Part 9. Comprehensive Computer Software

The computation of estimates and test statistics using the methods presented in this monograph often is difficult or, at best, tedious. Details on a computer software package are presented herein. Many options are discussed, and several examples are given. The program is easy to use on microcomputers, especially the interactive version.

9.1. Program Capabilities and Options

Program RELEASE is a general program that computes estimates of treatment effect ($\hat{S}$), survival ($\hat{\phi}_{ni}$), and capture ($\hat{p}_{ni}$) probabilities from release-recapture data for one or more groups under one of the four sampling protocols. The program reads a TITLE statement followed by the input of data either as a CH matrix (CHMATRIX) or a reduced m-array (LMREAD). These last two procedures (PROCs) allow a variety of options to be specified. Multiple analyses can be made, and PROC STOP terminates the execution. A semicolon is used as a delimiter at the end of each input statement. These four procedures are given in Table 9.1; more details appear in Table 9.2. The output of RELEASE provides parameter estimates, estimated standard errors, 95% CIs, and the tests given in Tables 2.1 to 2.4 for those appropriate models.

Other procedures (Table 9.1) allow data to be simulated and analyzed (SIMULATE), or contingency table data to be input and analyzed (CHISQ). PROC SIMULATE also has an option for exploring theoretical biases, precision, and test power. Finally, PROC SURVIV generates a data input file for analysis by program SURVIV (White 1983). Program SURVIV allows total flexibility in model building and analysis.

To improve the user's efficiency in using RELEASE, an interactive interface has been provided. The user is requested to select input from menus; this input is then translated into PROC statements and executed to produce the desired output. In addition, a log of the input is produced on an output file for later execution in batch mode. The interactive interface allows users unfamiliar with the program's input syntax to produce useful results.

Program RELEASE is available for IBM-PC microcomputers or equivalent compatibles running the DOS operating system. Microcomputers in this class with at least 512K of random access memory, a hard disk, and an 8087 math coprocessor (80287 for AT) can run RELEASE without alteration. Microcomputers lacking a hard disk can only run RELEASE with a smaller number of capture occasions and treatment groups. The program is written in FORTRAN 77 and is compiled with the Ryan-McFarland FORTRAN compiler.

Program RELEASE can be obtained without charge by writing to

Dr. Gary C. White
Department of Fishery and Wildlife Biology
Colorado State University
Fort Collins, Colorado 80523 USA.

Requests should include three double-density, double-sided 5.25-inch floppy disks that have been formatted on an IBM-PC compatible microcomputer or, preferably, a 1.2M, 5.25-inch floppy disk formatted on an AT-compatible machine. The FORTRAN 77 source code, executable module, and test data sets from this monograph will be returned.

Table 9.1. – Summary of procedures available in program RELEASE. All input can be either upper or lower case. Some procedures have numerous options.

PROC	Function
TITLE	Title to be used for a particular run.
CHMATRIX	Entry of data as a capture history matrix.
LMREAD	Entry of data as reduced m-arrays.
SIMULATE	Allows Monte Carlo data to be generated and analyzed; also allows some theoretical evaluations.
CHISQ	Allows entry and analysis of an $r \times c$ contingency chi-square table.
SURVIV	Generates input routine for program SURVIV.
STOP	Stops execution; end of job.

Table 9.2. – Detailed summary of command syntax for each of the procedures and their modifiers in program RELEASE. Modifiers enclosed in brackets ([ ]) indicate optional input. Program RELEASE allows comments enclosed between /* and */ symbols, and statements are delineated with semicolons; # means "number". Either upper or lower case characters may be used in the input.

```
 /* COMMENTS can be enclosed in these symbols */
PROC TITLE Any title information to be printed at top of each page; titles
      may be changed later in a data file with a second PROC TITLE;
PROC CHMATRIX  OCCASIONS=# of capture-recapture occasions
        GROUPS=# of groups
        [LOTS=# of replicate CH matrices to analyze (default is 1)];
        [UNBIAS (causes bias-corrected estimates of S and φ to be printed)]
        [SUMMARY | NODETAIL (either of these keywords reduces
        the amount of output to just a summary of the goodness
        of fit and model selection  tests and only the
        H(k-1,φ), H(2p), H(1φ), and H0 estimates)]
        [FULLM (causes the full m(ij) array to be printed in the output when k < 8)];

/*   PROC CHMATRIX is the primary procedure to enter the capture-recapture histories for each animal. If
     GROUPS=2, treatment and control animals are entered in that order; if GROUPS>2, the treatment groups
     are entered, in order, followed by the control group. */

/*   OCCASIONS, GROUPS, and LOTS can be in any order on the PROC statement. Within LOTS, GROUPS
     must have the same order on the capture history statements. */
      [LOT [=] 1;]  /* Only required if LOTS > 1 */

          capture_history_1     #_captures_group_1     #_captures_group_2     ...;
          capture_history_2     #_captures_group_1     #_captures_group_2     ...;
                 .                      .                      .              ...;
                 .                      .                      .              ...;
                 .                      .                      .              ...;
          capture_history_n     #_captures_group_1     #_captures_group_2     ...;

/*   capture_history_i is the string of ones and zeros or dots (.) indicating the capture occasions on which the asso-
     ciated numbers of animals were captured for each group. A dot (.) indicates that the user does not know the
     capture history sequence (i.e., unknown capture histories), whereas zero (0) indicates that the animals were
     not captured at this occasion. The number of ones and zeros must match the OCCASIONS= parameter on
     the PROC statement. Likewise, the number of group entries must match the GROUPS= parameter on the
     PROC statement. */
```

