
Subject: ENVI_INIT_TILE tiling problem
Posted by [a.l.j.ford](#) on Sat, 14 Mar 2009 13:10:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

I'm using interpolation to fill some holes in elevation data. Because of memory limitations I'm trying Envi's tiling capability for the first time! My code compiles OK, but when I run it (I have it embedded in an Envi User Function) I get this error in my IDL window:

```
% Variable is undefined: F_NS.  
% Execution halted at: ENVI_INIT_TILE
```

Here is the offending section of code. I have no idea where F_NS is (I'd understand if it was simply NS, i.e. number of samples) and despite searching I cannot deduce what is wrong or how to fix it. Does anyone have any ideas?? BTW, the elevation data has a single band and my_pos is set to [0].

```
tile_id=ENVI_INIT_TILE(output_DSM, my_pos)  
FOR i=0, num_tiles-1 DO BEGIN  
  tile_data_interp=ENVI_GET_TILE(tile_id, i)  
  
  ;Processing within Tiling  
  
  tile_data_interp = REPLICATE(0.0, dims[2], dims[4])  
  tile_data_interp = TRI_SURF(output_DSM, /REGULAR, XGRID=[1, 1],  
  YGRID=[1, 1], NX=dims[2], NY=dims[4])  
  
  ; Close Tiling  
  
ENDFOR  
ENVI_TILE_DONE, tile_id
```

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [Jean H.](#) on Mon, 16 Mar 2009 12:47:34 GMT
[View Forum Message](#) <> [Reply to Message](#)

```
> But this now results in the following error, which I don't understand  
> and can't find a reference to (including in tri_surf.pro, unless I've  
> missed something?). Any ideas?  
>  
> % Array dimensions must be greater than 0.  
> % Error occurred at: TRI_SURF      136 C:\Program Files\ITT  
> \IDL64\lib\tri_surf.pro  
>  
> output_DSM is certainly a 2D array, as I'm able to TVSCL it to view  
> the contents.
```

>
> output_DSM contains lots of holes (value of 0) which I'd like to
> interpolate over using TRI_SURF.

lines 135 and 136 are:

```
if n_elements(xgrid) eq 2 then begin
  x = findgen(nx) * xgrid[1] + xgrid[0]
  NX being computed from the Z input (your output_dsm array)
```

so, the problem is your Z input... double check, just before calling
tri_surf, what is the content and size of output_dsm!

Jean

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [a.l.j.ford](#) on Mon, 16 Mar 2009 20:57:12 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Mar 16, 12:47 pm, "Jean H." <jghas...@DELTHIS.ucalgary.ANDTHIS.ca>
wrote:

>> But this now results in the following error, which I don't understand
>> and can't find a reference to (including in tri_surf.pro, unless I've
>> missed something?). Any ideas?

```
>
>> % Array dimensions must be greater than 0.
>> % Error occurred at: TRI_SURF      136 C:\Program Files\ITT
>> \IDL64\lib\tri_surf.pro
```

>
>> output_DSM is certainly a 2D array, as I'm able to TVSCL it to view
>> the contents.

>
>> output_DSM contains lots of holes (value of 0) which I'd like to
>> interpolate over using TRI_SURF.

```
>
> lines 135 and 136 are:
>     if n_elements(xgrid) eq 2 then begin
>       x = findgen(nx) * xgrid[1] + xgrid[0]
> NX being computed from the Z input (your output_dsm array)
```

>
> so, the problem is your Z input... double check, just before calling
> tri_surf, what is the content and size of output_dsm!

>
> Jean

Hi Jean,

Yes, yet again you were absolutely correct. the problem was with my "output_dsm". In order to get a FID from output_dsm I used ENVI_ENTER_DATA (maybe this isn't the best way??), which then seemed to prevent output_dsm being used as an array. Therefore, before I used ENVI_ENTER_DATA I made a copy of output_dsm, called output_dsm_copy! This meant it was preserved as an array. This might not be the best way to do things (?), but it worked.

