PharmaSUG China 2019 - Paper CC-057

A SAS Macro for Converting Character Dates of Various Formats into Numeric

Hao Dong, Hutchison Medi Pharma

ABSTRACT

Dealing with dates can be troublesome, yet important and inevitable in almost all kinds of analysis. When we calculate with dates, we need numeric dates; however, most of the time there's only character dates available in the dataset. That's when "input" becomes a necessary process. SAS have a number of date informats, which take quite a lot of efforts to memorize and not get confused. Transformation and calculation take a significant and undesirable amount of time. We hope to have an effortless and reliable method—macro—to finish this job. There are macros for date transformation, but a lot of them can only transform dates of a certain format. Then we end up with many macros, each of them only deal with one type of format, and this is contrary to the original intention of using a macro. This paper presents a macro that handles many types of dates we usually encounter in analysis, converting a raw character date into a numeric date, imputing partial dates and transform dates into CDISC standard formats, with the intention to make our lives a little bit easier.

MOTIVATION

Dealing with dates is inevitable in almost all kinds of analysis; however, it can be a frustrating process. Different database produces different date formats. Some are numeric, some are character. The order of day, month and year differs, and delimiters also come in many forms. When we need to standardize and calculate them, the input step can be cumbersome. SAS have a number of date "informats", which take quite a lot of efforts to memorize and not get confused. We hope to have an effortless and reliable method—macro—to accomplish this task. There are macros for date transformation, but a lot of them can only transform dates of a certain type. Then we end up with many macros, each of them only deal with one type of formats, and this is contrary to the original intention of using a macro. This paper presents a macro that handles many types of dates we usually encounter in analysis, converting a raw character date into a numeric date, imputing partial dates, and displaying dates in CDISC standard format, with the intention to make our lives a little bit easier.

BASICS OF DATE

Programmers are often faced with raw data, encountering various non-standard formatted dates. Among the many seemingly chaotic variants, there are four perspectives can help us better understand and memorize date formats.

First, a date is usually consists of three parts: day, month and year. Due to language and customary reasons, the order of the three components is different across the world. For example, in Chinese environment, the order year-month-day is generally used, while in the United States, the common expression is month-day-year. The order day-month-year is widely accepted in countries such as France, Canada and the United Kingdom. It is primary and critical to look at the order when we deal with dates. SAS does not recognize the order of year, month, and day very well. For example, "010203" can be recognized as February 3, 2001. It can also be understood as February 1, 2003 or January 2, 2003. Therefore, labor is required. The order of the year, month, and day should be defined so that SAS does not misidentify.

Second, the length of dates varies. It is important to define the width of SAS informats or formats. For instance, to input a date "21052019" which is length of 8, if we use "ddmmyy." informat and do not define the width, SAS will not correctly identify the date. That is because the default width of "ddmmyy." is 6; SAS will only read the first 6 digits. Another case is, sometimes the leading zeros of day and month is missing (e.g. 2019-5-21, 2019-1-1). This is when SAS input function makes mistakes very easily, because the lengths of dates in one variable are different. All dates in a variable have to keep the same length, so the leading zeros have to be filled in before input.

Third, delimiters such as “-” and “/” are often used to separate year, month and day (e.g. 2019-05-21). Sometimes there is no delimiter (e.g. 20190521). The forms of delimiter are usually not important; it is important to count delimiters for length.

Four, partial data should be dealt with special care. In order to keep as much information as possible, character partial dates should not be displayed as missing, instead, should display whatever is available in a standard format. As we know, the SAS input function does not recognize partial date. In numeric date field, partial date will simply be missing. In other cases, partial dates need to be imputed. A common imputation rule is: 1. if year is missing, then does not impute; 2. if month and day are missing, then impute the missing part as 1st July; 3. if only day is missing, then impute the missing part as 15th. Imputation rules will vary according to analysis needs.

PURPOSE OF THE PROGRAM

All operations of date variables can be summarized into two purposes: one is for calculation and the other is for display. For dates of calculation purposes, we must convert them into numeric variables to be able to calculate. For the purpose of the display, we will follow the CDISC standard; CDISC adopts the ISO8601 format. So we usually need a numeric date and a character date, each is for a different purpose.

This macro is used to transform character date to numeric date, impute partial date, and display date in CDISC standard (ISO8601 format). The input character date has to contain year, month and day. Partial or missing date will be notified. Time and datetime variable cannot be transformed by this macro. This macro only impute 2 partial situations: 1.both month and day are missing, 2.only day is missing. Missing date will not be imputed.