Table 9.2. – Continued.

```
/*  #_captures_group_i is the number of animals in group i that had capture_history_i. A particular
    capture_history can be repeated more than once, but the program will produce a warning to notify the user.
    Animals lost on capture should be recorded as negative values (i.e., - 5 to indicate 5 animals were lost on cap-
    ture) on a separate statement from the animals that were released again. Thus the statements, 10100 243 269
    and 10100 - 24 - 27, tell the program that 243 treatment animals (group 1) and 269 control animals (group 2)
    were captured and released on occasion 3, but 24 treatment and 27 control animals were lost on capture and
    not rereleased on occasion 4. None of these animals had been captured on occasion 2, signified by the 0 in
    the second position of the capture history string. */

[LOT [=] 2;] /* Required for LOTS > 1, otherwise not needed. */

    capture_history_1    #_captures_group_1    #_captures_group_2    ...;
    capture_history_2    #_captures_group_1    #_captures_group_2    ...;
            .                    .                    .             ...;
            .                    .                    .             ...;
            .                    .                    .             ...;
    capture_history_n    #_captures_group_1    #_captures_group_2    ...;

/*  The CH matrices for each lot, separated by the LOT subcommand, must continue up to the value specified
    with the LOTS= parameter on the PROC statement. */

[GLABEL(1) = Identifying label for Group 1
        (default is "Treatment Group");]

    .
    .
    .

[GLABEL(#_groups) = Identifying label for last group
        (default is "Control Group");]

/*  The GLABEL statements may occur before the capture histories, if desired. GLABEL statements occur only
    once regardless of the number of lots. */
```

Table 9.2. – Continued.

PROC LMREAD OCCASIONS=# of capture-recapture occasions
GROUPS=# of treatment groups
[LOTS=# of replicate m_{ij} matrices to analyze
(default is 1)]
[UNBIAS (causes bias-corrected estimates of S and
ϕ to be printed)]
[SUMMARY | NODETAIL (either of these keywords reduces
the amount of output to just a summary of the goodness
of fit model selection tests and only the $H_{k-1,\phi}$,
$H_{2\phi}$, $H_{1\phi}$, and H_0 estimates)]
[FCH | UCH | SCHEMEB | SCHEMEA (identifies that the
m_{ij} matrix does not come from a complete capture
history matrix. FCH or UCH require only the first row of
m_{ij}, SCHEMEB requires the first two rows, and
SCHEMEA requires the full m_{ij} matrix.)]

/* PROC LMREAD is a second procedure for entering capture-recapture data. Not as much information for testing of assumptions is available with this data type as with PROC CHMATRIX; consequently, PROC CHMATRIX is preferred for most applications. The m_{ij} matrix (or matrices) for treatment animals must be entered first, followed by the matrix for control animals. */

[LOT [=] 1;] /* Only required for LOTS > 1 */

$$
\begin{array}{llll}
R_{t1} & R_{t2} & \cdots & R_{tk-1} \;; \\
m_{t1,2} & m_{t1,3} & \cdots & m_{t1,k} \;; \\
 & m_{t2,3} & \cdots & m_{t2,k} \;; \\
 & & \cdot & \;; \\
 & & \cdot & \;; \\
 & & \cdot & \;; \\
 & & & m_{tk-1,k} \;;
\end{array}
$$

/* The first row consists of the numbers of animals released for group 1 in LOT 1. The following k - 1 rows are the entries of the m_{ij} matrix for LOT 1, with rows of the matrix separated by semicolons. Each row after the second has one less entry than the previous row, and the last row has only one entry. */

Table 9.2. – Continued.

R_{c1} R_{c2} ... R_{ck-1} ;
$m_{c1,2}$ $m_{c1,3}$ ... $m_{c1,k}$;
$m_{2,3}$ ... $m_{2,k}$;
. ;
. ;
. ;
$m_{ck-1,k}$;

/* The above entries are just the R_i vector and the m_{ij} matrix for group 2, LOT 1. This pattern continues for the number of GROUPS specified on the PROC LMREAD statement. */

[LOT [=] 2;] /* Required for LOTS > 1 */

R_1 R_2 ... R_{k-1} ;
$m_{1,2}$ $m_{1,3}$ ... $m_{1,k}$;
$m_{2,3}$ ... $m_{2,k}$;
. ;
. ;
. ;
$m_{k-1,k}$;

/* The R_i and m_{ij} entries are continued for each group and each lot up to GROUPS= and LOTS= values from the PROC LMREAD statement. */

[GLABEL(1) = Identifying label for Group 1 (default is "Treatment Group");]
.
.
.
[GLABEL(#_groups) = Identifying label for last group (default is "Control Group");]

/* The GLABEL statements may occur before the capture data, if desired. */

PROC SIMULATE [NSIM = # of replications of each simulation (default is 5)]
OCCASIONS=# of capture-recapture occasions
[GROUPS=# of treatment groups, including the control group
(default is 2)]
[DETAIL (print details of each simulation,
with default of only simulation summaries)]
[SUMMARY (print a summary of each simulation with default
of only simulation summaries)]
[REMOVAL|FCH (causes first capture history only data to

Table 9.2. – Continued.

be simulated; recaptures are removed from the population)]
[SCHEMEB (causes scheme B capture history data to be simulated]
[SCHEMEA (causes scheme A capture history data to be simulated]
[UCH (causes unknown capture history data to be simulated)]
[SEED = random number seed (default is 7654321)]
[SFILE = file name to receive estimates of $\hat{S}$ and
standard errors from simulated data for later analysis
(default is no data output)]
[PFILE = file name to receive estimates of $\hat{p}$ and
standard errors from simulated data for later analysis
(default is no data output)]
[PHIFILE = file name to receive estimates of $\hat{\phi}$ and
standard errors from simulated data for later analysis
(default is no data output)]
[UNBIAS (causes bias-corrected estimates of S and ϕ to be computed)]
[EXPECT (generates expected data values and computes estimates
and test statistics as described in Chapter 3.6);]

/* FCH, SCHEMEB, SCHEMEA, and UCH options are mutually exclusive. If none of these options are specified, complete capture history data are simulated. */
[SUBPOPULATION [=] i [WEIGHT = weight];]

