OAK RIDGE
National Laboratory

Strategies for querying large-scale scientific data

Ana Gainaru, Liz Dulac, Greg Eisenhauer, Norbert Podhorszki, Scott Klasky

17th Scheduling for large-scale systems workshop, Aussois, France, June 24 - 27, 2024

ORNL is managed by UT-Battelle LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

gainarua@ornl.gov

Queries for scientific data

- High performance I/O abstraction to allow for on-line/off-line memory/file data subscription service
  - Can deal with TB/s
  - In-situ/post mortem analysis
  - Operators

Application	Nodes/GPUs	Data Size per step	I/O speed
SPECFEM3D	3200/19200	250 TB	~2 TB/sec
GTC	512/3072	2.6 TB	~2 TB/sec
XGC	512/3072	64 TB	1.2 TB/sec
LAMMPS	512/3072	457 GB	1 TB/sec

- Why queries?
  - Visualize in real time
  - Analyze on scientists' laptops
  - Remote access
  - AI

Contributors

Oak Ridge National Laboratory

kitware

Sandia National Laboratories

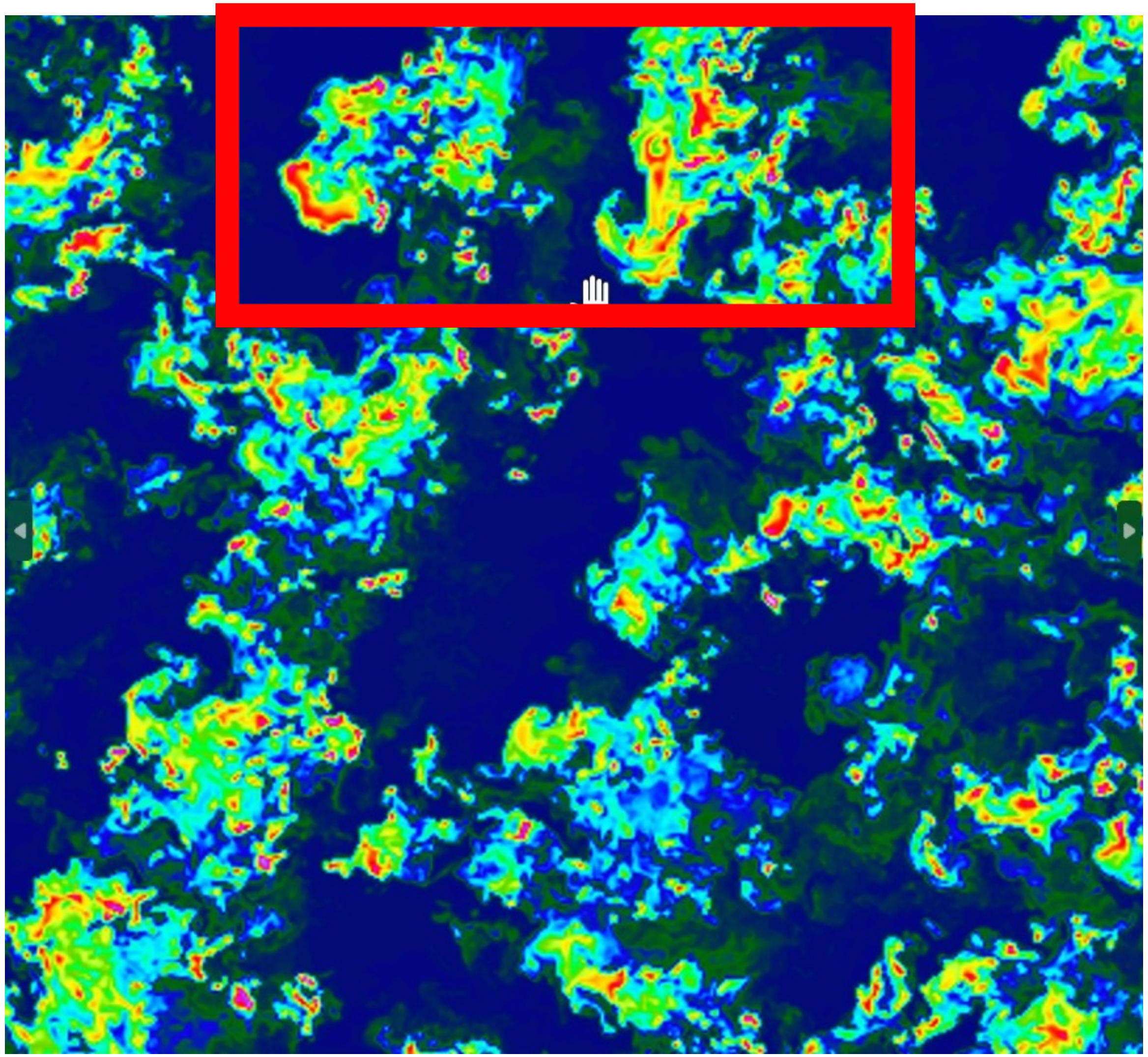
Georgia Tech

Illinois Institute of Technology

Typical query example

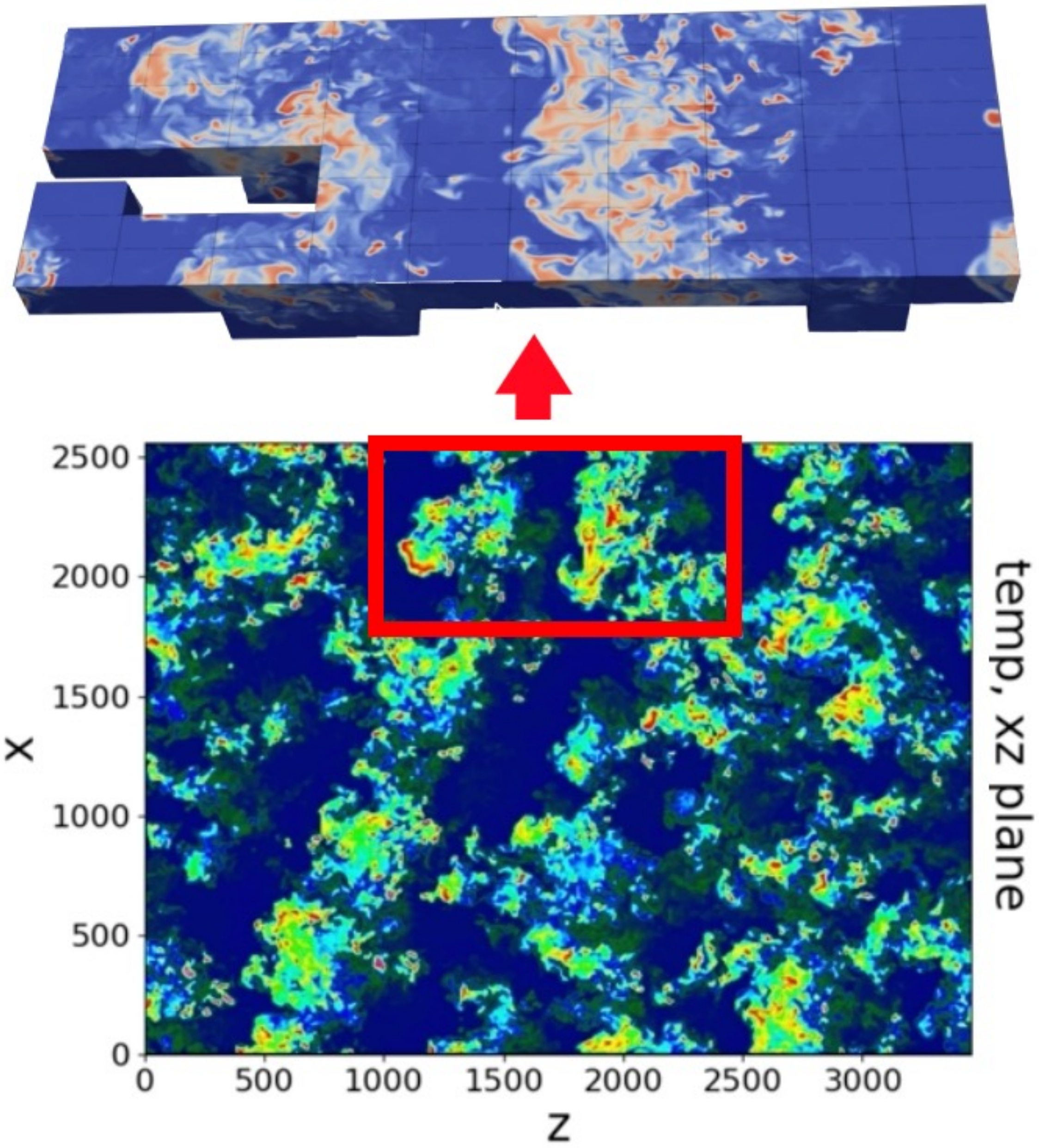
- S3D combustion data
  - 1.5 TB per step (including temp, velocity, species information, etc)
  - Visualize 2D slices of temperature
    - Identify areas of interest

- Implementation
  - Compute stats for blocks of data