Things are now working mostly OK and the tiling and interpolation appear to complete, except that the zero pixels in my array we're interpolated over. Therefore I changed my interpolation to the following (I decided to go with MIN_CURVE_SURF in this example, but the same should be true for TRI_SURF):

```
tile_id=ENVI_INIT_TILE(fid_output_DSM, my_pos,
num_tiles=number_of_tiles)
FOR i=0, number_of_tiles-1 DO BEGIN
tile_data_interp=ENVI_GET_TILE(tile_id, i)

;Processing within Tiling

;tile_data_interp = REPLICATE(0.0, dims[2], dims[4])

index= WHERE (output_dsm_copy GT 0.0)

x = index MOD DIMS[2]
y = index/DIMS[4]

z = output_dsm_copy [index]

tile_data_interp = MIN_CURVE_SURF (z, x, y, gs=[1,1],bounds=[1,1,DIMS
[2],DIMS[4]])

;tile_data_interp = TRI_SURF(output_DSM, /REGULAR, XGRID=[1, 1],
YGRID=[1, 1], NX=dims[2], NY=dims[4])

; Close Tiling

ENDFOR
ENVI_TILE_DONE, tile_id
```

Can you spot the problem?? When I run it the interpolation runs out of memory for creating the array (% Unable to allocate memory: to make array.

Not enough space). This is because I'm using DIMS for the original file outside of the tiling... whereas I need to use different, smaller DIMS within the tiles (the x,y, dimensions of the tiles themselves). How can I get the tile dimensions and use them here??

Many thanks again!

Subject: Re: ENVI_INIT_TILE tiling problem

Posted by [jeffnetles4870](#) on Tue, 17 Mar 2009 01:16:46 GMT

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

On Mar 16, 4:57 pm, a.l.j.f...@gmail.com wrote:

> On Mar 16, 12:47 pm, "Jean H." <jghas...@DELTHIS.ucalgary.ANDTHIS.ca>
> wrote:

>
>
>

>>> But this now results in the following error, which I don't understand
>>> and can't find a reference to (including in tri_surf.pro, unless I've
>>> missed something?). Any ideas?

>
>>> % Array dimensions must be greater than 0.
>>> % Error occurred at: TRI_SURF 136 C:\Program Files\ITT
>>> \IDL64\lib\tri_surf.pro

>
>>> output_DSM is certainly a 2D array, as I'm able to TVSCL it to view
>>> the contents.

>
>>> output_DSM contains lots of holes (value of 0) which I'd like to
>>> interpolate over using TRI_SURF.

>
>> lines 135 and 136 are:
>> if n_elements(xgrid) eq 2 then begin
>> x = findgen(nx) * xgrid[1] + xgrid[0]
>> NX being computed from the Z input (your output_dsm array)

>
>> so, the problem is your Z input... double check, just before calling
>> tri_surf, what is the content and size of output_dsm!

>
>> Jean

>
> Hi Jean,

>
> Yes, yet again you were absolutely correct. the problem was with my
> "output_dsm". In order to get a FID from output_dsm I used
> ENVI_ENTER_DATA (maybe this isn't the best way??), which then seemed
> to prevent output_dsm being used as an array. Therefore, before I used
> ENVI_ENTER_DATA I made a copy of output_dsm, called output_dsm_copy!
> This meant it was preserved as an array. This might not be the best
> way to do things (?), but it worked.

>

```

> Things are now working mostly OK and the tiling and interpolation
> appear to complete, except that the zero pixels in my array we're
> interpolated over. Therefore I changed my interpolation to the
> following (I decided to go with MIN_CURVE_SURF in this example, but
> the same should be true for TRI_SURF):
>
> tile_id=ENVI_INIT_TILE(fid_output_DSM, my_pos,
> num_tiles=number_of_tiles)
> FOR i=0, number_of_tiles-1 DO BEGIN
> tile_data_interp=ENVI_GET_TILE(tile_id, i)
>
> ;Processing within Tiling
>
> ;tile_data_interp = REPLICATE(0.0, dims[2], dims[4])
>
> index= WHERE (output_dsm_copy GT 0.0)
>
> x = index MOD DIMS[2]
> y = index/DIMS[4]
>
> z = output_dsm_copy [index]
>
> tile_data_interp = MIN_CURVE_SURF (z, x, y, gs=[1,1],bounds=[1,1,DIMS
> [2],DIMS[4]])
>
> ;tile_data_interp = TRI_SURF(output_DSM, /REGULAR, XGRID=[1, 1],
> YGRID=[1, 1], NX=dims[2], NY=dims[4])
>
> ; Close Tiling
>
> ENDFOR
> ENVI_TILE_DONE, tile_id
>
> Can you spot the problem?? When I run it the interpolation runs out of
> memory for creating the array (% Unable to allocate memory: to make
> array.
> Not enough space). This is because I'm using DIMS for the original
> file outside of the tiling... whereas I need to use different, smaller
> DIMS within the tiles (the x,y, dimensions of the tiles themselves).
> How can I get the tile dimensions and use them here??
>
> Many thanks again!

