This seminar is being broadcast live on YouTube.

Find the talk in the channel: https://bit.ly/2BgHY5G
The direct link to the broadcast *should* be, if all goes smoothly:
https://bit.ly/35LMKX5

Warm welcome to the remote viewers!

Please ask questions in the comments section

ML and Scientific Computing

Peter Braam
peter@braam.io

me

math physics @oxford (gauge theory, CFT)

distributed systems cs @cmu

@6 startups & jobs @3 acquirers - Lustre

SKA @cambridge

@oxford

work with 100’s of largest compute centers and many major system & CPU/GPU vendors

Origin of this seminar series

I worked extensively on HPC infrastructure in industry and more recently in Cambridge for the SKA telescope.

I and Prof Ian Shipsey ran a conference in London about *AI for CERN and SKA*.

We quickly decided it would be worthwhile to run this seminar.

ML for physics:

- Growth of the field is phenomenal.
- Extremely difficult to follow along.

Today's lecture - perhaps unusual

However - *I'm a computer scientist, technologist and mathematician. I am not a physicist, and not an ML specialist*.

I'll summarize structure and a few aspects of how we look at this field.

All other seven lectures will focus on physics. Some on theoretical physics, most on experimental physics.

I hope to leave a little time for discussion, and learn from the audience where this perspective might be improved. Also, I'm happy to answer short questions as we go along.

Physics and ML

- establish terminology
- attempt at categorizing

Basics of physics and ML

inference: using a trained ML model to create new computed data from input.

training: using experimental or computed data to discover & train an ML model of a physical practice

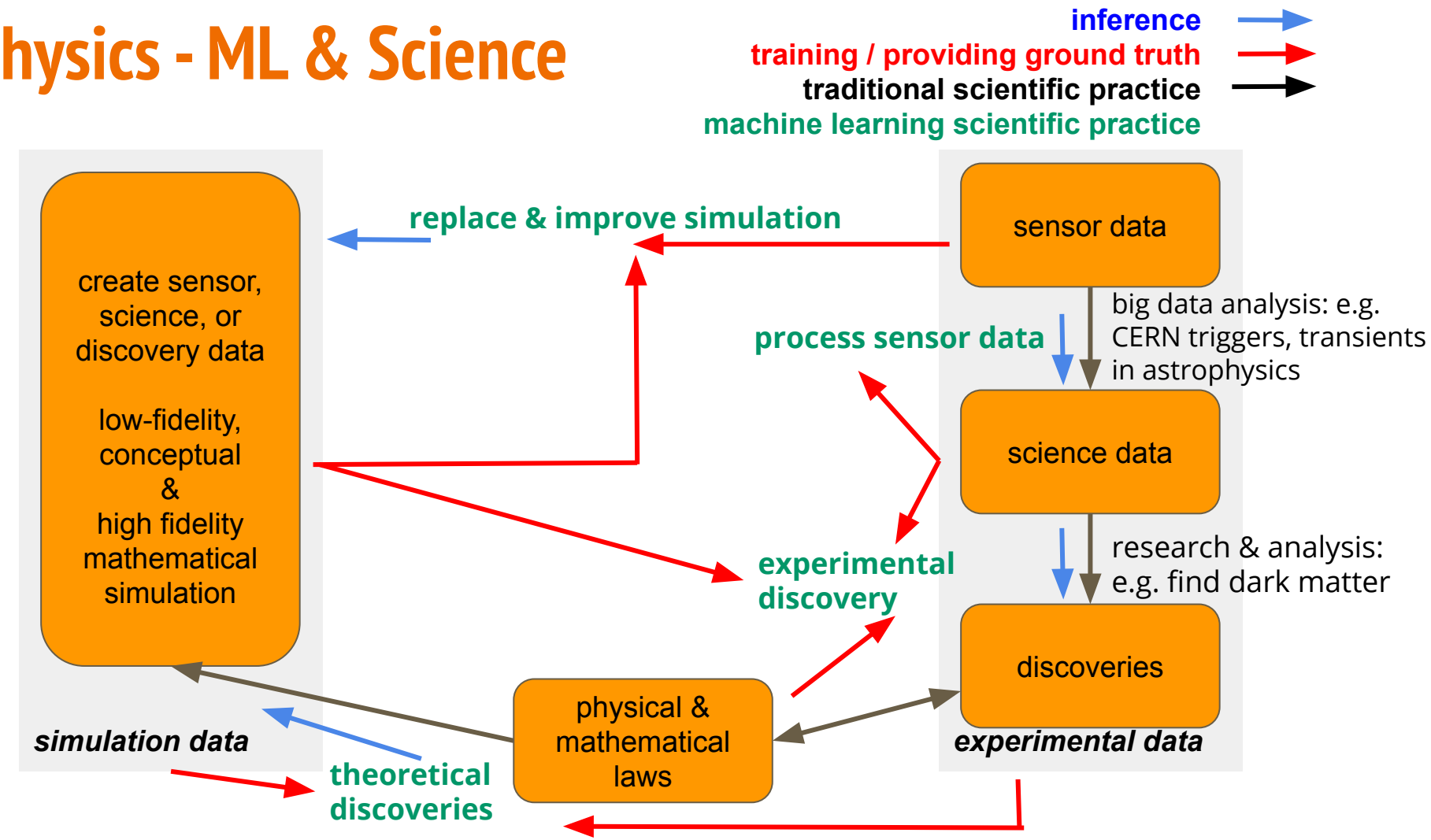
traditional scientific practice: all aspects of discovery and physics data handling without ML methodology

machine learning scientific practice: creating and using ML models

Identify 4 areas of ML scientific practice (probably some are missing):