Complex queries

- Queries on derived data
  - Quantities of interest that are not directly generated by the application
    - Magnitude of the velocity
    - Magnitude of the curl of the velocity
  - Needed for visualizing or training digital twins using quantities of interest



Current solutions for derived variables

- Writer side solutions
  - Workflows include analysis codes running with applications computing and storing the required derived data
- Reader side solutions
  - Visualization/analysis technology capable of computing derived variables on the fly (e.g. Paraview)
- **Offload this task to the I/O library**
  - Choose for the application the best strategy for computing the derived variables

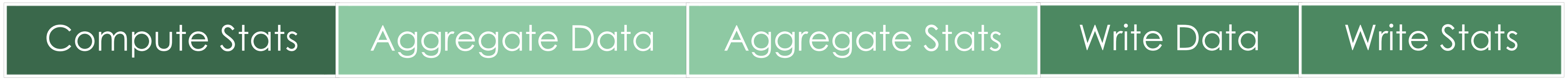
```
IO::CreateDerived(”Magnitude", velocityData);

for (i=0; i < simulationLoops; i++)
{
    // Compute new values for velocityData;
    IO::WriteToStorage(velocityData);
}
```

Offloading the computation to the I/O library

```
IO::WriteToStorage(anyData);
```

Write



Compute Stats

Aggregate Data

Aggregate Stats

Write Data

Write Stats

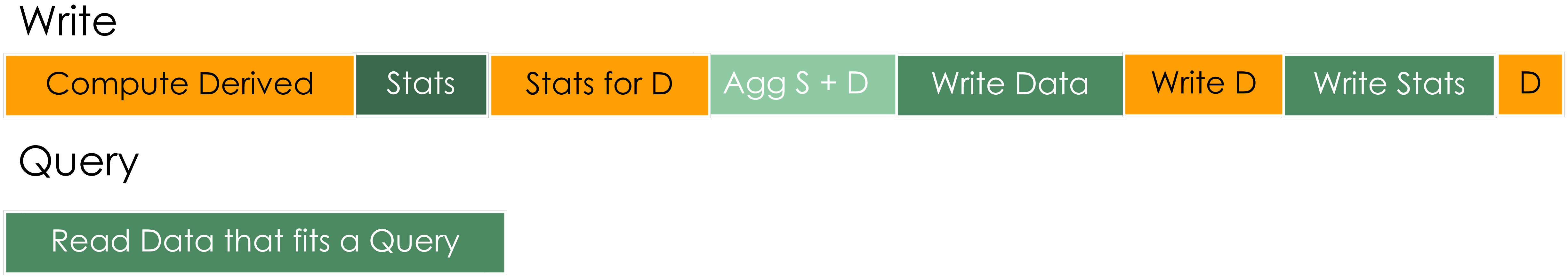
Query

Read Data that fits a Query

- **Normal behavior** (simple queries)
  - Trade-off between granularity of the stats and read size

