

5. Real with Missing Variables

5.1 Introduction

VPLX assumes that each observation included in the analysis has a valid value for each real variable. For each CREATE step in Chapters 2, 3, and 4, a count of observations in the data set was saved and used as the denominator in computing the mean of any real variable. Of course, there are numerous situations in which one wants to limit the domain of valid observations and to use a different denominator in estimating means:

- A question may be only asked of a specific group of respondents. For example, only those respondents indicating that they were unemployed or seeking new or additional employment may be asked what they did to find a job.
- Analytic interest may be restricted to a subset of interest, such as the poverty rate for children age 0-4.
- Data may be subject to nonresponse. Often, imputations are provided for missing data. If no imputations have been made, one may be interested in the rates or totals for respondents. Even if imputed values are present, one may wish to compare rates for respondents to rates including the imputed values.

VPLX includes several alternative approaches to restricting variables to a subset of observations:

- This chapter introduces *real with missing* variables. These variables keep track of a weighted total and number of valid observations, enabling computation of means for a specified subset of observations.
- Class variables, in Chapter 7, may be used to cross-classify other variables. For example, if rates for ages 16-19 and ages 20+ were of interest for a group of variables, then it is generally far more effective in VPLX to use age as a class variable than to create a series of real with missing variables.
- Chapter 8 describes how SELECT may be used with BLOCK statements to restrict the universe of groups of variables. This is a refined use of SELECT, compared to its description in Chapter 3, where SELECT affects entire observations.

5.2

In general, all three approaches, when properly applied, yield the same estimates and estimated variances. Thus, there is not a single right choice in one situation, but in some situations one approach may be far more convenient than another.

The material in this chapter assumes a reading of both Chapters 3 and 4. In general, subsequent chapters do not require the material in this one, and the chapter may be skipped on a first reading.

There are three ways to establish real with missing variables in the CREATE step:

- With a MISSING statement.
- Through ADD_MS, SUBTRACT_MS, MULTIPLY_MS, DIVIDE_MS, special forms of ADD, SUBTRACT, MULTIPLY, and DIVIDE that produce real with missing variables.
- Through LINK_MISSING to another file.

The first of these is the most common of the three, so the chapter is organized to allow a partial reading.

Summary of this chapter:

- Section 5.2 describes the MISSING statement.
- Section 5.3 discusses the treatment of real with missing variables by the DISPLAY step.
- Section 5.4 provides an example.
- Section 5.5 describes how real with missing variables are treated in the CREATE step with respect to features such as COPY, IF blocks, *etc.*
- Section 5.6 covers ADD_MS, SUBTRACT_MS, MULTIPLY_MS, and DIVIDE_MS.
- Section 5.7 provides an example.
- Section 5.8 presents the LINK_MISSING statement as an alternative to LINK.
- Section 5.9 illustrates LINK_MISSING.

5.2 The MISSING Statement

5.2.1 General Forms. The simplest form of the MISSING statement changes one or more real variables into real with missing variables and treats them as missing if the values fall into a given range. The syntax is:

```
MISSING  vlist1  ( range )
```

The variable or variables in *vlist1*, which must have been previously defined, will be classified as missing if their values fall into the given range. Section 3.6.3 describes specification of ranges.

It is also possible to create new variables with the use of INTO:

```
MISSING  vlist1 INTO vlist2 ( range )
```

In this form, the values of variables in *vlist1* will be checked against the range in creating *vlist2*, but the variables in *vlist1* will remain unchanged unless they also appear in *vlist2*. The variables in *vlist1* must be previously defined, but *vlist2* may contain previously undefined variables.

It is also possible to classify a variable as missing on the basis of the value of another variable. Different possible versions take the form:

```
MISSING  vlist  FOR vname ( range )
```

```
MISSING  vlist1 FOR vlist2  ( range )
```

```
MISSING  vlist1 INTO vlist2 FOR vname ( range )
```

```
MISSING  vlist1 INTO vlist2 FOR vlist3 ( range )
```

In the first of these, all variables in *vlist* are classified according to the value of *vname*. In the second, the checking is against the corresponding variable in *vlist2*, where both lists must contain the same number of variables, and the results are stored back into *vlist1*. The third and fourth form illustrate how FOR may be combined with INTO. The lengths of all three lists in the last version must agree.

