SAND2021-13853C

Integration of Optimization and Machine Learning for Improving Electrical Grid Operation

Prof. Carl D. Laird
Chemical Engineering, Carnegie Mellon University
Center for Advanced Decision Making

Jordan Jalving, Logan Blakely, Michael Eydenberg (Sandia)
Fani Boukouvala, Zachary Kilwein (Georgia Tech)

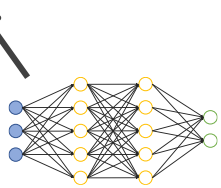
The authors would like to gratefully acknowledge funding from Sandia's Laboratory Directed Research and Development (LDRD) program.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

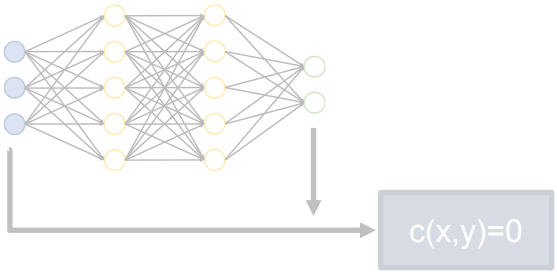
Carnegie Mellon University

Integration of Optimization and Machine Learning

$$\min f(x, y)$$
$$\text{s.t. } c(x, y) = 0$$
$$y = NN(x)$$



- Optimization of problems with embedded neural network models in the formulation
  - Neural network (NN) models as surrogates in optimization for data-driven components, challenging constraints, subproblems [Grimstad & Andersson, 2019; Schweidtmann Huster, Luthje, Mitsos, 2019]
  - Include classifiers within optimization formulations
  - Replace nonlinear constraints with piecewise linear constraints, e.g., ReLU networks represented as MIP [Anderson, Huchette,... Vielma (2020); Tsay, Kronqvist, Thebelt, Misener, (2021)]
  - Many other applications [e.g., Tjeng, Xiao & Tedrake, 2017; Venzke et al. (2020)

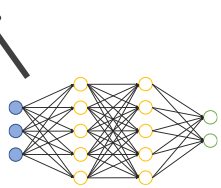


E.g., Mass and Energy Balances
Current and Voltage Laws
Physical bounds

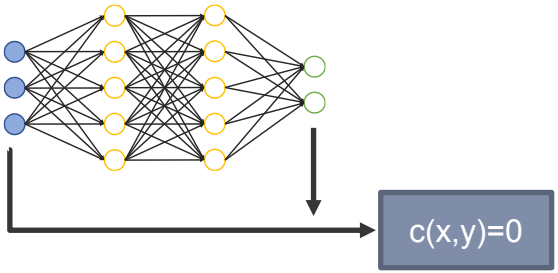
- Training physics-informed neural networks (NN)
  - No guarantee that traditional NN matches known physics
  - Integrate known physics in constrained training approach
  - Improved performance with limited data and on out-of-sample
  - Example Approaches: regularization, constrained optimization, Lagrangian-dual [Fioretto, Mak, Van Hentenryck (2021)]

Integration of Optimization and Machine Learning

$$\min f(x, y)$$
$$\text{s.t. } c(x, y) = 0$$
$$y = NN(x)$$



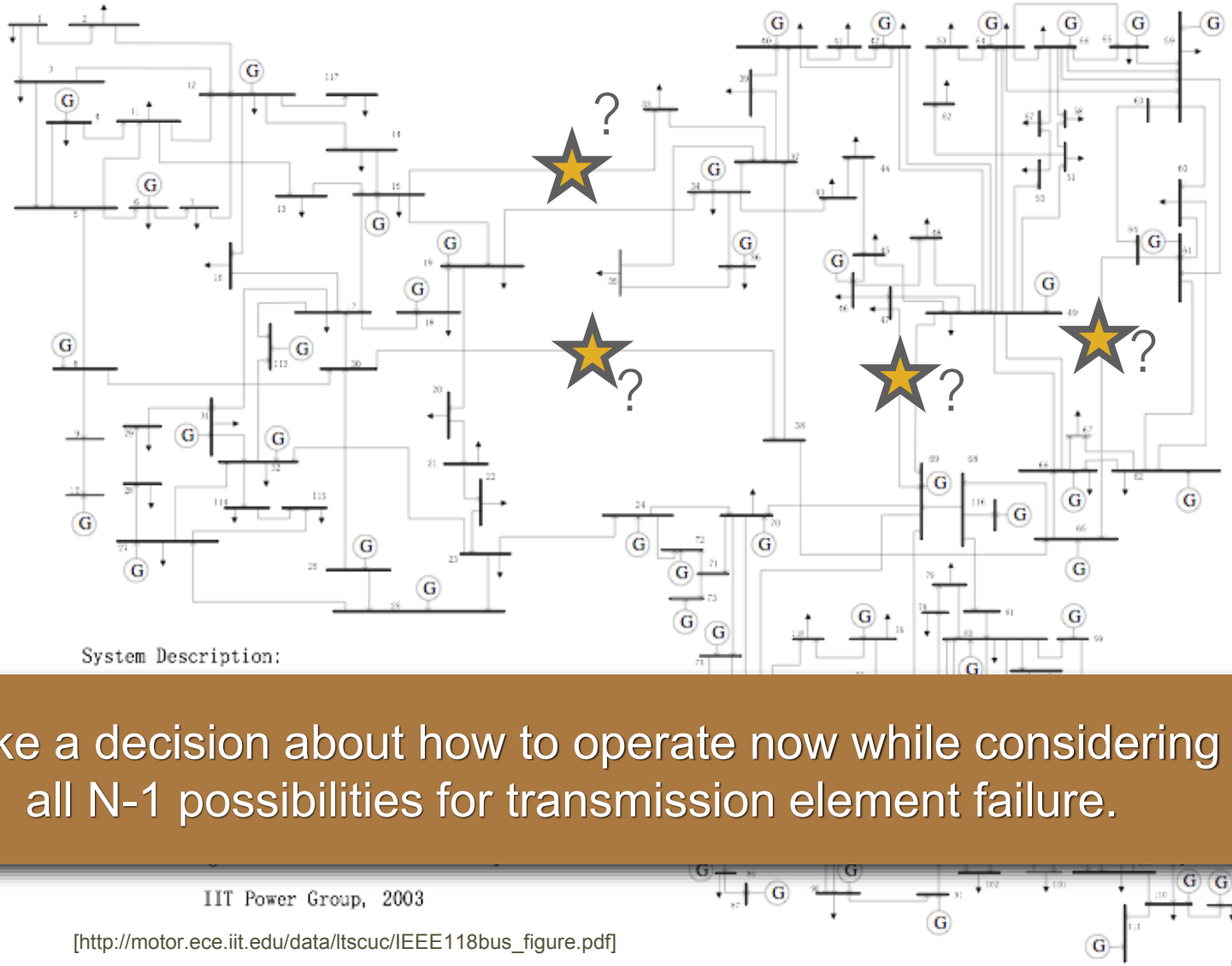
- Optimization of problems with embedded neural network models in the formulation
  - Neural network (NN) models as surrogates in optimization for data-driven components, challenging constraints, subproblems [Grimstad & Andersson, 2019; Schweidtmann Huster, Luthje, Mitsos, 2019]
  - Include classifiers within optimization formulations
  - Replace nonlinear constraints with piecewise linear constraints, e.g., ReLU networks represented as MIP [Anderson, Huchette,... Vielma (2020); Tsay, Kronqvist, Thebelt, Misener, (2021)]
  - Many other applications [e.g., Tjeng, Xiao & Tedrake, 2017; Venzke et al. (2020)]



