NIKA2 bandpasses as measured with the PIMP

J.F. Macías-Pérez, A. Bideau, S. Leclercq, A. Monfardini

January 2025, v0

Abstract

We present main results on the measurement of the bandpass of the three NIKA2 arrays using the PIMP spectrometer. The measurements were performed in the Nasmyth cabin of the IRAM 30 m telescope. Different angles of the PIMP polarizer with respect to the NIKA2 polarization reference frame were explored. We performed a first measurement campaign in Summer 2018 during which it was discovered a non homogeneous illumination of the NIKA2 focal plane. This was corrected for the second campaign, which was performed in September 2019. In both campaigns we have observed variations of the NIKA2 transmission as a function of the angle of the PIMP polarizer. Overall we find that the NIKA2 bandpasses reconstructed with the PIMP are fully consistent with those measured previously in the lab. We conclude that the PIMP allows precise bandpass measurements on site. We expect it to be an useful tool for future upgrades of the NIKA2 camera.

1 NIKA2 bandpass measurements with the PIMP

1.1 The PIMP spectrometer

The PIMP is a compact Martin Puplett Interferometer (MPI) constructed in house by Institut Néel and IRAM. It was specifically designed to be able to perform measurements of the NIKA2 transmission directly at the telescope in complement to those performed at the lab with the Neel-LPSC MPI.

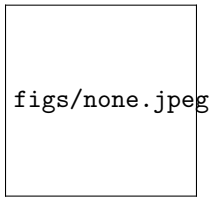

Figure 1: Picture of the PIMP during the 2019 observation campaign.

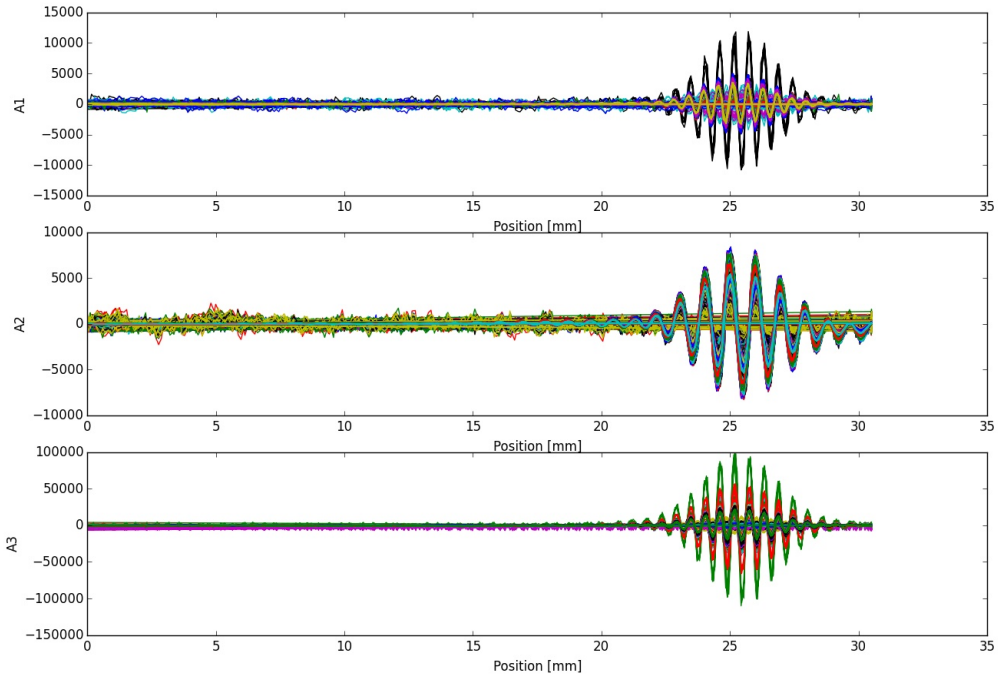

Figure 2: FWD interferograms per detector as a function of displacement mirror position for the the three NIKA2 arrays (A1, A2 and A3 from top to bottom). The data were obtained during the September 2019 campaign with the convergent lens and in the case of the PIMP polarizer at zero degrees with respect to the NIKA2 polarization reference frame.

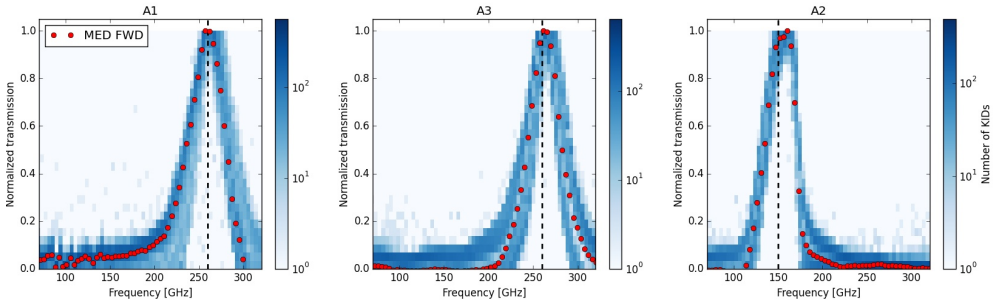

Figure 3: Histograms of the FWD spectra per detector as a function of frequency for the the three NIKA2 arrays (A1, A2 and A3 from left to right). The averaged median spectrum of all detectors is shown are red full dots. The data were obtained during the September 2019 campaign with the convergent lens and in the case of the PIMP polarizer at zero degrees with respect to the NIKA2 polarization reference frame.

Table 1: Details on the NIKA2 bandpass measurement campaigns using the PIMP spectrometer.

Campaign	Data File	PIMP angle	Comments
Summer 2018	X_2018_06_18_22h51m00_0183_I_2023+336	0	
	X_2018_06_18_23h46m35_0206_P_1739+522	90	
	X_2018_06_18_23h50m32_0207_I_1842+681	90	
	X_2018_06_18_23h51m52_0208_P_1842+681	90	
	2018_06_19_09h59m34	45	
	2018_06_19_10h59m27	75	
	2018_06_19_11h13m59	105	
	2018_06_19_11h30m16	15	
	2018_06_19_11h43m03	345	
September 2019			
	X_2019_09_20_21h16m49	-90	Convergent lens
	X_2019_09_20_21h20m23	-60	Convergent lens
	X_2019_09_20_21h23m17	-30	Convergent lens
	X_2019_09_20_21h26m15	0	Convergent lens
	X_2019_09_20_21h29m37	30	Convergent lens
	X_2019_09_20_21h32m20	60	Convergent lens
	X_2019_09_20_21h35m32	90	Convergent lens
	X_2019_09_20_23h36m27	-90	Divergent lens
	X_2019_09_20_23h39m28	-60	Divergent lens
	X_2019_09_20_23h41m52	-30	Divergent lens
	X_2019_09_20_23h43m52	0	Divergent lens
	X_2019_09_20_23h46m32	30	Divergent lens
	X_2019_09_20_23h48m55	60	Divergent lens
	X_2019_09_20_23h51m58	90	Divergent lens

1.2 Bandpass measurement campaigns

We had two main bandpass measurement campaigns in Summer 2018 and September 2019. In both campaigns measurements were performed with the PIMP input polarizer oriented at different angles with respect to the NIKA2 polarization reference frame. In this way we can check the variations of the NIKA2 bandpasses with respect to the incident polarization angle. In the Summer 2018 campaign, as discussed in more details below, we observed a non homogeneous illumination of the NIKA2 focal plane. To correct for this an extra divergent or convergent lens was added for the 2019 campaign. We present in Table 1 we present more details on the measurements performed.

1.3 Data analysis

We acquired few minutes of data for each setup to have a sufficiently large number of independent interferograms, which can then be averaged to increase the signal-to-noise. For each detector Time Ordered Data (TOI) of the signal was constructed using the *RF_dI_dQ* approximation. Low-frequency drifts in the TOIs were filtered out. In parallel, a TOI of the position of the PIMP displacement mirror is produced and the forward and backward directions are identified. The detector data is divided in two sets, FWD and BWD, corresponding to the forward and backward directions defined before. The FWD and BWD TOIs are reordered with respect to the displacement mirror position and the Zero Path Difference of the interferograms is identified for each detector. The FWD, and BWD, interferogram per detector are resampled and averaged into a common Optical Path Difference (OPD) grid. This operation leads to a BWD and FWD interferogram per detector. We show in Figure 2 the interferograms per detector for the three NIKA2 arrays A1, A2 and A3. We observe that the amplitude of the peak of the interfergrams as well as the ZPD (given by the mirror position of the peak) vary across detectors. The latter are transformed into spectra using a inverse cosine transform. Normalized individual detector spectra are combined to obtain the averaged spectra per array for FWD and BWD. We show in Figure3 the histogram of the reconstructed spectra for the FWD data for the three NIKA2 arrays. We also show as red dots the averaged spectra across detectors. These data were obtained during the September 2019 campaign with the convergent lens and in the case of the PIMP polarizer at zero degrees with respect to the NIKA2 polarization reference frame.

