HATCHPBT

A statistical tool to conduct an *a priori* analysis of the precision and accuracy of the maximum likelihood estimator of the proportion of hatchery-origin spawners using parentage-based tagging

Richard A. Hinrichsen

November 13, 2012

Preliminary DRAFT

# Introduction

Assessments of the status of endangered Columbia River salmon populations require reliable estimates of the proportion of hatchery-origin spawners on the spawning grounds. Without such an estimate, it would be impossible to estimate a trend in the wild-origin population abundance or estimate population extinction risks (Hinrichsen 2003; McClure et al. 2003). Furthermore, quantifying the potential for interbreeding between hatchery-origin spawners and wild-origin spawners in the wild, which may reduce the genetic fitness of subsequent generations of wild-origin fish, also depends on this estimate (Waples 1991). The potential genetic risks of all 178 hatchery programs in the Columbia River basin were assessed using the proportion of hatchery-origin spawners (Mobrand et al. 2005; HSRG 2009). To allow distinction between natural-origin and hatchery-origin salmon in the Columbia Basin, the U.S. Congress presently requires the US Fish and Wildlife Service to visibly mark all hatchery production intended for harvest.[[1]](#footnote-1) Visible marking of hatchery releases is a widespread practice among hatchery operators in the Columbia River basin, though non-visible marking procedures are sometimes substituted for or added to visible marks.

Despite the importance of estimates of proportion of hatchery-origin fish (*p*) on the spawning grounds, until recently, reliable estimation techniques have been lacking. The statistical difficulty of estimating the proportion of hatchery-origin escapement when some hatchery-fish are not visibly marked has been recognized for over thirty years (Hankin 1982). The difficulty is especially pronounced when different source hatcheries do not use the same marking fraction. Hinrichsen et al. (2012) addressed the problem of estimating *p* in a hatchery program that used visible marks and coded-wire tag recoveries. The authors developed a generalized least squares estimator of the proportion of hatchery origin spawners and compared it to a simplified method of moments estimator.

An alternative to this CWT approach to estimating the proportion of hatchery-origin spawners is to use genetic tagging called parentage-based tagging (PBT) of hatchery releases, which can be used to mark a high percentage of juveniles released. In this alternative approach, some juveniles are visibly marked (VM), parentage-based tagged (PBT), or both. Many (but not all) hatchery juveniles in the Columbia River basin are released with a VM with an adipose fin clip, a ventral fin clip, or a visible implant elastomer tag. PBT involves genotyping hatchery broodstock (parents) and adding these genotypes to a database (Steele et al. 2011; Anderson and Garza 2005; Anderson and Garza 2006). Genotyped progeny of these parents collected as juveniles or adults can be assigned back to their parents, thus creating a tag identifying the hatchery of origin. Thus offspring of the genotyped brood stock are genetically tagged. Software used to assign genotyped progeny to their parents, SNPPIT 1.0 developed by Anderson (2010), is available online at http://www.mybiosoftware.com/population-genetics/6013.

In a carcass survey, VM fish are recovered as adults at a spawning area together with fish that are the progeny of salmon spawning in the wild. A subsample of the sampled carcasses is then drawn and each fish in the subsample is genotyped to determine if it is PBT. The genotypes of subsampled carcasses are compared to the genotypes of parents in a database. If there is a match, the sampled carcass represents a PBT spawner, and the hatchery of origin is determined. Note that the subsample potentially consists of spawners in three different categories: VM spawners, not VM spawners, and wild-origin spawners. Therefore, it is not guaranteed that each genotyped spawner will carry a PBT.

The goal of this documentation is to present a method for evaluating different study designs aimed at estimating the proportion of hatchery-origin spawners in a program that uses VM and PBT to identify hatchery-origin spawners. To do this, I develop a maximum likelihood estimator of the proportion of hatchery-origin spawners that allows for the possibility of two or more source hatcheries. In general, these source hatcheries may use different VM fractions and (possibly) different PBT fractions. The precision of the MLE developed depends on several quantities, some of which may be selected by a researcher or manager: the number of carcasses sampled, the VM fractions, the PBT fractions, and the actual proportions of hatchery fish from each source hatchery. These represent management controls that may be manipulated to achieve a target level of precision in the proportion of hatchery-origin spawners.

For convenience, variable names and their definitions are given in Appendix A. Statistical code for the analysis, written in the R programming language, may be found in Appendix B.

# Methods

In order to estimate the proportion of hatchery-origin spawners, I specify a probability distribution for observed spawners using a two-stage sampling protocol (Figure 1). The protocol uses an initial sample of size *N* to determine how many fish are VM and not VM (denoted ~VM). Then a random (sub)sample of the VM group is taken and genotyped. And a random (sub) sample of the spawners ~VM is also taken and genotyped. One possibility, if the initial sample is not prohibitively large, is to genotype all of the spawners from the initial sample. At another extreme, the random subsamples of the VM and ~VM groups could represent a small fraction of the initial sample. One of the main goals of this work is to determine how large of a subsample is sufficient to obtain a precise estimate of the proportion of hatchery-origin spawners. The steps for collecting spawner data necessary to estimatethe proportion of hatchery-origin spawners are the following:

1. Take a random sample of size *N* of the spawners in the wild. Note that this sample will consist of two groups: fish that are VM and fish without a VM (i.e., ~VM)
2. Take a random (sub)sample of size from the VM group and a random (sub)sample of size from the ~VM group. Note that the total subsample size is .
3. Send these subsamples to the genetics lab to determine whether they are PBT and if so, their hatchery of origin.

The method I develop in this documentation is a maximum likelihood technique (Mood et al. 1974). Other estimation techniques are also possible, for example, a generalized least squares method based on the method of moments (Hinrichsen et al. 2012; Kariya and Kurata 2004). I use maximum likelihood because it has a well-developed theory. Maximum likelihood estimators have desirable statistical properties such as asymptotic normality, functional invariance, and consistency (Mood et al. 1974). Furthermore, the variance of the MLE may be derived as the inverse of the Fisher Information Matrix, which is the negative of the expected value of the Hessian of the likelihood function (Mood et al. 1974).

|  |
| --- |
| VM  PBT  Sample (*N*)  VM subsample (*n*1)  ~VM subsample (*n*2)  Universe of spawners |
| Figure 1.—Carcass sampling represented by a Venn diagram. The universe is represented by all returning spawners to a particular spawning ground in the wild. The sample of size *N* is a random sample of the universe of spawners. The subsamples of sizes *n*1 and *n*2 are random samples of the VM spawners and not VM (~VM) spawners, respectively. These subsamples, totaling , are sent to the lab and checked for a PBT, while the remaining carcasses, totaling , are not. |

