LYRA Flatfield Software: Relative Output as Function of Pointing

IED, 17 Aug 2006
(rev. 26 Apr 2006 after including NI data)

The IDL program for this purpose (see below) asks for the coordinates of the off-pointing. The solar center is assumed to be (x,y) = (0,0), where the first coordinate denotes the east-west dimension, x<0 being east and x>0 being west; the second coordinate denotes the south-north dimension, y<0 being south and y>0 being north.

The program then calculates the beam of the Sun on the LYRA detector plane. The beam is limited by the precision aperture, diameter = 3 mm, distance = 27.75 mm, and also limited by the view-limiting aperture, diameter = 6.4 mm, distance = 27.75 + 70.00 mm. The location of the two images is calculated with the help of a tangens-formula, the resulting real image is the intersection of the two.

The approach was initially limited to channels *-3 and *-4, using “homogeneity” values measured during the LYRA calibration at BESSY (GI campaign, March 2006). Meanwhile, data from the NI campaign was added, thus also covering channels *-1, *-2, and *-3 at a longer wavelength. The program now asks for the LYRA channel (1-1, 1-2,...,3-4) and reads the measured flatfield, i.e. the geometrically distributed relative responses. The solar beam on the detector plane is weighted with this homogeneity. The resulting sum is divided by the result that would be delivered by (0,0) i.e. no off-pointing. The quotient can be called "relative" or "normalized response" of the channel.

This approach uses the documents

- "LYRA Opto-Mechanical Properties", Silvio Koller 06.02.2004
- "Off-pointings for LYRA" from P2SC wiki
- "LYRA" by Hochedez et al. ASR 2006

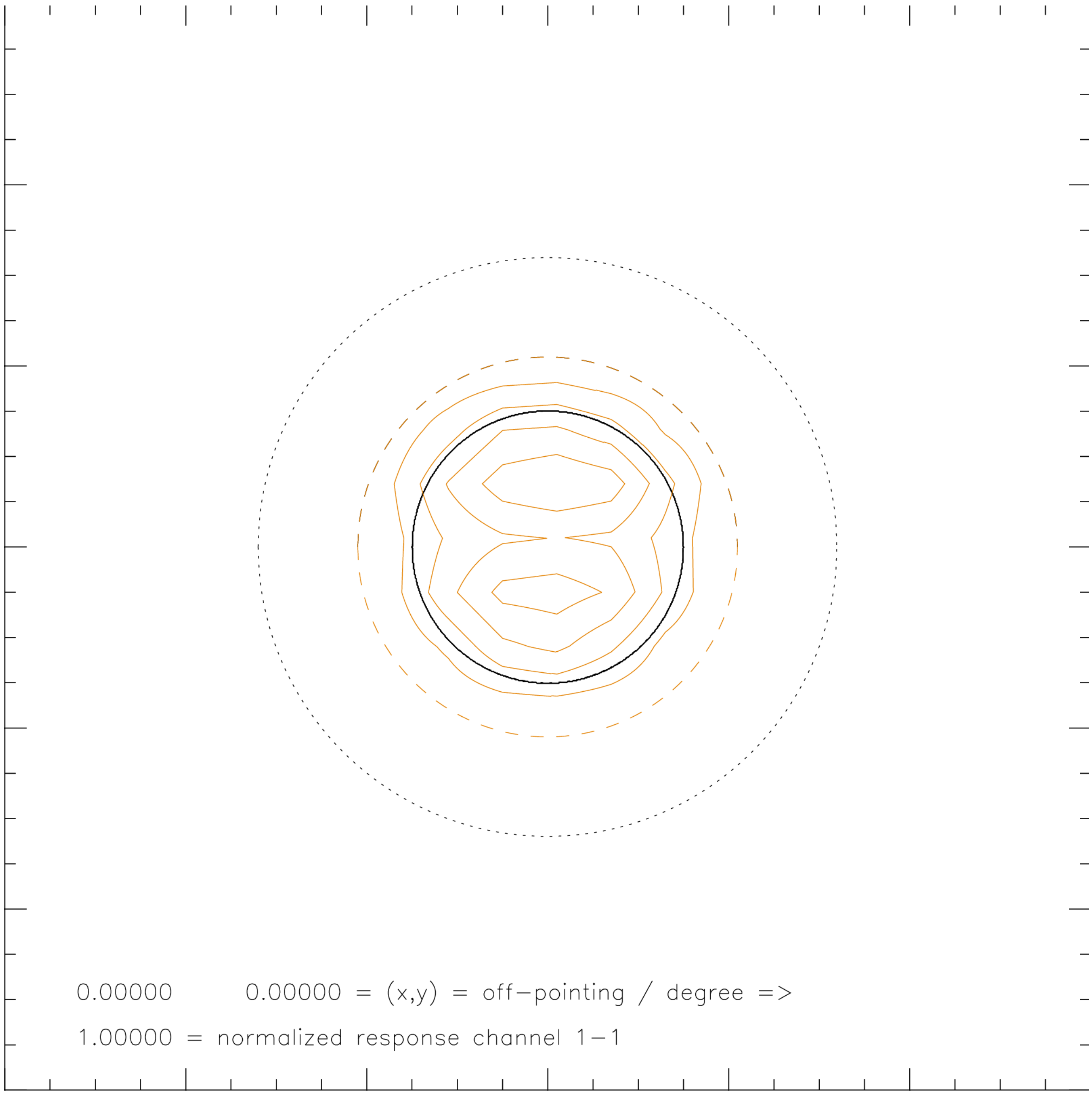
These sources contain differing values for the view-limiting aperture, 6.4 mm diameter TBC. They also contain differing values for the detector sensitive areas, but these do not matter since the actually measured values from the BESSY calibration campaign were used.

