Machine Learning for Solving Optimal Power Flow Problems

Minghua Chen[a] and Steven Low[b]

[a]School of Data Science, City University of Hong Kong
[b]Department of Computing & Mathematical Sciences, Caltech

香港城市大學
City University of Hong Kong

Caltech

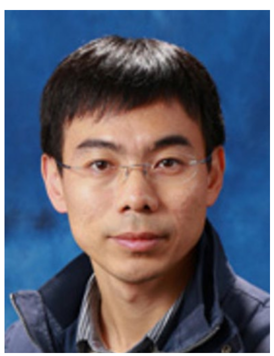
Acknowledgements

Xiang Pan
(CUHK; Tencent)

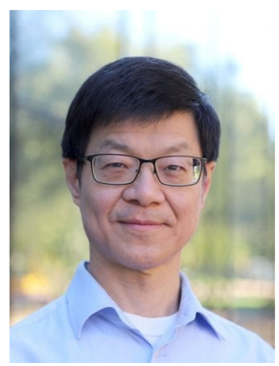
Tianyu Zhao
(CUHK; Lenovo Research)

Min Zhou
(CityU)

Enming Liang
(CityU)



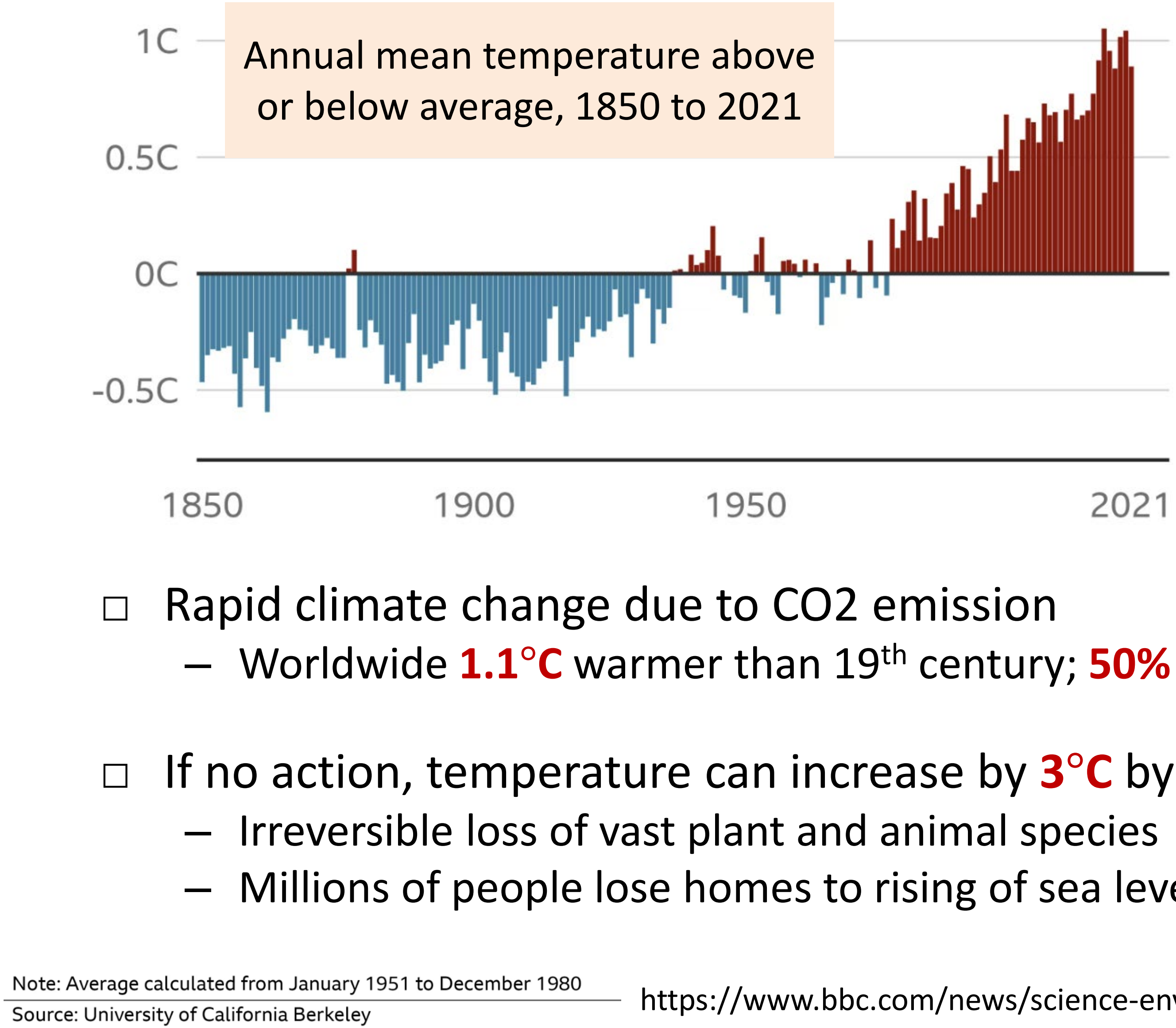
Shengyu Zhang
(Tencent)



Steven Low
(Caltech)

Wanjun Huang
(CityU; Beihang Univ)

Climate Change: the Biggest Threat to Humanity



Target: 1.5°C

- Rapid climate change due to CO_2 emission
  - Worldwide **1.1°C** warmer than 19th century; **50%** more CO_2 in the air
- If no action, temperature can increase by **3°C** by 2100
  - Irreversible loss of vast plant and animal species
  - Millions of people lose homes to rising of sea levels

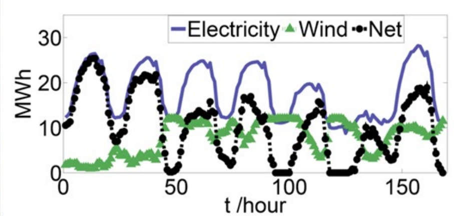
https://www.bbc.com/news/science-environment-24021772

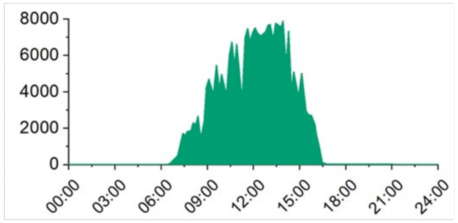
Grid Decarbonization to Fight Climate Change

- Power grids integrate renewable to fight climate change
  - Grids emit **25%** of global CO2
  - Could reach **60%** if all transportation electrified

	2021	2030 (target)
China	29.8%	40%
US	21%	35%
Denmark	82%	100%

- Renewables are volatile
  - More than **80%** renewables are wind or solar
  - Wind and solar are **volatile**
  - Net load inherits **renewable uncertainty**

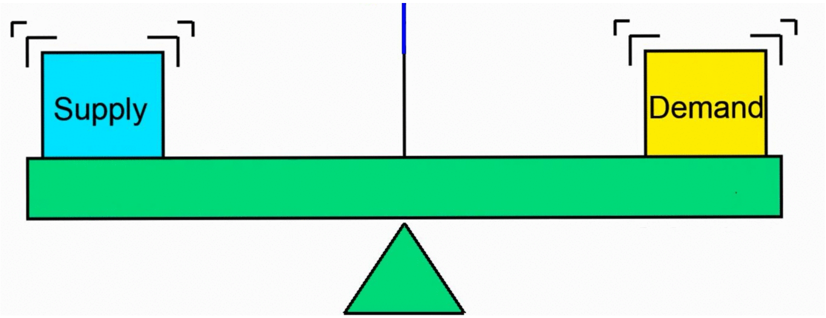




https://english.www.gov.cn/news/topnews/202206/01/content_WS6296ba55c6d02e533532b91f.html; US EIA Annual Energy outlook 2022; IRENA technical report for Denmark; 《新时代的中国能源发展》白皮书; 《广东省培育新能源战略性新兴产业集群行动计划（2021-2025年）》

Volatile Renewables Require Frequent Balancing

Grid operation: balance demand and supply in real-time



Balancing **volatile** renewable requires us to solve **optimal power flow** problems more frequently to track the optimal operating point

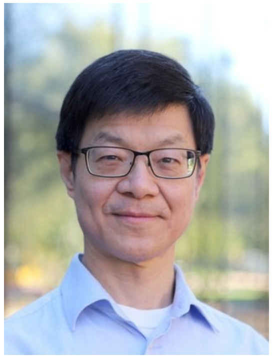
- **Past: once every 12 hours**
- **Present: once every 5 minutes**
- **Future: once every 1 minute**

Outline

- ☐ Optimal power flow (OPF) problems
- ☐ Example applications
- ☐ Future challenges

- ☐ Machine learning for constrained optimization
- ☐ Machine learning for solving OPF problems: overview
- ☐ Machine learning for DC-OPF and SC-DCOPF problems
- ☐ Machine learning for standard AC-OPF problems

- ☐ Ensuring DNN feasibility for constrained optimization
- ☐ Solving AC-OPF under flexible topologies
- ☐ Solving AC-OPF with multiple load-solution mappings
- ☐ Concluding Remarks

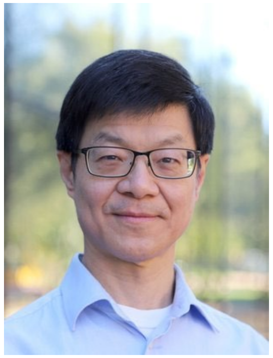


Outline

- ☐ Optimal power flow (OPF) problems
- ☐ Example applications
- ☐ Future challenges

- ☐ Machine learning for constrained optimization
- ☐ Machine learning for solving OPF problems: overview
- ☐ Machine learning for DC-OPF and SC-DCOPF problems
- ☐ Machine learning for standard AC-OPF problems

- ☐ Ensuring DNN feasibility for constrained optimization
- ☐ Solving AC-OPF under flexible topologies
- ☐ Solving AC-OPF with multiple load-solution mappings
- ☐ Concluding Remarks



Loading Steven’s Slides

Optimal power flow

Introduction

Outline

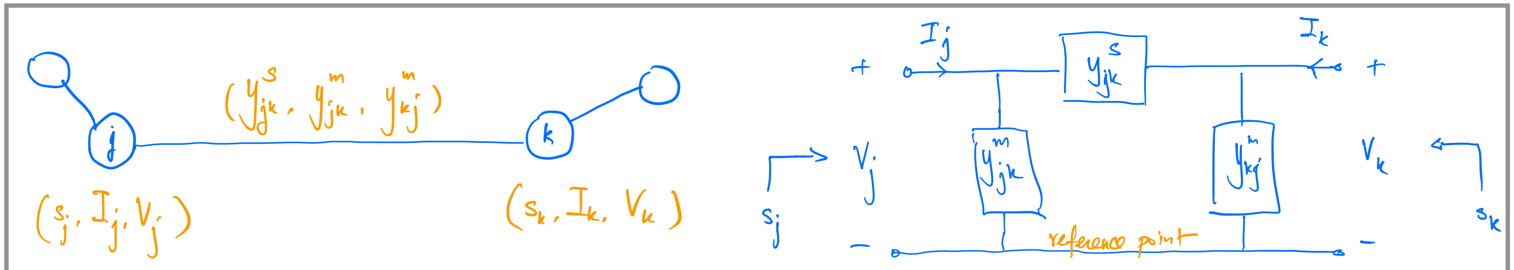
1. Optimal power flow problems
2. Example applications
3. Future challenges

Outline

1. Optimal power flow problems
    - Power flow models
    - OPF formulation
2. Example applications
3. Future challenges

Network model

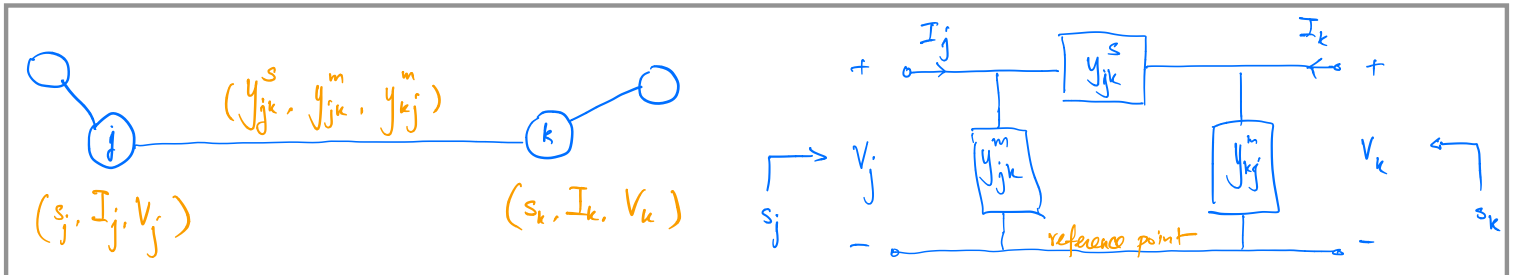
1. Network $G := (\overline{N}, E)$
   - $\overline{N} := \{0\} \cup N := \{0\} \cup \{1,\ldots,N\}$: buses/nodes
   - $E \subseteq \overline{N} \times \overline{N}$: lines/links/edges
2. Each line (j,k) is parameterized by $\left(y^s_{jk}, y^m_{jk}, y^m_{kj}\right) \in \mathbb{C}^3$
   - y^s_{jk} : series admittance
   - y^m_{jk}, y^m_{kj} : shunt admittances, generally different



(a) Graph representation (b) Π equivalent circuit

Network model

Branch currents



(a) Graph representation (b) Π equivalent circuit

Sending-end currents

$$I_{jk} \;=\; y^s_{jk}(V_j - V_k) \;+\; y^m_{jk}\, V_j, \qquad I_{kj} \;=\; y^s_{jk}(V_k - V_j) \;+\; y^m_{kj}\, V_k,$$

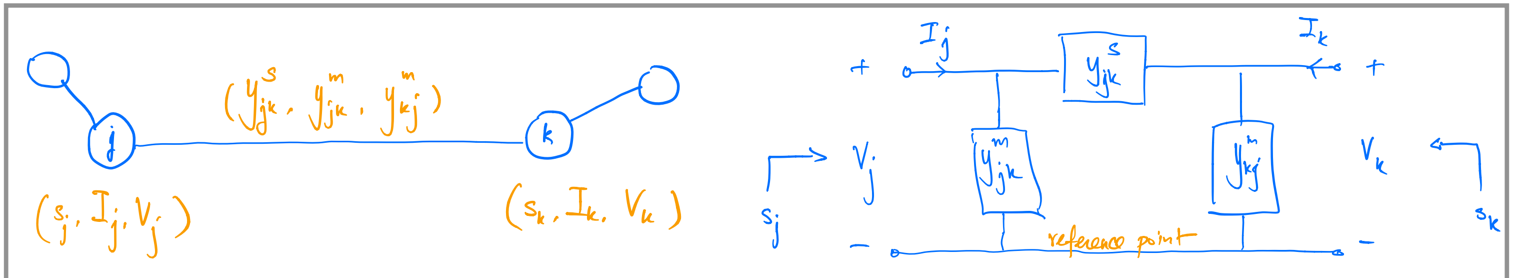
Their sum is total line current loss

$$I_{jk} + I_{kj} = \; y^m_{jk} V_j + y^m_{kj} V_k \quad \neq \quad 0$$

If $y^m_{jk} = y^m_{kj} = 0$, then $I_{jk} = -\, I_{kj}$

Network model

Nodal current balance (KCL)

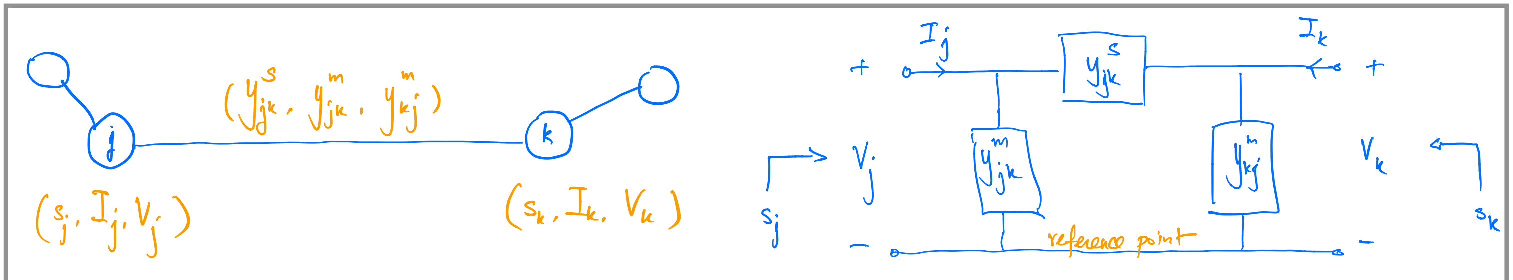


(a) Graph representation (b) Π equivalent circuit

$$I_j = \sum_{k:j\sim k} I_{jk}$$

Network model

Nodal current balance (KCL)