5.2.2 Storage of Real with Missing Variables: Unlike real variables, described in Chapter 3, VPLX keeps track whether the observation is to be included in the analysis separately for each real with missing variable. At the observation level, VPLX keeps track of the value, if it is valid, as well as a flag to indicate whether the observation is to be included. As the information from

5.4

the observation is added into the cumulative totals for the full sample and each replicate, both the weighted sum and weighted number of observations are recorded. Exhibit 5.1 illustrates the storage requirements for real with missing variables.

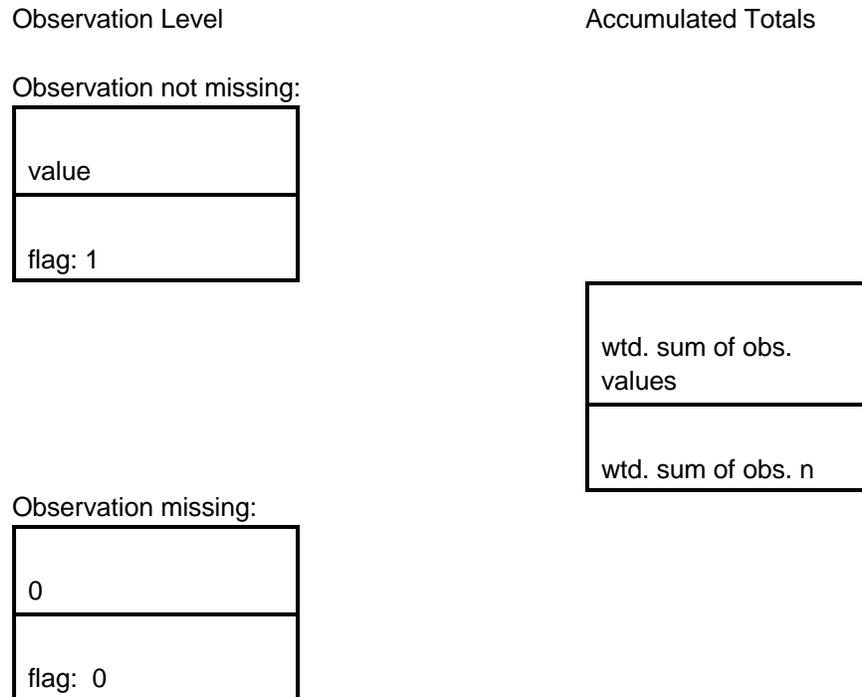


Exhibit 5.1 Representation at the observation and cumulative level for a real with missing variable. The left-hand side shows the observation-level treatment of a real with missing variable. The right-hand side shows the representation of cumulative sums over observations, both for the overall total and for each replicate. At both levels, 2 cells are reserved for each real with missing variable. If the variable is not missing at the observation level, then the first cell holds the value of the variable and the flag in the second cell is set to 1. If the variable becomes missing, then both the first and second cells are set to 0. At the end of processing observation level data, the contents of the observation-level representation can be multiplied by the weight, if any, and summed directly into the accumulated totals.

5.3 Treatment of Real with Missing Variables in the DISPLAY STEP

Generally, real with missing variables are treated in the DISPLAY step in essentially the same manner as real variables. The function N returns the valid number of observations for the real with missing variable. The following standard functions are available for real with missing variables:

MEAN or MEANS - This function is the default for real with missing variables. The numerator is the weighted total of valid observations and the denominator is the weighted count of valid cases. (1).

TOTAL or **TOTALS** - If this function is applied to a real with missing variable, then it shows the weighted sum of the variable over included observations.

N - This function provides the weighted count of included observations for a real with missing variable.

5.4 An Example of the Effect of MISSING in the CREATE Step.

The following example is based on an elaboration of exam11.crd of Section 3.9.

```

comment  EXAM15

comment  This example modifies EXAM11 and illustrates use of MISSING

create  in = exampl11.dat  out = exampl11.vpl

input   rooms persons cluster tenure

        4 variables are specified

format  (4f2.0)

comment  The input data set contains
5 7 1 2
6 8 2 2
5 2 3 1
4 1 4 2
8 4 5 1
8 2 6 1

comment  Create an individual-level ratio of rooms to persons

missing rooms into rooms_rent for tenure (1,3)

missing rooms into rooms_own  for tenure (2)

print  rooms tenure rooms_rent rooms_own

*** PRINT request    1

(Simple) jackknife replication assumed

Size of block    1 =          8

Total size of tally matrix =    8

Unnamed scratch file opened on unit 13

Unnamed scratch file opened on unit 14

**** End of CREATE specification/beginning of execution

