2022 ARM/ASR Open Science Workshop Abstracts

Virtual Workshop
May 10-13, 2022

ARM's Github Organization Presence

Adam Theisen
https://github.com/ARM-DOE

ARM has recently advanced their Github presence through the addition of two new Github organizations in addition to the existing ARM-DOE organization. The ARM-DOE organization is for ARM sponsored software and open-source repositories. ARM-Synergy was developed as a place for ARM/ASR PI's to house and share code. ARM-Development is a sandbox style area for new idea development for ARM infrastructure and others. Across all these organizations ARM is working to ensure that they follow standard practices and adhere to DOE policies which include vulnerability and antivirus scanning.

Open Science and the ARM Translators: Past, Present and Future.

Scott Collis, Jennifer Comstock, Scott Giangrande, Damao Zhang, Shaocheng Xie, John Shilling and Krista Gaustad
https://www.arm.gov/connect-with-arm/organization/translators

Translators serve a unique role in the U.S. Department of Energy (DOE)'s Atmospheric Radiation Measurement (ARM) Climate Research Facility, providing scientific input through various leadership and service roles, and directing the creation of value-added products (VAPs) and model/observational tools to make ARM measurements accessible to broader sectors of our scientific community. As the ARM facility, its instruments, and associated needs advance with an evolving scientific community, the Translator group also reviews, adapts and publicly presents to the ARM community a coordinated vision for these science needs on a triennial basis (our "Translator Plan") to meet such challenges. The current iteration of this Translator Plan, in addition to covering several aspects of continuing Translator roles, will specifically detail efforts by which Translators will increase ARM involvement in open science and accessibility of ARM's VAPs. This presentation will discuss past, current and planned open-science activities, such as ARM's ADI resources, the Py-ART radar toolkit, and future open-VAP resources.

Open Radar Science in Action

Maxwell Grover
https://projectpythiatutorials.github.io/radar-cookbook

The Open Radar community has been a proponent for open science and open source software since its inception. The Python ARM Radar Toolkit (also known as Py-ART), is just one component of this open radar science ecosystem. The goal of this community/set of packages is to provide a collaborative platform to develop, share and integrate free and open source community software for weather radar data processing. This is in contrast to a closed-ecosystem, where expensive licenses are required for

proprietary software and data is locked away in a place that is difficult to find and access, preventing inclusive open science. A further advantage to this open approach is it provides a conduit for the science community to contribute back to both the open source codebases and, in turn, the entities that support them.

Within this talk, we will provide an overview of open science in action using components of the open radar community, such as PyART, Wradlib, and PyDDA. This will include walkthroughs of how to access data, process data using common workflows, and visualize our analysis results. A recent point of collaboration has been the development of "Radar Data Cookbooks" which not only provide an overview of how to use open source packages for analysis, but also link to foundational python material (ex. Matplotlib, NumPy) required to understand package specific workflows. These cookbooks run on open platforms, such as Binder and the ARM Jupyterhub, providing users an easy entry point to getting started with the open radar community.

Open science beyond data sharing

Markus Petters

Professional societies and funding agencies are moving towards open data policies that require publication of raw data and computer software used to generate results. Our research group has been active in practicing open science, including distribution of data and analysis scripts with publications as well are producing self-contained software packages (https://doi.org/10.5194/amt-2021-51), open-hardware instrumentation (https://doi.org/10.1016/j.ohx.2022.e00266), and open-source education resources (https://doi.org/10.1175/BAMS-D-20-0072.1). These are distributed via GitHub, zenodo archives, and pre-build docker containers hosted on DockerHub. The sharing of data and source code to meet current funding agency requirements is straightforward. Commonly accepted objectives of open science include addressing issues related to replication, fraud, and to promote easy reuse of published datasets. However, additional goals exists. For example, one the goals of open science is to lower the barrier to entry into a particular field. A second goal is to lower cost and avoid duplication of effort. A third goal is to distribute the work at scale to an audience that are neither software nor hardware developers. Sharing code and hardware designs that are consistent with these goals is surprisingly difficult. Here I will present some of these challenges as I perceive them from our group's perspective. These include adopting the appropriate architecture, writing appropriate documentation, packaging the software, and designing the products in a manner that is appropriate for the intended audience. I recommend that funding agencies articulate specific goals of their policies to fully unlock the transformative potential of open science.

