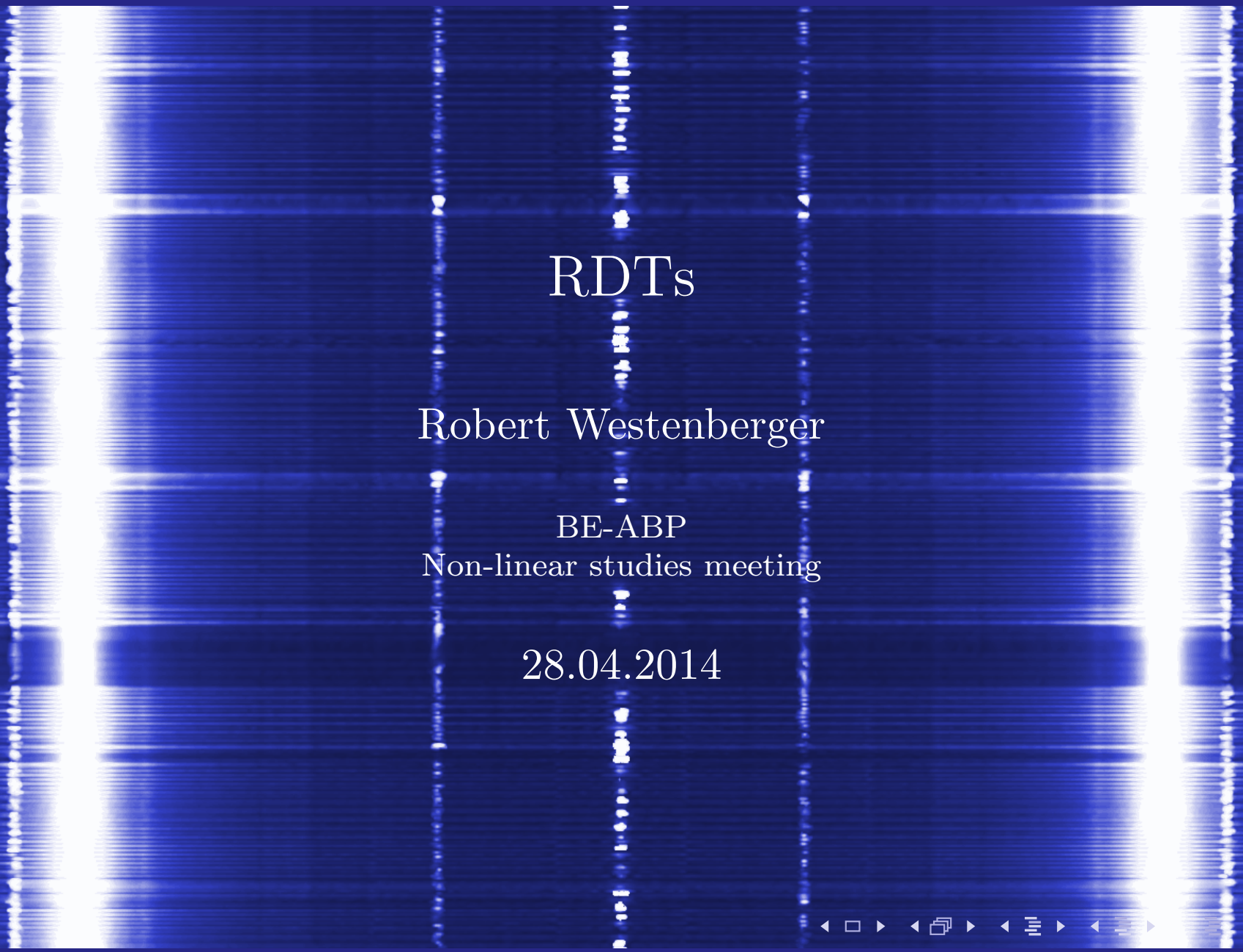


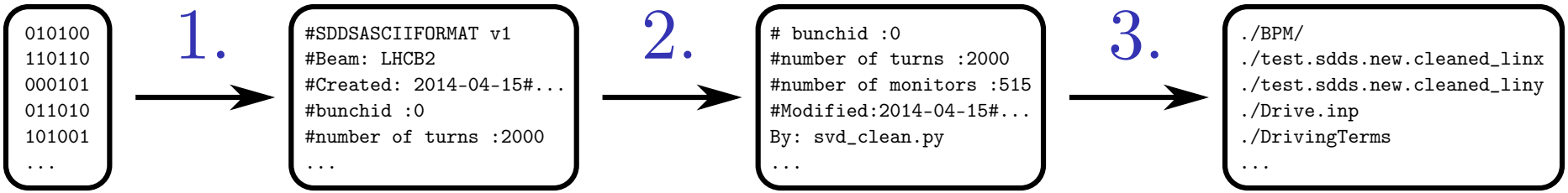
RDTs

Robert Westenberger

