# Checking the phase stability of the DBBC at the telescope

P. de Vicente

Informe Técnico IT-CDT/OAN 2014-4

# Revision history

| Version | Date | Author | Updates |
|---|---|---|---|
| 1.0 | 02-05-2014 | P. de Vicente | |

# Contents

# 1 Introduction

The usual procedure to validate a DBBC is to perform VBI observations in parallel with the legacy VLBI recorder and get information on the correlated scans between both systems from the correlator. In case of problems this is a lengthy procedure since it takes a lot of time between the observation and the results unless DifX is installed at the station and one can correlate at the site.

In this short report we describe a method to measure the phase stability of the DBBC plus the Mark5B when using the former in DDC mode. The method requires a phase cal system, usually available in many VLBI stations. We developed this method when we discovered that the DBBC at the 40m antenna was not behaving as expected and wanted to make hardware tests to narrow the cause of the problem. This method speeds up the debugging process since it allows the personnel at the stations to do hardware changes and see the results immediately after.

# 2 Phase unstability

We have done several parallel observations along 2014. Three of them with geodetic observations and a similar number within session 1 of the EVN (NMEs). The first ones were not successful.

R1631 was a geodetic observation performed in March 2014 in parallel with the VLBA5 backend. Fig. 1 shows the "fourfit" plot from the correlator for one scan on day 90 at 19:15 between the DBBC and the VLBA5 at Yebes. The plot clearly demonstrates phase drifts breaks in the phase for every X channel between both backends as a function of frequency. Other scans, see Figs. 2 and 3, for X and S band respectively, showed a better behaviour at different moments of the experiment.

According to the correlator the DBBC was not working properly and phases showed cyclic jumps of 400 ps approximately after several hours.

# 3 Measuring phase versus time at the telescope

In order to measure the phase stability of the DBBC channels at the telescope we used the phase cal system. The phase cal injects tones every 1 MHz starting in 10 KHz in the IF chain. It is possible to extract the amplitude and phase of the tones recording the data on a disk and later using *bpcal* (see Smythe 2009) on the recorded data. *bpcal* is a utility program that extracts the amplitude and phase of one tone from a subset of data. It only works with 2 bit Mark5B data recorded with the mask 0xffffffff. It's usage is very simple:

```
bpcal <filename_mk5b_data> <tone_frequency KHZ> [<#frames]>
```

where the tone frequency is given in KHz and the number of frames is optional.

We use the S/X receivers, tune the whole system with a 2 bit geodetic setup and record data continuously for 7 minutes. Then we extract 1 second of data every 1 minute and store it in a file in the Mark5B internal disk. *bpcal* is run on these files. Usualy we select tone 2010 KHz,


Figure 1: *Fourfit plot at X band between the DBBC and the VLBA5 terinals for Yebes. Scan at 19:15*

