Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Brian Larsen on Tue, 23 Sep 2008 16:54:58 GMT
View Forum Message <> Reply to Message

I think what you are looking for is krig2d. Look it up in help. The
only annoyance is that it is either looking for a regular grid (which
I think you have) or x,y,z triplets. I have never figured out how to
make x,y,z triplets other than using nested for loops to step through
the points.

From here read the krig2d help and work through the example there and
see if that solves your issue. If not let me know whats not working
and I'll see if I can help more.

One word of caution is that interpolation is great "inside" the range
where you have data, however "outside" the region is extrapolation and
is fraught with issues. I mean that your x's
IDL> print,vz
-1.62839 -1.23045 -0.628389 -0.327359 0.0483046
0.246672
and the new x's that you want
newx = [-2.0, -1.5, -1.0, -0.5, 0.0, 0.5]
some are outside and you need to be a little careful that the answer
actually makes sense as if often (maybe stronger than often) doesn't.

Cheers,

Brian

---------- ----------
Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Vince Hradil on Tue, 23 Sep 2008 17:14:53 GMT
View Forum Message <> Reply to Message

On Sep 23, 11:54 am, Brian Larsen <balar...@gmail.com> wrote:
> I think what you are looking for is krig2d. Look it up in help. The
> only annoyance is that it is either looking for a regular grid (which
> I think you have) or x,y,z triplets. I have never figured out how to

> make x,y,z triplets other than using nested for loops to step through
> the points.
>
> From here read the krig2d help and work through the example there and
> see if that solves your issue. If not let me know whats not working
> and I'll see if I can help more.
>
> One word of caution is that interpolation is great "inside" the range
> where you have data, however "outside" the region is extrapolation and
> is fraught with issues. I mean that your x's
> IDL> print,vz
> -1.62839 -1.23045 -0.628389 -0.327359 0.0483046
> 0.246672
> and the new x's that you want
> newx = [-2.0, -1.5, -1.0, -0.5, 0.0, 0.5]
> some are outside and you need to be a little careful that the answer
> actually makes sense as if often (maybe stronger than often) doesn't.
>
> Cheers,
>
> Brian
>
> ----------
> Brian Larsen
> Boston University
> Center for Space Physicshttp://people.bu.edu/balarsen/Home/IDL

Here's a way to get verts:

```
sz = size(array)
nx = sz[0]
ny = sz[1]
nz = sz[2]
ns = sz[sz[0]+2]
verts = findgen(ns)
verts = transpose([ [verts mod nx], [verts/nx mod ny], [verts/nx/
ny] ])
```

BTW, I'd like to find a faster way, if there is one.

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Brian Larsen on Tue, 23 Sep 2008 17:15:07 GMT
View Forum Message <> Reply to Message

Another thing I see that might help is the "An Alternative Gridding
Method" section from http://www.dfanning.com/code_tips/griddata.html.

Brian

Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Brian Larsen on Tue, 23 Sep 2008 17:25:46 GMT
View Forum Message <> Reply to Message

> Here's a way to get verts:
>
> sz = size(array)
> nx = sz[0]
> ny = sz[1]
> nz = sz[2]
> ns = sz[sz[0]+2]
> verts = findgen(ns)
> verts = transpose([[verts mod nx], [verts/nx mod ny], [verts/nx/
> ny]])
>
> BTW, I'd like to find a faster way, if there is one.

This looks like the right thing but doesn't seem to give the right
answer (or am I using it wrong?)

```
;; this is my data
array=findgen(15,3)
;; and get the verts
sz = size(array)
nx = sz[0]
ny = sz[1]
nz = sz[2]
ns = sz[sz[0]+2]
verts = findgen(ns)
verts = transpose([ [verts mod nx], [verts/nx mod ny], [verts/nx/
ny] ])

IDL> print, verts
     0.00000      0.00000      0.00000
     1.00000     0.500000    0.0333333
     0.00000      1.00000    0.0666667
     1.00000      1.50000     0.100000
     0.00000      2.00000     0.133333
```

1.00000 2.50000 0.166667

The z values that are here are not in my original array...

Sorry to hijack your post Paula,

Brian

---------- ----------
Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Jean H. on Tue, 23 Sep 2008 17:53:27 GMT
View Forum Message <> Reply to Message

> Here's a way to get verts:
>
> sz = size(array)
> nx = sz[0]
> ny = sz[1]
> nz = sz[2]
> ns = sz[sz[0]+2]
> verts = findgen(ns)

Shouldn't it be indgen() ...
> verts = transpose([[verts mod nx], [verts/nx mod ny], [verts/nx/
> ny]])
>
> BTW, I'd like to find a faster way, if there is one.

what about this:
print, array_indices(arr, indgen(ns))

Jean

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Vince Hradil on Tue, 23 Sep 2008 19:46:30 GMT
View Forum Message <> Reply to Message

On Sep 23, 12:25 pm, Brian Larsen <balar...@gmail.com> wrote:
>> Here's a way to get verts:
>

