NOV 12, 2018

# Benchmarking Machine Learning Methods for Performance Modeling of Scientific Applications

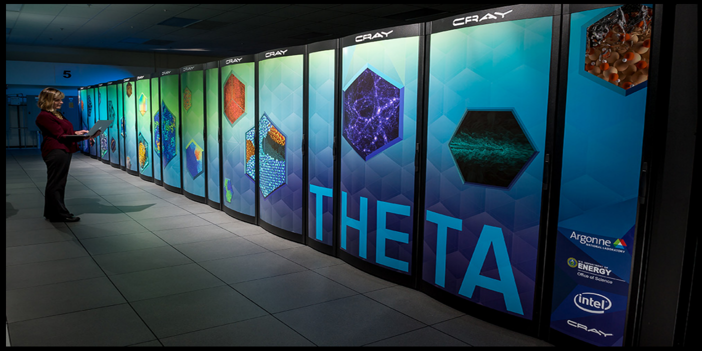

Preeti Malakar, **Prasanna Balaprakash**, Venkatram Vishwanath, Vitali Morozov, Kalyan Kumaran

**Argonne National Laboratory**

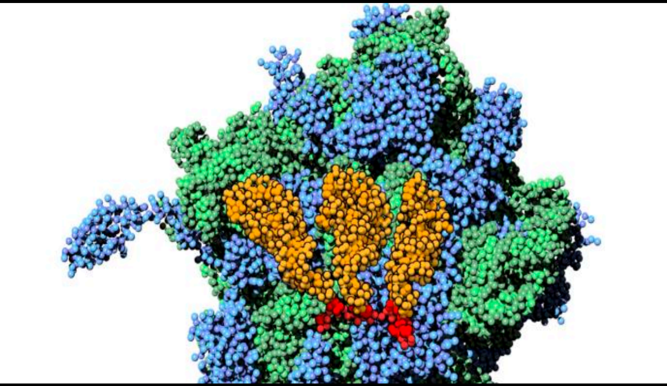


**9th IEEE International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS18)**

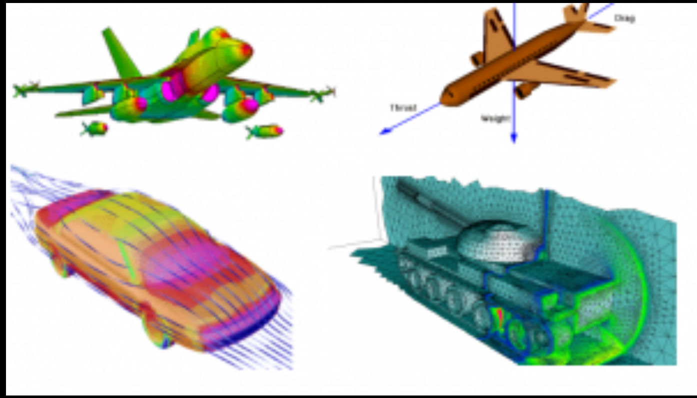
# Application performance modeling

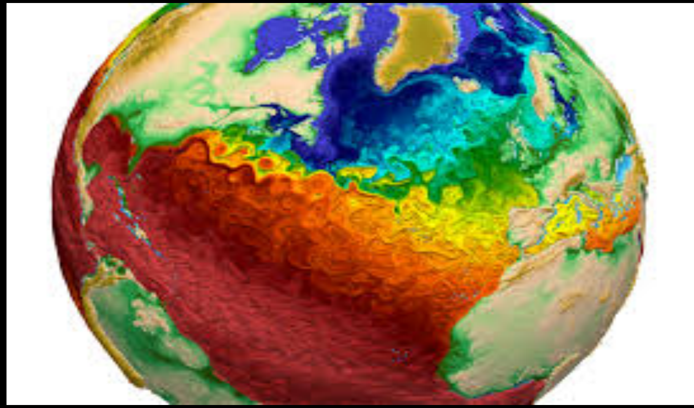




Molecular dynamics



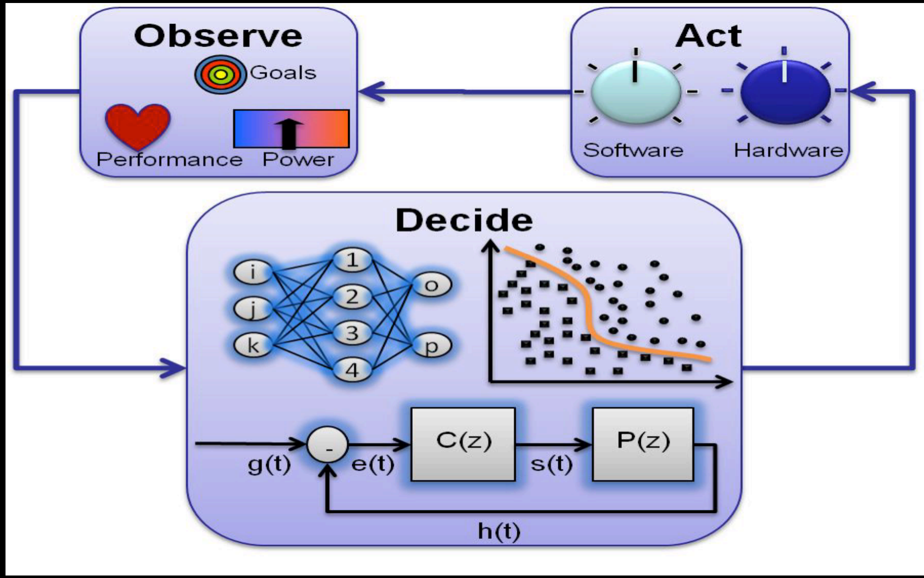
Computational fluid dynamics



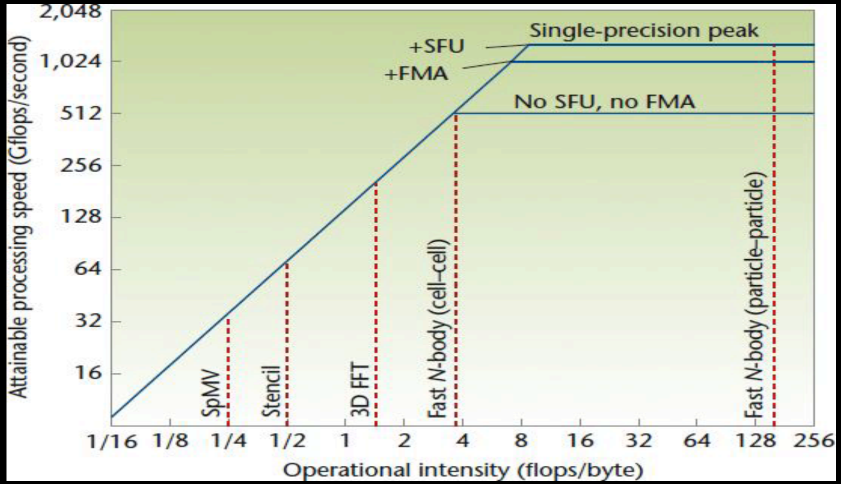
Weather simulations

- Predicting full application performance is ***still*** a challenge
- Shared resources (interconnect, file systems)
    - Background traffic, hardware degradation

