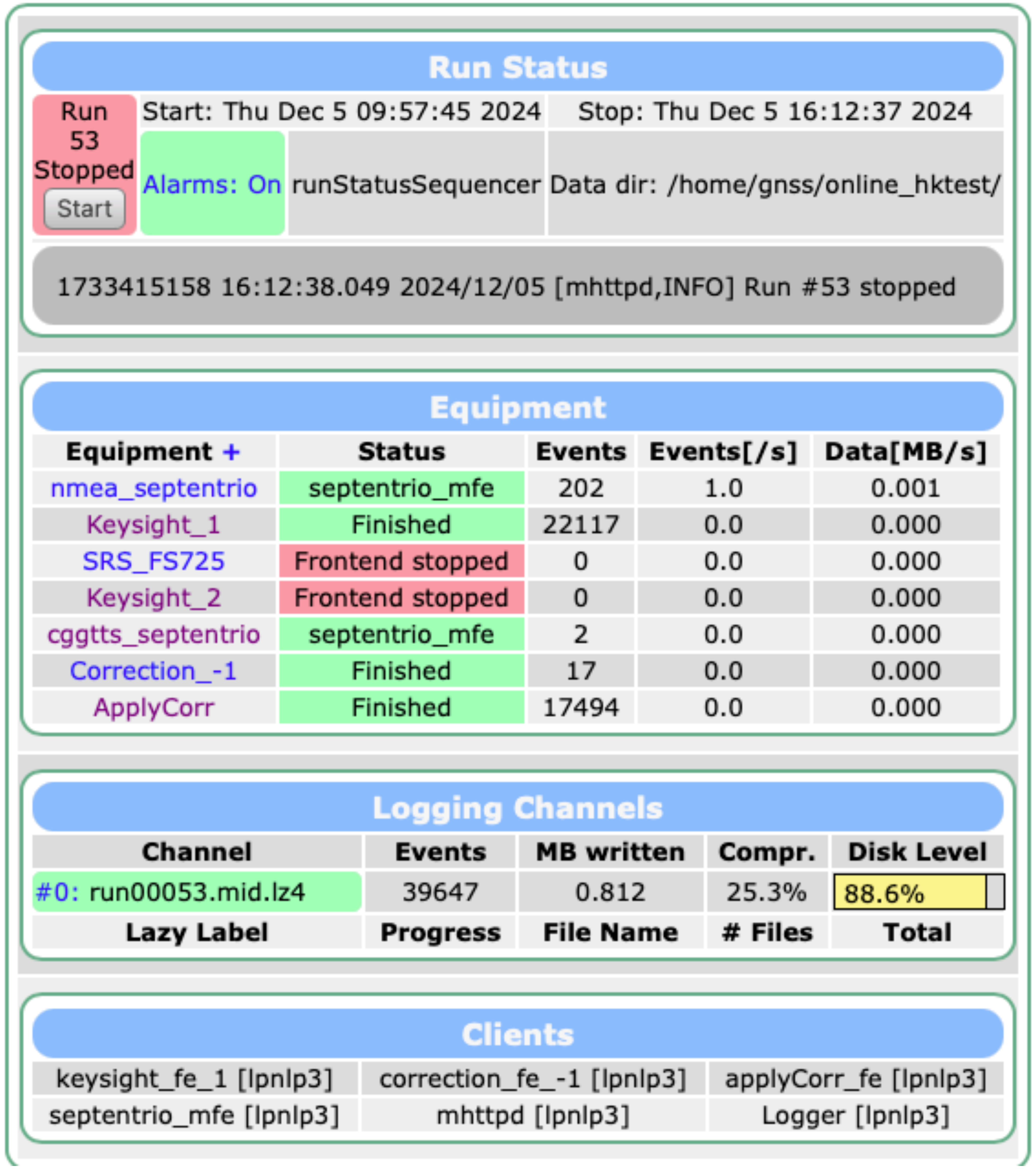
Clock drift correction system: real-time implementation

Claire Dalmazzone, 11th December 2024

Real-time correction calculations

Correction

- Done with the **Correction** Frontend, script correction_fe.py



Run Status

Run 53 Stopped Start	Start: Thu Dec 5 09:57:45 2024		Stop: Thu Dec 5 16:12:37 2024
	Alarms: On	runStatusSequencer	Data dir: /home/gnss/online_hktest/

```
1733415158 16:12:38.049 2024/12/05 [mhttpd,INFO] Run #53 stopped
```

Equipment

Equipment +	Status	Events	Events[/s]	Data[MB/s]
nmea_septentrio	septentrio_mfe	202	1.0	0.001
Keysight_1	Finished	22117	0.0	0.000
SRS_FS725	Frontend stopped	0	0.0	0.000
Keysight_2	Frontend stopped	0	0.0	0.000
cggtts_septentrio	septentrio_mfe	2	0.0	0.000
Correction_-1	Finished	17	0.0	0.000
ApplyCorr	Finished	17494	0.0	0.000

Logging Channels

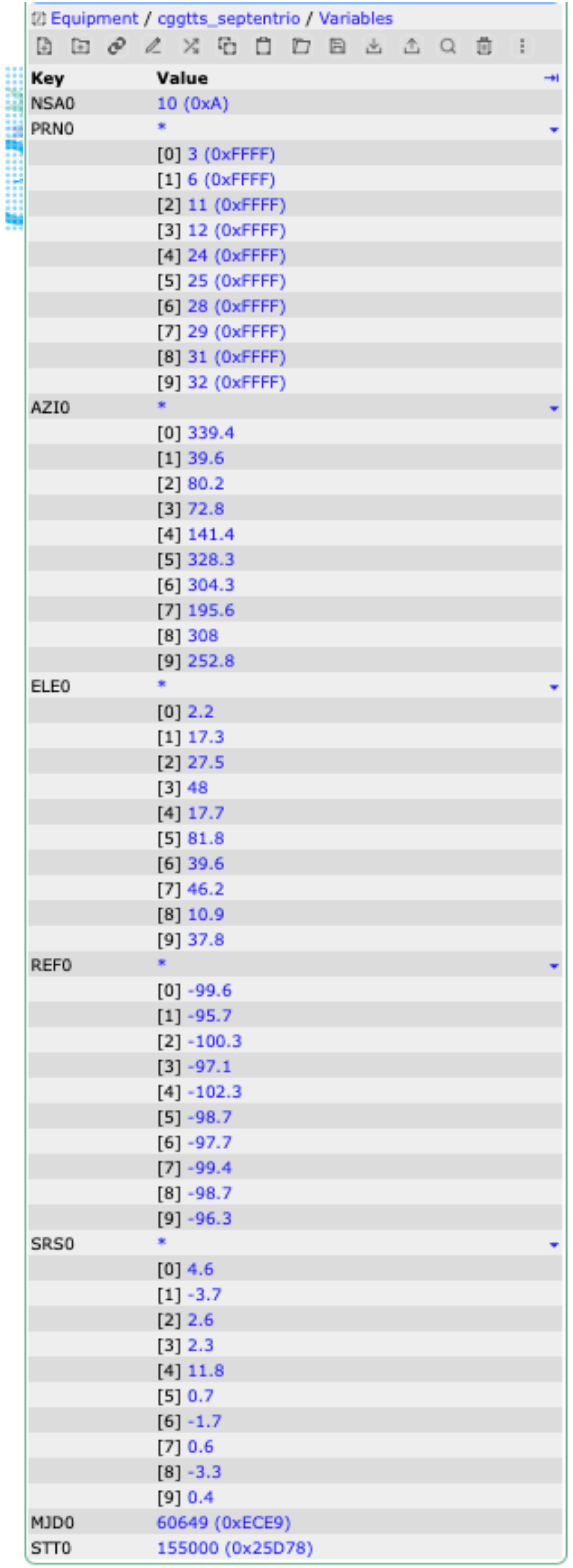
Channel	Events	MB written	Compr.	Disk Level
#0: run00053.mid.lz4	39647	0.812	25.3%	88.6%
Lazy Label	Progress	File Name	# Files	Total

