Fast, reproducible, and extensible:

new software tools for planetary science data access and analysis

Chase Million, Michael St. Clair, Sierra V. Kaufman

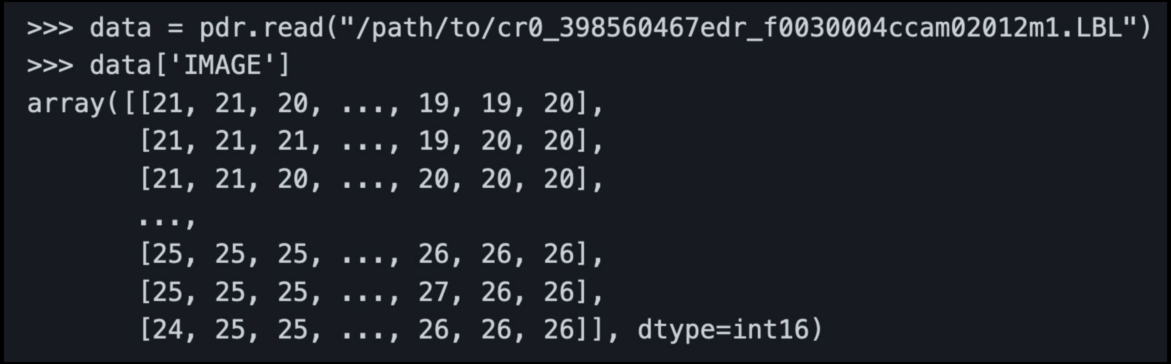
10⁶C Million Concepts

pdr

https://github.com/MillionConcepts/pdr

The [P]lanetary [D]ata [R]eader provides a single Python function — pdr.read([filename]) — to read planetary observational data into convenient Python data structures.

Associated data and metadata are available in standard Python data formats as attributes of the resulting data object, and can also be accessed using `dict`-like syntax.

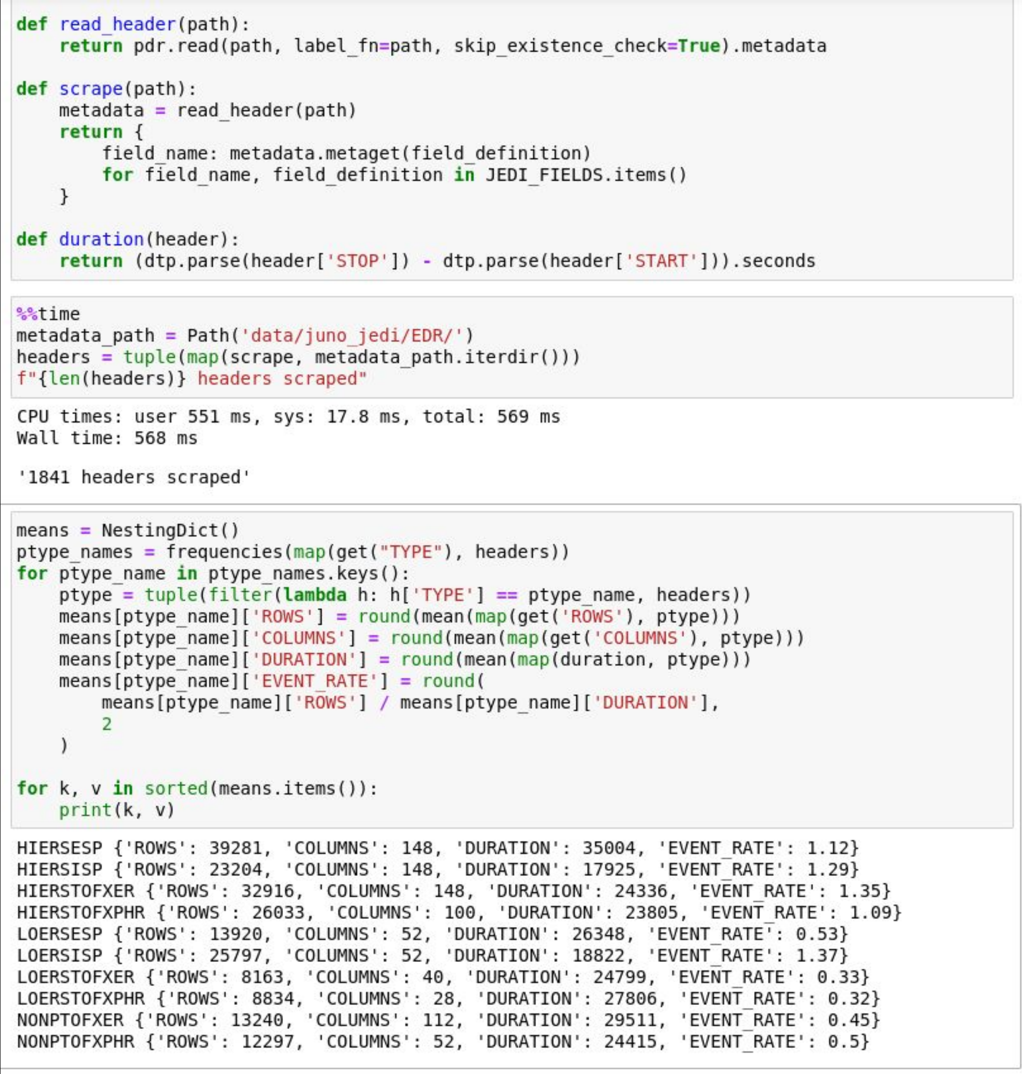


```
>>> data = pdr.read("/path/to/cr0_398560467edr_f0030004ccam02012m1.LBL")
>>> data['IMAGE']
array([[21, 21, 20, ..., 19, 19, 20],
       [21, 21, 21, ..., 19, 20, 20],
       [21, 21, 20, ..., 20, 20, 20],
       ...,
       [25, 25, 25, ..., 26, 26, 26],
       [25, 25, 25, ..., 27, 26, 26],
       [24, 25, 25, ..., 26, 26, 26]], dtype=int16)
```

Example:

It takes approx. half a second for pdr on a laptop to scrape the contents of >1800 JUNO JEDI headers into an array.

`pdr` is fast!



```
def read_header(path):
    return pdr.read(path, label_fn=path, skip_existence_check=True).metadata

def scrape(path):
    metadata = read_header(path)
    return {
        field_name: metadata.metaget(field_definition)
        for field_name, field_definition in JEDI_FIELDS.items()
    }

def duration(header):
    return (dtp.parse(header['STOP']) - dtp.parse(header['START'])).seconds
```

```
%%time
metadata_path = Path('data/juno_jedi/EDR/')
headers = tuple(map(scrape, metadata_path.iterdir()))
f"{len(headers)} headers scraped"
```

```
CPU times: user 551 ms, sys: 17.8 ms, total: 569 ms
Wall time: 568 ms

'1841 headers scraped'
```

