Checking Values of Numeric Variables

- range checks
  - when you know what the range of possible values is for a given quantitative variable
- internal consistency checks
  - when you don’t know in advance what a reasonable range is and simply want to look for extreme values relative to the rest of the data

Using proc means to inspect the data

- shows number of missing and nonmissing observations
- inspection of min and max values suggests there are errors and we need to do more checking

```
/* Program 2-1 Using PROC MEANS to detect invalid and missing values */
PROC MEANS DATA=PATIENTS N NMISS MIN MAX MAXDEC=3;
  TITLE "Checking Numeric Variables in PATIENTS data set";
  VAR HR SBP DBP;
RUN;
```

```
Variable          Label               N Miss Minimum Maximum
------------------------------------|-------|--------|--------|--------|
HR                Heart Rate          28    3       10.000 900.000
SBP               Systolic Blood Pressure 27    4       20.000 400.000
DBP               Diastolic Blood Pressure 28    3       8.000   200.000
```

Prettier data description using proc tabulate

- FORMAT option tells SAS to use numeric format 7.3 for all output in this procedure unless otherwise specified
  - field width of 7 with 3 places to the right of the decimal point
- VAR statement tells procedure which quantitative variables to produce summary statistics for
- content of TABLES statement
  - optional definition of separate pages of table goes before first comma if used (not used here)
  - definition of rows of table followed by comma
  - definition of columns of table
  - / optional additional options
• TABLES statement in this example
  - one row for each of the variables HR, SBP, and DBP
  - statistics N, NMISS, MEAN, MIN, and MAX in the columns
  - RTSPACE = 18 allows for 18 spaces for all row labels
  - N*F=7.0 NMISS*F=7.0 means to use format 7.0 for N and NMISS

• KEYLABEL statement replaces keywords for chosen statistics with more understandable labels

• proc tabulate output will not print correctly if you do not have the formchar option set

Using PROC UNIVARIATE to look for outliers

• proc univariate yields more detailed and useful information about values of numeric variables

• PLOT option provides m
  - stem-and-leaf plot if dataset is fewer than 200 observations
  - very ugly histogram otherwise
  - box plot
  - normal probability plot

Using ODS statement and ID statement to pinpoint check for extreme values

• ID statement prints values of one variable, in addition to observation number, in table of extreme values

• ODS (output delivery system) statement can be used to limit which parts of PROC UNIVARIATE output are printed
  - available in Version 7 and later of SAS
Using PROC PRINT with a WHERE statement to list invalid data values

- WHERE statement may have multiple conditions
- note logical operators used

Checking for Missing Values

- ways in which missing values can occur in a SAS data set
  - raw data value missing (intentionally or accidentally)
  - invalid value can cause missing value to be created
    * e.g. reading character value with a numeric informat
    * invalid dates
  - operations, such as assignment statements, can create missing values
    * e.g. trying to take log of negative number
- for some variables, missing values may be expected and may not create problems in analysis
- for other variables (such as patient IDs), missing values may not be permissible
Inspecting the SAS log

- if you know that a numeric variable in the data file contains invalid character values, might want to read that variable with a character informat and perform character to numeric conversion using INPUT function
  - keeps log file more readable
  - makes it easier to spot other errors reported in log

```sas
OPTIONS FORMCHAR = "|----|+|---+=|-\<>*" LS = 75 NODATE;
* LIBNAME CLEAN "C:\CLEANING";
*DATA CLEAN.PATIENTS;
DATA PATIENTS;
*INFILE "C:\temp\patients.dat" PAD;
INFILE "/group/ftp/pub/kcowles/datasets/patients.dat" PAD;
INPUT @1 PATNO $3. @4 GENDER $1. @5 VISIT MMDDYY10. @15 HR 3. @18 SBP 3. @21 DBP 3. @24 DX $3. @27 AE $1.;
LABEL PATNO = "Patient Number" GENDER = "Gender" VISIT = "Visit Date" HR = "Heart Rate" SBP = "Systolic Blood Pressure" DBP = "Diastolic Blood Pressure" DX = "Diagnosis Code" AE = "Adverse Event?";
FORMAT VISIT MMDDYY10.;
RUN;
```

