Subject: Re: Map image with a sparse array
Posted by Liam E. Gumley on Fri, 21 Jan 2000 08:00:00 GMT
View Forum Message <> Reply to Message

whdaffer@my-deja.com wrote:

> I have the following problem. I have an array of geographically
> colocated data (0.5 by 0.5 degree grid) that is the result of averaging
> all 253 swaths of one cycle of Topex data into this grid. Topex data has
> a small (3 or maybe 10km) swath, so the majority of the grid location
> (65%) are 'bad' in the sense that those grid elements contain no
> averaged data.
>
> I want to display this by mapping it using map_set/map_image. The old
> method simply 'tv'd the image to the screen and then finessed applying
> the continents/grid lines to the image. A bit of a boondogle, and not
> very upgradeable.
>
> The problem is that there seems to be no way to tell map_image (and
> map_patch too) that certain data (the 'bad' data value) should be
> excluded from whatever
> averaging/bilinear-interpolation/nearest-neighbor-chosing method is used
> and the 'mapped' image has places that are clearly corrupted by the
> presence of the bad data. The problem is ameliorated by use nearest
> neighbor rather than bilinear interpolation (i.e. bilinear=0) and I am
> setting compress=0, so that the inverse transformation is done on each
> pixel. Also, I've started out with a window set to the size of the input
> data array and with map_set,position=[0.,0,1,1] so that the mapping
> coordinate system occupies the entire window. These remedies I hit upon
> thinking that they would minimize the damage, and they have done that,
> but when I compare my results with the older, more 'pristine' but vastly
> less portible, upgradeable, maintainable method, there are big
> differences.
>
> The 'missing' keyword just sets elements outside the range input via
> the 'min' and 'max' keywords and those outside of the mapping
> coordinates to the bad value, it doesn't allow one to exclude data from
> the averaging/interpolation/chosing method.

Assuming
(1) Your data ('grid') is already on a 0.5 x 0.5 degree grid,
(2) The data range is 0.0 to 100.0 (adjust to taste),
(3) The missing value is -999.0 (adjust to taste),

```
image = bytscl(grid, min=0.0, max=100.0, top=!d.table_size-2) + 1B
index = where(grid eq -999.0, count)
if (count gt 0) then image[index] = 0B
```

```
map_set, /aitoff, /isotropic
mapped = map_image(image, startx, starty, xsize, ysize, $
  latmin=-90.0, latmax=90.0, lonmin=-180.0, lonmax=180.0, $
  compress=1, missing=0B, scale=0.05)

loadct, 13, bottom=1
tv, mapped, startx, starty, xsize=xsize, ysize=ysize
```

Cheers,
Liam.
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: Map image with a sparse array
Posted by Liam E. Gumley on Mon, 24 Jan 2000 08:00:00 GMT
View Forum Message <> Reply to Message

whdaffer@my-deja.com wrote:

> Can anyone see what I'm doing wrong?
>
> When I tv the mapped image and compare it to the unmapped image, it's
> clear that the mapped image has included some 'bad' data in the
> calculations of points within the grid.
>
> To be fair, this possibility is not explicitly excluded in the help
> file. If I'm reading it correctly, all it says is that output grid
> values which are within the valid mapping limits but which exceed input
> maxima will be set to the 'bad data' value, not that the input 'bad
> data' value will be excluded from caluclations.
>
> The effect is not large, but it isn't small either. The person I'm
> doing the work for noticed it right off.
>
> By the way, I did try your method first, (bytscl, then map_image) and
> got similar results, but thought that I should make the bytscl be the
> last step, since it reduces the 'color' resolution from 360 values to
> about 200. I'll go back and try your method again. Perhaps there's some
> magic that I'm not seeing.

William,

I pulled that section of code from a procedure that projects a 720x360
global grid with missing data onto various map projections. I cannot see any
hint that the missing data corrupted the resulting image:
ftp://origin.ssec.wisc.edu/pub/gumley/IDL/grid_project002.gi f

Cheers,

Liam.

