A graphical user interface (GUI) input-based algorithm to automate generation of multi-state models for release-recapture studies

Adam C. Pope

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington
2014

Committee:
John R. Skalski
Timothy E. Essington

Program Authorized to Offer Degree:
Quantitative Ecology and Resource Management

University of Washington

Abstract

A graphical user interface (GUI) input-based algorithm to automate generation of multi-state models for release-recapture studies

Adam C. Pope

Chair of the Supervisory Committee:
Professor John R. Skalski
School of Aquatic and Fishery Sciences

Release-recapture studies represent an important branch of population analysis and are the primary method used to investigate population survival and migration. As researchers seek to answer more detailed questions about the populations under study, and as the study designs themselves become increasingly complex, the models necessary to estimate survival and migration parameters must increase in complexity as well. Multi-state models are widely used for this purpose since they can accommodate multi-dimensional study designs and have a flexible parameterization. Model specification of a multi-state model for a complex release-recapture study design is far from trivial, and requires a thorough statistical understanding of these types of models. Additionally, multi-state models require specification of all possible capture histories for the study design in question. This task can be daunting when done manually as possible capture histories for some release-recapture studies can number in the tens of thousands.

This thesis describes the creation and implementation of a computer program (Program BRANCH) capable of multi-state model specification and parameter estimation for complex release-recapture studies. The program is divided into three main elements: (1) Allow the user to draw a study design diagram on the screen as input, specifying releases and recapture opportunities and the structure of survival and migration routes through the study, (2) translate the study design diagram into a multi-state model specified by a product-multinomial conditional likelihood equation, and (3) provide estimates and standard errors for biologically meaningful migration and survival parameters from study data. The construction of Program BRANCH is developed from digraph and maximum likelihood theory and is illustrated using five diverse release-recapture studies as test cases.

Acknowledgments

I would like, first and foremost, to acknowledge my wife Amanda. Her emotional support, commitment, and understanding were essential to the completion of this research. I am unable to express the depth of gratitude her unwavering encouragement has inspired. Thanks honey!

John Skalski served as my primary advisor and Thesis Committee Chair. His counsel was consistently invaluable and excellent, and his guidance enabled me to complete the daunting task of writing this thesis. To him and to my second committee member Tim Essington, who both provided their time and advice without reservation, I owe a huge debt of thanks. Additionally, Rebecca Buchanan, Richard Townsend, Cindy Helfrich and the rest of the staff at University of Washington's Columbia Basin Research freely and consistently gave of their time and experience to answer the many questions I frequently needed help with. I will miss working with all of them and am grateful for their support.

I would also like to thank my fellow graduate students in the Quantitative Ecology and Resource Management program. Academic research should always include the free exchange of ideas as a central tenet, and the brilliance of my fellow students often illuminated my own path to completion of this thesis.

This work was supported by Bonneville Power Administration, Project 1989-107-00, Contract No. 267893.

Table of Contents

List of Figures

Figure Number Page

List of Tables

Table Number Page

Chapter 1: Introduction

Estimation of population survival from one time period to the next or, in the case of anadromous fish, along a particular stretch of a river is of high interest to investigators. A popular method to obtain these estimates is the release-recapture study (Cormack 1964, Jolly 1965, Seber 1965), in which animals are captured, marked and released. This capture process is then repeated, with marked (recaptured) individuals noted. The subsequent recaptures enable estimation of population parameters such as abundance, survival, and capture probability.

1.1 Cormack-Jolly-Seber models

Cormack (1964), Jolly (1965), and Seber (1965) have developed and refined a multinomial likelihood model now commonly used to estimate these population parameters for release-recapture studies. This Cormack-Jolly-Seber (CJS) model has been extensively used in salmonid studies in the Pacific Northwest and elsewhere (Brownie et. al. 1993, Lebreton and Pradel 2002, Perry et. al. 2010). A CJS model can be represented as a diagram (Figure 1-1). In the hypothetical release-recapture study represented in Figure 1-1, individually identifiably tagged fish are released (circled R), travel downstream and are either detected or not at each of two downstream recapture opportunities. These recapture locations can be physical structures at which fish are collected and tags identified, or they can be antenna arrays which pick up a signal from the acoustic- or radio-tagged fish as they pass by. Because these recapture opportunities usually miss some passing tagged fish, they cannot be assumed to detect tagged fish with probability 1. Hence the probability of detection at a recapture location can only be estimated separately from the probability of having survived to that location by making use of additional data collected at subsequent recapture opportunities (Jolly, 1965).

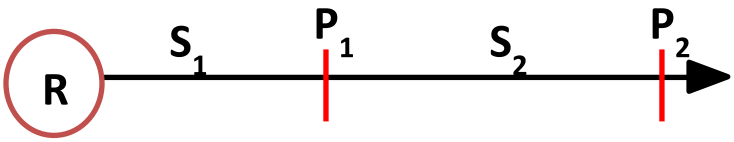


Figure 1-1. Diagrammatic representation of a Cormack-Jolly-Seber (CJS) model. S_i represents the probability of surviving through reach i; P_j represents the probability of detection at gate j.

The multinomial likelihood is constructed from each tagged fish's multi-Bernoulli detection history. For example, in Figure 1-1 an individual fish could either be detected at both gates (detection history 11), neither gate (00), the first gate but not the second (10), or the second gate but not the first (01). Each possible detection history is assigned a probability, with all possible histories' probabilities summing to 1. Again as an example, detection history 11 has probability $S_1P_1S_2P_2$. As mentioned above, however, S_2 and P_2 are not separately estimable, and so the joint probability of surviving reach 2 and being detected at gate 2 is represented by $\lambda = S_2P_2$. The number of estimable parameters is verified by noting that this k=4 celled multinomial model has k-1=3 minimum sufficient statistics and therefore 3 estimable parameters.

1.2 Multi-state models

A CJS model allows for recaptures over time or a single migration path through space. Darroch (1961) generalized the Lincoln-Petersen estimator of population size to apply to stratified populations. Arnason (1972, 1973) later developed a model for which animals in a release-recapture study with three recapture opportunities were allowed to migrate between strata between each successive recapture opportunity; Schwarz et. al. (1993) developed matrix notation for a likelihood-based approach to the Arnason model, extending its applicability to a generalized *k* recapture opportunities. This Arnason-Schwarz model is thus a natural extension of the CJS model (Schwarz and Arnason 2000).

Consider as an example a release-recapture study in which a single site is surveyed on 2 occasions after release of the marked animals. The CJS model for this study considers a detection history for each marked animal which is of the same form as in the example from the section above, i.e. a two-digit, binary detection history for each marked animal. An Arnason-Schwarz model of this study would allow for a finite number of locations, for example three, at each survey, with the possibility of tagged animals moving among locations between surveys. The detection history for each animal is still two-digit but is no longer binary. For example, an animal not detected during the first survey and detected at site 2 in the second survey would have capture history 02.

The Arnason-Schwarz model allows for separate transition parameters for animals migrating between different states within a survey period, denoted as ϕ_{ijk} to indicate that an animal transitioned from state i to state j in period k. Note that because these transition parameters depend only on the state of the animal in adjacent survey periods, the model is Markovian. In other words, the detection history in periods 1,...,k-1 is not considered in estimation of ϕ_{ijk}. The transition parameter ϕ_{ijk} may be thought of as the product of the probability of survival from state i to state j over period k with the probability of movement from state i to state j over period k, conditional on survival. Generally the survival and conditional movement parameters are not separately estimable.

This Arnason-Schwarz model became the basis for further generalizations, forming the class of product-multinomial likelihood models known as multi-state models (Lebreton and Pradel, 2002). For example, Brownie et. al. (1993) and Hestbeck et. al. (1991) have both extended the Arnason-Schwarz model to relax the Markovian movement assumption in cases when previous history may affect future movement patterns. In evolutionary biology, reproductive costs and life history strategies are important considerations, and multi-state models have been adapted to consider these factors as well (Nichols et. al. 1994, Clobert 1995). Bayesian methods, specifically Markov Chain Monte Carlo, have been employed when treating the multi-state model as a Markov chain with missing data (Dupuis 1995). In all approaches, marked animals are assumed to act independently, as the likelihood approach assumes a multinomial distribution.

As with CJS models, multi-state models may be represented diagrammatically (Figure 1-2). Although there are only 2 branching points, the multi-state model representing Figure 1-2 is much more complex than the CJS model of Figure 1-1. While calculation of the number of estimable parameters for this product-multinomial model is not as straightforward as in the CJS multinomial example above, there are 7 minimum sufficient statistics and therefore 7 estimable parameters for this model. But there are 5 detection probabilities, 7 survival probabilities, and 2 probabilities of movement conditional on survival for a total of 14 parameters in the model. Sorting out which parameters are estimable in these branching, multi-state models is therefore an issue of interest for field researchers.

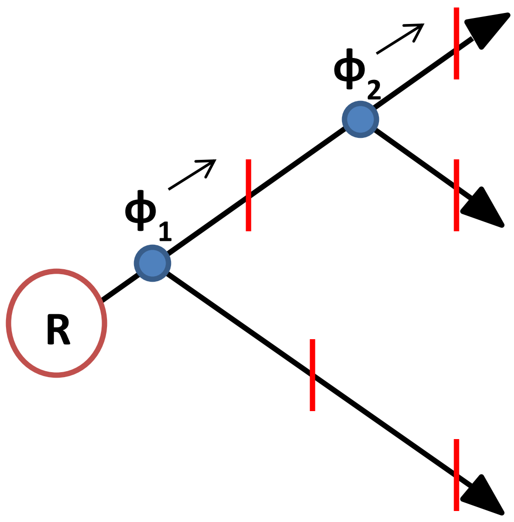


Figure 1-2. Diagrammatic representation of a simple multi-state model. As in Figure 1-1, each reach has probability of survival S_i and each gate probability of detection P_j, but here these are suppressed for clarity. ϕ_k represents the probability of entrainment into the topmost branch at fork k, and so the probability of entrainment into the bottom branch is $1-\phi_k$.

The method most commonly employed to develop a model for a given salmonid release-recapture study is to develop a schematic of the river system being studied with release points and detection gates indicated in a manner similar to Figures 1-1 and 1-2. The researcher then determines which parameters are estimable, writes the probability for each possible detection history, and assembles the data for each fish into numbers of fish showing each of those capture histories. This process can be made somewhat simpler by the use of conditional likelihood models (Brownie 1993), but is still labor intensive and increasingly complex as the complexity of river systems under study increases. I present here a methodology to generalize and automate the process of formulating branching or multi-state release-recapture models, given a graphical representation of the river system.

1.3 Motivation

Release-recapture studies modeled by CJS and multi-state models are used extensively to estimate population abundance, survival and migration. The use of CJS and multi-state models has expanded to include research in fisheries science as well as terrestrial wildlife biology. Skalski et. al. (2002) used a multi-state model to analyze survival and route-specific migration of juvenile Chinook salmon (*Oncorhynchus tshawytscha*) through a hydroelectric dam. A similar model was used by Perry et. al. (2010) for juvenile Chinook salmon survival and migration in the Sacramento-San Joaquin River Delta, where the various routes in the model were representative of Delta distributaries and cross-channels rather than routes through a dam. Buchanan and Skalski (2010) examine adult Chinook salmon migrating upstream from the Lower Columbia and Snake Rivers into tributary systems within the Columbia Basin using a multi-state model.

Avian studies have used release-recapture methods for decades (e.g. Cormack 1964), and multi-state models are often used to estimate survival and movement parameters for these studies. Doherty et.al (2004) used a multi-state model to estimate age-specific breeding and survival probabilities for red-tailed tropicbirds (*Phaethon rubricauda*). Natal dispersal, breeding dispersal, and age-specific recruitment rates for roseate terns (*Sterna dougallii*) were estimated using multi-state models by Lebreton et. al. (2003). Hestbeck et. al. (1991) analyzed movement between and fidelity to wintering areas among Canada geese (*Branta canadensis*) using a multi-state model. Multi-state models are useful to investigators in mammalian wildlife biology as well. Release-recapture methodology was used by Aars and Ims (1999) to explore the effects of habitat corridors on populations of root voles (*Microtus oeconomus*). More recently, a multi-state model was used by Pyne et. al. (2010) to estimate survival rates stratified by age, sex, and breeding status for a reintroduced population of plains bison (*Bison bison*). Given the wide use of multi-state models, any increase in the speed and ease with which the models can be used for release-recapture studies should be of value to investigators.

This breadth and range of studies that rely on the multi-state model for parameter estimation leads to a case-specific modeling process, in which the model structure for a particular study design must be specified in order to write the likelihood equations used to estimate survival and migration parameters. While there are several computing tools available to find maximum likelihood estimates once the model structure and likelihood are specified (Choquet et. al. 2004, White and Burnham 1999), the process of creating the CJS or multi-state model structure itself is generally done by hand. Moreover, any change to the study design, for example relocation, addition, or removal of a recapture opportunities, requires rewriting the entire likelihood equation. Automating this process should free valuable time and resources to be used for other aspects of these studies.

Additionally, researchers are attempting to scale these studies up to include multiple years or larger geographic areas in order to answer larger, ecosystem level questions. As the complexity of the areas under study grow, the details of modeling the data take increasing amounts of time and resources. Automation of model creation may in fact become necessary as increasingly complex systems make the current method infeasible (Fujiwara and Caswell 2002). Research that attempts to find commonalities, patterns, and rules to the structure of these models may also drive the ability to further increase the complexity and scope of release-recapture studies themselves.

What is missing from the available software packages used to estimate parameters for multi-state models is a user-friendly, intuitive way to specify the model type. Both M-SURGE (Choquet et. al. 2004) and MARK (White and Burnham 1999) allow specification of parameter resrictions, number of recapture opportunities and number of model states through typed input. While this approach does allow a great deal of flexibility in model specification, some level of statistical training is necessary to use these packages. Additionally, this format does not easily specify a model such as that in Figure 1-2, a realization of the Arnason-Schwartz model in which the possible transitions between states is greatly reduced. It seems much easier to specify this type of model by actually drawing the diagram as represented in Figure 1-2. A graphical user interface (GUI) input in a statistical analysis program would allow this capability and to my knowledge does not currently exist. While GUI are widely used in other fields such as electrical engineering (www.circuitlogix.com) and flowchart design (creately.com), I have not been able to find an example of this type of GUI paired with release-recapture statistical model design software.

This research is just as pertinent to issues in study design. Release-recapture studies whose design is not carefully thought out beforehand can often yield data which do not answer the questions intended by the study. Questions such as where to locate traps or antennas and when to survey should be at the forefront of the study design from the beginning, and are also increasingly difficult to answer in a way that maximizes the value of the data collected as these studies become increasingly complex. A software tool which updates in real time which parameters are estimable as researchers manipulate and fine tune user-created schematics of study designs will not only save time and resources, but will also yield studies that are more efficient and that answer the researcher's intended questions much more frequently.

1.4 Objectives

My first objective is to develop an algorithm to translate a graphical schematic such as the ones in Figures 1-1 and 1-2 into a tabular or matrix format to be read as computer input in a graphical user interface (GUI). I propose that any such schematic can be represented as a node and edge directed graph, and as such the information contained in the schematic is representable as a matrix. My second objective is to develop an algorithm to write a multi-state likelihood model from any such matrix derived from the associated schematic. I will rely on existing statistical theory, conditional likelihoods and minimum sufficient statistical theory to accomplish this, developing the theory further as necessary. Finally, I will test the software's effectiveness and versatility by using it to estimate parameters for different types of release-recapture studies, using data from actual studies to verify results.

Chapter 2: Developing Program BRANCH

2.1 Introduction

Release-recapture studies represent an important source of demographic information for wild populations (Seber 1982). Building models for these studies in order to make statistical inferences about underlying parameters associated with animal populations has become increasingly complex as the scope and study design of release-recapture studies has increased in complexity, as investigators seek to answer increasingly sophisticated and subtle questions about the animal populations being studied, and as computational power has grown exponentially (Lebreton and Pradel 2002). In a release-recapture study, individually identifiable tagged animals each display a capture history with multiple recapture opportunities. Often animals can take one of several possible pathways from one recapture opportunity to the next, or there may be a reason for the investigator to partition transitions between recapture opportunities in an attempt to study aspects of population survival or migratory behaviors. Multi-state models are a natural and capable choice for analysis of these studies, since as the name implies, animals are assumed to exist in one of multiple possible states at each recapture opportunity and can move between states during transitions from one recapture opportunity to the next in a manner restricted only by model construction. Thus the use of the multi-state model in survival and movement parameter estimation of release-recapture studies is widespread (Schwartz et. al. 1993, Hestbeck et. al 1991).

One drawback of these models is that their breadth and flexibility comes with added mathematical complexity (Lebreton et. al. 1992). An investigator without specialized statistical training may have a difficulty correctly developing and fitting a multistate model for a release-recapture study. Currently, no statistical software exists to assist investigators with the design and analysis of multi-state release-recapture studies. However, there are structural features of multistate models which suggest a generalized, automated approach to analysis of these models may be possible. One such feature is the association of a single transition parameter with each possible path from one recapture opportunity and/or model state to another. Another such feature is the directional nature of these models, as animals must move through either time or space unidirectionally for such a model to be useful. By linking the development of the model likelihood to the study design schematic, it should be possible to write a set of rules to automate this process. As studies seek to answer questions concerning increasingly complex population and community ecology level systems, an intuitive, user-friendly software tool to develop and fit data to multistate models would be useful.

2.2 Algorithm development

Ideally, a procedure that begins with a user-specified diagram or schematic of the system under study and ends with a likelihood model and estimable parameters would make these multi-state techniques more accessible to investigators. The development of this capability will require a number of analytical and computational steps.

First, there must be a way for the user to draw a schematic of the study design. A graphical user interface (GUI) allowing point-and click and mouse drag operations can achieve this task. The user can select among a series of icons to place on a canvas, and can drag from one placed icon to another to indicate adjacency and direction. Second, the software must interpret the types of icons and their adjacency relationships into a meaningful format for machine processing. Once the diagram has been interpreted, the third step is to completely enumerate and store each possible path a tagged animal can take through the study as well as the model parameters associated with survival, movement and detection. The fourth step is to write a product-multinomial likelihood equation using the parameters and spatial and temporal relationships specified in the diagram. Fifth, the program must decide what parameters of the model as specified are estimable, and which parameters are not separable and must be combined. Sixth, the program must be able to determine whether any additional re-parameterization is necessary given the data from the study, which is also input by the user. Finally, after estimation of the user-specified model parameters, the program should be capable of summarizing survival or movement processes across the entire system being modeled. This chapter will provide details of each of these developmental steps used in creating a graphical interface based multi-state release-recapture estimation algorithm.

2.2.1 Representation of a study design schematic

As an example, I present a diagram of a release-recapture study (Figure 2-1) for a simple branching process with arrows denoting the direction of movement of tagged juvenile anadromous fish (downstream). The fish are released and travel downstream to the first branching point (point F_1, Figure 2-1). Each fish reaching this point either moves into the upper or lower branch and continues downstream. A similar process occurs among those fish in the upper branch at point F_2. Detection arrays along the river comprise potential recapture locations and are represented in the diagram by vertical lines crossing the river channel (points G_i, $i=1,\ldots,7$, Figure 2-1). Each fish passing a detection array can either be detected or not detected at that array. The set of release points, branching points, and detection points bound the set of line segments representing individual river reaches and denoted in Figure 2-1 by L_1, …, L_9. Each fish can experience mortality at any point in its migration.

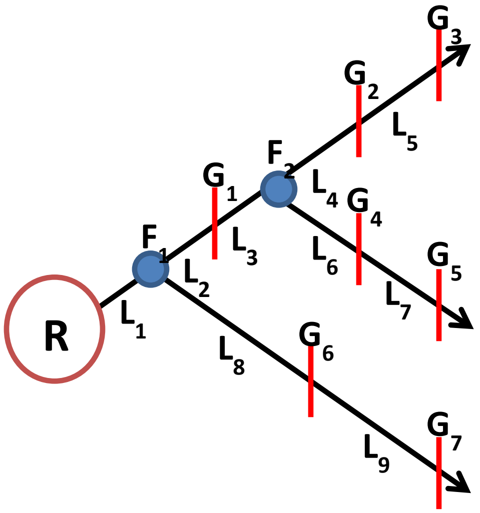


Figure 2-1. Diagrammatic representation of a hypothetical release-recapture study. Point R represents release point, L_i represent river reaches, F_j are two forks in the river, and G_k are recapture opportunities/detection arrays. Arrows indicate direction of movement.

If we define parameter γ_j, j=1,2 as the probability of entrainment into the topmost branch at branching point F_j, then the probability of entrainment into the lower branch at point F_j becomes $(1-\gamma_j)$. We can define parameter P_i as the probability of detection (recapture) at point G_i. The probability of a fish passing point G_i undetected is then clearly $(1-P_i)$. We define the probability of mortality at a point in the interval defined by L_k as $(1-S_k)$. Since the set L_k is a partition of the set of all points in the river, we have completely defined mortality in the delta, with survival along reach L_k defined by S_k. By combining the diagram in Figure 2-1 with the parameters defined above, we have a diagram representing the multi-state model for this release-recapture scenario (Figure 2-2). The estimability of these parameters will be considered later in this chapter; for now it is simply noted that not all of these parameters are separately estimable.

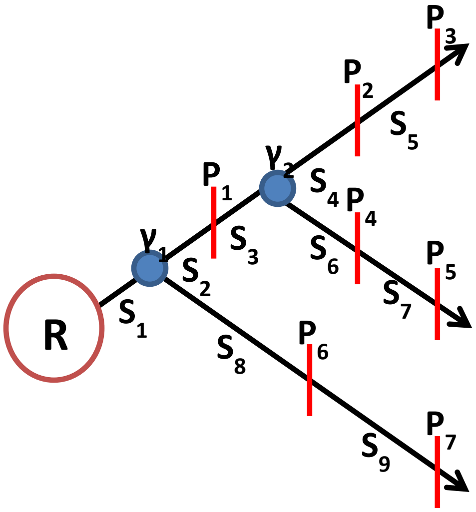
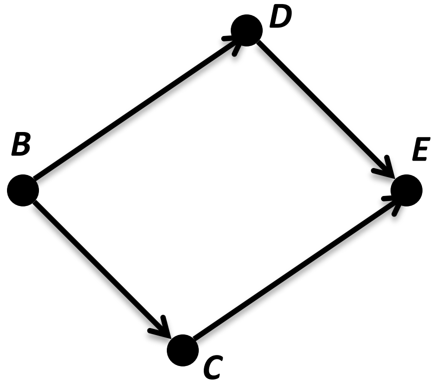


Figure 2-2. Multi-state model of the hypothetical release-recapture study in Figure 2-1. Not all parameters are estimable.

2.2.2 Diagram interpretation using digraph theory

In order to create an algorithm to generate a likelihood equation from a diagram representation such as the one in Figure 2-2, the first step is to be able to represent the information in the diagram in such a way that a machine can read and manipulate that information. Digraph theory presents one way to accomplish this (Bang-Jensen 2007). A digraph or directed graph consists of a set of vertices and a set of arcs, sometimes denoted *V* and *A* respectively. The vertices in *V* are simply the elements of the set *V*. The arcs in the set *A* are ordered pairs of vertices. The information in a digraph can equivalently be represented in either graphical format or by enumerating the sets *V* and *A* (Figure 2-3). The direction of the arcs is represented by an arrow in the graphical format and by the order of the pair of vertices in the list format.



$V=\{B,C,D,E\}$

$A=\{(B,C),(B,D),(C,E),(D,E)\}$

	B	C	D	E
B	0	1	1	0
C	0	0	0	1
D	0	0	0	1
E	0	0	0	0

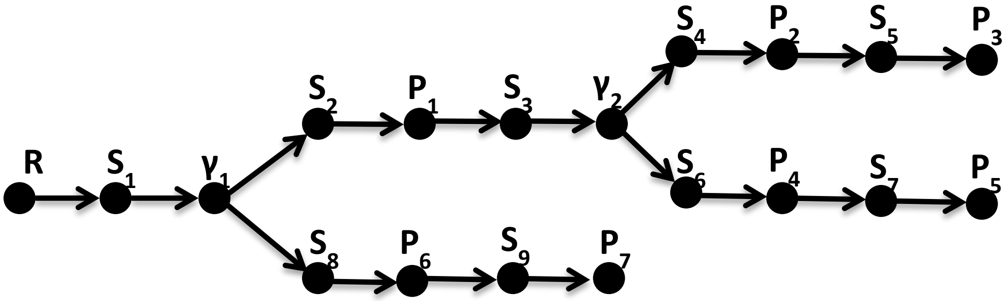
Figure 2-3. Example of a simple directed graph (digraph), with vertices comprising the set *V* and arcs comprising the set *A*. Matrix adjacency representation of the digraph is shown by the table.

Introducing some more terminology from digraph theory, the two vertices in an ordered pair representing an arc (e.g. (B,C) in Figure 2-3) are the end-vertices of that arc; any two such end-vertices of an arc in the set *A* are said to be adjacent (Bang-Jensen 2007). Although the vertices B and E are both elements of the set *V* in Figure 2-3, B and E are not adjacent because there is no arc (B,E) or (E,B) in the set *A*. The graph shown in Figure 2-3 is a connected graph, that is, by moving only between adjacent vertices it is possible to visit all of the vertices in the set *V* starting from any of the vertices in the set. It is also an acyclic graph, meaning that by tracing a path along directed arcs from one vertex to another, it is impossible to cycle through any vertex more than once. We will only be concerned with connected, acyclic digraphs.

Imagine now that the n vertices in the set *V* of a digraph are arbitrarily labelled $v_1,\ldots,v_n$. The adjacency matrix representation $M=[m_{ij}]$ of the digraph is an n x n matrix such that $m_{ij}=1$ if the arc (v_i,v_j) is an element of the set *A* and $m_{ij}=0$ otherwise (Bang-Jensen 2007). For example, the adjacency matrix representation of the digraph from the example above is 4 x 4 and is shown in Figure 2-3. The graphical representation, matrix adjacency representation, and enumeration of the elements of the sets *V* and *A* of a digraph each contain all of the information in the digraph and are equivalent and interchangeable (Bang-Jensen 2007).

The adjacency matrix representation is useful as a way to input the information from a schematic such as that in Figure 2-2 into an algorithm. Each release point, river branch, detection array, and river reach in Figure 2-2 has a single parameter associated with it. We can represent the diagram in Figure 2-2 as a digraph with vertices consisting of the parameters associated with the diagram, and arcs consisting of the adjacencies between these parameters (Figure 2-4).

Indeed, by treating each parameter as a vertex and the directed adjacencies between parameters as arcs, it is possible to construct a digraph from any such diagrammatic representation of a release-recapture study. This method of representation has several advantages. First, by equating the set *V* of vertices to the set of parameters arising from the model, we maintain focus on that which we wish to estimate. Second, the diagrammatic representation of the digraph relates to the spatial and/or temporal nature of the multistate model. Finally, the ability to represent the information in either schematic or matrix form facilitates the creation of a graphical user interface (GUI) which accepts a user-specified diagram and converts it into a matrix of adjacencies which can be further manipulated to create a written multinomial likelihood.



	R	S_1	γ_1	S_2	P_1	S_3	γ_2	S_4	P_2	S_5	P_3	S_6	P_4	S_7	P_5	S_8	P_6	S_9	P_7
R		1																	
S_1			1																
γ_1				1												1			
S_2					1														
P_1						1													
S_3							1												
γ_2								1				1							
S_4									1										
P_2										1									
S_5											1								
P_3																			
S_6													1						
P_4														1					
S_7															1				
P_5																			
S_8																	1		
P_6																		1	
S_9																			1
P_7																			

Figure 2-4. Digraph representation of the multi-state model from Figure 2-2. Table depicts the matrix adjacency representation of the digraph.

2.2.3 Traversing the digraph using recursive programming

In the example digraph above the adjacency matrix representation has rows v_i and columns v_j where i,j=1,…,n and n is the number of vertices in the digraph. The order of the rows and columns is unimportant; the vertices are labeled with an arbitrary index i which could have been applied in any order so long as it is consistent. Thus the row and column name for each vertex can be the parameter from the multistate model associated with that vertex (Figure 2-4). This naming scheme carries the information about the types of parameters into the adjacency matrix. Note that the column R in the matrix adjacency representation in Figure 2-4 has no entries equal to 1. This is because R is the release parameter and there are no arcs leading into the release point, only an arc leading out. Similarly, the rows P_3, P_5, and P_7 have no entries equal to 1 because they are associated with detection arrays with no subsequent opportunity for recapture, and thus have no arcs leading out from them in the digraph diagram.

A method for listing the adjacent parameters along each possible path through the model is then to start at each release point (parameters whose columns have no 1 entries) and find the 1 entries in that row. The columns of these entries are the next parameters along the path(s). If there is exactly one such column, that column becomes the next row in which to find 1 entries, and the process iterates. If there is more than one such column, a recursive procedure must be employed. Produce a number of copies of the parameter path-list matching the number of columns with 1's. Pick an arbitrary column from these as a place holder to return to, then recursively call the entire procedure with each of the other columns as start points. The list(s) returned from this recursive call are each appended to a copy of the existing path-list, then the procedure is continued with the original copy from the place-holder. When performing an iteration finds a row with no 1 entries, the procedure ends, returning the generated list(s) one nested level out in the recursive process. If the procedure is already in the outermost level, the returned lists will enumerate the successive parameters encountered along each possible path through the model.

2.2.4 Creating conditional likelihoods

Given the model schematic from Figure 2-2, a likelihood equation for the study design can be written. One approach is to construct a multinomial likelihood based on all possible migration pathways and recapture opportunities. This process creates a detection history for each fish which can be written as an n-digit binary number, where n is the number of recapture opportunities in the model, and each digit d_i (i=1,…,n) equals 1 if a fish is recaptured at detection array i and 0 otherwise. The multinomial likelihood equation is then created by writing a probability statement for each possible detection history which is then exponentiated by the number of fish displaying that detection history. A likelihood equation for the example model in Figure 2-2 is given here:

$$\begin{aligned}
L(S,\gamma,P) \propto {} & \left[S_1\gamma_1S_2P_1(1-S_3+S_3(\gamma_2(1-S_4+S_4(1-P_2)(1-S_5+S_5(1-P_3)))+(1-\gamma_2)(1-S_6+S_6(1-P_4)(1-S_7+S_7(1-P_5)))))\right]^{a_1} \\
& \times\left[S_1\gamma_1S_2P_1S_3\gamma_2S_4P_2(1-S_5+S_5(1-P_3))\right]^{a_{12}}\times\left[S_1\gamma_1S_2P_1S_3\gamma_2S_4(1-P_2)S_5P_3\right]^{a_{13}}\times\left[S_1\gamma_1S_2P_1S_3\gamma_2S_4P_2S_5P_3\right]^{a_{123}} \\
& \times\left[S_1\gamma_1S_2P_1S_3(1-\gamma_2)S_6P_4(1-S_7+S_7(1-P_5))\right]^{a_{14}}\times\left[S_1\gamma_1S_2P_1S_3(1-\gamma_2)S_6(1-P_4)S_7P_5\right]^{a_{15}}\times\left[S_1\gamma_1S_2P_1S_3(1-\gamma_2)S_6P_4S_7P_5\right]^{a_{145}} \\
& \times\left[S_1\gamma_1S_2(1-P_1)S_3\gamma_2S_4P_2(1-S_5+S_5(1-P_3))\right]^{a_2}\times\left[S_1\gamma_1S_2(1-P_1)S_3\gamma_2S_4(1-P_2)S_5P_3\right]^{a_3}\times\left[S_1\gamma_1S_2(1-P_1)S_3\gamma_2S_4P_2S_5P_3\right]^{a_{23}} \\
& \times\left[S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6P_4(1-S_7+S_7(1-P_5))\right]^{a_4}\times\left[S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6(1-P_4)S_7P_5\right]^{a_5}\times\left[S_1(1-\gamma_1)S_8P_6S_9P_7\right]^{a_{67}} \\
& \times\left[S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6P_4S_7P_5\right]^{a_{45}}\times\left[S_1(1-\gamma_1)S_8P_6(1-S_9+S_9(1-P_7))\right]^{a_6}\times\left[S_1(1-\gamma_1)S_8(1-P_6)S_9P_7\right]^{a_7} \\
& \times\begin{bmatrix}1-S_1+S_1(\gamma_1(1-S_2+S_2(1-P_1)(1-S_3+S_3(\gamma_2(1-S_4+S_4(1-P_2)(1-S_5+S_5(1-P_3))) \\ +(1-\gamma_2)(1-S_6+S_6(1-P_4)(1-S_7+S_7(1-P_5))))))+(1-\gamma_1)(1-S_8+S_8(1-P_6)(1-S_9+S_9(1-P_7))))\end{bmatrix}^{a_0}
\end{aligned} \tag{1}$$

In equation (1) above parameters S_i, P_k, γ_1 and γ_2 are the same as in Figure 2-2. Exponents a_j denote the number of marked animals with detection history j, where the digits of j are the numbered detection arrays at which detection occurs. Equation (1) is over-parameterized and not all of the parameters will be separately estimable. Re-parameterization of equation (1) to allow estimation will be discussed later in this chapter.

Alternatively, to develop an algorithm to write a likelihood equation from the (matrix format) information in a multistate model it would seem better to break the detection histories into single steps, and the probability statements into statements about successive detections. Rewriting the likelihood equation in a conditional format allows exactly that. Probability theory states that a joint probability $\Pr(X_0=x_0,X_1=x_1,X_2=x_2,\ldots,X_n=x_n)$ is equivalent to the conditional probability $\Pr(X_0=x_0)\Pr(X_1=x_1|X_0=x_0)\Pr(X_2=x_2|X_1=x_1)\ldots\Pr(X_n=x_n|X_{n-1}=x_{n-1})$. For the joint probability statement representing each complete detection history, let X_0 be the probability of release (which equals 1 for marked and released animals in the study). Then the joint probability can be represented as the probability of being first detected at array i_1, given release, times the probability of being next detected at array d_2, given detection at d_1, etc., times the probability of not being detected again, given detection at array d_m. Here d_i ($i=1,\ldots,m$) are the detection arrays at which a fish is detected and so exhibits a 1 in the binary detection history described above. The last detection, and so last 1 in the detection history, is d_m. The conditional product-multinomial likelihood equation equivalent to equation (1) is given here:

$$
\begin{aligned}
L(S,\gamma,P) \propto & \left\{\begin{array}{l}
\left[S_1\gamma_1S_2P_1\right]^{a_{1|R}} \times\left[S_1\gamma_1S_2(1-P_1)S_3\gamma_2S_4P_2\right]^{a_{2|R}} \times\left[S_1\gamma_1S_2(1-P_1)S_3\gamma_2S_4(1-P_2)S_5P_3\right]^{a_{3|R}} \times\left[S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6P_4\right]^{a_{4|R}} \\
\times\left[S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6(1-P_4)S_7P_5\right]^{a_{5|R}} \times\left[S_1(1-\gamma_1)S_8P_6\right]^{a_{6|R}} \times\left[S_1(1-\gamma_1)S_8(1-P_6)S_9P_7\right]^{a_{7|R}} \\
\times\left[\begin{array}{l}1-S_1+S_1(\gamma_1(1-S_2+S_2(1-P_1)(1-S_3+S_3(\gamma_2(1-S_4+S_4(1-P_2)(1-S_5+S_5(1-P_3))) \\
+(1-\gamma_2)(1-S_6+S_6(1-P_4)(1-S_7+S_7(1-P_5))))))+(1-\gamma_1)(1-S_8+S_8(1-P_6)(1-S_9+S_9(1-P_7))))\end{array}\right]^{a_{0|R}}
\end{array}\right\} \\
& \times\left\{\begin{array}{l}
\left[S_3\gamma_2S_4P_2\right]^{a_{2|1}} \times\left[S_3\gamma_2S_4(1-P_2)S_5P_3\right]^{a_{3|1}} \times\left[S_3(1-\gamma_2)S_6P_4\right]^{a_{4|1}} \times\left[S_3(1-\gamma_2)S_6(1-P_4)S_7P_5\right]^{a_{5|1}} \\
\times\left[1-S_3+S_3(\gamma_2(1-S_4+S_4(1-P_2)(1-S_5+S_5(1-P_3)))+(1-\gamma_2)(1-S_6+S_6(1-P_4)(1-S_7+S_7(1-P_5))))\right]^{a_{0|1}}
\end{array}\right\} \\
& \times\left\{\left[S_5P_3\right]^{a_{3|2}}\left[1-S_5+S_5(1-P_3)\right]^{a_{0|2}}\right\} \times\left\{\left[S_7P_5\right]^{a_{5|4}}\left[1-S_7+S_7(1-P_5)\right]^{a_{0|4}}\right\} \times\left\{\left[S_9P_7\right]^{a_{7|6}}\left[1-S_9+S_9(1-P_7)\right]^{a_{0|6}}\right\}
\end{aligned}
\qquad (2)
$$

Parameters in equation (2) are the same as in equation (1) except for the exponents $a_{j|i}$, which here are the number of fish detected at array j given detection at array i. Equation (2), like equation (1), is over-parameterized.

Writing a conditional product-multinomial likelihood equation such as in equation (2) can be accomplished by following a simple set of rules. The input to the algorithm is the matrix adjacency representation of the digraph format model schematic. Following the recursive algorithm above gives a list of parameters along each possible path through the model. To write the probability of detection at site j given detection at site i (where i=0 is the release point), identify and store all possible paths from site i to site j from the list of total possible paths. Each of these sub-paths is essentially atomic, that is the probability of reaching j from i via each sub-path can be expressed simply as the product of the parameters along the sub-path, with 1-P substituted for each parameter P in between i and j indicating non-detection at these intermediate sites. The total probability of detection at j given detection at i is the sum of these sub-path products.

For an example from Figure 2-2, consider the probability of a tagged fish being first detected at detection array P_4 given release at point R. First note there is only one possible pathway from point R to point P_4; that is, by taking the topmost branch at the first branching point (probability γ_1) and the lower branch from the second branching point (probability $1-\gamma_2$). A fish must have taken this path, and also must not have been detected at point P_1. Thus the probability we are interested in, constructed as outlined in the previous paragraph, is $\Pr(P_4|R)=S_1\gamma_1S_2(1-P_1)S_3(1-\gamma_2)S_6P_4$.

To write the probability of not being detected again, given detection at site i (i=0 indicates release site as above), the possibility of mortality must also be included. First, identify and store all paths including site i. To express the probability of surviving after detection at site i but not being detected again, write the product of the parameters along each sub-path starting at i and continuing to the end of the model, substituting 1-P for all P parameters to indicate non-detection; take the sum of these products as the probability. To express the probability of mortality after detection at site i, for each path beginning at site i, create a set of sub-paths ending with a S parameter. Find the product of the parameters along each sub-path substituting 1-P for P parameters as before, and substituting 1-S for the final S parameter of each sub path. The sum of these products is the probability of mortality.

Again using Figure 2-2 as an example, consider the probability of a fish not being detected again, given detection at point P_1. The fish could either have died in reach S_k, where k={3,4,5,6,7}, or could have survived and not have been detected again on its migration through the study area. The probability Pr(not seen again|P_1) is the sum of seven independent probabilities and can be expressed as Pr(not seen again|P_1) = $(1-S_3) + S_3\gamma_1(1-S_4) + S_3\gamma_1S_4(1-P_2)(1-S_5) + S_3(1-\gamma_1)(1-S_6) + S_3(1-\gamma_1)S_6(1-P_4)(1-S_7) + S_3\gamma_1S_4(1-P_2)S_5(1-P_3) + S_3(1-\gamma_1)S_6(1-P_4)S_7(1-P_5)$.