THE CALLING PROGRAM

To call the program, assume that a dataset “indata” has a character variable “testdate” looks like this: 21-5-2019, in the order of day-month-year.

	testdate
1	1-1-2019
2	3-2019
3	2019

Display 1. The original data

To output the date in numeric, impute partial date and display date in character of CDISC standard, use the following macro call.

```
%date(inds=indata,outds=outdata,invar=testdate,order=dmy,outvar=newdate,
impute=1,month_day=07/01,day=15);
```

THE DATE PROGRAM

DEFINITION

The macro has 8 parameters:

```
%macro date (inds=,outds=,invar=,order=,outvar=,impute=,month_day=, day=);
```

Details of input parameters are listed below:

- inds The input dataset.
- outds The output dataset.
- invar: the input date variable to be processed.
- order: the order of year, month and day. Put lower case ymd, mdy or dmy.

- outvar: Output date variable name.
- impute: If want to impute partial dates, put 1; else keep empty.
- month_day: Imputation rule when both month and day are missing，e.g. 07/01;
- day: Imputation rule when day is missing, e.g. 15

THE CONVERSION

The conversion from character to numeric includes 3 sub macros. The macro %numdate converts dates that only contain Arabic numerals, e.g. 2019-07-15; %chardate converts dates that contain alphabetic characters, e.g. 15Jul2019; %check_misspartial generate a new variable indicates the date is missing, partial or complete.

In %numdate macro, all dates do not contain alphabetic characters are characterized into 2 types: with delimiters or without any delimiter. Macro variables &y, &m and &d is calculated from the input parameter "order". This will be demonstrated later.

```
%macro numdate;

if strip(&invar)^=compress(&invar,,'kd') then do;
    year=scan(&invar,&y);
    month=scan(&invar,&m);
    day=scan(&invar,&d);
```

For those with delimiters, use the scan function to get year, month and day into separate variables. If month or day misses the leading zero, fill in so that all dates in the variable is the same in length. Then concatenate year, month and day into one variable "y_m_d". If length of year is 2, then length of y_m_d is 6; if length of year is 4, then length of y_m_d is 8. Then use informat yymmddw. to get the output variable. If any of year, month and day is missing, the output variable should be set to missing, or SAS will likely to produce incorrect result.

```
if missing(month) then month1="";else if length(month) in (1,3) then
month1="0"||compress(month,,'kd');else month1=month;
    if missing(day) then day1="";else if length(day) in (1,3) then
day1="0"||compress(day,,'kd');else day1=day;

    month=month1;
    day=day1;

    if missing(day) or missing(month) or missing(year) then do;
          &outvar=.;
    end;

    else do;

    y_m_d=compress(year,,'kd')||compress(month,,'kd')||compress(day,,'kd');
          if length(y_m_d)=6 then &outvar=input(y_m_d,yymmdd6.);
          if length(y_m_d)=8 then &outvar=input(y_m_d,yymmdd8.);
    end;
end;
```

Those without any delimiter are divided into 2 groups: length is 6 and length is 8. When length is 6, the lengths of year, month and day are all 2. When length is 8, the length of year is 4; the lengths of month and day are all 2. Use the substrn function to get them into separate variables. There are 6 kinds of arrangement in the order of year, month and day; each arrangement has different parameters for the substrn function. By convention, when a date contains no delimiter, it is impossible to miss the leading

