STAT 541

Chapter 22: Using Best Practices

Outline

- Executing Only Necessary Statements
- Eliminating Unnecessary Passes through the Data
- Reading and Writing Only Essential Data
- Storing Data in SAS Data Sets
- Avoiding Unnecessary Procedure Invocation

Executing Only Necessary Statements

- Position of subsetting IF
  - Even when subsetting IF is constructed from calculated variables, try to position it as soon as possible in the program to prevent unnecessary processing of cases that will not be output

Executing Only Necessary Statements

```
data doerasch;
set doe2012.doedb;
nonmiss=nmiss(of q1-q25);
if nonmiss le 7;
array q q1-q25;
sone=0;
stwo=0;
sthree=0;
sfour=0;
sfive=0;
totscore=0;
do over q;
totscore+q;
if q eq 1 then sone+1;
else if q eq 2 then stwo+1;
else if q eq 3 then sthree+1;
else if q eq 4 then sfour+1;
else if q eq 5 then sfive+1;
end;
run;
```

Executing Only Necessary Statements

- Using Conditional Logic Efficiently
  - IF-THEN/ELSE is efficient when
    - ***condition*** is based on character values
    - Data values are unevenly distributed
    - Condition comprises a small number of cases
    - Condition is ordered by frequency of occurrence
    - ELSE IF and DO/END constructions are used
  - SELECT (we've seen this once before) is efficient when
    - *condition* is based on numeric variables
    - Data values are evenly distributed
    - DO/END constructions are used

Executing Only Necessary Statements

- SELECT example

```
data a; set b;
select(varname);
when(num1) do; .. end;
when(num2) do; .. end;
..
otherwise do;.. end;
end;
```

Executing Only Necessary Statements

- IF-THEN/ELSE conditions suggest a macro
- The value(s) that determine whether the logical condition is true may be turned into value(s) of macro parameter(s)

Eliminating Unnecessary Passes through the Data

- Create multiple data sets from a single data set with OUTPUT (vs. multiple subsetting IFs)

```
data a b; set c;
if condition1 then output a;
else output b; run;
```

Eliminating Unnecessary Passes through the Data

- Use WHERE in PROCs (e.g., PROC PRINT and PROC SORT) rather than a subsetting IF in a DATA step following by the PROC
- Use PROC DATASETS to modify data attributes rather than the DATA step
  - Cannot modify data (e.g., LENGTH, type), only descriptions (e.g., LABEL, NAME)

Reading and Writing Only Essential Data

- WHERE is more efficient than subsetting IF for variables in the input data set
  - WHERE selects cases in the input buffer
  - Subsetting IF selects cases after they have been loaded from the input buffer into the PDV (Program Data Vector)
- Subsetting IF can operate on any variable in the PDV, including newly-created variables
- Subsetting IF can operate on external data sets
- Subsetting IF can be embedded in conditional statements

Reading and Writing Only Essential Data

```
proc print data=a (firstobs=4 obs=3);
where condition;
```

- FIRSTOBS and OBS refer to the observations in the subset, not the original data set

Reading and Writing Only Essential Data

- When creating a new data set from an external file (e.g., called with an INFILE statement), position subsetting IF *after* the INPUT statement to improve efficiency
  - This works when the INPUT statement only reads a subset of the variables in the external file (e.g., through column-formatting)

Reading and Writing Only Essential Data

- Using KEEP= and DROP= in a SET statement is more efficient than using them in either a DATA statement, or using KEEP and DROP statements in a DATA step
  - Reasoning is similar to reasoning for preferring WHERE over a subsetting IF (with similar disadvantages as well)

Storing Data in SAS Data Sets

- Advantages of storing data in a permanent SAS data set rather than reading from a raw data file when
  - More efficient when repeatedly using the data set in PROC or DATA steps
  - Self-documentation (labels, names, lengths, etc)
  - No processing of the raw data before using

Avoid Unnecessary Process Invocation

- Use procedures that can create multiple reports
  - PROC SQL
  - DATASETS (you can process multiple data sets by including MODIFY/RUN blocks in PROC DATASETS, or use implicit RUN statements)
  - FREQ (e.g. `TABLE A B A*B`)
  - TABULATE
  - BY groups
  - RUN group processing (DATASETS, CHART, PLOT, GLM REG, DATASETS