Syntax

Using SPSS/PC means becoming familiar with its language. Every effort has been made to keep the language natural, consistent, and straightforward.

Syntax Diagrams

Each SPSS/PC command described in this Command Reference includes a syntax diagram that shows all the subcommands, keywords, and specifications allowed for that command. The syntax diagram is used to show all the specifications for a command. By remembering the following rules, you can use the syntax diagram as a quick reference for each command.

- Elements shown in capital letters are keywords.
- Elements in lower case describe specifications supplied by the user.
- Elements in boldface type are defaults. Some defaults are indicated with **.
- Special delimiters, such as parentheses, apostrophes, or quotation marks, are required where indicated.
- Elements enclosed in square brackets ([ ]) are optional. When brackets would confuse the format, they are omitted. The command description explains which specifications are required or optional.
- Braces ({}) indicate a choice between elements.
- The word "varlist" stands for a list of variable names.
- The command terminator is not shown in the syntax diagram.

Command Order

There is little formal precedence order for SPSS/PC commands. Understanding how SPSS/PC works will make it easy to specify commands in the right order.

- Variables must be defined before they can be used in procedures or assigned missing values or labels. You must use one DATA LIST, GET, or IMPORT command to define variables for a session. IF, COUNT, and COMPUTE commands also define variables.
- The logical outcome of command processing frequently determines the order: although data transformations are not carried out until the data are read, the result is as though the commands were executed when encountered.
- The order of commands often affects the SPSS/PC active file.

Commands

Commands are the instructions that you give SPSS/PC to initiate an action. The following rules apply to all SPSS/PC commands.

- Commands begin with a keyword that is the name of the command.
- Each command ends with a command terminator. The default command terminator is a period (.).
- The maximum length of a command line is 80 characters, including the prompt and the command terminator.
- Commands can begin in any column of a command line and continue for as many lines as needed. The exception is the END DATA command, which must begin in the first column of the first line after the end of data.
- Spaces can be added or lines broken at any point where a single blank is allowed. The only exceptions are the END DATA command, which can have only one space between words, and title and subtitle specifications, which cannot be broken across two lines.

- Commands can be entered in upper and lower case. Commands, subcommands, keywords, and variable names are translated to upper case before processing. All user-specifications, including labels and data values, preserve upper and lower case.

Subcommands

Many commands contain options for locating data, handling data, and formatting the output display. These optional specifications are called subcommands.

- Subcommands begin with a keyword that is the name of the subcommand.
- Subcommands are specified as a single keyword or as a keyword with additional specifications, called arguments.
- Subcommands are separated from one another by a slash.
- A subcommand keyword is separated from its specifications, if any, by an equals sign. The equals sign is optional unless the first three characters of the keyword conflict with a variable name.
- Most subcommands can be named in any order. However, some commands require a specific order for subcommand specification.
- The description of each command states which subcommands are optional.

Keywords

Keywords are words specially defined by SPSS/PC to identify commands, subcommands, functions, operators, and other specifications.

- Keywords can be truncated to the first three characters of each word. The only exception is the keyword WITH, which must be spelled in full.
- Some keywords are reserved and cannot be used as variable names. Logical operators (AND, OR, and NOT), relational operators (EQ, GE, GT, LE, LT, and NE), and ALL, BY, TO, and WITH are the reserved keywords.
- Keyword ALL is used to refer to all user-defined variables in the active file.
- Keyword THRU is used between two values to specify a range. The specification includes the values as endpoints of the range.
- Keyword TO is used between two variable names to specify an inclusive list of variables.

Values

Values refer to specifications in commands or the data points processed by SPSS/PC.

- A number specified as an argument for a subcommand can be entered with or without leading zeros.
- Data values of numeric variables can be specified as integers or real numbers, with or without leading zeros.
- Whenever values of string (alphanumeric) variables are used in commands, they must be expressed in apostrophes or quotation marks, including all blanks.
- Blanks within apostrophes or quotation marks are significant.

Delimiters

Delimiters are used to separate data values, keywords, arguments, and specifications.

- The blank is usually used to delimit one specification from another, except when another delimiter serves the same purpose or where the comma is required.
- Arithmetic operators (+, -, *, and /) serve as delimiters in expressions and can be optionally preceded or followed by blanks.
- Special delimiters such as parentheses, apostrophes, quotation marks, the slash, the equals sign, and so forth, serve as delimiters and can be optionally preceded or followed by blanks.
- The slash is used primarily to separate subcommands and lists of variables. Although slashes are sometimes optional, entering them as shown in the syntax diagrams is good practice.

- The equals sign is used between a subcommand and its specifications, as in FILE='filename', and to show equivalence, as in old variable list=new variable list. Equals signs following subcommands are frequently optional, but it is best to enter them.
- Commas are required to separate arguments to functions. Otherwise, commas are generally valid substitutes for blanks and vice versa.

