101
F653t
#43

Technical Paper No. 43

DOCUMENTATION FOR THE SPATIAL ANALYSIS SYSTEM(SPAN) FOR RESOURCE USE BY ANIMALS

David A. McElroy
William J. Lindberg

FLORIDA SEA GRANT COLLEGE

DOCUMENTATION FOR THE SPATIAL ANALYSIS SYSTEM (SPAN) FOR RESOURCE USE BY ANIMALS

by

David A. McElroy
Department of Computer and Information Sciences
University of Florida, Gainesville, FL

and

William J. Lindberg
Department of Fisheries and Aquaculture
University of Florida, Gainesville, FL

Project No. R/LR-B-14
Grant No. NA85AA-D-SG059

Technical Papers are duplicated in limited quantities for specialized audiences requiring rapid access to information. They are published with limited editing and without formal review by the Florida Sea Grant College Program. Content is the sole responsibility of the author. This paper was developed by the Florida Sea Grant College Program with support from NOAA Office of Sea Grant, U.S. Department of Commerce, grant number NA85AA-D-SG059. It was published by the Sea Grant Extension Program which functions as a component of the Florida Cooperative Extension Service, John T. Woeste, Dean, in conducting Cooperative Extension work in Agriculture, Home Economics, and marine Sciences, State of Florida, U.S. Department of Commerce, and Boards of County Commissioners, cooperating. Printed and distributed in furtherance of the Acts of Congress of May 8 and June 14, 1914. The Florida Sea Grant College is an Equal Employment-Affirmative Action employer authorized to provide research, educational information and other services only to individuals and institutions that function without regard to race, color, sex, or national origin.

TECHNICAL PAPER NO. 43
February 1986

TABLE OF CONTENTS

Section		Page
1.0	RATIONAL AND SYSTEM OBJECTIVE FOR SPAN	1
2.0	CALCULATION OF STATISTICS FROM MAPPED DATA	2
2.1	Mean Size and Distance Differences	2
2.2	Kappa Statistics	3
3.0	PRELIMINARY DESIGN	5
3.1	Hardware Resources	5
3.2	Software Resources	6
3.3	Languages	6
3.4	Modules	6
4.0	DETAIL DESIGN	10
4.1	Required Structures	10
4.2	Global Variables	14
4.3	Subroutines	17
5.0	USER'S MANUAL	19
5.1	Invalid Responses	19
5.2	Interrupt	19
5.3	Prompts and Appropriate Responses	19
6.0	INTERPRETATIONS	22
6.1	Output for the Observed Values of Statistics	23
6.2	Output for the Monte Carlo Trials	24
6.3	Comparing Observed Values to Monte Carlo Trials	27
APPENDIX A	EXAMPLE	29
APPENDIX B	SUMMARY	32
APPENDIX C	PROGRAMS	35

Abstract: "Documentation for the Spatial Analysis System (SPAN) for resource use by animals", D.A. McElroy and W.J. Lindberg. 39 pages including 3 appendices.

Nearest-neighbor analyses have been used with mapped data for tests of spatial dispersion and association in plant and animal ecology. This paper fully describes a computer software package developed to use Monte Carlo trials instead of chi-squared distributions for assigning probabilities to observed values of nearest neighbor statistics. The program can factor-out the unique geometry of resources in a sample plot, which can affect locations of animals, thus testing for direct patterns among the animals independent of their resource patterns. The Kappa statistic for association is also calculated although its application has met with limited success. A users manual and the Fortran program language is included.

1.0 RATIONAL AND SYSTEM OBJECTIVE FOR SPAN

Nearest neighbor analyses have long been used in ecology for tests of dispersion (Clark & Evans, 1954, Ecology 30:445-453; Thompson, 1956, Ecology 37:391-394) and association (Pielou, 1961, J. Ecology 49:255-269). The procedure has been to map the locations of individuals (i.e. plants, animals, colonies, etc.), and then to designate each individual in turn as a base from which its nearest neighbor is determined. In tests of dispersion, distances among neighbors constitute the data, while in tests for association the relationships within pairs (e.g. male base-female neighbor, or species A base-species B neighbor) are tallied in contingency tables.

For tests of association Meagher and Burdick (1980, Ecology 61:1253-1255) called attention to the problem that reciprocal nearest neighbors violate the assumption of independence for use of chi-squared distributions. They properly advocated Monte Carlo trials, as an alternative to using chi-squared distributions, to avoid unacceptable levels of Type I errors.

The procedure offered by Meagher and Burdick is suited for testing association within single plots, as might apply to single sampling or mapping efforts. However, a test statistic is also needed which can serve as an index to the degree of association and can allow experimental comparisons among plots. Fleiss (1981, Statistical Methods for Rates and Proportions, John Wiley and Sons, p.217) describes the Kappa-statistic as having desired characteristics. Kappa corrects for the association expected by chance alone, and takes on values between -1 and +1, with negative values indicating negative association, values near 0 indicating no association, and positive values indicating positive association. The investigator may also choose to test for association between like-types (e.g. male-male) or unlike-types (e.g. male-female), depending upon the mechanisms hypothesized for the association. Despite these desirable attributes, the concern for Type I error from reciprocal nearest neighbors also exists for Kappa, so likewise, Kappa requires Monte Carlo trials.

For tests of dispersion that same argument about reciprocal nearest neighbors violating independence would seem to apply. Furthermore, Monte Carlo trials provide an excellent test for dispersion when the hypothesized pattern among individuals is confounded by a spatial pattern imposed by another factor. Such would be the case when locations of mapped individuals are, to some extent, dependent on a resource that is itself clumped or over dispersed.

We have been studying the behavioral ecology of benthic crabs associated with fixed refuges. One crab, *Pilumnus sayi*, occupies bryozoan colonies, or heads, attached to seagrass blades (Lindberg, 1980, Oecologia 46:338-342; Lindberg and Frydenborg, 1980, Behaviour 75:235-250).

Another, *Menippe mercenaria*, occupies burrows excavated primarily at the edge of seagrass banks or patchy rock outcroppings. An in situ experiment in which refuge spatial pattern was manipulated has already been conducted for *P. sayi* (Lindberg and Stanton, forthcoming), and a comparable experiment is underway for *M. mercenaria*. Analysis of these results prompted our development of computer software for tests of dispersion and association with the above considerations in mind.

The overall objective for this software package is to provide convenient tests for both dispersion among conspecifics and association between sexes. Furthermore, it can factor out whatever pattern might be imposed on the subjects by the patterns of their resources. The Monte Carlo trials on which these tests are based can also be structured to simulate behavior of the subjects, either empirically known or hypothesized. As with earlier procedures for testing dispersion, this package employs the mean distances between n'th nearest neighbors as test statistics. Size differences among n nearest neighbors can also be tested. For testing association, this package calculates Kappa-statistics for experimental or sampling plots, given their mapping coordinate inputs. Probability distributions for the test statistics are generated from a large number of Monte Carlo trials. Output, therefore, allows statistical analysis of dispersion and association within plots as well as comparisons among plots through unprogrammed follow-up tests.

2.0 CALCULATION OF STATISTICS FROM MAPPED DATA

2.1 Mean Size and Distance Differences

The mean size difference and mean distance apart are computed among all neighbors of a given rank, i.e. 1st NN, 2nd NN, 3rd NN, etc.. The procedure is:

1. Designate each subject in turn as the base, starting with the first subject and continuing to the last. For each base, find its nearest neighbor or in case of ties find the group of nearest neighbors.

2. For each base crab-neighbor crab pair, check to make sure the two crabs' have not yet been compared within the same nearest neighbor group. If the differences have not already been computed, determine the size and distance differences.

3. The overall size and distance differences are updated and the sample size for the nearest neighbor group is incremented.

4. If there are more crabs to be analyzed, go to step number 2.

5. The mean size and distance differences for each nearest neighbor group are computed by dividing the overall differences by the sample size in each nearest neighbor group.

6. If there are more base crabs to analyze, go to step number 1.

In step number 2 a check was used to avoid determining the size and distance differences for reciprocal neighbors. If you have two crabs who are each other's first nearest neighbor, their differences are only computed once to maintain the assumption of independence. Lets say the 4th crab is the base and the 5th crab is found to be in the 4th crabs 1st nearest neighbor group. This is the first time the two crabs are analyzed in the 1st nearest neighbor group so the differences are computed. Now, the 5th crab is the base and the 4th crab is found to be in the 5th crab's 1st nearest neighbor group. Because the differences between these two crabs have already been computed in the 1st nearest neighbor group, the differences will not be computed again. If the 4th crab was in the 5th crab's 2nd nearest neighbor group, the differences would be computed.

2.2 Kappa Statistics

The following procedure is used by the Resource Use Spatial Analysis (SPAN) System to calculate Kappa, in the context of nearest neighbor analyses for association among males and females. Conceptually, Kappa is (Io - Ie) divided by (1 - Ie), where Io is an observed proportional value, Ie is the proportional value expected simply by chance, and (1-Ie) is the maximum value possible for that difference between observed and expected. We denote the cell frequencies for a 2 by 2 contingency table as A, B, C and D, with the respective proportions as a, b, c and d (see 3 below). Input consists of the coordinates and sex of the crabs (i.e. subjects). The coordinates are use d to determine nearest neighbor relationships and the sex is used to determine the association among males and females.

1. Designate each subject in turn as the base, starting with the first subject and continuing to the last. For each base find its first nearest-neighbor (N-N). For the second N-N statistics, find each base's second N-N; for third N-N statistics, find each base's third N-N, etc.

2. For each crab in the base crab's N-N group, determine the sex relationship (i.e. male-male, male-female, female-male and female-female), where the first sex belongs to the base crab.

3. Tally the sex relationships into a 2 by 2 contingency table.

		N-N: M	N-N: F
Base	M	A	B
	F	C	D

For example, crab #1 in Appendix A is a male and its first N-N is crab #4, a female, therefore, the frequency for a male-female pair (B) should be incremented by 1. Now, if the base crab was #5, the first N-N's are #2, #3 and #6. In the case of ties, because there are 3 crabs in the base crab's N-N group, only 1/3 would be added to the appropriate frequency. The base crab is a male. Crabs #3 and #6 are males while crab #2 is a female. Therefore, A is incremented by 2/3 and B is incremented by 1/3. In the case of 2 crabs in the N-N group, only 1/2 would be added to the appropriate frequency.

4. After cell frequencies have been tallied, they are converted to proportions by dividing each cell frequency by the total number of subjects in the data set. For example, in Appendix A, A equals 1 with 7 crabs in the data set, therefore, the proportion (a) is 1/7.

5. After cell proportions are calculated, the marginal row totals (P1, Q1) and column totals (P2, Q2), the observed between-sex proportions (Po), and the expected proportion (Pe) are calculated according to the following formulas:

$$P1=a+b$$
$$Q1=c+d$$
$$P2=a+c$$

Q2=b+d
PO=b+c
PE=P1*Q2+P2*Q1

(Note that a+b+c+d will equal 1 since proportions are being used, therefore, P1+Q1=1 and P2+Q2=1.)

6. Next, Kappa-statistics are calculated for association among males (or females) and between males and females. These are referred to as Kappa (K) and Overall Kappa (Ok), respectively, to be consistent with the terminology of Fleiss (1981). Formulas for these statistics are:

$$K = \frac{2(ad-bc)}{Pe}$$ (formula 13.9 p. 217 of Fleiss. 1981)

$$OK = \frac{Po-Pe}{1-Pe}$$ (formula 13.12 p. 219 of Fleiss, 1981)

7. If only 1 iteration is requested the observed Kappa values for the data set are generated. However, if more than 1 iteration is requested (generally a large number), then that number of Monte Carlo trials are run to generate probability distributions for first through as many as five nearest neighbors. The procedures above are repeated for each iteration of the program.

3.0 PRELIMINARY DESIGN

3.1 Hardware Resources

3.1.1 Polycorder version 5 by Omnidata International, Inc.

This device is used to record data in the field and its communications protocol is used to transfer the data to a mainframe via a modem.

3.1.2 Mainframe - Digital VAX 750

This mainframe is used to store and to analyze statistically the data via programs developed for a particular purpose.

3.1.3 DEC (Digital Equipment Corporation) printers.

The printers are used to obtain copies of the original data and to obtain hard copies of the output

from system programs.

3.2 Software Resources

VAX editor, linker, and compilers.

3.3 Languages

The languages used in this system are FORTRAN 77, and DCL (DEC Control Language). FORTRAN 77 was chosen for its power in numerical analyses for determining various statistics. DCL is used to greatly reduce the number of operations the user has to perform. DCL is used to perform system level commands (i.e. COPY, RENAME, PRINT, etc.) that cannot be performed using conventional languages (i.e. PASCAL, FORTRAN, etc.).

3.4 MODULES

There are a number of modules (programs) that make up the system each designed for a specific purpose.

3.4.1 CRABS

CRABS is written in FORTRAN 77. Given cartesian coordinates for the location of unit resources (i.e. refuges) and the size and sex of the individuals (i.e. crabs) that occupy such units, this program will calculate the Kappa statistics for male-male and male-female association, the mean size difference, and the mean distance difference among n nearest-neighbors, with n being less than or equal to 5. The output will be in tabular form giving the observed values for these statistics which can then be compared to their frequency histograms from iterative Monte Carlo trials.

One Monte Carlo trial consists of randomly assigning crabs to refuges and then calculating the same statistics as for the observed. Such randomly generated values are stored while the process is repeated a large number of times to generate frequency distributions from which probabilities can be figured. The randomization of crabs on heads can take into account empirically determined or hypothesized behavior of the crabs:

1. Competition for refuges results in sole ownership, so the randomization can employ sampling with or without replacement of refuges back into the same pool.

2. Smaller, subordinate males might assume satellite tactics in the presence of larger,

dominate males, and thus obscure expected spatial patterns. Individuals most likely to be such satellites can be eliminated from analyses.

3. Crabs express preferences for qualitative and quantitative characteristics of refuges. A function in the randomization process can adjust the probabilities of refuge types receiving crabs according to empirical probabilities.

In generating both the observed values and the random values, this program avoids the double entries typically produced by reciprocal nearest neighbor pairs. Each unique pair constitutes the source of data, rather than a base-neighbor pair which can then have its reciprocal duplicate the entry. In other words, data from the pairing of any two individuals is only entered once, regardless of how many reciprocal pairs occur in the data set.