2 Results

The results obtained in the Summer 2018 and September 2019 campaigns are overall consistent. We present here a summary of those including overall results for the September 2019 campaign and some observed difference between the two.

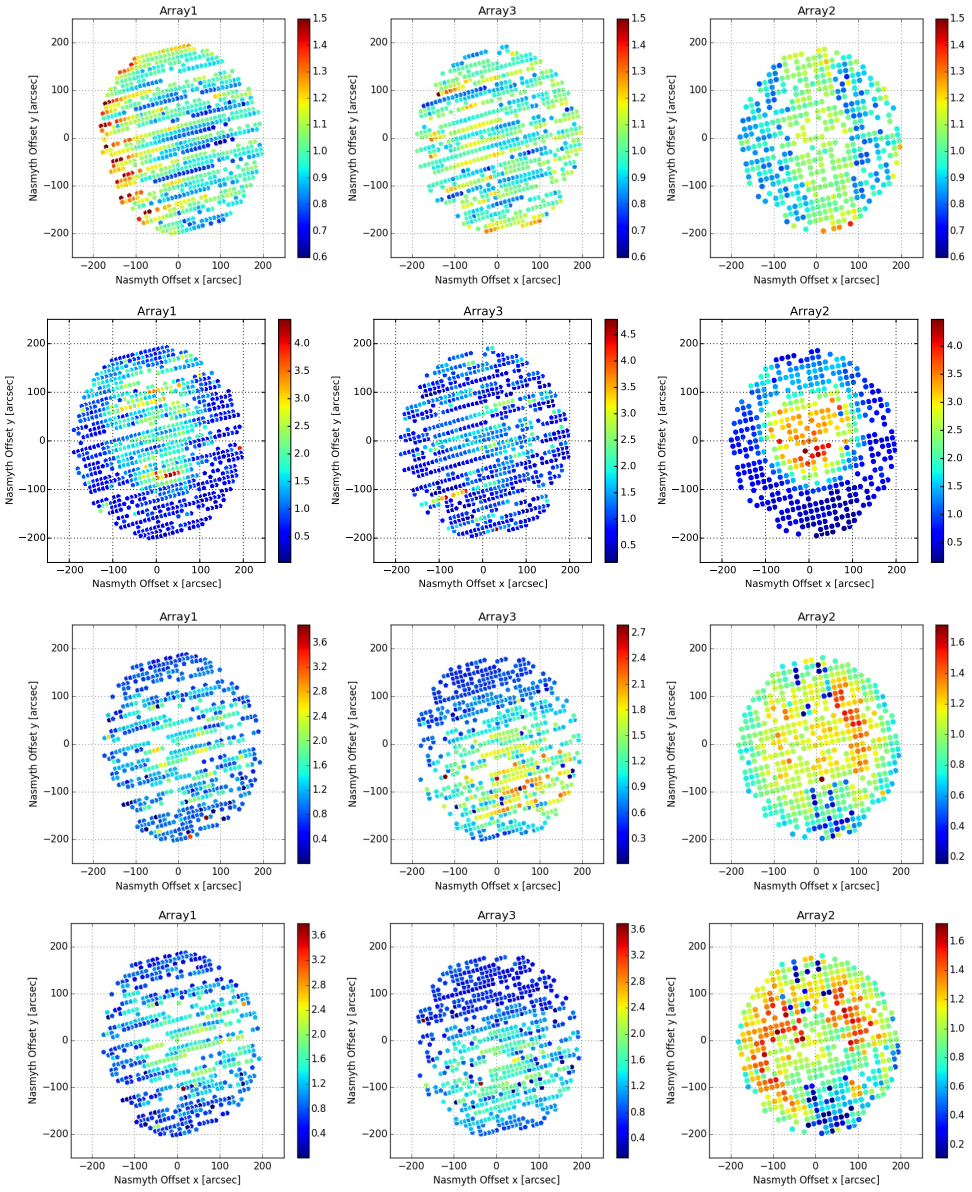

Figure 4: Top: NIKA2 near to far field calibration ratio across the FOV. Next three rows: Maximum amplitude of the interferograms for the Summer 2018 and the September 2019 (convergent and divergent extra lens) campaigns.

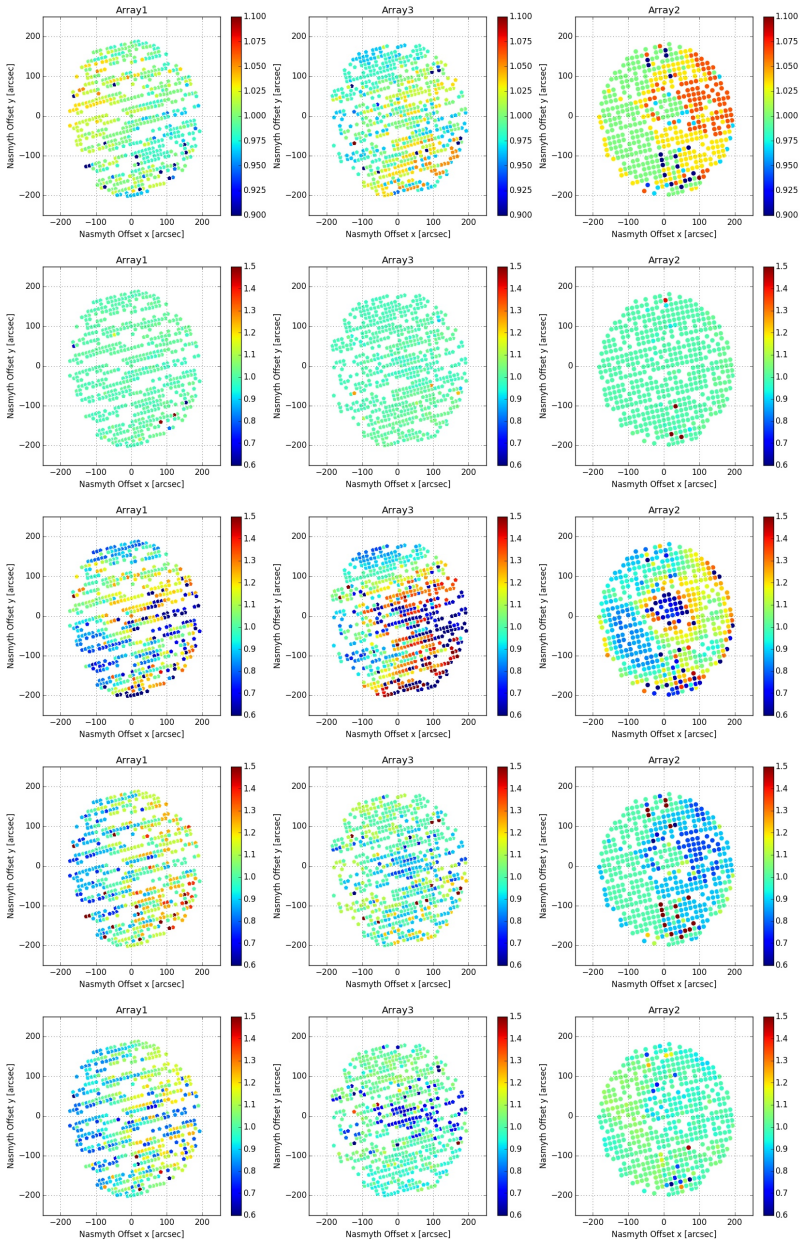

Figure 5: Relative variation of bandpass properties across the FOV for the three NIKA2 arrays. From top to bottom, we show the minimum and maximum frequency, the barycenter frequency, the FWHM and the integrated bandpass. The data were obtained during the September 2019 campaign with the convergent lens and in the case of the PIMP polarizer at 60 degrees with respect to the NIKA2 polarization reference frame.

