Stat 8931 (Exponential Families) Lecture Notes

Two Examples of Agresti

Charles J. Geyer

November 19, 2016

1 R

```
library(alabama)

## Loading required package:  numDeriv

library(numDeriv)
library(rcdd)

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

The version of R used to make this document is 3.3.1. The version of the `alabama` package used to make this document is 2015.3.1. The version of the `numDeriv` package used to make this document is 2016.8.1. The version of the `rcdd` package used to make this document is 1.1.10.

2 Introduction

The purpose of this handout is to present two toy problems from Agresti (2013) which are two-dimensional exponential families (hence graphable) and illustrate situations where the maximum likelihood estimate (MLE) for the canonical parameter does not exist, although it does exist as a limit of distributions in the family. The MLE canonical parameter values can be though of being “at infinity.” The MLE mean value parameter values are on the relative boundary of convex support of the family (the smallest closed convex set containing the canonical statistic with probability one).

Agresti (2013) calls these models examples of “complete separation” and “quasi-complete separation.” These terms do not generalize beyond logistic regression (or classification with two classes). The general terminology that applies to all exponential families that describes these toy models is MLE distribution concentrated at a single point (instead of complete separation) and MLE distribution not concentrated at a single point but concentrated on a proper subset of the support of the original model (instead of quasi-complete separation).

We use the methods of Geyer (2009b) to analyze these toy models.

3 Example I

Agresti (2013, Section 6.5.1) introduces the notion of complete separation with the following 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
```

Figure 1 shows these data.

Suppose we want to do "simple" logistic regression (one predictor x plus intercept, so the model is two-dimensional). Let's try to do it naively.

```
gout <- glm(y ~ x, family = binomial)

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

summary(gout)

##
## Call:
## glm(formula = y ~ x, family = binomial)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -1.045e-05  -2.110e-08   0.000e+00   2.110e-08   1.045e-05
##
```

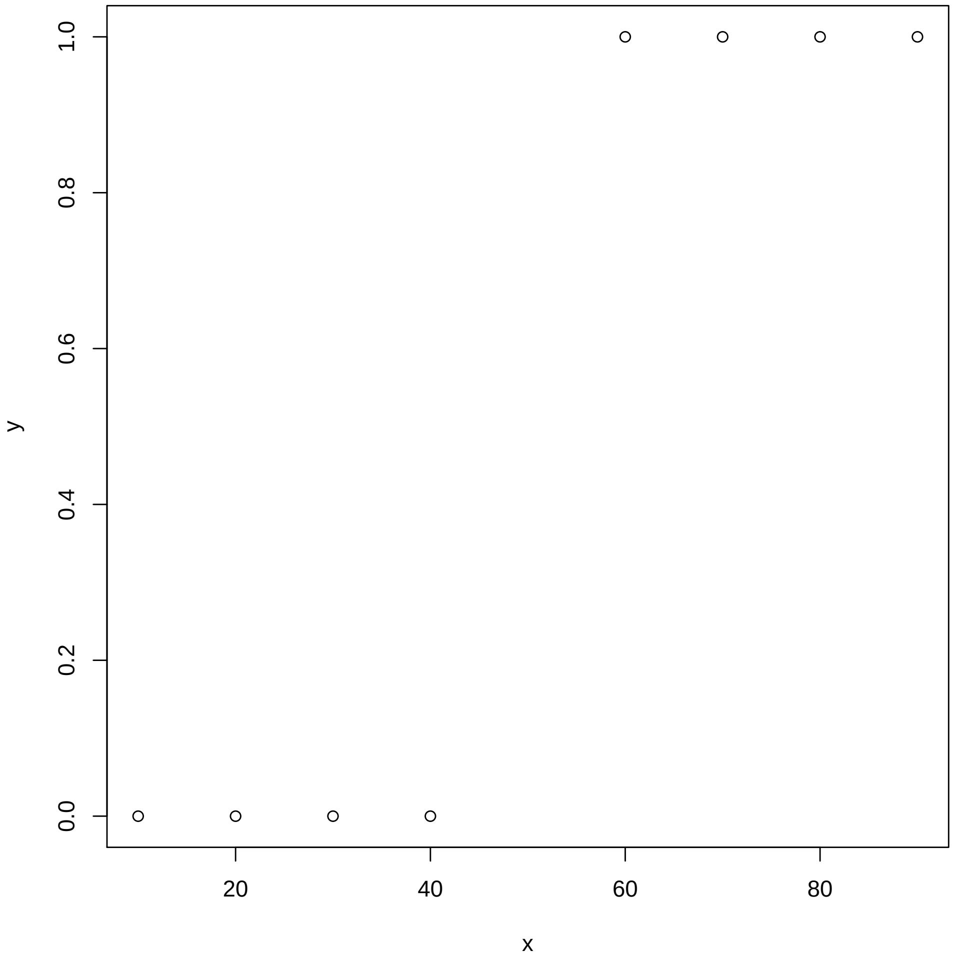

Figure 1: Logistic Regression Data for Example I.

```
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -118.158 296046.187       0        1
## x                2.363   5805.939       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.1090e+01  on 7  degrees of freedom
## Residual deviance: 2.1827e-10  on 6  degrees of freedom
## AIC: 4
##
## Number of Fisher Scoring iterations: 25
```

The R function `glm` does give a warning. But what are you supposed to do about it? If the output of the R function `summary` is to be taken seriously, you cannot tell whether either regression coefficient is nonzero. As we shall see, that is complete nonsense.

In fact, these data are not analyzable by the R function `glm` and its associated functions (generic functions having methods for class `"glm"`). So we will use the theory of Barndorff-Nielsen completions of exponential families from Geyer (2009b).

That theory tells us that we must look at the set of all possible values of the canonical statistic $M^T y$ where M is the model matrix and y is the response vector. For the model, M has two columns: the first column is all ones (the "intercept" column) and the second column is x. So let's find that set. 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.

```
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)
}
```

```
head(yy)

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    0    0    0    0    0    0    0    0
## [2,]    0    0    0    0    0    0    0    1
## [3,]    0    0    0    0    0    0    1    0
## [4,]    0    0    0    0    0    0    1    1
## [5,]    0    0    0    0    0    1    0    0
## [6,]    0    0    0    0    0    1    0    1

dim(yy)

## [1] 256   8
```

For those who know how to count in binary, row i is $i - 1$ expressed in binary. Have you heard the joke: there are two kinds of people in this world, those who divide everything into two kinds and those who don't? And its nerd version: there are 10 kinds of people in this world, those who know binary and those who don't?

For those who don't, the following code shows that every row of `yy` is different, every row contains only zeros and ones, and there are 2^n rows.

```
fred <- apply(yy, 1, paste, collapse = "")
head(fred)

## [1] "00000000" "00000001" "00000010" "00000011" "00000100"
## [6] "00000101"

length(unique(fred)) == length(fred)

## [1] TRUE

all(apply(yy, 1, function(x) all(x %in% 0:1)))

## [1] TRUE

nrow(yy) == 2^n

## [1] TRUE
```

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

```
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.

And now we are stuck. Figure 2 seems to show that the observed data vector is an extreme value, but we cannot easily figure out the direction of recession.

3.1 Tangent Vectors

Vectors $Y(\omega) - y$, where y is the observed value of the canonical statistic vector and $Y(\omega)$ are other possible values of the canonical statistic vector, are called *tangent vectors* (Geyer, 2009b, explains the reason they have this name). If

$$V = \{\, v_i : i \in I \,\}$$

is the set of tangent vectors, then the set of a nonnegative combinations of them, vectors of the form

$$\sum_{i \in A} a_i v_i$$

where A is a finite set and $a_i \geq 0$ for all i, is called the *tangent cone*. It is denoted $\operatorname{con}(\operatorname{pos} T)$ in Geyer (2009b).

Figure 3 shows the tangent vectors and tangent cone. The points in Figures 2 and 3 are the same except in Figure 3 they are moved so the one corresponding to the observed value of the canonical statistic is the origin $(0, 0)$. The gray area is the tangent cone (set of all nonnegative combinations of tangent vectors).

We are interested in the case where a finite subset of the tangent vectors gives the same tangent cone, that is, when S is a finite subset of T such that $\operatorname{con}(\operatorname{pos} S) = \operatorname{con}(\operatorname{pos} T)$. This is obviously the case, when the statistical model has finite support so T is finite, as in logistic regression and log-linear models for contingency tables with multinomial or product multinomial sampling. As we shall see, it is also the case for Poisson regression with log link and for log-linear models for contingency tables with Poisson sampling.

For generalized linear models (GLM) we do not need all the tangent vectors. For the saturated model, tangent vectors $Y(\omega) - y$ such that $Y(\omega)$ and

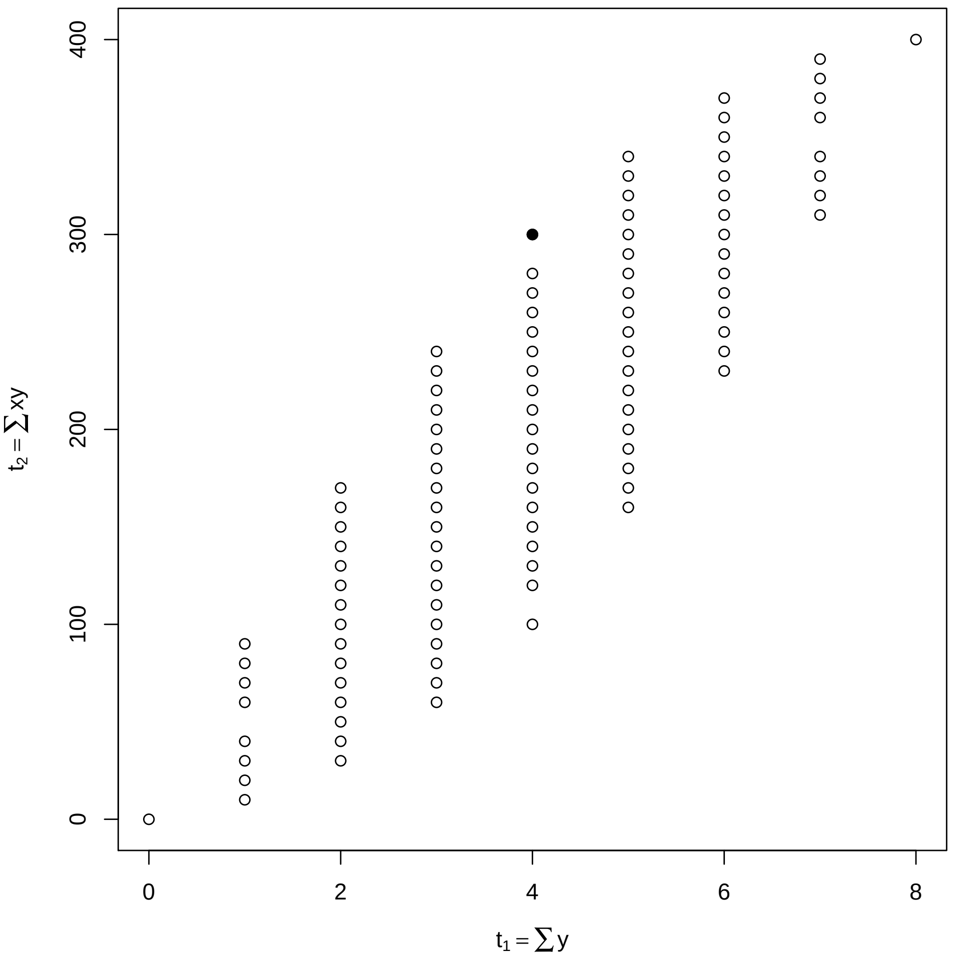

Figure 2: Possible values of the submodel canonical statistic vector $M^T y$ for the data shown in Figure 1. Solid dot is the observed value of the submodel canonical statistic vector.

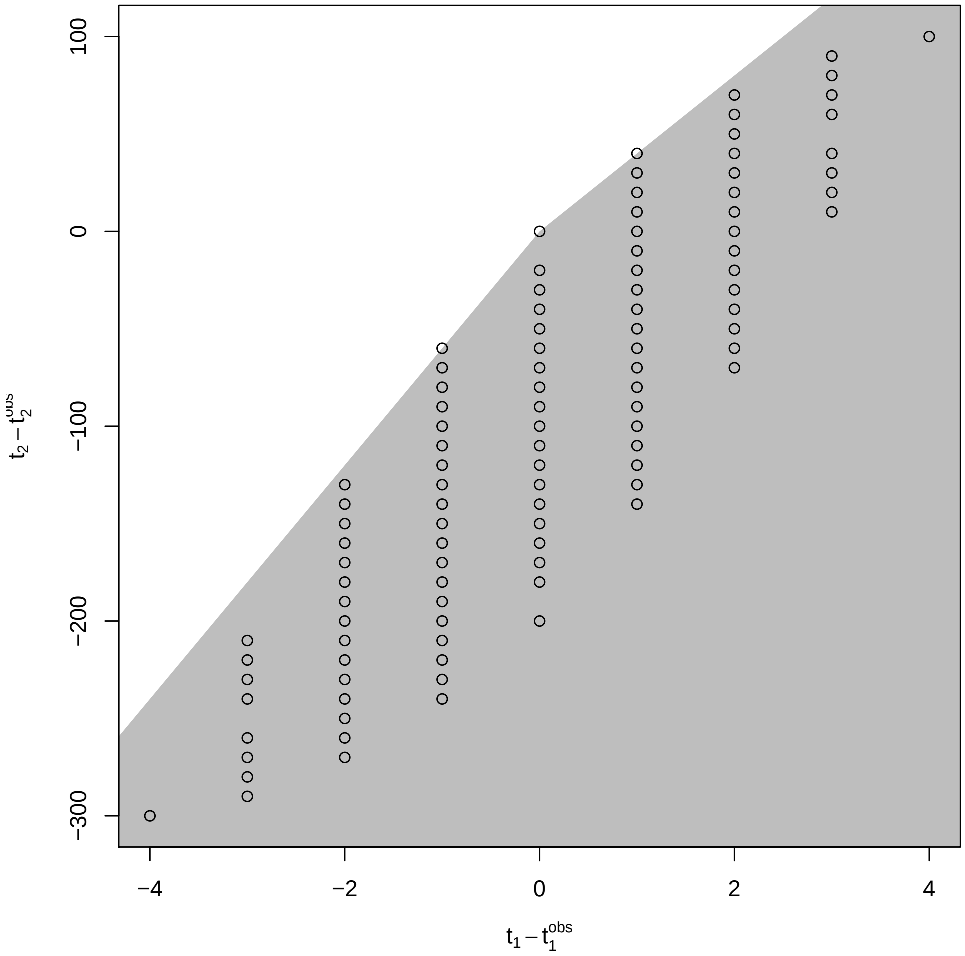

Figure 3: Tangent vectors and tangent cone for data shown in Figure 1. Dots are tangent vectors, gray region is tangent cone.

y differ only in one coordinate are enough to generate the whole tangent cone (Geyer, 2009b, Section 3.11). Moreover, if V_{sat} is a set of vectors generating the tangent cone for the saturated model, then

$$V_{\text{sub}} = \{\, M^T v : v \in V_{\text{sat}} \,\}$$

is a set of vectors generating the tangent cone for the canonical affine submodel with model matrix M (Geyer, 2009b, Section 3.10).

So now we need to learn how to find these tangent vectors for the saturated model.

First consider logistic regression when y has Bernoulli components (zero-or-one-valued). If the observed value of y_i is 0, the only other possible value is 1, so the vector e_i, which has all coordinates equal to 0 except the i-th component, which is 1, is a tangent vector. Similar reasoning says $-e_i$ is a tangent vector if $y_i = 1$.

Second consider logistic regression when y has binomial components. Now we have not only components y_i of the response vector but sample sizes n_i that go with them. We see that if $y_i = 0$, then e_i is a tangent vector (as before), and if $y_i = n_i$, then $-e_i$ is a tangent vector (as before), but now we also have the case $0 < y_i < n_i$ in which case it is possible to change the i-th coordinate either up or down, so both e_i and $-e_i$ are tangent vectors.

Third consider Poisson regression with log link. Just like in the binomial case we have e_i is a tangent vector when $y_i = 0$, and both e_i and $-e_i$ are tangent vectors when $0 < y_i < \infty$. Since there is no upper bound to the range of a Poisson random variable, there is no case where only $-e_i$ is a tangent vector.

3.2 Calculating the Linearity

We now want to calculate a GDOR, but that calculation proceeds in two steps in the algorithm of Geyer (2009b). First we need to find the *linearity* of the tangent cone (Geyer, 2009b, Section 3.12), which is the smallest vector subspace contained in the tangent cone, although Geyer (2009b) also (somewhat sloppily) uses the same term for a set of vectors spanning this vector subspace.

If V_{sub} is a set of vectors generating the tangent cone for the canonical affine submodel, then there is an R function `linearity` in the R package `rcdd` that calculates

$$L_{\text{sub}} = \{\, v \in V_{\text{sub}} : -v \in \operatorname{con}(\operatorname{pos} V_{\text{sub}}) \,\}. \tag{1}$$

This is the set of all the given tangent vectors that are in the linearity of the tangent cone. They also span it, hence determine it.

If we use L_{sub} as defined in (1) to denote a set of tangent vectors. Then the linearity considered as a vector space is denoted span L_{sub}.

The linearity is useful for three reasons. The hyperplane H defined that supports the limiting conditional model (LCM), which is defined in Theorem 6 in Geyer (2009b) and the discussion following it, can be expressed as $H = y + \operatorname{span} L_{\text{sub}}$. So the linearity tells us the support of the LCM. We also need to know what the linearity is in order to calculate a GDOR. Finally, the linearity tells whether the MLE exists or not. It exists if and only if $L_{\text{sub}} \neq V_{\text{sub}}$ (Geyer, 2009b, Theorem 4).

So let us calculate the linearity for our example, the data shown in Figure 1. We follow Section 4.1 of Geyer (2008).