# Application performance modeling



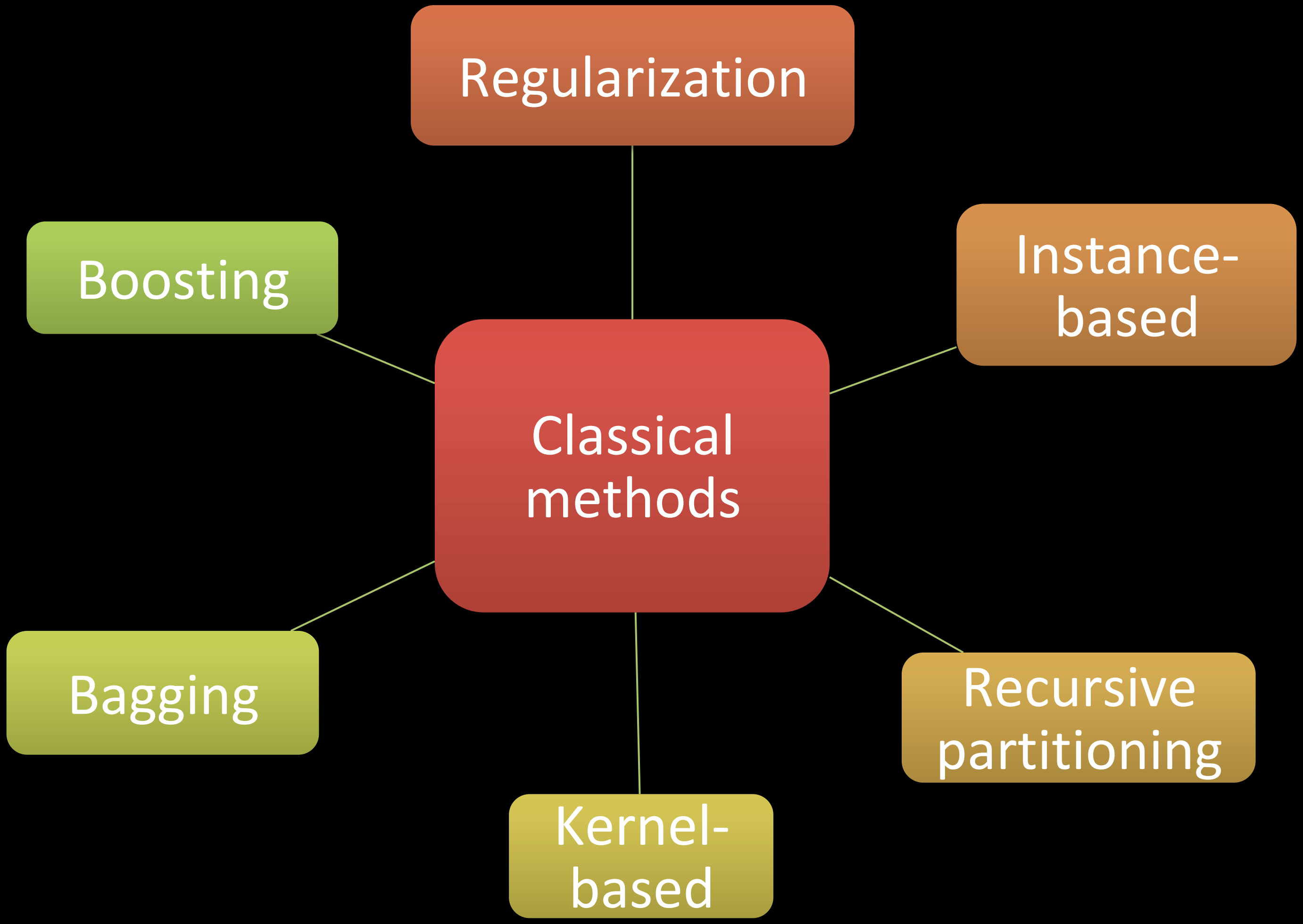

[H. Hoffmann, World Changing Ideas, SA 2009]

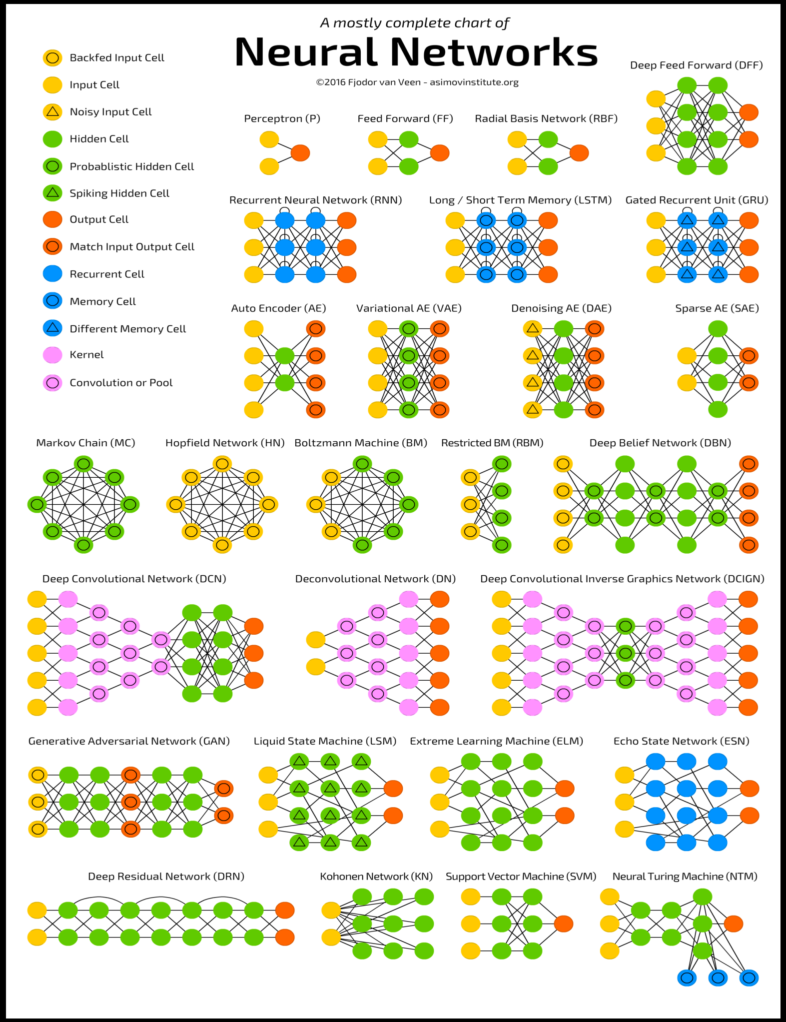



[S. Williams et al., ACM 2009]