## Experiment and Probability Model

This section is devoted to developing the maximum likelihood estimator (MLE) of the proportion of hatchery-origin spawners and its variance. I begin by defining the assumptions (Table 1) and the variables used in the study, which are used to develop the probability model. Let represent the fraction of spawners on the spawning grounds that originated at hatchery *i*. Let  represent the VM fraction that is applied to hatchery fish releases from hatchery *i*, and let  represent the PBT fraction that is applied to hatchery fish releases from hatchery *i*. Further assume that the total number of spawners sampled is *N* (fixed) and that  represents the number of sampled fish that are VM and  represents the number of sampled fish that are not VM. Here,  is a binomial deviate with the number of ‘trials’ equal to *N* and the probability of ‘success’ equal to the probability that a spawner is VM, which is (see Table A.1 for variable definitions).

|  |
| --- |
| Table 1.—Assumptions1 |
| (A1) Hatchery-specific VM fractions and number of spawners sampled are known. |
| (A2) Hatchery-specific PBT fractions are known. |
| (A3) Every individual spawner has the same probability of being sampled. |
| (A4) Every individual hatchery-origin spawner from the same hatchery has the same probability of having a VM. |
| (A5) Every individual hatchery-origin spawner from the same hatchery has the same probability of having a PBT. |
| (A6) Whether an individual is sampled has no effect on the probability that another individual is sampled. |
| (A7) Whether an individual hatchery-origin spawner is VM has no effect on the probability that another individual will be VM. |
| (A8) Whether an individual hatchery-origin spawner is PBT has no effect on the probability that another individual will be PBT. |
| 1 For convenience, I derived the estimators for releases grouped at the hatchery level. To split the data by release group instead, simply replace “hatchery” by “release” in the estimation method and interpret VM fractions and PBT fractions as release-specific. |

Further assume that a subsample of sample of *N* spawners is sent to the lab for genetic testing that allows investigators to assign spawners to their parents and therefore their hatchery of origin. A total of  VM spawners are genotyped and  unmarked spawners are genotyped to determine whether they are PBT. The total subsample size is then . To make the subsample sizes meaningful, I used the following restrictions:  , , and . I use these restrictions because  must be less than or equal to the number of VM fish in the sample () and  must be less than or equal to the number of unmarked fish in the total sample (), and the subsample size must be less than the total sample size. The lower bounds follow from applying the identities and  to these upper bound inequalities. The number of genotyped marked fish originating at hatchery i and determined to have a PBT is denoted by the random variable  and the number of genotyped unmarked fish originating at hatchery i and determined to have a PBT is. The valuesare considered cell counts from a multinomial distribution with  trials with cell probabilities. The valuesare cell counts from a multinomial distribution with trials and cell probabilities.

In developing the marking and PBT program, it is critical that all hatchery fish are released randomly, and each hatchery fish is equally likely to have a VM or PBT, and that the probability a hatchery fish is VM does not influence the probability that it is PBT and visa versa. (i.e., the event that a hatchery fish is VM is independent of the event that it is PBT). Note, hatchery managers need to be aware that these hatchery fish are all treated equally, as if any group is treated in a different manner, the Table 1 assumptions will be violated. Likewise, in the second event sampling (i.e., drawing of the subsamples), the subsamples taken should be truly random and representative of the populations (i.e. if complete mixing is not occurring/homogeneity of the sample is violated, this will not work).

The assumptions in Table 1 allow one to express the joint distribution of spawner counts as a product of multinomial distributions:

|  |  |
| --- | --- |
| . | (1) |

Given the joint distribution of the observations in equation (1), it is now possible to form the log-likelihood function of the unknown parameters  by taking the natural log of the joint distribution and treating the result as a function of the parameters:

|  |  |
| --- | --- |
| . | (2) |

## Score Function

To determine the MLEs of, I set the partial derivatives of the log likelihood function equal to zero and solve for the unknown values of. The partial derivatives of the log likelihood function are

|  |  |
| --- | --- |
| . | (3) |

The MLEs of (which are the zeroes of the score function) are found numerically using Fisher’s Scoring Method, which will be described in detail later in this documentation.

## Fisher Information Matrix

The next step in deriving the theoretical formulas for precision of the MLEs is to derive the Fisher Information Matrix. The inverse of the Fisher Information Matrix will supply the variances and covariances of the MLEs of . The Fisher Information Matrix is the negative of the expected value of the Hessian of the likelihood function. The Hessian of likelihood function (**H**) is found by further differentiating the log likelihood. Its diagonal elements are given by

|  |  |
| --- | --- |
| , | (4) |

and off-diagonal elements

|  |  |
| --- | --- |
| , | (5) |

where ; ; and . I assume that not all of the are zero, not all of the VM fractions are zero, and not all of the PBT fractions are equal to one; otherwise equations (4) and (5) will be undefined. The special case where all hatchery-specific PBTs equal one will be considered separately. I will also consider the special case where all VM fractions are zero.

To determine the Fisher Information Matrix, the expected values of , and are needed. These are given by

|  |  |
| --- | --- |
| , | (6) |

|  |  |
| --- | --- |
| , | (7) |

|  |  |
| --- | --- |
| , | (8) |

and

|  |  |
| --- | --- |
| . | (9) |

Taking the expected value of the Hessian by using the expected values of the observations of , and yields the following Fisher Information Matrix () with diagonal entries:

|  |  |
| --- | --- |
| , | (10) |

and off-diagonal entries

|  |  |
| --- | --- |
| , | (11) |

where  and . I assume that not all of the are zero, not all of the VM fractions are zero, and not all of the PBT fractions are equal to one; otherwise equations (9) and (10) would be undefined.

*Special case (all releases PBT)*.—In the special case where all the PBT fractions are 1, the information matrix is given by diagonal entries

|  |  |
| --- | --- |
|  | (12) |

and off-diagonal entries

|  |  |
| --- | --- |
| . | (13) |

*Special case (no releases VM)*.—In the special case where no releases are VM, the information matrix is given by diagonal entries

|  |  |
| --- | --- |
|  | (14) |

and off-diagonal entries

|  |  |
| --- | --- |
|  | (15) |

*Special case (n1 = 0 and two or more hatcheries VM all releases)*.—In the special case where there are no marked spawners sent to the lab to detect PBT and two or more hatcheries VM at 100%, none of the spawners from the hatcheries that VM at 100% will be sent to the lab. This makes the Fisher Information Matrix singular. One way to resolve this difficulty is by combining the hatcheries that VM at 100% into a single hatchery for the purposes of estimation. This makes the Fisher Information Matrix invertible by decreasing its dimension. If there are hatcheries that VM at 100%, then the revised “number of hatcheries” used for the purposes of analysis is . Assume that the hatchery indices are arranged so that the firsthatcheries are the ones that VM at 100%. Then the VM fractions for the purpose of analysis are given by . The PBT fractions for the purpose of analysis are given by , where I chose zero for the PBT fraction of the hatchery that uses 100% VM because with (), the PBT fractions for the 100% VM hatcheries do not enter into the calculation of the Fisher Information Matrix.

