
Subject: Bug in WAVE/WAVE Advantage Routine SUM

Posted by [black](#) on Tue, 01 Feb 1994 18:05:58 GMT

[View Forum Message](#) <> [Reply to Message](#)

We've just got WAVE v4.20 & WAVE Advantage v1.0 & I've spotted the VNI people have still not fixed a bug in the routine SUM. (The bug might exist in IDL as well).

ROUTINE: SUM

BUG : Partial failure for large output arrays, whose total number of elements exceeds 32767 ($2^{15} - 1$) [excuse the TeX notation]

CAUSE : USE OF INTEGERS (THESE ONLY HAVE 16 BITS!!!) & FIX FUNCTION

EXAMPLE: type in the following to a PV-WAVE session

```
A = LINGEN(2,50,1000)
```

```
B = SUM(A,0)
```

```
PRINT, B(*,999)
```

this produces zero's, when it shouldn't. If B is probed further the zero's begin on the 32,768th element, considering the array as a one dimensional one

I've E-mailed VNI, so they know about it

Here's some patched code, which I've called MY_SUM

```
=====
=====
FUNCTION MY_SUM,ARRAY,DIMENSION
;
;
; $Id: my_sum.pro, v1.0 91/09/04 15:38:00 wave Exp $
;
;
; NAME:
; SUM
; PURPOSE:
; Total up an array over one of its dimensions.
; HISTORY
; MODIFIED from SUM 91/09/04 by John Black
; This is a bug fix of the supplied routine SUM, which fails in some
; circumstances due to 16 bit arithmetic relating to array indices. The
; main bug I suspect of the original routine is the line
;
;
; XK = FIX( XIK / NI )
;
;
; here I've replaced the 'FIX' by a 'LONG'. I've also made a lot of the
; numbers involved in the for loops long as well for extra safety, but
; these might not be needed.
;
;
;
```

```

IF (N_PARAMS() LT 2) THEN BEGIN
  PRINT, '*** Function SUM must be called with two parameters:'
  PRINT, '          ARRAY , DIMENSION'
  RETURN, ARRAY
ENDIF
;
S = SIZE(ARRAY)
N_DIM = S(0)
IF N_DIM EQ 0 THEN BEGIN
  PRINT, '*** Variable must be an array, name= ARRAY, routine SUM.'
  RETURN, ARRAY
END ELSE IF (DIMENSION GE N_DIM) OR (DIMENSION LT 0) THEN BEGIN
  PRINT, '*** Dimension out of range, name= ARRAY, routine SUM.'
  RETURN, ARRAY
END ELSE IF N_DIM EQ 1 THEN BEGIN ;Trivial case, equivalent to TOTAL.
  F = TOTAL(ARRAY)
  RETURN, F
ENDIF
;
S2 = S(1+WHERE(INDGEN(N_DIM) NE FIX(DIMENSION)));Set up array for output variable.
F = MAKE_ARRAY(DIMENSION=S2, TYPE=S(S(0)+1))
;
; Calculate product of sizes of dimensions lower than, equal to, and higher
; than DIMENSION (NI,NJ,NK respectively).
;
NI = 1L ; Make sure that NI is a long integer and result from using it is too
IF DIMENSION GT 0 THEN FOR M = 1,DIMENSION DO NI = NI * S(M)
NJ = S(DIMENSION+1)
NK = 1L ; Make sure that NK is a long integer and result from it is too
IF DIMENSION LT N_DIM-1 THEN FOR M = DIMENSION+2,N_DIM DO NK = NK * S(M)
;
; Set up index arrays.
;
XIK = LINDGEN( NI * NK )
XJ = LINDGEN( NJ )
NIJ = NI*NJ
;
; Choose whether it is more efficient to loop over NI and NK ...
;
IF NI*NK LT NJ THEN BEGIN
  FOR I = 0L,NI-1 DO FOR K = 0L,NK-1 DO $
    F(I+NI*K) = TOTAL( ARRAY(I + NI*XJ + NIJ*K) )
  ;
  ; ... or over NJ.
  ;
END ELSE BEGIN
  XI = XIK MOD NI
  XK = LONG(XIK / NI) ;replaces  XK = FIX( XIK / NI )

```

```
FOR J = 0L,NJ-1 DO F = F + ARRAY(XI + NI*J + NIJ*XK)
ENDELSE
RETURN,F
END
```
