Subject: Re: Philosophical Question about NAN
Posted by pgrigis on Tue, 18 Nov 2008 16:42:44 GMT
View Forum Message <> Reply to Message

Reimar Bauer wrote:
> Paolo schrieb:
>> On the other hand,
>> NAN works much better than fixed values for
>> plots! (for instance, if nan=!values.f_nan
>> a=[1.0,2,nan,4,2]
>> will give a much better plot than if nan=-999,
>> even if one has a good yrange).
>>
>> Ciao,
>> Paolo
>
>
> the same is true for Inf values

Well, when I need to plot data with missing
values, I put in NANs in my array. If I wouldn't,
I would have to loop over the valid data chuncks
to do a nice plot...now, we don't want to do that,
do we? So I hold on to my point...

Ciao,
Paolo

>
> inf = 1.0 / 0
> a = [1.0, 2, inf, 4,2]
> plot, a
>
> print, finite(a)
> 1 1 0 1 1
>
> Just something is possible it does not make it automatically a great
> solution.
>
>
> Reimar
>
>
>>
>> Reimar Bauer wrote:
>>> Sometimes I wish people would use a defined missing value instead on
>>> NaN. NaN is only defined for float and double.

>>> If a NaN value is in you data everything can become difficult.
>>>
>>> IDL> a=[!values.f_nan,0,3,5]
>>> IDL> print,max(a)
>>>   NaN
>>> IDL> print,min(a)
>>>   NaN
>>> IDL> if a[0] gt 1 then print, 'yes' else print, 'no'
>>> no
>>> IDL> if a[0] lt 1 then print, 'yes' else print, 'no'
>>> no
>>> IDL> if a[0] eq 1 then print, 'yes' else print, 'no'
>>> no
>>>
>>> if you have read until here you may wonder about this
>>> IDL> if !values.f_nan eq !values.f_nan then print,'yes' else print, 'no'
>>> no
>>>
>>> Idl says "no"!!
>>>
>>> For functions we can easily set a key so that NaN numbers can be handled
>>> differently but if the default is to search for NaN a lot of other
>>> places needs a lot of changes.
>>>
>>> cheers
>>>
>>> Reimar
>>>
>>>
>>> Kenneth P. Bowman schrieb:
>>>> In article <MPG.238b3491ef337cc798a534@news.giganews.com>,
>>>>   David Fanning <news@dfanning.com> wrote:
>>>>
>>>> > Folks,
>>>> >
>>>> > I've had a couple of run-ins lately with NANs and I wonder
>>>> > why routines like TOTAL and MEAN don't have the NAN keyword
>>>> > set to 1 by default. Why does the user have to set it?
>>>> >
>>>> > I understand the argument that the NAN capability was
>>>> > added as an afterthought (or more likely when someone
>>>> > standardized the NAN bit pattern), and so the functionality
>>>> > was added as an optional addition that enhanced the function
>>>> > rather than changed it. But really...is there a reason
>>>> > why it is not the default now?
>>>> >
>>>> > One could argue, I suppose, that having a program stumble
>>>> > over a NAN alerts you to its presence in your data. That

>>>> > is useful, certainly. But, typically, once I add a NAN
>>>> > keyword to my code, I don't know (nor do I or care) if the
>>>> > argument has NANs. Is this lazy programming on my part?
>>>> >
>>>> > I am just wondering whether not setting the default value
>>>> > of the NAN keyword to 1 on routines like TOTAL, MEAN,
>>>> > et. al is the functional equivalent of not setting the
>>>> > default values of the COLOR and BITS_PER_PIXEL keywords
>>>> > to the PostScript device to something useful by default.
>>>> > That is, an act of negligence on the part of the
>>>> > manufacturer.
>>>> >
>>>> > What say you?
>>>> >
>>>> > Cheers,
>>>> >
>>>> > David
>>>> HI David,
>>>>
>>>> I think they chose correctly and erred on the side of safety.
>>>>
>>>> If I know there are Nans in my data, I'll take care of it.
>>>>
>>>> If there are Nans in the data that I don't expect, I don't want to
>>>> have to set a keyword somewhere to find that out. That is, I don't
>>>> want IDL to automatically skip those Nans.
>>>>
>>>> OTOH, I still find this to be frustrating and dangerous
>>>>
>>>> IDL> PRINT, TOTAL(REPLICATE(!VALUES.F_NAN, 5), /NAN)
>>>>   0.00000
>>>>
>>>> There are no valid numbers in the input vector, but TOTAL
>>>> returns a valid FLOAT. This makes the NAN keyword useless
>>>> in many situations.
>>>>
>>>> Ken
