Computer Exercises for Elementary Statistics

Student Manual

Herbert L. Dershem

compute

COMPUTER ORIENTED MATERIALS PRODUCTION FOR UNDERGRADUATE TEACHING

Computer Exercises for Elementary Statistics

Student Manual

Herbert L. Dershem
Department of Mathematics
Hope College

Published by: Project COMPUTe, Hanover New Hampshire

To accompany: "Computer Exercises for Elementary Statistics, Instructor Manual"

This material was prepared with the support of the National Science Foundation Grant No. GJ-28456.

Prepared for publication by

COMPUTe
Dartmouth College
Hanover, New Hampshire 03755

Manufactured in the United States of America

Acknowledgements

These materials were developed and prepared with the close cooperation of my colleague and friend, Elliot Tanis. Many Hope College students assisted in writing programs and testing exercises. Of these students, Roger Crisman and Richard Meyers made the most contributions.

I wish to thank my wife, Katie, who endured much that these materials might be published.

I gratefully acknowledge the financial support of the National Science Foundation and Project COMPUTe at Dartmouth College.

Table of Contents

Computer Exercises

Introduction

Students who use the computer to assist them in learning statistics react to this experience in many different ways. Some find the computer to be a very exciting tool and find its use in learning statistics to be highly rewarding. Others discover that the computer is really more of a stumbling block than a stepping stone to their mastery of statistics. The vast majority of students lie somewhere in between these two extremes.

This manual is based on the idea that the computer can benefit all statistics students. The computer exercises found here are designed to make the job of learning statistics an easier one and to assist in later use of statistics.

There are a number of ways in which careful use of these exercises can be helpful. Much of statistics deals with the analysis of large quantities of data. When a student learns the techniques involved in this analysis, he is usually forced to learn by applying them to small and meaningless sets of values.

This is necessary because the amount of time needed to analyze a large data set is prohibitive, and such a data set is usually not available. The computer allows us to avoid this difficulty with its ability to store, retrieve, and manipulate large amounts of data at very high speeds. The student then is able to perform analysis on realistic data, thus gaining a valuable perspective on the use of the statistical techinques as well as experience in interpreting results in a meaningful way.

A common complaint of statistics students is the amount of computational work necessary in the statistics course. This work is not only uninteresting and nonbeneficial, but also extremely time consuming. Computation is, of course, one of the things a computer does best. The computer-wise statistics student is able to delegate this kind of task to the computer with ease and avoid the pain of doing it himself.

The exercises in this manual also use the computer as a device which generates demonstrations of important concepts from probability and statistics. This valuable learning technique is applied in two ways: first, by use of computer-generated graphical demonstrations of concepts or techniques, and secondly, by a device known as simulation.

The study of statistical inference is based upon the concept of probability and the fact that after a large number of repetitions of an experiment we can expect a given outcome to occur a specified proportion of the time. The computer allows one to simulate the repetition of such an experiment, observe the number of occurrences in order to verify our expectations and demonstrate the meaning of one's conclusions.

Finally, the fact that the student is learning statistics with the assistance of a computer has the effect of helping in later applications since almost all statistical analysis done in practice is done with the aid of the computer. From experience with these exercises, the student will at least know his way around the computer center, know how to submit jobs, and have overcome his initial fear of the computer. And since the computer is a valuable tool in many other areas, too, the student will probably find this computer experience helpful in ways which have nothing to do with statistics.

All of the above advantages of using the computer could be obtained without doing any computer programming. Programs exist which provide solutions to all of the exercises in this manual and an examination of these solutions would accomplish much the same purposes described above. Indeed, such a use of these materials would be a valuable addition to a statistics class.

The author feels, however, that much of the benefit of using these materials is derived from the student doing his own programming. When he programs, he becomes an active learner rather than a passive one, and becomes involved in a true learn-by-doing situation. In addition, it is well-known that a valuable technique for increasing one's own understanding of an idea is by explaining it to someone else. Programming is nothing but explaining a technique to the computer and, hence, provides excellent exercise to enhance the student's understanding. Programming a technique can do more to help a student understand it than applying it to twenty problems.

A student need not feel, however, that he must be an expert programmer to use this manual. The exercises are written with the assumption that the student is learning how to program as he works through the exercises. The earliest exercises require almost no programming expertise, while some of the later ones require more. This manual presupposes the use of the FORTRAN programming language. If the student does not already know this language it is suggested that he obtain a textbook on FORTRAN and learn it as he proceeds.

When it is necessary, in the course of these exercises, to use some more involved program segment than the student is prepared to write, the student is referred to a subprogram which has been written and stored in the computer and may be used by any programs. These subprograms are described in Part 2 of this manual. One will find these subprograms useful, not only for doing the exercises in this manual, but also for assisting with a number of the problems that are found in the statistics textbook.

How to Use the Computer Exercises

Each computer exercise consists of five parts. The first part is a statement of the purpose of the exercise. Here the student will find the technique or concept that is being dealt with and a description of how this exercise is intended to assist in understanding it.

The "Description" section of the assignment describes the action which is to take place on the computer. Sometimes this description will be a detailed specification of the program to be written and sometimes it will take a more open-ended form, leaving the details to the student's discretion.

The "Output" section describes the information the student is to have his program provide as output. This usually specifies a minimum amount and most students will find it advantageous to include, along with those values specified, a copy of all variables read into the program, which is useful as a debugging tool and in identifing answers, as well as identifying labels for all values printed.

The "Question" section is the most important part of the exercise. These questions are intended to lead the student to the important ideas in his results and stimulate him into thinking about why they were what they were. These questions should be answered thoroughly. Many exercises will instruct the student to prepare a punched card which will summarize the results of the exercise. This is done so that the instructor might obtain a summary of results the entire class by using these punched cards as input data for a summary program. Follow the instructions carefully when you prepare these cards.

The section of the assignment called "Extra Things to Try" is provided for the student who has some time and interest left after completing the first part of the exercise. This section leads him into further exploration of the ideas treated by the exercise and usually requires more programming ability than the original exercise.

There are more exercises included in this manual than can be completed in one course. It is hoped that the instructor will assign that subset of these exercises which seems appropriate to him and that you, the student, will feel free to try any of the others which look interesting.

Computer Exercise 1: The Law of Averages

Purpose: The purpose of this exercise is to familiarize the student with the use of the keypunch, the procedures for running programs, and techniques for printing his answers on the computer. It is also intended to illustrate a property of probability which common sense commonly misinterprets.

Description: You are to write a program which uses the supplied subprogram FLIP to simulate the flipping of a coin and record the results of one hundred coin flips. For instructions on using FLIP, see Part 2 of this manual, Description of Subprograms.

Output: Your output should consist of 100 computer printed lines, each line containing either the word "HEAD" or the word "TAIL".

Questions:

1. How many times do three consecutive heads appear in your listing followed by another flip? Note that 4 consecutive

heads count as two occurrences of 3 consecutive heads in the following way:

```
First occurrence of  [HEAD
       three         |HEAD]  second occurrence
                     [HEAD|       of three
                      HEAD]
                      TAIL
```

2. For how many of these occurrences of three consecutive heads is the flip immediately following also a head?

3. So that we can compile the results of the entire class on the computer, punch a card summarizing your results with the following format:

   In columns 1-2 punch your answer to question 1 above with the rightmost digit of your answer punched in column 2. In columns 4-5 punch your answer to question 2 above with the rightmost digit of your answer punched in column 5.

4. A common interpretation of the law of averages is that if a coin is fair, then after three consecutive heads it should tend to favor landing with the tail side up on the next flip in order to average itself out. Do the results you obtained in the above experiment tend to verify this idea? What would your comment be about this idea based on your knowledge of probability?

<u>Extra things to try</u>:

1. Compute the number of occurrences of three consecutive heads you would expect to get in 100 flips by the laws of probability. (You don't need the computer for this.)

2. Can you write your program in such a way that the results of more than one flip occur on each line? Try for 2 on a line first and then see if you can put 6 on a line.

Computer Exercise 2: Sum of Pairs of Dice

Purpose: This exercise is to illustrate the use of subprograms for printing integers and tossing dice. It also demonstrates the probability of an event which is the union of several mutually exclusive outcomes.

Description: Write a program which simulates the tossing of two fair dice 100 times, using the subprogram ITOSS described in Part 2 of this manual. The program is to compute the sum of each resulting pair.

Output: Your output should consist of 100 computer printed lines, each line containing the two values appearing on the dice and their sum.

Questions:

1. How many times does each sum 2 through 12 appear in the list of 100 sums?

2. Again we wish to summarize the results of the entire class, so you must punch a card which contains your results.

The card should have the following format:

In columns:	Punch the number of sums that were equal to:
1-2	2
4-5	3
7-8	4
10-11	5
13-14	6
16-17	7
19-20	8
22-23	9
25-26	10
28-29	11
31-32	12

Always punch the rightmost digit in the rightmost column of its field.

3. What is the proportion of occurrence of each of the eleven values? This proportion is computed by dividing the number of occurrences of that value by the total number of times the experiment was performed.

4. What is the true probability of the occurrence of each of the eleven values?

5. Do your two sets of answers to the preceding problems agree? How can you explain the discrepancies? How do you think you could change the experiment to make the answers to 3 closer to the answer to 4?

<u>Extra things to try</u>:

1. Repeat the experiment for the sum of three dice instead of two and answer questions 1, 3, and 4 for this set of results.

2. Write your program so that it, instead of you, counts the number of occurrences of each sum.

Computer Exercise 3: Some Rules of Probability

Purpose: The purpose of this exercise is to illustrate the addition rule, the product rule, and conditional probability.

Description: Repeat the experiment you performed for computer exercise 2 which simulates 100 pairs of dice tosses. We wish to observe for each pair of tosses whether each of the following events occurs or not:

A: A 1 appears on the first die.
B: A 2 appears on the first die.
C: A 2 appears on the second die.
D: The sum of the values appearing on the dice is 3.
E: A 1 appears on the first die or a 2 appears on the second die (A or C).
F: A 1 appears on the first die or the sum totals 3. (A or D).
G: A 1 appears on the first die and a 2 appears on the second die (A and C).
H: A 1 appears on the first die and the sum totals 3 (A and D).

Output: Same as for computer exercise 2.

Questions:

1. Count the number of occurrences of each of the eight listed events.

2. Prepare a card which summarizes your results as follows:

In columns:	Punch the number of occurrences of event:
1-2	A
4-5	B
7-8	C
10-11	D
13-14	E
16-17	F
19-20	G
22-23	H

3. Calculate the proportion of occurrence of each event by dividing the number of occurrences of the event by the total number of times the experiment was performed.

4. What are the true probabilities of the occurrence of events A,B,C, and D?

5. Which of the pairs of events (A,B), (A,C), (A,D) are independent? Which are mutually exclusive?

6. What are the true probabilities of events E-H?

7. From your output, determine an estimate of the probability of D given A. Compare this with your estimate of probability of both A and D occurring divided by your estimate of the probability of A. Does this relationship make sense?

Extra things to try:

1. Write your program so that it determines the number of times event A occurs and prints it at the end of the experiment.

2. Write your program so that it determines the number of times each of the events occurs and prints that information.

Computer Exercise 4: Conditional Probability

Purpose: The purpose of this exercise is to illustrate conditional probability by means of a baseball problem and to illustrate the simulation of one physical experiment by another one.

Description: Careful studies indicate that when a baseball is pitched to a given batter, the probability that he will swing at it is 5/6. It has also been determined that 2/3 of the times he swings, he hits the ball, and 2/3 of the times he hits the ball, it is caught. Write a program which simulates 100 pitches to this batter, determining whether he swings the bat, hits the ball, and whether the ball is caught for each pitch. Simulate this experiment by tossing three dice for each pitch of the ball.

