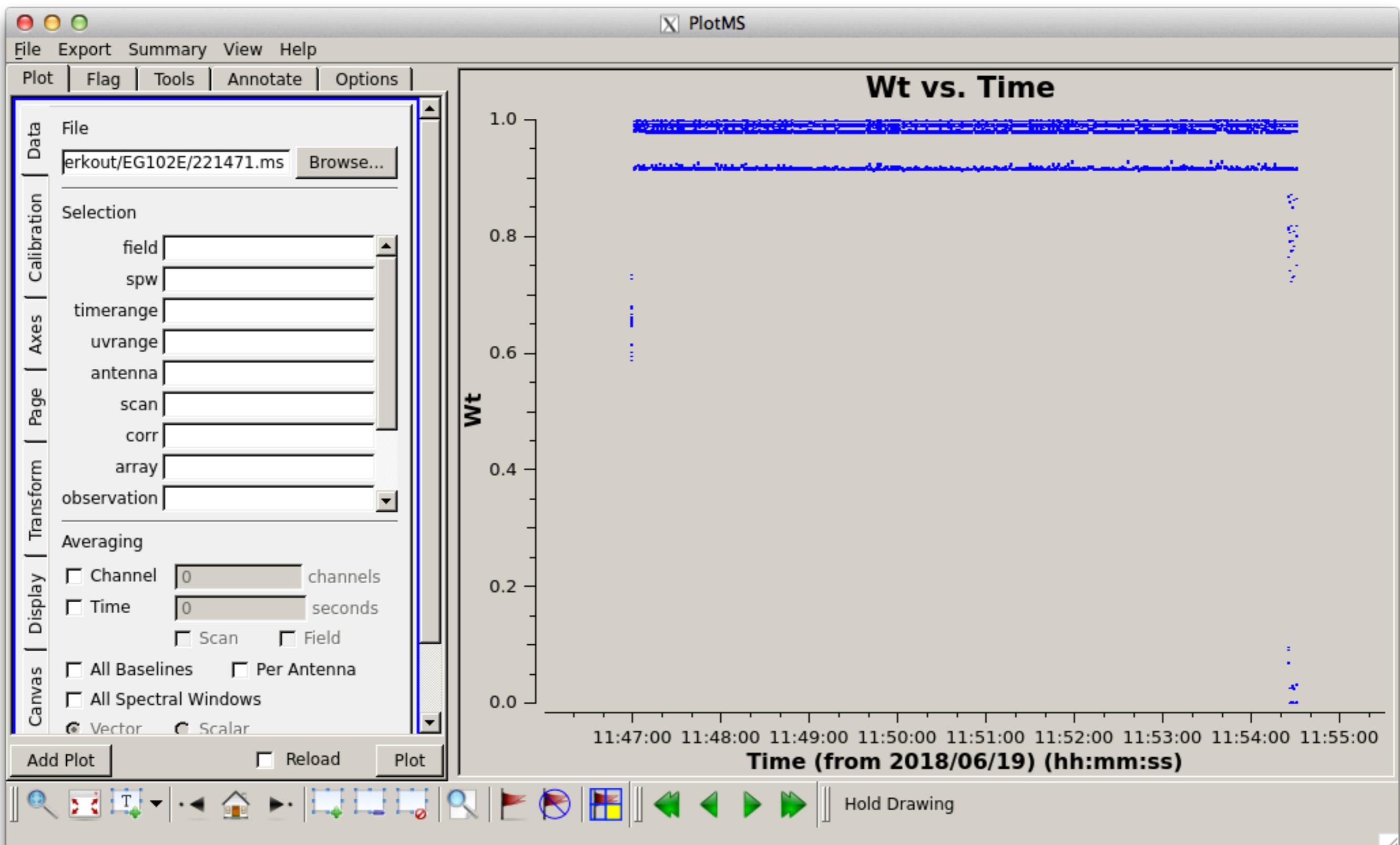
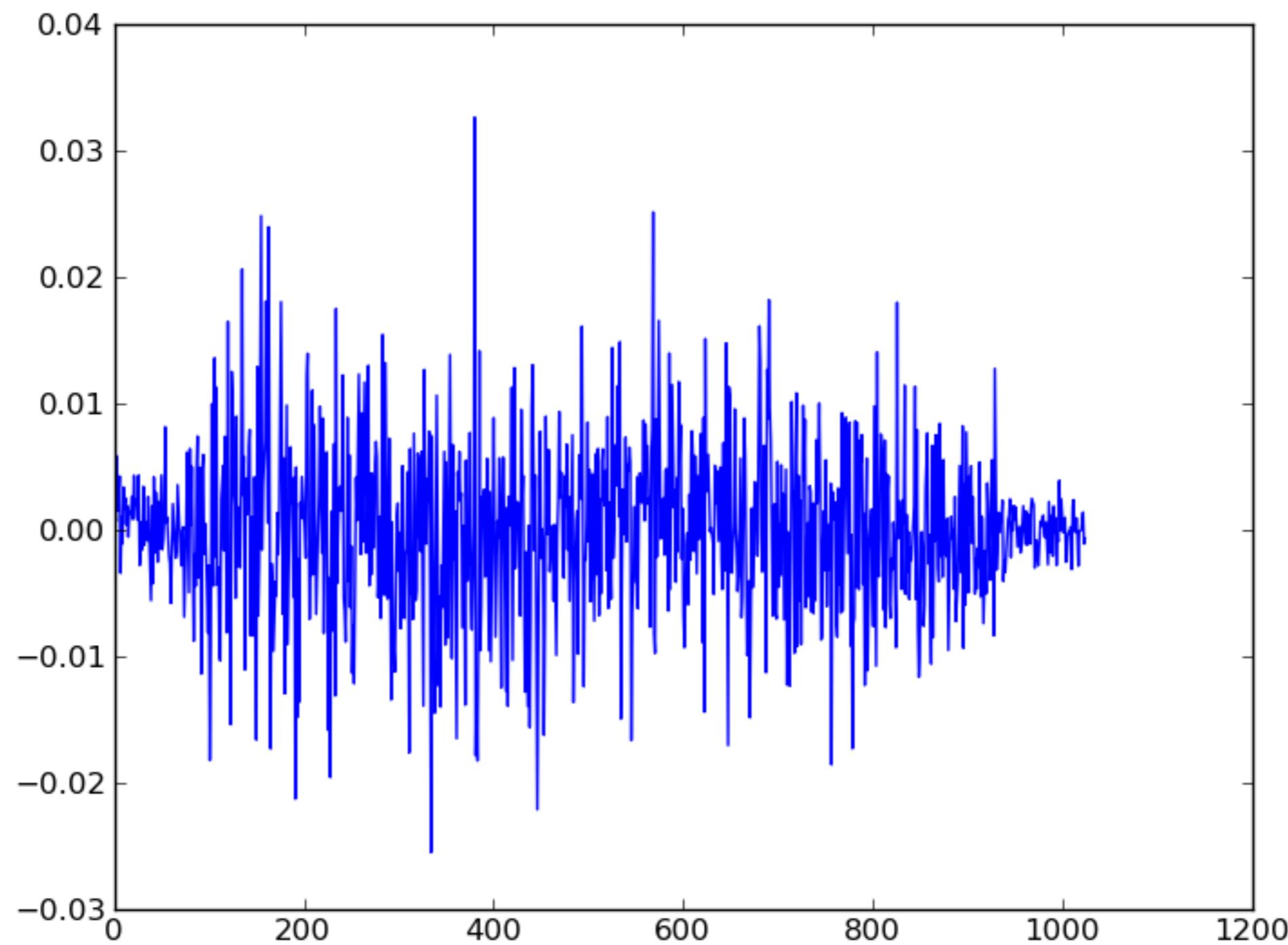


1 , 583 , 308 , 800





## EG102E

weight versus time

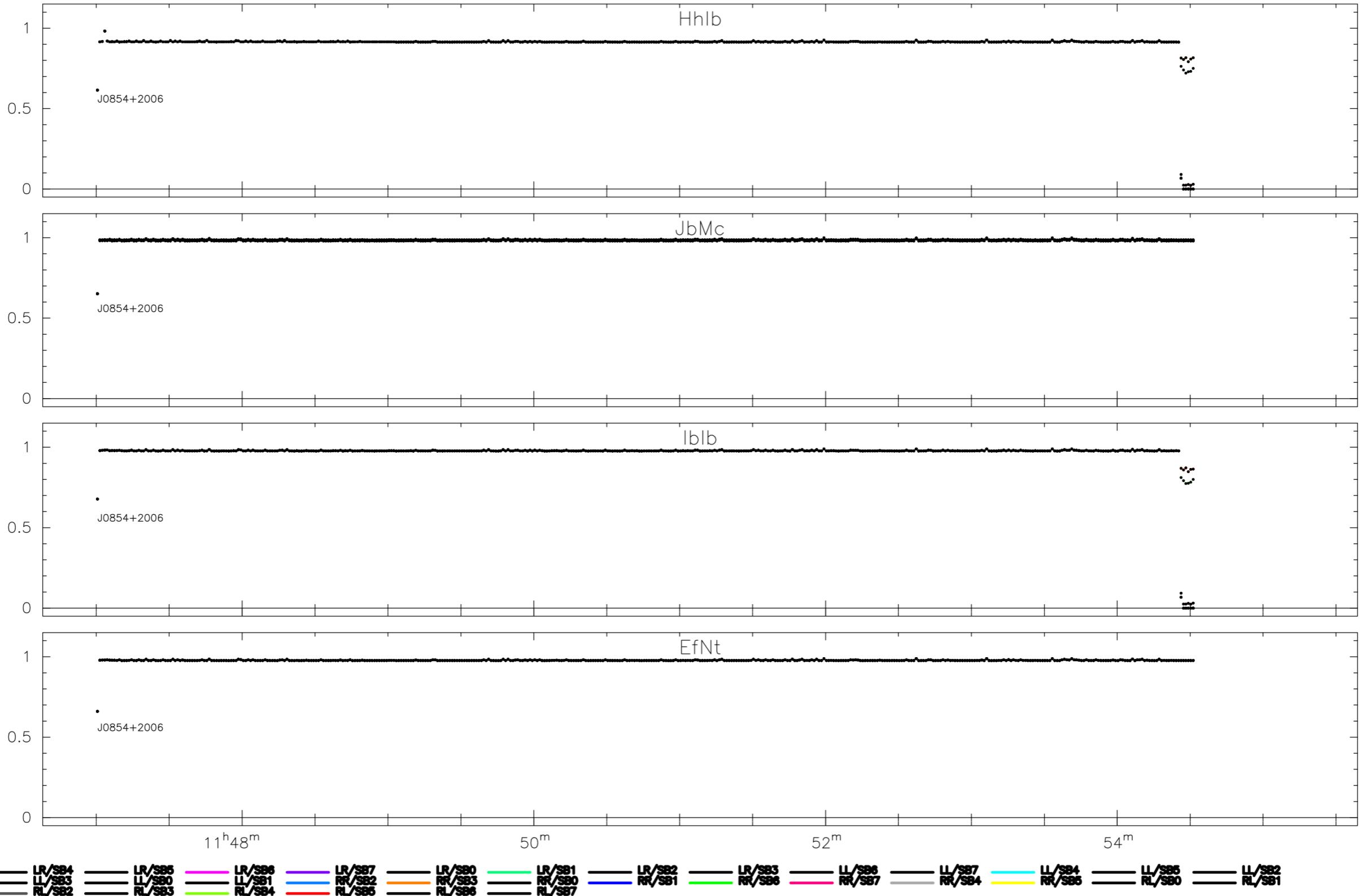
data: 221471.ms [DATA]

unique: sess218.C1024e/J0854+2006

verkout@&lt;??&gt; 2019-08-12T15:55:40

Pol=RL,LL,LR,RR;Nsub=\*&amp;;Ch=\*

page: 1/7



$1+2i$

CASA\_memo1\_msdefinition\_Kemball.pdf (page 1 of 52)

# MeasurementSet definition version 2.0

A.J. Kemball (NRAO) and M.H. Wieringa (ATNF), eds.

January 21, 2000

A pdf version of this note is available.

## Contents

<b>1</b>	<b>Summary</b>	<b>4</b>
<b>2</b>	<b>Introduction</b>	<b>4</b>
<b>3</b>	<b>Summary of changes</b>	<b>7</b>
3.1	MAIN table . . . . .	7
3.1.1	MAIN keywords . . . . .	7
3.1.2	Non-standard MAIN data . . . . .	8
3.1.3	DATA labeling . . . . .	9
3.1.4	Processor information . . . . .	9
3.1.5	State information . . . . .	9

- **JIVE**  
(Joint Institute for VLBI - internal since 1997)
- **ALMA**  
(Atacama Large Millimeter Array - ASDM\*)
- **EVLA**  
(Expanded Very Large Array - SDM\*)
- **MeerKAT**  
(More Karoo Array Telescope(s) - 7  $\Rightarrow$  64)
- **LOFAR**
- **(SKA?) ...**

\*SDM = *Science Data Model*



Table Browser

File Edit View Tools Export Help About



221471.ms



table data

table keywords

field keywords

	UVW	FLAG	WEIGHT	ANTENNA1	ANTENNA2	DATA_DESC_ID	EXPOSURE	FIELD_ID	TIME	DATA
0	[0, 0, 0]	[4, 1024] Boolean	[0.732761, 0.732761, 0.73...]	0	0	0	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
1	[0, 0, 0]	[4, 1024] Boolean	[0.732761, 0.732761, 0.73...]	0	0	1	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
2	[0, 0, 0]	[4, 1024] Boolean	[0.732761, 0.732761, 0.73...]	0	0	2	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
3	[0, 0, 0]	[4, 1024] Boolean	[0.732761, 0.732761, 0.73...]	0	0	3	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
4	[0, 0, 0]	[4, 1024] Boolean	[0.732633, 0.732633, 0.73...]	0	0	4	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
5	[0, 0, 0]	[4, 1024] Boolean	[0.732633, 0.732633, 0.73...]	0	0	5	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
6	[0, 0, 0]	[4, 1024] Boolean	[0.732633, 0.732633, 0.73...]	0	0	6	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
7	[0, 0, 0]	[4, 1024] Boolean	[0.732633, 0.732633, 0.73...]	0	0	7	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
8	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.65009, 0.65009, 0.650...]	0	1	0	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
9	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649962, 0.649962, 0.64...]	0	1	1	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
10	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.65009, 0.65009, 0.650...]	0	1	2	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
11	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649962, 0.649962, 0.64...]	0	1	3	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
12	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649962, 0.649962, 0.64...]	0	1	4	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
13	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649833, 0.649833, 0.64...]	0	1	5	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
14	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649962, 0.649962, 0.64...]	0	1	6	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
15	[-257848, 388868, -521...]	[4, 1024] Boolean	[0.649833, 0.649833, 0.64...]	0	1	7	0.999424	18	2018-06-19-11:47:0...	[4, 1024] Complex
16	-41911.7,	[4, 1024]	[0.645476,	0	3	0	0.999424	18	2018-06-19-11:47:0...	[4, 1024]

Restore Columns

Resize Headers

PAGE NAVIGATION

First

&lt;&lt; [ 1 / 102 ] &gt;&gt;

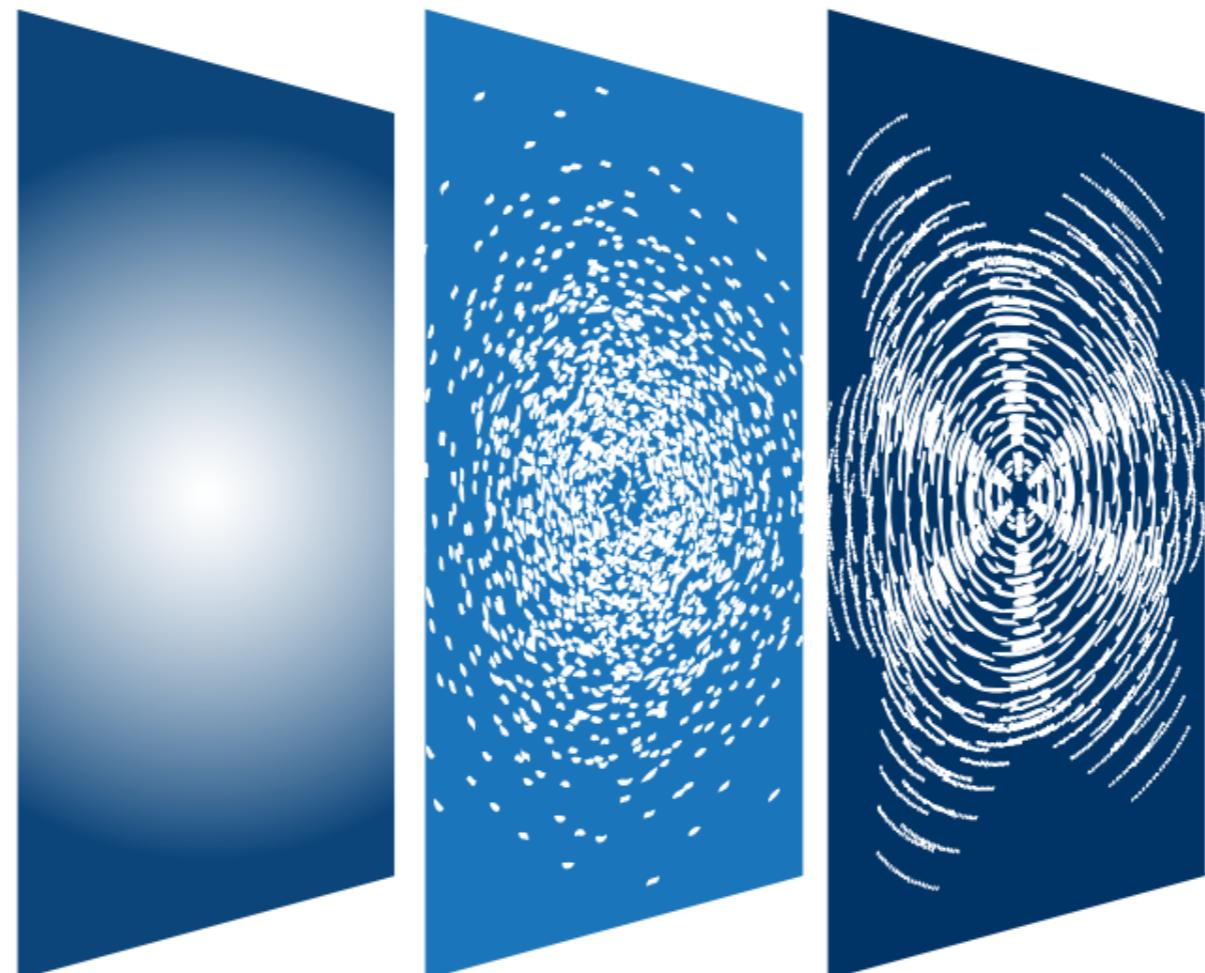
Last

1

Go

Loading 1000

rows.



