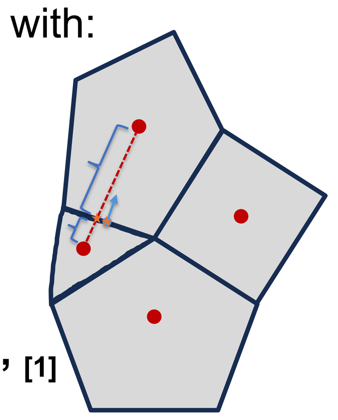
Scalable Vector Calculus for Geoscientific Analysis on Unstructured Grids in UXarray

Esther Gallmeier[1,2], Philip Chmielowiec[1], Orhan Eroglu[1], Katelyn Fitzgerald[1], Rajeev Jain[3]
[1]NSF National Center for Atmospheric Research (NCAR), [2]Cornell University, [3]Argonne National Laboratory

Unstructured Grids

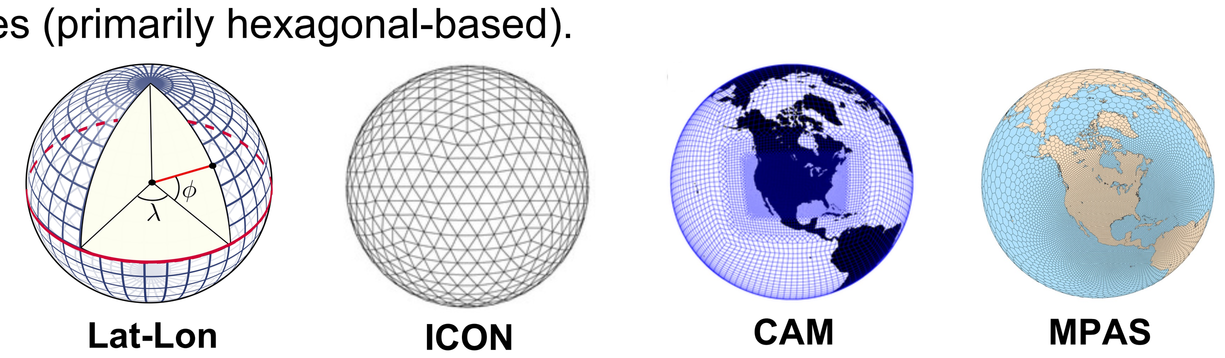
Unstructured grids on a sphere consist of arbitrary spherical polygons (*not necessarily the same shape or size)*, which can result in grids with:

- Unevenness
- Non-orthogonality
- Skewness



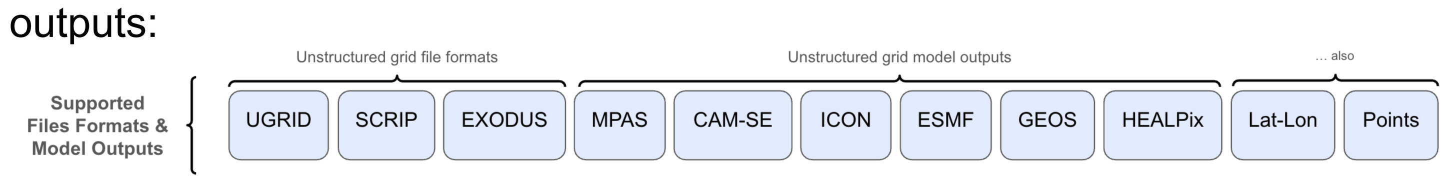
A classical grid is a **lat-lon** grid, where the sphere is discretized via the longitude and latitude lines. However, this discretization results in what is known as the **"pole problem."** [1]

Many climate models now use all sorts of unstructured grids. The German national meteorological service (DWD) uses an icosahedral grid **(ICON)**. NCAR's Community Atmosphere Model (**CAM**) uses a cubed sphere grid. LANL's and NCAR's Model for Prediction Across Scales (**MPAS**) uses Voronoi meshes (primarily hexagonal-based).



