Subject: Re: Finding strings values common to two (large!) arrays
Posted by Jeremy Bailin on Thu, 29 Oct 2015 18:17:22 GMT
View Forum Message <> Reply to Message

On Wednesday, October 28, 2015 at 5:53:32 PM UTC-4, Dick Jackson wrote:
> On Wednesday, 28 October 2015 14:48:13 UTC-7, Dick Jackson wrote:
>> On Wednesday, 28 October 2015 12:26:30 UTC-7, rrya...@gmail.com wrote:
>>> Yeah, I learned that trick from your posts on Fanning's webpage. It's been a revelation. But i still don't like the structure of:
>>>
>>> id = id[uniq(id,sort(id))]
>>>
>>> i wish uniq just had a built-in flag to do this for me... Under what circumstance would I want to *NOT* sort? Seems like if they just built uniq to sort by default, you could probably optimize this at the compiler level --- though I'm hardly an expert...
>>>
>>> id = id[uniq(id)] ;would be nice or just
>>>
>>> -R
>>
>> As of IDL 8.4, you can now do this:
>>
>> IDL> a = [6, 2, 8, 3, 1, 8, 5, 3] ; (first digits of tau)
>> IDL> a.uniq()
>> 1 2 3 5 6 8
>>
>> Reference: IDL_Variable::Uniq
>> http://www.exelisvis.com/docs/IDL_Variable.html#Uniq
>
> ... and to put a finer point on it, this method assumes the elements need to be sorted. In the case where they are already sorted (where sorting time can thus be saved, which is the default with the regular uniq() function), you can use:
> a.uniq(/no_sort)
>
> Cheers,
> -Dick
>
> Dick Jackson Software Consulting Inc.
> Victoria, BC, Canada --- http://www.d-jackson.com

Actually, there is a use case for running uniq intentionally without sorting, which I have used before: identifying duplications (similar to times you might use label_region).

```
IDL> q = [3,5,3,3,8,7,7]
IDL> q[uniq(q)]
      3       5       3       8       7
```

-Jeremy.