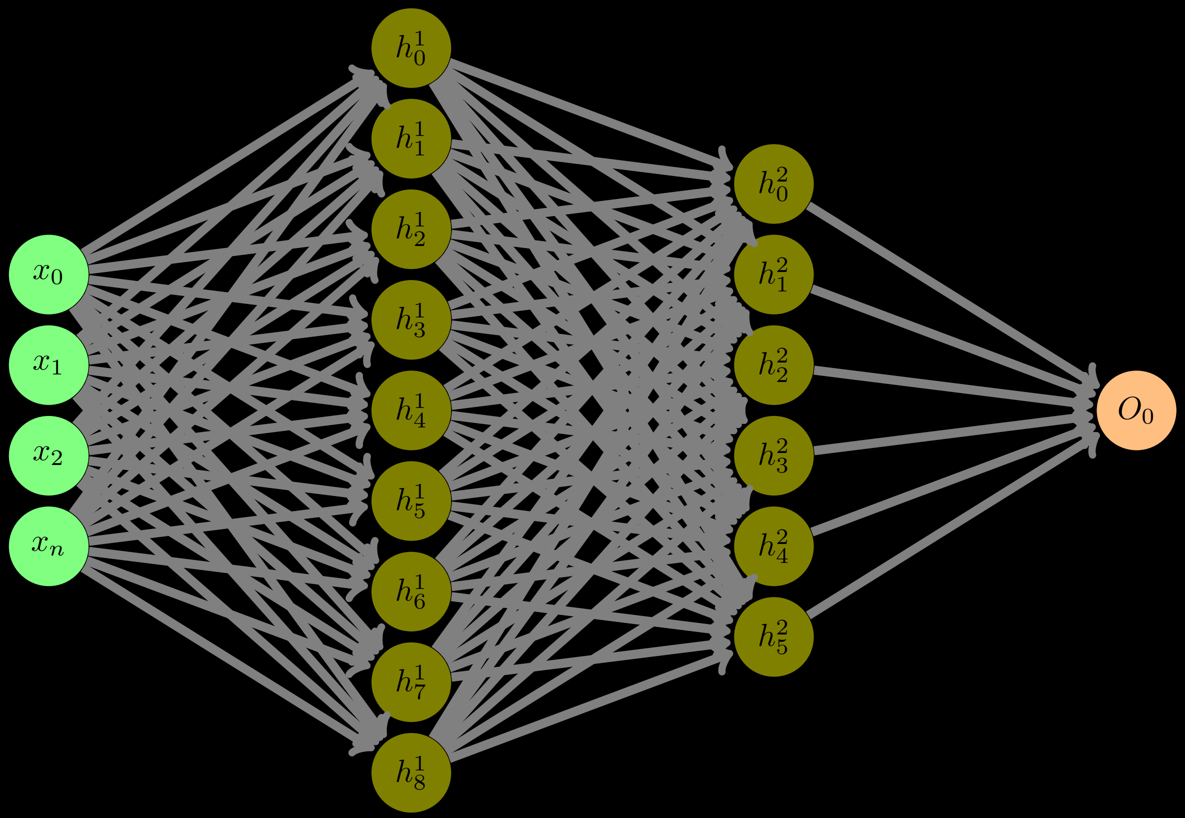
- ***Algebraic performance models*** increasingly ***challenging***
- ***Supervised machine learning performance*** models: an ***effective alternative***
  - ***small number of input-output points*** obtained from empirical evaluation
    - job scheduling , co-scheduling, autotuning