zero. So the step of checking the missing zero is not processed here. Same as before, use informat yymmddw. to get the output variable.

```
   else do;

       if length(strip(&invar))=6 then do;
             if &y=1 and &m=2 and &d=3 then do;
                   year=substrn(&invar,1,2);
                   month=substrn(&invar,3,2);
                   day=substrn(&invar,5,2);
             end;
             if &y=1 and &d=2 and &m=3 then do;
                   year=substrn(&invar,1,2);
                   month=substrn(&invar,5,2);
                   day=substrn(&invar,3,2);
             end;
             if &m=1 and &y=2 and &d=3 then do;
                   year=substrn(&invar,3,2);
                   month=substrn(&invar,1,2);
                   day=substrn(&invar,5,2);
             end;
             if &m=1 and &d=2 and &y=3 then do;
                   year=substrn(&invar,5,2);
                   month=substrn(&invar,1,2);
                   day=substrn(&invar,3,2);
             end;
             if &d=1 and &y=2 and &m=3 then do;
                   year=substrn(&invar,3,2);
                   month=substrn(&invar,5,2);
                   day=substrn(&invar,1,2);
             end;
             if &d=1 and &m=2 and &y=3 then do;
                   year=substrn(&invar,5,2);
                   month=substrn(&invar,3,2);
                   day=substrn(&invar,1,2);
             end;

       y_m_d=compress(year,,'kd')||compress(month,,'kd')||compress(day,,'kd');
             &outvar=input(y_m_d,yymmdd6.);
       end;

       if length(strip(&invar))=8 then do;
             if &y=1 and &m=2 and &d=3 then do;
                   year=substrn(&invar,1,4);
                   month=substrn(&invar,5,2);
                   day=substrn(&invar,7,2);
             end;
             if &y=1 and &d=2 and &m=3 then do;
                   year=substrn(&invar,1,4);
                   month=substrn(&invar,7,2);
                   day=substrn(&invar,5,2);
             end;
             if &m=1 and &y=2 and &d=3 then do;
                   year=substrn(&invar,3,4);
                   month=substrn(&invar,1,2);
                   day=substrn(&invar,7,2);
             end;
```

```
            if &m=1 and &d=2 and &y=3 then do;
                  year=substrn(&invar,5,4);
                  month=substrn(&invar,1,2);
                  day=substrn(&invar,3,2);
            end;
            if &d=1 and &y=2 and &m=3 then do;
                  year=substrn(&invar,3,4);
                  month=substrn(&invar,7,2);
                  day=substrn(&invar,1,2);
            end;
            if &d=1 and &m=2 and &y=3 then do;
                  year=substrn(&invar,5,4);
                  month=substrn(&invar,3,2);
                  day=substrn(&invar,1,2);
            end;

      y_m_d=compress(year,,'kd')||compress(month,,'kd')||compress(day,,'kd');
            &outvar=input(y_m_d,yymmdd8.);
      end;

   end;

   %mend numdate;
```

Macro %chardate converts date with alphabetic letters. Here the informat *anydtdte.* is the omnipotent one for all. This informat corresponds to quite a few types of date formats; however, if date partially missing, the function will not input correctly. So it's important to judge whether part of the date is missing. Since a full year and day should be Arabic numbers, if any alphabetic character is detected, date is partially missing. If partially missing, the output variable is set to missing.

```
   %macro chardate;

      if &y=1 or &d=1 then do;
            if anydigit(substrn(strip(&invar),1,1))=0 or
   anydigit(substrn(strip(reverse(&invar)),1,1))=0 then do;
                  &outvar=.;
            end;
            else if &d=1 and
            upcase(substrn(compress(&invar),3,3)) not in ("JAN" "FEB" "MAR"
   "APR" "MAY" "JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC")
            and upcase(substrn(compress(&invar),2,3)) not in ("JAN" "FEB"
   "MAR" "APR" "MAY" "JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC")
            then do;
                  &outvar=.;
            end;
            else if &y=1 and
            upcase(substrn(reverse(compress(&invar)),3,3)) not in ("JAN"
   "FEB" "MAR" "APR" "MAY" "JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC")
            and upcase(substrn(reverse(compress(&invar)),2,3)) not in ("JAN"
   "FEB" "MAR" "APR" "MAY" "JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC")
            then do;
                  &outvar=.;
            end;
            else do;
                  &outvar=input(&invar,anydtdte32.);
            end;
      end;
```

```
    else do;
          year=scan(&invar,&y);
          month=scan(&invar,&m);
          day=scan(&invar,&d);
          if missing(year) or missing(month) or missing(day) or
upcase(substrn(compress(&invar),1,3)) not in ("JAN" "FEB" "MAR" "APR" "MAY"
"JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC") then do;
                &outvar=.;
          end;
          else do;
                &outvar=input(&invar,anydtdte32.);
          end;

    end;

%mend chardate;
```