(a) Graph representation (b) Π equivalent circuit

$$I_j = \sum_{k:j\sim k} I_{jk} = \left(\sum_{k:j\sim k} y^s_{jk} + y^m_{jj} \right) V_j - \sum_{k:j\sim k} y^s_{jk} V_k$$

total shunt admittance: $y^m_{jj} := \sum_{k:j\sim k} y^m_{jk}$

Network model

Nodal current balance (KCL)

$$I_j = \sum_{k:j\sim k} I_{jk} = \left(\sum_{k:j\sim k} y^s_{jk} + y^m_{jj} \right) V_j - \sum_{k:j\sim k} y^s_{jk} V_k$$

In vector form:

$$I = YV \text{ where } Y_{jk} = \begin{cases} -y^s_{jk}, & j \sim k \quad (j \neq k) \\ \sum_{l:j\sim l} y^s_{jl} + y^m_{jj}, & j = k \\ 0 & \text{otherwise} \end{cases}$$

Network model

Nodal current balance (KCL)

Y can be written down by inspection of network graph

- Off-diagonal entry: $-$ series admittance
- Diagonal entry: $\sum$series admittances $+$ total shunt admittance

In vector form:

$$I \;=\; YV \quad \text{where} \quad Y_{jk} = \begin{cases} -y^s_{jk}, & j \sim k \quad (j \neq k) \\ \sum_{l:j\sim l} y^s_{jl} \;+\; y^m_{jj}, & j = k \\ 0 & \text{otherwise} \end{cases}$$

Network model

Nodal current balance (KCL)

A matrix Y is an admittance matrix iff it is complex symmetric

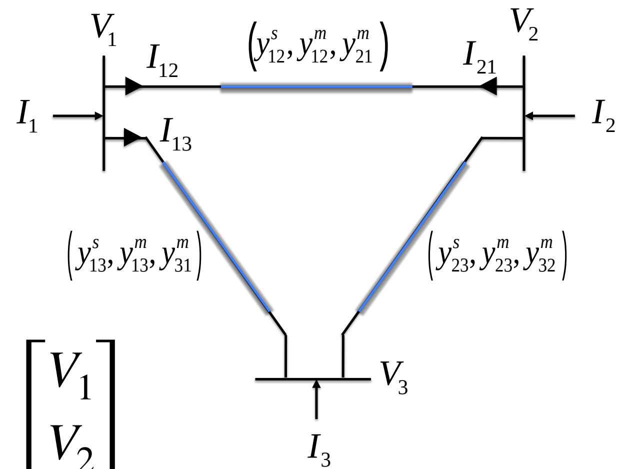
- Can be interpreted as a Π circuit

In vector form:

$$I \;=\; Y\,V \quad \text{where} \quad Y_{jk} = \begin{cases} -y^s_{jk}, & j \sim k \quad (j \neq k) \\ \sum_{l:j\sim l} y^s_{jl} \;+\; y^m_{jj}, & j = k \\ 0 & \text{otherwise} \end{cases}$$

Admittance matrix Y

Example



$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} y^s_{12} + y^s_{13} + y^m_{11} & -y^s_{12} & -y^s_{13} \\ -y^s_{12} & y^s_{12} + y^s_{23} + y^m_{22} & -y^s_{23} \\ -y^s_{13} & -y^s_{23} & y^s_{13} + y^s_{23} + y^m_{33} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix}$$

total shunt admittance: $y^m_{jj} \ := \ \sum_{k:j\sim k} y^m_{jk}$

Linear analysis

In absence of constant-power devices on network

- e.g. voltage sources, current sources, impedances
- Only linear analysis is needed based on $I = YV$

But applications often require power $s_j := V_j \bar{I}_j$

- e.g. EV charging needs 30 miles of energy (10 kWh) in 5 hours
- Leads to nonlinear analysis / optimization

Power flow models

Complex form

Define sending-end power from $j \to k$, $S_{jk} := V_j I_{jk}^H$:

$$S_{jk} = \left(y_{jk}^s\right)^H \left(|V_j|^2 - V_j V_k^H \right) + \left(y_{jk}^m\right)^H |V_j|^2$$

$$S_{kj} = \left(y_{jk}^s\right)^H \left(|V_k|^2 - V_k V_j^H \right) + \left(y_{kj}^m\right)^H |V_k|^2$$

Line loss

$$S_{jk} + S_{kj} = \left(y_{jk}^s\right)^H \left| V_j - V_k \right|^2 + \left(y_{jk}^m\right)^H |V_j|^2 + \left(y_{kj}^m\right)^H |V_k|^2$$

series impedance shunt impedances

Power flow models

Complex form

Bus injection model $s_j = \sum_{k:j\sim k} S_{jk}$:

$$s_j = \sum_{k:j\sim k} \left(y_{jk}^s\right)^H \left(|V_j|^2 - V_j V_k^H\right) + \left(y_{jj}^m\right)^H |V_j|^2$$

In terms of admittance matrix Y

$$s_j = \sum_{k=1}^{N+1} Y_{jk}^H V_j V_k^H$$

$N+1$ complex equations in $2(N+1)$ complex variables $\left(s_j, V_j, j \in \overline{N}\right)$

Power flow models

Polar form

Write $s_j \;=:\; p_j + iq_j$ and $V_j =: |V_j|\, e^{i\phi}$ with $y^s_{jk} =: g^s_{jk} + ib^s_{jk}$, $y^m_{jk} =: g^m_{jk} + ib^m_{jk}$:

$$p_j = \left(\sum_{k=0}^{N} g_{jk}\right) |V_j|^2 \;-\; \sum_{k\neq j} |V_j|\,|V_k| \left(g_{jk}\cos\theta_{jk} + b_{jk}\sin\theta_{jk}\right)$$

$$q_j = -\left(\sum_{k=0}^{N} b_{jk}\right) |V_j|^2 \;-\; \sum_{k\neq j} |V_j|\,|V_k| \left(g_{jk}\sin\theta_{jk} - b_{jk}\cos\theta_{jk}\right)$$

where $g_{jk} := \begin{cases} g^m_{jj} & \text{if } j = k \\ g^s_{jk} & \text{if } j \neq k,\ (j,k) \in E \\ 0 & \text{if } j \neq k,\ (j,k) \notin E \end{cases}$ $\qquad b_{jk} := \begin{cases} b^m_{jj} & \text{if } j = k \\ b^s_{jk} & \text{if } j \neq k,\ (j,k) \in E \\ 0 & \text{if } j \neq k,\ (j,k) \notin E \end{cases}$

Power flow models

Polar form

Write $s_j \;=:\; p_j + iq_j$ and $V_j =: |V_j| e^{i\phi}$ with $y^s_{jk} =: g^s_{jk} + ib^s_{jk}$, $y^m_{jk} =: g^m_{jk} + ib^m_{jk}$:

$$p_j = \left(\sum_{k=0}^{N} g_{jk}\right)|V_j|^2 - \sum_{k\neq j} |V_j||V_k|\left(g_{jk}\cos\theta_{jk} + b_{jk}\sin\theta_{jk}\right)$$

$$q_j = -\left(\sum_{k=0}^{N} b_{jk}\right)|V_j|^2 - \sum_{k\neq j} |V_j||V_k|\left(g_{jk}\sin\theta_{jk} - b_{jk}\cos\theta_{jk}\right)$$

$2(N+1)$ real equations in $4(N+1)$ real variables $\left(p_j, q_j, |V_j|, \theta_j, j \in \overline{N}\right)$

Power flow models

Cartesian form

Write $s_j \;=:\; p_j + iq_j$ and $V_j =: e_j + if_j$:

$$p_j = \left(\sum_k g_{jk}\right)\left(e_j^2 + f_j^2\right) - \sum_{k \neq j} \left(g_{jk}(e_j e_k + f_j f_k) + b_{jk}(f_j e_k - e_j f_k)\right)$$

$$q_j = -\left(\sum_k b_{jk}\right)\left(e_j^2 + f_j^2\right) - \sum_{k \neq j} \left(g_{jk}(f_j e_k - e_j f_k) - b_{jk}(e_j e_k + f_j f_k)\right)$$

$2(N+1)$ real equations in $4(N+1)$ real variables $\left(p_j,\, q_j,\, e_j, f_j,\, j \in \overline{N}\right)$

Power flow models

Types of buses

Power flow equations specify $2(N+1)$ real equations in $4(N+1)$ real variables

- Power flow (load flow) problem: given $2(N+1)$ values, determine remaining vars

Types of buses

- PV buses : $\left(p_j, |V_j|\right)$ specified, determine $\left(q_j, \theta_j\right)$, e.g. generator
- PQ buses : $\left(p_j, q_j\right)$ specified, determine V_j, e.g. load
- Slack bus 0 : $V_0 := 1\angle 0^\circ$ pu specified, determine $\left(p_j, p_j\right)$

Outline

1. Optimal power flow problems
   - Power flow models
   - OPF formulation
2. Example applications
3. Future challenges

OPF formulation

Optimal power flow (OPF) problems are fundamental

- Numerous power system applications can be formulated as OPF

Network model: graph $G = (\overline{N}, E)$

OPF is a constrained optimization program specified by

- Optimization variables
- Power flow equations
- Cost function
- Operational constraints

OPF formulation

Optimization vars

Optimization vars $(s, V) := \left(s_j, V_j, j \in \overline{N}\right)$

- s_j : real & reactive power injections
- V_j : voltage phasors
- Can express s_j in terms of V using power flow equations

We call this the bus injection model since s is the only variable (besides V)

OPF formulation

Power flow equations

Nodal power balance:

$$s_j \;=\; \sum_{k:j\sim k} S_{jk}(V), \qquad j \in \overline{N}$$

where

- Complex form: $S_{jk}(V) \;=\; \left(y_{jk}^s\right)^H \left(|V_j|^2 - V_j V_k^H\right) \;+\; \left(y_{jk}^m\right)^H |V_j|^2$
- Polar form:

$$P_{jk}(V) = \left(g_{jk}^s + g_{jk}^m\right)|V_j|^2 \;-\; |V_j||V_j|\left(g_{jk}^s \cos(\theta_j - \theta_k) \;-\; b_{jk}^s \sin(\theta_j - \theta_k)\right)$$

$$Q_{jk}(V) = \left(b_{jk}^s + b_{jk}^m\right)|V_j|^2 \;-\; |V_j||V_k|\left(b_{jk}^s \cos(\theta_j - \theta_k) \;+\; g_{jk}^s \sin(\theta_j - \theta_k)\right)$$

OPF formulation

Cost function

Total real power loss

$$C_0(V) \;:=\; \sum_j \text{Re}\left(s_j(V)\right) \;=\; \sum_j \text{Re}\left(\sum_{k:j\sim k}\left(y_{jk}^s\right)^{\mathsf{H}}\left(|V_j|^2 - V_j V_k^{\mathsf{H}}\right) \;+\; \left(y_{jj}^m\right)^{\mathsf{H}} |V_j|^2\right)$$

Total generation cost

$$C_0(V) := \sum_{j:\text{gens}} c_j \, \text{Re}(s_j(V)) \;=\; \sum_{j:\text{gens}} c_j \, \text{Re}\left(\sum_{k:j\sim k}\left(y_{jk}^s\right)^{\mathsf{H}}\left(|V_j|^2 - V_j V_k^{\mathsf{H}}\right) \;+\; \left(y_{jj}^m\right)^{\mathsf{H}} |V_j|^2\right)$$

Cost functions can usually be expressed as (possibly nonlinear) functions of V

Optimal formulation

Operational constraints

Injection limits: $\underline{s}_j \;\le\; s_j(V) \;\le\; \overline{s}_j$

Voltage limits: $\underline{v}_j \;\le\; |V_j|^2 \;\le\; \overline{v}_j$

Line limits: $|I_{jk}(V)|^2 \;\le\; \overline{I}_{jk}^2, \quad |I_{kj}(V)|^2 \;\le\; \overline{I}_{kj}^2$

or $|S_{jk}(V)| \;\le\; \overline{S}_{jk}, \quad |S_{kj}(V)| \;\le\; \overline{S}_{kj}$

Let feasible set be $\mathbb{V} := \left\{ V \in \mathbb{C}^{N+1} \mid V \text{ satisfies injection/voltage/line limits} \right\}$

Optimal formulation

OPF in bus injection model

$$\min_{V \in \mathbb{V}} \quad C_0(V)$$

Remarks

- Flexible formulation; e.g.
  - $\underline{s}_0 := -\infty - \mathrm{i}\infty$ or $\overline{s}_0 := \infty + \mathrm{i}\infty$ if s_0 has no limits
  - $\underline{s}_j = \overline{s}_j$ if s_j is a parameter (given inelastic demand)
- Can have multiple devices k with injections s_{jk} at bus j : net injection $s_j = \sum_k s_{jk}$

NP hardness

OPF is NP-hard

- Verma 2009, Bienstock & Verma 2019
- Lavaei & Low 2012
- Lehmann, Grastien & Van Hentenryck 2016

IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 31, NO. 1, JANUARY 2016

AC-Feasibility on Tree Networks is NP-Hard

Karsten Lehmann, Alban Grastien, and Pascal Van Hentenryck

Reduce NP-hard **subset sum problem** to:

Find $\left(\Theta_i, p_{[ij]}, q_{[ij]}\right)$ s.t. power flow equations & constraints

$$\forall i \in N_L : \sum_{[ij]\in E^d} p_{[ij]} = P_i \qquad \forall [ij]_b^g \in E^d : p_{[ij]} = g\,(1-\cos(\Theta_i-\Theta_j)) - b\sin(\Theta_i-\Theta_j)$$

$$\sum_{[ij]\in E^d} q_{[ij]} = Q_i \qquad q_{[ij]} = -\,b\,(1-\cos(\Theta_i-\Theta_j)) - g\sin(\Theta_i-\Theta_j)$$

$$\forall i \in N_G : \sum_{[ij]\in E^d} p_{[ij]} \geq 0 \qquad |\Theta_i-\Theta_j| \leq \overline{\Delta}.$$

Outline

1. Optimal power flow problems
2. Example applications
   - Optimal dispatch
   - Unit commitment
   - Security constrained dispatch/commitment
3. Future challenges

Central challenge

Balance supply & demand second-by-second

- While satisfying operational constraints, e.g. injection/voltage/line limits
- Unlike usual commodities, electricity cannot (yet) be stored in large quantity

Traditional approach

Bulk generators generate 80% of electricity in US (2020)

- Fossil (gas, coal): 60%, nuclear: 20%

They are fully dispatchable and centrally controlled

- ISO determines in advance how much each generates when & where

They mostly determine dynamics and stability of entire network

- System frequency, voltages, prices

Traditional approach

Challenges

- Large startup/shutdown time and cost
- Uncertainty in future demand (depends mostly on weather)
- Contingency events such as generator/transmission outages

Elaborate electricity markets and hierarchical control

- Schedule generators and determine wholesale prices
- Day-ahead (12-36 hrs in advance): unit commitment
- Real-time (5-15 mins in advance): economic dispatch
- Ancillary services (secs - hours): frequency control, reserves

All of these decisions can be formulated as OPF problems

Optimal power flow

OPF underlies many power system applications

Constrained optimization

$$\min_{u,x} \quad c(u,x) \qquad \text{s.t.} \qquad f(u,x) = 0, \quad g(u,x) \le 0$$

- Optimization vars: control u, network state x
- Cost function: $c(u,x)$
- Constraint functions: $f(u,x)$, $g(u,x)$
- They depend on the application under study

Optimal dispatch

Solved by ISO in real-time market every 5-15 mins

- Determine injection levels of those units that are online
- Adjustment to dispatch from day-ahead market (unit commitment)

Optimal dispatch

Problem formulation

Model

- Network: graph $G = (\overline{N}, E)$

Optimization vars

- Control:
  - Dispatch: power injections $u := (u_j, j \in \overline{N})$ (denoted by s_j previously)
- Network state:
  - Voltages $V := (V_j, j \in \overline{N})$
  - Line flows $S := \left(S_{jk}, S_{kj}, (j,k) \in E\right)$

Optimal dispatch

Problem formulation

Parameters

- Uncontrollable injections $\sigma := \left(\sigma_j, j \in \overline{N}\right)$, e.g. load forecast

Generation cost is quadratic in real power

$$c(u, x) \quad = \sum_{\text{generators } j} \left(a_j \left(\mathrm{Re}(u_j)\right)^2 \; + \; b_j \,\mathrm{Re}(u_j) \right)$$

Optimal dispatch

Constraints

Power flow equations: $S = S(V)$

- Complex form: $S_{jk}(V) \;=\; \left(y^s_{jk}\right)^H \left(\,|V_j|^2 - V_j V_k^H\right) \;+\; \left(y^m_{jk}\right)^H |V_j|^2$
- Polar form:

$$P_{jk}(V) = \left(g^s_{jk} + g^m_{jk}\right)|V_j|^2 \;-\; |V_j|\,|V_j|\left(g^s_{jk}\cos(\theta_j - \theta_k) \;-\; b^s_{jk}\sin(\theta_j - \theta_k)\right)$$

$$Q_{jk}(V) = \left(b^s_{jk} + b^m_{jk}\right)|V_j|^2 \;-\; |V_j|\,|V_k|\left(b^s_{jk}\cos(\theta_j - \theta_k) \;+\; g^s_{jk}\sin(\theta_j - \theta_k)\right)$$