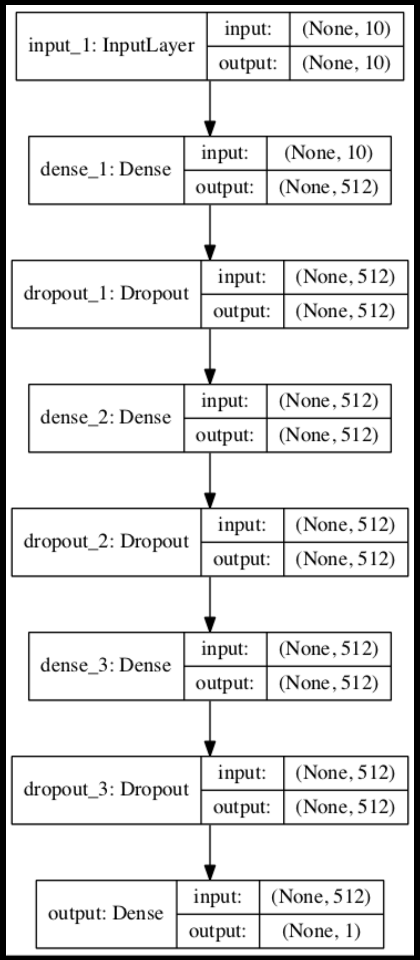
# Supervised learning methods



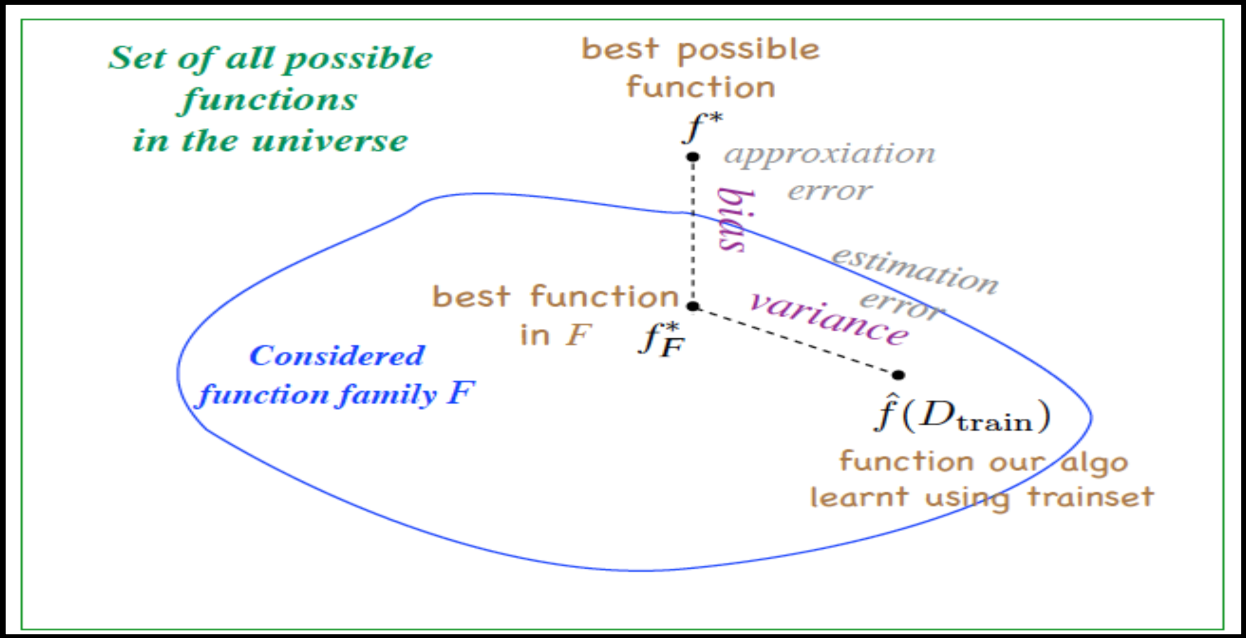



# Deep neural networks

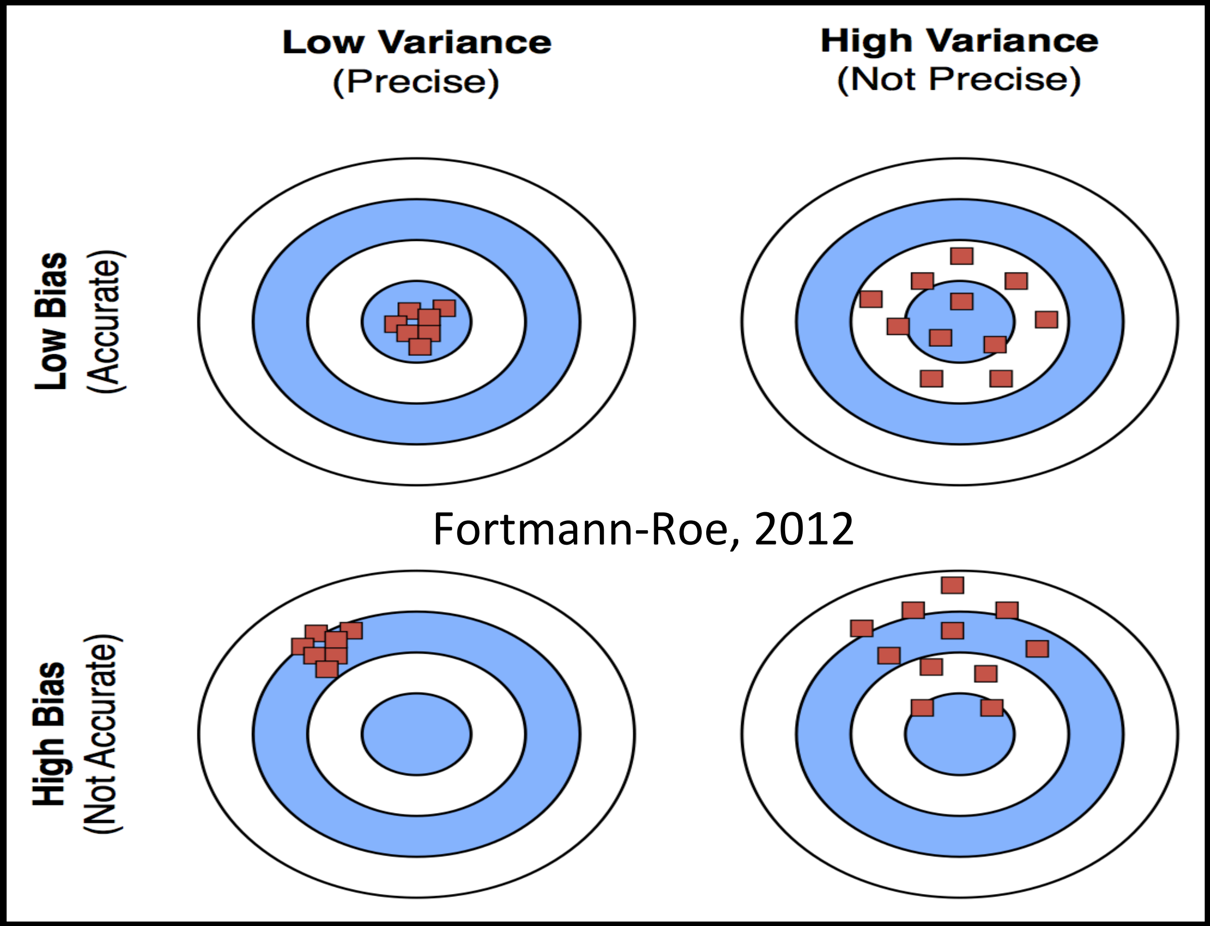


# Why benchmarking?




Deep learning summer school lecture, CIFAR, 2016




Fortmann-Roe, 2012

- ***No free lunch***: no single method will work well on all data set
- All supervised learning algorithms ***seek to reduce bias and variance*** in a different way

# Applications and platforms

| Name | Processor | Interconnect topology | Maximum # cores |
|---|---|---|---|
| Mira (Blue Gene/Q) | Power BQC 1.6 GHz | 5D torus | 131072 |
| Vesta (Blue Gene/Q) | Power BQC 1.6 GHz | 5D torus | 16384 |
| Edison (Cray XC30) | Intel Ivy Bridge 2.4 GHz | Aries with dragon-fly | 1728 |
| Hopper (Cray XE6) | AMD MagnyCours 2.1 GHz | Gemini with 3D torus | 12000 |

- Miniapps (# no of data points):
  - miniMD (< 2K); O(1024) nodes
  - miniAMR (< 1K); O(4096) nodes
  - miniFE (6K to 15K); O(8192) nodes
  - LAMMPS (< 1K ); O(1024) nodes

# Impact of domain-knowledge integration

- No Feature Engineering (No-FE)
  - application input parameters
- Feature Engineering (FE)
  - application input parameters
  - computation
    - ratio of the application problem size and the number of processes
  - communication
    - LogGP model terms
    - binary logarithm of number of processes
  - scaling
    - inverse of the number of processes