Clients

keysight_fe_1 [lpnlp3]	correction_fe_-1 [lpnlp3]	applyCorr_fe [lpnlp3]
septentrio_mfe [lpnlp3]	mhttpd [lpnlp3]	Logger [lpnlp3]

Correction

- Done with the **Correction** Frontend, script correction_fe.py
- It continuously reads the ODB of the **cggtts_septentrio** frontend.
- Every time a new measurement is available, it updates the correction coefficients



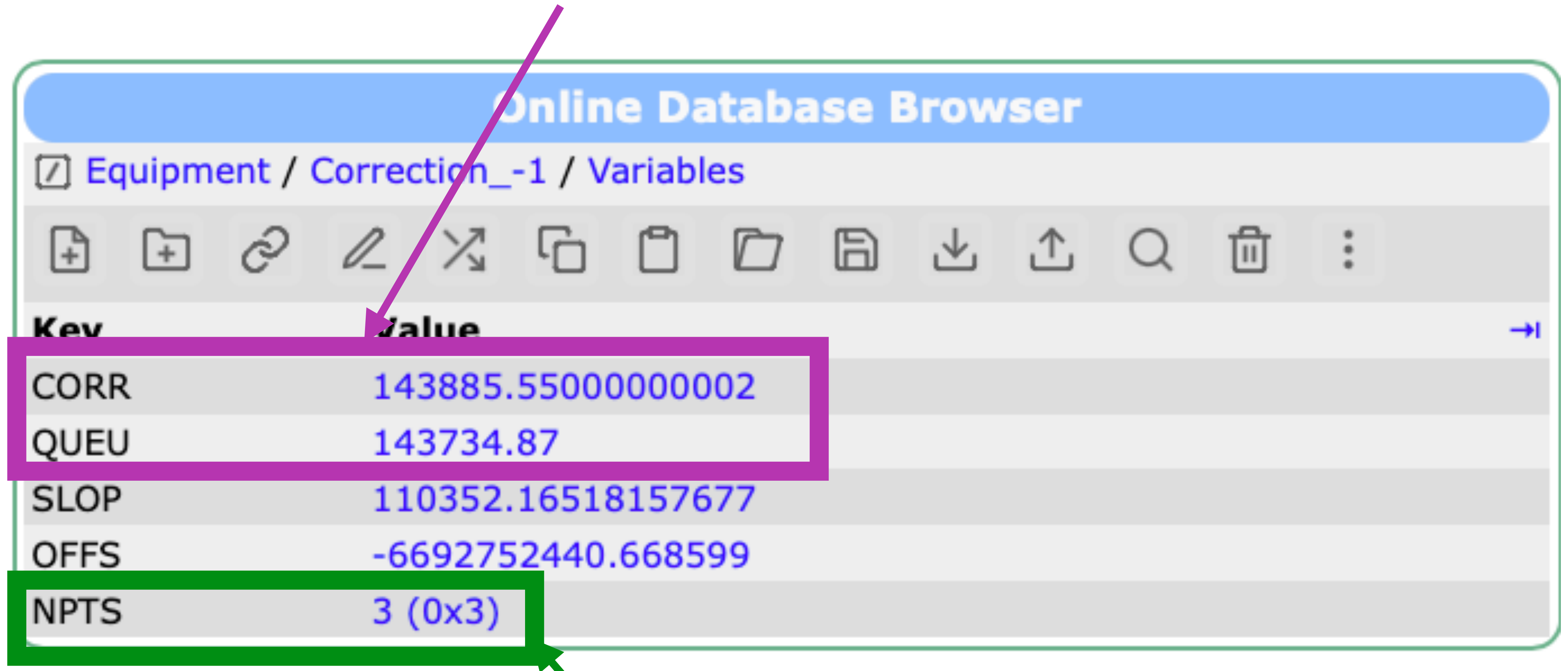
Equipment / cggtts_septentrio / Variables

```
Key     Value
NSA0    10 (0xA)
PRN0    *
        [0] 3 (0xFFFF)
        [1] 6 (0xFFFF)
        [2] 11 (0xFFFF)
        [3] 12 (0xFFFF)
        [4] 24 (0xFFFF)
        [5] 25 (0xFFFF)
        [6] 28 (0xFFFF)
        [7] 29 (0xFFFF)
        [8] 31 (0xFFFF)
        [9] 32 (0xFFFF)
AZI0    *
        [0] 339.4
        [1] 39.6
        [2] 80.2
        [3] 72.8
        [4] 141.4
        [5] 328.3
        [6] 304.3
        [7] 195.6
        [8] 308
        [9] 252.8
ELE0    *
        [0] 2.2
        [1] 17.3
        [2] 27.5
        [3] 48
        [4] 17.7
        [5] 81.8
        [6] 39.6
        [7] 46.2
        [8] 10.9
        [9] 37.8
REF0    *
        [0] -99.6
        [1] -95.7
        [2] -100.3
        [3] -97.1
        [4] -102.3
        [5] -98.7
        [6] -97.7
        [7] -99.4
        [8] -98.7
        [9] -96.3
SRS0    *
        [0] 4.6
        [1] -3.7
        [2] 2.6
        [3] 2.3
        [4] 11.8
        [5] 0.7
        [6] -1.7
        [7] 0.6
        [8] -3.3
        [9] 0.4
MJD0    60649 (0xECE9)
STT0    155000 (0x25D78)
```

