



# THE HIGH-END COMPUTING CAPABILITY AND ZIGGY: A FLEXIBLE, SCALABLE SCIENCE PIPELINE MANAGER

JON M JENKINS

ADVANCED SUPERCOMPUTING DIVISION

NASA AMES RESEARCH CENTER

NASA Ames Research Center  
Habitable Worlds Observatory  
Meeting #7  
29 April 2024

H A B I T A B L E  
W O R L D S  
O B S E R V A T O R Y



# HECC OVERVIEW



## NASA Ames' High-End Computing Capability (FY21 Budget: \$47M)

- NASA's only cloud-scale private HPC cloud infrastructure
- Similar economics of scale to CSPs who run multiple hyperscale data center
- Robust power infrastructure from aero/wind tunnel heritage (each >100MW)
- Current Pad/Power infrastructure can scale out to ~1 Billion SBUs (7.5x)
  - straight forward to double again
- > **663,000 CPU cores** > **2.2x10<sup>6</sup> GPU cores**
- > 17,000 compute nodes - (delivered 110 million SBUs FY21)
- > **116 PB of on-line data storage**
- > 1400 PB of off-line tape data
- **100-gigabit backbone supporting 1G, 10G and 25G hosts**
- Supplementary analysis systems
- Scientific Consulting for Optimization and Help (significant code speed up)
- Data Analysis and Visualization – as a service



## Traditional focus on modeling and simulation (Data Producer)

- **Evolving Support for Hybrid Computing** (On-prem/Cloud)
- Improvements around Latency (reservations and dedicated systems)

## Growth in the size and nature of SMD data sets and SPD-41 motivating changes in HECC

- ECCO (Ocean Only Model Outputs) – 4 PB data set accessible through the NAS Data Portal
- GEOS Coupled – 3 PB of scratch space ([https://gmao.gsfc.nasa.gov/GEOS\\_systems/](https://gmao.gsfc.nasa.gov/GEOS_systems/))
- ECCO with GEOS5 – increased simulation output ~10 – 20 PB
  - Ocean Biology (e.g., predict whale migration/feeding patterns based on food sources estimated from global models)
- NASA's Earth Exchange (NEX) - 5.9 PB of on-line storage, used for data cache based on projects requirements for a given funding cycle.
- SBG is expected to collect 2.4 TB day<sup>-1</sup> and produce 40 TB day<sup>-1</sup> of data products



# Problem Statement

- Science data processing software is typically custom-built for every science mission
  - This incurs technical, cost and schedule risk that is often realized
- Science data processing software is frequently assigned lower priority than other mission elements, which introduces risk to the mission science
  - For cost-constrained missions this typically falls on a postdoc who shows an aptitude and/or interest in software
  - This often results in a brittle workflow, not a robust software project
- This is an identified weakness in the Earth Venture-class missions



# Designing, Developing and Deploying New Science Data Systems is Expensive



Kepler Mission: SPIfY 1.0

Time to develop SDS:  
78 person years, or ~\$25M

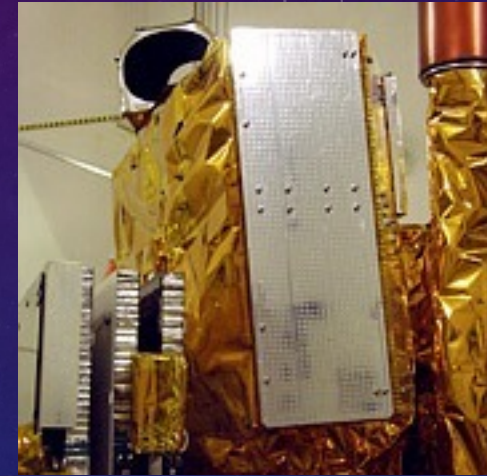
◀ Data Rate: 1.1 GB day<sup>-1</sup>  
Data Volume: 1.6 TB



