HiPAS GridLAB-D

TAC Meeting #6 (April 2020)

EPC 17-043 GLOW

EPC 17-046 HiPAS

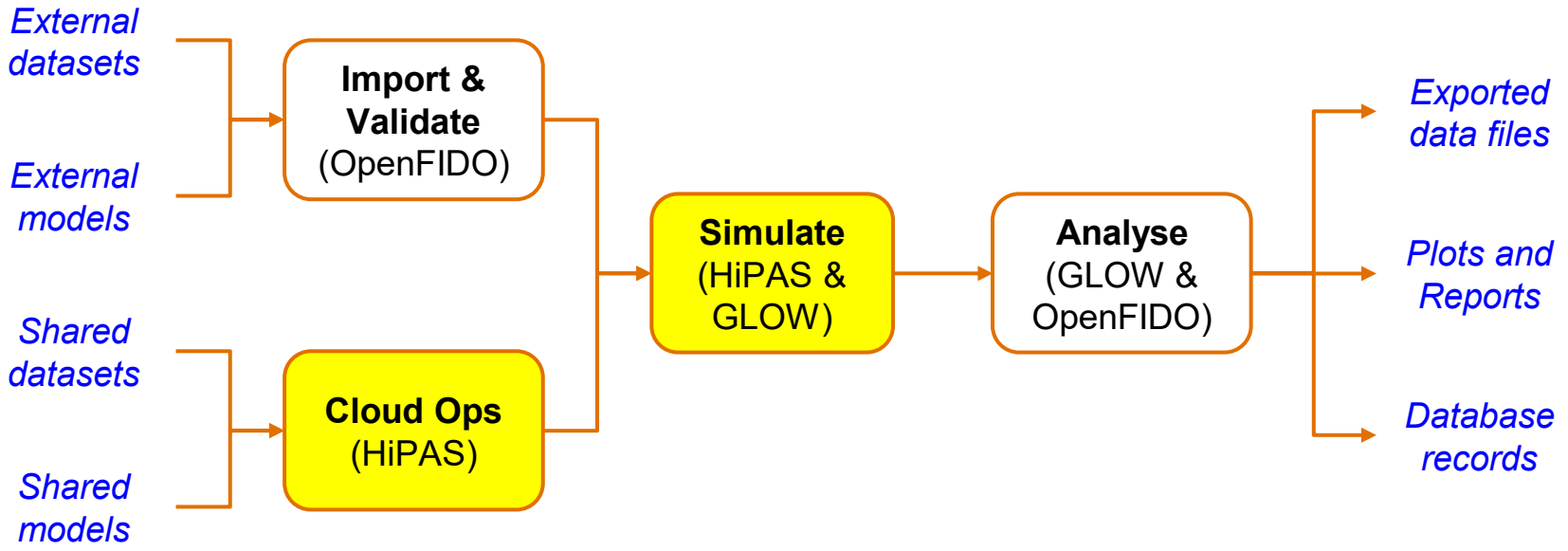
EPC 17-047 OpenFIDO

This presentation was prepared with funding from the California Energy Commission under grant EPC-17-046. SLAC National Accelerator Laboratory is operated for the US Department of Energy by Stanford University under Contract No. DE-AC02-76SF00515

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Project Focus Area



Project Objectives

Enhance GridLAB-D to support leading California use-cases

1. Integration Capacity Analysis - Support GLOW (in progress)
2. Resilience - GRIP analytics (substantially complete)
3. Tariff design - Powernet With Markets and revenue analysis (in progress)
4. Electrification - Enable decarbonization simulations (in progress)

HiPAS Enhancements Completed

SLAC

High-performance simulation

- Machine learning powerflow in Python performance analysis
- Fast initialization, large scale AWS operations, parallel job control

Improved data/model processing tools

- Basic file input/output (“any data, any format, anywhere”)
- New database module (e.g., InfluxDB, Amazon RDS)

New simulation modules/classes/templates

- Industrial loads (NERC-supported NAICS facilities)
- Residential loads (RBSA)
- Commercial loads (CEUS)
- Revenue module (existing PG&E tariffs and billing classes)

