Subject: Re: Generally accessing the rest of the elements in an array
Posted by thompson on Wed, 21 Feb 2001 00:12:20 GMT
View Forum Message <> Reply to Message

"tbowers" <tbowers@nrlssc.navy.mil> writes:

> How do I access the 2nd + dimensions of an array generally, without knowing
> the
> number of higher dims this array has. E.g. say a is a 3 column by
> n-dimensional
> aray, and n is unknown. Here, I'll define it as:

> a = indgen(3,2,4)

> I want the equivalent of (in this case):
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2

(rest deleted)

You should be able to do something like the following:

```
b = a[0,*,*,*,*,*,*,*]^2 + a[1,*,*,*,*,*,*,*]^2 + a[2,*,*,*,*,*,*,*]^2
```

even though A might not have so many dimensions. With your above example, you would then get

```
IDL> help,b
B               INT       = Array[1, 2, 4]
```

William Thompson

Subject: Re: Generally accessing the rest of the elements in an array
Posted by Paul van Delst on Wed, 21 Feb 2001 13:55:35 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
>
> "tbowers" <tbowers@nrlssc.navy.mil> writes:
>
>> How do I access the 2nd + dimensions of an array generally, without knowing
>> the
>> number of higher dims this array has. E.g. say a is a 3 column by
>> n-dimensional
>> aray, and n is unknown. Here, I'll define it as:
>
>> a = indgen(3,2,4)
>

>> I want the equivalent of (in this case):
>> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2
>
>   (rest deleted)
>
> You should be able to do something like the following:
>
>   b = a[0,*,*,*,*,*,*,*]^2 + a[1,*,*,*,*,*,*,*]^2 + a[2,*,*,*,*,*,*,*]^2

Ugh.

I think Mr/Dr Bowers should think about a new data structure that can deal with the flexibility he requires. An IDL structure perhaps?

When you start using arrays of more than three or four dimensions, and this is my very personal viewpoint only, I would definitely spend a couple of hours thinking about how to repackage either the data in the code or the code itself to change the "flow of data" (insert hand-waving here) to avoid the type of expressions like the above.

paulv

--
Paul van Delst A little learning is a dangerous thing;
CIMSS @ NOAA/NCEP Drink deep, or taste not the Pierian spring;
Ph: (301)763-8000 x7274 There shallow draughts intoxicate the brain,
Fax:(301)763-8545 And drinking largely sobers us again.
pvandelst@ncep.noaa.gov Alexander Pope.

Subject: Re: Generally accessing the rest of the elements in an array
Posted by Jaco van Gorkom on Wed, 21 Feb 2001 14:21:36 GMT
View Forum Message <> Reply to Message

tbowers wrote:
...
> a = indgen(3,2,4)
>
> I want the equivalent of (in this case):
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2
>
> but this requires *'ing the correct dimensions ([0,*,*] for 3 dims,
> [0,*,*,*] for 4 dims
> etc). What I need is a general way to access the "rest" of the data, as Paul
> Harvey
> would say.
>
> Actually, what I *really* want is to access it all generally so if a is 3
> columns, it'll be

> as above
>
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2
> but if it's 4 columns, it'll be
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2 + (a[3,*,*])^2
> 5 columns...
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2 + (a[3,*,*])^2 + (a[4,*,*])^2
> n columns...
> b = (a[0,*,*])^2 + (a[1,*,*])^2 + (a[2,*,*])^2 + ... + (a[n-1,*,*])^2
>
> but I don't think this is possible without a for loop.
>

Hi Todd,

I would guess that this is only a solution to your simplified example,
and not to your real problem:

b = TOTAL(a^2, 1)

Maybe it comes in useful somehow. It can be made more general by
combination with REFORM and TRANSPOSE, in order to sum over almost any
periodic subset of an array.

groetjes,
Jaco van Gorkom

Oops! Sorry, I meant to stay in lurking for 3 months... now I still
can't make it into David's thread! Hello anyway, I'm Jaco, no dog.

Subject: Re: Generally accessing the rest of the elements in an array
Posted by Jaco van Gorkom on Wed, 21 Feb 2001 17:50:09 GMT
View Forum Message <> Reply to Message

Paul van Delst wrote:
> William Thompson wrote:
>> "tbowers" <tbowers@nrlssc.navy.mil> writes:
>>> How do I access the 2nd + dimensions of an array generally, without knowing
>>> the
>>> number of higher dims this array has.
...
>> You should be able to do something like the following:
>> b = a[0,*,*,*,*,*,*,*]^2 + a[1,*,*,*,*,*,*,*]^2 + a[2,*,*,*,*,*,*,*]^2
>
> Ugh.
>

> I think Mr/Dr Bowers should think about a new data structure that can deal with the
> flexibility he requires. An IDL structure perhaps?
>

What makes IDL arrays different from arrays in some other languages is the fact that they *have* this flexibility. Whether one requires it or not. The dimensions of arrays can change anytime, and even IDL itself takes the freedom to remove dimensions if they happen to be of size 1. Now in *my* personal viewpoint, the challenge is to write IDL code which is able to handle input of any type and dimension for which a sensible reaction can be defined. So if I were to write a function which calculates the square of the Nth column, I would try to write it for any dimensions and for any type (except for strings maybe).

Jaco

PS: It's just that I wrote a lot of things originally for byte time traces. Later on I of course ended up wanting to apply them to complex arrays of 2 or 3 dimensions. In most cases I was amazed at how easy it *would* have been to write them fully generic in the first place.
