LUME Ecosystem Updates: Standardized Virtual Accelerators and Digital Twin Infrastructure

Sara Miskovich (SLAC)

July 16, 2026

smiskov@slac.stanford.edu, rroussel@slac.stanford.edu

Co-authors: R. Roussel, G. Bhardwaj, J. Lorelli, A. Edelen

SLAC NATIONAL ACCELERATOR LABORATORY

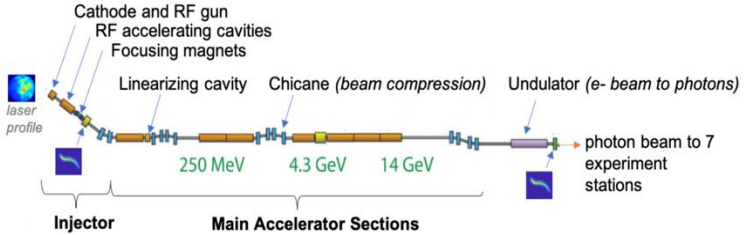
Stanford University

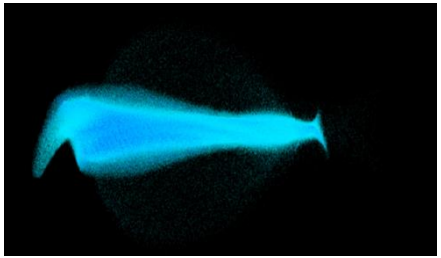
U.S. DEPARTMENT of ENERGY

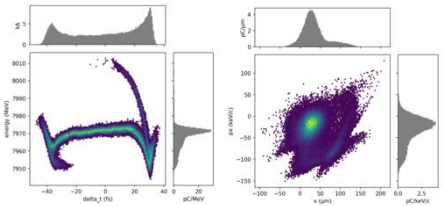
Overview

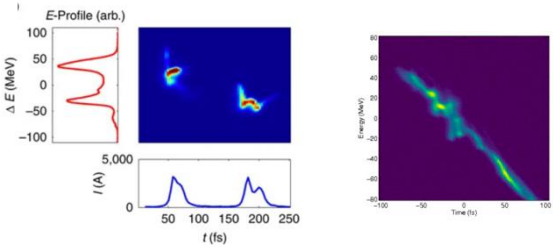
We Need Detailed & Robust System Models

Many tuning problems require detailed beam phase space customization for different experiments





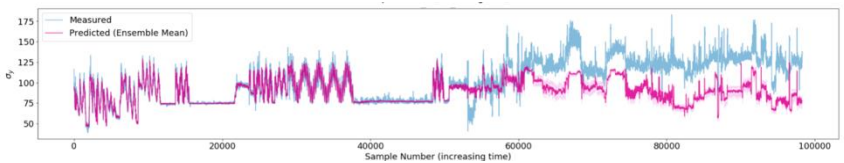




A. Marinelli, et al., Nat. Commun. 6, 6369 (2015)

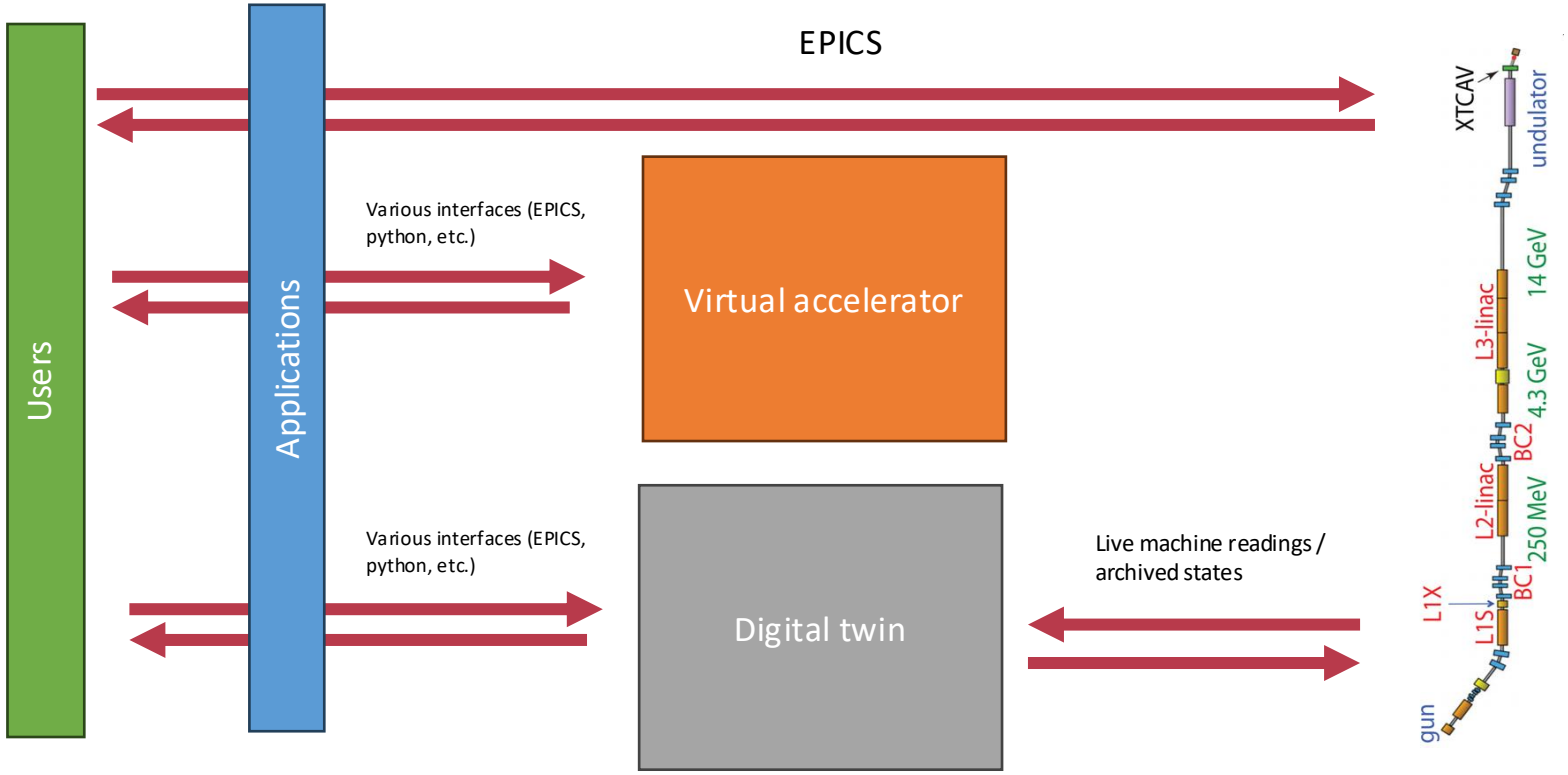
A. Marinelli, IPAC'18

- Nonlinear, high-dimensional optimization/control problem with incomplete information
- Digital twins and virtual accelerators can provide insight
- Infrastructure is a challenge
- Reliable model adaptation methods are key for putting online models to use operationally