BE-ABP
Non-linear studies meeting

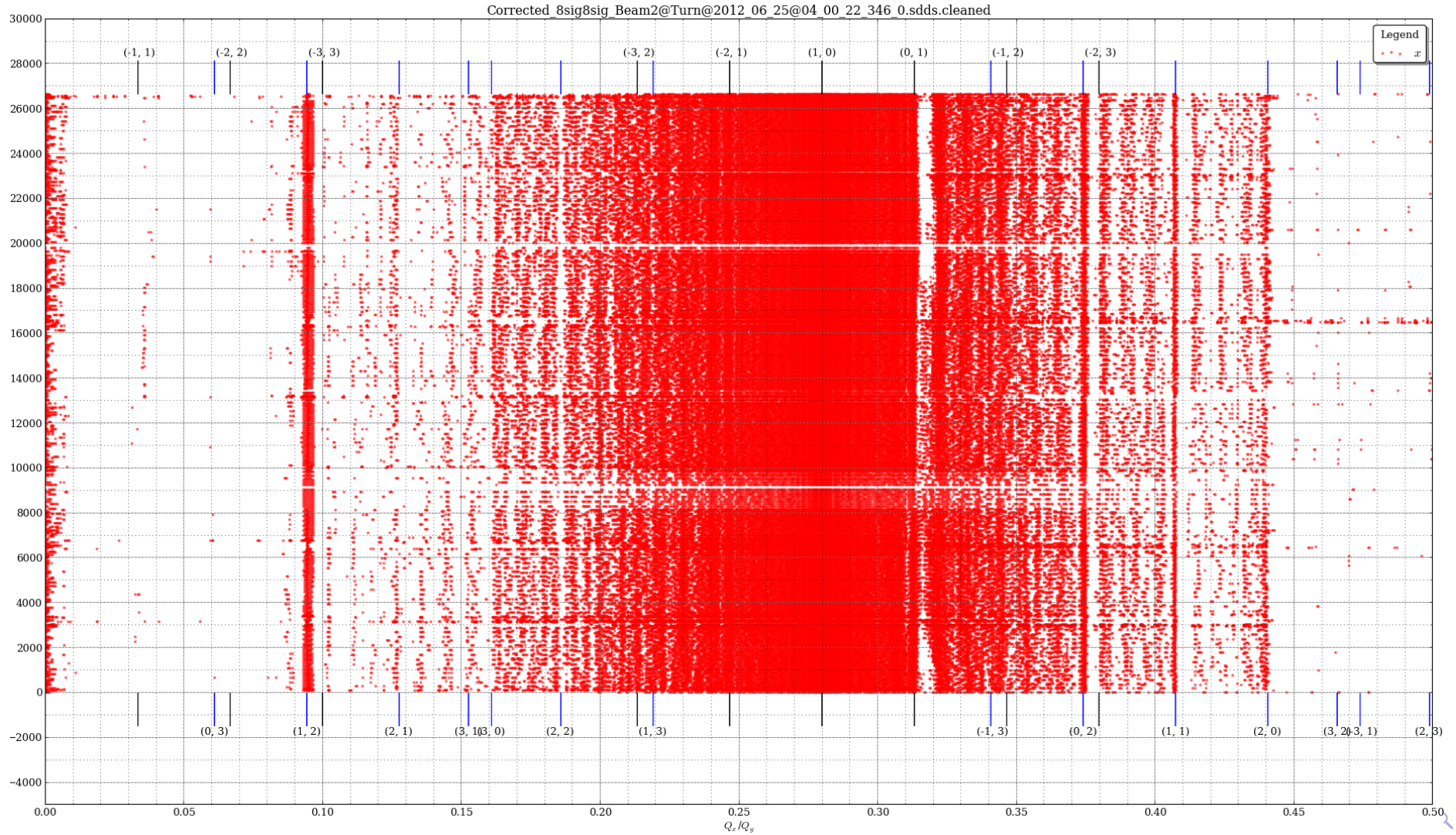
28.04.2014

Data preparation chain