The orientation of the detector homogeneity measurements relative to their position on PROBA or their position to SWAP were also determined. With this information, together with the possible off-pointing and the possible roll angle, a general normalization factor (for absolute radiometric calibration) can be calculated. An (x,y)-map of relative responses as a function of off-pointing can be produced, given the limits of the expected pointing range. This will be the content of another report.

One could think of several improvements to get better results for the "normalized response". But it has to be considered first if the greater exactness would be worth the effort:

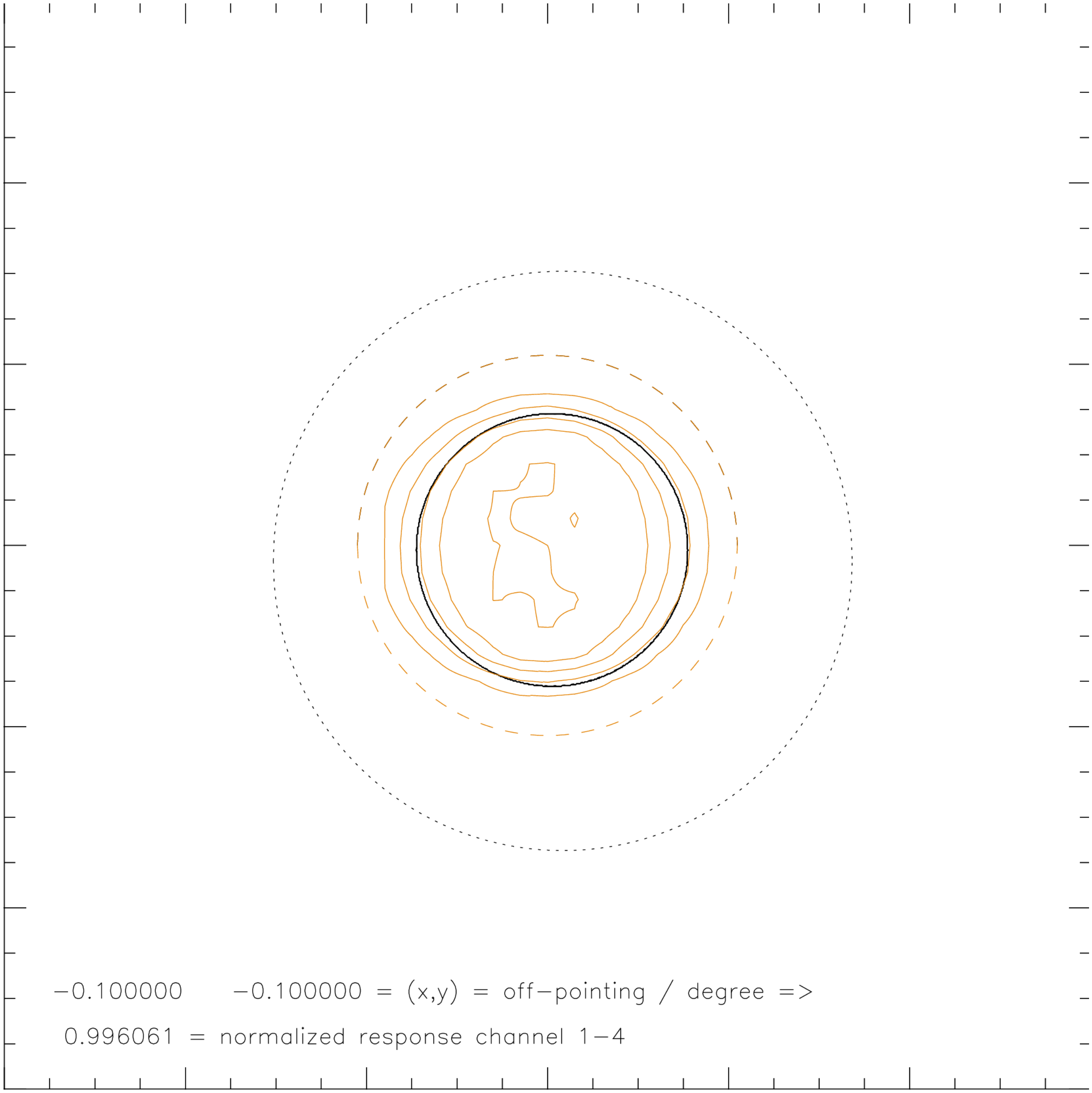
- The extension of the Sun (0.5 degree) leads to a halfshadow on both sides of the border of the solar images; this is, so far, not implemented.
- The resolution used for the numerical calculation of the solar images' intersection is 0.01 mm and could be made smaller.