1. **sensor data**: process sensor/measured data to create scientific data
2. **simulation:** refine and replace traditional simulation software
3. **experimental discovery:** make discoveries in experimental data
4. **theoretical discovery:** discover and compute theoretical data:

Physics - ML & Science



Seminar Series - overview

Simulation:

Wk 1 Oct 17: Peter Braam - ML approaches for scientific computing

Sensor data & simulation:

Wk 2 Oct 24: Professor David Rousseau, LAL Orsay: 'Two computing challenges for particle physics: the tracking challenge and event simulation with generative adversarial networks'

Simulation:

Wk 3 Oct 31: Professor Giles Louppe, Liege, 'Neural likelihood-free inference'

Experimental discovery:

Wk4 Nov 7: Vesna Lukic, Hamburg, 'Deep learning techniques applied to radio astronomy'

Seminar Series - overview

Theoretical discovery:

Wk 5, Nov 14: Professor Sergei Gukov, Caltech, 'A nobel prize to neural net?'

Simulation:

Wk 6, Nov 21: Dr Peter Battaglia, DeepMind 'Learning structured models of physics'

Experimental discovery:

Wk7, Nov 28: Dr Laurence Levasseur, Montréal 'Analysis of strong gravitational lensing data with machine learning'

Theoretical discovery:

Wk 8, Dec 5: Professor Yang-Hui He, London, ''Machine learning mathematical structures'

ML & Simulation

- characterize the fields
- categorize ML / HPC relationship
- motivation

What do we want to talk about today?

Simulation/HPC - *Very big* business since 2000 (~$50B/ year). Large scientific experiments analyze all data with HPC. In *classical physics simulations* solving PDE's are central, but there are others (e.g. MtCarlo). Now pervasive in industry.

ML - Growing into a ***huge*** business (already $60B/year). Focused on a subset of computational methods found in HPC. Moreover, ML software and hardware frameworks are maturing faster than HPC. Motto: "could be more revolutionary than electricity"

ML and HPC

Geoffrey Fox (2019) c.s. categorized (*some*) aspects of the relationship:

MLforHPC:

- **MLaroundHPC:** an ML model becomes a ***surrogate*** for a simulation
- **MLControl:** influence selection of simulation with surrogates
- **MLafterHPC:** study results of HPC with ML
- **MLautotuning:** tune HPC with ML
- **MLrunsHPC**: HPC software and hardware runs (not in Fox's categories)

HPCforML:

- **HPCrunsML**: ML infrastructure for HPC computations
- **SimulationtrainedML**: HPC simulations train ML model to influence experiments or other simulations

This talk contains examples of **orange aspects**.

https://arxiv.org/abs/1902.10810, Learning Everywhere: Pervasive Machine Learning for Effective High-Performance Computation

Geoffrey Fox, James A. Glazier, JCS Kadupitiya, Vikram Jadhao, Minje Kim, Judy Qiu, James P. Sluka, Endre Somogyi, Madhav Marathe, Abhijin Adiga, Jiangzhuo Chen, Oliver Beckstein, Shantenu Jha

Motivation

Why MLforHPC?

HPC is older subject. Real limits have been encountered:

- Some algorithms like adaptive mesh refinement have seen almost no performance improvement in 30 years!
- Storage, memory and network bottlenecks for most problems
- HPC is costly

ML brings completely new methods to HPC. Software & hardware for ML has radical new aspects.

Why HPCforML?

HPC has the biggest super-computers. Possibly can push the limits on ML computations.

Maybe this is not so different from usual HPC: simply pushing the limits

Specifically we will look at

(1) Some examples of surrogates

- Structure formation in early Universe
- Plasma Ignition
- Great variety of opportunities

(2) Mathematical questions re surrogates:

- **ML**: depends on data, while true mechanism may remain unclear
- **Simulation**: bottom up mathematical derivation
- Relates to **interpretability** of some models

(3) Example of ML running HPC

- ML hardware landscape
- ML software
- Benchmark results

Acknowledgement: The discovery and results are those of the cited authors. My own interests concern the generalization of the employed methods.

Learning to predict the cosmological structure formation

Reference: Siyu He, Yin Li, Yu Feng, Shirley Ho, Siamak Ravanbakhsh, Wei Chen, and Barnabás Póczos

PNAS July 9, 2019 116 (28) 13825-13832; first published June 24, 2019 https://doi.org/10.1073/pnas.1821458116

Problem statement: Take initial distribution of mass & velocity in (early) universe. Study its evolution, recognize structure formation (e.g. galaxies).

