EFFICIENTLY COMBINING MACHINE LEARNING WITH OPENFOAM USING SMARTSIM

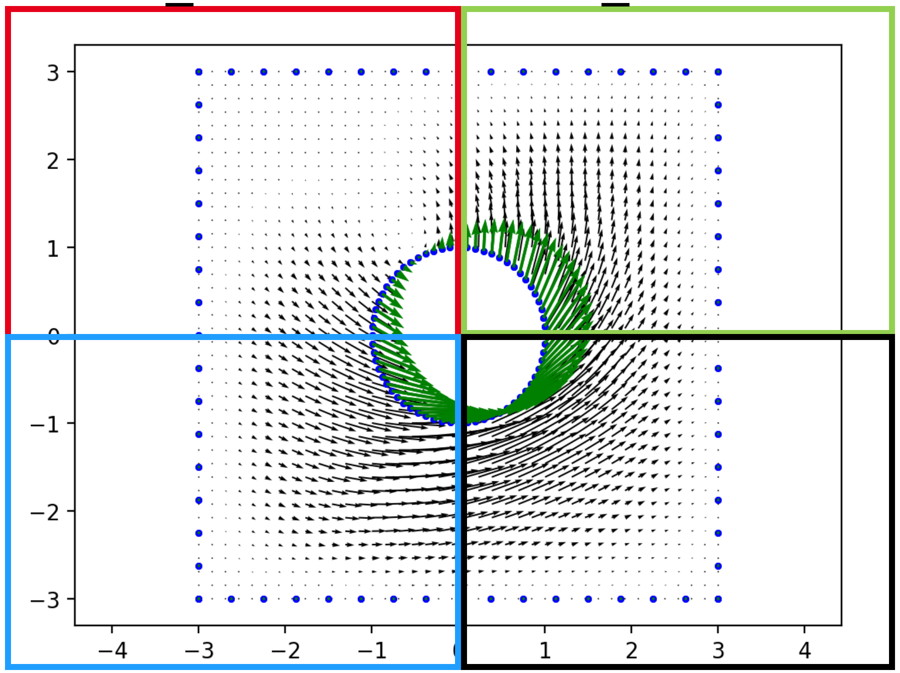
T. Maric | A. E. Shao | A. Rigazzi | M. Ellis | E. Fadeli | A. Weiner

TECHNISCHE UNIVERSITÄT DARMSTADT

OPENFOAM + ML

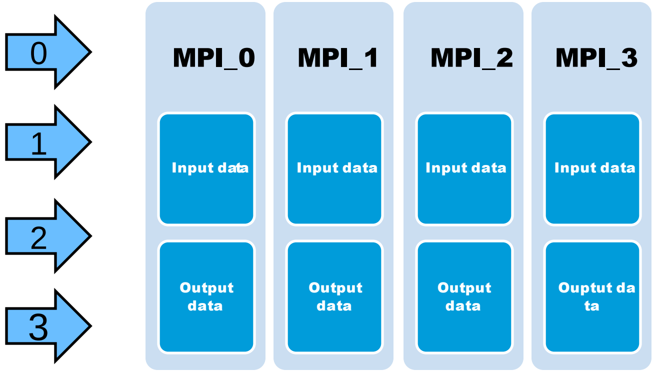
while (runTime.loop())

MPI_Rank 0 MPI_Rank 1

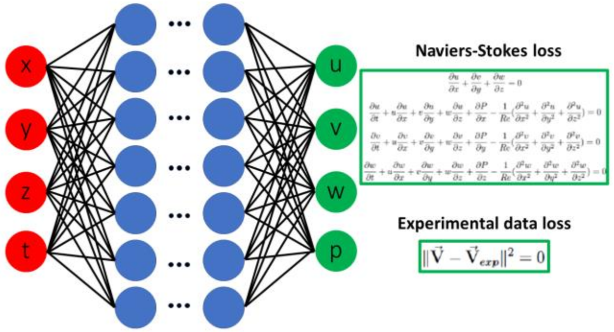


MPI_Rank 2 MPI_Rank 3

Training **Data**



- Agglomerates training data.
- Trains on other resources.

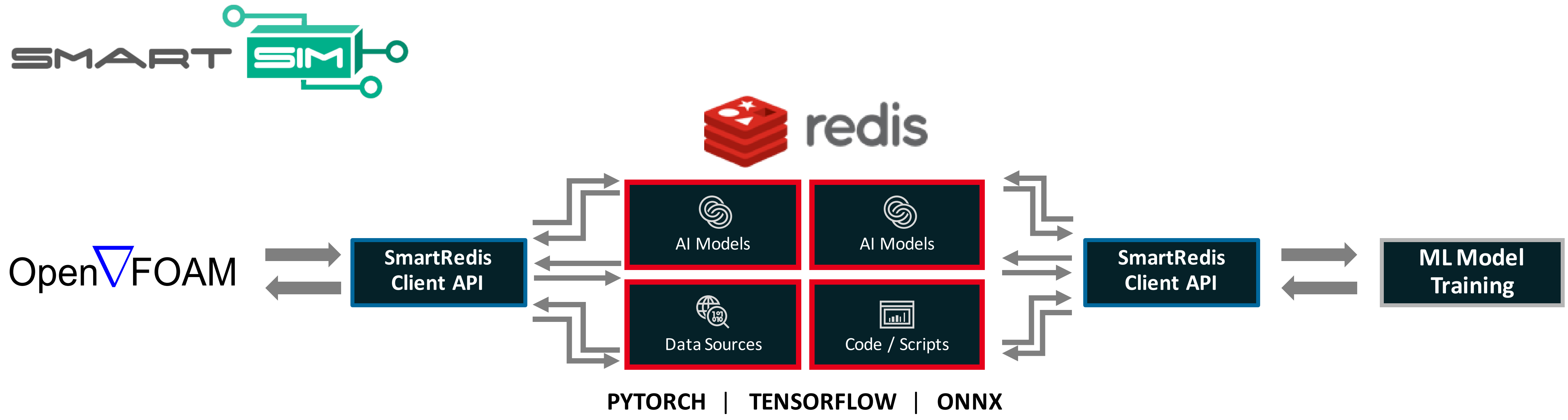


Physics-Informed Neural Network
Riccardo Munafò, CC BY-SA 4.0

- **Online training and inference** requires synchronization with the CFD algorithm.