TESS Mission SPIfY 2.0

Time to develop SDS:  
~31 person years or ~\$10M

Data Rate: 13-130 GB day<sup>-1</sup>  
Data Volume to Date: 156 TB



Hyperion/EO-1: Ziggy 1.0

Time to develop SDS:  
~3 person years or ~\$1M

Data Rate: ~8.8 GB day<sup>-1</sup>  
Data Volume: 56 TB



# Designing, Developing and Deploying New Science Data Systems is Expensive



Kepler Mission: SPIfY 1.0

Time to develop SDS:  
78 person years, or ~\$25M

Data Rate: 1.1 GB day<sup>-1</sup>  
Data Volume: 1.6 TB



TESS Mission SPIfY 2.0

Time to develop SDS:  
~31 person years or ~\$10M

Data Rate: 13-130 GB day<sup>-1</sup>  
Data Volume to Date: 156 TB



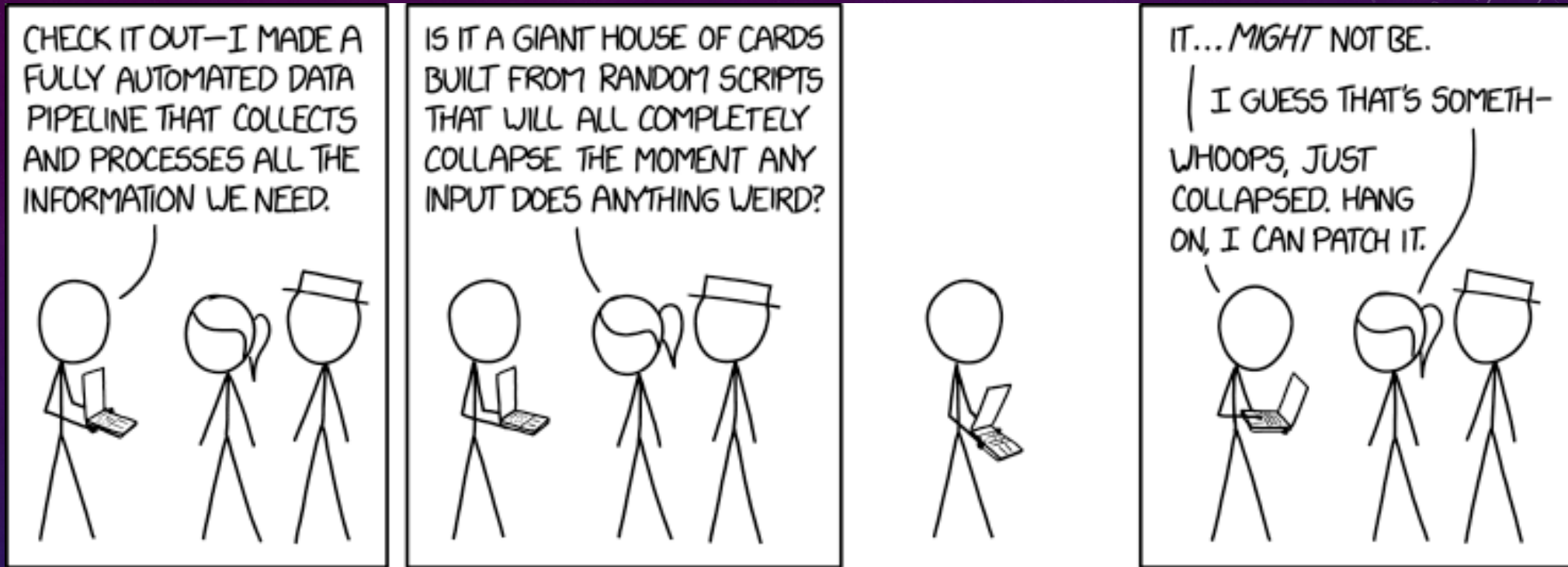
Hyperion/EO-1: Ziggy 1.0

Time to develop SDS:  
~3 person years or ~\$1M

Data Rate: ~8.8 GB day<sup>-1</sup>  
Data Volume: 56 TB

**Ziggy scales easily to data rates of 5+ TB day<sup>-1</sup>**

xkcd.com. Used under terms of Creative Commons License 2.5.



## Science data pipelines need to do a lot more than science:

- Logging
- Execution flow
- Execution monitoring
- Data accountability
- Configuration management
- Error handling
- And much more!



**ziggy** handles all the not-science and lets the scientists get on with the science!

Develop here\* ...



... deploy here!



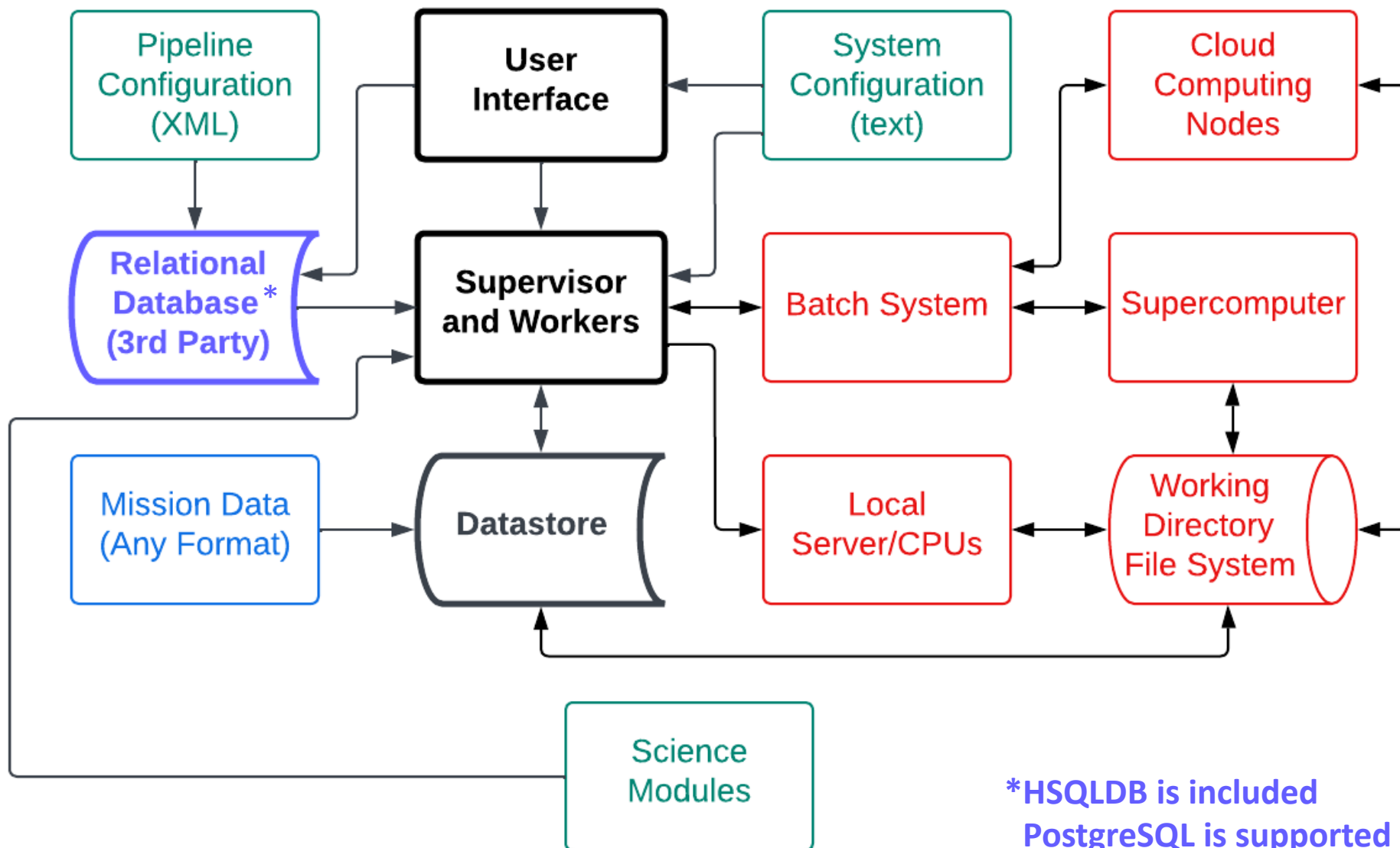
**Ziggy is Lightweight**

\*Dog not included.