Strings

The word "string" is used to refer to alphanumeric data or specifications such as titles and labels.

- String variables can contain alphabetical characters as well as numbers.
- String variables that contain 8 or fewer characters as values are referred to as *short string variables*. String variables that contain more than 8 characters and up to 255 characters are referred to as *long string variables*.
- String values in data files or entered between BEGIN DATA and END DATA commands do not need to be enclosed in special delimiters (see DATA LIST: Freefield Format for exceptions).
- Only short string variables or values can be used in SPSS/PC transformation commands.
- Each string used in a command can be specified in either apostrophes or quotation marks. You cannot mix apostrophes and quotation marks on a single variable.
- String specifications cannot be broken across command lines.

Files

SPSS/PC uses a number of files in its operation. This section provides an overview to the types of files you can use, as well as a discussion of the active file.

Types of files

The files that you can use in a session are

active file *The file most recently defined in a session by a DATA LIST, IMPORT, or GET command.* The active file contains the data and a data dictionary (see below) and is available until you enter FINISH. You can modify this file using transformations, analyze it using any of the procedures, and save it with EXPORT or SAVE. The active file is stored in SPSS/PC workspace and cannot be named.

command file *A file that contains SPSS/PC commands.* You can use an INCLUDE command to process the commands in a command file as an alternative to entering them interactively. For additional information, see INCLUDE.

data file *A file that contains only raw data.* You supply the name of this file on a DATA LIST command. Data files can be arranged in fixed or freefield format. In fixed-format files, the values of each variable for each case are recorded in the same location on each record. In freefield format, the values of each variable are recorded in the same order but not necessarily in the same location (see DATA LIST: Fixed Format and DATA LIST: Freefield Format).

disk file *A file containing output from SPSS/PC procedures.* You supply filenames for disk files with the SET command.

log file *A file created by SPSS/PC that contains all the commands you enter and that are processed during a session.* Filenames for log files can be specified on the SET command. You can use a log file as a command file in subsequent sessions.

results file *A file to contain procedure data output.* This file can contain a rectangular data set produced by the WRITE command or matrix materials produced by CLUSTER, CORRELATION, FACTOR, ONEWAY, or REGRESSION. Filenames for results files can be specified using the SET command.

system file *A binary file created by the SPSS/PC SAVE command.* System files contain data and a dictionary which can be read using the SPSS/PC GET command. You can name system files using the OUTFILE subcommand on the SAVE command. You can read such a file, using the FILE subcommand of the GET command.

portable file *A portable ASCII file containing data and a dictionary.* A portable file can be created by SPSS/PC or SPSS^X using the EXPORT command or by SAS using PROC TOSPSS. Portable files can be read by either SPSS/PC or SPSS^X. You name this file using the OUTFILE subcommand on the EXPORT command. Portable files are read using the IMPORT command, specifying the name of the file on the FILE subcommand. Once a portable file has been read, an active file is created.

- All filenames must be enclosed in apostrophes.
- Unless otherwise specified, file specifications default to the current drive and directory.
- Not all types of files can be read from or written to a directory other than the current directory.
- If a file can be read from or written to another directory, the path or directory name must be explicitly specified.
- Since the SPSS/PC key diskette is periodically checked for in drive A during operation, drive A is not recommended for reading and writing files.

SPSS/PC Active File

When SPSS/PC processes data definition commands, it builds an internal file called the *active file*. The active file contains data and an associated data dictionary (stored in memory) of variable names, variable and value labels, missing-value flags, and format specifications. The active file data can be altered by transformation commands and is used as input for SPSS/PC procedures.

- The active file is defined with a DATA LIST, GET, or IMPORT command.
- The active file can contain up to 200 variables.
- Each 8-character portion of a long string variable counts toward the system limit of 200 variables. For example a 20-character long string variable counts as 3 toward the system limit of 200 variables.
- The active file is available for use with SPSS/PC commands until the FINISH command is entered.
- The data contained in the active file are always read when a procedure is executed.
- The COMPUTE, COUNT, IF, and RECODE transformation commands cause the active file data to be altered prior to processing data for a procedure. When these commands are entered prior to a procedure, the data are passed twice: once to incorporate the instructions from the transformation commands, and then to perform the SPSS/PC procedure.
- The N command limits the number of cases processed from the active file. The SELECT IF command limits the number of cases processed based upon some logical criteria. SELECT IF permanently affects the number of cases in the active file.
- The SAMPLE and PROCESS IF commands affect the number of cases only for the next procedure.
- The FORMATS, VALUE LABELS, and VARIABLE LABELS commands affect the active file dictionary, not the data. No transformation pass occurs.
- The WEIGHT command affects the value of the existing system variable, $WEIGHT. No transformation pass occurs.

