Open Access | https://doi.org/10.25080/vxkf4244

Ocetrac: An Object-based Framework for Tracking and Quantifying Climate Events in Gridded Datasets

Cassia Cai[1], **Jacob T. Cohen**[1], **Hillary Scannell**[2], **Valentina Staneva**[3], and **LuAnne Thompson**[1]

[1]School of Oceanography, University of Washington, [2]Jupiter Intelligence, [3]eScience Institute, University of Washington

SciPy 2025

July 7 - July 13, 2025

Proceedings of the 24th Python in Science Conference

ISSN: 2575-9752

Abstract

Climate extremes such as marine heatwaves and ocean acidification events appear as dynamic, irregular structures in gridded datasets. Software tools are available for detecting these events. However, there is a need for standardized methods for tracking their spatiotemporal evolution and evolving characteristics. Ocetrac is an open-source Python package that bridges this gap by integrating morphological image processing (using `scikit-image`) with `xarray`-based data workflows to track and quantify amorphous two-dimensional features in space and time. The tracking algorithm identifies and tracks objects that split and merge. The Measures submodule computes shape-, motion-, temporal-, and intensity-based measures. These measures enable comparison between similar events that occur at different times (e.g., via clustering), machine learning integration such as feature extraction for classification/ regression (e.g., `scikit-learn`), and process-oriented analysis such as linking event behavior to physical drivers (e.g., surface heat fluxes). Designed for the Scientific Python ecosystem, Ocetrac simplifies workflows from detection to downstream analysis, supporting applications in climate science, oceanography, and atmospheric research.

Keywords spatiotemporal tracking, morphological operations, image processing, scientific Python, climate extremes

1. Introduction

Gridded climate datasets are important for studying phenomena like marine heatwaves (MHWs), deoxygenation zones, and atmospheric blocking events among others. These events exhibit complex spatiotemporal behavior including splitting, merging, and complex evolution. The ecological impact of these events is closely tied to their spatial and temporal characteristics. For example, in the case of MHWs, long-lived and spatially extensive events can have significant ecological impacts [1]. These impacts include habitat destruction [2] and shifts in marine species distributions [3]. Understanding these events requires not only detecting their occurrence but also tracking their evolution and quantifying their characteristics over time.

Conventional analysis approaches, including gridpoint-based statistics, fixed-region summaries, and empirical orthogonal functions (EOFs), offer valuable insight in explaining spatial patterns but fall short in characterizing events as coherent and evolving structures. Gridpoint statistical analysis obscures spatial connectivity while EOFs, which is a dimensionality reduction technique that decomposes spatiotemporal fields into orthogonal spatial patterns and their time-varying amplitudes, lack object-level granularity. Object tracking is a different approach and has been applied to both atmospheric and oceanic phenomena.

Published Jul 10, 2025

Correspondence to
Cassia Cai
fmc2855@uw.edu

Object-based approaches treat climate features as coherent objects in space (x, y) and time (t), thus enabling direct evaluation of properties like shape, connectivity, and persistence. For example, D. J. Gagne, A. McGovern, S. E. Haupt, R. A. Sobash, J. K. Williams, and M. Xue [4] used an enhanced watershed method to identify hailstorm objects in radar data. Similarly, `TempestExtremes` described in P. A. Ullrich, C. M. Zarzycki, E. E. McClenny, M. C. Pinheiro, A. M. Stansfield, and K. A. Reed [5] applies a MapReduce-style algorithm to detect and track compact, threshold-defined features such as tropical and extratropical cyclones. While effective for well-defined structures, such threshold-based methods can struggle to capture the extent and temporal evolution of more diffuse or amorphous events.

Object-based detection and tracking of spatiotemporal climate features is a growing area of research. Other climate features, such as tropical cyclones and atmospheric rivers, have benefitted from tracking algorithms [5]. Recent studies have used a variety of techniques to identify MHW events, each with its own benefits and limitations. All techniques track objects using a form of connected component analysis: an MHW grid cell is connected with adjacent MHW grid cells in space and time. Algorithms differ in how they assign connectivity, though. Some studies, such as G. Bonino, S. Masina, G. Galimberti, and M. Moretti [6] and R. I. Woolway, E. J. Anderson, and C. Albergel [7], connect grid cells only if they are directly adjacent to each other. D. Sun, Z. Jing, F. Li, and L. Wu [8] introduced a more flexible approach, using a Nearest Neighbor algorithm to connect nearby grid cells that are not necessarily adjacent to one another. Additional constraints exist in deciding what constitutes overlap for temporal tracking. D. Sun, F. Li, Z. Jing, and et al. [9] extended their tracking methodology to analyze MHW volumes. The TOOCAN algorithm [10], designed for organized deep convective systems, also uses a connected components approach with adjustable spatiotemporal constraints. Ocetrac, like D. Sun, Z. Jing, F. Li, and L. Wu [8], addresses spatial coherence through smoothing (via morphological operations rather than K-Nearest Neighbor filtering). D. Sun, Z. Jing, F. Li, and L. Wu [8] uses a temporal overlap parameter to determine whether two objects are the same event while Ocetrac connects objects that overlap by at least one grid cell in space and time.

