Stat 5421 Lecture Notes

Exponential Families, Part I

Charles J. Geyer

October 28, 2020

Contents

1 Introduction

1.1 Definition

An *exponential family of distributions* is a statistical model that has log likelihood of the form

$$l(\theta) = \langle y, \theta \rangle - c(\theta), \tag{1}$$

where y is a vector statistic and θ is a vector parameter, both having the same dimension, and

$$\langle y, \theta \rangle = y^T \theta = \theta^T y = y \cdot \theta = \textstyle\sum_i x_i \theta_i.$$

Your humble author has a reason for preferring this angle bracket notation to either matrix multiplication or to the "dot product" or "scalar product" notation, but the reason isn't important for what we are going to do.

This is not the usual definition (Barndorff-Nielsen, 1978, Chapter 8) but it defines the same concept as the usual definition. Our definition comes from Geyer (2009).

Recall from the likelihood handout that additive terms that do not contain the parameter can be dropped from log likelihoods, and they may need to be dropped to get the form (1). Also recall from introductory statistics that a statistic is a function of the data that does not depend on the parameter. So y need not be "the data" (in its usual form); rather it may be a function of the data. Thus if ω is the original data, the probability mass functions (PMF) or probability density functions (PDF), as the case may be (PMF for discrete data or PDF for continuous data) have the form

$$f_\theta(\omega) = e^{\langle y(\omega), \theta \rangle - c(\theta) + h(\omega)} \tag{2}$$

where $h(\omega)$ is a term not depending on the parameter that is dropped from the log likelihood to obtain the form (1).

I want to say recall from introductory statistics that any one-to-one function of the parameter is another parameter, but I am aware that most introductory statistics courses go out of their way to never mention more than one parameterization of any particular statistical model. So we will just say it: any one-to-one function of the (vector) parameter of a statistical model is another (vector) parameter, and one parameterization is just as good as another: they all serve to distinguish the probability distributions in the model.

Many well known and widely used statistical models are exponential families (including Bernoulli, binomial, Poisson, geometric, negative binomial when the shape parameter is fixed, exponential, gamma, beta, normal, multinomial, and multivariate normal). In this course, of course, we are mostly interested in the discrete ones.

But even if you have had the theory sequence (mathematical probability and statistics) you may not see a connection between the PMF and PDF you know for these families of distributions and (1). It may be that the usual data are not the y and the usual parameter is not the θ in (1), so one needs to figure out what y and θ are to get the log likelihood into exponential family form (examples below, Sections 2 and 5).

A statistic y and parameter θ that give a log likelihood of the form (1) are called *canonical* or *natural*. The function c is called the *cumulant function* of the family. See Section 1.6 below for more on cumulant functions.

1.2 Terminology

Following Barndorff-Nielsen (1978), we are using modern terminology about exponential families.

An older terminology says "the exponential family" for the collection of all of what the newer terminology calls exponential families.

- The old terminology says, for example, the family of Poisson distributions is *in the* exponential family.
- The new terminology says, for example, the family of Poisson distributions is *an* exponential family.

The old terminology has nothing to recommend it. It makes the primary term — exponential family — refer to a heterogeneous collection of statistical models of no interest in any application.

The new terminology describes a property of statistical models that implies many other properties. It is a key concept of theoretical statistics.

1.3 Affine Functions

An affine function is what most people, including most statisticians, call a linear function, but which mathematicians, at least when being pedantic, do not. Starting in courses called "linear algebra" a linear function (also called linear transformation) is a function f between vector spaces that preserves the vector space operations, that is,

$$f(x+y) = f(x) + f(y), \qquad \text{for all vectors } x \text{ and } y \tag{3a}$$
$$f(ax) = af(x), \qquad \text{for all scalars } a \text{ and vectors } x \tag{3b}$$

The special case of (3b) when $a = 0$ says $f(0) = 0$, which most people think linear functions do not have to satisfy.

What most people mean by linear function is a function whose graph is flat. They are the scalar-to-scalar functions of the form

$$f(x) = a + bx,$$

where a and b are scalars, and vector-to-vector functions of the form

$$f(x) = a + Bx,$$

where a is a vector and B is a matrix. Mathematicians also need to discuss these kinds of functions, so they give them a name: affine functions.

The relations between the two kinds of functions are (using linear in the pedantic sense here)

- an affine function is a linear function plus a constant function, and
- a linear function is an affine function f satisfying $f(0) = 0$.

Your humble author likes to use linear in the pedantic sense and affine for linear in the sloppy sense. Apologies for the pedantry.

1.4 Non-Uniqueness

The canonical statistic, canonical parameter, and cumulant function of an exponential family are not unique.

- Any one-to-one affine function of a canonical statistic is another canonical statistic.
- Any one-to-one affine function of a canonical parameter is another canonical parameter.
- Any scalar-valued affine function of the canonical parameter added to the cumulant function gives another cumulant function.

These changes cannot be done separately. Changing one requires changes in the others to keep the form (1).

We usually do not worry about this. We fix one choice of canonical statistic, canonical parameter, and cumulant function and say "the" canonical statistic, "the" canonical parameter, and "the" cumulant function.

1.5 Non-Degeneracy

The PMF or PDF (2) of an exponential family are never zero. This means all distributions in the family have the same support (more on this later).

1.6 Cumulant Functions

Cumulant functions satisfy

$$c(\theta) = c(\psi) + \log E_\psi \left\{ e^{\langle y, \theta - \psi \rangle} \right\} \tag{4}$$

(equation (5) in Geyer, 2009). Whenever θ is an interior point of their domain, they also satisfy

$$E_\theta(y) = \nabla c(\theta) \tag{5}$$

$$\operatorname{var}_\theta(y) = \nabla^2 c(\theta) \tag{6}$$

(Barndorff-Nielsen, 1978, Theorem 8.1), where, as in the likelihood handout,

- $E(y)$ denotes the vector having components $E(y_i)$,
- $\nabla c(\theta)$ denotes the vector having components $\partial c(\theta) / \partial \theta_i$,
- $\operatorname{var}(y)$ denotes the matrix having components $\operatorname{cov}(y_i, y_j)$,
- and $\nabla^2 c(\theta)$ denotes the matrix having components $\partial^2 c(\theta) / \partial \theta_i \partial \theta_j$.

It is part of the assertion of Theorem 8.1 in Barndorff-Nielsen (1978) that the derivatives exist whenever θ is an interior point of the domain.

If there is an equality constraint on the parameter vector, so its components are not separately variable, the derivatives in (5) and (6) do not exist anywhere, but then we can sometimes use (4) to extend the domain of the cumulant function, as described in the next section, so that derivatives do exist.

1.7 Fullness and Regularity

In (4) consider ψ as fixed and θ as variable, and let θ vary over the entire Euclidean space that contains the canonical parameter space, letting $c(\theta)$ have the value $+\infty$ at points θ such that the expectation on the right-hand side does not exist. Then define

$$\Theta = \{\, \theta : c(\theta) < \infty \,\}. \tag{7}$$

This is called the *effective domain* of the cumulant function. We can extend the exponential family to (7) by defining the PMF or PDF by

$$f_\theta(\omega) = f_\psi(\omega) e^{\langle y(\omega), \theta - \psi \rangle - c(\theta) + c(\psi)} \tag{8}$$

at points θ that were not in the originally given canonical parameter space.

We say an exponential family is *full* if its canonical parameter space is (7). We say a full exponential family is *regular* if (7) is an open subset of the Euclidean space where θ takes values. Another name for the set (7) is the canonical parameter space of the full exponential family.

For many exponential families (7) is the whole Euclidean space. But not for all. Geometric, negative binomial, and two-parameter normal are examples where (7) is not a whole Euclidean space.

Regularity is very important. Many of the desirable properties of exponential families hold only for regular full exponential families. For example, if the full family is regular, this means every canonical parameter value θ is an interior point of the full canonical parameter space (7), and, consequently, every distribution in the family has mean vector and variance matrix given by derivatives of the cumulant function (5) and (6).

Fortunately, almost all exponential families that arise in applications are full and regular. The only examples I know of that don't are some models for spatial point processes (Geyer and Møller, 1994; Geyer, 1999). Every exponential family we will look at in this course is regular.

2 Examples I

2.1 Binomial

The binomial distribution has log likelihood

$$l(\pi) = x \log(\pi) + (n - x) \log(1 - \pi)$$

where x is the "usual" data, π is the "usual" parameter, and n is neither data nor parameter but rather a constant. To get this into the form (1), assuming the family is one-dimensional (which having only one parameter suggests), we need to write this as a function of x (which will be the canonical statistic y) times a function of π (which will be the canonical parameter θ) minus a function of θ (the cumulant function). So isolate x

$$\begin{aligned} l(\pi) &= x\bigl[\log(\pi) - \log(1-\pi)\bigr] + n\log(1-\pi) \\ &= x\log\left(\frac{\pi}{1-\pi}\right) + n\log(1-\pi) \end{aligned}$$

and this tells us

- the canonical statistic is x,
- the canonical parameter is
$$\theta = \log\left(\frac{\pi}{1-\pi}\right), \tag{9a}$$
- and the cumulant function is
$$c(\theta) = -n\log(1-\pi). \tag{9b}$$

Of course (9b) doesn't really make sense because it has θ on one side and π on the other. We need to solve (9a) for π obtaining

$$\pi = \frac{e^\theta}{1+e^\theta} \tag{10}$$

and plug this back into (9b) obtaining

$$\begin{aligned} c(\theta) &= -n\log(1-\pi) \\ &= -n\log\left(1 - \frac{e^\theta}{1+e^\theta}\right) \\ &= -n\log\left(\frac{1}{1+e^\theta}\right) \\ &= n\log\left(1+e^\theta\right) \end{aligned}$$

Since this formula is valid for all θ, we see the binomial family is a regular full exponential family whose canonical parameter space is the whole of Euclidean space (one-dimensional Euclidean space in this case).

This change-of-parameter is so important that statisticians give it a name

$$\operatorname{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right) \qquad 0 < \pi < 1. \tag{11}$$

(logit is pronounced "low-jit" with a soft "g").

Note that 0 and 1 are not possible arguments of the logit function. We get $\log(0)$ or division by zero, either of which is undefined. The logit function maps $(0,1) \to (-\infty, +\infty)$. So the binomial family of distributions, considered as an exponential family, has "usual" parameter space $0 < \pi < 1$ and canonical parameter space $-\infty < \theta < +\infty$).

This is a little surprising because the usual "usual" parameter space is $0 \le \pi \le 1$ because otherwise we have no parameter values to be maximum likelihood estimates when we observe $x = 0$ or $x = n$. So exponential family theory causes some trouble here. Much more on this subject later.

If we think about it a bit more though, this is just the non-degeneracy property of exponential families (Section 1.5 above) at work. This is why the degenerate distributions concentrated at $x = 0$ and $x = n$ (for $\pi = 0$ and $\pi = 1$) cannot be included in the exponential family.

Summary For the binomial distribution

- the canonical statistic is x,
- the canonical parameter is $\theta = \operatorname{logit}(\pi)$,
- the canonical parameter space of the full family is $-\infty < \theta < +\infty$,
- and the cumulant function is

$$c(\theta) = n \log\bigl(1 + e^{\theta}\bigr). \tag{12}$$

2.2 Poisson

The Poisson distribution has log likelihood

$$l(\mu) = x \log(\mu) - \mu$$

where x is the "usual" data and μ is the "usual" parameter. To get this into the form (1), assuming the family is one-dimensional (which having only one parameter suggests), we need to write this as a function of x (which will be the canonical statistic y) times a function of μ (which will be the canonical parameter θ) minus a function of θ (the cumulant function). Obviously,

- the canonical statistic is x,
- the canonical parameter is
$$\theta = \log(\mu),$$
which has inverse function
$$\mu = e^\theta,$$
- and the cumulant function is
$$c(\theta) = \mu = e^\theta.$$

Note that 0 is not a possible argument of the log function. The log function maps $(0, +\infty) \to (-\infty, +\infty)$. So the Poisson family of distributions, considered as an exponential family, has "usual" parameter space $0 < \mu < +\infty$ and canonical parameter space $-\infty < \theta < +\infty$).

The Poisson distribution for $\mu = 0$ is degenerate, concentrated at zero, so by the non-degeneracy property (Section 1.5 above) it cannot be included in the exponential family.

Summary For the Poisson distribution

- the canonical statistic is x,
- the canonical parameter is $\theta = \log(\mu)$,
- the canonical parameter space of the full family is $-\infty < \theta < +\infty$,
- and the cumulant function is
$$c(\theta) = e^\theta. \tag{13}$$

2.3 Negative Binomial

The negative binomial distribution has PMF
$$f(x) = \binom{r + x - 1}{x} p^r (1 - p)^x, \qquad x = 0, 1, \ldots \tag{14}$$
where x is the "usual" data and r and p are the "usual" parameters and
$$\binom{r}{x} = \frac{r \cdot (r-1) \cdots (r - x + 1)}{x!}.$$

If we consider both r and p to be unknown parameters, this is not an exponential family. If we consider r to be known, so p is the only unknown parameter, then it is an exponential family. So we consider r as known.

Then the log likelihood is

$$l(p) = r \log(p) + x \log(1 - p).$$

Obviously,

- the canonical statistic is x,
- the canonical parameter is

  $$\theta = \log(1 - p),$$

  which has inverse function

  $$p = 1 - e^{\theta},$$

- and the cumulant function is

  $$c(\theta) = -r \log(p) = -r \log\bigl(1 - e^{\theta}\bigr). \tag{15}$$

The negative binomial distribution for $p = 0$ does not exist ($p = 0$ is not an allowed parameter value). The negative binomial distribution for $p = 1$ does exist but is degenerate, concentrated at zero. So this degenerate distribution cannot be included in the exponential family (by Section 1.5 above).

The log function maps $(0, 1) \to (-\infty, 0)$. So the negative binomial family of distributions, considered as an exponential family, has "usual" parameter space $0 < p < 1$ and canonical parameter space $-\infty < \theta < 0$).

We are tempted to use (4) to extend the family, but if we try, we find that (4) gives us the same cumulant function we already have, so the negative binomial family as we have described it is already a regular full exponential family.

Summary For the negative binomial distribution

- the canonical statistic is x,
- the canonical parameter is $\theta = \log(1 - p)$,
- the canonical parameter space of the full family is $-\infty < \theta < 0$,
- and the cumulant function is

  $$c(\theta) = -r \log(1 - e^{\theta}) \tag{16}$$

Commentary This provides an example where the canonical parameter space of the full family is not the whole Euclidean space (in this case one-dimensional Euclidean space).

2.4 Multinomial

The multinomial PDF is

$$f(x) = \binom{n}{x} \prod_{i=1}^{k} \pi_i^{x_i}$$

where $x = (x_1, \ldots, x_k)$ is the vector of cell counts of a contingency table (the "usual" data vector), where $\pi = (\pi_1, \ldots, \pi_k)$ is the vector of cell probabilities (the "usual" parameter vector), where n is the sample size (the sum of the cell counts), and where $\binom{n}{x}$ is a multinomial coefficient. The log likelihood is

$$l(\pi) = \sum_{i=1}^{k} x_i \log(\pi_i)$$

2.4.1 Try I

It looks like we have exponential family form with the canonical statistic vector x, canonical parameter vector θ having components $\theta_i = \log(\pi_i)$, and cumulant function the zero function.

As we shall eventually see, this is correct but misleading. If we use (5) and (6) on the zero function we get that y has mean vector zero and variance matrix zero, which is incorrect. But we are not allowed to use those formulas because the components of θ, as we have defined them in this section, are not separately variable: the components of π are constrained to sum to one. And that translates to a complicated nonlinear constraint on the canonical parameters

$$\sum_{i=1}^{k} e^{\theta_i} = 1. \tag{17}$$

That looks annoying. So we try something else.