Project Pythia: An Open Source Educational Resource for Geoscience Data Analysis

John Clyne
https://projectpythia.org/

Project Pythia is a web-based, community-owned, educational resource whose aim is to help teach geoscientists of all stripes and experience levels how to make effective use of the Scientific Python Ecosystem for the analysis and visualization of their data. With the support of an NSF Earth Cube award, Project Pythia was launched in the fall of 2020. A major milestone was reached in the summer of 2021

when the Project Pythia web site, https://projectpythia.org/, was launched with a broad collection of "foundational content" along with a gallery of curated links to other resources. This presentation will provide an overview of Project Pythia goals, current status and near-term plans, and most importantly will discuss how to get involved with this open science focused, community-owned resource.

Open-source hardware possibilities (and problems) from a manufacturer's perspective

Gavin Robert McMeeking and Ethan Emerson

https://github.com/NOAA-CSL/NOAA_BBB_PRU_DAQ_Cape

Open-source hardware raises intriguing new development pathways for scientific instrumentation and support tools, but also introduces challenges to current models for technology commercialization. We will provide a brief history of our work to commercialize the NOAA developed open hardware instrument, the Printed Optical Particle Spectrometer (POPS), which has been used in several ARM and ASR projects including airborne operations and ground-based deployments. Our focus is on development of the instrument hardware, which uses 3D printed components, and how these have been integrated into the commercial product. We also discuss how our company is using additive manufacturing in other products, and as part of our development cycle, and how a more community-based approach could improve development times and lead to an overall benefit. As part of our presentation, we will also raise and hope to inspire discussions within the broader ASR/ARM community the challenge of maintaining a viable commercial product while also committing to principles of open hardware development. Our hope is to explore new models for instrument development and hardware support beyond the traditional "purchase and/or service" or "build it on my own" models that are responsible for most atmospheric instrumentation in use today.

An adventure in open hardware design: The Open Snowflake Camera for Research and Education (OSCRE)

Aaron Kennedy

https://github.com/KennedyClouds/OSCRE

This presentation will serve as a candid discussion on the Open Snowflake Camera for Research and Education (OSCRE), an open and affordable DIY hydrometeor imager. Capabilities will be demonstrated with data collected during snowfall and blizzard events in North Dakota over the past few seasons. Planned improvements will be discussed, and its deployment during the 2022-2023 winter at the Surface Atmosphere Integrated Field Laboratory (SAIL) will be detailed. The latter half of the presentation will discuss the challenges of creating open hardware in a pandemic world and broad thoughts on the role of open instrumentation in science. It is hoped this talk will spur conversation on these topics and the resources needed to build communities around such instruments.

SAGE: Open Cyberinfrastructure for the Nation

Scott Collis, Jim Olds, William Miller, Aaron Packman, Pete Beckman, Rajesh Sankaran, Nicola Ferrier, Sean Shahkarami, Seongha Park, Robert Jackson, Yongho Kim, Joseph Swantek, Dario Dematties Rayes, Wolfgang Gerlach, Niel Conrad and Sergey Shemyakin, Charles Catlett

https://sagecontinuum.org/

Measurements no longer exist without computing. Even measurements manually entered into a website or spreadsheet require computing. Earth system measurements, due to the value of remote observations, require control and logging on-site. Sage provides an open framework to unleash the power of edge computing as a national-scale cyberinfrastructure. Sage touches all components of the open science landscape from open source code, open edge computing code, and the open hardware platform of Waggle. This presentation will describe Sage and its goals, showcase some early successes such as the NEON Mobile Deployment Platform COntrolled Burn Experiment (MDP-COBE) and demonstrate, through live coding in a Jupyter notebook, how anyone can access and participate in open science using Sage data.

