
Subject: Re: map_grid limitations

Posted by [Luis Alonso](#) on Thu, 11 Apr 2002 15:58:59 GMT

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

Hi!

Why don't you try, the same way you change the labels, you can change the header of your map in which the coordinates are stored. Just subtract 180 to the longitude.

In case it is stored in a header file where it gives the first pixel coordinate (such as ENVI format) just subtract 180 in the corresponding line (x_start for the ENVI case)

In case you have a matching array (lat_coord for example) storing the coordinates, just do:

```
lat_coord = lat_coord - 180
```

Hope it helps,

Luis

news:Pine.LNX.4.44.0204101511510.19210-100000@snoe.colorado.edu...

> Hi,

>

> I am interested in using IDLs mapping procedures on Mars, which has a
> fair number of datasets on the 0 to 360 *west* longitude system as
> opposed to the -180 to 180 *east* system.

>

> I posted this question a week or two ago, but got no responses. I
> don't think I stated the problem terribly clearly in the original
> post, so I am trying again. Also, IDL wrote me and said that:

>

>> Some national labs have changed these routines (which are *.pro
>> code) to work with the Moon, so check the newsgroup.

>

> After an extensive web-search, I found a "mapl_set" procedure
> mentioned here:

> <http://www.ias.fr/cdp/Futur/perso/biblio2.html>

> But there is no link and no code, only a description!

>

> -----

> Here is a demonstration of the problem:

> A simple map in IDL is created like this:

> IDL> map_set, 0, 0, /cylindrical

> IDL> map_grid, /label

>

> You will notice that the longitudes increase to the east, and are on a
> -180 to 180 degree grid. I can get the appearance of longitudes

> increasing to the west and on a 0 to 360 grid by these commands:

```

> IDL> map_set, 0, 0, /cylindrical
> IDL> map_grid, lons=[0,45,90,135,180,-135,-90,-45], $
> lonnames=[0,315,270,225,180,135,90,45], /label
>
> Everything appears fine. But the problem is that it is only the labels
> (that I hard-coded above) that *appear* to be westward, and 0 to 360.
> The entire map is still doing its math thing on a -180 to 180 east
> grid.
>
> This becomes apparent when you draw a box (in my case, the viewing
> area of the satellite instrument. If, for example the instrument is
> viewing the box from lat,lon coordinates [-30,45] (bottom left corner)
> to [0,0] to right corner:
>
> loadct, 39 & !p.color = 254 ; red box
> plots, 45, -30, /data
> plots, 45, 0, /data, /cont
> plots, 0, 0, /data, /cont
> plots, 0, -30, /data, /cont
> plots, 45, -30, /data, /cont
>
> If you do this after issuing the first set of mapping commands above,
> it appears as expected. But if you do it after the second map_set
> command, it appears in the wrong place (the same place as before).
>
> -----
> Finally, if the above make sense, I will give you an exact description
> of my problem. The above problem with drawing the box can be solved
> easily, because a very simple function will translate between the four
> possible grids (0-360 vs 180-180 vs 'east' vs 'west'). It would be
> nice to specify this via a keyword *once* when I call map_set, and not
> worry about it, but I can do the translations. But here is the real
> problem:
>
> If you hardwire the longitude labels, you have to hardwire their
> location (i.e. every degree or every 45 degrees, as above). My
> software is used to view the surface of mars, and allows the user to
> view it on a global scale, or down to a 1x1 degree grid.
>
> map_set, if allowed to 'do its own thing', will always generate nice
> grids for you. If you are looking at the globe, it will use 45 degree
> grids as shown above. If you are looking at a 1x1 degree grid, it will
> use .1 degrees.
>
> But if I hardcode in the longitude names so they are westward, I also
> have to code their locations. If I do every degree, you cannot see the
> globe, all you see is the grid covering it. If I do every 45 degrees,

```

> then you see no grids when you view a small area...
>
> Hence, my desire do use map_grid in a 'dumb' way (let it do the math
> to determine the grid spacing), and have it output westward 0-360
> grids.
>
> Finally, polar /ortho projections make this even more complicated than
> equator-based /cylindrical projections.
>
> Sorry for the length of this post...
>
> -k.
>
> --
> Kenneth Mankoff
> LASP://303.492.3264
> <http://lasp.colorado.edu/~mankoff/>
> <http://lasp.colorado.edu/snoe/>
> <http://lasp.colorado.edu/mars/>
> <http://lasp.colorado.edu/marsrobot/>
>
>
>
>

Subject: Re: map_grid limitations
Posted by [Ken Mankoff](#) on Thu, 11 Apr 2002 18:13:01 GMT
[View Forum Message](#) <> [Reply to Message](#)

I think your suggestion is to translate all my datasets to a -180 to
180 East grid. This is an option I am considering...

But all the globes in the office and around the lab are made on a 0 to
360 west grid, and it is nice to have the numbers on the screen agree
with those on the globe.

I do not want to try to change the coordinate systems of the planets
to meet IDLs requirements. Nor do I want my software to disagree with
most other peoples, the USGS, etc.