Sample: 10,000 simulations

Ground truth: an expensive simulation - FastPM

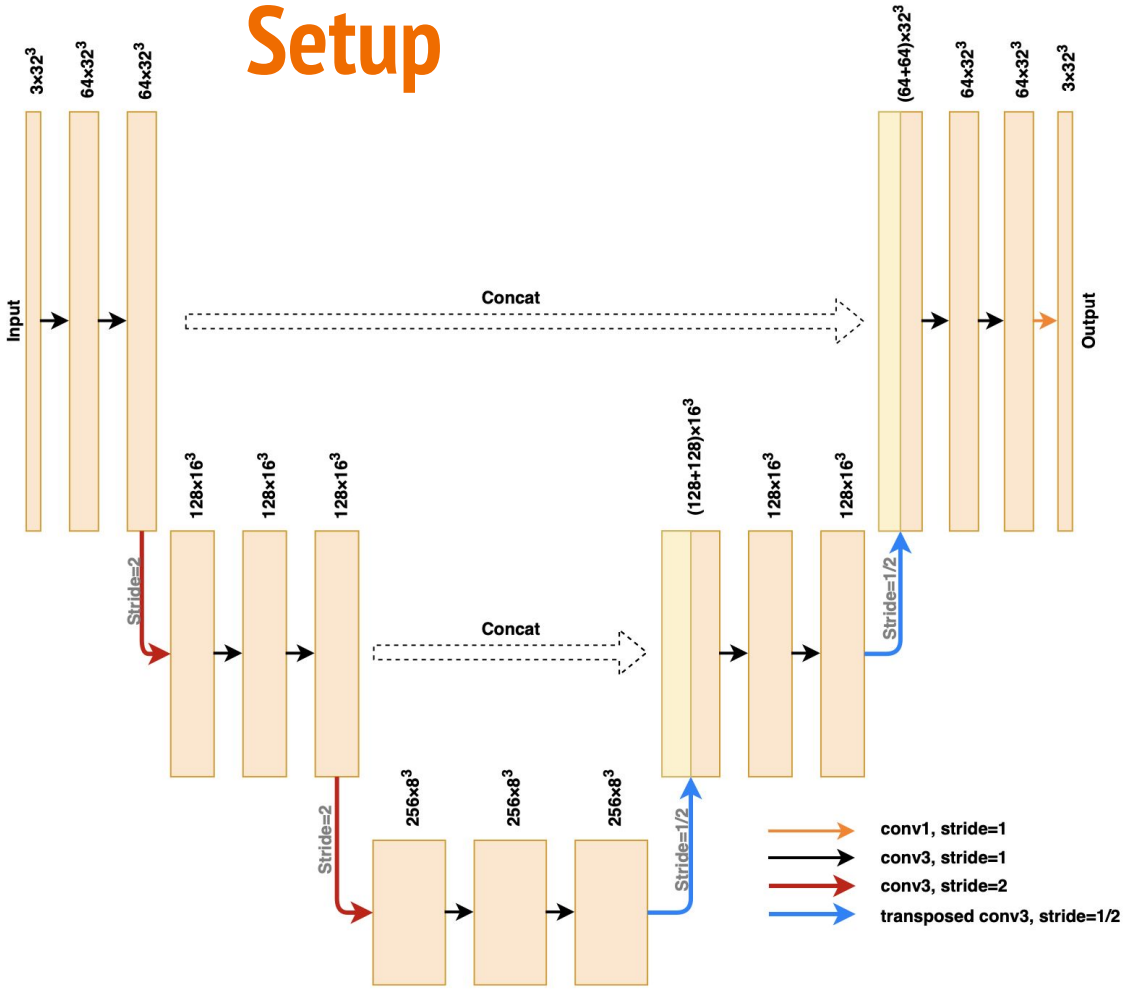
Cosmological structure formation: D3M model

- **Ground truth for training:** Fast-PM multi particle simulation
- **Model**: evolve along linear trajectories (so-called Zeldovich approximation), followed by a U-net.
- **Evaluate**: Compare with 2nd order perturbation theory (2LPT) and Fast-PM
- Also evaluate: different cosmological parameters & multi-point correlations

Result: ML model is more accurate, has much lower computational cost than 2LPT. Appears robust under parameter changes.

ML is 5x faster than FastPM, more accurate than 2LPT

Setup



Discretize $200*10^9$ ly^3 of early universe into 32^3 voxel cubes with N-body particle configurations

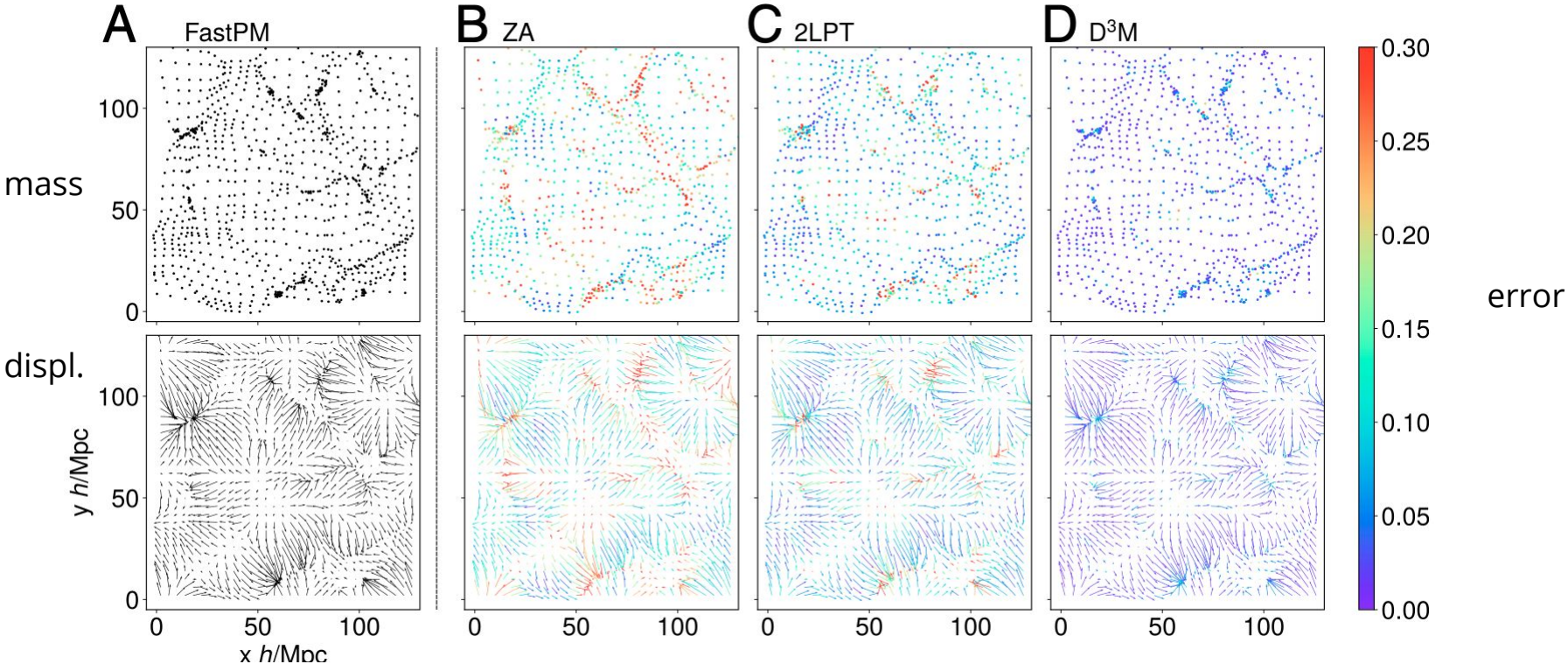
Displacement field F is initial velocity vector at each grid point.