E.g., Mass and Energy Balances
Current and Voltage Laws
Physical bounds

- Training physics-informed neural networks (NN)
  - No guarantee that traditional NN matches known physics
  - Integrate known physics in constrained training approach
  - Improved performance with limited data and on out-of-sample
  - Example Approaches: regularization, constrained optimization, Lagrangian-dual [Fioretto, Mak, Van Hentenryck (2021)]
  - Global optimization strategies for verification [Venzke et al.]

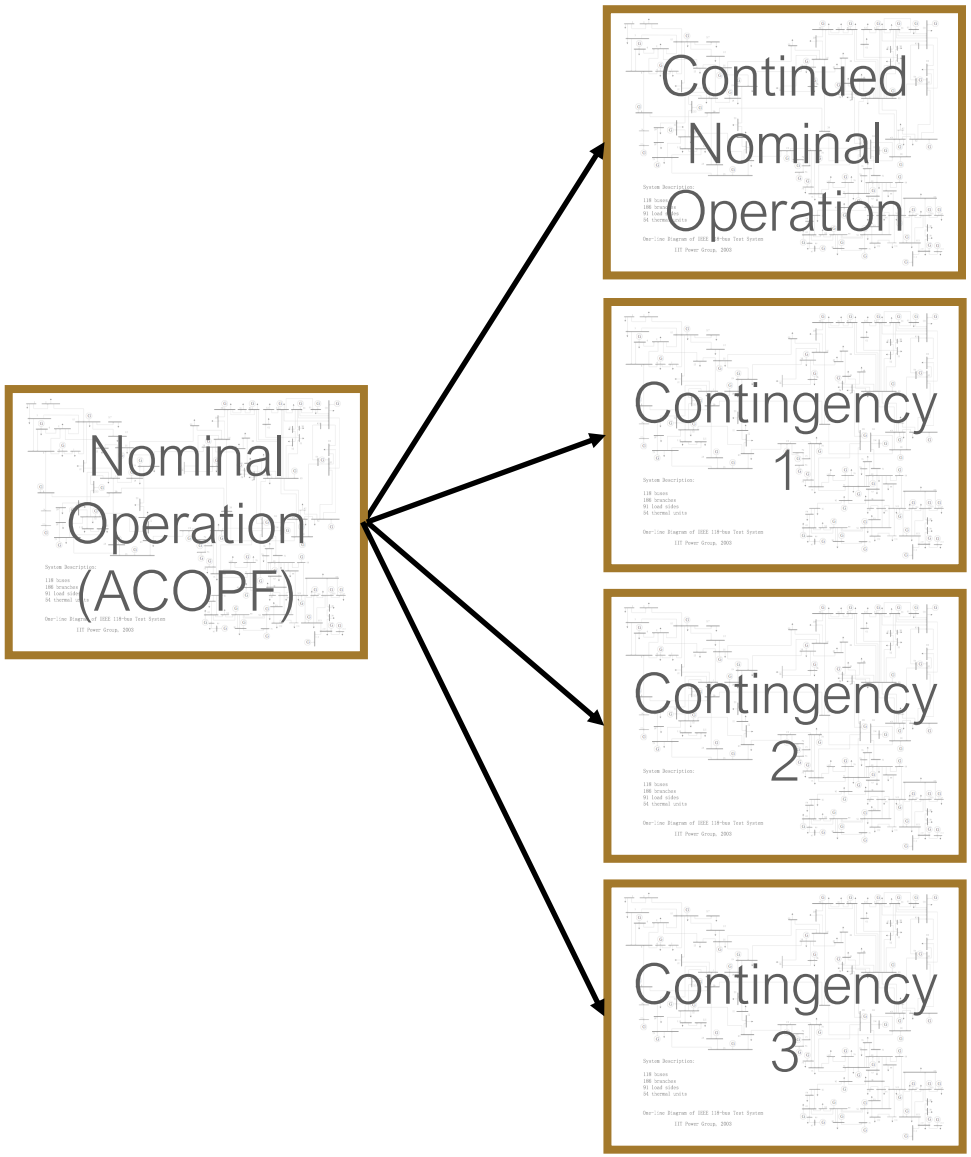
N-1 Contingency-Constrained ACOPF Problem



Make a decision about how to operate now while considering all N-1 possibilities for transmission element failure.