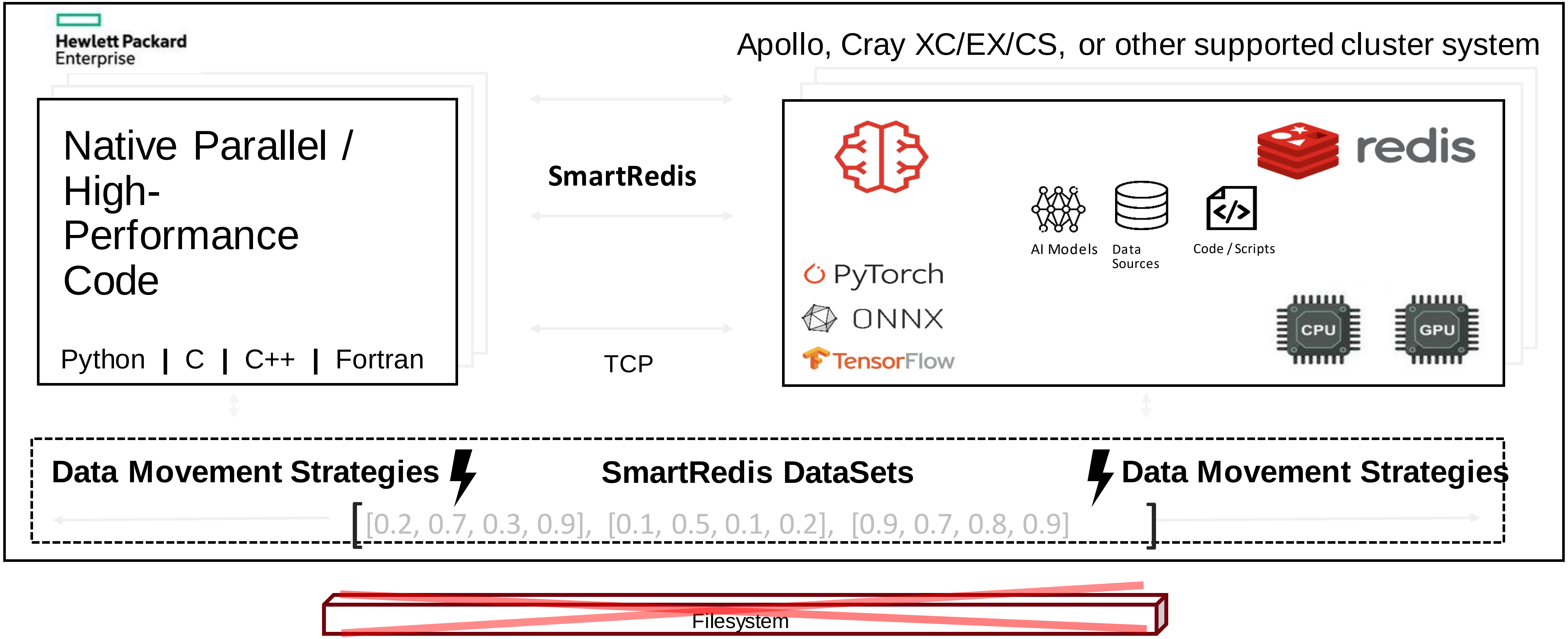
TECHNISCHE UNIVERSITÄT DARMSTADT

OPENFOAM + ML



- **SmartSim Orchestrator:** implementing the computational workflow.
  - Jupyter Notebook or Python script – straightforward API.
- **SmartRedis Database**: CFD data, trained model, model inference.
  - Straightforward API in C++ (!!) and Python.

SMARTSIM

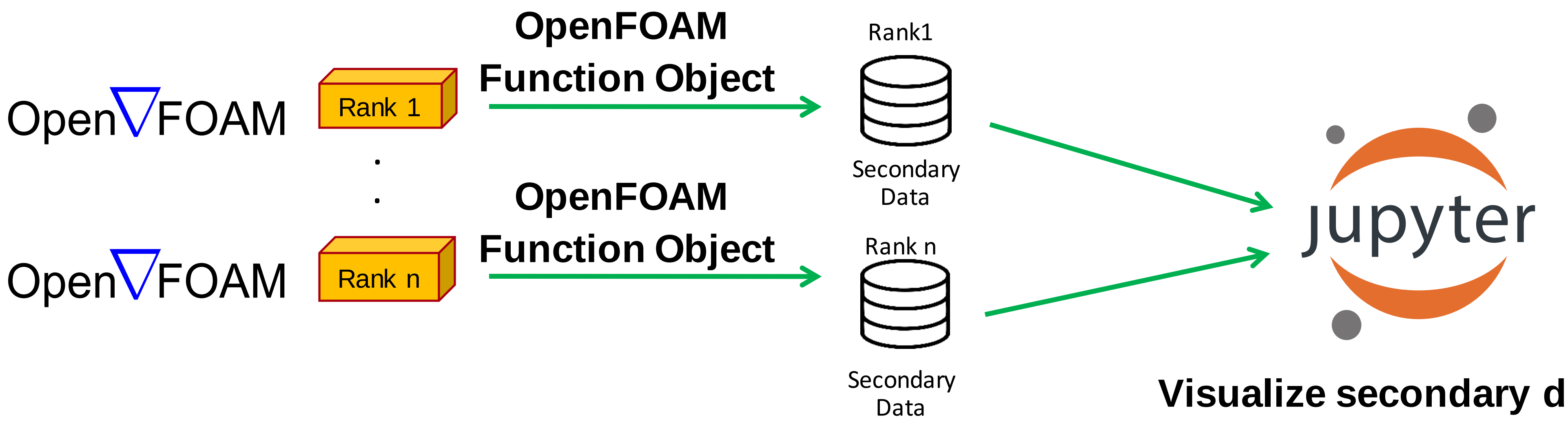

Hewlett Packard Enterprise
Apollo, Cray XC/EX/CS, or other supported cluster system
Native Parallel / High-Performance Code
Python | C | C++ | Fortran
SmartRedis
TCP
redis
AI Models
Data Sources
Code / Scripts
PyTorch
ONNX
TensorFlow
CPU
GPU
Data Movement Strategies
SmartRedis DataSets
Data Movement Strategies
[[0.2, 0.7, 0.3, 0.9], [0.1, 0.5, 0.1, 0.2], [0.9, 0.7, 0.8, 0.9]]
Filesystem

TECHNISCHE UNIVERSITÄT DARMSTADT

ONLINE POST-PROCESSING

User Story: I want to perform **postprocessing** while my **simulation is running.**

- We usually use OpenFOAM Function Objects to store secondary data (CSV) to disk.
- This data can be processed by a Jupyter Notebook and visualized and quantified live.