Macro %check_misspartial generates a new variable “check_&invar” that reports the condition of the date. If the date is completely missing, then output “MISSING DATE”; if partially missing, then output “PARTIAL DATE”; if the date is full then output “PASS”. This new variable is helpful when we need to distinguish between fully missing and partially missing dates.

```
%macro check_misspartial;

    if missing(&invar) then check_&invar="MISSING DATE";
    else if missing(&outvar) then check_&invar="PARTIAL DATE";
    else check_&invar="PASS";

%mend check_misspartial;
```

check_testdate
PASS
PARTIAL DATE
PARTIAL DATE

Display 2. The new variable that checks the integrity of dates

Now is time to call that three macros. First of all, get the order of year, month and day into macro variables &Y, &M and &D. Secondly, if detect any alphabetic letter in date, call %chardate; if not, call %numdate. Then, call %check_misspartial to check the integrity of the incoming date.

```
%let Y=%index(&order,y);
%if &Y>0 %then %do;
%put The order of year is &Y;
%end;
%else %do;
%put Year is missing;
%end;

%let M=%index(&order,m);
%if &M>0 %then %do;
%put The order of month is &M;
%end;
%else %do;
%put Month is missing;
```

```
%end;

%let D=%index(&order,d);
%if &D>0 %then %do;
%put The order of day is &D;
%end;
%else %do;
%put Day is missing;
%end;

data &outds;
    length &outvar 8. check_&invar &invar._iso8601 month month1 day day1
year y_m_d $20.;
    set &inds;

    if anyalpha(&invar)>0 then do;
          %chardate;
    end;

    else do;
          %numdate;
    end;

    %check_misspartial;

    format &outvar date9.;
    &invar._iso8601=put(&outvar,yymmdd10.);
    drop month month1 day day1 year y_m_d;
run;
```

THE IMPUTATION

When partial date needs imputation, use similar logic to scan year, month and day into separate variables, then complete them according to the input rule.

```
%if &impute=1 %then %do;

proc sql noprint;
    select length(&invar) into: lpass
    from &outds
    where check_&invar="PASS";
quit;
%put &lpass;

data &outds;
    set &outds;
    length m_iso y_iso $10;
    if check_&invar="PARTIAL DATE" then do;
```

Date consisting solely of Arabic numerals is imputed by the following code.

```
*ONLY NUMBERS;
if anyalpha(&invar)<=0 then do;
    *WITH delimiter;
    if strip(&invar)^=compress(&invar,,'kd') then do;
          if &y=1 and &m=2 and &d=3 then do;
```

```
                year=scan(&invar,&y);
                if length(scan(&invar,&m))=1 then
month="0"||strip(scan(&invar,&m));else month=strip(scan(&invar,&m));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
          if &y=1 and &m=3 and &d=2 then do;
                year=scan(&invar,&y);
                if length(scan(&invar,&m-1))=1 then
month="0"||strip(scan(&invar,&m-1));else month=strip(scan(&invar,&m-1));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
          if &y=2 and &m=1 and &d=3 then do;
                year=scan(&invar,&y);
                if length(scan(&invar,&m))=1 then
month="0"||strip(scan(&invar,&m));else month=strip(scan(&invar,&m));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
          if &y=3 and &m=1 and &d=2 then do;
                year=scan(&invar,&y-1);
                if length(scan(&invar,&m))=1 then
month="0"||strip(scan(&invar,&m));else month=strip(scan(&invar,&m));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
          if &y=2 and &m=3 and &d=1 then do;
                year=scan(&invar,&y-1);
                if length(scan(&invar,&m-1))=1 then
month="0"||strip(scan(&invar,&m-1));else month=strip(scan(&invar,&m-1));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
          if &y=3 and &m=2 and &d=1 then do;
                year=scan(&invar,&y-1);
                if length(scan(&invar,&m-1))=1 then
month="0"||strip(scan(&invar,&m-1));else month=strip(scan(&invar,&m-1));
                m_iso=month;
                day=strip(put(&day,best.));
          end;
    end;

    *WITHOUT delimiter;
    else do;
          if length(&invar)=6 then do;
                year=substrn(&invar,1,4);

    month=substrn(&invar,5,2);m_iso=substrn(&invar,5,2);
                day=strip(put(&day,best.));
          end;
          if length(&invar)=4 then do;
                if &lpass=6 then do;
                      year=substrn(&invar,1,2);

    month=substrn(&invar,2,2);m_iso=substrn(&invar,2,2);
                      day=strip(put(&day,best.));
```

```
                end;
                if &lpass=8 then do;
                      year=strip(&invar);
                      month=scan("&month_day",1);
                      day=scan("&month_day",2);
                end;
          end;
          if length(&invar)=2 then do;
                year=strip(&invar);
                month=scan("&month_day",1);
                day=scan("&month_day",2);
          end;
    end;
end;
```