```
tanv <- m
tanv[y == 1, ] <- (-tanv[y == 1, ])
vrep <- makeV(rays = tanv)
lout <- linearity(d2q(vrep), rep = "V")
lout

## integer(0)
```

$M^T e_i$ is just the i-th row of M, so the rows of `m` are either tangent vectors or -1 times tangent vectors. So we assign `tanv` to be `m` and then adjust the signs. For rows of `m` such that corresponding component of `y` is equal to one, we need to change the sign. So the second and third lines of the code chunk above make `tanv` a matrix whose rows are the elements of $V_{\texttt{sub}}$. Then next two lines are idiomatic usage of the R package `rcdd`. The result `lout` is an integer vector giving the indices of the tangent vectors in the linearity, that is, `tanv[linearity, ]` is a basis for the linearity considered as a vector subspace.

Here the result is a vector of length zero, which says the empty set of vectors spans the linearity, which means it is the trivial vector subspace $\{0\}$ that has only one point. We could actually see this in Figure 3, the gray area is a pointed cone, so it contains only the trivial subspace.

So this tells us that the support of the LCM for this example contains only one point. The MLE distribution is completely degenerate, concentrated at y. The MLE distribution says the only data we could have observed is what we did observe; no other data values were possible. Before anyone decides this is weird, let me remind you this is only an estimate, and, as always, estimates are not parameters. This degeneracy causes no problem so long as we don't overinterpret it.

This complete degeneracy of the MLE distribution is what Agresti calls "complete separation."

3.3 Calculating Generic Directions of Recession

If $L_{\text{sub}} \neq V_{\text{sub}}$ the MLE does not exist in the original model (OM), and we need to calculate a GDOR. In this we follow Geyer (2009b, Section 3.13). A vector η in the parameter space is a GDOR if and only if

$$\langle v, \eta \rangle = 0, \qquad v \in L_{\text{sub}} \tag{2a}$$

$$\langle v, \eta \rangle < 0, \qquad v \in V_{\text{sub}} \setminus L_{\text{sub}} \tag{2b}$$

and we can find one such η by solving the following linear program

$$\begin{aligned}
&\text{maximize} \\
&\qquad \varepsilon \\
&\text{subject to} \\
&\qquad\qquad \varepsilon \le 1 \\
&\qquad \langle v, \eta \rangle = 0, \qquad\qquad v \in L_{\text{sub}} \\
&\qquad \langle v, \eta \rangle \le -\varepsilon, \qquad\quad v \in V_{\text{sub}} \setminus L_{\text{sub}}
\end{aligned} \tag{3}$$

where η is a vector, ε is a scalar, and (η, ε) denotes a vector of length one more than the length of η. This vector is the vector of variables of the linear program. The η part of the solution is a generic direction of recession. The ε part does not matter.

So we solve this linear program to calculate the GDOR, still following Section 4.1 of Geyer (2008).

```
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 <- q2d(pout$primal.solution[1:p])
gdor

## [1] -5.0  0.1
```

The code chunk above is not general. It assumes the linearity is trivial, as in the particular example we are working on. More on this later.

So now we have a GDOR, we should put that on the plot, but we cannot. The reason is that η is a vector in the parameter space (as we have been saying over and over), but the space plotted in Figure 2 is the sample space for the canonical statistic vector. (I tried. There is no way to draw η into Figure 2.) What we can do is add the hyperplane

$$H = \{\, x \in \mathbb{R}^2 : \langle x, \eta \rangle = \langle y, \eta \rangle \,\} \tag{4}$$

See Figure 4. The fact that the only possible value of the canonical statistic vector that is on H is the observed value y again tells us that the LCM is completely degenerate, concentrated at the observed value.

3.4 Summary

So now we have the MLE for the LCM, which is the limit of distributions in the OM that maximizes the likelihood. We should say that it is somewhat unusual.

The MLE of the mean value parameter satisfies the "observed equals expected" property $\hat{\mu} = y$. That is because it is the MLE in the LCM.

The MLE of the canonical parameter is very weird. Firstly, it doesn't exist. Second, we can think of it as a point at infinity. Start at the MLE for the canonical parameter in the LCM (which does exist) and head to infinity in the direction of the GDOR. The likelihood in the OM increases all the way but does asymptote at the supremum of the likelihood. The supremum is never achieved (that is why we say "supremum" rather than "maximum"), but we do converge to it.

In Example I the MLE in the LCM is any point in $\mathbb{R}^2$. Since the LCM is completely degenerate, every direction in the parameter space is a direction of constancy (for the LCM) so every point in $\mathbb{R}^2$ corresponds to the same distribution. This is not surprising since there is only one distribution in the

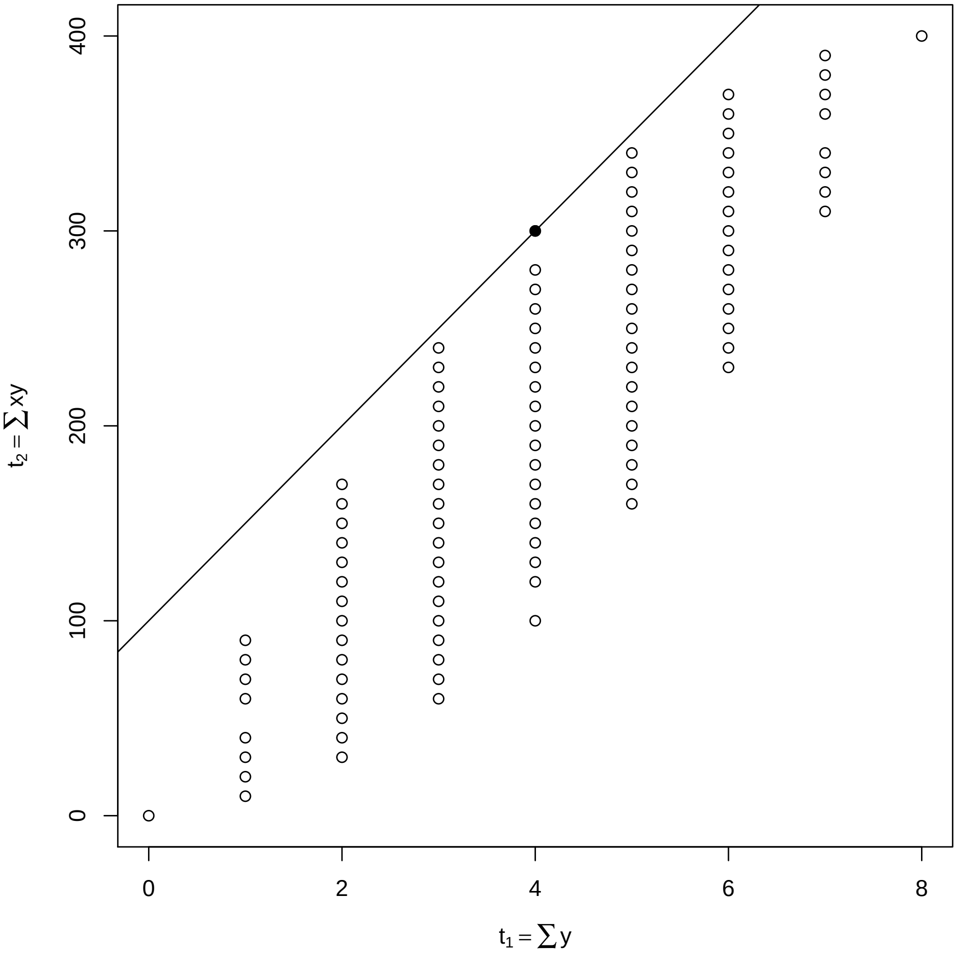

Figure 4: Possible values of the submodel canonical statistic vector $M^T y$ for the data shown in Figure 1. Solid dot is the observed value of the submodel canonical statistic vector. Solid line is the hyperplane (4) on which the LCM is concentrated.

LCM (the one concentrated at the observed data). Thus we can think of the MLE for the canonical parameter as start anywhere and head to infinity in the direction of the GDOR.

3.5 Hypothesis Tests

Hypothesis tests and confidence intervals become difficult and hard to understand when the MLE does not exist in the OM. We will say a lot more about both, but will creep up on them slowly, waiting until we have some interesting examples.

But there is one statistical procedure we can do on Example I. The usual Wilks and Rao hypothesis tests work just fine when n is large (their assumptions are satisfied) when the MLE exists for the null hypothesis. There is no need for the MLE to exist for the alternative hypothesis. The reason is that the Rao test statistic does not use the MLE for the alternative hypothesis, and the Wilks test only seems to use the MLE for both hypotheses. If Θ_0 and Θ_1 are the two hypotheses and l_n is the log likelihood, the Wilks test statistic can be written

$$T_n = 2 \left(\sup_{\theta \in \Theta_1} l_n(\theta) \right) - 2 \left(\sup_{\theta \in \Theta_0} l_n(\theta) \right)$$

and we can calculate the supremum of the log likelihood even when the MLE does not exist (just keep going uphill on the log likelihood until the increases in the steps get very small).

Hence

```
gout.0 <- glm(y ~ 1, family = binomial)
gout.1 <- glm(y ~ x, family = binomial)

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

anova(gout.0, gout.1, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 1
## Model 2: y ~ x
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1         7      11.09
## 2         6       0.00  1    11.09 0.0008678 ***
## ---
## Signif. codes:
## 0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

is completely valid (except for n being small) *despite the warning given by the R function* ***glm*** (we know what we are doing, and the Wilks test is valid). So it is clear that just using a smaller model for which the MLE exists is a non-starter. We have to use the theory of this document when we can prove that no model for which the MLE exists fits the data.

Similarly,

```
add1(gout.0, ~ x, test = "Rao")

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

## Single term additions
##
## Model:
## y ~ 1
##        Df Deviance   AIC Rao score Pr(>Chi)
## <none>       11.09 13.09
## x       1     0.00  4.00    6.6667 0.009823 **
## ---
## Signif. codes:
## 0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

is just as valid, again *despite the warning given by the R function* ***glm*** (we know what we are doing, and the Rao test is valid). The fact that these two tests do not agree about the P-value is because n is small and because P is so small (asymptotics say that absolute error is small for large n not relative error, so both tests do say that P is near zero (a lot smaller than 0.05, for example, and that is valid).

3.6 Confidence Regions

Geyer (2009b) proposes a method of making confidence regions when the MLE for the canonical parameter does not exist (when the data are on the relative boundary of the convex support). Suppose η is a direction of recession, and, as usual, let Y denote the canonical statistic and y its observed value. If we use $\langle Y, \eta \rangle$ as a test statistic for an upper-tailed test, then, because η is a direction of recession for observed data y, the observed value of the test statistic is its maximum value $\langle y, \eta \rangle$. We can also write the event $\langle Y, \eta \rangle = \langle y, \eta \rangle$ as $Y \in H$, where H is given by (4).

We can make a $100(1 - \alpha)\%$ confidence region by inverting a level α test. For each θ_0 in the parameter space, we perform the test with this test statistic and null hypothesis $\theta = \theta_0$. If the test accepts θ_0, then we include this point in the confidence region.

In this case, the test accepts θ_0

$$\operatorname{pr}_{\theta_0}(\langle Y, \eta \rangle \geq \langle y, \eta \rangle) = \operatorname{pr}_{\theta_0}(Y \in H) \geq \alpha. \tag{5}$$

So the confidence region (when the observed value of the canonical statistic is on the relative boundary of the convex support) is the set of parameter values that put probability at least α on the support of the LCM.

Geyer (2009b) uses fairly kludgy calculations for confidence regions. Here we try to be more accurate.

3.6.1 For Submodel Canonical Parameters

Let θ denote the saturated model canonical parameter vector and β the submodel canonical parameter vector, which are related by $\theta = M\beta$, where M is the model matrix.

Define a function f by

$$f(\beta) = \log \operatorname{pr}_\beta(Y \in H)$$

We want to plot the curve which is the locus of points β such that $f(\beta) = \log \alpha$. Of course, we have

$$f(\beta) = \sum_{i=1}^{n} \bigl[y_i \log p_i + (1 - y_i) \log(1 - p_i) \bigr] \tag{6}$$

and we have

$$p_i = \frac{e^{\theta_i}}{1 + e^{\theta_i}} = \frac{1}{1 + e^{-\theta_i}}$$

and

$$\log p_i = \theta_i - \log(1 + e^{\theta_i}) = -\log(1 + e^{-\theta_i})$$

and we want to use the case that guarantees no overflow, so we use the first when θ_i is negative and the second when θ_i is positive. We also want to use the R function `log1p`, which calculates $\log(1 + x)$ much more accurately than does the R expression `log(1 + x)` when `x` is near zero.

We do not want to calculate $q_i = 1 - p_i$ using subtraction because that can result in catastrophic cancellation. Hence we use

$$q_i = 1 - \frac{e^{\theta_i}}{1 + e^{\theta_i}} = \frac{1}{1 + e^{\theta_i}} = \frac{e^{-\theta_i}}{1 + e^{-\theta_i}}$$

and

$$\log q_i = -\log(1 + e^{\theta_i}) = -\theta_i - \log(1 + e^{-\theta_i})$$

and, as before, we want to use the case that guarantees no overflow, and also calculate the logs using the R function `log1p`.

Hence we can write this probability calculation as an R function as follows.

```r
alpha <- 0.05
fred <- function(beta) {
    theta <- m %*% beta
    logp <- ifelse(theta < 0, theta - log1p(exp(theta)),
        - log1p(exp(- theta)))
    logq <- ifelse(theta < 0, - log1p(exp(theta)),
        - theta - log1p(exp(-theta)))
    sum(logp[y == 1]) + sum(logq[y == 0]) - log(alpha)
}
```

Differentiating, we obtain

$$\frac{dp_i}{d\theta_i} = \frac{e^{-\theta_i}}{(1 + e^{-\theta_i})^2} = p_i(1 - p_i) = p_i q_i$$

so

$$\frac{d \log p_i}{d\theta_i} = \frac{p_i q_i}{p_i} = q_i$$

and

$$\frac{dq_i}{d\theta_i} = -p_i q_i$$

so

$$\frac{d \log q_i}{d\theta_i} = \frac{-p_i q_i}{q_i} = -p_i$$

Referring back to (6) we get

$$\begin{aligned} \frac{\partial f(\beta)}{\partial \beta_j} &= \sum_{i=1}^{n} \bigl[y_i q_i - (1 - y_i) p_i \bigr] \frac{\partial \theta_i}{\partial \beta_j} \\ &= \sum_{i=1}^{n} (y_i - p_i) m_{ij} \end{aligned} \tag{7}$$

where m_{ij} is the i, j component of the model matrix M.

So the derivative is given by the following code.

```
dfred <- function(beta) {
    theta <- m %*% beta
    p <- ifelse(theta < 0, exp(theta) / (1 + exp(theta)),
        1 / (1 + exp(- theta)))
    q <- ifelse(theta < 0, 1 / (1 + exp(theta)),
        exp(- theta) / (1 + exp(- theta)))
    foo <- ifelse(y == 1, q, - p)
    as.numeric(t(foo) %*% m)
}
```

Note that in calculating (7) we did not form the vector $y - p$ using subtraction because that could cause catastrophic cancellation, but instead used the R function `ifelse` to define the components not using subtraction.

Let's check that this works.

```
beta <- rnorm(2, sd = 0.1)
grad.one <- dfred(beta)
grad.too <- grad(fred, beta)
all.equal(grad.one, grad.too)

## [1] TRUE
```

Now

$$\frac{d^2 \log p_i}{d\theta_i^2} = \frac{dq_i}{d\theta_i} = -p_i q_i$$

so differentiating (7) gives

$$\frac{\partial^2 f(\beta)}{\partial \beta_j \partial \beta_k} = - \sum_{i=1}^{n} p_i q_i m_{ij} m_{ik} \tag{8}$$

Here's the R function.

```
ddfred <- function(beta) {
    theta <- m %*% beta
    p <- ifelse(theta < 0, exp(theta) / (1 + exp(theta)),
        1 / (1 + exp(- theta)))
    q <- ifelse(theta < 0, 1 / (1 + exp(theta)),
        exp(- theta) / (1 + exp(- theta)))
    foo <- as.numeric(- p * q)
    bar <- t(m) %*% diag(foo) %*% m
    dimnames(bar) <- NULL
    bar
}
```

We remove the `dimnames` of the result because (1) they are useless and (2) the `numDeriv` package doesn't put useless dimnames on its Hessian matrices.

And the check.

```
hess.one <- ddfred(beta)
hess.too <- hessian(fred, beta)
all.equal(hess.one, hess.too)

## [1] TRUE
```

Now we find one point on the curve. As a first step, we find a point on the inside of the confidence region, that is a point β such that $f(\beta) > 0$ (still using f as a synonym for `fred`). We can do that by starting at any point and moving in the direction of recession.

```
beta <- c(0, 0)
while (fred(beta) < 0) beta <- beta + gdor
beta

## [1] -5.0  0.1

fred(beta)

## [1] 1.981878
```

Then we find a point on the curve by inequality constrained minimization

```
objfun <- function(beta) sum(beta^2) / 2
objgrd <- function(beta) beta
confun <- fred
conjac <- function(beta) rbind(dfred(beta))
aout <- auglag(beta, objfun, objgrd, hin = confun, hin.jac = conjac,
    control.outer = list(trace = FALSE))
stopifnot(aout$convergence == 0)
beta.start <- aout$par
beta.start

## [1] -1.34039018  0.04033369

fred(beta.start)

## [1] 4.3086e-08
```

There seem to be no R packages that do what we want here, so we just invent our own algorithm. From the current point on the curve β we do the following, letting f stand for the function (6) that implicitly defines the curve we are following. Let ε stand for an arbitrary, fixed small positive number.

1. $g := \nabla f(\beta)$.
2. $v := (-g_2, g_1)$ $\{$ rotate g by 90° $\}$.
3. $v := v / \|v\|$.
4. $\beta := \beta + \varepsilon \cdot v$.

Now we have moved a ways along the curve (about ε in whatever norm $\| \cdot \|$ indicates), or, more precisely, "almost" along the curve, because we took a step along the straight line tangent to the curve.

We do Newton iterations to get back onto the curve. Again letting $g = \nabla f(\beta)$, we try to find the s such that $f(\beta + sg) = 0$. The Taylor series up to first derivative terms is

$$f(\beta + sg) \approx f(\beta) + s \langle \nabla f(\beta), g \rangle.$$

Setting this equal to zero and solving for s gives

$$s = - \frac{f(\beta)}{\|\nabla f(\beta\|^2},$$

where now we have to be using the Euclidean norm

$$\|g\|^2 = \langle g, g \rangle.$$

Then we set $\beta = \beta + s \nabla f(\beta)$, and then we iterate to convergence (till we are on the curve again). Let τ be another small number (the convergence tolerance). Then we continue our algorithm

5. Repeat the following steps.
   (a) If ($|f(\beta)| < \tau$) stop the repetition.
   (b) $g := \nabla f(\beta)$.
   (c) $s := -f(\beta)/\|g\|^2$.
   (d) $\beta := \beta + sg$.

And all of that takes us just one step along the curve. Then we repeat the whole thing over and over, taking many steps to trace the whole curve.

Since the curve goes to infinity when the MLE does not exist in the conventional sense, we stop when we have gone as far as we want to plot. In the following we make the step size proportional to the square of the norm of the parameter vector so the step size increases as we approach infinity.