Generally, existing tools tend to fall into two categories: specialized trackers designed for specific phenomena (e.g., tropical cyclones) that use rigid assumptions, or pixel-level anomaly detectors that lack spatial integration. Specialized trackers are often tailored to particular variables or regions, while pixel-level anomaly detectors can produce disjointed results that require manual post processing to reconstruct event trajectories. Moreover, many studies rely on custom scripts that are often not well-documented, making comparisons and reproducibility challenging. Despite their advantages, object-based techniques remain underutilized in many geophysical applications due to gaps in software accessibility and generality.

This paper describes Ocetrac, an open-source Python package designed for detecting, tracking, and quantifying irregular, evolving features in gridded climate data. Ocetrac combines morphological image processing using `scikit-image` [11] with labelled array workflows using `xarray` [12] to characterize irregular structures. Although Ocetrac uses an initial threshold to identify candidate regions, Ocetrac moves beyond a simple binary definition by using morphological operations to filter and refine these regions based on their spatial coherence and connectivity. The Measures submodule provides functionality to quantify spatial and temporal properties of the tracked events, and thus facilitates further statistical analysis, such as aggregation analysis of multiple events within large datasets. Its modular design supports:

1. Detection: Ocetrac smooths spatially connected regions (features) in a binary field, such as sea surface temperature (SST) anomalies, to identify objects that exceed a user-defined threshold (e.g., the 90th percentile of anomalies)

2. Tracking: Ocetrac labels objects across time steps, even when they split or merge
3. Quantification: Ocetrac computes shape, motion, and intensity measures
4. Integration: Ocetrac is compatible with the Scientific Python ecosystem, including `dask` [13], `xarray` [12] and `scikit-learn` [14]

Although originally developed for MHW analysis, Ocetrac can be used for other applications, from atmospheric rivers to hypoxic zones and bloom patches in the ocean. Its modular design allows users to customize detection thresholds, tracking logic, and measures of tracked objects. This paper details its design, demonstrates several use cases, and outlines future directions.

2. Goals and Motivations

Climate phenomena like MHWs are dynamic events that can propagate across basins, changing their size, shape, and intensity as they evolve over time. Traditional gridpoint or static regional analyses that evaluate variables at individual locations or within a fixed region may overlook the structure and the evolution of such events. Object-based methods address this limitation by treating features as one event. This approach shifts the focus from isolated grid cells to the geometry, movement, and intensity evolution of entire events.

Building on the need for accessible and general object-based tracking, Ocetrac, unlike phenomenon-specific tools, operates on intensity data that is thresholded (e.g., SST anomalies exceeding the 90th percentile), applying morphological operations to group connected regions into labeled objects before tracking their evolution. The design prioritizes interoperability with standard climate data formats (CF-compliant files following Climate and Forecast metadata conventions) and is compatible with observational products, reanalyses, and model output. For MHW-specific applications, Ocetrac is compatible with output from packages such as marineHeatWaves and xmhw, which allows for existing anomaly detection workflows to be combined with Ocetrac's object-based tracking. Because the tracking logic is decoupled from domain-specific assumptions, Ocetrac is a practical and adaptable tool for tracking climate features across datasets and can be used to answer diverse research questions. The Measures submodule processes tracked objects to generate quantitative diagnostics, which allows for standardized comparison across events and datasets.

3. Ocetrac: Tracking and Measures Submodules

Ocetrac is a flexible tracking framework designed for geophysical feature analysis. While we demonstrate its application using SST anomalies to identify MHWs, the algorithm itself is variable-agnostic. Ocetrac can be applied to any two-dimensional spatiotemporal field that can be thresholded into spatially coherent features and has sufficient temporal resolution to resolve event evolution. The tracking methodology is described briefly below and in detail in H. A. Scannell, C. Cai, L. Thompson, D. B. Whitt, D. J. Gagne, and R. P. Abernathey [15] where it was used to examine MHWs throughout the globe. To support event characterization beyond simple tracking, we introduce the Measures submodule, which enables the calculation of a suite of geometric, intensity-based, and motion-related diagnostics for each tracked event. The Ocetrac workflow is demonstrated below in Figure 1. The tutorial associated with this workflow is available here.

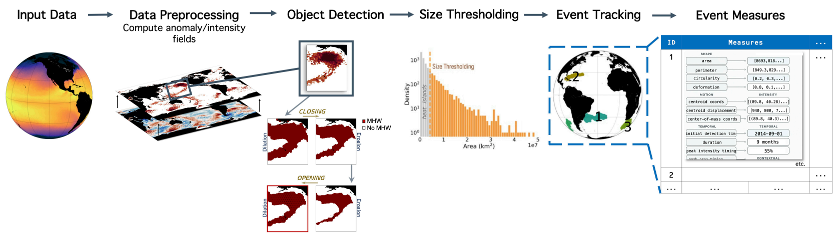


Figure 1. *Ocetrac workflow diagram*

3.1. *Input Specifications and Preprocessing*

Ocetrac requires input data as a 3D `xarray.DataArray` with dimensions ordered as (x, y, t), where (x, y) correspond to regular spatial coordinates, typically latitude and longitude, and t corresponds to time. The time dimension should be uniformly spaced to ensure optimal tracking performance of features through the temporal sequence. The time dimension should also have high enough spatial and temporal resolution to capture the temporal evolution of the target feature. Temporal gaps in the data can be filled using linear temporal interpolation to maintain continuity in the time series.

