ACROSS

ENABLING TIME DOMAIN AND MULTI-MESSENGER ASTROPHYSICS

Samuel Wyatt (GSFC), ACROSS Support Scientist

AAS 01/07/2026

Tools - Client Demo

Core Team:
- Chris Roberts, Brian Humensky, Craig Pellegrino, Tyler Pritchard, Kirill Vorobyev, Nitzan Frock (GSFC)
- Jamie Kennea (PSU)
- Dan Kocevski, Michelle Hui (Marshall)

ACROSS: Demo Walkthrough

- **Backend**
  - ACROSS - Server, Tools, SDK, Client, and Tools
    - Tools - python library that performs core astronomy calculations: e.g. footprint projection, overlap. agnostic bandpass, joint visibility calculations, plotting… etc
    - Server - FastAPI server that handles all endpoints (inherits Tools)
    - SDK - pure http requests (open-api - auto-generated)
    - Client - python library that wraps the SDK for astronomer convenience (also inherits Tools).
    - Data-Ingestion - FastAPI cron jobs that push to Server(inherits Tools, Client)

ACROSS: Demo Walkthrough

- **Backend**
  - ACROSS - Server, Tools, SDK, Client, and Tools
    - **Tools** - python library that performs core astronomy calculations: e.g. footprint projection, overlap. agnostic bandpass, joint visibility calculations, plotting… etc
    - Server - FastAPI server that handles all endpoints (inherits Tools)
    - SDK - pure http requests (open-api - auto-generated)
    - **Client** - python library that wraps the SDK for astronomer convenience (also inherits Tools).
    - Data-Ingestion - FastAPI cron jobs that push to Server(inherits Tools, Client)

ACROSS: Demo Walkthrough

- **OPEN SOURCE and Public (this is a big deal)**
  - https://github.com/NASA-ACROSS

ACROSS

NASA - Astrophysics Cross Observatory Science Support (ACROSS) Project

3 followers · United States of America · https://science.data.nasa.gov/data-...

Popular repositories

across-server — Public

Python · ☆ 1

across-tools — Public

A set of NASA multi-mission observing tools for situational awareness developed for the across-server, and other applications.

Python · ☆ 1

across-data-ingestion — Public

Ingestion container for observatory schedules

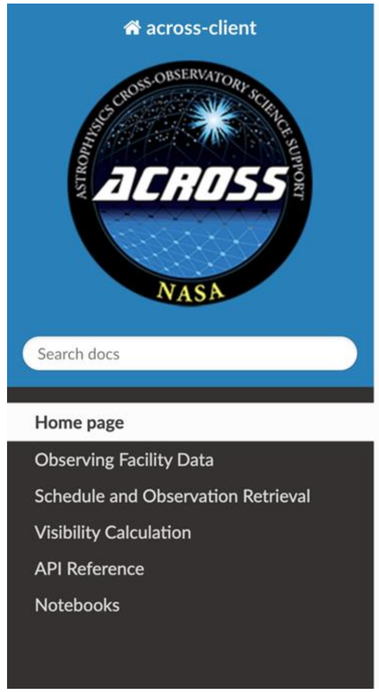
Python

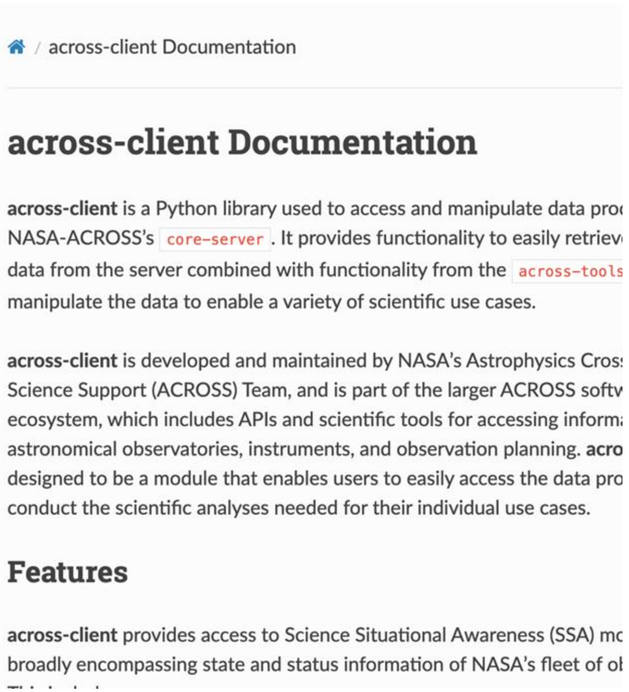
across-server-openapi-python — Public

Auto generated OpenAPI spec client for python

Python

```
pip install across-client
pip install across-tools
```





ACROSS: Demo Walkthrough

- **Backend**
  - ACROSS - Tools
    - Visibility Calculations

```python
from across.tools.visibility import EphemerisVisibility

vis = EphemerisVisibility(
    ra=ra,
    dec=dec,
    ephemeris=swift_ephemeris,
    constraints=constraints,
    begin=begin,
    end=end,
    observatory_name="Swift",
    observatory_id="8ef5f92d-c742-4484-9ca6-4d507e9924b7"
)

vis.compute()
```

