

DateDiff--*More Than* a SAS® Macro

Jerry Kagan, Wyeth-Ayerst® Laboratories



Abstract

DateDiff is a SAS statement macro that can be used to return the difference between two dates in years, months, and days. Using DateDiff makes possible the arrangement of a date difference in a more humanly understandable format than simply displaying a total number of days. This macro is significant for two reasons. First, it uses a unique algorithm which has not yet been published and perhaps has never been thought of. Second, this algorithm uses an interesting mathematical technique that I have coined "Intuitive Date Arithmetic," which has many applications outside of this simple macro. DateDiff is a valuable tool with appeal in many industries. It has particular value in the Pharmaceutical and Insurance industries, where date calculation and manipulation problems are common. DateDiff takes a unique approach to a problem that was previously unsolved.

Background

At first, subtracting two dates and displaying the difference in years, months, and days would seem to be a relatively trivial problem. However, I quickly realized that it was anything but trivial. The fact that SAS has a dozen or so date functions, but none like DateDiff, was my first clue.

SAS calculates the difference between two dates in exactly the correct number of days, including leap days, simply by subtracting the dates. The problem lies in breaking this number into years, months, and days. The days position is not a problem; a day is always a day. The problem is that months and years are not always equal to the same number of days; they are rubbery units of measure. Therefore, the results of any algorithm that displays the difference between dates in years, months, and days would be muddled because months and years, by definition, are not exact units of measure.

At this point the problem may seem to be impossible, and if possible, then perhaps impractical. How is measuring a distance in time with a unit of time that is constantly changing possible, and why would anyone want to display an answer in a unit of measure that was not exact?

The answer to the second question may be a matter of taste. In some instances it may be more desirable to display a date difference as years, months and days between two dates rather than in the traditional methods: displaying a total number of days (most accurate), or years, or months with decimal fractions (least accurate). A YY,MM,DD format would be more easily understood than a large integer number of days or some fraction of a year. It may even be more accurate than displaying the difference in total number of years or total number of months with fractions as this would introduce an ambiguity because years and months vary in size.

Nevertheless, I would like to present two algorithms that attempt to solve this problem. These algorithms approach the problem in different ways and produce different answers. Each of these algorithms may be useful, depending on the application and the user's philosophy.

Mathematically Accurate Approach

The first solution that came to mind was to endorse a new programming standard for units of time. If it could be agreed that a year is equal to 365.2425 days, based on 146,097 days in a 400 year cycle, then date differences could be displayed in YY,MM,DD format without (much) ambiguity. A month would simply be a standard year divided by 12 or 30.436875 days.

While this approach seems logical, the answers that it produces do not always look correct. For example, the difference between 01/01/80 and 03/01/80 is exactly 60 days which is displayed as 1 month, 30 days. This is mathematically correct using my proposed time standard, but it does not match intuitive reasoning, which says that the difference between 01/01/80 and 03/01/80 is two months, regardless of the number of days in a month.

Intuitive Approach

The second method of computing date differences takes an intuitive approach. This method is unique in that it avoids using any standard value for a unit of time, but instead subtracts the date values directly in a way similar to integer arithmetic. After some research, it was found that this approach is completely original, which makes it even more interesting, and I have named the process "Intuitive Date Arithmetic." Remember, you heard it here first!

For an example, let's take the difference between 04/30/80 and 02/01/81. The exact difference is 277 days; the first algorithm would express this as 09 months, 03 days; the intuitive algorithm would express the answer as 09 months, 01 day. Before I discuss which answer is more correct, let me illustrate the method employed to obtain that answer.

The exact algorithm can be found in the accompanying SAS source code, but the idea is as follows. In Step 1 the problem is set up like a simple integer subtraction with the beginning date on the bottom and the ending date on the top. In this case the beginning date is smaller than the ending date, so the answer will be positive.

STEP 1

	←	Months
	←	Days
	←	Years
02/01/81	←	Ending Date
- 04/30/80	←	Beginning Date
<hr/>		
11		

In Step 2 starting with the days position, subtract the beginning day from the ending day. In this example borrowing a month from the ending month and carrying the days are necessary.

