HatchPBTE

A statistical tool to estimate the proportion of hatchery-origin spawners using parentage-based tagging

Richard A. Hinrichsen

August 20, 2013

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 estimates, it is impossible to estimate trends in the wild-origin population abundance or to 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] 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 reliable estimation techniques have been lacking until recently. 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.

[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

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 (Anderson and Garza 2005; Anderson and Garza 2006; Steele et al. 2011; Steele et al. 2013). 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. 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 hatchery-origin spawners, not VM hatchery-origin 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 estimating the proportion of hatchery-origin spawners in a program that uses VM and PBT to identify hatchery-origin spawners. Equations for a maximum likelihood estimator and its precision and bias are developed. The estimator allows the possibility of two or more source hatcheries. In general, these source hatcheries may use different VM fractions and (possibly) different PBT fractions.

METHODS

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 (Kariya and Kurata 2004; Hinrichsen et al. 2012). 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 calculated using the Hessian of the likelihood function (Mood et al. 1974).

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. 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. The steps for collecting spawner data necessary to estimate the proportion of hatchery-origin spawners are summarized as follows:

1) Take a random sample of size *N* of the spawners in the wild. This sample will consist of two groups: fish that are VM and fish without a VM (i.e., ~VM)
2) Take a random subsample of size n_1 from the VM group and a random subsample of size n_2 from the ~VM group. Note that the total subsample size is $n = n_1 + n_2$.
3) Test these subsamples for PBT. Note the hatchery of origin for each fish that is PBT.

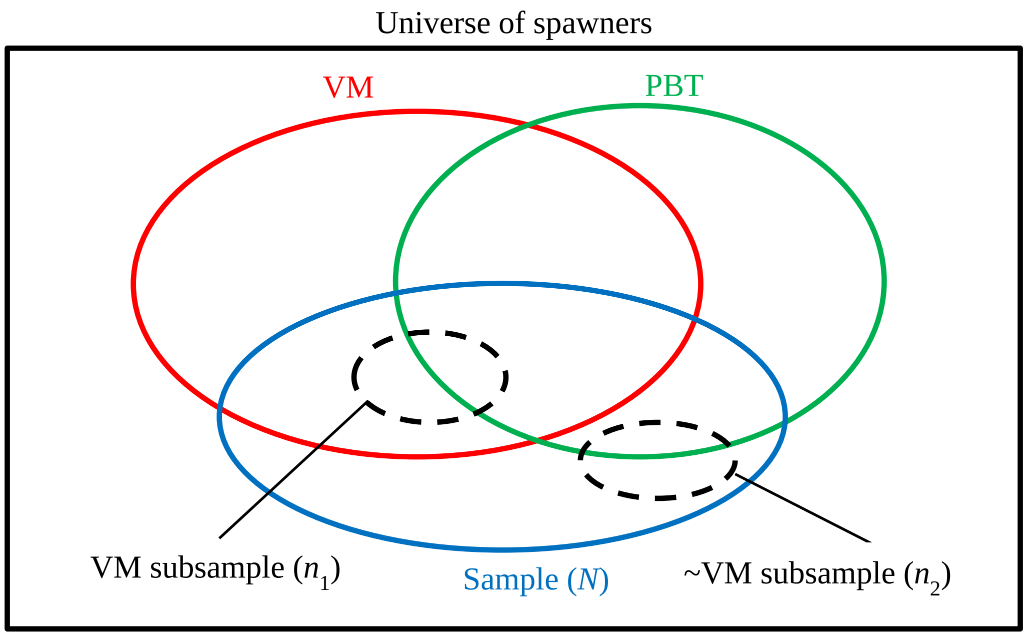

Figure 1.—Carcass sampling represented by a Venn diagram. The universe is represented by all returning spawners to a particular spawning ground in the wild. The sample of size *N* is a random sample of the universe of spawners. The subsamples of sizes n_1 and n_2 are random samples of the VM spawners and not VM (~VM) spawners, respectively. These subsamples, totaling $n = n_1 + n_2$, are checked for a PBT, while the remaining carcasses, totaling $N - n$, 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. 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.

Let p_i represent the fraction of spawners on the spawning grounds that originated at hatchery *i* out of m total hatcheries. Let λ_i represent the VM fraction that is applied to hatchery fish releases from hatchery *i*, and let ϕ_i 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 x_1 represents the number of sampled fish that are VM and $x_2 = N - x_1$ represents the number of sampled fish that are not VM. Here, x_1 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 $\sum_{i=1}^{m} \lambda_i p_i$.

Table 1.—Assumptions[1]

(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 tested for a PBT that allows investigators to determine the hatchery of origin. A total of n_1 VM spawners are genotyped and n_2 unmarked spawners are genotyped to determine whether they are PBT. The total subsample size is then $n = n_1 + n_2$. To make the subsample sizes meaningful, I used the following restrictions: $n_1 \le x_1$, $n_2 \le x_2$, and $0 < n \le N$. I use these restrictions because n_1 must be less than or equal to the number of VM fish in the sample (x_1) and n_2 must be less than or equal to the number of unmarked fish in the total sample (x_2), and the subsample size must be less or equal to the total sample size. The number of genotyped marked fish originating at hatchery i and determined to have a PBT is denoted by the random variable y_i and the number of genotyped unmarked fish originating at hatchery i and determined to have a PBT is z_i . The values y_i are considered cell counts from a multinomial distribution with n_1 trials with cell probabilities $\lambda_i \phi_i p_i / \sum_{i=1}^{m} \lambda_i p_i$. The values z_i are cell counts from a multinomial distribution with n_2 trials and cell probabilities $(1-\lambda_i)\phi_i p_i / \left(1-\sum_{i=1}^{m} \lambda_i p_i\right)$.

Some important assumptions are that all hatchery fish are released randomly, 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 vice versa. (i.e., the event that a hatchery fish is VM is independent of the event that it is PBT). Hatchery managers need to be aware that all hatchery fish should be treated equally. If any group is treated in a different manner, the assumptions in Table 1 will be violated. Likewise, in the second sampling sampling event (i.e., drawing of the subsamples), the subsamples taken should be truly random and representative of the populations.