2.4.2 Try II

If we eliminate one of the components, say π_k, we get log likelihood

$$l(\pi_1, \ldots, \pi_{k-1}) = x_k \log\left(1 - \sum_{i=1}^{k-1} \pi_i\right) + \sum_{i=1}^{k-1} x_i \log(\pi_i)$$

But since we are trying to put this in exponential family form in which the canonical statistic vector and canonical parameter vector must have the same dimension, we have to also eliminate the corresponding component of x. We can do that because the components of x are constrained to sum to n. If we use that fact to eliminate x_k we get

$$\begin{aligned} l(\pi_1, \ldots, \pi_{k-1}) &= \left(n - \sum_{i=1}^{k-1} x_i \right) \log \left(1 - \sum_{i=1}^{k-1} \pi_i \right) + \sum_{i=1}^{k-1} x_i \log(\pi_i) \\ &= n \log \left(1 - \sum_{i=1}^{k-1} \pi_i \right) + \sum_{i=1}^{k-1} x_i \left[\log(\pi_i) - \log \left(1 - \sum_{i=1}^{k-1} \pi_i \right) \right] \\ &= n \log \left(1 - \sum_{i=1}^{k-1} \pi_i \right) + \sum_{i=1}^{k-1} x_i \log \left(\frac{\pi_i}{1 - \sum_{i=1}^{k-1} \pi_i} \right) \end{aligned}$$

and this has exponential family form with

- canonical statistic vector $(x_1, \ldots, x_{k-1})$,
- canonical parameter vector $(\theta_1, \ldots, \theta_{k-1})$, where

$$\theta_i = \log \left(\frac{\pi_i}{1 - \sum_{i=1}^{k-1} \pi_i} \right) \tag{18}$$

- and cumulant function

$$c(\theta_1, \ldots, \theta_{k-1}) = -n \log \left(1 - \sum_{i=1}^{k-1} \pi_i \right), \tag{19}$$

except that (as usual) this doesn't make sense with θ's on the left-hand side and π's on the right-hand side, so we have to solve for the π's in terms of the θ's and plug in. Reintroduce $\pi_k = 1 - \sum_{i=1}^{k-1} \pi_i$ and plug into (18) obtaining

$$\theta_i = \log \left(\frac{\pi_i}{\pi_k} \right)$$

or

$$e^{\theta_i} = \frac{\pi_i}{\pi_k}$$

or

$$\pi_i = \pi_k e^{\theta_i} \tag{20}$$

and this also holds for $i = k$ if we define $\theta_k = 0$ (which we may do because there was no definition of θ_k before). Since the π's must sum to 1, summing (20) gives

$$1 = \pi_k \sum_{i=1}^{k} e^{\theta_i} = \pi_k \left(1 + \sum_{i=1}^{k-1} e^{\theta_i} \right)$$

and

$$\pi_k = \frac{1}{1 + \sum_{i=1}^{k-1} e^{\theta_i}}$$

and (plugging this back into (20))

$$\pi_i = \frac{e^{\theta_i}}{1 + \sum_{i=1}^{k-1} e^{\theta_i}}$$

and (plugging this back into (19))

$$\begin{aligned} c(\theta_1, \ldots, \theta_{k-1}) &= -n \log \left(1 - \sum_{i=1}^{k-1} \frac{e^{\theta_i}}{1 + \sum_{i=1}^{k-1} e^{\theta_i}} \right) \\ &= n \log \left(1 + \sum_{i=1}^{k-1} e^{\theta_i} \right) \end{aligned}$$

I don't know what you think of this, but I think it is horribly messy. Also arbitrary. Clearly, which π_i we choose to eliminate is arbitrary. But we also know (Section 1.4 above) that there is a lot more arbitrariness than that in choosing canonical statistic, canonical parameter, and cumulant function.

2.4.3 Try III

Let's try again, this time not eliminating parameters or statistics, and using (4) to define the cumulant function. Let ψ in (4) be the canonical parameter vector for the multinomial distribution having probability vector $p = (p_1, \ldots, p_k)$. Let S denote the sample space of the multinomial distribution: the set of all vectors x having nonnegative integer components that sum to n. Then (4) says

$$\begin{aligned}
c(\theta) &= c(\psi) + \log \sum_{x \in S} e^{\langle x, \theta - \psi \rangle} \cdot \binom{n}{x} \prod_{i=1}^{k} p_i^{x_i} \\
&= c(\psi) + \log \sum_{x \in S} e^{\sum_{i=1}^{k} x_i (\theta_i - \psi_i)} \cdot \binom{n}{x} \prod_{i=1}^{k} p_i^{x_i} \\
&= c(\psi) + \log \sum_{x \in S} \left[\prod_{i=1}^{k} e^{x_i (\theta_i - \psi_i)} \right] \cdot \binom{n}{x} \prod_{i=1}^{k} p_i^{x_i} \\
&= c(\psi) + \log \sum_{x \in S} \binom{n}{x} \prod_{i=1}^{k} p_i^{x_i} e^{x_i (\theta_i - \psi_i)} \\
&= c(\psi) + \log \sum_{x \in S} \binom{n}{x} \prod_{i=1}^{k} \left[p_i e^{\theta_i - \psi_i} \right]^{x_i} \\
&= c(\psi) + \log \left(\sum_{i=1}^{k} p_i e^{\theta_i - \psi_i} \right)^n \\
&= c(\psi) + n \log \left(\sum_{i=1}^{k} p_i e^{\theta_i - \psi_i} \right)
\end{aligned}$$

We use the multinomial theorem to evaluate the sum over the sample space. If we take ψ to be the vector with all components zero and p to be the vector with all components equal to $1/k$, which we are free to do because ψ and p were arbitrary, we get

$$\begin{aligned}
c(\theta) &= c(\psi) + n \log \left(\frac{1}{k} \sum_{i=1}^{k} e^{\theta_i} \right) \\
&= c(\psi) + n \log \left(\frac{1}{k} \sum_{i=1}^{k} e^{\theta_i} \right) \\
&= c(\psi) - n \log(k) + n \log \left(\sum_{i=1}^{k} e^{\theta_i} \right)
\end{aligned}$$

and, since we are free to choose $c(\psi)$, we can choose it to cancel the $-n \log(k)$, finally obtaining

$$c(\theta) = n \log \left(\sum_{i=1}^{k} e^{\theta_i} \right)$$

But now we have lost track of what the usual parameters are in this parameterization. We use (5) to find them again.

$$E(x_i) = n\pi_i = \frac{\partial c(\theta)}{\partial \theta_i} = \frac{n e^{\theta_i}}{\sum_{i=1}^k e^{\theta_i}}$$

so

$$\pi_j = \frac{e^{\theta_j}}{\sum_{i=1}^k e^{\theta_i}} \tag{21}$$

2.4.4 Summary

Try II For the multinomial family of dimension k and sample size n

- the canonical statistic is the vector $(x_1, \ldots, x_{k-1})$ whose components are all but one of the category counts,
- the canonical parameter is the vector $(\theta_1, \ldots, \theta_{k-1})$, where

$$\theta_i = \log\left(\frac{\pi_i}{\pi_k}\right), \tag{22a}$$

- the canonical parameter space of the full family is the whole of $(k-1)$-dimensional Euclidean space,
- and the cumulant function is

$$c(\theta) = n \log\left(1 + \sum_{i=1}^{k-1} e^{\theta_i}\right) \tag{22b}$$

Binomial Distribution The binomial distribution is the $k = 2$ case of the multinomial distribution when the "try II" parameterization is used. If x_1 is the number of successes and x_2 is the number of failures, then we eliminate x_2 from the canonical statistic vector, just leaving the scalar variable x_1. Because the π's sum to one we have $\pi_2 = 1 - \pi_1$ and plugging this into (22a) gives the analogous equation for the binomial distribution (9a). Also (22b) is the same as the analogous equation for the binomial distribution (12).

Try III For the multinomial family of dimension k and sample size n

- the canonical statistic is the vector x of category counts
- the canonical parameter is the vector θ,

- the canonical parameter space of the full family is the whole of k-dimensional Euclidean space,
- the cumulant function is
$$c(\theta) = n \log \left(\sum_{i=1}^{k} e^{\theta_i} \right) \tag{23}$$

Commentary This provides an example where we have three different ways to put the distribution in exponential family form and we can choose the one we like.

Try I seemed simple at first but is complicated by the nonlinear constraint on the parameters (17). Try II is ugly but simple to use. Try III is elegant but a little more complicated.

Try I is the special case of Try III where we impose the constraint (17). Try II is the special case of Try III where we impose the constraint $\theta_k = 0$ and then don't consider θ_k part of the canonical parameter vector and don't consider x_k part of the canonical statistic vector.

Try III is complicated by the fact that the mapping between canonical parameters and usual parameters (21) is not a one-to-one mapping. If one adds the same constant to all of the θ_i on the right-hand side of (21), then it does not change the left-hand side. In order to get a one-to-one relationship we need to impose a constraint on the canonical parameters. We can choose the constraint to be anything convenient. We can use the ones already mentioned, (17) and $\theta_k = 0$. But we can also use others.

Try III is important because it keeps all the cell counts as components of the canonical statistic vector. This makes it much easier to reason about models. In the complicated exponential family models that arise in aster models (Geyer, Wagenius, and Shaw, 2007; Geyer, 2015), it is essential that the canonical statistic vector be the full vector of counts. Thus we have to use the "try III" parameterization.

We can explain the difference between try II and try III as follows. In try II we think that getting a one-to-one correspondence between usual parameters and canonical parameters is so important that we do it right away and make all the math and other reasoning about models very messy. In try III we think that getting a one-to-one correspondence is not important, and we do not allow it to mess up the math and other reasoning. We can impose a constraint to get one-to-one correspondence whenever we want.

3 Independent and Identically Distributed

If we observe data independent and identically distributed (IID) from any statistical model, the joint distribution is the product of marginal distributions

$$f_\theta(x_1, \ldots, x_n) = \prod_{i=1}^n f_\theta(x_i)$$

so the likelihood is also a product

$$L_n(\theta) = \prod_{i=1}^n f_\theta(x_i)$$

and the log likelihood is a sum

$$l_n(\theta) = \sum_{i=1}^n \log f_\theta(x_i).$$

When we apply this to exponential families, the log likelihood is (1) with y changed to y_i and summed

$$\begin{aligned} l(\theta) &= \sum_{i=1}^n \bigl[\langle y_i, \theta \rangle - c(\theta)\bigr] \\ &= \left\langle \sum_{i=1}^n y_i, \theta \right\rangle - nc(\theta) \end{aligned}$$

Thus we see that IID sampling produces a new exponential family and

- the canonical statistic is $\sum_{i=1}^n y_i$,
- the canonical parameter is θ,
- and the cumulant function is $nc(\theta)$.

or

- the canonical statistic for sample size n is the sum of the canonical statistics for the sampled individuals,
- the canonical parameter is the same for all sample sizes,
- and the cumulant function for sample size n is n times the cumulant function for sample size one.

This explains many "addition rules" for distributions (sum of IID binomial is binomial, sum of IID Poisson is Poisson, sum of IID normal is normal, and so forth for other exponential families).

4 Sufficiency

4.1 Definition

Fisher (1922) invented the concept of sufficiency and sufficient statistics. A scalar or vector statistic (function of the data that does not depend on parameters) is *sufficient* if the the conditional distribution of the whole data given that statistic does not depend on parameters, in notation, if ω is the whole data and y is the statistic, then y is sufficient if the distribution of ω given y does not depend on the parameters.

4.2 The Sufficiency Principle

Fisher argued that the sufficient statistic extracts all of the information in the data relevant to estimating the parameters. If y is sufficient and ω is the whole data and we write joint equals marginal times conditional

$$f_\theta(\omega, y) = f(\omega \mid y) f_\theta(y),$$

only the marginal depends on the parameters. So Fisher argued that we should use that marginal to estimate parameters, because the conditional does not involve the parameters and hence has nothing to do with them. Fisher's argument was later dubbed the *sufficiency principle:* statistical inference should only depend on the data through the sufficient statistics.

The likelihood may be written

$$L(\theta) = f_\theta(y)$$

(as usual we may drop multiplicative terms from the likelihood that do not contain parameters). Thus likelihood inference and Bayesian inference automatically obey the sufficiency principle. It is only ad hoc methods of estimation that may violate it.

4.3 The Neyman-Fisher Factorization Criterion

The definition of sufficient statistics is hard to apply (you have to factor the joint distribution into marginal times conditional), but the *Neyman-Fisher factorization criterion* (Fisher, 1922; Neyman, 1935; Halmos and Savage, 1949) is much easier to apply. This says a vector statistic is sufficient if and only if there is a version of the likelihood (possibly obtained by dropping some multiplicative factors that do not contain the parameters) or log likelihood (possibly obtained by dropping some additive factors that do not contain the parameters) that depends on the whole data only through that statistic.

From (1) and the Neyman-Fisher factorization criterion we see that the canonical statistic vector of an exponential family is always a sufficient statistic vector. From Section 3 we see that the dimension of the sufficient statistic vector does not change as the sample size changes. This is not automatic. There are statistical models that have no sufficient statistic vector of smaller dimension than the whole data (the Cauchy location family, for example).

4.4 The Pitman-Koopman-Darmois Theorem

The so-called Pitman-Koopman-Darmois theorem (Pitman, 1936; Koopman, 1936; Darmois, 1935) says this property exponential families have that in IID sampling the dimension of the sufficient statistic vector does not change as the sample size changes actually characterizes exponential families under certain side conditions.

Without side conditions the theorem simply isn't true. The uniform $(0, \theta)$ statistical model has sufficient statistic $\max(x_1, \ldots, x_n)$, which is one-dimensional for all n, but this model is not an exponential family. The theorem rules this out by a side condition that requires the support of the distributions in the family to not depend on the parameter.

Pitman, Koopman, and Darmois each independently proved that under this side condition the only continuous statistical models that have sufficient statistics whose dimension does not depend on the sample size in IID sampling are exponential families.

There are analogs of the Pitman-Koopman-Darmois theorem for discrete data but they add ugly and unmotivated side conditions to get the theorem. They are what my brother-in-law's thesis advisor called "ham and eggs theorems" (if we had some ham, we'd have ham and eggs, if we had some eggs). Nevertheless, Pitman-Koopman-Darmois theory did kick off the subject of exponential families.

4.5 Linear Models

To a statistician, the term *linear model* (LM) means simple linear regression or multiple linear regression with the usual assumptions (IID normal errors having mean zero). This includes analysis of variance (ANOVA), which is just multiple linear regression when all predictors are categorical. More precisely, these usual assumptions are

$$y_i = \beta_1 + \beta_2 x_{i1} + \beta_3 x_{i2} + \cdots + \beta_p x_{i,p-1} + e_i \tag{24}$$

where the observed data for the i-th individual comprises

- y_i (response variable) and
- $x_{i1}, \ldots, x_{ip}$ (predictor variables),

where

- e_i is an unobserved random variable (error), the error variables for all individuals being assumed IID normal with mean zero,

and where

- $\beta_1, \ldots, \beta_p$ are unknown parameters,
- and the error variance is another unknown parameter.

In the original theory (Galton, 1886; Galton and Dickson, 1886) that gave the methodology its name "regression" and the phenomenon "regression towards the mean" the individuals were assumed IID with $(y_i, x_{i1}, \ldots, x_{ip})$ a multivariate normal random vector. Then the conditional distribution of y_i given $x_{i1}, \ldots, x_{ip}$ is normal with mean

$$\beta_1 + \beta_2 x_{i1} + \beta_3 x_{i2} + \cdots + \beta_p x_{i,p-1}$$

(for some parameters $\beta_1, \ldots, \beta_p$) and variance that does not depend on $x_{i1}, \ldots, x_{ip}$, that is, the regression equation (24) holds. Galton and Dickson (1886) showed this only for the $p = 1$ case, but this is now known to be a theorem about the multivariate normal distribution (Anderson, 2003, Theorem 2.5.1) that holds for general p.

