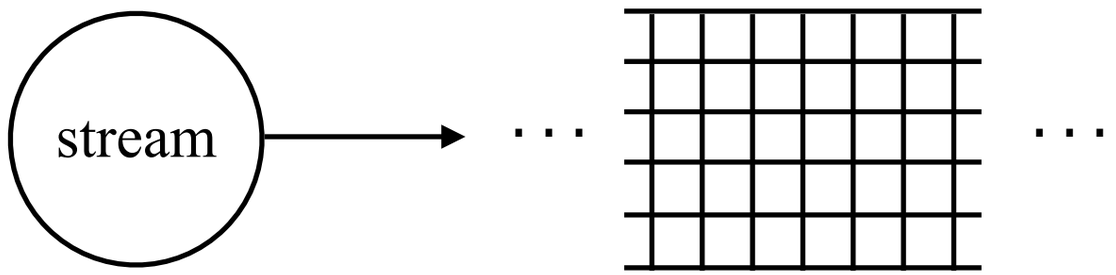
rf_pipelines update

KMS, 5 July 2016

Reminder: library with two types of objects

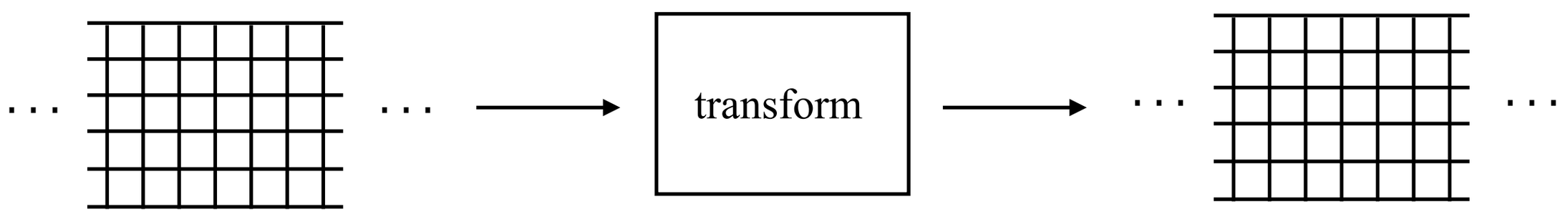
wi_stream: an object which generates weighted intensity data in incremental chunks



e.g. psrfits_stream
chime_file_stream
chime_network_stream

2d (freq,time) array
[+ a parallel weight array]

wi_transform: an object which operates on and modifies weighted intensity data in incremental chunks



Current status: python API is fairly complete and documented.
Under-the-hood C++ API still needs some optimization and cleanup.
Example python script to analyze CHIME data:

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

import rf_pipelines

# Stream object containing four arbitrarily chosen CHIME pathfinder files.
filename_list = [ '00003014.h5', '00003035.h5', '00003056.h5', '00003078.h5' ]
filename_list = [ os.path.join('/data/pathfinder/16-06-21',f) for f in filename_list ]
s = rf_pipelines.chime_stream_from_filename_list(filename_list)

# First, a "plotter transform" to make waterfall plots of the raw (pre-detrending) data
t1 = rf_pipelines.plotter_transform('raw_chime', img_nfreq=512, img_nt=1200, downsample_nt=16)

# Currently 'rf_pipelines' contains a very simple detrending transform
t2 = rf_pipelines.simple_detrender(1024)

# Another plotter after the detrender, to make detrended waterfall plots
t3 = rf_pipelines.plotter_transform('detrended_chime', img_nfreq=512, img_nt=1200, downsample_nt=

# The last "transform" doesn't actually modify its input buffers, it just runs dedispsersion.
# The output is a file 'triggers.hdf5' containing coarse-grained triggers.
# This file can be postprocessed with 'bonsai-plot-triggers.py' to make plots.
t4 = rf_pipelines.bonsai_dedisperser('bonsai_config.hdf5', 'triggers.hdf5')