Input to D3M model = U-net applied to Zeldovich Approximation displacement field

Loss is $|F^{FastPM} - F^{D3M}|_{\text{mean square}}$

Evaluation of accuracy of model

Also included are multi-point correlation functions for evaluation of the model, as customary in physics



Suggestions / questions

- Is there a different model that would eliminate the ZA approximation altogether, and solve the equation purely with ML? (Answer is almost certainly “yes”)

- The uNet has ~10^7 parameters. The (discretized) translation & rotation group ($(\mathbf{Z}/32\mathbf{Z})^3$ x rotations) introduces preservation of momentum. A model that is invariant under this has < 100 parameters left. Can this be leveraged?

Inertial Confinement Fusion (ICF)

Showing Enormous Variety of ML opportunities in a very difficult problem

- **What are ICF, NIF, LLNL**
- **Ultra large simulations**
- **Very difficult plasma physics**
- **transfer learning**

National Ignition Facility @ Lawrence Livermore

Built between 1996 and 2009. Studies plasma ignition.

Two pillars in US “weapon stockpile stewardship program”:

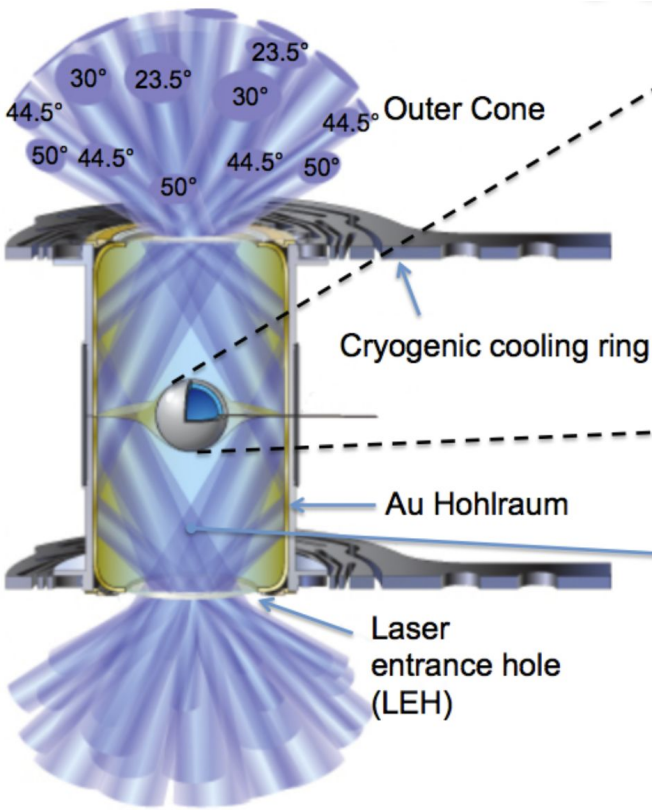
- #1 super computers
- NIF = National Ignition Facility

What is NIF about:
https://www.youtube.com/watch?v=yixhyPN0r3g&t=3m41s&end=5m1s

Simulations of pellet:
https://www.youtube.com/watch?v=KOZIx5JUHbc&t=17m50s

NIF laser facility: size of 3 soccer fields

Hohlraum and Pellet



Hohlraum (10 mm) simulation: extremely difficult, sometimes inaccurate

Pellet (1 mm) simulation: also complicated, but much simpler.