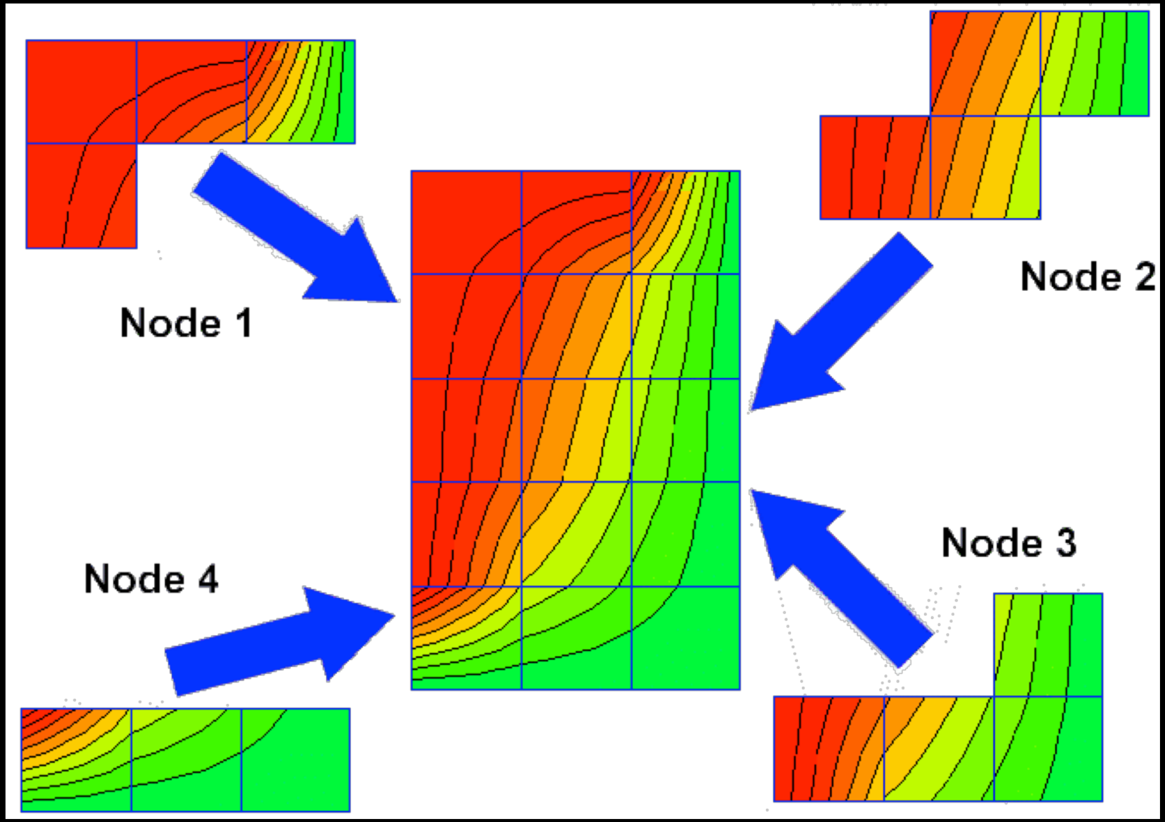
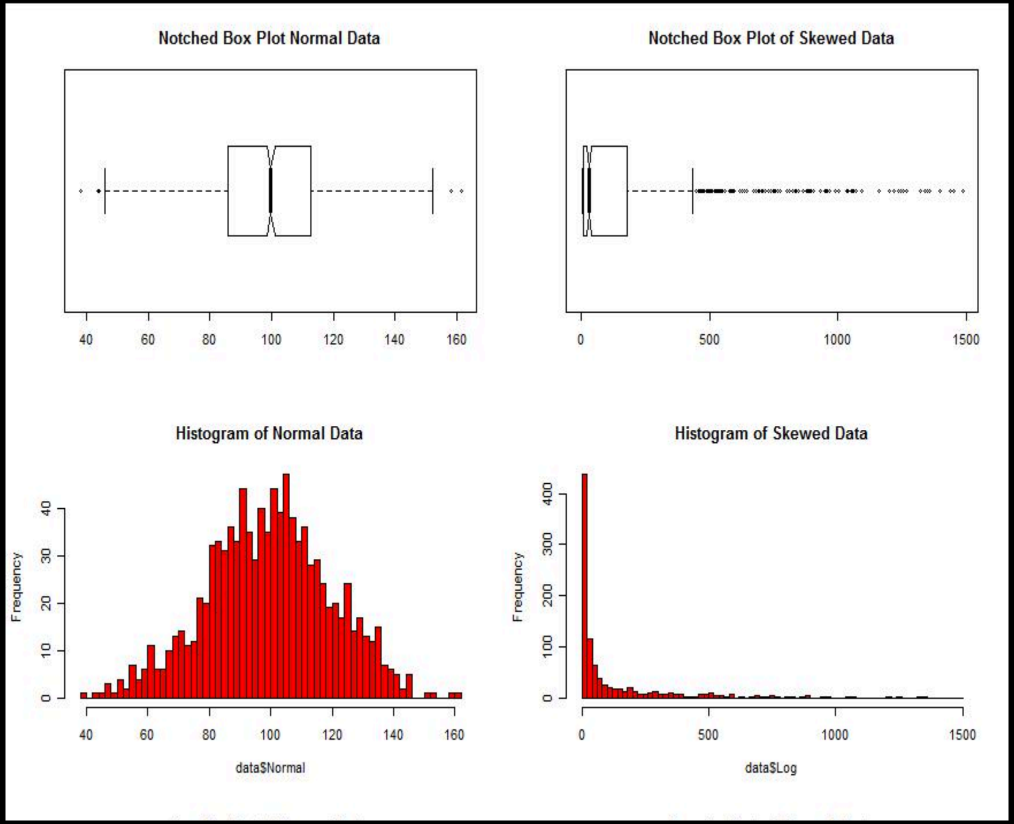
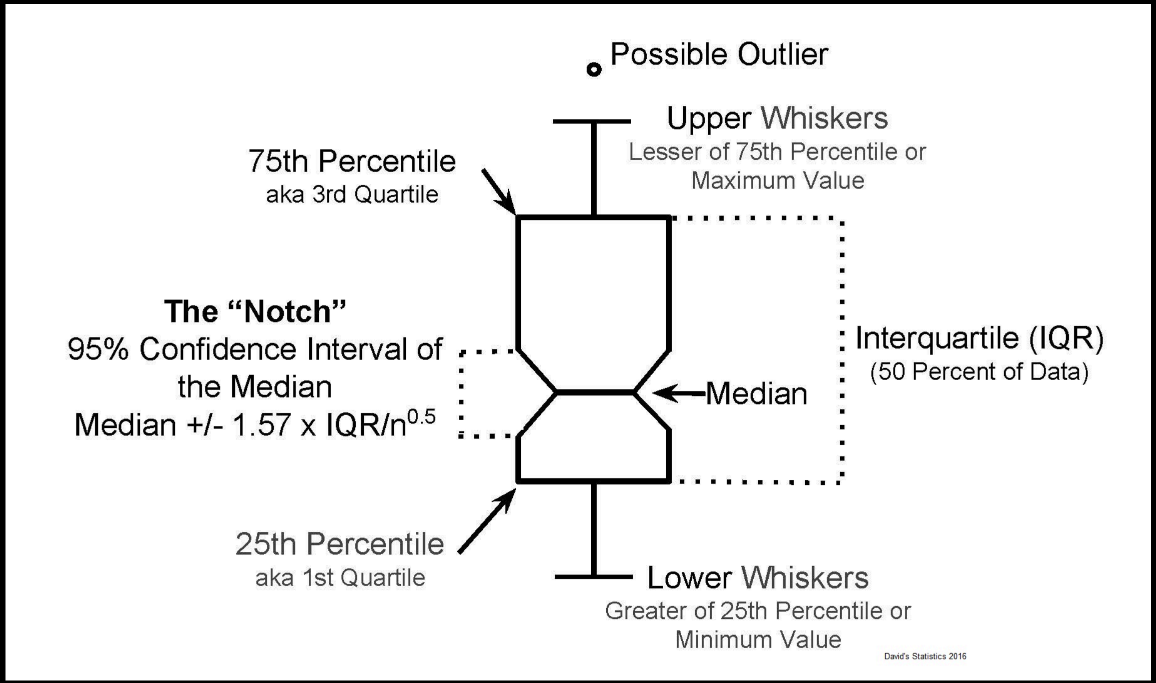