--
Liam E. Gumley
Space Science and Engineering Center, University of Wisconsin-Madison
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: Map image with a sparse array
Posted by whdaffer on Mon, 24 Jan 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <3888FAA7.56DC4905@ssec.wisc.edu>,
"Liam E. Gumley" <Liam.Gumley@ssec.wisc.edu> wrote:
> whdaffer@my-deja.com wrote:
>
>> I have the following problem. I have an array of geographically
>> colocated data (0.5 by 0.5 degree grid) that is the result of
averaging
>> all 253 swaths of one cycle of Topex data into this grid. Topex data
has
>> a small (3 or maybe 10km) swath, so the majority of the grid
location
>> (65%) are 'bad' in the sense that those grid elements contain no
>> averaged data.
>>
>> I want to display this by mapping it using map_set/map_image. The
old
>> method simply 'tv'd the image to the screen and then finessed
applying
>> the continents/grid lines to the image. A bit of a boondogle, and
not
>> very upgradeable.
>>
>> The problem is that there seems to be no way to tell map_image
(and
>> map_patch too) that certain data (the 'bad' data value) should be
>> excluded from whatever
>> averaging/bilinear-interpolation/nearest-neighbor-chosing method is
used
>> and the 'mapped' image has places that are clearly corrupted by the
>> presence of the bad data. The problem is ameliorated by use nearest
>> neighbor rather than bilinear interpolation (i.e. bilinear=0) and I
am
>> setting compress=0, so that the inverse transformation is done on
each
>> pixel. Also, I've started out with a window set to the size of the
input

>> data array and with map_set,position=[0.,0,1,1] so that the mapping
>> coordinate system occupies the entire window. These remedies I hit upon
>> thinking that they would minimize the damage, and they have done that,
>> but when I compare my results with the older, more 'pristine' but vastly
>> less portible, upgradeable, maintainable method, there are big
>> differences.
>>
>> The 'missing' keyword just sets elements outside the range input via
>> the 'min' and 'max' keywords and those outside of the mapping
>> coordinates to the bad value, it doesn't allow one to exclude data from
>> the averaging/interpolation/chosing method.
>
> Assuming
> (1) Your data ('grid') is already on a 0.5 x 0.5 degree grid,
> (2) The data range is 0.0 to 100.0 (adjust to taste),
> (3) The missing value is -999.0 (adjust to taste),
>
> image = bytscl(grid, min=0.0, max=100.0, top=!d.table_size-2) + 1B
> index = where(grid eq -999.0, count)
> if (count gt 0) then image[index] = 0B
>
> map_set, /aitoff, /isotropic
> mapped = map_image(image, startx, starty, xsize, ysize, $
> latmin=-90.0, latmax=90.0, lonmin=-180.0, lonmax=180.0, $
> compress=1, missing=0B, scale=0.05)
>
> loadct, 13, bottom=1
> tv, mapped, startx, starty, xsize=xsize, ysize=ysize
>
> Cheers,
> Liam.
> http://cimss.ssec.wisc.edu/~gumley
>
>

This is basically what I'm doing now, except that I'm warping the image to the mapping coordinates first and then byte scaling it.

Here's my code:
(hlo=-180, hhi=180, which is at about the 3 sigma level of the basic data distribution, baddata=32767, rgb_missing is the color for the 'bad' data)

<quote>
; Monkey with the data.
good = where(gridgood NE baddata,ngood)
bad = where(gridgood EQ baddata, nbad)
gridgood[good] = hlo> gridgood[good] < hhi

; Warp the data to the mapped coordinate system
newim = map_image(gridgood,sx,sy,latmin=minlat,latmax=maxlat,$
lonmin=0,lonmax=359.5,/compress,$
min=hlo,max=hhi, missing=baddata)

bad = where(newim EQ baddata,nbad)

; Scale the map from rgb_lo to rgb_hi
newim = bytscl(newim,min=hlo,max=hhi,top=rgb_hi-rgb_lo)+rgb_lo
IF nbad NE 0 THEN newim[bad] = rgb_missing

</quote>

Can anyone see what I'm doing wrong?

When I tv the mapped image and compare it to the unmapped image, it's clear that the mapped image has included some 'bad' data in the calculations of points within the grid.

To be fair, this possibility is not explicitly excluded in the help file. If I'm reading it correctly, all it says is that output grid values which are within the valid mapping limits but which exceed input maxima will be set to the 'bad data' value, not that the input 'bad data' value will be excluded from caluclations.

The effect is not large, but it isn't small either. The person I'm doing the work for noticed it right off.

By the way, I did try your method first, (bytscl, then map_image) and got similar results, but thought that I should make the bytscl be the last step, since it reduces the 'color' resolution from 360 values to about 200. I'll go back and try your method again. Perhaps there's some magic that I'm not seeing.

William