Although the example used throughout is a specific instance of a multistate model, the procedure outlined immediately above is general enough to apply to any multistate release-recapture product-multinomial model.

2.2.5 Re-parameterization and estimability

Unfortunately not all of the parameters in equation (2) are separately estimable. A feature of CJS and by extension multistate models is that in order to separately estimate both the probability of surviving to a recapture opportunity and the probability of being recaptured at that opportunity, there must be an additional opportunity for recapture later on (temporally and/or spatially). This attribute of the models constrains parameter estimation at the last recapture opportunity; only the joint probability of migrating and surviving from the penultimate to the last recapture opportunity, and of being recaptured there, is estimable (Jolly 1965, Brownie 1993). In the example model from Figure 2-2 there is no further opportunity for detection after migration past arrays P_3, P_5, or P_7. Thus the model must be re-parameterized with parameter λ_i defined as such a joint probability; here $\lambda_1 = S_5 P_3$, $\lambda_2 = S_7 P_5$, and $\lambda_3 = S_9 P_7$ (Figure 2-5).

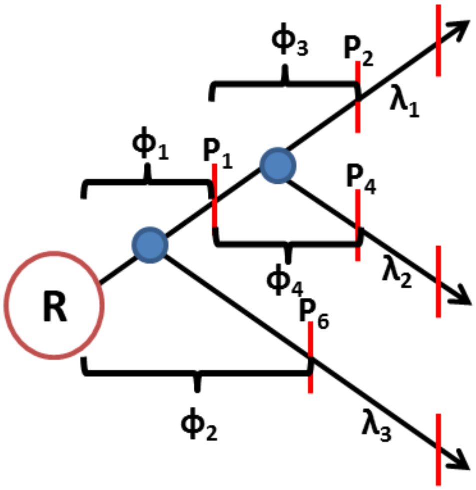


Figure 2-5. Diagram of re-parameterized multi-state model based on estimable parameters from the hypothetical release-recapture study in Figure 2-1.

An additional class of parameters is not estimable in the multistate model example we have constructed. These are parameters which always appear together wherever they are found in the likelihood equation. The four such collections of parameters are $S_1\gamma_1S_2$, $S_1(1-\gamma_1)S_8$, $S_3\gamma_2S_4$, and $S_3(1-\gamma_2)S_6$. Comparing these terms with the spatial arrangement of the model schematic in Figure 2-2, it becomes apparent that parameters accrue in this manner when no recapture opportunity exists between adjacent parameters. Another way to state this is that only one parameter may be estimated between any two recapture opportunities. This result is an intrinsic feature of both multi-state and CJS models (Brownie 1993, Jolly 1965).

We may, however, take advantage of the structure of the model and algebraically solve for certain inseparable parameters by making additional assumptions when appropriate. For example, in the model example there are the terms $S_1\gamma_1S_2$ and $S_1(1-\gamma_1)S_8$. If we make the assumption that the probability of survival through the left river branch downstream of point γ_1 to the detection array P_1 is equal to the probability of survival through the right river branch downstream of point γ_1 to the detection array P_6, then $S_2=S_8$. Since we can estimate $\phi_1 = S_1\gamma_1S_2$ and we can estimate $\phi_2 = S_1(1-\gamma_1)S_8$, this assumption lets us solve for γ_1 and S_1S_2 (which equals S_1S_8). Whether the assumption is reasonable will depend on how homogeneous is the survival process across different river reaches. Consequently, study-specific conditions may permit assumptions that allow further estimability of parameters on a case by case basis.

Wherever this type of assumption is reasonable, the route entrainment probability γ may be solved for algebraically and estimated separately from the survival probability S adjacent to it. Elsewhere we are only able to estimate a single parameter between recapture probabilities. In the case of the example model from Figure 2-2, we make no assumptions concerning individual reach survival parameters, and we introduce the parameter ϕ_i as the joint probability of migrating from a release point or detection array to the next detection array along a marked animal's path, and surviving this migration. Specifically, we define $\phi_i=S_1\gamma_1S_2$, $\phi_2=S_1(1-\gamma_1)S_8$, $\phi_3=S_3\gamma_2S_4$, and $\phi_4=S_3(1-\gamma_2)S_6$. This re-parameterization gives a model for which all parameters are estimable (Figure 2-5). Substituting this re-parameterization into the conditional product-multinomial likelihood equation (2) gives the following:

$$\begin{aligned}
L(S,\phi,P,\lambda) \propto & \begin{Bmatrix} \left[\phi_1\right]^{a_{1|R}} \times \left[\phi_1(1-P_1)\phi_3P_2\right]^{a_{2|R}} \times \left[\phi_1(1-P_1)\phi_3(1-P_2)\lambda_1\right]^{a_{3|R}} \\ \times\left[\phi_1(1-P_1)\phi_4P_4\right]^{a_{4|R}} \times \left[\phi_1(1-P_1)\phi_4(1-P_4)\lambda_2\right]^{a_{5|R}} \times \left[\phi_2P_6\right]^{a_{6|R}} \times \left[\phi_2(1-P_6)\lambda_3\right]^{a_{7|R}} \\ \times\left[1-\phi_1-\phi_2+\phi_1(1-P_1)(1-\phi_3-\phi_4+\phi_3(1-P_2)(1-\lambda_1)+\phi_4(1-P_4)(1-\lambda_2))+\phi_2(1-P_6)(1-\lambda_3)\right]^{a_{0|R}} \end{Bmatrix} \\
& \times \begin{Bmatrix} \left[\phi_3P_2\right]^{a_{2|1}} \times \left[\phi_3(1-P_2)\lambda_1\right]^{a_{3|1}} \times \left[\phi_4P_4\right]^{a_{4|1}} \times \left[\phi_4(1-P_4)\lambda_2\right]^{a_{5|1}} \\ \times\left[1-\phi_3-\phi_4+\phi_3(1-P_2)(1-\lambda_1)+\phi_4(1-P_4)(1-\lambda_2)\right]^{a_{0|1}} \end{Bmatrix} \\
& \times \left\{\left[\lambda_1\right]^{a_{3|2}}\left[1-\lambda_1\right]^{a_{0|2}}\right\} \times \left\{\left[\lambda_2\right]^{a_{5|4}}\left[1-\lambda_2\right]^{a_{0|4}}\right\} \times \left\{\left[\lambda_3\right]^{a_{7|6}}\left[1-\lambda_3\right]^{a_{0|6}}\right\}
\end{aligned} \tag{3}$$

Expanding the probability statements associated with no further detection given release and no further detection given detection at P_1 shows that the probability statements within each separate multinomial likelihood (separated by curly brackets) sum to 1. Because the likelihood equation is a product-multinomial, each set of curly brackets encloses a separate multinomial likelihood, which are all then multiplied together to form the product-multinomial. As such, each separate multinomial equation must sum to 1. This constraint can be used as a check by the algorithm to ensure that the equation has been written correctly. This exercise is a useful check as to whether the likelihood equation has been written correctly, and is included as part of the likelihood generating algorithm.

2.2.6 Data and sufficient statistics

The data necessary for estimation of the model parameters from the final likelihood equation are the counts $a_{j|i}$, which are the numbers of tagged animals detected at each recapture opportunity j given their detection at recapture opportunity i (for complete specification of the conditional likelihood equation i=0 can represent the release point). For general applications the count data should be in in column format and cross-specified by tag identification, location of recapture, and time of recapture for every recapture event. As long as the recapture location names in the data match those in the diagram, it is a simple matter for the algorithm to compute the needed $a_{j|i}$ statistics. As an example, consider a release-recapture study with six opportunities for recapture after release and three possible states at each recapture. Given the data as specified, it is simple to consider a six-digit detection history for each animal with each digit either 0, 1, 2, or 3 indicating either non-detection or detection in one of the three states at each of the six recapture opportunities (e.g. 010223). From this detection history a conditional detection history is constructed: detection in state 1 at site 2 given release, detection in state 2 at site 4 given detection at site 2, detection in state 2 at site 5 given detection at site 4, detection in state 3 at site 6 given detection at site 5, and not detected again given detection at site 6. Each of these conditional detections adds a unit to the count $a_{j|i}$ as defined above.

It is possible for additional model re-parameterization to be necessary based on features of the data. As an example, if no fish were detected at P_6 (Figure 2-2), then the counts $a_{7|6}$ and $a_{0|6}$ from equation (4) will both be 0 and the parameter $\lambda_3=S_9P_7$ is not estimable. In this case the rule would dictate eliminating $\lambda_3=S_9P_7$ from the model, leading to a re-parameterization of the model where the parameter $\phi_2P_6=S_1(1-\gamma_1)S_8P_6$ becomes λ_3. This must occur because as there are no detections at P_7 it is as if this detection array does not exist, making $P_6$the last recapture opportunity along this route.

2.2.7 System level parameter estimation

The estimates provided by the algorithm as described so far coincide as closely as possible with the parameters defined by initial specification of the model diagram. The only place re-parameterization has occurred has been out of necessity, to avoid over-parameterization or inestimability caused by sparse data. However, individual reach survival and detection probabilities such as those in Figure 2-2 are likely to be of limited interest. Instead, release-recapture studies are usually designed to estimate system-wide or population-wide parameters such as overall survival or route-specific survival from release through the end of the study area.

While no algorithm can predict with certainty, given the level of generality available in design of the diagram input, what overall metrics will be of interest to an investigator, it can make an attempt to provide estimates of certain metrics which may be of general interest. By defining system- or sub-system-wide metrics as functions of parameters (multiplication along a route, addition between separate routes) and using the delta method to estimate standard errors of these functions, it is possible that some biologically meaningful metrics may be automatically recovered. However, care must be taken in interpreting the meaning of these metrics. For example, the program may incorporate a rule to automatically estimate the probability of transition along each route from release to penultimate detection array on that route (because the last reach parameter estimate includes the probability of detection at the last array, typically these last reach probabilities are not included in study-wide survival metrics). In our example, the three route transition probabilities $\phi_1\phi_3$, $\phi_1\phi_4$, and ϕ_2 would be estimated. Note however that these are not survival probabilities since they include the probabilities of entrainment along that route.

In general it is difficult for an algorithm to predict consistently what will be of interest to an investigator. Instead, a user should be able to specify combinations of parameters which are of interest, and the program to compute these parameter combinations based on the invariance property of maximum likelihood estimation.

2.3 Algorithm Construction

Having outlined the theory necessary for each step of algorithm development, construction of the computer code based on these steps is relatively straightforward. Steps 1 and 2 take the algorithm from a diagrammatic representation of the study design to a matrix representation of the parameters that arise from that design in a multistate model context. The computer code which implements these steps creates a graphical user interface (GUI) allowing the user to create a study design diagram on the computer screen. The QTSDK application developed by Nokia and utilizing the C++ programming language (qt-project.org) was the programming platform used for these steps, and the written code is attached in Appendix A. Steps 3 through 7 start with the matrix adjacency representation and create a conditional product-multinomial likelihood in which all parameters are estimable and calculable from the data, as well as provide some study-wide level parameters as functions of the model parameters. The R programming language (http://cran.us.r-project.org/) was used to implement these steps with written code attached as Appendix B. Finally, the algorithm must actually provide parameter estimates and standard errors for the model as written. The program USER (http://www.cbr.washington.edu/analysis/apps/user) was utilized for this task. A diagram showing the breakdown of tasks by computer program is shown (Figure 2-6).

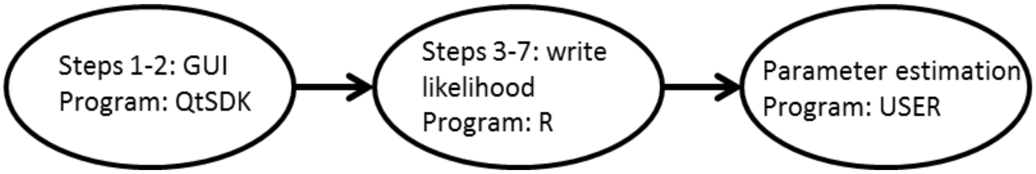


Figure 2-6. Algorithm construction using three different applications to implement steps 1-7 and parameter estimation as outlined in Algorithm Development sections.

2.4 Other types of multistate models

Thus far a single example has been used to illustrate the development of the likelihood-generating algorithm as clearly and simply as possible. However, release-recapture studies are not limited to studies of fishes migrating through a delta or tributary system. The format is used for a wide ranging variety of systems, from bird banding studies to terrestrial mammal migration and population estimation to cohort analysis. The algorithm framework described above must be general enough to accommodate this wide range of study designs.

The aforementioned cohort analysis provides a good example of a multistate model that looks quite different from the example used previously. Anadromous fish such as Chinook salmon (*Oncorhynchus tshawytscha*) can migrate seaward as juveniles in their first (age-0) or early second (age-1) year of life, and can return from the ocean at anywhere from age-3 to age-6. A release-recapture study might release tagged juvenile Chinook with several fixed detection arrays as the recapture opportunities at various points along the single river route from release to the ocean. If questions of interest center around differences in migration or survival depending on the age at which these fish choose to migrate either downstream as juveniles or upstream as adults, a multistate model is appropriate (Figure 2-7). Here the full meaning of the word multistate becomes readily apparent, as each migrating fish is assigned to a state depending on its age at migration. Figure 2-7 represents a model schematic analogous to Figure 2-2. Each horizontal line represents a state. Each tagged animal can exist in only one state at a time. In this example the states are temporal. Vertical lines represent recapture opportunities, in this case arranged in space. There are opportunities in the model for each fish to transition from one state to another. If a fish passes a detection array during migration, then waits a year before passing the next detection array, it has transitioned to the next state. Clearly no transition is possible from an older fish to a younger fish; as time is unidirectional, so must the transition possibilities be.

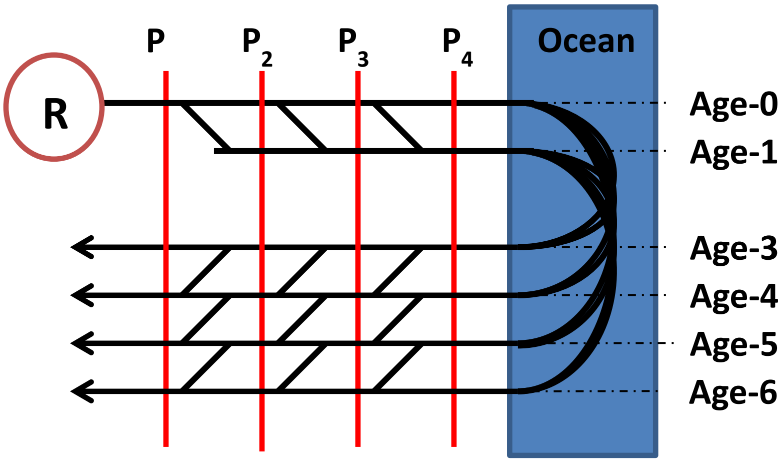


Figure 2-7. Multi-state model of a hypothetical cohort analysis release-recapture study.

Another example illustrates the versatility of the multistate model. A population of migratory birds are sampled, individually marked, and periodically surveyed for their location. If the birds prefer to spend summer in one location and winter in another, the timing of the migration from one location to another can be estimated by a multistate model. Here the horizontal lines are still the possible states and the vertical lines the recapture opportunities (Figure 2-8). In this example, however, the states are spatial, and the recapture opportunities temporal. Also, rather than only moving from state to state in one direction as in the cohort analysis example above, here birds are free to migrate back and forth between states.

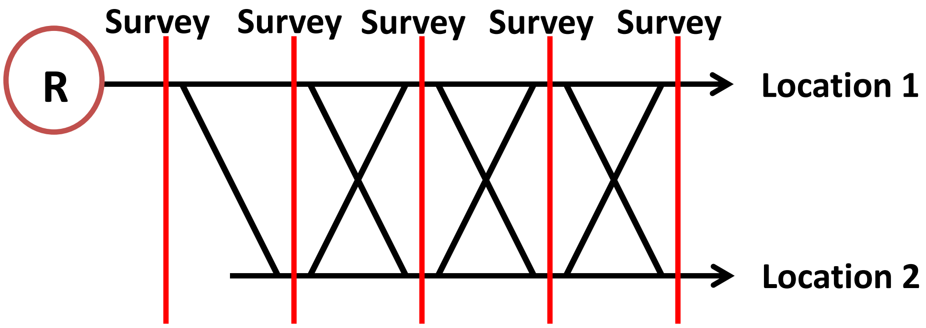


Figure 2-8. Multi-state model to estimate bird migration parameters for a hypothetical release-recapture study.

Although there is a wide array of release-recapture study designs which can be represented with multistate models, there are a few guidelines for the investigator useful in creating a study design diagram such as the ones in Figures 2-2, 2-7, and 2-8, a step necessary as input for the likelihood equation generating algorithm. The first step in creating such a diagram should be to decide whether the study design can be represented strictly in terms of a simplified schematic of the study area, such as is the case in Figure 2-2. In this case tagged animals migrate past fixed detection arrays in a single direction, with the possibility of doubling back past a detection array either nonexistent or made so by filtering of the data. Additionally, the timing of the animals' migration is not of interest as it relates to estimation of survival and movement parameters (such a study may include a separate analysis of travel times which is not of concern here).

If a simplified map of the study area cannot be easily transformed into a model diagram as in Figure 2-2, there are still relatively simple steps which should allow representation of the study design as a model schematic acceptable for input into the BRANCH algorithm. The first of these steps is to determine in what dimension the recapture opportunities are fixed, space or time. In the example represented in Figure 2-7, migrating salmon can be detected over multiple years as they migrate seaward and back upriver as adults, but the detection arrays remain fixed spatially. Thus the vertical lines representing the recapture opportunities in this case each comprise a fixed location. In the example from Figure 2-8, the converse is true. A bird can be recaptured in one of various locations, but the surveys occur on fixed dates, and so here recapture opportunities are fixed in time.

The determination of recapture opportunities should facilitate the next step in model construction. In a multi-state model one must of course determine the possible states in which animals can exist. This now becomes apparent based on the way in which recapture opportunities from the previous step are arranged. The essential question to be answered here is: In what possible configurations can animals be recaptured (or fail to be recaptured) at any of the recapture opportunities? For example, in Figure 2-7 a fish may migrate past a fixed detection array in one of several possible years. Of course, any partition of time is possible here, but the investigator is interested in brood year analysis, which involves differential migration and survival patterns based on the year of migration. Again, conversely in the example from Figure 2-8, on a given survey date birds may be recaptured at one of several locations of interest to the investigator. Thus the states of the model are such that the sum of all model states encompasses the total number of ways an animal may be recaptured at a given opportunity; also these states should be partitioned in a way that is of interest to investigators.

Although in these examples the dimensions of space and time are interchangeable, each model has its restrictions and complexities which arise from possible state transitions, number of states, and number of recapture opportunities. Both models are multi-state models of release-recapture studies, however, as is the model from Figure 2-2. These commonalities should be enough to allow generation of a multinomial likelihood equation from program BRANCH. The following chapter will test this versatility using diverse examples of release-recapture studies as test cases for the program.

Chapter 3: Applying Program BRANCH

3.1 Introduction

In designing an automated algorithm to write multistate product-multinomial likelihood models, it is important to make the capability as flexible and simple as possible. To that end, generality is a desirable feature of such an algorithm. The multistate model is used to fit data from a wide array of mark-release-recapture studies (Leberton and Pradel 2002, Schwartz et. al. 1993), and such studies themselves can be set in a variety of contexts (Perry et. al. 2010, Schwartz et. al. 1993, Hestbeck et. al 1991). If an automated algorithm is designed properly, the commonalities of these models should allow for application to most, if not all, of these types of models. By testing the algorithm in a wide a range of scenarios, not only are its capabilities demonstrated, but it may become apparent whether there are implicit assumptions restricting the types of models it can generate. By finding where such assumptions have to be made one can determine the scope of the algorithm. This is important not only for scope of use, but to insure that the generalized rules of the algorithm arise directly and naturally from basic features common to all multistate models.

In order to demonstrate the capabilities of the BRANCH program, and to test whether unstated assumptions have been made in its construction which could unintentionally limit its use to only certain types of multistate models, four different datasets were analyzed using models created by the program. The intent was to choose the four studies in such a way as to capture as much diversity in model structure as possible. As a baseline, a single-state (i.e. Cormack-Jolly-Seber, CJS) model was constructed. Next, two different types of multistate models were compared: one in which the model states were defined by time, and another in which the model states are related to location. Finally, a branching river delta served as the setting for a study in which individual fish could migrate seaward using any of several pathways.

In the first analysis, data from a study of juvenile Chinook salmon (*Oncorhynchus tshawytscha*) in the Lower Columbia River (Skalski et. al. 2012) were fitted to a Cormack-Jolly-Seber (CJS) model, in which there is only a single possible state for all animals at each recapture occasion. Conceptualized this way, the CJS model is essentially a collapsed multistate model if transition and detection probabilities are pooled across states. Any rules governing the creation of a multistate multinomial likelihood model ought to apply to a CJS model as well. Additionally, the simplicity of the CJS model relative to the multistate models in the subsequent analyses is useful in establishing a base case to compare to the more complex multistate models.

The second analysis presented here used recapture data from Passive Integrated Transponder (PIT) tagged juvenile fall Chinook salmon released into the Chiwawa River in central Washington. These salmon migrated seaward from autumn of 2010 to spring of 2011 with the possibility of residualizing over the winter along the migration route. The data were fitted to a multistate model in which the year of migration between detection arrays composed the distinct model states. This type of multistate model is useful in estimating animal movement and survival over discreet time intervals, including cohort analysis for fish stocks (Burnham et. al. 1987). An important feature of these models is the restriction in movement between time-based states; animals migrating later in time of course cannot travel back to an earlier time, so movement between states occurs only in one direction.

Following the Chiwawa River cohort analysis is an analysis of data from a study of Lost River suckers (*Deltistes luxatus*) and shortnose suckers (*Chasmistes brevirostis*) in Clear Lake Reservoir in northern California. The multistate model to which these data were fitted is closely related to that from the previous example. However, here the states are physical locations (in the lake or in a tributary to the lake) while the recapture events were periodic aerial surveys. Thus the states vary over space and the intervals between detection events are temporal, whereas the Chiwawa River example states vary over time and intervals between detections are spatial. The important difference between the two types of models arises from the greater freedom of movement between location-based states. The possibility exists with the sucker analysis for the fish to move back and forth between lake and river states, rather than unidirectionally through temporal states as in the previous example.

Finally, an analysis was performed of data from a study of juvenile Chinook salmon in the San Joaquin-Sacramento River Delta (Delta). The multistate model to which these data were fit was constructed from a schematic representing some of the various migration routes through the Delta, where the various states were then these distinct routes. At first glance this type of model looks somewhat different from the other examples, as both states and intervals between recapture events are spatial. Although the example here is for outward migration of juvenile Chinook salmon, this type of branching model is also useful for basin-wide studies of upriver migrating adult anadromous fishes as they are detected among tributaries in a river basin.

3.2 Example 1 – CJS model for juvenile Chinook salmon in the Lower Columbia River

3.2.1 Introduction

Program BRANCH is designed to generate a multistate, product-multinomial likelihood in a first-step conditional format from a user-specified diagram of a release-recapture study. Any such multistate model reduces to a Cormack-Jolly-Seber (CJS) model when all states are pooled, e.g. when marked animals move unidirectionally in one dimension (time or space), modeling no movement between states and with only survival/migration probabilities and probabilities of detection as estimable parameters. In other words, there exists only a single state for all animals at each recapture occasion, so this provides a good opportunity for a first demonstration of Program BRANCH. Data were obtained from acoustic-tagged yearling Chinook salmon detections from a compliance and monitoring study of survival and passage at McNary Dam on the Lower Columbia River (Skalski et.al. 2012). The resulting survival and detection probability estimates were compared to estimates in the Pacific Northwest National Laboratory (PNNL) report (Skalski et. al. 2012).

3.2.2 Study site

McNary Dam is located at river kilometer (rkm) 470 on the Columbia River bordering Washington and Oregon. Acoustic detection arrays used for this analysis were located at the McNary Dam face, at rkm 422 between McNary Dam and John Day Dam, at the John Day Dam face (rkm 349), and at rkm 325 between John Day Dam and The Dalles Dam.

3.2.3 Methods

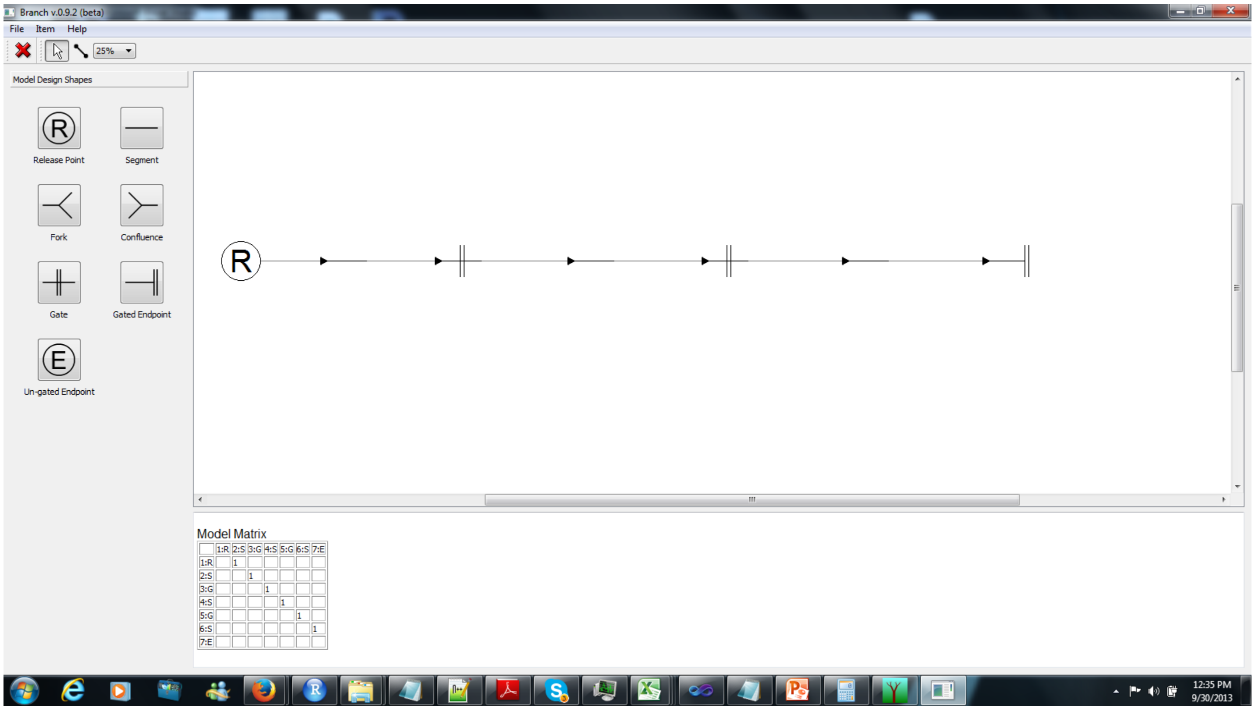
3.2.3.1 Tagging and data recovery

Tagging and data recovery methods are summarized here for yearling Chinook salmon and are more fully described in Skalski et.al. (2012) Yearling Chinook salmon were collected from the John Day Dam juvenile bypass system and surgically implanted with Advanced Telemetry Systems model SS300 acoustic tags. Tagged fish were released at rkm 503 upstream from McNary Dam. Release timing and location were designed in such a way as to facilitate downstream mixing of tagged fish, so that tagged yearling Chinook salmon arriving at McNary Dam face were distributed throughout the river and over the diel period similarly to yearling Chinook salmon from the run-at-large.

Detections of acoustic-tagged fish were identifiable by individual tag code, and were screened by PNNL staff to remove invalid detections (Skalski et. al. 2012). For this analysis, only tagged fish known to have arrived at McNary Dam face were considered. In this way the dam face constitutes a virtual release point for estimation of survival from the McNary Dam face to the John Day Dam face.

3.2.3.2 Data analysis

A CJS multinomial likelihood in first-step conditional format was created from a user specified diagram in Program BRANCH (Figure 3-1). The diagram illustrates three detection arrays downstream of the release point, resulting in two reaches in which survival is separately estimable, and a third, furthest downstream reach, in which the survival probability is not separable from the detection probability at the furthest downstream array. The input diagram shown in Figure 3-1 can also be represented as a model schematic with parameters marked and labeled (Figure 3-2a). Survival parameters are labeled S_1 for survival from the McNary Dam face to rkm 422, and S_2 for survival from rkm 422 to the John Day Dam face. Detection probabilities were labeled P_1 and P_2, for detection at rkm 422 and at John Day Dam respectively, and the joint probability of surviving the final reach and being detected at the final array at rkm 325 was labeled λ (Figure 3-2).



Model Matrix

	1:R	2:S	3:G	4:S	5:G	6:S	7:E
1:R		1					
2:S			1				
3:G				1			
4:S					1		
5:G						1	
6:S							1
7:E							

Figure 3-1. Screen capture of a user-specified diagram of a release-recapture study design of yearling Chinook salmon on the Lower Columbia River in 2012.

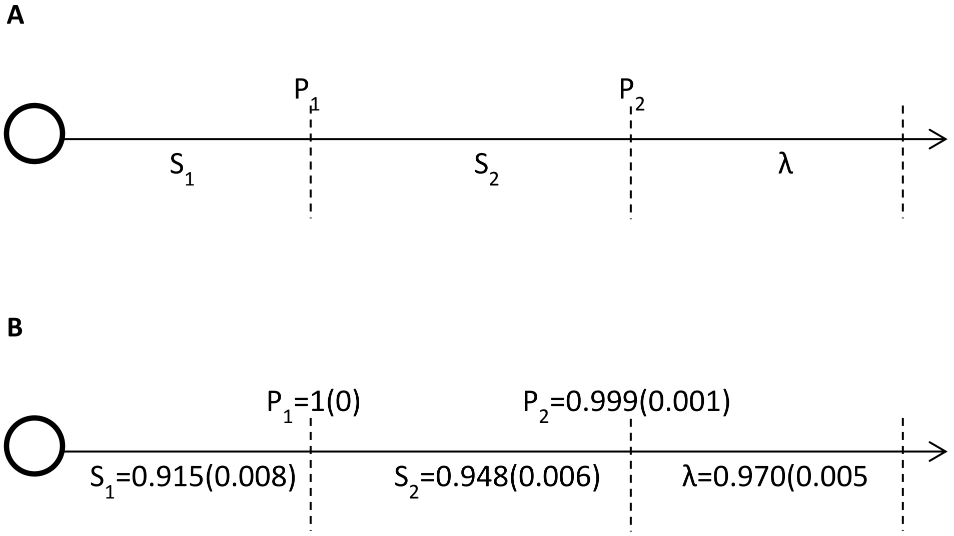


Figure 3-2. Multi-state model of a release-recapture study of yearling Chinook salmon on the Lower Columbia River in 2012. A) Survival parameters labeled S_1 and S_2, detection probabilities labeled P_1 and P_2, and joint probability of surviving the last reach and being detected at the last array labeled λ. B) Parameter estimates shown, with estimated standard errors in parentheses.

Once the single step, single-release recapture design was drawn, program BRANCH produced a product-multinomial likelihood equation which was then used to estimate survival for yearling Chinook salmon. The likelihood equation is of the form:

$$\begin{aligned} L(S_i,P_i,\lambda) \propto & \left\{ \left[S_1P_1\right]^{a_1} \times \left[S_1(1-P_1)S_2P_2\right]^{a_2} \times \left[S_1(1-P_1)S_2(1-P_2)\lambda\right]^{a_3} \times \left[1-S_1+S_1(1-P_1)\left(1-S_2+S_2(1-P_2)(1-\lambda)\right)\right]^{a_0} \right\} \\ & \times \left\{ \left[S_2P_2\right]^{a_2|a_1} \times \left[S_2(1-P_2)\lambda\right]^{a_3|a_1} \times \left[1-S_2+S_2(1-P_2)(1-\lambda)\right]^{a_0|a_1} \right\} \times \left\{ \left[\lambda\right]^{a_3|a_2} \times \left[1-\lambda\right]^{a_0|a_2} \right\} \end{aligned}$$

Parameter names in the equation follow terminology from the single-release recapture design schematic (Figure 3-2b). Statistics $a_{i|j}$ represent the number of identifiable tags detected at site j, given their detection at site i. Additionally, the program used the sufficient statistics of the data to determine that the detection probability at the first array (P_1) should be fixed at 1 (all fish detected at any array were also detected at rkm 422). Parameters were estimated via numeric methods in Program USER (http://www.cbr.washington.edu/analysis/apps/user, accessed 26 December 2013) from the likelihood thus specified, and an overall survival metric was recovered as a function of these estimated parameters using the Delta Method.

3.2.4 Results

Survival for yearling Chinook salmon from the virtual release point at the McNary Dam face to the detection array at the John Day Dam face was estimated at 0.868 ($\widehat{SE} =$ 0.009). This survival estimate is the product of the estimates for survival through the two reaches from virtual release to rkm 422 ($\widehat{S_1} =$ 0.915, $\widehat{SE} =$ 0.008) and from rkm 422 to the John Day Dam ($\widehat{S_2} =$ 0.948, $\widehat{SE} =$ 0.006). Probability of detection at rkm 422 was estimated at 0.999 ($\widehat{SE} =$ 0.001), while the joint probability of surviving the last reach and being detected at rkm 325 (λ) was estimated at 0.970 ($\widehat{SE} =$ 0.005; Figure 3-2b).

3.2.5 Discussion

Results obtained via Program BRANCH compare closely with those obtained by Skalski et. al. (2012) Slight differences between estimates are likely due to tag-failure probabilities incorporated into the PNNL study, as well as a single tag which was censored in the PNNL study after detection at the rkm 422 array. In order to use a true CJS model rather than a multistate model for this analysis neither tag failure probabilities nor the possibility of tag removal during the study was considered here. In conclusion, Program BRANCH was able to successfully generate a CJS model from the diagram in Figure 3-1 such that parameters were estimated in close alignment with independent estimates from Skalski et.al. (2012)

3.3 Example 2 – Multistate model for juvenile fall Chinook salmon 2010 cohort in the Chiwawa River basin

3.3.1 Introduction

Multistate multinomial models are conceptually related to the Cormack-Jolly-Seber (CJS) model seen in the previous analysis. In addition to the survival probabilities and probabilities of recapture estimated in a CJS model, the multistate model allows for probabilities of migration or non-migration from one recapture point to another. These migration or movement probabilities are not always separable from the survival probabilities. A transition probability in a multistate model is the joint probability of migrating from one recapture site to another and surviving the migration. In this way models can include various states, either in time, space, or both, among which animals migrate with varying restrictions depending on the specific model.

A relatively simple type of multistate model which builds on the CJS model is one in which movement is differentiated over time. Fish from the same brood year migrating down the same river over different years are treated as residing in different states. This type of model is particularly useful to investigators interested in cohort analysis of juvenile salmonids. Fall Chinook salmon, for example, hatched in the spring can migrate seaward in the fall as subyearlings or overwinter in streambeds and then migrate seaward the following spring as yearlings (Healey 1991). Because it is possible for these Chinook salmon to partially migrate downstream over varying distances, then residualize over the winter before continuing their migration the following spring (Rich 1920), a multistate model allowing transition from the first state (outmigration in fall of Year 1) to the second (outmigration in spring of Year 2) between any two detection arrays is useful in estimating survival parameters. In this analysis, data from detections of PIT-tagged juvenile Chinook salmon released into the Chiwawa River in central Washington State in 2010 and 2011 were fit to a multistate model created in program BRANCH.

3.3.2 Study site

The Wenatchee River is a tributary of the Columbia River in central Washington State (Figure 3-3). The Wenatchee joins the Columbia at river kilometer (rkm) 754, 284 km upstream of McNary Dam (rkm 470). The Chiwawa River is a tributary of the Wenatchee, joining it 77 km upstream of the Wenatchee confluence with the Columbia. Release sites for PIT-tagged juvenile Chinook salmon were along the Chiwawa River from 3 km to 46 km upstream from the confluence with the Wenatchee (www.ptagis.org, accessed Oct. 7, 2013). Detection sites used for this analysis include a single PIT-tag detection array 2 km upstream from the mouth of the Chiwawa River (CHL), a single PIT-tag detection array 50 km upstream from the mouth of the Wenatchee River (MWE), the PIT-tag detection system at the McNary Dam Juvenile Bypass Facility (MCJ), and several juvenile PIT-tag detection facilities at dams on the Columbia River downstream from McNary Dam which were pooled as a last detection array for outmigrating juvenile Chinook salmon (Figure 3-3).

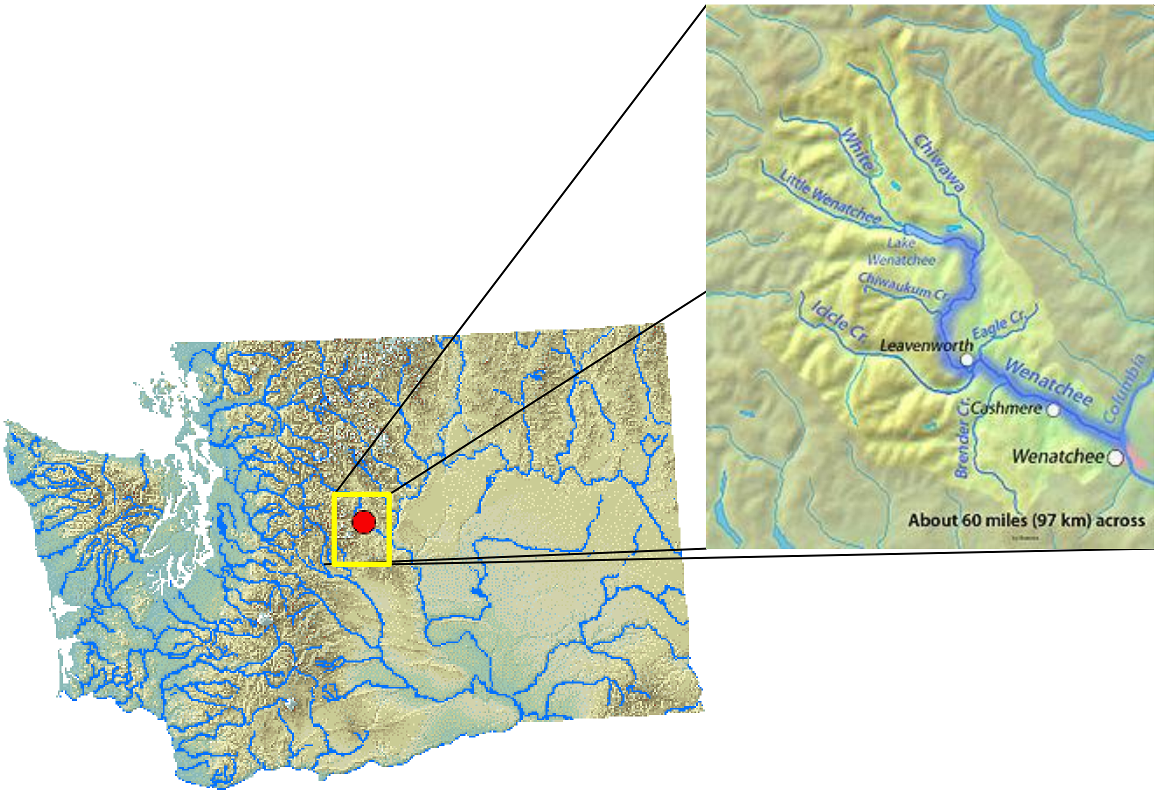


Figure 3-3. Map of the Wenatchee River basin including its tributary the Chiwawa River, with a portion of the Columbia River.