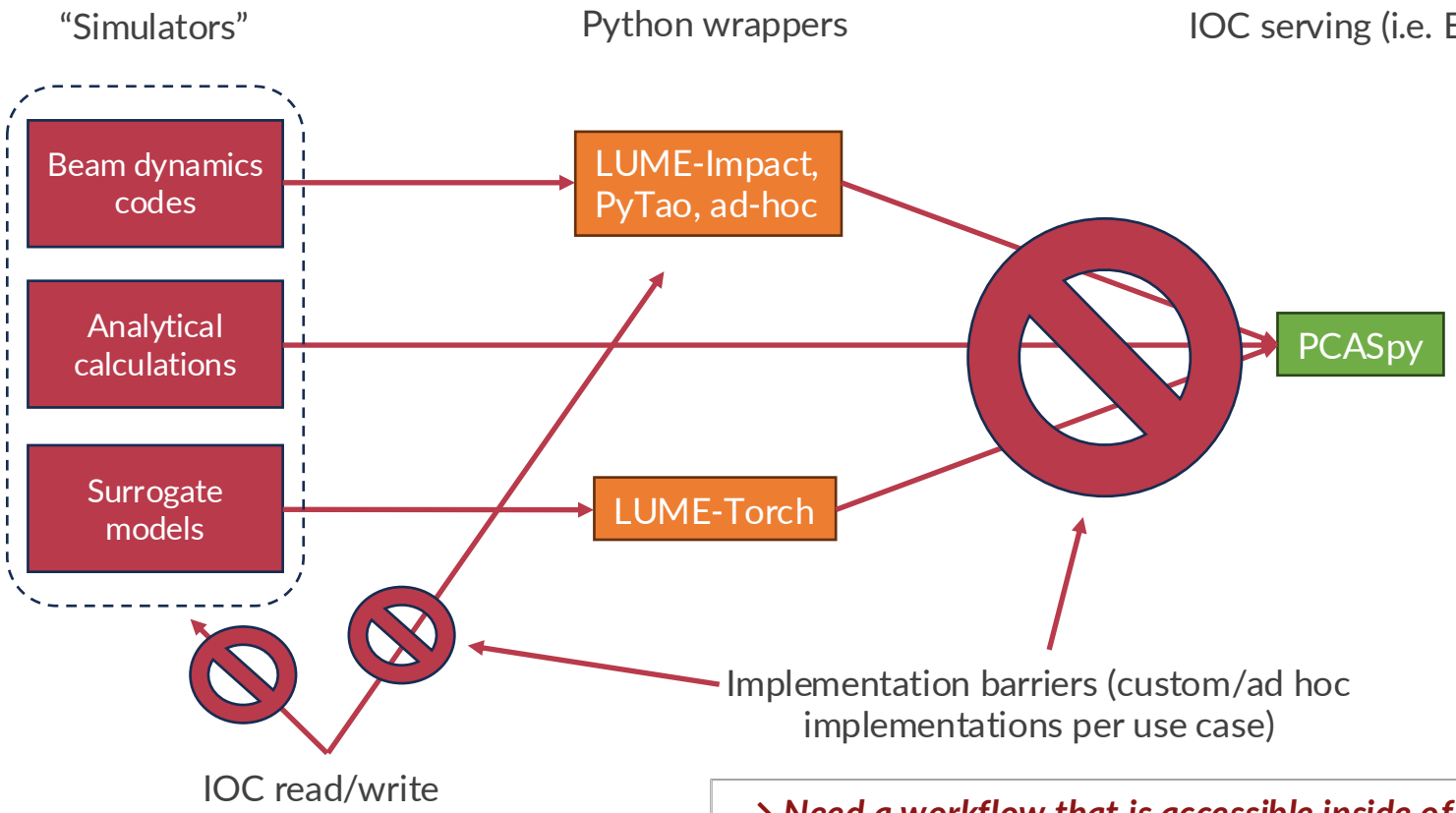


Example of statistical distribution shift over time in a small test accelerator highlights the need for adaptive tuning/continuous online learning and more generalizable modeling methods.

Nomenclature



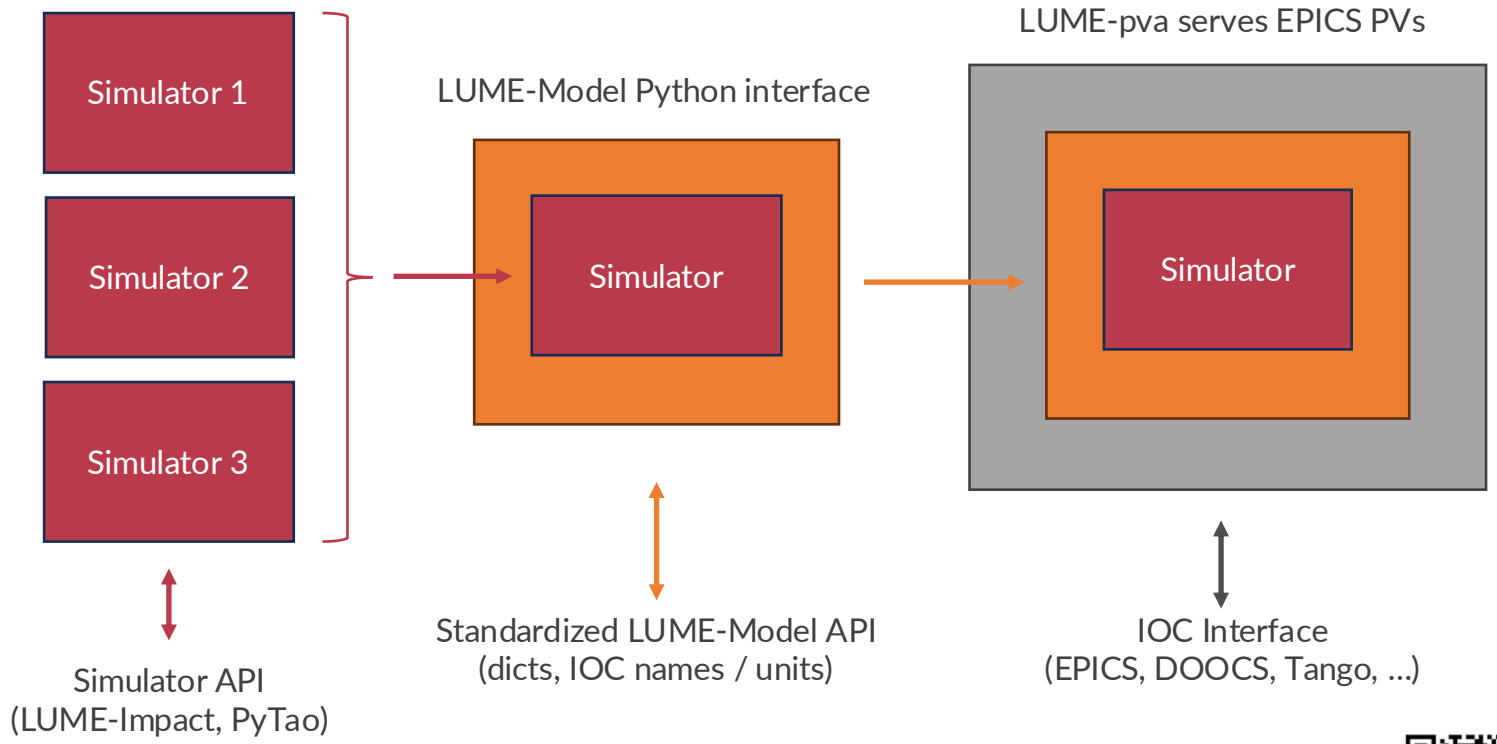
Combining and Deploying Models is Challenging