Variables

This section describes rules for defining variables in SPSS/PC.

Variable-Naming Conventions

Variable-naming conventions are the rules used to establish variable names in the active file dictionary and to refer to variables in commands.

- Variable names have a maximum of eight characters. The first character must be an alphabetical letter or a dollar sign ($).
- SPSS/PC creates special system variables which begin with a dollar sign ($). These variables cannot be named on the DATA LIST, GET, or IMPORT commands.
- Variable names are established on the DATA LIST, COMPUTE, COUNT, or IF command.
- You can establish the names of a set of variables on DATA LIST using the TO convention. Specify a character prefix with a numeric suffix before and after the keyword TO. The prefix can be any valid name and the number suffixes can be any integers, so long as the first number is smaller than the second. Each variable name, including the number, must not exceed eight characters. For example, ITEM1 TO ITEM5 establishes five variables named ITEM1, ITEM2, ITEM3, ITEM4, and ITEM5.
- With the TO convention, leading zeros used as suffixes are included in the variable name. For example, V001 TO V100 establishes 100 variables, V001, V002, V003, . . . , V100. V1 TO V100 establishes 100 variables, V1, V2, V3, . . . , V100.
- The TO keyword can also be used on commands other than DATA LIST to refer to a set of consecutive variables on the active file. AVAR TO VARB refers to the variables AVAR and all other variables up to and including VARB on the active file. Use the DISPLAY command to see the order of variables on the active file.
- Reserved keywords that cannot be used as variable names are

ALL AND BY EQ GE GT LE
LT NE NOT OR TO WITH

System Variables

SPSS/PC provides three system variables that are included as part of each case in the active file. The three variables are $DATE, $CASENUM, and $WEIGHT.

- System variables begins with a dollar sign ($).
- System variables cannot be named on a DATA LIST, GET, or IMPORT commands.
- System variables cannot be named as target variables on COUNT, COMPUTE, and IF commands.
- $DATE and $CASENUM are established when data are read with a DATA LIST, IMPORT, or GET command.
- The value of $DATE is the same for each case.
- The value of $CASENUM is the sequence number of each case as it is initially read by SPSS/PC.
- The value of $CASENUM remains unchanged, even after SORT and SELECT IF commands, so you can always identify the original case number.
- The initial value of $WEIGHT for each case is 1.00.
- The value of $WEIGHT is changed using the WEIGHT transformation.
- When a file is written using WRITE, SAVE, or EXPORT, the current values of each system variable are written to the file.

Variable Format

Values are stored internally and displayed or printed according to a specific format. This format is used to print values in procedures and to write data to other files. The format specification *does not* affect the precision of data values stored in memory. You can use the DISPLAY command specifying

VARIABLES=ALL to see the format of each of your variables.

Variable formats have two components, the variable type and the variable width. Variables can be one of two types, numeric or string. Numeric variables can contain numbers, decimal points, and optional leading plus or minus signs. String variable values can contain numbers, letters, and punctuation characters (see Strings).

The format of a variable is defined on the DATA LIST command or is assigned by SPSS/PC on an IF, COUNT, or COMPUTE command. Numeric variable formats can be changed using the FORMATS command.

Variable Type

- By default, the DATA LIST command assumes variables are numeric. String variables are indicated by the A format specification.
- Variables created by COMPUTE or IF commands are assigned a format type based on specifications in the assignment expression (see COMPUTE and IF).

Variable Width

- The width determines how the values are printed.
- Variables defined with DATA LIST using keyword FIXED use the column specifications to calculate the maximum width of data values. Implied decimal places add one column to the width required to print the value.
- Numeric variables read with DATA LIST FREE are assigned a print width of eight characters.
- String variables read with DATA LIST FREE are assigned a width of eight characters. If the actual width is greater than eight columns, you must specify the maximum number of characters after the format specification.
- Numeric variables created with COMPUTE or IF are assigned a width of eight characters, including a decimal point and two decimal digits.
- String variables created with COMPUTE or IF are assigned a width based on the assignment expression. When the assignment expression creates a variable by equating it to an existing variable, the width of the new variable is equal to the width of the existing variable. When the assignment expression creates a new string variable by equating it to a string constant, the width of the new string variable has a width equal to the initial specification of the string constant.

Format Specifications

The FORMATS command allows you to change the print and write width, print additional decimal values, and add a dollar sign and commas to values of numeric variables.

- You cannot change the format of string variables.
- You cannot change the variable type from string to numeric or vice versa.
- The FORMATS command changes only the print and write formats, not the internal representation, of a variable.
- If you do not allow enough room to print the values of a variable, SPSS/PC prints the value without decimal values, commas, or dollar signs. When the value cannot be reasonably represented in the width provided, SPSS/PC uses scientific notation or prints asterisks (**) in the available space.

