P-STAT®

Advanced Statistics

P-STAT®

P-STAT: Advanced Statistics

Second edition January 2013

This publication corresponds to **P-STAT Release 3.01, January 2013**. This publication documents the P-STAT commands for advanced statistics.

Please direct any questions to:

P-STAT, Inc.
230 Lambertville-Hopewell Rd.
Hopewell, New Jersey 08525-2809
U.S.A.

Telephone: 609-466-9200

Fax: 609-466-1688

Internet: support@pstat.com

Web Page URL: http://www.pstat.com

CONTENTS

Matrix Operations

CANONICAL.COR: Multivariate Correlation

Q.CLUSTER: Clustering Cases

DISCRIM: Discriminant Analysis

FACTOR: Factor Analysis and Rotation

ANOVA: Analysis of Variance

SURVIVAL: Survivorship Tables and Functions

FIGURES

1
Matrix Operations

A matrix is mathematically defined as an array of numbers. In P-STAT, it is a file of numeric values. The term matrix is used when the operation being performed is mathematical in nature. File is used as a generic term for the input and output data.

The matrix commands in P-STAT perform statistical and mathematical operations on data files. These commands manipulate files by simple element addition, multiplication, subtraction and division. They also perform more complex matrix operations such as multiplying matrices, finding the inverse and the determinant of a matrix, doing a Cholesky triangular decomposition, computing eigenroots and eigenvectors, and joining related correlation matrices. Files may also be transposed and normalized through the use of matrix operations.

1.1 SIMPLE MATRIX OPERATIONS

Transposing or “flipping” a file, so that the variables become the cases and the cases become the variables, is useful for reorganizational purposes as well as mathematical ones. Similarly, arithmetic operations on the elements or individual values of a matrix often serve simple data modification needs. These basic matrix operations use the TRANSPOSE, E.ADD, E.SUBTRACT, E.MULTIPLY and E.DIVIDE commands. The NORM.COL and NORM.ROW commands normalize the columns and rows of matrices.

1.2 Transposing a Matrix

The transpose of a matrix is obtained by interchanging its rows and columns. Both the TRANSPOSE and C.TRANSPOSE commands interchange rows and columns — TRANSPOSE produces an output file of only numeric variables, whereas C.TRANSPOSE produces a file of only character variables. TRANSPOSE is used for mathematically transposing or reorganizing files of purely numeric data; it is described in this chapter. C.TRANSPOSE is used for reorganization of files of numeric and/or character data.

The TRANSPOSE command requires the name of the input file and the identifier OUT followed by an output file name. Figure 1.1 illustrates transposing File B to create an output file named BTran. The rows become the columns and the columns become the rows. If the transpose of a matrix is the same as the original matrix, then the matrix is symmetric.

Figure 1.1 Transposing a Matrix

```
                                 TRANSPOSE  B,  OUT  BTran  $
                                 LIST  BTran  $

  File B:                      File BTran:

                                                     row
  S100  S200  S300               ROW1  ROW2  ROW3  ROW4  label

     1     -     0                  1     4     0     6  S100
     4     3     8                  -     3     5     6  S200
     0     5     2                  0     8     2     1  S300
     6     6     1
```

TRANSPOSE may be useful when input data is organized by variable rather than by case. If a file of numeric data has been built with the variables as the *rows* and the cases as the *columns*, TRANSPOSE rearranges the file into the case-by-variable format that P-STAT commands expect. Before the file is transposed, any character variables in it are omitted. The file to be transposed cannot contain more rows than the number of variables allowed in the P-STAT size being used. In Jumbo, the most commonly distributed size, the limit is 3,000.

If there are many variables in the transposed file, it may be helpful to supply more meaningful names for the transposed variables. An easy way to supply new variable names is to create a file with the desired names and no data, and then to concatenate that file with the transposed file using the concatenate operator (+):

```
MAKE  Fnames,  VARS  Age  Sex  Race  Education  Id:C ;
  $
MODIFY  Fnames + BTran,  OUT  BNamed  $
```

The file containing the variable names must have the same number of names and be of the same data types as the transposed file.

TRANSPOSE is also useful for examining a fairly large number of variables over a few cases. Following a series of data modifications, it may be desirable to check the results of a few cases to be sure that the modifications were completed as expected. Since more rows than columns fit on a terminal screen or a page of printer output, transposing the modified file often shows complete cases:

```
MODIFY  File   [ data modifications ],
  OUT   NewFile   $

TRANSPOSE  NewFile  [ CASES 1 TO 10 ],  OUT  TranFile  $
LIST  TranFile  $
```

Thus, a number of variables may be checked at one time. If the file contains character data, use the C.TRANSPOSE command instead.

1.3 Element Operations

There are four commands that perform element arithmetic on a pair of matrices. They are:

- E.ADD,
- E.SUBTRACT,
- E.MULTIPLY, and
- E.DIVIDE.

There are three requirements for each of these element operations:

1. the name of the first input file,
2. the name of the second input file, and
3. the identifier OUT followed by a name for the output file.

The number of rows and columns in the two input files must be the same. As a result, the number of rows and columns in the output file is the same as in the input files.

E.ADD is used to add each element of one matrix to the corresponding element of a second matrix. An output file of the sums is the result of this operation. Figure 1.2 illustrates the E.ADD command and its output. The input files are A and B. Each element in the output file AB is the sum of the corresponding elements in A and B. Since there is a missing value in the first row, second column of B, there is a missing value in the first row, second column of file AB.

Figure 1.2 **Matrix Operations: Adding the Elements**

```
File A:                  File B:

Var1  Var2  Var3           Var1  Var2  Var3

   1     4     7              1     -     0
   3     9     8              4     3     8
   2     6     4              0     5     2
   7     3     3              6     6     1


E.ADD  A  B,  OUT  AB  $
LIST   AB  $

Var1  Var2  Var3

   2     -     7
   7    12    16
   2    11     6
  13     9     4
```

The E.MULTIPLY command is similar to E.ADD. E.MULTIPLY performs element multiplication. Note that the *order* of the input files in both E.ADD and E.MULTIPLY is not significant, although the *number* of rows and columns must be the same in each of the input files.

The E.DIVIDE command performs element division. Each element of the first matrix is divided by the corresponding element in the second matrix. This is illustrated in Figure 1.3. The E.SUBTRACT command is similar to E.DIVIDE. E.SUBTRACT performs element subtraction whereby each element of the second matrix is subtracted from the corresponding element of the first matrix.

The rules and the identifiers for E.DIVIDE and E.SUBTRACT are the same as they are for element addition and multiplication. However, when E.DIVIDE and E.SUBTRACT are used, the *order* of the input files is significant. E.DIVIDE divides the first file by the second file; E.SUBTRACT subtracts the second file from the first file.

Figure 1.3 **Matrix Operations: Dividing the Elements**

```
E.DIVIDE  B  A,  OUT  BA  $

LIST  BA    $

 Var1        Var2        Var3

1.000       -           0.
1.333       0.333       1.000
0.          0.833       0.500
0.857       2.000       0.333
```

1.4 Normalizing Columns or Rows

Normalization is the process of scaling the rows (or the columns) in a matrix by dividing each row’s elements by the square root of the sum of the squares of the row. The result is a file with rows (or columns) such that the sum of the squares of each row is one. NORM.COL is used to normalize the columns of a matrix. NORM.ROW is used to normalize the rows. The input files may be any size, but missing data are *not* allowed. NORM.COL takes two passes through the data, whereas NORM.ROW only takes one. Both NORM.COL and NORM.ROW require an input file and the identifier OUT followed by an output file name.

Figure 1.4 illustrates the use of NORM.ROW. File A, shown in Figure 1.2, is input to NORM.ROW. The resultant output file has normalized rows. For illustrative purposes, a MODIFY is done to generate a new variable that is the sum of the squared normalized values. By definition of the normalization process, this sum is one.

Figure 1.4 Normalizing the Rows

```
      NORM.ROW  A,  OUT  NormRow   $


.     MODIFY NormRow [ GENERATE  SSQ = 0;

         /*  Loop through the three (now) normalized variables for
         each row squaring the value and adding it the new
         variable SSQ.  These normalized values should sum
         to one /*

              DO #J using Var1 TO Var3;
              SET SSQ = SSQ + V(#J) * V(#J);
              ENDDO ],
       OUT NormSum $


      LIST    NormSum  $

          Var1        Var2        Var3         SSQ

      0.123091    0.492366    0.861640    1.000000
      0.241747    0.725241    0.644658    1.000000
      0.267261    0.801784    0.534522    1.000000
      0.855186    0.366508    0.366508    1.000000
```

A knowledge of PPL (the P-STAT Programming Language is very important if you are using the matrix operators to produce statistics that are not available as commands in P-STAT. The manual “P-STAT: A Guide to the P-STAT Programming Language (PPL), Macros and Textwriter” provides complete PPL documentation. Figure 1.4 uses a DO loop to process all the variables in a single row of data. The sequence when working across the rows is easier because all of the data are in memory at the same time. To compute the sum of squares across the rows after a NORM.COL command requires saving the cumulative values for each variable across all the rows.

Because we are working with numeric data, it is easy to use the P (Permanent) vector to hold the intermediate calculations. Figure 1.5 illustrates NORM.COL and the MODIFY procedure to calculate the sum of squares across the rows.

Figure 1.5 Normalizing the Columns

```
  NORM.COL  A,  OUT  NormCOL

  NORM.COL completed. 4 rows processed.

  MODIFY NormCol [
         /* First DO Loop to create 3 new variables */
           DO #J = 1, 3;
           GEN ? ( 'Cum.SSQ.' & ) = 0;
           ENDDO ;

         /* Second DO loop to process each value in turn */
           DO #J = 1, 3;
         /* Initialize P vector only before first case */
           IF FIRST ( .FILE. )  SET P(#J) = 0;

         /* Now add the product into the P vector, store back
            into the new variable */
           SET P(#J) = P(#J) + V(#J) * V(#J);
           SET V(#J+3) = P(#J);
           ENDDO ],
      OUT NormCSum $


  LIST NormCSum, PLACES 4 $

                             Cum SSQ   Cum SSQ   Cum SSQ
  VAR1      VAR2      VAR3      VAR4      VAR5      VAR6

0.1260    0.3357    0.5959    0.0159    0.1127    0.3551
0.3780    0.7553    0.6810    0.1587    0.6831    0.8188
0.2520    0.5035    0.3405    0.2222    0.9366    0.9348
0.8819    0.2518    0.2554    1.0000    1.0000    1.0000
```

1.5 COMPLEX MATRIX OPERATIONS

Some matrix operations and their products may be useful in conjunction with correlation and multivariate regression analysis. More complex matrix operations available in P-STAT include:

- multiplication of matrices,
- inversion of matrices,
- production of partial, multiple and squared multiple correlations of matrices,
- calculation of the determinant of matrices,
- Cholesky triangular decomposition of positive semi-definite matrices, and
- calculation of eigenroots and eigenvectors.

In addition, matrices may be joined, providing the option to examine the output of many correlation matrices in one file.

1.6 Matrix Multiplication

The command for matrix multiplication is MULTIPLY. (The commands MULTIPLY and E.MULTIPLY do not perform the same function; they are not interchangeable). MULTIPLY requires the names of two input files and the OUT identifier followed by an output file name. Figure 1.6 illustrates the use of the MULTIPLY command. In order to multiply two matrices, the number of rows in the second matrix *must equal* the number of columns in the first matrix. The product of the rows and columns of the second input file cannot exceed 100,000.

Matrix multiplication is done in the order that the files are input to MULTIPLY. In Figure 1.6, the first file C is multiplied by the second file D. The number of rows in the second file must equal the number of columns in the first matrix. Multiplication of matrices is not commutative and therefore, the order of matrix multiplication is of significance. One exception is the multiplication of a matrix and its inverse, which yields the identity matrix regardless of multiplication order.

Figure 1.6 Matrix Multiplication

```
File  C:               File  D:

Var1 Var2              Var1  Var2  Var3  Var4

    1    5                 1     7     1     5
    7    2                 9     1     8     3
    0    4
    8    7


MULTIPLY  C  D,  OUT  CD  $
LIST  $


Var1  Var2  Var3  Var4

  46    12    41    20
  25    51    23    41
  36     4    32    12
  71    63    64    61
```

1.7 Inverse of a Matrix

The inverse of a file is the matrix that, when multiplied by the original matrix, yields an identity matrix. An identity matrix is a square matrix — a matrix with the same number of rows as columns, with ones on the diagonal and zeros elsewhere. (The identity matrix plays, in matrix algebra, the role corresponding to the integer 1 in scalar algebra.) Only square matrices have inverses, and the inverses of some square matrices may not exist. The inverses of certain square matrices do not exist because unique solutions to the equations for their matrix elements do not exist.

The INVERT command requires an input file and the identifier OUT followed by an output file name. The input file to INVERT *must be square* and it cannot contain any missing data. Further output matrices may be produced through the use of certain optional identifiers in place of OUT. For example, the identifier DET produces a scalar (a file with one number) that is the value of the determinant of that matrix:

```
    INVERT  File,  DET  FileDet  $
```

DET, like OUT, must be followed by an output file name.

Two other optional identifiers that may be used with INVERT are PAR and RSQ. When either of these identifiers are used, the input matrix *must be symmetric*. PAR produces a matrix with multiple correlations (not squared) in the diagonal and partial correlations elsewhere. The multiple correlations (diagonal values) are measures of association between that variable and an optimal linear combination of the other variables. The partial correlations (off-diagonal values) are the correlations of the two variables that define that column and row, with the common influence of the other additional variables removed. In other words, if the other variables are used to predict first the column variable and then the row variable, the correlation of the residuals from these two regressions is the partial correlation of the column and row variables.

RSQ produces a matrix similar to the input matrix except that the diagonal contains the squared multiple correlations. The squared multiple correlations are the proportion of the variance in that variable explained by an optimal linear combination of the other variables. This file is sometimes input to the FACTOR command in lieu of a traditional correlation matrix.

INVERT, used with the identifiers OUT or DET, supports asymmetric input matrices. However, when PAR or RSQ are used, input matrices must be symmetric. Note that the maximum number of rows (or columns) permitted in INVERT is 220.

One application of the INVERT command that uses an asymmetric input matrix is solving simultaneous equations. Given these three equations, for example:

```
2x -   y + 4z  =   30.5
 x + .5y - 2z  =  -17.25
7x +  2y -  z  =  -2
```

the matrix algebra solution is to solve for the unknowns by multiplying the inverse of the matrix of coefficients by the matrix of constants. These commands produce the solution:

```
MAKE  A,  NV 3 ;
  2 -1 4 / 1 .5 -2 / 7 2 -1  $
MAKE  C,  NV 1 ;
  30.5 / -17.25 / -2         $

INVERT    A,  OUT  AI        $
MULTIPLY  AI  C,  OUT  XYZ   $
```

File A must be a nonsingular matrix. Listing file XYZ displays the solution set:

```
LIST  XYZ  $

  VAR1

  -1.0
   7.5
  10.0
```

1.8 Cholesky Decomposition, Eigenroots and Eigenvectors

The CHOLESKY command does a Cholesky triangular decomposition of a positive semi-definite symmetric matrix. The output file is an upper triangle (U), such that the transpose of the upper triangle multiplied by the upper triangle equals the input file (B):

```
 U' * U  =  B
```

CHOLESKY requires the name of the input matrix, which must not have any missing data, and a name for an output file:

```
      CHOLESKY  File5,  OUT  File5C  $
```

Only the lower triangle of the input file is used. However, if the input matrix is not exactly symmetric, that fact is reported The output file contains nonzero values in the upper triangle and zeros in the lower triangle. As a check on computations, the lower triangle of the input file is recreated by multiplying the transpose of the upper triangle of the output file by the upper triangle (see the prior equation). The largest absolute difference is reported. Note that it is not possible to decompose some matrices.

The EIGEN command computes the eigenroots (R) and eigenvectors (V) of a symmetric matrix. The vectors multiplied by the roots multiplied by the transpose of the vectors equals the input file (B):

```
      B  =  V * R * V'
```

The output file of eigenvectors also has the properties that the transpose of the vectors multiplied by the vectors equals the identity matrix (all ones on the diagonal and zeros off the diagonal); that is, the transpose of the vectors equals the inverse of the vectors:

$$V' * V = I$$
$$V' = V^{-1}$$

EIGEN requires the name of the input matrix, which must not have any missing data, and ROOTS and/or VECTORS followed by names for the output files of eigenroots and/or eigenvectors:

```
      EIGEN  File5,  ROOTS  File5R,  VECTORS  File5V  $
```

As is the case with CHOLESKY, EIGEN uses only the lower triangle of the input file. However, if the input matrix is not exactly symmetric, that fact is reported. Upon completion, the EIGEN command reports the input diagonal sum, the eigenroot sum and the eigenroot product (which is the determinant). As a check on computations, the roots and vectors are used to recreate the lower triangle of the input matrix (see the prior equation with B). The largest absolute difference is reported. It is not possible to compute eigenroots and eigenvectors for some matrices.

1.9 Joining Correlation Matrices

The SJOIN command may be used to join separate correlation matrices to form one complete correlation matrix that summarizes relationships between all variables in the file. Often the original correlation matrices are computed by different commands. This is illustrated in Figure 1.7.

The CORRELATE command is used to obtain a square matrix of correlations between the *continuous* variables. The BISERIAL command is used to obtain a rectangular matrix of correlations between the continuous and the *dichotomous* variables. BISERIAL is used to correlate continuous with dichotomous variables when the variables underlying the dichotomies are actually continuous and normally distributed. The TET command is used to obtain a square matrix of correlations between the *dichotomous* variables. TET is used to correlate dichotomous variables when the variables underlying the dichotomies are normally distributed. (See the CORRELATE chapter for additional information on the various types of correlations and their appropriate usages.)

SJOIN joins the separate correlation matrices together, producing one matrix that summarizes the correlations between all the numeric variables in the file. SJOIN requires three input matrices, specified by the identifiers A11, A12 and A22 followed by input file names, and the identifier OUT followed by an output file name. The input matrices have certain restrictions on their form. SJOIN joins matrices such that the output matrix has a symmetric matrix in the upper left (A11), an asymmetric matrix in the upper right (A12), and a symmetric matrix in the lower right (A22). The transpose of the upper right is used to fill the lower left so that the output matrix is symmetric.

Note that the row and column labels of individual files must correspond. If they do not, SJOIN is not able to join them. Thus, the row labels of A11 and A12 must be the same, and the column labels of A12 and A22 must be the same. The size of the output file (rows * columns) cannot exceed 85,000.

Figure 1.7 Joining Correlation Matrices with SJOIN

```
      CORRELATE  Car [ KEEP  Mpg  Displacement  Horsepower ],  OUT  CarR  $
      LIST  $

        Mpg  Displacement  Horsepower  row label

   1.000000     -0.804177   -0.778427  Mpg
  -0.804177      1.000000    0.898397  Displacement
  -0.778427      0.898397    1.000000  Horsepower
```

```
      BISERIAL  Car
         [ KEEP  Mpg  Displacement  Horsepower   Wt.Gps  Acc.Gps ],
        ZERO  1,  NCV  3,  OUT  CarBi  $

      LIST   $

     Wt Gps     Acc Gps  row label

  -0.891338    0.397860  Mpg
   0.881943   -0.503336  Displacement
   0.781443   -0.652157  Horsepower
```

```
      TET  Car
         ( KEEP  Wt.Gps  Acc.Gps ),  ZERO 1,  OUT  CarTet  $
      LIST  $
     Wt Gps     Acc Gps  row label

   1.000000   -0.279397  Wt.Gps
  -0.279397    1.000000  Acc.Gps
```

```
      SJOIN,  A11  CarR,  A12  CarBi,  A22  CarTet,  OUT  CarCor  $
      LIST  $

        Mpg  Displacement  Horsepower      Wt Gps     Acc Gps  row label

   1.000000     -0.804177   -0.778427   -0.891338    0.397860  Mpg
  -0.804177      1.000000    0.898397    0.881943   -0.503336  Displacement
  -0.778427      0.898397    1.000000    0.781443   -0.652157  Horsepower
  -0.891338      0.881943    0.781443    1.000000   -0.279397  Wt.Gps
   0.397860     -0.503336   -0.652157   -0.279397    1.000000  Acc.Gps
```

SUMMARY

CHOLESKY

```
 CHOLESKY  File5,  OUT  File5C  $
```

The CHOLESKY command does a Cholesky triangular decomposition of a positive semi-definite symmetric matrix.

Required:

CHOLESKY fn

gives the name of the input file, which must be a positive semi-definite symmetric matrix. Only the lower triangle of the input file is used.

Required Identifiers:

OUT fn

provides a name for the output file, which is an upper triangle, such that the transpose of the upper triangle multiplied by the upper triangle equals the input file.

EIGEN

```
 EIGEN  File5,  ROOTS  File5R,  VECTORS  File5V  $
```

The EIGEN command computes the eigenroots and eigenvectors of a symmetric matrix. The vectors multiplied by the roots multiplied by the transpose of the vectors equals the input file.

Required:

EIGEN fn

gives the name of the input file, which must be a square symmetric matrix. Only the lower triangle of the input file is used.

Optional Identifiers:

ROOTS fn

provides a name for the output file of eigenroots. The diagonal values are the eigenroots and the off-diagonal values are zeros.

VECTORS fn

provides a name for the output file of eigenvectors.

E.ADD

```
E.ADD  A1  A2,  OUT  A12  $
```

The E.ADD command adds corresponding elements of matrices and produces an output matrix of sums.

Required:

E.ADD **fn fn**

supplies the names of two input matrices whose corresponding elements are added together.

Required Identifiers:

OUT **fn**

provides a name for an output matrix containing the sums.

E.DIVIDE

```
 E.DIVIDE  D1  D2,  OUT  D12  $
```

The E.DIVIDE command performs element division. Each element in the first matrix is divided by the corresponding element in the second matrix.

Required:

E.DIVIDE **fn fn**

supplies the names of two input matrices.

Required Identifiers:

OUT **fn**

provides a name for an output matrix containing the quotients.

E.MULTIPLY

```
 E.MULTIPLY  M1  M2,  OUT  M12  $
```

The E.MULTIPLY command multiplies corresponding elements of matrices and produces an output matrix of products.

Required:

E.MULTIPLY **fn fn**

supplies the names of two input matrices whose corresponding elements are multiplied together.

Required Identifiers:

OUT **fn**

provides a name for the output matrix containing the products.

E.SUBTRACT

```
 E.SUBTRACT  S1  S2,  OUT  S12  $
```

The E.SUBTRACT command performs element subtractions. The elements in the second file are subtracted from the corresponding elements in the first file.

Required:

E.SUBTRACT fn fn

supplies the names of two input matrices.

Required Identifiers:

OUT fn

provides a name for the output matrix of differences.

INVERT

```
 INVERT  F1,  OUT  Inverse  $
```

The INVERT command produces a matrix that, when multiplied by the original matrix, yields an identity matrix. The inverse of some matrices may not be calculable because there may not be unique solutions to the equations defining the elements of the inverse.

Required:

INVERT fn

supplies the name of the input matrix that must be a square matrix.

Optional Identifiers:

DET fn

requests and provides a name for a 1-by-1 matrix containing the value of the determinant of the input matrix.

OUT fn

provides a name for the output matrix that is the inverse of the input file.

PAR fn

requests and provides a name for an output file with multiple correlations in the diagonal and partial correlations elsewhere. The input file must be symmetric when PAR is used.

RSQ fn

requests and provides a name for an output file identical to the input file, except that the diagonal contains squared multiple correlations. The input file must be symmetric when RSQ is used.

MULTIPLY

```
MULTIPLY  P1  P2,   OUT  P1P2  $
```

The MULTIPLY command does matrix multiplication. Multiplication of matrices is *not* commutative, and therefore the order of specification of the input files determines the multiplication procedure.

Required:

MULTIPLY fn fn

supplies the names of two input matrices that are matrix multiplied. The matrices must be *conformable* to multiplication; that is, the number of rows in the second matrix must equal the number of columns in the first matrix. The first matrix is postmultiplied by the second matrix. The second matrix must fit into memory.

Required Identifiers:

OUT fn

provides a name for an output file containing the result of matrix multiplication of the first file by the second file.

NORM.COL

```
NORM.COL  F1,  OUT  NormCol  $
```

The NORM.COL command normalizes the columns of a matrix, such that the sum of the squares of the values in a column equals one.

Required:

NORM.COL fn

supplies the name of the input matrix. Missing data are not permitted.

Required Identifiers:

OUT fn

provides a name for the output matrix with normalized columns.

NORM.ROW

```
NORM.ROW  F1,  OUT  NormRow  $
```

The NORM.ROW command normalizes the rows of a matrix, such that the sum of the squares of the values in a row equals one.

Required:

NORM.ROW fn

supplies the name of the input matrix. Missing data are not permitted.

Required Identifiers:

OUT fn

provides a name for the output matrix containing normalized rows.

SJOIN

```
 SJOIN,  A11  UpLeft,    A12  UpRight,
         A22  LowRight,  OUT  SymOut  $
```

SJOIN joins a symmetric matrix in the upper left, an asymmetric matrix in the upper right, and a symmetric matrix in the lower right. The lower left is filled in with the transpose of the upper right so that the output is symmetric. It is typically used to join different types of correlation matrices.

Required Identifiers:

A11 fn

supplies the name of a symmetric matrix for the upper left.

A12 fn

supplies the name of an asymmetric matrix for the upper right. The row labels must match those of A11.

A22 fn

supplies the name of a symmetric matrix for the lower right. The column labels must match those of A12.

OUT fn

provides an output file name.

TRANSPOSE

```
 TRANSPOSE  F1,  OUT  Transp  $
```

The TRANSPOSE command rotates a matrix so that the rows become the columns and the columns become the rows. This may be desirable when data has been collected in a variable-by-case arrangement rather than the typical case-by-variable one. Use the C.TRANSPOSE command if the input file contains any character data.

Required:

TRANSPOSE fn

supplies the name of the input matrix to be transposed.

Required Identifiers:

OUT fn

provides a name for the transposed output file.

C.TRANSPOSE

```
 C.TRANSPOSE  F2,  OUT  Transp2  $
```

The C.TRANSPOSE command rotates a P-STAT system file and produces an output file containing *character* representations of all the data in the original file. In the transposed file, the variables (columns) are Variable, Case.1, Case.2, Case.3 and so on. The cases (rows) are the names and values of all of the variables.

C.TRANSPOSE is especially useful for examining the first 10 or so cases in a file, after modifications to the data are made, in a concise printout. Using FOLD in the LIST command keeps the printout concise when some of the variables have wide character values. The output file from C.TRANSPOSE is not suitable for matrix operations, even though the file is transposed — *all* values, *including numeric ones*, are in character form.

Required:

C.TRANSPOSE fn

supplies the name of the input file to be transposed.

Required Identifiers:

OUT fn

provides a name for the transposed output file of character data.

Optional Identifiers:

COMMAS

requests that commas be inserted in (formerly) numeric values.

2
CANONICAL.COR:
Multivariate Correlation

The CANONICAL.COR command performs *canonical correlation* to analyze the relationships between two sets or groups of variables. Canonical correlations between linear combinations of variables in each set are calculated and tested for significance. Standardized and raw canonical coefficients and correlations between the canonical variables and the original variables (structure coefficients) are produced. Output files of these statistics, the correlations between the original variables and the canonical variables may be produced.

2.1 BACKGROUND

Canonical correlation, developed by Hotelling in 1936, analyzes the relationships between two sets of continuous variables. One set may be considered the dependent or criteria variables and the other set the independent or predictor variables, but computationally this distinction between the two sets does not matter. Canonical correlation is an extension of multiple regression, which analyzes the relationships between one dependent variable and multiple independent variables, to multiple dependent and independent variables.

In multiple regression, a linear combination of the independent variables that has maximum correlation with the single dependent variable is sought. In canonical correlation, a linear combination of the independent variables that has maximum correlation with a linear combination of the dependent variables is sought. Many such linear combination pairs that are uncorrelated with each other may be found. The linear combinations are the *canonical variables* (variates or scores). The correlations between the pairs of canonical variables are the *canonical correlations*. The weights in the linear combinations are the *canonical coefficients* (partial regression coefficients).

Canonical correlation is used to predict the values of one set of variables from another set of variables whose values are known, to test whether two sets of variables are independent of each other, and to relate a set of background variables to a set of evaluation variables. For example, a student's success in college, measured by grade point average and number of extracurricular activities, may be predicted from a set of high school measurements such as class rank, SAT scores and grade average. Various consumer socio-economic measurements may be tested for independence of current purchasing patterns, or the selection of socio-economic measurements that relate to purchasing patterns may be sought.

2.2 BASIC USAGE

CANONICAL.COR requires an input P-STAT system file of raw data values and the LEFT identifier to specify the variables in the first set. All other variables in the file, except any character variables or any variables used as WEIGHT or CARRY variables, constitute the second set. The RIGHT identifier may be used to explicitly specify the variables in this set. Additional identifiers request output files, specify the contents of those files, and control the automatic listing of the STATS file of coefficients and correlations.

Figure 2.1 shows file TWA97, the data set used in the subsequent examples of canonical correlation. It gives the head lengths and breadths of brothers.[1] The first two variables are the first or left set; the next two or remainder are the second or right set.

[1] This data is from *An Introduction to Multivariate Statistical Analysis* by T.W. Anderson. The filename “TWA97” refers to the data set on page 97.

Figure 2.1 **Head Lengths and Breadths of First and Second Sons**

FILE TWA97:

HL	HB	HL	HB
Son1	Son1	Son2	Son2
191	155	179	145
195	149	201	152
181	148	185	149
183	153	188	149
176	144	171	142
208	157	192	152
189	150	190	149
197	159	189	152
188	152	197	159
192	150	187	151
179	158	186	148
183	147	174	147
174	150	185	152
190	159	195	157
188	151	187	158
163	137	161	130
195	155	183	158
186	153	173	148
181	145	182	146
175	140	165	137
192	154	185	152
174	143	178	147
176	139	176	143
197	167	200	158
190	163	187	150

2.3 Specifying Analysis Variables

The LEFT and RIGHT identifiers specify the two sets of variables. Only LEFT is required:

```
    CANONICAL.COR  TWA97,  LEFT  1 TO 2  $
```

All other variables, except for character variables and those designated as WEIGHT and CARRY variables, comprise the other set. The RIGHT identifier should be used when additional variables that are not analysis variables are present in the input file.

The arguments for LEFT and RIGHT are lists of variables. Ranges of variables that use the reserved word “TO” may be included. There should be a least two variables in each set. (When there is just a single variable in a set, linear regression analysis should be used.) Generally, there is not the same number of variables in each set. The size of the smaller set determines the maximum number of canonical correlations that can be computed.

