Primer on Hierarchical Bayesian Analysis*

First, don't let the fancy words confuse you.

It is Hierarchical simply because we have more than one level of analysis. We have individuals that are nested within species-stage groups (i.e. each individual can belong to only one species-stage group combination at the time of sampling and this assignment is not random). Individuals are also nested within a particular site – they are sampled in only one site, again not random. And finally sites are nested within watersheds.

We are interested in the probability that an individual from a particular species-stage group has Bd, but that will depend on: the individual (there is variance in Bd among individuals from the same spp-stage group at the same site), the spp-stage group to which the individual belongs (some species/stages are more likely to have Bd), the site the individual inhabits (is Bd present in that site? is Bd in nearby sites? is Bd even in the watershed to which the site belongs?). Maybe the probability an individual has Bd also depends on the season in which it is tested – individuals may be more likely to have Bd later in the season. I should note here that we are using individuals from the same species-stage group within a site as replicates.

So, we have a whole pile of possible independent variables affecting the probability that an individual has Bd, many of these variables are likely to interact, and we cannot assign independent variables at random to individuals because the variables are nested within each other. (Although it would be a cool trick if I could assign species at random to individuals!)

With me so far?
Now we could throw this all into a big regression model (using a logistic regression, of course, because presence of Bd is binomially distributed) and try to get a maximum likelihood estimate for the model with various independent variables included or not, and generate coefficients for each independent variable. In the original dataset we have 103 sites and 41 spp by stage combinations. If we had every spp-stage group for every site (and of course we do not) that is well over 4,000 2-way interactions to estimate. We have 1129 data points. Okay, so we cannot estimate interactions, we can only estimate the 154 main effects. BUT we do not have each spp-stage group in each site, so we have an extremely unbalanced design. Also in many instances we have only a single individual from a spp-stage group at a site, i.e. no replication. And I've not yet mentioned the nested effect of sites within watersheds.

Maximum likelihood estimators and their ilk still depend on data that is normally distributed with large (close to infinite) samples and uses that to estimate parameter values. But this is not always a good assumption, e.g. when you have binomial data that is zero-inflated (mostly zeroes with a few samples >0) – which is what we have. It is also a bad assumption when your data are

* This is a long-winded informal explanation I wrote for my co-author of why I used the analysis I did for the BC frogs data. 'Bd' is an abbreviation for the fungal disease that we tested for on amphibians throughout BC. We were interested in assessing the potential factors influencing the spread of Bd.

written by Jean Richardson

unbalanced (because you cannot unambiguously partition variance among different independent variables).

SO, what actually happens if you try to put these data (or rather a simplified version of them) into a logistic regression and estimate ML numerically?? Well, the computer thinks and thinks and thinks and then either crashes are sometimes actually politely lets you know that you have given it an impossible task. It is not even possible to re-create the analysis done here using standard statistical methods. I did try for fun quite awhile ago and I could not get R to run even the simplest model I tried (main effects only, no watersheds, etc.).

And this is where we get to the Bayesian part of the analysis. Instead of trying to take the data we have and fit a model to it, we start by creating what we think is a reasonable model of what is going on. So, for example, we decide that the probability Bd is in a site likely depends on the probability that Bd is in the watershed in which that site occurs and then some site-dependent probability of that particular site having Bd. But we also know that spread of Bd has a stochastic component (spores surviving transit, etc.) and so we also incorporate some error variance (epsilon) around the expected values for watershed and site. (See my annotated WinBUGS code for info on each of the model components.)

