Hewlett Packard Enterprise

MODULE 4:

APPLIED AUGMENTED AI AND HPC AI

AGENDA

1. APPLIED AI

2. AUGMENTED AI

3. CONVERGE AI IN HPC SIMULATIONS

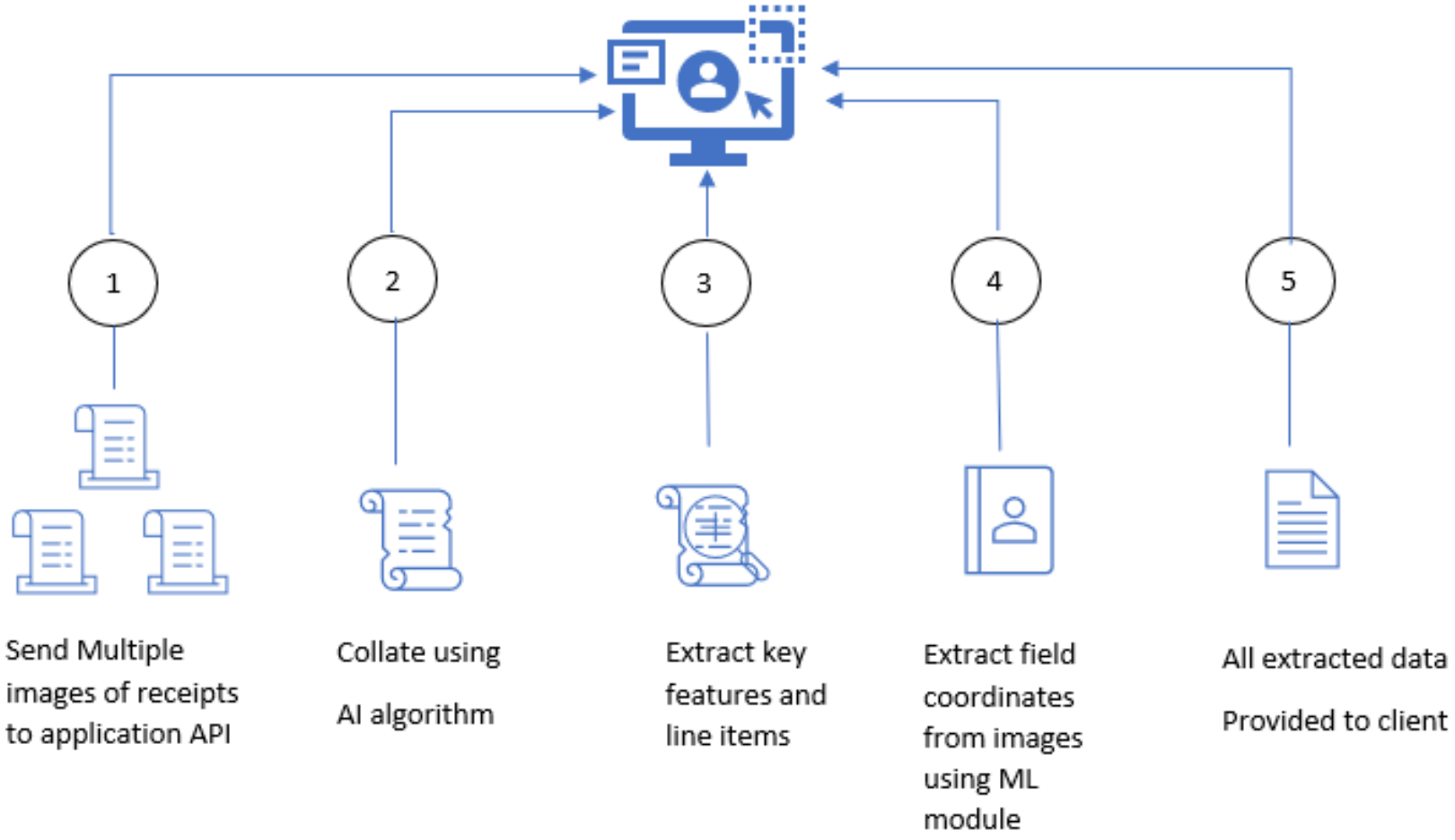
LESSONS

- Lesson 1: Applied AI
  - Form Recognition
  - Anomaly detection
    - –Types of Anomaly Detection
    - –Isolation Forest
    - –Applications
- Lesson 2: Augmented Artificial Intelligence (AAI)
  - –AAI – An insight
  - –Functional aspects of AAI
  - –AAI and Big data
  - –Differences between AI and AAI
- Lesson 3: SmartSim
  - Intro
  - Architecture
  - Use case
  - Weather Climate example

LESSON 2:

Applied AI

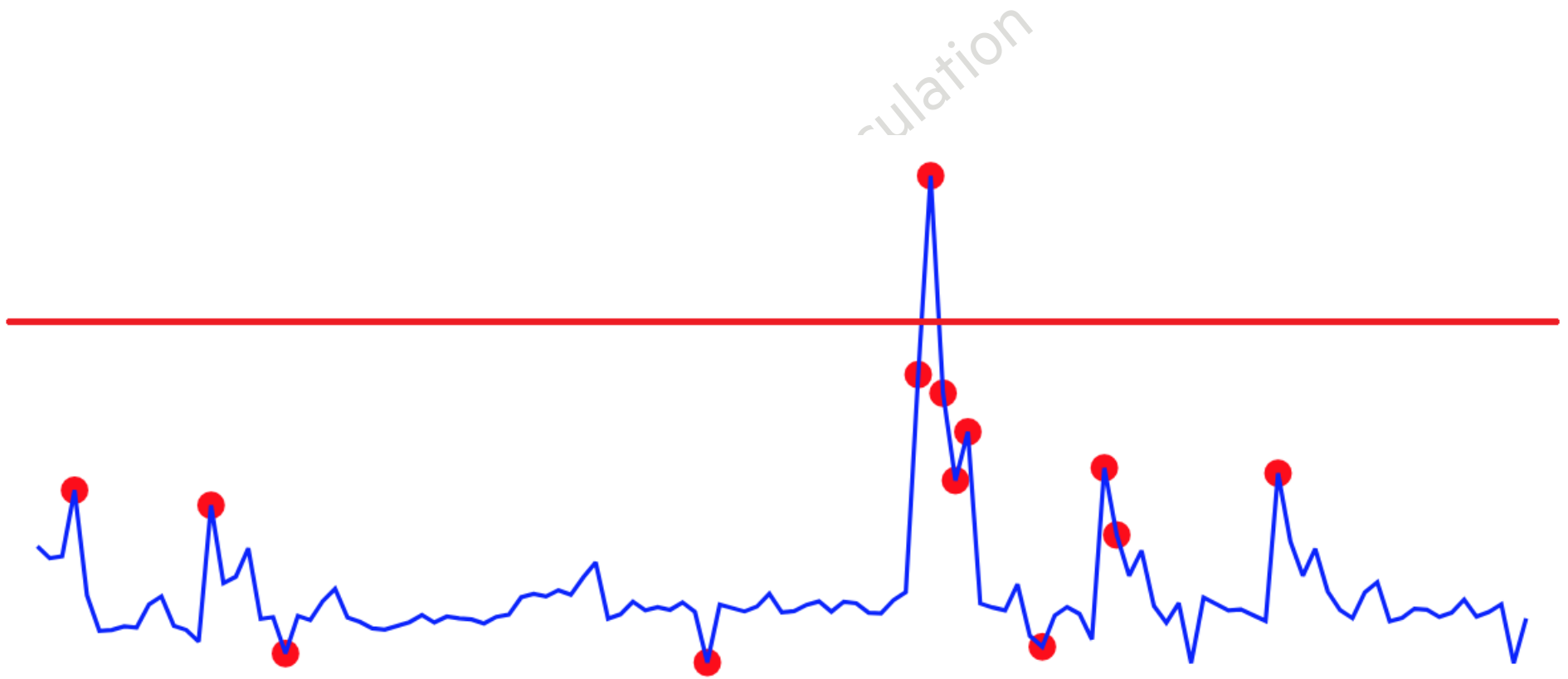
FORM RECOGNITION



FORM RECOGNITION

- Form Recognition helps to provide a well structure database that can be searched easily for information.
- In form recognizer key-values are extracted from form.

ANOMALY DETECTION



ANOMALY DETECTION

- The set of data points that are considerably different than the remainder of the data
- Abnormal instances are known as outliers(anomalies)
- Normal instances are known as inliers.
- The reasons for anomalies can be data errors, noise, and hidden patterns in the dataset.
- Can be important or a nuisance
  - Very high blood pressure
  - 200 pound for 2 year old