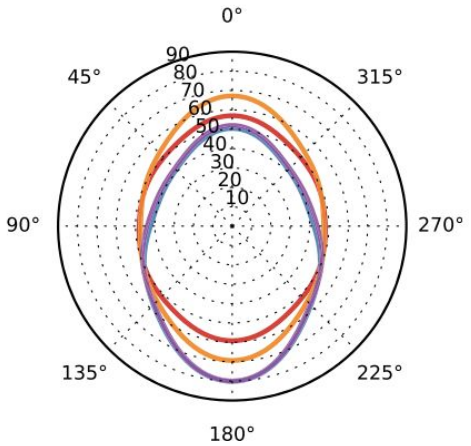
Experimental discovery in simulation data

ICF group deals with **profoundly difficult differential equations**: shock waves, turbulence, multi-physics. Results can be fairly inaccurate.

Surrogates have been created: 10's of Petabytes of simulation training data, months of simulation time.

Surrogates compute in <1% of simulation time.

Profound outcomes: through search in ML surrogate generated data it was discovered that - e.g. ovals may be better than spherical shapes are better than spherical shapes!



Zonal flow generation in inertial confinement fusion implosions
J. L. Peterson, K. D. Humbird, J. E. Field, S. T. Brandon, S. H. Langer, R. C. Nora, B. K. Spears, and P. T. Springer Citation: Physics of Plasmas 24, 032702 (2017); doi: 10.1063/1.4977912

Transfer Learning

Improve post-shot (pellet only) simulation.

Replace last layers in neural network.

Extraction of training data is elaborate combination of ML, experimental and simulation work (novel use of auto-encoders, I think).

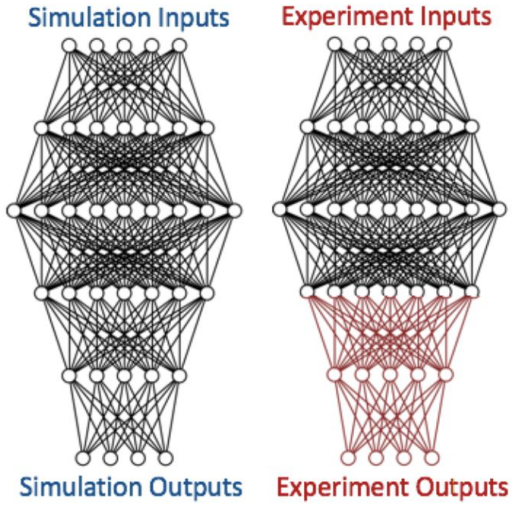


Figure 6.1: To transfer learn from simulations to experiments, the first three layers of the simulation-based network are frozen, and the remaining two layers are available for retraining with the experimental data.

Ref: PhD thesis: MACHINE LEARNING GUIDED DISCOVERY AND DESIGN FOR INERTIAL CONFINEMENT FUSION

KELLI DENISE HUMBIRD, Texas A&M 2019

Mathematical Perspective

mostly questions

- **what operators are ML models?**
- **limiting and scale behaviour**

What models? What operators?

Model Choice

Challenge: formulate a **systematic way** to find a surrogate for a class of simulations

Systematic approaches do exist to transition differential equations to e.g. finite element models.

The discovery of most ML models appears to require good intuition but is experimental.

What are ML surrogates as solution operators?

In simulation usually the mathematical origin of the computation is known: for example, approximate evolution of a differential equation.

Trained ML surrogates map inputs to outputs.

Question: What kind of operators are we dealing with?

Scale & limits of networks

Scale

Convolutional Neural Network inputs can represent initial conditions in space. The scale at which the initial conditions are sampled must be respected by the network.

E.g. in cosmological structure formation, the network cannot predict sub-cell evolution.

However, newer numerical methods like adaptive mesh refinement (AMR) adjust automatically.

Challenge: Create scale adaptive ML networks

Limits

Questions: As ML network gains width and depth do ML surrogates approximate the mathematical solutions? Are there phase transitions? What are the continuous limits?

Is there a relationship with study of over-parametrized networks? Reminiscent of limits (renormalization) in statistical physics, e.g. of Ising model

Ref: Sensitivity and Generalization in Neural Networks: an Empirical Study, Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, Jascha Sohl-Dickstein

Algebra of neural networks

Evolution of physical phenomena are generally composable: the operators transitioning between [T1, T2] and [T2, T3] can be composed to get the evolution from [T1, T3].

Classically this true for flows of differential equations, it also holds in QFTs.

Should be true for ML surrogates.

In the case of structure formation this would demonstrate that the effect of the composition of two u-Nets approximates that of a single one. Notice that the depth of ML network doubles!

Question: What is the algebraic & geometric structure on the space of neural networks and for what is such a structure useful?

MLrunsHPC

Reference:

TensorFlow Doing HPC, Steven W. D. Chien, Stefano Markidis, Vyacheslav Olshevsky, Yaroslav Bulatov, Erwin Laure, Jeffrey S. Vetter

- **Run some HPC benchmarks using TensorFlow**
- **very encouraging**
- **not “production ready”**

Run HPC benchmarks on Tensorflow: summary

Evaluated & good outcome:

- Programmability in TF is very good - short clear programs
- Performance of RDMA transport is very good
- Scaling number of GPUs is good

Evaluated - needs fixing:

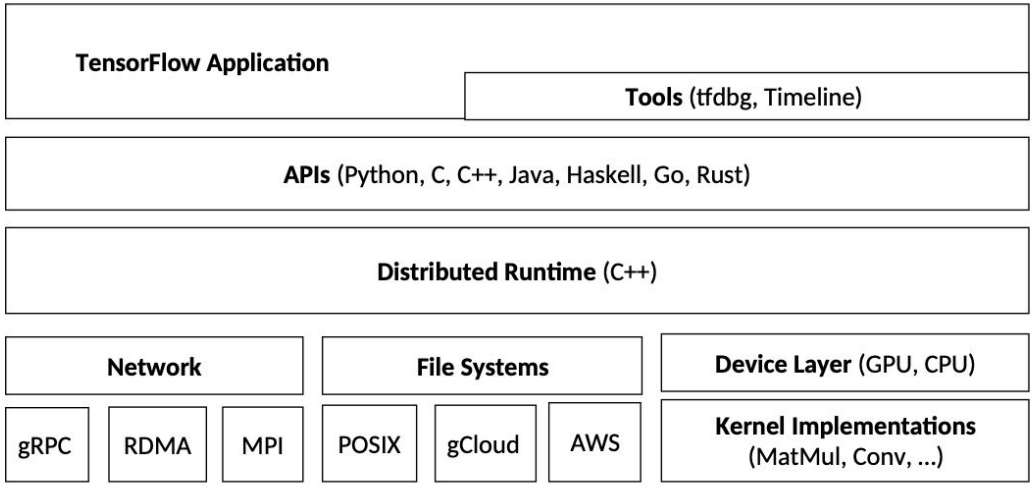
- Python itself too slow for e.g. slicing
- Some locking on queues too coarse

Next steps:

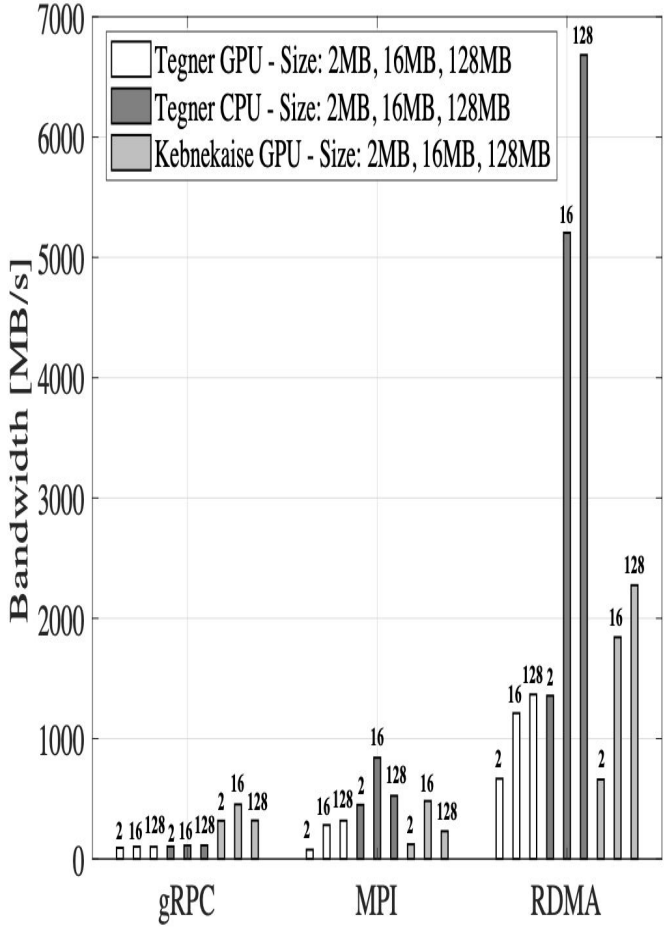
- Achieve comparison with true benchmarks
- Scale the number of nodes
- Leverage TF specific hardware

TensorFlow implementation & networks

TF has a data flow graph model, deployable on many different kinds of distributed hardware devices