We then take our reasonable model (summarized in the DAG (= directed acyclic graph) of figure 2) and we take the data we have from our samples and we use these fancy numerical algorithms (MCMC, Metroplis-Hastings, for example) to estimate the parameter values in our model that best reflect the actual data we have. We start with random estimates (if we have no prior information) and just keep plugging in new estimates until we end up with 'good' estimates. Now, we could conceivably try every possible parameter value for every parameter and all possible combinations, but this would take years to do (remember we have >250 parameters) and we do not want to take years, so that is where the fancy algorithms come in – they move around the possible parameter space in sophisticated way to obtain parameter estimates that are the best possible (we hope); we do a couple of things to try and make sure we get the best possible estimates. Because we cannot test all possible values and we are starting with random estimates, we run the model using 3 different (random) starting values [these are the 3 chains referred to in the literature]; we check that the estimates we are getting from each of the 3 chains overlap nicely so we know that our random starting values are not determining our final estimates. We also discard the first few iterations of our estimates, since we know that at the beginning the algorithms are likely to have parameter estimates that vary wildly until convergence is closer; this is called 'burn-in'. Finally, we do not use the estimates from every single iteration because the algorithms do depend on the previous estimate, so there can be some auto-correlation between sequential iterations; this is called 'thinning' – we only save the estimates from every x iterations (40 in our case).

Now, besides being able to actually estimate parameter values, there are a few other pluses to this methodology. Namely the unbalanced replication and even lack of replication for some combinations does not matter – the estimates are based on matching the model to the data we have whether we have replicates or not (although we of course get much larger CI with fewer replicates) and we get not just a mean parameter value estimate, but an actual parameter distribution estimate, so we have better information about the parameters and credible intervals

(instead of confidence intervals) that truly tell us gives us a parameter value range that has a 95% likelihood value of containing the true parameter estimate (as opposed to an estimate of where it will be if we took many independent estimates – remember that with traditional statistics we have to assume that at very large sample sizes the data become normally distributed).

Finally, once we have estimates for the parameter values, if we get more data we do not need to run the analysis from scratch – instead we can use the parameter estimates we have as our starting values and simply add in the additional data to update our model estimates. This is the truly Bayesian analysis that is now in the MS for the 3rd section of the analysis.

So, there are 3 parts to the data analysis in the MS:

1. Determining the best model. This is done using the original dataset and including only those spp/stage by site combinations with n>2 (1001 observations; 69 sites; 31 spp:stage groups; 24 watesheds). We try 10 different models (see Fig. 3) and use a cross-validation technique to assess the model. Cross-validation involves 3 steps: i) run the model with all available data; ii) re-run the model after randomly removing 10% of the data and coding it as 'missing'; iii) ascertain how frequently missing data are correctly assigned Bd status. This is repeated 10 times for each model and these values are what is plotted in Fig. 3. We presume that the more accurately the model can assign Bd status, the better the model is. If adding additional variables (e.g. Julian day) to the model does not improve its accuracy, then we know that variable is not contributing any information to the model. Like with AIC we look for the model that has the highest accuracy with the fewest number of predictors.
2. Having decided on the best model and run the full (original) dataset (1129 observations; 103 sites; 14 species; 28 watersheds; 41 spp:stage combinations) using this model, we now consider the ability of the model to estimate new data. We take the addition 839 observations (data from 2010-11 & from VI sampling pre-2008) and run the model incorporating the new observations as data with Bd status coded as 'missing'. For prior values (starting estimates) we use the estimates from running the model with the original data, interpolating appropriately (see MS) for starting values not previously estimated). As with the cross-validation above, we then consider the ability of the model to accurately estimate Bd status for these new individuals. This did not turn out to be that good a test of the model (see MS results text).
3. We do a 'true' Bayesian analysis using the additional data. We take the parameter estimates we have from running the model with the original dataset and use those as starting values in our model. For those parameters for which we do not have a previous estimate (there are 3 new watersheds, 6 new species-stage groups and 13 new sites), we use random, non-informative priors (as before). We run 3 chains (3 different starting values, with random values changed for each chain, but previously estimated parameter values not changing). We discard the first 10,000 of 20,000 iterations as burn-in and use only every 40th iteration (thin = 40). 10,000/40 = 250, so we have 250 estimates for each parameter from each chain; 250 x 3 chains = 750 estimates for each parameter. Thus, in figures 4 & 5, those density plots are based on 750 values. Credible intervals for each estimate are simply the 5th and 95th quantile of those 750 values.