ARGONNE ATPESC10 EXTREME-SCALE COMPUTING

Supercharge Your Science with Large-Scale Machine Learning

Bethany Lusch
Assistant Computer Scientist
Argonne Leadership Computing Facility
Argonne National Laboratory

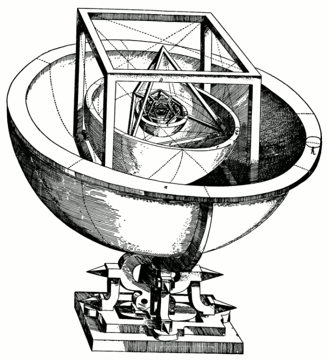
August 11, 2022
blusch@anl.gov

ECP Argonne NATIONAL LABORATORY

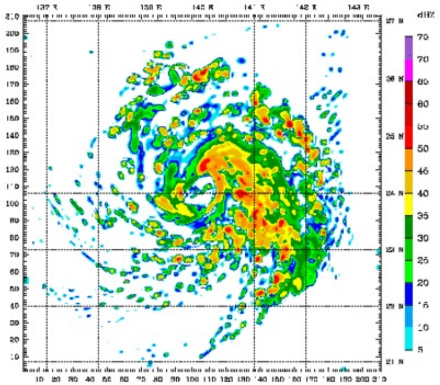
Paradigms of Science

Not mutually exclusive!

1. Experimental



2. Theoretical



3. Computational

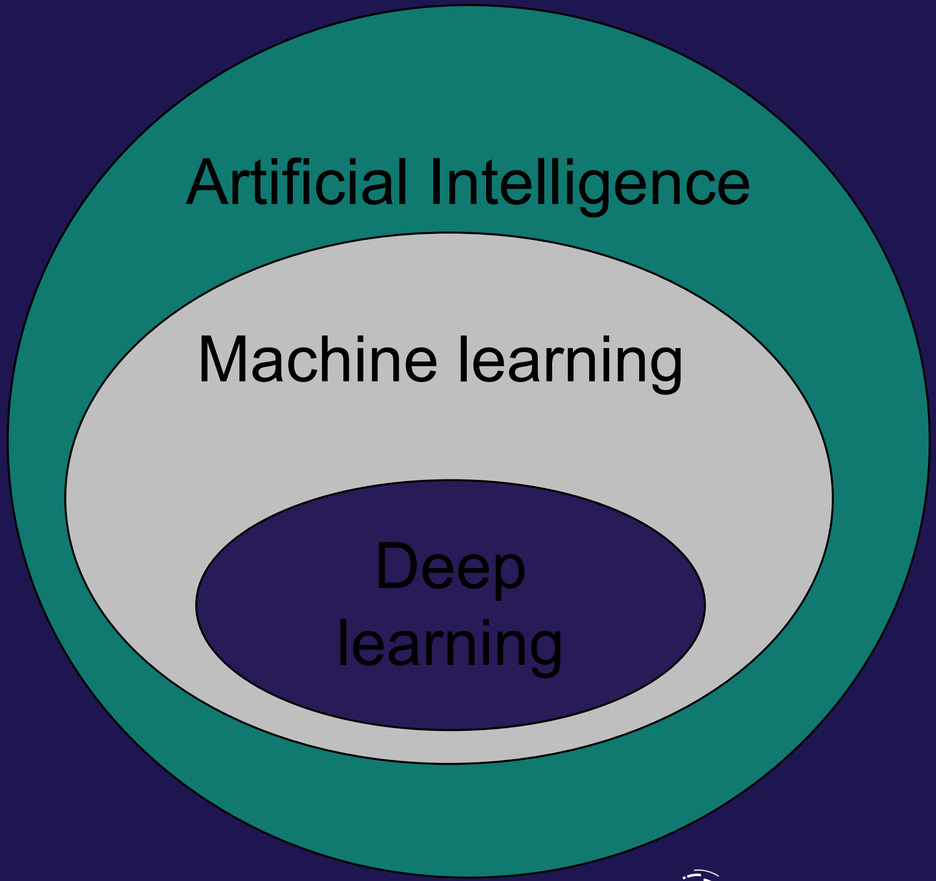
4. Data-intensive

Growing due to:
- More data
- Better computers
- Better methods

Sources: Saint Louis University Madrid Campus, *Mysterium Cosmographicum,* Wikimedia:Atmoz, Sean Ellis

