Appendix A
BASIC RULES OF TSP

This section summarizes the basic rules of the TSP language. A table at the end lists the complete TSP character set and each character's interpretation or legal use.

1. Rules for composing TSP names:

    a. Every name must begin with a letter, _ # % or @ (exceptions: 2SLS and 3SLS commands).

    b. Subsequent characters in a name may be letters or digits.

    c. The maximum number of characters permitted in a name is 64 (in versions prior to 4.4 it was 8).

2. Rules for composing text strings:

    a. A text string must be enclosed by matching pair of quotes (" or ').

    b. Quotes are allowed in a string when they are of a different type from the enclosing quotes, i.e. "Can't" or '"sometimes"' (interpreted as Can't and "sometimes").

    c. A semicolon (;) is not allowed in a text string.

3. Rules for composing numbers:

    a. Every number must begin with a ., +, -, or a digit.

    b. No spaces may appear within a number.

    c. One decimal point may appear.

    d. One E or D may appear followed immediately by a one or two digit number with or without a sign. This is interpreted as a power of 10 to multiply the first number. Example: 1E2 = 100.

    e. With free-format LOAD or READ commands, a . is interpreted as a missing value, and a repeat count with a * may be specified. For example: 3*0 is treated as 0 0 0 , and 53.100 is treated as 53, missing, 100.

4. Rules for composing algebraic formulas:

    a. In general, TSP rules for formulas are similar to Fortran or other scientific programming languages.

    b. A lag is indicated by putting an integer or a name in parentheses after a series name. The integer is negative for lags and positive for leads. A + sign is not necessary for leads. If the lag or lead is a name, it must have no more than four characters.

    c. A series may have a single numeric or variable subscript (or lag/lead). A matrix may have a single or double subscript (numeric or variable). See the SET command for detailed rules and examples.

    d. Arithmetic operators are

+	add	*	multiply	**	raise to the power
-	subtract	/	divide		

e. Functions are (for matrix functions, see the MATRIX command)

LOG()	Natural logarithm
EXP()	Exponential function
ABS()	Absolute value
LOG10()	Log base 10
SQRT()	Square root
SIN()	Sine (argument in radians)
COS()	Cosine (argument in radians)
TAN()	Tangent (argument in radians)
ATAN()	Arctangent (answer in radians)
NORM()	Standard normal density
CNORM()	Standard normal cumulative distribution function
CNORMI()	Inverse of the standard normal cumulative distribution function
LNORM()	Log of normal density
LCNORM()	Log of cumulative normal
DLCNORM()	Derivative of LCNORM = inverse Mills ratio
GAMFN()	Gamma function (not Gamma density)
LGAMFN ()	Log of Gamma function
DLGAMFN()	Derivative of LGAMFN = DIGAMMA()
TRIGAMMA()	Derivative of DIGAMMA() [non-differentiable]
FACT()	Factorial: FACT(X) = X! = GAMFN(X+1)
LFACT()	Log of factorial
SIGN()	Sign: -1 for X<0, 0 for X=0, 1 for X>0 [deriv=0]
POS()	Positive: POS(X) = max(0,X) Note: "min(A,B)" = B - POS(B-A), "max(A,B)" = A + POS(B-A)
MISS()	Missing: 1 for X missing, 0 otherwise [non-differentiable]
INT()	Integer: truncate (round towards 0) [non-differentiable]
CEIL()	Ceiling: round away from 0 [non-differentiable]
ROUND()	Round to nearest integer (.5 rounds to 1) [non-differentiable]

f. Relational operators are

= gives the value 1 when the variables on the left and on the right are equal; otherwise it is zero. (.EQ.)
^= gives the value 1 when the variables on the left and on the right are not equal; otherwise it is zero. (.NE.)
< gives the value 1 when the variable on the left is less than the variable on the right; otherwise it is zero. (.LT.)
> gives the value 1 when the variable on the left is greater than the variable on the right; otherwise it is zero. (.GT.)
<= gives the value 1 when the variable on the left is less than or equal to the variable on the right; otherwise it is zero. (.LE.)
>= gives the value 1 when the variable on the left is greater than or equal to the variable on the right; otherwise it is zero. (.GE.)

g. Logical operators are

& gives the value 1 when both the variable on the left and on the right are positive. (.AND.)
| gives the value 1 when either the variable on the left or the variable on the right is positive. (.OR.)
^ gives the value 1 when the variable on the right is negative or zero ('~' also works). (.NOT.)

