
Subject: Re: Finding the mean of a set of images
Posted by [Craig Markwardt](#) on Tue, 22 Oct 2002 13:33:02 GMT
[View Forum Message](#) <> [Reply to Message](#)

David Oesch <oesch@giub.unibe.ch> writes:

> Hello outthere,
>
> I know this topic was up before, but all I could find in the list was to
> go for CALL_EXTERNAL and use a FORTRAN etc. program. Here's my problem:
> Does anyone have an algorithm for finding the mean/standarddeviation etc
> at each pixel position for a set of equal size 2-D images? Currently the
> only way I have to do this is to extract all the values for a given
> pixel position into a 1-D array and find the mean/standarddeviation etc
> on that. Doing it pixel by pixel like this is inefficient in IDL so I am
> looking for an *array* based algorithm that would find all
> the mean/standarddeviation etc in parallel.
> Any progs so far in IDL for this problem?..or a decent fortran or C program?

Sure, if you stack your image into a 3D image cube, then you would
have something like IMAGE = FLTARR(NX, NY, NIMAGES)

Then the mean image is:

```
mean = total(image,3)/nimages
```

The standard deviation is:

```
meancube = rebin(reform(mean,nx,ny,1),nx,ny,nimages)  
std = sqrt(total((image - meancube)^2,3)/(nimages-1))
```

Now, what you meant by "etc" can get a little hairier. If you want to
do median you are probably in trouble, but min and max are easy too:

```
minimage = image(*,*,0)  
maximage = minimage  
for i = 1, nimages-1 do begin  
  minimage = minimage < image(*,*,i)  
  maximage = maximage > image(*,*,i)  
endfor
```

It's a loop, but unless you have a bazillion images, it will be fast.

Craig

--

Subject: Re: Finding the mean of a set of images
Posted by [David Oesch](#) on Tue, 22 Oct 2002 15:23:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

Craig Markwardt wrote:

```
> David Oesch <oesch@giub.unibe.ch> writes:
>
>> Hello outthere,
>>
>> I know this topic was up before, but all I could find in the list was to
>> go for CALL_EXTERNAL and use a FORTRAN etc. program. Here's my problem:
>> Does anyone have an algorithm for finding the mean/standarddeviation etc
>> at each pixel position for a set of equal size 2-D images? Currently the
>> only way I have to do this is to extract all the values for a given
>> pixel position into a 1-D array and find the mean/standarddeviation etc
>> on that. Doing it pixel by pixel like this is inefficient in IDL so I am
>> looking for an *array* based algorithm that would find all
>> the mean/standarddeviation etc in parallel.
>> Any progs so far in IDL for this problem?..or a decent fortran or C program?
>>
>
>
> Sure, if you stack your image into a 3D image cube, then you would
> have something like IMAGE = FLTARR(NX, NY, NIMAGES)
>
> Then the mean image is:
>
> mean = total(image,3)/nimages
>
> The standard deviation is:
>
> meancube = rebin(reform(mean,nx,ny,1),nx,ny,nimages)
> std = sqrt(total((image - meancube)^2,3)/(nimages-1))
>
> Now, what you meant by "etc" can get a little hairier. If you want to
> do median you are probably in trouble, but min and max are easy too:
>
> minimage = image(*,*,0)
> maximage = minimage
> for i = 1, nimages-1 do begin
>   minimage = minimage < image(*,*,i)
>   maximage = maximage > image(*,*,i)
```

```
> endfor
>
> It's a loop, but unless you have a bazillion images, it will be fast.
>
> Craig
>
```

Yes, I got that one with TOTAL, the problem is, it only works with imagesets where all datalayers consists of 100% valid pixels...but this is not always the case...

to eliminate those values, I came as far as

```
mean = total((image NE -9999),3)/nimages
```

but the problem is, nimages needs to be changed to...

up to now, I use the following simple function to get to my data...but it is time consuming ..

```
function mean3d,file,nvalid=nvalid
```

```

,*****
; finding the mean at each pixel position for a set of equal size 2-D images
,*****
;file= 3d array
;nvalid= novalid values
;calling sequence: a=mean3d(array,nvalid=-9999)
```

```
s=SIZE(file)
outarray=FLTARR(s(1),s(2))

i=0!
FOR i=0,fix(s(1))-1 DO BEGIN
  j=0!
  FOR j=0,fix(s(2))-1 DO BEGIN
    check=WHERE(file(i,j,*) NE nvalid,count_check)
    IF count_check GT 0 THEN BEGIN
      outarray(i,j)=MEAN(file(i,j,check))
    ENDIF
  ENDFOR
ENDFOR
ENDFOR
RETURN,outarray

END
```

Subject: Re: Finding the mean of a set of images

Posted by [Jaco van Gorkom](#) on Tue, 22 Oct 2002 16:33:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
news:onznt6k38x.fsf@cow.physics.wisc.edu...