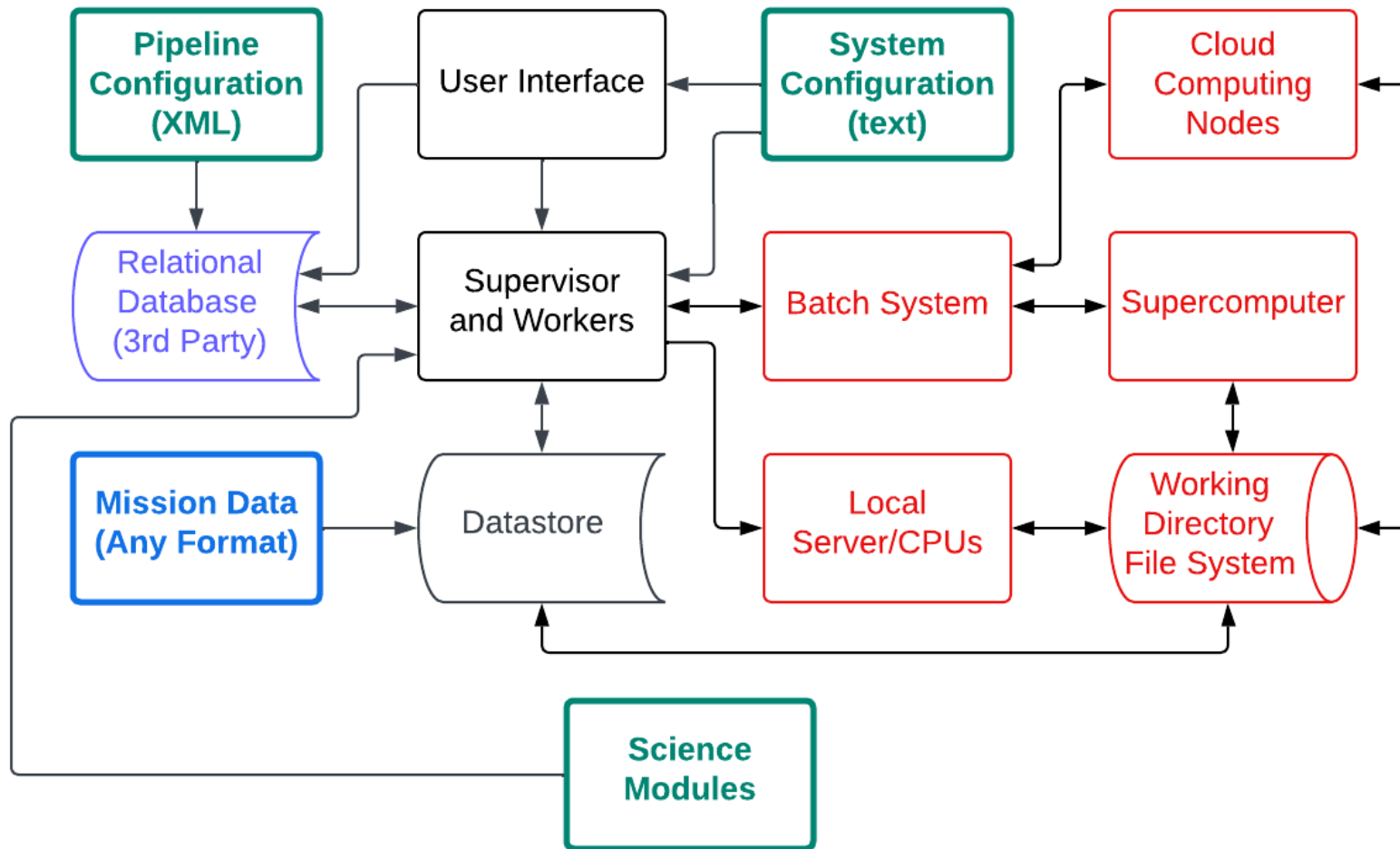


# Ziggy-Supplied Components





# User-Supplied Components



**Subject Matter Experts supply**

- **Algorithms**
- **Pipeline and system configuration and**
- **Mission data**

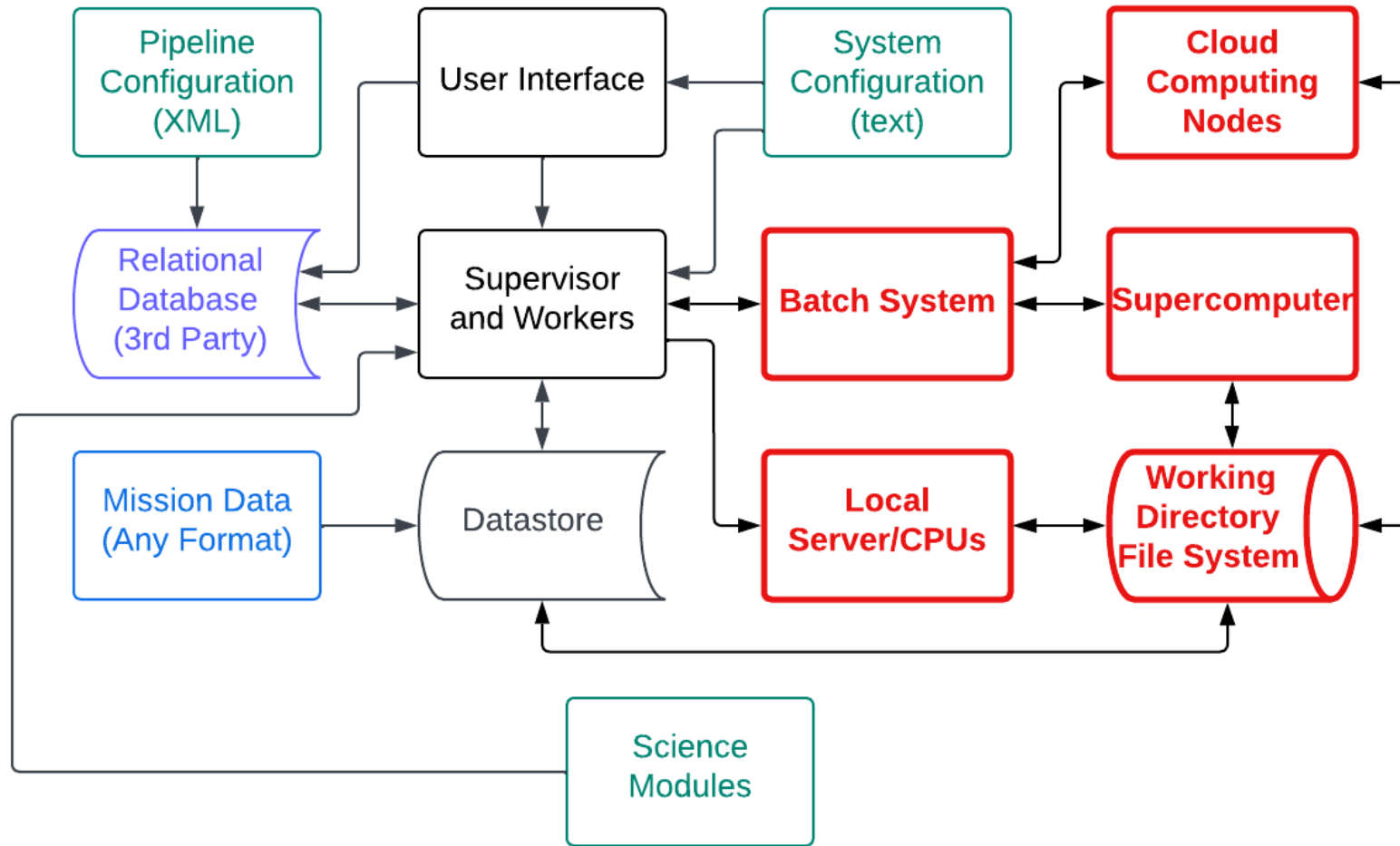
**Algorithms “As-is” – no design rules that must