[http://motor.ece.iit.edu/data/ltscuc/IEEE118bus_figure.pdf]

N-1 Contingency-Constrained ACOPF Problem



Nonlinear two-stage stochastic programming problem

AC power flow model for each scenario and stage

Contingency scenario for each (almost) transmission line (1000's)

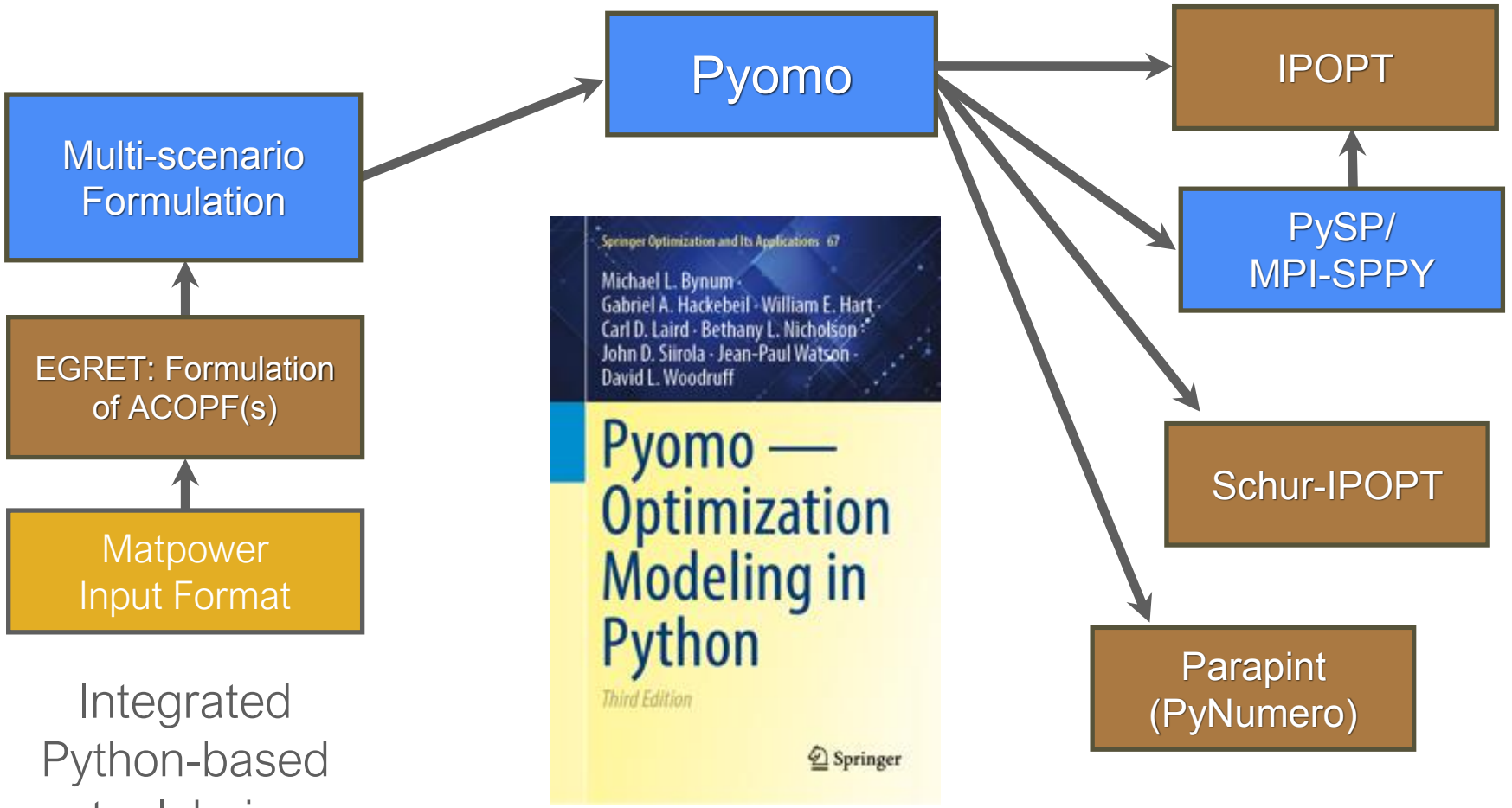
Penalty for inability to meet demands

Ramping constraints for changes in generator set points

Very large-scale nonlinear programming problem

See specific formulations and other approaches as part of the ARPA-E "GO" competition