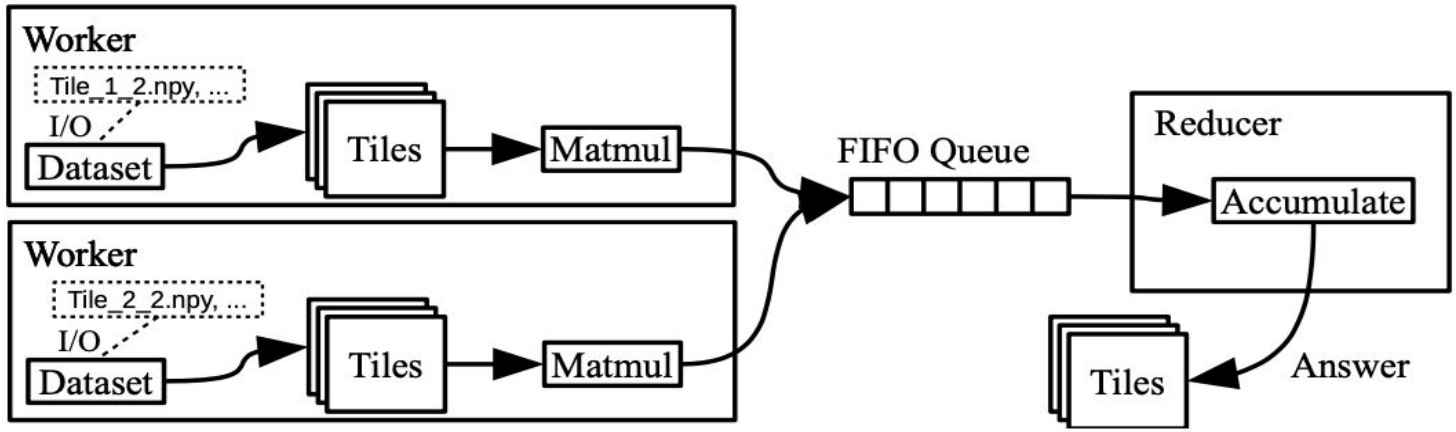


deemed very important for HPC



RDMA works extremely well

Distributed Tiled Matrix Multiplication



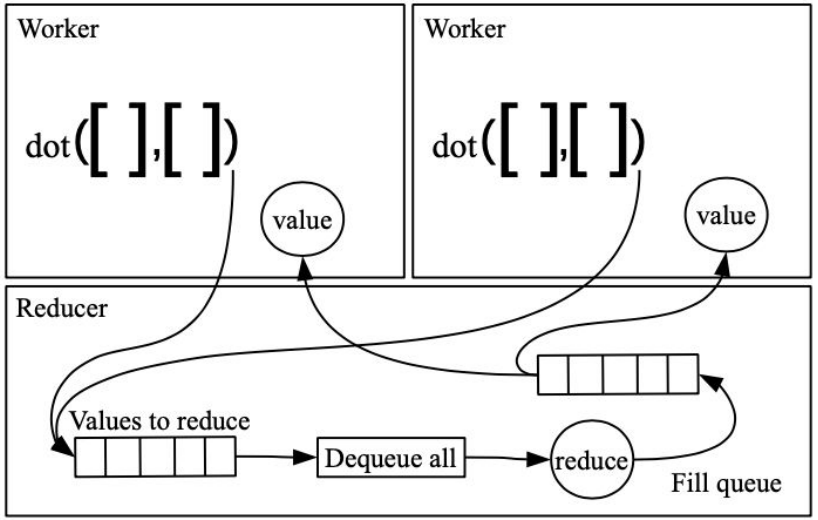
TF implementation of distributed matrix multiplication: effective and high performance.

Matrices are larger than single node memory. Workers and reducer run on different systems, and exploit different numbers of GPU's. Performance scaling on different clusters is good.

More complicated algorithms: easy to write

Conjugate Gradient (CG) & Fast Fourier Transform (FFT) involve more complicated programming:

- CG Reducer must wait for all results from workers. TF queues implement this.
- Tukey-Cooley FFT requires slicing of data. Python cannot do that fast.



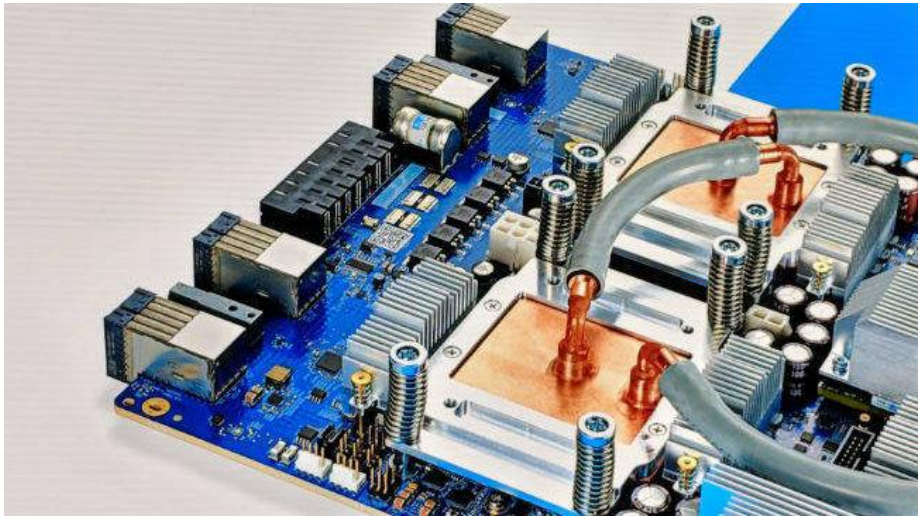
MLrunsHPC

TensorFlow.org. Publications about TPU's.

- **Domain specific chips for ML are emerging (dozens of startups!)**
- **10x - 100x higher HW capabilities**

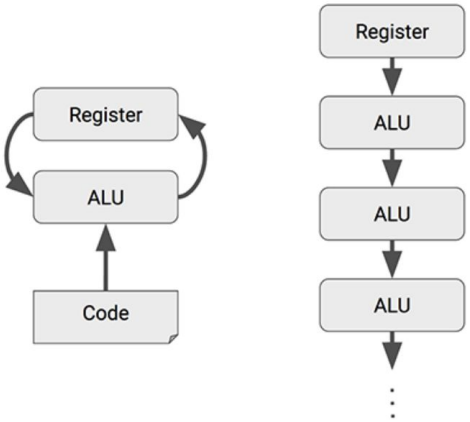
TPU - PCI ML accelerators: cards & clusters

TPU v3 PCI accelerator card

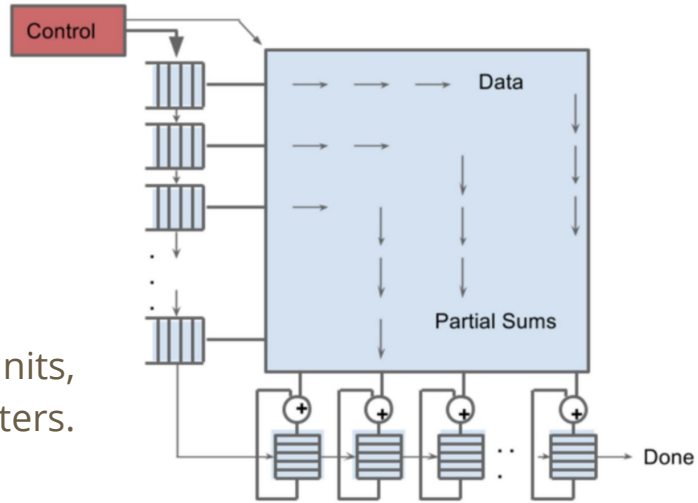


TPU v3 POD (“cluster”)

Example of ML hardware: TPU chips



TPU chips have 4 **systolic arrays** MXU (matrix multiply unit) reducing memory accesses by ~100x:



Pass data between ~100K ALU's. Small processing units, using a global clock, no registers.