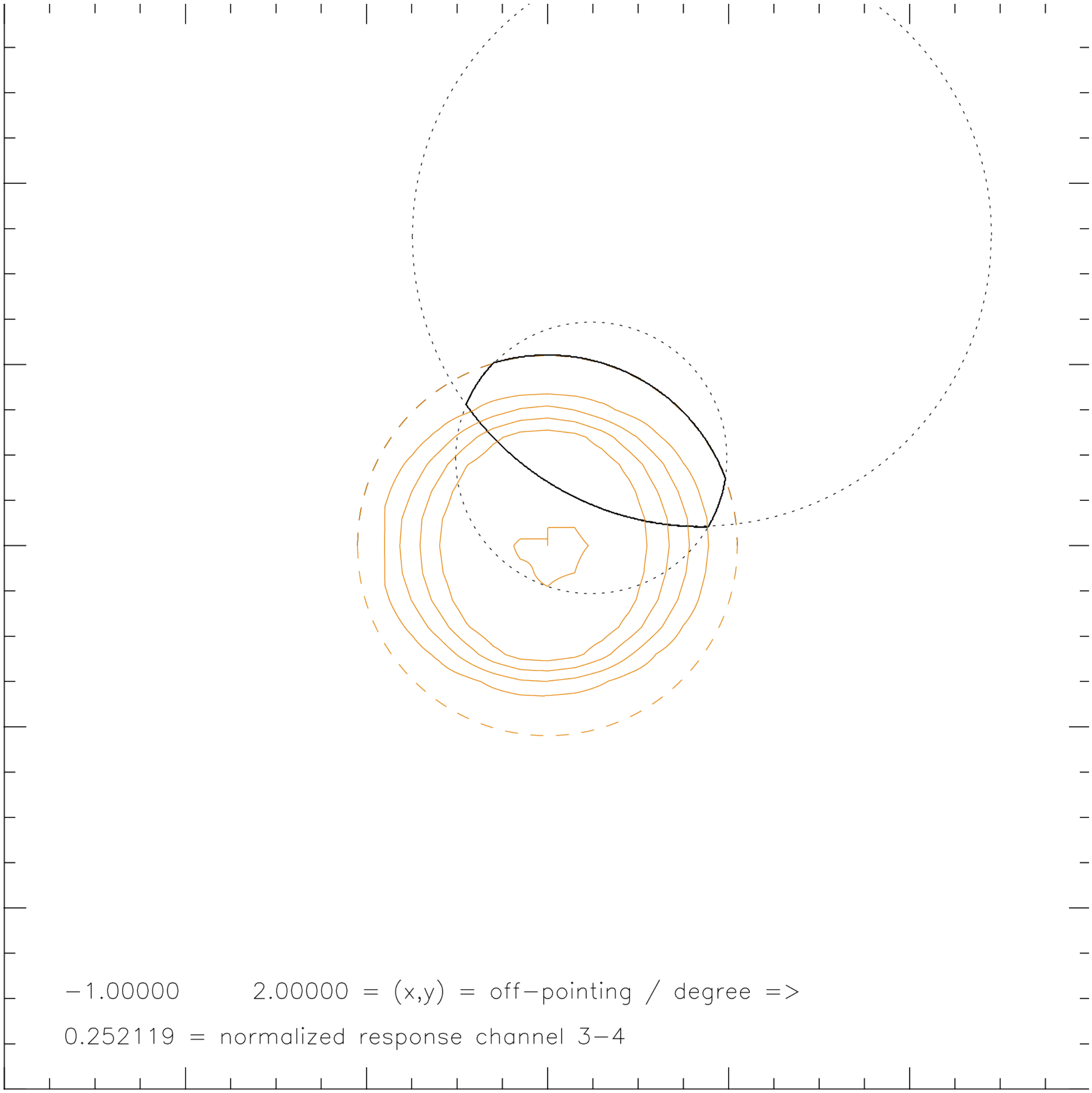
Below: Figures with five examples (various channels, various off-pointing), and IDL program listing.