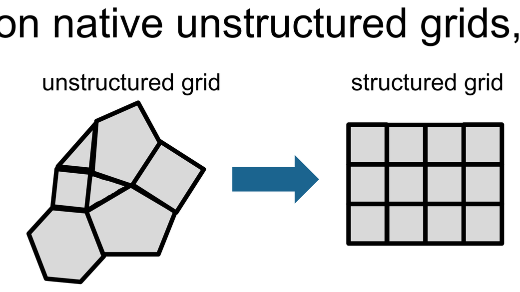
UXarray

UXarray is an open-source Python package for geoscientific **data analysis** & **visualization** on unstructured grids. There is no widely-used convention for storing arbitrary unstructured grids, which makes UXarray a useful tool because it supports a **wide range** of unstructured grid file formats & model outputs:



Thanks to UXarray's ability to operate directly on native unstructured grids, users can **avoid regridding** their unstructured mesh to a structured mesh, which means:

- No duplication of memory
- No introduction of discrepancy in data
- Eliminates extra overhead

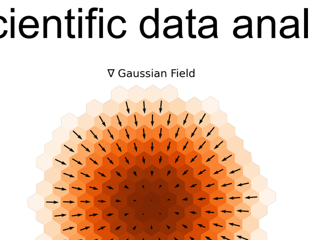


although some regridding options are still available in UXarray if needed.

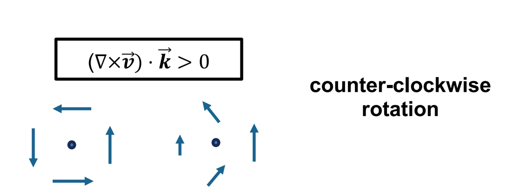
Motivation

Adding vector calculus to UXarray is essential for geoscientific data analysis:

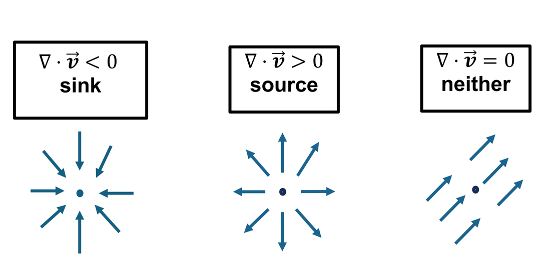
- **Gradient**
  Measures magnitude & direction of steepest change of a scalar field, *such as temperature, salinity, pressure, density,…*



- **Curl**
  Measures circulation of a vector field *from wind velocity, ocean currents, magnetic fields, …*



- **Divergence**
  Measures how a vector field acts like a "sink" or a "source" *to understand upwelling/downwelling, heat or gas fluxes, high/low pressure from rising or sinking motion in horizontal wind fields,…*

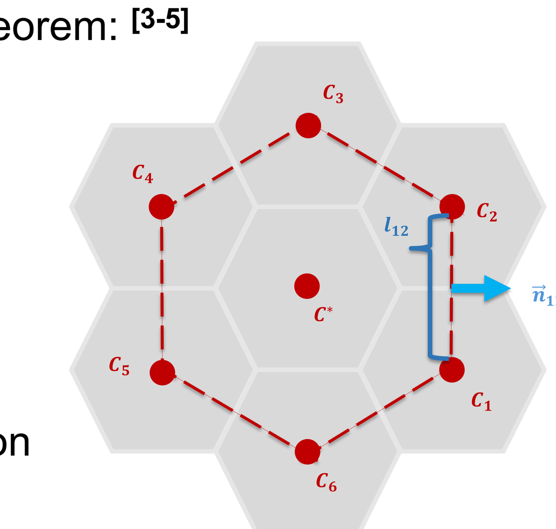


Gradient Implementation

We use a **finite volume discretization** of the Green-Gauss theorem: [3-5]

$$\int_V \nabla\phi \, dV = \oint_{\partial V} \phi \, dS$$

i.e., the integral of a gradient vector field over a closed region is equal to the boundary integral of the corresponding scalar field