ANOMALY DETECTION

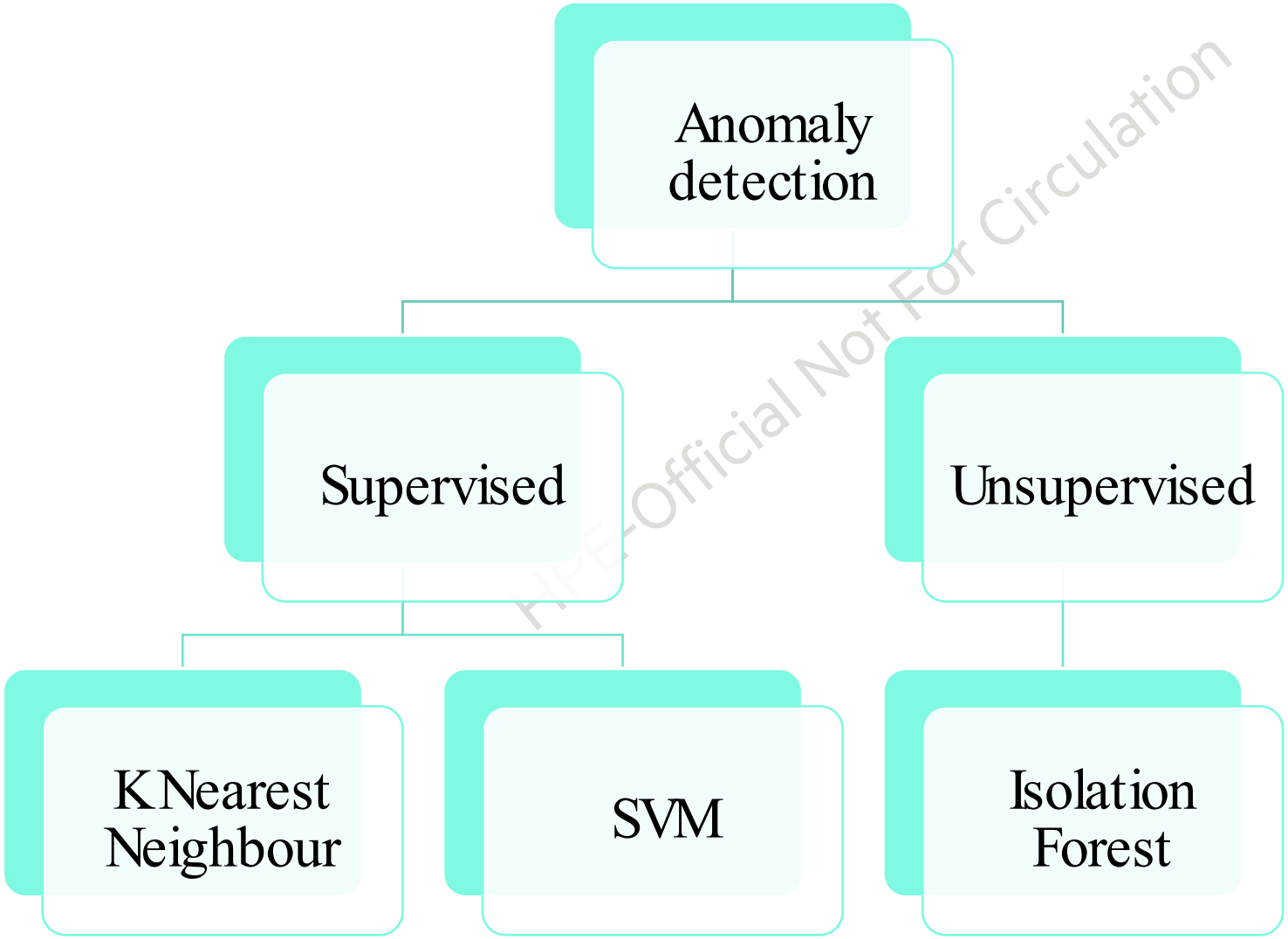
- Three possible cases:

❑Correct Detection: Abnormalities is detected correctly.

❑False Positives: Unexpected data values are observed, but the process continues to be normal.

❑False Negatives: Consequences don't get registered in the abnormal data, but the process becomes abnormal.

TYPES OF ANOMALY DETECTION



Two directions to search anomalies:

Outlier detection: If an observation differs from other data points.

Novelty detection: If the training data doesn't have any outlier and detecting anomaly in the new observation.

ISOLATION FOREST

- Random observations are isolated by selecting a feature and a split value is selected between maximum and minimum values of the selected feature.
- This method takes advantage of two properties of anomalies,

➢ They are minorities consisting of fewer instances

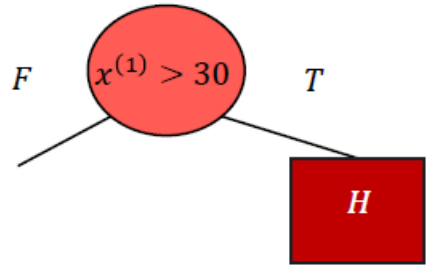
➢ Compared to normal instances, their attribute values are very different.

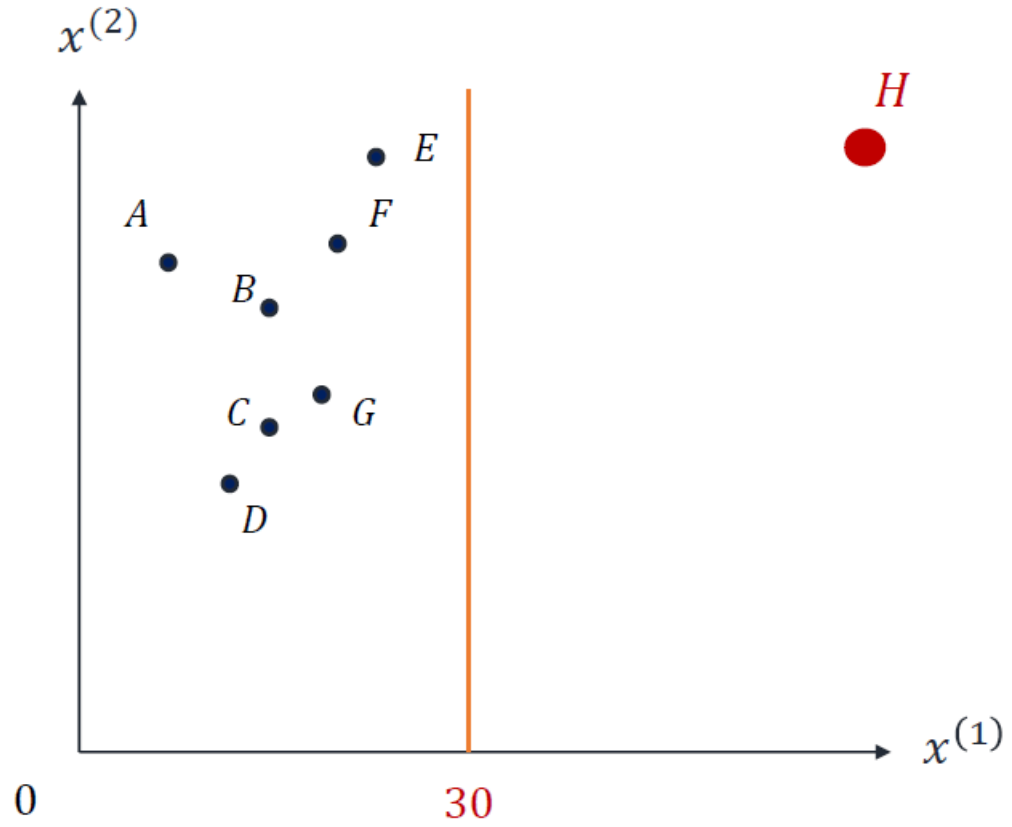
- Every single instance of anomaly can be isolated with a tree structure.

ISOLATION FOREST

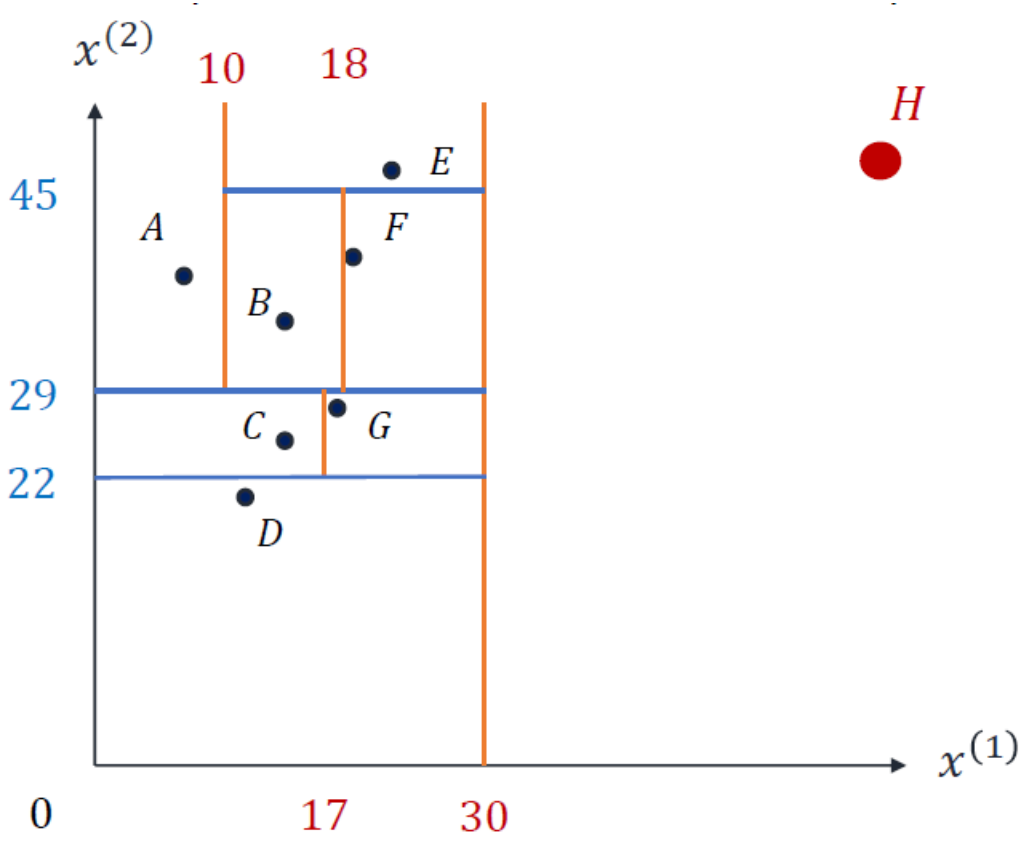
- Anomalies fall closer to the tree root and normal points will be at the deeper end of the tree.
- Detection of anomalies is based on these isolation characteristics of a tree known as the Isolation Tree or iTree.
- **IsolationTree(iTree)**:binary tree where each node in the tree has exactly zero or two daughter nodes
- An anomaly will be far away from normal instances, and it becomes isolated from other instances.
- On the iTree, instances that have short average path lengths are anomalies.