Building the model with Pyomo and PySP

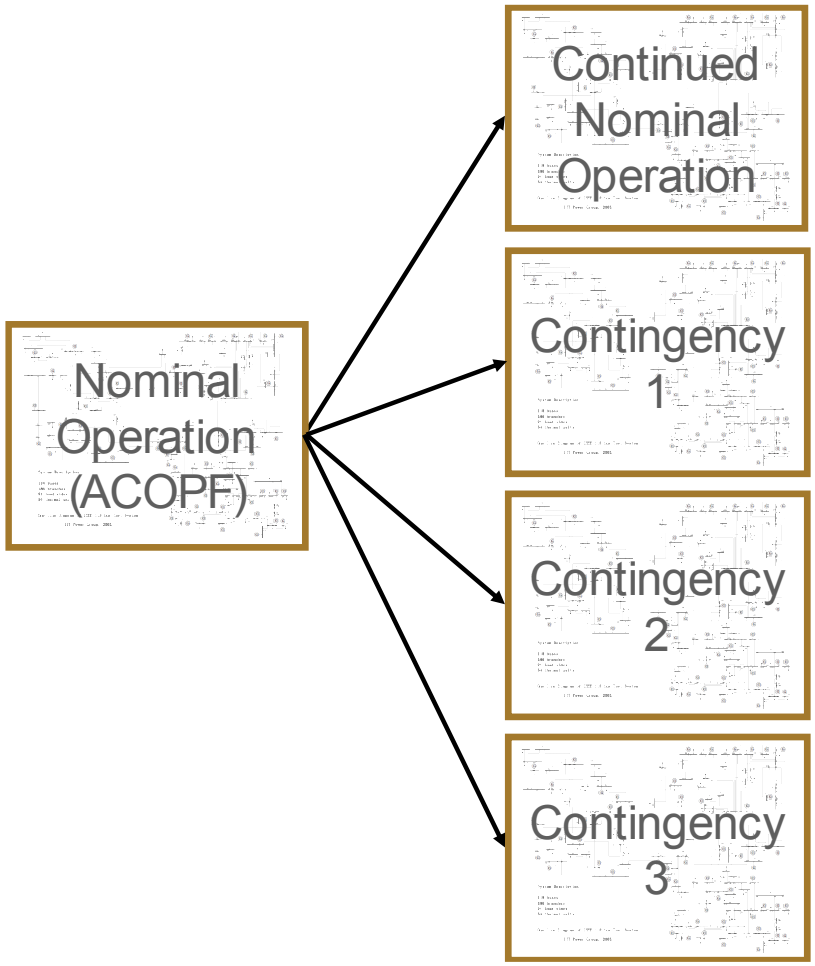


Integrated Python-based toolchain

Hart, Laird, Watson, Woodruff, Hackebeil, Nicholson, Siirola (2nd)
(3rd edition with Bynum at Springer)
Many others contributing to Pyomo

Case 118: ~400,000 vars
~ 400 sec. in serial
~ 30 sec. on HPC

Integration of Optimization with Machine Learning for CCOPF



Seek operation that is cost optimal and N-1 feasible

Multi-scenario problem is intractable for real-time applications on standard (non-HPC) hardware

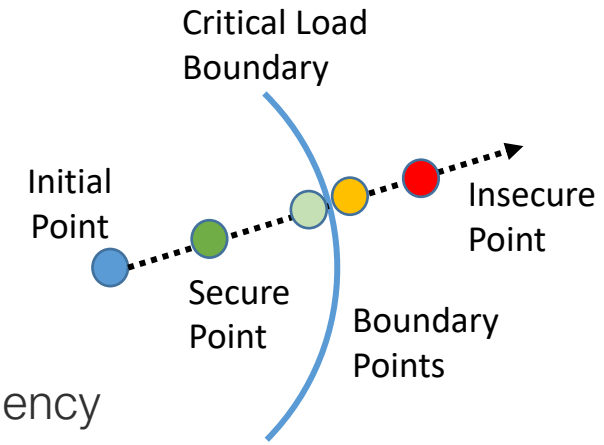
Contingency scenarios serve as constraints on feasibility restricting stage 1 operating point

Can we transfer online cost to offline cost?

- Train a neural network to represent the N-1 feasibility boundary
- Add that neural network as a constraint in the ACOPF problem
- Efficient solution with "nonlinear" NN and local solvers

- Capitanescu et al, 2011: Review of problem space and outstanding challenges
- Gutierrez-Martinez et al, 2011: single-layer network to find security boundary of a given load profile
- Velloso and Van Hentenryck, 2020: Replace the entire problem with NN model - includes physics constraints to improve performance with Lagrangian dual approach
- Venzke et al. 2020: Trained ReLU network for security boundary – tackle MINLP for ACOPF

Approach



Generate Training and Validation Data

- Sample power loads
- Solve ACOPF to get Pg/Qg
- Increase individual loads
- Solve ACPF for each contingency and check feasibility

Train N-1 Feasibility Classifier

- Use TensorFlow to train neural network classifier
- Output indicates feasibility w.r.t. N-1 contingencies

Generate ACOPF + NN Model in Pyomo

- Use EGRET to produce ACOPF model in Pyomo
- Import explicit model of NN into Pyomo for N-1 constraint

Solve "online" with IPOPT

- Check performance compared with stochastic programming formulation

Leveraging OMLT: Open-source package for ML in Pyomo

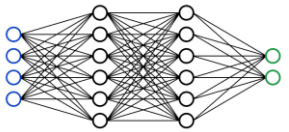
Leveraging code now in the OMLT Python package

- Open-source package developed in collaboration with Imperial College, Carnegie Mellon, and Sandia (OptML)
- Imports neural network models into Pyomo for optimization with trained neural networks (classifiers or regression surrogates)
- Several formulations for neural networks: Full-space, reduced-space, ReLU MIP and Complementarity
- Advanced solution and formulation approaches
  E.g., Tsay, kronqvist, Thebelt, Misener (2021); Kronqvist, Misener, Tsay (2021)
- Support for existing Machine Learning Tools: E.g., Keras (TensorFlow)
- Ongoing development for ONNX, Gradient Boosted Trees, Convolutional Neural Networks
- Initial release on GitHub – https://github.com/cog-imperial/OMLT

