IOOS GSoC Project Update Meeting #3

Friday August 28, 2026

Project #1: Enhancing noaa_coops

Project complete! Thanks Sreeya!

Enhancements:

- Added API support for missing products
- Added an entire new API layer
- Elegantly handle product length limits + chunking
- Consolidate + streamline various output types
- Test suite, docs updated

All PRs into dev repo complete

PR to merge into main (external) repo is [in progress]

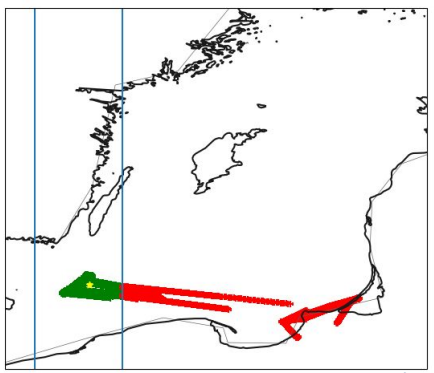
```
>>> from noaa_coops import Station
>>> seattle = Station(id="9447130")  # Seattle, WA
>>> df = seattle.get_data(
...     begin_date="20150101",
...     end_date="20150131",
...     product="water_level",
...     datum="MLLW",
...     units="metric",
...     time_zone="gmt",
... )
>>> df.head()
                       v      s       f  q
t
2015-01-01 00:00:00  1.799  0.023  0,0,0,0  v
2015-01-01 00:06:00  1.718  0.018  0,0,0,0  v
2015-01-01 00:12:00  1.639  0.013  0,0,0,0  v
2015-01-01 00:18:00  1.557  0.012  0,0,0,0  v
2015-01-01 00:24:00  1.473  0.014  0,0,0,0  v
```

Project #2: Filling the gaps for ocean glider data quality control

Stage 2: Completed the `required_tests`

- PRs for `location`, `range`, and `pressure`
  - Location test was expanded into additional tests: `location_on_land` and `location_bounds_test`
  - A statistical percentile test was added to compliment the range test
  - The pressure test was reworked to be more authentic to the manual
    - PR still pending! Some minor tweaks needed
- V3.0.0 published last week!
- Additional networking
  - Ocean Hack Week 2026
- Continuing after August
  - Stage 3 test assessments - especially the `climatology` and `density_inversion` tests
  - Build tools to help ID what good flagging parameters are for each type of test

Challenges: Changes to the API and packaging



Percentile-flagging real world "spoofing" lat/lon issue

OceanHackWeek 2026 in Victoria, Canada

Project #3: Integrating Empirical Dynamic Modeling (EDM) into FIMS

The project was developed across four main phases:

- **Delay Embedding:** build the time-series representation required by EDM.
- **Prediction Models:** implement Simplex, S-Map, and GP-EDM.
- **FIMS Integration:** connect EDM predictions with the FIMS/TMB framework.
- **R User Interface:** provide simple functions, evaluation tools, visualization, testing, and documentation.

All four phase integrations have been merged in 'main-edm' branch. Once the final changes are verified and approved by the mentors, the work will be integrated into the 'main' branch.

Architecture at a Glance:

- **C++ Core:** Delay embeddings, Simplex, S-Map, and GP-EDM functors with automatic differentiation.
- **TMB Integration:** Seamless objective function likelihood evaluation via EDMModel<Type>.
- **R User Layer:** High-level S4 API with automated parameter selection, multi-step forecasting, and diagnostics.

Project #3: Integrating Empirical Dynamic Modeling (EDM) into FIMS

R User Workflow: Parameter Selection & Delay Embedding

From raw time series to delay-reconstructed state space

R CODE:

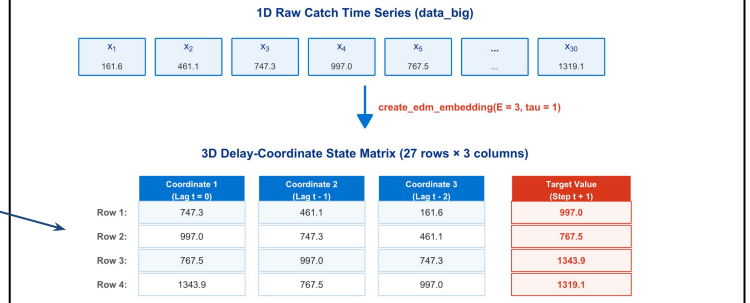
```
library(FIMS)

1. Load FIMS benchmark fishery data
ff <- FIMSFrame(data_big)

2. Automatically sweep and select optimal embedding dimension E (1
to 10)
edim <- select_embedding_dimension(ff, series_type = "catch",
series_name = "fleet1")
print(edim)

3. Construct Delay Embedding inside FIMSFrame
optimal_E <- edim$E[edim$optimal]  # E=1 but let's suppose E=3
ff <- create_edm_embedding(ff, "catch", "fleet1", E = optimal_E,
tau = 1L)
```

```
# A tibble: 10 × 4
      E   rho  rmse optimal
  <int> <dbl> <dbl> <lgl>
1     1     1     0 TRUE
2     2     1     0 FALSE
3     3     1     0 FALSE
4     4     1     0 FALSE
...
```



