ngEHT Requirements for a new HOPS

John Barrett, Geoff Crew, Dan Hoak and Violet Pfeiffer
MIT Haystack Observatory

Version 2.1, March 2, 2021

Executive Summary

This document spells out the requirements for the planned re-work of HOPS. With a defined set of requirements (Section 2), we can ensure that the new architecture and implementation support the needs of the ngEHT. We will generate a collection of test reports to demonstrate adherence to the defined requirements.

It is desirable to produce a suite of software that not only meets the stated needs, but which is also easier to maintain, and is more extensible for the future. These goals are not stated as requirements in this document since it is not possible to test against them. These efforts will therefore be described elsewhere.

Contents

1 Introduction

The next-generation EHT (ngEHT) is a Mid-scale Research Infrastructure (MSRI) project funded by the National Science Foundation (NSF), with the goal of developing transformative technologies for a second-generation Event Horizon Telescope (EHT). A significant component of the telescope is the software needed to properly operate, reduce and analyze the data taken by the member telescopes. This document defines the requirements for the refactoring of the Haystack Observatory Postprocessing System (HOPS) as part of the work package of the MSRI proposal.

Broadly speaking, the refactoring of HOPS is driven by these overarching desires:

- Everything currently possible in HOPS shall remain possible.
- The new HOPS shall be easier to use than the existing HOPS.
- The new HOPS shall support the required expansion of stations, baselines, and bandwidth expected for the ngEHT.

The next generation EHT is planned to include up to 30 stations and 435 baselines, with wider bandwidth (up to 256 Gbps over four dual-polarization, 4 GHz bands, if sampled at 2-bits). Support for greater bit depth is also of interest should recording media be available. The refactoring of HOPS shall support the full analysis of ngEHT data within these limits, and will be made smarter, more automatic, more robust, and easier to use. The number of channels shall have no practical limit, but shall be tested at least to 128.

For the needs of the EHT campaigns of 2017 it was decided to augment the existing HOPS package with some Python-based packages in order to create a pipeline for the initial reduction of data [1]. The new HOPS shall also support independent analyses, using Python or other languages, through accessible data archive formats (e.g. HDF5, JSON, XML, or similar), interactive tools, and conversion to human-readable ASCII files.

HOPS is currently in the version 3.x series; the new code will begin with the 4.x series, and we shall refer to the new codebase as HOPS4 (and the old as HOPS3). The primary development language will be C/C++, with a Python scripting layer to provide ease of use. Version control shall be provided by the MIT-hosted github.

This document is intended to be one in a series, with other documents describing:

- Specifications and architectural design [2]
- Coverage and test plan [3]
- The development plan [4]
- The user manual
- Function and class references, generated by e.g. Sphinx or Doxygen

Additional documents may be written during the course of the project.

2 Requirements

The requirements defined in this section are loosely organized by functionality. Requirements are numbered with a prefix according to the type of test that will be used to validate them. These prefixes are:

A, for automated tests. These are requirements that shall be validated with a "check" script (in shell, Python, or similar). These scripts can be executed during installation steps to verify the installation was successful, or in a nightly build to verify that daily updates have not broken any functionality. Existing tests within HOPS3 will be ported into the new HOPS4 code set.

S, for software-testable. These are more complicated tests such as regression tests that may be run at specific points in development. It is anticipated that each of these tests will be implemented by a script, and a report provided.

P, for procedural tests that require human interpretation (*e.g.* validating the display mechanics of a figure). These tests shall have explicit instructions for the user and the results shall be recorded.

D, for desires. These are not true requirements, but are general design goals for which testing does not apply. They are listed here as they are inputs the development plan.

Details of the testing process will be made available in the coverage and testing document [3].

2.1 General Requirements

The following requirements describe essential design features.

S-1 HOPS4 shall provide effectively equivalent results for any output that HOPS3 was capable of producing.

