Subject: Re: data to image? Please help...
Posted by Ibnc on Tue, 18 Nov 2008 06:30:00 GMT
View Forum Message <> Reply to Message

```
On 18 Nov, 04:28, mbwel...@gmail.com wrote:
> Hello.
>
> I have an output file that gives me data in the form of:
> Top, bottom, left, right sides of a cell (so basically the corners)
> and two separate magnitude data columns.
>
> What I need to be able to do is create two image files that
> graphically, either via contours, colored cell scheme, or other...,
> display the values of magnitude per cell (either Magnitude value 1 or
> value 2).
>
> Is this feasible to do in IDL, if not is there another program better
 suited?
>
  If anyone has any ideas, I would really appreciate it.
> Thanks,
> ~Matt
Although it will not be the fastest solution and produce large
PostScript files, I would suggest looping over all values, plotting
them using POLYFILL, like so:
;- establish axes, without actually plotting them, because the
polyfill would cover them
plot, [0], /nodata, xstyle=5, ystyle=5, $
 xrange=some_xrange, yrange=some_yrange
for i=0L, ndata-1L do $
 polyfill, [xstart[i], xend[i], xend[i], xstart[i], xstart[i]], $
  [ystart[i], ystart[i], yend[i], yend[i], ystart[i]], $
  color=data_value[i], noclip=0
;- plot axes
plot, [0], /nodata, /noerase, xstyle=1, ystyle=1, xrange=some_xrange,
yrange=some yrange
Cheers
```

Subject: Re: data to image? Please help... Posted by mbweller on Wed, 19 Nov 2008 03:40:26 GMT

Lasse

```
On Nov 17, 10:30 pm, I...@lbnc.de wrote:
> On 18 Nov, 04:28, mbwel...@gmail.com wrote:
>
>
>> Hello,
>> I have an output file that gives me data in the form of:
>> Top, bottom, left, right sides of a cell (so basically the corners)
>> and two separate magnitude data columns.
>
>> What I need to be able to do is create two image files that
>> graphically, either via contours, colored cell scheme, or other...,
>> display the values of magnitude per cell (either Magnitude value 1 or
>> value 2).
>> Is this feasible to do in IDL, if not is there another program better
>> suited?
>> If anyone has any ideas, I would really appreciate it.
>> Thanks.
>> ~Matt
> Although it will not be the fastest solution and produce large
> PostScript files, I would suggest looping over all values, plotting
> them using POLYFILL, like so:
>
> ;- establish axes, without actually plotting them, because the
> polyfill would cover them
> plot, [0], /nodata, xstyle=5, ystyle=5, $
  xrange=some_xrange, yrange=some_yrange
> for i=0L, ndata-1L do $
   polyfill, [xstart[i], xend[i], xend[i], xstart[i], $
    [ystart[i], ystart[i], yend[i], yend[i], ystart[i]], $
>
     color=data_value[i], noclip=0
>
> :- plot axes
> plot, [0], /nodata, /noerase, xstyle=1, ystyle=1, xrange=some_xrange,
> yrange=some yrange
>
> Cheers
> Lasse
```

Thanks for the info, but in looking at your code, it appears that it

would assign case 1 as one color and case 2 as another color, correct?

If so, what I need to do is create a grid that has a color range. Let us say the values are between 1 and 5, so that value one is purple and value two is red and all values in between are a gradation between the extremes. Of course this is a simple case and I will probably have values a couple orders in magnitude in range and will need to be able to "auto bin" them. Will this code work for that, or do I need something else?

Thanks, ~Matt

Subject: Re: data to image? Please help... Posted by Jeremy Bailin on Wed, 19 Nov 2008 15:04:50 GMT View Forum Message <> Reply to Message

```
On Nov 18, 10:40 pm, mbwel...@gmail.com wrote:
> On Nov 17, 10:30 pm, I...@lbnc.de wrote:
>
>
>> On 18 Nov, 04:28, mbwel...@gmail.com wrote:
>>> Hello,
>
>>> I have an output file that gives me data in the form of:
>>> Top, bottom, left, right sides of a cell (so basically the corners)
>>> and two separate magnitude data columns.
>>> What I need to be able to do is create two image files that
>>> graphically, either via contours, colored cell scheme, or other...,
>>> display the values of magnitude per cell (either Magnitude value 1 or
>>> value 2).
>
>>> Is this feasible to do in IDL, if not is there another program better
>>> suited?
>>> If anyone has any ideas, I would really appreciate it.
>>> Thanks,
>>> ~Matt
>> Although it will not be the fastest solution and produce large
>> PostScript files, I would suggest looping over all values, plotting
>> them using POLYFILL, like so:
```

```
>
>> :- establish axes, without actually plotting them, because the
>> polyfill would cover them
>> plot, [0], /nodata, xstyle=5, ystyle=5, $
   xrange=some_xrange, yrange=some_yrange
>> for i=0L, ndata-1L do $
    polyfill, [xstart[i], xend[i], xend[i], xstart[i], xstart[i]], $
      [ystart[i], ystart[i], yend[i], yend[i], ystart[i]], $
>>
      color=data value[i], noclip=0
>>
>> :- plot axes
>> plot, [0], /nodata, /noerase, xstyle=1, ystyle=1, xrange=some_xrange,
>> yrange=some yrange
>
>> Cheers
>> Lasse
>
  Thanks for the info, but in looking at your code, it appears that it
 would assign case 1 as one color and case 2 as another color, correct?
>
> If so, what I need to do is create a grid that has a color range. Let
> us say the values are between 1 and 5, so that value one is purple and
> value two is red and all values in between are a gradation between the
> extremes. Of course this is a simple case and I will probably have
> values a couple orders in magnitude in range and will need to be able
> to "auto bin" them. Will this code work for that, or do I need
> something else?
>
> Thanks.
> ~Matt
```

It might be better to do it as an image... let's say that the data bounds are stored in arrays left, right, top, and bottom, and the data range in the x-direction is given in data_xrange and the same for data_yrange, and you want your final image to be nx by ny pixels. Then you could use this to create the image:

```
image = fltarr(nx,ny)
deltax = (xrange[1]-xrange[0])/float(nx)
deltay = (yrange[1]-yrange[0])/float(ny)
for i=0l,ndata-1 do $
    image[(left[i]-xrange[0])/deltax:(right[i]-xrange[0])/deltax , $
    (bottom[i]-yrange[0])/deltay:(top[i]-yrange[0])/deltay] = magnitude
[i]
```

Then you can use the normal image viewing routines (e.g. tvimage) to plot them:

erase

loadct, 10 ; or whatever you want - the Brewer tables would probably be useful location = [0.1,0.1,0.9,0.9]tvimage, bytscl(image,top=250)+4, position=location plot, /noerase, /nodata, [0],[0], position=location, xrange=xrange, yrange=yrange

-Jeremy.