3.3.3 Methods

3.3.3.1 Tagging and data recovery

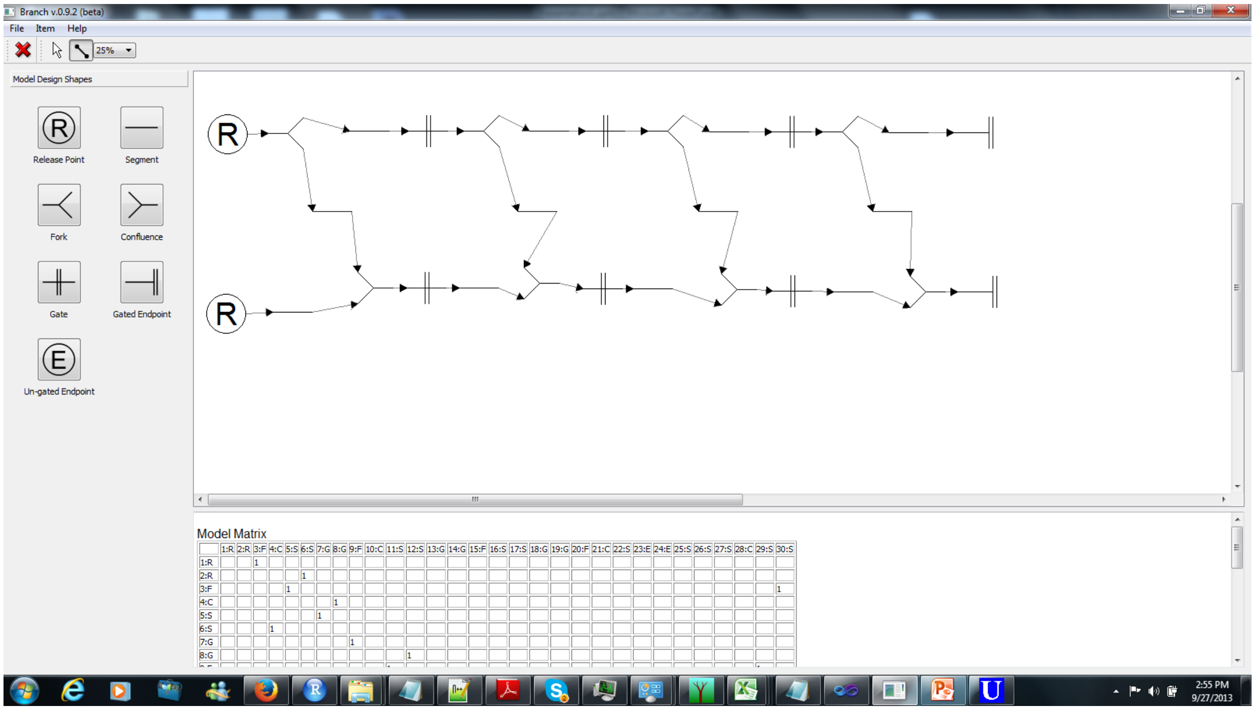
The Columbia Basin PIT Tag Information System (PTAGIS) is a centralized database operated by the Pacific States Marine Fisheries Commission for all PIT-tagged fish from the Columbia River basin (www.ptagis.org). The PTAGIS website was used to query and filter data used for this analysis. Complete detection records were accessed and downloaded from PTAGIS for all PIT-tagged juvenile Chinook salmon released anywhere in the Chiwawa River during calendar years 2010 and 2011. Each PIT tag carries a unique code and is individually identifiable. Additionally, PTAGIS maintains complete records for each PIT-tagged fish, including fish origin (wild or hatchery), weight, length, and age (determined and denoted by the tagger). From this metadata, data were filtered to provide only wild subyearling Chinook salmon from the 2010 brood year that were detected in either 2010 or 2011. Release dates for the subyearlings ranged from 2 July to 22 November 2010. Release dates for the yearlings ranged from 8 March to 11 August 2011. Finally, it was assumed that all juvenile Chinook salmon migrating downstream had done so by the end of 2011, as biologically, juvenile Chinook salmon outmigrate only as subyearlings or yearlings (Rich 1920). From this filtering process, data were analyzed for 3937 wild subyearling Chinook salmon and 4209 wild yearling Chinook salmon from the 2010 brood year that were PIT-tagged and released into the Chiwawa River.

3.3.3.2 Data analysis

Data were fit to a multistate, product-multinomial likelihood model created from a user specified diagram in Program BRANCH (Figure 3-4). The diagram illustrates four separate detection arrays downstream of the release point. As in the CJS model, the probability of surviving the last reach is not separable from the probability of being detected at the last site. Thus the detection arrays chosen for analysis (CHL, MWE, MCJ and a pooled last detection; see Study Site section) allowed estimation of overall cohort survival from the Chiwawa River to McNary Dam on the Columbia River, a reach which includes the entire length of the Wenatchee River and 284 km of the Columbia River traveled by these fish. Each detection at a particular site was assigned to one of two possible states based on a cutoff date for that site (Table 3-1). Detections before the cutoff date were assigned to the fall 2010 state, while detections after the date were considered to represent fish that had residualized and were assigned to the spring 2011 state. The cutoff date was chosen for each site separately by noting a length of time over which no detections occurred at that site.

Table 3-1. Site-specific date ranges for juvenile Chinook salmon detections assigned to State 2010 and 2011.

Detection Array	Dates assigned to State 2010	Dates assigned to State 2011
CHL	Aug 7 2010 – Dec 14 2010	Feb 11 2011 – Jun 8 2011
MWE	Jul 16 2010 – Feb 9 2011	Mar 2 2011 – May 15 2011
MCJ	NA	Apr 30 2011 – Jul 13 2011



Model Matrix

	1:R	2:R	3:F	4:C	5:S	6:S	7:G	8:G	9:F	10:C	11:S	12:S	13:G	14:G	15:F	16:S	17:S	18:G	19:G	20:F	21:C	22:S	23:E	24:E	25:S	26:S	27:S	28:C	29:S	30:S
1:R			1																											
2:R						1																								
3:F					1																									1
4:C								1																						
5:S							1																							
6:S				1																										
7:G									1																					
8:G												1																		

Figure 3-4. Screen capture of a user-specified diagram of a study design for a release-recapture study of the 2010 cohort of fall Chinook salmon in the Chiwawa River.

The multistate model used to estimate transition probabilities and overall survival for the juvenile Chinook salmon cohort outmigrating from the Chiwawa River in the fall of 2010 and spring of 2011 can also be represented as a schematic, with either all parameters or only estimable parameters labeled (Figure 3-5). The full or saturated multistate model for these data includes survival and state-change probabilities which are not necessarily separately estimable (Figure 3-5a). In fact, for this type of multistate model, where a possibility exists between each successive detection array of remaining in a state or transitioning to another state, in no case are the survival probabilities from one detection array to the next separable from the probability of changing states. Only the transition probability from one detection array to the next is estimable. This transition probability is the joint probability of either remaining in a state or changing states and surviving from one detection array to the next (Figure 3-5b). Program BRANCH was able to determine which probabilities were estimable in writing the multistate product-multinomial likelihood equation in first-step conditional format (Figure 3-6). Additionally, certain estimable parameters were not calculable based on the data. Program BRANCH used sufficient statistics to determine which parameters were calculable and to fix non-calculable parameters (e.g. no fish were detected at MCJ in 2010, so the transition probability from MWE to MCJ in 2010 is non-calculable and was fixed at 0). As in the previous analysis, calculable parameters were estimated via numeric methods in Program USER from the likelihood, and an overall cohort survival probability was calculated as a function of these estimates.

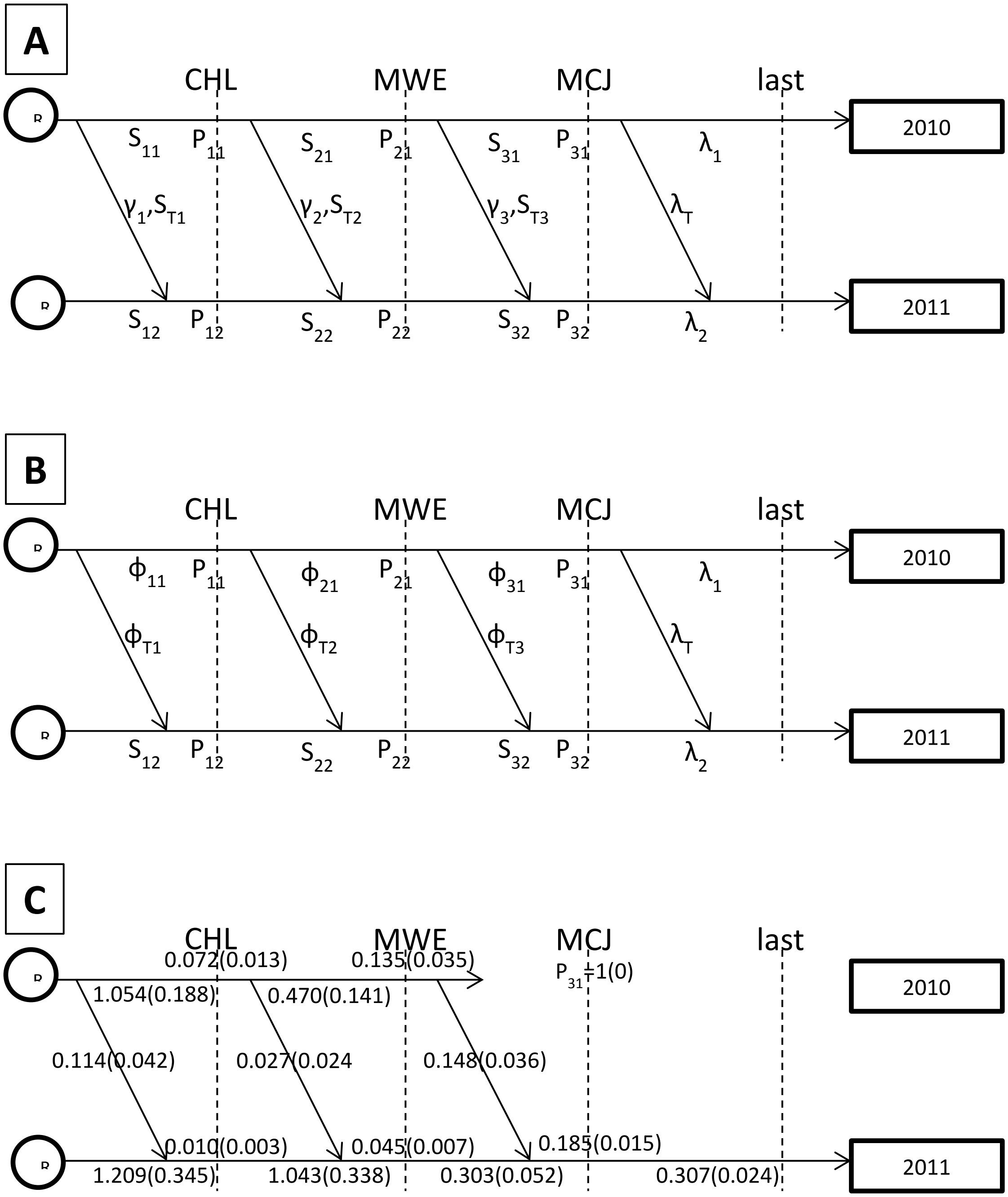


Figure 3-5. Multi-state model used for estimating migratory parameters for juvenile fall Chinook salmon data, 2010-11. A) Saturated model. Not all parameters are estimable. B) Model showing estimable transition probabilities. C) Model fit to data, with parameter estimates and standard errors.

$$
\begin{aligned}
L\left(S_i,\phi_i,P_i,\lambda_i\right)\propto & \left\{\begin{array}{l}
\left[\phi_{11}P_{11}\right]^{a_{11|r1}}\times\left[\phi_{T1}P_{12}\right]^{a_{12|r1}}\times\left[\phi_{11}\left(1-P_{11}\right)\phi_{21}P_{21}\right]^{a_{21|r1}}\times\left[\phi_{11}\left(1-P_{11}\right)\phi_{21}\left(1-P_{21}\right)\phi_{31}P_{31}\right]^{a_{31|r1}}\\
\times\left[\phi_{11}\left(1-P_{11}\right)\phi_{21}\left(1-P_{21}\right)\phi_{31}\left(1-P_{31}\right)\lambda_1\right]^{a_{l1|r1}}\times\left[\left(\phi_{11}\left(1-P_{11}\right)\phi_{T2}+\phi_{T1}\left(1-P_{12}\right)S_{22}\right)P_{22}\right]^{a_{22|r1}}\\
\times\left[\left(\phi_{11}\left(1-P_{11}\right)\left(\phi_{21}\left(1-P_{21}\right)\phi_{T3}+\phi_{T2}\left(1-P_{22}\right)S_{32}\right)+\phi_{T1}\left(1-P_{12}\right)S_{22}\left(1-P_{22}\right)S_{32}\right)P_{32}\right]^{a_{32|r1}}\\
\times\left[\begin{array}{l}
\phi_{11}\left(1-P_{11}\right)\phi_{21}\left(1-P_{21}\right)\phi_{31}\left(1-P_{31}\right)\lambda_T+\phi_{11}\left(1-P_{11}\right)\phi_{21}\left(1-P_{21}\right)\phi_{T3}\left(1-P_{32}\right)\lambda_2\\
+\phi_{11}\left(1-P_{11}\right)\phi_{T2}\left(1-P_{22}\right)S_{32}\left(1-P_{32}\right)\lambda_2+\phi_{T1}\left(1-P_{12}\right)S_{22}\left(1-P_{22}\right)S_{32}\left(1-P_{32}\right)\lambda_2
\end{array}\right]^{a_{l2|r1}}\\
\times\left[\begin{array}{l}
1-\phi_{11}-\phi_{T1}+\phi_{11}(1-P_{11})(1-\phi_{21}-\phi_{T2}+\phi_{21}(1-P_{21})(1-\phi_{31}-\phi_{T3}+\phi_{31}(1-P_{31})(1-\lambda_1-\lambda_T)\\
+\phi_{T3}(1-P_{32})(1-\lambda_2))+\phi_{T2}(1-P_{22})(1-S_{32}+S_{32}(1-P_{32})(1-\lambda_2)))\\
+\phi_{T1}(1-P_{12})(1-S_{22}+S_{22}(1-P_{22})(1-S_{32}+S_{32}(1-P_{32})(1-\lambda_2)))
\end{array}\right]^{a_{0|r1}}
\end{array}\right\}\\
& \times\left\{\begin{array}{l}
\left[S_{12}P_{12}\right]^{a_{12|r2}}\times\left[S_{12}\left(1-P_{12}\right)S_{22}P_{22}\right]^{a_{22|r2}}\times\left[S_{12}\left(1-P_{12}\right)S_{22}\left(1-P_{22}\right)S_{32}P_{32}\right]^{a_{32|r2}}\\
\times\left[S_{12}\left(1-P_{12}\right)S_{22}\left(1-P_{22}\right)S_{32}\left(1-P_{32}\right)\lambda_2\right]^{a_{l2|r2}}\\
\times\left[1-S_{12}+S_{12}(1-P_{12})(1-S_{22}+S_{22}(1-P_{22})(1-S_{32}+S_{32}(1-P_{32})(1-\lambda_2)))\right]^{a_{0|r2}}
\end{array}\right\}\\
& \times\left\{\begin{array}{l}
\left[\phi_{21}P_{21}\right]^{a_{21|11}}\times\left[\phi_{21}\left(1-P_{21}\right)\phi_{31}P_{31}\right]^{a_{31|11}}\times\left[\phi_{21}\left(1-P_{21}\right)\phi_{31}\left(1-P_{31}\right)\lambda_1\right]^{a_{l1|11}}\times\left[\phi_{T2}P_{22}\right]^{a_{22|11}}\\
\times\left[\left(\phi_{21}\left(1-P_{21}\right)\phi_{T3}+\phi_{T2}\left(1-P_{22}\right)S_{32}\right)P_{32}\right]^{a_{32|11}}\\
\times\left[\phi_{21}\left(1-P_{21}\right)\left(\phi_{31}\left(1-P_{31}\right)\lambda_T+\phi_{T3}\left(1-P_{32}\right)\lambda_2\right)+\phi_{T2}\left(1-P_{22}\right)S_{32}\left(1-P_{32}\right)\lambda_2\right]^{a_{l2|11}}\\
\times\left[\begin{array}{l}
1-\phi_{21}-\phi_{T2}+\phi_{21}(1-P_{21})(1-\phi_{31}-\phi_{T3}+\phi_{31}(1-P_{31})(1-\lambda_1-\lambda_T)+\phi_{T3}(1-P_{32})(1-\lambda_2))\\
+\phi_{T2}(1-P_{22})(1-S_{32}+S_{32}(1-P_{32})(1-\lambda_2))
\end{array}\right]^{a_{0|11}}
\end{array}\right\}\\
& \times\left\{\begin{array}{l}
\left[S_{22}P_{22}\right]^{a_{22|12}}\times\left[S_{22}\left(1-P_{22}\right)S_{32}P_{32}\right]^{a_{32|12}}\times\left[S_{22}\left(1-P_{22}\right)S_{32}\left(1-P_{32}\right)\lambda_2\right]^{a_{l2|12}}\\
\times\left[1-S_{22}+S_{22}\left(1-P_{22}\right)\left(1-S_{32}+S_{32}\left(1-P_{32}\right)\left(1-\lambda_2\right)\right)\right]^{a_{0|12}}
\end{array}\right\}\\
& \times\left\{\begin{array}{l}
\left[\phi_{31}P_{31}\right]^{a_{31|21}}\times\left[\phi_{T3}P_{32}\right]^{a_{32|21}}\times\left[\phi_{31}\left(1-P_{31}\right)\lambda_1\right]^{a_{l1|21}}\times\left[\phi_{31}\left(1-P_{31}\right)\lambda_T+\phi_{T3}\left(1-P_{32}\right)\lambda_2\right]^{a_{l2|21}}\\
\times\left[1-\phi_{31}-\phi_{T3}+\phi_{31}\left(1-P_{31}\right)\left(1-\lambda_1-\lambda_T\right)+\phi_{T3}\left(1-P_{32}\right)\left(1-\lambda_2\right)\right]^{a_{0|21}}
\end{array}\right\}\\
& \times\left\{\left[S_{32}P_{32}\right]^{a_{32|22}}\times\left[S_{32}\left(1-P_{32}\right)\lambda_2\right]^{a_{l2|22}}\times\left[1-S_{32}+S_{32}\left(1-P_{32}\right)\left(1-\lambda_2\right)\right]^{a_{0|22}}\right\}\\
& \times\left\{\left[\lambda_1\right]^{a_{l1|31}}\times\left[\lambda_T\right]^{a_{l2|31}}\times\left[1-\lambda_1-\lambda_T\right]^{a_{0|31}}\right\}\times\left\{\left[\lambda_2\right]^{a_{l2|32}}\times\left[1-\lambda_2\right]^{a_{0|32}}\right\}
\end{aligned}
$$

Figure 3-6. Product-multinomial likelihood equation used to estimate parameters for juvenile fall Chinook salmon, Wenatchee River basin, 2010-11. Parameter names follow terminology from Figure 3-5b. Statistics $a_{i|j}$ are the number of identifiable tags detected at site j, given detection at site i.

3.3.4 Results

Survival of the 2010 cohort of fall Chinook salmon from the Chiwawa River headwater to McNary Dam was estimated at 0.118 ($\widehat{SE} = 0.011$). Overall cohort survival is calculated as the sum of the transition probabilities for each possible route through the model. Each route-specific transition probability is in turn the probability that a fish successfully traversed that particular route from release to McNary Dam, and is calculated as the product of the individual transition probabilities that comprise that route. The equation for S_{cohort} is given here:

$$S_{cohort} = \phi_{11}\phi_{21}\phi_{31} + \phi_{11}\phi_{21}\phi_{T3} + \phi_{11}\phi_{T2}S_{32} + \phi_{T1}S_{22}S_{32}$$

Because one can employ the same methods to estimate cohort survival from release to each of the detection arrays, individual reach survival for the 2010 cohort is also calculable, as the quotient of survival from release to the farthest point downstream in the reach, divided by survival for release to the farthest upstream point in the reach. Equations for individual reach survival parameters S_i (i=1,2,3) are given here:

$$S_1 = \phi_{11} + \phi_{T1}$$

$$S_2 = \frac{\phi_{11}\phi_{21} + \phi_{11}\phi_{T2} + \phi_{T1}S_{22}}{\phi_{11} + \phi_{T1}}$$

$$S_3 = \frac{\phi_{11}\phi_{21}\phi_{31} + \phi_{11}\phi_{21}\phi_{T3} + \phi_{11}\phi_{T2}S_{32} + \phi_{T1}S_{22}S_{32}}{\phi_{11}\phi_{21} + \phi_{11}\phi_{T2} + \phi_{T1}S_{22}}$$

Individual reach survival probabilities are defined by S_i with i=1 the reach from release to CHL, i=2 the reach from CHL to MWE, and i=3 the reach from MWE to MCN. Reach survival probabilities for the 2010 cohort of fall Chinook salmon were estimated as follows: $S_1 = 1.168$ ($\widehat{SE} = 0.187$), $S_2 = 0.550$ ($\widehat{SE} = 0.132$), and $S_3 = 0.184$ ($\widehat{SE} = 0.036$; Table 3-2).

Table 3-2. Reach-specific and overall survival estimates for the 2010 juvenile fall Chinook salmon cohort, Chiwawa River.

Survival reach	Estimate (estimated SE)
Release to CHL	1.168 (0.187)
CHL to MWE	0.550 (0.132)
MWE to MCJ	0.184 (0.036)
Release to MCJ	0.118 (0.011)

The detection and transition probabilities displayed in Figure 3-5b were estimated directly from the likelihood models and were used to derive all reach and cohort survival parameters reported above (Figure 3-5c). Detection probability estimates were generally low and ranged from 0.010 at CHL in 2011 ($\widehat{SE} = 0.003$) to 0.185 at MCJ in 2011 ($\widehat{SE} = 0.015$). Parameter estimates greater than 1 (cohort survival from release to CHL, S_{12}, S_{22}, and ϕ_{11}) can likely be attributed to these low detection probabilities. In all cases where parameter estimates were greater than 1, the estimated standard error was large enough so that 95% confidence intervals overlapped with the admissible interval between 0 and 1.

3.3.5 Discussion

The model used to fit the juvenile Chinook data (Figure 3-5c), as well as the model showing all estimable parameters (Figure 3-5b), are specific realizations of multi-state models, used frequently in release-recapture studies. While several algorithms exist which are able to estimate parameters given data and a proper multinomial likelihood, the process of generating that likelihood has been heretofore left to the analyst. Program BRANCH required only the construction of the diagram shown in Figure 3-4, a list of state names and detection array names, and the data (a list of detections with tag identification, location and time of detection and release group) in order to automatically generate the specific model needed for parameter estimation. Parameter estimability is initially determined by the structure of the model itself while further refinement comes from features of the data. Specifically, sufficient statistics calculated from the data can show that certain parameters must be fixed at either 0 or 1, or that the sum of at least two parameters must be constrained to equal 1. For example, no fish were detected at either McNary Dam or downstream from McNary Dam in 2010. Thus the transition probabilities both into and out from MCJ in state 2010 (ϕ_{31}, λ_1, and λ_2) must be fixed at 0 prior to parameter estimation. This is equivalent to dropping the MCJ detection site from the model state 2010. By searching the data for these types of constraints, program BRANCH is able to arrive at Figure 3-5c. These rules are general enough that they should be applicable to almost any multistate model, making program BRANCH a widely useful tool for biologists involved in release-recapture studies.

Ecologically, the fact that no fish were detected at the McNary Dam juvenile bypass system in 2010 is significant and bears further discussion. The PTAGIS website metadata show that the McNary Dam juvenile bypass facility, where all PIT tag detections occur at the dam, shut down on November 22, 2010 and reopened on March 27, 2011. The de-watering of these fish facilities is a common practice at dams in the Columbia River basin. It is impossible to know how this may have affected migration and survival estimates for this analysis. It is possible that some proportion of juvenile Chinook salmon migrated seaward during the time that no PIT tag detections were possible at McNary Dam or at dams downstream. It is also possible that these fish residualized upstream from these dams, migrating the following spring in warmer river conditions. The uncertainty surrounding this issue presents an argument for supplementing PIT tag studies of migrating subyearling juvenile Chinook salmon with acoustic or radio telemetry studies.

3.4 Example 3 – Multistate model for Lost River and shortnose suckers in Clear Lake Reservoir

3.4.1 Introduction

Release-recapture studies are widely used throughout wildlife and fisheries biology as a means to estimate population parameters such as survival and movement (Perry et. al. 2010, Schwartz et. al. 1993, Hestbeck et. al 1991). The widespread use of this methodology leads to many types of multinomial and product-multinomial models. The multistate model represents one such category, where the assumption that all animals have homogeneous survival and detection processes is relaxed. In order to determine the versatility of the prototype program BRANCH, data from a multiyear study of two species of sucker in Clear Lake Reservoir, California were used to generate multistate models. The data and models were then used to estimate survival and migration parameters for these two fish populations.

3.4.2 Study site

Clear Lake Reservoir is located in northeastern California as part of the Clear Lake National Wildlife Refuge (Figure 3-7). The reservoir is dammed at its outflow into the Lost River, which exits the lake and flows northwesterly. There are a few small tributaries to Clear Lake Reservoir. Near the dam in the northeast corner of the lake is Willow Creek, a tributary which itself is joined by Boles Creek a mile upstream from its confluence into Clear Lake. The reservoir is home to two species of the Catostomidae family, the shortnose sucker (*Chasmistes* *brevirostis*) and the Lost River sucker (*Deltistes luxatus*). Both species are federally listed as endangered, and both species spawn in a riverine environment. Shortnose and Lost River suckers have been shown to spawn in Willow Crrek and Boles Creek but not in any other of the Clear Lake tributaries. Additionally, it is believed that the dam represents an impassible barrier from Lost River to Clear Lake for both species.

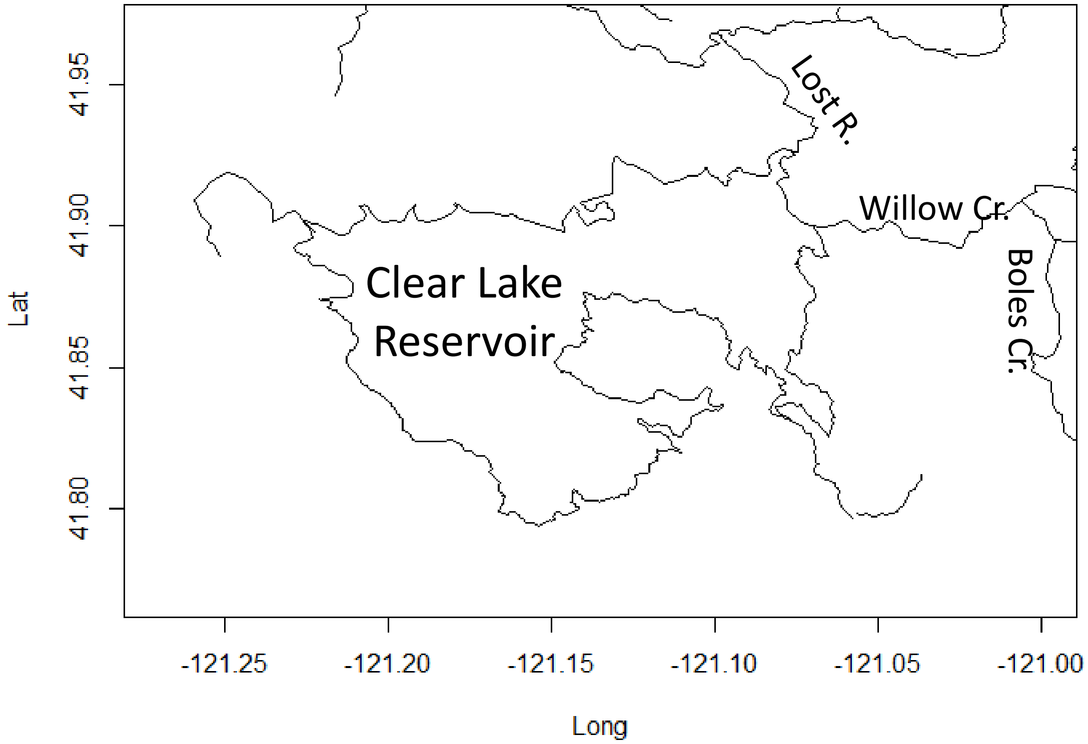


Figure 3-7. Map of Clear Lake Reservoir in northeastern California. Lost River sucker and shortnose sucker spawning occurs in the Willow Creek and Boles Creek tributaries.

3.4.3 Methods

3.4.3.1 Tagging and data recovery

All tagging and data recovery was performed by the US Bureau of Reclamation, Klamath Falls Office. A total of 75 fish (22 Lost River suckers, 53 shortnose suckers) were captured and tagged in the autumn of 2010. Tags were uniquely identifiable by frequency and code. Tagged fish were then released into Clear Lake Reservoir. A total of 16 aerial surveys of Clear Lake Reservoir and surrounding tributaries were performed between February 23 and August 25, 2011. Each survey constituted approximately two hours of airtime. Airplane locations were tracked via GPS, and tagged fish location was taken to be directly below the airplane at the time of the strongest signal from that tag. Tag detections whose signal strength never exceeded a set threshold were not recorded.

Tag detections were reconciled with the geography of Clear Lake Reservoir and its tributaries, and Lost River (Figure 3-8). This reconciliation was performed by staff at the University of Washington and resulted in each detection being designated as either in Clear Lake Reservoir, in the Willow Creek or its tributaries, or downstream of the dam in Lost River. Both tag expulsion and avian predation are known to occur. Individual detection events were also manually screened by UW staff for likely predation or tag expulsion. Predated fish were considered dead and all subsequent detections removed. Detections from tags considered expelled or in an impossible location were considered false detections and removed.

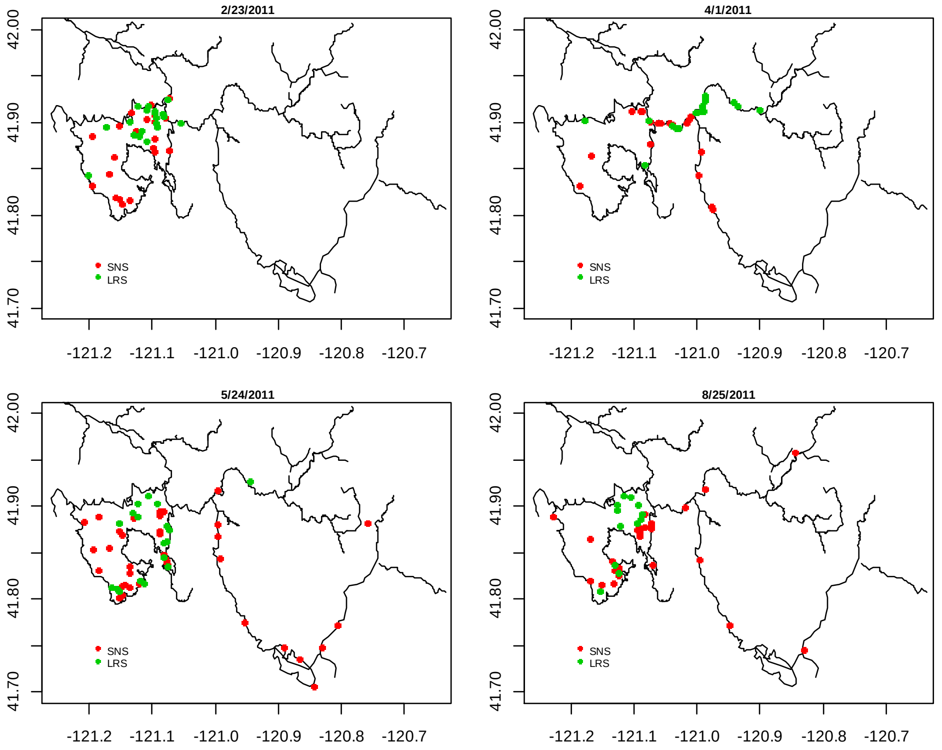
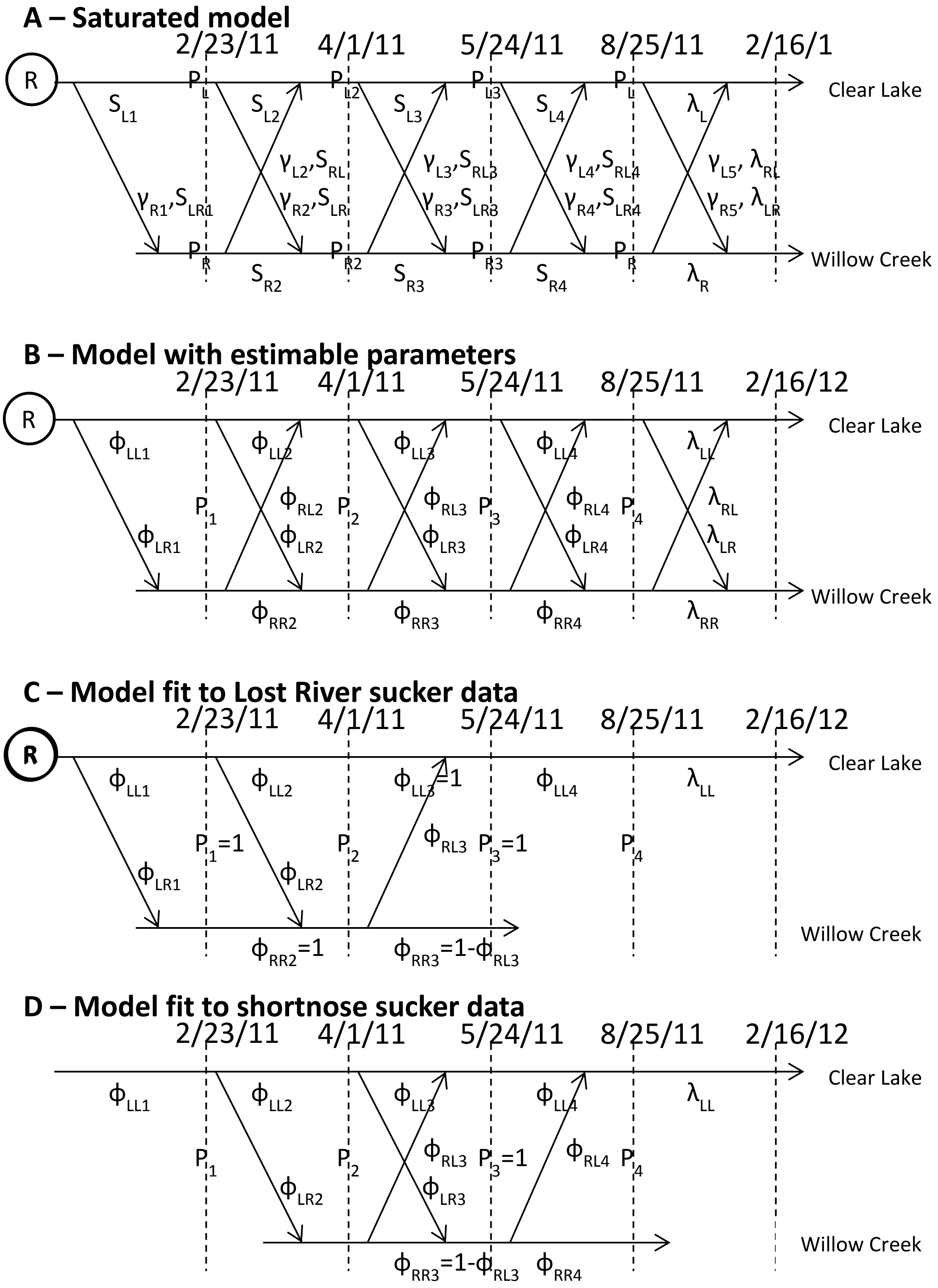


Figure 3-8. Map showing position of tag detections for both Lost River suckers (LRS, green dots) and shortnose suckers (SNS, red dots) during four survey periods in 2011.

3.4.3.2 Data analysis

Data were fit to a multistate, product-multinomial model in order to estimate survival and migration probabilities for Lost River and shortnose suckers separately (Figure 3-9). For clarity and relative model simplicity, four surveys from 2011 and one from 2012 were used in fitting the data. The February 16, 2012 survey was used as the last detection in order to estimate survival and migration through the final survey period in 2011. For each survey, tagged fish were classified as either detected in Clear Lake Reservoir, detected in Willow Creek or its tributaries, or not detected. Figure 3-9a shows the full, saturated model. Not all of the parameters shown in Figure 3-9a are estimable. Figure 3-9b shows the same schematic with only the estimable parameters displayed. Each estimable parameter is a function of one or more parameters from the saturated model.



A – Saturated model

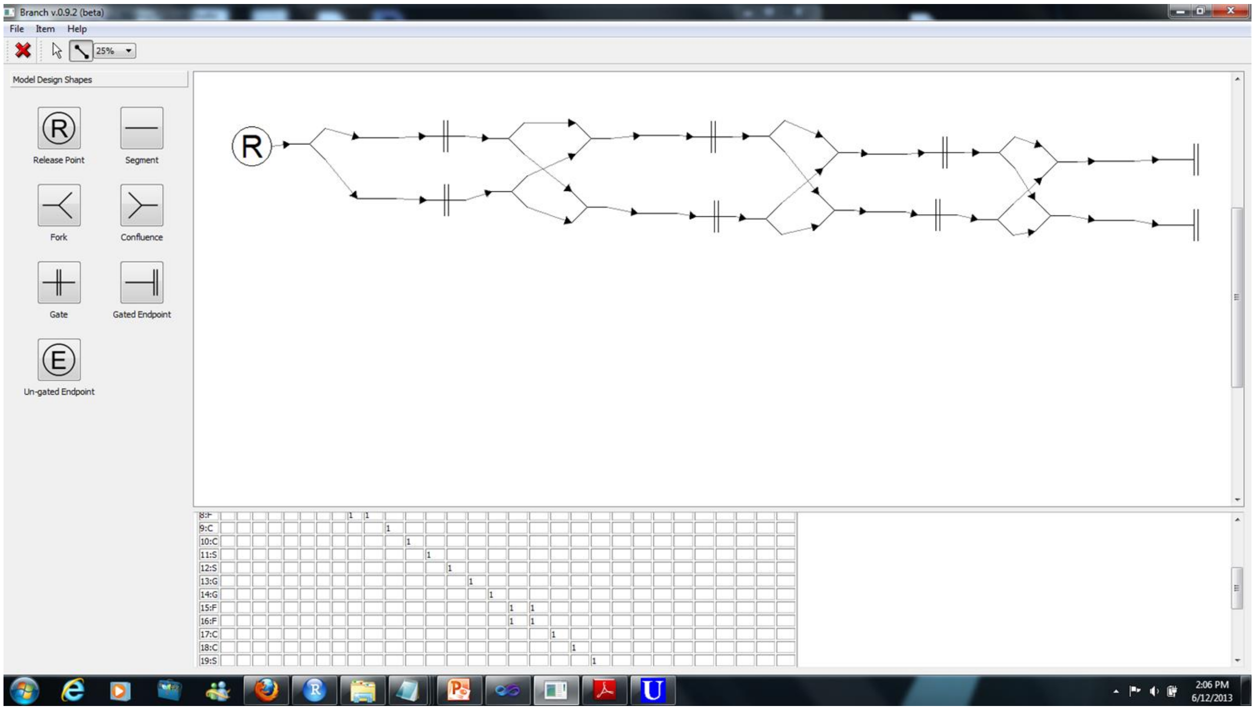
B – Model with estimable parameters

C – Model fit to Lost River sucker data

D – Model fit to shortnose sucker data

Figure 3-9. Multi-state models used for estimating migratory parameters for Lost River sucker and shortnose sucker data, 2011. A) Saturated model. Not all parameters are estimable. B) Model showing estimable transition probabilities. C) Model fit to Lost River sucker data. D) Model fit to shortnose sucker data.