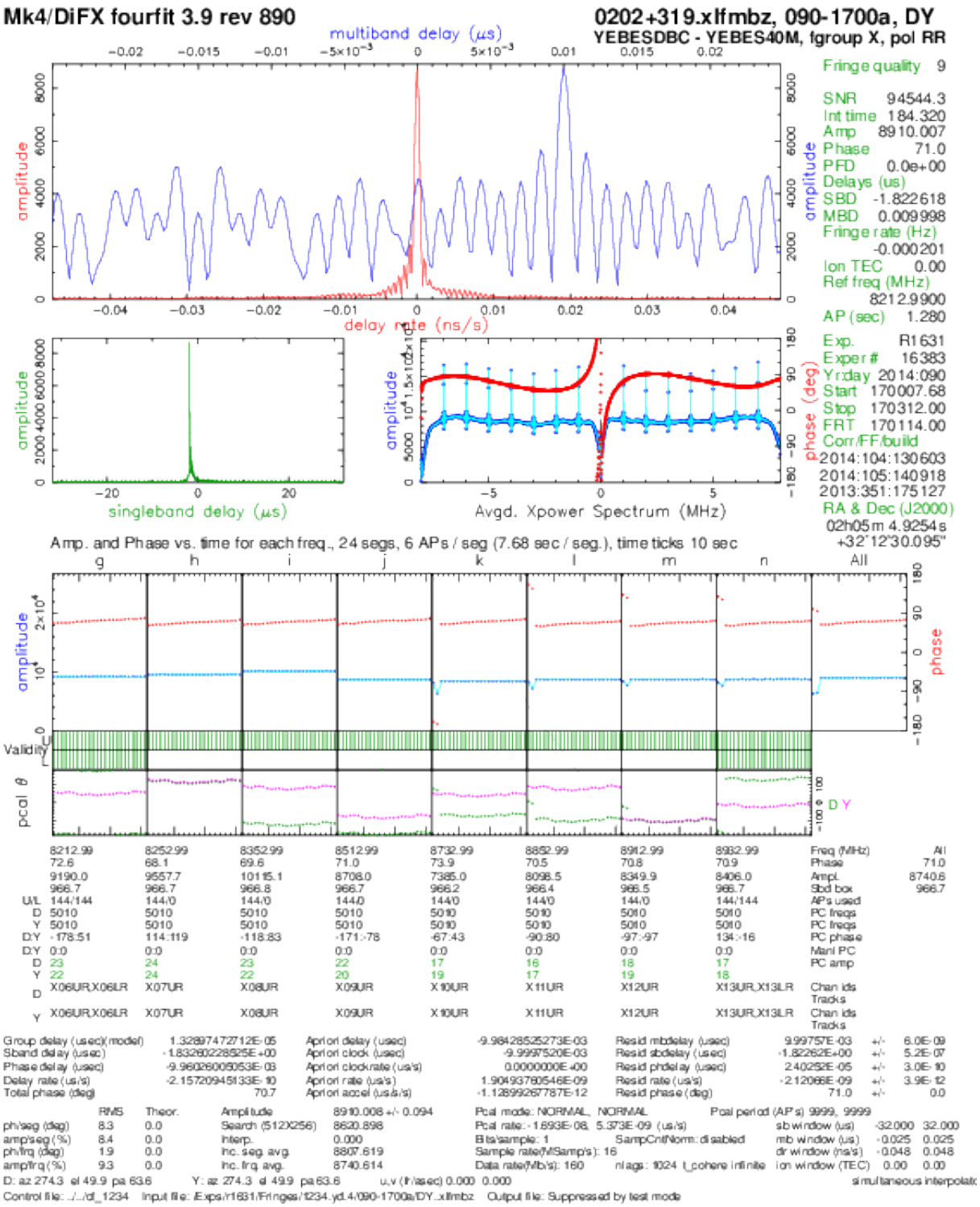


Figure 2: *Fourfit plot at X band between the DBBC and the VLBA5 terminals for Yebes. Scan at 17:00*

