APPENDIX A

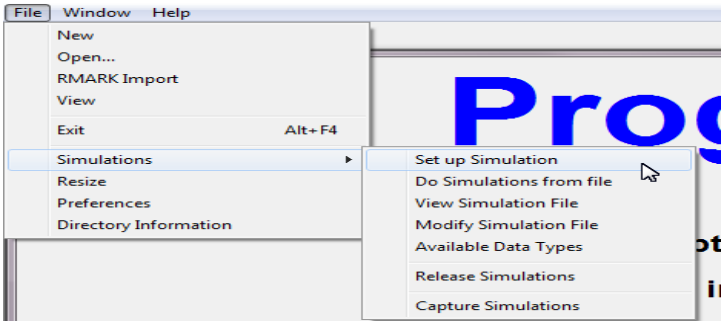
Simulations in MARK . . .

The ability to simulate data and fit models for various data types (e.g., Cormack-Jolly-Seber, multi-state, and so on) is very useful, for a variety of purposes. While it is possible to write your own simulation code (which has the advantage of making you acutely aware of the underlying structure of the model), **MARK** has a convenient and very powerful built-in simulation capability.

At present, **MARK** can simulate data under three primary systems: program **RELEASE** simulations, program **CAPTURE** simulations, and simulation of models developed in program **MARK**. In this appendix, we'll focus on the simulation of data for two common data types – a simple CJS design, using both **MARK** and **RELEASE** simulations.

A.1. Simulating CJS data

To simulate CJS data in **MARK**, start **MARK**, and select '**`File | Simulations`**'. At this point, you'll see that you are given several options of how you want to simulate the data:

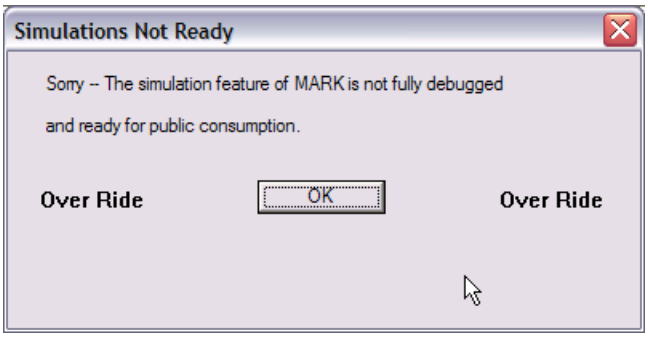


Notice that one of the options presented in the drop-down menu is for '**`Release Simulations`**'. You might recall that program **RELEASE** is appropriate for CJS data, which is what we're going to try to simulate here. So, in fact, for this example, we could select the '**`Release Simulations`**' options from the menu. However, here we introduce the more general approach to simulating data in **MARK**, which can be used for data types other than CJS (we will revisit **RELEASE** simulations in a moment).

In addition, one of the selections in the '**`Simulation`**' menu will generate a list of the '**`data types`**' in **MARK** under which you can simulate data. Most (but not all) data types can be simulated.

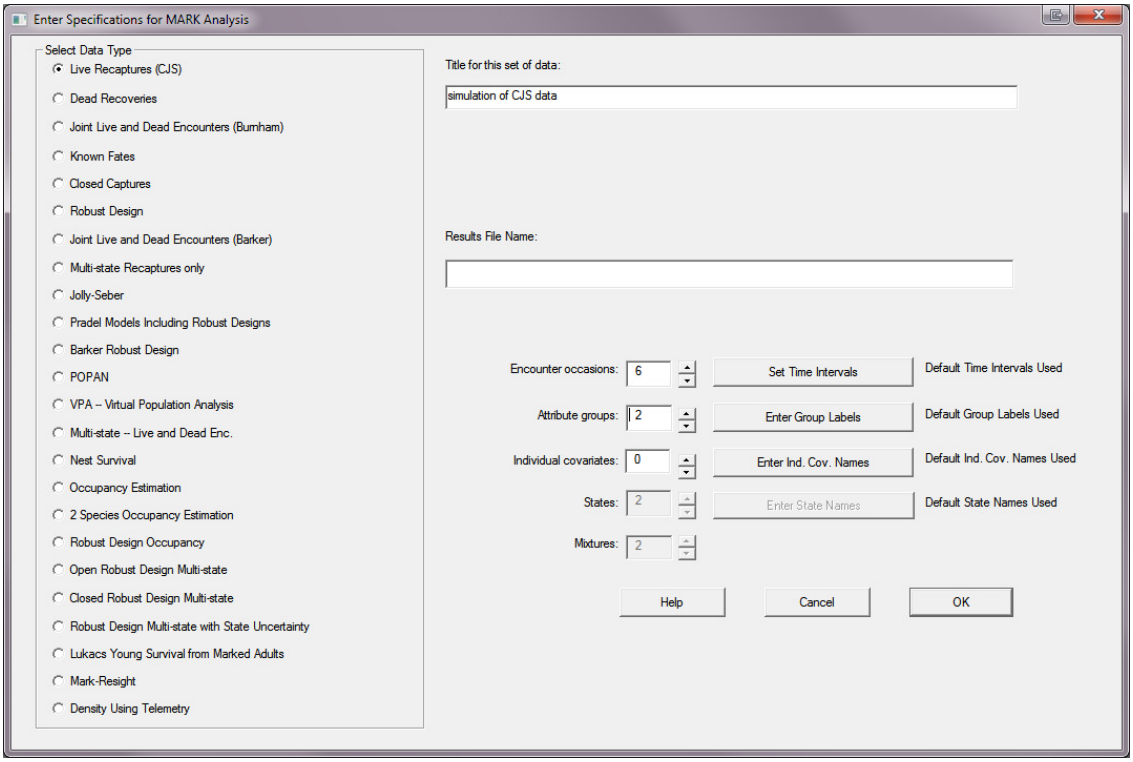
A.1.1. Simulating CJS data – MARK simulations

To simulate CJS data in **MARK**, we need to select '**`Set up Simulation`**' from the drop-down menu. Selecting this option in **MARK** will generate a puzzling popup window:



This window is trying to warn you that not all elements of the simulation feature in **MARK** are fully debugged. If you try to click the '**`OK`**' button, a little '**`OverRide`**' message will keep popping up, and jumping from side to side of the window as you attempt (in vain) to click on it.

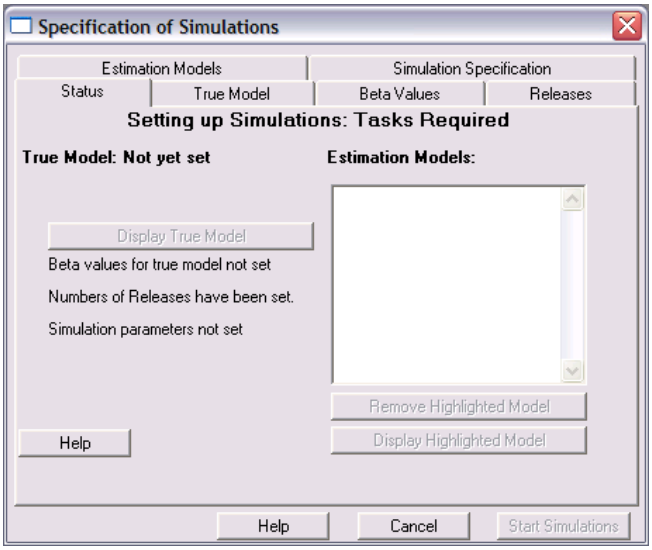
However, if you double-click anywhere in the grey area of the window (not the '**`OK`**' button), you are in fact able to over-ride the warning window, and will (finally) be presented with essentially the **MARK** specification window, which you're probably quite familiar with by now.