→ Need a workflow that is accessible inside of optimization software and online controls

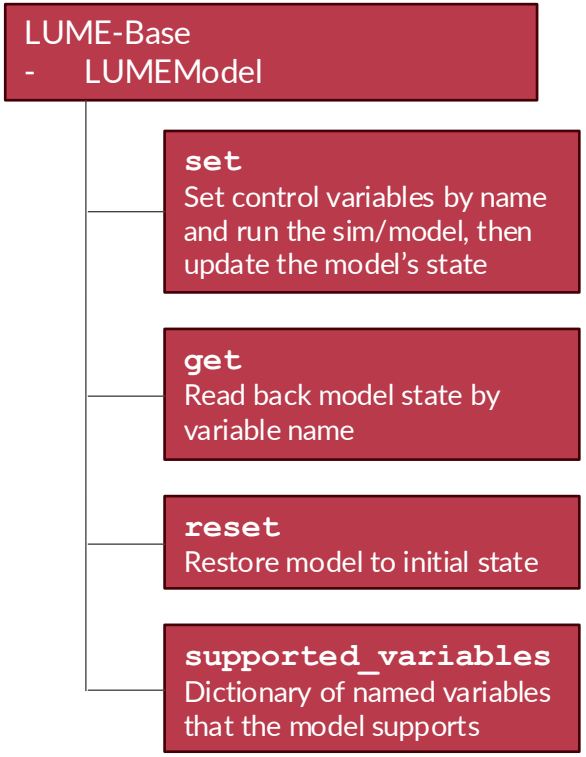
LUME for Virtual Accelerator and Digital Twin Implementations

A Unified and Standardized API with LUME



LUME Tech Paper
arxiv.org/html/2606.07250v1

Standard Interface for VA models and Digital Twins

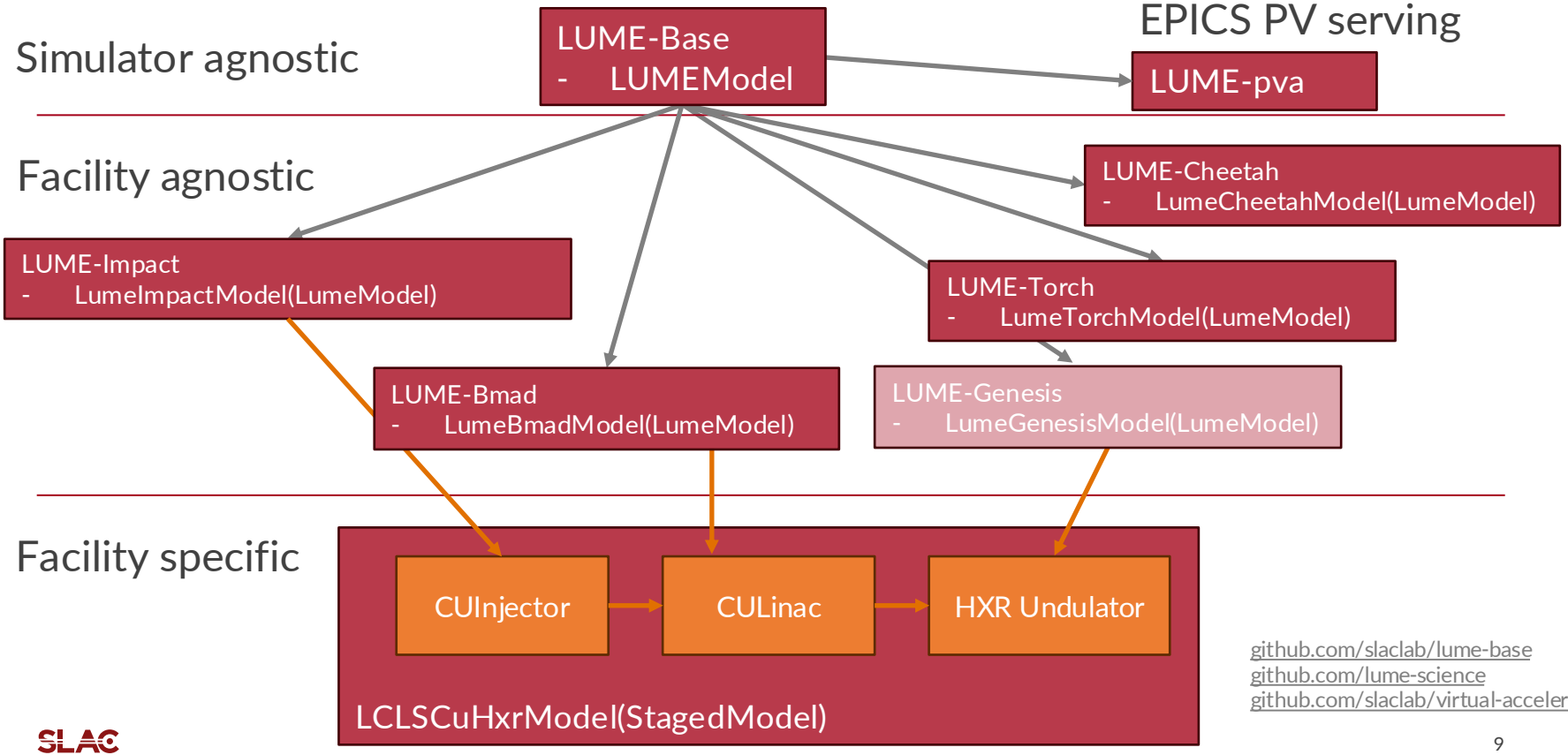


any simulator can implement this contract

Utilized by IOC server (e.g. LUME-pva) to create PVs – can be extended to other control system types

github.com/slaclab/lume-base/blob/main/docs/examples/lume_model.ipynb

LUME Package Structure



github.com/slaclab/lume-base
github.com/lume-science
github.com/slaclab/virtual-accelerator

