Subject: Re: bug in IDL's hanning() window-generating function
Posted by Brian Jackel on Wed, 01 Aug 2001 17:51:14 GMT
View Forum Message <> Reply to Message

Hi Scott

A friend and I noticed this several years ago. My recollection is that he reported this to RSI, but they weren't convinced of the problem. Of course, it *is* a (minor) problem.

At the time I was playing around with windows (or apodizing functions, or tapers) and wrote some IDL code. It's included below. The "Slepian" window is commented out because it requires something clever with eigenvalues, and the details have escaped me.

To see what Scott is talking about, try typing the following two commands:

```
  test= FFT_TAPER(512,'hanning',/PLOT)   ;correct way
  test= FFT_TAPER(512,'badhanning',/PLOT)  ;IDL way
```

Although this is old code, I would appreciate any comments or bug reports. Have fun.

Brian Jackel

```
 ;============================================================ ======
;distribute freely,
;use at own risk,
;send all bug reports to bjackel@phys.ucalgary.ca
;+
; NAME: FFT_TAPER
;
; PURPOSE: This function provides a selection of "tapers" (also known
;         as "windows" or "apodizing functions") to reduce sidelobe
;         leakage when calculating FFT's.
;
; CATEGORY: Time Series Analysis, Signal Processing
;
; CALLING SEQUENCE:    Result=
FFT_TAPER(Nelements,Windowname,[Parameter])
;
; INPUTS:
;    Nelements    the number of elements in the window ie. # of
;                 elements in the time series to be windowed.
;                 Must be a positive integer  (floats will be rounded).
```

```
;
;    Windowname    a string containing the name of the windowing function
;                  to use.  Not case sensitive, whitespace will be
ignored.
;
; OPTIONAL INPUTS:
;     Parameter    some tapers can be tuned with a parameter value.  See
;                  specific descriptions for details.
;
; KEYWORD PARAMETERS:
;
;   PLOT_EXAMPLE        if set, the window and amplitude spectrum will
;                       be plotted
;   RETURN_WINDOWLIST   if set, Result will be a string array of all
;                       valid Windownames
;
;
;
;
;
;
;
;  Kaiser-Bessel    This taper can be tuned.  Set the parameter to
;  (Nuttall p89)    the desired location (in scaled Lf/pi units) of
;                   the first null.  ie. 2, 3, or 4
;
;  van der Maas     This taper can be tuned.  Set the parameter to
;  (Nuttall p90)    the ratio of the log of the peak to sidelobe level
;                   ie. 2 (factor of 100 or 20dB) or 3 (30dB)
;
;

; MODIFICATION HISTORY:
;       Written by:     Brian Jackel 1995
;-

Function FFT_TAPER,Length,Windowtype,Parameter, $
         RETURN_WINDOWLIST=return_windowlist,PLOT_EXAMPLE=plot_exampl e

;
;Make a list of all the legal window names, and put them in an array.
If the
;keyword RETURN_WINDOWLIST is set, then just return the array of
strings.  This
;is useful for letting the user know what options are available.
;
  windowlist=['Hanning', 'Hamming', 'Blackman', 'Exact Blackman',
'Blackman-Harris']
  windowlist= [windowlist, 'Dolph-Chebyshev', 'Kaiser-Bessel',  'van der
```

```
Maas', 'Slepian']
  IF KEYWORD_SET(RETURN_WINDOWLIST) THEN return,windowlist


;First, test Length:
;  If not set, use 16.
;   If it's a scalar, that's the window length.
;   If it's a 1D array, then use the # of elements as the window length.
;   Otherwise, return with an error message.
;
 IF (N_PARAMS() LT 1) THEN length= 16
 siz= SIZE(length)
 CASE siz(0) OF
 0: L= length
 1: L= N_ELEMENTS(length)
 ELSE: MESSAGE,'Length must be a scalar or a 1D array'
 ENDCASE

;Check the Windowtype.
; If none given, use Hanning
; Remove spaces, convert to uppercase
;
 IF (N_PARAMS() LT 2) THEN BEGIN
  MESSAGE,'No Windowtype given, using default: Hanning',/INFORMATIONAL
  Windowtype= 'Hanning'
 ENDIF
 Windowtype= STRUPCASE(STRCOMPRESS(Windowtype,/REMOVE_ALL))
 print,windowtype

;If no additional parameter is provided, set it to 1
;
 IF (N_PARAMS() LT 3) THEN parameter=1.0

 result= 1.0
 nn= 2.0*!pi*FINDGEN(L)/L
 t= 2.0*FINDGEN(L)/L - 1.0       ;-1 to almost 1

  CASE (windowtype) OF


 ;---------------------------------------------------------- ---------------------
 ;The Hanning window is widely used, due to the advantages of a simple
 ; representation in the time domain and an analytic expression for the
 ; transform.  Unfortunately, it has neither a particularly narrow main
 ; lobe nor low sidelobes.
 ;
 'HANNING': wind= 1.0 + cos(!pi*t)
```

```
 ;------------------------------------------------------------ ----------
 ;Although the analytic expression of a taper is (usually) symmetric,
the
