Building Trust in Earth Science Findings through Data Traceability and Results Explainability

Paula Olaya*, Ricardo Llamas↟, Rodrigo Vargas↟,
Jay Lofstead†, and **Michela Taufer***

*University of Tennessee, Knoxville; ↟University of Delaware; †Sandia National Lab

Salishan Conference on High Speed Computing, 2023

Complexity in scientific workflows

Scientific workflows growing more complex:

- they integrate AI/ML methods with limited transparency;
- they are composable, including different modules; and
- they run on increasingly heterogeneous systems

For scientists who use these workflows to study scientific phenomena, trusting data, methods, software, and hardware becomes necessary!

Trusting scientific workflows

Scientists achieve trust in their findings by:

- Tracing back data lineage
- Explaining computational methods and output through record trials
- Preserving intermediate data

Trust enables the **reusability** of workflow components and the **composability** of complex workflows.

Modeling containerized workflows

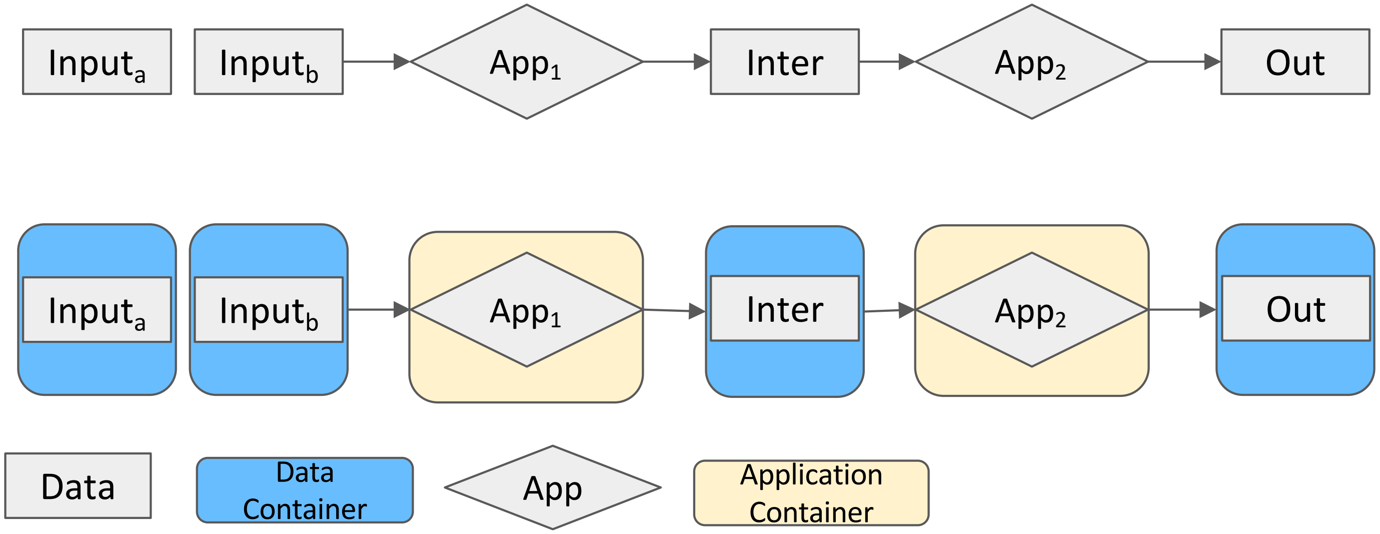
- We propose a computational environment that seamlessly builds on top of container technology
- We use a fine-grained approach to encapsulate each workflow component into its own independent container