Project #3: Integrating Empirical Dynamic Modeling (EDM) into FIMS

Model Comparison, In-library diagnostic and forecasting

Evaluating Simplex, S-Map, and GP-EDM

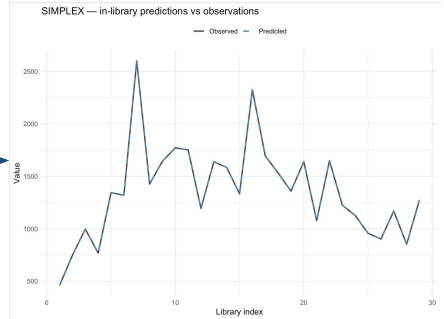
R CODE:

```
1. Fit and rank all EDM algorithms in single call
results <- edm_compare(ff)
print(results)

2. Fit optimal model & generate 2-panel diagnostic plot
fit <- fit_edm(ff, method = "simplex")
autoplot(fit, ff)

3. Generate 5-year ahead out-of-sample forecast
forecasts <- edm_forecast(fit, ff, n_ahead = 5L)
print(forecasts)
```

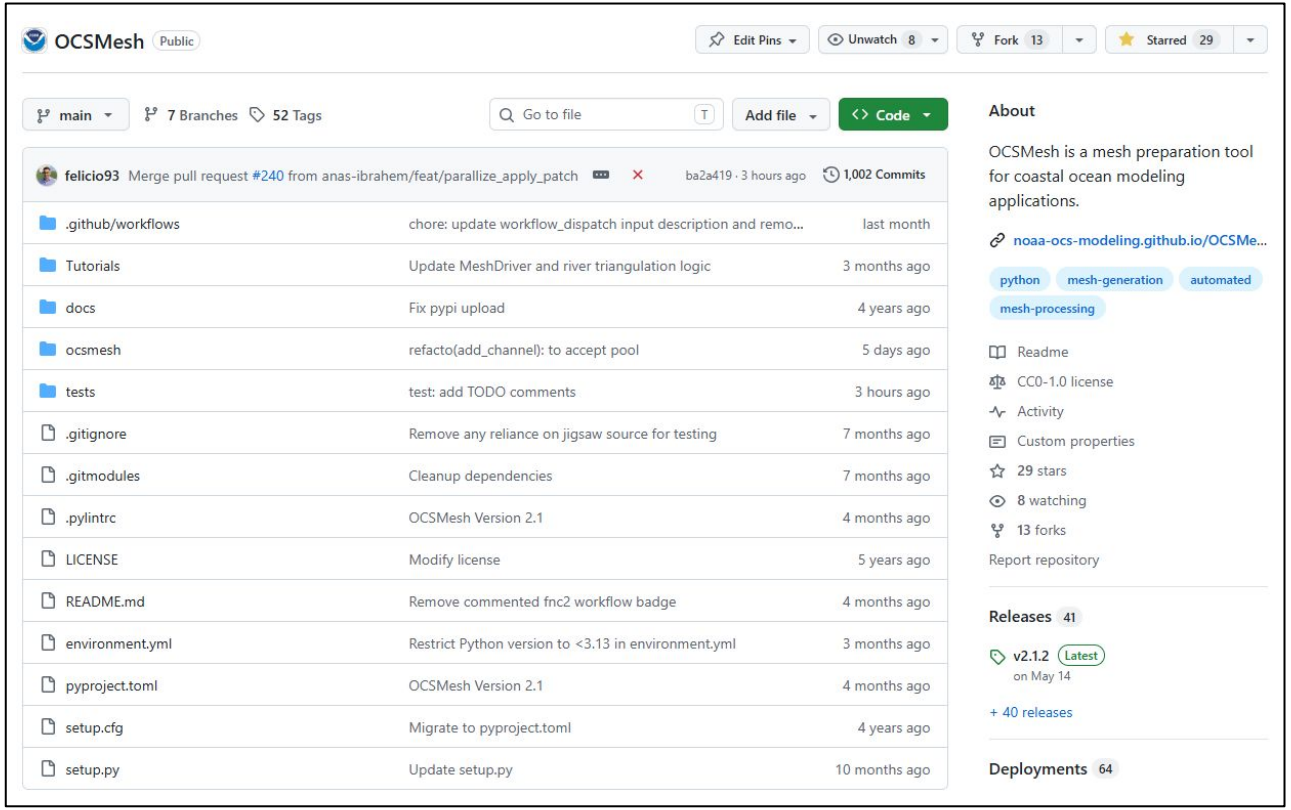
```
# A tibble: 3 × 7
    rho  rmse   mae     n method  embedding_name        best
  <dbl> <dbl> <dbl> <int> <chr>   <chr>                 <lgl>
1 1       0     0     29 simplex catch_fleet1_E1_tau1 TRUE
2 0.580 366.  276.    29 smap    catch_fleet1_E1_tau1 FALSE
3 0.563 378.  287.    29 gp      catch_fleet1_E1_tau1 FALSE
```




```
# A tibble: 5 × 4
   step  time forecast    se
  <int> <dbl>    <dbl> <dbl>
1     1    31    1274.    NA
2     2    32    1953.    NA
3     3    33    1953.    NA
4     4    34    1504.    NA
5     5    35    1504.    NA
```