# CASA

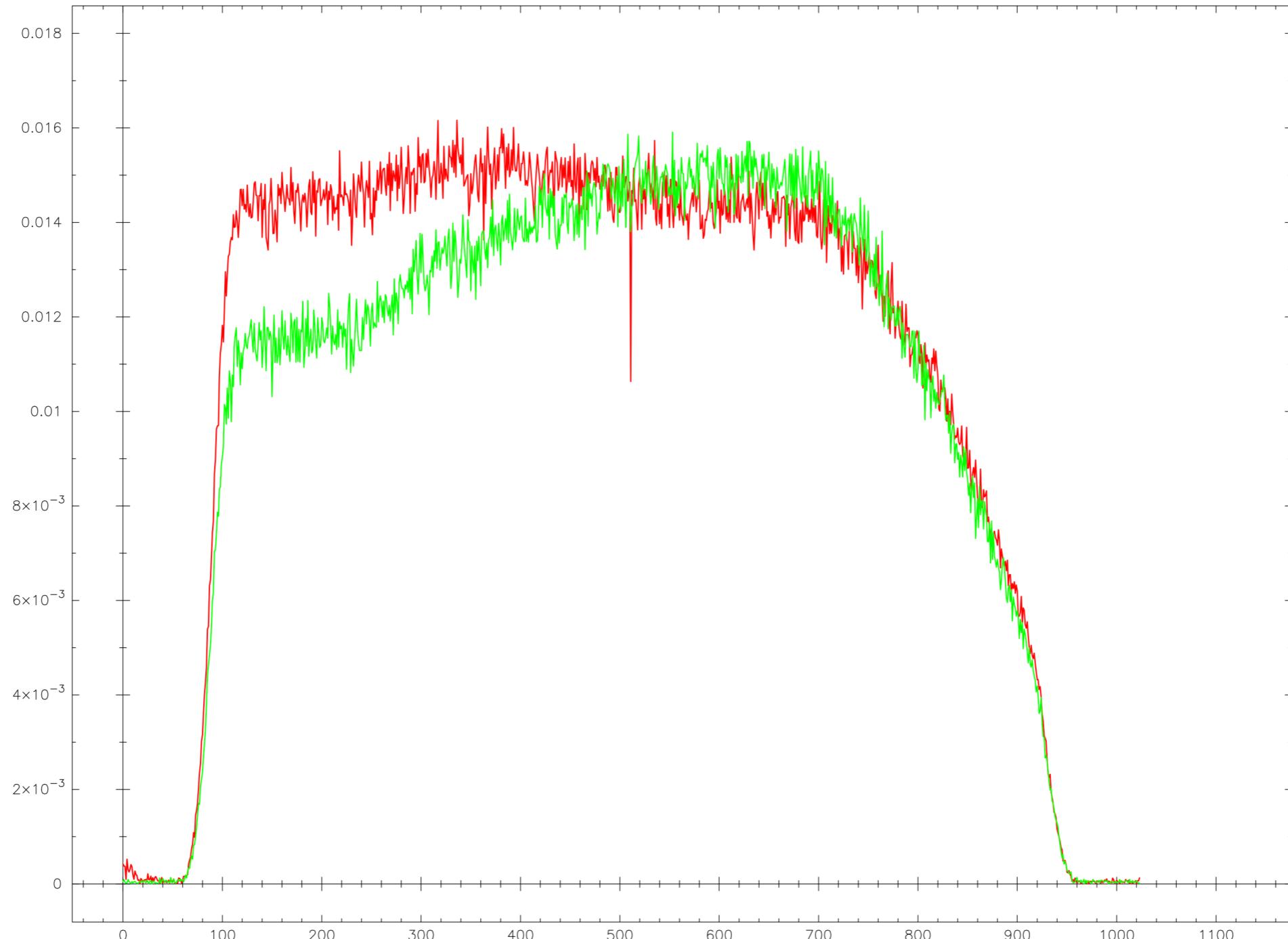
---

Common Astronomy  
Software Applications

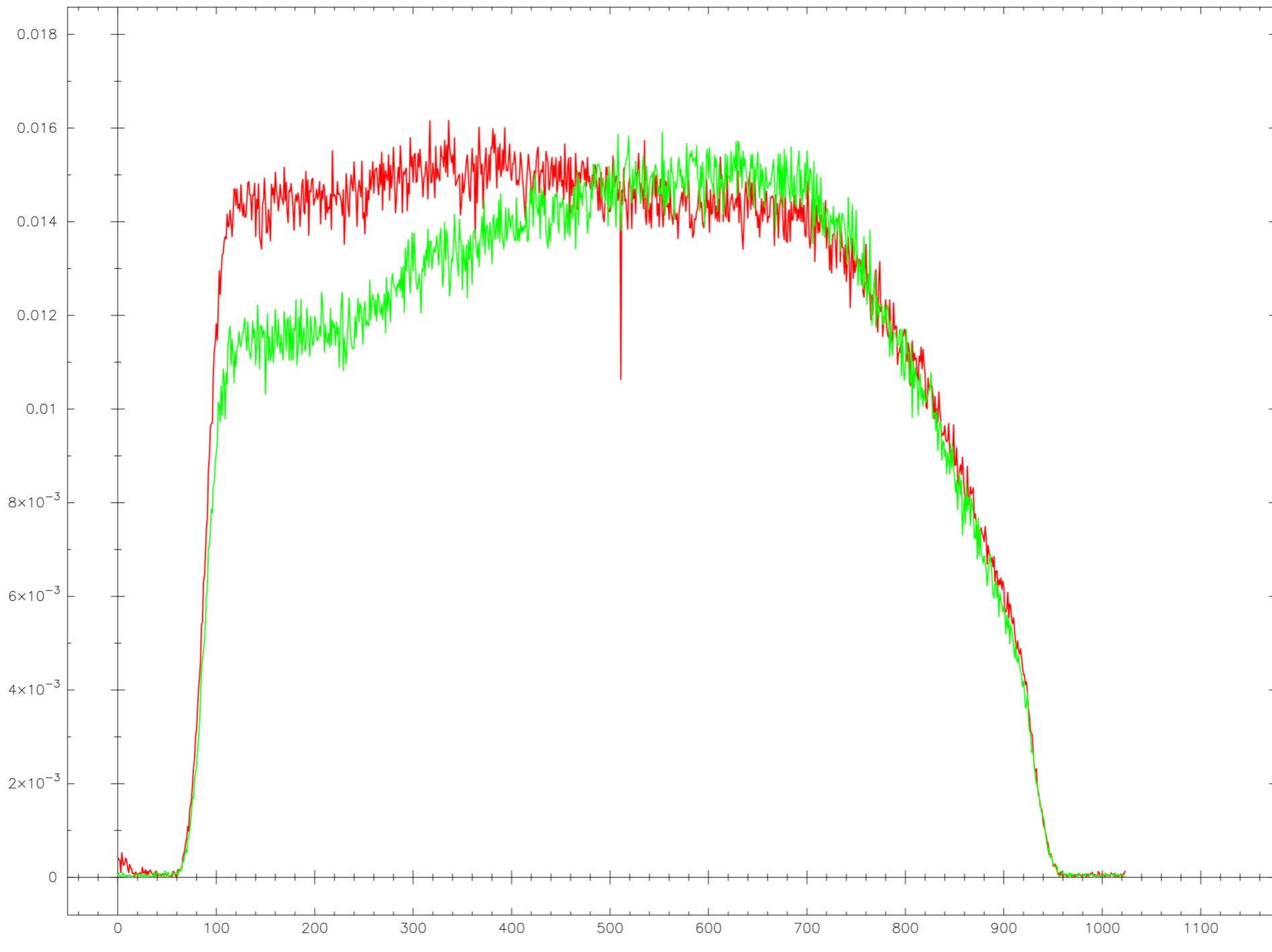
---

```
$> python  
>>> import pyrap.tables
```

old module: **pyrap**  
new module: **python-casacore**  
<https://github.com/casacore/python-casacore>



→amplitude(arbitrary units)



$\rightarrow\nu$  ( MHz )

a.k.a. "channels"

# Visualizing raw radio data

Visualizing  
raw radio  
data  
*and why that is a  
challenge*

37j	0.46+0.22j	0.68+0.73j	0.74+0.54j	0.00+0.95j	0.82+0.98j	0.39+0.35j	0.53+0.41j	0.17+0.29j	0.24+0.02j	0.79+0.62j	0.35+0.05j	0.61+0.01j	0.71+
18j	0.69+0.72j	0.63+0.53j	0.74+0.65j	0.79+0.10j	0.26+0.95j	0.11+0.40j	0.27+0.82j	0.02+0.88j	0.62+0.86j	0.90+0.63j	0.39+0.55j	0.33+0.33j	0.54+
10j	0.72+0.45j	0.04+0.27j	0.06+0.42j	0.94+0.44j	0.88+0.73j	0.31+0.21j	0.36+0.74j	0.62+0.68j	0.83+0.08j	0.51+0.01j	0.92+0.50j	0.78+0.29j	0.71+
01j	0.57+0.45j	0.53+0.59j	0.24+0.69j	0.22+0.03j	0.56+0.35j	0.09+0.36j	0.26+0.92j	0.99+0.79j	0.43+0.64j	0.22+0.50j	0.30+0.09j	0.48+0.09j	0.54+
51j	0.94+0.53j	0.53+0.87j	0.43+0.87j	0.77+0.31j	0.30+0.63j	0.86+0.42j	0.79+0.25j	0.60+0.87j	0.55+0.51j	0.77+0.59j	0.65+0.64j	0.70+0.04j	0.94+
79j	0.70+0.50j	0.81+0.16j	0.14+0.54j	0.76+0.49j	0.48+0.74j	0.49+0.20j	0.83+0.53j	0.55+0.75j	0.66+0.33j	0.33+0.24j	0.46+0.35j	0.45+0.41j	0.03+
09j	0.86+0.71j	0.22+0.20j	0.76+0.39j	0.41+0.99j	0.93+0.64j	0.76+0.57j	0.01+0.26j	0.91+0.38j	0.09+0.23j	0.13+0.01j	1.00+0.65j	0.75+0.20j	0.10+
49j	0.87+0.28j	0.93+0.42j	0.65+0.58j	0.38+0.66j	0.18+0.92j	0.34+0.88j	0.62+0.10j	0.21+0.29j	0.16+0.41j	0.19+0.82j	0.35+0.74j	0.52+0.84j	0.63+
73j	0.13+0.21j	0.17+0.54j	0.84+0.62j	0.71+0.02j	0.16+0.40j	0.57+0.14j	0.46+0.48j	0.49+0.92j	0.97+0.04j	0.98+0.18j	0.31+0.12j	0.32+0.90j	0.65+
32j	0.44+0.15j	0.20+0.30j	0.66+0.13j	0.88+0.14j	0.88+0.08j	0.81+0.76j	0.76+0.99j	0.23+0.26j	0.40+0.87j	0.23+0.64j	0.66+0.81j	0.45+0.91j	0.85+
57j	0.86+0.11j	0.96+0.14j	0.47+0.85j	0.64+0.36j	0.15+0.57j	0.46+0.59j	0.71+0.91j	0.71+0.20j	0.77+0.93j	0.09+0.60j	0.70+0.65j	0.38+0.40j	0.35+
90j	0.30+0.76j	0.88+0.20j	0.46+0.34j	0.62+0.95j	0.55+0.34j	0.20+0.92j	0.55+0.76j	0.63+0.79j	0.63+0.48j	0.89+0.99j	0.28+0.87j	0.56+0.46j	0.08+
65j	0.61+0.24j	0.89+0.10j	0.34+0.16j	0.48+0.78j	0.49+0.94j	0.85+0.28j	0.23+0.31j	0.27+0.90j	0.22+0.84j	0.98+0.93j	0.67+0.19j	0.62+0.38j	0.00+
00j	0.63+0.51j	0.11+0.34j	0.51+0.93j	0.78+0.70j	0.60+0.32j	0.65+0.75j	0.48+0.88j	0.60+0.35j	0.86+0.02j	0.61+0.81j	0.81+0.72j	0.05+0.19j	0.54+
62j	0.07+0.18j	0.97+0.72j	0.54+0.28j	0.87+0.46j	0.54+0.20j	0.58+0.40j	0.29+0.44j	0.56+0.09j	0.54+0.85j	0.06+0.54j	0.02+0.26j	0.99+0.81j	0.11+
74j	0.24+0.32j	0.60+0.81j	0.49+0.48j	0.12+0.93j	0.33+0.72j	0.47+0.11j	0.82+0.14j	0.02+1.00j	0.86+0.75j	0.23+0.16j	0.45+0.56j	0.05+0.99j	0.18+
04j	0.40+0.29j	0.51+0.84j	0.87+0.11j	0.37+0.64j	0.61+0.63j	0.64+0.22j	0.21+0.29j	0.23+0.14j	0.35+0.65j	0.70+0.59j	0.19+0.08j	0.78+0.02j	0.39+
65j	0.18+0.57j	0.72+0.44j	0.20+0.75j	0.96+0.29j	0.87+0.68j	0.99+0.82j	0.03+0.09j	0.31+0.33j	0.43+0.58j	0.41+0.11j	0.60+0.25j	0.30+0.94j	0.88+
38j	0.80+0.76j	0.44+0.99j	0.59+0.84j	0.77+0.09j	0.33+0.78j	0.88+0.19j	0.35+0.55j	0.27+0.19j	0.40+0.81j	0.74+0.17j	0.11+0.76j	0.72+0.78j	0.34+
78j	0.20+0.06j	0.82+0.58j	0.41+0.72j	0.83+0.95j	0.36+0.65j	0.01+0.25j	0.70+0.21j	0.71+0.94j	0.76+0.60j	0.69+0.30j	0.98+0.01j	0.14+0.41j	0.26+
30j	0.32+0.93j	0.47+0.19j	0.91+0.64j	0.87+0.50j	0.59+0.17j	0.06+0.25j	0.03+0.66j	0.43+0.79j	0.48+0.06j	0.85+0.54j	0.98+0.69j	0.02+0.57j	0.34+
85j	0.43+0.23j	0.44+0.72j	0.49+0.35j	0.63+0.99j	0.85+0.30j	0.67+0.89j	0.71+0.69j	0.64+0.30j	0.04+0.59j	0.04+0.14j	0.98+0.62j	0.12+0.62j	0.05+
17j	0.24+0.22j	0.14+0.61j	0.31+0.67j	0.67+0.80j	0.29+0.63j	0.92+0.91j	0.49+0.85j	0.85+0.65j	0.89+0.38j	0.79+0.18j	0.71+0.24j	0.77+0.41j	0.15+
72j	0.92+0.98j	0.89+0.71j	0.40+0.53j	0.96+0.45j	0.95+0.61j	0.69+0.90j	0.45+0.33j	0.15+0.05j	0.77+0.97j	0.69+0.14j	0.31+0.90j	0.10+0.46j	0.31+
00j	0.43+0.42j	0.09+0.05j	0.49+0.60j	0.89+0.17j	0.31+0.82j	0.60+0.61j	0.80+0.47j	0.61+0.51j	0.83+0.69j	0.92+0.86j	0.48+0.75j	0.08+0.99j	0.88+
78j	0.46+0.77j	0.89+0.75j	0.80+0.08j	0.38+0.77j	0.43+0.41j	0.98+0.50j	0.20+0.36j	0.50+0.13j	0.31+0.33j	0.90+0.39j	0.80+0.41j	0.18+0.73j	0.64+
11j	0.92+0.20j	0.33+0.13j	0.35+0.76j	0.61+0.88j	0.75+0.30j	0.79+0.26j	0.13+0.41j	0.86+0.12j	0.04+0.87j	0.91+0.54j	0.05+0.67j	0.14+0.90j	0.18+
89j	0.21+0.81j	0.52+0.35j	0.24+0.83j	0.30+0.10j	0.63+0.11j	0.52+0.37j	0.95+0.27j	0.64+0.05j	0.10+0.12j	0.56+0.76j	0.30+0.49j	0.58+0.67j	0.60+
01j	0.81+0.42j	0.26+0.12j	0.31+0.77j	0.42+0.81j	0.11+0.94j	0.98+0.69j	0.22+0.90j	0.85+0.06j	0.49+0.47j	0.55+0.28j	0.30+0.00j	0.71+0.91j	0.18+
08j	0.25+0.75j	0.52+0.44j	0.40+0.21j	0.85+0.49j	0.03+0.87j	0.74+0.98j	0.10+0.83j	0.17+0.41j	0.33+0.81j	0.22+0.61j	0.30+0.17j	0.75+0.56j	0.58+
30j	0.39+0.71j	0.61+0.59j	0.70+0.88j	0.09+0.84j	0.20+0.76j	0.01+0.57j	0.67+0.67j	0.17+0.35j	0.57+0.95j	0.71+0.89j	0.06+0.56j	0.10+0.19j	0.77+
83j	0.10+0.26j	0.99+0.76j	0.07+0.62j	0.84+0.18j	0.04+0.28j	0.41+0.79j	0.01+0.24j	0.29+0.24j	0.19+0.55j	0.91+0.21j	0.23+0.60j	0.87+0.32j	0.42+
02j	0.66+0.69j	1.00+0.88j	0.32+0.77j	0.73+0.98j	0.34+0.01j	0.92+0.50j	0.91+0.94j	0.91+0.83j	0.59+0.56j	0.59+0.56j	0.47+0.77j	0.84+0.75j	0.24+
58j	0.50+0.32j	0.61+0.18j	0.44+0.46j	0.47+0.70j	0.26+0.81j	0.01+0.42j	0.33+0.50j	0.52+0.86j	0.77+0.38j	0.99+0.77j	0.33+0.50j	0.80+0.75j	0.88+
66j	0.10+0.58j	0.23+0.45j	0.98+0.90j	0.61+0.22j	0.02+0.42j	0.85+0.88j	0.26+0.68j	0.75+0.61j	0.42+0.11j	0.76+0.57j	0.26+0.64j	0.85+0.43j	0.30+
10j	0.35+0.29j	0.32+0.36j	0.47+0.89j	0.51+0.07j	0.63+0.04j	0.78+0.17j	0.82+0.03j	0.76+0.75j	0.96+0.13j	0.08+0.82j	0.34+0.57j	0.45+0.02j	0.51+
57j	0.96+0.58j	0.08+0.60j	0.79+0.30j	0.55+0.26j	0.48+0.18j	0.18+0.58j	0.42+0.19j	0.44+0.82j	0.79+0.46j	0.42+0.80j	0.08+0.39j	0.33+0.98j	0.08+
13j	0.65+0.45j	0.47+0.65j	0.70+0.41j	0.65+0.75j	0.98+0.51j	0.76+0.77j	0.61+0.52j	0.61+0.35j	0.34+0.11j	0.10+0.67j	0.11+0.55j	0.40+0.56j	0.01+
13j	0.03+0.10j	0.14+0.85j	0.24+0.61j	0.99+0.68j	0.94+0.25j	0.77+0.48j	0.71+0.55j	0.68+0.87j	0.06+0.51j	0.41+0.85j	0.62+0.30j	0.01+0.17j	0.40+
73j	0.94+0.11j	0.48+0.56j	0.51+0.32j	0.86+0.72j	0.37+0.07j	0.14+0.98j	0.56+0.78j	0.60+0.96j	0.57+0.37j	0.25+0.15j	0.81+0.01j	0.24+0.26j	0.83+
89j	0.76+0.60j	0.51+0.90j	0.32+0.07j	0.04+0.41j	0.68+0.15j	0.66+0.92j	0.85+0.48j	0.97+0.2					

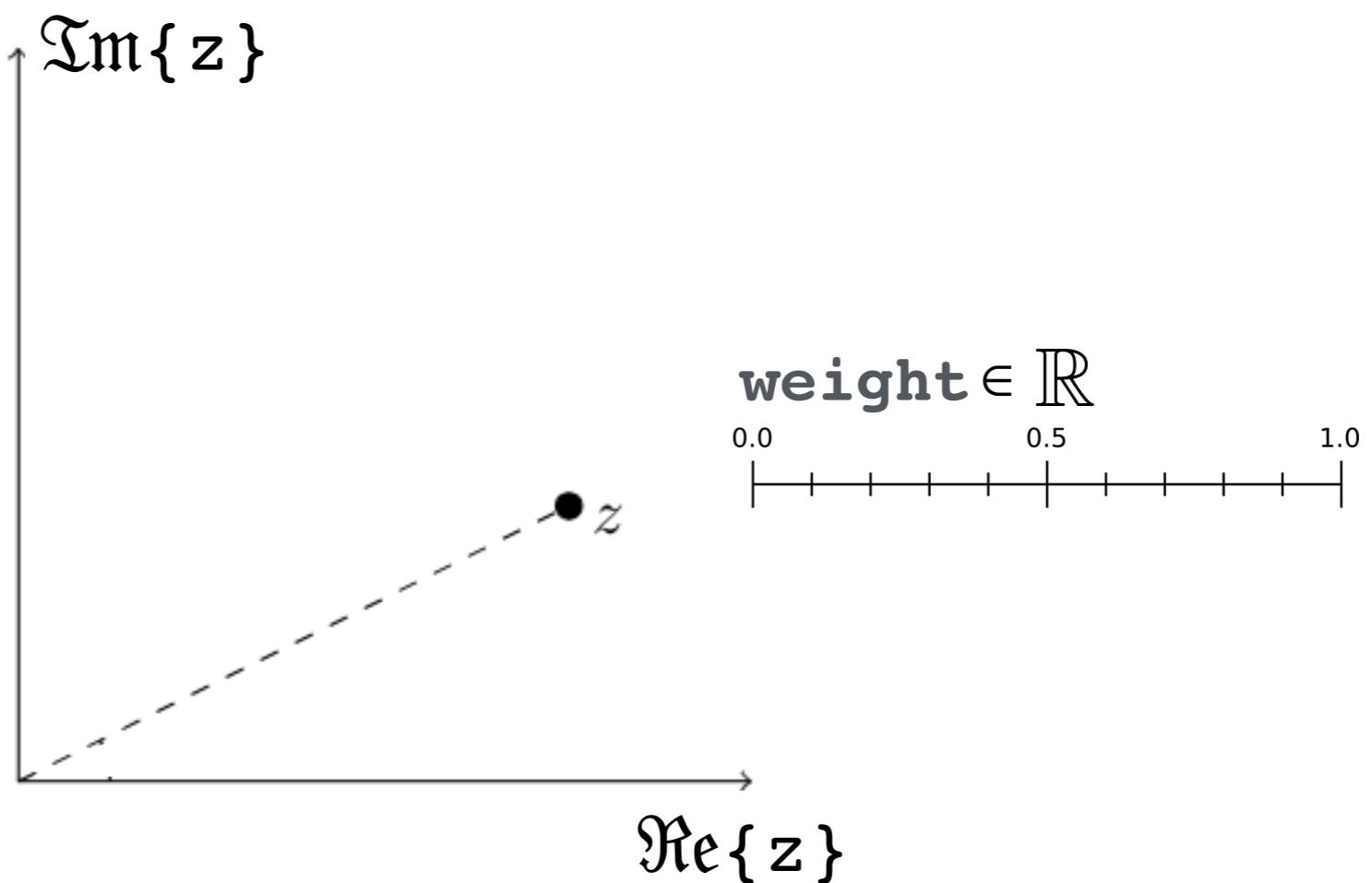
1 , 583 , 308 , 800



Only 120 4k screens here ...