3.4.2 EXECUTE.COM

The main driver for this system is called EXECUTE.COM. It is written in DCL. EXECUTE.COM is designed to reduce the amount of work needed for the user to operate the system. EXECUTE.COM performs the following:

1. Asks the user questions.
2. Copies any files needed.
3. Assigns any output or input files needed.
4. Runs appropriate programs.
5. Submits batch processes.
6. Deletes all temporary files created by EXECUTE.COM.
7. Creates appropriate output files.
8. Prints output files.
9. Exits the system.

To use the system, all the user has to do is type @EXECUTE. After entering the system, the user is asked a few questions, similar to:

1. Do you want to submit this job as a batch process?

2. What is the name of the file you wish to analyze?

3. Is this the original data file or has it already been formatted?

4. What would you like to name the output file?

5. Would you like a printout of the output?

6. Do you want to save the output file?

7. Would you like to analyze another file?

3.4.3 EDFILE*

The module called EDFILE* is written in FORTRAN 77. Because several investigators may be gathering data for this system, the original data files cannot be expected to have all the same formats. Although each data file must contain the same information, the format may be slightly different. To keep the system flexible, the format of the input file to CRABS remains constant, and therefore, small programs must be created to transform the original data files into a file with the specified input format. These programs are called EDFILE1, EDFILE2, etc. The input to these programs will be the original data and the output will be the formatted data. The input to these programs is read in from a file called OLD.DAT and the output is written to a file called IN.DAT. The filename IN.DAT was chosen because it is the name of the input file for the analysis program. For the input format of OLD.DAT, see the comments in the particular EDFILE program.

3.4.4 SETBATCH

There is a module called SETBATCH written in FORTRAN 77. It is used to create a file called BATCH.DAT which contains the input to the CRABS program that the user normally types in at a terminal. The questions SETBATCH will ask are exactly the same as the questions asked by CRABS.

IN.DAT is used as the input file to CRABS. (This software was originally developed to analyze data from crabs that occupied bryozoan colonies, or heads, as refuge. Therefore "crabs" is used throughout to refer to the subjects, while "heads" refers to their refuges or potential locations.) The program needs to know the coordinates of each head and, if the head is occupied by a crab, the size, type and sex of the crab. The

volume of the head must also be present when a weighting function (described in the detailed design specification) is used. The coordinates are used to compute distances between crabs to find nearest-neighbor groups. Because the data may contain several species of crabs, the type of crab is used to selectively choose certain species for analysis. The sex of the crab is used to compute sex association statistics and to identify males when satellite eliminations are required. The size of the crab is used to determine size differences between neighboring crabs. The volume of the head is used by a weighting function when randomizing in the Monte Carlo trials.

Given the above requirements, a format of IN.DAT can be determined. Each line of the file contains 6 fields.

1. The X coordinate of the head.

2. The Y coordinate of the head.

3. The type of the crab.

4. The sex of the crab.

5. The size of the crab.

6. The volume of the head.

The type, sex and size fields will only be used when a crab is present on the given head. The last field will only be used if a weighting function is being performed on the data. If more than one crab is present on a given head, the coordinates will be repeated on the following line with the next crab's type and size. The following is the IN.DAT format:

```
FORMAT(1X,F4.2,1X,F4.2,1X,A2,A1,1X,F5.2,1X,I2)
           |       |      |   |     |      |
           |       |      |   |     |      -> CVOLUME
           |       |      |   |     ----> CSIZE
           |       |      |   -----> CSEX
           |       |      ---> CTYPE
           |       ----> XCOORD
           ----> YCOORD
```

1. XCOORD and YCOORD represent the X and Y coordinates respectively. The possible values are 0.00 --> 99.9.

2. CTYPE represents the type of the crab. Possible

values are "XX" where X can be any uppercase letter, indicating initials of the scientific name.

3. CSEX represents the sex of the crab. Possible values are "M", "F" and "O" denoting male, female and ovigerous female respectively.

4. CSIZE represents the size of the crab. Possible values are 00.00 --> 99.99.

5. CVOLUME represents the volume of the head in milliliters. Possible values are 0 --> 99.

4.0 DETAIL DESIGN

4.1 Required Structures

4.1.1 HEAD

An array is required to hold the X and Y coordinates of each head. The heads are numbered according to the order in which they are read. The array is called HEAD(I,J) with each element having a real value. It is a 400 by 2 array to accommodate a maximum of 400 coordinates. HEAD(I,1) is the X coordinate of the I'th head, and similarly, HEAD(I,2) is the Y coordinate.

4.1.2 Input Information Structures

The crabs, like the heads, are numbered according to the order in which they were read. Arrays are needed to store information about each crab such as location, sex and size.

4.1.2.1 ICRAB -- Because all possible locations are already stored in the HEAD array, another array, called ICRAB(I), is needed to be used as an index to the HEAD array. Its dimensions are 100 by 1 to accommodate 100 crabs. ICRAB contains integer values which are used as indices into the HEAD array. ICRAB(I) corresponds to the head where the I'th crab is located.

4.1.2.2 SEX -- The sex of each crab must also be stored in an array which we call SEX(I). Its dimensions are 100 by 1 where each element is a character. SEX(I) corresponds to the sex of the I'th crab.

4.1.2.3 SIZE -- The size of each crab is stored in an array called SIZE(I). Its dimensions are 100 by

1 where each element is a real value corresponding to the size of the I'th crab.

4.1.2 IMALE

The satellite male of the data set will need to be found from time to time. Because of this, an array called IMALE(I) is used as an index into ICRAB, SEX and SIZE. Its dimensions are 100 by 1 where each element is an integer value. The male crabs are numbered according to the order in which they are read. For example, IMALE(3)=8 means that the third male in the data set is the 8th crab in the data set. If the 3rd male's size was needed, SIZE(IMALE(3)) would need to be examined. Similarly, the 3rd male's X coordinate can be found using HEAD(ICRAB(IMALE(3)),1).

4.1.3 DIST

The CRABS program also deals with distances between crabs, therefore, an array called DIST(I,J) is used to hold this information. It contains real values and has dimensions of 100 by 100. DIST(I,J) is a real value corresponding to the distance between the I'th crab and the J'th crab. DIST(I,J) is calculated using the standard distance formula

```
DIST(I,J)=SQRT((X1-X2)**2+(Y1-Y2)**2)
```

where X1 and Y1 correspond to the I'th crab's X and Y coordinates respectively, and X2, Y2 correspond to the J'th crab's coordinates.

4.1.4 ORDER

Another array called ORDER(I,J) is needed to store the order in which the crabs appear to each other. It is a 100 by 100 array with each element being an integer. The ORDER array is created while the DIST array is being established. ORDER(I,1) contains the index of the crab closest to the I'th crab. By using the ORDER array along with the DIST array, the nearest neighbor groups are determined.

From the example in Appendix A, the following is a list of first and second nearest neighbors for each crab.

CRAB	1st NN	2nd NN
1	4	2,3
2	3	4,5
3	2	4,5
4	1,2,3,6	5,7
5	2,3,6	4
6	4,5,7	2,3
7	6	4

Note that there can be ties or more than one 1st nearest neighbor, etc., for any given crab, thus the term nearest neighbor group. To better understand the above, look at the graph in Appendix A showing the head locations.

Assume for the moment that DIST and ORDER arrays have already been established. The 1st N-N (Nearest Neighbor) group of the I'th crab is determined by first finding the index of the crab closest to the I'th crab. This is accomplished by examining ORDER(I,1). For example, ORDER(5,1)=2 means that the crab closest to the 5th crab is crab #2. If we find out how far crab #2 is from crab #5, we can determine if there are any more 1st nearest neighbors to crab #5. From the example in Appendix A, DIST(5,2)=1.0. Now we can search through the ORDER array, picking up indices, until we reach a crab whose distance to crab #5 is greater than 1.0. We can see from the example that crabs #2,3 and 6 are all a distance 1.0 away from crab #5 while crab #4 is 1.41 away. The second nearest neighbor group will start with crab #4 and 1.41 will be the new reference distance. This procedure is used as an efficient alternative to creating an array to hold the indices of crabs for each nearest neighbor group. Such an array would have to be a 100 by 5 by 50 array (100 crabs with 5 N-N groups each and assuming a maximum of 50 crabs in any one N-N group), would take large amounts of time to initialize and update especially if the user requests the program to run around 1000 times (that's 100*5*50*1000 times just to initialize the array).

4.1.5 VOLUME

This array has 500 integer elements. It will be used when the weighting function is used. VOLUME will be used to randomly pick out a head. Depending on the size of the head, the head number will be entered one or more times into the VOLUME array. This will simulate that some heads are more likely to be chosen than other heads. For example, if heads with volumes greater than or equal to 17 milliliters are 9 times

more likely to be chosen than those less than 17 milliliters, then the head number for each larger head would be entered into VOLUME 9 times. Now the head has 9 chances of being chosen instead of only one. You can think of the VOLUME array as a hat full of head numbers. Some of the numbers will be in the hat more than once giving that number a better chance of being picked. To pick a head, all you have to do is reach in and get a number out of the hat.

4.1.6 Statistics Structures

Different statistics are calculated depending upon the user's requests. Each statistic can be computed for up to the 5th nearest neighbor, therefore, arrays are used to store the different values. For example, the Kappa statistics in the program can be computed for each nearest neighbor group, and array called KAP(I) is needed with the dimension 5 by 1 where each element is a real value. Each value corresponds to the Kappa value for the I'th nearest neighbor group in the data set. Similarly, arrays A, B, C, D, P1, Q1, P2, Q2, PO, PE, OKAP, DIST_MEAN and SIZE_MEAN are used to store their corresponding statistics. Each is a 5 by 1 array, and all have real values.

4.1.7 Histogram Structures

Other arrays are needed which hold values for the histograms, i.e. frequency distributions, generated upon request by Monte Carlo trials. They are:

1. KAPPA(5,-100:100)
2. O_KAPPA(5,-100:100)
3. SISTO(5,300)
4. DISTO(5,500)

As expected, each array has the value 5 in its dimensions corresponding to the 5 possible nearest neighbor groups.

The -100:100 range on the KAPPA and O_KAPPA arrays was determined after it was found that an interval width of 0.01 was sufficient when producing a histogram for these statistics. Because the values of the Kappa and Overall Kappa fall in the range -1 to 1 inclusively, the range -100 to 100 was chosen where each unit represents a 0.01 interval. For example, if KAP(1) turned out to be 0.346, KAPPA(1,34) would be incremented by 1 to indicate that another Kappa value

fell within the interval (0.34,0.35]. If the Kappa value is positive, you round down after multiplying the value by 100. For negative values, you round up. For example, if the Kappa or Overall Kappa value was 0.392, the value 39 would be used as the index to the KAPPA or O_KAPPA array. If the value was -0.392 then -40 would be used representing the interval (-0.40,-0.390].

For the SISTO and DISTO arrays, the user is allowed to determine the interval width. Because the size differences between crabs are usually much greater than the distance differences, not as many intervals are needed for the SISTO histogram, thus the 300 intervals as compared to the 500 intervals for the DISTO array. The maximum size and distance difference allowed between any two crabs shall be determined by multiplying the appropriate interval width by the number of intervals in the corresponding array (300 or 500). For example, if the user wants the interval width on the size difference histogram to be 0.01, the maximum size difference between any two crabs is 3 units.

4.2 Global Variables

The following are the global variables used by the CRABS program.

4.2.1 HEADS

A global variable called HEADS is needed to store the number of heads entered, as the upper limit index in the HEAD array. It also is used in the randomizing function. The FORTRAN 77 random function generates a number between 0 and 1 exclusively. Therefore, by multiplying that value by HEADS, taking the integer part and then adding 1, you get a value between 1 and HEADS inclusively. For example, if the random number was .83 and HEADS was 20, then the randomizing function would return a value of INT(20*.83)+1=17 (function returns integer values). The indices in the HEAD array are used to randomly assign a given crab to a head . If for example, you needed to randomly select a head for the 3rd crab, a random value between 1 and HEADS inclusively would be generated, let's say 12, and then ICRAB(3) would be assigned 12 to indicate that the 3rd crab in the data set now resides on the 12th head.

4.2.2 CRABS

This variable is used to hold the total number of crabs in the data set being analyzed. To save time, CRABS is used as an upper limit index when searching

through the ICRAB, SEX, SIZE, DIST, and ORDER arrays. This variable also is used in many DO loops when information is being calculated for each crab.

4.2.3 MALES

This variable stores the number of male crabs being analyzed in the data set. It serves as an upper limit index to the IMALE array. From this variable and CRABS, the number of females in the data set can be determined (# of females=CRABS-MALES). When searching for the satellite male, MALES is used in the same way CRABS is used for the entire data set. When searching for the satellite males, the data set is temporarily reduced to all males, that is, the IMALE array is used as an index to the ICRAB, SEX, and SIZE arrays. If you want to find the first male crab's size, normally you would look at SIZE(1), but when searching for the satellite male, you use SIZE(IMALE(1)) to find the size of the first male crab.

4.2.4 Variables for storing entered data.

There are four variables used for storing the data being read in from IN.DAT. They are:

4.2.4.1 YCOORD — stores the Y coordinate of the head being read from the input file.

4.2.4.2 XCOORD — stores the X coordinate of the head being read.

4.2.4.3 CTYPE — is the crab's type or species if a crab is present on the current head.

4.2.4.4 CSEX — is the crab's sex if a crab is present on the current head.

4.2.4.5 CSIZE — is the crab's size if a crab is present on the current head.

4.2.4.6 CVOLUME — is the volume of the current head.

All of the above values are real except for CSEX which is a character and CTYPE which is a character string of length 2.

4.2.5 OLDX and OLDY

When reading in the data, it must be realized that more than one crab can reside on a particular head. If this is the case, the X and Y coordinates of

the head would be repeated on the following line. Consequently two variables, OLDX and OLDY, are needed to hold the last X and Y coordinates to determine if the head currently being examined has already had its X and Y coordinates entered.

4.2.6 Input parameters

Several variables hold information that is typed in by the user in response to various questions. They are :

4.2.6.1 TYPE — is the type of the crab the user wishes to analyze. The program will search through the data and only pull out information on the crabs with that type.*

4.2.6.2 WEIGHT — is a 'Y' or 'N' value indicating wether or not the weighting function will be used on the data.

4.2.6.3 ANALYSIS — is an integer value from 1 to 3 used to indicate which statistical values should be evaluated. To avoid uneccessary calculations, ANALYSIS will be checked by the program from time to time. If the user requests only Kappa statistics to be evaluated, the program will know not to waste time evaluating size and distance difference statistics.

4.2.6.4 DATA_SET — the name of the data set. It will be a character string of length 6.

4.2.6.5 NEIGHBORS — is an integer value between 1 and 5 inclusively. It specifies the number of nearest neighbor groups to be analyzed.