Offloading the computation to the I/O library

```
IO::CreateDerived(”Magnitude", velocityData);
IO::WriteToStorage(anyData);
```



Write

Compute Derived | Stats | Stats for D | Agg S + D | Write Data | Write D | Write Stats | D

Query

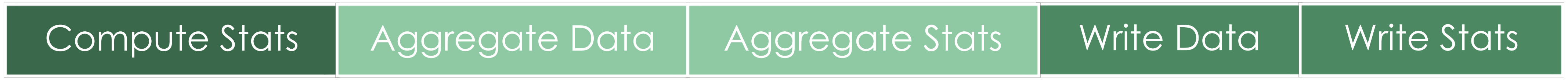
Read Data that fits a Query

- **Store** (Writer side solution)
  - The derived variables are computed during the write operation and both statistics and data for derived variables are stored on storage

Offloading the computation to the I/O library

```
IO::CreateDerived(”Magnitude", velocityData);
IO::WriteToStorage(anyData);
```

Write



Compute Stats

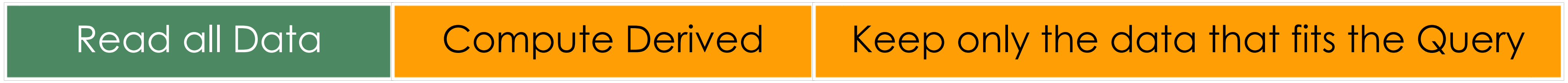
Aggregate Data

Aggregate Stats

Write Data

Write Stats

Query



Read all Data

Compute Derived

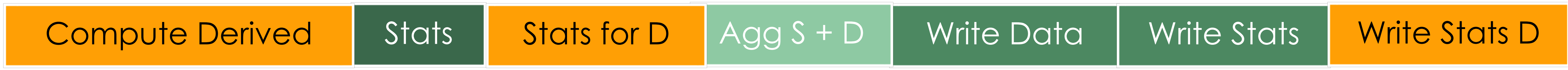
Keep only the data that fits the Query

- **Expression** (Reader side solution)
  - Only the math expression is being saved during the write operation and the derived variables are computed during the read operation.

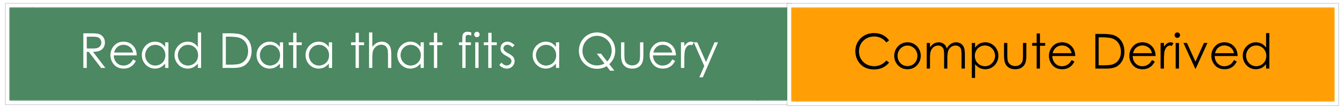
Offloading the computation to the I/O library

```
IO::CreateDerived("Magnitude", velocityData);
IO::WriteToStorage(anyData);
```

Write



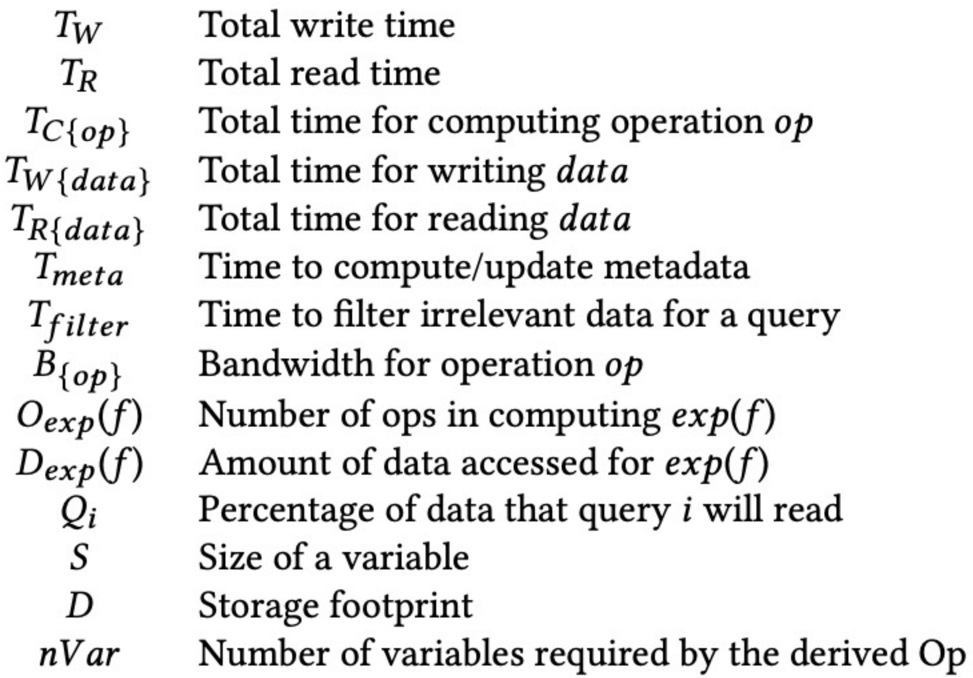
Query



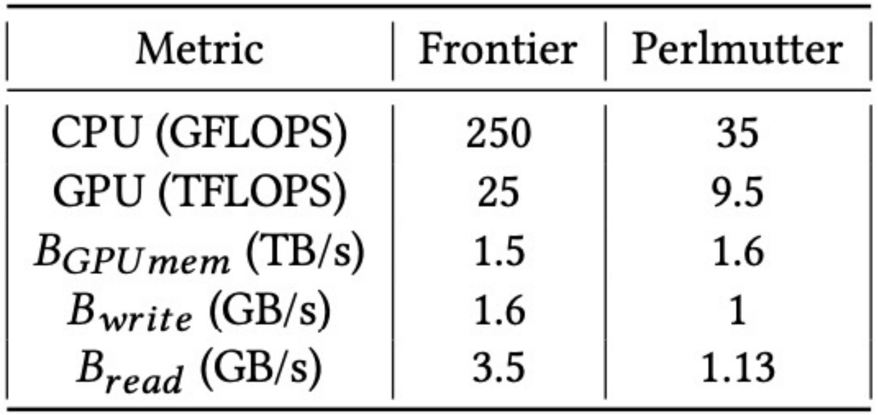
- **Stats** (Hybrid solution)
  - Derived variables are computed during the write operation and only metadata is stored

Trade-off between strategies

- What are the most important factors?