# A REALLY SMALL MeasurementSet

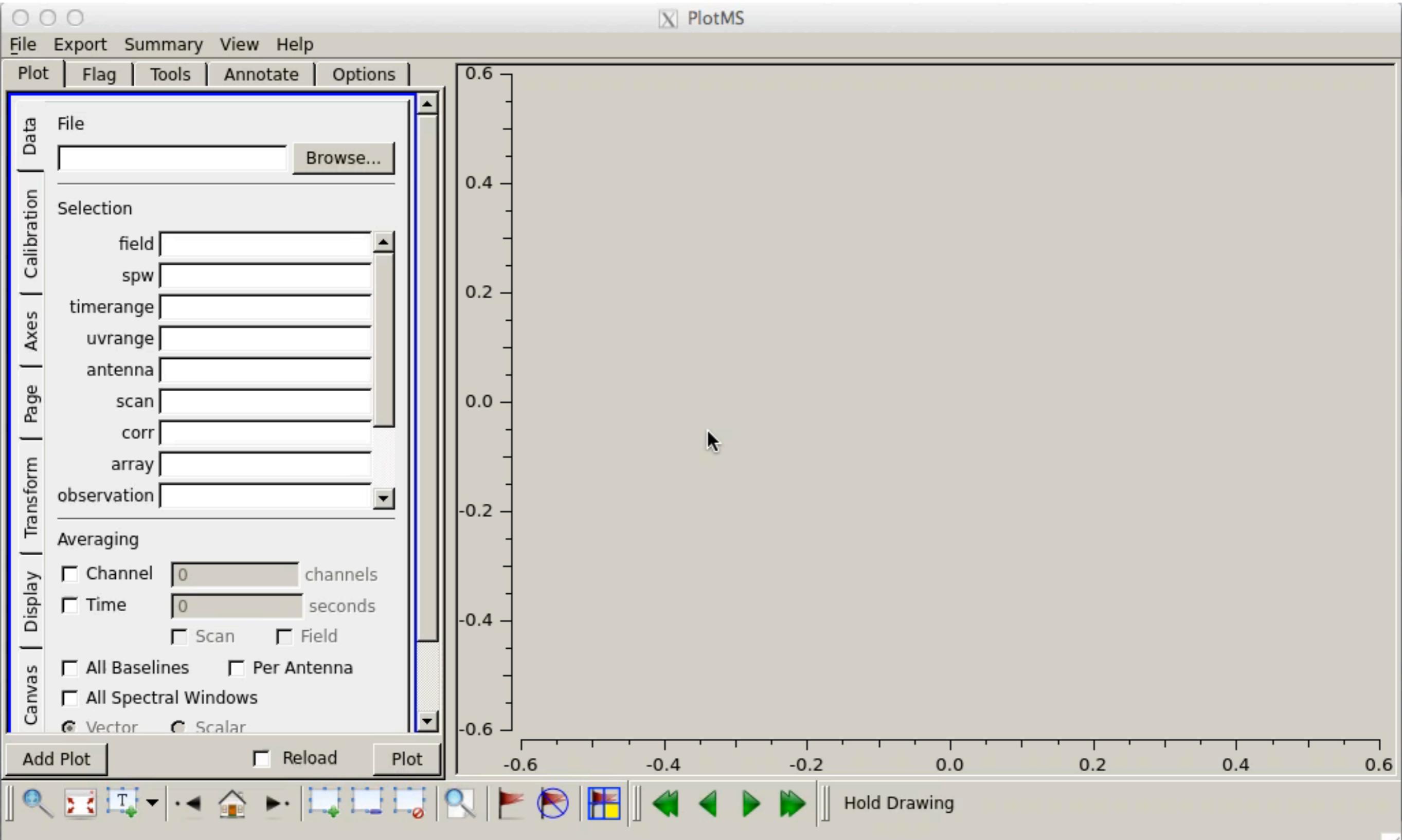
no seriously, it's *tiny*



# weight-vs-time

# weight-vs-time

- excellent indication of data quality
- where (in time) is my data good (or bad)?
- simple plot, should be *fast*



# CASA plotms tool

- slow
- *no idea what looking at*

**"It<sup>\*</sup>'s so easy,  
everyone writes their own"**

(\*) plotting raw radio data

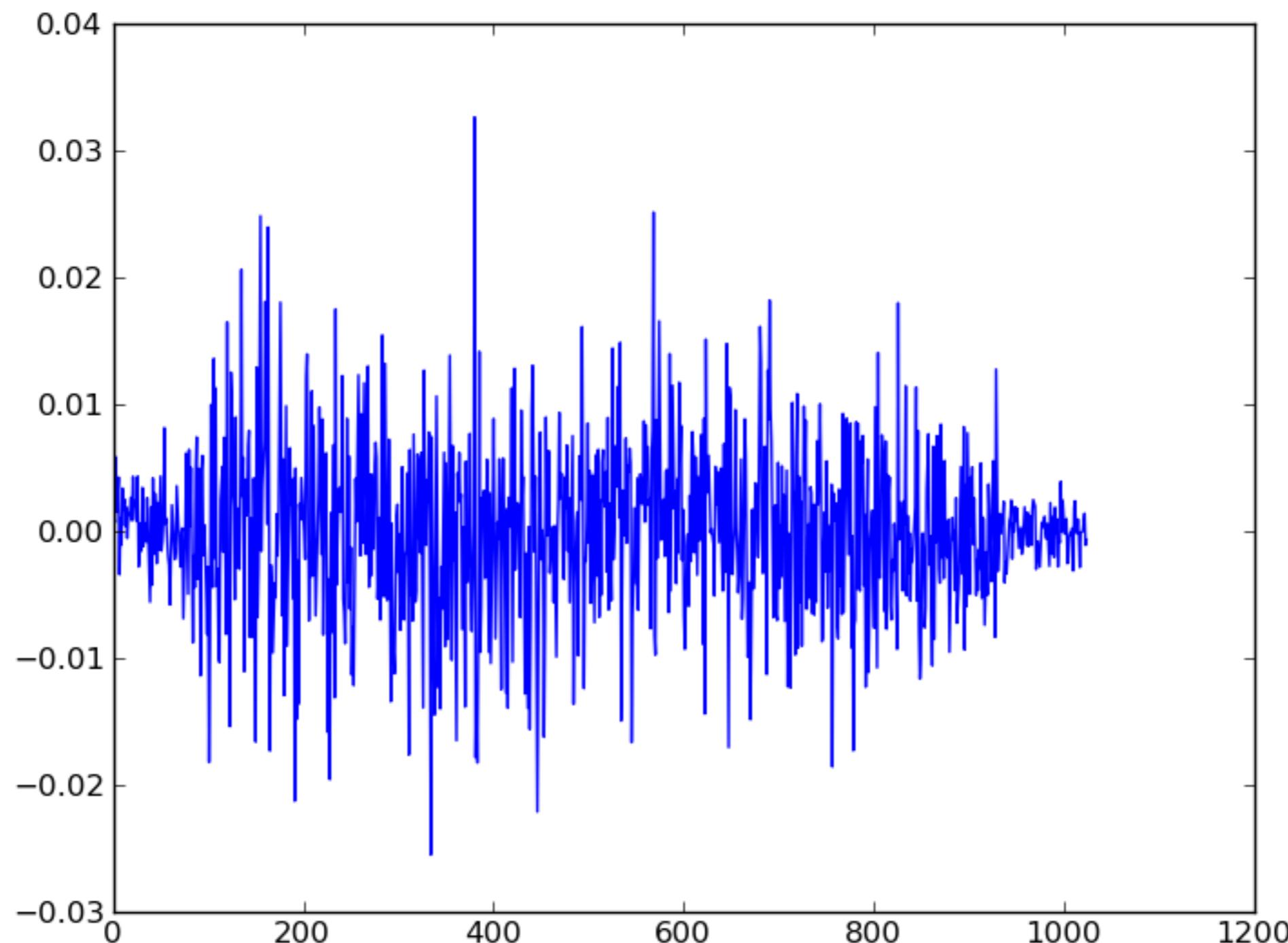
```
$> python
```

```
$> python
>>> import matplotlib.pyplot, pyrap.tables

>>> data = pyrap.tables.table('EG087A-unb.ms').query(
    'ANTENNA1=3 AND ANTENNA2=4 AND FIELD_ID=2
    AND DATA_DESC_ID=3').getcol('DATA')
Successful readonly open of default-locked table
EG087A-unb.ms: 22 columns, 1763520 rows

>>> matplotlib.pyplot.plot(data[0,:,:0])
[<matplotlib.lines.Line2D object at 0x20b2c10>]

>>> matplotlib.pyplot.show()
```

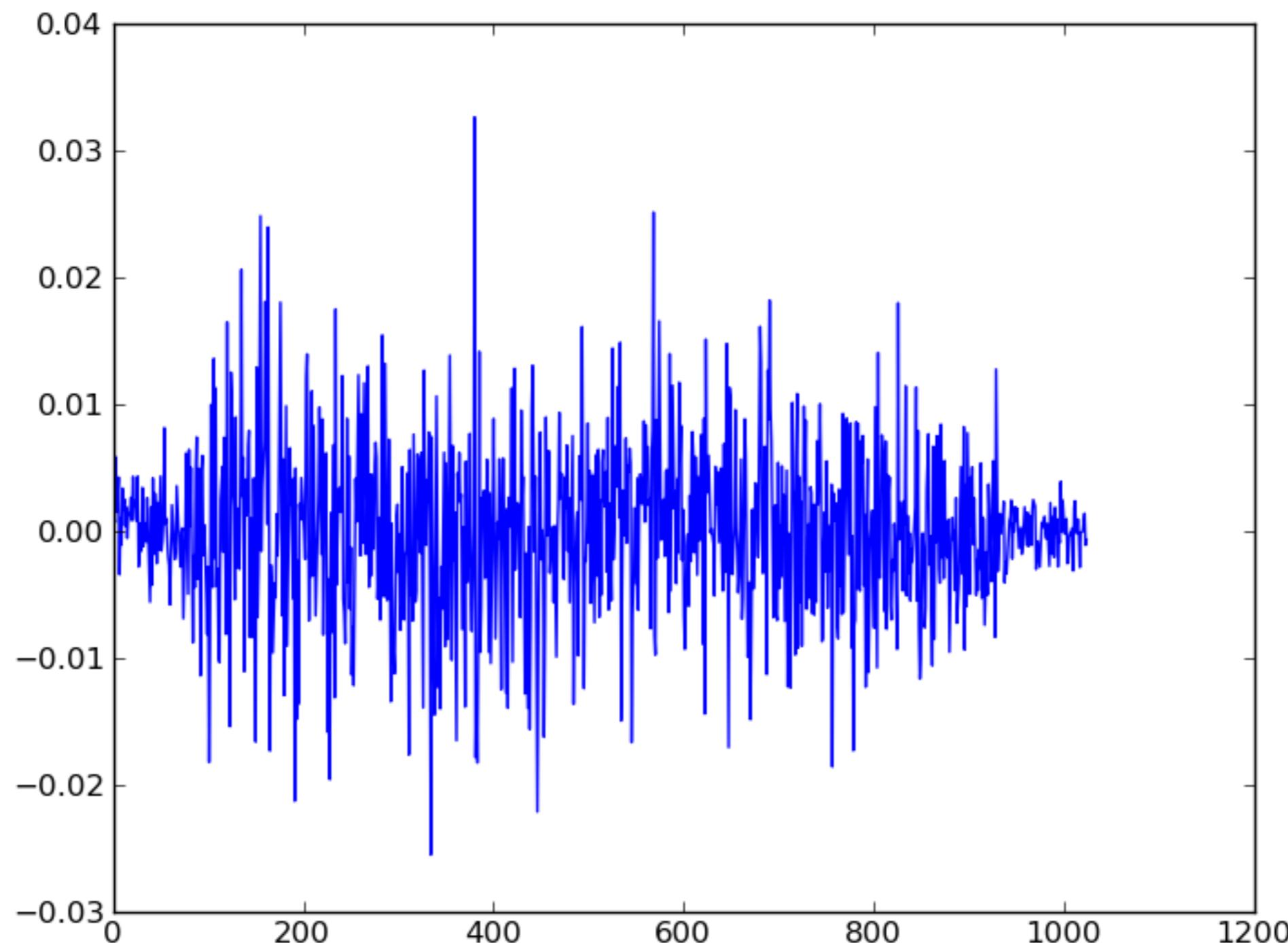


```
$> python
>>> import matplotlib.pyplot, pyrap.tables

>>> data = pyrap.tables.table('EG087A-unb.ms').query(
    'ANTENNA1=3 AND ANTENNA2=4 AND FIELD_ID=2
    AND DATA_DESC_ID=3').getcol('DATA')
Successful readonly open of default-locked table
EG087A-unb.ms: 22 columns, 1763520 rows

>>> matplotlib.pyplot.plot(data[0,:,:0])
[<matplotlib.lines.Line2D object at 0x20b2c10>]

>>> matplotlib.pyplot.show()
```



```
$> python
>>> import matplotlib.pyplot, pyrap.tables

>>> data = pyrap.tables.table('EG087A-unb.ms').query(
    'ANTENNA1=3 AND ANTENNA2=4 AND FIELD_ID=2
    AND DATA_DESC_ID=3').getcol('DATA')
Successful readonly open of default-locked table
EG087A-unb.ms: 22 columns, 1763520 rows

>>> matplotlib.pyplot.plot(data[0,:,:0])
[<matplotlib.lines.Line2D object at 0x20b2c10>]

>>> matplotlib.pyplot.show()
```

```
$> python
>>> import matplotlib.pyplot, pyrap.tables

>>> data = pyrap.tables.table('EG087A-unb.ms').query(
    'ANTENNA1=3 AND ANTENNA2=4 AND FIELD_ID=2
    AND DATA_DESC_ID=3').getcol('DATA')
Successful readonly open of default-locked table
EG087A-unb.ms: 22 columns, 1763520 rows

>>> matplotlib.pyplot.plot(data[0,:,:0])
[<matplotlib.lines.Line2D object at 0x20b2c10>]

>>> matplotlib.pyplot.show()
```

# jplotter

Python, MeasurementSet, Plotting



Harro Verkouter  
Joint Institute for VLBI - ERIC

```
$> ~/jiveplot/jplotter
+++++++
Welcome to cli ++++++
$Id: command.py,v 1.16 2015-11-04 13:30:10 jive_cc Exp $
'exit' exits, 'list' lists, 'help' helps
jcli>
```



```
○ ○ ○ 4. verkout@eee: /data1/verkout/ALMA (ssh)
[eee2]Okay->[ ]
```



# EG102E

weight versus time

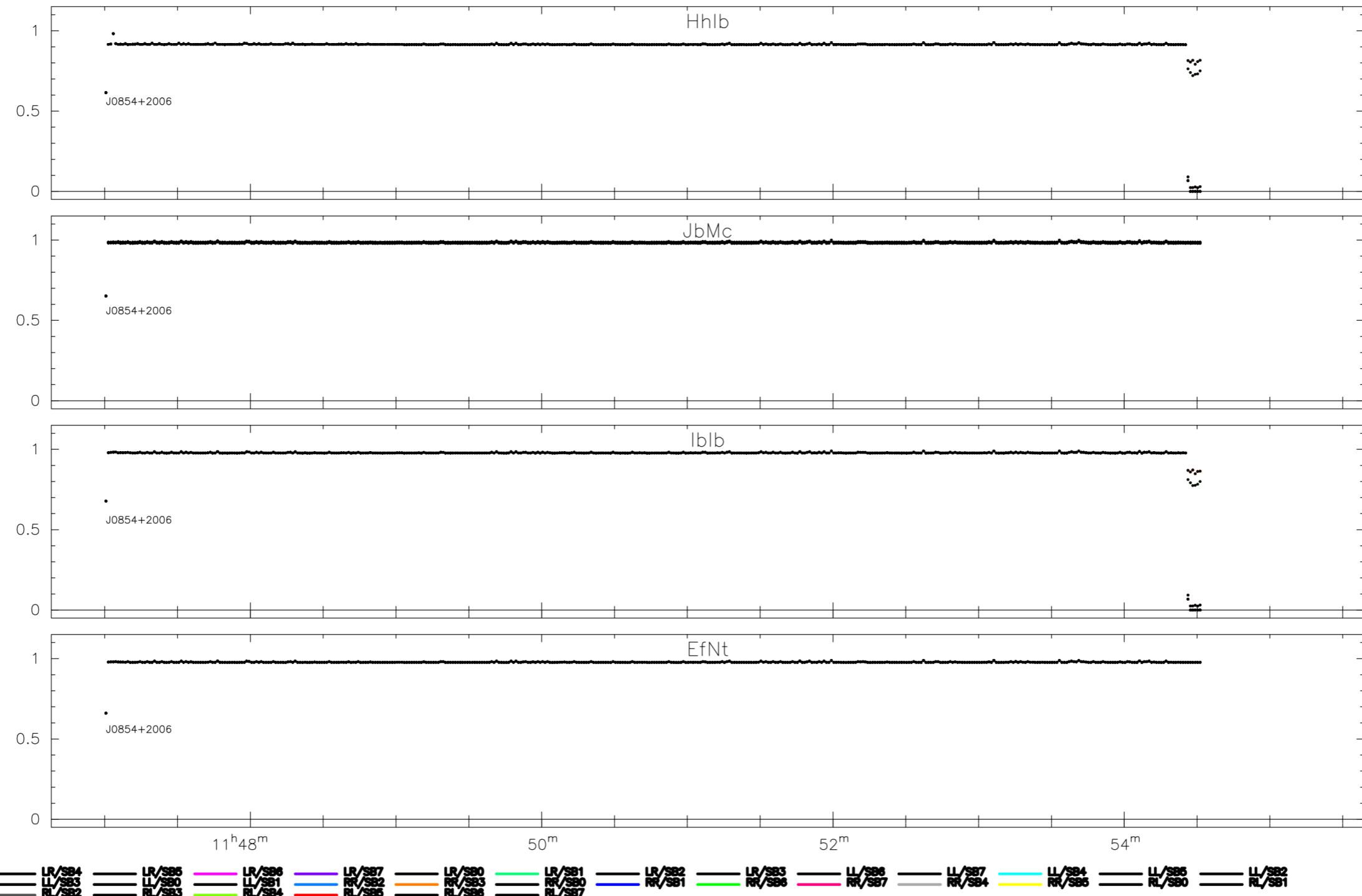
unique: sess218.C1024e/J0854+2006

Pol=RL,LL,LR,RR;Nsub=\*&;Ch=\*

data: 221471.ms [DATA]

verkout@<??> 2019-08-12T15:55:40

page: 1/7



weight versus time  
unique: sess218\_C102E\_J0854+2006  
Pol=RL,LL,LR,RR;Nsub=\*\*;;Ch=\*\*;

# EG102E

data: 221471.ms [DATA]  
verkouter@jive.eu 2019-08-12T15:55  
page: 1/7



jcli> r

```
jcli> r
listFreqs:      FREQID=0 [sess218.C1024e]
listFreqs:      SB 0: 4926.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 1: 4942.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 2: 4958.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 3: 4974.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 4: 4990.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 5: 5006.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 6: 5022.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 7: 5038.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listAntennas:   Ef ( 0) Jb ( 1) Mc ( 3) Nt ( 4) Ys ( 6) Hh ( 7) Ib ( 8)
listSources:    J0854+2006
listTimeRange:  19-Jun-2018/11:47:00.500 -> 19-Jun-2018/11:54:31.240  dT: 0.9
```

```
jcli> r
listFreqs:      FREQID=0 [sess218.C1024e]
listFreqs:      SB 0: 4926.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 1: 4942.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 2: 4958.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 3: 4974.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 4: 4990.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 5: 5006.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 6: 5022.5056MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listFreqs:      SB 7: 5038.4900MHz/16.0MHz nch=1024 P0=RR,LL,RL,LR
listAntennas:   Ef ( 0) Jb ( 1) Mg ( 3) Nt ( 4) Vs ( 6) Hh ( 7) Ib ( 8)
listSources:    T0854+2006
listTimeRange:  19-Jun-2018/11:47:00.500 -> 19-Jun-2018/11:54:31.240 dT: 0.9
```