ACROSS: Demo Walkthrough

- **Backend**
  - ACROSS - Tools
    - Visibility Calculations
    - Footprint Analysis

```
from across.tools.footprint import Footprint

nircam_footprint = [[{'x': ..., 'y': ...}]]
nircam_tools_footprint = Footprint.from_across_api(nircam_footprint)

projected_footprint = nircam_tools_footprint.project(
    Coordinate(ra=197.450341598, dec=-23.3814675445),
    roll_angle=45
)
projected_footprint.query_pixels()
```

✓ 0.0s

```
[8790213, 8782022, 8786118, 8794310, 8790214, 8786119]
```

ACROSS: Demo Walkthrough

- **Backend**
  - ACROSS - Tools
    - Visibility Calculations
    - Footprint Analysis
    - Bandpass analysis

```
from across.tools import EnergyBandpass, convert_to_wave

bandpass = EnergyBandpass(
    filter_name="Swift XRT", min=2.0, max=10.0, unit="keV"
)
convert_to_wave(bandpass)
```

✓ 0.0s

```
{'filter_name': 'Swift XRT',
'min': 1.2398419843320028,
'max': 6.199209921660012,
'type': 'WAVELENGTH',
'central_wavelength': 3.7195259529960074,
'peak_wavelength': None,
'bandwidth': 2.4796839686640046,
'unit': <WavelengthUnit.ANGSTROM: 'angstrom'>}
```

ACROSS: Demo Walkthrough

- ACROSS - Client
  - Observatory

```
from across.client import Client

client = Client()

client.observatory.get_many(
    name="jwst",
)
```

✓ 0.1s


```
{'id': 'fefe0ddc-07af-4312-b3a8-051c4e646619',
 'created_on': '2025-07-22T23:21:40.537450',
 'name': 'James Webb Space Telescope',
 'short_name': 'JWST',
 'type': 'SPACE_BASED',
 'telescopes': [{'id': '225ad468-585f-4f5e-8b64-09a4adfa1b7d',
   'name': 'James Webb Space Telescope',
   'short_name': 'JWST'}],
 'ephemeris_types': [{'ephemeris_type': 'tle',
   'priority': 1,
   'parameters': {'norad_id': 50463, 'norad_satellite_name': 'JWST'}}]}
```

ACROSS: Demo Walkthrough

- ACROSS - Client
  - Telescopes

```
from across.client import Client

client = Client()

client.telescope.get_many(
    name="jwst",
)
```

✓ 0.0s


```
[{'id': '225ad468-585f-4f5e-8b64-09a4adfa1b7d',
  'created_on': '2025-07-22T23:21:40.538993',
  'name': 'James Webb Space Telescope',
  'short_name': 'JWST',
  'observatory': {'id': 'fefe0ddc-07af-4312-b3a8-051c4e646619',
   'name': 'James Webb Space Telescope',
   'short_name': 'JWST'},
  'instruments': [{'id': 'd21c66bc-7173-454e-8f67-69df1b43590d',
    'name': 'Mid-Infrared Instrument',
    'short_name': 'JWST_MIRI'},
   {'id': '9899d36c-9e07-4927-8295-04c4ced70f1a',
    'name': 'Near-Infrared Camera',
    'short_name': 'JWST_NIRCAM'},
   {'id': '905bd0aa-6be2-43a1-b1d3-bb8df1632ac8',
    'name': 'Near-Infrared Imager and Slitless Spectrograph',
    'short_name': 'JWST_NIRISS'},
   {'id': '1e1b0263-7e49-4c00-985a-568927353700',
    'name': 'Near-Infrared Spectrograph',
    'short_name': 'JWST_NIRSPEC'}]}]
```

ACROSS: Demo Walkthrough

- ACROSS - Client
  - Instruments

```python
from across.client import Client

client = Client()

client.instrument.get_many(
    name="nircam",
)
```