*Cases making Fisher Information Matrix singular.*—There are cases that make the Fisher Information Matrix singular and make it impossible to estimate the fraction of hatchery-origin spawners. One case is where a common VM fraction is not used at all of the source hatcheries and none of the sampled spawners are sent to the lab for PBT identification (i.e., ). In this case, there is not enough information to estimate the fraction of the VM carcass sample from each source hatchery. That means it is impossible to “expand” the numbers of VM fish to get an estimate of the fraction of total spawners from each hatchery. Therefore, to estimate *p* in the case where VM fractions differ between source hatcheries, some of the fish must be sent to the lab for PBT identification. The way to remedy this design flaw is to either use a common VM fraction for all hatchery releases or to send some of sampled spawners to the lab for PBT testing.

Another singular case occurs when one of the VM fractions is zero and only fish that were VM are sent to the lab for testing (i.e., ). In this case, it is impossible to get an estimate of the number of fish in the sample that originated from the hatchery that visibly mark zero fish. This is an unknown that prevents estimation of the proportion of hatchery-origin spawners. The way to remedy this difficulty is to send some of the sampled fish that are not VM to the lab for PBT testing.

Note that both of these singular cases can be avoided in practice by drawing nonzero subsample sizes of  and , and/or using common (nonzero) VM fractions at all of the hatcheries.

## Precision

Given that the Fisher Information Matrix is calculated and it is invertible, it is possible to derive the variance of the MLE of the proportion of hatchery-origin spawners. From the variance, the standard error and coefficient of variation, both useful measures of precision may be calculated. The vector of hatchery-specific maximum likelihood estimators

|  |  |
| --- | --- |
|  | (16) |

is given by

|  |  |
| --- | --- |
| , | (17) |

which is the inverse of the Fisher Information Matrix. This formula holds provided that **I** is invertible, which is not guaranteed. To determine the variance of the estimator of the overall proportion of hatchery-origin spawners, , I use the formula

|  |  |
| --- | --- |
| , | (18) |

where is an *m*-vector of 1s. The standard error of is then

|  |  |
| --- | --- |
| , | (19) |

and the coefficient of variation of  is

|  |  |
| --- | --- |
| . | (20) |

The minimum SE is obtained when both  and are as large as possible. This occurs when, , and . To find the minimal standard error,, substitute these values in the Fisher Information matrix equations, then use equations (17)-(19). The  may be found by employing equations (17)-(20). This minimum SE is useful when evaluating the precision of alternative choices for subsample sizes *n1* and *n2*.

## Fisher’s Scoring Method

The MLEs of the hatchery-specific proportions of hatchery-origin spawners are calculated using Fisher’s Scoring Method, which is similar to Newton’s method for solving nonlinear equations (Press et al. 1992), but modified to use the expected Hessian (i.e., -**I**) in place of the usual Hessian (Jennrich and Sampson 1976). In this case, Fisher’s Scoring Method becomes

|  |  |
| --- | --- |
| , | (22) |

where is the maximum likelihood estimator of  on iteration *l*+1, is the maximum likelihood estimator of  on iteration *l*, is the inverse of the Fisher Information Matrix evaluated at , and  is the gradient or score function, which represents the vector of first partial derivatives of the likelihood function evaluated at . The algorithm proceeds by making a first initial guess at the MLE, then iterating equation (22) until the estimate changes by no more than a given error tolerance (e.g., 10-5).

## Constant Marking Fractions

One more special case should be considered. That is the case where the same marking fraction is used at each source hatchery and no spawners are sent to the lab for PBT testing (i.e., ). In this case, it is unknown what fraction of the fish originated at each source hatchery. However, in the special case of constant marking fractions, this information is not needed to estimate the total fraction of hatchery-origin spawners. This occurs because the “expansion” factor for a VM spawner is the same regardless of its hatchery of origin. In the case of constant marking fractions, the MLE of the fraction of hatchery origin spawners is

|  |  |
| --- | --- |
| , | (23) |

and its variance is given by

|  |  |
| --- | --- |
| . | (24) |

Note that in this case, a numerical routine such as Fisher’s Scoring Method is not needed to estimate the MLE, because the MLE is given by equation (23). It turns out that in the case of constant marking fractions, precision of will not be improved by sending VM fish to the lab for PBT testing. That is, the variance of will not depend on. However, precision can be increased by sending fish that are not VM to the lab for testing. The results of these tests can uncover more information about how many fish are hatchery-origin instead of wild-origin. This will be demonstrated in the application below.

## Monte Carlo Estimates of Precision and Bias

As an alternative approach to estimating the precision of, Monte Carlo simulation is used. Additionally, this approach yields an estimate of accuracy (bias). Monte Carlo estimates of precision and accuracy do not rely on asymptotic theory as in the previous sections. That is, Monte Carlo estimates of precision and bias will work with small sample sizes (*N*) and small subsample sizes,  and . The Monte Carlo method proceeds by drawing NSIM random samples from the joint distribution of  given by equation (1), then calculating the MLE of the proportion of hatchery-origin spawners for each Monte Carlo sample. This yields NSIM replications of the MLE, denoted by . The Monte Carlo estimate of the standard error of the MLE of the proportion of hatchery-origin spawners is then equal to the square root of the sample variance of the Monte Carlo replications:

|  |  |
| --- | --- |
| , | (25) |

where is the sample mean of the Monte Carlo replications. The coefficient of variation of  is

|  |  |
| --- | --- |
| . | (26) |

The relative bias is calculated as

|  |  |
| --- | --- |
| . | (27) |

## Developing an Experimental Design

The equations for precision and bias may be used to formulate an experimental design that has a high chance of delivering the best accuracy and precision given some design constraints. The precision, as measured by SE of, is a function of VM fractions, PBT fractions, sample size (N), and subsample sizes  and , which, to some extent, may be manipulated to give lowest possible SE. For example, as sample size and subsample sizes and PBT fractions are increased, precision will always increase. One possible constraint on the experimental design may be the number of carcasses sent to the lab for PBT testing. Suppose this quantity is fixed at, say, . The level of precision one can expect from such a study will vary according to the fraction of these 20 fish drawn from the VM group. By varying  and  such that they sum to 20, and calculating the SE for each possible combination, it will be possible to determine the optimal subsample sizes, which will yield the greatest precision (the combination giving smallest SE). This will yield valuable insight into a design: one can calculate the very best precision possible from a given design. This can in turn be used to select a design. The following application demonstrates such a use of the program HATCHPBT.

Note that alternatively, there may be no restrictions on the number of fish sent to the lab for PBT testing. In that case, to maximize precision, all of the N spawners sampled should be sent to the lab for PBT testing. This simple cases is handled by the above equations by setting  and . This case represents that maximum amount of precision that may be attained when using a sample of size *N*.

# ApplicationS

*Design problem #1.*—Assume that a carcass sample of size N=100 is drawn from the spawning grounds and 10% of the spawners are of hatchery origin. Further assume that there are two source hatcheries that contribute spawners to the spawning grounds, hatcheries A and B, where each contributes 5% of the total spawners. Also assume that the PBT fraction at both hatcheries is 0.95. There are two different scenarios we wish to test: (a) equal VM fractions of 0.5 (); (b) and unequal VM fractions of  and , at hatcheries A and B, respectively. In each of these cases, we assume that a subsample of is drawn from the sample of 100. The goal is to determine how many of this subsample of 50 should be drawn from the group of VM spawners and how many should be drawn from the group of ~VM spawners in order to maximize precision; in other words, what are the optimal values of  and ? Which of these two scenarios (a or b) gives the smallest CV() ?