Output: Your output should consist of 100 computer printed lines, one for each pitch of the ball. On each line, three numbers are to be printed. The first will be a one if the batter swings and a zero otherwise. The second will be a one if the batter hits the ball, and a zero otherwise. The third will be a one if the ball was caught, a zero otherwise.

Questions:

1. Determine (a) how many times the batter swung at the ball; (b) how many times the batter hit the ball; (c) how many times the ball was caught.

2. Summarize your results on a computer card as follows:

In columns:	Punch your answer to question 1, part
1-2	(a)
4-6	(b)
7-8	(c)

3. Call the event A when the batter swings, B when he hits the ball, and C that it is caught. What proportion of the 100 times did event A occur? Event B? Event C? What proportion of the times that event A occurred did event B occur also? What proportion of the times B occurred did C also occur? What proportion of the times A occurred did C occur?

4. What are the probabilities of events A,B,C? What are the conditional probabilities of B given A, C given B, and C given A? How do the true probabilities compare with the proportions you obtained? How could you modify the experiment to make them agree more closely?

Extra things to try:

1. Write your program so that it instead of you counts the occurrences of events A,B,C. Can you also have it determine

the proportion of times B occurs given that A occurs? C given B? C given A?

Computer Exercise 5: Bayes' Formula

Purpose: The purpose of this exercise is to illustrate Bayes' Formula by simulating a game using coins and dice.

Description: Simulate on the computer 100 plays of the following game: The player flips a fair coin. If it lands heads, he flips another fair coin and wins in dollars the number of heads he has showing on the two coins. If the first coin lands tails, he tosses a die and wins the number showing on the die in dollars.

Output: The computer will print 100 lines. On each line should be printed 3 numbers: (1) A 0 for tails or a 1 for heads on the first coin; (2) 0 for tails and 1 for heads on the second coin if the first coin was a head or the number showing on the die if the first coin was a tail; (3) the total number of dollars won.

Questions:

1. Examine your output and determine the proportion of the times a head was showing on the first coin out of the times the player won one dollar.

2. Determine the proportion of times a tail was showing on the first coin out of the times the player won two dollars.

3. Punch a card with the answer to question 1 in columns 1-10 and the answer to question 2 in columns 11-20.

4. Determine the expected answers to 1 and 2, i.e., use Bayes Formula to find the probability of a head on the first coin given that the player won one dollar and the probability of a tail on the first coin given that the player won two dollars.

5. (a) What is the average winning in the 100 plays simulated by the computer? (b) What is the expected average winning? (c) Could I expect to make money if I charged $3 to play the game?

<u>Extra things to try</u>:

1. Write your program so that it computes the answers to 1, 2, and 5(a).

<u>Computer Exercise 6</u>: <u>Permutations and Combinations</u>

<u>Purpose</u>: The purpose of this exercise is to illustrate by listings the idea of permutations and combinations of N objects taken K at a time.

<u>Description</u>: You will be calling two subprograms in order to perform this exercise. They are PER and COM and are described in Part 2 of this manual. The calls to these

subprograms are the only necessary statements in the program. You are to print all possible permutations and combinations of three letters chosen from among the first five letters of the alphabet. Also, you are to choose some set of names (no more than 6) and find all permutations and combinations of K (any $K \leq N$) objects selected from this set.

<u>Output</u>: All output will be done by PER and COM.

<u>Questions</u>:

1. Count the number of lines printed by each of these calls to a subprogram and verify that this is the correct number by the formulas you know for ${}_nP_r$ and ${}_nP_r$.

2. What relationship do you note between the set of all permutations and the set of all combinations?

<u>Extra things to try</u>:

1. Using the computer and your knowledge of probability compute the probability of being dealt a royal flush in poker, i.e., the ace, king, queen, jack, and ten of the same suit.

<u>Computer Exercise 7</u>: <u>Computations of Numbers in Permutations and Combinations</u>

<u>Purpose</u>: The purpose of this exercise is to aid the student's understanding of the mechanics of computing permutations and

combinations by writing a program to perform this calculation.

<u>Description</u>: Write a program which reads N and K and computes the number of permutations and combinations of N things taken K at a time. The program should then proceed to read another card, and continue the process until all the cards have been read.

<u>Output</u>: Your output should consist of a line containing N, K, the number of permutations and the number of combinations for each card read.

<u>Questions</u>:

1. Taking into account the largest value which can be represented in the computer, what is the largest N for which we can find N!?

2. Find some values of N and K for which your program will not work and punch them on a data card. What results do you get? Why?

<u>Extra things to try</u>:

1. Try to write your program so that it will work for all values of N and K which yield permutations less than or equal to the largest number which can be represented in the machine.

Computer Exercise 8: Computer Simulation--A Card Game

Purpose: The purpose of this exercise is to introduce the student to the use of the computer to simulate sampling experiments and to illustrate random sampling.

Description: Write a program to perform the following experiment: Draw a card from a deck of six cards, 3 of which are aces, 2 of which are twos and 1 of which is a three. After a card is drawn we assume that it is returned to the deck so that it may be chosen again for future draws. Simulate 100 draws from this set of cards. You may wish to use subprogram URN described in Part 2 of this manual.

Output: Your program should print the total number of each of the three types of cards drawn from your program.

Questions:

1. Punch a card summarizing your results in the following format:

In columns:	Punch the number of cards drawn that were:
1-2	aces
4-5	twos
7-8	threes

2. What is another physical problem that your results could be interpreted as simulating?

3. Suppose in the card drawing game described above, you were to be paid two dollars every time you drew an ace, but if you did not draw an ace, you had to pay as many dollars as the face value of the card. Would this be a worthwhile game for you to play? Back up your answer with some figures.

Extra things to try:

1. Write the above program so that it prints the results of each individual draw. You may have it print "1" when an ace is drawn if that is convenient.

Computer Exercise 9: Computer Simulation--A Carnival Game

Purpose: The purpose of this exercise is to introduce the student to the use of the computer to simulate sampling experiments and to illustrate random sampling.

Description: A carnival game consists of rolling a ball among a series of 20 holes, one of which the ball will eventually fall into. We assume that the ball is equally likely to fall into each hole. Eight of the holes are marked "lose" indicating that you are finished playing the game if your ball enters there. Eight of the holes are marked "rep" indicating that you may roll the ball again when your ball lands in that hold. The remaining four holes are labelled "win" and indicate that you win the teddy bear. Write a program to simulate the playing of this game for 100 turns. (A turn may consist of more than one roll of the ball.)

Output: The output should consist of one line for each roll of the ball. On that line should be printed an integer. A zero should be printed if the ball landed in a hole marked LOSE, a 1 if in a hole marked REP, and a 2 if in a hole marked WIN. In order to make the output easier to read, skip a line after each turn, i.e., after each WIN or LOSE.

Questions:

1. Punch a card summarizing your results in the following format:

In columns:	Punch the number of: turns that ended in:
1-2	LOSE
4-5	WIN

How do these results agree with what you would expect?

2. Count the total number of times the ball was tossed in the 100 turns simulated on the computer. Also count the number of tosses resulting in LOSE, REP, and WIN. How do these three values agree with what you would expect?

3. Suppose the teddy bears given as prizes are valued at 25 each and you must pay 15 for each turn. What is your expected winning (loss) when you play this game?

Extra things to try:

1. What would you expect the total number of tosses to be in 100 turns?

2. Write your program so that it prints "LOSE", "REP", or "WIN" instead of numerical code.

3. Write your program so that it computes some or all of the totals requested in questions 1 and 2.

Computer Exercise 10: Frequency Distribution--Dice

Purpose: This exercise is to give the student an opportunity to construct a frequency distribution for a familiar experiment.

Description: Write a program which tosses a pair of dice one hundred times and displays the results in a frequency distribution.

Output: The output of your program should consist of one line for each of the possible results, two through twelve, containing the sum and the number of times that sum occurred in 100 trials.

Questions:

1. What are the values you expect for each frequency? Do your results agree closely with these expectations?

2. How would an increase in the number of tosses affect the differences between the expected and observed frequency distributions?

Extra things to try:

1. Rewrite your program so that it constructs a relative frequency distribution.

2. Rewrite your program so that it constructs a cumulative frequency distribution.

Computer Exercise 11: Frequency Distribution--Live Data Sets

Purpose: This exercise is to give the student practice in constructing a frequency distribution and to familiarize him with the use of live data sets.

Description: Write a program which will read a card containing three numbers, the first specifying the left end point of the frequency distribution, the second the width of each frequency class, and the third the number of frequency classes. The program should then construct a frequency distribution for your live data set.

Output: The output of your program should consist of one line for each frequency class. This line should contain the left class boundary, the right class boundary, and the frequency for that class.

Questions:

1. Did you choose your left end point, class size and number of classes so that all data values are contained in your frequency distributions? If you did not, how would your program handle such data?

2. Why would your frequency distribution give little information if you had too few or too many frequency classes? Do you think the number you chose is good or should you have chosen more or less?

Extra things to try:

1. Rewrite your program so that it constructs a relative frequency distribution.

2. Rewrite your program so that it constructs a cumulative frequency distribution.

Computer Exercise 12: Histograms

Purpose: The purpose of this exercise is to introduce the use of the subprograms which allow the construction of histograms on the computer.

Exercise 12

<u>Description</u>: Write a program which will construct a pair of histograms of the live data sets assigned to you. Construct one histogram using the subprogram HIST1 which does not require you to determine the frequency classes, and another using subprogram HIST which does require specification of the frequency classes. HIST and HIST1 are described in Part 2 of this manual.

<u>Output</u>: The output will be generated by the subprograms.

<u>Questions</u>:

1. Does the variable which you are considering appear to be symmetric, skewed to the left, or skewed to the right?

2. What would be a verbal interpretation of your answer to question 1?

3. From looking at your histogram, what would you estimate the central or middle value of your variable to be?

4. Do you see any dangers in allowing the computer to choose the intervals? What are they?

<u>Extra things to try</u>:

1. Consider the random variable which consists of the number of heads occurring in 20 flips of an unfair coin whose probability of landing with its head showing is 0.25. Write

a program to simulate the generation of this random variable and record your results for 100 sets of 20 flips. Use HIST to construct a histogram from these 100 samples of the random variable and answer questions 1-3 for this variable.

Computer Exercise 13: Mean and Standard Deviation of Grouped Data

Purpose: This exercise is to give the student practice in computing the mean and standard deviation from grouped data. He is to compare these with the same statistics computed from an ungrouped sample.

Description: Write a program which constructs a frequency distribution with 10 frequency classes from your live data variable. From this frequency distribution, compute estimates of the mean and standard deviation. Also compute the true mean and standard deviation of the data for comparison purposes.

Output: Your output should consist of a printout of the mean and standard deviation computed both ways.

Questions:

1. Does there appear to be a significant difference between the two means which you have computed? What about the standard deviations?

2. Which of the two methods would you prefer, taking into consideration the amount of computing and accuracy? What would be a possible situation when you would prefer the other method?

Extra things to try:

1. Include in your output a printout of the frequency distribution.

2. Repeat the above with only 5 frequency classes. How does this affect your results? Is this what you would expect?

Computer Exercise 14: Testing a Random Number Generator

Purpose: The purpose of this exercise is to use the statistical summary tools available to test the randomness of the local random number generator.