Staged Model Example: ML Surrogate with Physics Simulation

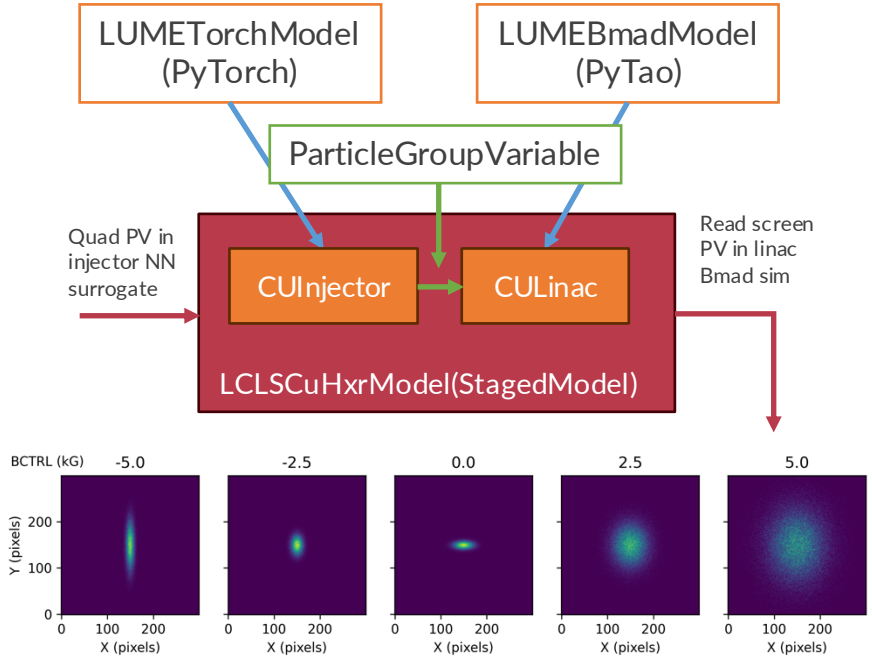
```
injector_surrogate = InjectorSurrogate()
cu_hxr_bmad_model = get_cu_hxr_bmad_model()

staged_model = StagedModel([injector_surrogate, cu_hxr_bmad_model])
```

Set PV in injector NN surrogate

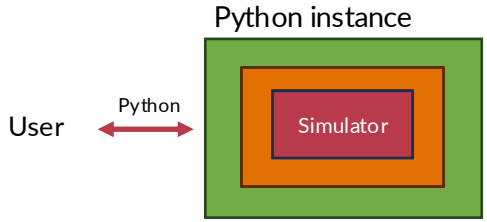
```
staged_model.set({SCAN_QUAD_PV: quad_value})
image = staged_model.get(OTR_IMAGE_PV)
```

Read screen PV in linac Bmad sim

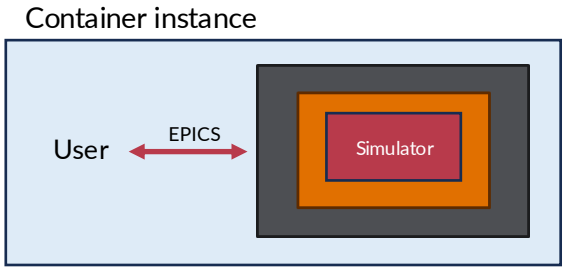


Virtual Accelerator and Digital Twin Implementations

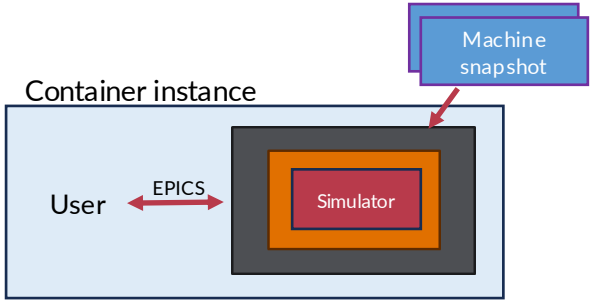
Virtual accelerator in Python



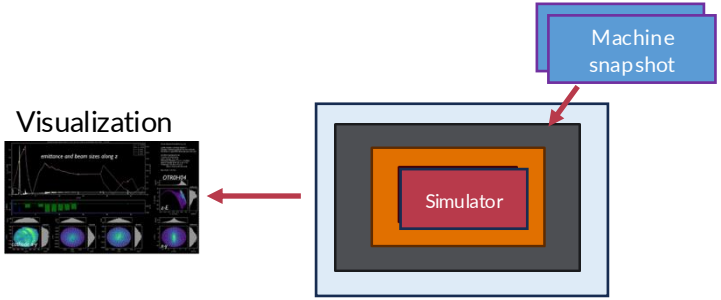
Virtual accelerator in container



Digital twin in container



Digital twin in global server



Badger Environment Testing / Continuous Integration

Use VA to perform nightly testing / validation of code used to interact with the machine