211 / 211 Passing R Unit Tests via testthat across 4 test suites (100% pass rate).
21 Passing GoogleTests in C++ for core projection engines.
Fully rebased and aligned with FIMS v0.10.0 (Version 0.10.0)

Project #4: Scalable OCSMesh - Parallelization and Spatial Partitioning



Milestones

- Phase 1: Parallelization of Refinements (multiprocessing)

Benchmark Test Case	Raster Setup	Serial Time	Parallel Time	Speedup	Validation
Test Case 1	4 Rasters (20% Overlap)	16.14 s	10.02 s	**1.61x**	**PASS**
Test Case 2	4 Rasters (100% Overlap)	30.02 s	11.98 s	**2.51x**	**PASS**

Completed in July 1st

- Phase 2: MPI-based parallelization and testing:
  - Static vs. Dynamic parallelization.
  - Initial IOOS Sandbox testing
  - Scalable NOAA RDHPCS testing

Completed in Aug 15th

Phase 2 Findings:

Key measurements (15 CUDEM tiles + 1 GEBCO, single node, 80 cores)

Config A (fast refs only, no constraints):

Mode	Total (s)	Speedup vs serial_mp
mpi	597	**4.2×**
parallel	2,804	0.9×
serial_mp	2,478	1.0× (baseline)

Config B (2 fast refs on every tile):

Mode	Total (s)	Speedup vs serial_mp
mpi	1,031	**2.7×**
parallel	4,606	0.6×
serial_mp	2,735	1.0× (baseline)

- Parallel was slower than serial for small jobs:
  - We were previously spawning 79 workers for jobs that required only 3 (spawn overhead exceeded the parallelization benefits)

Measured improvement (Config A, 3 flow-limiter tasks):

Stage	Before (79 workers)	After (3 workers)
`_apply_flow_limiters_parallel`	339 s	96 s
`_apply_const_val_parallel`	324 s	87 s

- MPI is 4x faster than serial!
  - After the fix parallel is comparable to serial for small jobs.

Phase 2 Other Findings:

The cProfile confirms the constraint path cost breakdown

Function	cumtime
`_apply_features`	4,018 s
`_apply_constraints_parallel`	2,976 s
`add_feature` (box + patch)	~900 s
`_calculate_and_write_hfun_to_disk_mpi`	355 s
MPI dispatch	290 s

- The constraint stage (_apply_constraints) accounts for 66% of total MPI wall time:
  - This is the main bottleneck we have right now.
  - Implementing this will speed up the non topofunc contraints and box refinements.
- We have to properly parallelize the global functions (Contour/Channel):

cProfile breakdown (Config D / mpi, 27,014 s meshdata):

Stage	cumtime	% of total
`_apply_features` (serial, rank 0)	19,196 s	**71%**
├─ `_apply_contours`	8,025 s	30%
├─ `_apply_channels`	6,689 s	25%
├─ constraints + patch/feature	~4,480 s	17%
`_calculate_and_write_hfun_to_disk_mpi`	7,099 s	26%

Phase 3 - Sep 21st:

- After the testing phase we decided that the full following implementations are more beneficial than the spatial partitioning (in order):
  - add_contour (from global to tile-based using multiprocessing and MPI)
  - add_channels
  - add_patch
  - constraints

- Full validation
  - Real-world STOFS-Atlantic mesh using 300+ CUDEM tiles.
  - Profiling and mesh testing.

- **Beyond GSoC**

Project #5: Enhancing CrocoLakeTools with IOOS Data Sync from ERDDAP

CrocoLakeTools builds **CrocoLake**, a harmonised database of oceanographic observations. It converts raw data from various upstream providers into one common parquet schema, that allows researchers to work on a unified data standard.

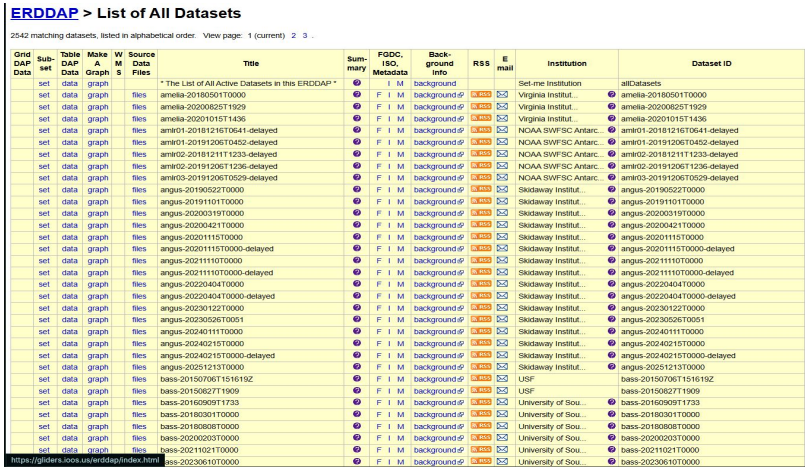
Limitation: Existing downloaders in crocolaketools maintain static links of sources for data ingestion. In IOOS GDAC's ERDDAP server there are thousands of dataset deployment that gets updated time to time, new datasets gets added. Maintaining a hardcoded list of datasource is an inefficient solution.

