Paper 109

Language interpretation of prescription label sigs: Computing daily consumptions in medication compliance studies

Patty J. Keys, Pharm. D.
Wayne State University, College of Pharmacy and AHP, Detroit, MI

ABSTRACT

Computing daily consumptions in medication compliance studies has been an arduous task. Pharmacy dispensing information systems typically report a 'days supply' data field. However, the information contained in it is unpredictable, at best. A more reliable method of computing daily consumptions is interpretation of the sig field on the prescription label. However, the data entered in it is commonly a combination of pre-determined abbreviations, custom abbreviations and free text entries which often times include misspelled words.

INTRODUCTION

SAS ® software can be used as a language interpreter of the sig data field. This provides the ability to interpret the information and convert it to daily consumption totals. The algorithm employs an ordered application of the index function in conjunction with conditional assignment statements and link statements to establish a match between the contents of the sig field and the daily consumption total.

This coding technique is targeted for SAS users with intermediate coding skills. The coding employs %INCLUDE, assignment statements, link statements, compress functions, and index functions.

ALGORITHM LOGIC

The algorithm is constructed in 4 sections. The main section contains the SAS coding which controls the flow of the program. The three remaining sections contain the language interpretation code for the dose, dose interval and excluded sigs. Each of these three sections are separate external files. The external file structure facilitates the process of customizing their content to include new sig language in the algorithms decision rule processes.

THE MAIN FILE: FLOW CONTROL

This file sequentially includes each of the 3 external files for processing. Additionally, it is constructed to facilitate on the fly assessments of the computations.

```
data csig (keep=n csig nddose dose ddose ddosemd
drg_name);
set csig_mis;
ddosemd=1;
dosemd=1;
n=_n_;

%INCLUDE EXCLUDE;
 if excludes=1 then do;
  put n= excludes= nddose= dose= ddose=
    csig= drg_name=;
  file drops;
  put n excludes nddose dose ddose
     csig $ drg_name $;
 end;
 else do;
    %INCLUDE DOSE;
    %INCLUDE INTERVAL;
    if excludes=2 or excludes=3 then do;
     put n= excludes= nddose= dose=
       ddose= csig= drg_name=;
     file drops;
     put n excludes nddose dose ddose
        csig $ drg_name $;
    end;
  end;
if excludes=. then do;
 nddose=ddose*dose*ddosemd*dosemd;
end;

drop q l ql a b c;
if excludes=.;
run;
```

THE EXCLUDED FILE

The excluded file contains a listing of all the possible representations of sigs that would be excluded from computing daily consumptions. This includes all possible representations of 'as needed' regimens, topically applied medications such as ointments, creams, eye drops, ear drops, etc. (See figure 1)

THE DOSE FILE

The dose file contains a listing of all possible representations of the dose instruction component of the prescription sig field. For example the coder needs to include all possible representations of the dose instruction 'Take one tablet'. This may include 'Take one tablet', 'Give one tablet', '1T', etc. This process is repeated for all possible dosage forms the medication instructions may reference. This may include capsules, teaspoonfuls, scoops, packets, patches, etc. Additionally, it is repeated for all possible dose quantities that may be referenced on the prescription label. The multitude of these representations is extensive and requires considerable mentation to opitmize the language interpretation of the algorithm. (See figure 2).

THE INTERVAL FILE

Lastly , the interval file contains a listing of all the possible representations of the dose interval component of the prescription sig field. For example the coder needs to include all possible representations of the dose interval instruction 'twice a day'. This may include 'BID', 'twice a day', 'twice daily', 'morning and evening', 'AMPM', 'two times daily', etc. This process is repeated for all possible dosing interval representations that may be referenced on the prescription label. As in the dose file, the multitude of these representations is extensive and requires an unwaivering commitment to exhaust the possible representations. (See figure 3).

ADDITIONAL CONSIDERATIONS

Prior to comparing the contents of the sig field to the external reference files, the data in the sig field must be modified to facilitate

interpretation. Most important in this process is the application of the compress function to the sig field data. This function removes all spaces in the field, resulting in an uninterrupted string of characters. In doing so, the interpretation of the code does not have to manage or recognize word delimiting boundaries.