be followed**

**Polyglot – “mix and match” any languages**

**So far we’ve used MATLAB, C++, Python, Shell Script – it all just works**



# Execution Options



Select execution in server or in HPC / Cloud on a per-algorithm basis

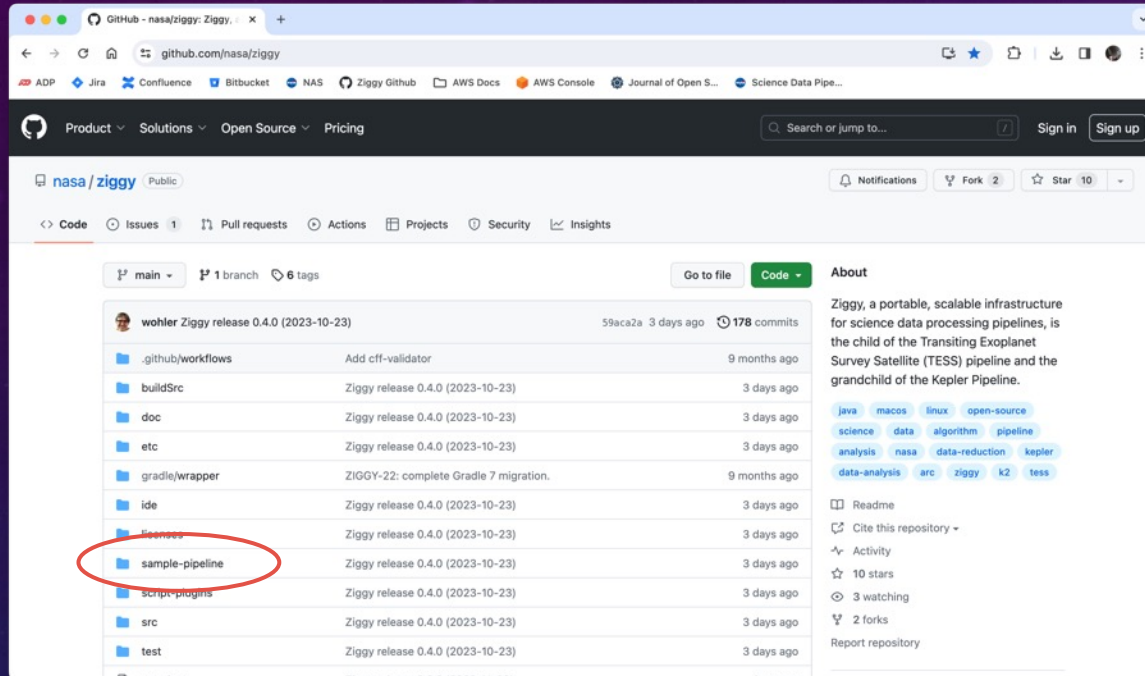
Current support for NAS (PBS Batch)