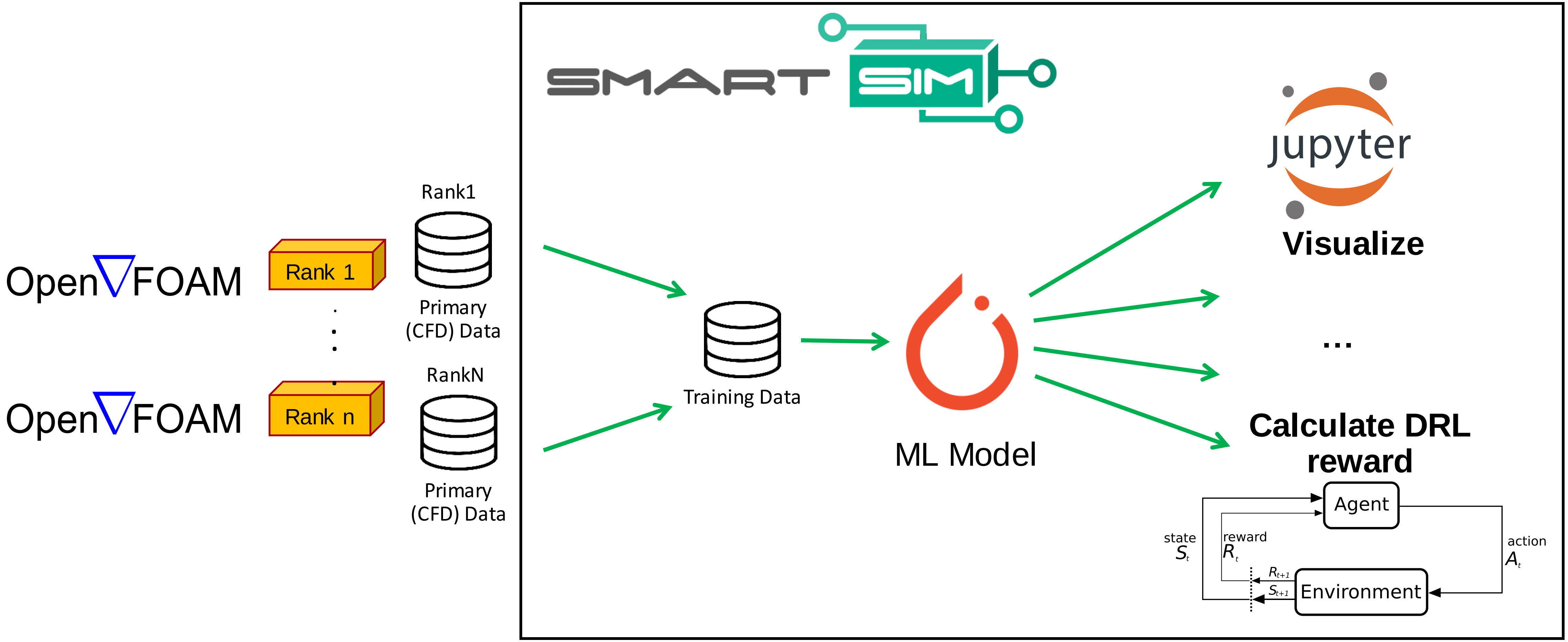


Visualize secondary data and stop bad simulations early

TECHNISCHE UNIVERSITÄT DARMSTADT

ONLINE POST-PROCESSING

User Story: I want to perform **postprocessing using Machine Learning** while my **simulation is running.**

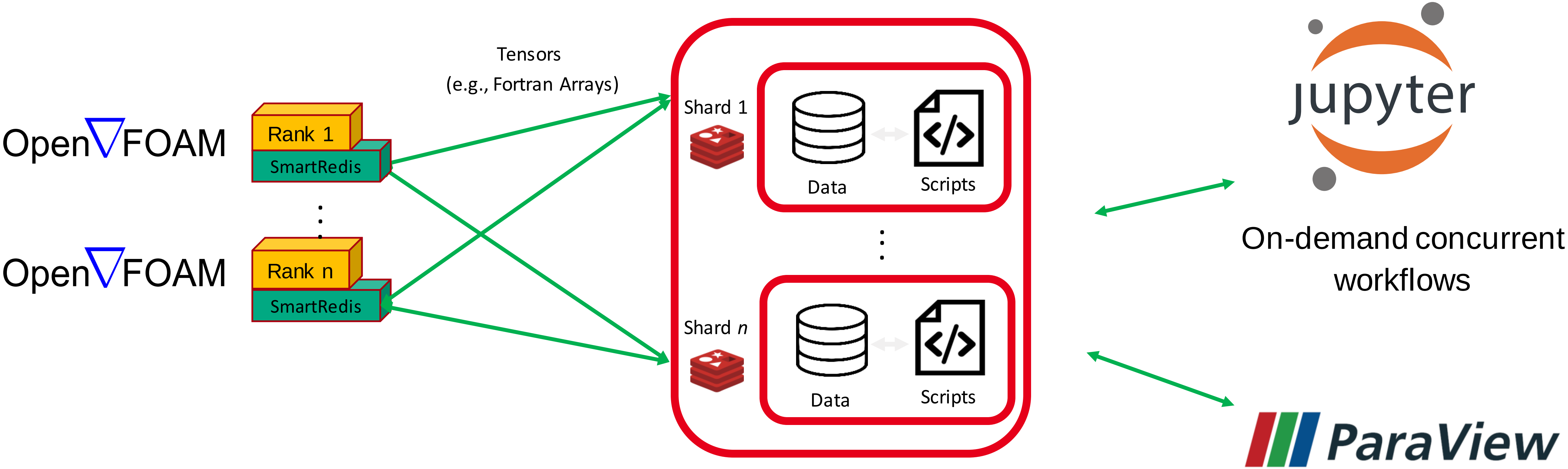


TECHNISCHE UNIVERSITÄT DARMSTADT

ONLINE POST-PROCESSING

User Story: I want to perform **visualization** and **analysis** while my **simulation is running**