Image from http://www.ddm.org/

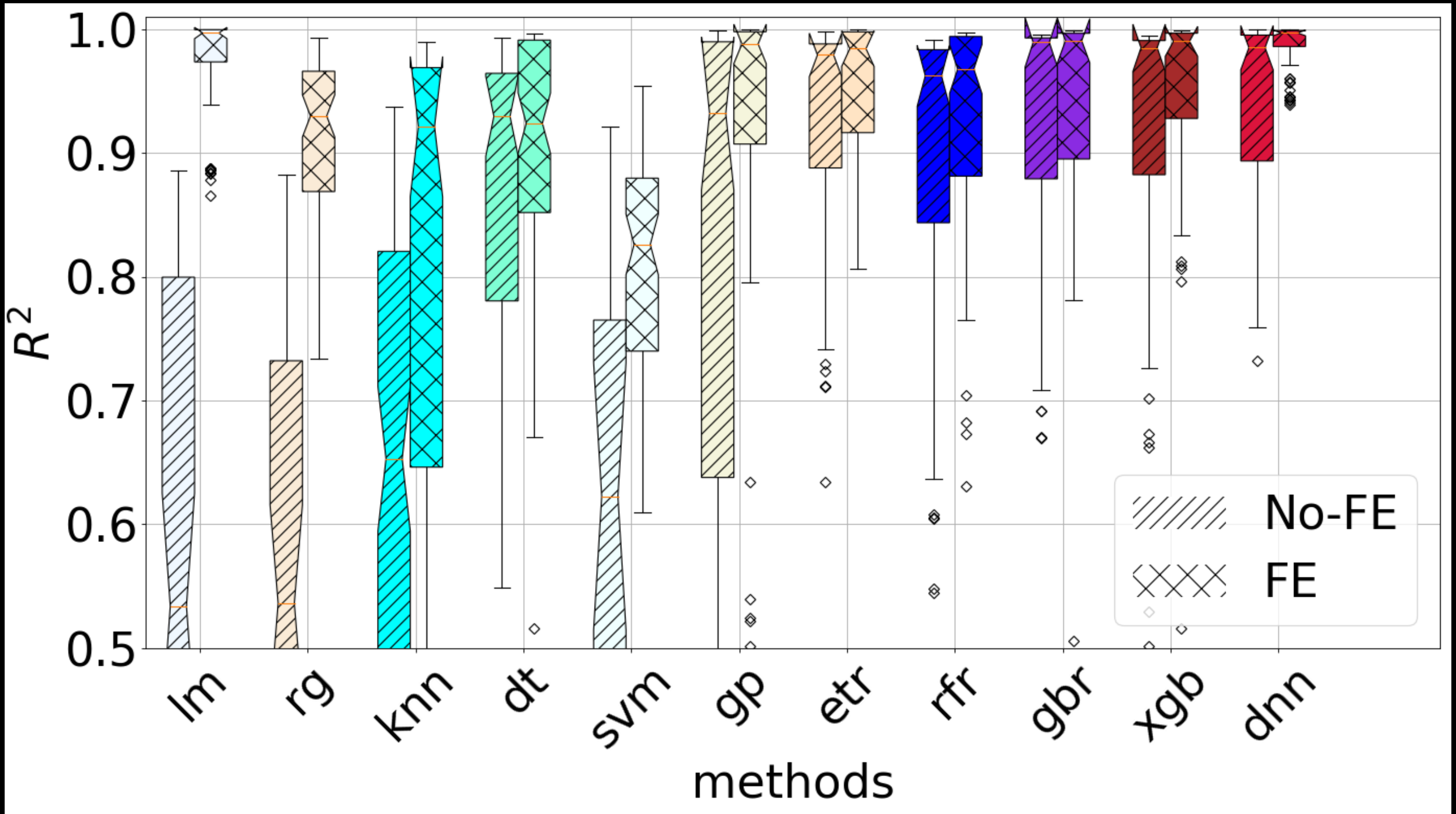
# Box-whisker plot




https://sites.google.com/site/davidsstatistics/home/notched-box-plots

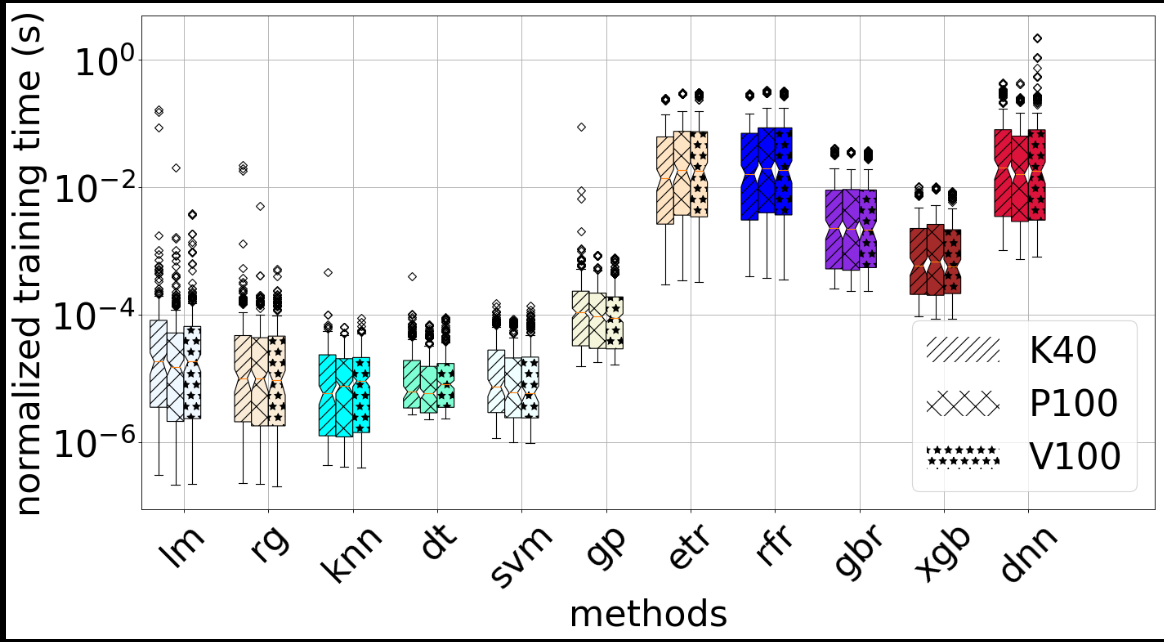
# Impact of domain-knowledge integration