/* The SUBPOPULATION statement is needed if more than one set of parameters is to be used to simulate subpopulations. The WEIGHT parameter specifies the relative weighting of each of the subpopulations so that a weighted mean over subpopulations for each parameter can be calculated to determine confidence interval coverage. */

PHI(1)=survival_prob_group_1 [survival_prob_group_2 ⋯];
PHI(2)=survival_prob_group_1 [survival_prob_group_2 ⋯];
.
.
.
PHI(k - 1)=survival_prob_group_1 [survival_prob_group_2 ⋯];

/* Only k - 1 ϕ values are required, different ϕ_{ij} values by group are optional. If only one value of ϕ_i is provided on a given line, this same value is used for the remaining groups. The parameter ϕ is a simple probability with a value in the interval 0,1. To simulate heterogeneity of survival rates, a beta probability density function can be specified as beta (α, β, lower_bound, upper_bound), where the lower_bound must be $\leq$ upper_bound, and both are in the interval 0,1. Alternatively, subpopulations can be defined with different parameters to allow the study of heterogeneity. */

Table 9.2. – Continued.

P(2)=capture_prob_group_1 [capture_prob_group_2] ··· ;
P(3)=capture_prob_group_1 [capture_prob_group_2] ··· ;
•
•
•
P(k)=capture_prob_group_1 [capture_prob_group_2] ··· ;

/* Only k-1 p values are required; there may be different values for each group. If only one value is provided per line, the remaining groups use this same value. The parameter p is a simple probability with a value in the interval 0,1. To simulate heterogeneity of capture probabilities, a beta probability density function can be specified as beta (α, β, lower_bound, upper_bound), where the lower_bound must be $\leq$ upper_bound, and both are in the interval 0,1. */

R[(1)]=#_releases_group_1 [#_releases_group_2] ··· ;
[R(2)=#_releases_group_1 [#_releases_group_2] ··· ;]
•
•
•
[R(k - 1)=#_releases_group_1 [#_releases_group_2] ··· ;]

/* Only R(1) is required for each group, as new animals must be released only on occasion 1. If new animals are released only at time 1, R = is allowed, with the occasion defaulting to 1. For multiple releases, the release identifier must be included in the specification. If a value for only group 1 is provided, the remaining groups are assumed to have the same value. */

[SUBPOPULATION [2] [WEIGHT = weight];]

*/ The SUBPOPULATION statement is required if more than one set of parameters is to be used to simulate subpopulations. */

/* The entire set of PHI, P, and R specifications can be repeated for additional subpopulations. */

PROC CHISQ [POOL (causes pooling of cells to achieve larger expected values)];

$$\begin{array}{cccc} n_{1,1} & n_{1,2} & \cdots & n_{1,\mathrm{ncols}}; \\ n_{2,1} & n_{2,2} & \cdots & n_{2,\mathrm{ncols}}; \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ n_{\mathrm{nrows},1} & n_{\mathrm{nrows},2} & \cdots & n_{\mathrm{nrows},\mathrm{ncols}}; \end{array}$$

Table 9.2. – Continued.

```
/*  The program prints the r×c contingency table, expected values, and standardized residual values. The POOL
    option only operates on cells with expected values less than 2. A total chi-square value is printed with the
    proper degrees of freedom and the significance level. For 2 × 2 tables with small expected values (<5),
    Fisher's exact test is also computed and printed. */

PROC SURVIV [CONSTRAIN (constrain estimates to 0,1 interval,
        default is unconstrained estimation)]
        [DETAIL (print details of RELEASE input estimates to SURVIV,
        with default of no output)]
        [SUMMARY (print summary of RELEASE input estimates to SURVIV,
        with default of no output)]
        [PARFILE = file name to receive SURVIVE input
        (default name is SURVIN)];

/*  PROC SURVIV generates input to program SURVIV so that numerical maximum likelihood methods can
    be used to estimate parameters and test assumptions for models not provided in RELEASE. Either
    LMREAD or CHMATRIX must have been previously executed to provide a starting model. */

PROC STOP /* Stops execution. */;
```

9.1.1. Input of Information

Two general procedures are available to input the release and recapture data to program RELEASE. First, PROC CHMATRIX allows the CH matrix to be entered for each lot. The full input stream for the example used in Chapter 2.4 for the complete capture history protocol is shown in Table 9.3. Use of the CH matrix allows full testing of model assumptions for the complete capture history protocol. As a second input format, the reduced m-array can be read with PROC LMREAD, thus allowing the user to input a summary of the data. Table 9.4 provides an example of the LMREAD procedure for the same data as those used in Chapter 2.4 and Table 9.3. PROC LMREAD (or proc lmread, as either upper or lower case is permissible) does not allow TEST 3 to be computed for the complete capture history protocol. In general, most users will want to input their sample data by using PROC CHMATRIX because this method allows the most complete analysis.

At least one space is needed to separate the capture histories in the PROC CHMATRIX input from the numbers of animals with this particular history. Extra spaces can be

added. Each command (usually a line) must end with a semicolon. The right-justified alignment, as shown in Table 9.3, is recommended for ease in checking for errors. No special line continuation character is needed because the program is searching for a semicolon to indicate the end of a statement. Note that the numbers of animals lost on capture are coded as negative values; this convention prevents the numbers of animals that are removed from being included with the numbers released. Each animal only appears in one row (i.e., capture history). A minus sign indicates that the animal is not to be included in the new releases; it does not indicate that these animals should be subtracted from a previous frequency.

Two additional options are available for the LMREAD and CHMATRIX procedures. UNBIAS generates bias-corrected estimates of ϕ_i and S (Chapter 3.4) instead of MLEs. SUMMARY or its synonym, NODETAIL, reduces the amount of output produced. Only summaries of the goodness of fit and model selection tests are printed, and the estimates of only models $H_{k-1,\phi}$, H_{2p}, $H_{1\phi}$, and H_0 are printed.