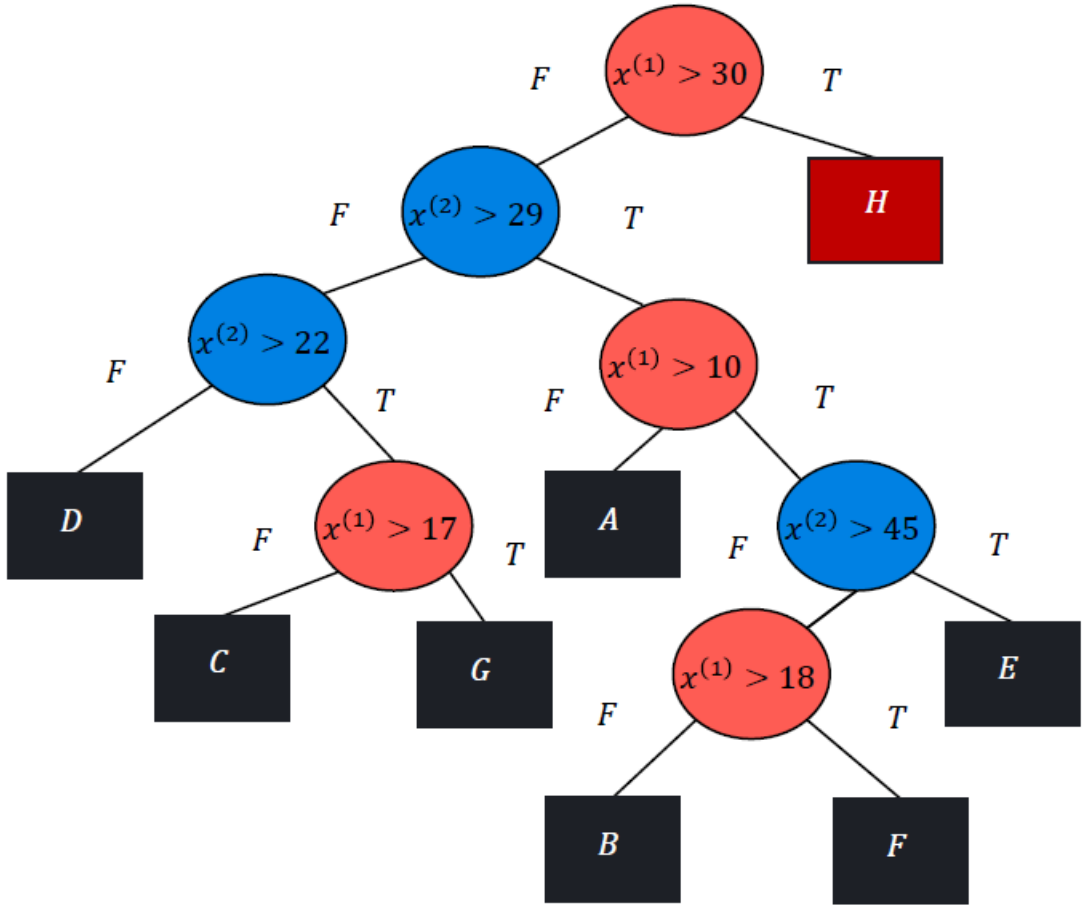
HOW ANOMALIES ARE DETECTED?





HOW ARE ANOMALIES DETECTED?





APPLICATIONS

- Security
- Banking,Insurance,Finance
- Fraudulent Email
- Healthcare
- Manufacture

LESSON-2

AUGMENTED ARTIFICIAL INTELLIGENCE (A2I)

ARTIFICIAL INTELLIGENCE (AI) – AN INSIGHT

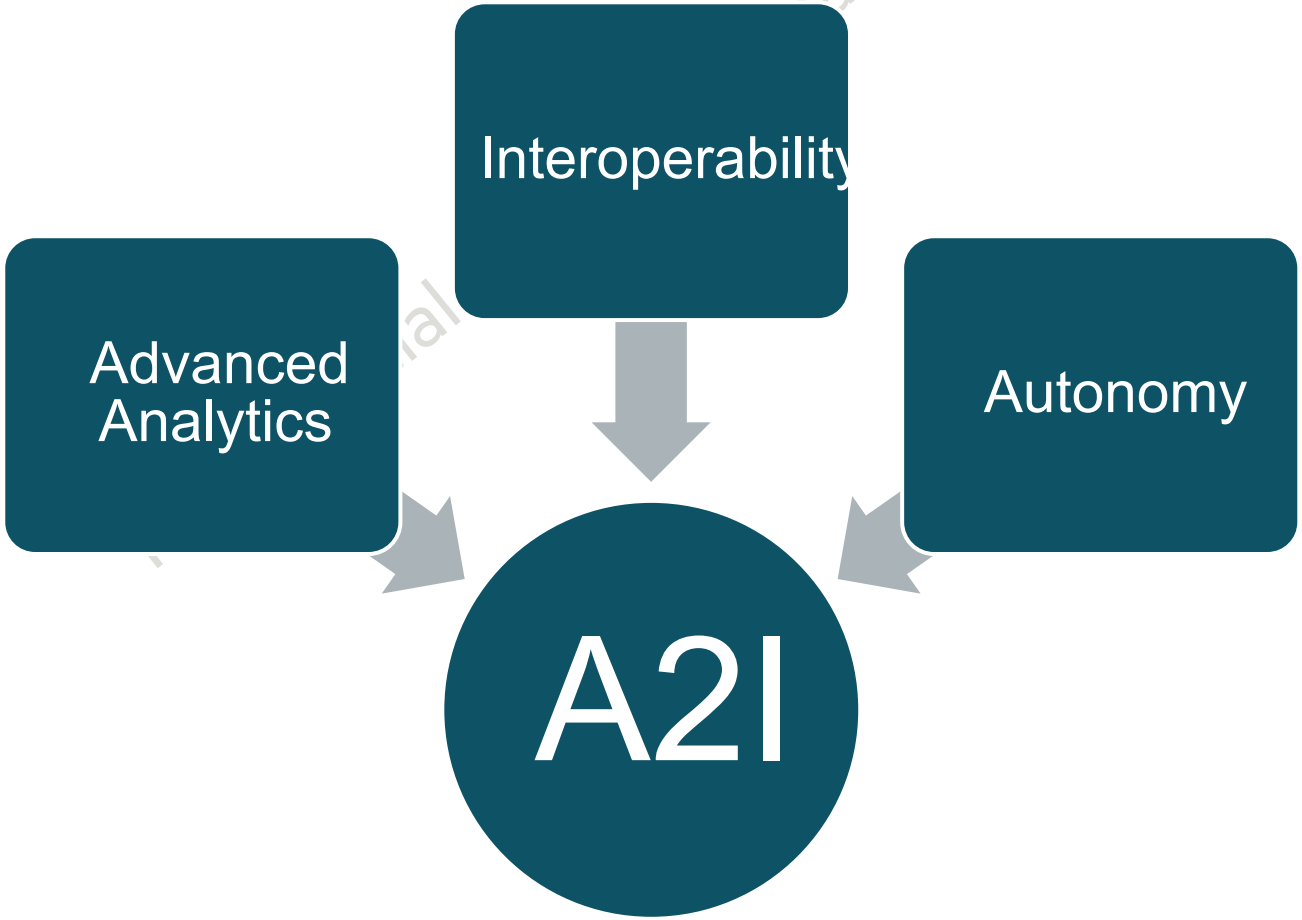
- Artificial Intelligence (AI)
  - Simulates Human intelligence and mimics the cognitive aspects
  - Machine learning (ML) – A subset of AI
  - AI & ML – aids in decision making function
  - Example: Automated classification of fruits into ripe and un-ripe categories

- The Biggest problem in AI:
  - Tries to replace human intelligence with machine intelligence
  - Cannot mimic human intelligence
  - This is exactly why self driven cars cannot be driven during bad weather conditions
  - Hence Augmented Artificial Intelligence (AAI)

AUGMENTED ARTIFICIAL INTELLIGENCE (A2I) – AN INSIGHT

- Augmented Artificial Intelligence (AAI)
  - Amplifies the functionality of AI
  - Machine intelligence – excellent in ingesting large volumes of data and execute repetitive tasks
  - Human Intelligence – Brilliant in exception handling, creativity and emotional intelligence
  - Combining Machine intelligence and human intelligence
  - Tries to coexist with humans and not replace them with machine intelligence
  - SO….**AAI – THE BEST OF BOTH THE WORLDS**

Functional aspects of A2I



FUNCTIONAL ASPECTS OF A2I

- Advanced analytics
  - Provisions associative recall (which makes the neural network to learn, interpret and remember the inherent relationships with unrelated aspects)
  - Facilitates a real-time processing of the information
  - Precise predictions and decision making can be achieved when combined with human intelligence

- Interoperability
  - AAI provides seamless access to real-time data
  - Helps to develop cognitive ability in the networks and machines
  - Suitable for self diagnosis and self healing applications

- Autonomy
  - Better performance without human interventions
  - Self maintain, sensory abilities, interpretation power, reaction to operative environment, navigational abilities and task performance

A2I & BIG DATA

- Big data
  - High volume, variety and variability
  - Cannot be analyzed using conventional data processing aspects
  - Helps to identify patterns and develop actionable insights with AI and ML
  - Better data driven efficient decision making with AI & ML
  - Better prediction with larger datasets

- A2I in Big Data analytics
  - Reduces the time taken for cleaning the data
  - Gives more time to analyze, visualize and recommend suggestions in Extract, Transform and Load (ETL) process