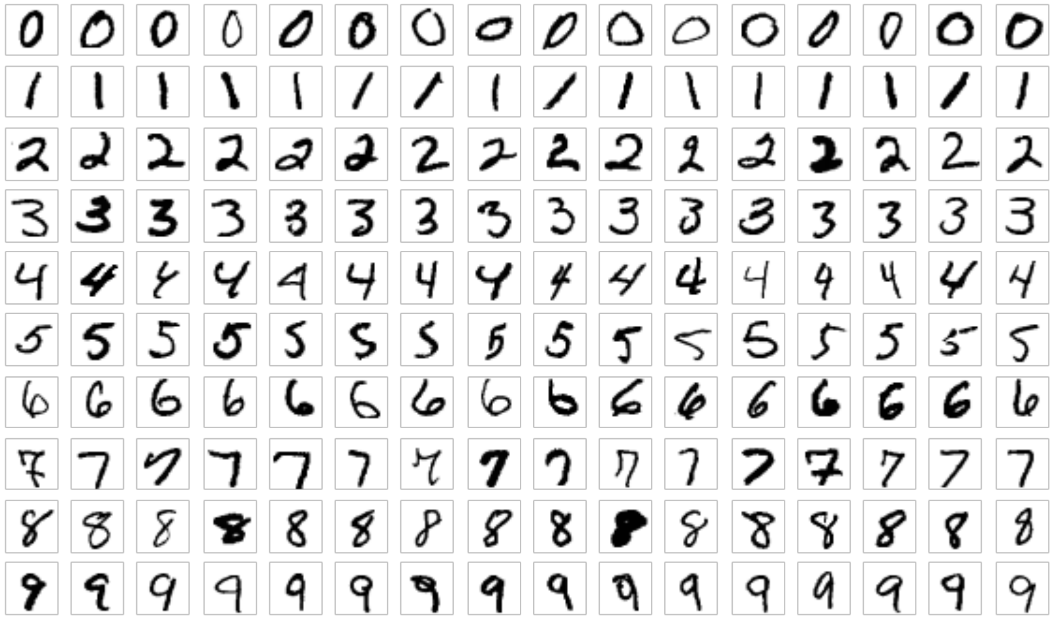
What is machine learning?

And how do you use it for science?



What is machine learning?

Field of study that gives computers the ability to learn without being explicitly programmed



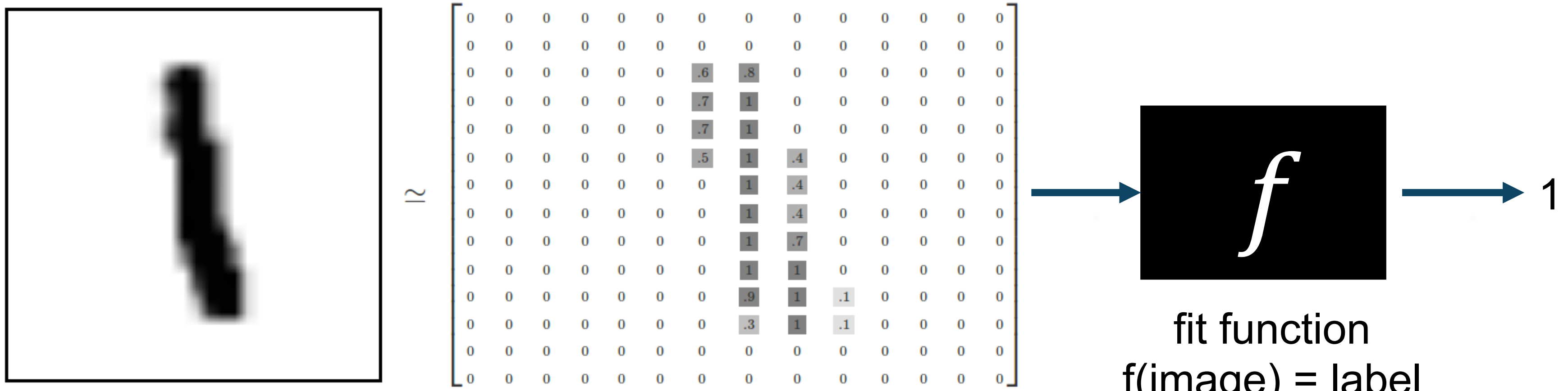
Example: post office wants machine to sort mail by zip code

Want to label each image as a digit 0…9

Explicit programming: IF 80% of black pixels are in middle 30% of image, THEN label as 1.