Description: A uniform random number generator on the interval (0,1) should generate numbers between zero and one with each possible number in this interval having an equal chance of being chosen. In reality, computers cannot do this since they cannot even represent all of these numbers. But a program, called a pseudo-random number generator, is available on the computer to approximate a random number generator. We wish to test the randomness of the numbers generated.

We shall generate one hundred numbers by means of the pseudo-random number generator, compute their mean and standard deviation and construct a frequency distribution with 10 equal classes.

Output: Your output should consist of the mean, standard deviation and the frequency distribution in any form you wish.

Questions:

1. What would you expect the mean and standard deviation to be? Give some explanation for your expectations.

2. What would you expect the frequencies to be?

3. How well do your results agree with what you expect? Do you conclude that the pseudo-random number generator is close to random? Give reasons for your answer.

4. Summarize your results by punching a card with the following format:

In columns:	Punch the:
1-4	Mean
6-10	Standard Deviation
11-15, 16-20, 21-25, 26-30, 31-35, 36-40, 41-45, 46-50, 51-55, 56-60	The Frequencies

5. Make a histogram of the frequency distribution which you obtain.

Extra things to try:

1. Also compute the mean and standard deviation from the grouped data using the frequency distribution you have generated.

2. Do the above assignment for the sums of pairs of pseudo-random numbers. That is, generate 200 numbers and pair them to form 100 sums. Now answer questions 1, 2, 3 and 5 for your results.

Computer Exercise 15: The Median

Purpose: The purpose of this exercise is to reveal properties of the median, the first and third quartiles and their relationship to the mean and standard deviation.

Description: Write a program which will find the median, first and third quartiles, and the inter-quartile range for the live data set assigned to you.

Output: Your output should consist of those four quantities which you are asked to find above.

Questions:

1. How does the median compare with the mean which you computed in exercise 13? What is the connection between this relationship and the skewness of the distribution?

2. How does the interquartile range compare with the standard deviation? Explain a possible reason for such a relationship.

3. Compare the mean, standard deviation measures to the median, interquartile range measures as far as the amount of work involved in their computation and reliability. Which would you prefer to compute?

Extra Things to try:

1. Modify your program for exercise 13 to compute the median of a set of grouped data.

Computer Exercise 16: Percentiles

Purpose: The purpose of this exercise is to introduce the idea of percentiles and get a feel for their meaning.

Description: Determine the 10th, 25th, 50th, 75th, and 90th percentiles for your live data set.

Output: Your output should consist of 5 lines, two numbers per line. The first number should indicate the level of the percentile, and the second number should be the percentile value.

Questions:

1. Discuss some advantages of percentiles over the other measurements discussed in the textbook. What are some of the disadvantages?

2. How did you handle the situation where x% of the number of data points was not an integer?

Extra things to try:

1. Write your program so that it prints all percentiles of your data set.

Computer Exercise 17: Chebychev's Inequality

Purpose: The purpose of this exercise is to provide an illustration of Chebyshev's Inequality.

Description: The program should read all the values of the live data set which was assigned to you and use the values of the mean and standard deviation of this data set. It should read a card containing a real value X greater than one and compute the number of data values which lie within X standard deviations of the mean. The program should also compute the lower bound on this number provided by Chebyshev's inequality. The program should then repeat by reading another value of X, continuing until all cards are read.

Output: Each output line should consist of three numbers: X, the number of data values within X standard deviations of the mean, and the lower bound computed from Chebyshev's inequality.

Questions:

1. Repeat the above for 5 different values of X. Do your results agree with the prediction of Chebyshev's inequality?

2. Within at least how many standard deviations of the mean can we be certain that 95% of a set of data values lie?

Extra things to try:

1. Repeat the above experiment, using for your data values the number of heads occurring in 20 flips of a fair coin. Repeat the experiment of flipping 20 coins 100 times, recording the number of heads each time; find the mean and standard deviation of this set of 100 values and compute and print the same values asked for above.

Computer Exercise 18: Probability Distributions

Purpose: To examine some properties of probability distributions through a given example.

Description: Write a program which will perform the following experiment 100 times: Toss three dice and record the number of dice which show a one. The sample space for the random variable thus generated consists of the integers 0, 1, 2, and 3. Compute the sample mean and standard deviation of the random variable for the sample generated.

Output: The program should print the frequency distribution on four lines, each line representing the number of ones showing (0 to 3) and the number of times that event occurred. Also a line should be printed indicating the sample mean and standard deviation.

Questions:

1. Compute the theoretical frequency distribution of this random variable and compare it with the sample you obtained.

2. Compute the theoretical mean and standard deviation of this random variable; compare with those obtained for the sample.

3. Summarize your results on a card punched with this format:

In columns:	Punch the number of times the number of ones was:
1-2	0
4-5	1
7-8	2
10-11	3

Extra things to try:

1. Suppose you were paid two dollars for every one appearing on a die. How much should you pay to roll three dice if it is to be a fair game?

2. Write a program to play the above game, keeping a tally of its winnings over 100 tosses of three dice. What was its average winning?

3. Make a histogram for the frequency distribution obtained in this exercise. In addition, make a histogram for the theoretical distribution derived in question 1. Compare the two.

Computer Exercise 19: The Binomial Distribution

Purpose: The purpose of this experiment is to illustrate some properties of the binomial distribution by simulating a binomial experiment.

Description: The probability that a student has a fifth hour class on Wednesday is 0.2. The teacher wishes to give a makeup test to seven students in a class on fifth hour next Wednesday. Simulate the choice of seven students one hundred times and for each choice of seven determine how many will have a class conflict.

Output: Your output should consist of a frequency distribution of the number of occurrences of 0,1,2,...,7 students with conflicts.

Questions:

1. Summarize your results on a punched card with the following format:

In column:	Punch the number of times there were:
1-2	0 conflicts
4-5	1 conflict
7-8	2 conflicts
10-11	3 conflicts
13-14	4 conflicts
16-17	5 conflicts
19-20	6 conflicts
22-23	7 conflicts

2. Compute the expected values for the 8 frequencies in the frequency table. Do they agree well with the results you obtained?

3. Compute the mean number of conflicts for your one hundred observations as well as the standard deviation. How do these values agree with what you would expect?

Extra things to try:

1. Construct a histogram for the frequency distribution obtained above and for the frequency distribution of the expected values. Compare the two.

Computer Exercise 20: Binomial Probability

Purpose: The purpose of this exercise is to familiarize the student with the computation of binomial probability.

<u>Description</u>: Write a program which will read values for n, p, and x and compute $\frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}$.

The program should continue reading values of n, p, and x until all data cards have been read. Obtain these same values from the subprogram BINOM described in Part 2 of this manual.

<u>Output</u>: Your output consists of one line for each card read. This line is to contain n, p, x, the binomial probability computed and the binomial probability from BINOM.

<u>Questions</u>:

1. Use the above program to solve the following problem: If the probability that a child of certain parents has blue eyes is 1/4, and if there are six children in the family, what is the probability that at least half will have blue eyes?

2. Calculate the probability that 0 will have blue eyes; that 1 will; that 2 will.

3. Use your program to calculate the probability that at least 4 heads occur when a fair coin is flipped 10 times.

<u>Extra things to try</u>:

1. Write a program using subprogram URN to conduct the experiment described in question 1 one hundred times. Compare the results with the theoretical results.

Computer Exercise 21: The Normal Distribution

Purpose: The purpose of this exercise is to illustrate properties of the normal distribution and to show the student how to obtain samples from the normal distribution.

Description: Take a sample of size 100 from a population with a normal distribution of mean μ and standard deviation σ using NORM1, which is described in Part 2 of this manual. The values of μ and σ should be read from a data card and you may choose any values you like for these parameters. Compute the mean and standard deviation of this set of 100 sample values and use the SUPER subroutine to obtain a histogram of this data set with the normal curve with mean μ and standard deviation σ superimposed.

Output: Your output is to consist of the mean and standard deviation of your population, the mean and standard deviation of your 100 sample values and the histogram with superimposed normal curve provided by SUPER.

Questions:

1. Are the population mean and standard deviation the same as the sample mean and standard deviation? Explain why or why not.

2. Does the histogram appear to be similar in shape to the normal curve? Is it symmetric? Does it have a single peak value?

3. Using the histogram, try to estimate how many of the 100 sample values lie within one sample standard deviation of the mean. Is this a value you would expect from Chebyshev's inequality? How does it relate to the value you would expect for the normal distribution?

<u>Extra things to try</u>:

1. Repeat the above exercise taking a sample of size 200 instead of 100. Do you expect these 200 points to have a histogram that looks more normal than the results above? Do your results verify your expectations?

<u>Computer Exercise 22</u>: <u>The Standard Normal Distribution</u>

<u>Purpose</u>: The purpose of this exercise is to show how the standard normal distribution is obtained and illustrate some of its properties.

<u>Description</u>: Generate a sample of size 100 from a normal population with mean μ and standard deviation σ using NORM1. The values of μ and σ should be read from data cards and you may choose any value for μ except 0 and any value for except 1. Compute the mean and standard deviation of this set of 100 sample values. Then for each of the 100 sample

values x_i, calculate and store $y_i=(x_i-\mu)/\sigma$, that is subtract from the sample value and divide the result by σ and store this result in a new subscripted variable y. Compute the sample mean and standard deviation of the y's. Then use SUPER to obtain a histogram of the y_i values with the normal curve with mean 0 and standard deviation 1 superimposed.

Output: Your output is to consist of the values of μ and σ, the sample mean and standard deviation of the x's, the sample mean and standard deviation of the y's, and the histogram with superimposed normal curve provided by SUPER.

Questions:

1. Could you have predicted the mean and standard deviation of the x values? Explain.

2. How could you have obtained a sample similar to the y sample directly from NORM1?

Extra things to try:

1. Write a program which takes a sample of size 100 from a normal distribution with mean 0 and standard deviation 1 and converts that sample into a sample from a normal distribution with mean 10 and standard deviation 5. Print out the same results prescribed above.

Computer Exercise 23: Normal Approximation to the Binomial

Purpose: The purpose of this exercise is to illustrate how the normal distribution can be used as an approximation to the binomial, when the approximation is good, and how to use the normal table subprogram FNRML.

Description: Write a program which reads n and p and using BINOM computes the binomial probability that x = k for k = 0,1,...n. In addition compute the normal probability that $k - 1/2 \leq x \leq k + 1/2$ for the normal distribution with mean np and variance np(1-p), and the same values of k. Find the normal probability by FNRML described in Part 2 of this manual.

Output: The output is to consist of one line containing n, one containing p, followed by n + 1 lines each containing k, the calculated binomial probability, and the calculated normal probability.

Questions:

1. Try out your program for a variety of values of n and p. Do the cases where np and n(1-p) are both greater than 5 show good accuracy? How is the accuracy when the above rule is violated?

2. Does the accuracy of the approximation tend to vary with k for fixed n and p? If so, how?

3. Do you notice that one probability is always larger than the other? Can you explain this?

Extra things to try:

1. Write a program which is the same as the one described above but which computes the probability that $x \leq k$ instead of $x = k$. Answer questions 1-3 for this program.

2. Use SUPER to superimpose the approximating normal curve over the histogram for the binomial you used above.

Computer Exercise 24: Normal Approximation to the Binominal-Histograms

Purpose: The purpose of this exercise is to use histograms to illustrate how the binomial distribution can be approximated by the normal distribution.

Description: Conduct the experiment of flipping 10 fair coins 100 times, each time recording the number of heads occurring. Make a histogram of these values and superimpose over it the curve for the normal distribution with mean 5 and standard deviation SQRT(2.5).