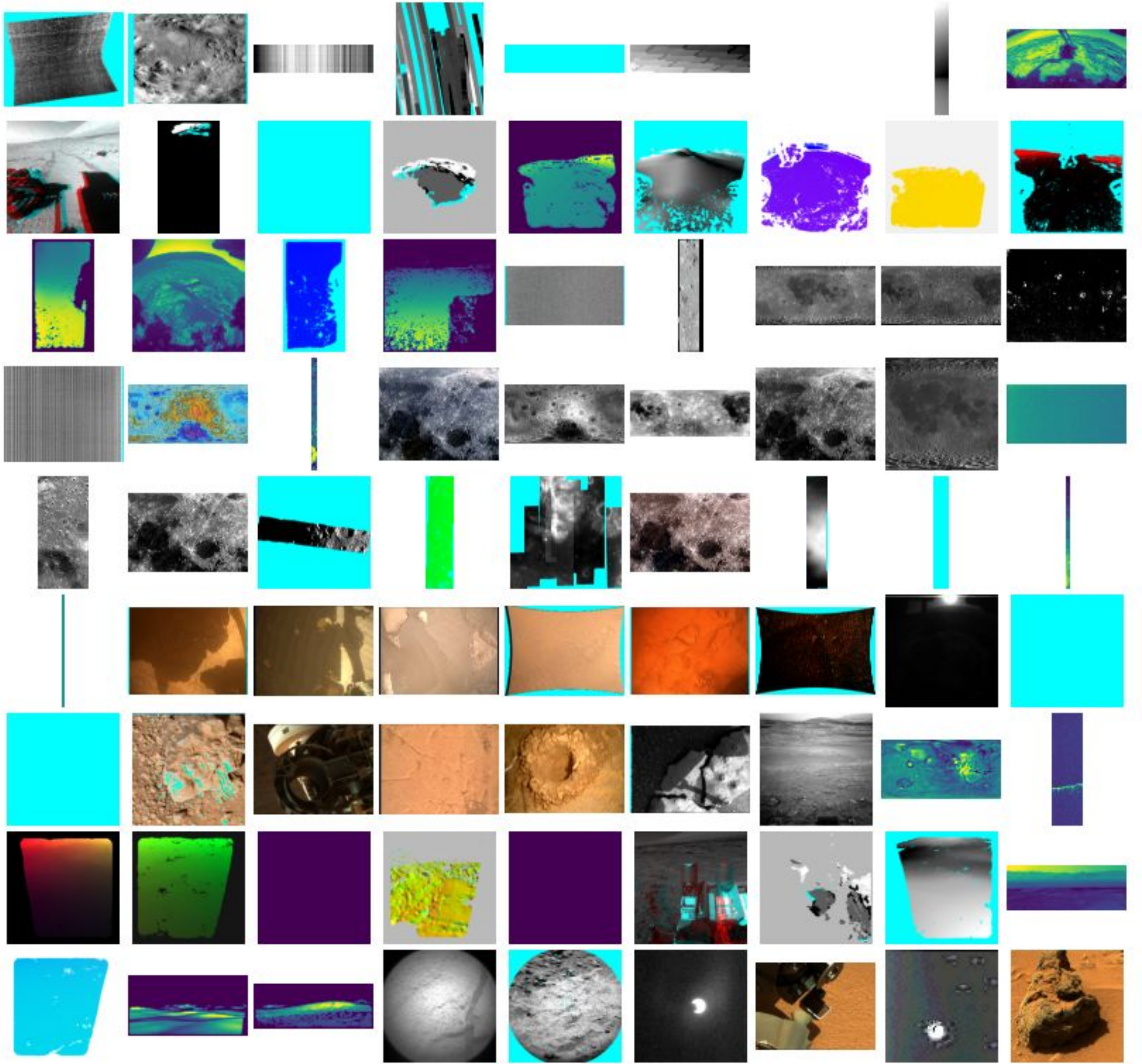
```
means = NestingDict()
ptype_names = frequencies(map(get("TYPE"), headers))
for ptype_name in ptype_names.keys():
    ptype = tuple(filter(lambda h: h['TYPE'] == ptype_name, headers))
    means[ptype_name]['ROWS'] = round(mean(map(get('ROWS'), ptype)))
    means[ptype_name]['COLUMNS'] = round(mean(map(get('COLUMNS'), ptype)))
    means[ptype_name]['DURATION'] = round(mean(map(duration, ptype)))
    means[ptype_name]['EVENT_RATE'] = round(
        means[ptype_name]['ROWS'] / means[ptype_name]['DURATION'],
        2
    )

for k, v in sorted(means.items()):
    print(k, v)
```

```
HIERSESP {'ROWS': 39281, 'COLUMNS': 148, 'DURATION': 35004, 'EVENT_RATE': 1.12}
HIERSISP {'ROWS': 23204, 'COLUMNS': 148, 'DURATION': 17925, 'EVENT_RATE': 1.29}
HIERSTOFXER {'ROWS': 32916, 'COLUMNS': 148, 'DURATION': 24336, 'EVENT_RATE': 1.35}
HIERSTOFXPHR {'ROWS': 26033, 'COLUMNS': 100, 'DURATION': 23805, 'EVENT_RATE': 1.09}
LOERSESP {'ROWS': 13920, 'COLUMNS': 52, 'DURATION': 26348, 'EVENT_RATE': 0.53}
LOERSISP {'ROWS': 25797, 'COLUMNS': 52, 'DURATION': 18822, 'EVENT_RATE': 1.37}
LOERSTOFXER {'ROWS': 8163, 'COLUMNS': 40, 'DURATION': 24799, 'EVENT_RATE': 0.33}
LOERSTOFXPHR {'ROWS': 8834, 'COLUMNS': 28, 'DURATION': 27806, 'EVENT_RATE': 0.32}
NONPTOFXER {'ROWS': 13240, 'COLUMNS': 112, 'DURATION': 29511, 'EVENT_RATE': 0.45}
NONPTOFXPHR {'ROWS': 12297, 'COLUMNS': 52, 'DURATION': 24415, 'EVENT_RATE': 0.5}
```

A lot of data already works!

pdr will eventually support nearly all data held by the PDS or stored in PDS-compliant formats (including legacy formats).

We are happy to entertain requests to prioritize support for specific data sets, especially if they are of immediate use in research or mission support.



`pdr-tests`

https://github.com/MillionConcepts/pdr-tests

`pdr-tests` is the test suite for `pdr`

Primarily contains regression tests. See Kaufman et. al, LPSC (2022) for more information.

Tests need to be very fast to be useful. `pdr-tests` contains a fast, flexible data indexing / sub-indexing tool called `ix`.

`ix` solves the problem of identify and retrieving very specific subsets of a data corpus, based on attributes of the filenames and file paths.

Example:

Do you want all of the EDRs from JUNO JEDI?

(1) Generated a "manifest" (database) of all possible files.

  (a) In this case, we have a manifest of everything held by the PDS Plasma node (in `PLASM_FILE`).

(2) Define the features of the files of interest.

