Information Theoretic Properties of the Poisson Binomial

Evan Rosenman

March 20, 2019

Contents

1 Introduction

We consider a number of information-theoretic results relating to the Poisson binomial distribution. The Poisson binomial is a generalization of the binomial distribution in which each trial need not have the same probability of success. If we suppose X is a Poisson binomial random variable with associated success probabilities $p_1, \ldots, p_n$, which we denote $X \sim \text{PoiBin}(p_1, \ldots, p_n)$, then this means X is distributed as

$$X = \sum_{i=1}^{n} Y_i$$

where $Y_i \overset{ind}{\sim} Bern(p_i)$ i.e. the Y_i are independent Bernoulli variables, each with associated success probability p_i.

The Poisson binomial distribution does not have a simple formula for its entropy, and its PDF can be challenging to work with because it involves a combinatorial sum. My project involves two separate components. In section 2, I review some of the key information-theoretic results about this distribution from the literature. In section 3, I consider maximum entropy modeling of Poisson binomial variables. I first explore the functional form of the entropy-maximizing solution, and then run some simulations to explore the performance of maximum entropy vs. maximum likelihood approaches.

2 Literature Review

2.1 Shepp-Olkin Conjecture

We begin with a discussion of the Shepp-Olkin Conjecture, first posited in Shepp and Olkin (1981). The authors begin by showing that the entropy of a Poisson binomial random variable X is a Schur-concave function of the success probability $(p_1, \ldots, p_n)$. A permutation-symmetric differentiable function $\psi(x)$ is Schur-concave (Marshall et al., 1974) if it satisfies

$$\left(\frac{\partial \psi(x)}{\partial x_i} - \frac{\partial \psi(x)}{\partial x_j}\right)(x_i - x_j) \leq 0$$

for all $i \neq j$.

The authors use the notation $\pi_k^n = P(Y_1 + \cdots + Y_n = k)$, with Y_i independent $Bern(p_i)$ random variables, to denote Poisson binomial probabilities. Then, some algebra yields the useful recursion

$$\pi_k^n = p_1 p_2 \pi_k^{n-2} + (p_1 q_2 + q_1 p_2)\, \pi_{k-1}^{n-2} + q_1 q_2 \pi_{k-2}^{n-2} \tag{1}$$

where $\pi_k^{n-2} = P(Y_3 + \cdots + Y_n = k)$. Taking derivatives of this expression gives us:

$$\frac{\partial \pi_k^n}{\partial p_1} - \frac{\partial \pi_k^n}{\partial p_2} = -(p_1 - p_2)\left(\pi_k^{n-2} - 2\pi_{k-1}^{n-2} + \pi_{k-2}^{n-2}\right) \tag{2}$$

Now, recall that the entropy is given by $h = -\sum_k \pi_k^n \log(\pi_k^n)$. Applying the Chain Rule and Equation 2 and reorganizing some sums, we get the expression

$$(p_1 - p_2)\left(\frac{\partial h}{\partial p_1} - \frac{\partial h}{\partial p_2}\right) = (p_1 - p_2)^2 \sum_k \pi_k^{n-2} \log\left(\frac{\pi_k^n \pi_{k-2}^n}{(\pi_{k-1}^n)^2}\right) \tag{3}$$

The proof is completed by observing that

$$\pi_k^n \pi_{k-2}^n \leq \left(\pi_{k-1}^n\right)^2 \tag{4}$$

via an extension of a proof by Hardy and Littlewood (Hardy and Littlewood). This makes the right-hand side of Equation 3 negative and thus yields Schur concavity.

Schur concavity is a weaker condition than concavity in $p_1, \ldots, p_n$. Shepp and Olkin conjectured that a Poisson binomial X is indeed concave in the vector of its success probabilities. But they were only able to prove that X is concave in each $p_i, i = 1, \ldots, n$ – not concavity in the whole vector $(p_1, \ldots, p_n)$.

The conjecture went unproven for 34 years. Yu and Johnson (2009) proved a special case when considering a "thinning" operation proposed by Renyi. Hillion et al. (2012) showed the conjecture holds when the success probabilities are either constant or equal to a parameter $t \in [0, 1]$. A final proof came in 2015 in Hillion et al. (2015). The proof is quite technical and lengthy, so we provide an abridged overview here.

The proof makes use of a simplification whereby we suppose each $p_i = p_i(t)$ is an affine function of some parameter $t \in [0, 1]$, such that $p_i' = \frac{d}{dt} p_i(t)$ is a constant function of t. It follows from composition rules that concavity of the entropy in t is sufficent to prove the Shepp-Olkin conjecture.

The authors use slightly different notation from Shepp and Olkin. They denote as f_k the probability that the Poisson binomial random variable $X = \sum_{j=1}^n Y_j$ is equal to k. They use $f_k^{(i)}$ and $f_k^{(i,j)}$ to denote the probability that the "leave-one-out" sum $\sum_{j \neq i} Y_j$ and "leave-two-out" sum $\sum_{\ell \notin \{i,j\}} Y_\ell$ are equal to k respectively. The analogous quantities in the prior paper were denoted by π_k^n, π_k^{n-1}, and π_k^{n-2} in the prior paper. The authors again appeal to the fact that Poisson binomial probabilities can be defined recursively (as seen in Equation 1); and, taking derivatives, obtain the relations:

$$\frac{df_k}{dt}(t) = g_{k-1} - g_k \tag{5}$$

$$\frac{d^2 f_k}{dt^2}(t) = h_k - 2h_{k-1} + h_{k-2} \tag{6}$$

where

$$g_k = \sum_i p_i' f_k^{(i)}$$

$$h_k = \sum_{i \neq j} p_i' p_j' f_k^{(i,j)} \, .$$

Write $H(t)$ to denote the entropy of the variable X as a function of t. Define $U(v) = v \cdot \log(v)$. Then:

$$\begin{aligned} H''(t) &= -\sum_{k=0}^n \frac{d^2}{dt^2} U\left(f_k(t)\right) \\ &= -\sum_{k=0}^n U''(f_k) \left(\frac{df_k}{dt}\right)^2 - \sum_{k=0}^n U'(f_k) \left(\frac{d^2 f_k}{dt^2}\right) \\ &= -\sum_{k=0}^n U''(f_k)(g_{k-1} - g_k)^2 - \sum_{k=0}^n U'(f_k) \left(h_k - 2h_{k-1} + h_{k-2}\right) \end{aligned}$$

