OAK RIDGE National Laboratory

Applying Machine Learning to Understand Write Performance of Large-scale Parallel Filesystems

Presented by Bing Xie

Bing Xie, Zilong Tan, Philip Carns, Jeff Chase,
Kevin Harms, Jay Lofstead, Sarp Oral,
Sudharshan S. Vazhkudai, Feiyi Wang

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

U.S. DEPARTMENT OF ENERGY

Applying Machine Learning to Understand Write Performance of Large-scale Parallel Filesystems

- Problem
  - Understand the write performance of HPC applications running on large-scale systems
- Contribution
  - Built accurate ML models for predicting the I/O write performance
  - Interpreted multi-stage write behaviors of large-scale I/O subsystems
- Impact
  - Demonstrated that ML can be applied to predict the write performance of large-scale I/O subsystems
  - Delivered a generic solution applicable to various large-scale I/O subsystems and technologies

Motivation: Reduce the Write Cost

- Configure write burst size/rate tradeoffs
- Guide I/O middleware (e.g., ROMIO) to adapt write patterns
- Inform system job schedulers to yield tighter/better estimates of I/O cost and application runtime

Related Works and Our Solution

- I/O performance studies
  - Profiling supercomputer I/O subsystems under production loads
  - Darshan toolkit
  - Statistical benchmarking
- I/O middleware systems
  - ROMIO, ADIOS
- ML in I/O performance prediction
  - Tune I/O parameters at application level
  - Learn I/O patterns from job logs and system monitoring data
- Our Solution
  - First ML work to predict write performance of large-scale parallel filesystems based on application write patterns, system architecture, and configurations

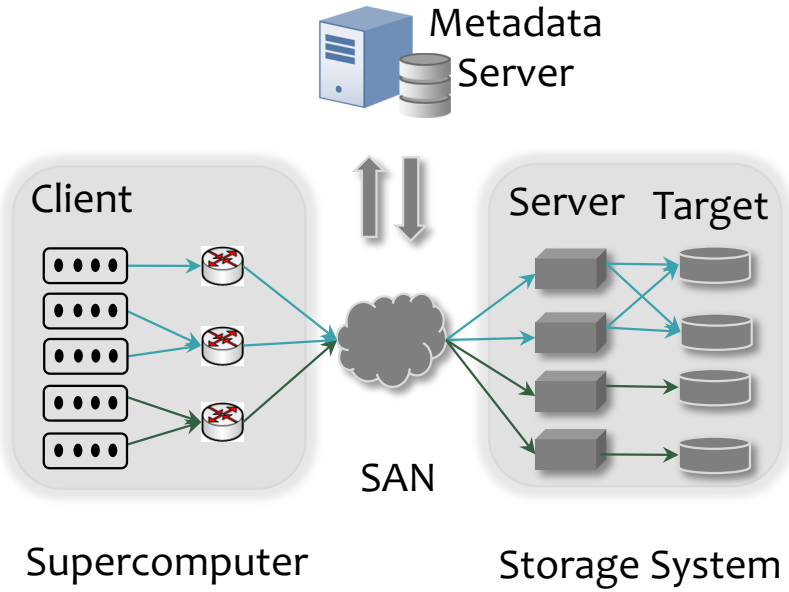
Typical Scientific Applications

- HPC codes compute for a long time at large scales
- Produce write bursts that stall application executions and impact application runtime

- A generic example: XGC
  - Evaluate **physical equations** iteratively over space: compute cost is predictable
  - **4 types of bursts** with different write frequencies and burst sizes:
    - state snapshots: 500MB to 1.2GB
    - diagnostic analysis bursts: 1MB – 400MB
    - Bursts are stored as independent files
  - **Write stalls** comprise 7-20% of run time

Target I/O systems

- Titan and Spider 2 at OLCF/ORNL
  - Cray XK7
  - Lustre filesystem
- Cetus and Mira-FS1at ALCF/ANL
  - IBM Blue Gene/Q
  - GPFS filesystem