To solve this design problem, the R-code for the theoretical formulas was used. The CV() associated with each feasible combination of and was plotted, and the combination of giving lowest CV() represented the optimal design. The CV() of the optimal design for scenario (a) was then compared to that of scenario (b). In solving this problem, the subsamples were constrained so that  did not exceed and that  did not exceed .

The optimal designs for scenarios a and b are evident in Figure 2. Notice that when VM fractions are equal (), then the optimal design uses and . When common VM fractions are used, precision is always maximized by making  as large as possible. In the scenario where  and , the optimal design uses and . Scenario b gives a smaller CV() than scenario a. Scenario (b) gives a minimum CV() of 0.3338 while scenario (a) gives a minimum CV() of 0.3535.

|  |
| --- |
|  |
| Figure 2. Plots of CV() for two different design scenarios (a) equal VM fractions and (b) unequal VM fractions. |

*Design problem #2*.— In this problem, the benefit of having a high PBT fraction is demonstrated. Assume that a carcass sample of size N=40 is drawn from the spawning grounds and 50% of the spawners are of hatchery origin. Further assume that there are two source hatcheries that contribute spawners to the spawning grounds, hatcheries A and B, where each contributes 25% of the total spawners. Also assume that the PBT fraction at both hatcheries are identical and vary over the values 0, 0.1, 0.2,…, 1.0. As in the last design problem, there are two different scenarios we wish to test: (a) equal VM fractions of 0.5 (); and (b) unequal VM fractions of  and , at hatcheries A and B, respectively. In each of these cases, we assume that a subsample of is drawn from the sample of 40. Assume that  and  are chosen so that CV() is as small as possible. Show how CV() under scenarios (a) and (b) varies with PBT fraction.

To determine the minimum CV() for a particular design with , CV() was calculated for each feasible pair of  and , and the smallest CV() over this set of pairs was chosen as the minimum value.[[2]](#footnote-2) The function phos.pbt.estimates from Appendix B was used to calculate the CV() associated with the alternative designs. The smallest CV() over feasible values of  and  were then plotted against PBT fraction.

The results of this analysis are shown in Figure 3. The CV() declines with the PBT fraction for both the (a) equal VM fraction and (b) unequal VM fraction scenarios. Also, when PBT fraction is small, the CV of scenario (a) has lower CV than scenario (b) even though (a) has smaller VM fractions. This occurs because when VM fractions are the same, estimating *p* does not hinge on numbers of PBT detections in adults: a viable estimate is possible without knowing anything about the PBT detections (see equation (23)). However, when VM fractions differ, then estimating *p* does hinge on the PBT detections. Thus when VM fraction differ and expected PBT detections are low (due to small values of PBT fractions), the precision of  is expected to be poor. This was also observed in Hinrichsen et al. (2012).

|  |
| --- |
|  |
| Figure 3. The CV() of scenarios (a) equal VM fractions () and (b) unequal VM fractions ( and ) plotted against the value of PBT fraction. |

*Design problem #3*.— In this problem, the benefit of having a high PBT fraction is demonstrated. Assume that a carcass sample of size N=100 is drawn from the spawning grounds and 25% of the spawners are of hatchery origin. Further assume that there are two source hatcheries that contribute spawners to the spawning grounds, hatcheries A and B, where each contributes 12.5% of the total spawners. Also assume that the PBT fraction at both hatcheries is 0.95. As in the last design problem, there are two different scenarios we wish to test: (a) equal VM fractions of 0.5 (); (b) and unequal VM fractions of  and , at hatcheries A and B, respectively. In each of these cases, we assume that a subsample of  is drawn from the total sample if size 100. Assume that  and  are chosen so that CV() is as small as possible. Show how CV() under scenarios (a) and (b) varies with the subsample size *n*. Show also how the optimal value of  changes with *n*.

|  |
| --- |
|  |
|  |
| Figure 4. CV() plotted against the total sample size. The optimal  and  represent the values of  and that minimize CV() subject to the constraints , , and  using two scenarios: scenarios (a) equal VM fractions () and (b) unequal VM fractions ( and ).  The results of this analysis are shown in Figure 4. When VM fractions were equal (i.e., ) it was always optimal to send more unmarked spawners to the lab than marked. In contrast, when VM fractions are unequal (i.e.,  and ), notice that for small total sample sizes (*n*), the optimal policy was to send only VM fish to the lab for PBT testing (Figure 4b). As the total sample size grows toward its maximum value of 100, the optimal policy was to send  and  for PBT testing. (In practice, these fractional numbers of fish should be rounded to the nearest whole number). Over the majority of the range of subsample sizes, it was optimal to send more unmarked spawners to the lab for genetic sampling than marked spawners. |

*Design problem #4*.—In this case, the benefit of visibly marking hatchery releases is demonstrated. Assume that 25% of the spawners are of hatchery origin and that there are two source hatcheries hatcheries A and B, and each contributes 12.5% of the total spawners. Also assume that equal VM fractions are used () where  ranges over the values 0.0, 0.1,…,1.0, all sampled carcasses are tested for PBT, and three alternative sample sizes N=25, 50, and 100. Also assume that the PBT fraction at both hatcheries takes on the alternative values of 0.50 and 0.95. Demonstrate how CV() changes with increasing VM fraction, . How does the relationship between CV() and change with sample size and PBT fraction?

This design problem is solved by plotting a graph of CV() versus  using the alternative sample sizes N=25, 50, and 100 and alternative PBT fractions of 0.50 and 0.95 (Figure 5). Notice that when PBT fraction is small (0.50), the marking fraction has a large effect on CV(), but when PBT is large (0.95), the marking fraction has little effect on CV(). This shows that when PBT is high for all source hatcheries, and all sampled carcasses are tested for a PBT, then visibly marking releases does little to increase the precision of the estimate of *p*.

|  |
| --- |
| (a) |
| (b) |
| Figure 5. CV() plotted against VM fraction for three alternative sample sizes N=25, 50, and 100, when PBT fractions are (a) 0.50 and (b) 0.95. All sampled carcasses were tested for PBT. |

The above results for design problem #4 were based on the assumption that 100% of the sampled carcasses were sent to the lab. How do the results change if 50% of the sampled carcasses are randomly selected and sent to the lab for testing? This sensitivity analysis is illustrated in Figure 6. Notice that when 50% of the sampled carcasses are tested for PBT instead of 100% that the CV() is much more sensitive to the VM fraction. This demonstrates that when not all of the sampled carcasses are tested for PBT, it is possible that increasing the VM marking fraction will greatly increase precision of, even when PBT fraction is high (0.95).

|  |
| --- |
| (a) |
| (b) |
| Figure 6. CV() plotted against VM fraction for three alternative sample sizes N=25, 50, and 100, when PBT fractions are (a) 0.50 and (b) 0.95. Half of the sampled carcasses were tested for PBT. |

# Acknowledgements

The author gratefully acknowledges the reviews of Rishi Sharma, Shawn Narum, Matt Campbell, Mike Ackerman, Craig Steele, Maureen Hess, Bill Young, and Barbara Shields. This work was supported by Bonneville Power Administration. Brian Maschhoff implemented this statistical analysis as a web-based tool at [www.onefishtwofish.net](http://www.onefishtwofish.net). Amber Parsons reviewed this documentation and tested the web-based tool HATCHPBT. The views expressed are solely those of the author and are not intended to represent the views of any organization with which the author is affiliated.