README.rst

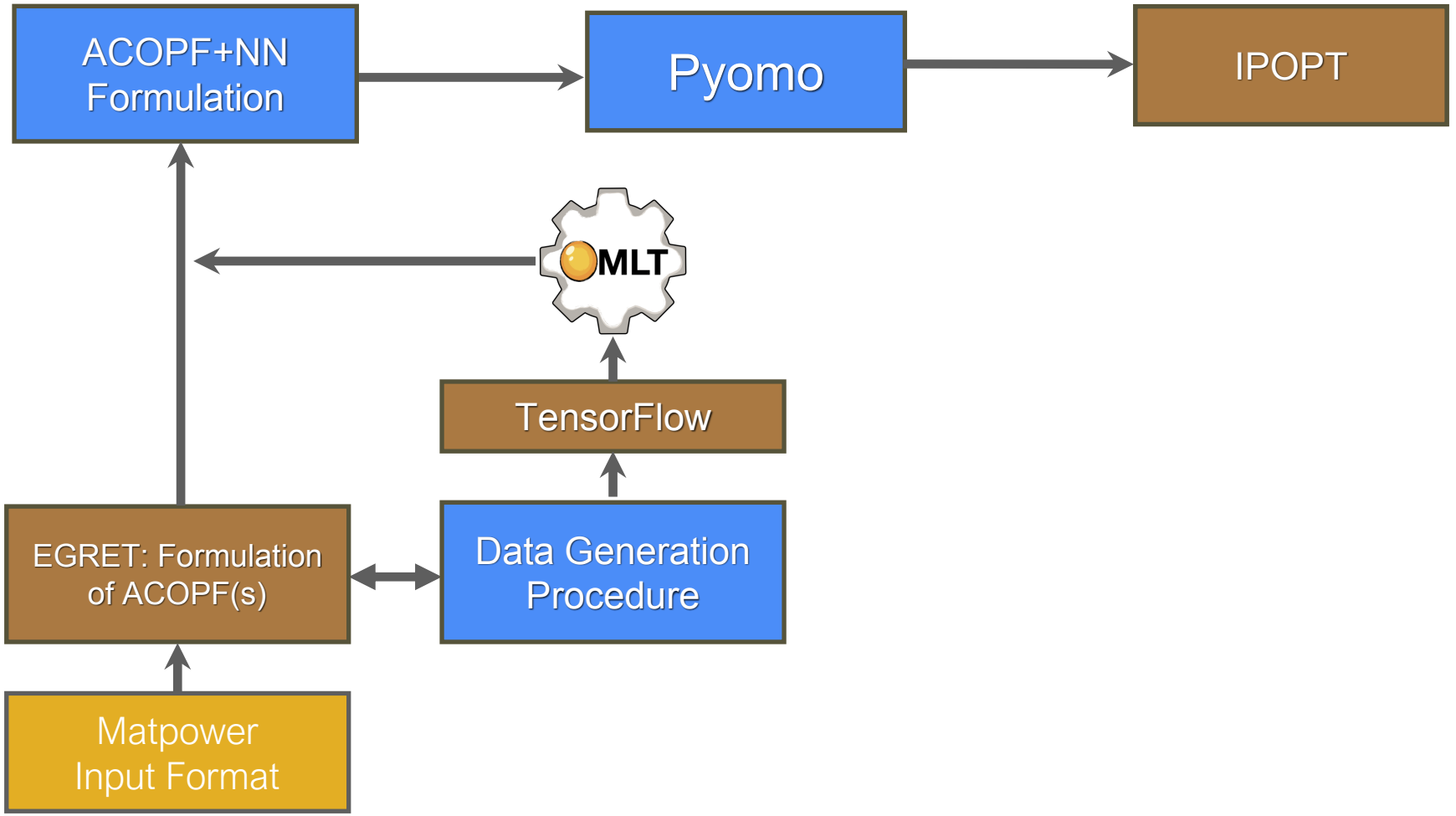
OMLT: Optimization and Machine Learning Toolkit

OMLT is a Python package for representing machine learning models (such as neural networks) within the Pyomo optimization environment. The package provides various formulations for representing machine-learning models (such as full-space, reduced-space, and MILP), as well as an interface to import sequential Keras models.



$$x_i = z_i$$
$$\hat{z}_i = \sum_{j=1}^{N} w_{ij} z_j + b_i$$
$$z_i = \sigma(\hat{z}_i)$$
$$y_i = z_i$$