Native

Original execution in an HPC or a cloud cluster

Fine-grained Workflow in Container

Decoupled workflow. Every workflow component (data and apps) as a single service

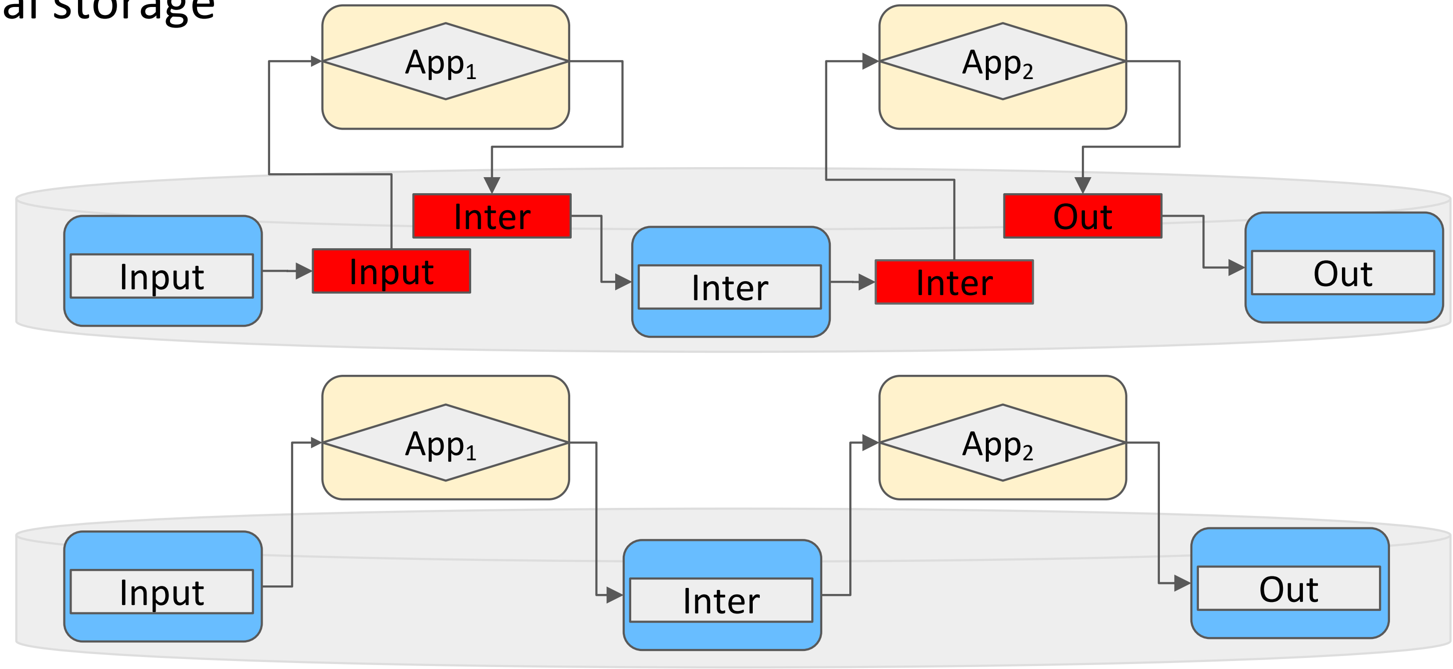


Connecting containerized workflows

- We connect our containers using a **zero-copy data transfer** approach. This allows containers to directly exchange data without creating extra copies or using external storage

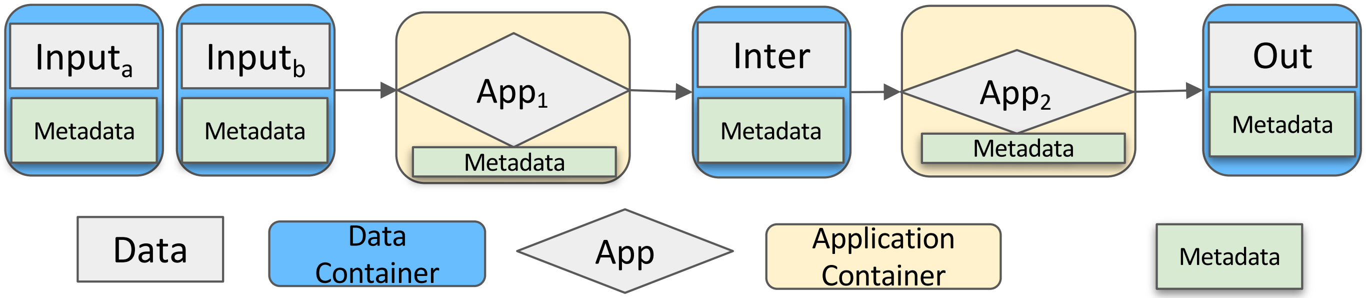
Two-copy data transfer in fine-grained containerization

