
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

CMP211-R

A framework for physics-informed neural networks on PyTorch

Srinivas Tadepalli

Head of GTM Strategy,
HPC Workloads
AWS

Ana Simoes

Principal Specialist,
AI/ML Frameworks
AWS

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- PINNs: State of the art and challenges
- Convergence of HPC & machine learning (ML)
- ML as a driver for scale
- Call to action

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

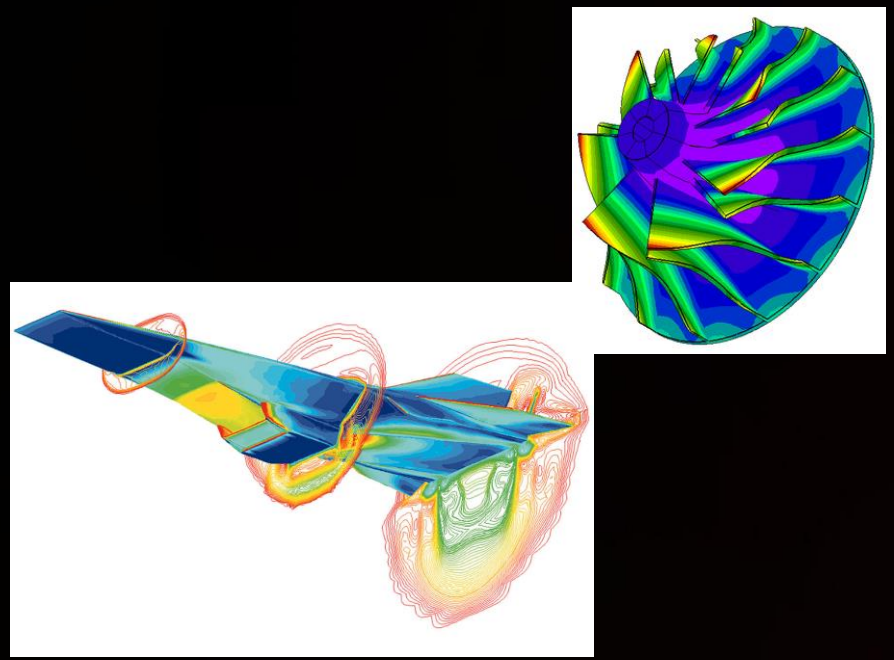
PINNs: State of the art and challenges

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

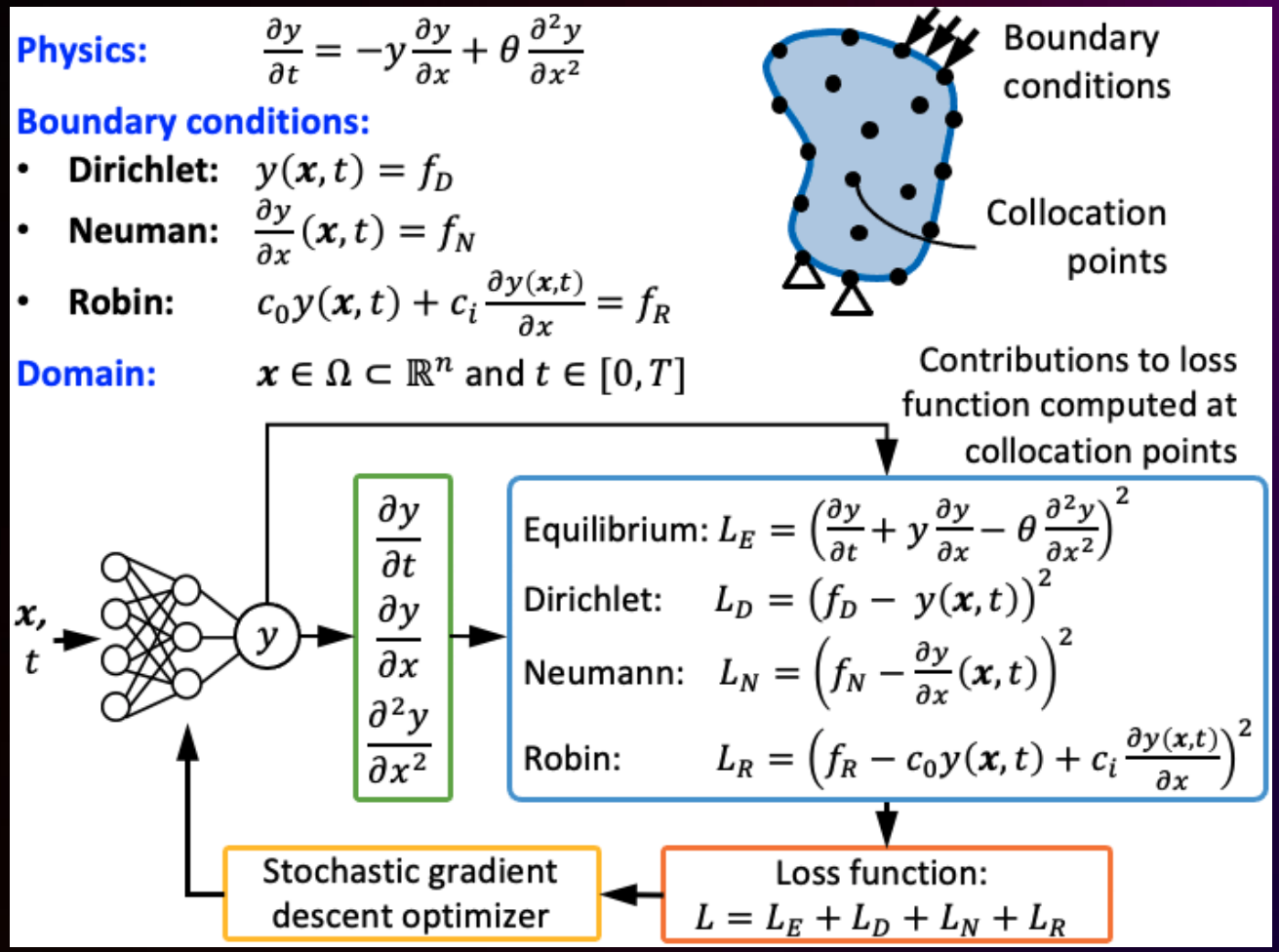
Physics-informed neural networks (PINNs) 101

Numerical methods used to solve partial differential equations (PDEs)

- Finite element modeling
- Computational fluid dynamics



Physics-informed neural networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

State of the art

JOURNAL OF COMPUTATIONAL PHYSICS 91, 110–131 (1990)

Neural Algorithm for Solving Differential Equations

HYUK LEE

Department of Electrical Engineering, Polytechnic Institute of New York, Brooklyn, New York 11201

AND

IN SEOK KANG

Department of Chemical Engineering, California Institute of Technology, Pasadena, California 91125

Received August 17, 1988; revised October 6, 1989

Finite difference equations are considered to solve differential equations numerically by utilizing minimization algorithms. Neural minimization algorithms for solving the finite difference equations are presented. Results of numerical simulation are described to demonstrate the method. Methods of implementing the algorithms are discussed. General features of the neural algorithms are discussed. © 1990 Academic Press, Inc.

IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 16, NO. 6, NOVEMBER 2005 1381

Finite-Element Neural Networks for Solving Differential Equations

Pradeep Ramuhalli, *Member, IEEE*, Lalita Udpa, *Senior Member, IEEE*, and Satish S. Udpa, *Fellow, IEEE*

***Abstract*—The solution of partial differential equations (PDE) arises in a wide variety of engineering problems. Solutions to most practical problems use numerical analysis techniques such as finite-element or finite-difference methods. The drawbacks of these approaches include computational costs associated with the modeling of complex geometries. This paper proposes a finite-element neural network (FENN) obtained by embedding a finite-element model in a neural network architecture that enables fast and accurate solution of the forward problem. Results of applying the FENN to several simple electromagnetic forward and inverse problems are presented. Initial results indicate that the FENN performance as a forward model is comparable to that of the conventional finite-element method (FEM). The FENN can also be used in an iterative approach to solve inverse problems associated with the PDE. Results showing the ability of the FENN to solve the inverse problem given the measured signal are also presented. The parallel nature of the FENN also makes it an attractive solution for parallel implementation in hardware and software.**

***Index Terms*—Finite-element method (FEM), finite-element neural network (FENN), inverse problems.**

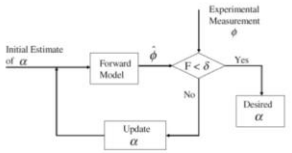


Fig. 1. Iterative inversion method for solving inverse problems.

resulting in the corresponding solution to the forward problem ($\hat{\phi}$). The model output is compared to the measurement (ϕ), using a cost function $F(\phi, \hat{\phi})$. If $F(\phi, \hat{\phi})$ is less than a tolerance δ, the estimate α is used as the desired solution. If not, α is updated to minimize the cost function.