  (a) in the Plasma manifest

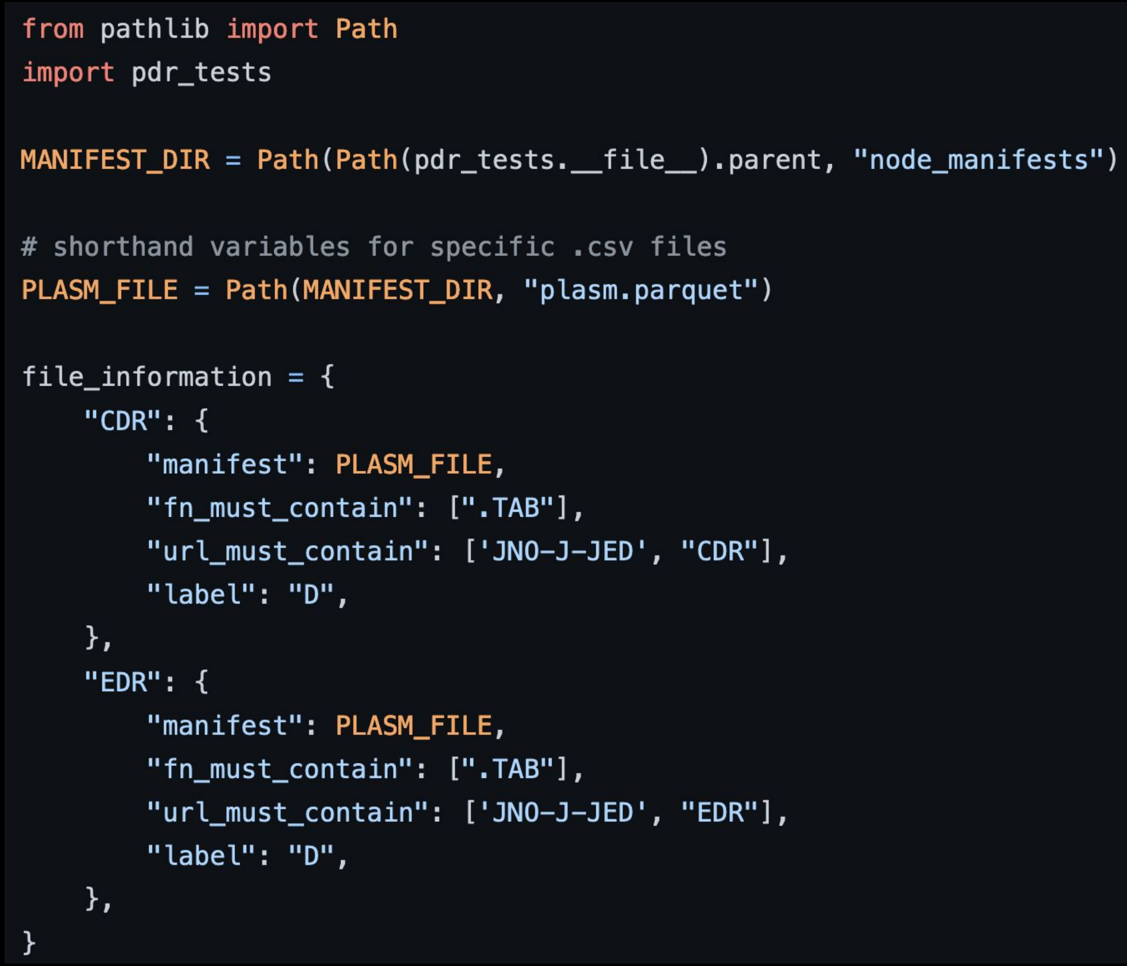
  (b) has a TAB extension

  (c) url contains JNO-J-JED and CDR

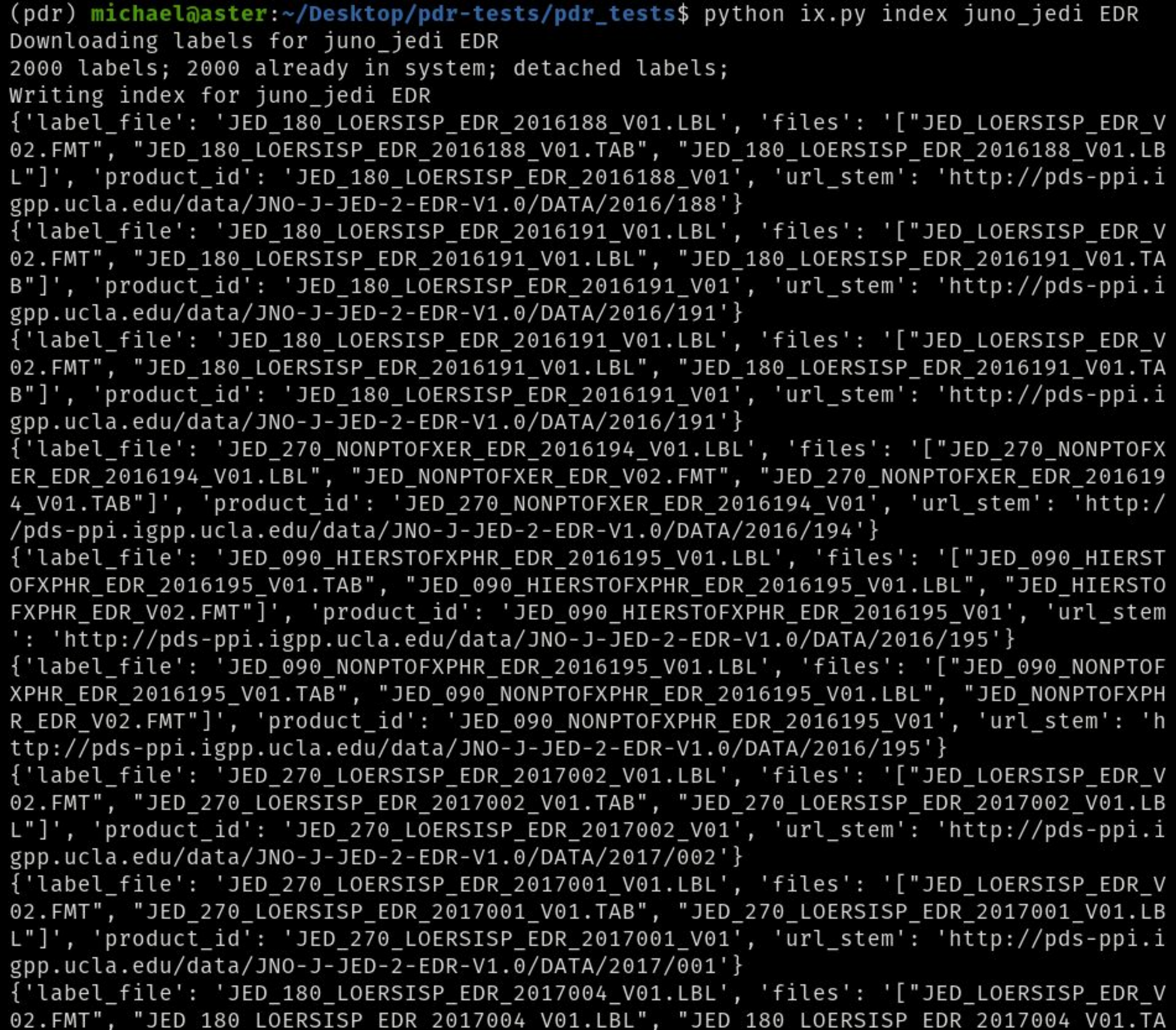
  (d) label is detached

(3) Then type

```
python ix.py index juno_jedi edr
```



```
from pathlib import Path
import pdr_tests

MANIFEST_DIR = Path(Path(pdr_tests.__file__).parent, "node_manifests")

# shorthand variables for specific .csv files
PLASM_FILE = Path(MANIFEST_DIR, "plasm.parquet")

file_information = {
    "CDR": {
        "manifest": PLASM_FILE,
        "fn_must_contain": [".TAB"],
        "url_must_contain": ['JNO-J-JED', "CDR"],
        "label": "D",
    },
    "EDR": {
        "manifest": PLASM_FILE,
        "fn_must_contain": [".TAB"],
        "url_must_contain": ['JNO-J-JED', "EDR"],
        "label": "D",
    },
}
```



```
(pdr) michael@aster:~/Desktop/pdr-tests/pdr_tests$ python ix.py index juno_jedi EDR
Downloading labels for juno_jedi EDR
2000 labels; 2000 already in system; detached labels;
Writing index for juno_jedi EDR
{'label_file': 'JED_180_LOERSISP_EDR_2016188_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_180_LOERSISP_EDR_2016188_V01.TAB", "JED_180_LOERSISP_EDR_2016188_V01.LB
L"]', 'product_id': 'JED_180_LOERSISP_EDR_2016188_V01', 'url_stem': 'http://pds-ppi.i
gpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/188'}
{'label_file': 'JED_180_LOERSISP_EDR_2016191_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_180_LOERSISP_EDR_2016191_V01.LBL", "JED_180_LOERSISP_EDR_2016191_V01.TA
B"]', 'product_id': 'JED_180_LOERSISP_EDR_2016191_V01', 'url_stem': 'http://pds-ppi.i
gpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/191'}
{'label_file': 'JED_180_LOERSISP_EDR_2016191_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_180_LOERSISP_EDR_2016191_V01.LBL", "JED_180_LOERSISP_EDR_2016191_V01.TA
B"]', 'product_id': 'JED_180_LOERSISP_EDR_2016191_V01', 'url_stem': 'http://pds-ppi.i
gpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/191'}
{'label_file': 'JED_270_NONPTOFXER_EDR_2016194_V01.LBL', 'files': '["JED_270_NONPTOFX
ER_EDR_2016194_V01.LBL", "JED_NONPTOFXER_EDR_V02.FMT", "JED_270_NONPTOFXER_EDR_201619
4_V01.TAB"]', 'product_id': 'JED_270_NONPTOFXER_EDR_2016194_V01', 'url_stem': 'http:/
/pds-ppi.igpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/194'}
{'label_file': 'JED_090_HIERSTOFXPHR_EDR_2016195_V01.LBL', 'files': '["JED_090_HIERST
OFXPHR_EDR_2016195_V01.TAB", "JED_090_HIERSTOFXPHR_EDR_2016195_V01.LBL", "JED_HIERSTO
FXPHR_EDR_V02.FMT"]', 'product_id': 'JED_090_HIERSTOFXPHR_EDR_2016195_V01', 'url_stem
': 'http://pds-ppi.igpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/195'}
{'label_file': 'JED_090_NONPTOFXPHR_EDR_2016195_V01.LBL', 'files': '["JED_090_NONPTOF
XPHR_EDR_2016195_V01.TAB", "JED_090_NONPTOFXPHR_EDR_2016195_V01.LBL", "JED_NONPTOFXPH
R_EDR_V02.FMT"]', 'product_id': 'JED_090_NONPTOFXPHR_EDR_2016195_V01', 'url_stem': 'h
ttp://pds-ppi.igpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2016/195'}
{'label_file': 'JED_270_LOERSISP_EDR_2017002_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_270_LOERSISP_EDR_2017002_V01.TAB", "JED_270_LOERSISP_EDR_2017002_V01.LB
L"]', 'product_id': 'JED_270_LOERSISP_EDR_2017002_V01', 'url_stem': 'http://pds-ppi.i
gpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2017/002'}
{'label_file': 'JED_270_LOERSISP_EDR_2017001_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_270_LOERSISP_EDR_2017001_V01.TAB", "JED_270_LOERSISP_EDR_2017001_V01.LB
L"]', 'product_id': 'JED_270_LOERSISP_EDR_2017001_V01', 'url_stem': 'http://pds-ppi.i
gpp.ucla.edu/data/JNO-J-JED-2-EDR-V1.0/DATA/2017/001'}
{'label_file': 'JED_180_LOERSISP_EDR_2017004_V01.LBL', 'files': '["JED_LOERSISP_EDR_V
02.FMT", "JED_180_LOERSISP_EDR_2017004_V01.LBL", "JED_180_LOERSISP_EDR_2017004_V01.TA
```