But later authors realized that only the conditional distribution mattered. It does not matter what the marginal distribution of $x_{i1}, \ldots, x_{ip}$ is if statistical inference is done conditionally. This also allows for the case where some or all of the predictor variables are not random but fixed by experimental design. For example, in ANOVA $x_{i1}, \ldots, x_{ip}$ are all dummy variables that indicate categories, and they aren't random. So modern practice is to assume nothing about the marginal distribution of $x_{i1}, \ldots, x_{ip}$ or even that they are random. The only assumptions are that (24) holds with the e_i IID mean-zero normal.

It is important to understand that it is called a linear model because (24) is linear in the β's not because it is linear in the x's. In fact, when nonlinear functions of the data are used for predictors, as in polynomial regression when

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \cdots + \beta_p x_i^{p-1} + e_i$$

it is still a linear model (because it is linear in the β's). In fact, one can (as in polynomial regression) make up new predictor variables that are functions of the originally given predictor variables. We do this all the time (dummy variables corresponding to categorical predictors, "interaction" terms, and lots more). So x_{i1}, $\ldots$, x_{ip} don't have to be the originally given data, they can be any made-up functions of the originally given data.

But (24) is very unwieldy notation. The notation becomes much simpler and also much more powerful if we shift to matrix notation. We do this in two steps. First we notice that in (24) we need not treat β_1 as special, because (24) is the special case of

$$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots + \beta_p x_{ip} + e_i \tag{25}$$

where $x_{i1} = 1$ for all i. Since (25) is more general than (24) we shift to that. Now we shift to matrix notation. If

- y is the random vector having components y_i,
- M is the non-random matrix having components x_{ij} (nonrandom because the inference is conditional on the x_{ij}),
- β is the non-random (parameter) vector having components β_j,
- e is the random vector having components e_i,

then (25) can be written

$$y = M\beta + e. \tag{26}$$

As an example of the power of matrix notation the maximum likelihood estimate (MLE) of β can be written

$$\hat{\beta} = (M^T M)^{-1} M^T y \tag{27}$$

but we won't even give a reference for that (it can be found in any book on linear regression) because it is irrelevant to this course. For us, linear models are just a step on the way to generalized linear models (GLM) and exponential family canonical affine models (about which more below). The only point (for us) of (27) is that you couldn't express it without matrix notation. Other advantages of matrix notation will arise as we go along.

The matrix M is called the *model matrix* by some statisticians and the *design matrix* by other statisticians. In a designed experiment in which none of the predictors are random, rather they are all fixed by the experimental design, M incorporates everything about the design that is relevant to statistical data analysis (hence the name "design matrix"). In an observational study in which some of the predictors may be random, but are treated as fixed in the statistical analysis (because conditioning on a variable is the same as treating it as fixed) and nothing was "designed," the name "model matrix" seems more appropriate. Your humble author being a computer nerd, always says "model matrix" because that is what R calls it (the R function `model.matrix` is what turns R formulas into model matrices; ordinary users never call this function, but it is called inside R functions like `lm` and `glm` to do this job).

4.6 Canonical Affine Submodels

The notion of canonical affine submodels of exponential families was only proposed recently (Geyer, et al., 2007) but in hindsight is only a generalization of generalized linear models like logistic regression (Agresti, 2013, Chapter 5) and Poisson regression with log link (Agresti, 2013, Section 4.3) and of log-linear models for contingency tables (Agresti, 2013, Chapters 9 and 10) that had long been used.

Geyer, et al. (2007) is about a class of exponential family models they call "aster models" that are like generalized linear models except that they allow

- different components of the response vector are allowed to have different families, some Bernoulli, some Poisson, some normal, etc.,
- and components of the response vector that are measurements on the same individual can be dependent with the dependency structure specified by a graphical model.

But we don't want to talk about aster models now (maybe another handout if there is time). We just wanted to explain why generalized linear models had to be generalized.

A *canonical affine submodel* of an exponential family arises when the canonical parameter vector of an exponential family is modeled affinely

$$\theta = a + M\beta \tag{28}$$

where

- θ is the canonical parameter of the full model,
- a is a non-random vector called the *offset vector*,
- M is a non-random matrix called the *model matrix*,

- and β is the submodel parameter vector.

The name comes from θ being the canonical parameter and it being an affine function of β.

When $a = 0$, which is the usual case, we say *canonical linear submodel* because then θ is a linear function of β.

The R functions `lm` and `glm` have an `offset` argument that, when used, would change the name to "affine model" or "generalized affine model" if one wanted to be pedantic, but nobody says this. Since offsets are very rarely used, this doesn't matter much. But the "linear" in LM and GLM is pedantically correct for models without offsets and sloppily correct (using "linear" in the sense most people use it) even with offsets.

Now we come to the theory of canonical affine submodels, which is very simple. Plugging (28) into (1) we get the submodel log likelihood

$$\begin{aligned} l(\beta) &= \langle y, a + M\beta \rangle - c(a + M\beta) \\ &= y^T(a + M\beta) - c(a + M\beta) \\ &= y^T a + y^T M\beta - c(a + M\beta) \\ &= y^T a + \beta^T M^T y - c(a + M\beta) \\ &= y^T a + (M^T y)^T \beta - c(a + M\beta) \\ &= \langle y, a \rangle + \langle M^T y, \beta \rangle - c(a + M\beta) \end{aligned}$$

and we are allowed to drop the additive term that does not contain the parameter β obtaining

$$l(\beta) = \langle M^T y, \beta \rangle - c(a + M\beta) \tag{29}$$

which we see again has exponential family form, hence the submodel is itself an exponential family with

- canonical statistic vector $M^T y$,
- canonical parameter vector β,
- and cumulant function defined by
$$c_{\text{sub}}(\beta) = c(a + M\beta). \tag{30}$$

The reader may be thinking, so what? But as we shall see, exponential family theory is very powerful. It has a lot of consequences we haven't even hinted at yet. For now all we know is that $M^T y$, being the submodel canonical statistic vector, is a sufficient statistic vector for the submodel. (It contains all of the information in the whole data that is useful for estimating β).

4.7 Sufficient Dimension Reduction

There is a lot of local interest in the School of Statistics at the University of Minnesota in the "sufficient dimension reduction" program (Cook and Weisberg, 1991; Cook, 1998; Cook and Adragni, 2009). This is not the place to explain that. We just want to remember that the original sufficient dimension reduction was Fisher's notion of a sufficient statistic (which Cook and co-authors are also using) and Koopman-Pitman-Darmois theory which says that exponential families have the sufficient dimension reduction property in IID sampling.

In exponential family contexts, the fact that the submodel canonical statistic vector is sufficient for a canonical affine submodel explains why linear models, some generalized linear models (logistic regression and Poisson regression with log link, for example), and log-linear models for categorical data also do "sufficient dimension reduction."

Of course, the point of the sufficient dimension reduction program of Cook and co-authors is to do sufficient dimension reduction nonparametrically without assuming a particular parametric submodel is correct. So these notions are a bit different. But they are very closely related.

5 Examples II

5.1 Logistic Regression

Suppose we have regression-like data, each individual has a response y_i and a bunch of predictors x_{i1}, $\ldots$, x_{ip} (possibly some originally given and some made up) and the y_i are all Bernoulli (zero-or-one-valued). As with linear models, we collect the y_i into a vector y and collect the predictors into a model matrix M.

Now, however, the "response equals mean function plus error" formalism used in linear models (equations (24), (25), and (26) above) makes no sense for two reasons.

- Modeling means linearly makes no sense because they are bounded. Write $\mu_i = E(y_i)$. Then $0 \le \mu_i \le 1$. But if we collect the means into a vector μ and model them as $\mu = M\beta$, we will get values of μ that violate the constraints.

  One might think there would be some mathematical magic that could make this idea work (constrained optimization perhaps), but there is none found in the literature, which suggests it is a hopeless idea.

- The "means plus error" is also bad because the "errors" $y_i - \mu_i$ do not have any distribution with well studied properties.

In short, to make progress, we need to ditch two key ideas of linear models

- response equals means plus error,
- and means are linear functions of the parameters.

What to do? Exponential families and sufficient dimension reduction to the rescue. The distribution of the response vector y is exponential family (Bernoulli is binomial with sample size one, and we have seen that binomial is exponential family (Section 2.1) if we assume (as in LM) that components of the response are independent (or conditionally independent given the predictors if we are fussy about thinking of the predictors as possibly random variables that we are treating as fixed by conditioning on them) then the joint distribution of the y_i is the product of the marginals and the distribution of the whole vector y is exponential family. Introducing the canonical parameters

$$\theta_i = \operatorname{logit}(\mu_i), \qquad \text{for all } i,$$

the log likelihood is

$$l(\theta) = \sum_{i=1}^{n} \bigl[y_i \theta_i - c(\theta_i) \bigr]$$

where c is the cumulant function for the Bernoulli family (the case $n = 1$ of equation (12)), and this has exponential family form

$$l(\theta) = \langle y, \theta \rangle - c(\theta)$$

where we collect the θ_i into a vector θ and define

$$c(\theta) = \sum_{i=1}^{n} c(\theta_i)$$

using what mathematicians call "abuse of notation" in that we are using the same letter c for two different functions; on the left-hand side it is a function of the vector θ, but on the right-hand side it is a function of the scalar θ_i.

Canonical linear models $\theta = M\beta$ (when there is no offset, the usual case) or canonical affine models $\theta = a + M\beta$ (when there is an offset) have the sufficient dimension reduction property.

This idea is what is usually called "logistic regression" for reasons that will be explained when we get to GLM theory.

This is a really good idea. These models work great and are widely used.

5.2 Poisson Regression with Log Link

Apply the same ideas to Poisson data. Now we assume the y_i are independent (or conditionally independent given the predictors if being pedantic) Poisson.

We get exactly the same story as in the preceding section except the canonical parameters for the Poisson

$$\theta_i = \log(\mu_i), \qquad \text{for all } i, \tag{31}$$

are different from those for the Bernoulli, as are the cumulant functions (equation (13)). Everything else is the same as in the preceding section. Canonical linear submodels and canonical affine submodels have the sufficient dimension reduction property. And this too is a really good idea. These models work great and are widely used.

The term "log link" won't make sense until we get to GLM. The "log" refers to the log in (31). The "link" is a term of GLM theory.

6 Maximum Likelihood Estimation

Maximum likelihood estimation is much nicer in exponential families than in general statistical models. The reason is that cumulant functions are convex functions (Barndorff-Nielsen, 1978, Theorem 7.1) and consequently log likelihoods are concave functions. Moreover, they are strictly convex and strictly concave (respectively) if the canonical statistic vector is not concentrated on a hyperplane, that is, if the components of the canonical statistic vector do not satisfy a linear constraint that holds with probability one (again Barndorff-Nielsen, 1978, Theorem 7.1).

This is not the place to define convexity and concavity. There are many equivalent characterizations (Rockafellar and Wets, 1998, Theorems 2.13 and 2.14). We will just explain the important consequences.

6.1 Identifiability

A parameterization is *identifiable* if each different parameter value corresponds to a different distribution. Every parameterization we have talked about in this document is identifiable except for one (the "try III" parameterization for the multinomial).

Identifiability in full exponential families is fairly simple, described by Theorem 1 in Geyer (2009), which we condense.

Theorem 1. *For a full exponential family with canonical statistic vector y, canonical parameter vector θ, canonical parameter space Θ, and log likelihood l,*

(a) *the canonical parameterization fails to be identifiable if and only if there exists a nonzero vector δ such that $\langle y, \delta \rangle$ is constant,*

(b) *and this happens if and only if θ and $\theta + r\delta$ correspond to the same distribution for all real r and all $\theta \in \Theta$,*

(c) *and this happens if and only if $l(\theta + r\delta)$ is a constant function of r for all real r and all $\theta \in \Theta$.*

Any vector δ satisfying these conditions is called a *direction of constancy* of the family by Geyer (2009).

Every parameterization discussed in this document is identifiable except one. The try III parameterization of the multinomial distribution is not identifiable by part (a) of the theorem because of the constraint

$$\sum_{i=1}^{k} y_i = n,$$

which can be written in vector notation

$$\langle y, \delta \rangle = n,$$

where $\delta = (1, 1, \ldots, 1)$. So this δ is a direction of constancy of the multinomial family of distributions with the "try III" canonical parameterization.

6.2 Mean-Value Parameters

For a regular full exponential family, use (5) to define

$$\mu = E_\theta(y) = \nabla c(\theta). \tag{32}$$

The vector μ is called the *mean-value parameter vector.*

Theorem 2. *The mean-value parameterization of a regular full exponential family is always identifiable: every distribution in the family has a mean vector, and different distributions have different mean vectors.*

A proof is given in Appendix A.

Let us look at mean-value parameter for canonical affine submodels (Section 4.6). In this context, it helps to introduce the name *saturated model* for the original model that the submodels are submodels of. Recall that the saturated model canonical parameter vector θ and the submodel canonical parameter β are related by (28).

By the multivariable chain rule

$$\frac{\partial c_{\text{sub}}(\beta)}{\partial \beta_j} = \frac{\partial c(\theta)}{\partial \beta_j} = \sum_i \frac{\partial c(\theta)}{\partial \theta_i} \frac{\partial \theta_i}{\partial \beta_j} = \sum_i \mu_i m_{ij} \tag{33}$$

where m_{ij} are the components of the model matrix M. And this can be written in matrix notation

$$\nabla c_{\text{sub}}(\beta) = M^T \mu.$$

Of course, this has to turn out this way. We know the canonical statistic of the submodel is $M^T y$, so its mean has to be $M^T \mu$ by linearity of expectation.

Let us denote the submodel mean-value parameter by τ. Then the relations between the submodel and saturated model parameters are

$$\begin{aligned} \theta &= a + M\beta \\ \tau &= M^T \mu \end{aligned}$$

6.3 Canonical versus Mean-Value Parameters

A quote from my master's level theory notes

> Parameters are meaningless quantities. Only probabilities and expectations are meaningful.

Of course, some parameters are probabilities and expectations, but most exponential family canonical parameters are not.

A quote from *Alice in Wonderland*

> 'If there's no meaning in it,' said the King, 'that saves a world of trouble, you know, as we needn't try to find any.'

Realizing that canonical parameters are meaningless quantities "saves a world of trouble." We "needn't try to find any."

We have to have canonical parameters to because PMF and canonical affine submodels are specified in terms of them. But we want our interpretations to be in terms of mean-value parameters, since those are the ones directly related to probabilities and expectations.

This is even more obvious when we specify canonical linear submodels using the R formula mini-language and we have categorical predictors. There are a great many arbitrary choices for the canonical parameters of a canonical linear submodel (Section 6.7.11 below). But expected cell counts (mean-value parameters) do not change so long as the submodel is the same.

6.4 Maximum Likelihood

Theorem 3. *For a regular full exponential family, MLE need not exist, but if they do they are points where the first derivative vector of the log likelihood is zero. Moreover, every point where the first derivative vector of the log likelihood is zero is an MLE. If the canonical parameterization is identifiable, then the MLE for the canonical parameter vector is unique. MLE for the mean-value parameter are always unique. If the canonical parameterization is not identifiable and $\hat{\theta}_1$ and $\hat{\theta}_2$ are distinct MLE for the canonical parameter, then $\hat{\theta}_1 - \hat{\theta}_2$ is a direction of constancy. Consequently, different MLE always correspond to the same probability distribution.*

This is Corollary 2 in Geyer (2009).

