Subject: Re: Matching Lats and Lons from two arrays
Posted by Brian Larsen on Tue, 26 Aug 2008 16:11:38 GMT
View Forum Message <> Reply to Message

I have tried this on several occasions (for a little different
application but I think its the same) and have had no luck eliminating
the for loop, so I just wrote it in a function to hide it from
myself. This is my try at this based on value locate:
 http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1/7_roun
d2array_(updated_7Jan2008).html

If others know how to eliminate the for loop that would be fantastic.

Cheers,

Brian

---------- ----------
Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Matching Lats and Lons from two arrays
Posted by pgrigis on Tue, 26 Aug 2008 16:50:25 GMT
View Forum Message <> Reply to Message

Would something like this work for sorted x (plus some fix for first
and last element)?
There's going to be an overhead for sorting x if not already sorted
however.

```
x=[3,3.5,4,6.5,7]
y=[3.4, 3.0, 6.8, 6.3]

a=value_locate(x,y)
result=a+( (y-x[a]) GT (x[a+1]-y))
print,result
```

Ciao,
Paolo

Brian Larsen wrote:
> I have tried this on several occasions (for a little different
> application but I think its the same) and have had no luck eliminating

> the for loop, so I just wrote it in a function to hide it from
> myself. This is my try at this based on value locate:
> http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1/7_roun
d2array_(updated_7Jan2008).html
>
> If others know how to eliminate the for loop that would be fantastic.
>
>
> Cheers,
>
> Brian
>
> -- ----------
> Brian Larsen
> Boston University
> Center for Space Physics
> http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Matching Lats and Lons from two arrays
Posted by Juggernaut on Tue, 26 Aug 2008 16:58:57 GMT
View Forum Message <> Reply to Message

On Aug 26, 12:11 pm, Brian Larsen <balar...@gmail.com> wrote:
> I have tried this on several occasions (for a little different
> application but I think its the same) and have had no luck eliminating
> the for loop, so I just wrote it in a function to hide it from
> myself. This is my try at this based on value
locate:http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1 /7_round2array_(...
>
> If others know how to eliminate the for loop that would be fantastic.
>
> Cheers,
>
> Brian
>
> -- ----------
> Brian Larsen
> Boston University
> Center for Space Physicshttp://people.bu.edu/balarsen/Home/IDL

Not sure if the following is what you're looking for but first off you
have in your one line
FOR K = 0, CLOSE_LATS DO BEGIN
which doesn't make sense based upon what search2d returns (an array)
maybe it was really n_elements(CLOSE_LATS)?...I don't know
But maybe the following will lead you to the promise land or far far
away from it

Well I'd say if you're dlat(192,139) means that you have 139 lats for each of the 192 columns then you could do something like
CS_LATLON(0,4607)
dlat(192,139)
x2 = rebin(reform(dlat[0,*],139),139,4607)
x3 = rebin(reform(CS_LATLON,1,4607), 139,4607)
indices = where(abs(x3-x2) LT 1e-4)
x2[indices] gives the matching lats to within 1e-4
In the end something like
ncols = n_elements(dlat[*,0])
nrows = n_elements(dlat[0,*])
nels = n_elements(CS_LATLON)
for i = 0, ncols-1 do begin
  x2 = rebin(reform(dlat[i,*],nrows),nrows,nels)
  x3 = rebin(reform(CS_LATLON,1,nels), nrows,nels)
  indices = where(abs(x3-x2) LT 1e-4)
  vals = x2[indices]
  ;- Now you can print these vals or store them to an array or whatever
endfor
Just repeat for the lons...still has for loops but I'm pretty sure that works. I did a simple test on a 5x5 compared to a 1x25.
If it doesn't....I never did this.
Hope this helps eliminate some loops anyway....and actually is relevant.