`marslab`

https://github.com/MillionConcepts/marslab

An eclectic collection of utilities for working with observational data of Mars, especially multispectral data from rovers. A lot of functionality not previously standardized in Python.

Includes:

(1) Implementations of spectral image and band parameter operations (e.g. band depth, decorrelation stretch, debayering).

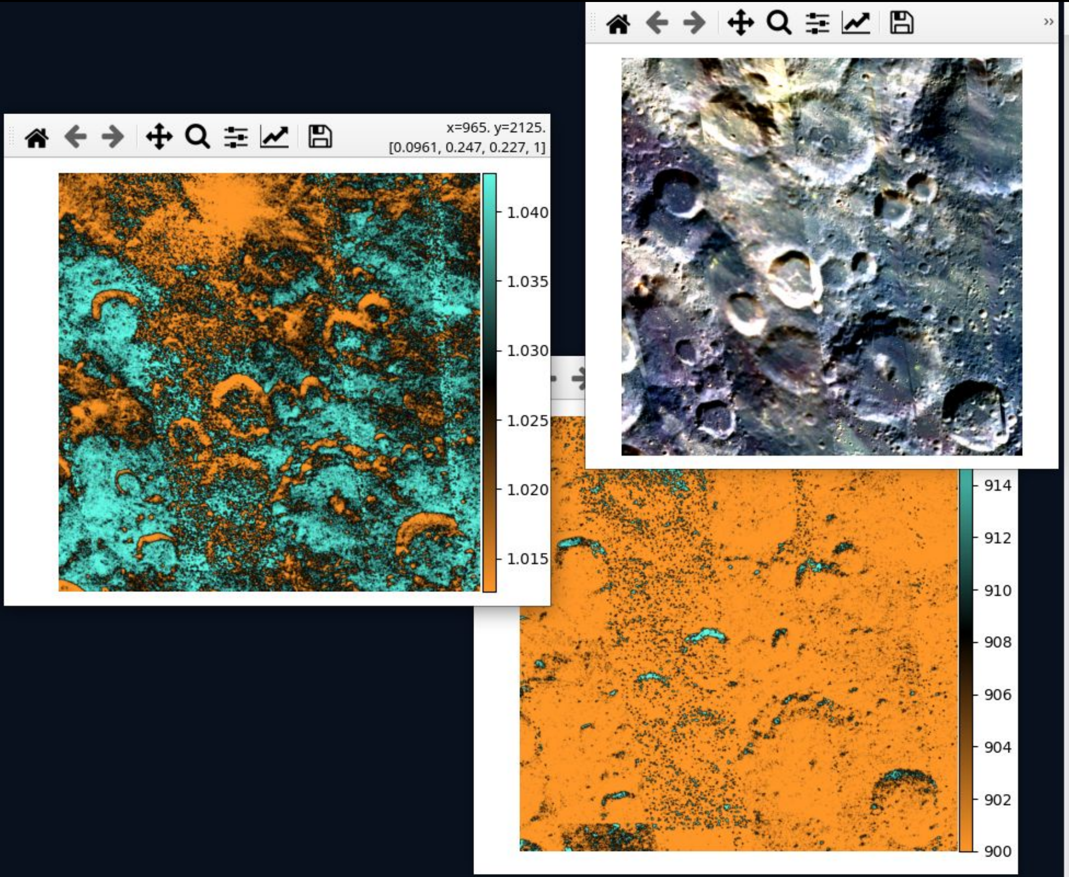
(2) Implementations for photometric ROI extraction, including proper handling of bayer filters and masks.

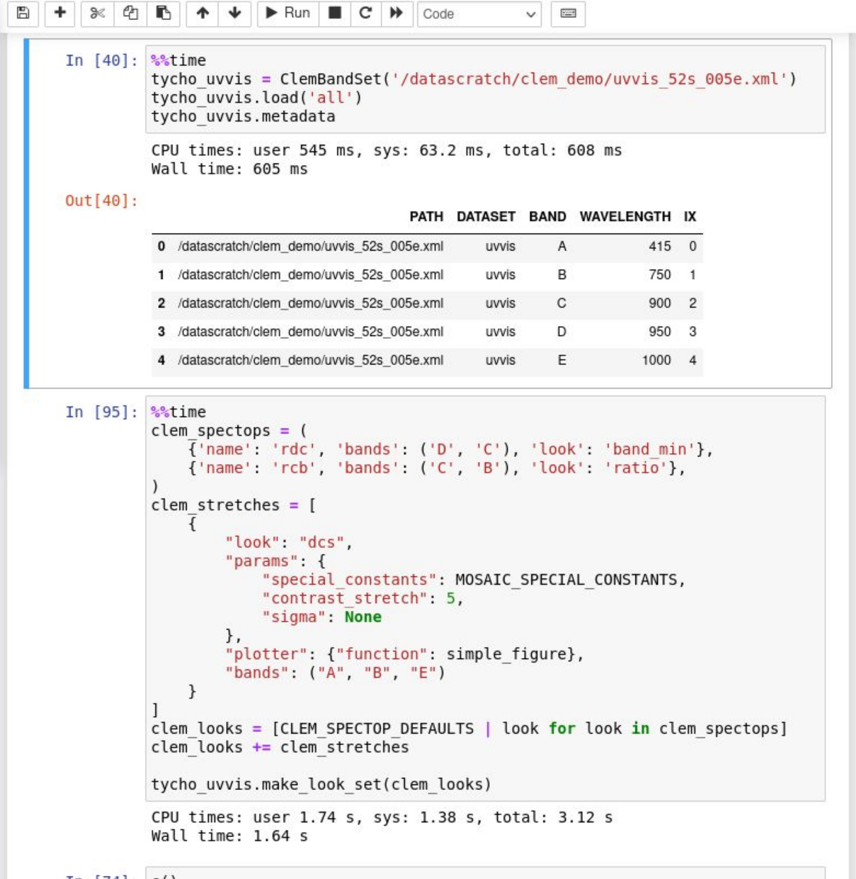
(3) Mission-specific metadata and filename parsing.