SCIENCE ADVANCES | RESEARCH ARTICLE

APPLIED MATHEMATICS

Data-driven discovery of partial differential equations

Samuel H. Rudy,[1]* Steven L. Brunton,[2] Joshua L. Proctor,[3] J. Nathan Kutz[1]

We propose a sparse regression method capable of discovering the governing partial differential equation(s) of a given system by time series measurements in the spatial domain. The regression framework relies on sparsity-promoting techniques to select the nonlinear and partial derivative terms of the governing equations that most accurately represent the data, bypassing a combinatorially large search through all possible candidate models. The method balances model complexity and regression accuracy by selecting a parsimonious model via Pareto analysis. Time series measurements can be made in an Eulerian framework, where the sensors are fixed spatially, or in a Lagrangian framework, where the sensors move with the dynamics. The method is computationally efficient, robust, and demonstrated to work on a variety of canonical problems spanning a number of scientific domains including Navier-Stokes, the quantum harmonic oscillator, and the diffusion equation. Moreover, the method is capable of disambiguating between potentially nonunique dynamical terms by using multiple time series taken with different initial data. Thus, for a traveling wave, the method can distinguish between a linear wave equation and the Korteweg–de Vries equation, for instance. The method provides a promising new technique for discovering governing equations and physical laws in parameterized spatiotemporal systems, where first-principles derivations are intractable.

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

M. Raissi [a], P. Perdikaris [b,*], G.E. Karniadakis [a]

[a] Division of Applied Mathematics, Brown University, Providence, RI, 02912, USA
[b] Department of Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA, 19104, USA

ARTICLE INFO

Article history:
Received 13 June 2018
Received in revised form 26 October 2018
Accepted 28 October 2018
Available online 3 November 2018

Keywords:
Data-driven scientific computing
Machine learning
Predictive modeling
Runge–Kutta methods
Nonlinear dynamics

ABSTRACT

We introduce *physics-informed neural networks* – neural networks that are trained to solve supervised learning tasks while respecting any given laws of physics described by general nonlinear partial differential equations. In this work, we present our developments in the context of solving two main classes of problems: data-driven solution and data-driven discovery of partial differential equations. Depending on the nature and arrangement of the available data, we devise two distinct types of algorithms, namely continuous time and discrete time models. The first type of models forms a new family of *data-efficient* spatio-temporal function approximators, while the latter type allows the use of arbitrarily accurate implicit Runge–Kutta time stepping schemes with unlimited number of stages. The effectiveness of the proposed framework is demonstrated through a collection of classical problems in fluids, quantum mechanics, reaction-diffusion systems, and the propagation of nonlinear shallow-water waves.

REVIEWS

Check for updates

Physics-informed machine learning

George Em Karniadakis[1,2], Ioannis G. Kevrekidis[3,4], Lu Lu[5], Paris Perdikaris[6], Sifan Wang[7] and Liu Yang[1]

Abstract | Despite great progress in simulating multiphysics problems using the numerical discretization of partial differential equations (PDEs), one still cannot seamlessly incorporate noisy data into existing algorithms, mesh generation remains complex, and high-dimensional problems governed by parameterized PDEs cannot be tackled. Moreover, solving inverse problems with hidden physics is often prohibitively expensive and requires different formulations and elaborate computer codes. Machine learning has emerged as a promising alternative, but training deep neural networks requires big data, not always available for scientific problems. Instead, such networks can be trained from additional information obtained by enforcing the physical laws (for example, at random points in the continuous space-time domain). Such physics-informed learning integrates (noisy) data and mathematical models, and implements them through neural networks or other kernel-based regression networks. Moreover, it may be possible to design specialized network architectures that automatically satisfy some of the physical invariants for better accuracy, faster training and improved generalization. Here, we review some of the prevailing trends in embedding physics into machine learning, present some of the current capabilities and limitations and discuss diverse applications of physics-informed learning both for forward and inverse problems, including discovering hidden physics and tackling high-dimensional problems.

Archives of Computational Methods in Engineering
https://doi.org/10.1007/s11831-021-09539-0

SURVEY ARTICLE

A Survey of Bayesian Calibration and Physics-informed Neural Networks in Scientific Modeling

Felipe A. C. Viana[1] · Arun K. Subramaniyan[2]

Received: 28 September 2020 / Accepted: 7 January 2021
© CIMNE, Barcelona, Spain 2021

Abstract

Computer simulations are used to model of complex physical systems. Often, these models represent the solutions (or at least approximations) to partial differential equations that are obtained through costly numerical integration. This paper presents a survey of two important statistical/machine learning approaches that have shaped the field of scientific modeling. Firstly we survey the developments on Bayesian calibration of computer models since the seminal work by Kennedy and O'Hagan. In their paper, the authors proposed an elegant way to use the Gaussian processes to extend calibration beyond parameter and observation uncertainty and include model-form and data size uncertainty. Secondly, we also survey physics-informed neural networks, a topic that has been receiving growing attention due to the potential reduction in computational cost and modeling flexibility. In addition, in order to help the interested reader to familiarize with these topics and venture into custom implementations, we present a summary of applications and software tools. Finally, we close the paper with suggestion for future research directions and a thought provoking call for action.

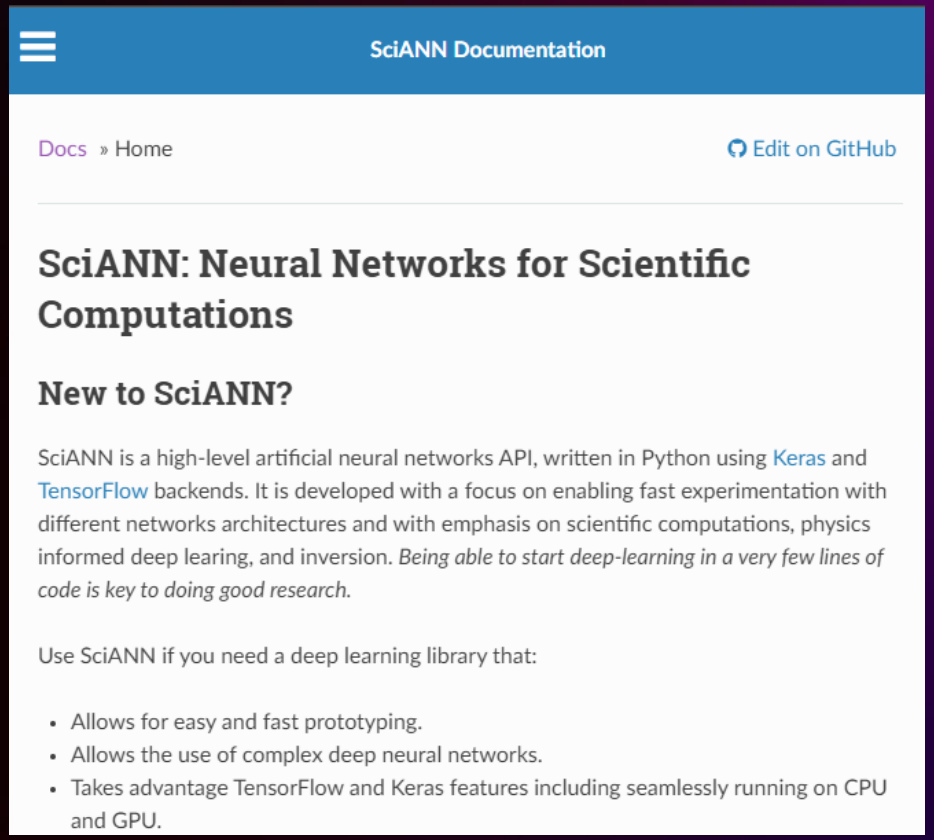
How about software?