HiPAS Enhancements in Development

SLAC

High-performance simulation

- Machine learning powerflow performance optimization

Improved data/model processing tools

- Advanced file input/output (“any model, anytime, anywhere”)
- OpenFIDO integration API (data)

New simulation modules/classes/templates

- Residential and commercial loads (physics-based multi-family residences)
- ICA and GRIP analysis templates
- Revenue module (existing SCE and SDG&E tariffs, tariff analysis)

HiPAS Enhancements Coming Next Year

SLAC

High-performance simulation

- Granular parallelization for all events (including python events)
- Remote job control, GCP/Azure operations

Improved data processing tools

- OpenFIDO Integration API (models)
- CYME 8 and 9 compatibility

New simulation modules/classes/templates

- Industrial loads (non-NERC NAICS facilities)
- Residential and commercial loads (census-based/AMI-fit models)
- Tariff design and electrification use-cases

Highlights: Cloud deployment

GitHub deployment

- Source code: https://source.gridlabd.us/
- Integrated online documentation at https://docs.gridlabd.us/

Docker containers maintained/updated automatically

- Docker hub repository located at https://docker.gridlabd.us/

Preparation for commercialization

- Support for transfer to non-SLAC hosted domains and resources

Learning-Accelerated Power Flow for GridLAB-D

Lily Buechler
(ebuech@stanford.edu)

Adithya Antonysamy, Tom Achache, Siobhan Powell, Ram Rajagopal, and David Chassin

Learning-Accelerated Power Flow

SLAC

Power Flow simulation in GridLAB-D

- 3-phase, unbalanced, quasi-steady power flow
- Map power injections to voltages via inverse power flow mapping

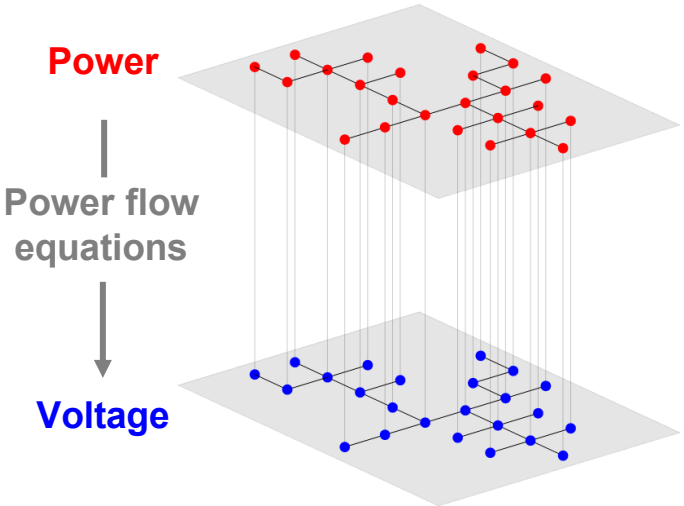
Standard approach

- Newton Raphson (NR) with the current injection method
- Computationally expensive for large networks

