

DC SAS User's Group

March 4, 2003
General Meeting
8:30am to 12noon
TBA

1

Objectives

- Use tested methodology
- Develop easily maintainable code
- Provide documentation

3

Generating a Complete Set of Ratio Edits Using SAS

Roger L. Goodwin
U.S. Bureau of the Census
4700 Silver Hill Road
Washington DC 20233

Roger.L.Goodwin@census.gov

2

Implied Edits

Explicit ratio edits

$$E_1: L_{ik} \leq \frac{v_i}{v_k} \leq U_{ik} \text{ and } E_2: L_{kj} \leq \frac{v_k}{v_j} \leq U_{kj}$$

Imply

$$E_3: L_{ik} L_{kj} \leq \frac{v_i}{v_j} \leq U_{ik} U_{kj}$$

Problem

Given a set of explicit ratio edits, find all implicit ratio edits

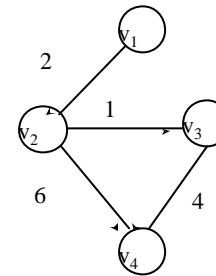
4

Example

- Given:
2.0 # APR/EMP # 4.0
1.0 # APR/QPR # 3.0
- Rewrite APR/EMP as
0.25 # EMP/APR # 0.5
- Implies the edit
0.25 # EMP/QPR # 1.5

5

Example



$$\begin{aligned}v_1 / v_2 &\leq 2 \\v_2 / v_3 &\leq 1 \\v_2 / v_4 &\leq 6 \\v_3 / v_4 &\leq 4\end{aligned}$$

7

Solution

- Explicit ratio edits are associated to a directed graph
- Two nodes are *connected* if a set of ratio edits exist between them
- “Travel” from node v_i to node v_k to calculate bounds

6

Implementation

SAS/IML (Two separate programs)

- Calculate implicit bounds using the product of the explicit edits’ bounds
- Calculate implicit bounds using Floyd’s shortest path algorithm

SAS/OR

- Calculate implicit bounds using shortest path algorithm as implemented in SAS network flow

8

Key Software Features

Environment

- Works in either a PC or Unix environment
- Save complete set of edits to a permanent SAS data set

9

Key Software Features

Other Features

- Handles multiple industries
- Allows explicit edits and data fields to differ between industries
- Implements infinity as a keyword parameter

11

Key Software Features

Simple Checks

- Missing bounds or fields
- Missing mnemonic names
- Redundant edits
- Inconsistent edits

10

Test Data

- Test data provided by ESMPD
- Tested
 - Explicit edits that imply impossible relationships
 - Varying number of implicit ratios per industry
 - Disconnected ratio tests

12

Comparison

SAS/IML Implementations

- Similar to current production code
- Fast run-time compared to OR code
- Requires 3 simple DO loops for computing the bounds

SAS/OR Implementation

- Requires installation of SAS/OR
- Slow run-time compared to the IML programs
- Uses a canned SAS procedure

13

Flow Chart

- Next 3 pages.

15

Comparison

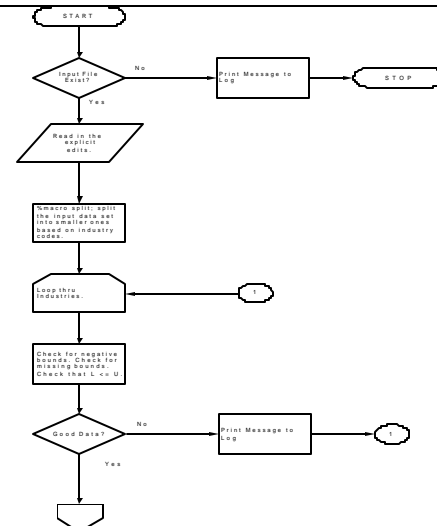
SAS/IML Implementation

- Prints bounds as they are updated, similar to production FORTRAN code
- Saving edits contributing to optimal bounds has prohibitive space requirements

