ehthops

Release 0.1

Lindy Blackburn, Chi-Kwan Chan, Iniyan Natarajan and others

Jul 05, 2023

CONTENTS:

CHAPTER

ONE

INTRODUCTION TO EHTHOPS

ehthops is a data calibration pipeline that uses the Haystack Observatory Preprocessing System (HOPS) and the EHT Analysis Toolkit (EAT) for calibrating millimetre-VLBI data, with a focus on the Event Horizon Telescope.

For details, refer to Blackburn et al. (2019).

This documentation explains the steps necessary for installing and running the ehthops pipeline, with some settings and steps tailored to 2021 data.

CHAPTER

TWO

INSTALLATION

The following instructions pertain to installing HOPS version 3.24 and the EHT-HOPS pipeline with python 3.10. This procedure has been tested on CentOS, Rocky OS, and Debian systems. If the software packages required to compile HOPS are installed in standard locations such as **/usr/local**, some of the following environment variables may not need to be defined.

2.1 Installing HOPS v3.24

Note: Build HOPS with the same Python version to be used by the EHT-HOPS pipeline (e.g. the mamba/conda environment used to run the pipeline). One way to ensure this is to set up the mamba/conda environment first (with at least future, numpy, scipy, matplotlib pre-installed), activate it, and then build HOPS.

Before installing HOPS, PGPLOT and FFTW must be installed. Some or all of the following packages may be necessary to be able to compile HOPS successfully. Note that the exact names might differ on different systems:

```
sudo apt install gcc make gfortran libx11-dev ghostscript libfftw3-dev parallel
sudo apt install gdb flex bison pkg-config autoconf automake gettext libtool
```

If FFTW3 is installed properly via apt, the following manual installation of FFTW3 may be safely skipped. Download PGPLOT and follow the instructions here to install it. Note that the switch from g77 to gfortran is necessary for any modern GNU/Linux system. Download FFTW and run the following commands:

```
./configure --prefix=</path/to/install/fftw> --enable-shared --enable-threads --enable-
↪openmp
make
make install
```

Define the following environment variables before compiling HOPS v3.24 so that PGPLOT and FFTW are discoverable during compilation:

```
export PGPLOT_DIR="</path/to/pgplot>"
export LD_LIBRARY_PATH="</path/to/pgplot>":"</path/to/fftw/lib>":$LD_LIBRARY_PATH
export LDFLAGS="-L</path/to/fftw/lib>"
export CFLAGS="-I</path/to/fftw/include>"
export CPPFLAGS="-I</path/to/fftw/include>"
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:"</path/to/fftw/lib/pkgconfig>"
```

On some systems the following flags may also have to be set manually to be able to configure HOPS-3.24:

```
export FFTW3_LIBS="</path/to/fftw/lib>"
export FFTW3_CFLAGS="</path/to/fftw/include>"
```

Download HOPS via anonymous ftp:

```
wget -r ftp://gemini.haystack.mit.edu/pub/hops
```

The above command checks out all past versions of HOPS under *gemini.haystack.mit.edu/pub/hops*. Untar the latest version of HOPS (as on June 2023, version 3.24). At the same location where hops-3.24 is untarred, create a build directory in which to compile HOPS:

```
mkdir bld-3.24
cd bld-3.24
../hops-3.24/configure --prefix=</path/to/install/hops-3.24> --enable-devel
make all
make install
```

Note: Without the **–enable-devel** flag, many necessary HOPS postprocessing executables will not be built.

To set up the HOPS environment, run the following command:

```
source </path/to/hops-3.24/bin/hops.bash>
```

2.2 Setting up a new python environment

We will use Mamba for creating Conda environments for running the pipeline. We will use python version 3.10 for the setup:

```
conda create -n eht310 python=3.10
```

Note: Note that python 3.8 can run the pipeline but cannot generate the summary notebooks using either nbconvert or papermill. Python 3.10 is recommended.

Install the following packages within the new environment:

```
conda install -c conda-forge astropy scipy matplotlib pandas seaborn scikit-learn future␣
↪pytables ipykernel papermill nbconvert jupyter
```

Note: If **seaborn** doesn't pull in **statsmodels**, then **statsmodels** should also be added to the above command.