(4) Support for Mars local true solar time.

(5) `bandset` data object class, providing an OO approach to multispectral image analysis.

Example: ~2 seconds to generate a DCS, band minimum, and band ratio map for Clementine UVVIS.





```
In [40]: %%time
         tycho_uvvis = ClemBandSet('/datascratch/clem_demo/uvvis_52s_005e.xml')
         tycho_uvvis.load('all')
         tycho_uvvis.metadata

         CPU times: user 545 ms, sys: 63.2 ms, total: 608 ms
         Wall time: 605 ms
```

Out[40]:

	PATH	DATASET	BAND	WAVELENGTH	IX
0	/datascratch/clem_demo/uvvis_52s_005e.xml	uvvis	A	415	0
1	/datascratch/clem_demo/uvvis_52s_005e.xml	uvvis	B	750	1
2	/datascratch/clem_demo/uvvis_52s_005e.xml	uvvis	C	900	2
3	/datascratch/clem_demo/uvvis_52s_005e.xml	uvvis	D	950	3
4	/datascratch/clem_demo/uvvis_52s_005e.xml	uvvis	E	1000	4

```
In [95]: %%time
         clem_spectops = (
             {'name': 'rdc', 'bands': ('D', 'C'), 'look': 'band_min'},
             {'name': 'rcb', 'bands': ('C', 'B'), 'look': 'ratio'},
         )
         clem_stretches = [
             {
                 "look": "dcs",
                 "params": {
                     "special_constants": MOSAIC_SPECIAL_CONSTANTS,
                     "contrast_stretch": 5,
                     "sigma": None
                 },
                 "plotter": {"function": simple_figure},
                 "bands": ("A", "B", "E")
             }
         ]
         clem_looks = [CLEM_SPECTOP_DEFAULTS | look for look in clem_spectops]
         clem_looks += clem_stretches

         tycho_uvvis.make_look_set(clem_looks)

         CPU times: user 1.74 s, sys: 1.38 s, total: 3.12 s
         Wall time: 1.64 s
```

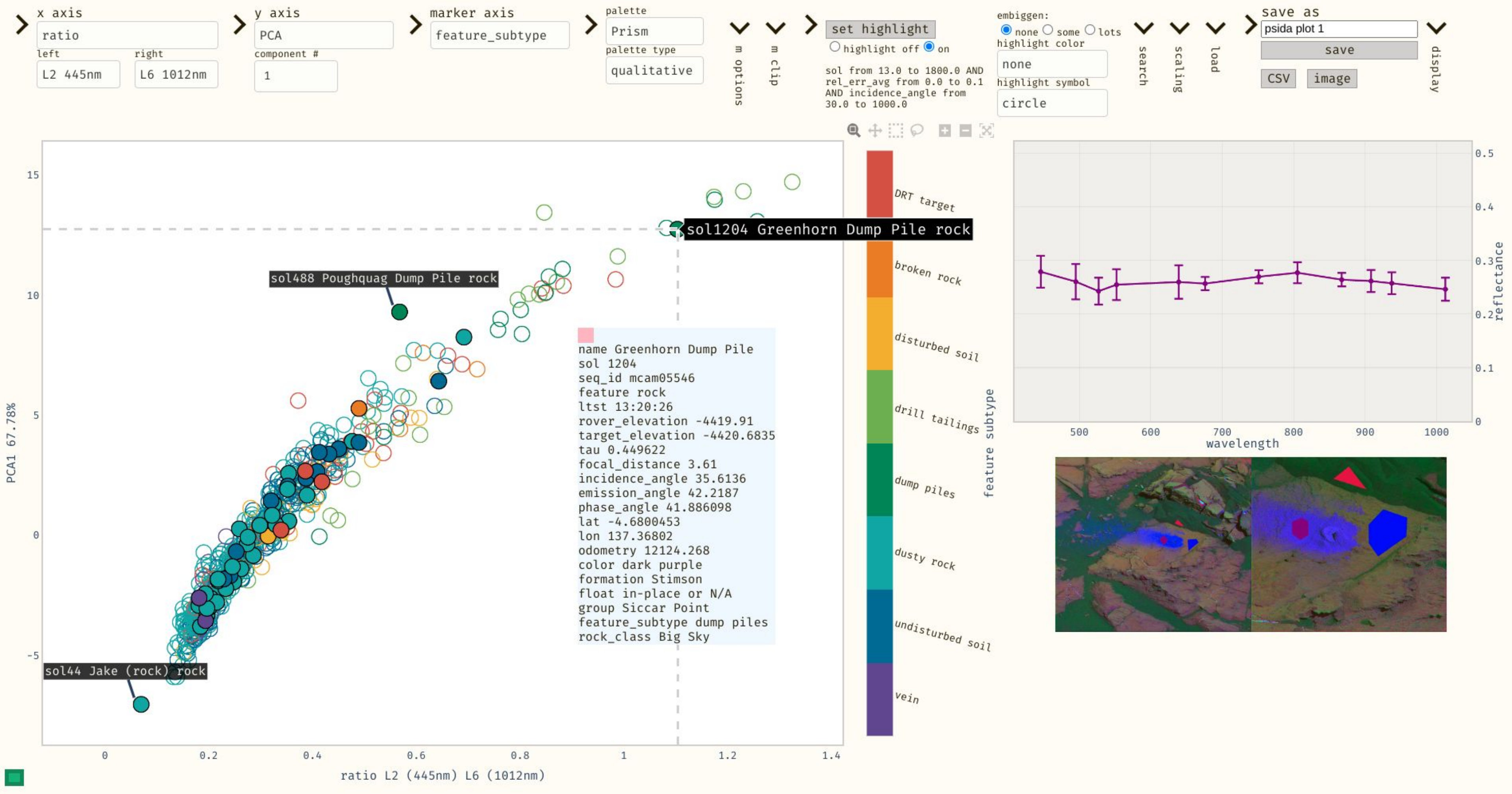
`multidex`

https://github.com/MillionConcepts/multidex

The [Multi]spectral [D]ata [Ex]plorer is an in-browser GUI that enables fast, massively multi-dimensional exploration of spectral imaging data.

The backend and UI are almost trivially extensible to any multispectral imager. Currently in use on Mastcam, Mastcam-Z, and Chemcam. Also works with laboratory spectra. Will work for any data that can be represented in the `marslab` format.

`multidex` helps with both scientific (e.g. rapid tactical and long term strategic analysis) and engineering uses cases (e.g. calibration validation and anomaly investigation).



x axis ratio

left L2 445nm right L6 1012nm

y axis PCA

component # 1

marker axis feature_subtype

palette Prism

palette type qualitative

set highlight

○ highlight off ● on

```
sol from 13.0 to 1800.0 AND
rel_err_avg from 0.0 to 0.1
AND incidence_angle from
30.0 to 1000.0
```

