Science Operations Center Supporting Package Model

J. Marcus Hughes, Sarah Kovac, Chris Lowder, Ritesh Patel, Daniel Seaton, Matthew West

All affiliated with Southwest Research Institute

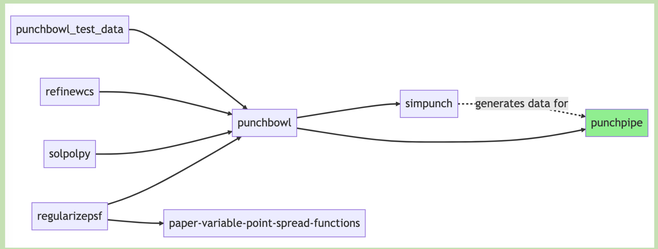
Overview

A primary objective of the PUNCH Science Operations Center (SOC) software design is to achieve sufficient flexibility and modularity that processing tools benefit projects outside of PUNCH and are easy to use outside our data pipeline. Instead of having a large monolithic package, we have a core package called *PUNCHBowl* that leverages standalone, special-purpose packages for more general tasks. For example, *PUNCHBowl* uses our supporting *regularizePSF* package to perform point spread function corrections. This allows the *regularizePSF* code to be used more easily in other applications, including those outside of solar physics. Here we highlight our software design and feature current supporting packages in the yellow boxes.

Design

Each yellow box shows a SOC supporting package. These are released as standalone Python packages for general use. They are automatically installed when using the core *PUNCHBowl* pipeline. *PUNCHBowl* is designed to not include the automation infrastructure required by the SOC.

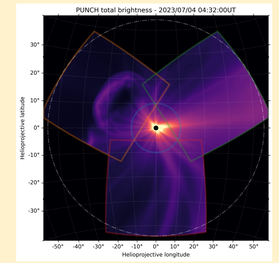
Forthcoming packages will be available on the PUNCH GitHub at this QR code.



Above: *PUNCHBowl* imports from the packages with incoming arrows. *punchpipe* is shown in green because it is not planned for public release. It only contains automation infrastructure needed by the SOC.

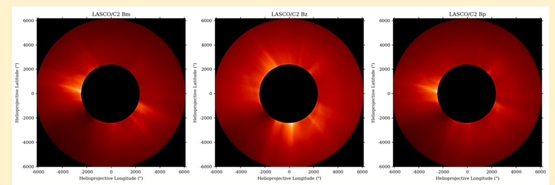
simPUNCH

simPUNCH is a forthcoming package that will create synthetic PUNCH observations for testing. At right is a preview based on the GAMERA model.



solpolpy

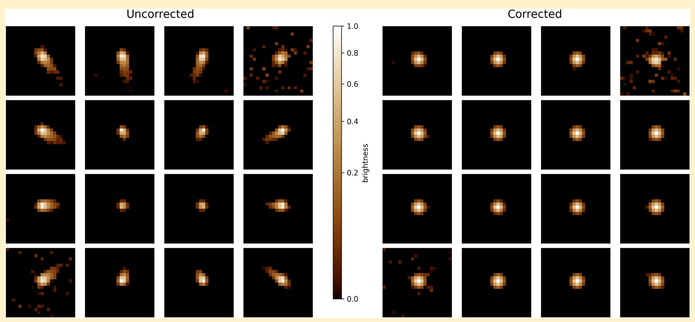
See "*PUNCH Polarization Resolver and the IMAX Effect*" poster



Above: solpolpy supports many polarization systems in DeForest et al. (2022).

regularizePSF

regularizePSF supports modeling point-spread functions (PSFs) both functionally and purely empirically, homogenizing PSFs to be uniform, synthesizing observations with a given PSF, and visualizing PSF models and results. It is publicly available on GitHub with an associated published paper.

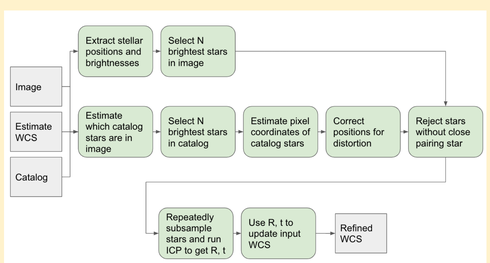


Paper Repo

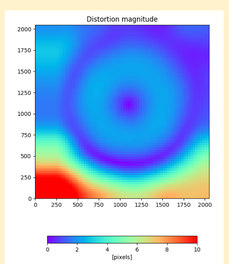
Above: Average stars in PUNCH WFI engineering model image regions before and after PSF homogenization, left and right respectively. Reproduced from Figure 4 of Hughes et al. (2023).

refineWCS

PUNCH requires very precise, sub-pixel pointing knowledge to carry out calibrations required to produce level 3 products. The PUNCH SOC has developed a pointing refinement algorithm that will be released in a forthcoming package called *refineWCS*. It takes an initial pointing estimate as a world coordinate system (WCS) and mathematically uses the stars visible in an image to determine a better pointing model. *refineWCS* is also used in calibration to build the optical distortion model.



Left: The *refineWCS* pointing refinement algorithm is based on existing computer vision techniques and follows these steps.



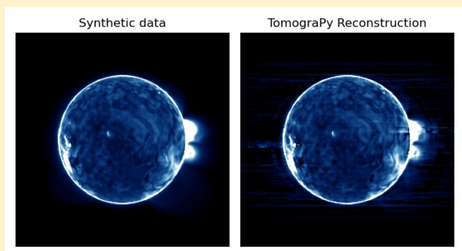
Right: Modeled optical distortion in WFI engineering data

TomograPy

While not a primary PUNCH SOC software, we have been modernizing the *TomograPy* package written by Barbey et al. (2011). *TomograPy* is a solar tomography package to reconstruct the three-dimensional solar environment from observations.

The package is not yet completely modernized from Python 2. We have chosen to develop the core functionality in Rust with a Python interface. We support modern helio data through *NDCube*.

Paper Repo



Above: *TomograPy* begins with many observations like shown on the left. Using tomographic techniques, a 3D data cube of the environment is modeled. This can then be validated by projecting back to observation space as shown on the right.