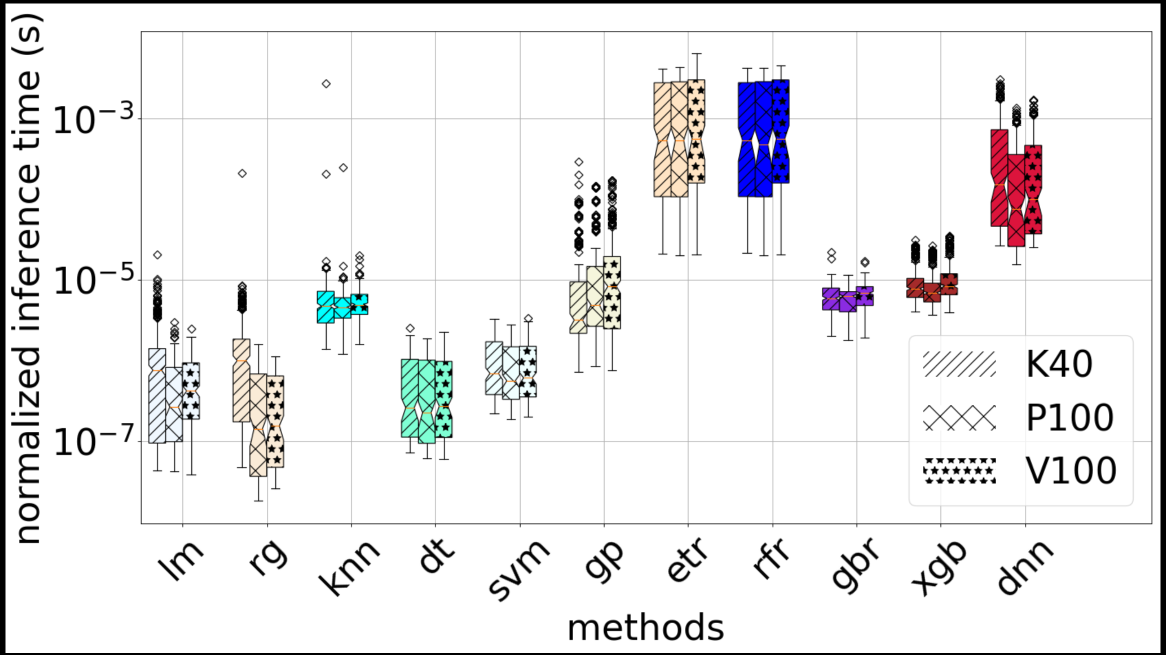



10 X 20:80 cross validation

- Domain-knowledge integration has a significant impact on the accuracy

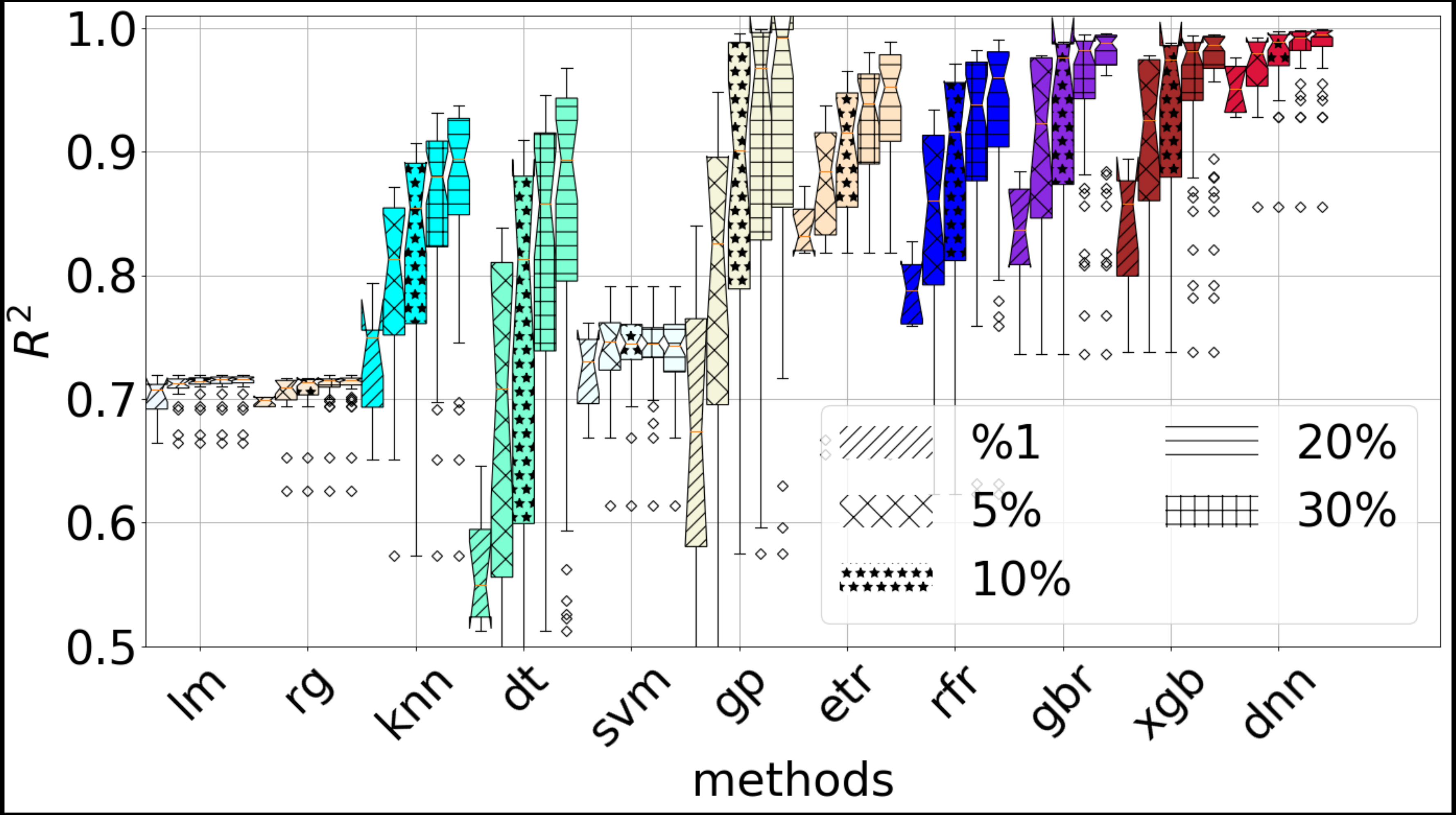
# Impact of hardware platforms



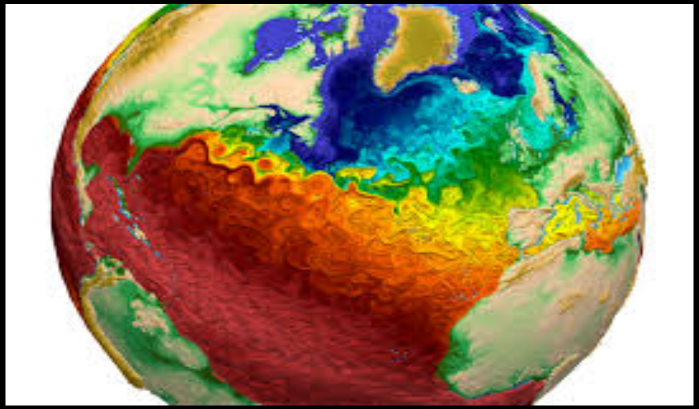

- Algorithmic complexity has more impact than (modern) hardware platforms

# Impact of training data size on accuracy




- Nonlinear methods leverage large training data size

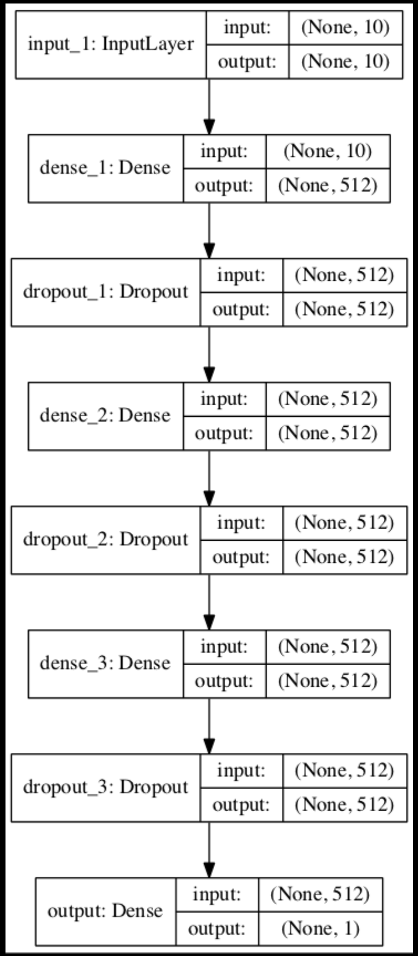
# Transfer learning



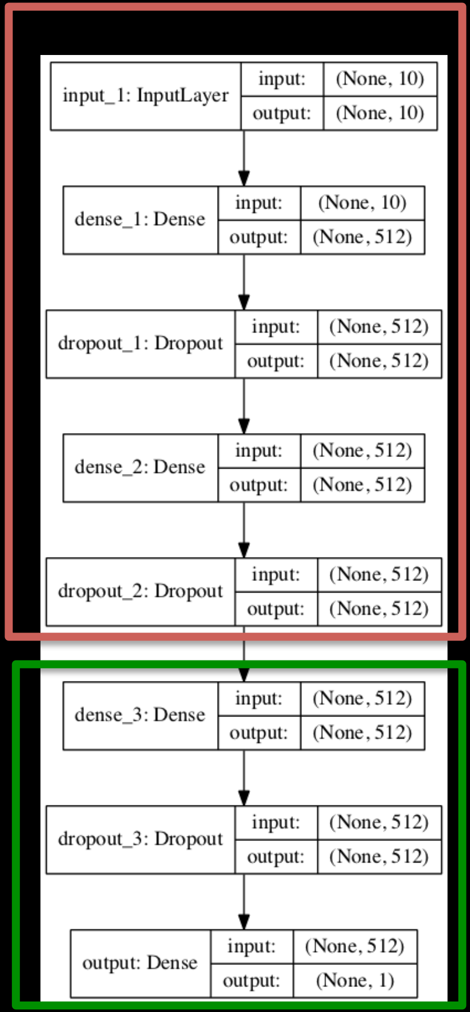


Same application run on two different target systems

# Transfer learning




Mira model

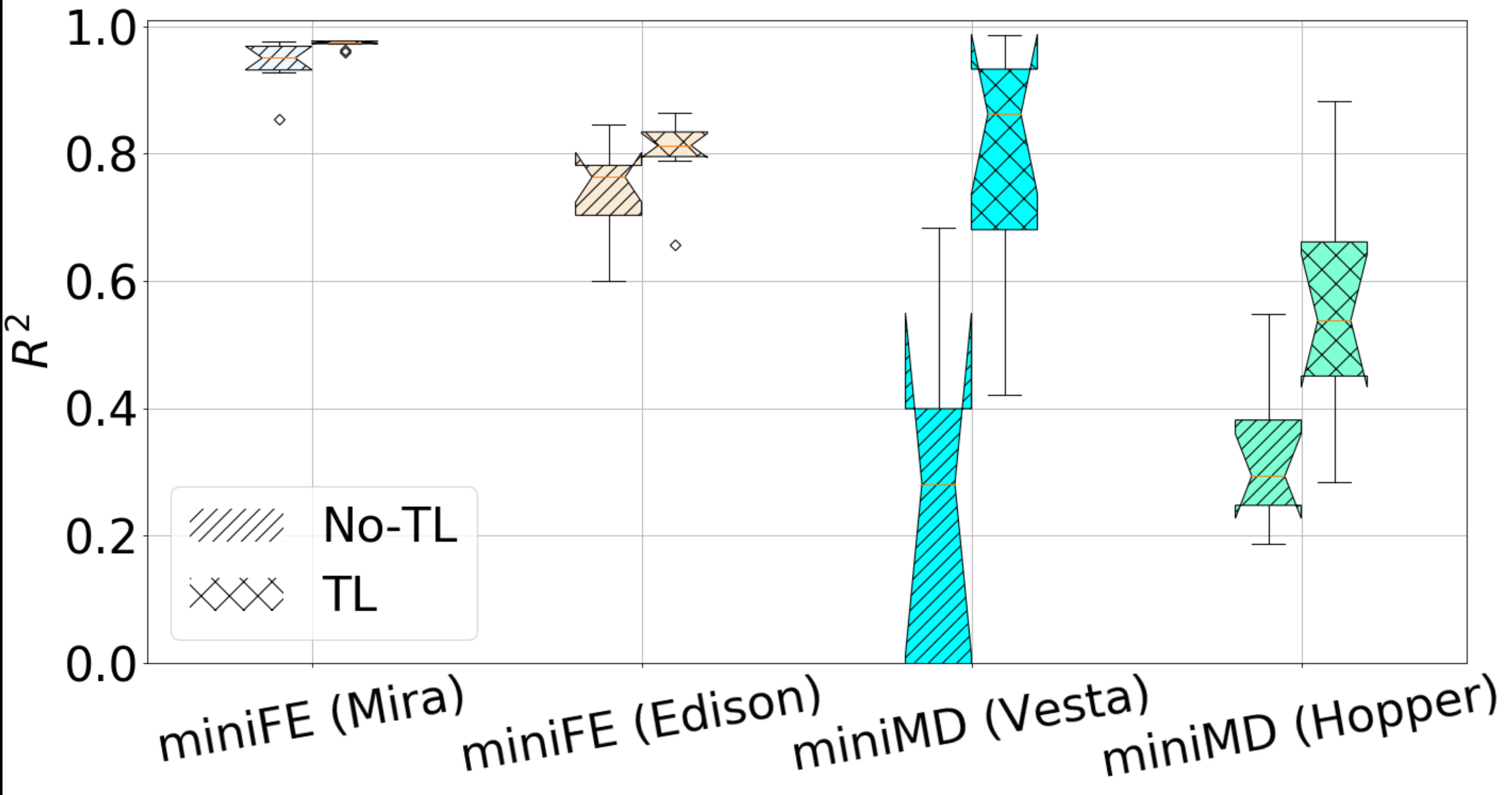



Freeze weights

Retrain weights (1% data from Hopper)

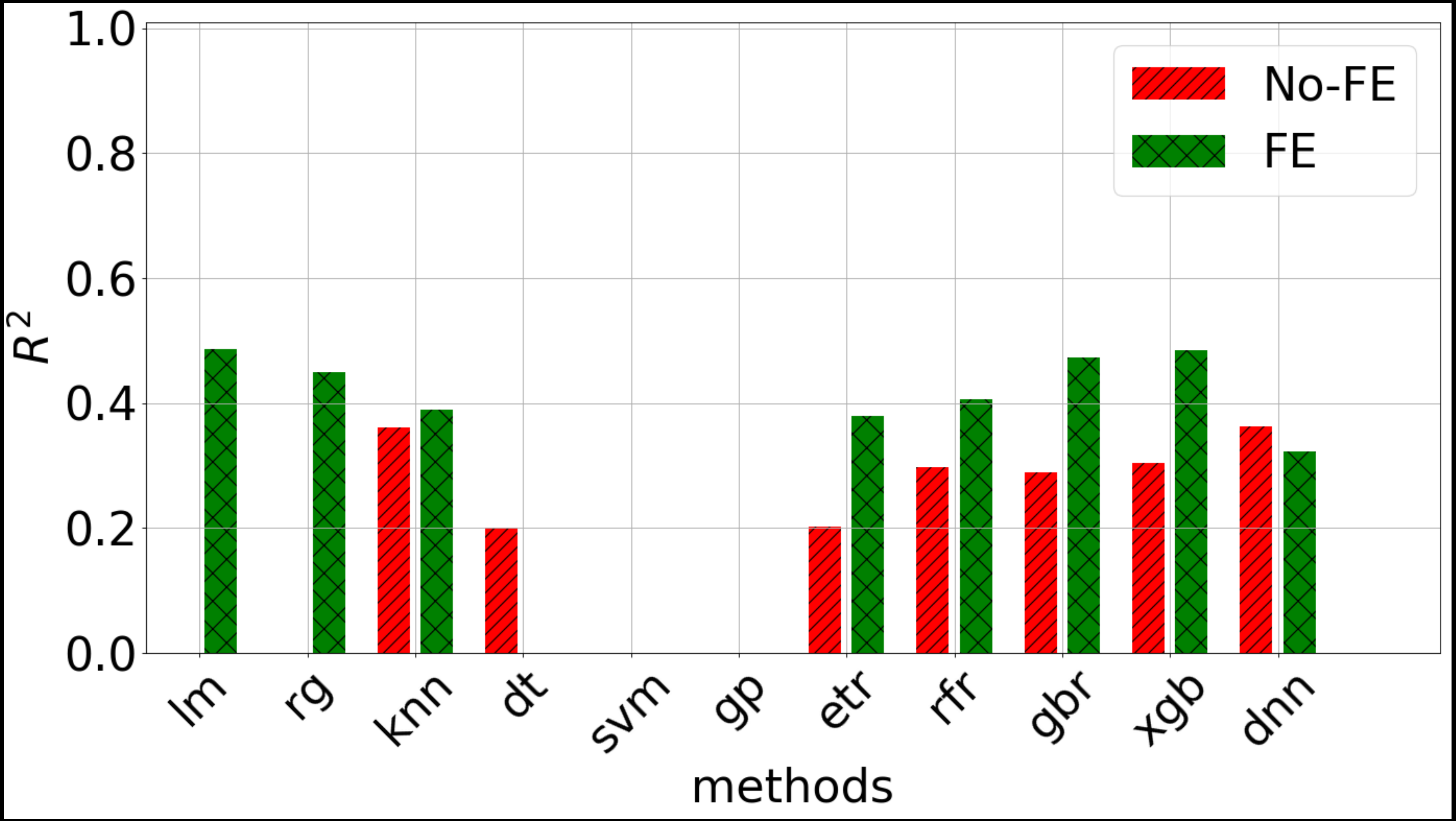
Hopper model

# Transfer learning




- Transfer learning significantly improves prediction accuracy

# Extrapolation




- miniFE: Learn from smaller process count (24–1,152) size to larger count (1,224–1,728)
- ML methods can’t extrapolate but FE helps

# Conclusion

- Explicit domain-knowledge integration/feature engineering significantly improves prediction accuracy
- Algorithmic and computational complexities of the ML methods have a significant impact on accuracy, model training, and inference times
- Bagging, boosting, and deep neural networks leverage large training datasets and produce better accuracy
- Deep neural network can enable transfer learning
- Extrapolation is difficult; domain-knowledge integration helps

# Future work

- Uncertainty quantification for variability
- Active learning for selecting training points
- Domain-knowledge integration
  - Transfer learning
  - Extrapolation
- Applications with I/O
- Subspace characterization
- Job scheduling & autotuning

# Acknowledgements

U.S. DEPARTMENT OF ENERGY


- DOE-ASCR, Early Career Research Program
- Argonne Leadership Computing Facility
- National Energy Research Scientific Computing Center

This work was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research program under contract number DE-AC02-06CH11357.

# Thank you