Table 9.3. – Input to program RELEASE for the example from Chapter 2.4, when PROC CHMATRIX is used to enter the capture history for animals in each group. Capture frequencies for treatment animals are in the middle column and those for control animals in the last column.

```
proc title Example from Chapter 2.4;
proc chmatrix occasions=6 groups=2;
/* If GLABELs are not used, the final column
    is assumed to be the control group */
    100000                25925    24605;
    100001                  563      605;
    100001                  -27      -36;
    100010                  508      522;
    100010                  -23      -25;
    100011                   17       23;
    100011                   -1       -1;
    100100                 1500     1678;
    100100                  -81      -57;
    100101                   45       48;
    100101                   -3       -1;
    100110                   37       44;
    100110                   -2       -2;
    100111                    1        2;
    101000                  193      207;
    101000                  -14      -10;
    101001                    5        9;
    101010                    7        4;
    101100                   16       14;
```

Table 9.3. – Continued.

```
    101100                    -1      -1;
    101101                     1       1;
    101110                     1       1;
    110000                   872     935;
    110000                   -29     -33;
    110001                    26      28;
    110001                    -1      -1;
    110010                    16      18;
    110010                    -1      -1;
    110100                    67      68;
    110100                    -3      -4;
    110101                     1       2;
    110110                     2       1;
    111000                    10      12;
    111001                     0       1;
    111100                     1       0;
glabel (1) = Treatment group;
glabel (2) = Control group;
proc stop;
```

The GLABEL statements (Tables 9.3 and 9.4) demonstrate the default labels if identifying labels for each group are not specified by the user. If three or more groups are being analyzed, the default labels for treatment groups are "Treatment Group 1," "Treatment Group 2," etc.; the final group is still considered the control group. The control group should always be the last group because it will then be the denominator in the estimates of $\hat{S} = \hat{\phi}_{t1}/\hat{\phi}_{c1}$. GLABEL statements do not change the ordering of treatments in calculating $\hat{S}$. The GLABEL statements may either occur at the beginning or end of input for the CHMATRIX and LMREAD procedures. For clarity with the LMREAD procedure, the GLABEL statements can be immediately before the m_{ij} matrix entries, as shown in Table 9.4.

Four mutually exclusive parameters are also allowed for the LMREAD procedures:

FCH	indicates m_{ij} values are from a first capture history protocol;
UCH	indicates m_{ij} values are from an unknown capture history protocol;
SCHEMEB	indicates m_{ij} values are from scheme B of the partial capture history protocol; and
SCHEMEA	indicates m_{ij} values are from scheme A of the partial capture history protocol.

No option is needed if data are collected under a complete capture history protocol.

For first capture history and unknown capture history data, only the first row of the m_{ij} matrix is needed, as that is all that is observed. For SCHEMEB data, only the first two rows are needed. The method of entering SCHEMEA data is the same as that for complete capture history data.

The capture histories for PROC CHMATRIX are normally specified as a string of zeros and ones (no embedded blanks) to specify animals that are not (0) and are (1) captured. To specify unknown capture history data, zeros are replaced with dots (.) to indicate that the user does not know actual capture histories.

Table 9.4. – Input to program RELEASE for the example from Chapter 2.4 in which PROC LMREAD is used to enter the reduced m-array for each group. Data from the treatment group are entered first, followed by the data from the control group.

```
proc title Example from Chapter 2.4;
proc lmread occasions=6 groups=2;
/* Treatment group */
glabel(1)=Treatment group;
30000    1000    235    1677    590   ;
 1029     238   1669     549    590   ;
           11     73      17     27   ;
                  20       7      5   ;
                           43     50   ;
                                  19   ;
/* Control group */
glabel(2) = Control group;
29000    1071    250    1862    616   ;
 1104     247   1832     571    641   ;
           13     75      19     29   ;
                  17       4     10   ;
                           50     52   ;
                                  26   ;
proc stop;
```

PROC STOP is used to exit the program and is provided to stop execution in a data file when additional data follow but the user does not desire to run through the program. This feature allows data sets to be "saved" by moving them to the end of the input file, below the PROC STOP statement. PROC STOP allows the stacking of multiple data sets in an input file, but runs only the first data sets. PROC STOP is not required at the end of the input file; the end of a file will cause RELEASE to end execution.

9.1.2. Program Output

All input instructions read by program RELEASE are echoed in the output with the identifying INPUT--- to delineate them. The user should check this information carefully to be certain that the input is correct. PROC CHMATRIX provides a summary of the data in reduced m-arrays for each group. Again, careful checking is recommended at this point to guard against data entry errors. For $k < 8$, PROC CHMATRIX will print the full m-array if the FULLM option is specified.

Program RELEASE then computes the parameter estimates, standard errors, and confidence intervals for each appropriate model. All the within- and between-group test statistics are printed with the associated degrees of freedom and significance levels. Tests are labeled as in Tables 2.1-2.4. Examples of the output of RELEASE are shown throughout Part 2.

9.1.3. Monte Carlo Simulator

PROC SIMULATE can be used to generate Monte Carlo data for any of the four sampling protocols. Particular models within a specific protocol are specified by the user's assignment of appropriate parameter values. The details of SIMULATE are given in Table 9.2. Table 9.5 provides an example where 1,000 data sets are to be generated under model $H_{1\phi}$ of the complete capture history protocol, with 10,000 marked animals initially released in each group. In the example, $S = \phi_{t1}/\phi_{c1} = 0.8/0.9 = 0.889$. Because other parameters are equal across groups, only one numerical value is necessary for each line (e.g., $p(4) = 0.1$ would serve for both treatment and control groups). Program RELEASE provides a detailed summary of the Monte Carlo study, including estimated expected values of the estimators under various models, empirical estimates of sampling variance, and performance of various tests. In the example in Table 9.5, one could estimate the power of the test of model H_0 versus model $H_{1\phi}$, among other things. The results of a simple model selection algorithm are printed at the end of a simulation run. The algorithm selects a model when the P value for a test is less than the α level. Computer time for 1,000 repetitions with two groups would take several hours on an IBM-PC/AT.