With the log likelihood given by (1), the first derivative vector is

$$\begin{aligned} \nabla l(\theta) &= y - \nabla c(\theta) \\ &= y - E_\theta(y) \\ &= y - \mu \end{aligned}$$

where in the last line we need to think of the mean-value parameter as a function of the canonical parameter. Setting this equal to zero and solving gives

$$\hat{\mu} = y \tag{34}$$

This says

> the MLE for the mean-value parameter ($\hat{\mu}$) is the observed value of the canonical statistic vector y

or, for short,

> observed equals expected.

This is the most important property of maximum likelihood. It is the key to interpreting MLE for regular full exponential families.

Let us look at maximum likelihood for canonical affine submodels (Section 4.6). By (33)

$$\nabla l_{\text{sub}}(\beta) = M^T(y - \mu) = M^T y - \tau. \tag{35}$$

Again the MLE is a point where the first derivative vector of the log likelihood is zero, so now we have

$$\hat{\tau} = M^T \hat{\mu} = M^T y \tag{36}$$

and this is

> a canonical affine submodel of a regular full exponential family is itself a regular full exponential family, so the MLE for its mean-value parameter ($M^T \hat{\mu}$) is the observed value of its canonical statistic vector $M^T y$

and we still say "observed equals expected" for short.

Equation (36) is not useful for computation. The PMF of the family are defined in terms of the canonical parameters (8). So we need MLE for canonical parameters, and the only way to find them is, in general, to maximize the log likelihood.

But "observed equals expected" is the key to *interpreting* MLE for regular full exponential families. It is the only simple property the MLE have. In regard to interpreting MLE, the mean-value parameters are meaningful and the canonical parameters meaningless.

6.5 Fisher Information

Applying the multivariable chain rule again to (33) gives

$$\begin{aligned}
\frac{\partial^2 c(\theta)}{\partial \beta_j \partial \beta_k} &= \sum_i \sum_r \frac{\partial^2 l(\theta)}{\partial \theta_i \theta_r} \frac{\partial \theta_i}{\partial \beta_j} \frac{\partial \theta_r}{\partial \beta_k} + \sum_i \frac{\partial l(\theta)}{\partial \theta_i} \frac{\partial^2 \theta_i}{\partial \beta_j \partial \beta_k} \\
&= \sum_i \sum_r \frac{\partial^2 l(\theta)}{\partial \theta_i \theta_r} m_{ij} m_{rk}
\end{aligned}$$

And this can be written in matrix notation

$$\nabla^2 c_{\text{sub}}(\beta) = M^T \left[\nabla^2 c(\theta) \right] M$$

Remembering the formula for Fisher information

$$I(\theta) = -E\{\nabla^2 l(\theta)\}$$

we see that for exponential families we have

$$\begin{aligned}\nabla l(\theta) &= y - \nabla c(\theta) \\ \nabla^2 l(\theta) &= -\nabla^2 c(\theta)\end{aligned}$$

and expectation of a constant is that constant, so

$$I(\theta) = \nabla^2 c(\theta)$$

the Fisher information matrix for the canonical parameter is the second derivative matrix of the cumulant function. Using our work above on submodels, we see that

$$I(\beta) = M^T I(\theta) M \tag{37}$$

where we are indulging in "abuse of notation" in using the same letter for two different functions. Here $I(\beta)$ is the Fisher information matrix for the submodel canonical parameter and $I(\theta)$ is the Fisher information matrix for the saturated model canonical parameter. Perhaps it would be better to write

$$I_{\text{sub}}(\beta) = M^T I_{\text{sat}}(\theta) M$$

and we are also assuming that θ and β are related by (28), that θ and β are values of the saturated model and submodel canonical parameters that correspond to the same distribution.

For the canonical parameter vector of a regular full exponential family there is no difference between observed and expected Fisher information, because observed Fisher information is already nonrandom.

Now we (or the computer) are ready to do any form of likelihood inference. We know the log likelihood and Fisher information, and everything we want to do is a function of them (Wilks, Rao, Wald hypothesis tests and confidence intervals).

6.6 Sufficient Dimension Reduction Revisited

Any one-to-one function of a sufficient statistic vector is another sufficient statistic vector. Thus if we are using identifiable parameterizations, MLE of all parameters are sufficient statistic vectors. So are other one-to-one functions of any of the sufficient statistic vectors we know.

6.7 Examples III

6.7.1 Data

We use the example data from Table 2.6 in Agresti (2013) which is about death penalty verdicts in court cases cross-classified by race of victim and race of defendant. Rather than put the data in a table the way Agresti does, we put them in the form that the R function `glm` likes. For binomial data, the response is specified "as a two-column matrix with the columns giving the numbers of successes and failures" (quoted from `help(glm)`). Here we, perhaps insensitively, have coded the first column in Agresti's table as "success" and that happens to be death penalty = `"yes"`. Mathematically, it doesn't matter which we call success and which failure so long as we keep track of which is which. The categorical predictors `victim` and `defendant` are vectors of class `factor`, as is usual for regression on categorical variables.

```
victim <- factor(rep(c("white", "black"), each = 2))
defendant <- factor(rep(c("white", "black"), times = 2))
deathpenalty <- matrix(c(53, 11, 0, 4, 414, 37, 16, 139),
        ncol = 2, dimnames = list(NULL, c("yes", "no")))
```

We check that our data matches the data in Agresti (2013).

```
data.frame(victim, defendant, deathpenalty)

##   victim defendant yes  no
## 1  white     white  53 414
## 2  white     black  11  37
## 3  black     white   0  16
## 4  black     black   4 139
```

6.7.2 Fit

Now we can try a logistic regression.

```
gout <- glm(deathpenalty ~ victim + defendant,
    family = binomial, x = TRUE)
gout.summary <- summary(gout)
gout.summary
```

```
##
## Call:
## glm(formula = deathpenalty ~ victim + defendant, family = binomial,
##     x = TRUE)
##
## Deviance Residuals:
##        1         2         3         4
##  0.02660  -0.06232  -0.60535   0.09379
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -3.5961     0.5069  -7.094 1.30e-12 ***
## victimwhite      2.4044     0.6006   4.003 6.25e-05 ***
## defendantwhite  -0.8678     0.3671  -2.364   0.0181 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22.26591  on 3  degrees of freedom
## Residual deviance:  0.37984  on 1  degrees of freedom
## AIC: 19.3
##
## Number of Fisher Scoring iterations: 4
```

This is an example of logistic regression with response vector having dimension four and model matrix

```
modmat <- gout$x
modmat

##   (Intercept) victimwhite defendantwhite
## 1           1           1              1
## 2           1           1              0
## 3           1           0              1
## 4           1           0              0
## attr(,"assign")
## [1] 0 1 2
## attr(,"contrasts")
## attr(,"contrasts")$victim
## [1] "contr.treatment"
##
## attr(,"contrasts")$defendant
## [1] "contr.treatment"
```

6.7.3 Observed Equals Expected

Now we want to check "observed equals expected" (it must check, of course, unless the R function `glm` is buggy, which we don't expect — we just want to see for ourselves).

```
y <- deathpenalty[ , "yes"]
n <- rowSums(deathpenalty)
mu <- predict(gout, type = "response")
y

## [1] 53 11  0  4

n

## [1] 467  48  16 143

mu

##          1          2          3          4
## 0.11310026 0.23296207 0.01138621 0.02669805
```

What the R function `glm` is calling mean values obviously aren't. They appear to be probabilities rather than means. That's right. Quoting from `help(predict.glm)`

> type: the type of prediction required. The default is on the scale of the linear predictors; the alternative `"response"` is on the scale of the response variable. Thus for a default binomial model the default predictions are of log-odds (probabilities on logit scale) and `type = "response"` gives the predicted probabilities.

So we need to multiply $\mu = np$

```r
mu <- n * mu
```

and now we check

```r
all.equal(t(modmat) %*% mu, t(modmat) %*% y)

## [1] TRUE
```

we use the R function `all.equal`, which allows for small error due to inexactness of computer arithmetic, to compare "real" numbers in the computer (understanding that the computer's "real" numbers are not real real numbers).

We see the fits from the R function `glm` with `family = binomial` and the default link, which is `"logit"`, do indeed satisfy the observed equals expected property. And what are the elements of the submodel canonical statistic $M^T y$? Obviously

- total number of death-penalty verdicts
- total number of death-penalty verdicts when victim was white
- total number of death-penalty verdicts when defendant was white

Subtracting either of the latter two from the first gives

- total number of death-penalty verdicts when victim was black
- total number of death-penalty verdicts when defendant was black

So observed and (MLE estimates of) expected for these quantities are also equal. And subtracting any of these from the corresponding sample sizes (which are considered non-random for binomial GLM) gives

- total number of non-death-penalty verdicts
- total number of non-death-penalty verdicts when victim was white
- total number of non-death-penalty verdicts when defendant was white
- total number of non-death-penalty verdicts when victim was black
- total number of non-death-penalty verdicts when defendant was black

So observed equals expected for these quantities too.

6.7.4 Wald Tests

The printout of the R function `summary.glm` gives the P-values for (two-tailed) Wald tests of null hypotheses that the various parameters are zero. The Wald test comparing this model (alternative hypothesis) with its submodel that sets the submodel canonical parameter named `victimwhite` to zero says this parameter is highly statistically significant ($P = 6.247 \times 10^{-5}$) and the analogous test for the submodel canonical parameter named `defendantwhite` says this parameter is statistically significant ($P = 0.0181$). The other parameter is uninteresting.

Of course, as sophisticated statisticians, we all know that correlation is not causation and regression isn't either. Since this is not data from a controlled experiment, it only shows association not causation. But what the P-values do say is that if we dropped the predictor variable `victim` from the model, that model would not fit the data as well. And similarly for dropping `defendant`.

Why there were statistically significantly more death penalty verdicts in these data when the victim was white and statistically significantly fewer death penalty verdicts in these data when the defendant was white is something these data cannot answer. Given American history, one suspects racial prejudice, but by who, when, or where is not something these data say anything about. Nor is it even certain, from these data, that that is the causal explanation. Correlation is not causation even when you don't have a story to explain why not.

We should also comment about, what is again obvious to us as sophisticated statisticians, that these procedures are approximate and that asymptotic distributions are nearly correct for large n in terms of *absolute error* ($|\text{approximate} - \text{exact}|$) not *relative error* ($|\text{approximate} - \text{exact}| / |\text{exact}|$).

A number like $P = 6.247 \times 10^{-5}$ has relatively small absolute error. The correct interpretation is that the true P-value is very small, less than 0.001, say, if n is large enough. There is no justfication for taking even the order of magnitude of the asymptotic P-value seriously. That would be assuming small relative error, not small absolute error. No matter how small the asymptotic P-values are, say 10^{-12}, the interpretation is still the same: very small, less than 0.001, say, if n is large enough. The farther the observed value of the test statistic is out in the tail of its asymptotic distribution under the null hypothesis, the worse the asymptotic approximation is in terms of relative error. Here, with the sample sizes

```
n

## [1] 467  48  16 143
```

not humongous, maybe we shouldn't be sure that $P \approx 6.247 \times 10^{-5}$ even means $P < 0.001$.

6.7.5 Wald Confidence Intervals

We extract the estimates and standard errors from the output of the method of R generic function `summary` for objects of class `"glm"`

```
beta <- gout.summary$coefficients[ , "Estimate"]
beta.stderr <- gout.summary$coefficients[ , "Std. Error"]
```

and then the Wald confidence intervals for confidence level

```
conf.level <- 0.95
```

are

```
crit <- qnorm((1 + conf.level) / 2)
crit

## [1] 1.959964

beta["victimwhite"] + c(-1, 1) * crit *
    beta.stderr["victimwhite"]

## [1] 1.227258 3.581629

beta["defendantwhite"] + c(-1, 1) * crit *
    beta.stderr["defendantwhite"]

## [1] -1.5872489 -0.1483444
```

6.7.6 Likelihood Ratio Tests

To do likelihood ratio (Wilks) tests, we need to actually fit the models that drop these parameters.

```
drop1(gout, ~ victim, test = "LRT")

## Single term deletions
##
## Model:
## deathpenalty ~ victim + defendant
##        Df Deviance   AIC   LRT Pr(>Chi)
## <none>      0.3798 19.30
## victim  1  20.7298 37.65 20.35 6.45e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since your humble author does not regularly use the R function `drop1` and wants to check that this is indeed doing what he thinks it is doing, we check it.

```
gout.no.victim <- glm(deathpenalty ~ defendant,
    family = binomial)
anova(gout.no.victim, gout, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: deathpenalty ~ defendant
## Model 2: deathpenalty ~ victim + defendant
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         2    20.7298
## 2         1     0.3798  1    20.35 6.45e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We do indeed get the same test statistic and P-value both ways, so the following must be correct too.

```
drop1(gout, ~ defendant, test = "LRT")

## Single term deletions
##
## Model:
```

```
## deathpenalty ~ victim + defendant
##           Df Deviance    AIC    LRT Pr(>Chi)
## <none>          0.3798 19.300
## defendant  1   5.3940 22.314 5.0142  0.02514 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the Wald and Wilks tests do not agree exactly, although they do agree approximately.

6.7.7 Likelihood Ratio Confidence Intervals

The R recommended package `MASS` (Ripley, et al., 2015), which goes with the book Venables and Ripley (2002), has a function `confint.glm` which purports to do likelihood based confidence intervals (and we saw that it agreed with clearly correct calculations in the one-dimensional case in a previous handout). Let's try that

```
library(MASS)
confint(gout)

## Waiting for profiling to be done...

##                    2.5 %     97.5 %
## (Intercept)    -4.775566 -2.7349458
## victimwhite     1.306854  3.7176025
## defendantwhite -1.563332 -0.1140439
```

Again, these agree approximately but not exactly with the Wald intervals.

6.7.8 Rao Tests

The R function `drop1` also has an option that purports to do Rao tests.

```
drop1(gout, ~ victim, test = "Rao")

## Single term deletions
##
```

```
## Model:
## deathpenalty ~ victim + defendant
##        Df Deviance   AIC Rao score  Pr(>Chi)
## <none>      0.3798 19.30
## victim  1  20.7298 37.65    19.638 9.359e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

drop1(gout, ~ defendant, test = "Rao")

## Single term deletions
##
## Model:
## deathpenalty ~ victim + defendant
##           Df Deviance    AIC Rao score Pr(>Chi)
## <none>          0.3798 19.300
## defendant  1    5.3940 22.314    5.8089  0.01595 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Again, these agree approximately but not exactly with the Wald and Wilks tests.

6.7.9 Summary

Test statistics and asymptotic P-values for test of dropping `victim` from the model

	Test Statistic	P-value
Wald	16.03	6.25×10^{-5}
Wilks	20.35	6.45×10^{-6}
Rao	19.64	9.36×10^{-6}

Test statistics and asymptotic P-values for test of dropping `defendant` from the model

	Test Statistic	P-value
Wald	5.59	0.0181
Wilks	5.01	0.0251
Rao	5.81	0.0159

95% confidence intervals for the submodel canonical parameter named `victimwhite`

	lower	upper
Wald	1.23	3.58
Wilks	1.31	3.72

95% confidence intervals for the submodel canonical parameter named `defendantwhite`

	lower	upper
Wald	-1.59	-0.15
Wilks	-1.56	-0.11

There does not seem to be any convenient way in R to do Rao intervals (a. k. a. score intervals).

6.7.10 Wald Intervals for Mean-Value Parameters

The R function `predict.glm` also does confidence intervals for mean-value parameters.

```
pout <- predict(gout, se.fit = TRUE, type = "response")
low <- pout$fit - crit * pout$se.fit
hig <- pout$fit + crit * pout$se.fit
cbind(low, hig)

##             low        hig
## 1  0.0844268406 0.14177368
## 2  0.1145521378 0.35137200
## 3 -0.0022005251 0.02497295
## 4  0.0008808397 0.05251525
```

As always, there is nothing that keeps these confidence intervals in the parameter space. And we have gone out. These are MLE of cell probabilities and one of the endpoints in negative.