An important consideration in constructing each of the external reference files is the order of the entry for each phrase representation. Since the program considers whether the conditional assignment statement is true or false in a sequential order and it scans the sig field at the level of a matching character string, the true statement must be presented before any representation of a substring of the sig reference. For example, the following phrases must appear as ordered below; '1Teaspoonful', '1T'. In this order, any sigs which contain the character string '1 Teaspoonful' will be properly matched to the daily dose associated with the sig reference '1Teaspoonful'.

In contrast, if the phrases were ordered as follows; '1T', '1Teaspoonful', then sigs having the meaning of '1 Teaspoonful' (which is a dose of 5) will be incorrectly matched to the dose of '1T' (which is a dose of 1). This would occur because contained within the phrase '1 Teaspoonful' is the phrase '1T'. And since '1T' is offered as a conditional assignment statement before '1Teaspoonful' is offered as a conditional assignment statement, then the finding of 'true' associated with '1T' would erroneously drive the assignment process.

Another consideration to manage in this process of language interpretation is the need to manage compound potency issues. An excellent example is associated with the drug product, Questran®. There are two formulations which contain two different concentrations of the active ingredient, Cholestyramine. Following the assignment of the daily dose based on the interpretation of the dose character string, the program code then scans (using the index function) the drug name field for character strings consistent with Questran or Questran Light®. If the index function finds only Questran, it modifies the computed daily dose nine-fold (dose *9). Alternatively, if the index function finds both the character strings 'Questran' and 'Light', then it modifies the dose 5-fold (dose*5). The dose is left unmodified if the index function does not find the character string 'Questran'.

INTERPRETATION EFFICIENCY AND ACCURACY

In terms of program code development, it is important to be able to assess the impact of the interpretation process on the daily dose assigments and sig exclusions. Without easily accessible information on the impact of the interpretation logic, the coders thought processes become overwhelmed and compromise the logic flow of the program. To address this, report files were generated with each interpretation run of the sig files. All excluded sigs were reported to two places, the SASLOG and a SAS dataset specific for these sigs. Included in these reports were the original number of the sig observation in the input file, a code for why it was excluded (based on un-interpretable dose, un-interpretable interval or matching sig exclusion criteria), the compressed sig variable, computed dose, computed interval, and computed daily consumption.

Similarly, all sigs which were successfully assigned daily doses were reported to a separate dataset. The same information is reported to this dataset as in the excluded sigs. This reporting system provides 'at a glance' capacity to assess the success rate of the language interpretation process. Additionally, it provides the information needed to determine the proportion of excluded sigs which are due to lack of language interpretation versus deliberate criteria based exclusion. Furthermore, it greatly enhances the ability to address sig exclusion based on inappropriate language interpretation, as well as, erroneous dose assignment computations.

CONCLUSION

The coding strategy demonstrated in this paper addresses a long-standing need in analyzing medication compliance pattern in existing prescription databases. Its use accomplishes computation of daily comsumption totals in more than 95% of scheduled dosing regimens. Furthermore, it accurately identifies and excludes non-scheduled medication regimens from the computation totals.

This coding technique can be successfully applied to scheduled medication regimens. Additionally, the algorithm may be modified to apply customized daily consumption decision rules for variable dose or dosing interval regimens. SAS base is the SAS product used in this coding strategy.

