A Demonstration of Polaris: An Interactive and Scalable Data Infrastructure for Polar Science

Yuchuan Huang, Ana Elena Uribe, Youssef Hussein, Grant Ogren, Kareem Eldahshoury, Mohamed F. Mokbel
University of Minnesota, USA
{huan1531,uribe055,husse408,ogren091,eldah001,mokbel}@umn.edu

ABSTRACT

This demonstration presents Polaris; a novel open-source system infrastructure for Polar science that is highly Interactive and Scalable. Polaris is designed based on three observations that distinguish the query workload of polar scientists, namely, all queries are spatio-temporal, not all data are equal, and the large majority of queries are aggregates. With this, Polaris is equipped with a hierarchical spatio-temporal index structure that stores precomputed aggregates for data of interest. Audience will be able to experience Polaris through various scenarios that show the interactivity and scalability as well as Polaris optimized query processes.

PVLDB Reference Format:
Yuchuan Huang, Ana Elena Uribe, Youssef Hussein, Grant Ogren, Kareem Eldahshoury, Mohamed F. Mokbel. A Demonstration of Polaris: An Interactive and Scalable Data Infrastructure for Polar Science. PVLDB, 18(12): 5347 - 5350, 2025.
doi:10.14778/3750601.3750668

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/iharp3/Polaris-VLDB2025/.

1 INTRODUCTION

Polar scientists face the challenge that even though there are huge amounts of climate and environment datasets archived online (e.g., ERA5 [8], CARRA [3], MERRA2 [13], ICESat-2 [10], CESM [5]), they are not easy to access and work on. For most of such datasets, scientists have to wait hours or days to first download the data, then write their own scripts for their analysis needs. This is definitely far from being an interactive user experience, which hinders the polar scientists' ability to perform their analysis and gain insightful thoughts from the data. For example, ERA5 [8], a reanalysis dataset for the global climate and atmosphere, is one of the most widely used datasets by polar scientists for simulation, prediction, and modeling [6, 9, 12, 18, 19]. ERA5 has 262 climate variables, including temperature, snowfall, ice sheet, etc. Each variable is recorded at every 0.25 × 0.25 latitude longitude degree spatial area for the whole world and for every hour from 1940 to present. With such high spatial and temporal resolutions, one variable for the whole world consumes an annual storage of 17GB. This makes the overall ERA5 dataset size for 262 variables and 84 years approximately 374TB.

Query	Running time	Result size	Data download	Polaris
Daily	12m46s	34 MB	20.12 GB	1.2s
Monthly	12m55s	1.2 MB	20.12 GB	0.3s

Table 1: Query Performance Metrics

Due to its size, it is hard for polar scientists to host and manage such data, not to mention have efficient data access and interactive analysis. Currently, polar scientists employ one of the following three options to work with ERA5 data: (1) The most straightforward way for polar scientists is to call public APIs to download the parts they need from ERA5 data to their local storage and then run their scripts from there. This is pretty inefficient as the downloading itself can take hours or more, then considerable efforts are needed to write and execute the analysis modules. (2) Download and store major parts of the data on a High Performance Computing (HPC) environment and run computations on HPC directly. Though this option will be the best in terms of interactive analysis, it is the least used approach, as such an HPC environment is not available to the large majority of polar scientists worldwide. (3) An emerging trend from the geoscience community [14] is to transform the data into Analysis-Ready Cloud-Optimized (ARCO) format stored in a cloud storage [1, 17], e.g., ARCO-ERA5 [4] is an ARCO version of ERA5 data hosted on Google Cloud. Compared with the API-based approach, the ARCO approach integrates better with modern data ecosystems, yet it is still far from being interactive. In particular, Table 1 gives performance measures of running two queries on ARCO-ERA5 [4] regarding daily and monthly temperatures of Alaska in 2020. Though the result size for the daily query is 34MB and the monthly query is 1.2MB, both queries end up downloading the same amount of data of 20GB. The main reason is that ARCO-ERA5 must scan all the raw data before aggregation, making it less interactive on aggregate queries (e.g., daily queries) that are highly used by polar scientists. So, it takes close to 13 minutes to execute both queries, which is not suitable for any interactive analysis.