Zero-copy data transfer in fine-grained containerization



Annotating containerized workflows

- Each container captures local metadata
- The metadata includes:
  - **Container identification [UUID, name]:** the unique identification of each component
  - **Creation time:** the point in time when the container is written to disk
  - **Command line:** the set of instructions to execute the application
  - **Container record trail {[UUID, name]}:** the pipeline of containers used to generate a new result
- The metadata enables building the in-depth **data lineage** and the complete **record trail** of the applications generating the results



Putting everything together using Singularity

Data container:

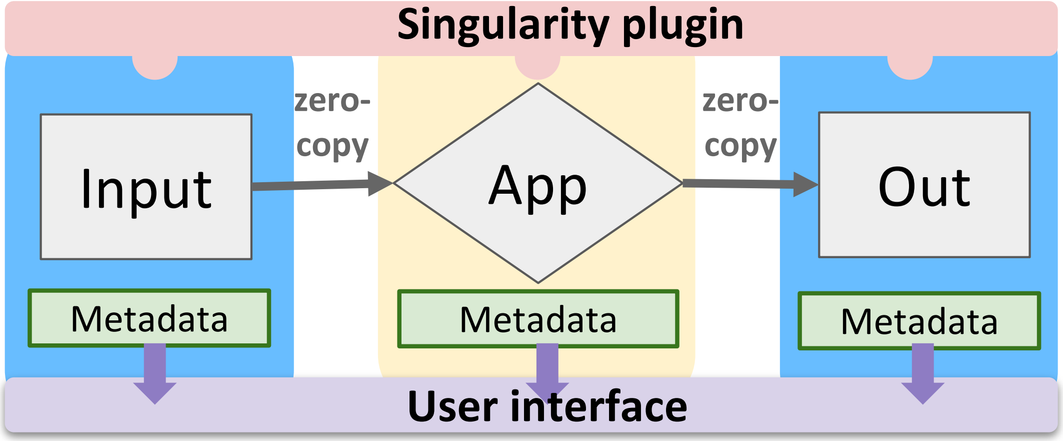
- Data compressed and added as an EXT3 file systems partition
- Metadata added as a JSON

Application container:

- Application executable or scripts + software stack compressed in a squashFS partition
- Metadata added as a JSON partition

Zero-copy data transfer:

- Bind mount functionality that links a directory from a source container to a directory in a destination container
- Multiple containers can be connected



Singularity plugin:

- Software package that interacts with the containerized environment to generate the metadata

User interface

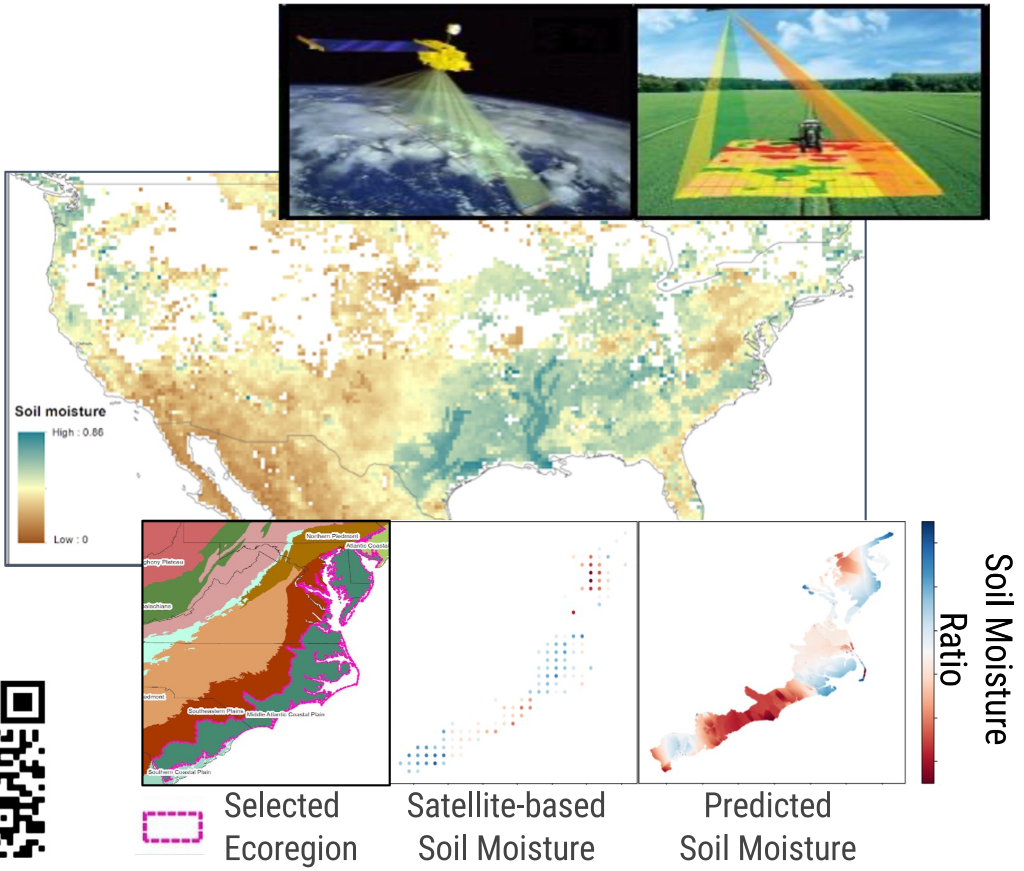
- Interface that facilitates the study of the metadata