gliders.ioos.us/erddap
tabledap - parquet - ~2,000+ delayed-mode datasets

Layer 3 - Shared ERDDAP base
DownloaderERDDAP -> Downloader
Catalogue query, time-chunking, gated parallelism, retry and back-off, chunk merge, timestamp compare

Layer 2 - Dataset-specific downloaders
DownloaderIOOSGliders
Delayed-mode filter, curated variable list, per-dataset variable intersection, file naming

Layer 1 - CLI scripts
download_ioos_gliders.py , ioos_gliders2parquet.py
--sync --dry-run --threads --overwrite


ERDDAP > List of All Datasets

2542 matching datasets, listed in alphabetical order. View page: 1 (current) 2 3 .

Grid DAP Data	Sub-set	Table DAP Data	Make A Graph	WMS	Source Data Files	Title	Sum-mary	FGDC, ISO, Metadata	Back-ground info	RSS	E mail	Institution	Dataset ID
	set	data	graph			* The List of All Active Datasets in this ERDDAP *		I	M	background			Set-me Institution	allDatasets
	set	data	graph		files	amelia-20180501T0000		F	I	M	background	RSS	Virginia Institut...	amelia-20180501T0000
	set	data	graph		files	amelia-20200825T1929		F	I	M	background	RSS	Virginia Institut...	amelia-20200825T1929
	set	data	graph		files	amelia-20201015T1436		F	I	M	background	RSS	Virginia Institut...	amelia-20201015T1436
	set	data	graph		files	amlr01-20181216T0641-delayed		F	I	M	background	RSS	NOAA SWFSC Antarc...	amlr01-20181216T0641-delayed
	set	data	graph		files	amlr01-20191206T0452-delayed		F	I	M	background	RSS	NOAA SWFSC Antarc...	amlr01-20191206T0452-delayed
	set	data	graph		files	amlr02-20181211T1233-delayed		F	I	M	background	RSS	NOAA SWFSC Antarc...	amlr02-20181211T1233-delayed
	set	data	graph		files	amlr02-20191206T1236-delayed		F	I	M	background	RSS	NOAA SWFSC Antarc...	amlr02-20191206T1236-delayed
	set	data	graph		files	amlr03-20191206T0529-delayed		F	I	M	background	RSS	NOAA SWFSC Antarc...	amlr03-20191206T0529-delayed
	set	data	graph		files	angus-20190522T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20190522T0000
	set	data	graph		files	angus-20191101T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20191101T0000
	set	data	graph		files	angus-20200319T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20200319T0000
	set	data	graph		files	angus-20200421T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20200421T0000
	set	data	graph		files	angus-20201115T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20201115T0000
	set	data	graph		files	angus-20201115T0000-delayed		F	I	M	background	RSS	Skidaway Institut...	angus-20201115T0000-delayed
	set	data	graph		files	angus-20211110T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20211110T0000
	set	data	graph		files	angus-20211110T0000-delayed		F	I	M	background	RSS	Skidaway Institut...	angus-20211110T0000-delayed
	set	data	graph		files	angus-20220404T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20220404T0000
	set	data	graph		files	angus-20220404T0000-delayed		F	I	M	background	RSS	Skidaway Institut...	angus-20220404T0000-delayed
	set	data	graph		files	angus-20230122T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20230122T0000
	set	data	graph		files	angus-20230526T0051		F	I	M	background	RSS	Skidaway Institut...	angus-20230526T0051
	set	data	graph		files	angus-20240111T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20240111T0000
	set	data	graph		files	angus-20240215T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20240215T0000
	set	data	graph		files	angus-20240215T0000-delayed		F	I	M	background	RSS	Skidaway Institut...	angus-20240215T0000-delayed
	set	data	graph		files	angus-20251213T0000		F	I	M	background	RSS	Skidaway Institut...	angus-20251213T0000
	set	data	graph		files	bass-20150706T1516192		F	I	M	background	RSS	USF	bass-20150706T1516192
	set	data	graph		files	bass-20150827T1909		F	I	M	background	RSS	USF	bass-20150827T1909
	set	data	graph		files	bass-20160909T1733		F	I	M	background	RSS	University of Sou...	bass-20160909T1733
	set	data	graph		files	bass-20180301T0000		F	I	M	background	RSS	University of Sou...	bass-20180301T0000
	set	data	graph		files	bass-20180808T0000		F	I	M	background	RSS	University of Sou...	bass-20180808T0000
	set	data	graph		files	bass-20200203T0000		F	I	M	background	RSS	University of Sou...	bass-20200203T0000
	set	data	graph		files	bass-20211021T0000		F	I	M	background	RSS	University of Sou...	bass-20211021T0000
	set	data	graph		files	bass-20230610T0000		F	I	M	background	RSS	University of Sou...	bass-20230610T0000

https://gliders.ioos.us/erddap/index.html

Status at submission

- #48 Downloader refactor **Merged**
- #53 + loader #10 **Merged**
- #65 ConverterIOOSGliders **In review**
- #50 ERDDAP base + sync **Merged**
- #59 ERDDAP constraints **Merged**
- erddapy #467 parquet **resolved by Filipe**

Engineering against a live, shared, memory-constrained server (project #5)

413 ERDDAP caps response size

From my experiments, whenever the response payload size got larger than 35mb, the erddap server raises http 413 error and download operation fails.