Figure 2.2 CANONICAL.COR Reports the Canonical Correlations and Significances

```
     CANONICAL.COR  TWA97,   LEFT  1 TO 2,    NO LIST,
     STATS  TWA97Sta,   COR  TWA97Cor,   OUT  TWA97CV,   FULL  $


Canonical correlation using file TWA97 completed.
A STATS file named TWA97Sta has been created.

       25 cases read.
       25 cases used.
        2 variables were used in the LEFT  set.
        2 variables were used in the RIGHT set.

    .7108 is the largest correlation of any LEFT variable
          (HL.Son1) with any RIGHT variable (HL.Son2).


                                                        mult. RSQ with
                                                        the other vars
variable          set            mean            s.d.   in its own set

HL.Son1             L          185.72      9.76183043          .539572
HB.Son1             L          151.12      7.37292344          .539572

HL.Son2             R          183.84     10.04025232          .704344
HB.Son2             R          149.24      6.70994287          .704344



        canonical   eigenvalues     can cor sq /
      correlation  (can cor sq)   (1-can cor sq)

  1       .788508       .621745        1.643717
  2       .053740       .002888         .002896


                  eigenvalues     Bartlett’s
     eigenvalue        tested     chi square        df   significance

  1     .621745      1 to   2         20.964         4         .000322
  2     .002888      2 to   2           .062         1         .803082


          Wilks’        Rao’s
         lambda      approx F         df1          df2   significance

        .377163        6.5972           4           42         .000326
```

Observations with missing values of any of the analysis variables are omitted from the analysis, although they are included in the OUT file. Variables with zero standard deviation and variables that make inversion of the correlation matrices of original variables impossible are dropped from the analysis. The maximum number of variables in the combined LEFT and RIGHT sets is 200.

2.4 Specifying CARRY and WEIGHT Variables

Only the specified numeric variables are used in the canonical correlation analysis. Character variables and numeric variables that are not used in the analysis may be carried into the output file of canonical variables. A variable to weight the individual cases may be specified.

The CARRY identifier may be used to specify variables that should be carried into the OUT file (the output file of canonical variables or variates). Typically, these variables identify the cases or supply auxiliary information — they are not used in the analysis. Any character variables in the input file are automatically carried into the OUT file. Numeric variables that are not analysis variables or CARRY variables are ignored.

The canonical correlation analysis may be weighted. The WEIGHT identifier gives the name of the variable whose values are case weights. The weights may be integers or fractional numbers. Each case “counts” as the value of its weight, instead of “counting as one.” Cases with negative or missing values of the weight variable are ignored. Weighting affects the calculation of correlations, means and standard deviations. The true count of cases in the analysis is used in determining the degrees of freedom and total count for significance tests. If the true count of cases should be affected by weighting, use .REPEAT. with integer values of a weight variable.

2.5 OUTPUT OPTIONS

The CANONICAL.COR command produces both a printed report and various optional output files. The file of statistics — the canonical coefficients and structural coefficients — is automatically listed. Other output files of correlations and canonical variables may be listed after the command completes by using the LIST command.

2.6 The Report Containing Canonical Correlations

The report produced by CANONICAL.COR contains summary descriptive statistics, the canonical correlations and tests of their significance. Figure 2.2 shows the CANONICAL.COR command and the report that is produced for the data set shown in Figure 2.1. NO LIST is included in the command to suppress the automatic listing of the file of statistics, which is listed subsequently using the LIST command and shown in Figure 2.3.

The CANONICAL.COR report gives the largest correlation between any LEFT variable with any RIGHT variable. The first canonical correlation must be at least as large as this value. Then it gives the mean and standard deviation of each variable in the LEFT and RIGHT sets and the multiple R-squared of each variable with all the other variables in its own set.

Next, the canonical correlations are reported in order of their magnitude. These are the correlations between the canonical variables (linear combinations of the input variables) in one set with those in the other set. The squares of the canonical correlations — the eigenvalues — are estimates of the variance shared by the canonical variables (not the input variables, but the linear combinations of them). The ratios of “can cor sq” (the square of the canonical correlation) to “one minus can cor sq” are equal to the *discriminant criteria* or eigenvalues in discriminant analysis. (Tatsuoka, 1971)

The last portion of the CANONICAL.COR report tests the significance of the canonical variables by testing the eigenvalues, which are the squares of the canonical variables. Wilk’s lambda (the likelihood ratio) uses Rao’s approximate F as an overall test whether there is any significant linear relationship between the two sets of variables — that is, whether the canonical coefficients (weights) differ from zero. When there is an overall significance, it is then necessary to test how many of the pairs of canonical variables are significant. Bartlett’s chi square tests the pairs of canonical variables — it tests whether the canonical variables differ from zero. The test of all the eigenvalues (the first test) is equivalent to the Wilk’s lambda test.

In Figure 2.2, Wilk’s lambda indicates that there is a significant linear relationship between the LEFT and RIGHT variables. Bartlett’s chi indicates that the first and second pairs of canonical variables differ from zero (chi = 20.964), but that the second canonical variable does not (chi = .062). Thus, the first canonical variable sufficiently describes the relationship between the two sets of variables.

Figure 2.3 File of Statistics Listed with BY Headings

```
   LIST  TWA97Sta,  BY Statistic,  STUB Set,  SKIP.VAR Set  $


      ------------- statistic: stand canonical coef ---------------

      set   variable       left1       left2      right1      right2

      L     HL.Son1     0.552190    1.366374    -           -
            HB.Son1     0.521537   -1.378365    -           -

      R     HL.Son2     -           -           0.504448    1.768570
            HB.Son2     -           -           0.538288   -1.758566

      ------------- statistic: raw   canonical coef ---------------

      set   variable       left1       left2      right1      right2

      L     HL.Son1     0.056566    0.139971    -           -
            HB.Son1     0.070737   -0.186950    -           -

      R     HL.Son2     -           -           0.050243    0.176148
            HB.Son2     -           -           0.080222   -0.262084

      ------------- statistic: cor with canon vars ----------------

      set   variable       left1       left2      right1      right2

      L     HL.Son1     0.935288    0.353888    0.737482    0.019018
            HB.Son1     0.927151   -0.374687    0.731066   -0.020136

      R     HL.Son2     0.753977    0.015729    0.956207    0.292690
            HB.Son2     0.758266   -0.014740    0.961647   -0.274290
```

2.7 Output File of Canonical and Structure Coefficients

An output file containing the canonical coefficients and structure coefficients (loadings) is produced and listed as part of the default output of the CANONICAL.COR command. When the STATS identifier is used to provide a name for this file, it is a regular P-STAT system file. When STATS is not used, the file is a temporary file with a name that begins with “WORK”. The file is automatically listed by a generated LIST command as the CANONICAL.COR command completes.

NO LIST may be included in the command to suppress this automatic listing. Figure 2.2 shows the CANONICAL.COR command with the STATS and NO LIST options. NO LIST is used so that the file may be listed subsequently (in Figure 49.3) with the BY identifier — this produces a narrower listing that may fit better on a page. The list produced by CANONICAL.COR is a stubbed listing, with the “statistic” variable on the left instead of in the dashed heading.

The USE identifier may be employed to pass a character string containing suitable options and their arguments to the LIST command. The character string is included in the LIST command generated by CANONICAL.COR. This command turns on predefined titles and directs the automatic listing to a disk file named “PrtFile”:

```
      CANONICAL.COR  TWA97,  LEFT  1 TO 2,  USE  'TITLES,  PR "PrtFile" ' $
```

The character string argument must be enclosed in a matched set of either single or double quotes. Notice that single quotes are used in this example, and that the portion of the USE argument that is also the argument for PR is enclosed in double quotes.

The LIST options that comprise the USE argument string must be logical additions to the LIST command generated by CANONICAL.COR. That command is:

```
      LIST  FileName,  STUB  statistic  set  variable,
       SKIP.VAR  set,  MAX.PLACES 5  $
```

“FileName” is the name of the STATS file, which is either a supplied name or a default name for a temporary file. A brief message gives the name of the file when it is not supplied.

The STATS file includes both standardized and raw (unstandardized) *canonical coefficients* (see Figure 2.3). These are the weights in the linear equations that yield the canonical variables. The weights for the “left” equations are applied to the variables in the LEFT group, and similarly the weights for the “right” equations are applied to the variables in the RIGHT group. (Thus, there are blank fields in the file corresponding to the LEFT variables and the “right” weights and vice versa.)

The standardized coefficients are applied to standardized data, and the raw coefficients are applied to unstandardized data. The standardized coefficients are typically of most interest because they make comparisons among the original variables possible despite the fact that the variables are measured in different units. They show the relative contribution or importance of the variables with which they are associated. (In the head measurement data used in the figures, standardization is not an issue because the same units are used in all measurements.)

Also included in the STATS file are the correlations between each original variable and each canonical variable. The correlations between a canonical variable and its constituent variables are known as *structure coefficients* or *loadings*. Typically, only structure coefficients equal to or greater than .30 and only those calculated for significant canonical correlations are regarded as meaningful. (Pedhazur, 1982) Structure coefficients are similar to unrotated factor loadings. By examining the variables that produce high structure coefficients, one can understand what variables in one group are responsible for the correlation with the other group. One can then reasonably name the coefficients. (Our example has only two variables in each group and both contribute evenly to their significant canonical variables. Thus, the structure coefficients cannot be meaningfully named.)

Structure coefficients may be rotated to maximize or minimize them, much as factor loadings are rotated. Use the ROTATE command and input just the relevant portions of the STATS file to rotate the “left” coefficients:

```
      ROTATE  TWA97Sta

       [ IF  statistic  =  'cor with canon vars'  AND  set = 'L', RETAIN;
         KEEP    variable  TO  left2 ;
         RENAME  variable  TO  row.label ],

      VF  TWA97Vf2  $
```

A corresponding command rotates the “right” structure coefficients. Often, after the coefficients are rotated, they are easier to interpret and name. (See the chapter on the FACTOR command for more information on rotation options.)

Another useful interpretation of the structure coefficients relates to variance. The mean of the squared structure coefficients for a given canonical variable gives the *proportion of the total variance* (PV) of its group of original variables that is accounted for or extracted by that canonical variable. Thus, the first left canonical variable in Figure 2.3 accounts for about 87% of the variance in head length and breadth for first sons:

$$[\ (.935288)^2\ +\ (.927151)^2\ ]\ /\ 2\ =\ .8671863$$

The sum of the variance accounted for by all the canonical variables in the smaller group equals one.

The cross correlations between each canonical variable and the original variables not in its group provide information about redundancy. The mean of the squared cross correlations of a group of original variables with a given canonical variable is the *redundancy index* of those original variables. In other words, the redundancy of those variables is the proportion of their variance that is redundant with or explained by the linear combination of variables in the other group. Thus, the redundancy index of the right original variables and the first left canonical variable is about 57%:

$$[\ (.753977)^2\ +\ (.758266)^2\ ]\ /\ 2\ =\ .5717243$$

That is, about 57% of the variance of head length and breadth of second sons is predictable from the first canonical variable or linear combination of first son measurements.

The redundancy index may also be calculated as the product of the proportion of total variance (PV) and the squared canonical correlation (R^2) — that is, the variance of the original RIGHT variables explained by their canonical variable right1 (the mean of $.956207^2$ and $.961647^2$) multiplied by the variance shared by the canonical variables (.621745).

2.8 Output Files of Correlations and Canonical Variables

Additional output files containing correlations between the original input variables and canonical variables are options requested using the COR and OUT identifiers, respectively. The PREFIX, FULL, LIMIT and RAW identifiers control the contents and naming conventions of variables in these and the STATS output files.

The COR identifier both requests and provides a name for an output file (matrix) of correlations between all pairs of the original input variables that are used in the canonical correlation analysis. Figure 2.4 shows the COR file requested in the CANONICAL.COR command shown in Figure 2.2. The variables in the COR file that define the columns of the correlation matrix are all the input variables (excluding any character, CARRY and WEIGHT variables) and a character variable named "row.label" that defines the rows of the matrix. The CORRELATE command may be used to request a file of covariances, as well as this file of correlations if COR was inadvertently omitted from CANONICAL.COR.

Figure 2.4 File of Correlations Between Input Variables

```
LIST  TWA97Cor  $

                                                  row
 HL Son1     HB Son1     HL Son2     HB Son2  label

1.000000    0.734556    0.710752    0.703981  HL.Son1
0.734556    1.000000    0.693157    0.708550  HB.Son1
0.710752    0.693157    1.000000    0.839252  HL.Son2
0.703981    0.708550    0.839252    1.000000  HB.Son2
```

The OUT identifier requests and provides a name for an output file containing the left and right canonical variables (variates or scores) and any character or CARRY variables. When FULL is included in the CANONICAL.COR command, the original input variables are also included in the OUT file. Figure 2.5 shows the OUT file produced when FULL is used.

The canonical variables in the OUT and STATS files are named left1, left2, ... , right1, right2 .... , unless the PREFIX identifier is used to provide alternate prefixes. The arguments for PREFIX are two quoted strings — the first replaces “left” and the second replaces “right”. This command names the four canonical variables in Figure 2.5 First.Son.1, First.Son.2, Second.Son.1 and Second.Son.2:

```
      PREFIX  'First.Son.'  'Second.Son.' ,
```

The prefixes should be legal P-STAT names that contain only letters, numbers and periods, and they should not be more than about 14 characters long because suffixes of 1 through the number of canonical variables are added to the prefixes. They may be supplied within single or double quotes.

Figure 2.5 File of Input Variables and Canonical Variables (Variates)

```
     LIST  TWA97CV,  SKIP 5  $


    HL    HB    HL    HB
  Son1  Son1  Son2  Son2       left1       left2      right1      right2

   191   155   179   145    0.573128    0.013683   -0.583317    0.258678
   195   149   201   152    0.374972    1.695265    1.083577    2.299348
   181   148   185   149   -0.487691   -0.077381    0.039028    0.267232
   183   153   188   149   -0.020875   -0.732187    0.189756    0.795675
   176   144   171   142   -1.053470   -0.029438   -1.225925   -0.364255

   208   157   192   152    1.676227    2.019292    0.631393    0.714017
   189   150   190   149    0.106312    0.668489    0.290241    1.147971
   197   159   189   152    1.195473    0.105711    0.480666    0.185573
   188   152   197   159    0.191219    0.154618    1.444163   -0.239829
   192   150   187   151    0.276010    1.088402    0.299958    0.095360

   179   158   186   148    0.106545   -2.226819    0.009048    0.705463
   183   147   174   147   -0.445296    0.389511   -0.674085   -1.146229
   174   150   185   152   -0.742181   -1.431078    0.279695   -0.519019
   190   159   195   157    0.799510   -0.874087    1.183233   -0.067957
   188   151   187   158    0.120483    0.341568    0.861515   -1.739225

   163   137   161   130   -2.283988   -0.540415   -2.691020    1.019269
   195   155   183   158    0.799393    0.573567    0.660544   -2.443816
   186   153   173   148    0.148824   -0.312273   -0.644106   -1.584460
   181   145   182   146   -0.699902    0.483468   -0.352367    0.525039
   175   140   165   137   -1.392983    0.578389   -1.928493   -0.110724

   192   154   185   152    0.558958    0.340604    0.279695   -0.519019
   174   143   178   147   -1.237339   -0.122430   -0.473115   -0.441637
   176   139   176   143   -1.407154    0.905310   -0.894490    0.254402
   197   167   200   158    1.761368   -1.389886    1.514669    0.550699
   190   163   187   150    1.082457   -1.621885    0.219736    0.357444
```

The OUT file contains standardized canonical variables. The RAW identifier may be used in the CANONICAL.COR command to request that it contain unstandardized canonical variables. The LIMIT identifier controls the number of canonical variables that are included in both the OUT and STATS files. Normally, all the canonical variables are present in both files. LIMIT is followed by the desired number of canonical variables — LIMIT 3 would include only the three variables with the highest canonical correlations in the OUT and STATS files. All canonical correlations, however, are calculated and included in the report.

REFERENCES

1. Anderson, T.W. (1984). *An Introduction to Multivariate Statistical Analysi*s, John Wiley & Sons, New York.
2. Cohen, Jacob and Cohen, Patricia. (1975). *Applied Multiple Regression / Correlation Analysis for the Behavioral Sciences*, John Wiley & Sons, New York.
3. Cooley, William W. and Lohnes, Paul R. (1971). *Multivariate Data Analysis*, John Wiley & Sons, New York.
4. Pedhazur, Elazar J. (1982). *Multiple Regression in Behavioral Research*, Holt, Rinehart and Winston, New York.
5. Tatsuoka, Maurice M. (1971). *Multivariate Analysis*, John Wiley & Sons, New York.

SUMMARY

CANONICAL.COR

```
 CANONICAL.COR  TWA97,  LEFT  1 TO 2,
   STATS  TWA97Sta,  COR  TWA97Cor,  OUT  TWA97CV  $
```

The CANONICAL.COR command analyzes the relationships between two sets of variables by finding a linear combination of variables in one set that has maximum correlation with a linear combination in the other set. Multiple pairs of linear combinations are sought. The canonical correlations and their significances are reported. Output files of canonical coefficients (weights) and correlations (structure coefficients), correlations of the input variables, and canonical variables (variates or scores) may be requested.

Required:

CANONICAL.COR fn

specifies the name of the input P-STAT system file of raw data values. The variables may be grouped so that one set is on the left and the other on the right. Cases with missing values of any of the analysis variables are excluded from the analysis — they are, however, included in the OUT file.

Required Identifiers:

LEFT vn list

specifies the variables making up one set, that on the “left” of the input file. (The variables in the first set need not literally be on the left side of the file — just specify them individually as members of the first set.) Variable names and ranges of variables may be in the argument list following LEFT. RIGHT should be used to define the variables in the other set, unless all other variables (except any character, CARRY or WEIGHT variables) comprise the other set.

Optional Identifiers:

CARRY vn list

gives the names or ranges of variables that are to be carried into the OUT file. These variables are not used in the analysis. Typically, these are case-identifying variables. Any character variables in the input file are automatically carried.

COR fn

provides a name for and requests an output file of correlations between all pairs of the original analysis variables. The CORRELATE command may be used to produce a file of covariances (as well as this file of correlations), if desired.

FULL

requests that the original input variables be included in the OUT file that contains the canonical variables.

LIMIT nn

specifies the maximum number of canonical variables (variates) to include in the STATS and OUT files. (All canonical correlations are included in the report produced by CANONICAL.COR.)

LIST

requests that the output STATS file of canonical coefficients (weights) and correlations be listed. This is assumed, unless NO LIST is included in the CANONICAL.COR command. Then, the STATS file is still produced, but is not listed — it may be listed subsequently using the LIST *command*.

OUT fn

provides a name for and requests an output file containing the left and right canonical variables (variates or scores) and any character and CARRY variables. The canonical variables are standardized unless RAW is used. The original input variables are not included in this file unless FULL is used.

PREFIX 'cs' 'cs'

supplies two quoted strings to use as prefixes in place of "left" and "right" in naming the canonical coefficients and variables in the STATS and OUT files. The prefixes should be legal names for P-STAT variables — that is, they should begin with letters and contain only letters, numbers and periods. Suffixes of 1 through the number of canonical variables are added to the prefixes.

RAW

requests that the OUT file contain raw (unstandardized) canonical variables instead of standardized ones.

RIGHT vn list

specifies the variables making up the second set, that on the "right" of the input file (the variables need not literally be on the right). Variable namess and ranges of variables may be in the argument list. When RIGHT is not used, all other variables in the input file (except any character, CARRY or WEIGHT variables) comprise the second set.

STATS fn

provides a name for the output file of statistics produced by CANONICAL.COR. This file contains standardized and raw canonical coefficients (weights or partial regression coefficients) and the correlations of the input variables with the canonical variables (structure coefficients or loadings). When STATS is not used to provide a name for this file, it is given a name beginning with "WORK" and it is a temporary file.

USE 'cs'

gives a character string, enclosed in quotes, that should be added to the LIST command automatically generated by CANONICAL.COR to list the STATS file. The character string should be composed of valid LIST options. A useful option is PR with a print destination to direct the listing to a disk file or printer:

```
CANONICAL.COR  TWA97,  LEFT  1 TO 2,
 STATS  TWA97Sta,  COR  TWA97Cor,  OUT  TWA97CV,
 USE   "PR  'PrtFile' ",  PR  'PrtFile'  $
```

The command phrase that is the USE argument (the one in double quotes):

```
PR 'PrtFile'
```

is added to the generated LIST command and directs the listing to the disk file named PrtFile. Notice that PR must also be used in the CANONICAL.COR command to direct the report that is produced to the disk file as well.

WEIGHT vn

specifies the name of a variable whose values are case weights. Weighting affects all calculations, although the true count is used in determining the degrees of freedom and total for significance tests. Missing and nonpositive weights are ignored.

3
Q.CLUSTER: Clustering Cases

The Q.CLUSTER command clusters cases of data into the requested numbers of disjoint groups. Large numbers of cases with many variables may be "quickly" clustered in from one to five different numbers of groups in a single usage of the command. The cases may have integer or fractional weights, and they may be included or excluded from the clustering by the number of missing data values per case. The data is standardized unless the user requests no standardization. If desired, the number of iterations for both the drift and fixed passes and a convergence criterion may be specified.

Q.CLUSTER prints the initial cluster seeds, the classification cluster centers, the counts in each cluster with the cluster-to-center distances, and an analysis of the variance contributed by each variable. The cluster centers and the counts with cluster-to-center distances can be printed after any of six steps in the clustering procedure. Output files containing: 1) the variables used in clustering, any carried variables and the cluster distances and assignments, and 2) the cluster centers (means) may be requested.

3.1 BACKGROUND

The Q.CLUSTER procedure typically has four steps:

1. initial seeds,

2. drift iterations,

3. fixed iterations, and

4. classification.

The *initial seeds* step selects k widely-spaced cases with non-missing data values as the initial cluster seeds or centers, where k is the number of desired clusters. The *drift iterations* step assigns cases to the clusters and readjusts the cluster centers dynamically (a "k-means" method). The centers, which are the means of the cases currently in each cluster, drift as the clusters acquire more cases. The *fixed iterations* step uses the current cluster centers and assigns all cases to their best cluster. After all assignments, the cluster centers are recalculated. The *classification* step uses the fixed cluster centers to classify all cases into their final clusters and to calculate the distances of the cases from their cluster centers.

The initial seeds step is based on Hartigan's "leader" algorithm (Hartigan, 1975). It is not completely case-order independent, but it does reasonably well in selecting well-separated cases to serve as the initial cluster centers. By default, *one* drift iteration and *two* fixed iterations are done. You may specify an alternate number of iterations for the drift and fixed steps that may be zero. Each iteration requires a pass of the input file. The rate of drift in the drift iteration step is dampened to lessen case-order sensitivity. The fixed iteration step is case-order independent. Fewer than the specified number of drift and fixed iterations may be done because successive changes in the cluster centers are within the supplied convergence criterion or equal to zero. The data is standardized and distances are actually squared Euclidean distances, unless you request otherwise.

Figure 3.1 Q.CLUSTER: Finding Three Clusters in the Iris Data

```
  Q.CLUSTER  Iris,  TRY 3,  CARRY  Species.No,
       OUT  IrisCLus,  NO STANDARDIZE  $
```

```
File Iris - 3 cluster solution:

--------------------cluster centers after initial seed pass---------------

cluster  Sepal Length   Sepal Width  Petal Length   Petal Width
      1   58.00000000   40.00000000   12.00000000    2.00000000
      2   77.00000000   38.00000000   67.00000000   22.00000000
      3   49.00000000   25.00000000   45.00000000   17.00000000

--------------------cluster centers used for classification---------------

cluster  Sepal Length   Sepal Width  Petal Length   Petal Width
      1   50.06000000   34.28000000   14.62000000    2.46000000
      2   68.69444444   30.86111111   57.69444444   21.05555556
      3   59.20312500   27.51562500   44.20312500   14.34375000


--------------------classification  after the final pass------------------

                      weighted       mean distance    largest distance
cluster     cases        cases   to cluster center   to cluster center
      1        50        50.00            30.30200           155.75800
      2        38        38.00            63.08776           224.20910
      3        62        62.00            64.33500           286.52905

  total       150       150.00            52.67470           286.52905

--------------------analysis of variance----------------------------------

variable         between ms  df1    within ms        df2          F   prob

Sepal.Length      3688.7647    2      19.3150      147.0    190.979  .0000
Sepal.Width        639.8805    2      10.5506      147.0     60.649  .0000
Petal.Length     21910.8775    2      17.7604      147.0   1233.690  .0000
Petal.Width       3886.4352    2       6.0144      147.0    646.184  .0000


-----Q.CLUSTER of file Iris completed-----
    1 drift pass and 2 fixed passes were done.
  150 cases were read.
  150 cases were available for clustering.
```

3.2 BASIC USAGE

The Q.CLUSTER command requires only an input file and the desired number of clusters. The TRY identifier specifies the number of clusters:

```
Q.CLUSTER  Iris,  TRY 3  $
```

In this example, “Iris” is the name of the input P-STAT system file.[1] Three groups or clusters are requested. From one to five different numbers of clusters may be specified:

```
TRY 3 4 5 8 ,
```

Here, four different clusterings are requested. Q.CLUSTER will compute a three cluster solution, a four cluster one, a five cluster one and an eight cluster one. Requesting *different* numbers of clusters is often appropriate when the data does not have an obvious number of groups. Requesting *large* numbers of clusters helps detect outliers — an outlier may be the sole member of a cluster. Each cluster solution takes new passes of the data file. However, the initial reading of the file is done only once regardless of the number of clusterings requested.

3.3 Standardization

The input data are standardized unless NO STANDARDIZE is used in the Q.CLUSTER command. Standardization rescales all variables so that their variance is one. This is appropriate for most data sets, particularly those where the values of all the variables are not in the same units. All clustering variables are weighted equally when the data are standardized. Occasionally, a better clustering is obtained when the data are *not* standardized. This is the case with the iris data in Figure 3.1. Unstandardized data results in more weight being given to variables with larger variances. These may be the variables that best distinguish between clusters in certain data.

3.4 Requesting Output Files

The Q.CLUSTER command in Figure 3.1 requests a three cluster solution with no standardization of the input data. The OUT identifier requests an output file containing the input data and the cluster assignments and distances. The CARRY identifier specifies numeric variables that are to be “carried” with the input data, but not used in the actual clustering. These variables are placed in the output file. Any character variables in the input file are automatically carried — that is, they are placed in the output file but not used in the clustering. *All numeric variables* that are not carried or specified as a weight variable are used in the clustering. The PPL instructions KEEP and DROP may also be used to select variables.

The SAVE.CENTERS identifier may be used to request an output file of cluster classification centers (means). This file contains the means of each variable for each cluster, as well as an overall mean and standard deviation (unless the data has not been standardized). The means and standard deviations are not weighted and are based on the entire input file, not just on the cases that were clustered. If multiple clusterings were requested (multiple arguments to TRY), only the centers from the *last* clustering are saved in this file. This file may be listed and/or input to the RECLUSTER command to cluster additional data using these centers and standardization criteria.

3.5 The Cluster Report

Figure 3.1 also shows the output report produced by Q.CLUSTER. The cluster centers after the initial seed pass are the values of three widely-separated cases selected as initial centers. The cluster centers used for classification are the final centers computed after all the iteration steps. The classification after the final pass shows the number of cases in each cluster, the mean distance of the cases from their centers, and the largest distance in each cluster. The first cluster of the Iris data is relatively compact, for example, compared with the second and third clusters, which have mean distances that are more than twice as large as that of the first.

[1] “Iris” is a file of the Fisher (1936) iris data, used in many examples illustrating cluster and discriminant analysis.

The analysis of variance table, the final portion of the Q.CLUSTER report, shows the amount of variance explained by each of the variables used in the clustering. The variance within a cluster (within mean square or ms) is compared with the variance between clusters (between ms) and a univariate F statistic and its associated probability are calculated. A small within ms relative to a large between ms yields a large F value with a very small probability that such a value is due to chance. This suggests that the clusters are different. However, the F statistics *cannot* be used to test the null hypothesis of no differences among the clusters because the cases have been specifically assigned to the clusters to maximize differences. A nonsignificant F test does indicate a variable that does not aid in discriminating among clusters. Including NO ANOVA in the Q.CLUSTER command suppresses this portion of the output report.

3.6 Missing Data

Cases may have missing data on some of the cluster variables. Only cases with *all good* data are selected as cluster seeds (initial centers). Cases with *more than half* of their values missing are excluded from clustering. They are carried into the output file, but they have missing values on their cluster assignments and distances. The MAX.MISSING identifier controls how many variables in a case may have missing data and not force exclusion of the case. To exclude cases with more than three missing values, use:

```
      MAX.MISSING 3 ,
```

in the Q.CLUSTER command. To exclude cases with *any* missing values, use:

```
      MAX.MISSING 0 ,
```

Cases with missing data that are not excluded are assigned to clusters on the basis of their good data values. The good values contribute to the recalculation of the cluster mean; the missing values do not. In addition, any *variables* that are missing in more than fifty percent of the cases are excluded from the clustering. These variables are not used to calculate centers or classify cases.

3.7 Weighting

Cases may have integer or fractional weights. Cases with weights that are zero, negative or missing are ignored. Normally, each case “counts as one.” A weight of 1.5 causes a case to “count as one and one half." The variable that contains the case weights is indicated using the WEIGHT identifier:

```
      WEIGHT  Wt.Factor ,
```

The report produced by Q.CLUSTER shows both the actual number and the weighted number of cases in each cluster. The cluster centers and mean distances reflect the weighting; the cluster-to-center distances do not. The weight variable is carried into the output file.

3.8 Comparing Clusters with Known Categories

When the true groupings or categories of a particular data set are known, the calculated clusters may be compared with the actual categories using either the SURVEY or TABLES command. (SURVEY is part of *TABS* — P-STAT’s enhanced crosstabulation software.) Figure 3.2 illustrates this. The variable “Species.No” in the Iris data was carried into the output file. Within SURVEY, the columns and rows of a table are defined by the variables “Species.No” and “Cluster.3” (the cluster assignments in a three cluster solution). Correctly classified cases appear in the diagonal of the table. Options in the Q.CLUSTER command that minimize the number of off-diagonal cases should be selected.

Figure 3.2 Comparing the Iris Clusters with the Actual Species

```
SURVEY  IrisClus;

 BANNER  Species.No,  STUB  Cluster.3,
 NO SUMMARY,  NO MISSING,  NO PERCENTS ;


                        ==== Species.No ====

                  Total
                 Sample      1      2      3

 Total
 Sample             150     50     50     50

 cluster.3

        1            50     50      -      -

        2            38      -     36      2

        3            62      -     14     48
```

Plotting pairs of variables (when a reasonably small number of variables have been used in the clustering) and observing first the cluster assignment and then the known category at each point also aids in evaluating cluster assignments. Figure 3.3 shows a plot of “Petal.Length” by “Petal.Width” with the cluster assignment shown at each point — that is, a “3” means a case in cluster three with a petal whose length and width is indicated by the scale on the x and y axes. An asterisk indicates two cases with different cluster assignments at that point.