Power balance: $u_j \;+\; \sigma_j \;\;=\;\; \sum_{k:j\sim k} S_{jk}(V)$

Optimal dispatch

Constraints

Injection limits: $\underline{u}_j \ \leq \ u_j \ \leq \ \overline{u}_j$

Voltage limits: $\underline{v}_j \ \leq \ |V_j|^2 \ \leq \ \overline{v}_j$

Line limits: $|S_{jk}(V)| \ \leq \ \overline{S}_{jk}, \quad |S_{kj}(V)| \ \leq \ \overline{S}_{kj}$

Optimal dispatch

$$\min_{u,x} \quad c(u,x)$$

$$\text{s.t.} \quad u_j + \sigma_j = \sum_{k:j\sim k} S_{jk}(V)$$

$$\underline{u}_j \leq u_j \leq \overline{u}_j$$

$$\underline{v}_j \leq |V_j|^2 \leq \overline{v}_j$$

$$|S_{jk}(V)| \leq \overline{S}_{jk}, \quad |S_{kj}(V)| \leq \overline{S}_{kj}$$

$u^{\text{opt}}(\sigma)$: optimal dispatch driven by σ

Optimal dispatch

Interpretation

- ISO dispatches u_j^{opt} to unit j as generation setpoint (needs incentive compatibility)
- Resulting network state x^{opt} satisfies operational constraints

Economic dispatch in practice

- Real-time market use linear approximation, e.g., DC power flow, instead of AC (nonlinear) power flow equations
- ISO solves linear program for dispatch and wholesale prices
- AC power flow equations are used to verify that operational constraints are satisfied if dispatched
- If not, DC OPF is modified and procedure repeated
- Even this is a highly simplified approximation of the actual market process

Outline

1. Optimal power flow problems
2. Example applications
   - Optimal dispatch
   - Unit commitment
   - Security constrained dispatch/commitment
3. Future challenges

Unit commitment

Solved by ISO in day-ahead market 12-36 hrs in advance

- Determine which generators will be on (commitment) and their output levels (dispatch)
- For each hour (or half hour) over 24-hour period
- Commitment decisions are binding
- Dispatch decisions may be binding or advisory

Two-stage optimization

- Determine commitment, based on assumption that dispatch will be optimized

Unit commitment

Problem formulation

Model

- Network: graph $G = (\overline{N}, E)$
- Time horizon: $T := \{1,2,\ldots,T\}$, e.g., $t = 1$ hour, $T = 24$

Optimization vars

- Control:
  - Commitment: on/off status $\kappa(t) := \left(\kappa_j(t), j \in \overline{N}\right), \ \kappa_j(t) \in \{0,1\}$
  - Dispatch: power injections $u(t) := (u_j(t), j \in \overline{N})$
- Network state:
  - Voltages $V(t) := (V_j(t), j \in \overline{N})$
  - Line flows $S(t) := \left(S_{jk}(t), S_{kj}(t), (j,k) \in E\right)$

Unit commitment

Problem formulation

Capacity limits: injection is bounded if it is turned on

$$\underline{u}_j(t)\kappa_j(t) \quad \leq \quad u_j(t) \quad \leq \quad \overline{u}_j(t)\kappa_j(t)$$

Startup and shutdown incur costs regardless of injection level

$$d_{jt}(\kappa_j(t-1), \kappa_j(t)) = \begin{cases} \text{startup cost} & \text{if} \quad \kappa_j(t) - \kappa_j(t-1) = 1 \\ \text{shutdown cost} & \text{if} \quad \kappa_j(t) - \kappa_j(t-1) = -1 \\ 0 & \text{if} \quad \kappa_j(t) - \kappa_j(t-1) = 0 \end{cases}$$

UC problems in practice includes other features

- Once turned on/off, bulk generator stays in same state for minimum period

Unit commitment

Problem formulation

Two-stage optimization

$$\min_{\kappa \in \{0,1\}^{(N+1)T}} \quad \sum_t \sum_j d_{jt}\Big(\kappa_j(t-1), \kappa_j(t)\Big) \quad + \quad c^*(\kappa)$$

where $c^*(\kappa)$ is optimal dispatch cost over entire horizon T:

$$c^*(\kappa) \quad := \quad \min_{(u,x)} \quad \sum_t c_t(u(t), x(t); \kappa(t))$$

$$\text{s.t.} \quad f_t(u(t), x(t); \kappa(t)) = 0, \quad g_t(u(t), x(t); \kappa(t)) \le 0, \quad t \in T$$

$$\tilde{f}(u, x) = 0, \quad \tilde{g}(u, x) \le 0$$

- Each time t constraint includes injection limits
- $\tilde{f}(u, x) = 0, \quad \tilde{g}(u, x) \le 0$ can include ramp rate limits

Unit commitment

UC is more computationally challenging than optimal dispatch

- Discrete variables (nonconvexity)
- Multi-interval (larger problem size)

Unit commitment

UC in practice

- Binary variable and multiple intervals make UC computationally difficult for large networks
- Typically use linear model, e.g., DC power flow, and solve mixed integer linear program

Serious effort underway in R&D community to scale UC solution with AC model

- e.g., ARPA-E Grid Optimization Competition Challenge 2 (more later)

Outline

1. Optimal power flow problems
2. Example applications
   - Optimal dispatch
   - Unit commitment
   - Security constrained dispatch/commitment
3. Future challenges

System security

- System security refers to ability to withstand contingency events
- A contingency event is an outage of a generator, transmission line, or transformer
- Contingency events are rare, but can be catastrophic
- NERC's (North America Electricity Reliability Council) $N-1$ rule the outage of a single piece of equipment should not result in violation of voltage or line limits

System security

Secure operation

- Analyze credible contingencies that may lead to voltage or line limit violations
- Account for these contingencies in optimal commitment and dispatch schedules (security constrained UC/ED)
- Monitor system state in real time and take corrective actions when contingency arises

Optimal dispatch

Recall: OPF without security constraints (base case):

$$\min_{(u_0, x_0)} \quad c_0\left(u_0, x_0\right)$$

$$\text{s.t.} \quad f_0\left(u_0, x_0\right) \; = \; 0, \quad g_0\left(u_0, x_0\right) \; \leq \; 0$$

where

- u_0 : dispatch in base case
- x_0 : network state in base case
- $f_0\left(u_0, x_0\right)$: power flow equations, etc.
- $g_0\left(u_0, x_0\right)$: operational constraints

Security constrained OPF

Preventive approach

Basic idea

- Augment optimal dispatch (OPF) with additional constraints …
- … so that the (new) network state under optimal dispatch u^{opt} will satisfy operational constraints after contingency events
- Dispatch remains unchanged until next update period, even if a contingency occurs in the middle of control interval

Security constrained OPF

Preventive approach

Security constrained OPF (SCOPF)

$$\min_{(u_0, x_0,\ \tilde{x}_k,\ k \geq 1)} \quad c_0\left(u_0, x_0\right)$$

$$\text{s.t.} \quad f_0\left(u_0, x_0\right) = 0, \quad g_0\left(u_0, x_0\right) \leq 0 \quad \text{base case constraints}$$

$$\tilde{f}_k\left(u_0, \tilde{x}_k\right) = 0, \quad \tilde{g}_k\left(u_0, \tilde{x}_k\right) \leq 0 \quad \text{constraints after cont. } k$$

where

- $\tilde{x}_k$: new state under same dispatch u_0 after contingency k
- $\tilde{f}_0\left(u_0, \tilde{x}_0\right)$: power flow equations for post-contingency network
- $\tilde{g}_0\left(u_0, \tilde{x}_0\right)$: (more relaxed) emergency operational constraints after contingency k

Security constrained OPF

Corrective approach

Basic idea

- Compute optimal dispatch not only for base case, but also for each contingency k
- System operator can dispatch a response immediately after contingency without waiting till next dispatch period

Security constrained OPF

Corrective approach

Security constrained OPF (SCOPF)

$$
\begin{aligned}
\min_{(u_k, x_k,\ k \geq 0)} \quad & \sum_{k \geq 0} w_k\, c_k \left(u_k, x_k\right) \\
\text{s.t.} \quad & f_k \left(u_k, x_k\right) \ = \ 0, \quad g_k \left(u_k, x_k\right) \ \leq \ 0, \quad k \geq 0 \\
& \|u_k - u_0\| \ \leq \ \rho_k, \quad k \geq 1 \qquad \text{ramp rate limits}
\end{aligned}
$$

where

- $\left(u_k, x_k\right)$: dispatch & state in base case $k = 0$ and after contingency $k \geq 1$
- $\left(f_k, g_k\right)$: power flow equations & operational constraints for $k \geq 0$
- $\|u_k - u_0\|$: ramp rate limits

Outline

1. Optimal power flow problems
2. Example applications
3. Future challenges

Computational challenges

Practical OPF

Non-convergence

- Ill conditioning, bad initial point, nonconvexity

Nonconvexity of power flow equations

- Quadratic, trigonometric

Large problem size

- Large number of variables and constraints
- Relaxation based methods difficult to scale

Nonsmoothness

- Nonsmooth constraints, logical constraints, complementary constraints, mixed integer constraints

Computational methods

Newton-Raphson is the most widely used solution method

- Good (well understood) convergence property

Other popular methods

- Fast decoupled methods: approximate Newton-Raphson
- Interior point methods

Recent approaches

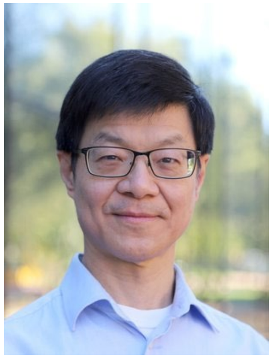
- Based on convex relaxations: semidefinite relaxations, strong SOCP, QC relaxation
- Based on machine learning and neural networks (this tutorial)

Outline

- □ Optimal power flow (OPF) problems
- □ Example applications
- □ Future challenges

- □ Machine learning for constrained optimization
- □ Machine learning for solving OPF problems: overview
- □ Machine learning for DC-OPF and SC-DCOPF problems
- □ Machine learning for standard AC-OPF problems

- □ Ensuring DNN feasibility for constrained optimization
- □ Solving AC-OPF under flexible topologies
- □ Solving AC-OPF with multiple load-solution mappings
- □ Concluding Remarks

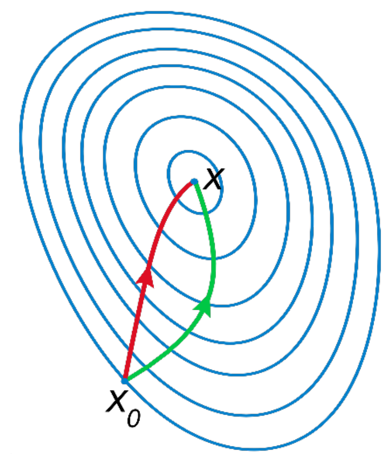


Machine learning for constrained optimization

Constrained Optimization

$$\begin{aligned} \min_{x} \quad & f(x, z) \\ s.t. \quad & g_i(x, z) = 0, \; i = 1, \dots. n \\ & \mathrm{h}_j(x, z) \leq 0, \; j = 1, \dots. m \end{aligned}$$

z: input parameter vector
x: decision variable vector



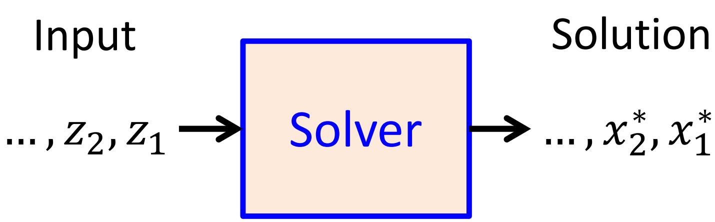
Gradient descent (green)
Newton's method (red)

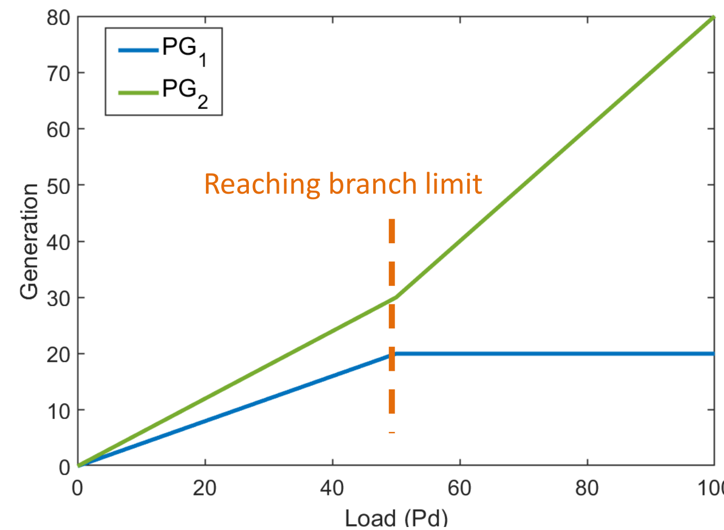
- Tremendous applications; many off-the-shelf solvers
- Given z, solvers apply iterative strategies to ... ns
  - E.g., the gradient descent method (v...
    $x(t+1) = x$ ...
  - E.g., the Newton-Raphson metho... ...dditional curvature information

Universal but high run-time complexity

Picture source: wiki

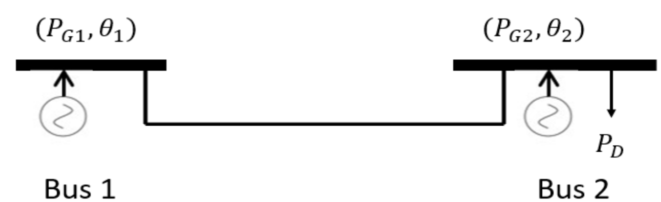
An Input-Solution Mapping Perspective



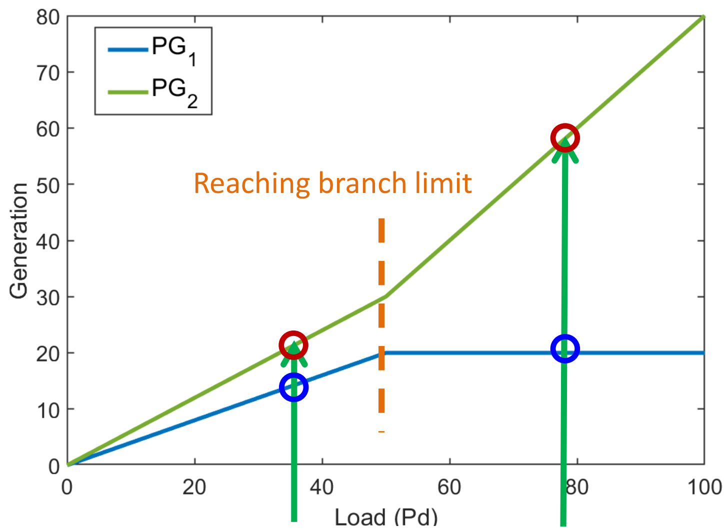


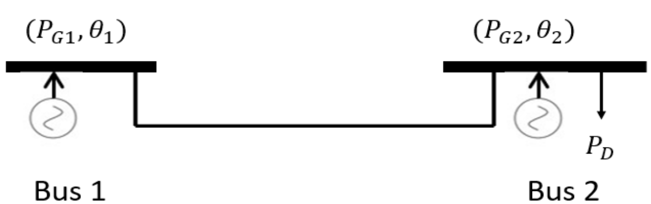
- A solver implicitly characterizes an input-solution mapping for a problem

- Example: The load-generation mapping for a DC-OPF problem over a 2-bus instance



New Machine Learning Viewpoint





- Learn the input-solution mapping for a given problem
- Pass inputs through the mapping for solutions
  - No iterative updates needed
  - Trade learning complexity for low run-time complexity
- **Q**: can we learn such a mapping?