Transformation Expressions

Transformation expressions are used in COMPUTE, IF, PROCESS IF, and SELECT IF commands. The following sections describe the different types of operators and functions that can be used in transformation expressions.

Arithmetic Operators

Arithmetic operators are used with numeric variables in expressions on COMPUTE, IF, and SELECT IF commands. String variables and constants cannot be used with arithmetic operators. The arithmetic operators are

Addition	Subtraction
Multiplication	Division
Exponentiation	

Syntax

- No two operators can appear consecutively.
- Arithmetic operators cannot be entered before or after relational and logical operators (see Logical Expressions).
- Blanks (not commas) can be inserted before and after an operator to improve readability.

Operations

- Arithmetic operators are executed after functions (see Functions).
- The order of operations is exponentiation, then multiplication and division, and then addition and subtraction.
- Operators at the same level are executed from left to right.
- Use parentheses to override the order of operation. Execution begins with the innermost set of parentheses and progresses out.
- Addition and subtraction always propagate missing values. If any variables or values added or subtracted are missing, the result is system-missing.
- If any variables or values multiplied are missing, the result is system-missing. The exception is when a missing value is multiplied by 0. In this case, the result is 0.
- If any variables or values divided are missing, the result is system-missing. The exception is when 0 is divided by a missing value. In this case, the result is 0.
- Any value or variable divided by 0 is missing.
- If any values raised to a power are missing, the result is system-missing. The exceptions are when a missing value is raised to the 0 power (the result is 1) and when 0 is raised to a missing power (the result is 0).
- A negative number raised to a noninteger power is system-missing.

Functions

Functions are used on COMPUTE, IF, and SELECT IF command expressions. All functions are available for use with numeric variables. Only the LAG function can be used with short string variables. For examples using functions, see COMPUTE, IF, and SELECT IF commands.

The expression that is transformed by a function is called the ***argument***. Most functions have a variable name or a list of variable names as arguments. Arguments are always enclosed in parentheses. For example, to generate the square root of variable X, specify X as the argument to the SQRT function, as in SQRT(X).

Arguments can include arithmetic and exponential operators and numeric constants to form a complex expression. The expression VARA+VARB, enclosed in parentheses, forms an argument that can be used by most functions. You can use sets of parentheses to form complex expressions used as arguments. For functions that take multiple arguments, such as a list of of variables, each argument is separated by a comma within the parentheses.

- By default, functions in expressions are evaluated before arithmetic, relational, and logical operators. The default order of evaluation is first numeric functions, then exponentiation, then arithmetic operators, then relational operators, and then logical operators.
- Functions at the same level are executed from left to right.
- Use parentheses to override the default order of operation. Execution begins with the innermost set of parentheses and progresses out.

Numeric Functions

- Numeric functions always return numbers (or the system-missing value whenever the result is indeterminate).
- All numeric functions take one argument enclosed in parentheses.

ABS(arg) *Absolute value.* ABS(−4.7) is 4.7; ABS(4.7) is 4.7. The argument can be an expression or a variable name. Returns missing if the argument is missing.

RND(arg) *Round the absolute value to an integer (and reaffix the sign).* RND(−4.7) is −5. The argument can be an expression or a variable name. Returns missing if the argument is missing.

TRUNC(arg) *Truncate to an integer.* TRUNC(−4.7) is −4. The argument can be an expression or a variable name. Returns missing if the argument is missing.

MOD10(arg) *Remainder (modulus) of the argument divided by 10.* MOD(198) is 8. The argument can be an expression or a variable name. Returns 0 if the argument is 0 and missing if the argument is missing.

SQRT(arg) *Square root.* The argument can be an expression or a variable name. Returns missing if the argument is negative or missing.

EXP(arg) *Exponential. e* is raised to the power of the argument. The argument can be an expression or a variable name. Returns missing if the argument is missing or if the argument produces a result too large to be represented.

LG10(arg) *Base 10 logarithm.* The argument can be an expression or a variable name. Returns missing if the argument is negative, 0, or missing.

LN(arg) *Natural or Naperian logarithm (base e).* The argument can be an expression or a variable name and must be positive and greater than 0. Returns missing if the argument is missing.

ARTAN(arg) *Arctangent* (alias ATAN). The result is given in radians. Returns missing if the argument is missing. Returns missing if the absolute value of the argument exceeds 1.

SIN(arg) *Sine.* The argument must be specified in radians. Returns missing if the argument is missing.

COS(arg) *Cosine.* The argument must be specified in radians. Returns missing if the argument is missing.

Missing-Value Functions

- Each of the missing-value functions takes on variable name enclosed in parentheses as an argument.

VALUE(arg) *Include all values except system-missing.* The argument must be a single variable name. The VALUE function should be used with a relational operator in IF and SELECT IF logical expressions.

