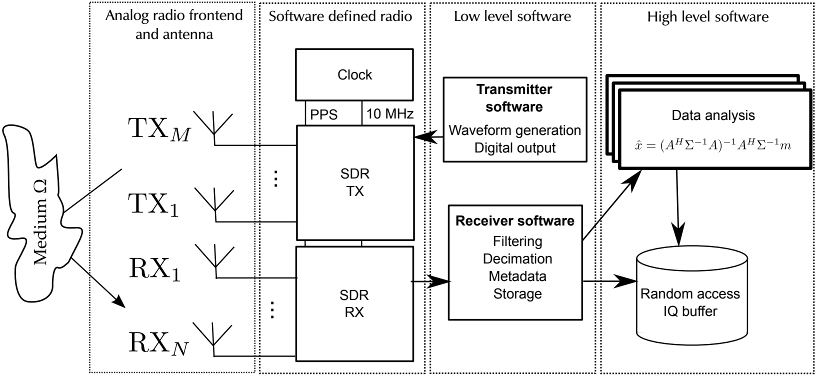
Tutorial on Software Defined Radio and Geophysical Remote Sensing

Juha Vierinen

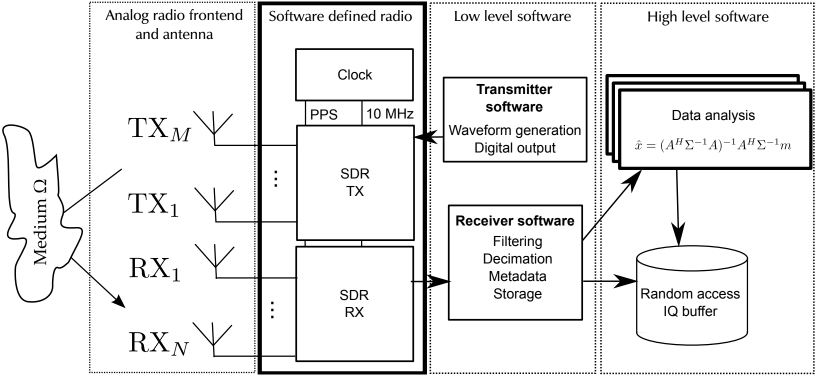
MIT Haystack Observatory, Westford MA
x@mit.edu

May 30, 2014

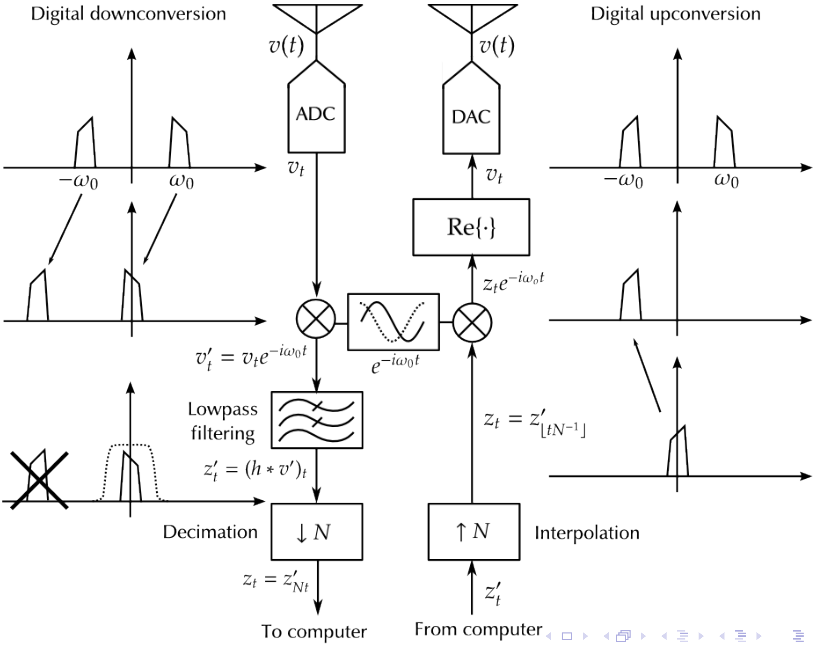
Software defined radio



Modern software defined radio remote sensing architecture



Software defined radio



Undersampling, Nyquist zones

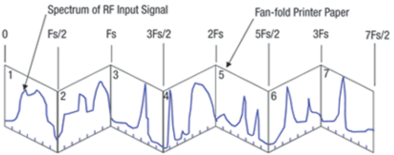


Figure 1 Fan-fold paper showing a spectrum of baseband signal.

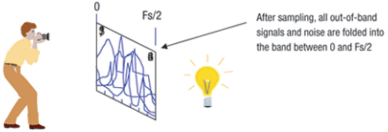


Figure 2 Looking through the collapsed stack reveals the resulting spectrum.

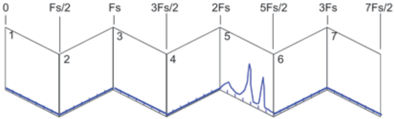


Figure 3 Fan-fold paper showing a spectrum of bandpass IF signal.

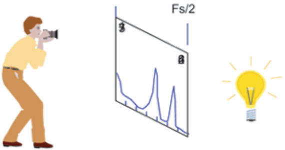
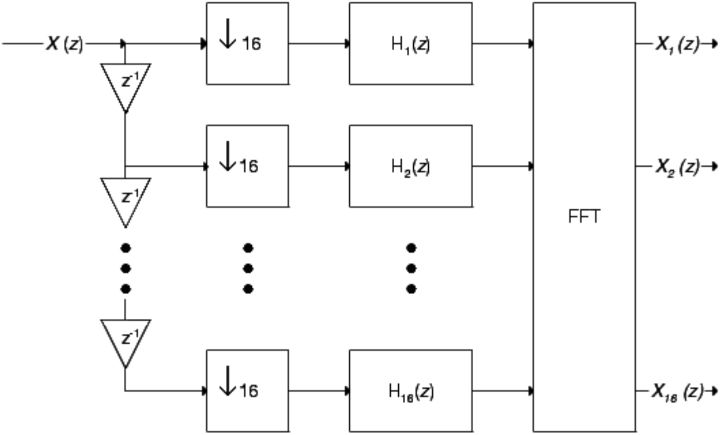


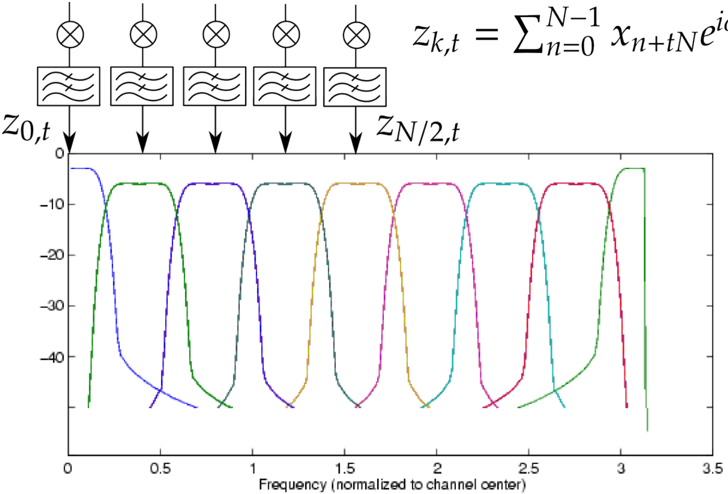
Figure 4 With a bandpass IF signal, undersampling preserves the signal.