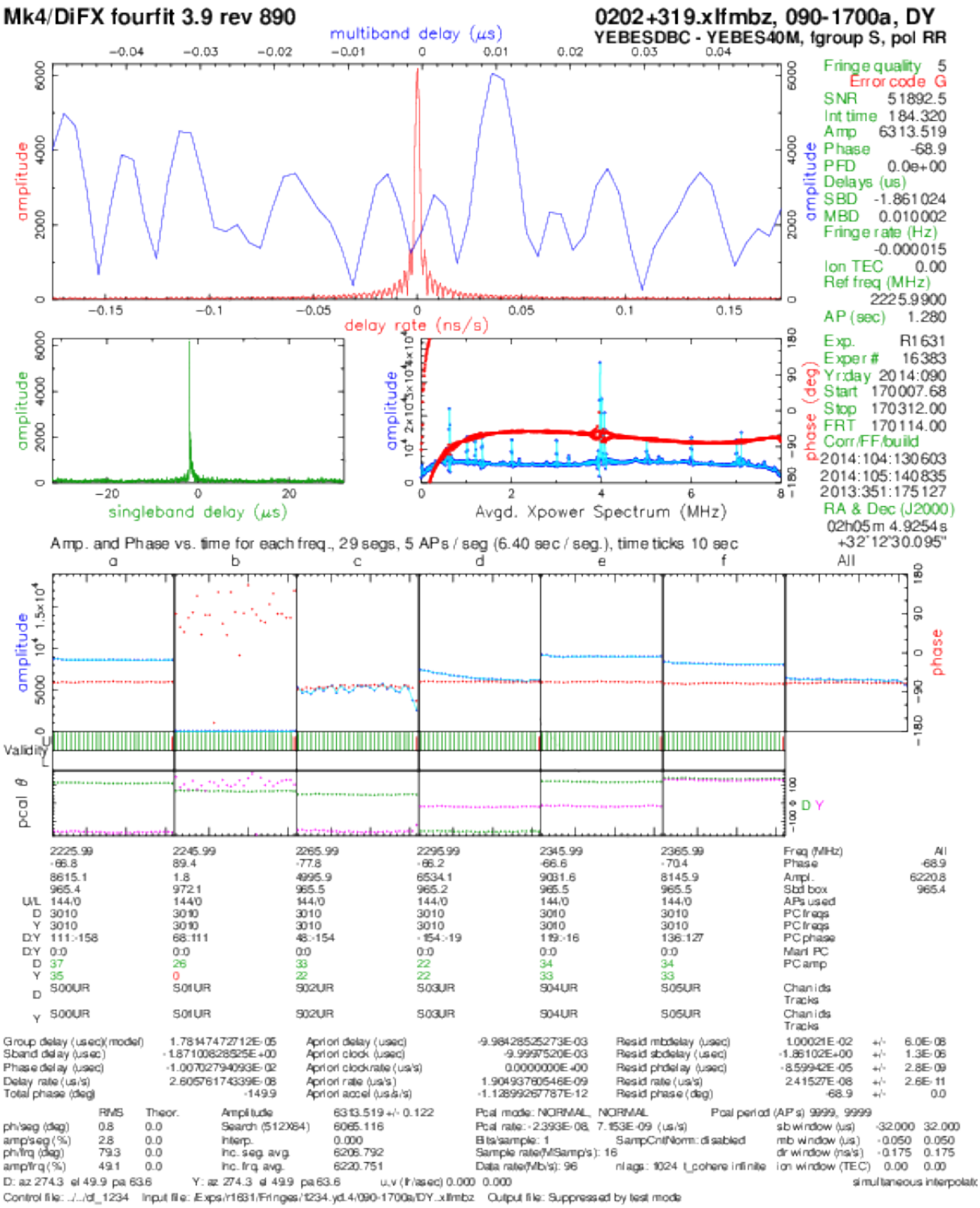


Figure 3: *Fourfit plot at S band between the DBBC and the VLBA5 terminals for Yebes. Scan at 17:00*

but any other tone within the band of each channel would also work. The program displays the phase and amplitude of the tone for the 16 channels on the screen. We collect these values for different times and we finally represent them as a function of time. In a working system the phase should stay stable over long times. Phase cal phase changes are easily spotted, even if the time sampling (1 sample per minute) is so coarse.

It is important to notice that in a geodetic setup only the upper side bands are recorded. Channels 0 to 7 correspond to the upper side band (USB) of BBC channels 1 to 8, channels 8 and 9 come from the lower side band BBC channels 1 and 8, and channels 10 to 15 correspond to USB of BBC channels 9 to 14. There will be no amplitude and phase signal in channels 8 and 9 since the local oscillators of individual BBC channels are usually set to a frequency multiple of 1 MHz plus .99 MHz to obtain tones of 10 KHz above multiples of 1 MHz after the mix.

This method is not restricted to a geodetic setup, but we believe it is the best option since the geodetic setup allows to sample 14 channels, which requires 3 or 4 IFs in a DBBC and hence debugs the behaviour of the matching A/D boards (3 or 4) and COREs (4) at one single measurement.

Below we show a tipical output from *bpcal* with a note at the right side of each line that specifies where the signal comes from.

```
integration time  0.062 sec

ch  amp   phase(dg)
 0     37    161.3 USB1
 1     38    132.8 USB2
 2     35   -176.5 USB3
 3     34    -64.3 USB4
 4     28   -106.4 USB5
 5     30     82.8 USB6
 6     32     11.3 USB7
 7     29     91.4 USB8
 8      1   -166.4 LSB1
 9      1    147.7 LSB8
10     61    166.4 USB9
11     54    166.4 USB10
12     54   -169.1 USB11
13     35     68.9 USB12
14     29   -172.4 USB13
15     29    -85.3 USB14
```

In the example above we see that the amplitude for channels 8 and 9 is 1, and differs by more than a factor 30 from other channels, as we have explained above. This number is noise from the measurement.

## 3.1 Phase stability during R1631

Some days after R1631 we extracted a small interval of data using the previous method. The results are shown in Fig. 4. BBC channels 0-3 and 10 to 15 behave correctly since the phase changes smoothly. Channels 4 to 7 show jumps at a given time. This jumps affect mainly the extended X band in the R1631 setup.