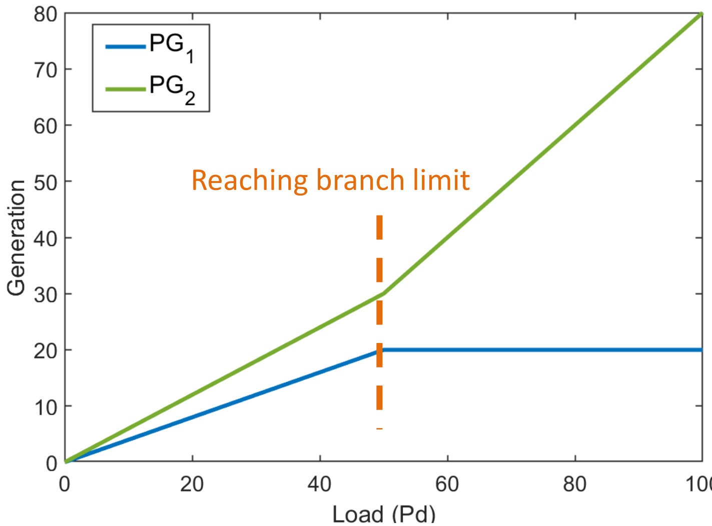
[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

Continuous Mapping upon Unique Solution

$$\min_{x} \quad f(x, z)$$
$$s.t. \quad g_i(x, z) = 0, \; i = 1, \dots . n$$
$$\qquad h_j(x, z) \le 0, \; j = 1, \dots . m$$

z: input parameter vector
x: decision variable vector

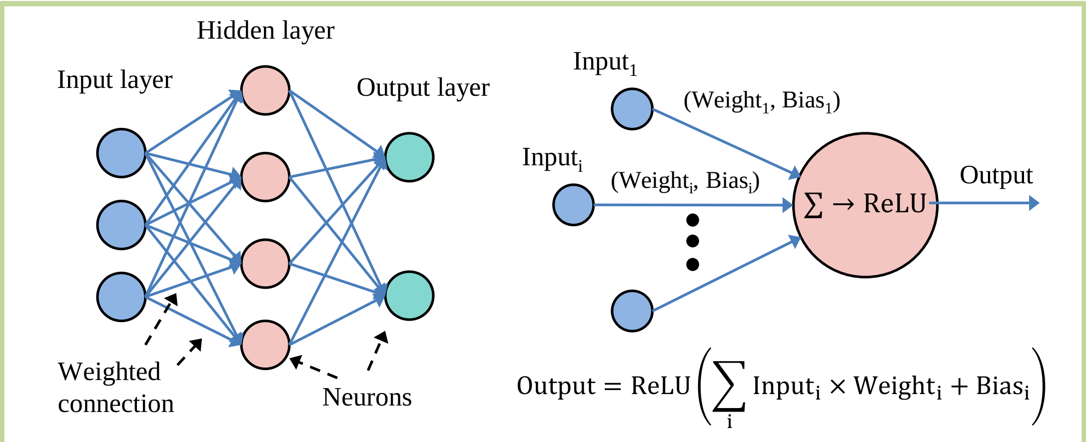
Continuous Mapping Theorem. *[1][2]*
For continuous f, g, h, if the input domain D is compact and the optimal solution x^* is unique for any $z \in D$, then the input-solution mapping $z \to x^*$ is continuous.

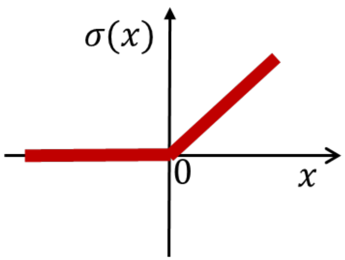


DC-OPF is a quadratic problem with unique optimal solution

[1] Maximum Theorem in Chapter 6, Section 3, *Claude Berge, "Topological Spaces". Oliver and Boyd. p. 116. (1963)*
[2] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A feasibility-optimized Deep Neural Network Approach for AC Optimal Power Flow Problems," arXiv preprint arXiv:2007.01002, 2020.

NN Can Approximate Continuous Mapping





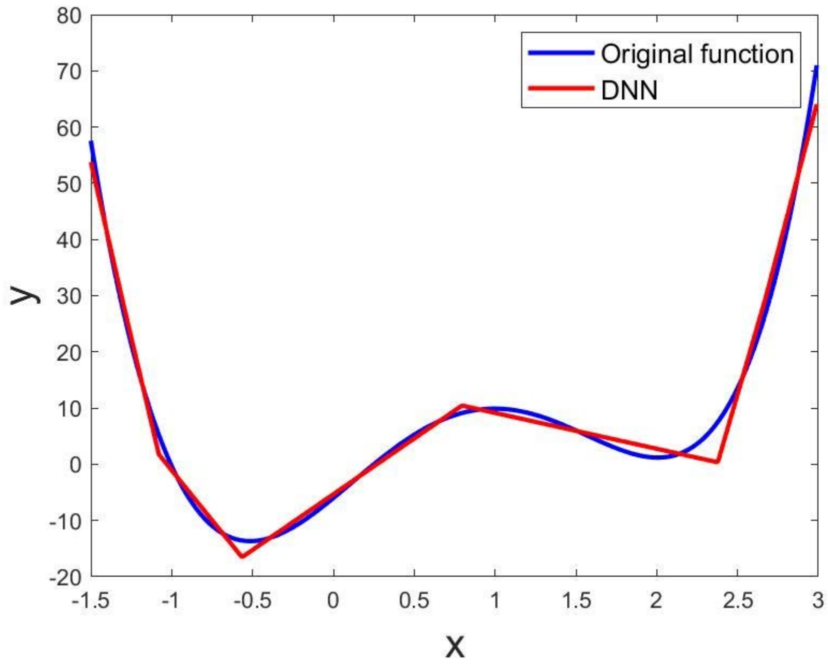
Rectified linear unit (ReLU): $\max(x, 0)$

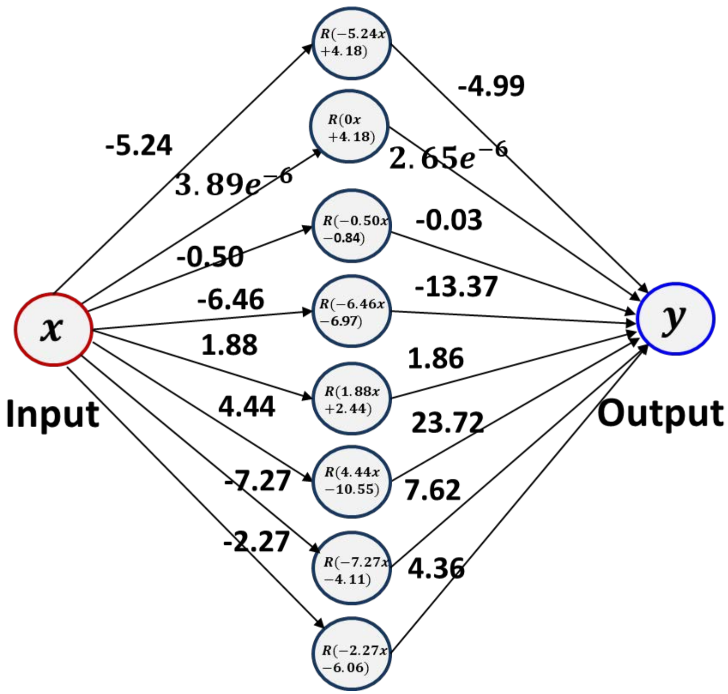
- **Theorem** [1-3]: With non-polynomial activation functions, feedforward networks can approximate "any" function/mapping arbitrarily well, with sufficient neurons.
  - Any function whose p-th power of the absolute value is Lebesgue integrable
  - Activation function is bounded, non-constant, and continuous
  - Even by using single (hidden) layer NNs

[1] K. Hornik, "Approximationcapabilitiesofmultilayerfeedforwardnetworks," Neural networks, vol.4, no.2, pp. 251–257,1991.
[2] G. Cybenko, "Approximation by superpositions of a sigmoidal function", Math. Control Signals Systems, 2(4):303–314, 1989.
[3] Pinkus, Allan. "Approximation theory of the MLP model in neural networks." Acta numerica, 8 :143-195, 1999.

An Illustrating Example

- Approximating $y = 0.3x^5 + 4.8x^4 - 18.8x^3 + 6x^2 + 23.6x - 6$ by an 8-node single-layer ReLU NN





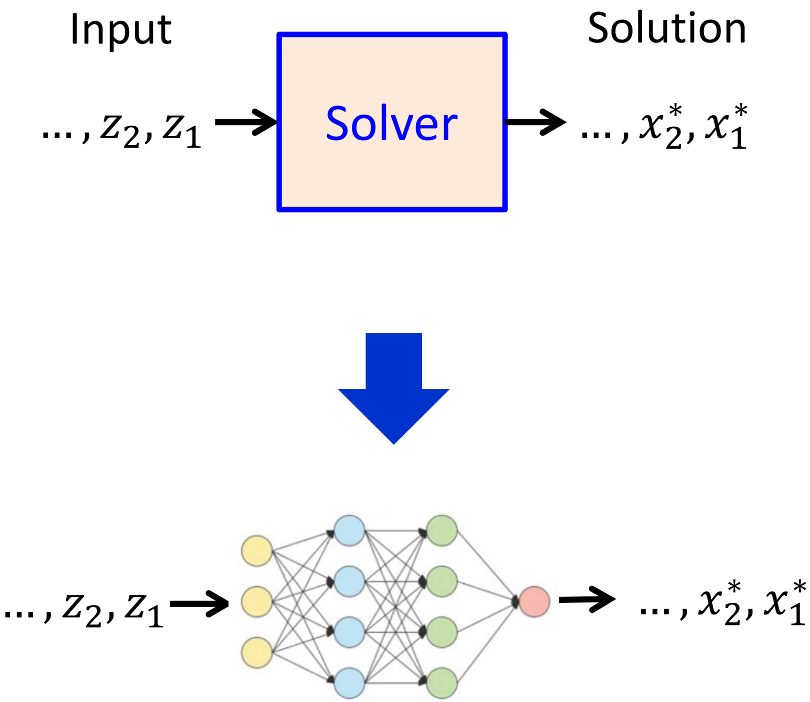
Some Recent Advances

- Results extended to RNN, CNN, ResNet, etc.

- Deep networks use exponentially less neurons [1-3]: A ReLU DNN can approximate a sufficiently smooth f up to an l_∞ error ϵ, with both width and depth at most $\log 1/\epsilon$
  - To approximate a function in a wide family of twice-differentiable functions, a DNN needs at least a width of $\mathrm{poly}(1/\epsilon)$ if the depth is fixed [3]

[1] D. Yarotsky, "Error bounds for approximations with deep ReLU networks", Neural Network, vol 94, pp. 103-114, Oct. 2017.
[2] S. Liang and R. Srikant, "Why Deep Neural Networks for Function Approximation?", ICLR, 2017.
[3] I. Safran and O. Shamir, "Depth-Width Tradeoffs in Approximating Natural Functions with Neural Networks", ICML, 2017,
[4] D. Zhou, "Universality of deep convolutional neural networks", Applied and computational harmonic analysis 48 (2), 787-794, 2020.

Our Machine Learning Viewpoint



- Learn the input-solution mapping for a given problem
- Pass input through the mapping for the solution
  - Trade learning complexity for low run-time complexity
- **Yes, we can learn the input-solution mapping by NN**
  - Learning complexity is amortized if the problem is solved repeatedly, e.g., OPF

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

ML Approach: Solving CO as NN Regression

given ↓   given ↓

$$\begin{bmatrix} u_1 & \dots & u_n \\ x_1 & \dots & x_n \end{bmatrix} = \mathrm{F}\left(\begin{bmatrix} P_{d1} \\ \dots \\ P_{dn} \end{bmatrix}\right)$$

↑ to learn

- **Training**: Given the inputs and solutions, learn the mapping

given ↓

learned ↓

$$\begin{bmatrix} u_1 & \dots & u_n \\ x_1 & \dots & x_n \end{bmatrix} = \mathrm{F}\left(\begin{bmatrix} P_{d1} \\ \dots \\ P_{dn} \end{bmatrix}\right)$$

↑ to generate

- **Applying**: Given input, directly generate the solution by the mapping

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

ML for Constrained Optimization in Power System Operation

- Application in OPF
  - Facilitate conventional solvers, e.g., [1]
  - Directly generate solutions (upcoming slides)
- Other Applications
  - Frequency control, e.g., [3,4,5]
  - Network reconfiguration, e.g., [6]
  - Economic dispatch, e.g., [7]

[1] Y. Ng, S. Misra, L. A. Roald and S. Backhaus, “Statistical Learning for DC Optimal Power Flow”, in Proc. IEEE PSCC, Dublin, Ireland, Jun. 11 - 15, 2018.
[2] X. Pan, T. Zhao, and M. Chen, “Deepopf: Deep neural network for DC optimal power flow,” in Proc. IEEE SmartGridComm, Beijing, China. 2019.
[3] W. Cui and B. Zhang, "Lyapunov-Regularized Reinforcement Learning for Power System Transient Stability," in IEEE Control Systems Letters, vol. 6, pp. 974-979, 2022.
[4] W. Cui, Y. Jiang, and B. Zhang, “Reinforcement learning for optimal primary frequency control: A Lyapunov approach,” arxiv, 2021.
[5] T. Zhao, J. Wang, X. Lu, and Y. Du, “Neural Lyapunov control for power system transient stability: A deep learning-based approach,” IEEE Trans. Power Syst., vol. 37, no. 2, pp. 955–966, Mar. 2022.
[6] Y. Gao, J. Shi, W. Wang and N. Yu, "Dynamic Distribution Network Reconfiguration Using Reinforcement Learning," in Proc. IEEE SmartGridComm, Beijing, China. 2019.
[7] F. Hasan and A. Kargarian, "Topology-aware Learning Assisted Branch and Ramp Constraints Screening for Dynamic Economic Dispatch", accepted for publication in IEEE Transactions on Power Systems (early access), 2022.

Machine Learning for Solving OPF Problems: Overview

OPF for Setting System Operating Points

$$\min_u f(u)$$

$$\text{s.t.} \quad g(x, y, u) = 0$$

$$h(x, y, u) \geq 0$$

- Total generation cost
- Power balance constraints
- Physical, operational, and security constraints (e.g., line limits)

- Recall*:* The Optimal Power Flow (OPF) Problem is to determine the outputs of generators to
  - Satisfy the load in real-time (reliability)
  - Minimize the overall generation cost (efficiency)

[1] J. Carpentier, "Contribution to the economic dispatch problem," Bulletin de la Societe Francoise des Electriciens, vol. 3, no. 8, 1962.
[2] Cain M B, O'neill R P, Castillo A. History of optimal power flow and formulations. Federal Energy Regulatory Commission, 2012, 1: 1-36.

Observation

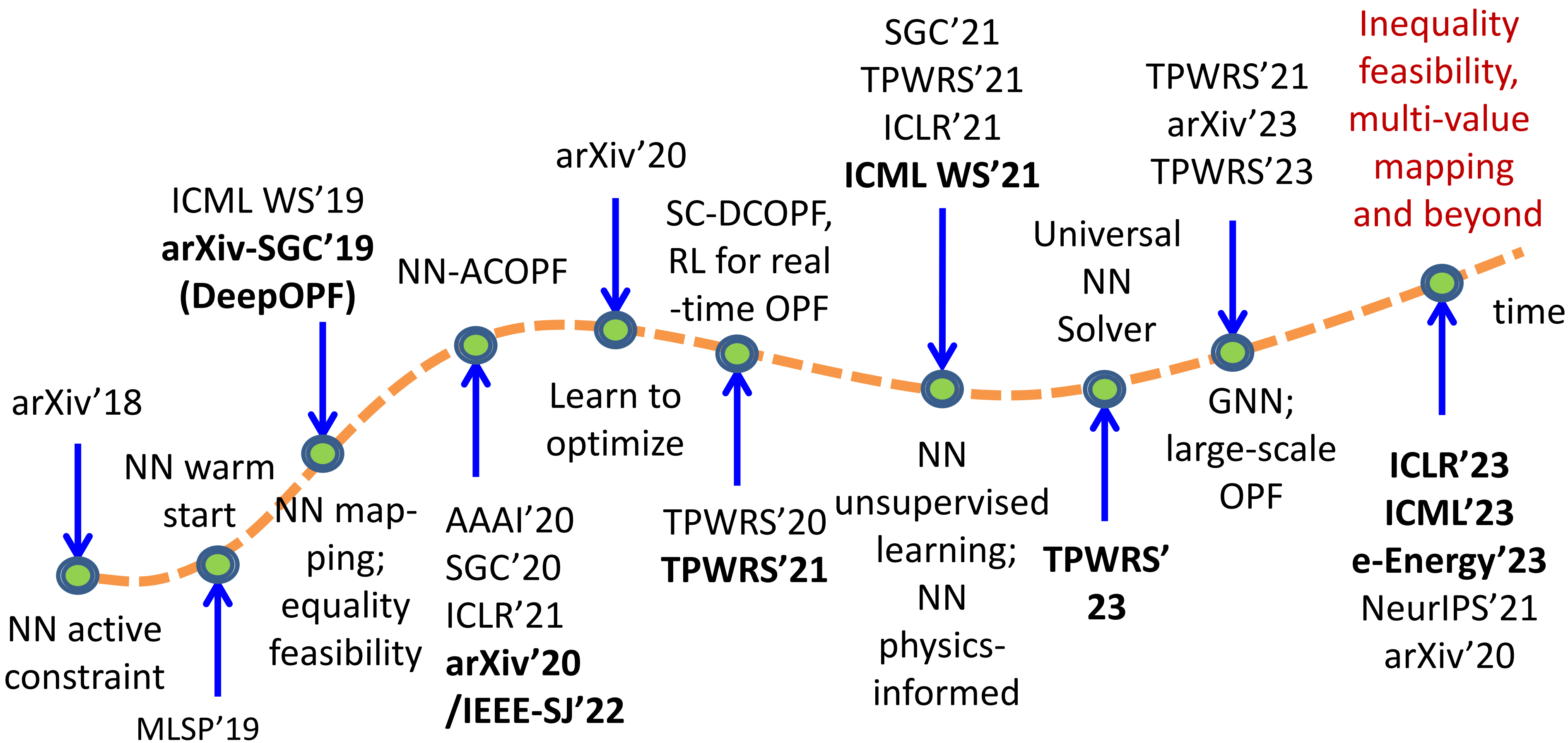
- AC-OPF problem is non-convex and NP-hard, difficult to solve in real-time
  - Practical OPF can involve more than 1M variables
- To accommodate renewable, market operators need to solve OPFs every 5 minutes
  - Previously, every half day or every 2 hours
  - In the future, every one minute
- Operators often terminate iterative methods early, or resort to solve (linearized) DC-OPF, both giving sub-optimal results
- How to solve OPF problem in real-time?