Coming soon: support for AWS / Slurm batch



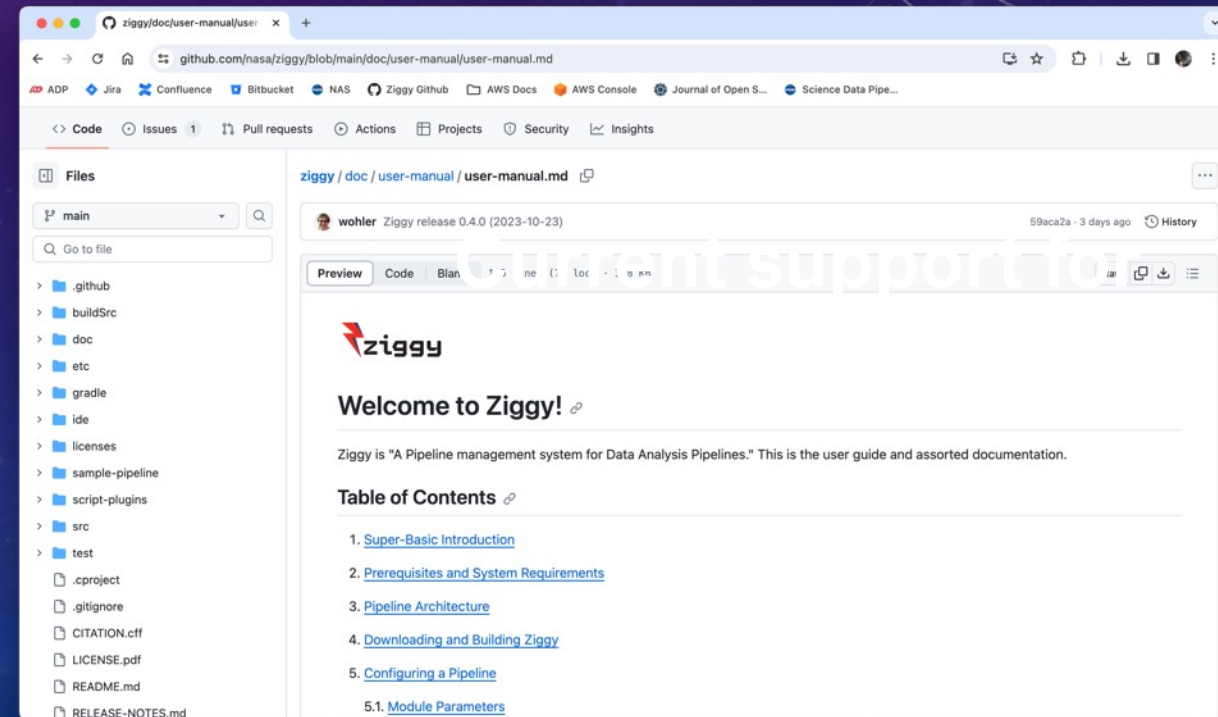
# Ziggy is Open Source!

[github.com/nasa/ziggy](https://github.com/nasa/ziggy)



Super-basic sample pipeline that shows you how to do it all

Extensive easy-to-read documentation





# Kepler Science Operations Center Architecture



## Ziggy's Predecessor

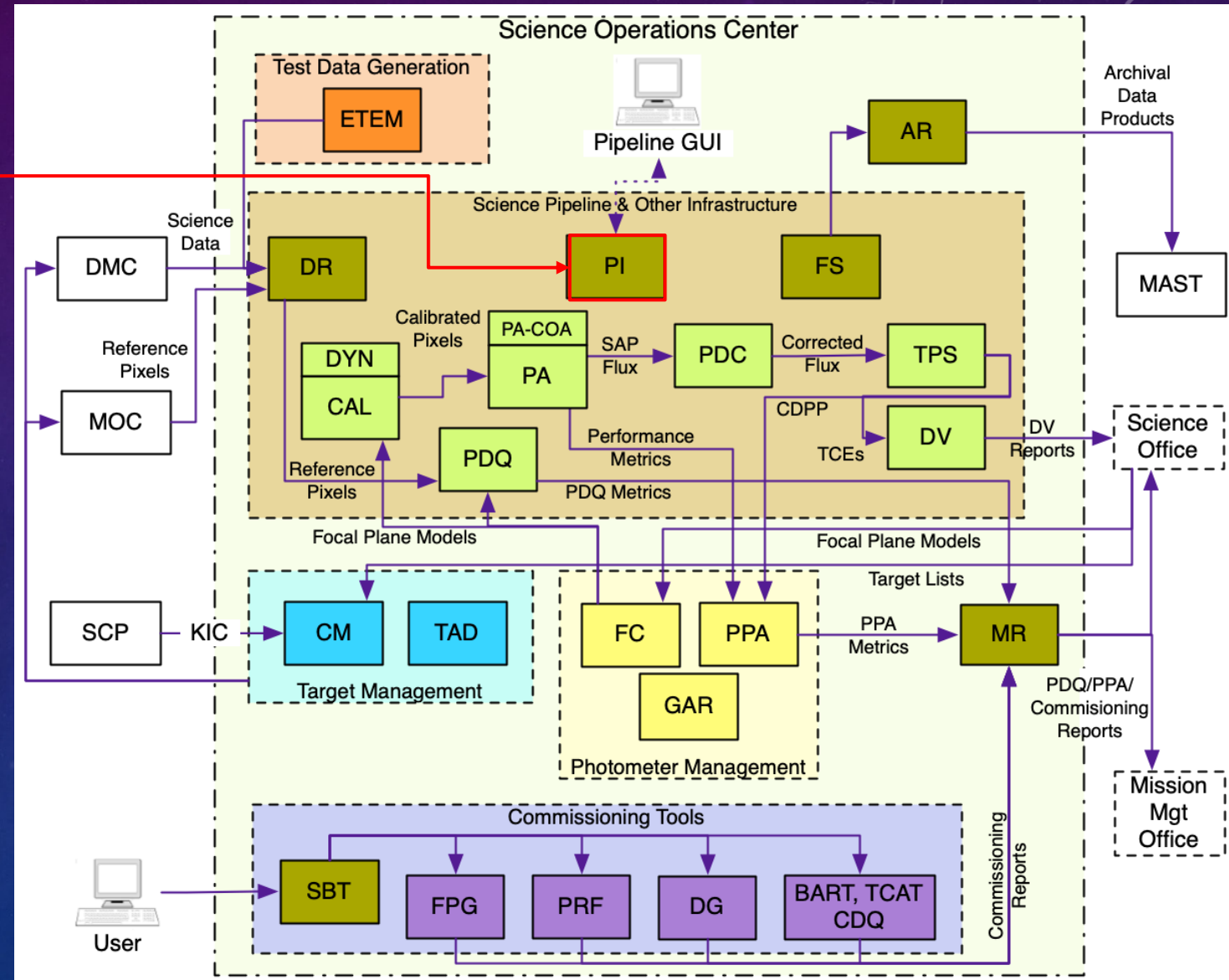
The Kepler SOC Pipeline consisted of 24 Computer Software Configuration Items, including the Pipeline Infrastructure (PI), which has been split off as its own software project, *Ziggy*