TRADEMARKS

CONTACT INFORMATION

Patty J. Keys, Pharm. D.
Asst Professor, Pharmacoeconomic Research
Wayne State University
College of Pharmacy and Allied Health Professions
228 Shapero Hall
Detroit, MI 48202
Work Phone: 313-577-5404
FAX: 313-577-5864
EMAIL: p.keys@wayne.edu

Figure 1: Exclude File

```
c=index(csig,'PRN');  link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'PRF'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'ASNEE'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'UD'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'OR1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'OR2'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
…
c=index(csig,'X1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'X2'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
…
c=index(csig,'F1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'F2'); link exclude; if c ge 1 then goto last1; else if c eq 0 then

…
…
c=index(csig,'5-1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'2-1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'2-2'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'1-1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'1-2'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'1-3'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'1-4'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'2-3'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'2-4'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'3-4'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'4-5'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
…
…
c=index(csig,'5TO1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'2TO1'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
…
…
c=index(csig,'QDTOBID'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'BIDTOTID'); link exclude; if c ge 1 then goto last1; else if c eq 0 then
c=index(csig,'TIDTOQID'); link exclude; if c ge 1 then goto last1; else if c eq 0 then

goto last1;

exclude: if c ge 1 then excludes=1; return;



last1:
```

Figure 2: Dose File

```
b=index(csig,'APPLYPAT'); link dose1;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'APPLYONE'); link dose1;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'APPLY1P'); link dose1;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'A1P');  link dose1;  if b ge 1 then goto last; else if b eq 0 then
…
b=index(csig,'11/2TEASPOONFUL');  link dose7_5;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'11/2TSP');  link dose7_5;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'11/2TABLESPOON');  link dose225;  if b ge 1 then goto last; else if b eq 0 then
b=index(csig,'11/2T');  link dose15;  if b ge 1 then goto last; else if b eq 0 then
…
b=index(csig,'1&1/2TEASPOONFUL');  link dose7_5;  if b ge 1 then goto last; else if b eq 0 then
…
b=index(csig,'1/2TEASPOONFUL');  link dose2_5;  if b ge 1 then goto last; else if b eq 0 then
…
b=index(csig,'2TEASPOONFUL');  link dose10;  if b ge 1 then goto last; else if b eq 0 then
…
…
do;
if excludes eq . then excludes=2; end; goto last;


  dose1: if b ge 1 and dose eq . then do; dose=1; end; return;
  dose15: if b ge 1 and dose eq . then do; dose=1.5; end; return;
  dose7_5: if b ge 1 and dose eq . then do; dose=7.5; end; return;
  dose225: if b ge 1 and dose eq . then do; dose=22.5; end; return;
  dose2_5: if b ge 1 and dose eq . then do; dose=2.5; end; return;
  dose10: if b ge 1 and dose eq . then do; dose=10; end; return;


last:

quest='QUEST'; light='LIGHT'; qu_light='RAN_LI';
    q=index(drg_name,quest);
    l=index(drg_name,light);
    ql=index(drg_name,qu_light);
    if q ne 0 then do;
      if l ne 0 or ql ne 0 then do;
        if dose lt 5 then do;
          if dose ne 2.5 then dose=dose * 5;
        end;
      end;
      if l eq 0  or ql eq 0 then do;
        if dose lt 5 then do;
          if dose ne 4.5 then dose=dose*9;
        end;
      end;
    end;
    drop quest light;
```

Figure 3: Interval File

```
a=index(csig,'QD');  link ddose1;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'QOD');  link ddose1;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'ONEVE');  link ddose12A;   if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'ONODD');  link ddose12A;   if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'YOTH');  link ddose12A;    if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'ALTERN');  link ddose12A;   if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'THREETIM');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'THRETIM');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'3TIM');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'3TM');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'TI');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'Q8');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'8H');  link ddose3;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AM&PM');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AMANDPM');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AMPM');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AM&HS');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AMANDHS');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'AMHS');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
a=index(csig,'MORNINGAND');  link ddose2;     if a ge 1 then goto last2; else if a eq 0 then
do;
if excludes eq . then excludes=3; end; goto last2;


  ddose1: if a ge 1 and ddose eq . then do; ddose=1; end;
  ddose2: if a ge 1 and ddose eq . then do; ddose=2; end; return;
  ddose3: if a ge 1 and ddose eq . then do; ddose=3; end; return;
  ddose4: if a ge 1 and ddose eq . then do; ddose=4; end; return;
  ddose12A: if a ge 1 and ddose eq . then do; ddose=1; ddosemd=ddose*0.5; d =0.5; end; return;

last2:
```