For face-centered data

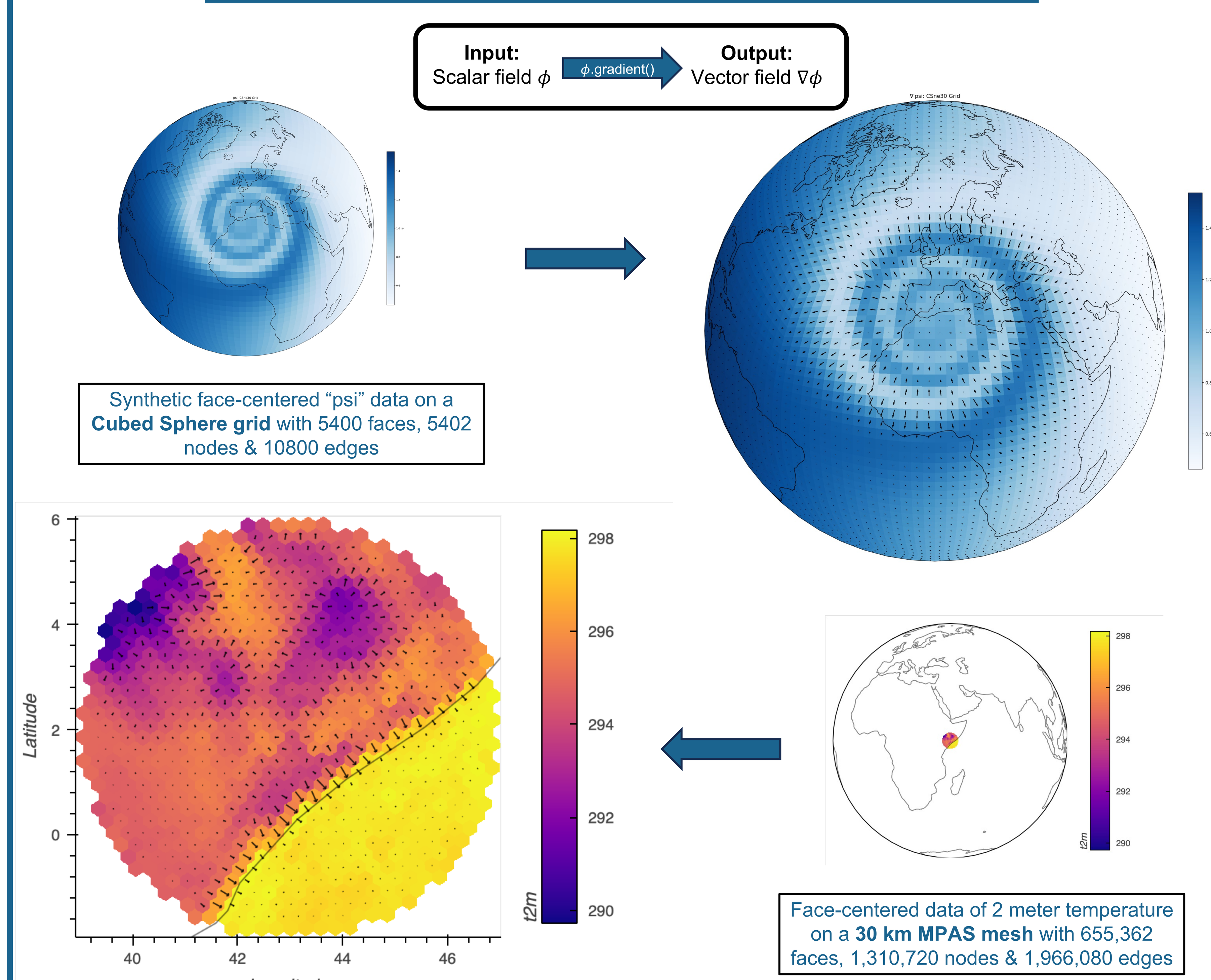
Want: Compute gradient at the face center C^*

Choose: Closed region to connect faces which share a common node with the face center C^*

Approximate: $$\nabla\phi(C^*) \approx \frac{1}{Vol(C^*)} \sum_{i,j} \frac{\phi(C_i)+\phi(C_j)}{2} \, l_{ij} \, \vec{n}_{ij}$$

Gradient Visualization

Input: Scalar field ϕ → ϕ.gradient() → **Output:** Vector field $\nabla\phi$

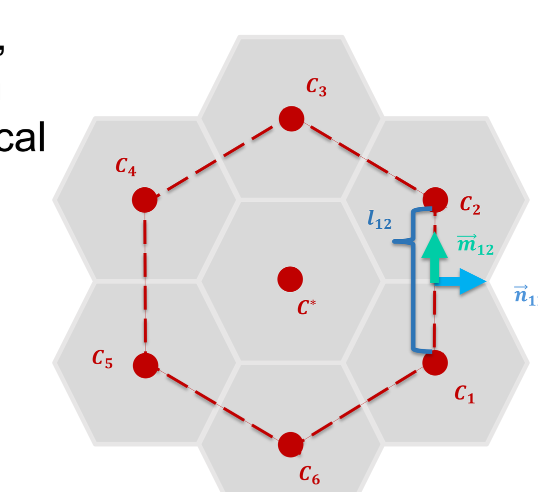


Synthetic face-centered "psi" data on a **Cubed Sphere grid** with 5400 faces, 5402 nodes & 10800 edges