Exemplary ML-based scientific workflow

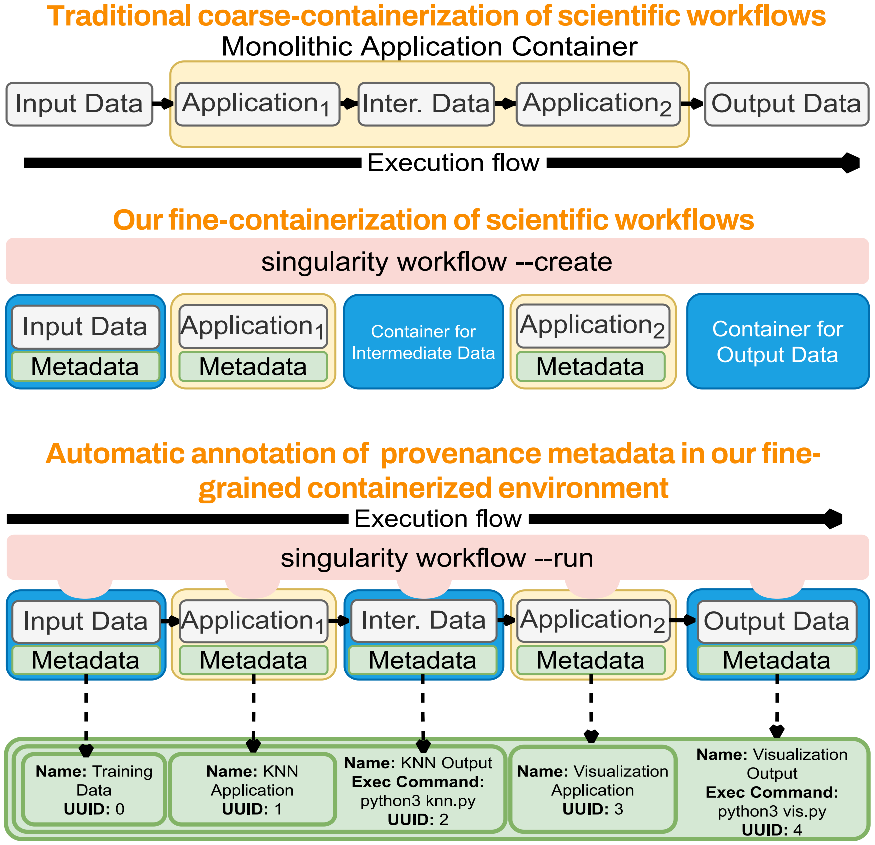
We demonstrate the capabilities of our environment with the study of SOMOSPIE, an earth science workflow

SOMOSPIE uses a suite of ML modeling techniques to predict soil moisture values from the 27 km resolution satellite data down to higher resolutions necessary for policy making and precision agriculture

SOMOSπ



SOMOSPIE in a fine-grained containerized environment



Traditional coarse-containerization of scientific workflows

Our fine-containerization of scientific workflows

singularity workflow --create
Input Data
Metadata
Application$_1$
Metadata
Container for Intermediate Data
Application$_2$
Metadata
Container for Output Data

Automatic annotation of provenance metadata in our fine-grained containerized environment