It's at this point we specify the data type we want to simulate – in this case, CJS '**`recaptures only`**'

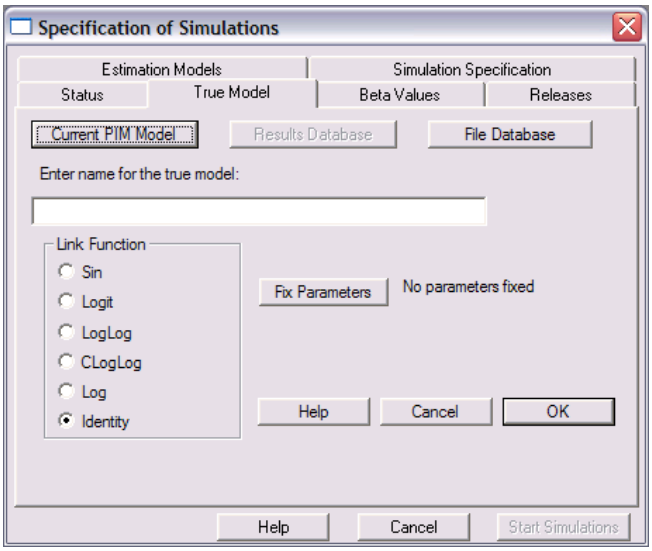
data. Let's simulate 2 groups, 6 occasions. Note that we've entered a title for the simulation, but have left the box for '**`Results File Name`**' blank – since neither are needed.

Once you click the '**`OK`**' button, you'll be presented with the main simulation setup window.



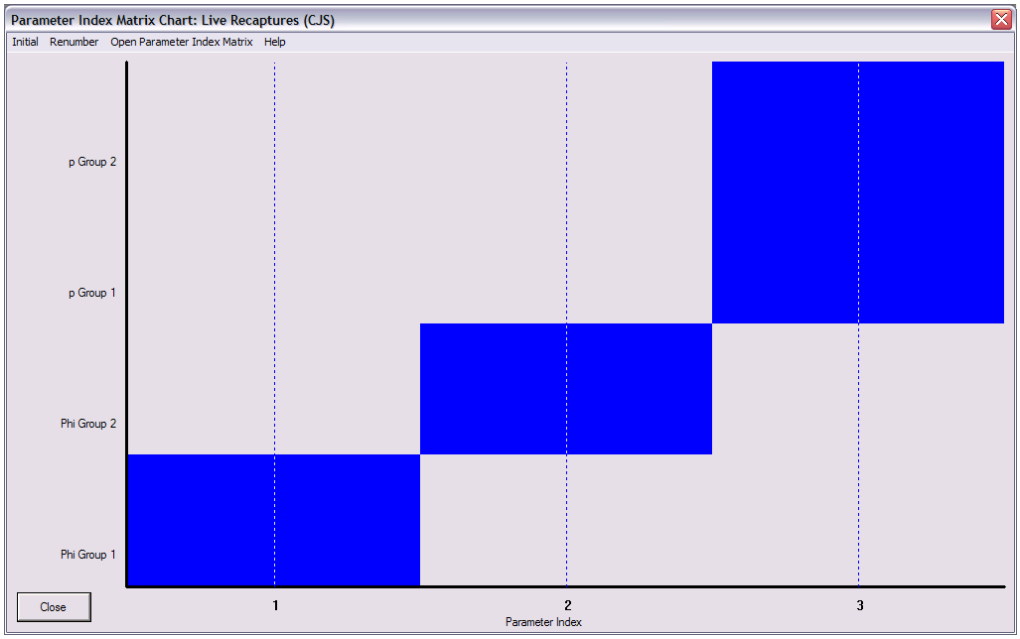
The window consists of a series of tabs, each of which correspond to a particular aspect of the simulation that needs to specified before you can run the simulation. It's easiest to go through the various steps sequentially, corresponding to each of the tabs in the simulation setup window.

The first step is to specify the '**`True Model`**', by selecting that tab.



Now, we need to specify the parameter structure of the 'true model' we want to simulate. For purposes of demonstration, we'll use model - $\{\varphi_g p_.\}$ – differences between the two groups for φ, but no time variation, and no group or time differences for p. So, the first thing we need to do is set up the appropriate

parameter structure – this is most easily accomplished by manipulating the PIM chart. By now this should be pretty familiar territory.



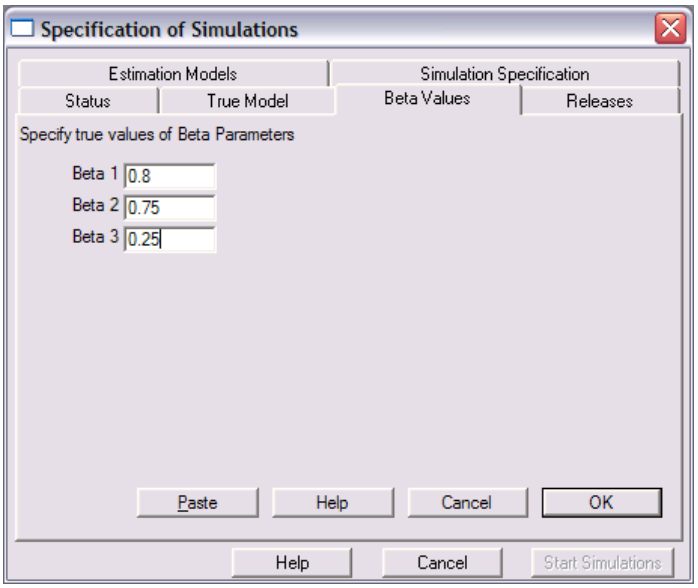
Several important things to notice here before we proceed. First, as shown on the preceding page, we've specified the identity link (rather than the sin or logit link) – this is important since we want to enter the parameter values for our model on the real scale (this is simply for convenience – we could of course calculate the sin or logit transformed values for each parameter, if we so chose). Next, notice that there are 3 radio buttons just above the title box: one (left-most) is for '**`Current PIM Model`**'. The next (greyed-out) is for the '**`Results Database`**'. Finally, right-most, is the '**`File Database`**'. For the moment, the one we're interested in is the the first one – we click '**`Current PIM Model`**', so that **MARK** will know to use the parameter structure we just created (corresponding to $\{\varphi_g p_.\}$), for our simulation. When you click this button, the title will be replaced momentarily with '**`???`**'. No worries – simply enter the title for your true model (you might use '`true model`', or something more explicit). Then, click the '**`OK`**' button. Notice that now the button '**`Display True Model`**' is now active (it was previously greyed-out). This button allows you to check the true model, if you want.

Next, we select the '**`Beta values`**' tab – here is where we put in the scale-appropriate β values for the linear model corresponding to our model. Since we specified the identity link, all we need to do is specify the parameter values on the real scale. For this example, we'll use the following parameter values: $\varphi_{grp\,1} = 0.80$, $\varphi_{grp\,2} = 0.75$, $p = 0.25$. So, a difference of 0.05 in apparent survival between the two groups, and a (relatively) low detection probability. So, we might be interested in how large a sample size of newly marked individuals we need to release on each occasion in order to allow us to detect a difference in survival of this magnitude.