3.9 CONTROLLING OUTPUT AND ITERATIONS

The amount of information shown in the report produced by Q.CLUSTER and the number of iterations or passes through the data may be controlled. This is of particular interest when there are very many variables and/or cases used in the clustering. An output file of cluster centers may be requested and used to cluster additional data.

3.10 Showing Cluster Centers

The report produced by Q.CLUSTER includes the cluster centers obtained after the initial seed pass and the cluster centers (means) used for classifying the cases. Printouts of the cluster centers may be requested at six specific points in the clustering procedure:

1. at the start of the initial seed pass,
2. at the end of the initial seed pass,
3. after each drift iteration,
4. after each fixed iteration,
5. at the start of the classification pass, and
6. after the classification pass.

The SHOW.CENTERS identifier is included in the Q.CLUSTER command. Numeric arguments corresponding to the numbers in the prior list give the points at which the centers are to be printed. Normally, Q.CLUSTER assumes:

```
SHOW.CENTERS  2  5 ,
```

and cluster centers are printed at the end of the seed pass and at the start of the classification pass. The following prints centers at *all* possible points:

```
SHOW.CENTERS  1  2  3  4  5  6 ,
```

An argument of zero suppresses all printout of cluster centers. This argument:

```
SHOW.CENTERS  6 ,
```

prints centers at points 2, 5 *and* 6. To print centers *only* at point 6, use:

Figure 3.3 Plot of Iris Clusters: Petal Length by Petal Width

```
PLOT  IrisClus;

P  Petal.Length  BY  Petal.Width,
   OBSERVE  Cluster.3  ALIKE,  CHARS  '123*' ;
```

```
 Plot of Petal.Length by Petal.Width (Legend: 1=1, 2=2, ..., *=4+)
         Observations are Like values of variable Cluster.3


       70 +                                            2
          |                                      2 2 2
          |                                  2   2
P      60 +                                  2 2   2   2   2
e         |                          2   2   2     2 2 2 2 2
t         |                                    2 2 2   2
a      50 +                            * 3 2 3 3 *     2 3
l         |                      3 3 3 3 3 3 3
.         |                      3 3 3 3
L      40 +                  3 3 3 3 3
e         |                  3 3   3
n         |                  3
g      30 +                    3
t         |
h         |
       20 +  1   1
          |1 1 1 1 1 1
          |1 1 1 1
       10 +1 1
           --------+---------+---------+---------+---------+
                   5         10        15        20        25
                              Petal.Width
```

```
AGAIN,  OBSERVE  Species.No  ALIKE ;
```

```
 Plot of Petal.Length by Petal.Width (Legend: 1=1, 2=2, ..., *=4+)
         Observations are Like values of variable Species.No

       70 +                                            3
          |                                      3 3 3
          |                                  3   3
P      60 +                                  3 3   3   3   3
e         |                          3   3   3     3 3 3 3 3
t         |                                    3 3 3   3
a      50 +                            * 2 2 3 3 3     3 3
l         |                      2 2 2 2 2 3 *
.         |                      2 2 2 2
L      40 +                  2 2 2 2 2
e         |                  2 2   2
n         |                  2
g      30 +                    2
t         |
h         |
       20 +  1   1
          |1 1 1 1 1 1
          |1 1 1 1
       10 +1 1
           --------+---------+---------+---------+---------+
                   5         10        15        20        25
                              Petal.Width
```

```
      SHOW.CENTERS  -2  -5  6 ,
```

or:

```
      SHOW.CENTERS  0  6 ,
```

When there are a large number of variables and many clusters are requested, the printout of cluster centers may be very lengthy. For example, with 100 variables and 15 clusters, a *single* printout of centers will require about 200 lines. You may want to suppress or limit the printout of cluster centers in these circumstances. On the other hand, when there is a reasonably small number of variables and clusters, observing the changes in the means of the clusters at the different points in the clustering procedure may be informative.

3.11 Showing Cluster Counts and Distances

The Q.CLUSTER report also includes a printout of the cluster classifications. This gives the count of cases in each cluster, a weighted count, and the mean and maximum distances of cases to their cluster centers. This printout may be requested at any or all of three points in the cluster procedure:

3. after each drift iteration,
4. after each fixed iteration, and
5. at the start of the classification pass.

The SHOW.NS identifier with numeric arguments corresponding to this list specifies the printout points. Q.CLUSTER assumes:

```
      SHOW.NS  5 ,
```

As is the case with SHOW.CENTERS, negative arguments suppress specific printouts and a zero argument suppresses all printouts.

The mean and maximum distances shown in the cluster report are *squared* Euclidean distances. Including the EUCLIDEAN identifier in the Q.CLUSTER command results in true Euclidean distances. This does not affect the cluster solution or assignments. The reported distances, however, are smaller. Because many square roots must be computed, the Q.CLUSTER procedure may be slightly slower with large data sets when EUCLIDEAN is requested. SQ.EUCLIDEAN is the default.

3.12 Drift and Fixed Iterations

Q.CLUSTER normally makes *one* drift pass and *two* fixed passes through the data as part of the clustering procedure. The number of drift and fixed iterations may be controlled with the DRIFT and FIXED identifiers. A single numeric argument giving the number of passes may follow DRIFT and/or FIXED:

```
DRIFT 2,  FIXED 3,
```

An argument of zero suppresses a pass. Additional passes through the data file take more time, but they may aid in getting a better cluster solution. Alternatively, with a very large number of cases, fewer passes may make the cluster solution feasible.

3.13 Convergence Criterion

A convergence criterion may be specified to control the number of drift and fixed passes. Normally, either the default number or the specified number of iterations is made. The CONVERGE identifier gives a value to use as a criterion for stopping iterations. After a drift or fixed iteration ends, the *largest change in any cluster center (L)* is compared with the distance between the *two closest cluster seeds (D)* at the end of the initial seed pass. If the largest change is less than or equal to the distance times the criterion, iterations stop. For example, if:

```
CONVERGE .003 ,
```

is specified, iterations stop when:

```
L  <=  (D * .003)
```

The default value for CONVERGE is zero — iterations stop when there is no change in the cluster centers. As long as there are changes, iterations continue until the default number or the number specified with DRIFT and/or FIXED is done. Consider the type of distance measurement in use — squared Euclidean, the default, or true Euclidean — when specifying a CONVERGE value.

3.14 Large Numbers of Variables and Cases

When a large number of *variables* are used in clustering, several options may optimize the clustering procedure. “Large number” can be considered to be 100 or more variables, although the size of P-STAT in use and whether output is directed to the terminal or a disk file may influence the concept of large. These options are useful when clustering very many variables:

1. SHOW.CENTERS 0,
2. SAVE.CENTERS OutFile, and
3. SHOW.NS 3 4 5.

The first option turns off the printing of the cluster centers — the table of variable means that normally prints at two points in the cluster analysis. This table can be quite large when it includes many variables. The second option saves the classification centers (means) in an output file that can be listed at another time. The third option turns on the printing of cluster counts and distances at all possible points in the cluster analysis. This table is relatively compact and may help in evaluating the correct number of clusters for a particular data set.

When the input file has a very large number of *cases*, and “large” again depends on the size of P-STAT in use and the speed of the computer, a sample of cases may be input to Q.CLUSTER. For example, if the input file has

100,000 cases, a 2,000 case sample may be clustered. The centers or means are saved in an output file using the SAVE.CENTERS identifier. This file is input to the RECLUSTER command to get cluster assignments for all cases.

REFERENCES

1. Anderberg, Michael R. (1973). *Cluster Analysis for Applications,* Academic Press, New York.
2. Hartigan, John A. (1975). *Clustering Algorithms*, John Wiley & Sons, New York.
3. Fisher, R.A. (1936). "The Use of Multiple Measurements in Taxonomic Problems," *Annals of Eugenics*, 7, 179-188.

SUMMARY

Q.CLUSTER

```
 Q.CLUSTER  Iris,  TRY 3,  CARRY  Species.No,
   OUT  IrisClus,  NO STANDARDIZE  $
```

The Q.CLUSTER command groups cases into disjoint clusters. From one to five different numbers of clusters may be requested in a single usage of the command. The data may have integer or fractional weights and missing values on up to half the variables. Data is standardized, so the values need not be in the same units of measurement. One drift pass and two fixed passes are normally made during clustering.

Required:

Q.CLUSTER fn

gives the name of the required input file. All numeric variables in the file are used in the clustering unless they are specified as CARRY or WEIGHT variables. Any character variables are automatically carried to the output file of cluster assignments and distances, if one is requested.

Required Identifiers:

TRY nn nn

specifies the desired number or numbers of clusters. From one to five different numbers of clusters may be requested. Each cluster solution takes new passes of the data file. However, the initial reading of the file is done only once.

Optional Identifiers:

ANOVA

requests that the analysis of variance table be included in the report produced by Q.CLUSTER. This is assumed; NO ANOVA suppresses the table. The ANOVA table shows the amount of variance explained by each of the variables used in the clustering. The univariate F statistics cannot be used to test the null hypothesis of no differences among the clusters, however, because the cases have been specifically assigned to the clusters to maximize differences. Nonsignificant F’s indicate variables that do not aid in discriminating among clusters.

CARRY vn vn

names variables in the input file that should be carried to the output file and *not* used in the clustering. All numeric variables that are not CARRY or WEIGHT variables are used to cluster cases. Character variables are automatically carried.

CONVERGE nn

gives a convergence criterion to control the number of drift and/or fixed iterations. Normally, CONVERGE 0 is assumed and the default or requested number of iterations continue until there is no change in the cluster centers from one pass to the next. The largest change in any cluster center (L) after an iteration is compared with the distance between the two closest cluster centers (D) at the end of the initial seeds pass. If L is less than or equal to D times the criterion, iterations stop.

DRIFT nn

gives the number of drift passes to make through the input file. Normally, one drift pass and two fixed passes are made. DRIFT 0 suppresses all drift passes. During a drift iteration, cluster centers are updated dynamically as each cases is assigned to a cluster.

EUCLIDEAN

requests that true Euclidean distances be calculated. This does not affect cluster assignments, but does result in smaller reported distances of cases from their centers. However, computation time is increased. Squared Euclidean distances are calculated when EUCLIDEAN is not used.

FIXED nn

gives the number of fixed passes to make through the input file. Normally, one drift pass and two fixed passes are made. FIXED 0 suppresses all fixed passes. During a fixed iteration, cluster centers are updated after all cases are assigned to all clusters.

MAX.MISSING nn

specifies the maximum number of missing data values a case may have and still be included in the cluster analysis. Normally, cases with *more than half* of their values missing are excluded from the clustering. MAX.MISSING 0 excludes cases with *any* missing values. Cases with missing data that are not excluded are assigned to clusters on the basis of their good data values. Similarly, *variables* that are missing in more than fifty percent of the cases are excluded from the clustering.

OUT fn

provides a name for and requests an output file of cluster assignments and distances. All cluster variables and any CARRY or WEIGHT variables are included in this file.

SAVE.CENTERS fn

provides a name for and requests an output file of cluster classification centers (means). The means of each variable for each cluster, as well as an overall mean and standard deviation are typically included in this file. The file may be listed and/or input to the RECLUSTER command to cluster additional data using these centers and standardization criteria.

SHOW.CENTERS nn nn

requests that the cluster centers print at the specified points in the cluster analysis. Six specific points may be specified by number:

1. at the start of the initial seed pass,
2. at the end of the initial seed pass,
3. after each drift iteration,
4. after each fixed iteration,
5. at the start of the classification pass, and
6. after the classification pass.

SHOW.CENTERS 2 5 is assumed by Q.CLUSTER when other points are not specified. An argument of zero suppresses all printout of cluster centers. Negative arguments suppress printouts at just the specified points.

SHOW.NS nn nn

requests that the cluster counts and distances print at the specified points in the cluster analysis. Three specific points may be specified by number:

3. after each drift iteration,
4. after each fixed iteration, and
5. at the start of the classification pass.

SHOW.NS 5 is assumed by Q.CLUSTER when other points are not specified. As is the case with SHOW.CENTERS, negative arguments suppress specific printouts and a zero argument suppresses all printouts.

SQ.EUCLIDEAN

requests that squared Euclidean distances be calculated. This is the default unless EUCLIDEAN is used.

STANDARDIZE

causes the data to be standardized prior to clustering. This is assumed by Q.CLUSTER. Standardization weights all clustering variables equally, regardless of the original units of measurement of those variables. NO STANDARDIZE turns off standardization — more weight is then given to variables with larger variances.

WEIGHT vn

gives the name of a variable whose values are the amount of weight to give each case. When WEIGHT is not used, each case “counts as one.” Integer or fractional weights may be supplied. Negative, zero or missing weights are ignored.

4
DISCRIM:
Discriminant Analysis

Discriminant analysis is a procedure that is used to: 1) decide whether groups are different based on the information available in some number of analysis variables, or 2) identify group membership based on functions produced in a previous discriminant run. When a number of variables are to be analyzed simultaneously, the DISCRIM command must be used to perform a multiple group, multivariate discriminant analysis.

A school might use discriminant analysis to try to identify variables that classify students into either a group of dropouts or a group of graduates. The procedure would be to collect data, that is, test scores and background items, for students who had graduated and for those who had not graduated. In such a case, group membership is clearly defined. The goal of this discriminant analysis might be to find out if the groups are different based on the available analysis variables and to isolate variables that could be used to predict group membership for new students.

4.1 THE DISCRIMINANT ANALYSIS PROCEDURE

In order to perform a discriminant analysis, a pass is made through the data file to compute a set of weights for each group. These weights are based on the different values of the analysis variables for each group's cases. The weights are coefficients that, with a constant, form an equation for predicting group membership. All the variables that can be included without numerical problems are entered simultaneously. The identifier STEP requests a backward stepping run to delete any variables that contribute little to the discriminant process. STEP is assumed in an interactive run, but must be explicitly specified in batch mode to request a stepwise analysis. In an interactive run, the stepping is controlled by the user. In a batch run, the stepping is automatic with a few options to force the inclusion of certain variables.

A score is computed for each case using the weights of each of the groups. The case is then assigned to the group that gave it the largest score. For example, if a student who actually graduated had data which, according to the weights, looked more like dropout data, that student would be assigned to the dropout group. After all the cases are assigned, a classification table, which provides a frequency distribution of how well the assignments were made, is printed. If all the graduates are correctly assigned to the graduate group and all the dropouts are correctly assigned to the dropout group, the discriminating power of the variables that were used is very good. If, however, only 50 percent of the cases are correctly assigned, the discriminating power of the variables that were used provide no better than chance selection.

The number of groups that may be defined depends on the P-STAT size. The maximum number of groups is40. The maximum number of analysis variables is 190.

4.2 Three Types of Discriminant Runs

There are three ways in which the DISCRIM command can be used:

1. A full analysis and classification run — DISCRIM reads group definitions and the input file. It computes functions and classifies cases. The functions and the classifications of each case can both be saved as P-STAT files.

```
DISCRIM   File1,  FUN  FunFile1 ;
  Graduate  Degree  1
```

```
      Dropout   Degree  0
      $
```

2. A classification run with known group membership — In this example, the group to which the input cases belong is known, but we wish to use functions generated previously to see where they are actually assigned:

```
     DISCRIM   File2,  IN.FUN  FunFile1 ;
       Graduate  Degree  1
       Dropout   Degree  0
       $
```

3. A classification run with unknown group membership — This is done when group membership is unknown, but is being predicted on the basis of previously computed functions.

```
     DISCRIM  File2,  IN.FUN  FunFile2  $
```

An input file name is always required. If no file name is given, the most recently referenced file is used. The input file can have any number of cases, but the cases should have complete data on the analysis variables. Any cases with incomplete data are omitted from the analysis. In addition, the number of usable cases should exceed the number of analysis variables.

4.3 Output Printed by the Discriminant Run

The output from DISCRIM includes the number of cases and the definition rules for each group. The means are reported on each variable for all groups taken together and for each group taken separately. This is followed by a table of univariate F values for each variable. These are not affected by the inclusion or deletion of other analysis variables. They are identical to the F values produced by the FREQ program, assuming that the same groups are compared. The multivariate F values are then printed for each variable. In automatic stepping the variable with the lowest multivariate F is the next variable deleted.

Wilk’s lambda and its approximate F equivalent is printed after each step. This is the overall significance of the group separation. At the end of the analysis, the Mahalanobis D square, which tests the hypothesis that the group means on the variables do not differ, is printed. A classification table displaying the degree of success of the classifying procedure is also printed. If a case from original group 2 is classified into group 4 (on the basis of the student’s discriminant scores), it is counted in the second row and fourth column of the table. If all cases are correctly classified, the table has non-zero counts only in the diagonal.

4.4 Defining the Groups

Subcommand records that define the groups must be supplied unless the analysis is run using previously computed functions (IN.FUN) and subgroup membership is not known, but is being predicted. The use of the semicolon (;) rather than the currency symbol ($) indicates that subcommands follow. The $ is used when all the groups have been defined and the command is complete.

The order in which the fields appear in the subcommand records is important, but the spacing is not. The group name, which must be specified first, may be up to 16 characters long. It must be enclosed in quotes if it does not conform to the rules for legal P-STAT names. The variable, which must follow the group name, may be referenced by its name. The variable name is followed by values defining the range to be included in the group. For example:

```
GROUPNAME1  VARIABLE  LOW  HIGH
GROUPNAME2  VARIABLE  LOW  HIGH
$
```

Figure 4.1 Discriminant Analysis: Input and Partial Output

```
    DISCRIM  Survey ;

    Male     Sex  1
    Female   Sex  2
    $

                           UNIV.   UNIV.       MULT.   MULT.
                               F   PROB.           F   PROB.

  1              Age        1.13  0.2892        3.64  0.0578
  2        Education        0.01  0.9371        0.43  0.5149
  3       Occupation        1.64  0.2010        3.29  0.0709
  4   Marital.Status        0.61  0.4361        1.31  0.2538
  5      Work.Status        3.85  0.0509        0.04  0.8361
  6         Children        0.58  0.4480        3.80  0.0525
  7         Siblings W      0.02  0.8808        0.01  0.9269
  8    Hrs.Last.Week       15.05  0.0001       11.68  0.0008


...  WILKS LAMBDA=  0.90336,   APPROX. F=        3.00,  PROB=  0.003
...  DF=       8. AND       224.0
.....ENDING   STEP   1 WITH   8 VARIABLES IN USE.........

 GENERALIZED MAHALANOBIS D SQUARE =         24.71

 WITH     8 DEGREES OF FREEDOM.

 INTERPRETED AS A CHI SQUARE, THIS D SQUARE VALUE
 INDICATES A PROBABILITY OF   0.0019 THAT THE MEAN VALUES
 ARE THE SAME IN ALL THE GROUPS FOR EACH OF THE VARIABLES.

 CLASSIFICATION TABLE.  ROWS ARE ORIGINAL GROUPS.
 COLUMNS ARE ASSIGNED GROUPS BASED ON LARGEST SCORE.

                             1        2

                          Male   Female

  1             Male        85       57
  2           Female        33       58

               143 OUT OF      233 WERE ASSIGNED TO THEIR ORIGINAL GROUP.
 THAT IS   61.4 PER CENT.
```

In Figure 4.1, the group Male is defined as having a value of 1 of variable Sex. Since no high value is provided, it is assumed to be the same as the low value. Two variables may be used to define a group. This could be used to define a group based on variables named Sex and Age:

```
       Males.Under.20    Sex 1 1     Age 1 19
```

In this case, the high value for the first variable must be provided even though it is the same as the low value. The high value can only be omitted when it is the same as the low value and also the last field on the record:

```
        Males.Under.20    Age 1 19    Sex 1
```

4.5 Optional Files

FUN provides a name for an optional output file of discriminant functions. It contains a case (row) for the weights of each analysis variable, plus a final case for a constant. Thus, the first variable (column) contains the first discriminant function with the constant in the last case. These are the weights that are used to produce scores for membership in the first group. The second variable contains the weights for the second group, and so forth. (Notice that the discriminant functions are scaled so that the discriminant scores center around zero.)

IN.FUN is an input file of functions generated by FUN in a previous run. Using IN.FUN causes the DISCRIM program to apply those previously generated functions to the input file instead of calculating functions from the data input file. A classification table is printed. Using IN.FUN allows cross-validation of a discriminant analysis, or classification of new cases using previously computed weights. It cannot be used in the same command as FUN.

The identifier OUT may be used to request an optional output file of probabilities and discriminant scores. It has a case (row) for each input case that was used. The number of variables (columns) is four more than the number of groups. The first variable in the output file contains the original group for that case. The second variable in the output file contains the group that the case probably belongs to, based on its discriminant scores computed using each group’s function. Variable three in the output file contains the probability that a case belongs to its original group. This is a number between 0 and 1. The fourth variable contains the probability that a case belongs to its highest score group.

If a case is assigned by the program to its own group, its scores in the output file on the first two variables are identical, and its scores (probabilities) on variables three and four are also identical. The fifth variable contains the discriminant score for membership in the first group. The sixth variable contains the discriminant score for membership in the second group. If there are more than two groups, there are more variables, one for each additional group.

4.6 CONTROLLING VARIABLE SELECTION

The classification procedure can be told to take differences in group sizes into account. PRIOR specifies that the probabilities are from a prior run with group sizes that are different from the current ones. PRIOR uses the current group sizes unless it is followed by arguments for each group to take the differences in group sizes into account. This:

```
       PRIOR 20 40 40
```

indicates that groups two and three are actually twice as large as group one, and cases are therefore more likely to be classified into those larger groups.

The identifier STEP causes backwards stepping during a run. The stepping decisions are made by the user, either dynamically from the terminal or from previously supplied responses in the edit file. STEP is assumed in an interactive run and need not be explicitly specified. In a batch run, the use of STEP causes variables that seem less useful (by an automatic rule) to be deleted one-by-one from the analysis until only effective ones are left. If STEP is not used, the program simply uses *all* analysis variables, no matter how little significance some of them may have.

A variable that is linearly predictable from other variables (for example, one which is the difference of two others) can cause problems in programs like DISCRIM that use matrix inversion. The technique used here is to invert by sequentially pivoting on the largest available diagonal. This is fine until, after some pivoting, the remaining diagonals dissolve into rounding error because the remaining variables are linear combinations of some of the variables that have already been included. When this occurs, DISCRIM simply keeps those variables upon which it had pivoted and drops the rest. Reports are produced as it does this. This situation can occur in a non-stepwise run or prior to the first step of a stepwise run.

Sometimes certain variables are more important than others and should be kept in the analysis if at all possible. The identifier FORCE may be used to identify which variables should remain in the analysis. FORCE 4, for example, tells DISCRIM that the first four variables should not be dropped. They are pivoted upon first, so that colinearity with other variables causes dropping of those other variables rather than the force variables. If, however, they are colinear among themselves, the run ends with an error message. They are never dropped in an automatic stepping run, but can be dropped in an interactive run if desired.

Note that the FORCE variables must be the left-most variables in the input file. Variable selection can be used to rearrange the order of the variables. This is the only constraint on the positions of the variables in the file.

4.7 Probability and F Values

As indicated earlier, the use of STEP causes backwards stepping to occur. Variables are dropped, one by one, until no more can or need to be dropped. The variable to be dropped is normally the one with the lowest multivariate F value. This continues until all remaining (non-force) variables have F values high enough so that the probability of each F (which also depends on sample size) is .05 or less. The identifier PROB can be used to modify the probability threshold. It is followed by an argument specifying the probability setting:

```
PROB  .025
```

The identifier F.DELETE specifies that a particular F value be used as a criterion for deletion instead of the probability setting normally assumed. For example:

```
F.DELETE 4
```

indicates that an F value of 4 rather than a probability should be used. The F is an absolute value not related to sample size. The probability that a given F is significant depends on the sample size.

Setting the verbosity level to 4 (V 4) causes the full information for each step to be printed. This includes the F for each variable that is still included in the analysis. If V is less than 4, only Wilk’s lambda and its approximate F and probability are printed after the individual steps.

You may indicate a maximum number of variables to be kept, a minimum number of variables to be kept, or both. If MIN 5 and MAX 10 are specified, the following procedures are performed (assuming testing is on the F values):

1. Non-force variables are dropped until a total of ten (force and non-force) are left, no matter what their F values are.
2. Once MAX is satisfied, stepping could stop if at some point all F values are above the F threshold (on the non-force variables).
3. Once only five variables are left, stepping ends. This indicates that the best seven variables should be kept:

```
MIN 7,  MAX 7
```

This example indicates that the first three variables should be kept no matter what, keep at least two others, but no more than seven others:

```
MIN 5,  MAX 10,  FORCE 3
```

4.8 Controlling Stepping

As stated earlier, an interactive run is automatically a stepwise run. The DISCRIM program prompts the user for a decision. Most responses are single letters like W or C. The following options are available:

1. **H** requests HELP by describing all the options.

2. **Q** requests that the command be quit.
3. **L** requests that the multivariate F value and probability for each variable be listed again.
4. **K** requests that all remaining variables be kept and that the stepping ends.
5. **W** requests that the worst variable be dropped, and that the stepping continues. The worst variable is signalled by a W after its name. It has the lowest F of any non-forced variable.
6. **nn** requests that the nnth variable be dropped and that stepping continues. (nn is a number from 1 to the number of the current variable.)
7. **A** requests that automatic control resumes and interactive control stops.
8. **A,nn** requests that automatic control resumes for nn steps and then returns to interactive control unless one of the criteria for stopping occurs first. (nn is a number such as 3.)
9. **R** requests that the most recently dropped variable be restored and stepping resumes. (Note: You cannot go back more than one step.)
10. **C** requests that a classification table be produced. A pass is made through the file. Control then returns for another decision.
11. **C,nn** requests that a classification table be produced with only those cases whose highest group probability is nn more than the next highest group's probability. (nn is a number such as .10 .)

The status of other options during an interactive run is as follows:

1. STEP is assumed.
2. FORCE is helpful when the run begins with colinearity problems.
3. Even when the run is interactive, stepping stops either when the number of variables equals the number of groups, when a MIN level is reached, or when only FORCE variables are left.
4. MAX, MIN, PROB, and F.DELETE are in normal usage whenever automatic mode (A) is in control.

Figure 4.2 shows the entirety of an interactive run in which automatic stepping is specified. After each step a brief report is printed naming the variable that was dropped, its F value and probability and the current value of Wilks Lambda. If we had been controlling the stepping ourselves, we could have printed a classification table after each step to see just how well we had done in predicting the group membership.

Figure 4.2 Interactive DISCRIM: Save the Functions

```
         DISCRIM purchase, FUN Pfun;

         Planned  Planned.purchase 1
         Spontaneous Planned.purchase 2
         $
```

```
 Discriminant analysis using input file Purchase

     207 cases were read from the file
         and tested for group membership.

     203 were placed in some group (see ROWS READ)
         and then tested for missing data.
```

```
    178 were kept in a group (see ROWS USED)
        i.e., no missing data.

                        ROWS   ROWS          DEFINITION OF
                        READ   USED          EACH GROUP

 1          Planned     92     74 Planned.Purchase         1
 2      Spontaneous    111    104 Planned.Purchase         2

means of each variable for the combined groups
and  for each group separately.
                                                         Spo
                          combined       Planned      ntaneous

  1    Family.Income  13.691011236  13.743243243  13.653846154
  2    Age.Male.Head  4.0280898876  3.9054054054  4.1153846154
  3   Education.Male  3.7865168539  3.8918918919  3.7115384615
  4 Education.Female  3.7134831461  3.8513513514  3.6153846154
  5       Occupation  3.5561797753  3.7297297297  3.4326923077
  6      Market.Size  5.1292134831  4.9189189189  5.2788461538
  7             Race  1.0112359551  1.0270270270             1
  8     Relationship  4.9662921348  3.3648648649  6.1057692308
  9         Occasion  2.0393258427  1.9729729730  2.0865384615
 10        Influence  2.3033707865  2.0405405405  2.4903846154

 UNIVARIATE AND MULTIVARIATE F VALUES ( AND PROBABILITIES )

   univariate degrees of freedom are       1. and       176.
 multivariate degrees of freedom are       1. and       167.

                             UNIV.   UNIV.         MULT.   MULT.
                                 F   PROB.             F   PROB.

  1    Family.Income         0.030  0.8631         2.928  0.0889
  2    Age.Male.Head W       0.841  0.3604         0.184  0.6685
  3   Education.Male         0.889  0.3471         1.737  0.1894
  4 Education.Female         5.045  0.0259         8.200  0.0047
  5       Occupation         0.483  0.4880         2.469  0.1180
  6      Market.Size         0.547  0.4606         0.424  0.5158
  7             Race         2.856  0.0928         3.161  0.0772
  8     Relationship        14.476  0.0002         4.946  0.0275
  9         Occasion         0.353  0.5533         0.315  0.5755
 10        Influence        13.643  0.0003        11.382  0.0009

...  WILKS LAMBDA is .811301227544, df are 10 and 167
...  its approximate F is 3.884, prob = .0001
.....Ending step 1 with 10 variables in use.
```

----------Prompt For Next Step and Reply-----------------------------

enter decision, of H or Q.
a

```
.....STARTING STEP   2 WITH    Age.Male.Head dropped.....
...  it had a mult. F in the previous step of       0.184,  prob=  0.6685
...
...  WILKS LAMBDA is .812195237305, df are 9 and 168
...  its approximate F is 4.316, prob = .0000
.....Ending step 2 with 9 variables in use.


.....STARTING STEP   3 WITH         Occasion dropped.....
...  it had a mult. F in the previous step of       0.266,  prob=  0.6069
...
...  WILKS LAMBDA is .813479667515, df are 8 and 169
...  its approximate F is 4.844, prob = .0000
.....Ending step 3 with 8 variables in use.


.....STARTING STEP   4 WITH      Market.Size dropped.....
...  it had a mult. F in the previous step of       0.284,  prob=  0.5948
...
...  WILKS LAMBDA is .814846715472, df are 7 and 170
...  its approximate F is 5.518, prob = .0000
.....Ending step 4 with 7 variables in use.


.....STARTING STEP   5 WITH   Education.Male dropped.....
...  it had a mult. F in the previous step of       1.207,  prob=  0.2736
...
...  WILKS LAMBDA is .820630417568, df are 6 and 171
...  its approximate F is 6.229, prob = .0000
.....Ending step 5 with 6 variables in use.


.....STARTING STEP   6 WITH    Family.Income dropped.....
...  it had a mult. F in the previous step of       2.296,  prob=  0.1315
...
...  WILKS LAMBDA is .831649655845, df are 5 and 172
...  its approximate F is 6.964, prob = .0000
.....Ending step 6 with 5 variables in use.


.....STARTING STEP   7 WITH       Occupation dropped.....
...  it had a mult. F in the previous step of       1.135,  prob=  0.2882

MULTIVARIATE F VALUES AND PROBABILITIES. DF=  1. AND       173.

                           MULT. F   PROB

  1 Education.Female         5.709  0.0179
  2             Race W       4.693  0.0317
  3     Relationship         7.141  0.0083
  4        Influence        10.205  0.0017

...  WILKS LAMBDA is .837137227765, df are 4 and 173
...  its approximate F is 8.414, prob = .0000
```

```
.....Ending step 7 with 4 variables in use.

Stepping ended, all probabilities under  0.0500

Generalized Mahalanobis D square= 34.240321613464, df= 4 .
Interpreted as a chi square, there is a probability
of .0000 that the group means are the same.


 CLASSIFICATION TABLE.  Rows are original groups.
 Columns are assigned groups based on largest score.

                             1        2
                                    Spo
                       Planned ntaneous

  1          Planned        47       27
  2      Spontaneous        29       75

122 of 178 cases (68.5 percent) were correctly assigned.
```

Figure 4.3 Using the Discriminant Functions From a Previous Data Set

```
       DISCRIM Purchas2,
          STEP,
          IN.FUN Pfun ;

       Planned      Planned.purchase 1
       Spontaneous  Planned.purchase 2
       $
```

```
Discriminant analysis using input file Purchas2

    207 cases were read from the file
        and tested for group membership.

    203 were placed in some group (see ROWS READ)
        and then tested for missing data.

    181 were kept in a group (see ROWS USED)
        i.e., no missing data.

                        ROWS   ROWS          DEFINITION OF
                        READ   USED          EACH GROUP

 1          Planned     88     74 Planned.Purchase         1
 2      Spontaneous    115    107 Planned.Purchase         2
```

```
means of each variable for the combined groups
and  for each group separately.
                                                           Spo
                          combined       Planned      ntaneous

  1    Family.Income  13.657458564  14.324324324  13.196261682
  2    Age.Male.Head  4.0662983425  3.9324324324  4.1588785047
  3   Education.Male  3.7348066298  3.8918918919  3.6261682243
  4 Education.Female  3.6961325967  3.7297297297  3.6728971963
  5       Occupation  3.6353591160  3.2972972973  3.8691588785
  6      Market.Size  5.1933701657  5.5945945946  4.9158878505
  7             Race  1.0055248619  1.0135135135             1
  8     Relationship  5.3812154696  5.0540540541  5.6074766355
  9         Occasion  2.1270718232  2.1081081081  2.1401869159
 10        Influence  2.3591160221  2.1216216216  2.5233644860


 CLASSIFICATION TABLE.  Rows are original groups.
 Columns are assigned groups based on largest score.

                             1        2
                                    Spo
                       Planned ntaneous

  1          Planned        39       35
  2      Spontaneous        32       75

114 of 181 cases (63.0 percent) were correctly assigned.
```

SUMMARY

DISCRIM

```
 DISCRIM  Survey,  FUN  SurvFun,  OUT  SurvProb ;
   Male    Sex  1
   Female  Sex  2
   $
```

The DISCRIM command performs multivariate discriminant analysis, a statistical procedure that uses combinations of variables to distinguish or "discriminate" between two or more groups of cases. Optional output files of discriminant functions, scores and probabilities may be requested.