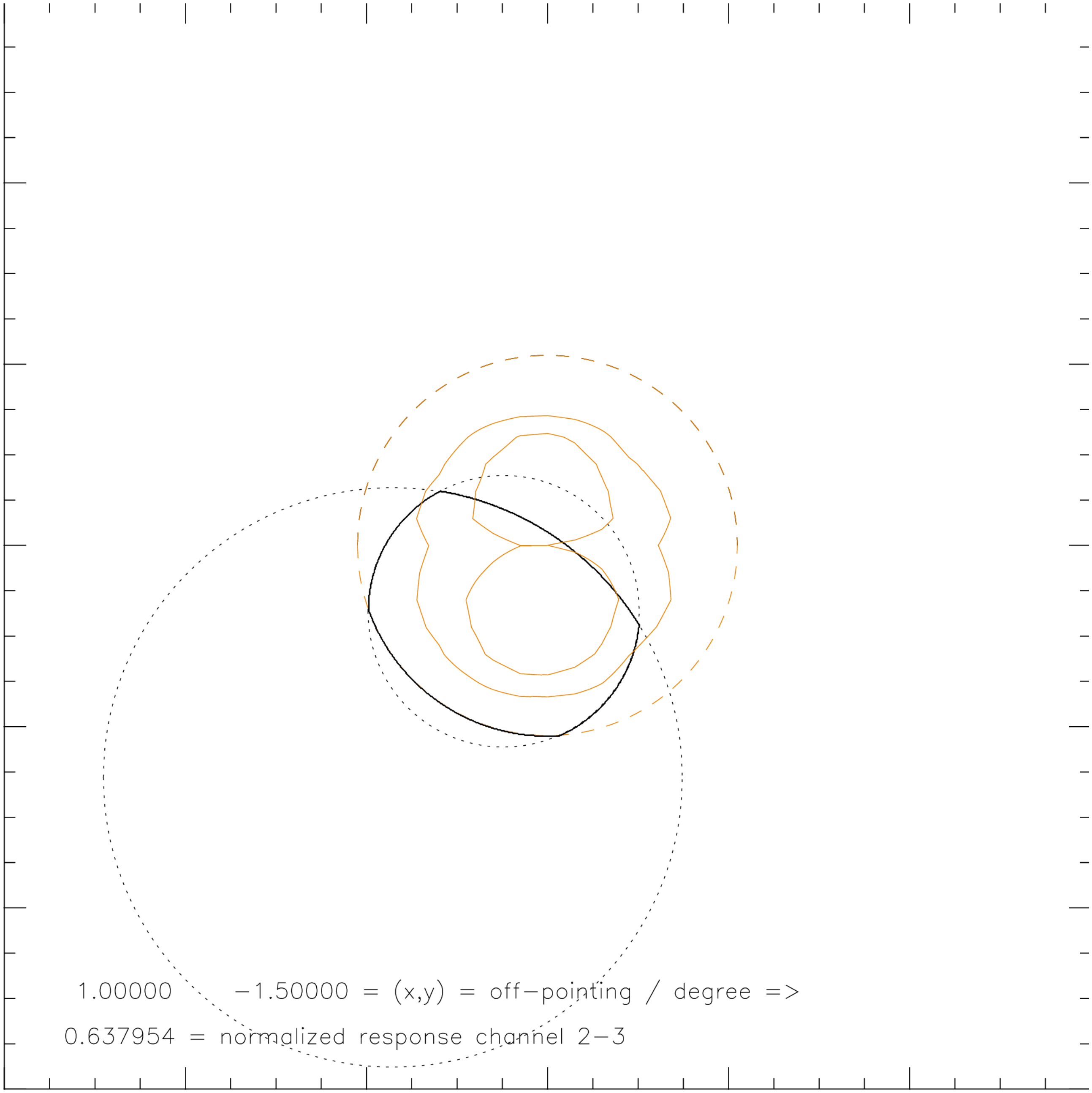


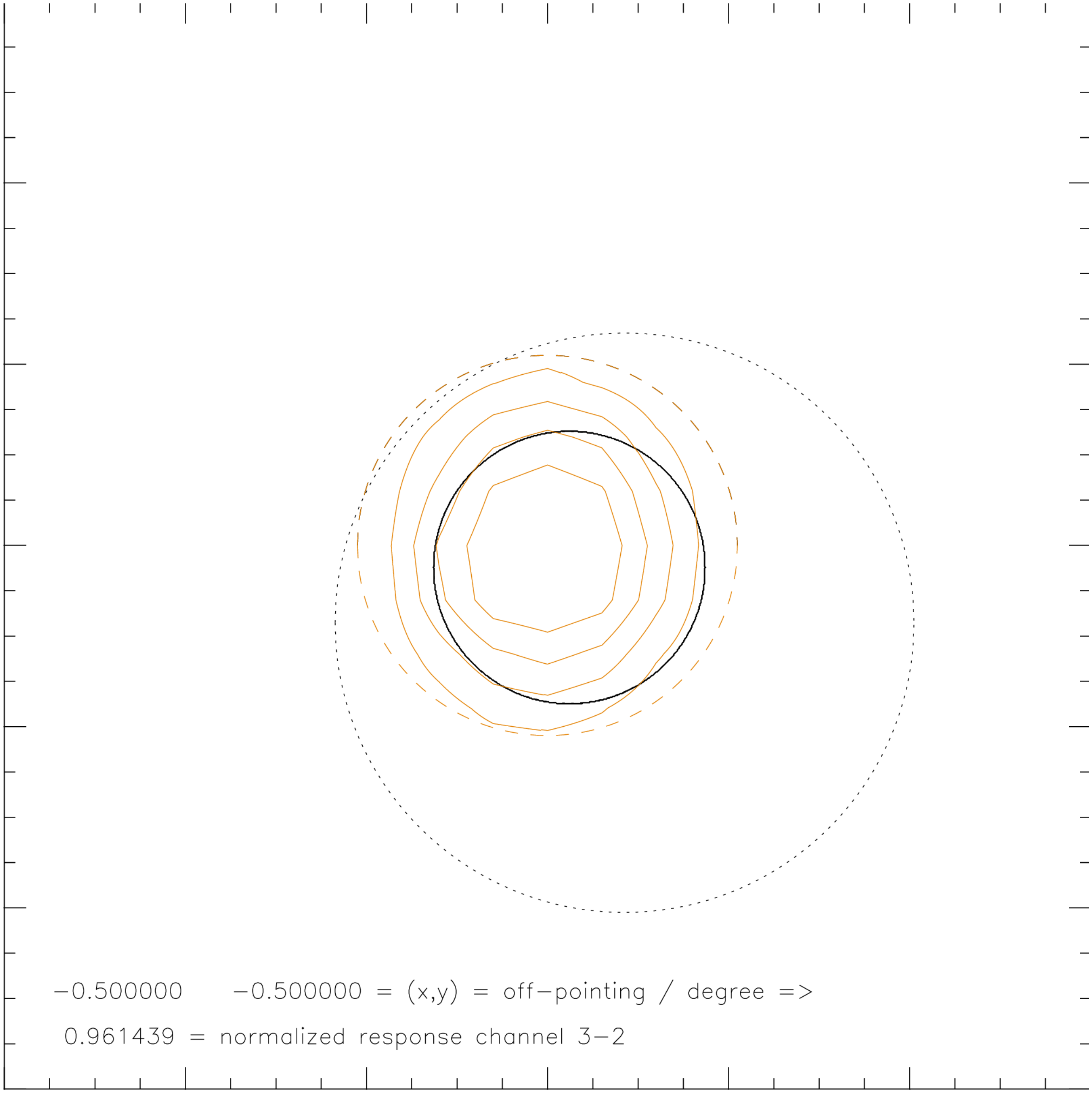
Caption for this and the following figures

- 12 mm x 12 mm extension in detector plane
- red, dashed: detector limits
- red, solid: detector flatfield contour
- black, larger, dotted: image of Sun through 6.4 mm view-limiting aperture
- black, smaller, dotted: image of Sun through 3 mm precision aperture
- black, thick solid: intersection, resulting image of Sun in detector plane


−0.100000 −0.100000 = (x,y) = off−pointing / degree =>
0.996061 = normalized response channel 1−4


−1.00000 2.00000 = (x,y) = off−pointing / degree =>
0.252119 = normalized response channel 3−4


1.00000 -1.50000 = (x,y) = off-pointing / degree =>
0.637954 = normalized response channel 2-3