Output: Your output will be provided by the subroutine SUPER. In addition, print the frequency distribution for the number of heads in 10 flips of the coin.

Questions:

1. How does the frequency distribution obtained compare to the expected values for such a frequency distribution?

2. Would you expect the normal approximation to be better if the coins were not fair? Explain.

3. Would you expect the normal approximation to be better if you were measuring the number of heads in 20 flips? Explain.

Extra things to try:

1. Try modifications suggested by questions 2 and 3 in your program. Do they yield the expected results?

Computer Exercise 25: Random Number Generator

Purpose: The purpose of this experiment is to allow the student to gain experience in the use of the random number generator and to learn some ways it is applied.

Description: Use the random number generator RAN to generate and print 100 random numbers. Use HIST1 to construct a histogram with 10 frequency classes. Then convert these 100 random numbers into 100 random integers between 0 and 99 inclusive.

Output: Your output should consist of the 100 random numbers, the histogram provided by HIST1 and the 100 random integers. It might make it easier for you to answer question 2 if you print the 100 integers in increasing order.

Questions:

1. What does the histogram tell you about the randomness of the numbers you generated? Is it close to what you expected?

2. Do any of the random integers occur more than once in the list of 100? Which one occurs most often and how often does it occur? Does this surprise you?

3. Suppose you wished to generate random integers between 1 and 100 instead of 0 and 99. How would you modify your program to do this?

4. Suppose you wished to generate random integers between 10 and 20 inclusive. How would you modify your program to do this?

Extra things to try:

1. Write a program which reads from a card two integer values K1 and K2, and generates 100 random integers between K1 and K2 inclusive.

Computer Exercise 26: Random Sampling from a Probability Distribution

Purpose: This exercise is to give the student experience in obtaining a random sample on the computer.

Description: It is known that at Hope College 29.6% of the students are freshmen, 28.1% are sophomores, 22.6% are juniors, and 19.7% are seniors. Suppose we are conducting a survey which is to be a random sample from the entire student body and we want to be certain that each class is sampled by each interviewer as frequently as is indicated by its percentage.

In order to do this we select a freshman with probability 0.296, a sophomore with probability 0.281, etc. If each interviewer is to interview 10 students, write a program which will determine, for each interviewer, how many of his 10 interviews should come from each of the four classes. Use the subprogram RAN described in Part 2 of this manual to make sure the choice is random.

Assign the values 1-4 to freshmen through senior respectively and compute the numerical mean for each sample of size 10. Repeat this five times; that is, generate five sets of samples of size 10 and calculate the mean.

Output: The output should consist of a list of the classes of the ten students chosen (you may print 1 for freshman, 2 for sophomore, etc.) and the mean of the ten values.

Questions:

1. What is the theoretical mean of the above distribution? Do the calculated means agree closely?

2. Punch a card summarizing your results using the format:

In columns:	Punch the mean for sample number:
1-10	1
11-20	2
21-30	3
31-40	4
41-50	5

3. What is the median class of Hope students? What is the modal class?

Computer Exercise 27: Random Sampling from Data Sets

Purpose: The purpose of this exercise is to illustrate how a random number generator can be used to take a random sample from a population data set.

Description: Use the random number generator RAN to obtain a random sample of size 50 from your live data set. You can do this by generating a random integer between 1 and N, where N is the number of values in your data population.

Output: Your output should consist of fifty lines, each containing the data value chosen and its position within the data set.

Questions:

1. Were any values chosen more than once in the fifty choices? Does this make your random sample invalid? How might you correct this?

2. What is the probability of choosing the same value more than once when choosing a random sample of size 50 from your data set? You may wish to use the computer to compute this.

Extra things to try:

1. Rewrite your program so that it cannot choose the same sample value more than once in the sample.

Computer Exercise 28: Sampling Distribution of the Mean

Purpose: The purpose of this exercise is to illustrate the Central Limit Theorem by means of repeated random sampling from a live data set.

Description: Write a program which reads from a card a value of n, selects 100 random samples of size n from the live data set assigned to you using RAN, and computes and stores the mean of each sample of size n. The program is then to compute the mean and the standard deviation of 100 sample means. Use for input values of n, the values 1, 4, 10, and 25.

Output: Your output should consist of the value of n, the mean of the 100 sample means, the standard deviation of the 100 sample means, and a histogram of the sample means.

Questions:

1. Does the distribution from which you are sampling appear to be approximately normal? Give reasons for your answer.

2. The Central Limit Theorem says that we can expect the distribution of the sample mean to become normal as n increases. Do you notice this happening with your data? Explain why you would expect the mean of the sample means of samples of size 25 to be nearer the true mean of the population than the mean of the sample means of samples of size 10.

3. What would you expect the standard deviation of the sample means to approach as n gets large? Do your results verify your expectations? Explain.

Extra things to try:

1. Repeat the experiment above, only obtain your samples from a population which is normal with mean 50 and variance 100. Use subprogram NORM1 to obtain your random samples. Compare and contrast your results with those obtained above.

Computer Exercise 29: Unbiased Estimates

Purpose: The purpose of this exercise is to intuitively motivate the use of the unbiased estimate of the variance by experimentation and to emphasize the distinction between sample statistics and population parameters.

Description: Obtain 100 random samples of size 5 from a population which is normal with mean 50 and variance 100 using NORM1 described in Part 2 of this manual. For each sample of size 5, compute the biased estimate of the variance by

$$s_b^2=\Sigma(x_i-\bar{x})^2/n,$$

the unbiased estimate of the variance

$$s_u^2=\Sigma(x_i-\bar{x})^2/(n-1),$$

and two corresponding estimates of the standard deviation given by $\sqrt{s_b^2}$ and $\sqrt{s_u^2}$. After all samples have been generated, compute the mean of the 100 values of s_b^2, s_u^2, $\sqrt{s_b^2}$ and $\sqrt{s_u^2}$.

Output: Your output should consist of the four computed values for each of the 100 samples of size 5 and a final line containing the 4 means of the previously printed values.

Questions:

1. Which of the two estimates of the variance came closest to the actual variance of 100? Summarize your results on a card punched with the following format:

In columns:	Punch the means of the values:
1-10	s_b^2
11-20	s_u^2
21-30	$\sqrt{s_b^2}$
31-40	$\sqrt{s_u^2}$

2. Neither of the two estimates used above for the standard deviation is unbiased. Is this noticeable from your sample data? Explain.

3. Do you think the difference between these estimates could be eliminated if more than 100 samples were taken?'

Extra things to try:

1. As mentioned above, neither of the two estimates of the standard deviation is unbiased. The unbiased estimate is given by

$$S = \Gamma\left[\frac{n-1}{2}\right]\sqrt{\frac{n-1}{2}}\ \sqrt{s_u^2}\Big/\Gamma\left[\frac{n}{2}\right]$$

where Γ is a special function known as the gamma function. The gamma function can be evaluated by the subprogram GMMA as described in Part 2 of this manual. Add this statistic to the printout above and note whether this is nearer to 10 than the other two estimates.

2. Run the above program for samples of size 10 rather than 5. Does the difference between biased and unbiased estimators diminish as n becomes larger? Why?

Computer Exercise 30: A Statistical Subroutine

Purpose: The purpose of this exercise is to write a general purpose statistical subroutine to calculate a number of descriptive statistics, thus giving the student experience in the writing of subroutines and also allowing him to have a useful subroutine available for later use.

Description: Write a subroutine subprogram which accepts as input arguments a subscripted variable A and an integer N which specifies the number of sample values. The subroutine is to compute the mean and standard deviation of the set of data values and return them as output arguments.

Output: Your output should be sufficient to check whether the subroutine is performing properly or not.

Questions:

1. State what advantages you feel using a subroutine like the one written above gives to someone who wishes to compute the mean and standard deviation.

2. What disadvantages can you think of to using such a subroutine?

3. If someone wishes to use this subroutine, he must have instruction in its use. Write a description of your subprogram which is complete enough that anyone knowing as much FORTRAN as you would be able to use it after reading your instructions.

Extra things to try:

1. Include in your subroutine the calculation of the median and midrange of the data set and have these values returned as output arguments.

2. Include in your subroutine the capability of printing a histogram of the data set with ten frequency classes.

Computer Exercise 31: Generate Statistical Tables

Purpose: The purpose of this exercise is to provide the student with experience in designing computer output and an understanding of statistical tables.

Description: Write a program which will prepare a normal table of the same form as the normal table in your textbook. You may use the subprogram FNRML to obtain the values.

Output: Your output should be a page exactly like the page in your textbook. You may omit any vertical and horizontal lines in your output.

Questions:

1. Do all of your values agree with those in the textbook tables? Which table do you think is correct? What do you think causes this difference? How could you go about making them agree?

Extra things to try:

1. Write a program to reproduce a table of binomial probabilities whose first few lines are:

		p										
n	x	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
2	0	0.902	0.810	0.640	0.490	0.360	0.250	0.160	0.090	0.040	0.010	0.002
	1	0.095	0.180	0.320	0.420	0.480	0.500	0.480	0.420	0.320	0.180	0.095
	2	0.002	0.010	0.040	0.090	0.160	0.250	0.360	0.490	0.640	0.810	0.902
3	0	.	.									
	1	.	.									
	2	.	.									
	3	.	.									
	.	.	.									
	.	.	.									
	.	.	.									
	.	.	.									
	.	.	.									
	.	.	.									

Do for all n up to n = 9.

Computer Exercise 32: An Alphabetical Frequency Distribution

Purpose: The purpose of this exercise is to give the student experience in character manipulation and non-numeric frequency distributions.

Exercise 33

Description: Your program is to read a count card on which an integer N is punched specifying the number of text cards that follows. It then reads N cards, each of which contains a portion of a written text. Your program is to print a copy of the text and make a frequency distribution which lists the number of occurrences of each letter in the text.

Output: Your output should consist of the text and a relative frequency distribution of the letters printed in any form you find convenient.

Questions:

1. Do you think your relative frequency distribution is near to the true relative frequency distribution for all English texts? What factors influence this?

Extra things to try:

1. Repeat the above, but add to the set of characters the digits and special characters available on the keypunch and printer. Then apply the program to a FORTRAN program you've previously written.

Computer Exercise 33: Printing a Data Set

Purpose: The purpose of this exercise is to give the student practice in designing printouts.

Description: Write a program to print your data set with all values properly labelled. Place a heading on your listing describing the nature of the values. Add anything else to the printout that will make it easier to read.

Output: Your output will consist of a listing of your data values along with such descriptive material that you feel is necessary so that anyone reading the output will know what it is.

Computer Exercise 34: Confidence Intervals

Purpose: The purpose of this exercise is to illustrate confidence intervals.

Description: Write a program which selects 50 samples of size 10 from a normal population with mean 50 and variance 100. For each sample construct a 95% confidence interval by finding its end points assuming that $\sigma = 10$ is known. Use subprogram CONIN described in Part 2 of this manual to print an illustration of the 50 confidence intervals you have constructed. Repeat the above experiment for the same 50 samples of size 10 only for each sample use the value of s computed for that sample as an estimate of σ in computing the confidence interval, that is, assume σ is unknown. Use CONIN to illustrate these 50 intervals.

Output: All output will be provided by CONIN.

Exercise 34

Questions:

1. Count the number of intervals that contain the mean for each of the two sets of 50 confidence intervals and punch your results on a card with the following format:

In columns:	Punch the number of intervals containing the mean for:
1-2	known
4-5	unknown

2. What is the expected number of intervals containing the mean?

3. How would you go about making your 95% confidence intervals narrower?

4. What would be the effect on the size of the intervals if you used a 90% confidence limit rather than 95%?

5. Do you seem to make any serious errors by using s in place of σ? Explain.

Extra things to try:

1. Repeat the above experiment for samples of size 20. Is the result what you expected?

2. Repeat the above experiment using a 90% confidence limit.

Computer Exercise 35: Student's t Distribution

Purpose: The purpose of this exercise is to illustrate the advantage of using Student's t distribution for small sample tests.