Only 7 minutes

1 , 583 , 308 , 800

# Focus on data selection

# 7-dimensional data

Label	Is short for	Description
TIME	TIME	The time stamp of the data point
SRC	SOURCE	The name of the associated FIELD of this data point
BL	BASELINE	The baseline on which this data point was measured
FQ	FREQUENCY GROUP	The frequency group name (a.k.a. the frequency setup name or observing mode)
SB	SUBBAND	The subband number within the frequency setup
P	POLARIZATION	The (human readable) polarization combination of the data point
CH	CHANNEL	The channel number of the data point

note: the “r” command is short for “range along axes”

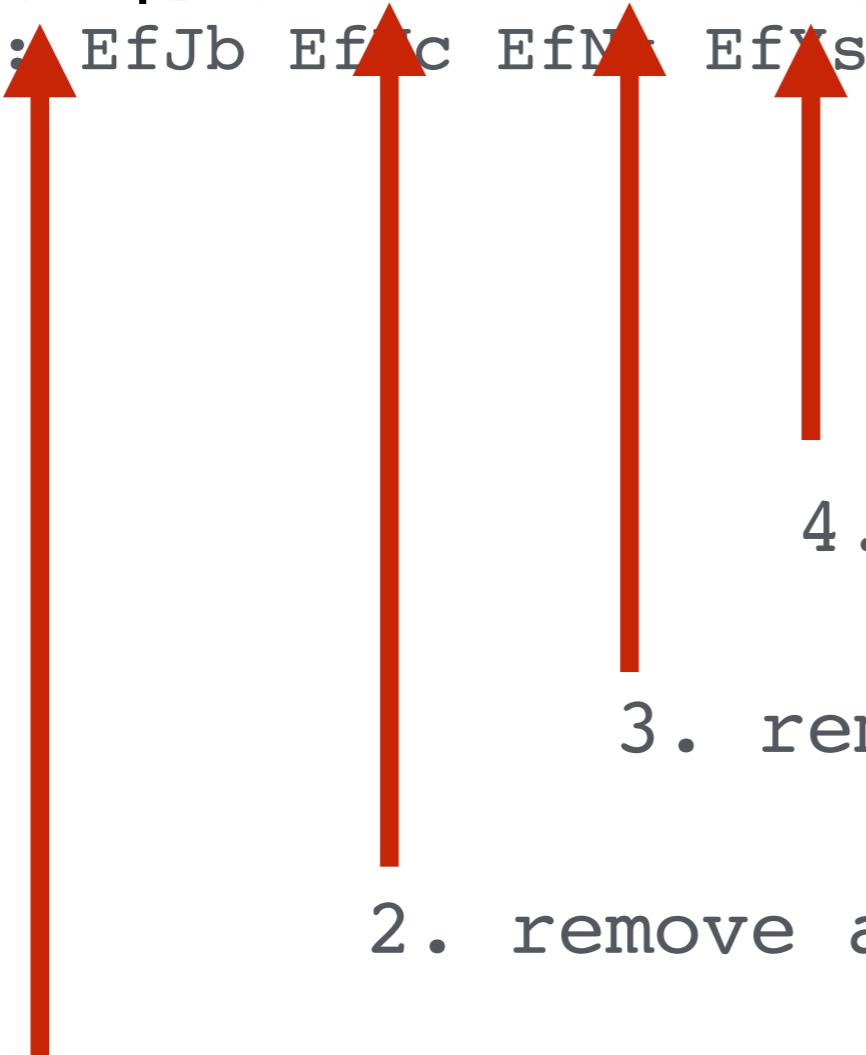
```
jcli> bl auto  
baselines: EfEf JbJb McMc NtNt YsYs HhHh IbIb
```

```
jcli> bl auto
```

```
baselines: EfEf JbJb McMc NtNt YsYs HhHh IbIb
```

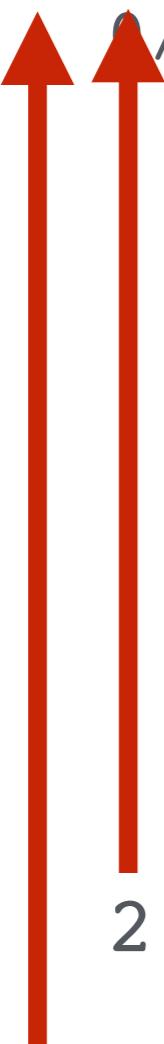
```
jcli> bl (ef|ys)* -auto -hh +(hh)ys
```

```
baselines: EfJb EfIc EfNt EfYs EfIb JbYs McYs NtYs YsHh YsIb
```



1. Select all baselines to **Ef** and **Ys**
2. remove all autocorrelations
3. remove baselines to **Hh**
4. but add **Hh-Ys** back in

```
jcli> fq */p  
freqsel: */0:7/0:RR,LL
```



1. Select **all subbands** (frequency bands)
2. and subselect all **parallel polarizations**

```
jcli> fq */p  
freqsel: 0/0:7/0:RR,LL
```

Same command also works for **XX**, **YY** polarization  
(or **HH**, **VV** if CASA will support that)

```
jcli> fq */p  
freqsel: 0/0:7/0:RR,LL
```

```
jcli> fq 1:3/rr,rl  
freqsel: 0/1:3/0:RR,RL
```

```
jcli> time 0/11:51:00 to 19-6-2018/11h52m0s
```

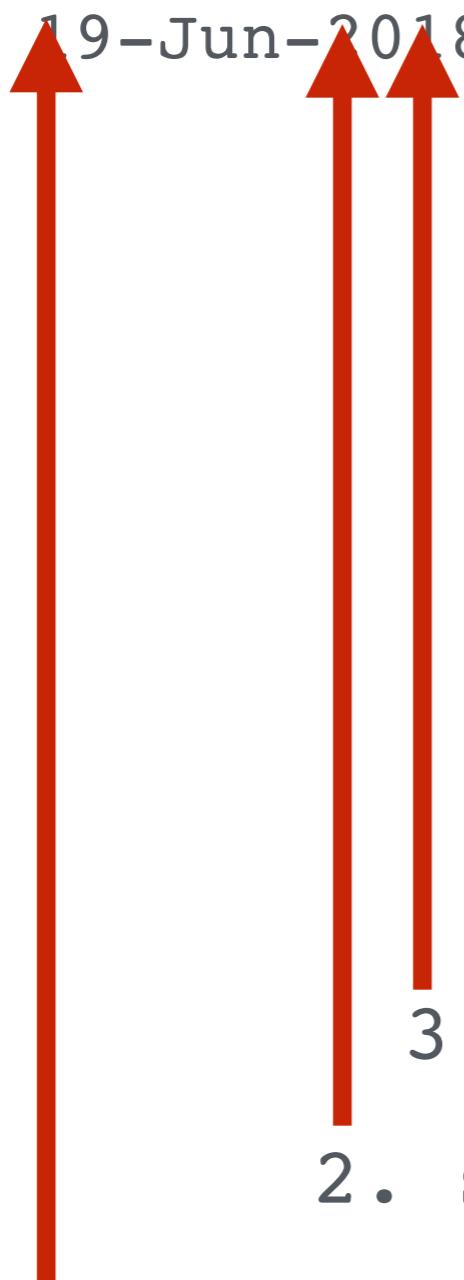
```
time: 19-Jun-2018/11:51:00.000 -> 19-Jun-2018/11:52:00.000
```



1. Day offset wrt day of observation
2. H:M:S time format
3. D-M-Y date format
4. 00h00m00s time format

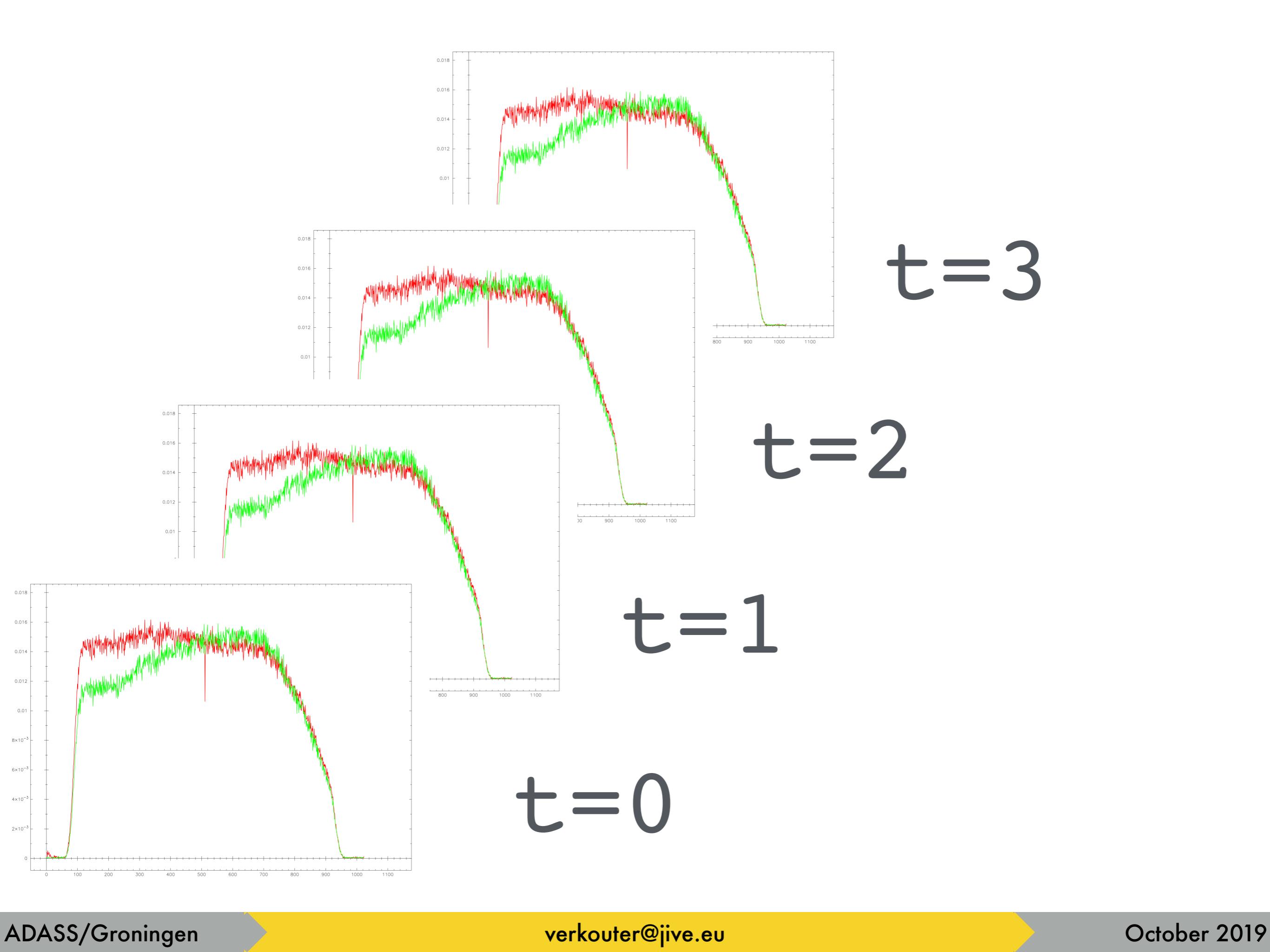
```
jcli> time $start + 1m10s to +30s
```

```
time: 19-Jun-2018/11:48:10.500 -> 19-Jun-2018/11:48:40.500
```

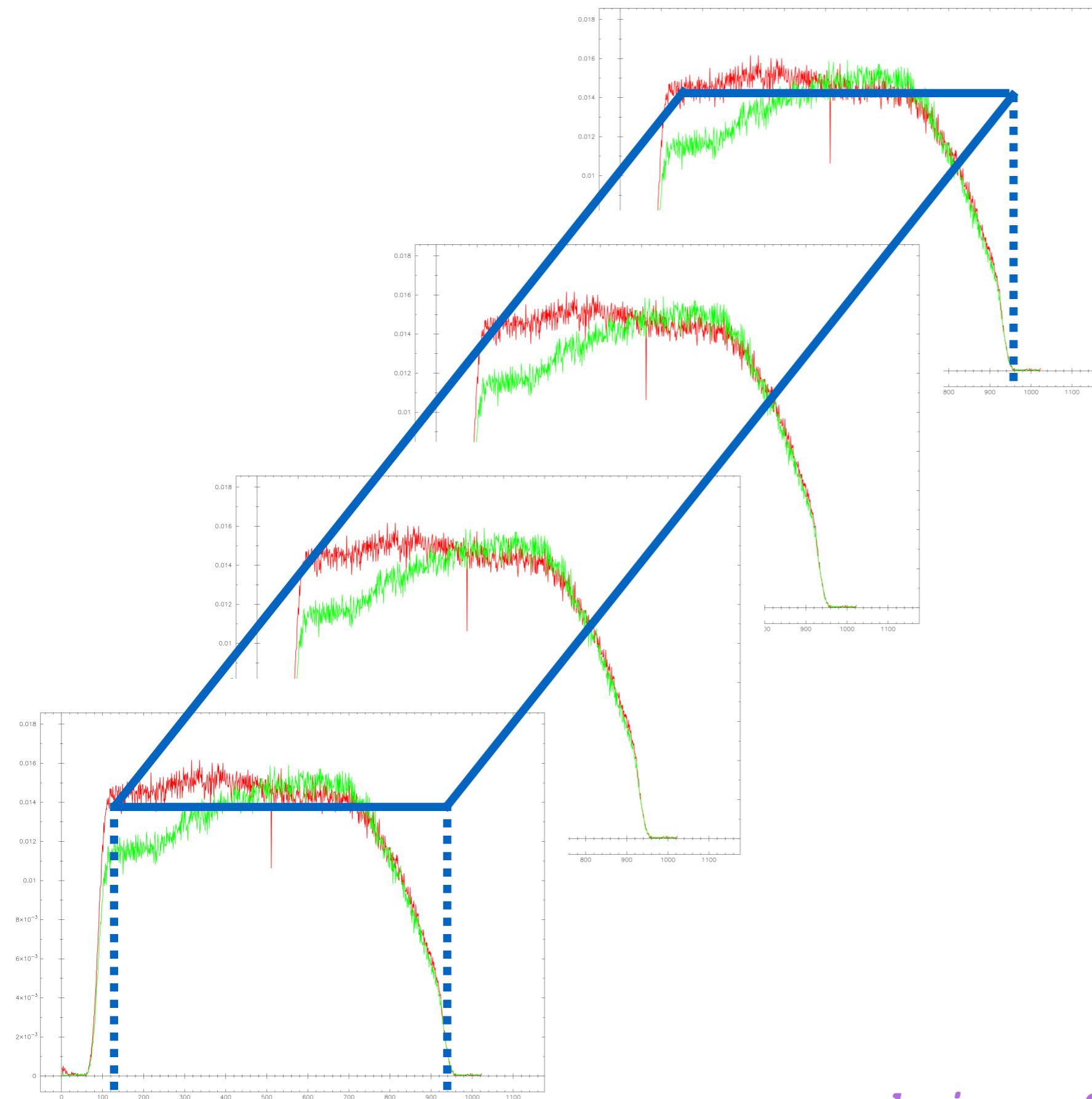


1. **\$start, \$end, \$mid** always dynamically computed from time range of current MS
2. support **arithmetic**
3. with **0d0h0m0.0s** time offsets