Data-driven approach

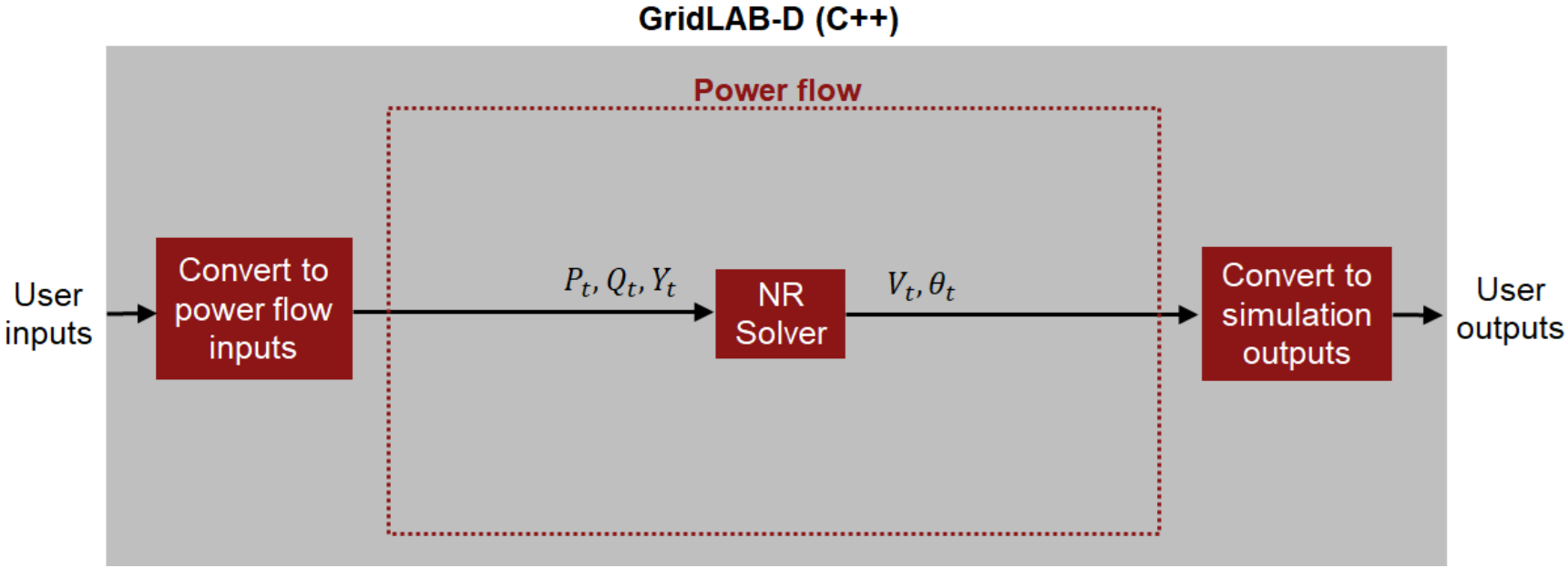
- Adaptively learn power flow mapping during simulation from previous NR solutions

Inverse power flow mapping



Standard Power Flow Solver

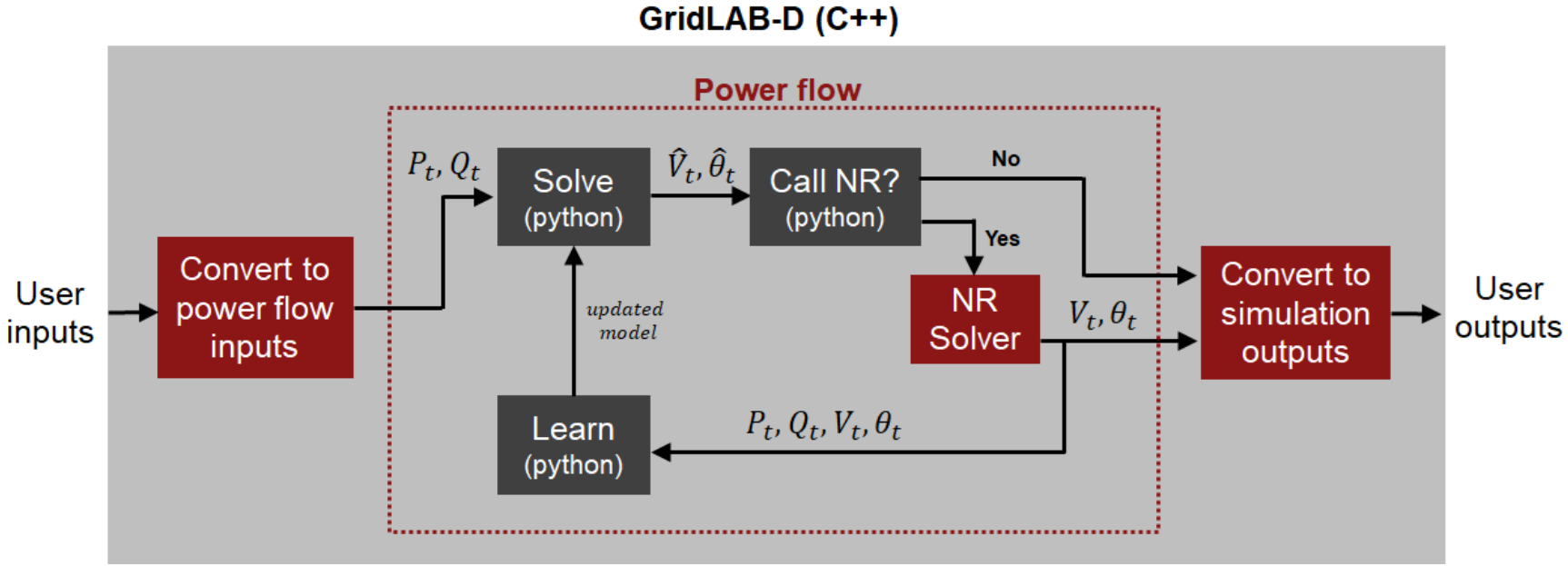
Solve power flow equations with Newton Raphson via the current injection method [1]



