Description and Generation of the HRRR-Zarr AWS Open Data Registry

Alexander A. Jacques[1], Adair Kovac[1], John D. Horel[1], and Taylor Gowan[2]

[1]Department of Atmospheric Sciences, University of Utah [2]DTN

Introduction and Motivations

- NOAA Open Data Dissemination (NODD) Program led to vast improvements in accessibility to real-time and historical Numerical Weather Prediction (NWP) datasets
- Existing formats of output (e.g., grib2, netcdf) less able to fully take advantage of cloud-centric capabilities:
  - May need to download entire file to extract data subset
  - Download many files to extract data across time
- **Primary Initiative:** Explore cloud-optimized storage to improve access time to NWP output from operational High Resolution Rapid Refresh (HRRR) model

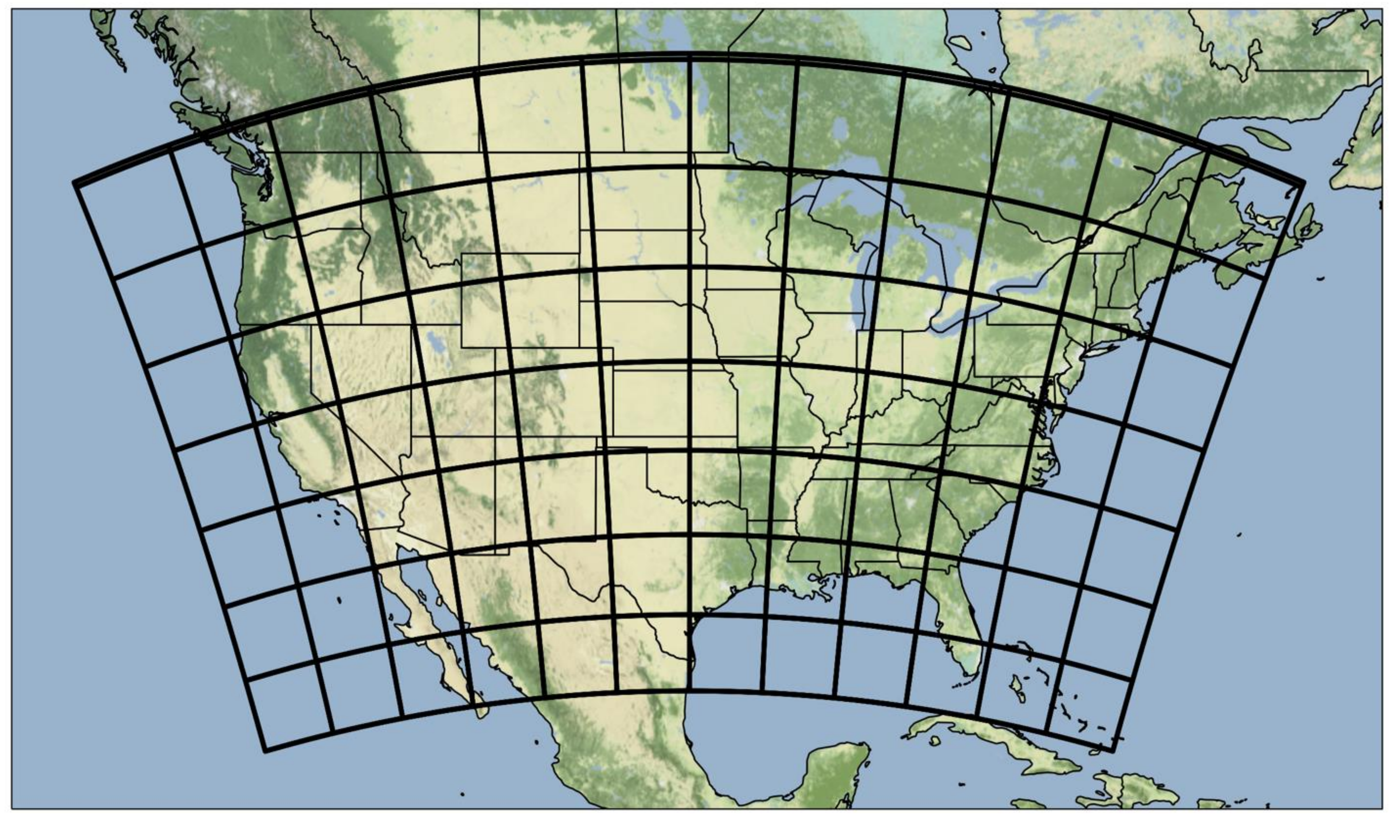
HRRR Zarr Format Description

What is Zarr? - https://zarr.readthedocs.io/

- Storage format designed for efficient compression of chunked N-dimensional arrays of data
- Why use it for this HRRR archive?
  - Sub-sectoring of data spatially into "chunks"
  - Concatenate forecast times: one "chunk" contains all forecast times (F01-F18 or F01-F48 every 6 h)
  - Object-oriented storage in S3 - reduce egress costs by only needing to access necessary data and metadata