The GUI representation of the release-recapture model developed in the program BRANCH prototype (Figure 3-10) determined the estimable parameters shown in Figure 3-9b. A salient feature of these multistate models is that in no case is the probability of surviving from one survey period to the next directly separable from the transition probability from one state to another during the same period. For this reason only the joint probability of migration and survival from one survey period to the next is estimable in the likelihood.



8:F									1	1																				
9:C											1																			
10:C												1																		
11:S													1																	
12:S														1																
13:G															1															
14:G																1														
15:F																	1	1												
16:F																	1	1												
17:C																			1											
18:C																				1										
19:S																					1									

Figure 3-10. Screen capture of GUI environment for program BRANCH. User-created graphic representation is equivalent to the schematic in Figure 3-9b.

Additional modifications to the multistate model were made in program BRANCH based on the sparseness of the data for both shortnose and Lost River suckers. The resulting models are shown in Figures 3-9c and 3-9d. These are the models to which the shortnose sucker and Lost River sucker data were respectively fit. Mathematically, the likelihood model used to estimate parameters for the Lost River suckers and that for shortnose suckers are displayed in Figures 3-11 and 3-12, respectively.

$$
\begin{aligned}
L(\phi_{ijk},\lambda_{ij},P_k) \propto & \left\{\begin{array}{l}
[\phi_{LL1}P_1]^{a_{L1}} \times [\phi_{LR1}P_1]^{a_{R1}} \times [\phi_{LL1}\phi_{LL2}(1-P_1)P_2]^{a_{L2}} \times [(\phi_{LL1}\phi_{LR2}+\phi_{LR1}\phi_{RR2})(1-P_1)P_2]^{a_{R2}} \\
\times[(\phi_{LL1}\phi_{LL2}\phi_{LL3}+\phi_{LL1}\phi_{LR2}\phi_{RL3}+\phi_{LR1}\phi_{RR2}\phi_{RL3})(1-P_1)(1-P_2)P_3]^{a_{L3}} \\
\times[(\phi_{LL1}\phi_{LR2}\phi_{RR3}+\phi_{LR1}\phi_{RR2}\phi_{RR3})(1-P_1)(1-P_2)P_3]^{a_{R3}} \\
\times[(\phi_{LL1}(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3})\phi_{LL4}+\phi_{LR1}\phi_{RR2}\phi_{RL3}\phi_{LL4})(1-P_1)(1-P_2)(1-P_3)P_4]^{a_{L4}} \\
\times[(\phi_{LL1}(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3})\phi_{LL4}+\phi_{LR1}\phi_{RR2}\phi_{RL3}\phi_{LL4})\lambda_{LL}(1-P_1)(1-P_2)(1-P_3)(1-P_4)]^{a_{L5}} \\
\times\left[\begin{array}{l}
1-\phi_{LL1}-\phi_{LR1}+\phi_{LL1}(1-P_1)(1-\phi_{LL2}-\phi_{LR2}+\phi_{LL2}(1-P_2)(1-\phi_{LL3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4} \\
+\phi_{LL4}(1-P_4)(1-\lambda_{LL})))+\phi_{LR2}(1-P_2)(\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL})) \\
+\phi_{RR3}(1-P_3)))+\phi_{LR1}(1-P_1)(1-\phi_{RR2}+\phi_{RR2}(1-P_2)(\phi_{RR3}(1-P_3) \\
+\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))))
\end{array}\right]^{a_0}
\end{array}\right\} \\
& \times\left\{\begin{array}{l}
[\phi_{LL2}P_2]^{a_{L2|L1}} \times [\phi_{LR2}P_2]^{a_{R2|L1}} \times [(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3})(1-P_2)P_3]^{a_{L3|L1}} \\
\times[\phi_{LR2}\phi_{RR3}(1-P_2)P_3]^{a_{R3|L1}} \times [(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3})\phi_{LL4}(1-P_2)(1-P_3)P_4]^{a_{L4|L1}} \\
\times[(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3})\phi_{LL4}\lambda_{LL}(1-P_2)(1-P_3)(1-P_4)]^{a_{L5|L1}} \\
\times\left[\begin{array}{l}
1-\phi_{LL2}-\phi_{LR2}+\phi_{LL2}(1-P_2)(1-\phi_{LL3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))) \\
+\phi_{LR2}(1-P_2)(\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{RR3}(1-P_3))
\end{array}\right]^{a_{0|L1}}
\end{array}\right\} \\
& \times\left\{\begin{array}{l}
[\phi_{RR2}P_2]^{a_{R2|R1}} \times [\phi_{RR2}\phi_{RL3}(1-P_2)P_3]^{a_{L3|R1}} \times [\phi_{RR2}\phi_{RR3}(1-P_2)P_3]^{a_{R3|R1}} \\
\times[\phi_{RR2}\phi_{RL3}\phi_{LL4}(1-P_2)(1-P_3)P_4]^{a_{L4|R1}} \times [\phi_{RR2}\phi_{RL3}\phi_{LL4}\lambda_{LL}(1-P_2)(1-P_3)(1-P_4)]^{a_{L5|R1}} \\
\times[1-\phi_{RR2}+\phi_{RR2}(1-P_2)(\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{RR3}(1-P_3))]^{a_{0|R1}}
\end{array}\right\} \\
& \times\left\{\begin{array}{l}
[\phi_{LL3}P_3]^{a_{L3|L2}} \times [\phi_{LL3}\phi_{LL4}(1-P_3)P_4]^{a_{L4|L2}} \times [\phi_{LL3}\phi_{LL4}\lambda_{LL}(1-P_3)(1-P_4)]^{a_{L5|L2}} \\
\times[1-\phi_{LL3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))]^{a_{0|L2}}
\end{array}\right\} \\
& \times\left\{\begin{array}{l}
[\phi_{RL3}P_3]^{a_{L3|R2}} \times [\phi_{RR3}P_3]^{a_{R3|R2}} \times [\phi_{RL3}\phi_{LL4}(1-P_3)P_4]^{a_{L4|R2}} \times [\phi_{RL3}\phi_{LL4}\lambda_{LL}(1-P_3)(1-P_4)]^{a_{L5|R2}} \\
\times[\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{RR3}(1-P_3)]^{a_{0|R2}}
\end{array}\right\} \\
& \times\left\{[\phi_{LL4}P_4]^{a_{L4|L3}} \times [\phi_{LL4}\lambda_{LL}(1-P_4)]^{a_{L5|L3}} \times [1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL})]^{a_{0|L3}}\right\} \\
& \times\left\{[\lambda_{LL}]^{a_{L5|L4}} \times [1-\lambda_{LL}]^{a_{0|L4}}\right\}
\end{aligned}
$$

Figure 3-11. Product-multinomial likelihood equation used to estimate parameters for Lost River suckers, Clear Lake Reservoir, 2011. Parameter names follow terminology from Figure 3-9c. Statistics $a_{i|j}$ are the number of identifiable tags detected at site j, given their detection at site i.

$$
\begin{aligned}
L\left(\phi_{ijk},\lambda_{ij},P_k\right)\propto & \left\{\begin{aligned}
&\left[\phi_{LL1}P_1\right]^{a_{L1}}\times\left[\phi_{LL1}\phi_{LL2}\left(1-P_1\right)P_2\right]^{a_{L2}}\times\left[\phi_{LL1}\phi_{LR2}\left(1-P_1\right)P_2\right]^{a_{R2}}\\
&\times\left[\left(\phi_{LL1}\phi_{LL2}\phi_{LL3}+\phi_{LL1}\phi_{LR2}\phi_{RL3}\right)\left(1-P_1\right)\left(1-P_2\right)P_3\right]^{a_{L3}}\\
&\times\left[\left(\phi_{LL1}\phi_{LR2}\phi_{RR3}+\phi_{LL1}\phi_{LL2}\phi_{LR3}\right)\left(1-P_1\right)\left(1-P_2\right)P_3\right]^{a_{R3}}\\
&\times\left[\phi_{LL1}\left(\phi_{LL2}\phi_{LL3}\phi_{LL4}+\phi_{LR2}\phi_{RL3}\phi_{LL4}+\phi_{LL2}\phi_{LR3}\phi_{RL4}+\phi_{LR2}\phi_{RR3}\phi_{RL4}\right)\left(1-P_1\right)\left(1-P_2\right)\left(1-P_3\right)P_4\right]^{a_{L4}}\\
&\times\left[\phi_{LL1}\left(\phi_{LL2}\phi_{LR3}\phi_{RR4}+\phi_{LR2}\phi_{RR3}\phi_{RR4}\right)\left(1-P_1\right)\left(1-P_2\right)\left(1-P_3\right)P_4\right]^{a_{R4}}\\
&\times\left[\phi_{LL1}\left(\phi_{LL2}\phi_{LL3}\phi_{LL4}+\phi_{LR2}\phi_{RL3}\phi_{LL4}+\phi_{LL2}\phi_{LR3}\phi_{RL4}+\phi_{LR2}\phi_{RR3}\phi_{RL4}\right)\lambda_{LL}\left(1-P_1\right)\left(1-P_2\right)\left(1-P_3\right)\left(1-P_4\right)\right]^{a_{L5}}\\
&\times\left[\begin{aligned}
&1-\phi_{LL1}+\phi_{LL1}(1-P_1)(1-\phi_{LL2}-\phi_{LR2}+\phi_{LL2}(1-P_2)(1-\phi_{LL3}-\phi_{LR3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4}\\
&+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{LR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4)))\\
&+\phi_{LR2}(1-P_2)(\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{RR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}\\
&+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4))))
\end{aligned}\right]^{a_0}
\end{aligned}\right\}\\
&\times\left\{\begin{aligned}
&\left[\phi_{LL2}P_2\right]^{a_{L2|L1}}\times\left[\phi_{LR2}P_2\right]^{a_{R2|L1}}\times\left[\left(\phi_{LL2}\phi_{LL3}+\phi_{LR2}\phi_{RL3}\right)\left(1-P_2\right)P_3\right]^{a_{L3|L1}}\\
&\times\left[\left(\phi_{LR2}\phi_{RR3}+\phi_{LL2}\phi_{LR3}\right)\left(1-P_2\right)P_3\right]^{a_{R3|L1}}\\
&\times\left[\left(\phi_{LL2}\phi_{LL3}\phi_{LL4}+\phi_{LR2}\phi_{RL3}\phi_{LL4}+\phi_{LL2}\phi_{LR3}\phi_{RL4}+\phi_{LR2}\phi_{RR3}\phi_{RL4}\right)\left(1-P_2\right)\left(1-P_3\right)P_4\right]^{a_{L4|L1}}\\
&\times\left[\left(\phi_{LL2}\phi_{LR3}\phi_{RR4}+\phi_{LR2}\phi_{RR3}\phi_{RR4}\right)\left(1-P_2\right)\left(1-P_3\right)P_4\right]^{a_{R4|L1}}\\
&\times\left[\left(\phi_{LL2}\phi_{LL3}\phi_{LL4}+\phi_{LR2}\phi_{RL3}\phi_{LL4}+\phi_{LL2}\phi_{LR3}\phi_{RL4}+\phi_{LR2}\phi_{RR3}\phi_{RL4}\right)\lambda_{LL}\left(1-P_2\right)\left(1-P_3\right)\left(1-P_4\right)\right]^{a_{L5|L1}}\\
&\times\left[\begin{aligned}
&1-\phi_{LL2}-\phi_{LR2}+\phi_{LL2}(1-P_2)(1-\phi_{LL3}-\phi_{LR3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))\\
&+\phi_{LR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4)))\\
&+\phi_{LR2}(1-P_2)(\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))\\
&+\phi_{RR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4)))
\end{aligned}\right]^{a_{0|L1}}
\end{aligned}\right\}\\
&\times\left\{\begin{aligned}
&\left[\phi_{LL3}P_3\right]^{a_{L3|L2}}\times\left[\phi_{LR3}P_3\right]^{a_{R3|L2}}\times\left[\left(\phi_{LL3}\phi_{LL4}+\phi_{LR3}\phi_{RL4}\right)\left(1-P_3\right)P_4\right]^{a_{L4|L2}}\times\left[\phi_{LR3}\phi_{RR4}\left(1-P_3\right)P_4\right]^{a_{R4|L2}}\\
&\times\left[\left(\phi_{LL3}\phi_{LL4}+\phi_{LR3}\phi_{RL4}\right)\lambda_{LL}\left(1-P_3\right)\left(1-P_4\right)\right]^{a_{L5|L2}}\\
&\times\left[\begin{aligned}
&1-\phi_{LL3}-\phi_{LR3}+\phi_{LL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{LR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}\\
&+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4))
\end{aligned}\right]^{a_{0|L2}}
\end{aligned}\right\}\\
&\times\left\{\begin{aligned}
&\left[\phi_{RL3}P_3\right]^{a_{L3|R2}}\times\left[\phi_{RR3}P_3\right]^{a_{R3|R2}}\times\left[\left(\phi_{RL3}\phi_{LL4}+\phi_{RR3}\phi_{RL4}\right)\left(1-P_3\right)P_4\right]^{a_{L4|R2}}\times\left[\phi_{RR3}\phi_{RR4}\left(1-P_3\right)P_4\right]^{a_{R4|R2}}\\
&\times\left[\left(\phi_{RL3}\phi_{LL4}+\phi_{RR3}\phi_{RL4}\right)\lambda_{LL}\left(1-P_3\right)\left(1-P_4\right)\right]^{a_{L5|R2}}\\
&\times\left[\begin{aligned}
&\phi_{RL3}(1-P_3)(1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))+\phi_{RR3}(1-P_3)(1-\phi_{RL4}-\phi_{RR4}+\phi_{RR4}(1-P_4)\\
&+\phi_{LL4}(1-P_4)(1-\lambda_{LL}))
\end{aligned}\right]^{a_{0|R2}}
\end{aligned}\right\}\\
&\times\left\{\left[\phi_{LL4}P_4\right]^{a_{L4|L3}}\times\left[\phi_{LL4}\lambda_{LL}\left(1-P_4\right)\right]^{a_{L5|L3}}\times\left[1-\phi_{LL4}+\phi_{LL4}(1-P_4)(1-\lambda_{LL})\right]^{a_{0|L3}}\right\}\\
&\times\left\{\begin{aligned}
&\left[\phi_{RL4}P_4\right]^{a_{L4|R3}}\times\left[\phi_{RR4}P_4\right]^{a_{R4|R3}}\times\left[\phi_{RL4}\lambda_{LL}\left(1-P_4\right)\right]^{a_{L5|R3}}\\
&\times\left[1-\phi_{RL4}-\phi_{RR4}+\phi_{RL4}(1-P_4)(1-\lambda_{LL})+\phi_{RR4}(1-P_4)\right]^{a_{0|R3}}
\end{aligned}\right\}\\
&\times\left\{\left[\lambda_{LL}\right]^{a_{L5|L4}}\times\left[1-\lambda_{LL}\right]^{a_{0|L4}}\right\}
\end{aligned}
$$

Figure 3-12. Product-multinomial likelihood equation used to estimate parameters for shortnose suckers, Clear Lake Reservoir, 2011. Parameter names follow terminology from Figure 3-9d. Statistics $a_{i|j}$ are the number of identifiable tags detected at site j, given their detection at site i.

Once the data were fit, overall system metrics were estimated from model parameter estimates and variances calculated using the delta method. As in the previous analysis, overall cohort survival is calculated as the sum of the transition probabilities for each possible route through the model, with its equation given here:

$$
\begin{aligned}
S_{2011} = {} & \phi_{LL1}\phi_{LL2}\phi_{LL3}\phi_{LL4} + \phi_{LL1}\phi_{LL2}\phi_{LL3}\phi_{LR4} + \phi_{LL1}\phi_{LL2}\phi_{LR3}\phi_{RR4} + \phi_{LL1}\phi_{LL2}\phi_{LR3}\phi_{RL4} + \phi_{LL1}\phi_{LR2}\phi_{RR3}\phi_{RR4} + \phi_{LL1}\phi_{LR2}\phi_{RR3}\phi_{RL4} \\
& + \phi_{LL1}\phi_{LR2}\phi_{RL3}\phi_{LL4} + \phi_{LL1}\phi_{LR2}\phi_{RL3}\phi_{LR4} + \phi_{LR1}\phi_{RR2}\phi_{RR3}\phi_{RR4} + \phi_{LR1}\phi_{RR2}\phi_{RR3}\phi_{RL4} + \phi_{LR1}\phi_{RR2}\phi_{RL3}\phi_{LL4} + \phi_{LR1}\phi_{RR2}\phi_{RL3}\phi_{LR4} \\
& + \phi_{LR1}\phi_{RL2}\phi_{LL3}\phi_{LL4} + \phi_{LR1}\phi_{RL2}\phi_{LL3}\phi_{LR4} + \phi_{LR1}\phi_{RL2}\phi_{LR3}\phi_{RR4} + \phi_{LR1}\phi_{RL2}\phi_{LR3}\phi_{RL4}
\end{aligned}
$$

Survival throughout the entire study area from one survey period to the next is calculated in a manner analogous to the reach-specific survival estimates from the previous analysis, as the quotient of survival from release to the latter of the two survey dates and survival from release to the earlier of the two surveys. Period-specific survival equations are given here:

$$
\begin{aligned}
S_1 &= \phi_{LL1} + \phi_{LR1} \\
S_2 &= \frac{\phi_{LL1}\phi_{LL2} + \phi_{LL1}\phi_{LR2} + \phi_{LR1}\phi_{RR2} + \phi_{LR1}\phi_{RL2}}{\phi_{LL1} + \phi_{LR1}} \\
S_3 &= \frac{\phi_{LL1}\phi_{LL2}\phi_{LL3} + \phi_{LL1}\phi_{LL2}\phi_{LR3} + \phi_{LL1}\phi_{LR2}\phi_{RR3} + \phi_{LL1}\phi_{LR2}\phi_{RL3} + \phi_{LR1}\phi_{RR2}\phi_{RR3} + \phi_{LR1}\phi_{RR2}\phi_{RL3} + \phi_{LR1}\phi_{RL2}\phi_{LL3} + \phi_{LR1}\phi_{RL2}\phi_{LR3}}{\phi_{LL1}\phi_{LL2} + \phi_{LL1}\phi_{LR2} + \phi_{LR1}\phi_{RR2} + \phi_{LR1}\phi_{RL2}} \\
S_4 &= S_{2011} / \left(\phi_{LL1}\phi_{LL2}\phi_{LL3} + \phi_{LL1}\phi_{LL2}\phi_{LR3} + \phi_{LL1}\phi_{LR2}\phi_{RR3} + \phi_{LL1}\phi_{LR2}\phi_{RL3} + \phi_{LR1}\phi_{RR2}\phi_{RR3} + \phi_{LR1}\phi_{RR2}\phi_{RL3} + \phi_{LR1}\phi_{RL2}\phi_{LL3} + \phi_{LR1}\phi_{RL2}\phi_{LR3}\right)
\end{aligned}
$$

Also calculated were the joint probabilities of migrating into a tributary at any point during 2011 and surviving this migration (ϕ_{LR}), as well as of migrating into a tributary and back to Clear Lake Reservoir at any point during 2011 and surviving the round trip migration(ϕ_{LRL}). Both formulas are presented here:

$$
\begin{aligned}
\phi_{LR} &= \phi_{LR1} + \phi_{LL1}\phi_{LR2} + \phi_{LL1}\phi_{LL2}\phi_{LR3} + \phi_{LL1}\phi_{LL2}\phi_{LL3}\phi_{LR4} \\
\phi_{LRL} &= \phi_{LR1}\left(\phi_{RL2} + \phi_{RR2}\phi_{RL3} + \phi_{RR2}\phi_{RR3}\phi_{RL4}\right) + \phi_{LL1}\phi_{LR2}\left(\phi_{RL3} + \phi_{RR3}\phi_{RL4}\right) + \phi_{LL1}\phi_{LL2}\phi_{LR3}\phi_{RL4}
\end{aligned}
$$

3.4.4 Results

Overall survival in 2011 in both Clear Lake Reservoir and Willow Creek (S_{2011}) was 0.594 ($\widehat{SE} =$ 0.083) for shortnose suckers and 0.709 ($\widehat{SE} =$ 0.106) for Lost River Suckers (Table 3-3). Survival was also estimated for each species between each survey (parameters S_i in Table 3, i=1, …, 4). Between surveys survival ranged from 0.787 ($\widehat{SE} =$ 0.086) to 0.950 ($\widehat{SE} =$ 0.049) for shortnose suckers and from 0.821 ($\widehat{SE} =$ 0.102) to 1 ($\widehat{SE} =$ 0) for Lost River suckers.

Table 3-3. Parameter estimates and standard errors for Lost River sucker and shortnose sucker data, Clear Lake Reservoir 2011.

	Lost River suckers		**Shortnose suckers**	
Parameter	Estimate	SE	Estimate	SE
S_1	0.955	0.044	0.858	0.077
S_2	0.905	0.064	0.787	0.086
S_3	1	0	0.950	0.049
S_4	0.821	0.102	0.926	0.088
S_{2011}	0.709	0.106	0.594	0.083
ϕ_{LR}	0.719	0.098	0.484	0.083
ϕ_{LRL}	0.673	0.102	0.355	0.081
R_1	0.048	0.046	0	0
R_2	0.833	0.088	0.650	0.107
R_3	0.053	0.051	0.265	0.076
R_4	0	0	0.222	0.080
L_1	0.952	0.046	1	0
L_2	0.167	0.088	0.350	0.107
L_3	0.947	0.051	0.735	0.076
L_4	1	0	0.778	0.080
P_1	1	0	0.571	0.084
P_2	0.947	0.051	0.559	0.085
P_3	1	0	1	0
P_4	0.833	0.108	0.857	0.094

The parameter ϕ_{LR} was estimated at 0.484 ($\widehat{SE} =$ 0.083) for shortnose suckers and 0.719 ($\widehat{SE} =$ 0.098) for Lost River suckers (Table 3-3). The parameter ϕ_{LRL} was estimated at 0.355 ($\widehat{SE} =$ 0.081) for shortnose suckers and 0.673 ($\widehat{SE} =$ 0.102) for Lost River suckers (Table 3-3).

The proportion of fish detected in Willow Creek out of all fish of that species detected during survey i is denoted as R_i (i=1, …, 4). Analogously, the proportion of fish detected in Clear Lake Reservoir during survey i, out of all fish of that species detected during that survey, is denoted L_i (i=1, …, 4), and is the compliment of R_i where $L_i=1-R_i$ (Table 3-3). For shortnose suckers these parameters were estimated as follows: $\hat{R}_1 = 0$ ($\widehat{SE} = 0$), $\hat{R}_2 = 0.650$ ($\widehat{SE} = 0.107$), $\hat{R}_3 = 0.265$ ($\widehat{SE} = 0.076$), and $\hat{R}_4 = 0.222$ ($\widehat{SE} = 0.080$; Table 3-3). For Lost River suckers the estimates were: $\hat{R}_1 = 0.048$ ($\widehat{SE} = 0.046$), $\hat{R}_2 = 0.833$ ($\widehat{SE} = 0.088$), $\hat{R}_3 = 0.053$ ($\widehat{SE} = 0.051$), and $\hat{R}_4 = 0$ ($\widehat{SE} = 0$; Table 3-3). For both species and all survey periods, $L_i=1-R_i$ as mentioned, with the standard error for L_i equal in all cases to that for R_i. These proportions suggest that the migration of Lost River suckers into Willow Creek is more pronounced and abrupt than that of shortnose suckers.

The detection and transition probabilities shown in Figures 3-9c and 3-9d were estimated directly from the likelihood models, and were used to derive all S and ϕ parameters reported above (Figure 3-13a and 3-13b; Table 3-4). Probabilities of detection for shortnose suckers ranged from 0.559 ($\widehat{SE} = 0.085$) to 1 ($\widehat{SE} = 0$); detection probabilities for Lost River suckers ranged from 0.833 ($\widehat{SE} = 0.108$) to 1 ($\widehat{SE} = 0$; Table 3-3). All L and R parameters were derived from numbers of fish detected and were calculated outside of the likelihood models.

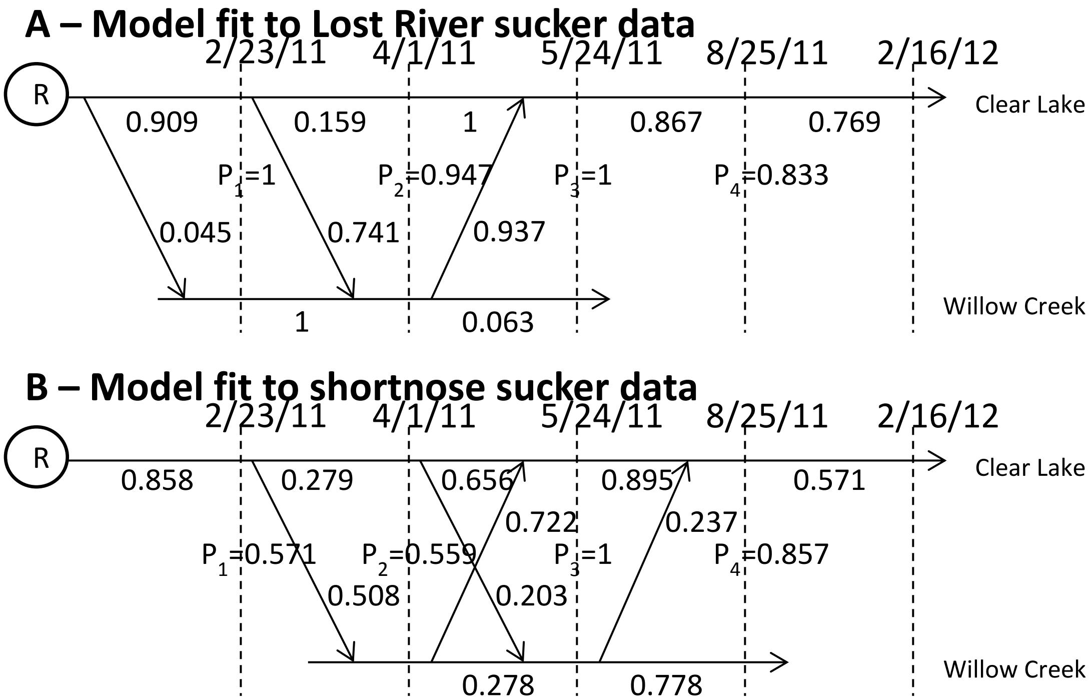


Figure 3-13. Multi-state models with parameter estimates for Lost River sucker and shortnose sucker data, 2011. Parameter estimates displayed; standard error estimates can be found in Table 3-4. A) Lost River sucker parameter estimates. B) Shortnose sucker parameter estimates.

Table 3-4. Transition parameter estimates and estimated standard errors for Lost River suckers and shortnose suckers in Clear Lake Reservoir, 2011. Parameter names follow terminology in Figures 3-9c and 3-9d. Gaps allow side by side comparison of analogous parameters between species.

Lost River suckers			**Shortnose suckers**		
Parameter	Estimate	SE	Parameter	Estimate	SE
ϕ_{LL1}	0.909	0.061	ϕ_{LL1}	0.858	0.077
ϕ_{LR1}	0.045	0.044			
ϕ_{LL2}	0.159	0.084	ϕ_{LL2}	0.279	0.093
ϕ_{LR2}	0.741	0.100	ϕ_{LR2}	0.508	0.100
ϕ_{RR2}	1	0			
ϕ_{LL3}	1	0	ϕ_{LL3}	0.656	0.166
			ϕ_{LR3}	0.203	0.129
ϕ_{RL3}	0.937	0.061	ϕ_{RL3}	0.722	0.100
ϕ_{RR3}	0.063	0.061	ϕ_{RR3}	0.278	0.100
ϕ_{LL4}	0.867	0.096	ϕ_{LL4}	0.895	0.090
			ϕ_{RL4}	0.237	0.148
			ϕ_{RR4}	0.778	0.202
λ_1	0.769	0.117	λ_1	0.571	0.108

Because only the joint probability of migration and survival from one survey period to the next was estimable, it was not possible to estimate separate survival probabilities for fish in Clear Lake versus those in Willow Creek. Stated more intuitively, because it is unknown at what point between survey periods a fish would have migrated from lake to river or vise-versa, it is not possible to say what portion of that fish's survival was spent in the Clear Lake and what portion in Willow Creek. The same principle also holds for estimation of migration probabilities: only the joint probability of migrating from river to lake or lake to river and surviving from one survey period to the next is estimable here. Careful consideration should therefore be given to interpretation of parameters ϕ_{LR} and ϕ_{LRL}, as these parameters are not entirely intuitive.

3.4.5 Discussion

3.4.5.1 Ecology

In 2011, Lost River suckers appear to have migrated into Willow Creek and back into Clear Lake Reservoir more abruptly than did shortnose suckers. Lost River suckers generally stayed in Clear Lake Reservoir before February 23 (Period 1), migrated into Willow Creek between February 23 and April 1 (Period 2), and returned to the lake between April 1 and May 24 (Period 3; Figure 3-13a, Table 3-4). Additionally, all tagged Lost River suckers successfully migrating back to Clear Lake Reservoir had done so by May 24 (Table 3-4, Figure 3-13a). In contrast, the migration pattern of shortnose suckers in 2011 was slightly less well-defined. While the probability of successful migration into Willow Creek, and of successful return to Clear Lake Reservoir for shortnose suckers was highest during Period 2 and Period 3, respectively, a sizeable fraction of shortnose suckers delayed migration to Willow Creek until Period 3 and return during Period 4 (Figure 3-13b, Table 3-4). Overall, shortnose suckers displayed a lower success rate for migration to Willow Creek and back to Clear Lake Reservoir than did Lost River suckers (Table 3-3).

It should be noted that conclusions can only be drawn concerning the periods between surveys chosen for analysis. The survey dates in the models were chosen for analytical convenience rather than based on any biological or ecological principle. Since more surveys were conducted than were analyzed here, further analysis of additional survey dates could bring further clarity to the research presented.

3.4.5.2 Modeling

There are a number of similarities between this analysis and the previous juvenile Chinook salmon cohort analysis, as can be seen from a comparison of the model structures from Figures 3-5 and 3-9. These two analyses are illustrative of the similarities as well as potential differences among multistate models. The recapture opportunities in the sucker study design are arranged in time; in the cohort analysis these were arranged instead in space. Conversely, the states in the sucker analysis were differing spatial locations whereas in the cohort analysis these states were temporal. Another key difference is that while movement between states in the cohort analysis was restricted unidirectionally (i.e. fish cannot move backwards in time), in this analysis suckers are free to move back and forth between river and lake states.

As in the cohort analysis, here parameter estimability is determined by the structure of the model while further refinement comes from the data. In the model used to estimate parameters for the Lost River sucker data (Figure 3-9d), the data show that one tagged fish was detected in Willow Creek on February 23, 2011 (survey period 1). The same fish was detected again in Willow Creek on April 1 2011. This fact indicates that all fish detected in the River state at survey period 1 are subsequently detected downstream, a condition of the sufficient statistics of the data which forces all transitional parameters flowing out of that state to sum to 1, since there is no mortality from that state (i.e. $\phi_{RR2} + \phi_{RL2} = 1$). Additionally, no fish detected in the River state at survey period 1 are detected in the Lake state in survey period 2, a condition which forces ϕ_{RL2} to be fixed at zero. By searching the data for these types of constraining conditions, program BRANCH is able to arrive at Figure 3-13a. This study thus represents another example of a multi-state model suitable for analysis via Program BRANCH despite its differences from the previous examples.

3.5 Example 4 – Multi-state model for juvenile Chinook salmon in the San Joaquin-Sacramento Delta

3.5.1 Introduction

The Vernalis Adaptive Management Plan (VAMP) is a multi-year study of juvenile Chinook salmon (*Oncorhynchus tshawytscha*) migration and survival through the Sacramento-San Joaquin Delta (*Delta*) in northern California. In 2011 the VAMP study tagged and released 1,895 juvenile Chinook salmon and tracked their movement through the Delta. Data collected from these salmon were fit to a multistate product-multinomial model created using program BRANCH in order to estimate survival and migration parameters.

3.5.2 Study site

The Sacramento-San Joaquin River Delta is located in the Central Valley of California (Figure 3-14). The San Joaquin River flows northwest past Stockton, California before joining the

Sacramento River near Chipps Island. The Delta study area is bounded to the east and north by the San Joaquin River. South of Stockton, near Mossdale, the Old River splits from the San Joaquin River and flows westward before turning north near the Central Valley Project (CVP) and State Water Project (SWP), two water export facilities that draw from the Delta. The Old River then rejoins the San Joaquin River just west of Mandeville Island. The Old River bounds the Delta study area on the southwest and west. The interior of the Delta is made up of the Middle River, which splits from the Old River, and various channels and cuts (Figure 3-14). The upstream boundary of the study area is the receiver array (A0) just upstream of the release point at Durham Ferry; the receiver array at Chipps Island marks the downstream limit of the study area.

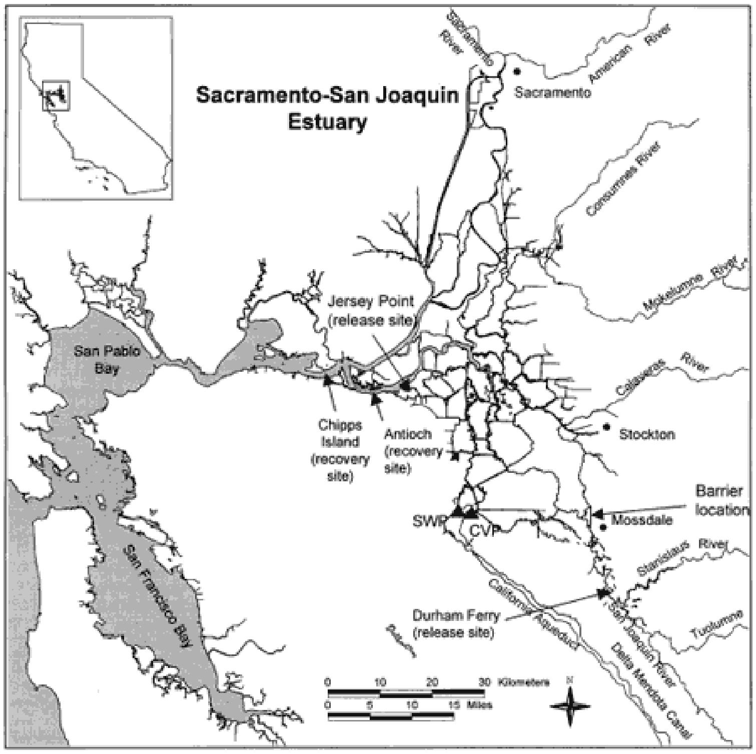


Figure 3-14. Map of the Sacramento-San Joaquin River Delta. Release point at Durham Ferry, and farthest downstream detection array at Chipps Island shown (From San Joaquin River Group Authority, www.sjrg.org).

3.5.3 Methods

3.5.3.1 Tagging and data recovery

In 2011, a total of 1,895 juvenile Chinook salmon were tagged and released into the San Joaquin River at Durham Ferry (site A1) by the consulting firm FishBio (Figure 3-14). Tagged Chinook were released in 4 groups with each release group occurring over a non-overlapping five day period from May 17 to June 19 2011 (Table 3-5). Each tag was uniquely identifiable by frequency and code.

Table 3-5. Release dates and total numbers for tagged juvenile Chinook salmon, Sacramento-San Joaquin Delta, 2011. Fish were released at Durham Ferry.

Release Group	Number Released	Dates Released
1	475	May 17 – May 21
2	473	May 22 – May 26
3	473	June 7 – June 11
4	474	June 15 – June 19

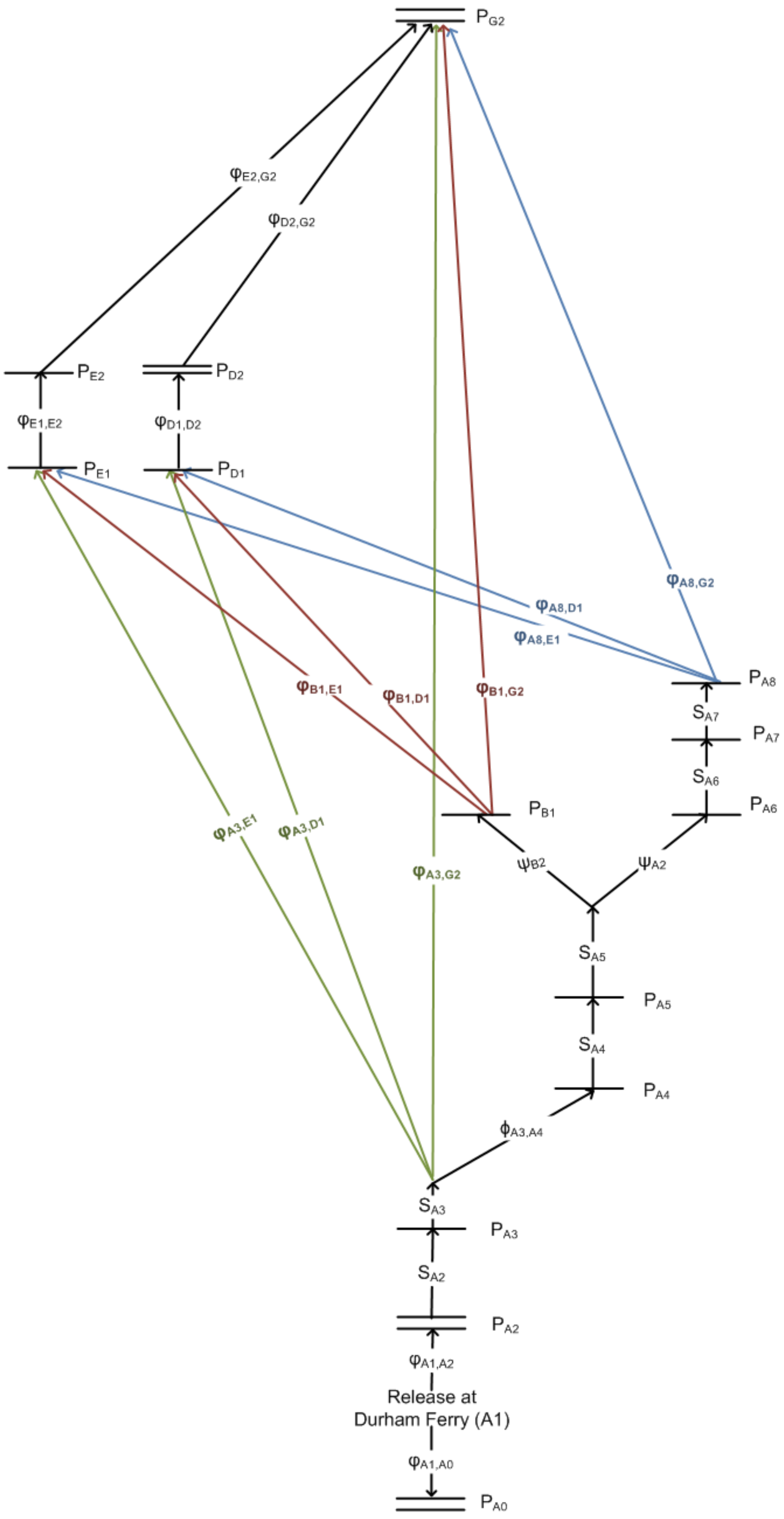
Detection arrays were located throughout the Delta, were maintained by several federal and state agencies, and operated during the length of the study. Detection arrays located in the San Joaquin River are herein numbered A0 through A8. As stated above, array A0 was located upstream of the release point at Durham Ferry. Array A2 was located just downstream of the release point. Arrays A3 through A6 were located upstream and downstream, respectively, of the entrances to Paradise Cut and Old River, respectively. Arrays A7 and A8 were both located near Stockton, with A8 the farthest downstream array in the San Joaquin River before its junction with the Sacramento River. Array B1 was located in Old River just downstream from its split with the San Joaquin River. Arrays D1 and D2 were at the SWP, and E1 and E2 were at the CVP. Array G2 was located at Chipps Island and was the farthest downstream array in the study area. Arrays A0, A2, A6, B1, D1, D2, and G2 were dual arrays, meaning that at each of these sites there were two independent arrays at which tagged fish could be detected.