Reading Zip Codes

Field of study that gives computers the ability to learn without being explicitly programmed



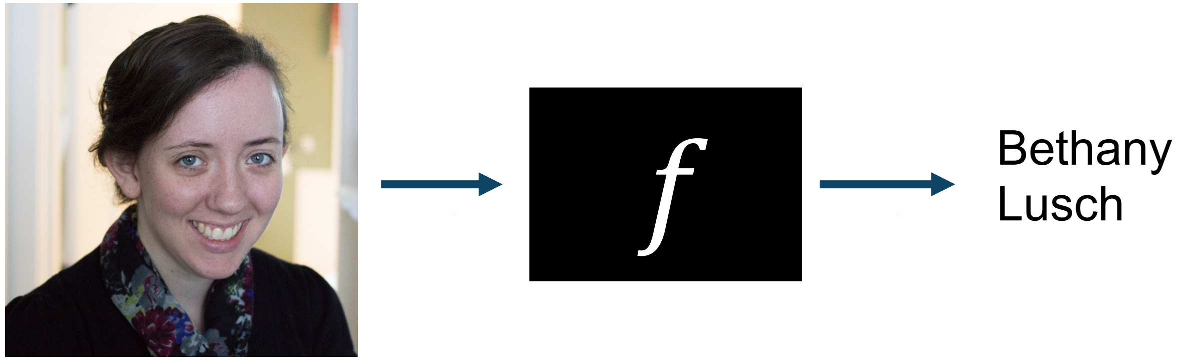
fit function
f(image) = label

by considering many image & label pairs
“learns” as sees more examples

Classification

Have a category label for each data point, learn to categorize

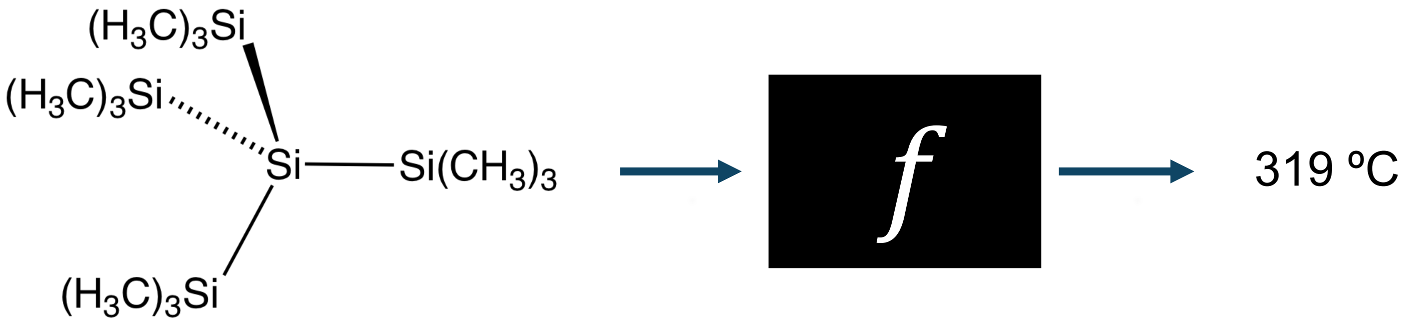
- Learn how to tag Facebook photos with the right name (after we tag many other photos of our friends)
- Learn how to label x-ray images with a diagnosis (after seeing many images labeled by experts)



Regression

Have a numeric label for each data point, learn to predict number

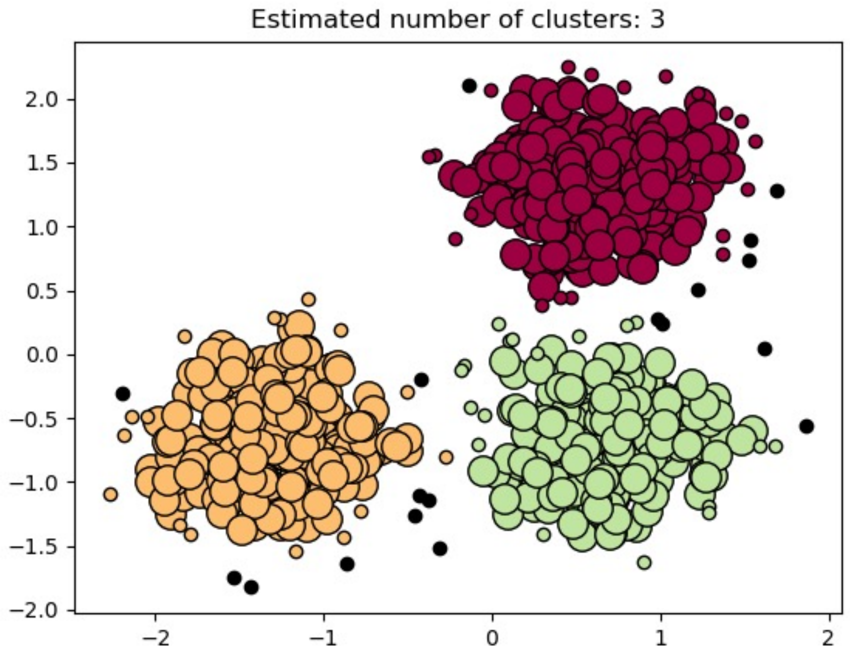
- Learn how to predict stock prices (after seeing historical stock data)
- Learn how to predict the melting point of a molecule (after seeing lots of experimental data)