Building the model with Pyomo and OMLT



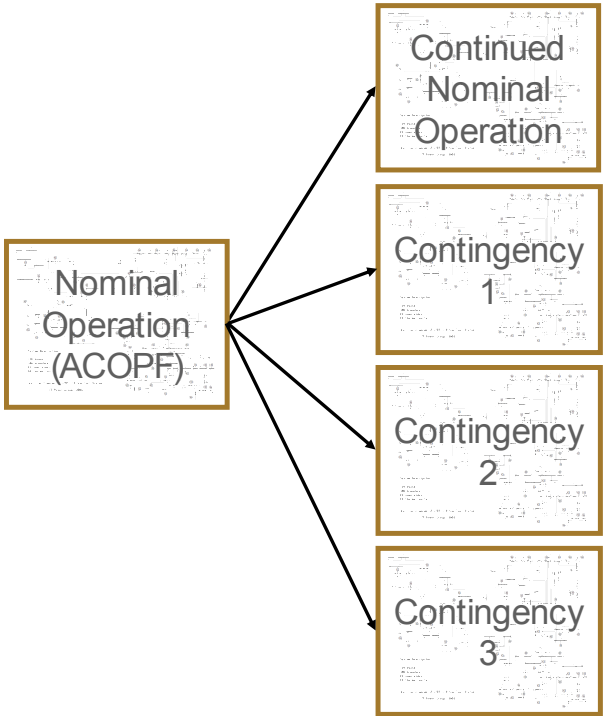
Integrated Python-based toolchain

ACOPF with Neural Network Contingency Constraints

$$\min f(x, y)$$
$$\text{s.t. } c(x, y) = 0$$
$$y = NN(x)$$

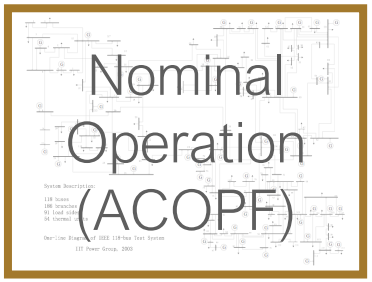
Current efforts focused on:

- Optimization-based approaches for improved sampling
- More extensive performance comparisons (computational timing and solution quality)
- Nonlinear formulations for ReLU networks (complementarity formulations with local solvers)



Train NN on security boundary

Case 118:
~400,000 vars → ~2000 vars
400 sec. → < 10 sec.



+

$$NN(x) \leq \alpha$$

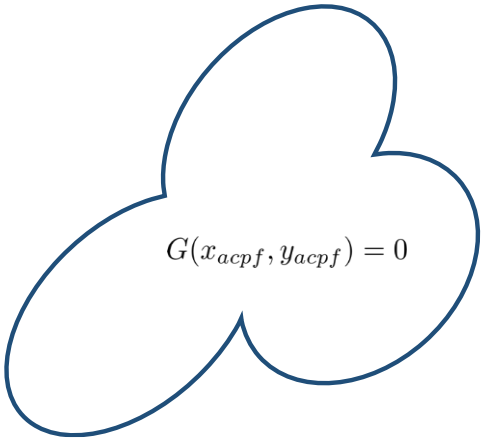
[Kilwein, et al. (2021). AC-Optimal Power Flow Solutions with Security Constraints from Deep Neural Network Models. In Computer Aided Chemical Engineering (Vol. 50, pp. 919-925). Elsevier.]

Neural Network Representation of ACPF Equations

Why "learn" ACPF equations?

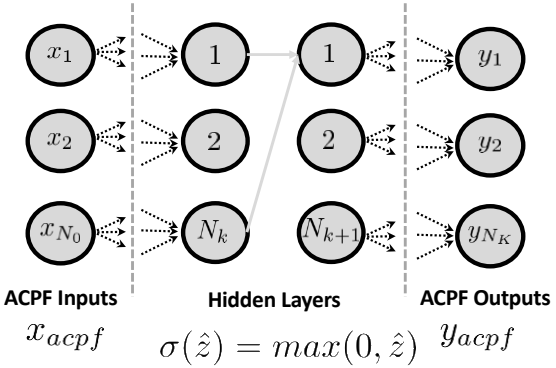
- AC power flow forms the physical basis for planning and operation in power systems
- Applications to ACOPF, SCOPF/CCOPF, Multiperiod OPF, Unit Commitment
- Learning AC power flow equations provides greater flexibility than learning the optimal solution directly (e.g., solution of ACOPF)
- Seek to replace the nonlinear AC power flow equations with a piecewise linear neural network representation (e.g., ReLU)

Nonlinear/Non-convex ACPF Equations



ReLU Network is MILP representable

$$\hat{y}_{acpf} = NN(x_{acpf})$$



Neural Network Representation of ACPF Equations

Why "learn" ACPF equations?