Correction

- Done with the **Correction** Frontend, script correction_fe.py
- It continuously reads the ODB of the **cggtts_septentrio** frontend.
- Every time a new measurement is available, it updates the correction coefficients
- Stores the coefficients in its ODB

Should be deleted



Online Database Browser

Equipment / Correction_-1 / Variables

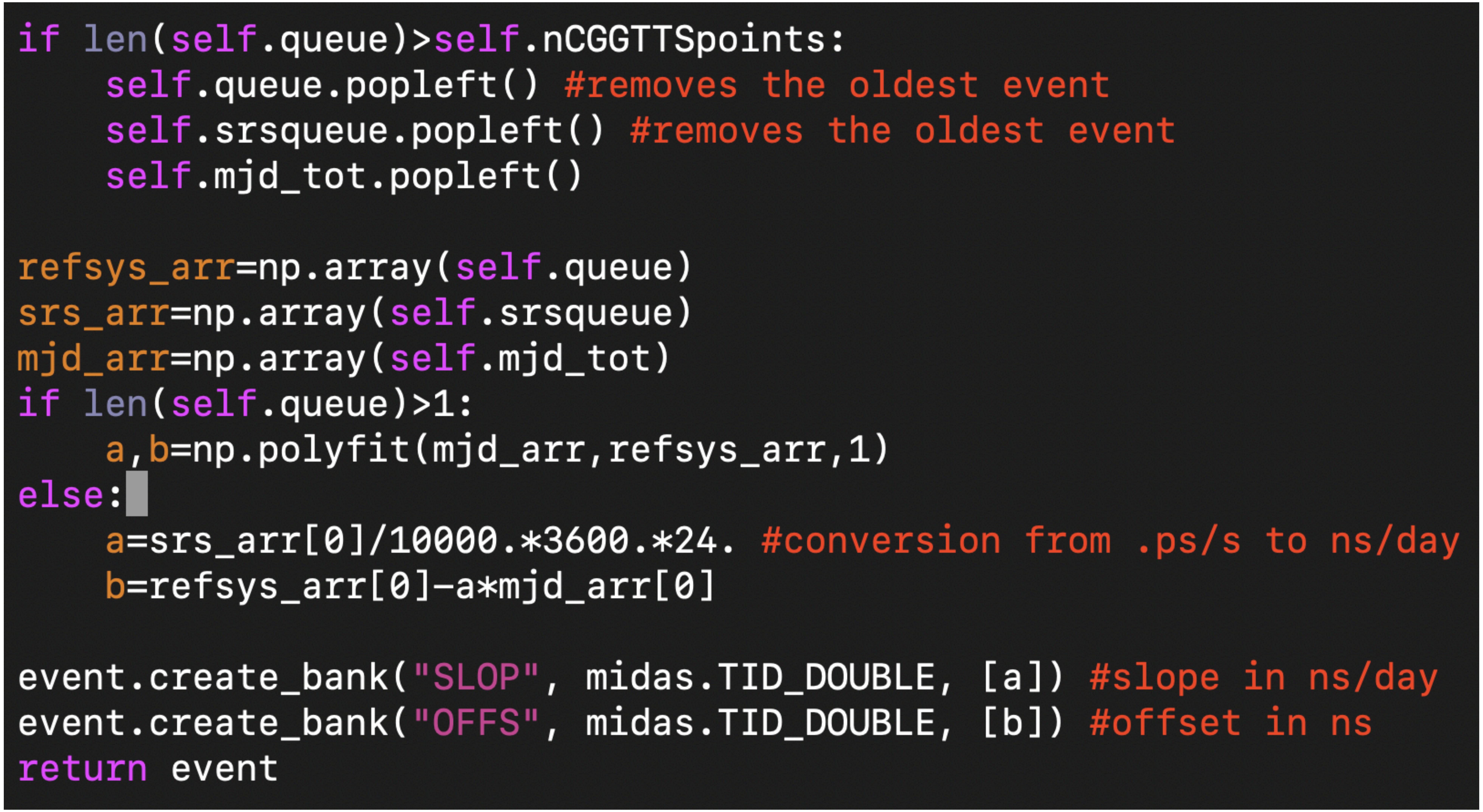
Key	Value
CORR	143885.55000000002
QUEU	143734.87
SLOP	110352.16518157677
OFFS	-6692752440.668599
NPTS	3 (0x3)

Number of Septentrio measurements to use.
Should be in Settings, not Variables...

Correction

- In permanent regime, we always use the last X (X fixed by user) Septentrio measurements to compute correction.
- The measurements are stored in a queue, the oldest element is thrown away every time a new one arrives.
- What happens at the beginning, or if the frontend crashes and is restarted?
- For now I use a similar method (linear fit using the points available): see code on next slide.
- Still need to figure out what to do when the queue is empty: for now it will take the last stored correction coefficients (probably the safest way).