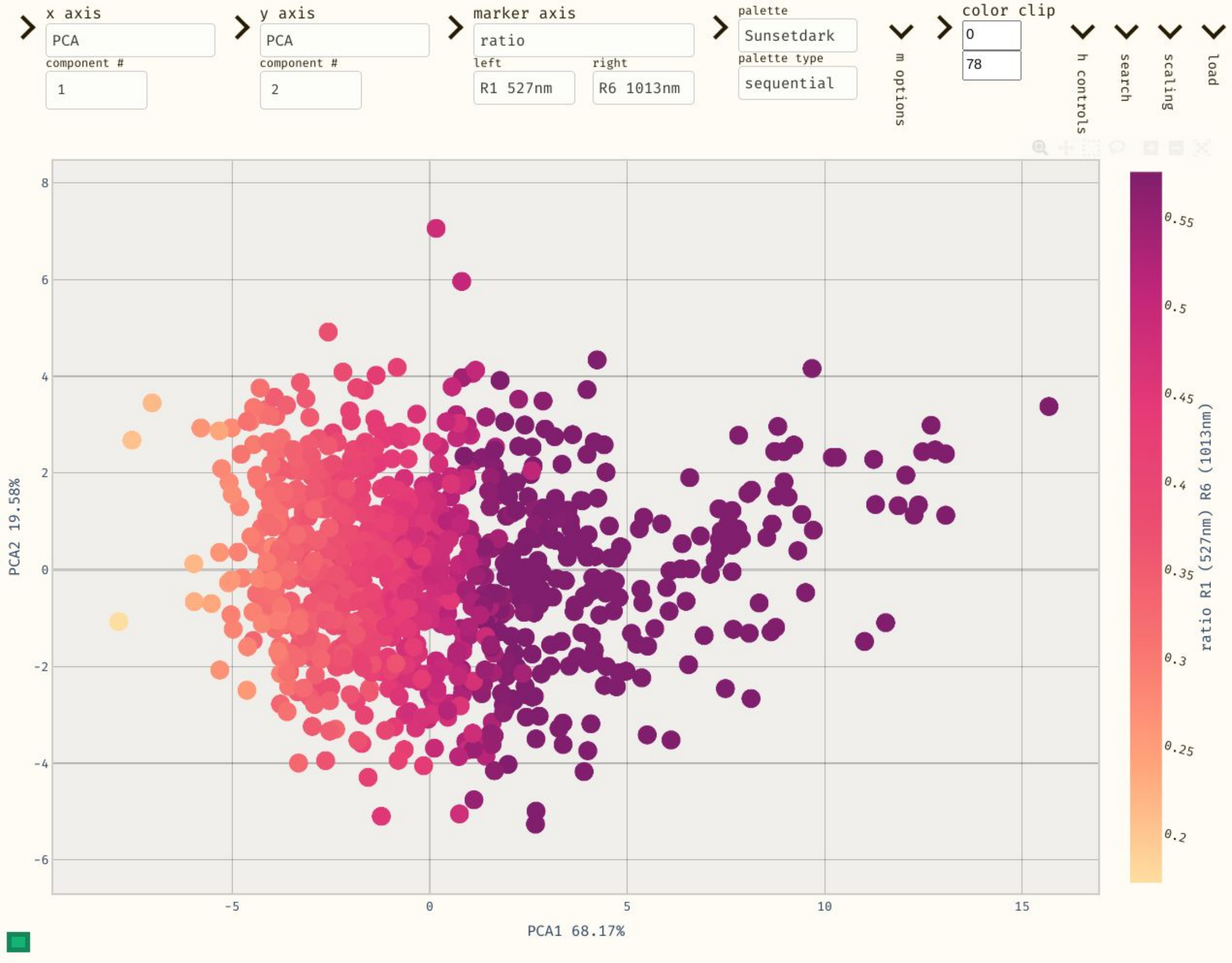
embiggen: ● none ○ some ○ lots

highlight color none

highlight symbol circle

save as psida plot 1

```
name Greenhorn Dump Pile
sol 1204
seq_id mcam05546
feature rock
ltst 13:20:26
rover_elevation -4419.91
target_elevation -4420.6835
tau 0.449622
focal_distance 3.61
incidence_angle 35.6136
emission_angle 42.2187
phase_angle 41.886098
lat -4.6800453
lon 137.36802
odometry 12124.268
color dark purple
formation Stimson
float in-place or N/A
group Siccar Point
feature_subtype dump piles
rock_class Big Sky
```



x axis: PCA
component #: 1

y axis: PCA
component #: 2

marker axis: ratio
left: R1 527nm
right: R6 1013nm

palette: Sunsetdark
palette type: sequential

color clip: 0 – 78

`asdf`

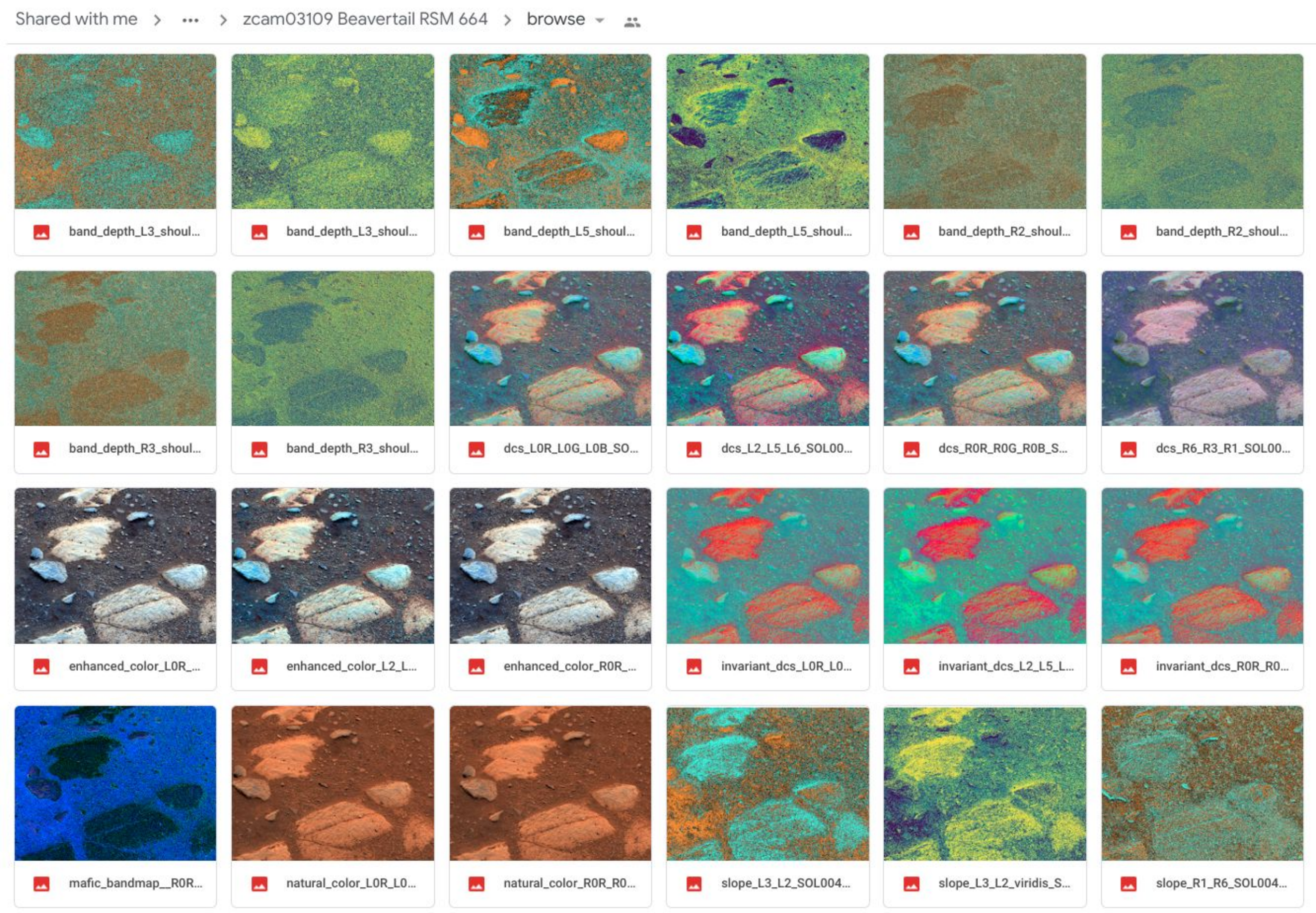
[code not yet publicly available]
[happy to discuss implementation details]

[a]rchive [s]pectral [d]ata [f]unctions generates rapid last-mile data reduction for multispectral imagers with nearly a keysmash: `asdf [path-to-data]`

Generates a large number of derived representations of the data very quickly. Estimated five orders of magnitude better time-to-analysis compared to prior (highly manual) methods.