Polyphase filterbank



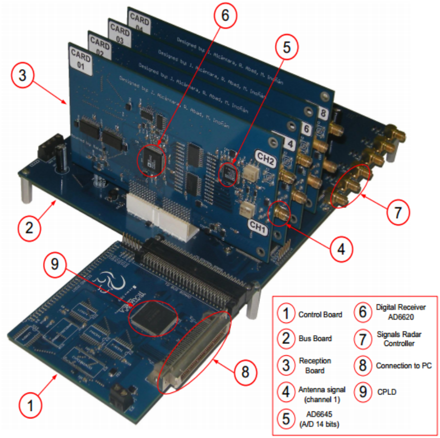
Polyphase filterbank

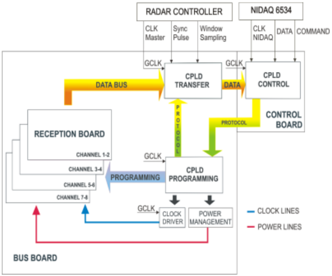
$$z_{k,t} = \sum_{n=0}^{N-1} x_{n+tN} e^{i\omega_k n}$$



Examples of systems: JARS

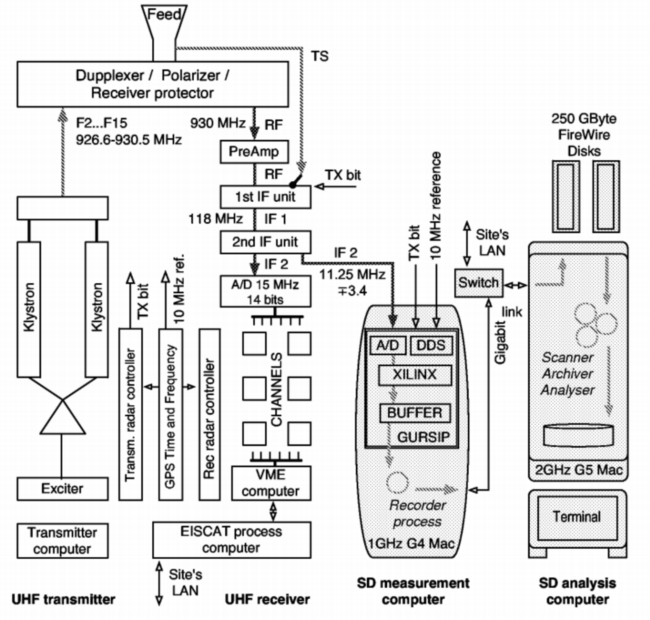
JARS - Jicamarca Acquisition Radar System



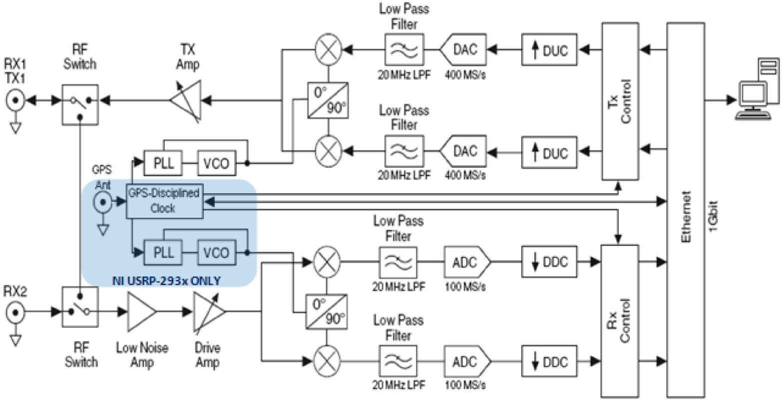


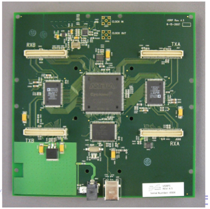
8 digital receivers @ 1MHz
ADC 14 bits (84 dB)

Examples of systems: GURSIP ($\approx$ 1995)



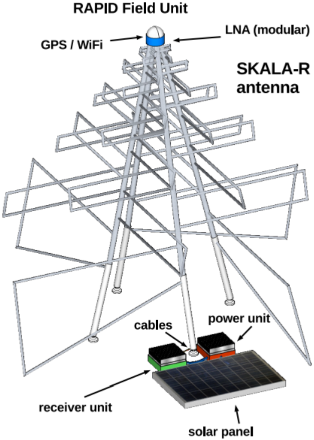
Examples of systems: USRP

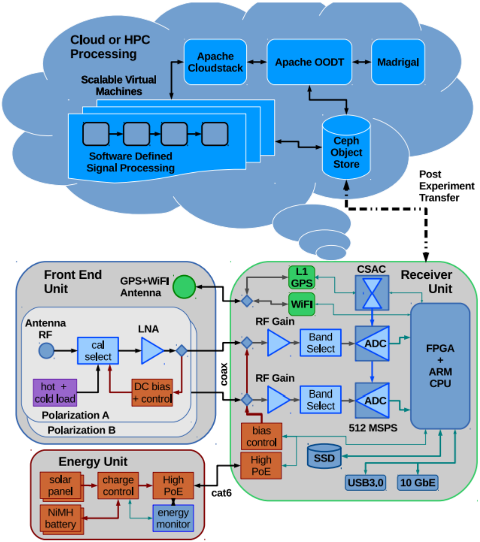




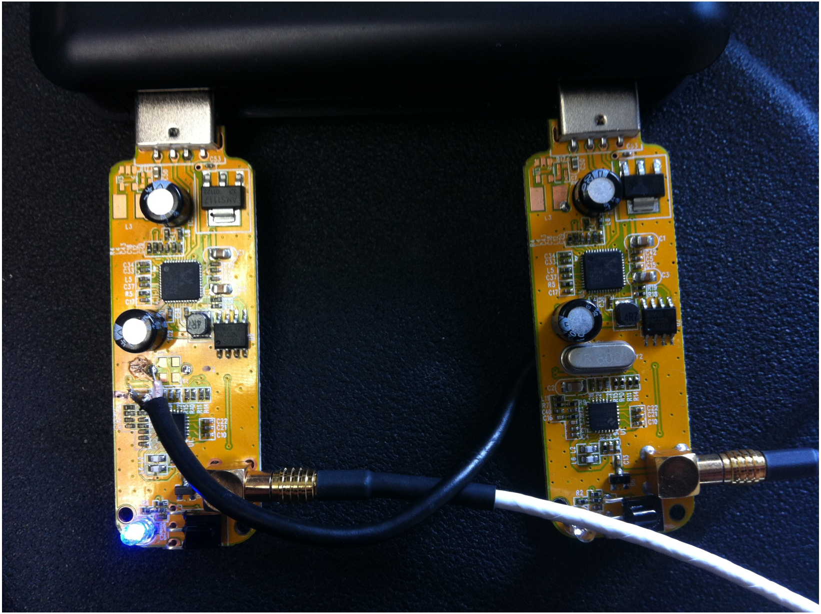
Examples of systems: LOFAR polyphase filterbank chained beamformer