```

5.6

```

PRINT request 1
  rooms          5.000000
  tenure         2.000000
  rooms_rent     5.000000
  rooms_own      (M)
PRINT request 1
  rooms          6.000000
  tenure         2.000000
  rooms_rent     6.000000
  rooms_own      (M)
PRINT request 1
  rooms          5.000000
  tenure         1.000000
  rooms_rent     (M)
  rooms_own      5.000000
PRINT request 1
  rooms          4.000000
  tenure         2.000000
  rooms_rent     4.000000
  rooms_own      (M)
PRINT request 1
  rooms          8.000000
  tenure         1.000000
  rooms_rent     (M)
  rooms_own      8.000000
PRINT request 1
  rooms          8.000000
  tenure         1.000000
  rooms_rent     (M)
  rooms_own      8.000000

```

End of primary input file after obs # 6

display

option ndecimal=2 totals

```

list  rooms persons rooms_rent rooms_own
      n ( rooms persons rooms_rent rooms_own )
      mean (rooms rooms_rent rooms_own )

```

		Estimate	Standard error
rooms	: TOTAL	36.00	4.10
persons	: TOTAL	24.00	7.10
rooms_rent	: TOTAL	15.00	6.88
rooms_own	: TOTAL	21.00	9.77
rooms	: WEIGHTED N	6.00	.00
persons	: WEIGHTED N	6.00	.00

rooms_rent	:	WEIGHTED N	3.00	1.34
rooms_own	:	WEIGHTED N	3.00	1.34
rooms	:	MEAN	6.00	.68
rooms_rent	:	MEAN	5.00	.65
rooms_own	:	MEAN	7.00	1.12

Exhibit 5.2 An example illustrating MISSING in the CREATE step. ROOMS_RENT and ROOMS_OWN are created as real with missing on the basis of TENURE. The number of valid observations is retained for each separately, and is available to the DISPLAY step for display and to compute means.

5.5 Treatment of Real with Missing Variables in the CREATE Step

5.5.1 COPY. If a real with missing variable is copied into a target variable, the target assumes all values and attributes of the real with missing variable. In particular, if the original variable is flagged as missing, the target variable will be as well.

5.5.2 IF Blocks. A real with missing variable may be compared to a range:

```
IF   vname ( range ) THEN
```

If the value is not missing at that point, the value will be compared to the range in the same manner as real variables. If the variable is missing, however, then the condition will be treated as false, even for the global interval (*res*). In general, VPLX will consider the comparison of a variable with a missing value to a range as not satisfied (2)

Handling of real with missing variables within IF blocks is subject to some restrictions. Allowing a variable of the type real to be redefined as real with missing would represent a change of variable type. VPLX enforces a general rule forbidding a variable to change its type within an IF block. Thus, if any of the variables in *vlist1* is real, the following is **not allowed**:

```
MISSING vlist1 ( range )
```

Creation of new real with missing variables with INTO, however, is allowed:

```
MISSING vlist1 INTO vlist2 ( range )
```

As Section 3.6.1 notes, when a real variable is created within an IF block, it takes the value 0 for each case in which it is not assigned a value, that is, when an IF condition is not satisfied. The

5.8

analogous rule is that, if a new real with missing variable is defined within an IF block, then it has the value missing in each case that is not assigned.

5.6 ADD_MS, SUBTRACT_MS, MULTIPLY_MS, DIVIDE_MS

The functions ADD_MS, SUBTRACT_MS, MULTIPLY_MS, and DIVIDE_MS are similar to their counterparts, ADD, SUBTRACT, MULTIPLY, and DIVIDE, described in Section 3.7, but the target variables become, in most cases, classified as real with missing variables.

There is a specific exception: if all of the operands in ADD_MS, SUBTRACT_MS, or MULTIPLY_MS are real, then the target variable remains classified as real and these functions behave identically to ADD, SUBTRACT, and MULTIPLY, respectively. If one or more of the operands is real with missing, however, then the target variable becomes classified as real with missing. Furthermore, for each observation in which any operand is missing, the target will be classified as missing.