We can avoid this behavior by making Wald intervals for the saturated model canonical parameters and mapping them to the mean-value parameter scale.

```
pout <- predict(gout, se.fit = TRUE)
low <- pout$fit - crit * pout$se.fit
hig <- pout$fit + crit * pout$se.fit
invlogit <- function(theta) 1 / (1 + exp(- theta))
low <- invlogit(low)
hig <- invlogit(hig)
cbind(low, hig)

##           low        hig
## 1 0.087439334 0.14509456
## 2 0.135367242 0.37074835
## 3 0.003432917 0.03707991
## 4 0.010054429 0.06897312
```

The latter seems better for small n. Of course, the two methods are asymptotically equivalent.

Note: these are confidence intervals for cell probabilities rather than expected cell counts. Each interval would have to be multiplied by the corresponding sample size to get the latter.

6.7.11 Arbitrariness of Canonical Parameterization

This section is a short riff on the meaningless of canonical parameters. It shows various ways in which specifying the model differently would have lead to very different canonical parameters for *exactly the same model.*

First principle: when there are any categorical predictors, models with and without intercept are the same

```
gout.foo <- glm(deathpenalty ~ 0 + victim + defendant,
    family = binomial)
coefficients(gout)

##   (Intercept)   victimwhite defendantwhite
##    -3.5961040     2.4044434     -0.8677967

coefficients(gout.foo)

##   victimblack   victimwhite defendantwhite
##    -3.5961040    -1.1916606     -0.8677967
```

```
mu.foo <- n * predict(gout.foo, type = "response")
all.equal(mu.foo, mu)

## [1] TRUE
```

The submodel canonical parameters don't even have the same names, and for some that do have the same names, the values have changed. But the mean values are the same, so the MLE probability distribution is the same.

Second principle: when there are multiple categorical predictors, the order of terms in the formula can matter

```
gout.bar <- glm(deathpenalty ~ 0 + defendant + victim,
    family = binomial)
coefficients(gout.foo)

##    victimblack    victimwhite defendantwhite
##     -3.5961040     -1.1916606     -0.8677967

coefficients(gout.bar)

## defendantblack defendantwhite    victimwhite
##      -3.596104      -4.463901       2.404443

mu.bar <- n * predict(gout.bar, type = "response")
all.equal(mu.bar, mu)

## [1] TRUE
```

One number is the same in the canonical parameter estimates, but for parameters that are named differently! The parameters that are named the same have different numbers! But the model is still the same.

Third principle: when R sets up a factor, the default order of factor levels is alphabetical, but it doesn't have to be. Then when setting up model matrices it drops the dummy variable for the first factor level. But which was first was arbitrary.

```
victim.save <- victim
victim <- factor(as.character(victim),
    levels = rev(levels(victim)))
victim.save
```

```
## [1] white white black black
## Levels: black white

victim

## [1] white white black black
## Levels: white black

gout.baz <- glm(deathpenalty ~ victim + defendant,
    family = binomial)
coefficients(gout)

##    (Intercept)    victimwhite defendantwhite
##     -3.5961040      2.4044434     -0.8677967

coefficients(gout.baz)

##    (Intercept)    victimblack defendantwhite
##     -1.1916606     -2.4044434     -0.8677967

mu.baz <- n * predict(gout.baz, type = "response")
all.equal(mu.baz, mu)

## [1] TRUE
```

Again, the names of the canonical parameters change. And the values of some parameters with the same name stay the same while the values of other parameters with the same name are different. But the model remains the same.

Put `victim` back the way it was.

```
victim <- victim.save
```

Forth principle: the coding of response labels is arbitrary, which of the `deathpenalty` values is considered “success” and which “failure” is arbitrary. Reverse them.

```
deathpenalty.save <- deathpenalty
deathpenalty <- deathpenalty[ , 2:1]
deathpenalty
```

```
##       no yes
## [1,] 414  53
## [2,]  37  11
## [3,]  16   0
## [4,] 139   4

gout.qux <- glm(deathpenalty ~ victim + defendant,
    family = binomial)
coefficients(gout)

##   (Intercept)   victimwhite defendantwhite
##    -3.5961040     2.4044434     -0.8677967

coefficients(gout.qux)

##   (Intercept)   victimwhite defendantwhite
##     3.5961040    -2.4044434      0.8677967

mu.qux <- n * predict(gout.qux, type = "response")
all.equal(n - mu.qux, mu)

## [1] TRUE
```

The canonical parameters all have the same names, but all now have opposite signs. In showing that the means are the same so the model is the same, we have to take into account that `mu.qux` is predicted “successes” by the second definition (death penalty “no”) and that `mu` is predicted “successes” by the first definition (death penalty “yes”) so one is `n` minus the other.

All of the above, and even more ways that could be illustrated with more complicated models (we could play lots of games with “interactions” if we had them), is what we mean when we say canonical parameters are meaningless.

Put `deathpenalty` back the way it was.

```
deathpenalty <- deathpenalty.save
```

For more examples of the arbitrariness of canonical parameterization, see Section 7.5 below.

7 Different Sampling Schemes

A very important point about categorical data analysis (analysis of contingency table data) is that one gets the same results regardless of whether the assumed sampling scheme "Poisson," "multinomial," or "product multinomial." First we have to explain what these are.

7.1 Sampling Schemes

A *partition* of a set S is a family $\mathcal{A}$ of subsets of of S having the property that for every $x \in S$ there is exactly one $A \in \mathcal{A}$ such that $x \in A$. If $\mathcal{A}$ and $\mathcal{B}$ are two partitions of S, we say that $\mathcal{A}$ is *finer* than $\mathcal{B}$ or that $\mathcal{B}$ is *coarser* than $\mathcal{A}$ if for every $A \in \mathcal{A}$ there is exactly one $B \in \mathcal{B}$ such that $A \subset B$.

Suppose we have a contingency table with k categories. Let I denote the index set

$$I = \{1, 2, \ldots, k\}.$$

Here we are interested in partitions of I.

We can think of vectors as functions $I \mapsto \mathbb{R}$. A vector y is the function $i \mapsto y_i$. For any subset A of I, let y_A denote the "subvector" of y whose components have indices in A. Considered as functions, y and y_A have the same rule $i \mapsto y_i$ but different domains: but y has domain I and y_A has domain A.

The point is that components of y_A know which components of y they are equal to. If $A = \{3, 7, 10\}$, then the components of y_A are y_3, y_7, and y_{10}. We don't renumber them to have the indices be 1, 2, 3 out of some misguided notation that components of three-dimensional vectors have to be numbered 1, 2, 3 or it will make us cry (or whatever). If we don't renumber, then the notation "remembers" which components of y correspond to which components of y_A. If we insist on renumbering then we have lost the correspondence between components of y and components of y_A and have to keep track of it by some additional, and probably very clumsy, notation. What was called a "misguided notation" that vectors are indexed by consecutive numbers starting with one is responsible for more clumsy notation in statistics than any other idea.

The set of all such functions is denoted $\mathbb{R}^I$ in set theory. So y is an element of $\mathbb{R}^I$ and y_A is an element of $\mathbb{R}^A$. In set theory, everything is a set. The number d is the set $\{0, 1, 2, \ldots, d-1\}$. So in set theory $\mathbb{R}^d$ means $\mathbb{R}^S$ where $S = \{0, 1, 2, \ldots, d-1\}$. Thus our new notation for finite-dimensional vector spaces $\mathbb{R}^S$ for some finite set S essentially the same as the conventional notation $\mathbb{R}^d$ for some nonnegative integer d. The only difference is that set theory starts counting at zero instead of at one.

The kinds of sampling schemes commonly assumed for contingency tables are the following.

Poisson The cell counts are independent Poisson random variables.

multinomial The cell counts are components of a multinomial random vector.

product multinomial The index set I has a partition $\mathcal{A}$, and the vector y of cell counts is similarly partitioned. The subvectors y_A for $A \in \mathcal{A}$ are independent multinomial random vectors.

It is called "product multinomial" because independence implies the joint PMF is the product of the marginal PMF's

$$f(y) = \prod_{A \in \mathcal{A}} f(y_A)$$

(abusing notation by denoting different functions by the same letter f).

7.2 Sampling Schemes and Conditioning

These sampling schemes are related as follows.

- Multinomial is Poisson conditioned on the sample size. If f is the PMF for the Poisson scheme, then the PMF for the multinomial scheme is

  $$f\left(y \,\middle|\, \textstyle\sum_{i \in I} y_i = n\right) \tag{38}$$

  where n is the sample size.

- Product multinomial is Poisson conditioned on the sample sizes for the elements of the partition. If f is the PMF for the Poisson scheme, then the PMF for the product multinomial scheme for partition $\mathcal{A}$ is

  $$\textstyle\prod_{A \in \mathcal{A}} f\left(y_A \,\middle|\, \sum_{i \in A} y_i = n_A\right) \tag{39}$$

  where n_A is the sample size for subvector y_A.

- Product multinomial is also multinomial conditioned on the sample sizes for the elements of the partition. If f is the PMF for the multinomial scheme, then the PMF for the product multinomial scheme for partition $\mathcal{A}$ is again given by (39).

- Product multinomial is also product multinomial for a coarser partition conditioned on the sample sizes for the elements of a finer partition. Suppose $\mathcal{A}$ and $\mathcal{B}$ are partitions of the index set I with $\mathcal{A}$ finer than $\mathcal{B}$. If f is the PMF for the product multinomial scheme for partition $\mathcal{B}$, then the PMF for the product multinomial scheme for partion $\mathcal{A}$ is again given by (39).

7.3 Sampling Schemes and Maximum Likelihood Estimates

7.3.1 Poisson versus Multinomial

The key is to understand Poisson versus multinomial. When we later consider product multinomial, the principles remain the same, the math gets messier but not harder.

Suppose we assume Poisson sampling, have a canonical affine submodel with parameterization (28), observed data y, cumulant function c given by

$$c(\theta) = \sum_{i \in I} e^{\theta_i}, \tag{40}$$

and MLE's

- $\hat{\beta}$ (for the submodel canonical parameter β),
- $\hat{\theta}$ (for the saturated model canonical parameter θ),
- $\hat{\mu}$, (for the saturated model mean-value parameter μ), and
- $\hat{\tau}$, (for the submodel mean-value parameter τ),

which, of course, are related by

$$\hat{\theta} = a + M\hat{\beta} \tag{41a}$$
$$\hat{\mu} = \nabla c(\hat{\theta}) \tag{41b}$$
$$\hat{\tau} = M^T \hat{\mu} \tag{41c}$$

and are all determined by the observed equals expected property

$$M^T y = \hat{\tau}. \tag{42}$$

If the submodel parameterization is identifiable, then $\hat{\beta}$ is the unique submodel canonical parameter value that makes (41a), (41b), and (41c) lead to (42).

If the submodel parameterization is not identifiable, then $\hat{\beta}$ is a (non-unique) submodel canonical parameter value that makes (41a), (41b), and (41c) lead to (42).

In the latter case there is a direction δ in the submodel parameter space that is a direction of constancy,

$$\langle M^T y, \delta \rangle = y^T M \delta$$

is a constant random variable, and $\hat{\beta} + r\delta$ is also an MLE for β for any real number r.

Now we consider the multinomial sampling scheme. Since we want to compare Poisson and multinomial, we use the same saturated model canonical statistic vector y for both. This means we cannot have an identifiable parameterization for the saturated multinomial. Instead we have what we called the "try III" parameterization in Section 2.4 above. Then we repeat everything we said above about the Poisson for the multinomial and everything is the same except that the Poisson cumulant function is (40) whereas the multinomial cumulant function is (23).

Now we make a key assumption. The model for the Poisson sampling scheme must include an "intercept" term or be equivalent to one that does. We saw in the preceding section that if there are any categorical predictors, then leaving out the "intercept" does not change the model (hence the scare quotes around "intercept"). Formally the requirement is that the vector $\delta_I = (1, 1, \ldots, 1)$ be in the column space of the model matrix M, that is, some linear combination of the columns of M has all components equal to one.

Theorem 4. *Suppose we have a Poisson regression model with the parameterization of Section 5.2 and a multinomial model with the parameterization of Section 2.4.3. Suppose we use the same canonical affine submodel for both with parameterization* (28). *And suppose the vector with all components equal to one is in the column space of the model matrix. Then MLE for the models agree in the sense that, for the same data, any MLE for the Poisson model for any of the four parameter vectors, submodel canonical, submodel mean-value, saturated model canonical, or saturated model mean-value, is also an MLE for the corresponding parameter vector of the multinomial model.*

Since only part of this theorem is usually stated in textbooks, we give a proof. Observe that it is crucial that we use the nonidentifiable "try III" parameterization of the multinomial to make canonical parameters match.

Proof. Since both models have the same canonical statistic vector and the same model matrix, they have the same "observed equals expected" equation (42) determining they have the same MLE for the submodel mean-value parameter.

In fact, we see that the Poisson and multinomial sampling schemes not only agree on (42), they also agree on (41a) and (41c). They only disagree on (41b), where for the Poisson scheme we differentiate (40) obtaining

$$\hat{\mu}_i = e^{\hat{\theta}_i} \tag{43}$$

and for the multinomial scheme we differentiate (23) obtaining

$$\hat{\mu}_i = \frac{n e^{\hat{\theta}_i}}{\sum_{j \in I} e^{\hat{\theta}_j}} \tag{44}$$

Suppose $\hat{\theta}$ is the Poisson MLE. Then (43) holds. Because of the key assumption that δ_I is in the column space of the model matrix, we have, by "observed equals expected"

$$\delta_I^T \hat{\mu} = \sum_{i \in I} \hat{\mu}_i = \delta_I^T y = \sum_{i \in I} y_i. \tag{45}$$

Introduce an abbreviation

$$n = \sum_{i \in I} y_i. \tag{46}$$

(For the multinomial scheme n is the multinomial sample size, but here, for the Poisson scheme, n isn't anything in particular, it is just an abbreviation for the right-hand side of (46)). Plugging (46) into (45) and then using (43) gives

$$\sum_{i \in I} \hat{\mu}_i = \sum_{i \in I} e^{\hat{\theta}_i} = n. \tag{47}$$

All of this holds for the Poisson MLE's. Now we want to show that (44) also holds for the Poisson MLE's. Plug (47) into (44) obtaining

$$\hat{\mu}_i = \frac{n e^{\hat{\theta}_i}}{\sum_{j \in I} e^{\hat{\theta}_j}} = \frac{n e^{\hat{\theta}_i}}{n} = e^{\hat{\theta}_i}$$

and, reading end to end, this is just (43) which characterizes the Poisson MLE. Hence we conclude that if $\hat{\theta}$ is the Poisson MLE, then (43) and (44) both hold. But (44) characterizes multinomial MLE. Hence the Poisson MLE is also a multinomial MLE.