[1] Garcia, Paulo AN, et al. "Three-phase power flow calculations using the current injection method." *IEEE Transactions on Power Systems* 15.2 (2000): 508-514.

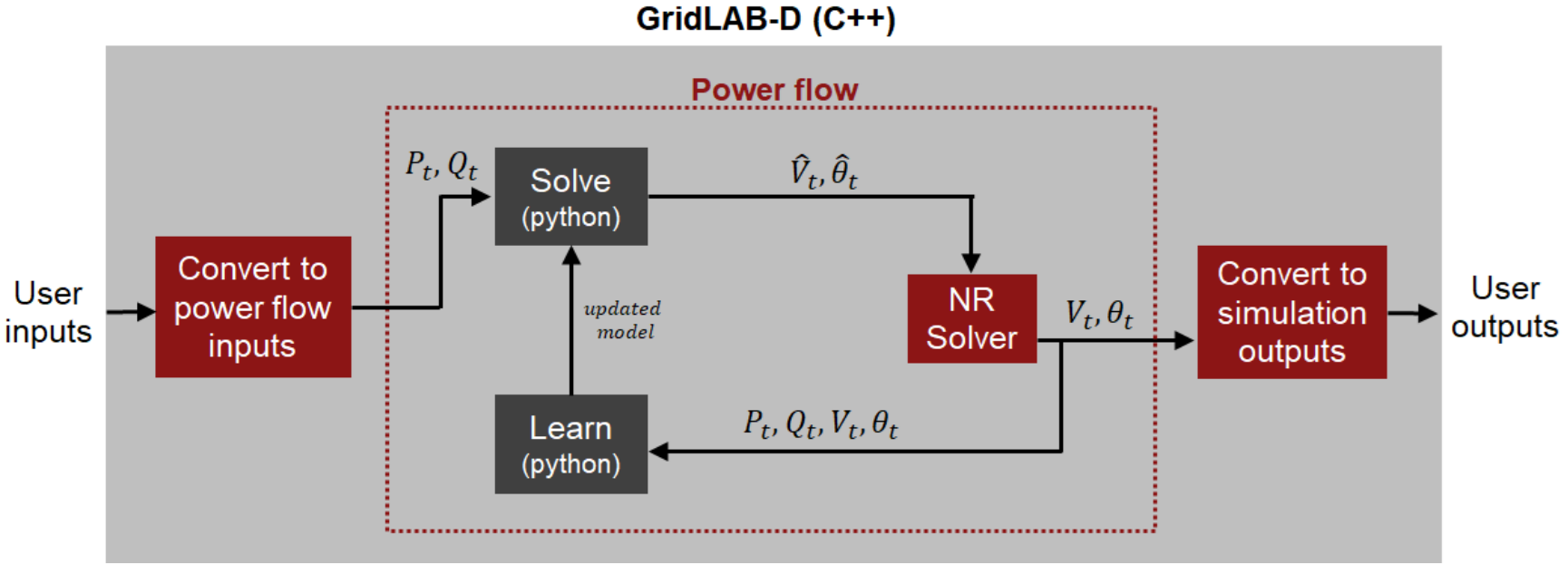
Learning-Accelerated Approach: ML + NR

Adaptively switch between using the Newton Raphson solver and a learned data-driven power flow mapping during simulation