NOTE: The infile "/group/ftp/pub/kcowles/datasets/patients.dat" is:
File Name="/group/ftp/pub/kcowles/datasets/patients.dat",
Owner Name=UNKNOWN, Group Name=UNKNOWN,
Before using dataset for any serious purpose,

- invalid dates need to be checked
- decision needs to be made concerning ‘NA’ value for heart rate

Using *proc means* and *proc freq* to count missing values

- using *proc means* to check for missing numeric values is straightforward

```
TITLE "Missing Value Check for the PATIENTS data set";
PROC MEANS DATA=PATIENTS N NMISS;
RUN;
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Label</th>
<th>N</th>
<th>Miss</th>
</tr>
</thead>
<tbody>
<tr>
<td>VISIT</td>
<td>Visit Date</td>
<td>24</td>
<td>7</td>
</tr>
<tr>
<td>HR</td>
<td>Heart Rate</td>
<td>28</td>
<td>3</td>
</tr>
<tr>
<td>SBP</td>
<td>Systolic Blood Pressure</td>
<td>27</td>
<td>4</td>
</tr>
<tr>
<td>DBP</td>
<td>Diastolic Blood Pressure</td>
<td>28</td>
<td>3</td>
</tr>
</tbody>
</table>

Using *proc freq* to count missing values of character variables

- may not be sensible to create one-way frequency tables for all character variables
  - some may take on thousands of unique values
- create character format that has only two values, one for missing and one for nonmissing
- to make *proc freq* display output for all character values in dataset, use SAS keyword `_CHARACTER_` in the tables statement (or provide list of names of character variables)

```
PROC FORMAT;
   VALUE $MISSCNT ' ' = 'MISSING'
             OTHER = 'NONMISSING';
RUN;
PROC FREQ DATA=PATIENTS;
   TABLES _CHARACTER_ / NOCUM MISSING;
   FORMAT _CHARACTER_ $MISSCNT.;
RUN;
```

```
<table>
<thead>
<tr>
<th>Patient Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATNO</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>MISSING</td>
</tr>
<tr>
<td>NONMISSING</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENDER</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>MISSING</td>
</tr>
<tr>
<td>NONMISSING</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diagnosis Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
**List Missing data values and ID variable**

- counting missing values usually is not enough
- if you have variables for which missing values are not allowed, you need to locate the observations so that the original data values can be checked and corrected
- using the SAS internal name `_NULL_` in a `DATA` statement causes SAS not to actually create another dataset in its memory
  - used when we want SAS to send output to a file or display
- `PUT` statement sends row of output to file or display

---

**Attempting to locate missing or invalid unique identifiers**

- obviously you can’t list which patient number is missing!
- one solution: report the patient number or numbers preceding the missing number *(in the original order of the raw data file)*
- if you sort the dataset first, all missing values will “float” to the top and you won’t have a clue which patients they belong to
- add the observation number to the output by printing the value of the internal SAS variable `_N_`
/****************************************************************************
Program 3-3  Attempting to locate a missing or invalid patient ID by listing
two previous ID's
****************************************************************************
DATA _NULL_;  
SET PATIENTS;  
***Be sure to run this on the unsorted data set;  
FILE PRINT;  
TITLE "Listing of Missing Patient Numbers";  
PREV_ID = LAG(PATNO);  
PREV2_ID = LAG2(PATNO);  
IF PATNO = ' ' THEN PUT "Missing Patient ID. Two previous ID's are:"
      PREV2_ID "and " PREV_ID / @5 "Missing Record is number " _N_;  
ELSE IF INPUT(PATNO,?? 3.) = . THEN
    PUT "Invalid Patient ID: " PATNO +(-1)""). Two previous ID's are:"
    PREV2_ID "and " PREV_ID / @5 "Missing Record is number " _N_;  
RUN;

Listing of Missing Patient Numbers

Invalid Patient ID: XX5. Two previous ID's are: 003 and 004
Missing Record is number 5
Missing Patient ID. Two previous ID's are: 006 and 007
    Missing Record is number 8