# References

Anderson, E.C. 2010. Computational algorithms and user-friendly software Computational algorithms and user-friendly software for parentage-based tagging of Pacific salmonids. Final report submitted to the Pacific Salmon Commission’s Chinook Technical Committee (US Section). 46 p. Available: http://swfsc.noaa.gov/uploadedFiles/Divisions/FED/Staff\_Pages/Eric\_Anderson/PBT\_PSC\_final\_report.pdf.

Anderson, E. C., and J. C. Garza. 2005. A description of full genotyping. Report submitted to the Pacific Salmon Commission, Vancouver, British Columbia. 11p. Available: http://swfsc.noaa.gov/publications/FED/00675.pdf.

Anderson E. C., and J. C. Garza. 2006. The power of single-nucleotide polymorphisms for large-scale parentage inference. Genetics 172: 2567–2582.

Hankin, D.G. 1982. Estimating escapement of pacific salmon: marking practices to discriminate wild and hatchery fish. Transpactions of the American Fisheries Society 111:286-298.

HSRG (Hatchery Scientific Review Group). 2009. Columbia River hatchery reform systemwide report. Available: [www.hatcheryreform.us](http://www.hatcheryreform.us).

Hinrichsen, R. A. 2003. The power of experiments for estimating the relative reproductive success of hatchery-origin spawners. Canadian Journal of Fisheries and Aquatic Sciences 60:864–872.

Hinrichsen, R.A., R. Sharma, T.R. Fisher. 2012. Precision and accuracy of estimators of the proportion of hatchery-origin spawners. Transactions of the American Fisheries Society 142:437-454.

Jennrich, R. I., and P. F. Sampson. 1976. Newton-Raphson and related algorithms for maximum likelihood variance component estimation. Technometrics 18:11-17.

Kariya, T. and H. Kurata. 2004. Generalized Least Squares. Wiley Series in Probability and Statistics. Wiley. New York, New York.

McClure, M.M., E.E. Holmes, B.L. Sanderson, and C.E. Jordan. 2003. A large-scale multispecial status assessment: anadromous salmonids in the Columbia River basin. Ecological Applications 13:964-989.

Mobrand, L. E., J. Barr, L. Blankenship, D. D. Campton, T. T. P. Eveleyn, T.A. Flagg, C. V. W. Mahnken, L. W. Seeb, P. R. Seidel, and W. W. Smoker. 2005. Hatchery reform in Washington State: principles and emerging issues. Fisheries 30(6):11–23.

Mood, A. M., F. A. Graybill, and D. C. Boes. 1974. Introduction to the theory of statistics, 3rd edition. McGraw-Hill, New York.

Press, W. H., S.A., Teukolsky, W.T. Vetterling, and B.P. Flannery, Brian P. 1992. Numerical Recipesin Fortran: The Art of Scientific Computing (2rd ed.). New York: Cambridge University Press.

Steele, C., M. Ackerman, J. McCane, M. Campbell, M. Hess, N. Campbell, and S. Narum. 2011. Parentage-based tagging of Snake River hatchery steelhead and Chinook salmon. 2010 Annual Report. Idaho Department of Fish and Game and Columbia River Inter-Tribal Fish Commision. IDFG Report Number 11-111. June 2011. Available: https://research.idfg.idaho.gov/Fisheries%20Research%20Reports/Res11-111Steele2010%20Parentage%20Based%20Tagging%20Snake%20River%20Steelhead%20Salmon.pdf

Waples, R.S. 1991. Genetic interactions between hatchery and wild salmonids: lessons from the Pacific Northwest. Canadian Journal of Fisheries and Aquatic Sciences. 48(Suppl. 1): 124-133.

# Appendix A. Names of variables

|  |  |  |
| --- | --- | --- |
| Table A.1.—Variables. | | |
| R code | Mathematical derivation | Definition |
| i | *i* | Hatchery index |
| nhatch | *m* | Number of source hatcheries |
| phos | *p* | Proportion of hatchery-origin spawners |
| phosi[i] |  | Hatchery-specific proportion of hatchery-origin spawners |
| ppbt[i] |  | PBT fraction |
| lambda[i] |  | VM fraction |
| Nsamp | *N* | Total sample size |
| x1 |  | Number of VM spawners in total sample |
| x2 |  | Number of unmarked spawners in total sample |
| y[i] |  | Number of subsampled VM spawners testing positive for PBT from hatchery *i*. |
| z[i] |  | Number of the subsampled unmarked spawners testing positive for PBT from hatchery *i*. |
| n |  | Subsample size |
| n1 |  | Number of VM spawners tested for PBT |
| n2 |  | Number of unmarked spawners tested for PBT |
| Ex1 |  | Expected value of |
| Ex2 |  | Expected value of |
| I | **I** | Fisher Information Matrix |
| SE.phoshat |  | Standard error of the estimate of the proportion of hatchery-origin spawners |
| CV.phoshat |  | Coefficient of variation of the estimate of the proportion of hatchery-origin spawners |
| SE\_MIN.phoshat |  | Minimum standard error of the estimate of the proportion of hatchery-origin spawners, achieved with and *,* where *n = N.* |
| CV\_MIN.phoshat |  | Minimum coefficient of variation of the estimate of the proportion of hatchery-origin spawners achieved with and *,* where *n = N.* |
| SE2.phoshat |  | Monte Carlo standard error of the estimate of the proportion of hatchery-origin spawners |
| CV2.phoshat |  | Monte Carlo coefficient of variation of the estimate of the proportion of hatchery-origin spawners |
| SE\_MIN2.phoshat |  | Monte Carlo minimum standard error of the estimate of the proportion of hatchery-origin spawners, achieved with and *,* where *n = N.* |
| CV\_MIN2.phoshat |  | Monte Carlo minimum coefficient of variation of the estimate of the proportion of hatchery-origin spawners achieved with and *,* where *n = N.* |
| BIAS2.phoshat |  | Monte Carlo relative bias which is bias divided by the true value of *p*. |
|  |  |  |

# Appendix B. R-code

|  |
| --- |
| Table B.1. R-code |
|  |

#Program to calculate properties of phos estimates using Monte Carlo simulation and theoretical results

#This code treats the general case of inputs from several hatcheries with potentially different marking fractions

#and different fraction of marked fish given a parentage-based tag.