HRRR Archive Zarr Structure

Operational CONUS grid per variable (1799 x 1059) broken up into 96 spatial "chunks" of size 150 x 150



Time dimension incorporated via concatenating of 2D arrays:

HRRR Zarr Storage	Forecast Hours Included	Final Array Dimensions
Analysis (**hh**z_anl.zarr)	F00	1 x 150 x 150
Forecast (**hh**z_fcst.zarr)	F01-F18 hourly F01-F48 every 6 h	18 or 48 x 150 x 150

LZ4 Level 9 compression applied to reduce file sizes of individual chunks to ~10 KB (~1 MB) for analysis (forecast) zarr files

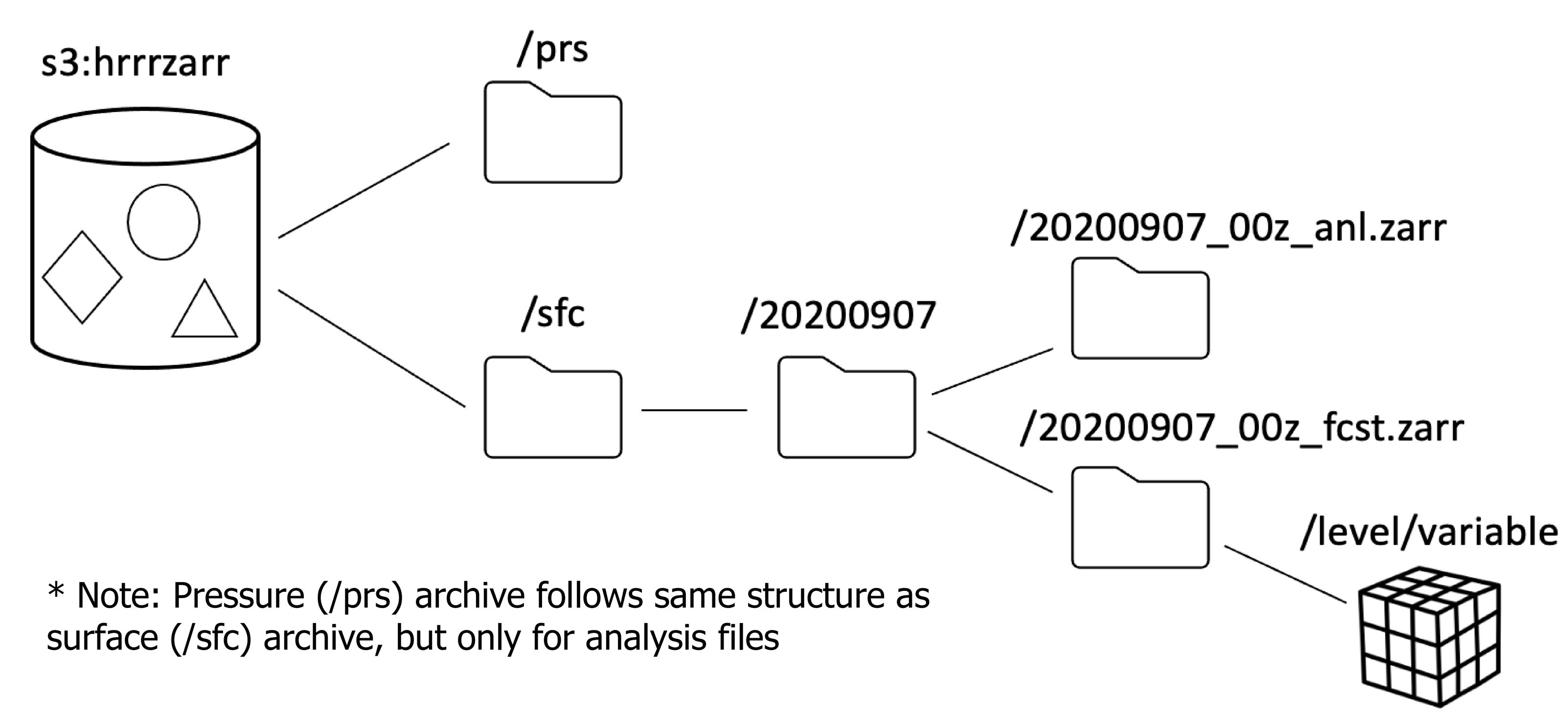
HRRR Zarr AWS S3 Open Registry Archive

AWS Archive Location and Documentation

- HRRR AWS Open Registry: https://registry.opendata.aws/noaa-hrrr-pds/
  - Zarr format link and CLI access under "Resources on AWS"
  - S3 Bucket Explorer: https://hrrrzarr.s3.amazonaws.com/index.html
- Documentation: https://mesowest.utah.edu/html/hrrr/
- Total Size (30 Dec 2022): 201.2 TB (avg growth rate ~140.7 GB/day)