# Averaging

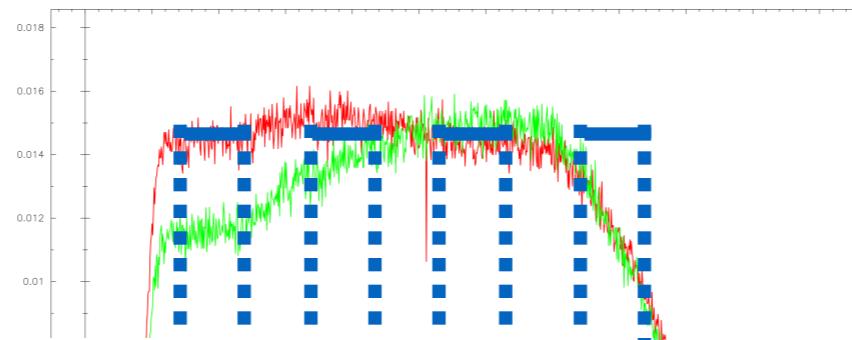


jcli> avt scalar

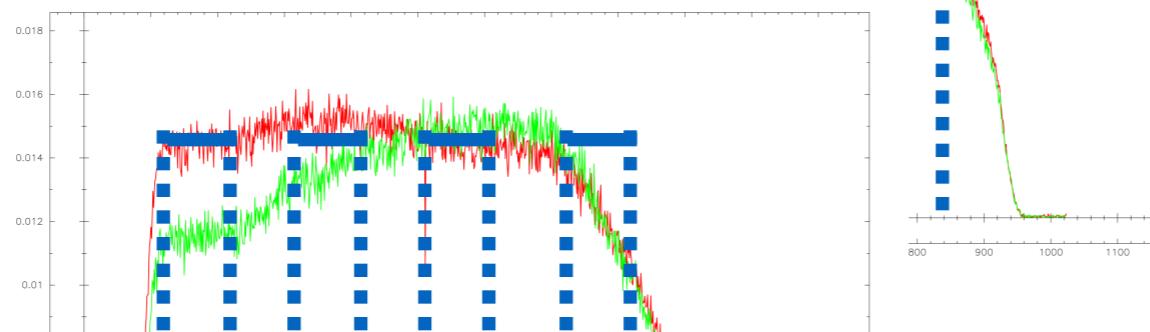


*avt = average time  
choice of "vector" or "scalar"*

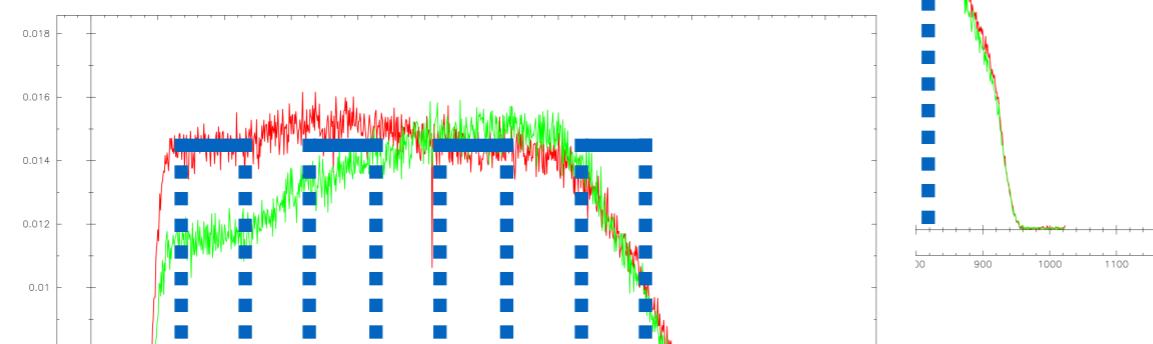
```
jcli> avc vector; nchav 2;
```



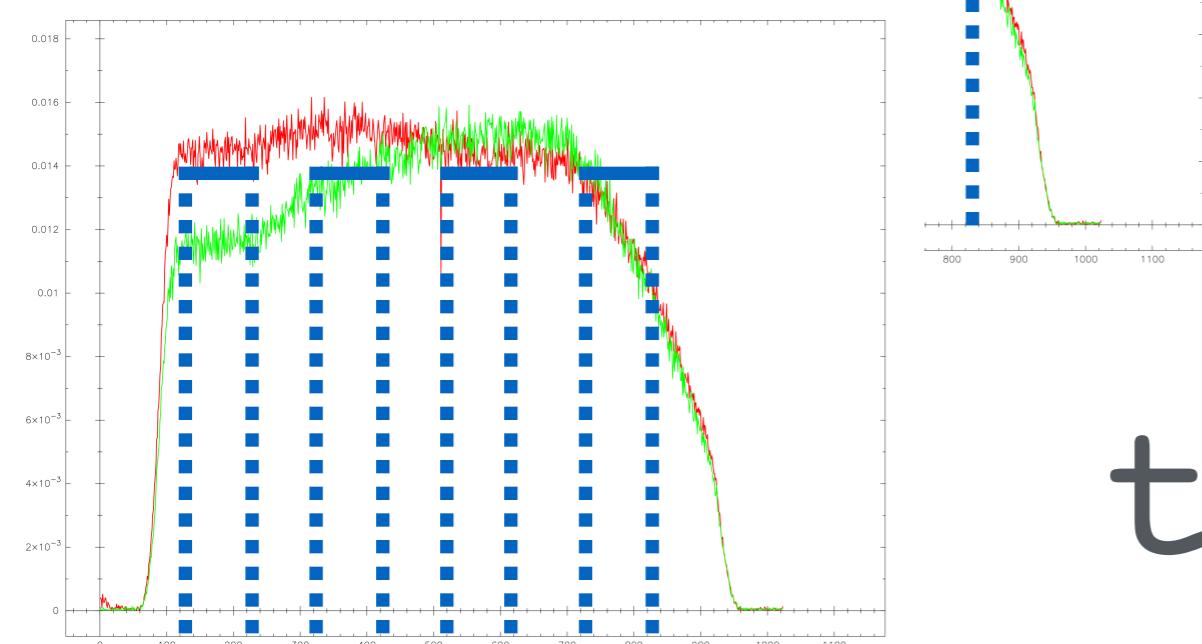
**t=3**



**t=2**

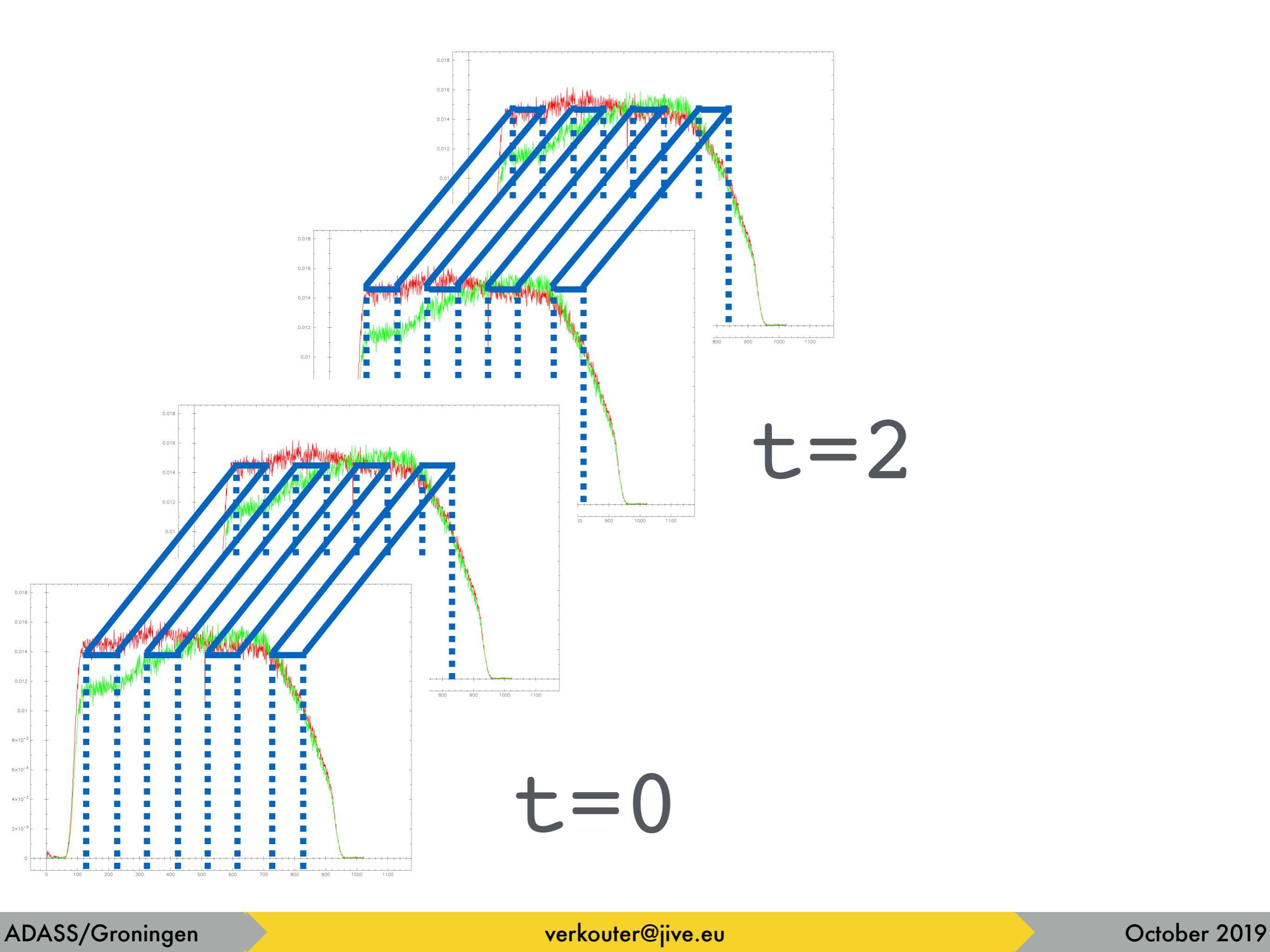


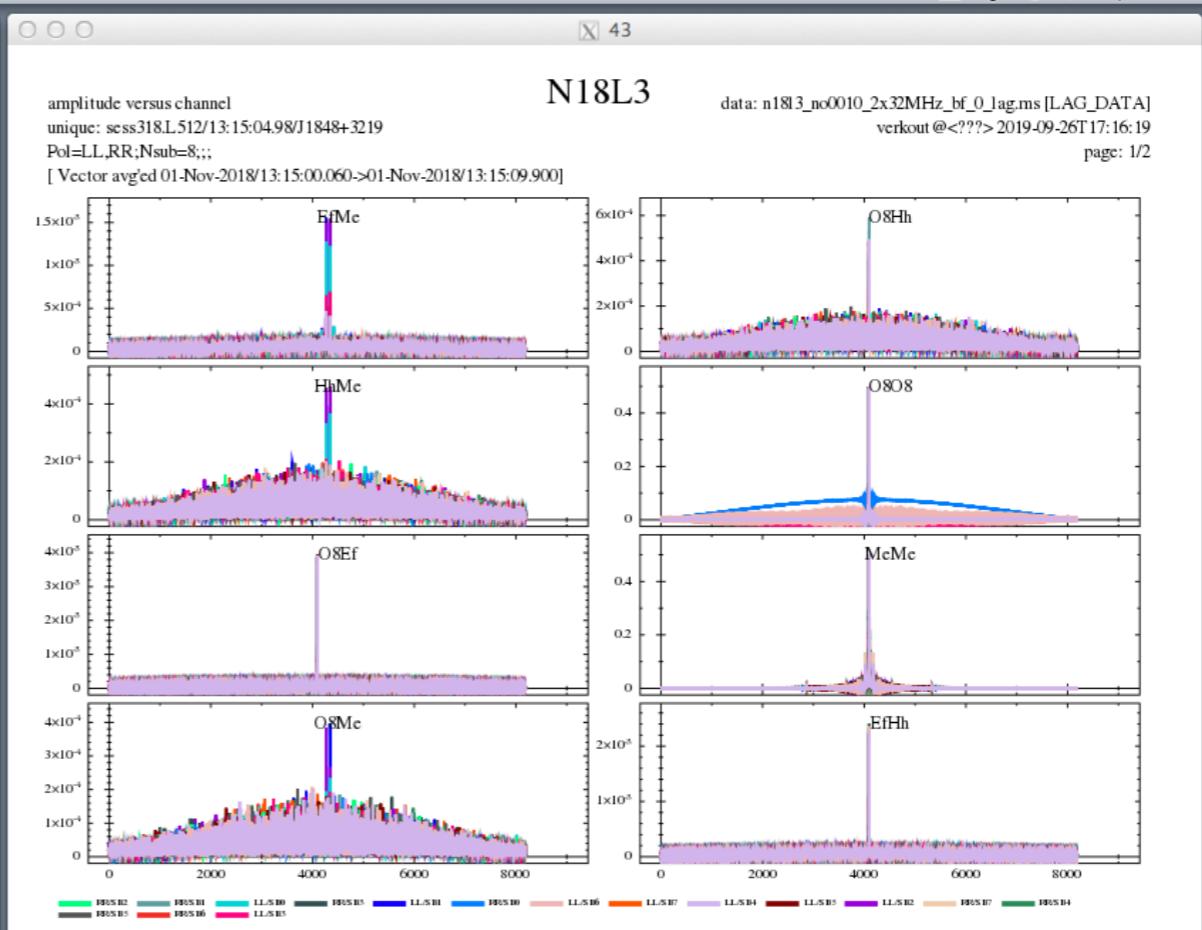
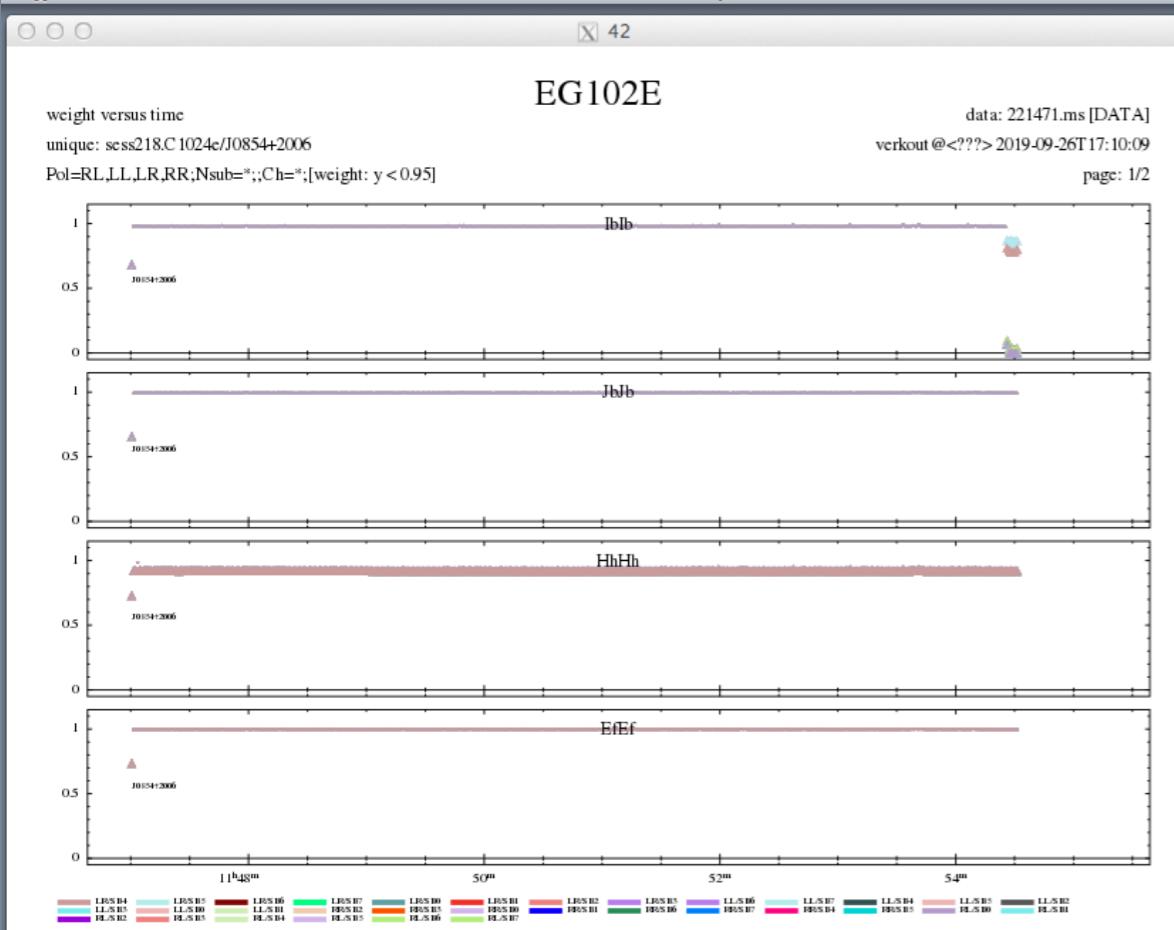
**t=1**



**t=0**

*avc = average channel  
choice of "vector" or "scalar"  
nchav = #-of-channels averaged*





jcli>

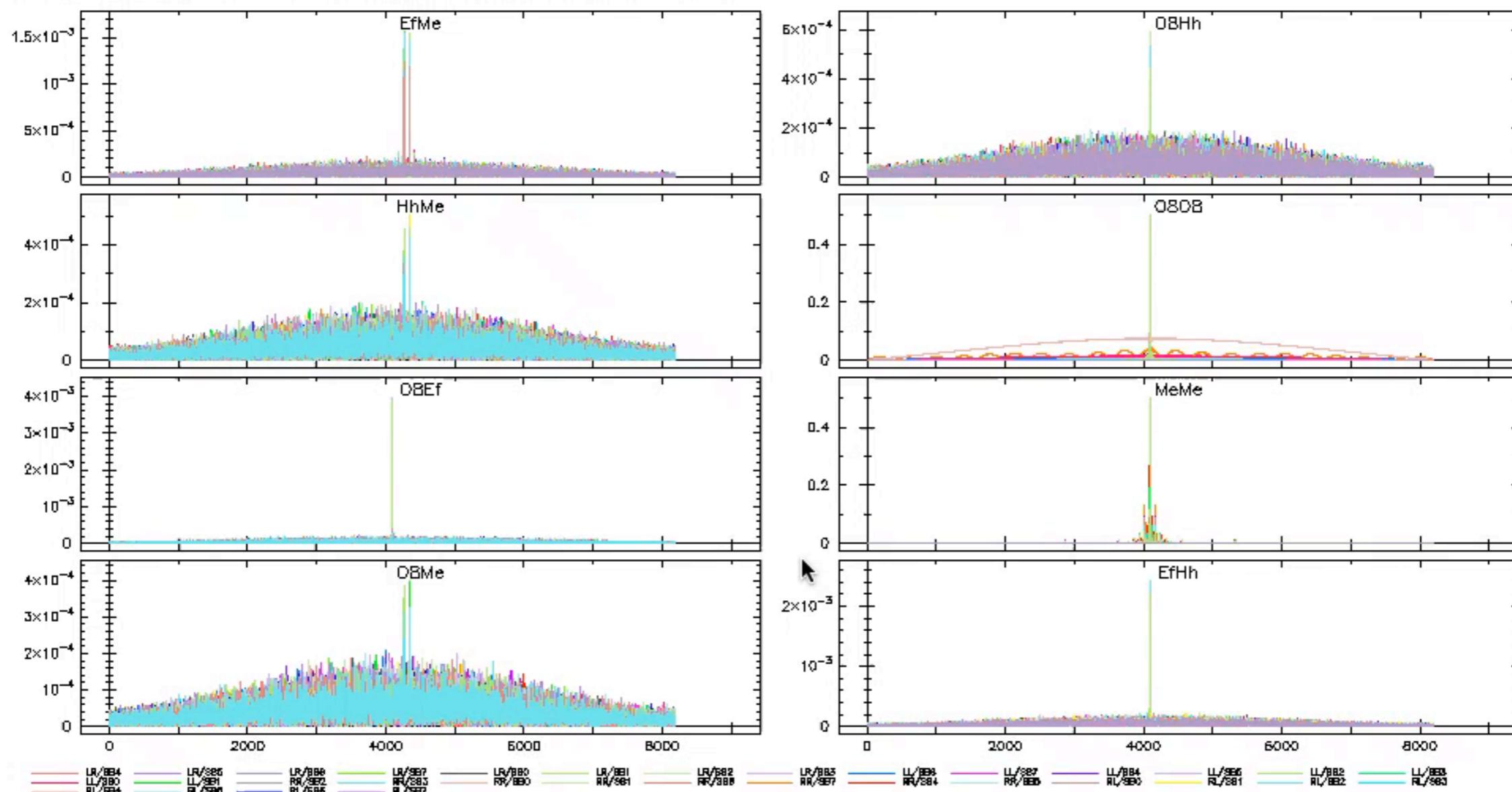
4. verkout@jop89:/export/jive/verkout (ssh)



amplitude versus channel  
 unique: sess318.L512/13:15:04.98/J1848+3219  
 Pol=RL,LL,LR,RR;Naub=\*\*;  
 [ Vector avg'ed 01-Nov-2018/13:15:00.060->01-Nov-2018/13:15:09.900]

N18L3

data: n18l3\_no0010\_2x32MHz\_bf\_0\_lag.ms [LAG\_DATA]  
 verkout@<??> 2019-09-27T10:32:36  
 page: 1/2



jcli>

4. verkout@jop89:/export/jive/verkout (ssh)

phase versus time

# EG087A

data: EG087A-onehour-sfxc.ms [DATA]

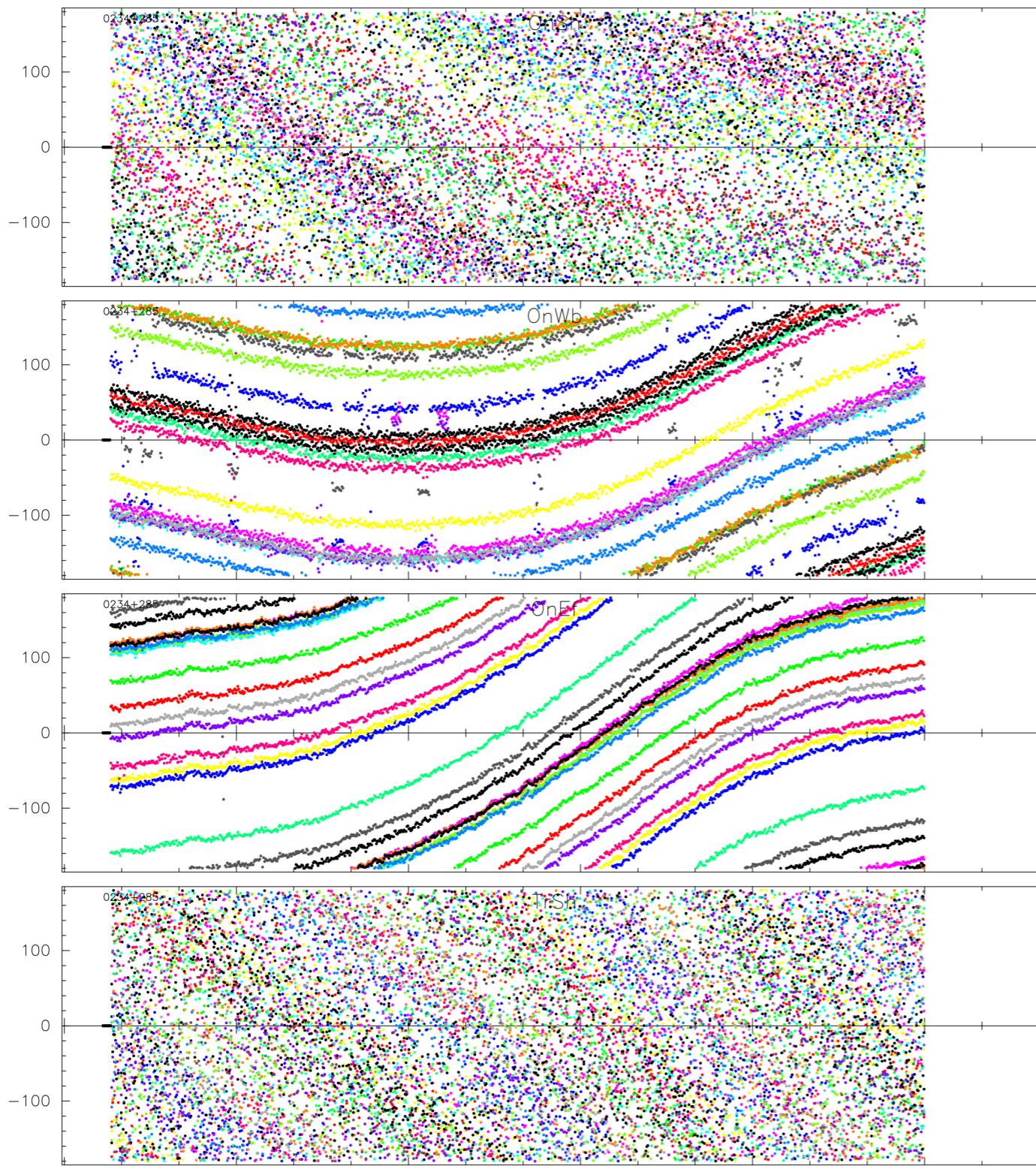
verkout@<??> 2019-09-27T16:03:54

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

page: 1/4

[Vectoraveraged channels 384:640]