Backs up analyses immediately to archive-ready formats. Integrates tightly with Google Workspace ecosystem for collaboration. Makes extensive use of `marslab` and `pdr`. Outputs are compatible with `multidex`.

See Million et al., LPSC (2022) for more information.



Shared with me > ••• > zcam03109 Beavertail RSM 664 > browse

band_depth_L3_shoul...

band_depth_L3_shoul...

band_depth_L5_shoul...

band_depth_L5_shoul...

band_depth_R2_shoul...

band_depth_R2_shoul...

band_depth_R3_shoul...

band_depth_R3_shoul...

dcs_L0R_L0G_L0B_SO...

dcs_L2_L5_L6_SOL00...

dcs_R0R_R0G_R0B_S...

dcs_R6_R3_R1_SOL00...

enhanced_color_L0R_...

enhanced_color_L2_L...

enhanced_color_R0R_...

invariant_dcs_L0R_L0...

invariant_dcs_L2_L5_L...

invariant_dcs_R0R_R0...

mafic_bandmap__R0R...

natural_color_L0R_L0...

natural_color_R0R_R0...

slope_L3_L2_SOL004...

slope_L3_L2_viridis_S...

slope_R1_R6_SOL004...

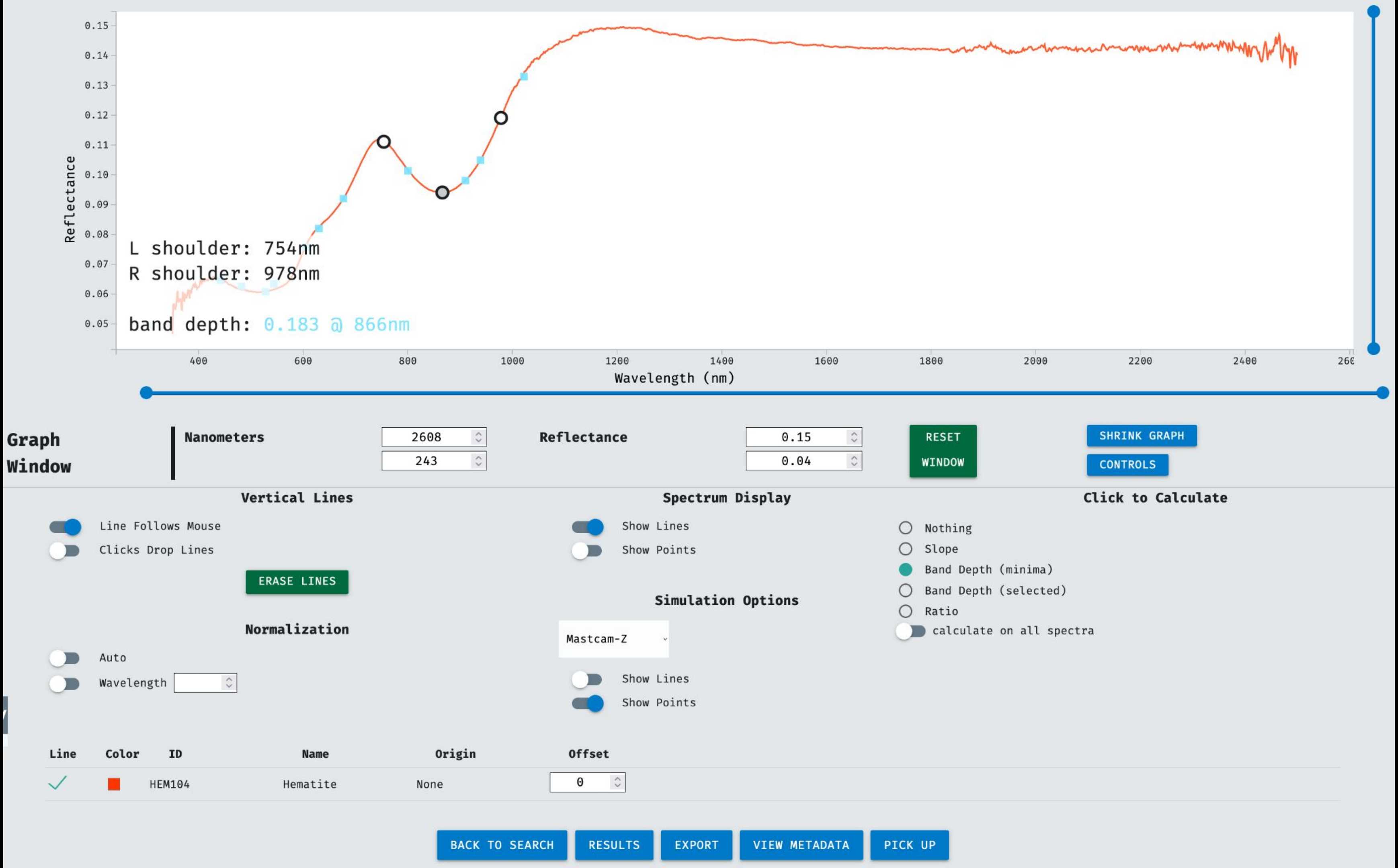
VISOR

https://westernreflectancelab.com/visor/

[Vis]ible-[I]nfrared [S]pectral [O]bject [R]epository is a searchable catalog of compiled high-resolution VIS-IR laboratory reflectance spectra.

Point-and-click operations for basic band parameter operations.

Data can be convolved to mission bandpasses and exported into the `marslab` format.



L shoulder: 754nm
R shoulder: 978nm

band depth: 0.183 @ 866nm

Graph Window

Nanometers 2608 / 243

Reflectance 0.15 / 0.04

Vertical Lines

- ● Line Follows Mouse
- ○ Clicks Drop Lines

Normalization

- ○ Auto
- ○ Wavelength

Spectrum Display

- ● Show Lines
- ○ Show Points

Simulation Options

Mastcam-Z

- ○ Show Lines
- ● Show Points

Click to Calculate

- ○ Nothing
- ○ Slope
- ● Band Depth (minima)
- ○ Band Depth (selected)
- ○ Ratio
- ○ calculate on all spectra

Line	Color	ID	Name	Origin	Offset
✓	🟥	HEM104	Hematite	None	0

`killscreen`

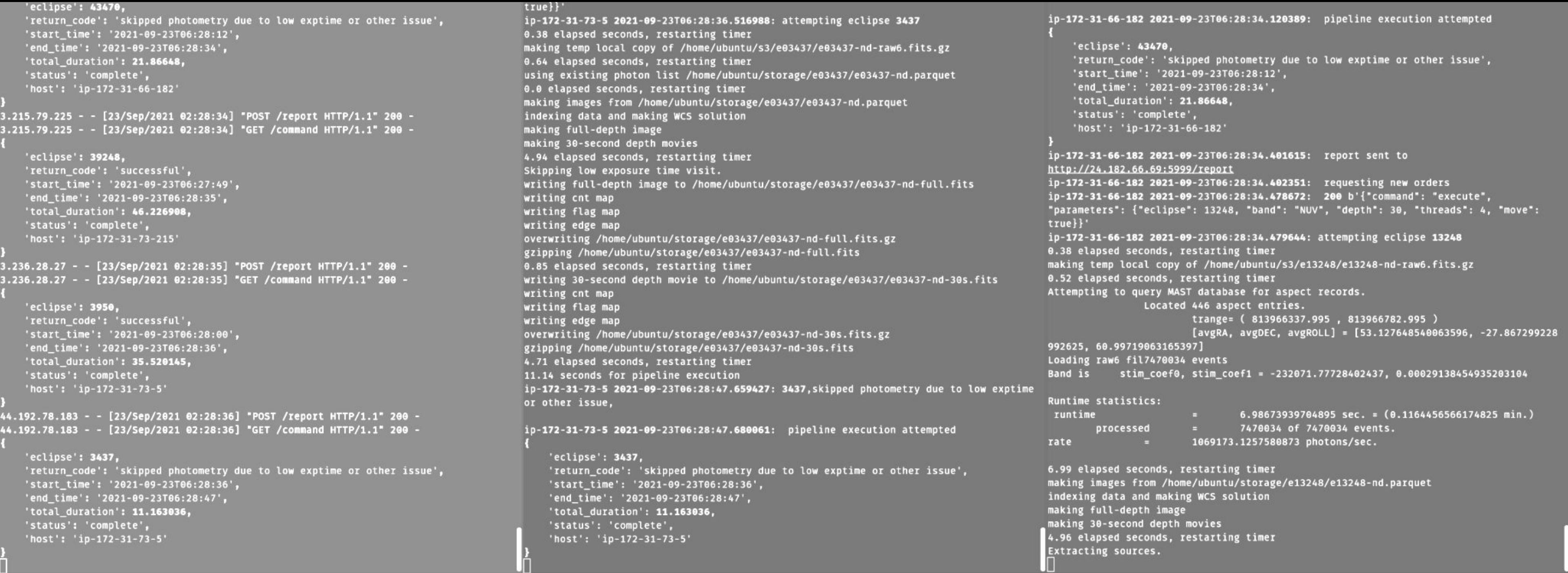
https://github.com/MillionConcepts/killscreen