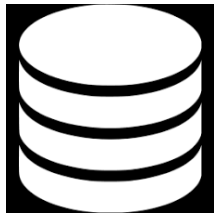
DIFFERENCES BETWEEN AI AND A2I

Artificial Intelligence(AI)	Augmented Artificial Intelligence(A2I)
Machine based systems are involved in the process of decision making	A combination of machines and human beings are involved in the process of decision making with NLP
Final output is machine dependant	The final output is not entirely dependant on the machines, but are human based as well
The decisions are often not suitable in real-time scenarios	The results are humane and ethical in nature
The pace of the decision making is very fast, based on the computational capability of the machines	Decision making process is slower as human inputs are involved as well
The concept of "Black box" makes it difficult to explain the intermediate steps in decision making	Easier to explain the steps involved to arrive at the output stage due to the absence of black-box approach

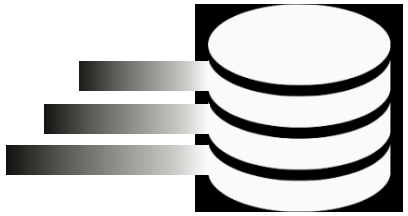
LESSON 3:

SmartSim

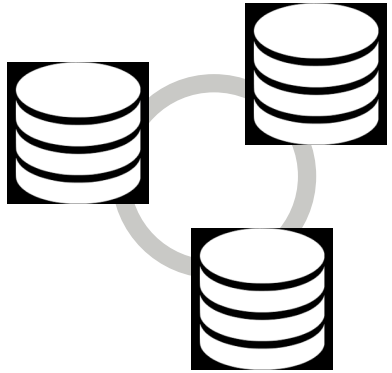
CONTEXT



Data at rest: HPE AI Development Environment

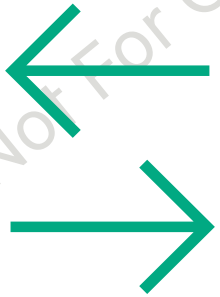


Data in motion: SmartSim



Distributed data: Swarm Learning

MOTIVATION – OPENING UP POSSIBILITIES



Simulation

C/C++/Fortran

Machine learning

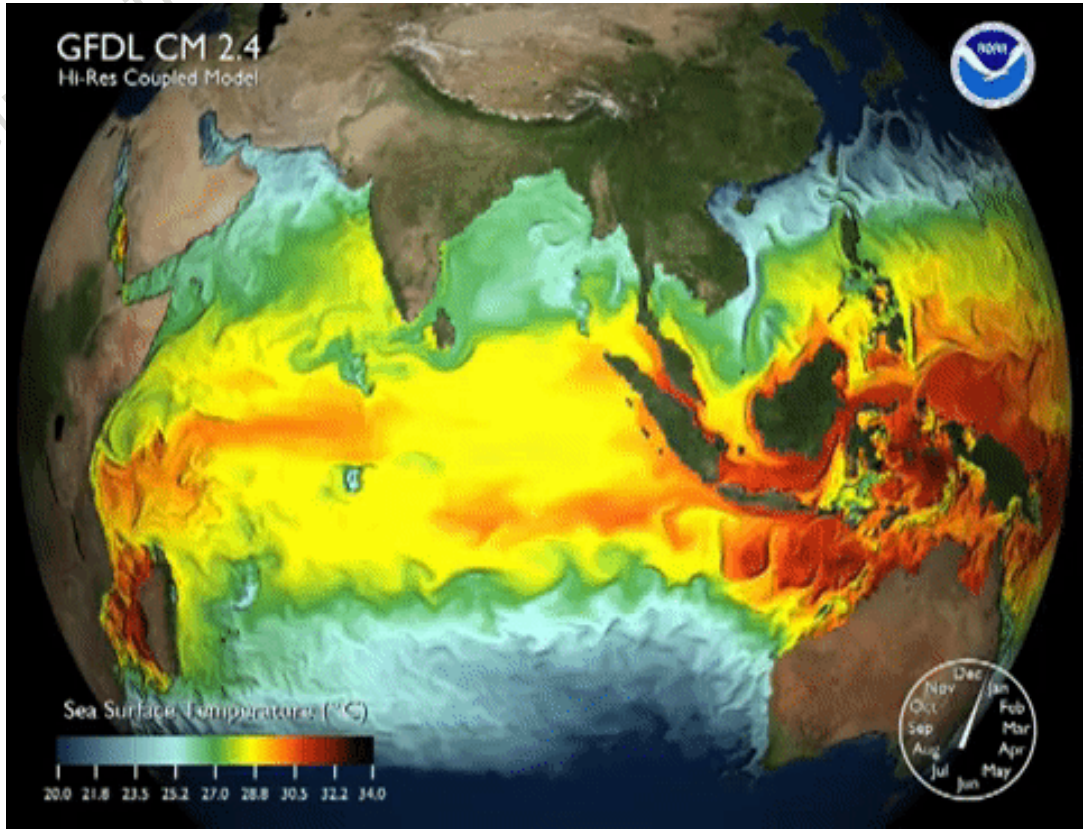
Python

Add Machine learning methodology to your simulations with just couple of lines of cod

SIMULATION MODELS

What kind of simulations/codes are we talking about here?

- **Chemistry:** QMCPACK, GAMESS
- **Fluid Dynamics:** OpenFOAM, SU2
- **Ocean:** MOM6, MIT-GCM
- **Coupled Climate:** E3SM, CESM
- **General Multi-Physics:** MOOSE
- **Molecular Dynamics:** LAMMPS, OpenMM, CP2K, GROMMACS
- **Biology:** NAMD
- **Quantum Chromodynamics:** CHROMA, MILC
- **Atmosphere:** CAM, SPCAM



SMARTSIM – RETAINING VALUABLE RESOURCES

Question – Should we abandon C/C++/Fortran and rewrite in Python?

Tribal/community knowledge

Time-proven techniques

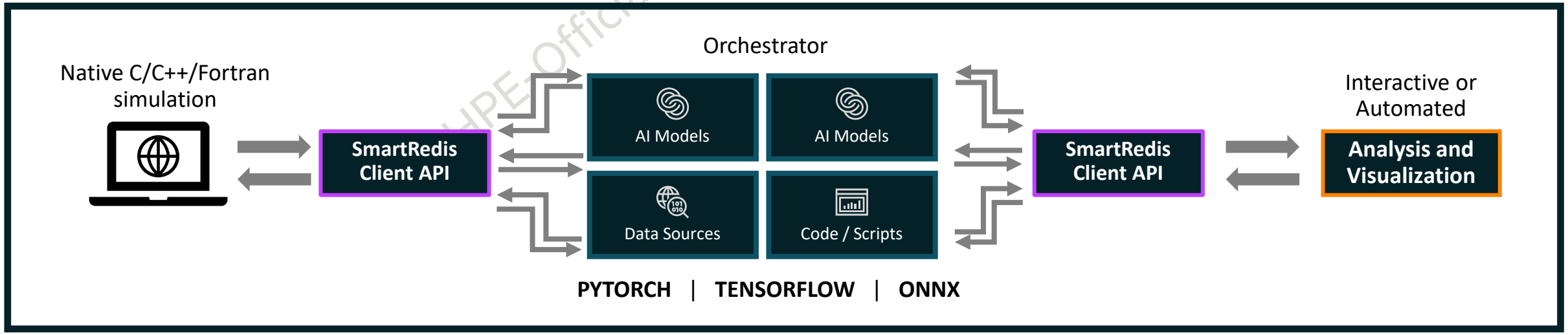
Development cost

Solution – SmartSim future forwards the value of your investment

SMARTSIM– CAPABILITIES

- Use Machine Learning (ML) models in existing Fortran/C/C++ simulations
- Communicate data between C, C++, Fortran, and Python applications
- Train ML models and make predictions using TensorFlow, PyTorch, and ONNX
- Analyze data streamed from HPC applications while they are running

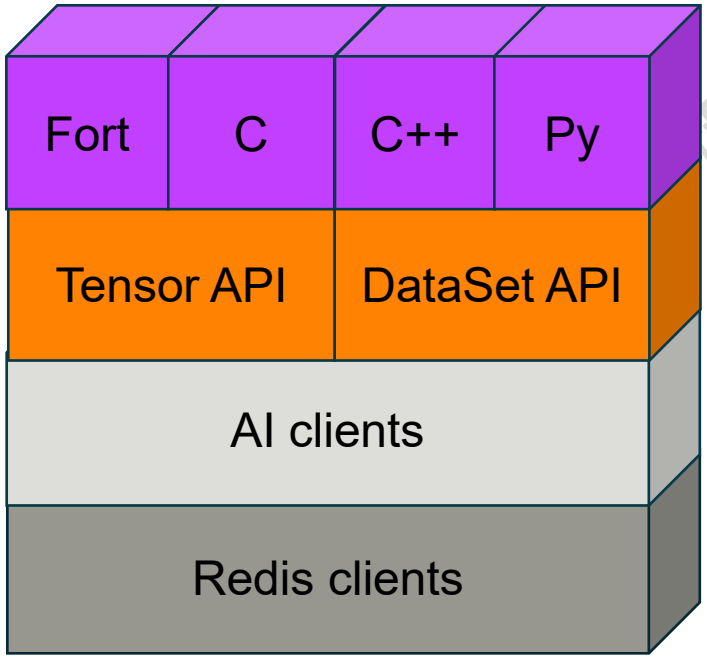
All of these can be done without touching the filesystem, i.e. data-in-motion