# Runs the stream through a sequence of transforms
s.run([t1,t2,t3,t4])
```

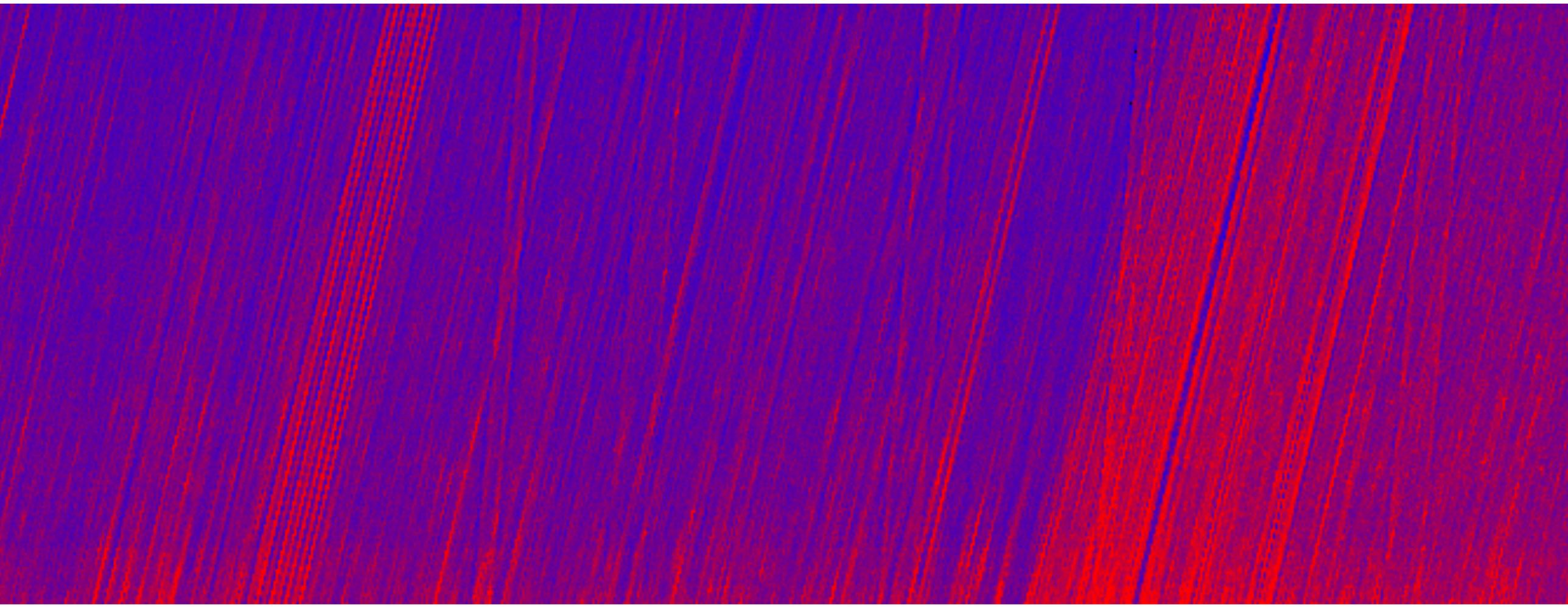
Output plot #1: waterfall (post detrending)

What's happening here is that the bright RFI near the top of the band is saturating the plot scale, and not much else is visible.

Needs RFI masking transforms!

Output plot #2: dedispersion output.

Also RFI-dominated.



Library currently includes the following builtins:

Streams:

- CHIME streams (from hdf5 capture files)
- PSRFITS stream (e.g. gbncc)
- Gaussian noise stream (for simulation)

Transforms:

- Simple detrender
- FRB injector (for simulation)
- Plotter (primitive, needs work)
- Dedisperser

More transforms needed!

New transforms can be written in python by subclassing a base class with a few methods. (See source file or doctoring for API documentation).

“Plugin” architecture intended to help collaboration.

```
class py_wi_transform(wi_transform):
    def set_stream(self, stream):
        pass

    def start_substream(self, isubstream, t0):
        pass

    def process_chunk(self, t0, t1, intensity, weights, pp_intensity, pp_weights):
        pass

    def end_substream(self):
        pass
```

My pet todo list of useful transforms (in rf_pipelines README):

- More detrenders and RFI-removing filters is a huge priority!
- In particular a detrender which correctly handles the CHIME switched noise source.
- Another concrete example: bad channel mask (should initialize from text file so it can be used for either CHIME or GBNCC)
- The plotting_transform is currently very primitive and could use improvement (e.g. more sensible choice of plot scale)
- Same goes for the 'bonsai-plot-triggers' script.
- It would be great to have more postprocessing scripts for the bonsai triggers.hdf5 output file (e.g. peak-finders)
- A transform which allows the bonsai code to be run through its new python interface (in case you want to write a trigger postprocessing callback in python)