Table 9.5. – Example input to program RELEASE to simulate an experiment where the only difference in survival between two groups of animals is in $\phi_{a1} \neq \phi_{c1}$. Model $H_{1\phi}$ is shown with 1,000 repetitions requested (i.e., nsim = 1000).

```
proc title Simulation of model H1PHI with complete capture history protocol;
proc simulate nsim=1000 occasions=5 groups=2 seed=4567655;
        phi(1)=0.8 0.9;
        phi(2)=0.9 0.9;
        phi(3)=0.9 0.9;
        phi(4)=0.9 0.9;
        p(2)=0.1 0.1;
        p(3)=0.1 0.1;
        p(4)=0.1 0.1;
        p(5)=0.1 0.1;
        R=10000 10000;
proc stop;
```

The DETAIL specification on PROC SIMULATE causes detailed output from each of the simulations. Thus, parameter estimates for each model and χ^2 goodness of fit tables are all printed. If the number of simulations performed is large, printed output will be large. Thus, DETAIL should only be used with NSIM ≤ 10. The SUMMARY specification causes a less detailed printout of each simulation. Only a summary of the χ^2 goodness of fit tables is printed. Likewise, only a summary of the tests for differences between groups is printed. Finally, estimates from only the most complex model ($H_{k-1,\phi}$) and the three simplest models (H_{2p}, $H_{1\phi}$, and H_0) are printed. For data sets with a large number of occasions or a large number of groups, the SUMMARY option saves a great deal of space in the output. However, as with the DETAIL option, only a small number of simulations should be performed while this option is set.

The SIMULATE procedure also supports the UNBIAS parameter specification, allowing simulation of estimates that have been bias-corrected. The default is to provide MLEs, which are asymptotically unbiased but may exhibit small sample bias (Section 3.4).

Three parameters on PROC SIMULATE are used to name output files to receive the estimates and associated standard errors for each of the parameters estimated under each of the models. SFILE specifies a file to receive the estimates of S and se($\hat{S}$), with PFILE and PHIFILE providing the same capabilities for p and ϕ, respectively. These files might be used as input to a statistical package to perform a more thorough analysis of the simulations than the summary printed by PROC SIMULATE (e.g., to examine the distribution of estimators such as $\hat{S}$). The columns of the file specified with the SFILE parameter are

Column	Information
1	iteration number (1,2, ..., NSIM)
2	model name
3	group number of numerator
4	group number of denominator
5	occasion
6	$\hat{S}$
7	se($\hat{S}$)

The columns of the files specified with the PFILE and PHIFILE parameters are

Column	Information
1	iteration number (1,2, ..., NSIM)
2	model name
3	group number
4	occasion
5	$\hat{p}$ or $\hat{\phi}$ depending on the file
6	se($\hat{p}$) or se($\hat{\phi}$) depending on the file

Values of estimates that were not identifiable are written to these files as missing values, signified by a single period (.).

A random number seed to simulate data with PROC SIMULATE is provided in the SEED = value specification. Generally the seed should be a 5- or 7-digit, odd random integer. Specifying the same seed for two PROC SIMULATEs generates identical data (on a given computer) if the same model and parameters are used.

A final option to SIMULATE, EXPECT, can be used to generate the expected values of the capture histories. These expected data values are then used to compute the various tests and estimators. A limitation of the EXPECT option is that the expected capture history frequencies are only precise to the nearest integer. Thus, rounding errors can be large. One approach to circumventing this limitation is to make the number of animals released large, i.e., R_{v1} large.

The subcommands for PROC SIMULATE specify the true parameter values to be simulated for ϕ, p, and the numbers of animals released, as shown in Table 9.2. Normally the parameters ϕ and p are simple probabilities. However, if a more realistic experiment is to be simulated, each animal can be given a unique value for ϕ_i and p_i by generating the value of ϕ or p from a beta probability density function. If BETA (α, β, lower_bound, upper_bound) is specified in place of a simple probability, the parameter is generated from a beta probability density function with parameters α and β and then scaled to the interval (lower_bound, upper_bound). Examples of the many possible shapes of the beta distribution for a range of αs and βs are shown in Figure 9.1. Note that only ϕ and p can be specified as beta random variables. The initial releases (R_{vi}) must be fixed, not random, variables.