Note: Verify that numpy version is <=1.23. If not, downgrade it to 1.23 via conda. This is necessary because some ehtim functionality do not work properly with >=1.24.

Install the EHT Analysis Toolkit (EAT) in developer mode:

```
git clone https://github.com/sao-eht/eat.git
pip install -e eat
```

eht-imaging is needed for post-processing. Check out the *dev* branch of **eht-imaging** and install locally with **pip**:

```
git clone https://github.com/achael/eht-imaging.git
cd eht-imaging
git checkout dev
pip install .
```

Note: Verify that libstdcxx-ng version is >=12. This is necessary to avoid potential *glibcxx not found* errors during execution.

Some systems may not have GNU parallel installed by default. If this is the case, install parallel from source and add it to the system path:

```
export PATH=$PATH:"/path/to/parallel/bin"
```

CHAPTER

THREE

RUNNING EHTHOPS

The ehthops pipeline consists of multiple stages, each consisting of multiple steps. A detailed description of the working of the pipeline can be found in Blackburn et al. (2019).

3.1 Directory structure

Stage 0 in band 1 (**hops-b1/0.bootstrap**) contains all the scripts that are common to all steps in each stage in each band. Step-specific scripts are hosted in the corresponding directory under **hops-b1**. All other bands symlink to these scripts so that only one physical copy of this set of scripts is necessary.

Stages 0 and 1 in band 1 (**hops-b1/0.bootstrap** and **hops-b1/1.+flags+wins**) host the preset control files and flags. Other bands symlink to these control files where necessary, aside from hosting band-specific control files and flags at the corersponding stages.

3.2 Driver scripts

The *dev-template/scripts* directory contains driver scripts to run the pipeline in two different enivironments.

driver_cannon.sh and *hops2021.slurm* are sample scripts that can be modified to run on any SLURM cluster (e.g. the Harvard FAS cluster).

driver_cloud.sh is a sample script tailored to run on eht-cloud. The environment setup lines are hidden inside another script hosted on eht-cloud.

The driver scripts must be run from the *dev-template/hops-bx* directories. They set the following environment variables that **must be** verified before each execution:

```
SET_SRCDIR -- sets the parent directory containing the various revisions of the data
SET_CORRDAT -- sets the revisions to be processed as a colon-separated list of directory␣
↪names
SET_EHTIMPATH -- sets the path to the source code of eht-imaging
```

Note: At all stages from 0 to 5, SRCDIR points to the directory that hosts the archival data. At stage 6, SRCDIR must point to the directory '5.+close/data' in the current band.

cleanup.sh deletes all data generated as a result of a previous run and leaves the repo at the default state.

The pipeline can be executed by typing the following in a linux terminal or a screen session (or in the case of a SLURM cluster, placing this line in the script submitted to SLURM):

```
source <script-name>
```

On a SLURM cluster, the above line is placed inside *hops2021.slurm* and the SLURM job can be submitted by:

```
sbatch hops2021.slurm
```

Note: Instructions to run as a Docker image to be added.

3.3 EHT2021 data specific instructions

HOPS operates by assigning single letter codes to frequency channels, restricting the number of channels that can be represented. Since the EHT 2021 campaign observed at two different frequency bands (230 GHz and 345 GHz), it is better to keep the reduction clean, by processing these data separately.

The 2021 campaign observed at 345 GHz only on day 109 (expt_no 3769, track e21f19). Hence the preset flags corresponding to track e21f19 correspond only to 345 GHz observations. While reducing 230 GHz data, replace the contents of *hops-b1/1.+flags_wins/cf1_flags_e21f19* with the following:

```
* Flag all 345 GHz scans
if scan > 109-000000
  skip true  * 109 is the day of the 345 GHz obs in 2021
```

hops-b1/1.+flags_wins/cf1_flags_e21f19 is the only control file pertinent to 345 GHz data reduction. Hence, the other control files containing the flags (prefixed *cf1_flags_*) can safely be deleted from the working copy of the repo if only 345 GHz data are being processed.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search