Our goal is to show that this quantity is negative. Plugging in the fact that $U'(v) = 1 + \log(v)$ and $U''(v) = 1/v$, we obtain the expression

$$u_k = h_k \log\left(\frac{f_k f_{k+2}}{f_{k+1}^2}\right) + \left(\frac{g_k^2}{f_k} - 2\frac{g_k g_{k+1}}{f_{k+1}} + \frac{g_{k+1}^2}{f_{k+2}}\right)$$

and observe that if $u_k \geq 0$ for all values of k, then the second derivative of the entropy is indeed negative. The second term in u_k can be shown to be positive via the arithmetic-geometric mean inequality, while the logarithmic term is negative by the same argument as in Equation 4. Thus, $h_k \leq 0 \implies u_k \geq 0$ and we need only consider the case that $h_k > 0$.

The authors establish positivity of u_k in the case of $h_k > 0$ via an algebraically involved – but not particularly interesting – argument. In short, a useful inequality can be shown for a real-valued function ϕ and a set of constants satisfying certain properties. The expression for u_k can be massaged into this form through a series of direct calculations. Lastly, a bound on the value of h_k is derived, which implies the necessary constraints for the original inequality to hold. This completes the proof.

2.2 Poisson Binomial Distributions as Maximum Entropy Distributions

A related set of results was discovered by Harremoës (2001). He demonstrates that the entropy of a Poisson binomial distribution is less than or equal to the entropy of a binomial distribution with the same number of trials and same mean; and also less than or equal to the entropy of a Poisson distribution with the same mean.

Harremoës' proof proceeds in several stages. For discrete distributions defined on the positive integers, he defines the information divergence

$$D(P||Q) = \sum_{i=0}^{\infty} p_i \log\left(\frac{p_i}{q_i}\right)$$

and then defines an operator

$$D(Y) = \min_{\lambda} D(Y||\Pi(\lambda))$$

where $\Pi(\lambda)$ represents a Poisson distribution with mean λ. Some simple algebra shows that for Y a Bernoulli random variable with success probability p_i

$$D(Y_i) = (1 - p_i)(-p_i) + p_i$$

It is also straightforward to show that the $D(\cdot)$ operator is subadditive for independent random variables, from which it follows that

$$D(X) \leq \sum_{i=1}^{n} D(Y_i) \leq \sum_{i=1}^{n} p_i^2$$

for $X = \sum_{i=1}^{n} Y_i$, $Y_i \sim Bern(p_i)$ independently (i.e. X is a Poisson binomial distribution with success probabilities p_i). This line of argumentation provides a proof to Poisson's Law, which says that if $p_{max} = \max_i(p_1, \ldots, p_n) \to 0$, then $D(X)$ also converges to 0. In words, this means that a Poisson binomial random variable can be approximated very well by a Poisson distribution if all the success probabilities are small.

As Harremoës' paper preceded the proof of the Shepp-Olkin Conjecture, he relies on a weakened version of the conjecture. But his proofs simplify if we use the concavity of $H(X)$, the entropy of X, in $(p_1, \ldots, p_n)$. We suppose X has mean λ and maximum probability p_{max}. We consider the binomial distribution with number of trials $m = \lfloor \frac{\lambda}{p_{max}} \rfloor$ and success probability λ/m. He claims

$$H(X) \geq H(\text{Binomial}(m, \lambda/m))$$

where the RHS is the entropy of the aforementioned binomial distribution. This can be shown by an appeal to concavity. Consider R, the minimum entropy Poisson binomial distribution with mean λ for whom the largest success probability is at most λ/m. Observe that if R had two distinct success probabilities p_i, p_j such that $0 < p_i < p_j < \lambda/m$, then concavity would imply that substituting $p_i - \epsilon, p_j + \epsilon$ would yield a lower entropy, which is a contradiction. Hence, there can be at most one p_i such that $p_i \in [0, \lambda/m)$, but this, too, yields a contradiction, since λ must be the average of the success probabilities. It follows that every probability must lie in $\{0, \lambda/m\}$, and hence that R is a binomial distribution with parameters $(m, \lambda/m)$. This is precisely the given result.

This same line of argumentation tells us the main result: that among all Poisson binomial distributions with mean λ and number of trials n, the Binomial$(n, \lambda/n)$ distribution has maximal entropy. The given concavity means that if we have two probabilities p_i, p_j that are equal, then shifting them in a mean-preserving way (i.e. taking $p_i \to p_i + \epsilon, p_j \to p_j - \epsilon$) will necessarily reduce entropy. Hence, a Binomial$(n, \lambda/n)$ is the maximum entropy distribution.

Lastly, Harremoës uses previous results on Poisson's Law to show that the bound can be extended to show that the Poisson distribution with mean λ has entropy equal to that of the maximal entropy among any Poisson Binomial distribution of any size n with mean λ.

3 Maximum Entropy Modeling

3.1 Background

We now pivot to a different application of information-theoretic concepts in this setting: binary classification.

In a typical binary classification problem, unit labels are known at the individual level. A common modeling technique is logistic regression, in which the individual success probabilities are modeled as $p_i = \sigma(\beta^T x_i)$ for $\sigma(v) = 1/(1 + exp(-v))$ the sigmoid function, x_i a vector of individual covariates, and β a parameter vector. The model is trained via maximum likelihood, meaning we explicitly assume each class label is a Bernoulli random variable and select β to maximize the probability of the observed data under this model.