An optional binary land mask (1 for valid grid cells, 0 for excluded regions such as land or sea ice) can be provided to omit specific areas from detection and tracking. All preprocessing, such as detrending and anomaly calculation, thresholding, must be performed before using Ocetrac. Common thresholding approaches include: percentile-based (e.g., values exceeding the 90th percentile of anomalies), absolute value thresholds (e.g., values exceeding 28°C), or statistical significance thresholds (e.g., values exceeding two standard deviations from the mean). Ocetrac is agnostic to the thresholding method, as long as the input is a binary spatiotemporal field.

3.2. *Tracking Submodule*

The tracking process begins with converting anomalies into binary maps using a threshold (typically the 90th percentile for MHWs). Grid cells exceeding the threshold are marked as features where active grid cells are assigned the value of 1 and all others are inactive and assigned the value of 0.

To refine features, Ocetrac applies sequential morphological operations:

1. Closing (dilation followed by erosion): Fills small holes within features and connects nearby regions belonging to the same object. This operation helps maintain spatial coherence by ensuring that nearby features that should logically be part of the same event are connected.
2. Opening (erosion followed by dilation): Removes isolated pixels, smooths feature boundaries, and cleans residual artifacts introduced by closing. This operation helps eliminate noise and small-scale features that may not be physically meaningful.

The choice of these morphological operations follows image processing practices for feature enhancement and noise reduction. The closing operation is applied first to connect nearby features, followed by opening to remove small artifacts and refine the boundaries of the features. This sequence is important for maintaining the integrity of the detected features while minimizing fragmentation. The morphological operations are implemented using `scikit-image` [11] and are designed to be computationally efficient, allowing for processing large datasets in parallel using `dask` [13]. While other sequences of operations (e.g., opening followed by closing) can be implemented, empirically, this sequence (closing followed by opening) optimizes feature integrity while minimizing fragmentation. The intermediate results of morphological operations are shown and described in greater detail in H. A. Scannell, C. Cai, L. Thompson, D. B. Whitt, D. J. Gagne, and R. P. Abernathey [15].

The operations are performed using a structuring element, which defines the neighborhood around each pixel used to determine its new value. Ocetrac uses a circular structuring element, which is a disk-shaped kernel that treats all directions equally. The size of this disk is defined by its radius R (in number of grid cells). The radius is a parameter that controls the spatial scale of the filtering: a larger radius results in more aggressive smoothing and merging of features, while a smaller radius preserves finer details but may retain more noise.

For 0.25° resolution data:

- R = 4 to 6 grid cells (1 - 1.5°): Preserves smaller-scale features while removing noise
- R = 6 to 8 grid cells (1.5 to 2°): Emphasizes larger, more coherent structures
- R > 8 grid cells: May merge distinct features or even fail to identify features as valid objects altogether

For data at a different resolution, the radius must be scaled to maintain a consistent physical filtering scale. For example, for higher-resolution 0.125° data, the grid cell size is halved. To achieve an equivalent physical filter size, R should be doubled.

The choice of R should be informed by the typical spatial scale of the features of interest and the resolution of the input data and represents a trade-off between feature completeness (higher R retains more connected areas) and spatial precision (lower R preserves finer-scale details) (Figure 2). The optimal radius depends on the spatial resolution of the input data and the typical size of the features being tracked. Users should experiment with different radius values to find the best balance between feature retention and noise reduction for their specific application. Users should also validate this parameter against known feature scales in their domain. Features, after undergoing morphological operations, are now referred to as objects.

Ocetrac then applies size-based filtering to focus on the most spatially coherent objects. It calculates the area of each detected object in grid cells, then removes objects smaller than a user-defined area threshold P. By default, it keeps objects larger than 75% of all detected features (the 75th percentile) for the full dataset, meaning the smallest 25% of all objects are filtered out. This filtering step helps eliminate noise and isolate the most physically meaningful objects in the dataset.

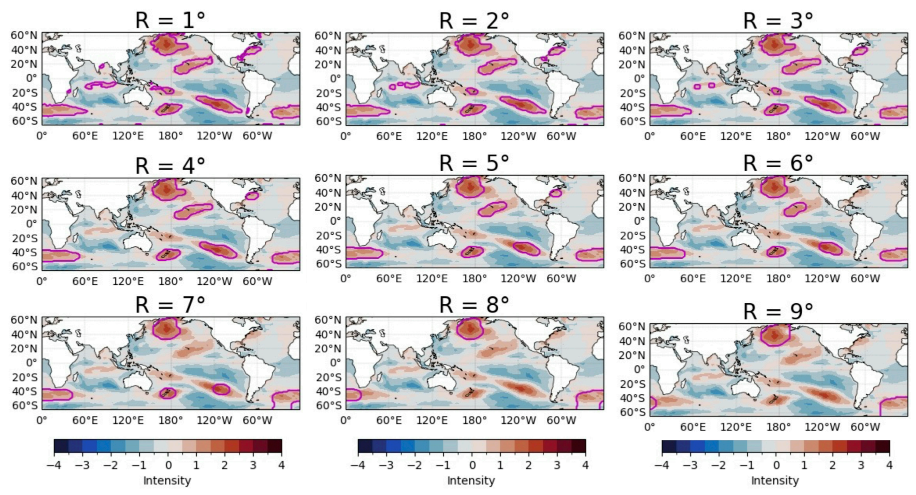


Figure 2. *Objects outlined (purple outlines) detected using varying R values (R ranging from 1° to 7°). The background shows SST anomalies (°C) from February 2007 of an ensemble member of the CESM2-LENS dataset. The horizontal resolution is 1°.*