STEP 2

Subtract 1 from the Ending Month.
Carry the number of days in the Beginning Month.

```

    01 30
    02/01/81
  - 04/30/80
  -----
    |01|
  
```

Notice that the number of days carried is the number of days in the beginning month, April, not the number of days in the ending month, February. Using the days in the beginning month insures that enough days will be added to the ending day position to cover the range of days possible in the beginning month. In this example the number of days in the ending month can range from 1 to 28 while the number of days in the beginning month can range from 1 to 30. If the 28 days of February were carried, they would not be enough to cover the range of days in April and a non-acceptable negative value would result.

In Step 3, moving to the months position, subtract the beginning month from the ending month. Again, borrowing, this time from the ending year, which becomes 80, is necessary. Then, adding

12 months to the ending month results in 13 months. Subtracting 4 from 13 results in 9 months.

STEP 3

```

    12
    01 30 80 ← Subtract 1 from the Ending
    02/01/81 ← Year; carry 12 months to
  - 04/30/80 ← the Ending Month.
  -----
    09/01/
  
```

Finally, in Step 4 subtract the beginning year from the ending year to get 0 and a final answer of 09 months, 01 day.

STEP 4

```

    12
    01 30 80
    02/01/81
  - 04/30/80
  -----
    09/01/00 ← Intuitive result is 9 Months, 1 Day
  
```

Date Difference Examples					
Beginning Date	Ending Date	Difference in Days	Difference in Years	Mathematical Difference (YY,MM,DD)	Intuitive Difference (YY,MM,DD)
07/01/79	06/30/79	-1	-0.003	0, 0, -1	0, 0, -1
06/30/79	06/30/79	0	0.000	0, 0, 0	0, 0, 0
06/29/79	06/30/79	1	0.003	0, 0, 1	0, 0, 1
06/28/79	06/30/79	2	0.005	0, 0, 2	0, 0, 2
06/27/79	06/30/79	3	0.008	0, 0, 3	0, 0, 3
05/04/80	02/01/81	273	0.747	0, 8, 30	0, 8, 28
05/03/80	02/01/81	274	0.750	0, 9, 0	0, 8, 29
05/02/80	02/01/81	275	0.753	0, 9, 1	0, 8, 30
05/01/80	02/01/81	276	0.756	0, 9, 2	0, 9, 0
04/30/80	02/01/81	277	0.758	0, 9, 3	0, 9, 1
04/29/80	02/01/81	278	0.761	0, 9, 4	0, 9, 2
04/28/80	02/01/81	279	0.764	0, 9, 5	0, 9, 3
05/19/64	05/17/91	9859	26.993	26, 11, 28	26, 11, 29
05/18/64	05/17/91	9860	26.996	26, 11, 29	26, 11, 30
05/17/64	05/17/91	9861	26.999	26, 11, 30	27, 0, 0
05/16/64	05/17/91	9862	27.001	27, 0, 0	27, 0, 1
05/15/64	05/17/91	9863	27.004	27, 0, 1	27, 0, 2

Is this an by the m clear. In and 02/ myself ti day from day. TI algorithm algorithm "known"

Accuracy

Although opposed an alterr results th using eit will be i accuracy methods made pc

Displayir and unar date dif meaning of 10, 20 By using days, wh

The secc total ye drawbac a date d or great could ni differenc a range acceptat For exar 0.747 ye

Is this answer more correct than the 09 months, 03 days obtained by the mathematical version? The answer to this question is not clear. Instinctively, I feel that the difference between 05/01/80 and 02/01/81 is exactly 09 months. Intuitively, I can convince myself that if I add one day to the ending date or subtract one day from the beginning date, the answer would be 09 months, 01 day. Therefore, I can be convinced that the results of this algorithm are reasonable. In fact, all answers produced by this algorithm can be justified by extrapolating forward from any "known" difference.

Accuracy

Although I strongly endorse the intuitive version of DateDiff, as opposed to the mathematical version, which was only included as an alternative approach, neither of these algorithms produces results that can be reconverted into an exact value. Therefore, by using either of these algorithms to display date differences, one will be introducing an ambiguity and removing a degree of accuracy from reports. However, when compared to the common methods of displaying date differences, the YY,MM,DD format, made possible with DateDiff, is superior in most instances.