✓ 0.0s


```
[{'id': '9899d36c-9e07-4927-8295-04c4ced70f1a',
  'created_on': '2025-07-22T23:21:40.539962',
  'name': 'Near-Infrared Camera',
  'short_name': 'JWST_NIRCAM',
  'telescope': {'id': '225ad468-585f-4f5e-8b64-09a4adfa1b7d',
   'name': 'James Webb Space Telescope',
   'short_name': 'JWST'},
  'footprints': [[{'x': -0.04306000000002541, 'y': -0.01833},
    {'x': -0.02528000000009295, 'y': -0.01833},
    {'x': -0.02528000000009295, 'y': -0.00055},
    {'x': -0.04306000000002541, 'y': -0.00055},
    {'x': -0.04306000000002541, 'y': -0.01833}],
   [{'x': 0.00611, 'y': 0.00055},
    {'x': 0.02389, 'y': 0.00055},
    {'x': 0.02389, 'y': 0.01833},
    {'x': 0.00611, 'y': 0.01833},
    {'x': 0.00611, 'y': 0.00055}],
```

ACROSS: Demo Walkthrough

- ACROSS - Client
  - Observations

```python
from across.client import Client

client = Client()

client.observation.get_many(
    date_range_begin="2025-08-09T00:00:00",
    date_range_end="2025-08-10T00:00:00",
    cone_search_ra=260.0,
    cone_search_dec=58.0,
    cone_search_radius=1
)
```

✓ 0.3s

```
[{'instrument_id': '9899d36c-9e07-4927-8295-04c4ced70f1a',
  'object_name': 'DRACO-F1',
  'pointing_position': {'ra': 260.19702, 'dec': 57.91138},
  'date_range': {'begin': '2025-08-09T17:10:16', 'end': '2025-08-09T17:36:24'},
  'external_observation_id': '9225:1:1',
  'type': 'imaging',
  'status': 'planned',
```

ACROSS: Demo Walkthrough

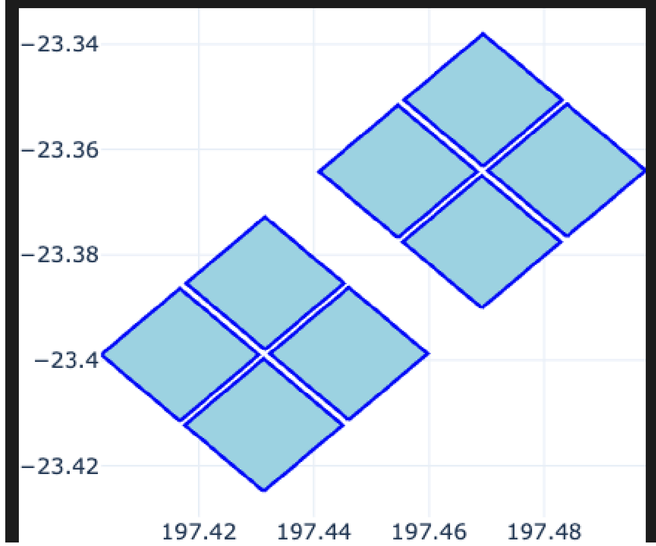
- ACROSS - Client + Tools
  - Client to gather instrument footprint
  - Tools to project the footprint

```
from across.client.footprint import api_footprint, plot_footprint

nircam = client.instrument.get_many(name="nircam")[0]
nircam_footprint = api_footprint(nircam["footprints"])

#project to a famous position...
nircam_projected = nircam_footprint.project(
    Coordinate(ra=197.450341598, dec=-23.3814675445),
    roll_angle=45
)
#plot the result
fig = plot_footprint(footprints=[nircam_projected])
fig.show(width=100, height=100)
```

✓ 0.1s



ACROSS: Demo Walkthrough

- ACROSS - Client + Tools
  - “Is my newly discovered transient in the current TESS FOV?”

```python
from across.client import Client

client = Client()

new_astronomical_transient = MyTransient(
    name="AT2026A",
    ra=150.456,
    dec=-30.345,
    discovery_date=datetime.now()
)
```

ACROSS: Demo Walkthrough

- ACROSS - Client + Tools
  - “Is my newly discovered transient in the current TESS FOV?”
    - Use the client to find where tess was looking at the discovery date

```python
from across.client import Client

client = Client()

new_astronomical_transient = MyTransient(
    name="AT2026A",
    ra=150.456,
    dec=-30.345,
    discovery_date=datetime.now()
)

tess_instrument = client.instrument.get_many(name="tess")[0]

tess_observations = client.observation.get_many(
    status=ObservationStatus.PLANNED,
    instrument_ids=[tess_instrument.id],
    date_range_begin=new_astronomical_transient.discovery_date,
    date_range_end=new_astronomical_transient.discovery_date
)
tess_observations
```