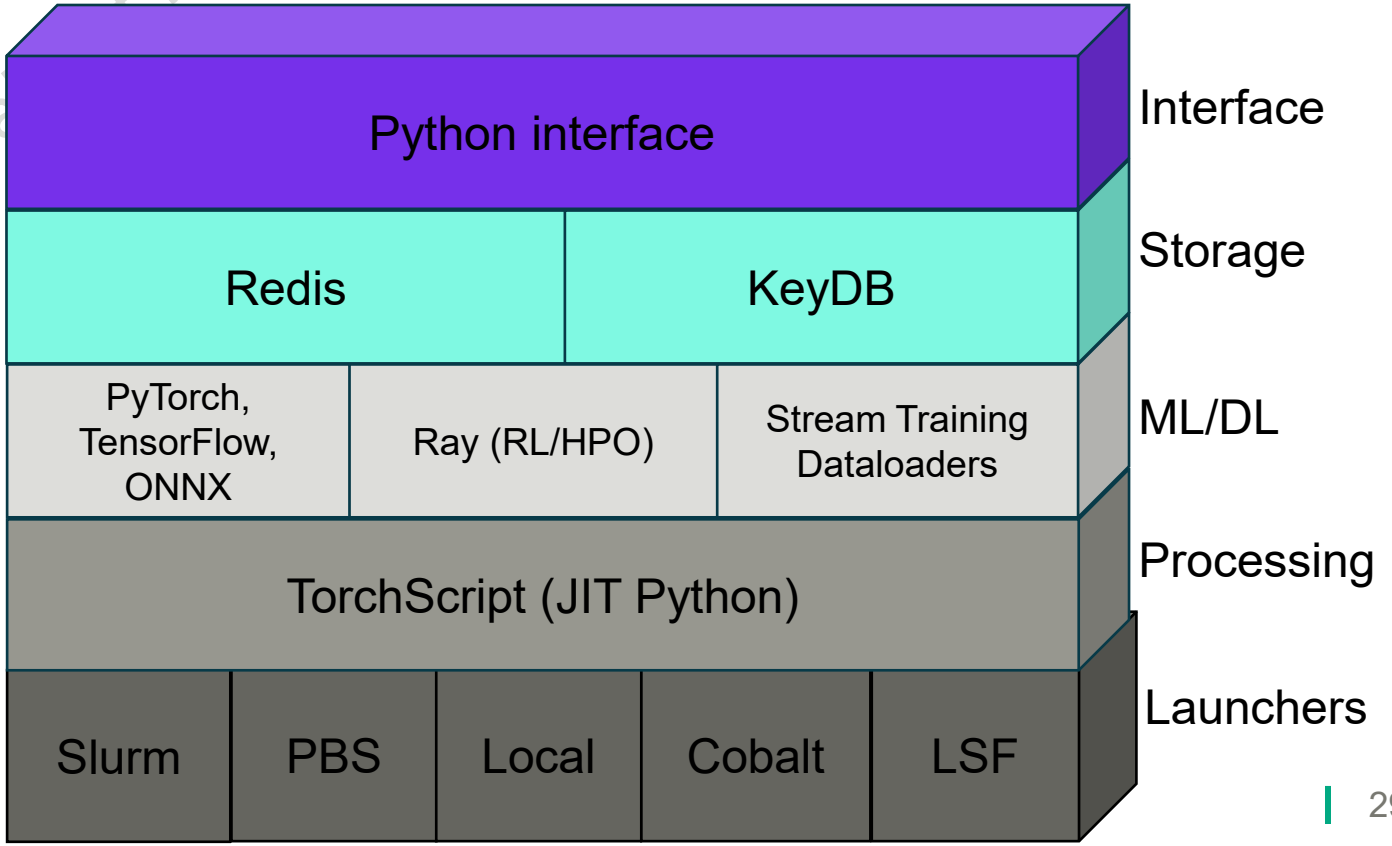
SMARTSIM - COMPONENTS

- SmartSim is composed of two libraries
- The **infrastructure library** facilitates application and ML/DL deployment
- The **client library** enables any application to communicate data and remotely execute ML/DL models and scripts from C++, C, Fortran, and Python

Client library (SmartRedis)



Infrastructure library



SMARTREDIS API

Tensor data types
- Double
- Float
- Int8
- Int16
- Int32
- Int64
- UInt8
- UInt16

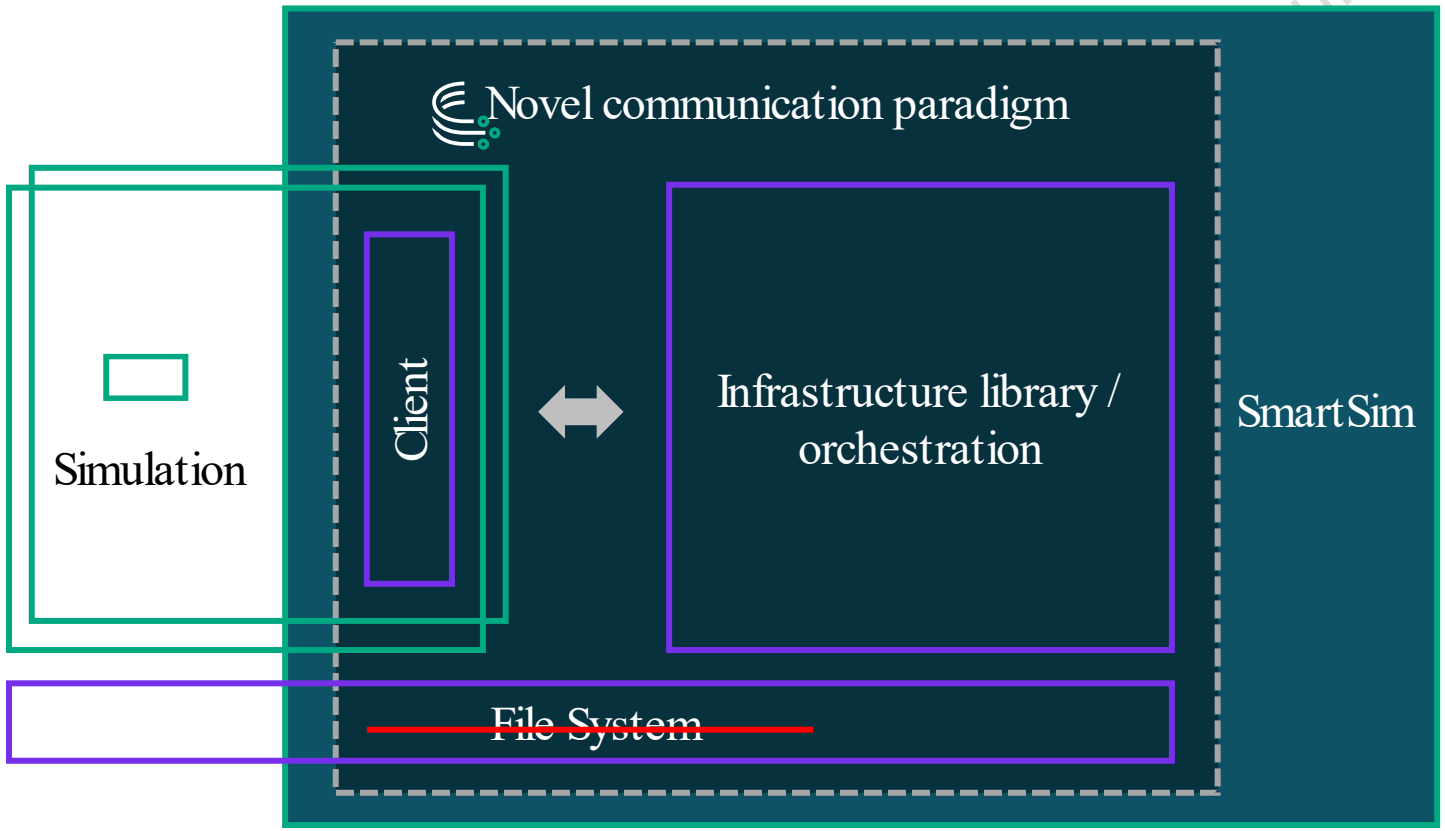
Metadata types
- Double
- Float
- Int64
- Int32
- Uint64
- Uint32
- String

Same API in Python, Fortran, C, C++

Client data structure	API	Description
Tensor	Client.put_tensor()	Set a tensor
	Client.get_tensor()	Retrieve a tensor
	Client.unpack_tensor()	Fill user-provided memory space with retrieved tensor
Model	Client.set_model()	Store and distribute ML/DL model
	Client.get_model()	Retrieve model
	Client.run_model()	Execute a ML/DL model on some stored data
Script	Client.set_script()	Store and distribute TorchScript script
	Client.get_script()	Retrieve a TorchScript script
	Client.run_script()	Run a TorchScript function within a script on some data