[1] Bienstock D, Verma A. Strong NP-hardness of AC power flows feasibility. Operations Research Letters, 2019.
[2] Reddy S S, Bijwe P R. Day-ahead and real time optimal power flow considering renewable energy resources. International Journal of Electrical Power & Energy Systems, 2016, 82: 400-408.

Approaches

- □ General Newton-like iterative algorithms — Not scalable
- □ Linearization to solve OPF approximately — Inaccurate
- □ Convexification to solve OPF optimality — Only applicable to special case
- □ Machine learning to solve OPF problems directly
  - Sub-percentage optimality loss (better than linearized OPF)
  - 1,5000x speedup for AC-OPF over a 2000-bus network
  - Approaches evaluated over actual RTE networks with 9,241 buses, also realistic load profiles with 40% variation
- □ Machine learning to facilitate existing iterative solvers

Historical Roadmap



- Our works in bold font
- https://energy.hosting.acm.org/wiki/index.php/ML_OPF_wiki

Wiki and Overview Webpage

- A wiki page hosted by ACM SIGEnergy
  - https://energy.hosting.acm.org/wiki/index.php/ML_OPF_wiki
- A wiki page hosted by Climate Change AI (on more general topics)
  - https://wiki.climatechange.ai/wiki/Welcome_to_the_Climate_Change_AI_Wiki
- An overview webpage by Letif Mones
  - https://invenia.github.io/blog/2021/10/11/opf-nn/
- Dataset or data generators for training NN for OPF problems
  - https://github.com/NREL/OPFLearn.jl
  - https://github.com/invenia/OPFSampler.jl/

Machine Learning for Solving DC-OPF and SC-DCOPF Problems

- X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", IEEE SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019)
- X. Pan, T. Zhao, M. Chen and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", in IEEE Transactions on Power Systems, vol. 36, no. 3, pp. 1725 - 1735, May. 2021.
- A. Velloso, P. V. Hentenryck, "Combining Deep Learning and Optimization for Preventive Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, July, 2021.
- L. Zhang, D. Tabas, B. Zhang, "An Efficient Learning-Based Solver for Two-Stage DC Optimal Power Flow with Feasibility Guarantees" arXiv:2304.01409, 2023

DC-OPF: Linearized OPF Problems

$$
\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{N}} \left(\lambda_{i,2}p_{gi}^2 + \lambda_{i,1}p_{gi} + \lambda_{i,0}\right) \\
\text{s.t.} \quad & \mathrm{B}\boldsymbol{\theta} = \boldsymbol{p}_G - \boldsymbol{p}_D, \\
& b_{ij}\left(\theta_i - \theta_j\right) \le S_{ij}^{\max}, \quad \forall (i,j)\in\mathcal{E}, \\
& \boldsymbol{P}_G^{\min} \le \boldsymbol{p}_G \le \boldsymbol{P}_G^{\max} \\
\text{var.} \quad & p_{Gi}, \theta_i, \forall i \in \mathcal{N}.
\end{aligned}
$$

Quadratic generation cost

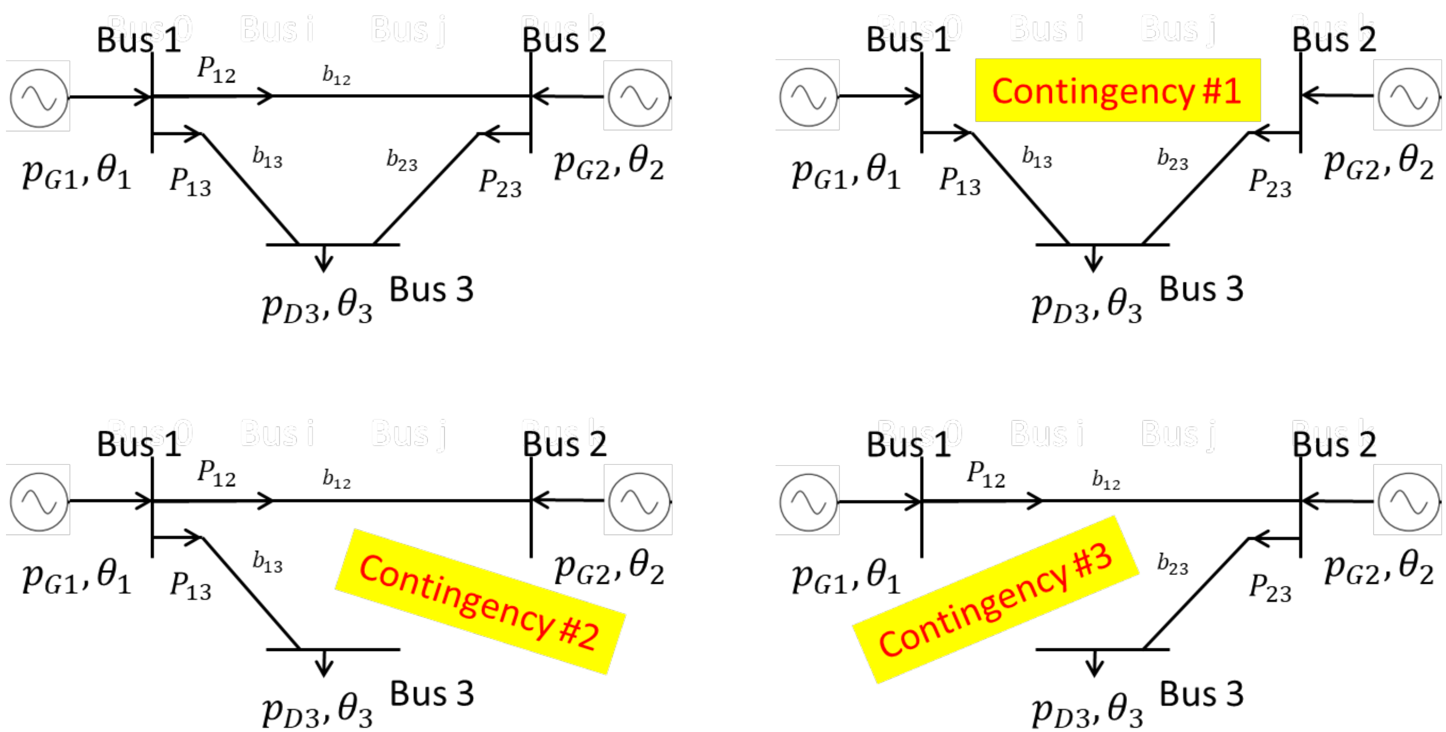
Linearized power balance equation

Branch flow and power generation limits

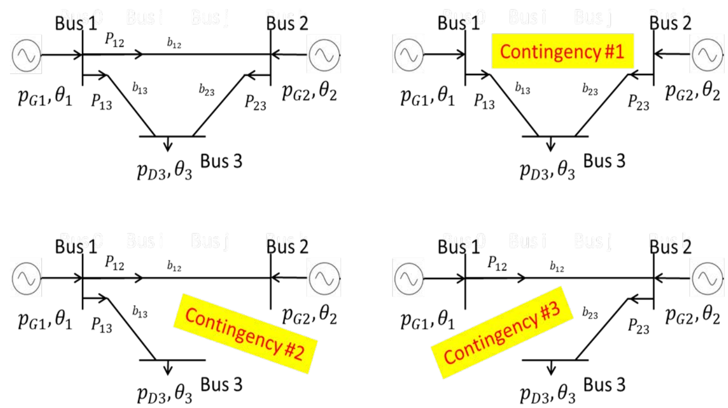
- A quadratic problem in active power generation $\boldsymbol{p}_g$ and voltage angel $\boldsymbol{\theta}$
- Easy to solve by iterative solvers. Why NN approach then?
  - Approaches generalizable to AC-OPF or other nonlinear problems
  - Security-constrained DC-OPF still challenging to solve

(N-1) Security-Constrained DC-OPF

- US operators require OPF solutions to be (N-1) secure
  - Preventive SCOPF: OPF generation still supports the load upon any single line failure (could be transient)



(N-1) Security-Constrained DC-OPF



$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{N}} \left(\lambda_{i,2} p_{gi}^2 + \lambda_{i,1} p_{gi} + \lambda_{i,0} \right) \\
\text{s.t.} \quad & \mathrm{B}^c \boldsymbol{\theta}^c = \boldsymbol{p}_G - \boldsymbol{p}_D, \ \boxed{\forall c \in \mathcal{C},} \\
& b_{ij}^c \left(\theta_i^c - \theta_j^c \right) \le S_{ij}^{\max}, \ \forall (i,j) \in \mathcal{E} \boxed{c \in \mathcal{C},} \\
& \boldsymbol{P}_G^{\min} \le \boldsymbol{p}_G \le \boldsymbol{P}_G^{\max}, \\
\text{var.} \quad & p_{Gi}, \forall i \in \mathcal{N}, \theta_i^c, \forall i \in \mathcal{N} \boxed{, c \in \mathcal{C}.}
\end{aligned}
$$

c = 0,1,2,3:
standard case: #0
contingency: #1,#2,#3

Complexity of SC-DCOPF (Quadratic Prob.)

- Solving (N-1) SC-DCOPF problems: $O(N^{12})$ time complexity
  - N is the number of buses
  - $O(N^4)$ time complexity for DC-OPF
  - For a 300-bus network, Gurobi solves an (N-1) SC-DCOPF problem in 5 seconds
    - On a quad-core (i7@3.40G Hz) workstation with 16GB RAM
  - For a 600-bus network, it would take 6.5 hours!

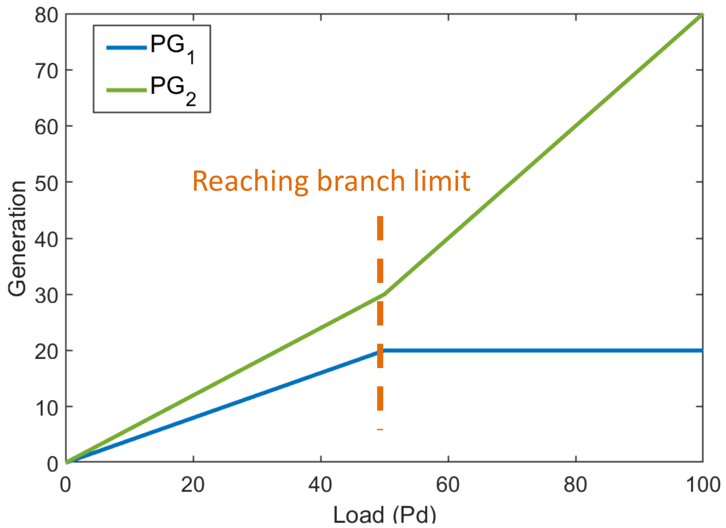
- We focus on SC-DCOPF as it is basically a large-scale DC-OPF

[1] Y. Ye and E. Tse, "An extension of karmarkar's projective algorithm for convex quadratic programming," Mathematical Programming, vol. 44, no. 1, pp. 157–179, May 1989.
[2] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

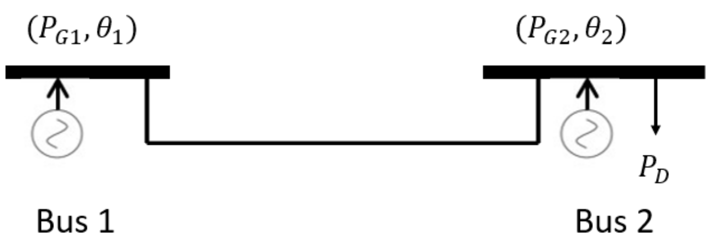
Piecewise Affine Mapping for SC-DCOPF

Theorem [1]: The input-solution mapping of a strictly convex quadratic problem is *piecewise affine*.

- $\mathbf{p}_{\mathrm{D}} \in R^{N}$ to $(\boldsymbol{p}_G, \boldsymbol{\theta}^c) \in R^{N^3}$



$$
\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{N}} \left(\lambda_{i,2}p_{gi}^2 + \lambda_{i,1}p_{gi} + \lambda_{i,0}\right) \\
\text{s.t.} \quad & \mathrm{B}^c\boldsymbol{\theta}^c = \boldsymbol{p}_G - \boldsymbol{p}_D, \quad \forall c \in \mathcal{C}, \\
& b_{ij}^c\left(\theta_i^c - \theta_j^c\right) \le S_{ij}^{\max}, \quad \forall (i,j)\in\mathcal{E}, c\in\mathcal{C}, \\
& \boldsymbol{P}_G^{\min} \le \boldsymbol{p}_G \le \boldsymbol{P}_G^{\max}, \\
\text{var.} \quad & p_{Gi}, \forall i \in \mathcal{N}, \theta_i^c, \forall i\in\mathcal{N}, c\in\mathcal{C}.
\end{aligned}
$$



A 2-bus 2-generator example

[1] N. P. Fa´ ısca, V. Dua, and E. N. Pistikopoulos, “Multiparametric linear and quadratic programming,” pp. 3–23, 2007.

Idea and Challenge

- **Idea**: learn the load-solution mapping by DNN; use it to obtain solution from input instantly

- **Challenge**:
  - We would use DNN, but how many neurons?
  - How to train the DNN?
  - Meeting equality and inequality constraints

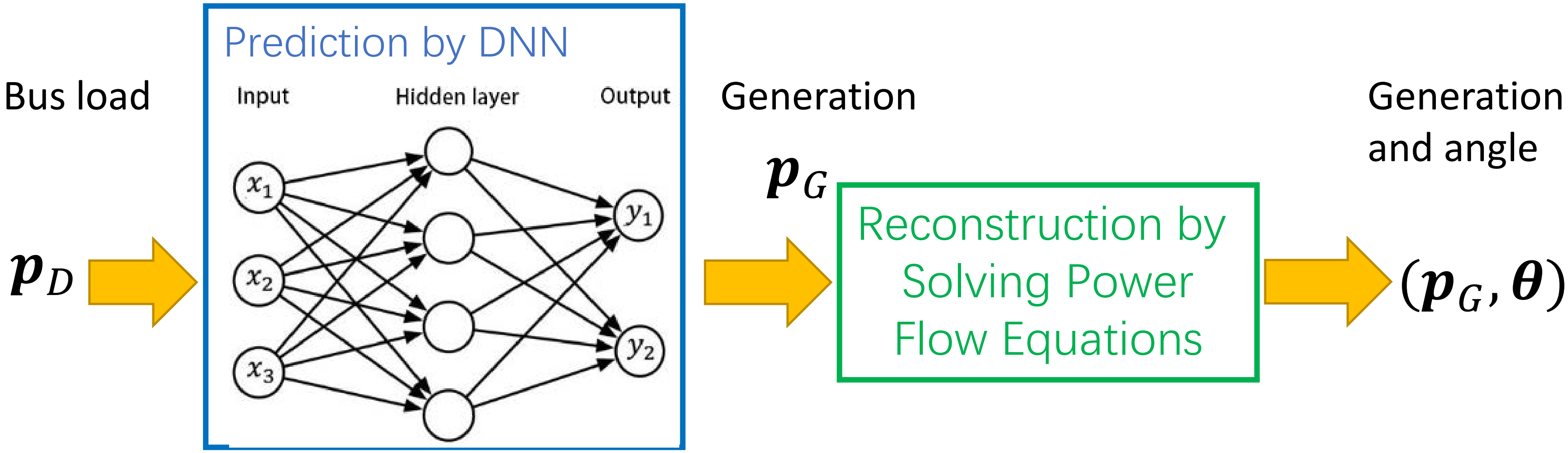
  - Standard projection back to the feasible set can be computationally expensive

Approaches for DC-OPF/SC-DCOPF

- How many neurons to have in the DNN? [1]
- Training design [1,2,3]: approximation error + constraint violation
- Meeting equality and inequality constraints
  - A Predict-and-reconstruct (PR2) approach to guarantee equality constraints [1] (also independently in [4])
  - Promoting inequality feasibilities by using penalty/KKT loss function [1,2]
  - Post-processing to recover feasible solutions [1,2]
  - A preventive-learning approach to guarantee inequality feasibility [3]
- Approaches extended to AC-OPF problems (later)

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.
[2] A. Velloso, P. V. Hentenryck, “Combining Deep Learning and Optimization for Preventive Security-Constrained DC Optimal Power Flow”, IEEE Transactions on Power Systems, July, 2021.
[3] T. Zhao, X. Pan, M. Chen, and S. H. Low, “Ensuring DNN Solution Feasibility for Optimization Problems with Convex Constraints and Its Application to DC Optimal Power Flow Problems”, arXiv preprint arXiv:2112.08091, 2021.
[4] N. Guha, Z. Wang, A. Majumdar, “Machine Learning for AC Optimal Power Flow”, ICML Climate Change workshop, June 14, 2019.