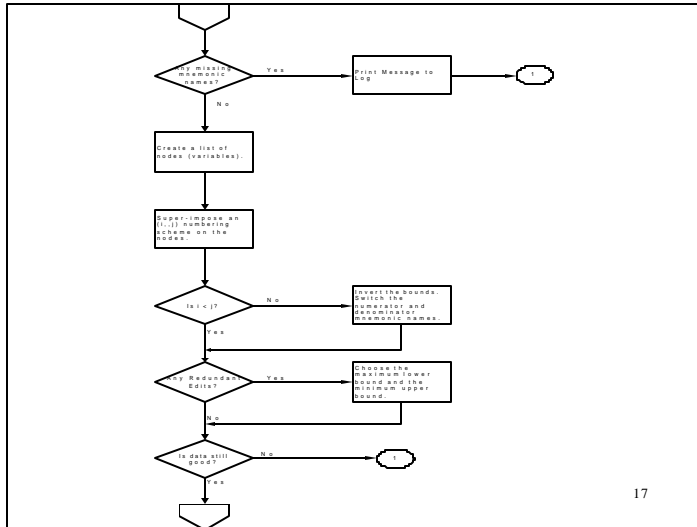
SAS/OR Implementation

- Immediately see edits leading to optimal bounds
- Space requirement not an issue

14



16

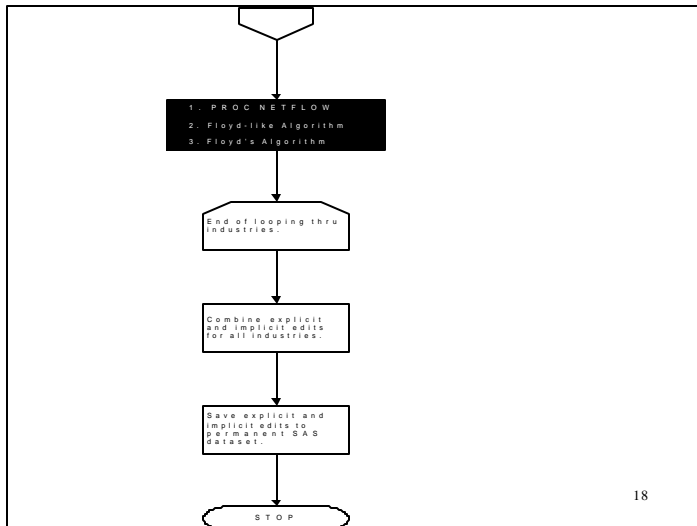


17

PROC NETFLOW

- Solves network problems (e.g. transportation problems).
- Can find shortest paths.
- Calculates solutions using:
 - Simplex Method (default)
 - an Interior Point Algorithm
- Audit trail easy to understand.

19



18

```

/* Specify labels for nodes. In particular, use the names of the fields
contained in the explicit edits. Initially
specify that each node is neither a source or a sink. Later in program
specify source/sink status for pairs of nodes; */

```

```

datas_nodes;
set summaries end = eof;
keep node supdem;

```

```

supdem = 0;
node = items;
p+1;
call symput("x"||left(p), put(node, 32.0)); /* macro variables needed by macro allpaths */

```

```

if eof then call symput("nfields", put(_n_, 10.0));

```

```

run;

```

20

```

/* Specify arcs and costs. In particular, the arcs correspond to the
explicit-edit relationships and the costs are logs of the upper
and lower tolerances.*/

```

```

DATA NETWORK(keep = from to edit cost);
set explicit_edits;

```

```

FROM = trim(left(my_numerator));
TO = trim(left(my_denominator));
EDIT = trim(FROM)||" / "||trim(TO);
COST = LOG2(U);
OUTPUT;
FROM = trim(left(my_denominator));
TO = trim(left(my_numerator));
EDIT = trim(left(FROM)||" / "||trim(left(TO));
if L ne 0 then COST = LOG2(1/L); else cost = &infy;
OUTPUT;

```

```
run;
```

21

```
/* look for infeasible solutions */
```

```
data _null_;
```

```

status = scan("&_ornetfl ", 5, "=");
call symput("dscnctd ", put(status, 10.0));

```

```
run;
```

```
/*
```

```

%put;
%put;
%put;
%put *** Message: Value of NETFLOW macro variable _ornetfl is &_ornetfl;
%put;
%put;
%put;
*/

```

```
/* add only non-infeasible solutions to the data sets */
```

```
%if &dscnctd ne INFEASIBLE %then %do;
```

23

```

/* =====
Macro PATH terminate shortest path between a pair of nodes. See
pages 41 and 42 of the SAS/OR User's Guide, Version 6, First Edition; */