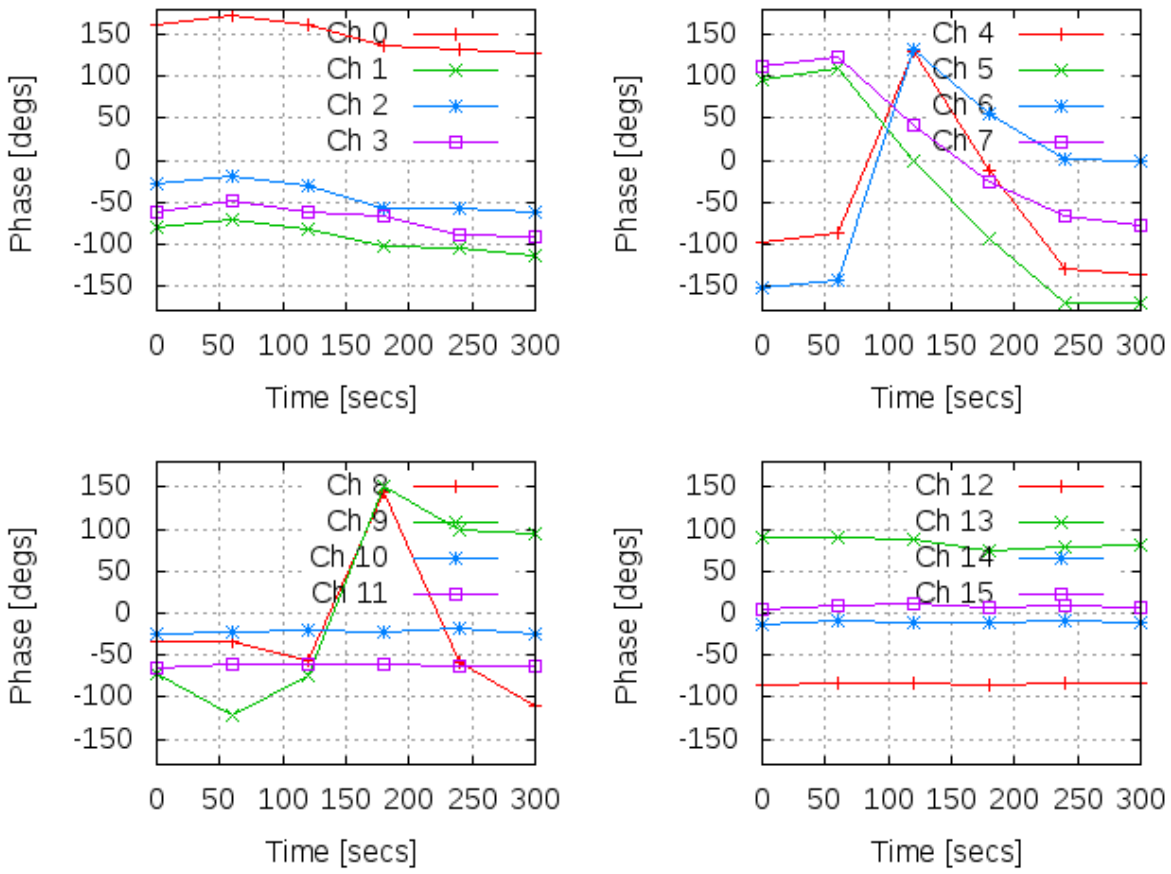


Figure 4: *Dependence of the phase from the phase cal tones for 16 channels as a function of time. Channels 8 and 9 do not contain valid data. Data obtained some days after R1631. Three IFs and A/D boards plus 4 COREs were used.*

## 3.2 Phase stability previous to C1414

The DBBC was sent to Max Planck Institut für Radioastronomie in May for a third check and some hardware corrections were applied. It was found that the last boards of the bus did not comply with the nominal voltage. This was probably the most important correction applied. Afterwards the DBBC was back in Yebes and a phase stability test was run. The result is displayed in Fig. 5. In this case all channels show a flat response as a function of time. Channel 0 apparently shows a jump, but it comes from the ambiguity between -180 and 180 degrees.

