April 19, 2022

Software for the Deployable Low-band Ionosphere and Transient Experiment (DLITE)

Joseph Helmboldt, U.S. Naval Research Laboratory, Cod 7213

Introduction

The Deployable Low-band Ionosphere and Transient Experiment (DLITE) is an NRL-developed telescope array optimized for remote sensing of density irregularities within Earth's ionosphere (~100-1,000 km altitude). A DLITE system consists of four inverted vee dipole antennas that were developed and patented by NRL Code 7210 for the Long Wavelength Array (LWA) project (https://lwa.unm.edu). These four antennas are used as an interferometer with a digital processing backend comprised of software defined radios (SDRs) and a control computer. Separating the antennas from one another by 200-500 meters and using a relatively large bandwidth (~8-10 MHz) enables time difference of arrival (TDOA) and frequency difference of arrival (FDOA) methods for resolving 2-4 extremely bright cosmic radio sources that are visible on the sky at any given time. By operating in a 30-40 MHz band where the impact of the ionosphere is relatively large, the array can be used to measure fluctuations in the cosmic radio sources' sky positions and intensities due to ionospheric irregularities on scales of ~1 km or more. These measurements can then be used to characterize the irregularities. Helmboldt et al. (2021) provide a detailed description of the system and analysis methods along with examples using two prototype arrays located at NRL-Pomonkey (DLITE-POM) and in central/western New Mexico (DLITE-NM).

Software

Software written for DLITE falls into two categories: operation and analysis. All code is written in the Python language and utilize external modules/packages. The code that operates the system is called the correlator because it performs cross correlations of signals from all six unique pairs of antennas and coherent averaging of these correlations in real time. The only external package used by the correlator is GnuRadio (https://www.gnuradio.org). The correlator has two versions, dlite_npol_conf.py and dlite_squelch_conf.py, both of which can be run with either Python2.7 or Python3 (currently tested up to Python3.8). The correlator contains a single source block that has eight inputs, which accommodates both polarizations from all four antennas. These are connected to several processing blocks, which ultimately output the cross-correlation data, or "visibilities," to unformatted binary files (32-bit). The dlite_squlech_conf.py version contains extra processing blocks for excising strong interference above a user-defined threshold.

The correlator is run from the command line with a single argument, which is the name of a text file that defines several parameters. The file should have a format like this:

```
BANDWIDTH = 100.e6/12.
NCHAN = 512
FREQ = 35.e6
TINT = 1.024
DURATION = 86164
NPOL = 2
OUTDIR = '/home/dlite/daily'
GAIN = 20
THRESH = [-56,-56,-66,-66,-53,-53,-63,-63]
```

Where BANDWIDTH sets the streaming bandwidth, NCHAN sets the number of frequency channels in the output visibilities, FREQ sets the central frequency, DURATION is the duration in seconds, NPOL is the number of output polarization products (2 or 4), OUTDIR is the name of the directory where the output will be written, and GAIN is the gain in dB for the SDRs. The THRESH parameter is only used by the dlite_squelch_conf.py version and defines the clipping threshold in dB per antenna and polarization. The order is [antenna1-polarization1, antenna1-polarization2, antenna2-polarization1, antenna2-polarization2, antenna3-polarization1, antenna3-polarization2, antenna4-polarization1, antenna4-polarization2]. The correlator creates a directory within OUTDIR, the name of which is the Unix timestamp at the start of the observation. The configuration file is copied to a text file named SETUP within this directory. The binary visibility data are saved within this directory with file names, e.g., visfile_12_xx.bin which in this case is the cross correlation of X (north south) polarization from antennas 1 and 2.

To facilitate analysis of the data output by the correlator, a Python module with several useful functions has been created called dliteTools.py. This module requires the NumPy (https://numpy.org) and Astropy (https://www.astropy.org) packages. The table below gives one-line descriptions of the functions within this module.