Examples of systems: RAPID (Lind et.al., 2014)

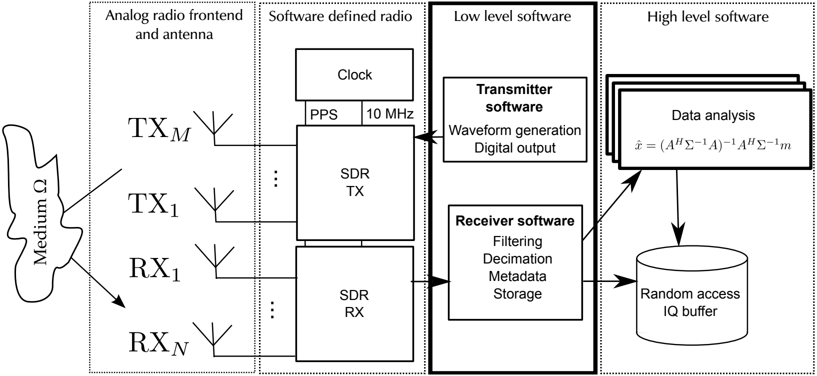




Examples of systems: RTLSDR ($8 digital receiver)



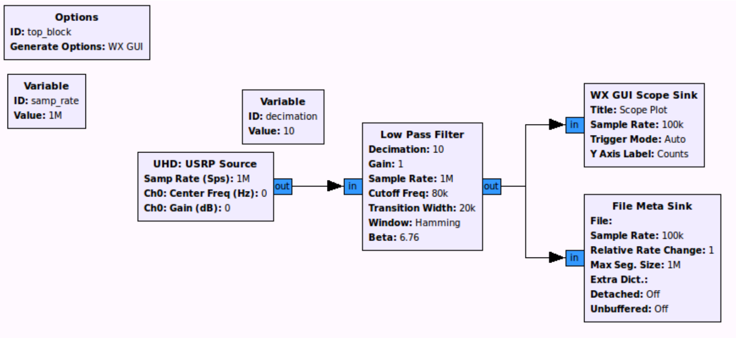
Low level software



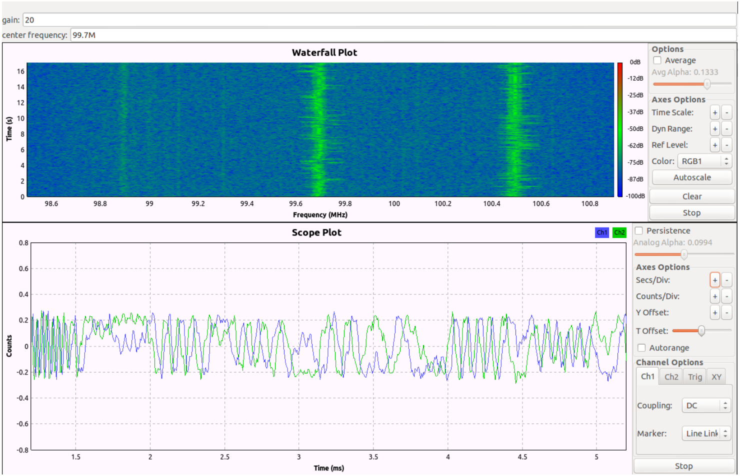
Gnuradio

- Flowgraph based framework for signal processing (like Labview from National Instruments)
- Implemented in C++ and Python
- Stable and extensible
- Supports different hardware

Gnuradio Flowgraph Example



Gnuradio Flowgraph Screenshot



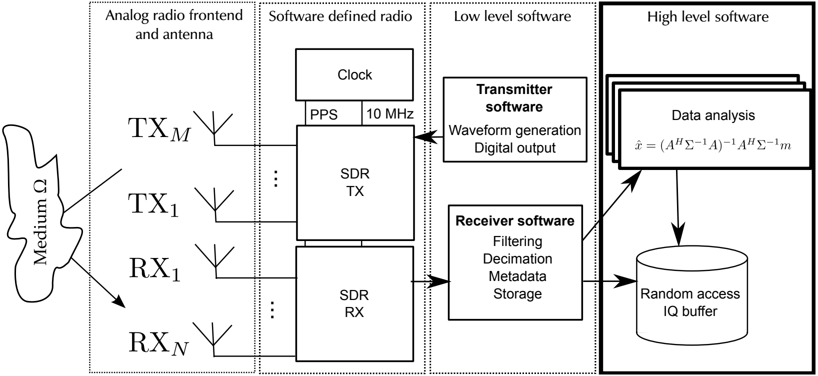
Gnuradio example: transmit waveform

```
# gnuradio flowgraph
tb = gr.top_block()
# usrp sink (signal generator)
sink = uhd.usrp_sink(device_addr=dev_str,
                     channels=range(1))
sink.set_start_time(start_time)
sink.set_samp_rate(op.sample_rate)
sink.set_center_freq(iono_frequencies[0], 0)
code_vector = numpy.fromfile(codefile,
                             dtype=numpy.complex64)
code_vector.tolist()
code_source = gr.blocks.vector_source_c(code_vector,True)
tb.connect(code_source, sink)
tb.start()
```

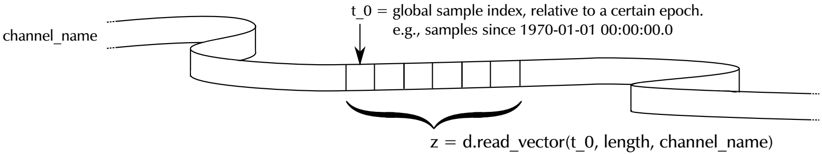
Gnuradio example: transmit waveform

```
# open usrp receiver channel
u = uhd.usrp_source(device_addr)
u.set_samp_rate(op.sample_rate)
u.set_center_freq(op.center_freq, 0)
u.set_gain(op.gain)
# gnuradio flowgraph
fg = gr.top_block()
# boxcar low pass filter
lpf = gr.integrate_cc(op.dec)
# write samples to disk
filesink = juha.filesink(dir_name, gr.sizeof_gr_complex,
int(op.filesize))
# cable up flowgraph
fg.connect(u, lpf, filesink)
fg.start()
```

High level software



Read interface (Digital RF)



