Subject: Re: Ordered index array
Posted by Paolo Grigis on Thu, 08 Sep 2005 12:30:25 GMT
View Forum Message <> Reply to Message

Emmanuel Christophe wrote:
> Thanks Paolo,
> Your function does exactly what I want, but my problem is precisely that
> I need to do it on huge array (300 000 elements at least), that why i'm
> looking for a more 'IDL' way to do it :)
>
> Emmanuel
Ok, let's do it. We'll use a loop, but this should not be too
bad as long as you don't have more than a few million elements,
provided that we just do simple operations in the loop.

```
PRO test2

a=[1,3,3,1,2,3,2,2,1,2,3]

c=lonarr(n_elements(a))

h=histogram(a,min=1L,reverse_ind=ri)

listlowestind=replicate(-1L,n_elements(h))

FOR i=0L,n_elements(h)-1 DO BEGIN
    IF ri[i] NE ri[i+1] THEN listlowestind[i]=ri[ri[i]]
ENDFOR

indok=where(listlowestind GT -1L,countindok)
indoffset=n_elements(listlowestind)-countindok
indsorted=(sort(listlowestind))

FOR i=0L,countindok-1 DO BEGIN
    this_element=a[listlowestind[indsorted[i+indoffset]]]
    c[ri[ri[this_element-1]:ri[this_element]-1]]=i+1
ENDFOR

print,c

end
```

The idea here is to use histogram to scout for the first occurrence
of each integer number and store it in listlowestind. We then
sort the list, such that we know which element occurs first,
and then with a second loop we fill the output array c with
the occurrences of each element, and since we have sorted the
array first, we know which rank to assign to each element.

This should be moderately efficient. I assumed that the array
a has integer elements starting from 1, if not the case, just
sum 1-min(a) to the array before performing the operation.

Ciao,
Paolo

>
>
>
>
>> But this will fail if 'a' has more elements than the number of
>> its different values, for instance a=[3,3,1,2,3,2,2,1,2,3].
>>
>> One could try this:
>>
>> PRO test
>>
>> a=[3,3,1,2,3,2,2,1,2,3]
>>
>> b=a
>> c=intarr(n_elements(a))
>>
>> h=histogram(a,min=1,reverse_ind=ri)
>>
>> done=0
>> rank=1
>> WHILE NOT done DO BEGIN
>>   actual_value=b[0]
>>   ind_actual_value=ri[ri[actual_value-1]:ri[actual_value]-1]
>>   c[ind_actual_value]=rank
>>   rank=rank+1
>>   indremove=where(b NE actual_value,count)
>>   IF count GT 0 THEN b=b[indremove] ELSE done=1
>> ENDWHILE
>>
>> print,a
>> print,c
>>
>> END
>>
>>
>> but it will get inefficient as the numbers of different values
>> in 'a' grows, as the code in the loop get called more and more
>> times...
>>
>> Ciao,

>> Paolo
>>
>>
>>> Cheers,
>>>
>>> David
>>>
>>>