(1, 11), rep(0, 7))
```

These data are included in the `glmdr` package.

```
data(quadratic)
all.equal(quadratic, as.data.frame(cbind(x,y)))

## [1] TRUE
```

7.2 The MLE in the LCM

The LCM is completely degenerate and has no identifiable parameters. We fit these data using R function `glmdr`.

```
gout <- glmdr(y ~ x + I(x^2), family = "binomial", data = quadratic)
summary(gout)

##
## MLE exists in Barndorff-Nielsen completion
## it is completely degenerate
## the MLE says the response actually observed is the only
## possible value that could ever be observed
```

7.3 One-sided confidence intervals for mean value parameters

We now provide one-sided confidence intervals for mean value parameters whose MLE is on the boundary.

```
mus.CI <- inference(gout)
mus.CI

##    intercept  x I.x.2. y      lower        upper
## 1           1  1      1 0 0.00000000 6.563500e-12
## 2           1  2      4 0 0.00000000 1.915859e-10
## 3           1  3      9 0 0.00000000 4.570606e-09
## 4           1  4     16 0 0.00000000 8.919127e-08
## 5           1  5     25 0 0.00000000 1.425473e-06
## 6           1  6     36 0 0.00000000 1.869697e-05
## 7           1  7     49 0 0.00000000 2.019500e-04
## 8           1  8     64 0 0.00000000 1.806200e-03
## 9           1  9     81 0 0.00000000 1.345421e-02
## 10          1 10    100 0 0.00000000 8.242264e-02
## 11          1 11    121 0 0.00000000 3.741233e-01
## 12          1 12    144 0 0.00000000 9.481161e-01
## 13          1 13    169 1 0.05330860 1.000000e+00
## 14          1 14    196 1 0.65501216 1.000000e+00
## 15          1 15    225 1 0.86723176 1.000000e+00
## 16          1 16    256 1 0.92920573 1.000000e+00
## 17          1 17    289 1 0.95066373 1.000000e+00
## 18          1 18    324 1 0.95616871 1.000000e+00
## 19          1 19    361 1 0.95066368 1.000000e+00
## 20          1 20    400 1 0.92920575 1.000000e+00
## 21          1 21    441 1 0.86723190 1.000000e+00
## 22          1 22    484 1 0.65501207 1.000000e+00
## 23          1 23    529 1 0.05350799 1.000000e+00
## 24          1 24    576 0 0.00000000 9.479931e-01
## 25          1 25    625 0 0.00000000 3.741229e-01
## 26          1 26    676 0 0.00000000 8.242262e-02
## 27          1 27    729 0 0.00000000 1.345423e-02
## 28          1 28    784 0 0.00000000 1.806203e-03
## 29          1 29    841 0 0.00000000 2.019507e-04
## 30          1 30    900 0 0.00000000 1.869705e-05
```

Note that for some cells of the mean value parameter vector the lower or upper bound of our confidence interval is close to the quick and dirty limit. (Section 6.3.2 above). In particular, for $x = 12$ and $x = 24$ the upper bound is close to 0.95 and for $x = 13$ and $x = 23$ the lower bound is close to 0.05. But for other components of the response vector there are much more restrictive bounds.

Now make a plot of these intervals.

```
bounds.lower.p <- mus.CI$lower
bounds.upper.p <- mus.CI$upper
par(mar = c(4, 4, 0, 0) + 0.1)
plot(x, y, axes = FALSE, type = "n",
    xlab = expression(x), ylab = expression(mu(x)))
segments(x, bounds.lower.p, x, bounds.upper.p, lwd = 2)
box()
axis(side = 1)
axis(side = 2)
points(x, y, pch = 21, bg = "white")
```

Our Figure 3 agrees with Figure 2 in [Geyer, 2009], which was done by methods that are much more messy and made obsolete by the methods presented here.

7.4 Linearity by computational geometry

Calculate linearity using R package `rcdd` like in Section 6.5 above.

```
## calling glm to:
## 1) get model matrix and
## 2) illustrate that it outputs a warning message when fit to this data
out <- glm(y ~ x + I(x^2), family = "binomial",
  data = quadratic, x = TRUE)

## Warning:  glm.fit:  algorithm did not converge
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred

tanv <- modmat <- out$x
tanv[y == 1, ] <- (-tanv[y == 1, ])
vrep <- makeV(rays = tanv)
lout <- linearity(d2q(vrep), rep = "V")
lout

## integer(0)
```

So this agrees with our analysis in Section 7.2 above.

7.5 Generic direction of recession

Calculate a GDOR using R package `rcdd` like in Section 6.6 above.

```
p <- ncol(tanv)
hrep <- cbind(0, 0, -tanv, -1)
hrep <- rbind(hrep, c(0, 1, rep(0, p), -1))
objv <- c(rep(0, p), 1)
pout <- lpcdd(d2q(hrep), d2q(objv), minimize = FALSE)
names(pout)

## [1] "solution.type"   "primal.solution" "dual.solution"
## [4] "optimal.value"

pout$solution.type

## [1] "Optimal"

gdor <- pout$primal.solution[1:p]
gdor

## [1] "-587/11" "72/11"   "-2/11"

pout$optimal.value

## [1] "1"
```

This agrees with the GDOR found in the technical report [Geyer, 2008, Section 4.1] that is supplementary material for [Geyer, 2009].

Clean R global environment.

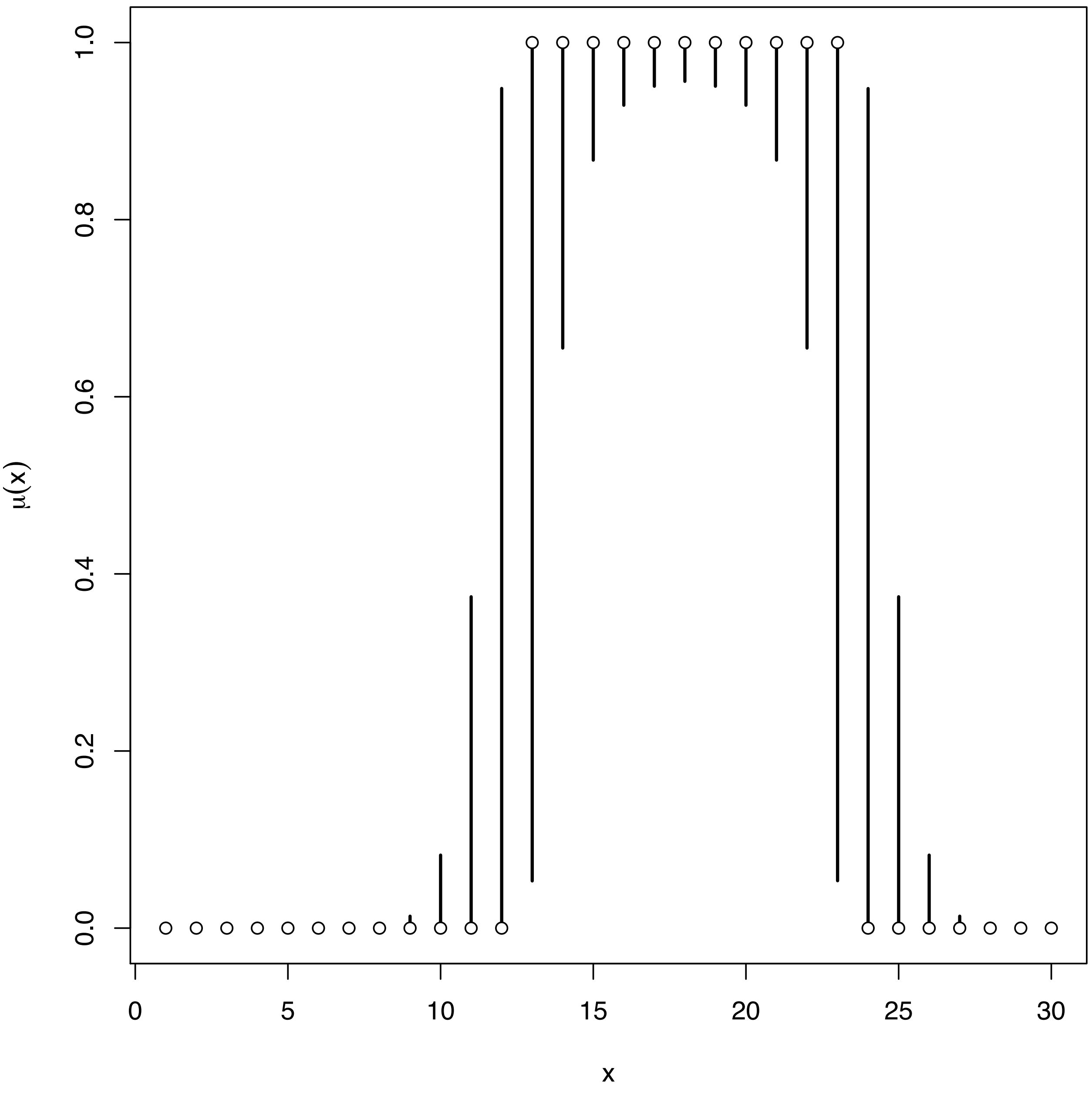

Figure 3: One-sided 95% confidence intervals for mean value parameters. Bars are the intervals. Vertical axis is the probability of observing response value one when the predictor value is x. Solid dots are the observed data.

