SIGNIF, A C Function to Compute Significant Digits

B.M. Horwedel, Martin Marietta Energy Systems
J.E. Horwedel, Martin Marietta Energy Systems

ABSTRACT

This poster presents a C function, SIGNIF, that can be called from a SAS® program to return a value rounded to a specified number of significant digits. The rules for rounding a number to n significant digits as implemented by SIGNIF are:

1. Truncate the number to n digits, and treat the excess digits as a decimal fraction;
2. If the fraction is greater than 1/2, increment the least significant digit;
3. If the fraction is less than 1/2, do not increment;
4. If the fraction equals 1/2, increment the least significant digit only if it is odd.

INTRODUCTION

The number of significant figures refers to the number of numerical digits that is used to express the value. An appropriate number of digits is that which is warranted by the accuracy of the reported value. In general, measurement data seldom justify more than two or three significant figures for the value. In reporting generalized statistics such as mean or standard error, tables listing these statistics with more than two or three significant digits are misleading and imply more accuracy than that found in the original measurements.

DISCUSSION

Significant figures and round-off are discussed in Ref. 1.

The pseudocode for rounding a number to n significant digits is as follows:

MOVE selected number of significant digits to left of decimal point.
Set VALUE equal to truncated decimal.
If fractional part is GREATER than 0.5 increment VALUE.
Else, if fractional part EQUALS 0.5 then if VALUE is ODD Increment VALUE.

SIGNIF is used like any other SAS function and requires two arguments. The first argument is the number to be rounded; the second argument specifies the number of significant digits for the return value. Consider the following example:

\[ X = 8.234; \]
\[ Y = \text{SIGNIF}(X,2); \]

\( Y \) will be set to the value 8.2.

Details for compiling, linking and using a C function in a SAS program are provided in Ref. 2.

The SAS function ROUND does provide some of the capabilities of SIGNIF; however, ROUND is more limited. The ROUND function rounds a value to the nearest roundoff unit which must be supplied by the user. It is difficult to specify one roundoff unit for data with a large range. SIGNIF is designed to work with ranges of numbers.

RESTRICTIONS and LIMITATIONS

SIGNIF was implemented and tested on a VAX® computer under the VMS® environment. Due to precision, the maximum number of significant digits is limited to six. The approximate range of values tested was from ±E-37 through ±E+38.

CONCLUSION

SIGNIF is a C function callable from a SAS program to round a value to a specified number of digits. SIGNIF allows reporting statistics on measurement data limited to the number of significant digits warranted by the accuracy of the original measurements without the limitations imposed by the SAS ROUND function.

REFERENCES


The submitted manuscript has been authored by a contractor of the U.S. Government under contract No. DE-AC05-84OR21400. Accordingly, the U.S. Government retains a nonexclusive royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.
Appendix A. Listing of SIGNIF.C.

/* Function: SIGNIF (a, sig) */

* Purpose: SIGNIF returns the value of the first argument rounded to the number of significant digits specified by the second argument.

* Authors: B. M. Horwedel, J. E. Horwedel
  Computing and Telecommunications Division
  Martin Marietta Energy Systems
  Oak Ridge National Laboratory
  Building 6011, MS 6370
  P.O. Box 2008
  Oak Ridge, TN 37831-6370
  Phone numbers: (615)-574-5413
  (FTS)-624-5413

* Created: December 12, 1989

* Arguments:
  a value to be rounded; type double;
  approximate range: + or - (e-38 to e+38)

  sig required number of significant digits; type integer;
  range: 0 < sig < 7

* Notes: SIGNIF is a C function that is used in a SAS program to return a value with a specified number of significant digits. The method is to truncate the number to the specified number of digits. Treat the excess as a fraction. If the fraction is greater than 0.5, increment the number. If the fraction is equal to 0.5, increment the number only if it is odd. For further discussion see Reference 1.

* Making a C function callable from SAS is discussed in Reference 2.

* References:

* Restrictions: Both SAS and C must be available on the VAX/VMS for implementation of SIGNIF.

* Included Files:
  * func.h available on VAX/VMS computer

* sasinterface.h The contents of this file are discussed in Ref. 2. The file is attached to this listing.

* Pseudo code:
  * Determine factor to move sig digits to left of decimal.
  * Use factor to scale up (or down).
  * Get integral value and fractional part.
    * If fractional part is greater than 0.5 increment value;
    * else if fractional part equals 0.5 increment if value is odd.
    * Scale down (or up) with factor.
    * return value

*/
#include "func.h"
long signif_init ( dcl )
#include "sasinterface.h"

double a, sig, factor, c, dif, neg;
double pow(), ceil(), log10();
int ival;

/* Get arguments through SAS interface */
parms = *parmptr;
a = *(( double *) ( parms[ 0 ].PARMPTR ));
sig = *(( double *) ( parms[ 1 ].PARMPTR ));
c = a;

/* Algorithm requires positive value */
neg=1.;
if ( c < 0 ) neg= -t.;
c= neg*c;

/* Determine scaling factor and scale up (or down) */
factor = pow(10., sig-ceil(log10(c)));
c = c*factor;

/* Get integral and fractional parts */
ival=c;
dif=c-ival;

/* Increment based on rules. Note values are not set at exactly 0.5 to avoid precision problems. */
if(dif > 0.5001000000000) ival++;
else if(dif > 0.4999999) if ( ival%2 == 1) ival++;

/* Convert back to real and scale down (or up) */
c=ival;
c=neg*c/factor;
*(( double *) ( parms[ 2 ].PARMPTR )) = c;
parms[ 2 ].PARMLEN = 0;
return 0;
}
Appendix B. Listing of SASINTERFACE.H.
/* File: SASINTERFACE.H

Purpose: Provides definitions of variables and pointers necessary to interface SIGNIF with a user's SAS program. The specifics are discussed in Ref. 2 (see SIGNIF.C).
*/
FUNC_DCL_PTR *dcl;
{
    register FUNC_DCL_PTR p;
    long signif();
    p = *dcl;
    strncpy( p-> FUNC_NAME, "signif", 8);
    strncpy( p-> FUNC_ALIAS,*signif ,8);
    p-> FUNC_ADDR = signif;
    p-> FUNC_MIN_ARG = 2;
    p-> FUNC_MAX_ARG = 2;
    p-> FUNC_ARG_TYPE = NV( 1) | NV( 2) | NV( 3);
    p-> FUNC_RETURN_LENGTH = 0;
    return 0;
}
long signif( nargsptr, parmptr )
    long * nargsptr;
    PARMS_PTR * parmptr;
{
    PARMS_PTR parms;
}