Subject: Re: Vector comparison.
Posted by tam on Thu, 20 Nov 2003 15:03:02 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

```
>  Reimar Bauer writes:
>
>
>> P.S. If we have all read it as carefully as mentioned what would we have
>> answered. Pavel will uses loops as well, we boths have not read
>> carefully enough, David hasn't replied.
>
>
>  No, David was interested in the *answer*. He didn't
>  have a solution of his own. :-)
>
>  Cheers,
>
>  David
```

How about the following, which handles duplicates in both arrays
and uses no loops.

```
; Find all of the indices of the elements
; in the first array that are matched in the second
; array
function match, a, b

    m = n_elements(a)
    n = n_elements(b)


    ; Create an array with each possible pairing of the first
    ; and last elements

    cmp = replicate(a[0], 2, m*n)

    ind  = lindgen(m*n)
    ind0 = ind mod m
    ind1 = ind  /  m

    ; Just lay out the 'a' array multiple times.
    cmp[0,*] = a[ind0]

    ; Repeat each element of the 'b' array m times so that
    ; we get each a element paired with each b element.
    cmp[1,*] = b[ind1]
```

```
     ; Now find all of the matches.
     w = where(cmp[0,*] eq cmp[1,*])

     if (w[0] eq -1) then begin
         return, w
     endif else begin

 ; Handle multiples in the 'b' set.
 ; If the elements in 'b' are guranteed
 ; to be unique then we can just return 'w mod m'

 h = histogram(w mod m, min=0)
 return, where(h ne 0)
     endelse
end
```

Have I missed something here... I think this would be reasonably efficient. Probably don't want to have separate ind, ind0, and ind1 arrays, but I thought that might show the algorithm more clearly.

```
      cmp[*,*] = [[a[lindgen(m*n) mod m][b[lindgen(m*n)/m]]
```

is not the kind of thing I can follow!

Regards,
Tom McGlynn