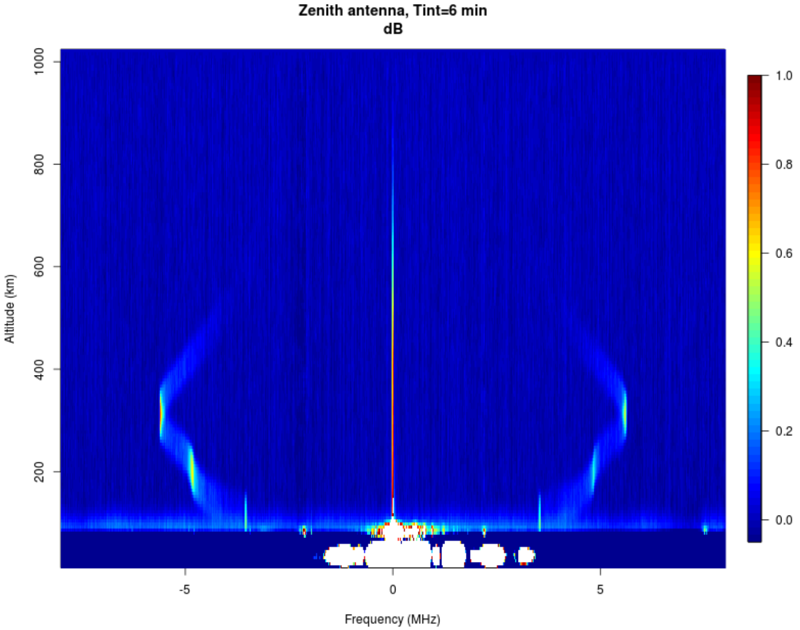
experiment_name-{timestamp}/channel_name/2014-03-30T12:35:29/rf@1396379502.000.h5

http://openradar.org/projects/digital_rf

Digital RF example: simple plasma line spectrum

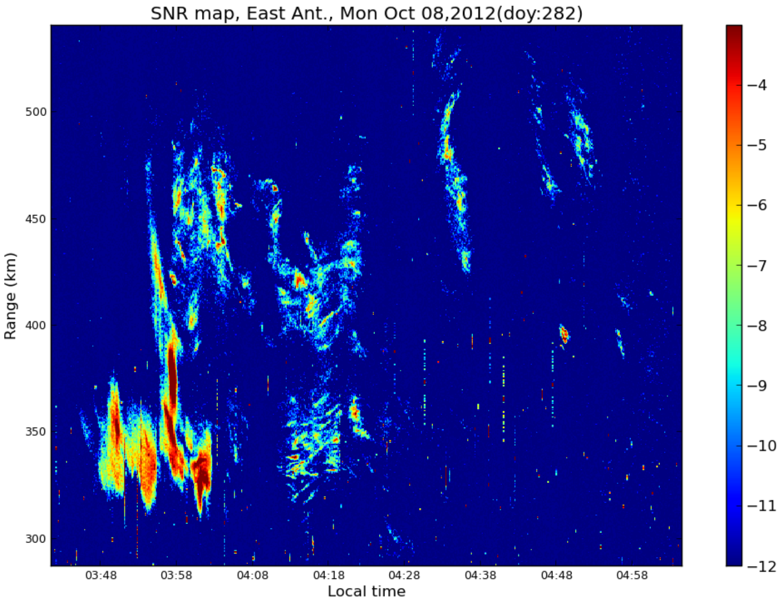
```
for j in range(n_int):
    # read tx pulse
    tx = n.conjugate(d.read_vector(i0 + ipp_idx*ipp_len,
                                   code_len, tx_chan))
    # read echo
    echo = d.read_vector(g, i0 + ipp_idx*ipp_len,
                         ipp_len, rx_chan)
    # multiply transmit pulse with echo and take an fft.
    for k in range(n_ranges):
        gated_echo = echo[idx+int(ranges[k])]
        F = n.fft.fft(tx*gated_echo)))
        spec[:,k] += n.abs(n.fft.fftshift(F)
        ipp_idx += 1
```

Examples: Incoherent scatter receiver



Examples of systems: IRIS radar (Illinois group)

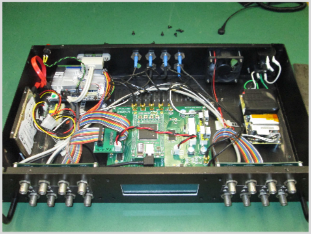
Examples of systems: IRIS radar (Illinois group)

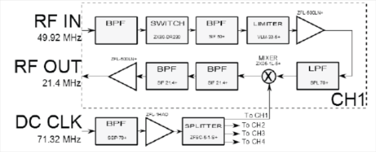


Examples of systems: CIRI (Penn State group)

PENNSTATE

Hardware for CIRI





Examples of systems: CIRI (Penn State group)

PENNSTATE

1 8 5 5

Classification Results

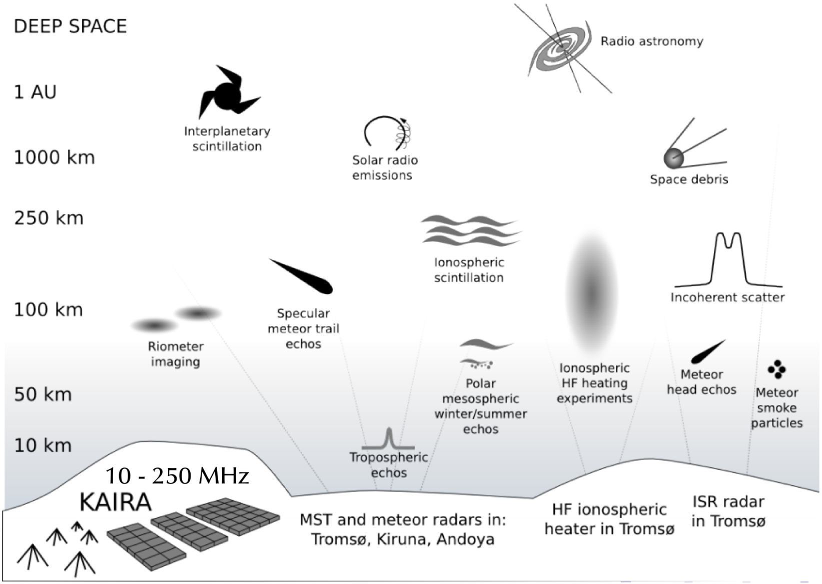
- Classification on feature vectors not used for training
- Decisions made choosing event with maximum probability of occurrence

Class Provided GMM model

True Class of Event

	Spread F	150km Ech	Electrojet
Spread F	**89.66%**	0%	0%
150km Ech	0%	**94.94%**	0%
Electrojet	0%	0%	**94.35%**
Unknown	10.34%	5.06%	5.65%

KAIRA, a LOFAR radio telescope gone geophysical (Vierinen et.al. 2014)



KAIRA, a LOFAR radio telescope gone geophysical (photo: Arttu Jutila)

KAIRA, multistatic ISR (Virtanen et.al. in prep.)