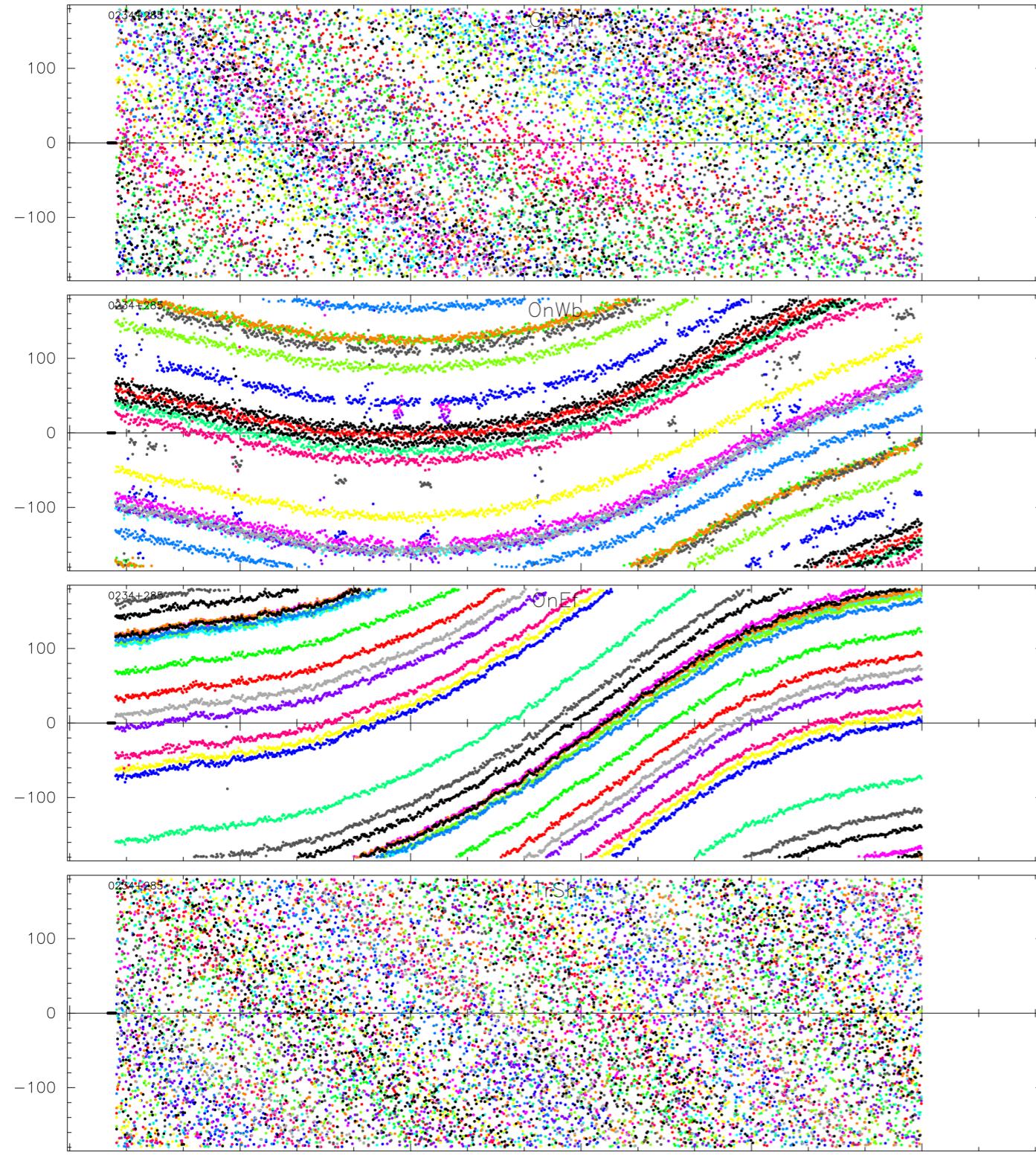
phase versus time

EG087A

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

[Vectoraveraged channels 384:640]



LL/SB6 RR/SB1 LL/SB0 LL/SB7 RR/SB5 RR/SB0 RR/SB2 RR/SB3 LL/SB4 LL/SB6 LL/SB3  
RR/SB4 RR/SB1 LL/SB1 LL/SB2 RR/SB7

data: EG087A-onehour-sfxc.ms [DATA] phase versus time

verkout@<??> 2019-09-27T16:03:54

page: 1/4

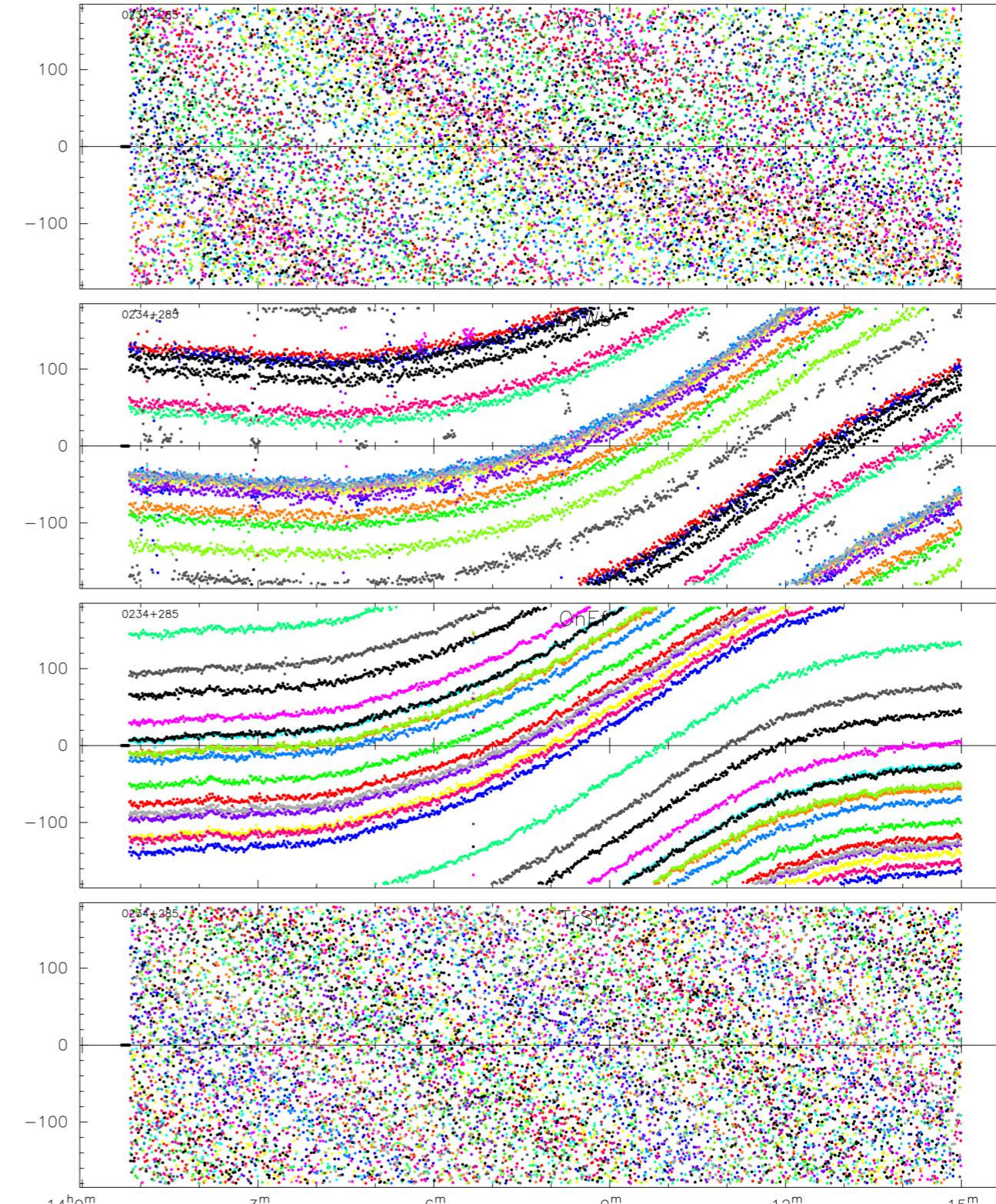
EG087A

data: EG087A-onehour.ms

verkout@<??> 2019-09-27T1

Pol=LL,RR;Nsub=8;;Ch=384:640;

[Vectoraveraged channels 384:640]



LL/SB6 RR/SB1 LL/SB0 LL/SB7 RR/SB5 RR/SB0 RR/SB2 RR/SB3 LL/SB4 LL/SB6 RR/SB6  
RR/SB4 LL/SB1 LL/SB2 RR/SB7

$$A = B \Rightarrow$$

$$A - B = 0$$

```
jcli> win 42
jcli> ms EG087A-onehour.ms
jcli> ...
# store all the plots into a variable
jcli> store as phatime_unb
```

phase versus time

# EG087A

data: EG087A-onehour-sfxc.ms [DATA]

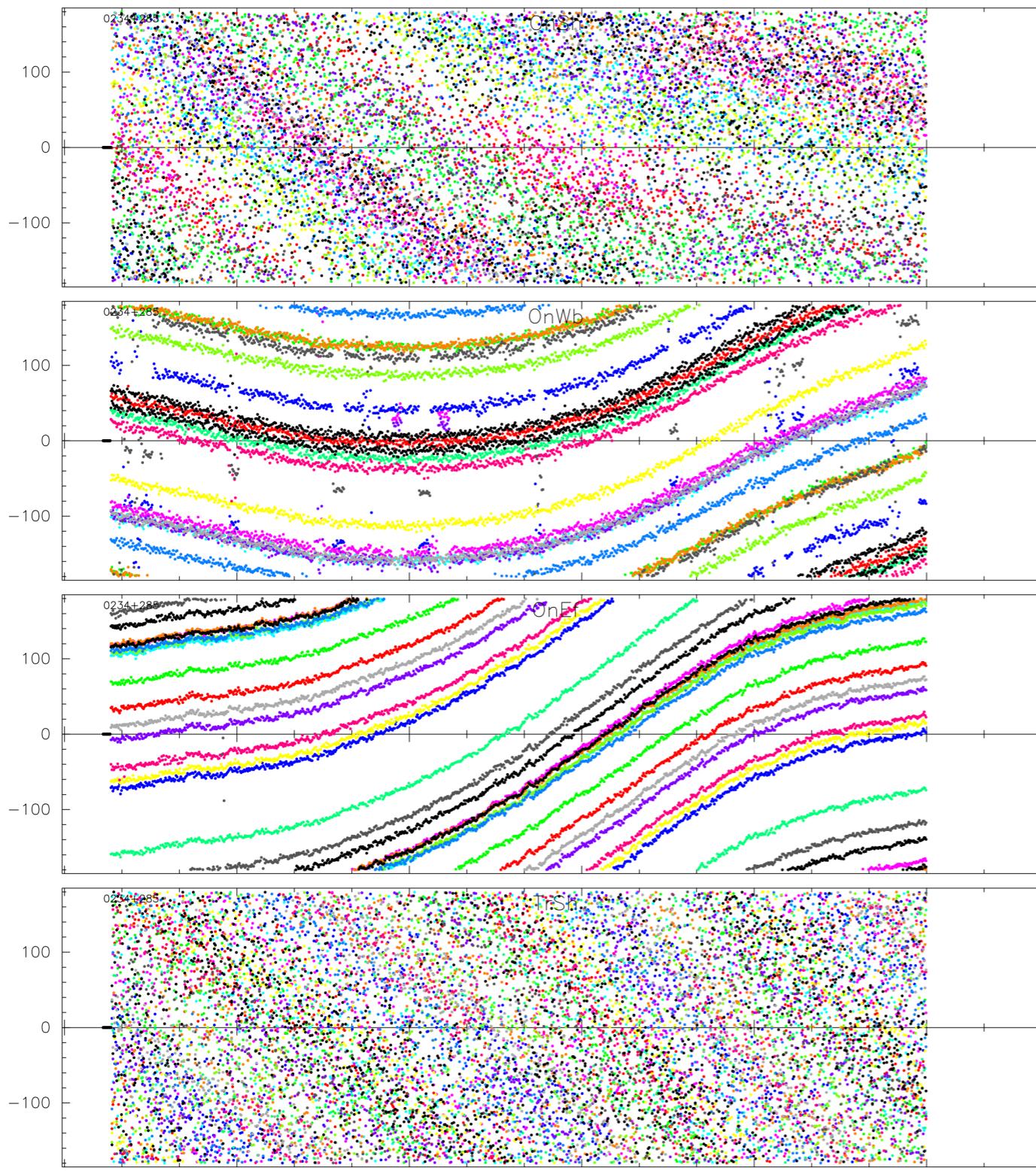
verkout@<??> 2019-09-27T16:03:54

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

page: 1/4

[Vectoraveraged channels 384:640]



*# Repeat for data from the other correlator*

```
jcli> win 43
jcli> ms EG087A-onehour-sfxc.ms
jcli> ...
jcli> store as phatime_sfxc
```

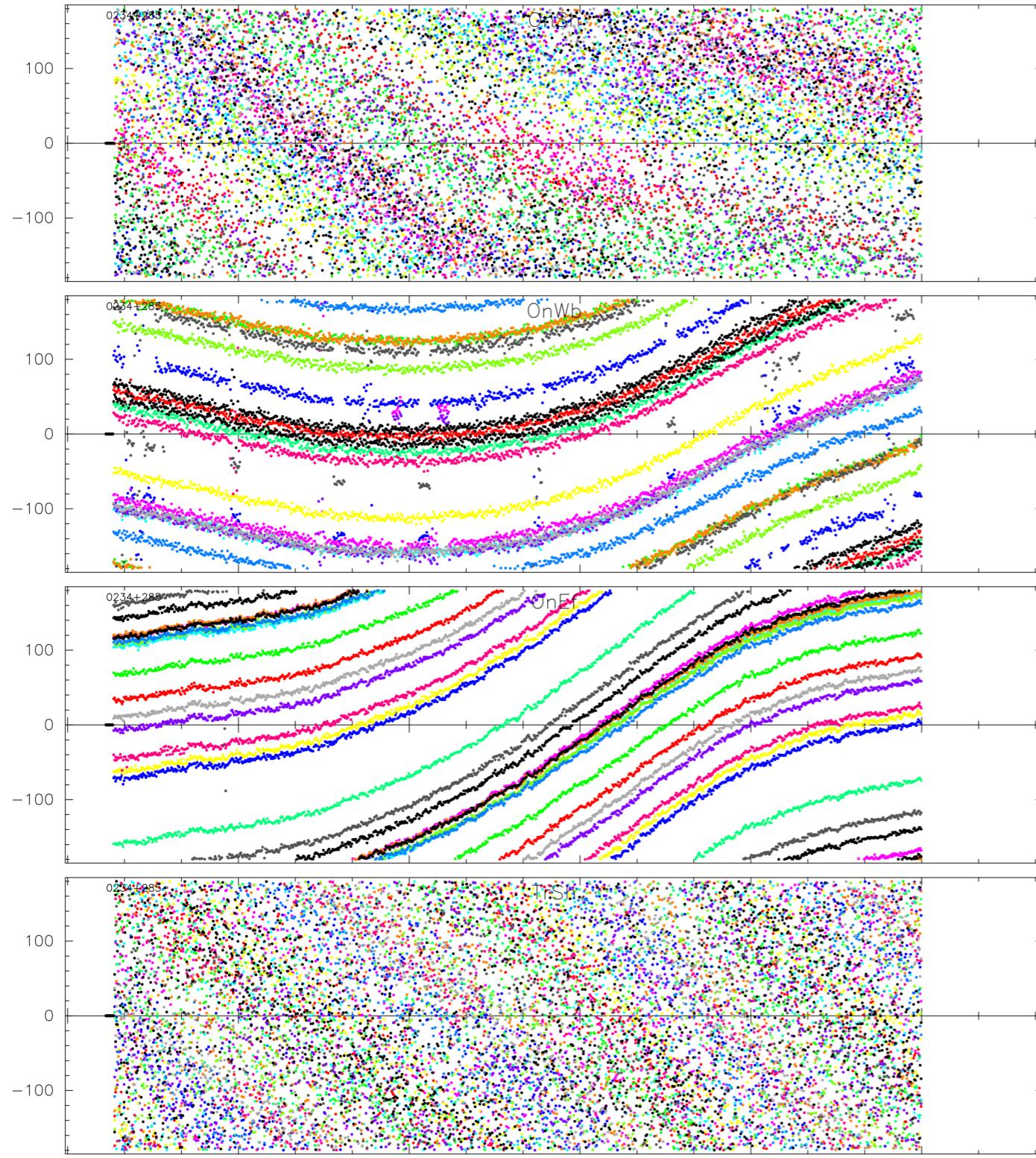
phase versus time

EG087A

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

[Vectoraveraged channels 384:640]



EG087A

data: EG087A-onehour-sfxc.ms [DATA] phase versus time

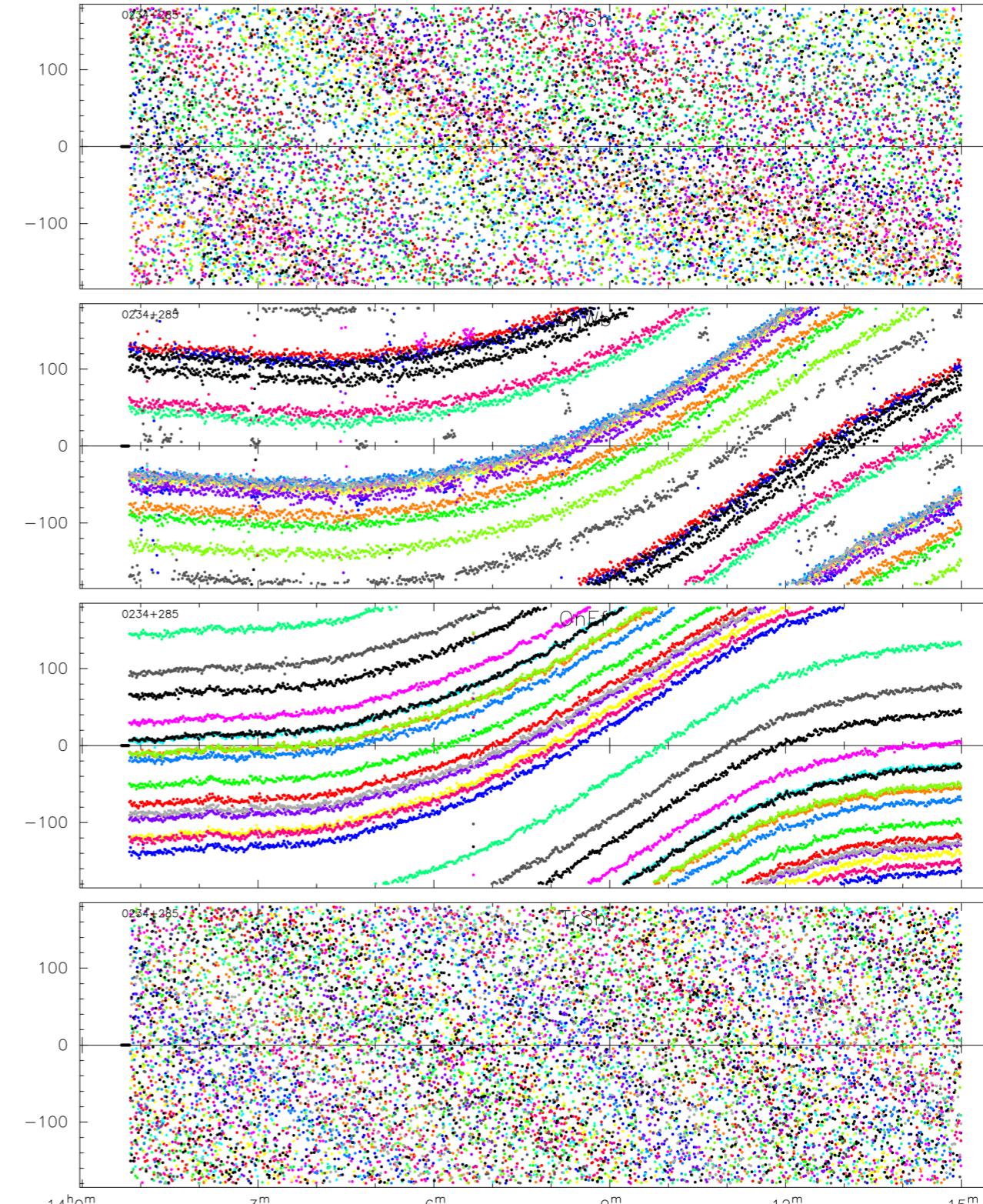
verkout@<??> 2019-09-27T16:03:54

page: 1/4

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

[Vectoraveraged channels 384:640]



data: EG087A-onehour.ms

verkout@<??> 2019-09-27T1

pa

*# Plot straight difference for all plots!*

```
jcli> load phatime_unb - phatime_sfxc
```

phase versus time

unique: sess115.L1024/CH\*/0234+285

Pol=LL,RR;Nsub=8;;Ch=384:640;