```

```
%macro path(_from=_to=);
```

```
*Set source/sink status for beginning and ending nodes;
```

```

DATA NODES;
SET S_NODES;
IF trim(NODE) eq "%trim(&_from)" THEN SUPDEM=1;
If trim(NODE) eq "%trim(&_to)" THEN SUPDEM=-1;
RUN;

```

```

proc netflow arcout=arc_sav
arcdata=network nodedata=nodes;

```

```

node node;
supdem supdem;
tail from;
head to;
cost cost;
/* reset verbose=1; */

```

```
run;
```

22

```

DATA EDIT(KEEP=EDIT _EDIT PART UPPR)
PATH(KEEP=EDIT EDIT_PATH _EDIT_PATH UPPR_PATH);

```

```
SET ARC_SAV END=END;
```

```
EDIT= "&_from / &_to";
```

```
IF "&_from" < "&_to" THEN
```

```
DO;
```

```
_EDIT=EDIT;
```

```
PART="1";
```

```
END;
```

```
ELSE
```

```
DO;
```

```
_EDIT="&_to / &_from";
```

```
PART="2";
```

```
END;
```

```
IF _FLOW_=1 THEN
```

```
DO;
```

```
EDIT_PATH=trim(left(FROM))||" / "||trim(left(TO));
```

```
_EDIT_PATH=trim(left(TO))||" / "||trim(left(FROM));
```

```
if cost < log(&infy) then UPPR_PATH=2**COST; else uppr_path = &infy;
```

```
TCOST+COST;
```

```
OUTPUT PATH;
```

```
END;
```

24

```

IF END THEN
DO;
  if tcost < log(&infty) then UPPR=2**TCOST; else uppr = &infty;
  OUTPUT EDIT;
END;
RUN;

```

```

PROC APPEND BASE=EDITS DATA=EDIT;
RUN;

```

```

PROC APPEND BASE=PATHS DATA=PATH;
RUN;

```

```

%end; /* of being a feasible solution */

```

```

%mend path;

```

25

PROC IML

- IML programming similar to FORTRAN.
- Floyd-like Algorithm:
 - Uses 3 nested DO loops.
 - Multiply costs (not add them).
 - Audit trail is difficult to understand.
 - A good audit trail requires a binary tree:
 - ✓ $2^{d+1} - 1$ array.
 - ✓ where d is the depth.

27

PROC NETFLOW Demonstration

- Demo the NETFLOW code and output here.

26

```

/**** Floyd's algorithm; */

```

```

proc iml;

```

```

  use explicit_edits;
  read all var{ij U} into ub;
  read all var{ij L} into lb;

```

```

  use summaries;
  read all var{items} into names;
  ...

```

28

```

**** calculate bounds;

do k=1 to n;
  do i=1 to n;
    if u[i,k] < inf then
      do j=1 to n;
        if u[k,j]<inf & i^=j then
          do;
            temp=u[i,k]*u[k,j];
            if temp<u[i,j] then do;
              u[i,j]=temp;
              ed1=i||k||u[i,k]; *keep record of edits updated;
              ed2=k||j||u[k,j]; ed3=i||j||u[i,j];
              ed4=ed1||ed2||ed3;
              chari=names[i]; chark=names[k];
              charj=names[j]; uik=char(u[i,k]);
              ukj=char(u[k,j]); uij=char(u[i,j]);
              b1=chari||chark||uik; b2=chark||charj||ukj;
              b3=chari||charj||uij; ed5=b1||b2||b3;
              trail=trail//ed5; AudTrail=AudTrail//ed4;
            end;
          end;
        end;
      end;
    end;
  end;
end;
end;

```

29

PROC IML Demonstration

- Demo the IML code and output here.

31

```

....
create imp_edits&q from imp;
append from imp;
close imp_edits;

create imp_names&q from names3;
append from names3;
close imp_names;

quit; /* iml code */

```

30

SAS' %Window Macro

- Easy to use.
- Similar to dBASE III programming.
- Note semicolon in the window definition.

