
Subject: Re: Interpolation of missing data
Posted by [mirko_vukovic](#) on Tue, 20 Jan 1998 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <6a2t1p\$mfd@maze.dpo.uab.edu>,
rkj@dukebar.crml.uab.edu (R. Kyle Justice) wrote:
>
> I have a 2-D array with missing data. Is there an easy way to
> interpolate the missing values?
>
> I would like to replace a missing value with the average of
> its neighbors.
>
> Kyle J.

I found the `median_filter` (under image processing applications)
usefull in similar situations.

good luck,

mirko

-----==== Posted via Deja News =====
<http://www.dejanews.com/> Search, Read, Post to Usenet

Subject: Re: Interpolation of missing data
Posted by [meron](#) on Tue, 20 Jan 1998 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <6a2t1p\$mfd@maze.dpo.uab.edu>, rkj@dukebar.crml.uab.edu (R. Kyle Justice) writes:
> I have a 2-D array with missing data. Is there an easy way to
> interpolate the missing values?
>
> I would like to replace a missing value with the average of
> its neighbors.
>
I have a routine which'll do it for an array with an arbitrary number
of dimensions. E-mail me if you need assistance

Mati Meron | "When you argue with a fool,
meron@cars.uchicago.edu | chances are he is doing just the same"

Subject: Re: Interpolation of missing data
Posted by [Martin Schultz](#) on Tue, 20 Jan 1998 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

R. Kyle Justice wrote:

```
>  
> I have a 2-D array with missing data. Is there an easy way to  
> interpolate the missing values?  
>  
> I would like to replace a missing value with the average of  
> its neighbors.  
>  
> Kyle J.
```

This may not be exactly what you want, but you could try to "re-sample" your data as an array and use the TRI_SURF (or MIN_CURVE_SURF) function. I have a piece of code that does something like

```
; create x and y vectors that match with zz array  
goodx = findgen(nx+3)/(nx+2)*(xrange(1)-xrange(0))+xrange(0)  
goody = findgen(ny+3)/(ny+2)*(yrange(1)-yrange(0))+yrange(0)  
  
; little trick to get the indices of valid zz's in goodx and goody  
xind = (ind mod (nx+2))  
yind = (ind/(nx+2)) ; integer division !!  
  
newx = reform(goodx(xind),n_elements(ind))  
newy = reform(goody(yind),n_elements(ind))  
  
zzz = TRI_SURF(newz,newx,newy,gs=[dx,dy], $  
              bounds=[xrange(0),yrange(0),xrange(1),yrange(1)] )
```

(for regular readers: this turned out to be the best solution to my contour problem that I described earlier - but I must warn of the use of MIN_CURVE_SURF: it takes *forever* [i.e. I did not want to wait more than 3 minutes for a data set of ~1000 points and interrupted])

Regards,
Martin

PS: another solution (which would involve a loop [nasty word ;-]) would be to compute the averages of surrounding grid boxes like

```
ind = where(data eq MISSING) ; supply your code for missing data  
if (ind(0) ge 0) then begin  
  for i=0,n_elements(ind)-1 do begin  
    x = (i mod (NX+2)) ; get indices in data array  
    y = (i/(NX+2)) ; integer division !!
```

```

; create index array for neighbouring points
xind = [ x-1>(-1), x, x+1<NX, x ]
yind = [ y, y-1>(-1), y, y+1<NY ]
; find out valid neighbours
ok = where(xind ne MISSING and yind ne MISSING)
if (ok(0) ge 0) then $
    data(x,y) = total(data(xind(ok),yind(ok)))/ $
        float(n_elements(ok))
endifor
endif

```

This would of course only work if at least one neighbour is a valid data point. In case you are not familiar with the < and > operators: they are great to limit value ranges, I just recently understood them and loved them immediately! Please NOTE: I did not test this code, but it should give you something to start with at least.

Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
186 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
IDL-homepage: <http://www-as.harvard.edu/people/staff/mgs/idl/>

Subject: Re: Interpolation of missing data
Posted by [rkj](#) on Wed, 21 Jan 1998 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

mirko_vukovic@notes.mrc.sony.com wrote:
: In article <6a2t1p\$mfd@maze.dpo.uab.edu>,
: rkj@dukebar.crml.uab.edu (R. Kyle Justice) wrote:
: >
: > I have a 2-D array with missing data. Is there an easy way to
: > interpolate the missing values?
: >
: > I would like to replace a missing value with the average of
: > its neighbors.
: >
: > Kyle J.

: I found the median_filter (under image processing applications)
: usefull in similar situations.

: good luck,

: mirko

The only problem here is that the median filter ignores boundary values. That is, if the missing value to be replaced is on the edge of the "image" I am out of luck. Is it supposed to do this???

Kyle J.

Subject: Re: Interpolation of missing data
Posted by [thompson](#) on Thu, 22 Jan 1998 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

rkj@dukebar.crml.uab.edu (R. Kyle Justice) writes:

> mirko_vukovic@notes.mrc.sony.com wrote:
> : In article <6a2t1p\$mfd@maze.dpo.uab.edu>,
> : rkj@dukebar.crml.uab.edu (R. Kyle Justice) wrote:
> : >
> : > I have a 2-D array with missing data. Is there an easy way to
> : > interpolate the missing values?
> : >
> : > I would like to replace a missing value with the average of
> : > its neighbors.
> : >
> : > Kyle J.

> : I found the median_filter (under image processing applications)
> : usefull in similar situations.

> : good luck,

> : mirko

> The only problem here is that the median filter ignores boundary
> values. That is, if the missing value to be replaced is on the
> edge of the "image" I am out of luck. Is it supposed to do this???

Yes, unfortunately.

At one point I wrote a routine called fmedian which did the same thing as the

built-in median function, but with smoothly decreasing filter width at the edges. It also allows for different widths in the two spatial directions. You can find the routine at

<http://sohowww.nascom.nasa.gov/solarsoft/gen/idl/util/>

<ftp://sohoftp.nascom.nasa.gov/solarsoft/gen/idl/util/>

You'll need both `fmedian.pro` and `fmedian_slow.pro`. There's also some `CALL_EXTERNAL` software to speed up the routine at `/gen/idl_external`. It's written in Fortran--sorry. The routine will work, however, without the `CALL_EXTERNAL` support--it'll just be slower.

A different way to do almost the same thing with the built-in median procedure would be to embed your image in a bigger image with appropriate data at the edges. Take the median of the bigger image, and then throw away the edges. The trick would be to figure out what data to put at the edges of the bigger image before taking the median filter.

Bill