#FILE: pbt-web-10-25-2012.s

#AUTHOR: Richard A. Hinrichsen, 25 October 2012

#REVISION: Used n as an input instead of n2 (n2 is calculated as n2=n-n1)

#Variables and parameters used in the analysis

#inputs

#phosi = true proportions of hatchery origin spawners (hatchery-specific)

#Nsamp = total number of spawners sampled on spawning grounds

#n = total number of spawners sent to the lab and checked for PBT

#n1 = number of marked spawners sent to the lab and checked for PBT

#lambda = marking rate (lambda) (hatchery-specific)

#ppbt=fraction of marked fish that are also parentage-based tagged (hatchery-specific)

#MONTE = FALSE for theoretical results, TRUE for Monte Carlo results

#NSIM = number of Monte Carlo simulations (needed if MONTE=TRUE)

#

#

#Select intermediate variables

#nhatch=number of hatcheries supplying spawners in the wild

#n2 = n - n1

#I = Fisher Information Matrix

#I2 = FIsher Information Matrix giving minimum SE for phos (occurs when n=N)

#Results

#phos = true proportion of hatchery-origin spawners

#Ex1 = expected number of VM spawners (summing over hatcheries)

#Ex2 = expected number of not VM spawners (summing over hatcheries)

#SE\_MIN.phoshat = standard error (SE) when ALL sampled fish genetically tested

#CV\_MIN.phoshat = Coefficient of variation when ALL sampled fish genetically tested

#SE.phoshat = standard error (SE)

#CV.phoshat = Coefficient of variation

#BIAS.phoshat = relative bias estimate (NA if MONTE=FALSE)

#top level function

phos.pbt.main<-function(phosi=.1\*c(1/20,1/20,9/20,9/20),

Nsamp=1000,n=200,n1=50,lambda=c(1,.95,.5,.5),ppbt=c(.95,.95,.95,.95),

MONTE=FALSE,NSIM=1000){

check.inputs(phosi,Nsamp,n,n1,lambda,ppbt,MONTE,NSIM)

if(!MONTE){res<-phos.pbt.estimates(phosi=phosi,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda,ppbt=ppbt)}

if(MONTE){res<-phos.pbt.estimates2(NSIM=NSIM,phosi=phosi,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda,ppbt=ppbt)}

return(res)

}

check.inputs<-function(phosi,Nsamp,n,n1,lambda,ppbt,MONTE,NSIM){

if(!is.logical(MONTE))stop("MONTE must be TRUE or FALSE")

if(MONTE){

if(floor(NSIM)!=NSIM){stop("NSIM must be a positive integer")}

if(NSIM<=0){stop("NSIM must be a positive integer")}}

if(floor(Nsamp)!=Nsamp){stop("Nsamp must be a positive integer")}

if(Nsamp<=0){stop("Nsamp must be a positive integer")}

#check dimension of inputs

k1<-length(phosi);k2<-length(lambda);k3<-length(ppbt)

mytest<-abs(k1-k2)+abs(k2-k3)

if(mytest>0) stop("dimensions of phosi, lambda, and ppbt must match")

nhatch<-length(phosi)

#check that the subsample size is less than sample size

if(n>Nsamp)stop(paste("n must be less than or equal to Nsamp=",Nsamp))

#check ppbts (must all exceed zero)

if(sum(ppbt<=0))stop("ppbts must all be greater than zero")

#check that each ppbt is less than or equal to one

if(sum(ppbt>1))stop("ppbts must all be less than or equal to 1.0")

#check that all lambdas are between zero and 1.0

if(sum(lambda<0))stop("lambdas must all be greater than or equal to zero")

if(sum(lambda>1))stop("lambdas must all be less than or equal to one")

#check that all phosi are between zero and 1.0

if(sum(phosi<=0))stop("phosi must all be greater than zero")

if(sum(phosi>1))stop("phosi must all be less than or equal to one")

#fail if n=0 and lambdas not constant

if((n==0)&(sum(lambda==mean(lambda))< nhatch)){

stop("Error: subsample sizes are zero and lambdas vary among hatcheries. Try setting n1>0 or n>n1")

}

if(sum(lambda==0)&n==n1){

stop("Error: Some lambdas are zero and n = n1. Try setting n > n1.")

}

#get expected values of observations

Ex1<-Nsamp\*sum(lambda\*phosi)

Ex2<-Nsamp-Ex1

#check subsample

if(n1<0)stop("n1 must be nonnegative")

if(n<n1)stop("n must be greater than or equal to n1")

if(n1>Ex1)stop(paste("n1 must not exceed the expected number of VM spawners in sample=",Ex1))

if(n1<(n-Ex2))stop(paste("n1 must be >= n minus the expected number of ~VM spawners in sample=",n-Ex2))

return(NULL)

}

#Theoretical results

phos.pbt.estimates<-function(phosi=.1\*c(1/20,1/20,9/20,9/20),

Nsamp=1000,n=200,n1=50,lambda=c(1,.95,.5,.5),ppbt=c(.95,.95,.95,.95)){

check.inputs(phosi=phosi,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda,ppbt=ppbt,MONTE=FALSE,NSIM=NA)

n2<-n-n1

nhatch<-length(lambda)

#get expected values of observations

Ex1<-Nsamp\*sum(lambda\*phosi)

Ex2<-Nsamp-Ex1

phos<-sum(phosi)

#check case of n1=0,n-n1=0 and constant lambdas

if((n==0)&(sum(lambda==mean(lambda))==nhatch)){

res<-onehatchery.estimates(phos=phos,lambda=mean(lambda),Nsamp=Nsamp)

res2<-phos.pbt.estimates(phosi=phosi,Nsamp=Nsamp,n=Nsamp,n1=Ex1,lambda=lambda,ppbt=ppbt)

myres<-list(MONTE=FALSE,

NSIM=NA,

phosi=phosi,

Nsamp=Nsamp,

n=n,

n1=n1,

lambda=lambda,

ppbt=ppbt,

phos=phos,

Ex1=Ex1,

Ex2=Ex2,

SE\_MIN.phoshat=res2$SE\_MIN.phoshat,

CV\_MIN.phoshat=res2$CV\_MIN.phoshat,

SE.phoshat=res$SE.phoshat,

CV.phoshat=res$CV.phoshat,

BIAS.phoshat=NA)

return(myres)

}

#combine cells with lambda=1 into a single cell and try again!