-k.

On Thu, 11 Apr 2002, Luis Alonso wrote:

> Hi!
> Why don't yo try, the same way you change the labels, you can change the
> header of you map in which the coordinates are stored. Just sustract 180 to

> the longitude.

> In case it is stored in a header file where it gives the first pixel

> coordinate (such as ENVI format) just subtract 180 in the corresponding line

> (x_start for the ENVI case)

> In case you have a matching array (lat_coord for example) storing the

> coordinates, just do:

> lat_coord = lat_coord -180

>

> Hope it helps,

> Luis

>

>

>

> "Ken Mankoff" <mankoff@I.HATE.SPAM.cs.colorado.edu> escribió ½ en el mensaje

>

>> Hi,

>>

>> I am interested in using IDLs mapping procedures on Mars, which has a

>> fair number of datasets on the 0 to 360 *west* longitude system as

>> opposed to the -180 to 180 *east* system.

>>

>> I posted this question a week or two ago, but got no responses. I

>> don't think I stated the problem terribly clearly in the original

>> post, so I am trying again. Also, IDL wrote me and said that:

>>

>>> Some national labs have changed these routines (which are *.pro

>>> code) to work with the Moon, so check the newsgroup.

>>

>> After an extensive web-search, I found a "mapl_set" procedure

>> mentioned here:

>> <http://www.ias.fr/cdp/Futur/perso/biblio2.html>

>> But there is no link and no code, only a description!

>>

>> -----

>> Here is a demonstration of the problem:

>> A simple map in IDL is created like this:

>> IDL> map_set, 0, 0, /cylindrical

>> IDL> map_grid, /label

>>

>> You will notice that the longitudes increase to the east, and are on a

>> -180 to 180 degree grid. I can get the appearance of longitudes

>> increasing to the west and on a 0 to 360 grid by these commands:

>> IDL> map_set, 0, 0, /cylindrical

>> IDL> map_grid, lons=[0,45,90,135,180,-135,-90,-45], \$

>> lonnames=[0,315,270,225,180,135,90,45], /label

>>

>> Everything appears fine. But the problem is that it is only the labels

>> (that I hard-coded above) that *appear* to be westward, and 0 to 360.

```

>> The entire map is still doing its math thing on a -180 to 180 east
>> grid.
>>
>> This becomes apparent when you draw a box (in my case, the viewing
>> area of the satellite instrument. If, for example the instrument is
>> viewing the box from lat,lon coordinates [-30,45] (bottom left corner)
>> to [0,0] to right corner:
>>
>> loadct, 39 & !p.color = 254 ; red box
>> plots, 45, -30, /data
>> plots, 45, 0, /data, /cont
>> plots, 0, 0, /data, /cont
>> plots, 0, -30, /data, /cont
>> plots, 45, -30, /data, /cont
>>
>> If you do this after issuing the first set of mapping commands above,
>> it appears as expected. But if you do it after the second map_set
>> command, it appears in the wrong place (the same place as before).
>>
>> -----
>> Finally, if the above make sense, I will give you an exact description
>> of my problem. The above problem with drawing the box can be solved
>> easily, because a very simple function will translate between the four
>> possible grids (0-360 vs 180-180 vs 'east' vs 'west'). It would be
>> nice to specify this via a keyword *once* when I call map_set, and not
>> worry about it, but I can do the translations. But here is the real
>> problem:
>>
>> If you hardwire the longitude labels, you have to hardwire their
>> location (i.e. every degree or every 45 degrees, as above). My
>> software is used to view the surface of mars, and allows the user to
>> view it on a global scale, or down to a 1x1 degree grid.
>>
>> map_set, if allowed to 'do its own thing', will always generate nice
>> grids for you. If you are looking at the globe, it will use 45 degree
>> grids as shown above. If you are looking at a 1x1 degree grid, it will
>> use .1 degrees.
>>
>> But if I hardcode in the longitude names so they are westward, I also
>> have to code their locations. If I do every degree, you cannot see the
>> globe, all you see is the grid covering it. If I do every 45 degrees,
>> then you see no grids when you view a small area...
>>
>> Hence, my desire do use map_grid in a 'dumb' way (let it do the math
>> to determine the grid spacing), and have it output westward 0-360
>> grids.
>>
>> Finally, polar /ortho projections make this even more complicated than

```

>> equator-based /cylindrical projections.
>>
>> Sorry for the length of this post...
>>
>> -k.
>>
>> --
>> Kenneth Mankoff
>> LASP://303.492.3264
>> <http://lasp.colorado.edu/~mankoff/>
>> <http://lasp.colorado.edu/snoe/>
>> <http://lasp.colorado.edu/mars/>
>> <http://lasp.colorado.edu/marsrobot/>
>>
>>
>>
>>
>
>
>

--
Kenneth Mankoff
LASP://303.492.3264
<http://lasp.colorado.edu/~mankoff/>
<http://lasp.colorado.edu/snoe/>
<http://lasp.colorado.edu/mars/>
<http://lasp.colorado.edu/marsrobot/>