*Ziggy* is capable of running complex science algorithms on large data sets on a laptop, workstation or on the NAS Pleiades supercomputer.

We've demonstrated running *Ziggy* in AWS

TESS SOC directly connected to NAS Pleiades backbone network

Performant Access to Compute and Filesystems



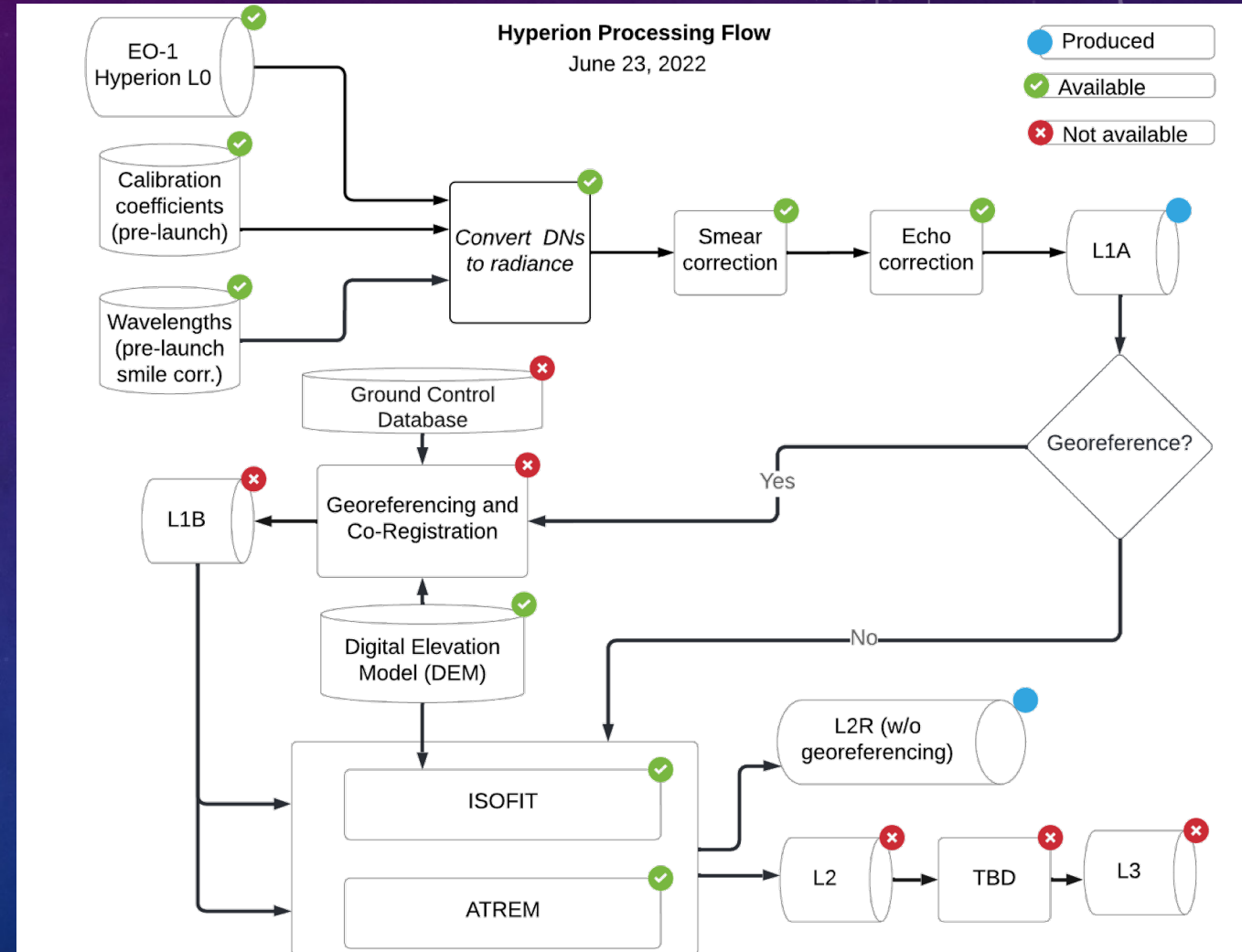


# SURFACE BIOLOGY & GEOLOGY PATHFINDER STUDIES



## Hyperion Pipeline:

- We've resurrected the Hyperion pipeline and reprocessed nearly the entire data set to L2 (surface reflectance)
- Currently checking consistency of Hyperion surface reflectance results (L2) (RadCalNet and AVIRIS/Hyperion Comparisons)
- Future work:
  - Incorporate ground control database and georeferencing and co-registration