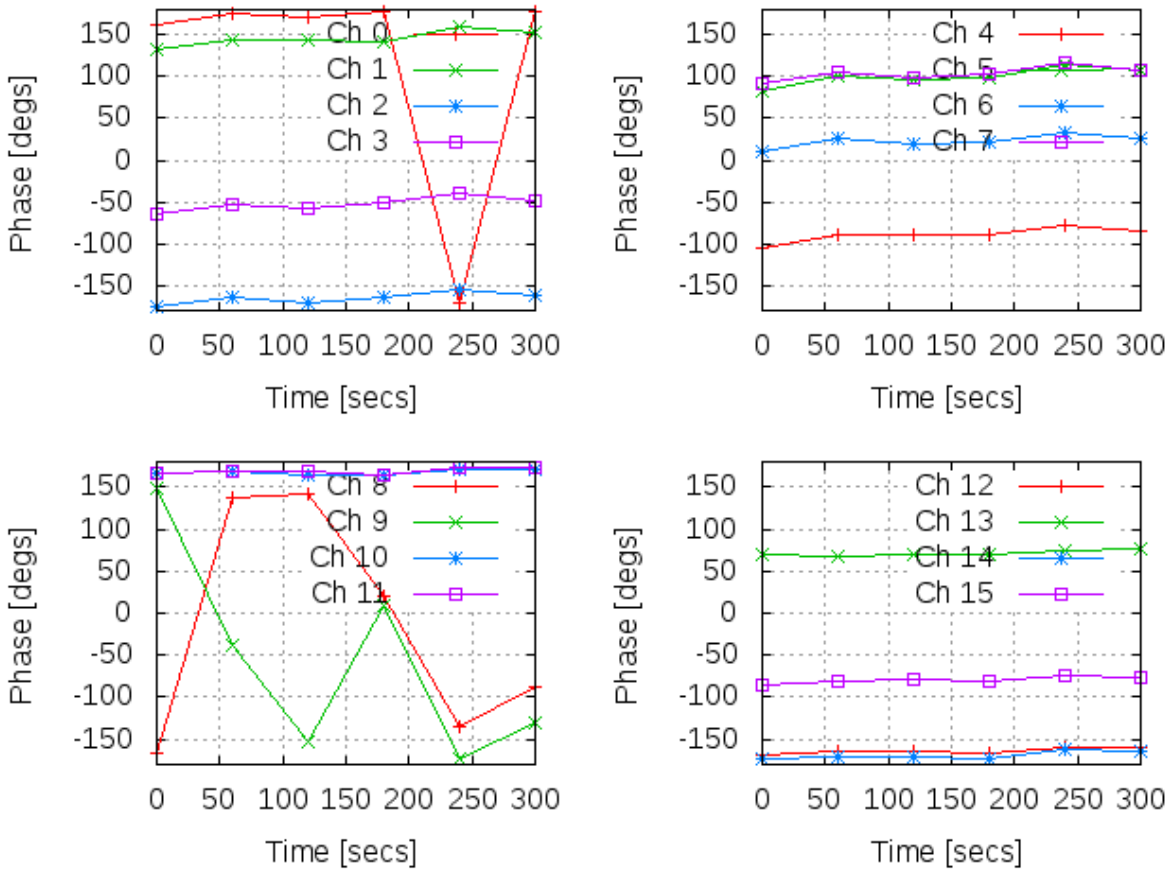


Figure 5: *Dependence of the phase from the phase cal tones for 16 channels as a function of time. Channels 8 and 9 do not contain valid data. The DBBC had been repaired at Bonn*

We concluded that after the repair in Bonn, the DBBC was behaving correctly. In order to check it, observation C1414, which was included in the CONT14 observation, was observed in parallalel with the VLBA5 in May 19th. The results are summarized below.

According to fourfit no phase jumps were seen. Two issues have been spotted, one is the low amplitude of the correlated signal. One should expect values close to 10000 and we get numbers as low as 4000. Most of them come from the extended X band. We believe this is due to RFI caused by the local oscillator of the extended X band receiver which leaks into the IF. The RFI forces the auto gain to work with less gain and affects the whole band. For future observations the target gain will be modified from 38000 to 42000 to get higher level signals all across the band. The second issue is related to RFI in several channels in the S band. This RFI is due to a very polluted environment in the nearby of the observatory and does not have a simple solution. A slight increase in the power might increase the level count of the correlated signal, but this is not as clear as in the previous case.

In any case no instrumental phase drift is observed and we estimate that the DBBC is behaving correctly.

# 4 The automated procedure

We have developed an automated procedure that performs all the steps required to measure the phase versus time as described above. The procedure belongs to *station.prc* and blocks the Field System until it completes and displays the result on the screen. The procedure in station is composed by the following instructions:

```
phasecalvsti  00000000000x
mk5=dot_set:force
log=pcalti
pcalon
!+2s
ifa=1,agc,1,40000
ifb=1,agc,1,40000
ifc=1,agc,1,40000
ifd=1,agc,1,40000
form=geo
mk5b_mode=ext,0xffffffff,2
lo=loa,7650.00,usb,rcp,1
lo=lob,8100.00,usb,rcp,1
lo=loc,1530.00,usb,rcp,1
lo=lod,1530.00,usb,rcp,1
bbc01=562.99,a,8.00
bbc02=602.99,a,8.00
bbc03=702.99,a,8.00
bbc04=862.99,a,8.00
bbc05=632.99,b,8.00
bbc06=752.99,b,8.00
bbc07=812.99,b,8.00
bbc08=832.99,b,8.00
bbc09=695.99,c,8.00
bbc10=715.99,c,8.00
```