Learning-Accelerated Approach: ML-Seeded NR

Seed Newton Raphson solver with estimate from learned data-driven model

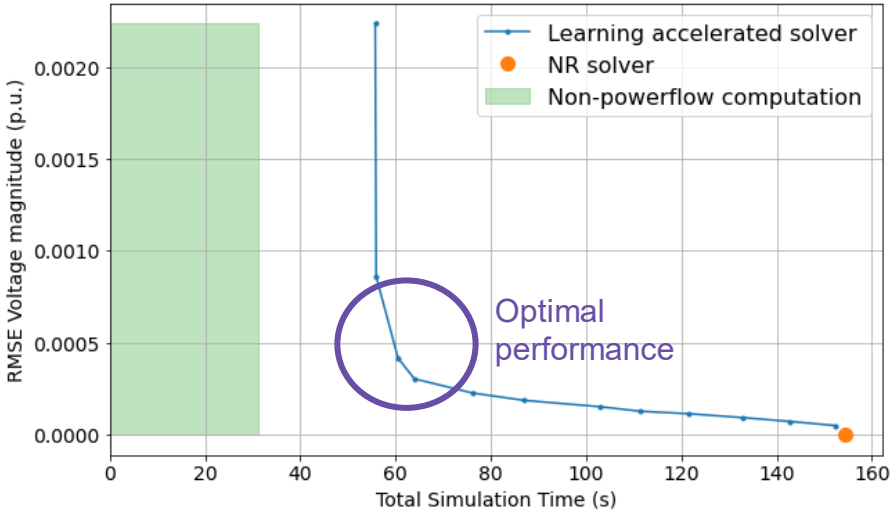


Computational Performance: ML + NR

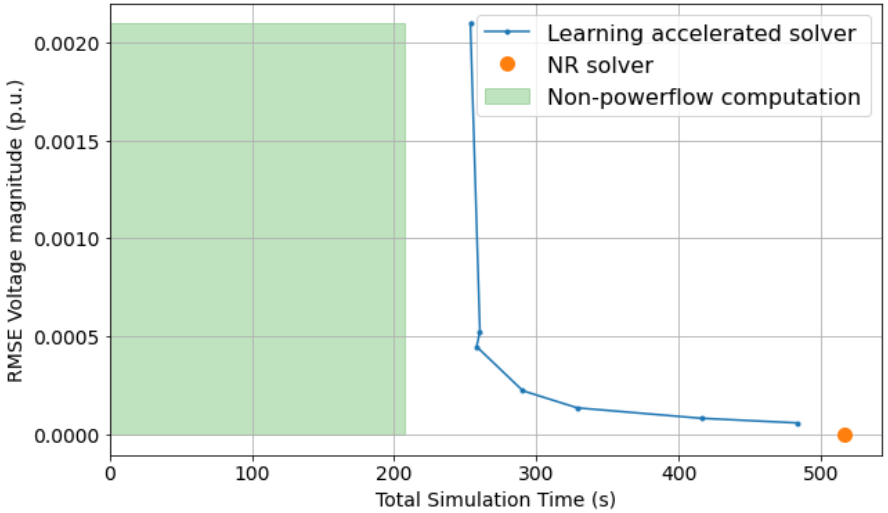
Results from latest implementation (recursive least squares prediction):

- Reduction in total GridLAB-D runtime of 50-65%
- Reduction in power flow computation time of 80-90%
- Trade off between speed and accuracy

IEEE 123 bus



PNNL R2-12.47-2 (851 nodes)



Next Steps

SLAC

Python-based online implementation

- Further optimize computational performance
- Integrate other ML model types
- Analyze other methods for solver selection

C++ based online implementation

- Convert python module to C++ to speed up performance
- Validation on test cases

TAC Questions

1. General feedback and questions?
2. Testing data and models?
3. Testing staff and projects?
4. New or emerging use-cases?

Thank You

Contact: dchassin@slac.stanford.edu