Data were screened by the USGS-CRRL lab in Cook, WA and again by University of Washington staff to remove invalid detections (SJRGA 2013). Data were then screened by UW staff to differentiate between detections of tagged live juvenile salmon and detections of predatory fishes, and detections of predators were removed. Additionally, for this analysis detections at the dual arrays A6, B1, and D1 were pooled and treated as if detected at a single array. Finally, in order to assign individual tagged fish to a route within the survival model, subsequent detections of tagged fish upstream of earlier detections were removed.

3.5.3.2 Data analysis

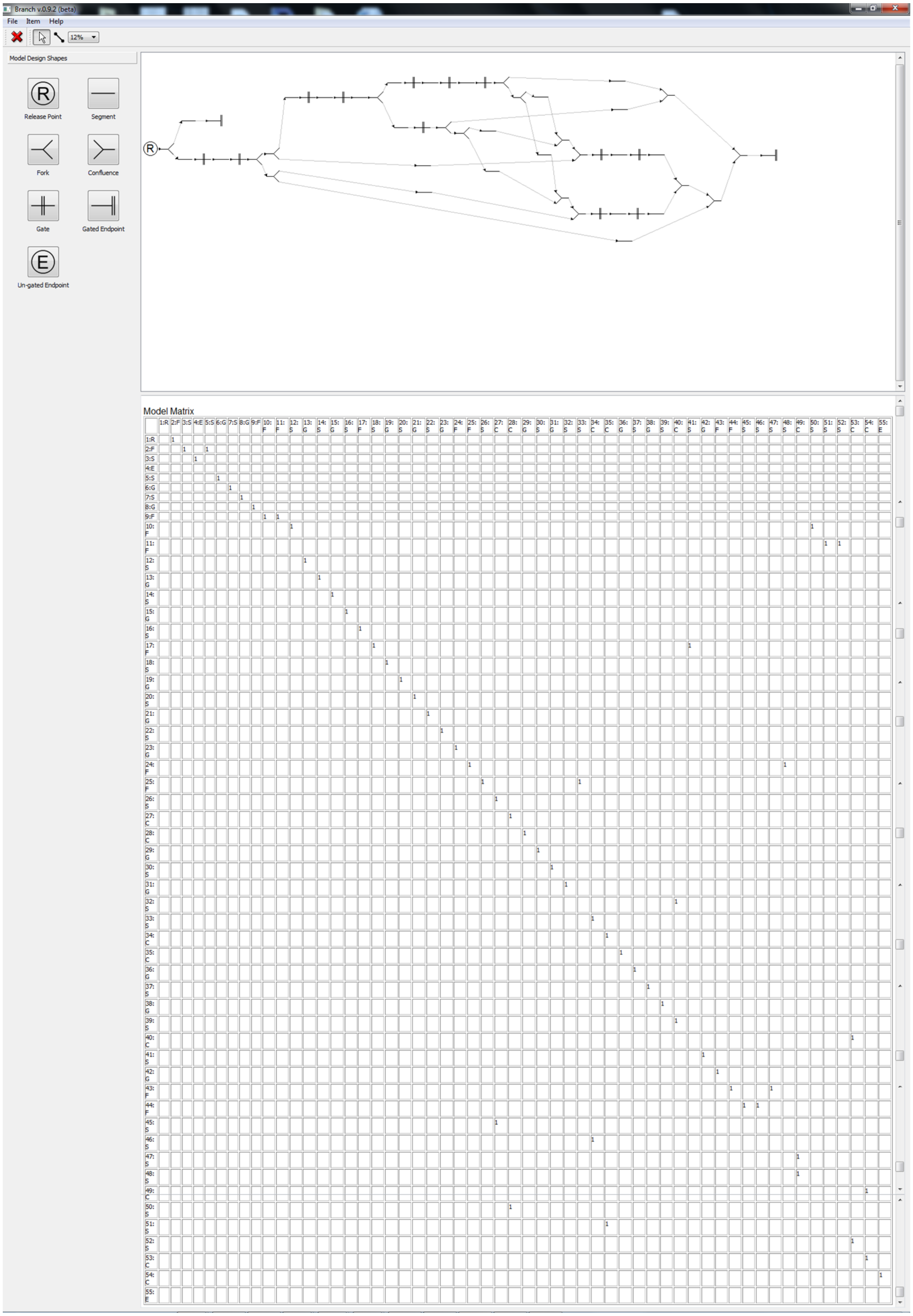
A multistate, product-multinomial model was created in program BRANCH and used to estimate survival and migration parameters for juvenile Chinook salmon through the Delta in 2011 (Figure 3-15). The model was similar to ones developed by Buchanan and Skalski (2010) and Perry et. al. (2010), although somewhat simplified. The model states represented different routes through the Delta, with detection arrays A2 through A8 in the San Joaquin River, array B1 in Old River, D1 and D2 at SWP, and E1 and E2 at CVP. All model states converge at array G2 at Chipps Island (Figure 3-15). Additionally, the model contains transition parameters representing migration routes through Paradise Cut to either SWP or CVP (ϕ_{A3D1} and ϕ_{A3E1}), from the head of Old River through the South Delta to Chipps Island (ϕ_{B1G2}), and from the San Joaquin River through Turner Cut and the Delta to either SWP or CVP (ϕ_{A8D1} and ϕ_{A8E1}; Figure 3-15). All parameters shown in Figure 3-15 are estimable. The parameters labeled ϕ_{ij} represent joint probability of migration and survival from state *i* to state *j*. The GUI input for program BRANCH (Figure 3-16) represents each survival and migration parameter, estimable or not. Program BRANCH then determined which of these parameters to amalgamate into estimable joint probabilities (Figure 3-15). The model is also described by a product-multinomial likelihood equation (Figure 3-17).

Additional modifications to the model were made based on features of sufficient statistics from the 2011 Chinook data. The parameter ϕ_{A3G2} was fixed to 0, as no tagged Chinook were detected migrating via this route. Also, the detection probabilities at arrays A0b, E1, E2, and A8 were fixed at 1. Data were then fit to this final model and parameters estimated via numeric methods in Program USER. Overall Delta survival and transition probabilities for several routes through the Delta were recovered as functions of these estimated parameters.



$\phi_{A3,A4}$

Figure 3-15. Multi-state model used to estimate parameters for Delta Chinook salmon, 2011.



Model Matrix

	1:R	2:F	3:S	4:E	5:S	6:G	7:S	8:G	9:F	10:F	11:F	12:S	13:G	14:S	15:G	16:S	17:F	18:S	19:G	20:S	21:G	22:S	23:G	24:F	25:F	26:S	27:C	28:C	29:G	30:S	31:G	32:S	33:S	34:C	35:C	36:G	37:S	38:G	39:S	40:C	41:S	42:G	43:F	44:F	45:S	46:S	47:S	48:S	49:C	50:S	51:S	52:S	53:C	54:C	55:E
1:R		1																																																					
2:F			1		1																																																		
3:S				1																																																			
4:E																																																							
5:S						1																																																	
6:G							1																																																
7:S								1																																															
8:G									1																																														
9:F										1	1																																												
10:F												1																																						1					
11:F																																																			1	1			
12:S													1																																										
13:G														1																																									
14:S															1																																								
15:G																1																																							
16:S																	1																																						
17:F																		1																							1														
18:S																			1																																				
19:G																				1																																			
20:S																					1																																		
21:G																						1																																	
22:S																							1																																
23:G																								1																															
24:F																									1																							1							
25:F																										1							1																						
26:S																											1																												
27:C																												1																											
28:C																													1																										
29:G																														1																									
30:S																															1																								
31:G																																1																							
32:S																																								1															
33:S																																		1																					
34:C																																			1																				
35:C																																				1																			
36:G																																					1																		
37:S																																						1																	
38:G																																							1																
39:S																																								1															
40:C																																																					1		
41:S																																										1													
42:G																																											1												
43:F																																												1			1								
44:F																																													1	1									
45:S																											1																												
46:S																																		1																					
47:S																																																	1						
48:S																																																	1						
49:C																																																						1	
50:S																												1																											
51:S																																			1																				
52:S																																																					1		
53:C																																																						1	
54:C																																																							1
55:E																																																							

Figure 3-16. Screen capture of user-specified study design in Program BRANCH.

$$
L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}
[\phi_{A1A0}P_{A0a}P_{A0b}]^{a_{A0ab}}\times[\phi_{A1A0}P_{A0a}(1-P_{A0b})]^{a_{A0a}}\times[\phi_{A1A0}(1-P_{A0a})P_{A0b}]^{a_{A0b}}\times[\phi_{A1A2}P_{A2a}P_{A2b}]^{a_{A2ab}}\times[\phi_{A1A2}P_{A2a}(1-P_{A2b})]^{a_{A2a}}\\
\times[\phi_{A1A2}(1-P_{A2a})P_{A2b}]^{a_{A2b}}\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}P_{A3}]^{a_{A3}}\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})\phi_{A3A4}P_{A4}]^{a_{A4}}\\
\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})\phi_{A3A4}(1-P_{A4})S_{A4A5}P_{A5}]^{a_{A5}}\\
\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}P_{A6}]^{a_{A6}}\\
\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}P_{A7}]^{a_{A7}}\\
\times[\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}P_{A8}]^{a_{A8}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))P_{D1}\end{array}\right]^{a_{D1}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}P_{D2a}P_{D2b}\end{array}\right]^{a_{D2ab}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}P_{D2a}(1-P_{D2b})\end{array}\right]^{a_{D2a}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}(1-P_{D2a})P_{D2b}\end{array}\right]^{a_{D2b}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3E1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}))P_{E1}\end{array}\right]^{a_{E1}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3E1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}))(1-P_{E1})S_{E1E2}P_{E2}\end{array}\right]^{a_{E2}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2})))P_{G2a}P_{G2b}\end{array}\right]^{a_{G2ab}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2})))P_{G2a}(1-P_{G2b})\end{array}\right]^{a_{G2a}}\\
\times\left[\begin{array}{l}\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})S_{A2A3}(1-P_{A3})(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2})))(1-P_{G2a})P_{G2b}\end{array}\right]^{a_{G2b}}\\
\times\left[\begin{array}{l}1-\phi_{A1A0}-\phi_{A1A2}+\phi_{A1A0}(1-P_{A0a})(1-P_{A0b})+\phi_{A1A2}(1-P_{A2a})(1-P_{A2b})(1-S_{A2A3}+S_{A2A3}(1-P_{A3})(1-\phi_{A3A4}-\phi_{A3D1}\\
-\phi_{A3E1}+\phi_{A3D1}(1-P_{D1})(1-S_{D1D2}+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))\\
+\phi_{A3E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{A3A4}(1-P_{A4})(1-S_{A4A5}\\
+S_{A4A5}(1-P_{A5})(1-\phi_{A5A6}-\phi_{A5B1}+\phi_{A5B1}(1-P_{B1})(1-\phi_{B1D1}-\phi_{B1E1}-\phi_{B1G2}+\phi_{B1G2}(1-P_{G2a})(1-P_{G2b})\\
+\phi_{B1E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{B1D1}(1-P_{D1})(1-S_{D1D2}\\
+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b}))))+\phi_{A5A6}(1-P_{A6})(1-S_{A6A7}+S_{A6A7}(1-P_{A7})(1\\
-S_{A7A8}+S_{A7A8}(1-P_{A8})(1-\phi_{A8G2}-\phi_{A8D1}-\phi_{A8E1}+\phi_{A8G2}(1-P_{G2a})(1-P_{G2b})+\phi_{A8D1}(1-P_{D1})(1-S_{D1D2}\\
+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{A8E1}(1-P_{E1})(1-S_{E1E2}\\
+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))))))))\end{array}\right]^{a_0}
\end{array}\right\}
$$

Figure 3-17. Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}
\left[S_{A2A3}P_{A3}\right]^{a_{A3|A2}}\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}P_{A4}\right]^{a_{A4|A2}}\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}P_{A5}\right]^{a_{A5|A2}}\\
\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)\phi_{A5A6}P_{A6}\right]^{a_{A6|A2}}\\
\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}P_{A7}\right]^{a_{A7|A2}}\\
\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}P_{A8}\right]^{a_{A8|A2}}\\
\times\left[S_{A2A3}\left(1-P_{A3}\right)\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)\phi_{A5B1}P_{B1}\right]^{a_{B1|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1})+\phi_{A3D1})P_{D1}\end{array}\right]^{a_{D1|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1})+\phi_{A3D1})\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\end{array}\right]^{a_{D2a|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1})+\phi_{A3D1})\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\end{array}\right]^{a_{D2b|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1})+\phi_{A3D1})\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\end{array}\right]^{a_{D2ab|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1E1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1})+\phi_{A3E1})P_{E1}\end{array}\right]^{a_{E1|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1E1}\\
+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1})+\phi_{A3E1})\left(1-P_{E1}\right)S_{E1E2}P_{E2}\end{array}\right]^{a_{E2|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A3E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1G2}+\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8G2}\\
+\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})))P_{G2a}\left(1-P_{G2b}\right)\end{array}\right]^{a_{G2a|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A3E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1G2}+\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8G2}\\
+\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})))\left(1-P_{G2a}\right)P_{G2b}\end{array}\right]^{a_{G2b|A2}}\\
\times\left[\begin{array}{l}S_{A2A3}\left(1-P_{A3}\right)(\phi_{A3D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A3E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{A3A4}\left(1-P_{A4}\right)S_{A4A5}\left(1-P_{A5}\right)(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1G2}+\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8G2}\\
+\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2})))P_{G2a}P_{G2b}\end{array}\right]^{a_{G2a|A2}}\\
\times\left[\begin{array}{l}1-S_{A2A3}+S_{A2A3}\left(1-P_{A3}\right)(1-\phi_{A3D1}-\phi_{A3E1}-\phi_{A3A4}+\phi_{A3D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}\\
+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))+\phi_{A3E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{D2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-\phi_{G2b}\right)))\\
+\phi_{A3A4}\left(1-P_{A4}\right)(1-S_{A4A5}+S_{A4A5}\left(1-P_{A5}\right)(1-\phi_{A5B1}-\phi_{A5A6}+\phi_{A5B1}\left(1-P_{B1}\right)(1-\phi_{B1G2}-\phi_{B1D1}-\phi_{B1E1}\\
+\phi_{B1G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)+\phi_{B1E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\
+\phi_{B1D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))))+\phi_{A5A6}\left(1-P_{A6}\right)(1\\
-S_{A6A7}+S_{A6A7}\left(1-P_{A7}\right)(1-S_{A7A8}+S_{A7A8}\left(1-P_{A8}\right)(1-\phi_{A8G2}-\phi_{A8D1}-\phi_{A8E1}+\phi_{A8G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)\\
+\phi_{A8D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\
+\phi_{A8E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))))))))))\end{array}\right]^{a_{0|A2}}
\end{array}\right\}$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$
L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}
[\phi_{A3A4}P_{A4}]^{a_{A4|A3}}\times[\phi_{A3A4}(1-P_{A4})S_{A4A5}P_{A5}]^{a_{A5|A3}}\times[\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}P_{A6}]^{a_{A6|A3}}\\
\times[\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}P_{A7}]^{a_{A7|A3}}\\
\times[\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}P_{A8}]^{a_{A8|A3}}\times[\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})\phi_{A5B1}P_{B1}]^{a_{B1|A3}}\\
\times[(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))P_{D1}]^{a_{D1|A3}}\\
\times[(\phi_{A3E1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}))P_{E1}]^{a_{E1|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3E1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}))(1-P_{E1})S_{E1E2}P_{E2}
\end{array}\right]^{a_{E2|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}P_{D2a}(1-P_{D2b})
\end{array}\right]^{a_{D2a|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}(1-P_{D2a})P_{D2b}
\end{array}\right]^{a_{D2b|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}\\
+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}))(1-P_{D1})S_{D1D2}P_{D2a}P_{D2b}
\end{array}\right]^{a_{D2ab|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2}\\
+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})))P_{G2a}(1-P_{G2b})
\end{array}\right]^{a_{G2a|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2}\\
+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})))(1-P_{G2a})P_{G2b}
\end{array}\right]^{a_{G2b|A3}}\\
\times\left[\begin{array}{l}
(\phi_{A3D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A3E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{A3A4}(1-P_{A4})S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1G2}\\
+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8G2}\\
+\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2})))P_{G2a}P_{G2b}
\end{array}\right]^{a_{G2ab|A3}}\\
\times\left[\begin{array}{l}
1-\phi_{A3A4}-\phi_{A3D1}-\phi_{A3E1}+\phi_{A3E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))\\
+\phi_{A3D1}(1-P_{D1})(1-S_{D1D2}+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))\\
+\phi_{A3A4}(1-P_{A4})(1-S_{A4A5}+S_{A4A5}(1-P_{A5})(1-\phi_{A5B1}-\phi_{A5A6}+\phi_{A5B1}(1-P_{B1})(1-\phi_{B1E1}-\phi_{B1D1}-\phi_{B1G2}\\
+\phi_{B1E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{B1D1}(1-P_{D1})(1-S_{D1D2}\\
+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{B1G2}(1-P_{G2a})(1-P_{G2b}))+\phi_{A5A6}(1-P_{A6})(1\\
-S_{A6A7}+S_{A6A7}(1-P_{A7})(1-S_{A7A8}+S_{A7A8}(1-P_{A8})(1-\phi_{A8G2}-\phi_{A8D1}-\phi_{A8E1}+\phi_{A8G2}(1-P_{G2a})(1-P_{G2b})\\
+\phi_{A8D1}(1-P_{D1})(1-S_{D1D2}+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))\\
+\phi_{A8E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b}))))))))
\end{array}\right]^{a_{0|A3}}
\end{array}\right\}
$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}
\left[S_{A4A5}P_{A5}\right]^{a_{A5|A4}}\times\left[S_{A4A5}(1-P_{A5})\phi_{A5A6}P_{A6}\right]^{a_{A6|A4}}\times\left[S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}P_{A7}\right]^{a_{A7|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}P_{A8}\right]^{a_{A8|A4}}\times\left[S_{A4A5}(1-P_{A5})\phi_{A5B1}P_{B1}\right]^{a_{B1|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}\right)P_{D1}\right]^{a_{D1|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}\right)P_{E1}\right]^{a_{E1|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1E1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8E1}\right)(1-P_{E1})S_{E1E2}P_{E2}\right]^{a_{E2|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}\right)(1-P_{D1})S_{D1D2}P_{D2a}(1-P_{D2b})\right]^{a_{D2a|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}\right)(1-P_{D1})S_{D1D2}(1-P_{D2a})P_{D2b}\right]^{a_{D2b|A4}}\\
\times\left[S_{A4A5}(1-P_{A5})\left(\phi_{A5B1}(1-P_{B1})\phi_{B1D1}+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})\phi_{A8D1}\right)(1-P_{D1})S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|A4}}\\
\times\left[\begin{array}{l}
S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8G2}))P_{G2a}(1-P_{G2b})
\end{array}\right]^{a_{G2a|A4}}\\
\times\left[\begin{array}{l}
S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8G2}))(1-P_{G2a})P_{G2b}
\end{array}\right]^{a_{G2b|A4}}\\
\times\left[\begin{array}{l}
S_{A4A5}(1-P_{A5})(\phi_{A5B1}(1-P_{B1})(\phi_{B1D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}+\phi_{B1E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}(1-P_{A6})S_{A6A7}(1-P_{A7})S_{A7A8}(1-P_{A8})(\phi_{A8D1}(1-P_{D1})S_{D1D2}(1-P_{D2a})(1-P_{D2b})\phi_{D2G2}\\
+\phi_{A8E1}(1-P_{E1})S_{E1E2}(1-P_{E2})\phi_{E2G2}+\phi_{A8G2}))P_{G2a}P_{G2b}
\end{array}\right]^{a_{G2ab|A4}}\\
\times\left[\begin{array}{l}
1-S_{A4A5}+S_{A4A5}(1-P_{A5})(1-\phi_{A5B1}-\phi_{A5A6}+\phi_{A5B1}(1-P_{B1})(1-\phi_{B1D1}-\phi_{B1E1}-\phi_{B1G2}+\phi_{B1D1}(1-P_{D1})(1-S_{D1D2}\\
+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{B1E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}\\
+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{B1G2}(1-P_{G2a})(1-P_{G2b}))+\phi_{A5A6}(1-P_{A6})(1-S_{A6A7}+S_{A6A7}(1-P_{A7})(1-S_{A7A8}\\
+S_{A7A8}(1-P_{A8})(1-\phi_{A8D1}-\phi_{A8E1}-\phi_{A8G2}+\phi_{A8D1}(1-P_{D1})(1-S_{D1D2}+S_{D1D2}(1-P_{D2a})(1-P_{D2b})(1-\phi_{D2G2}\\
+\phi_{D2G2}(1-P_{G2a})(1-P_{G2b})))+\phi_{A8E1}(1-P_{E1})(1-S_{E1E2}+S_{E1E2}(1-P_{E2})(1-\phi_{E2G2}+\phi_{E2G2}(1-P_{G2a})(1-P_{G2b})))\\
+\phi_{A8G2}(1-P_{G2a})(1-P_{G2b})))))
\end{array}\right]^{a_{0|A4}}
\end{array}\right\}$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$
L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}
\left[\phi_{A5A6}P_{A6}\right]^{a_{A6|A5}}\times\left[\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}P_{A7}\right]^{a_{A7|A5}}\times\left[\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}P_{A8}\right]^{a_{A8|A5}}\times\left[\phi_{A5B1}P_{B1}\right]^{a_{B1|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\right)P_{D1}\right]^{a_{D1|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1E1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}\right)P_{E1}\right]^{a_{E1|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1E1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}\right)\left(1-P_{E1}\right)S_{E1E2}P_{E2}\right]^{a_{E2|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\right)\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\right)\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|A5}}\\
\times\left[\left(\phi_{A5B1}\left(1-P_{B1}\right)\phi_{B1D1}+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\right)\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|A5}}\\
\times\left[\begin{array}{l}
(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2}))P_{G2a}\left(1-P_{G2b}\right)
\end{array}\right]^{a_{G2a|A5}}\\
\times\left[\begin{array}{l}
(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2}))\left(1-P_{G2a}\right)P_{G2b}
\end{array}\right]^{a_{G2b|A5}}\\
\times\left[\begin{array}{l}
(\phi_{A5B1}\left(1-P_{B1}\right)(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\\
+\phi_{B1G2})+\phi_{A5A6}\left(1-P_{A6}\right)S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\
+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2}))P_{G2a}P_{G2b}
\end{array}\right]^{a_{G2ab|A5}}\\
\times\left[\begin{array}{l}
1-\phi_{A5B1}-\phi_{A5A6}+\phi_{A5B1}\left(1-P_{B1}\right)(1-\phi_{B1D1}-\phi_{B1E1}-\phi_{B1G2}+\phi_{B1D1}\left(1-P_{D1}\right)(1-S_{D1D2}\\
+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{B1E1}\left(1-P_{E1}\right)(1-S_{E1E2}\\
+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{B1G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))\\
+\phi_{A5A6}\left(1-P_{A6}\right)(1-S_{A6A7}+S_{A6A7}\left(1-P_{A7}\right)(1-S_{A7A8}+S_{A7A8}\left(1-P_{A8}\right)(1-\phi_{A8D1}-\phi_{A8E1}-\phi_{A8G2}\\
+\phi_{A8D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\
+\phi_{A8E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\
+\phi_{A8G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))))
\end{array}\right]^{a_{0|A5}}
\end{array}\right\}
$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{aligned}
&\left[S_{A6A7}P_{A7}\right]^{a_{A7|A6}}\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}P_{A8}\right]^{a_{A8|A6}}\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}P_{D1}\right]^{a_{D1|A6}}\\
&\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|A6}}\\
&\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|A6}}\\
&\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|A6}}\\
&\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}P_{E1}\right]^{a_{E1|A6}}\times\left[S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}P_{E2}\right]^{a_{E2|A6}}\\
&\times\begin{bmatrix}S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}\left(1-P_{G2b}\right)\end{bmatrix}^{a_{G2a|A6}}\\
&\times\begin{bmatrix}S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})\left(1-P_{G2a}\right)P_{G2b}\end{bmatrix}^{a_{G2b|A6}}\\
&\times\begin{bmatrix}S_{A6A7}\left(1-P_{A7}\right)S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}P_{G2b}\end{bmatrix}^{a_{G2ab|A6}}\\
&\times\begin{bmatrix}1-S_{A6A7}+S_{A6A7}\left(1-P_{A7}\right)(1-S_{A7A8}+S_{A7A8}\left(1-P_{A8}\right)(1-\phi_{A8D1}-\phi_{A8E1}-\phi_{A8G2}+\phi_{A8D1}\left(1-P_{D1}\right)(1-S_{D1D2}\\ +S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{A8E1}\left(1-P_{E1}\right)(1-S_{E1E2}\\ +S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{A8G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\end{bmatrix}^{a_{0|A6}}
\end{aligned}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{aligned}
&\left[S_{A7A8}P_{A8}\right]^{a_{A8|A7}}\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}P_{D1}\right]^{a_{D1|A7}}\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|A7}}\\
&\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|A7}}\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|A7}}\\
&\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}P_{E1}\right]^{a_{E1|A7}}\times\left[S_{A7A8}\left(1-P_{A8}\right)\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}P_{E2}\right]^{a_{E2|A7}}\\
&\times\begin{bmatrix}S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}\left(1-P_{G2b}\right)\end{bmatrix}^{a_{G2a|A7}}\\
&\times\begin{bmatrix}S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})\left(1-P_{G2a}\right)P_{G2b}\end{bmatrix}^{a_{G2b|A7}}\\
&\times\begin{bmatrix}S_{A7A8}\left(1-P_{A8}\right)(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\\ +\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}P_{G2b}\end{bmatrix}^{a_{G2ab|A7}}\\
&\times\begin{bmatrix}1-S_{A7A8}+S_{A7A8}\left(1-P_{A8}\right)(1-\phi_{A8D1}-\phi_{A8E1}-\phi_{A8G2}+\phi_{A8D1}\left(1-P_{D1}\right)(1-S_{D1D2}\\ +S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{A8E1}\left(1-P_{E1}\right)(1-S_{E1E2}\\ +S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{A8G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))\end{bmatrix}^{a_{0|A7}}
\end{aligned}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{aligned}
&\left[\phi_{A8D1}P_{D1}\right]^{a_{D1|A8}}\times\left[\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|A8}}\times\left[\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|A8}}\\
&\times\left[\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|A8}}\times\left[\phi_{A8E1}P_{E1}\right]^{a_{E1|A8}}\times\left[\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}P_{E2}\right]^{a_{E2|A8}}\\
&\times\left[(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|A8}}\\
&\times\left[(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|A8}}\\
&\times\left[(\phi_{A8D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{A8E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{A8G2})P_{G2a}P_{G2b}\right]^{a_{G2ab|A8}}\\
&\times\begin{bmatrix}1-\phi_{A8D1}-\phi_{A8E1}-\phi_{A8G2}+\phi_{A8D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\ +\phi_{A8E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{A8G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)\end{bmatrix}^{a_{0|A8}}
\end{aligned}\right\}$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}\left[\phi_{B1D1}P_{D1}\right]^{a_{D1|B1}}\times\left[\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|B1}}\times\left[\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|B1}}\\ \times\left[\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|B1}}\times\left[\phi_{B1E1}P_{E1}\right]^{a_{E1|B1}}\times\left[\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}P_{E2}\right]^{a_{E2|B1}}\\ \times\left[(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{B1G2})P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|B1}}\\ \times\left[(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{B1G2})\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|B1}}\\ \times\left[(\phi_{B1D1}\left(1-P_{D1}\right)S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}+\phi_{B1E1}\left(1-P_{E1}\right)S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}+\phi_{B1G2})P_{G2a}P_{G2b}\right]^{a_{G2ab|B1}}\\ \times\left[\begin{array}{l}1-\phi_{B1D1}-\phi_{B1E1}-\phi_{B1G2}+\phi_{B1D1}\left(1-P_{D1}\right)(1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))\\ +\phi_{B1E1}\left(1-P_{E1}\right)(1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)))+\phi_{B1G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)\end{array}\right]^{a_{0|B1}}\end{array}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}\left[S_{D1D2}P_{D2a}\left(1-P_{D2b}\right)\right]^{a_{D2a|D1}}\times\left[S_{D1D2}\left(1-P_{D2a}\right)P_{D2b}\right]^{a_{D2b|D1}}\times\left[S_{D1D2}P_{D2a}P_{D2b}\right]^{a_{D2ab|D1}}\\ \times\left[S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|D1}}\times\left[S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|D1}}\\ \times\left[S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)\phi_{D2G2}P_{G2a}P_{G2b}\right]^{a_{G2ab|D1}}\\ \times\left[1-S_{D1D2}+S_{D1D2}\left(1-P_{D2a}\right)\left(1-P_{D2b}\right)(1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))\right]^{a_{0|D1}}\end{array}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\begin{array}{l}\left[S_{E1E2}P_{E2}\right]^{a_{E2|E1}}\times\left[S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|E1}}\times\left[S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|E1}}\\ \times\left[S_{E1E2}\left(1-P_{E2}\right)\phi_{E2G2}P_{G2a}P_{G2b}\right]^{a_{G2ab|E1}}\times\left[1-S_{E1E2}+S_{E1E2}\left(1-P_{E2}\right)(1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right))\right]^{a_{0|E1}}\end{array}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\left[\phi_{D2G2}P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|D2}}\times\left[\phi_{D2G2}\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|D2}}\times\left[\phi_{D2G2}P_{G2a}P_{G2b}\right]^{a_{G2ab|D2}}\times\left[1-\phi_{D2G2}+\phi_{D2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)\right]^{a_{0|D2}}\right\}$$

$$L\left(S_{ij},\phi_{ij},P_i\right)\propto\left\{\left[\phi_{E2G2}P_{G2a}\left(1-P_{G2b}\right)\right]^{a_{G2a|E2}}\times\left[\phi_{E2G2}\left(1-P_{G2a}\right)P_{G2b}\right]^{a_{G2b|E2}}\times\left[\phi_{E2G2}P_{G2a}P_{G2b}\right]^{a_{G2ab|E2}}\times\left[1-\phi_{E2G2}+\phi_{E2G2}\left(1-P_{G2a}\right)\left(1-P_{G2b}\right)\right]^{a_{0|E2}}\right\}$$

Figure 3-17 (continued). Product-multinomial likelihood equation used to estimate parameters for Delta Chinook salmon, 2011.

3.5.4 Results

Overall survival for juvenile Chinook salmon through the San Joaquin-Sacramento Delta in 2011 (from Durham Ferry to Chipps Island) was estimated at 0.018 ($\widehat{SE} = 0.003$). The joint probability of migration and survival from Durham Ferry to Chipps Island was also estimated for three distinct routes: through the San Joaquin River ($\widehat{\Phi_A} = 0.0039$, $\widehat{SE} = 0.0014$); through Old River ($\widehat{\Phi_B} = 0.0136$, $\widehat{SE} = 0.0027$); and through Paradise Cut ($\widehat{\Phi_P} = 0.0001$, $\widehat{SE} < 0.0001$). The probabilities of entrainment into either the San Joaquin River or Old River at the head of Old River near Mossdale were estimable assuming that survival from the head of Old River to array

B1 was equal to survival from the head of Old River to array A6. Under this assumption, the probability of entrainment into Old River (ψ_B) was estimated at 0.413 ($\widehat{SE} = 0.012$), while the probability of entrainment into the San Joaquin River at the head of Old River (ψ_A) was estimated at 0.587 ($\widehat{SE} = 0.012$).

Table 3-6. Performance metrics for Delta Chinook in 2011. S_i are survival probabilities in region *i*; ϕ_i are joint probabilities of migration and survival through region *i*; ψ_i are probabilities of entrainment into route *i* (i=A, B, P, Delta, SD). *A* represents route A through the San Joaquin River; *B* represents route B through Old River, and the San Joaquin Delta or either export facility; *P* represents route P through Paradise Cut, and the San Joaquin Delta or either export facility. *SD* represents the South Delta.

Parameter	Estimate	Estimated Standard Error
S_{Delta}	0.018	0.003
S_A	0.007	0.002
S_B	0.033	0.006
ϕ_P	<0.001	<0.001
S_{A_SD}	0.699	0.013
ϕ_{B_SD}	0.481	0.018
ϕ_{P_SD}	0.002	0.001
ψ_A	0.587	0.012
ψ_B	0.413	0.012

Among juvenile Chinook salmon migrating through the San Joaquin River (Route A), survival from Durham Ferry to Chipps Island was estimated at 0.0066 ($\widehat{SE} = 0.0024$), while survival from Durham Ferry to array A8 near Stockton was estimated at 0.699 ($\widehat{SE} = 0.013$). Among Chinook migrating through the Old River (Route B), survival from Durham Ferry to Chipps Island was estimated at 0.033 ($\widehat{SE} = 0.006$), while the joint probability of migrating and surviving from Durham Ferry to either SWP or CVP among Chinook migrating through Old River was estimated at 0.481 ($\widehat{SE} = 0.018$). Overall South Delta survival (S_{SD}) was defined as the joint probability of migrating through the San Joaquin River and surviving to Stockton (array A8) or migrating and surviving through either Old River or the Paradise cut to either SWP or CVP, and was estimated at 0.611 ($\widehat{SE} = 0.011$).

3.5.5 Discussion

3.5.5.1 Ecology

Survival of juvenile Chinook salmon through the Delta in 2011 was low across all routes, although survival was significantly higher (α=0.05) among Chinook salmon migrating through Old River than among those migrating through San Joaquin River. Survival in the southern portion of the Delta appeared to be much higher than survival overall. However, because Chinook migrating through Old River that did not enter SWP or CVP had no possibility of being detected until Chipps Island, only a transition probability, and not a survival probability, was estimable for the southern Delta in Old River.

Juvenile Chinook salmon were entrained into the San Joaquin River route at a significantly higher rate than into the Old River (α=0.05). Previous analyses (Buchanan and Skalski 2010; SJRGA 2013) have explored the relationship between entrainment at the head of Old River and factors such as flow in the San Joaquin River, flow through the entire Delta, exports from the CVP and SWP, and others. Buchanan and Skalski (2013) for example found that both flow in the San Joaquin River and the operation of a barrier at the head of Old River were associated with an increased probability of fish remaining in the San Joaquin River route.

3.5.5.2 Modeling

Program BRANCH performed as expected in generating a product-multinomial likelihood for tagged Chinook detections in the Delta, and migration and survival estimates for the 2011 Chinook were commensurate with estimates obtained by SJRGA (2013). As the choice of model schematic for this analysis was somewhat more simplified than those in other analyses of the

Delta, some spatial resolution was lost; this was however by design and does not reflect a limitation of the modeling software.

As with the Clear Lake Reservoir sucker analysis, not all estimable transition parameters were calculable from the data. Certain features of the sufficient statistics necessitated fixing transition probabilities to 0 or detection probabilities to 1. Particularly, the probability of transition from array A3 to array G2 at Chipps Island (ϕ_{A3G2}) was fixed at 0. This was necessary even though some tags detected at array A3 were subsequently detected at array G2. The reason for fixing the parameter value at 0 was that each of these tags was also detected at some intermediate array along a different route, and so no tags could be assigned to the direct transition from A3 to G2. This subtlety is an example of the care that must be taken when modeling complex systems like the Delta. An algorithm such as the BRANCH software will correctly identify which of these parameters must be fixed or constrained based on the model sufficient statistics. In contrast, there is a significant potential for human error modeling these complex systems manually.

Chapter 4: Hypothesis Testing

4.1 Introduction

The same 2011 Sacramento-San Joaquin River Delta juvenile Chinook salmon study shown in Chapter 3-5 is used here in order to illustrate the whether Program BRANCH can be useful in a release-recapture study where null hypotheses are statistically tested. During the 2011 salmon smolt migration period, three temporary agricultural barriers were constructed at locations within the Vernalis Adaptive Management Program (VAMP) study area in the Sacramento-San Joaquin River Delta. These barriers were earthen dams constructed with a spillway allowing high flows to pass over the dam and one way culverts allowing water to pass upstream of the dams during flood tide. A barrier was installed in Old River near Tracy, a second barrier was placed in the Middle River, and a third in the Grant Line Canal (Figure 4-1). The start, finish, and duration of construction of the barriers differed slightly (Table 4-1).

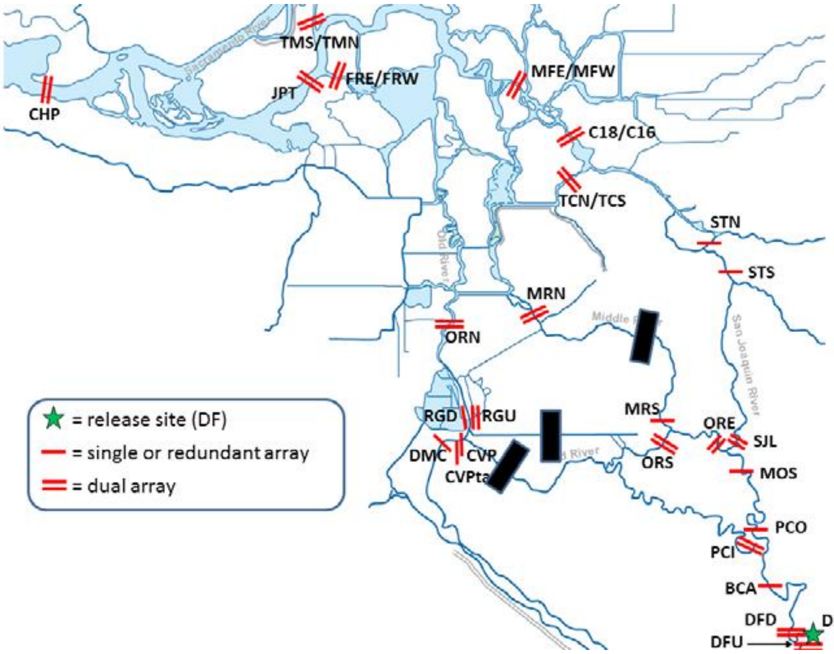


Figure 4-1. Map of study area with temporary agricultural barriers (black cars).