4.2.6.6 ELIMINATIONS — is an integer value between 0 and MALES inclusively. It holds the number of satellite males to be eliminated from the data set.

4.2.6.7 ITERATIONS — is an integer value greater than 0. It holds the number of times the data should be randomized except when ITERATIONS=1. In that case, the original data is analyzed and no randomization takes place.

4.2.6.8 DINT — is a real value greater than 0. It holds the interval width of the distance difference histogram.

4.2.6.9 SINT — is a real value greater than 0. It holds the interval width of the size difference

histogram.

4.2.6.10 SEED — is an integer value used in the randomizing function.

4.2.6.11 REPLACE — is a character (Y or N) used to indicate if the sampling of heads during the randomization of crabs onto heads should be performed with (Y) or without (N) replacement.

4.2.7 Miscellaneous

The remaining global variables are:

4.2.7.1 UPPERLIMIT — is an integer corresponding to the number of entries in the VOLUME array.

4.2.7.2 ISAT — is an integer used to hold the index of the satellite male crab. SIZE(ISAT) will be the size of the satellite male.

4.2.7.3 MALE — is an integer used to hold the index of the satellite male in the IMALE array. SIZE(IMALE(MALE)) is the size of the satellite male.

4.2.7.4 AMT — is a real value used to determine the histogram indices.

4.2.7.5 MM — is an integer used to index the histograms.

4.3 Subroutines

4.3.1 PRINT_HEAD

First, a subroutine called PRINT_HEAD prints out the header information, listing input parameters.

4.3.2 GENERATE_DATA

A subroutine called GENERATE_DATA executes the Monte Carlo trials, i.e. randomizes original data and generates new statistical values, if requested by the user. This subroutine simulates the randomizing of crabs onto heads with or without replacement. For each crab, a random number (NUM) is generated. If the weighting function is used, NUM will be in the range [1..UPPERLIMIT] otherwise it will be in the range [1..HEADS]. If the weighting function is being used, the head is chosen from the VOLUME array using NUM as the index. If the weighting function is not being used, NUM will be the number of the head chosen. If

the randomization is being performed without replacement, an array called IPICK must be checked to see if the head has already been chosen. If it has not been chosen before, the crab is simply assigned to the head just picked. This procedure continues until all of the crabs have been assigned a head.

4.3.3 FIND_SATELLITE

This subroutine finds the satellite male in the data set. A satellite pair is defined as being a pair of male crabs who have the greatest size difference where at least one crab is the first nearest neighbor of the other. If more than one pair have the same maximum size difference, then the pair with the smallest distance difference will be designated as the satellite pair. The satellite male of the data set is defined as the smallest crab of the satellite pair.

4.3.4 ELIMINATE

This subroutine uses the ISAT and MALE values as indices to eliminate the satellite male from the data set. In order for it to be eliminated from the data set, it must be deleted from ICRAB, SEX, SIZE and IMALE arrays. Instead of leaving the appropriate element in each array blank, entries below the proper index are all shifted up to fill in the empty space. The upper limit on the arrays, CRABS and MALES, are decremented by 1 giving each array a new upper bound. This procedure causes the last element in each array to be duplicated in the next to last position, but due to the new upper bounds, this duplicate last element will never be accessed.

4.3.5 FIND_NN

This subroutine determines the values for the ORDER and DIST arrays, using the ICRAB and HEAD arrays. CRABS is used as the upper limit in the do loops.

4.3.6 FIND_NEIGHBORS

This subroutine is the heart of the CRABS program. Given the DIST, ORDER, and SEX arrays, this subroutine determines the A, B, C and D values used in Kappa calculations for each nearest neighbor group. The DIST and ORDER arrays are used to distinguish from one nearest neighbor group to the next. The SEX array is used to determine the base-crab to nearest neighbor sex relationships required for evaluating the statistics.

4.3.7 KAPPA_OUTPUT

This subroutine prints out the statistics in tabular format for the original data only. For each nearest neighbor group analyzed, a table will be printed with the format seen in the example file called EXAMPLE.DAT.

4.3.8 KAPPA_HISTO

This subroutine prints out the histograms for the Kappa and Overall Kappa statistics for each nearest neighbor group analyzed. The format of each histogram can be seen in the file EXAMPLE.DAT.

4.3.9 PRINT_HEAD

This subroutine prints out the parameters used by the CRABS program. This will allow the user to run the program again using the same input.

4.3.10 PRINTOUT

This subroutine will print out the size and distance difference means for each nearest neighbor group being analyzed. The output will be in the form of a table with the values clearly labeled.

4.3.11 PRINT_HISTO

This subroutine prints out the size and distance difference histograms. The interval will be printed followed by appropriate values for each nearest neighbor group.

5.0 USER'S MANUAL

5.1 Invalid responses:

All invalid responses will result in the user being prompted again for the appropriate input.

5.2 Interrupt:

The user may stop execution of the program anytime by depressing the control key (CTRL) and the Y key at the same time.

5.3 Prompts and Appropriate responses:

The following prompts will appear on the terminal.

After each prompt, there will be a brief description of valid responses and examples if needed.

1. Prompt: What species do you wish to analyze?

   Answer: Enter in a 2 character string corresponding to the type of the subject. Responses must be in CAPITAL letters.

   Example: PS, MN, etc.

2. Prompt: Is there a weighting function?

   Answer: Enter in a 'Y' or 'N'

3. Prompt: What kind of analysis do you want to perform on the data?
   1=Kappa only 2=Size and Distance differences 3=All

   Answer: Enter in a number from 1 to 3 inclusively corresponding to the analysis you wish to be performed on the data.

4. Prompt: What is the name of the data set?(up to 6 characters)

   Answer: Just type in your data set name, but keep the length to 6 characters or less.

   Example: CLP20 correct
   SCHIZO correct
   ABCDEFG incorrect

5. Prompt: There are __ subjects in the _____ data set.
   How many nearest neighbors would you like to be analyzed?

   Answer: Enter in a number from 1 to 5 inclusively. If the number of subjects in the data set is less than 5, enter in a number from 1 to the number of subjects you have.

6. Prompt: There are __ males in the data set.
   How many satellite males do you wish to be eliminated?

   Answer: Enter in a number from 0 to the number of males there are in the data set inclusively. If the number of subjects minus the number of satellite males you wish to eliminate is less than the number

of nearest neighbors you are analyzing, you will be prompted again for the number of nearest neighbors to be analyzed.

7. Prompt: How many iterations would you like?

   Answer: If you enter in a '1', the program will use the original data set to calculate the observed statistics. If you enter in a number greater than one, the program will execute that number of Monte Carlo trials, randomizing the data each time. For an explanation of the above procedure, consult the Design Specifications.

NOTE: If you entered in a '1' for the above response, there will be no more prompts until the program is fully executed. If you entered a number greater than one, you will see the following prompts:

8. Prompt: What is the value of the seed to be used?

   Answer: Enter in a number that you want to be used as a seed or starting point by the randomizing function. Choose a positive prime number, the bigger the better.

   Example: 1237
   17
   13
   23

NOTE: The next two prompts will appear only if you answered prompt number 3 with a 2 or 3.

9. Prompt: What would you like the interval width of the distance difference histogram to be?

   Answer: Enter in a number, reasonable for your analysis, with at most 2 decimal places. If you enter in a 0, the default width of .01 will be used. The units appropriate to the distance differences will be the same as those used in defining the location coordinates. The upper limit of the distribution will be 500 multiplied by the interval width. .p -10,1,0

   Example: For an interval width of .01, the maximum distance difference would be 5.0. The units could be centimeters, meters, etc.

10. Prompt: What would you like the interval width of the size difference histogram to be?

    Answer: Enter in a number with at most 2 decimal places. If you enter in a 0, the default width of .05 will be used. The units will be the same as those used to record the size of subjects. The upper limit of the distribution will be 300 multiplied by the interval width.

    Example: If you want the histogram to go from 0 to 15 units, you will need an interval width of .05 since .05*300=15

11. Prompt: With replacement?

    Answer: Should the Monte Carlo trials employ sampling with replacement? Enter in a 'Y' for yes and an 'N' for no. (Please note that this response may be in upper or lower case letters.) If no, then sampling in the trials will be without replacement.

There will be no more prompts until the program is fully executed.

6.0 Interpretations

To illustrate the interpretation of SPAN output, the example in Appendix A was run with the following prompts and responses:

1. What species do you wish to analyze? PS
2. Is the weighting function being used? N
3. What kind of analysis do you want to do on this data? 1=Kappa only, 2=Size and Distance differences, 3=All. 3
4. What is the name of the data set? EXAMPLE
5. How many nearest neighbors would you like to be analyzed? 3
6. How many satellite males do you wish to eliminate? 0
7. How many iterations would you like? 1 and then 100
8. What would you like the interval width of the distance difference histogram to be? 0.01

9. What would you like the interval width of the size difference histogram to be? 0.05

10. What is the value of the seed to be used? 1237

11. With replacement? N

6.1 Output for the Observed Values of Statistics

When prompt 7 was answered with 1, the observed values for mean size difference, mean distance difference, Kappa(males) and Kappa(males-females) were calculated. Analysis to the 3rd nearest neighbor was requested in prompt 5, but this data set lacked some 3rd nearest neighbors. Therefore a message to that effect was printed and values were given for the first and second nearest neighbors. Normally this problem would occur only with small data sets. The output is shown below.

The number of nearest neighbors being analyzed was reduced to 2 because the 4th crab does not have a 3rd nearest neighbor.

Nearest Neighbors	Mean Size Difference	Mean Dist. Difference
1st	1.444	0.889
2nd	1.620	1.249

1st Nearest Neighbor

Kappa Statistics
Observed proportion (Po)= 0.8214
Expected proportion (Pe)= 0.4949
Kappa (males/females) = 0.6465
Kappa (males) =-0.6598

Proportions

Base	M	F	
M	0.1429	0.4286	0.5714
F	0.3929	0.0357	0.4286
	0.5357	0.4643	1.0000

2nd Nearest Neighbor

Kappa Statistics

Observed proportion (Po) =	0.5000
Expected proportion (Pe) =	0.5204
Kappa (males/females) =	-0.0426
Kappa (males) =	0.0392

Proportions

Base	M	F	
M	0.2143	0.3571	0.5714
F	0.1429	0.2857	0.4286
	0.3571	0.6429	1.0000

6.2 Output for the Monte Carlo Trials

When prompt 7 was answered with 100, the frequency distribution for the above statistics were generated based on 100 trials. Normally, many more trials would be used. With the conditions specified above, the following tables were generated in which the tabled values are frequencies of occurrence for each interval.

The number of nearest neighbors being analyzed was reduced to 2 because the 4th crab does not have a 3rd nearest neighbor.

Nearest Neighbor #1

Kappa Histogram:

	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
-1.0	0	0	0	1	0	0	0	0	0	0
-0.9	0	0	0	0	0	0	0	0	0	0
-0.8	0	0	0	0	0	3	0	0	0	0
-0.7	0	0	3	0	0	0	6	0	0	0
-0.6	1	0	0	2	3	0	0	2	0	0
-0.5	1	4	0	1	0	0	0	0	0	0
-0.4	0	0	0	0	0	0	0	1	0	1
-0.3	0	4	2	2	0	0	0	1	0	0
-0.2	0	3	0	1	0	1	0	5	0	0
-0.1	4	0	0	0	0	5	0	0	0	0
0.0	4	2	1	0	1	0	0	0	0	1
0.1	0	2	4	2	1	0	0	0	0	2
0.2	0	1	0	0	0	3	2	0	2	0
0.3	0	1	3	0	3	0	3	0	1	0
0.4	0	0	0	0	0	0	0	0	0	0
0.5	0	1	0	0	0	0	1	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0
0.7	1	0	0	0	0	1	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0

OKappa Histogram:

	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
-1.0	0	0	0	0	0	0	0	0	0	0
-0.9	0	0	0	0	0	0	0	0	0	0
-0.8	0	0	0	0	0	0	1	0	0	0
-0.7	0	1	0	0	0	0	0	0	0	0
-0.6	0	0	0	0	1	0	0	0	0	0
-0.5	1	0	0	0	0	0	0	0	0	0
-0.4	0	1	0	0	0	3	0	3	0	4
-0.3	0	2	0	3	2	0	0	0	0	0
-0.2	2	1	0	0	1	0	0	4	2	2
-0.1	1	0	0	0	0	1	0	1	2	1
0.0	3	0	0	3	2	0	0	0	0	4
0.1	0	0	5	1	0	1	0	3	0	0
0.2	0	1	2	0	0	1	0	0	5	1
0.3	1	0	0	0	0	0	0	0	0	0
0.4	1	0	4	0	0	0	1	0	0	0
0.5	2	1	0	1	1	0	3	0	6	0
0.6	0	0	0	3	0	0	0	0	0	0
0.7	0	0	3	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0

Nearest Neighbor #2

Kappa Histogram:

	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
-1.0	0	0	0	2	0	0	0	0	0	0
-0.9	0	0	1	0	0	0	0	0	0	0
-0.8	0	0	0	0	0	0	0	0	0	0
-0.7	0	0	3	0	0	0	5	0	0	0
-0.6	0	0	0	0	0	0	1	2	0	2
-0.5	0	0	0	0	5	0	0	3	0	3
-0.4	2	0	0	0	5	0	0	0	11	0
-0.3	0	0	2	0	0	0	0	0	1	1
-0.2	0	0	15	0	0	0	0	0	0	0
-0.1	0	0	0	0	0	7	0	1	0	0
0.0	10	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	8	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0
0.4	0	3	0	1	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0
0.6	0	0	5	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0

OKappa Histogram:

	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
-1.0	0	0	0	1	0	0	0	0	0	0
-0.9	0	0	0	0	0	0	0	0	0	0
-0.8	0	0	0	0	0	0	0	0	0	0
-0.7	0	0	0	0	0	0	5	0	0	0
-0.6	0	0	0	0	0	0	0	0	0	0
-0.5	0	0	0	0	1	0	0	0	0	3
-0.4	0	0	0	0	0	0	0	0	0	0
-0.3	0	0	0	0	0	0	0	0	0	0
-0.2	0	0	0	0	8	0	0	0	0	0
-0.1	0	0	0	0	0	0	0	0	0	0
0.0	10	1	0	7	0	0	0	0	0	0
0.1	0	0	0	0	0	0	1	15	0	1
0.2	0	0	2	0	0	0	11	0	0	0
0.3	5	0	0	2	0	0	0	0	0	0
0.4	0	3	0	0	0	0	2	3	0	0
0.5	0	0	0	5	0	0	0	0	7	0
0.6	0	1	0	3	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0
0.9	0	0	1	0	0	0	0	0	0	0