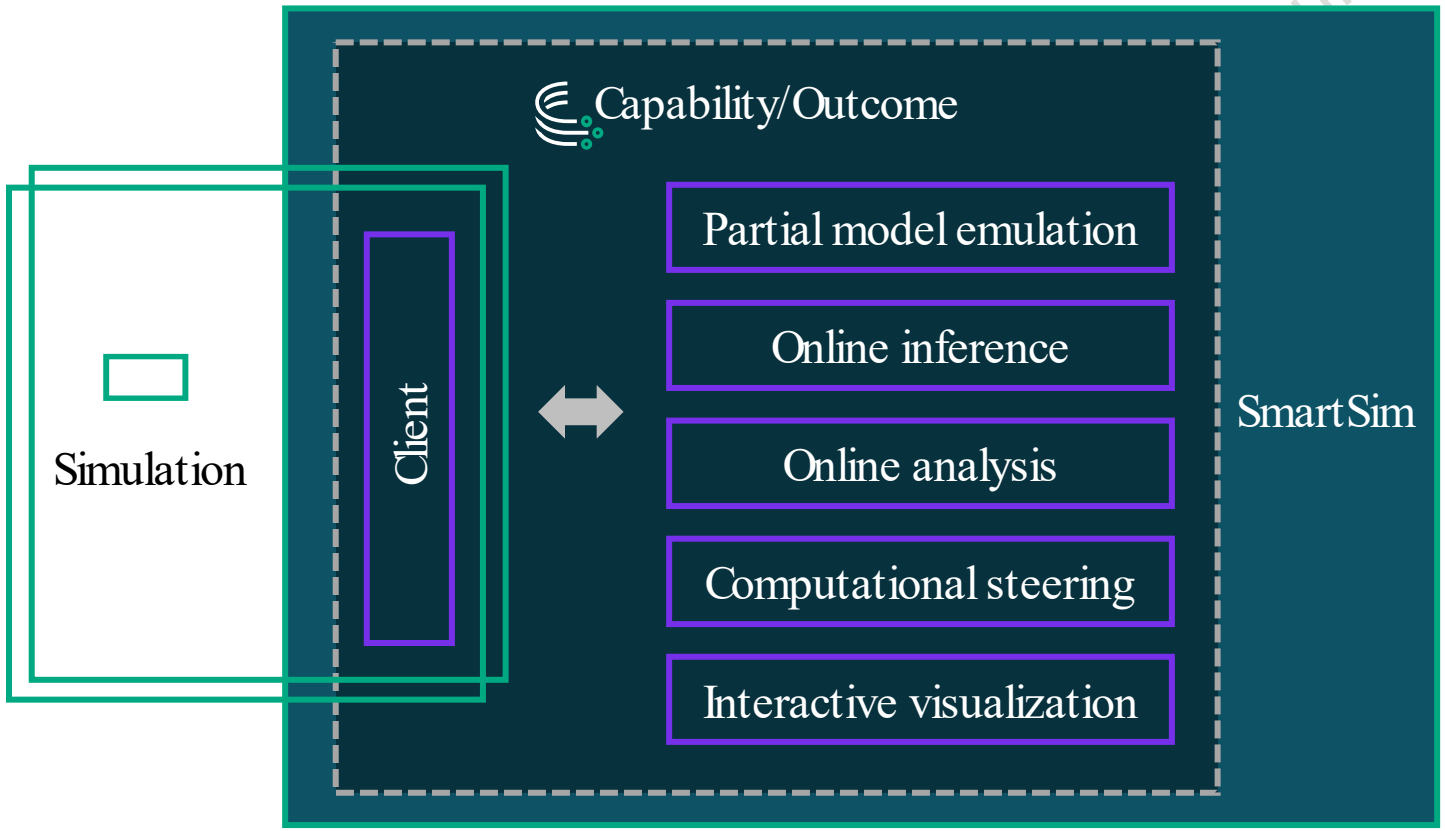
DataSetactions	API	Description
Construct	DataSet.add_tensor()	Add a tensor to the DataSet
	DataSet.add_meta_scalar()	Add a metadata value to a scalar field
	DataSet.add_string_scalar()	Add a metadata value to a string field
Store	Client.put_dataset()	Set a DataSet
	Client.get_dataset()	Retrieve a DataSet
Inspect	DataSet.get_tensor()	Get a tensor from the DataSet
	DataSet.unpack_tensor()	Fills user-provided memory space with tensor

DATA FLOW IN SMARTSIM
Streamlining management and data exchange



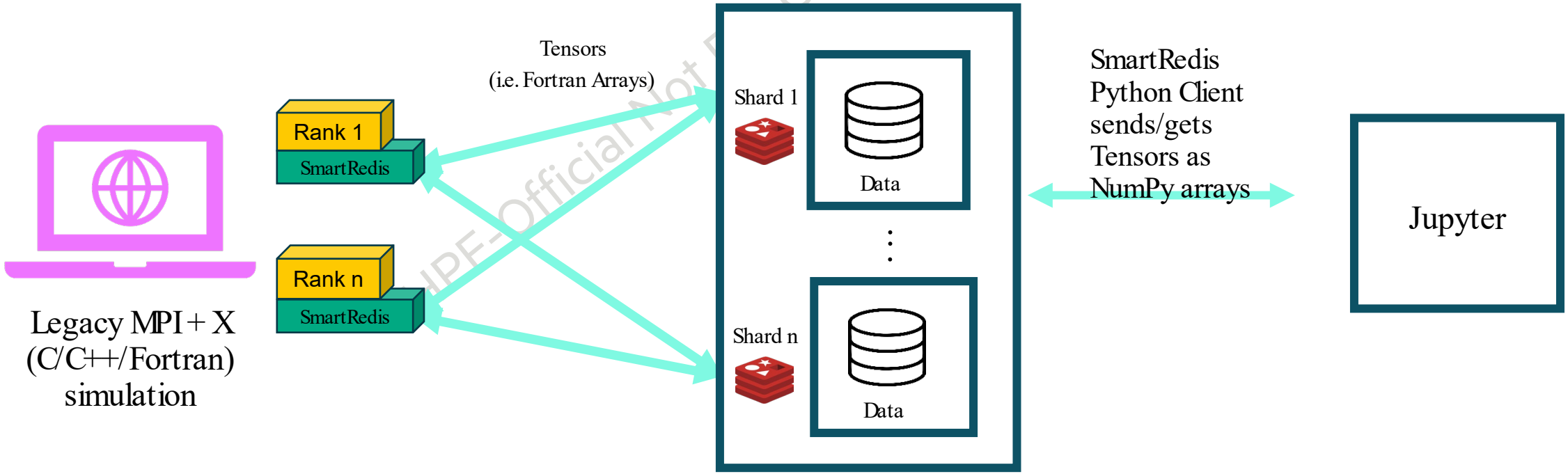
HOW CAN YOU USE SMARTSIM?

Key characteristic = data exchange between simulation and in-memory database



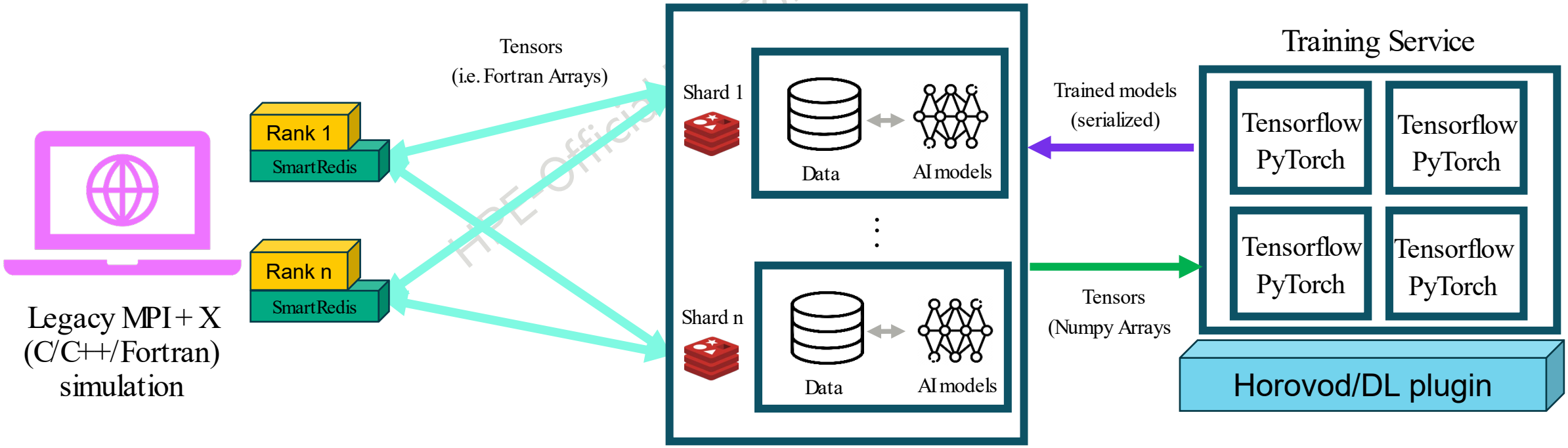
ONLINE ANALYSIS

- Stream data from C/C++/Fortran simulations for analysis, and visualization in real time
- Arrays transformations between languages handled by SmartRedis (Fortran -> NumPy)
- Clients are lightweight: single static library
- No reading/writing to slow shared filesystems