T_W	Total write time
T_R	Total read time
$T_{C\{op\}}$	Total time for computing operation *op*
$T_{W\{data\}}$	Total time for writing *data*
$T_{R\{data\}}$	Total time for reading *data*
T_{meta}	Time to compute/update metadata
T_{filter}	Time to filter irrelevant data for a query
$B_{\{op\}}$	Bandwidth for operation *op*
$O_{exp}(f)$	Number of ops in computing $exp(f)$
$D_{exp}(f)$	Amount of data accessed for $exp(f)$
Q_i	Percentage of data that query i will read
S	Size of a variable
D	Storage footprint
$nVar$	Number of variables required by the derived Op

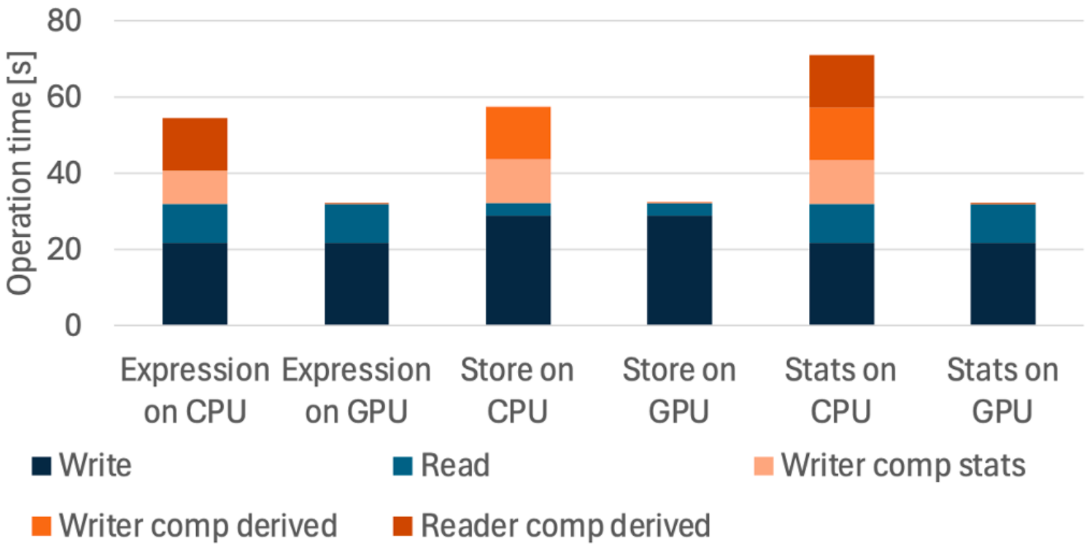


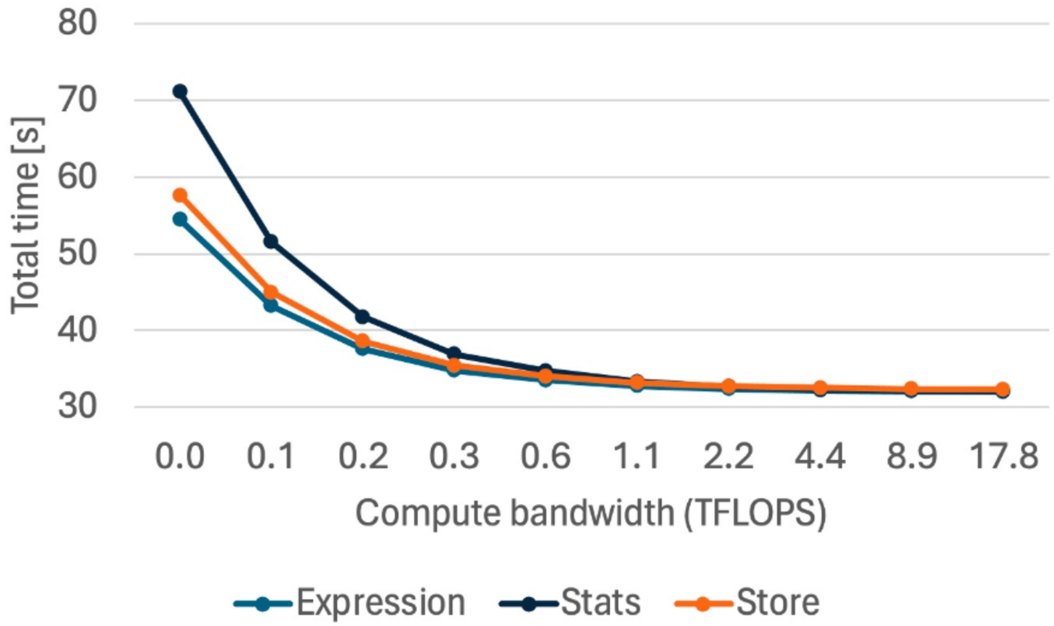
Metric	Frontier	Perlmutter
CPU (GFLOPS)	250	35
GPU (TFLOPS)	25	9.5
B_{GPUmem} (TB/s)	1.5	1.6
B_{write} (GB/s)	1.6	1
B_{read} (GB/s)	3.5	1.13

Compute FLOPS

- Store and Expression have similar performance
  - For complex kernels Expression is the best (since we are reading the entire dataset)

- Heavyweight kernels wanting to use the Stats strategy
  - Will have to pay the price of a less performant write operation.

