Data Reduction with Jupyterhub

Steve Crawford

9 October 2019
James Webb Space Telescope

JWST is 6.5m space telescope expected to launch in March 2021.

Suite of near- and mid-infrared imaging and spectroscopic instruments:

- FGS
- MIRI
- NIRCAM
- NIRISS
- NIRSPEC

A fully assembled JWST (August 2019)
JWST Calibration software: A single package to reduce them all

P10.19 H. Bushouse The JWST Science Calibration Pipeline

https://github.com/spacetelescope/jwst
The JWST calibration software testing includes:

- **Verification through Integration and Testing with Science Operations Center software**
- **Validation by Instrument team Scientists**

Along with the automated tests, Astronomers are writing Jupyter notebooks for manual testing of the calibration software.

P9.16 R. Diaz End-to-end validation framework
Motivation

How do we provide a reproducible environment for testing?

How can we provide an environment that can be seamlessly updated?

How will we enable investigators to quickly extract the scientific results from complex observations?
Jupyterhub provides a browser-based science platform.

- Running on AWS provides flexibility and scalability
- Used previously for TESS and WFIRST workshops
Components in Jupyterhub

CRDS ingests, stores, associates, and serves the references files for JWST. A cloud based cache has been enabled to allow users on the Jupyterhub system to directly read from S3 storage.
Access to Data

Astroquery plus MAST authentication

```python
>>> from astroquery.mast import Observations

>>> obs_table = Observations.query_object("M8", radius="0.02 deg")
>>> data_products_by_obs = Observations.get_product_list(obs_table[0:2])
>>> print(data_products_by_obs)
```

Data Redirector

```python
base_url = 'https://data.science.stsci.edu/redirect/JWST/jwst-data_analysis_tools/miri_simulated_data/
association_filename = "det_dithered_5stars.json"
urllibretrieve(base_url + association_filename, association_filename)
print("Downloaded: {}".format(association_filename))
```
Demo of the Environment
Challenges in the Jupyterhub Environment

• Security
  • How do we secure the environment from external and internal sources?
  • How do we control exclusive access data in the environment?

• Collaboration
  • How do we provide an environment that enables collaboration while being secure?

• Observability
  • What are the important metrics to monitor?
  • How do we have real time insight into the system?

• Cost
  • How do we control costs for different users and different use cases?
  • How do we provide access for different users?

• Adoption
  • As a new tool, how do we introduce it to the community?
Future Opportunities

• Authenticated Cloud service for direct access to data
• Providing the full suite of JWST Scientific tools in a single, browser based user interface
  • P11.8 J Taylor: JWST Data Simulations and How to Use Them
  • P10.5 M. Bourque: The James Webb Space Telescope Quicklook Application (JWQL)
• Integrating with addition astronomer and data science software to enable a range of scientific use cases
• Jupyter-widget Glueviz based data analysis tools
• Integrate with batch processing software for job control and monitoring
• Enable cloud based reprocessing and analysis for HST public data set on AWS S3
• The data size of WFIRST will require *bringing users to the data* and this is an early prototype of the High Level Processing Partition
Thank you

Jacob Matuskey Mary Romelfanger Andrew Cortese Brian Hayden Todd Miller Edward Slavich Christine Slocum Michael Gough Pey-Lian Lim Joe Hunkeler Erik Tollerud Michael Fox Clara Brasseur Jonathan Hargis Yuvi Panda

Howard Bushouse Jonathan Eisenhamer Nadia Dencheva James Davies David Grumm Philip Hodge Jane Morrison Megan Sosey Robert Jedrzejewski David Davis

Alicia Canipe Julien Girard Bryan Misty Cracraft Rosa Diaz Anton Koekemoer Karl Gordon Sarah Kendrew Mees Fix

Arfon Smith Ivelina Momcheva Joshua Peek

Software available at: https://github.com/spacetelescope

Interested in joining the team?
http://www.stsci.edu/opportunities
Demo of the environment
Demo of the environment
Demo of the environment
Demo of the environment

Running Pipelines using MIRI image data from MAST

Table of Contents:
- Detector1 Pipeline
- Resources and Documentation
- Download Data from MAST
- Run Pipeline with Default Configuration
- Run Pipeline with Default Configuration Files
- Run Pipeline with Parameters Set Programmatically
- Run Individual Steps with Configuration Files
- Run Level 1, 2, 3, Pipelines in Succession

Detector1 Pipeline
Stage 1 consists of detector-level corrections that are performed on a group-by-group basis, followed by ramp fitting.

More information can be found at: https://jwst-pipeline.readthedocs.io/en/latest/jwst/pipeline/calwebb_detector1.html#calwebb-detector1

Inputs: The inputs to stage 1 processing will usually be level-1b raw files.
Outputs: The output of stage 1 processing is a countrate image per exposure, or per integration for some modes.

Level 1 pipeline:
- Calwebb Detector1 (jwst.pipeline, calwebb_detector1, Detector1Pipeline) (calwebb_detector1.cfg)

Level 1 pipeline steps:
- Group Scale (jwst.group_scale, group_scale_step, GroupScaleStep) (group_scale.cfg)
Demo of the environment

Image Registration and Combination using the JWST Level 3 Pipeline - MIRI example

Stage 3 Image processing intends for combining the calibrated data from multiple exposures (e.g., a dither or mosaic pattern) into a single distortion corrected product. Before being combined, the exposures receive additional corrections for the purpose of astrometric alignment, background matching, and outlier rejection.

**Inputs:** The inputs to calwebb_image3 will usually be in the form of an association (ASN) file that lists multiple associated 2D calibrated exposures to be processed and combined into a single product. The individual exposures should be calibrated ("cal") from calwebb_image2 processing. It is also possible to use a single "cal" file as input, in which case only the resample and source_catalog steps will be applied.

**Outputs:** A resampled/rectified 2D Image product with suffix "12D" is created, containing the rectified single exposure or the rectified and combined association of exposures (the direct output of the resample step). A source catalog produced from the "12D" product is saved as an ASCII file in "ecsv" format, with a suffix of "cat". If the outlier_detection step is applied, a new version of each input calibrated exposure product is created, which contains a .DD array that has been updated to flag pixels detected as outliers. This updated product is known as a CR-flagged product and the file is identified by including the association candidate ID in the original input "cal" file name and changing the suffix to "cr".

Level 3 pipeline steps:

- **Tweakreg** (jwst.tweakreg, tweakreg_step, TweakRegStep)
- **Sky Match** (jwst.skymatch, skymatch_step, SkyMatchStep)
- **Outlier Detection** (jwst.outlier_detection, outlier_detection_step, OutlierDetectionStep)
- **Resample** (jwst.resample, resample_step, ResampleStep)
- **Source Catalog** (jwst.source_catalog, source_catalog_step, SourceCatalogStep)

(for more information on individual steps see: [https://jwst-pipeline.readthedocs.io/en/latest/jwst/package_index.html](https://jwst-pipeline.readthedocs.io/en/latest/jwst/package_index.html))