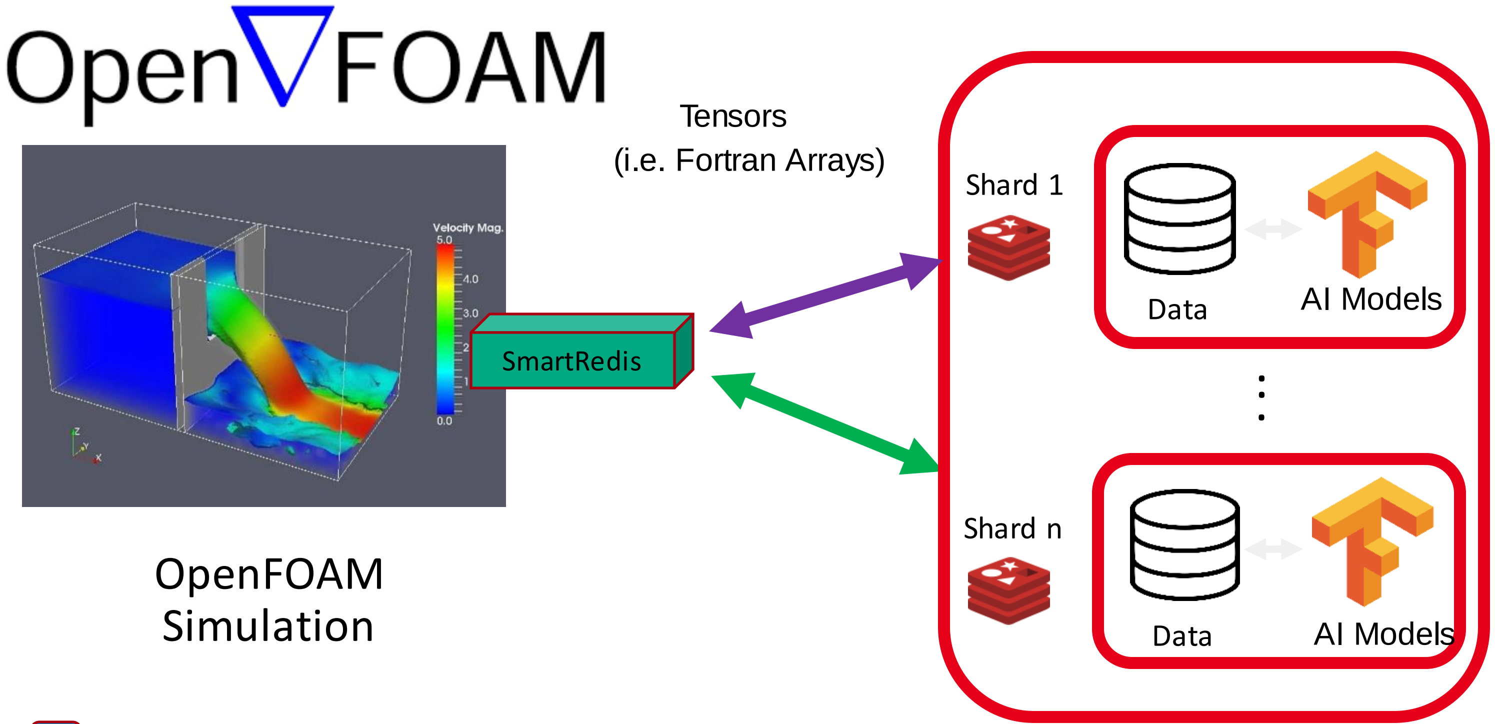
- Stream data from C/C++/Fortran simulations for analysis, and visualization in real time.
- No reading/writing to slow shared filesystems



TECHNISCHE UNIVERSITÄT DARMSTADT

ML FOR TURBULENCE PREDICTION IN OPENFOAM

- **User story:** I want to use a machine-learning model to reduce my time-to-solution.



ML inference results to simulation

Data to Orchestrator

“Orchestrator”

Datastore and inference engine

- **Tensorflow model** provides a pre-conditioned state that **reduces** number of solver **iterations**
- **Multi-stage** workflow in **one** Python **script**
    - Mesh decomposition
    - ML training
    - ML inference
    - Solver Integration
    - Analysis
    - Visualization

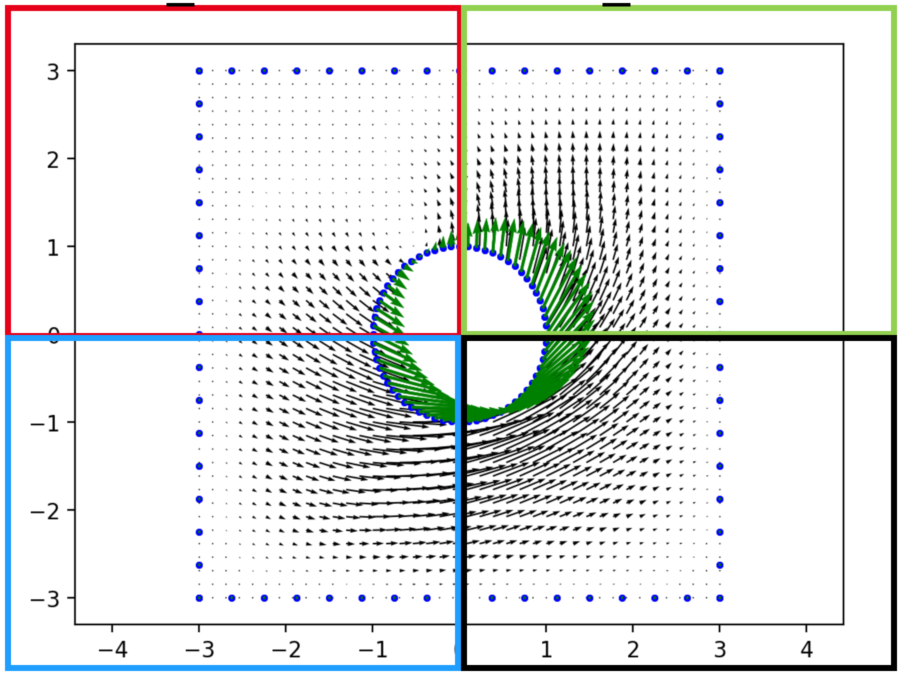
Code repo: https://github.com/CrayLabs/smartsim-openFOAM

ONLINE ML MESH MOTION

- **User story:** I want to use a machine-learning model to approximate mesh-motion displacements.