ACROSS: Demo Walkthrough

- ACROSS - Client + Tools
  - “Is my newly discovered transient in the current TESS FOV?”
    - Use the client to find where tess was looking at the discovery date
    - Use the tools repository to project the footprint, and plot it with the new transient source

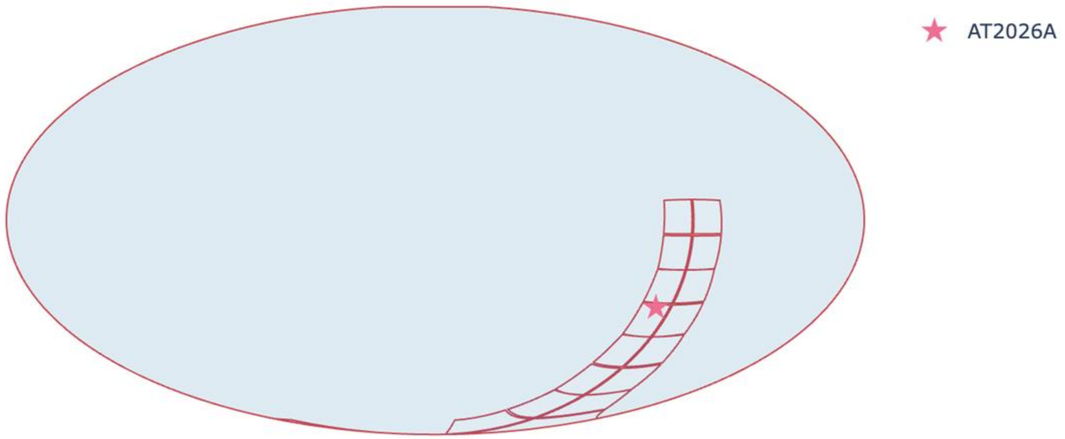
```
tools_footprint = api_footprint(tess_instrument.footprints)

projected_footprint = tools_footprint.project(
    Coordinate(ra=ra, dec=dec), roll_angle=0
)

fig = plot_footprint_with_source(
    projected_footprint, new_astronomical_transient
)

fig.show()
```

TESS Footprint



ACROSS: Demo Walkthrough

- ACROSS - Client + Tools
  - “Is my newly discovered transient in the current TESS FOV?”
    - Use the client to find where tess was looking at the discovery date
    - Use the tools repository to project the footprint, and plot it with the new transient source
    - Calculate follow-up visibilities with Swift

```python
swift = client.instrument.get_many(name="swift")[0]

visibility_windows = client.visibility_calculator.calculate_windows(
    instrument_id=swift.id,
    ra=new_astronomical_transient.ra,
    dec=new_astronomical_transient.dec,
    date_range_begin=new_astronomical_transient.discovery_date,
    date_range_end=new_astronomical_transient.discovery_date + timedelta(days=2),
)
print(visibility_windows.model_dump_json(indent=4))
```

✓ 15.7s


```
"instrument_id": "1c074192-f6d5-465d-844e-85a0010b2d87",
"visibility_windows": [
    {
        "window": {
            "begin": {
                "datetime": "2026-01-07T00:43:00",
                "constraint": "Window",
                "observatory_id": "d5faadca-c0bb-4bb8-b4e4-75149b435ae8"
            },
            "end": {
                "datetime": "2026-01-07T00:44:00",
```

Conclusions

ACROSS aims to make cross coordination of observations easier for NASA missions ***and beyond***, to enable TDAMM science.

While the TDAMM science cases are the drivers for this work, the benefits will apply to a much broader range of science cases

Any Questions?

samuel.d.wyatt@nasa.gov

ACROSS: Demo Walkthrough

- **OPEN SOURCE and Public (this is a big deal)**
  - https://github.com/NASA-ACROSS

ACROSS

NASA - Astrophysics Cross Observatory Science Support (ACROSS) Project

3 followers · United States of America · https://science.data.nasa.gov/data-...

Popular repositories

across-server — Public

Python · 1

across-tools — Public

A set of NASA multi-mission observing tools for situational awareness developed for the across-server, and other applications.

Python · 1

across-data-ingestion — Public

Ingestion container for observatory schedules

Python

across-server-openapi-python — Public

Auto generated OpenAPI spec client for python

Python

```
pip install across-client
pip install across-tools
```