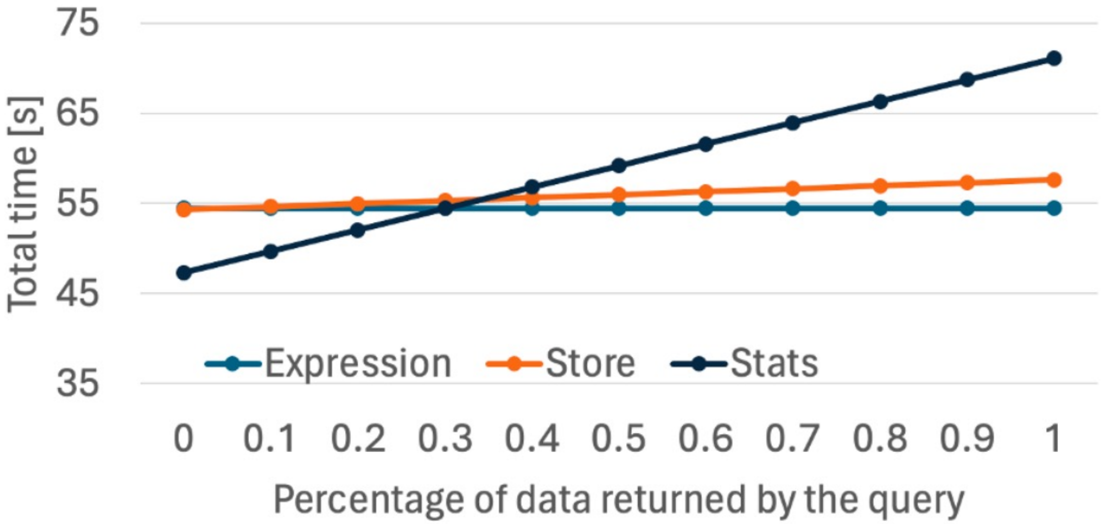




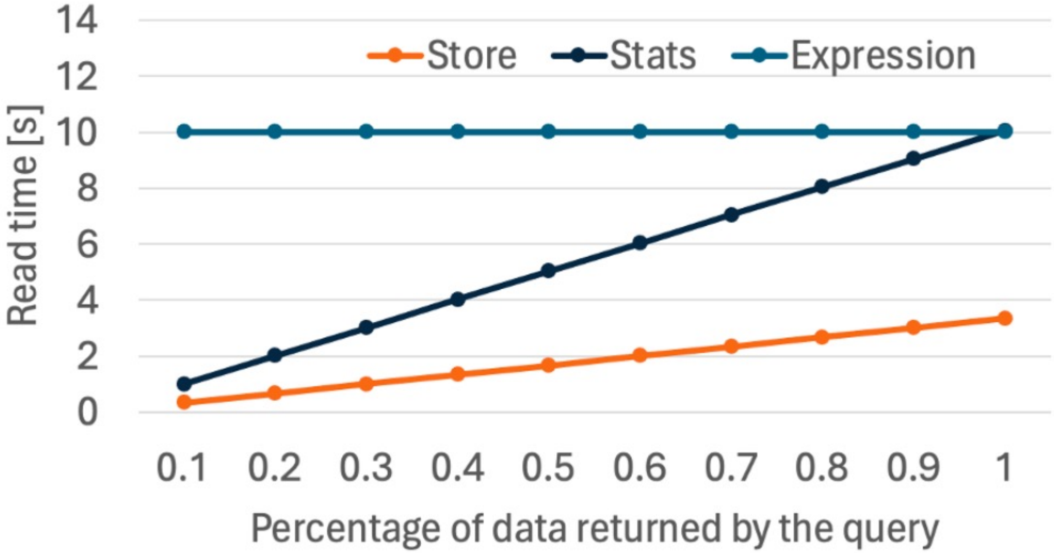
3 primary variables with magnitude, each of 11.7GB, and reading the entire dataset

Percentage of data being read

For the same case as before, queries that return over 30% of the data are more costly with Stats



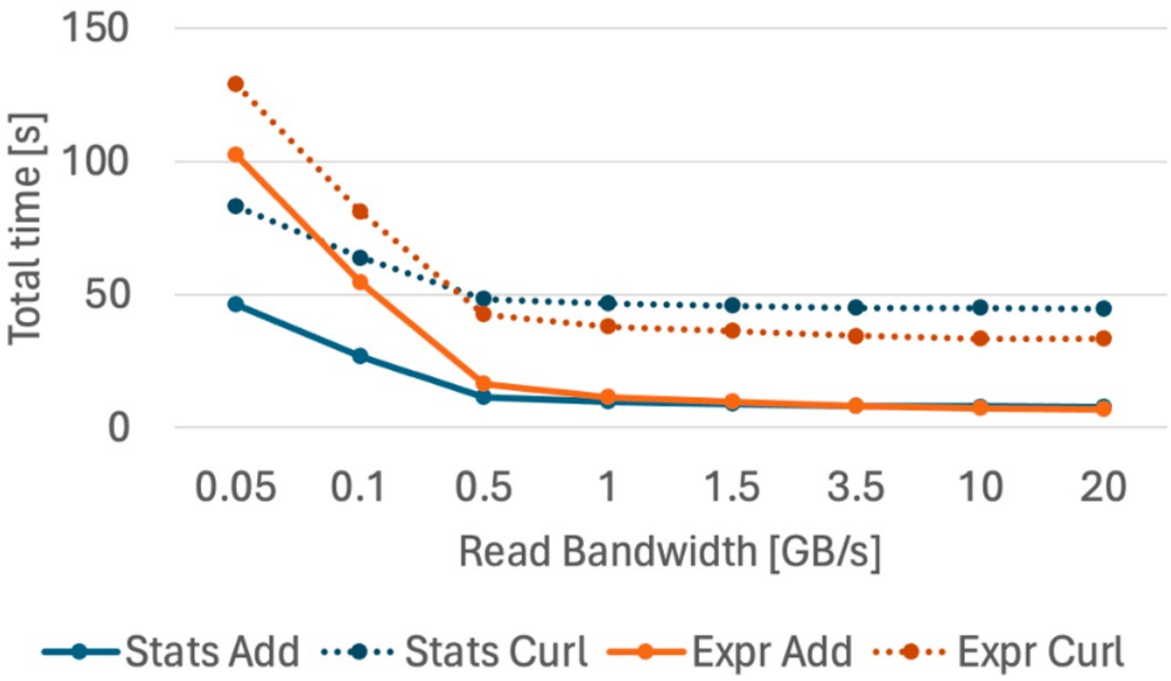
Looking just at the read time (assuming the data exists and we want to continuously query it)



Network bandwidth

- The Expression strategy is not efficient for remote access
  - Should only be used for analysis on local clusters
  - For both lightweight Add and heavyweight Curl

- A decrease in kernel complexity or an increase in compute capability influences the turning point
  - e.g. on the GPU on Frontier the Stats strategy is a better choice for read bandwidths lower than 3 GB/s even if the entire derived dataset is read).