An idiomatically Pythonic interface to cloud data processing resources. Fits neatly within Python data science workflows (e.g. Jupyter notebooks).

From a blank Python session, launching an arbitrarily large cluster is ~5 lines of code.

Example:

Killscreen managing a cluster of EC2 instances to recalibrate the ~2Tb data corpus from the Galaxy Evolution Explorer (GALEX) mission.



```
    'eclipse': 43470,
    'return_code': 'skipped photometry due to low exptime or other issue',
    'start_time': '2021-09-23T06:28:12',
    'end_time': '2021-09-23T06:28:34',
    'total_duration': 21.86648,
    'status': 'complete',
    'host': 'ip-172-31-66-182'
}
3.215.79.225 - - [23/Sep/2021 02:28:34] "POST /report HTTP/1.1" 200 -
3.215.79.225 - - [23/Sep/2021 02:28:34] "GET /command HTTP/1.1" 200 -
{
    'eclipse': 39248,
    'return_code': 'successful',
    'start_time': '2021-09-23T06:27:49',
    'end_time': '2021-09-23T06:28:35',
    'total_duration': 46.226908,
    'status': 'complete',
    'host': 'ip-172-31-73-215'
}
3.236.28.27 - - [23/Sep/2021 02:28:35] "POST /report HTTP/1.1" 200 -
3.236.28.27 - - [23/Sep/2021 02:28:35] "GET /command HTTP/1.1" 200 -
{
    'eclipse': 3950,
    'return_code': 'successful',
    'start_time': '2021-09-23T06:28:00',
    'end_time': '2021-09-23T06:28:36',
    'total_duration': 35.520145,
    'status': 'complete',
    'host': 'ip-172-31-73-5'
}
44.192.78.183 - - [23/Sep/2021 02:28:36] "POST /report HTTP/1.1" 200 -
44.192.78.183 - - [23/Sep/2021 02:28:36] "GET /command HTTP/1.1" 200 -
{
    'eclipse': 3437,
    'return_code': 'skipped photometry due to low exptime or other issue',
    'start_time': '2021-09-23T06:28:36',
    'end_time': '2021-09-23T06:28:47',
    'total_duration': 11.163036,
    'status': 'complete',
    'host': 'ip-172-31-73-5'
}
```

```
true}}'
ip-172-31-73-5 2021-09-23T06:28:36.516988: attempting eclipse 3437
0.38 elapsed seconds, restarting timer
making temp local copy of /home/ubuntu/s3/e03437/e03437-nd-raw6.fits.gz
0.64 elapsed seconds, restarting timer
using existing photon list /home/ubuntu/storage/e03437/e03437-nd.parquet
0.0 elapsed seconds, restarting timer
making images from /home/ubuntu/storage/e03437/e03437-nd.parquet
indexing data and making WCS solution
making full-depth image
making 30-second depth movies
4.94 elapsed seconds, restarting timer
Skipping low exposure time visit.
writing full-depth image to /home/ubuntu/storage/e03437/e03437-nd-full.fits
writing cnt map
writing flag map
writing edge map
overwriting /home/ubuntu/storage/e03437/e03437-nd-full.fits.gz
gzipping /home/ubuntu/storage/e03437/e03437-nd-full.fits
0.85 elapsed seconds, restarting timer
writing 30-second depth movie to /home/ubuntu/storage/e03437/e03437-nd-30s.fits
writing cnt map
writing flag map
writing edge map
overwriting /home/ubuntu/storage/e03437/e03437-nd-30s.fits.gz
gzipping /home/ubuntu/storage/e03437/e03437-nd-30s.fits
4.71 elapsed seconds, restarting timer
11.14 seconds for pipeline execution
ip-172-31-73-5 2021-09-23T06:28:47.659427: 3437,skipped photometry due to low exptime
or other issue,

ip-172-31-73-5 2021-09-23T06:28:47.680061:  pipeline execution attempted
{
    'eclipse': 3437,
    'return_code': 'skipped photometry due to low exptime or other issue',
    'start_time': '2021-09-23T06:28:36',
    'end_time': '2021-09-23T06:28:47',
    'total_duration': 11.163036,
    'status': 'complete',
    'host': 'ip-172-31-73-5'
}
```

```
ip-172-31-66-182 2021-09-23T06:28:34.120389:  pipeline execution attempted
{
    'eclipse': 43470,
    'return_code': 'skipped photometry due to low exptime or other issue',
    'start_time': '2021-09-23T06:28:12',
    'end_time': '2021-09-23T06:28:34',
    'total_duration': 21.86648,
    'status': 'complete',
    'host': 'ip-172-31-66-182'
}
ip-172-31-66-182 2021-09-23T06:28:34.401615:  report sent to
http://24.182.66.69:5999/report
ip-172-31-66-182 2021-09-23T06:28:34.402351:  requesting new orders
ip-172-31-66-182 2021-09-23T06:28:34.478672:  200 b'{"command": "execute",
"parameters": {"eclipse": 13248, "band": "NUV", "depth": 30, "threads": 4, "move":
true}}'
ip-172-31-66-182 2021-09-23T06:28:34.479644: attempting eclipse 13248
0.38 elapsed seconds, restarting timer
making temp local copy of /home/ubuntu/s3/e13248/e13248-nd-raw6.fits.gz
0.52 elapsed seconds, restarting timer
Attempting to query MAST database for aspect records.
                Located 446 aspect entries.
                        trange= ( 813966337.995 , 813966782.995 )
                        [avgRA, avgDEC, avgROLL] = [53.127648540063596, -27.867299228
992625, 60.99719063165397]
Loading raw6 fil7470034 events
Band is     stim_coef0, stim_coef1 = -232071.77728402437, 0.00029138454935203104

Runtime statistics:
 runtime                =       6.98673939704895 sec. = (0.1164456566174825 min.)
        processed       =       7470034 of 7470034 events.
rate            =       1069173.1257580873 photons/sec.

6.99 elapsed seconds, restarting timer
making images from /home/ubuntu/storage/e13248/e13248-nd.parquet
indexing data and making WCS solution
making full-depth image
making 30-second depth movies
4.96 elapsed seconds, restarting timer
Extracting sources.
```

Warning:

`killscreen` has no safety switch!

Make sure that you know what you're doing
and how much it will cost.

pdr	https://github.com/millionconcepts/pdr
pdr-tests	https://github.com/MillionConcepts/pdr-tests
marslab	https://github.com/MillionConcepts/marslab
multidex	https://github.com/MillionConcepts/multidex
asdf	
VISOR	https://westernreflectancelab.com/visor/
killscreen	https://github.com/MillionConcepts/killscreen

The development of `pdr` is supported by NASA grant No. 80NSSC21K0885. We would like to thank the Planetary Data System (PDS) for their continued cooperation. The development of `asdf`, `multidex`, `marslab`, and `VISOR` have been supported in part by the Mars 2020 and Mars Science Laboratory projects.