The Journal of Open Source Software

Virgo: A Versatile Spectrometer for Radio Astronomy

Apostolos Spanakis-Misirlis[1], **Cameron L. Van Eck**[2], **and E.P. Boven**[3]

1 Department of Informatics, University of Piraeus, Greece **2** Dunlap Institute for Astronomy and Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S 3H4, Canada **3** C.A. Muller Radioastronomie Station (CAMRAS), Dwingeloo, the Netherlands

DOI: 10.21105/joss.03067

Software

- Review
- Repository
- Archive

Editor: Juanjo Bazán

Reviewers:

- @astrom-tom
- @ygrange

Submitted: 25 January 2021
Published: 01 June 2021

Introduction

For the past few decades, radio astronomy has been a rapidly developing area of observational astronomy. This is due to the fact that a variety of celestial objects emit electromagnetic radiation at radio wavelengths, which has led to the development of radio telescopes capable of revealing the otherwise-hidden astrophysical properties of the universe. An important requirement that makes radio astronomy observations and analysis possible is an appropriate software pipeline compatible with the spectrometers with which radio observatories are equipped. In this work, we present `Virgo`: a versatile software solution for radio telescopes.

Statement of Need

`Virgo` is a Python package for the acquisition, processing and analysis of data from radio telescopes. It is an easy-to-use open-source spectrometer and radiometer based on the GNU Radio framework (https://www.gnuradio.org), and is conveniently applicable to any radio telescope working with a GNU Radio-supported software-defined radio (SDR; a radio receiver architecture where some conventional hardware-based steps are replicated in software, Markus Dillinger (2003)). Although in-house acquisition solutions have been independently created by many, composing their own software for simple observations, these are not comparable in terms of installation count, ease of use, cross-platform support, functionality, reproducibility, maintenance and level of documentation that this package has to offer.

Designed to be used by students, educators and amateurs in the field of radio astronomy, `Virgo` has already been adopted by a number of small and large-aperture radio telescopes, permitting both spectral and continuum observations with great success. These instruments include the ISEC TLM-18 (18m), the ACRO RT-320 (3.2m), the JRT (1.9m), and the PICTOR Telescope (1.5m), among others.

Although the hardware aspect of a radio telescope is generally handled by newcomers with relative ease, the skill set needed to integrate a complete software pipeline to support observations is not something most users are equipped with. `Virgo` tackles this problem by providing non-experts with a tool to collect and interpret data from radio telescopes, without requiring expertise in digital signal processing and software engineering. An example use case is classroom experiments in which students build a small-aperture antenna connected to a low-noise amplifier followed by an SDR, and with the help of `Virgo`, obtain data to map out the galactic distribution of neutral hydrogen and/or derive the rotation curve of the Milky Way. An example observation of the 21-cm hydrogen line acquired and processed with `Virgo` is shown in Figure 1. The package's versatility also provides a convenient solution for researchers wishing to rapidly deploy low-cost radio telescopes with commercial hardware.

