
Subject: spatial interpolation

Posted by [Isa Usman](#) on Sat, 27 Sep 2003 10:08:55 GMT

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

Hello All,

I have a program which interpolates an irregularly gridded set of data points onto another irregular grid. I have tried as much as possible to make the calculations as fast as possible (using the dreaded reverse indices in Histogram) but i am at my wits end. It currently takes about two days to go through the whole data. Anybody got any suggestions on speed-up improvements? The code is shown below.

Cheers

Isa

PRO ppi2star

```
*****STAGE 1- INITIALISE VARIABLES AND CONSTRUCT SEGMENTS*****
```

```
*****
```

```
start = SYSTIME(1)
```

```
;set the azimuth and range spacing for the original data
```

```
ngate = 68L
```

```
range = 20.0 + (FINDGEN(ngate) + 1) * 0.3
```

```
naz = 208L
```

```
az = (FINDGEN(naz) + 1) * 0.24038 ;az ranged 0-50 degrees
```

```
x = FLTARR(ngate*naz)
```

```
y = x
```

```
;set the azimuth and range spacing for the interpolated data
```

```
nrange = 91L
```

```
naz2 = 361L
```

```
az2 = FINDGEN(naz2)
```

```
range2 = FINDGEN(nrange) * 0.1
```

```
x1 = FLTARR(nrange*naz2)
```

```
y1 = x1
```

```
f1 = y1
```

```
;generate the cartesian x and y points for the data
```

```
FOR i = 0, naz - 1 DO BEGIN
```

```
polrec,Range, REPLICATE(az[i],ngate), xtemp, ytemp, /DEGREES
```

```
x[(i * ngate):(i * ngate) + (ngate - 1)] = xtemp
```

```
y[(i * ngate):(i * ngate) + (ngate - 1)] = ytemp
```

```
ENDFOR
```

```

;scale points onto a unit square
x0 = MIN(x, MAX = xmax)
y0 = MIN(y, MAX = ymax)
scale = ((xmax - x0) > (ymax - y0)) ;Scale factor for unit square
xs = (x - x0) / scale ;Scale into 0-1 rectangle to enhance accuracy
ys = (y - y0) / scale

;PRINT,x0,y0,xmax,ymax,scale

max_range = MAX(range, MIN = min_range)
max_azimuth = MAX(az, MIN = min_azimuth)

;get x,y centre of data
polrec, (max_range + min_range) / 2.0, ((max_azimuth + min_azimuth) / 2.0) *
!dtor, xo, yo
;PRINT,xo,yo

;generate the cartesian x and y points for the interpolated data centred
over the
;original data and also scale onto a unit square
FOR i = 0, naz2 - 1 DO BEGIN

    polrec,Range2,REPLICATE(az2[i],nrange),xtemp,ytemp, /DEGREES
    x1[(i*nrange):(i*nrange)+(nrange-1)]=(xtemp+xo)-x0/scale
    y1[(i*nrange):(i*nrange)+(nrange-1)]=(ytemp+yo)-y0/scale

ENDFOR

;Plot to see that point fall within the ppi grid
;DEVICE, FILENAME = assign_filename('ps')
;PLOT, xs, ys, PSYM = 1
;OPLOT, x1, y1, PSYM = 2, COLOR = 150
;DEVICE, /CLOSE

;d = 1

;*****
;
;construct a mesh over the data points and interpolating points by taking
;a histograms of both over the same bin sizes and ranges
;we shall be using the reverse indices to get the x, y values that correspond
to each bin

;construct array of neighbouring bins that are required including original
index

surround = 24 ;options are 8,24