The groups are defined at the subcommand level. When DISCRIM is used interactively, the user may control the backward stepping procedure.

Required:

DISCRIM fn

specifies the name of the required input file.

Optional Identifiers:

F.DELETE nn

specifies an F value to be used as a criterion for deletion instead of the probability setting that is normally used. (The F value is an absolute value not related to sample size.)

FORCE nn

specifies the forced inclusion of the specified number of variables into the analysis. The argument is the number of variables, counting from the left side of the file. Variable rearrangement may be necessary.

FUN fn

requests an output file of discriminant functions.

IN.FUN fn

requests the classification of cases into groups on the basis of previously computed functions. The name of an input file of discriminant functions created previously is necessary. (FUN is used with DISCRIM prior to this run to produce the file of discriminant functions.)

MAX nn

indicates the maximum number of variables to be kept in the analysis.

MIN nn

indicates the minimum number of variables to be kept in the analysis.

OUT fn

requests an output file of probabilities and discriminant scores.

PRIOR

specifies that the probabilities are from a prior run with group sizes that are different from the current ones. Either current group sizes are used, or arguments (one for each group) are provided that give the current group sizes or ratios:

```
 PRIOR  20 40 40,
```

PROB nn

specifies the probability setting. A variable whose probability associated with its F value is above the probability setting is dropped from the analysis. When PROB is not used, the probability setting is .05. (The probability that a given F value is significant depends on sample size.)

STEP

requests a backwards stepping run. STEP is assumed in interactive mode, but must be specified explicitly in batch mode to request a stepwise analysis. NO STEP is assumed in batch mode.

INTERACTIVE OPTIONS

In an interactive run, the following options are available to control stepping:

1. **H** requests HELP.
2. **Q** quits the command.
3. **L** lists the multivariate F value and probability for each variable.
4. **K** keeps all remaining variables and ends stepping.
5. **W** drops the worst variable and continues stepping.
6. **nn** drops the nnth variable and continues stepping.
7. **A** resumes automatic control.
8. **A,nn** resumes automatic control for nn steps and then returns to interactive control.
9. **R** restores the most recently dropped variable and resumes stepping.
10. **C** produces a classification table.
11. **C,nn** produces a classification table with only those cases whose highest group probability is nn more than the next highest group’s probability.

5
FACTOR:
Factor Analysis and Rotation

Factor analysis refers to techniques for analyzing the interrelationships among variables. The goal is to reduce a large number of variables to a manageable number by grouping related variables into single factors. Related variables are highly correlated variables that describe a single underlying concept or factor. Several P-STAT commands are used in the factor analysis process. The commands and their functions are:

- CORRELATE computes the correlations between pairs of variables.
- CLEANCOR sets the diagonal of the correlation matrix to one and deletes columns or rows with correlations below a specified threshold.
- FACTOR performs an *iterative* or *principal components* factor analysis.
- ROTATE, PROMAX rotate the factors to maximize or minimize the factor loadings.
- GROUPCOR, BPRINT group and display the factors.
- F.COEF computes factor *scores* for individual cases.
- NEWFAC computes factor loadings for variables not included in the original factor analysis and merges them with the original loadings.

5.1 BACKGROUND

The factor analysis model expresses each variable as a function of *factors* common to several variables and a factor unique to that variable. The factor *loadings* are the coefficients:

$$\text{Var}_j = L_{j1}\,\text{Fac}_1 + L_{j2}\,\text{Fac}_2 + ... + L_{jn}\,\text{Fac}_n + \text{Unique}_j$$

There should be a relatively small *number* of factors (*n* in the equation above), and the factor that is unique for each variable should also be small. The factor loadings should be either very large or very small — so that each variable is associated with a minimum number of factors.

The basic steps in factor analysis are to:

1. compute the correlations between the variables,
2. extract the factor loadings,
3. rotate the factors to make the loadings either large or small, and
4. compute the factor scores for each case.

The initial correlations between the variables are computed using the CORRELATE command. There should be some large correlations, or it is not reasonable to use factor analysis. Factor loadings are estimated using either principal components analysis (a common procedure) or iterative factor analysis (the default). The factor loadings range from -1 to +1. They show the degree of association of each variable with the factor.

The factors are rotated to make them more interpretable — that is, either large or small, and not something in between. Factor *scores* are computed for each case. They are the coefficients for each factor, and they give the relative amount of each factor that the case possesses. Factors are generally named after analyzing the variables comprising them. The name is subjective and merely summarizes the common concept underlying the variables. The new factor variables can be used in other analyses.

5.2 FACTORING A MATRIX

FACTOR, the P-STAT command for factor analysis, can perform an iterative or a principal components analysis. The input file for FACTOR is either a correlation matrix produced by the CORRELATE command or a “cleaned” correlation file produced by the CLEANCOR command. CLEANCOR deletes columns and rows, whose largest absolute off-diagonal correlations are all below a supplied threshold, from a correlation matrix . Cleaned correlation matrices have 1’s on the diagonal, a necessary condition if a principal components analysis is to be performed. (See the section in this chapter entitled “Cleaning a Correlation Matrix” for more information on CLEANCOR.)

Figure 5.1 A Factor Analysis

```
FACTOR  Cc,   SF  Fac  $

 FACTOR completed.

ROTATE  Fac,  VF  Vfac  $

 ROTATE of file Fac completed.

GROUPCOR,     INFAC  Vfac,  OUTFAC  Gf,
   INCOR  Cc,  OUTCOR  Gc  $

 GROUPCOR of factorm matrix Vfac
 and correlation matrix Cc completed.
```

Figure 5.1 illustrates a common sequence of commands in a factor analysis. The first command, FACTOR, requires an input file as an argument. The input file must be square and is usually a correlation matrix. The input matrix cannot contain more 440 rows.

FACTOR produces an output file of significant factors when the SF identifier is used. A name for this file is provided as the argument for SF. There is one factor for each significant root. The program iterates until the number of significant roots has been decided and the diagonal estimate becomes stable.

This occurs when the sum of squares of the factor loadings for a variable is within .02 of the diagonal value in the factored correlation matrix. That diagonal value is the sum of squares of the previous factor loadings. The test value of .02 can be changed by using the identifier TEST and supplying a new test value as its argument. When the identifier PC is used, a *principal components* analysis is done. There are no iterations and all roots with a value greater than 1 are considered significant.

The number of factors produced by the FACTOR command can be controlled by the use of the identifiers NFAC, MINFAC, and MAXFAC. NFAC specifies the exact number of roots to be considered significant; MINFAC specifies the minimum number of roots to be considered significant; MAXFAC specifies the maximum number of roots to be considered significant. If none of these identifiers and their numeric arguments are used, the program decides how many roots to include.

After each of the first five iterations (if the factor analysis goes that far), the roots are examined to locate a cutoff point between those that are significant and those that are not. Each pair of roots between 0.7 and 1.1 (the borderline area) is examined to find the largest gap. This gap defines the cutoff point between significant and nonsignificant roots. The factors of the significant roots are then used to reestimate the diagonal and the roots are reexamined for significance. This process continues for five iterations.

There are three optional identifiers to control the values in the starting diagonal. The identifier ONE sets the diagonal of the input file to 1. (ONE should be used if the input file was produced as the output from the CORRELATE command, because the diagonals may not be *exactly* equal to 1.) ZERO sets the diagonal to 0. LARGE sets the diagonal to the largest absolute value in its row and column. In an iterative analysis, the diagonal in the correlation matrix can be replaced by the squared multiple correlations by first using INVERT (see the chapter on Matrix commands) and producing the optional output file associated with the identifier RSQ. That file may then be used as the input file in FACTOR.

5.3 Optional Output Files

An output file of the roots can be obtained by using the identifier ROOTS with a file name as its argument. This file has one row for each variable in the input file and three columns. These columns are the roots, the starting diagonal and the final diagonal. The identifier VEC, with a file name as its argument, produces an output file of vectors. This file is a square matrix with the same number of rows and columns as the input file. The first column of the file is the vector associated with the largest root, and so on.

The identifier FAC, with a file name as its argument, produces a file of factors. This file has one row for each variable in the input file and one column for each of the positive roots. The identifier DROOT, with a file name argument, requests a file with roots in the diagonal and zeros for all the off-diagonal elements. Its dimensions are the same as the input file.

5.4 Cleaning a Correlation Matrix

The CLEANCOR command “cleans” a correlation matrix, producing an output matrix better able to support a factor analysis. All variables whose largest absolute off-diagonal correlation are all below a user-supplied threshold are deleted. Diagonals are set exactly to 1, a prerequisite for a principal components analysis. The CLEANCOR command requires the name of the input file, which is typically a correlation matrix, and a name for an output matrix of correlations:

```
        CLEANCOR  File1Cor,  OUT  File1Cl,  THRESHOLD .2  $
```

The optional identifier THRESHOLD, followed by an argument, provides a deletion threshold. If a threshold value is not supplied, the only variables deleted are those with *all zero* correlations with the other variables. This can occur in a missing data correlation when the data on a variable is completely missing.

5.5 Rotating the Factor Matrix

Sometimes the significant factors produced in the file associated with SF are clear without further processing. Other times however, rotation is used to clarify the factor structure. Three different orthogonal rotations can be done in P-STAT by using the ROTATE command . In Figure 5.1, the input file for the ROTATE command is Fac, and the output file is Vfac, which is given as the argument for the identifier VF. The input file is the file of factors. VF requests an output file of varimax factors. QF could be used to request a file of quartimax factors. EF could be used to request a file of equamax factors.

Three other optional output files may be requested to produce the corresponding transformation matrices: 1) QTR requests an output file containing the transformation matrix from the quartimax rotation; 2) ETR requests an output file containing the transformation matrix from the equamax rotation; and 3) VTR requests an output file containing the transformation matrix from the varimax rotation.

The first step in the rotation process is to normalize the rows of the input matrix so that the sum of squares of each row equals one. Then the variance of the squared elements in each column is computed, and the sum of those variances, one for each column, is saved. After a rotation cycle is performed, a new sum of variances is computed and compared with the previous sum of variances. Iterations cease when the difference of those sums of variances is less than .0001 on three different (not necessarily successive) iterations or when ten iterations have been performed.

The identifier MAX, followed by a numerical argument “nn”, can be used to set a different value for the maximum number of iterations. This number may be between 1 and 50. If MAX nn is used, the program will stop after nn iterations or after the .0001 convergence, whichever comes first. A large (300 by 60) rotation can be quite slow and thus, five or six rotations is usually sufficient.

There are size restrictions in ROTATE associated with the size of P-STAT in use. The number of rows in the input file cannot exceed 3,000. In addition, the product of the rows and the columns cannot exceed 100,000.

Figure 5.2 Factor Analysis Output

```
    BPRINT  Gf,   THRESHOLD .25,  DOTS  $


     PAGE=     1,  THRESHOLD= 0.25
    File Vfac in factor groupings                   FILE=Gf

  POSITION                   1     2
                LABEL     VAR1  VAR2

     1        Education    -80     .
     2         Siblings     48     .
     3       Occupation     46     .
     4              Age     45     .
     5         Children     31     .
     6   Marital.Status      .     .
     7      Work.Status      .    81
     8    Hrs.Last.Week      .   -72
     9              Sex      .    39

     BPRINT  Gc,   THRESHOLD .25,  DOTS  $


     PAge=     1,  THRESHOLD= 0.25
    Correlation matric Cc grouped bv Vfac            FILE=Gc

  POSITION               1     2     3     4     5     6     7     8     9
            LABEL                                  Marit
                                                      al   Wort   Hrs
                      Educ  Sibl Occup        Chil   Sta   Sta  Last
                     ation  ings ation   Age  dren   tus   tus  Week   Sex

 1        Education    100   -38   -47   -38     .     .     .     .     .
 2         Siblings    -38   100     .     .     .     .     .     .     .
 3       Occupation    -47     .   100     .     .     .     .     .     .
 4              Age    -38     .     .   100    29     .     .     .     .
 5         Children      .     .     .    29   100     .     .     .     .
 6   Marital.Status      .     .     .     .     .   100     .     .     .
 7      Work.Status      .     .     .     .     .     .   100   -59    32
 8    Hrs.Last.Week      .     .     .     .     .     .   -59   100     .
 9              Sex      .     .     .     .     .     .    32     .   100
```

5.6 Oblique Rotations

The command PROMAX can be used to do an oblique factor rotation:

```
PROMAX,  INFAC  A,  OUTFAC  B,  COR  C  $
```

The size restrictions for the input file are the same as the size restrictions for ROTATE. The PROMAX command requires the identifier INFAC to designate an input file of factors. Four optional output files may be produced. OUTFAC requests an output file of promax factors. COR requests an output file of the correlations between the factors. TRANS requests an output file containing the transformation matrix. COSINE requests an output file containing a matrix of cosines.

5.7 Grouping and Displaying the Factors

Once the factors have been computed and rotated, the final step is to group and display them in a way that highlights the relationships. The third command in Figure 5.1, GROUPCOR, is used to rearrange the variables in factor and correlation matrices according to their factor loadings.

GROUPCOR requires the identifier INFAC, and the name of the input file of factors as its argument. If the correlations are to be grouped, the identifier INCOR should be used, and the name of the input file of correlations must be supplied as its argument. The identifier for the output file of grouped factors is OUTFAC. The identifier for the output file of grouped correlations is OUTCOR. Each of these identifiers requires a file name as its argument.

In Figure 5.1, INFAC is followed by a file named Vfac, which was created as a result of the previous ROTATE command. In addition, an output file named Gf is created with the OUTFAC identifier. Gf is a file of factors grouped by their factor loadings. Since the correlations are to be grouped, an input file named Cc is defined by using the identifier INCOR. Finally, an output file of grouped correlations named Gc is created following the identifier OUTCOR.

Two separate BPRINT (“Blank Print”) commands are then used to print both the factor groupings and the grouped correlation matrices. In the first BPRINT command in Figure 5.2, the file of factor groupings, Gf, is printed. In the second BPRINT command in Figure 5.2, the grouped correlation matrix, file Gc, is printed. In both BPRINT commands, requests are made for a THRESHOLD of .25 and that DOTS replace the blanks that are present when values are below this threshold. When the less significant correlations and factors are replaced by blanks or decimal points, the significant results are easy to see.

By examining this output carefully, we can conclude that there are two factors present in this analysis. The variables Work.Status, Hrs.Last.Week and Sex are all highly correlated and are grouped together on the factor loadings. Education, Siblings, Occupation, Age and Children belong to the other factor. Marital.Status, on the other hand, does not seem to be related to anything.

5.8 OTHER RELATED COMMANDS

Two related commands use the output files of FACTOR. The F.COEF command requests files containing the factor coefficients and the post multiplier matrix. The NEWFAC command calculates factor loadings for additional variables.

5.9 F.COEF: Factor Coefficients

Factor coefficients or scores can be produced by using the command F.COEF. There are two required identifiers with associated input files: 1) SDATA followed by the name of a file of *standardized* scores, and 2) FACTOR followed by the name of a file containing the factor matrix. OUT is the identifier for the output file of factor coefficients. It is also required.

```
F.COEF,  FACTOR Fac,  SDATA S89SD,  OUT S89FCoef $
```

The file of standardized scores can be obtained by using the STANDARDIZE command. The file of standardized scores can be any size and may contain missing data. If there are missing data, the identifier BADINPUT must be used to indicate what treatment is to be given the missing values. The possibilities are the three arguments, MEAN, MISSING or USEGOOD.

```
F.COEF,  FACTOR Fac,  SDATA S89SD,  OUT S89FCoef,  BADINPUT MEAN $
```

When BADINPUT MEAN is used, an input row with missing data has all of its output scores set to 50 (the arbitrary mean used by F.COEF), unless the additional identifier MEAN is used with a numerical argument to provide a different value. (See MEAN below.) When BADINPUT MISSING is used, an input row with missing data has all scores in the output row set to missing. When BADINPUT USEGOOD is used, the missing input score is skipped and the output scores are computed using all available non-missing scores. However, if the entire input row is missing, the output factor scores for that row are set to missing.

There are limitations on the size of the input FACTOR matrix. For the various sizes of P-STAT, the limits are: 3,000 by 150. The product of the rows and the columns cannot exceed 100,000.

Each variable in the factor matrix must be present in the SDATA file. SDATA can, however, have extra columns that are ignored.

The identifier OUT with a file name as its argument produces a file of factor coefficients. This has as many rows as SDATA and as many columns as FACTOR. Each coefficient has a mean of 50, unless another value is specified with the identifier MEAN (see below), and a variance of about 1.

The optional identifier PMULT produces the post multiplier matrix. If additional data are collected and standardized according to the original means and variances, the new matrix of standardized scores can be multiplied by this post multiplier matrix and factor scores can be obtained for the additional cases much as if they had been in the original file.

The identifier MEAN is used to supply a constant that is added to each factor score when it is computed. Normally this is not used and the program adds the value 50, an arbitrary mean value used by F.COEF. This ensures that the factor coefficients are probably all positive.

5.10 NEWFAC: New Factors

The command NEWFAC solves a least squares estimate for factor loadings on variables that were not included in the original analysis:

```
NEWFAC,  INCOR  A,  INFAC  B,  OUTFAC  C  $
```

All three identifiers are required. INCOR supplies the input correlation matrix that contains correlations for all the variables, new and old. INFAC supplies the input factor loadings for the variables that were in the original analysis. OUTFAC is the output factor matrix. It contains the original factor loadings with the new factor loadings merged on the bottom. The variables in the INFAC matrix can be anywhere in the INCOR matrix and in any order. The input correlation matrix cannot contain more 250 variables. The input factor matrix cannot exceed 60 by 250

SUMMARY

CLEANCOR

```
CLEANCOR  File1Cor,  OUT  File1Cl,  THRESHOLD .2  $
```

The CLEANCOR command “cleans” a correlation matrix, producing an output matrix better able to support a factor analysis. All variables whose largest absolute off-diagonal correlations are all below a user-supplied threshold are deleted. Diagonals are set exactly to 1, a prerequisite for a principal components analysis.

Required:

CLEANCOR fn

provides the name of the input file, typically a correlation matrix.

Required Identifiers:

OUT fn

provides a name for the output matrix of correlations.

Optional Identifiers:

THRESHOLD nn

provides a deletion threshold. If a delete value is not supplied, the only variables deleted are those with all zero correlations with the other variables. This can occur in a missing data correlation when the data on a variable is completely missing.

FACTOR

```
FACTOR  File1Cl,  FAC  File1Fac,
  SF    File1SF,  PC  $
```

The FACTOR command performs an iterative factor analysis or a principal components analysis. Optional output files of factors, significant factors, vectors and roots may be requested. (See the F.COEF command for factor scores.)

Required:

FACTOR fn

provides the name of the input file, typically a correlation matrix.

Optional Identifiers:

DROOT fn

requests an output file containing roots (eigenvalues) in the diagonal and zeros in the off-diagonal elements, and provides a name for the file. The roots are the variance explained by that variable.

FAC fn

requests an output file of factor loadings and provides a name for the file.

LARGE

requests that the diagonal of the input file be set to the largest absolute number in its row and column.

MAXFAC nn

gives the maximum number of factors to be considered significant.

MINFAC nn

gives the minimum number of factors to be considered significant.

ONE

requests that the diagonal of the input file be set to one.

PC

requests a principal components analysis. There are no iterations.

ROOTS fn

requests an output file of roots (eigenvalues) and provides a name for the file.

SF fn

requests an output file of significant factors and provides a name for the file. (Significant factors are sometimes referred to as the factor pattern.)

TEST nn

gives a number to be used as a test of stability. This is normally set to .02.

VEC fn

requests an output file of vectors and provides a name for the file.

NFAC nn

gives the number of factors to be considered significant.

ZERO

requests that the diagonal of the input file be set to zero.

F.COEF

```
F.COEF,  FACTOR  File1Fac,  SDATA  File1Sd,
  FC  File1Fc  $
```

The F.COEF command produces factor coefficients (factor scores) for individual cases. A file of standardized scores and a factor matrix are required input files. This file can be joined with the original file, so that each case has factor scores and its original data.

Required Identifiers:

FACTOR fn

provides the name of the input factor matrix.

SDATA fn

provides the name of the input file of standardized data values (z scores). (The STANDARDIZE command produces standardized scores.)

Optional Identifiers:

BADINPUT MEAN

provides an option to be used when input data are missing. Choices are one of these three arguments: MEAN, MISSING or USEGOOD. When MEAN is used as the argument for BADINPUT, an input row with missing data will have all of its output scores set to 50 (the arbitrary mean used by F.COEF), unless the additional identifier MEAN is used with a numerical argument that provides a different value.

When MISSING is used, an input row with missing data will have all of its output scores set to missing. When USEGOOD is used, any missing scores in an input row are skipped and the output scores are computed using all available non-missing scores. However, if the entire input row is missing, the output factor scores for that row will be set to missing.

FC fn

requests an output file of factor coefficients and provides a name for the file.

MEAN nn

provides a constant to be added to the factor scores, instead of 50, which is used when another value is not provided.

PMULT fn

requests an output file containing the post multiplier output matrix and provides a name for the file.

GROUPCOR

```
 GROUPCOR,  INFAC  File1SF,  OUTFAC  File1SFG  $
```

The GROUPCOR command groups the variables in factor and correlation matrices according to the magnitude of their factor loadings. This aids in identifying groups of related variables. The BPRINT command may be used to print the grouped factor and correlation matrices attractively, with values below a threshold blanked out.

Required Identifiers:

INFAC fn

provides the name of the input file of factors.

Optional Identifiers:

INCOR fn

provides the name of the input correlation matrix. This may have more variables than the INFAC file. Extra variables will be discarded. All the variables in the INFAC file must be present in the INCOR file. INCOR is necessary if OUTCOR is used.

OUTCOR fn

requests an output file of correlations grouped by their factor loadings and provides a name for the file.

OUTFAC fn

requests an output file of factors grouped by their factor loadings, and provides a name for the file.

NEWFAC

```
NEWFAC,   INCOR  File1Cor,  INFAC  File1SF,
  OUTFAC  File2SF  $
```

The NEWFAC command solves a least squares estimate for factor loadings on variables not included in a prior factor analysis.

Required Identifiers:

INCOR fn

provides the name of the input correlation matrix of all the variables, both new and old.

INFAC fn

provides the name of the input factor matrix containing the original factors.

OUTFAC fn

provides a name for the output file containing the factor matrix which will have both the old and the newly estimated factors.

PROMAX

```
PROMAX,  INFAC  File1SF,  OUTFAC  File1P  $
```

The PROMAX command does an oblique factor rotation.

Required Identifiers:

INFAC fn

provides the name of the file of factors.

Optional Identifiers:

COR fn

requests an output file of the correlations between the factors and provides a name for the file.

COSINE fn

requests an output file containing the cosine matrix and provides a name for the file.

OUTFAC fn

requests an output file of promax factors and provides a name for the file.

TRANS fn

requests an output file containing the transformation matrix, and provides a name for the file.

ROTATE

```
 ROTATE  File1SF,  VF  File1VF  $
```

The ROTATE command does orthogonal factor rotations, producing either varimax, quartimax or equamax factors. The corresponding transformation matrices may also be requested.

Required:

ROTATE fn

provides the name of the input file, usually containing factors.

Optional Identifiers:

EF fn

requests an output file of equamax factors and provides a name for the file.

ETR fn

requests an output file containing the transformation matrix from the equamax rotation, and provides a name for the file.

MAX nn

gives the maximum number of iterations to be done. The maximum, when this identifier is not used, is ten iterations.

QF fn

requests an output file of quartimax factors and provides a name for the file.

QTR fn

requests an output file containing the transformation matrix from the quartimax rotation and provides a name for the file.

VF fn

requests an output file of varimax factors and provides a name for the file. VF is one of the most frequently used rotations.

VTR fn

requests an output file containing the transformation matrix from the varimax rotation and provides a name for the file.

6
ANOVA:
Analysis of Variance

Analysis of Variance (ANOVA) is a powerful statistical procedure that analyzes experimental data, permitting an experimenter to determine whether or not a classification or treatment has an effect on a measured response. ANOVA divides the observed variability in continuous response variables into parts, each attributable to a specified experimental classification or treatment, or to random error. If the variability associated with a classification or treatment is more than expected based on the general variability of the experimental population, then you could infer that the classification or treatment has an effect.

P-STAT's ANOVA command is interactive, although it may also be used in batch mode. Thus, you may specify your experimental design, view the analysis, modify the design specification if desired, and again view the resultant analysis. A wide variety of both balanced and unbalanced designs may be analyzed. Any confounding of effects will be detected and reported in the output table.

6.1 INTRODUCTION

A simple example will help explain the concepts involved, the terminology used, and the way to express the experimental design to the computer. Suppose an experiment is run to see if a new type of laboratory feed (Feed 3) is better for mice than the two usual feeds (Feeds 1 and 2). The mice are randomly divided into three groups and each group is given one of the feeds. After two weeks, the mice are weighed to determine if the average weights of the groups are different. Figure 6.1 shows hypothetical data collected for 23 mice.

Analysis of variance allows you to determine if the variability in the average weight of the mice in the different groups was due to their different feeds, or if it just reflected the normal variability that can be found when one weighs a number of mice.

The initial hypothesis might be that there will be no significant difference in the average weight of each group of mice. The means of the groups will be effectively equal, and they will be representative of the mean weight of the general mouse population. Analysis of variance tests this hypothesis and permits you to determine whether or not it should be rejected. Rejecting the hypothesis implies concluding that the means of the three groups are not equal — that the three groups of mice have different weights. One or more of the groups may have an average weight significantly different from that of the general mouse population. The inference is that this is due to the different diets.

6.2 ANOVA Vocabulary

The individual mice are the subjects or units in this experiment. Their weight is the “dependent” or the “response” variable. The feed is the “treatment” or “classification” factor — the mice are classified into three groups by the FEED treatment factor.

The variability in the mean weights of the mice eating Feed 1, Feed 2, and Feed 3 is the “between-group variance. ” The initial or “null hypothesis” is a statement that there is no significant difference in the average weight of the three groups of mice, and that this between-group variance is small and equivalent to that of the general mouse population.

Figure 6.1 Weights (In Grams) for Mice on Three Feeds

```
LIST  Mice,  CONTROL  Feed  $

     Feed   Weight

        1      120
        1      126
        1      119
        1      126
        1      118
        1      123
        1      125

        2      123
        2      119
        2      126
        2      117
        2      122
        2      125
        2      118
        2      120

        3      125
        3      128
        3      124
        3      130
        3      132
        3      129
        3      127
        3      129
```

There are also other sources of variability affecting the weights of the mice: some mice grow more vigorously, some may be genetically larger than others, some may have a warmer location in the laboratory, and so on. You try to reduce this variability by using just one strain of mice, by choosing mice of the same age, by placing their cages in the same location, and by other means. However, this natural variability cannot be eliminated altogether, and it does influence the weights of the mice. This experimentally uncontrollable variance is the “within-group variance” — it occurs from mouse to mouse, and not from group to group. It is also often called the “residual variance” or the “experimental error.”

6.3 Using P-STAT ANOVA

To use P-STAT ANOVA to analyze your data, you tell it the name of the P-STAT file containing your data:

```
ANOVA  Mice ;
```

Then you tell it the MODEL or relationship you hypothesize will explain any effects associated with the experimental treatment:

```
MODEL  Weight  =  Feed ;
```

“Weight” is the dependent variable (what was measured). “Feed” is the treatment variable — it is the factor you think may be associated with the variability in the weight measurements.

The key words on the left of each statement, ANOVA and MODEL, are always the same. The ones on the right are the names of your data file and your variables. An equal-sign (=) is required in the MODEL statement. It is necessary to separate the dependent variable or variables from the treatment variable or variables. The semi-colon (;) is necessary after the ANOVA command. The ANOVA program then prompts for subcommands (except in batch mode). The MODEL statement is the first subcommand. After each subcommand, the analysis requested is done and immediately displayed. Further analyses with various changes or options are possible. The ANOVA output for this simple example is shown in Figure 6.2.