Correction

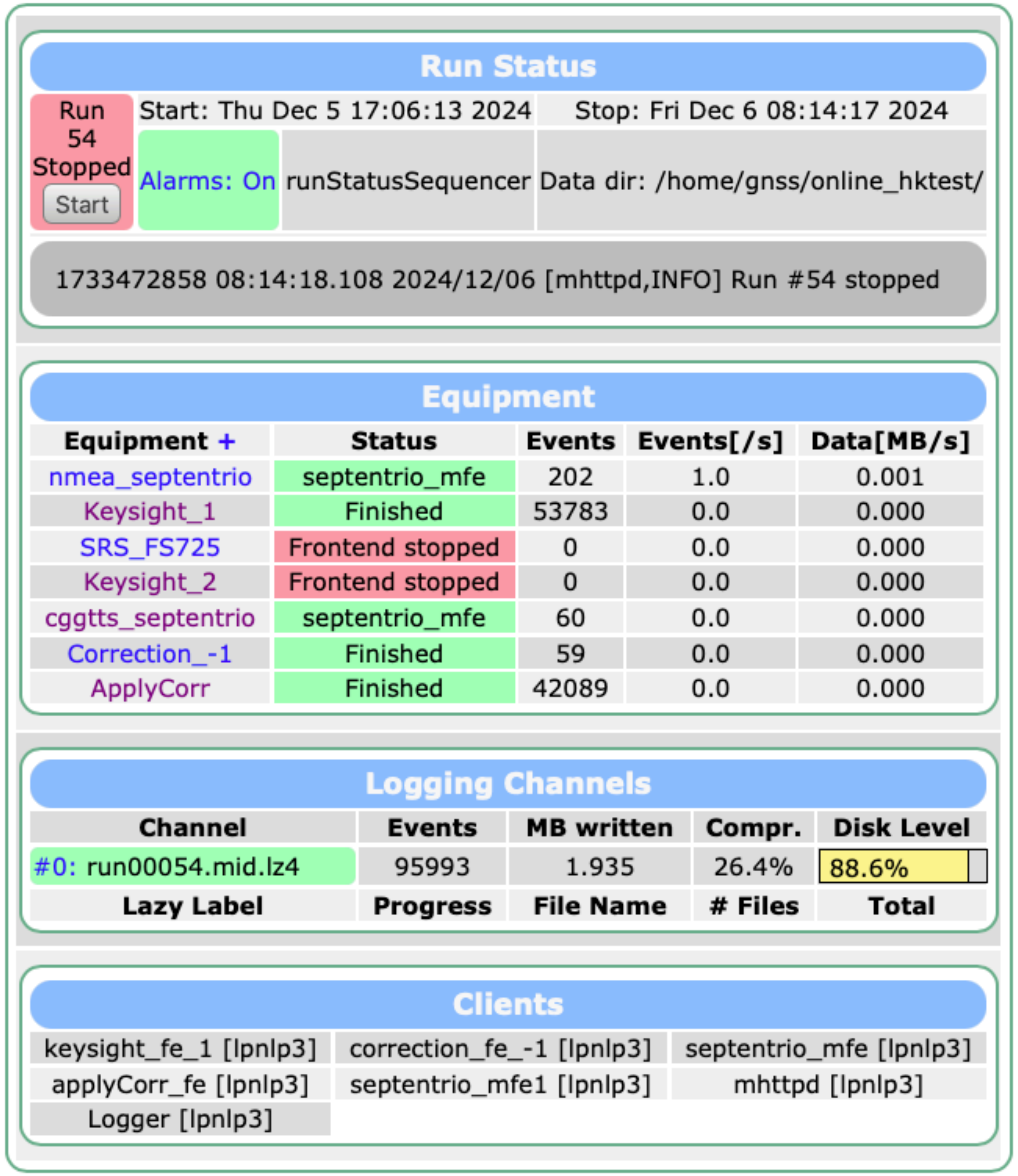


```
if len(self.queue)>self.nCGGTTSpoints:
    self.queue.popleft() #removes the oldest event
    self.srsqueue.popleft() #removes the oldest event
    self.mjd_tot.popleft()

refsys_arr=np.array(self.queue)
srs_arr=np.array(self.srsqueue)
mjd_arr=np.array(self.mjd_tot)
if len(self.queue)>1:
    a,b=np.polyfit(mjd_arr,refsys_arr,1)
else:
    a=srs_arr[0]/10000.*3600.*24. #conversion from .ps/s to ns/day
    b=refsys_arr[0]-a*mjd_arr[0]

event.create_bank("SLOP", midas.TID_DOUBLE, [a]) #slope in ns/day
event.create_bank("OFFS", midas.TID_DOUBLE, [b]) #offset in ns
return event
```

Real-time correction application

Correction test

- To test the correction, added a **ApplyCorr** frontend that applies it to a time signal measured by a **Keysight** frontend
- This frontend would not be required in the final setup, it is just for tests purposes



Run Status

Run 54 Stopped Start	Start: Thu Dec 5 17:06:13 2024	Stop: Fri Dec 6 08:14:17 2024
Alarms: On	runStatusSequencer	Data dir: /home/gnss/online_hktest/

```
1733472858 08:14:18.108 2024/12/06 [mhttpd,INFO] Run #54 stopped
```

Equipment

Equipment +	Status	Events	Events[/s]	Data[MB/s]
nmea_septentrio	septentrio_mfe	202	1.0	0.001
Keysight_1	Finished	53783	0.0	0.000
SRS_FS725	Frontend stopped	0	0.0	0.000
Keysight_2	Frontend stopped	0	0.0	0.000
cggtts_septentrio	septentrio_mfe	60	0.0	0.000
Correction_-1	Finished	59	0.0	0.000
ApplyCorr	Finished	42089	0.0	0.000

Logging Channels

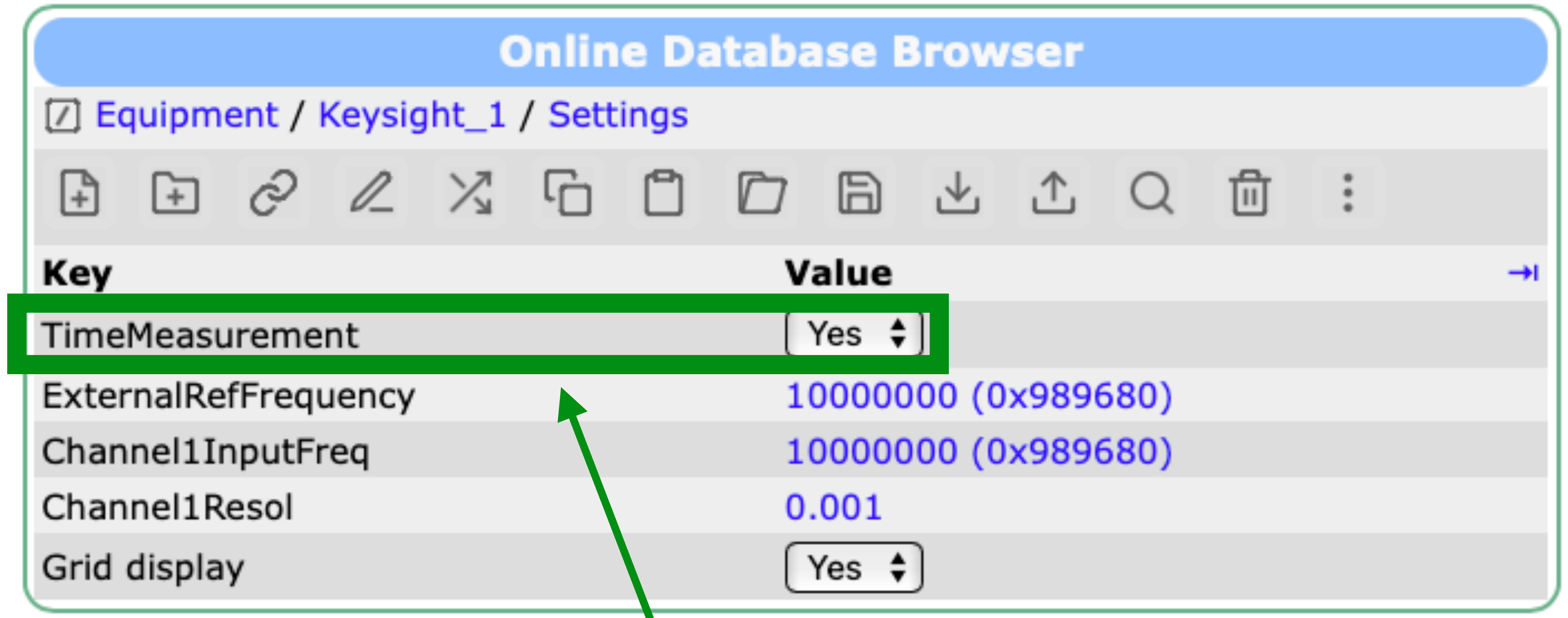
Channel	Events	MB written	Compr.	Disk Level
#0: run00054.mid.lz4	95993	1.935	26.4%	88.6%
Lazy Label	**Progress**	**File Name**	**# Files**	**Total**