Subject: Re: Matching Lats and Lons from two arrays
Posted by Jean H. on Tue, 26 Aug 2008 17:07:04 GMT
View Forum Message <> Reply to Message

pgrigis@gmail.com wrote:
> Would something like this work for sorted x (plus some fix for first
> and last element)?
> There's going to be an overhead for sorting x if not already sorted
> however.
>
> x=[3,3.5,4,6.5,7]
> y=[3.4, 3.0, 6.8, 6.3]
>
> a=value_locate(x,y)
> result=a+((y-x[a]) GT (x[a+1]-y))
> print,result
>
>
> Ciao,
> Paolo

this would work if you only have 1 coordinate (latitude), not with 2
(lat,long)...
Jean

>
> Brian Larsen wrote:
>> I have tried this on several occasions (for a little different
>> application but I think its the same) and have had no luck eliminating
>> the for loop, so I just wrote it in a function to hide it from
>> myself. This is my try at this based on value locate:
>> http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1/7_roun
d2array_(updated_7Jan2008).html
>>
>> If others know how to eliminate the for loop that would be fantastic.
>>
>>
>> Cheers,
>>
>> Brian
>>
>> ---------- ----------
>> Brian Larsen
>> Boston University
>> Center for Space Physics
>> http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Matching Lats and Lons from two arrays
Posted by Juggernaut on Tue, 26 Aug 2008 17:29:49 GMT
View Forum Message <> Reply to Message

On Aug 26, 1:07 pm, Jean H <jghas...@DELTHIS.ucalgary.ANDTHIS.ca>
wrote:
> pgri...@gmail.com wrote:
>> Would something like this work for sorted x (plus some fix for first
>> and last element)?
>> There's going to be an overhead for sorting x if not already sorted
>> however.
>
>> x=[3,3.5,4,6.5,7]
>> y=[3.4, 3.0, 6.8, 6.3]
>
>> a=value_locate(x,y)
>> result=a+((y-x[a]) GT (x[a+1]-y))
>> print,result
>
>> Ciao,

>> Paolo
>
> this would work if you only have 1 coordinate (latitude), not with 2
> (lat,long)...
> Jean
>
>
>
>> Brian Larsen wrote:
>>> I have tried this on several occasions (for a little different
>>> application but I think its the same) and have had no luck eliminating
>>> the for loop, so I just wrote it in a function to hide it from
>>> myself. This is my try at this based on value locate:
>>> http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1/7_roun d2array_(...
>
>>> If others know how to eliminate the for loop that would be fantastic.
>
>>> Cheers,
>
>>> Brian
>
>>> ---------- ----------
>>> Brian Larsen
>>> Boston University
>>> Center for Space Physics
>>> http://people.bu.edu/balarsen/Home/IDL

You could just do them simultaneously and only take the intersection
of the values....

```
for i = 0, ncols-1 do begin
  x2 = rebin(reform(dlat[i,*],nrows),nrows,nels)
  x3 = rebin(reform(CS_LATLON,1,nels), nrows,nels)
  indices = where(abs(x3-x2) LT 1e-4)
  vals = x2[indices]
  x2 = rebin(reform(dlon[i,*],nrows),nrows,nels)
  x3 = rebin(reform(CS_LATLON,1,nels), nrows,nels)
  indices2 = where(abs(x3-x2) LT 1e-4)
  vals = x2[indices2]
  intersecting = setintersection(indices,indices2)
endfor
```

Couldn't you?

Subject: Re: Matching Lats and Lons from two arrays
Posted by pgrigis on Tue, 26 Aug 2008 17:54:35 GMT
View Forum Message <> Reply to Message

Jean H wrote:
> pgrigis@gmail.com wrote:
>> Would something like this work for sorted x (plus some fix for first
>> and last element)?
>> There's going to be an overhead for sorting x if not already sorted
>> however.
>>
>> x=[3,3.5,4,6.5,7]
>> y=[3.4, 3.0, 6.8, 6.3]
>>
>> a=value_locate(x,y)
>> result=a+((y-x[a]) GT (x[a+1]-y))
>> print,result
>>
>>
>> Ciao,
>> Paolo
>
> this would work if you only have 1 coordinate (latitude), not with 2
> (lat,long)...
> Jean

Yes, this is not going to help the original poster.
But I think Brian's routine below is 1-dimensional.