Ocetrac identifies objects in each timestep by grouping together adjacent pixels, including diagonal connections (8-connectivity). This is a permissive approach for the initial detection phase to ensure that complex, irregularly shaped features are identified as single, coherent objects rather than being artifically fragmented into multiple parts. Then, to link objects across consecutive timesteps, Ocetrac uses a more conservative 6-connectivity approach where it matches objects that appear in either the exact same grid cell or directly adjacent cells (up/down/left/right) in the next timestep. This conservative approach allows objects to move gradually and prevents discontinuous jumps. The system maintains consistent ID numbers for objects as they evolve and handles global datasets where the map wraps around the globe at 0°/360° longitude, and tracks when objects merge together or split apart. The only requirement is that objects must overlap by at least one grid cell from one time step to the next to be in the same event.

The current implementation of Ocetrac is designed for rectilinear (latitude-longitude) grids. A limitation of this framework is its handling of high-latitude regions. At high latitudes, the convergence of meridians leads to distortions in area representation, which can affect the accuracy of object detection and tracking. Future versions of Ocetrac will address this limitation by incorporating grid-aware algorithms that adjust for varying grid cell sizes and shapes at different latitudes.

Implementation example

```python
# Initialize tracker with user-defined parameter
obj_Tracker = ocetrac.Tracker(
    binary_out_afterlandmask,   # Thresholded binary field
    newmask,                    # Land/ice mask (optional)
    radius=3,                   # R in grid cells
    min_size_quartile= 0.75,    # P (determines minimum object size)
    timedim = 'time',           # Time dimension name
    xdim = 'lon',               # Longitude dimension name
    ydim='lat',                 # Latitude dimension name
    positive=True)              # Track positive anomalies

# Execute tracking and morphological operations
blobs = obj_Tracker.track()

mo = obj_Tracker._morphological_operations()
```

Here Ocetrac provides basic diagnostics (e.g., initial/final object counts, area retention percentage) to evaluate tracking performance and feature characteristics.

3.3. Measures Submodule

The output of the tracker is a `xarray.Dataset` with many labelled events. The Measures submodule allows users to process large numbers of tracked events by automating repetitive calculations. Its standardized outputs allow for direct comparison across different events and datasets, which streamlines analysis workflows. The submodule supports both individual case studies and ensemble analyses. In brief, the Measures submodule processes the output of Ocetrac's tracking algorithm and provides quantitative metrics to characterize detected events. These measures fall into five categories: shape, motion, intensity, and temporal, and contextual and are summarized in Table 1.

Shape measures characterize geometric properties including area, perimeter, and circularity that are relevant for quantifying structural evolution. Shape measures can be used for understanding ecological impacts, for example, the area/extent can be used for assessing potential habitat affected, while the deformation quantifies shape changes between timesteps. These measures can help answer questions such as (1) How does event shape affect local ecosystems? and (2) Do certain shapes persist or evolve predictably? Motion measures track movement and displacement patterns via centroids and intensity-weighted centers of mass, while also handling longitudinal wrapping. Motion measures can help identify propagation directions and whether events remain stationary. Intensity measures capture magnitude variations through spatial statistics (mean, maximum, percentile extremes) of the underlying anomaly field. Intensity measures can be used for assessing thermal stress on ecosystems. Temporal measures record lifecycle characteristics including duration, initiation timing, and timing peaks of intensity and area. Temporal measures can determine event development stages, seasonal dependencies, and potential climate connections. Some of the measures can also be used to create contextual measures (e.g., object counts per timestep) which capture splitting and merging objects. Together, these measures enable analysis of event dynamics.

Table 1. *Event measures in the Measures submodule*

Category	**Measure**	**Definition**	**Interpretation**
SHAPE	Area (Extent)	Spatial coverage in km² or grid cells. Calculated as the sum of cell areas within bounds.	Larger values = greater spatial influence
	Perimeter	Boundary length (km), computed via geodesic contours. Calculated as Haversine sum of contour points.	Complex shapes yield high perimeters.
	Circularity	Deviations from a perfect circle. Calculated as $4\pi \frac{Area}{Perimeter^2}$.	1 = perfect circle; 0 = highly irregular
	Deformation	Shape stability between timesteps. Calculated as $1 - \frac{Shared\ Area}{Total\ Area}$.	0 = no change; 1 = complete deformation
	Convex Hull Area	Object area relative to its convex hull area. Calculated as $\frac{Area}{Convex\ Hull\ Area}$.	Lower values = more concave/irregular
MOTION	Centroids per timestep	Geometric centers of all objects at each timestep, handling longitudinal wrapping (0°-360°).	Used to identify object locations.
	Centroid displacement	Distance (km) between centroids across timesteps. Calculated using Haversine distance.	Tracks movement paths; large values = faster movement
	Center-of-mass coordinates	Intensity-weighted mean position per timestep.	Reflects mass distribution shifts
	Center-of-mass displacement	Distance between center-of-mass positions across timesteps.	Quantifies intensity-weighted movement