```
tracer <- function(beta.start, confun, conjac, epsilon,
    max.norm.beta, tol = 1e-6) {

    norm <- function(beta) sqrt(sum(beta^2))
    is.out <- function(beta) norm(beta) > max.norm.beta

    if (is.out(beta.start))
        stop("norm of beta.start is already greater than max.norm.beta")

    move.to.curve <- function(beta) {
        repeat {
            fbeta <- confun(beta)
            if (abs(fbeta) < tol) return(beta)
            g <- as.vector(conjac(beta))
            s <- (- fbeta) / sum(g^2)
            beta <- beta + s * g
        }
    }

    beta.start <- move.to.curve(beta.start)

    beta <- beta.start
    betas <- beta
    repeat {
        g <- as.vector(conjac(beta))
        v <- c(- g[2], g[1])
        v <- v / norm(v)
        beta <- beta + v * epsilon * norm(beta)^2
        beta <- move.to.curve(beta)
        if (is.out(beta)) break
        betas <- rbind(betas, beta)
    }

    beta <- beta.start
    repeat {
        g <- as.vector(conjac(beta))
        v <- c(g[2], - g[1])
        v <- v / norm(v)
        beta <- beta + v * epsilon * norm(beta)^2
        beta <- move.to.curve(beta)
        if (is.out(beta)) break
        betas <- rbind(beta, betas)
    }

    dimnames(betas) <- NULL
    betas
}
betas <- tracer(beta.start, confun, conjac, 0.001, 400)
dim(betas)

## [1] 1499    2
```

The result is shown in Figure 5.

3.6.2 Confidence Intervals via Projections

Now that we have our basic confidence region (Figure 5) we can make confidence intervals for the regression coefficients by finding the maximum and minimum values of each over the region.

Just from looking at the picture (Figure 5), it is obvious that the minimum for β_1 is $-\infty$, and the maximum for β_2 is $+\infty$, so these will be one-sided confidence intervals.

We can find the other endpoints by constrained optimization.

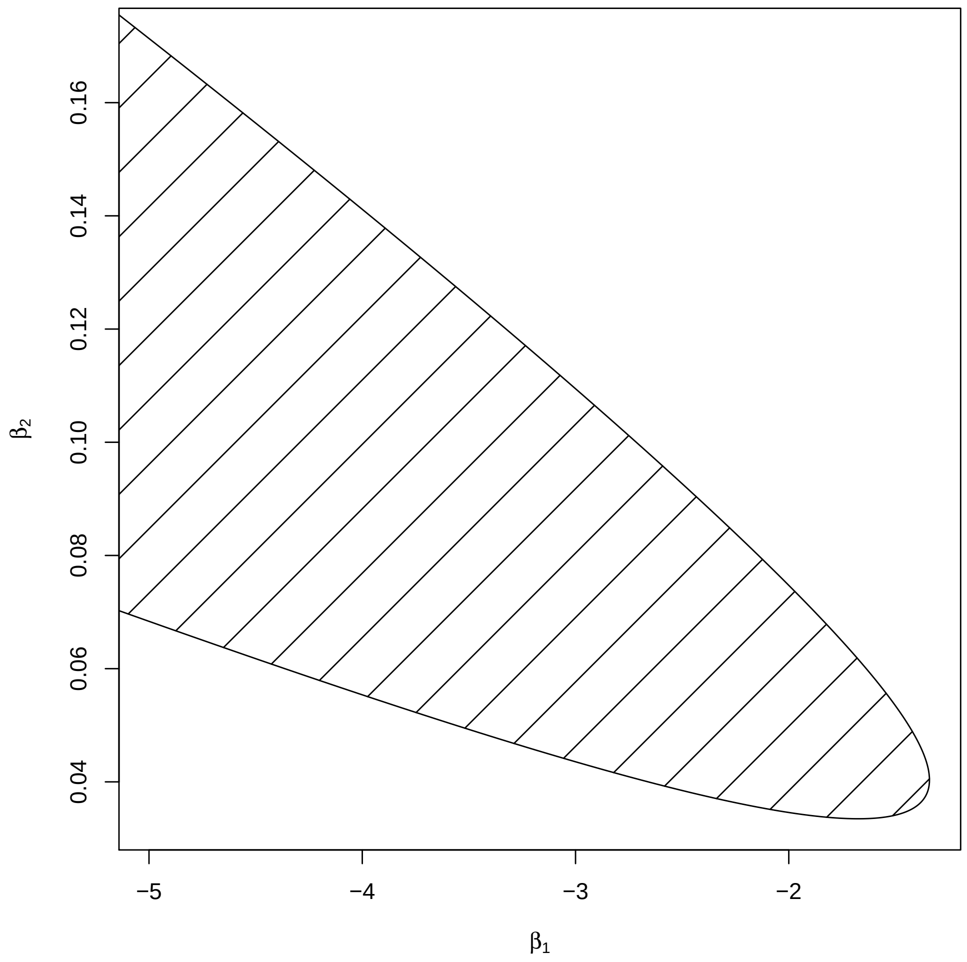

Figure 5: 95% Confidence Region for Submodel Canonical Parameter Vector β (a. k. a., Coefficients) for Example I.

```
aout <- auglag(beta.start, function(beta) - beta[1], function(beta) c(-1, 0),
    hin = confun, hin.jac = conjac, control.outer = list(trace = FALSE))
stopifnot(aout$convergence == 0)
aout$par

## [1] -1.3403900  0.0403407
```

The 95% confidence interval for β_1 is $(-\infty, -1.34)$. Only the value of β_1 at the solution of this minimization problem matters. Ignore the value of β_2.

```
aout <- auglag(beta.start, function(beta) beta[2], function(beta) c(0, 1),
    hin = confun, hin.jac = conjac, control.outer = list(trace = FALSE))
stopifnot(aout$convergence == 0)
aout$par

## [1] -1.34427951  0.03892884
```

The 95% confidence interval for β_2 is $(0.039, \infty)$. Only the value of β_2 at the solution of this minimization problem matters. Ignore the value of β_1.

Observe that these confidence intervals loose a lot of information that is available in the confidence region. But that is the nature of separate confidence intervals.

Also observe that, because these confidence intervals come from a confidence region, they have simultaneous 95% coverage. In this respect, as in many respects, they are unlike the confidence intervals put out by the R generic functions `predict` and `confint`.

There is nothing special about these two particular confidence intervals. Clearly we could make a confidence interval for any function of β by maximizing and minimizing the function over the region shown in Figure 5.

3.6.3 For Submodel Mean Value Parameters

Let us map the confidence region shown in Figure 5 to the (submodel) mean value parameter scale. We know from exponential family theory (Theorems 16 and 43 of the lecture notes for this course Geyer, 2016a).

The submodel mean value parameter is $E(M^T y)$, where M is the model matrix and y is the response vector.

```
thetas <- m %*% t(betas)
thetas <- t(thetas)
```

```
# as with betas, rows of thetas are parameter vectors
mus <- 1 / (1 + exp(- thetas))
taus <- t(m) %*% t(mus)
taus <- t(taus)
# as with betas, rows of taus are parameter vectors
# in this case submodel mean value parameter vectors
dim(taus)

## [1] 1499    2
```

And we plot this curve along with the support of the submodel canonical statistic (Figure 6).

To make progress, we need to figure out where the curve is hitting the boundary. Where is it going as we let the norm of β go to infinity? We can get an idea what is happening by looking at μ (the mean of the response vector).

```
round(head(mus, n = 1), 5)

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    0    0    0 0.95    1    1    1    1

round(tail(mus, n = 1), 5)

##         [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1499,]    0    0    0    0 0.05    1    1    1
```

Both of these have all but one of the mean values agreeing with the observed values, and the mean value for one component of the response vector moved to get $\text{pr}(Y \in H) = 0.05$. Because we are just doing toy problems using methods that do not generalize to more than two parameters, we will not bother to work this out theoretically, and just assume the above is the limit.

Figure 7 shows the confidence region for the submodel mean value parameter vector τ. It is clear from this figure that even with this small amount of data (8 components of the response vector) the confidence region is a small fraction of the whole submodel mean value parameter space (the convex hull of the dots in the figure).

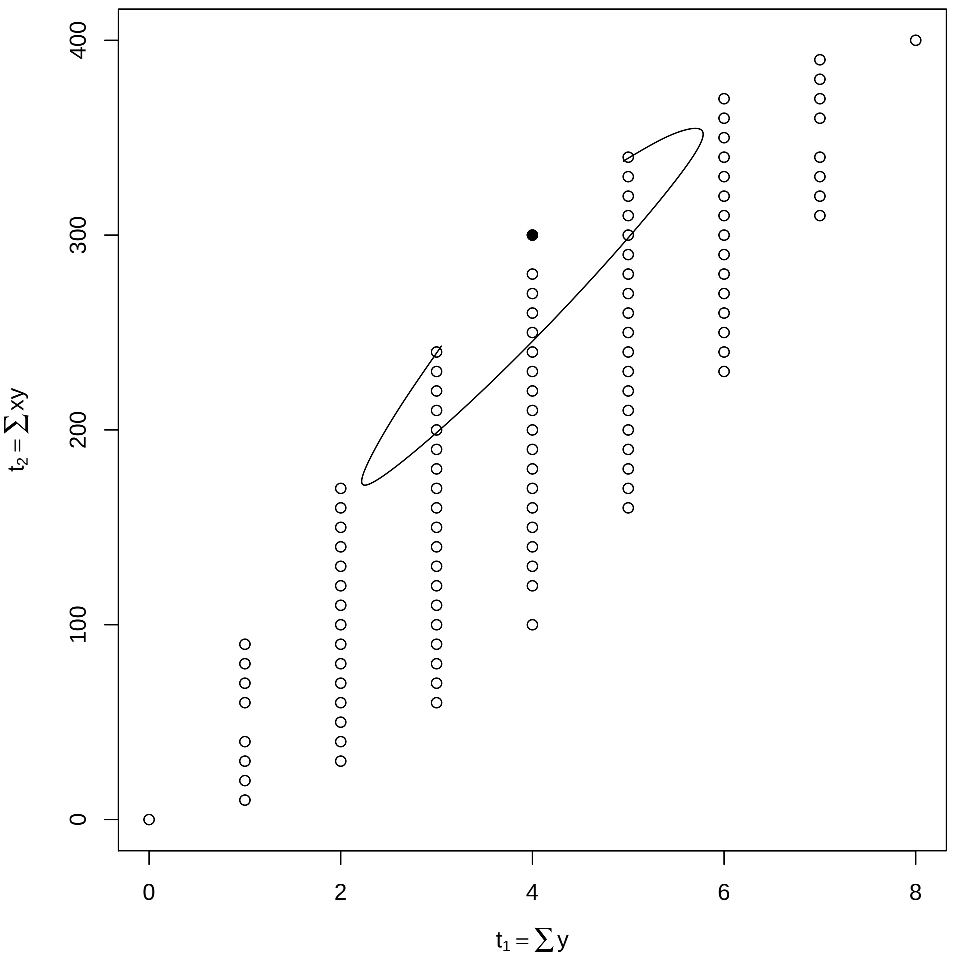

Figure 6: Possible values of the submodel canonical statistic vector $M^T y$ for the data shown in Figure 1. Solid dot is the observed value of the submodel canonical statistic vector. Line is the curve of β values that is the boundary of the confidence region in Figure 5 mapped to submodel mean value parameter values $\tau = M^T \mu$, where μ is the mean of the response vector. See Figure 7 for a better plot of this confidence region.

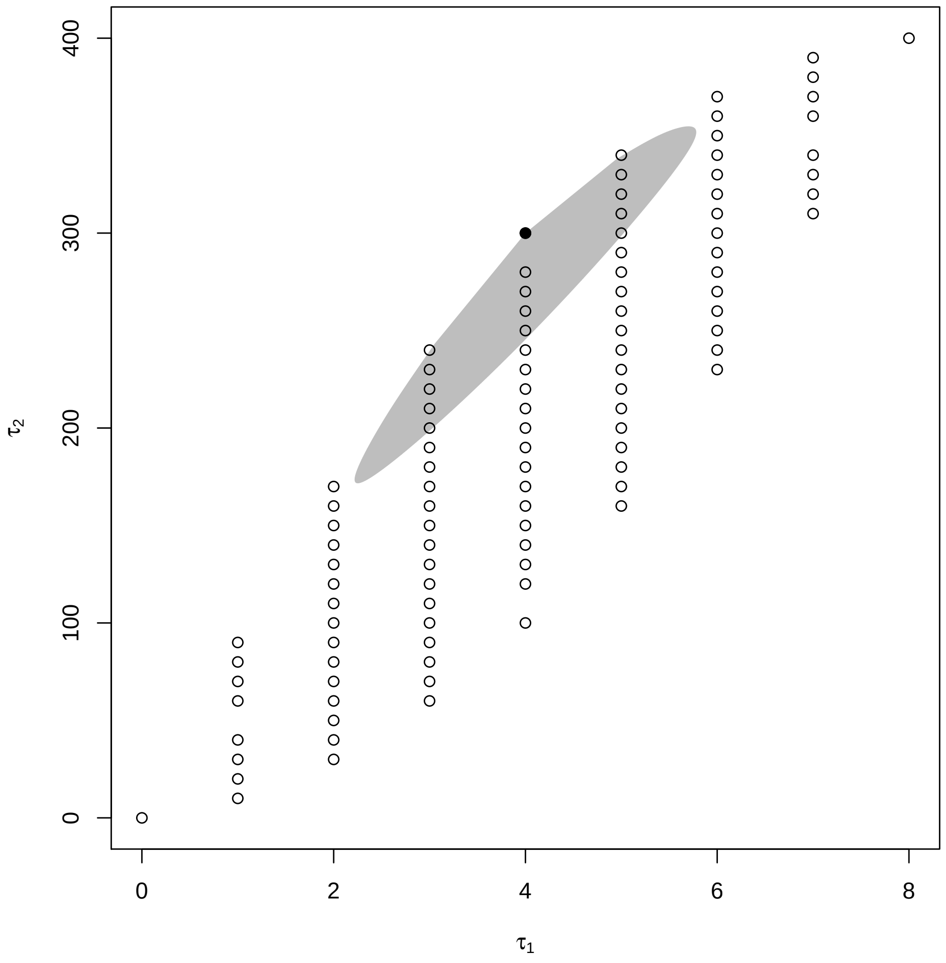

Figure 7: 95% Confidence Region (shaded gray) for Submodel Mean Value Parameter ($\tau = M^T \mu$, where μ is the mean of the response vector) for Example I. Dots are as in Figure 2 (hollow dots are possible values of the submodel canonical statistic, solid dot is the observed value.

3.6.4 For Saturated Model Mean Value Parameters

After all of the above, the analogous confidence region for the saturated model mean value parameter vector $\mu = E(y)$, where y is the response vector, goes easier (the curve has already been calculated for x values in the observed data (the R object `mus`), but we would like to have predictions for all possible x values, not just those in the observed data. Thus we are also doing "prediction intervals" for unobserved predictor values x. These have the form

$$\mu(x) = \operatorname{logit}^{-1}(\beta_1 + \beta_2 x), \tag{9}$$

where $\operatorname{logit}^{-1}$ denotes the inverse of the logit function.

In order for the calculations to not get out of hand, we do calculate on a grid of values. Although (9) makes sense for any real x, we only cover x values near the observed predictor values (interpolating not extrapolating).

```
x.new <- seq(-20, 120, length = 1401)
m.new <- cbind(1, x.new)
thetas.new <- m.new %*% t(betas)
thetas.new <- t(thetas.new)
mus.new <- 1 / (1 + exp(- thetas.new))
dim(mus.new)

## [1] 1499 1401
```

Somewhere along these curves of μ vectors, maximum and minimum values are achieved. As with the τ vectors, we take an (assumed) limit by rounding and hoping it is correct, without doing the required real analysis.

```
mu.new.low <- apply(mus.new, 2, min)
mu.new.hig <- apply(mus.new, 2, max)
```

So here is the plot (Figure 8).

This plot looks a bit odd near $x = 40$ and $x = 60$. Some further thought yields the insight that the limits

```
round(head(mus, n = 1), 5)