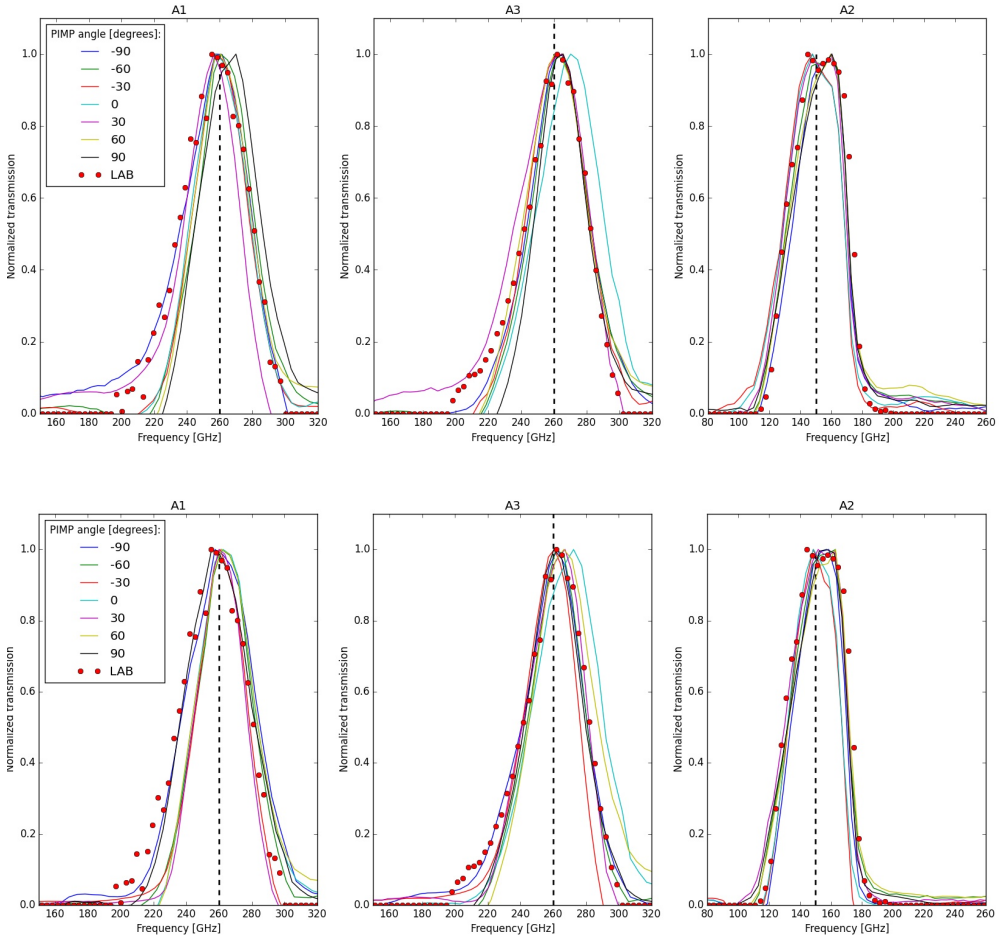

Figure 6: Reconstructed bandpasses for the three NIKA2 arrays at different PIMP polarizer angles for the convergent (top) and divergent (bottom) extra lenses. The data were obtained during the September 2019 campaign.

2.1 Illumination of the focal plane

During the August 2018 campaign we observed a clear non-uniform illumination of the NIKA2 focal plane by the PIMP. This can be observed in the second row of Figure 4 where we show the relative maximum amplitude of the measured interferograms for each KID as a function of their position in the focal plane. We considered in the figure measurements with the PIMP polarizer at zero degrees, but it can also seen for other angles. We observe an over-illuminated region in the center of the array while the outskirts are poorly illuminated. This is particularly the case for array 3. This illumination does not seem to be related NIKA2 instrinsic properties as suggested by the relative near to far field calibration ratio presented in the first row of the figure. To correct for this effect an extra convergent or divergent lens was added to the PIMP for the September 2019 campaign. We show in the last two rows of the figure the results obtained with them for the same configuration than in August 2018. We observe a significant improvement on the homogeneity of the illumination.

2.2 Bandpass property variations across the FOV

As shown in Figure 3 we observe variations in the bandpass across detector. In Figure **??** investigate those variations as a function of the position of the detector in the NIKA2 focal for the three arrays. From top to bottom we consider the relative variation on the minimum, maximum and barycenter frequencies for which the normalized bandpass transmission is above 0.5. We also show the FHWM for the best Gaussian fit and the integrated bandpass. Overall we observe relative small variations that in some cases seems to be correlated to the relative near to far field calibration ratio shown on the first row of Figure 4.

2.3 Bandpass variation with polarization angle

By averaging the normalized bandpass for all detectors (we have excluded badly measured ones) in a given array we obtain the NIKA2 bandpasses per array for the different PIMP polarizer orientations disccussed above. We show these averaged bandpassed in Fig 6 and compare them to the transmission measured in the laboratory. Although we find some variations with the angle the median transmission is consistent with that measured in the laboratory.

3 Conclusions

The PIMP have proved a very useful tool to measure the NIKA2 transmission directly at the telescope. After two successful measurement campaigns we have been able to confirm the NIKA2 laboratory measurements. We have also found some variation of the NIKA2 bandpasses with respect to the orientation of the PIMP polarizer with respect to NIKA2 polarization reference frame. Although we find some variations across the FOV these seem to be minor. We anticipate the PIMP will be very useful to tool for future NIKA2 upgrades.