if((sum(lambda==1)>1)&(n1==0)){

warning("collapsing cells with lambda=1 into a single cell since n1=0")

iii<-lambda==1

lambda1.new<-1.0

ppbt1.new<-sum(ppbt[iii]\*phosi[iii])/sum(phosi[iii])

phosi1.new<-sum(phosi[iii])

lambda.new<-c(lambda1.new,lambda[!iii])

ppbt.new<-c(ppbt1.new,ppbt[!iii])

phosi.new<-c(phosi1.new,phosi[!iii])

res<-phos.pbt.estimates(phosi=phosi.new,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda.new,ppbt=ppbt.new)

myres<-list(MONTE=FALSE,

NSIM=NA,

phosi=phosi,

Nsamp=Nsamp,

n=n,

n1=n1,

lambda=lambda,

ppbt=ppbt,

phos=phos,

Ex1=Ex1,

Ex2=Ex2,

SE\_MIN.phoshat=res$SE\_MIN.phoshat,

CV\_MIN.phoshat=res$CV\_MIN.phoshat,

SE.phoshat=res$SE.phoshat,

CV.phoshat=res$CV.phoshat,

BIAS.phoshat=NA)

return(myres)

}

#check to see if all pbts are one (special case)

pbttest<-FALSE

if(sum(ppbt==1)==nhatch)pbttest<-TRUE

#check to see if all lambdas are zero (special case)

lambdatest<-FALSE

if(sum(lambda==0)==nhatch)lambdatest<-TRUE

#get Fisher Information Matrix

if(lambdatest){

I<-getI3(Nsamp,n1,n2,lambda,ppbt,phosi)

I2<-getI3(Nsamp,Ex1,Ex2,lambda,ppbt,phosi)

}

else{

if(pbttest){

I<-getI2(Nsamp,n1,n2,lambda,ppbt,phosi)

I2<-getI2(Nsamp,Ex1,Ex2,lambda,ppbt,phosi)

}

else{

I<-getI(Nsamp,n1,n2,lambda,ppbt,phosi)

I2<-getI(Nsamp,Ex1,Ex2,lambda,ppbt,phosi)

}

}

#check to see if Fisher Information Matrix is computationally singular

if(is.na(rcond(I)))stop("Condition number of I is NA")

if(rcond(I)<1.e-16)stop("Fisher Information matrix I is computationally singular")

varmat<-solve(I)

e<-rep(1,nhatch)

phos.var<-t(e)%\*%varmat%\*%e

if(is.na(rcond(I2)))stop("Condition number of I2 is NA")

#get var when all sampled spawners sent to the lab

if(rcond(I2)<1.e-16)stop("Fisher Information matrix I2 is computationally singular")

varmat2<-solve(I2)

e<-rep(1,nhatch)

phos2.var<-t(e)%\*%varmat2%\*%e

SE.phoshat<-sqrt(phos.var)

CV.phoshat<-SE.phoshat/phos

SE\_MIN.phoshat<-sqrt(phos2.var)

CV\_MIN.phoshat<-SE\_MIN.phoshat/phos

SE\_MIN.phoshat<-as.numeric(SE\_MIN.phoshat)

CV\_MIN.phoshat<-as.numeric(CV\_MIN.phoshat)

SE.phoshat<-as.numeric(SE.phoshat)

CV.phoshat<-as.numeric(CV.phoshat)

myres<-list(MONTE=FALSE,

NSIM=NA,

phosi=phosi,

Nsamp=Nsamp,

n=n,

n1=n1,

lambda=lambda,

ppbt=ppbt,

phos=phos,

Ex1=Ex1,

Ex2=Ex2,

SE\_MIN.phoshat=SE\_MIN.phoshat,

CV\_MIN.phoshat=CV\_MIN.phoshat,

SE.phoshat=SE.phoshat,

CV.phoshat=CV.phoshat,

BIAS.phoshat=NA)

return(myres)

}

#get SE and CV in single hatchery case (lambdas constant and n1=n2=0)

onehatchery.estimates<-function(phos,Nsamp,lambda){

SE.phoshat<-phos\*(1-phos\*lambda)/(Nsamp\*lambda)

SE.phoshat<-sqrt(SE.phoshat)

CV.phoshat<-SE.phoshat/phos

return(list(SE.phoshat=SE.phoshat,CV.phoshat=CV.phoshat))

}

#Fisher Information Matrix

getI<-function(Nsamp,n1,n2,lambda,ppbt,phosi){

Ex1<-Nsamp\*sum(lambda\*phosi)

Ex2<-Nsamp-Ex1

theta1<-n1/Ex1

theta2<-n2/Ex2

v<-lambda

I<- v%\*%t(v)\*Nsamp\*(1-theta1)/sum(v\*phosi)

I<-I +v%\*%t(v)\*Nsamp\*(1-theta2)/(1-sum(v\*phosi))

v<-(1-ppbt)\*lambda

I<-I+v%\*%t(v)\*Nsamp\*theta1/sum(v\*phosi)

v<-lambda\*(1-ppbt)+ppbt

I<-I+v%\*%t(v)\*Nsamp\*theta2/(1-sum(v\*phosi))

#fix diagonal

mydiag<-diag(I)+Nsamp\*ppbt\*(theta1\*lambda+theta2\*(1-lambda))/phosi

diag(I)<-mydiag

return(I)

}

#Fisher Information matrix (used when all pbts are one)

getI2<-function(Nsamp,n1,n2,lambda,ppbt,phosi){

Ex1<-Nsamp\*sum(lambda\*phosi)

Ex2<-Nsamp-Ex1

theta1<-n1/Ex1

theta2<-n2/Ex2

v<-lambda

I<- v%\*%t(v)\*Nsamp\*(1-theta1)/sum(v\*phosi)

I<-I +v%\*%t(v)\*Nsamp\*(1-theta2)/(1-sum(v\*phosi))

I<-I+Nsamp\*theta2/(1-sum(phosi))

#fix diagonal

mydiag<-diag(I)+Nsamp\*(theta1\*lambda+theta2\*(1-lambda))/phosi

diag(I)<-mydiag

return(I)

}

#Fisher Information matrix (used when all lambdas are zero)

getI3<-function(Nsamp,n1,n2,lambda,ppbt,phosi){

Ex2<-Nsamp

theta2<-n2/Ex2

v<-ppbt

I<-v%\*%t(v)\*Nsamp\*theta2/(1-sum(v\*phosi))

#fix diagonal

mydiag<-diag(I)+Nsamp\*ppbt\*theta2/phosi

diag(I)<-mydiag

return(I)

}

#Monte Carlo estimates of standard error and bias

phos.pbt.estimates2<-function(NSIM=1000,phosi=.1\*c(1/20,1/20,9/20,9/20),

Nsamp=1000,n=n,n1=50,lambda=c(1,.95,.5,.5),ppbt=c(.95,.95,.95,.95)){

check.inputs(phosi=phosi,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda,ppbt=ppbt,MONTE=TRUE,NSIM=NSIM)

n2<-n-n1

nhatch<-length(phosi)

#get expected values of observations

Ex1<-Nsamp\*sum(lambda\*phosi)

Ex2<-Nsamp-Ex1

phos<-sum(phosi)

#check for singularity. Combine cells with lambda=1 into a single cell and try again!

