Lab 4 - Goodness-of-fit Testing and Covariate Standardization.

Part 1 – GOF Testing

Testing for model Goodness of fit (GOF) is a very important step in demographic analysis. A GOF test provides an assessment of how well your data matches the assumptions of the model you are trying to fit. It asks a fairly basic question of whether the distribution of your data matches the expectations generated by the model. So far we've ignored this important topic, largely for the sake of convenience while developing our understanding of the foundations of working in MARK. In reality, GOF testing should be one of your first steps when conducting a demographic analysis, because it can fundamentally change how you proceed with your modelling, or determine if you should proceed at all. Cooch and White cover GOF testing in great detail in CH 5 of the Gentle Introduction. What follows is intended to be a brief but practical review of that chapter and the methods they discuss.

We will continue to work in the recaptures-only CJS model structure, which has the best-developed theory and most practical tools available for GOF testing. In other situations, you will find that adequate GOF tests are just plain not available, such as for known-fate or nest survival models. Other modelling frameworks, such as multi-state and robust design models, will require you to use other specialized software (in those cases UCARE and RDSURVIV, respectively) to conduct a GOF test. The most important practical advice I can give you is to not shy away from GOF testing, and to think about it and investigate methods for applying it early on in any analysis. For one, there is the very real and important philosophical principle that you should not be making inferences from models that do not adequately fit your data. Second, there is the practical reality that most reviewers who understand CMR methods are not likely to give you positive feedback on a manuscript if you have not addressed this important step.

Overview – Recall that there are four fundamental assumptions inherent in a CJS analysis:

1) All marked individuals have an equal probability of recapture during each sampling occasion (*i*).
2) All marked individuals have the same probability of survival during the interval between occasions *i* and *i +1.*
3) Marks are not lost, and all marks are detected on captured animals.
4) Sampling is instantaneous and individuals are immediately released, relative to the interval between occasions *i* and *i +1.*

In situations where these assumptions are not met, estimates of ϕ or *p* are likely to be biased relative to the true probability distribution for the population. In that situation, extra-binomial variation is said to exist in the data. In practice, assumptions #3 and #4 are more easily met (but not always), and assumptions #1 and #2 are more likely to be violated. A violation of any assumption is indicative of *heterogeneity* in your input data – certain individuals or groups of individuals have capture or survival probabilities that deviate substantially from the sample mean. This is not automatically the end of the world, because we've already learned that we can account for heterogeneity in a number of ways. For example age structure is likely to produce heterogeneity if older individuals are more or less likely to survive or to be detected than younger ones, and this is the reason we explicitly incorporate age as a grouping structure

in the models. In fact, it's often true that the biological hypothesis we are interested in testing revolve around explaining heterogeneity among individuals with respect to their survival.

As a first step though, we must convince ourselves that we're using a model that adequately characterizes the underlying heterogeneity in the data (or that we've collected data that at least minimally conforms to the model assumptions). Remember that under an information-theoretic approach, we are trying to select from among a candidate set the model structure that provides adequate fit to the data while also reducing complexity – the model that is the most parsimonious. To do this, we start with a very general model (for example a group*time structure) and work progressively towards less complex models. Evidence in support of the simpler models rests on whether they provide similar or better fit to the data when compared with models that are more general. A prerequisite for that assessment is that our most general model provides us with an adequate fit to begin with. GOF testing gives us a method to conduct this assessment.

The "most general" model is the model with the greatest inherent complexity that can be built using a dataset and a model type (e.g. CJS). Typically this will be the full group*time structure on both ϕ and p in cases where groups of individuals are present, or simply the time-varying model if there is no group structure inherent to the data. You should always set your PIMs prior to attempting any GOF testing, especially if you are including age structure, as this will help to explain underlying heterogeneity in your data and improve the results of the GOF test.

MARK offers a number of methods for assessing GOF, and you should read through CH 5 to really get a good feeling for the differences among them. The most practical method in the CJS model structure uses Program RELEASE, which is a standalone application that MARK "calls" to perform the GOF testing. For an "under the hood" description of what RELEASE is up to, see the text beginning on 5-7. There are several alternatives to RELEASE that are discussed in CH 5, namely bootstrapped GOF and median c-hat. I will leave you to explore these alternatives on your own, and focus on Program RELEASE in class.