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

$$f(x_1, x_2, y_1, \ldots y_m, z_1, \ldots z_m) = \binom{N}{x_1, x_2} \left(\sum_{i=1}^{m} \lambda_i p_i \right)^{x_1} \left(1 - \sum_{i=1}^{m} \lambda_i p_i \right)^{x_2} \tag{1}$$

$$\times \binom{n_1}{y_1, \ldots y_m, n_1 - \sum_{i=1}^{m} y_i} \prod_{i=1}^{m} \left(\frac{\lambda_i \phi_i p_i}{\sum_{k=1}^{m} \lambda_k p_k} \right)^{y_i} \left(1 - \frac{\sum_{k=1}^{m} \lambda_k \phi_k p_k}{\sum_{k=1}^{m} \lambda_k p_k} \right)^{n_1 - \sum_{k=1}^{m} y_k}$$

$$\times \binom{n_2}{z_1, \ldots z_m, n_2 - \sum_{i=1}^{m} z_i} \prod_{i=1}^{m} \left[\frac{(1-\lambda_i)\phi_i p_i}{1 - \sum_{k=1}^{m} \lambda_k p_k} \right]^{z_i} \left[1 - \frac{\sum_{k=1}^{m} (1-\lambda_k)\phi_k p_k}{1 - \sum_{k=1}^{m} \lambda_k p_k} \right]^{n_2 - \sum_{k=1}^{m} z_k} .$$

Given the joint distribution of the observations in equation (1), it is now possible to form the log-likelihood function of the unknown parameters $p_1, p_2, \ldots p_m$ by taking the natural log of the joint distribution and treating the result as a function of the parameters:

$$
\begin{aligned}
l(p_1, p_2 \ldots, p_m) = {} & x_1 \log\left(\sum_{i=1}^{m} \lambda_i p_i\right) + x_2 \log\left(1 - \sum_{i=1}^{m} \lambda_i p_i\right) \\
& + \sum_{i=1}^{m} y_i \log\left(\frac{\lambda_i \phi_i p_i}{\sum_{k=1}^{m} \lambda_k p_k}\right) + \left(n_1 - \sum_{k=1}^{m} y_k\right) \log\left(1 - \frac{\sum_{k=1}^{m} \lambda_k \phi_k p_k}{\sum_{k=1}^{m} \lambda_k p_k}\right) \\
& + \sum_{i=1}^{m} z_i \log\left[\frac{(1-\lambda_i)\phi_i p_i}{1 - \sum_{k=1}^{m} \lambda_k p_k}\right] + \left(n_2 - \sum_{k=1}^{m} z_k\right) \log\left[1 - \frac{\sum_{k=1}^{m} (1-\lambda_k)\phi_k p_k}{1 - \sum_{k=1}^{m} \lambda_k p_k}\right].
\end{aligned}
\qquad (2)
$$

Score Statistic

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

$$\frac{\partial l}{\partial p_i} = \frac{\lambda_i(x_1 - n_1)}{\sum_{k=1}^{m}\lambda_k p_k} - \frac{\lambda_i(N - x_1 - n_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{y_i + z_i}{p_i} + \frac{(n_1 - \sum_{k=1}^{m} y_k)(1-\phi_k)\lambda_k}{\sum_{k=1}^{m}(1-\phi_k)\lambda_k p_k} \tag{3}$$

$$- \frac{(n_2 - \sum_{k=1}^{m} z_k)\left[(1-\phi_i)\lambda_i + \phi_i\right]}{1-\sum_{k=1}^{m}\left[(1-\phi_k)\lambda_k + \phi_k\right]p_k}.$$

The score statistic is the gradient of the log likelihood, whose components consist of the partial derivatives in equation (3). The MLEs of p_i (which are the zeroes of the score statistic) are found numerically using Fisher's Scoring Method, which will be described in detail later in this documentation. Notice that there are special cases that must be considered where the log-likelihood function in equation (3) is undefined: namely, when the PBT fractions (ϕ_i) are all one, or when the VM fractions (λ_i) are all zero.

Special case (all visually marked releases PBT).—In the special case where all the visually marked releases are PBT, the log-likelihood equation becomes

$$\frac{\partial l}{\partial p_i} = \frac{\lambda_i(x_1 - n_1)}{\sum_{k=1}^{m}\lambda_k p_k} - \frac{\lambda_i(N - x_1 - n_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{y_i + z_i}{p_i} - \frac{(n_2 - \sum_{k=1}^{m} z_k)\phi_i}{1-\sum_{k=1}^{m}\phi_k p_k}. \quad [(1\text{-}\phi_i)\lambda_i = 0 \text{ for } i = 1,\ldots,n] \tag{4}$$

Special case (no releases VM).—In the special case where no releases are VM, then the log-likelihood equation becomes

$$\frac{\partial l}{\partial p_i} = \frac{z_i}{p_i} - \frac{(n_2 - \sum_{k=1}^{m} z_k)\phi_i}{1 - \sum_{k=1}^{m} \phi_k p_k} \qquad (\lambda_i = 0 \text{ for } i = 1, \ldots, n) \tag{5}$$

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 p_i. 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

$$H_{ii} = -\frac{\lambda_i^2 (x_1 - n_1)}{\left(\sum_{k=1}^{m} \lambda_k p_k\right)^2} - \frac{\lambda_i^2 (N - x_1 - n_2)}{\left(1 - \sum_{k=1}^{m} \lambda_k p_k\right)^2} - \frac{(y_i + z_i)}{p_i^2} - \frac{\left(n_1 - \sum_{k=1}^{m} y_k\right)(1-\phi_i)^2 \lambda_i^2}{\left[\sum_{k=1}^{m} (1-\phi_k)\lambda_k p_k\right]^2}$$
$$- \frac{\left(n_2 - \sum_{k=1}^{m} z_k\right)\left((1-\phi_i)\lambda_i + \phi_i\right)^2}{\left\{1 - \sum_{k=1}^{m} \left[(1-\phi_k)\lambda_k + \phi_k\right] p_k\right\}^2}, \tag{6}$$

and its off-diagonal elements are given by

$$H_{ij} = H_{ji} = -\frac{\lambda_i \lambda_j (x_1 - n_1)}{\left(\sum_{k=1}^{m} \lambda_k p_k\right)^2} - \frac{\lambda_i \lambda_j (N - x_1 - n_2)}{\left[1 - \sum_{k=1}^{m} \lambda_k p_k\right]^2} - \frac{\left(n_1 - \sum_{k=1}^{m} y_k\right)(1-\phi_i)(1-\phi_j)\lambda_i \lambda_j}{\left[\sum_{k=1}^{m} (1-\phi_k)\lambda_k p_k\right]^2} \tag{7}$$

$$-\frac{\left(n_2-\sum_{k=1}^{m} z_k\right)\left[(1-\phi_i)\lambda_i+\phi_i\right]\left[(1-\phi_j)\lambda_j+\phi_j\right]}{\left\{1-\sum_{k=1}^{m}\left[(1-\phi_k)\lambda_k+\phi_k\right]p_k\right\}^2},$$

where $i \neq j$; $i = 1,2,\ldots m$; and $j = 1,2,\ldots m$. I assume that not all of the p_i are zero, not all of the VM fractions are zero, and not all of the PBT fractions are equal to one; otherwise equations (6) and (7) 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 x_1, x_2, y_i, and z_i are needed. These are given by

$$E(x_1) = N\sum_{i=1}^{m}\lambda_i p_i\ , \tag{8}$$

$$E(x_2) = N\left(1-\sum_{i=1}^{m}\lambda_i p_i\right), \tag{9}$$

$$E(y_i) = \frac{n_1\lambda_i\phi_i p_i}{\sum_{k=1}^{m}\lambda_k p_k}, \tag{10}$$

and

$$E(z_i) = \frac{n_2(1-\lambda_i)\phi_i p_i}{1-\sum_{k=1}^{m}\lambda_k p_k}. \tag{11}$$