Name: Training Data **UUID:** 0

Name: KNN Application **UUID:** 1

Name: KNN Output **Exec Command:** python3 knn.py **UUID:** 2

Name: Visualization Application **UUID:** 3

Name: Visualization Output **Exec Command:** python3 vis.py **UUID:** 4

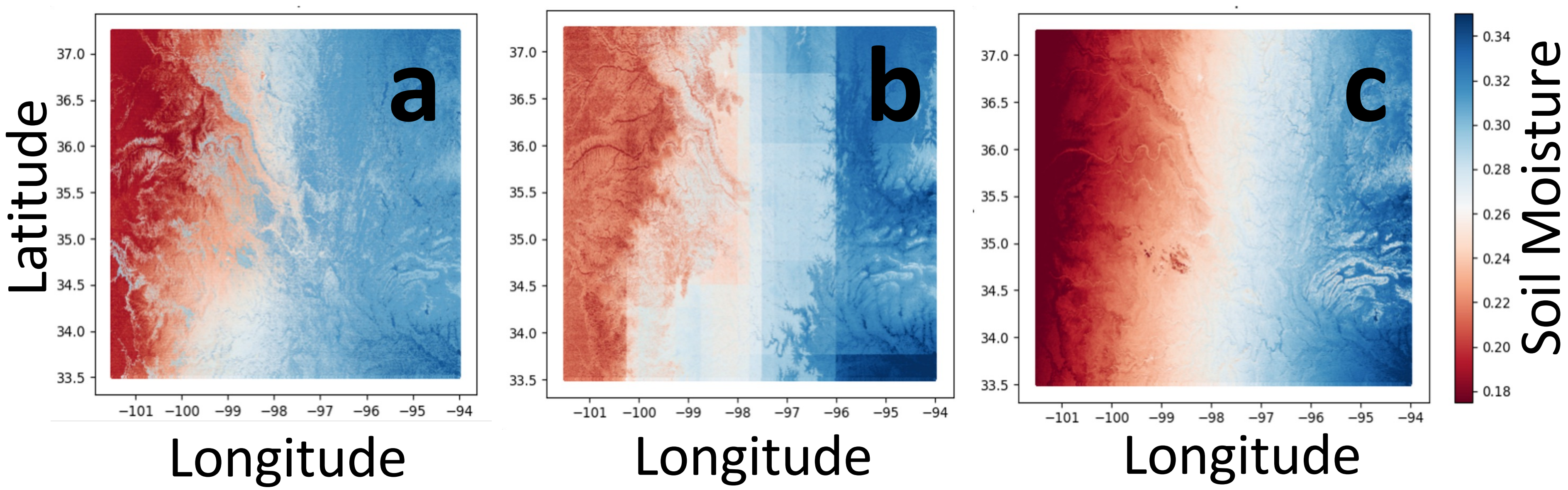
How does the fine-grained containerized environment enable scientist to trust SOMOSPIE findings ?

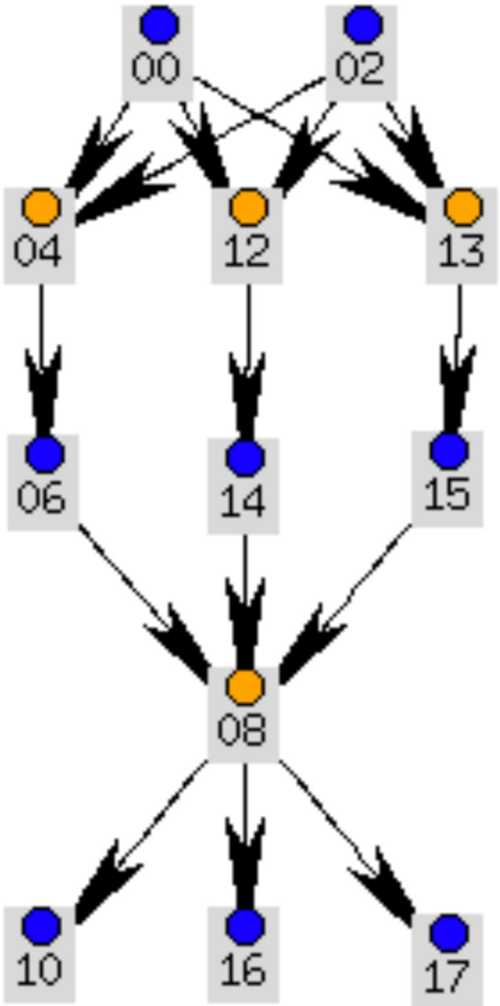
Traceability of data
(data lineage and transformations)

Explainability of results
(computational methods)

Tracing soil moisture data

Can our fine-grained containerized environment enable scientists to trace and explain the results from Figures a, b and c?