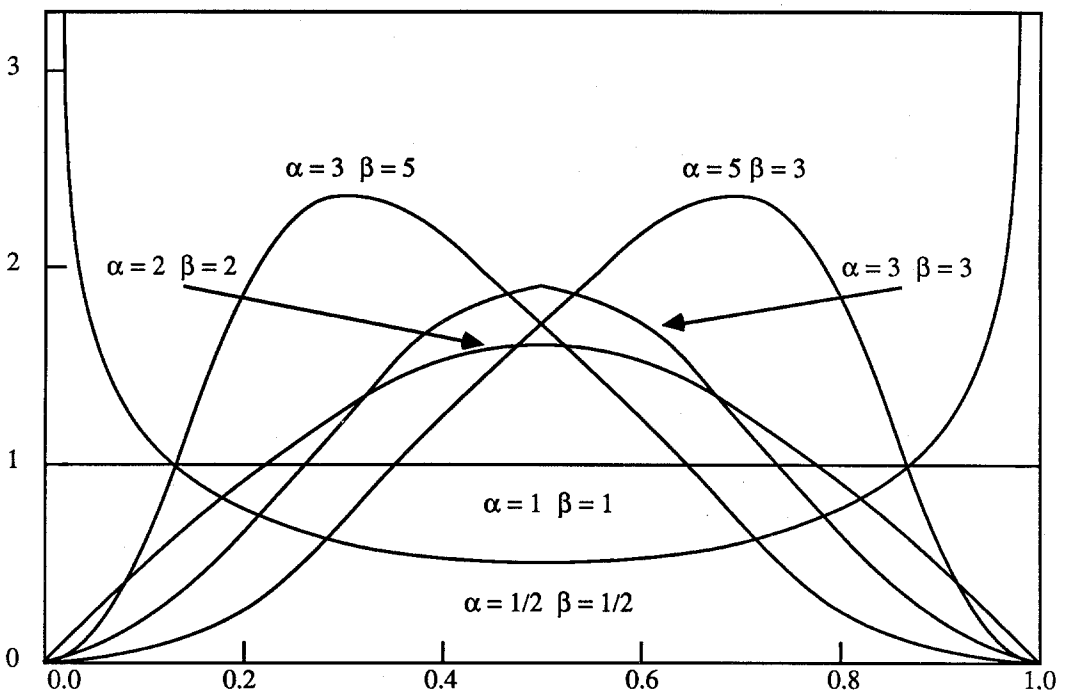

Figure 9.1. – Examples of the beta distribution for various values of α and β.

The beta probability density function allows one way to simulate the heterogeneity of population parameters. Only variability within a single population is assumed. Often, however, the biological population to be studied is thought to consist of two or more subpopulations. PROC SIMULATE allows such a multifaceted population to be simulated through the use of the SUBPOPULATION statement (as another way to simulate heterogeneity). Each SUBPOPULATION statement tells SIMULATE to set up another subpopulation with its own unique parameter values (which may include beta distributions).

The SUBPOPULATION statement also has the optional parameter WEIGHT, which may be specified to provide the weight of this particular subpopulation relative to the others being simulated. These weights are needed to calculate the average of each parameter over all subpopulations. For example, if subpopulation 1 is thought to occur one-half as often as subpopulation 2, the weights for 1 and 2 might be WEIGHT = 1 and WEIGHT = 2 or WEIGHT = 0.5 and WEIGHT = 1, both of which indicate that subpopulation 2 is to be weighted twice as heavily as subpopulation 1. If weights are not specified in the SUBPOPULATION statements, the default for each subpopulation is 1.0.

Normally, the weight of each subpopulation would be proportional to the R_i values. The average across subpopulations would then approximate the population mean. However,

because the capture probabilities may vary by subpopulation, the R_i values may not always provide the relative weight of the subpopulation. The WEIGHT specification provides a completely general approach to providing the relative weights for each subpopulation, from which the true population parameter values are determined.

A second example is a Monte Carlo study of model H_{2p} under scheme B over four occasions with a treatment and control group (Table 9.6). The output of program RELEASE is shown in Table 9.7. Material in Part 5 was generated by using PROC SIMULATE.

Table 9.6. – Example input to program RELEASE that simulates an experiment under model H_{2p} when the scheme B sampling protocol is used.

```
proc title Simulation of H2p under schemeB;
proc simulate nsim=1000 occasions=4 groups=2 schemeB;
  phi(1)=0.56 0.8;
  phi(2)=0.8;
  phi(3)=0.7;
  p(2)=0.3 0.2;
  p(3)=0.2;
  p(4)=0.2;
  R=2000 2000;
  proc stop;
```

Table 9.7. – Output of program RELEASE for the input given in Table 9.6.

Simulation of H2p under schemeB

Simulation Results

Model	Group	Parameter	Mean	Standard Deviation	95% CI Lower	95% CI Upper	Cover	n
H2Phi	1	Phi(1)	0.565253	0.047447	0.562312	0.568193	0.947	1000
		SE Phi(1)	0.046768	0.007197	0.046322	0.047214		
		p(2)	0.299061	0.027234	0.297373	0.300749	0.952	1000
		SE p(2)	0.027368	0.001031	0.027304	0.027431		
	2	Phi(1)	0.806544	0.074056	0.801954	0.811134	0.946	1000
		SE Phi(1)	0.071395	0.011812	0.070663	0.072127		

Table 9.7. – Continued.

		p(2)	0.200008	0.020675	0.198726	0.201289	0.951	1000
		SE p(2)	0.019991	0.000888	0.019936	0.020046		
	All	S(1,2)	0.706586	0.086604	0.701218	0.711954	0.944	1000
		SE S(1,2)	0.085198	0.011021	0.084515	0.085881		
H2p	1	Phi(1)	0.562724	0.038212	0.560355	0.565092	0.943	1000
		SE Phi(1)	0.038045	0.003950	0.037800	0.038290		
		p(2)	0.299681	0.022966	0.298258	0.301105	0.954	1000
		SE p(2)	0.023620	0.001043	0.023556	0.023685		
	2	Phi(1)	0.804295	0.057696	0.800719	0.807871	0.941	1000
		SE Phi(1)	0.054673	0.006154	0.054291	0.055054		
		p(2)	0.199920	0.017275	0.198849	0.200990	0.938	1000
		SE p(2)	0.016648	0.000858	0.016595	0.016701		
	All	S(1,2)	0.701283	0.045117	0.698487	0.704080	0.946	1000
		SE S(1,2)	0.044108	0.002670	0.043943	0.044274		
H1Phi	1	Phi(1)	0.620578	0.041482	0.618007	0.623149	0.735	1000
		SE Phi(1)	0.041288	0.004313	0.041020	0.041555		
		p(2)	0.240779	0.016320	0.239768	0.241791	0.058	1000
		SE p(2)	0.016403	0.000456	0.016375	0.016431		
	2	Phi(1)	0.746440	0.050344	0.743320	0.749560	0.735	1000
		SE Phi(1)	0.047853	0.005220	0.047529	0.048176		
		p(2)	0.240779	0.016320	0.239768	0.241791	0.294	1000
		SE p(2)	0.016403	0.000456	0.016375	0.016431		
	All	S(1,2)	0.832419	0.040765	0.829892	0.834945	0.077	1000
		SE S(1,2)	0.041181	0.002024	0.041056	0.041307		
H0	1	Phi(1)	0.683509	0.042774	0.680858	0.686160	0.093	1000
		SE Phi(1)	0.041285	0.004661	0.040997	0.041574		
		p(2)	0.240779	0.016320	0.239768	0.241791	0.058	1000
		SE p(2)	0.016403	0.000456	0.016375	0.016431		
	2	Phi(1)	0.683509	0.042774	0.680858	0.686160	0.230	1000
		SE Phi(1)	0.041285	0.004661	0.040997	0.041574		
		p(2)	0.240779	0.016320	0.239768	0.241791	0.294	1000
		SE p(2)	0.016403	0.000456	0.016375	0.016431		