```

There's a couple ways you can figure this out. You can actually control the interleave of the tile, which would mean you would know the tile dimensions most of the time. This isn't true 100% of the time, but i've never run into a case where it wasn't. But anyway, a pretty reliable way to do this would be to put this code in your tile

loop:

```
if i eq 0 then s = size(tile_data, /dimensions)
```

which would make s[0] the size in the x direction and s[1] size in the y direction. You can feed that into min_curve_surf or tri_surf later.

Subject: Re: ENVI_INIT_TILE tiling problem

Posted by [Jean H.](#) on Tue, 17 Mar 2009 14:09:23 GMT

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

Hi,

```
>> Yes, yet again you were absolutely correct. the problem was with my
>> "output_dsm". In order to get a FID from output_dsm I used
>> ENVI_ENTER_DATA (maybe this isn't the best way??), which then seemed
>> to prevent output_dsm being used as an array. Therefore, before I used
>> ENVI_ENTER_DATA I made a copy of output_dsm, called output_dsm_copy!
>> This meant it was preserved as an array. This might not be the best
>> way to do things (?), but it worked.
```

If you have a file, use ENVI_OPEN_FILE, it will give you a valid FID. Otherwise, you can use ENVI_SELECT or save your array and open it this way. You don't want to have the same content twice in memory!

```
>>
>> Things are now working mostly OK and the tiling and interpolation
>> appear to complete, except that the zero pixels in my array we're
>> interpolated over. Therefore I changed my interpolation to the
>> following (I decided to go with MIN_CURVE_SURF in this example, but
>> the same should be true for TRI_SURF):
>>
>> tile_id=ENVI_INIT_TILE(fid_output_DSM, my_pos,
>> num_tiles=number_of_tiles)
>> FOR i=0, number_of_tiles-1 DO BEGIN
>> tile_data_interp=ENVI_GET_TILE(tile_id, i)
```

ok, you get the data

```
>>
>> ;Processing within Tiling
>> index= WHERE (output_dsm_copy GT 0.0)
>>
>> x = index MOD DIMS[2]
>> y = index/DIMS[4]
>>
>> z = output_dsm_copy [index]
```

? Don't you want to use the tile data?? tile_data_interp
What is the point of tiling your data if you end up using the whole array?

```
>>  
>> tile_data_interp = MIN_CURVE_SURF (z, x, y, gs=[1,1],bounds=[1,1,DIMS  
>> [2],DIMS[4]])
```

You just erase the data extracted... ok, result of the interpolation

```
>> ; Close Tiling  
>>  
>> ENDFOR  
>> ENVI_TILE_DONE, tile_id
```

but you have not saved the content of the tile.
You had the array, have modified it and that's it!

Read the help file.. you have the choice of 1) create an array of the size of the complete image, store the result of each tile in it and load it in memory (using envi_enter_data this time), or 2) open the output file (new file), write the content of each tile in it using writeu, and when you are done processing every tiles, set up the header.

Jean

```
>>  
>> Can you spot the problem?? When I run it the interpolation runs out of  
>> memory for creating the array (% Unable to allocate memory: to make  
>> array.  
>> Not enough space). This is because I'm using DIMS for the original  
>> file outside of the tiling... whereas I need to use different, smaller  
>> DIMS within the tiles (the x,y, dimensions of the tiles themselves).  
>> How can I get the tile dimensions and use them here??  
>>  
>> Many thanks again!  
>  
> There's a couple ways you can figure this out. You can actually  
> control the interleave of the tile, which would mean you would know  
> the tile dimensions most of the time. This isn't true 100% of the  
> time, but i've never run into a case where it wasn't. But anyway, a  
> pretty reliable way to do this would be to put this code in your tile  
> loop:  
>  
> if i eq 0 then s = size(tile_data, /dimensions)  
>  
> which would make s[0] the size in the x direction and s[1] size in the
```

> y direction. You can feed that into min_curve_surf or tri_surf later.