INTENSITY	Cumulative	Sum of intensity values across space per timestep	Total event magnitude
	Mean	Spatial average intensity	Baseline event strength
	Max.	Spatial maximum intensity	Peak local magnitude
	Std. Dev.	Spatial variability	Higher values = more heterogeneous
	Percentile	Threshold-exceeding intensity (e.g., 90th)	Robust extreme value detection
TEMPORAL	Initial Detection Time	First timestep of event occurrence	Event onset
	Duration	Total timesteps the event persists	Distinguishes transient vs. persistent
	Peak Intensity Timing	When max intensity occurs (% duration)	Early peak = rapid intensification
	Peak Area Timing	When maximum extent occurs	Early peak = rapid growth
CONTEXTUAL	Object counts per timestep	Number of distinct objects detected	Higher counts indicate fragmentation

The computed measures are designed for application across both individual events and multiple events. Each measure is stored in a structured Python dictionary keyed by event ID, supporting single-event analysis and multi-event comparison, where ensemble statistics can be calculated when aggregating across multiple event IDs. The submodule includes built-in visualization tools that generate plots of trajectory maps (centroid/center-of-mass paths with arrow markers indicating directionality).

Implementation example for a few measures

```
event_binary, event_intensity = get_object_masks(blobs, var_notrend, object_id=226.) # specific event
ID

shape = ShapeMeasures(lat_resolution=111.320,
                      lon_resolution=111.320,
                      use_decorators=False)
spatial_extent_data = shape.calc_spatial_extents(event_binary) # Area-related measures

motion = MotionMeasures(use_decorators=False)
centroid_path, centroid_displacements = motion.calculate_centroid_displacement(
  event_binary) # Calculate centroid displacements

# Intensity
intensity_metrics = calculate_intensity_metrics(event_intensity)
```

The nested dictionary output structure accommodates mixed dimensionality, where scalars (e.g., maximum intensity) coexist with vectors (e.g., centroid path) and expansion, where new measures can be added without restructuring existing outputs. This approach is particularly valuable for comparative analyses, where aggregating specific measures across events needs only simple dictionary comprehensions. The Measures submodule allows for user-defined extension, which is an important feature for domain-specific applications.

Implementation example

```
object_ids = [8., 11., 16.]           # Example IDs from Ocetrac tracking output

# Toggle which measures to compute for the objects (True/False flags)
run_shape_flag = True
run_motion_flag = False               # Example: turn off motion measures
run_temporal_flag = True
run_intensity_flag = True


results_for_objects = process_objects_and_calculate_measures(
   object_ids,                        # List of object IDs to process
   blob_data = labelled_field,        # Ocetrac's labeled output (tracked objects)
   intensity_data = intensity_field,  # Intensity data (e.g., SST anomalies)
   run_shape=run_shape_flag,
   run_motion=run_motion_flag,
   run_temporal=run_temporal_flag,
   run_intensity=run_intensity_flag,
   lon_resolution_value=lon_resolution_value,
   lat_resolution_value=lat_resolution_value
)
```

3.4. *Flexibility and Extensibility*

Ocetrac's modularity supports diverse applications beyond MHWs and can be used to analyze atmospheric systems (with geopotential height data), precipitation events, ocean eddies (using sea surface height anomalies), and freshwater plumes (via salinity fields), amongst others. Ocetrac integrates with `xarray` for labeled data handling, `dask` for parallel processing, and common geospatial toolkits, making the package suitable for studying different geophysical phenomena.

4. Example use cases

To demonstrate Ocetrac's broad applicability, we present several example use cases focused on MHWs. These examples illustrate Ocetrac's capabilities in analyzing global patterns, investigating regional dynamics, placing events within a historical context, and evaluating forecast skill.

4.1. *Application to Observational Data*

We apply Ocetrac to the NOAA 1/4° Optimum Interpolation Sea Surface Temperature (OISST) dataset [16], [17] to demonstrate its functionality for global MHW analysis. This methodology for anomaly calculation and results is described in greater detail in H. A. Scannell, C. Cai, L. Thompson, D. B. Whitt, D. J. Gagne, and R. P. Abernathey [15]. This study demonstrates Ocetrac successfully identifying known MHW events and their spatial structure, temporal evolution, and intensity distributions from the observational record. This shows how Ocetrac captures both large-scale and regional MHW events in observational products. The workflow is shown in Figure 3.