Running Release –

1. Open your analysis from the males-only European dipper dataset. If you no longer have that .dbf file, simply start a new analysis using the "ed_males.inp" input file.
2. The most general model from these data and the one we'll use is the phi(t) p(t) model. Either retrieve this model, or run the model if starting a new analysis (note the PIMs default to this structure, so all you need do is name it correctly and run it).
3. Once the model is run, go to the "Tests" tab on the menu bar and select "Program RELEASE GOF". Mark will just chug away without any further inputs, and will open a text file with the results.
4. The Release Output is extensive, and there are a number of things it can tell you about your data, particularly the "Test 2" and Test 3" statistics. If you are curious about these diagnostics you should read CH 5 in detail – they are actually fairly slick. Basically, they use patterns in the capture histories that allow you to diagnose whether extra-binomial variation in the data is a result of detection heterogeneity or survival heterogeneity. If you run release on your own data and get an unfortunately large value for c-hat, then you'll

definitely want to explore the results of Test 2 and 3, which will help you to understand where your problems lie. For our purposes today, we are interested in a single summary table, which is found about ¾ of the way down the results:

Goodness of Fit Results (TEST 2 + TEST 3) by Group

Group	Chi-square	df	P-level
1	9.5598	9	0.3873

Program RELEASE - Survival Rate Estimation with Capture-Recapture Data
Version 3.0(Win 95) Sep 1997 2-Feb-2014 14:05:06

This gives us the combined Test 2 and 3 Chi-Squared value, and the model df. Using these values, we can estimate the variance inflation parameter, ĉ, which provides an assessment of overdispersion, as:

$$\hat{c} = \frac{X^2}{df}$$

So in this case, ĉ would be estimated as 9.5598 / 9 = 1.06.

5. Based on the value we estimate for ĉ, we can move forward in a number of different ways.

Some basic guidelines for interpreting c-hat and moving forward in your analysis:

1. If ĉ ~ 1.0 then fit is adequate – no need to do anything. Happy Days.
2. If 1.0 < ĉ <3.0, the data shows some degree of overdispersion, and you should adjust AICc to QAICc.
3. If ĉ > 3.0, then unfortunately there are some major underlying issues with your data, and they may not be adequate for the model you had planned on running. You should not automatically panic here, because it is possible that you could include additional structure to your models (such as group or age variation) to reduce the amount of extrabinomial noise in your data. This is where taking a good hard look at the test 2 and 3 results may give you some insights.
4. What if ĉ<1.0? This is not an unusual "problem" to have, and it's one that doesn't have a terribly clear solution in the literature. The best advice I can give in this situation is to proceed as though your model adequately fits the data and use a value of ĉ=1.0. This is also what Cooch and White recommend (see sidebar on page 5-6).

Adjusting to QAICc – if you find yourself in situation #2 above, you should correct your AICc model selection to reflect the additional. The formula to do this is given as:

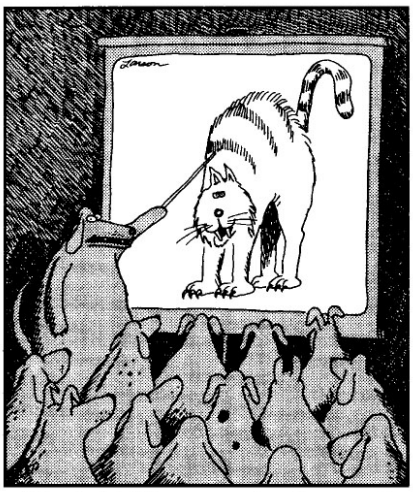
$$QAIC_c = \frac{-2\ln(L)}{\hat{c}} + 2K + \frac{2K(K+1)}{n-K-1}$$

The only real difference here from regular old AICc is that now the -2 log likelihood term is divided by our value of ĉ. This adjustment makes interpretation among competing models more conservative, because we've acknowledged uncertainty in whether our most general model adequately fits the data. In practice, you will find it more difficult to distinguish among models when adjusting to QAIC. Notice that when ĉ =1.0, QAICc is exactly the same as AICc.