if((sum(lambda==1)>1)&(sum(lambda==1)< nhatch)&(n1==0)){

warning("collapsing cells with lambda=1 into a single cell since n1=0")

iii<-lambda==1

lambda1.new<-1.0

ppbt1.new<-sum(ppbt[iii]\*phosi[iii])/sum(phosi[iii])

phosi1.new<-sum(phosi[iii])

lambda.new<-c(lambda1.new,lambda[!iii])

ppbt.new<-c(ppbt1.new,ppbt[!iii])

phosi.new<-c(phosi1.new,phosi[!iii])

res<-phos.pbt.estimates2(NSIM=NSIM,phosi=phosi.new,Nsamp=Nsamp,n=n,n1=n1,lambda=lambda.new,ppbt=ppbt.new)

myres<-list(MONTE=TRUE,

NSIM=NSIM,

phosi=phosi,

Nsamp=Nsamp,

n=n,

n1=n1,

lambda=lambda,

ppbt=ppbt,

phos=phos,

Ex1=Ex1,

Ex2=Ex2,

SE\_MIN.phoshat=res$SE\_MIN.phoshat,

CV\_MIN.phoshat=res$CV\_MIN.phoshat,

SE.phoshat=res$SE.phoshat,

CV.phoshat=res$CV.phoshat,

BIAS.phoshat=res$BIAS.phoshat)

return(myres)

}

phos.sim<-rep(NA,NSIM)

for(ii in 1:NSIM){

phos.sim[ii]<-get.estimates2(phosi=phosi,Nsamp=Nsamp,n1=n1,n2=n2,lambda=lambda,ppbt=ppbt)

}

SE.phoshat<-sqrt(var(phos.sim,na.rm=T))

CV.phoshat<-SE.phoshat/phos

mymean<-mean(phos.sim,na.rm=T)

BIAS.phoshat<-(mymean-phos)/phos

#Now get minimum variance and CV

phos.sim<-rep(NA,NSIM)

for(ii in 1:NSIM){

phos.sim[ii]<-get.estimates2(phosi=phosi,Nsamp=Nsamp,n1=Ex1,n2=Ex2,lambda=lambda,ppbt=ppbt)

}

SE\_MIN.phoshat<-sqrt(var(phos.sim,na.rm=T))

CV\_MIN.phoshat<-SE\_MIN.phoshat/phos

myres<-list(MONTE=TRUE,

NSIM=NSIM,

phosi=phosi,

Nsamp=Nsamp,

n=n,

n1=n1,

lambda=lambda,

ppbt=ppbt,

phos=phos,

Ex1=Ex1,

Ex2=Ex2,

SE\_MIN.phoshat=SE\_MIN.phoshat,

CV\_MIN.phoshat=CV\_MIN.phoshat,

SE.phoshat=SE.phoshat,

CV.phoshat=CV.phoshat,

BIAS.phoshat=BIAS.phoshat)

return(myres)

}

#simulate data

pbtsim<-function(phosi,Nsamp,n1,n2,lambda,ppbt){

m<-length(lambda)

#first get binomial sample of fish marked and unmarked

P<-sum(phosi\*lambda)

x1<-rbinom(n=1,size=Nsamp,prob=P)

x2<-Nsamp-x1

#next use multinomial distribution to simulate

#how many fish are pbt and how many are not pbt

py<-ppbt\*phosi\*lambda/P

pz<-ppbt\*phosi\*(1-lambda)/(1-P)

y<-rmultinom(n=1,size=min(n1,x1),prob=c(py,1-sum(py)))

z<-rmultinom(n=1,size=min(n2,x2),prob=c(pz,1-sum(pz)))

return(list(Nsamp=Nsamp,n1=n1,n2=n2,x1=x1,x2=x2,y=y[1:m],z=z[1:m]))

}

#Use Fisher's scoring algorithm to estimate phosi

get.estimates2<-function(phosi,Nsamp,n1,n2,lambda,ppbt){

NTRIAL<-100

tolx<-1.e-5

#simulate data

res<-pbtsim(phosi=phosi,Nsamp=Nsamp,n1=n1,n2=n2,lambda=lambda,ppbt=ppbt)

x1<-res$x1

x2<-res$x2

y<-res$y

z<-res$z

nhatch<-length(phosi)

w<-y+z

iii<-(w>0)

if(sum(iii)==0){

if(sum(lambda==mean(lambda))==nhatch){

if(mean(lambda)>0)return((x1/Nsamp)/mean(lambda))

}

return(NA)

}

#reduce the dimension if zeroes present. In this case phosis associated with zeroes

#are estimated to be zero.

y<-y[iii]

z<-z[iii]

ppbt<-ppbt[iii]

lambda<-lambda[iii]

phosi<-phosi[iii]

nhatch=length(y)

#check for special cases

#check to see if all pbts are one (special case)

pbttest<-FALSE

if(sum(ppbt==1)==nhatch)pbttest<-TRUE

#check to see if all lambdas are zero (special case)

lambdatest<-FALSE

if(sum(lambda==0)==nhatch)lambdatest<-TRUE

#get the right Fisher Information Matrix

if(lambdatest){

my.getI<-getI3

}

else{

if(pbttest){

my.getI<-getI2

}

else{

my.getI<-getI

}

}

#use Fisher's scoring algorithm method to find where the partial derivatives of the

#log-likelihood are zero. Use Fisher Information matrix in Newton's method to approx -Hessian

#set the initial guess equal to the true phosi

phosi.hat<-phosi

errf<-0.0

alpha<-0.9

for(ii in 1:NTRIAL){

I<-my.getI(Nsamp,n1,n2,lambda,ppbt,phosi.hat)

df<-dlike(phosi.hat,x1,x2,y,z,Nsamp,n1,n2,lambda,ppbt)

if(is.na(rcond(I))){warning("condition number of I is NA");return(NA)}

if(rcond(I)<1.e-15){warning("singular information matrix");return(NA)}

delx<-solve(I)%\*%df

phosi.hat<-phosi.hat+delx\*(1-alpha)

errx<-sum(abs(delx))/sum(abs(phosi))

alpha<-alpha\*alpha

if(errx<=tolx)break

}

if(ii==NTRIAL){warning("maximum number of iterations was reached");return(NA)}

return(sum(phosi.hat))

}

#get gradient of the log likelihood function

#phosi is the current best estimate of phosi

dlike<-function(phosi,x1,x2,y,z,Nsamp, n1,n2,lambda,ppbt){

#estimate phosi

sum1<-sum(lambda\*phosi)

sum2<-sum((1-ppbt)\*lambda\*phosi)

res<-lambda\*(x1-n1)/sum1-lambda\*(Nsamp-x1-n2)/(1-sum1)+y/phosi

res<-res+(n1-sum(y))\*(1-ppbt)\*lambda/sum2+z/phosi

res<-res-(n2-sum(z))\*(lambda\*(1-ppbt)+ppbt)/(1-sum1-sum2)

return(res)

}

1. On June 27, 2007, the House passed (amended) H.R. 2643, including a provision requiring the U.S. Fish and Wildlife Service to implement a system of mass marking of salmonid stocks that are released from federal hatcheries [↑](#footnote-ref-1)
2. Subsamples were constrained so that  did not exceed and that  did not exceed . [↑](#footnote-ref-2)