A Analysis code

```
import numpy as np
import matplotlib.pyplot as plt
import sys
#sys.path.insert(0, "/home/macias/NIKA/Processing/NIKA_lib_AB_OB_gui/Readdata/Python/")
sys.path.insert(0, "/home/macias/NIKA/Processing/NIKA_lib_V3_IRAM/Readdata/Python/")
sys.path.insert(0, "/home/macias/NIKA/Processing/Pipeline/Pynk/")
import read_nika_data as rnd
import os
from scipy.ndimage import median_filter
from scipy import signal
from scipy.interpolate import interp1d
import astropy.constants as cst
from scipy.fftpack import dct, idct
from matplotlib.colors import LogNorm
import hfits
import astropy.io.fits as fits
import string

def get_pimp_listdata():
    list_data= 'sample  I ofs_X ofs_Y Q dI dQ subscan scan scan_st obs_st RF_didq '
    list_data += 'Y_pimp_pos Y_pimp_tcold Y_pimp_thot  Y_pimp_angle Y_pimp_scan Y_pimp_sscan  Y_pimp_i
    boxes = string.ascii_uppercase[0:20]
    for box in boxes:
        list_data = list_data + ' '+box+'_time_pps'
        list_data = list_data + ' '+box+'_hours'
    return list_data

def kid_hist(data, detsel= [] , nbins = 10,crange= []):
    """
    Compute histogram of data across KIDs

    Parameters
    ----------
    data: numpy array 2D (nkids,nd)
          data for KIDs

    delsel: numpy array, optional
          Indices of KIDs selected for the analysis

    nbins:  integer, optional
          Number of bins for the histograms

    crange: (float, float), optional
          The lower and upper range of the bins


    Returns
    -------
    rhist: numpy array 2D
```

```
        KIDs histogram for quantity

    Notes
    -----

    Examples
    --------

    History:
    -------
    First version: JFMP sep 2019

    """
    # Check main keywords
    if len(detsel) == 0: detsel = np.arange(data.shape[0])
    if len(crange) == 0: crange= (data.min(),data.max())
    #Start work
    nd = data.shape[1]
    rhist = np.zeros((nbins,nd))
    for index in range(nd):
        mh,mbed = np.histogram(data[detsel,index].ravel(),bins=nbins,range=crange)
        rhist[:,index] = mh
    return rhist


def int2spec(interin,xpos,n2):
    """
    Compute spectre for a given input interferogramme

    Inputs:
    -------


    Outputs:
    -------
    """
    c_val = cst.c.to('mm/s')
    interbis = np.copy(interin)
    interbis[np.isnan(interbis)] = 0.0
    posmaxmed = np.argmax(interbis)
    pstart = posmaxmed - n2
    if pstart < 0:
        #print "Stop program !!!!"
        pstart = 0
        posmaxmed = n2
    interbis = interbis[pstart:posmaxmed+1]

    x0 = (xpos[pstart:]-xpos[posmaxmed])
    x0max = np.max(np.abs(x0))
    delta_freq = c_val.value/4.0/x0max/1e9
```

```
    freq = np.arange(n2+1)*delta_freq

    internew = np.concatenate((interbis,np.flip(interbis[1:],0)),axis=0)
    internew = np.roll(internew,-(n2+1))

    # FFFT inverse
    spfft = np.fft.ifft(internew).real
    spfft[0] = 0.0
    spfft = spfft[0:n2+1]

    # COST inverse
    spcost = idct(internew[0:n2+1])
    spcost[0] = 0.0

    return freq, spfft,spcost,interbis,internew

def match_numdet(numdet1,numdet2):

    n1 = len(numdet1)
    n2 = len(numdet2)

    w1=[]
    w2=[]
    for i1 in range(n1):
        for i2 in range(n2):
            if numdet1[i1] == numdet2[i2]:
                w1.append(i1)
                w2.append(i2)

    return np.asarray(w1),np.asarray(w2)


def spec_prop(fr,spc):

    maxfreq = fr[np.argmax(spc)]
    medfreq = np.sum(fr * spc)/np.sum(spc)

    pos  = np.where(spc/np.max(spc) >= 0.5)
    fminc = np.min(fr[pos])
    fmaxc = np.max(fr[pos])
    fwhm = np.max(fr[pos]) -  np.min(fr[pos])
    bp   = np.sum(spc/np.max(spc))* (fr[2]-fr[1])

    return maxfreq,medfreq,fwhm,bp,fminc,fmaxc

#interactive mode
#plt.ion()
plt.ioff()

dir_plot_base ='/home/macias/NIKA/Plots/PIMPSeptember2019Divergente/'
```

```
day1 = 'X36_2019_09_20'

# Convergente
label_arr1 = np.asarray(['X_2019_09_20_21h16m49','X_2019_09_20_21h20m23','X_2019_09_20_21h23m17','X_20
                         'X_2019_09_20_21h29m37','X_2019_09_20_21h32m20','X_2019_09_20_21h35m32'])



# Divergente
#label_arr1=['X_2019_09_20_23h36m27', 'X_2019_09_20_23h39m28','X_2019_09_20_23h41m52','X_2019_09_20_23

# Lentille divergenteday1 = 'X36_2019_09_20'

dir_plot_base ='/home/macias/NIKA/Plots/PIMPSeptember2019/'
day1 = 'X36_2019_09_20'

day_arr1 = np.repeat(day1,len(label_arr1))

file_arr1 = label_arr1



plate_pos1 = np.asarray(['-90','-60','-30','0','30','60','90'])


day_arr = day_arr1 #np.concatenate((day_arr1,day_arr2),axis=0)
label_arr = label_arr1 # np.concatenate((label_arr1,label_arr2),axis=0)
file_arr = file_arr1 #np.concatenate((file_arr1,file_arr2),axis=0)
plate_pos = plate_pos1 # np.concatenate((plate_pos1,plate_pos2),axis=0)

hdu = fits.open('/home/macias/NIKA/Processing/Kidpars/kidpar_N2R29_ref_baseline_BL.fits')
kidparref = hdu[1].data
w2ok = np.where( (kidparref['array'] == 2) & (kidparref['type'] ==1))[0]
w1ok = np.where( (kidparref['array'] == 1) & (kidparref['type'] ==1))[0]
w3ok = np.where( (kidparref['array'] == 3) & (kidparref['type'] ==1))[0]


for ifile in range(len(label_arr)):
    mfile = label_arr[ifile]
    day   = day_arr[ifile]
    dir_data = '/mnt/NewData/NIKA/Data/run47Pimp_X/'+day+'/'
    #file_nika = dir_data + 'X_'+mfile+'_AA_man'
    file_nika = dir_data + file_arr[ifile]+'_AA_man'
    print ("Working on file:" + file_nika)
    dir_plot = dir_plot_base+ mfile+'/'
    if os.path.isdir(dir_plot) == False:
        os.mkdir(dir_plot)

    silent = 1
    datas =  rnd.read_nika_data(file_nika,silent=1,list_data='all', nodata=True)
```

```
    data =  rnd.read_nika_data(file_nika,silent=silent,list_data=  get_pimp_listdata())

    ntotdet, ntotsamples = data.RF_didq.shape

    w2 = np.where( (data.kidpar['array'] == 2) & (data.kidpar['typedet'] ==1))[0]
    w1 = np.where( (data.kidpar['array'] == 1) & (data.kidpar['typedet'] ==1))[0]
    w3 = np.where( (data.kidpar['array'] == 3) & (data.kidpar['typedet'] ==1))[0]

    w1,ww1 = match_numdet(data.kidpar['num'],kidparref['NUMDET'][w1ok])
    w2,ww2 = match_numdet(data.kidpar['num'],kidparref['NUMDET'][w2ok])
    w3,ww3 = match_numdet(data.kidpar['num'],kidparref['NUMDET'][w3ok])

    # check data per array


    smbase = 101
    dint = np.copy(-1.0*data.RF_didq)
    for index in range(len(dint)):
    #    dint[index,:] -= np.median(dint[index,:])
        dint[index,:] -= median_filter(dint[index,:],smbase)



    istart = 0
    iend   = len(data.Y_pimp_pos)
#    if ifile == 0: istart = 160
#    if ifile == 2: iend = 1780

    wscan = np.where(data.Y_pimp_sscan >=0)[0]
    istart = wscan[0]
    iend = wscan[-1]
    Y_pimp_pos = data.Y_pimp_pos[istart:iend]
    dint = dint[:,istart:iend]
    # check backward and forward scans
    wfwd = np.where(np.gradient(Y_pimp_pos) > 0.0)[0]
    wbwd = np.where(np.gradient(Y_pimp_pos) < 0.0)[0]


    # some plots for checking
    fig1,ax1 = plt.subplots(ncols=1,nrows=3,figsize=(15,10))

    for index in w1: ax1[0].plot(Y_pimp_pos[wfwd]/2000.0,dint[index,wfwd])
    ax1[0].set_xlabel('Position [mm]')