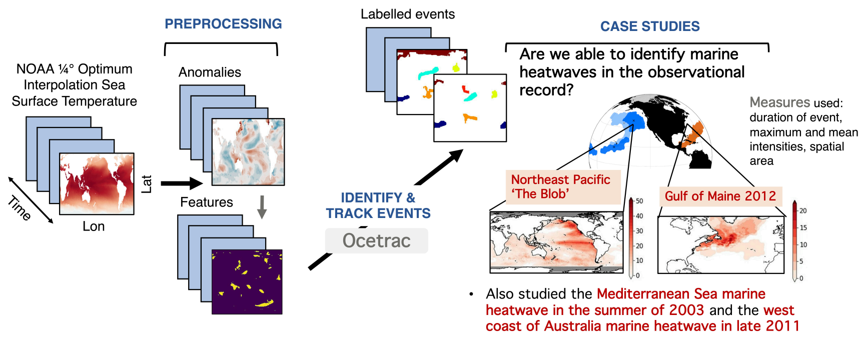

PREPROCESSING
NOAA ¼° Optimum Interpolation Sea Surface Temperature
Time
Lon
Lat
Anomalies
Features
Labelled events
IDENTIFY & TRACK EVENTS
Ocetrac
CASE STUDIES
Are we able to identify marine heatwaves in the observational record?
Measures used: duration of event, maximum and mean intensities, spatial area
Northeast Pacific 'The Blob'
Gulf of Maine 2012
Also studied the Mediterranean Sea marine heatwave in the summer of 2003 and the west coast of Australia marine heatwave in late 2011

Figure 3. *Ocetrac workflow for MHW analysis in an observational data product (in this example, the NOAA 1/4° Optimum Interpolation Sea Surface Temperature dataset). The diagram illustrates the processing pipeline from model output through event detection, tracking, and characterization of observational MHWs.*

4.2. *Characterizing Events in a Large Ensemble*

This use case demonstrates that Ocetrac allows for comprehensive analysis of MHWs across a large ensemble of climate simulations. By processing many ensemble members, we generate an extensive event catalog that is orders of magnitude larger than the observational record. This provides robust statistics for MHW characteristics calculated using the Measures submodule, This expanded dataset supports clustering analyses to identify recurrent event types and their associated physical drivers.

The large number of simulated events allows us to address key questions about MHW diversity: What canonical patterns emerge? How do their dynamical drivers differ? Such analyses would not be possible using the observational record alone. The approach is particularly valuable for characterizing rare but high-impact events, where large ensembles provide sufficient samples to assess their physical rarity and driving mechanisms. The workflow is shown in Figure 4.

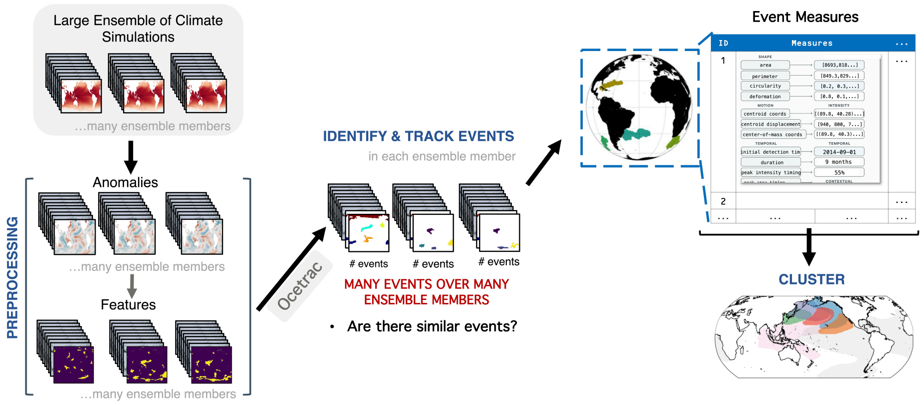


Figure 4. *Ocetrac workflow for MHW analysis in climate ensembles. This workflow focuses on events in the North Pacific Ocean. This approach allows for statistical analysis of event properties.*

4.3. *Comparing Observed Marine Heatwaves Against Many Simulated Events*

In this scenario, we show the use of Ocetrac to compare MHWs in the observational record (i.e., the spatially large and long-lasting Northeast Pacific MHW 'The Blob' in 2014-2016) with events simulated in a large ensemble of climate models, such as the Community Earth System Model version 2 Large Ensemble Community Project (CESM2-LENS) [18]. The approach facilitates a statistical framework for assessing their rarity, spatial structure, and dynamical drivers. By applying consistent object-based detection and tracking to both observations and model output, Ocetrac allows direct comparisons that address questions such as (1) How unusual was the observed event in the context of internal climate variability?, (2) What is the modeled likelihood of similar or more intense MHWs?, and (3) What types of MHWs does the model suggest are possible but have not yet been observed? An analysis workflow is illustrated in Figure 5.