Taking the expected value of the Hessian by using the expected values of x_1, x_2, y_i, and z_i yields the following Fisher Information Matrix ($\mathbf{I}$) with diagonal entries:

$$I_{ii} = -E(H_{ii}) = \frac{N\lambda_i^2(1-\theta_1)}{\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\lambda_i^2(1-\theta_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\phi_i\left[\theta_1\lambda_i + \theta_2(1-\lambda_i)\right]}{p_i} + \frac{N\theta_1(1-\phi_i)^2\lambda_i^2}{\sum_{k=1}^{m}(1-\phi_k)\lambda_k p_k}$$
$$+ \frac{N\theta_2\left[(1-\phi_i)\lambda_i + \phi_i\right]^2}{1-\sum_{k=1}^{m}\left[(1-\phi_k)\lambda_k + \phi_k\right]p_k}, \tag{12}$$

and off-diagonal entries

$$I_{ij} = I_{ji} = -E(H_{ij}) = \frac{N\lambda_i\lambda_j(1-\theta_1)}{\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\lambda_i\lambda_j(1-\theta_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\theta_1(1-\phi_i)\lambda_i(1-\phi_j)\lambda_j}{\sum_{k=1}^{m}(1-\phi_k)\lambda_k p_k}$$
$$+ \frac{N\theta_2\left[(1-\phi_i)\lambda_i + \phi_i\right]\left[(1-\phi_j)\lambda_j + \phi_j\right]}{1-\sum_{k=1}^{m}\left[(1-\phi_k)\lambda_k + \phi_k\right]p_k}, \tag{13}$$

where $\theta_1 = n_1(N\sum_{i=1}^{m}\lambda_i p_i)^{-1}$ and $\theta_2 = n_2\left[N\left(1-\sum_{i=1}^{m}\lambda_i p_i\right)\right]^{-1}$. I assume that not all of the p_i are zero, not all of the VM fractions are zero, and not all of the PBT fractions are equal to one; otherwise equations (11) and (12) would be undefined.

Special case (all visually marked releases PBT).—In the special case where all the visually marked releases are PBT, the information matrix is given, for diagonal entries, by

$$I_{ii} = \frac{N\lambda_i^2(1-\theta_1)}{\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\lambda_i^2(1-\theta_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\left[\theta_1\lambda_i + \theta_2\left(1-\lambda_i\right)\right]}{p_i} \qquad [(1\text{-}\phi_i)\lambda_i = 0 \text{ for } i = 1,\ldots,n] \tag{14}$$

$$+\frac{N\theta_2\phi_i^2}{1-\sum_{k=1}^{m}\phi_k p_k},$$

and, for off-diagonal entries, by

$$I_{ij} = I_{ji} = \frac{N\lambda_i\lambda_j(1-\theta_1)}{\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\lambda_i\lambda_j(1-\theta_2)}{1-\sum_{k=1}^{m}\lambda_k p_k} + \frac{N\theta_2\phi_i\phi_j}{1-\sum_{k=1}^{m}\phi_k p_k}. \qquad [(1\text{-}\phi_i)\lambda_i = 0 \text{ for } i = 1,\ldots,n] \tag{15}$$

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

$$I_{ii} = \frac{N\phi_i\theta_2}{p_i} + \frac{N\theta_2\phi_i^2}{1-\sum_{k=1}^{m}\phi_k p_k}, \qquad (\lambda_i = 0 \text{ for } i = 1,\ldots,n) \tag{16}$$

And, for off-diagonal entries, by

$$I_{ij} = I_{ji} = \frac{N\theta_2\phi_i\phi_j}{1-\sum_{k=1}^{m}\phi_k p_k}. \qquad (\lambda_i = 0 \text{ for } i = 1,\ldots,n) \tag{17}$$

Precision

Given that the Fisher Information Matrix is calculated and that 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

$$\hat{\mathbf{p}} = [\hat{p}_1 \quad \hat{p}_2 \quad \ldots \quad \hat{p}_m]' \tag{18}$$

is given by

$$\text{var}(\hat{\mathbf{p}}) = \mathbf{I}^{-1}, \tag{19}$$

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, $\hat{p} = \sum_{i=1}^{m} \hat{p}_i$, I use the formula

$$\text{var}(\hat{p}) = \text{var}(\mathbf{e}'\hat{\mathbf{p}}) = \mathbf{e}'\,\text{var}(\hat{\mathbf{p}})\mathbf{e}, \tag{20}$$

where $\mathbf{e}$ is an *m*-vector of 1s. The standard error of $\hat{p}$ is then

$$SE(\hat{p}) = \sqrt{\text{var}(\hat{p})}, \tag{21}$$

and the coefficient of variation of $\hat{p}$ is

$$CV(\hat{p}) = \frac{SE(\hat{p})}{\hat{p}}. \tag{22}$$

The minimum SE is obtained when both n_1 and n_2 are as large as possible. This occurs when $n = N$, $n_1 = E(x_1)$, and $n_2 = E(x_2)$. To find the minimal standard error, $SE_{\min}(\hat{p})$, substitute these values in the Fisher Information matrix equations, then use equations (19)-(21). The $CV_{\min}(\hat{p})$ may be found by employing equations (19)-(22). This minimum SE is useful when evaluating the precision of alternative choices for subsample sizes n_1 and n_2.

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

$$\hat{\mathbf{p}}_{l+1} = \hat{\mathbf{p}}_l + \mathbf{I}^{-1}(\hat{\mathbf{p}}_l)\nabla l(\hat{\mathbf{p}}_l), \tag{23}$$

where $\hat{\mathbf{p}}_{l+1}$ is the maximum likelihood estimator of $\mathbf{p}$ on iteration l+1, $\hat{\mathbf{p}}_l$ is the maximum likelihood estimator of $\mathbf{p}$ on iteration l, $\mathbf{I}^{-1}(\hat{\mathbf{p}}_l)$ is the inverse of the Fisher Information Matrix evaluated at $\hat{\mathbf{p}}_l$, and $\nabla l(\hat{\mathbf{p}}_l)$ is the gradient or score statistic, which represents the vector of first partial derivatives of the likelihood function evaluated at $\hat{\mathbf{p}}_l$. The algorithm proceeds by making a first initial guess at the MLE, then iterating equation (23) until the estimate changes by no more than a given error tolerance (e.g., 10^{-5}).