Table 9.7. – Continued.

Power of Tests

Test	<0.01	<0.05	<0.10	<0.20	<0.50
TEST 1.T3	0.016	0.063	0.115	0.223	0.534
TEST 1.R2	0.010	0.048	0.100	0.210	0.527
TEST 1.T2	0.976	0.993	0.997	0.998	1.000
TEST 1.R1	0.882	0.964	0.987	0.995	1.000
TEST 1	0.992	1.000	1.000	1.000	1.000

Model Selection Results

Type I Error Level for Individual Tests

Model	0.01	0.05	0.10	0.20	0.30
H3p	0.016	0.063	0.115	0.223	0.333
H2Phi	0.010	0.044	0.089	0.163	0.209
H2p	0.950	0.886	0.793	0.613	0.458
H1Phi	0.022	0.007	0.003	0.001	0.000
H0	0.002	0.000	0.000	0.000	0.000

The final example deals with the simulation of populations exhibiting heterogeneity. A beta distribution is used to mimic individual heterogeneity in survival or capture processes, or (as in Table 9.8) two subpopulations are used, each with different parameters but weighted equally because the default weights are 1.0 for each subpopulation. For example, for ϕ_{11}, treatment group 1 has values of 0.85 (200 animals) and 0.75 (200 animals), whereas treatment group 2 has values of 0.95 (200 animals) and 0.85 (200 animals). The Monte Carlo results in Part 5 for heterogeneity were generated by using the subpopulation approach.

Table 9.8. – Example input to program RELEASE to simulate an experiment under model $H_{1\phi}$ with heterogeneity between two subpopulations. Here, heterogeneity affects both the survival and capture probabilities.

```
proc title model H1PHI for complete capture history data;
proc simulate seed=1923949 nsim=1000 occasions=5 groups=2;
subpopulation 1;
     phi(1)=.85 .95;
     phi(2)=.95 .95;
     phi(3)=.95 .95;
     phi(4)=.95 .95;
     p(2)=.9;
     p(3)=.9;
     p(4)=.9;
     p(5)=.9;
     R=200 200;
subpopulation 2;
     phi(1)=.75 .85;
     phi(2)=.85 .85;
     phi(3)=.85 .85;
     phi(4)=.85 .85;
     p(2)=.7;
     p(3)=.7;
     p(4)=.7;
     p(5)=.7;
     R=200 200;
     proc stop;
```

9.2. Examples

In this section, we present two examples of the input stream for program RELEASE, using material given earlier. Table 9.9 provides the RELEASE input for the example in Chapter 2.5, and Table 9.10 provides the input for the starling example discussed in Chapter 7.4.

Table 9.9. – Input for the example given in Chapter 2.5 for the partial capture history protocol under scheme A. Input procedures for the CH matrix (top) and *m*-array (bottom) are illustrated. Output from program RELEASE can be seen in Chapter 2.5.

```
proc title Example from Chapter 2.5, scheme A, using CHMATRIX;
proc chmatrix occasions=6, groups=2;
/* Frequencies for the treatment group are entered first, following the
    capture histories */
    100000   25925    24605;
    100001     563      605;
    100001     -27      -36;
    100010     508      522;
    100010     -23      -25;
    100011     -18      -24;
    100100    1500     1678;
    100100     -81      -57;
    100101     -48      -49;
    100110     -40      -48;
    101000     193      207;
    101000     -14      -10;
    101001      -5       -9;
    101010      -7       -4;
    101100     -19      -17;
    110000     872      935;
    110000     -29      -33;
    110001     -27      -29;
    110010     -17      -19;
    110100     -73      -75;
    111000     -11      -13;
  proc stop;

  proc title Example from Chapter 2.5, scheme A, using LMREAD;
  proc lmread schemeA occasions=6 groups=2;
 30000    1000     224    1588    526;
  1029     238    1669     549    590;
            11      73      17     27;
                    19       7      5;
                            40     48;
                                   18;
```

Table 9.9. – Continued.

```
29000    1071     237    1775    546;
 1104     247    1832     571    641;
           13      75      19     29;
                   17       4      9;
                           48     49;
                                  24;

proc stop;
```

Table 9.10. – Input stream for the starling data used in Chapter 7 (Stromborg et al., in press). The reduced m-array is read with PROC LMREAD; selected output is given in Chapter 7.4.

```
proc title Dosing study of starlings;
proc lmread occasions=6 groups=2;

   /* Treatment group */
    60       24     28     34     32;
    24        6      9      2      0;
             22      1      0      0;
                    24      0      0;
                           30      0;
                                  21;
    /* Control group */
    61       22     31     40     35;
    22       13      9      2      0;
             18      1      0      0;
                    30      0      0;
                           33      1;
                                  28;
 proc stop;
```

9.3. Interface with Program SURVIV

PROC SURVIV is used to generate input to program SURVIV (White 1983) so that more general models can be used. Program SURVIV allows more flexibility in the construction of tests and estimates from the data; however, it also requires a better

understanding of the statistical methods being used. Thus, we do not recommend program SURVIV for the casual user.