> David Oesch <oesch@giub.unibe.ch> writes:

>> ...

>> Does anyone have an algorithm for finding the mean/standard deviation etc
>> at each pixel position for a set of equal size 2-D images? Currently the
>> only way I have to do this is to extract all the values for a given
>> pixel position into a 1-D array and find the mean/standard deviation etc
>> on that. Doing it pixel by pixel like this is inefficient in IDL so I am
>> looking for an *array* based algorithm that would find all
>> the mean/standard deviation etc in parallel. ...

>

> Sure, if you stack your image into a 3D image cube, then you would
> have something like IMAGE = FLTARR(NX, NY, NIMAGES)

>

> Then the mean image is:

>

> mean = total(image,3)/nimages

>

> The standard deviation is:

>

> meancube = rebin(reform(mean,nx,ny,1),nx,ny,nimages)

> std = sqrt(total((image - meancube)^2,3)/(nimages-1))

>

> Now, what you meant by "etc" can get a little hairier. If you want to
> do median you are probably in trouble, but min and max are easy too:

>

> minimage = image(*,*,0)

> maximage = minimage

> for i = 1, nimages-1 do begin

> minimage = minimage < image(*,*,i)

> maximage = maximage > image(*,*,i)

> endfor

I believe that IDL 5.5 offers the luxury of

maximage = MAX(image, MIN=minimage, DIMENSION=3)

As for median, well, there was a thread on "Finding the median of a set
of images" back in '96. Should be valid still, I guess. Improvising with
transpose(), reform(), median(image, Nimages) and rebin() should be fun, but
not be very fast.

Jaco

Subject: Re: Finding the mean of a set of images
Posted by [Jaco van Gorkom](#) on Tue, 22 Oct 2002 17:48:51 GMT
[View Forum Message](#) <> [Reply to Message](#)

"David Oesch" <oesch@giub.unibe.ch> wrote in message
news:3DB56D82.2090407@giub.unibe.ch...
> Craig Markwardt wrote:
>> David Oesch <oesch@giub.unibe.ch> writes:
>>> ...
>>> Does anyone have an algorithm for finding the mean/standarddeviation etc
>>> at each pixel position for a set of equal size 2-D images? Currently the
>>> only way I have to do this is to extract all the values for a given
>>> pixel position into a 1-D array and find the mean/standarddeviation etc
>>> on that. Doing it pixel by pixel like this is inefficient in IDL so I am
>>> looking for an *array* based algorithm that would find all
>>> the mean/standarddeviation etc in parallel.
>>
>> Sure, if you stack your image into a 3D image cube, then you would
>> have something like IMAGE = FLTARR(NX, NY, NIMAGES)
>>
>> Then the mean image is:
>>
>> mean = total(image,3)/nimages
>>
>> ...
> Yes, I got that one with TOTAL, the problem is, it only works with
> imagesets where all datalayers consists of 100% valid pixels...but this
> is not always the case...
> to eliminate those values, I came as far as
>
> mean = total((image NE -9999),3)/nimages
>
> but the problem is, nimages needs to be changed to...

Yes. I suppose you mean total(image * (image NE -9999),3....

How about
valid = image NE -9999
mean = TOTAL(image * valid, 3) / TOTAL(valid, 3)

Or, if you would use NaN (Not-a-Number, floating point!) as missing value:
mean = (SMOOTH(image, [1,1,Nimages], /NaN))[* , *, Nimages/2]
For that to work you need to enforce an odd number of images.

cheers,
jaco

Subject: Re: Finding the mean of a set of images
Posted by [David Oesch](#) on Wed, 23 Oct 2002 05:18:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

<html>
<head>
</head>
<body>

Jaco van Gorkom wrote:

<blockquote type="cite" cite="mid:ap3uhr\$erme\$1@zam602.zam.kfa-juelich.de">

<pre wrap="">"Craig Markwardt" <a class="moz-txt-link-rfc2396E"

href="mailto:craigmnet@cow.physics.wisc.edu"><craigmnet@cow.physics.wisc.edu>

wrote in message
<a class="moz-txt-link-freetext"

href="news:onznt6k38x.fsf@cow.physics.wisc.edu">news:onznt6k38x.fsf@cow.physics.wisc.edu
...
</pre>

<blockquote type="cite">

<pre wrap="">David Oesch <a class="moz-txt-link-rfc2396E"

href="mailto:oesch@giub.unibe.ch"><oesch@giub.unibe.ch> writes:
</pre>

<blockquote type="cite">

<pre wrap=""> ...
Does anyone have an algorithm for finding the mean/standarde deviation
etc
at each pixel position for a set of equal size 2-D images? Currently the
only way I have
to do this is to extract all the values for a given
pixel position into a 1-D array and find the
mean/standarde deviation etc
on that. Doing it pixel by pixel like this is inefficient in IDL so I
am
looking for an *array* based algorithm that would find all
the mean/standarde deviation
etc in parallel. ...
</pre>