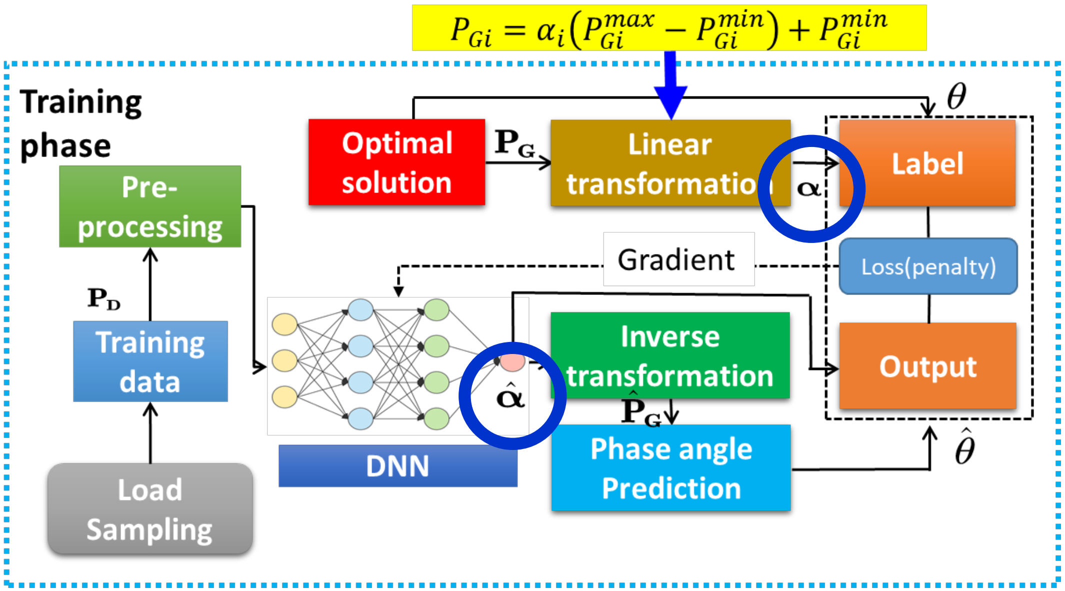
Equality Constraint: Opportunity, not Issue



- **Predict-and-Reconstruct** (PR2) [1]: predict $\boldsymbol{p}_G$ and reconstruct $\boldsymbol{\theta} = \boldsymbol{B}^{-1}(\boldsymbol{p}_G - \boldsymbol{p}_D)$ by solving power flow equations
  - Ensure power-flow balance equality constraints
  - Reduce the number of variables to predict (and thus DNN size)
  - Applicable to AC-OPF and constrained optimization problems

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

DeepOPF for SC-DCOPF: Training

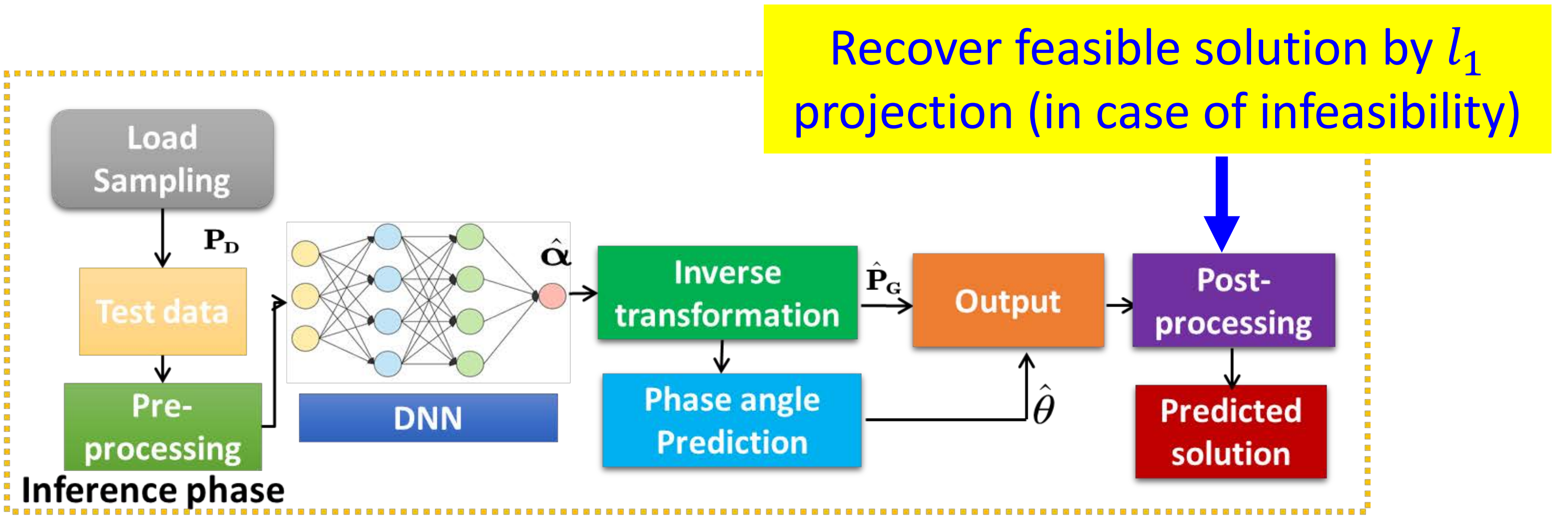


- Ensure box constraints for $\boldsymbol{p}_G$ [1]: $p_{Gi} = \alpha_i(P_{Gi}^{max} - P_{Gi}^{min}) + P_{Gi}^{min}, \alpha \in [0,1]$
- Incorporate other inequality constraint violations into the loss function:
  - $w_1 \cdot loss_{pred} + w_2 \cdot loss_{pen}$ [1]
  - Replace w_2 with dual variables, in a different SC-DCOPF formulation [2]

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in May 2021.

[2] A. Velloso, P. V. Hentenryck, “Combining Deep Learning and Optimization for Preventive Security-Constrained DC Optimal Power Flow”, IEEE Transactions on Power Systems, July, 2021.

DeepOPF for SC-DCOPF: Testing



- Output the DNN solution if feasible
- The l_1-projection problem is essentially an LP; complexity lower than solving the original QP

$$
\begin{aligned}
\min \quad & \left\| \boldsymbol{p}_G - \boldsymbol{p}_G^{DNN} \right\|_1 \\
\text{s.t.} \quad & \mathbf{B}^c \boldsymbol{\theta}^c = \boldsymbol{p}_G - \boldsymbol{p}_D, \quad \forall c \in \mathcal{C}, \\
& b_{ij}^c \left(\theta_i^c - \theta_j^c \right) \leq S_{ij}^{\max}, \quad \forall (i,j) \in \mathcal{E}, c \in \mathcal{C}, \\
& \boldsymbol{P}_G^{\min} \leq \boldsymbol{p}_G \leq \boldsymbol{P}_G^{\max}, \\
\text{var.} \quad & p_{Gi}, \forall i \in \mathcal{N}, \theta_i^c, \forall i \in \mathcal{N}, c \in \mathcal{C}.
\end{aligned}
$$

Justification of DNN Solving OPF Problems

- **Theorem** [4]: Let f^* be the piecewise-affine load-generation mapping of an (N-1) SC-DCOPF problem with a Lipschitz constant a. Then the l_∞ error of a DNN with n hidden layers and m neurons per layer
  - Decreases to 0 as m tends to infinity
  - Is lowed bounded by $a \cdot d/[4(2m)^n]$; d is the input region diameter

- Similar results for AC-OPF problems in [5]

- **Corollary**: NN width and depth to achieve an l_∞ error of ϵ satisfies

$$(2m)^n \geq a \cdot d/(4\epsilon)$$

  - Tighter than existing lower bounds (which are for general functions)

[1] D. Yarotsky, "Error bounds for approximations with deep ReLU networks", Neural Network, vol 94, pp. 103-114, Oct. 2017.
[2] S. Liang and R. Srikant, "Why Deep Neural Networks for Function Approximation?", ICLR, 2017.
[3] I. Safran and O. Shamir, "Depth-Width Tradeoffs in Approximating Natural Functions with Neural Networks", ICML, 2017,
[4] X. Pan, T. Zhao, M. Chen, and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, 2021.
[5] X. Pan, M. Chen, T. Zhao and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv preprint arXiv:2007.01002, 2020.

Run-Time Complexity of DeepOPF: $O(N^{7.5})$

- The time complexity for solving (N-1) SC-DCOPF is $O(N^{12})$

- **Proposition** [3]. The time complexity to obtain a solution for (N-1) SC-DCOPF by DeepOPF is

$$O(\underbrace{nm^2}_{\text{DNN perdition}} + \underbrace{N^5}_{\text{reconstructing the phase angles and checking feasibility}} + \underbrace{N^{7.5}}_{\text{post-processing}})$$

- In DeepOPF: $n = 3, m = O(N)$

[1] Y. Ye and E. Tse, "An extension of karmarkar's projective algorithm for convex quadratic programming," Mathematical Programming, vol. 44, no. 1, pp. 157–179, May 1989

[2] Vaidya P. Speeding-up linear programming using fast matrix multiplication, 30th Annual Symposium on Foundations of Computer Science. 1989: 332-337.

[3] X. Pan, T. Zhao, M. Chen, and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, 2021.

Performance under Typical Load

IEEE Test Case	Feasibility rate (%)	Ave. cost ($/hr)		Opt. loss (%)	Run. time (millisecond)		Speedup
		DeepOPF	Ref.		DeepOPF	Ref.	
IEEE-case30	100	225.7	225.7	<0.1	0.72	17	x24
IEEE-case57	100	9022.9	9021.6	<0.1	0.76	102	x133
IEEE-case118	100	29197.9	29149.0	<0.1	2.48	698	x281
IEEE-case300	100	156601.8	156542.5	<0.1	81.4	5766	x318

- i5-8500@3.00G Hz CPU; 8GB RAM; 50K training data, 5K testing data; baseline: Gurobi; 3-layers NN; 256/128/64 neurons; up to 95K variables in formulations

[1] X. Pan, T. Zhao, M. Chen, and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, May, 2021.

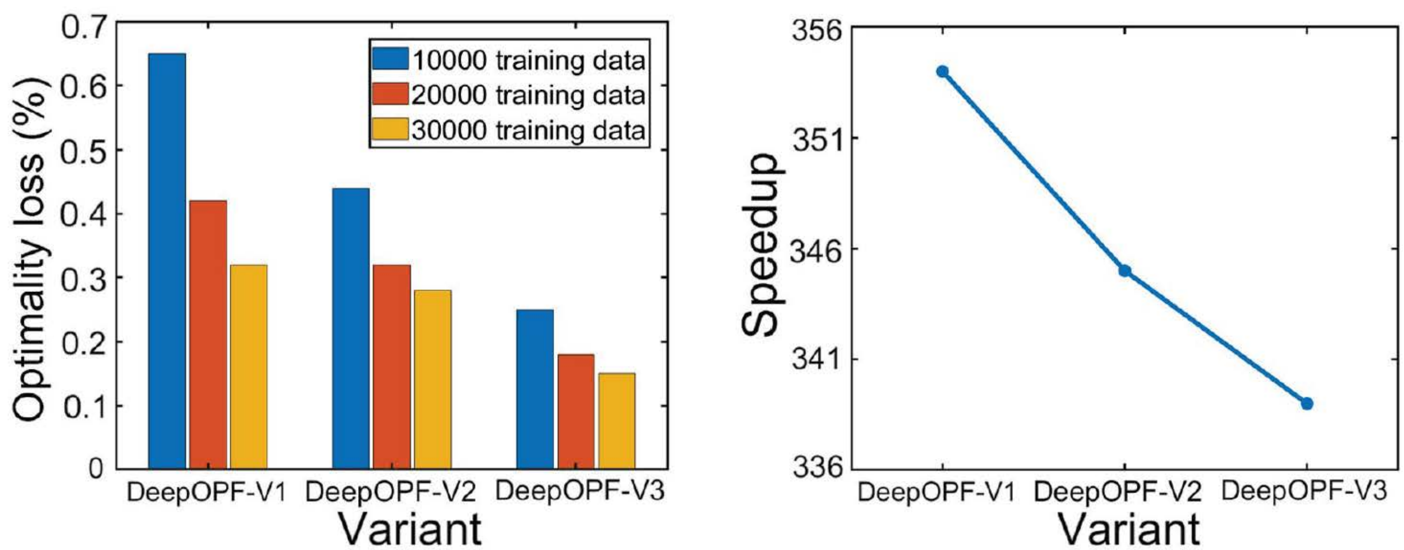
Performance with Frequent l_1-Projection

- Lightly-congested: [50%,150%] variation; 85% instances 1+ line binding
- Heavily-congested: [150%, 160%] variation; all instances 20% line binding

Scheme	Variants	Lightly-congested			Heavily-congested		
		Feasibility rate (%)	Optimality gap (%)	Speedup	Feasibility rate (%)	Optimality gap (%)	Speedup
DNN	with ℓ_1-projection	100	<0.2	56	100	<0.2	$\times 16.4$
	without ℓ_1-projection	15.7	<0.2	315	0	<0.2	–
KNN -50K	with ℓ_1-projection	100	<0.6	0.7	100	<0.3	$\times 1.5$
	without ℓ_1-projection	0	<0.9	–	0	<0.3	–

Test case: IEEE Case118

Optimality vs. Speedup (IEEE Case118)



(a) (b)

- One can trade optimality loss with speedup performance by tuning the neural network sizes
  - DeepOPF-V1/V2/V3: DeepOPF with different NN size

Summary

- DeepOPF: the first DNN scheme to solve OPF directly
  - Theoretical justification for NN to learn load-solution mapping
  - Run-time complexity $O(N^{7.5})$ for solving SC-DCOPF problems

- DeepOPF generates feasible solutions for SC-DCOPF with <0.1% optimality loss, with up to 300x speedup than Gurobi

- Approaches for ensuring/promoting solution feasibility
  - Predict-and-reconstruct (PR2) to guarantee equality constraints
  - Penalty approach promote feasibility w.r.t. inequality constraints

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019) The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in May 2021.

Machine Learning for Solving AC-OPF Problems

- X. Pan, M. Chen, T. Zhao and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv 2020. IEEE Systems Journal 2023
- G. Neel, Z. Wang and A. Majumdar, "Machine Learning for AC Optimal Power Flow", In Proceedings of the 36th International Conference on Machine Learning Workshop, Long Beach, CA, USA, 2019.
- D. Owerko, F. Gama, A. Ribeiro, Optimal Power Flow Using Graph Neural Networks, ICASSP, 2020
- W. Huang, X. Pan, M. Chen, and S. H. Low, "DeepOPF-V: Solving AC-OPF Problems Efficiently", IEEE Transactions on Power Systems, vol. 37, no. 1, pp. 800 - 803, Jan. 2022.
- X. Lei, Z. Yang, J. Yu, J. Zhao, Q. Gao and H. Yu, "Data-Driven Optimal Power Flow: A Physics-Informed Machine Learning Approach", in IEEE Transactions on Power Systems, Jan. 2021.
- F. Fioretto, T. Mak, and P. V. Hentenryck, "Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods", AAAI, 2020.
- A. Zamzam and K. Baker, "Learning Optimal Solutions for Extremely Fast AC Optimal Power Flow", IEEE SmartGridComm, 2020.
- P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", ICLR, 2021.
- W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth", In Proceedings of the 38th International Conference on Machine Learning Workshop, virtual conference, Jul. 23, 2021.
- M. Chatzos, T. W. K. Mak and P. Vanhentenryck, "Spatial Network Decomposition for Fast and Scalable AC-OPF Learning," in *IEEE Trans. on Power Systems, 2022*
- E. Liang, M. Chen and S. H. Low, "Low Complexity Homeomorphic Projection to Ensure Neural-Network Solution Feasibility for Optimization over (Non-) Convex Set“, ICML, 2023

Standard AC-OPF Formulation

- Minimizing generation cost to serve the load, with an accurate AC model

$$\min \quad \sum_{i\in\mathcal{N}_g} \left(\lambda_{i,2}P_{gi}^2 + \lambda_{i,1}P_{gi} + \lambda_{i,0}\right)$$

generation cost

$$\text{s.t.} \quad P_{gi} - P_{di} = \sum_{j:(i,j)\in\mathcal{E}} V_i V_j (g_{ij}\cos\theta_{ij} + b_{ij}\sin\theta_{ij}), \quad \forall i \in \mathcal{N},$$

$$Q_{gi} - Q_{di} = \sum_{j:(i,j)\in\mathcal{E}} V_i V_j (g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}), \quad \forall i \in \mathcal{N},$$

AC power flow equations

$$P_{gi}^{\min} \le P_{gi} \le P_{gi}^{\max}, \quad \forall i \in \mathcal{N}_g,$$

$$Q_{gi}^{\min} \le Q_{gi} \le Q_{gi}^{\max}, \quad \forall i \in \mathcal{N}_g,$$

$$V_i^{\min} \le V_i \le V_i^{\max}, \quad \forall i \in \mathcal{N}_g,$$

$$\theta_{ij}^{\min} \le \theta_{ij} \le \theta_{ij}^{\max}, \quad \forall (i,j) \in \mathcal{E},$$