Figure 6.2 Output for Simple Analysis of Mouse Weight

The command and subcommands

```
ANOVA  Mice ;
MODEL   Weight = Feed ;
MEANS;
$
```

The printed output

```
              ANALYSIS OF VARIANCE:   Weight

 SOURCE OF VARIATION     DF     SS      MS       F    Pr > F

 Feed                     2   205.74  102.87   10.65  0.0007
 Residual                20   193.21    9.66

 Adjusted Total          22   398.96

  GRAND MEAN                   123.96
 Number of Observations           23




  CELL CONTENTS ARE....
    CELL COUNTS
 ---MEAN SCORE OF VARIABLE  -----Weight-----
                                            ROW
                 1        2        3      TOTALS
            --------------------------
 Feed      |     7        8        8  |     23
           |122.43   121.25   128.00  | 123.96
            --------------------------
```

The table of arithmetic means, also shown in this figure, is produced by the optional request:

```
MEANS ;
```

Because ANOVA is fully conversational, means may be requested as part of the original request or after the initial output. A $ ends the whole ANOVA command. Note that it is not necessary to have the same number of mice (subjects) in each group; that is, the data do not need to be balanced.

6.4 BASIC ANOVA DESIGNS

Different experimental designs can be used to analyze the effects or interactions of various factors. The following sections discuss many of the basic designs and the way to specify them to the ANOVA program.

6.5 Completely Randomized Design

The mouse experiment is an example of a completely randomized design — the mice were randomly assigned to the three feed groups. This design is also called a “one-factor” design or “one-way analysis of variance” because only one treatment is involved. The important aspect of this design, and of most others as well, is that the subjects (mice) are randomly assigned to the treatment levels (groups). The treatment levels, however, are considered to be fixed ones, rather than random ones. The three feeds are chosen by the experimenter. This is a common aspect of most experimental designs — subjects are randomly assigned to fixed treatment levels.

The mean weights of mice eating Feed 1, Feed 2, and Feed 3 are tested for equality. The statements in the following box are used in the completely randomized or one-factor design. The difference in feeds is the only known source of any possible variation. Residual variation is variation from unknown sources — experimental error.

1. COMPLETELY RANDOMIZED DESIGN

INPUT:

```
    ANOVA Mice ;
      MODEL Weight = Feed ;
```

OUTPUT:

```
    ANALYSIS OF VARIANCE: Weight
         SOURCE OF VARIATION
         Feed
         Residual
```

6.6 F Ratio

An F ratio is used to test whether the mean weights of the mice on the three types of feed are significantly different. This ratio compares the between-group variance with the residual variance (within-group or within-cells variance here).[1] More specifically, the F ratio compares the between-group mean square (MS in the ANOVA table) for Feed with the residual mean square. The between-group or Feed mean square is the sum of the squares (SS in the

[1] The residual variance (here, the within-group variance) is used as the denominator or “error term” in the F ratio. This is generally correct for most simple experimental designs. However, for more complex designs, error terms of varying complexity may (and should) be specified.

ANOVA table) of the deviations of each group mean from the grand mean, divided by its degrees of freedom (DF in the ANOVA table). The within-group or residual mean square is the sum of the squares of the deviations of each response from its group mean, divided by its degrees of freedom. Degrees of freedom correspond to the number of deviations about a mean free to vary. Generally, one deviation cannot vary because the total of the deviations about a mean must add up to zero. The mean square is a measure of average variability; it is a variance estimate. We will sometimes speak of it as a variance, however, for simplicity.

The value of F in the analysis of variance in Figure 6.2 is 10.65. Pr is the probability level associated with this F. In Figure 6.2, Pr = 0.0007, which is .07%. This can be interpreted as follows: When all the groups of mice are given the same feed, an F value this large or larger will be found only .07% of the time. This is a very small percentage. Therefore, you can conclude that obtaining such a large F value, which is a very rare occurrence when all mice eat the same feed, means that the mice must not be eating the same feeds — and that this must be having a differential effect on weight gain. Generally, an experimenter feels secure in rejecting the null hypothesis, and assuming that differences exist, when Pr is .05 (5%) or less.

The analysis of variance table lets you know if there are any detectable differences. However, it does not tell you which level of the treatment factor is responsible for the differences, nor in which direction the differences are found. Inspection of the table of means usually provides this information, but formal tests are often necessary. In Figure 6.2, the table of means indicates that Feed 3 (the new feed) produced a higher average weight than the other feeds. There most likely is no difference in average weight produced by Feeds 1 and 2.

When there are only two levels of the completely randomized or one-factor design, either ANOVA or a t test (available in the FREQ command) may be used. Either procedure will lead to the same conclusion because, with two levels of a single treatment factor, the square root of F equals t.

Figure 6.3 Design for a 2 X 3 Factorial Experiment

```
                                        TEXTBOOK

W                              Text 1      Text 2      Text 3
O   O                        ----------------------------------
R   P      Use of           |           |           |           |
K   T      Workbook         |  Class 5  |  Class 2  |  Class 4  |
B   I                       |-----------|-----------|-----------|
O   O      Non-Use of       |           |           |           |
O   N      Workbook         |  Class 6  |  Class 1  |  Class 3  |
K                            ----------------------------------
```

6.7 Factorial Design

The factorial design is similar to the completely randomized or one-factor design, except that two or more treatment factors are considered simultaneously. Subjects are randomly assigned to the different fixed treatment groups. The simplest design involves several levels of two treatments, such as a 3 x 2 factorial design.

An example of this is an educational experiment involving three different math textbooks and the use or non-use of the corresponding workbook. Students would be randomly assigned to one of these six groups, and those students using one text and its workbook would be compared with students using that text without its workbook, and similarly compared with students using the second and third texts with and without their workbooks. The experimental layout for this factorial design might be the one shown in Figure 6.3.

Scores on the final examination would be the dependent variable, and the analysis of variance table would show the variation due to the text, due to the workbook option, due to an interaction between them, and due to unknown sources (residual variance). The statements in the following box would be used for this design. The residual variance is used as the error term.

```
2. FACTORIAL DESIGN

INPUT:

   ANOVA Math ;
     MODEL Test.Score = Text | Workbook ;

OUTPUT:

   ANALYSIS OF VARIANCE: Test.Score
        SOURCE OF VARIATION
        Text
        Workbook
        Text*Workbook
        Residual
```

The vertical bar symbol (|) indicates “crossing” to the ANOVA command. (For terminals without a vertical bar character, the exclamation point (!) can be used instead.)

In a factorial design, the factors cross. This design could also be specified with the subcommand:

```
MODEL  Test.Score  =
  Text  +  Workbook  +  Text*Workbook ;
```

which contains the expansion of the Text | Workbook design statement. Text and Workbook are “main effects” and are specified by name. A “+” separates the effects from each other. Text*Workbook is an “interaction effect” between the Text and the Workbook effects. The “*” defines the interaction effect as the product of these main effects.

There may be many treatment factors, as well as many levels of each factor, in a factorial design. An example of multiple factors, each with multiple levels, would be a design testing the effects of three textbooks, with two workbook options, with students in grades four through eight. This would be a 3 x 2 x 5 three-factor design; there are three levels of the textbook factor, two levels of the workbook option factor, and five levels of the grade factor. An example in agriculture would be a test of the effects of four types of fertilizer on three varieties of corn. The resultant yields on the 12 different fields of corn would be measured. This would be a 4 x 3 two-factor factorial design with 12 groups.

6.8 COMPLEX ANOVA DESIGNS

The simpler ANOVA designs use the within-group variance (also called the residual variance) as the error term in the F ratio. The more complex ANOVA designs permit the specification of block and error term strata to reflect the grouping of the units or subjects into factors and levels.

6.9 Randomized Block Design

The randomized block design increases the accuracy of an experiment by providing some control over the diversity of the experimental subjects. This is done by grouping the subjects on the basis of some factor that is suspected of varying fairly widely in the subject population. The testing of additional factors and effects, the ones of prime interest, is thus more accurate because it is within more homogeneous groups.

For example, an experimenter studying the effect of candy on the activity level of children may want to take into account the natural activity differences between boys and girls. Therefore, he or she may classify the children into two groups by sex. An interaction between sex and the level of the candy factor is not expected, since the amount of candy is expected to affect the activity level the same amount for both sexes. If such an interaction was expected, a factorial design would be a better choice.

Structurally, the randomized block design has the appearance of a factorial design, where one of the factors is always a classification variable, as opposed to a treatment variable. This factor is the “blocking” variable. It takes into account the fact that the experimental subjects have inherent variations due to their particular sex. The levels of the blocking factor (Sex) would be boys and girls, and the levels of the treatment factor (Candy) might be none, low and high. The levels (boys and girls) of the blocking factor (Sex) are fixed ones, as are the levels (none, low and high) of the treatment factor (Candy). However, youngsters within a given block, boys for example, are randomly assigned to each of the three treatment groups. The layout for this experiment would look like the one in Figure 6.4.

Figure 6.4 Design for Randomized Block Experiment

```
                 CANDY

                    None      Low     High
                 -----------------------------
B               |         |         |         |
L    Boys       |  Gp B2  |  Gp B3  |  Gp B1  |
O               |---------|---------|---------|
C               |         |         |         |
K    Girls      |  Gp G3  |  Gp G1  |  Gp G2  |
                 -----------------------------
```

Experimental error (the residual or within-group variance) is reduced by classifying the youngsters by sex, rather than by classroom, for example, because the resulting groups are believed to be more homogeneous. The variation associated with the blocking variable is absorbed or effectively removed from the experimental error (residual variance). Therefore, the test for treatment effects, dependent on the ratio of the treatment variance to the residual variance, will be more sensitive.

The P-STAT statements for the analysis of this randomized block design are shown in the following box (Box 3). The residual variance is used as the error term. The MODEL phrase shows the dependent variable and the effects associated with its variability. The BLOCK phrase indicates the blocking factor or effect.

The sources of variation in the ANOVA output are those due to the blocking variable, the treatment variable, and the residual variance. The blocking variable, Sex, appears first, flush left. It is the name of the blocking stratum in this design. The name, “Within Sex”, appears second, also flush left. It is the name of the error term stratum in this design. The treatment variable, Candy, appears next, indented in its error term stratum. The error term, the denominator in the F ratio, is the residual of the Within Sex stratum. It appears last, also indented.

The sources of variation in the ANOVA output are those due to the blocking variable, the treatment variable, and the residual variance. The blocking variable, Sex, appears first, flush left. It is the name of the blocking stratum in this design. The name, “Within Sex”, appears second, also flush left. It is the name of the error term stratum in this design. The treatment variable, Candy, appears next, indented in its error term stratum. The error term, the denominator in the F ratio, is the residual of the Within Sex stratum. It appears last, also indented.

```
3. RANDOMIZED BLOCK DESIGN

INPUT:

    ANOVA Candy.Experiment ;
      MODEL Activity.Measure = Candy + Sex,
      BLOCK Sex ;

OUTPUT:

    ANALYSIS OF VARIANCE: Activity.Measure
         SOURCE OF VARIATION
         Sex
         Within Sex
           Candy
           Residual
```

6.10 Blocking and Error Term Strata

The division of the ANOVA table into strata reflects the grouping of experimental units (subjects) into factors and levels. Blocking strata and error term strata are identically defined, except for the way they are handled by the ANOVA command. The sum of squares for an error stratum is partitioned into sums of squares for treatments (whenever that is possible) and a residual sum of squares for that stratum. The sum of squares for a blocking stratum could (possibly) be similarly partitioned, but the specification as a blocking stratum, rather than an error stratum, tells the program not to attempt the partition.

Blocking strata sums of squares are calculated first by the program and appear first in the output. They are followed by the error term strata sums of squares, which are partitioned into treatments and residual. The treatment sums of squares within each stratum are tested against the residual of that stratum. The stratum sums of squares and the residual sums of squares are tested against the residual sum of squares in the bottom-most residual. Their F ratios appear in the extreme right columns. The sum of squares and the residual sum of squares in the bottom-most stratum are not themselves tested. (See Appendix C, at the end of this chapter, which shows a complex example. The output table shows three error term strata, with their corresponding treatments (if any) and their residuals.)

The F value for Candy shows whether there is a significant difference in the activity levels of the groups with no candy, small amounts and large amounts of candy. The F value for Sex shows if the means are different for the two sexes. If there is a significant difference between boys and girls, it would confirm the choice of this design with the blocking by sex. Without that blocking, the variation within any mixed-sex group would have been higher and would have artificially inflated the experimental error.

6.11 Latin Square Design

The Latin square design is one of the most common of a group known as incomplete factorial designs. (This group also includes the fractional factorial design and the Greco-Latin design.) These designs are often more feasible when a large number of treatment factors are to be studied simultaneously, because they reduce the required size of the experiment — only a fraction of the possible combinations of factors is tested.

The Latin square design involves three treatment factors and k levels of each factor; 1/k of the k-cubed possible combinations are tested. For example, an experiment using a Latin square design might involve three treatments affecting the achievement of ninth graders in algebra, each with three fixed levels: the teaching approach (traditional lecture, self-paced individual, small group), the teacher (Mrs. White, Mr. Smith, Ms. Cable), and the time of the school day the class was taught (period one, period three, period five). The achievement of the students in algebra could be measured by a standardized test at the end of the year. One example of the Latin square design for this experiment would look like Figure 6.5.

Figure 6.5 Latin Square Design for an Educational Experiment

		TEACHER		
		White	Smith	Cable
TIME OF DAY	Period 1	Lecture	Self Paced	Small Group
	Period 3	Self Paced	Small Group	Lecture
	Period 5	Small Group	Lecture	Self Paced

One-third of the 27 possible combinations of the three factors in a complete 3 x 3 x 3 design is represented in Figure 6.5. A complete balance of main effects is obtained by applying each level of the teaching Approach factor — the treatment of most interest — at random, so that each occurs once in each row and once in each column. For example, this could be done by choosing a standard Latin square design, available in many experimental design texts, and using a table of random numbers to reorder the rows and columns. Students are then randomly assigned to each of the nine groups.

When the treatment means for the different teaching approaches are compared, the variability among the periods (rows) and the variability among the teachers (columns) are filtered out. One must assume that no interactions exist — none between Teacher and Time.of.Day, Approach and Time.of.Day, or Approach and Teacher. This is the “price paid” for the ability to reduce the required number of groups by using a Latin square design.

```
4. LATIN SQUARE DESIGN

INPUT:
    ANOVA Grade ;
      MODEL Algebra.Test = Approach + Teacher + Time.of.Day,
      BLOCK Teacher + Time.of.Day ;

OUTPUT:

    ANALYSIS OF VARIANCE: Algebra.Test
         SOURCE OF VARIATION
         Teacher Time.of.Day
         Within Teacher*Time.of.Day
           Approach
           Residual
```

The Latin square design may be thought of as a design with two blocking factors, the row treatment factor and the column treatment factor, because interactions are of no concern. Blocking by Time.of.Day and by Teacher will provide more homogeneous groups and thus reduce the residual variance. The test for the (teaching) Approach treatment, the one of prime interest here, will then be more sensitive. The F values on the ANOVA output would show tests for differences in algebra scores for the blocking factors (Time.of.Day and Teacher) and for the treatment factor (Approach). The tests for the block strata would appear first. The test for the treatment would appear next, indented in its error term stratum, Within Teacher*Time.of.Day. The P-STAT statements in the following box would be used for this design. The error term is the residual variance.

An alternate method of specifying this design is:

```
DEPENDENT  Algebra.Test,  TREATMENT  Approach,
  BLOCK    Teacher  +  Time.of.Day ;
```

The MODEL phrase is replaced by equivalent phrases here. The dependent variable is indicated by the DEPENDENT variable phrase, and the blocking effects by the BLOCK design phrase. The TREATMENT design phrase indicates the treatment effect. Identical analyses are obtained with either method of design specification. The full words, DEPENDENT, TREATMENT and BLOCK, or their respective abbreviations, DEP, TR and BL, can be used.

6.12 Nested Design

In research, it is often not possible to cross each treatment or classification factor completely. Yet, these factors usually have an effect on the outcome of the experiment that should be taken into account in the design and subsequent analysis of the experiment.

Consider a hypothetical industrial experiment undertaken to compare the productivity of six factories, each with a different management style exemplified by either a Japanese or an American management team. Each factory has American workers. It is not practical to cross each factory with each management team because of the physical separation of the factories and the period of time necessary for six successive groupings. Therefore, a nested design is chosen for this experiment.

The Managemt.Style treatment factor has two fixed levels — Japanese and American. The Factory treatment factor is “nested” within the Managemt.Style factor. This means that only some of the factories are within each of the levels of Managemt.Style. The Factory factor has six random levels — factories F1 through F6. They are randomly assigned to one of the two levels of the Managemt.Style factor. Productivity at each factory would be measured; for example, automobiles produced per week, would be measured over a multi-week period. The design layout for this experiment might look like the one pictured in Figure 6.6.

Figure 6.6 Nested Design for an Industrial Experiment

```
                        MANAGEMENT STYLE


                  Japanese                American
                --------------          --------------

Factories:      F4     F1     F6        F3     F2     F5
```

It is necessary to explicitly specify an error term in this nested design; an error term is the denominator that should be used in the F ratio. The residual variance is used as the denominator when another error term is not specified, but this is not correct here.[2] The statements used to specify this nested design with random levels of the nested factor and fixed levels of the main factor and to specify the appropriate error term are in the following box.

The MODEL phrase states the hypothesis that automobiles produced per week is associated with Managemt.Style nests Factory. The slash (/) indicates this nesting. The design statement:

```
Managemt.Style  /  Factory,
```

is equivalent to:

```
Managemt.Style  +  Factory(Managemt.Style),
```

In words, “Managemt.Style nests Factory” is equivalent to “the sum of the Managemt.Style effect and the Factory nested within Managemt.Style effect.”

The ERROR.TERM design phrase specifies the denominator to be used in the F ratio testing Managemt.Style :

```
ERROR.TERM  Factory(Managemt.Style) ;
```

5. NESTED DESIGN

INPUT:

```
ANOVA Managemt ;
  MODEL Autos.Per.Week = Managemt.Style / Factory,
  ERROR.TERM Factory(Managemt.Style) ;
```

OUTPUT:

```
ANALYSIS OF VARIANCE: Autos.Per.Week
     SOURCE OF VARIATION
     Factory(Managemt.Style)
       Managemt.Style
       Residual
     Within Factory(Managemt.Style)
       Residual
```

The parentheses indicate the nested relationship of Factory within Managemt.Style. Tests of effects appear in the appropriate error term strata. The sources of variation shown on the ANOVA table would be for these two effects: Factory(Managemt.Style) and Managemt.Style. Managemt.Style is tested against the residual in the Factory(Managemt.Style) stratum, and the Factory(Managemt.Style) effect is tested against the bottom-most residual.

It is not possible to obtain information on interaction effects involving the nested variable, Factory, and the other treatment factor, Managemt.Style, in which it is nested. This is because each factory is involved in only one type of treatment. Generally, the interactions involving nested variables are not of particular interest to the researcher. Should they be, a factorial or crossed design should be used.

[2] The residual variance is often the within-cells variance, but it may also include additional variances, such as that from a potentially identifiable interaction effect that is not part of the treatment. The residual variance is used as the error term (denominator for the F ratios) when another error term is not explicitly specified.

6.13 Split-Plot Design

A split-plot design is often used in agricultural research. It may be more easily understood by noting its equivalence to a partially nested design. A typical split-plot design might appear as in Figure 6.7.

Figure 6.7 A Typical Split-Plot Design

```
               Plot 1          Plot 2          Plot 3          Plot 4
               ------------------------------------------------------
                                     FERTILIZER
                (F2)            (F1)            (F1)            (F2)


         (W1)    SP1             SP3             SP2             SP2
WEED
INHIB-   (W2)    SP2             SP1             SP3             SP1
ITOR
         (W3)    SP3             SP2             SP1             SP3
```

There are four plots of land (columns) to be used in an agricultural experiment in this design. There are two treatments, Fertilizer and Weed Inhibitor. The levels of Fertilizer (F1 and F2) are randomly assigned to the plots. Each plot is divided into three subplots (SP1 to SP3). The levels of Weed Inhibitor (W1 to W3) are randomly assigned to the subplots within the levels of Fertilizer. This split-plot design is equivalent to the partially nested design shown in Figure 6.8.

Figure 6.8 A Split-Plot Design as a Partially Nested Design

```
                                        FERTILIZER
                             (F1)                       (F2)
                        --------------             --------------
PLOTS:                   2          3               1          4

          ----------------------------------------------------------

          (W1)          SP3        SP2            SP1         SP2
WEED
INHIB-    (W2)          SP1        SP3            SP2         SP1
ITOR
          (W3)          SP2        SP1            SP3         SP3
```

Plots 2 and 3 are nested within level 1 of the Fertilizer treatment, and plots 1 and 4 are nested within level 2. This is a partially nested design because there is no nesting within the Weed Inhibitor treatment. Fertilizer is crossed with Weed Inhibitor and, because of the nesting of plot within Fertilizer, each plot is crossed with Weed Inhibitor. The subplots (SP1 to SP3) merely correspond to groups of subjects within the context of nested designs, except that there is only one response measurement per subplot. The ANOVA statements that could specify this design are in the following box.

```
6. SPLIT-PLOT DESIGN

INPUT:

    ANOVA Agricul ;
      MODEL Yield = (Fertliz / Plot) | Weed.Inhib,
      ERROR.TERM  Plot(Fertliz) + Weed.Inhib*Plot(Fertliz) ;

OUTPUT:

    ANALYSIS OF VARIANCE: Yield
         SOURCE OF VARIATION
         Plot(Fertliz)
           Fertliz
           Residual
         Weed.Inhib*Plot(Fertliz)
           Weed.Inhib
           Fertliz*Weed.Inhib
           Residual
```

The MODEL phrase tells ANOVA that Fertilizer nest Plot is crossed with Weed Inhibitor . The ERROR.TERM design phrase tells ANOVA the denominators to use in the F ratios: the Plot effect nested under Fertilizer, and the Weed Inhibitor and Plot interaction, also nested under Fertilizer. When multiple error terms are specified, the ANOVA command ascertains which treatments are to be tested against which error terms. It can do this because the command notes the linear dependencies among the factors specified in the ERROR.TERM and TREATMENT phrases. Under the source of variation, tests for the main effects of Fertilizer and Weed Inhibitor, and for the interaction between them, are shown in their appropriate error strata. There is no “Within” error stratum in this design output because the within-cells stratum has been explicitly defined as “Weed.Inhib*Plot(Fertliz)”. Within that stratum is just one dependent measurement per cell — yield per subplot, and hence, no further subdivision is possible.

An alternate method of specifying this design, which would have a “Within” stratum, is:

```
TR  Fertilizer | Weed.Inhib,  ER  Plot ;
```

(Further discussion of the specification of split-plot designs in this manner appears in the following section: “Specifying Complex Designs Interactively — Implicit Nesting”.)

6.14 Repeated Measures Design

The same group of experimental subjects is used under multiple levels of a treatment in a repeated measures design. The subjects experience two or more treatment levels in turn.

Repeated measures designs are used primarily for three reasons. First, they “match” a subject with himself or herself and thus provide highly homogeneous subject groups. Second, they reduce the number of subjects necessary in an experiment, since each subject is utilized more than once. Third, repeated measures designs are necessary when one of the treatments is the passage of time, as in experiments measuring forgetting or physical growth.

7. REPEATED MEASURES DESIGN

INPUT:

```
    ANOVA Hospital
      ( SPLIT INTO 5,
        CARRY Medicine Patient,
        INDEX Day 5,
        CREATE Temp Day.1 TO Day.5 ) ;

      MODEL Temp  = (Medicine / Patient) | Day,
      ERROR.TERM  Patient(Medicine) + Day*Patient(Medicine) ;
```

OUTPUT:

```
    ANALYSIS OF VARIANCE: Temp
         SOURCE OF VARIATION
         Patient(Medicine)
           Medicine
           Residual
         Day*Patient(Medicine)
           Day
           Day*Medicine
           Residual
```

There are, however, several major disadvantages to this type of design. All treatments cannot be applied simultaneously, since the same group of subjects must appear in two or more treatments. Also, the passage of time may change some environmental conditions by the time the last treatment is applied — carry over in learning or deterioration and fatiguing may occur.

Consider an experiment in a hospital designed to determine the effect of various doses of a medicine on the temperature of postoperative patients. A repeated measures factorial design could be used so that the effects of the medicine and of the passage of time could be determined. The design for this experiment could be the one shown in Figure 6.9. Each row across, “t1, t2, t3, t4, t5,” represents temperature measurements over a five-day postoperative period for one patient randomly assigned to one of the levels of the medicine treatment. Note that patients are nested within the levels of medicine, but crossed with the levels of day.

The data for an experiment need to be in a format that clearly indicates the different treatment levels to the ANOVA program. While the data organization displayed here describes the treatment levels well to the experimenter, the ANOVA program requires that they be arranged with a single dependent variable and all of its classification factors individually specified. SPLIT, a function in the P-STAT programming language (explained in the following section), will make the necessary conversion as the data file is input into ANOVA. The P-STAT statements that would be used are shown in the preceding box.

Tests for the main effects of Medicine and of Day, and for an interaction between them, are shown in their appropriate error strata under source of variation. As in the previous split-plot design example, there is no “Within” stratum in the output table because that stratum has been explicitly defined as “Day*Patient(Medicine).” Within that stratum is just one dependent measurement per cell — each patient temperature.

An alternate method of specifying this design is:

```
     TR  Medicine | Day,  ER  Patient ;
```

(Designs of this type are further discussed in the section: “Specifying Complex Designs Interactively — Implicit Nesting”.)

Repeated measures may be in any design or part of any design — the completely randomized design, the factorial design, the randomized block design, the nested design, and others.

Figure 6.9 Repeated Measures Design for a Hospital Experiment

```
                                       DAY

            Patient      Day.1    Day.2    Day.3    Day.4    Day.5
          ------------------------------------------------------------

               101         t1       t2       t3       t4       t5
               102         t1       t2       t3       t4       t5
    1           .           .        .        .        .        .
                .           .        .        .        .        .
               10n         t1       t2       t3       t4       t5


M
E              201         t1       t2       t3       t4       t5
D              202         t1       t2       t3       t4       t5
I   2           .           .        .        .        .        .
C               .           .        .        .        .        .
I              20n         t1       t2       t3       t4       t5
N
E
               301         t1       t2       t3       t4       t5
               302         t1       t2       t3       t4       t5
    3           .           .        .        .        .        .
                .           .        .        .        .        .
               30n         t1       t2       t3       t4       t5
```

6.15 Basic Assumptions in ANOVA

There are some basic assumptions underlying analysis of variance in general, and repeated measures designs in specific. A brief digression to mention these assumptions is in order. Experimental design as a subject deals with the problem of structuring observations so that the comparison of typical values from a number of well defined cells will give information on the effects of the classification factors that define those cells.

The usual arithmetic used to make the comparisons is analysis of variance, and the usual basis for the comparison is the F distribution. These require a set of assumptions to be made by the experimenter. The principal assumption is that the data follow a linear model, expressed in terms of the discrete levels of the classification and treatment factor levels. The deviation of the actual observations from this model (error) is assumed to be independently, randomly distributed from the normal distribution with a common variance.

The necessary arithmetic can always be done to a set of data. However, the validity of interpretations or inferences from the arithmetic is not always assured. Generally, the assumptions cannot be verified with a finite sample. The residuals can be checked for approximate consistency with the assumptions and any potential gross violations. Any systematic behavior evidenced in plots of the residuals would suggest that additional information might be gained from further analysis of the data.

The assumption of normally distributed errors can sometimes be relaxed when there is random assignment of treatments to the experimental subjects or units. Proper randomization provides justification for the use of the F distribution.

The common variance assumption is often extended to a more general assumption of common variances and covariances. In more complex designs, there are different variances associated with the various strata (groupings), and common values of correlation among observations within the same level of a stratum. This assumption has relevance to repeated measures designs, where a constant correlation among responses to successive treatment levels is assumed.

6.16 DATA ARRANGEMENT, TERMINOLOGY

The arrangement of data for analysis and the terminology for specifying models is common to all ANOVA designs. Data may be rearranged into the form required by the ANOVA command by using the SPLIT instruction, alone or in conjunction with other PPL instructions and functions. Precise definitions of the terminology used in the ANOVA language will clarify the subsequent discussion on interactive specification.

Figure 6.10 Using SPLIT To Rearrange Data

```
LIST  Hospital
        [ SPLIT INTO 5,  CARRY Medicine  Patient,
          INDEX  Day 5,  CREATE Temperature  Day.1 TO Day.5 ]  $

  Medicine      Patient   Day   Temperature
         1          101     1         101.2
         1          101     2          99.7
         1          101     3          99.6
         1          101     4          98.7
         1          101     5          98.1

         1          102     1         102.5
         1          102     2         101.9
         1          102     3          99.2
         1          102     4          98.6
         1          102     5          98.7

         .            .     .             .
         .            .     .             .
         .            .     .             .

         3          30n     1         102.6
         3          30n     2         101.7
         3          30n     3         100.2
         3          30n     4          99.5
         3          30n     5          99.6
```

6.17 Arranging Data With Split

The data collected in an experiment is often not in the exact form you need for the ANOVA program. However, changing the arrangement of that data is simple and can be done at the same time the data file is input to ANOVA.

For example, the data for the mice experiment, shown in Figure 6.1, may have originally been collected in a file like this: [3]

Feed	W1	W2	W3	W4	W5	W6	W7	W8
1	120	126	119	126	118	123	125	-
2	123	130	126	128	120	125	128	127
3	125	128	124	130	132	129	127	129

SPLIT, an instruction in the P-STAT programming language, can be used to rearrange this data. You use SPLIT as the data are input to ANOVA:

```
ANOVA   Mice
  [ SPLIT   INTO 8,  CARRY  Feed,
    CREATE  Weight   W1 TO W8  ] ;
```

Your data then appear as they do in Figure 6.1.

The phrase, "SPLIT INTO 8," tells P-STAT that you want to split each row of your data file into eight separate rows. The phrase, "CARRY Feed," means to carry the value of Feed across all eight rows with its corresponding values of Weight. The phrase, "CREATE Weight W1 to W8 ," tells P-STAT the variable name you want for these measurements. "Weight" will be this name for all the measurements that were previously called "W1" to "W8".

If your data had originally been in a slightly different arrangement, such as the following one:

Feed.1	Feed.2	Feed.3
120	123	125
126	130	128
119	126	124
126	128	130
118	120	132
123	125	129
125	128	127
-	127	129

you would again use SPLIT, but with slightly different specifications:

```
ANOVA   Mice
  [ SPLIT   INTO 3,  INDEX  Feed 3,
    CREATE  Weight   Feed.1  TO  Feed.3 ] ;
```

Your data would again appear as in Figure 6.1. The phrase, "INDEX Feed 3," tells P-STAT to create a new variable that will be named "Feed" and have values going from 1 to 3. The first value, 1, will go with the first measurement, 120, and the second value, 2, will go with the second measurement, 123, and so on. The phrase, "INDEX Feed," by itself, would also work.