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Academic and industry-grade software

Non-comprehensive list of libraries/packages

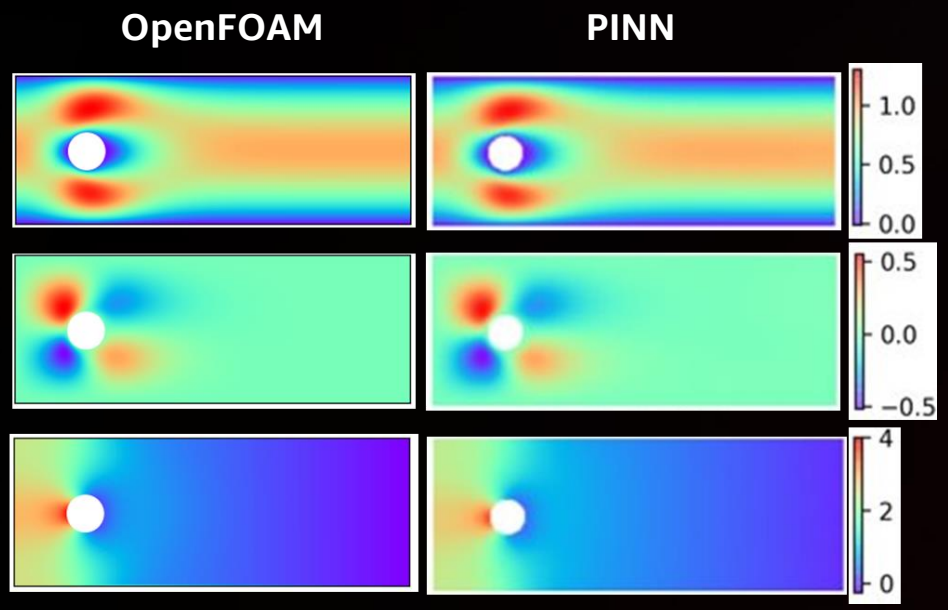
Package	Language	Link
SciANN	Python	https://www.sciann.com
Modulus	Python	https://developer.nvidia.com/modulus
DeepXDE	Python	https://deepxde.readthedocs.io
PyDEns	Python	https://github.com/analysiscenter/pydens
NeuroDiffEq	Python	https://neurodiffeq.readthedocs.io/
NeuralPDE	Julia	https://github.com/SciML/NeuralPDE.jl
ADCME	Julia	https://github.com/kailaix/ADCME.jl



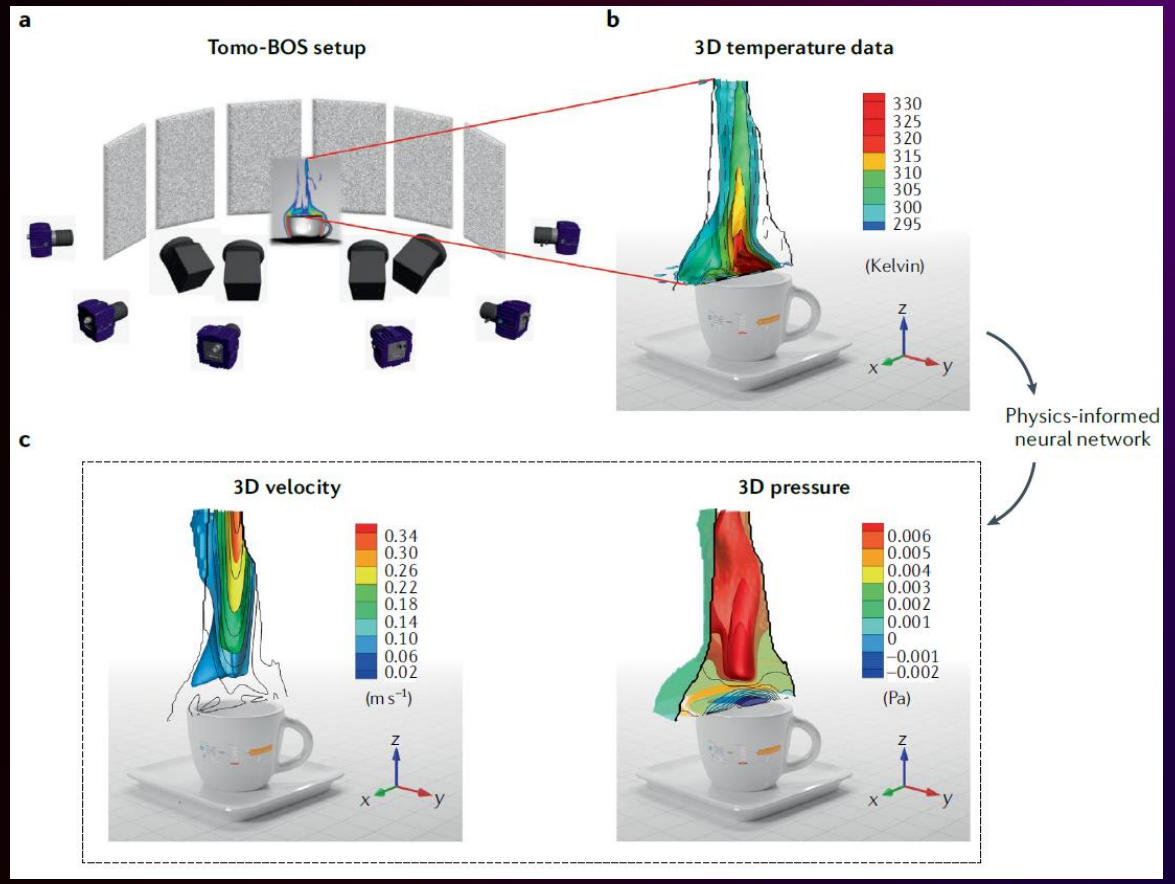
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Teaser applications

A. Sample of our results



B. Sample from literature



Source: Karniadakis, Kevrekidis, Lu, Perdikaris, Wang & Yang, "Physics-informed machine learning," 422-440, https://www.nature.com/articles/s42254-021-00314-5.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Current challenges

Complex phenomena: multiphysics and multiscale, and problems

- Chemical reactions, integrated computational materials engineering, global climate modeling
- Fluid-structure interaction, thermo-structural coupling, structural-acoustic interaction

Implementation: algorithms and computational frameworks

- Non-convex optimization inherently found in PINNs
- Multiple contributions to the loss function
- Neural network architectures
- Frameworks supporting higher-order derivatives found in PDEs/PINNs

Benchmarking: what is the ImageNet-equivalent for PINNs?

- Reproducibility of results
- SOTA: Computational cost and accuracy

Fundamental research

- Theoretical foundation versus current empirical-based approaches
- Error estimates: Approximation, discretization, optimization

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Convergence of HPC & ML

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Every HPC and ML workload can be run on AWS