```
rm(list = ls())
```

8 Sports standings example of Geyer

This is the example in Section 2.4 of Geyer [2009]. Its behavior is different from any of the preceding examples, because the LCM is not completely degenerate and also because the binomial sample size is two for all components of the response vector.

8.1 Data

Data

```
team.names <- c("ants", "beetles", "cows", "dogs",
    "egrets", "foxes", "gerbils", "hogs")
data <- matrix(c(NA, 2, 2, 2, 2, 2, 2, 2, 0, NA,
    1, 2, 2, 2, 2, 2, 0, 1, NA, 2, 1, 2, 2, 2, 0,
    0, 0, NA, 1, 1, 2, 2, 0, 0, 1, 1, NA, 1, 2, 2,
    0, 0, 0, 1, 1, NA, 2, 2, 0, 0, 0, 0, 0, 0, NA,
    1, 0, 0, 0, 0, 0, 0, 1, NA), byrow = TRUE, nrow = 8)
dimnames(data) <- list(team.names, team.names)
print(data)

##         ants beetles cows dogs egrets foxes gerbils hogs
## ants      NA       2    2    2      2     2       2    2
## beetles    0      NA    1    2      2     2       2    2
## cows       0       1   NA    2      1     2       2    2
## dogs       0       0    0   NA      1     1       2    2
## egrets     0       0    1    1     NA     1       2    2
## foxes      0       0    0    1      1    NA       2    2
## gerbils    0       0    0    0      0     0      NA    1
## hogs       0       0    0    0      0     0       1   NA
```

We model these data with Bradley-Terry model. We code this differently from the technical report [Geyer, 2008] accompanying Geyer [2009].

First we format the data the way R function `glm` likes (in a data.frame).

```
wins <- data[upper.tri(data)]
team.plus <- row(data)[upper.tri(data)]
team.minus <- col(data)[upper.tri(data)]
modmat <- matrix(0, length(wins), nrow(data))
for (i in 1:ncol(modmat)) {
    modmat[team.plus == i, i] <- 1
    modmat[team.minus == i, i] <- (-1)
}
losses <- 2 - wins
resp <- cbind(wins, losses)

colnames(modmat) <- team.names
sportsdata <- cbind(modmat, wins, losses)
sportsdata <- as.data.frame(sportsdata)
```

These data are included in the `glmdr` package.

```
data(sports)
all.equal(sports, sportsdata)

## [1] TRUE
```

8.2 Fitting the Model

We first fit the model using the R function `glmdr`.

```
gout <- glmdr(cbind(wins, losses) ~ 0 + .,
  family = "binomial", data = sports)
summary(gout)

##
## MLE exists in Barndorff-Nielsen completion
## it is conditional on components of the response
## corresponding to object$linearity == FALSE being
## conditioned on their observed values
##
## GLM summary for limiting conditional model
##
##
## Call:
## stats::glm(formula = cbind(wins, losses) ~ 0 + ., family = "binomial",
##     data = sports, subset = c("3", "5", "6", "8", "9", "10",
##     "12", "13", "14", "15", "28"), x = TRUE, y = TRUE)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1692  -0.1970   0.3941   0.5038   0.6153
##
## Coefficients: (3 not defined because of singularities)
##           Estimate Std. Error z value Pr(>|z|)
## ants            NA         NA      NA       NA
## beetles  3.024e+00  1.487e+00   2.034   0.0419 *
## cows     2.310e+00  1.328e+00   1.740   0.0819 .
## dogs    -5.189e-17  1.080e+00   0.000   1.0000
## egrets   5.609e-01  1.078e+00   0.520   0.6029
## foxes           NA         NA      NA       NA
## gerbils  0.000e+00  1.414e+00   0.000   1.0000
## hogs            NA         NA      NA       NA
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 13.863  on 11  degrees of freedom
## Residual deviance:  3.391  on  6  degrees of freedom
## AIC: 21.709
##
## Number of Fisher Scoring iterations: 5
```

8.3 Linearity

As explained in Section 6.3 of the main article [Eck and Geyer, 2018], the components of the response vector that are random in the LCM are those for which the null space projected to canonical parameter space of the saturated model have corresponding zeros. These components are those for which the `linearity` of the object returned by R function `glmdr` is `true`

```
gout$linearity

##     1     2     3     4     5     6     7     8     9    10
## FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##    11    12    13    14    15    16    17    18    19    20
## FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
##    21    22    23    24    25    26    27    28
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

8.4 Labels

We now want to make some confidence intervals, but first we make some short labels for components of the response vector.

```
foo <- sports[ , ! (colnames(sports) %in% c("wins", "losses"))]
teams <- colnames(foo)
winner <- apply(foo == 1, 1, function(x) teams[x])
loser <- apply(foo == -1, 1, function(x) teams[x])
label <- paste(winner, "beat", loser)
head(label)

## [1] "ants beat beetles" "ants beat cows"
## [3] "beetles beat cows" "ants beat dogs"
## [5] "beetles beat dogs" "cows beat dogs"
```

8.5 Confidence Intervals

We now want to fit confidence intervals. These come in two kinds. First, there are confidence intervals for means of components of the response vector that are in the linearity. These are the usual sort of confidence intervals for GLM, based on asymptotics, and produced by the `glm` method of the R generic function `predict`. Second, there are confidence intervals for means of components of the response vector that are not in the linearity. These are non-asymptotic intervals, described in Section 4.4, and produced by R function `inference` in R package `glmdr`. These latter intervals are necessarily one-sided because the MLE mean value parameter estimates for these components of the response vector are on the boundary of the range of possible values.