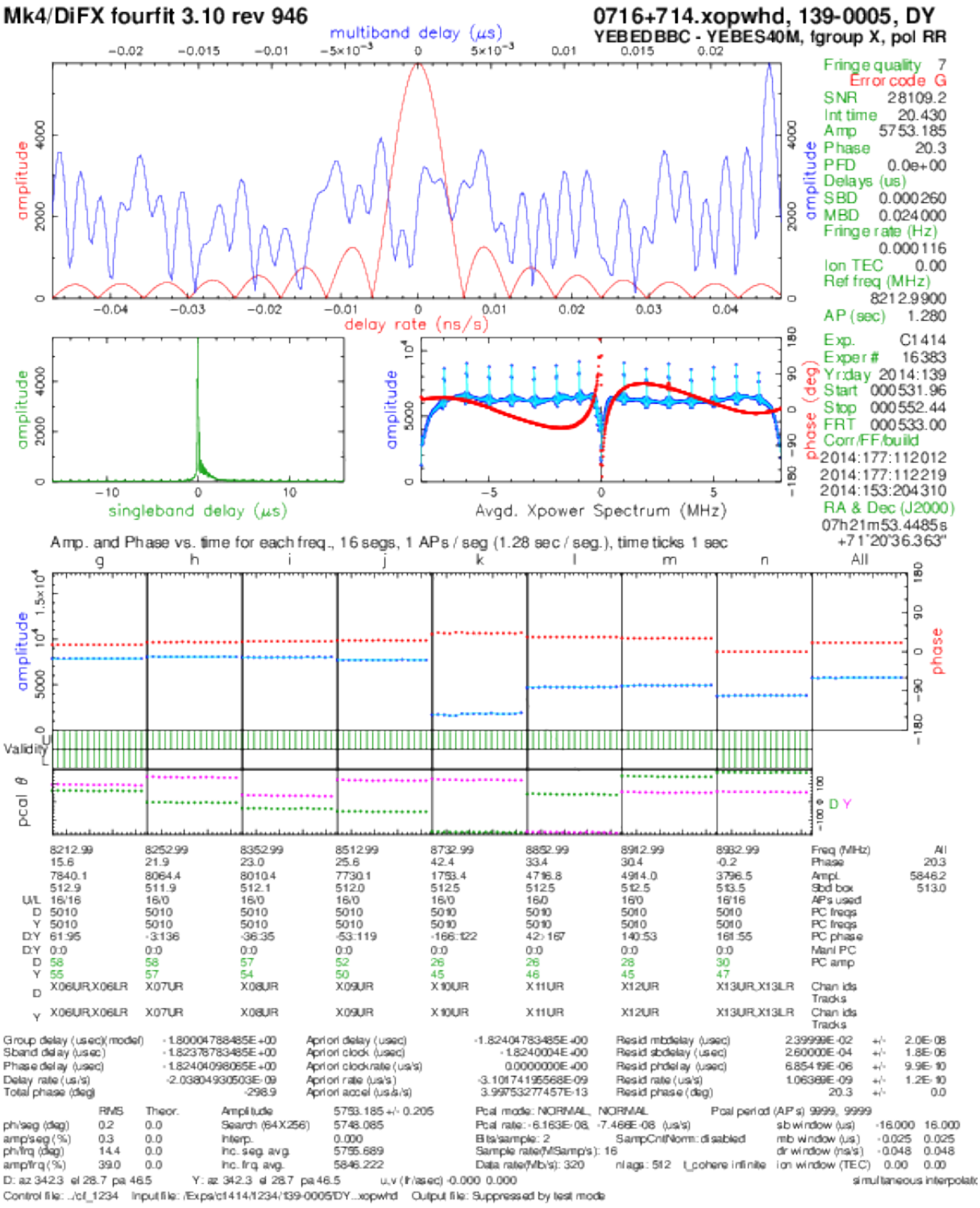


Figure 6: *Fourfit plot at X band between the DBBC and the VLBA5 terminals for Yebes for experiment C1414. Scan at 11:20*