# Conclusions

HECC represents a critical data processing resource for NASA's Science Mission Directorate

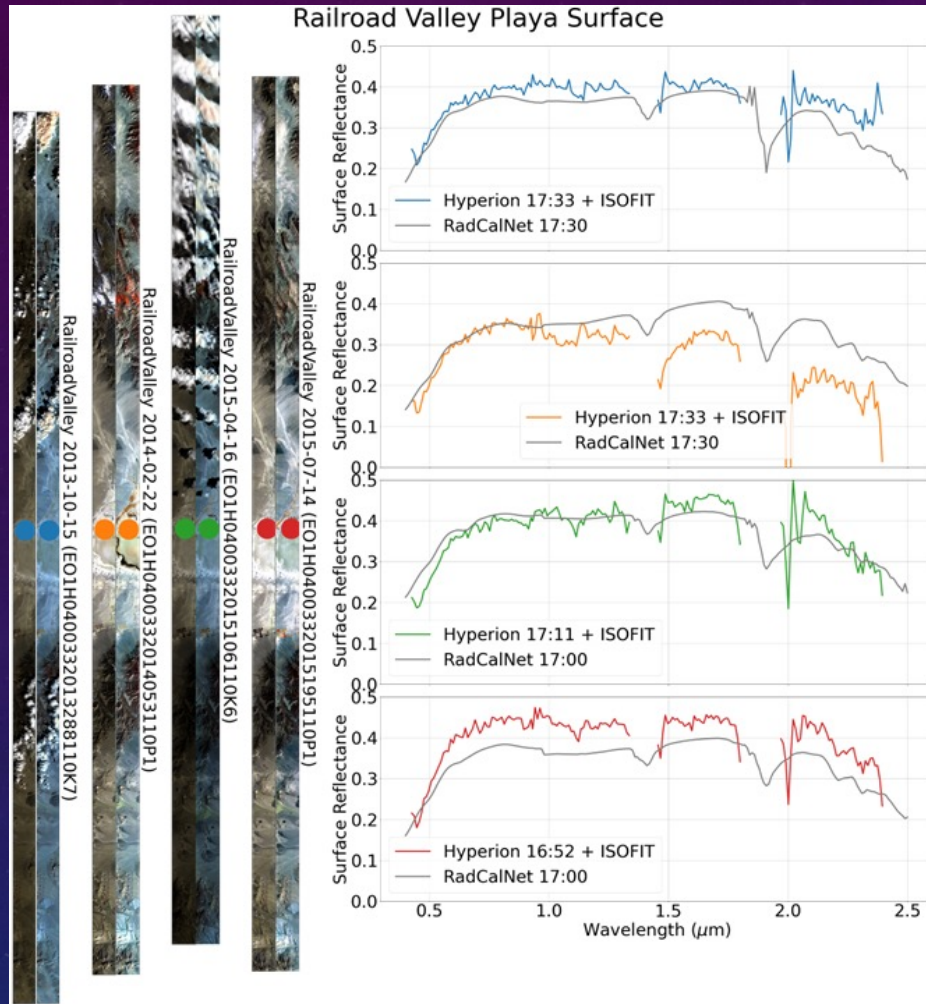
Ziggy solves a number of problems:

- Supports Open-Source Science & SPD-41a: released by NASA to [github.com/nasa/Ziggy](https://github.com/nasa/Ziggy)
- Easy to use and ready to use: new pipelines can be set up in days
- Ensures replicability and repeatability of science results: stores pipeline and algorithm configurations in a relational database
- Allows for highly-automated, event-driven operations
- Reduces cost and risk of developing and operating complex science pipelines
- Ziggy is TRL-7, NPR-7150-compliant as a class C software project