Rationale*: To ensure that HOPS4 replicates the existing functionality of HOPS3 except where otherwise specified. HOPS4 will replace HOPS3*

comment*: A set of historical DiFX output has been captured and HOPS4 results will be periodically compared to HOPS3 results on observables that HOPS3 supports to ensure consistency.*

A-2 HOPS4 shall have no practical limit on the number of stations, baselines, channels, or accumulation periods other than which is imposed by the memory limitations of the computer hardware on which it is run.

Rationale*: To adapt to changing specifications in radio science some of which are TBD.*

A-3 HOPS4 shall support at least 30 stations and 435 baselines.

input required*: HOPS4 requires an upper bound merely to verify that it supports "more* *than enough" stations.*

A-4 HOPS4 shall (1) not impose a limit on the number of channels/sub-bands, and (2) will explicitly test that the software functions with at least 128 channels.

***input required**: HOPS4 requires an upper bound merely to implement with some sensible channel labeling scheme and verify that the requirement is satisfied.*

A-5 HOPS4 shall support a (raw station data) bit depth of 2 bits.

***input required**: Support for 1-bit is available in HOPS3, and hooks for other bit depths will be provided, but the code for other depths will require some development in order to ensure proper normalization and treatment of noise statistics, etc.*

A-6 Every library (C/C++) or module (python) shall have a corresponding test suite to verify expected functionality. This may include, but is not limited to, unit tests (for individual classes/functions) with the appropriate test cases to be determined.

***Rationale**: In keeping with general software development best practices, preserving code modularity and maintaining associated testing is important in detecting and easily repairing bugs.*

The existing package has a number of dependencies which are becoming a challenge to maintain—PGPLOT is such an example. Ideally HOPS4 should be installable and usable without any dependencies. (However, we shall not require that the software provide complete functionality on a computer lacking such optional dependencies.) The build machinery should test for the presence of these packages and make appropriate accommodation for the Linux distribution. Note that we will only support three major distributions, Debian/Ubuntu, Fedora, and NixOS. Testing will be done frequently for only these three distributions. There are no plans to officially support other distributions, Mac OSX, or Windows.

P-7 HOPS4 shall operate without loss of functionality in the absence of FFTW3, and/or GSL, with the understanding that performance (runtime) might be affected.

***comment**: For example FFT calls should be wrapped so that other packages or native code may be used instead.*

***Rationale**: As part of software development best practices it is important to write software that is modular, so that individual libraries can be replaced as needed. This is important for both long term maintenance and for testing in environments which may temporariliy not have all the software dependencies.*

P-8 PGPLOT shall not be required for basic functionality (*e.g.* fringe-fitting). HOPS4 shall include the same plotting functionality as HOPS3, but using standard modern packages (*e.g.* Python, Matplotlib). The user should be able to substitute their own plotting routines as desired, and PGPLOT will be optionally supported for some plotting/display options.

***Rationale**: PGPLOT is a dependency in HOPS3 that has been deprecated considering the core maintainer stopped supporting it in 2000. It causes build stability issues in HOPS3.*

Thus all plotting package calls need to be managed within an optional plugin which leaves core functionality unaffected by its presence. At a minimum, plotting functionality similar to the current fringe-plots shall be constructed with a set of independent Python tools.

2.2 File I/O and Supported File Types

A-9 HOPS4 shall support input from the DiFX [5, 6] correlator (in Swinburne format).

A-10 every application should provide `--help` and `--version` responses (with zero exit status) to behave as conventional GNU/Linux applications.

Rationale: *Command-line arguments should be implemented with a standard argument parsing library for each application distributed with this project, and should respond helpfully when given bad command-line syntax. This aligns with development best practices for Linux applications.*

A-11 HOPS4 shall support input from VEX version 1.5.[1]

Rationale: *Whether or not VEX V.1.5 is to be deprecated is TBD by the astronomy community.*

A-12 All plotting tools provided in HOPS4 shall have the ability to save the image to a standard graphics format.