Time-windowed chunking with adaptive step-down

Split the dataset time range into 24 h windows, fetch each, merge locally into one parquet.
Still 413? Retry the dataset at 24h -> 12h -> 6h, then log and move on.

503 Naive parallelism took the server down

Downloading multiple files directly without any bound caused server downtime.
Reason: ERDDAP builds the entire response in memory before sending a byte. So simultaneous request from multiple threads causes heavy memory footprint.

First Byte Gate: ERDDAP build phase is serialized by controlling the parallelism during client side requests. Receiving first download byte on a thread means build phase is finished for that thread's request.

404 Default behavior?

Every chunk represents a time window, some time window can simply be empty hence ignored without raising an exception.

Each status mapped to desired fallback

404 = empty window: counted and reported, never an abort. 503 / 429 / 500 -> back off 15s, 30s, 60s.
Chunks stream to .tmp, then os.replace(): a failed chunk abandons the merge entirely.

? 3,500 unique variables across the catalogue

One target variable can have multiple name/alias. Creating a curated mapping of such alias was necessary.

Curated list + per-dataset intersection

~130 source names scraped by frequency, mapped by hand to CrocoLake conventions (PHY, BGC, _QC).
Before each URL, ask /info what the dataset has and request only the overlap.

config-driven constraints, and the converter (project #5)

PR #59: Config-driven ERDDAP constraints MERGED 25 AUG

All seven tabledap operators **=** **!=** **=~** **<** **<=** **>** **>=** handled through config.yaml.

```
constraints:
  time>=:      "2015-01-01T00:00:00Z"
  time<=:      "2026-01-01T00:00:00Z"
  latitude>=:  -90.0
  latitude<=:   90.0
  longitude>=: -180.0
  longitude<=:  180.0
  # users can add other PHY/BGC variable constraints.
```

use case:
A user may want their data to be associated with a specific location, zone or time. It is also possible that, someone may be concerned with variables within a specific measurement.

Also merged: PR #53 + crocolakeloader #10 removed the git-submodule dependency. Shared db_names / db_params now live inside CrocoLakeTools, so a contributor no longer has to clone a second repo to run the converters.

PR #65: Converter for IOOS Gliders IN REVIEW

Unplanned: core project was covered early so we decided to invest remaining time for writing a converter.

Raw parquet · ERDDAP source names

↓

read_to_ddf()
one dask task per parquet row group

↓

process_df() · 250k-row sub-chunks
BGC schema is ~120 columns wide, a full row group blows the dask worker limit

↓

Normalise and standardise
tz-aware -> tz-naive UTC, PRES from depth via gsw.p_from_z() (TEOS-10), merge_alternate_columns() folds chlorophyll / fluorescence / CPHL into CHLA, rename to CrocoLake schema, DOXY µmol/L -> µmol/kg, drop all-NA rows

↓

IOOS_GLIDERS_PHY / IOOS_GLIDERS_BGC parquet
merge into crocolake

Stretch goals, open threads, and what happens next (project #5)

Animal Telemetry Network downloader ON HOLD

DownloaderATN
Layer 2: a subclass, delayed-mode filter + variable map only

↓

DownloaderERDDAP
Layer 3: chunking, gating, retry, sync: all inherited, nothing to rewrite

Blocker: I scraped every variable on atn.ioos.us/erddap and went through them with Enrico. None of them map onto CrocoLake's PHY/BGC measurements of interest.

Raised at the 24 July IOOS meeting
It was acknowledged and was expected that the variables of interest to be published soon. But may not be within the GSoC window. As of today the ATN server still doesn't carry them.

db_names.py sync CI · #54 ON HOLD

A GitHub Actions workflow that detects drift between the two repos' copies of db_names.py was written but it needed organisation-level permissions across crocolaketools and crocolakeloader repo to run.

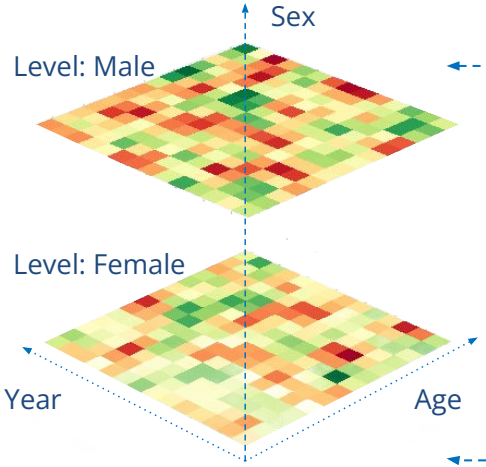
Likely Overridden
Nothing but CrocoLakeTools consumes CrocoLakeLoader. Integrating it in CrocolakeTools can simply solve the problem entirely.