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    0    0    0 0.95    1    1    1    1
```

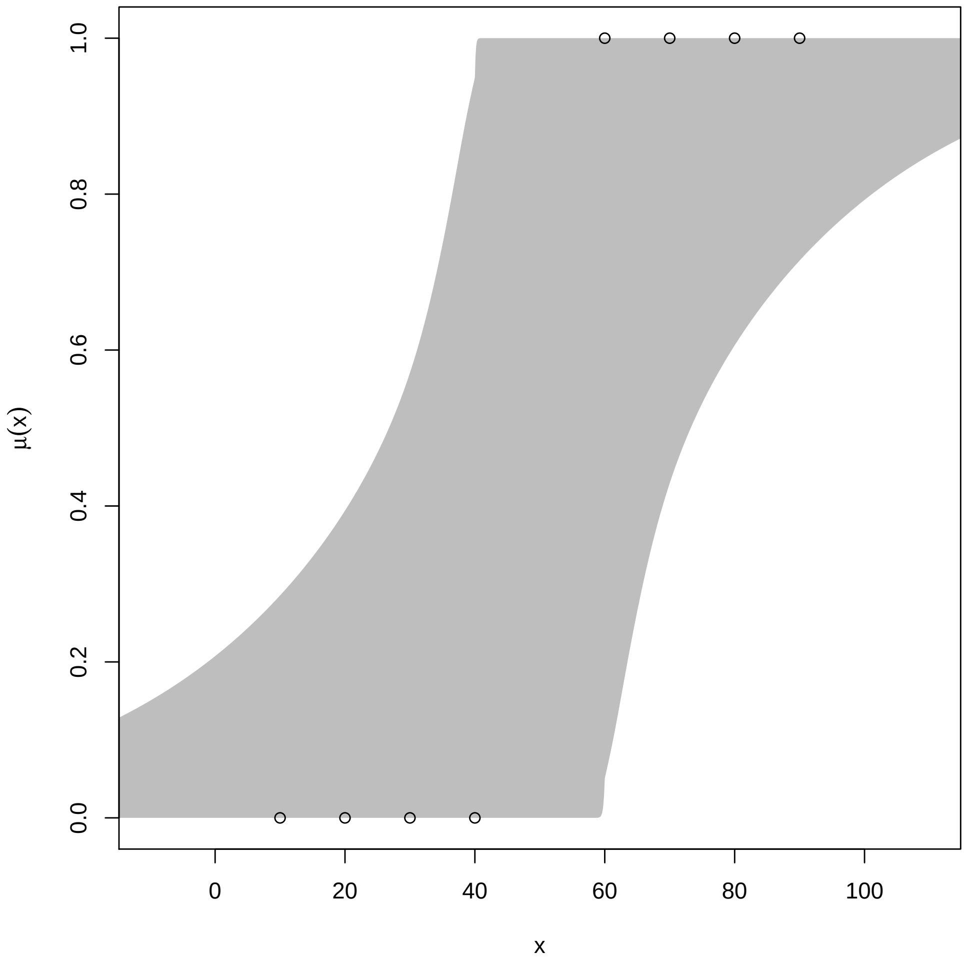

Figure 8: 95% Confidence Region (shaded gray) for Saturated Model Mean Value Parameter and Prediction Intervals (9) for Example I. Dots are as in Figure 1 (scatter plot of response versus predictor). See Figure 9 for a corrected version of this plot.

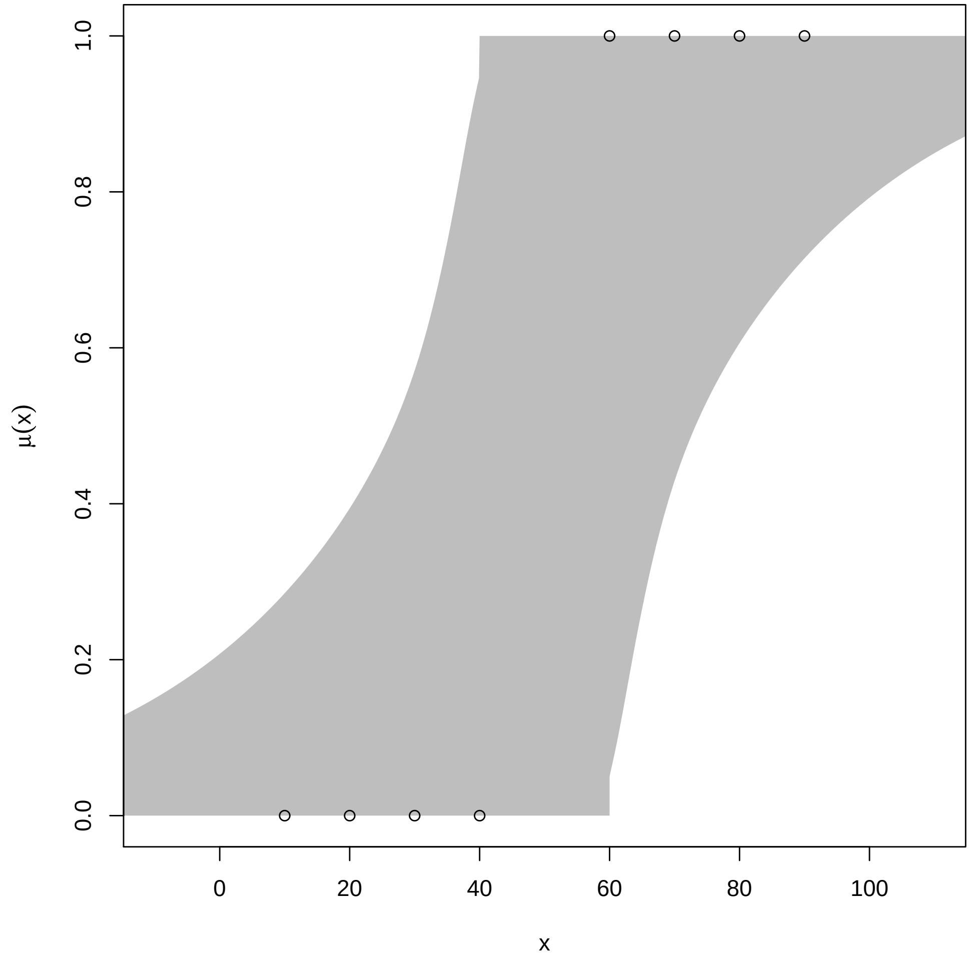

Figure 9: 95% Confidence Region (shaded gray) for Saturated Model Mean Value Parameter and Prediction Intervals (9) for Example I. Dots are as in Figure 1 (scatter plot of response versus predictor).

```
round(tail(mus, n = 1), 5)

##          [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1499,]    0    0    0    0 0.05    1    1    1
```

occur when the “slope” coefficient goes to infinity. Thus the boundary of the confidence interval must have a vertical section there. We correct Figure 8 accordingly. The corrected figure is Figure 9.

This figure may seem more interpretable than Figure 5 or Figure 7, but it actually carries a lot less information. As we saw in Section 3.6.2 above, (one-dimensional) confidence intervals do not contain all of the information provided by the (two-dimensional) confidence region. Here even though we have an infinite number of confidence intervals, one for $\mu(x)$ for each x, all of these intervals together do not tell us what either two-dimensional confidence regions do (Figures 5 and 7). Confidence intervals are just less informative than confidence regions (everywhere in statistics, not just in this context).

As we observed in Section 3.6.2 above, because these confidence intervals come from a confidence region, they have simultaneous 95% coverage, even though there are an infinite number of them.

3.6.5 For Saturated Model Canonical Parameters

We make a similar map for saturated model canonical parameters and similar “prediction intervals” given by

$$\theta(x) = \beta_1 + \beta_2 x \tag{10}$$

(Figure 10).

Similar comments about loss of information and simultaneous coverage to those made at the end of Sections 3.6.2 and 3.6.4 can be made here. (These confidence intervals all together do not contain as much information as the confidence regions, and these confidence intervals have simultaneous 95% coverage.)

3.7 Discussion and Criticism

The confidence regions discussed in Section 3.6 were presented in Geyer (2009b), at least implicitly (only plots and intervals similar to Figure 9 were shown in that article). They were published with no more theoretical backing than was presented here.

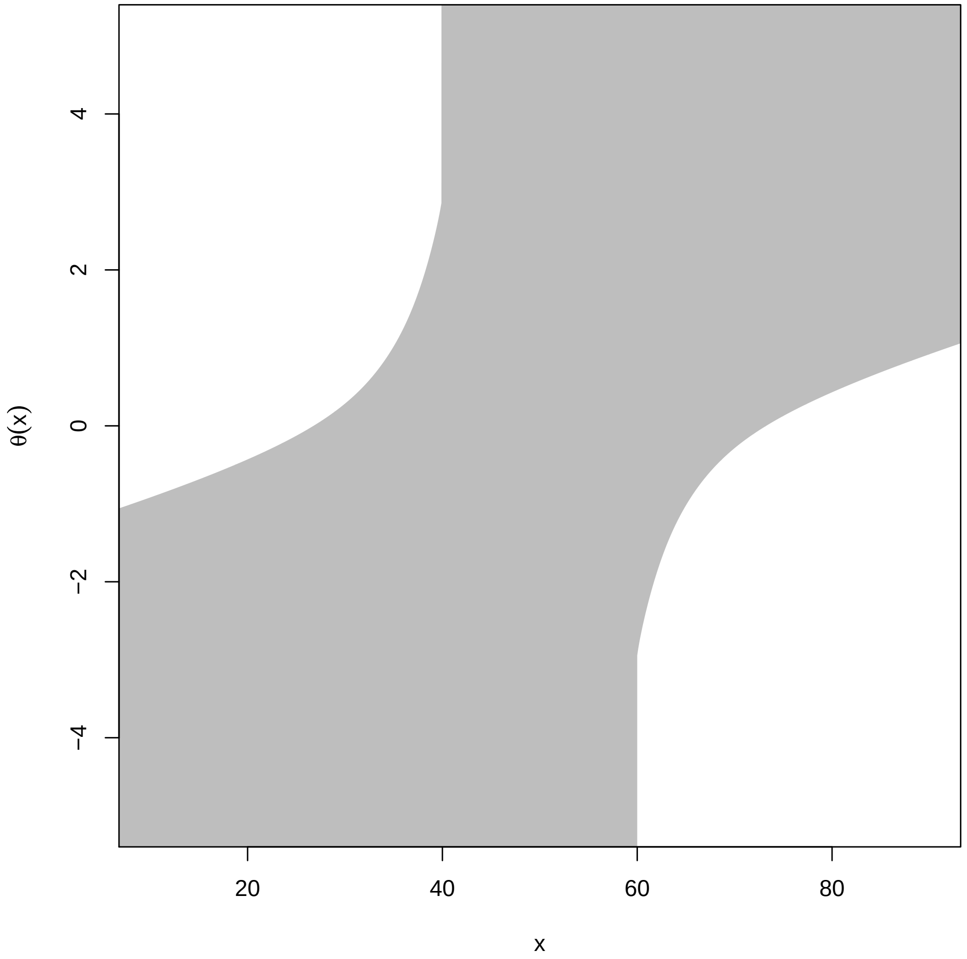

Figure 10: 95% Confidence Region (shaded gray) for Saturated Model Canonical Parameter and Prediction Intervals (10) for Example I.

Strictly speaking, the argument provided here and there makes no sense. It may provide some motivation but does not prove what we and Geyer (2009b) seem to claim. Geyer (2009b) in several places calls the intervals derived from these confidence regions "exact" meaning they are not asymptotic.

But consider the following. The "test statistic" $\langle Y, \eta \rangle$ for the "hypothesis test" being inverted to make the confidence intervals depends on the direction of recession η, which depends on the data, and we did not take this dependence into account when we calculated probabilities to construct confidence intervals and regions. Hence we put "test statistic" and "hypothesis test" in scare quotes above.

Worse, we have so far only given a recipe for confidence regions when the data are at a vertex of the convex support. When we discuss Agresti's other example below, we will give a recipe for confidence regions when the data are on the relative boundary of the convex support. Both recipes come from Geyer (2009b). But what about when the data are in the relative interior of the convex support so the MLE exists in the conventional sense? Geyer (2009b) suggests that one just do whatever one would usually do in that case. For GLM that presumably means confidence intervals produced by the R generic functions `predict` and `confint`. The Neyman-Pearson theory of confidence intervals and confidence regions says one averages over all possible data to get coverage probabilities (which are functions of the true unknown parameter values). This means we have to average over all kinds of data (on the relative boundary and in the relative interior) to get the coverage probability. It is literally meaningless to assign coverage probabilities to recipes that only work for part of the data space.

So the most we can honestly and pedantically claim for the confidence regions and intervals presented so far is that they are well motivated and will do their part *when combined with recipes for confidence regions and intervals for other other parts of the data space* to get approximately correct coverage.

We say "approximately" correct coverage because no procedure for confidence intervals or regions for discrete data can be exact in the sense of providing coverage that is exactly the nominal coverage (95% here) for all values of the unknown parameters. Let $R(x)$ be the confidence region for observed data x. It is a subset of the parameter space for the parameter in question. Then the coverage probability is

$$\text{cover}(\theta) = \text{pr}_\theta\{\theta \in R(X)\} = \sum_{x \in \mathcal{X}} I_{R(x)}(\theta) f_\theta(x)$$

where $\mathcal{X}$ is the sample space, I_A is the indicator function of the set A, and f_θ is the probability mass function for the model. Consider a model where $f_\theta(x)$ is strictly positive for all $x \in \mathcal{X}$ (logistic regression, Poission regression, log-linear models for categorical data, for example). Then any $x \in \mathcal{X}$ as θ moves from inside to outside of $R(x)$, the coverage probability jumps discontinuously by $f_\theta(x)$. Thus $\text{cover}(\theta)$ cannot be a constant function of θ.

See Section 1.1 in Geyer and Meeden (2005) for more discussion of this issue. Papers cited by Geyer and Meeden (2005) and their discussants give yet more discussion of the issue. The web page Geyer (2016b) gives still more examples. It also shows that the confidence region recipe presented above works well to repair otherwise badly performing conventional confidence intervals (Wald intervals and intervals based on the variance-stabilizing transformation for the binomial distribution).

We can say that the recipe presented by Geyer (2009b) and illustrated here is the only general method that has been put forward in the literature recommended for all exponential family statistical models. We continue this discussion in Section 4.6.3 below.

It has since occurred to us that there is another recipe that is well known that also does something sane when the observed data are on the relative boundary of the convex support. That is confidence intervals derived by inverting the likelihood ratio test (Wilks test), those done by the R function `confint`, for example. However, as of R version 3.3.1 the function confint is broken when applied to binomial data on the boundary of the convex support. So this isn't really a competitor. Also the 5102 course web page Geyer (2016b), which now includes likelihood based confidence intervals, shows they do worse than the proposal of Geyer (2009b), at least for the specific example of binomial confidence intervals.

4 Example II

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 Example I.

```
x <- c(x, 50, 50)
y <- c(y, 0, 1)
```

Figure 11 shows these data.

Again we want to do "simple" logistic regression (one predictor x plus

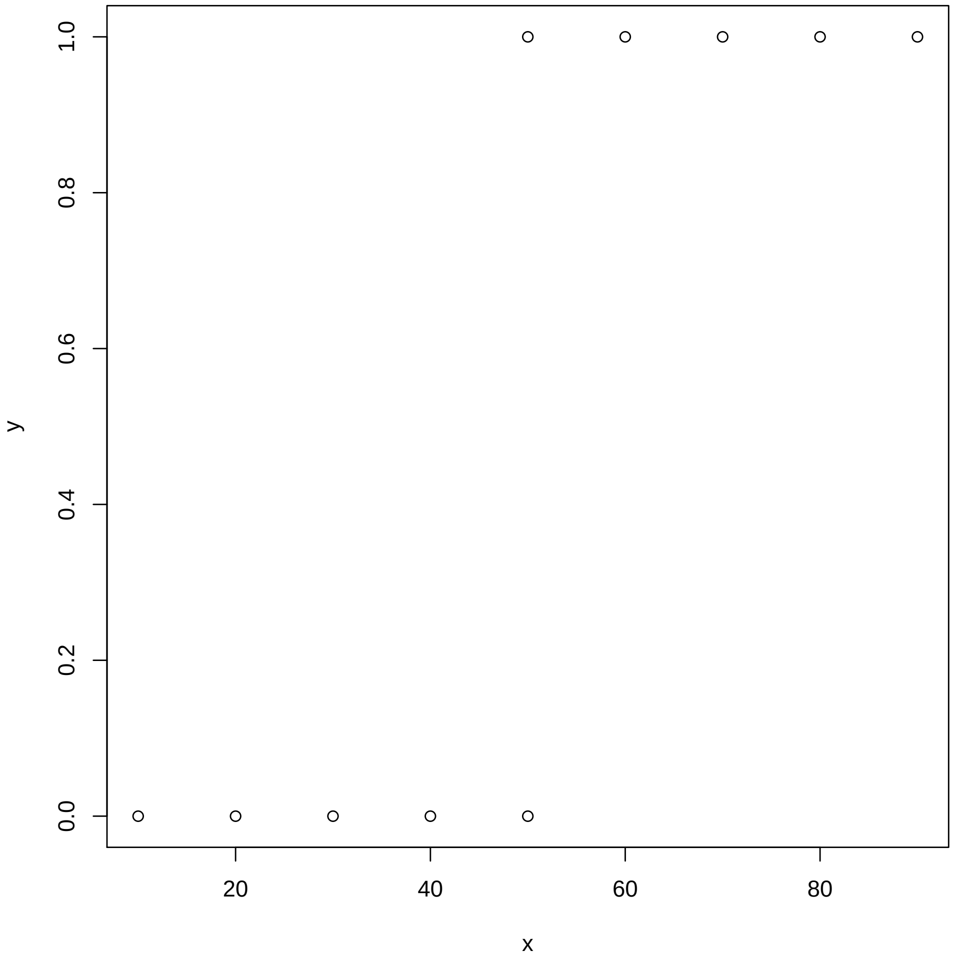

Figure 11: Logistic Regression Data for Example II.

intercept, so the model is two-dimensional). Again, if we try to do it naively, the R function `glm` complains.

```
gout <- glm(y ~ x, family = binomial)

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

summary(gout)

##
## Call:
## glm(formula = y ~ x, family = binomial)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.177   0.000   0.000   0.000   1.177
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -98.158  39288.592  -0.002    0.998
## x               1.963    785.772   0.002    0.998
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 13.8629  on 9  degrees of freedom
## Residual deviance:  2.7726  on 8  degrees of freedom
## AIC: 6.7726
##
## Number of Fisher Scoring iterations: 21
```

But, again, the warning is useless because R does not give us any help in doing anything valid with these data.

As in Section 3 we find the support of the submodel canonical statistic.

```
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)
}
```

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

and plot it (Figure 12).

4.1 Tangent Vectors

As before we plot the tangent cone (Figure 13).

4.2 Calculating the Linearity

As in Section 3.2 we need to calculate the linearity of the tangent cone, and we do it the same way as before.

```
tanv <- m
tanv[y == 1, ] <- (-tanv[y == 1, ])
vrep <- makeV(rays = tanv)
lout <- linearity(d2q(vrep), rep = "V")
lout

## [1]  9 10
```

This tells us that the support of the LCM for is one-dimensional, the point $M^T y$ (observed value of the submodel canonical statistic) plus the one-dimensional vector space spanned by L_{sub}.