Rationale: *HOPS4 has the desire to support as much plotting flexibility as possible, including different plot formats (fonts, panels, etc), but at a minimum is must support generating useful and readable plots.*

D-13 All new native data types (to be detailed in the specification document) shall be stored **little-endian!**.

Rationale: *Legacy data types are stored using big-endian which is a deprecated format in the vast majority of modern computing systems and results in unecessary byte-flipping on each read/write.*

D-14 HOPS4 shall provide an application to support export of native data types to a TBD archival format (*e.g.* HDF5).

Rationale: *This is to ensure long-term compatibility with the data storage devices we have at our disposal and to provide users with a path for data access outside the HOPS4 toolset.*

D-15 HOPS4 shall incorporate a C-library to interface with the legacy Mk4 data types (stored **big-endian!**) in their current form for testing purposes, and in order to access archived data.

Rationale: *This is needed to ensure backward compatibility and enable regression testing.*

[1] VEX 1.5 is the only extant working file format for VLBI experiment description, however, there does exist a VEX 2.0 specification which is not currently in active use but may be adopted at any time.

2.3 A-tools

The following requirements are related to the "A-suite" tools: alist, aedit, and adump. The command-line interface for these tools shall be wrapped in Python in HOPS4 and shall support alist file version 6[2].

A-16 The alist program shall generate validly formatted ASCII A-list files in version 6.

***Rationale**: A-list files are an ASCII tab-separated column formatted file type which serve as a summary table containing the results of the `fourfit` post-processing for each scan/baseline. Compatibility with the existing version is necessary for backwards compatibility and regression testing.*

A-17 The adump program shall provide ASCII text representations of columns of data from an A-list of version 6.

***Rationale**: The ASCII text representations are necessary to provide a human readable format, and adump extracts useful subsets of the data for further analysis.*

A-18 The aedit program shall process version 6 A-list files with respect to flagging, selecting, summarizing and generating a new output A-list.

***Rationale**: Compatibility with the alist version 6 is necessary for backwards compatibility and regression testing.*

2.4 Post Processing and Fringe Plots

The following requirements describe wrappers and tools in HOPS3 for post-processing and examination of the fringe results. They shall be maintained in HOPS4.

A-19 HOPS4 shall maintain the fourmer tool, which shall properly relabel channels when it combines two sub-bands.

***Rationale**: This application must be ported to HOPS4 to maintain current functionality.*

A-20 The fringex program shall retain existing functionality to rotate fringes with respect to Fourfit fringe solutions.

***Rationale**: This application must be ported to HOPS4 to maintain current functionality.*

A-21 The "average" capability, implemented by average used in concert with fringex to subdivide explore and average fringes, shall be preserved with equivalent functionality.

***Rationale**: This application must be ported to HOPS4 to maintain current functionality.*

A-22 The functionality of the program cofit shall be preserved.

***Rationale**: This application must be ported to HOPS4 to maintain current functionality.*

[2] Version 6 is the standard for the EHT, and a version 7 format seems sensible but is not yet required. If in detailing the specifications a version 7 (or 8, *&c.*) emerges, these requirements transfer to the those version(s) as well.

A-23 The functionality of the program search shall be preserved.

Rationale: *This application must be ported to HOPS4 to maintain current functionality.*

A-24 The ability to explore fringes (as is currently done with the combination of fringex, average and search must be preserved. HOPS4 shall support Python scripting to aid the user in searching fringe space.

Rationale: *These applications must be ported to HOPS4 to maintain current functionality.*

P-25 It shall be possible to make 2D plots of the type currently possible within `aedit` of HOPS3.

Rationale: *These are plots such as SNR vs. time, broken out by baseline with separate symbols per source. This is needed to maintain current functionality.*

P-26 It shall be possible to make 3D visualization plots.

Rationale: *search makes contour plots, but a 3D visualization of amplitude with delay and delay-rate would be useful for exploring the parameter space near the fringe solution.*

P-27 HOPS4 shall implement interactive visualization tools in fourfit.