What I'm doing after GSoC

- Take ConverterIOOSGliders (#65) through review to merge, with the optimizations Enrico suggested.
- Providing support if needed to settle CrocoLakeLoader dependency in CrocolakeTools by enhancing CI pipeline, or integrate the package.
- May work on the ATN downloader if IOOS publishes the PHY/BGC variable within a reasonable time scope.
- Keep maintaining and extending the ERDDAP layer as new sources are added. Helping in documentation of known ERDDAP issue.

Thanks to Enrico Milanese and IOOS.

Project #6: Add sex structure to FIMS statistical-catch-at-age model



NOAA-FIMS/FIMS on Jun 26

Add partition spec and index layout in-frastructure for sex strata #1581

Open **Approved**

Open **Introduce the explicit two-sex model for the FIMS catch-at-age population dynamics** #1625

Parent: #1582

Feat/explicit two sex ×

#3 opened 4 days ago by szu-yun-ko Owner • Draft 8 of 15 tasks

In Progress

Model Type

Specified

Expose partition demand to R and gate partitioned fleet DQs ×

#2 opened 2 weeks ago by szu-yun-ko Owner

Ready for Review

Demand User

Empty

Default, pooled behavior (current)

Project #6: Add sex structure to FIMS statistical-catch-at-age model

User-supplied, initial sex-ratio

Total mortality is the sum of natural (M) and fishing (F) mortality

$$N^{\text{fem}}_{a,y} = p^{\text{fem}}_{a-1,y-1} N_{a-1,y-1} e^{-Z^{\text{fem}}_{a-1,y-1}}$$

$$N^{\text{male}}_{a,y} = \left(1 - p^{\text{fem}}_{a-1,y-1}\right) N_{a-1,y-1} e^{-Z^{\text{male}}_{a-1,y-1}}$$

$$N_{a,y} = N^{\text{fem}}_{a,y} + N^{\text{male}}_{a,y}$$

$$p^{\text{fem}}_{a,y} = \frac{N^{\text{fem}}_{a,y}}{N_{a,y}}$$

$$Z^{fem}_{a,y} = M^{fem}_{a,y} + F^{fem}_{a,y}$$

$$F_{f,a,y} = F_{f,y} f_y S_f(a)$$

Fishing mortality is determined by selectivity (S), which can be sex-specific

Abundance (age, year) then becomes sex-specific

Dynamically update p_fem at the next step using female abundance

Feat/explicit two sex ×

#3 opened 4 days ago by szu-yun-ko Owner · Draft 8 of 15 tasks

Project #7: Enhance NOS skill assessment package's user and developer experience

Phase 1 - The User Experience

Status: Completed, Merged
Before (Issues the package had earlier)

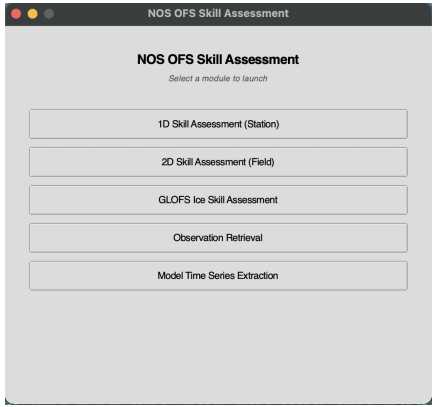
- Only one GUI, wrapping the 1D pipeline
- Windows-only Tk theme would not launch on macOS/Linux
- No tooltips, no validation, no way to run other pipelines from the UI

My work -

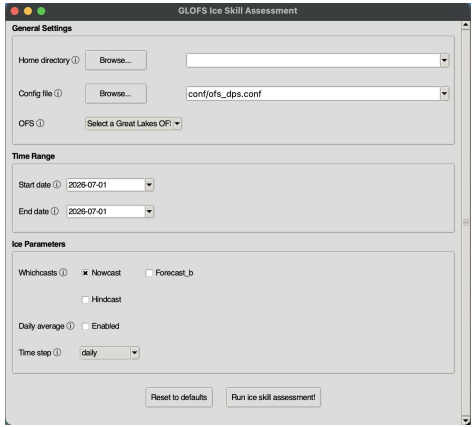
- 4 phased PRs: #164, #174, #178, #187
- 5 tools + 1 launcher: 1D, 2D, GLOFS Ice, Obs retrieval, Model time series extraction
- Shared UX: tooltips, validation, confirm dialog, progress overlay, session memory
- Cross-platform: macOS, Windows, Linux (ttk, calendar fix, mouse-wheel scroll)
- Same code path as CLI, the GUI calls main, no duplicated pipeline logic
- Documented in project wiki - Link