</blockquote>

<pre wrap="">Sure, if you stack your image into a 3D image cube, then you would
have
something like IMAGE = FLTARR(NX, NY, NIMAGES)

Then the mean image is:

mean = total(image,3)/nimages

The standard deviation is:

 meancube =
rebin(reform(mean,nx,ny,1),nx,ny,nimages)
 std = sqrt(total((image -
meancube)^2,3)/(nimages-1))

Now, what you meant by "etc" can get a little hairier. If you
want to
do median you are probably in trouble, but min and max are easy too:

minimage = image(*,*,0)
 maximage = minimage
 for i = 1, nimages-1 do begin

minimage = minimage < image(*,*,i)
 maximage = maximage > image(*,*,i)

endfor
</pre>

</blockquote>

<pre wrap=""><!-->
I believe that IDL 5.5 offers the luxury of
 maximage =
MAX(image, MIN=minimage, DIMENSION=3)

As for median, well, there was a thread on
"Finding the median of a set
of images" back in '96. Should be valid still, I guess. Improvising
with
transpose(), reform(), median(image, Nimages) and rebin() should be fun, but
not be
very fast.

 Jaco

</pre>

</blockquote>

Ok, I think now is the time to upgrade from IDL 5.4 to 5.5 :-).

Thanks for your tips

Dave

</body>
</html>

Subject: Re: Finding the mean of a set of images
Posted by [David Oesch](#) on Wed, 23 Oct 2002 14:28:10 GMT
[View Forum Message](#) <> [Reply to Message](#)

```
<html>
<head>
</head>
<body>
<br>
<br>
Jaco van Gorkom wrote:<br>
<blockquote type="cite" cite="mid:ap3uhr$erme$1@zam602.zam.kfa-juelich.de">
  <pre wrap="">"Craig Markwardt" <a class="moz-txt-link-rfc2396E"
href="mailto:craigmnet@cow.physics.wisc.edu">&lt;craigmnet@cow.physics.wisc.edu&gt;</a>
wrote in message<br><a class="moz-txt-link-freetext"
href="news:onznt6k38x.fsf@cow.physics.wisc.edu">news:onznt6k38x.fsf@cow.physics.wisc.edu
</a>...<br></pre>
  <blockquote type="cite">
    <pre wrap="">David Oesch <a class="moz-txt-link-rfc2396E"
href="mailto:oesch@giub.unibe.ch">&lt;oesch@giub.unibe.ch&gt;</a> writes:<br></pre>
    <blockquote type="cite">
      <pre wrap=""> ...<br>Does anyone have an algorithm for finding the mean/standarde deviation
etc<br>at each pixel position for a set of equal size 2-D images? Currently the<br>only way I have
to do this is to extract all the values for a given<br>pixel position into a 1-D array and find the
mean/standarde deviation etc<br>on that. Doing it pixel by pixel like this is inefficient in IDL so I
am<br>looking for an *array* based algorithm that would find all<br>the mean/standarde deviation
etc in parallel.  ...<br></pre>
    </blockquote>
    <pre wrap="">Sure, if you stack your image into a 3D image cube, then you would<br>have
something like IMAGE = FLTARR(NX, NY, NIMAGES)<br><br>Then the mean image is:<br><br>
mean = total(image,3)/nimages<br><br>The standard deviation is:<br><br>
meancube =
rebin(reform(mean,nx,ny,1),nx,ny,nimages)<br> std = sqrt(total((image -
meancube)^2,3)/(nimages-1))<br><br>Now, what you meant by "etc" can get a little hairier. If you
want to<br>do median you are probably in trouble, but min and max are easy too:<br><br>
minimage = image(*,*,0)<br> maximage = minimage<br> for i = 1, nimages-1 do begin<br>
minimage = minimage &lt; image(*,*,i)<br>    maximage = maximage &gt; image(*,*,i)<br>
endfor<br></pre>
    </blockquote>
    <pre wrap=""><!--><br>I believe that IDL 5.5 offers the luxury of<br> maximage =
MAX(image, MIN=minimage, DIMENSION=3)<br><br>As for median, well, there was a thread on
"Finding the median of a set<br>of images" back in '96. Should be valid still, I guess. Improvising
with<br>transpose(), reform(), median(image, Nimages) and rebin() should be fun, but<br>not be
very fast.<br><br> Jaco<br><br><br></pre>
  </blockquote>
</pre>
```

```
<pre class="moz-signature" cols="$mailwrapcol">Well it works fine in 5.5 , but this feature is  
not explained in the help of the idl 5.5 WIN/PC edition  
... in 5.5 the help is obviously&nbsp;still the help of 5.4...
```

```
</pre>
```

```
<br>
```

```
</body>
```

```
</html>
```