Total time to write and read for different reading bandwidths when the 40% of the derived data is required for analysis.

Problem statement

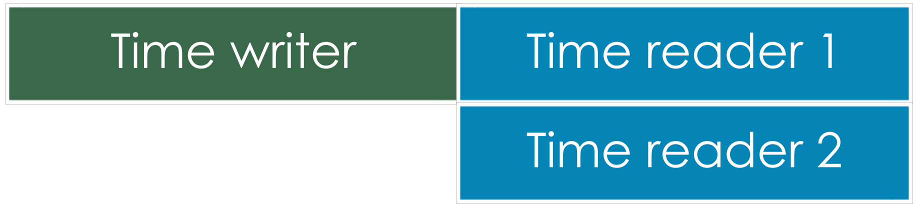
- Given a set of primary and derived variables (size, complexity)
- Given nodes with different compute capabilities
- For a given read pattern (in-situ/file-based, multiple queries, exploratory/fixed pattern, remote/local)

- **What is the strategy for each derived variable?**
  - **Future: which compute resource to use for each derived variable and where to cache data; granularity of the query**

Solution used

- For each variable
  - Compute the cost of each strategy using the model

- For file-based queries, the objective is to minimize the total execution time

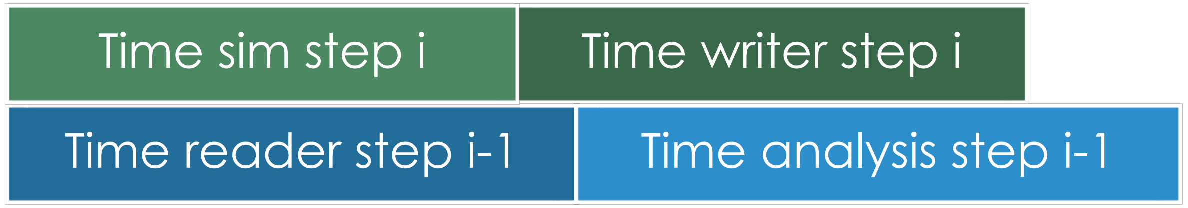


Time writer

Time reader 1

Time reader 2

- For the in-situ query, the objective is to balance the writer and reader



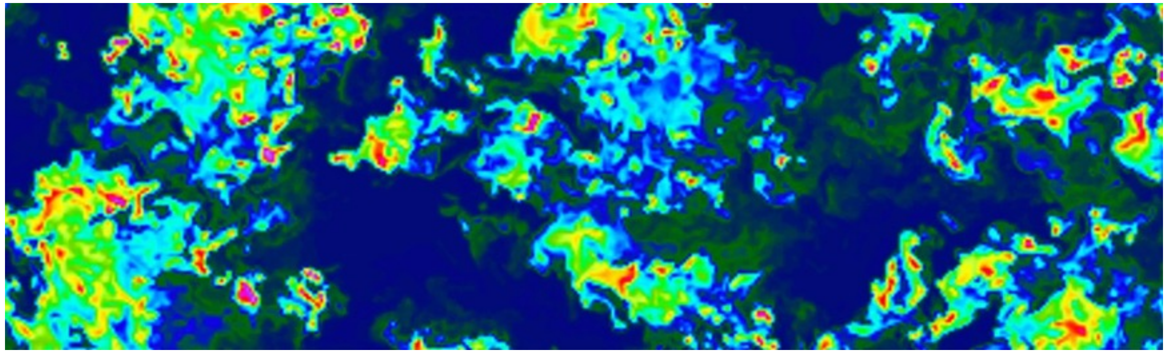
Time sim step i

Time writer step i

Time reader step i-1

Time analysis step i-1

Applications on Frontier



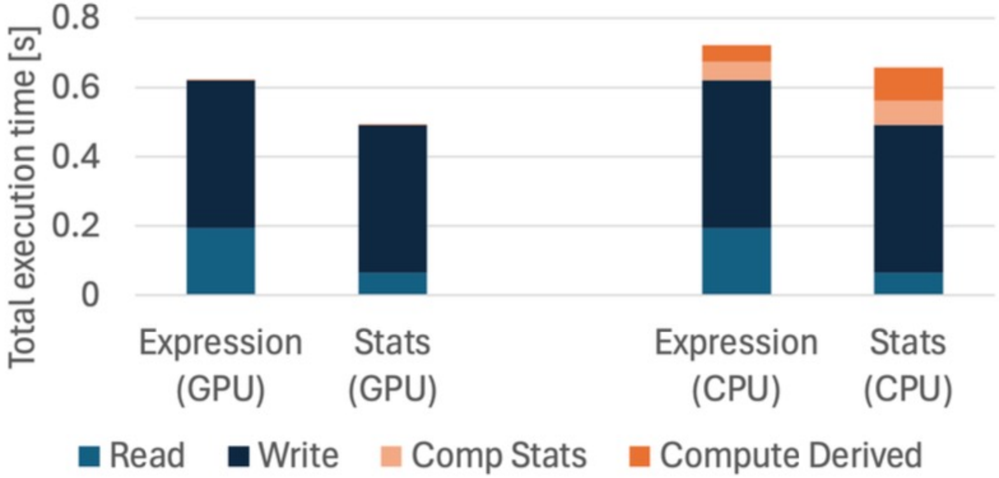
- The S3D simulation
  - Generates 1.5 TB of data in each step through 24 primary variables
  - Particles are stored in 3D arrays of 280x280x1280 size
    - Velocity is stored using 3 of separate variables, each requiring 64 GB on 900 ranks
  - Query on magnitude either in-situ or on remote laptop, plot of temp