Current GUI Suite -

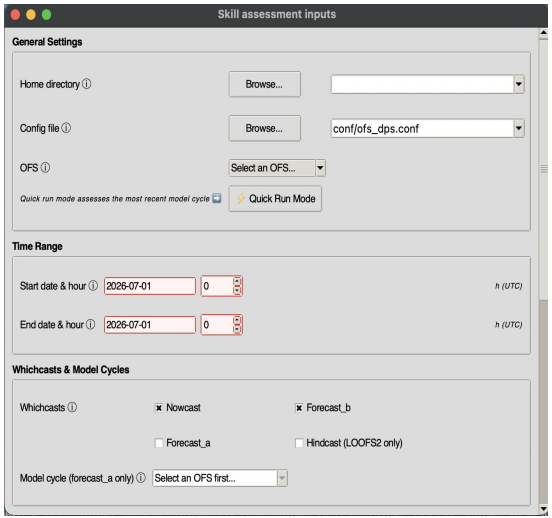
The Launcher -

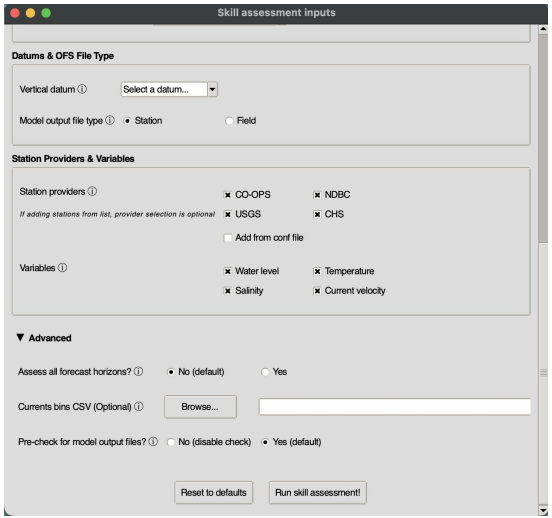


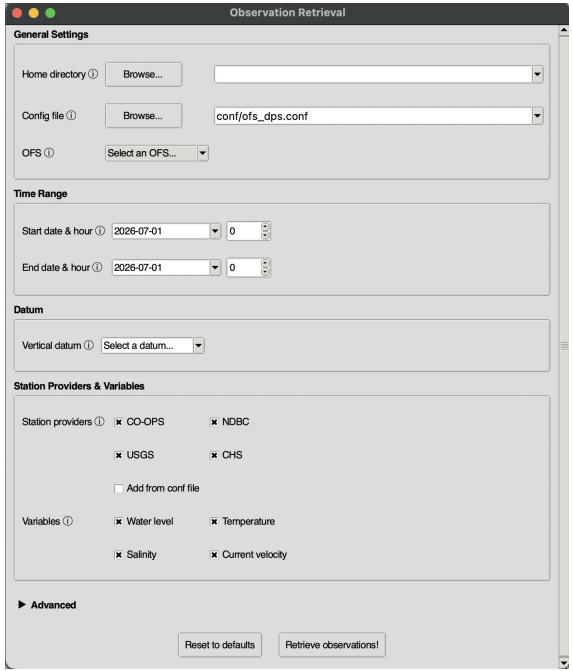
GLOFS Ice Skill GUI -



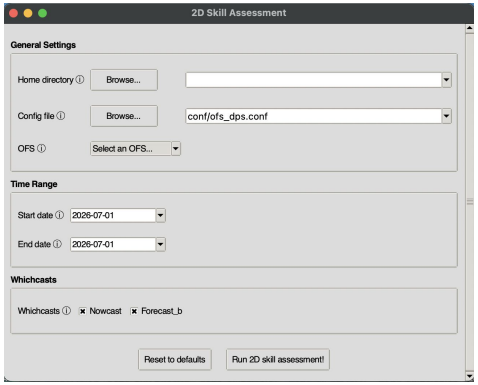
1D Skill GUI -



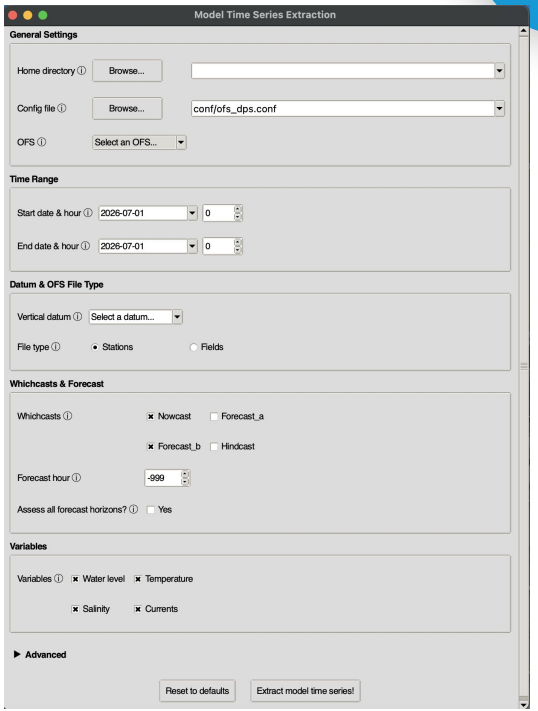




OBS Retrieval GUI -



2D Skill GUI -



Timeseries Extraction GUI -

Phase 2 - The Developer Experience

Status: Completed, Merged
Before (Issues the package had earlier)