Interactive Exercise: Search for YOUR data on the ARM data discovery portal with a metadata expert

Maggie Davis
https://adc.arm.gov/discovery/

In this interactive exercise, ARM staff will provide new data users an overview of the ARM data discovery and will show participants how to find ARM observational data for their research and take advantage of other ARM facility resources. Metadata managers at the ARM Data Center at ORNL provide the "what, where, and when" about ARM's 2 PB of data to assist users in finding the best data for their needs. This session will familiarize new ARM users with this accurate and abundant metadata, the classification of data files and variables within the files, as well as the value of increased details about data that are displayed in the ARM Data Discovery Tool. Users will leave this session with an understanding of how to search for and download the data that they want to use in future sessions of this workshop.

An Earth System Model Aerosol-Cloud Diagnostics Package (ESMAC Diags) to evaluate E3SM predicted aerosols and clouds using ARM data

Shuaiqi Tang
https://github.com/eagles-project/ESMAC_diags

An Earth System Model (ESM) aerosol-cloud diagnostics package is developed to facilitate the routine evaluation of aerosols, clouds and aerosol-cloud interactions simulated by the Department of Energy's (DOE) Energy Exascale Earth System Model (E3SM). Currently ESMAC Diags focuses on four geographical regions: Eastern North Atlantic (ENA), Central U.S. (CUS), Northeastern Pacific (NEP) and Southern Ocean (SO), where frequent liquid clouds exist and extensive in-situ and remote-sensing ARM measurements are available. Various types of diagnostics and metrics are performed for aerosol number, size and composition properties, cloud microphysical and optical properties, as well as the relationships between aerosols and clouds. The diagnostics package is coded and organized in a way that can be easily extended to other field campaign datasets and adapted to higher-resolution model simulations.

Using the Earth Model Column Collaboratory (EMC²) open-source ground-based instrument simulator and subcolumn generator to facilitate direct comparisons between observations and models

Israel Silber

https://github.com/columncolab/EMC2

Climate models are essential for our comprehensive understanding of Earth's atmosphere and can provide critical insights into future changes decades ahead. Because of these critical roles, today's climate models are continuously being developed and evaluated using constraining observations and measurements obtained by satellites, airborne, and ground-based instruments. Instrument simulators can provide a bridge between the measured or retrieved quantities and their sampling in models and field observations while considering instrument sensitivity limitations. Here we present the Earth Model Column Collaboratory (EMC²), an open-source ground-based lidar and radar instrument simulator and subcolumn generator, specifically designed for large-scale models, in particular climate models, but also applicable to high-resolution model output. EMC² provides a flexible framework enabling direct comparison of model output with ground-based observations, including the generation of subcolumns that may statistically represent finer model spatial resolutions. In addition, EMC² emulates ground-based (and air- or space-borne) measurements while remaining faithful to large-scale models' physical assumptions implemented in their cloud or radiation schemes. The simulator uses either single particle or bulk particle size distribution lookup tables, depending on the selected scheme approach, to perform the forward calculations. To facilitate model evaluation, EMC² also includes three hydrometeor classification methods, namely, radar- and sounding-based cloud and precipitation detection and classification, lidar-based phase classification, and a Cloud Feedback Model Intercomparison Project Observational Simulator Package (COSP) lidar simulator emulator. The software is written in Python, is easy to use, and can be straightforwardly customized for different models, radars, and lidars. In this tutorial, the logic, functionality, features, and software structure of EMC² are briefly described, followed by a simple demonstration of utilizing EMC² to emulate variables measured by certain U.S. Department of Energy (DOE) Atmospheric Radiation Measurement (ARM) radars and lidars using output from DOE's Energy Exascale Earth System Model (E3SM).