Displaying a total number of days is always the most accurate and unambiguous way to display a date difference. However, as date differences become larger, this format begins to lose meaning. We all have a good understanding of what a difference of 10, 20, or even 100 days means, but how long is 9,860 days? By using DateDiff, 9,860 days becomes 26 years, 11 months, 30 days, which conveys much more meaning.

The second method often used is displaying date differences in total years with a decimal fraction. This method also has drawbacks. One problem is the degree of precision. Expressing a date difference in years displayed to the thousandths' position or greater would imply a degree of precision that most likely could not be justified. On the other hand, displaying the difference to the hundredths' position would reduce precision to a range of over 3 days. For large differences this format is acceptable, but for small differences readability problems occur. For example, 26.996 years may be understandable, but what is 0.747 years or

0.003 years? With DateDiff these can be displayed as 08 months, 30 days or 01 day, respectively.

Conclusion

DateDiff is a tool with many applications in many industries. It has particular value in the Pharmaceutical and Insurance industries, where date calculation and manipulation problems are common. The technique of Integer Date Arithmetic presented here is unique and has applications outside its use in this macro. DateDiff itself is a unique approach to a problem that was previously unsolved.

The future of DateDiff looks bright. It is gaining acceptance and its use is spreading throughout my department. It will soon be rewritten as a function in C when SAS/ToolKit becomes available at my installation. I leave the final application of DateDiff up to you and will welcome any comments or suggestions.

Acknowledgements

SAS is a registered trademark of SAS Institute, Inc., Cary, NC, USA

Wyeth-Ayerst is a registered trademark of American Home Products Corp., NY, USA

Beetle Bailey reprinted with special permission of King Features Syndicate

Mr. Gary Carmon, Mr. Allan Glaser, Mr. Nathaniel Williams

Contact

Jerry Kagan
Clinical Programming
Wyeth-Ayerst Laboratories
145 King of Prussia Road
Radnor, PA 19081
(215) 341-5788

Why use DateDiff?

BEETLE BAILEY



Appendix 1 - Calling the DateDiff Macro

```

//*****
//*** EXAMPLE: Calling the DateDiff statement macro
//*****
//MACROS DD DSN=BIO.CCSTOOLS.SAS,DISP=SHR
//SYSIN DD *
* IMPLMAC option needed to call a statement macro.;
OPTIONS IMPLMAC;

* Include macro.;
%INCLUDE MACROS(DATEDIFF);

DATA DATES (DROP=START INTERVAL);

FORMAT  BEG_DATE  MMDDYY8.
        END_DATE  MMDDYY8.;

START    = MDY(06,27,1979);*** Begin at this date. ***;
END_DATE = MDY(06,30,1979); *** End at this date. ***;
INTERVAL = 1;                *** Increment start date. ***;
DO BEG_DATE = START
  TO END_DATE
  BY INTERVAL;

* Calculate the MATHEMATICAL difference. ;
DATEDIFF  TYPE      = MATH
          BEG_DATE  = BEG_DATE
          END_DATE  = END_DATE
          DAYS      = M_DAYS
          MONTHS    = M_MONTHS
          YEARS     = M_YEARS
          ;

* Calculate the INTUITIVE difference. ;
DATEDIFF  TYPE      = INTU
          BEG_DATE  = BEG_DATE
          END_DATE  = END_DATE
          DAYS      = I_DAYS
          MONTHS    = I_MONTHS
          YEARS     = I_YEARS
          ;

* Calculate the exact difference in days. ;
DIFF_DAY  = END_DATE - BEG_DATE;

* Calculate the difference in average years. ;
DIFF_YR   = DIFF_DAY / 365.2425;

OUTPUT;
END;

* Do a quick print of the results. ;
PROC PRINT;
/*
//