A well-known result states that "the exponential model [for] logistic regression, when trained according to the maximum likelihood criterion, also finds the maximum entropy distribution subject to the constraints from the feature functions" (Martin and Jurafsky, 2009). In other words, logistic regression has the desirable property of *also* finding the set of probabilities that maximize the entropy, subject to reasonable constraints that the predicted covariate averages in each class match the observed covariate averages for that class.

We are interested in extending these results to the setting of ecological inference, in which class labels are not known at the individual level but are known at some higher level of aggregation. The Poisson binomial distribution is the natural distribution to use for modeling in this setting, as we observe only summed outcomes rather than individual-level outcomes. This kind of model is particularly interesting for modeling the outcomes of political elections, since voter characteristics are known at the individual level but vote tallies are only reported at each voting precinct.

In the next section, we set up the problem mathematically. We then consider whether the same equivalence of maximum likelihood and maximum entropy models occurs in our generalized binary classification problem. The section concludes with some numerical examples, demonstrating the empirical benefits of a maximum entropy approach.

3.2 Problem Set-Up

Mount (2011) provided a short proof demonstrating that the sigmoid is the form of the entropy-maximizing function under minimal assumptions in the binary classification problem. We follow a similar path to explore the appropriate functional form in the case of a Poisson binomial likelihood.

We use the elections setting as our motivating example. Let $p \in \{1, 2, \ldots, P\}$ index our *precincts* at which vote tallies are known, and let $i = 1, \ldots, n_p$ index the individual within the precinct. We suppose the election only includes a binary choice between a Democrat and a Republican. Each individual has an associated vector of covariates $x_{pi} \in \mathbb{R}^d$ as well as a pair of (unseen) indicators $D_{pi} \in \{1, 0\}, R_{pi} \in \{1, 0\}$ such that $D_{pi} = 1 - R_{pi}$. For each p, we observe

$$D_p = \sum_{i=1}^{n_p} D_{pi}, \quad R_p = \sum_{i=1}^{n_p} R_{pi}\,,$$

the total number of Democratic and Republican votes in each precinct.

We model $D_{pi} \sim Bern(d_{pi})$ where $d_{pi} \in (0, 1)$ is the individual's probability of voting Democrat. Under such a model, D_p is a Poisson binomial random variable. We suppose there exists a function $\pi_d(\cdot)$ such that

$$\pi_d(x_{pi}) = d_{pi}\,.$$

It is intuitively obvious that we need only model d_{pi} since the Republican voting probability r_{pi} is just $1 - d_{pi}$. But we work in full generality and define

$$\pi_r(x_{pi}) = r_{pi}$$

and enforce the "summing-to-unity" constraint explicitly.

3.3 Maximum Entropy Approach

Following the approach of Mount, we write down properties of desirable functional forms $\pi_d(\cdot), \pi_r(\cdot)$ and then maximize the entropy subject to these constraints. Obvious choices of constraints are

$$0 \leq \pi_d(x), \pi_r(x) \leq 1 \quad \text{and} \quad \pi_d(x) + \pi_r(x) = 1$$

for any choice of x, since these values are meant to represent probabilities.

Choosing a covariate balance constraint is more complex. In the logistic regression setting, in which D_{pi} (and thus R_{pi}) are observed, the constraint is simply

$$\sum_p \sum_i \pi_d(x_{pi}) x_{pi} = \sum_p \sum_i D_{pi} x_{pi} \tag{7}$$

i.e. the model predicted covariate averages for the Democratic and Republican voters should match the observed covariate averages.

In this case, we argue that the analogous conditions are

$$\sum_p \sum_i \pi_d(x_{pi}) x_{pi} = \sum_p \sum_i P\left(D_{pi} = 1 \mid \sum_{j=1}^{n_p} D_{pj} = D_p\right) x_{pi}$$
$$\sum_p \sum_i \pi_r(x_{pi}) x_{pi} = \sum_p \sum_i P\left(R_{pi} = 1 \mid \sum_{j=1}^{n_p} R_{pj} = R_p\right) x_{pi}\,.$$

This appears somewhat strange at first. The left-hand side is the predicted (unconditional) covariate means within each population. The right-hand side is the covariate means conditional on the precinct vote totals. This expression is analogous in the sense that we're taking a model-based average and trying to match it with an observed average; the challenge here is that we can't observe averages directly because the labels aren't known. The probabilistic expression on the right-hand side is conditioning on what little information we do know: the precinct-level vote totals.

A problem remains, of course, in that we don't know $P(D_{pi} = 1), P(R_{pi} = 1)$ and hence cannot actually calculate the conditional probabilities on the right hand side. But if we plug in $\pi_d(x_{pi}), \pi_r(x_{pi})$, this will still be a meaningful condition because we are conditioning on the vote totals. Observe

$$P\left(D_{pi} = 1 \mid \sum_{j=1}^{n_p} D_{pj} = D_p\right) = P(D_{pi} = 1) \frac{P\left(\sum_{j=1, j \neq i}^{n_p} D_{pj} = D_p - 1\right)}{P\left(\sum_{j=1}^{n_p} D_{pj} = D_p\right)} = \pi_d(x_{pi}) \frac{\phi^d_{p,-i}(D_p - 1)}{\phi^d_p(D_p)}$$

where

$$\phi^d_{p,-i}(D_p - 1) = P\left(\sum_{j=1, j\neq i}^{n_p} D_{pi} = D_p - 1\right), \quad \phi^d_p(D_p) = P\left(\sum_{j=1}^{n_p} D_{pi} = D_p - 1\right)$$

with analogous results for Republican votes. So this gives us a more compact expression for the condition:

$$\sum_p \sum_i \pi_d(x_{pi}) x_{pi} = \sum_p \sum_i \pi_d(x_{pi}) \frac{\phi^d_{p,-i}(D_p - 1)}{\phi^d_p(D_p)} x_{pi}\,. \tag{8}$$

Another way to consider this expression is to take its limits. Two limits are noteworthy:

- Take n_p to 1 for all p. In this case, the probability on the right-hand side simply becomes an indicator variable, $I(D_p = 1)$, which is simply D_{pi}! In this case, we will have recovered the logistic regression condition given by (7).
- Take n_p to ∞ for all p. In this case, the impact of any one indicator D_{pi} is negligible and the probability ratio on the right-hand side of (8) goes to 1. Hence, the expression reduces to a tautology.