This demo presents Polaris; a novel open-source system infrastructure for Polar science that is highly Interactive and Scalable. Polaris came out as part of the iHARP project (institute for HArnessing data and model Revolution in the Polar regions) [11], which is a large collaboration effort between computer and polar scientists to provide system infrastructure and data analysis techniques for polar scientists. Unlike all previous approaches used by polar scientists, Polaris is tailored to the query workload and access patterns of polar scientists, and hence it provides a highly interactive and scalable performance for the large majority of the queries it receives. For example, Polaris answers the daily and

This work is supported by NSF grants OAC-2118285 and IIS-2203553.

Proceedings of the VLDB Endowment, Vol. 18, No. 12 ISSN 2150-8097.
doi:10.14778/3750601.3750668

monthly queries of the ERA5 data in Table 1 in 1.2 and 0.3 seconds, respectively, with no data download.

Polaris is built with three main concepts in mind. (1) *All polar scientists queries are spatio-temporal.* Queries are always asking about any of the 262 variables of ERA5 within specific spatial and temporal ranges. (2) *Data Inequality.* Not all data are of equal importance. Some of the variables are needed in some areas of the world more than others at higher or lower resolutions. (3) *Most queries are aggregates.* Queries typically ask about an aggregate value of a specific variable, spatio-temporal range, and spatial and temporal resolutions, e.g., get the days (temporal resolution) of last year (temporal range) with maximum snowfall (aggregate) in any 1-degree area (spatial resolution) of Alaska (spatial range).

The demo lets conference attendees experience Polaris in action. Attendees will be able to issue various queries like polar scientists through a web-based GUI and see the visualized results at an interactive response time. To help attendees understand Polaris internals, they will be able to see the execution plan of each query.

2 POLARIS QUERY SIGNATURE AND ARCHITECTURE

Polaris is designed to support the following query signature, which represents the family of queries that are most popular for polar scientists. Per the query signature, it is mandatory to have spatial and temporal predicates in all Polaris queries, which drives the internal index structure design and query processing of Polaris.

```
SELECT <Spatial Resolution>, <Temporal Resolution>,
       <[<Min/Max/Avg>] variable>,
  FROM Data
 WHERE <Spatial Predicate> AND <Temporal Predicate>
[ GROUP BY <Spatial Resolution (0.25/0.5/1-degree)>,
           <Temporal Resolution (Hour/Day/Month/Year)>,
  [ HAVING <group predicate> ] ]
```

Following the signature, there are five representative query types that Polaris can support, namely:

Get Variable Query. This query requests the aggregate value within certain spatial and temporal ranges (at certain resolutions), e.g., *"Get the daily average temperature in 2020 of Alaska at 1-degree"*.

Heatmap Query. This query outputs a two-dimensional array where the average value of an area is reported, basically composing a heatmep. For example, *"Build a 1-degree heatmap of the average temperature of Alaska during 2021/01/01 to 2023/01/31"*.

Timeseries Query. This query essentially creates a time series (i.e., one-dimensional array) indicating how the requested variable changes within an area during a time range. For example, *"Get the daily average temperature in 2022 of Greenland"*.

Find Area Query. This query finds the areas, at the given spatial resolution, where a certain value predicate is satisfied. For example, *"Find the 0.25-degree areas in Alaska that had an average temperature greater than 300 Kelvin in 2023"*.

Find Time Query. This query finds the time periods, at a given temporal resolution, where a certain value predicate is satisfied. For example, *"Find the days in 2023 where the average temperature in Antarctica is greater than 300 Kelvin"*.

To efficiently support each query above, Polaris has a system architecture depicted in Figure 1 which is composed of three main components, and will be described in the following three sections.