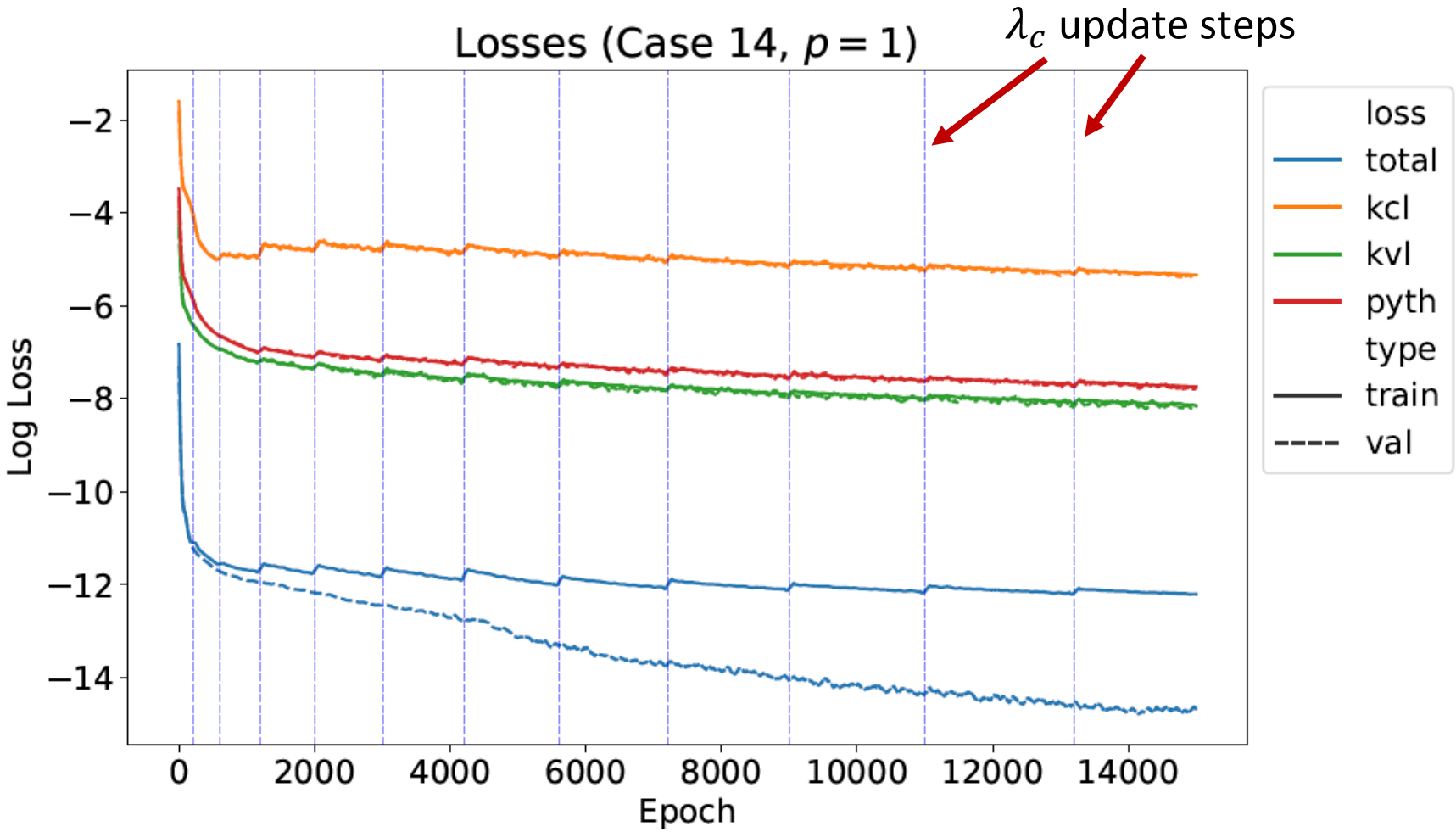
- AC power flow forms the physical basis for planning and operation in power systems
- Applications to ACOPF, SCOPF/CCOPF, Multiperiod OPF, Unit Commitment
- Learning AC power flow equations provides greater flexibility than learning the optimal solution directly (e.g., solution of ACOPF)
- Seek to replace the nonlinear AC power flow equations with a piecewise linear neural network representation (e.g., ReLU)

Eventual goal is to improve general approaches for neural network surrogates

Physics-informed/constrained machine learning for ACPF

- Sampling strategies focused on input space and reasonable operating ranges
- Lagrangian-dual approach for physics constraints in loss function [Fioretto, Mak, Van Hentenryck 2020, 2021]
- Using w, c, s outputs improves training performance
- Optimization-based verifier to provide guarantees on worst-case violation
- Model pruning (sparse networks) improves verifier performance (and can improve accuracy)