Estimated proportions of hatchery-origin spawners equal to zero.—A special case emerges when conducting Monte Carlo simulations with the Fisher's Scoring Method. If a hatchery has a positive expected number of tag recoveries, a marking fraction of zero, and zero PBT recoveries then the estimate of the proportion of hatchery-origin spawners is zero for that hatchery (i.e., $\hat{p}_k^* = 0$). In this case, the Fisher Information Matrix (evaluated at $\hat{p}_k^* = 0$) is not defined due to division by zero. To remedy this difficulty, the hatchery contributing an estimated zero spawners is removed from the estimation procedure, effectively reducing the dimension of the estimation problem and eliminating the division by zero.

Special singular cases in the estimation problem

There are important special cases to consider in the estimation of p when the Fisher Information is singular and therefore Fisher's Scoring Method cannot be applied as described in the previous section. In the first case I present, the dimension of the estimation problem may be reduced allowing estimation of p to proceed. In the second case, the estimation problem cannot be solved.

Singular case for the Fisher Information Matrix where p is estimable.—In the special case where two or more hatcheries have expected tag recoveries of zero, the Fisher Information matrix is singular and it is impossible to estimate the individual contributions of these individual hatcheries. However, if each of these hatcheries uses the same marking rate, p is still estimable; an alternative, lower dimension Fisher Information matrix may be formulated. To estimate p, combine the hatcheries with expected tag recoveries of zero, i.e., $E(y_i + z_i) = 0$, into a single hatchery for the purposes of estimation. If there are m_0 hatcheries that have expected tag recoveries of zero (and constant marking fractions), then the revised "number of hatcheries" used for the purposes of analysis is $m_{new} = m - m_0 + 1$. Assume that the hatchery indices are arranged so that the first m_0 hatcheries are those with expected numbers of tag recoveries equal to zero. Then the VM fractions for the purpose of analysis are given by $\left[\lambda_1, \lambda_{m_0+1}, \ldots, \lambda_m\right]'$, where λ_1 is the marking fraction of the first m_0 hatcheries. The PBT fractions for the purpose of analysis are given by $\left[0,\phi_{m_0+1},\ldots,\phi_m\right]'$. I chose zero for the PBT fraction of the hatcheries with expected tag recoveries of zero, but any PBT fractions would suffice because they do not enter into the calculation of $\hat{p}$. Note that when a single hatchery alone has an expected number of tag recoveries of zero and a positive visual marking fraction, then the original Fisher Information Matrix is not singular, and estimation of all the hatchery-specific proportions of hatchery-origin spawners may proceed without reducing the dimension of the estimation problem.

Note that when marking fractions at all source hatcheries are identical and there are no expected PBT recoveries from any of the hatcheries, this fits the preceding special case. This is the case of the constant marking fraction, and the statistical properties of the MLE of *p* are well-known Hinrichsen et al. (2012).

Singular cases for the Fisher Information Matrix where p is not estimable.—There are important singular cases where it is impossible to estimate the fraction of hatchery-origin spawners. This occurs when the expected number of PBT recoveries is zero at two or more hatcheries and VM fractions differ at these hatcheries or are zero. Estimation is also impossible when a single hatchery has an expected number of PBT recoveries of zero and a visual marking fraction of zero. In these cases, the estimation problem cannot be solved by reducing its dimension: a redesign of the study is required. One possible redesign that would ensure that *p* is estimable would be to use the same visual marking fractions at hatcheries with expected PBT recoveries of zero. Another possible redesign would be to insist that the expected number of PBT recoveries for all hatcheries is not zero.

Boostrap Estimates of Precision and Bias

As an alternative approach to estimating the precision of $\hat{p}$, Bootstrap simulation can be used. Additionally, this approach yields an estimate of accuracy (bias). Boostrap estimates of precision and accuracy do not rely on asymptotic theory as in the previous sections. That is, bootstrap estimates of precision and bias will work with small sample sizes (*N*) and small subsample sizes, n_1 and n_2. The bootstrap method proceeds by drawing NBOOT random samples from the joint distribution of $x_1, x_2, y_1,\ldots, y_m, z_1,\ldots z_m$ given by equation (1), then calculating the MLE of the proportion of hatchery-origin spawners for each bootstrap sample. This yields NBOOT replications of the MLE, denoted by $\hat{p}_1^*, \hat{p}_2^*,\ldots, \hat{p}_{NSIM}^*$. The bootstrap 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 boostrap replications:

$$SE^*\left(\hat{p}\right)=\sqrt{\sum_{k=1}^{NBOOT}\frac{\left(\hat{p}_k^*-\bar{\hat{p}}^*\right)^2}{NBOOT-1}}\ , \tag{26}$$

where $\bar{\hat{p}}^*$ is the sample mean of the boostrap replications. The coefficient of variation of $\hat{p}$ is

$$CV^*(\hat{p}) = \frac{SE^*(\hat{p})}{\hat{p}} \quad . \tag{27}$$

The relative bias is calculated as

$$bias^*(\hat{p}) = \frac{\hat{p} - \bar{\hat{p}}^*}{\hat{p}} \quad . \tag{28}$$

APPLICATION

Hanford Reach.— For the purposes of illustration, I use data in Hinrichsen et al. (2012) to estimate the proportion of hatchery-origin spawners in the Hanford Reach spawning areas, located in the mainstem Columbia River (Figure 2). Although these are coded-wire tag (CWT) data, we can treat the tagging fractions as “PBT” and use the methods outlined in this report to estimate the fraction of hatchery-origin spawners. The Hinrichsen et al. (2012) method was a generalized least squares estimation (GLSE) approach, while the approach in this report is a maximum likelihood estimation (MLE) approach. Therefore treating the 2010 Hanford Reach data with the MLE approach allows us to see how the MLE and GLSE estimates compare. The data for the Hanford Reach are summarized in Table 2.

Annual carcass surveys are used to estimate the proportion of hatchery-origin spawners in this reach. Carcasses are collected primarily by boat but foot surveys along the shorelines are also conducted. Carcasses with a visible mark (adipose fin clip) are tested with a hand-held CWT detection wand to determine whether a CWT is present. Carcasses without are visible mark are considered part of the total numbers sampled but are not checked for CWTs . If a carcass contains a CWT, its snout (containing the CWT) is removed, frozen, and sent to the Washington Department of Fish and Wildlife coded-wire tag lab in Olympia, WA, to be examined. At the lab, the CWT is extracted and the CWT code is read to identify hatchery of origin. CWT codes are traced to the hatchery and release group of origin by biologists. This information is accessible via the Pacific States Marine Fisheries Commission’s Regional Mark Processing Center at www.rmpc.org.

In 2010, the hatcheries contributing spawners with CWTs in the Hanford Reach were Little White Salmon brood year (BY) 2005, Lyons Ferry (Snake River) BY 2006, Priest Rapids BY 2005 and BY 2007, Ringold Springs BY 2006 and BY 2007, and Umatilla BY 2007 (Table 2). For convenience, each hatchery-BY pair was treated as a separate hatchery in the equations describing the estimators and their variances. Thus there were a total of seven input hatchery-BY pairs (n=7). Of the 9,791 the carcasses sampled in 2010, 23 had a visible mark and a CWT inserted at a hatchery, and 308 had a visible mark, but no CWT.