Clients

keysight_fe_1 [lpnlp3]	correction_fe_-1 [lpnlp3]	septentrio_mfe [lpnlp3]
applyCorr_fe [lpnlp3]	septentrio_mfe1 [lpnlp3]	mhttpd [lpnlp3]
Logger [lpnlp3]		

Correction test

- To test the correction, added a **ApplyCorr** frontend that applies it to a time signal
- This frontend would not be required in the final setup, it is just for tests purposes
- It continuously reads the ODB of the **Keysight_1(2)** frontend.

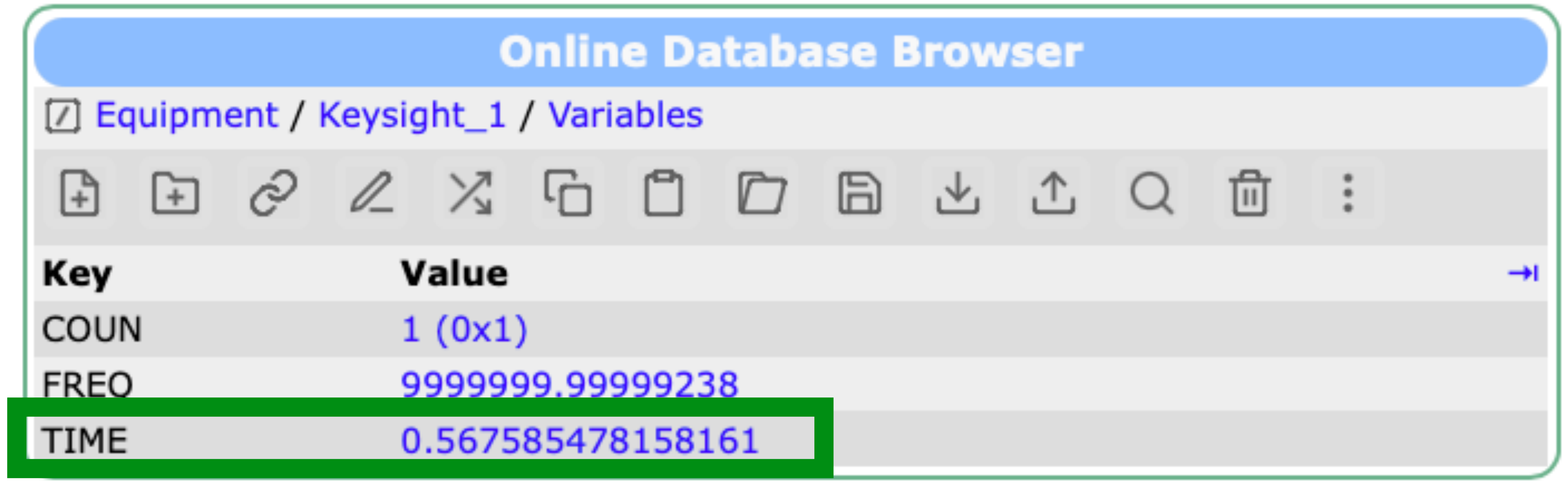


Online Database Browser

Equipment / Keysight_1 / Settings

Key	Value
TimeMeasurement	Yes
ExternalRefFrequency	10000000 (0x989680)
Channel1InputFreq	10000000 (0x989680)
Channel1Resol	0.001
Grid display	Yes