These limits give the expected behavior and increase our confidence that this is the "right" balance condition.

Now, we can write down a Lagrangian. Following Mount, we exclude the inequality constraints and write

$$L = -\sum_p \sum_i \left(\pi_d(x_{pi})\log(\pi_d(x_{pi})) + \pi_r(x_{pi})\log(\pi_r(x_{pi}))\right) +$$
$$\sum_p \sum_i \lambda_d^T \left(1 - \frac{\phi^d_{p,-i}(D_p - 1)}{\phi^d_p(D_p)}\right) \pi_d\left(x_{pi}\right) x_{pi} + \sum_p \sum_i \lambda_r^T \left(1 - \frac{\phi^r_{p,-i}(R_p - 1)}{\phi^r_p(R_p)}\right) \pi_r\left(x_{pi}\right) x_{pi} +$$
$$\sum_p \sum_i \beta_{pi}\left(\pi_d(x_{pi}) + \pi_r(x_{pi}) - 1\right)$$

Now, we choose some arbitrary $p^\star, i^\star$ and take the derivative in $\pi_d(x_{p^\star i^\star})$. By the KKT conditions, any optimum of the original problem will be a zero of the derivative of the Lagrangian in each parameter. We hope that, as in Mount (2011), the resulting equation will also give us insights into the functional form of the entropy-maximizing probabilities. A challenge arises, though, in that $\phi^d_{p^\star,-i}(\cdot), \phi^d_{p^\star}(\cdot)$ both depend on $\pi_d(x_{p^\star i^\star})$. We will also need to make use of the following recursion:

$$\phi^d_p(D_p) = \pi_d(x_{pi})\phi^d_{p,-i}(D_p - 1) + (1 - \pi_d(x_{pi}))\phi^d_{p,-i}(D_p) \tag{9}$$

We can then write:

$$\frac{\partial L}{\partial \pi_d(x_{p^\star i^\star})} = -1 + \beta_{p^\star i^\star} - \log(\pi_d(x_{p^\star i^\star})) + \left(1 - \frac{\phi^d_{p\star,-i^\star}(D_{p^\star} - 1)\phi^d_{p\star,-i^\star}(D_{p^\star})}{\phi^d_{p^\star}(D_{p^\star})^2}\right)\lambda_d^T x_{p^\star i^\star} +$$
$$\frac{1}{\phi^d_{p^\star}(D_{p^\star})^2}\sum_{i \neq i^\star}\left(\phi^d_{p^\star,-\{i,i^\star\}}(D_{p^\star} - 1)\phi^d_{p^\star,-i^\star}(D_{p^\star} - 1) - \right.$$
$$\left.\phi^d_{p^\star,-\{i,i^\star\}}(D_{p^\star} - 2)\phi^d_{p^\star,-i^\star}(D_{p^\star})\right)\pi_d(x_{p^\star i})\lambda_d^T x_{p^\star i}$$

While this derivative is quite complex, many of its terms are not dependent on $\pi_d(x_{p^\star i^\star})$ (though they do depend on the other probabilities). For the purposes of simplicity, we can make use of Equation 9 and define the following constants

$$\begin{aligned} c_1 &= \phi^d_{p\star,-i^\star}(D_{p^\star} - 1) \\ c_2 &= \phi^d_{p\star,-i^\star}(D_{p^\star}) \\ c_3 &= \sum_{i \neq i^\star}\left(\phi^d_{p^\star,-\{i,i^\star\}}(D_{p^\star} - 1)\phi^d_{p^\star,-i^\star}(D_{p^\star} - 1) - \right. \\ &\qquad \left.\phi^d_{p^\star,-\{i,i^\star\}}(D_{p^\star} - 2)\phi^d_{p^\star,-i^\star}(D_{p^\star})\right)\pi_d(x_{p^\star i})\lambda_d^T x_{p^\star i} \end{aligned}$$

so as to write

$$\begin{aligned} \frac{\partial L}{\partial \pi_d(x_{p^\star i^\star})} &= -1 + \beta_{p^\star i^\star} - \log(\pi_d(x_{p^\star i^\star})) + \left(1 - \frac{c_1 c_2}{\left((c_1 - c_2)\pi_d(x_{p^\star i^\star}) + c_2\right)^2}\right)\lambda_d^T x_{p^\star i^\star} + \\ &\quad \frac{c_3}{\left((c_1 - c_2)\pi_d(x_{p^\star i^\star}) + c_2\right)^2} \\ &= -\log(\pi_d(x_{p^\star i^\star})) + \frac{c_3 - c_1 c_2 \lambda^T x_{p^\star i^\star}}{\left((c_1 - c_2)\pi_d(x_{p^\star i^\star}) + c_2\right)^2} + \left(\lambda_d^T x_{p^\star i^\star} + \beta_{p^\star i^\star} - 1\right) \end{aligned}$$

In this form, we can see that a zero of the Lagrangian will form a transcendental equation that will not have a straightforward analytic form. So the simple equivalency between maximum entropy modeling and logistic regression does not generalize to this case. However, numerical methods can be used to estimate maximum entropy models, as we will see in the next section.