```
tanv[lout, ]

##            x
## [1,]  1  50
## [2,] -1 -50
```

Although there are two vectors here, they are parallel, hence span a one-dimensional space.

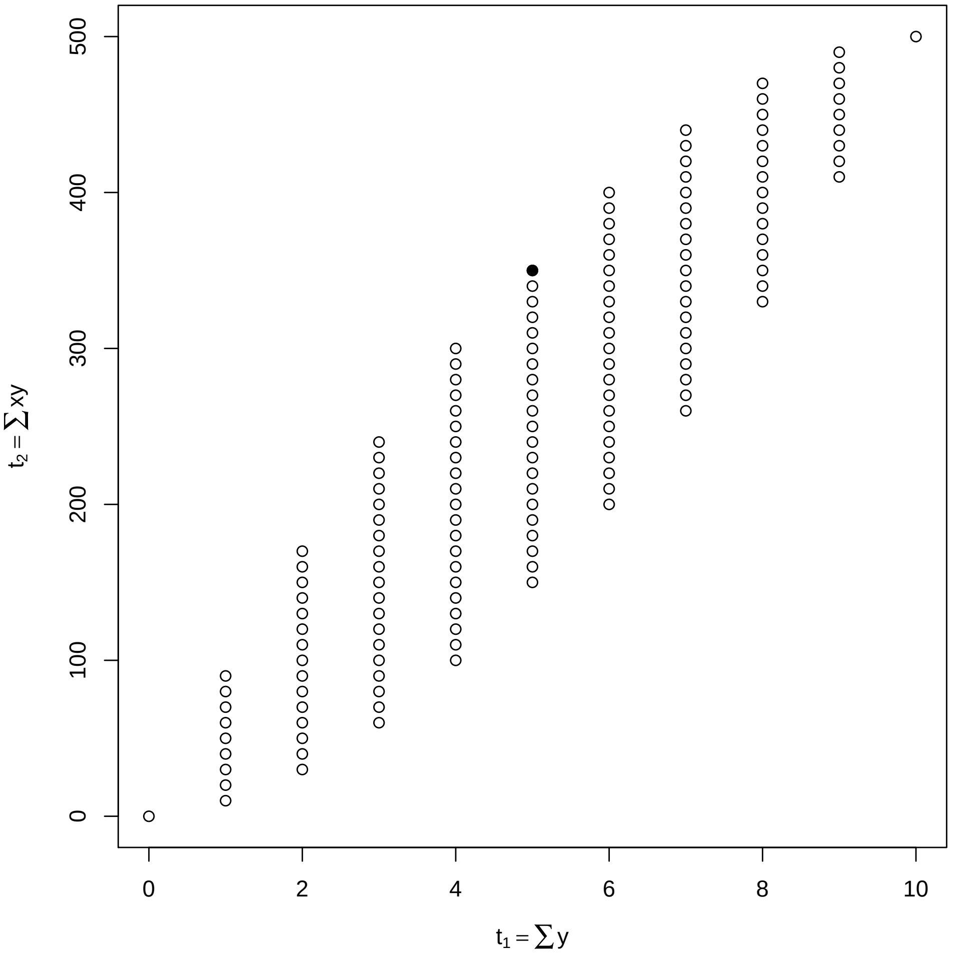

Figure 12: Possible values of the submodel canonical statistic vector $M^T y$ for the data shown in Figure 11. Solid dot is the observed value of the submodel canonical statistic vector.

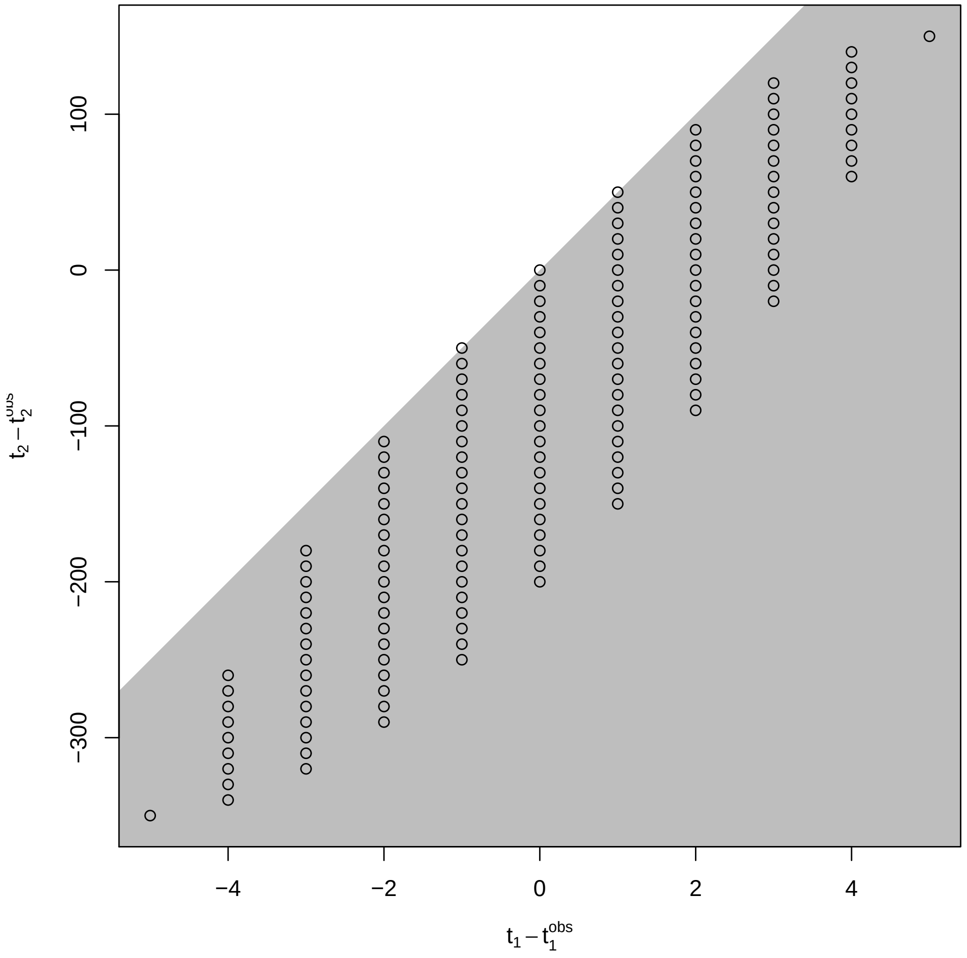

Figure 13: Tangent vectors and tangent cone for data shown in Figure 11. Dots are tangent vectors, gray region is tangent cone.

```
lsub <- as.numeric(tanv[lout, ][1, ])
lsub

## [1]  1 50
```

This partial degeneracy of the MLE distribution is what Agresti calls "quasi-complete separation."

In this calculation we could have made use of the features of the R package `rcdd` that allows us to have V-representations that indicate lines as well as rays. This means that when there are vectors we know are in the linearity (as in this example), we can indicate this to the R function `linearity` and shorten its calculation. There seems no point in explaining this further because the calculation is trivial for this toy example.

4.3 Calculating Generic Directions of Recession

Now, as in Section 3.3 above, we need to calculate a generic direction of recession (GDOR), and now we have to do a different calculation from the one in that section, because it was not general (that calculation was only valid for Example I).

Next we determine the GDOR. The R code now has to account for the linearity not being trivial.

```
p <- ncol(tanv)
hrep <- cbind(0, 0, -tanv, -1)
hrep[lout, 1] <- 1
hrep[lout, p + 3] <- 0
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 <- q2d(pout$primal.solution[1:p])
gdor

## [1] -5.0  0.1
```

```
rownames(hrep) <- NULL
colnames(hrep) <- c("code", "rhs", "eta1", "eta2", "epsilon")
names(objv) <- c("eta1", "eta2", "epsilon")
dimnames(tanv) <- NULL
objv
```

```
##    eta1    eta2 epsilon
##       0       0       1
```

```
hrep
```

```
##       code rhs eta1 eta2 epsilon
##  [1,]    0   0   -1  -10      -1
##  [2,]    0   0   -1  -20      -1
##  [3,]    0   0   -1  -30      -1
##  [4,]    0   0   -1  -40      -1
##  [5,]    0   0    1   60      -1
##  [6,]    0   0    1   70      -1
##  [7,]    0   0    1   80      -1
##  [8,]    0   0    1   90      -1
##  [9,]    1   0   -1  -50       0
## [10,]    1   0    1   50       0
## [11,]    0   1    0    0      -1
```

```
tanv
```

```
##       [,1] [,2]
##  [1,]    1   10
##  [2,]    1   20
##  [3,]    1   30
##  [4,]    1   40
##  [5,]   -1  -60
##  [6,]   -1  -70
##  [7,]   -1  -80
##  [8,]   -1  -90
##  [9,]    1   50
## [10,]   -1  -50
```

```
lout
```

```
## [1]  9 10
```

For once we explain how the arguments to the R function `lpcdd` encode the linear program (3), which, for convenience, we repeat here exactly

$$
\begin{array}{lll}
\text{maximize} & & \\
\quad \varepsilon & & \\
\text{subject to} & & \\
& \varepsilon \leq 1 & \\
& \langle v, \eta \rangle = 0, & v \in L_{\text{sub}} \\
& \langle v, \eta \rangle \leq -\varepsilon, & v \in V_{\text{sub}} \setminus L_{\text{sub}}
\end{array}
$$

The state vector of the linear program is $(\eta_1, \eta_2, \varepsilon)$. The objective function (what is maximized) is the linear function of these variables $\langle \texttt{objv}, (\eta_1, \eta_2, \varepsilon) \rangle$. We want this to be just ε, so we must have

$$\texttt{objv} = (0, 0, 1)$$

The argument `hrep` represents the linear constraints. From the documentation for the R function `lpcdd`

> Let
>
> ```
> l <- hrep[ , 1]
> b <- hrep[ , 2]
> v <- hrep[ , - c(1, 2)]
> a <- (- v)
> ```
>
> Then the convex polyhedron in question is the set of points `x` satisfying
>
> ```
> axb <- a %*% x - b
> all(axb <= 0)
> all(l * axb == 0)
> ```

The first column of the H-representation is a code: zero indicates an inequality constraint, one indicates an equality constraint. The second column is the right-hand-side vector of the constraints. The remaining columns are the coefficient matrix of the constraints. If b is that vector and V is that matrix, then the constraint set is

$$\{\, x : V x \geq b \,\}$$

where the inequalities act coordinatewise and the inequalities are actually equalities when the "code" is one.

We have one row of `hrep` for each row of `tanv` whose index is in `lout` that represents the equality constraint $\langle v, \eta \rangle = 0$. And we have one row of `hrep` for each other row of `tanv` that represents the inequality constraint $\langle v, \eta \rangle \leq -\varepsilon$. And the last row of `hrep` represents the inequality constraint $\varepsilon \leq 1$.

As in Section 3.3, we show the hyperplane that supports the LCM (Figure 14 below).

Unlike the comparable figure for Example I (Figure 4) which shows the LCM concentrated at one point (just one point in the hyperplane), in Figure 14 there are three points in the hyperplane (three points in the support of the LCM).

4.4 Calculating the Maximum Likelihood Estimate

Having found the MLE (for the canonical parameter vector) does not exist, we find the MLE in the Barndorff-Nielsen completion by finding the MLE in the LCM. Following Section 3.14.2 in Geyer (2009b), this is simple.

```
gout.lcm <- glm(y ~ x, family = binomial, subset = lout)
summary(gout.lcm)

##
## Call:
## glm(formula = y ~ x, family = binomial, subset = lout)
##
## 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
```

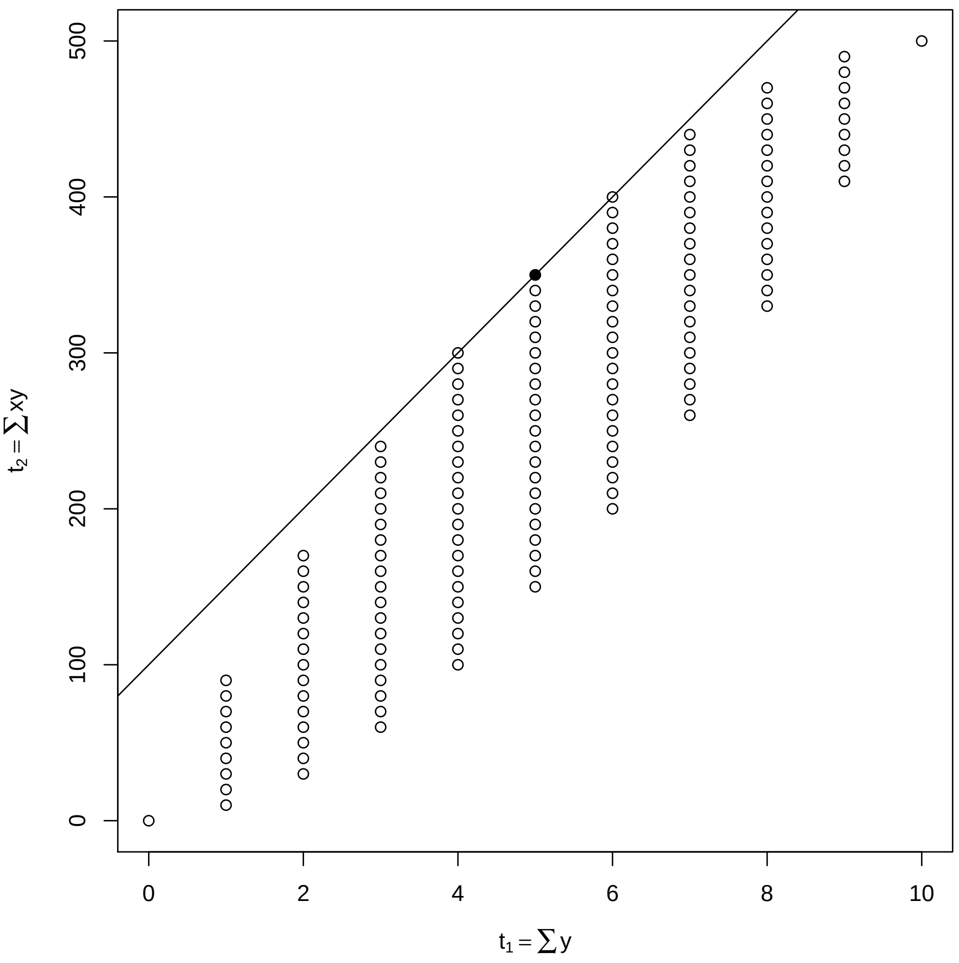

Figure 14: Possible values of the submodel canonical statistic vector $M^T y$ for the data shown in Figure 11. Solid dot is the observed value of the submodel canonical statistic vector. Solid line is the hyperplane (4) on which the LCM is concentrated.

```
##
## Number of Fisher Scoring iterations: 2
```

4.4.1 MLE in LCM, Take 1

For this toy problem, once we understand the principle of limiting conditional models (Geyer, 2009b, Theorem 6), it is obvious what the MLE in the LCM is. The LCM conditions response values for all $x \neq 50$ to be their observed values (because all the data for those x values is at one or the other end point of the range of possible values). The LCM does not restrict in any way response values for $x = 50$. There are two such components of the response vector. Because they have the same predictor value, they have the same mean value parameter value. Hence they are IID Bernoullis whose sum has the Binomial$(2, p)$ distribution, where p is their mean value parameter. The LCM is the model so described (data for $x \neq 50$ conditioned on the observed values, data for $x = 50$ essentially binomial with sample size two).

The MLE in the LCM is thus the sample proportion for $x = 50$, which is $\hat{p} = 0.5$. The MLE mean value parameter for $x = 50$, "predicts" equally likely success and failure (the reason "predicts" is in scare quotes is that this is just a point estimate, the MLE of a parameter, there is no sense of predicting future data).

4.4.2 MLE in LCM, Take 2

Now, instead of using our intuition, we want to read what the R functions `glm` and `summary.glm` have done, because for non-toy data, they are what we have to use.

The output of `summary.glm` reports only one point estimate, for the `"(Intercept)"` parameter. It says `NA` for the other (slope) parameter. The `NA` means it dropped the corresponding predictor from the model, but this is the same as constraining that coefficient to be zero. So wherever any method of the R generic function `summary` prints `NA` in the column labeled `Estimate` of the `Coefficients` table of its printout, we should read zero (not `NA`). Of course, since this zero is not an estimate but a constrained value, it is not random, so we do want `NA` for the rest of the entries in this row: there is no standard error, test statistic, or p-value for this parameter.

So R is saying that the point estimate of the saturated model canonical parameter for the LCM data (all of which have $x = 50$ in this toy problem) is

$$\theta = \beta_1 + \beta_2 \cdot x$$

with $\hat{\beta}_1 = 0$ (actually the computer says $4.7102774 \times 10^{-16}$ but we know that is just inexactness of computer arithmetic) and $\beta_2 = 0$ by constraint. Thus the estimate of the canonical parameter for $x = 50$ is $\hat{\theta} = 0$ and the corresponding mean value parameter estimate is $\hat{p} = \text{logit}^{-1}(0) = 0.5$.

So we arrive at the same result. The computer agrees with out intuition. Our two takes agree.

4.5 Hypothesis Tests

We do the hypothesis tests of Section 3.5 for these data.

Wilks test (likelihood ratio test)

```
gout.0 <- glm(y ~ 1, family = binomial)
gout.1 <- glm(y ~ x, family = binomial)

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

anova(gout.0, gout.1, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: y ~ 1
## Model 2: y ~ x
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1         9    13.8629
## 2         8     2.7726  1    11.09 0.0008678 ***
## ---
## Signif. codes:
## 0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

Rao test (score test)

```
add1(gout.0, ~ x, test = "Rao")

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

## Single term additions
##
## Model:
```

```
## y ~ 1
##        Df Deviance     AIC Rao score Pr(>Chi)
## <none>     13.8629 15.8629
## x       1   2.7726  6.7726    6.6667 0.009823 **
## ---
## Signif. codes:
## 0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

The actual numbers are different, but the conclusions are the same. The only model for which the MLE exists in the conventional sense does not fit the data. The slope parameter is statistically significantly different from zero.

4.6 Confidence Regions

The confidence regions explained and illustrated in Section 3.6 above are inadequate when the LCM is not completely degenerate.

We can think of a completely degenerate LCM as providing no information about the parameters (since all parameter values correspond to the same distribution in the LCM in the completely degenerate case). When the LCM is not completely degenerate, it does provide information about the parameters.

The confidence regions proposed by Geyer (2009b) that are derived by inverting the hypothesis test with test statistic $\langle Y, \eta \rangle$, where η is a GDOR, tell us how close the canonical parameters are to infinity and how close the mean value parameters are to the relative boundary of the convex support. They do not use any information provided by the LCM.

Geyer (2009b) proposes combining the "usual" confidence intervals based on the LCM with the new proposal explained and illustrated in Section 3.6 above. But no examples of this are given in that paper. Of course, there are a multitude of "usual" confidence intervals, those based on inverting the Wilks, Rao, and Wald tests for a start. So which one of these we might choose depends on our whims (all are asymptotically equivalent, none are good for small sample sizes, opinions based on simulations are worthless).

For simplicity, let us call the confidence regions and confidence intervals based on inverting the hypothesis test with test statistic $\langle Y, \eta \rangle$, the set of parameter values satisfying (5) above, "new," and the "usual" confidence regions and confidence intervals "old."

4.6.1 For Submodel Canonical Parameters

Because of the way Agresti constructed our toy problems, the "new" confidence regions for both are exactly the same because the data involved in the event $Y \in H$ are exactly the same, in both we have the same response values for predictor values $x \neq 50$, and the event $Y \in H$ is the event that these response values equal their observed values. Thus Figure 5 shows the "new" confidence region for both Examples I and II.

For no particular reason other than that they are the most widely used, let us take Wald intervals for the LCM. We find in the output of `summary.glm`

```
sout <- summary(gout.lcm)
sout$coefficients[1, 1]

## [1] 4.710277e-16

sout$coefficients[1, 2]

## [1] 1.414214

old.ci.low <- sout$coefficients[1, 1] - qnorm(0.975) * sout$coefficients[1, 2]
old.ci.hig <- sout$coefficients[1, 1] + qnorm(0.975) * sout$coefficients[1, 2]
old.ci.low