>> sz = size(array)
>> nx = sz[0]
>> ny = sz[1]
>> nz = sz[2]
>> ns = sz[sz[0]+2]
>> verts = findgen(ns)
>> verts = transpose([[verts mod nx], [verts/nx mod ny], [verts/nx/
>> ny]])
>
>> BTW, I'd like to find a faster way, if there is one.
>
> This looks like the right thing but doesn't seem to give the right
> answer (or am I using it wrong?)
>
> ;; this is my data
> array=findgen(15,3)
> ;; and get the verts
> sz = size(array)
> nx = sz[0]
> ny = sz[1]
> nz = sz[2]
> ns = sz[sz[0]+2]
> verts = findgen(ns)
> verts = transpose([[verts mod nx], [verts/nx mod ny], [verts/nx/
> ny]])
>
> IDL> print, verts
>   0.00000   0.00000   0.00000
>   1.00000   0.500000   0.0333333
>   0.00000   1.00000   0.0666667
>   1.00000   1.50000   0.100000
>   0.00000   2.00000   0.133333
>   1.00000   2.50000   0.166667
>
> The z values that are here are not in my original array...
>
> Sorry to hijack your post Paula,
>
> Brian
>
> -- -------------
> Brian Larsen
> Boston University
> Center for Space Physicshttp://people.bu.edu/balarsen/Home/IDL

My example is for 3d arrays only! For 2d arrays, just use:
nx = sz[1]
ny = sz[2]

verts = lindgen(nx*ny)
verts = transpose([[verts mod nx], [verts/nx]])

There was a typo in the original, too. nx=sz[1] not sz[0], and so on.

Yes, it should be lindgen...

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by Brian Larsen on Tue, 23 Sep 2008 20:26:42 GMT
View Forum Message <> Reply to Message

> My example is for 3d arrays only!

Right duh, engage brain before pressing send :)

Brian

---------- ----------
Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: what is the best way to do a surface (or 2D) interpolation?
Posted by paulartcoelho on Wed, 24 Sep 2008 11:01:02 GMT
View Forum Message <> Reply to Message

hi brian,

thanks for your reply.

krig2d i couldn't make work, it returns a flat new grid with a weird
value. but i'll confess that i haven't understood a word of the "Model
Parameter Keywords" so i just took the same parameters of the example,
duh.

the method in
> "An Alternative Gridding
> Method" section from http://www.dfanning.com/code_tips/griddata.html.

is giving me promising results. i can obtain a new_grid that is
similar to the original, but i'm struggling a bit with the details
now. values that were low in the original grid, say <~ 10 will
disappear in the new one, i guess because i have too many zeros

around. if i try the cubic=-0.5 suggested in the help to improve the reconstruction, i'll end up with values below 0 in some bins (no negative values in the original). also i noticed that:

```
IDL> print,total(grid)
      100.000

IDL> print,total(new_grid)
      61.3827
```

i can re-normalize the new_grid to 100, but that won't retrieve the lost low values, of course.

> One word of caution is that interpolation is great "inside" the range
> where you have data, however "outside" the region is extrapolation and
> is fraught with issues. I mean that your x's
> IDL> print,vz
> -1.62839 -1.23045 -0.628389 -0.327359 0.0483046
> 0.246672
> and the new x's that you want
> newx = [-2.0, -1.5, -1.0, -0.5, 0.0, 0.5]
> some are outside and you need to be a little careful that the answer
> actually makes sense as if often (maybe stronger than often) doesn't.

you're right. i have several sets of data and some of them can go out to extreme values. but for the data that don't, i'd just need the values there to be assumed to be zero. i thought i could do that just adding the missing = 0 keyword in interpolate, but if i do that i end up with a flat new_grid = 0. i wonder now if i should prepare the original data before applying the interpolation, say extend it with zeros myself?

i'm copying below what i'm doing:

```
IN_X          FLOAT     = Array[16]
IN_Y          FLOAT     = Array[6]
GRID          FLOAT     = Array[16, 6]

IDL> print,in_x
      8.00000      8.14613      8.30103      8.44716      8.60206
8.74819      8.89763      9.04922      9.19866      9.35025
9.49969      9.65031
      9.80003      9.94988      10.1000      10.2000

IDL> print,in_y
     -1.62839     -1.23045     -0.628389     -0.327359     0.0483046
0.246672
```

```
IDL> print,grid
      0.00000      26.1465      0.00000      0.00000      0.00000
0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000     0.820449
      0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000
      0.00000      2.53796      0.00000      0.00000      0.00000
0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000      61.5708
      0.00000      0.00000      0.00000      0.00000      0.00000
0.00000      0.00000      8.92427      0.00000      0.00000
0.00000      0.00000
      0.00000      0.00000      0.00000      0.00000
```

```
out_x = [6.0,6.5,7.0,7.5,8.0,8.5,9.0,9.5,10.0,10.5]
out_y = [-2.0,-1.5,-1.0,-0.5,0.,0.5]
nx = n_elements(out_x)
ny = n_elements(out_y)
x = Interpol(Findgen(N_Elements(in_x)), in_x, out_x)
y = Interpol(Findgen(N_Elements(in_y)), in_y, out_y)
xx = Rebin(x, nx, ny, /SAMPLE)
yy = Rebin(Reform(y, 1, ny), nx, ny, /SAMPLE)
new_grid = interpolate(grid,xx,yy)
```

> Sorry to hijack your post Paula,

oh, be my guest :)

cheers,
paula