8.5.1 Two-Sided Intervals

We get estimated means and standard errors as follows.

```
preds <- predict(gout$lcm, type = "response", se.fit = TRUE)
preds.tab <- cbind(preds$fit, preds$se.fit)
colnames(preds.tab) <- c("fit", "se")
rownames(preds.tab) <- label[gout$linearity]
round(preds.tab, 3)
```

```
##                       fit    se
## beetles beat cows   0.671 0.274
## beetles beat dogs   0.954 0.066
## cows beat dogs      0.910 0.109
## beetles beat egrets 0.921 0.103
## cows beat egrets    0.852 0.159
## dogs beat egrets    0.363 0.249
## beetles beat foxes  0.954 0.066
## cows beat foxes     0.910 0.109
## dogs beat foxes     0.500 0.270
## egrets beat foxes   0.637 0.249
## gerbils beat hogs   0.500 0.354
```

And turn this into 95% confidence intervals as follows.

```
ci.tab <- apply(preds.tab, 1, function(x) x[1] + c(-1,1) * qnorm(0.975) * x[2])
ci.tab <- t(ci.tab)
colnames(ci.tab) <- c("lwr", "upr")
round(ci.tab, 3)

##                         lwr   upr
## beetles beat cows    0.134 1.208
## beetles beat dogs    0.825 1.082
## cows beat dogs       0.696 1.123
## beetles beat egrets  0.720 1.123
## cows beat egrets     0.541 1.163
## dogs beat egrets    -0.126 0.852
## beetles beat foxes   0.825 1.082
## cows beat foxes      0.696 1.123
## dogs beat foxes     -0.029 1.029
## egrets beat foxes    0.148 1.126
## gerbils beat hogs   -0.193 1.193
```

As always, there is no reason why Wald confidence intervals cannot go outside the boundaries of the parameter space, as some of these intervals do. As noted in the discussion of [Geyer, 2009], the sample sizes here are by no means “large”. The last confidence interval (gerbils versus hogs) is based on exactly two games (these teams played two games and each won one, no other games are relevant to this inference). So for these data, the confidence intervals produced in this section are of questionable validity.

8.5.2 One-Sided Intervals

We get one-sided intervals as follows. These numbers agree with Table 5 in [Geyer, 2009], which was done by methods that are much more messy and made obsolete by the methods presented here.

```
ci.tab.too <- inference(gout)
rownames(ci.tab.too) <- label[! gout$linearity]
round(ci.tab.too, 3)

##                       lower upper
## ants beat beetles    0.893     2
## ants beat cows       1.245     2
## ants beat dogs       1.886     2
## ants beat egrets     1.809     2
## ants beat foxes      1.886     2
## ants beat gerbils    1.993     2
## beetles beat gerbils 1.970     2
## cows beat gerbils    1.940     2
## dogs beat gerbils    1.526     2
## egrets beat gerbils  1.699     2
## foxes beat gerbils   1.526     2
## ants beat hogs       1.993     2
## beetles beat hogs    1.970     2
## cows beat hogs       1.940     2
## dogs beat hogs       1.526     2
## egrets beat hogs     1.699     2
## foxes beat hogs      1.526     2
```

With $n = 2$ (each team plays each other team twice), quick and dirty confidence intervals go from zero to

$$1 - \alpha^{1/2} = 0.7763932$$

(when $\alpha = 0.05$) or from

$$\alpha^{1/2} = 0.2236068$$

to one (again when $\alpha = 0.05$). None of the careful intervals calculated above are anywhere near as wide as the quick and dirty intervals.

8.6 Linearity by computational geometry

Calculate linearity using R package `rcdd` like in Section 6.5 above. We follow Section 5 of J. [2009], except that seems to have some errors, which we correct here.

```
tanv <- modmat
tanv[losses == 0, ] <- (- tanv[losses == 0, ])
vrep <- cbind(0, 0, tanv)
vrep[wins > 0 & losses > 0, 1] <- 1
lout <- linearity(d2q(vrep), rep = "V")
```

This result only includes the additional components found to be in the linearity (in addition to the ones already known). So we have to add the others to get the correct linearity.

```
linearity.too <- seq(along = wins) %in% lout
linearity.too[wins > 0 & losses > 0] <- TRUE
identical(as.vector(gout$linearity), linearity.too)

## [1] TRUE
```

So this agrees with our analysis in Section 8.3 above.

8.7 Generic direction of recession

Calculate a GDOR using R package `rcdd` like in Section 6.6 above. More specifically, we follow Section 6 of [Geyer, 2008], so we necessarily agree with the GDOR given in Table 4 of [Geyer, 2009].

```
p <- ncol(tanv)
hrep <- cbind(0, 0, -tanv, 0)
hrep[! gout$linearity, ncol(hrep)] <- (-1)
hrep[gout$linearity, 1] <- 1
hrep <- rbind(hrep, c(0, 1, rep(0, p), -1))
```

```
objv <- c(rep(0, p), 1)
pout <- lpcdd(hrep, objv, minimize = FALSE)
gdor <- pout$primal.solution[1:p]
names(gdor) <- team.names
print(gdor)

##    ants beetles    cows    dogs  egrets   foxes gerbils
##       2       1       1       1       1       1       0
##    hogs
##       0
```

Clean R global environment.

```
rm(list = ls())
```

9 Quasi-complete separation example of Agresti

9.1 Data

Agresti [2013, Section 6.5.1] introduces the notion of quasi-complete separation with the following example, which adds two data points to the data for his other example (Section 6 above).

```
x <- seq(10, 90, 10)
x <- x[x != 50]
y <- as.numeric(x > 50)
x <- c(x, 50, 50)
y <- c(y, 0, 1)
```

These data are included in the glmdr package.

```
data(quasi)
all.equal(quasi, data.frame(x, y))

## [1] TRUE
```

9.2 Maximizing the OM likelihood

Again, we fit these data using R function glmdr.

```
gout <- glmdr(y ~ x, family = "binomial", data = quasi)
summary(gout)

##
## MLE exists in Barndorff-Nielsen completion
## it is conditional on components of the response
## corresponding to object$linearity == FALSE being
## conditioned on their observed values
##
## GLM summary for limiting conditional model
##
##
## Call:
```

```
## stats::glm(formula = y ~ x, family = "binomial", data = quasi,
##     subset = c("9", "10"), x = TRUE, y = TRUE)
##
## Deviance Residuals:
##      9      10
## -1.177   1.177
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 4.710e-16  1.414e+00       0        1
## x                  NA         NA      NA       NA
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.7726  on 1  degrees of freedom
## Residual deviance: 2.7726  on 1  degrees of freedom
## AIC: 4.7726
##
## Number of Fisher Scoring iterations: 2
```

9.3 Linearity

We extract the linearity from the `glmdr` function call.

```
gout$linearity

##     1     2     3     4     5     6     7     8     9    10
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
```

9.4 One-sided confidence intervals for mean value parameters

We now provide one-sided confidence intervals for mean value parameters whose MLE is on the boundary.

```
inference(gout)

##       lower      upper
## 1 0.0000000 0.07082447
## 2 0.0000000 0.14043775
## 3 0.0000000 0.27199887
## 4 0.0000000 0.51720648
## 5 0.4827935 1.00000000
## 6 0.7280012 1.00000000
## 7 0.8595623 1.00000000
## 8 0.9291755 1.00000000
```

Note that for some components of the mean value parameter vector the lower or upper bound of our confidence interval is not close to the quick and dirty limit (Section 6.3.2 above) like they were in the case of complete separation.