Lagrangian-Dual Training

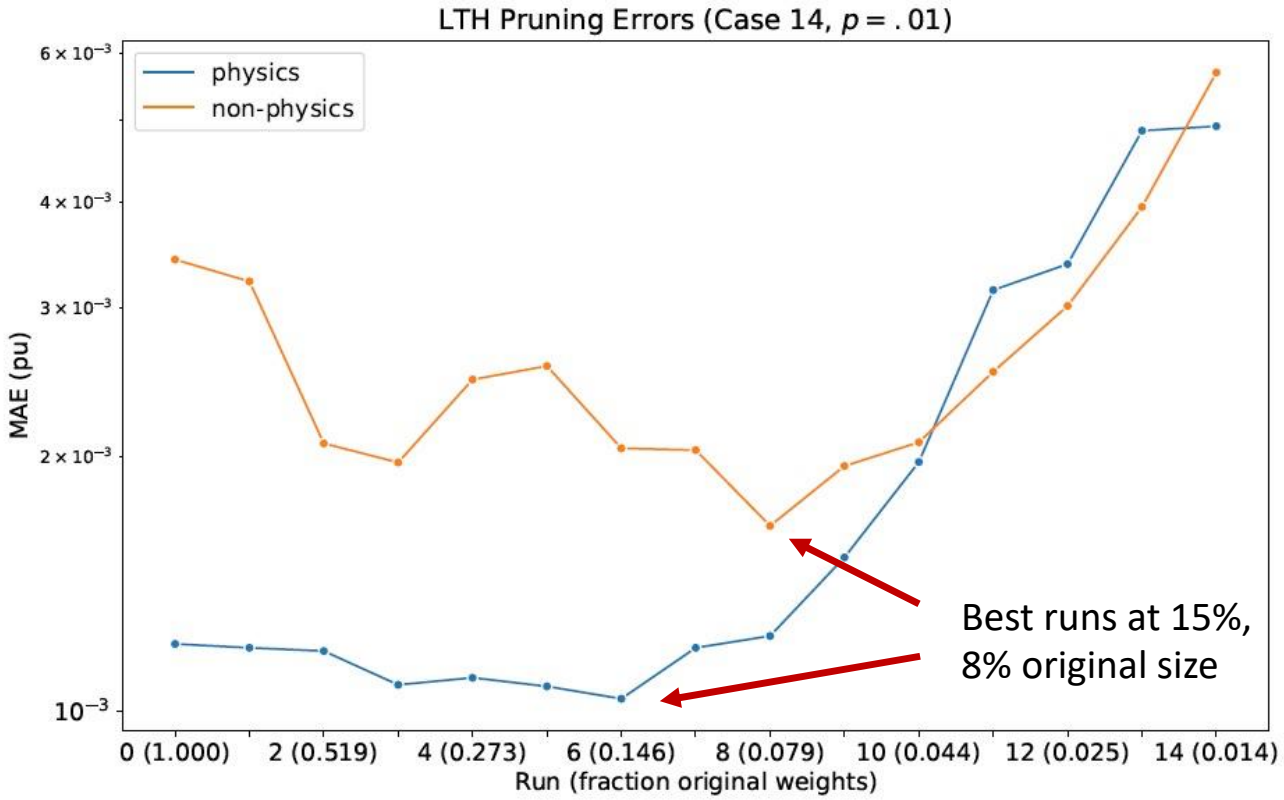


Fioretto, et al. "Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods," *AAAI Conf. on AI*, 2020.

Model Pruning to Reduce Network Size (for Verifier)

PINN model sizes

- 14 bus case:
  - 57k params
  - 400 activations
- 118 bus case:
  - 1.7M params
  - 2000 activations
- Still small by NN standards



Pruning greatly improves downstream NN verification

- Lottery Ticket method: train multiple runs, dropping bottom fraction of params
- Significant reduction in size while maintaining prediction accuracy

Frankle and Carbin. "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks," *arXiv preprint arXiv:1803.03635*, 2019

Neural Network Verifier

Verification Formulation

$$\max_{x} \quad f(x) \qquad \text{Verifier performance metric}$$
$$\text{s.t. } \hat{y} = NN^{ReLU}(x) \qquad \text{NN surrogate}$$
$$h(x, y) = 0 \qquad \text{Nonlinear model}$$
$$g_1(x, y) \leq 0 \qquad \text{Operating constraints}$$
$$g_2(x, \hat{y}) \leq 0 \qquad \text{Operating constraints}$$
$$x^L \leq x \leq x^U \qquad \text{Input limits}$$

- Problem is an MINLP (ReLU + nonlinear model)
- Relaxing nonlinear constraints provides conservative guarantee on verification formulation
- Verifier coded in PowerModels.jl and JuMP

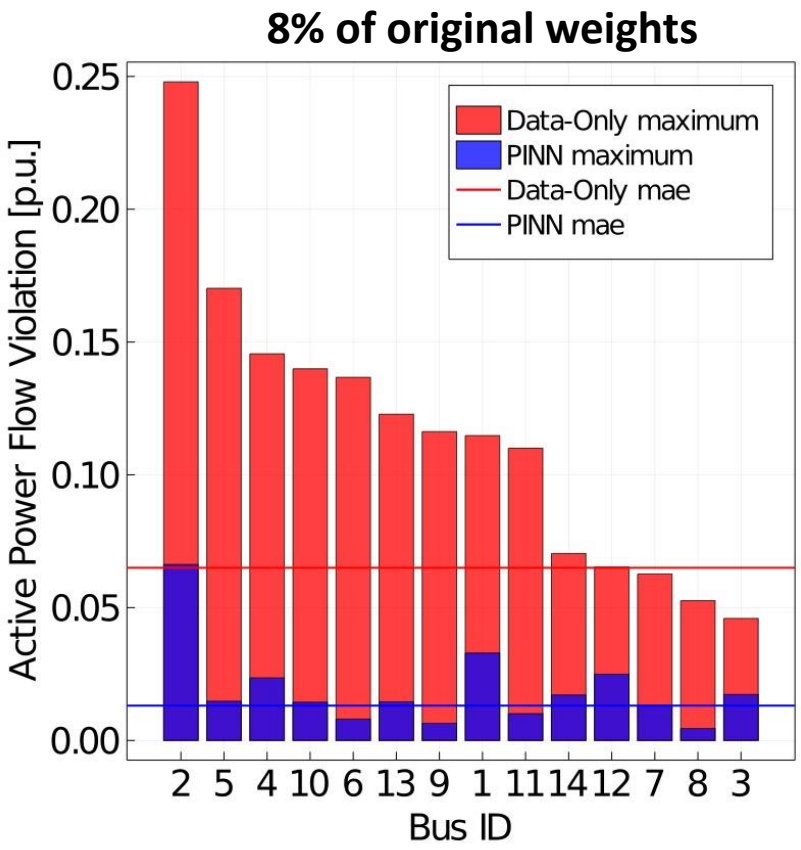
README.md

PowerModels.jl

Status: CI passing | codecov 94% | Documentation passing

PowerModels.jl is a Julia/JuMP package for Steady-State Power Network Optimization. It is designed to enable computational evaluation of emerging power network formulations and algorithms in a common platform. The code is engineered to decouple problem specifications (e.g. Power Flow, Optimal Power Flow, ...) from the power network formulations (e.g. AC, DC-approximation, SOC-relaxation, ...). This enables the definition of a wide variety of power network formulations and their comparison on common problem specifications.

Example: Worst-case power balance violation on single bus



Relaxation and MILP formulation for ReLU enables global verification with MIQP solvers

Summary and Conclusions

There are significant opportunities for integration of optimization and machine learning beyond training

- Efficient formulations and solution strategies for optimization problems with embedded machine learning models (as classifiers or model surrogates)
- Development of physics-informed machine learning approaches with treatment of constraints beyond regularization
- Global optimization strategies for verification of neural network models

Power grid applications:

- Capabilities exist to replace challenging constraints / models with neural network representations
- Capabilities improving to develop piecewise-linear representations of complex nonlinear models

The authors would like to gratefully acknowledge funding from Sandia's Laboratory Directed Research and Development (LDRD) program.