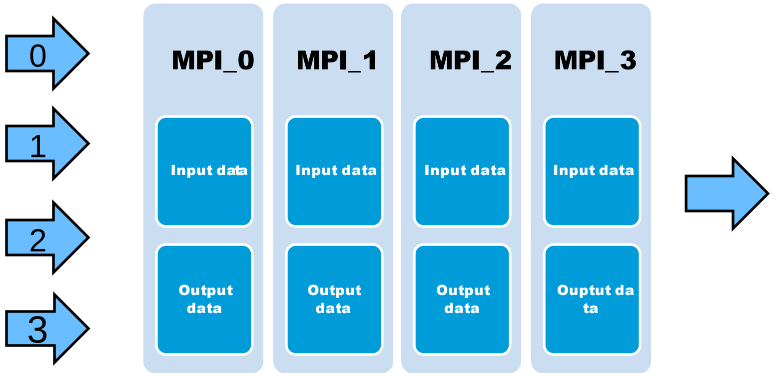
while (runTime.loop())

MPI_Rank 0 MPI_Rank 1

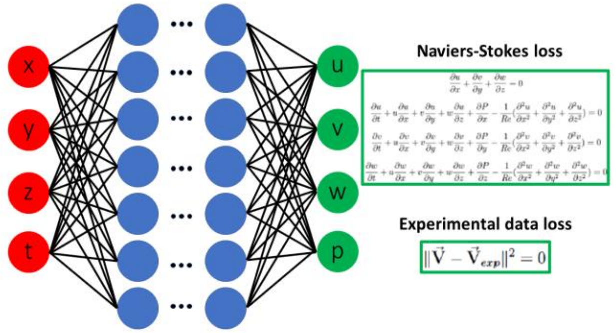


MPI_Rank 2 MPI_Rank 3

Training **Data**



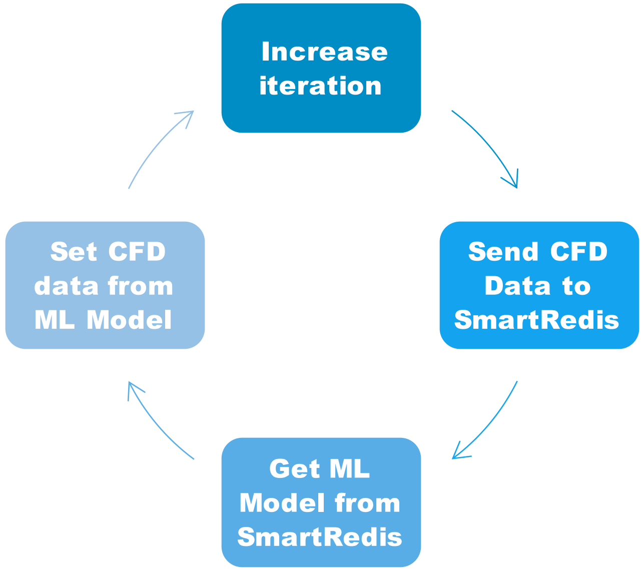
- Agglomerates training data.
- Trains on other resources.

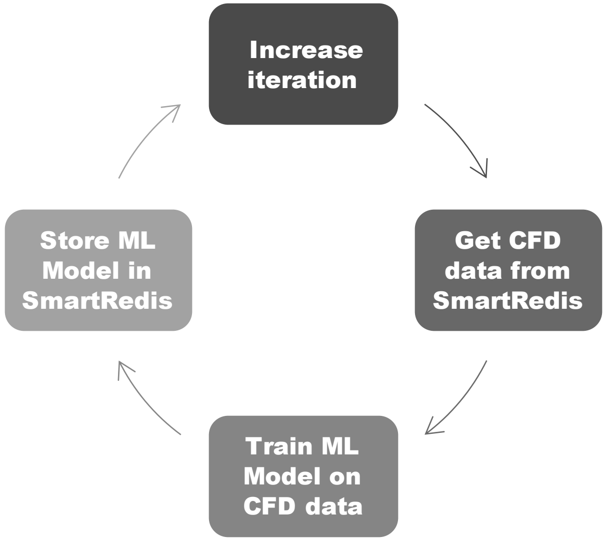


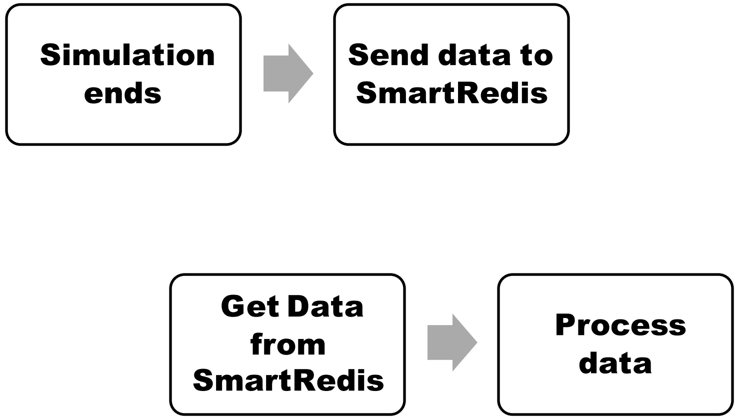
Physics-Informed Neural Network
Riccardo Munafò, CC BY-SA 4.0

- **Online training and inference** requires synchronization with the CFD algorithm.

DATA FLOW PATTERNS







- Concurrent data flow patterns require synchronization.
- Synchronization is done by checking for data (e.g. keys) in the SmartRedis database.

TECHNISCHE UNIVERSITÄT DARMSTADT

OPENFOAM FIELDS AS TENSORS

```
template<class T>
class UList
{
    // Private Data

        //- Number of elements in UList
        label size_;

        //- Vector of values of type T
        T* __restrict__ v_;
```

- All OpenFOAM Fields are ULists.
- UList<T> is a wrapper for T*
- Fields provide cdata() to reinterpret them as (void*).
- This is necessary not only for interpreting OpenFOAM fields as SmartRedis tensors, but also tensors in Machine Learning Frameworks.