Luckily, MARK will adjust to QAICc for us. Under the "Adjustments" tab on the menu bar, go to the c-hat option, and enter in the value of 1.06. You will see that now the model selection results have changed to reflect QAICc, and the values in the results browser should have changed slightly. Recognize that in a full set of candidate models, this may cause some models to no longer perform well, and it may change the relative ranking of models within those that are competitive. So, you should not get too far along in an analysis before testing and adjusting for c-hat, because you risk "loosing" results that you may have thought were significant.

A few final caveats – Current methods for GOF testing do not incorporate individual covariate values, which in some cases may explain a substantial amount of heterogeneity among individuals. For example, if individual body mass has a strong effect on survival probability, you can account for that in your analysis using an individual covariate, but it will not be reflected in your GOF testing. RELEASE will still see that noise in your data as unexplained by the most general model structure. For that reason you should always consider ĉ to represent a conservative estimate of overdispersion in your data. It is possible that other methods for GOF testing, such as bootstrapping, can incorporate individual covariates and those methods are worth exploring if you run into overdispersion issues you suspect may be related to your covariates.

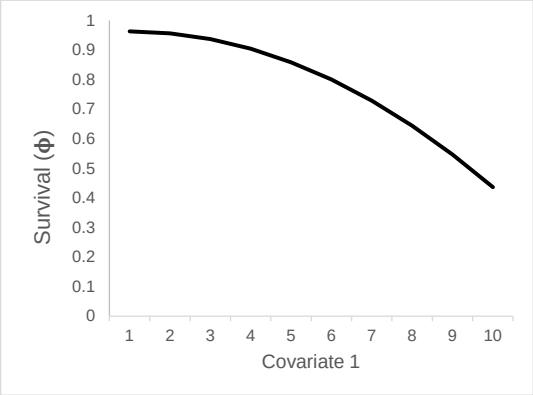
Finally, the above is a very brief primer on GOF testing designed to expose you to the general concept and the methods for accounting for it in MARK. It is sufficient for our purposes in this class and for you to move forward with your semester projects, but if you are using CMR methods for your graduate research or plan to publish CMR analyses, I highly recommend sitting down with Ch.5. and digesting the information contained there completely.

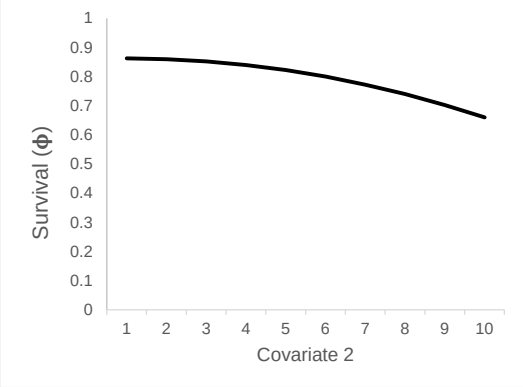


"Now, in this slide we can see how the cornered cat has seemed to suddenly grow bigger. … Trickery! Trickery! Trickery!"

Part 2 - Standardizing covariate values.

The beauty of using covariates in survival analysis is that they allow us to test for relationships between continuous explanatory variables and survival. This means we can not only evaluate the importance of biologically meaningful sources of variation, but we can also represent the consequences of that variation across a continuous range of possible values. This provides a lot of power in assessing the relative importance of any given effect; a covariate with a "significant" effect, but that produces a relatively small response in survival probability across the range of values you observe in the wild, may not be that biologically important. Consider the following two survival curves:





Both represent negative effects on survival that (depending on the strength of the dataset) may be supported as "significant" effects (i.e. we can conclude their slopes ≠0.0). However, across the range of observed values (1-10 in this case) covariate 1 produces a much stronger effect on survival than that of covariate 2. Therefore, the relative importance of Covariate 1 in determining the survival rates of our study organisms is also much greater. From a management perspective, for example, we might conclude then that we should place higher importance on addressing issues related to covariate 1, and can be less aggressive with respect to our treatment of covariate 2.

In the GLM framework we are using for survival analysis in Program MARK, the β coefficients for our covariate terms describe the slopes of the relationship between a covariate and survival (or abundance, recruitment, etc.). The sign of β tells us whether the effect is positive or negative, the absolute value of β tells us what the magnitude of the effect is, and the variance of β (SE and confidence intervals) give us a diagnostic as to whether the effect is "significant" (slope ≠0.0). You might surmise then, and you would be correct, that interpretation of the β values is a pretty important aspect of conducting a survival analysis with covariates (individual or group) in MARK. The

purpose of this “supplemental handout” is to describe a few “tricks” that you can use to help with use and interpretatopm of covariates.