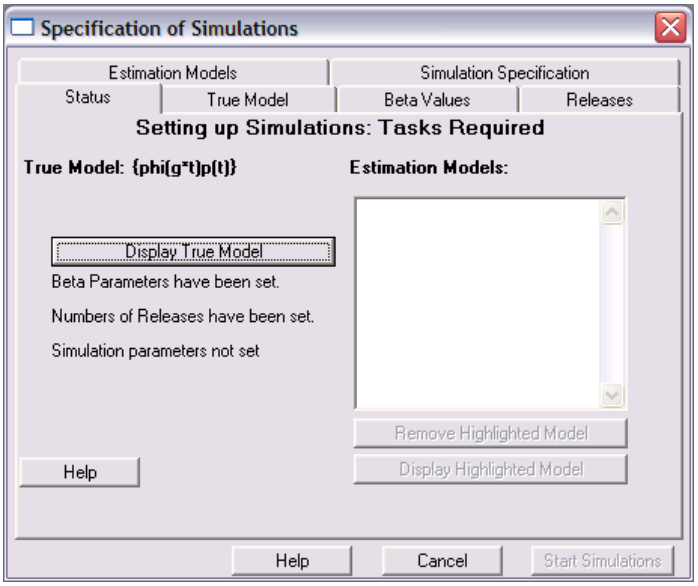
So, all that we need to do is enter these β values into **MARK**. These values are used with the the design matrix and the link function to produce the value of the real parameters. A good choice for the link function to be used with an identity design matrix is the *identity link function*, because then the values for the β parameters are the same as for the real parameters. For other link functions, the β parameters must be set to values that will produce the correct value of the real parameter via the link function. There is, however, an exception when the β values entered define the exact parameter being

estimated. This issue occurs with simulation of the robust design models (see section A.4).

For the moment, though, we can proceed using the identity link function. Simply click the '**Beta Values**' tab, and enter the parameter values.



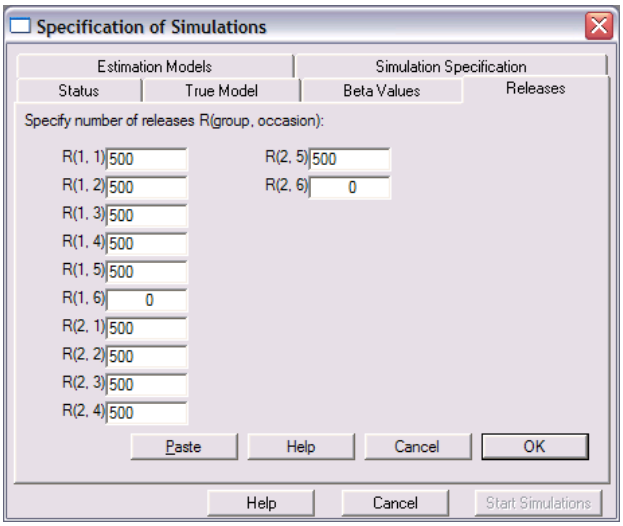
Once you've entered the appropriate values, and clicked the '**OK**' button, you'll be popped back to the main part of the setup window.



Notice that **MARK** conveniently provides you with visual indications of which steps in setting up the simulations you've completed (never let it be said that **MARK** isn't user-friendly!).

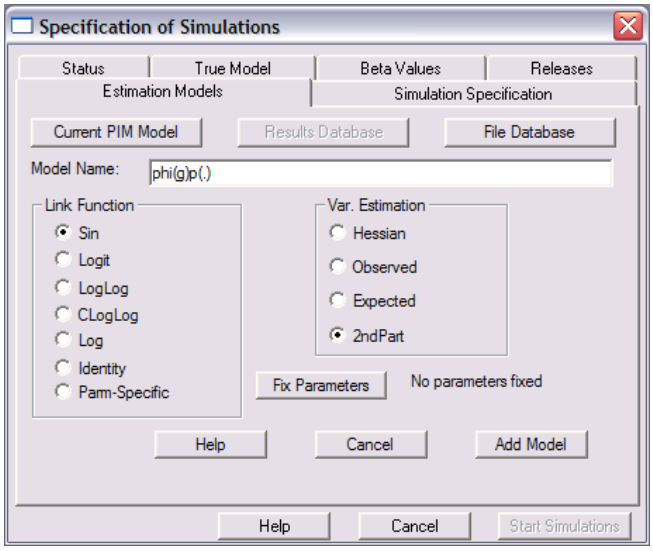
Next, we want to specify the number of releases of newly marked individuals at each occasion. So,

we click the '**Releases**' tab. For this simulation, we'll try 4 different 'release experiments' – to explore the influence of sample size on the ability to detect true differences in survival between the two groups. So, we'll try releasing 500, 250, 100, and 50 newly marked individuals at each occasion, respectively. We'll need to do this one 'experiment' at a time – here is what we would enter for the '500' sample size simulation:



Notice we don't release new individuals on the last occasion, since these individuals will provide no information (given that the study terminates on the last occasion).

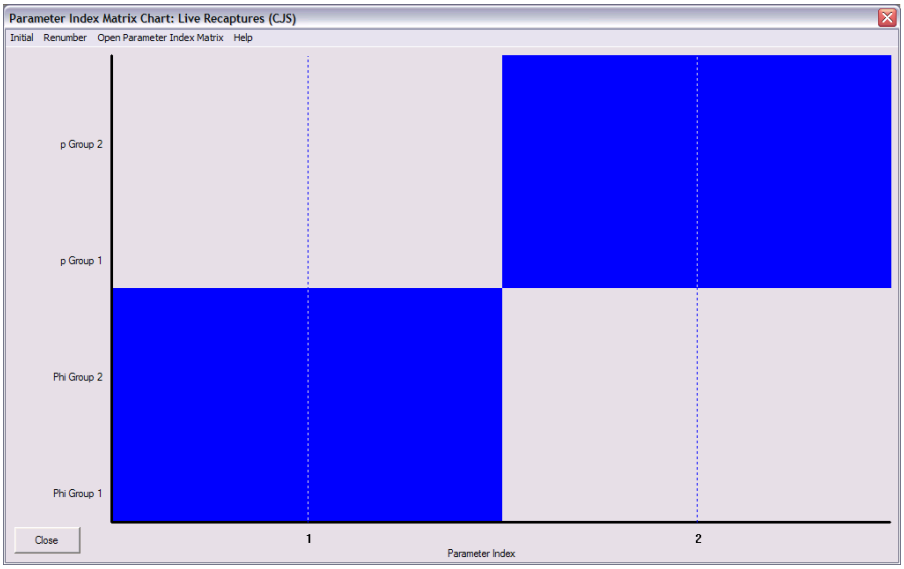
Next, we want to specify the models we want to fit to the simulated data. To do this, click the '**Estimation Models**' tab.