Subject: Re: ENVI_INIT_TILE tiling problem

Posted by [a.l.j.ford](#) on Tue, 17 Mar 2009 23:04:24 GMT

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

On Mar 17, 2:09 pm, "Jean H." <jghas...@DELTHIS.ucalgary.ANDTHIS.ca> wrote:

> Hi,

>

>>> Yes, yet again you were absolutely correct. the problem was with my
>>> "output_dsm". In order to get a FID from output_dsm I used
>>> ENVI_ENTER_DATA (maybe this isn't the best way??), which then seemed
>>> to prevent output_dsm being used as an array. Therefore, before I used
>>> ENVI_ENTER_DATA I made a copy of output_dsm, called output_dsm_copy!
>>> This meant it was preserved as an array. This might not be the best
>>> way to do things (?), but it worked.

>

> If you have a file, use ENVI_OPEN_FILE, it will give you a valid FID.
> Otherwise, you can use ENVI_SELECT or save your array and open it this
> way. You don't want to have the same content twice in memory!

>

>

>

>>> Things are now working mostly OK and the tiling and interpolation
>>> appear to complete, except that the zero pixels in my array we're
>>> interpolated over. Therefore I changed my interpolation to the
>>> following (I decided to go with MIN_CURVE_SURF in this example, but
>>> the same should be true for TRI_SURF):

>

```
>>> tile_id=ENVI_INIT_TILE(fid_output_DSM, my_pos,  
>>> num_tiles=number_of_tiles)  
>>> FOR i=0, number_of_tiles-1 DO BEGIN  
>>> tile_data_interp=ENVI_GET_TILE(tile_id, i)
```

>

> ok, you get the data

>

>

>

```
>>> ;Processing within Tiling  
>>> index= WHERE (output_dsm_copy GT 0.0)
```

>

```
>>> x = index MOD DIMS[2]  
>>> y = index/DIMS[4]
```

>

```
>>> z = output_dsm_copy [index]
```

>

```

> ? Don't you want to use the tile data??  tile_data_interp
> What is the point of tiling your data if you end up using the whole array?
>
>
>
>>> tile_data_interp = MIN_CURVE_SURF (z, x, y, gs=[1,1],bounds=[1,1,DIMS
>>> [2],DIMS[4]])
>
> You just erase the data extracted... ok, result of the interpolation
>
>>> ; Close Tiling
>
>>> ENDFOR
>>> ENVI_TILE_DONE, tile_id
>
> but you have not saved the content of the tile.
> You had the array, have modified it and that's it!
>
> Read the help file.. you have the choice of 1) create an array of the
> size of the complete image, store the result of each tile in it and load
> it in memory (using envi_enter_data this time), or 2) open the output
> file (new file), write the content of each tile in it using writeu, and
> when you are done processing every tiles, set up the header.
>
> Jean
>
>
>
>>> Can you spot the problem?? When I run it the interpolation runs out of
>>> memory for creating the array (% Unable to allocate memory: to make
>>> array.
>>> Not enough space). This is because I'm using DIMS for the original
>>> file outside of the tiling... whereas I need to use different, smaller
>>> DIMS within the tiles (the x,y, dimensions of the tiles themselves).
>>> How can I get the tile dimensions and use them here??
>
>>> Many thanks again!
>
>> There's a couple ways you can figure this out. You can actually
>> control the interleave of the tile, which would mean you would know
>> the tile dimensions most of the time. This isn't true 100% of the
>> time, but i've never run into a case where it wasn't. But anyway, a
>> pretty reliable way to do this would be to put this code in your tile
>> loop:
>
>> if i eq 0 then s = size(tile_data, /dimensions)
>
>> which would make s[0] the size in the x direction and s[1] size in the

```

```
>> y direction. You can feed that into min_curve_surf or tri_surf later.  
>  
>
```

Yes, very good points!!