```

Appendix 2 - DateDiff Source Code

```

%*** -----
%*** Ma
%***
%*** Ty
%*** Be
%*** Er
%*** D:
%*** M
%*** Y:
%*** --

%macro

%***
%*** LIBRARY:  BIO.CCSTOOLS.SAS
%***
%*** NAME:     DateDiff - Date Difference
%*** LANGUAGE: SAS MACRO
%***
%*** PURPOSE:
%*** This SAS macro will return the difference between two
%*** dates in days, months, and years.  Beg_Date is the
%*** beginning date and must be specified.  End_Date is the
%*** ending date and will default to the system date if not
%*** specified.  The difference between the two dates will be
%*** returned in three variables: Days, Months, and Years.
%***
%*** NOTES:
%*** This macro can calculate date differences using one of
%*** two algorithms, intuitive or mathematical.  Specify a
%*** Type of INTU or MATH.  If Type is not specified, the
%*** macro will default to the intuitive algorithm.
%***
%*** TYPE = MATH
%*** The result will be mathematically accurate but not
%*** always intuitively correct.  For example, the difference
%*** between 1/1/80 and 3/1/80 will result in 1 month and
%*** 30 days, not 2 months which may seem more
%*** reasonable.  This is due to using an average year of
%*** 365.2425 days, which is based on 146,097 days in a
%*** 400 year cycle.
%***
%*** TYPE = INTU
%*** The result will always be intuitively correct but may not
%*** always be mathematically accurate.  For example, the
%*** difference between 1/1/80 and 3/1/80 will result in
%*** exactly 2 months.
%***
%*** PROTOCOL:
%*** DATEDIFF < TYPE      = <INTU> | <MATH> >
%***          BEG_DATE  = SAS_Beginning_Date
%***          < END_DATE = SAS_Ending_Date >
%***          < DAYS    = Days_Place_Holder >
%***          < MONTHS  = Months_Place_Holder >
%***          < YEARS   = Years_Place_Holder >
%***
%***
%*** EXAMPLE(s):
%*** DATEDIFF BEG_DATE = MDY(5,01,64);
%***
%*** DATEDIFF TYPE      = MATH
%***          BEG_DATE  = MDY(5,1,64)
%***          END_DATE  = MDY(1,1,90)
%***          DAYS      = NUM_DAY
%***          MONTHS    = NUM_MOS
%***          YEARS     = NUM_YRS
%***
%***
%*** AUTHOR NAME:  Jerry Kagan
%*** DATE CREATED: January 1991
%***
%*** REVIEWED BY:  Nate Williams
%***              Allan Glaser
%***
%*** -----

```

```

%*** -----;
%*** Macro Parameter Definitions:
%***
%*** Type - Must be MATH or INTU, defaults to INTU.
%*** Beg_Date - BEGinning DATE, must be specified.
%*** End_Date - ENDing DATE, defaults to system date.
%*** Days - Number of DAYS, defaults to DAYS.
%*** Months - Number of MONTHS, defaults to MONTHS.
%*** Years - Number of YEARS, defaults to YEARS.
%*** -----;

```

```

%macro DateDiff ( Type = INTU
, Beg_Date =
, End_Date = Today()
, Days = Days
, Months = Months
, Years = Years
) / STMT;

```

```

%*** -----;
%*** If Beg_Date is not specified, print an error message.
%*** -----;

```

```

%if %length(%quote(&Beg_Date)) = 0 %then
%do;

%put ERROR: Missing Beg_Date in macro DateDiff;

%end;

```

```

%*** -----;
%*** -----;
%*** If Type = MATH then subtract BegDate from EndDate
%*** using the Mathematical algorithm, and an average year of
%*** 365.2425 days.
%*** -----;
%*** -----;

```

```

%else %if %upcase( %substr( &Type,1,4 ) ) = MATH %then
%do;
%put NOTE: Mathematical algorithm used to calculate date
difference.;

```

```

%*** -----;
%*** ___Diff is a SAS date value representing the difference
%*** between the two dates in exactly the correct number of
%*** days.
%*** -----;

```

```

___Diff = ( &End_Date - &Beg_Date );

```

```

*** Calculate &Years using an average year...;

```

```

&Years = int( ___Diff/365.2425 );

```

```

*** Calculate &Months using an average year/12...;

```

```

&Months = int( ___Diff/30.436875 -
&Years * 12 );

```

```

*** Subtracting &Months and &Years results in &Days...;

