Subject: How to obtain contour data through contour command?
Posted by jacobsvensmark on Tue, 22 Oct 2013 12:12:05 GMT

So I have regularly gridded f(x,y) values, and I wish to know for which x,y I have f(x,y)=k. I imagine this is exactly what the contour command does, though I don't believe it can only plot the curve, not output the curve in an array.

I apologize if this is super trivial.

Subject: Re: How to obtain contour data through contour command?
Posted by on Tue, 22 Oct 2013 13:14:18 GMT

On 2013-10-22 14:12, jacobsvensmark@gmail.com wrote:
> So I have regularly gridded f(x,y) values, and I wish to know for which x,y I have f(x,y)=k. I imagine this is exactly what the contour command does, though I don't believe it can only plot the curve, not output the curve in an array.

The array indices are easy to find:

```
IDL> f = [[1, 2, 3], [2, 3, 2]]
IDL> print, f
      1     2     3
      2     3     2
IDL> print, array_indices(f, where(f eq 2))
        1         0
        0         1
        2         1
```
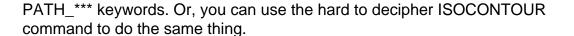
If the x and y coordinates are different from the grid indices you will
need to use the indices to get them. How that is done depends on how
those coordinates are specified.

Subject: Re: How to obtain contour data through contour command?
Posted by David Fanning on Tue, 22 Oct 2013 13:32:23 GMT

jacobsvensmark@gmail.com writes:

> So I have regularly gridded f(x,y) values, and I wish to know for which x,y I have f(x,y)=k. I imagine this is exactly what the contour command does, though I don't believe it can only plot the curve, not output the curve in an array.

You can get the output contour values from the Contour command using the

PATH_*** keywords. Or, you can use the hard to decipher ISOCONTOUR
command to do the same thing.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: How to obtain contour data through contour command?
Posted by kagoldberg on Tue, 22 Oct 2013 21:50:42 GMT

View Forum Message <> Reply to Message

An example of looks like this

contour, image, levels=[30.,40.], path_info=pinfo, path_xy=pxy, /path_data_coords, /closed

Here, pinfo is a structure that tells you about the information contained in pxy.
You have to write some flexible code to handle this properly because pxy is a single, 2xN array
that actually contains all of the contours you requested, strung together vertically. pxy[0,*] are the
x values, and pxy[1,*] are the y values.

To make it usable, you have to sort through pinfo. pinfo is an array of structures with fields {type,
high_low, level, n, offset, value}. The pinfo.offset field elements tell you the index values of pxy
where each contour starts, and pinfo.n elements tell you how many points are in each contour.

Here is one array for each contour.
xy0 = pxy[0:1, pinfo[0].offset : pinfo[0].offset + pinfo[0].N - 1]
xy1 = pxy[0:1, pinfo[1].offset : pinfo[1].offset + pinfo[1].N - 1]

Where you have to be careful, I've found, is that sometimes you get two (or more?) contours for
the same level value. In my example I have 2 levels, but you could end up with 3 or more contour
"pieces." So check the pinfo.value field to see if you have separate contours with matching level
values, and group/concatenate them if necessary.

-Ken