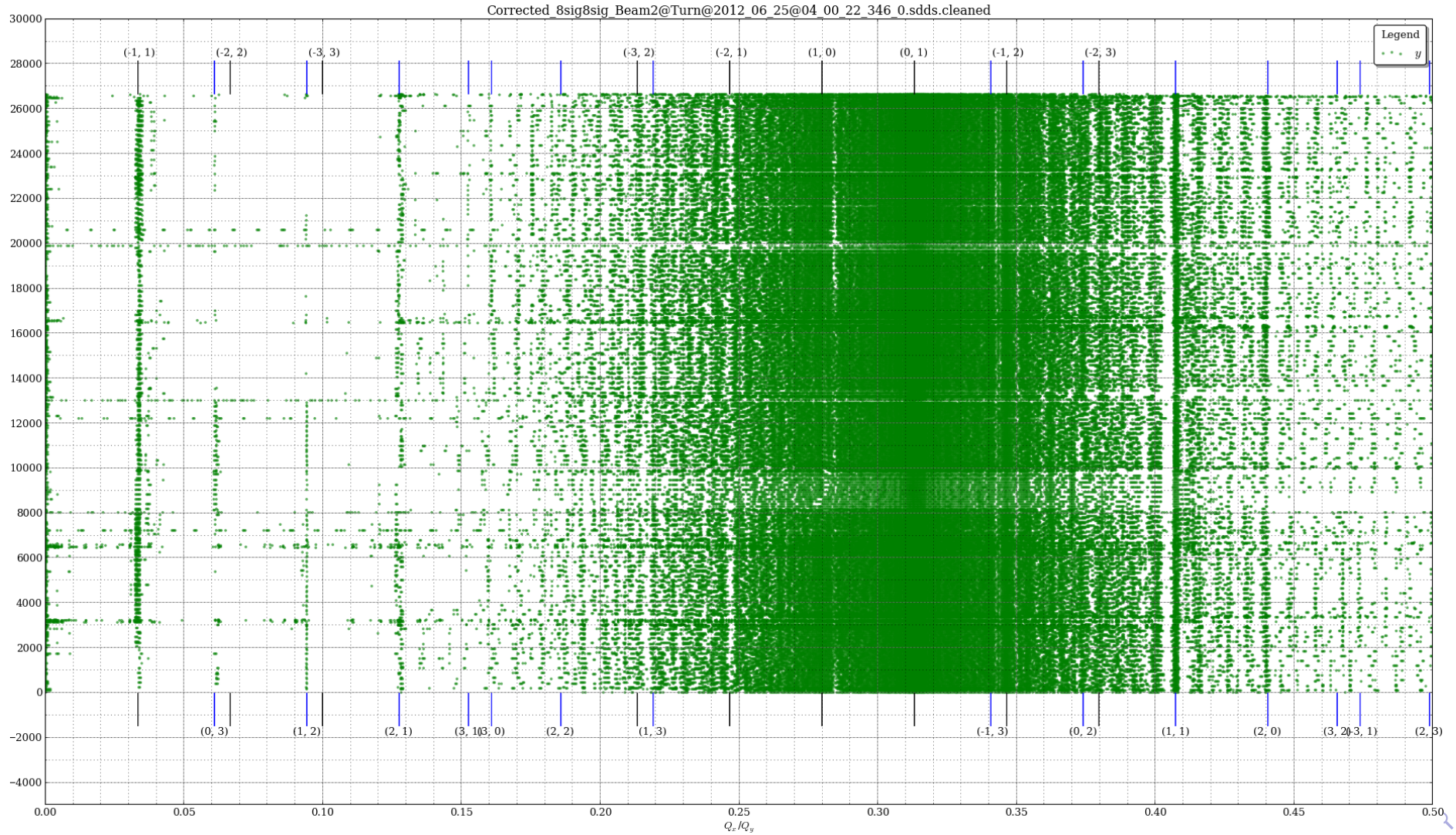
1. Binary `sdds` data to ASCII data
2. `svd_clean`
3. `Drive`