Clustering

Have an unlabeled dataset, find groups of similar points

- Find communities in a social network (after seeing Twitter data)
- Find subtypes of breast cancer (after seeing data from a bunch of patients)



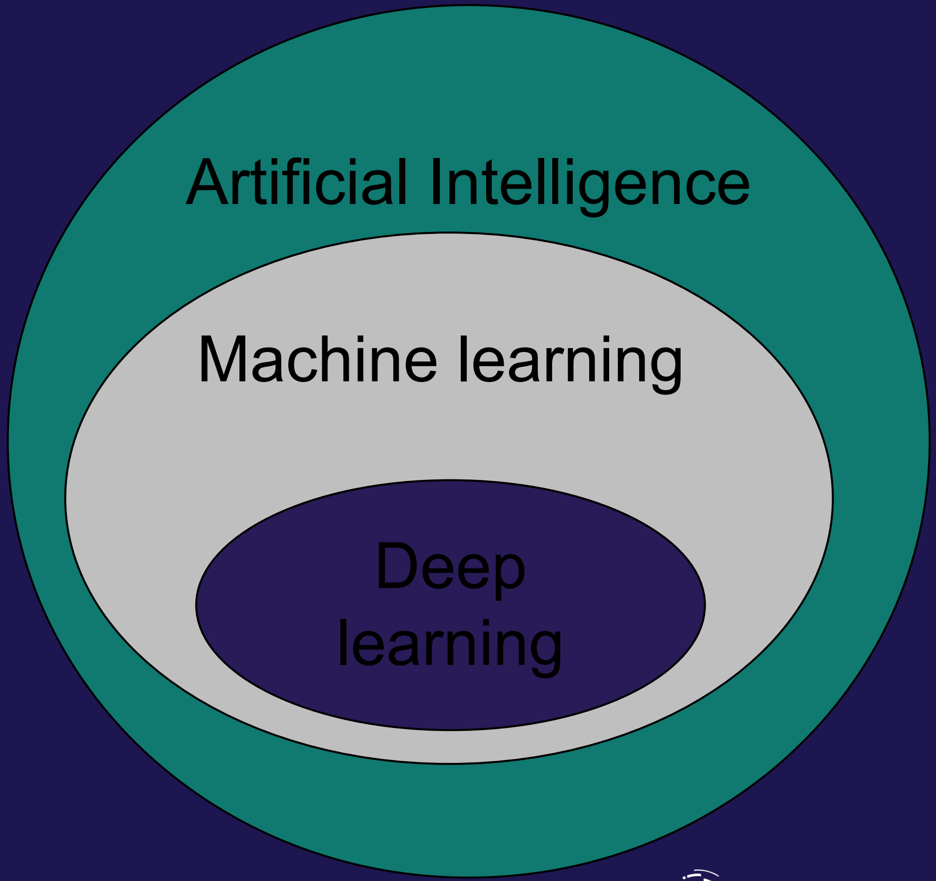
Reinforcement Learning

An agent explores an environment and learns how to get rewarded

- Learn to play Frogger by playing the game and receiving feedback (score)
- Learn to suggest useful chemical reactions