Here, we could simply simulate the current (true) model, by clicking the '**Current PIM Model**' button, or we could specify another model. If we choose another model, we need to specify the PIM structure for

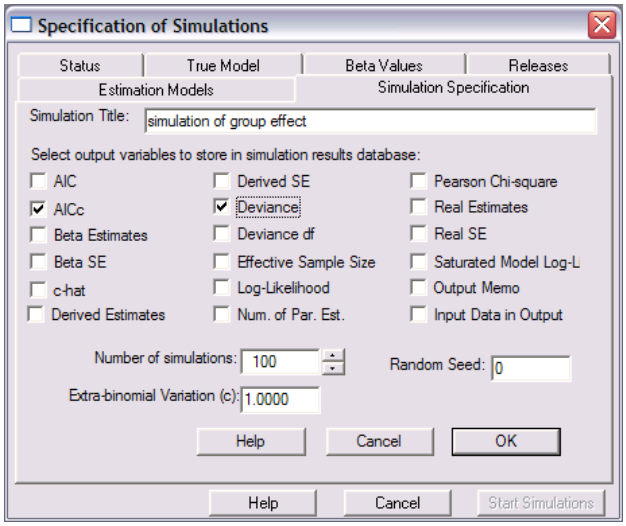
the model, in the usual way. For our experiment, we want to compare fitting the true model $\{\varphi_g p_.\}$ with a reduced parameter model with no group effect on survival $\{\varphi_. p_.\}$. Since the true model is reflected by the current PIM structure, simply click the '**`Current PIM Model`**' button. Enter '`phi(g)p(.)`' as the model name. Then, specify the link function. What you're doing here is telling **MARK** which link function you want to use during the numerical estimation for a given simulated data set. Usually, we would use either the sin or logit link.

Once finished, click the '**`Add Model`**' button. Now, we want to add the reduced parameter model $\{\varphi_. p_.\}$. To do this, we simply need to select the '**`Estimation Models`**' tab again, open up the PIM chart, and change the parameter structure to reflect this model.



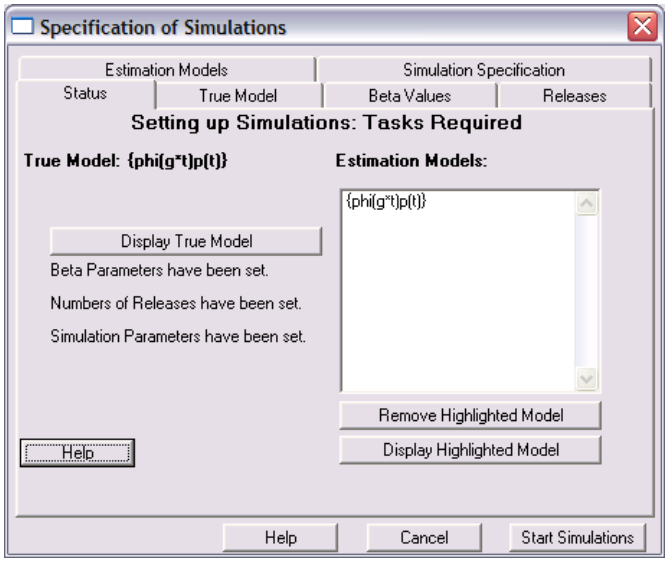
Once you've modified the PIM chart, simply click the '**`Current PIM model`**' button, enter '`phi(.)p(.)`' as the model title, and select the appropriate link function. Then, click the '**`Add model`**' button.

Finally, we'e ready to specify the simulation – click the '**`Simulation Specification`**' tab.



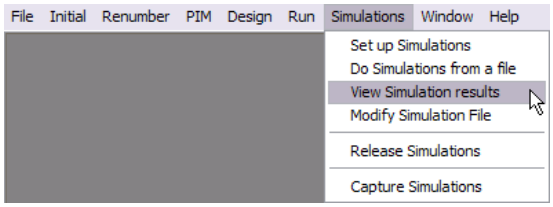
Here, you can specify the various statistics you want to output, how many simulations you want to run, and whether or not you want to add extra-binomial noise to your data (i.e., specify a specific, true value for c). For now, we'll run 100 simulations, and output both the AIC_c and the deviance for each model for each of the simulated data sets.

Once you've completed specifying the simulations, click the '**OK**' button. This will bring you back to the main simulation window.

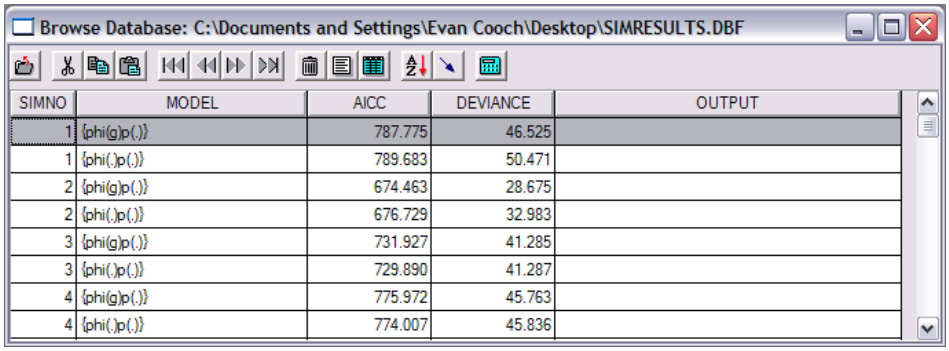


At this point, all that is left is to run the simulations. Simply click the '**Start Simulations**' button – **MARK** will then ask you where you want to store the simulation output. Once you've done that, **MARK** will starting grinding through however many simulations you've specified – clearly, the length of time this takes is dependent upon the model(s) you are simulating, the number of releases at each occasion, and how 'hot-rod' your computer is. **MARK** will present a progress bar as it works through each simulation.

Once completed, you need to 'do something' with the simulation results. While for our experiment we will ultimately want to extract summary data from the output, for the moment, we'll explore a couple of different ways to view the simulation results. To do so, select the '**View Simulation results**' option from within **MARK**.

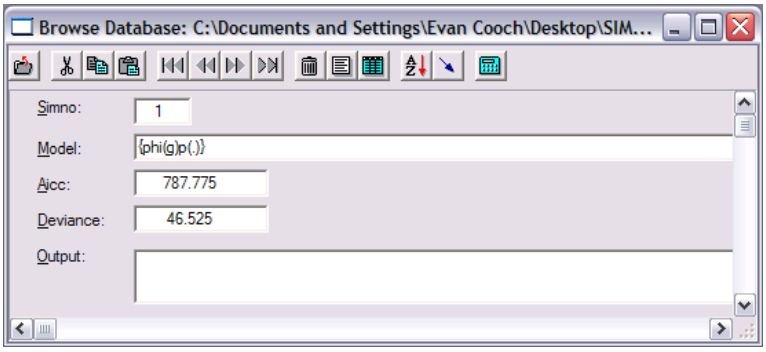


MARK will present you with a database tabulation of the simulation results – model name(s), and any of the statistics you requested (in this case '**AICC**' (i.e., AIC_c), and '**DEVIANCE**'). See the figure at the top of the next page.



SIMNO	MODEL	AICC	DEVIANCE	OUTPUT
1	{phi(g)p(.)}	787.775	46.525	
1	{phi(.)p(.)}	789.683	50.471	
2	{phi(g)p(.)}	674.463	28.675	
2	{phi(.)p(.)}	676.729	32.983	
3	{phi(g)p(.)}	731.927	41.285	
3	{phi(.)p(.)}	729.890	41.287	
4	{phi(g)p(.)}	775.972	45.763	
4	{phi(.)p(.)}	774.007	45.836	

The 'view' we're looking at is known as the '*browse view*'. There is another view known as '*form view*', which has some advantages we'll explore later on. To access 'form view', simply select the '**`Form View`**' icon on the toolbar (or by puling down the '**`View`**' menu and selecting the '**`Form View`**' option). Again, we'll discuss further uses of this view later, but for now, here's what it looks like at least – simply select a particular record from the browser, and then select '**`Form View`**':



For our experiment, we're interested in how much sample size affects our ability to detect a difference in survival between the two groups. Recall that in our true model, the difference was 0.05 (0.80 vs 0.75). There are a number of ways we can approach this question, using the results of our analysis of the simulated data, but for simplicity, we'll invoke the classical likelihood ratio test (LRT) approach (if you forget the basics of the LRT, see chapter 4). For each of the 100 simulations, we'll calculate the LRT test statistic, and assess whether or not the difference in model deviances (between the true and reduced parameter models) is significant at the $\alpha = 0.05$ level, for each of the 4 simulated sample sizes. For purposes of comparison, we'll also present mean ΔAIC_c and evidence ratios and model likelihoods for each sample size.

