Subject: Re: Using where() on slices of data cubes
Posted by Paul Van Delst[1] on Tue, 20 Oct 2009 15:00:32 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Conor writes:
>
>> I feel like this should be an easy one, but I've never quite figured
>> it out. Let's say I got a data cube and I want to do something on
>> just a slice of it, say I want to turn certain values in a column into
>> something else:
>>
>> w = where(cube[1,*,*] lt 0)
>>
>> It seems like you should be able to do something like this:
>>
>> cube[1,w] = 1e24
>>
>> But that doesn't work... Somehow I can't quite figure out the right
>> way to do this.
>
> It *seems* like there should be a simple way to do this,
> but if there is, I haven't found it. What would make
> sense to me is this:
>
> (cube[1,*,*])[w] = 1e24
>
> But this gives the error message that you can't store into
> a temporary variable. (The ol' pass by reference/pass by value
> thing, I suppose.)

Is it possible at all to use IDL pointer for aliasing? E.g. in Fortran I would do something like the following

```
real, target  :: cube(:,:,:)
real, pointer :: alias(:,:) => null()

alias => cube(1,:,:)

where ( alias < 0.0 )
  alias = 1e24
end where

nullify(alias)
```

Is there an IDL equivalent to the "alias => cube(1,:,:)" statement in Fortran?

If there is, then that might get around the tired old "can't store into a temporary variable" issue that plagues IDL due to it sticking to pass by reference only. (Man, I wish they would fix that.)

The only way I can figure out to do the aliasing ends up copying the data.

Anyway...

cheers,

paulv