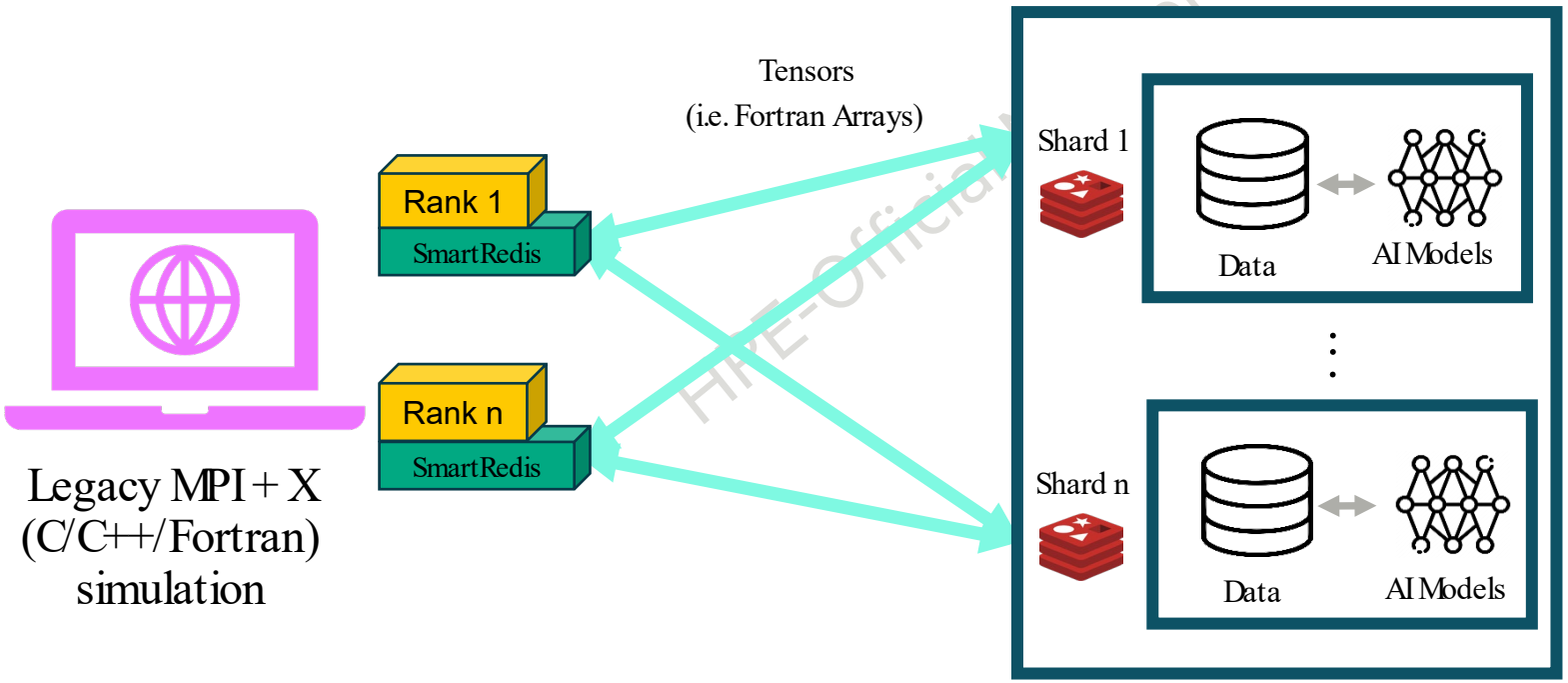
ONLINE TRAINING

- SmartRedis Client can be used inside DataLoaders for TensorFlow and PyTorch to perform Stream Training
- Trained models can be checkpointed, saved and later sent to the database for online inference
- Can be set from any language and called from any language (i.e. model set from Python, called from Fortran)



ONLINE INFERENCE

- Call TensorFlow, PyTorch, ONNX (scikit-learn, spark, etc) models stored inside the database
- Result of inference stored at new key for retrieval when needed. Minimizes data movement
- Multiple levels of batching – database auto-batches to optimize GPU memory usage

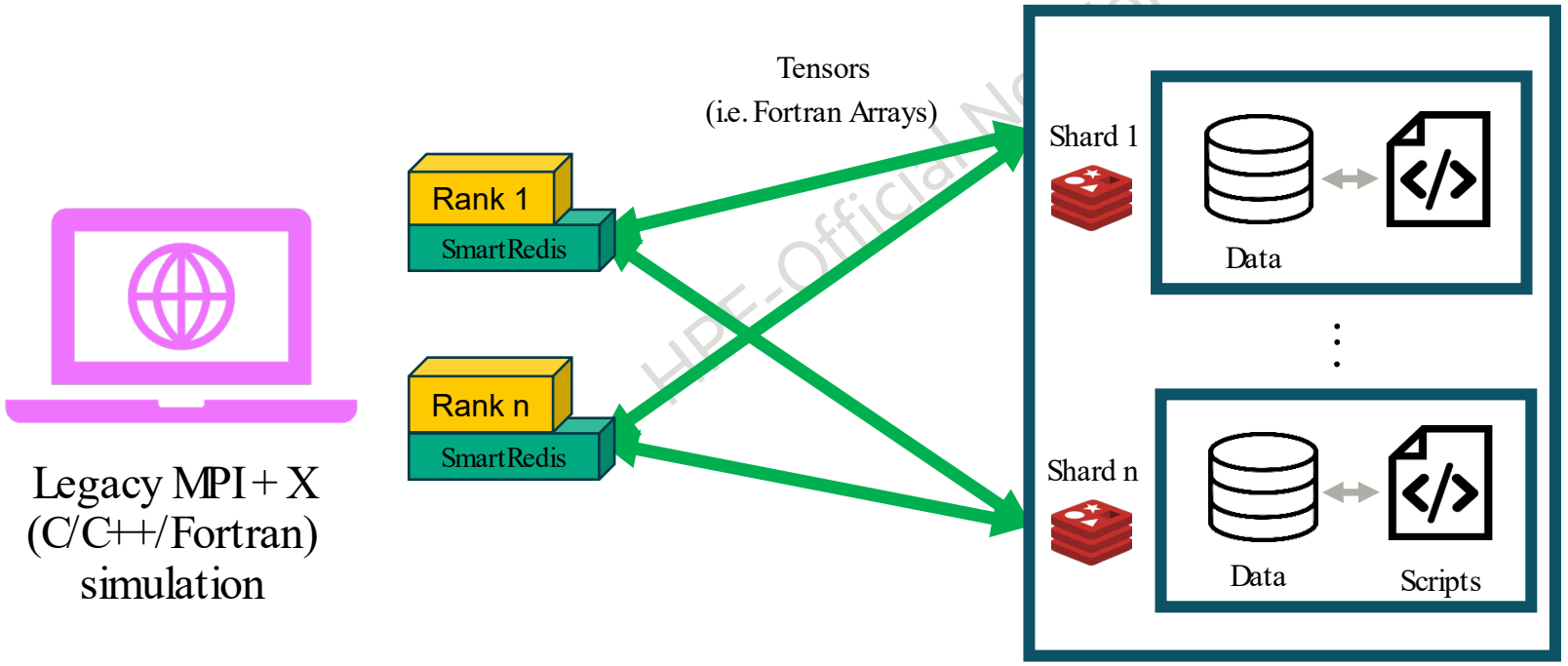


red=Infrastructure blue=Client

```python
import io
import torch
import numpy as np
import torch.nn as nn
from smartsim import Experiment
from smartsim.database import Orchestrator
from smartredis import Client

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv = nn.Conv2d(1, 1, 3)

    def forward(self, x):
        return self.conv(x)

exp = Experiment("svd", launcher="local")
db = Orchestrator()
exp.start(db)

client = Client(address="127.0.0.1:6379",
                cluster=False)
n = Net()
example_forward_input = torch.rand(1, 1, 3, 3)
module = torch.jit.trace(n, example_forward_input)
model_buffer = io.BytesIO()
torch.jit.save(module, model_buffer)

client.set_model("cnn", model_buffer.getvalue(),
                 "TORCH", device="CPU")
client.put_tensor("input", example_forward_input.numpy())
client.run_model("cnn", inputs=["input"], outputs=["output"])

output = client.get_tensor("output")
print(f"Prediction: {output}")

exp.stop(db)
```

ONLINE PROCESSING

- Same SmartRedis Clients can call TorchScript scripts (JIT-traced Python) stored inside database
- Can be set from any language and called from any language (i.e. set from Python, called from Fortran)
- Simple example calculates Singular Value Decomposition (SVD) of tensors streamed to database



red=Infrastructure blue=Client

```python
import numpy as np
from smartsim import Experiment
from smartsim.database import Orchestrator
from smartredis import Client

def calc_svd(input_tensor):
    return input_tensor.svd()

exp = Experiment("svd", launcher="local")
db = Orchestrator()
exp.start(db)

client = Client(address="127.0.0.1:6379",
                cluster=False)
tensor = np.random.randint(0, 100,
                           size=(30, 10, 2))
client.put_tensor("input", tensor)
client.set_function("svd", calc_svd)
client.run_script("svd", "calc_svd",
                  "input", ["U", "S", "V"])
U = client.get_tensor("U")
S = client.get_tensor("S")
V = client.get_tensor("V")

print(f"U: {U}, S: {S}, V: {V}")
exp.stop(db)
```

VALUE PROPOSITION – SMARTSIM

Bridging together HPC and AI with a new modeling paradigm

Extending value of current models/approaches

- Ready-to-consume popular model integrations for immediate use
- Unlocks previously prohibitive real-time data access
- Ability to scale training and analyses to consume full system resources
- Reduce/eliminate wasted development cycles and learning curves for high-cost resources
- Easily integrate open-source alternatives with ML/Data Science stack