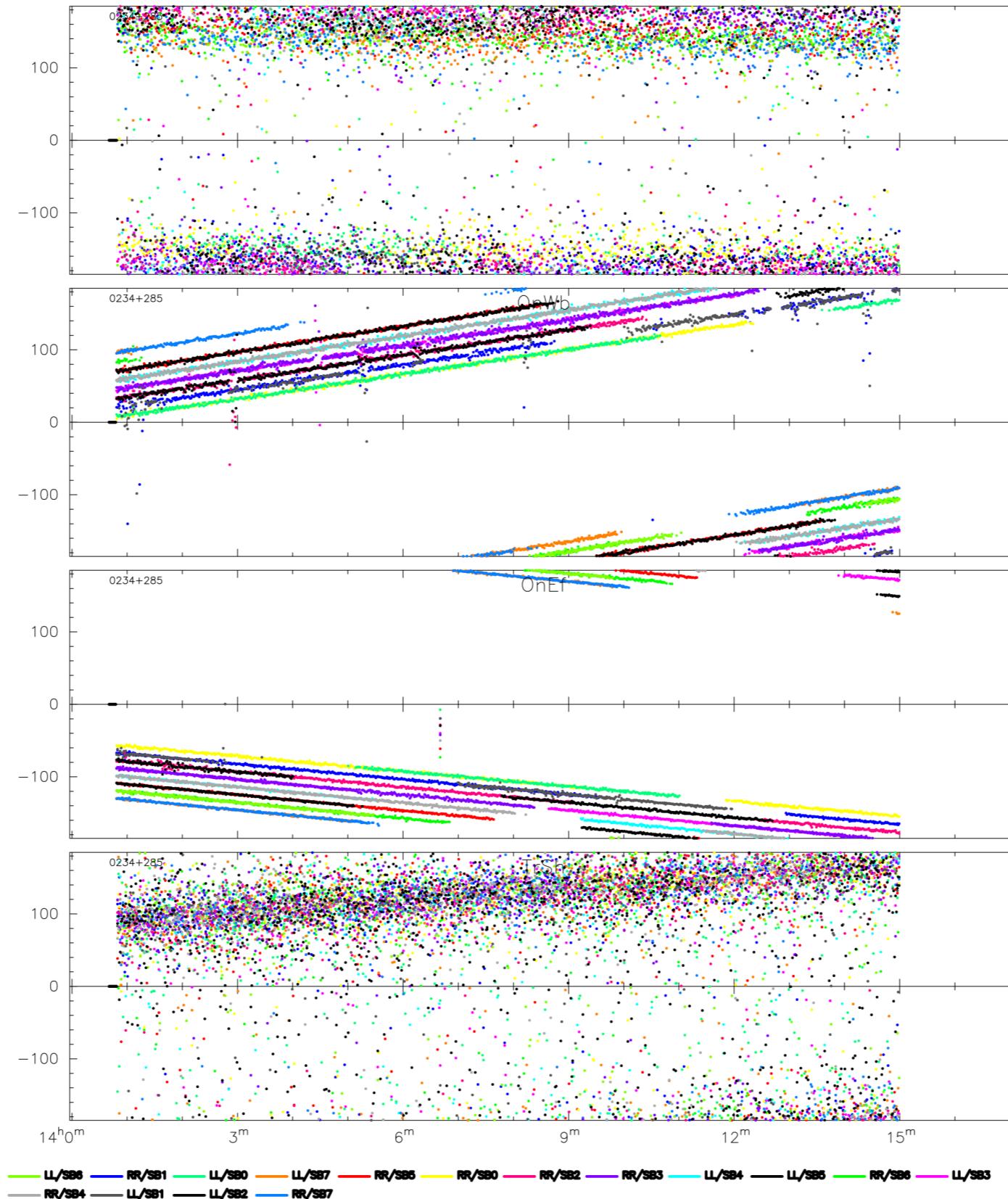
[Vectoraveraged channels 384:640]

EG087A

data: phatime\_unb - phatime\_sfxc [DATA]

verkout@<??> 2019-09-27T16:03:34

page: 1/4



# Built-in help

```
jcli> help mark
```

```
jcli> help mark
```

```
mark [index:] [expression]
```

mark points satisfying [expression] in a visually distinctive manner

Sometimes you want to be able to quickly find points satisfying a special condition, or just see IF there are data points satisfying a particular criterion. For those needs there is this 'mark' command.

Without arguments or just an index it will print the current marking expression for that y-axis type or for all of the y-axis types if no specific index was given. The colon separates an index from an (optional) expression.

The expression is a boolean condition which is evaluated for every data point. Points for which the evaluated expression yields 'True' are drawn in such a way they stand out w.r.t. the rest of the plotted points.

For plots where two quantities are plotted (e.g. "amplitude and phase versus ...") it is possible to specify a different marking condition per y-axis type.

```
jcli> help mark
```

```
mark [index:] [expression]
```

```
    mark points satisfying [expression] in a visually distinctive manner
```

Sometimes you want to be able to quickly find points satisfying a special condition, or just see IF there are data points satisfying a particular criterion. For those needs there is this 'mark' command.

Without arguments or just an index it will print the current marking expression for that y-axis type or for all of the y-axis types if no specific index was given. The colon separates an index from an (optional) expression.

The expression is a boolean condition which is evaluated for every data point. Points for which the evaluated expression yields 'True' are drawn in such a way they stand out w.r.t. the rest of the plotted points.

For plots where two quantities are plotted (e.g. "amplitude and phase versus ...") it is possible to specify a different marking condition per y-axis type.

```
jcli> help mark
```

...

Examples:

Display marking conditions for all data sets:

```
> mark
```

In a weight-versus-time plot (plottype 'wt'), mark all points that have a weight below 0.98:

```
> mark y<0.98
```

Mark all points that are more than 2.0 standard deviations away from the average:

```
> mark abs(y-avg)>2.0*sd
```

Or mark all points that have an x-axis value between 33 and 65 in the 'phase' sub plot of \*-and-phase-versus-\*:

```
> mark phase: x>33 and x<65
```

# N18L3

data: n18l3\_no0010\_2x32MHz\_bf\_0\_lag.ms [LAG\_DATA]

verkout@&lt;??&gt; 2019-09-25T16:33:27

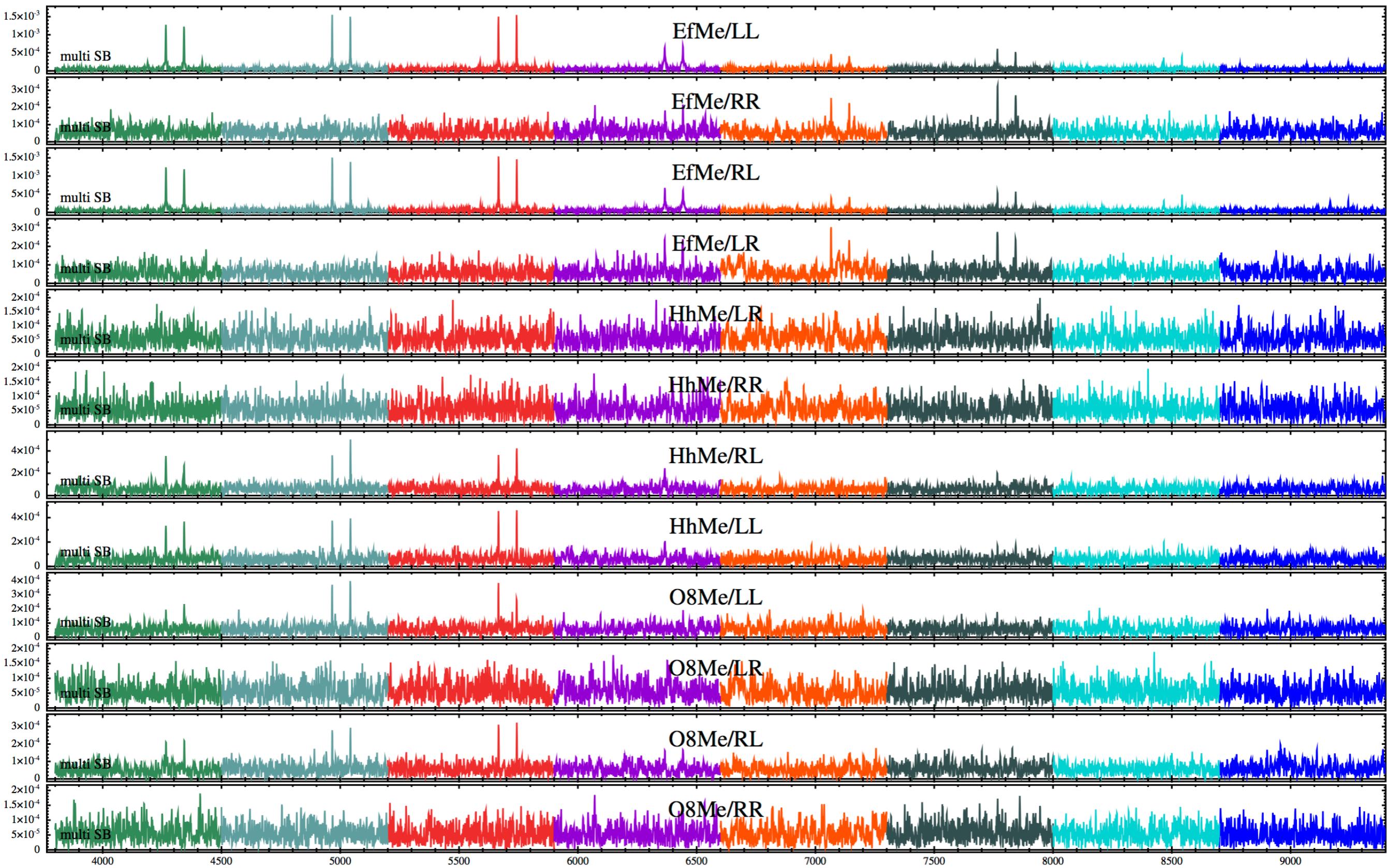
page: 1/1

amplitude versus channel

unique: sess318.L512/13:15:04.98/J1848+3219

Pol=RL,LL,LR,RR;Nsub=\*,;;;

[ Vector avg'ed 01-Nov-2018/13:15:00.060-&gt;01-Nov-2018/13:15:09.900]



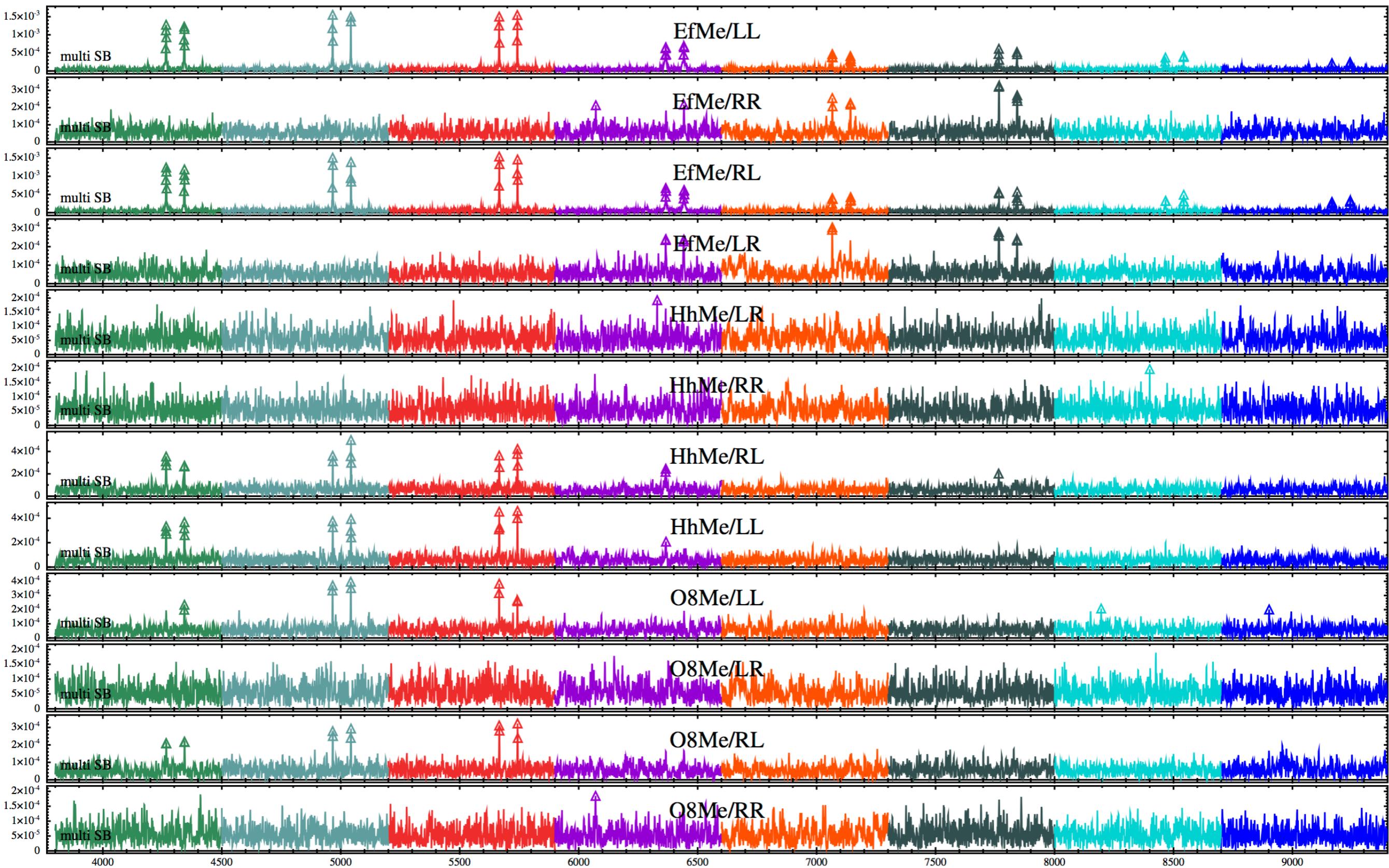
```
jcli> mark y > avg + 4.5*sd  
mark[ampchan/amplitude]: y > avg + 4.5*sd
```

amplitude versus channel

unique: sess318.L512/13:15:04.98/J1848+3219

Pol=RL,LL,LR,RR;Nsub=\*[;;][amplitude: y &gt; avg + 4.5\*sd]

[ Vector avg'ed 01-Nov-2018/13:15:00.060-&gt;01-Nov-2018/13:15:09.900]



- very flexible organization
  - separate data into plots/datasets
  - sorting of plots, layout on screen
  - programmable data set colouring
- handles flagged data
  - under all combinations of averaging
  - options to (not) read/show/hide flagged data
- insert own Python postprocessing func
  - after data read, before display, dynamically loaded
- automation methods
  - defining macros
  - play scriptfile w/ arguments
  - read commands from Python generator/list
- output to file
  - PGPLOT backend: PostScript
  - Giza backend: PDF, PNG, PostScript
- batch plotting direct to file (no X11)
- write out current selection as new MS

For all discussed commands below the reader is cordially invited to read the online help inside *jplotter* for the command of interest. The **help <command>** tells all.

## time

The **time** command allows selection of (multiple) time range(s) of interest. The symbolic constants **\$start**, **\$mid**, **\$end** and **\$t\_int** (which dynamically evaluate to values for the current data set) can be used together with arithmetic and human readable time formats. This allows for natural-looking time selection commands like **time \$start to +1hr10m** to select only the first hour-and-ten-minutes from the experiment. Also check the **scan** command further down for convenient scan-based selections. Multiple time ranges are separated by “,” (a comma):

```
jcli> time $start to + 5 * $t_int , $end - 1h to +20m
time: 06-Mar-2014/12:30:00.500 -> 06-Mar-2014/12:30:05.500
time: 06-Mar-2014/14:29:59.481 -> 06-Mar-2014/14:49:59.481
```

## bl

The baseline selection command **bl** was already introduced earlier. It supports the pseudo baseline names **auto** and **cross** which select what one would imagine. “All baselines to Ef” becomes **bl ef\***. Or try this: “all baselines to either Jb or Hh”: **bl (jb|hh)\***

The **bl** command supports multiple arguments. Each argument is a ‘selector’ and selects one or more baselines. The arguments are evaluated from left-to-right and a set of selected baselines is built. Each selector can add (default) or subtract baselines from the set so far. All cross-baselines to Ef would easily be selected as **bl ef\* -auto**. The minus sign indicates to remove baselines matching the selector from the set.

## src

The **src** source selection command is much like the **bl** command: it compiles a set of sources (de)selected by the argument(s) to the command, evaluated from left-to-right. Being a text based selection mechanism, common UNIX shell wildcards (“\*” and “?”) are supported: **src j19\*** selects all sources whose name start with ‘j19’ (case insensitive). A limited form of regular expression syntax is supported: **src 3c(18|192)** selects both sources 3C18 and 3C192.

## fq

This is the main subband/polarization selection command. Subbands are numbered *0..n* (as relabelled by *jplotter*<sup>4</sup>) within a frequency setup or group. For most data sets it will present the frequency information in a natural manner, to wit the output of the **r** command.

A bit was unveiled earlier when only the parallel polarization combinations for all subbands were needed. The very short **fq \*/p** effected just that. The **fq** command also accepts multiple arguments to select multiple, specific, subband+polarization combinations. For example **fq 1,2/rr 5:7/1\*** would select the RR polarization from subbands 1 and 2 and LL and LR from subbands 5 through 7 (inclusive).

## ch

The frequency selection is completed by the **ch** command, to select channels inside all selected subbands (see the **fq** command above). Channels are numbered *0..n*. The pseudo values **first**, **mid** and **last** are defined, which can be used in simple arithmetic expressions. The author’s favourite is the magic

**ch 0.1\*last:0.9\*last** which selects the inner 80% of the channels of each band, or this one: **ch mid-2:mid+2** to select just a few channels around the center.

<sup>4</sup> Read the help for both the **r** and **ms** commands, specifically about how meta data is dealt with in *jplotter* and what you can do to alter the behaviour in case it is not appropriate. ALMA Measurement Sets do come to mind.

## ptsz lnew

If the default point-size when drawing the plots with points or the width of the lines when drawing with lines are not to taste, add a bit extra using the **ptsz** or **lnew** commands. The argument to both commands is just a number; the requested point size/line width (this should, however, not come as a total surprise).

## y[01] x

*jplotter*’s default behaviour is to give each plot the same scale such that the plots can be directly visually compared. Under certain circumstances this may be undesirable, e.g. when both auto- and cross-correlation spectra are plotted.

Using the **x** and **y** command the scaling of the indicated axis of the plots can be set to either of the three values as described in the table below. In multi-panel plots (e.g. amplitude and phase versus \*), the scale of the individual panels can be set using the **y0** command for the bottom panel or **y1** for the top panel.

The scaling settings are kept on a per-plot type basis.

### global

The scale of each panel is set to fit the global minimum and maximum over all data sets displayed in all plots.

### local

Each panel will be scaled to fit the minimum and maximum of all data sets displayed in that panel.

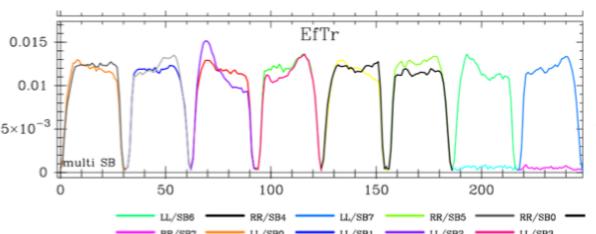
### <min> <max>

Set an explicit scale by manually providing a minimum and maximum value. All panels get this scaling.

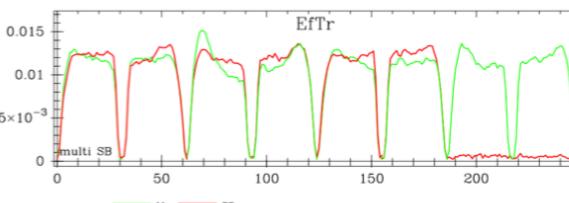
## ckey

Another command that combines a mini-language and label types. *jplotter*’s default data set colouring algorithm is quite simple: each unique data set label gets its own color. Again, not always is this the best choice. Specifically for those occasions *jplotter* allows (very) fine grained control over the data set colouring algorithm via the **ckey** (colour key) command. If desired specific values can be given a specific colour. The syntax is documented in the **ckey** help documentation.

The two plots below should give an idea of what the **ckey** command does. The same plot as under the description of the **multi** command is used.



Each data set - a combination of a subband and a polarization - has its own colour. Presented like this, the colours per subband do not add information whilst at the same time it is difficult to see which polarization(s) fail in subbands 6 and 7.



After issuing the following command:  
`jcli> ckey p`  
the plot looks like this. This instructs *jplotter* to colour the data sets by **p**(olarization) only and the affected polarization is readily identified.