HRRR Operational Version	Surface Analysis Zarr	Surface Forecast Zarr	Pressure Analysis Zarr
V4 (2 Dec 2020 forward)	Available	Available	Available
V3 (12 July 2018 – 1 Dec 2020)	Available	Available	Available
V2 (23 Aug 2016 – 11 July 2018)	Available	Not Available	Not Available

AWS S3 Zarr Structure



Zarr Archive Generation and Usage Statistics

+1:00-1:30 past model initialization time
HRRR grib2 files made available on the AWS Open Registry

→

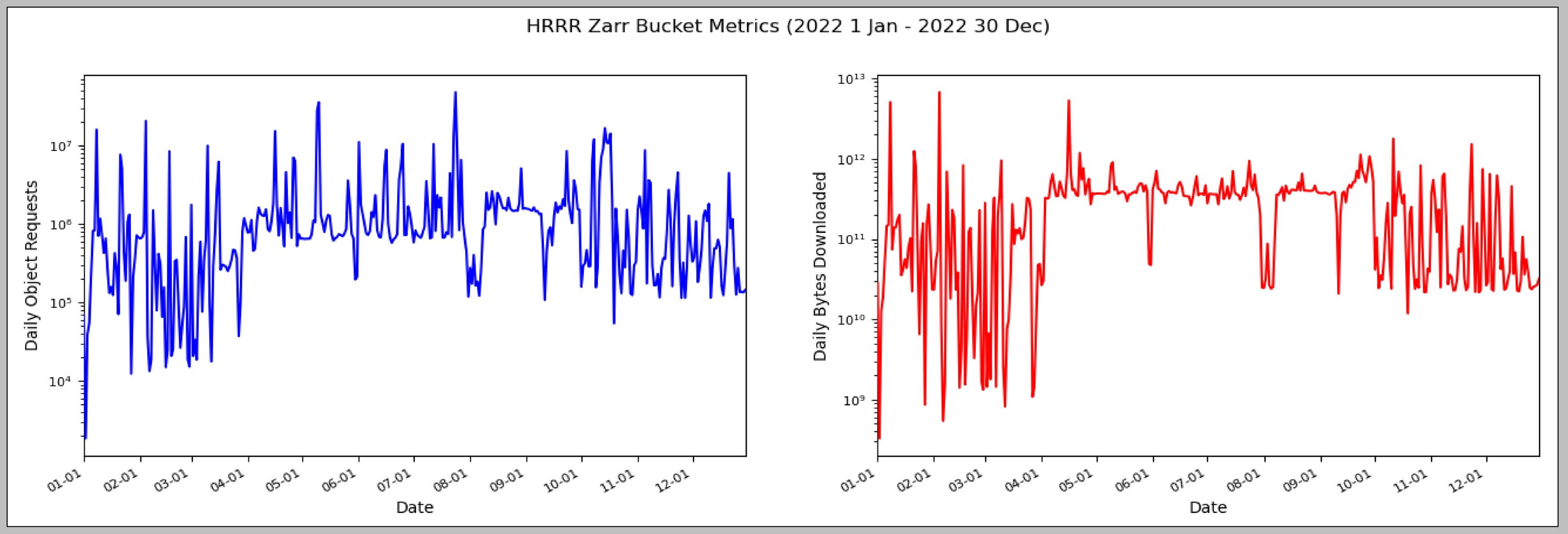
+1:30-1:55 past model initialization time
Grib2 files downloaded to AWS instance for Zarr processing and generation

↓

Zarr Generation and Upload to S3
Sfc Analysis: ~3 minutes
Sfc Forecast: ~5-10 minutes
Prs Analysis: ~20 minutes