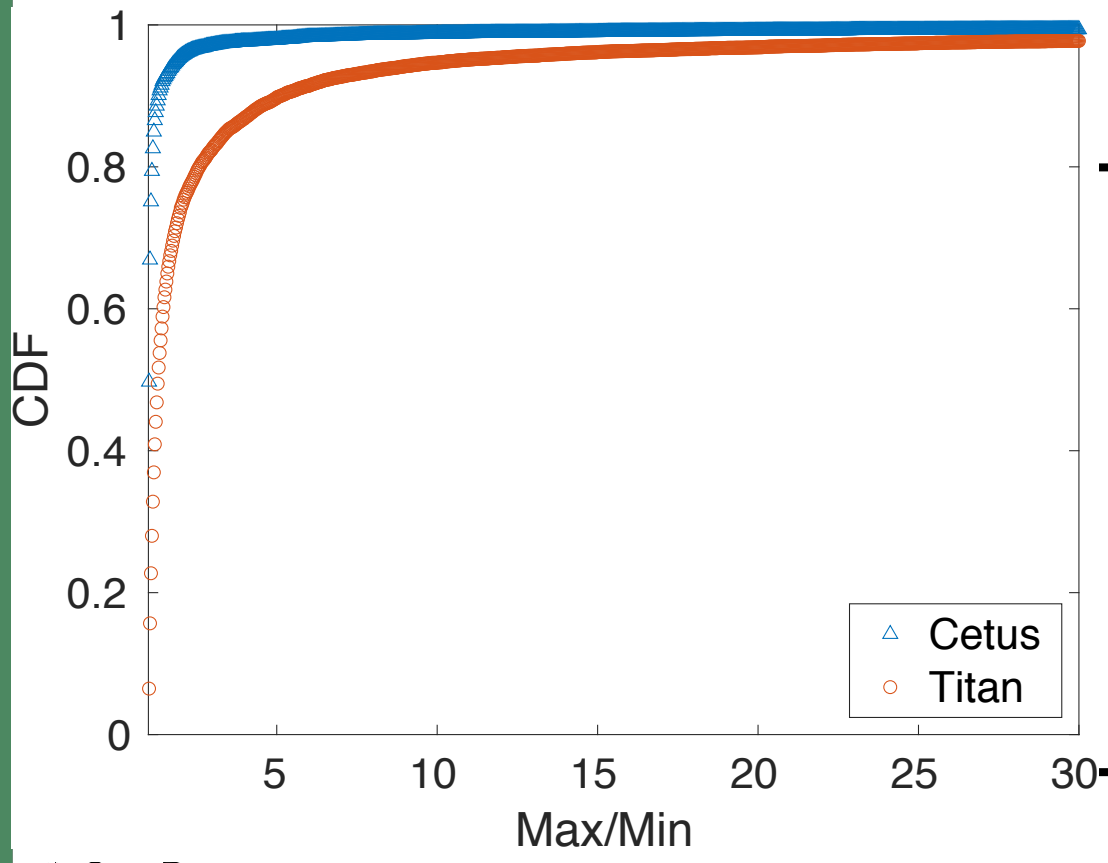


Challenges

- High performance variability
- Limited filesystem visibility for end-users

High Performance Variability

1. CDFs of write performance variations on Titan and Cetus.



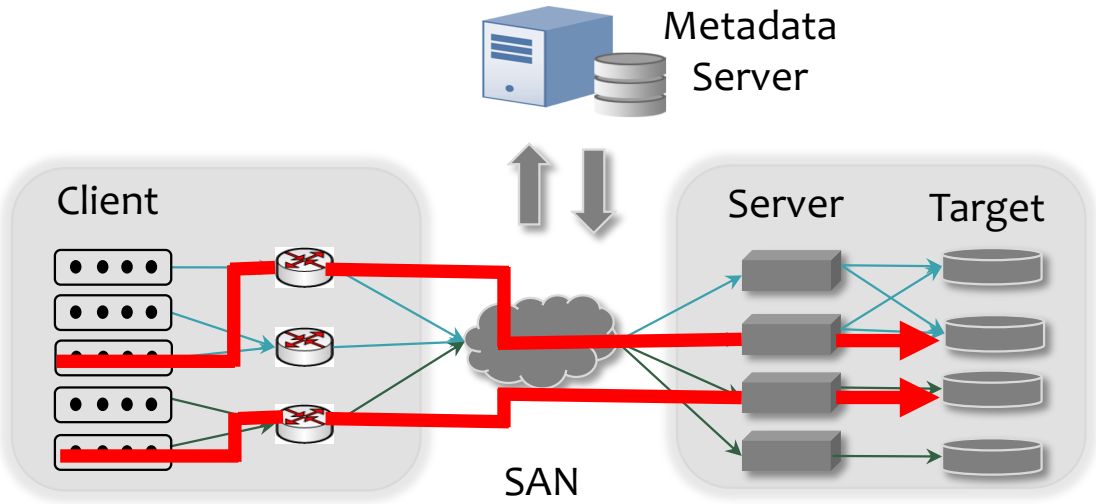
3. Write performance on Titan and Cetus is highly variable.