Description: Write a program similar to the one written for exercise 34 which selects 50 samples of size n from a population which is normal with mean 50 and variance 100. It should construct the 95% confidence intervals for these samples replacing by s as done in exercise 34. Using the same samples, construct 95% confidence intervals using the small sample test with n - 1 degrees of freedom.

Output: Output will consist of two sets of 50 confidence intervals, all provided by subprogram CONIN. Label your two sets of intervals by hand.

Questions:

1. Run the above experiment for n = 5, 10, 20, 30. Count the number of intervals containing the mean in each case.

Summarize your results on a card punched with the following format:

In columns:	Punch the number of intervals containing the mean for the:	with n =
1-2	large sample test	5
4-5	large sample test	10
7-8	large sample test	20
10-11	large sample test	30
13-14	small sample test	5
16-17	small sample test	10
19-20	small sample test	20
22-23	small sample test	30

2. Which of the two techniques appears to give the more accurate confidence intervals? Explain your answer. Is that what you would expect? What is the reason?

3. What is the effect of increasing n on the validity of the large sample test?

Extra things to try:

1. Observe the effect of using a non-normal population by conducting the above experiment for n = 5 with your live data set. How many of the confidence intervals contain the mean?

Computer Exercise 36: Determination of Sample Size

Purpose: The purpose of this exercise is to give the student experience in determining the sample size needed to construct a confidence interval of prescribed length.

<u>Description</u>: Write a program which samples from a given distribution with unknown mean and standard deviation until the 95% confidence interval has a length which is less than some prescribed tolerance. Your program should read a value of T, the tolerance, and then continue sampling from your data set, constructing a 95% confidence interval after each individual value is added to the sample. When the confidence interval is found to be of length less than T, that will be the confidence interval which you use.

<u>Output</u>: Your output should consist of T, the sample size N for which this tolerance was reached, the sample mean and standard deviation for your sample of size N, and the endpoints of the confidence interval, all properly labelled.

<u>Questions</u>:

1. Is the true population mean in the determined confidence interval?

2. How would your program be different if σ were known? Would it be simpler?

3. If you completely recompute $\overline{x}$ and s after each value is added to the sample, see if you can't find some way of saving computer effort here.

4. How would a larger population standard deviation affect the sample size N needed? Is this certain or just probably true?

5. How would a larger value of T affect the sample size N needed? Is this certain, or just probably true?

6. Did you decide to use the normal or t values to obtain the confidence intervals? Explain why.

<u>Extra things to try</u>:

1. Write a program which uses the suggestions you made in your answer to question 2 and determine the sample size necessary to obtain a confidence interval of length less than T for your data set using the known population standard deviation.

<u>Computer Exercise 37</u>: <u>Estimation of a Proportion</u>

<u>Purpose</u>: The purpose of this exercise is to give the student experience in estimating a proportion.

<u>Description</u>: Use the subroutine DRAW described in Part 2 of this manual to "deal" 50 poker hands from an ordinary bridge deck. A poker hand consists of five cards and the program is to shuffle the deck, deal 10 hands, shuffle the deck again, deal 10 hands, and so on, until 50 hands have been dealt. The program is to count the number of hands out of the 50 which contain at least one pair, that is, two cards of the same rank such as two queens. Then a 95% confidence interval is to be constructed for the proportion of all poker hands that contain at least one pair.

Output: Your output should consist of the number of hands containing a pair, and the endpoints of your confidence interval. You may wish to check your program by also printing the contents of each of the fifty hands.

Questions:

1. Use your knowledge of probability to compute the actual proportion of hands containing at least a pair. You may use the computer to do this if you wish.

2. Assuming your estimate of the proportion is correct how many hands would you need to deal to be 95% confident that the true proportion lies in an interval of length less than 0.05?

3. Assuming no knowledge about the true proportion, what is the smallest number of hands you would need to be 95% confident that the true proportion lies in an interval of length less than 0.05?

Extra things to try:

1. Write a program which reads a sentence and estimates the proportion of letters that are vowels. Compute a 95% confidence interval.

Computer Exercise 38: Testing of Hypotheses

Purpose: The purpose of this exercise is to introduce the student to testing a hypothesis about a sample mean and to illustrate type I and type II errors.

Exercise 38

Description: Write a function subprogram which has the following arguments:

XMUO	The hypothesized value of μ.
XMU	The actual value of μ.
SIG	The actual value of σ.
N	The sample size.

The subprogram then generates 100 samples of size N from a normal population with mean XMU and standard deviation SIG. For each sample, a 95% confidence interval is constructed assuming σ known and a test is made as to whether XMUO is in the confidence interval or not, i.e., whether μ = XMUO is accepted or rejected. A count is made of how many times the hypothesis is accepted. This is the value to be returned for the function. Write a calling program which calls with the following values for its parameters.

XMUO	XMU	SIG	N
20	20	5	10
20	22	5	10
20	25	5	10
20	30	5	10

Output: Your output should consist of XMUO, XMU, SIG, N, and the value of the function for each call of the function. All answers should be labelled.

Questions:

1. Relate the results of each call to the subprogram to either type I or type II errors. Specify which.

2. Punch a card summarizing your results with the following format:

In columns:	Punch the number of "accepts" when XMU is:
1-2	20
4-5	22
7-8	25
10-11	30

3. What would be the effect on your answers if SIG were 10 instead of 5? What if N were 20 instead of 10? What if we used a 99% confidence interval instead of 95%?

4. Compute the theoretical probability of making an error in each of the four performed tests of hypothesis. Indicate for each whether it is a type I or type II error. Compare these with your results.

5. The above program tests the hypothesis $\mu = 20$ against the two-sided alternative $\mu \neq 20$. How would your answers be changed if a one-sided alternative $\mu > 20$ were used? What if $\mu < 20$ were the alternative?

<u>Extra things to try</u>:

1. Add enough generality to your program so that you can try some of the things suggested in questions 3 and 5 above.

Computer Exercise 39: Testing Hypotheses for Live Data

Purpose: The purpose of this exercise is to give the student an opportunity to apply what he has learned about hypothesis testing to the live data sets.

Description: It is well known that the scores on the SAT are scaled so that the mean score over the entire population tested is 500 while the standard deviation is 100. The exercise is to choose one of the two SAT exams (math or verbal) or their sum, and test the hypothesis that the mean for the population sampled equals the mean for all students tested. You choose whichever scores you wish to test, you sample whatever and however you think is applicable and you may choose the type of test you think is applicable. You may wish to perform several tests.

Output: Print for each test you perform, the level of significance, one or two sided, the sample size, the sample mean and whether you accept or reject $\mu = 500$.

Questions:

1. Discuss the results of your tests and tell what decision you have reached on the original hypothesis. Discuss your reasons for performing the tests you did.

2. Did you assume that $\sigma = 100$ or that σ is unknown? Explain why.

Extra things to try:

1. Perform the above analysis on the standard deviation of the samples to test whether the population sampled has SAT scores with standard deviation 100. What are your conclusions?

Computer Exercise 40: Testing the Difference of Two Means

Purpose: The purpose of this exercise is to illustrate the test for the difference of two means by an application to SAT scores.

Description: Write a program to test the hypothesis that the mean of SAT math scores is equal to the mean of SAT verbal scores against the alternative that they are not equal. Sample from the population described in computer exercise 39. Perform the test on samples of size N for each score. The program is to read a value of the level of significance, α, and a value of N and perform the test. You may assume the population standard deviation is 100.

Output: Your output should consist of the values of α, N, $\bar{x}_1$, $\bar{x}_2$ and an indication of whether the null hypothesis was accepted or rejected.

Questions:

1. Run your program for $\alpha = 0.05$, N = 10, 20, 30.

2. Comment on the assumption that $\sigma = 100$. Is this reasonable?

3. How would you change your program if you were to test against a one-sided alternative?

Extra things to try:

1. Rewrite your program for the above exercise assuming σ unknown and comment on your results.

Computer Exercise 41: Testing a Proportion

Purpose: The purpose of this experiment is to give the student a better understanding of the technique involved in testing hypotheses about proportions.

Description: The Encyclopedia Britannica states that 0.130 of the letters used in the English language are e's. Write a program which will test this hypothesis on a sample. The program is to cause the computer to read a text of any length from a set of data cards. Any time five consecutive blanks are encountered in the text, the program is to consider the text terminated and should proceed to test the hypothesis that the true proportion of letters that are e's is 0.130 against the alternative that it is not at the 0.05 significance level using the normal approximation to the binomial.

Output: The output of your program should consist of an exact copy of the text under examination, the proportion of letters in the text which were e's and whether the hypothesis was accepted or rejected.

Questions:

1. Could you have tested the hypothesis using the binomial distribution properties alone? Explain how you would go about that.

2. Does your program work for any size text? If not, what is the largest text for which it will work? Can you modify your program so that it works for larger texts?

Extra things to try:

1. Write a program to test the same hypothesis but use only the properties of the binomial distribution.

Computer Exercise 42: Scattergrams

Purpose: The purpose of this exercise is to illustrate the relation between a pair of variables by a scattergram.

Description: Use the computer and subprogram SCAT described in Part 2 of this manual to construct a scattergram of the pair of variables which was assigned to you from the live data. Choose a sample of 50 pairs from the population.

Output: The output will be handled by SCAT.

Questions:

1. Does it appear that there is any relationship between the variables? Can you explain this from what you know about the source of the variables?

2. From the scattergram, would you say the variables are correlated positively or negatively? Sketch a straight line which you feel comes the closest to fitting the points on the scattergram.

3. Observe the value of r printed at the bottom of your scattergram. Does that value seem reasonable?

Extra things to try:

1. Add to your program the computation of the linear correlation coefficient for the entire population from which your sample was chosen. Is the value obtained from the sample a good approximation to this value?

Computer Exercise 43: Correlation

Purpose: The purpose of this exercise is to allow the students to examine scattergrams of various distributions so that they might have some feel for the meaning of a correlation coefficient.

Description: Use the subroutine CORRE described in Part 2 of this manual to choose sample pairs from a bivariate population with given population correlation coefficient. The program should read values of N, the size of sample to be chosen and RO, the population correlation coefficient. It should then use subroutine SCAT to choose sample pairs and print their scattergram.

Output: All output is provided by subroutine SCAT.

Questions:

1. Run your program for N = 50 and RO = -0.9, -0.5, 0.0, 0.5, and 0.9. You may also wish to choose some other values of N and RO and see how the scattergram looks.

2. Note the value of R printed at the bottom of the scattergrams. Why is this value not the same as RO? How are R and RO defined?

3. Use the tables in the back of your text to test the hypothesis that RO = 0 in each case against the two-sided alternative at the 0.05 significance level.

Computer Exercise 44: Regression and Standard Error of Estimate

Purpose: The purpose of this exercise is to illustrate with live data the regression line and standard error of estimate.

Description: For the same data that you used for exercise 42, use the subprogram SCAT to construct a scattergram and its regression line together with its 95% prediction band.

Output: The output will be handled by SCAT.

Questions:

1. Is the value of r which you obtain significant at the 0.05 level? What does it mean for it to be significant?

2. What would be the effect on the prediction band if we increased the sample size? What if we use a significance level of 0.01 instead of 0.05?