One final situation worth considering is that of $c_2 \approx c_1$. This is actually the standard case in the elections setting, in which precincts typically report a few hundred votes such that the probabilities will not be much affected by the presence of absence of a single voter. A Taylor Expansion yields the approximate condition

$$\pi_d(x_{p^\star i^\star})\left(\frac{2\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{2c_3(c_1-c_2)}{c_2^3}\right) - \log(\pi(x_{p^\star i^\star})) - \frac{\lambda_d^T x_{p^\star i^\star}(c_1-c_2)}{c_2} + \frac{c_3}{c_2^2} - 1 + \beta_{p^\star i^\star} = 0$$

$$\pi_d(x_{p^\star i^\star})\left(\frac{2\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{2c_3(c_1-c_2)}{c_2^3}\right) - \log(\pi(x_{p^\star i^\star})) = \frac{\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{c_3}{c_2^2} + 1 - \beta_{p^\star i^\star}$$

Exponentiating yields

$$\frac{1}{\pi_d(x_{p^\star i^\star})} e^{\pi_d(x_{p^\star i^\star})\left(\frac{2\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{2c_3(c_1-c_2)}{c_2^3}\right)} = e^{\frac{\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{c_3}{c_2^2} + 1 - \beta_{p^\star i^\star}}$$

To further simplify notation, we introduce the symbol θ:

$$\theta = \left(\frac{2\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{2c_3(c_1-c_2)}{c_2^3}\right)$$

and then rewrite this equation as

$$\frac{1}{\theta\pi_d(x_{p^\star i^\star})} e^{\pi_d(x_{p^\star i^\star})\theta} = e^{\frac{\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} - \frac{c_3}{c_2^2} + 1 - \beta_{p^\star i^\star} - \log(\theta)}$$

$$-\theta\pi_d(x_{p^\star i^\star}) e^{-\pi_d(x_{p^\star i^\star})\theta} = -e^{-\frac{\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} + \frac{c_3}{c_2^2} - 1 + \beta_{p^\star i^\star} + \log(\theta)}$$

In this form, we can see that the approximate solutions will take the form of a Lambert W Function (Corless et al., 1996). Denoting the function as $W_0(\cdot)$ and supposing we have an expression

$$z = we^w$$

where w is real and satisfies $w \geq -1$ while $z > -1/e$, then $W_0(z) = w$. In our case, we can see:

$$-\frac{1}{\theta} W_0\left(-e^{-\frac{\lambda^T x_{p^\star i^\star}(c_1-c_2)}{c_2} + \frac{c_3}{c_2^2} - 1 + \beta_{p^\star i^\star} + \log(\theta)}\right) = \pi_d(x_{p^\star i^\star})$$

Because the constants θ, c_1, c_2, c_3 will all depend on the other probabilities, this insight could only really be used for implementation of a coordinate-descent algorithm, wherein each of the $\pi_d(x_{pi})$ are updated sequentially (Wright, 2015). Such a method might be efficient in some circumstances, but will likely not scale well to large problems. An additional challenge in such an algorithm would be enforcement of the requirement that $0 < \pi_d(x_{p^\star i^\star}) < 1$.

Nonetheless, this provides some useful insights about the additional layer of complexity in our problem. In the classification problem in which labels are known at the individual level, the maximum entropy solution can be immediately found to have the form of a sigmoid. But when labels are only known at the aggregate level, the optimal solutions are (approximately) Lambert W functions, which cannot be written in terms of elementary functions!

3.4 Numerical Maximum Entropy Results

Finally, we fit these models to simulated data in order to explore their properties. Code can be found in the appendix. Since our covariate balance constraint is non-linear and relies on Poisson binomial probabilities (which can only be computed with a few specialized libraries), we cannot make use of standard solvers like CVX. Instead, we use the `constrOptim()` function in `R`, designed for constrained optimization tasks.

In our simulations, we suppose we have $n = 32$ individuals. We will vary the number of precincts in each simulation, which makes indexing the individuals tricky. For purposes of clarity, we will use i to denote each individual's index among the 32 total individuals (so $i = 1, \dots, 32$) and will use j to denote each individual's index within his or her precinct (so $j = 1, \dots, n_p$). Now, we assume each individual i has an associated covariate vector $x_i \in \mathbb{R}^d$, where we explore the $d = 2$ and $d = 3$ cases. The first entry in each x_i is an intercept term with the remaining entries sampled as independent Gaussians. β is chosen to be the vector in $\mathbb{R}^d$ whose entries are all equal to 1/2. The true outcomes, which we denote as $D_i^\star$ representing whether individual i voted for a Democrat, are sampled as $Bern(\sigma(\beta^T x_i))$ random variables. Hence, the logistic regression is the true model in this setting.

In each simulation, we vary the total number of precincts P across $1, 2, 4, 8, 16$, and 32. At each value of P, we sum up the associated indicators within each precinct to get our precinct toals. For example, when $P = 1$, we have one precinct total $D_1 = \sum_{i=1}^{32} D_i^\star$. When $P = 2$, we have two precinct totals, $D_1 = \sum_{i=1}^{16} D_i^\star$ and $D_2 = \sum_{i=17}^{32} D_i^\star$. At $P = 32$, the problem reduces to logistic regression.

To solve for an approximate optimum, we define our optimization variables as the Democratic voting probabilities d_{pi} for $p = 1, \dots, P, i = 1, \dots, n_p$ for each choice of P. We then use `constrOptim()` to minimize the objective

$$\sum_{p=1}^{P} \sum_{j=1}^{n_p} d_{pj} \log(d_{pj}) + (1 - d_{pj}) \log(1 - d_{pj}) + \lambda^T \sum_{p=1}^{P} \sum_{j=1}^{n_p} \left(1 - \frac{\phi^d_{p,-j}(D_p - 1)}{\phi^d_p(D_p)} \right) d_{pj} X_i$$

subject to the $0 \le d_{pi} \le 1$ constraint (note that the first term is the *negative* of the entropy). In practice, we explore a grid of large λ values to ensure the penalty term enforces the covariate balance constraint within some small ϵ.

The above procedure gives us a set of 32 entropy-maximizing probabilities for each value of p. We also compute the probabilities through maximum likelihood. Specifically, we model the d_{pi} explicitly as $d_{pi} = \sigma(\beta^T x_i)$ and then choose β to minimize the Poisson binomial likelihood that results from modeling individual i as $Bern(d_{pi})$. Note that our modeling assumptions are correct in this setting – the model relating x_i to d_{pi} really *is* a logistic regression – and we are only trying to recover the true β.

In Figure 1 we plot the results for $x_i \in \mathbb{R}^2$. Each colored line represents one of the 32 fitted probabilities, and we show how they evolve from $P = 1$ to $P = 32$. Both procedures converge to the same fitted probabilities at $P = 32$, those of a logistic regression; these final probabilities are plotted as black dots.