 ; FFT form shouldn't be, as the last point is the same as the first,
and
 ; shouldn't be explicitly included.  Here's what the Hanning window
looks
 ; like if done incorrectly.  Not a big difference, but worth getting
right.
 ;
 'BADHANNING': BEGIN
               t= 2.0*FINDGEN(L)/(L-1.0) - 1.0
               wind= 1.0 + cos(!pi*t)
             END


 ;------------------------------------------------------------ ----------
 ;The Hamming window has a time domain representation similar to the
Hanning.
 ;It has a comparable mainlobe width, but a lower first sidelobe.
 ;
 ;
 'HAMMING': wind= 1.0 + 0.46/0.54*cos(!pi*t)


 ;------------------------------------------------------------ ----------
 ;The Blackmann window has a moderately wide main lobe, and low
sidelobes
 ;
 'BLACKMAN':wind= 1.0 + 0.50/0.42*cos(!pi*t) + 0.08/0.42*cos(2.0*!pi*t)



 ;------------------------------------------------------------ ----------
 ;
 ;
 "EXACTBLACKMAN": wind= 1.0 + 1.16402*cos(!pi*t) +
0.180146*cos(2.0*!pi*t)

 'DOLPH-CHEBYSHEV':BEGIN
                   order= L
                   x= 10.0^parameter
                   z0 = COSH( ALOG(x + SQRT(x^2-1.0) ) / order )
;Harris calls this 'beta'
                   arg= z0 * COS(nn/2.0)
                   temp= (-1)^FINDGEN(L) * TCHEBYSHEV(arg,order) / x
                  ; stop
```

```
                wind= FLOAT(FFT(temp,1))
               END

 'KAISER-BESSEL':BEGIN
                b= parameter*!pi
                arg= SQRT( (1.0-t^2) > 0.0)                ;avoid
problems at arg^2 = 1
                wind= BESELI(b*arg,0) / (SINH(b)/b)
               ; stop
               END

 'NORTON-BEER':BEGIN
              n=  FIX(parameter)
              CASE n OF                          ;four possible
window types (0,1,2,3)
               0: c= [1.0, 0.0, 0.0, 0.0, 0.0]
               1: c= [0.384093, -0.087577, 0.703484, 0.0, 0.0]
               2: c= [0.152442, -0.136176, 0.983734, 0.0, 0.0]
               3: c= [0.045335, 0.0, 0.554883, 0.0, 0.399782 ]
               ELSE:MESSAGE,'For Norton-Beer, parameter should be an
integer from 0 (no apodization) to 3 (strong apodization)'
              ENDCASE
              u= 2.0 *  FINDGEN(l) / l - 1.0
              arg= 1.0 - u^2
              wind= c(0)
              FOR i=1,4 DO BEGIN
               IF ( c(i) NE 0.0 ) THEN wind= wind + c(i)*arg^i
              ENDFOR
             END

 'VANDERMAAS':BEGIN
              r= 10.0^parameter         ;peak to sidelobe ratio
              b= ALOG(r + SQRT(r^2-1) ) ;scale factor
              arg= SQRT( (1.0-t^2) > 1.0E-12)
              wind= b* BESELI(b*arg,1) / arg
              wind(0)= (b^2)/2.0     ;ensure that the limit is good for
small arguments
              wind(0)= wind(0) + L  ;then tack on a bit (delta
function)
              wind= wind / COSH(b)  ;normalize for unit peak in
spectrum
            END


 ;------------------------------------------------------------ ------------
 ;
 ;
 ;
```

```
 ;'SLEPIAN':BEGIN
 ;          IF (N_ELEMENTS(parameter) GT 1) THEN order=parameter(1)
ELSE order=0
 ;          n= l+1              ;get 1 more point than necessary
 ;          w= FLOAT(parameter(0))/n    ;frequency width (sort of)
 ;          indx= INDGEN(n)        ;useful index array
 ;          diag= ((n-1.0d0-2.0d0*indx)/2.0d0)^2 * COS(2.0*!dpi*w)
;diagonal elements
 ;          offdiag= indx*(n-indx)/2.0d0                ;off-diagonal
elements;
 ;
 ;          lo= 0.0d0
 ;          hi= 0.45d0*n^2
 ;  ;         evalue=
 TRI_EIGENVALUE(diag,offdiag,evector,RANGE=[lo,hi],ORDER=orde r)
 ;          wind= evector(0:n-2)
 ;         END

 ELSE:BEGIN

       MESSAGE,'Invalid WINDOWTYPE '+windowtype,/INFORMATIONAL
       MESSAGE,'Should be one of: '+windowlist
     END
 ENDCASE


 IF (siz(0) EQ 1) THEN BEGIN
  result= length*wind
  result= SHIFT(result,L/2)
 ENDIF ELSE result= wind


IF KEYWORD_SET(PLOT_EXAMPLE) THEN BEGIN
  !p.multi=[0,1,2]
  spec= 4.0*ABS(FFT([result,REPLICATE(0.0,l*3)],-1))
  spec= 10.0*ALOG10(spec)
  xspec= 2.0*FINDGEN(l*4.0)/(4.0*l)
  every4= FINDGEN(l)*4
  PLOT,t,result,TITLE=windowtype + ' Window',PSYM=-4,SYMSIZE=0.5
  XYOUTS,0.7,!y.crange(1)*0.85,STRING(l,FORMAT='("N=",I3)')
  IF (parameter(0) NE 1) THEN BEGIN
    pstring= STRCOMPRESS( STRING(parameter(0),FORMAT='("p=",F6.1)'))
    XYOUTS,0.7,!y.crange(1)*0.75,pstring
  ENDIF
  !p.multi=[2,2,2]
  PLOT,xspec,spec,TITLE='Amplitude
 Spectrum',YRANGE=[-100,0],XRANGE=[0,0.5],XSTYLE=1,YTITLE='dB ',XTITLE='f
/ fN',XTICKS=2
```

```
  PLOT,xspec*l,spec,TITLE='Amplitude
 Spectrum',YRANGE=[-50,0],PSYM=-4,SYMSIZE=0.3,XRANGE=[0,12],X
STYLE=1,XTICKS=2,XTITLE='Delta
f'
  OPLOT,xspec(every4)*l,spec(every4),PSYM=4,SYMSIZE=0.7
ENDIF

RETURN,result
END
```