## [1] -2.771808

old.ci.hig

## [1] 2.771808
```

Those are the endpoints of our "old" 95% confidence interval for β_1. Recall that β_2 is constrained to be zero in the LCM (it is not unknown). But we do have a direction of constancy of the LCM. It is the GDOR of the OM. (When only one parameter is "not defined because of singularities" as `summary.glm` says, the GDOR of the OM is the only direction of constancy of the LCM. When there are more such parameters, there are more directions of constancy of the LCM.) Hence our confidence region in the original parameter space consists of all points of the form

$$(\beta_1, 0) + s(\eta_1, \eta_2),$$

where η is the GDOR, s is any real number, and β_1 is in our "old" confidence interval having endpoints calculated above.

The two confidence regions shown in Figure 15 are not adjusted for simultaneous coverage, and we may not want them to be. It depends on what we are doing. For example, if we want to make confidence intervals for the two submodel canonical parameters (like those we made in Section 3.6.2 above). The "old" confidence region does not restrict those parameters at all: from it we would get the interval $(-\infty, +\infty)$ for both β_1 and β_2. Hence we should (for this task) ignore the "old" confidence region entirely, and just base intervals for β_1 and β_2 on the "new" region. But since the "new" region is the same for both toy models (Example I and Example II), we get the same confidence intervals for β_1 and β_2 as we got in Section 3.6.2.

If we do want to make a valid confidence region out of these two confidence regions, then we do have to make some correction to get simultaneous coverage. The obvious thing to do is Bonferroni correction. But perhaps there are better corrections.

4.6.2 A Digression on Faces of a Convex Set

Under a certain regularity condition called "a condition of Brown (1986)" in Geyer (2009b), the convex supports of the limiting conditional models of an exponential family are the faces of the convex support of the original model. This gives a language to talk about all of the limiting conditional models, which together make up the Barndorff-Nielsen completion of the original model.

A subset F of a convex C is a *face* of C if whenever x and y are points of C and the line segment between them

$$\{\, x + s(y - x) : 0 < s < 1 \,\}$$

contains a point of F, then actually $x, y \in F$. Vacuously, this definition both the empty set and C itself faces of C. They are called *improper faces.* All other faces are *proper faces.*

A subset F of a convex C is an *exposed face* of C if F is the locus of points where some affine function achieves its maximum over C. It is easy to see that (as the name suggests) every exposed face is a face. Consider a convex set C and an affine function f and define

$$\begin{aligned} m &= \sup_{x \in C} f(x) \\ F &= \{\, x \in C : f(x) = m \,\} \end{aligned}$$

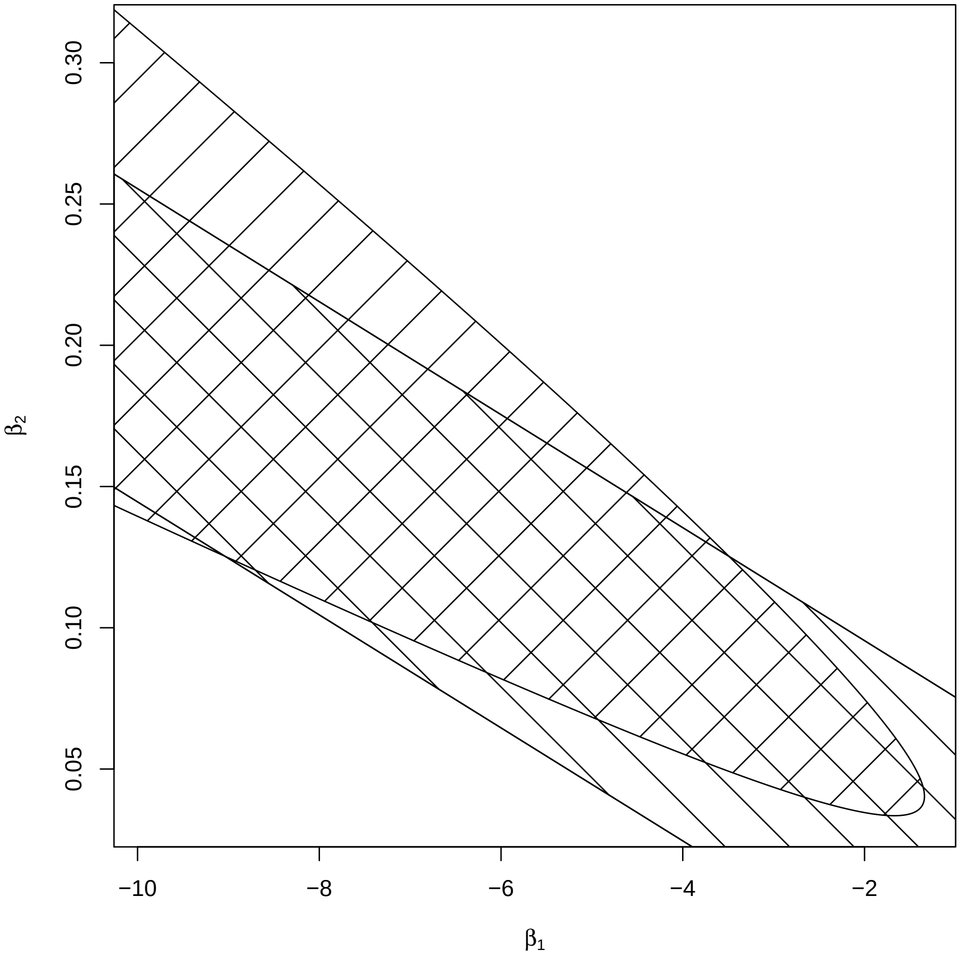

Figure 15: 95% Confidence Regions for Submodel Canonical Parameter Vector β (a. k. a., Coefficients) for Example II. Region with curved boundary is “new” region based on (5). Region with straight boundary is “old” region based on Wald confidence interval for LCM.

(it may be that $m = +\infty$ in which case F is the empty set).

Now consider x and y in C and $0 < s < 1$ such that $x + s(y - x) \in F$. (If there are no such x, y, and s, then F is vacuously a face of C.) An affine function is both convex and concave. To see this consider the definition of affine function (Section 8.3 of Geyer, 2016a), which says f is an affine function if for any point x

$$g(v) = f(x + v) - f(x)$$

defines a linear function g on the translation space of the affine space on which f is defined. For $0 < s < 1$ we have

$$sg(v) = g(sv) = f(x + sv) - f(x)$$

so

$$f(x + sv) = f(x) + sg(v) = f(x) + s\bigl[f(x + v) - f(x)\bigr]$$

and, introducing $y = x + v$ makes this look like the convexity equality

$$f\bigl(x + s[y - x]\bigr) = f(x) + s\bigl[f(y) - f(x)\bigr]$$

except for the inequality being replaced by equality. Hence both f and $-f$ satisfy the convexity inequality. Now, returning to our proof about exposed faces, we know $f(x) \leq m$ and $f(y) \leq m$ and $f(x + s[y - x]) = m$, so

$$m = (1 - s)f(x) + s(f) \leq m$$

can only be satisfied with equality if $f(x) = f(y) = m$, and that implies $x, y \in F$ and hence that F is a face of C.

Not every face is an exposed face. The simplest example is when C is the union of a disk and a square such that one side of the square is a diameter of the disk Figure 16 shows this.

Now consider a regular full exponential family having convex support K. We will assume

(i) Every face of K is exposed.

(ii) Every face of K has positive probability.

(iii) For every direction η in the canonical parameter space, the convex support of the limiting conditional model taking limits in the direction η is the face of K where the affine function $y \mapsto \langle y - z, \eta \rangle$ achieves its maximum over K.

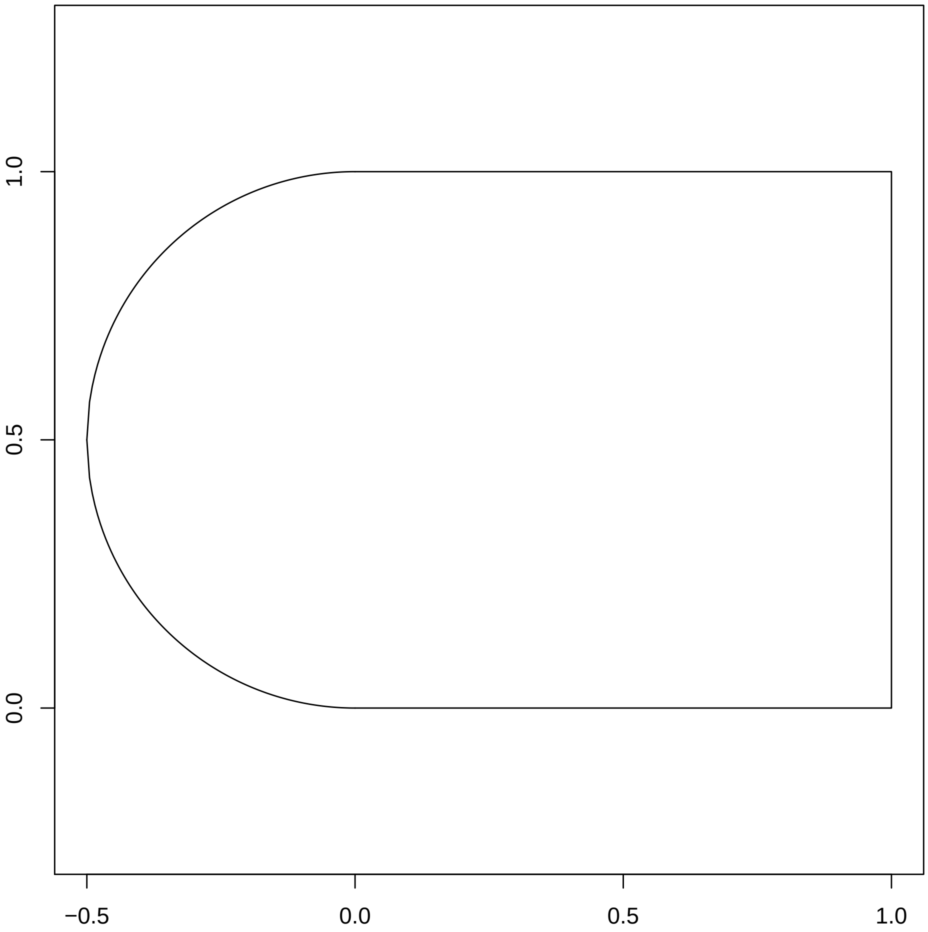

Figure 16: Convex Set having Non-Exposed Faces. The points $(0, 0)$ and $(0, 1)$ are non-exposed faces of the convex set whose boundary is the solid line.

(iv) Every such LCM, including the OM (the case $\eta = 0$) is a regular full exponential family.

These are similar to, but slightly different from, the assumptions under which Geyer (2009b) works (those assumptions are summarized in Section 3.7 of that paper).

We now comment on the assumptions in the list above. Assumption (ii) rules out convex supports with more than a countable number of faces. This assumption does not quite imply Assumption (i). Consider the set of points in $\mathbb{R}^2$ given by

$$S = \{(0,1), (1,1)\} \cup \{\, (\cos(2/n), -\sin(2/n)), n = 1, 2, \ldots \,\}$$

Figure 17 shows $K = \operatorname{con}(S)$.

The full standard exponential family generated by any finite measure that gives positive measure to every point of S satisfies Assumption (ii) but not Assumption (i) because the point $(0,1)$ is a nonexposed face of K.

An example where Assumption (iii) fails is the two-dimensional example with support shown in Figure 18. Consider the two-dimensional full exponential family generated by the measure λ that puts mass $\exp(-k^2)$ at the point $(k,0)$ for all integers k and mass 1 at the points $(0,1)$ and $(1,1)$. The convex support is the infinite strip

$$K = \{\, (x,y) \in \mathbb{R}^2 : 0 \le y \le 1 \,\}.$$

It has three, nonempty faces: K itself, and the lines

$$\begin{aligned} F_0 &= \{\, (x,0) : x \in \mathbb{R} \,\} \\ F_1 &= \{\, (x,1) : x \in \mathbb{R} \,\} \end{aligned}$$

Now F_0 is the convex support of the LCM conditioned on the event $Y \in F_0$, but F_1 is not the convex support of the LCM conditioned on the event $Y \in F_1$. The latter conditional model has support $\{(0,1), (1,1)\}$ and convex support the line segment between these two points, which is not all of F_1.

This model does not have limiting conditional models in one-to-one correspondence with faces of K. The LCM conditioned on the event $Y \in F_1$ has two further LCM, which are the completely degenerate models concentrated at the points $(0,1)$ and $(1,1)$.

So when Assumption (iii), which Geyer (2009b) calls the "condition of Brown (1986)," fails we do not have a one-to-one correspondence between limiting conditional models and nonempty faces of the convex support K.

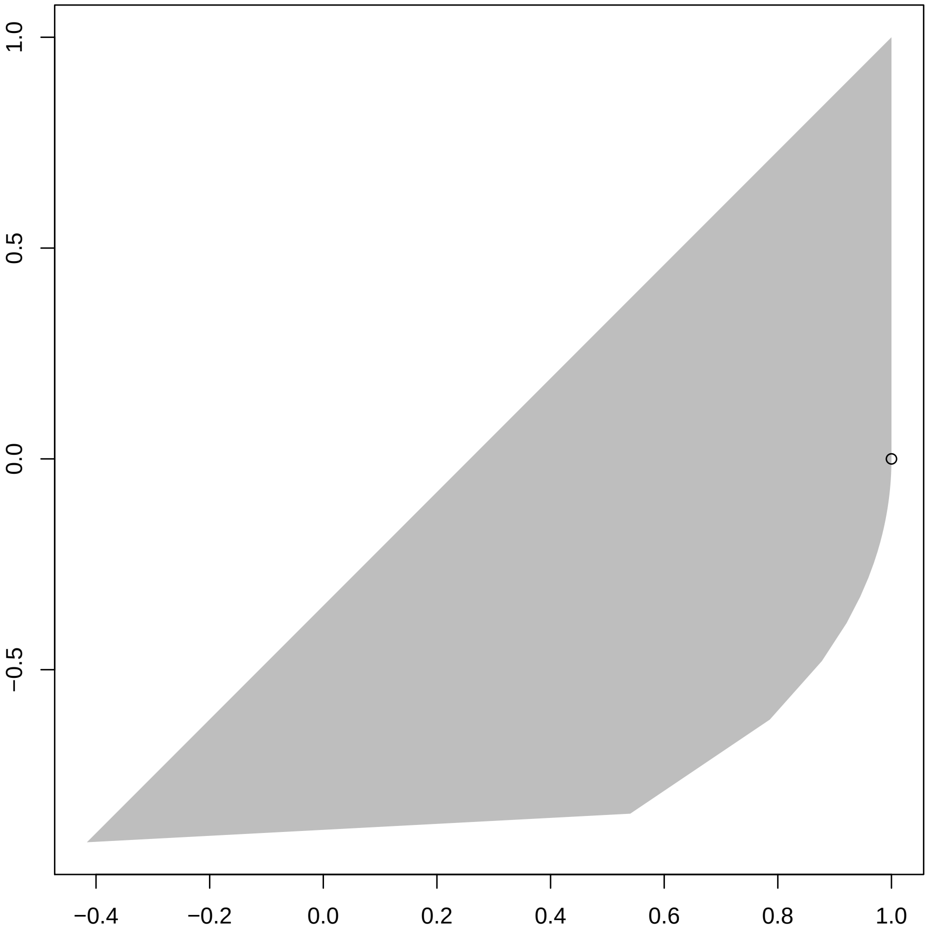

Figure 17: Convex Support with Countable Number of Faces and One Non-Exposed Face. Small circle is centered at the non-exposed face, which is a single point.

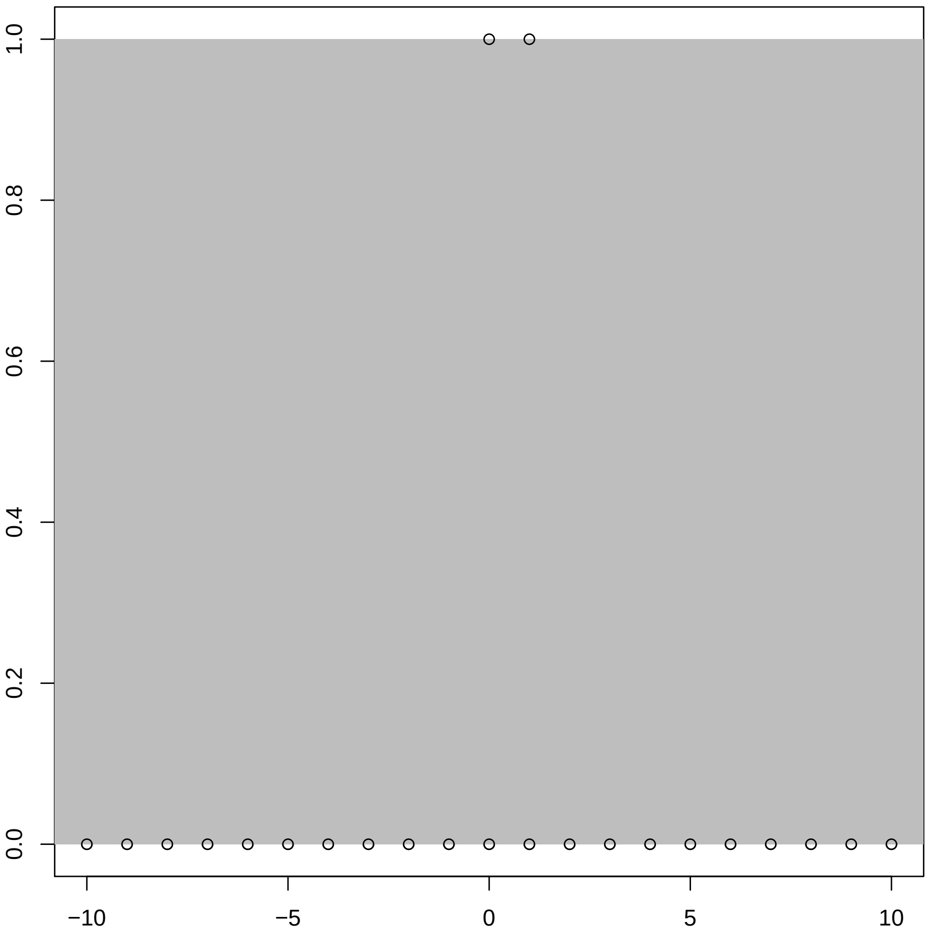

Figure 18: Support of Exponential Family for which the Condition of Brown Fails. The family is discrete and point of the support are shown as circles. The bottom row extends infinitely far in each direction. The gray area is the convex support. It two extends infinitely far to the left and right.

The main reason we are assuming this assumption is to get this one-to-one correspondence.

In order to have this one-to-one correspondence we need to associate an LCM with K itself (because K is an improper face of itself). And it is clear that the model having that support is the OM, which we also consider to be the LCM when limits are taken in the direction $\eta = 0$. (That is, the limits are trivial limits of a constant sequences).

We could drop Assumption (iv) if we were willing to deal with constrained optimization. Chapter 2 of Geyer (1990) does this. But since many models of practical interest satisfy all four of these assumptions, it seems harmless to assume them here. When Assumption (iv) does not hold (Geyer and Thompson, 1992; Geyer and Møller, 1994), the discussion here can be easily modified to cover that case.

A comment on Assumption (iv). The parameter space of the OM may need to be enlarged to make the parameter space of an LCM full. A simple example of this is the geometric distribution, which has canonical parameter $\theta = \log(1-p)$ The OM has canonical parameter space $(-\infty, 0)$. If we take the limit in the direction -1 (that is as the canonical parameter goes to minus infinity), we get the completely degenerate model, whose only distribution is concentrated at zero (the lower endpoint of the convex support of the OM). But this LCM must have canonical parameter space $(-\infty, \infty)$ to be full.

In every case where this "enlargement" is necessary, it does not "enlarge" the LCM in the sense of adding probability distributions to it that were not already in it; what is does is add canonical parameter values that were not already in it. If Γ_{LCM} denotes the constancy space of the LCM and Θ_{OM} and Θ_{LCM} the full canonical parameter spaces of the OM and LCM, respectively, then

$$\Theta_{\texttt{LCM}} = \Gamma_{\text{LCM}} + \Theta_{\text{OM}}. \tag{11}$$

4.6.3 Return to Confidence Regions

We continue assuming the assumptions of the preceding section. If K is the convex support of the OM and $\mathcal{F}$ is the set of nonempty faces of K (one of which is K itself), then every LCM of the OM has the form

$$\{\, f_\theta(\,\cdot\,|Y \in F) : \theta \in \Theta \,\} \tag{12}$$

for some $F \in \mathcal{F}$, where Θ is the canonical parameter space of the OM. But we may need to "enlarge" the canonical parameter space of the LCM using (11) to obtain a full family.

Recall the concept of relative interior of a convex set (Geyer, 2016a, Section 8.15). It is an important fact about convex sets that the relative interiors of nonempty faces of a convex set partition it (Rockafellar, 1970, Theorem 18.2). Thus

$$\{ \operatorname{ri} F : F \in \mathcal{F} \}$$

is a partition of K.

Let $\widehat{F}(y)$ denote the face of the convex support K that is the support of the LCM in which the MLE for observed data y exists. Then $\widehat{F}(y)$ is the unique face of K containing y in its relative interior (unique because of the partitioning theorem just mentioned). The reason for this is that if y is on the boundary of a face F, then there is a direction of recession in the LCM having support F, which is (12), that direction of recession being the η that determines the affine function that is maximized over K at points of F. So $y \in \operatorname{ri} \widehat{F}(y)$.

Now we can get a better, more complete, more formal view of our confidence region proposals. The "new" confidence regions are determined by $\widehat{F}(Y)$, which is a random element of $\mathcal{F}$, and this is a discrete random thingummy under Assumption (ii) of the preceding section. The "old" confidence regions are determined by the distribution of the response given $\widehat{F}(Y)$, that is, by the LCM having $\widehat{F}(Y)$ as its convex support. Since the $\operatorname{ri} \widehat{F}(y)$ partition K, we have a factorization of the "joint" distribution of the response vector (with "joint" in scare quotes because it isn't the joint distribution of two random thingummies) factored into the marginal of the random thingummy $\operatorname{ri} \widehat{F}(Y)$, and the conditional of Y given $\operatorname{ri} \widehat{F}(Y)$. The "new" confidence regions depend only on this marginal, and the "old" confidence regions depend only on this conditional.

If y is itself discrete, then we can factor its probability mass function (PMF) as

$$f_\theta(y) = f_\theta(y \mid Y \in \operatorname{ri} F) f_\theta(Y \in \operatorname{ri} F), \qquad y \in K, \ F \in \mathcal{F}, \ \theta \in \Theta,$$

and if y is continuous, we can do the same with its probability density function (PDF). And if y is neither discrete nor continuous, we can get the same factorization using regular conditional probability from measure theory.

Our "new" confidence regions described in Section 3.6 above can be rewritten using our new notation as

$$\left\{ \theta \in \Theta : \operatorname{pr}_\theta\bigl(Y \in \widehat{F}(y)\bigr) \geq \alpha \right\}.$$

and we see that the probability here is almost but not quite the marginal probability in the factorization described above. There are two reasons we use $\widehat{F}(y)$ here rather than its relative interior. First, this notion of "new" confidence intervals grew slowly. They were much less formal in the first draft of Geyer (2009b), see Section 3.16 of the supporting technical report Geyer (2008) for more or less what this first draft said. Even the published version of Geyer (2009b) was not as formal as we are here. Second, it is much more work to calculate relative interiors of faces than faces. The computing that needs to be done to carry out the recommendations of Geyer (2009b) is chosen to be very efficient computational geometry. (Even this "very efficient" computation can be annoyingly slow, many seconds for non-toy problems.) The R function `allfaces` in the R package `rcdd` finds all the faces of a polyhedral convex set, but this takes much more time than the calculations using the R functions `linearity` and `lpcdd` whose use for calculating MLE in the Barndorff-Nielsen completion are illustrated above and in the technical reports supporting Geyer (2009b). (And "much more time" can be far longer than anyone is willing to wait for an answer.) We really do not want to have to calculate all faces and their relative interiors if we do not have to. They are nice and elegant for theoretical discussion, but they are a practical horror. That is why Geyer (2009b) completely avoids the notion of faces of a convex set, except when comparing its theory with pre-existing theory (Barndorff-Nielsen, 1978; Brown, 1986).

In Example I above, we have $\widehat{F}(y) = \{y\}$ (the LCM is concentrated at the single point y) and the relative interior of a singleton set is that set itself. So we also have $\operatorname{ri} \widehat{F}(y) = \{y\}$.

In Example II above, we have $\widehat{F}(y)$ consisting of three points, which are the three points on the line (the hyperplane H) in Figure 14 above. As discussed in the text near that figure, these three points are where all components of the response vector are fixed at their observed values except for the two components corresponding to the predictor value $x = 50$ (and those components are random, so either can have the value zero or one, which seems to be four points, but the canonical statistic does not distinguish one being zero and the other being one or the reverse). Essentially, we have a binomial distribution with sample size two: the components of Y corresponding to $x = 50$ have zero, one, or two successes. As we can see in Figure 14 above, the three points that support the LCM (for the observed data) are in a line so the middle one is in the relative interior and the other two are on the relative boundary. Thus for Example II we have $\operatorname{ri} \widehat{F}(y)$ is a proper subset of $\widehat{F}(y)$ and we have

$$\operatorname{pr}_\theta\bigl(Y \in \operatorname{ri} \widehat{F}(y)\bigr) = 2 \operatorname{pr}_\theta(Y = y)$$

because for the observed data vector y we have one success corresponding to $x = 50$ and the other possible data vector having the same values of the canonical statistics has the same probability. This in this particular toy problem we have no trouble calculating $\operatorname{ri} \widehat{F}(y)$, but the computing difficulty could limit our ability to do this in real applications. So we continue with the "new" proposal of Geyer (2009b), but we can see how much it loses by not doing exactly the right thing (TRT) by doing TRT for Example II.

We obtain a valid (corrected for simultaneous coverage) confidence region by inverting a valid "omnibus" test (corrected for multiple testing), that is, the confidence region consists of θ_0 accepted at level α for the "omnibus" test. This "omnibus" test accepts θ_0 at level α when both the "new" and "old" tests accept. That is we accept with probability

$$\begin{aligned}
&\sum_{F \in \mathcal{F}} \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} F) \operatorname{pr}_{\theta_0}\bigl(\text{“new” and “old” c. i. cover } \theta_0 \bigm| Y \in \operatorname{ri} F\bigr) \\
&\quad = \sum_{\substack{F \in \mathcal{F} \\ \operatorname{pr}_{\theta_0}(Y \in F) \ge \alpha_{\text{new}}}} \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} F) \operatorname{pr}_{\theta_0}\bigl(\text{“old” c. i. covers } \theta_0 \bigm| Y \in \operatorname{ri} F\bigr) \\
&\qquad \approx (1 - \alpha_{\text{old}}) \sum_{\substack{F \in \mathcal{F} \\ \operatorname{pr}_{\theta_0}(Y \in F) \ge \alpha_{\text{new}}}} \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} F)
\end{aligned}$$

where α_{new} and α_{old} are the significance levels for the test inverted to make the "new" and "old" confidence intervals, respectively, and the $\approx$ comes from our "old" confidence intervals being only approximate.

But this is wrong. It does not special case $F = K$ where the "new" region is simply the whole parameter space, because $\operatorname{pr}_{\theta_0}(K) = 1$ whatever the value of θ_0. Nor does it special case when F is a singleton set (a vertex of K) where the "old" region is simply the whole parameter space, whatever "old" procedure we use, because then the LCM is completely degenerate and its constancy space is the whole space. Hence let $\mathcal{V}$ denote the vertices of K ($\mathcal{V}$ is a subset of $\mathcal{F}$, a proper superset unless K is a singleton set, which is uninteresting). Then we want to use the desired coverage $1 - \alpha$ instead of $1 - \alpha_{\text{new}}$ when $F = K$, and we want to use (as we did in Example I) the desired significance level α (one minus the coverage probability) when

$F \in \mathcal{V}$. Then our coverage is

$$
\begin{aligned}
\operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} K) \operatorname{pr}_{\theta_0}\bigl(\text{“old” c. i. covers } \theta_0 \bigm| Y \in \operatorname{ri} K\bigr) \\
+ \sum_{\substack{V \in \mathcal{V} \\ \operatorname{pr}_{\theta_0}(Y \in V) \ge \alpha}} \operatorname{pr}_{\theta_0}(Y \in V) \\
+ \sum_{\substack{F \in \mathcal{F} \setminus \mathcal{V} \setminus \{K\} \\ \operatorname{pr}_{\theta_0}(Y \in F) \ge \alpha_{\text{new}}}} \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} F) \operatorname{pr}_{\theta_0}\bigl(\text{“old” c. i. covers } \theta_0 \bigm| Y \in \operatorname{ri} F\bigr) \\
\approx (1 - \alpha) \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} K) \\
+ \sum_{\substack{V \in \mathcal{V} \\ \operatorname{pr}_{\theta_0}(Y \in V) \ge \alpha}} \operatorname{pr}_{\theta_0}(Y \in V) \\
+ (1 - \alpha_{\text{old}}) \sum_{\substack{F \in \mathcal{F} \setminus \mathcal{V} \setminus \{K\} \\ \operatorname{pr}_{\theta_0}(Y \in F) \ge \alpha_{\text{new}}}} \operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} F)
\end{aligned}
$$

(the reason why $Y \in V$ rather than $Y \in \operatorname{ri} V$ when V is a vertex is that then $V = \operatorname{ri} V$). Even with this modification, it is hard to see what is going on with the “new” procedure (even for me). But some special cases are easy to understand.

The “usual” asymptotics of maximum likelihood requires

$$\operatorname{pr}_{\theta_0}(Y \in \operatorname{ri} K) \approx 1, \tag{13}$$

although that isn't the regularity condition stated in theorems about asymptotic normality of the MLE. What conventional theorems say is θ_0 is in the interior of the parameter space, and that does imply (13) but only when n has actually gone almost all the way to infinity. Here we see that n may need to be humongously large for (13) to hold. Our complicated formula gives $\approx 1 - \alpha$ for that case, which is what is desired.

Another special case is when θ_0 is such that

$$\operatorname{pr}_{\theta_0}(Y \in V) \approx 1 \tag{14}$$

Our complicated formula gives ≈ 1 for that case, which is conservative but the best one can do. As we saw when we discussed performance of confidence intervals for discrete data, when looking at the web page on this (Geyer, 2016b), all *any* confidence region procedure can do in this case is ≈ 0 or ≈ 1 and clearly ≈ 1 is better.

But the general formula is very complicated and it is not clear what we get out of it in general. We do see that the simple argument that motivated the "new" procedure proposed in Geyer (2009b) isn't really valid. It's a lot more complicated than that. Nevertheless, we have a problem that needs a solution. Conventional theory (the "usual" asymptotics of maximum likelihood) does not provide *any* solution. The "new" proposal of Geyer (2009b) is the only proposal in the literature AFAIK.

Just in case it is not abundantly clear that the "usual" asymptotics of maximum likelihood is no good for discrete models, we return to the binomial distribution. We know that when $p \approx 0$ and $n \approx \infty$ but np is moderate-sized, which we write $0 \ll np \ll \infty$, then the distribution of X is not approximately normal but rather approximately Poisson. Then all aspects of conventional asymptotics are simply wrong (because they are based on asymptotic normality). Moreover, it is clear that this has nothing to do with the binomial distribution but rather holds for any distribution that is discrete in both the usual measure-theoretic sense (countable support) but also the topological sense (the support has no limit points). Whenever the mean value parameter gets close to the relative boundary of the convex support, the distribution of the canonical statistic cannot be approximately normal and must still be very discrete (having atoms with appreciable probability). The distribution may not be Poisson, which is a very special approximation for the binomial distribution. But it will be discrete. No matter how large the sample size, for mean value parameter vector τ sufficiently close to the relative boundary of K, the distribution of the canonical statistic $M^T Y$ is not only far from normal but also still very discrete.

This is the reason why the "new" proposal of Geyer (2009b) is not based on conventional asymptotics. It just couldn't work if it were. This is the reason why no proposal that wants to compete with the one in Geyer (2009b) can look anything like conventional theory. In particular, likelihood-based confidence regions, mentioned at the end of Section 3.7 above, cannot work unless they are calibrated in some way that recognizes the discreteness of the distribution of the likelihood ratio test statistic in some regions of the parameter space (does not use asymptotic normality in any way).

Thus our discussion in this section is ultimately unsatisfactory, unless the reader is looking for an open research question. The "new" proposal of Geyer (2009b) is undeniably a thing to do (TTD) and is AFAICS the only TTD in the literature, except for the special case of the binomial distribution, which has been beaten to death in the literature, see Geyer and Meeden (2005) and its discussion and all the literature cited therein. It is not clear that any of the proposals for confidence intervals for the binomial distribution or for any one-dimensional exponential family (all of which Geyer and Meeden, 2005, handle, at least in principle) generalizes to two or more dimensions.

Hence we are left with the current unsatisfactory situation. The "new" proposal of Geyer (2009b) is merely a TTD. The argument given for it is (at best) merely a motivation, not a tight mathematical argument. When we try to give such an argument, as we did above. We see that the details are very complicated. So it is not clear (1) how the proposal of Geyer (2009b) actually works in complicated situations, (2) what, if anything, could do better, and (3) how we could tell what is better except by simulations in toy models, which would *prove* exactly nothing.

So let us move on from this somewhat depressing situation, leave confidence regions, and return to confidence intervals. In the examples in Geyer (2009b) confidence regions were not used on the examples. And all of the confidence interval examples used either the "new" or "old" procedure but not both. Let us just continue with that, not attempting any correction for simultaneous coverage, following Geyer (2009b), who is just following the conventional usage as embodied, for example in the output of the R generic function `summary`, at least the methods of it that are in the R core (for R objects of class `"lm"`, `"glm"`, `"gam"`, and so forth).

4.6.4 For Submodel Mean Value Parameters

This section is just like Section 3.6.3 above except for Example II rather than Example I. As we saw in Section 4.6.1 above, the "new" procedure gives the same confidence region for the submodel canonical parameter vector β for both examples. But the regions for the submodel mean value parameter $\tau = M^T \mu$ will be different, because the mapping $\beta \mapsto \tau$ is different for the two models.

Thus we have to redo this mapping.