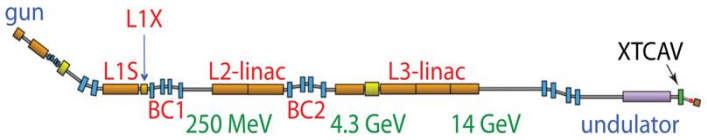
Badger GUI

LCLS Badger Environment class

Class instance

EPICS

not available for software testing



LCLS Machine

Nightly testing workflow

EPICS

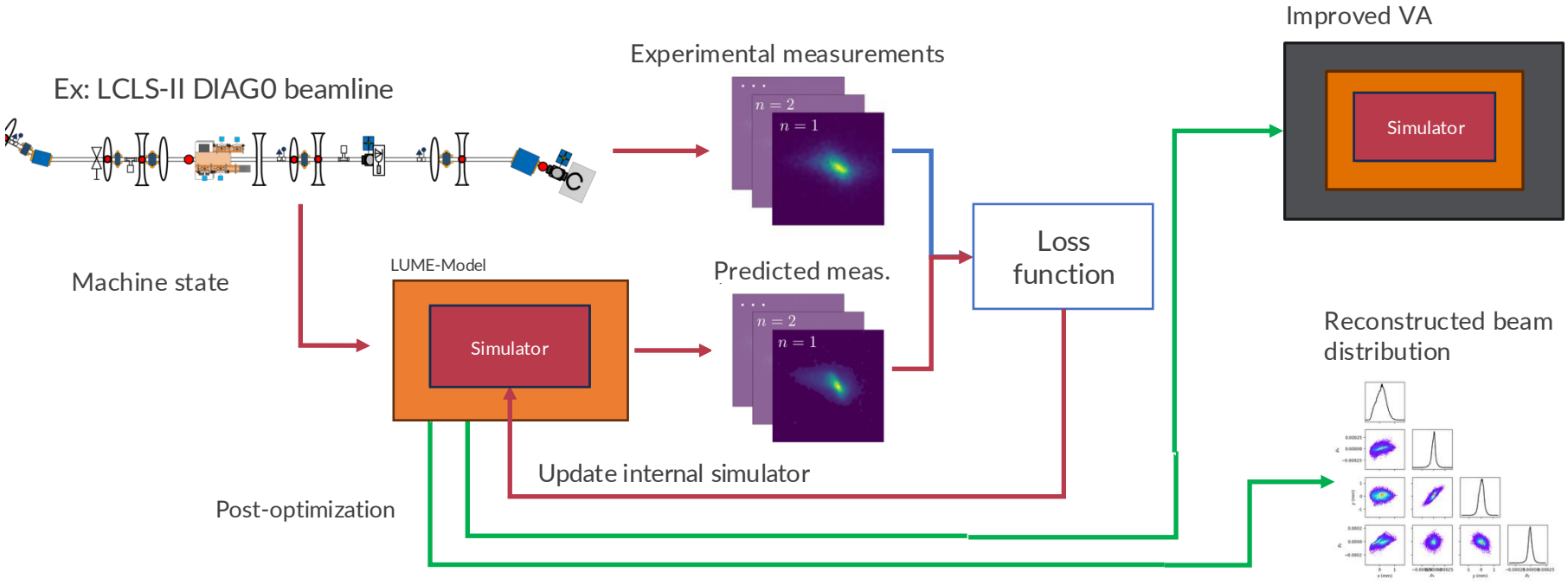
Simulator

LCLS Virtual Accelerator

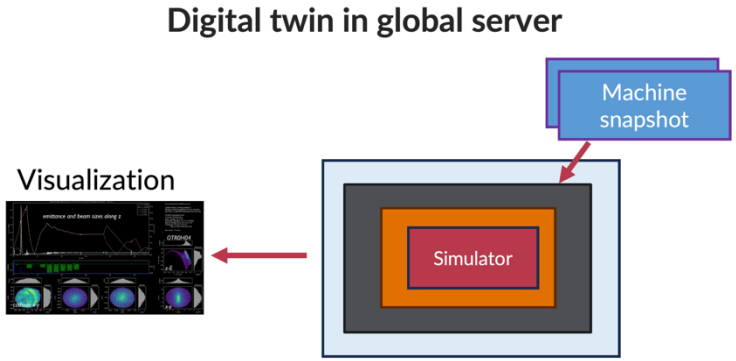
Testing report

VA for Model Calibration / Phase Space Measurement

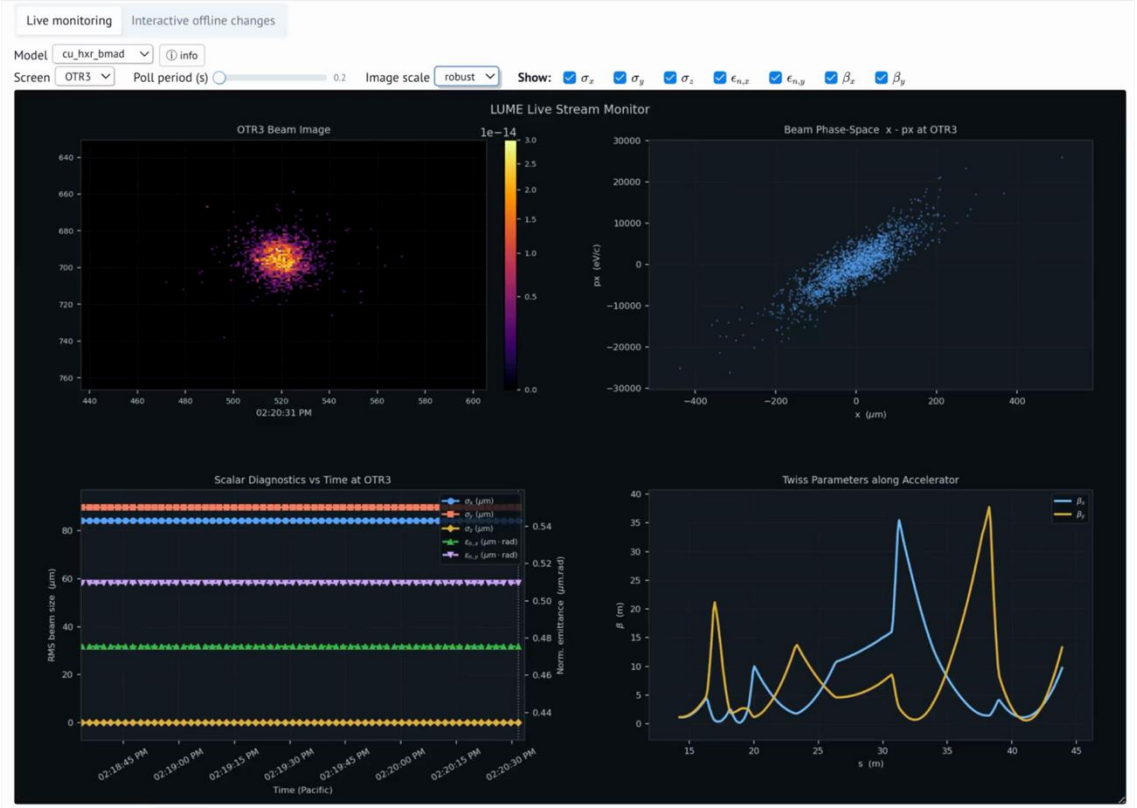
Black-box (Impact / Bmad) and differentiable simulators (Cheetah, surrogates) can be used inside VAs to perform model calibration, phase space reconstruction