AstraZeneca

Life sciences

storengy
Une société de ENGIE

Energy

ampersand

Large-scale ML

Design and engineering

DTN

Climate and earth sciences

Aurora

Autonomous vehicles

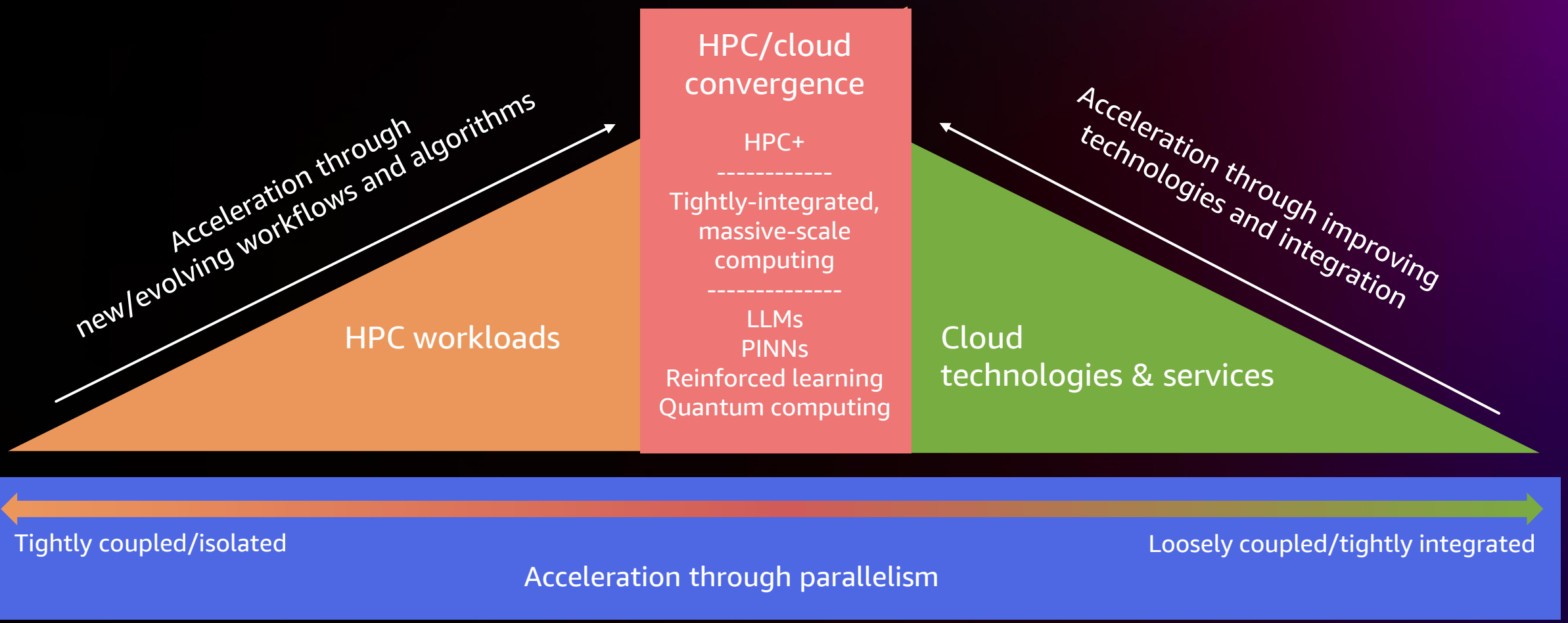
AI21 labs

Large language models

aws

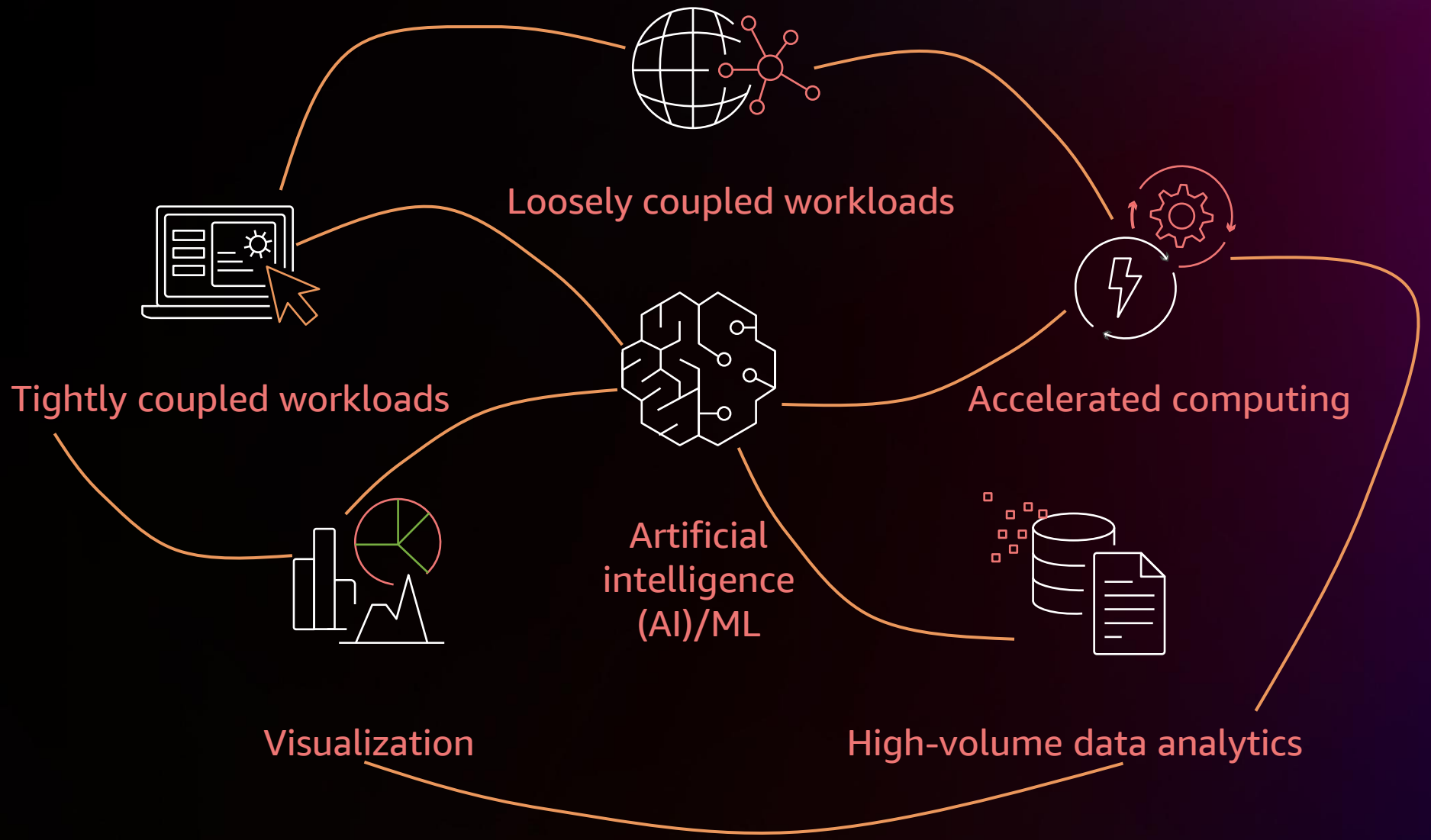
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The convergence of cloud, HPC, and ML



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

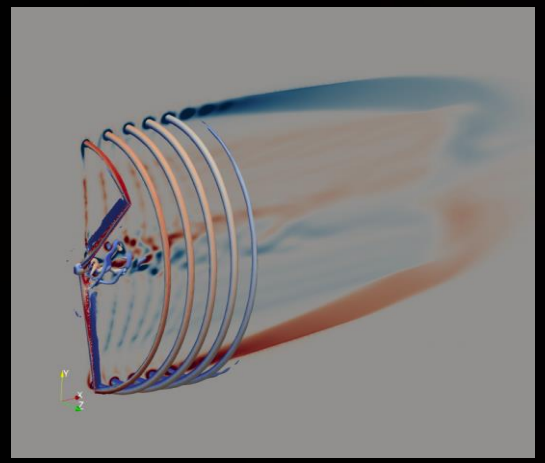
Varied compute and throughput characteristics



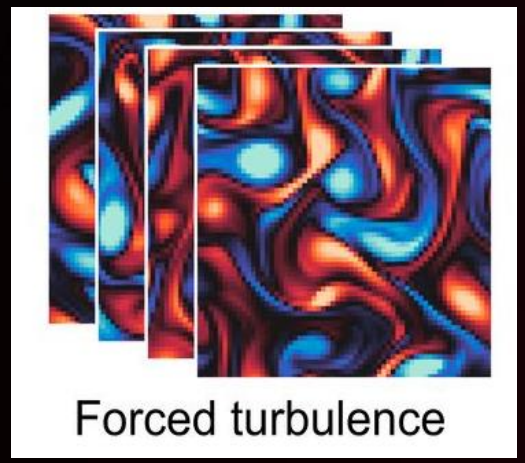
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

From HPC simulations to predictive modeling

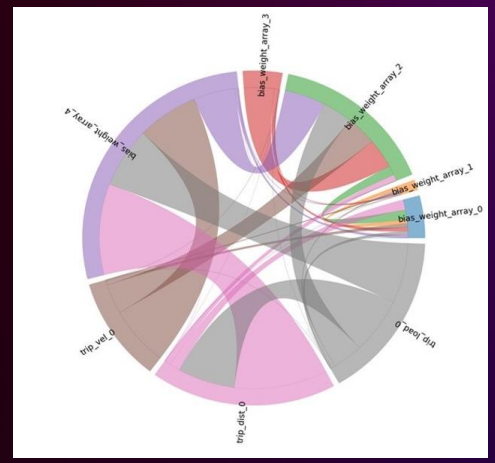
Computational fluid dynamics (CFD)



Accelerate CFD with AI/ML



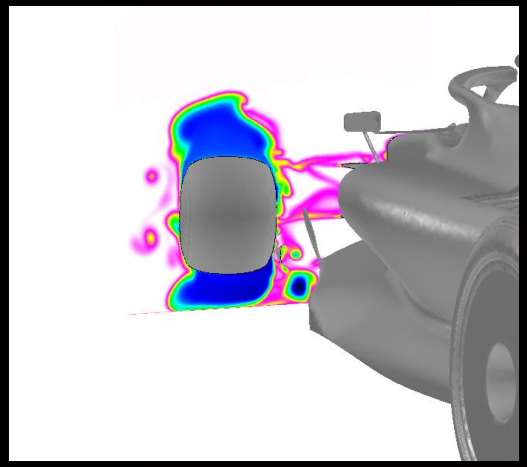
Predictive modeling for digital twins



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

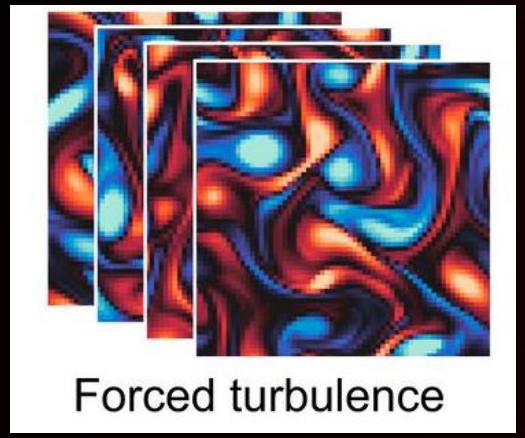
Convergence of HPC + ML

Extract relationships from simulation data



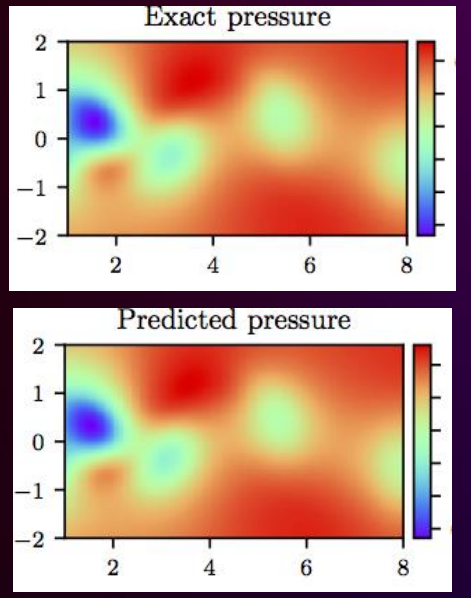
Credit: F1

Accelerate HPC simulations with embedded ML models



Credit: “Machine learning–accelerated computational fluid dynamics,” https://www.pnas.org/content/118/21/e2101784118