This is where I am now. So far I don't get any errors, but the processing takes forever and never finishes, as if it has hung. Am I still going wrong somewhere. I have to say I'm finding the mix of tiling and interpolation somewhat challenging, so please forgive me! I'm aware that you can use ENVI_REPORT_INIT, ENVI_REPORT_INC and ENVI_REPORT_STAT to show progress, but right now I dare not use them, for fear of introducing even more errors! One step at a time I say! Many thanks for your continued help.

```
openw, unit, 'output_test', /get_lun
```

```
;Open Tiling
```

```
tile_id=ENVI_INIT_TILE(fid_output_DSM, my_pos,  
num_tiles=number_of_tiles)
```

```
FOR i=0, number_of_tiles-1 DO BEGIN
```

```
tile_data=ENVI_GET_TILE(tile_id, i)
```

```
;Processing within Tiling
```

```
if i eq 0 then tile_size = size(tile_data, /dimensions)
```

```
index= WHERE (tile_data GT 0.0)
```

```
x = index MOD tile_size[0]
```

```
y = index/tile_size[1]
```

```
z = tile_data [index]
```

```
tile_data_interpolated = MIN_CURVE_SURF (z, x, y, gs=[1,1],bounds=  
[1,1,tile_size[0],tile_size[1]])
```

```
writeu, unit, tile_data_interpolated
```

```
ENDFOR
```

```
free_lun, unit
```

```
envi_setup_head, fname=output_test, ns=ns, nl=nl, nb=nb, $  
data_type=data_type, offset=0, interleave=0, $
```

```
xstart=xstart+dims[1], ystart=ystart+dims[3], $
descrip='Interpolation output', /write, /open
```

ENVI_TILE_DONE, tile_id

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [David Fanning](#) on Tue, 17 Mar 2009 23:09:06 GMT
[View Forum Message](#) <> [Reply to Message](#)

a.l.j.ford@gmail.com writes:

> This is where I am now. So far I don't get any errors, but the
> processing takes forever and never finishes, as if it has hung.

I know next to nothing about ENVI, but I have never
once used MIN_CURVE_SURF that it didn't occur to
me that I would have grandchildren sooner than I
would get a smooth surface. :-(

Cheers,

David

--

David Fanning, Ph.D.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [a.l.j.ford](#) on Wed, 18 Mar 2009 00:08:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

On 17 Mar, 23:09, David Fanning <n...@dfanning.com> wrote:

> a.l.j.f...@gmail.com writes:
>> This is where I am now. So far I don't get any errors, but the
>> processing takes forever and never finishes, as if it has hung.
>
> I know next to nothing about ENVI, but I have never
> once used MIN_CURVE_SURF that it didn't occur to
> me that I would have grandchildren sooner than I
> would get a smooth surface. :-(
>
> Cheers,
>
> David

>
> --
> David Fanning, Ph.D.
> Coyote's Guide to IDL Programming (www.dfanning.com)
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

LOL It could be the same in my case... I'm still waiting! :-) I'll let it run for as long as I can stand it, then try something less time consuming than min_curve_surf.

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [a.l.j.ford](#) on Wed, 18 Mar 2009 19:47:47 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Mar 18, 12:08 am, a.l.j.f...@gmail.com wrote:
> On 17 Mar, 23:09, David Fanning <n...@dfanning.com> wrote:
>
>
>
>> a.l.j.f...@gmail.com writes:
>>> This is where I am now. So far I don't get any errors, but the
>>> processing takes forever and never finishes, as if it has hung.
>
>> I know next to nothing about ENVI, but I have never
>> once used MIN_CURVE_SURF that it didn't occur to
>> me that I would have grandchildren sooner than I
>> would get a smooth surface. :-(
>
>> Cheers,
>
>> David
>
>> --
>> David Fanning, Ph.D.
>> Coyote's Guide to IDL Programming (www.dfanning.com)
>> Sepore ma de ni thui. ("Perhaps thou speakest truth.")
>
> LOL It could be the same in my case... I'm still waiting! :-) I'll let
> it run for as long as I can stand it, then try something less time
> consuming than min_curve_surf.