- The e3sm simulation
  - Outputs model data at the 6-hourly interval generating around 24 GB through 9 primary variables on 96 ranks
  - Tropical cyclone track code queries the magnitude of curl of velocity

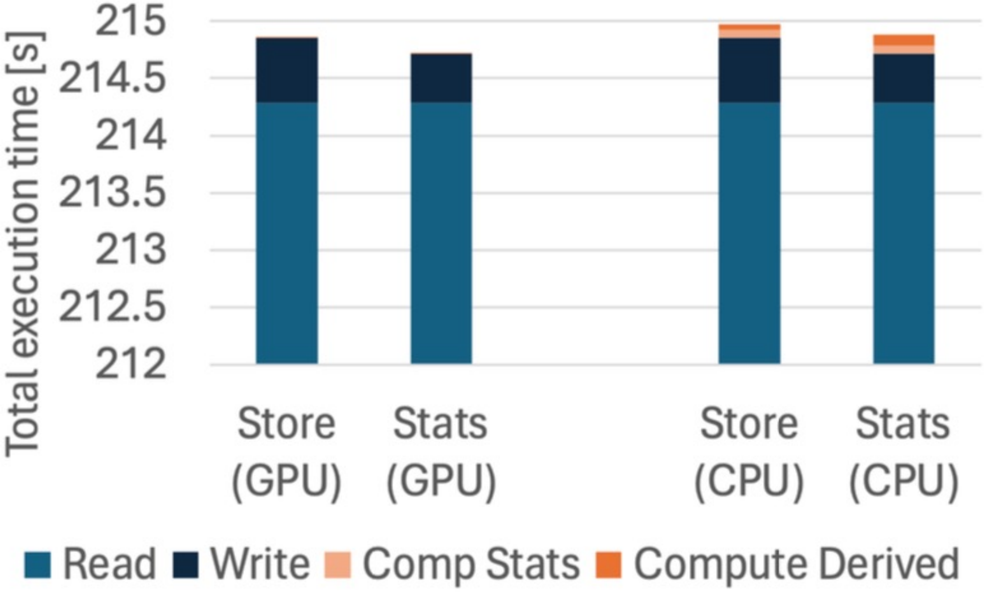
S3D

- The magnitude derived variable has a size equal to the number of particles
  - The Store strategy adds 64 GB of data for each simulated step
  - For 900 ranks the stats are12 MB