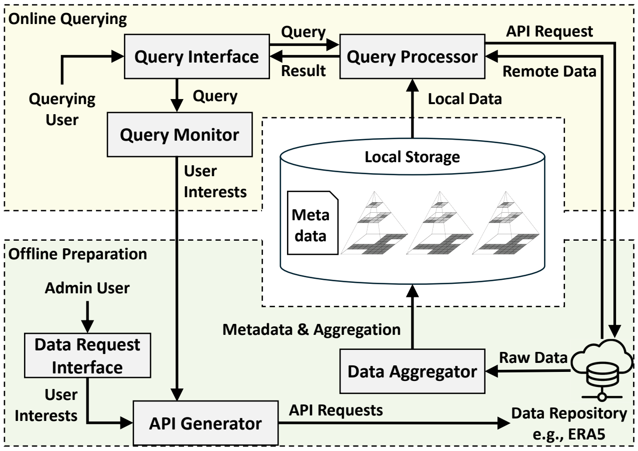


Figure 1: Polaris **System Architecture**

3 POLARIS DATA PREPARATION

This layer is responsible for storing the most important data of ERA5 such that the large majority of queries received by Polaris are completely supported from local storage. This layer's input specifies the data of interest and comes from an admin user through a GUI data request interface. Internally, this input is converted to a set of APIs that download the corresponding raw data, which is then aggregated to the requested spatial and temporal resolutions, and sent to the local indexed storage. Thus, the following three main components make up this layer:

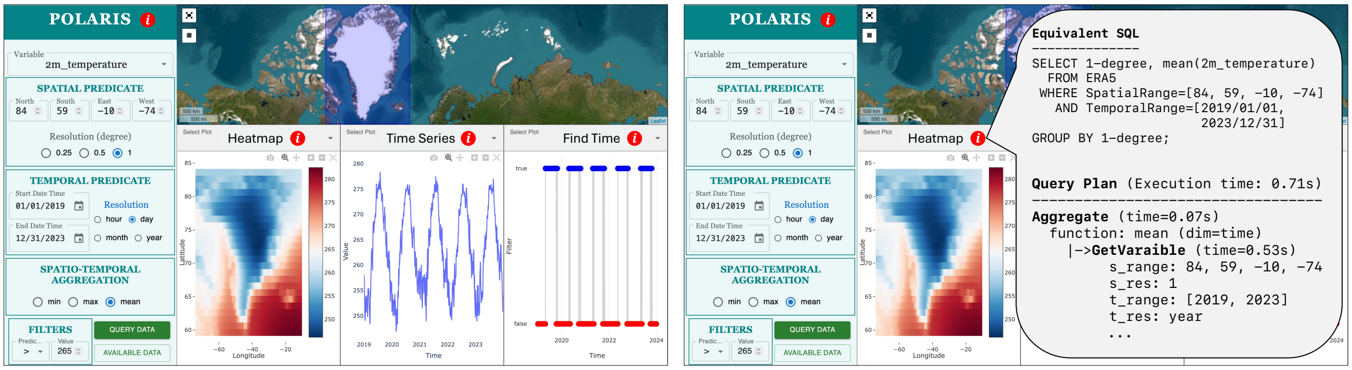
Data Request Interface. This GUI map interface allows admin users to indicate users' interests on different variables for any spatial and temporal ranges and resolutions. The specified user interests, along with the ones derived from historical queries by the query monitor (Section 5), trigger the API generator module.

API Generator. This module is triggered by the input of user interests, where it digests all the user entries to find the minimum set of APIs that need to be issued to ERA5 cloud services [8] to download the requested data.

Data Aggregator. This module takes the raw data downloaded by the API generator and performs three operations: (1) It aggregates the raw data to the requested resolution while deleting the unneeded raw data. (2) It precomputes the aggregations at coarser resolutions to store alongside the requested resolutions. (3) It stores a metadata table for all locally stored aggregated data, which is used by the index structure and query processor to locate such data.