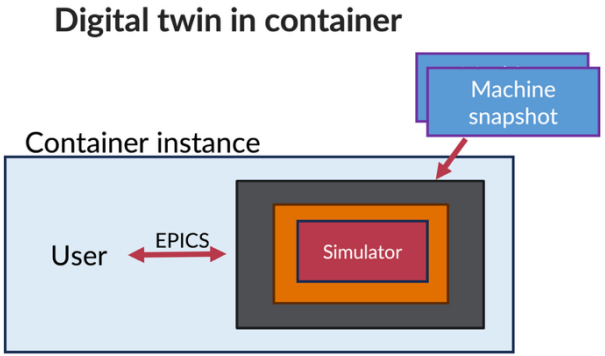
Digital Twin Live Deployment



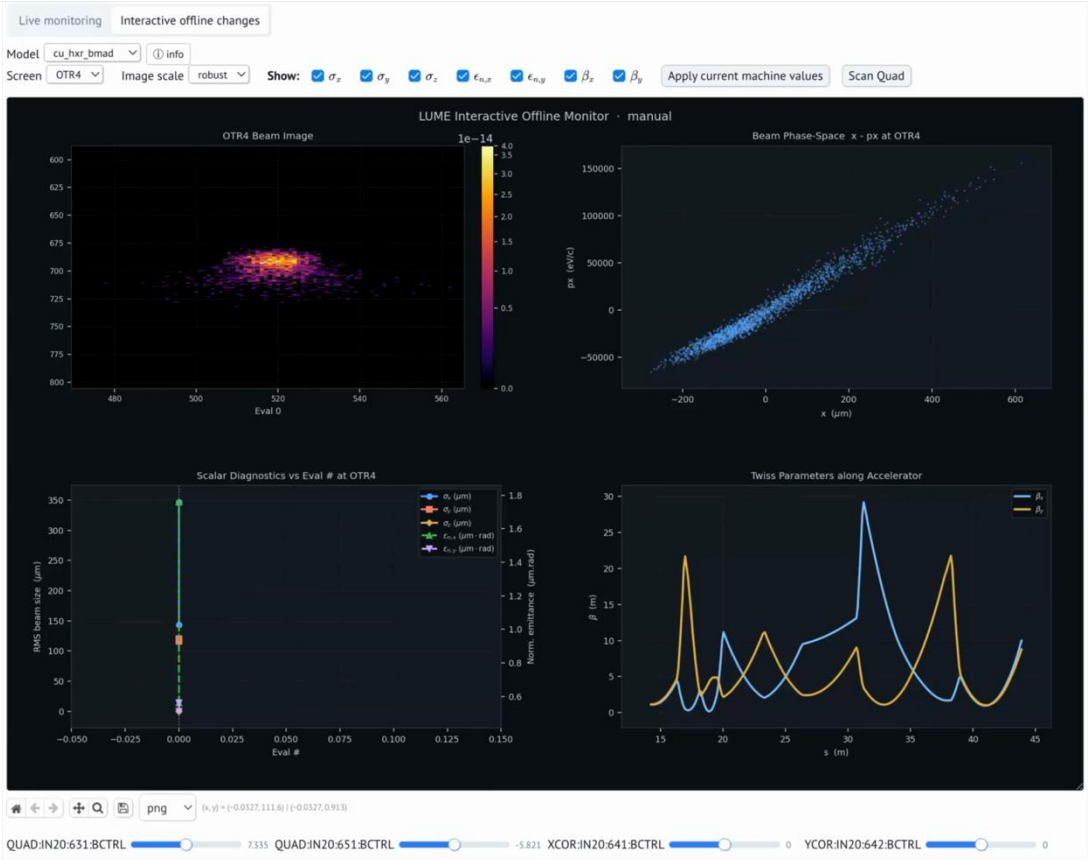
Continuously updating LUME-Model state from live PV snapshots



Interactive VA Mode from Live State

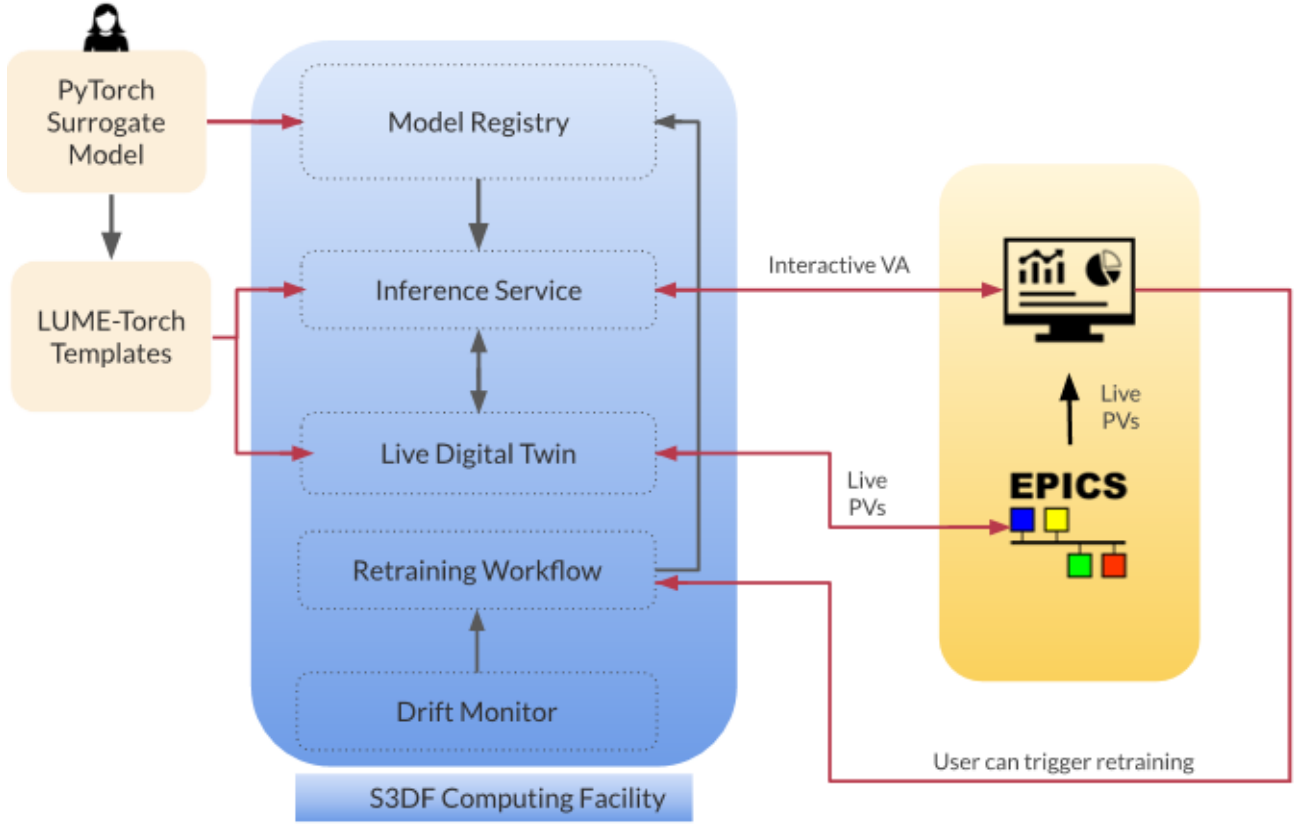


User can play around with LUME-Model PV inputs, starting from machine state



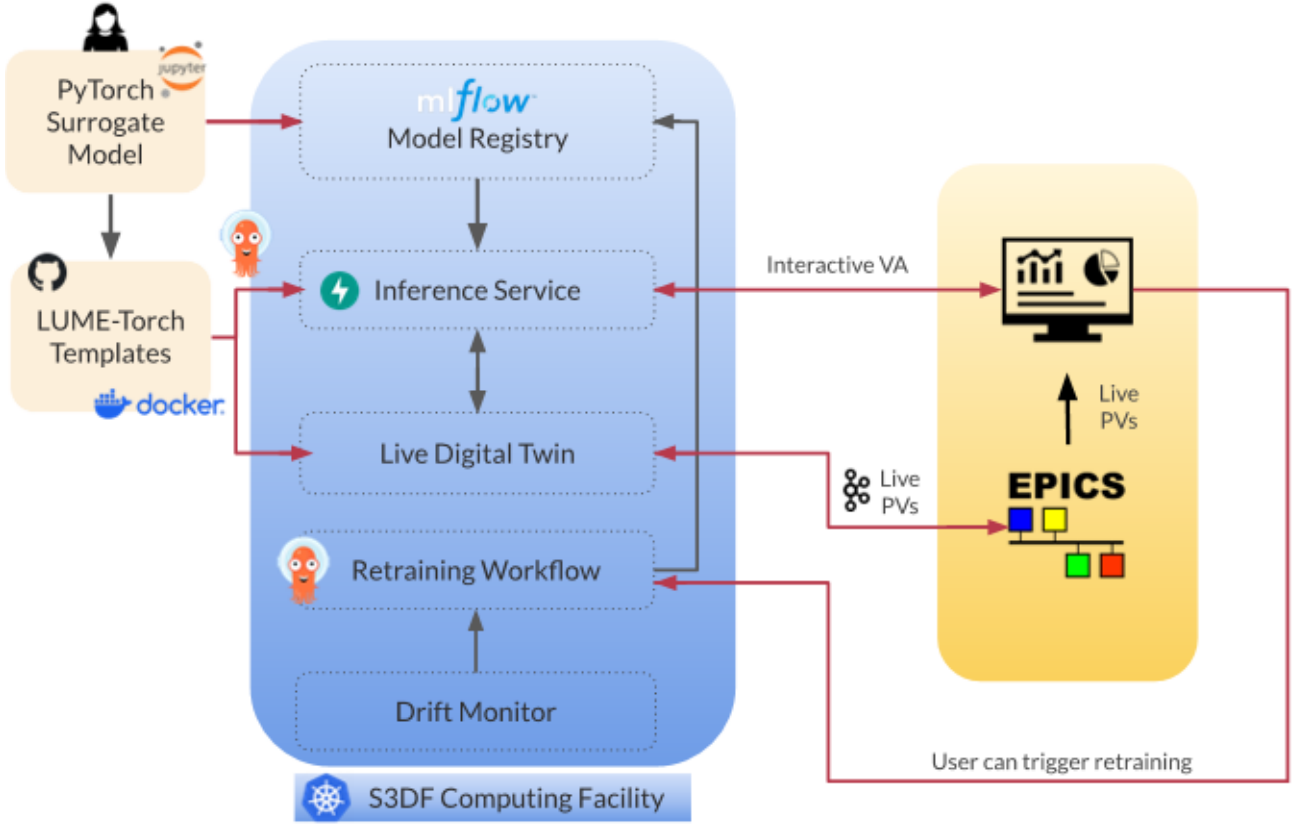
Surrogate Model Deployment and Lifecycle