    ax1[0].set_ylabel('A1')


    for index in w2: ax1[1].plot(Y_pimp_pos[wfwd]/2000.0,dint[index,wfwd])
    ax1[1].set_xlabel('Position [mm]')
    ax1[1].set_ylabel('A2')
```

```
for index in w3: ax1[2].plot(Y_pimp_pos[wfwd]/2000.0,dint[index,wfwd])
ax1[2].set_xlabel('Position [mm]')
ax1[2].set_ylabel('A3')
fig1.savefig(dir_plot+'FWD_all.jpeg')
plt.close(fig1)

fig1,ax1 = plt.subplots(ncols=1,nrows=3,figsize=(15,10))

for index in w1: ax1[0].plot(Y_pimp_pos[wbwd]/2000.0,dint[index,wbwd])
ax1[0].set_xlabel('Position [mm]')
ax1[0].set_ylabel('A1')

for index in w2: ax1[1].plot(Y_pimp_pos[wbwd]/2000.0,dint[index,wbwd])
ax1[1].set_xlabel('Position [mm]')
ax1[1].set_ylabel('A2')

for index in w3: ax1[2].plot(Y_pimp_pos[wbwd]/2000.0,dint[index,wbwd])
ax1[2].set_xlabel('Position [mm]')
ax1[2].set_ylabel('A3')
fig1.savefig(dir_plot+'BWD_all.jpeg')
plt.close(fig1)

pwfwd = np.where(np.abs(wfwd - np.roll(wfwd,-1)) > 200)[0]
if pwfwd[0] > 0: pwfwd = np.insert(pwfwd,0,0)
nsfwd = len(pwfwd)-1
pwbwd = np.where(np.abs(wbwd - np.roll(wbwd,-1)) > 200)[0]
if pwbwd[0] > 0: pwbwd = np.insert(pwbwd,0,0)
nsbwd = len(pwbwd)-1

pminfwd = -1e6
pmaxfwd = 1e6
dpos_fwd = 0.0
for index in range(nsfwd):
    r1 = pwfwd[index].astype(np.long)
    r2 = pwfwd[index+1].astype(np.long)
    pminfwd = np.max( [pminfwd, np.min(Y_pimp_pos[wfwd[r1:r2]])])
    pmaxfwd = np.min( [pmaxfwd, np.max(Y_pimp_pos[wfwd[r1:r2]])])
    dpos_fwd = np.max( [dpos_fwd, np.median(np.gradient(Y_pimp_pos[wfwd[r1:r2]]))])
    print pminfwd, pmaxfwd, dpos_fwd


xpos_fwd = np.arange(pminfwd,pmaxfwd,dpos_fwd)/2000.0
dint_fwd = np.zeros((ntotdet,nsfwd,len(xpos_fwd)))
for idet in range(ntotdet):
    for index in range(nsfwd):
        r1 = pwfwd[index].astype(np.long)
        r2 = pwfwd[index+1].astype(np.long)
        mx = Y_pimp_pos[wfwd[r1:r2]]/2000.0
        my = dint[idet,wfwd[r1:r2]]
        myfinterp = interp1d(mx,my)
```

```
        dint_fwd[idet, index,:] = myfinterp(xpos_fwd)

pminbwd = -1e6
pmaxbwd = 1e6
dpos_bwd = 0.0
for index in range(nsbwd):
    r1 = pwbwd[index].astype(np.long)
    r2 = pwbwd[index+1].astype(np.long)
    pminbwd = np.max( [pminbwd, np.min(Y_pimp_pos[wbwd[r1:r2]])])
    pmaxbwd = np.min( [pmaxbwd, np.max(Y_pimp_pos[wbwd[r1:r2]])])
    dpos_bwd = np.max( [dpos_bwd, np.abs(np.median(np.gradient(Y_pimp_pos[wbwd[r1:r2]])))])
    print pminbwd, pmaxbwd,dpos_bwd

xpos_bwd = np.arange(pminbwd,pmaxbwd,dpos_bwd)/2000.0
dint_bwd = np.zeros((ntotdet,nsbwd,len(xpos_bwd)))/2000.0
for idet in range(ntotdet):
    for index in range(nsbwd):
        r1 = pwbwd[index].astype(np.long)
        r2 = pwbwd[index+1].astype(np.long)
        mx = Y_pimp_pos[wbwd[r1:r2]]/2000.0
        my = dint[idet,wbwd[r1:r2]]
        myfinterp = interp1d(mx,my)
        dint_bwd[idet, index,:] = myfinterp(xpos_bwd)


idet = 2000
fig, ax = plt.subplots(ncols=1,nrows=2,sharex=True,squeeze=True,figsize=(12,8))
fig.subplots_adjust(hspace=0)
ax[0].plot(xpos_fwd,dint_fwd[idet,:,:].T,'o')
ax[0].plot(xpos_fwd,np.median(dint_fwd[idet,:,:],axis=0),linewidth=2,label='MEDIAN')
ax[0].plot(xpos_fwd,np.mean(dint_fwd[idet,:,:],axis=0),linewidth=2,label='MEAN')
ax[0].legend(loc=3)
ax[0].grid()
ax[0].set_ylabel('Power [Hz]')
ax[0].set_title('Detector '+ data.kidpar['name'][idet])

ax[1].plot(xpos_bwd,dint_bwd[idet,:,:].T,'o')
ax[1].plot(xpos_bwd,np.median(dint_bwd[idet,:,:],axis=0),linewidth=2,label='MEDIAN')
ax[1].plot(xpos_bwd,np.mean(dint_bwd[idet,:,:],axis=0),linewidth=2,label='MEAN')
ax[1].grid()
ax[1].set_xlabel('Mirror position [mm]')
ax[1].set_ylabel('Power [Hz]')
fig.savefig(dir_plot+'FWD_BWD_single_det.jpeg')
plt.close(fig)

# Compute median and mean interferogramme and spectra

h0  =fits.PrimaryHDU()
h1  = fits.ImageHDU(data=xpos_fwd,name='XPOS')
h2  = fits.ImageHDU(data=dint_fwd,name='INTER')
```

```
hdu = fits.HDUList([h0,h1,h2])
hdu.writeto(dir_plot+'inter_fwd.fits',overwrite='True')


#intermaxpos_fwd = []
intermean_fwd = []
intermed_fwd  = []
#intermaxpos_bwd = []
intermean_bwd = []
intermed_bwd  = []

spmean_fwd = []
spmed_fwd = []
spcmean_fwd = []
spcmed_fwd = []
freq_fwd = []

spmean_bwd = []
spmed_bwd = []
spcmean_bwd = []
spcmed_bwd = []
freq_bwd = []



n2 = 225
for idet in range(ntotdet):

    intermean = np.mean(dint_fwd[idet,:,:],axis=0)
    freq, spfft,spcost,interbis,internew = int2spec(intermean,xpos_fwd,n2)
    spmean_fwd.append(np.asarray(spfft))
    spcmean_fwd.append(np.asarray(spcost))
    freq_fwd.append(freq)

    intermed = np.median(dint_fwd[idet,:,:],axis=0)
    freq, spfft,spcost,interbis,internew = int2spec(intermed,xpos_fwd,n2)
    spmed_fwd.append(np.asarray(spfft))
    spcmed_fwd.append(np.asarray(spcost))

    intermean = np.mean(dint_bwd[idet,:,:],axis=0)
    freq, spfft,spcost,interbis,internew = int2spec(intermean,xpos_bwd,n2)
    spmean_bwd.append(np.asarray(spfft))
    spcmean_bwd.append(np.asarray(spcost))
    freq_bwd.append(freq)

    intermed = np.median(dint_bwd[idet,:,:],axis=0)
    freq, spfft,spcost,interbis,internew = int2spec(intermed,xpos_bwd,n2)
    spmed_bwd.append(np.asarray(spfft))
    spcmed_bwd.append(np.asarray(spcost))
```

```
spmed_fwd = np.asarray(spmed_fwd)
spcmed_fwd = np.asarray(spcmed_fwd)
spmean_fwd = np.asarray(spmean_fwd)
spcmean_fwd = np.asarray(spcmean_fwd)


# Getting median spectra
fr_fwd   = freq_fwd[0]
spcmed_fwd = np.asarray(spcmed_fwd)
h0=fits.PrimaryHDU()
h1 = fits.ImageHDU(data=fr_fwd,name='FREQ')
h2 = fits.ImageHDU(data=spcmed_fwd,name='TRANS')
h3 = fits.ImageHDU(data=spmed_fwd,name='TRANS_FFT')
hdu = fits.HDUList([h0,h1,h2,h3])
hdu.writeto(dir_plot+'spcmed_fwd.fits',overwrite='True')

fr_bwd   = freq_bwd[0]
spcmed_bwd = np.asarray(spcmed_bwd)
h0=fits.PrimaryHDU()
h1 = fits.ImageHDU(data=fr_bwd,name='FREQ')
h2 = fits.ImageHDU(data=spcmed_bwd,name='TRANS')
h3 = fits.ImageHDU(data=spmed_bwd,name='TRANS_FFT')
hdu = fits.HDUList([h0,h1,h2,h3])
hdu.writeto(dir_plot+'spcmed_bwd.fits',overwrite='True')


fr_bwd = freq_bwd[0]
spcmed_bwd = np.asarray(spcmed_bwd)
h0=fits.PrimaryHDU()
h1 = fits.ImageHDU(data=fr_bwd,name='FREQ')
h2 = fits.ImageHDU(data=spcmed_bwd,name='TRANS')
hdu = fits.HDUList([h0,h1,h2])
hdu.writeto(dir_plot+'spcmed_bwd.fits',overwrite='True')







# Plotting spectra

fig3,(ax1,ax2,ax3) = plt.subplots(ncols=3,nrows=1,figsize=(15,6))

for index in w1:
    ax1.plot(freq_fwd[index],np.abs(spcmed_fwd[index])/np.max(np.abs(spcmed_fwd[index])))

for index in w2:
    ax2.plot(freq_fwd[index],np.abs(spcmed_fwd[index])/np.max(np.abs(spcmed_fwd[index])))
```

```
for index in w3:
    ax3.plot(freq_fwd[index],np.abs(spcmed_fwd[index])/np.max(np.abs(spcmed_fwd[index])))

fig3.savefig(dir_plot+'A123_spec_FWD_alldet.jpeg')
plt.close(fig3)


fig4,(ax1,ax2,ax3) = plt.subplots(ncols=3,nrows=1,figsize=(15,6))

for index in w1:
    ax1.plot(freq_fwd[index],spcmed_fwd[index])

for index in w2:
    ax2.plot(freq_fwd[index],spcmed_fwd[index])

for index in w3:
    ax3.plot(freq_fwd[index],spcmed_fwd[index])

fig4.savefig(dir_plot+'A123_spec_FWD_alldet_unnorm.jpeg')
plt.close(fig4)