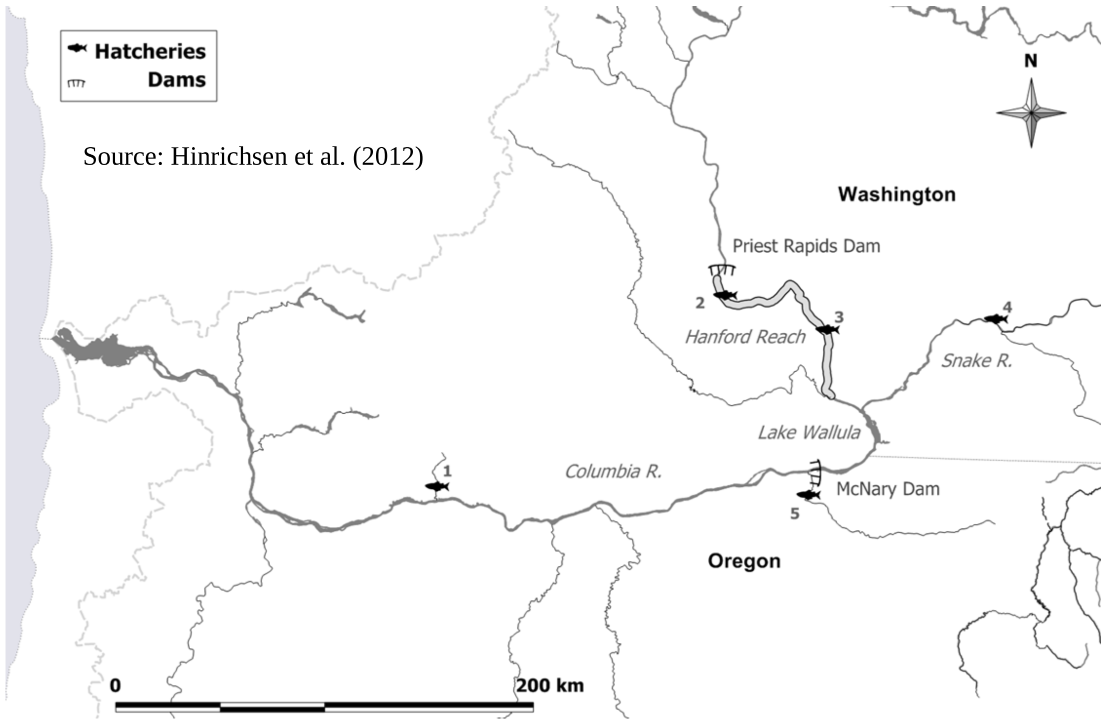

Figure 2. The Hanford Reach is a 90-km stretch of the Columbia River extending from the upper end of McNary Dam reservoir to Priest Rapids Dam (river km 639) where 2010 carcass surveys recovered numerous CWTs. Fish symbols are hatchery facilities that contributed CWT fall Chinook salmon spawners to the Hanford Reach, and numbers refer to the hatcheries in Table 2.

Table 2.—Visible marking and coded-wire tagging at source hatcheries that provide spawner inputs to Hanford Reach spawning grounds. The total number of spawning ground carcasses sampled in 2010 was N=9,791. Only fish with a mark were checked for a CWT, so n1=n =331. PBT is given in quotes as a reminder that the data are really CWT data. Numbers (#) refer to hatchery locations in Figure 2.

#	Hatchery	Brood year	VM fraction, λ	"PBT" fraction, ϕ	y	z
1	Little White Salmon NFH	2005	1.00	0.25	1	0
2	Priest Rapids H	2005	0.27	0.11	3	0
		2007	0.04	1.00	7	0
3	Ringold Springs H	2006	0.07	1.00	2	0
		2007	0.79	0.09	7	0
4	Lyons Ferry H	2006	0.51	0.99	1	0
5	Umatilla H	2007	1.00	1.00	2	0

The R-code used to produce the 2010 Hanford reach maximum likelihood estimate of p is given in Table 3. The resulting MLE estimate is $\hat{p}_{MLE} = 0.0766$ ($SE = 0.0091$). Hinrichsen et al. (2012) calculate a GLSE estimate of $\hat{p}_{GLSE} = 0.0766$ ($SE = 0.0090$). Thus the two estimates are identical to the fourth decimal place.

Table 3. R-code used to produce the MLE of *p* for Hanford Reach spawners in 2010.

```
hanford<-function(BOOT=FALSE)
{
 hatchnames<-c("LTL WHITE SALMON NFH 2005",
"PRIEST RAPIDS HATCHERY 2005",
"PRIEST RAPIDS HATCHERY 2007",
"RINGOLD SPRINGS HATCHERY 2006",
"RINGOLD SPRINGS HATCHERY 2007",
"LYONS FERRY HATCHERY 2006",
"UMATILLA HATCHERY 2007")
 ppbt<-c(0.248669107,0.109102059,0.996002576,1,0.090513147,0.992826116,1)
 lambda<-c(1,0.265849608,0.044715769,0.06545306,0.791664689,0.507376537,1)
 z<-c(0,0,0,0,0,0,0)
 y<-c(1,3,7,2,7,1,2)
 n1<-331
 n<-331
 x1<-n1
 Nsamp<-9791
 res<-phos.pbte.main(Nsamp=Nsamp,x1=x1,n=n,n1=n1,y=y,z=z,lambda=lambda,ppbt=ppbt,
BOOT=BOOT,NBOOT=1000)
 return(res)
}
```

ACKNOWLEDGEMENTS

The author gratefully acknowledges the reviews of Mike Ackerman, Matt Campbell, Maureen Hess, Brian Maschhoff, Shawn Narum, Rishi Sharma, Barbara Shields, Craig Steele, and Bill Young. This work was supported by Bonneville Power Administration. Brian Maschhoff implemented this statistical analysis as a web-based tool at www.onefishtwofish.net. Amber Parsons reviewed this documentation and tested the web-based tool HatchPBTE. 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 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.

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://collaboration.idfg.idaho.gov/FisheriesTechnicalReports/Res11-111Steele2010%20Parentage%20Based%20Tagging%20Snake%20River%20Steelhead%20Salmon.pdf

Steele, C., E.C. Anderson, M.W. Ackerman, M.A. Hess, N.R. Campbell, S.R. Narum, and M.R. Campbell. 2013. A validation of parentage-based tagging using hatchery steelhead in the Snake River basin. Canadian Journal of Fisheries and Aquatic Sciences 70: 10.1139/cjfas-2012-0451.