32

```

%macro my_edits;

%macro select_4;

%let choice_sel4 = ;
%let location = "C:\Implied";
%let file_name = MyEdits;
%let ioutfile = All_Edits;
%let print = Yes;
%let infy = 1000000;
%let subset_first = 1;
%let subset_last = 1163;

%local not_used;
%let not_used = No;

%include "C:\Implied\GenBndsIML.sas";
%include "C:\Implied\GenBndsORsas";

```

33

```

#28 @28 "Choose Program For Calculating the Implicit Edits" attr = highlight color = blue

#31 @10 "1. Floyd Bounds program based on shortest paths (GenBndsIML.sas)"

#33 @10 "2. PROC NETFLOW program based on shortest paths (GenBndsORsas)"

#35 @10 "3. Exit"

#38 @15 "Select One:" @28 choice_sel4 2 attr = underline required = yes
;

```

35

```

>window implicit_edits

```

```

#5 @28 "      ImpliedWin v 1.0" attr = highlight color = blue
#6 @28 "Calculate Implicit Edits From Explicit Edits" attr = highlight color = blue

#10 @10 "Location of input file (include quotes):" @52 location 40 attr = underline

#12 @10 "      Name of input file (no quotes):" @52 file_name 15 attr = underline

#14 @10 "Send output to SAS Output Window? (Y/N):" @52 p rint 4 attr = underline

#16 @10 "      Infinity is defined as:" @52 infy 8 attr = underline

#18 @10 "      Name of output file:" @52 ioutfile 15 attr = underline

#20 @10 "      Choose range of industries (Y/N):" @52 not_used 4 attr = underline
#21 @10 "      First industry number:" @52 s ubset_first 9 attr = underline
#22 @10 "      Last industry number:" @52 s ubset_last 9 attr = underline

```

34

```

%do %while(&choice_sel4 ne 3);

%display implicit_edits delete;

%if &choice_sel4 eq 1 %then
    %implied2(location = &location, in_file_name = &file_name, print = &print,
infy = &infy, out_file_name = &ioutfile, first_industry = &subset_first,
last_industry = &subset_last);

%else
%if &choice_sel4 eq 2 %then
    %implied1(location = &location, in_file_name = &file_name, print = &print,
infy = &infy, out_file_name = &ioutfile, first_industry = &subset_first,
last_industry = &subset_last);

%end;

%mend select_4;

%select_4;

%mend my_edits;

```

36

```

%my_edits;

```

%Window Demonstration

- Demo the %Window code here.

37

Conclusion (con't)

- SPEERS
- Plain Vanilla
- FH Theorem works for any edits (not just ratios).

39

Conclusion

What to do with the complete set of edits?

- BOC passes questionnaire responses through edits.
- Responses that fail are corrected.
- The Fellegi-Holt Theorem (1976) states:
 - Given a complete set of edits, “[You] can identify a minimal set of variables, when changed, will result in satisfying all edits” without failing on some other edit.

38

References

1. Brassard, G., and Bratley P., *Algorithmics: Theory and Practice*, Prentice Hall, NJ, 1988.
2. Fagan, J., "Generating Implied Ratio Edits", unpublished Census manuscript. October 1999.
3. Fellegi, I. P. and Holt, D., "A Systematic Approach to Automatic Edit and Imputation," *Journal of the American Statistical Association*, No. 71, 1976.
4. Garica, M. and Goodwin, R., "Developing SAS Software for Generating a Complete Set of Ratio Edits," SRD Report RR-Statistics #2002-6.
5. Goodwin R., Garcia, M., and Thompson, K., "The GenBounds Software for Generating a Complete Set of Ratio Edits: %Implied User's Guide," SRD Research Report SS-Statistics #2002-01.

40

References (con't)

6. Greenberg, B. and Petkunas, T., "SPEER (Structured Program For Economic Editing and Referrals)", Proceedings of the Section on Survey Research Methods, ASA, 1990.
7. Hillier, Frederick S. and Lieberman, Gerald J., "*Introduction to Operations Research 5th Edition*," McGraw-Hill Publishing Co, NY, NY, 1990.
8. SAS Institute Inc., *SAS/OR User's Guide: Mathematical Programming, Version 8* , Cary, NC: SAS Institute Inc., 1999, 566pp.
9. Thompson, K. J. and Sigman, R., "Statistical Methods for Developing Ratio Edit Tolerances for Economic Data", J. Official Statistics, 2000.
10. Thompson, K.J. and Sigman, R., "User Guide for Generalized SAS Ratio Edit Parameter Development Programs", Washington, DC.: U.S. Bureau of the Census (Technical Report #ESM-9602, available from the Economic Statistical Methods and Programming Division).

41

References (con't)

11. Winkler, W. and Draper, L. , "The SPEER Edit System", Proceedings of the Conference of European Statisticians, Section on Statistical Data Editing, UNECE, 1997.

42