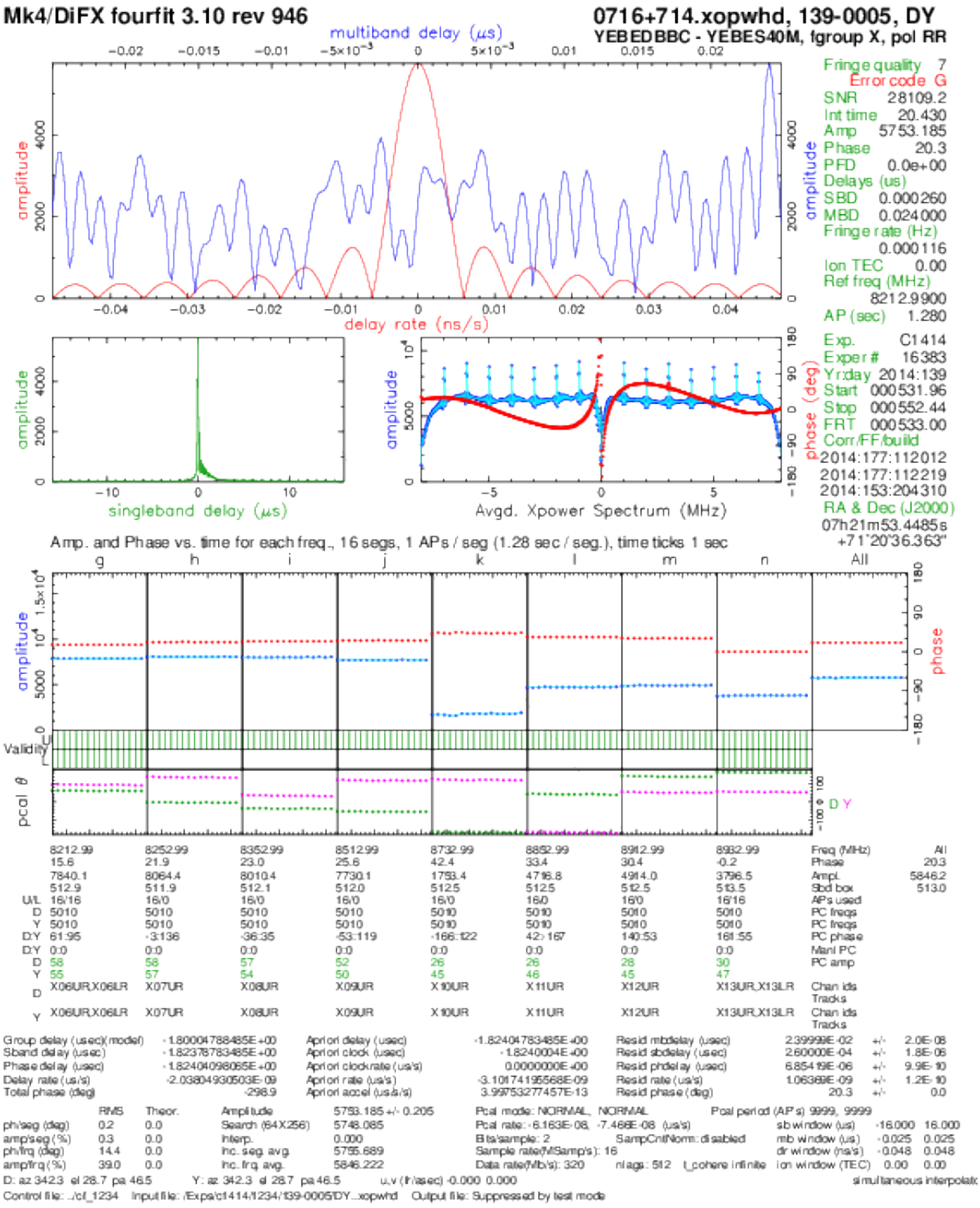


Figure 7: *Fourfit plot at S band between the DBBC and the VLBA5 terminals for Yebes for experiment C1414. Scan at 11:20*

```
bbc11=735.99,c,8.00
bbc12=765.99,c,8.00
bbc13=815.99,d,8.00
bbc14=835.99,d,8.00
bbc15=853.99,d,8.00
bbc16=938.99,d,8.00
!+10s
mk5b_mode
form
iread
bread
!+1s
scan_name=dbbc_pcal_ti
!+2s
disk_record=on
!+7m
disk_record=off
!+5s
disk2file=,/home/data/dbbc_pcal_vstime_1.m5b,+10s,+11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_2.m5b,+1m10s,+1m11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_3.m5b,+2m10s,+2m11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_4.m5b,+3m10s,+3m11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_5.m5b,+4m10s,+4m11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_6.m5b,+5m10s,+5m11s
!+10s
disk2file=,/home/data/dbbc_pcal_vstime_7.m5b,+6m10s,+6m11s
!+10s
sy=exec /usr2/oper/bin/checkphases.py
enddef
```