4 POLARIS LOCAL STORAGE AND INDEX STRUCTURE

This layer is responsible for indexing the aggregated data from the offline data preparation layer, which is then accessed by the online query layer for interactive data analysis. Polaris designs its own index structure, as existing big data systems (e.g., TileDB [15], Apache Sedona [16], or SpatialHadoop [7]) lack spatio-temporal support, which is immensely needed per the query signature defined in Section 2. In particular, the query signature calls for an index stricture that is: (a) *spatio-temporal*, as spatial and temporal predicates are mandatory, (b) *hierarchical*, as queries impose a natural hierarchy of spatio-temporal resolutions, and (c) *precomputed aggregates in*



POLARIS

Variable: 2m_temperature

SPATIAL PREDICATE

North 84 | South 59 | East −10 | West −74

Resolution (degree): ○ 0.25 ○ 0.5 ● 1

TEMPORAL PREDICATE

Start Date Time: 01/01/2019

End Date Time: 12/31/2023

Resolution: ○ hour ● day ○ month ○ year

SPATIO-TEMPORAL AGGREGATION

○ min ○ max ● mean

FILTERS

Predic...: > Value: 265

QUERY DATA

AVAILABLE DATA

```
Equivalent SQL
--------------
SELECT 1-degree, mean(2m_temperature)
  FROM ERA5
 WHERE SpatialRange=[84, 59, -10, -74]
   AND TemporalRange=[2019/01/01,
                      2023/12/31]
GROUP BY 1-degree;


Query Plan (Execution time: 0.71s)
------------------------------------
Aggregate (time=0.07s)
  function: mean (dim=time)
    |->GetVaraible (time=0.53s)
        s_range: 84, 59, -10, -74
        s_res: 1
        t_range: [2019, 2023]
        t_res: year
        ...
```

(a) POLARIS **Query Interface**

(b) POLARIS **Query Process Explanation**

Figure 2: POLARIS **Query Interface and Process**

index entries, as most queries are requesting aggregate values. To this end, POLARIS employs a basic index unit that supports spatial predicates, hierarchy, and aggregates; and a full index structure that adds support temporal-wise.

Basic Spatial Indexing Unit. The basic indexing unit in POLARIS indexes a certain variable and stores its aggregated values over multiple spatial resolutions for a specific temporal interval and specific temporal resolution. The index takes the shape of an incomplete pyramid structure [2], as shown in Figure 1. The lowest pyramid level basically divides the geographical space of the whole world into non-overlapping equal-size cells. The medium and top pyramid level stores the aggregation of every four cells of its lower level. Not all cells are maintained, and hence the term *incomplete* pyramid.

Spatio-temporal Index Structure. The full spatio-temporal index structure for each indexed variable in POLARIS is composed of multiple instances of the basic incomplete pyramid unit to support temporal aggregation and resolutions, where the basic incomplete pyramid unit is replicated for each hour, day, month, year. Just like not all cells of a basic pyramid unit are maintained, not all pyramids are maintained at all temporal levels.

5 POLARIS QUERY PROCESS

This layer gets its query input through a GUI interface, and is responsible for providing efficient query processing through the indexed local storage. It is also equipped with a monitoring module that monitors the system query workload to dynamically adjust the local storage given workload changes.

Query Processor. POLARIS query processor aims to fully exploit its hierarchical index structure, described in Section 4, to efficiently answer incoming queries from local storage. The basic strategy is to answer the query from high-level pre-aggregation as much as possible, which significantly reduces the data access and computing overhead. For those queries that cannot be fully answered from local storage, the query processor partially answers the query, and then calls an API for the part of the query that is not locally available.