`plotSpectrum.py`:
read BPM files ($\rightarrow Amp(Q)$) and plot s over $Q_{x/y}$:



`plotSpectrum.py`:

Now most resonance lines can be identified easily:



SUSSIX

Problem: Not all seen lines can be found in the `SUSSIX` output.
Solution: Extend `SUSSIX` code to be able to output them to file.

Also:

- output lines for all BPMs (`BPM/` subdirectory)
- adjustable number of lines per BPM (`Drive_God_lin.cpp`: `FREQS_PER_BPM`)
- new `SUSSIX` test, which now also covers newly added lines automatically (amplitudes and phases)
  → generates fake signal and checks output against it

Current set of output lines

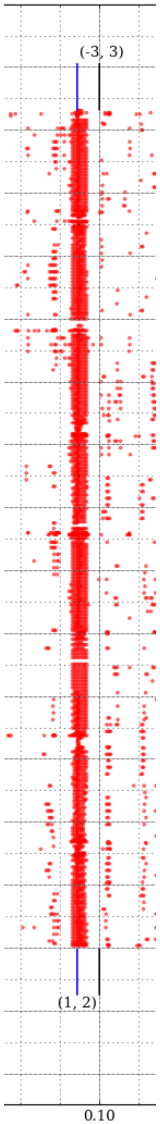
x	(1,0)	(0,1)	(1,1)	(2,0)	(0,2)	(1,2)
	(1,-2)	(-2,1)	(-3,0)	(2,-2)	(-1,3)	natQ_x
y	(0,1)	(1,0)	(1,1)	(1,-1)	(-2,0)	(0,-2)
	(2,1)	(-1,2)	(0,-3)	(-1,3)	natQ_y	