SPLIT was used in the ANOVA statement in the specification for the repeated measures design in Box 7. The phrase, "SPLIT INTO 5," told P-STAT to make five rows out of each row of measurements (the t1's, t2's, etc.). The "CARRY Medicine Patient" was an instruction to keep the corresponding values for medicine and patient identification with each measurement. The phrase, "INDEX Day 5," told P-STAT to make a new variable, "Day" that went from 1 to 5, and the phrase, "CREATE Temp. Day.1 to Day.5," said to make another new variable, "Temp," that had the measurements presently called "Day.1," "Day.2," and so on. "CREATE Temp. Day.? ", meaning all those variables whose names presently begin with "Day. ", would also work because the index value is assumed to be the number of rows into which the line was split.

[3] The missing value under W8 was deleted in the list in Figure 6.1 for simplicity. It would appear, however, in the input to ANOVA above. It does not matter whether it is deleted or not. In either case, the ANOVA program analyzes unbalanced data. It performs an exact least squares analysis, detecting any confounding and reporting it in the appropriate error strata.

If actual numbers were used in the illustration for the repeated measures design, instead of the t1’s, t2’s, etc., and if SPLIT were used while the data were input to ANOVA (or LIST, as is the case here), the data would be arranged correctly and would appear as they do in Figure 6.10. The variables in the rearranged file appear in the order that they were named.

There may be multiple indexing, as in “INDEX Batch 5 Treatment 2.” However, the product of the index numbers must equal the number of measurements in the original row. Here, that would have to be ten. The measurements could be rows of ten numbers each, measuring the moisture content of a chemical. The measurements in the rearranged file, using double indexing, would appear as they do in Figure 6.11.

Figure 6.11 Use of SPLIT with Double Indexing

```
LIST  Chemical
       [ SPLIT INTO 10,  INDEX  Batch 5 Treatment 2,
         CREATE  Moisture.Content  Moisture?   ]  $
```

```
 Batch   Treatment   Moisture.Content


     1          1                .02
     1          2                .04
     2          1                .01
     2          2                .07
     3          1                .02
     3          2                .06
     4          1                .03
     4          2                .09
     5          1                .02
     5          2                .07

     1          1                .03
     1          2                .09
     2          1                .02

     .          .                  .
     .          .                  .
     .          .                  .

     5          2                .06
```

Other instructions in the P-STAT programming language in addition to SPLIT, such as SET, GENERATE, IF, DO, and so on, may also be useful in modification and selection of variables for an analysis of variance. SPLIT may be used with many other commands besides ANOVA, such as LIST, CORRELATE and MODIFY when a rearrangement of a data file is desired. The inverse of SPLIT, called COLLECT, is also part of the programming language.

6.18 Terminology

A brief summary of some of the terminology used in discussion of the ANOVA program may be helpful at this point. Everything entered after the “Enter subcommand: ” prompt and through the semicolon (;) is a *subcommand.* An ANOVA command and one subcommand are:

```
            Enter a command:
            ANOVA   Medical ;

            Enter subcommand:
            MODEL  Temperature  =  Dosage | Doctor ;
```

Everything separated by commas (,) is a *phrase.* Subcommands are often composed of several phrases. Two phrases in a subcommand are:

```
       DEP  Temperature,  TR  Dosage | Doctor ;
```

The first phrase is a dependent variable phrase; the second phrase is a design phrase. The previously shown MODEL subcommand is a short way to write a dependent variable phrase and a design phrase.

Everything in a design phrase that is after the equal sign (=) is a *design statement.* It is an algebraic expression. An example of a design statement is as follows:

```
       Dosage | Doctor  +  Hospital
```

Everything in a design statement before and/or after a plus sign (+) is a *term.* A design statement with two terms is:

```
       Hospital / Doctor  +  Dosage
```

Some terms are themselves effects and some terms expand to a sum of effects.

Single factors — Dosage, an interaction of several factors — Dosage*Doctor, or a nesting of a factor within another factor — Doctor(Hospital), are *effects.* Effects are the terms in a fully expanded design statement. Each effect has a line on the ANOVA table. An example of a term and its expansion into a sum of effects is:

```
       Hospital / Doctor
       Hospital  +  Doctor(Hospital)
```

Integer-valued variables that describe a classification or nominal-valued variables that describe a treatment are *factors.* Two factors and their values and value labels are:

```
       DOSAGE   (1) .02 ml.  (2) .04 ml.  (3) .06 ml.  /
       DOCTOR   (1) Wiley    (2) Smith    (3) Brown    /
```

The relationship of factors in an experiment is a *design.* The design has TREATMENT, and sometimes ERROR.TERM and BLOCK, design components. An example of a design with three design components is:

```
       MODEL  Temperature  =  Doctor | Dosage,
         ER   Doctor*Dosage,  BLOCK  Doctor ;
```

The three design components are:

```
       TREATMENT     Dosage
       ERROR.TERM    Doctor*Dosage
       BLOCK         Doctor
```

6.19 USING ANOVA INTERACTIVELY

Immediate interactive response to user specifications is one of the major advantages of P-STAT’s ANOVA. The user can repeatedly modify his design and see the resultant analyses. When one component of the design specification is changed, the remaining components are remembered by the program and used to construct and display a new analysis table.

The initial design specification may be in either of two equivalent forms:

```
MODEL  Sys.Blood    =  Drug | Doctor ;  or
DEP    Sys.Blood,  TR  Drug | Doctor ;
```

When the semicolon (;) is received, the end of the subcommand is signalled, and the analysis of variance table is immediately calculated and displayed. The general form of the ANOVA table for the preceding design is shown in Figure 6.12.

Figure 6.12 Factorial Experiment with Default Error Term

```
MODEL  Sys.Blood = Drug | Doctor ;

           - - - - -

ANALYSIS OF VARIANCE:  Sys.Blood

    SOURCE OF VARIATION

       Drug
       Doctor
       Drug*Doctor
       Residual
```

The ANOVA command remembers the design specification. Means of all treatment effects may be requested by typing only:

```
MEANS ;
```

or means of just particular treatment or error term effects may be requested:

```
MEANS  Drug ;
```

The requested tables of means are immediately calculated and displayed.

Various modifications of the current design specification may be made. The subcommand:

```
SHOW  DESIGN ;
```

will show the current design, without modification and without performing an analysis. An error term, other than the default residual mean square, may be specified:

```
ER  Drug*Doctor ;
```

This design phrase repositions the Drug*Doctor effect from the treatment component of the design to the error term component. The resultant treatment component is now Drug + Doctor. The ANOVA table for this design is then calculated and displayed. The general form of this table is shown in Figure 6.13.

Error terms, not originally in the MODEL statement, may also be specified:

```
ER  Hospital ;  or
ER  Drug*Doctor  +  Hospital ;
```

If either of the preceding error term specifications were made instead of the prior one, a new term, Hospital, would be added to the design. With the first design phrase, Hospital would be the error term. With the second design phrase, both Drug*Doctor and Hospital would be error terms.

Figure 6.13 Factorial Experiment with Interaction Error Term

```
MODEL  Sys.Blood = Drug | Doctor, ER  Drug*Doctor ;

                 - - - - -

      ANALYSIS OF VARIANCE:  Sys.Blood

            SOURCE OF VARIATION

              Drug*Doctor
                Drug
                Doctor
                Residual

              Within Drug*Doctor
                Residual
```

Returning to the design where Drug*Doctor was specified as the error term (Figure 6.13), we can respecify this design without the explicit error term by entering:

```
TR  @  +  Drug*Doctor ;
```

The Drug*Doctor effect is then repositioned from the error term component of the design specification to the treatment component. The at-sign (@) designates the previously specified treatment effects. The at-sign (@) may be used either to repeat a prior design (as in TR. @ ;) or to add effects to the prior design (as in the above usage). (It may also be used with ER and BLOCK to designate those previously specified effects.) The ANOVA table is again immediately calculated and displayed. This specification yields the same table as that in Figure 6.12.

To add a blocking factor, enter:

```
BL  Doctor ;
```

The doctor effect is repositioned from the treatment component of the design to the block component. The use of:

```
TR  Doctor  +  @ ,  SHOW  DESIGN ;
```

will reposition the Doctor effect into the treatment component of the design, effectively cancelling the block, and will display this new design. The table will not be calculated at this time, however, because of the use of SHOW DESIGN. Note that the ORDER of the treatment effects is now different — Doctor precedes Drug.[4]

[4] The order of calculation of the mean square (variance) attributable to an effect often changes that mean square. For example, changing the order of calculation of the mean squares attributable to main effects results in different calculated values when there are unbalanced or missing data. With balanced data, no changes occur. Changing the order of calculation of effects such that the interaction mean square is calculated before that of the main effects is to be avoided, because it will result in the use of all degrees of freedom for the interaction effect, with none remaining for the main effects.

To calculate and display the ANOVA table, enter:

```
TR  @ ;
```

The analysis for the current design specification (the one just displayed) will now be calculated and the ANOVA table will be shown.

The following:

```
Y  Temperature ;
```

will use the last specified design, but with a different dependent variable.

```
Y  ( Temperature,  Pulse,  Hemoglobin ) ;
```

will repeat the last design three times with three different dependent variables. The subcommand:

```
SHOW  VARIABLES ;
```

will show the variables in the file currently input to ANOVA. This is particularly helpful when a range of dependent variables is to be indicated:

```
DEP  ( Temperature  TO  Hemoglobin ) ;
```

and a refresher as to what variables are included in that range is necessary. SHOW VARIABLES may be abbreviated to SHOW", but SHOW DESIGN must be abbreviated to SHOW D.

6.20 Deleting Effects

Assume a current design (Figure 6.12) with the treatment component:

```
Drug | Doctor
```

which expands to:

```
Drug  +  Doctor  +  Drug*Doctor
```

An effect or several effects, such as the Drug*Doctor interaction, may be deleted from this current design specification in three different ways:

1. Respecification without an effect:

   ```
   TR  Drug  +  Doctor ;
   ```

2. Subtraction of an effect:

   ```
   TR  @  -  Drug*Doctor ;
   ```

3. Use of NO:

   ```
   ER  Drug*Doctor ;
   NO  ER ;
   ```

Respecification of any design phrase, with some previously specified effects left out, will result in the deletion of those effects from the design. In the first example above, the respecification of TREATMENT without including the Drug*Doctor interaction will result in the deletion of that interaction. Drug and Doctor, in that order, will be tested against the new residual mean square.

Subtraction of an effect or several effects will have a similar result. In the second example above, the Drug*Doctor interaction is subtracted from the current TREATMENT specification leaving Drug and Doctor, in the order they were in the last specification, to be tested against the residual mean square.

The use of NO, with either ERROR.TERM or BLOCK, will remove that complete design component and all effects in that component. In the third preceding example, an ERROR.TERM of the Drug*Doctor interaction was specified. The resultant table was calculated and displayed. Then, NO ER (an abbreviation for NO ERROR.TERM) was used to cancel the error term component, deleting the Drug*Doctor interaction from the design.

Drug and Doctor, in the order that they were in the last specification, will now be tested against the residual mean square.

6.21 Resetting the Design Specification

The reuse of the MODEL statement resets the design specification process. For example:

```
MODEL  (Temperature,  Pulse)  =
  Doctor  +  Drug  +  Hospital ;
```

would begin the specification process anew, with dependent variables of Temperature and Pulse and treatment effects of Doctor, Drug and Hospital. The error term would be the residual mean square.

Figure 6.14 Design Specification: Positioning Effects

Either

```
MODEL  Sys.Blood = Drug|Doctor, ER  Drug*Doctor,  BL  Doctor ;
```

or

```
DEP    Sys.Blood,  TR  Drug,    ER  Drug*Doctor,  BL  Doctor ;
```

creates two groups:

```
 --------------------         --------------------
|                    |       |                    |
| Dependent Variable |       |       Effect       |
|        List        |       |        Pool        |
|                    |       |                    |
 --------------------         --------------------
```

and positions the dependent variables and the design components:

```
DEPENDENT              TREATMENT          ERROR TERM           BLOCK
                   |
Sys.Blood          |      Drug            Drug*Doctor          Doctor
                   |
```

6.22 Summary of Interactive Behavior

The interactive behavior of the ANOVA phrases can be summarized as follows:

1. TWO GROUPS — The MODEL phrase creates two groups — one is a list of dependent variables and the other is a “pool” of effects (the items in the design statement to the right of the equal sign). The simultaneous use of DEPENDENT and TREATMENT, and optionally ERROR.TERM and/or BLOCK phrases, also creates these two groups. See Figure 6.14.

2. TREATMENT — All effects in the effect pool are positioned in the treatment component of the design, unless there are ERROR.TERM and/or BLOCK phrases, which position some of the effects elsewhere.

3. ERROR AND BLOCK — Specified ERROR.TERM and/or BLOCK phrases position those designated effects in the error term and/or the block design components, unless there are any redundancies. (Redundancies — effects specified in more than one position in the design — are positioned according to a hierarchy. This is explained in the following section: "Specifying Complex Designs Interactively — Redundant Effects".)

4. RESIDUAL VARIANCE (MEAN SQUARE) — When an error term is not explicitly specified, the residual variance (more technically, the residual mean square) is used as the error term. Often the residual variance is the within-cells variance, but it may also include other potentially identifiable effects (such as an interaction).

5. MODIFICATION — New design phrases modify the current design by repositioning effects, specifying additional effects, or by deleting effects. New dependent variable phrases use the current design, but calculate the ANOVA table with the newly specified dependent variables.

6. RESET — The use of MODEL resets the dependent variable list and the effect pool to the newly specified variables and effects. Previously specified TR, ER and BL components are removed.

6.23 SPECIFYING COMPLEX DESIGNS INTERACTIVELY

Design specifications, complex and simple, may be in either of two equivalent forms:

```
MODEL  SAT.Score   =   (Teacher/Class) | Method,
  ER   Class(Teacher)  +  Method*Class(Teacher) ;    OR

DEP    SAT.Score,  TR  Teacher | Method,
  ER   Class(Teacher)  +  Method*Class(Teacher) ;
```

This experiment might be one involving two teachers, each of whom taught three randomly assigned SAT preparation classes. The teachers randomly divided the students in each class into two groups. Both groups in each class received the same instruction in vocabulary and test-taking procedures for the first part of each lesson. However, each group used a different reinforcement method, either a programmed text or a traditional one, for independent work during the second part of each lesson. The score on an old SAT test at the end of the term might be the response measure (dependent variable).

The MODEL design statement is a complete description of the dependent variable and of all of the effects and their relationships. Class, nested under Teacher, is crossed with Method.

The design statement (right side) in the MODEL statement above expands into this sum of effects:

```
(Teacher/Class)  |  Method   =
(Teacher/Class)  +  Method   +  (Teacher/Class)*Method  =
 Teacher  +  Class(Teacher)  +  Method
          +  Teacher*Method  +  Method*Class(Teacher) ;
```

These effects constitute the effect pool. The preceding ER phrase states that the Class within Teacher and Method*Class interaction within Teacher effects are to be positioned in the error term component of this design. The remaining effects are positioned in the treatment component. (An optional TR phrase could follow the MODEL phrase and explicitly state the treatment effects.)

The ANOVA program identifies the error term appropriate for testing each treatment effect. It calculates the F ratios and places them in the correct strata in the ANOVA table whose general form is shown in Figure 6.15.

Figure 6.15 Factorial Design with Explicit Nesting

```
MODEL  SAT.Score = (Teacher/Class) | Method,
  ER   Class(Teacher) + Method*Class(Teacher) ;

                 - - - - -

       ANALYSIS OF VARIANCE:  SAT.Score

          SOURCE OF VARIATION

           Class(Teacher)
             Teacher
             Residual

           Method*Class(Teacher)
             Method
             Teacher*Method
             Residual

           Within Method*Class(Teacher)
             Residual
```

The alternate form for specifying a design to ANOVA states the effects to be positioned in each component of the design:

```
DEP   Sat.Score,  TR  Teacher | Method,
  ER  Class(Teacher)  +  Method*Class(Teacher) ;
```

Any component of the design that needs expansion, such as the TR phrase, is expanded into a sum of effects:

```
TR  Teacher  |  Method
```

is expanded to:

```
TR  Teacher  +  Method  +  Teacher*Method ;
```

This alternate specification yields the same sum of effects as did the specification using MODEL and an identical ANOVA table.

6.24 Redundant Effects

Any redundantly specified effects, that is, effects that appear in more than one design component, are resolved according to the following hierarchy:

```
TREATMENT      (Highest)
ERROR.TERM
BLOCK
MODEL          (Lowest)
```

All effects in the MODEL statement are positioned in the treatment design component unless there are ERROR.TERM and/or BLOCK design phrases which position some of these effects elsewhere. An effect in BLOCK , and also in either ERROR.TERM or TREATMENT components, is dropped from the BLOCK. An effect in either BLOCK or ER, and also in the TREATMENT component, is dropped from the ERROR.TERM or BLOCK.

When relatively complex designs are used, it may be simpler to specify the TREATMENT, ERROR.TERM and BLOCK components separately, than to describe their overall relationship in a MODEL statement. However, either method of design specification will yield identical analyses.

6.25 Implicit Nesting

The ANOVA program will discern a nesting relationship by the index values that the factor variables have. For example, the previous experimental design could have been specified as follows:

```
DEP   SAT.Score,  TR  Teacher | Method,
  ER  Method | Class  ;
```

The nesting relationship of class within teacher would have to be indicated by the indexing of class — its values would run from 1 to 6, and only classes 1 to 3 would have teacher 1 and only classes 4 to 6 would have teacher 2. The general ANOVA table for this design specification would look like that in Figure 6.16. (Note that after expansion of the treatment and error term components of the design specification, Method dropped out of the error term in accordance with the hierarchy previously described.)

Figure 6.16 Factorial Design with Implicit Nesting

```
DEP  SAT.Score, TR  Teacher | Method, ER  Method | Class ;

                  - - - - -

       ANALYSIS OF VARIANCE:  SAT.Score
            SOURCE OF VARIATION

              Class
                Teacher
                Residual

              Method*Class
                Method
                Teacher*Method
                Residual

              Within Method*Class
                Residual
```

Observe the similarity of the analysis table in Figure 6.16 with the one in Figure 6.15. The tests calculated for both tables are the same. Only the effect names differ slightly. The nesting within teacher is not explicit in the names in the current table, although the nesting relationship is clear to the ANOVA program because of the factor indexing. This design specification, with implicit nesting due to the factor indexing, is marginally more efficient for small designs, but may become much more efficient (or even make analyses possible) for larger designs. Computer storage space and time are saved, with a concomitant saving in expense.[5]

Consider an experimental design similar to the prior educational one, but involving split-plots:

[5] Computer storage used is proportional to the square of the number of degrees of freedom explicitly specified in the design statements. Computer time for the analysis itself (as distinct from the interpretation of the subcommands and the formatting of the output tables) is proportional to the cube of the number of degrees of freedom.

```
       MODEL  Yield  =  (Fertilizer/Plot) | Insecticide,
         ER   Plot(Fertilizer)  +  Insecticide*Plot(Fertilizer) ;
```

In this design, three Plots are treated with Fertilizer 1 and three other Plots are treated with Fertilizer 2 — that is, Plots are nested within Fertilizer. Each Plot is divided into two subplots, and the subplots are treated with two different insecticides.

This design is essentially the same as the educational one — Fertilizer parallels Teacher, Plot parallels Class, and Insecticide parallels Method. However, the number of response measures differs. In the educational experiment, there were many student SAT scores per experimental group. In the agricultural experiment, there is one yield measurement per subplot. Therefore, there is no within-cells variance to use as the denominator to test the Insecticide*Plot(Fertilizer) effect.

Figure 6.17 **Split-Plot Design with Implicit Subplots**

```
     MODEL  Yield = Fertilizer | Insecticide, ER  Plot ;

                          - - - - -

                ANALYSIS OF VARIANCE:  Yield

                    SOURCE OF VARIATION

                      Plot
                        Fertilizer
                        Residual

                      Within Plot
                        Insecticide
                        Fertilizer*Insecticide
                        Residual
```

Therefore, this design could be specified more efficiently as follows:

```
       MODEL  Yield   =    Fertilizer | Insecticide,  ER  Plot ;   or
       DEP    Yield,  TR   Fertilizer | Insecticide,  ER  Plot ;
```

This design specification calculates the same F tests, but is more efficient both in time and in expense, because the triple-indexed effect, Insecticide*Plot(Fertilizer), is not explicitly calculated. Instead, its sum of squares is allowed to default to the .WITHIN. Plot residual. The general form of the ANOVA table that would be calculated is shown in Figure 6.17.

6.26 MULTIPLE COMPARISONS: PRE-PLANNED

The ANOVA command can be used with multiple pre-planned comparisons (also called orthogonal contrasts) among different group means. The necessary steps include generating the contrast variables and specifying a design which uses the double slash (//) operator. The double slash operator requests that the sum of squares of the expression preceding the slashes be partitioned using the expression following the slashes.

6.27 One-Way analysis With Contrasts

In the following data set[1], a one way anova is used to compare the mean tensile strength of cotton at five different cotton percentage levels. Pre-planned comparisons of interest are:

1. mean of the fourth group versus the mean of the fifth group
2. mean of groups one and three versus mean of groups four and five
3. mean of group one versus mean of group three
4. mean of group two versus all the rest.

Figure 6.18 Data Set for Pre-Planned Comparisons

Cotton	obs	y
1	1	-8
1	2	-8
1	3	0
1	4	-4
1	5	-6
2	1	-3
2	2	2
2	3	-3
2	4	3
2	5	3
3	1	-1
3	2	3
3	3	3
3	4	4
3	5	4
4	1	4
4	2	10
4	3	7
4	4	4
4	5	8
5	1	-8
5	2	-5
5	3	-4
5	4	0
5	5	-4

Continued in the next column

The four contrast variables must either exist in the data set or they can be GENerated using the PPL as illustrated in Figure 6.19. Four new variables are generated, one for each of the contrasts of interest. The first three contrasts have three values each. Zero is used for the means that are not of interest. One and minus one are used for the means of interest. The fourth contrast, group two means against all the rest, needs only two values: minus four for group two and one for the remainder.

Any contrast that has more than two values must be specified as a covariate to obtain a single degree of freedom for that contrast.

```
COVARIATE (Mean.4.vs.5,Mean.13.vs.45,Mean.1.vs.3),
```

[1] This data set is from *Design and Analysis of Experiments* by Douglas C Montgomery, Page 54.

Figure 6.19 One-way Analysis with Contrasts

```
ANOVA Montgom

    /*   GENERATE THE CONTRASTS   */

    [ GEN Mean.4.vs.5 = 0;
      GEN Mean.13.vs.45 = 1;
      GEN Mean.1.vs.3 = 0;
      GEN Mean.2.vs.1345 = 1 ]

    [ IF Cotton = 4, SET Mean.4.vs.5 =  1 ;
      IF Cotton = 5, SET Mean.4.vs.5 = -1 ;

      IF Cotton = 2, SET Mean.13.vs.45 =  0 ;
      IF Cotton = 4, SET Mean.13.vs.45 = -1 ;
      IF Cotton = 5, SET Mean.13.vs.45 = -1 ;

      IF Cotton = 1, SET Mean.1.vs.3 =  1 ;
      IF Cotton = 3, SET Mean.1.vs.3 = -1 ;

      IF Cotton = 2, SET Mean.2.vs.1345 = -4  ]

    /*   SUPPLY COVARIATES AND THE DESIGN AS SUBCOMMANDS */

LABELS Mont.Lab ;

    COVARIATE ( Mean.4.vs.5,Mean.13.vs.45,Mean.1.vs.3 ),

    MODEL y = Cotton  // ( Mean.4.vs.5  +  Mean.13.vs.45  +
                           Mean.1.vs.3  +  Mean.2.vs.1345 );

    MEANS$
```

Because Mean.2.vs.1345 has only 2 levels (1 and -4) it need not be specified as a covariate.

The model statement uses the double slash (//) operator to specify partitioning of the Cotton sum of squares into the four orthogonal contrasts.

```
  MODEL y = Cotton  // ( Mean.4.vs.5  +  Mean.13.vs.45 +
                         Mean.1.vs.3  +  Mean.2.vs.1345 );
```

The output in Figure 6.20 shows that contrast one and contrast three are highly significant. Contrast two has a p value of .063 and contrast four shows that mean two does not appear to differ from the average of the other four groups.

In other words, mean four appears to differ from mean five and mean three appears to differ from mean one. It is less clear that the average of three and one differs from the average of four and five.

Figure 6.20 ANOVA Output with Contrasts

```
--------------------  P-STAT ANOVA  Version  2.20  -----------------
ANALYSIS OF VARIANCE:   y
            in File:   Montgom

SOURCE OF VARIATION       DF        SS         MS       F    Pr > F
  Cotton                   4      475.76     118.94   14.76  0.0001
    Mean.4.vs.5            1      291.60     291.60   36.18  0.0001
    Mean.13.vs.45          1       31.25      31.25    3.88  0.0630
    Mean.1.vs.3            1      152.10     152.10   18.87  0.0003
    Mean.2.vs.1345         1        0.81       0.81    0.10  0.7545
  Residual                20      161.20       8.06

Adjusted Total            24      636.96


GRAND MEAN                              0.04
Number of Observations                 25



    Cell Contents are....
       Cell Counts
    ---Mean Score Of Variable  -------y--------

                                                                Row
                   15       20       25       30       35      Totals
               --------------------------------------------
 Cotton       |     5        5        5        5        5  |      25
              | -5.20     0.40     2.60     6.60    -4.20  |   0.04
               --------------------------------------------
```

6.28 Two-Way Analysis with Contrasts

The following data set[2] is used to illustrate a somewhat more complex problem. This is a two-way analysis of variance to see how protein source and protein level affect weight. In this case the contrasts of interest are beef protein versus pork protein, and animal protein versus vegetable protein.

In the listing of the file in Figure 6.22, the variable Level.Protein is a numeric variable which is coded 1 and 2. The values High and Low are printed because a labels file was used in the LIST command. Source.Protein is also a numeric variable. Cereal has a code of 2. Beef has a code of 1, and pork has a code of 3. The labels make the listing more meaningful but they are not the values found in the file. The PPL in the ANOVA command (Figure 6.22), which is used to create the contrasts, references the actual numeric values.

[2] This data set is taken from *Statistical, Methods* by George Snedecorf and William Cochran, 1980, page 305.

Figure 6.21 **Data Set for Two-Way Analysis with Contrasts**

	Level	Source
Weight	Protein	Protein
73	High	Beef
102	High	Beef
118	High	Beef
104	High	Beef
81	High	Beef
107	High	Beef
100	High	Beef
87	High	Beef
117	High	Beef
111	High	Beef
98	High	Cereal
74	High	Cereal
56	High	Cereal
111	High	Cereal
95	High	Cereal
88	High	Cereal
82	High	Cereal
77	High	Cereal
86	High	Cereal
92	High	Cereal
94	High	Pork
79	High	Pork
96	High	Pork
98	High	Pork
102	High	Pork
102	High	Pork
108	High	Pork
91	High	Pork
120	High	Pork
105	High	Pork
90	Low	Beef
76	Low	Beef
90	Low	Beef
64	Low	Beef
86	Low	Beef
51	Low	Beef
72	Low	Beef
90	Low	Beef
95	Low	Beef
78	Low	Beef
107	Low	Cereal
95	Low	Cereal
97	Low	Cereal
80	Low	Cereal
98	Low	Cereal
74	Low	Cereal
74	Low	Cereal
67	Low	Cereal
89	Low	Cereal
58	Low	Cereal
49	Low	Pork
82	Low	Pork

Figure 6.22 **Two-Way Analysis with Orthogonal Contrasts**

```
ANOVA SC305
     [ GEN Animal.v.Veg  = 1 ;
       GEN Beef.v.Pork = 0 ;

       IF Source.Protein = 2, SET Animal.v.Veg = -1 ;

       IF Source.Protein = 1, SET Beef.v.Pork  = 1 ;
       IF Source.Protein = 3, SET Beef.v.Pork = -1 ]
  COVARIATE = ( Beef.v.Pork ),

  MODEL  Weight =  Source.Protein  //
     (Animal.v.Veg + Beef.v.Pork) | Level.Protein ;
  MEANS $
```

```
--------------------  P-STAT ANOVA  Version  2.20  -----------------

ANALYSIS OF VARIANCE:   Weight
             in File:   SC305

SOURCE OF VARIATION       DF        SS         MS       F    Pr > F

  Source.Protein           2      266.53     133.27    0.62  0.5411
    Animal.v.Veg           1      264.03     264.03    1.23  0.2722
    Beef.v.Pork            1        2.50       2.50    0.01  0.9144
  Level.Protein            1     3168.27    3168.27   14.77  0.0003
  Source.Pr*Level.Prot     2     1178.13     589.07    2.75  0.0732
    Animal.v.*Level.Prot   1     1178.13    1178.13    5.49  0.0228
    Source.Pr*Level.Prot   1        0.00       0.00    0.00  1.0000
  Residual                54    11586.00     214.56

  Adjusted Total            59    16198.93

 GRAND MEAN                          87.87
 Number of Observations              60



   Cell Contents are....
      Cell Counts
   ---Mean Score Of Variable  -----Weight-----

               Level.Protein

Source                              Row
Protein          High     Low       Totals
              -----------------
Beef         |    10       10  |     20
             |100.00    79.20  |  89.60
             |                 |
Cereal       |    10       10  |     20
             | 85.90    83.90  |  84.90
             |                 |
Pork         |    10       10  |     20
             | 99.50    78.70  |  89.10
              -----------------

Total N           30       30        60
Mean           95.13    80.60     87.87
```

```
     Cell Contents are....
        Cell Counts
     ---Mean Score Of Variable  -----Weight-----

                 Level.Protein

  Animal.v.                          Row
  Veg              High     Low      Totals
                 -----------------
  Beef and     |    20       20  |     40
  Pork         | 99.75    78.95  |  89.35
               |                 |
  Cereal       |    10       10  |     20
               | 85.90    83.90  |  84.90
                 -----------------


  Total N           30       30        60
  Mean           95.13    80.60     87.87
```

Figure 6.22 contains the ANOVA command with the PPL to generate the contrasts and the necessary COVARIATE and MODEL statements to do the analysis. The contrast between animal and vegetable protein has only two levels; one for the combination of pork protein and beef protein; and the second for vegetable protein. The contrast between beef protein and pork protein has three levels; one for beef, the second for pork, and a third (the zeros) for the remaining cases which are not part of this contrast.

Because the beef versus pork contrast has more than two levels, it must be included in the COVARIATE statement so that the degrees of freedom are correctly allocated.

```
      COVARIATE = ( Beef.v.Pork ),
```

The model specifies a two way interaction with one factor, Source.Protein, decomposed into two orthogonal comparisons. The double slash (//) indicates that the expression which follows contains the partitions to be used when allocating the sum of squares in Source.Protein. The vertical bar indicates that the results of this partitioning are to be crossed with Level.Protein.

```
      MODEL  Weight =  Source.Protein  //
            (Animal.v.Veg + Beef.v.Pork) | Level.Protein ;
```

The results do not show any apparent difference between animal and vegetable protein or between beef and pork protein overall. However, the interaction contrast of animal versus vegetable with level of protein is signifigant at the 5% level.