Ciao,
Paolo

>
>>
>> Brian Larsen wrote:
>>> I have tried this on several occasions (for a little different
>>> application but I think its the same) and have had no luck eliminating
>>> the for loop, so I just wrote it in a function to hide it from
>>> myself. This is my try at this based on value locate:
>>> http://people.bu.edu/balarsen/Home/IDL/Entries/2008/1/7_roun
d2array_(updated_7Jan2008).html
>>>
>>> If others know how to eliminate the for loop that would be fantastic.
>>>
>>>
>>> Cheers,
>>>
>>> Brian
>>>
>>> ---------- ----------
>>> Brian Larsen

>>> Boston University
>>> Center for Space Physics
>>> http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Matching Lats and Lons from two arrays
Posted by Conor on Tue, 26 Aug 2008 18:41:27 GMT
View Forum Message <> Reply to Message

On Aug 26, 11:47 am, wilsona <awils...@bigred.unl.edu> wrote:
> I have 2 seperate arrays of Latittudes and Longitudes.
> CS_LATLON(0,4607) is one latitude array and dlat(192,139) is the
> other
> CS_LATLON(1,4607) is one longitude array and dlon(192,139) is the
> other.
> I want to index through each element in both CS_LATLON arrays and
> find
> which point(s) in the dlat and dlong arrays are closest.
> I tried using nested loops but that gave me 12 million+ loops which
> is
> too many for my liking. I now am trying search2d
> NUM_PNTS = N_ELEMENTS(CS_LATLON(0, *)) - 1
>
> FOR J = 0, NUM_PNTS DO BEGIN
> CLOSE_LATS = SEARCH2D(dlat, 0, 0, CS_LATLON(0,J),
> CS_LATLON(0,J), INCREASE=0.5, $
> DECREASE=0.5)
> lat1 = CS_LATLON(0,J) * PI / 180.0
> FOR K = 0, CLOSE_LATS DO BEGIN
> lat2 = dlat(K) * PI / 180.0
> d_long = CS_LATLON(1,J) - dlon(K)) * PI / 180.0
> DISTANCE = 10800.0 / PI * acos(sin(lat1) * sin(lat2)
> +
> cos(lat1) * $
> cos(lat2) * cos(d_long))
> ENDFOR ; K
> ENDFOR ; J
> This is not working they way I would like. Any suggestions on this
> would be greatly appreciated.

I often have to match up two star fields, which is pretty much the
same thing. You can download the routine I use here:

astro.ufl.edu/~cmancone/pros/qfind.pro

Here's the source. It basically just uses histogram to bin everyhing
and then searches one bin left and right:

```
function qfind,x1,y1,x2,y2,posshift=posshift

if n_elements(posshift) eq 0 then posshift = 1

n1 = n_elements(x1)
n2 = n_elements(x2)

; this is where I'll store the result
res = lonarr(2,n1)

; this mask list will tell us if an entry is from list one or list two
allinds = lindgen(n2)

; the histogram will tell us how many stars left and right we have to
check
hist = histogram(x2,binsize=posshift,omin=minx,reverse_indices=ri)

; calculate which bin each x went into
bins = floor((x1-minx)/posshift)>0
nbins=n_elements(hist)

f = 0L
for i=0L,n1-1 do begin
 ; figure out which bin this star is in
 bin = bins[i]

 ; adjust the indexes accordingly
 inds = ri[ri[(bin-1)>0]:ri[((bin+2)<nbins)]-1]

 ; calculate the distance from this star to its neighbors
 dist = sqrt( (x2[inds]-x1[i])^2 + (y2[inds]-y1[i])^2 )

 ; get the closest star within posshift that is from the second list
 mindist = min( dist, wm )

 ; no stars found
 if mindist gt posshift then continue

 ; record result
 res[*,f] = [i,inds[wm]]
 ++f
endfor

if f eq 0 then return,-1

; get rid of any blank entries
res = res[*,0:f-1]
```

return,res

end

Subject: Re: Matching Lats and Lons from two arrays
Posted by Conor on Tue, 26 Aug 2008 18:42:57 GMT
View Forum Message <> Reply to Message