We see an interesting pattern from these simulations: the maximum entropy probabilities evolve somewhat "smoothly" toward the logistic regression solutions as the number of precincts increases, while the maximum likelihood probabilities oscillate wildly between 1 and 16 precincts. Some investigation quickly reveals why: perfect separation. In logistic regression, any hyperplane which perfectly separates the two classes will have a normal vector $\beta^\star$ such that

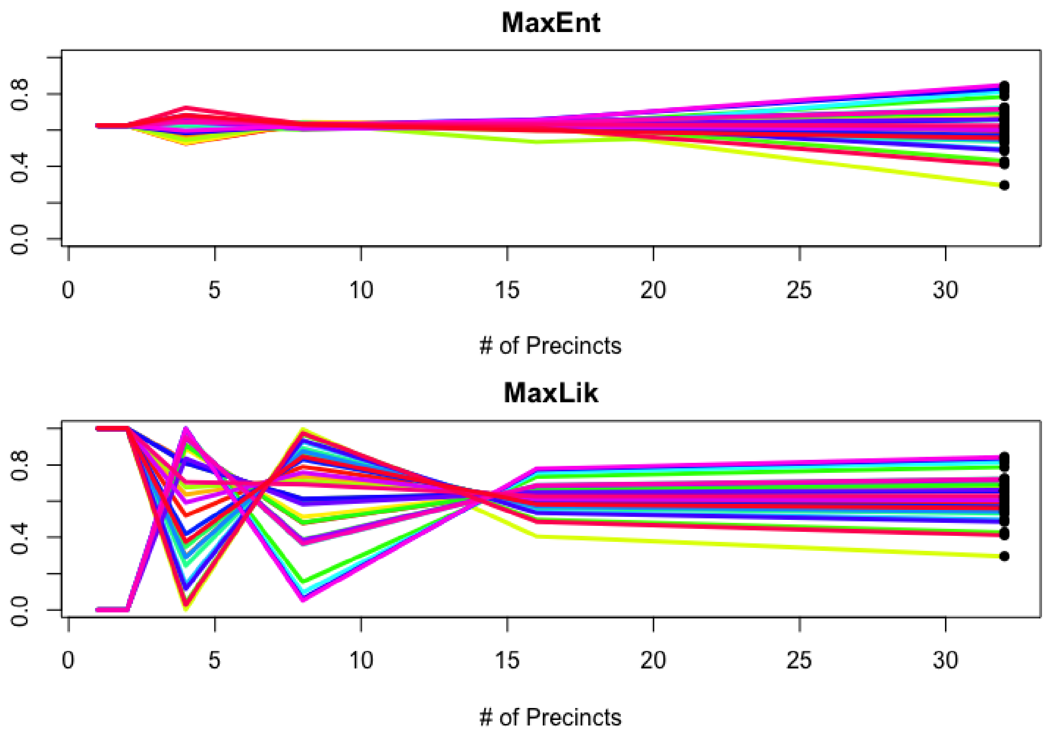

Figure 1: Evolution of fitted probabilities for the as the number of precincts increases, for the $x_i \in \mathbb{R}^2$ case. Plots are given for both maximum entropy and maximum likelihood fitting procedures

a vector of infinite norm in the direction of $\beta^\star$ yields a log-likelihood of 0. Conventional solvers will thus yield fitted probabilities of 0 or 1 in this case. The problem is in some sense "worse" in our ecological inference setting. Any hyperplane which separates any subset of D_p units from the remaining $n_p - D_p$ units for all $p = 1, \ldots, P$ precincts will have normal vector $\beta^\star$ such that an infinite norm vector in the direction of $\beta^\star$ is an MLE. Hence, the fitted probabilities also go to 0 or 1. This is is what seems to be happening when $P \leq 8$. But the maximum entropy solutions are naturally regularized toward 1/2 and hence are much more stable at low values of P.

As we know the true Democratic-voting probabilities in these simulations, we can compute the MSE for our fitted vs. true probabilities at each value of P. The results are plotted in Figure 2 on a log scale. Here, we can see even more explicitly that the maximum entropy fitted probabilities are closer on average for $P = 1, 2, 4$, and 8. Only at $P = 16$ is there sufficient data such that the maximum likelihood fitted probabilities are closer.

As we can see in Figure 3, these trends remain largely the same for the $x \in \mathbb{R}^3$ case, though the maximum likelihood solutions outperform at $P = 8$.

Together, these results demonstrate that there may be some practical utility to maximum entropy modeling in the ecological inference setting. In particular, we see that the solutions are naturally regularized away from 0 and 1, which helps us to fit more accurate models when we have relatively little data available. Only when we have more outcome data does a maximum likelihood procedure outperform.