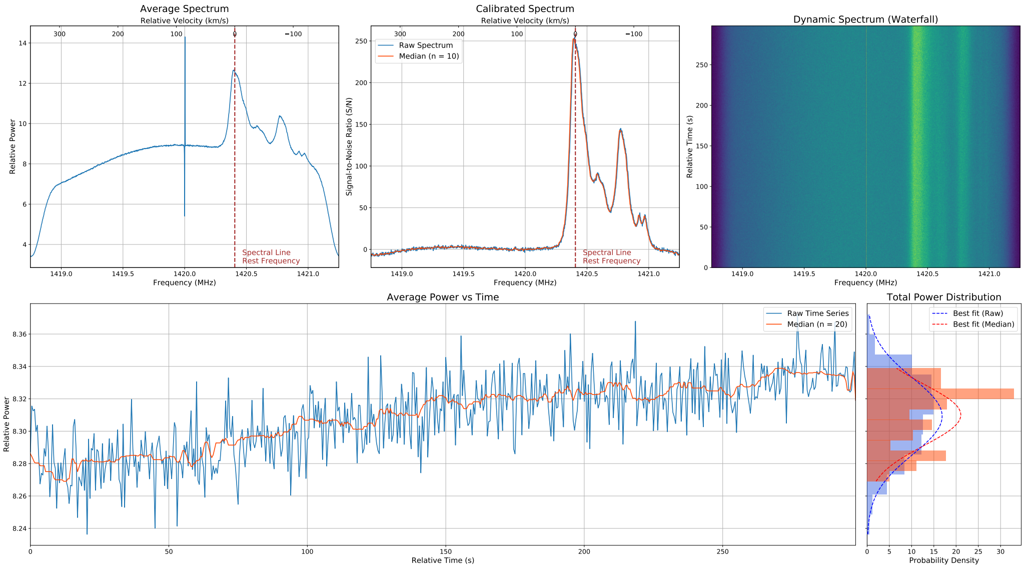

Figure 1: Observation of galactic clouds of neutral hydrogen toward the constellation of Cygnus ($\alpha = 20^{\mathrm{h}}$, $\delta = 40^{\circ}$, $l = 77^{\circ}$, $b = 3^{\circ}$), observed by the TLM-18 Telescope in New Jersey, U.S. with `Virgo`. The average spectrum (top left), the calibrated spectrum (top center), the dynamic spectrum (top right) and the time series along with the total power distribution (bottom) are all plotted by the software automatically.

Features

One of the key features of `Virgo` is its implementation of a four-tap weighted overlap-add (WOLA) Fourier transform (FT) spectrometer, offering a significant reduction in spectral leakage compared to a simple FT filterbank spectrometer that does not make use of the WOLA method, with a minimal increase in computational requirements (Crochiere & Rabiner, 1996)[1]. In addition to its data-acquisition functionality that performs data reduction by time-averaging spectra in real time, `Virgo` also carries out automated analysis of the recorded samples. The time-averaged spectrum, the calibrated spectrum, the dynamic spectrum (waterfall), the time series (power vs time) and the total power distribution of the observation are all automatically computed and plotted with the help of the `Numpy` (Harris et al., 2020) and `Matplotlib` (Hunter, 2007) packages.

Because of the nature of the RF instrumentation that radio telescopes are equipped with, the spectra acquired by SDRs have an unwanted frequency-dependant sensitivity, also known as the bandpass shape. In general, this frequency response makes it difficult to distinguish true signals, originating from the sky, from instrumentation artifacts. For that reason, `Virgo` performs bandpass calibration by taking the ratio of the observed spectrum over the calibration spectrum. However, because this ratio is arbitrarily scaled (due to the difference in the noise floor levels), the power axis is automatically rescaled to units of signal-to-noise ratio. In case the resulted spectrum has an unwanted slope due to e.g. inconsistent conditions between the calibration and the observation, the software can also automatically correct poorly-calibrated spectra using linear regression.

Furthermore, `Virgo` supports optional median operations, both in the frequency and time domain, for the suppression of narrowband and/or short-duration radio frequency interference (RFI), while allowing the user to export the raw observation data as a FITS (Pence et al.,

[1] The package also supports a plain FT filterbank pipeline for observatories with limited computational resources.

2010) or csv-formatted file. Frequencies contaminated with RFI may also be stamped out with the software's built-in channel masking capability.

Lastly, the package may also be used for the detection of giant pulses (irregularly intense bursts of radio emission by pulsars). Due to the frequency-dependent dispersion caused by the plasma distribution in the interstellar medium (ISM), observed pulses naturally appear smeared in time, depending on the dispersion measure (integrated column density of free electrons from the observer to the source). To prevent implied degradations of the signal-to-noise ratio, incoherent dedispersion is optionally applied to dynamic spectra of pulsar observations, compensating for the unwanted dispersive effects of the ISM.

