
Subject: Re: Phase Unwrapping algorithm in IDL
Posted by [Robert Weiss](#) on Fri, 23 Jun 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Richard Tyc wrote:

> Would anyone have a 2D phase image unwrapping algorithm they would be
> willing to share ? (an algorithm which corrects for phase spillover on
> images which may only contain data between 0 and 2PI but in fact the data
> has a larger range which gets wrapped into the 0 -> 2PI range)
>
> If not in IDL, one in C or Fortran would also help.
>
> Thanks in Advance
>
> Richard Tyc

Hello Richard,

all I've got is a routine I basically use after
(real,imaginary)->(argument,phase) transformations. However, I've attached
it; but if you want to reduce noisy data (e.g. SAR images) with it, I think
it won't be of much use.

Hope this helps.

Robert.

P.S.: Put these routines into two different files 'unwrap_phase.pro' and
'unwrap_phase_2d.pro'. Sometimes, if the jumps are very close to PI, the
routine is not reliable, so watch out.

```
;  
;+  
; NAME:  
;   UNWRAP_PHASE  
;  
;  
; PURPOSE:  
;   unwrap phase jumps in one and two-dimensional arrays  
;  
;  
; EXPLANATION:  
;   when transforming from a (real,imaginary)-representation of  
;   complex numbers to a (argument, phase)-representation,  
;   undesired phase jumps may occur. This routines searches for  
;   these jumps and corrects them by adding/subtrating multiples of  
;   2*PI. This routine may be called with one and two-dimensional  
;   arrays. If the data is 2D, the routine UNWRAP_PHASE_2D will be  
;   automatically called.  
;  
;
```

```

; CALLING SEQUENCE:
;   result=unwrap_phase(data)
;
; INPUTS:
;   data:  a 1D or 2D real array (NO COMPLEX ARRAYS!!)
;
; OUTPUTS:
;   an hopefully phase unwrapped array
;
; PROCEDURES USED:
;   UNWRAP_PHASE_2D
;
; REVISION HISTORY:
;   written by A. R. Weiss (ARW), MPI for Astronomy, Heidelberg,
;   Germany, Sep. 1999
;-
FUNCTION unwrap_phase, data

decision = size(data)
IF decision[0] GT 2 THEN BEGIN
    print, 'Array dimensions higher than 2 not supported.'
    return, -1
ENDIF
IF decision[0] EQ 2 THEN BEGIN
    result = unwrap_phase_2d(data)
    return, result
ENDIF ELSE BEGIN
    ;; find the phase jump locations
    jumps = data - shift(data,1)
    jumps[0]=0.0
    jumps = long(jumps GE !DPI) - long(jumps LE -!DPI)
    jumpindex = where(jumps NE 0)
    result = data
    IF jumpindex[0] NE -1 THEN BEGIN
        FOR count=0,n_elements(jumpindex)-1 DO BEGIN
            result[jumpindex[count]:*] = $
                result[jumpindex[count]:*] $
                -2*!DPI*jumps[jumpindex[count]]
        ENDFOR
    ENDIF
    return, result
ENDELSE

END

```

```
FUNCTION unwrap_phase_2d, data
;; first correct a reference column
result = data
refcol = transpose(result[0,*])
result[0,*]=unwrap_phase(refcol)
;; now correct all rows
FOR count = 0,(size(result))[2]-1 DO BEGIN
  refrow = result[* ,count]
  result[* ,count]=unwrap_phase(refrow)
ENDFOR
;; then correct all columns again to be sure
FOR count = 0,(size(result))[1]-1 DO BEGIN
  refrow = transpose(result[count,*])
  result[count,*]=unwrap_phase(refrow)
ENDFOR
return, result

END
```

File Attachments

- 1) [unwrap_phase.pro](#), downloaded 105 times
 - 2) [unwrap_phase_2d.pro](#), downloaded 101 times
-