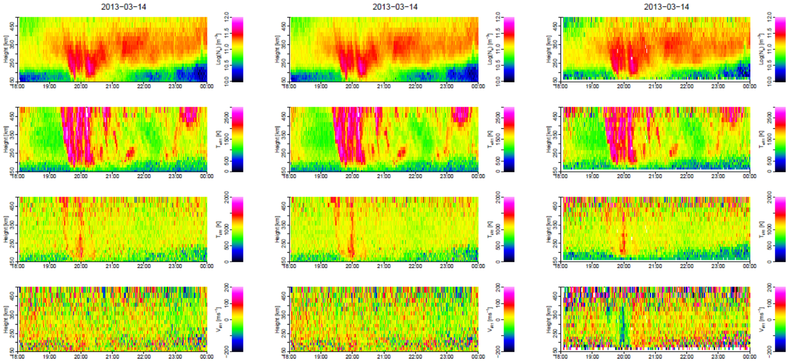
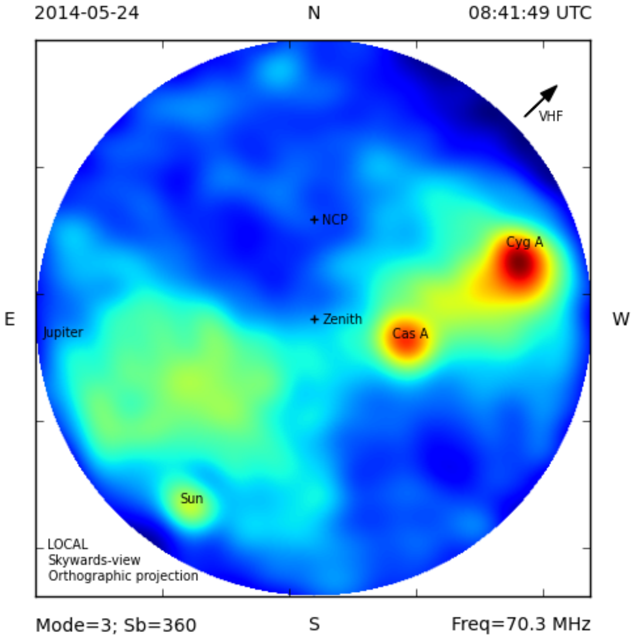
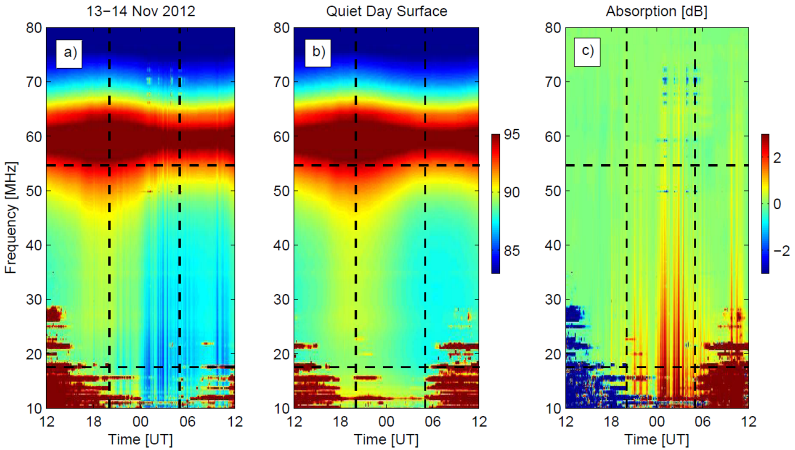


Figure 1. Plasma parameters from EISCAT VHF alone (left), monostatic projections from bistatic analysis to the VHF (middle) and from KAIRA data alone (right).

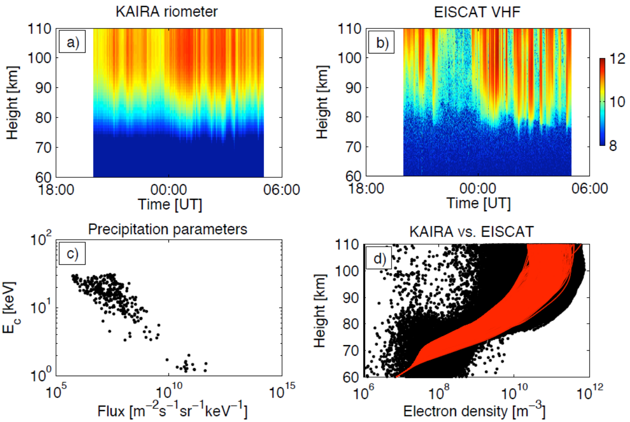
KAIRA: Wide band imaging riometer



KAIRA: Wide band riometer (Kero et.al., in prep)



Wide band riometer (Kero et.al., in prep)



Passive radar

Passive radar (Lind et.al., 2014)

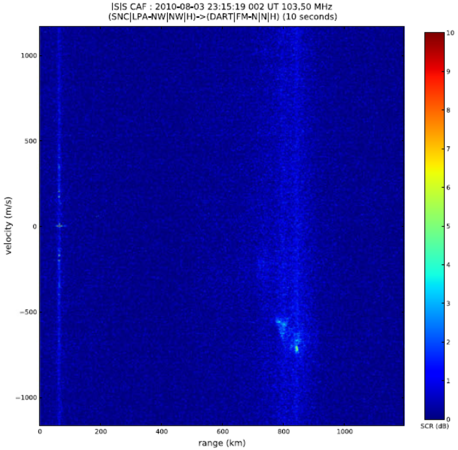
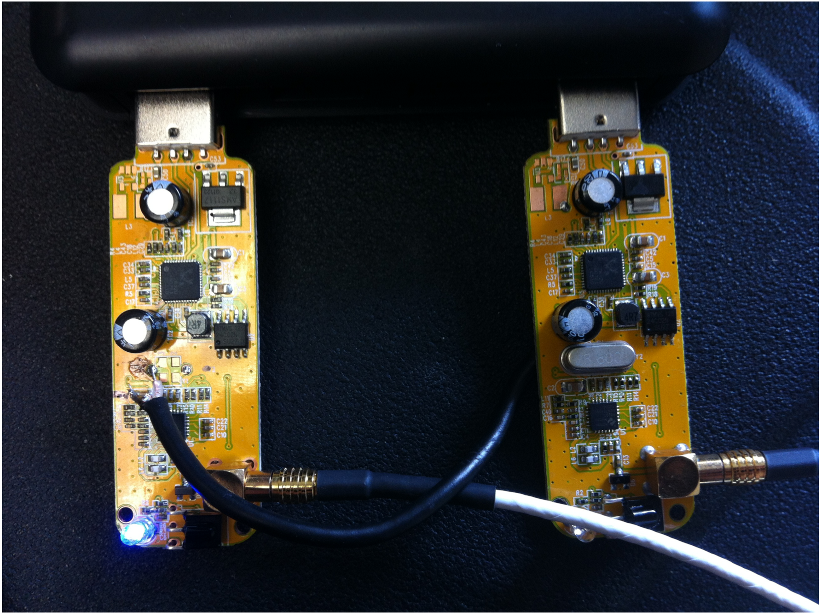
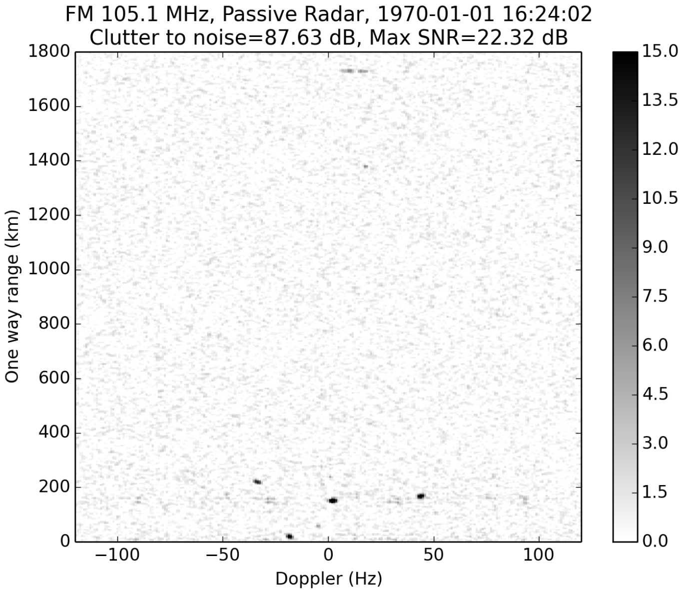