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	Estimated proportion of hatchery-origin spawners
phosi[i]	p_i	Estimated proportion of hatchery-origin spawners from hatchery *i*.
ppbt[i]	ϕ_i	PBT fraction for hatchery *i*.
lambda[i]	λ_i	VM fraction for hatchery *i*.
Nsamp	N	Total sample size
x1	x_1	Number of VM spawners in total sample
x2	x_2	Number of unmarked spawners in total sample
y[i]	y_i	Number of subsampled VM spawners testing positive for PBT from hatchery *i*.
z[i]	z_i	Number of the subsampled unmarked spawners testing positive for PBT from hatchery *i*.
n	n	Subsample size
n1	n_1	Number of VM spawners tested for PBT
n2	n_2	Number of unmarked spawners tested for PBT
Ex1	$E(x_1)$	Expected value of x_1
Ex2	$E(x_2)$	Expected value of x_2
I	$\mathbf{I}$	Fisher Information Matrix
SE.phosi[i]	$SE(\hat{p}_i)$	Standard error of the estimate of the proportion of hatchery-origin spawners from hatchery *i*
SE.phos	$SE(\hat{p})$	Standard error of the estimate of the proportion of hatchery-origin spawners
CV.phos	$CV(\hat{p})$	Coefficient of variation of the estimate of the proportion of hatchery-origin spawners
BIAS.phos	$bias^*(\hat{p})$	Bootstrap relative bias which is bias divided by the estimated value of *p*.