−0.500000 −0.500000 = (x,y) = off−pointing / degree =>

0.961439 = normalized response channel 3−2

```
; plot_detim_offp.pro
;-------------------------------------------------------------------------------
; IED  25 Apr 2007
;-------------------------------------------------------------------------------
; IDL program to calculate the beam of the Sun on a LYRA detector with 4.2 mm
; diameter, given an off-pointing.
; Coordinates (x=east-west, y=south-north) of solar center = (0,0),
; coordinates of pointing in degrees east x<0, west x>0, south y<0, north y>0
; Aperture with diameter 3 mm at 27.75 mm distance,
; aperture with diameter 6.4 mm at 27.75 + 70.00 mm distance.
; After calculating the beam, the program asks for a LYRA channel (1-1,...,3-4),
; reads the measured homogeneity values, plots them, and calculates the
; normalized detector response.
; Update: Homogeneity values (flatfields) from NI campaigns added,
; values from GI campaigns with coordinate transformations.
; !! .run read_allflatfields.pro first !!
;-------------------------------------------------------------------------------

chpr=0
loadct,3
read,'print .PS (Y=1) ? ',chpr
if (chpr eq 1) then begin
 set_plot,'ps'
 ;device,/portrait,xoffset=1,yoffset=7,xsize=19,ysize=19,bits=8,/color
 device,/portrait,xoffset=1,yoffset=7,xsize=17,ysize=17,bits=8,/encapsulated,/color
 endif else begin
 set_plot,'x'
 window,xsize=600,ysize=600,retain=2
 endelse
xoff=0. & yoff=0.
read,'off-pointing in degree (east-west,south-north) #,# ? ',xoff,yoff
dist=2775.
dist2=dist+7000.
ximgoff=-dist*tan(xoff/180.*!pi)
yimgoff=dist*tan(yoff/180.*!pi)
ximgoff2=-dist2*tan(xoff/180.*!pi)
yimgoff2=dist2*tan(yoff/180.*!pi)
print,ximgoff,yimgoff
print,ximgoff2,yimgoff2
!xmin=-600 & !xmax=600 & !ymin=-600 & !ymax=600
ddet=420
rdet=ddet/2.
dpra=300
rpra=dpra/2.
dvla=640
rvla=dvla/2.
xdet=findgen(ddet+1)-rdet
ydet=sqrt(rdet*rdet-xdet*xdet)
plot,xdet,ydet,linestyle=2,charsize=0.01
loadct,4
oplot,xdet,ydet,linestyle=2,color=200
oplot,xdet,-ydet,linestyle=2,color=200
loadct,0
ximg=findgen(dpra+1)-rpra
ximg2=findgen(dvla+1)-rvla
yimg=sqrt(rpra*rpra-ximg*ximg)
yimg2=sqrt(rvla*rvla-ximg2*ximg2)
oplot,ximg+ximgoff,yimg+yimgoff,linestyle=1
oplot,ximg+ximgoff,-yimg+yimgoff,linestyle=1
oplot,ximg2+ximgoff2,yimg2+yimgoff2,linestyle=1
```

```
oplot,ximg2+ximgoff2,-yimg2+yimgoff2,linestyle=1
detimg=fltarr(1201,1201)
for i=0,ddet do begin
 xco=0>(round(xdet(i))+600)<1200
 yco1=0>(round(-ydet(i))+600)<1200
 yco2=0>(round(ydet(i))+600)<1200
 detimg(xco,yco1:yco2)=detimg(xco,yco1:yco2)+1.
 endfor
for i=0,dpra do begin
 xco=0>(round(ximg(i)+ximgoff)+600)<1200
 yco1=0>(round(-yimg(i)+yimgoff)+600)<1200
 yco2=0>(round(yimg(i)+yimgoff)+600)<1200
 detimg(xco,yco1:yco2)=detimg(xco,yco1:yco2)+1.
 endfor
for i=0,dvla do begin
 xco=0>(round(ximg2(i)+ximgoff2)+600)<1200
 yco1=0>(round(-yimg2(i)+yimgoff2)+600)<1200
 yco2=0>(round(yimg2(i)+yimgoff2)+600)<1200
 detimg(xco,yco1:yco2)=detimg(xco,yco1:yco2)+1.
 endfor
!xmin=0 & !xmax=0 & !ymin=0 & !ymax=0
detimg(0,*)=0
detimg(1200,*)=0
detimg(*,0)=0
detimg(*,1200)=0
detimg(where(detimg lt 3))=0.
detimg=detimg/3.
ok=''
read,'ok ? ',ok
contour,detimg,/noerase,charsize=0.01

ch=''
print,'(NI2006:) 11,12,13n,  ,21,22,23n,  ,31,32,33n,  '
