Subject: Re: array chicanery
Posted by Mark Hadfield on Fri, 16 Nov 2001 00:35:47 GMT
View Forum Message <> Reply to Message

From: "Rochelle Hatton" <nrh@imag.wsahs.nsw.gov.au>
> I've experienced some weird business with arrays, and I'm
> hoping somebody can explain.

Sure can!

> I have been using the IDL function PCOMP, calculating the principal
> components of some data. If I feed in a float array to the function,
> eg.
> result=PCOMP(data, coefficients=eigenvectors, $
> eigenvalues=eigenvalues, /covariance, /standardize)
>
> I get a result, but when I redisplay the data array, it has changed.
> Since PCOMP only returns a result, what is it doing to my original
> array, and why?

PCOMP doesn't "only return a result".

Looking at the source code for PCOMP and searching for "Array" (the name of the first argument) one sees...

```
;Standardize the columns of the input array with a mean of 0.0
;and a variance of 1.0
if KEYWORD_SET(Standardize) ne 0 then $
  Array = STANDARDVAR(Array, Dimension, Double = Double)
```

So "Array" is modified and if this corresponds to an argument passed by reference then the original will be modified. In your case you have passed it an array ("data") so it is fair game. For more info on this issue see "Procedures and Functions/Parameter Passing Mechanisms" in "Building IDL Applications". This is a common source of confusion, usually in the opposite sense: people pass array segments and structure tags to routines and are surprised when they *aren't* modified.

The easiest way of protecting your original array is to pass PCOMP a copy of the data. This should work

```
result=PCOMP(reform(data),...)
```

BTW can anyone suggest a better general purpose function for this purpose?

I will refrain from commenting on whether PCOMP's behaviour represents good programming practice... But I do think they could have warned you.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

--