Table 4-1. Temporary barrier installation and removal schedule, 2011. For VAMP analysis purposes, installation date was taken to be 'Started' date, as all three barriers were near closure soon after installation began.

	Installation			**Notched**	**Removal**		
Barrier	Started	Closed	completed		Started	Breached	completed
Old River	05/27/11	06/10/11	06/15/11	09/15/11	10/10/11	10/11/11	10/31/11
Middle River	06/01/11	06/06/11	06/06/11	09/15/11	10/10/11	10/11/11	10/18/11
Grant Line Canal	06/10/11	07/14/11	08/02/11	No Notch	10/17/11	10/19/11	11/04/11

Because each barrier had a spillway and culverts, they were not impassable to salmon smolts; however it is reasonable to posit they may have had some effect on salmon smolt survival, travel time, and route entrainment within the delta. In order to determine whether this was so, separate analysis was carried out on the data from the 1,895 tagged salmon smolts monitored in the 2011 VAMP study. Three hypotheses were tested: (1) salmon smolt survival was lower after barrier installation; (2) route entrainment probability into Old River was lower after barrier installation; and (3) travel time through the Old River corridor was longer after barrier installation.

4.2 Methods

For all three barriers the date marking the edge of the periods before and after barrier installation was taken as the start date of barrier installation (Table 4-1), as all three barriers nearly spanned the waterway very soon after installation began. Data were then grouped by whether the release date for the tagged salmon was before or after the installation date for the barrier in question. Because the installation start dates for the Old River and Middle River barriers both occurred between the second and third release groups for Chinook, the effects of these two barriers could not be measured separately. Analysis to determine possible effects of the Old River and Middle River barriers together compared summary measures of data from tagged smolts in release groups 1 and 2 combined (before Old R./Middle R. barrier installation) with summary measures of data from smolts in release groups 3 and 4 combined (after Old R./Middle R. barrier installation). The installation start date for the Grant Line Canal barrier was between the third and fourth release groups for Chinook, and so its effects could be measured apart from the other two barriers. For analysis of possible effects of the Grant Line Canal barrier, data from smolts in release group 3 (before Grant Line Canal barrier installation) were compared against data from smolts in release group 4 (after Grant Line Canal barrier installation). Although release groups 1 and 2 were also released before installation of the Grant Line Canal barrier, they were omitted from analysis of the Grant Line Canal barrier effects to limit confounding barrier effects with seasonal effects. Hypotheses (1) and (2) regarding survival and route entrainment were tested using parameter estimates from a multi-state statistical release-recapture model, while hypothesis (3) regarding travel time was tested using parameter estimates derived from a linear regression of individual smolt travel time data.

4.2.1 Survival and route entrainment analysis

The multi-state statistical release-recapture model developed for the overall 2011 VAMP study was also used to estimate salmon smolt survival and route entrainment probabilities before and after installation of three agricultural barriers in the study area, with terminology and abbreviations for detection arrays and survival, route entrainment, and transition parameters following that outlined in the Statistical Methods – Survival Model subsection above. The multinomial likelihood model was fit to the observed capture histories using Program USER. The estimated parameters of interest are maximum likelihood estimates and so are asymptotically normally distributed; thus differences in estimated survival and route entrainment parameters before and after barrier installation can be statistically assessed using a one-sided Z-test.

For hypotheses (1) and (2) regarding survival and route entrainment respectively we have as reasonable null and alternative hypotheses the following:

$$H_0: \theta_{before} \leq \theta_{after} \quad \text{vs.} \quad H_a: \theta_{before} > \theta_{after},$$

where θ_{before} is a survival or route entrainment parameter before barrier installation, θ_{after} is the same parameter after barrier installation, and we are interested in whether survival or route entrainment has decreased after a barrier was in place. The logarithmic scale Z-statistic

$$Z(\theta) = \frac{ln(\hat{\theta}_{before}) - ln(\hat{\theta}_{after})}{\sqrt{CV^2(\hat{\theta}_{before}) + CV^2(\hat{\theta}_{after})}}$$

provides a means to test H_0 vs. H_a using the parameters estimated by the multinomial likelihood model described above. Here $CV^2(\hat{\theta})$ is the coefficient of variation squared, $(\hat{\sigma}/\hat{\mu})^2$. Testing at the α level rejects H_0 in favor of H_a if $1 - \Phi(Z(\theta)) < \alpha$.

In all cases sparse detections at some sites prevented fitting the full survival model. Simplified models estimating overall delta survival provided more robust estimates, while still enabling hypothesis testing. The pooled release groups used for the Old River and Middle River barrier analysis (releases 1 and 2 compared with releases 3 and 4) required several simplifications to the likelihood model used for the full study. For the pooled release groups 1 and 2, the model was simplified to estimate delta survival without the Jersey Point (site G1) and False River (site H1) parameters. Additionally, inability to estimate transition probabilities to Middle River

South (site C1) and Old River South (site B2) for pooled release groups 3 and 4 forced model simplification to estimate transition probabilities to sites downstream of these two sites directly from the Old River Entrance site (B1). The data from pooled releases 1 and 2 were fit to the same reduced model as data from pooled releases 3 and 4 so that comparisons of parameters before and after Old River and Middle River barrier installation were more robust (Figure 4-2).

Analysis of effects of the Grant Line Canal barrier required comparing release groups 3 (before) and 4 (after), and fitting these data separately required a different set of simplifications to the VAMP study likelihood model than those for the Old R./Middle R. barrier analysis described above. The model fitting data from release group 3 was simplified by removing parameters associated with sites G1, H1, and A10 (Medford Island site). Additionally, because of sparse detections in the Middle River for release group 3, the model fitting this release was further simplified, removing parameters associated with the Middle River South (C1), Middle River North (C2), and Old River North (B3) sites. Transition probabilities were estimated directly to Chipps Island from the Old River Entrance (B1), Old River South (B2), Clifton Court Forebay (D1), and Central Valley Project trashracks (E1). As above, the same reduced model was fit to both release groups 3 and 4 in order to obtain survival and route entrainment parameter estimates for before and after Grant Line Canal barrier installation (Figure 4-3).

For all release groups detections were pooled across both receivers at the dual array sites at the San Joaquin near Lathrop (A6), the San Joaquin downstream of the Turner Cut Junction (A9), the head of Old River (B1), and the Clifton Court Forebay (D1). Finally, in all cases, transition, survival, and detection probabilities were fixed to 1.0 or 0.0 as appropriate, based on the observed detections.

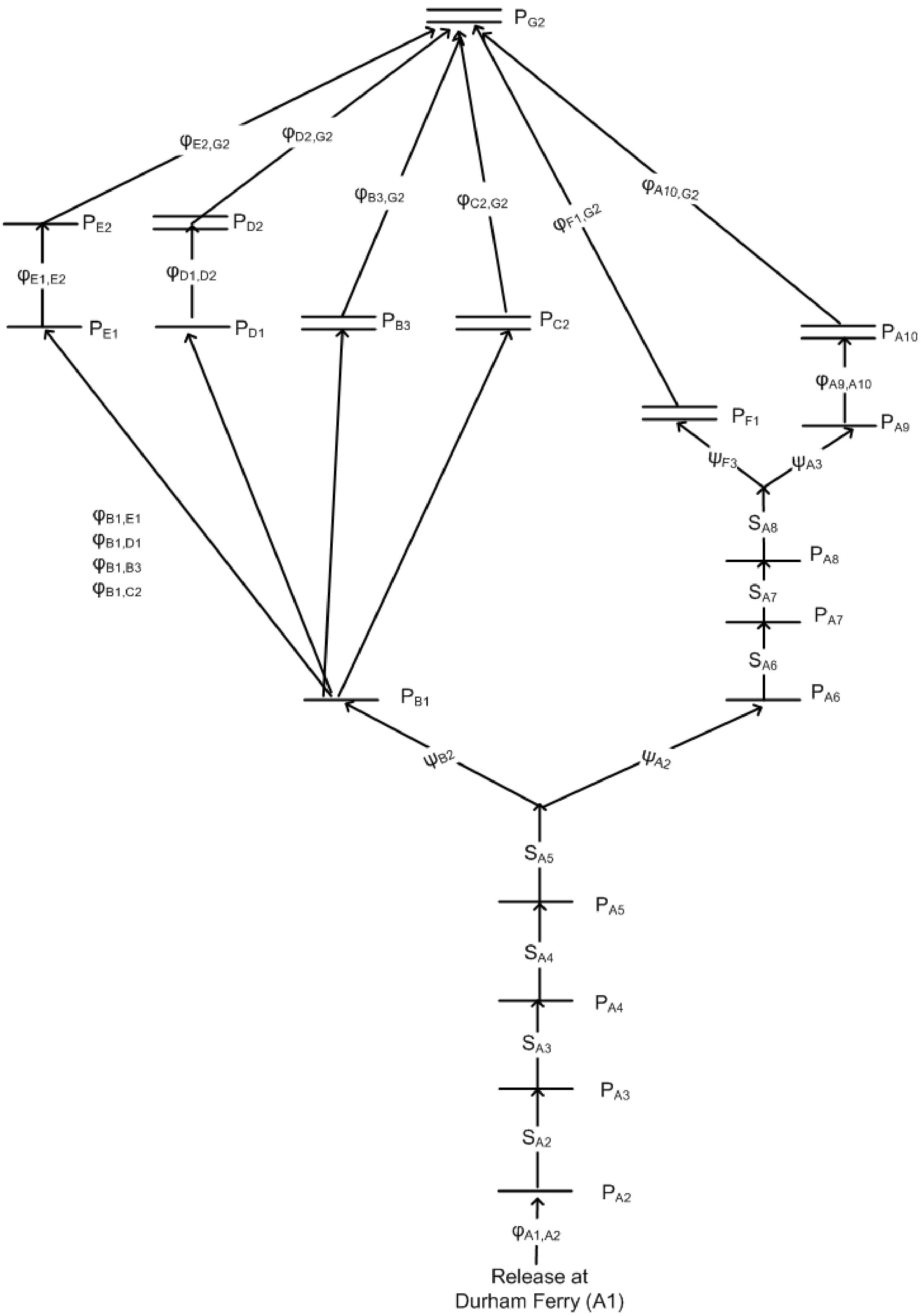


Figure 4-2. Schematic of multinomial likelihood mark-recapture model used to estimate survival and entrainment probabilities for Chinook smolts released before and after installation of the temporary barriers at Old River and Middle River, 2011.

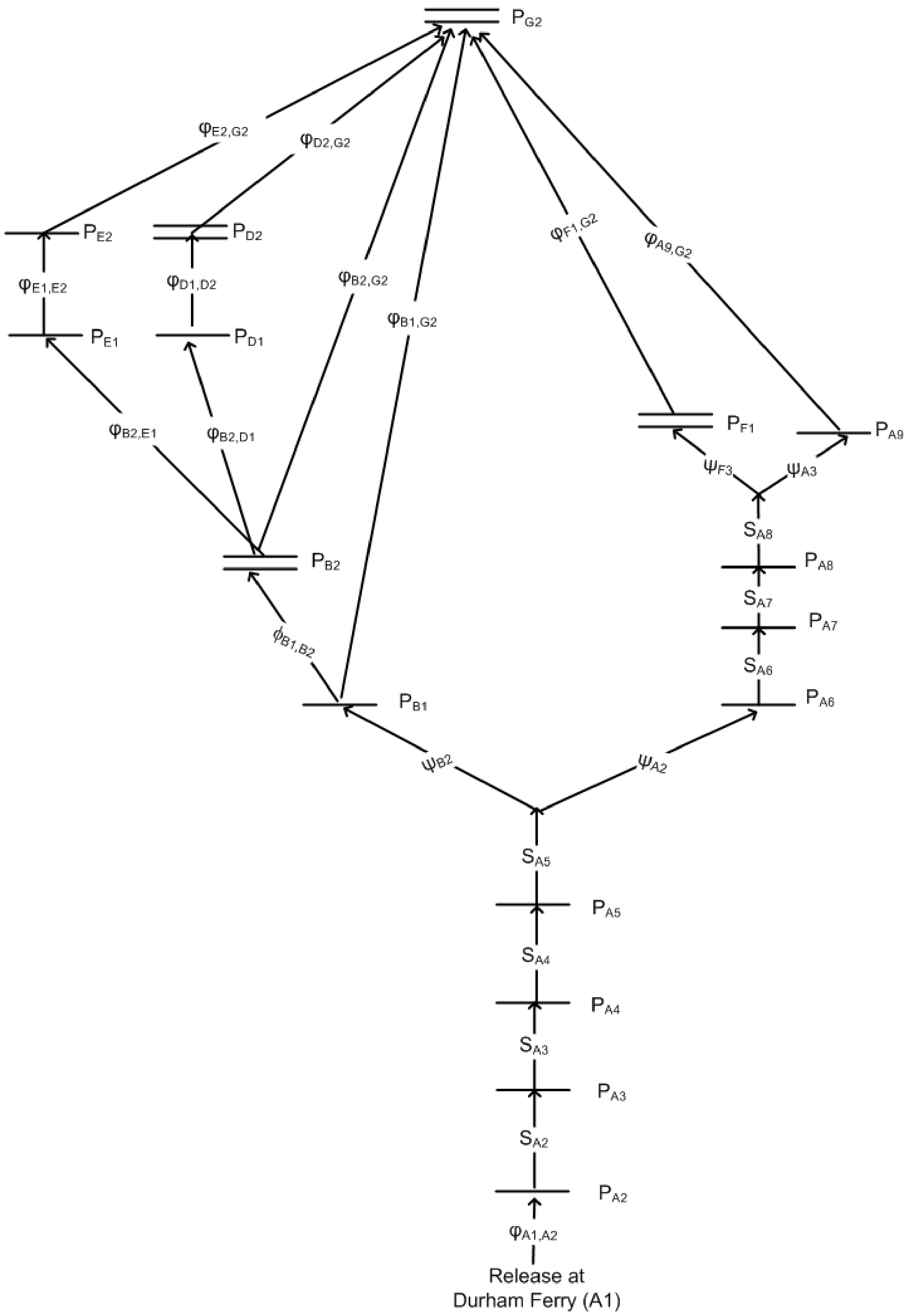


Figure 4-3. Schematic of multinomial likelihood mark-recapture model used to estimate survival and entrainment probabilities for Chinook smolts released before and after installation of the temporary barrier at Grant Line Canal, 2011.

4.2.2 Travel time analysis

In order to test hypothesis (3), that travel time through the Old River corridor was longer after installation of the agricultural barriers, individual smolt travel times were measured from Mossdale (site A5) to several points in the Old River and Middle River Deltas. Specifically, travel time was measured for fish detected at Mossdale and subsequently detected at either the Clifton Court (D1 or D2) or Central Valley (E1 or E2) Projects, or at the Old River North (B3) or Middle River North (C2) sites. Regression analysis was then performed with the logarithm of individual smolt travel times as the response variable, and covariates including a binary variable indicating presence of an agricultural barrier, a river flow measure at Mossdale at the smolt's time of detection there, total project exports on the date of a smolt's detection at Mossdale, and smolt fork length. In order to minimize the effect of both tidal and diurnal (due to pumping and agricultural diversion) fluctuations, river flow was measured as the root mean square of river flow measurements at a site over the 24-hour period centered at the time of a smolt's detection at that site.

Although attempts were made to analyze travel times to each of the four sites listed above, several factors allowed only the analysis of travel times from Mossdale to the Clifton Court and Central Valley Projects. Only 3 smolts of the 1444 detected at Mossdale were subsequently detected at the Middle River North site; additionally no in-river flow data were available from near the site. Although 98 smolts detected at Mossdale were subsequently detected at the Old River North site, only 1 of these left Mossdale after installation of the Old River and Middle River barriers, and none were detected after installation of the Grant Line Canal barrier.

In all, four regression analyses were carried out. Separate analyses were carried out for travel times to the CC and CVP sites. Additionally, effects of the Grant Line Canal agricultural barrier on travel times were treated separately from effects of the other two barriers (because the initiation dates of construction on the Old River and Middle River barriers was so close, the effects of these two barriers could not be treated separately). For analysis of the Grant Line Canal barrier, only data from smolts detected at Mossdale after installation of the Old R. and Middle R. barriers were used, in order to avoid confounding the effects of the different barriers. In each of the four regressions the association between covariates dictated a single covariate model, that is

$$\log(y_i) = \beta_{0j} + \beta_{1j}x_{ij} + \varepsilon_{ij}$$

where y_i is the ith smolt's individual travel time to either the CCF or CVP sites, x_{ij} is the jth covariate (j=1,…,4) for that smolt, ε_{ij} is a stochastic error term and β_{0j}, β_{1j} are regression coefficients for the jth covariate in that regression. Thus, β_{1j} represents the (multiplicative) change in expected travel time from Mossdale to either the CCF or CVP associated with a change in covariate j. For each single covariate regression model an F-test was performed to assess significant association between the response and covariate. AIC selection among models whose covariates were significantly associated with travel times was then used to choose which models represented a good fit to the data, and thus which covariates were most closely associated with travel time.

4.3 Results

4.3.1 Survival and route entrainment analysis

The results presented here compare survival and entrainment metrics from before and after mid-season installation of three agricultural barriers. It is important to note that any differences in these metrics, however statistically significant, cannot be attributed solely to effects of these barriers. Factors such as river flow, water temperature, and water exports from the Central Valley and State Water Projects may also have an effect on smolt survival and route selection. As water temperature tends to rise later in the spring, tagged smolts released after barrier installation likely experienced higher water temperatures during migration. As is shown in the travel time analysis section of this report, both river flow at Mossdale and total project exports show an association with barrier status for all three agricultural barriers as well.

4.3.1.1 Old River/Middle River barriers

Table 4-2. Parameter estimates before and after installation of the Old River and Middle River temporary barriers, 2011. Parameters and standard errors estimated from fitted multi-state release-recapture model with program USER.

	Before OR/MR barrier		**After OR/MR barrier**			
Parameter	Estimate	Std. Err.	Estimate	Std. Err	Z-score	p-value
$\boldsymbol{S_{TotalD}}$	0.0092	0.0033	0.0330	0.0065	-3.15	0.9992
$\boldsymbol{S_{A\_hor\_D}}$	0.0060	0.0035	0.0090	0.0045	-0.53	0.7014
$\boldsymbol{S_{B\_hor\_D}}$	0.0136	0.0061	0.0674	0.0142	-3.25	0.9994
$\boldsymbol{\psi_{B2}}$	0.4227	0.0167	0.4105	0.0181	0.49	0.3109

Table 4-2 shows estimates of parameters used to test hypotheses (1) and (2) regarding survival and route entrainment relating to installation of the Old River and Middle River barriers. S_{TotalD} is an estimate of smolt survival across the entire San Joaquin Delta, $S_{A\_hor\_D}$ is an estimate of survival from the head of Old River to Chipps Island for smolts remaining in the San Joaquin River (route A), and $S_{B\_hor\_D}$ estimates survival from the head of Old River to Chipps Island for smolts entrained into Old River at Mossdale, regardless of subsequent route choice (Old R., Middle R., transport from CCF or CVP combined, route B). ψ_{B2} is an estimate of route entrainment probability into the Old River at its head near Mossdale. All are derived from the release-recapture likelihood model shown in Figure 4-2. P-values show that the null hypothesis cannot be rejected for any of the parameters measured. That is, smolt survival was not significantly lower either in the San Joaquin River channel, the Old River/Middle River Delta, or across the entire San Joaquin Delta, nor was entrainment into the Old River at Mossdale significantly lower, after installation of the Old River and Middle River barriers in 2011.

In fact, testing the converse hypothesis for S_{TotalD} and $S_{B\_hor\_D}$ shows that survival was higher for smolts entering the Old River at Mossdale, and across the entire San Joaquin Delta, after installation of the Old R. and Middle R. barriers ($p<0.01$ for both). Although this result is statistically significant, it is important to remember that correlation is not causation, and that the change in survival may be attributed as much or more to other seasonally-influenced factors (measured or unmeasured).

4.3.1.2 Grant Line Canal barrier

Table 4-3. Parameter estimates before and after installation of the Grant Line Canal temporary barrier, 2011. Parameters and standard errors estimated from fitted multi-state release-recapture model with program USER.

	Before GLC barrier		**After GLC barrier**			
Parameter	Estimate	Std. Err.	Estimate	Std. Err	Z-score	p-value
$\boldsymbol{S_{TotalD}}$	0.0320	0.0091	0.0339	0.0093	-0.15	0.5609
$\boldsymbol{S_{A\_hor\_D}}$	0.0129	0.0074	0.0047	0.0047	0.65	0.1913
$\boldsymbol{S_{B\_hor\_D}}$	0.0641	0.0207	0.0702	0.0195	-0.21	0.5848
$\boldsymbol{\psi_{B2}}$	0.3723	0.0253	0.4461	0.0256	-2.05	0.9797

All parameters listed in Table 4-3 represent the same as in the Old R./Middle R. barrier analysis above; all were derived from the release-recapture model shown in Figure 4-3. P-values show that the null hypothesis cannot be rejected for any of the parameters measured. That is, smolt survival was not significantly lower either in the San Joaquin River channel, the Old River/Middle River Delta, or across the entire San Joaquin Delta, nor was entrainment into the Old River at Mossdale significantly lower, after installation of the Grant Line Canal barrier in 2011. Testing the converse hypothesis for entrainment probability ψ_{B2}, however, shows that entrainment into Old River at Mossdale was significantly higher after installation of the Grant Line Canal Barrier (p=0.02). As with the Old R./Middle R. barrier results, however, we cannot attribute this increase solely to the Grant Line Canal barrier as other variables may be as much or more of a cause.

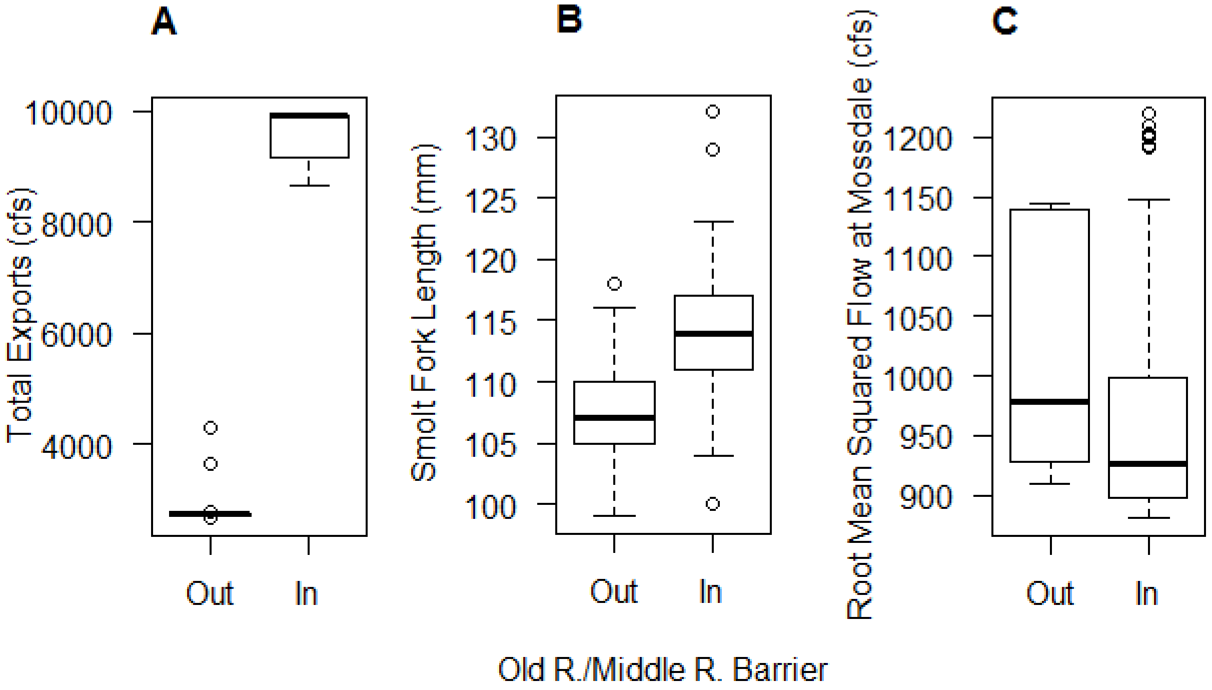
4.3.2 Travel time analysis

Of the 1444 smolts detected at the Mossdale site (A5) during the 2011 VAMP study, 171 smolts (73 before Old R and Middle R. barrier installation, 98 after) were subsequently detected at the Clifton Court Forebay (site D1 or D2), and 124 (48 before and 76 after OR/MR barrier installation) were subsequently detected at the Central Valley Project (sites E1 or E2). Of the 98 fish travelling to the CCF detected at Mossdale after OR/MR barrier installation, 15 were detected before and 83 after the Grant Line Canal barrier was installed. Of the 76 fish travelling to the CVP detected at Mossdale after OR/MR barrier installation, 13 were detected before and 63 after the Grant Line Canal barrier was installed. As mentioned above, sparse data at the Old River North and Middle River North sites prevented comparison of travel times to these sites.

4.3.2.1 Old River/Middle River barriers

Travel time among tagged smolts from Mossdale to the Clifton Court Forebay was shorter after installation of the Old River and Middle River agricultural barriers (harmonic mean time before=20.83 hours, $\widehat{SE}$=0.97 hours, harmonic mean time after=15.03 hours, $\widehat{SE}$=0.47 hours). However, because seasonal trends involving each of the other covariates coincided with the installation of the barriers, the potential for confounding among covariates calls the implications of this association into question. As seen in Figure 4-4, each of the fork length, root mean square of flow at Mossdale, and total project exports covariates shares some association with whether the barriers were installed or not at the time of a fish’s detection. Exports showed a particularly strong association, explained by the marked increase of project exports around June 1, nearly the same time as installation of the agricultural barriers. Regression analysis showed that each covariate on its own had a significant (α=0.05) association with travel time (Table 4-4), and both total exports and barrier status (AIC of 202.8 and 204.2 respectively) showed a better fit to the travel time data than did fork length and river flow (AIC of 225.4 and 229.1, respectively). For the Old R./Middle R. barrier status covariate, $\exp(\beta_0)*24=23.29$ is the expected travel time (hours) from Mossdale to the CCF sites in the absence of the Old R. and Middle R. barriers, while $\exp(\beta_0+\beta_1)*24=15.86$ is the expected travel time in hours after those barriers were present.

Clearly as travel time was shorter after barrier installation, we cannot reject the null hypothesis (3) in favor of the alternative that travel time was longer with the barriers installed. However, the very small p-value suggests that the shorter travel time was associated with the barriers. Total exports showed just as strong of an association with travel time from Mossdale to the CCF sites, with $\exp(\beta_1)=0.9439$ representing the multiplicative change in expected travel time per additional 1000 cfs in total exports. As an example, the average of the daily export values used in the Old R./Middle R. barrier analysis was 6683 cfs. At this export level the regression equation gives an expected travel time from Mossdale to the Clifton Court Forebay of 18.7 hours. The multiplicative effect of β_1 in the regression model means that increasing exports to 7683 cfs would yield expected travel time $(18.7*0.9439)=17.6$ hours; increasing to 8683 gives expected travel time $(18.7*0.9439^2)=16.6$ hours, etc. Given the strength of association between total exports and Old R./Middle R. barrier status (Figure 4-4), it is impossible to separate their effects on travel time.



.

Figure 4-4. (A) Total project exports, (B) fork length, and (C) flow at Mossdale vs. Old River/Middle River barrier status for tagged smolts travelling from Mossdale to the Clifton Court Forebay. Plots produced for smolts travelling from Mossdale to the Central Valley Project sites showed similar results.

Table 4-4. Regression coefficients, p-values, and AIC scores for single regressions modeling fish travel times to the CCF from Mossdale. Models are of the form $\mathbf{y_i=exp(\beta_0+\beta_1 x)+\epsilon}$, where **x** is the covariate of interest and $\mathbf{y_i}$ is travel time (days). Thus, $\mathbf{\beta_1}$ represents the (multiplicative) change in expected travel time (days) associated with a unit change in the corresponding covariate. While all β_1 values were significant (α=0.05), the lower AIC scores of the OR/MR barrier and total exports models suggest that these two were in fact associated with travel times.

Covariate (x)	**β_0, β_1**	**p-value (β_1)**	**AIC**
Old/Middle R. Barriers	-0.030, -0.384	<0.0001	204.2
Flow at Mossdale (kcfs)	-1.041, 0.796	0.0229	229.1
Total Exports (kcfs)	0.136, -0.058	<0.0001	202.8
Smolt Fork Length (mm)	1.724, -0.018	0.0029	225.4

Regression analysis for the effect of the Old River and Middle River barriers on travel time from Mossdale to the CVP sites was similar to that performed for travel to the CCF sites; because the time frame was the same for both, project exports, flow at Mossdale, and smolt fork length covariates showed a similar association with barrier status for smolts travelling to CVP sites as they did for those travelling to CCF sites. Linear regression models found that at the $\alpha=0.05$ level, none of the covariates, taken singly, had a significant effect on travel time from Mossdale to the CVP sites ($p=0.0982$ for river flow at Mossdale, $p>0.30$ for all other covariates).

4.3.2.2 Grant Line Canal barrier

Installation of the Grant Line Canal barrier did not result in any significant effect on travel time among tagged smolts detected at Mossdale and subsequently detected at either the Clifton Court Forebay or the Central Valley Project. Total project exports and flow at Mossdale both again showed an association with barrier status for the Grant Line Canal, although smolt fork length did not (Figure 4-5). Neither smolt fork length, river flow at Mossdale, total project exports nor Grant Line Canal barrier status was significant in explaining travel time from Mossdale to either the CCF or CVP sites ($p>0.30$).

4.4 Discussion

As stated in the Results section above, the null hypothesis cannot be rejected for any of the three hypotheses tested. It was not found that (1) salmon smolt survival was lower after barrier installation; (2) route entrainment probability into Old River was lower after barrier installation; or (3) travel time through the Old River corridor was longer after barrier installation

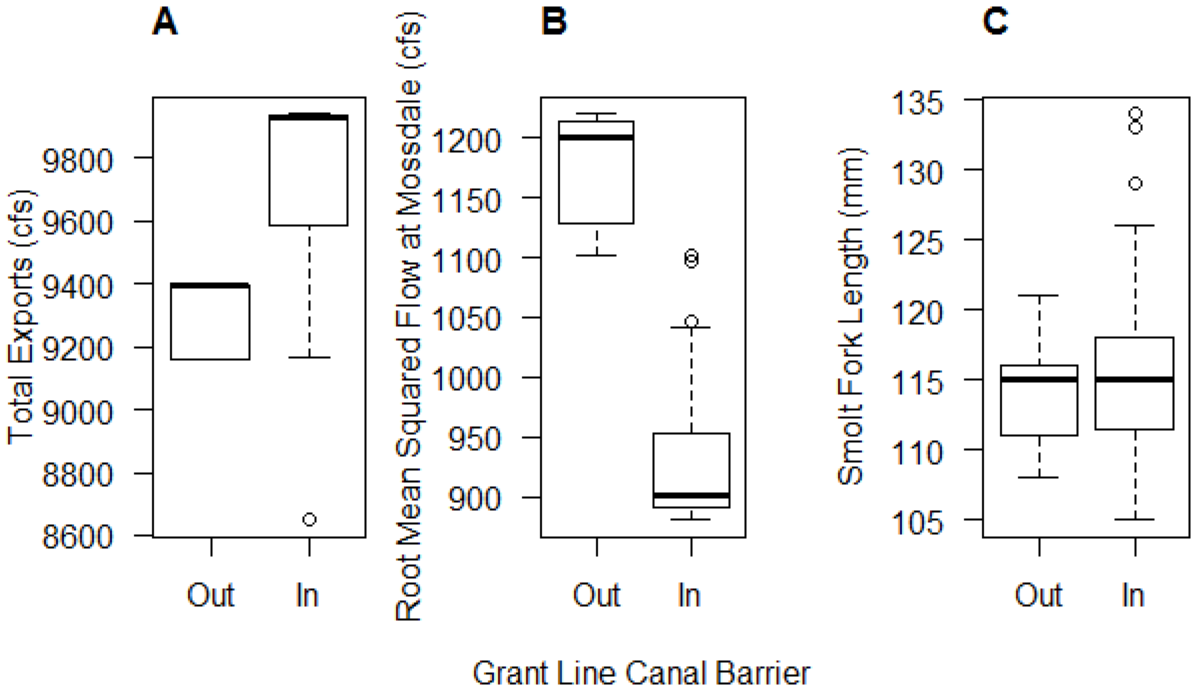


Figure 4-5. (A) Total project exports, (B) flow at Mossdale, and (C) fork length vs. Grant Line Canal barrier status for tagged smolts travelling from Mossdale to the Clifton Court Forebay. Plots produced for smolts travelling from Mossdale to the Central Valley Project sites showed similar results.

for any of the three agricultural barriers installed in 2011. The complexity of the Sacramento-San Joaquin Delta system seems to be one reason for a lack of conclusive evidence regarding these tests.

Both seasonal influences such as temperature and river flow and project exports from the two water export facilities in the Delta varied with the construction and completion of the three agricultural barriers. Any potential effects the barriers may have had on survival, route entrainment, and travel time could well have been masked by the change in the amount of water exported through these facilities that occurred around the same time as the barrier installation, or by seasonal changes occurring around this time. Future studies would be well advised to attempt to separate these effects as much as is possible. While seasonal changes are clearly unavoidable, it may be possible to time barrier installation so as not to coincide with intense seasonal change. Also, while water exports are determined to some extent by demand and cannot be completely controlled, there should be some opportunity to schedule exports in a way that allows separation of these effects from the effects of barrier installation. Additionally, construction of the barriers did not coincide directly with the run timing of juvenile Chinook salmon in the Delta in 2011. Fewer fish released after the barriers were installed, particularly the Grant Line Canal barrier, increased the likelihood that data would be too sparse to draw firm conclusions.

Program BRANCH allows flexible and efficient model specification in complex systems such as the Sacramento-San Joaquin River Delta. As such, it should improve the ability to perform study design analysis for studies of this type before implementation. Investigators have in BRANCH a tool which allows multiple site configurations to be analyzed simply by changing the design schematic input to the program. Testing of the study design in a flexible and inexpensive manner should improve the ability of many such release-recapture studies to draw firm conclusions in the future.

Chapter 5: Conclusions

5.1 Introduction

This thesis has detailed the creation of a method to automatically generate a product-multinomial conditional likelihood equation by using a diagram of a release recapture study design, and given data for such a study have demonstrated that such an automated methodology is capable of further refining the likelihood as necessary to generate biologically meaningful migration and survival parameter estimates. The integration of an interactive GUI as an input device for drawing the study design diagram is an accomplishment which is to my knowledge the first of its kind for such a statistical analysis software package. The result is a user-friendly interface which simplifies and speeds up the process of developing statistically sophisticated multi-state release-recapture models. Consequently, Program BRANCH should expand the use of multi-state models to allow investigators without formal statistical training to design valid and effective release-recapture studies.

As release-recapture studies become increasingly complex, streamlining the modeling and analysis process is increasingly important, saving time and resources and allowing studies to move forward more quickly. Additionally, the ability to quickly map various potential study designs and receive feedback as to the scale and nature of estimable survival and migration parameters before committing resources to implementing the study will be useful to investigators. The ability to use Program BRANCH in both of these capacities was a key goal in my research, and I believe this thesis has demonstrated its capability in this regard.

5.2 Scope and limitations of Program BRANCH

5.2.1 Scope and flexibility

Program BRANCH is designed to accommodate a variety of study design types as demonstrated in Chapters 3 and 4 of this thesis. Release-recapture studies can feature recapture opportunities over time, as in the Clear Lake Reservior sucker analysis (Chapter 3.4), or over space, as in the Chiwawa River basin juvenile Chinook salmon cohort analysis (Chapter 3.3). The interchangeability of dimensions did not pose a problem for Program BRANCH in creating likelihoods and estimating parameters for both of these multi-state models. BRANCH can also be used to model more limited study designs, such as the CJS model used for the single-year Lower Columbia River juvenile Chinook salmon analysis (Chapter 3.2).

Any of the preceding examples detailed in Chapter 3.2, 3.3, and 3.4 could have been analyzed using other existing software such as MARK or M-SURGE (White and Burnham 1999, Choquet et. al. 2004), although the user would need either substantial statistical training or a high degree of prior familiarity with both release-recapture studies and the software package being used. However, even a user with this training and familiarity would not easily be able, using MARK or M-SURGE, to analyze the Sacramento-San Joaquin River Delta juvenile Chinook salmon example from Chapter 3.5 and Chapter 4. This is because existing software is generally designed to analyze multi-state models from release-recapture studies only considering animal movement in a single dimension, resulting in a parameterization which can be represented as a diagram similar to those in Figures 2-8, 3-6, and 3-10. The Sacramento-San Joaquin River Delta study design and others like it represent animal movement in two dimensions; the resulting multi-state models for these types of studies are quite limited in which transitions are possible from one state to another as compared to the classic Arnason-Schwarz type of multi-state model represented by the study designs in Chapters 3.2, 3.3, and 3.4. A comparison of the study design diagrams of Figures 2-8, 3-6, and 3-10 with that in Figures 3-16, 4-2, and 4-3 illustrates this difference: the classic Arnason-Schwartz models in the former case resemble matrices and can be specified by listing dimensions, states, and recapture opportunities as typed input, while the multi-state models in the latter case more closely resemble ‘trees’ whose ’branches’ (possible transitions) are not easily specified by typed input, but are easily drawn as graphical input.

Because existing software packages such as MARK and M-SURGE do not have the graphical component that Program BRANCH has, specifying these more sparse parameterizations for studies such as those in Chapter 3.5 and Chapter 4 is not efficient or practical. In a study design where transitions from one state to another are only occasionally possible, mapping these transitions graphically is a more intuitive and efficient means of model specification. The ease with which BRANCH can accommodate these types of study designs sets it apart from existing software.

5.2.2 Limitations

Achieving the goal of making BRANCH as easy to use as possible necessitated placing certain limitations on the types of models the program is capable of building and analyzing. As mentioned in Chapter 1, the Arnason-Schwartz class of multi-state models assume that the probability of transition from one state and recapture to the next does not depend on previous capture history. Certain other classes of multi-state models operate without this assumption, and these models may contain as many transition parameters from each location to the next as there are prior capture histories up to that location. Program BRANCH is able to construct and analyze only the Arnason-Schwartz type of multi-state model. In a practical sense this may limit the types of release-recapture study designs for which BRANCH is useful. For example, some studies of migrating birds' breeding locations have found, in certain populations, a tendency to return to a prior breeding ground in subsequent years (Lebreton and Pradel 2002, Hestbeck et.al. 1991). If this tendency is not believed to be absolute, two birds may not have the same probability of breeding in area j in year k+1, if one bird has bred in area j in a prior year and another has not.