By additionally providing the observer with an important set of utilities, `Virgo` also makes for a great tool for planning (radio) observations. This includes the ability to compute the position of astronomical sources in the sky for a given date (see Figure 2), and conversely, to estimate the right ascencion and declination given the observer's coordinates along with the altitude and azimuth the telescope is pointing to, with the help of the `Astropy` package (Astropy Collaboration et al., 2018, 2013). On top of that, the package comes with a basic set of calculators for quickly carrying out a variety of computations involving the theoretical sensitivity and performance of a given instrument.

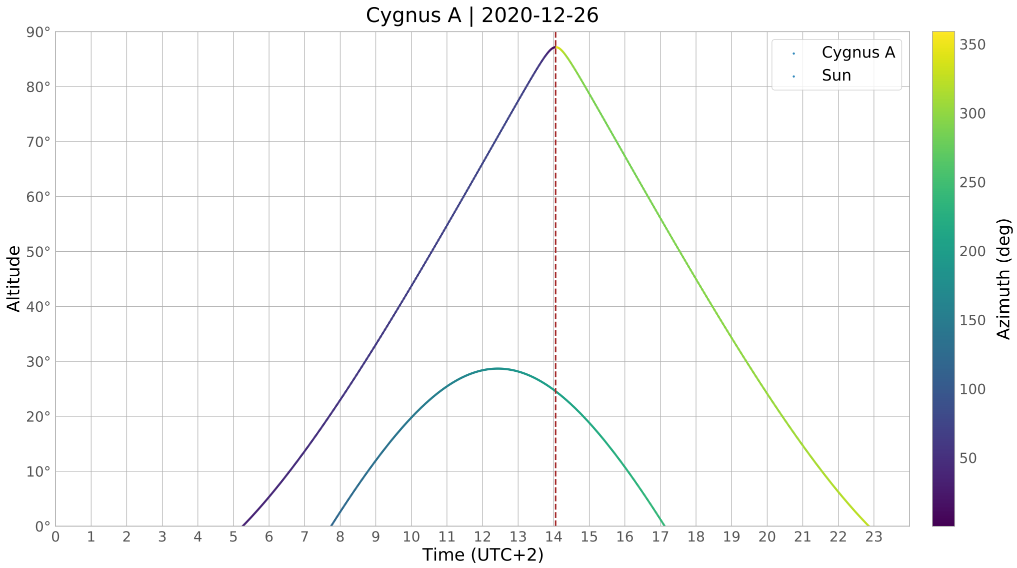

Figure 2: Example prediction of the location of the Cygnus A radio galaxy (3C 405) in the celestial sphere of the observer.

Likewise, the software provides a handy tool for retrieving HI profiles based on the Leiden/Argentine/Bonn (LAB) Survey of Galactic HI (Kalberla et al., 2005). These spectra (see Figure 3 for an example) can be correlated with the integrated 21 cm all-sky map previewer shown in Figure 4.