SIZE DIFFERENCE HISTOGRAM

NN			1	2
(	0.850 -	0.900]	0	2
(	1.000 -	1.050]	1	3
(	1.050 -	1.100]	2	2
(	1.100 -	1.150]	2	2
(	1.150 -	1.200]	1	3
(	1.200 -	1.250]	5	6
(	1.250 -	1.300]	3	4
(	1.300 -	1.350]	6	8
(	1.350 -	1.400]	7	7
(	1.400 -	1.450]	6	7
(	1.450 -	1.500]	11	4
(	1.500 -	1.550]	9	8
(	1.550 -	1.600]	4	2
(	1.600 -	1.650]	9	5
(	1.650 -	1.700]	5	1
(	1.700 -	1.750]	5	10
(	1.750 -	1.800]	5	5
(	1.800 -	1.850]	7	4
(	1.850 -	1.900]	1	2
(	1.900 -	1.950]	6	4
(	1.950 -	2.000]	0	2
(	2.000 -	2.050]	1	3
(	2.050 -	2.100]	2	1
(	2.100 -	2.150]	1	4
(	2.150 -	2.200]	1	0
(	2.200 -	2.250]	0	1

DISTANCE DIFFERENCE HISTOGRAM

NN			1	2
(	0.990 -	1.000]	90	0
(	1.050 -	1.060]	10	0
(	1.410 -	1.420]	0	67
(	1.530 -	1.540]	0	3
(	1.580 -	1.590]	0	10
(	1.600 -	1.610]	0	11
(	1.800 -	1.810]	0	9

6.3 Comparing Observed Values to Monte Carlo Trials

The probability of obtaining by chance alone a value equal to or less than (or greater than) an observed value can be figured from the proportion of Monte Carlo trials yielding values in the tail of the corresponding distribution (see section 6.2). For example, the observed mean size difference for the first nearest neighbors in section 6.1 was 1.444. The interval containing that

value in the corresponding table (section 6.2) is found by inspection, and then the frequency within that interval is summed with frequencies for all lesser intervals. By chance alone 32 out of 100 values would be less than or equal to the observed value, i.e. P=.32 and the null hypothesis of no difference would be retained. The same procedures would apply to the remaining statistics, however, the Kappa output may be a bit more confusing to interpret.

The output for the observed Kappa (see section 6.1) contains the values for Kappa(males) and Kappa(males-females), as well as the proportions giving rise to those statistics. In most cases, testing one of these statistics will be sufficient, while the proportions can aid interpretation. Kappa values can either be used as data points in an experimental design in which treatment effects are tested, or individual Kappa's can be compared to their own distributions generated by Monte Carlo trials.

The Monte Carlo output in section 6.2 was designed to save space and needs some explanation. The row values along the vertical axis indicate larger divisions of the distributions than do the column values along the horizontal axis. To find the cell in the table containing frequencies for a given Kappa value, first find the row and column value which when summed comes closest to the given value. For example, Kappa(males) for 1st NN is -0.6598 (section 6.1). In the corresponding Kappa Histogram (section 6.2), -0.7 plus .04 equals -0.66, thus the cell at the intersection of that row and column is our entry point for the table. Now, sum the cell frequencies for the -0.66 cell and all smaller (more negative) values, i.e. sum to the left and up. In this case, 7 out of 100 Monte Carlo trials produced values equal to or less than the observed value; P=.07, so again, the null hypothesis would be retained.

In any event the testing and interpretation of SPAN output must be based on the understanding of what is happening conceptually in the program. The output will always require some manipulation by the investigator to be meaningful. Either observed values are compared to distributions from Monte Carlo trials, or observed values are used as data for more standard statistical tests, or both are approaches are used.

APPENDIX A **EXAMPLE**

File: IN.DAT

```
1.00 1.00
1.00 2.00 PSM 12.10
1.00 3.00
2.00 1.00 PSF  9.80
2.00 1.00 PSM 10.30
2.00 2.00 PSF 11.10
2.00 3.00
3.00 1.00 PSM  8.90
3.00 2.00 PSM 12.60
3.00 3.00 PSF 10.70
```

	SIZE
1	12.10
2	9.80
3	10.30
4	11.10
5	8.90
6	12.60
7	10.70

	HEAD	
1	1.00	1.00
2	1.00	2.00
3	1.00	3.00
4	2.00	1.00
5	2.00	2.00
6	2.00	3.00
7	3.00	1.00
8	3.00	2.00
9	3.00	3.00

	ICRAB
1	2
2	4
3	4
4	5
5	7
6	8
7	9

	IMALE
1	1
2	3
3	5
4	6

	SEX
1	M
2	F
3	M
4	F
5	M
6	M
7	F

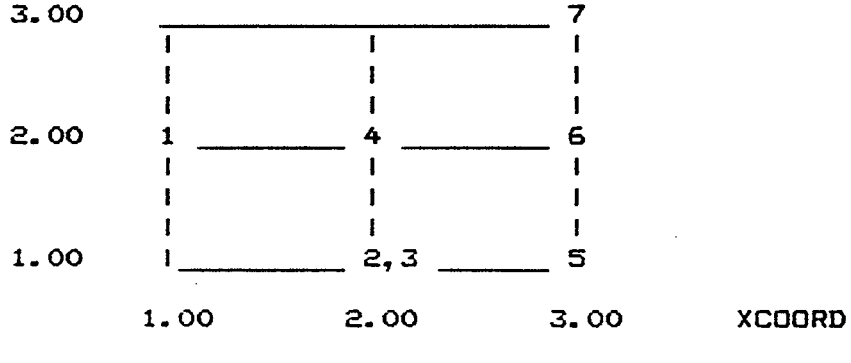
DIST

	1	2	3	4	5	6	7
1	0.00	1.41	1.41	1.00	2.24	2.00	2.24
2	1.41	0.00	0.00	1.00	1.00	1.41	2.24
3	1.41	0.00	0.00	1.00	1.00	1.41	2.24
4	1.00	1.00	1.00	0.00	1.41	1.00	1.41
5	2.24	1.00	1.00	1.41	0.00	1.00	2.00
6	2.00	1.41	1.41	1.00	1.00	0.00	1.00
7	2.24	2.24	2.24	1.41	2.00	1.00	0.00

ORDER

Neighbor Rank

crab #	1	2	3	4	5	6
1	4	2	3	6	5	7
2	3	4	5	1	6	7
3	2	4	5	1	6	7
4	1	2	3	6	5	7
5	2	3	6	4	7	1
6	4	5	7	2	3	1
7	6	4	5	1	2	3



APPENDIX A — EXAMPLE

For 1st nearest neighbors statistics

A = 1/3 + 1/3 + 1/3 = 1
B = 1 + 1 + 1/3 + 1/3 + 1/3 = 3
C = 1 + 1/4 + 1/4 + 1/4 + 1 = 11/4
D = 1/4

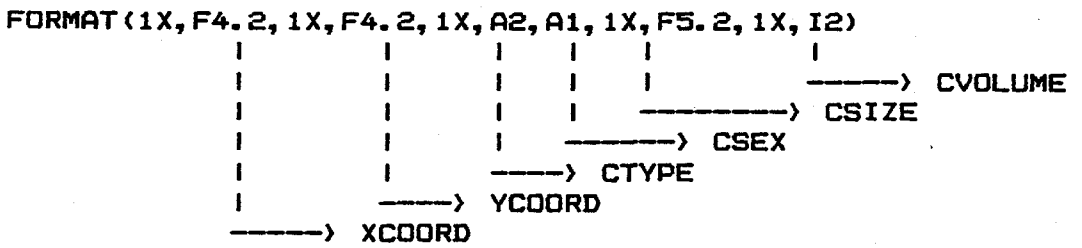
a = 1/7 = 0.1429
b = 3/7 = 0.4286
c = 11/28 = 0.3929
d = 13/28 = 0.0357

P1 = 4/7 = 0.5714
Q1 = 3/7 = 0.4286
P2 = 15/28 = 0.5357
Q2 = 1/28 = 0.4643

PO = 0.8214
PE = 0.4949

KAP = 0.6465
OKAP= -0.6598

Input File Format:



Each line contains information on a particular refuge. (Originally, refuges were bryozoan heads, or colonies. Therefore, refuge variables are continually labeled "head" or "heads" in the program.) If no crab is present on the refuge, there will be no CTYPE, CSEX or CSIZE values. If no weighting function is being used, CVOLUME will not be used.
XCOORD and **YCOORD** represent the X and Y coordinates respectively of a refuge's location. 0.00 --> 9.99
CTYPE is the type of the crab. XX where X can be any uppercase letter.
CSEX is the sex of the crab. M, F or O
CSIZE is the size of the crab. 00.00 --> 99.99
CVOLUME is the volume of the head. 0 --> 99

Arrays:
For a better understanding of all the arrays listed and described below, please see the example in appendix A.

1. HEAD(I,J)..Size: 400 X 2
   Type: REAL
   Possible Values: 0.00 --> 9.99

   This array contains the X and Y coordinates corresponding to each head. HEAD(3,1) is the X coordinate for the 3rd head.

2. ICRAB(I)...Size: 100
   Type: INTEGER
   Possible Values: 1 --> # of heads

   This array serves as an index used to locate the coordinates of the I'th crab which are found in the HEAD array.

   Example: ICRAB(4)=10

   Meaning: The 4th crab is located on the 10th head and

has as its X and Y coordinates, HEAD(ICRAB(4),1) and HEAD(ICRAB(4),2) respectively.

3. SIZE(I)....Size: 100
   Type: REAL
   Possible Values: 00.00 --> 99.99

   This array contains the size of the I'th crab. SIZE(3) will contain the size of the 3rd crab.

4. SEX(I).....Size: 100
   Type: String of length 3
   Possible Values: __M,__F (where __ denotes the species of crab)

   This array contains the sex of the I'th crab. SEX(3) will contain the sex of the 3rd crab.

5. IMALE(I)...Size: 100
   Type: INTEGER
   Possible Values: 1 --> # of crabs.

   This array serves as an index used to locate the size and coordinates of the male crabs.

   Example: IMALE(3)=9

   Meaning: The 3rd male crab is the 9th crab in the data set. SIZE(IMALE(3)) will give you the size of the 3rd male crab. HEAD(ICRAB(IMALE(3)),1) gives you the X coordinate of the 3rd male crab.

6. DIST(I,J)..Size: 100 X 100
   Type: REAL
   Possible Values: any real >= 0

   This array contains values corresponding to the distance between the I'th crab and the J'th crab.

7. ORDER(I,J).Size: 100 X 100
   Type: INTEGER
   Possible Values: 1 --> # of crabs

   This array contains indices of crabs according to the distance they are away from the I'th crab. Distances are in ascending order. ORDER(I,1) is the index of the crab closest to the I'th crab.

Example: ORDER(4,1)=3

Meaning: The 3rd crab is the closest to the 4th crab.