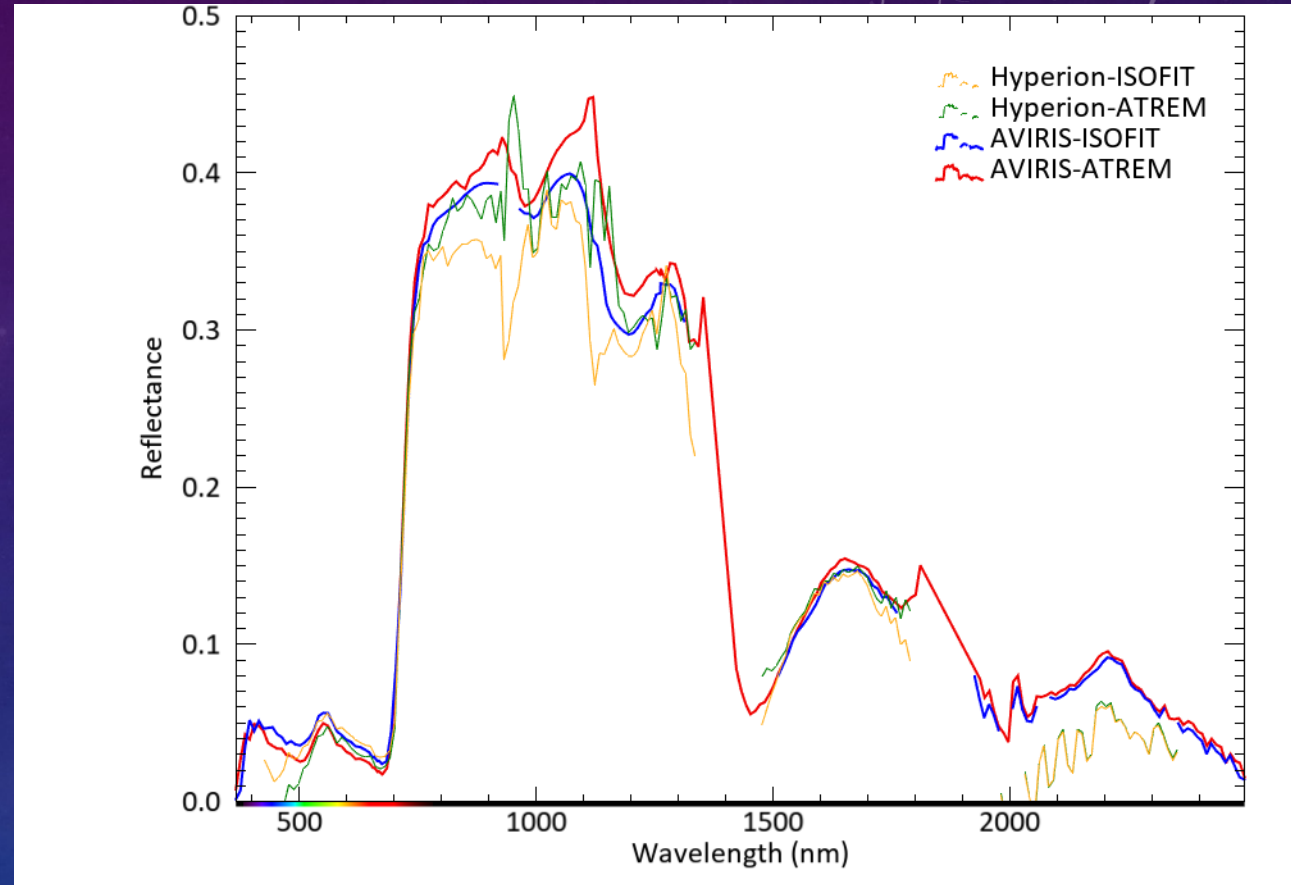
Backup



# SURFACE BIOLOGY & GEOLOGY PATHFINDER STUDIES



Comparison of RadCalNet measurements with Hyperion surface reflectance retrievals for scenes observed in Railroad Valley.



Comparison of surface reflectance spectra retrieved with the Hyperion and AVIRIS sensors, each with ISOFIT and ATREM retrieval algorithms. The results are shown for a vegetation site near Half Moon Bay, CA, observed on April 30, 2015.



File

Help

NASA

powered by

ziggy

Pi

W

Pr

A

Parameter Library

Pipelines

**Instances**

Status

Data Receipt

Event Definitions

Module Library

Pipeline instances

Clear filters

Show instances

Started within the last 

10

 days

1d

7d

14d

30d

90d

Limit states

☐ Initialized

☐ Errors (running)

☐ Processing

☐ Errors (stalled)

☐ Completed

Instance name

Search

| ID | Pipeline name         | Date            | State          | P-time   |
|----|-----------------------|-----------------|----------------|----------|
| 1  | sample: data-recei... | 2023-11-06 0... | COMPLETED      | 00:02:08 |
| 2  | sample:               | 2023-11-06 0... | COMPLETED      | 00:01:57 |
| 3  | sample: data-recei... | 2023-11-06 0... | ERRORS_STALLED | 00:20:11 |

Pipeline tasks

| Module       | Submitted | Processing | Completed | Failed | Subtasks    |
|--------------|-----------|------------|-----------|--------|-------------|
| data-receipt | 0         | 0          | 2         | 0      | 0 / 0 / 0   |
| permuter     | 0         | 0          | 6         | 0      | 24 / 24 / 0 |
| flip         | 0         | 0          | 0         | 6      | 8 / 0 / 0   |
|              | 0         | 0          | 8         | 6      | 32 / 24 / 0 |

| ID | Module       | UOW      | State     | Worker      | P-time   | P-state        |
|----|--------------|----------|-----------|-------------|----------|----------------|
| 12 | data-receipt | sample-1 | COMPLETED | localhost:1 | 00:00:00 | C (0 / 0 / 0)  |
| 13 | data-receipt | sample-2 | COMPLETED | localhost:2 | 00:00:00 | C (0 / 0 / 0)  |
| 14 | permuter     | 1        | COMPLETED | localhost:2 | 00:00:35 | C (4 / 4 / 0)  |
| 15 | permuter     | 2        | COMPLETED | localhost:1 | 00:00:36 | C (4 / 4 / 0)  |
| 16 | permuter     | 3        | COMPLETED | localhost:2 | 00:00:35 | C (4 / 4 / 0)  |
| 17 | permuter     | 4        | COMPLETED | localhost:1 | 00:00:35 | C (4 / 4 / 0)  |
| 18 | permuter     | 5        | COMPLETED | localhost:2 | 00:00:35 | C (4 / 4 / 0)  |
| 19 | permuter     | 6        | COMPLETED | localhost:1 | 00:00:35 | C (4 / 4 / 0)  |
| 20 | flip         | 1        | ERROR     | localhost:1 | 00:00:03 | Ae (4 / 0 / 0) |
| 21 | flip         | 2        | ERROR     | localhost:2 | 00:00:03 | Ae (4 / 0 / 0) |
| 22 | flip         | 3        | ERROR     | -           | -        | I (0 / 0 / 0)  |
| 23 | flip         | 4        | ERROR     | -           | -        | I (0 / 0 / 0)  |
| 24 | flip         | 5        | ERROR     | -           | -        | I (0 / 0 / 0)  |
| 25 | flip         | 6        | ERROR     | -           | -        | I (0 / 0 / 0)  |