9.5 Two-sided confidence intervals for mean value parameters

As in the preceding example, confidence intervals for means of components of the response vector in the linearity are given by R generic function `predict`.

```
preds <- predict(gout$lcm, type = "response", se.fit = TRUE)
preds.tab <- cbind(preds$fit - qnorm(0.975) * preds$se.fit,
    preds$fit + qnorm(0.975) * preds$se.fit)
colnames(preds.tab) <- c("lower", "upper")
round(preds.tab, 3)

##     lower upper
## 9  -0.193 1.193
## 10 -0.193 1.193
```

As we saw with the sports data, these asymptotic confidence intervals are not good for toy data. Again we effectively have $n = 2$ for these intervals, so they are exactly the same as the one for gerbils versus hogs in the sports data.

Clean R global environment.

```
rm(list = ls())
```

10 Categorical data analysis example of Geyer

10.1 Data

This is the example in Section 2.3 of Geyer [2009]. Its behavior is very similar to the quasi-complete separation example of Agresti in Section 9 above.

```
foo <- "https://conservancy.umn.edu/bitstream/handle/11299/197369/catrec.txt"
bar <- sub("^.*/", "", foo)
if (! file.exists(bar))
    download.file(foo, bar)
dat <- read.table(bar, header = TRUE)
dim(dat)

## [1] 128   8

names(dat)

## [1] "v1" "v2" "v3" "v4" "v5" "v6" "v7" "y"
```

These data are included in the `glmdr` package.

```
data(catrec)
all.equal(catrec, dat)

## [1] TRUE
```

10.2 Fitting the Model

Following Geyer [2009] we assume Poisson rather than multinomial sampling. These two sampling schemes have the same MLE, even when the MLE is in the Barndorff-Nielsen completion [Agresti, 2013, Section 8.6.7; Geyer, 2009, Section 3.17] but Poisson sampling is the easiest to fit. We can use R function `glm` if the MLE exists in the conventional sense, and R function `glmdr` otherwise.

```
gout <- glmdr(y ~ (v1 + v2 + v3 + v4 + v5 + v6 + v7)^3,
    family = "poisson", data = dat)
summary(gout)
```

```
##
## MLE exists in Barndorff-Nielsen completion
## it is conditional on components of the response
## corresponding to object$linearity == FALSE being
## conditioned on their observed values
##
## GLM summary for limiting conditional model
##
##
## Call:
## stats::glm(formula = y ~ (v1 + v2 + v3 + v4 + v5 + v6 + v7)^3,
##     family = "poisson", data = dat, subset = c("2", "3", "4",
##     "5", "6", "7", "8", "10", "11", "12", "13", "14", "15", "16",
##     "17", "18", "19", "21", "22", "23", "24", "25", "26", "27",
##     "29", "30", "31", "32", "34", "35", "36", "37", "38", "39",
##     "40", "42", "43", "44", "45", "46", "47", "48", "49", "50",
##     "51", "53", "54", "55", "56", "57", "58", "59", "61", "62",
##     "63", "64", "66", "67", "68", "69", "70", "71", "72", "74",
##     "75", "76", "77", "78", "79", "80", "81", "82", "83", "85",
##     "86", "87", "88", "89", "90", "91", "93", "94", "95", "96",
##     "98", "99", "100", "101", "102", "103", "104", "106", "107",
##     "108", "109", "110", "111", "112", "113", "114", "115", "117",
##     "118", "119", "120", "121", "122", "123", "125", "126", "127",
##     "128"), x = TRUE, y = TRUE)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.63571  -0.30009  -0.02353   0.27258   1.42540
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.150481   0.585423   3.673 0.000239 ***
## v1           0.069795   0.587067   0.119 0.905364
## v2          -0.524215   0.513583  -1.021 0.307396
## v3           0.052966   0.551965   0.096 0.923552
## v4          -0.709525   0.580147  -1.223 0.221326
## v5           0.243002   0.548686   0.443 0.657853
## v6          -1.163256   0.563668  -2.064 0.039044 *
## v7          -0.990704   0.597335  -1.659 0.097208 .
## v1:v2        0.384345   0.543024   0.708 0.479079
## v1:v3       -0.630375   0.570151  -1.106 0.268888
## v1:v4        0.008801   0.511458   0.017 0.986271
## v1:v5       -1.022805   0.570440  -1.793 0.072971 .
## v1:v6        0.540164   0.493879   1.094 0.274079
## v1:v7        0.097178   0.536628   0.181 0.856297
## v2:v3        0.602411   0.437371   1.377 0.168405
## v2:v4        0.748226   0.486811   1.537 0.124295
## v2:v5       -0.068926   0.428100  -0.161 0.872090
## v2:v6        0.297165   0.487409   0.610 0.542071
## v2:v7        0.274198   0.508369   0.539 0.589634
## v3:v4       -0.124465   0.541056  -0.230 0.818060
## v3:v5       -0.439354   0.468418  -0.938 0.348268
## v3:v6        0.024399   0.530220   0.046 0.963296
## v3:v7       -0.104400   0.556960  -0.187 0.851310
## v4:v5       -0.169421   0.521323  -0.325 0.745194
## v4:v6        0.756513   0.474213   1.595 0.110644
## v4:v7        0.780671   0.500911   1.559 0.119114
## v5:v6        1.245629   0.510770   2.439 0.014739 *
## v5:v7       -0.262620   0.523125  -0.502 0.615652
## v6:v7        0.697014   0.489957   1.423 0.154852
## v1:v2:v3    -0.349902   0.483330  -0.724 0.469102
## v1:v2:v4     0.101569   0.389778   0.261 0.794416
## v1:v2:v5     0.655208   0.493737   1.327 0.184496
## v1:v2:v6    -0.329286   0.390979  -0.842 0.399670
## v1:v2:v7    -0.520368   0.393042  -1.324 0.185520
## v1:v3:v4     0.353292   0.406623   0.869 0.384932
## v1:v3:v5     0.638711   0.484979   1.317 0.187843
## v1:v3:v6     0.352694   0.402715   0.876 0.381143
## v1:v3:v7    -0.001586   0.413554  -0.004 0.996941
## v1:v4:v5     0.664745   0.400212   1.661 0.096717 .
## v1:v4:v6    -0.463885   0.368214  -1.260 0.207732
## v1:v4:v7    -0.342583   0.372009  -0.921 0.357103
## v1:v5:v6     0.044968   0.399958   0.112 0.910481
## v1:v5:v7     0.447641   0.404364   1.107 0.268283
## v1:v6:v7     0.218868   0.371499   0.589 0.555763
## v2:v3:v4    -0.325914   0.404392  -0.806 0.420280
## v2:v3:v5           NA         NA      NA       NA
## v2:v3:v6    -0.247853   0.405621  -0.611 0.541168
## v2:v3:v7     0.028322   0.414520   0.068 0.945527
## v2:v4:v5     0.004655   0.394418   0.012 0.990583
## v2:v4:v6    -0.111152   0.373713  -0.297 0.766141
## v2:v4:v7    -0.148061   0.376692  -0.393 0.694279
## v2:v5:v6    -0.766051   0.394925  -1.940 0.052412 .
## v2:v5:v7     0.075213   0.399004   0.189 0.850482
## v2:v6:v7     0.460826   0.381109   1.209 0.226597
## v3:v4:v5    -0.063494   0.423318  -0.150 0.880771
## v3:v4:v6     0.357746   0.366298   0.977 0.328741
## v3:v4:v7    -0.106368   0.371567  -0.286 0.774672
## v3:v5:v6    -0.234816   0.422424  -0.556 0.578295
## v3:v5:v7     0.804923   0.423843   1.899 0.057550 .
## v3:v6:v7    -0.659090   0.371085  -1.776 0.075714 .
## v4:v5:v6    -0.427957   0.375755  -1.139 0.254734
## v4:v5:v7     0.125167   0.377356   0.332 0.740119
## v4:v6:v7     0.014192   0.370131   0.038 0.969413
## v5:v6:v7    -0.811516   0.377098  -2.152 0.031397 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 156.215  on 111  degrees of freedom
## Residual deviance:  31.291  on  49  degrees of freedom
## AIC: 526.46
##
## Number of Fisher Scoring iterations: 5
```

This agrees with the result in the technical report [Geyer, 2008, Section 4.2.1] accompanying Geyer [2009].