TECHNISCHE UNIVERSITÄT DARMSTADT

SOLVER

```
turbulence->validate();

Info<< "Creating SmartRedis client..." << endl;
SmartRedis::Client client(false);

// Dimensions of communicated fields
std::vector<size_t> dims = {1, 1, 1};
dims[0] = mesh.nCells();

…

// Put pressure field into SmartRedis
client.put_tensor(p.name(), (void*)p.internalField().cdata(), dims,
                  SRTensorTypeDouble, SRMemLayoutContiguous);
```

SOLVER SCRIPT

```
from smartsim import Experiment
openfoam_case = "pitzDaily"
exp = Experiment("local-db", launcher="local")
db = exp.create_database(port=8000, interface="lo")
print(f"Creating database on  port {db.ports}")
exp.start(db)
print('DB started...')
blockMesh_settings = exp.create_run_settings(exe="blockMesh", exe_args=f"-case {openfoam_case}")
blockMesh_model = exp.create_model(name="blockMesh", run_settings=blockMesh_settings)
simpleFoam_settings = exp.create_run_settings(exe="simpleRedisFoam", exe_args=f"-case {openfoam_case}")
simpleFoam_model = exp.create_model(name="simpleRedisFoam", run_settings=simpleFoam_settings)
exp.start(blockMesh_model, block=True, summary=True)
exp.start(simpleFoam_model, block=True, summary=True)
exp.stop(db)
```

FUNCTION OBJECT

```
forAll(fieldNames_, fieldI)
{
    // Set field dimensions
    // - nCells x 1 for a scalar field
    // - nCells x 3 for a vector field
    // - nCells x 6 for a symmTensor field
    std::vector<size_t> dims = {size_t(mesh_.nCells()),
                                size_t(fieldDimensions_[fieldI])};

    if(fieldDimensions_[fieldI] == 1) // scalar field
    {
        // Get the cell-centered scalar field from the mesh (registry).
        const volScalarField& sField = mesh_.lookupObject<volScalarField>(fieldNames_[fieldI]);
        // Send the cell-centered scalar field to SmartRedis
        client_.put_tensor(sField.name(), (void*)sField.internalField().cdata(), dims,
                           SRTensorTypeDouble, SRMemLayoutContiguous);
```

Open∇FOAM

TECHNISCHE UNIVERSITÄT DARMSTADT

FUNCTION OBJECT SCRIPT

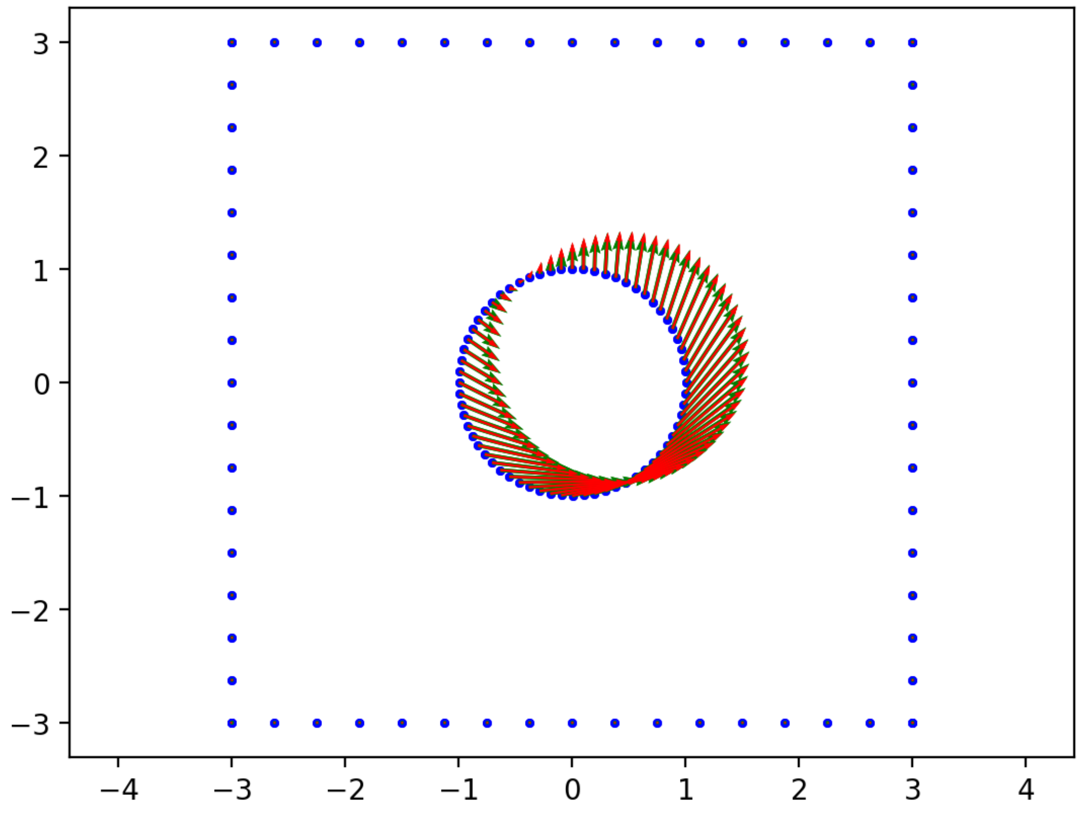
```
...
# Run simpleFoam solver
# - The pitzDaily/system/controlDict file contains the input for the function
#   object that will within simpleFoam_model connect and write to smartredis
exp.start(simpleFoam_model, summary=True, block=True)
# Get the names of OpenFOAM fiels from controlDict.functionObject
control_dict = ParsedParameterFile(os.path.join(of_case_name, "system/controlDict"))
client = Client(address=db.get_address()[0], cluster=False)
client.set_function("svd", calc_svd)
# Apply SVD to fields
field_names = list(control_dict["functions"]['smartSim']['fieldNames'])
...
```

TECHNISCHE UNIVERSITÄT DARMSTADT

ONLINE ML MESH MOTION

ONLINE ML MESH MOTION

- **User story:** I want to use a machine-learning model to approximate mesh-motion displacements.