Figure 13. A brief interval of E region irregularities was detected by the ISIS Array on 3 August 2010. The FM passive radar technique allows for the observation of the range, doppler, and time evolution of the irregularities. A cross-ambiguity range doppler plot of the Siena to Dartmouth scattering path is shown with signal to clutter ratio normalized to the median clutter floor. E region irregularities are clearly

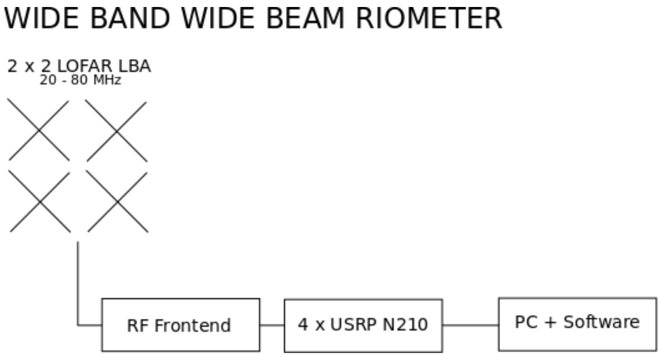
Passive radar with RTLSDR



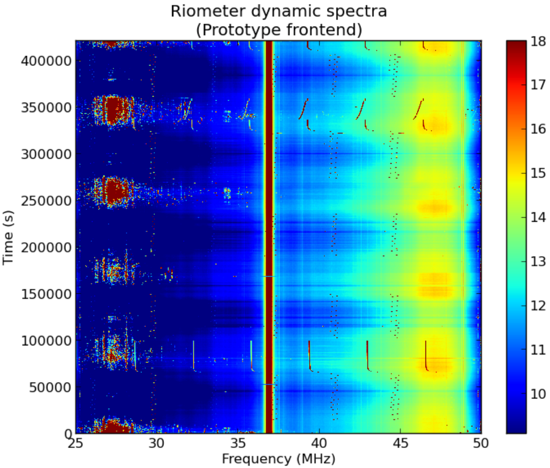
Passive radar with RTLSDR



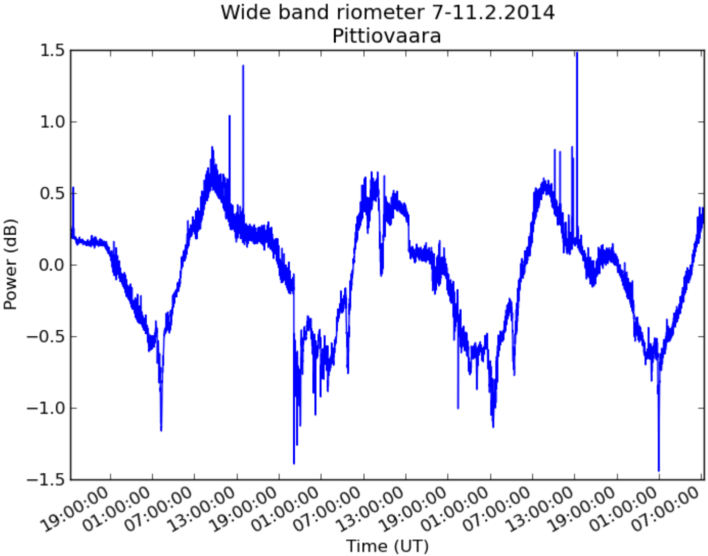
Wide band USRP riometer



Wide band USRP riometer



Wide band USRP riometer



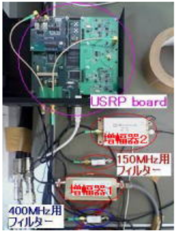
150/400 MHz beacon satellite receiver (Yamamoto 2008)

GNU Radio Beacon Receiver (GRBR)

- GRBR home
- Design information
- Beacon Satellite Info. (TLE file and source)
- Where is GRBR?
- Link
- .
- .
- .

GRBR = GNU Radio & USRP based free receiver for ionospheric TEC measurement

``GNU Radio Beacon Receiver (GRBR)'' is a simple digital receiver developed for the satellite-ground beacon experiment to measure ionospheric total electron content (TEC). The free software toolkit for the software defined radio, GNU Radio, was utilized to define basic function of the receiver, and perform fast signal processing. The software was written in Python for a LINUX PC. The free hardware called Universal Software Radio Pheripheral (USRP), which best matches the GNU Radio, was used as a frontend to acquire the satellite beacon signals of 150 and 400 MHz. This homepage is dedicated to distribute detailed information of the GRBR, and enhance related research activities.



Contact: Mamoru Yamamoto, RISH, Kyoto University (yamamoto@rish.kyoto-u.ac.jp)

150/400 MHz Ionospheric tomography receiver (Vierinen et.al., 2014)

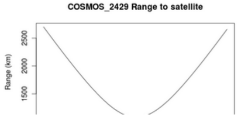
jitter

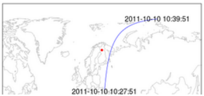
GNU Ionospheric Tomography Receiver

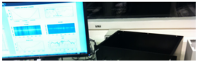
DOWNLOAD CONTACT DOCUMENTATION EXAMPLES

Introduction

The GNU Ionospheric Tomography Receiver (Jitter) is a software package capable of measuring phase curves from dual frequency beacon satellites, such as the Russian Tsykada satellites, DMSP F15, Radcal, or the FORMOSAT fleet. This is done using the gnuradio framework and Ettus Research software defined radio hardware. These phase curves can be used to calculate relative total electron content curves. With a network of multiple receivers, this data can be used as an input to a limited angle tomographic algorithm to produce a tomographic reconstruction of the ionosphere.