Either the CHMATRIX or LMREAD procedures must have been used to enter a set of capture-recapture data into program RELEASE. These data are then used by RELEASE to construct the input for SURVIV. The PARFILE parameter specifies the file for the SURVIV input, with the default being SURVIN.

An important option on the PROC SURVIV statement is CONSTRAIN, which sets up the SURVIV run with constrained parameter estimation of the ϕs and ps – i.e., the parameter estimates are constrained to the interval (0,1). The estimates of ϕ_{ai} computed by program RELEASE are not constrained, and occasionally exceed 1.0, meaning that the estimate is outside the range of admissible parameter values. The DETAIL and SUMMARY options specify that the output produced by the parameter process should be listed in the output file. DETAIL produces full output as would the LMREAD or CHMATRIX procedures. SUMMARY produces limited output corresponding to the SUMMARY option of LMREAD or CHMATRIX. The default is to produce no output from PROC SURVIV, the only item of interest being the input file to program SURVIV.

9.4. Executing RELEASE

Program RELEASE can be executed in two different modes: interactive and batch. Running RELEASE without additional parameters places the program in the interactive mode; all output scrolls across the screen and is not saved to a file. In the interactive mode, the user is queried for input from the screen. Numerous menus provide input to each of the procedures in Table 9.2. Based on the user's responses, RELEASE maintains a log of the input statements produced from the interactive session. This log is initially placed on the file RELEAS00.INP, but successive executions of RELEASE produce sequentially numbered files. Thus, if the file RELEAS00.INP already exists, the log file would be placed in RELEAS01.INP.

Output from program RELEASE can be saved to a file by using the O=filename parameter on the execution line. By specifying O=XYZ on the execution line, output from RELEASE is placed in file XYZ. Using the O parameter causes no output to be produced on the screen.

RELEASE is placed in batch mode by specifying I=filename on the execution line. The I parameter specifies an input file for the run, thus telling RELEASE to read this file rather than query the user for input. The command

RELEASE I=TEST.DAT O=TEST.OUT

specifies that the file TEST.DAT is to be read to obtain input, and output is to be placed in the file TEST.OUT. If an input file is not specified, RELEASE is put into the interactive mode. If an output file is specified, but no input file specified, the user is queried for input

through the interactive mode, but output is saved to the output file and not printed on the screen. Typically, a user would use the interactive mode to build an input file through the log mechanism, verifying the output as it was scrolled across the screen. Once the appropriate commands are found to work, the log file created would be executed in batch mode to produce an output file for printing. An editor could be used to modify the log file, if necessary, before RELEASE is run in the batch mode to produce an output file.

Besides the I and O parameters on the execution line, two additional parameters can be specified. NOECHO causes the input lines to be omitted in the output. Although this option saves some paper when the output is printed, it is not advised because of the difficulty in locating errors in the input. LINES=# specifies the number of output lines to print per page; the default is LINES=60.

9.5. Programming Details

Program RELEASE has about 8,000 lines of FORTRAN code in 78 subroutines. RELEASE is easy to use on a microcomputer. It can be modified for use on mainframe computers or microcomputers that are not IBM-compatible if a FORTRAN 77 compiler is available and the interactive interface is removed. Program RELEASE is provided on three 5.25-inch double-sided, double-density floppy disks. Five files are present on the first disk: RELEASE.EQE is the executable code in compressed form; RELEASE.DAT is example input, including many of the examples shown in this monograph; INSTALL.BAT installs RELEASE on a hard disk as RELEASE.EXE, and UNSQ.COM and SQPC.COM are public domain utilities to compress and uncompress files. The install command provided on the disk uses UNSQ.COM to uncompress RELEASE.EQE into RELEASE.EXE on the hard disk. RELEASE.EXE requires 425k of disk space, and is dimensioned for 15 capture occasions, 3 groups, and up to 512 different capture histories. The second disk contains the source code for RELEASE, and all but one of the necessary files to construct a new version of the code. The third disk contains a RMFORT-compatible library necessary to construct a new version of RELEASE and USERMAN.DOC, a printable copy of this part of the monograph plus updates to the documentation.

9.5.1. Source Code

The source code for program RELEASE is available to users as described above. For the IBM-PC, the code (consisting of 84 files) is distributed on a 360K diskette. The main program is in the file RELEASE.FOR. All subroutines are in the files *.F. Six files containing common blocks are INCLUDE files: HEADST (containing pagination headings), STATUS (containing I/O units and variables controlling input), MODELC (containing variables pertaining to data), ESTCOM (containing variables pertaining to estimates), SIMCOM (containing variables pertaining to simulations), and SCREEN (containing variables pertaining to the interactive interface). Each of these six files includes comments describing

each of the variables in the associated common blocks. The file MAKEFILE is provided as input to MAKE.EXE (also supplied) to compile and link these routines. The file LINKREL.INP provides input to the LINK command. The Ryan-McFarland Fortran (or IBM Professional Fortran) compiler must be used to maintain compatibility with the object library provided.

Program RELEASE has been designed to be easy to change by experienced FORTRAN programmers. The most likely change will involve the dimension limitations for the numbers of treatment groups and capture occasions. These limits are set in common blocks through PARAMETER statements. The number of occasions is controlled by the parameter MAXOCC set in the INCLUDE file MODELC. Several other parameters that depend on MAXOCC are also set in MODELC, as described by comment statements. Likewise, the parameter MAXGRP is set on MODELC to define the maximum number of treatment groups. Again, additional parameters that depend on MAXGRP are described in comment statements. If MAXGRP or MAXOCC are changed in the file MODELC, all the routines that include this file must be recompiled. This may be done with the MAKE utility, which will generate a new RELEASE.EXE file. Many variables, including I/O unit numbers, are set in the main program. These defaults can be changed by recompiling RELEASE.FOR and relinking the code by using MAKE.