As we said before, since (42), (41a), and (41c) obviously agree for both schemes, and we have just shown that (41b) also agrees for both schemes, all MLE for all parameterizations, $\hat{\beta}$, $\hat{\theta}$, $\hat{\mu}$, and $\hat{\tau}$ agree for both schemes. □

7.3.2 Poisson versus Product Multinomial

Theorem 5. *Suppose we have a Poisson regression model with the parameterization of Section 5.2 and a product multinomial model for partition $\mathcal{A}$, each multinomial of the product having the parameterization of Section 2.4.3. Suppose we use the same canonical affine submodel for both with parameterization (28). Define for each $A \in \mathcal{A}$ the indicator vectors δ_A, the i-th component of which is zero if $i \notin A$ and one if $i \in A$. And suppose that each of these δ_A is in the column space of the model matrix. Then MLE for the models agree in the sense that, for the same data, any MLE for the Poisson model for any of the four parameter vectors, submodel canonical, submodel mean-value, saturated model canonical, or saturated model mean-value, is also an MLE for the corresponding parameter vector of the product multinomial model.*

Proof. The proof is just like the proof of Theorem 4 except that the multinomial cumulant function (23) is replaced by the product multinomial cumulant function

$$c(\theta) = \sum_{A \in \mathcal{A}} n_A \log \left(\sum_{i \in A} e^{\theta_i} \right). \tag{48}$$

Differentiating and plugging in $\hat{\theta}$ for θ gives

$$\hat{\mu}_i = \frac{n_A e^{\hat{\theta}_i}}{\sum_{j \in A} e^{\hat{\theta}_j}}, \qquad i \in A. \tag{49}$$

So in this proof (49) plays the role of (44) in the proof of Theorem 4.

Then, because of the assumption that δ_A is in the column space of the model matrix, we have

$$\delta_A^T \hat{\mu} = \sum_{i \in A} \hat{\mu}_i = \delta_A^T y = \sum_{i \in A} y_i. \tag{50}$$

holding for the Poisson MLE's. This replaces (45) in the other proof. Then we just follow the pattern of the other proof, introducing

$$n_A = \sum_{i \in A} y_i \tag{51}$$

we see that (50) implies, still for Poisson MLE's, that

$$\sum_{i \in A} e^{\hat{\theta}_i} = n_A \tag{52}$$

and plugging this into (49) we see that (49) becomes (43). So the equation that characterizes product multinomial MLE's actually holds for Poisson MLE's. The rest of the two proofs is the same. □

7.3.3 Multinomial versus Product Multinomial

Theorem 6. *Suppose we have a multinomial regression model and a product multinomial model for partition $\mathcal{A}$, each multinomial having the parameterization of Section 2.4.3. Suppose we use the same canonical affine submodel for both with parameterization* (28). *Define δ_A as in Theorem 5, and suppose that each of these δ_A is in the column space of the model matrix. Then MLE for the models agree in the sense that, for the same data, any MLE for the multinomial model for any of the four parameter vectors, submodel canonical, submodel mean-value, saturated model canonical, or saturated model mean-value, is also an MLE for the corresponding parameter vector of the product multinomial model.*

Proof. The proof is just like the proofs of Theorems 4 and 5 except that now start with multinomial MLE's satisfying (44) and need to show that they also satisfy (49).

Again, because of the assumption that δ_A is in the column space of the model matrix, we have (50), and, introducing the abbreviation (51) we have

$$n_A = \sum_{i \in A} \hat{\mu}_i = \sum_{i \in A} \frac{n e^{\hat{\theta}_i}}{\sum_{j \in I} e^{\hat{\theta}_j}} = \frac{n \sum_{i \in A} e^{\hat{\theta}_i}}{\sum_{j \in I} e^{\hat{\theta}_j}}. \tag{53}$$

This differs from (52) because we plugged in the formula (44) for multinomial means for $\hat{\mu}_i$ rather than the formula (43) for Poisson means.

Now we want to show that if the MLE's are for the multinomial scheme, then (49) also holds. Rewrite (53) as

$$n_A \sum_{j \in I} e^{\hat{\theta}_j} = n \sum_{i \in A} e^{\hat{\theta}_i}$$

and plug into (49) obtaining (when $i \in A$)

$$\hat{\mu}_i = \frac{n_A e^{\hat{\theta}_i}}{\sum_{j \in A} e^{\hat{\theta}_j}} = \frac{n_A e^{\hat{\theta}_i}}{\frac{n_A}{n} \sum_{j \in I} e^{\hat{\theta}_j}} = \frac{n e^{\hat{\theta}_i}}{\sum_{j \in I} e^{\hat{\theta}_j}}$$

and, reading end to end, this is (44). So the equation that characterizes product multinomial MLE's actually holds for multinomial MLE's. The rest of two proofs is the same as for the two preceding proofs. □

By now the pattern should be obvious. We won't bother to state or prove the analogous theorem for product multinomials with coarser and finer partitions.

7.3.4 Example

Instead we move to an example, the same example we have been doing all along. But now we use Poisson regression to fit the model. We consider the logistic regression we have been doing an instance of product multinomial where there are four multinomials (in this case binomials) in the product.

To do Poisson regression, we reformat the data as follows.

```
pdata <- data.frame(verdicts = as.vector(deathpenalty),
    victim = rep(victim, times = 2),
    defendant = rep(defendant, times = 2),
    deathpenalty = factor(rep(colnames(deathpenalty),
        each = 4)))
```

We had to stretch the data out into a vector `verdicts`, and correspondingly had to make the predictors `victim` and `defendant` twice as long to match and also had to make up a new predictor `deathpenalty` to indicate which elements of the new response vector `verdicts` corresponded to which columns of the old response matrix `deathpenalty` (having a response matrix rather than a response vector being a curiosity, of the way the R function `glm` specifies binomial GLM's).

The data are now

```
pdata

##   verdicts victim defendant deathpenalty
## 1       53  white     white          yes
## 2       11  white     black          yes
## 3        0  black     white          yes
## 4        4  black     black          yes
## 5      414  white     white           no
## 6       37  white     black           no
## 7       16  black     white           no
## 8      139  black     black           no
```

To fit this model by Poisson regression and have it be equal to the corresponding product binomial model (logistic regression), we need to have what Theorem 5 calls the δ_A in the model, so we need to include a term `victim * defendant` in the R formula, because the dummy variables it produces are the δ_A. (The four parts of the product binomial are for the four victim-race-defendant-race combinations).

We also have a term `deathpenalty : (victim + defendant)` in the model because we want to know how well `victim + defendant` does in "predicting" whether the death penalty is imposed or not. We did not need the `deathpenalty :` when doing logistic regressions because this was implicit in what we called the "curiosity of the way the R function `glm` specifies binomial GLM's" above (having a response matrix rather than a response vector).

So do it

```
gout.poisson <- glm(verdicts ~ victim * defendant +
    deathpenalty : (victim + defendant), family = poisson,
    data = pdata, x = TRUE)
summary(gout.poisson)

##
## Call:
## glm(formula = verdicts ~ victim * defendant + deathpenalty:(victim +
##     defendant), family = poisson, data = pdata, x = TRUE)
##
## Deviance Residuals:
##        1          2          3          4          5          6          7          8
##  0.02505  -0.05463  -0.60362   0.09251  -0.00895   0.03000   0.04572  -0.01545
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     4.93578    0.08471  58.265  < 2e-16 ***
## victimwhite                    -1.32980    0.18479  -7.196 6.19e-13 ***
## defendantwhite                 -2.17465    0.26377  -8.245  < 2e-16 ***
## victimwhite:defendantwhite      4.59497    0.31353  14.656  < 2e-16 ***
## victimblack:deathpenaltyyes    -3.59610    0.50691  -7.094 1.30e-12 ***
## victimwhite:deathpenaltyyes    -1.19166    0.33809  -3.525 0.000424 ***
## defendantwhite:deathpenaltyyes -0.86780    0.36707  -2.364 0.018074 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1225.07955  on 7  degrees of freedom
## Residual deviance:    0.37984  on 1  degrees of freedom
## AIC: 52.42
##
## Number of Fisher Scoring iterations: 3
```

And we check that we have the same mean vectors for both models, taking into account what we are now beginning to see as the weirdness of the way binomial GLM's work in R.

```
mu.binomial <- predict(gout, type = "response")
mu.binomial <- n * mu.binomial
mu.binomial <- c(mu.binomial, n - mu.binomial)
mu.poisson <- predict(gout.poisson, type = "response")
all.equal(mu.binomial, mu.poisson, check.attributes = FALSE)

## [1] TRUE
```

We needed `check.attributes = FALSE` because the names of the components are wrong (again because of the "weirdness of the way binomial GLM's work in R") and without this optional argument R function `all.equal` checks attributes, including names, as well as values. It is an error if the names don't match.

Now that we know the mean vectors match we know that, since mean vectors parameterize regular full exponential family models, we have matching MLE probability distributions.

Now look at saturated model canonical parameters.

```
theta.binomial <- predict(gout)
theta.binomial <- c(theta.binomial,
    rep(0, length(theta.binomial)))
theta.binomial <- as.vector(theta.binomial)
theta.poisson <- predict(gout.poisson)
theta.poisson <- as.vector(theta.poisson)
matrix(theta.binomial - theta.poisson, ncol = 2)

##            [,1]      [,2]
## [1,] -6.026306 -6.026306
## [2,] -3.605982 -3.605982
## [3,] -2.761137 -2.761137
## [4,] -4.935784 -4.935784
```

We have formatted the difference of the two saturated model MLE canonical parameter vectors in the same way the R function `glm` requires binomial response vectors be formatted (in a matrix with two columns). Each row corresponds to an independent binomial in the product binomial (logistic regression model). And we know that the vector with all all components equal to one is a direction of constancy for a multinomial (with "try III" parameterization, Section 2.4.3). Hence from Theorem 1 so is any vector with all components equal to each other, because if $\langle y, \delta \rangle$ is almost surely constant, then so is $\langle y, a\delta \rangle = a\langle y, \delta \rangle$ for any constant a.

So we see that the difference of the two saturated model MLE canonical parameter vectors is a direction of constancy of the product binomial model, as must be from the theory of exponential families.

Now look at submodel model canonical parameters, and we get in trouble. The R formula mini-language doesn't give us anything comparable, even the lengths of these vectors

```
names(coefficients(gout))

## [1] "(Intercept)"    "victimwhite"    "defendantwhite"

names(coefficients(gout.poisson))

## [1] "(Intercept)"
## [2] "victimwhite"
## [3] "defendantwhite"
## [4] "victimwhite:defendantwhite"
## [5] "victimblack:deathpenaltyyes"
## [6] "victimwhite:deathpenaltyyes"
## [7] "defendantwhite:deathpenaltyyes"
```

We could get a closer match if we used different formulas to specify the models (so-called intercepts behave really weirdly in complicated situations, get rid of them, so do `*`'s for interactions, get rid of them too).

```
gout.poisson.alternate <- glm(verdicts ~ 0 + victim : defendant
    + deathpenalty : (victim + defendant), family = poisson,
    data = pdata, x = TRUE)
all.equal(predict(gout.poisson, type = "response"),
    predict(gout.poisson.alternate, type = "response"))

## [1] TRUE

gout.binomial.alternate <- glm(deathpenalty ~ 0 + victim
    + defendant, family = binomial, x = TRUE)
all.equal(predict(gout, type = "response"),
    predict(gout.binomial.alternate, type = "response"))

## [1] TRUE
```

Now the $\hat{\beta}$'s match more closely

```
coefficients(gout.binomial.alternate)

##    victimblack    victimwhite defendantwhite
##     -3.5961040     -1.1916606     -0.8677967

coefficients(gout.poisson.alternate)

##     victimblack:defendantblack
##                      4.9357837
##     victimwhite:defendantblack
##                      3.6059820
##     victimblack:defendantwhite
##                      2.7611372
##     victimwhite:defendantwhite
##                      6.0263059
##    victimblack:deathpenaltyyes
##                     -3.5961040
##    victimwhite:deathpenaltyyes
##                     -1.1916606
## defendantwhite:deathpenaltyyes
##                     -0.8677967

all.equal(as.vector(coefficients(gout.binomial.alternate)),
    as.vector(coefficients(gout.poisson.alternate))[5:7])

## [1] TRUE
```

The first four components of `coefficients(gout.poisson.alternate)` are the cofficients for what Theorem 5 calls the δ_A, the dummy variables for the sums that are fixed (multinomial sample sizes) in the product binomial model. The remaining three components of that $\hat{\beta}$ vector correspond to the three components of the $\hat{\beta}$ vector for the product binomial (logistic regression) model. The R formula mini-language doesn't name them exactly the same, but they are essentially the same parameters.

One last point before we leave this issue. Theorems 4, 5, and 6 are about using the same model matrix for both models. But here we clearly didn't do that. The reason is identifiability. The extra columns of the model matrix for `gout.poisson.alternate` correspond to directions of constancy for the other model. We need to leave them out for identifiability. We could do exactly what Theorem 5 says except that the weirdness of the way binomial GLM's work in R defeats us. So we won't bother to match that up.

7.4 Sampling Schemes and Likelihood Ratio Tests

Suppose we want to test nested canonical affine submodels with the same offset vector and model matrices M_1 and M_2. The nesting hypothesis means that the column space of M_1 is a subspace of the column space of M_2. Usually, but not always, this means that every column of M_1 is also a column of M_2. When we think of models specified by formulas , thus usually means that every term in the formula for the smaller model is also a term in the formula for the bigger model.

Theorem 7. *Not only do MLE agree for Poisson, multinomial, and product multinomial sampling schemes, so do likelihood ratio test statistics.*

Proof. We have already seem that the MLE $\hat{\beta}$, $\hat{\theta}$, $\hat{\mu}$, and $\hat{\tau}$ can be considered the same for both sampling schemes. For the Poisson scheme, the maximized log likelihood is

$$l(\hat{\beta}) = \langle y, \hat{\theta} \rangle - \sum_{i \in I} e^{\hat{\theta}_i}. \tag{54}$$

For the multinomial scheme, it is

$$l(\hat{\beta}) = \langle y, \hat{\theta} \rangle - n \log \left(\sum_{i \in I} e^{\hat{\theta}_i} \right). \tag{55}$$

For the product multinomial scheme for partition $\mathcal{A}$, it is

$$l(\hat{\beta}) = \langle y, \hat{\theta} \rangle - \sum_{A \in \mathcal{A}} n_A \log \left(\sum_{i \in A} e^{\hat{\theta}_i} \right) \tag{56}$$

and we need to show that these are the same or at least give the same results when used in constructing test statistics. Using (46) and (51), equations (54), (55), and (56) become (respectively)

$$
\begin{aligned}
l(\hat{\beta}) &= \langle y, \hat{\theta} \rangle - n \\
l(\hat{\beta}) &= \langle y, \hat{\theta} \rangle - n \log(n) \\
l(\hat{\beta}) &= \langle y, \hat{\theta} \rangle - \sum_{A \in \mathcal{A}} n_A \log(n_A)
\end{aligned}
$$

and these agree except for constants that cancel when we calculate likelihood ratio test statistics (which involve differences of log likelihoods). □

Because likelihoods are invariant under reparameterization, Theorem 7 holds for any parameterization.

7.4.1 Example Continued: Wilks Tests

Recall that we did the following likelihood ratio test back in Section 6.7.6

```
anova(gout.no.victim, gout, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: deathpenalty ~ defendant
## Model 2: deathpenalty ~ victim + defendant
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         2    20.7298
## 2         1     0.3798  1    20.35 6.45e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Recall that we put the data back in the form for logistic (product binomial) regression. Put it back in the form for Poisson regression, and fit the model with no victim effect. Since we put the data back in form for logistic regression, we have to again convert it to the form for Poisson regression.

```
gout.poisson.no.victim <- glm(verdicts ~ victim * defendant +
    deathpenalty : defendant, family = poisson,
    data = pdata, x = TRUE)
anova(gout.poisson.no.victim, gout.poisson, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: verdicts ~ victim * defendant + deathpenalty:defendant
## Model 2: verdicts ~ victim * defendant + deathpenalty:(victim + defendant)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         2    20.7298
## 2         1     0.3798  1    20.35 6.45e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The same.