Face-centered data of 2 meter temperature on a **30 km MPAS mesh** with 655,362 faces, 1,310,720 nodes & 1,966,080 edges

Curl + Divergence

In a similar manner as the gradient implementation (see above), we can use finite volume discretizations of the Stokes' Theorem and the Gauss's Theorem to obtain approximations for the vertical component of the curl and the divergence.



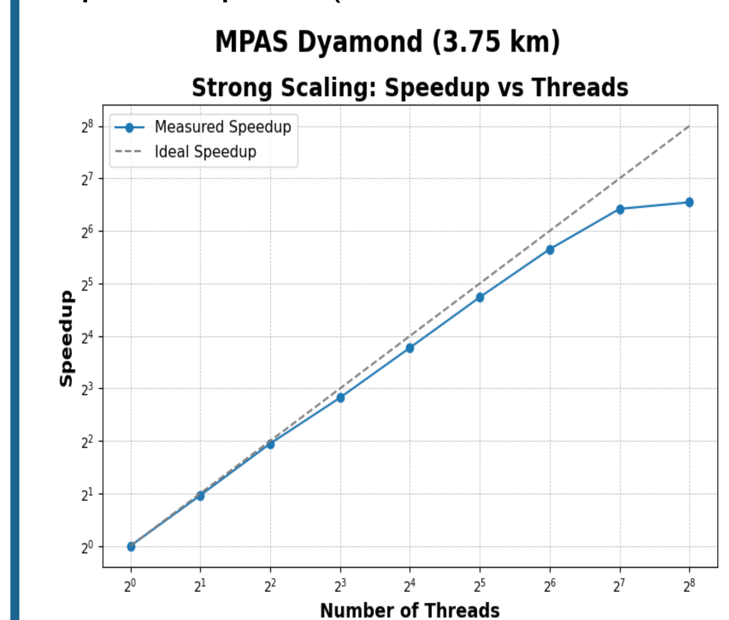
Curl: $$\nabla\times\vec{v}(C^*)\cdot\hat{k} \approx \frac{1}{Vol(C^*)}\sum_{i,j}\frac{\vec{v}(C_i)+\vec{v}(C_j)}{2} \, l_{ij} \cdot \vec{m}_{ij}$$

Divergence: $$\nabla\cdot\vec{v}(C^*) \approx \frac{1}{Vol(C^*)}\sum_{i,j}\frac{\vec{v}(C_i)+\vec{v}(C_j)}{2} \, l_{ij} \cdot \vec{n}_{ij}$$

Gradient Performance

Geoscientific grids can be large—e.g., **~18.45 GB for the 3.75 km MPAS grid**—and high-resolution climate simulations can produce terabytes of data across hundreds of variables and timesteps. Performance is therefore critical. Below are two scaling plots showing gradient performance on a single data variable and timestep, tested on one CPU node of NCAR Derecho's supercomputer (3rd Gen AMD dual-socket, 64 cores/socket, 2 threads/core).[6]

MPAS Dyamond (3.75 km)