Standardizing Covariate Values

One common method that is used when including covariates, in a survival analysis or otherwise, is to z-standardize the covariate values prior to analysis. Performing a z-transformation is relatively straight-forward:

$$Z = \frac{x - \bar{x}}{SD} \qquad \text{eq. 1}$$

Where the statndardized score for each observation (x) is calculated based on the mean covariate value for the sample (x-bar) and the sample standard deviation (SD).

Notice that this transformation is uniform across all levels of x; it does not change the distribution of the data as, for example, a log-transformation would. What it does is to normalize the distribution such that the sample mean = 0.0 and the sample SD = 1.0. When considering multiple covariates in a single analysis, whose raw values may be measured in different units, the z-standardized scores will place the covariates on a common scale. What this means is that the β coefficients and their variances will now be directly comparable among all covariates in your analysis. A covariate with β=0.6 has twice the magnitude of effect as a covariate with β=0.3. This offers some utility in “eye-balling” β values among multiple covariates to assess the strength of effects.

Also, and from a more practical perspective, the β values will now be, in general, more practical to work with, because most of your covariate values will fall somewhere between -3 and +3. Imagine an alternative scenario where you derive covariate values that span a very large range of values, say between 100 and 10,000, and they produce a “significant” effect on survival. Because survival probability is bounded by 0.0 and 1.0 (given the logit), the resulting value for β may need to be exceptionally small (e.g., β =0.000005)* to fit the true effect of the covariate on survival. It turns out that this can challenge the numerical optimization procedures when trying to maximize the likelihood using this covariate. For these reasons, a z-standardization is often useful.

**Note – I am conjuring these values out of thin air for the purpose of illustrating a point; I make no claim that they are mathematically plausible.*

In theory, if you are working with covariate values that span a reasonable range, and you are not interested in comparing among competing covariates, there’s no real reason to z-standardize. However, it was recommended to me that performing this transformation on covariate values is a very good practice to get into, and that we

should always do it on principle, and that's my recommendation to you as well. This is true of demographic analysis, and regression-based modelling in general. Performing a z-standardization should have no effect on the final interpretation of your analysis – the same models should perform equally well (in fact, AICc should not change) and the "significance" of covariate effects should be the same with or without standardization.

To check this, let's revisit our covariate models from earlier, but this time we will run the models with and without a standardization in place. There are a couple of ways to do this. We'll do it the "hard way" (really not that hard) first by calculating our own z-standardized values and appending them to the input file. This will also give you some practical experience manipulating and saving input files. To do this, use the following steps:

1. Open the indcov.inp file in Excel. Notice that you will need to change the file type in the "open" window to "All Files".
2. Excel should open a text import wizard to bring in the .inp (equivalent to a .txt) file extension. Select the "Fixed Width" option and click next.
3. On the next screen, you'll see a data preview window with several black lines running through it. These are indicating where Excel will split the data into multiple columns. Notice there is nothing distinguishing the last data field (the squared body condition values) from the semicolons. Click here to add another delimiting line, and hit next.
4. Finally, we need to tell Excel that the encounter history is a string of text. This is important, because Excel may attempt to read the history as a number, which will prevent us from bringing it back into MARK. For example, a history of 000011 may be read as 11. In the final window, highlight the encounter history column, and click the "Text" button above. Click Finish.
5. Once the spreadsheet opens, scroll down to the bottom (row 3500). Remember that column C represents our body mass value, and column D represents the body mass squared. For each column we can compute the mean and SD using the average() and stdev() functions in excel.
6. Based on the mean and SD for each covariate, we can compute the standardized scores in the next two adjacent columns, given the formula above. Notice that if you anchor the row numbers for the mean and SD cells (e.g. C$3501) you can drag the formula across rows and columns to fill in all ~7000 cells fairly quickly.
7. Finally, we need to set the cells with the values we just calculated (currently they contain formulas). To do this, highlight the two columns we just created, right click and select "copy", right click again and select "paste special" and chose the "values" option. Click "o.k."
8. Now, delete all the mean and SD values we used earlier (MARK will read these as data otherwise). In practice, it's often useful to copy the entire sheet to a

second worksheet at this point, so that you retain the original un-standardized values and the estimated mean and SD for each. We'll see why this is important later.