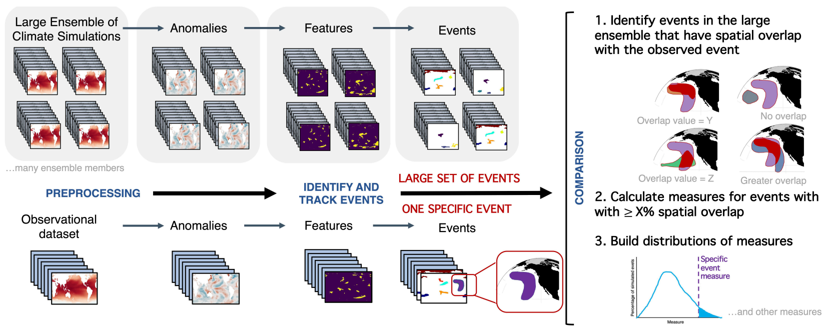
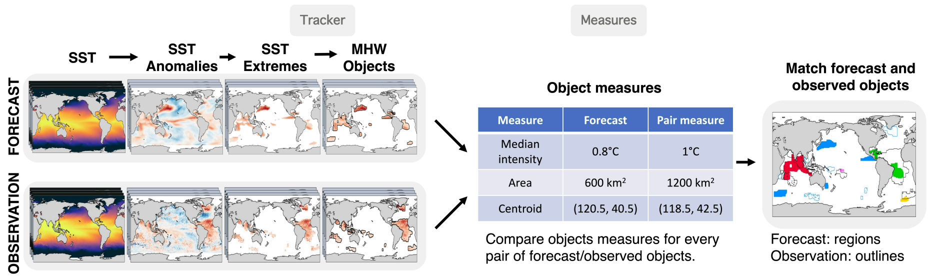

Large Ensemble of Climate Simulations
Anomalies
Features
Events
...many ensemble members
PREPROCESSING
IDENTIFY AND TRACK EVENTS
LARGE SET OF EVENTS
ONE SPECIFIC EVENT
Observational dataset
Anomalies
Features
Events
COMPARISON
1. Identify events in the large ensemble that have spatial overlap with the observed event
Overlap value = Y
No overlap
Overlap value = Z
Greater overlap
2. Calculate measures for events with ≥ X% spatial overlap
3. Build distributions of measures
Specific event measure
Percentage of simulated events
Measure
...and other measures

Figure 5. *Ocetrac workflow for comparing observed and simulated MHWs.*

4.4. *Evaluating Marine Heatwave Predictions Using Object-Based Metrics*

Object-based analysis of MHWs can be applied not only to observations and model output, but also to seasonal forecasts. In recent work, J. T. Cohen, L. Thompson, E. Maroon, A.-L. Deppenmeier, and C. Cai [19] combine Ocetrac with the Method for Object-based Diagnostic Evaluation (MODE) to evaluate MHW forecasts. More information about MODE can be found in C. A. Davis, B. G. Brown, and R. G. Bullock [20], C. A. Davis, B. G. Brown, and R. G. Bullock [21], and R. Bullock, T. Fowler, and B. Brown [22]. Events were identified with `ocetrac.Tracker`, then forecast and observed events were compared and matched based on the similarity of their attributes. Event matching and attribute calculations were performed using MODE. The Measures submodule, however, will allow users to perform this analysis within the Python ecosystem (Figure 6).



Measure	Forecast	Pair measure
Median intensity	0.8°C	1°C
Area	600 km²	1200 km²
Centroid	(120.5, 40.5)	(118.5, 42.5)

Compare objects measures for every pair of forecast/observed objects.

Forecast: regions
Observation: outlines

Figure 6. *Ocetrac workflow for comparing forecast and observed MHW events. Objects are identified and tracked with* `ocetrac.Tracker` *and compared between the forecast and observation datasets with ocetrac.measures*

5. Discussion

Ocetrac is an open-source software suite designed to identify, track, and quantify irregular, evolving features in gridded climate data, such as MHWs. It is actively being developed with extensibility in mind. Future plans include: (1) expanding grid compatibility (e.g., Parallel Ocean Program (POP) grid, (2) enhancing tracking to incorporate vertical structure (e.g., MHWs can have subsurface signatures), and (3) customizing analysis (e.g., different kinds of user-defined thresholding and intensity field extraction). Beyond MHWs, Ocetrac is being adapted for other spatiotemporal phenomena, including atmospheric blocking, ocean eddies, and biogeochemical extremes, with practical examples to be demonstrated in interactive tutorials. The Measures submodule will also be extended to include new measures (e.g., co-occurring event counts) and visualization features. Although Ocetrac was originally developed for MHW research, this package is a versatile tool for ocean, atmospheric, and climate sciences. As a Python-based tool, Ocetrac is designed to evolve with both community needs and computational Earth science advancements. We invite collaboration through conferences, workshops, and open development on GitHub, particularly from researchers working on event detection and feature tracking. These efforts will advance Ocetrac's capacity to analyze the structure of ocean and atmospheric events in observational and model data.

Acknowledgments

We gratefully acknowledge the software and methods contributions to Ocetrac from Ryan Abernathey, Julius Busecke, David John Gagne, and Daniel Whitt (listed alphabetically). The work was supported by funding from NSF, NSF NCAR, the Leonardo DiCaprio Foundation Foundation, Microsoft, the Gordon and Betty Moore Foundation, and the University of Washington eScience Institute.

References

[1] D. Smale *et al.*, "Marine heatwaves threaten global biodiversity and the provision of ecosystem services," *Nat. Clim. Change*, vol. 9, pp. 306–312, 2019, doi: 10.1038/s41558-019-0412-1.

[2] T. P. Hughes *et al.*, "Global warming and recurrent mass bleaching of corals," *Nature*, vol. 543, pp. 373–377, 2017, doi: 10.1038/nature21707.

[3] K. E. Mills *et al.*, "Fisheries management in a changing climate: Lessons from the 2012 ocean heat wave in the Northwest Atlantic," *Oceanography*, vol. 26, no. 2, pp. 191–195, 2013, doi: 10.5670/oceanog.2013.27.