APPENDIX B. R-CODE

Table B.1. R-code

```
#R-code to estimate phos using maximum likelihood theory. Statistical properties of the estimator
#are estimated using maximum likelihood theory and bootstrapping. This code treats the general case
#of inputs from several hatcheries with potentially different visible marking fractions and different
#parentage-based tag fractions.
#FILE: pbte-web-8-20-2013.s
#AUTHOR: Richard A. Hinrichsen, 20 August 2013

#Variables and parameters used in the analysis
#inputs
#Nsamp = total sample size
#x1 = number of marked spawners in sample of size Nsamp
#n  = total subsample sizes (number tested for PBT)
#n1 = number of marked spawners tested for PBT
#y = number of marked spawners tested that were PBT (hatchery-specific)
#z = number of unmarked spawners tested that were PBT (hatchery-specific)
#lambda = marking rate (lambda)  (hatchery-specific)
#ppbt=fraction of marked fish that are also parentage-based tagged (hatchery-specific)
#BOOT = FALSE for theoretical results, TRUE for Boostrap results
#NBOOT = number of Bootstrap simulations (needed if BOOT=TRUE)
#
#
#Select intermediate variables
#nhatch=number of hatcheries supplying spawners in the wild
#I = Observed Fisher Information Matrix
#x2 = number of unmarked spawners in sample of size Nsamp
#n2 = number of unmarked spawners tested for PBT

#Results
#phosi= maximum likelihood estimates of the proportions of hatchery-origin spawners
#SE.phosi= standard errors of the estimates of the proportions of hatchery-origin spawners
#phos = maximum likelihood estimate of the proportion of hatchery-origin spawners
#SE.phos = standard error (SE) of the estimate of phos
#CV.phos = Coefficient of variation of the estimate of phos
#BIAS.phos = relative bias of the estimate of phos (NA if BOOT=FALSE)

#top level function
phos.pbte.main<-function(Nsamp=1000,x1=50,n=200,n1=50,
                  y=c(16,15,8,8),
                  z=c(0,1,14,14),
                  lambda=c(1,.95,.5,.5),
                  ppbt=c(.95,.95,.95,.95),
                  BOOT=FALSE,NBOOT=1000){
 x2<-Nsamp-x1
 n2<-n-n1
 check.input(x1=x1,x2=x2,n1=n1,n2=n2,y=y,z=z,lambda=lambda,ppbt=ppbt,BOOT=BOOT,NBOOT=NBOOT)
 if(!BOOT){res<-
phos.pbte.estimates(x1=x1,x2=x2,n1=n1,n2=n2,y=y,z=z,lambda=lambda,ppbt=ppbt,suppress=FALSE)}
 if(BOOT){res<-
phos.pbte.estimates2(NBOOT=NBOOT,x1=x1,x2=x2,n1=n1,n2=n2,y=y,z=z,lambda=lambda,ppbt=ppbt)}
 return(res)
}

#check input to phos.pbte.main,phos.pbte.estimates,phos.pbte.estimates2
check.input<-function(x1,x2,n1,n2,y,z,lambda,ppbt,BOOT,NBOOT)
{
 Nsamp<-x1+x2
 if(!is.logical(BOOT))stop("BOOT must be TRUE or FALSE")
 if(floor(Nsamp)!=Nsamp){stop("Nsamp must be a positive integer")}
 if(Nsamp<=0){stop("Nsamp must be a positive integer")}
 if(BOOT){
 if(floor(NBOOT)!=NBOOT){stop("NBOOT must be a positive integer")}
 if(NBOOT<=0){stop("NBOOT must be a positive integer")}
 }
#check dimension of inputs
 k2<-length(lambda);k3<-length(ppbt);k4<-length(y);k5<-length(z)
 mytest<-abs(k2-k3)+abs(k2-k4)+abs(k2-k5)
 if(mytest>0) stop("dimensions of lambda, ppbt, y, and z must match")
#check that subsample size is less than sample size
 n<-n1+n2
 if(n>Nsamp)stop(paste("n=n1+n2 must be less than or equal to Nsamp=x1+x2=",Nsamp))
#check sample and subsample subsample
 if(x1<0)stop("x1 must be nonnegative")
 if(x2<0)stop("x2=Nsamp-x1 must be nonnegative")
 if(n1<0)stop("n1 must be nonnegative")
 if(n2<0)stop("n2=n-n1 must be nonnegative")
 if(n1>x1)stop(paste("n1 must not exceed the number of VM spawners in sample=",x1))
 if(n2>x2)stop(paste("n2 must not exceed the number of ~VM spawners in the sample=",x2=Nsamp-x1))
#check that all ppbts are between zero and 1.0
 if(sum(ppbt<0))stop("ppbts must all be greater than or equal to zero")
 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 for consistency between y,x1,z,and x2,and lambda
 if(sum(y<0))stop("ys must all be greater than or equal to zero")
 if(sum(z<0))stop("zs must all be greater than or equal to zero")
 if(sum(y)>n1)stop("ys must not sum to greater than n1")
 if(sum(z)>n2)stop("zs must not sum to greater than n2=n-n1")
 iii<-(y>0)&(lambda==0)
 if(sum(iii))stop("y cannot be > 0 if no releases are visibly marked")
 iii<-(z>0)&(lambda==1)
 if(sum(iii))stop("z cannot be > 0 if all releases are visibly marked")
 if((sum(lambda)==0)&(x1>0))stop("x1 cannot be > 0 if no hatchery releases are visibly marked")
 if((sum(lambda*(1-ppbt))==0)&(sum(y)!=n1)){
  stop("sum(y) should equal n1 since all ppbt=1 for all hatcheries with lambda>0")}
#When a group of hatcheries has zero expected PBT recoveries, they must use constant VM fractions
 iii<-n1*lambda*ppbt+n2*(1-lambda)*ppbt==0
 onelambda2<-sum(mean(lambda[iii])==lambda[iii])==sum(iii)
 onelambda2<-onelambda2&(mean(lambda[iii])>0)
 if((!onelambda2)&sum(iii))stop("Expected tag recoveries must not be zero when marking fractions differ")
 return(NULL)
}

#Maximum Likelihood Theory results
```

```
phos.pbte.estimates<-function(x1,x2,n1,n2,y,z,lambda,ppbt,suppress){
 nhatch<-length(lambda)
 Nsamp<-x1+x2

#An important case for combining cells occurs when the E(y+z)=0
#in this case, constant lambdas over these cells saves the estimation.
#This also takes care of the case of a single lambda for all hatcheries
iii<-n1*lambda*ppbt+n2*(1-lambda)*ppbt==0
onelambda2<-sum(lambda[iii]==mean(lambda[iii]))==sum(iii)
if((sum(iii)>1)&onelambda2){
 if(!suppress)warning("collapsing cells with expected tag recoveries of zero into single cell since lambda is
constant")
 lambda1.new<-mean(lambda[iii])
 ppbt1.new<-0.0
 lambda.new<-c(lambda1.new,lambda[!iii])
 ppbt.new<-c(ppbt1.new,ppbt[!iii])
 y.new<-c(0,y[!iii])
 z.new<-c(0,z[!iii])
 nhatch.new<-length(lambda.new)
 res<-
phos.pbte.estimates(x1=x1,x2=x2,n1=n1,n2=n2,y=y.new,z=z.new,lambda=lambda.new,ppbt=ppbt.new,suppress=su
ppress)
 phosi<-rep(NA,nhatch)
 SE.phosi<-rep(NA,nhatch)
 phosi[!iii]<-res$phosi[2:nhatch.new]
 SE.phosi[!iii]<-res$SE.phosi[2:nhatch.new]

 myres<-list(BOOT=FALSE,
                  NBOOT=NA,
                  Nsamp=Nsamp,
                  x1=x1,
                  x2=x2,
                  n=n1+n2,
                  n1=n1,
                  n2=n2,
                  y=y,
                  z=z,
                  lambda=lambda,
                  ppbt=ppbt,
                  phosi=phosi,
                  SE.phosi=SE.phosi,
                  phos=res$phos,
                  SE.phos=res$SE.phos,
                  CV.phos=res$CV.phos,
                  BIAS.phos=NA)
 return(myres)
}

#get initial estimate of phosi
phosi.init<-init(x1,x2,n1,n2,y,z,lambda,ppbt,suppress=suppress)

myres<-get.estimates2(phosi.init,x1,x2,n1,n2,y,z,lambda,ppbt,suppress=suppress)
phos<-myres$phos
phos.var<-myres$phos.var
```

```
phosi<-myres$phosi
SE.phosi<-sqrt(myres$phosi.var)
SE.phos<-sqrt(phos.var)
CV.phos<-SE.phos/phos
SE.phos<-as.numeric(SE.phos)
CV.phos<-as.numeric(CV.phos)

myres<-list(BOOT=FALSE,
            NBOOT=NA,
            Nsamp=Nsamp,
            x1=x1,
            x2=x2,
            n=n1+n2,
            n1=n1,
            n2=n2,
            y=y,
            z=z,
            lambda=lambda,
            ppbt=ppbt,
            phosi=phosi,
            SE.phosi=SE.phosi,
            phos=phos,