6.29 Nested Design with Contrasts

The data set[3] in Figure 6.23 is used to illustrate an even more complex analysis with partitioning in a nested design.

[3] This data set is taken from *Experimental Designs*, by W.G. Cochran and G.M. Cox, chapter 3 and is also analyzed by Richard Heiberger in *Computation for the Analysis of Designed Experiments*, page 632.

Figure 6.23 **Contrasts in a Nested Design**

```
D1    F1    D2    F2    D3    F3    D4    F4

 0     0     2     1     1     3     1     2
 1     4     0     0     0     0     2     2
 2     4     1     1     0     0     2     3
 1     1     0     0     1     4     2     1
 0     0     2     3     2     4     1     3
 2     2     0     0     1     2     0     0


D5    F5    D6    F6    D7    F7    D8    F8

 2     2     2     4     2     1     0     0
 1     1     1     3     1     2     0     0
 0     0     0     0     2     3     1     4
 2     1     0     0     1     1     1     2
 0     0     2     3     2     4     0     0
 1     4     1     3     0     0     2     2

   C1    C2    C3    C4    C5    C6    C7    C8

  269   283   252   212    95   127    80   134
  138   100   197   263   107    89    41    74
  282   230   216   145    88    25    42    62
  124   211   194   222   193   209   109   153
  102   193   128    42    29     9    17    19
  162   191   107    67    23    19    44    48


   S1    S2    S3    S4    S5    S6    S7    S8

  466   280   398   386   199   166   142   590
  194   219   421   379   236   332   176   137
  372   256   708   304   356   212   308   221
  268   505   433   408   292   352   132   454
  363   561   311   222   254    92    28   106
  365   563   415   338    80   114   268   298
```

One part of the analysis deals with the effectiveness of different fumigants in minimizing the number of eelworms in the soil. Two questions of interest are: 1) are there differences in the effectiveness of different fumigants? and 2) is the average response to fumigation linear in the amount of fumigation?

The answer to the first question involves the nesting of fumigants within dose. A set of contrasts is created for each dosage level (Fum.D1 and Fum.d2). To answer the second question of a linear response, dose is partitioned into a linear and a quadratic contrast. Most of the work is done in the MODIFY command which uses SPLIT to rearrange the data and create index variables.

Figure 6.24 Nested Design with Contrasts

```
MODIFY CC46.Raw
      [ GEN Block.row TO RECODE(.N., 1/3=0, 4/6=2);
        SPLIT INTO 8,

         DEFINE Dose         = D?,
         DEFINE Fumigant     = F?,
         DEFINE First.Count  = C?,
         DEFINE Second.Count = S?,

         CARRY Block.row,
         INDEX Block.column 2 Col 4 ;

         GEN Block TO Block.row + Block.column ;
         IF Dose = 1, GEN  Fum.D1 TO Fumigant, F.SET Fum.D1 TO 0 ;
         IF DOSE = 2, GEN  Fum.D2 TO Fumigant, F.SET Fum.D2 TO 0 ;

         GEN Ctl.vs.Treat TO RECODE( Fumigant, 0=0, X=1 );

         GEN Dose.Lin.Avg TO Dose -1 ;

         GEN Dose.Quad    TO RECODE( Dose, 1=2, x=-1 ) ],
  OUT CC46 $

 ANOVA  CC46;
    DEPENDENT = Second.Count,
    BLOCKS = Block,

    TR = Dose // ( D.L.A + D.Q) +
             ( Fumi(Dose))//(Fum.D1 + Fum.D2),

    COVARIATE = ( D.L.A, D.Q ) $
```

The specification of the model in the ANOVA command is relatively simple once the data are properly arranged and the contrasts created. Note the use of abbreviations (D.L.A) for long variable names such as Dose.Lin.Avg. The use of the first double slash (//) partitions Dose into D.L.A and D.Q. The use of the second double slash partitions the nesting of Fumigant within Dose into the two contrasts Fum.D1 and Fum.D2.

```
TR = Dose // ( D.L.A + D.Q) +
         ( Fumi(Dose))//(Fum.D1 + Fum.D2),

COVARIATE = ( D.L.A, D.Q ) $
```

The contrasts D.L.A and D.Q which have three levels must be included as covariates so that the degrees of freedom will be properly allocated. Had the D.L.A not been declared as a covariate, it would have been interpreted as a 3-valued factor, equivalent to the dose factor. Because Fum.D1 and Fum.D2 have only two levels they are not included as covariates.

Figure 6.25 ANOVA Output: Nested Design with Contrasts

```
--------------------  P-STAT ANOVA  Version  2.20  -----------------

         ANALYSIS OF VARIANCE:   Second.Count
                     in File:   CC46

SOURCE OF VARIATION       DF        SS          MS       F    Pr > F

BLOCK                      3   289426.50   96475.50    6.38  0.0014

.WITHIN. BLOCK            44   702138.17   15957.69
  Dose                     2    88346.54   44173.27    2.92  0.0668
    Dose.Lin.Avg           1    57206.53   57206.53    3.78  0.0597
    Dose.Quad              1    31140.01   31140.01    2.06  0.1600
  Dose*Fumigant            6    69101.38   11516.90    0.76  0.6051
    Fum.D1                 3    45460.69   15153.56    1.00  0.4033
    Fum.D2                 3    23640.69    7880.23    0.52  0.6707
  Residual                36   544690.25   15130.29

Adjusted Total            47   991564.67



GRAND MEAN                            305.83
Number of Observations                 48
```

Figure 6.25 contains the ANOVA output for the nested design with contrasts. In this set of data the Fum.D1 and Fum.D2 contrasts are not significant. The observed significance level of the linear effect is .06 .

6.30 ANOVA TESTS

Other identifiers request tests, which can only be done on simple, one-way designs. The post hoc multiple comparison tests are requested by using the following identifiers;

1. BARTLETT, a homogeneity of variance test.
2. LSD, the Least Significant Difference method of treatment means comparison.
3. MOD.LSD, a modified LSD which can also be called a BONFERRONI comparison.
4. SCHEFFE, a test for comparing all possible contrasts between treatment means.
5. DUNCAN, a multiple range test for comparing all pairs of means.
6. SNK, Student-Neuman-Keuls is similar to the Duncan test but not as powerful. It uses studentized range values.
7. TUKEY.A, a test also based on studentized range statistic. It uses the largest step value for all of the pairwise tests
8. TUKEY.B, uses the average of the SNK and the TUKEY.A values. Otherwise it is like TUKEY.A..

9. DUNNETT, a test comparing the treatment means with the mean of a control group. DUNNETT 11, defines the group with a value of 11 on the factor variable as the control group. If just DUNNETT, is used, the control group is assumed to be the group with the LOWEST value on the factor variable.

Figure 6.26 Data Set for Tests

```
LIST Mont53 [ CASES 1-8 ] $


     f     v

    15     7
    15     7
    15    15
    15    11
    15     9
    20    12
    20    17
    20    12


LIST Mont53 [ COLLECT 5, BY f ] $


     f   v.1   v.2   v.3   v.4   v.5

    15     7     7    15    11     9
    20    12    17    12    18    18
    25    14    18    18    19    19
    30    19    25    22    19    23
    35     7    10    11    15    11
```

Figure 6.26 contains the data set[4] which is used in Figure 6.27 to illustrate the output produced by each of the available tests. All the tests except Bartlett and Dunnett produce a table which looks like:

```
1: sig at the 1 percent level,  5: sig at 5 percent

mean              n group  30 25 20 35 15

21.60000          5    30      5  1  1  1
17.60000          5    25   5     .  1  1
15.40000          5    20   1  .     5  1
10.80000          5    35   1  1  5     .
9.800000          5    15   1  1  1  .
```

[4] This data set is from *Design and Analysis of Experiments* by Douglas C Montgomery, Page 53.

A dot (.) indicates that the two groups were not found to be significantly different. A 1 is used if the difference is significant at the 1 percent level. A 5 is used if the difference is significant at the 5 percent level. Thus in the example above all the groups were found to have some significant difference except 20-25 and 15-35.

Figure 6.27 Output From the Tests

```
  The command:

  ANOVA Mont53, TEST NAME HERE;
   MODEL v=f;


          ANALYSIS OF VARIANCE:   v
                       in File:   Mont53

 SOURCE OF VARIATION         DF        SS            MS          F   Pr > F

   f                         4      475.7600      118.9400   14.757   .0001
   Residual                  20      161.2000      8.060000

 Adjusted Total              24      636.9600


 GRAND MEAN                          15.04000
 Number of Observations                    25


---------------------------------------------------------------------------

 LSD (t-tests)
 1: sig at the 1 percent level,  5: sig at 5 percent

 mean              n group  30 25 20 35 15

 21.60000          5    30      5  1  1  1
 17.60000          5    25   5     .  1  1
 15.40000          5    20   1  .     5  1
 10.80000          5    35   1  1  5     .
 9.800000          5    15   1  1  1  .


---------------------------------------------------------------------------

 Mod LSD (Bonferroni)
 1: sig at the 1 percent level,  5: sig at 5 percent

 mean              n group  30 25 20 35 15

 21.60000          5    30      .  5  1  1
 17.60000          5    25   .     .  5  1
 15.40000          5    20   5  .     .  .
 10.80000          5    35   1  5  .     .
 9.800000          5    15   1  1  .  .
```

```
Scheffe
1: sig at the 1 percent level,  5: sig at 5 percent

mean              n group  30 25 20 35 15

21.60000          5    30      .  .  1  1
17.60000          5    25   .     .  5  5
15.40000          5    20   .  .     .  .
10.80000          5    35   1  5  .     .
9.800000          5    15   1  5  .  .

---------------------------------------------------------------------------
```

```
Duncan
1: sig at the 1 percent level,  5: sig at 5 percent

mean              n group  30 25 20 35 15

21.60000          5    30      5  1  1  1
17.60000          5    25   5     .  1  1
15.40000          5    20   1  .     5  1
10.80000          5    35   1  1  5     .
9.800000          5    15   1  1  1  .

---------------------------------------------------------------------------
```

```
Tukey:a
1: sig at the 1 percent level,  5: sig at 5 percent

mean              n group  30 25 20 35 15

21.60000          5    30      .  5  1  1
17.60000          5    25   .     .  1  1
15.40000          5    20   5  .     .  5
10.80000          5    35   1  1  .     .
9.800000          5    15   1  1  5  .

---------------------------------------------------------------------------
```

```
Tukey:b
1: sig at the 1 percent level,  5: sig at 5 percent

mean              n group  30 25 20 35 15

21.60000          5    30      .  5  1  1
17.60000          5    25   .     .  1  1
15.40000          5    20   5  .     5  5
10.80000          5    35   1  1  5     .
9.800000          5    15   1  1  5  .
```

```
Test for homogeneity of variances:
  Bartlett F. . . . . =      .101,   Pr>f   =  .9822
  Bartlett chi-square =      .933,   Pr>chi =  .9198
Begin subcommand, or type  Q  or  H:
(1 record  has been read from the editor.)
```

```
---------------------------------------------------------------------------
```

```
Dunnett's test
Control group is 15,   mean=9.800000,   n=5
1: sig at the 1 percent level,  5: sig at 5 percent

        mean   difference      n  group  1-tail  2-tail

    21.60000     11.80000      5     30       1       1
    17.60000     7.800000      5     25       1       1
    15.40000     5.600000      5     20       1       5
    10.80000            1      5     35       .       .
```

REFERENCES

1. Cochran, W.G. and Cox, G.M. (1957), *Experimental Designs*, John Wiley & Sons, New York.
2. Heiberger, Richard (1989), *Computation for the Analysis of Designed Experiments*, John Wiley & Sons, New York.
3. Montgomery, Douglas (1984), *Design and Analysis of Experiments*, John WIley & Sons, New York.
4. Snedecor, G.W. and Cochran, W.G. (1980), *Statistical Methods*, 7th edition, Iowa State University Press, Ames, Iowa, USA.

Appendix A: DESIGN STATEMENTS

A design statement is an algebraic expression composed of terms. The terms are either single effects, or they specify relationships between factors (using operators) that expand into a sum of effects.

Simple Terms:

	A	An effect due to factor A
+	A + B	Addition of effects
*****	A*B	An interaction effect
( )	B(A)	A nested effect: A nested within B

Complex Terms, Operators and Their Expansion:

|	A | B	**Crossing of factors:** A cross B A | B = A + B + A*B

(The ! can be used on terminals without the vertical bar.)

/	A / B	**Nesting of factors:** A nests B A / B = A + B(A)
( )	A | (B / C)	**Grouping of factors:** A cross (B nests C) A | (B / C) = A + (B / C) + A*(B / C) = A + B + C(B) + A*B + A*C(B)
-	A | B | C - A*B*C	**Subtracting effects:** A cross B cross C minus A*B*C A | B | C - A*B*C = A + B + C + A*B + A*C + B*C
-/	A -/ B	**Subtracting nesting:** A minus the nesting of B A -/ B = A - B(A) = A - A*B
	(C | D) -/ D	(C | D) -/ D = (C|D) - C*D = C + D
-|	A -| B	**Subtracting crossing:** A minus the crossing of B A -| B = A - A*B - B
	(C | D) -| D	(C | D) -| D = (C|D) - C*D - D = C
//	A // A.1	**Partitioning sums of squares (SS):** A SS partitioned into A.1
	A | B // (B.1 + B.2)	A cross B, with B SS partitioned into B.1 and B.2

Design Expansion and Redundancies:

1. Expand operators as listed above.

2. Delete any duplicate effects in the expansion of a design statement. For example,

   A + (A | B) = A + A + B + A*B = A + B + A*B.

3. Delete any effects from the BLOCK design component that also appear in the ERROR.TERM or TREATMENT design components.

4. Delete any effects from the ERROR.TERM design component that also appear in the TREATMENT design component.

5. When there is a MODEL phrase, all effects not additionally positioned in the ERROR.TERM or BLOCK design components constitute the TREATMENT design component.

NOTE: The expansion of design statements and the hierarchical deletion, described above, result in an effect or sum of effects. This effect or sum may itself be specified initially as the design statement.

Examples of Design Expansion:

```
1.  ANOVA   Drug ;
      MODEL   Temperature = Medication.Amount ;
```

Expansion:

```
    ANOVA   Drug ;
      MODEL   Temperature = Medication.Amount ;
     (Here, ERROR.TERM will be the Residual Variance.)
```

```
2.  ANOVA   JulyCrop ;
      DEPENDENT   Yield ,
      TREATMENT   Fertilizer | Pesticide ,
      ERROR.TERM  Fertilizer / Field ;
```

Expansion:

```
    ANOVA   JulyCrop ;

      DEPENDENT   Yield ,
      TREATMENT   Fertilizer + Pesticide+ Fertilizer*Pesticide ,
      ERROR.TERM  Fertilizer + Field(Fertilizer) ;
                  =  Field(Fertilizer) ;
```

```
 3.  ANOVA  Reading ;
      DEPENDENT   Score,
      TREATMENT   (Method | Intelligence | Sex) ,
      ERROR.TERM  (Method / Class) | Sex ;
```

Expansion:

```
    ANOVA  Reading ;

      TREATMENT   (Method + Intell + Method*Intell) | Sex ,

                = (Method + Intell + Method*Intell) + Sex
                   + (Method + Intell + Method*Intell)*Sex ,

                = Method + Intell + Method*Intell
                   + Sex + Method*Sex + Intell*Sex
                   + Method*Intell*Sex ,

                = Method + Intell + Sex
                   + Method*Intell + Method*Sex + Intell*Sex
                   + Method*Intell*Sex ,

      ERROR.TERM  (Method / Class) + Sex+ (Method / Class)*Sex ;

                = Method + Class(Method) + Sex
                   + Sex*Method + Sex*Class(Method) ;

                = Class(Method) + Sex*Class(Method) ;
```

Appendix B: SYNONYMS

MODEL	MOD	SHOW DESIGN	SHOW D
DEPENDENT	DEP	SHOW VARIABLES	SHOW
Y	Y	SHOW NAMES	
RESPONSE	RES		
TREATMENTS	TR		
TRIAL.FACTORS	TR		
HYPOTHESIS	HY		
NUMERATORS	NUM		
ERROR.TERMS	ER	NO ERROR.TERM	NO ER
ERRORS	ER		
GROUPING.FACTORS	GR		
DENOMINATORS	DEN		
BLOCKS	BL	NO BLOCK	NO BL
ABSORB	AB		
MEANS	ME		
Y.SUBSET	Y.S		
X.SUBSET	X.S		
XY.SUBSET	XY.S		
COVARIATE	COV		
X			

NOTE: Any unique abbreviation, sufficient to distinguish between the expressions above, will be acceptable to the ANOVA program. The abbreviations given are suggested.

Appendix C: ANOVA EXAMPLE: Input and Output

(See Example 3 on page 6.39 .)

Figure 6.28 Partial Listing of ANOVA Input

```
 >   LIST  Reading  $

          ID   SEX  CLASS  METHOD  INTELL  SCORE

           1     1      1       1       1    100
           2     2      1       1       1    100
           3     1      1       1       1    100
           4     2      1       1       1    100
           5     1      1       1       1    100
           6     2      1       1       1     92
           7     1      1       1       1     91
           8     2      1       1       1     87
           9     1      1       1       1     87
          10     2      1       1       1    100
          11     1      2       1       1    100
          12     2      2       1       1    100
          13     1      2       1       1    100
          14     2      2       1       1    100
          15     1      2       1       1     90
          16     2      2       1       1     72
           .     .      .       .       .      .
           .     .      .       .       .      .
           .     .      .       .       .      .
         178     2     18       2       3     82
         179     1     18       2       3     95
         180     2     18       2       3     81
```

Figure 6.29 ANOVA Table

The command and subcommands

```
ANOVA  Reading ;

DEP  Score,  TR  Method | Intell | Sex ,
 ERROR.TERM  (Method / Class) | Sex ;
```

The output

```
               ANALYSIS OF VARIANCE:   SCORE

                                             Within Stratum Between Strata
SOURCE OF VARIATION   DF      SS        MS      F   Pr > F      F   Pr > F

CLASS(METHOD)         17   15466.91    909.82                 4.46  0.0001
  METHOD               1    2706.69   2706.69   7.02  0.0212
  INTELL               2    3821.74   1910.87   4.96  0.0270
  METHOD*INTELL        2    4312.41   2156.21   5.59  0.0192
  Residual            12    4626.07    385.51

SEX*CLASS(METHOD)     18    3514.20    195.23                 0.96  0.5132
  SEX                  1     347.22    347.22   2.32  0.1536
  METHOD*SEX           1     180.00    180.00   1.20  0.2942
  INTELL*SEX           2     118.01     59.01   0.39  0.6825
  METHOD*INTELL*SEX    2    1073.43    536.72   3.59  0.0601
  Residual            12    1795.53    149.63

Within SEX*CLAS(METH 144   29394.00    204.12
  Residual           144   29394.00    204.12

-- Adjusted Total -- 179   48375.11

GRAND MEAN                           77.78
Number of Observations             180
```

Figure 6.30 ANOVA Table of Means for Method

```
 MEANS  Method ;


CELL CONTENTS ARE....
   CELL COUNTS
---MEAN SCORE OF VARIABLE  -----Score------

                                 ROW
                1        2      TOTALS
           -----------------
Method    |    90       90  |    180
          | 81.66    73.90  |  77.78
           -----------------
```

Figure 6.31 Table of Means for Class Nested Within Method

```
    MEANS  Class(Method) ;


   CELL CONTENTS ARE....
      CELL COUNTS

   ---MEAN SCORE OF VARIABLE  -----Score------

                                     CLASS
                                                                       ROW
METHOD           1        2        3        4        5        6     TOTALS
            -----------------------------------------------------
         1 |    10       10       10                             |     90
           | 95.70    93.90    91.30                             |  81.66
           |                                                     |
         2 |                               10       10       10  |     90
           |                            79.30    71.90    68.90  |  73.90
            -----------------------------------------------------

TOTAL N         10       10       10       10       10       10       180
MEAN         95.70    93.90    91.30    79.30    71.90    68.90     77.78
```

```
   ...............


    CELL CONTENTS ARE....
      CELL COUNTS

   ---MEAN SCORE OF VARIABLE  -----Score------

                                   CLASS
                                                                     ROW
METHOD           7        8        9       10       11       12     TOTALS
            -----------------------------------------------------
         1 |    10       10       10                             |     90
           | 75.70    84.40    82.70                             |  81.66
           |                                                     |
         2 |                               10       10       10  |     90
           |                            77.90    66.80    78.20  |  73.90
            -----------------------------------------------------

TOTAL N         10       10       10       10       10       10       180
MEAN         75.70    84.40    82.70    77.90    66.80    78.20     77.78


 ...............



  CELL CONTENTS ARE....
      CELL COUNTS

   ---MEAN SCORE OF VARIABLE  -----Score------

                                   CLASS
                                                                      ROW
METHOD          13       14       15       16       17       18     TOTALS
            -----------------------------------------------------
         1 |    10       10       10                             |     90
           | 64.80    69.20    77.20                             |  81.66
           |                                                     |
         2 |                               10       10       10  |     90
           |                            80.00    62.80    79.30  |  73.90
            -----------------------------------------------------

TOTAL N         10       10       10       10       10       10       180
MEAN         64.80    69.20    77.20    80.00    62.80    79.30     77.78
```

SUMMARY

ANOVA

```
 ANOVA  Drug.Data ;
   MODEL  Temperature = Doctor | Drug ;   or

 ANOVA  Drug.Data ;
   DEPENDENT  Temperature ,
   TREATMENT  Doctor | Drug ;
```

The ANOVA command tests the equality of the means of multiple treatment or classification groups. It does this by comparing the observed variance between the groups with the variance associated with random error within a group (the residual variance). An F statistic and its associated probability are calculated. A significant F value permits the inference that the group means are not equal and that the treatment or classification has a differential effect.

One-way completely randomized, factorial, block, Latin Square, nested, split-plot, and repeated measures designs may be specified using an algebraic notation. Error terms may be explicitly defined, or allowed to default to the residual variance. Sums of squares may be partitioned among contrasts. Covariates may be specified and tables of means may be requested.

The programming language function, SPLIT, may be used to rearrange data collected in a horizontal format into a vertical format with appropriate indices for treatment or classification groups. Two equivalent methods of specifying ANOVA designs are illustrated in the examples above. The MODEL phrase incorporates the DEPENDENT and the TREATMENT phrases. The equal-sign (=) is required in the MODEL phrase.

Required:

ANOVA fn

specifies the name of the required P-STAT file that has the input data. The name of an optional external label file may also be given at the command level.

Optional Identifiers:

LABELS 'fn'

specifies the name of an optional external label file:

```
 ANOVA  DataFile,  LABELS  'LabFile' ;
```

Labels will be used in any requested table of means.

PR 'fn'

changes the print output destination name. The terminal is the default interactive output destination, and the printer is the default batch output destination.

Optional Identifiers: tests

The following tests can be requested for simple one way design analyses

BARTLETT

homogeneity of variance test

DUNCAN

a multiple range test for comparing all pairs of means

DUNNETT nn

compare treatment means with a control group. If the control group value (nn) is not supplied, the group with the lowest value is used.

LSD

Least Significant Difference comparison test.

MOD.LSD

a modification of the LSD test.

SCHEFFE

a test for comparing all possible contrasts between treatment means.

SNK

Student-Neuman-Keuls test.

TUKEY.A

a test based on studentized ranges.

TUKEY.B

a test which uses the average of the SNK and Tukey.a values.

Required Subcommands:

MODEL vn = ds

specifies the dependent variable(s) and the treatment portion of the design for analysis. The MODEL phrase is composed of two parts. The first part is the dependent variable (the response measurement) or a list of such variables. A list should be enclosed in parentheses:

```
MODEL  ( Var.1, Var.2 )      =  design.statement ;
MODEL  ( Var.1  TO  Var.6 )  =  design.statement ;
```

The second part, following the required equal-sign, is a design statement. A design statement is an algebraic sum of terms. The terms are either effects or they expand to a sum of effects. An effect is either a single factor, an interaction of several factors, or a nesting of a factor within another factor. (See Appendix A, preceding the summary of this chapter, for a more comprehensive description of design statements.) An equivalent method of specifying designs that uses the distinct subcommand phrases DEPENDENT and TREATMENT may also be used.

DEPENDENT vn

specifies a dependent variable phrase with the name of the dependent variable (the response measurement) or a list of such variables. DEP is an abbreviation for DEPENDENT, and Y is a synonym.

TREATMENT ds

specifies a design phrase using a design statement. (See the description of design statements under MODEL.) TREATMENT may be abbreviated to TR. A list of variables may also be included as part of a design statement:

```
TR  ( A.1  TO  A.4 ) | B ;
```

which expands to:

```
TR  ( A.1  +  A.2  +  A.3  +  A.4 )  |  B ;
```

which then expands to:

```
TR    A.1  +  A.2  +  A.3  +  A.4   +  B  +
      A.1*B  +  A.2*B  +  A.3*B  +  A.4*B ;
```

Optional Subcommands:

BLOCK ds

specifies an effect or sum of effects whose sum of squares are to be removed from the residual sum of squares — that is, effects for which blocking is desired. Any confounding of treatment effects with blocking effects will not be reported. The treatment effects will be adjusted for the blocks. (Should the confounding be of interest, the effects should be specified with an ERROR.TERM phrase, instead of a BLOCK phrase.) BL is an abbreviation for BLOCK.

COVARIATE ds

specifies a covariate variable phrase with the name of the covariate variable or a list of such variables. COV is an abbreviation for COVARIATE, and X is a synonym.

ERROR.TERM ds

specifies an error term (denominator in F test) when other than the assumed error term of the residual mean square (variance) is desired. The error term specification is a design statement (an effect or sum of effects) which will be used as the denominator in the F tests of the effects in the TREATMENT. The effects in the ERROR.TERM statement will themselves be tested against the residual mean square. ER is an abbreviation for ERROR.TERM. Note: The at-sign (@) may be used as a term in a design statement in either TREATMENT, ERROR.TERM or BLOCK design phrases. It designates the previously specified effects in that design component, and is often used to repeat an analysis:

```
TR  @  ;
```

or to modify an analysis:

```
ER  @  +  Drug ;
```

The ANOVA table is immediately calculated and displayed.

BOX

specifies the outlining of the cell boxes in MEANS tables. This is normally assumed. NO.BOX suppresses the printing of the cell box outlines. See also EDGES.

EDGES T B

indicates which parts of the cell boxes are to be printed. Choices are Top, Bottom, Left, Right, and for symmetry, Horizontal and Vertical, for interior lines. These may be abbreviated to one letter each:

```
EDGES  T B
```

The preceding example will give the table top and bottom edges only.

MEANS ds

specifies a design phrase with a design statement (an effect or sum of effects) for which tables of means are desired.

PLACES nn

specifies the number of places to be used in printing tables of MEANS. The program normally makes a decision about the number of decimal places that can be used, depending upon the size of the values and the cell size available. When PLACES is used, the program will use the number given without checking on the size of the values.

SHOW DESIGN

shows the current design, without modification and without performing an analysis. This is helpful in ascertaining the current error term or block. When this phrase is appended to a modification phrase, such as:

```
ER  Drug*Doctor,  SHOW DESIGN ;
```

the current design will be modified and then displayed.

SHOW VARIABLES

shows the variables in the file currently input to ANOVA. This is helpful when a range of dependent variables is to be indicated and you want to see the order of the variables. This may be abbreviated to SHOW.

V nn

changes the verbosity level within the ANOVA command. This new level remains in effect for the duration of the ANOVA command, unless it is again reset. Verbosity level 2 is the default interactive level and verbosity level 3 is the default batch level. Verbosity level 2 displays the ANOVA and MEANS tables. Verbosity level 3 also automatically displays the expansion of all design statements and, when the output is directed to the printer, prints the expansion before each ANOVA table. Verbosity level 4 also provides additional information on observations with missing values.

Y.SUBSET vn

specifies dependent variables (when more than one dependent or Y variable is used) for which means are to be calculated by subsequent MEANS phrases. If this phrase is not used, means are produced for *all* the dependent variables. X.SUBSET, XY.SUBSET and YX.SUBSET may also be used to specify subsets of the covariates and of the dependent and covariate variables.

7
SURVIVAL:
Survivorship Tables and Functions

The SURVIVAL command analyzes lifetime (survival time) data that may include censored observations. P-STAT provides two commands for survival analysis. The SURVIVAL command uses the product limit method. LIFE.TABLE uses the life table or actuarial method.

A censored observation is one in which the terminating response has not yet occurred. The survival distribution function uses product-limit estimation. When grouping variables define strata, the survival function is computed for each level (subgroup) and tested for homogeneity over strata. When covariates are specified, they are tested for association with lifetime across pooled strata levels.

A plot of the survival distribution function is produced. When strata are defined, overlays on the plot show the different survival functions. Plots of additional related functions are generated automatically. The survival function estimates are written in a P-STAT system file.

The first part of this chapter describes the usage, input and output of the SURVIVAL command and the product limit method. The final section of this chapter describes the usage, input and output of the LIFE.TABLE command which uses the actuarial method.

7.1 BACKGROUND

Survival analysis procedures analyze data that measure the length of time until a terminating response. Typically, the response is death or failure. However, the data may be measurements of time until any terminating response. For example, months until death from cancer, hours of continued usage until transistor failure, days of training until demonstration of a specific skill, or years of marriage until divorce are valid lifetime data. The data may even be measurements other than time-to-response, such as pounds lost until achievement of goal weight or total amount spent until attainment of car ownership.

The fact that the terminating response does not occur in some observations, but that use is made of this information nonetheless, is what distinguishes survival analysis from other nonparametric techniques. When the terminating response does not take place, the observation is considered to be *censored*. Typically, the end of the study “censors” the time-to-response; that is, the study ends before the patient dies or the transistor fails. Sometimes, a patient withdraws or one transistor is lost and this results in the censoring of an observation before the end of the study. Survival analysis uses the fact that the lifetime is at least as long as the time until censoring in its computations.

The requirements for lifetime data are:

1. There is a well-defined starting point for each subject or observation. It may coincide with the start of the study or it may be at some later point when the subject entered the study.
2. There is a well-defined ending point for each subject or observation. It may be when the response (death or failure) occurred or when censoring (withdrawal, loss or end of the study) occurred.
3. The subjects or observations should be from a common population or they may be from random independent groups that have the same patterns of censorship.

Survival analysis is able to handle both the varying starting and ending points and the fact that the exact length of time-to-response is not known for censored observations.

The SURVIVAL command estimates the *survival distribution function* (the survival function). This function describes the lifetimes of the population or of a single group. For a given time, the survival function gives the probability of living at least that long or longer. SURVIVAL uses the *Kaplan-Meier product-limit method* of estimating the survival function. The product-limit method estimates the survival function for each *case*.