3. Count the number of points that lie outside the prediction band. How many would you expect to lie outside?

Computer Exercise 45: Testing the Central Limit Theorem

Purpose: The purpose of this exercise is to test the Central Limit Theorem by means of the chi-square test.

Description: Write a program which chooses M samples of size N from your live data set and computes the mean for each sample. It is to construct a frequency distribution for the sample means with six frequency classes. The class boundaries are to be $\mu-3\sigma/\sqrt{N}$, $\mu-\sigma/\sqrt{N}$, $\mu-1/2*\sigma/\sqrt{N}$, μ, $\mu+1/2*\sigma/\sqrt{N}$, $\mu+\sigma/\sqrt{N}$, and $\mu+3\sigma/\sqrt{N}$ where μ and σ are the mean and standard deviation of the population. The program tests whether the results agree with a theoretical normal distribution with mean μ and standard deviation $\sigma/\sqrt{N}$ by means of chi-square.

<u>Output</u>: The output should consist of M, N, the chi-square value and the probability that a chi-square value exceeds the obtained value by chance. This probability can be obtained by means of subprogram FCHSQ described in Part 2 of this manual. Also print the frequency distribution. Label all of your answers.

<u>Questions</u>:

1. Run your program for the following combinations of values:

   N = 1, M = 30
   N = 1, M = 50
   N = 5, M = 30
   N = 5, M = 50
   N = 30, M = 30
   N = 50, M = 50

2. In each case, decide if you would be able to reject normality at the 0.05 level.

3. What is the effect of increasing N on your test? Explain.

4. What is the effect of increasing M? Explain.

5. What is the connection between this exercise and the Central Limit Theorem?

Extra things to try:

1. Do the above experiment with samples chosen from a normal population with mean 20 and standard deviation 5, testing the distribution of means against a normal distribution with mean 20, standard deviation $5/\sqrt{N}$.

Computer Exercise 46: Testing a Random Number Generator

Purpose: The purpose of this exercise is to test the randomness of a random number generator by a chi-square goodness of fit test.

Description: Write a program which will read a value of N and take a sample of size N from numbers randomly selected between 0 and 1 by the random number generator. A frequency distribution with 10 equal classes is then to be constructed from these N sample values and a chi-square test is to be performed to determine if a number is equally likely to occur in any of the frequency classes. Test this hypothesis at the $\mu = 0.05$ level.

Output: Your output should consist of your value of N, the frequency distribution, the chi-square value obtained, its probability determined by FCHSQ described in Part 2 of this manual, and whether the null hypothesis was accepted or rejected.

Questions:

1. For what values of N is this test not valid? Why?

2. What is the effect on your test of increasing N?

3. What is the effect of increasing the number of frequency classes?

4. Compare your results with those obtained from computer exercise 25. What are you able to do better now?

Extra things to try:

1. Write a program which tests the subroutine NORM1 which generates a random sample from the normal distribution.

Computer Exercise 47: Contingency Tables

Purpose: The purpose of this exercise is to illustrate the use of contingency tables to test the relation of two variables.

Description: Construct a contingency table from the two variables assigned to you from the live data set. Compute the chi-square value from the table and find its probability.

Output: Your output should consist of a labelled printout of the contingency table, the chi-square value computed, and its probability determined by FCHSQ.

Questions:

1. Choose a sample of appropriate size and run the above program. Is there a relation between the two variables at the 0.05 significance level? Interpret your answer.

2. The correlation coefficient also tests the relation between two variables. Could it be used in this situation?

Computer Exercise 48: Testing a Median

Purpose: The purpose of this exercise is to illustrate how to apply the sign test to test the median of a live data set.

Description: In exercise 39 you tested the hypothesis that the mean SAT score is 500. For this exercise use the sign test to test the hypothesis that the median of the scores is 500. Choose the same exam (or the sum of the two) which you chose for exercise 39 and once again you sample whatever and however you feel appropriate.

Output: Print, for each test you perform, the level of significance, one- or two-sided test, the sample size and whether or not you accept the hypothesis that the median is 500.

Questions:

1. Discuss the results of your tests and tell what decision you have reached on the original hypothesis.

2. How do your results compare with your results for exercise 39? Can you explain this?

3. Compare the test used in exercise 39 with the ones used above as far as computational complexity, efficiency, and usefulness of results.

Extra things to try:

1. Use a chi-square test to test the hypothesis that the scores above are normally distributed. How is this pertinent to our choice of tests?

Computer Exercise 49: Testing the Difference of Two Medians

Purpose: The purpose of this exercise is to illustrate the use of the rank-sum test for testing the difference of two medians

Description: Write a program which applies the rank-sum test to test the hypothesis that the median score on the SAT verbal is equal to the median score on the SAT math. Write your program in such a way that it will obtain independent random samples of scores on the two examinations and test the hypothesis at any desired level of significance. You may test in any way you feel appropriate.

<u>Output</u>: Print, for each test you perform, the level of significance, one-or two-sided test, the sample sizes and whether the hypothesis was accepted or rejected.

<u>Questions</u>:

1. Discuss the results of your tests and tell what decision you have reached on the original hypothesis.

2. How does this test compare with testing the difference of two means? What assumptions must one make in order to apply the latter test?

3. How would you attempt to verify that testing the difference of two means would be valid for this problem?

<u>Extra things to try</u>:

1. Write a program which implements the suggestions in your answer to question 3.

Subprograms

Introduction

On the following pages you will find descriptions of how to use some subprograms which are helpful in doing the computer exercises. Each description begins with a general form of the call of the subprogram. Immediately following the general form, all parts of that form which are variable are identified and qualified. They will be qualified as to type (integer or real), as subscripted, if applicable, and as variable or expression. If a portion of the call is identified as a real expression, for example, it may then be replaced in the call by any real variable, real constant, or any other real expression.

The description portion describes what the subprogram does. The notes specify any other information of which the user of the subprogram needs to be made aware.

Not all of these subprograms are necessary for working the exercises in this manual. Those that are not are included because it was felt that they might be useful to you in other aspects of your study and use of statistics.

BINOM

General Form: X=BINOM(N,I,P)

Where:

X is a real variable into which the value returned by BINOM will be stored.

N is an integer expression which specifies the number of trials.

I is an integer expression which specifies the number of successes.

P is a real expression which specifies the probability of success on a single trial.

Description: BINOM computes and returns to X the binomial probability of I successes in N trials given that the probability of success is P on each of the N trials.

Note:

1. BINOM computes the value:

$$\frac{N!}{I!(N-I)!} P^{I}(1-P)^{N-I}$$

.
.
.

BIVNO

<u>General Form</u>: CALL BIVNO (XMEAN,XVAR,YMEAN,YVAR,N,RO,X,Y)

<u>Where</u>:

XMEAN is a real expression which specifies the mean of X.

XVAR is a real expression which specifies the variance of X.

YMEAN is a real expression which specifies the mean of Y.

YVAR is a real expression which specifies the variance of Y.

N is an integer expression which specifies the sample size.

RO is a real expression which contains the value of the correlation coefficient.

X is a real subscripted variable which returns the observed values of X.

Y is a real subscripted variable which returns the observed values of Y.

<u>Description</u>: BIVNO generates a random sample of size N from a bivariate normal distribution with the given parameters. It uses the fact that X has a normal distribution N(XMEAN,XVAR), and given X = x, Y has a conditional normal distribution

$$N(YMEAN + RO(YVAR/XVAR)^{1/2}(x - XMEAN),\ YVAR(1 - RO^2)).$$

<u>Note</u>:

1. BIVNO is called by subroutine CORRE.
2. BIVNO requires subprograms NORM1 and RAN.

COM

General Form: CALL COM(N,K)

Where:

N is an integer expression specifying the number of objects in the set under consideration.

K is an integer expression specifying the number of objects to be chosen from the set at a time.

Description: COM will read N cards, each containing the name of an object in the first eight columns and print a listing of all combinations of those N objects taken K at a time, one combination to a line.

Note:

1. The print out will begin at the top of a new page.
2. N must lie in the range $1 \leq N \leq 12$, and K must be such that $0 \leq K \leq 6$ and $K \leq N$.
3. COM requires subprogram COMB.

COMB

General Form: CALL COMB(N,K,IA,IEND)

Where:

N is an integer expression specifying the number of elements in the set.

K is an integer expression specifying the number of elements in each combination.

IA is an integer subscripted variable specifying inclusion.

IEND is an integer variable which specifies initialization and termination of a sequence of combinations.

Description: COMB generates a sequence of inclusion vectors for combinations of N objects taken K at a time. If IA(I) is 1, the Ith element of the set of objects is included in the combination. If IA(I) is 0, it is not. When COMB is called with IA containing one inclusion vector, it will generate the next inclusion vector in the sequence. If COMB is called with IEND ≥ 1, an initial inclusion vector will be generated and IEND set to zero. If COMB is called with IEND ≤ 0, the next inclusion vector in the sequence is generated. When COMB generates the last inclusion vector in the sequence, it returns the value 1 for IEND.

Note:

1. COMB is called by subroutines COM and PER.
2. IA must be subscripted to at least N.

CONIN

General Form: CALL CONIN(XMIN,XMAX,ENTPL,ENTPR,PARAM)

Where:

XMIN is a real expression whose value is the minimum value expected for the left end point of the confidence intervals.

XMAX is a real expression whose value is the maximum value expected for the right end point of the confidence intervals.

ENTPL is a real subscripted variable containing 50 left end points.

ENTPR is a real subscripted variable containing 50 right end points.

PARAM is a real expression specifying the hypothesized value of the parameter.

<u>Description</u>: CONIN prints 50 intervals whose left end points are ENTPL(1),....,ENTPL(50) and whose corresponding right end points are ENTPR(1),....,ENTPR(50). The intervals are printed in the form of a line of asterisks on 50 consecutive lines. On each line, a minus sign is printed to indicate the location of PARAM relative to the interval. XMIN and XMAX are used to determine the scale. If any ENTPL value is less than XMIN, the scale is redefined to include the smallest left end point. Likewise, the scale is redefined if any ENTPR is larger than XMAX.

<u>Note</u>:

1. The user has the option of defining his own scale, by choice of XMIN or XMAX, or allowing the subroutine to choose it by simply setting both XMIN and XMAX to the same value as PARAM.

2. ENTPL and ENTPR must be subscripted to at least 50.

CORRE

<u>General Form</u>: CALL CORRE(N,A,B,RO)

<u>Where</u>:

N is an integer expression which indicates the size of the samples to be chosen.

A is a real subscripted variable into which the first sample is to be stored.

B is a real subscripted variable into which the second sample is to be stored.

RO is a real expression whose value is the correlation coefficient of the populations from which A and B are chosen.

Description: CORRE generates two samples of size N in subscripted variables A and B from a bivariate population with correlation coefficient RO. The means of the two variables are 50 and the variances are 100.

Note:

1. A and B must be subscripted to at least N.
2. CORRE requires subprograms BIVNO, NORM1, and RAN.

DRAW

General Form: CALL DRAW (IST,IVAL,ISHUF,KARD)

Where:

IST is an integer variable which will be assigned a value between 1 and 4 to represent the suit of the card drawn.

IVAL is an integer variable which will be assigned a value between 1 and 13 to represent the face value of the card drawn.

ISHUF is an integer variable used internally by DRAW.

KARD is an integer subscripted variable used internally by DRAW.