Using Satpy-based tools for easy meteorological satellite processing

David Hoese

https://satpy.readthedocs.io/en/stable/

The amount of data sent to earth by meteorological satellites increases with each new satellite generation. These data are used by scientists, researchers, the interested public, and are assimilated to forecasting models, but become more difficult to use as technology allows for higher spatial and spectral resolutions. These datasets are usually provided to users in new file formats with multiple wavelengths, levels of calibration, polarization, and resolutions which can require new complicated software to read. Additionally, scientific analysis typically requires compositing various channels, reprojecting or interpolating, and writing to various file formats to suit the needs of the user. The python library Satpy was created for anyone who wants to work with satellite data and do it quickly and easily. Satpy is developed by a group of scientists and developers from around the world known as Pytroll; a group that has created and maintained open source science software for more than 12 years. Satpy is able to read over 30 different satellite data file formats from geostationary and polar-orbiting satellites and from

various data sources. It provides multiple resampling algorithms, various RGB recipes that work across multiple sensors, and can write multiple image and data formats including GeoTIFF and NetCDF. Satpy provides common interfaces and data structures regardless of what data is being worked with and benefits from the dimension and coordinate handling of the Xarray python library. By leveraging the parallelization made available in the Dask library, Satpy is able to quickly process large arrays of satellite data that would normally not be possible on traditional user workstations. Satpy can be used as an interactive research tool or used as a stable operational processing library as many meteorological agencies and organizations are doing now. This talk will provide a basic overview of Satpy's features and how Satpy has been used in other tools and workflows. Other tools that use Satpy will also be discussed including the command line Polar2Grid and Geo2Grid projects that wrap Satpy and the SIFT graphical user interface for visualization data loaded with Satpy. By providing all of this functionality and generating products in a free and open source software package we hope Satpy will become the go-to library for working with satellite data.

PySP2: An open source Python package for processing Single Particle Soot Photometer data

Robert Jackson

https://github.com/ARM-DOE/PySP2

The Single Particle Soot Photometer (SP2) is an instrument that detects the scattering and incandescence signals from black carbon-containing particles that can be used to derived the mass loadings of these particles along with their number and mass size distributions. The SP2 has been deployed in numerous ARM field operations in order to provide a particle-by-particle database of BC measurements. However, processing of this data for ARM has been complicated by the fact that the SP2 produces such large volumes of data (e.g., 10s of GBs per day) that distributing computing is required to process it. The manufacturer's processing software provided is not scalable to distributed computing, adaptable to Linux architectures, nor is it open source for wider use by the community.

Therefore, motivated by these logistic and technical issues ARM has pursued the development of open source Python package for processing SP2 datasets - PySP2. This new Python package currently supports the same primary capabilities of the manufacturer's processing software: processing BC number and mass distributions with artifact filtering and analysis of individual particles. Since PySP2 is written in Python and usable on Linux, it can be combined with dask in order to process large amounts of data in parallel. For example, PySP2 was able to provide BC mass and number distributions for the NSA site and MARCUS field project in about 6 hours for NSA and 2 days for MARCUS. This process would have taken multiple weeks with the manufacturer's software. Therefore, the reduced processing time and increased adaptability enables routine processing of SP2 data for ARM field experiments. To this regard, PySP2 is currently used to process SP2 data for SAIL and TRACER as a part of the aossp2bc60s.b1 ingest. This therefore demonstrates how open software can help provide increased access to quality aerosol datasets for the scientific community.

Radar Tracking and Quality (RadTraQ)

Adam Theisen and Kenneth Kehoe

https://github.com/ARM-Development/RadTraQ

Radar Tracking and Quality is a python library to house functions related to assessing and monitoring the quality of a radars status and calibration. This repository is still in development but currently houses functions for corner reflector analysis, CFADs, cloud masking and comparisons, Zdr bias calculations, and more. The goal is to build this toolkit out as a resource for ARM and the broader community when it comes to determine the status of a radar system based on the data. The ARM Data Quality Office has integrated this into their processing so any contributions by the community can impact ARM monitoring and assessment.

Docker Takeaways

William Roberts

https://github.com/wroberts4/docker_take_aways

Creating, managing, and deploying software in containers is the future. Docker can make it easy to containerize both existing and new products. In this talk, we will focus on basic tricks, takeaways, and "gotchas" revolving around Docker containers that will make development and deployment easier. Here are a few exciting aspects to Docker that we will cover: 1) Use a wide range of pre-built tools/environments with only one command that works and behaves the same on almost any system that has docker installed: Jupyter, numpy, java, databases, nginx servers, etc. 2) Containers are the building blocks of running applications in the cloud. If your code is containerized, it (most likely) can run in the cloud! 3) Version control and deployment becomes as easy as just pushing the image to dockerhub or another online registry! 4) Controlling resources such as CPU and memory can be controlled with command line flags. 5) Create multi-stage deployment to protect secrets and reduce software size.