Recall that the evidence ratio of a given model relative to the best model is given as

$$\frac{w_1}{w_j} \equiv \frac{1}{e^{-1/2\Delta_j}} \equiv e^{1/2\Delta_j}$$

and the model likelihood (i.e., the probability that model $\{\varphi\, p\,.\}$ is the K-L best model) is simply the inverse of the evidence ratio (i.e., w_j/w_1). Note that you can't do these calculations directly with **MARK** – the simulation results DBF file must be 'imported' into some external spreadsheet or statistics program to allow you to generate summary statistics; e.g., calculate and tabulate LRT statistics.

sample size	*% significant*	*mean ΔAIC*	*evidence ratio*	*likelihood*
50	17	1.84	2.51	0.399
100	35	2.62	3.71	0.270
250	69	5.61	16.49	0.061
500	93	11.00	244.74	0.004

What is pretty clear from this table is that for a release (sample) size <250, there is a low probability of detecting a real survival difference of 0.05, given an encounter probability of $p = 0.25$. Only when >250 individuals are newly marked at each occasion is the probability of detecting a true difference of 0.05 in survival ~ 95%. These conclusions are supported by the evidence ratios and model likelihoods. Note that for sample sizes < 500, the likelihood for model $\{\varphi\, p_{.}\}$ is > 0.05, meaning that based on a nominal error rate of $\alpha = 0.05$, you would have no basis for 'rejecting' model $\{\varphi\, p_{.}\}$, even though it is false. Only when sample size is > 250 is the likelihood of model $\{\varphi\, p_{.}\}$ being K-L best < 0.05.

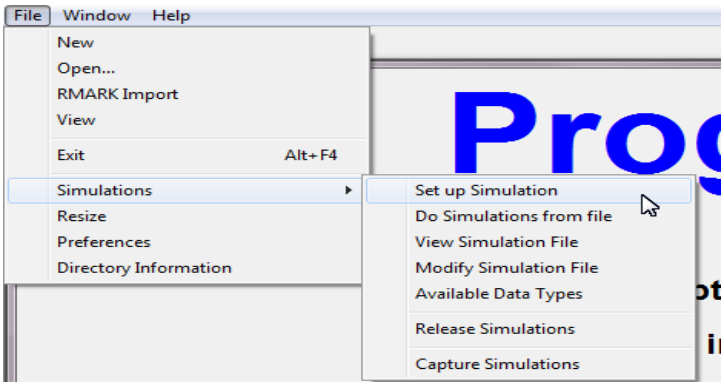
Of course, you may be familiar with many projects that would be hard-pressed to release even 100 newly marked individuals at each occasion. So, in such cases, your only option is to increase the encounter probability p (this is generally known as 'the big law' – in general, you should do everything possible to increase encounter rate). Failing that, you'll have to accept that there will be real limits to the power of the inference you can make from your data.

The preceding is an example of using the simulation capabilities in **MARK** to perform what is in effect a 'power analysis' – this could (and should) be done prior to a study to help evaluate the amount of 'effort' your study will require in order to achieve an inference of a given precision. Of course, in this example we've assumed a particular 'true model' – the results of your simulations will clearly be influenced by the choice of true models.

A.1.2. Simulating CJS data – RELEASE simulations

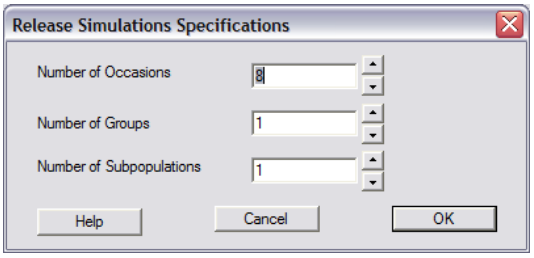
For this example, we'll look at simulating data under program **RELEASE** – to demonstrate that $\hat{c}$ is in fact an estimate of some underlying parameter, c (and as such, is estimated with uncertainty). This is important when using estimates of $\hat{c}$ for quasi-likelihood adjustments of AIC_c (see chapter 5).

To simulate CJS data under program **RELEASE**, simply start up **MARK**, and select '**`File | Simulations`**'. At this point, you'll see that you are given several options of how you want to simulate the data:



Notice that one of the options presented in the drop-down menu is for '**`Release Simulations`**'. Select this option. This will bring up a smaller window where you will be asked to specify the number of release

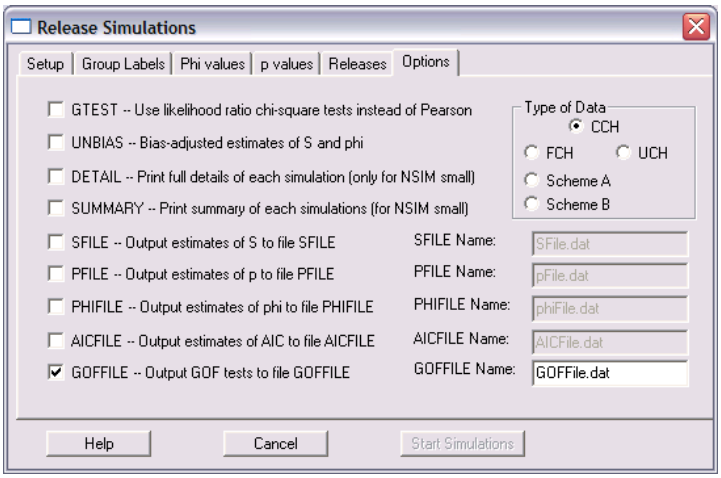
occasions, the number of groups, and number of subgroups. For this example, we want 8 occasions, and 1 group (and 1 subgroup):



Once you click the '**OK**' button, you'll be presented with yet another tabbed window, where each tab corresponds to a specific parameter or option. The default is for 1000 simulations, with extra binomial variation of 1 (corresponding to $\hat{c}$=1), and a random number seed of zero (meaning some random function of the computer clock will be used to seed the simulations). This is shown at the top of the next page. All you need to do at this stage is answer/complete each of the 'tabbed windows'.

Start by setting a group label, then setting value for φ and p respectively, the number of releases on each occasion, and some options. For our simulations, we want some time variation in both φ and p, with 250 releases on each occasion. For example, we'll use some random values between 0.5 and 0.75 for both parameters.