8. VOLUME(I)..Size 1500 X 1
Type: INTEGER
Possible values: 1 --> # of heads

This array is used to pick a head at random when the weighting function is being used. If a head is 9 times more likely to be picked than the others, the head number will appear in the VOLUME array 9 times instead of once.

```
C       This program takes in data from a file called OLD.DAT
C       and creates a file called IN.DAT. The file OLD.DAT is
C       the original data file. The program CRABS.EXE must have
C       it's input formatted in a certain way. This program will
C       transform the original data into the format required for
C       CRABS.EXE. This enables the format on the original data
C       file to be more flexible.
C
C       The data file is read in one line at a time. An '*' in
C       the first field indicates the end of the file.
C
C       Variable dictionary:
C               ID      -->     The crab's identification number.
C               T(1)    -->     The type of the first crab.
C               T(2)    -->     The type of the second crab.
C               T(3)    -->     The type of the third crab.
C               CW(1)   -->     The size of the first crab.
C               CW(2)   -->     The size of the second crab.
C               CW(3)   -->     The size of the third crab.
C               VOL     -->     The volume of the head.
C               X       -->     The x coordinate of the head.
C               Y       -->     The y coordinate of the head.
C               FLAG    -->     Indicates if there are any crabs
C                               occupying a certain head.
C
C       Input file format:
C
C       --IDX--Y------T1----CW1----T2----CW2----T3----CW3---VOL
C
C       The '-'s indicate spaces.
C
C       T1,T2,T3 are character strings of length 3.
C       CW1,CW2,CW3 are real numbers in the format xxx.x (000.0 -
999.9)
C       ID is a character string of length 6.
C       X,Y are real numbers in the format xx.xx (00.00 - 99.99)
C       VOL is an integer (0 - 99)
C
C       Output file format:
C
C       -X-Y-TYPE-SIZE-VOL
C
C       X,Y are real numbers in the format xx.xx (00.00 - 99.99)
C       TYPE if a character string of length 3.
C       SIZE  is a  real number  in the  format xxx.xx  (000.00 -
999.99)
C       VOL is an integer (0 - 99)
C
C
        CHARACTER*6 ID
        CHARACTER*3 T(3)
```

```
      DIMENSION CW(3)
      INTEGER VOL,FLAG
C
C     Open each file
C
      OPEN(UNIT=1,FILE='OLD.DAT',STATUS='OLD')
      OPEN(UNIT=2,FILE='IN.DAT',STATUS='NEW')
C
C     Read in a line of data
C
1
READ(1,2,END=20)ID,X,Y,T(1),CW(1),T(2),CW(2),T(3),CW(3),VOL
2     FORMAT(2X,A6,F4.2,2X,F4.2,3(4X,A3,4X,F4.1),2X,I2)
      FLAG=0
C
C     If the first character in the ID is an '*', then we
C     have reached the end of the file.
C
      IF(ID(1:1).EQ.'*')GOTO 20
C
C     The following loop checks the type of each of the three
C     possible crabs on each data line. If the type is not an
C     '*', indicating that a crab is resident on the current
C     head, the x and y coordinates,sex and size of the crab
C     will be written to the output file along with the volume
C     of the head.
C
      DO 5 I=1,3
            IF(T(I).NE.'*  ')THEN
                  FLAG=1
                  WRITE(2,10)X,Y,T(I),CW(I),VOL
            ENDIF
5     CONTINUE
10    FORMAT(1X,F4.2,1X,F4.2,1X,A3,1X,F5.2,1X,I2)
C
C     If there aren't any crabs in the current data  line, just
print
C     out the coordinates and the volume of the head.
C
      IF(FLAG.EQ.0)WRITE(2,10)X,Y,T(3),CW(3),VOL
      GOTO1
20    END
```

```
C    This  program is  used to  create a file which will hold the
input
C data for CRABS. This program  will only  be used  when CRABS is
being
C executed in a batch process. The data the user would type in at
the
C terminal will be saved in a file called BATCH.DAT. This program

C simply  asks the user the same exact questions that CRABS does,

C therefore, most of its code was taken directly from CRABS.FOR.
C
      INTEGER ELIMINATIONS,CRABS,HEADS,ANALYSIS
      INTEGER*4 SEED
      CHARACTER*1 REPLACE,CSEX,WEIGHT
      CHARACTER*6 DATA_SET
      CHARACTER*2 CTYPE,TYPE
C
C Open the input and output files
C
      OPEN(UNIT=1,FILE='IN',STATUS='OLD')
      OPEN(UNIT=2,FILE='BATCH',STATUS='NEW')
C
C Initialize the counting variables
C
      MALES=0
      CRABS=0
      HEADS=0
C
C Ask the user for the type of crab he or she wishes to analyze
C
      PRINT*,'What specise do you wish to alalyze?'
      READ(*,2)TYPE
C
C Ask the user if the weighting function will be used.
C
1     PRINT*,'Is the weighting function being used?'
      READ(*,7)WEIGHT
      IF(WEIGHT.EQ.'y')WEIGHT='Y'
      IF(WEIGHT.EQ.'n')WEIGHT='N'
      IF(WEIGHT.NE.'Y'.AND.WEIGHT.NE.'N')GOTO 1
2     FORMAT(A2)
C
C Read in a line of data. Goto 22 if end of file.
C
3     READ(1,4,END=15)XCOORD,YCOORD,CTYPE,CSEX,CSIZE
4     FORMAT(1X,F4.2,1X,F4.2,1X,A2,A1,1X,F5.2)
C
C Determine if the type of the subject is one of the ones we want
```

```
to analyze
C
5       IF(CTYPE.EQ.TYPE)THEN
C
C Increment CRABS
C
                CRABS=CRABS+1
C
C If the subject is a male
C
                IF(CSEX.EQ.'M')MALES=MALES+1
        ENDIF
C
C Go back and read in the next line of data
C
        GOTO 3
C
C These are the formats for the interactive user input and output

C
7       FORMAT(A1)
8       FORMAT(I4)
9       FORMAT(I5)
10      FORMAT(I1)
11      FORMAT(A6)
C
C The following PRINT statements ask the user the required
information
C
15        PRINT*,'What kind of analysis do you want to do on this
data?'
        PRINT*,'1=Kappa only,  2=Size  and  Distance differences,
3=All'
        READ(*,10)ANALYSIS
        IF(ANALYSIS.GT.3.OR.ANALYSIS.LT.1)GOTO 15
22         PRINT*,'What is  the name  of the  data set?  (up to 6
characters)'
        READ(*,11)DATA_SET
23      PRINT 24,CRABS,DATA_SET
24      FORMAT(' There are ',I2,' crabs in the ',A,' data set.')
        PRINT*,'How many nearest neighbors  would you  like to be
analyzed?'
        READ(*,10)NEIGHBORS
C
C Check for invalid responce
C
        IF(NEIGHBORS.GT.5.OR.NEIGHBORS.LT.1)GOTO 23
25      PRINT 26,MALES
26      FORMAT(' There are ',I2,' males in the data set.')
        PRINT*,'How  many  satelite  males  do  you  wish  to  be
eliminated?'
```

```
        READ(*,10)ELIMINATIONS
C
C Check for invalid responce
C
        IF(ELIMINATIONS.LT.0.OR.ELIMINATIONS.GT.MALES)GOTO 25
        IF(CRABS-ELIMINATIONS.LT.NEIGHBORS)THEN
          PRINT*,'After eliminating the satelite males, there are
not'
          PRINT*,'enough crabs  to analyze  the number of nearest
neighbors'
          PRINT*,'you requested.'
          GOTO 23
        ENDIF
27      PRINT*,'How many iterations would you like?'
        READ(*,8)ITERATIONS
C
C Check for invalid responce
C
        IF(ITERATIONS.LT.1)GOTO 27
C
C If ITERATIONS = 1 then analysis is on the original data and the
following
C questions will be skipped.
C

WRITE(2,100)TYPE,ANALYSIS,DATA_SET,NEIGHBORS,ELIMINATIONS,
     $               ITERATIONS
        IF(ITERATIONS.NE.1)THEN
C
C If  size and  distance difference  histograms are to be printed
out
C
           IF(ANALYSIS.GT.1)THEN
                PRINT*,'What would you like the interval width of
the'
                PRINT*,'distance difference histogram to be?'
                READ(*,29)DINT
                IF(DINT.EQ.0.0)DINT=0.01
                PRINT*,'What would you like the interval width of
the'
                PRINT*,'size difference histogram to be?'
                READ(*,29)SINT
                IF(SINT.EQ.0.0)SINT=0.05
29              FORMAT(F3.2)
                WRITE(2,29)DINT
                WRITE(2,29)SINT
           ENDIF
                PRINT*,'What is  the  value  of  the  seed  to be
used?'
                READ(*,9)SEED
                WRITE(2,9)SEED
```

```
30                PRINT*,'With replacement?'
                  READ(*,7)REPLACE
C
C Change lower case responce to upper case
C
                  IF(REPLACE.EQ.'n')REPLACE='N'
                  IF(REPLACE.EQ.'y')REPLACE='Y'
C
C Check for invalid responce
C
                  IF(REPLACE.NE.'Y'.AND.REPLACE.NE.'N')GOTO 30
                  WRITE(2,7)REPLACE
          ENDIF
100       FORMAT(A2/I1/A/I1/I2/I4)
          END
```

```
C Variable declarations
      REAL KAP(5),MMS(5),MMD(5),FFS(5),FFD(5)
      INTEGER ELIMINATIONS,CRABS,HEADS,SISTO(5,300),
     $ FFSISTO(5,300),DISTO(5,500)
      INTEGER O_KAPPA(5,-100:100),ORDER(100,100),
     $ ANALYSIS,CVOLUME,VOLUME(1500),UPPERLIMIT
      INTEGER*4 SEED
      CHARACTER*1 REPLACE,SEX(100),CSEX,WEIGHT,TEST,MALEONLY
     $ ,FEMALEONLY
      DIMENSION IMALE(100),MMDISTO(5,500),MMSISTO(5,300)
      CHARACTER*10 DATA_SET
      CHARACTER*2 CTYPE,TYPE
      INTEGER FFDISTO(5,500)
      COMMON /VOL/VOLUME
      COMMON /COM1/ORDER,DIST(100,100)
      COMMON /COM2/SEX,SIZE(100)
      COMMON /COM3/ICRAB(100),HEAD(400,2)
      COMMON /KAPPAS1/A(5),B(5),C(5),D(5)
      COMMON /KAPPAS2/P1(5),Q1(5),P2(5),Q2(5),PE(5),PO(5),
     $                KAP,OKAP(5)
      COMMON /KAPPAS3/KAPPA(5,-100:100),O_KAPPA
      COMMON /MEANS1/SIZE_MEAN(5),DIST_MEAN(5)
      COMMON /MEANS2/SISTO,DISTO
      COMMON /MEANS3/MMS,MMD,FFS,FFD
C Open the input and output files

      OPEN(UNIT=1,FILE='IN',STATUS='OLD')
      OPEN(UNIT=2,FILE='OUT',STATUS='NEW')

C Initialize the counting variables

      MALES=0
      CRABS=0
      HEADS=0
      TEST='N'

      PRINT*,'Will you be testing input data? (Y or N)'
      READ(*,7)TEST


C Ask the user for the type of crab he or she wishes to analyze

      PRINT*,'What species do you wish to analyze?'
      READ(*,1)TYPE
      IF(TEST.EQ.'Y')PRINT*,TYPE
1     FORMAT(A2)
      MALEONLY='N'
      FEMALEONLY='N'
      PRINT*,'Would you  like only the males to be analyzed? (Y
```

```
or N)'
        READ(*,7)MALEONLY
        IF(MALEONLY.EQ.'Y')GOTO 2
        PRINT*,'Would you like only  the females  to be analyzed?
(Y or N)'
        READ(*,7)FEMALEONLY

C  Ask  the  user  if  he  or she would like to use the weighting
function.

2       PRINT*,'Is the weighting function being used?'
        READ(*,7)WEIGHT
        IF(TEST.EQ.'Y')PRINT*,WEIGHT

C Change lower case to upper case

        IF(WEIGHT.EQ.'y')WEIGHT='Y'
        IF(WEIGHT.EQ.'n')WEIGHT='N'

C Check for invalid response

        IF(WEIGHT.NE.'Y'.AND.WEIGHT.NE.'N')GOTO 2

C OLDX  and  OLDY  are  used  to  determine  if  HEADS  should be
incremented

        OLDX=-9.99
        OLDY=-9.99
        UPPERLIMIT=0

C Read in a line of data. Goto 22 if end of file.

3       READ(1,4,END=15)XCOORD,YCOORD,CTYPE,CSEX,CSIZE,CVOLUME
4       FORMAT(1X,F4.2,1X,F4.2,1X,A2,A1,1X,F5.2,1X,I2)

C If still getting information on the same head goto 6

        IF(OLDX.EQ.XCOORD.AND.OLDY.EQ.YCOORD)GOTO 6

C  Increment  HEADS  and  initialize  the  HEAD  array  with  the
coordinates

        HEADS=HEADS+1
        HEAD(HEADS,1)=XCOORD
        HEAD(HEADS,2)=YCOORD

C If the weighting function is being used,  see if  the volume of
the
C head  is greater  than 16  ml. If  it is, then the current head
will
C be 9 times more likely to be picked then others. To accomplish
```

```
C this during randomization, the number of the head is  placed in
the
C VOLUME array 9 times instead of once.

        IF(WEIGHT.EQ.'Y')THEN

C Determine  the number  of times  the head number will be put in
the array

                K=1
                IF(CVOLUME.GE.17)K=9

C Initialize the VOLUME array and update the UPPERLIMIT

                DO 5 J=1,K
                        UPPERLIMIT=UPPERLIMIT+1
                        VOLUME(UPPERLIMIT)=HEADS
5               CONTINUE
        ENDIF

C Determine if the sex of the crab is one of the ones we  want to
analyze

6       IF(CTYPE.EQ.TYPE)THEN
           MOK=0
           IF((MALEONLY.EQ.'Y').AND.(CSEX.EQ.'M'))MOK=1
           IF((FEMALEONLY.EQ.'Y').AND.((CSEX.EQ.'F').OR.(CSEX.EQ.

     $     'O')))MOK=1
           IF((MALEONLY.EQ.'N').AND.(FEMALEONLY.EQ.'N'))MOK=1
C Increment CRABS and initialize the appropriate arrays
           IF(MOK.EQ.1)THEN

                CRABS=CRABS+1

C If  the sex  of the  crab is  O (Ovigerous),  change it to F to
indicate
C that the cab is a female. This program only  cares if  the crab
is a
C male or female.

                IF(CSEX.EQ.'O')CSEX='F'
                SEX(CRABS)=CSEX
                SIZE(CRABS)=CSIZE
                ICRAB(CRABS)=HEADS

C If the crab is a male

                IF(CSEX.EQ.'M')THEN

C Increment MALES and initialize the IMALE array
```

```
                        MALES=MALES+1
                        IMALE(MALES)=CRABS
                    ENDIF
                ENDIF
            ENDIF

C Update the values

            OLDX=XCOORD
            OLDY=YCOORD

C Go back and read in the next line of data

            GOTO 3

C These are the formats for the interactive user input and output


7           FORMAT(A1)
8           FORMAT(I4)
9           FORMAT(I5)
10          FORMAT(I1)
11          FORMAT(A10)

C The following PRINT statements ask the user the required
information.
C If no crabs were found with the required type, print a message.


15          CALL PRINTSET(CRABS,SEX,ICRAB,HEAD,SIZE)

            IF(CRABS.EQ.0)THEN
                    PRINT*,'There are no crabs in the dataset.'
                    STOP
            ENDIF

16          PRINT*,'What kind of analysis do you want to do on this
data?'
            PRINT*,'1=Kappa only, 2=Size and Distance differences,
3=All'
            READ(*,10)ANALYSIS
            IF(TEST.EQ.'Y')PRINT*,ANALYSIS
            IF(ANALYSIS.GT.3.OR.ANALYSIS.LT.1)GOTO 16
22            PRINT*,'What is the name of the data set? (up to 10
characters)'
            READ(*,11)DATA_SET
            IF(TEST.EQ.'Y')PRINT*,DATA_SET
23          PRINT 24,CRABS,DATA_SET
24          FORMAT(' There are ',I2,' crabs in the ',A,' data set.')
            PRINT*,'How many nearest neighbors would you like to be
```

```
analyzed?'
        READ(*,10)NEIGHBORS
        IF(TEST.EQ.'Y')PRINT*,NEIGHBORS

C Check for invalid responce

        IF(NEIGHBORS.GT.5.OR.NEIGHBORS.LT.1)GOTO 23
25      PRINT 26,MALES
26      FORMAT(' There are ',I2,' males in the data set.')
        PRINT*,'How many satellite males do you wish to be
eliminated?'
        READ(*,10)ELIMINATIONS
        IF(TEST.EQ.'Y')PRINT*,ELIMINATIONS

C Check for invalid responce

        IF(ELIMINATIONS.LT.0.OR.ELIMINATIONS.GT.MALES)GOTO 25
        IF(CRABS-ELIMINATIONS.LT.NEIGHBORS)THEN
          PRINT*,'After eliminating the satellite males, there
are not'
          PRINT*,'enough crabs to analyze the number of nearest
neighbors'
          PRINT*,'you requested.'
          GOTO 23
        ENDIF
27      PRINT*,'How many iterations would you like?'
        READ(*,8)ITERATIONS
        IF(TEST.EQ.'Y')PRINT*,ITERATIONS

C Check for invalid responce

        IF(ITERATIONS.LT.1)GOTO 27

C If ITERATIONS = 1 then analysis is on the original data and the
following
C questions will be skipped.