UUID	00
Container_name	train_27km.sif
Creation_time	2021-09-01T12:07:26-4
Command_line	noop
Record_trail	Out: [00, train_27km.sif]

UUID	02
Container_name	eval_250m.sif
Creation_time	2021-09-09T13:40:02-4
Command_line	noop
Record_trail	Out: [02, eval_250m.sif]

UUID	04
Container_name	knn.sif
Creation_time	2021-09-07T12:14:54-EDT
Command_line	noop
Record_trail	NULL

UUID	12
Container_name	rf.sif
Creation_time	2021-09-09T21:08:55-EDT
Command_line	noop
Record_trail	NULL

UUID	13
Container_name	sbm.sif
Creation_time	2021-09-07T10:34:56-EDT
Command_line	noop
Record_trail	NULL

UUID	06
Container_name	predictions_oklahoma.sif
Creation_time	2021-09-09T13:40:06-4
Command_line	python3 knn.py Train/train.csv Eval/eval_250m.csv Predictions/predictions_oklahoma.csv
Record_trail	Out: [06, predictions_oklahoma.sif]
	App: [04, knn.sif]
	In: [02, eval_250m.sif]
	In: [00, train_27km.sif]

UUID	14
Container_name	predictions_oklahoma.sif
Creation_time	2021-09-09T22:37:47-4
Command_line	python3 rf.py Train/train.csv Eval/eval_250m.csv Predictions/predictions_oklahoma.csv
Record_trail	Out: [14, predictions_oklahoma.sif]
	App: [12, rf.sif]
	In: [02, eval_250m.sif]
	In: [00, train_27km.sif]

UUID	15
Container_name	predictions_oklahoma.sif
Creation_time	2021-09-08T08:22:50-4
Command_line	python3 sbm.py Train/train.csv Eval/eval_250m.csv Predictions/predictions_oklahoma.csv
Record_trail	Out: [15, predictions_oklahoma.sif]
	App: [13, sbm.sif]
	In: [02, eval_250m.sif]
	In: [00, train_27km.sif]

UUID	08
Container_name	visualization.sif
Creation_time	2021-09-11T04:29:08-EDT
Command_line	noop
Record_trail	NULL

a

UUID	10
Container_name	output_oklahoma.sif
Creation_time	2021-09-11T04:34:17-4
Command_line	python3 visualization.py Predictions/predictions_oklahoma.csv Output/out_oklahoma.png 0.175 0.35
Record_trail	Out: [10, output_oklahoma.sif]
	App: [08, visualization.sif]
	In: [06, predictions_oklahoma.sif]

b

UUID	16
Container_name	output_oklahoma.sif
Creation_time	2021-09-11T06:08:37-4
Command_line	python3 visualization.py Predictions/predictions_oklahoma.csv Output/out_oklahoma.png 0.175 0.35
Record_trail	Out: [16, output_oklahoma.sif]
	App: [08, visualization.sif]
	In: [14, predictions_oklahoma.sif]

c

UUID	17
Container_name	output_oklahoma.sif
Creation_time	2021-09-11T06:08:37-4
Command_line	python3 visualization.py Predictions/predictions_oklahoma.csv Output/out_oklahoma.png 0.175 0.35
Record_trail	Out: [17, output_oklahoma.sif]
	App: [08, visualization.sif]
	In: [15, predictions_oklahoma.sif]

Takeaways

As **scientific workflows** become **more complex and integrate AI**, tracing data provenance and explaining results become harder and more urgent to achieve

We **leverage container technology** to automatically annotate data transformation and creates a workflow execution record trail, enabling data provenance and results explainability

Containerization supports **trust to the scientific results**, easy retrieval for reusability, reproducibility, and portability of the workflow

Check our work

Thank you to:

TPDS paper

10.1109/TPDS.2022.3220539

Our environment

github.com/TauferLab/ContainerizedEnv