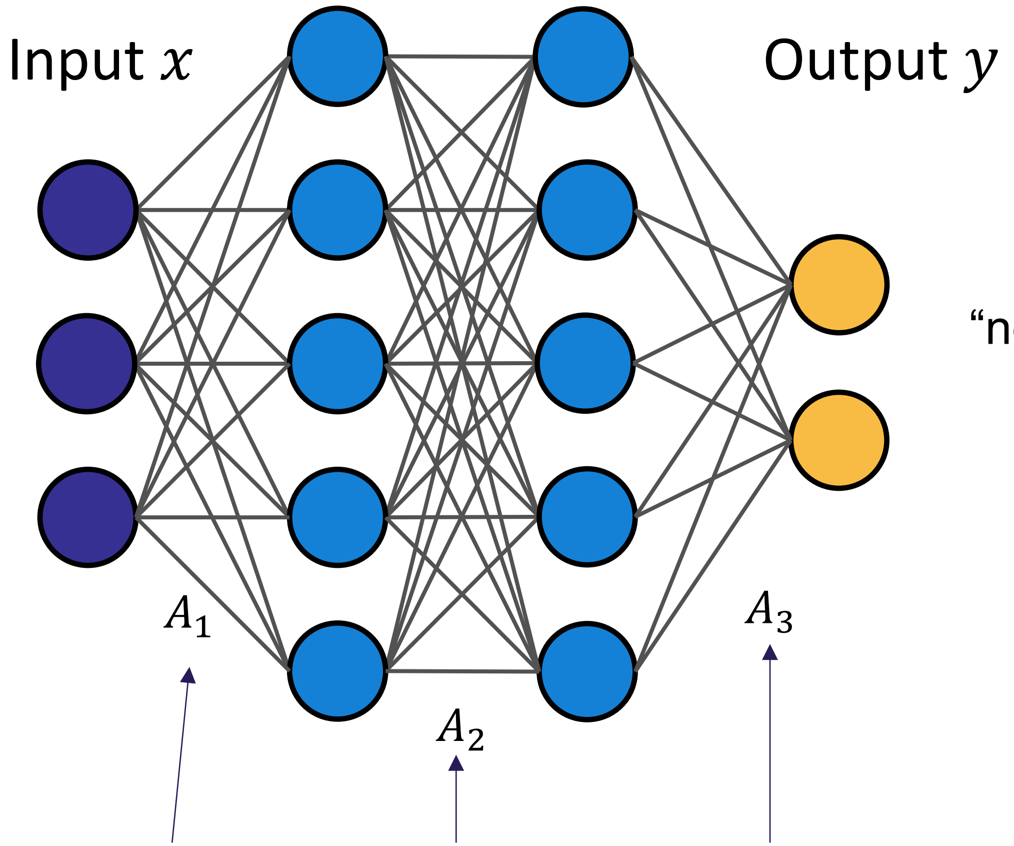
What is deep learning?

And how do we do it on supercomputers?



Preview of tomorrow!

Crash course: deep learning



“neural network”

Some error/”loss” function to minimize

Basic version: each layer multiplies by a matrix, adds a vector, and applies a nonlinear function

“many” layers: “deep” learning

Iteratively improve those matrices and vectors to reduce the error/loss (fit the data)

Preview of tomorrow!

Deep Learning in Parallel

- Lots of linear algebra: fast on GPUs
- Lots of knobs to tune: can try many in parallel (embarrassingly parallel)
- Data parallelism: put different data examples on different ranks. Based on local examples, estimate how to improve the fit. Then communicate (average) across ranks and update the model.

Common case

- Model parallelism: Model doesn't fit on one rank: have to communicate more often
- Spatial parallelism: special case – split each example spatially across ranks, such as large mesh

More challenging, lots of research to be done

Machine Learning Software on Supercomputers

- Deep Learning: Python packages TensorFlow and PyTorch
  - Can program in Python and choose appropriate backends (NVIDIA/CUDA vs. Intel vs. AMD/ROCm, etc.)
  - Can add other packages such as Horovod, DeepSpeed for distributed
- “Classical” machine learning: Python packages such as scikit-learn
  - Vendor-specific acceleration
  - NVIDIA: RAPIDS, Intel: oneDAL backend for scikit-learn, etc.

Main point: you can program using the Python API with lots of high-level functionality, but the backend is fast (CUDA, SYCL, etc.). High portability!

How is machine learning supercharging science?

Cancer Research

- CANDLE project: part of Exascale Computing Project and Aurora Early Science Program
- PI Rick Stevens (Argonne), DOE (4 national labs) and National Cancer Institute
- Science goals:
  - Predict drug responses
  - Understand the molecular basis of certain protein interactions in the RAS pathway, and
  - Develop treatment strategies
- Machine learning contribution:
  - Drug response: learn nonlinear relationships between drugs and tumors
  - RAS pathway: machine learning guides molecular dynamics simulations
  - Treatment strategy: read and encode clinical reports
- Supercomputing contribution:
  - Run simulations and machine learning on the same platform
  - Process large amounts of data
  - Train an ensemble of many models and/or train very large models

https://www.exascaleproject.org/research-project/candle/

Neuroscience Research

- Connectomics project: part of Aurora Early Science Program
- PI: Nicola Ferrier (Argonne)
- Science goal:
  - — Create map of neurons and their connections from brain images
- Machine learning contribution:
  - — Accurate segmentation of neurons
- Supercomputing contribution:
  - — Enables processing increasingly larger datasets, such as moving from mm^3 towards a cm^3 at the 10 nm scale (mouse brain)