MetPy: A Community-driven Python Toolkit for Meteorology and Atmospheric Science

Ryan May

https://github.com/Unidata/MetPy

MetPy is an open-source Python package for meteorological and atmospheric science applications, leveraging significantly many other pieces of the scientific Python stack (e.g. numpy, matplotlib, scipy, etc.). Its goal is to provide tested, reusable components suitable to a wide array of tasks, including scripted data visualization and analysis. The guiding principle is to make MetPy easy to use with any dataset that can be read into Python. MetPy's general functionality breaks down into: reading data, meteorological calculations, interpolation, and meteorology-specific plotting. MetPy also has significant integration with XArray, as well as extended support for interpreting netCDF Climate and Forecasting Convention metadata.

MetPy's development takes place entirely using the GitHub collaborative development platform, making extensive use of issues, discussions, and the Pull Request feature. This allows MetPy's community of users to submit bug reports, feature suggestions, and even make contributions for documentation, examples, and code. This talk presents an overview of the functionality present within MetPy and also how MetPy's community gets involved and contributes to the project.

Atmospheric data Community Toolkit

Adam Theisen, Ken Kehoe, Zach Sherman, Bobby Jackson, Alyssa Sockol, Corey Godine, Max Grover, Jason Hemedinger, Jenni Kyrouac, Maxwell Levin and Michael Giansiracusa
https://github.com/ARM-DOE/ACT

The Atmospheric data Community Toolkit (ACT) is an open source Python toolkit for working with atmospheric time-series datasets of varying dimensions. The toolkit has functions for every part of the scientific process; discovery, IO, quality control, corrections, retrievals, visualization, and analysis. It is a community platform for sharing code with the goal of reducing duplication of effort and better connecting the science community with programs such as the Atmospheric Radiation Measurement (ARM) User Facility. This talk will provide a broad overview of ACT, how it's integrated into ARM, and how the user community can contribute.

Reproducible Experiment of Species Distribution Modeling in the Amazon Basin

Renato Okabayashi Miyaji; Felipe Valencia de Almeida; Pedro Luiz Pizzigatti Corrêa; Luciana Varanda Rizzo
https://github.com/amazon-bioclim/reproducible-sdm-amazon

The Amazon is one of the largest Rainforests in the world. It hosts more than 20,000 different species, representing a relevant portion of the world's biodiversity. Manaus, a city located in the central region of the Amazon Basin, has been growing rapidly. Nevertheless, only a few studies have been conducted with the objective of understanding the changes due to anthropic action. In this context, here we present a Virtual Research Environment (VRE) focused on bioclimatic analysis that allows users to perform Species Distribution Modeling (SDM) experiments using Machine Learning algorithms, aiming to determine the influence of pollutants and the anthropic action on the local biodiversity.

The VRE is a script and a Jupyter Notebook written in Python. Its workflow is based on three major steps. Initially, a dataset from the Atmospheric Radiation Measurement (ARM), which contains meteorological and aerosol data, is imported. Then, a linear spatial interpolation is performed, in order to generate a grid with high resolution of the data over the region of interest. A dataset from the Global Biodiversity Information Facility (GBIF), which contains data about global biodiversity, is also imported, and a join operation, based on the keys of latitude, longitude, and date, is performed to obtain a unique dataset of bioclimatic data.

In the next step, the user can choose the species to be analyzed and the predictive variables to be used. In the final step, a Machine Learning algorithm – Logistic Regression – is fitted and its hyperparameters are automatically tuned. Then, it is used to generate data visualizations that allow the user to analyze the performance of the Machine Learning model and evaluate the spatial variability of the probability of occurrence of the species along the region of analysis.