Given Dirichlet (fixed value) conditions for mesh motion on mesh boundary patches $\{\partial\Omega_k\}_k$

- We use displacements $\delta(x,\, t) \in \Omega$

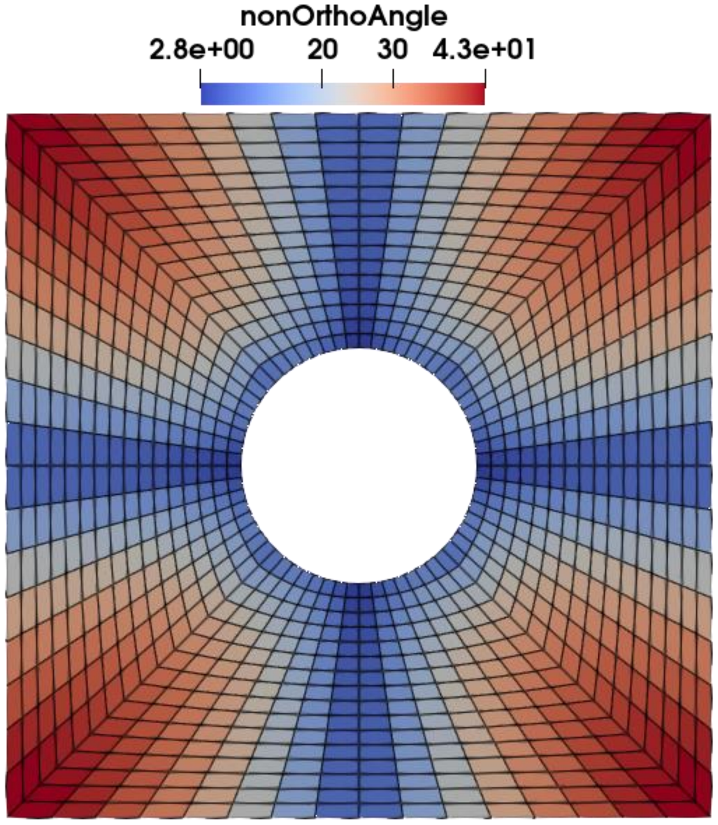
Approximate displacements in the solution domain as

$$\tilde{\delta}(x, t,\, \theta)$$

with θ as model parameters.

PDE-BASED MESH MOTION

- **User story: why would** I **want to** use a machine-learning model for mesh-motion?



PDE-based mesh motion like the Laplacian mesh motion

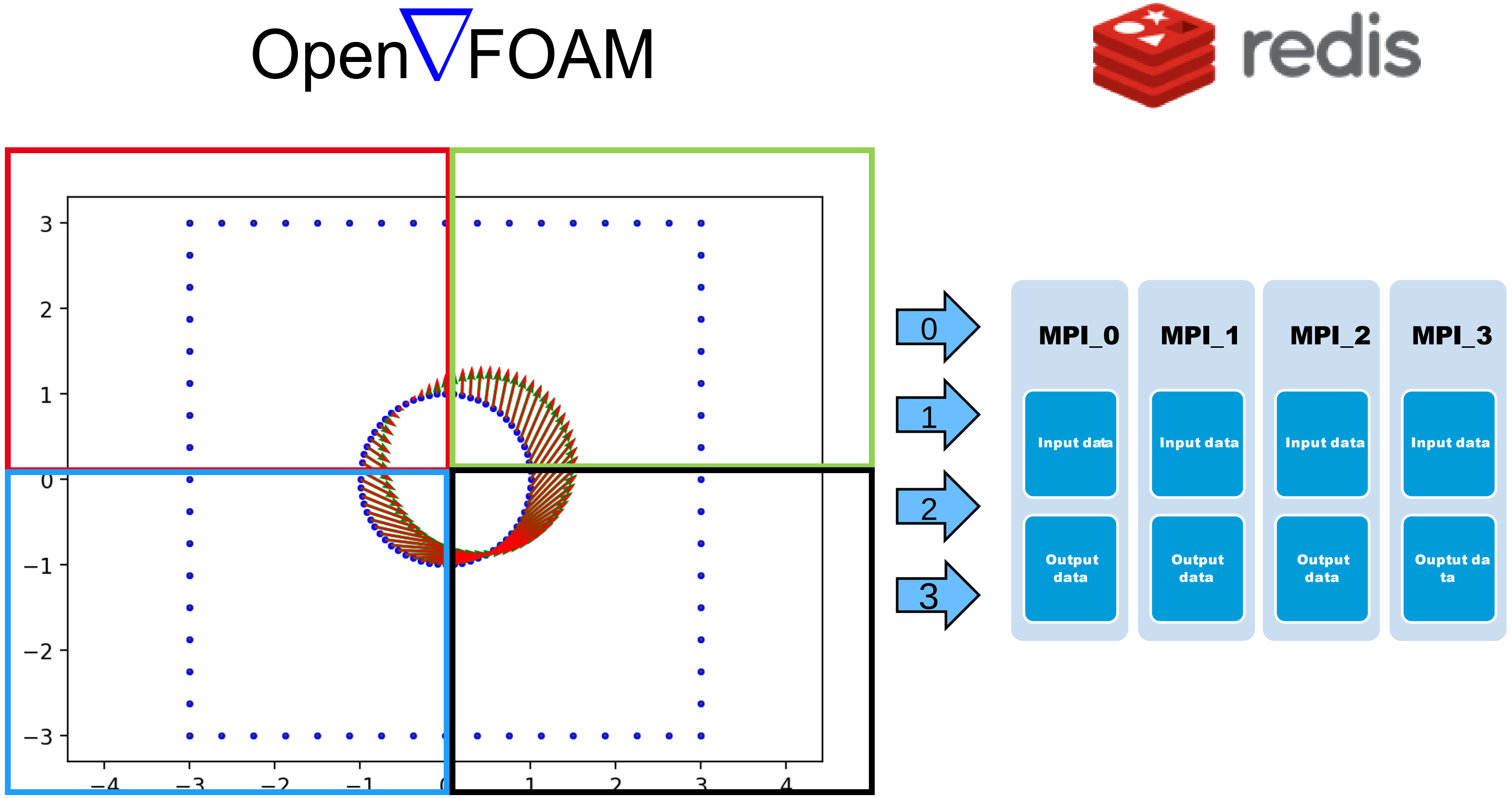
$$\nabla \cdot (\lambda \nabla \delta) = 0$$

$$\delta(x,t) = d(x,t), \, \forall x \in \{\partial\Omega_k\}_{\{k \in K\}}$$

- Approximatively solves the PDE on a deforming mesh with increasingly deteriorating quality (e.g. non-orthogonality).
- Approximation or interpolation-based mesh motion is smoother and can potentially deliver higher mesh quality for stronger deformations.

