Subject: Re: Vector comparison.
Posted by tam on Thu, 20 Nov 2003 18:55:18 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

```
> David  Fanning <david@dfanning.com> writes:
>
>> Timm Weitkamp writes:
>>
>>
>>> Darnit! I was so proud of my answer... only to find that Chris Lee already
>>> proposed the same method (the "memory-hogging" one), albeit using more
>>> lines and this "if count eq 0 then ..." after WHERE, which always annoys
>>> me for some stupid reason. It can be avoided, which gives me an excuse to
>>> post my suggestion anyway:
>>>
>>> function Get_Match, a, b
>>>   na = n_elements(a) & nb = n_elements(b)
>>>   return, where(total((rebin(a,na,nb) - rebin(transpose(b), na, nb)) eq 0, 2))
>>> end
>>
>> Now there you go! That's a solution I can
>> steal for my web page. :-)
>
>
> I think this is the same solution as Tom McGlynn.  This is a classic
> solution by brute force *memory* hogging instead of compute hogging.
>
> If the lists are really large, then the arrays will be
> n_elements(a)*n_elements(b) in size, which can be pretty big.  For
> example if each list had 10000 elements, then those 100 million
> element arrays would start to suck a large amount of virtual memory.
>
> Craig
>
```

If the range of elements in b is not too large, then I suppose
something like:

```
  mn = min(a, max=mx)-1
  mx = mx+1
  h  = histogram(a,min=mn,max=mx)
  w  = where(h[a-mn] ne 0)
```

would do the trick....   We make the histogram one
to large on both sides so that we can use IDL's quirk
that going beyond the edge of an array is treated as

the last (or first) element. So long as the range in
b is smaller than the product of the number of elements
in the two arrays, this may be faster.

But it could only work on integers.
 Tom

P.S. Note that I'm doing my best in all my solutions to ensure
that the histogram function is called.