<https://github.com/haavee/jiveplot>

<https://github.com/haavee/jiveplot>

## Dependencies

Python binding to MeasurementSet

<https://github.com/casacore/python-casacore>

Python PGPLOT interface

<https://github.com/haavee/ppgplot>

Can be linked against:

- PGPLOT (the fast FORTRAN version)
- Giza (slower but very nice graphics)  
<https://github.com/danieljprice/giza>

<https://github.com/haavee/jiveplot>

## Dependencies

Python binding to MeasurementSet

<https://github.com/casacore/python-casacore>

Python PGPLOT interface

<https://github.com/haavee/ppgplot>

Can be linked against:

- PGPLOT (the fast FORTRAN version)
- Giza (slower but very nice graphics)  
<https://github.com/danieljprice/giza>

## Also available on

Docker: <https://hub.docker.com/r/haavee/jiveplot/>

Singularity: <shub://haavee/jiveplot>

Danke für Ihre  
Aufmerksamkeit

```
jcli> lp
```

Known plot-types:

```
phatime => phase versus time  
rnichan => real+imag versus channel  
imchan => imag versus channel  
ampfreq => amplitude versus frequency  
uv      => V versus U  
imtime  => imag versus time  
phafreq  => phase versus frequency  
anpfreq  => amplitude+phase versus frequency  
amptime  => amplitude versus time  
anptime  => amplitude+phase versus time  
ampchan  => amplitude versus channel  
rechan   => real versus channel  
wt       => weight versus time  
retime   => real versus time  
anpchan  => amplitude+phase versus channel  
ampuv    => amplitude versus UV distance  
rnitime  => real+imag versus time  
phachan  => phase versus channel
```

```
jcli> lp
```

Known plot-types:

**phatime => phase versus time**

rnichan => real+imag versus channel

imchan => imag versus channel

ampfreq => amplitude versus frequency

**uv => v versus U**

imtime => imag versus time

phafreq => phase versus frequency

**anpfreq => amplitude+phase versus frequency**

amptime => amplitude versus time

anptime => amplitude+phase versus time

ampchan => amplitude versus channel

rechan => real versus channel

wt => weight versus time

retime => real versus time

**anpchan => amplitude+phase versus channel**

**ampuv => amplitude versus UV distance**

rnitime => real+imag versus time

phachan => phase versus channel

```
$> python
>>> import pyrap.tables

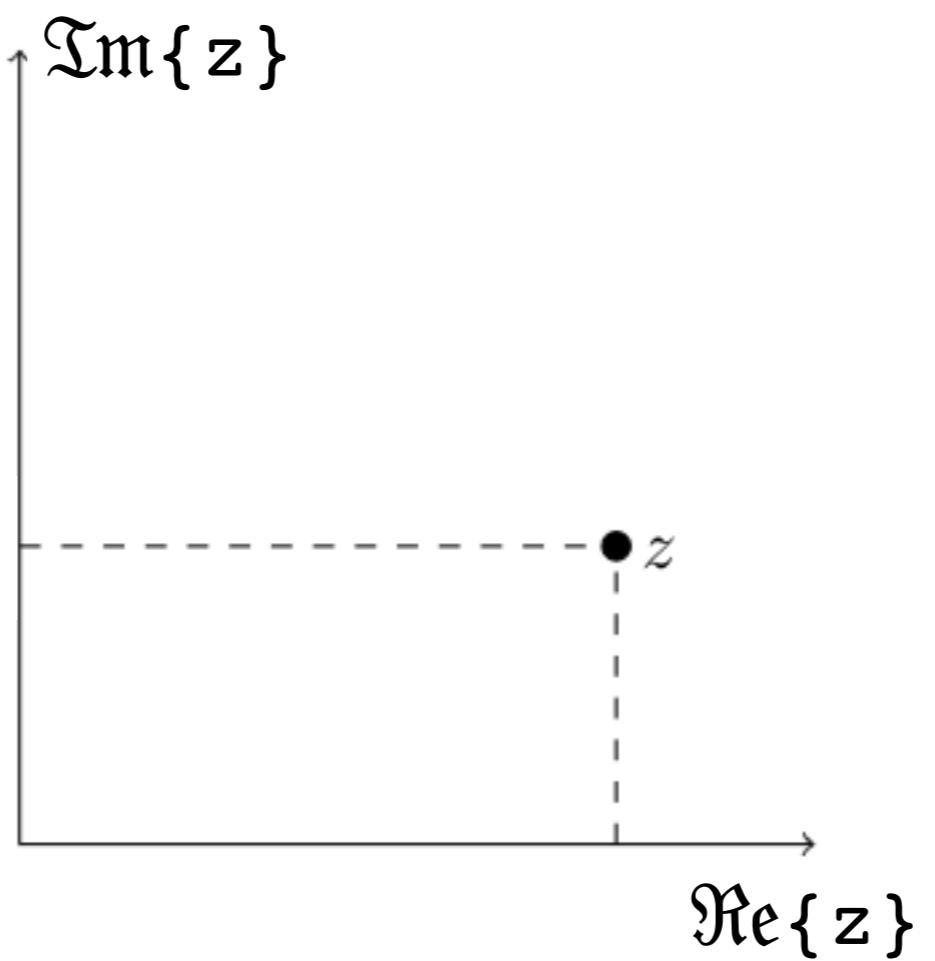
>>> ms = pyrap.tables.table('221471.ms')
Successful readonly open of default-locked table 221471.ms: 22 columns, 101248 rows
```

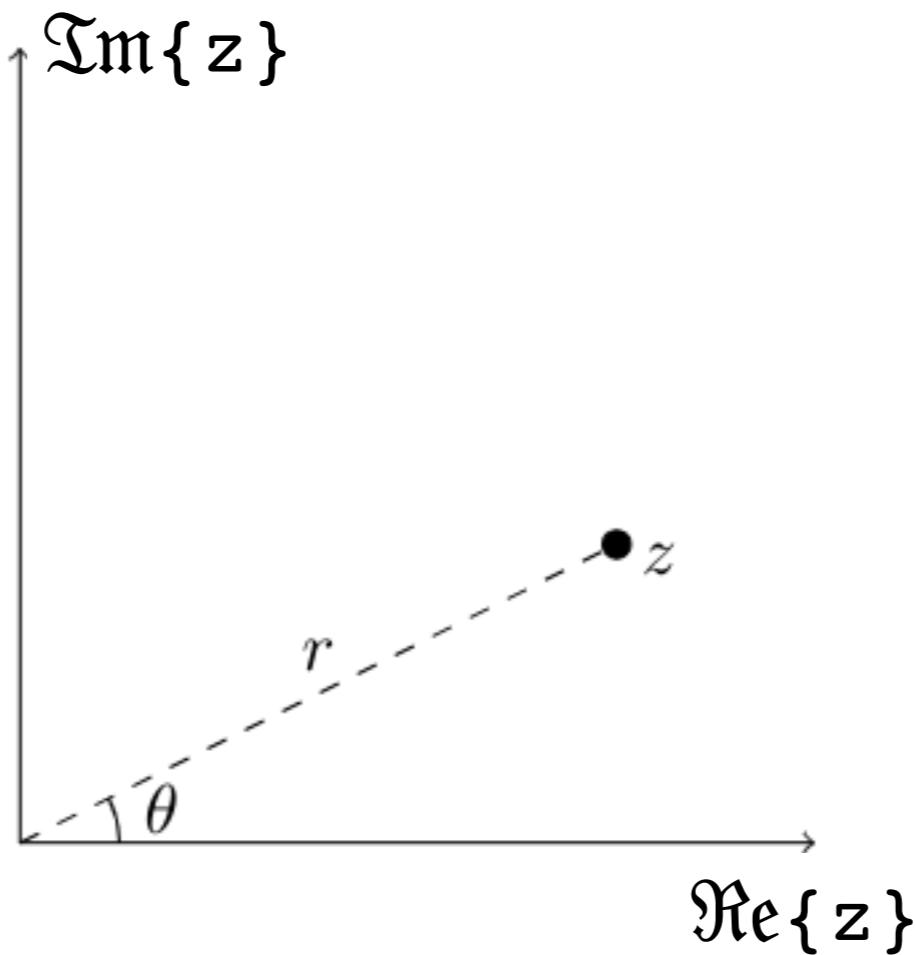
```
# Access row #0, column 'DATA' - a numpy array
>>> ms[0]['DATA'].shape
(1024, 4)
```

```
>>> ms[0]['DATA'].dtype
dtype('complex64')
```

old module: **pyrap**  
new module: **python-casacore**

<https://github.com/casacore/python-casacore>



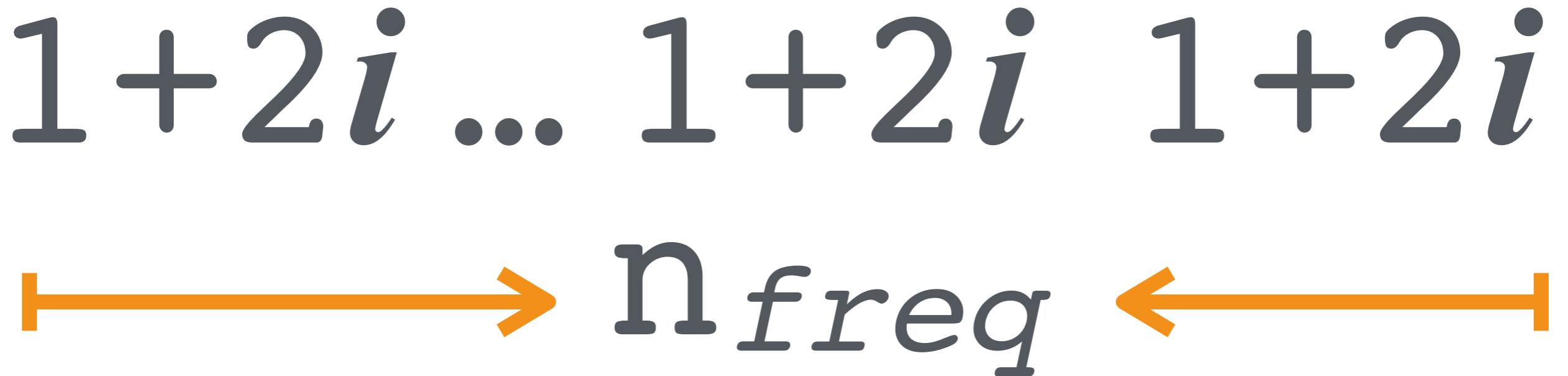


$r$  = “**amplitude**” =  $\sqrt{(\Re e^2 + \Im m^2)}$

$\theta$  = “**phase**” =  $\tan^{-1}(\Im m / \Re e)$

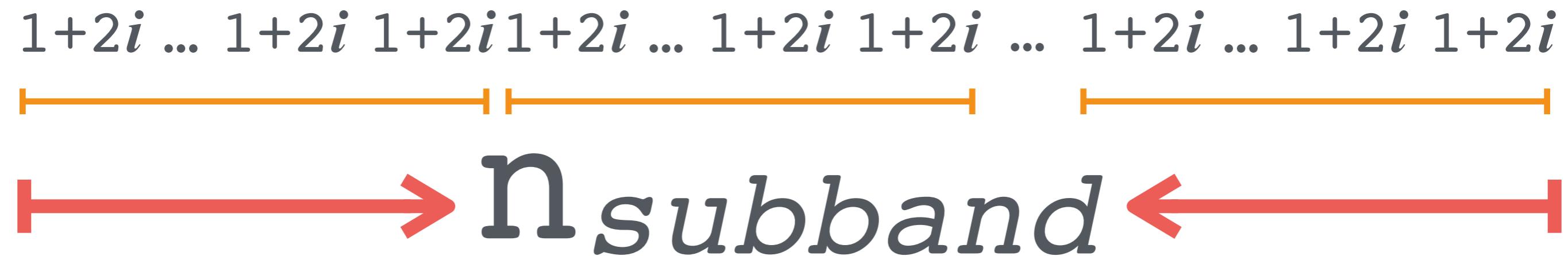
$1+2i \dots 1+2i \quad 1+2i$

$n_{freq}$



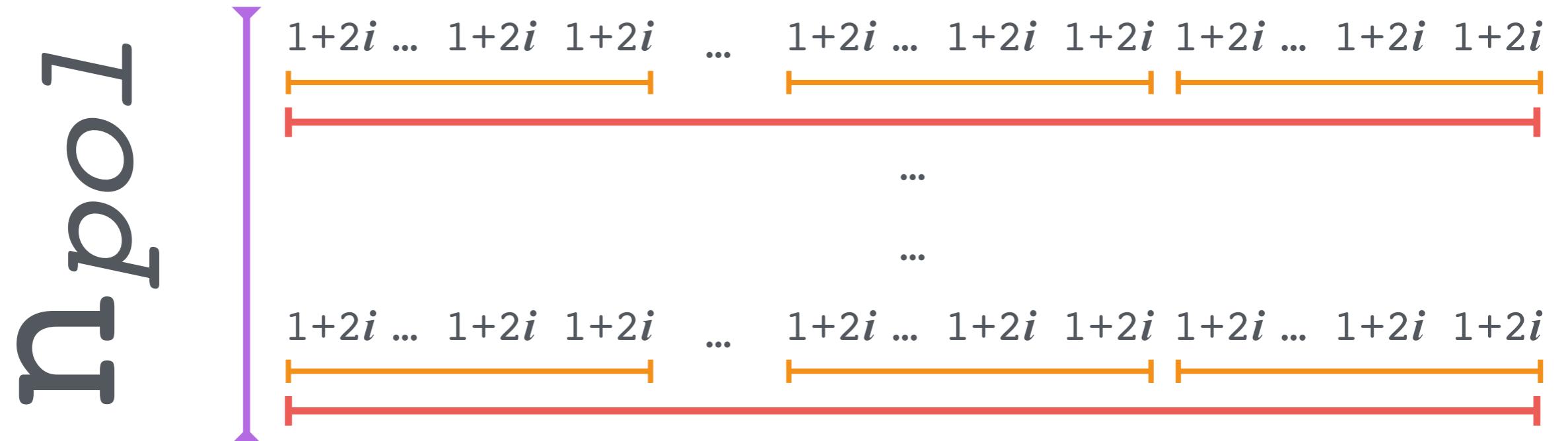
$n_{freq} \approx 16 \dots 2048$

$1+2i \dots 1+2i$   $1+2i$   $1+2i$   $1+2i \dots 1+2i$   $1+2i$   $1+2i \dots 1+2i$   $1+2i$   $1+2i$



$n_{subband}$

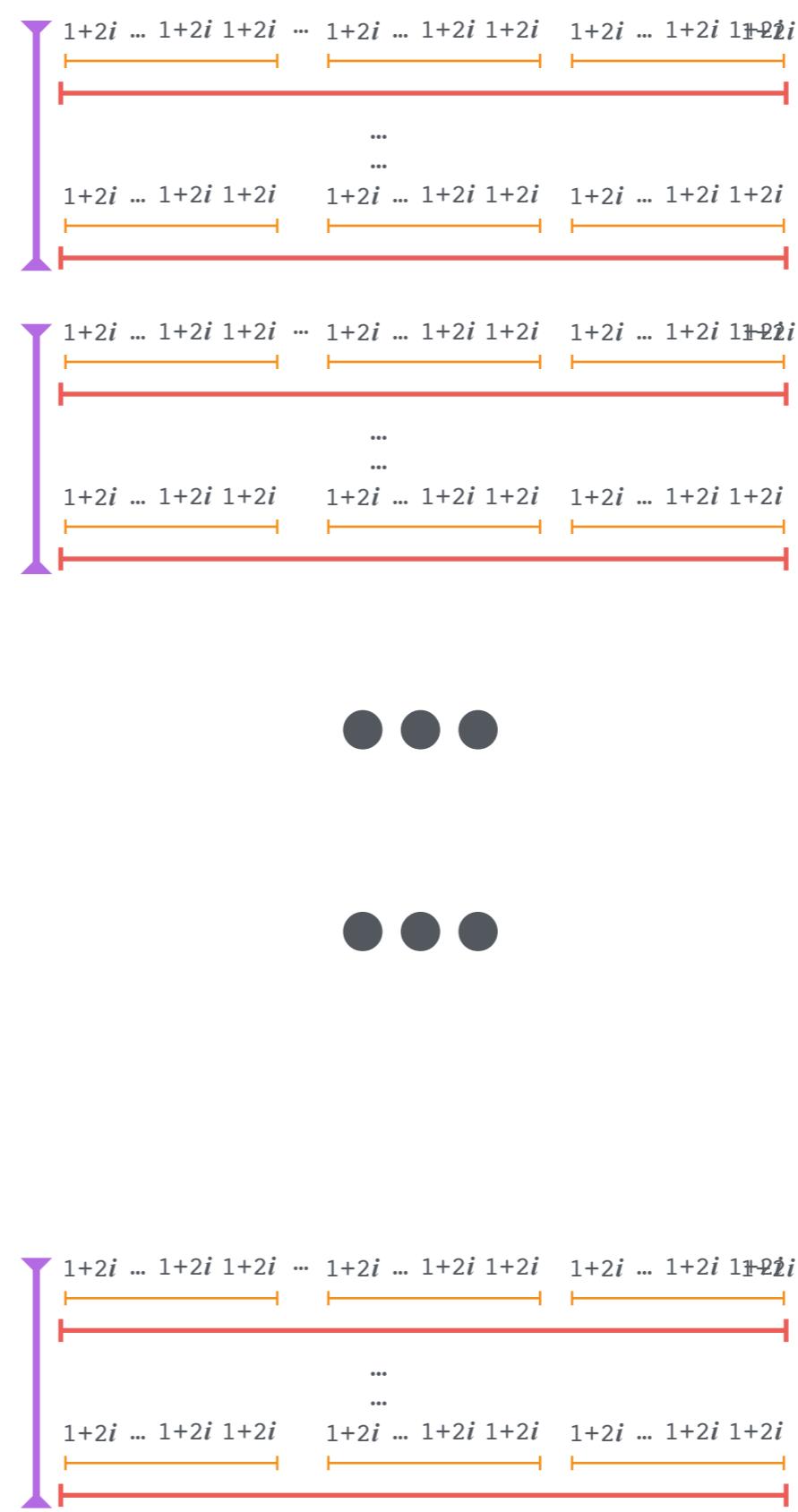
$$n_{subband} \approx 4 \dots 64$$



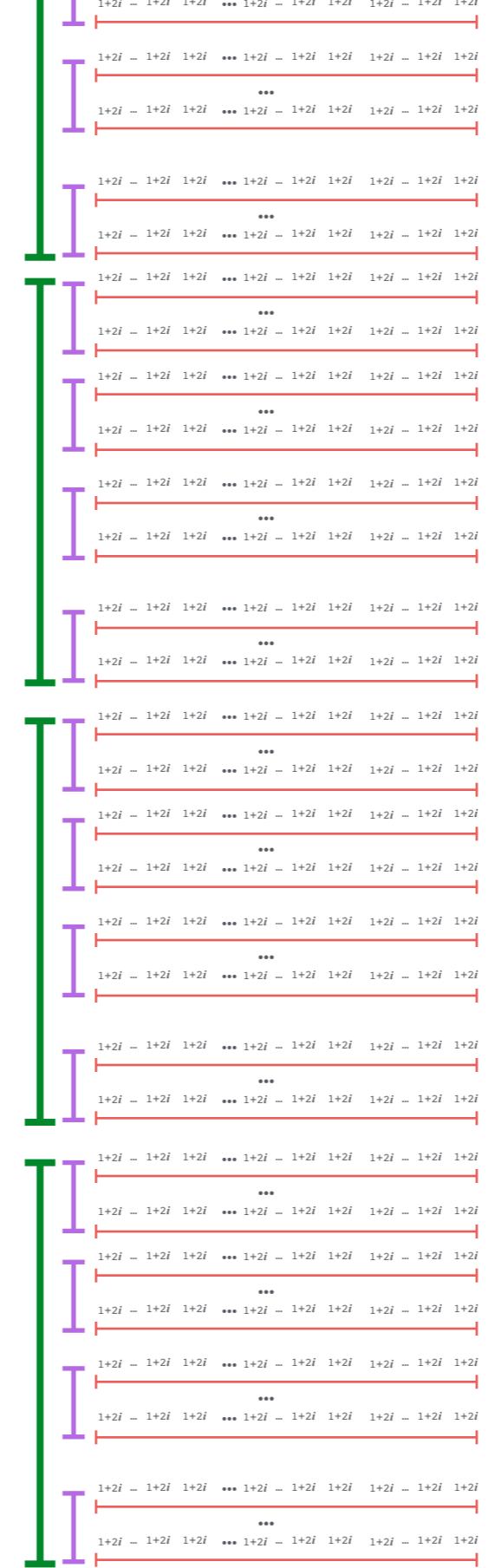
$n_{pol} = 1 \dots 4$

# $n_{\text{baseline}}$

$$n_{\text{baseline}} \propto n_{\text{antenna}}^2, \quad n_{\text{antenna}} \approx 10 \dots 30$$



**n<sub>int</sub>**



$$t_{obs} \approx \text{hours}, \quad n_{int} \approx 4 \cdot 10^3 / \text{hour}$$