Subject: Re: tensor multiplication
Posted by Craig Markwardt on Thu, 18 May 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Paul van Delst <pvandelst@ncep.noaa.gov> writes:

> Daniel Luebbert wrote:
>>
>> Hi,
>>
>> does anybody out there know an efficient and elegant way (i.e., without
>> for-loops)
>> to implement a tensor multiplication in IDL?
>>
...
>> But now, when I take one more dimension, like
>> c = indgen(2,3,4)
>> d = indgen(4)
>> then
>> help, c#d
>> gives an error! (incompatible matrix dimensions...).
>> What a would like to get is obviously an
>> ARRAY[2,3]
>>
>> Does anybody know how?
>
> Great question! I have always only thought about 2D matrices, but why
> should nD be any different? Maybe when IDL says "matrix" is really means
> 2-D array?
>
> how about
>
> sz = size(c)
> e = INTARR(sz(1), sz(2))
> FOR i = 0, sz(1) - 1 DO BEGIN
> tmp_e = REFORM(c[i,*,*]) # d
> e[i,*] = TEMPORARY(tmp_e)
> ENDFOR

I would have done something similar. The point is that FOR loops are
reasonable to use if you can do enough processing within one loop
iteration. Here, "enough" is an entire matrix multiply.

Yorick, a language similar to IDL, has a cool tensor contraction
operator which would fit the bill nicely, or at least make it look
more compact.

> p.s. Can someone explain to me the utility/need for having both the #

> *and* ## operator? I understand their operation but why both?
> Convenience? Performance?

I do occasionally use ## for matrix multiply. I use # more often to expand a vector into an array; there's been discussion here before on which is faster, # or rebin.

Craig

--

---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------