MISSING(arg) *Return 1 (true) if the value is missing and 0 (false) otherwise.* The argument must be a single variable name.

SYSMIS(arg) *Return 1 (true) if the value is missing and 0 (false) otherwise.* The argument must be a single variable name.

Cross-Case Function

LAG(arg) *The value of the variable one case before.* The argument must be a numeric variable. LAG(GNP) returns the value of GNP for the case before the current one. If you are selecting cases from a file, LAG returns the value for the case previously selected.

Random-Number Functions

- The seed value used for random number functions can be changed using the SEED specification on the SET command.

UNIFORM(arg) *A uniform pseudo-random number.* The random number is uniformly distributed with values varying between 0 and the value of the argument. The argument must be a real numeric value.

NORMAL(arg) *A normal pseudo-random number.* The random number is randomly distributed with a mean of 0 and a standard deviation equal to the argument. The argument must be a positive nonzero numeric value.

Date Function

YRMODA(arg list) *Convert year, month, and day to a day number.* The number returned is the number of days since October 15, 1582 (day 1 of the Gregorian calendar).

- Arguments can be variables, constants, or any other type of numeric expression but must yield integers.

- Year, month, and day must be specified in that order.
- The first argument can be any year from 1582 to 47516.
- If the first argument yields a number between 00 and 99, 1900 through 1999 is assumed.
- The month can range from 1 through 13. Month 13 yields the last day of the year—as in YRMODA(YEAR,13,0)—or the first month of the next—as in YRMODA(YEAR,13,DAY).
- The day can range from 0 through 31. Day 0 is the last day of the previous month: YRMODA(YEAR,MONTH + 1,0) is the last day of MONTH.
- Returns missing if any of the three arguments is missing.
- Returns missing if the arguments do not form a valid date.

Logical Expressions

A logical expression is an expression that can be evaluated as true, false, or missing, based upon conditions found in the data. Logical expressions can be simple logical relations among variables, or they can be complex logical tests involving variables, constants, functions, relational operators, and logical operators. Logical expressions can be used on the IF and SELECT IF commands. Not all logical epressions can be used with PROCESS IF.

Relational operators used in expressions are

EQ or =	Equal to	NE or ~= or <>	Not equal to
LT or <	Less than	LE or <=	Less than or equal to
GT or >	Greater than	GE or >=	Greater than or equal to

Logical operators are

AND or & OR or | NOT or ~

Syntax

- Parentheses enclosing a logical expression are required.
- All relational operators can be used with logical expressions on IF and SELECT IF.
- Only EQ (and =) can be used on PROCESS IF.
- No logical operators can be used on PROCESS IF.
- Blanks (not commas) must separate the relational and logical operators from the expressions.
- String values must be enclosed in apostrophes or quotation marks.
- Only strings of the same length are compared using EQ or NE.
- The operators EQ and NE must compare one string variable to another or one numeric variable to another.

Operations

- In expressions, functions and arithmetic operators are executed before relational and logical operators.
- Relational operators are executed before logical operators.
- Logical NOT is executed first, then AND, and then OR.
- Operators at the same level are executed from left to right.
- Use parentheses to override the order of operation. Execution begins with the innermost set of parentheses and progresses out.

Limitations

- The complexity of a logical expression is limited by available memory.

AND and OR

- AND returns true if both expressions are true.
- OR returns true if either expression is true.

NOT

- NOT reverses the outcome of the expression.
- NOT affects only the expression that immediately follows (unless controlled by parentheses.)

Logical Outcomes

- Logic is indeterminate and the outcome is missing if the expression on either side of the relational operator is missing.

- When relations are joined by logical operators AND and OR, if any one of the relations can be determined, SPSS/PC tries to return true or false according to the logical outcomes:

Expression	Outcome	Expression	Outcome
true AND true	= true	true OR true	= true
true AND false	= false	true OR false	= true
false AND false	= false	false OR false	= false
true AND missing	= missing	true OR missing	= true*
missing AND missing	= missing	missing OR missing	= missing
false AND missing	= false*	false OR missing	= missing

*evaluated with incomplete information

ANOVA

```
ANOVA [VARIABLES=] varlist BY varlist(min,max) [WITH varlist]
      [/[VARIABLES=] varlist ...]

      [/OPTIONS=option numbers]

      [/STATISTICS={statistic numbers}]
                   {ALL              }
```

Options:

1 Include user-missing values
2 Suppress labels
3 Suppress all interaction terms
4 Suppress three-way terms
5 Suppress four-way terms
6 Suppress five-way terms
7 Covariates with main effects
8 Covariates after main effects
9 Regression approach
10 Hierarchical approach
11 Narrow format

Statistics:

1 MCA table
2 Unstandardized regression coefficients for covariates
3 Display cell means and counts

Example:

```
ANOVA VARIABLES=YVAR1,YVAR2 BY XVAR(1,3) ZVAR1,ZVAR2(1,2)
/OPTIONS=4
/STATISTICS=3.
```

Overview

Procedure ANOVA performs analysis of variance for factorial designs. The default is the full factorial model if there are five or fewer factors. Analysis of variance tests the hypothesis that the group means of the dependent variable are equal. The dependent variable is interval level, and one or more categorical variables define the groups. These categorical variables are termed *factors*. ANOVA also allows you to include continuous explanatory variables, termed *covariates*. Other SPSS/PC procedures that perform analysis of variance are ONEWAY and MEANS.