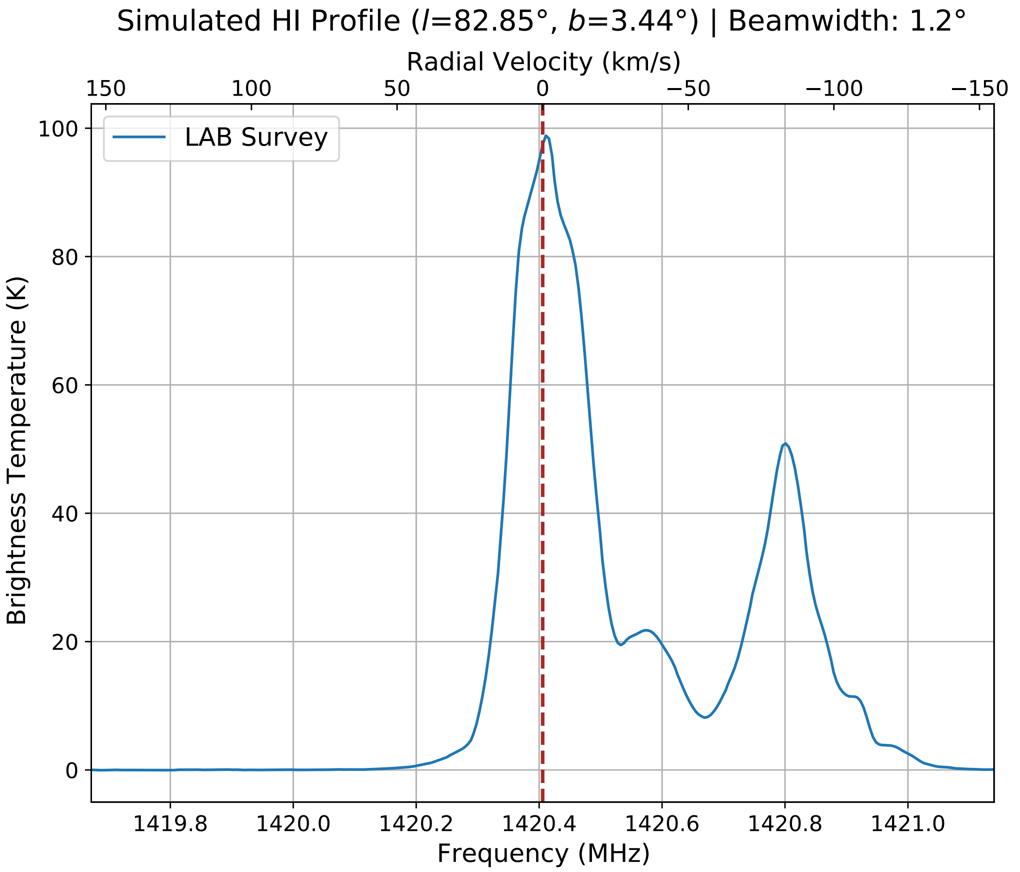

Figure 3: Sample HI profile ($\alpha = 20^{\mathrm{h}}30^{\mathrm{m}}$, $\delta = 45^{\circ}$) obtained with the package's `virgo.simulate()` function.

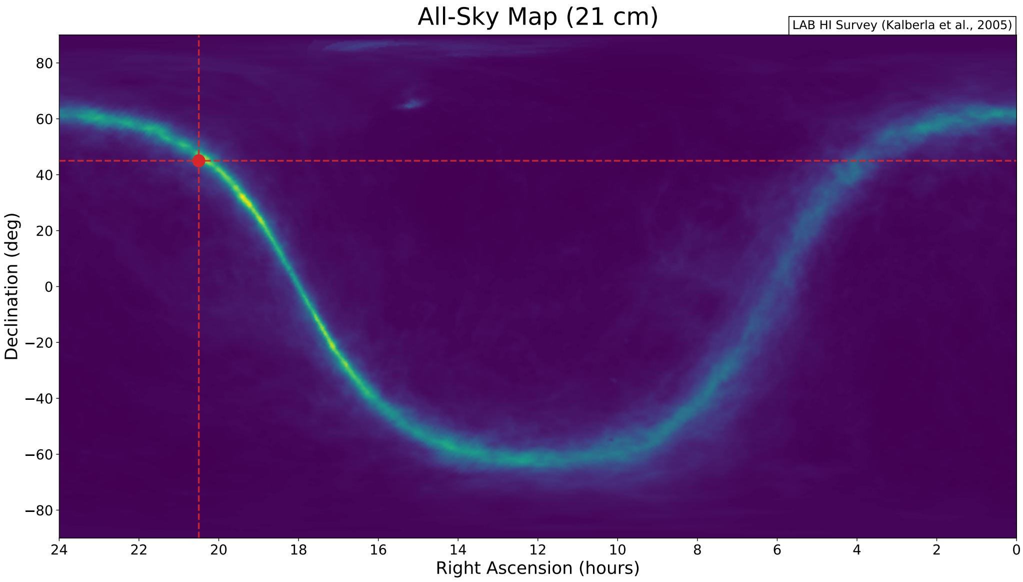

Figure 4: 21 cm all-sky map rendered by the software. The red dot indicates the position of the telescope's beam in the sky, provided by the user.