150/400 MHz Ionospheric tomography receiver (Vierinen et.al., 2014)

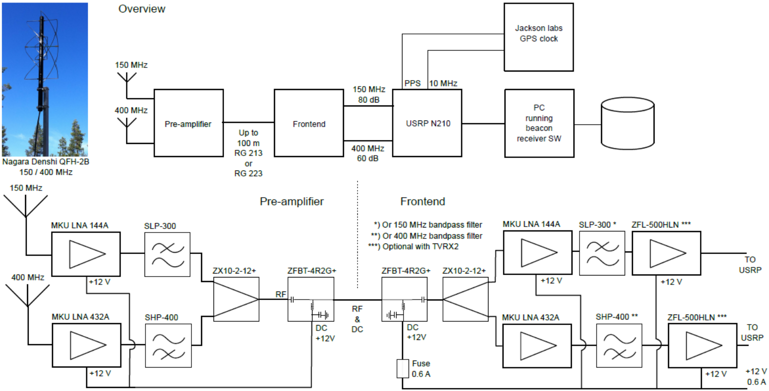
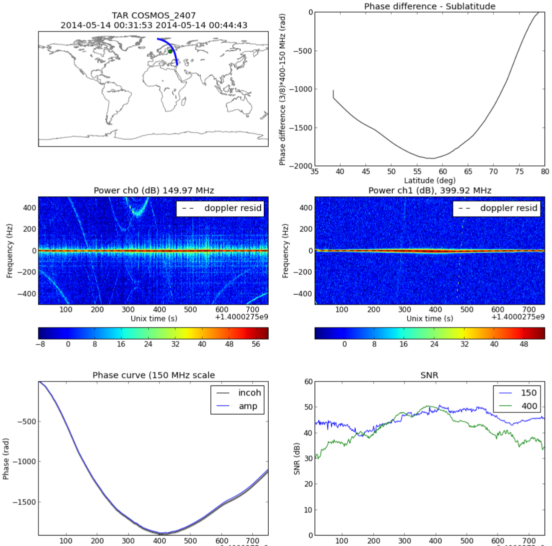
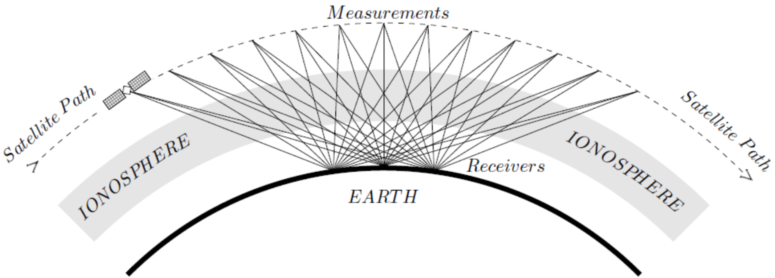


Figure 3. A block diagram of the RF front end used in the FMI beacon receiver chain.

150/400 MHz Ionospheric tomography receiver (Vierinen et.al., 2014)



150/400 MHz beacon satellite receiver (Vierinen et.al., 2014)



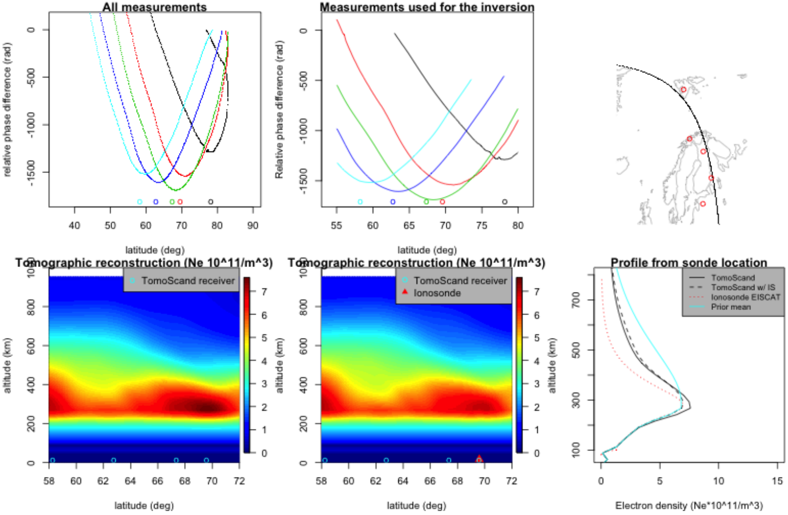
$$m_{t,n} = \frac{1}{N}\sum_{i=1}^{N} N_e(\mathbf{l}_i^{t,n};\theta) + \gamma_n + \xi_{t,n}, \qquad (24)$$

$$\mathbf{m} = \mathbf{A}\mathbf{x} + \xi, \qquad (26)$$

$$\hat{\mathbf{x}}_{\mathrm{MAP}} = \mathbf{\Sigma}_{\mathrm{post}}\mathbf{A}^{\mathrm{T}}\mathbf{\Sigma}^{-1}\mathbf{m}, \qquad (27)$$

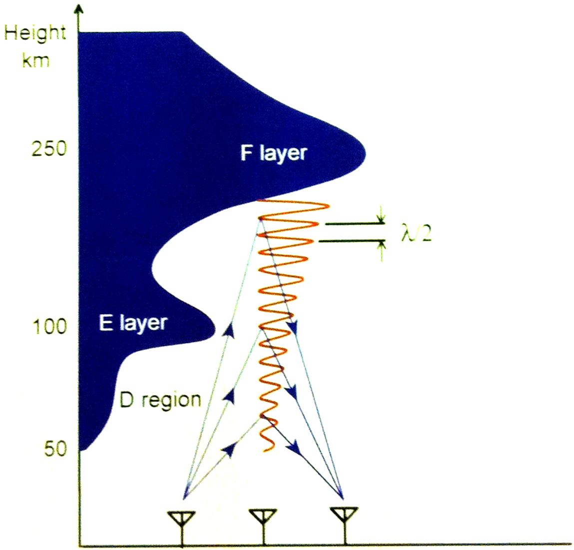
$$\mathbf{\Sigma}_{\mathrm{post}} = (\mathbf{\Sigma}_{\mathrm{pr}}^{-1} + \mathbf{A}^{\mathrm{T}}\mathbf{\Sigma}^{-1}\mathbf{A})^{-1}. \qquad (28)$$

150/400 MHz Ionospheric tomography receiver (Norberg et.al., 2014)

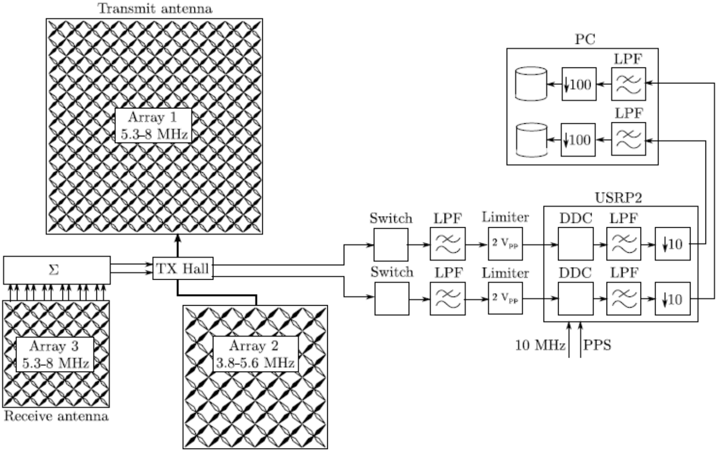


EISCAT HF Heater (600 MW ERP)