7.4.2 Warning

The R function `drop1` does not do the right thing with the example of the preceding section.

```
drop1(gout.poisson, ~ victim, test = "LRT")

## Single term deletions
##
## Model:
## verdicts ~ victim * defendant + deathpenalty:(victim + defendant)
##        Df Deviance    AIC    LRT  Pr(>Chi)
## <none>       0.380  52.42
## victim  1   64.272 114.31 63.892 1.314e-15 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The reason is that it tries to drop everything involving the "predictor" `victim` (scare quotes around "predictor" because it isn't data but just an indicator variable that tells which cells of the contingency table we are talking about). If there is a way to get `drop1` to do the right thing here, your humble author cannot figure it out.

The lesson here is that `anova` is a lot safer than `drop1`. With `anova` you have R objects for both models being compared. You can do `summary` on them, look at their coefficients, and see that they are actually the models you want to compare. With `drop1` you can only look at the big model and can only hope that `drop1`'s futzing with the R formula does the right thing. And sometimes, like here, it doesn't.

7.5 Dropping Variables to Gain Identifiability

The R functions `lm` and `glm` that fit linear models (LM) and generalized linear models (GLM) automatically drop variables to gain identifiability (or avoid so-called collinearity).

Because the saturated model for an LM or GLM is always identifiable, non-identifiability for a canonical affine submodel can only arise due to the model matrix M being such that $\theta = M\beta$ is not a one-to-one change of parameter. Then different β's (submodel canonical parameter values) correspond to the same θ (saturated model canonical parameter value), so the model is obviously not identifiable.

What `lm` and `glm` and other R functions that fit regression models and have copied `lm` and `glm` do in this case is to find a maximal set of linearly independent columns of M and use the matrix M_2 composed of these columns as the model matrix. In the language of linear algebra, the columns of M_2 are a basis for the column space of M. That means that any vector $\theta = M\beta$ can also be expressed at $\theta = M\beta_2$ for some vector β_2. And this means that M and M_2 are different model matrices for the same statistical model.

Because `lm` and `glm` always pick the columns of M_2 to be a subset of columns of M, we always have M_2 a submatrix of M.

Our subvector notation (p. 47 above) can be extended to matrices to cover this case. Matrices are vectors because they can be added and multiplied by scalars. We can think of a matrix M having components m_{ij} as the function $(i, j) \mapsto m_{ij}$. If we let I and J denote the sets over which i and j range, then our matrix is an element of the vector space $\mathbb{R}^{I \times J}$. An element of $I \times J$ is a pair (i, j), so an element of $\mathbb{R}^{I \times J}$ is a function $(i, j) \mapsto m_{ij}$. Then if $A \subset J$, our subvector notation says $M_{I \times A}$ is the function $(i, j) \mapsto m_{ij}$ defined on $I \times A$. So we are just restricting j to be in A. So this is a submatrix that has some of the columns of M but not all of them.

So suppose $M_{I \times A}$ is what we were calling M_2 before. Then $M_{I \times A}\beta_A = M\beta$ if $\beta_j = 0$ for $j \notin A$. In short, using the model matrix M_2 is like constraining the components of β that correspond to columns of M that are not in M_2 to be zero. The R functions `lm` and `glm` do not say they have constrained these components of β to be zero, instead they put in `NA`.

Here is a made-up example.

```r
color <- c("red", "orange", "orange", "yellow", "yellow", "green", "green")
fruit <- c("apple", "orange", "kumquat", "apple", "lemon", "apple", "lime")
count <- rpois(length(color), 10)
foof <- data.frame(color = color, fruit = fruit, count = count)
foof
```

```
##    color   fruit count
## 1    red   apple    15
## 2 orange  orange    11
## 3 orange kumquat     6
## 4 yellow   apple    12
## 5 yellow   lemon     9
## 6  green   apple     5
## 7  green    lime    15
```

```r
gout.foof <- glm(count ~ color + fruit, family = poisson, data = foof)
summary(gout.foof)
```

```
##
## Call:
## glm(formula = count ~ color + fruit, family = poisson, data = foof)
##
## Deviance Residuals:
## [1]  0  0  0  0  0  0  0
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.6094     0.4472   3.599  0.00032 ***
## colororange    0.7885     0.5394   1.462  0.14379
## colorred       1.0986     0.5164   2.127  0.03338 *
## coloryellow    0.8755     0.5323   1.645  0.10003
## fruitkumquat  -0.6061     0.5075  -1.194  0.23236
## fruitlemon    -0.2877     0.4410  -0.652  0.51414
## fruitlime      1.0986     0.5164   2.127  0.03338 *
## fruitorange        NA         NA      NA       NA
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
```

```
##
##     Null deviance:  9.7158e+00  on 6  degrees of freedom
## Residual deviance: -3.9049e-29  on 0  degrees of freedom
## AIC: 42.893
##
## Number of Fisher Scoring iterations: 3
```

What happened? These R functions know they should drop one dummy variable for each factor if they include an intercept. So it automatically drops `colorgreen` and `fruitapple`. But it automatically keeps the rest. But here if you know that `colororange` $= 1$ and `fruitkumquat` $= 0$, then you know `fruitorange` $= 1$, because the only two orange fruits in these data are oranges and kumquats. That's non-identifiability or collinearity. So we have to drop one of these three dummy variables, and which one doesn't matter. We get the same statistical model either way, one more example of regression coefficients are meaningless. The R function `glm` decides to drop `fruitorange`.

IMHO this function should print `0` rather than `NA` for the estimate for `fruitorange`. Instead it leaves it to us to know that `NA` here means the same thing as constraining this regression coefficient to be zero.

If we change the order of the levels of the factors, so R will drop different dummy variables, we get a different collinearity.

```
color <- factor(color, levels = sort(unique(color), TRUE))
fruit <- factor(fruit, levels = sort(unique(fruit), TRUE))
foof <- data.frame(color = color, fruit = fruit, count = count)
foof

##    color   fruit count
## 1    red   apple    15
## 2 orange  orange    11
## 3 orange kumquat     6
## 4 yellow   apple    12
## 5 yellow   lemon     9
## 6  green   apple     5
## 7  green    lime    15

gout.foof <- glm(count ~ color + fruit, family = poisson,
    data = foof, x = TRUE)
coefficients(gout.foof)
```

```
##  (Intercept)     colorred  colororange   colorgreen
##   2.48490665   0.22314355  -0.08701138  -0.87546874
##    fruitlime   fruitlemon fruitkumquat   fruitapple
##   1.09861229  -0.28768207  -0.60613580           NA
```

Now it drops `fruitapple` because

```
modmat <- gout.foof$x
modmat.good <- modmat[ , colnames(modmat) != "fruitapple"]
modmat.bad <- modmat[ , colnames(modmat) == "fruitapple"]
solve(modmat.good, modmat.bad)

##  (Intercept)     colorred  colororange   colorgreen
##            1            0           -1            0
##    fruitlime   fruitlemon fruitkumquat
##           -1           -1            0
```

which says

$$\texttt{fruitapple} = \texttt{(Intercept)} - \texttt{colororange} - \texttt{fruitlime} - \texttt{fruitlemon}$$

(in these data a fruit that is not colored orange and is not a lime or a lemon is an apple). This is one more example of regression coefficients are meaningless.

If we change the order of terms in the formula, it will drop something different.

```
gout.foof <- glm(count ~ fruit + color, family = poisson,
    data = foof, x = TRUE)
coefficients(gout.foof)

##  (Intercept)    fruitlime   fruitlemon fruitkumquat
##   2.39789527   1.18562367  -0.20067070  -0.60613580
##   fruitapple     colorred  colororange   colorgreen
##   0.08701138   0.22314355           NA  -0.87546874
```

It drops `colororange` because

```
modmat <- gout.foof$x
modmat.good <- modmat[ , colnames(modmat) != "colororange"]
modmat.bad <- modmat[ , colnames(modmat) == "colororange"]
solve(modmat.good, modmat.bad)
```

```
##  (Intercept)    fruitlime    fruitlemon fruitkumquat
##            1           -1            -1            0
##   fruitapple     colorred   colorgreen
##           -1            0            0
```

which says

$$\texttt{colororange} = \texttt{(Intercept)} - \texttt{fruitlime} - \texttt{fruitlemon} - \texttt{fruitapple}$$

(in these data a fruit that is not a lime, lemon, or apple is colored orange). This is one more example of regression coefficients are meaningless.

But none of the above was what this section has to do with sampling schemes. That is the following.

If the saturated model has directions of constancy, as it does for the multinomial and product multinomial sampling schemes, to get identifiability one must also drop columns of the model matrix M that are in the constancy space of the saturated model.

If the saturated model is multinomial, then one must drop columns of M that are scalar multiples of what was called δ_I above and is labeled `"(Intercept)"` when R constructs model matrices (the vector having all components equal to one). If the saturated model is product multinomial for partition $\mathcal{A}$, then one must drop columns of M that are linear combinations of what was called δ_A, $A \in \mathcal{A}$. These are the dummy variables that are put into the model to force

$$\sum_{i \in A} y_i = \sum_{i \in A} \hat{\mu}_i, \qquad A \in \mathcal{A},$$

to be among the "observed equals expected" equations.

In either case, one drops all of these columns of M when fitting the multinomial or product multinomial (as the case may be) model. And this is necessary to obtain identifiability.

7.6 Sampling Schemes and Rao Tests

REVISED DOWN TO HERE

this section is actually wrong

For the Rao test, we do need to be clear about the big and little models with model matrices M_2 and M_1, respectively. (We didn't need to do this for the Wilks test because both tests use the same maximized log likelihood.)

The Rao test statistic is given by equations (11) or (12) in the likelihood handout, which are the same for exponential families because observed and expected Fisher information for the canonical parameter are the same. We repeat equation (11) in the likelihood handout here

$$R_n = \bigl(\nabla l_n(\tilde{\theta}_n)\bigr)^T I_n(\tilde{\theta}_n)^{-1} \nabla l_n(\tilde{\theta}_n).$$

Here l_n is the log likelihood for the big model, $I_n(\theta)$ is Fisher information for the canonical parameter θ for sample size n, which is n times Fisher information for sample size one, but the MLE $\tilde{\theta}_n$ is the MLE for the little model.

For our purposes here we drop the n's obtaining If sample size is "large enough"

$$R = \bigl(\nabla l(\tilde{\theta})\bigr)^T I(\tilde{\theta})^{-1} \nabla l(\tilde{\theta}). \tag{57}$$

we have good asymptotic approximation, if not, then we don't. But either way, this is our formula for the Rao test statistic. In (57) we still have l is the log likelihood for the big model, $I(\theta)$ is the log likelihood for the big model, and $\tilde{\theta}$ is the MLE for the little model.

We are going to make a few simplifying assumptions about the model matrices. These are made "without loss of generality" as the saying goes. We could always make the model matrices have this form (as will be explained as we go along). The R formula mini-language may not construct model matrices of this form, but there exist model matrices for the same models that do have this form. We assume that both models are identifiable. What this means for M_1 and M_2 depends on what the saturated model is.

Let V_i denote the column space of M_i. Each V_i is a vector subspace of $\mathbb{R}^J$. Then the nested model condition is $V_1 \subset V_2$.

If the saturated model is Poisson, hence identifiable, then this just says that the ranks of M_1 and M_2 are equal to their column dimensions. Alternatively, it says that the columns of M_1 are a linearly independent set of vectors, a basis for V_1, and similarly for the columns of M_2.

If the saturated model is multinomial, hence not identifiable, having a one-dimensional constancy space, whose one basis vector can be chosen to be δ_I (defined above, p. 50, to be the vector having all components equal to one), then this says that the columns of M_1 are a linearly independent set of vectors, a basis for V_1 and that $\delta_I \notin V_1$, and similarly for the columns of M_2.

The reason for the condition $\delta_I \notin V_i$ is that if this were false, there would exist a β such that $\delta_I = M_i \beta$. But then this β would be a direction of constancy for the i-th model, and that would mean the model would not be identifiable, which we don't want.

If the saturated model is product multinomial for partition $\mathcal{A}$, hence not identifiable, having a constancy space whose dimension is the number of elements of $\mathcal{A}$, a basis for which can be taken to be the vectors δ_A, $A \in \mathcal{A}$ (defined above, p. 52, a δ_A is the indicator function of the set A), then this says that the columns of M_1 are a linearly independent set of vectors, a basis for V_1 and that $\delta_A \notin V_1$ for all $A \in \mathcal{A}$, and similarly for the columns of M_2.

The reason for the condition $\delta_A \notin V_i$ for all $A \in \mathcal{A}$ is similar to the reason for the similar condition for the multinomial scheme.

We are also going to insist that M_2 have the form

$$M_2 = \begin{pmatrix} M_1 & N_2 \end{pmatrix}$$

(a partitioned matrix), so all of the columns of M_1 are also columns of M_2, and the columns of M_1 appear in the same order in M_2 and before any other columns of M_2. The matrix N_2 is the left over columns of M_2, those that are not columns of M_1. The reason why there is "no loss of generality" here is that $V_1 \subset V_2$, and it is a theorem of linear algebra that any basis for V_1 can be extended to be a basis for V_2. Whatever M_2 we started with we can throw away and define a new M_2 as follows. Take the columns of M_1 to be a basis for V_1. Then extend this basis to be a basis for V_2, letting N_2 be the matrix whose columns are the vectors added to the basis for V_1 to get a basis for V_2. The old and new M_2 determine the same model because both have the same column space V_2.

Now with all of that setup we are finally ready to get back to the Rao test. By (35) and (37) we have

$$\begin{aligned} \nabla l_2(\theta) &= M_2^T (y - \mu) \\ I_2(\theta) &= M_2^T I(\theta) M_2 \end{aligned}$$

where quantities with subscripts 2 refer to the big model (still a submodel of the saturated model, and quantities without subscripts refer to the saturated model). Now into $l_2(\theta)$ we want to plug in the MLE for the little model, which is $\tilde{\mu}$ characterized by the "observed equals expected" equation for the little model $M_1^T \tilde{\mu} = M_1^T y$.

So do that plug in

$$\begin{aligned}
\nabla l_2(\tilde{\theta}) &= M_2^T (y - \tilde{\mu}) \\
&= \begin{pmatrix} M_1^T \\ N_2^T \end{pmatrix} (y - \tilde{\mu}) \\
&= \begin{pmatrix} M_1^T (y - \tilde{\mu}) \\ N_2^T (y - \tilde{\mu}) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ N_2^T (y - \tilde{\mu}) \end{pmatrix}
\end{aligned}$$

7.6.1 Example Continued: Rao Tests

Recall that in Section 7.4.1 we compared via Wilks's test the R objects `gout.poisson.no.victim` (little model) and `gout.poisson` (big model) produced by Poisson regression. We now know that the MLE's don't depend on the sampling scheme. The ones we want are for the little model

```r
theta.twiddle <- predict(gout.poisson.no.victim)
mu.twiddle <- predict(gout.poisson.no.victim, type = "response")
```

Fisher information for the saturated model evaluated at the MLE for the little model is

```r
fish.sat <- diag(mu.twiddle)
```

The corresponding Fisher information for the big model is

```r
modmat <- gout.poisson$x
fish.big <- t(modmat) %*% fish.sat %*% modmat
fish.big.inverse <- solve(fish.big)
```

The score (log likelihood gradient) for the big model evaluated at the MLE in the little model

```r
y.poisson <- pdata$verdicts
score.big <- t(modmat) %*% (y.poisson - mu.twiddle)
score.big <- zapsmall(score.big)
score.big
```

```
##                                        [,1]
## (Intercept)                         0.00000
## victimwhite                         0.00000
## defendantwhite                      0.00000
## victimwhite:defendantwhite          0.00000
## victimblack:deathpenaltyyes        -8.98606
## victimwhite:deathpenaltyyes         8.98606
## defendantwhite:deathpenaltyyes      0.00000
```

So the Rao test statistic is