Description: This subroutine simulates a draw of one card from a deck of playing cards. If ISHUF = 0 when DRAW is called, the deck is shuffled and the top card is drawn. If ISHUF is positive but less than 52, ISHUF is incremented by one and the ISHUF-th card is drawn from the deck. IST and IVAL return the values for the suit and face value of the card drawn in the following code:

Suit	Face Value
1 = Heart	1 = Ace
2 = Spade	2-10 = 2-10
3 = Club	11 = Jack
4 = Diamond	12 = Queen
	13 = King

Note:

1. After all 52 cards have been drawn from the deck, that is, when ISHUF is greater than or equal to 52 when DRAW is called, DRAW returns zeros for both IST and IVAL.

2. KARD must be subscripted to at least 52.

3. DRAW requires subprogram RAN.

FCHSQ

General Form: DF = FCHSQ(X,NDF)

Where:

NDF is an integer expression giving the number of degrees of freedom for the chi-square distribution.

X is the point at which the function is being evaluated.

Description: This function subprogram returns to DF the value of the distribution function for the chi-square distribution with NDF degrees of freedom. FCHSQ is evaluated at X.

Note:

1. FCHSQ requires subprogram FNRML.

FF

General Form: DISF = FF(X,NDFN,NDFD)

Where:

DISF is a real variable into which is stored the value of the F distribution function.

NDFN is an integer expression giving the numerator degrees of freedom.

NDFD is an integer expression giving the denominator degrees of freedom.

X is a real expression whose value is the point at which the F distribution function is evaluated.

Description: This function subprogram returns to DISF the value of the distribution function for the F distribution with NDFN and NDFD degrees of freedom evaluated at X.

Note:

1. FF requires subprogram FT.

FLIP

General Form: A = FLIP(X,H,T)

Where:

A is a real variable which will contain the result of the flip of the coin.

X is a real expression which determines the probability of flipping a head.

H is a real expression which is the value assigned to A when a head occurs.

T is a real expression which is the value assigned to A when a tail occurs.

Description: FLIP simulates the flip of a coin and stores the result in A as H if a head and T if a tail. X determines the probability of a head on a given flip. If 0 < X < 1, the probability of a head is X. Otherwise, it is 0.5.

Note:

1. FLIP requires subprogram RAN.

FNRML

General Form: DISF = FNRML(X)

Where:

DISF is a real variable into which the value of the standard normal distribution function will be stored.

X is a real expression whose value is the point at which the function is evaluated.

Description: This function subprogram returns to DISF the value of the distribution function for the standard normal distribution, N(0,1), evaluated at X.

FT

General Form: DISF = FT(T,NDF)

Where:

DISF is a real variable into which is stored the value of the t distribution function.

T is a real expression whose value is the point at which the function is to be evaluated.

NDF is an integer expression whose value is the number of degrees of freedom.

Description: This function subprogram returns to DISF the value of the distribution function for the t distribution with NDF degrees of freedom evaluated at T.

GMMA

General Form: G = GMMA(X)

Where:

G is a real variable into which will be stored the value of the function.

X is a real expression which specifies the point at which the function is to be evaluated.

Description: GMMA computes an important mathematical function called the gamma function. The value of the gamma function at X is stored in G.

Note:

1. The gamma function at any positive integer n is equal to the factorial of n - 1. The gamma function has a value for all real arguments except zero and negative integers.

HIST

General Form: CALL HIST(N,X,ALFT,ALN,NTOT)

Where:

N is an integer expression specifying the number of observations to be included in the histogram.

X is a real subscripted variable containing the observations.

ALFT is a real expression specifying the left end point of the left-most frequency class.

ALN is a real expression which specifies the width of each frequency class.

NTOT is an integer expression specifying the number of frequency classes.

Description: HIST will print a histogram of the data found in the first N locations of X. The frequency classes are determined by choosing NTOT classes of width ALN starting at ALFT.

Note:

1. The user must take care that all of his data points are within the frequency classes he has defined. If he does not know the range of his values, it might be to his advantage to use HIST1. Those values outside the range specified will be ignored.

2. NTOT can be at most 20.

3. X must be subscripted for at least N values.

HIST1

General Form: CALL HIST1(N,X,TOT)

Where:

N is an integer expression specifying the number of observations to be included in the histogram.

X is a real subscripted variable containing the observations.

NTOT is an integer expression specifying the number of frequency classes.

Description: HIST1 will print a histogram of the data found in the first N locations of X. The frequency classes are determined automatically by the subroutine.

Note:

1. NTOT must be no larger than 20.
2. X must be subscripted for at least N values.
3. HIST1 requires subprogram HIST.

ITOSS

General Form: J = ITOSS(X)

Where:

J is an integer variable into which the result of the die toss will be stored.

X is any real expression.

Description: ITOSS simulates the tossing of a fair die and returns the count of the upward face into J. X is a dummy parameter and is not used at all by the function ITOSS.

Note:

ITOSS requires subprogram RAN.

NORM1

General Form: CALL NORM1(N,XMEAN,SIGMA,X)

Where:

N is an integer expression which specifies the number of observations to be generated from the normal distribution.

XMEAN is the mean of the distribution.

SIGMA is the standard deviation of the distribution.

X is a real subscripted variable in which the random sample is stored.

Description: NORM1 selects a random sample of size N from a normal distribution with mean XMEAN and standard deiviation SIGMA and returns the sample values in X.

Note:

1. X must be subscripted to at least N.

2. NORM1 requires subprogram RAN.

PER

General Form: CALL PER(N,K)

Where:

N is an integer expression specifying the number of objects in the set under consideration.

K is an integer expression specifying the number of objects to be chosen from the set at a time.

Description: PER will read N cards, each containing the name of an object in the first eight columns, and print a listing of all permutations of these N objects when taken in all possible combinations of K at a time. One permutation is printed per line and a line is left blank between each pair of combinations.

Note:

1. The printout will begin at the top of a new page.

2. N must lie in the range $1 \leq N \leq 12$, and K must be such that $1 \leq K \leq 6$ and $K \leq N$.

PERM

General Form: CALL PERM(N,IN,IEND)

Where:

N is an integer expression indicating the number of elements in the set.

IN is an integer subscripted variable containing the permutation.

IEND is an integer variable which specifies initialization and termination of a sequence of permutations.

Description: PERM generates a sequence of all permutations of the first N integers. If IN contains one permutation when PERM is called, IN will contain the next permutation in the sequence when a RETURN is executed. If IEND is not zero when PERM is called, an initial permutation is generated and IEND is set to zero. When the last permutation in the sequence is generated, IEND returns with a value of one.

Note:

1. PERM is called by subroutine PER.
2. IN must be subscripted to at least N.

PRCHI

General Form: C = PRCHI(P,NDF)

Where:

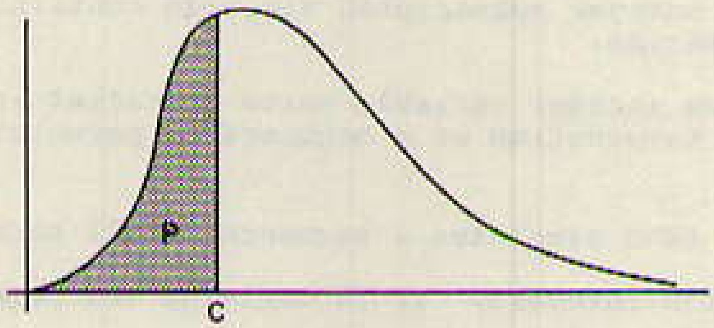
C is a real variable in which the result is stored.

P is a real expression specifying a probability.

NDF is an integer expression which specifies the number of degrees of freedom for a chi-square distribution.

<u>Description</u>: PRCHI computes and returns a number C such that

P(X - C) = P where X has a chi-square distribution with NDF degrees of freedom.



<u>Note</u>:

1. PRCHI requires subprograms FCHSQ and FNRML.

PRF

<u>General Form</u>: C = PRF(P,NDFN,NDFD)

<u>Where</u>:

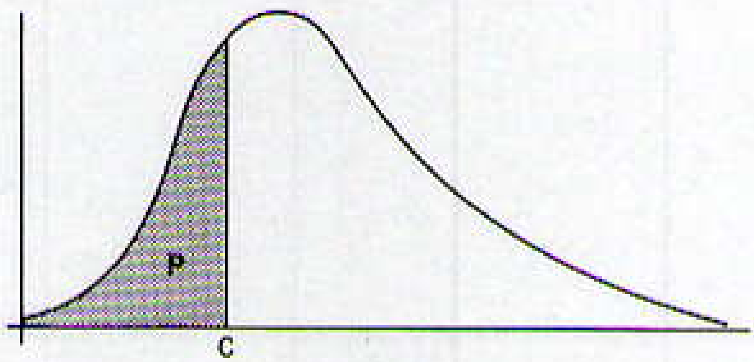
C is a real variable in which the result is stored.

P is a real expression specifying the probability.

NDFN and NDFD are integer expressions which specify the number of degrees of freedom in the numerator and denominator, respectively, for an F distribution.

Description: PRF computes and returns a number C such that

P(X - C) = , where X has an F distribution with NDFN and NDFD degrees of freedom, respectively.



Note:

1. PRF requires subprograms FF and FT.

PRT

General Form: C = PRT(P,NDF)

Where:

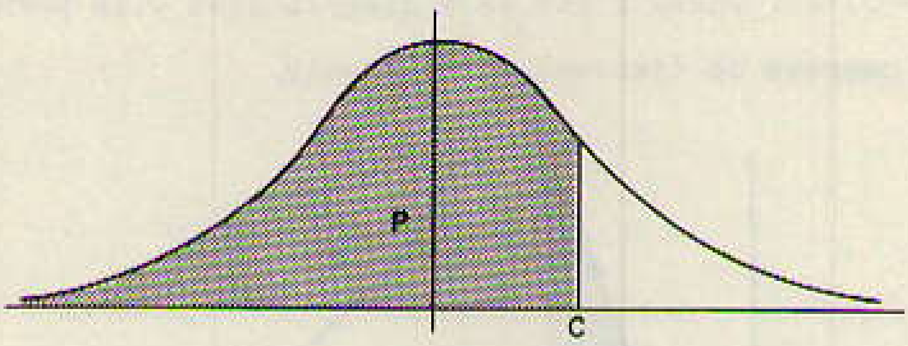
C is a real variable in which the result is stored.

P is a real expression specifying a probability.

NDF is an integer expression which specifies the number of degrees of freedom for the t distribution.

Description: PRT computes and returns a number C such that

P(X - C) = P where X has a t distribution with NDF degrees of freedom.



Note:

1. PRT requires subprogram FT.

PRZ

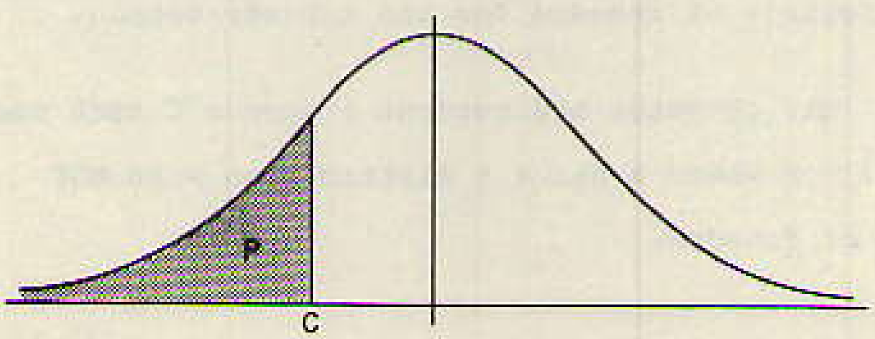
General Form: C = PRZ(P)

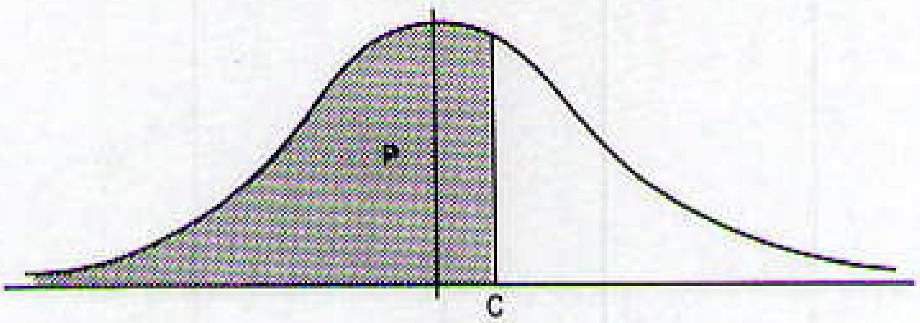
Where:

C is a real variable in which the result is stored.