- The Expression strategy requires storing 256 GB on the remote site



(a) In-situ analysis

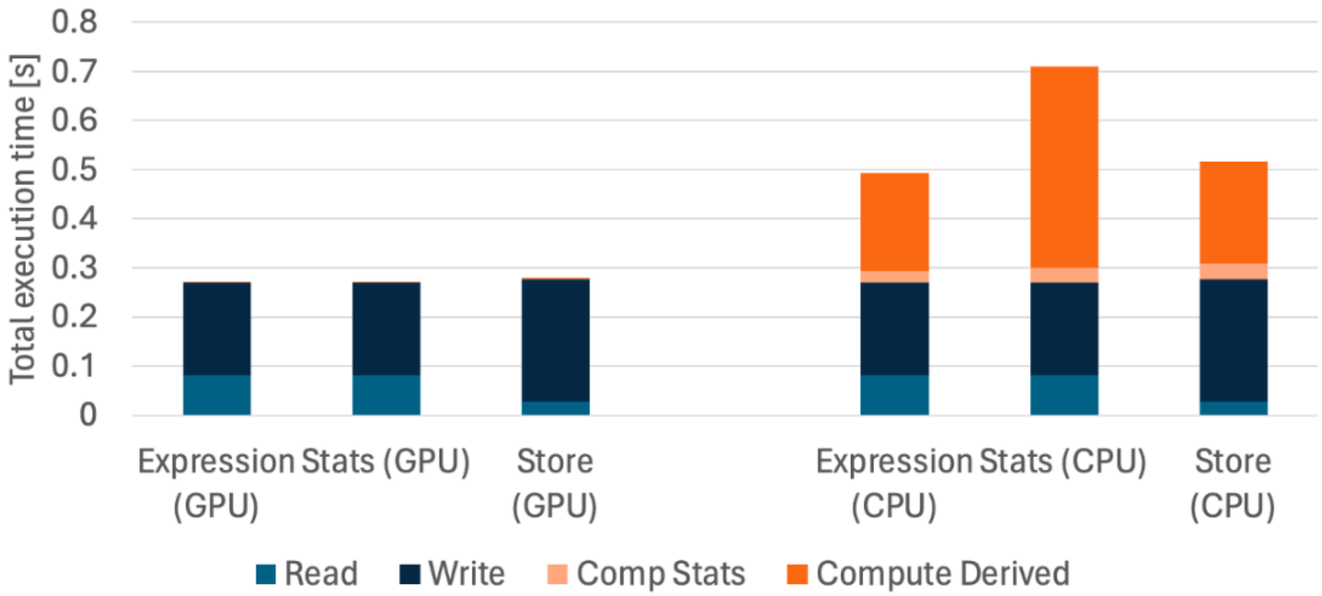


(b) Remote analysis

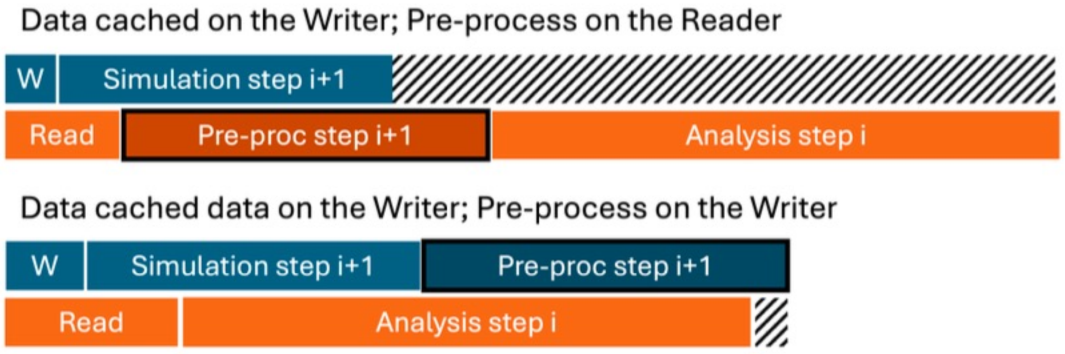
E3SM

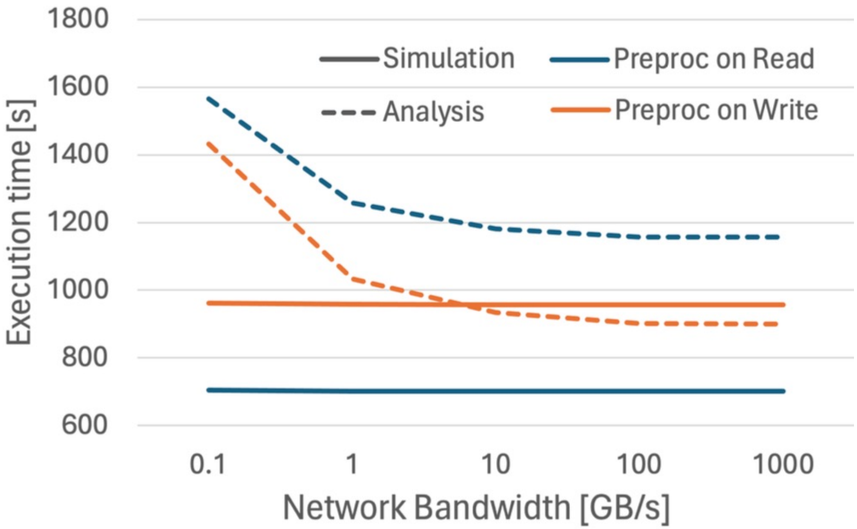
- The size of the curl variables is 4 GB
  - The Store strategy adds 28 GB
  - The stats for 96 ranks is 1 MB

- Stats strategy is 1.5x slower
  - Curl has high complexity
  - The curl values are needed by the analysis

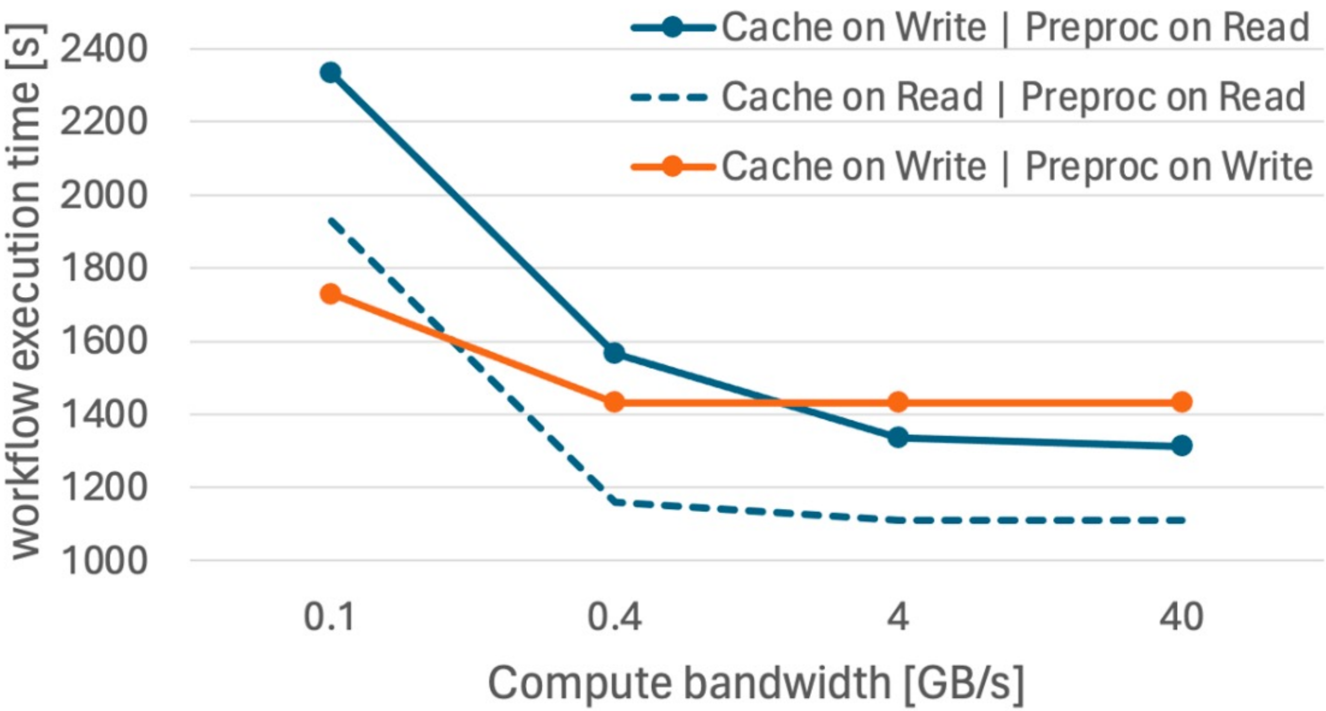


AI training coupled with simulation





- Caching strategies
  - Need to consider the cost of recomputing
  - How long to cache? Eviction strategies need to be weighted



Conclusion

- Simple models work for file-based case
  - Straight forward input parameters
  - Easy to understand trade-offs
  - Offloading derived variable computation to I/O libraries offers flexibility in switching strategies

- Future work
  - Include compute on storage/ compute in-transit
  - Include pre/post processing as derived variable
  - In-situ performance needs to be modeled better: stochastic behavior
  - Given an energy cap, query granularity