Once the data has been extracted in the internal Mark5B disk, script *checkphases.py* is triggered. This script starts *bpcal* in the Mark5, extracts the amplitude and phase of a specified tone for the seven samples, creates a file with the data, transfers it to the Field System computer at */usr2/log* and plots the results using gnuplot in the Field System computer. The gnuplot script is coded inside this script and all its instructions are stored in */usr2/oper/bin/plotPhases.gnuplot* every time the script is run. To end the whole procedure the user should click the left mouse button on the plot. Some lines have been splitted to make them fit in the page.

```
#!/usr/bin/env python
#

import sys
import os,math, string

def getDisk2file_Dir():
        fileIn = open('/usr2/control/skedf.ctl','r')
        linea = 'i'
        while linea != '':
                linea = fileIn.readline()
```

```
                if 'disk2file_Dir' in linea:
                        path = linea.split()[1]
                        break
        fileIn.close()
        try:
                return path
        except Exception, e:
                print "Can't find disk2file_Dir in skedf.ctl."
                sys.exit(-1)

def getMark5b_address():
        fileIn = open('/usr2/control/mk5ad.ctl','r')
        linea = 'i'
        while linea != '':
                linea = fileIn.readline()
                if '*' not in linea:
                        mark5b_address = linea.split()[0]
                        break
        fileIn.close()
        try:
                return mark5b_address
        except Exception, e:
                print "Can't find mark5b_address in mk5ad.ctl."
                sys.exit(-1)

def fillTheMark5BScript(dataPathAtMark5B, tone, m5fileExtension, file_basename, fileName):
        '''
        '''

        #First create the python script for extracting data in the Mark5B

        fOut = open('/usr2/oper/bin/extractPhases.py', 'w')

        strLine = '#!/usr/bin/env python\nfrom commands import *\n\n'
        fOut.write(strLine)

        strLine = "PATH = '%s'\nfile_basename = '%s'\next = '%s'\ntone='%s'\nfileName='%s'\n\n" %
(dataPathAtMark5B, file_basename, m5fileExtension, tone, fileName)
        fOut.write(strLine)

        strLine = "nfiles = 6\n\nfOut = open(fileName, 'w')\n\n"
        fOut.write(strLine)

        strLine = 'for i in range(0, nfiles):\n\tstrLine = "%s/%s%d%s %s" %
(PATH, file_basename, i+1, ext, tone)\n\t'
        fOut.write(strLine)

        strLine = "process_command = '/home/oper/bin/bpcal %s' % strLine\n\toutStr = getoutput(process_commanc
toneList = []\n\ttoneList.append(outStr.split('phase(dg)\\n')[1].split('\\n'))\n\t"
        fOut.write(strLine)

        strLine = 'strTime = "%d " % (i * 60)\n\tfor channelAmpPhase in toneList[0]:\n\t\t'
        fOut.write(strLine)

        strLine = "strTime = strTime + channelAmpPhase + ' '\n\tstrTime = strTime + '\\n'\n\t"
        fOut.write(strLine)

        strLine = "fOut.write(strTime)\n\nfOut.close()"
        fOut.write(strLine)

        fOut.close()


def generateGNUplots(datafile, gnuplotfile):
        '''
        '''
```

```
        #fOut = open('/usr2/oper/bin/plotPhases.gnuplot', 'w')
        fOut = open(gnuplotfile,'w')
        #datafile = '/usr2/log/phasecal_testN.dat'

        fOut.write('set yrange [-180:180]\n')
        fOut.write('set grid\n')
        fOut.write("set xlabel 'Time [secs]'\n")
        fOut.write("set ylabel 'Phase [degs]'\n")
        fOut.write("set multiplot\n")
        fOut.write('set size 0.5,0.5\n')
        fOut.write('set origin 0.0,0.5\n')
        strLine = 'plot '
        channelColumn = 4
        for channel in range(0,4):
                strLine = strLine + '"%s" using 1:%d with lp title "Ch %d"' %
(datafile, channelColumn, channel)
                if channel < 3:
                        strLine = strLine + ', '
                channelColumn = channelColumn + 3
        strLine = strLine + '\n'
        fOut.write(strLine)
        fOut.write('set origin 0.5,0.5\n')
        strLine = 'plot '
        for channel in range(4,8):
                strLine = strLine + '"%s" using 1:%d with lp title "Ch %d"' %
(datafile, channelColumn, channel)
                if channel < 7:
                        strLine = strLine + ', '
                channelColumn = channelColumn + 3
        strLine = strLine + '\n'
        fOut.write(strLine)
        fOut.write('set origin 0.0,0.0\n')
        strLine = 'plot '
        for channel in range(8,12):
                strLine = strLine + '"%s" using 1:%d with lp title "Ch %d"' %
(datafile, channelColumn, channel)
                if channel < 11:
                        strLine = strLine + ', '
                channelColumn = channelColumn + 3
        strLine = strLine + '\n'
        fOut.write(strLine)
        fOut.write('set origin 0.5,0.0\n')
        strLine = 'plot '
        for channel in range(12,16):
                strLine = strLine + '"%s" using 1:%d with lp title "Ch %d"' %
(datafile, channelColumn, channel)
                if channel < 15:
                        strLine = strLine + ', '
                channelColumn = channelColumn + 3
        strLine = strLine + '\n'
        fOut.write(strLine)
        fOut.write('unset multiplot\n')
        fOut.write('pause mouse "click with the mouse to exit"\n')
        fOut.close()

#Basic settings _____________________

mark5b_username = 'oper'
mark5b_address = getMark5b_address()
dataPathAtMark5B = getDisk2file_Dir()

# Other settings _____________________
tone = '2010'   # You can choose 2010, 3010, 4010, 5010, 6010, and so on. Limited by your bandwidth
m5fileExtension = '.m5b'
file_basename = 'dbbc_pcal_vstime_'
fileName = 'phasecal_testN.dat'
myPythonScript = 'extractPhases.py'
```

```
datafile = '/usr2/log/%s' % fileName
gnuplotfile = '/usr2/oper/bin/plotPhases.gnuplot'
#_____________________________


fillTheMark5BScript(dataPathAtMark5B, tone, m5fileExtension, file_basename, fileName)

# Copy the file to the Makr5B
process_command = "scp /usr2/oper/bin/extractPhases.py %s@%s:/%s/extractPhases.py"  % (mark5b_username, mark5b
os.system(process_command)

#change the execution mode to be able to run it
process_command = "rsh %s@%s 'chmod a+x %s/extractPhases.py'"  % (mark5b_username, mark5b_address, dataPathAtM
os.system(process_command)

#change the execution mode to be able to run it
process_command = "rsh %s@%s '%s/extractPhases.py'"  % (mark5b_username, mark5b_address, dataPathAtMark5B)
os.system(process_command)
#print outStr

process_command = "scp %s@%s:%s /usr2/log/"  % (mark5b_username, mark5b_address, fileName)
os.system(process_command)
#print outStr

generateGNUplots(datafile, gnuplotfile)

process_command = "gnuplot %s" % (gnuplotfile)
os.system(process_command)
#print outStr
```

# References

[1] D. Smythe 2009. Mark5B Utility Programs. *http://www.haystack.edu/tech/vlbi/mark5/mark5_memos/073.2*