P is a real expression specifying a probability.

Description: PRZ computes and returns a number C such that P(Z - C) = P where Z has a standard normal distribution, N(0,1).





Note:

1. PRZ requires subprogram FNRML.

RAN

General Form: X = RAN(I)

Where:

X is a real variable in which a random value is stored.

I is an integer expression whose use will depend on your local computer installation.

Description: RAN will generate a sequence of random numbers from the uniform distribution on the interval (0,1); that is, every number between 0 and 1 has an approximately equal chance of being returned.

Note:

1. Since random number generators are very dependent upon the computer used, the details on the use of RAN will be provided by your instructor.

SAMPL

General Form: CALL SAMPL(N,A,M,B,IRPLC)

Where:

N is an integer expression specifying the size of the population from which you are sampling.

A is a real subscripted variable which contains the N values of the population.

M is an integer expression specifying the size of the desired random sample taken from A.

B is a real subscripted variable into which the random sample is returned.

IRPLC is an integer expression which indicates whether sampling is with or without replacement.

Description: SAMPL generates a random sample of size M from a population of N values stored in A. The random sample is returned in B. If IRPLC = 0, then sampling is without replacement. If IRPLC = 1, the sampling is with replacement.

Note:

1. If M is greater than N, then sampling must be with replacement.
2. A must be subscripted to at least N.
   B must be subscripted to at least M.

SCAT

General Form: CALL SCAT (I,XMIN,XMAX,YMIN,YMAX,N,X,Y)

<u>Where</u>:

I is an integer expression which specifies the features desired: I = 1 specifies a scattergram only; I = 2 adds the least squares regression line; I = 3 adds the 95% confidence band.

XMIN is a real expression which specifies the lower limit on the x axis.

XMAX is a real expression which specifies the upper limit on the x axis.

YMIN is a real expression which specifies the lower limit on the y axis.

YMAX is a real expression which specifies the upper limit on the y axis.

N is an integer expression which specifies the number of pairs of observations to be plotted.

X is a real subscripted variable which contains the N observations of X.

Y is a real subscripted variable which contains the N corresponding observations of Y.

<u>Description</u>: Subroutine SCAT prints a scatter diagram of N ordered pairs (x,y). On the scatter diagram, a 1 indicates 1 data point, a 2 indicates 2 data points, a 3 indicates 3 data points, a 4 indicates 4 or more data points. The mean, unbiased variance and maximum likelihood variance are calculated for the observations of X and for the observations of Y. These values are printed along with the correlation coefficient. The least squares regression line is printed when I = 2, where I is the first argument of SCAT. A 95% confidence band for the regression line is printed when I = 3.

Note:

1. The limits on the x and y axes will be redefined within SCAT if the values of X and Y input make this necessary. The redefined values are not returned to the calling program.

2. The order of the elements in X and Y is changed by SCAT. The arrays are returned with the Y values in increasing order.

3. X must be subscripted to at least N.
   Y must be subscripted to at least N.

SMASV

General Form: CALL SMASV(XBAR,SVAR,X,N)

Where:

XBAR is a real variable in which is returned the sample mean.

SVAR is a real variable in which is returned the sample variance.

X is a real subscripted variable containing the values of the sample.

N is an integer expression specifying the size of the sample.

Description: SMASV calculates the sample mean and sample variance of the first N values in X by

$$XBAR = (1/N) \sum_{I=1}^{N} X(I),$$

$$SVAR = (1/N) \sum_{I=1}^{N} (X(I)-XBAR)^2$$

Note:

1. X must be subscripted to at least N.

SUPER

General Form: CALL SUPER(XMIN,XMAX,N,X,XMU,SIGMA)

Where:

XMIN is a real variable which specifies the lower limit on the x axis.

XMAX is a real variable which specifies the upper limit on the x axis.

N is an integer expression which is the number of observations to be included in the histogram.

X is a real subscripted variable containing the N observations.

XMU is a real expression which specifies the mean of the normal distribution.

SIGMA is a real expression which specifies the standard deviation of the normal distribution.

Description: Subroutine SUPER plots a relative frequency histogram with a superimposed normal probability density function. The histogram contains 10 classes.

Note:

1. XMIN and XMAX are redefined if necessary and the redefined values are returned to the calling program.

2. X must be subscripted to at least N.

URN

General Form: CALL URN(I1,I2,I3,N,IREP,K1,K2,K3)

Where:

I1 is an integer expression which specifies the number of balls in the urn of type 1.
I2 is an integer expression which specifies the number of balls in the urn of type 2.

I3 is an integer expression which specifies the number of balls in the urn of type 3.

N is an integer expression which specifies the number of balls to be drawn.

IREP is an integer expression which specifies whether the balls are drawn with or without replacement.

K1 is an integer variable which specifies the number of balls drawn of type 1.

K2 is an integer variable which specifies the number of balls drawn of type 2.

K3 is an integer variable which specifies the number of balls drawn of type 3.

Description: URN simulates the drawing of N balls from an urn containing balls of three distinguishable types; I1 of type 1, I2 of type 2, and I3 of type 3. The result is that K1 balls are drawn of type 1, K2 of type 2 and K3 of type 3. If IREP is zero, drawing is with replacement; if IREP is one, drawing is without replacement.

Note:

1. No values other than zero and one are allowable for IREP.

2. If N > I1 + I2 + I3 and IREP is one, then only I1 + I2 + I3 balls will be drawn.

Data Sets

Introduction

Several of the exercises in this manual require that analysis be carried out with live data sets. In order to do this such data sets must be accessible on your computer. Five such data sets are included with this manual and your instructor may wish to make others available in addition to or instead of these. In this part of the manual we will describe these five data sets and show you how they can be accessed.

These data sets could be made available to your program by means of a deck of cards which would be input to your program among your data cards. However, since these data sets are rather large, this would result in a rather bulky deck which would have to be made available to everyone in the class. In addition in order to read a card near the end of the deck, all preceding cards must be read first. Instead, it is recommended that these data sets be stored on a random access device, such as a magnetic disk, which is connected to the computer. This allows one copy of the data set to be available to all users, and makes it possible for them to access it quickly and easily.

Because different computers differ in the way they handle random access files, there will probably be cards which you need to include in your deck when using this data which are not described in this manual. We will, however, describe a subprogram which will enable you to conveniently access such information.

REDE

General Form: CALL REDE (NDTST,NVAR,SAMP,KLO,KHI)

Where:

NDTST is an integer expression which specifies the number of the data set to be accessed.

NVAR is an integer expression which specifies the number of the variable to be accessed.

SAMP is a real subscripted variable in which the data will be placed.

KLO is an integer expression which specifies the number of the first record to be accessed within the data set.

KHI is an integer expression which specifies the number of the last record to be accessed within the data set.

Description: The data values of variable NVAR of subjects KLO through KHI in data set NDTST will be read from the random access file and stored in the first KHI-KLO+1 locations of SAMP.

Note:

1. If KLO is input as zero, the entire data set will be read into SAMP.

2. SAMP must be subscripted to at least KHI-KLO+1.

Examples:

CALL REDE (1,3,A,6,25)

This call will read into A(1) through A(20) the values of variable 3 for subjects 6 through 20 of data set 1.

CALL REDE (3,5,SAM,28,28)

This call will read into SAM(1) the value of variable 5 for subject 28 of data set 3.

CALL REDE (2,3,VEC,0,0)

This call will read into VEC(1) through VEC(220) all values of variable 3 of data set 2.

DATA SET 1

Calculus Students

The following data was collected in order to determine what factors significantly predict success in Freshman Calculus at a large university. Each subject is a student who completed the first calculus course at this large university.

Number of variables: 6

Number of subjects: 364

Description of variables:

Variable 1. High school GPA. The subject's final high school grade point average to the nearest tenth. (4.0 perfect).

Variable 2. ACT Math. The subject's score on the mathematics portion of the ACT college entrance exam.

Variable 3. ACT Comp. The subject's composite score on the ACT exam.

Variable 4. Sex. The subject's sex, coded as 1 for male and 0 for female.

Variable 5. Final Exam. The subject's score on the final exam in the calculus course.

Variable 6. Final Grade. The subject's final course grade in calculus coded as 0 for F, 1 for D, 2 for C, 3 for B, and 4 for A.

DATA SET 2

Metropolitan Areas

The data in this set consists of specific economic data collected for the metropolitan areas of cities of the United States. This data is found in <u>Statistical Abstract of the United States, 1970</u>. Each subject is a metropolitan area in the United States.

Number of variables: 5

Number of subjects: 220

Description of variables:

Variable 1. Popul. The population of the subject in units of 10,000.

Variable 2. Income. The subject's personal income per capita, in dollars.

Variable 3. Tax. The property tax per capita collected in the subject, in dollars.

Variable 4. Deposits. The amount of bank deposits per capita, in the subject, in dollars.

Variable 5. Sales. The amount of total sales per year per capita in the subject, in dollars.

DATA SET 3

Programming Students

The set consists of data collected on students enrolled in an introductory programming course. Each subject is a student who has completed this course.

Number of variables: 6

Number of subjects: 152

Description of variables:

Variable 1. Subject's sex (1 = male, 0 = female)

Variable 2. Subject's class (1 = freshman, 2 = sophomore, 3 = junior, 4 = senior)

Variable 3. Row in which subject sat in the class (1-4)

Variable 4. Percentage of total points subject earned on the homework (could be more than 100)

Variable 5. Subject's final exam score (200 points possible)

Variable 6. Subject's final grade in the course (A = 4.0, A- = 3.7, B+ = 3.3, B = 3.0,etc.)

DATA SET 4

Liberal Arts Colleges

This data was collected from the <u>College Blue Book</u>, <u>1969/70</u>. Each subject is a liberal arts college in the United States.

Number of variables: 6

Number of subjects: 319

Description of variables:

Variable 1. Type of school (0 = Protestant, 1 = Catholic, 2 = Private, 3 = State)

Variable 2. Tuition for one year in dollars.

Variable 3. Total yearly cost per student for tuition, fees, room, and board in dollars.

Variable 4. Subject's total enrollment.

Variable 5. Subject's faculty-student ratio.

Variable 6. Number of volumes in subject's library, in thousands.

DATA SET 5

SAT Scores

This data set is a collection of SAT examination scores for a set of students entering college. Each subject is an entering student.

Number of variables: 2

Number of subjects: 452

Description of variables:

Variable 1. SAT Math. The score received on the mathematics portion of the Scholastic Aptitude Test given by the College Entrance Examination Board.

Variable 2. SAT Verbal. The score received on the verbal portion of the Scholastic Aptitude Test given by the College Entrance Examination Board.