Defaults

By default, the model includes all interaction terms up to five-way interactions. In the default model, the sums of squares are decomposed using the classical experimental approach, in which covariates, main effects, and ascending orders of interaction are assessed separately in that order. The default display includes an analysis of variance table with variable labels, sums of squares, degrees of freedom, mean square, *F*, probability of *F* for each effect, and a count of valid and missing cases. By default, a case that has a missing value for any variable in an analysis list is omitted from the analysis.

Tailoring

Statistical Display. You can choose among three methods for controlling the order in which covariates and main effects are assessed. You can also select among three methods for decomposing the sums of squares, and you can pool interaction effects in the error term. In addition, you can request a labeled multiple classification analysis (MCA) table, a table of means of the dependent variable within groups formed by the factors, and unstandardized regression coefficients for the covariates.

Display Format. You can suppress the display of variable labels on all tables and the display of value labels on the MCA table.

Missing Values. You can include cases with user-missing values in the analysis.

Syntax

- The minimum specification is a single VARIABLES subcommand with an analysis list. The actual keyword VARIABLES may be omitted.
- The minimum analysis list specifies a list of dependent variables, the keyword BY, a list of factor variables, and the minimum and maximum integer values of the factors in parentheses.
- Subcommands can be specified in any order and must be separated by slashes.

Operations

- ANOVA is a procedure and causes the data to be read.
- A separate analysis of variance is performed for each dependent variable in an analysis list, using the same factors and covariates.
- With the exception of cell means and counts (Statistic 3), the output is always displayed in narrow format, regardless of the width defined on SET.

Limitations

- Maximum 5 VARIABLES subcommands.
- Maximum 1 each STATISTICS and OPTIONS subcommands.
- Maximum 5 dependent variables per VARIABLES subcommand.
- Maximum 10 factors per VARIABLES subcommand.
- Maximum 10 covariates per VARIABLES subcommand.
- Maximum 5 interaction levels.

  Maximum 25 value labels per variable displayed in the MCA table.
- The combined number of categories for all factors in an analysis list plus the number of covariates must be less than the sample size.
- Memory requirements for ANOVA are roughly proportional to the square of the product of the number of values for each independent variable.
- Both the number of categories in each factor and the number of interaction terms included in the model will determine the amount of workspace required.

Example

```
ANOVA VARIABLES=YVAR1,YVAR2 BY XVAR(1,3) ZVAR1,ZVAR2(1,2)
/OPTIONS=4
/STATISTICS=3.
```

- The VARIABLES subcommand specifies two three-way analyses of variance: YVAR1 by XVAR, ZVAR1, and ZVAR2; and YVAR2 by XVAR, ZVAR1, and ZVAR2.
- Variables ZVAR1 and ZVAR2 both have the values 1 and 2 included in the analysis.
- The OPTIONS subcommand pools all three-way interaction terms into the error sum of squares.
- The STATISTICS subcommand requests the display of a table of cell means of YVAR1 and YVAR2 within the combined categories of XVAR, ZVAR1, and ZVAR2.

VARIABLES Subcommand

The VARIABLES subcommand specifies the analysis list. The actual keyword VARIABLES may be omitted.

- Variables named before the BY keyword are dependent variables.
- Variables named after the BY in the analysis list are factor (independent) variables. All factors are used simultaneously in the analysis of variance requested by that analysis list.
- Every factor variable must have a value range indicating its minimum and maximum values. The values must be separated by a space or comma and enclosed in parentheses.

- Variables named after the keyword WITH are covariates.
- Each analysis list can include only one BY and one WITH keyword.
- Factor variables must be integer. Noninteger values for factors are truncated.
- Cases with values outside the range specified for a factor are excluded from the analysis.
- If two or more factors have the same value range, you can specify the value range once following the last factor to which it applies.
- You can specify a single minimum and maximum value range that encompasses the ranges of all factors in the list. However, this may reduce performance and cause memory problems if the specified range is larger than the actual range.
- You can specify multiple VARIABLES subcommands on a single ANOVA command. The slash between subcommands is required; the VARIABLES keyword is not.