2. The x-axis represents the relative measures (max/min) of the write bandwidths of the experiment data (IOR benchmarks)

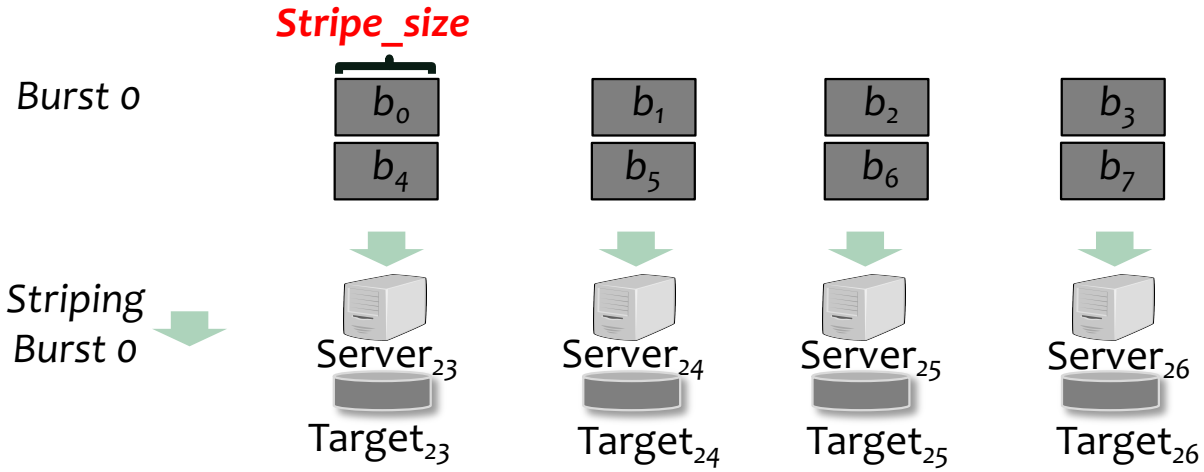
Our Approach

- Highly variable, but reverts to mean over time
  - **Model the mean performance**
  - **Effectively address the repeated I/O writes and aggregate impact**
- Limited visibility for end users
  - **Extract features from write patterns and system architecture and configurations**
- Interference
  - **Address noise as features**
- ML solution
  - **Convergence-guaranteed sampling method**
  - **Lasso models**
  - **Systematic ML methodology**

End-to-end I/O Write Path



Each Target is a RAID array.

