Subject: Re: Inconstant log(-1) handling
Posted by dit on Tue, 27 Apr 1993 20:19:23 GMT
View Forum Message <> Reply to Message

In article <C64BH2.Ex0@murdoch.acc.Virginia.EDU>,
gsh7w@fermi.clas.Virginia.EDU (Greg Hennessy) writes:

> I strongly get the idea that this is one of those "Doctor, it hurts
> when I do this." "Don't do that." type posts, but here it goes. In IDL
> v3.0.0 on a sparcstation 10, running 4.0.3b there seems to be an
> inconsistency with how logarithms of negative numbers are done. Don't
> sugguest that I shouldn't be doing this in the first place, I know I
> shouldn't, however what IDL is giving me seems a tad bit weird. I do
> know that if I do a
> IDL>tmp=check_math(trap=0) & delvar,tmp
> in my startupfile, IDL returns NaN's for the illegal values, but it
> does not seem to do so by default.
>
> Greg Hennessy
>
>
>> idl
> IDL. Version 3.0.0 (sunos sparc).
> Copyright 1989-1992, Research Systems, Inc.
> All rights reserved. Unauthorized reproduction prohibited.
> Site: 2762.
> Licensed for use by: UVA (Perseus)
>
> % Compiled module: CINIT.
> SUNIDL>x=findgen(6)-3
> SUNIDL>print,x
> -3.00000 -2.00000 -1.00000 0.00000 1.00000 2.00000
> SUNIDL>print,alog10(x)
> % Program caused arithmetic error: Floating illegal operand
> % Detected at $MAIN$ (ALOG10).
> % Program caused arithmetic error: Floating divide by 0
> % Detected at $MAIN$ (ALOG10).
> 200000. 200000. 200000. 0.00000 0.00000 0.301030
> SUNIDL>print,alog10(x)
> % Program caused arithmetic error: Floating illegal operand
> % Detected at $MAIN$ (ALOG10).
> % Program caused arithmetic error: Floating divide by 0
> % Detected at $MAIN$ (ALOG10).
> 1.00000 1.00000 1.00000 0.00000 0.00000 0.301030
> SUNIDL>
>

That's what I've just seen running this test on a VAX under OpenVMS 5.5-2

with no Check_Math set (IDL's default):

```
; IDL Version 3.0.0 (vms vax)
; Journal File for VAXSER::SYSTEM
; Working directory: SYS$SYSROOT:[SYSMGR]
; Date: Tue Apr 27 22:03:48 1993

x=findgen(6)-3
print,x
;     -3.00000     -2.00000     -1.00000     0.000000      1.00000      2.00000
print, alog10(x)
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
;     0.000000     0.000000     0.000000     0.000000     0.000000     0.301030
print, alog10(x)
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
; % Program caused arithmetic error: Logarithm of 0 or negative
;     0.000000     0.000000     0.000000     0.000000     0.000000     0.301030
```

It seems to be, there are 'some differences' between SUN- and VAX-IDL.
Hope this helps.

Karl-Heinz.

Subject: Re: Inconstant log(-1) handling
Posted by thompson on Thu, 29 Apr 1993 15:02:06 GMT
View Forum Message <> Reply to Message

dit@vaxser.grumed.fu-berlin.de (K.-H. Dittberner) writes:

> In article <C64BH2.Ex0@murdoch.acc.Virginia.EDU>,
> gsh7w@fermi.clas.Virginia.EDU (Greg Hennessy) writes:

>> I strongly get the idea that this is one of those "Doctor, it hurts
>> when I do this." "Don't do that." type posts, but here it goes. In IDL
>> v3.0.0 on a sparcstation 10, running 4.0.3b there seems to be an
>> inconsistency with how logarithms of negative numbers are done. Don't
>> sugguest that I shouldn't be doing this in the first place, I know I
>> shouldn't, however what IDL is giving me seems a tad bit weird. I do
>> know that if I do a
>> IDL>tmp=check_math(trap=0) & delvar,tmp
>> in my startupfile, IDL returns NaN's for the illegal values, but it
>> does not seem to do so by default.

>>
>> Greg Hennessy
>>
>>
>>> idl
>> IDL. Version 3.0.0 (sunos sparc).
>> Copyright 1989-1992, Research Systems, Inc.
>> All rights reserved. Unauthorized reproduction prohibited.
>> Site: 2762.
>> Licensed for use by: UVA (Perseus)
>>
>> % Compiled module: CINIT.
>> SUNIDL>x=findgen(6)-3
>> SUNIDL>print,x
>>   -3.00000   -2.00000   -1.00000   0.00000   1.00000   2.00000
>> SUNIDL>print,alog10(x)
>> % Program caused arithmetic error: Floating illegal operand
>> % Detected at $MAIN$ (ALOG10).
>> % Program caused arithmetic error: Floating divide by 0
>> % Detected at $MAIN$ (ALOG10).
>>   200000.   200000.   200000.   0.00000   0.00000   0.301030
>> SUNIDL>print,alog10(x)
>> % Program caused arithmetic error: Floating illegal operand
>> % Detected at $MAIN$ (ALOG10).
>> % Program caused arithmetic error: Floating divide by 0
>> % Detected at $MAIN$ (ALOG10).
>>   1.00000   1.00000   1.00000   0.00000   0.00000   0.301030
>> SUNIDL>
>>

> That's what I've just seen running this test on a VAX under OpenVMS 5.5-2
> with no Check_Math set (IDL's default):

> ; IDL Version 3.0.0 (vms vax)
> ; Journal File for VAXSER::SYSTEM
> ; Working directory: SYS$SYSROOT:[SYSMGR]
> ; Date: Tue Apr 27 22:03:48 1993
>
> x=findgen(6)-3
> print,x
> ;   -3.00000   -2.00000   -1.00000   0.000000   1.00000   2.00000
> print, alog10(x)
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ;   0.000000   0.000000   0.000000   0.000000   0.000000   0.301030
> print, alog10(x)

> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; % Program caused arithmetic error: Logarithm of 0 or negative
> ; 0.000000 0.000000 0.000000 0.000000 0.000000 0.301030

> It seems to be, there are 'some differences' between SUN- and VAX-IDL.
> Hope this helps.

> Karl-Heinz.

It's probably not IDL itself, but just differences between the way the
operating systems themselves handle numerical errors.

In my opinion the statement at the top of the original message is right--you
shouldn't do that. At best the results are "unpredictable".

Bill Thompson