Checks if it is a Time measurement



Online Database Browser

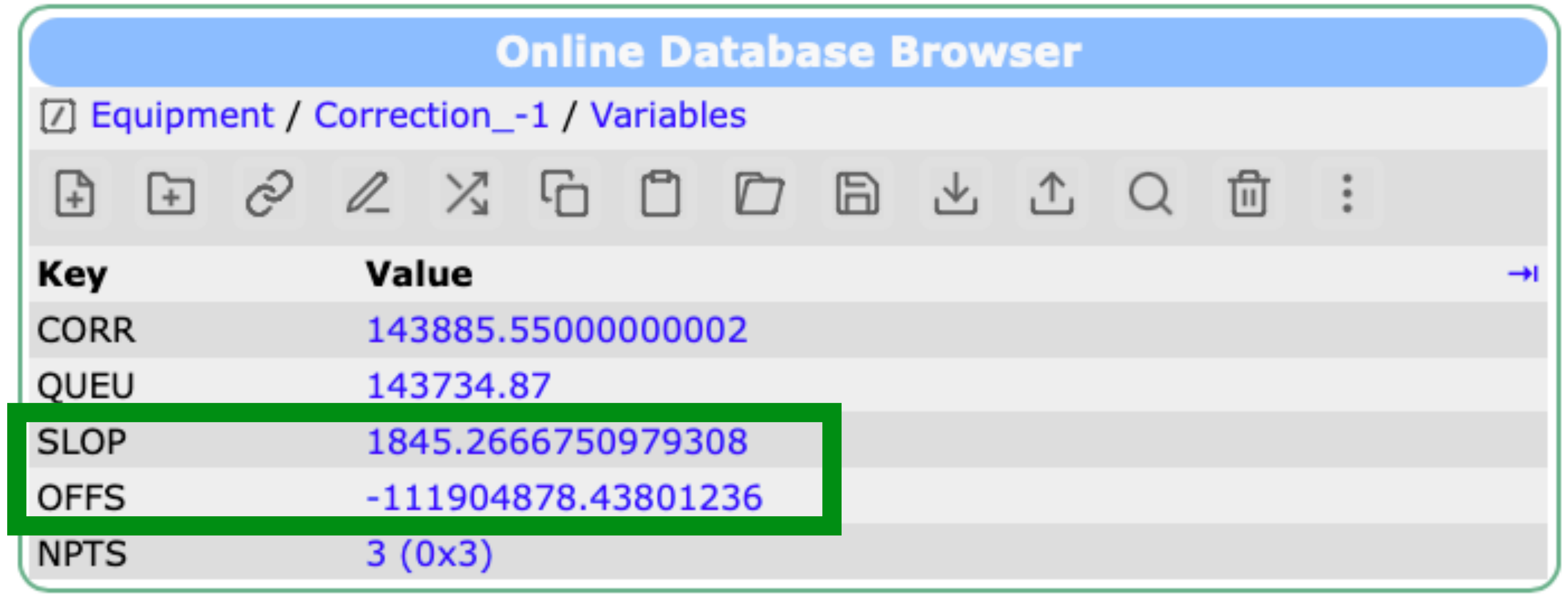
Equipment / Keysight_1 / Variables

Key	Value
COUN	1 (0x1)
FREQ	9999999.99999238
TIME	0.567585478158161

Reads the TIME value only

Correction test

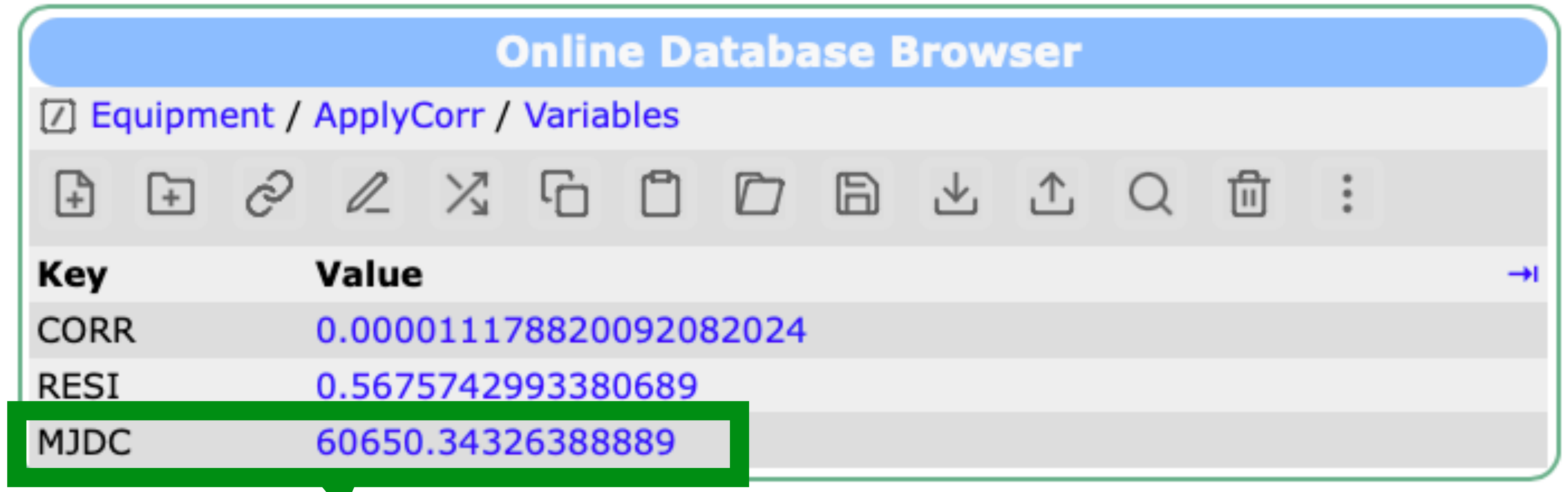
- To test the correction, added a **ApplyCorr** frontend that applies it to a time signal
- This frontend would not be required in the final setup, it is just for tests purposes
- It continuously reads the ODB of the **Keysight_1(2)** frontend.
- Every time a new **Keysight** measurement is available, it reads the current value of the correction coefficient in the **Correction** ODB and stores both the computed correction and the residual (key sight measurement - correction)



Online Database Browser

Equipment / Correction_-1 / Variables

Key	Value
CORR	143885.55000000002
QUEU	143734.87
SLOP	1845.2666750979308
OFFS	-111904878.43801236
NPTS	3 (0x3)