Learn solutions to governing equations directly

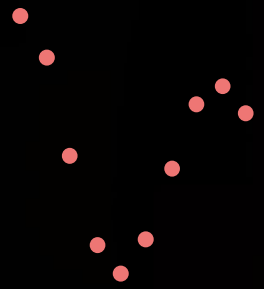


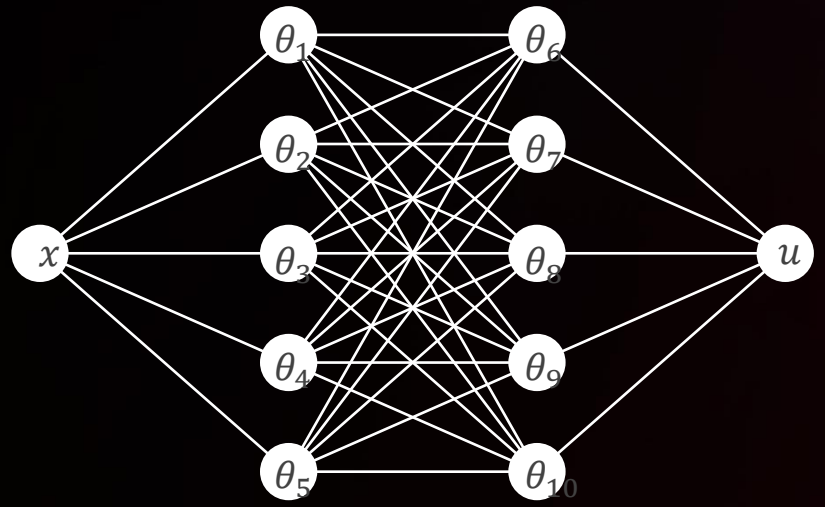
Credit: “Physics Informed Deep Learning,” https://maziarraissi.github.io/PINNs/

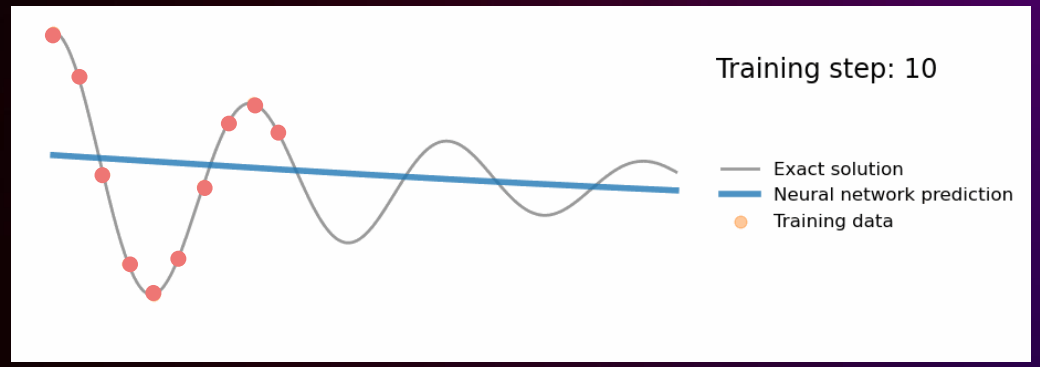
AI-driven infrastructure optimization

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Neural network







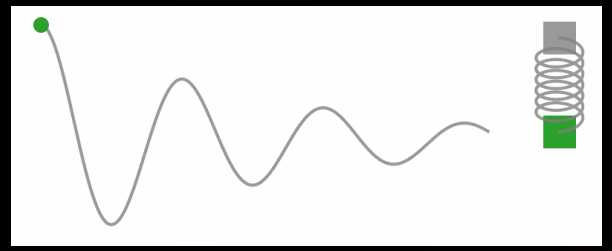
experimental data

$$\min \frac{1}{n} \sum_{i}^{n} \left(u_{nn}(x_i, \theta) - u_{true(x_i)} \right)^2$$

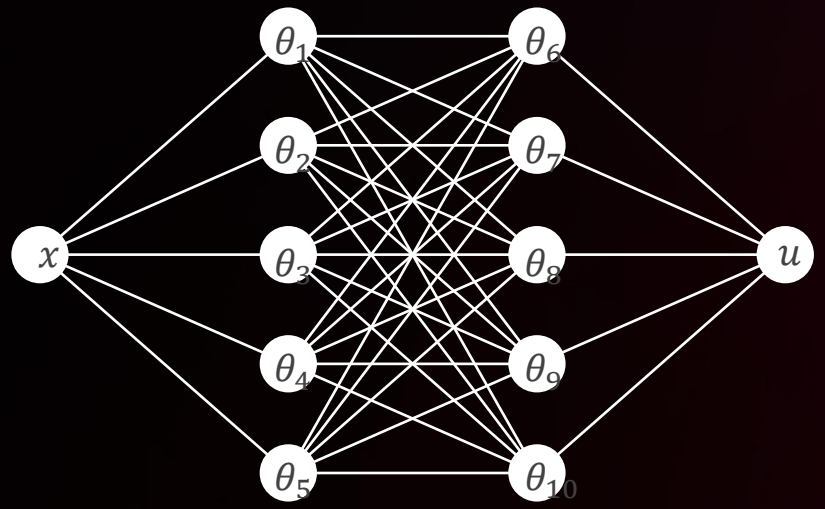
aws

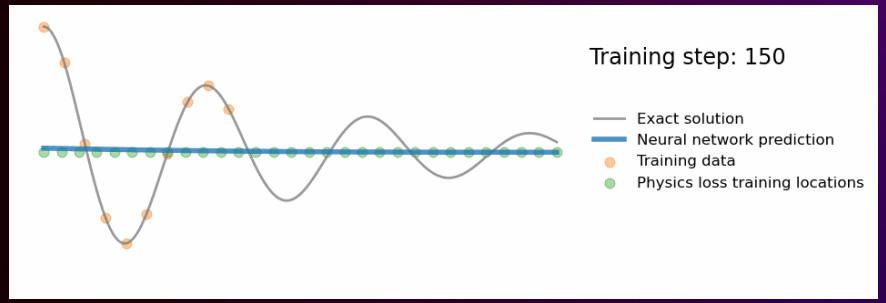
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Physics Informed Neural Network (PINN)



$$m\frac{\mathrm{d}^2u}{\mathrm{d}x^2} + \mu\frac{du}{dx} + ku$$





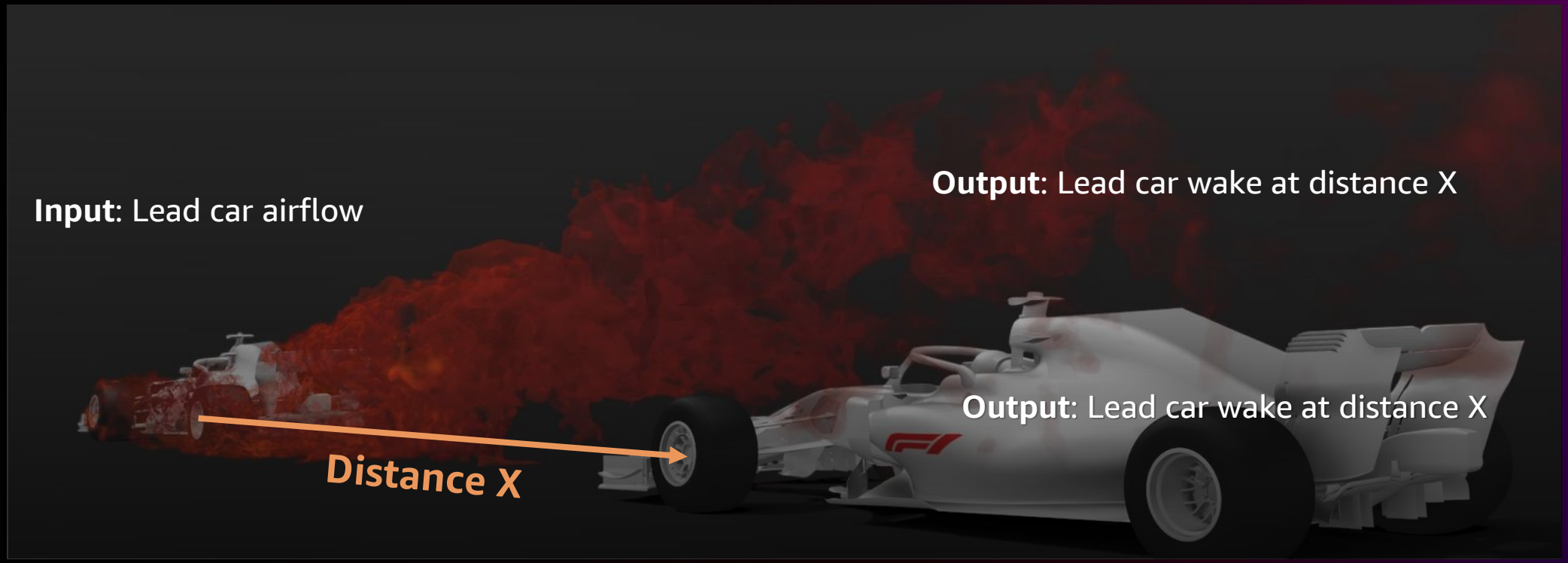
$$\min \frac{1}{n}\sum_{M}^{i}\left(u_{nn}(x_i,\theta) - u_{true(x_i)}\right)^2$$

$$+\frac{1}{M}\sum_{j}([m\frac{d^2}{dx^2} + \mu\frac{d}{\mathrm{d}x} + k]u_{nn}(x_j,\theta))^2$$