There may also be limitations on the ability of Program BRANCH to model very complex studies, related to computing power and memory. Particularly with studies similar to the Clear Lake Reservoir sucker analysis, where animals have the opportunity to transition to any other model state between any two recapture opportunities, adding more surveys or recapture opportunities to the study can increase the number of capture histories exponentially, even in a conditional likelihood format. Preliminary studies (J.R. Skalski and A.G. Seaburg, personal communication) have indicated that program memory buffers may be exceeded in Program BRANCH for a two-state model with more than eight recapture opportunities, if animals may transition between states at any time without restriction. This limitation does not seem to be one which is insurmountable as BRANCH is currently designed for proof of principle, and refinements to the computer code should increase the ability to model more complex study designs.

5.3 Directions for future research

The limitation of Program BRANCH to only Arnason-Schwartz type multi-state models can be viewed as an opportunity for future research. This limitation, which assumes that transition probabilities do not depend on previous capture was a choice made in the program design to maximize clarity of purpose and user-friendliness of the program, may not be necessary. There is likely a way to allow the user to relax the assumption of homogeneity of transition rates regardless of prior capture history in a way that neither requires extensive statistical training on the part of the user nor makes the interface unduly complicated. Similarly, other capabilities may be added to BRANCH in the future. The ability to model multiple release groups over time and/or space is one such area for research. If this capability can be added, it should be possible to generalize the types of study designs with movement in two dimensions, such as the Sacramento-San Joaquin River Delta study from Chapter 3.5, into a tiered three-dimensional model in much the same way that the multi-state models from the studies in Chapters 3.2, 3.3, and 3.4 represent a generalization of one-dimensional movement into a two-dimensional model. The greater the flexibility of the software in handling various types of study designs, the larger the number of investigators and studies for which Program BRANCH can be used.

As with most new computer applications, efficiency and computing power are potential areas for future study. The need for the software to handle increasingly complex study designs will only grow as release-recapture studies become larger and more ecosystem-wide. The usefulness of Program BRANCH as a tool for investigators of these studies will directly hinge on its capability to model large, complex study designs.

The capability of a computer program such as BRANCH to quickly and efficiently allow a user to specify complex multi-state models, and to identify biologically meaningful survival and migration parameters, is novel in the field of release-recapture study. Opportunities to refine and expand the software as release-recapture studies continue to become more complex should only add to the achievements I have contributed in this thesis.

Literature Cited

Aars, J., and R. A. Ims. 1999. The effect of habitat corridors on rates of transfer and interbreeding between vole demes. Ecology 80:1648-1655.

Arnason, A. N. 1972. Parameter estimation for mark-recapture experiments on two populations subject to migration and death. Researches in Population Ecology 13:97-113.

Arnason, A. N. 1973. The estimation of population size, migration rates, and survival in a stratified population. Researches in Population Ecology 15:1-8.

Bang-Jensen, J., and G. Gutin. 2007. Digraphs: Theory, Algorithms, and Applications. pp 1-30. Springer-Verlag: Berlin.

Brownie, C., J. E. Hines, J. D. Nichols, K. H. Pollock, and J. B. Hestbeck. 1993. Capture-recapture studies for multiple strata including non-Markovian transitions. Biometrics 49:1173-1187.

Buchanan, R. A., and J. R. Skalski. 2010. Using multistate mark-recapture methods to model adult salmonid migration in an industrialized river. Ecological Modelling 221: 582-589.

Burnham, K. P., D. R. Anderson, G. C. White, C. Brownie, and K. H. Pollock. 1987. Design and analysis methods for fish survival experiments based on release-recapture. American Fisheries Society, Monograph 5, Bethesda, Maryland.

Choquet, R., A. M. Reboulet, R. Pradel, O. Gimenez, and J. D. Lebreton. 2004. M-SURGE: new software specifically designed for multistate capture-recapture models. Animal Biodiversity and Conservation 27:207-215.

Clobert, J. 1995. Capture-recapture and Evolutionary Ecology: a difficult wedding? Journal of Applied Statistics 22:989-1008.

Cormack, R. M. 1964. Estimates of survival from the sighting of marked animals. Biometrika 51:429-438.

Darroch, J. N. 1961. The two-sample capture-recapture census when tagging and sampling are stratified. Biometrika 46:336-351.

Doherty, P. F., E. A. Schreiber, J. D. Nichols, J. E. Hines, W. A. Link, G. A. Schenk, and R. W. Schreiber. 2004. Testing life history predictions in a long-lived seabird: a population matrix approach with improved parameter estimation. Oikos 105:606-618.

Dupuis, J. A. 1995. Bayesian estimation of movement and survival probabilities from capture-recapture data. Biometrika 82:761-772.

Fujiwara, M. and H. Caswell. 2002. Estimating population projection matrices from multi-stage mark-recapture data. Ecology 83:3257-3265.

Healey, M. C. 1991. Life history of Chinook salmon (*Oncorhynchus tshawytscha*), p. 311-393. *In* C. Groot and L. Margolis [ed.] Pacific salmon life histories. University of British Columbia Press, Vancouver, B.C.

Hestbeck, J. B, J. D. Nichols, and R. A. Malecki. 1991. Estimates of movement and site fidelity using mark-resight data of wintering Canada Geese. Ecology 72:523-533.

Jolly, G. M. 1965. Explicit estimates from capture-recapture data with both death and immigration stochastic model. Biometrika 52:225-247.

Lebreton, J. D., K. P. Burnham, J. Clobert, and D. R. Anderson. 1992. Modeling survival and testing biological hypotheses using marked animals: a unified approach with case studies. Ecological Monographs 62:67-118.

Lebreton, J. D., J. E. Hines, R. Pradel, J. D. Nichols, and J. A. Spendelow. 2003. Estimation by capture-recapture of recruitment and dispersal over several sites. Oikos 101:253-264.

Lebreton, J. D., and R. Pradel. 2002. Multistate recapture models: modeling incomplete individual histories. Journal of Applied Statistics 29: 353-369.

Nichols, J. D., J. E. Hines, K. H. Pollock, R. L. Hinz, and W. A. Link. 1994. A methodological framework for estimating breeding proportions and testing biological hypotheses about the cost of reproduction with capture-recapture data. Ecology 75:2052-2065.

Perry, R. W., J. R. Skalski, P. L. Brandes, P. T. Sandstrom, A. P. Klimley, A. Ammann, and B. MacFarlane. 2010. Estimating survival and migration route probabilities of juvenile Chinook salmon in the Sacramento-San Joaquin River Delta. North American Journal of Fisheries Management 30: 142-156.

Pyne, M. I., K. M. Byrne, K. A. Holfelder, L. McManus, M. Buhnerkempe, N. Burch, E. Childers, S. Hamilton, G. Schroeder, and P. F. Doherty. 2010. Survival and breeding transitions for a reintroduced bison population: a multistate approach. Journal of Wildlife Management 74:1463-1471.

Rich, W. H. 1920. Early history and seaward migration of Chinook salmon in the Columbia and Sacramento Rivers. Bulletin of the United States Bureau of Fisheries 37: 1-74.

San Joaquin River Group Authority (SJRGA). 2013. 2011 Technical Report: On implementation and monitoring of the San Joaquin River Agreement and the Vernalis Adaptive Management Plan (VAMP): Prepared by San Joaquin River Group Authority for California Water Resource Control Bd, 338 p. Available http://www.sjrg.org/technicalreport/ (accessed 30 August 2013).

Schwarz, C. J., and A. N. Arnason. 2000. Estimation of age-specific breeding probabilities from capture-recapture data. Biometrics 56:59-64.

Schwarz, C. J., J. F. Schweigert, and A. N. Arnason. 1993. Estimating migration rates using tag-recovery data. Biometrics 49: 177-193.

Seber, G. A. F. 1965. A note on the multiple recapture census. Biometrika 52:249-259.

Skalski, J. R., R. L. Townsend, J. Lady, A. E. Giorgi, J. R. Stevenson, and D. McDonald. 2002. Estimating route-specific passage and survival probabilities as a hydroelectric project from smolt radiotelemetry studies. Canadian Journal of Fisheries and Aquatic Sciences 59:1385-1393.

Skalski, J. R., R. L. Townsend, A. G. Seaburg, J. S. Hughes, M. A. Weiland, G. R. Ploskey, C. M. Woodley, and T. J. Carlson. 2012. Compliance Monitoring of Yearling and Subyearling Chinook Salmon and Juvenile Steelhead Survival and Passage at McNary Dam, 2012. Pacific Northwest National Labratory, Richland, Washington.

White, G. C. and K. P. Burnham. 1999. Program MARK: survival estimation from populations of marked animals. Bird Study 46:120-139.

Appendix A: QtSDK code

```
/****************************************************************************
**
** Copyright (C) 2011 Nokia Corporation and/or its subsidiary(-ies).
** All rights reserved.
** Contact: Nokia Corporation (qt-info@nokia.com)
**
** This file is part of the examples of the Qt Toolkit.
**
** $QT_BEGIN_LICENSE:BSD$
** You may use this file under the terms of the BSD license as follows:
**
** "Redistribution and use in source and binary forms, with or without
** modification, are permitted provided that the following conditions are
** met:
**   * Redistributions of source code must retain the above copyright
**     notice, this list of conditions and the following disclaimer.
**   * Redistributions in binary form must reproduce the above copyright
**     notice, this list of conditions and the following disclaimer in
**     the documentation and/or other materials provided with the
**     distribution.
**   * Neither the name of Nokia Corporation and its Subsidiary(-ies) nor
**     the names of its contributors may be used to endorse or promote
**     products derived from this software without specific prior written
**     permission.
**
** THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
** "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
** LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
** A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
** OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
** SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
** LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
** DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
** THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
** (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
** OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE."
** $QT_END_LICENSE$
**
****************************************************************************/

#include <QtGui>

#include "mainwindow.h"

int main(int argv, char *args[])
{
    Q_INIT_RESOURCE(branch);

    QApplication app(argv, args);
    MainWindow mainWindow;
    mainWindow.setGeometry(100, 100, 800, 700);
    mainWindow.show();

    return app.exec();
}
```

```
// MainWindow for branch app
//
// Create menus and toolbars and set up scene. Handle interactions
// between elements and the creation of the model file
//

#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QMap>

#include "DiagramItem.h"

class DiagramScene;

QT_BEGIN_NAMESPACE
class QAction;
class QToolBox;
class QSpinBox;
class QComboBox;
class QFontComboBox;
class QButtonGroup;
class QLineEdit;
class QGraphicsTextItem;
class QFont;
class QToolButton;
class QAbstractButton;
class QGraphicsView;
class QTextEdit;
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
   MainWindow();

private slots:

    void buttonGroupClicked(int id);
    void deleteItem();
    void pointerGroupClicked(int id);
    void bringToFront();
    void sendToBack();
       void saveModel();

    void itemInserted(DiagramItem *item);
       void lineInserted(DiagramItem *startItem, DiagramItem *endItem);

    void sceneScaleChanged(const QString &scale);
    void itemSelected(QGraphicsItem *item);
    void about();

private:
       QString programName;
       QString programVersion;
```

```
    void createToolBox();
    void createActions();
    void createMenus();
       void createToolbars();
       QWidget *createCellWidget(const QString &text, DiagramItem::DiagramType type);

       DiagramScene *scene;
       QGraphicsView *view;

       QAction *saveAction;
       QAction *exitAction;
    QAction *addAction;
    QAction *deleteAction;

    QAction *toFrontAction;
    QAction *sendBackAction;
    QAction *aboutAction;

    QMenu *fileMenu;
    QMenu *itemMenu;
       QMenu *aboutMenu;

       QToolBar *textToolBar;
       QToolBar *editToolBar;
       QToolBar *pointerToolbar;

       QComboBox *sceneScaleCombo;

       QTextEdit *modelEdit;

       QToolBox *toolBox;
       QButtonGroup *buttonGroup;
       QButtonGroup *pointerTypeGroup;
       QAction *lineAction;

       // Parse diagram items and create model export text
       // update model text window
       void updateModelText();
       void updateAdjMatrix();

       // Keep track of all items on the scene
       QVector<DiagramItem*> diagramItems;
       // keep track of diagram item edges
       QVector<DiagramItem*> adjMatrix;
};

#endif


#include <sstream>

#include <QtGui>
#include <QLabel>
#include <QGraphicsView>
#include <QHBoxLayout>
#include <QButtonGroup>
#include <QToolBox>
```

```
#include <QAbstractButton>
#include <QToolButton>
#include <QToolBar>
#include <QAction>
#include <QMenu>
#include <QMenuBar>
#include <QTextEdit>
#include <QFileDialog>
#include <QComboBox>
#include <QMessageBox>

#include "MainWindow.h"
#include "DiagramItem.h"
#include "DiagramScene.h"
#include "DiagramTextItem.h"
#include "Arrow.h"

const int InsertTextButton = 10;

MainWindow::MainWindow() : programName("Branch"), programVersion("0.9.2")
{
	createActions();
	createToolBox();
	createMenus();

	scene = new DiagramScene(itemMenu, this);
	scene->setSceneRect(QRectF(0, 0, 10000, 5000));
	connect(scene, SIGNAL(itemInserted(DiagramItem*)), this,
SLOT(itemInserted(DiagramItem*)));
	connect(scene, SIGNAL(lineInserted(DiagramItem*, DiagramItem*)), this,
SLOT(lineInserted(DiagramItem*, DiagramItem*)));
	connect(scene, SIGNAL(textInserted(QGraphicsTextItem*)), this,
SLOT(textInserted(QGraphicsTextItem*)));
	connect(scene, SIGNAL(itemSelected(QGraphicsItem*)), this,
SLOT(itemSelected(QGraphicsItem*)));
	createToolbars();

	QVBoxLayout *vlayout = new QVBoxLayout;
	view = new QGraphicsView(scene);
	vlayout->addWidget(view);

	modelEdit = new QTextEdit();
	modelEdit->setReadOnly(true);
	modelEdit->setMinimumHeight(200);
	vlayout->addWidget(modelEdit);

	QHBoxLayout *hlayout = new QHBoxLayout;
	hlayout->addWidget(toolBox);
	hlayout->addLayout(vlayout);

	QWidget *widget = new QWidget;
	widget->setLayout(hlayout);

	setCentralWidget(widget);
	QString title = QString("%1 v.%2 (beta)").arg(programName).arg(programVersion);
	setWindowTitle(title);
	setUnifiedTitleAndToolBarOnMac(true);
}
```

```
void MainWindow::buttonGroupClicked(int id)
{
	QList<QAbstractButton *> buttons = buttonGroup->buttons();
	foreach (QAbstractButton *button, buttons)
	{
		if (buttonGroup->button(id) != button)
			button->setChecked(false);
	}

	scene->setItemType(DiagramItem::DiagramType(id));
	scene->setMode(DiagramScene::InsertItem);
}

void MainWindow::pointerGroupClicked(int)
{
	scene->setMode(DiagramScene::Mode(pointerTypeGroup->checkedId()));
}

void MainWindow::bringToFront()
{
	if (scene->selectedItems().isEmpty())
		return;

	QGraphicsItem *selectedItem = scene->selectedItems().first();
	QList<QGraphicsItem *> overlapItems = selectedItem->collidingItems();

	qreal zValue = 0;
	foreach (QGraphicsItem *item, overlapItems)
	{
		if (item->zValue() >= zValue && item->type() == DiagramItem::Type)
			zValue = item->zValue() + 0.1;
	}
	selectedItem->setZValue(zValue);
}


void MainWindow::sendToBack()
{
	if (scene->selectedItems().isEmpty())
		return;

	QGraphicsItem *selectedItem = scene->selectedItems().first();
	QList<QGraphicsItem *> overlapItems = selectedItem->collidingItems();

	qreal zValue = 0;
	foreach (QGraphicsItem *item, overlapItems)
	{
		if (item->zValue() <= zValue && item->type() == DiagramItem::Type)
			zValue = item->zValue() - 0.1;
	}
	selectedItem->setZValue(zValue);
}

void MainWindow::itemInserted(DiagramItem *item)
{
	pointerTypeGroup->button(int(DiagramScene::MoveItem))->setChecked(true);
	scene->setMode(DiagramScene::Mode(pointerTypeGroup->checkedId()));
```

```
        buttonGroup->button(int(item->diagramType()))->setChecked(false);

        diagramItems.push_back(item);
        updateModelText();
}

void MainWindow::lineInserted(DiagramItem *, DiagramItem *)
{
        updateModelText();
}

void MainWindow::deleteItem()
{
        foreach (QGraphicsItem *item, scene->selectedItems())
        {
                if (item->type() == Arrow::Type) {
                        scene->removeItem(item);
                        Arrow *arrow = qgraphicsitem_cast<Arrow *>(item);
                        arrow->startItem()->removeArrow(arrow);
                        arrow->endItem()->removeArrow(arrow);
                        delete item;
                }
        }

        foreach (QGraphicsItem *item, scene->selectedItems())
        {
                if (item->type() == DiagramItem::Type)
                {
                        qgraphicsitem_cast<DiagramItem *>(item)->removeArrows();
                }
                scene->removeItem(item);

                diagramItems.remove(diagramItems.indexOf(qgraphicsitem_cast<DiagramItem
*>(item)));
                delete item;
        }

        updateModelText();
}

// For each diagram item we can have
//
void MainWindow::updateModelText()
{
        modelEdit->clear();
        updateAdjMatrix();

        const int numItems = diagramItems.size();
        QTextCursor cursor = modelEdit->textCursor();

        QTextCharFormat h1;
        h1.setFont(QFont("Arial", 12, 2));
        QTextCharFormat p;
        p.setFont(QFont("Arial", 9, 1));

        cursor.insertBlock();
        cursor.insertText("Model Matrix", h1);
```

```
	cursor.insertBlock();
	if (numItems == 0)
	{
		cursor.insertText("No items", p);
	}
	else
	{
		QTextTable * table = cursor.insertTable(numItems + 1, numItems + 1);

		for (int i = 0; i < numItems; i++)
		{
			DiagramItem *item = diagramItems[i];
			QString headerLabel = QString("%1:%2").arg(i+1).arg(item-
>getTypeString());

			table->cellAt(i + 1,
0).firstCursorPosition().insertText(headerLabel);
			table->cellAt(0, i +
1).firstCursorPosition().insertText(headerLabel);
		}

		for (int i = 0; i < numItems; i++)
		{
			for (int j = 0; j < numItems; j++)
			{
				const int index = numItems * i + j;
				if (index < adjMatrix.size() && adjMatrix[index])
				{
					//DiagramItem * item = adjMatrix[index];
					int weight = 1;
					table->cellAt(i + 1, j +
1).firstCursorPosition().insertText(QString::number(weight));
				}
			}
		}
		cursor.movePosition(QTextCursor::End);
	}

#if 0
	modelText += QString("%1\t%2\t%3\t%4\n")
		.arg(i++)
		.arg(item->getTypeString())
		.arg(parentId)
		.arg(childId);
			QString modelText;

	modelEdit->setText(modelText);
#endif

}

void MainWindow::updateAdjMatrix()
{
	const int numItems = diagramItems.size();
	adjMatrix.clear();
	adjMatrix.resize(numItems * numItems);
	foreach (DiagramItem *item, diagramItems)
	{
```

```
        const QList<Arrow*> & arrows = item->getArrows();
        foreach (Arrow * arrow, arrows)
        {
            DiagramItem * startItem = arrow->startItem();
            DiagramItem * endItem = arrow->endItem();
            int startId = diagramItems.indexOf(startItem);
            int endId = diagramItems.indexOf(endItem);

            const int index = numItems * startId + endId;
            adjMatrix[index] = item;
        }
    }
}

void MainWindow::saveModel()
{
    QString dir = "";
    QString fileName = QFileDialog::getSaveFileName(this, tr("Save Model File"), dir,
        tr("Text Files (*.txt);; All Files (*.*)"));
    if (!fileName.isEmpty())
    {
        QFile modelFile(fileName);
        if (modelFile.open(QFile::WriteOnly | QFile::Truncate))
        {
            QTextStream out(&modelFile);

            const int numItems = diagramItems.size();
            for (int i = 0; i < numItems; i++)
            {
                if (i > 0)
                    out << ",";
                DiagramItem *item = diagramItems[i];
                out << QString("%2.%1").arg(i+1).arg(item->getTypeString());
            }
            out << "\t" << endl;
            for (int i = 0; i < numItems; i++)
            {
                DiagramItem *item = diagramItems[i];
                out << QString("%2.%1").arg(i+1).arg(item->getTypeString());
                for (int j = 0; j < numItems; j++)
                {
                    out << ",";
                    const int index = numItems * i + j;
                    if (index < adjMatrix.size() && adjMatrix[index])
                    {
                        int weight = 1;
                        out << weight;
                    }
                    else out << 0;
                }
                out << "\t" << endl;
            }


            //out << "{" << endl;
            //out << "\t" << "\"fields\": [";
            //for (int i = 0; i < numItems; i++)
            //{
```

```
                        //      if (i > 0)
                        //              out << ",";
                        //      DiagramItem *item = diagramItems[i];
                        //      out << QString("\"%1:%2\"").arg(i+1).arg(item-
>getTypeString());
                        //}
                        //out << "]," << endl;

                        //out << "\t" << "\"data\": [";
                        //for (int i = 0; i < numItems; i++)
                        //{
                        //      for (int j = 0; j < numItems; j++)
                        //      {
                        //              if (j > 0)
                        //                      out << ",";
                        //              const int index = numItems * i + j;
                        //              if (index < adjMatrix.size() && adjMatrix[index])
                        //              {
                        //                      int weight = 1;
                        //                      out << weight;
                        //              }
                        //      }

                        //}
                        //out << "]" << endl << "}" << endl;
                }
                modelFile.close();
        }
}


void MainWindow::sceneScaleChanged(const QString &scale)
{
        double newScale = scale.left(scale.indexOf(tr("%"))).toDouble() / 100.0;
        QMatrix oldMatrix = view->matrix();
        view->resetMatrix();
        view->translate(oldMatrix.dx(), oldMatrix.dy());
        view->scale(newScale, newScale);
}

void MainWindow::itemSelected(QGraphicsItem *)
{
}

void MainWindow::about()
{
        QString aboutString = QString("%1 v.%2 (%3)\n%4")
                .arg(programName).arg(programVersion)
                .arg(tr("beta")).arg(tr("Interactive model builder."));
        QMessageBox::about(this, tr("About Branch"), aboutString);
}

void MainWindow::createToolBox()
{
        buttonGroup = new QButtonGroup(this);
        buttonGroup->setExclusive(false);
        connect(buttonGroup, SIGNAL(buttonClicked(int)), this,
SLOT(buttonGroupClicked(int)));
```

```
       QGridLayout *layout = new QGridLayout;

       int row = 0;
       layout->addWidget(createCellWidget(tr("Release Point"), DiagramItem::Release),
row, 0);
       layout->addWidget(createCellWidget(tr("Segment"), DiagramItem::Segment), row++,
1);

       layout->addWidget(createCellWidget(tr("Fork"), DiagramItem::Fork), row, 0);
       layout->addWidget(createCellWidget(tr("Confluence"), DiagramItem::Confluence),
row++, 1);

       layout->addWidget(createCellWidget(tr("Gate"), DiagramItem::Gate), row, 0);
       layout->addWidget(createCellWidget(tr("Gated Endpoint"),
DiagramItem::GatedEndPoint), row++, 1);

       layout->addWidget(createCellWidget(tr("Un-gated Endpoint"),
DiagramItem::OpenEndPoint),  row++, 0);

       layout->setRowStretch(row, 10);
       layout->setColumnStretch(2, 10);

       QWidget *itemWidget = new QWidget;
       itemWidget->setLayout(layout);

       toolBox = new QToolBox;
       toolBox->setSizePolicy(QSizePolicy(QSizePolicy::Maximum, QSizePolicy::Ignored));
       toolBox->setMinimumWidth(itemWidget->sizeHint().width());
       toolBox->addItem(itemWidget, tr("Model Design Shapes"));
}

void MainWindow::createActions()
{
       toFrontAction = new QAction(QIcon(":/images/bringtofront.png"), tr("Bring to
&Front"), this);
       toFrontAction->setShortcut(tr("Ctrl+F"));
       toFrontAction->setStatusTip(tr("Bring item to front"));
       connect(toFrontAction, SIGNAL(triggered()), this, SLOT(bringToFront()));

       sendBackAction = new QAction(QIcon(":/images/sendtoback.png"), tr("Send to
&Back"), this);
       sendBackAction->setShortcut(tr("Ctrl+B"));
       sendBackAction->setStatusTip(tr("Send item to back"));
       connect(sendBackAction, SIGNAL(triggered()), this, SLOT(sendToBack()));

       deleteAction = new QAction(QIcon(":/images/delete.png"), tr("&Delete"), this);
       deleteAction->setShortcut(tr("Delete"));
       deleteAction->setStatusTip(tr("Delete item from diagram"));
       connect(deleteAction, SIGNAL(triggered()), this, SLOT(deleteItem()));

       saveAction = new QAction(tr("S&ave model to file"), this);
       saveAction->setShortcuts(QKeySequence::Save);
       saveAction->setStatusTip(tr("Save current model to text file."));
       connect(saveAction, SIGNAL(triggered()), this, SLOT(saveModel()));

       exitAction = new QAction(tr("E&xit"), this);
       exitAction->setShortcuts(QKeySequence::Quit);
       exitAction->setStatusTip(tr("Quit branch"));
```

```
        connect(exitAction, SIGNAL(triggered()), this, SLOT(close()));

        aboutAction = new QAction(tr("A&bout"), this);
        aboutAction->setShortcut(tr("Ctrl+B"));
        connect(aboutAction, SIGNAL(triggered()), this, SLOT(about()));
}

void MainWindow::createMenus()
{
        fileMenu = menuBar()->addMenu(tr("&File"));
        fileMenu->addAction(saveAction);
        fileMenu->addSeparator();
        fileMenu->addAction(exitAction);

        itemMenu = menuBar()->addMenu(tr("&Item"));
        itemMenu->addAction(deleteAction);
        itemMenu->addSeparator();
        itemMenu->addAction(toFrontAction);
        itemMenu->addAction(sendBackAction);

        aboutMenu = menuBar()->addMenu(tr("&Help"));
        aboutMenu->addAction(aboutAction);
}

void MainWindow::createToolbars()
{
        editToolBar = addToolBar(tr("Edit"));
        editToolBar->addAction(deleteAction);

        QToolButton *pointerButton = new QToolButton;
        pointerButton->setCheckable(true);
        pointerButton->setChecked(true);
        pointerButton->setIcon(QIcon(":/images/pointer.png"));
        QToolButton *linePointerButton = new QToolButton;
        linePointerButton->setCheckable(true);
        linePointerButton->setIcon(QIcon(":/images/linepointer.png"));

        pointerTypeGroup = new QButtonGroup(this);
        pointerTypeGroup->addButton(pointerButton, int(DiagramScene::MoveItem));
        pointerTypeGroup->addButton(linePointerButton, int(DiagramScene::InsertLine));
        connect(pointerTypeGroup, SIGNAL(buttonClicked(int)), this,
SLOT(pointerGroupClicked(int)));

        sceneScaleCombo = new QComboBox;
        QStringList scales;
        scales << tr("25%") << tr("50%") << tr("75%") << tr("100%") << tr("125%") <<
tr("150%");
        sceneScaleCombo->addItems(scales);
        sceneScaleCombo->setCurrentIndex(2);
        connect(sceneScaleCombo, SIGNAL(currentIndexChanged(QString)), this,
SLOT(sceneScaleChanged(QString)));

        pointerToolbar = addToolBar(tr("Pointer type"));
        pointerToolbar->addWidget(pointerButton);
        pointerToolbar->addWidget(linePointerButton);
        pointerToolbar->addWidget(sceneScaleCombo);
}
```

```
QWidget *MainWindow::createCellWidget(const QString &text, DiagramItem::DiagramType type)
{
        DiagramItem item(type, itemMenu);
        QIcon icon(item.image());

        QToolButton *button = new QToolButton;
        button->setIcon(icon);
        button->setIconSize(QSize(50, 50));
        button->setCheckable(true);
        buttonGroup->addButton(button, int(type));

        QGridLayout *layout = new QGridLayout;
        layout->addWidget(button, 0, 0, Qt::AlignHCenter);
        layout->addWidget(new QLabel(text), 1, 0, Qt::AlignCenter);

        QWidget *widget = new QWidget;
        widget->setLayout(layout);

        return widget;
}
```

```
#ifndef DIAGRAMITEM_H
#define DIAGRAMITEM_H

#include <QAbstractGraphicsShapeItem>
#include <QList>

QT_BEGIN_NAMESPACE
class QPixmap;
class QGraphicsItem;
class QGraphicsScene;
class QTextEdit;
class QGraphicsSceneMouseEvent;
class QMenu;
class QGraphicsSceneContextMenuEvent;
class QPainter;
class QStyleOptionGraphicsItem;
class QWidget;
class QPolygonF;
QT_END_NAMESPACE

class Arrow;


class DiagramItem : public QAbstractGraphicsShapeItem
{
public:
    enum { Type = UserType + 15 };
    enum DiagramType { Step, Conditional, StartEnd, Io, Release, Fork, Confluence,
Segment, Gate, GatedEndPoint, OpenEndPoint };

    DiagramItem(DiagramType diagramType, QMenu *contextMenu, QGraphicsItem *parent = 0);

        void paint(QPainter * painter, const QStyleOptionGraphicsItem * option = 0,
QWidget * widget = 0);
        QRectF boundingRect () const;
```

```
    void removeArrow(Arrow *arrow);
    void removeArrows();
       const QList<Arrow*> & getArrows() const { return arrows; }
    DiagramType diagramType() const { return myDiagramType; }
    QPolygonF polygon() const;
    void addArrow(Arrow *arrow);
    QPixmap image() ;
    int type() const { return Type;}
       QString getTypeString() const;

       QPointF upstreamPos(int attachPt = 0) const;
       QPointF downstreamPos(int attachPt = 0) const;

protected:
    void contextMenuEvent(QGraphicsSceneContextMenuEvent *event);
    QVariant itemChange(GraphicsItemChange change, const QVariant &value);

private:

    DiagramType myDiagramType;
       QList<QPolygonF> myPolygons;
    QMenu *myContextMenu;
    QList<Arrow *> arrows;
};


#endif

#include <QtGui>
#include <QGraphicsScene>
#include <QMenu>
#include <QEvent>
#include <QGraphicsSceneContextMenuEvent>

#include "DiagramItem.h"
#include "Arrow.h"



DiagramItem::DiagramItem(DiagramType diagramType, QMenu *contextMenu, QGraphicsItem
*parent) : QAbstractGraphicsShapeItem (parent)
{
       myDiagramType = diagramType;
       myContextMenu = contextMenu;

       setFlag(QGraphicsItem::ItemIsMovable, true);
       setFlag(QGraphicsItem::ItemIsSelectable, true);
       setFlag(QGraphicsItem::ItemSendsGeometryChanges, true);
}

void DiagramItem::removeArrow(Arrow *arrow)
{
       int index = arrows.indexOf(arrow);

       if (index != -1)
              arrows.removeAt(index);
}
```

```
void DiagramItem::removeArrows()
{
	foreach (Arrow *arrow, arrows)
	{
		arrow->startItem()->removeArrow(arrow);
		arrow->endItem()->removeArrow(arrow);
		scene()->removeItem(arrow);
		delete arrow;
	}
}

void DiagramItem::addArrow(Arrow *arrow)
{
	arrows.append(arrow);
}

void DiagramItem::paint(QPainter * painter, const QStyleOptionGraphicsItem* , QWidget*)
{
	QFont f("Arial", 120);
	painter->setFont(f);

	switch (myDiagramType)
	{
	case Release:
		painter->drawEllipse(-100, -100, 200, 200);
		painter->drawText(QPointF(-55, 60), "R");
		break;
	case Fork:
		painter->drawLine(-100, 0, 0, 0);
		painter->drawLine(0, 0, 80, -80);
		painter->drawLine(0, 0, 80, 80);
		break;
	case Confluence:
		painter->drawLine(-80, -80, 0, 0);
		painter->drawLine(-80, 80, 0, 0);
		painter->drawLine(0, 0, 100, 0);
		break;
	case Segment:
		painter->drawLine(-100, 0, 100, 0);
		break;
	case Gate:
		painter->drawLine(-100, 0, 100, 0);
		painter->drawLine(-10, -80, -10, 80);
		painter->drawLine(10, -80, 10, 80);
		break;
	case GatedEndPoint:
		painter->drawLine(-100, 0, 80, 0);
		painter->drawLine(80, -80, 80, 80);
		painter->drawLine(100, -80, 100, 80);
		break;
	case OpenEndPoint:
		painter->drawEllipse(-100, -100, 200, 200);
		painter->drawText(QPointF(-55, 60), "E");
		break;
	default:
		break;
	}
```

```
}

QPolygonF DiagramItem::polygon() const
{
	return QPolygonF(boundingRect());
}

QPointF DiagramItem::upstreamPos(int attachPt) const
{
	QPointF pt(0, 0);
	switch (myDiagramType)
	{
	case Confluence:
		if (attachPt == 0)
			pt = QPointF(-80, -80);
		else
			pt = QPointF(-80, 80);
		break;
	default:
		pt = QPointF(-100, 0);
		break;
	}

	return mapToParent(pt);
}

QPointF DiagramItem::downstreamPos(int attachPt) const
{
	QPointF pt(0, 0);
	switch (myDiagramType)
	{
	case Fork:
		if (attachPt == 0)
			pt = QPointF(80, -80);
		else
			pt = QPointF(80, 80);
		break;
	default:
		pt = QPointF(100, 0);
		break;
	}
	return mapToParent(pt);
}

QString DiagramItem::getTypeString() const
{
	switch (myDiagramType)
	{
	case Release:
		return "R";
	case Fork:
		return "F";
	case Confluence:
		return "C";
	case Segment:
		return "S";
	case Gate:
		return "G";
```

```
        case GatedEndPoint:
                return "E";
        case OpenEndPoint:
                return "O";
        default:
                return "X";
        }
}

QRectF DiagramItem::boundingRect() const
{
        switch (myDiagramType)
        {
        default:
                return QRectF(-125, -125, 250, 250);
        }
}

QPixmap DiagramItem::image()
{
        QPixmap pixmap(250, 250);
        pixmap.fill(Qt::transparent);
        QPainter painter(&pixmap);
        painter.setPen(QPen(Qt::black, 8));
        painter.translate(125, 125);
        paint(&painter);

        return pixmap;
}

void DiagramItem::contextMenuEvent(QGraphicsSceneContextMenuEvent *evt)
{
        scene()->clearSelection();
        setSelected(true);
        myContextMenu->exec(evt->screenPos());
}

QVariant DiagramItem::itemChange(GraphicsItemChange change,
        const QVariant &value)
{
        if (change == QGraphicsItem::ItemPositionChange)
        {
                foreach (Arrow *arrow, arrows)
                {
                        arrow->updatePosition();
                }
        }

        return value;
}



// The diagram scene for the model (diagram) items
// Handles mouse press events on the scene, sends
// signal when item added.

#ifndef DIAGRAMSCENE_H
```

```
#define DIAGRAMSCENE_H

#include <QGraphicsScene>
#include "DiagramItem.h"

QT_BEGIN_NAMESPACE
class QGraphicsSceneMouseEvent;
class QMenu;
class QPointF;
class QGraphicsLineItem;
class QFont;
class QGraphicsTextItem;
class QColor;
QT_END_NAMESPACE

class DiagramScene : public QGraphicsScene
{
    Q_OBJECT

public:
    enum Mode { InsertItem, InsertLine, InsertText, MoveItem };

    DiagramScene(QMenu *itemMenu, QObject *parent = 0);
    QFont font() const { return myFont; }
    QColor textColor() const { return myTextColor; }
    QColor itemColor() const { return myItemColor; }
    QColor lineColor() const { return myLineColor; }
    void setLineColor(const QColor &color);

public slots:
    void setMode(Mode mode);
    void setItemType(DiagramItem::DiagramType type);

signals:
    void itemInserted(DiagramItem *item);
    void itemSelected(QGraphicsItem *item);
       void lineInserted(DiagramItem *startItem, DiagramItem *endItem);

protected:
    void mousePressEvent(QGraphicsSceneMouseEvent *mouseEvent);
    void mouseMoveEvent(QGraphicsSceneMouseEvent *mouseEvent);
    void mouseReleaseEvent(QGraphicsSceneMouseEvent *mouseEvent);

private:
    bool isItemChange(int type);

    DiagramItem::DiagramType myItemType;

    QMenu *myItemMenu;
    Mode myMode;
    bool leftButtonDown;
    QPointF startPoint;
    QGraphicsLineItem *line;
    QFont myFont;

    QColor myTextColor;
    QColor myItemColor;
    QColor myLineColor;
```

```
};

#endif


#include <QtGui>
#include <QGraphicsSceneMouseEvent>

#include "DiagramScene.h"
#include "Arrow.h"


DiagramScene::DiagramScene(QMenu *itemMenu, QObject *parent) : QGraphicsScene(parent)
{
	myItemMenu = itemMenu;
	myMode = MoveItem;
	myItemType = DiagramItem::Step;
	line = 0;

	myItemColor = Qt::white;
	myTextColor = Qt::black;
	myLineColor = Qt::black;
}

void DiagramScene::setLineColor(const QColor &color)
{
	myLineColor = color;
	if (isItemChange(Arrow::Type))
	{
		Arrow *item =
			qgraphicsitem_cast<Arrow *>(selectedItems().first());
		item->setColor(myLineColor);
		update();
	}
}

void DiagramScene::setMode(Mode mode)
{
	myMode = mode;
}

void DiagramScene::setItemType(DiagramItem::DiagramType type)
{
	myItemType = type;
}

void DiagramScene::mousePressEvent(QGraphicsSceneMouseEvent *mouseEvent)
{
	if (mouseEvent->button() != Qt::LeftButton)
		return;

	DiagramItem *item;
	switch (myMode)
	{
	case InsertItem:
		item = new DiagramItem(myItemType, myItemMenu);
		addItem(item);
		item->setPos(mouseEvent->scenePos());
```

```
            emit itemInserted(item);
            break;
      case InsertLine:
            line = new QGraphicsLineItem(QLineF(mouseEvent->scenePos(), mouseEvent-
>scenePos()));
            line->setPen(QPen(myLineColor, 2));
            addItem(line);
            break;
      default:
            ;
      }
      QGraphicsScene::mousePressEvent(mouseEvent);
}

void DiagramScene::mouseMoveEvent(QGraphicsSceneMouseEvent *mouseEvent)
{
      if (myMode == InsertLine && line != 0)
      {
            QLineF newLine(line->line().p1(), mouseEvent->scenePos());
            line->setLine(newLine);
      }
      else if (myMode == MoveItem)
      {
            QGraphicsScene::mouseMoveEvent(mouseEvent);
      }
}

void DiagramScene::mouseReleaseEvent(QGraphicsSceneMouseEvent *mouseEvent)
{
      if (line != 0 && myMode == InsertLine)
      {
            QPointF startPoint = line->line().p1();
            QPointF endPoint = line->line().p2();

            QList<QGraphicsItem *> startItems = items(startPoint);
            if (startItems.count() && startItems.first() == line)
                  startItems.removeFirst();

            QList<QGraphicsItem *> endItems = items(endPoint);
            if (endItems.count() && endItems.first() == line)
                  endItems.removeFirst();

            removeItem(line);
            delete line;

            if (startItems.count() > 0 && endItems.count() > 0 && startItems.first()-
>type() == DiagramItem::Type &&
                  endItems.first()->type() == DiagramItem::Type && startItems.first()
!= endItems.first())
            {
                  DiagramItem *startItem = qgraphicsitem_cast<DiagramItem
*>(startItems.first());
                  DiagramItem *endItem = qgraphicsitem_cast<DiagramItem
*>(endItems.first());
                  Arrow *arrow = new Arrow(startItem, endItem);
                  arrow->setColor(myLineColor);
                  startItem->addArrow(arrow);
                  endItem->addArrow(arrow);
```

```
            arrow->setZValue(-1000.0);

            // find attachment points for fork and confluence. If the mouse
            // was released in the top half of the figure, attach to the top
            // branch
            if (startItem->diagramType() == DiagramItem::Fork)
            {
                QRectF boundingRect = startItem->boundingRect();
                QRectF mappedRect = QRectF(startItem-
>mapToScene(boundingRect.topLeft()),
                    QSizeF(boundingRect.width(), boundingRect.height() /
2));
                if (mappedRect.contains(startPoint))
                    arrow->setStartAttachPoint(0);
                else
                    arrow->setStartAttachPoint(1);
            }
            if (endItem->diagramType() == DiagramItem::Confluence)
            {
                QRectF boundingRect = endItem->boundingRect();
                QRectF mappedRect = QRectF(endItem-
>mapToScene(boundingRect.topLeft()),
                    QSizeF(boundingRect.width(), boundingRect.height() /
2));
                if (mappedRect.contains(endPoint))
                    arrow->setEndAttachPoint(0);
                else
                    arrow->setEndAttachPoint(1);
            }

            addItem(arrow);
            arrow->updatePosition();

            emit lineInserted(startItem, endItem);
        }
    }
    line = 0;
    QGraphicsScene::mouseReleaseEvent(mouseEvent);
}

bool DiagramScene::isItemChange(int type)
{
    foreach (QGraphicsItem *item, selectedItems())
    {
        if (item->type() == type)
            return true;
    }
    return false;
}


#ifndef DIAGRAMTEXTITEM_H
#define DIAGRAMTEXTITEM_H

#include <QGraphicsTextItem>
#include <QPen>

QT_BEGIN_NAMESPACE
```

```
class QFocusEvent;
class QGraphicsItem;
class QGraphicsScene;
class QGraphicsSceneMouseEvent;
QT_END_NAMESPACE

class DiagramTextItem : public QGraphicsTextItem
{
    Q_OBJECT

public:
    enum { Type = UserType + 3 };

    DiagramTextItem(QGraphicsItem *parent = 0, QGraphicsScene *scene = 0);

    int type() const
        { return Type; }

signals:
    void lostFocus(DiagramTextItem *item);
    void selectedChange(QGraphicsItem *item);

protected:
    QVariant itemChange(GraphicsItemChange change, const QVariant &value);
    void focusOutEvent(QFocusEvent *event);
    void mouseDoubleClickEvent(QGraphicsSceneMouseEvent *event);
};

#endif
```

```
#include <QtGui>

#include "diagramtextitem.h"
#include "diagramscene.h"

DiagramTextItem::DiagramTextItem(QGraphicsItem *parent, QGraphicsScene *scene)
    : QGraphicsTextItem(parent, scene)
{
    setFlag(QGraphicsItem::ItemIsMovable);
    setFlag(QGraphicsItem::ItemIsSelectable);
}

QVariant DiagramTextItem::itemChange(GraphicsItemChange change,
                     const QVariant &value)
{
    if (change == QGraphicsItem::ItemSelectedHasChanged)
        emit selectedChange(this);
    return value;
}

void DiagramTextItem::focusOutEvent(QFocusEvent *event)
{
    setTextInteractionFlags(Qt::NoTextInteraction);
    emit lostFocus(this);
    QGraphicsTextItem::focusOutEvent(event);
}
```

```
void DiagramTextItem::mouseDoubleClickEvent(QGraphicsSceneMouseEvent *event)
{
    if (textInteractionFlags() == Qt::NoTextInteraction)
        setTextInteractionFlags(Qt::TextEditorInteraction);
    QGraphicsTextItem::mouseDoubleClickEvent(event);
}
```

```
#ifndef ARROW_H
#define ARROW_H

#include <QGraphicsLineItem>

#include "diagramitem.h"

QT_BEGIN_NAMESPACE
class QGraphicsPolygonItem;
class QGraphicsLineItem;
class QGraphicsScene;
class QRectF;
class QGraphicsSceneMouseEvent;
class QPainterPath;
QT_END_NAMESPACE

class Arrow : public QGraphicsLineItem
{
public:
    enum { Type = UserType + 4 };

    Arrow(DiagramItem *startItem, DiagramItem *endItem, QGraphicsItem *parent = 0,
QGraphicsScene *scene = 0);

    int type() const { return Type; }
    QRectF boundingRect() const;
    QPainterPath shape() const;
    void setColor(const QColor &color) { myColor = color; }
    DiagramItem *startItem() const { return myStartItem; }
    DiagramItem *endItem() const { return myEndItem; }

       void setStartAttachPoint(int n) { myStartAttachPoint = n; }
       void setEndAttachPoint(int n) { myEndAttachPoint = n; }

    void updatePosition();

protected:
    void paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget
= 0);

private:
       int myStartAttachPoint; // used for confluences
       int myEndAttachPoint;   // used for forks
    DiagramItem *myStartItem;
    DiagramItem *myEndItem;
    QColor myColor;
    QPolygonF arrowHead;
};
```

```
#endif

#include <QtGui>

#include "Arrow.h"
#include <math.h>

const qreal Pi = 3.14;

Arrow::Arrow(DiagramItem *startItem, DiagramItem *endItem, QGraphicsItem *parent,
QGraphicsScene *scene)
	: QGraphicsLineItem(parent, scene), myStartAttachPoint(0), myEndAttachPoint(0)
{
	myStartItem = startItem;
	myEndItem = endItem;
	setFlag(QGraphicsItem::ItemIsSelectable, true);
	myColor = Qt::black;
	setPen(QPen(myColor, 2, Qt::SolidLine, Qt::RoundCap, Qt::RoundJoin));
}

QRectF Arrow::boundingRect() const
{
	qreal extra = (pen().width() + 20) / 2.0;

	return QRectF(line().p1(), QSizeF(line().p2().x() - line().p1().x(),
		line().p2().y() - line().p1().y()))
		.normalized()
		.adjusted(-extra, -extra, extra, extra);
}

QPainterPath Arrow::shape() const
{
	QPainterPath path = QGraphicsLineItem::shape();
	path.addPolygon(arrowHead);
	return path;
}

void Arrow::updatePosition()
{
	QLineF line(mapFromItem(myStartItem, 0, 0), mapFromItem(myEndItem, 0, 0));
	setLine(line);
}

void Arrow::paint(QPainter *painter, const QStyleOptionGraphicsItem *, QWidget *)
{
	if (myStartItem->collidesWithItem(myEndItem))
		return;

	QPen myPen = pen();
	myPen.setColor(myColor);
	qreal arrowSize = 40;
	painter->setPen(myPen);
	painter->setBrush(myColor);

	setLine(QLineF(myEndItem->upstreamPos(myEndAttachPoint), myStartItem-
>downstreamPos(myStartAttachPoint)));
```

```
        double angle = ::acos(line().dx() / line().length());
        if (line().dy() >= 0)
               angle = (Pi * 2) - angle;

        QPointF arrowP1 = line().p1() + QPointF(sin(angle + Pi / 3) * arrowSize,
               cos(angle + Pi / 3) * arrowSize);
        QPointF arrowP2 = line().p1() + QPointF(sin(angle + Pi - Pi / 3) * arrowSize,
               cos(angle + Pi - Pi / 3) * arrowSize);

        arrowHead.clear();
        arrowHead << line().p1() << arrowP1 << arrowP2;

        painter->drawLine(line());
        painter->drawPolygon(arrowHead);
        if (isSelected())
        {
               painter->setPen(QPen(myColor, 1, Qt::DashLine));
               QLineF myLine = line();
               myLine.translate(0, 4.0);
               painter->drawLine(myLine);
               myLine.translate(0,-8.0);
               painter->drawLine(myLine);
        }
}
```