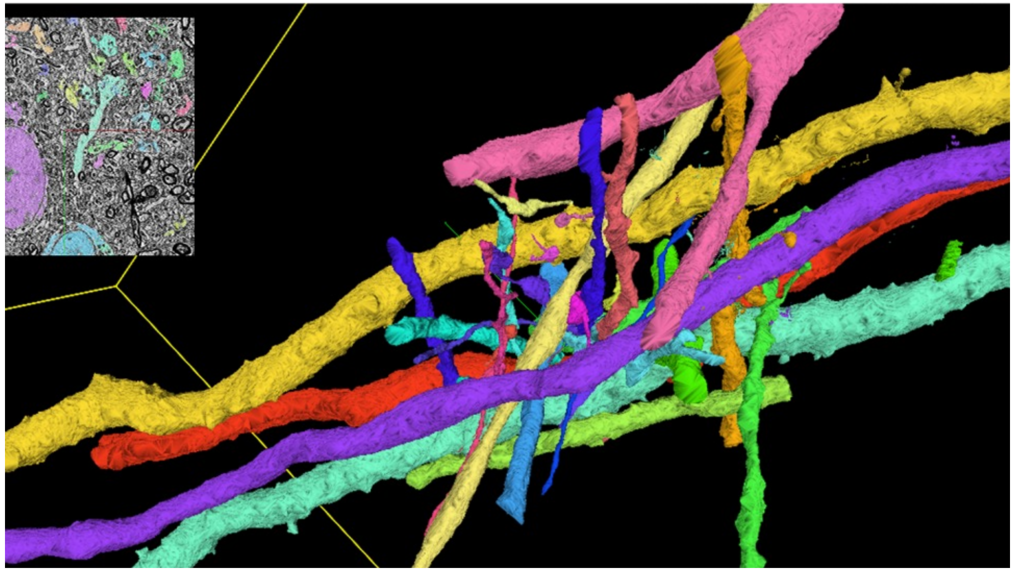


Image: Nicola Ferrier, Narayanan (Bobby) Kasthuri, and Rafael Vescovi, Argonne National Laboratory

https://www.alcf.anl.gov/news/preparing-exascale-argonne-s-aurora-supercomputer-drive-brain-map-construction

Particle Physics Research

- Lattice Quantum Chromodynamics (LatticeQCD) machine learning project for Aurora Early Science Program
- PI: William Detmold (MIT). Team includes Phiala Shanahan (co-PI), Denis Boyda, and others
- Science goal: Calculate possible interactions between candidate dark matter particles and nuclei, then informing experimental sources. (Calculations currently intractable)
- Machine learning contribution: use ML model to improve sampling algorithm (more efficiently sample a target probability distribution), even as move to finer spacing in lattice
- Supercomputing contribution: Need enormous memory as scale to finer lattices and incorporate full physics

https://www.nextplatform.com/2021/08/06/aurora-exascale-system-to-advance-dark-matter-research/

Other Examples Preparing for Aurora

- Predicting & mitigating disruptions in fusion (for a clean energy source)
- Discovering singlet fission materials for efficient solar cells
- Scaling fluid dynamics simulations, such as an airplane tail

Combining Simulations and Machine Learning

Example: surrogate models

- A simplified mapping from inputs to outputs mimicking a more complex process (such as a simulation)
- AKA: an emulator
- We use machine learning to fit a surrogate to training data

Motivation For Surrogate Models

- Simulations can be computationally expensive
- Surrogates can be orders of magnitude faster
- Can compromise: surrogate for just part of simulation

Enabling:

- Exploring parameter space
- Preliminary evaluations of designs (such as of an engine)
- Faster data assimilation (e.g. observational data from sensors)
- Large ensembles exploring effect of uncertain inputs
- Saving compressed representation of simulation due to I/O limitations

Accelerating RANS Simulations

- Science goal: (proof of concept example) simulate flow past a backward-facing step
- Machine learning contribution: replace one PDE solve
- Prediction from machine learning model fed back into simulation to solve rest of equations
- Results: reasonable accuracy at 5x – 7x faster