9. Finally, we need to move our semicolon column to the right of the data field so that the file will again conform to MARK input file format. A quick "cut" and "insert cut cells" operation is the easiest way to do this.
10. Now, we need to save the input file again, but need to ensure it remains in a readable format for MARK. There are probably a number of ways we can do this, but I will walk you through my standard work flow. First, save the spreadsheet as a CSV (Comma Delimited) file type – be sure you don't choose one of the other CSV options. Also give it a new name (indcov2, for example) so you don't overwrite the original.
11. Now, open the .csv file you just created in Notepad. You'll see that all the unique data are now separated by commas. Strictly speaking, MARK will read this file as is. In practice, I've found that space delimited files give me less trouble, and so I always replace the commas with a space first. To do this, go to Edit -> Replace, and tell it to find the commas and replace them with a single space.
12. Simply save the resulting file using the "save as" option, and add the .inp extension on the end of the file name (e.g., "indcov2.inp"). You should be ready to bring the file back into MARK.
13. Start a new MARK file using the CJS module. Set the model run up as before, but be sure the results file is saved using a new file name. Instead of 2 covariates, now we'll specify 4 (the original body mass and (body mass)^2 covariates, and our newly standardized covariates. Name the original two variables the same thing you did in the prior analysis (BM and BM2); we'll see why this is important shortly. Name the new ones BMS and BM2S.
14. Once mark opens, let's run the most general model to start, which requires no adjustment of the PIM structure. Name and run that model.
15. Now for a neat trick – from the menu bar, select "Run" and the "File Model(s)" option. Navigate to the previous .dbf file you created from these data, mine was called "IndivCovTest.dbf". MARK will warn you about uneven numbers of covariates, click o.k. to ignore it. A window will open with all of the model structures we previously ran that were stored in that results file. Click the "select all" option, and o.k.
16. MARK will chug through all of your old model structures, saving you the time needed to re-build the design matrices, and you should be left with our original results window.
17. Retrieve the "best" model, which contained both the body mass and (body mass)^2 effects. In the design matrix, replace the original covariates with our new standardized variables (BMS, BM2S). Run the model and somehow name it so that you can distinguish it from the previous one.

18. You should automatically see that the AIC results are identical. Let's take a few minutes to look through the Real Parameter and Beta results as well.

Finally, let's also look at the "easy way" to standardize our covariate values. Again, retrieve the best model that was built using the raw covariate vales. Click to run it, and this time check the box under Numerical Estimation Options that reads "Standardize Individual Covariates". Give the model another distinguishing name, and run it, and marvel at the results, which are (or should be) identical.

So, why would we ever take the trouble to do it "the hard way"? Well, without getting too far ahead of ourselves, this is one of those cases where MARK is doing something that is a little black-boxy that can get us into trouble if we don't think about how it is working with the data. In this case, there are no inherent problems. However, when we begin working with more complex covariate structures (such as next class when we'll finally get a chance to use time-varying covariates), the approach we'll need to take to standardization may be very different than what MARK can perform for us. This takes some serious thought, and a good understanding of what you are actually doing when you correct the data using the z-standardization. For that reason, and until you really get comfortable with standardizing data, I recommend that you do it yourself in all cases.

Assignment 4 – Fish and Parasites

The data for this assignment is designed to replicate a scenario where we've conducted a mark-recapture study of a cool-water fish across an 8-year period. Fish are grouped by sex (Group 1 = Male, Group 2 = Female) and we have two covariates. The first will describe the length (fork length) and the second will describe a parasite load. We'll assume here that the parasites are visible externally, so we can gauge an individual's parasite load by counting the number of parasites observed at the skin surface. Finally, we've measured the average summertime water temperature for inclusion as a group covariate; these values are listed in the header of the input file (lab4assigntment.inp; available on Blackboard). You will notice that all covariates in this are z-standardized.

Please analyze this dataset appropriately using the Design Matrix, making sure to check for Goodness-of-Fit and adjust your results accordingly. In addition to turning in your AIC table, parameter estimates (real and beta), and Mark files, this week I would like you to submit a figure that shows the effect of one of the covariates on survival. Your x-axis for the figure should be given in *unstandardized* units. Any legible formatting is fine.