Higher productivity

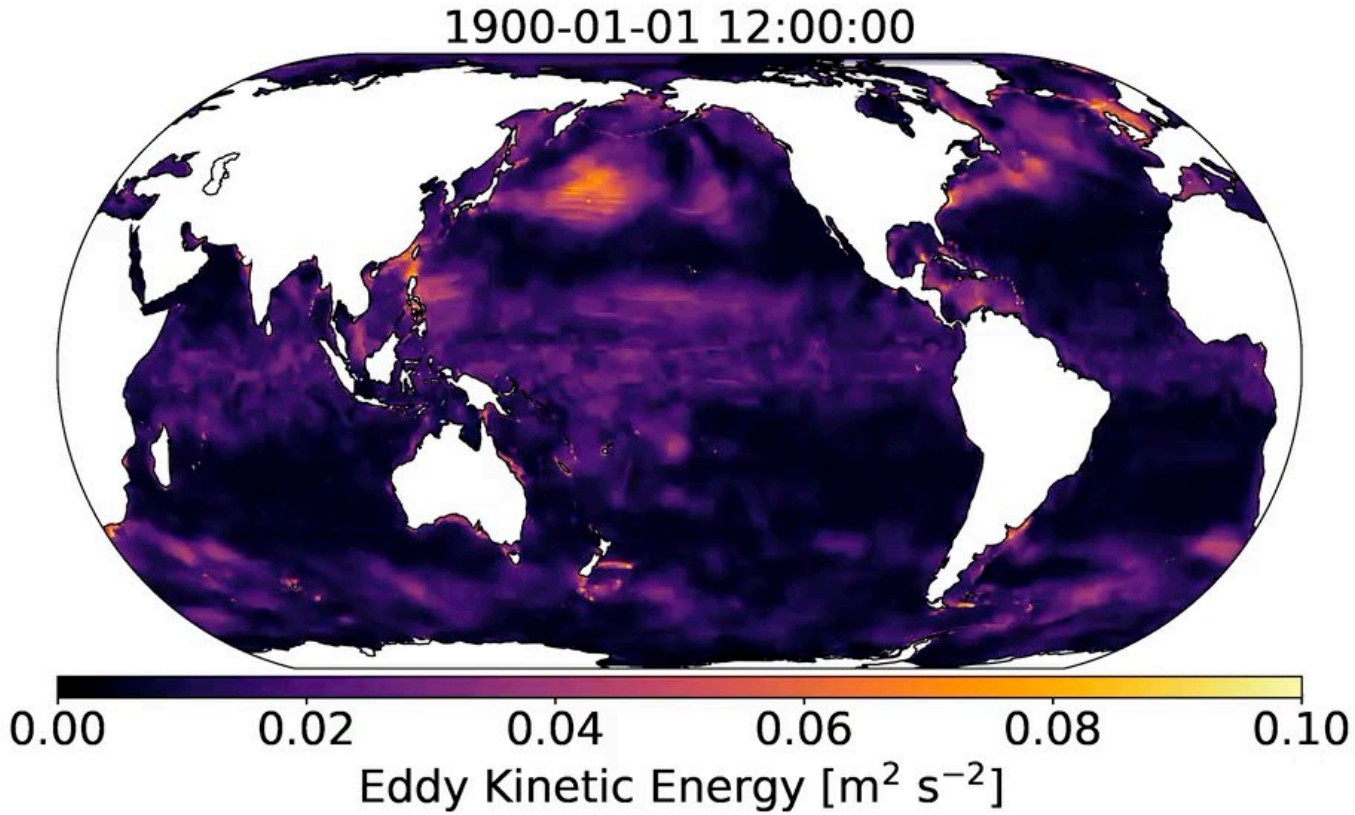
- Experiment Tracking at the individual and team level, removing potential loss of data and time
- Ability for Domain Experts and Machine Learning Engineers to combine expertise into a single workflow
- Ability to integrate with open-source visualization frameworks for on-the-fly insight
- Faster model and analytics iterations to quickly find successful working hypothesis
- Maximize expensive resources' ability to focus on development and not HPC infrastructure

Faster performance and TTM

- Integrate modern ML stack into legacy codes
- Management of complex workflows on Slurm, PBS, LSF, and Cobalt
- Execute, monitor, and analyze legacy codes with modern ML methods
- Data communication between legacy codes and Python easily and at scale
- Analyze data streamed from simulations in real time
- Enables domain and ML experts to focus on methodology not underlying infrastructure

USER PERSONAS

	Domain scientist/SME	Machine learning engineer	IT MLOps	LOB leader	Executive
Who?	• Subject matter domain expert	• Hands on model developers	• Bring models to production	• Director of research • Leader of Model Dev team	• C-level sponsor
What do they care about?	• Augmenting their simulations/codes using latest AI software stack • Real-time simulation control	• Accessing data from simulations/historical codes • ML data feeds at scale • Understanding data from those codes	• Provide necessary infra for ML Engineering team to train and deploy models • Efficiently managing infra	• Get the most out of their expensive people and infrastructure resources. • Get the most out of ML models to achieve business objectives	• Lead ML/AI transformation for entire company
Key pain points	• Weak ML methods and ML coding skills • Manually implementing ML features through piecemeal code	• Scaling to production-ready models • Weak HPC knowledge	• Difficult to stay in front of evolving ML infra needs • Ability to scale resources on-the-fly	• Optimizing ROI • Visibility and Risk management • Optimal utilization of infrastructure/resources	• Roadmap for AI expansion
Value proposition	**Faster time to Insight** • Enhance existing methodological approaches with ML • Greenfield analytic capability • Solutions level capability	**Faster time to models and market** • Iterate quicker for better methods • Focus on ML development, not HPC infrastructure	• **Accelerated ML dev cycles** • Single pane of glass for process/systems insight • Reduced support req for SME and ML Engineer	**Cost savings** • ML team collaboration and efficiency • Predictable and transparent pricing	**ROI** • Faster time to market • Relationship with a transformation partner that has data scientists, AI infrastructure experts, and advisory
Role	• Key influencer	• Key influencer	• Key influencer	• Decision-maker/veto	• Veto

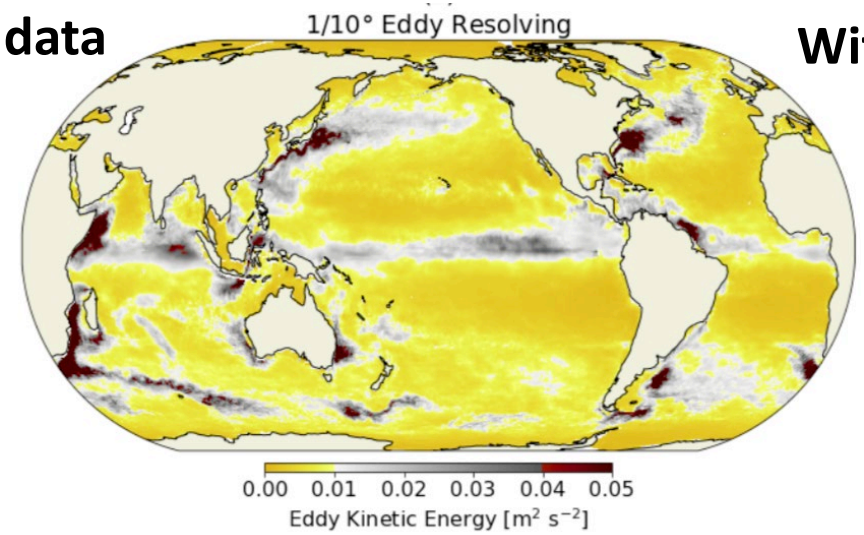


SmartSim has shown breakthrough capability in the climate and weather vertical.

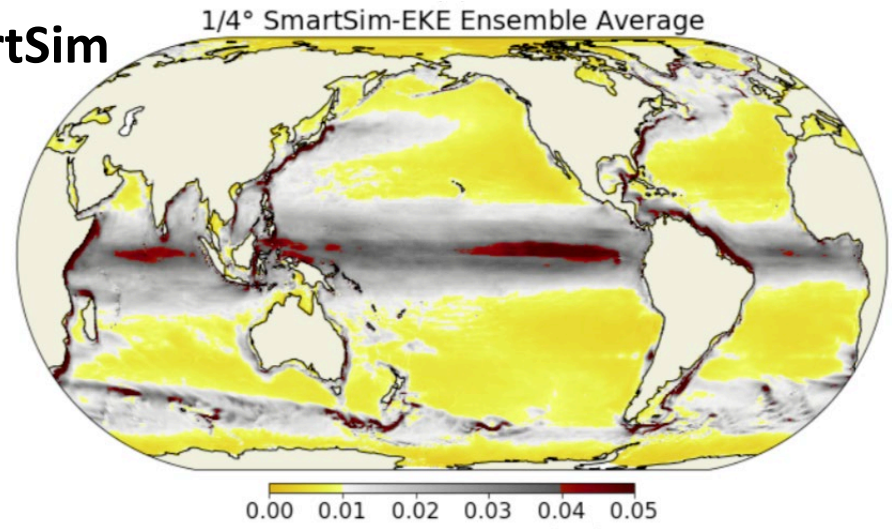
First research to ever successfully integrate a production-grade, global climate model (MOM6) with a ML surrogate for a stable decadal simulation.

RESULTS

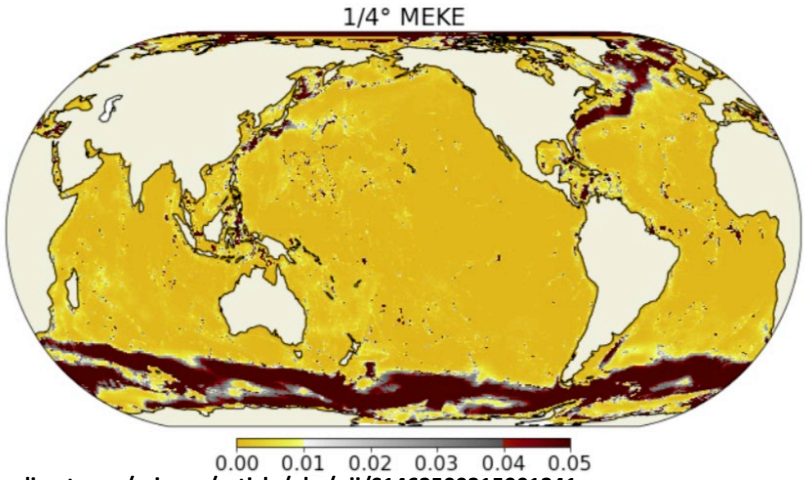
“Truth” data



With SmartSim



Previous state of the art



https://www.sciencedirect.com/science/article/abs/pii/S1463500315001341

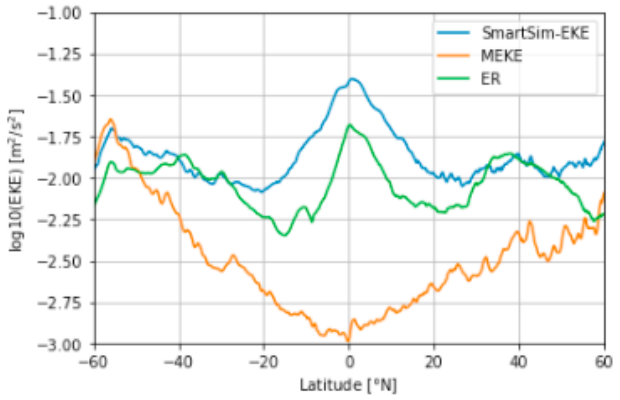


Fig. 7. Zonally averaged EKE (on a log10 scale) as a function of latitude from the ER, SmartSim-EKE, and MEKE.

THANK YOU