References

Helmboldt, J. F., Markowski, B. B., Bonanno, D. J., Clarke, T. E., Dowell, J., Hicks, B. C., et al. (2021). The deployable low-band ionosphere and transient experiment. *Radio Science*, 56, e2021RS007298. https://doi.org/10.1029/2021RS007298

Function	Required arguments	Optional arguments (with defaults)	Description
detrend	t x w	nfit=3	De-trends time series of x(t) by subtracting linear fit within slideing widow of width w. Returns de-trended data.
tstamp2lst	lng ts		Converts Unix time stamps (ts) to local sidereal time for given longitude (lng). Returns array of LST.
tstamp2ut	ts		Converts Unix time stamps (ts) to universal time. Returns array of UT.
lstsq	x y	yerr=None	Performs linear least-squares fit with independent variable(s) in matrix x and observed values in y and (optional) errors in y (yerr). Returns coefficients and if yerr is not None errors.
spline	x y	yp1=1.e30 ypn=1.e30 nmax=100000	Computes cubic spline.
splint	xa ya y2a x		Function required by spline function.
cspline	x xa ya		Perform cubic spline interpolation of data given in xa and ya onto specified locations x. Returns array of interpolated values.
comp_altaz_ha	ha dec lat		Compute altitude and azimuth of a source from its hour angle (ha) and declination (dec) at the latitude (lat) of the telescope. Returns altitude and azimuth arrays.
comp_hadec	alt az lat		Oppolsite of comp_altaz_ha.
deriv	x y		Numerically compute derivative of y(x).
read_file_sink	infile lfft		Read data from a binary file (infile) output by the DLITE correlator with the number of frequency channels specified (lfft). Returns 2-D array of complex numbers.
read_vis	indir	na=4	Reads the visibility data from a directory created by the DLITE correlator (indir) for a specified number of antennas (default na=4). Returns a 4-D array: Nb x Npol x Nt x Nchan.
mad	x		Returns the median absolute deviation for array x.
flag_vis	vis	wch=8 wt=60 flim=7	Flag visibilities in array vis. The window used for statistics is wch frequency channels wide and wt integration times long. Amplitudes that are flim times the MAD larger than the median are flag. Returns array of flags (1=flagged; 0=un-flagged).
rfisub	vis	wdetrend=59	Subtracts stationary interference with sliding window of width = wdetrend.
delay_spec	vis	flg=None hamm=True	Produces complex spectrum from visibility array vis and (optional) flag array flg that converts frequency to delay (will use Hamming window if hamm=True).
bispec	vis		Computes the mean bispectrum for visibility array vis.
read_ant	infile	aref=1	Reads antenna positions from an antenna text file and converts them to linear positions relative to antenna aref.
get_tstamps	indir		Returns the Unix timestamps for visibility data in directory indir.
ateam_dircos	antfile tstamps		Computes direction cosines for the A-Team for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 2-D arrays of direction cosines ll (=cos(alt.)*sin(az.)); mm (=cos(alt.)*cos(az.)); and nn (=sin(alt.)).
ateam_delays	antfile tstamps		Computes antenna delays for the A-Team for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 2-D array of delays.
sun_dircos	antfile tstamps		Computes direction cosines for the Sun for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 1-D arrays of direction cosines ll (=cos(alt.)*sin(az.)); mm (=cos(alt.)*cos(az.)); and nn (=sin(alt.)).
sun_delays	antfile tstamps		Computes antenna delays for the Sun for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 1-D array of delays.
jupiter_dircos	antfile tstamps		Computes direction cosines for Jupiter for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 1-D arrays of direction cosines ll (=cos(alt.)*sin(az.)); mm (=cos(alt.)*cos(az.)); and nn (=sin(alt.)).
jupiter_delays	antfile tstamps		Computes antenna delays for Jupiter for a given antenna file (antfile) and Unix timestamps (tstamps). Returns 1-D array of delays.
fringe_rate	d tstamps frq		Returns array of fringe rates (in Hz) for array of delays (d) timestamps (tstamps) and frequency (frq; in Hz).
dircos2ipp	ll mm nn	lat=38.56 lng=-77.054 zion=300.	Convert direction cosines (ll mm nn) to ionospheric pierce point locations for a telescope located at latitude lat and longitude lng and for a height of zion. Returns arrays of pierce-point latitudes and longitudes.
dircos2range	ll mm nn	lat=38.56 lng=-77.054 zion=300.	Convert direction cosines (ll mm nn) to distance from a telescope located at latitude lat and longitude lng and for a height of zion. Returns array of distances.