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Demonstrating ML + HPC

Extract relationships from HPC simulation results



Problem statement:

Determine optimal lead car features to obtain target flow profile at distance X

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

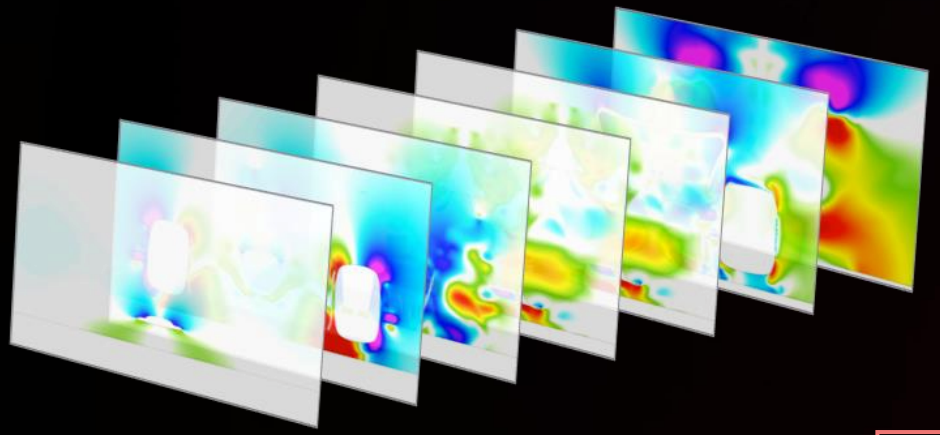
ML + HPC solution

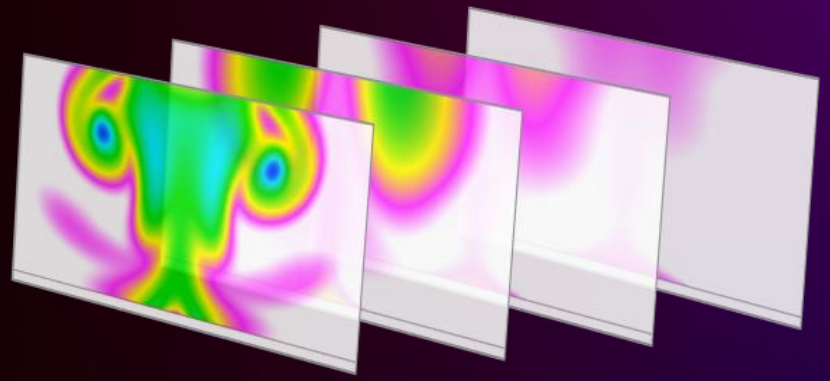
3D geometry

Input flow field

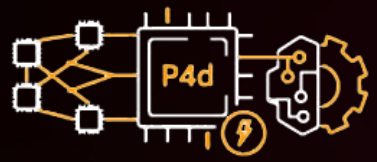
DL model

Output flow field





EC2 UltraClusters

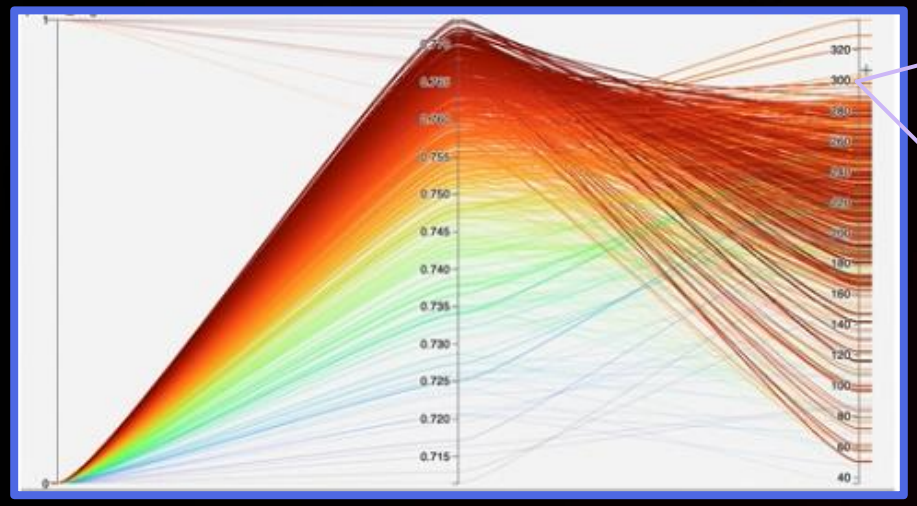


aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

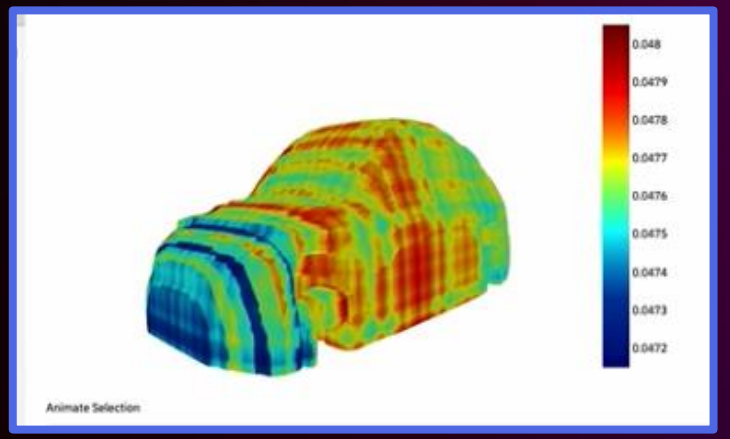
Global optimization with physics-enabled models