Current line set of `SUSSIX` output, not listed are $(-i, -j)$ for some lines

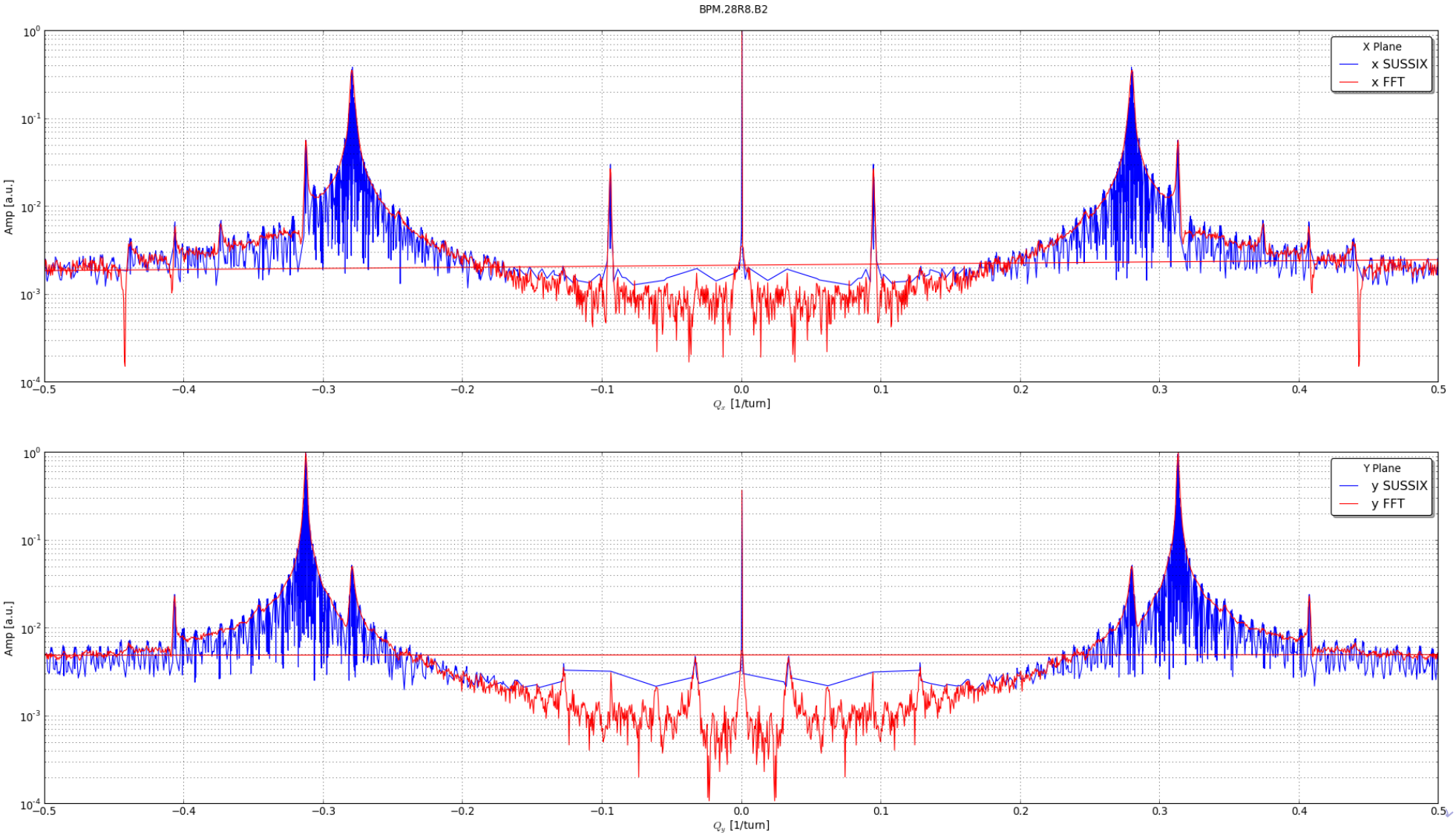
$\rightarrow$ new RDTs have to be calculated