OK, I eventually realised that I'd missed the " from around my output filename in envi_setup_head. It now reads:

```
envi_setup_head, fname='output_test'
```

I decided to try things out on a much smaller array size. My input

array with missing values was 100 x 100, which meant no Tiling was needed. Using both TRI_SURF and MIN_CURVE_SURF the process completed both times! SUCCESS! The output from TRI_SURF was dire though, and I

doubt I'll use it again. The output from MIN_CURVE_SURF looked very representative. So, I then tried TRI_SURF on a 500 x 600 array. The output from this was even worse! Here is the input, with missing values as 0 (black) to be interpolated over:

[IMG] <http://img156.imageshack.us/img156/8798/arraymissingvalues.jpg/>
[IMG]

And here is the result from TRI_SURF. Tiling kicked in and split the array into two. Not only is the interpolation of the missed values very poor, there is a big mis-match between the tiles, where elevations suddenly rise in a step.

[IMG] <http://img156.imageshack.us/img156/7532/arraytrisurf.jpg/> [IMG]

Can anyone tell me, is this an inevitable consequence of doing Interpolation withing Tiling, or have I done something wrong again?! I certainly hope there is a way around this.

In any case, TRI_SURF spat out this 500 x 600 interpolated array in no time at all... but when I did the exact same thing with MIN_CURVE_SURF it took so long I gave up. Then I tried a 500 x 250 array... same thing! Then just a 200 x 200 array... still so long (hours) I

gave up. This surely can't be right, or am I being naive?!

Could/should I be using different interpolation techniques? Is so, what would you recommend for the kind of data you see in my example above? I'm after nice smooth interpolation if possible.

Thanks again

Subject: Re: ENVI_INIT_TILE tiling problem
Posted by [d.poreh](#) on Fri, 10 Apr 2009 11:41:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Apr 10, 2:07 pm, nitink...@gmail.com wrote:
> On Mar 14, 11:11 pm, "Jeff N." <jeffnettles4...@gmail.com> wrote:
>
>
>
>
>

```

>> On Mar 14, 9:10 am, a.l.j.f...@gmail.com wrote:
>
>>> I'm using interpolation to fill some holes in elevation data. Because
>>> of memory limitations I'm trying Envi's tiling capability for the
>>> first time! My code compiles OK, but when I run it (I have it embedded
>>> in an Envi User Function) I get this error in my IDL window:
>
>>> % Variable is undefined: F_NS.
>>> % Execution halted at: ENVI_INIT_TILE
>
>>> Here is the offending section of code. I have no idea where F_NS is
>>> (I'd understand if it was simply NS, i.e. number of samples) and
>>> despite searching I cannot deduce what is wrong or how to fix it. Does
>>> anyone have any ideas?? BTW, the elevation data has a single band and
>>> my_pos is set to [0].
>
>>> tile_id=ENVI_INIT_TILE(output_DSM, my_pos)
>>> FOR i=0, num_tiles-1 DO BEGIN
>>> tile_data_interp=ENVI_GET_TILE(tile_id, i)
>
>>> ;Processing within Tiling
>
>>> tile_data_interp = REPLICATE(0.0, dims[2], dims[4])
>>> tile_data_interp = TRI_SURF(output_DSM, /REGULAR, XGRID=[1, 1],
>>> YGRID=[1, 1], NX=dims[2], NY=dims[4])
>
>>> ; Close Tiling
>
>>> ENDFOR
>>> ENVI_TILE_DONE, tile_id
>
>> I would check to make sure that the first argument to ENVI_INIT_TILE
>> is supplied correctly. You should be giving it the FID of the input
>> file, but the "output_DSM" doesn't sound like an input FID to me. In
>> fact, since you use it as an argument to TRI_SURF it looks like
>> output_DSM has to be an actual data array. So i think you're giving
>> ENVI_INIT_TILE an incorrect argument.
>
> Hi Jeff this is Nitin, I am a Research Scholar, and i am also facing
> same kind of problem during creation of DEM from geoeye pan data(I am
> using ENVI 4.6). When i run the process in lower level(5th level),
> process runs smoothly and result come out. but when i run the process
> in Maximum level(8th) at the end of the process a error appears as
> "Array dimension is must be greater than 0"
> will you suggest me, why this kind of error is occurring, is it due to
> heavy data size.

```

Folks

I also have a lot of problem for filing data. Does anybody could make an article to show how to do this analysis for single band?
Dave