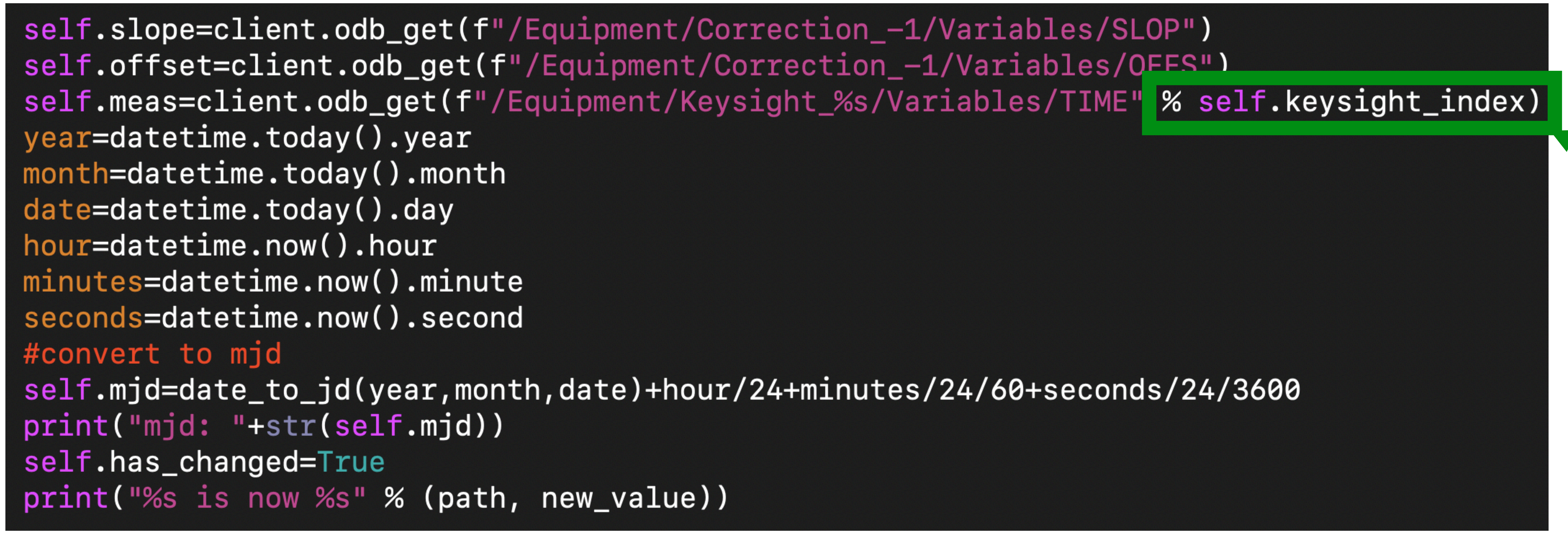
Online Database Browser

Equipment / ApplyCorr / Variables

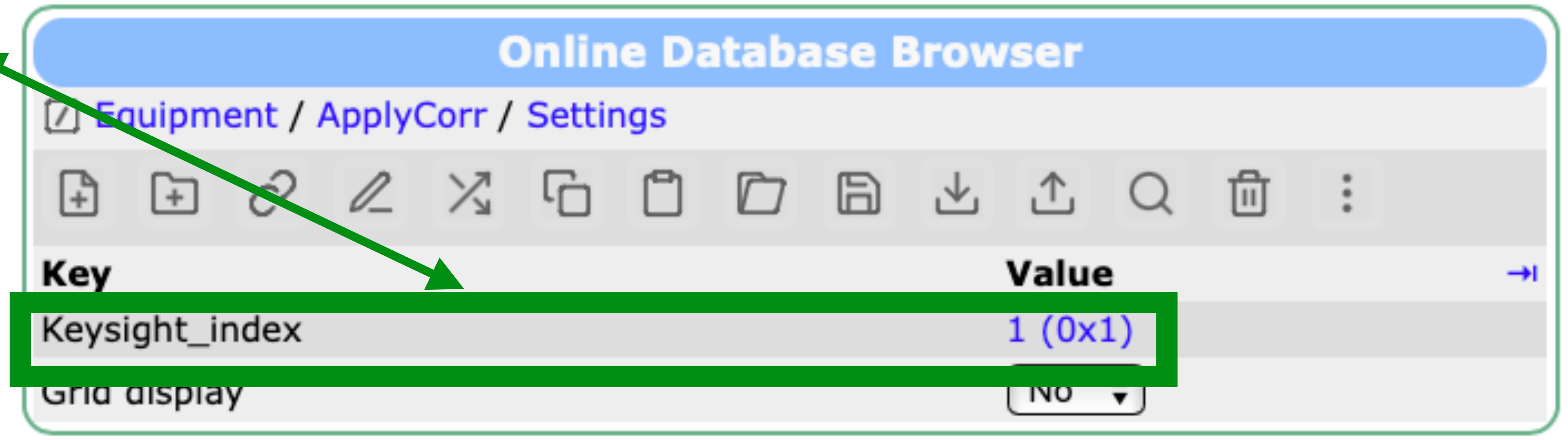
Key	Value
CORR	0.00011178820092082024
RESI	0.5675742993380689
MJDC	60650.34326388889

Date (in MJD) of the applied correction

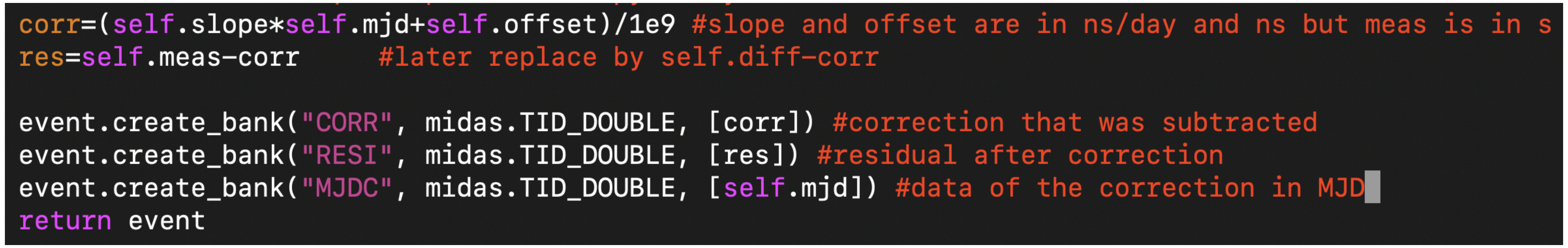
Correction test



```
self.slope=client.odb_get(f"/Equipment/Correction_-1/Variables/SLOP")
self.offset=client.odb_get(f"/Equipment/Correction_-1/Variables/OFFS")
self.meas=client.odb_get(f"/Equipment/Keysight_%s/Variables/TIME" % self.keysight_index)
year=datetime.today().year
month=datetime.today().month
date=datetime.today().day
hour=datetime.now().hour
minutes=datetime.now().minute
seconds=datetime.now().second
#convert to mjd
self.mjd=date_to_jd(year,month,date)+hour/24+minutes/24/60+seconds/24/3600
print("mjd: "+str(self.mjd))
self.has_changed=True
print("%s is now %s" % (path, new_value))
```