Only 2012 injection data yet

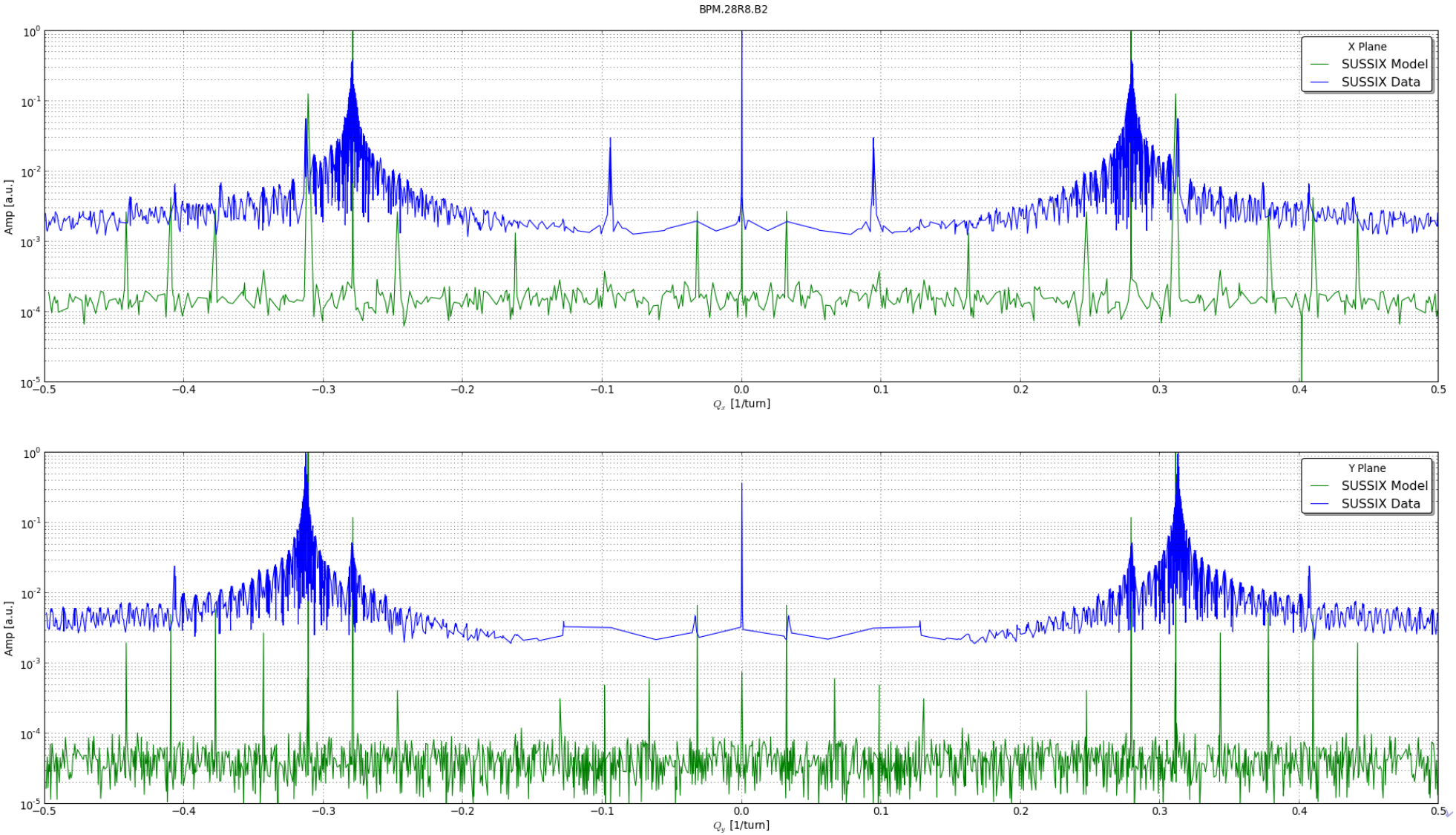
- comparison with model is already started
  → strong (1,2) line in x spectrum seems to be some kind of fragment (either `SUSSIX` or BPM problem?)
- better (and faster) way of first line identification:
  → simple FFT of data also does the job of identification? Seems so at least from first tests.



FFT vs. SUSSIX



Data vs. Model



Todo

- wider analysis of available data (script for automatic analysis is implemented already)
  - include all interesting data sets
  - comparison with model
- improve quick identification of lines to be able to select interesting data sets more efficiently
- maybe some GUI implementation?

`doSingleAnalysis.py`:
Binary data $\xrightarrow{1}$ ASCII data $\xrightarrow{2}$ `SVD_clean` data $\xrightarrow{3}$ complete `Drive` input set $\xrightarrow{4}$ linx/y + BPM files

Steps 1-4 are automatically performed by a Python script:

1. `sddsdata.py` from `Python_Classes4MAD` used to read in custom script to write out ASCII data
2. `svd_clean` called on ASCII data file with given singular values
3. `Drive.inp` and `DrivingTerms` is generated by the script and written to the ASCII data directory
4. `Drive_God_lin` is called on the generated files