←

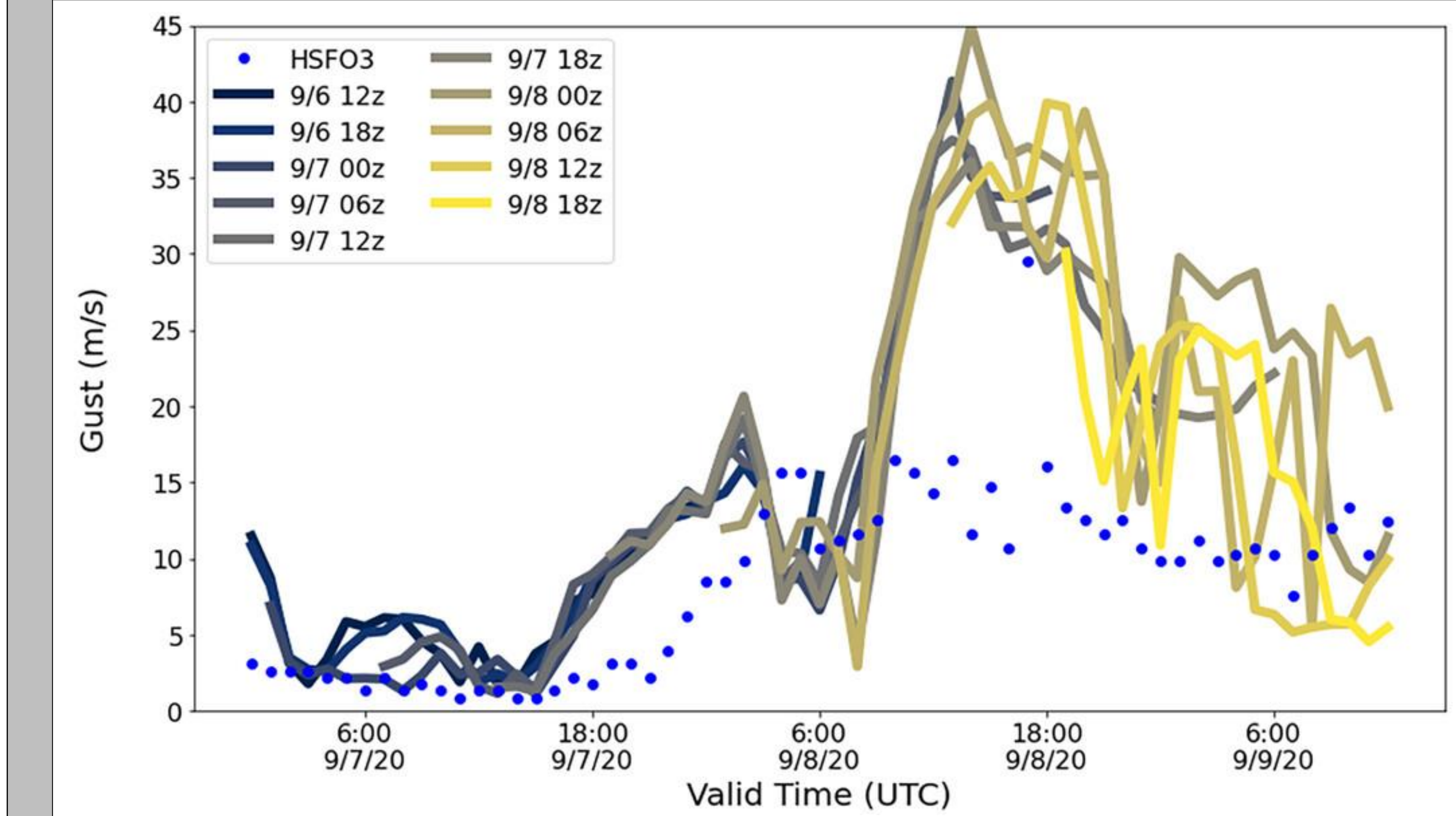
+1:45-2:15 past model initialization time
Zarr formatted files made available on AWS Open Registry



Use Case Example (Gowan et al. 2022 - JTECH)

Western Oregon: Destructive Wind/Wildfires (7-9 Sep 2020)

- Over 1300 km2 burned between Riverside and Beechie Creek fires
- Downstream impacts on air quality in Portland, Salem, and Eugene, OR



Retrospective Time Series Analysis of HRRR Forecasts

- Forecast time series of maximum winds at RAWS weather station location HSFO3 (Horse Creek, OR) show general consistency for 48h HRRR forecasts leading up the event
- Creating this analysis using grib2 data: ~54 GB of data **downloaded**
- Creating analysis using zarr archive: ~10 MB total data **accessed**

Summary and Lessons Learned

- Zarr offers alternative for acquiring NWP output for applications where full gridded dataset isn't needed (e.g., retrospective regional research use cases, long-term machine learning applications, etc.)
- Zarr usage in geosciences still gaining momentum
  - Primary python package for chunk generation (iris-grib) still growing
  - Forecast time concatenation needs to better manage missing/null data
  - Extraction functions (e.g., Python zarr, xarray modules) still developing
- Potential for application to next generation of NWP output from systems such as the Rapid Refresh Forecast System (RRFS)

Acknowledgements

We would like to thank the Amazon Sustainability Data Initiative for their support with hosting the AWS Open Data Registry and for computational credits to generate the archive. We also thank the University of Utah Center for Higher Performance Computing (CHPC) for their support of the Pando storage system, which previously hosted a HRRR historical archive at the University of Utah. For additional information, please see the manuscript Gowan et al. 2022 - https://doi.org/10.1175/JTECH-D-21-0106.1