```
thetas <- m %*% t(betas)
thetas <- t(thetas)
# as with betas, rows of thetas are parameter vectors
mus <- 1 / (1 + exp(- thetas))
taus <- t(m) %*% t(mus)
taus <- t(taus)
# as with betas, rows of taus are parameter vectors
# in this case submodel mean value parameter vectors
dim(taus)

## [1] 1499    2
```

Making this confidence region is harder than before because we also have to deal with the vertices of the convex support.

```
chout <- chull(t1, t2)
t1.vert <- t1[chout]
t2.vert <- t2[chout]
t1.rint <- mean(t1.vert)
t2.rint <- mean(t2.vert)
```

The following code plots the convex hull (so we can see what we have done)

```
plot(t1, t2, xlab = expression(t[1]), ylab = expression(t[2]))
points(t1.vert, t2.vert, pch=19)
points(t1.rint, t2.rint, pch=23, cex=2)
polygon(t1.vert, t2.vert)
```

Figure 19 shows the output. Unlike in Figure 12 the observed value of the canonical statistic vector is not a solid dot (it is the hollow dot on the upper boundary). We see that we do indeed have the vertices of the convex hull and a relative interior point. Next we look at the angles of the vertices going around the relative interior point.

```
angle <- function(tt, origin = c(t1.rint, t2.rint)) {
    stopifnot(is.numeric(tt))
    stopifnot(is.numeric(origin))
    stopifnot(length(tt) == length(origin))
    foo <- tt - origin
    atan2(foo[1], foo[2] / 10)
}
ang <- apply(cbind(t1.vert, t2.vert), 1, angle)
ang

##  [1]  2.94419709 -3.07502449 -3.03671571 -3.00606494
##  [5] -2.97644398 -2.94419709 -2.89661399 -2.78282198
##  [9] -2.03444394 -0.19739556  0.06656816  0.10487694
## [13]  0.13552771  0.16514868  0.19739556  0.24497866
## [17]  0.35877067  1.10714872
```

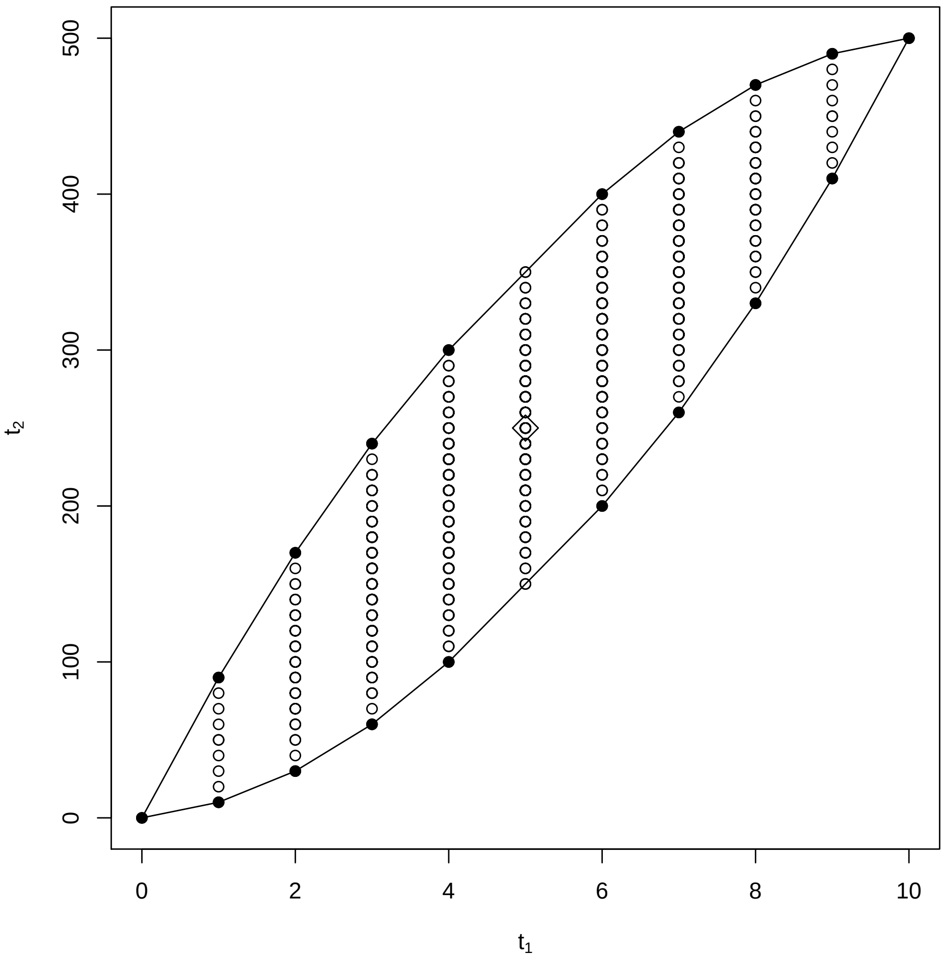

Figure 19: Vertices and Relative Interior Point of Convex Support for Example II. Dots are possible values of canonical statistic vectors. Solid dots are vertices of the convex support. Polygon is boundary of the convex support. Diamond is a relative interior point of the convex support.

Next we find out which vertices of K are between the ends of the curve (which is the R object `taus`).