DIVIDE_MS always results in a target classified as real with missing. The outcome will be missing for each observation when either operand is missing. It will also be classified as missing if the denominator is 0. (Division by 0 with DIVIDE results in the outcome 0.)

For example:

```
missing a for b (1-high)
add_ms a plus c into d
```

If b is 1 or more, then a becomes missing, as does d. The result d also becomes missing if c is missing.

5.7 An Example of the Effect of ADD_MS and DIVIDE_MS.

The following example modifies exam15.crd.

```
comment EXAM16

comment This example modifies EXAM15 to illustrate the use of
        DIVIDE_MS, ADD_MS etc.

create in = exampl11.dat out = exampl11.vpl

input rooms persons cluster tenure
```

4 variables are specified

```
format      (4f2.0)

comment    The input data set contains
5 7 1 2
6 8 2 2
5 2 3 1
4 1 4 2
8 4 5 1
8 2 6 1

comment    The following operations result in adj_persons = 0 for
           2 of the observations in the data set. Use of this
           denominator illustrates differences between divide and
           divide_ms, and between add and add_ms

constant   2 into c2

subtract   persons minus c2 into adj_persons

divide     rooms by adj_persons into ratio

divide_ms  rooms by adj_persons into ratio_ms

add        persons plus ratio      into  test1

add_ms     persons plus ratio_ms   into  test2

print     rooms persons adj_persons ratio ratio_ms test1 test2

*** PRINT request      1

           (Simple) jackknife replication assumed

           Size of block      1 =                12

           Total size of tally matrix =          12

           Unnamed scratch file opened on unit 13

           Unnamed scratch file opened on unit 14

**** End of CREATE specification/beginning of execution
PRINT request      1
rooms              5.000000
persons            7.000000
adj_persons        5.000000
ratio              1.000000
ratio_ms           1.000000
test1              8.000000
test2              8.000000
PRINT request      1
rooms              6.000000
persons            8.000000
adj_persons        6.000000
ratio              1.000000
ratio_ms           1.000000
```

5.10

```

      test1          9.000000
      test2          9.000000
PRINT request      1
      rooms          5.000000
      persons        2.000000
      adj_persons    .000000
      ratio          .000000
      ratio_ms      (M)
      test1          2.000000
      test2          (M)
PRINT request      1
      rooms          4.000000
      persons        1.000000
      adj_persons    -1.000000
      ratio          -4.000000
      ratio_ms      -4.000000
      test1          -3.000000
      test2          -3.000000
PRINT request      1
      rooms          8.000000
      persons        4.000000
      adj_persons    2.000000
      ratio          4.000000
      ratio_ms      4.000000
      test1          8.000000
      test2          8.000000
PRINT request      1
      rooms          8.000000
      persons        2.000000
      adj_persons    .000000
      ratio          .000000
      ratio_ms      (M)
      test1          2.000000
      test2          (M)

```

End of primary input file after obs # 6

display

option ndecimal=2 totals

```

list      rooms persons adj_persons ratio ratio_ms test1 test2
      n( rooms persons adj_persons ratio ratio_ms test1 test2 )
      mean ( test1 test2 )

```

		Estimate	Standard error
rooms	: TOTAL	36.00	4.10
persons	: TOTAL	24.00	7.10
adj_persons	: TOTAL	12.00	7.10
ratio	: TOTAL	2.00	6.32

ratio_ms	:	TOTAL	2.00	6.32
test1	:	TOTAL	26.00	11.66
test2	:	TOTAL	22.00	12.84
rooms	:	WEIGHTED N	6.00	.00
persons	:	WEIGHTED N	6.00	.00
adj_persons	:	WEIGHTED N	6.00	.00
ratio	:	WEIGHTED N	6.00	.00
ratio_ms	:	WEIGHTED N	4.00	1.26
test1	:	WEIGHTED N	6.00	.00
test2	:	WEIGHTED N	4.00	1.26
test1	:	MEAN	4.33	1.94
test2	:	MEAN	5.50	3.00

Exhibit 5.3 An example illustrating DIVIDE_MS and ADD_MS in the CREATE step. RATIO_MS becomes missing for 2 observations where there is attempted division by 0, whereas RATIO is treated as 0 for these cases. TEST2 illustrates how the target of ADD_MS becomes missing if either operand is missing. These effects are evident from the PRINT statements in the CREATE step and in the outcome of the DISPLAY step.