```

```
IF surround EQ 8 THEN BEGIN
```

```
  nhist = 80
  index_segments = [0, -1, 1, nhist[0], nhist[0]+1,
nhist[0]-1, -nhist[0], -nhist[0]+1, -nhist[0]-1]
```

```
ENDIF
```

```
IF surround EQ 24 THEN BEGIN
```

```
  nhist = 80
  nhist2 = nhist[0]*2
  array1 = [0, -1, 1, nhist[0], nhist[0]+1,
nhist[0]-1, -nhist[0], -nhist[0]+1, -nhist[0]-1]
  array2 = [nhist2-2, nhist2-1, nhist2, nhist2+1, nhist2+2, nhist-2,
nhist+2, -2, 2, -nhist-2, -nhist+2, -nhist2-2,$
  -nhist2-1, -nhist2, -nhist2+1, -nhist+2]
  index_segments = [array1,array2]
```

```
ENDIF
```

```
v = TRANSPOSE([[xs],[ys]])
```

```
MIN_DATA = [MIN(xs),MIN(ys)]
MAX_DATA = [MAX(xs),MAX(ys)]
```

```
hist = HIST_ND(V,MIN = 0.0,MAX = 1.0,NBINS = nhist, REVERSE_INDICES = ri0)
```

```
v = 0.0
v = TRANSPOSE([[x1],[y1]])
```

```
hist1 = HIST_ND(V,MIN = 0.0,MAX=1.0, NBINS = nhist, REVERSE_INDICES = ri1)
v = 0.0
```

```
;find where the bins are non-empty to give us the locations we want to
interpolate
index = WHERE(hist1 GT 0,count)
```

```
nseg = N_ELEMENTS(index_segments) ;surrounding number of segments
```

```
;***** STAGE 2- READ DATA AND INTERPOLATE *****
;*****
;
rainrate = FLTARR(naz,ngate)
```

```
main_path = !dpaths.radar ;main path where files are located
```

```
;open files
OPENR, lun, main_path+'processed\ppi\R.dat',/get_lun ;binary file containing
```

```

ppo data
OPENW, lun1, main_path+'processed\cppi\rbf\Rstar.dat',/get_lun ;binary file
to contain star data

n=0 ;number of scans counter

WHILE (NOT EOF(lun)) DO BEGIN

    READU, lun, rainrate
    ;print,max(rainrate)
    fs = REFORM(TRANSPPOSE(rainrate),ngate*naz)

    FOR i = 0, count - 1 DO BEGIN

        index2 = index[i] ;index of bin

        index_segments1 = index2+index_segments ;index of bin + surrounding bins

        ;get x,y values that correspond to the central bin to interpolate
        seg_0_int_x = x1[ri1[ri1[index_segments1[0]]:ri1[index_segments1[0]+1]-1]]
        seg_0_int_y = y1[ri1[ri1[index_segments1[0]]:ri1[index_segments1[0]+1]-1]]

        ;get data x,y values that correspond to the central bin + surrounding bins
        ;central bins for the data
        seg_dat_x = xs[ri0[ri0[index_segments1[0]]:ri0[index_segments1[0]+1]-1]]
        seg_dat_y = ys[ri0[ri0[index_segments1[0]]:ri0[index_segments1[0]+1]-1]]
        seg_dat_f = fs[ri0[ri0[index_segments1[0]]:ri0[index_segments1[0]+1]-1]]

        ;array index where the results are to be put
        index3 = ri1[ri1[index_segments1[0]]:ri1[index_segments1[0]+1]-1]

        ;before doing the interpolation check to see if the surface
        ;of the central segment/bin is flat
        max_f = MAX(seg_dat_f, MIN = min_f)
        IF (max_f EQ min_f) THEN BEGIN ;surface is flat so no need to interpolate

            seg_0_int_f = REPLICATE(max_f, N_ELEMENTS(seg_0_int_x))
            f1[index3] = seg_0_int_f
            ;print,'tic'
            CONTINUE

        ENDIF ELSE BEGIN

            FOR j = 1, nseg - 1 DO BEGIN ;surrounding bins for the data

                ;first check to see if the current segment is empty
                IF (ri0[index_segments1[j]] EQ ri0[index_segments1[j]+1]) THEN CONTINUE
            
```

```

temp = xs[ri0[ri0[index_segments1[j]]:ri0[index_segments1[j]+1]-1]]
seg_dat_x = [TEMPORARY(seg_dat_x),temp] ;concatenate

temp=ys[ri0[ri0[index_segments1[j]]:ri0[index_segments1[j]+1 ]-1]]
seg_dat_y = [TEMPORARY(seg_dat_y),temp] ;concatenate

temp = fs[ri0[ri0[index_segments1[j]]:ri0[index_segments1[j]+1]-1]]
seg_dat_f = [TEMPORARY(seg_dat_f),temp] ;concatenate

ENDFOR

;print,n_elements(seg_dat_x)
;now do the interpolation
seg_0_int_f = MIN_CURVE_SURF(seg_dat_f, seg_dat_x, seg_dat_y ,
XPOUT=seg_0_int_x, YPOUT = seg_0_int_y, /DOUBLE, /TPS)
;clean-up small interpolation errors
index_temp = WHERE(seg_0_int_f lt 0.01,count1)
IF count1 GT 0 THEN seg_0_int_f[index_temp] = 0.0
f1[index3] = FLOAT(seg_0_int_f)
;print,'toc'

ENDELSE

seg_0_int_x = 0 & seg_0_int_y = 0 & seg_0_int_f = 0 & seg_dat_x = 0 &
seg_dat_y = 0 & seg_dat_f = 0

ENDFOR

PRINT, n
;reformat back array to the correct dimensions
new_rainrate = TRANSPOSE(REFORM(f1,nrange,naz2))

WRITEU,lun1,new_rainrate
;reset array
f1 = TEMPORARY(f1)*0.0
n = n + 1 ;set counter

ENDWHILE

CLOSE,lun
CLOSE,lun1

FREE_LUN,lun
FREE_LUN,lun1

stop = SYSTIME(1)
PRINT,'time for stage is', stop-start

```

END