print,'(GI2006:)   ,  ,13g,14,  ,  ,23g,24,  ,  ,33g,34'
read,'channel ## ? ',ch
detff=fltarr(1201,1201)

if (ch eq '11') then begin
 ffhom=ni6_ff11
 ffhom00=congrid(ffhom,600,480,/interp)
 ;detff(600-250,600-210)=ffhom00
 ;detff(600-270,600-210)=ffhom00
 detff(600-290,600-230)=ffhom00
 norm00=43166.0
 endif

if (ch eq '12') then begin
 ffhom=ni6_ff12
 ffhom00=congrid(ffhom,540,540,/interp)
 ;detff(600-250,600-290)=ffhom00
 detff(600-240,600-240)=ffhom00
 norm00=49028.9
 endif

if (ch eq '13n') then begin
 ffhom=ni6_ff13
 ffhom00=congrid(ffhom,600,600,/interp)
 ;detff(600-260,600-250)=ffhom00
 detff(600-270,600-270)=ffhom00
 norm00=48482.6
 endif
```

```
if (ch eq '13g') then begin
 ffhom=gi6_ff13t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-180,600-170)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=48165.6
 endif

if (ch eq '14') then begin
 ffhom=gi6_ff14t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-180,600-170)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=59970.8
 endif

if (ch eq '21') then begin
 ffhom=ni6_ff21
 ffhom00=congrid(ffhom,450,450,/interp)
 ;detff(600-250,600-160)=ffhom00
 ;detff(600-200,600-200)=ffhom00
 detff(600-220,600-200)=ffhom00
 norm00=34516.1
 endif

if (ch eq '22') then begin
 ffhom=ni6_ff22
 ffhom00=congrid(ffhom,540,540,/interp)
 ;detff(600-250,600-290)=ffhom00
 detff(600-240,600-240)=ffhom00
 norm00=49171.4
 endif

if (ch eq '23n') then begin
 ffhom=ni6_ff23
 ffhom00=congrid(ffhom,600,600,/interp)
 ;detff(600-280,600-240)=ffhom00
 detff(600-270,600-270)=ffhom00
 norm00=48197.0
 endif

if (ch eq '23g') then begin
 ffhom=gi6_ff23t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-180,600-180)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=66855.6
 endif

if (ch eq '24') then begin
 ffhom=gi6_ff24t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-180,600-180)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=70408.6
 endif

if (ch eq '31') then begin
 ffhom=ni6_ff31
 ffhom00=congrid(ffhom,540,540,/interp)
 ;detff(600-250,600-240)=ffhom00
 ;detff(600-240,600-240)=ffhom00
 detff(600-260,600-260)=ffhom00
 norm00=53151.3
 endif

if (ch eq '32') then begin
 ffhom=ni6_ff32
 ffhom00=congrid(ffhom,540,540,/interp)
 ;detff(600-250,600-300)=ffhom00
 detff(600-240,600-240)=ffhom00
 norm00=49321.3
 endif

if (ch eq '33n') then begin
 ffhom=ni6_ff33
 ffhom00=congrid(ffhom,600,600,/interp)
 ;detff(600-310,600-240)=ffhom00
 detff(600-270,600-270)=ffhom00
 norm00=52200.3
 endif

if (ch eq '33g') then begin
 ffhom=gi6_ff33t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-195,600-175)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=56904.9
 endif

if (ch eq '34') then begin
 ffhom=gi6_ff34t
 ffhom00=congrid(ffhom,390,390,/interp)
 ;detff(600-180,600-210)=ffhom00
 detff(600-180,600-180)=ffhom00
 norm00=59571.6
 endif

loadct,4
contour,detff,/noerase,color=200,xstyle=4,ystyle=4,charsize=0.01
loadct,0
detimgff=total(detimg*detff)
print,detimgff,'=> normalized: ',detimgff/norm00
chstr=strmid(ch,0,1)+'-'+strmid(ch,1,1)
xyouts,0,100,string(xoff)+string(yoff)+' = (x,y) = off-pointing / degree =>'
xyouts,0,50,string(detimgff/norm00)+' = normalized response channel '+chstr
if (chpr eq 1) then begin
 device,/close
 ;spawn,'lpr -P lpsidc idl.ps'
 set_plot,'x'
 endif

loadct,0
end
```