For the '**options**' tab, we want to output the GOF test statistics for each simulated data set (all 1000 of them, in this case), so we check the **GOFFILE** box. You may recall from some long-ago statistics class that you can use either a Pearson χ^2 approach to contingency analysis, or a likelihood-based G-test approach. The simulation option windows lets you choose either. For the moment, we'll use the Pearson χ^2 approach, so leave the **GTEST** box unchecked:



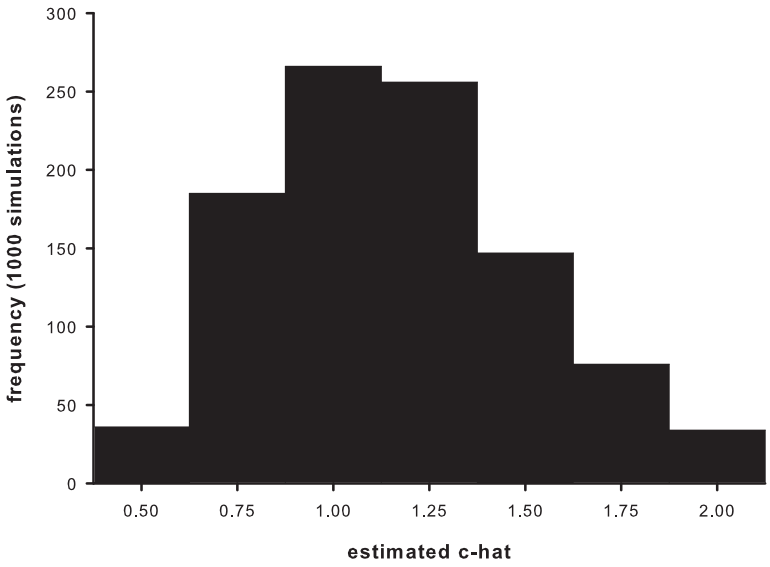
In the upper right corner of the options window are several radio-buttons corresponding to different data types. CCH refers to 'complete capture histories'. The other options are explained in detail in the 'big blue book', where **RELEASE** is first described. Most of the time, you're likely to be using CCH, so this option is selected by default.

Now you're ready to start the simulations. If you haven't completed all of the 'tabbed windows', the

'start simulations button' will be greyed-out (see bottom of previous page). If you are ready to run the simulations, this button will be active. Go ahead and run the simulations. If all goes well, the first thing you'll see is a Notepad window showing the basic **RELEASE** output – this can be ignored for the moment.

In addition (and most important for our purposes in this case), a file called `GOFFILE.dat` will be created on your computer. This file contains all of the results of the individual χ^2 tests, and **TEST 2** and **TEST 3** separately, as well as the sum of **TEST 2** + **TEST 3**. At this point, you'll need to parse this output file to extract just the **TEST 2** + **TEST 3** results. We'll leave details of how to do this up to you. But, for completeness, here are the results of our simulation.

If we plot $\hat{c}$ estimated as (**TEST2** + **TEST3**)/df



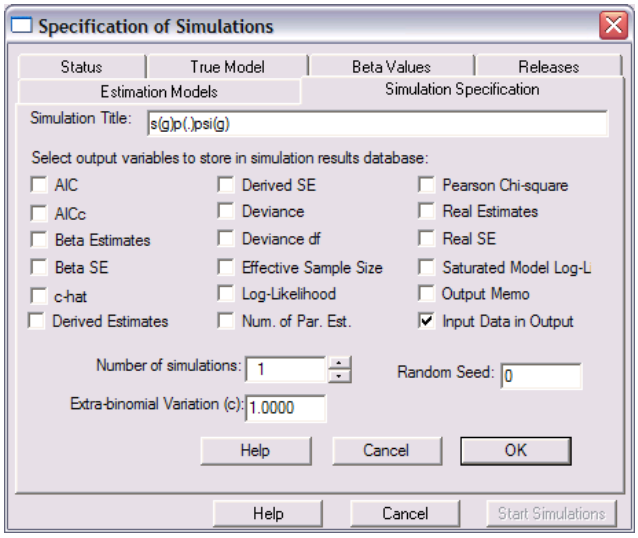
then we see that even though the true value for c in the simulated data is 1.0, there is considerable variation among simulated data sets in estimated $\hat{c}$. The implications of this uncertainty are discussed in some detail in Chapter 5.

A.2. Generating encounter histories – program MARK

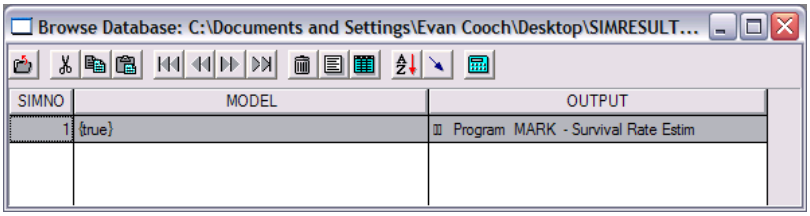
Here, we simulate some multi-state data. Our purpose here is primarily to illustrate some of the other features of the simulation tool in **MARK** – in particular, the ability to extract the encounter histories generated by the simulations (up until now, we've only considered the analysis results, not the encounter histories themselves). We'll simulate $\{S_g p \psi_g\}$ – simple group differences in survival and movement (no time variation), but no differences in encounter probability between groups, or over time.

As with earlier examples, start **MARK**, and and start the setup for a new **MARK** simulation. Select '**`Multi-strata recaptures`**' as the data type, 6 occasions, and 2 strata. Then, using the PIM chart, setup $\{S_g p \psi_g\}$ as the true model. Use whatever values you like for each of the parameters, and 100 releases of newly marked individuals on each occasion. Since we're interested in looking at the encounter histories for the true model, we use the current PIM structure for the '**`Estimation Model`**'.

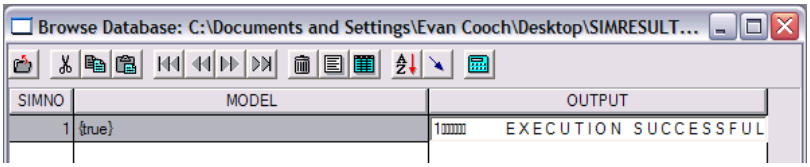
Finally, the only thing we need to pay any real attention to is the '**Simulation Specification**' tab. Here, we want to specify only a single simulation (i.e., change the '**Number of simulations**' to 1). Then, we check '**Input Data in Output**' – this will cause the input data (i.e., the simulated encounter histories) to be written into the simulation output.



Because you're only running 1 simulation, it will run very quickly. Next step is to '**View Simulation Results**'. We'll start with the standard '**Browser View**'.



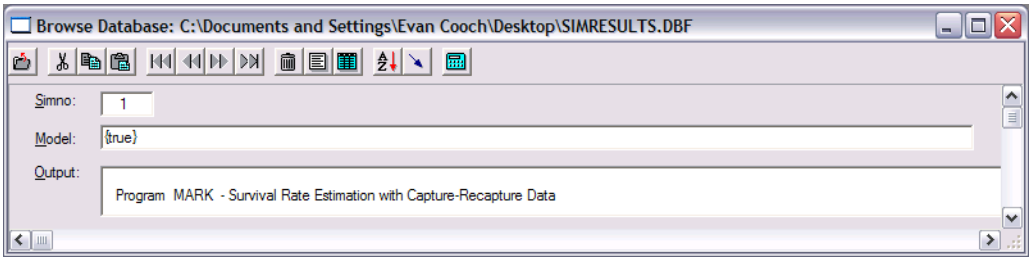
We see that the browser contains the simulation number (only 1, in this case, see we ran only a single simulation), the name of the model, and then a column labeled '**Output**'. To extract the encounter histories, double-click the cell in the '**Output**' column. When you do so, you'll see that the text in the cell changes:



What you're actually seeing is the first line of the entire (full) output that **MARK** generates. This full output includes the encounter histories. How do you actually access these histories (given that you can't scroll down the file from within the browser)? What you need to do is copy the contents of the cell

– you can do this either by using `ctrl-A`, or by right-clicking from within the cell, and selecting '**Copy**'. Once you have copied the cell contents into the clipboard, you next need to paste the contents into your favorite editor. After doing so, scroll down a few lines until you find the encounter histories. All that remains is to extract the encounter histories from this file (this is either easy or difficult, depending on your choice of editor). Thats it!