[4] D. J. Gagne, A. McGovern, S. E. Haupt, R. A. Sobash, J. K. Williams, and M. Xue, "Storm-based probabilistic hail forecasting with machine learning applied to convection-allowing ensembles," *Weather and Forecasting*, vol. 32, pp. 1819–1840, 2017, doi: 10.1175/WAF-D-17-0010.1.

[5] P. A. Ullrich, C. M. Zarzycki, E. E. McClenny, M. C. Pinheiro, A. M. Stansfield, and K. A. Reed, "TempestExtremes v2.1: a community framework for feature detection, tracking, and analysis in large datasets," *Geoscientific Model Development*, vol. 14, no. 8, pp. 5023–5048, 2021, doi: 10.5194/gmd-14-5023-2021.

[6] G. Bonino, S. Masina, G. Galimberti, and M. Moretti, "Southern Europe and western Asian marine heatwaves (SEWA-MHWs): a dataset based on macroevents," *Earth System Science Data*, vol. 15, pp. 1269–1285, 2023, doi: 10.5194/essd-15-1269-2023.

[7] R. I. Woolway, E. J. Anderson, and C. Albergel, "Rapidly expanding lake heatwaves under climate change," *Environmental Research Letters*, vol. 16, no. 9, p. 94013, 2021, doi: 10.1088/1748-9326/ac1a3a.

[8] D. Sun, Z. Jing, F. Li, and L. Wu, "Characterizing global marine heatwaves under a spatio-temporal framework," *Progress in Oceanography*, vol. 211, p. 102947, 2023, doi: 10.1016/j.pocean.2022.102947.

[9] D. Sun, F. Li, Z. Jing, and et al., "Correction to: Frequent marine heatwaves hidden below the surface of the global ocean," *Nature Geoscience*, vol. 17, p. 171, 2024, doi: 10.1038/s41561-023-01359-0.

[10] T. Fiolleau and R. Roca, "A database of deep convective systems derived from the intercalibrated meteorological geostationary satellite fleet and the TOOCAN algorithm (2012–2020)," *Earth System Science Data*, vol. 16, no. 9, pp. 4021–4050, 2024, doi: 10.5194/essd-16-4021-2024.

[11] S. van der Walt *et al.*, "scikit-image: Image processing in Python," *PeerJ*, vol. 2, p. e453, 2014, doi: 10.7717/peerj.453.

[12] S. Hoyer and J. Hamman, "xarray: N-D labeled arrays and datasets in Python," *Journal of Open Research Software*, vol. 5, no. 1, p. 10, 2017, doi: 10.5334/jors.148.

[13] M. Rocklin, "Dask: Parallel computation with task scheduling," *Proceedings of the 14th Python in Science Conference*, pp. 130–136, 2015, doi: 10.25080/majora-629e541a-00e.

[14] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, doi: 10.48550/arXiv.1201.0490.

[15] H. A. Scannell, C. Cai, L. Thompson, D. B. Whitt, D. J. Gagne, and R. P. Abernathey, "Spatiotemporal Evolution of Marine Heatwaves Globally," *Journal of Atmospheric and Oceanic Technology*, vol. 41, no. 12, pp. 1247–1263, 2024, doi: 10.1175/JTECH-D-23-0126.1.

[16] R. W. Reynolds, N. A. Rayner, T. M. Smith, D. C. Stokes, and W. Wang, "An improved in situ and satellite SST analysis for climate," *J. Climate*, vol. 15, pp. 1609–1625, 2002, doi: 10.1175/1520-0442(2002)015<1609:AIISAS>2.0.CO;2.

[17] R. W. Reynolds, T. M. Smith, C. Liu, D. B. Chelton, K. S. Casey, and M. G. Schlax, "Daily high-resolution-blended analyses for sea surface temperature," *J. Climate*, vol. 20, pp. 5473–5496, 2007, doi: 10.1175/2007JCLI1824.1.

[18] K. B. Rodgers *et al.*, "Ubiquity of human-induced changes in climate variability," *Earth Syst. Dynam.*, vol. 12, pp. 1393–1411, 2021, doi: 10.5194/esd-12-1393-2021.

[19] J. T. Cohen, L. Thompson, E. Maroon, A.-L. Deppenmeier, and C. Cai, “Object-based evaluation of seasonal-to-multiyear marine heatwave predictions,” *ESS Open Archive*, 2025, doi: 10.22541/essoar.173870867.74048627/v1.

[20] C. A. Davis, B. G. Brown, and R. G. Bullock, “Object-based verification of precipitation forecasts, Part I: Methodology and application to mesoscale rain areas,” *Mon. Wea. Rev.*, vol. 134, pp. 1772–1784, 2006, doi: 10.1175/mwr3145.1.

[21] C. A. Davis, B. G. Brown, and R. G. Bullock, “Object-based verification of precipitation forecasts, Part II: Application to convective rain systems,” *Mon. Wea. Rev.*, vol. 134, pp. 1785–1795, 2006, doi: 10.1175/MWR3146.1.

[22] R. Bullock, T. Fowler, and B. Brown, “Method for Object-Based Diagnostic Evaluation,” 2016. doi: 10.5065/D61V5CBS.