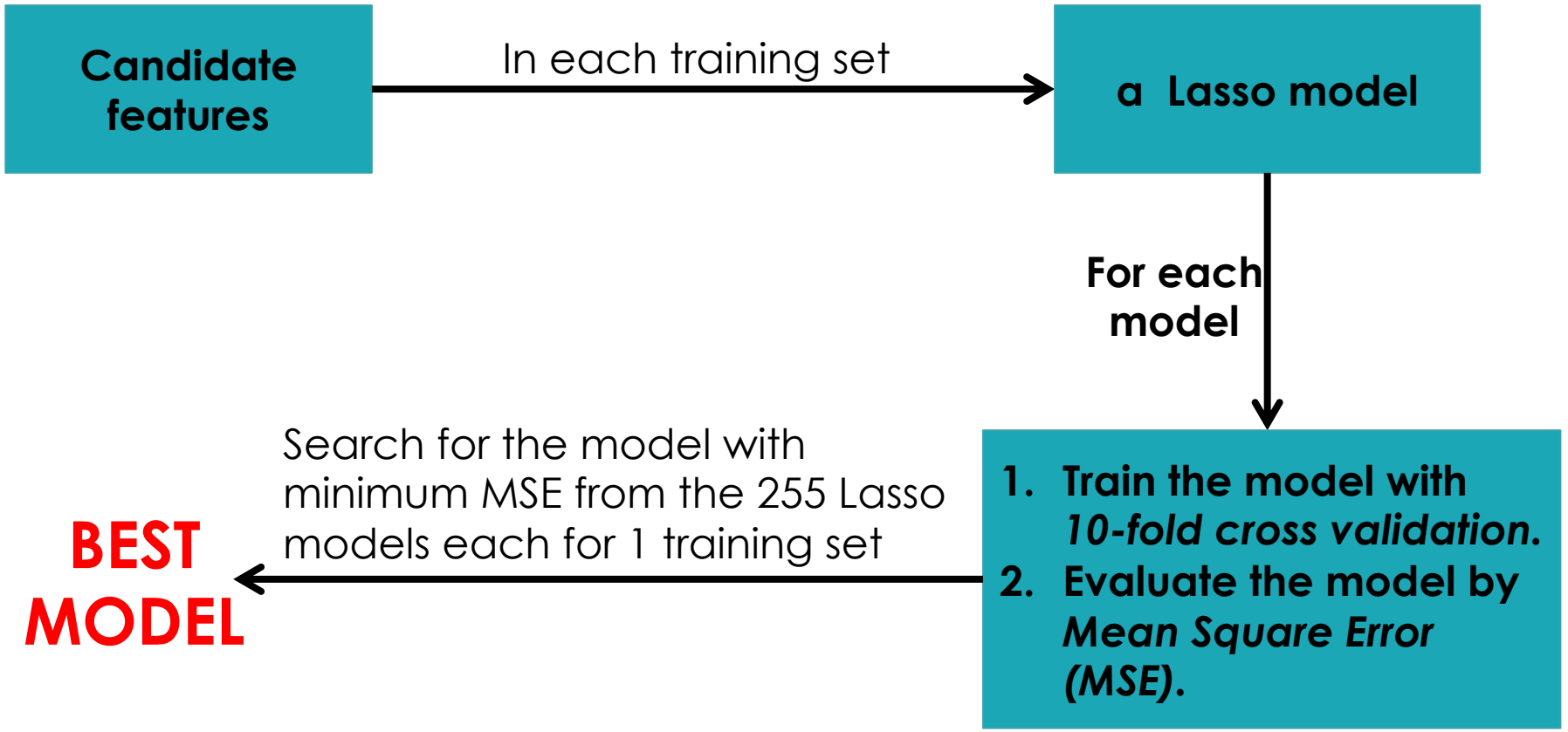


Example:
***Stripe_Count*=4**
***Starting_OST*=23**

Extract Features

- Insight: infer end-to-end burst absorption time based on performance-related parameters (**write load**, **load skew**, **resources in use**) at each stage
- Collectable performance-related parameters on Titan and Cetus
- Predictable performance-related parameters on Spider 2 and Mira-FS1
- Positive and inverse forms of performance-related parameters on separate stages, adjacent stages, and noise
- Titan/Spider 2: 41 features; Cetus/Mira-FS1: 30 features

Systematic Machine Learning Approach



Experiments

- Train models on a small scale data set
  - 3,465 (Titan) and 4,715 (Cetus) converged samples collected with multiple IOR benchmarks on the scale of 1-128 compute nodes
- Evaluate models on medium scale
  - 668 (Titan) and 874 (Cetus) converged samples produced by 200 -512 compute nodes
- Evaluation criteria
  - Accuracy of the best model
  - Effectiveness of features

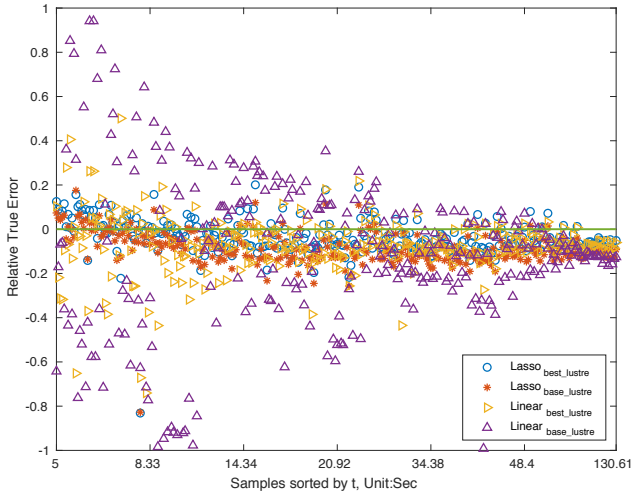
Reported 4 models

- $Lasso_{best}$
  - *With minimum Mean Square Error* from 255 Lasso models across the training set candidates
- $Lasso_{base}$
  - The Lasso model trained on the write scales of 1-128 compute nodes
- $Linear_{best}$
  - *With minimum Mean Square Error* from 255 Linear models across the training set candidates
- $Linear_{base}$
  - The Linear model trained on the write scales of 1-128 compute nodes