- Challenges:
  - Communicating between simulation and machine learning library
  - Designing problem (inputs and outputs) to enable some generalization

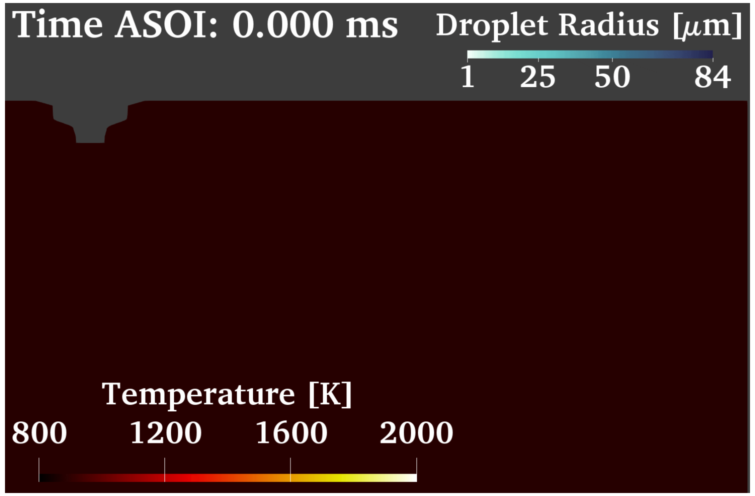
arXiv:1910.10878
Computers & Fluids 2021
Romit Maulik, Himanshu Sharma, Saumil Patel, Bethany Lusch, and Elise Jennings (all Argonne)

ICEF 2021 Mondal, Magnotti, Lusch, Maulik, Torelli
SAE 2021 Mondal, Torelli, Lusch, Milan, Magnotti

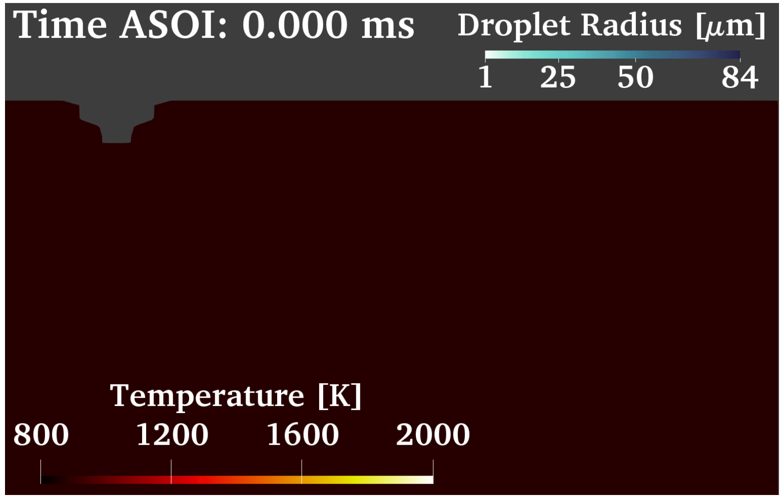
Accelerating Engine Design

- Science goal: design an efficient automotive engine
- Machine learning contribution:
  - Accelerate exploration of design parameter space
  - Replace expensive part of simulation with surrogate model
- Prediction of flow fields exiting the injector fed into rest of the simulation
- Results: Surrogate is **38 million times faster** (but then still run less expensive part of simulation)

Injection Map from CFD ("Truth")



Injection Map from Emulator



Accelerating Weather Prediction

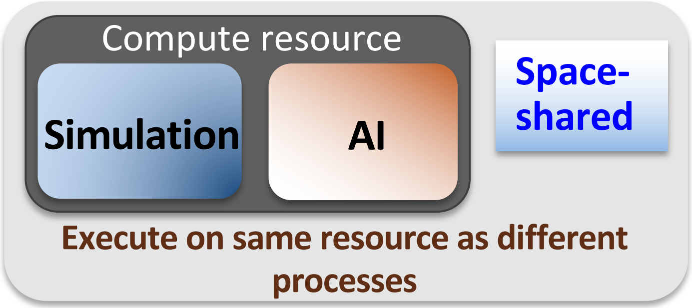
- Science goal: (proof of concept) predict geopotential height on the weather scale
- Machine learning contribution:
  - — Replace expensive simulation with faster (and differentiable) surrogate model
  - — Then apply data assimilation to the surrogate model
- Results: data assimilation is O(1000) times faster
  - — Assimilating into fast surrogate, and gradients are easy
- Vision: able to predict more quantities, integrate more observational data, and move to climate scale. Replace only part of climate model.