Rationale: *The HOPS3 version of fourfit provides a fixed fringe summary plot. HOPS4 should have an interactive plotting capability to enable zooming and viewing fourfit results with an expanded scale; This will be a significant improvement.*

P-28 The the existing fourfit control file format shall be replaced with a Python-based control file. The currently existing functionality shall be preserved.

Rationale: *The existing control file should continue to be supported, but all new and existing functionality will be interfaced with a Python based control file. New features will not likely be back-ported to HOPS3. Details of the control file design will be described in the Specifications document [2].*

2.5 Miscellaneous Requirements

A-29 It shall be possible to automatically discard correlator AP (integrations) with weights below a user defined threshold.

Rationale: *When data is missing the correlation products will be noisy, possibly to incorrect results. Flagging poorly recorded data segments is currently done in HOPS3 provided DiFX properly notices the loss and assigns a low weight to the AP bin. The threshold weight value below which data is discarded is user determined.*

A-30 HOPS4 shall have the ability to flag data based on a user-supplied list (*e.g.* of frequency intervals with time in some flag file).

comment: *This is needed especially when RFI or calibration tones signals are present. Details of the flagging methods will be described in the Specifications document [2].*

A-31 Every data type written to disk in HOPS4 shall be convertable to form that is amenable to human examination (e.g. ASCII or CSV), as is done in HOPS3 by the `CorAsc2` program.

Rationale: *This is an important feature for direct user inspection of the data in light of unexpected results.*

A-32 HOPS4 shall implement an algorithm for solving for station-based quantities from baseline-based quantities as a global fringe fitter.

Rationale: *This is required for converting sets of three (triangular) baselines to calculate the delays at individual stations. This is new for HOPS.*

A-33 The HOPS4 fourfit shall have the ability to apply complex bandpass correction.

Rationale: *This is a new capability that might well be prototyped in the existing HOPS3 package. Here "complex" refers to both amplitude and phase variation of the receiver frequency response. HOPS3 already has rather sophisticated phase calibration handling. The "manual" phase calibrations already handle the phase variation from between channels. HOPS3 has no amplitude adjustment capability. The desire is for a more flexible arrangment that is not limited by channel boundaries, and has the ability to correct either phase or amplitude or both. Details of the bandpass correction methods will be described in the Specifications & Design document [2].*

A-34 HOPS4 shall provide a method to solve for complex bandpass corrections.

Rationale: *This method should accept some set of scans and stations and solve, possibly using an LSF method, for per-station bandpass solutions. Supporting plots will be generated.*

A-35 Python wrappers for the new HOPS4 data objects shall be provided.

Rationale: *This is a core component of the user interface, and allows the user access to the underlying data for modificiation in custom scripting and access/export the data as needed outside of the HOPS4 toolset.*

D-36 The HOPS4 suite must provide mechanisms to preserve the correlator output data.

Rationale: *A PERL script, `hops_data_links.pl`, exists to manage symbolic links to analysis files in a working directory separate from the original correlator output directory.*

D-37 HOPS4 should preserve the the Single Band Delay, Multi Band Delay and Delay Rate search algorithm in its current form.

Rationale: *This algorithm must be ported to HOPS4 to preserve existing functionality. The desire is to make the algorithm modular, such that the user may (de)select search dimensions. For example the current loop in fourfit includes a loop over TEC, which is not needed at EHT frequencies and could be ignored in EHT analysis. Similarly, for spectral line data the multi-band delay is not meaningful. The revised implementation should make it easier to develop and introduce new algorithms..*

D-38 Benchmarking and performance analysis should be augmented with a more sophisticated tools.

Rationale: *This is to ensure HOPS4 is optimized as part of development best practices and at the least performs as well as HOPS3. The existing HOPS package supports a* `account` *library that does rudimentary profiling. Introducing* `gprof` *into some specific build tests, should be straightforward and help to optimize fringing.*

D-39 HOPS4 should support spectral-line VLBI.