Moreover, the package comes with an integrated frequency-domain RFI measurement pipeline, allowing observers to rapidly carry out a survey outlining the compatibility of the telescope's environment with radio observation standards.

Last but not least, the software's modularity allows users to effortlessly integrate Virgo's functionalities into other pipelines, permitting a variety of automation applications invloving the acquisition and/or processing of telescope data.

Example Usage

Virgo can either be called directly from the command line using e.g.,

`virgo -rf 10 -if 20 -bb 20 -f 1420e6 -b 5e6 -c 2048 -t 1 -d 60 -s 10 -n 20 -m 35 -r 1420.4057517667e6 -C calibration.dat -W obs.fits`,

or imported and used as a package:

```python
import virgo

# Define observation parameters
obs = {
    'dev_args': '',
    'rf_gain': 10,
    'if_gain': 20,
    'bb_gain': 20,
    'frequency': 1420e6,
    'bandwidth': 5e6,
    'channels': 2048,
    't_sample': 1,
    'duration': 60
}

# Check source position
virgo.predict(lat=39.83, lon=-74.87, source='Cas A', date='2020-12-26')

# Begin data acquisition in 10 sec
virgo.observe(obs_parameters=obs, obs_file='observation.dat', start_in=10)

# Analyze data, mitigate RFI and export the data as a FITS file
virgo.plot(obs_parameters=obs, n=20, m=35, f_rest=1420.4057517667e6,
           obs_file='observation.dat', cal_file='calibration.dat',
           rfi=[1419.2e6, 1419.3e6], waterfall_fits='obs.fits',
           slope_correction=True, plot_file='plot.png')
```

Future Work

As Virgo gets adopted by more and more radio telescopes, the need for expanding the software's capabilities grow. Additional features that have been proposed include but are not limited to a more robust and intelligent system for the detection and mitigation of RFI, and the support for a data acquisition/analysis pipeline for pulsar astronomy, both of which offer the potential of appealing to a broader user audience.

References

Astropy Collaboration, Price-Whelan, A. M., SipHocz, B. M., G"unther, H. M., Lim, P. L., Crawford, S. M., Conseil, S., Shupe, D. L., Craig, M. W., Dencheva, N., Ginsburg, A., Vand erPlas, J. T., Bradley, L. D., P'erez-Su'arez, D., de Val-Borro, M., Aldcroft, T. L., Cruz, K. L., Robitaille, T. P., Tollerud, E. J., … Astropy Contributors. (2018). The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. *Aj*, *156*(3), 123. https://doi.org/10.3847/1538-3881/aabc4f

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., Greenfield, P., Droettboom, M., Bray, E., Aldcroft, T., Davis, M., Ginsburg, A., Price-Whelan, A. M., Kerzendorf, W. E., Conley, A., Crighton, N., Barbary, K., Muna, D., Ferguson, H., Grollier, F., Parikh, M. M., Nair, P. H., … Streicher, O. (2013). Astropy: A community Python package for astronomy. *558*, A33. https://doi.org/10.1051/0004-6361/201322068

Crochiere, R. E., & Rabiner, L. R. (1996). *Multirate digital signal processing* (1st ed.). Prentice Hall PTR. ISBN: 0136051626

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Kalberla, P. M. W., Burton, W. B., Hartmann, D., Arnal, E. M., Bajaja, E., Morras, R., & Pöppel, W. G. L. (2005). The leiden/argentine/bonn (LAB) survey of galactic HI. *Astronomy & Astrophysics*, *440*(2), 775–782. https://doi.org/10.1051/0004-6361:20041864

Markus Dillinger, N. A., Kambiz Madani. (2003). (Vol. 1). John Wiley & Sons. ISBN: 0470851643

Pence, W. D., Chiappetti, L., Page, C. G., Shaw, R. A., & Stobie, E. (2010). Definition of the Flexible Image Transport System (FITS), version 3.0. *524*, A42. https://doi.org/10.1051/0004-6361/201015362