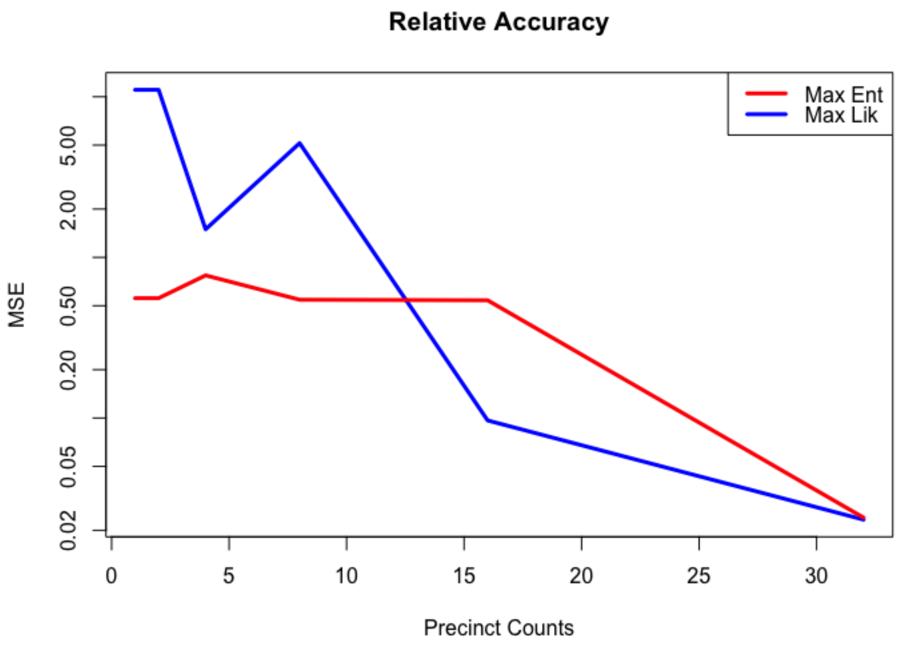

Figure 2: Mean squared error of maximum likelihood vs. maximum entropy fitting procedures for recovering the true Democratic voting probabilities d_{pi} at different values of P, the total number of precincts.

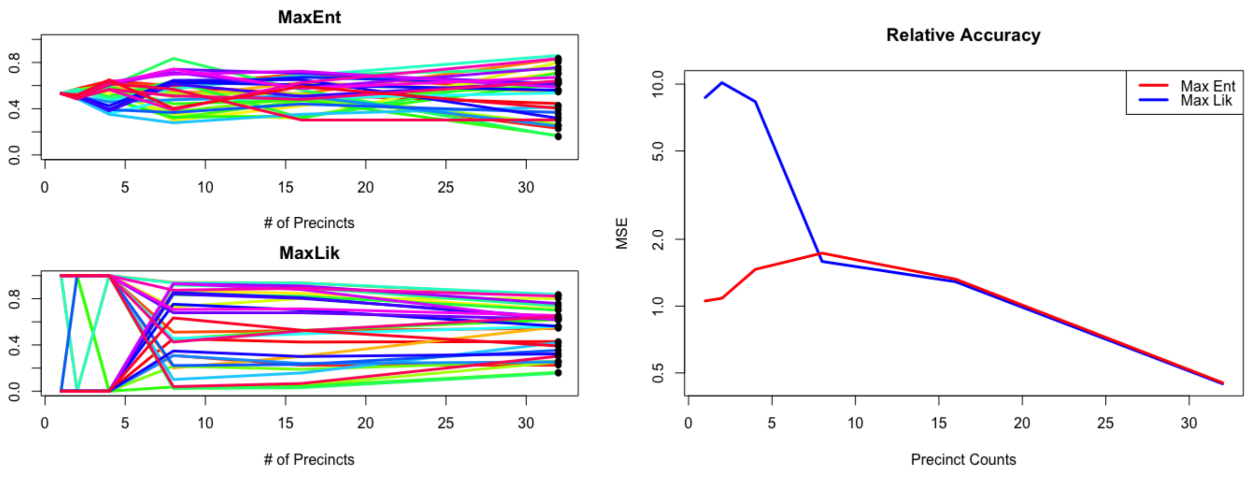

Figure 3: Evolution of fitted probabilities and MSE of the two methods in the $x_i \in \mathbb{R}^3$ case.

References

Corless, R. M., G. H. Gonnet, D. E. Hare, D. J. Jeffrey, and D. E. Knuth (1996). On the lambertw function. *Advances in Computational mathematics 5*(1), 329–359.

Hardy, G. and J. Littlewood. G. polya (1952), inequalities.

Harremoës, P. (2001). Binomial and poisson distributions as maximum entropy distributions. *IEEE Transactions on Information Theory 47*(5), 2039–2041.

Hillion, E. et al. (2012). Concavity of entropy along binomial convolutions. *Electronic communications in probability 17.*

Hillion, E., O. Johnson, et al. (2015). A proof of the shepp–olkin entropy concavity conjecture. *Bernoulli 23*(4B), 3638–3649.

Marshall, A. W., I. Olkin, et al. (1974). Majorization in multivariate distributions. *The Annals of Statistics 2*(6), 1189–1200.

Martin, J. H. and D. Jurafsky (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Pearson/Prentice Hall Upper Saddle River.

Mount, J. (2011). The equivalence of logistic regression and maximum entropy models. *URL: http://www. win-vector. com/dfiles/LogisticRegressionMaxEnt. pdf.*

Shepp, L. A. and I. Olkin (1981). Entropy of the sum of independent bernoulli random variables and of the multinomial distribution. In *Contributions to probability*, pp. 201–206. Elsevier.

Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming 151*(1), 3–34.

Yu, Y. and O. Johnson (2009). Concavity of entropy under thinning. In *2009 IEEE International Symposium on Information Theory*, pp. 144–148. IEEE.

