Subject: array manipulation (TOTAL-ing or MEDIAN-ing) in uneven bins
Posted by havok2063 on Wed, 12 Dec 2012 16:16:21 GMT
View Forum Message <> Reply to Message

I have several unrelated problems that I'm solving in the same efficient way (with loops). I'm trying to perform some array operation on an array, according to a list of (let's call them) uneven bins.

I have an array, say d, of 146 elements. I have a separate array that represents uneven bins that I want to perform the operation on, like MEDIAN, or TOTAL. For example,

ntot = [15,45,56,90,116,146]

I want as output an array, of 6 elements, that contains the MEDIAN (or TOTAL) of array d according to the indices listed in ntot.

So the 1st element would contain median(d[0:14],/even), the 2nd median(d[15:44],/even), etc....

Or the same thing with total....total(d[0:14]), total(d[15:44]) , etc...

Right now I'm looping over the number of elements in ntot to do this and I don't much care for loops.

I don't think this is quite the same thing as the example given in the "Horror and Disgust of Histogram" article nor does this sound like something I can do with value_locate, although I'm not too familiar with value_locate.

Any ideas on this? Thanks a lot.

Subject: Re: array manipulation (TOTAL-ing or MEDIAN-ing) in uneven bins
Posted by Jeremy Bailin on Wed, 12 Dec 2012 22:03:51 GMT
View Forum Message <> Reply to Message

On 12/12/12 10:16 AM, havok2063@gmail.com wrote:
>
> I have several unrelated problems that I'm solving in the same efficient way (with loops). I'm trying to perform some array operation on an array, according to a list of (let's call them) uneven bins.
>
> I have an array, say d, of 146 elements. I have a separate array that represents uneven bins that I want to perform the operation on, like MEDIAN, or TOTAL. For example,
>
> ntot = [15,45,56,90,116,146]
>
> I want as output an array, of 6 elements, that contains the MEDIAN (or TOTAL) of array d according to the indices listed in ntot.
>

> So the 1st element would contain median(d[0:14],/even), the 2nd median(d[15:44],/even), etc....
>
> Or the same thing with total....total(d[0:14]), total(d[15:44]) , etc...
>
> Right now I'm looping over the number of elements in ntot to do this and I don't much care for loops.
>
> I don't think this is quite the same thing as the example given in the "Horror and Disgust of Histogram" article nor does this sound like something I can do with value_locate, although I'm not too familiar with value_locate.
>
> Any ideas on this? Thanks a lot.
>

As David says, this screams VALUE_LOCATE. And HISTOGRAM. They play very nicely together for this sort of problem!

First we need to label the bin for each element:

```
nelements = 146
binlabel = value_locate(ntot, lindgen(nelements))
```

Then use histogram to group the elements by bin label. Notice that the way you've defined ntot, elements 0 through 14 will be labelled "-1" by value_locate, so we start the histogram there:

```
nbin = n_elements(ntot)
hist = histogram(binlabel, min=-1, max=nbin-1, reverse_indices=ri)
```

And finally we do the usual loop through the reverse indices to calculate the statistics:

```
medianbin = fltarr(nbin)
totbin = fltarr(nbin)
for i=0L,nbin-1 do if hist[i] gt 0 then begin
  these = ri[ri[i]:ri[i+1]-1]
  medianbin[i] = median(d[these], /even)
  totbin[i] = total(d[these])
endif
```

-Jeremy.

Subject: Re: array manipulation (TOTAL-ing or MEDIAN-ing) in uneven bins
Posted by Jeremy Bailin on Wed, 12 Dec 2012 22:18:07 GMT
View Forum Message <> Reply to Message

On 12/12/12 4:03 PM, Jeremy Bailin wrote:

> On 12/12/12 10:16 AM, havok2063@gmail.com wrote:
>>
>> I have several unrelated problems that I'm solving in the same
>> efficient way (with loops). I'm trying to perform some array
>> operation on an array, according to a list of (let's call them) uneven
>> bins.
>>
>> I have an array, say d, of 146 elements. I have a separate array that
>> represents uneven bins that I want to perform the operation on, like
>> MEDIAN, or TOTAL. For example,
>>
>> ntot = [15,45,56,90,116,146]
>>
>> I want as output an array, of 6 elements, that contains the MEDIAN (or
>> TOTAL) of array d according to the indices listed in ntot.
>>
>> So the 1st element would contain median(d[0:14],/even), the 2nd
>> median(d[15:44],/even), etc....
>>
>> Or the same thing with total....total(d[0:14]), total(d[15:44]) , etc...
>>
>> Right now I'm looping over the number of elements in ntot to do this
>> and I don't much care for loops.
>>
>> I don't think this is quite the same thing as the example given in the
>> "Horror and Disgust of Histogram" article nor does this sound like
>> something I can do with value_locate, although I'm not too familiar
>> with value_locate.
>>
>> Any ideas on this? Thanks a lot.
>>
>
> As David says, this screams VALUE_LOCATE. And HISTOGRAM. They play very
> nicely together for this sort of problem!
>
> First we need to label the bin for each element:
>
> nelements = 146
> binlabel = value_locate(ntot, lindgen(nelements))
>
> Then use histogram to group the elements by bin label. Notice that the
> way you've defined ntot, elements 0 through 14 will be labelled "-1" by
> value_locate, so we start the histogram there:
>
> nbin = n_elements(ntot)
> hist = histogram(binlabel, min=-1, max=nbin-1, reverse_indices=ri)
>
> And finally we do the usual loop through the reverse indices to

> calculate the statistics:
>
> medianbin = fltarr(nbin)
> totbin = fltarr(nbin)
> for i=0L,nbin-1 do if hist[i] gt 0 then begin
> these = ri[ri[i]:ri[i+1]-1]
> medianbin[i] = median(d[these], /even)
> totbin[i] = total(d[these])
> endif
>
> -Jeremy.

Actually, for the total you can do a lot better by using cumulative:

runningtotal = total(d, /cumulative)
totbin = runningtotal[ntot] - [0,runningtotal[ntot]]

-Jeremy.

Subject: Re: array manipulation (TOTAL-ing or MEDIAN-ing) in uneven bins
Posted by Jeremy Bailin on Wed, 12 Dec 2012 22:19:07 GMT
View Forum Message <> Reply to Message

On 12/12/12 4:18 PM, Jeremy Bailin wrote:
> On 12/12/12 4:03 PM, Jeremy Bailin wrote:
>> On 12/12/12 10:16 AM, havok2063@gmail.com wrote:
>>>
>>> I have several unrelated problems that I'm solving in the same
>>> efficient way (with loops). I'm trying to perform some array
>>> operation on an array, according to a list of (let's call them) uneven
>>> bins.
>>>
>>> I have an array, say d, of 146 elements. I have a separate array that
>>> represents uneven bins that I want to perform the operation on, like
>>> MEDIAN, or TOTAL. For example,
>>>
>>> ntot = [15,45,56,90,116,146]
>>>
>>> I want as output an array, of 6 elements, that contains the MEDIAN (or
>>> TOTAL) of array d according to the indices listed in ntot.
>>>
>>> So the 1st element would contain median(d[0:14],/even), the 2nd
>>> median(d[15:44],/even), etc....
>>>
>>> Or the same thing with total....total(d[0:14]), total(d[15:44]) , etc...
>>>
>>> Right now I'm looping over the number of elements in ntot to do this

>>> and I don't much care for loops.
>>>
>>> I don't think this is quite the same thing as the example given in the
>>> "Horror and Disgust of Histogram" article nor does this sound like
>>> something I can do with value_locate, although I'm not too familiar
>>> with value_locate.
>>>
>>> Any ideas on this? Thanks a lot.
>>>
>>
>> As David says, this screams VALUE_LOCATE. And HISTOGRAM. They play very
>> nicely together for this sort of problem!
>>
>> First we need to label the bin for each element:
>>
>> nelements = 146
>> binlabel = value_locate(ntot, lindgen(nelements))
>>
>> Then use histogram to group the elements by bin label. Notice that the
>> way you've defined ntot, elements 0 through 14 will be labelled "-1" by
>> value_locate, so we start the histogram there:
>>
>> nbin = n_elements(ntot)
>> hist = histogram(binlabel, min=-1, max=nbin-1, reverse_indices=ri)
>>
>> And finally we do the usual loop through the reverse indices to
>> calculate the statistics:
>>
>> medianbin = fltarr(nbin)
>> totbin = fltarr(nbin)
>> for i=0L,nbin-1 do if hist[i] gt 0 then begin
>>   these = ri[ri[i]:ri[i+1]-1]
>>   medianbin[i] = median(d[these], /even)
>>   totbin[i] = total(d[these])
>> endif
>>
>> -Jeremy.
>
> Actually, for the total you can do a lot better by using cumulative:
>
> runningtotal = total(d, /cumulative)
> totbin = runningtotal[ntot] - [0,runningtotal[ntot]]
>
> -Jeremy.