Query Monitor. The main performance promise of POLARIS is based on the idea that the large majority of queries will be answered from local storage. To ensure such promise, POLARIS employs a query monitoring module that tracks: (1) user-issued queries in terms of their variable and spatial/temporal range and resolution and (2) the ratio of queries that were not completely supported from local storage. Once the ratio exceeds a user-specified value, this module will analyze the query history to get the real user interests, send the interests to the API generator and trigger the data preparation in the offline layer.

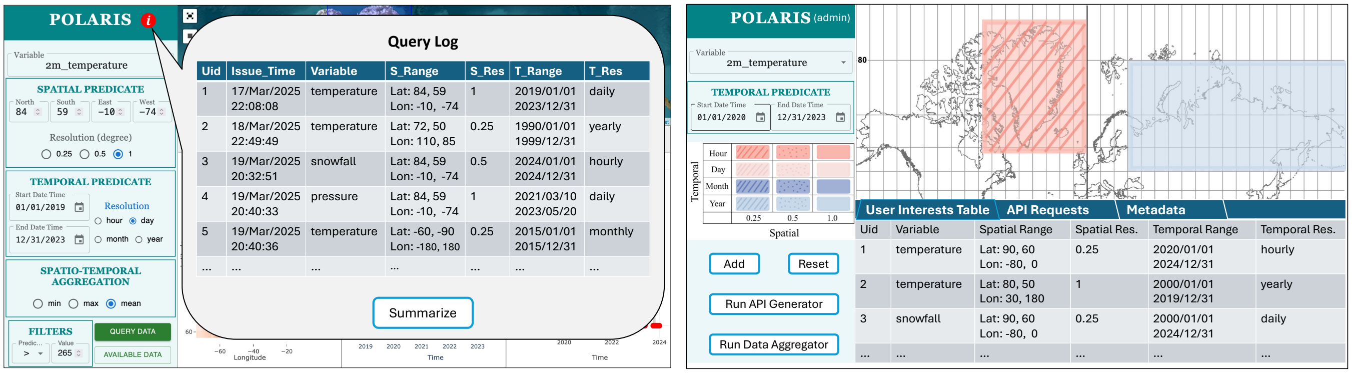
6 DEMO SCENARIOS

This section shows four demo scenarios where the conference attendees can interact with POLARIS to understand its operations from the point of view of: (a) Polar scientists who need to use the system to explore datasets and gain instant insights, and (b) Researchers, developers, and practitioners who want to understand the system internals. Since the data preparation is an offline process that takes time, we will download, aggregate, and index the data in advance. Attendees will mainly experience the interactive online query process. A demo deployment of POLARIS[1] is available online.

Demo Scenario 1: Interactive Queries. This demo aims to show the interactive query experience of POLARIS. Figure 2(a) depicts the user interface of POLARIS, on which conference attendees can issue queries to POLARIS to explore the ERA5 datasets and gain quick insights on the data. On the left sidebar, attendees can select a variable that POLARIS has downloaded and indexed offline. They can specify the temporal range by selecting the start and end datetime, and specify the spatial range by either typing the latitude and longitude boundaries or drawing a box on the map, and select the desired spatial and temporal resolution, and aggregate function. Attendees can also set a filter for find time and find area queries by choosing a predicate and a filter value. Query results will be rendered in the right bottom panels. In Figure 2(a), three results for the heatmap, time series and find time queries about the temperature for Greenland from 2019 to 2023 are demonstrated from left to right. In the heatmap, the spatial area of Greenland is partitioned into 1-degree cells (spatial resolution) and the cells are colored to indicate the average temperature (aggregate function) over the five years. In the time series, the line shows how the average temperature of Greenland changes day by day (temporal resolution). Lastly, the find time query plots the days where the average temperature of Greenland is greater than 265 Kelvin (value filter) as *True*.