Maulik, et al. “Efficient high-dimensional variational data assimilation with machine-learned reduced-order models” Geoscientific Model Development, 2022

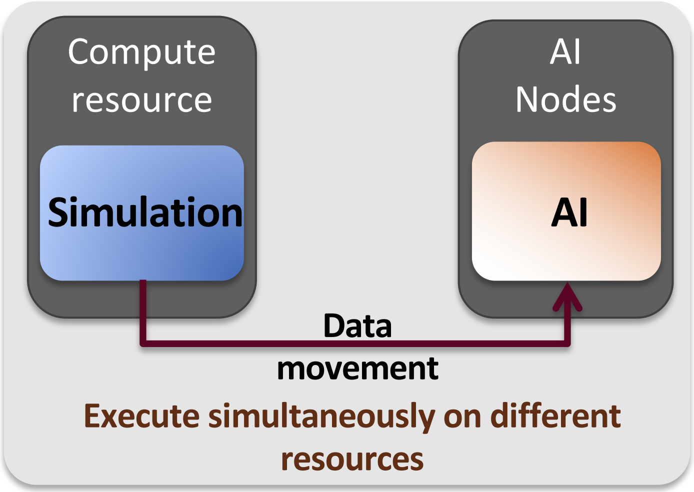
Coupling ML and Simulations

Example Modes

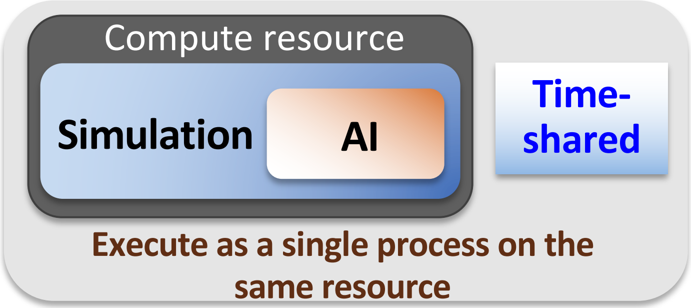
Loosely-coupled



Example: simulation running on some CPUs, data is passed to some GPUs where a surrogate model is trained (skip I/O bottleneck)



Tightly-coupled



Example: at every step of simulation, apply surrogate model to replace one component

Figure adapted from Venkat Vishwanath

"A terminology for in situ visualization and analysis systems" by Childs et al. 2020

Open Challenges

Challenges with Machine Learning

Or areas of open research!

- Want to generalize well to future data (not “overfit”)
  - Extrapolation in terms of the input space is especially rough/impossible
- Often hard to interpret
- Typically lacking in guarantees
- Want to build trust, such as by including uncertainty estimates
- Want to incorporate domain knowledge instead of wasting compute relearning it
- Can be hard to troubleshoot

Need careful formulation of problem and proper held-out test data

Challenges with ML for Simulations

Or areas of open research!

- Limited training data, especially when each simulation is expensive
  - How do you choose diverse simulations with limited budget?
- The larger the simulation, the easier it is to overfit?
- ML is more commonly trained on smaller examples – non-trivial to train when even one example (one time step) doesn't fit in one GPU
- Unclear how to efficiently handle unstructured meshes
  - Convolutional layers are efficient for images, but can't be straightforwardly applied here
- Time-series models like RNNs struggle with long-term stability

Or areas of open research!

Challenges in Coupling ML and Simulations

- Keeping resources busy
  - Are certain processors always running simulations and others always running ML?
  - If not, can you dynamically adjust?
  - Do these pieces need different hardware, like simulations on CPUs and ML on GPUs? Do you have the right balance near each other?
- Low overhead if passing data
- Software issues, such as
  - Communicating between C++ simulations and ML in Python
  - If the simulation is distributed but not memory-intensive, do you use fewer nodes for the ML, requiring a different domain decomposition?
- If simulations are too large to save and doing online training:
  - Can you return to older data?
  - Are the batches diverse? Are the arriving in a special order?
- If deploying ML “online” within a simulation:
  - Do errors accumulate too much, causing instabilities?
  - Does the ML pass something non-physical to the simulation?
- If the surrogate is just for part of the simulation: is the training done “off-line” without feedback from the simulation? Is there a computationally-feasible way to train end-to-end?

In summary:

Large-scale machine learning can enable tackling scientific questions previously out of reach

But many open challenges (or potential research)

Thank you!

blusch@anl.gov