Appendix B: R code

```
#-------------------------------
#WRITE MULTI-STATE LIKELIHOOD FOR USER FROM MATRIX INPUT OF SCHEMATIC

#FUNCTIONS DEFINED BELOW; SECTION ENDS AT LINE 154; ONLY NEED TO RUN ONCE (BUT DOESN'T HURT ANYTHING TO RUN EACH TIME)
#--------------------------------
#FUNCTION REVSTRING REVERSES A STRING (USED TO SEARCH FROM THE END OF A STRING)
revstring <- function(s) paste(rev(strsplit(s,"")[[1]]),collapse="")

#DEFINE FUNCTION; CONVERTS INPUT MATRIX TO LIST OF CHARACTER STRINGS OF POSSIBLE PATHS
foo<-function(path.trace, tabl,startnode) {
  children <- colnames(tabl[which(tabl[startnode,]==1)])  #CHILDREN IS SYMBOL NAMES OF ICONS IMMED. DOWNSTREAM FROM CURRENT NODE
  node<-startnode
  while (!is.na(children[1])) { #REPEAT AS LONG AS THE CURRENT NODE HAS AT LEAST 1 ICON DOWNSTREAM
      #print(path.trace[[1]])  #DEBUGGING OUTPUT
      #print(children[1])      #DEBUGGING OUTPUT
    if (length(children)>1) { #ONLY IF CURRENT NODE IS A FORK
      if (length(path.trace)>1) { #RUN IF AT LEAST ONE BRANCH OF THE FORK HAS ALREADY BEEN PROCESSED; CREATES EMPTY PATH W/INDEX 1 AS OTHER BRANCHES GET BUMPED TO INDEX 2+
        for (k in 1:(length(children)-1)) {    #UNCOMMENT TO HANDLE BRANCHES WITH >2 CHILDREN
          for (j in length(path.trace):2) {
            path.trace[[j+1]]<-path.trace[[j]]
            #print(path.trace)
            #print("add")
          }
        }                              #UNCOMMENT TO HANDLE BRANCHES WITH >2 CHILDREN
      }
      path.trace[c(1:length(children))]<-list(path.trace[[1]])  #REPLACE (1,2) WITH (1:length(children)); COPIES CURRENT PATH AS MANY TIMES AS NUMBER OF CHILDREN IN PREP TO ADD FURTHER ROUTES
      #print(path.trace)
      #print("break")
      for (k in 1:(length(children)-1)) {      #UNCOMMENT TO HANDLE BRANCHES WITH >2 CHILDREN
        path.trace[[k+1]]<-paste(path.trace[[k+1]],children[k+1])   #CHANGE 2'S TO (k+1)'S; ADD NODE IMMED DOWNSTREAM TO PATH FOR BRANCHES INDEXED 2+
        path.trace[[k+1]]<-foo(path.trace[[k+1]], tabl, children[k+1])  #CHANGE 2'S TO (k+1)'S; RECURSIVE CALL FOR BRANCHES INDEXED 2+
      }                                #UNCOMMENT TO HANDLE BRANCHES WITH >2 CHILDREN
```

```
   #print(path.trace)
  }
  path.trace[[1]] <- paste(path.trace[[1]],children[1]) #AFTER BRANCHES INDEXED 2+ ARE
RETURNED, CONTINUE ALONG BRANCH INDEXED 1, ADDING NODE IMMED. DOWNSTREAM
  node<-which(rownames(tabl)==children[1])  #CURRENT NODE BECOMES NODE(S) IMMED
DOWNSTREAM
  children<-colnames(tabl[which(tabl[node,]==1)]) #CHILDREN BECOME CHILDREN OF CHILD
NODE(S)
  #print(node)
  #print(children)
  #print("")
 }
 return(path.trace)
}

#DEFINE FUNCTION BRANCH; CONVERTS FORK NAMES INTO GAMMA AND 1-GAMMA CHAR
STRINGS
branch<-function(branches) {  #BRANCHES IS INPUT LIST OF POSS. PATHS
 gates<-unique(unlist(regmatches(branches, (gregexpr("F\\.([^[:blank:]]+) ([^F^[:blank:]]+)",
branches)))))  #FINDS AND RETURNS LIST OF FORK ICON NAMES (START WITH "F") FOR EACH
PATH OUT OF LIST OF PATHS
 if (length(gates)==0) return(branches)  #IF NO FORKS THEN NO CHANGES TO INPUT
 gatenums<-unique(regmatches(gates, (regexpr("[^[:blank:]]+", gates))))  #RETURNS NAME OF
FIRST FORK ICON ALONG EACH POSS PATH; BLANK STRINGS FOR PATHS WITH NO FORKS
 for (i in 1:length(gatenums)) {
  #print(gates)
  #print(gatenums[i])
  thisgate<-grep(paste(gatenums[i]," ",sep=""), gates, value=T, fixed=T)  #
  #print(thisgate)
  for (j in 1:(length(thisgate)-1)) {
   sgate<-""
   tempgate<-""
   if (length(thisgate)>2) {
    tempgate<-as.character(j)
    sgate<-as.character(c(1:(length(thisgate)-1)))
   }
   extra<-substr(thisgate[j],3,regexpr("$",thisgate[j]))
   param<-paste("* gamma",tempgate,"n",extra,sep="")
   branches<-gsub(thisgate[[j]], param, branches,fixed=T)
  }
  extra<-substr(thisgate[length(thisgate)], regexpr(" ",thisgate[length(thisgate)]),
regexpr("$",thisgate[length(thisgate)]))
  param<-paste("gamma",sgate,"n",substr(thisgate[length(thisgate)],3,(regexpr("
",thisgate[length(thisgate)])[[1]]-1)),sep="",collapse="-")
```

```
    param<-paste("* (1-",param,")",extra,sep="")
    branches<-gsub(thisgate[length(thisgate)], param, branches, fixed=T)
  }
  return(branches)
}

#DEFINE FUNCTION WRITE.PATH; CONVERTS SINGLE PATH TO PROBABILITY, DOES NOT HANDLE
FORKS, ONLY WRITES 'NOT SEEN' EG (1-Px)
write.path<-function(path.in) {
  path.out<-path.in
  path.out<-gsub("C\\.([^[:blank:]]+) ", "", path.out)
  startpos<-regexpr("[^[:blank:]]",path.out)
  startchar<-substr(path.out,startpos,startpos)
  path.out[which(startchar=="S")]<-sub("S\\.([^[:blank:]]+)", "S\\1",
path.out[which(startchar=="S")])
  path.out[which(startchar=="G")]<-sub("G\\.([^[:blank:]]+)", "(1-P\\1)",
path.out[which(startchar=="G")])
  path.out[which(startchar=="E")]<-sub("E\\.([^[:blank:]]+)", "(1-lastP\\1)",
path.out[which(startchar=="E")])
  path.out<-gsub("G\\.([^[:blank:]]+)", "* (1-P\\1)", path.out)
  path.out<-gsub("R\\.([^[:blank:]]+)", "rel\\1", path.out)
  path.out<-gsub("S\\.([^[:blank:]]+)", "* S\\1", path.out)
  path.out<-gsub("E\\.([^[:blank:]]+)", "* (1-lastP\\1)", path.out)
  return(path.out)
}

#DEFINE FUNCTION WRITE.PROB; CONVERTS PATHLIST TO LIKELIHOOD (USES WRITE.PATH),
NEEDS PATHLIST, GATELIST, START POINT
write.prob<-function(pathlist,gatelist,startgate) {
  detect<-""
  gatenums<-substr(gatelist,3,100) #NOTE ASSUMES GATES ARE 2 CHARS (EG "E." OR "G,")
FOLLOWED BY GATE NAME; NAME IS MAX 98 CHARS LONG
  #print(c("Gate Numbers:",gatenums))
  startgates<-unique(unlist(regmatches(pathlist,
(gregexpr("(R\\.([^[:blank:]]+)([^[:alpha:]^(]+))|(G\\.([^[:blank:]]+)([^[:alpha:]^(]+))|(E\\.([^[:bla
nk:]]+)(([^[:alpha:]^(]+)|$))", pathlist)))))
  #print(startgates)
  start<-startgates[which(grepl(startgate,startgates))]
  #print(c("Start Gate:",start))
  starttemp<-regexpr(start,pathlist,fixed=T)
  attr(starttemp,"match.length")[which(starttemp<0)]<-0
  starttemp[which(starttemp<0)]<-nchar(pathlist[which(starttemp<0)])+1
  startpos<-attr(starttemp,"match.length")+starttemp
  #print(as.expression(gatelist))
```

```
 for (i in 1:length(gatelist)){
  #START BY SUBSETTING EACH PATH FROM REL TO GATE X
  temp<-regexpr(paste(as.expression(gatelist[i]),"(([(+*[:blank:]-])|($))",sep=""), pathlist)
  #print(temp)
  endpos<-attr(temp,"match.length")+temp-1
  trunc.paths<-substr(pathlist,startpos,endpos)
  #print(trunc.paths)
  if (grepl("E",gatelist[i]))  trunc.paths<-gsub(gatelist[i], paste("* lastP",gatenums[i], sep=""),
trunc.paths)
  if (grepl("G",gatelist[i]))  trunc.paths<-gsub(gatelist[i], paste("* P",gatenums[i], sep=""),
trunc.paths)
  trunc.paths<-write.path(trunc.paths)
  #print(trunc.paths)
  detect.i<-paste(unique(trunc.paths[which(trunc.paths!="")]), collapse=" + ")
  names(detect.i)<-paste(startgate,gatelist[i],sep=" ")
  detect<-c(detect,detect.i[which(detect.i!="")])
  #print(detect)
 }
 #print(pathlist)
 detect.final<-detect[which(detect != "")]
 return(detect.final)
}

#DEFINE FUNCTION WRITE.PROB.0;
write.prob.0<-function(pathlist,startgate) {
 startgates<-unique(unlist(regmatches(pathlist,
(gregexpr("(R\\.([^[:blank:]]+)([^[:alpha:]^(]+))|(G\\.([^[:blank:]]+)([^[:alpha:]^(]+))|(E\\.([^[:bla
nk:]]+)(([^[:alpha:]^(]+)|$))", pathlist)))))
 start<-startgates[which(grepl(startgate,startgates))]
 #print(start)
 starttemp<-regexpr(start,pathlist,fixed=T)
 attr(starttemp,"match.length")[which(starttemp<0)]<-0
 starttemp[which(starttemp<0)]<-nchar(pathlist[which(starttemp<0)])+1
 startpos<-attr(starttemp,"match.length")+starttemp  #+1 INCLUDES BLANK SPACE AFTER
START GATE
 pathlist.gate<-substr(pathlist,startpos,nchar(pathlist))
 notseen.gate<-write.path(pathlist.gate)
 #print(notseen.gate)
 ways.to.die<-""
 for (i in 1:length(notseen.gate)) {
  next.s.die<-notseen.gate[i]
  while(grepl("([[:blank:]]|^)S([^[:blank:]]+)",next.s.die)) {
   next.s.die<-revstring(sub("([^[:blank:]]+)S([[:blank:]]|$)", ")\\1S-1(\\2",
revstring(next.s.die))) #uses revstring to search for last occurrance
```

```
      temp<-regexpr("1-S([^[:blank:]]+)", next.s.die)
      endpos<-temp+attributes(temp)$match.length-1
      ways.to.die<-c(ways.to.die,substr(next.s.die,1,endpos))
    }
  }
  #print(ways.to.die)
  #print(unique(unlist(ways.to.die)))
  notseen.or.died<-c(unique(notseen.gate), unique(unlist(ways.to.die)))
  if (all(notseen.or.died == "")) {
    notseen.or.died<-c("1", notseen.or.died)
  }
  notseen.or.died<-sort(notseen.or.died[which(notseen.or.died != "")])
  detect.0<-paste(notseen.or.died,collapse=" + ")
  names(detect.0)<-paste(startgate,"0",sep=" ")
  detect.0.final<-detect.0[which(detect.0 != "")]
  return(detect.0.final)
}

library(rSymPy)
sympyStart()

#END DEFINE FUNCTIONS SECTION
#--------------------------------------
directory <- "C:/Users/user/Desktop/Grad School/Thesis/Tribpit/"  #CHANGE TO DIRECTORY
YOU WANT TO WORK IN
setwd(directory)
example <- read.table("unit test gui output matrix.txt", header=T,sep=",")  #FILE YOU WANT TO
READ IN; WHATEVER YOU SAVED OUTPUT FROM GUI UNDER

#CLEAN MATRIX: COMBINE CONSECUTIVE BRANCHES INTO SINGLE MULTI-FORKED BRANCH
for (i in 1:length(example)) {
  if (grepl("F", rownames(example[i,]))) {
    while (grepl("F", colnames(example[which(example[i,]==1)]))) {
      kids<-which(example[i,]==1)
      for (j in (intersect(grep("F", colnames(example[i,])), kids))) {
        xfer <- which(example[j,]==1)
        example[j,xfer]<-example[j,xfer]-1
        example[i,xfer]<-example[i,xfer]+1
        example[i,j]<-example[i,j]-1
        newrow<-rownames(example)
        newrow[j]<-paste("REMOVE",j,sep="")
        rownames(example)<-newrow
        newcol<-colnames(example)
        newcol[j]<-paste("REMOVE",j,sep="")
```

```
        colnames(example)<-newcol
      }
    }
  }
}
blankcol<-grep("REMOVE",colnames(example))
blankrow<-grep("REMOVE",rownames(example))
if (length(blankrow)>0) example<-example[-blankrow,]
if (length(blankcol)>0) example<-example[-blankcol]


#USE FUNCTION FOO TO TRACE POSSIBLE PATHS FROM RELEASE(S)
path<-vector("list",0)
zeronode <- rownames(example[grep("R", rownames(example)),])  #ASSUMES RELEASE ICONS
ARE LABELED WITH "R" AND NOTHING ELSE IS
for (i in which(rownames(example) %in% zeronode)) {
  path<-c(path,list(rownames(example[i,]))) #ADDS RELEASE SYMBOL TO LIST OF PATHS AS
START OF A NEW PATH
  path[[i]]<-as.list(foo(path[[which(rownames(example) %in% zeronode)[i]]],example,i))  #CALLS
FUNCTION FOO, PATH.TRACE INPUT IS RELEASE SYMBOL ADDED TO PATH IN PREVIOUS LINE
}

#GET FIRST DETECTED AT PROBABILITIES
#---------------------------------

#START BY USING FUNCTION BRANCH TO WRITE PROPER GAMMA OR 1-GAMMA FOR
BRANCHES
paths<-unlist(path) #REMOVE NESTED LIST STRUCTURE FROM PATHS, CREATE SINGLE-LEVEL
LIST OF POSS. PATHWAYS
gammas<-branch(paths)

#NOW WRITE 1ST SEEN AT GATE X PROBS
first.detect<-""
first.detect.0<-""
gates<-unique(unlist(regmatches(paths, (gregexpr("(G\\.([^[:blank:]]+))|(E\\.([^[:blank:]]+))",
paths)))))

for (i in 1:length(zeronode)) {
  first.detect<-c(first.detect,write.prob(gammas,gates,zeronode[i]))
  first.detect.0<-c(first.detect.0,write.prob.0(gammas,zeronode[i]))
}

#PUT IT TOGETHER
first.detect<-c(first.detect[which(first.detect != "")],first.detect.0[which(first.detect.0 != "")])
```

```
#END FIRST DETECTED AT SECTION
#---------------------------------

#CONDITIONAL LIKELIHOOD FOR EACH GATE; PUT TOGETHER WITH 1ST SEEN TO FORM TOTAL
#---------------------------------
#SET UP LIST STRUCTURE; 1ST LIST ELEMENT IS '1ST SEEN' LKLHD, REST ARE SUB-LKLHDS FOR
EACH GATE
liklhd.names<-c("1st Seen at:", paste("Given detection at ",gates,":",sep=""))
cond.lklhd<-as.list(rep("",length(gates)+1)); names(cond.lklhd)<-liklhd.names
cond.lklhd[[1]]<-first.detect
for (i in 1:length(gates)) {
  cond.lklhd[[i+1]]<-c(write.prob(gammas,gates,gates[i]),write.prob.0(gammas,gates[i]))
}

#VERIFY THAT ALL PROBS SUM TO 1 USING SYMPY
#rm(.Rsympy)   #THIS LINE MAY HELP IF SYMPY DOESN'T WORK
vars<-unique(unlist(regmatches(first.detect,gregexpr("[[:alpha:]]+[[:alnum:]]+",first.detect))))
var<-sympy("var('",paste(vars,collapse=","),"')")
#releases<-paste(grep("rel",vars,value=T),sep=","); class(releases)<-"Sym"  #UNCOMMENT FOR
MULTIPLE RELEASES

#first.det.sum<-paste(first.detect,collapse=" + ")
#for (i in 1:length(releases)) {                                   #UNCOMMENT FOR MULTIPLE RELEASES
#  first.det.sum<-Sym("(",first.det.sum,").subs(",releases[[i]],", 1)")     #UNCOMMENT FOR
MULTIPLE RELEASES
#}                                                        #UNCOMMENT FOR MULTIPLE RELEASES

cond.sum<-as.list(rep("",length(cond.lklhd))); names(cond.sum)<-liklhd.names
expand.cond.sum<-as.list(rep("",length(cond.lklhd))); names(expand.cond.sum)<-liklhd.names
for (i in which(nchar(cond.lklhd)<=10000)) {
  cond.sum[[i]]<-paste(cond.lklhd[[i]],collapse=" + ")
  if (cond.sum[[i]] != "1") {
    expand.cond.sum[[i]]<-Sym("(", cond.sum[[i]], ").expand()")
    tf<-Sym("(",expand.cond.sum[[i]]," == 1)")
    print(paste("T/F:  ", liklhd.names[i]," Likelihood sums to 1 --", sep=""))
    print(tf)
  }  else print(paste("T/F:  ", liklhd.names[i]," Likelihood sums to 1 -- ", (cond.sum[[i]] == "1"),
sep=""))
}

#END COND. LKLHD
#---------------------------------
```

```
#WHAT IS ESTIMABLE (REPLACE WITH LAMBDAS, PHIS)
#---------------------------------
maybe.lambda<-(unlist(cond.lklhd)[grep("last",unlist(cond.lklhd))])
maybe.lambda<-unlist(strsplit(maybe.lambda,"\\+"))
lambdas<-unique(maybe.lambda[grep("([^t])P",maybe.lambda,ignore.case=F,invert=T)])
lambdas<-lambdas[grep("\\(1-S([^[:blank:]]+)",lambdas,invert=T)]
lambdas<-lambdas[grep("1-last",lambdas,fixed=T,invert=T)]
names(lambdas)<-paste("lambda",1:length(lambdas),sep="")
cond.lklhd.lambda<-cond.lklhd
for (i in 1:length(cond.lklhd)) {
  if (all(cond.lklhd.lambda[[i]] != "1")) {
    cond.lklhd.lambda[[i]][grepl(" 0",names(cond.lklhd.lambda[[i]]),fixed=T)]<-paste("1 - (",
paste(cond.lklhd.lambda[[i]][!grepl(" 0",names(cond.lklhd.lambda[[i]]),fixed=T)], collapse=" + "),
")", sep="")
  }
  for (j in 1:length(lambdas)) {
    cond.lklhd.lambda[[i]]<-gsub(lambdas[j],names(lambdas[j]),cond.lklhd.lambda[[i]], fixed=T)
  }
}
none<-unlist(strsplit(unlist(cond.lklhd.lambda), split="\\+"))
split<-c(rep("",5))
split[3]<-"(( \\* )?rel([^[:blank:]]+)( \\* ))"
split[4]<-"(( \\* )?P([^[:blank:]]+)( \\* )?)"
split[5]<-"(( \\* )?lambda([^[:blank:]]+)( \\* )?)"
split[1]<-"(( \\* )?\\(1-P([^[:blank:]]+)\\)( \\* )?)"
split[2]<-"(( \\* )?\\(1-lambda([^[:blank:]]+)\\)( \\* )?)"
spl<-paste(split,collapse="|")
phis<-unique(unlist(strsplit(none,split=spl)))
phis<-phis[which(!grepl("(1[[:blank:]]?-[[:blank:]]?\\(([^[:blank:]]+)?)|^$",phis) & phis != "1")]
phis<-sub("[[:blank:]]+$", "", phis)
phis<-unique(sub("^[[:blank:]]+", "", phis))
phis<-phis[grep("[[:alnum:]]",phis)]
names(phis)<-paste("phi",1:length(phis), sep="")
cond.lklhd.est<-cond.lklhd.lambda

for(j in 1:length(cond.lklhd.est)) {
  for (i in 1:length(phis)) {
    cond.lklhd.est[[j]]<-gsub(phis[i],names(phis[i]),cond.lklhd.est[[j]], fixed=T)
  }
}
param<-
unique(unlist(regmatches(unlist(cond.lklhd.est),gregexpr("[[:alpha:]]+[[:alnum:]]+",unlist(cond.l
klhd.est)))))
```

```
#END WHAT IS ESTIMABLE
#----------------------------------

#CHANGE GATE NAMES TO MATCH SUCKER DATA (COMMENT THESE LINES OUT FOR MORE
GENERAL)
#---------------------------------------
oldgate<-gates
newgate<-c("Gate1","Gate2_a","Gate3_a","Gate4_a","Gate4_b","Gate2_b","Gate3_b")
#c("L_223","L_41","L_524","L_825", "L_2012")
#c("A0","A2","A3","A4","A5","A6","A7","A8","D1","D2","G2","E1","E2","B1")
#c("L_223","L_41","L_524","L_825", "L_2012", "R_2012", "R_825","R_524","R_41","R_223")
oldprobs<-c("P3","P6","P10","lastP15","lastP19","P8","P12")  #c("P3", "P5", "P7", "P9",
"lastP11")
#c("lastP4","P6","P8","P13","P15","P19","P21","P23","P33","P35","lastP55","P40","P42","P27")
#c("P5","P13","P21", "P31", "lastP29","lastP30", "P32", "P22","P14","P6")
newprobs<-c("PGate1","PGate2a","PGate3a","PGate4a","PGate4b","PGate2b","PGate3b")
#c("P223", "P41", "P524", "P825", "lastP2012") #c("PA0","PA2","PA3","PA4", "PA5", "PA6",
"PA7","PA8","PD1","PD2","PG2","PE1","PE2","PB1")
for (i in 1:length(oldgate)) {
  for (j in 1:length(cond.lklhd.est)) {
    cond.lklhd.est[[j]]<-gsub(oldgate[i],newgate[i],cond.lklhd.est[[j]],fixed=T)
    cond.lklhd.est[[j]]<-gsub(oldprobs[i],newprobs[i],cond.lklhd.est[[j]],fixed=T)
    names(cond.lklhd.est[[j]])<-gsub(oldgate[i],newgate[i],names(cond.lklhd.est[[j]]))
  }
  names(cond.lklhd.est)<-gsub(oldgate[i],newgate[i],names(cond.lklhd.est))
  param<-gsub(oldprobs[i],newprobs[i],param)
  lambdas<-gsub(oldprobs[i],newprobs[i],lambdas)
}

# gates<-newgate

#WRITE TO TEXT FILE FOR PROGRAM USER
#----------------------------------
options(useFancyQuotes = FALSE) #IMPORTANT! IF DON'T SET THIS OPTION THEN DOUBLE
QUOTES WILL NOT BE READABLE BY USER
filename<-paste(directory, "unit test negative lklhd.txt", sep="") #OUTPUT FILE NAME
write("parameters", file=filename)
write(paste(paste(param, collapse=", "), ";", sep=""), file=filename, append=T)
write("variables", file=filename, append=T)
write(paste("#", names(phis), " = ", phis,sep=""), file=filename, append=T)
write(paste("#", names(lambdas), " = ", lambdas,sep=""), file=filename, append=T)

for (i in 1:length(cond.lklhd.est)) {
  write("likelihood", file=filename, append=T)
```

```
  write(dQuote(names(cond.lklhd.est)[[i]]), file=filename, append=T)
  cat(paste(paste(names(cond.lklhd.est[[i]]), cond.lklhd.est[[i]], sep=", "), collapse=", \n"),
file=filename, append=T)
  cat("; \n", file=filename, append=T)
}

#write.csv(data.output, paste(directory, "example CJS data.csv", sep="")) #
"C:/Users/user/Desktop/Grad School/Thesis/Practice/example data.csv")
#data.raw<-data.raw[which(data.raw$gate != "0"),]
#-----------------------------
#CONSTRUCT CONDITIONAL DETECTION HISTORY FOR EACH TAG; CALCULATE SUFF. STATS FOR
EACH SITE
#START WITH DATA IN 4 COLUMN FORM: RELEASE GROUP, TAG ID, DATETIME OF DETECTION,
LOCATION OF DETECTION.
#EITHER 1 LINE PER DETECTION OR FIRST AND LAST DETECTIONS AT EACH SITE
#-----------------------------
#data.raw<-
data.frame(data.clean$Rel.Grp,data.clean$TAG,data.clean$Det.Time,data.clean$site.code,data.
clean$array.code)
#names(data.raw)<-c("release","tag","datetime","gate","array")
data.in<-read.table(paste(directory, "chiwawa_data_clean.csv", sep=""),header=T, sep=",",
colClasses="character") #"C:/Users/user/Desktop/Grad School/Thesis/Practice/example
data.csv",header=T, sep=",", colClasses="character")
data.raw<-
data.frame(data.in$release,data.in$tag,data.in$datetime,data.in$gate,stringsAsFactors=F)
names(data.raw)<-c("release","tag","datetime","gate")
#COLUMN NAMES SHOULD BE: release FOR RELEASE GROUP, datetime FOR DATE/TIME OF
DET., tag FOR TAG ID, gate FOR DET. LOCATION
Tags <- unique(data.raw$tag)
det.list<-as.list(rep("",length(Tags))); names(det.list)<-Tags
cond.hist<-as.list(rep("",length(Tags))); names(cond.hist)<-Tags
# temp<-0
# data.final <- data.raw[0,]
for (i in 1:length(Tags)) {
  data.tagi<-data.raw[which(data.raw$tag == Tags[i]),]
#   dual.arrays<-c("A0", "A2", "D2", "G2")   #LIST OF GATES THAT ARE DUAL ARRAYS IN THE
MODEL
#   ab.gates<-as.character(data.tagi$array[which(duplicated(data.tagi$array))])
#   temp<-temp+length(ab.gates)
#   if (length(ab.gates)>0) {
#     for (j in 1:length(ab.gates)) {
#       data.tagi$gate[which(data.tagi$array %in% ab.gates[j])]<-paste(ab.gates[j],"ab",sep="")
#     }
#     data.tagi<-data.tagi[-(which(duplicated(data.tagi$array))),]
```

```
#   }
#   data.tagi$gate[which(!data.tagi$array %in% dual.arrays)]<-
data.tagi$array[which(!data.tagi$array %in% dual.arrays)]
  det.list[[i]]<-c(data.tagi$release[1],data.tagi$gate,"0")
#
#   data.final <- rbind(data.final,data.tagi)

  for (j in 1:(length(det.list[[i]])-1)) {
      cond.hist[[i]][j]<-paste(det.list[[i]][j],det.list[[i]][j+1],collapse=" ")
  }

}
write.csv(data.final,file=paste(directory,"clean data for trish.csv",sep=""))
cond.count<-tapply(unlist(cond.hist),unlist(cond.hist),length)
names(cond.count)<-gsub("rel","R.1",names(cond.count),fixed=T)
sum(cond.count)+temp
write(paste(names(cond.count),cond.count, sep=" : "), file=paste(directory, "condlklhd unit test
1 counts.txt", sep=""))    #CHANGE BACK


#----------------------------
#TRY TO GET b,g SUFF STATS BROKEN DOWN BY GATE (e.g. b(ij))

data.gates<-unique(unlist(det.list))
data.gates<-data.gates[-(which(data.gates == "0"))]
a<-rep(0,length(data.gates)); names(a)<-data.gates
b<-rep(0,length(data.gates)); names(b)<-data.gates
b.ij<-matrix(0,length(data.gates), length(data.gates), dimnames = list(data.gates,data.gates))
d<-rep(0,length(data.gates)); names(d)<-data.gates
g<-rep(0,length(data.gates)); names(g)<-data.gates
g.ij<-matrix(0,length(data.gates), length(data.gates), dimnames = list(data.gates,data.gates))
out.paths<-vector("list",length(data.gates)); names(out.paths)<-data.gates
out.trans<-vector("list",length(data.gates)); names(out.trans)<-data.gates
for (i in 1:length(data.gates)) {
  oldgatei<-oldgate[which(newgate == data.gates[i])]
  if (data.gates[i] == "rel_2010") oldgatei<-"R.1"
  if (data.gates[i] == "rel_2011") oldgatei<-"R.2"
  paths.seeni<-paths[grep(oldgatei,paths)]
  g.paths<-character()
  out.paths[[i]]<-character()
  out.trans[[i]]<-character()
  for (j in 1:length(paths.seeni)) {
    g.paths<-c(g.paths,unlist(strsplit(paths.seeni[j],oldgatei))[2])
    out.paths[[i]]<-c(out.paths[[i]],unlist(strsplit(g.paths[j], "[GE]"))[1])
```

```
 }
 out.paths[[i]]<-unique(write.path(branch(out.paths[[i]])))
 out.trans[[i]]<-out.paths[[i]]
 for (j in 1:length(phis)) out.trans[[i]]<-gsub(phis[j],names(phis)[j],unlist(out.trans[[i]]),fixed=T)
 for (j in 1:length(lambdas)) {
  lam<-unlist(strsplit(lambdas[[j]],"\\* l"))[1]
  out.trans[[i]]<-gsub(lam,names(lambdas[j]),out.trans[[i]], fixed=T)
 }
 for (j in 1:length(det.list)) {
  oldgates.seen<-c("R.1", oldgate[which(newgate %in% det.list[[j]])])
  if (any(oldgates.seen %in% unique(unlist(strsplit(g.paths," "))))) g[i]<-g[i]+1
  for (k in 1:length(oldgates.seen)) {
   if (oldgates.seen[k] == "R.1") newgate.seen<-"rel" else newgate.seen<-
newgate[which(oldgate == oldgates.seen[k])]
   if (oldgates.seen[k] %in% unique(unlist(strsplit(g.paths," ")))) g.ij[i,which(data.gates ==
newgate.seen)]<-g.ij[i,which(data.gates == newgate.seen)] + 1
  }
  if (data.gates[i] %in% det.list[[j]]) {
   a[i]<-a[i]+1
   temp<-det.list[[j]][(grep(data.gates[i],det.list[[j]])+1):length(det.list[[j]])]
   if ((any(temp %in% data.gates)) & (length(temp)>0)) {
    b.ij[i,which(data.gates %in% temp)]<-b.ij[i,which(data.gates %in% temp)]+1
    b[i]<-b[i]+1
   }
   #if (any(det.list[[j]][(grep(data.gates[i],det.list[[j]])+1):length(det.list[[j]])] == "c")) d[i]<-d[i]+1
  }
 }
}
data.gates[which(a == b)]
out.trans[data.gates[which(a == b)]]
```