Subject: Re: Mode function for floating point arrays
Posted by Rob Klooster on Mon, 08 Jul 2013 08:24:08 GMT
View Forum Message <> Reply to Message

On second thought, it will be more efficient to treat the array as a sparse array and use value_locate, as in David's article:

http://www.idlcoyote.com/code_tips/valuelocate.html

```
sortedarray = array[Sort(array)]
arrayenum = sortedarray[Uniq(sortedarray)]
mappedarray = Value_Locate(arrayenum, array)
hist = histogram(mappedarray, min=0)
mode = arrayenum[where(hist eq max(hist))]
```

Maybe you can update the uniq function to accept a value for epsilon to decide whether two floating values are equal or not.

Regards,
Rob.

> Op maandag 8 juli 2013 10:01:16 UTC+2 schreef Rob Klooster het volgende:
> Hi Matthew,
>
>
>
> Histogram also works on floating point arrays, you just need to set the binsize:
>
>
>
> hist = histogram(array, binsize=epsilon, locations=locations)
>
> mode = locations[where(hist eq max(hist))]
>
>
>
> Note that for small values of epsilon, the resulting histogram array can become very large.
>
>
>
> Regards,
>
> Rob.
>
>
>

>
>
> Op vrijdag 5 juli 2013 20:55:23 UTC+2 schreef Matthew Argall het volgende:
>
>> PEAMBLE:
>
>>
>
>> I need a function that finds the mode of a floating point array. I have read David Fanning's article about integer arrays
>
>>
>
>>
>
>>
>
>> http://www.idlcoyote.com/code_tips/mode.html
>
>>
>
>>
>
>>
>
>> From this article about majority voting, it seems like "Hist_ND" works for floating point values, but I have no experience with the magic of HISTOGRAM
>
>>
>
>>
>
>>
>
>> https://groups.google.com/forum/#!searchin/comp.lang.idl-pvw
ave/Mode$20of$20a$20floating$20point$20array/comp.lang.idl-p
vwave/YZK2ey-O5sE/9fLvx_AG2IAJ
>
>>
>
>>
>
>>
>
>>
>
>>
>

```
>> QUESTION:
>
>>
>
>> Here is my attempt. Can anyone make it better/faster?
>
>>
>
>>
>
>>
>
>> ;-----------------------------------------------------
>
>>
>
>> function mrmode, array, $
>
>>
>
>> EPSILON=epsilon
>
>>
>
>>     compile_opt idl2
>
>>
>
>>
>
>>
>
>>
>
>>     ;Number of points in ARRAY
>
>>
>
>>     npts = n_elements(array)
>
>>
>
>>
>
>>
>
>>
>
>>     ;Default value for EPSILON
>
>>
>
```

```
>>     if n_elements(epsilon) eq 0 then epsilon = 1d-5
>
>>
>
>>
>
>>
>
>>     ;[index, count] for keeping track of mode statistics
>
>>
>
>>     mode_count = lonarr(2, npts)
>
>>
>
>>
>
>>
>
>>     ;Store first ~unique number. Count the how many ~unique numbers there are.
>
>>
>
>>     mode_count[*,0] = [0,1]
>
>>
>
>>     nunique = 1
>
>>
>
>>
>
>>
>
>>     ;Step through all points in ARRAY
>
>>
>
>>     for i = 1, npts - 1 do begin
>
>>
>
>>        match_found = 0
>
>>
>
```

```
>>
>
>>
>
>>       ;Try to pair the new point with other mode candidates
>
>>
>
>>       for j = 0, nunique - 1 do begin
>
>>
>
>>          if array[i] gt array[mode_count[0,j]]-epsilon && $
>
>>
>
>>             array[i] lt array[mode_count[0,j]]+epsilon $
>
>>
>
>>          then begin
>
>>
>
>>
>
>>
>
>>             mode_count[1,j] += 1
>
>>
>
>>             match_found = 1
>
>>
>
>>          endif
>
>>
>
>>       endfor
>
>>
>
>>
>
>>
>
>
```

```
>>        ;If no match was found, create a new mode candidate
>
>>
>
>>        if match_found eq 0 then begin
>
>>
>
>>            mode_count[*,nunique] = [i,1]
>
>>
>
>>            nunique += 1
>
>>
>
>>        endif
>
>>
>
>>    endfor
>
>>
>
>>
>
>>
>
>>    ;Get the mode
>
>>
>
>>    void = max(mode_count[1,*], iMode)
>
>>
>
>>    mode = array[mode_count[0,iMode]]
>
>>
>
>>
>
>>
>
>>    return, mode
>
>>
>
```

```
>> end
>
>>
>
>>
>
>>
>
>>
>
>>
>
>>   ;------------------------------------------------------------ ----------
>
>>
>
>>  ;Example Program (IDL> .r mrmode) ////////////////////////////////////////
>
>>
>
>>   ;------------------------------------------------------------ ----------
>
>>
>
>>  array = [1.2, 0.1, 3.3, 0.1, 2.0, 3.3, 4.8, 1.2, 0.1, 0.1, 6.7, 3.3]
>
>>
>
>>  mode = MrMode(array)
>
>>
>
>>  print, FORMAT='(%"The mode is: %f")', mode
>
>>
>
>>
>
>>
>
>>  end
```