The survival distributions for different strata levels are compared (tested for homogeneity) using three nonparametric tests adjusted for the presence of censored observations and a parametric test that assumes an exponential model. The following tests are produced when strata are defined:

1. Gehan's Generalized Wilcoxon Rank Test,
2. Cox-Mantel or Savage Log-Rank Test,
3. Tarone-Ware (square root) Rank Test, and
4. Likelihood Ratio Test.

When covariates are specified, they are tested for association with the time variable across the pooled strata levels. The following tests of association (generalized versions of the tests for homogeneity) are produced:

1. Wilcoxon Rank Test and
2. Log-Rank Test.

When strata are defined, separate plots of the survival distribution function for each strata are overlayed on a single set of axes. They afford a visual comparison of the survival functions. Additional plots include the negative log of the survival function. The plot of the log survival function aids in evaluating whether the survival function is exponential.

7.2 BASIC USAGE

The SURVIVAL command requires an input file containing lifetime measurements and the status of each observation. The lifetime is typically the time elapsed until the occurrence of the terminal event or until censoring of the observation. (Note: time should not equal 0). The status of a case is an indication as to whether the terminal event took place or did not. Missing values on any of the specified variables cause an observation to be excluded from the survival analysis.

7.3 Survival Time Data

The file of lifetime data is input to SURVIVAL directly after the command name:

```
    SURVIVAL  KP2,  TIME  DAYS,  STRATA  Group,  STATUS  Event  $
```

The TIME, STRATA and STATUS identifiers indicate the lifetime, grouping and status-indicating variables. The cases in the input file should be individual observations. They need not be ordered in any way.

Figure 7.1 shows an input data set, file KP2. [1] The variable Days is the lifetime of rats, measured from insult with DMBA, a carcinogen, to death from vaginal cancer or to censoring. The variable Group distinguishes between two pretreatments, and the variable Event indicates censored observations with a code of zero and terminal observations with a code of one.

The TIME identifier designates the lifetime variable, which may be measured in any units. Typical units are time related — days, weeks, years and so on, but they may be any measurements. The STRATA (or BY) identifier specifies one or more grouping variables. (STRATA is discussed in the subsequent section "STRATA AND COVARIATES.") The STATUS identifier indicates the variable that gives the status of each case. The status is either uncensored or censored.

[1] This data is from *The Analysis of Failure Time Data* by John D. Kalbfleisch and Ross L. Prentice. The file name "KP2" refers to the data set shown on page 2. The original source of this data is M.C. Pike, "A method of analysis of certain class of experiments in carcinogenesis, *Biometrics*, 22, 142-161.

Figure 7.1 Lifetime Data: TIME, STRATA and STATUS Variables

```
FILE  KP2:

Days  Group   Event            Days  Group   Event

 143      1       1             156      2       1
 164      1       1             163      2       1
 188      1       1             198      2       1
 188      1       1             205      2       1
 190      1       1             232      2       1
 192      1       1             232      2       1
 206      1       1             233      2       1
 209      1       1             233      2       1
 213      1       1             233      2       1
 216      1       1             233      2       1
 220      1       1             239      2       1
 227      1       1             240      2       1
 230      1       1             261      2       1
 234      1       1             280      2       1
 246      1       1             280      2       1
 265      1       1             296      2       1
 304      1       1             296      2       1
 216      1       0             323      2       1
 244      1       0             204      2       0
 142      2       1             344      2       0
```

7.4 Case Status

The status of each observation may be indicated in one (and only one) of two ways:

1. in a status variable, or
2. as a negative or positive lifetime.

In Figure 7.1, a STATUS variable named Event contains this information. Uncensored observations, that is, those in which the terminal event took place, have a status value of 1. Censored observations have a value of 0. The STATUS identifier indicates which variable contains the status values. That variable may have any name.

Censored observations need not be coded 0 — any status values other than 1 indicate censoring. This permits the use of different values to indicate the various reasons why cases are censored. Withdrawal from treatment, moved to new location, accidental death and other explanations may be coded uniquely. *Uncensored observations should be coded 1.*

The second way to indicate the status of observations is to code the TIME variable *negative for censored* observations and *positive for uncensored* observations. In Figure 7.1, for example, the values for Days of 216 and 244 in group 1 and 204 and 344 in group 2 could be given as -216, -244, -204 and -344, respectively. Then, the STATUS variable would not be necessary.

Figure 7.2 **Comparing the Survival Functions in Different Strata**

```
     SURVIVAL  KP2,  TIME  DAYS,   STRATA  Group,  STATUS  Event $

               PRODUCT LIMIT SURVIVAL ANALYSIS
               -------------------------------

 -------------------------------------------------------------
      GROUP:            Group = 1


QUANTILES
---------
 q1 (25 pct) 190
 q2 (50 pct) 216               Mean           218.75657894737
 q3 (75 pct) 234               Standard Error 9.403179023235


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED
 19      17      2       .1053

 -------------------------------------------------------------
       GROUP:           Group = 2


QUANTILES
---------
 q1 (25 pct) 232
 q2 (50 pct) 233               Mean           240.79464285714
 q3 (75 pct) 280               Standard Error 11.205971747346


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED

 21      19      2       .0952

 -------------------------------------------------------------

 ------- Homogeneity of Survival Functions over Strata


 Test             Chi-Square  DF   Probability

 Wilcoxon            2.651     1      .1035
 Log-Rank            3.123     1      .0772
 Tyrone-Ware         2.977     1      .0845

 Likelihood Ratio                .077     1      .7807
```

Figure 7.3 **Product-Limit Survival Function Estimates: Partial Listing**

```
 ------------------------- strata.number: 1 ----------------------------

                                                    Number  Number   Sta
 Group  Days     Survival  Survival SE     Failure  Failed    Left   tus

     1     0     1.000000    0.          0.            0      19     1
         143     0.947368    0.0512278   0.052632      1      18     1
         164     0.894737    0.0704059   0.105263      2      17     1
         188     0.842105    0.0836547   0.157895      3      16     1
         188     0.789474    0.0935288   0.210526      4      15     1
         190     0.736842    0.1010226   0.263158      5      14     1
         192     0.684211    0.1066392   0.315789      6      13     1
         206     0.631579    0.1106647   0.368421      7      12     1
         209     0.578947    0.1132690   0.421053      8      11     1
         213     0.526316    0.1145489   0.473684      9      10     1
         216     0.473684    0.1145489   0.526316     10       9     1
         216*    -           -           -            10       8     0
         220     0.414474    0.1145153   0.585526     11       7     1
         227     0.355263    0.1124262   0.644737     12       6     1
         230     0.296053    0.1081624   0.703947     13       5     1
         234     0.236842    0.1014502   0.763158     14       4     1
         244*    -           -           -            14       3     0
         246     0.157895    0.0934313   0.842105     15       2     1
         265     0.078947    0.0727921   0.921053     16       1     1
         304     0.          0.          1.000000     17       0     1

------------------------- strata.number: 1 ----------------------------
  -                        Log Neg
 Group   Neg Log.S       Log.S    Log Time

     1    0.          -           -
          0.054067   -2.917527    4.962845
          0.111226   -2.196194    5.099866
          0.171850   -1.761132    5.236442
          0.236389   -1.442277    5.236442
          0.305382   -1.186193    5.247024
          0.379490   -0.968928    5.257495
          0.459532   -0.777546    5.327876
          0.546544   -0.604141    5.342334
          0.641854   -0.443395    5.361292
          0.747214   -0.291403    5.375278
          -           -           -
          0.880746   -0.126986    5.393628
          1.034896    0.034301    5.424950
          1.217218    0.196568    5.438079
          1.440362    0.364894    5.455321
          -           -           -
          1.845827    0.612927    5.505332
          2.538974    0.931760    5.579730
          -           -           5.717028
```

PPL may be used to change from one type of coding to the other. In the following command a STATUS variable is generated and variable Housrs is recoded as a P-STAT system file is input to the SURVIVAL command:

```
       SURVIVAL  Bulbs
           [ IF  Hours  >=  0,  GEN  Code = 1,  F.SET  Code = 0 ],
        STATUS  Code,  ....
```

Alternatively, this makes the lifetimes of censored observations negative:

```
      SURVIVAL  LargeTV
           [ IF  End.Code  NE  1,  SET  Time  =  -1 * Time;
             DROP  End.Code  ],  ....
```

There is no need to change from one type of status coding to the other — either is acceptable in the SURVIVAL command.

Figure 7.4 Plot of Survival Distribution Function

```
Legend for Survival Plots

A: Group = 1
B: Group = 2


                     Survival Distribution Estimates

      1.0 +*----------------------+
          |                       *-++
S      .9 +                         BA---+
u         |                          B---A-+
r      .8 +                              A B+
v         |                              A+ B---+
i      .7 +                               A-+   B
v         |                                 A   B
a      .6 +                                 A+  B
l         |                                  A+ B
       .5 +                                   | B
F         |                                   A B+
u      .4 +                                   A+ B
n         |                                    A+B---+
c      .3 +                                     A    B--+
t         |                                     A-+     B
i      .2 +                                       |     B-+
o         |                                       A--+    B
n      .1 +                                          A----B-+--+
          |                                                 |  B
       .0 +                                                 A
           +-------+-------+-------+-------+-------+-------+-------+
           0       50     100     150     200     250     300     350
                                     Days
```

7.5 Product-Limit Estimation

Kaplan-Meier product-limit estimates of the survival function are calculated by default.

```
SURVIVAL  KP2,  TIME  DAYS,  STRATA  Group,  STATUS  Event $
```

Figure 7.2 shows the report produced by SURVIVAL for file KP2. Figures 7.3, 7.4, 7.5 and 7.6 show the listing and plot of the product-limit estimates of the survival function. Figure 7.7 shows the plot of the censored observations.

Figure 7.5 Plot of Negative Log Survival Estimates

```
                       Negative Log Survival Estimates

      3.0 +                                                    B
          |
          |
          |
      2.5 +                                          A
          |
N         |                                               B
e         |
g     2.0 +
a         |                                      A       B
t         |
i         |                                             B
v     1.5 +                                    A
e         |                                             B
          |                                    A    B
L         |
o     1.0 +                                   A B
g         |                                  A  B
          |                                  A B
          |                                 A  B
       .5 +                                A   B
          |                              A     B
          |                             A BB
          |                        B*   A
       .0 +*                      *
           +-------+-------+-------+-------+-------+-------+-------+
           0       50     100     150     200     250     300     350
                                     Days
```

7.6 Listing the Output

The survival estimates and their standard error are written in a P-STAT system file. When the OUT identifier supplies a name for the system file:

```
SURVIVAL  KP2,  TIME  DAYS,  STRATA  Group,  STATUS  Event,
     OUT  KP2ple  $
```

the estimates are written in that file. When OUT is not used, the estimates are written in a temporary system file whose name begins with “WORK” — this file disappears when the P-STAT run ends. It may be renamed prior to exiting P-STAT. The "*" in the listing indicates a censored observation.

The system file of estimates is automatically listed with an appropriate format when SURVIVAL completes. The LIST *command* can be used to produce a listing of the estimates file at any subsequent time. This file is a regular P-STAT system file and, like other system files, it may be input to any P-STAT commands.

The listing that is produced by SURVIVAL is a stubbed one. Any STRATA variables and the TIME variable are positioned on the left of each page of the listing.

Either the listing or the plots can be surpressed:

```
SURVIVAL KP2,  NO LIST,  .....
```

would produce the printout show in Figure 7.2 and the four plots. The listings in Figure 7.3 would not be printed.

```
SURVIVAL KP2,   NO LIST,  NO PLOT,  ....
```

would produce neither plots nor listings. Only the printout shown in Figure 7.2 would be shown.

Figure 7.6 Log (Negative Log) Survival Plot

```
                      Log (Negative Log) Survival

      2.0 +
S         |
D     1.5 +
F         |
      1.0 +                                         A           B
-         |                                               B
       .5 +                                    A       B
L         |                                AA      B   B
o      .0 +                               A   B
g         |                             AA  B
      -.5 +                           AA    B
N         |                          A      B
e    -1.0 +                     A           B
g         |                     A   B
a    -1.5 +                    A  B
t         |            B       A
i    -2.0 +
v         |         B  A
e    -2.5 +
          |
L    -3.0 +   BA
o         |
g    -3.5 +
          |
     -4.0 +
           +-----+-----+-----+-----+-----+-----+-----+-----+-----+
          4.9    5    5.1   5.2   5.3   5.4   5.5   5.6   5.7   5.8
                                  Log.Time
```

Figure 7.7 Plot of Censored Observations

```
                Censored Observations

s         |
t         |
r         |
a         |
t       2 + B                                         B
a         |
.         |
n         |
u       1 +     A       A
m         |
b         |
e         |
r         |
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
          200   220   240   260   280   300   320   340
             210   230   250   270   290   310   330   350
                               Days
```

7.7 STRATA AND COVARIATES

Typically, an investigator wants to compare the survival functions of different groups. For example, she might be interested if there is any difference between cancer patients receiving chemotherapy or radiation treatment after surgery, or between laptop computers used at temperatures above 80 degrees and those used at lower temperatures. In addition, a scientist may want to assess the relationship of other variables to lifetime. For example, she might be curious if cholesterol level is associated with the lifetimes of cancer patients, or if the amount of disk storage or access speed are related to laptop lifetime.

These additional variables can be used either to define the strata which permits any differences in the survival distribution (SDF) functions to be visually inspected or they can be used to prepare statistics which measure the association between each covariate and survival time. When STRATA (or its synonym BY) specifies grouping variables, the SURVIVAL command computes separate survival functions and provides four tests that aid in evaluating differences between them. When the COVARIATES identifier designates one or more possibly-related variables, two tests evaluate the strength of their association with lifetime across the pooled strata levels.

7.8 Comparing Strata

From one to twelve numeric and/or character STRATA variables may be specified. The *case* of character STRATA variables is not considered significant in defining strata. If case should be considered, the EXACT identifier should be included in the SURVIVAL command. Then, for example, the values “A” and “a” of the STRATA variable Treatment would define separate groups. Normally, the strata levels go from the lowest through the highest values in the output file. If the opposite order is desired, the DOWN identifier should be included in SURVIVAL.

Figure 7.8 SURVIAL Output With Covariates

```
            PRODUCT LIMIT SURVIVAL ANALYSIS
              -------------------------------

  137 cases are being used from file kp223,
    0 cases were dropped because of missing data.

    2 is the largest number of non-censored cases in any strata
      that are tied on a single time value.

    4 strata are being processed.
    5 covariates will be analyzed.


----------------------------------------------------------
        GROUP:             Cell = 1


QUANTILES
---------
q1 (25 pct) 33
q2 (50 pct) 118              Mean           230.22506265664
q3 (75 pct) 357              Standard Error 48.474615711217


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED
35      31      4        .1143


----------------------------------------------------------
        GROUP:             Cell = 2


QUANTILES
---------
q1 (25 pct) 20
q2 (50 pct) 51               Mean           78.981095679012
q3 (75 pct) 99               Standard Error 14.837346199483


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED
48      45      3        .0625


----------------------------------------------------------
        GROUP:             Cell = 3


QUANTILES
---------
q1 (25 pct) 19
```

```
q2 (50 pct) 51              Mean           65.555555555556
q3 (75 pct) 92              Standard Error 10.126944295397


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED
27      26      1        .0370


----------------------------------------------------------
        GROUP:             Cell = 4


QUANTILES
---------
q1 (25 pct) 53
q2 (50 pct) 156             Mean           170.50617283951
q3 (75 pct) 231             Standard Error 25.097949422328


TOTAL  EVENTS  CENSORED  PROPORTION CENSORED
27      26      1        .0370


----------------------------------------------------------
 Homogeneity of Survival Functions over Strata


Test             Chi-Square    DF   Probability

Wilcoxon            19.4331     3         .0002
Log-Rank            25.4037     3         .0000
Tyrone-Ware         22.5728     3         .0000

Likelihood
Ratio               33.9343     3         .0000


Rank Tests for the Association of Survival Time with Covariates

    Chi-Square Statistics for the Log-Rank Test

                    Test
    Variable        Statistic    Variance     Chi-Square   Probability

Age                 -40.73832    11175.4402        .1485      .7000

Prior               -19.94349     2207.4607        .1802      .6712

Months              -115.8562     9578.6865       1.4013      .2365

Perform             1123.1412    29015.6156      43.4747      .0000

Therapy             -4.207553      25.40904        .6967      .4039
```

When DOWN is not used, UP is assumed. As is the case in other P-STAT sort situations, specific sort directives may follow individual STRATA (BY) variables. This portion of a SURVIVAL command specifies that the STRATA variables sort in descending order, except for Dosage, which sorts in ascending order:

```
STRATA  Group  Dosage  (U)  Temperature,  DOWN
```

Any individual sort directives override UP and DOWN for the specific variable they follow. The data used in Figure 7.8[2] comes from a study of males with inoperable lung cancer. The cancers are classified into four different cell-types. The Cell variable is used as the STRATA variable. The SURVIVAL command is:

```
SURVIVAL  KP223,  TIME Days,  STATUS Event,  STRATA Cell
          COVARIATES  Age  Prior  Months  Perform  Therapy,
          NO  LIST,   NO  PLOT $
```

An examination of the test results which measure the equality of the SDF between the different treatment groups indicates that the survival times between the different cell types are significantly different. All four tests have a probability that the chi-square is significant of less than .000.

7.9 Testing Covariates

The COVARIATES identifier is used to provide a list of other variables which may influence survival. The tests done on the covariates are for done for each variable in turn. The tests adjust for effects of the strata variables by pooling over any defined strata.

In the data set used in Figure 7.8 five variables are entered as covariates. The only variable among the five that appears to have any prognostic effect is “Perform” which is a measure of how hospitalized the patients are. This variable has a chi-square of 43.4747 and a probability of .0000 on the log-rank test.

7.10 LIFE TABLE ANALYSIS

The LIFE.TABLE command analyzes lifetime (survival time) data using the lifetable (actuarial) method. There are three command forms.

1. LIFE.TABLE KP2, TIME days, STRATA group $
2. LIFE.TABLE KP2, TIME days, STRATA group, STATUS event $
3. LIFE.TABLE KP2I, TIME time, STRATA group, FAIL dead, WITHDRAWN lost $

The first form has no STATUS variable, and uses a negative value on TIME to indicate a censored case. The second form has a STATUS variable; a value of one on the STATUS variable shows not censored, anything else means censored. The TIME variable cannot be negative in this input form. The third form permits grouped cases for a given TIME value. The FAIL variable holds the non-censored count. The WITHDRAWN variable holds the censored count.

Figure 7.9 LIFE.TABLE Example

```
LIFE.TABLE KP2, TIME Days,  STRATA group,  STATUS Event, NO LIST $


         LIFE TABLE ANALYSIS
         -------------------

40 cases are being used from file KP2,
 0 cases were dropped because of missing data.
```

[2] The data set is from *The Analysis of Failure Time Data* by John D. Kalbfleisch and Ross L. Prentice, Page 223.

```
      2 strata are being processed.

--------------------------------------
        GROUP:            Group = 1


QUANTILES
---------
q3 (75 pct) 190.92
q2 (50 pct) 218.17380952381
q1 (25 pct) 240.54404761905


TOTAL EVENTS CENSORED  PROPORTION CENSORED
19      17      2        .1053



--------------------------------------
        GROUP:            Group = 2


QUANTILES
---------
q3 (75 pct) 207.18024193548
q2 (50 pct) 228.42016129032
q1 (25 pct) M1


TOTAL EVENTS CENSORED  PROPORTION CENSORED
21      19      2        .0952


 Homogeneity of Survival Functions over Strata




Test             Chi-Square    DF   Probability

Wilcoxon              .7881     1         .3747
Log-Rank             1.5076     1         .2195



Legend for Survival Plots



A: Group = 1

B: Group = 2
```

```
                       Survival Distribution Estimates

      1.0 +*-----*----*-----*----*-----+
          |                            |
S      .9 +                            A----+
u         |                            B----|
r      .8 +                                 |
v         |                                 B-----+
i      .7 +                                 |     |
v         |                                 A-----|
a      .6 +                                       |
l         |                                       |
       .5 +                                       |
F         |                                       |
u      .4 +                                       |
n         |                                       B----+
c      .3 +                                       |    B-----+
t         |                                       A----+     |
i      .2 +                                            |     |
o         |                                            |     |
n      .1 +                                            A-----B
          |                                                  |
       .0 +                                                  A
           +-------+-------+-------+-------+-------+-------+-------+
           0       50     100     150     200     250     300     350
                                     Time
```

```
                         Hazard Function Estimates

      .06 +
          |                                                     A
H         |                                                    +
a     .05 +                                                    +
z         |                                                   +
a         |                                                  +
r     .04 +                                                  +
d         |                                                 +
          |                                                +
F     .03 +                                                +    B
u         |                                        A      +    +
n         |                                       + +++++A     +
c     .02 +                                      + B          +
t         |                                     + + +        +
i         |                                    + +   +      +
o     .01 +                                  A+ +     +    +
n         |                               +++  +       +  +
          |                            A+++++B+         +B
      .00 +   A+++++A+++++A++++++A+++++
           +--------+--------+--------+--------+--------+--------+--------+
           0        50      100      150      200      250      300
                                 Time.mid.pt
                         Density Function Estimates

     .012 +                                        B
D         |                                       +A
e         |                                       ++
n    .010 +                                      ++ +
s         |                                     ++  +
i         |                                    + +  ++
t    .008 +                                   + +    +
y         |                                  A  +    ++
          |                                 +  +      +
F    .006 +                                +   +      ++        B
u         |                               +    +      ++       +
n         |                              +    +        ++     +
c    .004 +                            B+     +        + A   +
t         |                           +A+++  +         +  ++++++A
i         |                          +     ++B          +  +
o    .002 +                         +                   + +
n         |                        +                     B
          |                       +
     .000 +   A+++++A+++++A++++++A
           +--------+--------+--------+--------+--------+--------+
           0        50      100      150      200      250      300
                                 Time.mid.pt
```

```
                       Negative Log Survival Estimates

      3.0 +
          |
          |
N     2.5 +
e         |                                                  B
g         |                                            A    +
a     2.0 +                                           +     +
t         |                                          +     +
i         |                                         +     +
v     1.5 +                                        +     +
e         |                                       A     +
          |                                      +     B
L     1.0 +                                     + B++++
o         |                                    + +
g         |                                   +++
       .5 +                                 A++
          |                               ++B+
          |                            A++++
       .0 +A     A++++A+++++A    A+++++
           +-------+-------+-------+-------+-------+-------+-------+
           0       50     100     150     200     250     300     350
                                     Time
```

```
                         Log (Negative Log) Survival

      1.0 +
S         |                                            A     B
D      .5 +                                          ++   +++
F         |                                       A++  B++
       .0 +                                      +B++++
-         |                                    +++
      -.5 +                                   + +
L         |                                 A+ +
o    -1.0 +                                +  +
g         |                                +B+
     -1.5 +                               ++
N         |                            B++
e    -2.0 +                             +
g         |                            A
a    -2.5 +
t         |
i    -3.0 +
           +-------+-------+-------+-------+-------+-------+-------+
           0       50     100     150     200     250     300     350
                                     Time
```

```
                        Log (Negative Log) Survival

      1.0 +
S         |                                               A B
D      .5 +                                              + +
F         |                                             A B
       .0 +                                            +B+
-         |                                            +
      -.5 +                                           ++
L         |                                          A+
o    -1.0 +                                         + +
g         |                                         +B
     -1.5 +                                        ++
N         |                                       B+
e    -2.0 +                                       +
g         |                                       A
a    -2.5 +
t         |
i    -3.0 +
           +--------+--------+--------+--------+--------+--------+
           3       3.5       4       4.5       5       5.5       6
                                  Log.Time
```

7.11 LIFE.TABLE Identifiers:

LIFE.TABLE requires a P-STAT system file as input. If the P-STAT system file is not named, the most recently referenced file is used as input. TIME specifies the name of the numeric variable that is the survival time. Can be negative if using the first command form. The TIME variable is requires in all three forms of the command. STA-

TUS is required in the second form. It specifies the name of the numeric variable whose values give the status of each observation. Uncensored data must be coded with a 1 on the status variable. FAIL which is required in the third form, specifies the name of the numeric variable containing the number of cases that failed at the current time (or in the current interval that includes the time) WITHDRAWN is required in the third form above. It specifies the name of the numeric variable containing the number of cases that were censored.

There are a number of optional Identifiers. STRATA (or BY which is a synonym) supplies the names of from 1 to 12 grouping variables. Separate survival functions are calculated for each subgroup. EXACT requests that the sort of the character BY variables respect the case of the character strings. OUT supplies a name for the output P-STAT system file which contains the life table and survival functions. WIDTH n requests intervals of size n.

INTERVALS n requests n intervals. TIME.INTERVALS can be used if the time values are integer interval values. It produces interval boundries corresponding to the values. No more than one of WIDTH, INTERVALS, and TIME.INTERVALS can be used. If none are used, INTERVALS 10 is assumed.

HIGH n specifies that the intervals should reflect time values from 0 to n, which should be greater than or equal to the largest time value. A LOW value may also be specified. NO LIST prevents listing of the results file. NO PLOT prevents various plots from occurring.

Figure 7.9 provides an example of the plots produced by the LIFE.TABLE command. The list of the survival estimates was suppressed by the NO LIST identifier.

7.12 REFERENCES

1. Breslow, N. E. (1974). *Covariance Analysis of Censored Survival Data*, Biometrics, 30, 89-99.
2. Kalbfleisch, John D. and Prentics, Ross L. (1980). *The Statistical Analysis of Failure Time Data*, John Wiley & Sons, New York.
3. Lee, Elisa T. (1980). *Statistical Methods for Survival Data Analysis*, Lifetime Learning Publications, Belmont, California.
4. Miller, R. G., Jr. (1981). *Survival Analysis*, John Wiley & Sons, New York.
5. Petro, R. (1972). *Contributions to the Discussion of a Paper by D. R. Cox*, Journal of the Royal Statistical Society B, 34, 205-207.
6. Woolson, Robert F. (1987). *Statistical Methods for the Analysis of Biomedical Data*, John Wiley & Sons, New York

SUMMARY

SURVIVAL

```
SURVIVAL  Study3,  BY   Group,  STATUS  Alive,
    TIME  Days,    OUT  Study3LT  $
```

The SURVIVAL command calculates the survival distribution function using the product-limit estimator. The output tables are saved as P-STAT system files. When the survival analysis is done by strata (subgroups), the homogeneity of the survival functions across the strata is tested. When covariates are specified, they are tested for association with the time variable. Cases with missing values on any of the specified variables are excluded from the analysis.

Required:

SURVIVAL fn

gives the name of the input P-STAT system file. The cases in the file are expected to be individual observations (“subjects”). The variables in the input file typically include a TIME variable giving the survival time, a STATUS variable telling whether the observation is censored (“lost”) or uncensored (“dead”) and, optionally, one or more BY variables defining strata and optionally one or more COVARIATES variables.

Required Identifiers:

TIME vn

specifies the name of the numeric variable that is the survival time — this is the time between when the subject entered the study and when either death or censorship (end of the study, drop out, etc.) occurred. The survival times of *censored* observations should be preceded by a minus sign (-) or the STATUS identifier should be used to indicate any censored observations (1 = uncensored).

Optional Identifiers:

BY vn vn ....

supplies the names of from one to twelve character or numeric variables that define strata or subgroups. When BY is used, the output is ordered automatically by the grouping variables. Separate survival functions are calculated for each subgroup. The functions are tested for homogeneity across groups. STRATA is a synonym for BY.

COVARIATES vn vn ....

specifies numeric variables that may be covariates of the time variable. The variables are tested for association with the time variable and pooled over all strata.

DOWN

requests that the output be in descending order of the grouping or strata variables. When DOWN is not used, the grouping variables are in ascending order. Individual sort directives may followed any of the BY variables — U for up and D for down:

```
BY  Dosage (U)  Temperature,  DOWN,
```

Here, Dosage is in ascending order and Temperature is in descending order. Any sort directives override the DOWN or UP identifiers for the specific variable they follow.

EXACT

requests that the sort of the character BY variables respect the case of the character strings. Normally, the case of strings is ignored in sorting.

NO LIST

requests that the automatic listing of the OUT file be suppressed.

NO PlOT

requests that the automatic plots be suppressed.

OUT fn

supplies a name for the output P-STAT system file that contains the life table and survival functions. When OUT is not used, the output is placed in a temporary file (one whose name begins with “WORK”). This file disappears when the P-STAT run ends, unless it is renamed prior to then. The output file is automatically listed.

PR 'fn'

directs the output produced by the SURVIVAL command to the specified disk file or on-line printer. The report, listing and plots do not appear on the terminal, but in the disk file or printout.

STRATA vn vn ....

supplies the names of from one to twelve character or numeric variables that define strata or subgroups. STRATA is a synonym for BY — see the explanation under BY.

STATUS vn

specifies the name of the numeric variable whose values give the status of each observation. Uncensored observations should have a status of 1; censored observations may have any other values.

When STATUS is not used, censored observations should be indicated by coding the corresponding values of the TIME variable as *negative* numbers.

UP

requests that the output be in ascending order of any grouping variables. This is assumed unless DOWN is specified. (See the example under DOWN.)

LIFE.TABLE

```
LIFE.TABLE T432,  TIME Days,  Strata  Group $

LIFE,TABLE T432,  TIME Days,  Strata  Group,   STATUS Event $

LIFE.TABLE T432I, Time Days,  Strata  Group,
     FAIL V99,    WITHDRAWN  V100 $
```

The LIFE.TABLE command analyzes lifetime (survival time) data using the lifetable (actuarial) method. The examples above illustrate the three forms of the command.

Required:

LIFE.TABLE fn

provides the name for the P-STAT system file

TIME vn

provides the name for the variable containing the survival time. In the first form in the examples above, censored cases are indicated by a negative time variable.

STATUS vn

required in the second form of the command provides the name of the variable containing the status of censored data. A 1 is used for uncensored data. Anything else is censored.

FAIL vn

required in the third form where the data are grouped, provides the name of the variable which contains the number of cases in the group that failed at the current time or within the current time interval.

WITHDRAWN vn

required in the third form where the data are grouped, provides the name of the variable which contains the number of cases in the group that were censored.

Optional Identifiers:

STRATA vn vn ..

provides the name of the group variable. BY is a synonym for strata

DOWN

allows downward sort values for the strata variable.

EXACT

requests that the sort of the character STRATE (BY) variables respect the case of the character strings.

OUT fn

supplies a name for the output P-STAT system file which contains the life table and survival functions.

WIDTH nn

requests intervals of size nn.

INTERVALS nn

requests nn intervals.

TIME.INTERVALS

can be used if the time values are integer interval alues. Produces interval boundries corresponding to the values.

HIGH nn

specifies that the intervals should reflect time values from 0 to nn, which should be greater than or equal to he largest time value.

LOW nn

specifies the low values which must be less than the value specified for HIGH.

NO LIST

prevents listing of the results file.

NO PLOT

prevents the plots from occurring

UP

allows the STRATA vars to be sorted upwards.