Objective 1 Objective 2

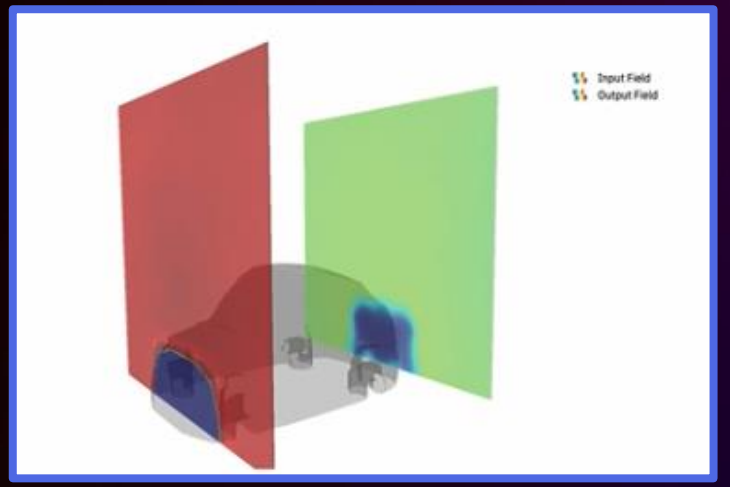


10,000 designs explored

Optimal geometry



Optimal flow field



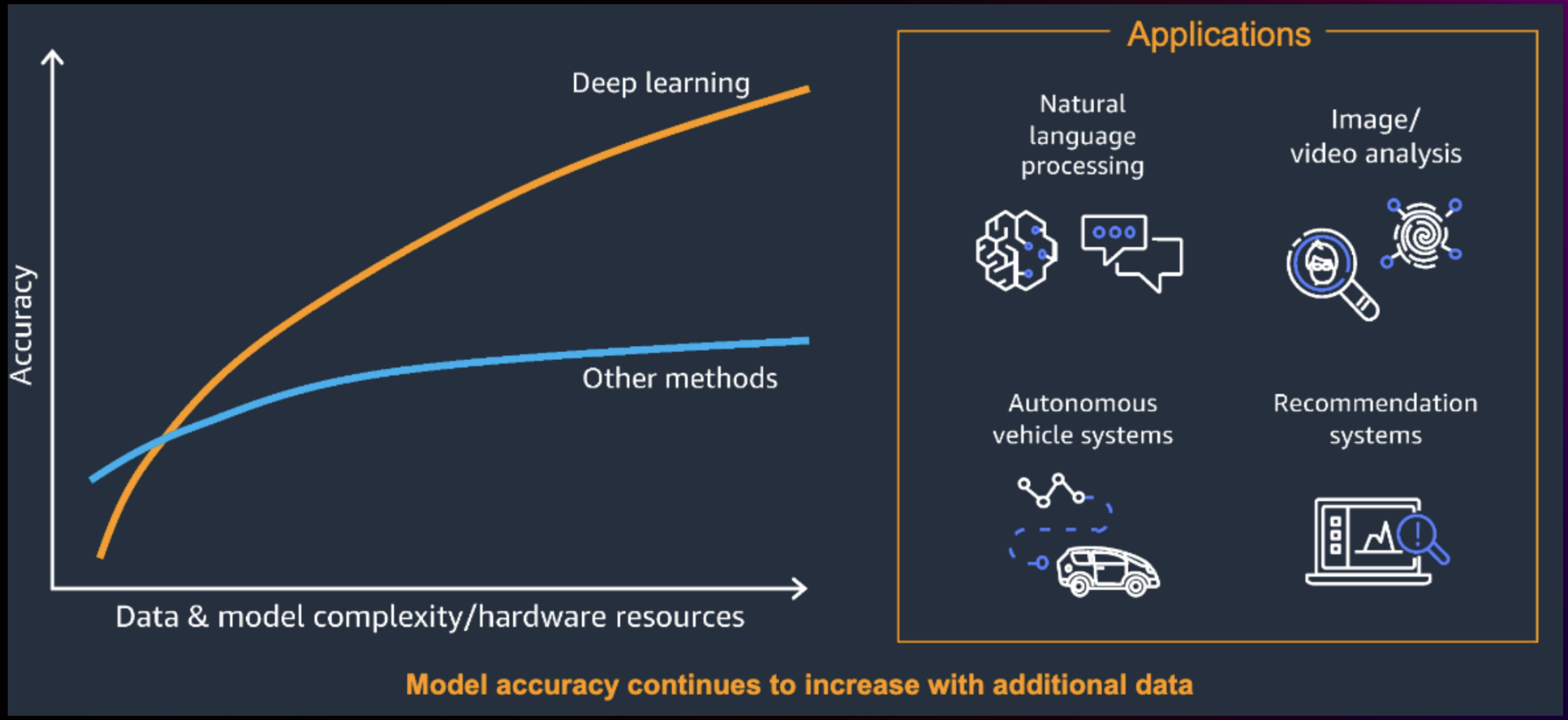
Optimal geometry and flow field simultaneously predicted

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

ML as a driver for scale

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Drivers behind scale in deep learning



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The AWS ML stack: Broadest and most complete set of ML capabilities

AI Services

Vision
Amazon Rekognition

Speech
Amazon Polly
Amazon Transcribe

Chatbots
Amazon Lex

Text
Amazon Comprehend
Amazon Translate
Amazon Textract

Business tools
Amazon Personalize
Amazon Forecast
Amazon Fraud Detector
Amazon Lookout for Metrics

Contact centers
Contact Lens
Amazon Connect Voice ID

Search
Amazon Kendra

Code + DevOps
Amazon CodeGuru
Amazon DevOps Guru

Healthcare
Amazon HealthLake
Amazon Comprehend Medical
Amazon Transcribe Medical

Industrial
AWS Panorama Appliance and SDK, Amazon Monitron, Amazon Lookout for Equipment, Amazon Lookout for Vision

Amazon SageMaker

Label data

SageMaker Studio IDE

Data collection prep | Store features | Detect bias, explain predictions | Visualize in notebooks | Pick algorithm | Train models faster | Tune parameters | Deploy in production | Manage, monitor

CI/CD

Manage edge devices

Self-managed ML

PYTORCH
mxnet TensorFlow

Deep learning AMIs, containers

GPUs

Habana Gaudi accelerators

Inferentia

Trainium

FPGA

TensorFlow, the TensorFlow logo and any related marks are trademarks of Google Inc.; PyTorch, the PyTorch logo and any related marks are trademarks of Facebook, Inc.

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

ML frameworks & infrastructure layer

Tools

Data processing & labeling


Amazon Kinesis


AWS Glue


Amazon SageMaker ground truth

Prebuilt images


DLAMI


DL containers

Dev, training & tuning



Notebooks



Horovod

Kubeflow operators

Inference servers

TorchServe

NVIDIA Triton

TF Serving

MLOps

Kubeflow

Orchestration

AWS services


Amazon EKS


Amazon ECS


AWS Fargate


AWS Batch


AWS ParallelCluster

Workflow

Open source software

Kubernetes

Kubeflow

Airflow

Ray

Other services

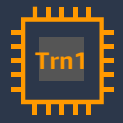

AWS Lambda


AWS Step Functions


AWS CloudFormation

Core infrastructure

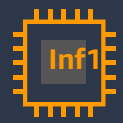
Compute













Storage


Amazon S3


Amazon EFS



Amazon FSx

Networking


EFA

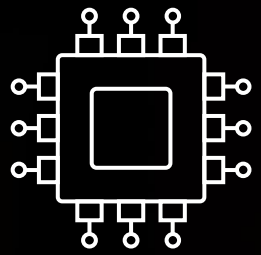
Data transfer


AWS Direct Connect

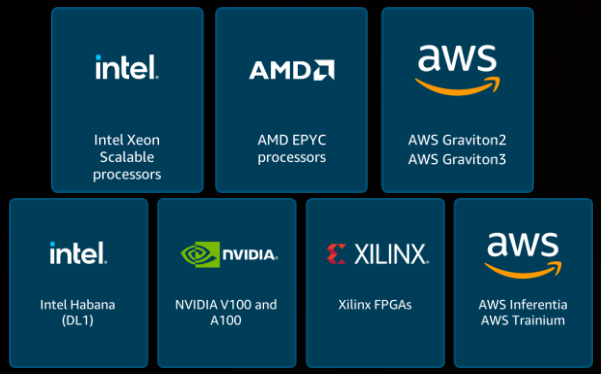
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key services that enable HPC and self-managed ML on AWS