Ack, hit send to soon. That should be:

runningtotal = total(d, /cumulative)

totbin = runningtotal[ntot-1] - [0,runningtotal[ntot-1]]

-Jeremy.

Subject: Re: array manipulation (TOTAL-ing or MEDIAN-ing) in uneven bins
Posted by havok2063 on Thu, 13 Dec 2012 20:11:16 GMT
View Forum Message <> Reply to Message

On Wednesday, December 12, 2012 5:19:07 PM UTC-5, Jeremy Bailin wrote:
> On 12/12/12 4:18 PM, Jeremy Bailin wrote:
>
>> On 12/12/12 4:03 PM, Jeremy Bailin wrote:
>
>>> On 12/12/12 10:16 AM, havok2063@gmail.com wrote:
>
>>>>
>
>>>> I have several unrelated problems that I'm solving in the same
>
>>>> efficient way (with loops). I'm trying to perform some array
>
>>>> operation on an array, according to a list of (let's call them) uneven
>
>>>> bins.
>
>>>>
>
>>>> I have an array, say d, of 146 elements. I have a separate array that
>
>>>> represents uneven bins that I want to perform the operation on, like
>
>>>> MEDIAN, or TOTAL. For example,
>
>>>>
>
>>>> ntot = [15,45,56,90,116,146]
>
>>>>
>
>>>> I want as output an array, of 6 elements, that contains the MEDIAN (or
>
>>>> TOTAL) of array d according to the indices listed in ntot.
>
>>>>
>
>>>> So the 1st element would contain median(d[0:14],/even), the 2nd
>

>>>> median(d[15:44],/even), etc....
>
>>>>
>
>>>> Or the same thing with total....total(d[0:14]), total(d[15:44]) , etc...
>
>>>>
>
>>>> Right now I'm looping over the number of elements in ntot to do this
>
>>>> and I don't much care for loops.
>
>>>>
>
>>>> I don't think this is quite the same thing as the example given in the
>
>>>> "Horror and Disgust of Histogram" article nor does this sound like
>
>>>> something I can do with value_locate, although I'm not too familiar
>
>>>> with value_locate.
>
>>>>
>
>>>> Any ideas on this? Thanks a lot.
>
>>>>
>
>>>
>
>>> As David says, this screams VALUE_LOCATE. And HISTOGRAM. They play very
>
>>> nicely together for this sort of problem!
>
>>>
>
>>> First we need to label the bin for each element:
>
>>>
>
>>> nelements = 146
>
>>> binlabel = value_locate(ntot, lindgen(nelements))
>
>>>
>
>>> Then use histogram to group the elements by bin label. Notice that the
>

>>> way you've defined ntot, elements 0 through 14 will be labelled "-1" by
>
>>> value_locate, so we start the histogram there:
>
>>>
>
>>> nbin = n_elements(ntot)
>
>>> hist = histogram(binlabel, min=-1, max=nbin-1, reverse_indices=ri)
>
>>>
>
>>> And finally we do the usual loop through the reverse indices to
>
>>> calculate the statistics:
>
>>>
>
>>> medianbin = fltarr(nbin)
>
>>> totbin = fltarr(nbin)
>
>>> for i=0L,nbin-1 do if hist[i] gt 0 then begin
>
>>>   these = ri[ri[i]:ri[i+1]-1]
>
>>>   medianbin[i] = median(d[these], /even)
>
>>>   totbin[i] = total(d[these])
>
>>> endif
>
>>>
>
>>> -Jeremy.
>
>>
>
>> Actually, for the total you can do a lot better by using cumulative:
>
>>
>
>> runningtotal = total(d, /cumulative)
>
>> totbin = runningtotal[ntot] - [0,runningtotal[ntot]]
>
>>
>

>> -Jeremy.
>
>
>
> Ack, hit send to soon. That should be:
>
>
>
> runningtotal = total(d, /cumulative)
>
> totbin = runningtotal[ntot-1] - [0,runningtotal[ntot-1]]
>
>
>
> -Jeremy.

Excellent. That really hits the spot. Thanks a lot. I was hovering somewhere around there but couldn't quite converge on what to do with value_locate.