Earlier, we mentioned that there was another '**view**' option for looking at the results of your simulations. You can also look at your results using the '**Form**' view option. To access the '**Form**' view, simply view the simulation results. **MARK** will typically default to the browser view. You can switch to the form view either by selecting the appropriate button on the toolbar, or using the appropriate view menu option. For the current simulation of multi-state data, here is what the form view would look like:



As with the browser option, you can extract the encounter histories simply by selecting the text in the output window, and then copying the text to your favorite editor.

A.3. simulating data from a prior MARK analysis

Occasionally, it will be of interest to simulate data based on model structures you may have constructed for some other **MARK** analysis. How can you 'pull in' the model structure from another **MARK** project file into the simulation tool in **MARK**?

In fact, it isn't really that hard. There are at least a couple of ways you can do this. In one approach, you simply

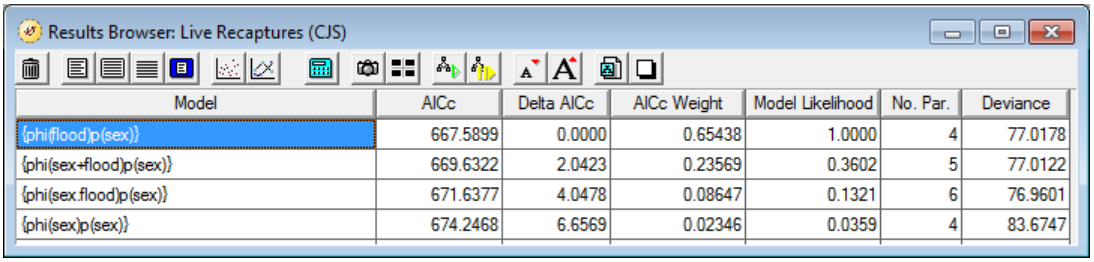
1. open up the **MARK** project of interest
2. retrieve the model of interest
3. then, from within the open **MARK** project, select '**Simulations | Setup Simulations**', and then '**override**' the popup window.
4. this will spawn the 'specification of simulations' window introduced earlier. Simply click the '**true model**' tab, and then click the '**current PIM model**' button. This will make the 'active model' (which you retrieved in step 2, above) the 'active model' for the simulation.
5. remaining steps are pretty much as described earlier. However, in this case, the number of releases is 'set' to equal the number of releases in the **MARK** analysis. You can change this, if you like. The default link function is left at '**identity**', and the default β parameters are left at the default value of 0.5.

Alternatively, you can

1. open up **MARK**, and select '**Simulations | Setup Simulations**', and then '**override**' the popup window.

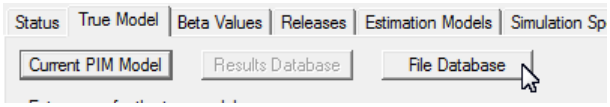
2. enter the details for the analysis you're interested in simulating (e.g., if you want to run one or more models from an analysis you did of a data set consisting of live encounters, 7 encounter occasions, an 2 groups, then you need to enter this data specification).
3. Once you've finished step (3), and clicked '**OK**', this will spawn the 'specification of simulations' window introduced earlier. Simply click the '**true model**' tab.
4. Look for a button labeled '**File database**', and click it. This will let you browse to a particular `.dbf` file for some other **MARK** project, select it, and then pick the model you want to simulate from those contained in the `.dbf` file (note, you can only pick one model at a time)
5. remaining steps are pretty much as described earlier. However, in this case, the link function, the beta parameters (for the link function), but not the number of releases is 'set' to equal the values from the **MARK** analysis. Again, you can manually override whatever you like.

We'll demonstrate the second approach using the full Dipper data set. Recall that the Dipper data consists of live encounter data, 7 occasions, and 2 groups (males and females). We'll assume we've already analyzed these data (say the results are contained in `ED.DBF`), fitting 4 different models. The 4 models are shown in the following browser:



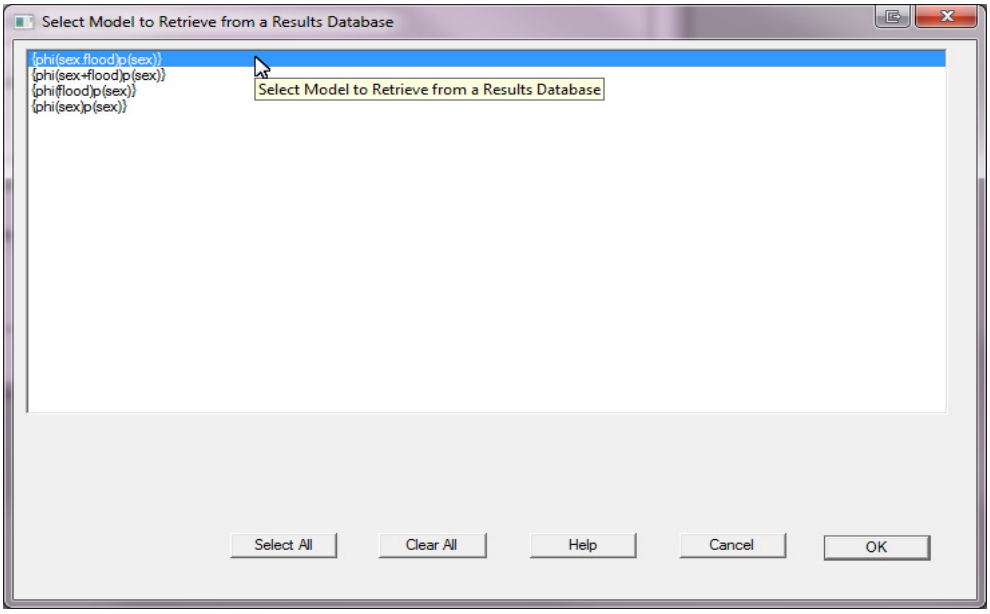
Model	AICc	Delta AICc	AICc Weight	Model Likelihood	No. Par.	Deviance
{phi(flood)p(sex)}	667.5899	0.0000	0.65438	1.0000	4	77.0178
{phi(sex+flood)p(sex)}	669.6322	2.0423	0.23569	0.3602	5	77.0122
{phi(sex.flood)p(sex)}	671.6377	4.0478	0.08647	0.1321	6	76.9601
{phi(sex)p(sex)}	674.2468	6.6569	0.02346	0.0359	4	83.6747

Suppose we want to simulate data under the model $\{\varphi_{sex.flood}p_{sex}\}$. From above, all we need to do is start **MARK**, and select '**Simulations | Setup Simulations**', and then '**override**' the popup window. Next, we click the appropriate data type ('**Recaptures only**'), and specify 7 occasions, and 2 groups (it isn't necessary to label the groups, or to make the group labels equivalent to the labels used in the original Dipper analysis). Then, in the '**Specification of Simulations**' window, select the '**True Model**' tab. Click the '**File Database**' button