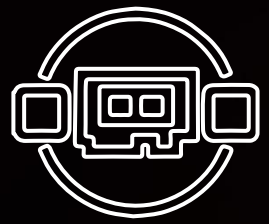
Compute



Amazon EC2



Networking



EFA

Up to 400 Gbps networking

Storage

FSx

FSx for Lustre

AFSx for NetApp ONTAP

FSx for OpenZFS

Orchestration

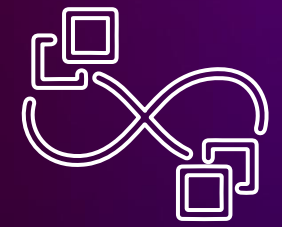


AWS Batch

AWS ParallelCluster

EnginFrame

Visualization

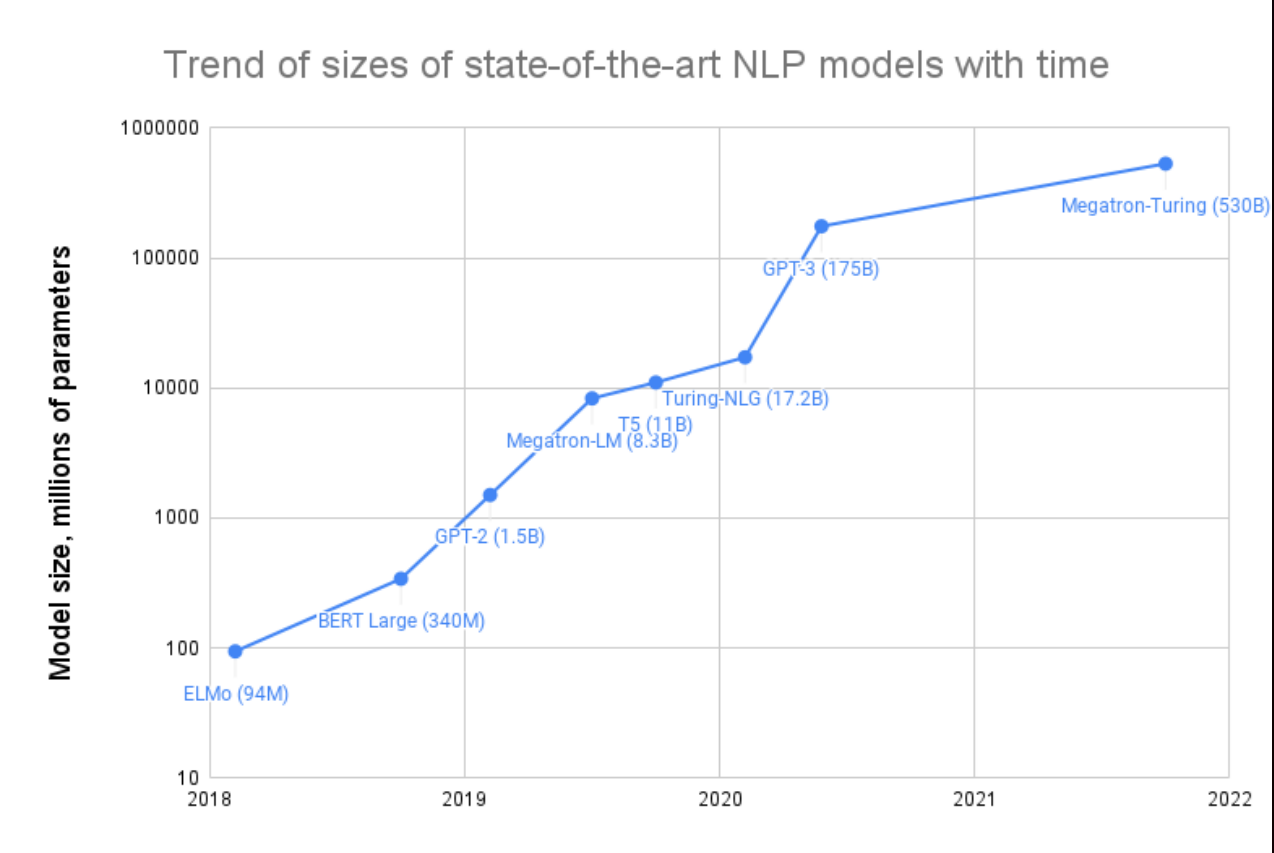


NICE DCV

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training a large NLP model: HPC class

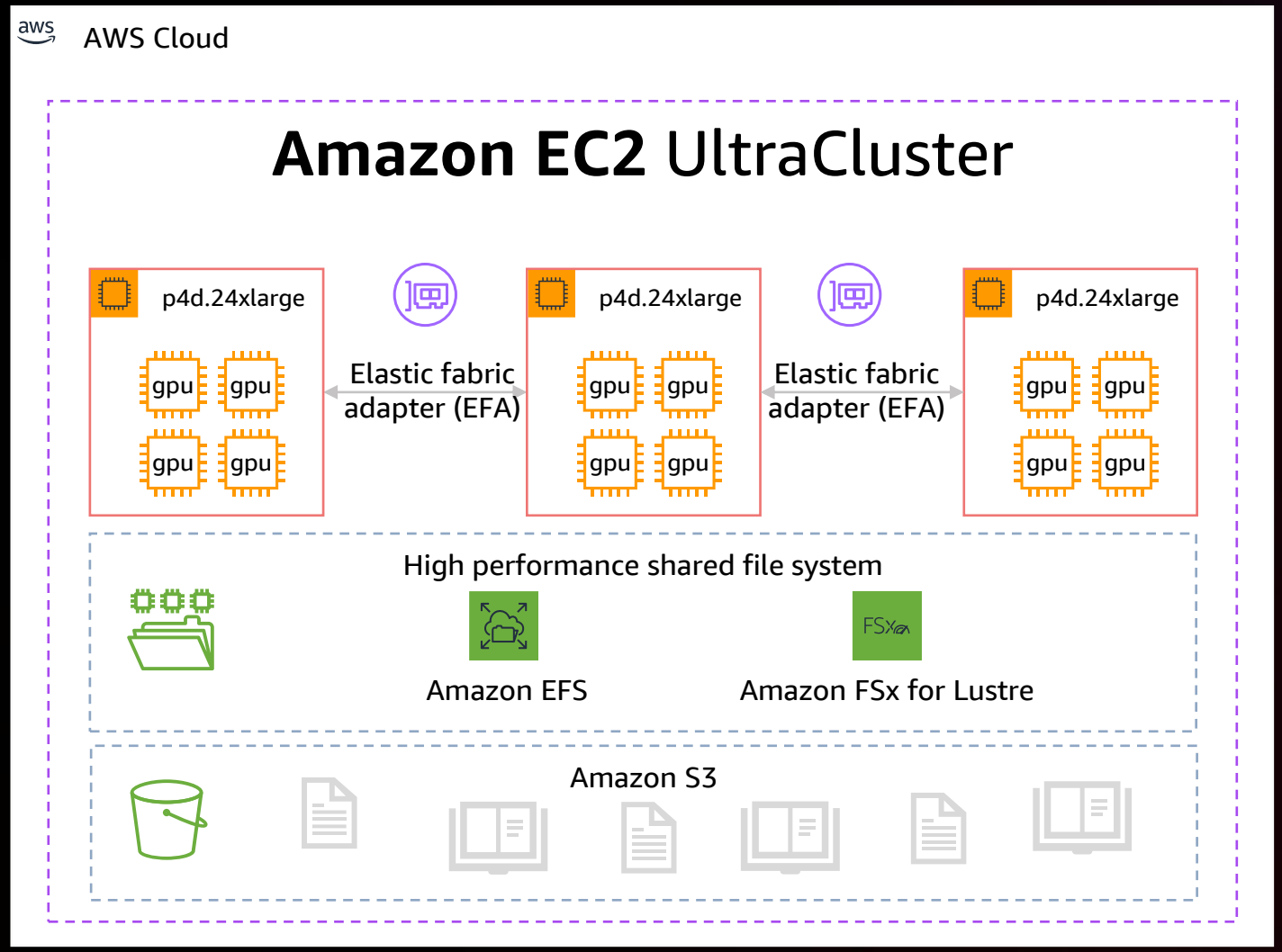


Key challenges

- Limited compute and memory
- Increased inter-, intra-node communication
- Scalable distributed training requires software tools to be
  - Generic, not particular to a neural network architecture
  - User friendly
  - Efficient and scalable

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed ML on p4d UltraClusters



Customers pushing the limits

- **Anthropic** (XXL): 2,000 P4ds (16,000 GPUs) and Amazon EKS to train 2T+ param model
- **AI21** (XL): 104 P4ds (800+ GPUs) trained a 178 B param model
- **Meta** (XXL): 10,000 GPUs with ParallelCluster and Slurm for NLU research

All using PyTorch to scale

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

GPT-3 experiments: HPC resources

Compute nodes: 64× p4d.24xlarge

Network: 400 Gbps ENA & EFA

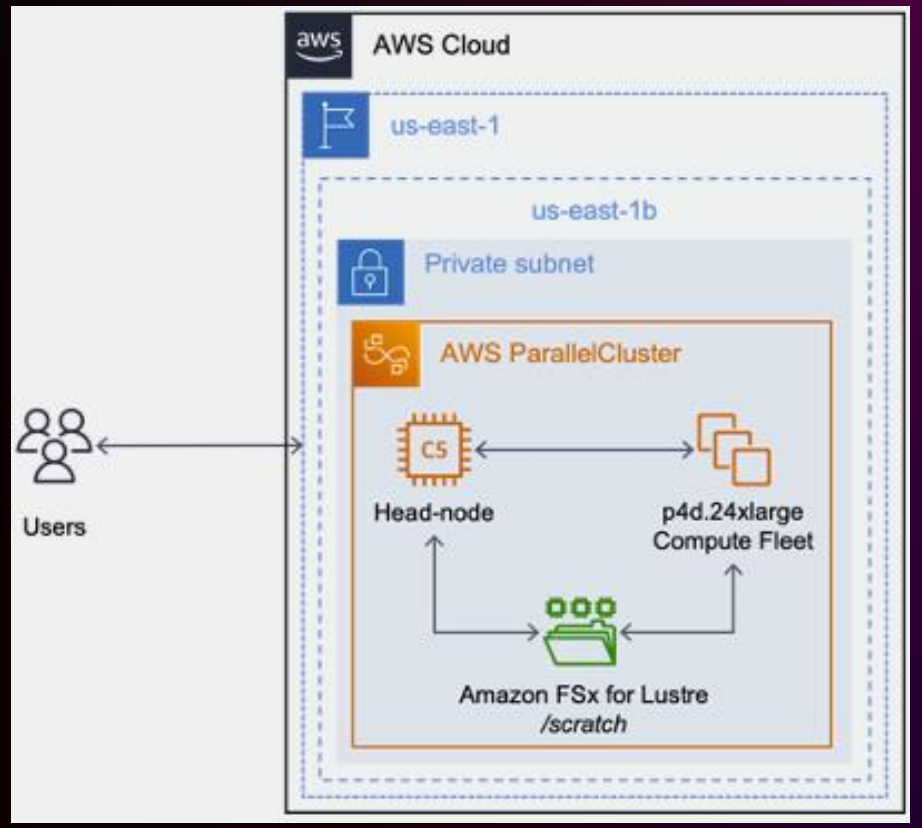
Storage: 8 × 1 TB NVMe + Amazon EBS

Shared file systems

- NFS/apps: Amazon EBS GP3 – 200 GB
- FSx for Lustre /scratch: 4.8 TB

Cluster stack

- Slurm 17.X
- CUDA 11.4, NVIDIA driver 470
- PyTorch 1.10



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!



Please complete the session survey in the **mobile app**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.