Suppressing Interaction Effects

By default, all interaction effects up to and including fifth-order interaction effects are tested. You can suppress any of these higher-order interactions and pool them into the error (residual) sums of squares by specifying Option 3, 4, 5, or 6. When you specify any of these options, cell means corresponding to suppressed interaction terms are not displayed.

Option 3 *Suppress all interaction terms.*

Option 4 *Suppress three-way and higher-order interaction terms.*

Option 5 *Suppress four-way and higher-order interaction terms.*

Option 6 *Suppress five-way and higher-order interaction terms.*

Specifying Order of Entry of Covariates

By default, covariates are assessed first and main effects are assessed after adjusting for the covariates. To change this order, use Option 7 or 8.

Option 7 *Process covariates concurrently with main effects for factors.*

Option 8 *Process covariates after main effects for factors.* Option 8 is ignored when Option 9 is in effect.

Decomposing Sums of Squares

By default, each type of effect is assessed separately in the following order (unless Option 7 or 8 has been specified):

- Effects of covariates.
- Main effects of factors.
- Two-way interaction effects.
- Three-way interaction effects.
- Four-way interaction effects.
- Five-way interaction effects.

To change this order, specify Option 9 or 10.

Option 9 *Regression approach.* All effects are assessed simultaneously, with each effect adjusted for all other effects in the model. Option 9 overrides Options 7 and 8. Statistics 1 and 3 are not available with Option 9.

Option 10 *Hierarchical approach.* Factor main effects and covariate effects are assessed hierarchically. Factor main effects are adjusted only for factor main effects already assessed, and covariate effects are adjusted only for covariates already assessed. Factors are assessed in the order they are listed in the analysis list on the ANOVA command.

The table below shows how effects would be assessed under Options 9, 10 and the default for the following command:

```
ANOVA VARIABLES=Y BY A B C (0 3).
```

Terms adjusted for under each option

Effect	Default	Option 9	Option 10
A	B,C	ALL OTHERS	NONE
B	A,C	ALL OTHERS	A
C	A,B	ALL OTHERS	A,B
AB	A,B,C,AC,BC	ALL OTHERS	A,B,C,AC,BC
AC	A,B,C,AB,BC	ALL OTHERS	A,B,C,AB,BC
BC	A,B,C,AB,AC	ALL OTHERS	A,B,C,AB,AC
ABC	A,B,C,AB,AC,BC	ALL OTHERS	A,B,C,AB,AC,BC

Summary of Analysis Methods

The table below describes the results obtained with various combinations of methods for controlling entry of covariates and decomposing the sums of squares.

	Assessments between types of effects	Assessments within the same type of effect
Default	**Covariates** *then* **Factors** *then* **Interactions**	**Covariates:** Adjust for all other covariates **Factors:** Adjust for covariates and all other factors **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders
Option 7	**Factors** **Covariates** concurrently *then* **Interactions**	**Covariates:** Adjust for factors and all other covariates **Factors:** Adjust for covariates and all other factors **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders
Option 8	**Factors** *then* **Covariates** *then* **Interactions**	**Factors:** Adjust for all other factors **Covariates:** Adjust for factors and all other covariates **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders
Option 9	**Covariates** **Factors** **Interactions** simultaneously	**Covariates:** Adjust for factors, interactions, and all other covariates **Factors:** Adjust for covariates, interactions, and all other factors **Interactions:** Adjust for covariates, factors, and all other interactions
Option 10	**Covariates** *then* **Factors** *then* **Interactions**	**Covariates:** Adjust for preceding covariates **Factors:** Adjust for covariates and preceding factors **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders
Options 7 and 10	**Factors** **Covariates** concurrently *then* **Interactions**	**Factors:** Adjust only for preceding factors **Covariates:** Adjust for factors and preceding covariates **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders
Options 8 and 10	**Factors** *then* **Covariates** *then* **Interactions**	**Factors:** Adjust only for preceding factors **Covariates:** Adjust for factors and preceding covariates **Interactions:** Adjust for covariates, factors, and all other interactions of the same and lower orders

Multiple Classification Analysis

Multiple classification analysis is useful for displaying the results of analysis of variance when there are no significant interaction effects (see Andrews et al., 1973).

- For each category of each factor, the MCA table presents the unadjusted mean of the dependent variable expressed as a deviation from the grand mean; the deviation from the grand mean of the category mean adjusted for other factors; and the deviation from the grand mean of the category mean adjusted for both factors and covariates.
- For each factor, the complete MCA display displays the correlation ratio (eta) with the unadjusted deviations (the square of eta indicates the proportion of variance explained by all categories of the factor); a partial beta equivalent to the standardized partial regression coefficient that would be obtained by assigning the unadjusted deviations to each factor category and regressing the dependent variable on the resulting variables; and the parallel partial betas from a regression that includes covariates in addition to the factors, and the multiple R and R^2 from this regression.