            SE.phos=SE.phos,
            CV.phos=CV.phos,
            BIAS.phos=NA)
 return(myres)
}

#Fisher Information Matrix (general case)
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 sum((1-ppbt)*lambda) is zero
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))

 v<-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 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)
}

#Bootstrap estimates of standard error and bias
#consider cases where some of the phosi are missing
phos.pbte.estimates2<-function(NBOOT,x1,x2,n1,n2,y,z,lambda,ppbt){
 nhatch<-length(lambda)
 Nsamp<-x1+x2
#get MLE using theoretical results
 res<-phos.pbte.estimates(x1,x2,n1,n2,y,z,lambda,ppbt,suppress=FALSE)
 phosi<-res$phosi
 phosi.orig<-res$phosi
 phos<-res$phos
 phos.sim<-rep(NA,NBOOT)
 phosi.sim<-matrix(NA,nrow=NBOOT,ncol=nhatch)
 if(!is.na(phos)){
 for(ii in 1:NBOOT){
  iii<-is.na(phosi)
  phosi[iii]<-rep(phos-sum(phosi,na.rm=T),sum(iii))/sum(iii)
  mysim<-pbtsim1(phosi,Nsamp,n1,n2,res$lambda,res$ppbt)
  my.n1<-n1
  my.n2<-n2
  if(mysim$x1<n1)my.n1<-mysim$x1
  if(mysim$x2<n2)my.n2<-mysim$x2
  res<-phos.pbte.estimates(x1=mysim$x1,x2=mysim$x2,n1=my.n1,n2=my.n2,
                y=mysim$y,z=mysim$z,lambda=lambda,ppbt=ppbt,suppress=TRUE)
  phos.sim[ii]<-res$phos
  phosi.sim[ii,]<-res$phosi
 }}

 SE.phosi<-apply(phosi.sim,c(2),var,na.rm=T)
 SE.phosi<-sqrt(SE.phosi)
 SE.phos<-sqrt(var(phos.sim,na.rm=T))
 CV.phos<-SE.phos/phos
 mymean<-mean(phos.sim,na.rm=T)
 BIAS.phos<-(mymean-phos)/phos

 myres<-list(BOOT=TRUE,
                NBOOT=NBOOT,
                Nsamp=Nsamp,
                x1=x1,
                x2=x2,
                n=n1+n2,
                n1=n1,
                n2=n2,
                y=y,
                z=z,
                lambda=lambda,
                ppbt=ppbt,
                phosi=phosi.orig,
                SE.phosi=SE.phosi,
                phos=phos,
                SE.phos=SE.phos,
                CV.phos=CV.phos,
                BIAS.phos=BIAS.phos)
 return(myres)
}

#simulate data when phosi is available
pbtsim1<-function(phosi,Nsamp,n1,n2,lambda,ppbt){
 m<-length(phosi)
#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)
 if(P>0){y<-rmultinom(n=1,size=min(n1,x1),prob=c(py,1-sum(py)))}
 if(P==0){y<-matrix(0,ncol=1,nrow=length(phosi))}
 if(P<1){z<-rmultinom(n=1,size=min(n2,x2),prob=c(pz,1-sum(pz)))}
 if(P==1){z<-matrix(0,ncol=1,nrow=length(phosi))}

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

#Use R.A. Fisher's scoring algorithm to estimate phosi
#phosi represents the intial guess on input
get.estimates2<-function(phosi.init,x1,x2,n1,n2,y,z,lambda,ppbt,suppress){
 Nsamp<-x1+x2
 NTRIAL<-100
 tolx<-1.e-5
 nhatch<-length(phosi.init)
 nhatch.orig<-nhatch
 w<-y+z
 nas<-rep(NA,nhatch)
 phosi<-nas
 phosi.var<-nas
 phos<-NA
 phos.var<-NA

#in a rare case init can return zeroes even though the true estimate is not zero
#which would defeat Fisher's scoring method
jjj<-phosi.init==0
phosi.init[jjj]<-phosi.init[jjj]+.00001
#find zeroes that occur when lambda=0 and n1*lambda*ppbt+n2*(1-lambda)*ppbt>0
jjj<-(n1*lambda*ppbt+n2*(1-lambda)*ppbt>0)&(lambda==0)&(w==0)
if(sum(jjj)>=1){
 y<-y[!jjj]
 z<-z[!jjj]
 ppbt<-ppbt[!jjj]
 lambda<-lambda[!jjj]
 phosi.init<-phosi.init[!jjj]
 nhatch<-length(y)}

#check for special cases
#check to see if all pbts are one (special case)
pbttest<-sum(ppbt==1)==nhatch
#modify slightly
pbttest<-sum((ppbt-1)*lambda)==0
#check to see if all lambdas are zero (special case)
lambdatest<-sum(lambda==0)==nhatch

#get the right Fisher Information function and dlike function
 if(lambdatest){
  my.getI<-getI3
  dlike<-dlike3
 }
 else{

 if(pbttest){
  my.getI<-getI2
  dlike<-dlike2
 }
else{
  my.getI<-getI
  dlike<-dlike1
 }
 }

#use R.A. Fisher's scoring algorithm to find where the partial derivatives of the
#log-likelihood are zero. Use Fisher Information matrix in Newton's method to approx -Hessian
 phosi<-phosi.init
 errf<-0.0
 alpha<-0.9
 for(ii in 1:NTRIAL){
  I<-my.getI(Nsamp,n1,n2,lambda,ppbt,phosi)
  df<-dlike(phosi,x1,x2,n1,n2,y,z,lambda,ppbt)
  size<-prod(dim(I))
  if(size==0){
    if(!suppress)warning("dimension of I is 0 x 0")
    return(list(phosi=nas,phosi.var=nas,phos=NA,phos.var=NA))}
  if(is.na(rcond(I))){
    if(!suppress)warning("condition number of I is NA")
    return(list(phosi=nas,phosi.var=nas,phos=NA,phos.var=NA))}
  if(rcond(I)<1.e-15){
    if(!suppress)warning("computationally singular information matrix")
    return(list(phosi=nas,phosi.var=nas,phos=NA,phos.var=NA))}
  delx<-solve(I)%*%df
  phosi<-phosi+delx*(1-alpha)
  phosi<-abs(phosi)
  errx<-sum(abs(delx))/sum(abs(phosi))
  alpha<-alpha*alpha
  if(errx<=tolx)break
  }
 if(ii==NTRIAL){
  if(!suppress)warning("maximum number of iterations was reached")
  return(list(phosi=nas,phosi.var=nas,phos=NA,phos.var=NA))}
 phos<-sum(phosi)
 e<-rep(1,nhatch)
 myvar<-solve(I)
 phos.var<-t(e)%*%myvar%*%e
 phosi.var<-diag(myvar)
 full.phosi.var<-rep(0,nhatch.orig)
 full.phosi<-rep(0,nhatch.orig)
#recall that jjj represents hatcheries with estimates of phosi=0
#reduction occurs only when sum(jjj>=1)
 if(sum(jjj)>=1){
  full.phosi.var[!jjj]<-phosi.var
  full.phosi[!jjj]<-phosi
 }
 if(sum(jjj)<1){
  full.phosi.var<-phosi.var
  full.phosi<-phosi

 }
 return(list(phosi=full.phosi,phosi.var=full.phosi.var,phos=phos,phos.var=phos.var))
}

#get gradient of the log likelihood function
#phosi is the current best estimate of phosi
#used in most general case
dlike1<-function(phosi,x1,x2,n1,n2,y,z,lambda,ppbt){
#estimate phosi
 Nsamp<-x1+x2
 sum1<-sum(lambda*phosi)
 sum2<-sum((1-ppbt)*lambda*phosi)
 sum3<-sum(phosi*ppbt)
 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-sum2-sum3)
 return(res)
}

#get gradient of the log likelihood function
#phosi is the current best estimate of phosi
#used in the case where sum(lambda*(1-ppbt))===0
dlike2<-function(phosi,x1,x2,n1,n2,y,z,lambda,ppbt){
#estimate phosi
 Nsamp<-x1+x2
 sum1<-sum(lambda*phosi)
 sum3<-sum(phosi*ppbt)
 res<-lambda*(x1-n1)/sum1-lambda*(Nsamp-x1-n2)/(1-sum1)+y/phosi
 res<-res+z/phosi
 res<-res-(n2-sum(z))*ppbt/(1-sum3)
 return(res)
}

#get gradient of the log likelihood function
#phosi is the current best estimate of phosi
#used when lambda=0 at all hatcheries
dlike3<-function(phosi,x1,x2,n1,n2,y,z,lambda,ppbt){
#estimate phosi
 Nsamp<-x1+x2
 sum3<-sum(phosi*ppbt)
 res<-z/phosi
 res<-res-(n2-sum(z))*ppbt/(1-sum3)
 return(res)
}

#get initial estimates of phosi
#by equating x1,y,and z to their expectations
#this yields 2*nhatch +1 equations with nhatch unknowns
#which is, in general, overdetermined.
init<-function(x1,x2,n1,n2,y,z,lambda,ppbt,suppress=FALSE){
Nsamp<-x1+x2
nhatch<-length(lambda)
A1<-lambda
B1<-x1/Nsamp
A2<--n1*diag(lambda*ppbt,ncol=nhatch,nrow=nhatch)
LAMBDAMAT<-matrix(lambda,ncol=nhatch,nrow=nhatch)
LAMBDAMAT<-t(LAMBDAMAT)
YDIAG<-diag(y,nrow=nhatch,ncol=nhatch)
ZDIAG<-diag(z,nrow=nhatch,ncol=nhatch)
A2<-A2+YDIAG%*%LAMBDAMAT
B2<-rep(0,nhatch)
A3<-n2*diag((1-lambda)*ppbt,nrow=nhatch,ncol=nhatch)
A3<-A3+ZDIAG%*%LAMBDAMAT
B3<-z
A<-rbind(A1,A2,A3)
B<-c(B1,B2,B3)
if(rcond(t(A)%*%A)<1.e-15){
    if(!suppress)warning("matrix t(A)%A in init() is computationally singular")}
phosi<-solve(t(A)%*%A)%*%t(A)%*%B
return(abs(phosi))
}
```