Demo Scenario 2: Understanding POLARIS **Query Process.** This scenario aims to help attendees understand POLARIS' query processes and optimizations. At the side of each query plot, there is a

[1] https://iharpv.cs.umn.edu/



Variable: 2m_temperature

SPATIAL PREDICATE

North 84 South 59 East −10 West −74

Resolution (degree) ○ 0.25 ○ 0.5 ● 1

TEMPORAL PREDICATE

Start Date Time 01/01/2019

End Date Time 12/31/2023

Resolution ○ hour ● day ○ month ○ year

SPATIO-TEMPORAL AGGREGATION

○ min ○ max ● mean

FILTERS Predic... > Value 265

Query Log

Uid	Issue_Time	Variable	S_Range	S_Res	T_Range	T_Res
1	17/Mar/2025 22:08:08	temperature	Lat: 84, 59 Lon: -10, -74	1	2019/01/01 2023/12/31	daily
2	18/Mar/2025 22:49:49	temperature	Lat: 72, 50 Lon: 110, 85	0.25	1990/01/01 1999/12/31	yearly
3	19/Mar/2025 20:32:51	snowfall	Lat: 84, 59 Lon: -10, -74	0.5	2024/01/01 2024/12/31	hourly
4	19/Mar/2025 20:40:33	pressure	Lat: 84, 59 Lon: -10, -74	1	2021/03/10 2023/05/20	daily
5	19/Mar/2025 20:40:36	temperature	Lat: -60, -90 Lon: -180, 180	0.25	2015/01/01 2015/12/31	monthly
...	...	...	...	...	...	...

Summarize

(a) Polaris Query Monitoring

Variable: 2m_temperature

TEMPORAL PREDICATE

Start Date Time 01/01/2020

End Date Time 12/31/2023

Uid	Variable	Spatial Range	Spatial Res.	Temporal Range	Temporal Res.
1	temperature	Lat: 90, 60 Lon: -80, 0	0.25	2020/01/01 2024/12/31	hourly
2	temperature	Lat: 80, 50 Lon: 30, 180	1	2000/01/01 2019/12/31	yearly
3	snowfall	Lat: 90, 60 Lon: -80, 0	0.25	2000/01/01 2024/12/31	daily
...	...	...	...	...	...

(b) Polaris Offline Data Preparation

Figure 3: Polaris Monitoring and Preparation

red *info* button which, when clicked, shows the query that creates the plot in terms of the query signature described in Section 2, as well as the execution plan in Polaris. Figure 2(b) shows the heatmap query, which gets the average temperature of Greenland at 1-degree spatial resolution from 2019 to 2023 and the corresponding query plan. The query plan displays the query execution time of 0.71 second, which confirms the interactivity of Polaris. The detailed query process is listed in a tree structure where each sub-execution is highlighted by a → mark with parameters listed as indented underneath. The whole plan is executed from the inner to the outmost level. In particular, to answer this heatmap query, Polaris issues a *GetVariable* query to get the temperature data of Greenland at a yearly resolution for the five years. Then Polaris performs an *Aggregation* to compute the average over the "time" dimension. Thus, for each 1-degree cell, Polaris computes the average values of the five yearly temperature values.

Demo Scenario 3: Query Monitoring. In order to show how query monitoring works in Polaris, attendees can click the red *info* button next to the POLARIS header to view a log of historical queries, as shown in Figure 3(a). The log will contain the information of each query, including its variable name, spatial range and resolution, temporal range and resolution, as well as the time that the query was issued. Attendees can then click the *Summarize* button at the bottom, which will summarize the query history and generate a user interest table that can be used to guide the offline data preparation. The bottom panel on Figure 3(b) gives an example of the user interest table.

Demo Scenario 4: Offline Data Preparation. Although data preparation is an offline process that is hard to demo, we mimic the process so that attendees can have a complete understanding of the whole lifecycle of Polaris. Attendees will play on the interface (shown in Figure 3(b)) to indicate their data of interest. To create a user interest, attendees can first select a variable and specify a time period on the left sidebar. On the right, there is a map where the whole world is partitioned into blocks. Attendees can fill the blocks with colors chosen from the sidebar, which indicate a specific combination of spatial and temporal resolution for the spatial area of the blocks. In this way, attendees specify their interest in a specific variable at a spatial and temporal range of certain spatial and temporal resolution. By clicking the *Add* button, they add the interest to the user interest table. After inputting several interests, attendees can click the *Run API Generator* button to simulate the function of that module and see the set of API requests created by the API generator. The requests will not be sent to the ERA5 repository, rather we will pretend the data is downloaded so attendees can click the *Run Data Aggregator* button to mimic the data aggregation process and see the results of pre-aggregation as entries in the metadata.