Rationale: *This feature needs to be refactored and improved. The existing HOPS code assumes a continuum.*

D-40 HOPS4 should support pulsar folding with a user specified period or ephemeris file with blanking.

Rationale: *This would be a (new) desireable feature for users of the ngEHT interested in VLBI pulsar observations.*

D-41 HOPS4 should enable multi-threading or multi-processing for batch jobs.

Rationale: *This is required as part of an optimization effort and to reduce the compute resources required to run the HOPS4 programs. Parallelization is a strong desire, but the exact details of the implementation (particularly the requirements of threads vs processes) need to be carefully designed.*

3. References

[1] Lindy Blackburn et al. "EHT-HOPS pipeline for millimeter VLBI data reduction". In: *The Astrophysical Journal Letters* (2019).

[2] J Barrett et al. *ngEHT Specifications for a new HOPS*. Jan. 2021.

[3] J Barrett et al. *ngEHT Coverage and Testing for a new HOPS*. Jan. 2021.

[4] J Barrett et al. *ngEHT Development Plan for a new HOPS*. Jan. 2021.

[5] Adam T Deller et al. "DiFX: a software correlator for very long baseline interferometry using multiprocessor computing environments". In: *Publications of the Astronomical Society of the Pacific* 119.853 (2007), p. 318.

[6] Adam T Deller et al. "DiFX-2: a more flexible, efficient, robust, and powerful software correlator". In: *Publications of the Astronomical Society of the Pacific* 123.901 (2011), p. 275.

A Acronyms, Commands, and Glossary

A-list a one line description of baseline fringes used by HOPS

adump a program that dumps columns from a one line description of baseline fringes used by HOPS (A-list) scan data

alist a program for creating a file of A-list scan data

aedit a program for editing a file of A-list scan data

average a program that calculates averages on A-list scan data

AP Acquision Period which refers to a period of time over which the correlator integrates the input (noisy) data to produce a usable output. Terms such as dump or "integration" are also sometimes used, but both can be ambiguous.

channel an ambigous term which refers either to a spectral channel, *i.e.* frequency point of an FFT or to a sub-band of a larger receiver band.

cofit a HOPS tool to assess atmospheric coherence in terms of Signal to Noise Ratio (SNR) and short for "amplitude" the correlation coefficient (AMP) variation with integration interval

CorAsc2 Correlator to Ascii (2nd version)

DiFX the "distributed" a general term for correlation that does the cross-correlation after first transforming to frequency space (FX) correlator

EHT Event Horizon Telescope

ngEHT next-generation Event Horizon Telescope

fourfit the main fringe-finding command in HOPS

fringex an HOPS tool to explore the fringe

fourmer a program that combines data from two sub-bands into a larger common band

GNU/Linux a family of operating systems using Linus' kernel and GNU's software packages

HOPS Haystack Observatory Postprocessing System

LSF Least Squares Fit

Mk4 The fourth in a series of Very Long Baseline Interferometry (VLBI) hardware correlators. The Mark4 replaced the Mark3 near the beginning of the millenium, and was finally put to rest by the "distributed" FX correlator (DiFX) in the mid 2010's

MSRI Mid-scale Research Infrastructure

NSF National Science Foundation

ngEHT next-generation EHT

PGPLOT a “pretty good” plotting package developed and maintained by Tim Pearson at Caltech. He’s retired now, so it is stuck at verion 5.2.2, (released Feb 2001)

PERL Practical Extraction and Reporting Language created by Larry Wall

Python a programming language named in honor of Monty Python’s Flying Circus

RFI Radio Frequency Interference which is what you have when your receiver picks up signals you do not want

search this is a tool that searches in delay/delay-rate space to allow visualization of a fringe peak and to aid in establishing the validity of more marginal-SNR cases

SNR Signal to Noise Ratio

TEC Total Electron Content, refering to the column density of electrons in the line of sight through the ionosphere. Conventionally one TEC Unit is 10^{16} electrons / m^2

VEX VLBI EXperiment (file), a means of fully describing a planned VLBI experiment or observation