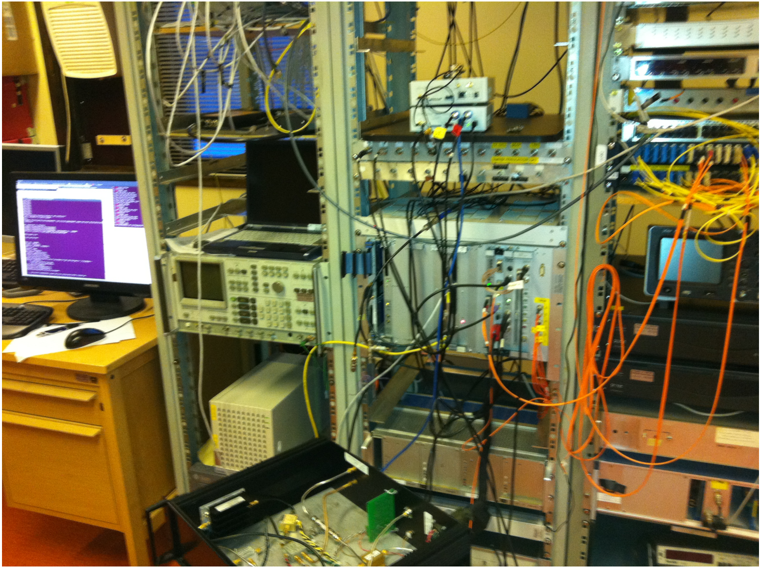
EISCAT HF Heater, Artificial Periodic Irregularity (API) experiment, (Belikovich et.al., 2002)



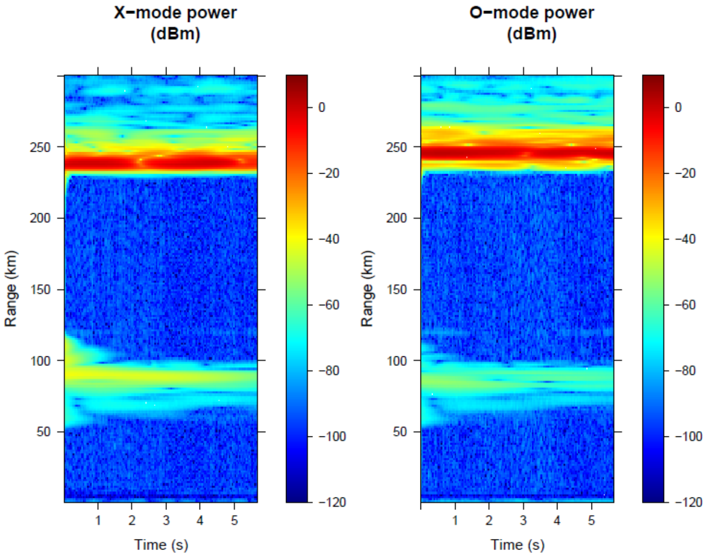
EISCAT HF Heater, API experiment



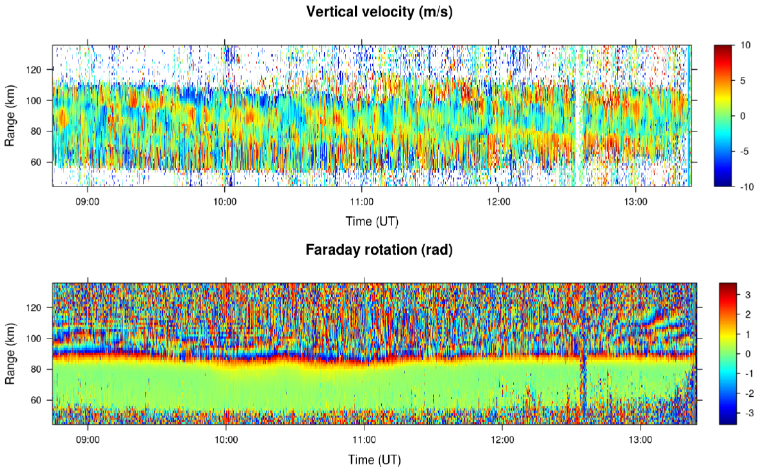
EISCAT HF Heater (600 MW ERP)



EISCAT HF Heater, API experiment



EISCAT HF Heater, API experiment (Vierinen et.al., 2014)



GNU Chirp Sounder

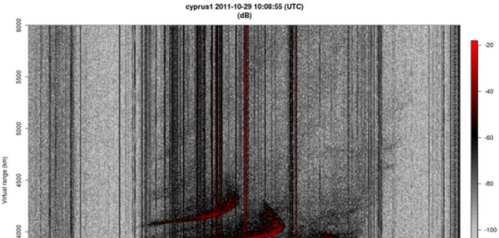
gcs

GNU Chirp Sounder

DOWNLOAD CONTACT USAGE LINKS

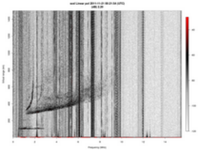
Introduction

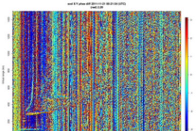
The GNU Chirp Sounder is a software defined radio based receiver for monitoring ionospheric sounders (ionosondes) and over-the-horizon radars that use linear frequency sweep FM-CW transmissions. The software is based on gnuradio and relies on Ettus research USRP2 and USRP N210 based digital receivers. The receiver can be used to receive the whole HF band (typically at 25 MHz bandwidth) simultaneously, and to receive multiple sounders simultaneously. The current receiver can be used to perform single or dual polarization (channel) soundings. The dual channel recorded can be used to determine the polarization form vertical soundings, or for angle of arrival measurements for horizontal soundings.



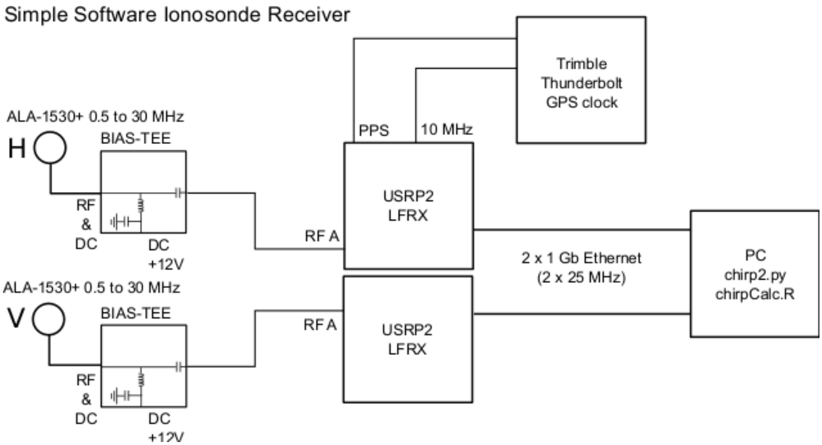
Realtime

These measurements at Sodankylä are real-time. Currently I'm testing a two channel interferometric angle of arrival measurement with the linear polarization antennas spaced 150 meters apart.





GNU Chirp Sounder



GNU Chirp Sounder