for idet in range(ntotdet): spcmed_fwd[idet,:] /= np.max(spcmed_fwd[idet,:])
spec_a1_fwd = np.median(spcmed_fwd[w1,:],axis=0)
spec_a2_fwd = np.median(spcmed_fwd[w2,:],axis=0)
spec_a3_fwd = np.median(spcmed_fwd[w3,:],axis=0)

spec_a1_fwd /= spec_a1_fwd.max()
spec_a2_fwd /= spec_a2_fwd.max()
spec_a3_fwd /= spec_a3_fwd.max()

res_fwd ={'FR':fr_fwd, 'TA1':spec_a1_fwd,'TA2':spec_a2_fwd,'TA3':spec_a3_fwd}
hfits.dict2fits(res_fwd,dir_plot+'spcmed_fwd_perarray.fits')

for idet in range(ntotdet): spcmed_bwd[idet,:] /= np.max(spcmed_bwd[idet,:])

spec_a1_bwd = np.median(spcmed_bwd[w1,:],axis=0)
spec_a2_bwd = np.median(spcmed_bwd[w2,:],axis=0)
spec_a3_bwd = np.median(spcmed_bwd[w3,:],axis=0)

spec_a1_bwd /= spec_a1_bwd.max()
spec_a2_bwd /= spec_a2_bwd.max()
spec_a3_bwd /= spec_a3_bwd.max()

res_bwd ={'FR':fr_bwd, 'TA1':spec_a1_bwd,'TA2':spec_a2_bwd,'TA3':spec_a3_bwd}
hfits.dict2fits(res_bwd,dir_plot+'spcmed_bwd_perarray.fits')

fig1,(ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(17,8),sharey=True,sharex=True)
fig1.subplots_adjust(wspace=0)
```

```
ax1.plot(fr_fwd, spec_a1_fwd,label='FWD')
ax1.plot(fr_bwd, spec_a1_bwd,label='BWD')
ax1.legend()
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.set_title('A1')
ax1.set_xlim([0,500])
ax1.set_ylim([-0.2,1.2])

ax3.plot(fr_fwd, spec_a3_fwd,label='FWD')
ax3.plot(fr_bwd, spec_a3_bwd,label='BWD')
ax3.legend()
ax3.set_xlabel('Frequency [GHz]')
#ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')

ax2.plot(fr_fwd, spec_a2_fwd,label='FWD')
ax2.plot(fr_bwd, spec_a2_bwd,label='BWD')
ax2.legend()
ax2.set_xlabel('Frequency [GHz]')
#ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
fig1.savefig(dir_plot+'A123_medspec_fullrange.jpeg')
plt.close(fig1)

fig2,ax = plt.subplots(ncols=1,nrows=1,figsize=(10,8))

ax.plot(fr_fwd, spec_a1_fwd,color='b',label='A1 F')
ax.plot(fr_fwd, spec_a3_fwd,color='cyan',label='A3 F')
ax.plot(fr_fwd, spec_a2_fwd,color='r',label='A2 F')

ax.plot(fr_bwd, spec_a1_bwd,'bo', label='A1 B')
ax.plot(fr_bwd, spec_a3_bwd, 'co', label='A3 B')
ax.plot(fr_bwd, spec_a2_bwd,'ro',label='A2 B')

ax.set_xlabel('Frequency [GHz]')
ax.set_ylabel('Normalized transmission')
ax.set_xlim([90,340])
ax.set_ylim([-0.2,1.2])
ax.legend()
ax.grid()
fig2.savefig(dir_plot+'A123_medspec_zoom.jpeg')
plt.close(fig2)

# Compute KID histograms per array

mnbins = 30
fr = freq_fwd[0].ravel()
pint = np.linspace(0,1.0,mnbins)
```

```
xi,yi = np.meshgrid(fr,pint)

spc_fwd_a1_h = kid_hist(spcmed_fwd, detsel=w1, nbins = 30,crange=(0.0,1.0))
spc_fwd_a3_h = kid_hist(spcmed_fwd, detsel=w3, nbins = 30,crange=(0.0,1.0))
spc_fwd_a2_h = kid_hist(spcmed_fwd, detsel=w2, nbins = 30,crange=(0.0,1.0))


fig4, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,5))

im1 = ax1.pcolormesh(xi,yi,spc_fwd_a1_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w1)))
cb1 = fig4.colorbar(im1,ax=ax1)
ax1.plot(fr,spec_a1_fwd,'ro',label='MED FWD')
ax1.set_xlim([70,320])
ax1.set_ylim([0.0,1.05])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.legend(loc=2)
ax1.set_title('A1')
ax1.vlines(260.0,0,1.05,linewidth=2,linestyle='dashed')

im3 = ax3.pcolormesh(xi,yi,spc_fwd_a3_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w3)))
cb3 = fig4.colorbar(im3,ax=ax3)
ax3.plot(fr,spec_a3_fwd,'ro',label='MED')
ax3.set_xlim([70,320])
ax3.set_ylim([0.0,1.05])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.05,linewidth=2,linestyle='dashed')


im2 = ax2.pcolormesh(xi,yi,spc_fwd_a2_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w2)))
cb2 = fig4.colorbar(im2,ax=ax2)
ax2.plot(fr,spec_a2_fwd,'ro',label='MED')
ax2.set_xlim([70,320])
ax2.set_ylim([0.0,1.05])
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
ax2.vlines(150.0,0,1.05,linewidth=2,linestyle='dashed')
cb2.set_label('Number of KIDs')
fig4.savefig(dir_plot+'A123_spec_FWD_hisdet.jpeg')
plt.close(fig4)

spc_bwd_a1_h = kid_hist(spcmed_bwd, detsel=w1, nbins = 30,crange=(0.0,1.0))
spc_bwd_a3_h = kid_hist(spcmed_bwd, detsel=w3, nbins = 30,crange=(0.0,1.0))
spc_bwd_a2_h = kid_hist(spcmed_bwd, detsel=w2, nbins = 30,crange=(0.0,1.0))


fig5, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,5))
```

```
    im1 = ax1.pcolormesh(xi,yi,spc_bwd_a1_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w1)))
    cb1 = fig5.colorbar(im1,ax=ax1)
    ax1.plot(fr,spec_a1_bwd,'ro',label='MED BWD')
    ax1.set_xlim([70,320])
    ax1.set_ylim([0.0,1.05])
    ax1.set_xlabel('Frequency [GHz]')
    ax1.set_ylabel('Normalized transmission')
    ax1.legend(loc=2)
    ax1.set_title('A1')
    ax1.vlines(260.0,0,1.05,linewidth=2,linestyle='dashed')

    im3 = ax3.pcolormesh(xi,yi,spc_bwd_a3_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w3)))
    cb3 = fig5.colorbar(im3,ax=ax3)
    ax3.plot(fr,spec_a3_bwd,'ro',label='MED')
    ax3.set_xlim([70,320])
    ax3.set_ylim([0.0,1.05])
    ax3.set_xlabel('Frequency [GHz]')
    ax3.set_ylabel('Normalized transmission')
    ax3.set_title('A3')
    ax3.vlines(260.0,0,1.05,linewidth=2,linestyle='dashed')


    im2 = ax2.pcolormesh(xi,yi,spc_bwd_a2_h+1e-6,cmap='Blues',norm=LogNorm(vmin=1, vmax=len(w2)))
    cb2 = fig5.colorbar(im2,ax=ax2)
    ax2.plot(fr,spec_a2_bwd,'ro',label='MED')
    ax2.set_xlim([70,320])
    ax2.set_ylim([0.0,1.05])
    ax2.set_xlabel('Frequency [GHz]')
    ax2.set_ylabel('Normalized transmission')
    ax2.set_title('A2')
    ax2.vlines(150.0,0,1.05,linewidth=2,linestyle='dashed')
    cb2.set_label('Number of KIDs')
    fig5.savefig(dir_plot+'A123_spec_BWD_hisdet.jpeg')


# PLOT ALL RESULTS


fig6, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

ax1.set_xlim([150,320])
ax1.set_ylim([0.0,1.0])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.legend(loc=2)
ax1.set_title('A1')
ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')
```

```
ax3.set_xlim([150,320])
ax3.set_ylim([0.0,1.0])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax2.set_xlim([80,260])
ax2.set_ylim([0.0,1.0])
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')

wpos = np.argsort(np.array(plate_pos).astype(np.long))
for index in wpos:
    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    mdict= hfits.fits2dict(dir_plot+'spcmed_fwd_perarray.fits')

    ax1.plot(mdict['FR'],mdict['TA1'],label=plate_pos[index])
    ax3.plot(mdict['FR'],mdict['TA3'],label=plate_pos[index])
    ax2.plot(mdict['FR'],mdict['TA2'],label=plate_pos[index])
    ax1.legend(loc=2)

fig6.savefig(dir_plot_base+'A123_spec_FWD_differentposition.jpeg')
plt.close(fig6)

fig7, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

ax1.set_xlim([150,320])
ax1.set_ylim([0.0,1.0])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.legend(loc=2)
ax1.set_title('A1')
ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax3.set_xlim([150,320])
ax3.set_ylim([0.0,1.0])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax2.set_xlim([80,260])
ax2.set_ylim([0.0,1.0])
```

```
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')

wpos = np.argsort(np.array(plate_pos).astype(np.long))
for index in wpos:
    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    mdict= hfits.fits2dict(dir_plot+'spcmed_bwd_perarray.fits')

    ax1.plot(mdict['FR'],mdict['TA1'],label=plate_pos[index])
    ax3.plot(mdict['FR'],mdict['TA3'],label=plate_pos[index])
    ax2.plot(mdict['FR'],mdict['TA2'],label=plate_pos[index])
    ax1.legend(loc=2)

fig7.savefig(dir_plot_base+'A123_spec_BWD_differentposition.jpeg')
plt.close(fig7)

# Plot data at different angles


fig8, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

ax1.set_xlim([150,320])
#ax1.set_ylim([0.0,1.0])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.set_title('A1')
ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax3.set_xlim([150,320])
#ax3.set_ylim([0.0,1.0])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax2.set_xlim([80,260])
#ax2.set_ylim([0.0,1.0])
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')

wpos = np.argsort(np.array(plate_pos).astype(np.long))

for index in wpos:
```

```
    mfile = label_arr[index]
    print mfile
    dir_plot = dir_plot_base+ mfile+'/'
    hdu= fits.open(dir_plot+'spcmed_fwd.fits')

    freq = hdu[1].data
    spcost = hdu[2].data
    spfft = hdu[3].data
    plt.plot(freq,spcost[w1[50],:])

    ax1.plot(freq,spcost[w1[50],:],label=plate_pos[index])
    ax2.plot(freq,spcost[w2[50],:],label=plate_pos[index])
    ax3.plot(freq,spcost[w3[50],:],label=plate_pos[index])


ax1.legend(loc=2,title='PIMP angle [degrees]:')

#ax1.set_ylim([-20000,200000])
#ax3.set_ylim([-20000,200000])
#ax2.set_ylim([-10000,70000])

fig8.savefig(dir_plot_base+'A123_spec_FWD_differentposition_unorm.jpeg')
plt.close(fig8)



fig9, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

ax1.set_xlim([150,320])
#ax1.set_ylim([0.0,1.0])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.set_title('A1')
ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax3.set_xlim([150,320])
#ax3.set_ylim([0.0,1.0])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax2.set_xlim([80,260])
#ax2.set_ylim([0.0,1.0])
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
ax2.set_title('A2')
```

```
ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')


wpos = np.argsort(np.array(plate_pos).astype(np.long))

for index in wpos:
    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    hdu= fits.open(dir_plot+'spcmed_fwd.fits')
    freq = hdu[1].data
    spcost = hdu[2].data
    spfft = hdu[3].data
    ax1.plot(freq,spfft[w1[100],:],label=plate_pos[index])
    ax2.plot(freq,spfft[w2[100],:],label=plate_pos[index])
    ax3.plot(freq,spfft[w3[100],:],label=plate_pos[index])


ax1.legend(loc=2,title='PIMP angle [degrees]:')

ax1.set_ylim([-50,300])
ax3.set_ylim([-50,300])
ax2.set_ylim([50,300])

fig9.savefig(dir_plot_base+'A123_specfft_FWD_differentposition_unorm.jpeg')
plt.close(fig9)


# Comparison to LAB spectra
fig10, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

ax1.set_xlim([150,320])
ax1.set_ylim([0.0,1.1])
ax1.set_xlabel('Frequency [GHz]')
ax1.set_ylabel('Normalized transmission')
ax1.set_title('A1')
ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


ax3.set_xlim([150,320])
ax3.set_ylim([0.0,1.1])
ax3.set_xlabel('Frequency [GHz]')
ax3.set_ylabel('Normalized transmission')
ax3.set_title('A3')
ax3.vlines(260.0,0,1.1,linewidth=2,linestyle='dashed')


ax2.set_xlim([80,260])
ax2.set_ylim([0.0,1.1])
ax2.set_xlabel('Frequency [GHz]')
ax2.set_ylabel('Normalized transmission')
```

```
ax2.set_title('A2')
ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')
wpos = np.argsort(np.array(plate_pos).astype(np.long))

#wpos = np.asarray([0, 7, 4, 5, 2,6,8])
sp1arr = []
for index in wpos:
    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    hdu= fits.open(dir_plot+'spcmed_fwd_perarray.fits')
    freq = hdu[1].data['FR']
    sp1 =  hdu[1].data['TA1']
    sp2 =  hdu[1].data['TA2']
    sp3 =  hdu[1].data['TA3']
    print index, mfile, sp1.shape
    sp1arr.append(sp1)
    ax1.plot(freq,sp1,label=plate_pos[index])
    ax2.plot(freq,sp2,label=plate_pos[index])
    ax3.plot(freq,sp3,label=plate_pos[index])


filebpnew = "/home/macias/NIKA/Processing/Pipeline/Calibration/BP/Transmission_2017_Jan_NIKA2_v1.fits"
hdunew = fits.open(filebpnew)

bpa1n = hdunew[1].data
bpa3n = hdunew[2].data
bpa2n = hdunew[3].data
fr1 =  bpa1n['FREQ'][1:]
fr3 =  bpa3n['FREQ'][1:]
fr2 =  bpa2n['FREQ'][1:]
bp1m = bpa1n['NIKATRANS'][1:]/np.max(bpa1n['NIKATRANS'][1:])
bp3m = bpa3n['NIKATRANS'][1:]/np.max(bpa3n['NIKATRANS'][1:])
bp2m = bpa2n['NIKATRANS'][1:]/np.max(bpa2n['NIKATRANS'][1:])

ax1.plot(fr1,bp1m,'ro',label='LAB')
ax2.plot(fr2,bp2m,'ro',label='LAB')
ax3.plot(fr3,bp3m,'ro',label='LAB')

ax1.legend(loc=2,title='PIMP angle [degrees]:')


fig10.savefig(dir_plot_base+'A123_allpos_vs_lab_spectra.jpeg')
plt.close(fig10)




# Plot focal plane figures
dirfig =  dir_plot_base
```

```
nasx = kidparref['NAS_X']
nasy = kidparref['NAS_Y']

var1 = kidparref['CORR2CM']
var2 = kidparref['CALIB_FIX_FWHM']
fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = var2[w1ok[ww1]]/var1[w1ok[ww1]]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = var2[w3ok[ww3]]/var1[w3ok[ww3]]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = var2[w2ok[ww2]]/var1[w2ok[ww2]]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.6,vmax=1.5)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'kid_gradient.jpeg')
plt.close(fig)




# 30 degrees.
ipos = 4
# 60 degrees
```

```
ipos=5
mfile = label_arr[ipos]
dir_plot = dir_plot_base+ mfile+'/'
hdu= fits.open(dir_plot+'spcmed_fwd.fits')

freq = hdu[1].data
spcost = hdu[2].data
spfft = hdu[3].data

ndet,npts = spcost.shape

maxfr_arr = []
medfr_arr = []
fwhm_arr  = []
bp_arr    = []
tmax_arr  = []
fminc_arr = []
fmaxc_arr = []

for idet in range(ndet):
    spc = spcost[idet,:]
    tmax_arr.append(np.max(spc))
    spc /= max(spc)
    maxfreq,medfreq,fwhm,bp,fminc,fmaxc = spec_prop(freq,spc)
    maxfr_arr.append(maxfreq)
    medfr_arr.append(medfreq)
    fminc_arr.append(fminc)
    fmaxc_arr.append(fmaxc)
    fwhm_arr.append(fwhm)
    bp_arr.append(bp)



fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(maxfr_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.8,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
```

```
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(maxfr_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.8,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(maxfr_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.8,vmax=1.2)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+np.str(plate_pos[ipos]) +'_maxfreq.jpeg')
plt.close(fig)

fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(medfr_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(medfr_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
```

```
ax2.grid()
c2 = np.asarray(medfr_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.6,vmax=1.5)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+np.str(plate_pos[ipos])+'_barycenterfreq.jpeg')
plt.close(fig)


fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(fwhm_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(fwhm_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(fwhm_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.6,vmax=1.5)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+np.str(plate_pos[ipos])+'_fwhm.jpeg')
plt.close(fig)


fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))
```

```
ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(bp_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(bp_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(bp_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.6,vmax=1.5)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+plate_pos[ipos]+'_bp.jpeg')
plt.close(fig)

fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(fmaxc_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb1 = fig.colorbar(sct1,ax = ax1)
```

```
ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(fmaxc_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.6,
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(fmaxc_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.6,vmax=1.5)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+np.str(plate_pos[ipos])+'_fmaxc.jpeg')
plt.close(fig)


fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(fminc_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.9,
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(fminc_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0),vmin=0.9,
cb3 = fig.colorbar(sct3,ax = ax3)
```

```
ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(fminc_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60,vmin=0.9,vmax=1.1)
cb2 = fig.colorbar(sct2,ax = ax2)

fig.savefig(dirfig+'pos_'+np.str(plate_pos[ipos])+'_fminc.jpeg')
plt.close(fig)


fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

ax1.set_xlim([-250,250])
ax1.set_ylim([-250,250])
ax1.set_title('Array'+np.str(1))
ax1.set_xlabel('Nasmyth Offset x [arcsec]')
ax1.set_ylabel('Nasmyth Offset y [arcsec]')
ax1.grid()
c1 = np.asarray(tmax_arr)[w1]
c1 = c1/np.median(c1)
sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0)) #,vmin=0
cb1 = fig.colorbar(sct1,ax = ax1)

ax3.set_xlim([-250,250])
ax3.set_ylim([-250,250])
ax3.set_title('Array'+np.str(3))
ax3.set_xlabel('Nasmyth Offset x [arcsec]')
ax3.set_ylabel('Nasmyth Offset y [arcsec]')
ax3.grid()
c3 = np.asarray(tmax_arr)[w3]
c3 = c3/np.median(c3)
sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0)) #,vmin=0
cb3 = fig.colorbar(sct3,ax = ax3)

ax2.set_xlim([-250,250])
ax2.set_ylim([-250,250])
ax2.set_title('Array'+np.str(2))
ax2.set_xlabel('Nasmyth Offset x [arcsec]')
ax2.set_ylabel('Nasmyth Offset y [arcsec]')
ax2.grid()
c2 = np.asarray(tmax_arr)[w2]
c2 = c2/np.median(c2)
sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60) #,vmin=0.9,vmax=1.1)
cb2 = fig.colorbar(sct2,ax = ax2)
```

```
fig.savefig(dirfig+'pos_'+plate_pos[ipos]+'_ampmax.jpeg')
plt.close(fig)


for index in range(len(label_arr)):

    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    hdu= fits.open(dir_plot+'spcmed_fwd.fits')

    freq = hdu[1].data
    spcost = hdu[2].data
    spfft = hdu[3].data

    tmax_arr = np.max(spcost,axis=1)

    ndet,npts = spcost.shape


    fig, (ax1,ax3,ax2) = plt.subplots(ncols=3,nrows=1,figsize=(20,5))

    ax1.set_xlim([-250,250])
    ax1.set_ylim([-250,250])
    ax1.set_title('Array'+np.str(1))
    ax1.set_xlabel('Nasmyth Offset x [arcsec]')
    ax1.set_ylabel('Nasmyth Offset y [arcsec]')
    ax1.grid()
    c1 = np.asarray(tmax_arr)[w1]
    c1 = c1/np.median(c1)
    sct1=ax1.scatter(nasx[w1ok[ww1]], nasy[w1ok[ww1]], c=c1,edgecolors=[1,1,1],s=60,marker=(5,0)) #,vm
    cb1 = fig.colorbar(sct1,ax = ax1)

    ax3.set_xlim([-250,250])
    ax3.set_ylim([-250,250])
    ax3.set_title('Array'+np.str(3))
    ax3.set_xlabel('Nasmyth Offset x [arcsec]')
    ax3.set_ylabel('Nasmyth Offset y [arcsec]')
    ax3.grid()
    c3 = np.asarray(tmax_arr)[w3]
    c3 = c3/np.median(c3)
    sct3=ax3.scatter(nasx[w3ok[ww3]], nasy[w3ok[ww3]], c=c3,edgecolors=[1,1,1],s=60,marker=(5,0)) #,vm
    cb3 = fig.colorbar(sct3,ax = ax3)

    ax2.set_xlim([-250,250])
    ax2.set_ylim([-250,250])
    ax2.set_title('Array'+np.str(2))
    ax2.set_xlabel('Nasmyth Offset x [arcsec]')
    ax2.set_ylabel('Nasmyth Offset y [arcsec]')
    ax2.grid()
    c2 = np.asarray(tmax_arr)[w2]
```

```
    c2 = c2/np.median(c2)
    sct2=ax2.scatter(nasx[w2ok[ww2]], nasy[w2ok[ww2]], c=c2,edgecolors=[1,1,1],s=60) #,vmin=0.9,vmax=1
    cb2 = fig.colorbar(sct2,ax = ax2)

    fig.savefig(dirfig+'pos_'+plate_pos[index]+'_ampmax.jpeg')
    plt.close(fig)


# ----------------------------------------------------------------------

fig12, (ax1,ax3,ax2) = plt.subplots(ncols=3, nrows=1,figsize = (20,8))

#ax1.set_xlim([-10,120])
#ax1.set_ylim([0.0,1.0])
ax1.set_xlabel('PIMP position angle [degrees]')
ax1.set_ylabel('Maximum transmission')
ax1.set_title('A1')
#ax1.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


#ax3.set_xlim([-10,120])
#ax3.set_ylim([0.0,1.0])
ax3.set_xlabel('PIMP position angle [degrees]')
ax3.set_ylabel('Maximum transmission')
ax3.set_title('A3')
#ax3.vlines(260.0,0,1.0,linewidth=2,linestyle='dashed')


#ax2.set_xlim([-10,120])
#ax2.set_ylim([0.0,1.0])
ax2.set_xlabel('PIMP position angle [degrees]')
ax2.set_ylabel('Maximum transmission')
ax2.set_title('A2')
#ax2.vlines(150.0,0,1.0,linewidth=2,linestyle='dashed')

posd = np.array(plate_pos).astype(np.long)
wpos = np.argsort(posd)

#wpos = np.asarray([0, 7, 4, 5, 2,6,8])

amp1 = np.zeros(len(wpos))
amp2 = np.zeros(len(wpos))
amp3 = np.zeros(len(wpos))

fwhm1 = np.zeros(len(wpos))
fwhm2 = np.zeros(len(wpos))
fwhm3 = np.zeros(len(wpos))


for index in wpos:
```

```
    mfile = label_arr[index]
    dir_plot = dir_plot_base+ mfile+'/'
    hdu= fits.open(dir_plot+'spcmed_fwd.fits')

    freq   = hdu[1].data
    spcost = hdu[2].data
    spfft  = hdu[3].data

    pos1  = np.where(spcost[w1[100],:]/np.max(spcost[w1[100],:]) >= 0.5)
    fwhm1[index] = np.max(freq[pos1]) -  np.min(freq[pos1])
    pos2  = np.where(spcost[w2[100],:]/np.max(spcost[w2[100],:]) >= 0.5)
    fwhm2[index] = np.max(freq[pos2]) -  np.min(freq[pos2])
    pos3  = np.where(spcost[w3[100],:]/np.max(spcost[w3[100],:]) >= 0.5)
    fwhm3[index] = np.max(freq[pos3]) -  np.min(freq[pos3])

    amp1[index] = np.max(spcost[w1[100],:])
    amp2[index] = np.max(spcost[w2[100],:])
    amp3[index] = np.max(spcost[w3[100],:])

#    amp1[index] = np.max(np.median(spcost[w1,:],axis=0))
#    amp2[index] = np.max(np.median(spcost[w2,:],axis=0))
#    amp3[index] = np.max(np.median(spcost[w3,:],axis=0))

ax1.plot(posd[wpos],amp1,'ro')
ax2.plot(posd[wpos],amp2,'ro')
ax3.plot(posd[wpos],amp3,'ro')

ax1.plot(posd[wpos],amp1,'b')
ax2.plot(posd[wpos],amp2,'b')
ax3.plot(posd[wpos],amp3,'b')

#ax1.plot(posd[wpos],fwhm1,'ro')
#ax2.plot(posd[wpos],fwhm2,'ro')
#ax3.plot(posd[wpos],fwhm3,'ro')


#ax1.set_ylim([-20000,140000])
#ax3.set_ylim([-20000,140000])
#ax2.set_ylim([-10000,70000])

fig12.savefig(dir_plot_base+'A123_spec_FWD_differentposition_unorm.jpeg')
plt.close(fig12)
```