        IF(ITERATIONS.NE.1)THEN

C If size and distance difference histograms are to be printed
out

            IF(ANALYSIS.GT.1)THEN
                PRINT*,'What would you like the interval width of
the'
                PRINT*,'distance difference histogram to be?'
                READ(*,29)DINT
                IF(DINT.EQ.0.0)DINT=0.010
                IF(TEST.EQ.'Y')PRINT*,DINT
                PRINT*,'What would you like the interval width of
the'
```

```
                PRINT*,'size difference histogram to be?'
                READ(*,29)SINT
                IF(SINT.EQ.0.0)SINT=0.050
                IF(TEST.EQ.'Y')PRINT*,SINT
29              FORMAT(F4.3)
            ENDIF
                PRINT*,'What  is  the  value  of  the  seed to be
used?'
                READ(*,9)SEED
                IF(TEST.EQ.'Y')PRINT*,SEED
30              PRINT*,'With replacement?'
                READ(*,7)REPLACE
                IF(TEST.EQ.'Y')PRINT*,REPLACE

C Change lower case responce to upper case

                IF(REPLACE.EQ.'n')REPLACE='N'
                IF(REPLACE.EQ.'y')REPLACE='Y'

C Check for invalid responce

                IF(REPLACE.NE.'Y'.AND.REPLACE.NE.'N')GOTO 30
        ENDIF

C If ELIMINATIONS <>  0 then  eliminate the  satellite males from
the dataset

        IF(ELIMINATIONS.NE.0)THEN

C For each satellite male to be eliminated

                DO 45 I=1,ELIMINATIONS

C Find the satelite male in the data set

                        CALL FIND_SATELLITE(MALES,HEAD,ICRAB,
     $                                      IMALE,SIZE,ISAT,MALE)

C Eliminate the satellite male from the dataset

                        CALL
ELIMINATE(CRABS,MALES,ISAT,SIZE,ICRAB,
     $                                  SEX,MALE,IMALE)
45              CONTINUE
        ENDIF

C Print out the header information

        CALL
PRINT_HEAD(REPLACE,DATA_SET,SEED,ITERATIONS,ELIMINATIONS,
     $                  CRABS,MALES,HEADS,TYPE,WEIGHT)
```

```
C Initialize the histogram arrays to 0

        DO 49 IJ=1,NEIGHBORS

C Check  to see  if the  size and  distance difference histograms
need to be
C initialized.

          IF(ANALYSIS.GT.1.AND.ITERATIONS.NE.1)THEN
               DO 46 JJ=1,500
                       DISTO(IJ,JJ)=0
46             CONTINUE
               DO 47 JJ=1,300
                       SISTO(IJ,JJ)=0
47             CONTINUE
          ENDIF

C Check to see if the Kappa histograms need to be initialized

          IF(ANALYSIS.NE.2.AND.ITERATIONS.NE.1)THEN
               DO 48 JJ=-100,100
                       KAPPA(IJ,JJ)=0
                       O_KAPPA(IJ,JJ)=0
48             CONTINUE
          ENDIF
49      CONTINUE

C For each ITERATION

        DO 55 I=1,ITERATIONS
        IF(TEST.EQ.'Y'.AND.I.EQ.((I/10)*10))PRINT*,I

C  Initialize  the  following  arrays  if  Kappa  statistics  are
evaluated

          IF(ANALYSIS.NE.2)THEN
               DO 51 IJ=1,NEIGHBORS
                       A(IJ)=0.0
                       B(IJ)=0.0
                       C(IJ)=0.0
                       D(IJ)=0.0
51             CONTINUE
          ENDIF

C If the crabs are to be placed on the heads at random

           IF(ITERATIONS .NE. 1)

C Generate the data randomly
```

```
     $           CALL GENERATE_DATA(SEED,CRABS,HEADS,REPLACE,
     $                              UPPERLIMIT,WEIGHT)

C Determine the DIST and ORDER arrays

           CALL FIND_NN(CRABS)

C Determine the required statistics

           CALL FIND_NEIGHBORS(CRABS,NEIGHBORS,
     $      ANALYSIS)

C For each nearest neighbor group being analyzed

           DO 54 I2=1,NEIGHBORS

C Determine the statistics

                IF(ANALYSIS.NE.2)THEN

C Get the frequencies

                   A(I2)=A(I2)/CRABS
                   B(I2)=B(I2)/CRABS
                   C(I2)=C(I2)/CRABS
                   D(I2)=D(I2)/CRABS
                   P1(I2)=A(I2)+B(I2)
                   P2(I2)=A(I2)+C(I2)
                   Q1(I2)=C(I2)+D(I2)
                   Q2(I2)=B(I2)+D(I2)
                   PO(I2)=B(I2)+C(I2)
                   PE(I2)=P1(I2)*Q2(I2)+P2(I2)*Q1(I2)
                   KAP(I2)=0.0
                   IF(PE(I2).NE.0)KAP(I2)=(2*(A(I2)*D(I2)
     $                                -B(I2)*C(I2)))/PE(I2)
                   OKAP(I2)=0.0
                   IF(PE(I2).NE.1)OKAP(I2)=(PO(I2)-PE(I2))
     $                                /(1-PE(I2))
                ENDIF

C If analysis is not on the original data

         IF(ITERATIONS.NE.1)THEN

C Determine the histogram values for the appropriate analysis

            IF(ANALYSIS.NE.2)THEN

C Update the KAPPA histogram

C Determine the correct index MM
```

```
                MM=100*KAP(I2)
                IF(MM.LT.0)MM=MM-1
                IF(MM.EQ.0.AND.KAP(I2).LT.0)MM=-1
                KAPPA(I2,MM)=KAPPA(I2,MM)+1

C Update the O_KAPPA histogram

C Determine the correct index MM
                MM=100*OKAP(I2)
                IF(MM.LT.0)MM=MM-1
                IF(MM.EQ.0.AND.OKAP(I2).LT.0)MM=-1
                O_KAPPA(I2,MM)=O_KAPPA(I2,MM)+1
            ENDIF

C If analysis is on size and distance differences

            IF(ANALYSIS.GT.1)THEN

C Update the DISTO histogram

                AMT=DIST_MEAN(I2)/DINT

C Determine the index MM

                MM=AMT
                IF(AMT-REAL(MM).GT.0)THEN
                        DISTO(I2,MM+1)=DISTO(I2,MM+1)+1
                ELSE
                        DISTO(I2,MM)=DISTO(I2,MM)+1
                ENDIF

C Update the SISTO histogram

                AMT=SIZE_MEAN(I2)/SINT

C Determine the index MM

                MM=AMT
                IF(AMT-REAL(MM).GT.0)THEN
                        SISTO(I2,MM+1)=SISTO(I2,MM+1)+1
                ELSE
                        SISTO(I2,MM)=SISTO(I2,MM)+1
                ENDIF
            ENDIF
        ENDIF
54          CONTINUE
55      CONTINUE

C If analysis is on the original data

        IF(ITERATIONS.EQ.1)THEN
```

```
C For each nearest neighbor group being analyzed
C print out the statistics

            IF(ANALYSIS.GT.1)CALL PRINTOUT(NEIGHBORS)

C Print out the statistics

            IF(ANALYSIS.NE.2)CALL KAPPA_OUTPUT(NEIGHBORS)
         ELSE

C Print out the histograms
            IF(ANALYSIS.NE.2)CALL KAPPA_HISTO(NEIGHBORS)
            IF(ANALYSIS.GT.1)CALL PRINT_HISTO(NEIGHBORS,
     $      SINT,DINT,SISTO,DISTO)
         ENDIF
110      END

C End of MAIN

C_____________________________________________________________

         SUBROUTINE PRINT_HEAD(R,I,J,K,L,M,N,O,T,W)

C This subroutine prints out the parameters for the program. This

C will allow the  user to  run the  program again  using the same
input.
C The variable name have been changed to make life simpilier, but

C a dictionary will follow. The format of the header will be as
C follows:


C    Examples
C    For ITERATIONS ) 1                    For ITERATIONS = 1

C    Data set= CLP9                        Data set= CLP9
C    Type     = PS                         Type     = PS
C    Heads    = 80                         Heads    = 80
C    Crabs    = 32                         Crabs    = 32
C    Males    = 12                         Males    = 12
C    Females  = 20                         Females  = 20

C    Seed          = 1237                  Iteration    = 1
C    Iterations    = 1000                  Eliminations= 0
C    Eliminations= 0

C    Without replacement
C    Using weighting function
```

```
C R=REPLACE
C I=DATA_SET
C J=SEED
C K=ITERATIONS
C L=ELIMINATIONS
C M=CRABS
C N=MALES
C O=HEADS
C T=TYPE
C W=WEIGHT

        CHARACTER*6 I
        CHARACTER*2 T
        INTEGER O
        CHARACTER*1 R,W
        WRITE(2,5)I,T,O,M,N,M-N
5       FORMAT('0Data Set= ',A10/' Type      = ',A2/'  Heads     =
',I3/
     $' Crabs   = ',I2/' Males   = ',I2/' Females = ',I2)

C If ITERATIONS ) 1

        IF(K.NE.1)THEN
                WRITE(2,10)J,K,L
10                    FORMAT('0Seed         = ',I6/' Iterations  =
',I4/' Elimina
     $tions= ',I1)
                IF(R.EQ.'Y')THEN
                        WRITE(2,20)'With '
                ELSE
                        WRITE(2,20)'Without '
                ENDIF
                IF(W.EQ.'Y')WRITE(2,19)
19      FORMAT(' Using weighting function')
20      FORMAT('0',A,'Replacement')

C If ITERATIONS = 1

        ELSE
                WRITE(2,30)L
30              FORMAT('0Iteration   = 1'/' Eliminations= ',I4)
        ENDIF
        RETURN
        END

C End of PRINT_HEAD
```

```
C_____________________________________________________________________

      SUBROUTINE GENERATE_DATA(SEED,CRABS,HEADS,REPLACE,
     $                         UPPERLIMIT,WEIGHT)
C This subroutine simulates the placing of crabs on heads
randomly.
C Each crab is taken from its current head and placed on another
head
C at random. If the user wants the randomization to occur without

C replacement, an array IPICK will be used to determine if the
head
C is already occupied. If IPICK(NUM)=1 then the head has already
been
C chosen, therefore, another head must be picked at random until
C IPICK(NUM)=0. After the head has been chosen, the crab is
placed on
C that head by assigning ICRAB(I)=NUM ( The I'th crab now resides
at
C the NUM'th head. If the weighting function is being used, the
VOLUME
C array will be used to pick out a head.

C Variable declarations

      DIMENSION IPICK(400)
      CHARACTER*1 WEIGHT,REPLACE
      INTEGER*4 SEED
      INTEGER CRABS,HEADS,UPPERLIMIT,VOLUME(1500)
      COMMON /VOL/VOLUME
      COMMON /COM3/ICRAB(100)

C If the crabs are to be put on the heads without replacement,
initialize
C the IPICK array to all 0's to indicate that none of the heads
have
C been chosen yet.

      IF(REPLACE.EQ.'N')THEN
            DO 2 I=1,HEADS
                  IPICK(I)=0
2           CONTINUE
      ENDIF

C For each crab, assign it to a head at random

      DO 10 I=1,CRABS
            IF(WEIGHT.EQ.'Y')THEN

C Generate a random number
```

```
5                       NUM=RAN(SEED)*UPPERLIMIT+1
C NUM is in the range [1,UPPERLIMIT]
                        IF(REPLACE.EQ.'N')THEN
C If the head has already been chosen, get another one
                                IF(IPICK(VOLUME(NUM)).EQ.1)GOTO 5
                                IPICK(VOLUME(NUM))=1
                        ENDIF
C Assign the I'th crab to the NUM'th entry in the VOLUME array
                        ICRAB(I)=VOLUME(NUM)
                ELSE
C Generate a random number
6                       NUM=RAN(SEED)*HEADS+1
C NUM is in the range [1,HEADS]
                        IF(REPLACE.EQ.'N')THEN
C If the head has already been chosen, choose another one
                                IF(IPICK(NUM).EQ.1)GOTO 6
                                IPICK(NUM)=1
                        ENDIF
C Assign the I'th crab to the NUM'th head
                        ICRAB(I)=NUM
                ENDIF
10      CONTINUE
        RETURN
        END

C_______________________________________________________________________

        SUBROUTINE FIND_SATELLITE(MALES,HEAD,ICRAB,IMALE,SIZE,
     $                            ISAT,MALE)
C This  subroutine will  find the satellite male in the data set.
The
C satellite male is determined by first finding  a pair  of males
who
C  are  first  nearest  neighbors  and  who have the largest size
difference.
```

```
C If for some reason there is a tie and more than one pair of
males
C have the same maximum size difference, the pair with the
smallest
C distance difference between them will become the satellite
pair.
C After the satellite pair has been found, the satellite male
will
C be the smallest male of the two.

C Variable declarations

      DIMENSION DIST(50,50),HEAD(400,2),IMALE(100),
     $          ICRAB(100),SIZE(100)
      REAL MIN(50),MAX,IX,IY,JX,JY

C DIST(I,J) will hold the distance between the I'th and the J'th
male.
C Although DIST can be thought of as a square matrix, this
subroutine
C will treat DIST as a upper-diagonal matrix. This will save time
in
C traversing DIST. It is obvious that if you get the distance
between
C the first and second male ( DIST(1,2) ), you also get the same
value
C if you get the distance between the second and the first male
C ( DIST(2,1) ). Also, the distance between any crab and itself
is 0.
C Therefore, only using the upper diagonal is quite sufficient
for
C our purposes. One more advantage is that when traversing an
upper-
C diagonal matrix, you avoid the problem of recipricol neighbors.

C HEAD, IMALE, SIZE and ICRAB are as described in the
C main part of the program. MIN(I) will contain the smallest
distance
C from the I'th male to any other male, thus giving the reference

C distance for the first nearest neighbors. MAX will contain the
C maximum size difference encountered up to a given moment.
OLDDIST
C will be the distance between the satellite pair whose size
C difference corresponds to the MAX.


C Initialize the minimum values for each male crab to some large
value

      DO 5 I=1,MALES
```

```
                MIN(I)=999.9
5       CONTINUE

C Initialize the maximim size difference to some small value

        MAX=0.00

C Initialize  the distance  between the satellite pair to a large
value

        OLDDIST=999999.9

C Determine the  upper-diagonal  matrix  (DIST)  and  the minimum
values
C MIN.
C I is the base male

        DO 20 I=1,MALES-1

C  IX,  IY  are  the  X  and  Y  coordinates  of  the  I'th  male
respectively

                IX=HEAD(ICRAB(IMALE(I)),1)
                IY=HEAD(ICRAB(IMALE(I)),2)

C J is the target male

                DO 15 J=I+1,MALES

C  JX,  JY  are  the  X  and  Y  coordinates  of  the  J'th  male
respectively

                        JX=HEAD(ICRAB(IMALE(J)),1)
                        JY=HEAD(ICRAB(IMALE(J)),2)

C DISTANCE is the distance between the I'th and J'th male crabs

                        DISTANCE=SQRT((IX-JX)**2+(IY-JY)**2)

C Update the minimum values for the males if need be.

                        IF(DISTANCE.LT.MIN(I))MIN(I)=DISTANCE
                        IF(DISTANCE.LT.MIN(J))MIN(J)=DISTANCE

C Update the matrix

                        DIST(I,J)=DISTANCE
15              CONTINUE
20      CONTINUE

C With the DIST and MIN arrays established, we are now ready to
```

```
C traverse the matrix to find the satelite male.

      DO 40 I=1,MALES-1
            DO 30 J=I+1,MALES

C See if the crabs are first nearest neighbors

                  IF(ABS(DIST(I,J)-MIN(I)).LT.0.0001.OR.
     $               ABS(DIST(I,J)-MIN(J)).LT.0.0001)THEN

C DIFF is the size difference between the crabs

                     DIFF=SIZE(IMALE(I))-SIZE(IMALE(J))

C Get the absolute diffenence

                     ADIFF=ABS(DIFF)

C  If  the  current  size  difference  if greater or equal to the
maximum

                     IF((ADIFF-MAX).GE.-0.001)THEN

C If the current size difference is equal to the maximum

                        IF(ABS(ADIFF-MAX).LT.0.001)THEN

C If  the  current  distance  difference  is  less  than  the old