10.3 Linearity

We then find the linearity as in preceding sections.

```
linearity <- gout$linearity
catrec[!linearity, ]

##     v1 v2 v3 v4 v5 v6 v7 y
## 1    0  0  0  0  0  0  0 0
## 9    0  0  0  1  0  0  0 0
## 20   1  1  0  0  1  0  0 0
## 28   1  1  0  1  1  0  0 0
## 33   0  0  0  0  0  1  0 0
## 41   0  0  0  1  0  1  0 0
## 52   1  1  0  0  1  1  0 0
## 60   1  1  0  1  1  1  0 0
## 65   0  0  0  0  0  0  1 0
## 73   0  0  0  1  0  0  1 0
## 84   1  1  0  0  1  0  1 0
## 92   1  1  0  1  1  0  1 0
## 97   0  0  0  0  0  1  1 0
## 105  0  0  0  1  0  1  1 0
## 116  1  1  0  0  1  1  1 0
## 124  1  1  0  1  1  1  1 0
```

This agrees with (part of) Table 2 in [Geyer, 2009].

10.4 One-sided confidence intervals: Poisson sampling

We now provide one-sided confidence intervals for mean value parameters whose MLE is on the boundary as done before.

```
system.time(tab <- inference(gout))

##    user  system elapsed
##   0.564   0.012   0.578

upper <- tab$upper
cbind(catrec[!linearity, ], upper)

##     v1 v2 v3 v4 v5 v6 v7 y      upper
## 1    0  0  0  0  0  0  0 0 0.28630976
## 9    0  0  0  1  0  0  0 0 0.14082947
## 20   1  1  0  0  1  0  0 0 0.21996699
## 28   1  1  0  1  1  0  0 0 0.42095570
## 33   0  0  0  0  0  1  0 0 0.08946242
## 41   0  0  0  1  0  1  0 0 0.09376644
## 52   1  1  0  0  1  1  0 0 0.19302341
## 60   1  1  0  1  1  1  0 0 0.28869770
## 65   0  0  0  0  0  0  1 0 0.10631113
## 73   0  0  0  1  0  0  1 0 0.11415034
## 84   1  1  0  0  1  0  1 0 0.09128766
## 92   1  1  0  1  1  0  1 0 0.26461098
## 97   0  0  0  0  0  1  1 0 0.06669488
## 105  0  0  0  1  0  1  1 0 0.15477613
## 116  1  1  0  0  1  1  1 0 0.14096916
## 124  1  1  0  1  1  1  1 0 0.32392016
```

This agrees with Table 2 in [Geyer, 2009].

10.4.1 Theory

Here we modify Section 6.3.1 above, changing what needs to be changed for Poisson regression rather than logistic regression.

As in Section 6.3.1 above, let β denote the vector of submodel canonical parameters, let $l(\beta)$ denote the log likelihood, and let $\hat{\beta}$ denote an MLE in the LCM. We will use the vector `gout$lcm$coefficients` with `NA` values replaced by zeros. Let I denote the index set of the components of the response vector on which we condition the OM to get the LCM (the indices of components of `linearity` that are `FALSE`), and let Y_I and y_I denote the corresponding components of the response vector considered as a random vector and as an observed value, respectively. Then endpoints for a $100(1-\alpha)\%$ confidence interval for a scalar parameter $g(\beta)$ are given by (47), when it does give a one-sided interval.

Since the only boundary of the mean value parameter space of the Poisson distribution is zero, in this section, we will be doing confidence intervals for mean value parameters for cells of the contingency table where the MLE in the LCM is zero. And we know the min is zero, so we only have to calculate the max.

In (47) pr denotes probability with respect to the OM not the LCM. As always in categorical data analysis, we have different possible sampling models: Poisson, multinomial, and product multinomial. So we get different intervals depending on which sampling model we use. In this section we are assuming Poisson.

Let M denote the model matrix. Let $\theta = M\beta$ denote the saturated model canonical parameter (usually called "linear predictor" in GLM theory).

Let $\mu = \exp(\theta)$ denote the mean value parameter (here exp operates componentwise like the R function of the same name does), then

$$\text{pr}_\beta(Y_I = y_I) = \text{pr}_\beta(Y_I = 0) = \exp\left(-\sum_{i \in I} \mu_i\right)$$

We could take the confidence interval problem to be

$$\begin{aligned} &\text{maximize} && \mu_k \\ &\text{subject to} && \exp\left(-\sum_{i \in I} \mu_i\right) \geq \alpha \end{aligned} \tag{51}$$

where μ is taken to be the function of γ described above. And this can be done for any $k \in I$.

But the problem will be more computationally stable if we state it as

$$\begin{aligned} &\text{maximize} && \theta_k \\ &\text{subject to} && -\sum_{i \in I} \mu_i \geq \log(\alpha) \end{aligned} \tag{52}$$

Since $\mu_k = \exp(\theta_k)$ is a monotone transformation and log is a monotone transformation, the two problems are equivalent (a solution for one is also a solution for the other).

We maximize canonical rather than mean value parameters to avoid extreme inexactness of computer arithmetic in calculating mean value parameters near zero. We take logs in the constraint for the same reasons we take logs of likelihoods.

Because optimizers expect to optimize over $\mathbb{R}^q$ for some q, let N be a matrix whose columns are a basis for Γ_{lim} (the R matrix `nulls` calculated above, for example). Then every $\gamma \in \Gamma_{\text{lim}}$ can be written as $\gamma = N\xi$ for some $\xi \in \mathbb{R}^q$, where q is the column dimension of N and the dimension of Γ_{lim}.

To an optimizer (we will use R function `auglag` in CRAN package `alabama`) problem (52) has the abstract form (50) and the optimization works better if derivatives of f and g are provided. Because R function `auglag` only does minimization, the objective function must be the negation of what we have in (52). That is

$$
\begin{aligned}
f(\xi) &= -\theta_k \\
\frac{\partial f(\xi)}{\partial \xi_j} &= -o_{kj} \\
g(\xi) &= -\sum_{i \in I} \mu_i - \log(\alpha) \\
\frac{\partial g(\xi)}{\partial \xi_j} &= -\sum_{i \in I} \mu_i o_{ij}
\end{aligned}
$$

where o_{ij} are the components of $O = MN$.

10.4.2 Quick and dirty intervals

As a sanity check and as a quick and dirty conservative (perhaps very conservative) confidence interval, we note that since all the μ_i are nonnegative, the only way the constraint in (51) can be satisfied is if $\mu_k \le -\log(\alpha)$. For $\alpha = 0.05$ this upper bound is

```
- log(0.05)

## [1] 2.995732
```

No upper bound for a one-sided 95% confidence interval for the mean value parameter for a cell for which the MLE in the LCM is zero can be larger than that.

10.5 One-sided confidence intervals: Multinomial sampling

10.5.1 Theory

We use the same notation as in Section 10.4.1 above, except where modified here.