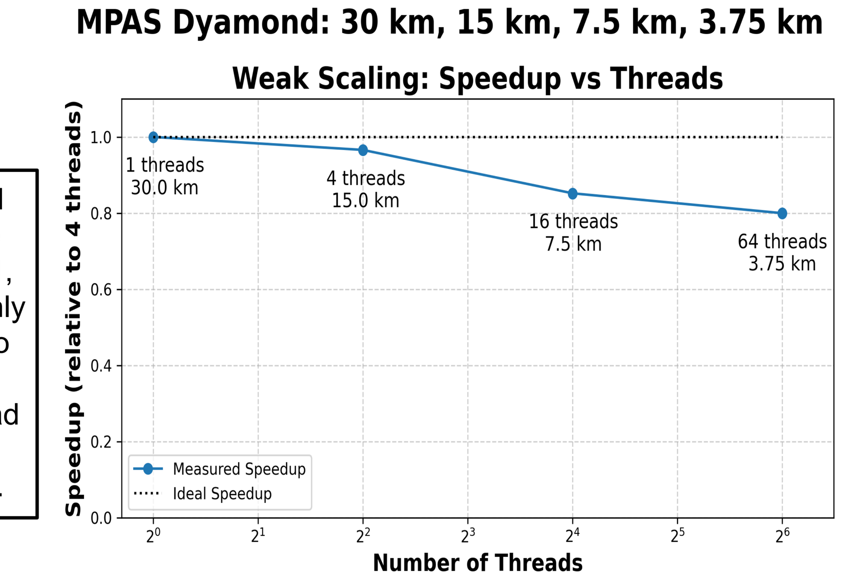
Strong Scaling
(**reduced** workload per thread)

In the strong scaling plot on the left, the grid resolution is held at a **constant grid resolution** of 3.75 km for the MPAS mesh while the **number of threads increase** from 1, 2, 4, …, 256, resulting in a reduced workload per thread. In the ideal scenario, the speedup would scale linearly (see dashed line). However, we see the speedup taper off, in particular between 128 & 256 threads.

MPAS Dyamond: 30 km, 15 km, 7.5 km, 3.75 km

Weak Scaling
(**constant** workload per thread)

In the weak scaling plot on the right, the **grid resolution increases** (i.e., gets finer) as we also **increase the number of threads** from 1, 2, 4, …, 256. Halving the resolution size roughly quadruples the amount of work per thread, so we need to simultaneously quadruple the number of threads to keep a constant workload per thread. Ideally, we would not see a decrease in performance (see dashed lines).



Future Work

Looking ahead, we plan to:

- Implement robust and optimized versions of **additional operators**, including curl, divergence & Laplacian
- Develop **comprehensive documentation** and **workflows** to support the community
- Add vector field visualization functionality to UXarray
- Look for **feedback** from our users

uxarray
https://uxarray.readthedocs.io

Please reach out to us!

GitHub
https://github.com/UXARRAY/uxarray

Acknowledgements

Firstly, I would like to thank **you** (the audience) for reading about my work. I would also like to acknowledge my mentors **Philip Chmielowiec**, **Orhan Ergolu**, **Katelyn Fitzgerald** and **Rajeev Jain** for providing me this opportunity to work with them on this project. Much gratitude goes to **Virginia Do** and **Jessica Wang** both of whom organized the SIParCS internship behind the scenes for it to run so smoothly and effectively. Many thanks go to all of the **summer interns** – we remain connected through friendships and memories. This summer I also had insightful discussions with **Allison Baker** and **Peter Lauritzen** – thank you for talking to me even though I was not your intern. Finally, I enjoyed all the wonderful interactions I have had with people at lunch.

References

[1] *Staniforth, Andrew, and John Thuburn. "Horizontal grids for global weather and climate prediction models: a review." Quarterly Journal of the Royal Meteorological Society 138.662 (2012): 1-26.*
[2] *Syrakos, Alexandros, et al. "A critical analysis of some popular methods for the discretisation of the gradient operator in finite volume methods." Physics of Fluids 29.12 (2017).*
[3] *Barth, Timothy, and Dennis Jespersen. "The design and application of upwind schemes on unstructured meshes." 27th Aerospace sciences meeting. 1989.*
[4] *Tomita, Hirofumi, et al. "Shallow water model on a modified icosahedral geodesic grid by using spring dynamics." Journal of Computational Physics 174.2 (2001): 579-613.*
[5] *Kritsikis, Evaggelos, et al. "Conservative interpolation between general spherical meshes." Geoscientific Model Development 10.1 (2017): 425-431*
[6] *Computational and Information Systems Laboratory. 2023. Derecho: HPE Cray EX System (NCAR Community Computing). Boulder, CO: National Center for Atmospheric Research.* ***doi:10.5065/qx9a-pg09****.*

"This material is based upon work supported by the U.S. National Science Foundation National Center for Atmospheric Research, which is a major facility sponsored by the U.S. National Science Foundation under Cooperative Agreement No. 1755088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. National Science Foundation."