distance

                           IF((DIST(I,J)-OLDDIST).LT.0.0001)THEN

C Update the old distance

                              OLDDIST=DIST(I,J)

C Find the smallest male of the pair and designate it as the
C satellite male.

                              IF(DIFF.GT.0)THEN
                                 ISAT=IMALE(J)
                                 MALE=J
                              ELSE
                                 ISAT=IMALE(I)
                                 MALE=I
                              ENDIF
                        ENDIF

C If the size difference is greater than the maximum

                     ELSE
```

```
C Update MAX and OLDDIST

                                        MAX=ADIFF
                                        OLDDIST=DIST(I,J)

C Find the smallest male of the pair and designate it as the
C satellite male.

                                        IF(DIFF.GT.0)THEN
                                              ISAT=IMALE(J)
                                              MALE=J
                                        ELSE
                                              ISAT=IMALE(I)
                                              MALE=I
                                        ENDIF
                                     ENDIF
                                  ENDIF
                               ENDIF
30                  CONTINUE
40          CONTINUE
            RETURN
            END


C_____________________________________________________________

            SUBROUTINE ELIMINATE(CRABS,MALES,ISAT,SIZE,
           $                          ICRAB,SEX,MALE,IMALE)

C  This  subroutine  eliminates  the satellite male from the data
set.  In
C order to do this,  we  need to  delete it  from the  ICRAB,  SEX,
SIZE  and
C IMALE arrays.  The elements below the satellite male will all be
moved
C up thus writing  over the  satellite male's  position.  The last
element
C in  each array  will be duplicated,  but the upper bounds on the
arrays
C will  be decremented  by 1  , therefore,  the duplicate element
will never
C be exaimned.

C Variable declaration

            DIMENSION ICRAB(100),SIZE(100),IMALE(100)
            INTEGER CRABS
            CHARACTER*1 SEX(100)

C Delete the satellite male from the SIZE, SEX and ICRAB arrays
```

```
        DO 10 I=ISAT,CRABS-1
                SIZE(I)=SIZE(I+1)
                ICRAB(I)=ICRAB(I+1)
                SEX(I)=SEX(I+1)
10      CONTINUE

C Delete the satellite male from the IMALE array

        DO 20 I=MALE,MALES-1
                IMALE(I)=IMALE(I+1)-1
20      CONTINUE

C Decrement the variables due to the loss of a male crab from the

C data set.

        MALES=MALES-1
        CRABS=CRABS-1
        RETURN
        END

C end of ELIMINATE
C_____________________________________________________________________

        SUBROUTINE FIND_NN(CRABS)
        INTEGER ORDER(100,100)
        COMMON /COM1/ORDER,DIST(100,100)
        COMMON /COM3/ICRAB(100),HEAD(400,2)
        REAL MAX
        INTEGER CRABS

C For each base crab

        DO 40 I=1,CRABS

C Find out which head the base crab is on

                II=ICRAB(I)

C Initialize MAX to some large value

                MAX=99999

C M tells how mwny entries there are in the ORDER array

                M=0

C For each terget crab

                DO 30 J=1,CRABS
```

```
C If the base crab and the target crab are the same

                        IF(I.EQ.J)THEN

C The distance between them is 0.0

                              DIST(I,J)=0.0

C Get the next target crab

                              GOTO 30
                        ENDIF

C Find out which head the target crab is on

                        IJ=ICRAB(J)

C Determine the distance between the base and the target crab

                        DISTANCE=SQRT((HEAD(II,1)-HEAD(IJ,1))**2+
     $                                (HEAD(II,2)-HEAD(IJ,2))**2)

C Increment  M to indicate that one more entry is about to be put
into
C the OREDR array.
                        M=M+1

C Updata the DIST array

                        DIST(I,J)=DISTANCE


C If the current distance is  less  than  the  maximum,  then the
index
C belongs somewhere in the ORDER aray
C
                        IF(DISTANCE.LT.MAX.AND.M.NE.1)THEN

C Find the proper position for the index

                           DO 20 K=1,M-1
                              X=DIST(I,ORDER(I,K))
                              IF(X.GT.DISTANCE)THEN

C Shift  everything so  that the index value can be placed in the
array

                                 DO 10 L=M,K+1,-1
```

```
ORDER(I,L)=ORDER(I,L-1)
10                                          CONTINUE

C Put the index in its pproper position

                                            ORDER(I,K)=J
                                            GOTO 25
                                        ENDIF
20                                  CONTINUE
25                                  MAX=DIST(I,ORDER(I,M))
                                ELSE
                                    ORDER(I,M)=J
                                    MAX=DISTANCE
                                ENDIF
30                      CONTINUE
40              CONTINUE
                RETURN
                END

C_____________________________________________________________________

        SUBROUTINE FIND_NEIGHBORS(CRABS,NEIGHBORS,
     $    ANALYSIS)

C This subroutine will determine the appropriate statistics using
the
C DIST and ORDER arrays. CRABS is used as the upperlimit on
C the DIST, ORDER, SEX and SIZE arrays. NEIGHBORS is used as the
C upper limit on the A, B, C, D, SIZE_MEAN, and DIST_MEAN arrays.

C ANALYSIS is used to determine which statistics should be
evaluated.

C Variable declarations

        REAL MMS(5),MMD(5),FFS(5),FFD(5)
        INTEGER ORDER(100,100)
        CHARACTER*1 SEX(100)
        COMMON /COM1/ORDER,DIST(100,100)
        COMMON /COM2/SEX,SIZE(100)
        COMMON /KAPPAS1/A(5),B(5),C(5),D(5)
        COMMON /MEANS1/SIZE_MEAN(5),DIST_MEAN(5)
        DIMENSION MASK(100,100,5)
        REAL MM,MF,INDEX(5)
        INTEGER CRABS,ANALYSIS
        CHARACTER*1 BASE
        CHARACTER*2 TEMP1,TEMP2

C Initialize the variables corresponding to the sex relationships

C i.e. MM= male-male, MF=male-female, etc. They are used to
```

```
determine
C the Kappa statistics.

        MM=0.0
        MF=0.0
        FF=0.0
        FM=0.0

C If the size and distance differences are to be computed

        IF(ANALYSIS.GT.1)THEN

C Initialize the arrays to 0.0

                DO 5 I=1,NEIGHBORS

C INDEX is used to hold the number of crabs there are in the I'th

C nearest neighbor group.

                        INDEX(I)=0.0
                        SIZE_MEAN(I)=0.0
                        DIST_MEAN(I)=0.0
5               CONTINUE

C Initialize the MASK array. It is used to avoid evaluating size
and
C distance difference values more than once for a given pair of
crabs
C in the same nearest neighbor group.

                DO 15 I=1,CRABS
                        DO 12 J=1,CRABS
                                DO 10 L=1,NEIGHBORS
                                        MASK(I,J,L)=0
10                              CONTINUE
12                      CONTINUE
15              CONTINUE
        ENDIF

C For each crab

        DO 100 I=1,CRABS

C Designate it as the base. BASE will hold the sex of the base
crab

                BASE=SEX(I)

C Initialize the pointer in the ORDER array
```

```
                J=1

C  K  is  the  number  of  crabs there are in the current nearest
neighbor group

                K=0

C L is the current nearest neighbor group

                L=1

C M is the next closest crab to the base crab

20              M=ORDER(I,J)

C If at the end of the ORDER array, get a new base

                IF(J.EQ.CRABS)GOTO 50

C Determine the appropriate distance. If  you are  not looking at
the first
C element, get the previous distance differnece.

                IF(J.NE.1)THEN
                        X=DIST(I,ORDER(I,J-1))
                ELSE
                        X=DIST(I,M)
                ENDIF

C X is used to tell when you are going from one nearest neighbor
C  group  to  another.  This  happens  when the previous distance
diffreence
C dosn't equal the current distance difference.

C If we are still in the current nearest neighbor group

                IF(ABS(X-DIST(I,M)).LE.0.00001)THEN

C Increment the number of crabs  in the  current nearest neighbor
group

                        K=K+1

C  Find  their  sex  relationship  and  increment the appropriate
values

                        IF(BASE.NE.SEX(M))THEN
                                IF(BASE.EQ.'M')THEN
                                        MF=MF+1.0
                                ELSE
                                        FM=FM+1.0
```

```
                        ENDIF
                    ELSEIF(BASE.EQ.'M')THEN
                        MM=MM+1.0
                    ELSE
                        FF=FF+1.0
                    ENDIF
                ELSE

C If we are determining Kappa statistics

50              IF(ANALYSIS.NE.2)THEN
                    A(L)=A(L)+MM/K
                    B(L)=B(L)+MF/K
                    C(L)=C(L)+FM/K
                    D(L)=D(L)+FF/K
                    MM=0.0
                    MF=0.0
                    FF=0.0
                    FM=0.0
                    IF(J.EQ.CRABS)THEN

C Check to see if NEIGHBORS needs to be reduced.

                        IF(L.LT.NEIGHBORS)THEN
                            IF(I.EQ.1)THEN
                                TEMP1='st'
                            ELSEIF(I.EQ.2)THEN
                                TEMP1='nd'
                            ELSEIF(I.EQ.3)THEN
                                TEMP1='rd'
                            ELSE
                                TEMP1='th'
                            ENDIF
                            IF(L+1.EQ.1)THEN
                                TEMP2='st'
                            ELSEIF(L+1.EQ.2)THEN
                                TEMP2='nd'
                            ELSEIF(L+1.EQ.3)THEN
                                TEMP2='rd'
                            ELSE
                                TEMP2='th'
                            ENDIF
                            WRITE(2,55)L,I,TEMP1
     $                                    ,L+1,TEMP2
                            NEIGHBORS=L
                        ENDIF
                        GOTO 100
                    ENDIF
                ENDIF
55              FORMAT('0','The  number of  nearest neighbors
being
```

```
     $ analyzed was reduced to ',I1,' because'/' the ',I1,A,'
crab does
     $ not have a ',I1,A,' nearest neighbor.')

C If all the nearest neighbor groups have been examined, get a
new base

                        IF(L.EQ.NEIGHBORS)GOTO 100

C Otherwise evaluate the appropriate values

                        IF(BASE.NE.SEX(M))THEN
                                IF(BASE.EQ.'M')THEN
                                        MF=MF+1.0
                                ELSE
                                        FM=FM+1.0
                                ENDIF
                        ELSEIF(BASE.EQ.'M')THEN
                                MM=MM+1.0
                        ELSE
                                FF=FF+1.0
                        ENDIF

C Initialize the number of entries to 1 and the N-N group to 1

                        K=1
                        L=L+1
                ENDIF

C If determining size and distance differences

                IF(ANALYSIS.GT.1)THEN

C See if the pair has already been looked at

                    IF(MASK(M,I,L).NE.1)THEN

C Evaluate the statistics


SIZE_MEAN(L)=SIZE_MEAN(L)+ABS(SIZE(I)-SIZE(M))
                        DIST_MEAN(L)=DIST_MEAN(L)+DIST(I,M)

C Update the MASK and INDEX arrays

                        MASK(I,M,L)=1
                        INDEX(L)=INDEX(L)+1
                    ENDIF
                ENDIF

C Increment J to get next closest crab
```

```
                J=J+1

C Get the next base crab

                GOTO 20
100     CONTINUE

C If determining size and distance differences
        IF(ANALYSIS.GT.1)THEN
                DO 110 L=1,NEIGHBORS
                        SIZE_MEAN(L)=SIZE_MEAN(L)/INDEX(L)
                        DIST_MEAN(L)=DIST_MEAN(L)/INDEX(L)
110             CONTINUE
        ENDIF
        RETURN
        END

C_______________________________________________________________________

        SUBROUTINE KAPPA_OUTPUT(NEIGHBORS)

C  This  subroutine  prints  out  the  kappa  statistics for each
nearest
C neighbor group.

C Variable declarations

        CHARACTER*2 TEMP
        REAL KAP(5)
        COMMON /KAPPAS1/A(5),B(5),C(5),D(5)
        COMMON /KAPPAS2/P1(5),Q1(5),P2(5),Q2(5),PE(5),PO(5),
     $                  KAP,OKAP(5)

C For each nearest neighbor group, print out the kappa statistics


      DO 100 NN=1,NEIGHBORS

C See if printing should resume on a new page

        IF(NN.EQ.1.OR.NN.EQ.3.OR.NN.EQ.5)WRITE(2,58)

C Print out the table

        WRITE(2,1)
1       FORMAT('0 ',73('_'))
        WRITE(2,2)
2       FORMAT(' |',35X,'|',37X,'|')
        IF(NN.EQ.1)THEN
                TEMP='st'
```

```
          ELSEIF(NN.EQ.2)THEN
                  TEMP='nd'
          ELSEIF(NN.EQ.3)THEN
                  TEMP='rd'
          ELSE
                  TEMP='th'
          ENDIF
          WRITE(2,3)NN,TEMP
3                       FORMAT('    |    ',I1,A2,'  Nearest  Neighbor
',12X,'|',7X,'Kappa
     $ Statistics',14X,'|')
          WRITE(2,2)
          WRITE(2,4)'|  Observed proportion (Po)= ',PO(NN)
          WRITE(2,4)'|  Expected proportion (Pe)= ',PE(NN)
          WRITE(2,4)'|  Kappa (males/females)    = ',OKAP(NN)
          WRITE(2,4)'|  Kappa (males)            = ',KAP(NN)
4         FORMAT(' |',35X,A,F7.4,'   |')
          WRITE(2,5)
5         FORMAT(' |',14X,'Proportions',10X,'|',37('_'),'|')
          WRITE(2,6)
6         FORMAT(' |',73X,'|')
          WRITE(2,7)
7         FORMAT(' |',14X,'M',9X,'F',48X,'|')
          WRITE(2,8)
8         FORMAT(' |',10X,21('_'),42X,'|')
          WRITE(2,9)
9         FORMAT(' |',9X,'|',10X,'|',10X,'|',41X,'|')
          WRITE(2,10)'M',A(NN),B(NN),P1(NN)
10        FORMAT(' |',6X,A,3(2X,'|',2X,F6.4),33X,'|')
          WRITE(2,9)
          WRITE(2,11)
11        FORMAT(' | Base     |',2(10('-'),'|'),41X,'|')
          WRITE(2,9)
          WRITE(2,10)'F',C(NN),D(NN),Q1(NN)
          WRITE(2,12)
12        FORMAT(' |',9X,3('|',10('_')),31X,'|')
          WRITE(2,13)
13        FORMAT(' |',31X,'|',41X,'|')
          WRITE(2,14)P2(NN),Q2(NN),1.0000
14        FORMAT(' |',12X,F6.4,5X,F6.4,2X,'|',2X,F6.4,33X,'|')
          WRITE(2,15)
15        FORMAT(' |',73('_'),'|')
58        FORMAT('1'/'0')
100       CONTINUE
          RETURN
          END

C_____________________________________________________________________

          SUBROUTINE KAPPA_HISTO(NEIGHBORS)
```