Since the only boundary of the mean value parameter space of the multinomial distribution is where one or more components of the state vector are zero, we will be doing confidence intervals for mean value parameters for cells of the contingency table where the MLE in the LCM is zero. And we know the min is zero, so we only have to calculate the max. (If the MLE in the LCM for mean value parameter vector had all but one component equal to zero, so the other was equal to one, then we could make one-sided intervals for all components. But that is not a situation we see in any of our examples, and we will leave that as an exercise for the reader.)

For multinomial sampling, contingency table cell probabilities are defined by

$$
p_i = \frac{e^{\theta_i}}{\sum_{j \in J} e^{\theta_j}}, \qquad i \in J, \tag{53}
$$

where J is the index set for the whole table.

Now

$$
\text{pr}_\beta(Y_I = y_I) = \text{pr}_\beta(Y_I = 0) = \left(\sum_{i \in J \setminus I} p_i \right)^n
$$

where

$$
n = \sum_{j \in J} y_j
$$

is the multinomial sample size, where I is the index set of the cells that have mean value zero for the MLE in the LCM.

So we could take the confidence interval problem to be

$$\begin{aligned} &\text{maximize} \quad p_k \\ &\text{subject to} \quad \left(\sum_{i \in J \setminus I} p_i \right)^n \geq \alpha \end{aligned} \tag{54}$$

where p is taken to be the function of γ described above. And this can be done for any $k \in I$.

Unlike preceding theory for this problem, we cannot take θ_k to be the objective function because p_k is not a function of θ_k only (much less a monotone function of it). Consequently, to obtain computational stability, we will take logs of both equations obtaining

$$\begin{aligned} &\text{maximize} \quad \theta_k - \log\left(\sum_{j \in J} e^{\theta_j} \right) \\ &\text{subject to} \quad n \log\left(\sum_{i \in J \setminus I} e^{\theta_i} \right) - n \log\left(\sum_{j \in J} e^{\theta_j} \right) \geq \log(\alpha) \end{aligned} \tag{55}$$

The parameterization (53) introduces a direction of constancy (DOC) [Geyer, 2009, Theorem 1 and the following discussion] that is the same as the DOC we had in the Bradley-Terry model (Section 8 above), the vector all of whose components are the same.

So perhaps we should redo our null space of the Fisher information matrix calculation using the multinomial distribution. But this is not necessary. Movement along the DOC does not change any of the p_i so does not change any of the equations in either of our optimization problems. We do not need to add it to the null space we obtained from the Poisson analysis. (Section 3.17 in [Geyer, 2009] shows that every DOR for the Poisson model is also a DOR for the multinomial model.)

Thus our problem has the abstract form (50) with

$$f(\xi) = -\theta_k + \log\left(\sum_{j \in J} e^{\theta_j} \right) \tag{56}$$

$$\frac{\partial f(\xi)}{\partial \xi_j} = -o_{kj} + \frac{\sum_{i \in J} e^{\theta_i} o_{ij}}{\sum_{i \in J} e^{\theta_i}} \tag{57}$$

where o_{kj} are the components of $O = MN$, and

$$g(\xi) = n \log\left(\sum_{i \in J \setminus I} e^{\theta_i} \right) - n \log\left(\sum_{j \in J} e^{\theta_j} \right) - \log(\alpha) \tag{58}$$

$$\begin{aligned} \frac{\partial g(\xi)}{\partial \xi_j} &= n \frac{\sum_{i \in J \setminus I} e^{\theta_i} o_{ij}}{\sum_{k \in J \setminus I} e^{\theta_k}} - n \frac{\sum_{i \in J} e^{\theta_i} o_{ij}}{\sum_{k \in J} e^{\theta_k}} \\ &= n \sum_{i \in J} (p_i^* - p_i) o_{ij} \end{aligned} \tag{59}$$

where

$$p_i^* = \begin{cases} e^{\theta_i} \big/ \sum_{j \in J \setminus I} e^{\theta_j}, & i \in J \setminus I \\ 0, & \text{otherwise} \end{cases}$$

(p is the vector of probabilities in the OM, p^* is the vector of probabilities in the LCM).

10.5.2 Quick and dirty intervals

If $p_i > 0$ for some $i \in I$, then

$$\left(\sum_{j \in J \setminus I} p_j \right)^n \le (1 - p_i)^n$$

Introducing $\mu_i = n p_i$ we get

$$\alpha \le \left(\sum_{i \in J \setminus I} p_i \right)^n \le \left(1 - \frac{\mu_i}{n} \right)^n \approx \exp(-\mu_i)$$

for large n. Thus this agrees with our analysis in Section 10.4.2 when n is large.

We get the exact inequality

$$\alpha \le \left(1 - \frac{\mu_i}{n} \right)^n$$

or

$$\alpha^{1/n} \le 1 - \frac{\mu_i}{n}$$

or

$$\mu_i \le n(1 - \alpha^{1/n}) = 2.9875$$

when $n = 544$, which is what it is in this example, and $\alpha = 0.05$. And this too agrees approximately with our analysis in Section 10.4.2 above.

10.5.3 Careful coding

We can modify (56) above as

$$f(\xi) = a - \theta_k + \log \left(\sum_{j \in J} e^{\theta_j - a} \right)$$

where a is any real number. We avoid overflow and catastrophic cancellation if we choose

$$a = \theta_m = \max_{j \in J} \theta_j$$

in which case we have

$$f(\xi) = \theta_m - \theta_k + \operatorname{log1p} \left(\sum_{j \in J \setminus \{m\}} e^{\theta_j - \theta_m} \right)$$

in which overflow cannot occur and we avoid catastrophic cancellation in $\log(1 + x)$ for small x.

Using the same definition of θ_m, we modify (57) above as

$$\frac{\partial f(\xi)}{\partial \xi_j} = -o_{kj} + \frac{e^{\theta_k - \theta_m} o_{kj}}{\sum_{i \in J} e^{\theta_i - \theta_m}} = \left[-1 + \frac{e^{\theta_k - \theta_m}}{\sum_{i \in J} e^{\theta_i - \theta_m}} \right] o_{kj}$$

in which overflow cannot occur.

We can modify (58) above as

$$g(\xi) = nb + n \log \left(\sum_{i \in J \setminus I} e^{\theta_i - b} \right) - na - n \log \left(\sum_{j \in J} e^{\theta_j - a} \right) - \log(\alpha)$$

where a and b are any real numbers. We avoid overflow and catastrophic cancellation if we choose a as above and

$$b = \theta_{m^*} = \max_{i \in J \setminus I} \theta_i$$

in which case we have

$$g(\xi) = n \left[\theta_{m^*} - \theta_m + \operatorname{log1p} \left(\sum_{i \in (J \setminus I) \setminus \{m^*\}} e^{\theta_i - \theta_{m^*}} \right) - \operatorname{log1p} \left(\sum_{j \in J \setminus \{m\}} e^{\theta_j - \theta_m} \right) \right] - \log(\alpha)$$

in which overflow cannot occur and we avoid catastrophic cancellation in $\log(1 + x)$ for small x.

Then using the same definitions of θ_m and θ_{m^*} we modify (59) above as

$$\frac{\partial g(\xi)}{\partial \xi_j} = n \left[\frac{\sum_{i \in J \setminus I} e^{\theta_i - \theta_{m^*}} o_{ij}}{\sum_{k \in J \setminus I} e^{\theta_k - \theta_{m^*}}} - \frac{\sum_{i \in J} e^{\theta_i - \theta_m} o_{ij}}{\sum_{k \in J} e^{\theta_k - \theta_m}} \right]$$

in which overflow cannot occur.

