The objective is to estimate the average prevalence of correlated binary outcomes. Specifically, assuming that there are patients and there are exchangeable binary outcomes per patient denoted by the parameter of interest is . We want to estimate this parameter and construct its 95% confidence interval. The observed data can be summarized by where is the number of patients with exactly positive responses. Under the exchangeability assumption:do we actually need the exchangeability assumption? The vectors are IID, so their sums are IID with sample space {0,1,2,3}, which we treat as the categories of a multinomial

where Under this model, the parameter of interest

To this end, we first consider a test statistic for testing the null hypothesis

where

The exact p-value can be calculated by

where the probability is with respect to the random variable

where

and

If is available, then we can reject or accept the hypothesis that depending on if

To calculate we need to calculate for all such that which is infeasible in practice. Instead, we can select “dense” grid sets to cover the parameter space

For a given we can select a small number and approximate by

where Here can be easily calculated by a Monte-Carlo method. Specifically, we can simulate a large number of ), and calculate the corresponding is the proportion of s greater than observed

The 95% confidence interval of is then can be constructed as all with an estimated

The entire procedure can be described in the following algorithm

* For
  + Simulate from unit exponential distribution and let
  + For
    - Simulate
    - Calculate
    - Calculate
    - Calculate where does the variance formula come from? We just want transpose(contrast)\*(var matrix of multinomial at p)\*contrast, don’t we? But this doesn’t seem to match eg binomial terms on the diagonal.
    - Yes, there are typos. Essentially, it should be something like
* For
  + Identify all such that
  + For each identified 
    - Calculate
  + Calculate the proportion denoted by which is this? Is it the same in “For Since is sampled under the corresponding to , so shouldn’t we use that to center ?
  + The here is the former. Calculate
* Denote the resulting p values by and the final 95% confidence interval for can be constructed as

In addition, we may consider a different test statistic. In particular, when is expected to be very close to 0 or 1, then it can be more appropriate to consider a test statistic in the form of

In my implementation, I find nice behavior for a range of true parameters values. However, there is a somewhat complicated relationship between the tuning parameters epsilon, the number of p vectors we sample from the simplex, and the density of the theta values in our grid.

I focus on a hard case here, a sample size N=10, with 4 multinomial categories as above, using the simple test statistic T, and the true p vector is close to the boundary. Results are still reasonable when the true p vector is (.1,.1,.1.7). Here is the empirical CDF of the p-values at the null in a simulation, which are close the expectd uniform CDF. For example empirical coverage is .946 at the alpha=.05 CI level.

Chart, scatter chart

Description automatically generated

When the true p is (.05,.05,.05,.85), the coverage inflates to .98 unless adjustment is made to the epsilon. At p=(.01,.01,.01,.97), the coverage goes to 1. It becomes necessary to increase epsilon. Otherwise, this close to the boundary of the simplex, there will be no sample p’s in an epsilon neighborhood. (The other option would be to drastically increase the number of sample p’s we take.) The max over such a large neighborhood tends to be larger. We will probably have to give guidance on the choice of tuning parameters; we can give simple suggestions relating the suspected distance from the edge to the required number of sample p points. How are your student’s results?

* + She has had systematic summary yet. She only tried the good cases (not extreme probability vector) and the coverage as expected is quite close to 95%. The epsilon can be overcame, for example, we can

1. Identify all such that
2. Project those probability vector onto the hyperplane
3. Generate theta.star with those projected probability vector.
4. Note sure if we use the new test statistic (the last display) can make the CI less conservative.

I wondered about a different approach. Instead of sampling on the simplex everywhere and and looking at a neighbordhood of a theta value, maybe we can sample from the pre-image of theta. I.e., we start with a grid of theta values [theta\_1,…theta\_n], then for each theta\_i, we sample m probabilities p\_ij,j=1,…m, where each p\_ij is a probability vector [p\_ij0,…p\_ij3] such that theta=0/3\*p\_ij1+…3/3\*p\_ij3.

We can do that. I think that the projection described above does achieve the same effect here. The only thing I am still not entirely happy is it seems that we still need to simulate those (p[0], p[1], p[2], p[3]) such that p[1]/3+2p[2]/3+p[3]=theta. Is there a non-random way to generate those grid points over the simplex with user specified “density” globally or near a “theta” value.

Here is the case where the truth p=(.01,.01,.01,.97). With sample size N=50 the p-values at the null become much closer to uniform; coverage of a nominal 95% CI is observed to be 88%.

Chart, line chart, scatter chart

Description automatically generated

Technically this approach is a little more difficult since it requires sampling on the intersection of a hyperplane with the probability simplex—I think I figured out a way, but I am not sure if the sampling is uniform. On the other hand, the benefit of this approach is that the tuning parameters are much simpler: A parameter for the density of theta, and a parameter for the density of the probability vectors corresponding to a given theta value. Increasing one or both straightforwardly increases the accuracy of the CI. (We could of course also have the density of the probability vectors per theta depend on the value of theta to reflect prior knowledge of where theta is.) Whereas in the current approach, even for fixed grid size, epsilon cannot be too large or too small, there is an ideal range it must lie in. I can investigate this approach further if you think it is a good idea, or we can stick with the original plan, or we could present both sampling methods. I don’t want to over-complicate the project, though, since it is already far along.