Key	Value
Keysight_index	1 (0x1)
Grid display	No



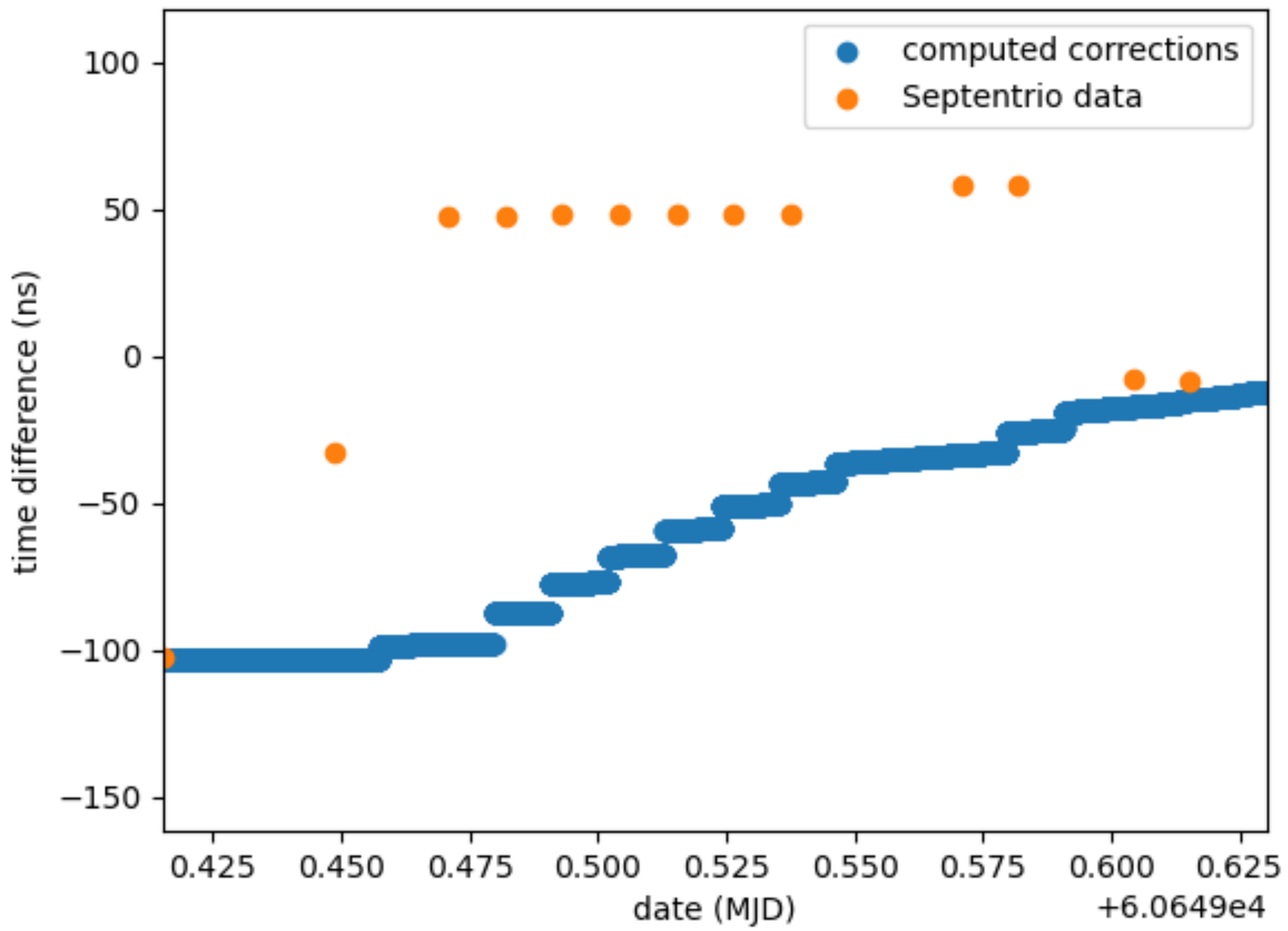
```
corr=(self.slope*self.mjd+self.offset)/1e9 #slope and offset are in ns/day and ns but meas is in s
res=self.meas-corr     #later replace by self.diff-corr

event.create_bank("CORR", midas.TID_DOUBLE, [corr]) #correction that was subtracted
event.create_bank("RESI", midas.TID_DOUBLE, [res]) #residual after correction
event.create_bank("MJDC", midas.TID_DOUBLE, [self.mjd]) #data of the correction in MJD
return event
```

First results

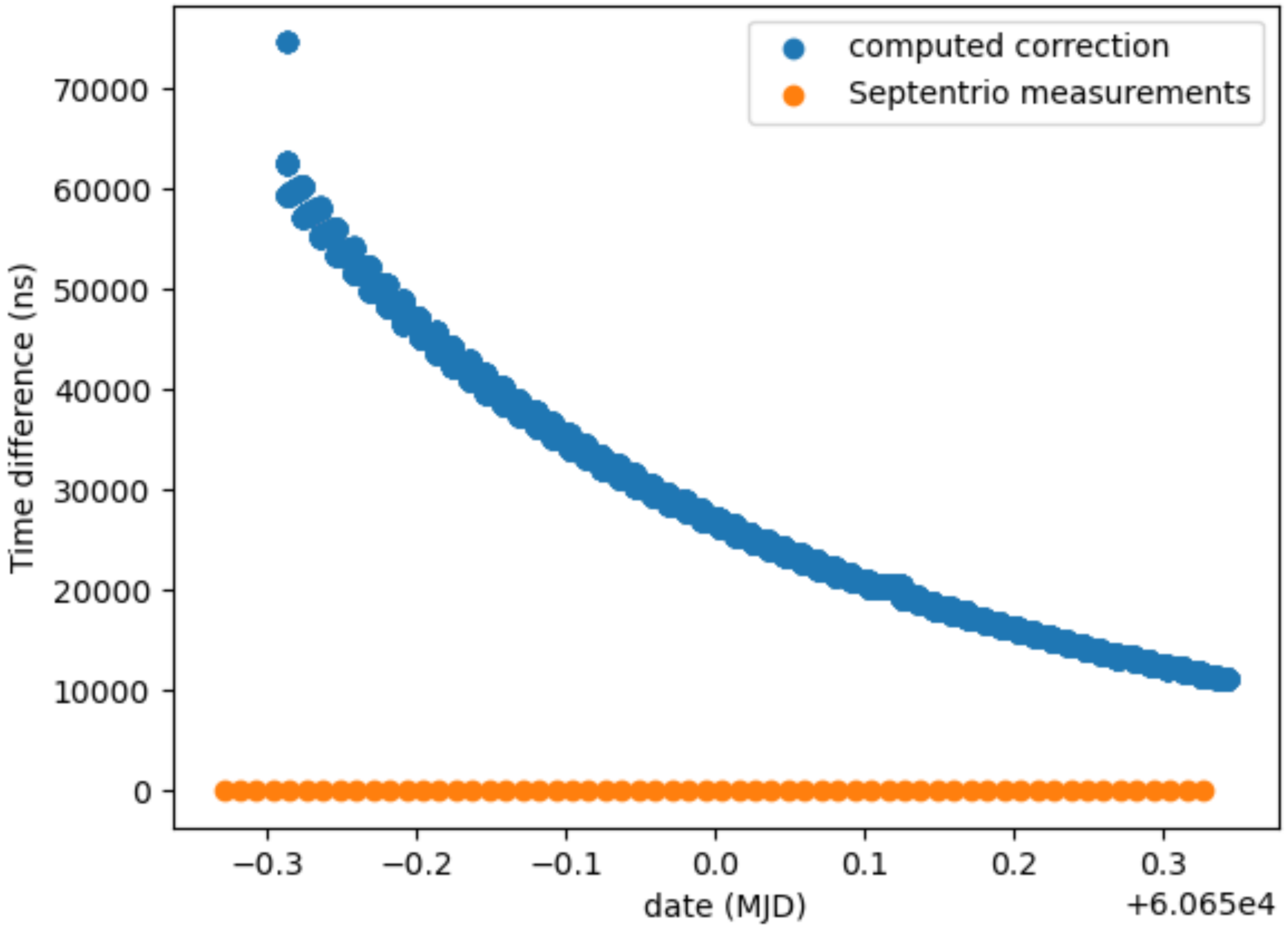
Current setup

- In parallel with Vincent’s work, started some tests
- The WR link between Rb and Septentrio is inverted so I **cannot use those results to test the quality of the correction**
- However I can **check that the computed correction corresponds to the Septenrio** measurements
- I am using a “random” time signal to apply this correction to: time difference between Refimeve and PHM

First results



First test, seems to slow to react to jump…



Second test: clearly something wrong

Was using 3 Septentrio points, clearly something is wrong. Will test again with 10 Septentrio points.