10.6 Linearity by computational geometry

Calculate linearity using R package `rcdd` like in Section 6.5 above. We follow Section 4.2 of [Geyer, 2008].

```
tanv <- gout$modmat
vrep <- cbind(0, 0, tanv)
vrep[dat$y > 0, 1] <- 1
system.time(lout <- linearity(d2q(vrep), rep = "V"))

##    user  system elapsed
##   4.334   0.003   4.337

linearity.too <- dat$y > 0
linearity.too[lout] <- TRUE
identical(as.vector(linearity), linearity.too)

## [1] TRUE
```

So this agrees with our analysis in Section 10.3 above, except that the repeated linear programming implementation is slower than the implementation developed here.

10.7 Generic direction of recession

Calculate a GDOR using R package `rcdd` like in Section 6.6 above. More specifically, we follow Section 4.2 of [Geyer, 2008], so we necessarily agree with the GDOR given in Table 1 of [Geyer, 2009].

```
modmat <- gout$modmat
hrep <- cbind(0, 0, -tanv, 0)
hrep[! linearity, ncol(hrep)] <- (-1)
hrep[linearity, 1] <- 1
hrep <- rbind(hrep, c(0, 1, rep(0, ncol(gout$modmat)), -1))
objv <- c(rep(0, ncol(gout$modmat)), 1)
pout <- lpcdd(d2q(hrep), d2q(objv), minimize = FALSE)
gdor <- pout$primal.solution[-length(pout$primal.solution)]

foo <- gdor
names(foo) <- colnames(modmat)
cbind(foo[foo != "0"])

##             [,1]
## (Intercept) "-1"
## v1          "1"
## v2          "1"
## v3          "1"
## v5          "1"
## v1:v2       "-1"
## v1:v3       "-1"
## v1:v5       "-1"
## v2:v3       "-1"
## v2:v5       "-1"
## v3:v5       "-1"
## v1:v2:v3    "1"
## v1:v3:v5    "1"
## v2:v3:v5    "1"
```

This agrees with Table 1 in [Geyer, 2009]. Clean R global environment.

```
rm(list = ls())
```

11 A big data example

11.1 Data

Load the data.

```
foo <- "https://conservancy.umn.edu/bitstream/handle/11299/197369/bigcategorical.txt"
bar <- sub("^.*/", "", foo)
if (! file.exists(bar))
    download.file(foo, bar)
dat <- read.table(bar, header = TRUE, stringsAsFactors = TRUE)
dim(dat)

## [1] 1024    6

names(dat)

## [1] "x1" "x2" "x3" "x4" "x5" "y"
```

The response vector is `y`, the predictors `x1` through `x5` are all categorical. The components of `y` are all counts, so this is a categorical data analysis. This contingency table has 1024 cells and the multinomial sample size (sum of cell counts) is 1055. These data are included in the `glmdr` package.

```
data(bigcategorical)
all.equal(dat, bigcategorical)

## [1] TRUE
```

11.2 Hypothesis Tests

As in Section 10 above, we assume a Poisson sampling model rather than a multinomial sampling model for the reasons stated in that section. Actually, as is well known [Agresti, 2013, Section 8.6.7], the MLE for the mean value parameter vector and the asymptotic chi-square distribution of test statistics is the same for Poisson, multinomial, and product multinomial sampling. So nothing in this section depends on the sampling model.

```
out1 <- glm(y ~ 0 + .,
    family = poisson, data = dat, x = TRUE,
    control = list(maxit = 1e3, epsilon = 1e-12))
out2 <- glm(y ~ 0 + (.)^2,
    family = poisson, data = dat, x = TRUE,
    control = list(maxit = 1e3, epsilon = 1e-12))
out3 <- glm(y ~ 0 + (.)^3,
    family = poisson, data = dat, x = TRUE,
    control = list(maxit = 1e3, epsilon = 1e-12))
out4 <- glm(y ~ 0 + (.)^4,
    family = poisson, data = dat, x = TRUE,
    control = list(maxit = 1e3, epsilon = 1e-12))

## Warning:  glm.fit:  fitted rates numerically 0 occurred

anova(out1, out2, out3, out4, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 0 + (x1 + x2 + x3 + x4 + x5)
## Model 2: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^2
## Model 3: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^3
## Model 4: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^4
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1      1008    1182.17
## 2       918    1076.55  90   105.62    0.1247
## 3       648     811.73 270   264.83    0.5774
## 4       243     277.37 405   534.36 1.605e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Despite the warning from R function `glm`, all of these hypothesis tests are valid because in none of them is `gout4` the null hypothesis.

Tests done as in Table 2 in main article [Eck and Geyer, 2018].

```
anova(out1, out4, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 0 + (x1 + x2 + x3 + x4 + x5)
## Model 2: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^4
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1      1008    1182.17
## 2       243     277.37 765    904.8 0.0003447 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova(out2, out4, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^2
## Model 2: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^4
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1       918    1076.55
## 2       243     277.37 675   799.18 0.0006633 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova(out3, out4, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^3
## Model 2: y ~ 0 + (x1 + x2 + x3 + x4 + x5)^4
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1       648     811.73
## 2       243     277.37 405   534.36 1.605e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These agree with Table 2 in the main article [Eck and Geyer, 2018].

11.3 Maximizing the likelihood

We fit these data using R function `glmdr`.

```
gout <- glmdr(y ~ 0 + (.)^4, family = "poisson",
  data = bigcategorical)
```

11.4 Linearity

We then find the linearity as in preceding sections.

```
linearity <- gout$linearity
sum(linearity)

## [1] 942

sum(! linearity)

## [1] 82
```

