Subject: IDL 4.0.1, best way to deal with missing/bad data
Posted by rfinch on Fri, 09 Feb 1996 08:00:00 GMT

View Forum Message <> Reply to Message

IDL 4.0.1, Solaris.

We are using a database (HECDSS) connected to a system of IDL routines
to view and manipulate time-series data. The database had special
values to indicate missing data, as well as the ability to store a
companion 32-bit word in which bits are set to indicate different
types of data (screened, good, reject, questionable, missing, ...).

The question comes as to the best way to handle missing/bad data
within IDL. By handle I mean don't use the data in computations, and
don't plot it. I can think of three ways:

- use the max_value keyword along with my own special, large number to
  indicate bad/missing data
  Problem: Not all routines use this, so it's not a universal solution.

- use the IEEE NAN to indicate the unwanted data
  Problem: to avoid bogus calcs you have to use the Finite function,
  an annoyance to put into every computation (we have hundreds), plus
  presumably things would run slower with the Finit function.

- use the IEEE INF to indicate the unwanted data
  Problem: what does plot do when it hits this? The docs hint that
  calcs don't blow up on this like NAN, is that true in every case?

Any ideas which is 'best', overall?
--
"Nada burra la chamaca." A.G.
Opinions expressed are mine, not my employer or news host.
rfinch@toe.cs.berkeley.edu

Subject: Re: IDL 4.0.1, best way to deal with missing/bad data
Posted by thompson on Sat, 17 Feb 1996 08:00:00 GMT

View Forum Message <> Reply to Message

rfinch@toe.CS.Berkeley.EDU (Ralph Finch) writes:

(stuff deleted)

> I've talked to RSI about this problem; they think that the next
> release, all computational routines will recognize missing data with
> the /NAN keyword, so all you have to do is replace your missing values
> with NANs. For now I guess I will use the following construct:

The trouble with using NaN values is that not all computers use IEEE floating
point notation. Specifically, I'm talking about the VAX floating point
notation used in VMS, which is still an extremely important platform to us.
(As far as I'm aware, that may be the only exception among modern computers.)

Perhaps RSI has figured out a way around this difficulty.

Bill Thompson

Subject: Re: IDL 4.0.1, best way to deal with missing/bad data
Posted by rfinch on Mon, 19 Feb 1996 08:00:00 GMT

View Forum Message <> Reply to Message

>>>> > "Bill" == William Thompson <thompson@orpheus.nascom.nasa.gov> writes:

Bill> rfinch@toe.CS.Berkeley.EDU (Ralph Finch) writes:

Bill> The trouble with using NaN values is that not all computers use
Bill> IEEE floating point notation. Specifically, I'm talking about
Bill> the VAX floating point notation used in VMS, which is still an
Bill> extremely important platform to us. (As far as I'm aware, that
Bill> may be the only exception among modern computers.)

Bill> Perhaps RSI has figured out a way around this difficulty.

What I'd really like is to have a MISS keyword, so you could define
the missing value yourself; MISS=-901.0 in my case. Every
computational and plot routine would understand this keyword.
Computations would ignore those values, plots would skip them.
--
"Nada burra la chamaca." A.G.
Opinions expressed are mine, not my employer or news host.
rfinch@toe.cs.berkeley.edu

Subject: Re: IDL 4.0.1, best way to deal with missing/bad data
Posted by f055 on Thu, 22 Feb 1996 08:00:00 GMT

View Forum Message <> Reply to Message

->>>>> "Bill" == William Thompson <thompson@orpheus.nascom.nasa.gov> writes:
Bill> The trouble with using NaN values is .... <cut>

Another problem with NaN is (I've just discovered to my cost) that there's
no way to represent it in integer or long variables - only double or floats.

I've been using NaN since upgrading to IDL4, and have just realised an error
in a couple of my programs: I read in a dataset, set all missing data to
!values.f_nan. Then, to replicate someone else's results who did some
analysis with a precision of 1 decimal place, I rounded all my values to
1 d.p. with:

```
fd = float( round( fd*10. ) ) / 10.
```

The round, of course, altered all values to integer and set all !values.f_nan
to zeros, which I then converted back to floats. I never noticed, oops.
If I'd known, I could've kept a copy of the original fd and used that to
remask the new fd where appropriate. But even so, I'm sure some applications
would want to use data as ints or longs with some kind of missing code.

```
......................... Dr Tim Osborn                  . t.osborn@uea.ac.uk
.... ___/.. __ /.. /.. /. Senior Research Associate       . phone:01603 592089
... /..... /. /.. /.. /.. Climatic Research Unit          . fax:  01603 507784
.. /.....  __/.. /.. /... School of Environmental Sciences.
. /..... /\ ... /.. /.... University of East Anglia       .
____/.._/..\_..____/..... Norwich  NR4 7TJ                .
......................... UK                              .
```