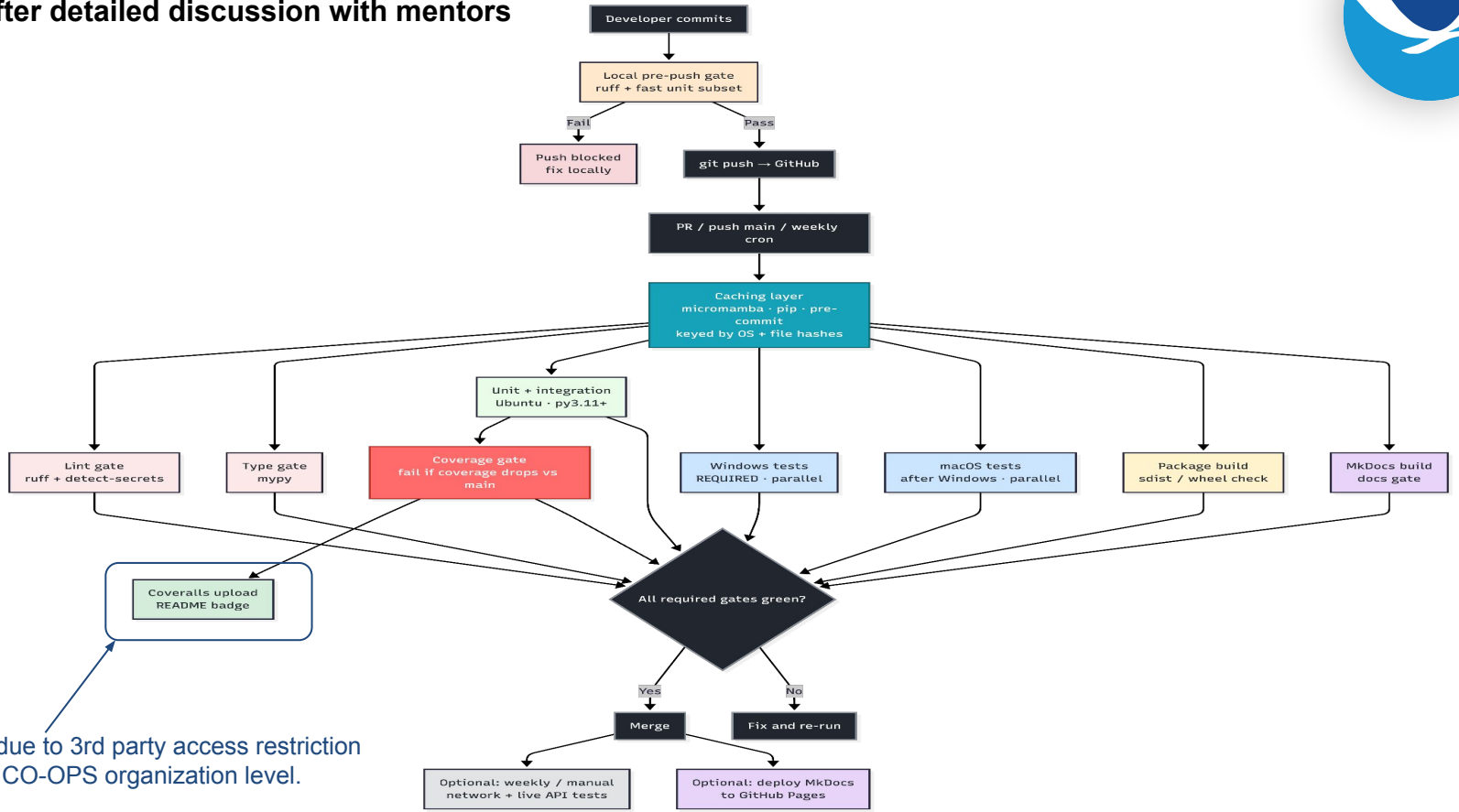
- Single Ubuntu CI job, no lint/types/docs gates
- No coverage floor, no Windows/macOS testing in CI
- Live API tests mixed with PR checks; no developer docs beside the wiki

My work -

- Parallel gates: lint, types check, docs, package checks, tests and Build Cache usage
- Cross-platform matrix: Ubuntu / Windows / macOS × Python 3.11 / 3.12
- Quality tools: ruff, detect-secrets, mypy, mkdocs --strict, build + twine
- Coverage floor (fail_under 39% will get an increment whenever testing level grows)
- Mock APIs usage to avoid API retrieval failures
- Weekly live network tests (USGS) scheduled, not blocking PRs
- Local parity: make ci-local + pre push hook
- MkDocs site under docs/ with auto-generated API reference (mkdocstrings)
- Strict build wired into CI; GitHub Pages deploy
- Wiki = user docs · MkDocs = developer docs

CI Flowchart -

Made after detailed discussion with mentors



Dropped due to 3rd party access restriction set at the CO-OPS organization level.

GSoC 2026 Wrap Up

Final code submissions (for many) due next week. Final evaluations due the following week.

GSoC Project Showcase webinar?

Blog post or other summary of GSoC experience, in addition to a public final submission for GSoC requirements, is highly encouraged! See some examples:

- https://github.com/ioos/gsoc/blob/main/2024/project-results.md
- https://github.com/ioos/gsoc/blob/main/2025/project-results.md