5.7 LINK_MISSING

Section 3.10 describes LINKing files, which treats input variables as real. On a keyed LINK, VPLX will treat as a fatal error any observation that cannot be matched when the corresponding INPUT statement is encountered.

LINK_MISSING provides an alternative handling of keyed link. If there is no matching record according to the key variable or variables, then the remaining variables become missing, but VPLX continues. The LINK_MISSING statement is of the form:

```
LINK_MISSING  fname
```

In most respects, the rules for LINK_MISSING parallel those for LINK. For example, the first LINK or LINK_MISSING statement in a CREATE step may only appear after the INPUT and FORMAT statements for the primary IN= input file. Similarly, the INPUT and FORMAT must be established for a linked file before another LINK or LINK_MISSING statement may appear. LINK and LINK_MISSING may be used together within the same CREATE step.

5.12

The LINK and LINK_MISSING statements declare the file for which the next INPUT statement applies, but the placement of the INPUT statements determines when data are to be read. Consequently, whether a LINK or LINK_MISSING statement occurs within or outside of an IF block does not affect the conditions under which the file will be read, whereas placement of the INPUT statement is critical.

FORMATs should be established for each linked file in the same way as the primary file.

The INPUT statement for LINK_MISSING is:

```
INPUT  vlist / KEY keylist
```

where:

- a) *vlist* and *keylist* are both lists of variables;
- b) the variables in *keylist* are included in *vlist*;
- c) the variables in *keylist* are defined previously, that is, before this INPUT statement;
- d) the variables in *vlist* but not in *keylist* have not yet been defined and become defined as real with missing variables.

Under LINK_MISSING, the new variables in *vlist* but not in *keylist* all begin as type real with missing. If the keyed match fails, then these variables are set to missing, otherwise they are set equal to the value. If the end-of-file is reached on the secondary file before the primary file, then all subsequent values for the new variables in *vlist* are set to missing.

For keyed match under either LINK or LINK_MISSING, unmatched records on the secondary file are ignored and not treated as an error. An end-of-file on the primary file before the end on the secondary file is similarly not an error.

5.8 An Example of LINK_MISSING.

The following example is based on modification to exam13.crd.

```
comment  EXAM17  
  
comment  This example modifies EXAM13.  The additional file for  
         owners is matched through LINK_MISSING
```

```

create in = exampl11.dat out = exampl11.vpl

input  rooms persons cluster tenure

      4 variables are specified

format  (4f2.0)

comment The input data set contains
5 7 1 2
6 8 2 2
5 2 3 1
4 1 4 2
8 4 5 1
8 2 6 1

link_missing  exampl17.dat
      Assigned to unit 19

comment  The input data set contains:
3 100000 75000
5 150000 100000
6 100000 0

input  cluster assessment mortgage / key cluster

      3 variables are specified

format (f2.0,2f7.0)

print tenure assessment mortgage
*** PRINT request 1

      (Simple) jackknife replication assumed

      Size of block 1 = 8

      Total size of tally matrix = 8

      Unnamed scratch file opened on unit 13

      Unnamed scratch file opened on unit 14

**** End of CREATE specification/beginning of execution
PRINT request 1
      tenure 2.000000
      assessment (M)
      mortgage (M)
PRINT request 1
      tenure 2.000000
      assessment (M)
      mortgage (M)
PRINT request 1
      tenure 1.000000
      assessment 100000.000000
      mortgage 75000.000000
PRINT request 1
      tenure 2.000000
      assessment (M)

```

5.14

```
mortgage                (M)
PRINT request           1
  tenure                 1.000000
  assessment             150000.000000
  mortgage              100000.000000
PRINT request           1
  tenure                 1.000000
  assessment             100000.000000
  mortgage               .000000

End of primary input file after obs #      6

End on unit 19 after obs #                 3
```

Exhibit 5.4 An example illustrating LINK_MISSING. With LINK_MISSING, the new variables listed on the corresponding INPUT statement become real with missing. Observations defined by the primary input file that do not match according to the key are assigned missing values. In contrast, a keyed link that cannot be satisfied results in a fatal error.

NOTES

1. In other words, the number of observations after exclusions from SELECT statements, if any.
2. A possibly useful extension of the VPLX syntax would be to add the range "MISSING" in addition to the meanings of LOW, HIGH, and RES. This extension would treat a comparison of a missing value to the range "MISSING" as true. This extension has not yet been implemented, however.