On Aug 26, 2:41 pm, Conor <cmanc...@gmail.com> wrote:
> On Aug 26, 11:47 am, wilsona <awils...@bigred.unl.edu> wrote:
>
>
>
>> I have 2 seperate arrays of Latittudes and Longitudes.
>> CS_LATLON(0,4607) is one latitude array and dlat(192,139) is the
>> other
>> CS_LATLON(1,4607) is one longitude array and dlon(192,139) is the
>> other.
>> I want to index through each element in both CS_LATLON arrays and
>> find
>> which point(s) in the dlat and dlong arrays are closest.
>> I tried using nested loops but that gave me 12 million+ loops which
>> is
>> too many for my liking. I now am trying search2d
>> NUM_PNTS = N_ELEMENTS(CS_LATLON(0, *)) - 1
>
>> FOR J = 0, NUM_PNTS DO BEGIN
>> CLOSE_LATS = SEARCH2D(dlat, 0, 0, CS_LATLON(0,J),
>> CS_LATLON(0,J), INCREASE=0.5, $
>> DECREASE=0.5)
>> lat1 = CS_LATLON(0,J) * PI / 180.0
>> FOR K = 0, CLOSE_LATS DO BEGIN
>> lat2 = dlat(K) * PI / 180.0
>> d_long = CS_LATLON(1,J) - dlon(K)) * PI / 180.0
>> DISTANCE = 10800.0 / PI * acos(sin(lat1) * sin(lat2)
>> +
>> cos(lat1) * $
>> cos(lat2) * cos(d_long))
>> ENDFOR ; K
>> ENDFOR ; J
>> This is not working they way I would like. Any suggestions on this
>> would be greatly appreciated.
>
> I often have to match up two star fields, which is pretty much the
> same thing. You can download the routine I use here:
>

```
>  astro.ufl.edu/~cmancone/pros/qfind.pro
>
>  Here's the source.  It basically just uses histogram to bin everyhing
>  and then searches one bin left and right:
>
>  function qfind,x1,y1,x2,y2,posshift=posshift
>
>  if n_elements(posshift) eq 0 then posshift = 1
>
>  n1 = n_elements(x1)
>  n2 = n_elements(x2)
>
>  ; this is where I'll store the result
>  res = lonarr(2,n1)
>
>  ; this mask list will tell us if an entry is from list one or list two
>  allinds = lindgen(n2)
>
>  ; the histogram will tell us how many stars left and right we have to
>  check
>  hist = histogram(x2,binsize=posshift,omin=minx,reverse_indices=ri)
>
>  ; calculate which bin each x went into
>  bins = floor((x1-minx)/posshift)>0
>  nbins=n_elements(hist)
>
>  f = 0L
>  for i=0L,n1-1 do begin
>          ; figure out which bin this star is in
>          bin = bins[i]
>
>          ; adjust the indexes accordingly
>          inds = ri[ri[(bin-1)>0]:ri[((bin+2)<nbins)]-1]
>
>          ; calculate the distance from this star to its neighbors
>          dist = sqrt( (x2[inds]-x1[i])^2 + (y2[inds]-y1[i])^2 )
>
>          ; get the closest star within posshift that is from the second list
>          mindist = min( dist, wm )
>
>          ; no stars found
>          if mindist gt posshift then continue
>
>          ; record result
>          res[*,f] = [i,inds[wm]]
>          ++f
>  endfor
>
```

> if f eq 0 then return,-1
>
> ; get rid of any blank entries
> res = res[*,0:f-1]
>
> return,res
>
> end

I asked this same question before. You might find the discussion
useful.

http://groups.google.com/group/comp.lang.idl-pvwave/browse_t
hread/thread/629cbb2a852c5371/b568d74f6d539b79?hl=en&lnk
=gst&q=That+works+well+enough%2C+but+is+certainly+not+op
timal.++It+uses+the#b568d74f6d539b79

Subject: Re: Matching Lats and Lons from two arrays
Posted by wilsona on Tue, 26 Aug 2008 19:15:19 GMT
View Forum Message <> Reply to Message

Thanks for the responses! Very appreciated

Subject: Re: Matching Lats and Lons from two arrays
Posted by Brian Larsen on Tue, 26 Aug 2008 22:46:12 GMT
View Forum Message <> Reply to Message

Paolo,

thanks you were spot on. That bought me another x10 speed improvemnt
in that routine.