```
ang.head <- angle(head(taus, 1))
ang.tail <- angle(tail(taus, 1))
ang.head

## [1] 0.1033538

ang.tail

## [1] -1.915444

between <- ang.head > ang & ang > ang.tail
ang[between]

## [1] -0.19739556  0.06656816

t1.vert[between]

## [1] 4 6

t2.vert[between]

## [1] 300 400
```

So the "between" vertices go in the direction from "tail" to "head" and the vertices of the polygon that R needs to draw for the "new" confidence region are

```
poly.new <- rbind(taus, cbind(t1.vert[between], t2.vert[between]))
```

Figure 20 shows this polygon.

Now we try to figure out the "old" confidence region mapped to the submodel mean value parameter scale.

```
ss <- seq(-5, 5, length = 1001)
ss <- ss^3
betas.low <- sweep(outer(ss, gdor), 2, c(old.ci.low, 0), "+")
betas.hig <- sweep(outer(ss, gdor), 2, c(old.ci.hig, 0), "+")
# map to theta (saturated model canonical parameter)
```

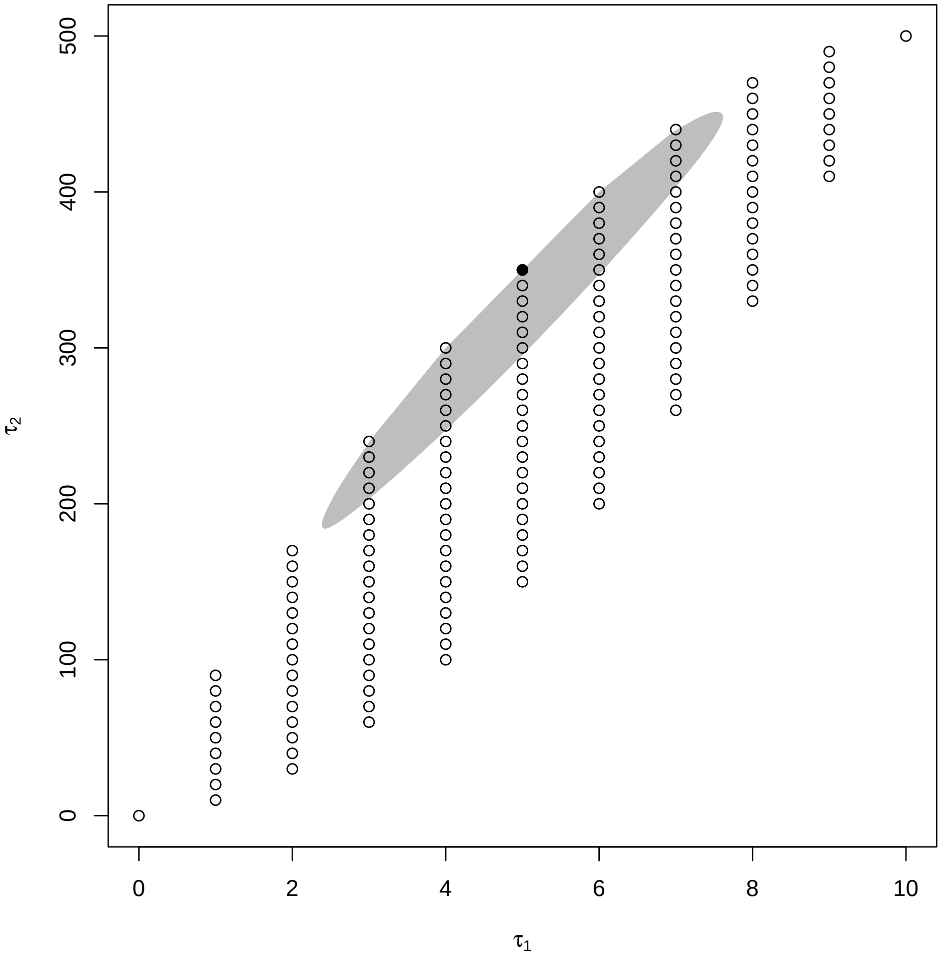

Figure 20: 95% “New” Confidence Region (shaded gray) for Submodel Mean Value Parameter for Example II. Dots are as in Figure 12 (hollow dots are possible values of the submodel canonical statistic, solid dot is the observed value.

```
thetas.low <- betas.low %*% t(m)
thetas.hig <- betas.hig %*% t(m)
# map to mu (saturated model mean value parameter)
mus.low <- 1 / (1 + exp(- thetas.low))
mus.hig <- 1 / (1 + exp(- thetas.hig))
# map to tau (submodel mean value parameter)
taus.low <- mus.low %*% m
taus.hig <- mus.hig %*% m
```

And now we have to figure out what part of the boundary vertices are between the endpoints of these curves.

```
ang.low <- angle(head(taus.low, 1))
ang.hig <- angle(head(taus.hig, 1))
ang.low

## [1] -3.080449

ang.hig

## [1] 2.985018
```

This is tricky. Note that we have passed the branch cut of R's implementation of the arc cosine function (`acos`). It's angles (in radians) go from $-\pi$ to π with $-\pi$ and π representing the same angle. Thus we are starting here just a bit below $-\pi$ and finishing just above, which is the same as just below π. So our notion of "between" is tricky in the same way.

```
between <- ang < ang.low | ang.hig < ang
sum(between)

## [1] 0
```

There are no vertices between these endpoints of these curves. So check the other ends.

```
ang.low <- angle(tail(taus.low, 1))
ang.hig <- angle(tail(taus.hig, 1))
ang.low

## [1] -0.1565748
```

```
ang.hig

## [1] 0.06114401

between <- ang.hig > ang & ang > ang.low
sum(between)

## [1] 0
```

No vertices between at the other end either. In hindsight, this is not surprising. The example is symmetric with respect to interchange of success and failure and left and right.

And the vertices of the polygon that R needs to draw for the "old" confidence region are

```
poly.old <- rbind(taus.low, taus.hig[nrow(taus.hig):1, ])
```

Let's check this (Figure 21). This confidence region is humongous. It rules out almost no mean value parameters! Maybe we don't want this at all. More on this later.

4.6.5 For Saturated Model Mean Value Parameters

Our "old" procedure does say something about mean value parameters corresponding to $x = 50$. I think. Let's check. If B is the "old" confidence interval for β_1, then the "old" confidence region is (as we said above)

$$(\beta_1, 0) + s(\eta_1, \eta_2), \qquad \beta_1 \in B, \; s \in \mathbb{R},$$

where η is the GDOR. The saturated model canonical parameters thought of as a function of x are

$$\theta(x) = \beta_1 + x\beta_2$$

so the thetas corresponding to the confidence region and $x = 50$ are

$$\beta_1 + s\eta_1 + 50s\eta_2 = \beta_1 + (-5) \cdot s + 0.1 \cdot 50 \cdot s = \beta_1$$

when we plug in the actual GDOR. So the confidence interval for the mean value parameter for $x = 50$ is just the confidence interval for β_1 mapped to the mean value parameter scale, which is (0.0588668, 0.9411332).

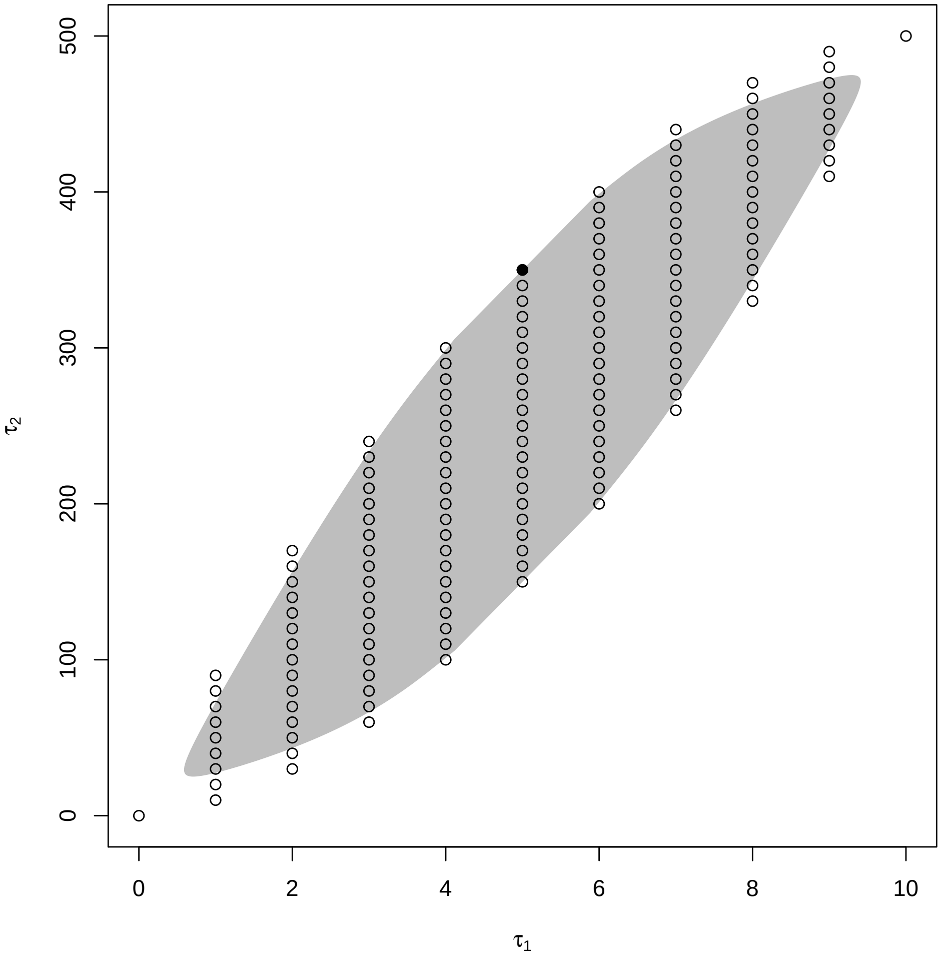

Figure 21: 95% “Old” Confidence Region (shaded gray) for Submodel Mean Value Parameter for Example II. Dots are as in Figure 12 (hollow dots are possible values of the submodel canonical statistic, solid dot is the observed value.

But for x not exactly 50 we get the “new” interval $(-\infty, +\infty)$. Doing the same calculation above but for general x we get a “new” confidence interval for $\theta(x)$ that is

$$\{\, \beta_1 + s\eta_1 + sx\eta_2 : \beta_1 \in B,\ s \in \mathbb{R} \,\}$$

and just to check the exact GDOR (calculated with infinite-precision rational arithmetic) is

```
pout$primal.solution[1:p]

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

so our confidence interval is

$$\{\, \beta_1 + s(x - 50) : \beta_1 \in B,\ s \in \mathbb{R} \,\}.$$

We get a finite “old” confidence interval for $\theta(x)$ only for $x = 50$. For $x \neq 50$, we get, as we said above, $(-\infty, +\infty)$.

Thus we see that these “old” confidence intervals, based on the LCM, are useless except for certain specific predictor values. They may have some use when all predictors are categorical, as in, for example, the second and third examples in Geyer (2009b). But if one truly thinks a predictor quantitative, and new individuals to “predict” will have predictor values different from those in the “training” data, then these “old” confidence intervals are completely useless.

It is hard to know what lessons to draw from a toy problem. But if we think that x is a quantitative predictor for Agresti’s toy problems, then we should think that “old” intervals are entirely useless and that Figures 9 and 10 describe Example II just as well as they describe Example I.

5 Discussion

5.1 New and Old

The results of the preceding section came as a shock to your humble author. Never having tried out predicting for “new” individuals not in the “training” data before, it had simply never occurred to me that “old” confidence intervals based on the LCM would be entirely useless when predictors are quantitative. It had never even occurred to me that qualitative and quantitative predictors would behave differently.

So the “old” confidence intervals may be of use when all “predictors” are categorical, that is, for what is usually called “categorical data analysis via log-linear models." But they are no use for typical regression and classification problems (where there are quantitative predictors).

So this tells that all of our worries about correcting "old" and "new" regions for simultaneous coverage was mostly useless. This could only be a problem if one is fixated on confidence regions for β, which is the most meaningless parameter. As we have seen these "old" regions and intervals tell us almost nothing about the more meaningful parameters τ, $\mu(x)$, and $\theta(x)$.

And if we don't use the "old" procedures, then we don't need to correct the "new" procedures. I would further claim that if we only use the "old" procedures for certain predictor values and the "new" procedures for other predictor values, then we do not need any correction for simultaneous coverage either. For Example II here. We use the "new" procedures for $x \neq 50$ (Figures 9 and 10) and the "old" procedures (based on the LCM) for $x = 50$.

So there is much less worry about correction for simultaneous coverage (in this context) than was previously thought.

5.2 Users

So what is a poor user to make of all of this? A general lesson is that we can do statistics when the MLE does not exist in the conventional sense (but does exist in the Barndorff-Nielsen completion of the exponential family). The fact that the MLE (in the Barndorff-Nielsen completion) doesn't make a lot of sense (as in Example I here where it is a completely degenerate distribution that says the observed value of the response vector is the only value it could possibly have) is not a problem. The sample is not the population. Estimates are not parameters. And so forth. The usual statistical apparatus of hypothesis tests and confidence intervals can be made to work in this situation.

But what about specific lessons? Can users be taught to use this stuff? Can we have computer programs that do all of this for the user so they are not burdened with doing all the calculations illustrated here and in the technical reports supporting Geyer (2009b)? Only time will tell. We haven't even gotten started on that.

5.3 More Complicated Procedures

Aster models (Geyer, et al., 2007; Shaw et al., 2008) are complicated exponential family models that are just like GLM in that they are exponential family regression models with a response vector y and multiple predictor vectors x_1, x_2, …, x_p that may be either quantitative or categorical, and they model the conditional distribution of y given x_1, x_2, …, x_p. But there are some differences

- In GLM the components of the response vector are all in the same "family," all normal, all binomial, or all Poisson, and so forth. In aster models the components of the response vector can be in different "families," some binomial, some Poisson, some normal, some zero-truncated Poisson, and so forth.

- In GLM the components of the response vector are independent (strictly speaking, conditionally independent given the predictors). In aster models the components of the response vector may be dependent *even after conditioning on the predictor variables.* The dependence is governed by a simple graphical model and follows rules that make the joint distribution of the response vector (conditional on predictors) an exponential family.

This is not the place to go into the details of aster models. The first aster software, the CRAN package `aster`, does not understand "solutions at infinity." Like the R function `glm`, the R function `aster` tests for solutions at infinity (non-existence of the MLE in the conventional sense) by ad hoc methods that can yield both false positives and false negatives. And when they give a warning (that may be a false positive), they give no help to the user. A follow-on CRAN package `aster2` was intended to understand solutions at infinity. It does have the correct basic plumbing to deal with all LCM of aster models. But it was never finished. To this day the pressing open research questions about aster models are those involving random effects. Solutions at infinity have had to wait.

The only reason we mention aster models here is their much more general notion of "prediction." The R function `predict.aster`, which is the method of the R generic function `predict` that is invoked when passed a first argument of class `"aster"` that should be produced by calling the R function `aster` to fit an aster model, does more general "prediction" than the R function `predict.glm`. It can be used to "predict" via the delta method any differentiable function of parameters. And this often comes in useful in applications of aster models.

So do our conclusions in Section 5.1 above carry over to aster models? We have not thought about that deeply, and probably need to work through some examples. That too is an open research question.

5.4 Likelihood-Based Competitors

Just for completeness, we mention again the argument near the end of Section 4.6.3 above that says that the sampling distribution of the log likelihood is never approximately normal when the mean value parameter is near the boundary of the convex support (and the data are discrete). This means that likelihood-based confidence intervals and confidence regions cannot be calibrated using critical values from chi-square distributions and work correctly. (At least their justification from conventional asymptotics is bogus.)

Any valid competitor for the proposals of Geyer (2009b) is an open research question.

References

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

Barndorff-Nielsen, O. (1978). *Information and Exponential Families.* Wiley, Chichester.

Brown, L. D. (1986). *Fundamentals of Statistical Exponential Families: with Applications in Statistical Decision Theory.* Institute of Mathematical Statistics, Hayward, CA.

Geyer, C. J. (1990). Likelihood and Exponential Families. PhD thesis, University of Washington. `http://purl.umn.edu/56330`.

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

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

Geyer, C. J. (2009b). Likelihood inference in exponential families and directions of recession. *Electronic Journal of Statistics*, **3**, 259–289.

Geyer, C. J. (2015a). R package `aster` (Aster Models), version 0.8-31. `http://cran.r-project.org/package=aster`.

Geyer, C. J. (2015b). R package `aster2` (Aster Models), version 0.2-1. `http://cran.r-project.org/package=aster2`.

Geyer, C. J. (2016a). Stat 8931 (Exponential Families) Lecture Notes: The Eggplant that Ate Chicago (the Notes that Got out of Hand). `http://www.stat.umn.edu/geyer/8931expfam/convex.pdf`.

Geyer, C. J. (2016b). Statistics 5102 (Geyer, Fall 2016) Examples: Coverage of Confidence Intervals. `http://www.stat.umn.edu/geyer/5102/examp/coverage.html`.

Geyer, C. J. and Thompson, E. A. (1992). Constrained Monte Carlo maximum likelihood for dependent data, (with discussion). *Journal of the Royal Statistical Society, Series B*, **54**, 657–699.

Geyer, C. J. and Møller, J. (1994). Simulation procedures and likelihood inference for spatial point processes. *Scandinavian Journal of Statistics*, **21**, 359–373.

Geyer, C. J., and Meeden, G. D. (2005). Fuzzy and randomized confidence intervals and P-values (with discussion). *Statistical Science*, **20**, 358–387.

Geyer, C. J., Wagenius, S., and Shaw, R. G. (2007). Aster models for life history analysis. *Biometrika*, **94**, 415–426.

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

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H., and Etterson, J. R. (2008). Unifying life history analysis for inference of fitness and population growth. *American Naturalist*, **172**, E35–E47.