Generation and voltage limit constraints

$$0 \le P_{ij}^2 + Q_{ij}^2 \le \left(S_{ij}^{\max}\right)^2, \quad \forall (i,j) \in \mathcal{E},$$

$$P_{ij} = g_{ij}\left(V_i^2 - V_i V_j \cos\theta_{ij}\right) - b_{ij} V_i V_j \sin\theta_{ij}, \quad \forall (i,j) \in \mathcal{E},$$

$$Q_{ij} = b_{ij}\left(-V_i^2 + V_i V_j \cos\theta_{ij}\right) - g_{ij} V_i V_j \sin\theta_{ij}, \quad \forall (i,j) \in \mathcal{E},$$

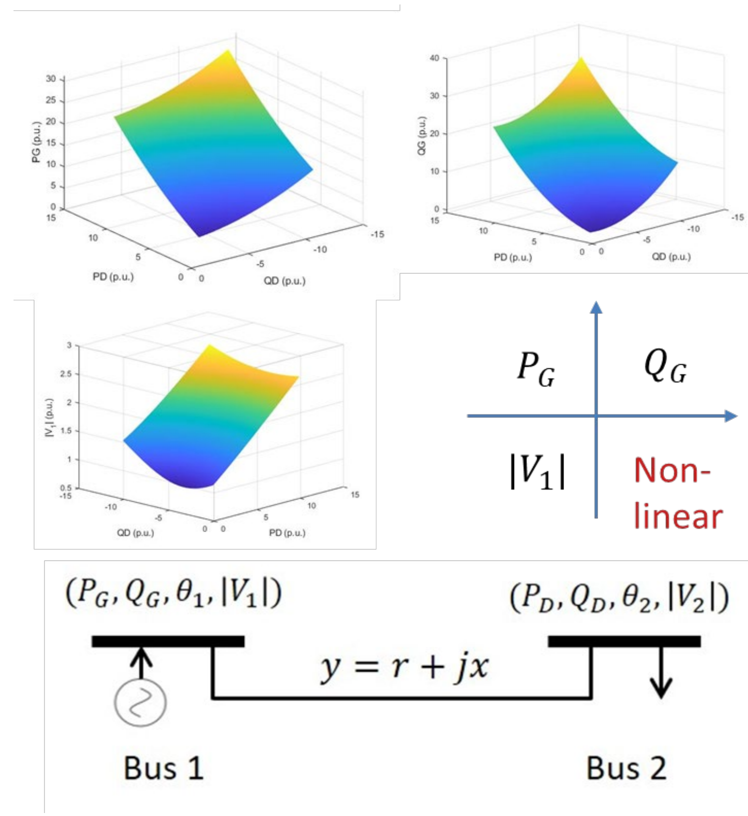
Branch flow limit constraints

$$\text{var.} \quad P_{gi}, Q_{gi}, \forall i \in \mathcal{N}_g; V_i, \theta_i, \forall i \in \mathcal{N}.$$

J. Carpentier, “Contribution to the economic dispatch problem,” Bulletin de la Societe Francoise des Electriciens, vol. 3, no. 8, pp. 431–447, 1962.

Load-Solution Mapping of AC-OPF

- **Theorem** [4]: Assumed the load domain is compact and the optimal AC-OPF solution is unique for any given load in the domain, the load-solution mapping is continuous.
  - AC-OPF has a unique solution in "typical" load regions, or radial network under certain conditions [1], or with monotonic power flow equations [2,3]



[1] S. H. Low, "Convex relaxation of optimal power flow—Parts I: Formulations and equivalence," IEEE Trans. Control Netw. Syst., vol. 1, no. 1, pp. 15–27, Mar. 2014.

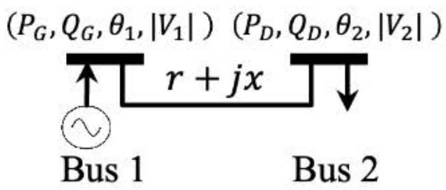
[2] Park, S., Zhang, R.Y., Lavaei, J. and Baldick, R., '' Uniqueness of power flow solutions using monotonicity and network topology,'' IEEE Transactions on Control of Network Systems, 8(1), pp.319-330, 2020

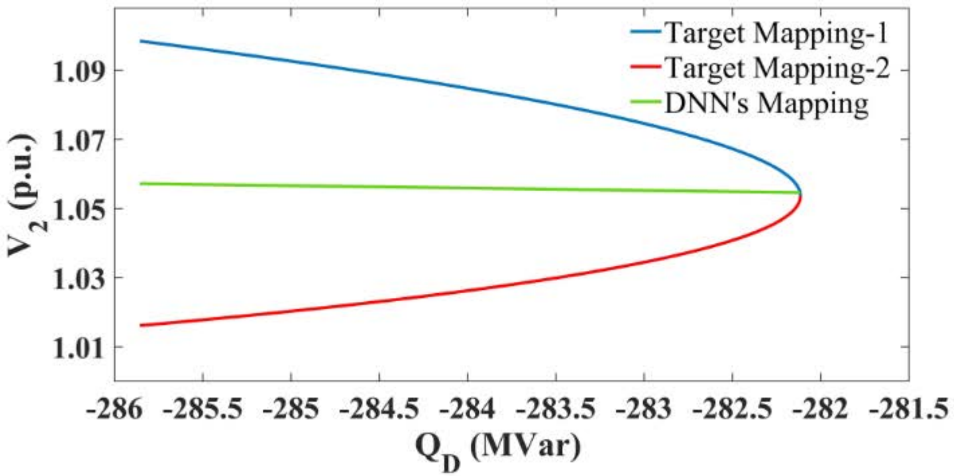
[3] Dvijotham, K., Low, S. and Chertkov, M., ''Solving the power flow equations: A monotone operator approach,'' arXiv preprint arXiv:1506.08472, 2015

[4] X. Pan, M. Chen, T. Zhao and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv preprint arXiv:2007.01002, 2020.

Multiple AC-OPF Load-Solution Mappings

- AC-OPF problem is non-convex and can admit multiple optimal or near-optimal solutions
- Supervised learning with randomly sampled load-solution pairs may fail to learn a legitimate mapping [1]





[1] X. Pan, W. Huang, M. Chen, and S. Low, "DeepOPF-AL: Augmented Learning for Solving AC-OPF Problems with Multiple Load-Solution Mappings", arXiv preprint, June 2022. https://arxiv.org/abs/2206.03365

(New) Challenge

- We would use DNN, but how many neurons?
- Meeting equality and inequality constraints
  - Predict-and-Reconstruct (PR2) approach still works, but requires solving nonlinear PF equations
  - Computing the penalty gradients is non-trivial
  - Projection is non-trivial (Part III)
- Preparing AC-OPF training data is time-consuming
- How to deal with the learnability of multiple load-solution mappings? (Part III)

Approaches for AC-OPF

- ☐ How many neurons to have in the DNN? [1]
- ☐ Training design [1-6]: supervised and unsupervised training
- ☐ Meeting equality and inequality constraints
  - – Standard and low-complexity PR2 approaches to guarantee equality constraints [1-5] (also called equality completion in [6])
  - – Computing penalty gradient by implicit function theorem and zero-order methods [1,6]
  - – Inequality feasibility guarantee in Part III

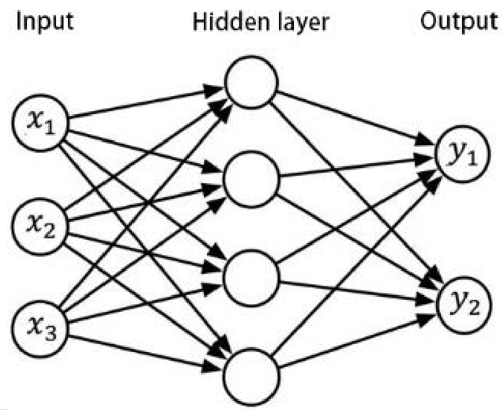
[1] X. Pan, M. Chen, T. Zhao and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv preprint arXiv:2007.01002, 2020.
[2] G. Neel, Z. Wang and A. Majumdar, "Machine Learning for AC Optimal Power Flow", In Proceedings of the 36th International Conference on Machine Learning Workshop, Long Beach, CA, USA, Jun. 10 - 15, 2019.
[3] W. Huang, X. Pan, M. Chen, and S. H. Low, "DeepOPF-V: Solving AC-OPF Problems Efficiently", IEEE Transactions on Power Systems, vol. 37, no. 1, pp. 800 - 803, Jan. 2022.
[4] F. Fioretto, T. Mak, and P. V. Hentenryck, "Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods", AAAI, 2020.
[5] A. Zamzam and K. Baker, "Learning Optimal Solutions for Extremely Fast AC Optimal Power Flow", SmartGridComm, 2020.
[6] P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", ICLR, 2021.

Predict and Reconstruct

Bus load

$\boldsymbol{p}_D$
$\boldsymbol{q}_D$

Prediction by DNN



Generation

$\boldsymbol{p}_G$
$\boldsymbol{q}_G$

Reconstruction by Solving **nonlinear AC-PF** Equations

Generation and voltage

$(\boldsymbol{p}_G, \boldsymbol{q}_G, \boldsymbol{v}, \boldsymbol{\theta})$

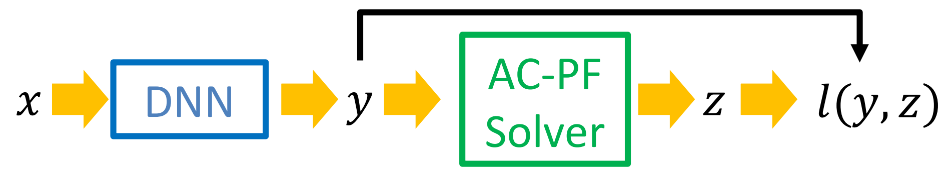
- Ensure box constraints for $\boldsymbol{p}_G, \boldsymbol{q}_G$, e.g., $p_{Gi} = \alpha_i(P_{Gi}^{max} - P_{Gi}^{min}) + P_{Gi}^{min}, \alpha \in [0,1]$; same technique as in DC-OPF and SC-DCOPF

- Incorporate inequality constraint violations into the loss function:
  - $w_1 \cdot loss_{pred} + w_2 \cdot loss_{pen}$

[1] X. Pan, T. Zhao, M. Chen, and S. Zhang, “DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow”, IEEE Transactions on Power Systems, 2021.

Obtaining Penalty Gradient for DNN Training

- Loss function:
  - $w_1 \cdot loss_{pred} + w_2 \cdot loss_{pen}$



$x \in R^{\mathrm{d}}$: load input, $y \in R^m$: independent variables, $z \in R^n$: dependent variables, l: penalty function

- Computing Penalty Gradient by the chain rule:
  - The mapping between y and z does not admit an explicit form

$$\nabla l(y) = \begin{bmatrix} \frac{\partial l(y,z)}{\partial y_1} \\ \vdots \\ \frac{\partial l(y,z)}{\partial y_m} \end{bmatrix}^T + \begin{bmatrix} \frac{\partial l(y,z)}{\partial z_1} \\ \vdots \\ \frac{\partial l(y,z)}{\partial z_n} \end{bmatrix}^T \cdot \begin{bmatrix} \frac{\partial z_1}{\partial y_1} & \cdots & \frac{\partial z_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_n}{\partial y_1} & \cdots & \frac{\partial z_n}{\partial y_m} \end{bmatrix}$$

[1] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A feasibility-optimized Deep Neural Network Approach for AC Optimal Power Flow Problems," arXiv preprint arXiv:2007.01002, 2020.

[2] P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", in Proc. ICLR, 2021.

Computing Penalty Gradient Directly

- The AC-PF equations implicitly encode the y–z mapping
  - Penalty gradient can be computed by exploring implicit function theorem[1]

Denote AC−PF equations by:
$h_i(y,z) = 0, i = 1, \dots n$

$$\begin{bmatrix} \frac{\partial h_1}{\partial y_1} & \cdots & \frac{\partial h_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial y_1} & \cdots & \frac{\partial h_n}{\partial y_m} \end{bmatrix} + \begin{bmatrix} \frac{\partial h_1}{\partial z_1} & \cdots & \frac{\partial h_1}{\partial z_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial z_1} & \cdots & \frac{\partial h_n}{\partial z_n} \end{bmatrix} \begin{bmatrix} \frac{\partial z_1}{\partial y_1} & \cdots & \frac{\partial z_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_n}{\partial y_1} & \cdots & \frac{\partial z_n}{\partial y_m} \end{bmatrix} = 0$$

$$\begin{bmatrix} \frac{\partial z_1}{\partial y_1} & \cdots & \frac{\partial z_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_n}{\partial y_1} & \cdots & \frac{\partial z_n}{\partial y_m} \end{bmatrix} = -\left(\begin{bmatrix} \frac{\partial h_1}{\partial z_1} & \cdots & \frac{\partial h_1}{\partial z_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial z_1} & \cdots & \frac{\partial h_n}{\partial z_n} \end{bmatrix} \right)^{-1} \begin{bmatrix} \frac{\partial h_1}{\partial y_1} & \cdots & \frac{\partial h_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial y_1} & \cdots & \frac{\partial h_n}{\partial y_m} \end{bmatrix}$$

[1] P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", in Proc. ICLR, 2021.

Estimating Penalty Gradient

- The penalty function is a composite function of y:

- Two-point gradient estimation [1]
  - Estimate gradient by perturbating y and computing the penalty twice
  - Better empirical performance than the implicit function theorem-based method

Inputs		Penalty
$y + \mu\delta$	➡	$l(y + \mu\delta)$
$y - \mu\delta$		$l(y - \mu\delta)$

δ: smooth parameter, m: the input dimensions,$\mu \in R^m$: a uniformly-sampled vector from the unit ball

$$\nabla l(y) \approx \frac{l(y + \mu\delta) - l(y - \mu\delta)}{2\delta} m \cdot \mu$$

[1] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A feasibility-optimized Deep Neural Network Approach for AC Optimal Power Flow Problems," arXiv preprint arXiv:2007.01002, 2020.

Simulation Settings

- Test cases: IEEE 30-/118-/300-bus and a synthetic 2000-bus mesh power network [1]

#Bus	#P-V Bus	#P-Q bus	#Branch	#Hidden layers	#Neurons per layer
30	5	24	41	2	64/32
118	53	63	231	2	256/128
300	68	231	411	2	512/256
2000	177	1822	3693	2	2048/1024

- Workstation: CentOS 7.6 with quad-core (i7-3770@3.40G Hz) CPU and 16GB RAM

- Datasets: (i) synthetic dataset with $\pm$10% variation; (ii) California demand curve with up to 40% variation; 10,000 training samples and 2,500 for testing

- Schemes: DeepOPF(-AC), Pypower, DNN-warm start [2], DNN-E [3]

[1] Powergrid Lib 2000-bus synthetic test case," 2022, https://electricgrids.engr.tamu.edu/electric-grid-test-cases/activsg2000/
[2] W. Dong, Z. Xie, G. Kestor, and D. Li, "Smart-PGSim: Using Neural Network to Accelerate AC-OPF Power Grid Simulation," in Proc. SC20, St. Louis, MO, USA, 2020
[3] A. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow, IEEE SmartGridComm, 2020.