To obtain an MCA table, specify Statistic 1 on the STATISTICS subcommand.

Statistic 1 *Multiple classification analysis.* The MCA display is affected by the form of analysis specified by Options 7 through 10 or their defaults. When covariates are specified, a complete MCA table can be obtained only in conjunction with Option 8, Options 8 and 10, or Options 7 and 10. With a model in which factors are not processed first, effects adjusted only for factors do not appear. The MCA table cannot be produced when Option 9 is in effect.

Statistical Display

You can request the following optional statistics on the STATISTICS subcommand:

Statistic 2 *Display unstandardized regression coefficients for covariates.* The regression coefficients are computed when the covariates are entered into the equation. Thus, their values depend on the design you specify with Options 7 through 10 or the defaults.

Statistic 3 *Display cell means and counts for the dependent variable.* A table is created by crossing all factors. Marginal means and counts as well as cell means and counts are displayed. With Options 3 through 6, cells corresponding to the suppressed interaction terms are not displayed. Statistic 3 is not available when Option 9 is in effect.

ALL *Display all statistics.* Includes display produced by Statistics 2 and 3.

Display Format

In the default display, the ANOVA table is labeled with the variable labels and the MCA table includes value labels. In addition, the table of cell means and counts (Statistic 3) uses the available width. To change these defaults, specify the following on the OPTIONS subcommand:

Option 2 *Suppress variable and value labels.*

Option 11 *Narrow format.* Restricts the display produced by Statistic 3 to narrow width regardless of the width defined on SET.

Missing Values

By default, a case that has a missing value for any variable named on a VARIABLES subcommand is omitted from all analyses requested by that subcommand. You can change the treatment of missing values by specifying the following on the OPTIONS subcommand:

Option 1 *Include user-missing values.* User-missing value specifcations are ignored. A case with a system-missing value for any variable named on a VARIABLES subcommand is omitted from the analyses requested by that subcommand.

References

Andrews, F., J. Morgan, J. Sonquist, and L. Klein. *Multiple classification analysis.* 2d ed. Ann Arbor: University of Michigan, 1973.

BEGIN DATA—END DATA

```
BEGIN DATA
lines of data
END DATA
```

Example:

```
BEGIN DATA.
1  3424  274 ABU DHABI 2
3 39932   86 AMSTERDAM 4
3  8889  232 ATHENS
2  3424  294 BOGOTA    3
END DATA.
```

Overview

The BEGIN DATA command signals the beginning of data lines in a command file, and the END DATA command signals the end of data lines. Both BEGIN DATA and END DATA must be used when data is part of an SPSS/PC command file (inline data). BEGIN DATA and END DATA are also used for inline matrix data or matrix materials.

Syntax

- BEGIN DATA, the data, and END DATA must appear before the first SPSS/PC procedure.
- Both BEGIN DATA and END DATA include the command terminator.
- Each line of data must *not* have the command terminator.
- The BEGIN DATA command must be entered immediately before the first line of inline data.
- The END DATA command must be entered immediately after the last line of data, beginning in the column 1.
- Only a single space is allowed between the words END and DATA.
- The END DATA command cannot be truncated.
- Procedures and additional transformations can follow the END DATA command.

Operations

- When SPSS/PC encounters BEGIN DATA, it begins to read and process data on the next input line. All preceding transformation commands are processed as a file is built for use in SPSS/PC procedures.
- SPSS/PC continues to evaluate input lines as data until it encounters END DATA, at which point it begins evaluating input lines as SPSS/PC commands.
- No other SPSS/PC commands are recognized between BEGIN DATA and END DATA.
- You can use a file that contains BEGIN DATA, data lines, and END DATA with the INCLUDE command, provided you omit the FILE specification on your DATA LIST command (see INCLUDE).

Limitations

- BEGIN DATA and END DATA can occur only once in an SPSS/PC session.

Example

```
DATA LIST /XVAR 1 YVAR ZVAR 3-12 CVAR 14-22(A) JVAR 24.
BEGIN DATA.
1  3424  274 ABU DHABI 2
3 39932   86 AMSTERDAM 4
3  8889  232 ATHENS
2  3424  294 BOGOTA    3
4 11323  332 HONG KONG 3
3   323  232 MANILA    1
4  3234  899 CHICAGO   4
1 78998 2344 VIENNA    3
2  8870  983 ZURICH    5
END DATA.
MEANS XVAR BY JVAR.
```

- The DATA LIST command defines the names and column locations of the variables. The FILE subcommand is omitted because the data are inline.
- The BEGIN DATA command contains the command terminator.
- There are nine cases in the inline data.
- Each line of data is completed by pressing the enter key.
- The END DATA command signals the end of lines of data.