De Boer, A., Van der Schoot, M. S., & Bijl, H. (2007). Mesh deformation based on radial basis function interpolation. *Computers & structures*, *85*(11-14), 784-795.

OPENFOAM DATA STRUCTURE

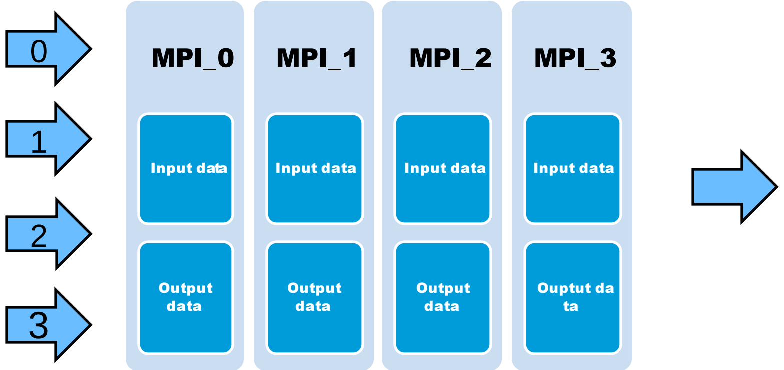


runTime++;

- Each MPI rank sends its own data to increase efficiency.
- We don't need MPI process boundary data.
- We don't need empty patches.
- Boundary field of a pointField stores a list of all boundary patches – zero length for those not on the MPI Rank.
- Input data are datasets:
  - point_patch_timeStep_rank
  - displ_patch_timeStep_rank

TECHNISCHE UNIVERSITÄT DARMSTADT

DATASETS AND AGGREGATION LISTS



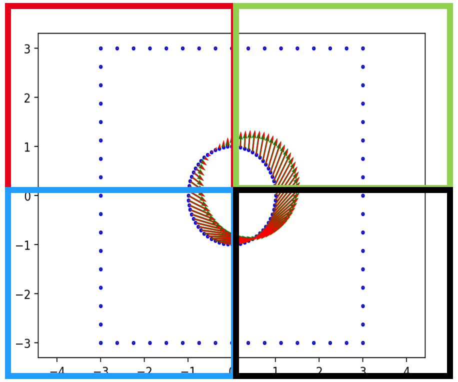
- Input data are datasets:
  - point_**patch**_timeStep_**rank**
  - displ_**patch**_timeStep_**rank**
- **What does this looks like for 30 patches and 50 MPI ranks?**
  - How to check if this data is available in SmartRedis?
- **Dataset: agglomerate over patches**
  - pointsDataset_timeStep_rank
  - displacementsDataSet_timeStep_rank
- **Aggregation list: agglomerate datasets over time steps**
  - If we know MPI_Comm_size and the time index, we know if all the data is available in SmartRedis. **How?**

TECHNISCHE UNIVERSITÄT DARMSTADT

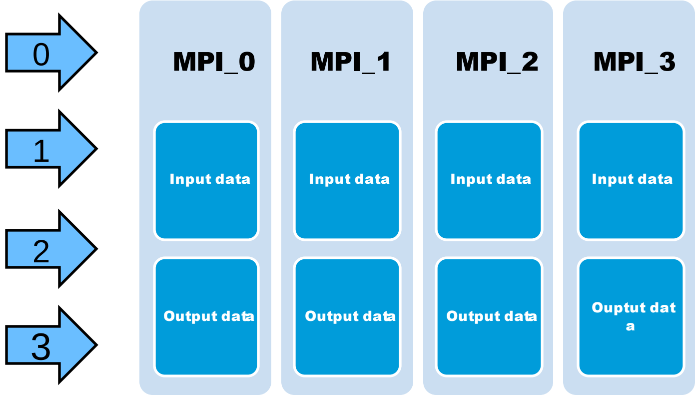
SYNCHRONISE TIME STEPS

OpenFOAM

runTime++;



redis



SMARTSIM

runTime++;

- Aggregation list length
  - MPI_Comm_size * timeIndex

```
client.poll_list_length("displacementsDatasetList",
                        time_index * num_mpi_ranks,
                        10, 1000);
```

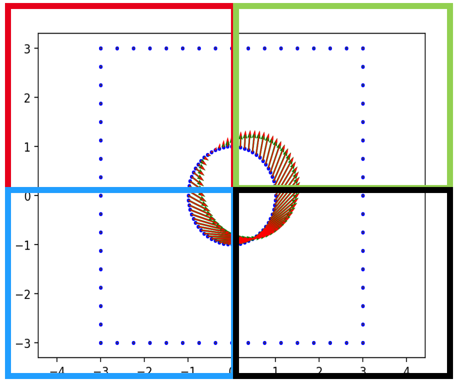
- Is there anything else?

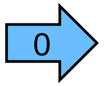
TECHNISCHE UNIVERSITÄT DARMSTADT

SYNCHRONISE TIME STEPS

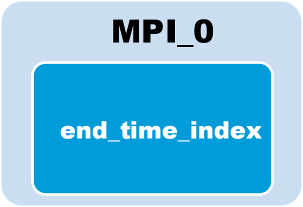
OpenFOAM

runTime++;





redis



SmartSim

runTime++;

- **How does SmartSim Python script know that the OpenFOAM simulation has ended?**

```
if client.poll_key("end_time_index", 10, 100):
        print ("End time reached.")
        break
```

TECHNISCHE UNIVERSITÄT DARMSTADT

ONLINE ML MESH MOTION

- **Let's look at some source code.**

TECHNISCHE UNIVERSITÄT DARMSTADT

OPENFOAM+SMARTSIM MODULE

- OpenFOAM+SmartSim Module is in development.
- Minimal Working Examples (MWEs):
  - Pre-processing utility.
  - Solver.
  - Function object.
  - fvOption.
- Complex example: mesh motion solver.
- Any suggestions?

TECHNISCHE UNIVERSITÄT DARMSTADT

JOIN US!

OpenFOAM SMARTSIM

https://wiki.openfoam.com/Data_Driven_Modelling_Special_Interest_Group

Give the repositories some love (stars) :)

https://github.com/OFDataCommittee/OFMLHackathon

https://github.com/CrayLabs/SmartSim

https://github.com/CrayLabs/SmartRedis