Thanks,

Brian

Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Matching Lats and Lons from two arrays

Posted by Gaurav on Wed, 27 Aug 2008 05:52:35 GMT
View Forum Message <> Reply to Message

Forgive my trespass but why is nobody suggesting the use of the DISTANCE_MEASURE or MAP_2POINTS functions of IDL? Or is the problem only with the length of the loop?

Or did I not get the problem right?

Gaurav

Subject: Re: Matching Lats and Lons from two arrays
Posted by Jeremy Bailin on Wed, 27 Aug 2008 12:49:47 GMT
View Forum Message <> Reply to Message

On Aug 26, 11:47 am, wilsona <awils...@bigred.unl.edu> wrote:
> I have 2 seperate arrays of Latittudes and Longitudes.
> CS_LATLON(0,4607) is one latitude array and dlat(192,139) is the
> other
> CS_LATLON(1,4607) is one longitude array and dlon(192,139) is the
> other.
> I want to index through each element in both CS_LATLON arrays and
> find
> which point(s) in the dlat and dlong arrays are closest.
> I tried using nested loops but that gave me 12 million+ loops which
> is
> too many for my liking. I now am trying search2d
> NUM_PNTS = N_ELEMENTS(CS_LATLON(0, *)) - 1
>
> FOR J = 0, NUM_PNTS DO BEGIN
> CLOSE_LATS = SEARCH2D(dlat, 0, 0, CS_LATLON(0,J),
> CS_LATLON(0,J), INCREASE=0.5, $
> DECREASE=0.5)
> lat1 = CS_LATLON(0,J) * PI / 180.0
> FOR K = 0, CLOSE_LATS DO BEGIN
> lat2 = dlat(K) * PI / 180.0
> d_long = CS_LATLON(1,J) - dlon(K)) * PI / 180.0
> DISTANCE = 10800.0 / PI * acos(sin(lat1) * sin(lat2)
> +
> cos(lat1) * $
> cos(lat2) * cos(d_long))
> ENDFOR ; K
> ENDFOR ; J
> This is not working they way I would like. Any suggestions on this
> would be greatly appreciated.

You might find WITHINSPHRAD in JBIU useful:

http://astroconst.org/jbiu/jbiu-doc/astro/withinsphrad.html

-Jeremy.

Subject: Re: Matching Lats and Lons from two arrays
Posted by JD Smith on Fri, 05 Sep 2008 22:53:53 GMT
View Forum Message <> Reply to Message

I should point out that the MATCH_2D routine mentioned in the above thread and at:

http://www.dfanning.com/code_tips/matchlists.html

assumes euclidean distance measures apply, which of course is *not* strictly correct on a sphere. For points of latitude and longitude (or ra/dec), this will work for small patches of earth (or sky), but if you are near either pole, or cover appreciable areas, the 2D distance measure will falter.

You can "fake" a "pretty good" euclidean distance measure by pre-multiplying all longitudes by the cos of latitude. This works if the latitude range is not overly large, and if you don't cross any poles. Do do this correctly would require replacing the d=dx^2 + dy^2 with:

d = 2 asin(sqrt(sin(ddec/2)^2 + cos(dec1)cos(dec2)sin(dra/2)^2))

which is obviously much costlier to evaluate. More importantly, you'll need to think a bit about whether the stock "platte carre" projection (i.e. (ra, dec) -> (x,y)) is wasteful or not for your distribution of points on the sphere, or adversely affects your matching radius criterion. Adopting another projection prior to binning and running the histogram might offer gains in efficiency and more "uniform" distance performance (the issue being that square bins in a projected sphere do not have constant area). It should be straightforward to "wrap-around" the poles.

JD