Digital Twin Deployment of Surrogate Models



github.com/slaclab/inference-service
github.com/slaclab/lume-model-deployment-template

Digital Twin Deployment of Surrogate Models



docker
Runs our software anywhere

Templating, deployment specs, container builds and container registry

Argo tools to run workflows and do GitOps

kubernetes
Keeps our software/containers running

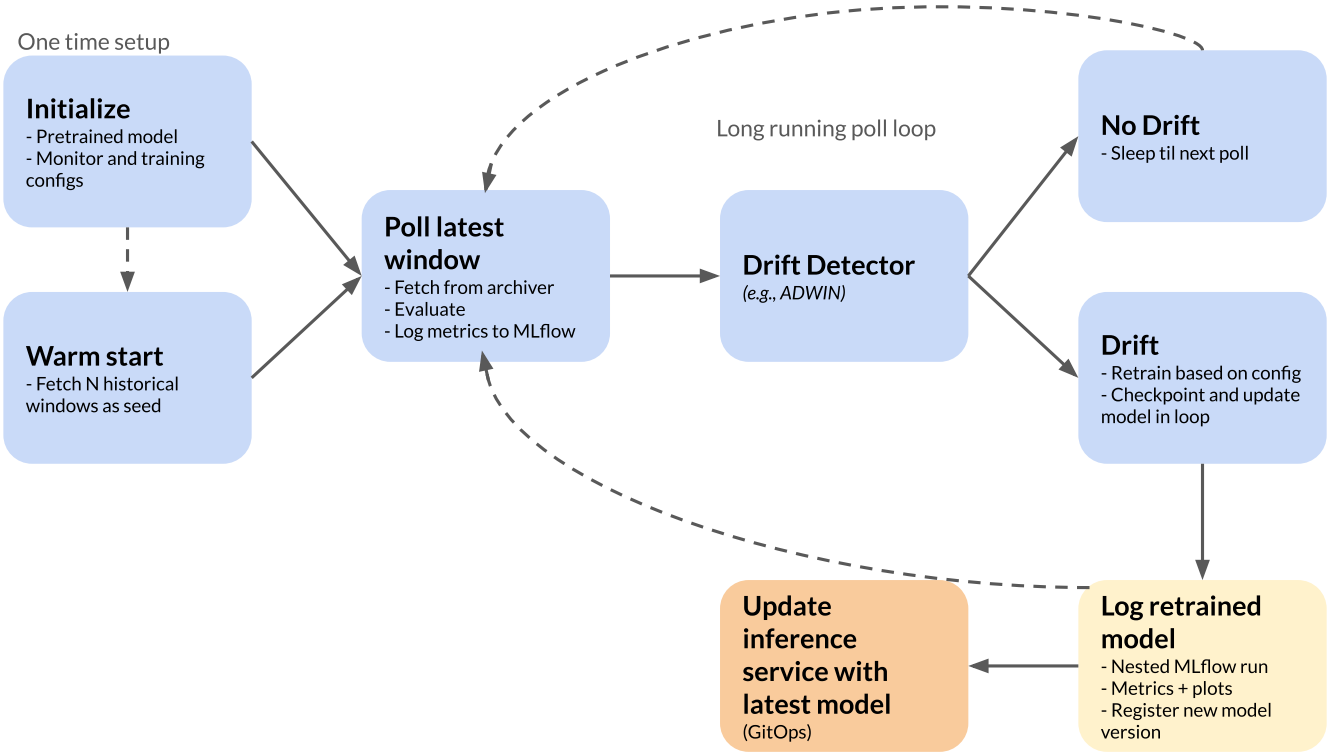
mlflow
Model versioning, artifact storing, tracking training metrics

FastAPI
REST API industry standard for serving ML models

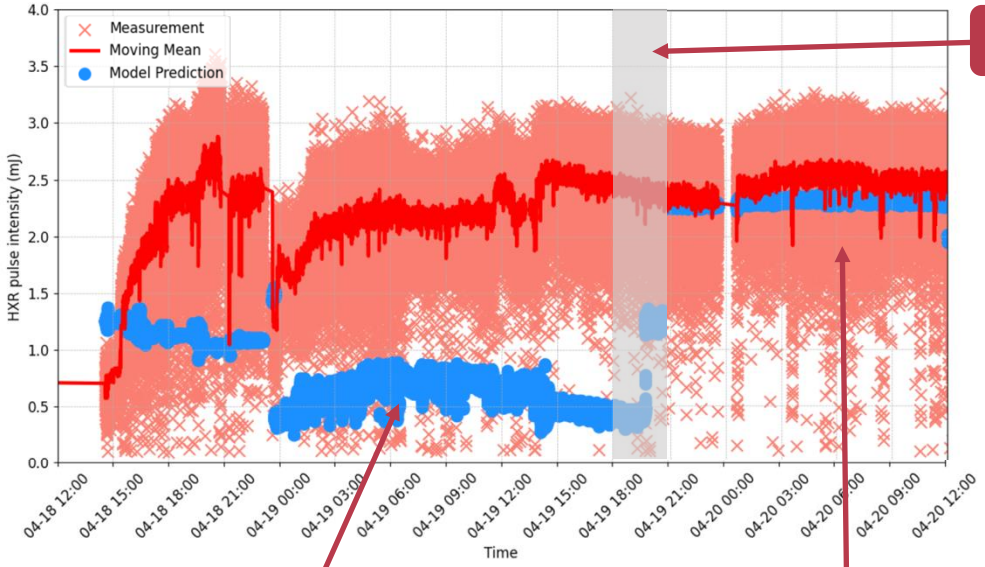
K2eg: Integration between EPICS and data processing pipelines using Apache Kafka
github.com/slaclab/k2eg-python

github.com/slaclab/inference-service
github.com/slaclab/lume-model-deployment-template