REFERENCES

[1] R. Abernathey, T. Augspurger, A. Banihirwe, C. C. Blackmon-Luca, T. J. Crone, C. L. Gentemann, J. Hamman, N. Henderson, C. Lepore, T. A. McCaie, N. H. Robinson, and R. P. Signell. Cloud-Native Repositories for Big Scientific Data. *Comput. Sci. Eng.*, 23(2):26–35, 2021.

[2] W. G. Aref and H. Samet. Efficient Processing of Window Queries in The Pyramid Data Structure. In *PODS*, 1990.

[3] Arctic regional reanalysis on single levels from 1991 to present. https://cds.climate.copernicus.eu/datasets/reanalysis-carra-single-levels.

[4] R. W. Carver and A. Merose. ARCO-ERA5: An Analysis-Ready Cloud-Optimized Reanalysis Dataset. https://github.com/google-research/arco-era5.

[5] NCAR Community Earth System Model. https://www.cesm.ucar.edu/.

[6] J. C. Dullaart, S. Muis, N. Bloemendaal, and J. C. Aerts. Advancing global storm surge modelling using the new ERA5 climate reanalysis. *Climate Dynamics*, 54:1007–1021, 2020.

[7] A. Eldawy and M. F. Mokbel. SpatialHadoop: A MapReduce framework for spatial data. In *ICDE*, 2015.

[8] ERA5 hourly data on single levels from 1940 to present. https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels.

[9] H. Hersbach, B. Bell, P. Berrisford, S. Hirahara, A. Horányi, J. Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, D. Schepers, et al. The ERA5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020.

[10] Ice, Cloud, and land Elevation Satellite-2 (ICESat-2). https://icesat-2.gsfc.nasa.gov/.

[11] iHARP: NSF HDR Institute for Harnessing Data and Model Revolution in the Polar Regions. https://iharp.umbc.edu/.

[12] P. Mateus, J. C. Fernandes, V. B. Mendes, and G. Nico. An ERA5-Based Hourly Global Pressure and Temperature (HGPT) Model. *Remote. Sens.*, 12(7):1098, 2020.

[13] Modern-Era Retrospective analysis for Research and Applications, Version 2. https://gmao.gsfc.nasa.gov/reanalysis/MERRA-2/.

[14] Pangeo: A community for open, reproducible, scalable geoscience. https://www.pangeo.io/.

[15] S. Papadopoulos, K. Datta, S. Madden, and T. G. Mattson. The TileDB Array Data Storage Manager. *PVLDB*, 10(4):349–360, 2016.

[16] Apache Sedone. https://sedona.apache.org/.

[17] C. Stern, R. Abernathey, J. Hamman, R. Wegener, C. Lepore, S. Harkins, and A. Merose. Pangeo forge: crowdsourcing analysis-ready, cloud optimized data production. *Frontiers in Climate*, 3:782909, 2022.

[18] C. Vitolo, F. Di Giuseppe, C. Barnard, R. Coughlan, J. San-Miguel-Ayanz, G. Libertá, and B. Krzeminski. ERA5-based global meteorological wildfire danger maps. *Scientific data*, 7(1):216, 2020.

[19] Y.-R. Wang, D. O. Hessen, B. H. Samset, and F. Stordal. Evaluating global and regional land warming trends in the past decades with both MODIS and ERA5-Land land surface temperature data. *Remote Sensing of Environment*, 280:113181, 2022.