```
rao.poisson <- t(score.big) %*% fish.big.inverse %*% score.big
rao.poisson <- as.vector(rao.poisson)
rao.poisson

## [1] 19.63791
```

Now we want to calculate the same thing assuming product binomial (logistic regression) sampling.

MLE's

```
theta.twiddle <- predict(gout.no.victim)
p.twiddle <- predict(gout.no.victim, type = "response")
mu.twiddle <- n * p.twiddle
```

Fisher information for the saturated model evaluated at the MLE for the little model

```
fish.sat <- diag(n * p.twiddle * (1 - p.twiddle))
```

Corresponding Fisher information for the big model

```
modmat <- gout$x
fish.big <- t(modmat) %*% fish.sat %*% modmat
fish.big.inverse <- solve(fish.big)
```

Score for the big model evaluated at the MLE in the little model

```
score.big <- t(modmat) %*% (y - mu.twiddle)
score.big <- zapsmall(score.big)
score.big

##                   [,1]
## (Intercept)    0.00000
## victimwhite    8.98606
## defendantwhite 0.00000
```

Rao test statistic

```
rao.binomial <- t(score.big) %*% fish.big.inverse %*% score.big
rao.binomial <- as.vector(rao.binomial)
rao.binomial

## [1] 19.63791

all.equal(rao.binomial, rao.poisson)

## [1] TRUE
```

REVISED DOWN TO HERE

8 Multivariate Monotonicity

The important mapping (32) between canonical and mean value parameters of a regular full exponential family has an important property called *multivariate monotonicity*. If θ_1 and θ_2 are possible values of the canonical parameter vector and

$$\mu_1 = \nabla c(\theta_1)$$
$$\mu_2 = \nabla c(\theta_2)$$

are the corresponding values of the mean-value parameter vector, then

$$\langle \mu_1 - \mu_2, \theta_1 - \theta_2 \rangle \geq 0, \tag{58}$$

and if the canonical parameterization is identifiable

$$\langle \mu_1 - \mu_2, \theta_1 - \theta_2 \rangle > 0, \qquad \theta_1 \neq \theta_2. \tag{59}$$

Property (58) holding for all θ_1 and θ_2 in the full canonical parameter space is called *multivariate monotonicity* of the mapping from θ to μ (Rockafellar and Wets, 1998, Chapter 12). It follows from the derivative matrix of this mapping being a variance matrix, hence a positive semi-definite matrix, our equation (6) and Proposition 12.3 in Rockafellar and Wets (1998).

Property (59) holding for all θ_1 and θ_2 in the full canonical parameter space with $\theta_1 \neq \theta_2$ is called *strict multivariate monotonicity* of the mapping from θ to μ (Rockafellar and Wets, 1998, Chapter 12). When the canonical parameterization is identifiable, it follows from our Theorem 1 that there is no nonzero vector δ such that $\langle y, \delta \rangle$ is almost surely constant, which implies both sides of (6) are positive definite matrices. And this implies strict multivariate monotonicity by Proposition 12.3 in Rockafellar and Wets (1998).

Since canonical affine submodels of regular full exponential families are themselves regular full exponential families, the same theory discussed above applies to them. If β_1 and β_2 are possible values of the submodel canonical parameter vector and

$$\begin{aligned}\tau_1 &= \nabla c_{\text{sub}}(\beta_1) = M^T \nabla c(a + M\beta_1) \\ \tau_2 &= \nabla c_{\text{sub}}(\beta_2) = M^T \nabla c(a + M\beta_1)\end{aligned}$$

are the corresponding values of the submodel mean-value parameter vector, then

$$\langle \tau_1 - \tau_2, \beta_1 - \beta_2 \rangle \geq 0, \tag{60}$$

and if the canonical parameterization is identifiable

$$\langle \tau_1 - \tau_2, \beta_1 - \beta_2 \rangle > 0, \qquad \beta_1 \neq \beta_2. \tag{61}$$

So much for the theory of multivariate monotonicity. What is the point? Why is it even called that?

In the case where there is just one parameter (so θ and μ are scalars rather than vectors) equation (58) becomes

$$(\mu_1 - \mu_2)(\theta_1 - \theta_2) \geq 0,$$

so $\mu_1 - \mu_2$ and $\theta_1 - \theta_2$ are either both nonnegative or both nonpositive. Hence $\theta_1 \leq \theta_2$ implies $\mu_1 \leq \mu_2$, and the mapping from θ to μ is a nondecreasing function.

The term monotone applied to scalar-to-scalar functions usually means either nondecreasing or nonincreasing. If a scalar-to-scalar function is multivariate monotone, then it is nondecreasing. So the match is not perfect, but we can see why the name.

In the case where there is just one parameter equation (59) becomes

$$(\mu_1 - \mu_2)(\theta_1 - \theta_2) > 0,$$

so $\mu_1 - \mu_2$ and $\theta_1 - \theta_2$ are either both negative or both positive. Hence $\theta_1 < \theta_2$ implies $\mu_1 < \mu_2$, and the mapping from θ to μ is an increasing function.

The term strictly monotone applied to scalar-to-scalar functions usually means either decreasing or increasing. If a scalar-to-scalar function is multivariate monotone, then it is increasing. So again the match is not perfect, but we can see why the name.

So why is this the multivariate analog of univariate monotonicity? Consider the mapping restricted to a line. Pick two distinct points θ_1 and θ_2 in the canonical parameter space and consider the line determined by these two points

$$r\theta_1 + (1 - r)\theta_2, \qquad r \in \mathbb{R}.$$

Then we consider the part of this line that is in the full canonical parameter space Θ given by (7). Define

$$\begin{aligned} r_{\text{min}} &= \inf\{\, r \in \mathbb{R} : r\theta_1 + (1 - r)\theta_2 \in \Theta \,\} \\ r_{\text{max}} &= \sup\{\, r \in \mathbb{R} : r\theta_1 + (1 - r)\theta_2 \in \Theta \,\} \end{aligned}$$

($r_{\text{min}} = -\infty$ and $r_{\text{max}} = +\infty$ are possible values). Then

$$r\theta_1 + (1 - r)\theta_2, \qquad r_{\text{min}} < r < r_{\text{max}}$$

is the part of this line in the full canonical parameter space.

The map from canonical to mean-value parameters is nonlinear, so the corresponding mean values

$$\mu_r = \nabla c\bigl(r\theta_1 + (1 - r)\theta_2\bigr)$$

do not lie on a line but rather on a smooth curve. What does the multivariate monotonicity property say here? Consider r_1 and r_2 with $r_{\text{min}} < r_1 < r_2 < r_{\text{max}}$. Then

$$\bigl[r_1\theta_1 + (1 - r_1)\theta_2\bigr] - \bigl[r_2\theta_1 + (1 - r_2)\theta_2\bigr] = (r_1 - r_2)(\theta_1 - \theta_2)$$

and the multivariate monotonicity property becomes

$$0 \le \langle \mu_{r_1} - \mu_{r_2}, (r_1 - r_2)(\theta_1 - \theta_2) \rangle = (r_1 - r_2)\langle \mu_{r_1} - \mu_{r_2}, \theta_1 - \theta_2 \rangle$$

so

$$\langle \mu_{r_1} - \mu_{r_2}, \theta_1 - \theta_2 \rangle \le 0$$

and

$$\langle \mu_{r_1}, \theta_1 - \theta_2 \rangle \le \langle \mu_{r_2}, \theta_1 - \theta_2 \rangle$$

Thus the multivariate monotonicity property says

$$r \mapsto \langle \mu_r, \theta_1 - \theta_2 \rangle \tag{62}$$

is a nondecreasing function. And following the same logic for the strictly multivariate monotonicity property would show that (62) is an increasing function.

If we apply what we just learned to points that change only one coordinate of θ, so $\theta_1 - \theta_2$ points along a coordinate axis we get the following, which your humble calls the "dumbed down" version of multivariate monotonicity. (As always θ is the canonical parameter vector and μ is the mean-value parameter vector and θ_i and μ_i are their components.)

- Increase θ_i holding the other components of θ fixed, and μ_i either either increases or stays the same. Other components of μ can go any which way.

- If the canonical parameterization is identifiable, increasing θ_i holding the other components of θ fixed increases μ_i.

The reason we call this "dumbed down" is that it is not enough information to capture the full multivariate monotonicity phenomenon. For example Shaw and Geyer (2010, Appendix A) found that an important bit of scientific reasoning about aster models (which are exponential family models) required the full force of (61). Thinking one coordinate at a time didn't do the job.

Nevertheless, the "dumbed down" version is a lot easier to explain to people and so is often used when the full force of the technical definition is not needed.

In GLM, independence of components of the response vector means that the model is product exponential family, each component having an exponential family distribution that does not involve other components. This means that the mapping $\theta_i \mapsto \mu_i$ is scalar-to-scalar, hence an increasing function (we never use a nonidentifiable canonical parameterization for GLM saturated models, because that would make the canonical parameter space one-dimensional and the mean-value parameter space zero-dimensional, which would give us a degenerate model with only one distribution, and that would be pointless).

So for GLM saturated models, we can reason componentwise. Each μ_i is a function of θ_i only, and this function is increasing. But this does not work for GLM canonical affine submodels. Changing one component of β (the "intercept" component, for example) can change all the components of the submodel canonical parameter vector τ (and also all the components of θ and μ).

For other exponential family models, including multinomial models, we need multivariate monotonicity. There is no available univariate monotonicity.

9 Maximum Entropy

10 Summary

A Proofs

Proof of Theorem 2. What is to be shown is that, if

$$\begin{aligned} \mu_1 &= \nabla c(\theta_1) \\ \mu_2 &= \nabla c(\theta_2) \end{aligned}$$

and $\mu_1 = \mu_2$, then $\theta_1 - \theta_2$ is a direction of constancy, so θ_1 and θ_2 correspond to the same distribution and so μ_1 and μ_2 correspond to the same distribution.

Let $\delta = \theta_1 - \theta_2$, and let l be the log likelihood. Then the change of parameter

$$\theta = \theta_2 + s\delta$$

is a one dimensional canonical affine submodel that has canonical statistic $\langle y, \delta \rangle$, canonical parameter s, and log likelihood

$$l_{\text{sub}}(s) = \langle y, \delta \rangle s - c(\theta_2 + s\delta)$$

which has derivative

$$l'_{\text{sub}}(s) = \langle y, \delta \rangle - \langle \nabla c(\theta_2 + s\delta), \delta \rangle \tag{63}$$

and second derivative

$$l''_{\text{sub}}(s) = -\langle \nabla^2 c(\theta_2 + s\delta)\delta, \delta \rangle \tag{64}$$

Using (6) we can rewrite (64) as

$$\begin{aligned}
-\langle \nabla^2 c(\theta_2 + s\delta)\delta, \delta \rangle &= -\delta^T \left[\nabla^2 c(\theta_2 + s\delta) \right] \delta \\
&= -\delta^T \operatorname{var}_{\theta_2 + s\delta}(y) \delta \\
&= -\operatorname{var}_{\theta_2 + s\delta}(\langle y, \delta \rangle)
\end{aligned}$$

Now there are two cases.

TCase I: $\operatorname{var}_{\theta_2+s\delta}(\langle y, \delta \rangle) = 0$ for some s, in which case $\langle y, \delta \rangle$ is almost surely constant and δ is a direction of constancy by Theorem 1 in Geyer (2009) But that would imply that θ_1 and θ_2 index the same distribution so $\mu_1 = \mu_2$.

Case II: $\operatorname{var}_{\theta_2+s\delta}(\langle y, \delta \rangle) > 0$ for all s such that $\theta_2 + s\delta$ is in the full canonical parameter space (7), in which case, since $\operatorname{var}_{\theta_2+s\delta}(\langle y, \delta \rangle) > 0$ is also $-l''_{\text{sub}}(s)$ by (64), it follows that $-l'_{\text{sub}}(s)$ is increasing in s for all s and $l'_{\text{sub}}(s)$ is decreasing in s for all s. From (63) it follows that $l'_{\text{sub}}(s)$ is decreasing in s if and only if $-\langle \nabla c(\theta_2 + s\delta), \delta \rangle$ decreasing in s, which happens if and only if $\langle \nabla c(\theta_2 + s\delta), \delta \rangle$ is increasing in s. But by (5) we have

$$\langle \nabla c(\theta_2 + s\delta), \delta \rangle = \langle E_{\theta_2 + s\delta}(y), \delta \rangle$$

and plugging in zero and one into this expression gives

$$\langle \mu_2, \delta \rangle < \langle \mu_1, \delta \rangle$$

and this implies $\mu_1 \neq \mu_2$.

In conclusion, $\mu_1 = \mu_2$ implies we are in Case I so $\delta = \theta_1 - \theta_2$ is a direction of constancy. $\square$

References

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

Anderson, T. W. (2003). *An Introduction to Multivariate Statistical Analysis*, third edition. John Wiley & Sons, Hoboken.

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

Cook, R. D. (1998). *Regression Graphics: Ideas for Studying Regressions through Graphics.* John Wiley & Sons, New York.

Cook, R. D., and Adragni, K. P. (2009). Sufficient dimension reduction and prediction in regression. *Philosophical Transactions of the Royal Society, Series A*, **367**, 4385–4405.

Cook, R. D., and Weisberg, S. (1991). Sliced inverse regression for dimension reduction: Comment. *Journal of the American Statistical Association*, **86**, 328–332. Discussion of Li (1991).

Darmois, G. (1935). Sur les lois de probabilités a estimation exhaustive. *Comptes Rendus de l'Académie des Sciences*, **200**, 1265–1266.

Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London, Series A*, 222, 309–368.

Galton, F. (1886). Regression towards mediocrity in hereditary stature. *Journal of the Anthropological Institute*, **15**, 246–263.

Galton, F., and Dickson, J. D. H. (1886). Family likeness in stature. *Proceedings of the Royal Society of London*, **40**, 42–73. Dickson supplied an appendix to Galton's paper.

Geyer, C. J. (1999). Likelihood inference for spatial point processes. In *Stochastic Geometry: Likelihood and Computation*, eds. W. Kendall, O. Barndorff-Nielsen and M. N. M. van Lieshout. Chapman & Hall/CRC, London.

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

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

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

Halmos, P. R., and Savage, L. J. (1949). Application of the Radon-Nikodym theorem to the theory of sufficient statistics. *Annals of Mathematical Statistics*, **20**, 225–241.

Koopman, B. O. (1936). On distributions admitting a sufficient statistic. *Transactions of the American Mathematical Society*, **39**, 399–409.

Li, K-C. (1991). Sliced inverse regression for dimension reduction (with discussion). *Journal of the American Statistical Association*, 86, 316–342.

Neyman, J. (1935). Sur un teorema concernente le cosiddette statistiche sufficienti. *Giornale dell'Istituto Italiano degli Attuari*, **6**, 320–334. The title translates to "On a theorem concerning so-called sufficient statistics."

Pitman, E. J. G. (1936). Sufficient statistics and intrinsic accuracy. *Proceedings of the Cambridge Philosophical Society*, **32**, 567–579.

Ripley, B., Venables, B., Bates, D. M., Hornik, K., Gebhardt, A., and Firth, D. (2015). R package `MASS` (support functions and datasets for Venables and Ripley's MASS), version 7.3-45. This is an R recommended package, installed by default in every installation of R.

Rockafellar, R. T., and Wets, R. J.-B. (1998). *Variational Analysis*. Springer-Verlag, Berlin. (The corrected printings contain extensive changes. We used the third corrected printing, 2010.)

Shaw, R. G., and Geyer, C. J. (2010). Inferring fitness landscapes. *Evolution*, **64**, 2510–2520.

Venables, W. N., and Ripley, B. D. (2002). *Modern Applied Statistics with S*, fourth edition. Springer, New York.