Only for TensorFlow ops.
~100T Ops/cycle (limited precision)

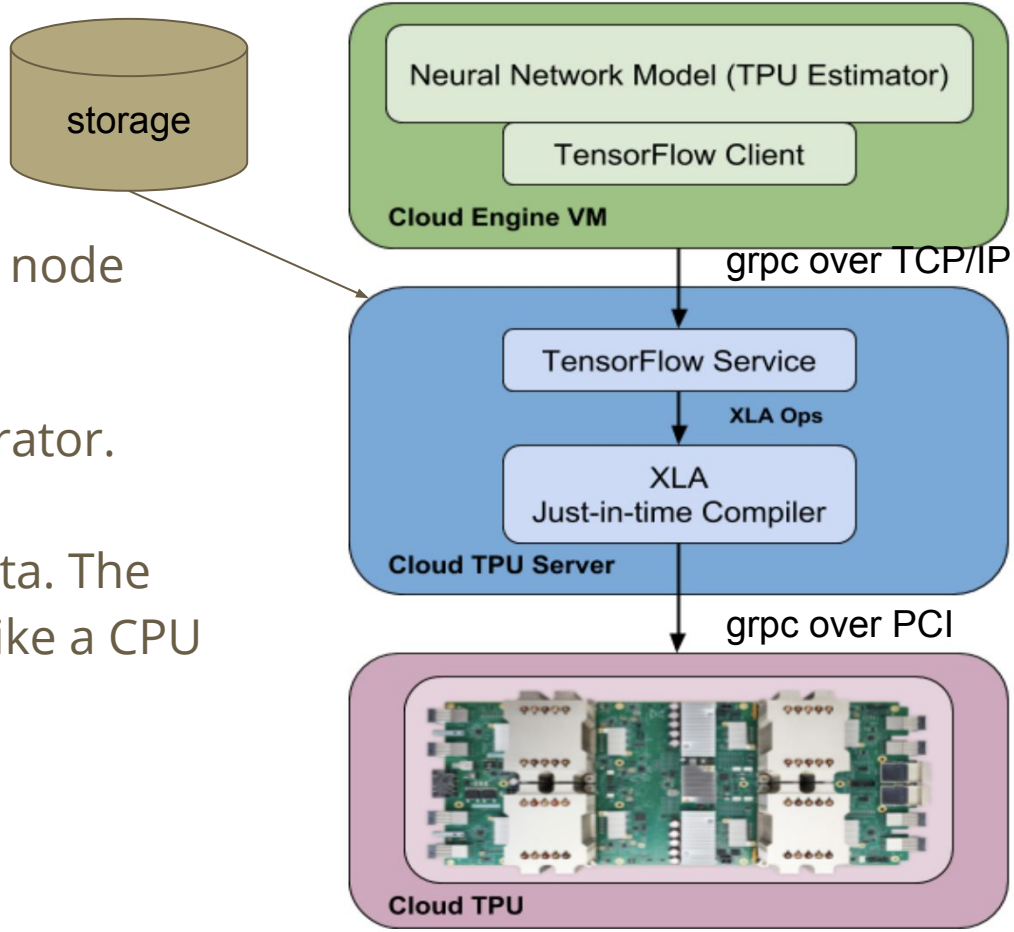
Matrix Multiplier Unit (MXU) of TPU

System Organization

Send TF graph as a whole to a TF node

Send individual TF operations
with their data to the TPU accelerator.

This includes instructions and data. The TPU does not fetch instructions like a CPU



TPU v3.0 specs

	TPU 3.0 card	TPU 3.0 node	TPU pod
#TPU's	1 card, 4 chips, 16 MXU	4 cards	1024 cards, 256 nodes
mem BW	5 TB/sec	20 TB/sec	5 PB/sec
flops / sec (*)	100 TF/sec	400 TF/sec	100 PF/sec

Operations per clock cycle

CPU	10's (cores)
CPU vectorized	1000 (core x vector length)
GPU	10K 's
TPU	128K (TPU v1)

* flops are of various precisions

This should raise eyebrows …

256 nodes for 5PB/sec of BW and 100PF ???

pretty much a top 5 machine in top500.org with 100x fewer systems (or 25x fewer allowing for 16 vs 64 bit precision).

It would work very well for moderate granularity computations, like SKA (and AI for which it was made). Wouldn't help with AMR likely, but surrogates may do that.

GPU's around 2003 evolved to GPGPU's through HPC.

Possible further enablers:

1. Are TensorFlow operations sufficient for HPC?
2. does a more general systolic network interconnect offer more opportunities?
3. Is mixed floating point precision required? (cf. posithub.org)

Conclusions
Thank you
Questions

- This is a fast moving emerging area
- Profound practical applications
- Likely deep source for theory
- Relationship between physical laws and ML may be easier to understand than “real world” ML