11.5 One-sided confidence intervals: Poisson sampling

We now provide one-sided confidence intervals for mean value parameters whose MLE is on the boundary as done before. This is the full table referenced in Eck and Geyer [2018].

```
system.time(mus.CI <- inference(gout))

##    user  system elapsed
##  57.416 161.807  60.102

upper <- round(mus.CI[, ncol(mus.CI)], 4)
tab <- cbind(dat[!linearity, ], upper)
tab
```

```
##     x1 x2 x3 x4 x5 y  upper
## 17   a  a  b  a  a 0 0.1695
## 21   a  b  b  a  a 0 0.1354
## 25   a  c  b  a  a 0 0.2292
## 29   a  d  b  a  a 0 2.4616
## 48   d  d  c  a  a 0 0.0002
## 57   a  c  d  a  a 0 0.0133
## 58   b  c  d  a  a 0 0.5647
## 59   c  c  d  a  a 0 0.2790
## 60   d  c  d  a  a 0 2.1519
## 105  a  c  c  b  a 0 0.1060
## 106  b  c  c  b  a 0 0.6088
## 107  c  c  c  b  a 0 2.2809
## 108  d  c  c  b  a 0 1.4718
## 112  d  d  c  b  a 0 2.9921
## 121  a  c  d  b  a 0 0.0167
## 176  d  d  c  c  a 0 0.0008
## 183  c  b  d  c  a 0 0.0103
## 185  a  c  d  c  a 0 2.9448
## 222  b  d  b  d  a 0 0.0607
## 240  d  d  c  d  a 0 0.0027
## 249  a  c  d  d  a 0 0.0509
## 285  a  d  b  a  b 0 1.8519
## 286  b  d  b  a  b 0 0.0995
## 287  c  d  b  a  b 0 0.0027
## 288  d  d  b  a  b 0 1.1411
## 297  a  c  c  a  b 0 2.8903
## 301  a  d  c  a  b 0 2.2144
## 350  b  d  b  b  b 0 2.9408
## 361  a  c  c  b  b 0 0.0850
## 364  d  c  c  b  b 0 0.0154
## 365  a  d  c  b  b 0 0.6450
## 377  a  c  d  b  b 0 2.8350
## 397  a  d  a  c  b 0 2.9956
## 413  a  d  b  c  b 0 0.0001
## 414  b  d  b  c  b 0 0.0549
## 417  a  a  c  c  b 0 0.5509
## 421  a  b  c  c  b 0 2.4449
## 425  a  c  c  c  b 0 0.0860
## 429  a  d  c  c  b 0 0.0004
## 439  c  b  d  c  b 0 2.0680
## 445  a  d  d  c  b 0 0.0000
## 478  b  d  b  d  b 0 0.2229
## 489  a  c  c  d  b 0 0.0204
## 493  a  d  c  d  b 0 0.1364
## 505  a  c  d  d  b 0 1.2175
## 506  b  c  d  d  b 0 0.2057
## 507  c  c  d  d  b 0 1.5164
## 508  d  c  d  d  b 0 0.0560
## 517  a  b  a  a  c 0 1.1298
## 518  b  b  a  a  c 0 1.0333
## 519  c  b  a  a  c 0 0.1836
## 520  d  b  a  a  c 0 0.6489
## 525   a  d  a  a  c 0 0.0570
## 541   a  d  b  a  c 0 2.8136
## 557   a  d  c  a  c 0 0.0098
## 573   a  d  d  a  c 0 0.1153
## 588   d  c  a  b  c 0 2.4637
## 604   d  c  b  b  c 0 0.2193
## 620   d  c  c  b  c 0 0.0064
## 633   a  c  d  b  c 0 0.1588
## 636   d  c  d  b  c 0 0.3127
## 695   c  b  d  c  c 0 0.4586
## 734   b  d  b  d  c 0 0.0004
## 793   a  c  b  a  d 0 2.6538
## 834   b  a  a  b  d 0 0.0049
## 850   b  a  b  b  d 0 0.0008
## 857   a  c  b  b  d 0 0.0392
## 866   b  a  c  b  d 0 0.0019
## 876   d  c  c  b  d 0 2.9803
## 882   b  a  d  b  d 0 2.9881
## 889   a  c  d  b  d 0 0.0018
## 921   a  c  b  c  d 0 0.0484
## 951   c  b  d  c  d 0 0.4589
## 965   a  b  a  d  d 0 0.2902
## 981   a  b  b  d  d 0 1.9221
## 985   a  c  b  d  d 0 0.2544
## 990   b  d  b  d  d 0 2.9346
## 997   a  b  c  d  d 0 0.7834
## 1009  a  a  d  d  d 0 2.9673
## 1013  a  b  d  d  d 0 0.0169
## 1017  a  c  d  d  d 0 0.0211
## 1021  a  d  d  d  d 0 0.0073
```

Table 3 in Eck and Geyer [2018] is provided below

```
head(tab)
```

```
##    x1 x2 x3 x4 x5 y  upper
## 17  a  a  b  a  a 0 0.1695
## 21  a  b  b  a  a 0 0.1354
## 25  a  c  b  a  a 0 0.2292
## 29  a  d  b  a  a 0 2.4616
## 48  d  d  c  a  a 0 0.0002
## 57  a  c  d  a  a 0 0.0133
```

This also agrees with the first version of this document, which took several hours to do this job (and we are taking only a few seconds). The earlier version got correct results, it just took much longer to do it (because it did not do all of our "careful computing").

11.6 Linearity by computational geometry

Calculate linearity using R package `rcdd` like in Section 10.6 above. Except that we are going to cache the result and the time it takes to compute it. Rather than using the cache feature of R package `knitr`, which should not be committed under version control, we cache it ourselves.

```
tanv <- modmat <- out4$x
vrep <- cbind(0, 0, tanv)
vrep[dat$y > 0, 1] <- 1
suppressWarnings(foo <- try(load("foo-linearity.rda"), silent = TRUE))
if (inherits(foo, "try-error")) {
    time.linearity.big.data <- system.time(
        lout <- linearity(d2q(vrep), rep = "V")
    )
    hostname.linearity.big.data <- NULL
    cpuinfo.linearity.big.data <- NULL
    if (Sys.info()["sysname"] == "Linux") {
        foo <- scan("/proc/cpuinfo", what = character(0), sep = "\n")
        bar <- grep("^model name", foo, value = TRUE)
        bar <- unique(bar)
        baz <- sub("^model name\\t: ", "", bar)
        qux <- system("nslookup `hostname`", intern = TRUE)
        quux <- grep("^Name:", qux, value = TRUE)
        quuux <- sub("^Name:\\t", "", quux)
        quacks <- unique(quuux)
        hostname.linearity.big.data <- quacks[1]
        cpuinfo.linearity.big.data <- baz
    }
    save(lout, time.linearity.big.data,
        hostname.linearity.big.data, cpuinfo.linearity.big.data,
        file = "foo-linearity.rda")
}
linearity.too <- dat$y > 0
linearity.too[lout] <- TRUE
identical(as.vector(linearity), linearity.too)

## [1] TRUE
```

11.7 Times

The linearity operation computed by R function `linearity` in Section 11.6 above took 3 days, 4 hours, 0 minutes, and 40.937 seconds. (This was on `oak.stat.umn.edu`, which is an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz.)

References

A. Agresti. *Categorical Data Analysis*. John Wiley & Sons, Hoboken, NJ, third edition, 2013.

P. Billingsley. *Convergence of Probability Measures*. John Wiley & Sons, New York, second edition, 1999. doi: 10.1002/9780470316962.

P. Billingsley. *Probability and Measure*. John Wiley & Sons, Hoboken, NJ, anniversary edition, 2012.

D. J. Eck and C. J. Geyer. Likelihood inference in exponential families when the maximum likelihood estimator does not exist. 2018.

C. J. Geyer. *Likelihood and Exponential Families*. PhD thesis, University of Washington, 1990. `http://hdl.handle.net/11299/56330`.

C. J. Geyer. Supporting theory and data analysis for “likelihood inference in exponential families and directions of recession”. Technical Report 672, School of Statistics, University of Minnesota, 2008. `http:www.stat.umn.edu/geyer/gdor/phaseTR.pdf`.

C. J. Geyer. Likelihood inference in exponential families and directions of recession. *Electron. J. Stat.*, 3: 259–289, 2009. doi: 10.1214/08-EJS349.

C. J. Geyer. Two examples of agresti, 2016. `http://www.stat.umn.edu/geyer/8931expfam/infinity.pdf`, knitr source `http://www.stat.umn.edu/geyer/8931expfam/infinity.Rnw`.

C. J. Geyer and D. J. Eck. *R package `glmdr`: Exponential Family Generalized Linear Models Done Right, version 0.1*, 2016. `https://github.com/cjgeyer/glmdr/tree/master/package`.

P. R. Halmos. *Finite-Dimensional Vector Spaces*. Springer-Verlag, New York, second edition, 1974. Reprint of 1958 edition published by Van Nostrand.

Geyer. C. J. More supporting data analysis for “likelihood inference in exponential families and directions of recession”. Technical Report 673, School of Statistics, University of Minnesota, 2009. `http:www.stat.umn.edu/geyer/gdor/phase2TR.pdf`.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, 1970.

W. Rudin. *Functional Analysis*. McGraw-Hill, New York, second edition, 1991.