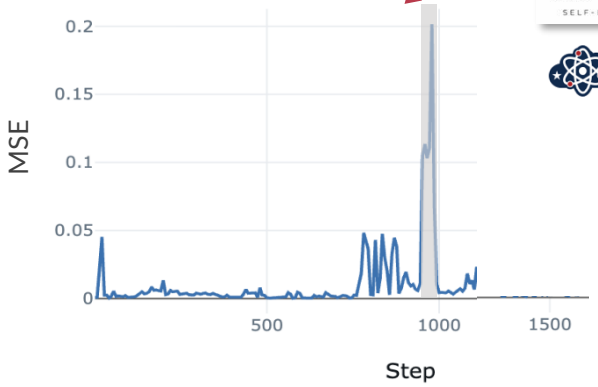
Continual Learning: Drift Monitor Implementation



Continual Learning Demo for FEL Surrogate on Historical Data



Drift Detected



All drift monitoring and training metrics are logged to AmSC MLflow

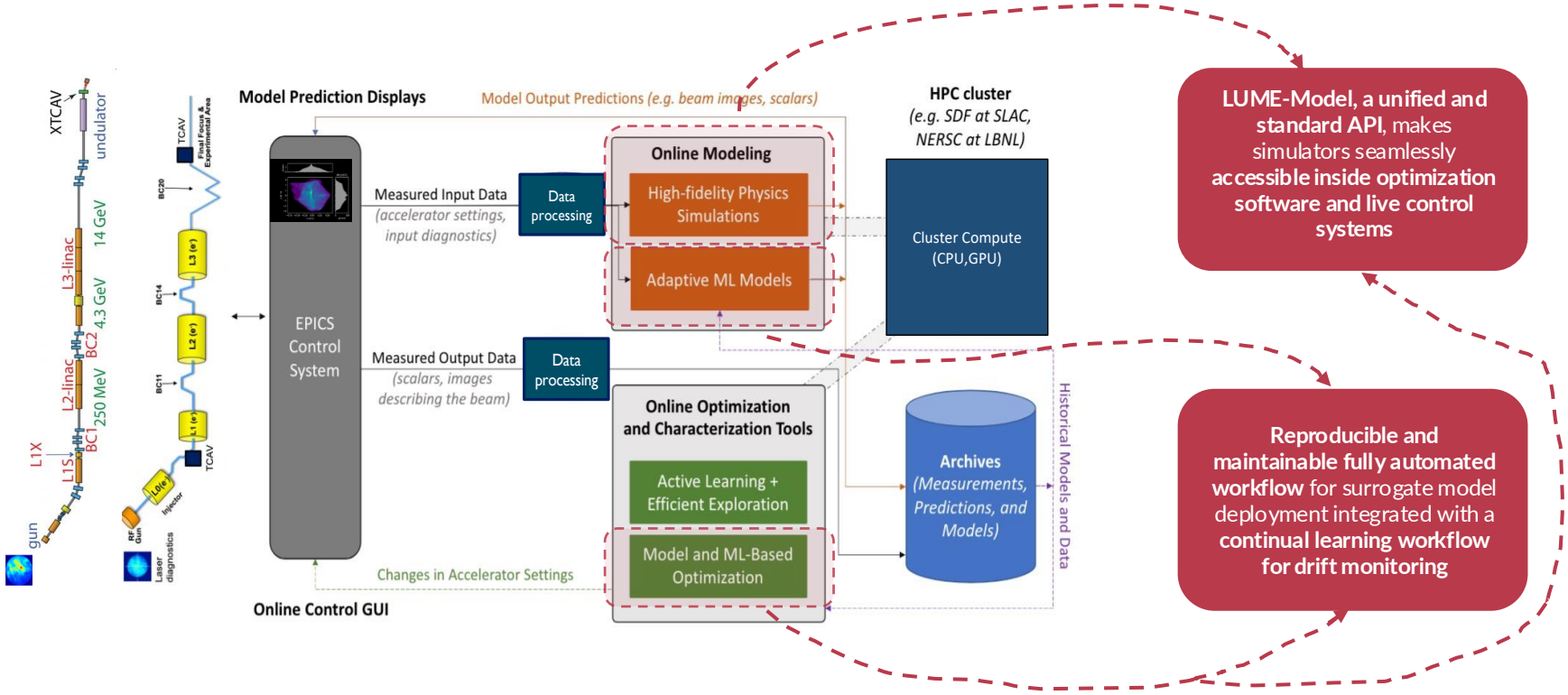
Making progress on improving FEL model accuracy alongside CL workflow development

APEIRON keeps the model synced with data during regime changes

Demo code: github.com/AI-ModCon/BaseSIM_APEIRON/tree/slac-fel

Summary and Next Steps

Overcoming Challenges to Enable DT & VA Integrations at Particle Accelerators



Advanced modeling/control R&D goals and operational deployment vision coming together

Next Steps

Short Term

- Continuing to improve LCLS virtual accelerators and deploying on-demand VA service at SLAC
- Expanding LCLS staged model to include FEL surrogate model after Bmad linac stage
- Expand VA capabilities in terms of the underlying physics sims (variables/physical processes)
- General improvements to accuracy of surrogate models
- Improvements on drift monitoring and training workflows

Long Term

- Calibration/training support in LUME-Torch
- Infrastructure for continual calibration of system models/sim2real workflow
- Operator training/adoption of tools
- Linking system models to Osprey Agentic workflow (github.com/als-apg/osprey)
- Adoption of MLDP (ML Data Platform): enables metadata tagging for easier integration with CL/training workflows
  - github.com/osprey-dcs/data-platform
  - doi: 10.18429/JACoW-ICALEPCS2025-WEPD083

Thank you and resources

Thanks to our amazing colleagues

Gopika Bhardwaj, Ryan Roussel, Ernest Williams, Claudio Bisegni, Jeremy Lorelli, Jingchen Zhou, Auralee Edelen, Chris Garnier, Micha Okun, Hans Thorsten Schwander

And to our collaborators for many useful discussions:

Remi Lehe (LBNL), Revathi Jambunathan (LBNL), and Mat Leputa (ISIS)

Resources

LUME Resources

- Tech paper https://arxiv.org/html/2606.07250v1
- https://github.com/lume-science
- https://github.com/slaclab/lume-base

SLAC VA Implementations

- https://github.com/slaclab/virtual-accelerator

Surrogate Templates and Inference Repos

- https://github.com/slaclab/lume-model-deployment-template
- https://github.com/slaclab/inference-service

Continual Learning in collaboration with Modcon

- https://github.com/AI-ModCon/BaseSIM_APEIRON/tree/slac-fel