Appendix: R Code

```
rm(list = ls())

# libraries
library(poibin)
library(BBmisc)
library(mvtnorm)

# constants
numUnits <- 32
dimension <- 3

###############################################
##             helper functions              ##
###############################################

# lagrangian function
penalty.onePrecinct <- function(probs, X, d) {
  if(length(probs) > 1) {
    loo.probs <- sapply(1:length(probs), FUN = function(i) {
      dpoibin(kk = d - 1, pp = probs[-i])})
  } else {
    loo.probs <- as.numeric(d == 1)
  }
  probRatios <- loo.probs/dpoibin(kk = d, pp = probs)#dpoisbinom(d, probs)

  if(length(probRatios) == 1) {
    return((1-probRatios)*probs*X)
  } else
    return(colSums((1-probRatios)*probs*X))
}
penalty <- function(probs, data, numPrecincts) {

  # append the proposed probabilities to the data list
  numUnitsPerPrecinct <- length(probs)/numPrecincts
  for(i in 1:numPrecincts) {
    data[[i]]$probs <- probs[((i-1)*numUnitsPerPrecinct +1):(i*numUnitsPerPrecinct)]
  }

  # get the lagrangian value for each precinct
  return(rowSums(sapply(1:numPrecincts, FUN = function(i) {
    p <- data[[i]]
    penalty.onePrecinct(p$probs, p$X, p$d)
  })))
}

lagrangian <- function(probs, lambda, data, numPrecincts) {

  # append the proposed probabilities to the data list
  numUnitsPerPrecinct <- length(probs)/numPrecincts
  for(i in 1:numPrecincts) {
    data[[i]]$probs <- probs[((i-1)*numUnitsPerPrecinct +1):(i*numUnitsPerPrecinct)]
  }

  # get the lagrangian value for each precinct
  penalty <- lambda %*% abs(rowSums(sapply(1:numPrecincts, FUN = function(i) {
    p <- data[[i]]
    penalty.onePrecinct(p$probs, p$X, p$d)
  })))

  sum(probs*log(probs) + (1-probs)*log(1-probs)) + penalty
}

# function to generate precincts
precinctGen <- function(X, W, numPrecincts) {

  # get the precinct indices
  indices <- chunk(1:length(W), n.chunks = numPrecincts)

  # generate the precincts
  data <- lapply(indices, FUN = function(i) {
    list(X = X[i,], d = sum(W[i]))
  })
  return(data)
}

##############################################
##             generate data               ##
##############################################

# suppose the true data *is* a sigmoid function
beta <- rep(0.5, dimension)

# sample the data
X <- cbind(1, rmvnorm(numUnits, rep(0, dimension - 1)))
probs <- 1/(1 + exp(-X %*% beta))
W <- as.numeric(runif(numUnits) < probs)

##################################################
##        solve the maxent probabilities        ##
##################################################

# use a range of lambda values
precinctCounts <- c(1, 2, 4, 8, 16, 32)
lambdaSeq <- 2^seq(from = -1, to = 3, by = 0.5)
penaltyBounds <- 0.01

probValues.maxent <- sapply(precinctCounts, FUN = function(numPrecincts) {

  print(numPrecincts)

  # generate the data
  data <- precinctGen(X, W, numPrecincts)

  # iterate through the lambda options
  results <- lapply(lambdaSeq, FUN = function(lam) {

    lambda <- rep(lam, dimension)

    # solve under the given lambda value
    results <- constrOptim(rep(mean(W), numUnits),
                           f = lagrangian, ui = rbind(diag(numUnits), -diag(numUnits)),
                           ci = c(rep(0, numUnits), rep(-1, numUnits)), grad = NULL,
                           lambda = lambda, data = data, numPrecincts = numPrecincts)

    # store the data
    list(entropy = sum(-results$par*log(results$par)),
         penalty = penalty(results$par, data, numPrecincts),
         convergence = results$convergence,
         probabilities = results$par)
  })

  # choose the maximum entropy choice that satisfies the penalty condition
  satisfied.conditions <- which(sapply(results, FUN = function(x) {
         mean(abs(x$penalty) < penaltyBounds) == 1}))
  results[satisfied.conditions][[which.max(sapply(results[satisfied.conditions],
    FUN = function(x) {x$entropy}))]]$probabilities

})

# ensure approximate equality with logistic results
logistic.predicted.probs <- (1/(1+ exp(-X %*%
       glm.fit(x = X, y = W, family = binomial())$coefficients)))
plot(x = logistic.predicted.probs, y = probValues.maxent[,ncol(probValues.maxent)], pch = 16)
abline(a = 0, b = 1, lwd = 3)

# plot the evolution
par(mfrow = c(2, 1))
par(mar = c(4.1, 2.1, 2.1, 0.1))
cols <- rainbow(numUnits)
plot(y = probValues.maxent[1,], x = precinctCounts, type = "l", lwd = 3, col = cols[1],
     ylim = c(0, 1.0), xlab = "# of Precincts", ylab = "Probability",
     main = "MaxEnt")
sapply(2:numUnits, FUN = function(i) {
  lines(y = probValues.maxent[i,], x = precinctCounts, type = "l", lwd = 3, col = cols[i])
})
points(y = logistic.predicted.probs,
       x = rep(precinctCounts[length(precinctCounts)], numUnits),
       pch = 16)

#####################################################
##         solve the max lik probabilities         ##
#####################################################

# likelihood function
log.likelihood <- function(beta, data) {

  -sum(sapply(data, FUN = function(p) {
    probs <- 1/(1 + exp(-(p$X %*% beta)))
    log(dpoibin(pp = probs, kk = p$d))
  }))
}

probValues.maxlik <- sapply(precinctCounts, FUN = function(numPrecincts) {

  print(numPrecincts)

  # generate the data
  data <- precinctGen(X, W, numPrecincts)
  opt.beta <- optim(rep(0, dimension), log.likelihood, data = data)$par

  round(1/(1 + exp(-X %*% opt.beta)), 5)
})


# plot the evolution
cols <- rainbow(numUnits)
plot(y = probValues.maxlik[1,], x = precinctCounts, type = "l", lwd = 3, col = cols[1],
     ylim = c(0, 1.0), xlab = "# of Precincts", ylab = "Probability",
     main = "MaxLik")
sapply(2:numUnits, FUN = function(i) {
  lines(y = probValues.maxlik[i,], x = precinctCounts, type = "l", lwd = 3, col = cols[i])
})
points(y = logistic.predicted.probs, x = rep(precinctCounts[length(precinctCounts)], numUnits)
       pch = 16)
par(mfrow = c(1, 1))

#####################################################
##               which is closer?                 ##
#####################################################

par(mar = c(4.1, 4.1, 4.1, 2.1))
mse.values <- rbind(apply(probValues.maxent, 2, FUN = function(x) {sum((x-probs)^2)}),
                    apply(probValues.maxlik, 2, FUN = function(x) {sum((x-probs)^2)}))

plot(mse.values[2,], x = precinctCounts, col = "blue", lwd = 3,
     main = "Relative Accuracy", xlab = "Precinct Counts",
     ylab = "MSE", type = "l", log = "y")
lines(mse.values[1,], x = precinctCounts, col = "red", lwd = 3)
legend("topright", col = c("red", "blue"), lwd = 3,
       legend = c("Max Ent", "Max Lik"))
```