APPENDIX C **CRABS.FOR**

```
C  This  subroutine  simply  prints  out  the  KAPPA  and O_KAPPA
histograms
C for each nearest neighbor group.

C Variable declaration

        INTEGER O_KAPPA(5,-100:100)
        COMMON /KAPPAS3/KAPPA(5,-100:100),O_KAPPA

C For each nearest neighbor group  print out  the histogram. Here
is an
C an  example of  a histogram for the first nearest neighbor With
10
C iterations.

C       Nearest Neighbor #1


C       Kappa Histogram:


C                  .00 .01 .02 .03 .04 .05 .06 .07 .08 .09
C                 |-------------------------------------------|
C          -1.0   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.9   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.8   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.7   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.6   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.5   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.4   |  0   0   0   0   0   0   0   0   0   0   |
C          -0.3   |  0   0   0   0   0   0   1   0   0   0   |
C          -0.2   |  0   0   0   0   1   0   0   0   0   0   |
C          -0.1   |  0   1   0   0   0   1   0   0   1   0   |
C           0.0   |  0   0   0   1   1   0   0   1   0   0   |
C           0.1   |  1   0   1   0   0   0   0   0   0   0   |
C           0.2   |  0   0   0   0   0   0   0   0   0   0   |
C           0.3   |  0   0   0   0   0   0   0   0   0   0   |
C           0.4   |  0   0   0   0   0   0   0   0   0   0   |
C           0.5   |  0   0   0   0   0   0   0   0   0   0   |
C           0.6   |  0   0   0   0   0   0   0   0   0   0   |
C           0.7   |  0   0   0   0   0   0   0   0   0   0   |
C           0.8   |  0   0   0   0   0   0   0   0   0   0   |
C           0.9   |  0   0   0   0   0   0   0   0   0   0   |
C                 |-------------------------------------------|


        DO 100 NN=1,NEIGHBORS

C Print out the header
```

```
                WRITE(2,60)NN
60              FORMAT('1'/'0Nearest Neighbor #',I1///3X,'
     $Kappa Histogram:'//)

C Print out the intervals across the top

                WRITE(2,66)(I*0.01,I=0,9)
66              FORMAT(10X,10(1X,F3.2))

C Print out the top part of the table

                WRITE(2,67)
67              FORMAT(9X,'|',42('-'),'|')

C For each of the intervals

                DO 70 I=-100,90,10

C Print out the appropriate values

                    WRITE(2,68)I*0.01,(KAPPA(NN,I+J),J=0,9)
68                  FORMAT(3X,F4.1,2X,'|',10(2X,I2),2X,'|')
70              CONTINUE

C Print out the bottom part of the table

                WRITE(2,67)

C Now  for the  OKAPPA histogram.  The output  will have the same
format
C as in the above example.

C Print out the header

                WRITE(2,71)
71              FORMAT('0'/'0OKappa Histogram:'//)

C Print out the intervals accross the top of the table

                WRITE(2,66)(I*0.01,I=0,9)

C print out the top part of the table

                WRITE(2,67)

C For each interval

                DO 80 I=-100,90,10

C Print out the appropriate value
```

```
                    WRITE(2,78)I*0.01,(O_KAPPA(NN,I+J),J=0,9)
78                  FORMAT(3X,F4.1,2X,'|',10(2X,I2),2X,'|')
80              CONTINUE

C Print out the bottom of the table

                WRITE(2,67)
100     CONTINUE
        RETURN
        END

C_____________________________________________________________

        SUBROUTINE PRINTSET(CRABS,SEX,ICRAB,HEAD,SIZE)
        DIMENSION ICRAB(100),HEAD(400,2),SIZE(100)
        CHARACTER*1 SEX(100)
        INTEGER CRABS
        DO 1 I=1,CRABS
                WRITE(2,2)I,SEX(I),ICRAB(I),SIZE(I),
     $                    HEAD(ICRAB(I),1),HEAD(ICRAB(I),2)
1       CONTINUE
2       FORMAT(1X,I3,2X,A,2X,I3,3(2X,F5.2))
        RETURN
        END

C_____________________________________________________________

        SUBROUTINE PRINTOUT(NEIGHBORS)

C  This  subroutine  prints  out the size and distance difference
means for
C each nearest neighbor group being analyzed.

C Variable declaration
        COMMON /MEANS1/SIZE_MEAN(5),DIST_MEAN(5)
        CHARACTER*49 LINE
        CHARACTER*2 NN

C Here is an example of what the output will look like.

C
C       ____________________________________________________
C       |                |                |                |
C       |   Nearest      |   Mean Size    |   Mean Dist.   |
C       |  Neighbors     |  Difference    |  Difference    |
C       |________________|________________|________________|
C       |                |                |                |
C       |     1st        |     1.527      |     0.249      |
C       |     2nd        |     1.901      |     0.403      |
C       |     3rd        |     1.578      |     0.532      |
C       |     4th        |     1.630      |     0.630      |
C       |     5th        |     2.162      |     0.716      |
```

```
C      |_________________|_________________|_________________|

C Initialize the table lines

LINE(1:49)='-------------------------------------------------'

C Print out 4 blank lines

        WRITE(2,5)

C Print out the top part of the table

        WRITE(2,1)LINE
1       FORMAT(1X,A)

C Put bars in LINE to print out other lines in the table

        DO 2 J=1,4
                K=(J-1)*16+1
                LINE(K:K)='|'
2       CONTINUE

C Print out the header

        WRITE(2,3)
3       FORMAT(' |',4X,'Nearest',4X,'|',3X,'Mean Size',3X,'|',3X,

     $  'Mean Dist.',2X,'|'/' |',3X,'Neighbors',3X,'|',2(3X,
     $  'Difference',2X,'|'))
5       FORMAT('0'/'0')

C Print out the middle line of the table

        WRITE(2,1)LINE

C For each nearest neighbor group, print out the means

        DO 10 L=1,NEIGHBORS

C Print out the appropriate N-N group number

                IF(L.EQ.1)THEN
                        NN='st'
                ELSEIF(L.EQ.2)THEN
                        NN='nd'
                ELSEIF(L.EQ.3)THEN
                        NN='rd'
                ELSE
                        NN='th'
                ENDIF
```

```
C Print out the means

                WRITE(2,20)L,NN,SIZE_MEAN(L),DIST_MEAN(L)
10      CONTINUE

C Print out the bottom line of the table

        WRITE(2,1)LINE
20      FORMAT(' ',5X,I2,A2,6X,'|',2(4X,F6.3,5X,'|'))
        WRITE(2,30)
30      FORMAT('0')
        RETURN
        END

C End of PRINTOUT

C_____________________________________________________________________

        SUBROUTINE PRINT_HISTO(NEIGHBORS,SINT,DINT,SISTO,DISTO)

C  This  subroutine  prints  out the size and distance difference
histograms

        CHARACTER*75 HEADER,LINE
        INTEGER SISTO(5,300),DISTO(5,500)

C Make the header and the output line all blanks

        DO 1 J=1,75
                HEADER(J:J)=' '
                LINE(J:J)=' '
1       CONTINUE

C Go to the next page on the output

        WRITE(2,3)

C Here is an example of what the  size difference  histogram will
look
C  like  with  2  nearest  neighbor  groups being analyzed and 10
iterations.

C       SIZE DIFFERENCE HISTOGRAM
C       NN                          1       2

C       ( 1.45 -  1.50]             0       2
C       ( 1.50 -  1.55]             0       1
C       ( 1.65 -  1.70]             2       3
C       ( 1.70 -  1.75]             3       0
C       ( 1.75 -  1.80]             1       0
```

```
C        ( 1.80 -  1.85]        2      3
C        ( 1.90 -  1.95]        2      1


C End of PRINT_HISTO
         WRITE(2,*)'SIZE DIFFERENCE HISTOGRAM'

C Insert the nearest neighbor numbers into the header

         DO 2 J=1,NEIGHBORS
                 K=19+6*J
                 HEADER(K:K)=CHAR(48+J)
2        CONTINUE
3        FORMAT('1'/'0')
         HEADER(1:2)='NN'

C Print out the header. In the example, the header is
C        NN                          1      2

         WRITE(2,5)HEADER
5        FORMAT('0',A/1X)

C For  each interval,  print out the number of times a value fell
in that
C interval if the value is not 0.

         DO 14 J=1,300

C See if there are any non zero values for  the current interval.
If the
C values  are 0  in all  the nearest neighbor groups, there is no
need to
C print them out. That only wastes space and paper.

                 DO 8 K=1,NEIGHBORS
                         IF(SISTO(K,J).NE.0)GOTO 10
8                CONTINUE
                 GOTO 14

C Multiply the number of the  interval by  the interval  width to
get the
C interval  upper and lower limits. (AMT1 - AMT2]. AMT1 is simply
the
C interval number multiplied by the interval width.  AMT2 is AMT1
minus
C the interval width.

10               AMT1=J*SINT
                 AMT2=AMT1-SINT

C For  each nearest  neighbor group,  print out the values in the
```

```
current
C interval.

                DO 13 I=1,NEIGHBORS

C K tells how far over in the LINE the value should be inserted.

                        K=6*I-1

C Blank  out the  position in  the LINE  where the value is to be
placed

                        LINE(K:K+2)='   '

C L1 is the digit in the 100's place

                        L1=SISTO(I,J)/100

C If the value is > 0, put the digit in the  appropriate place in
the LINE

                        IF(L1.NE.0)LINE(K:K)=CHAR(48+L1)

C L2 is the digit in the 10's place

                        L2=(SISTO(I,J)-L1*100)/10

C  If  the  value  is  >  0  or  L1  <>  0,  put the digit in the
appropriate
C place in the LINE

                        IF(L2.NE.0.OR.L1.NE.0)LINE(K+1:K+1)=
     $                                          CHAR(48+L2)

C L3 is the digit in the 1's place

                        L3=SISTO(I,J)-(L2*10)-(L1*100)

C Put that digit in the appropriate place

                        LINE(K+2:K+2)=CHAR(48+L3)
13              CONTINUE

C Print out the line with the interval  first. It  will look like
this
C       ( 1.45 -  1.50]         0       2

                WRITE(2,15)AMT2,AMT1,LINE
14      CONTINUE
15      FORMAT(' (',F6.3,' - ',F6.3,'] ',A)
```

```
C Here is an example of what the distance difference histogram
will look
C like with 2 nearest neighbor groups being analyzed and 10
iterations.


C       DISTANCE DIFFERENCE HISTOGRAM
C       NN                          1      2

C       ( 0.26 -  0.27]             1      0
C       ( 0.27 -  0.28]             1      0
C       ( 0.28 -  0.29]             1      0
C       ( 0.29 -  0.30]             3      0
C       ( 0.30 -  0.31]             3      0
C       ( 0.34 -  0.35]             1      0
C       ( 0.36 -  0.37]             0      1
C       ( 0.40 -  0.41]             0      1
C       ( 0.41 -  0.42]             0      2
C       ( 0.42 -  0.43]             0      1
C       ( 0.43 -  0.44]             0      2
C       ( 0.44 -  0.45]             0      2
C       ( 0.45 -  0.46]             0      1

C Go to the next page of the output

        WRITE(2,3)
        WRITE(2,*)'DISTANCE DIFFERENCE HISTOGRAM'

C Print out the header

        WRITE(2,5)HEADER

C For each interval, print out the values

        DO 24 J=1,500

C Check to make sure all values in the current interval are not
all 0

                DO 18 K=1,NEIGHBORS
                      IF(DISTO(K,J).NE.0)GOTO 20
18              CONTINUE
                GOTO 24

C Determine the upper and lower limit of the interval (AMT1 -
AMT2]

20              AMT1=J*DINT
                AMT2=AMT1-DINT

C For each nearest neighbor group, print out the values as before
```

```
                DO 23 I=1,NEIGHBORS
C K tells how far over in the LINE the value should be inserted.
                        K=6*I-1
C Blank  out the  position in  the LINE  where the value is to be
placed
                        LINE(K:K+2)='   '
C L1 is the digit in the 100's place
                        L1=DISTO(I,J)/100
C If the value is > 0, put the digit in the  appropriate place in
the LINE
                        IF(L1.NE.0)LINE(K:K)=CHAR(48+L1)
C L2 is the digit in the 10's place
                        L2=(DISTO(I,J)-L1*100)/10
C  If  the  value  is  >  0  or  L1  <>  0,  put the digit in the
appropriate
C place in the LINE
                        IF(L2.NE.0.OR.L1.NE.0)LINE(K+1:K+1)=
     $                                          CHAR(48+L2)
C L3 is the digit in the 1's place
                        L3=DISTO(I,J)-(L2*10)-(L1*100)
C Put that digit in the appropriate place
                        LINE(K+2:K+2)=CHAR(48+L3)
23              CONTINUE
C Print out the line of values with the interval first
                WRITE(2,15)AMT2,AMT1,LINE
24      CONTINUE
        WRITE(2,3)
        RETURN
        END
```