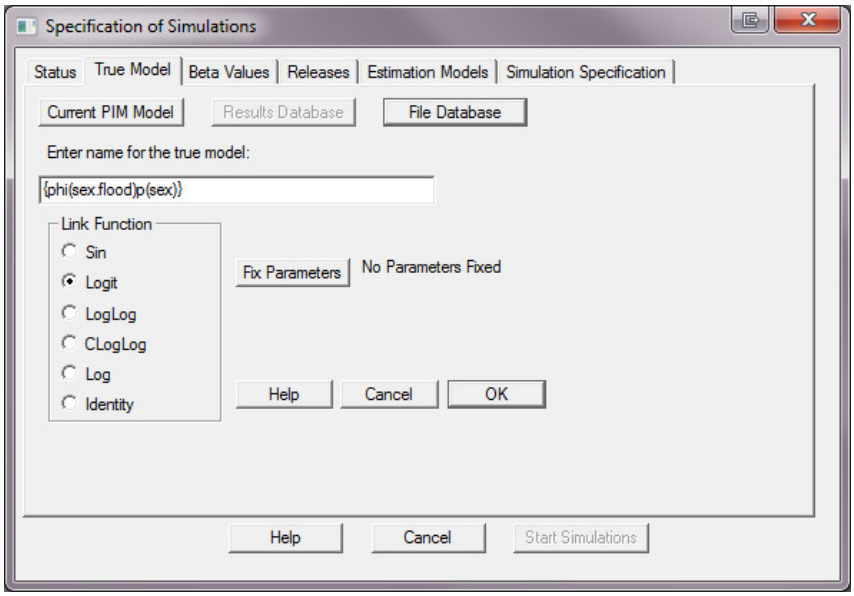
and browser to '`ED.DBF`' (i.e., the `.dbf` file containing the results of your earlier Dipper analysis).

Once you've selected `ED.DBF`, you will be presented with a list of all 4 of the models contained in the results database for the Dipper analysis (shown at the top of the next page). Simply highlight/select the model you want to simulate from the list (as shown below, for model $\{\varphi_{sex.flood}p_{sex}\}$):



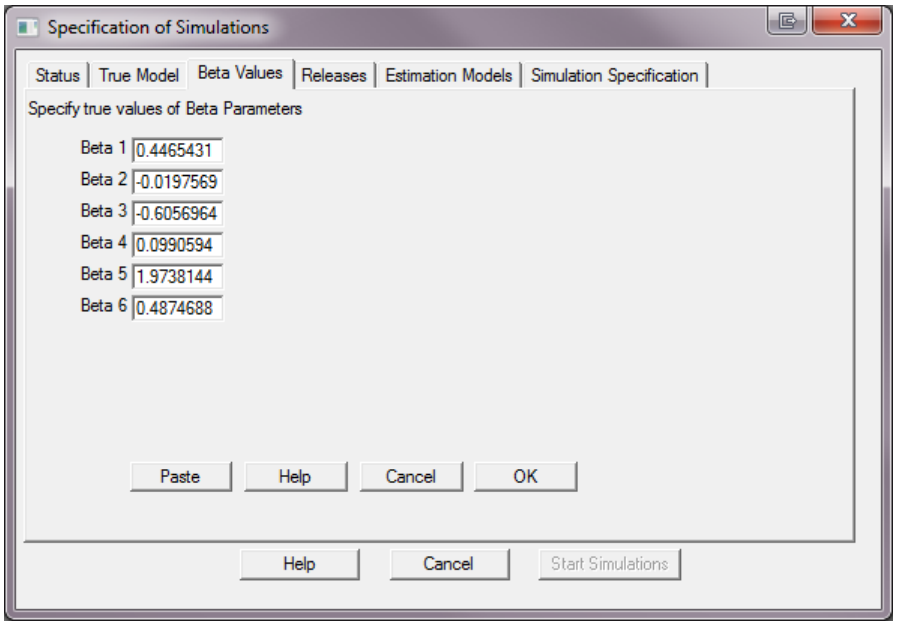
Note: don't click the '**Select all**' option. Trying to bring all the files models into the **MARK** simulation tool at the same time will cause **MARK** to crash. Retrieve one file model at a time.

Once you retrieve the file model, **MARK** will drop you back into the '**Specification of Simulations**' window. You'll see (below) that the name for the true model has been set (reflecting the name of the file model you just retrieved). It is also important to note that **MARK** has also set the link function to whatever link function you used for that model in `ED.DBF` (in this case, the logit link).

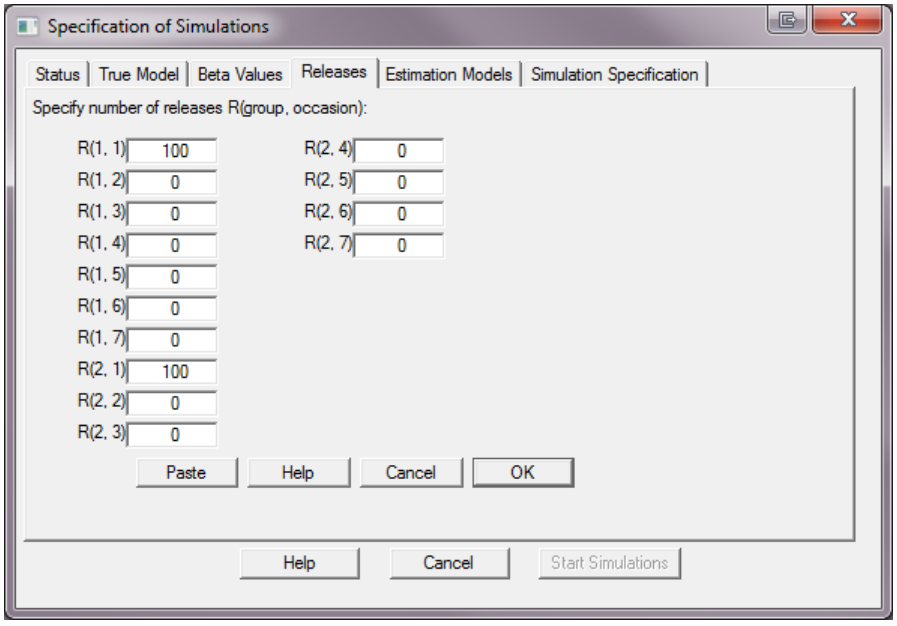


Click the '**OK**' button.

Next, if you click the '**`Beta values`**' tab, you'll see that **MARK** has also 'retrieved' the beta value estimates from the file model:



However, as noted earlier, **MARK** does not retrieve the original number of releases from the file model:



In summary, it is relatively easy to 'retrieve' a previous file structure into the simulation capability of **MARK**. Simply use a method which is most convenient for your purposes.

A.4. Simulation of robust design + closed capture data – special considerations

Simulation of the robust design model (Chapter 15) requires that the user enter parameters for the simulation that are interpreted differently than what is estimated. This issue occurs for the population estimates for the second and later primary sessions. The user would expect that the values entered for the population sizes for each primary session would be the actual population sizes. This assumption is true for the first primary session. However, for the second and later primary sessions, the value of N entered is the number of new animals entering the population at that point, i.e., N is now the number of animals available for capture *that have never previously been available for capture*. This rather strange arrangement is caused because of the way that the robust design models are structured (see Chapter 15).

For the closed captures models (see Chapter 14) in the robust design where N is in the likelihood, the values entered for the N parameters are always assumed to have the identity link. Thus, regardless of what link function is specified for the true model, the values entered for the N parameters are the values for N. For the Huggins closed captures models (where N does not appear in the likelihood), N values are entered in a separate tab window labeled as the true population size. However, again, the actual values entered are either the initial population size (first primary session) or else the number of new animals entering the population (second and later primary sessions).

A.5. Summary

The simulation capability of **MARK** is a very powerful and effective way to look at the impacts of number of occasions, sample size, number of releases, and other factors, on the strength of your inference, before you actually conduct your study. Conducting a study involving marked individuals requires careful planning, and the simulation tool in **MARK** is an effective first step.