Note: the .OP. form of the relational and logical operators given in parentheses is the alternative to the symbolic notation (but it is not valid in nested DOT loops).

h. As many parentheses as necessary may be used to indicate the order of evaluation of a formula. The special parentheses [] and {} are treated as ().

i. In the absence of parentheses, evaluation proceeds from left to right in the following order:

1. Functions
2. Exponentiation (**)
3. Multiplication and division
4. Addition, subtraction, and negation (unary -)
5. Relational operators
6. .NOT. (^)
7. .AND. (&) and .OR. (|)

5. Rules for composing TSP statements:

a. Every statement begins with a command name.

exceptions: X=Y; [implicit GENR],
X(I)=Y(I); [implicit SET]
100 <statement>; [statement label for GOTO]

The command name may be abbreviated, as long as it is uniquely identified.

b. Many statements can have options specified in parentheses after the command name. Option names may be abbreviated, like command names. There are three kinds of options:

1. Boolean options, either on or off. On is specified by the name of the option, as in PRINT, and off is specified by the option name with NO in front of it, as in NOPRINT.

2. Options of the form *option name = option value*. The value may be the name of a variable, a numerical value, or just a keyword, depending on the context.

3. Options which give lists of variables, and are of the form *option name* = (*list of variables*). Note that the parentheses are required, unless the list contains only one name, or the list is a listname.

c. A few statements can have an algebraic formula: GENR, SET, SMPLIF, SELECT, FRML, IDENT, IF, GOTO.

d. Most statements use one or more series names, separated by commas or spaces. These series names may include lags. An implicit list (such as X1-X5) can be used directly in a statement without making an intermediate listname. See the LIST command for a complete description of implicit list syntax.

e. The end of a statement is marked by a semicolon (;) or dollar sign ($).

Character Set for TSP

Character	Symbol	Use
letter	A to Z,_#%@	Parts of names. Lowercase letters are allowed on most computers; they are treated like uppercase letters. # % cannot be used for matrix names.
digit	0 to 9	Parts of numbers or names.
decimal point	.	Marks the decimal point in numbers; sets off logical operators; specifies string substitution in the DOT procedure.
comma	,	Separates the words in a list; spaces may be used, but commas are often preferred for clarity.
colon	:	Part of date.
semicolon	;	Marks the end of a statement.
dollar sign	$	Equivalent to ; . Semicolon is preferred.
quotation mark	"	Marks the beginning and end of a text string (title or filename); specifies matrix inversion.
apostrophe	'	Marks a text string; specifies matrix transposition.
parentheses	() [] {}	Encloses a list of options or expressions/lags in algebraic formulas.
question mark	?	Delimits the beginning of comments. (Comments are terminated by the end of the input line or logical record.)
plus sign	+	Specifies addition.
minus sign	-	Specifies subtraction, a lag, or a list.
star	*	Specifies multiplication or is part of power (**).
slash	/	Specifies division.
pound sign	#	Matrix Kronecker product (⊗).
percent	%	Matrix Hadamard product (element by element).
equal sign	=	Specifies equality or definition of data; relational operator (.EQ., .NE., .LE.,.GE.).
ampersand	&	Logical operator (.AND.).
vertical bar	|	Logical operator (.OR.).
caret or hat	^	Logical operator (.NOT., .NE.) or power (**), depending on context.
tilde	~	Logical operator (.NOT., .NE.).
less than	<	Relational operator (.LT., .LE.).
greater than	>	Relational operator (.GT., .GE.).
continuation	\	Continuation of a line (interactive).
miscellaneous	!`	Reserved for future use.