```

```

&Days = round( ___Diff -

```

```

&Years * 365.2425 -
&Months * 30.436875 );

```

```

%*** -----;
%*** Now that the difference between the dates is represented
%*** in days, months, and years, drop the temporary variable
%*** ___Diff.
%*** -----;

```

```

drop ___Diff;
%end;

```

```

%*** -----;
%*** -----;
%*** If Type was not set to MATH, then subtract BegDate from
%*** EndDate using the Intuitive algorithm, which uses "Date
%*** Arithmetic" to calculate the result.
%*** -----;
%*** -----;

```

```

%else %do;
%put NOTE: Intuitive algorithm used to calculate date
difference.;

```

```

%*** -----;
%*** If Beg_Date is greater than End_Date, swap them and set
%*** ___Sign.
%*** -----;

```

```

if &Beg_Date > &End_Date then
do;
___Sign = &Beg_Date;
&Beg_Date = &End_Date;
&End_Date = ___Sign;
___Sign = -1;
end;
else ___Sign = 1;

```

```

%*** -----;
%*** Breakup each date into days, months and years.
%*** -----;

```

```

___BDay = day( &Beg_Date );
___BMonth = month( &Beg_Date );
___BYear = year( &Beg_Date );

___EDay = day( &End_Date );
___EMonth = month( &End_Date );
___EYear = year( &End_Date );

```

```

%*** -----;
%*** Starting with the days position, subtract ___BDay from
%*** ___EDay. If ___EDay is smaller than ___BDay, borrow from
%*** the months position.
%*** -----;

```

```

%*** Note: The number of days to carry should be the
%*** number of days in the month of ___BMonth, not the
%*** number of days in the month of ___EMonth. This is
%*** calculated by converting day 1 of the month after
%*** ___BMonth into a SAS date using the mdy function. Then
%*** subtracting 1 from this SAS date results in a SAS date
%*** number for the last day of the month of ___BMonth.

```

```

%*** Using the day function then returns the number of days
%*** in the month of __BMonth, which is the correct number
%*** of days to carry.
%*** -----
if __EDay < __BDay then
do;

    *** Borrow a month from __EMonth and carry the days...;

    if __EMonth = 1 then
    do;
        __EYear + (- 1);
        __EMonth = 12;
    end;
    else
        __EMonth + (- 1);

    *** Calculate the number of carry days...;

    if __BMonth = 12 then
    do;
        __TempYr = __BYear + 1;
        __TempMo = 1;
    end;
    else do;
        __TempYr = __BYear;
        __TempMo = __BMonth + 1;
    end;

    *** Add carry days to __EDay...;

    __EDay + day( mdy( __TempMo, 1, __TempYr) - 1);

end;

*** Now calculate days position...;

&Days = ( __EDay + (- __BDay) ) * __Sign;

%*** -----
%*** Moving to the months position, subtract __BMonth from
%*** __EMonth. If __EMonth is smaller than __BMonth,
%*** borrow from the years position.
%*** -----
if __EMonth < __BMonth then
do;
    __EYear + (- 1);
    __EMonth + 12;
end;

*** Now calculate months position...;

&Months = ( __EMonth - __BMonth ) * __Sign;

%*** -----
%*** Finally, subtract __BYear from __EYear to calculate the
%*** years position.
%*** -----
&Years = ( __EYear - __BYear ) * __Sign;

%*** -----

```

```

%*** If __Sign is negative, swap &Beg_Date and &End_Date.
%*** -----
if __Sign < 0 then
do;
    __Sign = &Beg_Date;
    &Beg_Date = &End_Date;
    &End_Date = __Sign;
end;

%*** -----
%*** Drop all temporary variables.
%*** -----
drop __BDay __BMonth __BYear
    __EDay __EMonth __EYear
    __TempMo __TempYr __Sign
;

%end;
%mend DateDiff;

```

INT
Wh
you
and
and
ser
typi
the
Ma
mu
cor
tor
use
ma
hav
Th
mu
set
Wr
be:
Dic
Dic
Dic
Ch
tir
se
Th
As
we
of
co
bo
of
O
co
(P
up
re
va
er
By
ct
Si
ct
sp
st
as