Date combined by numbers and letters is imputed as below. Whenever alphabetic letters appear in a date, they always play the role of the month. The first three letters of the month will always be there, whether it is an abbreviation or a full spell. So we only need to see if the string contains the first three letters of the twelve months, then we can judge whether the month is missing or not. Also, since the day is definitely missing, the only possible number here is the year.

```
          *NUMBERS and LETTERS;
          else do;
                year=compress(&invar,,'kd');
                if find(&invar,"JAN",'i')>0 then m_iso=month="01";
                else if find(&invar,"FEB",'i')>0 then m_iso=month="02";
                else if find(&invar,"MAR",'i')>0 then m_iso=month="03";
                else if find(&invar,"APR",'i')>0 then m_iso=month="04";
                else if find(&invar,"MAY",'i')>0 then m_iso=month="05";
                else if find(&invar,"JUN",'i')>0 then m_iso=month="06";
                else if find(&invar,"JUL",'i')>0 then m_iso=month="07";
                else if find(&invar,"AUG",'i')>0 then m_iso=month="08";
                else if find(&invar,"SEP",'i')>0 then m_iso=month="09";
                else if find(&invar,"OCT",'i')>0 then m_iso=month="10";
                else if find(&invar,"NOV",'i')>0 then m_iso=month="11";
                else if find(&invar,"DEC",'i')>0 then m_iso=month="12";
                else m_iso=month="";
                if month^="" then day=strip(put(&day,best.));
                else do;
                      month=scan("&month_day",1);
                      day=scan("&month_day",2);
                end;
          end;


    y_m_d=compress(year,,'kd')||compress(month,,'kd')||compress(day,,'kd');
          if length(y_m_d)=6 then imputed=input(y_m_d,yymmdd6.);
          if length(y_m_d)=8 then imputed=input(y_m_d,yymmdd8.);
          &outvar=imputed;
          y_iso=substrn(strip(put(&outvar,date9.)),6,4);
          if missing(m_iso) then &invar._iso8601=strip(y_iso);
          else &invar._iso8601=strip(y_iso)||"-"||strip(m_iso);
    end;

    format &outvar date9.;
    drop year month day y_m_d imputed m_iso y_iso;
run;
```

```
%end;

%else %do;

data &outds;
    set &outds;
    drop &invar._iso8601;
run;

%end;

%mend date;
```

	testdate	check_testdate	newdate	testdate_iso8601
1	1-1-2019	PASS	01JAN2019	2019-01-01
2	3-2019	PARTIAL DATE	15MAR2019	2019-03
3	2019	PARTIAL DATE	01JUL2019	2019

Display 3. The output data

CONCLUSION

This macro is used to transform character date to numeric date, impute partial date, and display date in CDISC standard (ISO8601 format). It is efficient and effortless, can handle most of common non-standard date types we encounter in different database. Nevertheless, there are restrictions of this macro. By default, a date should include year, month, and day. Partial or missing date will be notified. Time and datetime data cannot be transformed by this macro. This macro only impute 2 partial situations: 1.both month and day are missing, 2.only day is missing. Missing date will not be imputed.

FUTURE WORK

Sometimes we need to impute missing or partial date in a more complicated rule. For example, when death date is missing or partial, use following imputation rules: if year is missing (or death date completely missing), impute as the last known alive date +1; if day and month are missing, impute as January 1st of the year; if only the day is missing, use the first day of the month as the date of death; compare the imputed death date to the last known date of survival, the larger of which will be considered the date of death. This requires comparison of dates, but will be practical in analysis.

REFERENCES

[1] Carpenter, A. L. (1960, January). Looking for a Date? A Tutorial on Using SAS® Dates and Times. In *SUGI* (Vol. 30, pp. 255-30). Available at http://caloxy.com/papers/57-255-30.pdf

[2] Chiflikyan, R., Chiflikyan, M., & Medeiros, D. A Macro for Transforming Almost Any Character Calendar Date into a SAS Date Value. Available at http://www.pharmasug.org/download/papers/CC22.pdf

[3] Fahmy, A. SAS® MACROS: Offer the Best Dating Service. In NESUG 2004, Baltimore, Maryland. Available at https://www.lexjansen.com/nesug/nesug04/pm/pm04.pdf

ACKNOWLEDGMENTS

I want to thank Zhuo Chen for many valuable suggestions and help testing this program.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Hao Dong
cindyd@hmplglobal.com