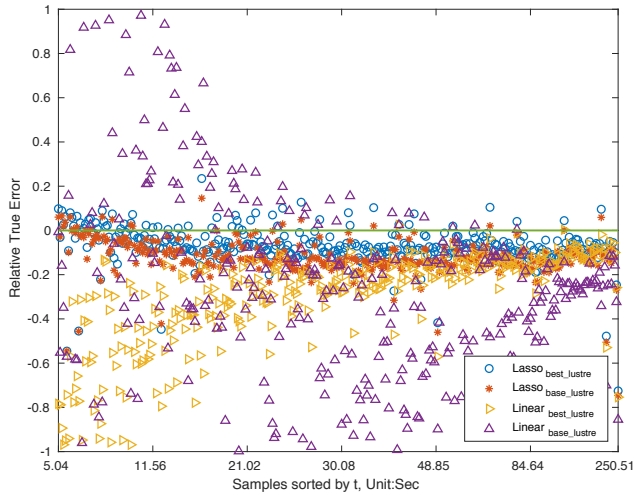
Results on Titan and Cetus

Titan/Spider 2

test set with 200, 256 nodes

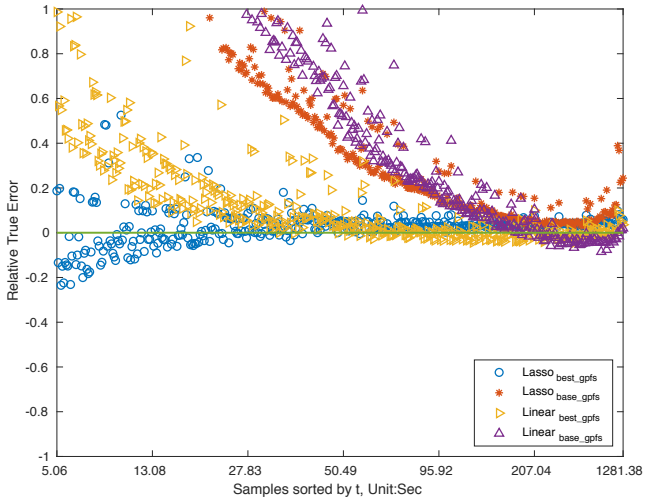


test set with 400, 512 nodes

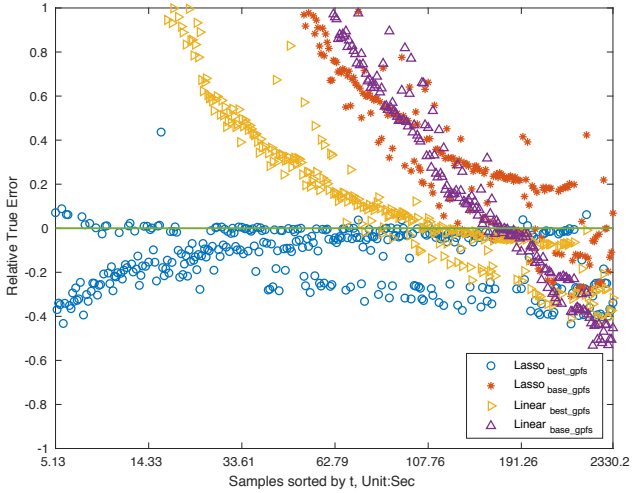


Cetus/Mira-FS1

test set with 200, 256



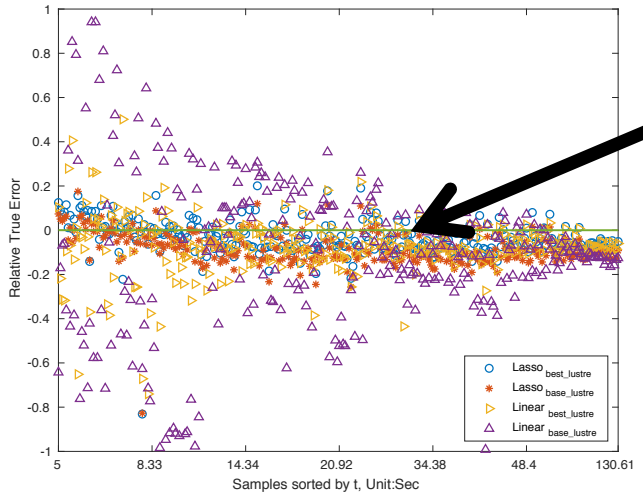
test set with 400, 512

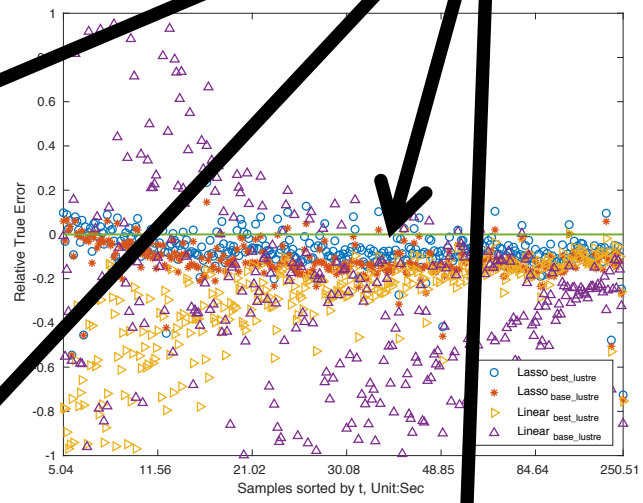


Results on Titan and Cetus

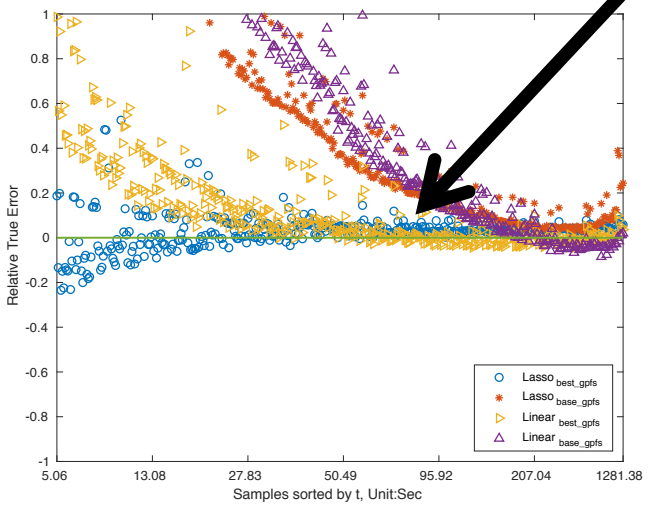
***Lasso_best* is highly accurate and the best model**

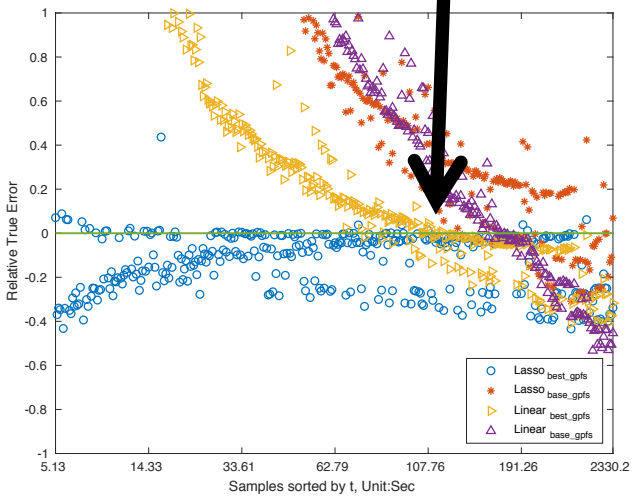
Titan/Spider 2





Cetus/Mira-FS1





OAK RIDGE National Laboratory

Conclusions

- Problem
  - Understand the I/O write performance of large-scale supercomputers
- Our Solution
  - Systematic ML approach with Lasso
  - Modeling the mean performance, extracting features from application write patterns, system architecture and configurations, convergence-guaranteed sampling
- Findings
  - $Lasso_{best}$ is the most accurate model for both Titan and Cetus
  - Most effective features are load skew in supercomputers and resources in use on the system side
- Applicability
  - Lasso models, features: Lustre, GPFS deployment
  - Systematic modeling method: generic supercomputer I/O subsystems

Acknowledgement