Simulations for Realistic Load w. 40% Variation

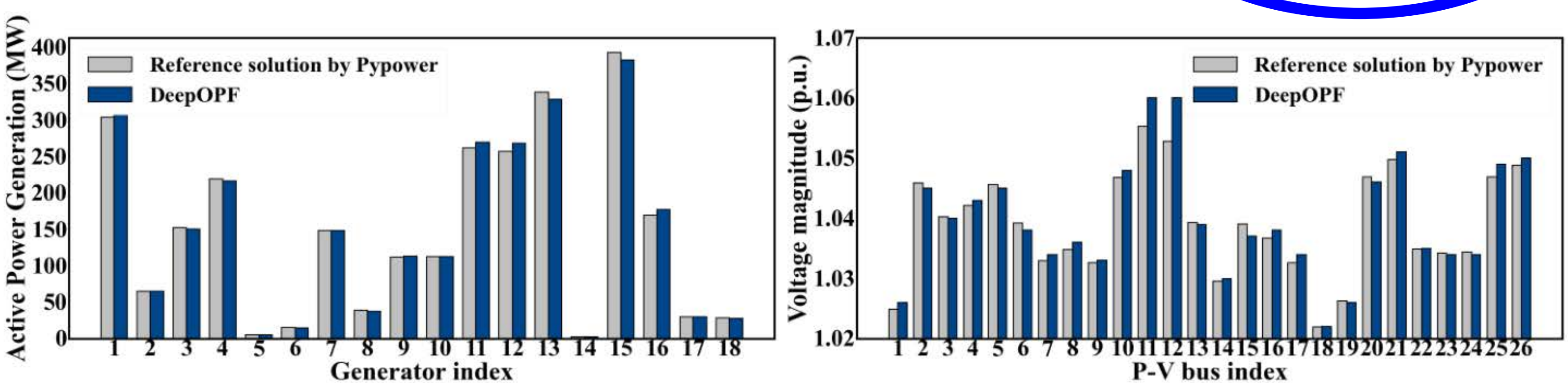
Test case	Feasibility rate (%) before feasibility-recovery*			Average cost difference (%)			Average speedup		
	DNN-E	DNN-W	DeepOPF	DNN-E	DNN-W	DeepOPF	DNN-E	DNN-W	DeepOPF
IEEE Case30	36	100	100	< 0.1	0	< 0.1	$\times 7.3$	$\times 1.0$	$\times 13$
IEEE Case118	80	100	99	< 0.1	0	< 0.1	$\times 11$	$\times 1.1$	$\times 12$
IEEE Case300	49	100	100	< 0.2	0	< 0.2	$\times 16$	$\times 1.7$	$\times 33$
IEEE Case2000	60	100	100	< 0.2	0	< 0.2	$\times 44$	$\times 0.9$	$\times 70$

- Speedups are higher for DNN-E and DeepOPF than DNN-W
- DeepOPF for AC-OPF achieves a speedup lower than for DC-OPF due to solving nonlinear AC-PF vs linear DC-PF in PR2
  - IEEE Case300: x33 for AC-OPF and x135 for DC-OPF (x318 for SC-DCOPF)

Simulations for Synthetic Load: ±10% Variation

- Speedup higher than realistic loads with 40% variation

Test case	Feasibility rate (%) before feasibility-recovery*			Average cost difference (%)			Average speedup		
	DNN-E	DNN-W	DeepOPF	DNN-E	DNN-W	DeepOPF	DNN-E	DNN-W	DeepOPF
IEEE Case30	42	100	100	<0.1	0	<0.1	×12	×1.1	×24
IEEE Case118	22	100	100	<0.1	0	<0.1	×4.2	×1.3	×22
IEEE Case300	21	100	99	<0.1	0	<0.1	×5.4	×1.8	×20
IEEE Case2000	29	100	99	<0.1	0	<0.1	×24	×1.0	×123



Comparison of DeepOPF and Pypower solutions for IEEE Case118 test case

Observation

- DeepOPF speedups AC-OPF solving time by ~100x with <0.2% optimality loss, over a 2000-bus system
  - PR2 with a penalty approach can guarantee equality constraints and promote inequality constraint feasibility
- Limitation #1: The speedup is lower than DC-OPF
  - The PR2 design requires solving nonlinear AC-PF
- Limitation #2: Preparing training data for AC-OPF is time-consuming
- Limitation #3: training complexity is high for large-scale AC-OPF problems

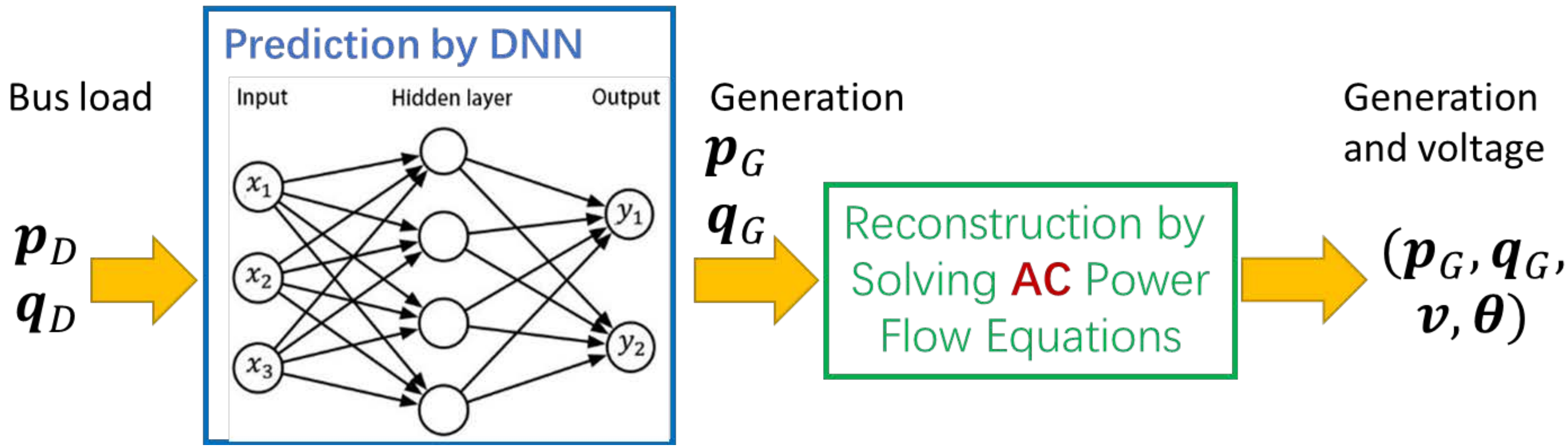
Further Improving Speedup

- Approach #1: avoid PR2; predict the generation and voltage solutions directly e.g., [2]
  - Signficant speedup
  - Solutions do not respect equality constraints
  - Projection to recover feasibility is computationally expensive

- Approach #2: alterative PR2 design for better speedup [1]

[1] W. Huang, X. Pan, M. Chen, and S. H. Low, “DeepOPF-V: Solving AC-OPF Problems Efficiently”, IEEE Transactions on Power Systems, Jan. 2022.
[2] F. Fioretto, T. Mak and P. V. Hentenryck, "Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods", AAAI, 2020.

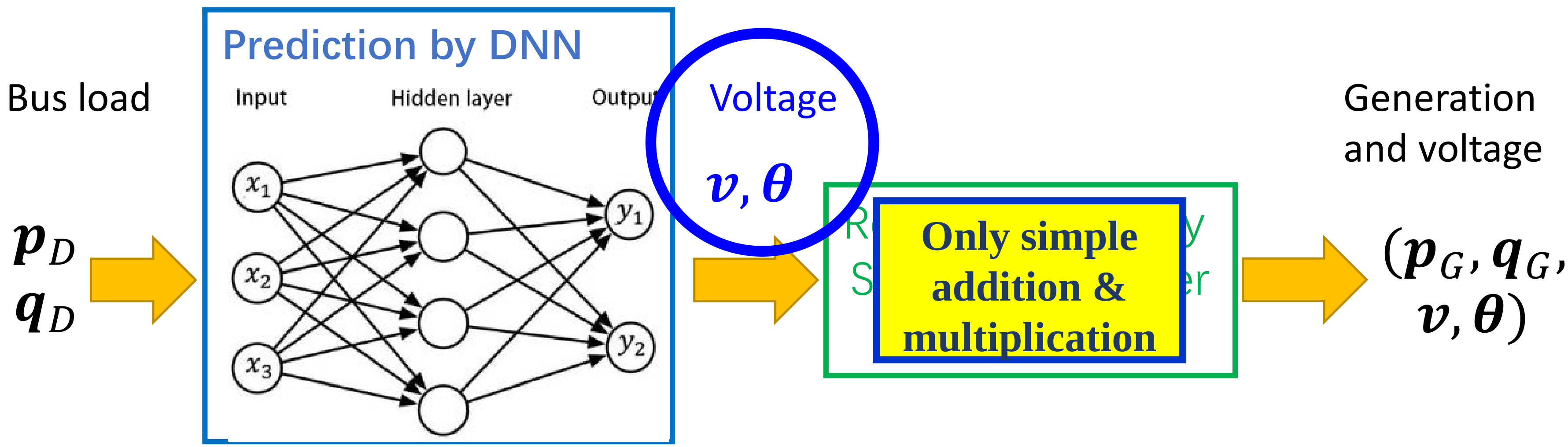
Predict and Reconstruct (PR2) Revisit

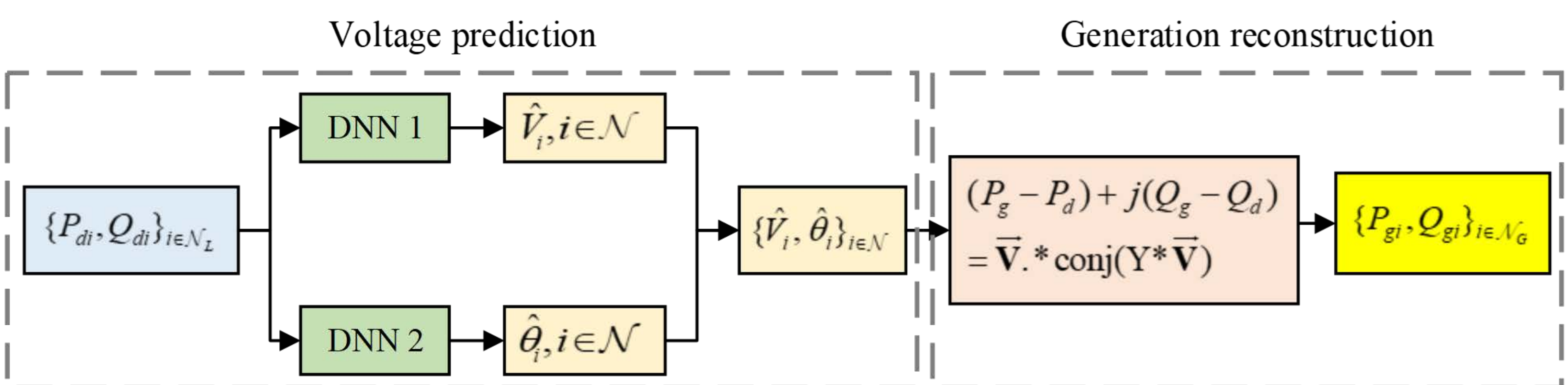


- Solving nonlinear AC power flow equations is time-consuming
- To improve speedup, we predict a different set of independent variables for efficient reconstruction

[1] X. Pan, T. Zhao, M. Chen, and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, early access, 2020.

DeepOPF-V: Low Complexity PR2 Design





[1] W. Huang, X. Pan, M. Chen, and S. H. Low, “DeepOPF-V: Solving AC-OPF Problems Efficiently”, IEEE Transactions on Power Systems, Jan. 2022

The New PR2 Design is Effective

- Speedup improves from 100+ in DeepOPF to 15,000+ in DeepOPF-V, for the 2000-bus test case

SIMULATION RESULTS IN THE 300-BUS AND 2000-BUS SYSTEMS

Metric	IEEE 300-bus system		2000-bus system	
	Before PP	After PP	Before PP	After PP
$\eta_{opt}(\%)$	0.11	0.11	0.15	0.14
$\eta_{\boldsymbol{V}}(\%)$	100.0	100.0	100.0	100.0
$\eta_{\boldsymbol{P_g}}(\%)/\eta_{\boldsymbol{Q_g}}(\%)$	99.9/99.3	100.0/99.8	100.0/100.0	100.0/100.0
$\Delta_{\boldsymbol{P_g}}$(p.u.)	0.0020	0.0020	0	0
$\Delta_{\boldsymbol{Q_g}}$(p.u.)	0.3350	0.3350	0	0
$\eta_{\boldsymbol{S_l}}(\%)$	100.0	100.0	99.71	99.71
$\Delta_{\boldsymbol{S_l}}$(p.u.)	0	0	0.0247	0.0247
$\eta_{\boldsymbol{\theta_l}}(\%)$	100.0	100.0	100.0	100.0
$\eta_{\boldsymbol{P}_d}(\%)/\eta_{\boldsymbol{Q}_d}(\%)$	99.6/99.5	99.6/99.4	99.83/99.53	99.84/99.53
t_{mips}/t_{dnn} (ms)	3213.3/1.7	3213.3/2.1	39107.8/2.7	39107.8/2.9
η_{sp}	$\times$1890	$\times$1530	$\times$16543	$\times$15374

Simulation with Realistic Load Profiles

- 42% variation in a realistic load profile with bus correlation
- DNN structure: 3 hidden layers, each with 768 neurons

SIMULATION RESULTS IN THE MODIFIED IEEE 300-BUS SYSTEM WITH REAL-TIME LOAD DATA

Metric	Before PP	After PP	
		$\boldsymbol{F}_{\theta V}^{his}$	$\boldsymbol{F}_{\theta V}$
$\eta_{opt}(\%)/\eta_{\boldsymbol{V}}(\%)$	-0.01/100.0	-0.01/100.0	-0.01/100.0
$\eta_{\boldsymbol{P_g}}(\%)/\Delta_{\boldsymbol{P_g}}$(p.u.)	99.6/0.0007	100.0/0	100.0/0
$\eta_{\boldsymbol{Q_g}}(\%)/\Delta_{\boldsymbol{Q_g}}$(p.u.)	99.8/0.0019	100.0/0	100.0/0
$\eta_{\boldsymbol{S_l}}(\%)/\eta_{\boldsymbol{\theta_l}}(\%)$	100.0/100.0	100.0/100.0	100.0/100.0
$\eta_{\boldsymbol{P_d}}(\%)/\eta_{\boldsymbol{Q_d}}(\%)$	99.90/99.90	99.95/99.94	99.95/99.94
η_{sp}	$\times$1887	$\times$1562	$\times$647

Observation

- DeepOPF can speedup AC-OPF solving time by two orders of magnitudes with <0.2% optimality loss
  - PR2 with a penalty approach can guarantee equality constraints and promote inequality constraint feasibility
- Limitation #1: The speedup is lower than DC-OPF
  - The PR2 design requires solving nonlinear AC-PF
- Limitation #2: Preparing training data for AC-OPF is time-consuming
- Limitation #3: training complexity is high for large-scale AC-OPF problems

Unsupervised Learning for AC-OPF

- Solving 10,000 AC-OPF instances on a 2742-bus system takes 3+ days [3]
  - Workstation, dual Intel 2.10GHz CPUs and 128GB RAM

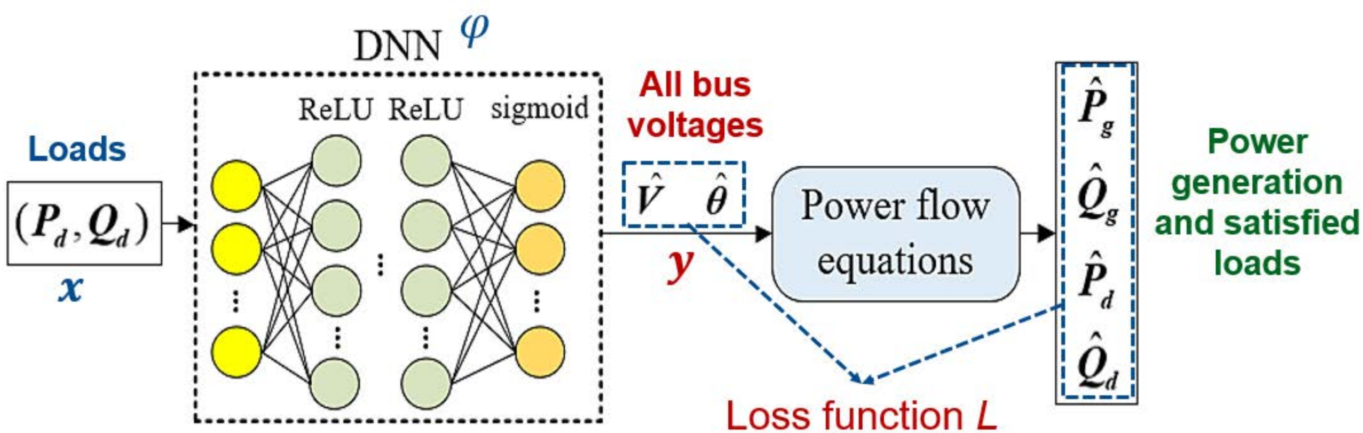
- Approach: unsupervised training [1, 2]
  - No training data ground truth (OPF solutions) needed
  - Use the OPF objective and constraint violation to guide the DNN training

[1] W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth", In Proceedings of the 38th International Conference on Machine Learning Workshop, virtual conference, Jul. 23, 2021.
[2] P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", ICLR, 2021.
[3] S. Babaeinejadsarookolaee, et al., "The power grid library for benchmarking ac optimal power flow algorithms", arXiv preprint arXiv:1908.02788, 2019.

DeepOPF-NGT: DeepOPF-V with No Ground Truth

- Use objective and constraints violation to guide DNN training

$$L = k_o \underbrace{L_o(x,\varphi)}_{\text{OPF objective}} + k_c \underbrace{L_c(x,\varphi)}_{\text{constraint violation}} + k_d \underbrace{L_d(x,\varphi)}_{\text{load mismatch}}$$



Mapping: $y = F(x, \varphi)$

[1] W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth", In Proceedings of the 38th International Conference on Machine Learning Workshop, virtual conference, Jul. 23, 2021.

Adaptive Learning Rate Adjustment

- We use the following adaptive learning rate in DeepOPF-NGT
  - At iteration t of the training, coefficient k_d^t and k_c^t are updated as

$$k_d^t = \min\{\frac{k_o \mathcal{L}_o(x, \quad \varphi)}{\mathcal{L}_d(x, \varphi)}, \overline{k}_d\}$$

$$k_c^t = \min\{\frac{k_o \mathcal{L}_o(x, \quad \varphi)}{\mathcal{L}_c(x, \varphi)}, \overline{k}_c\}$$

$\overline{k}_d$ and $\overline{k}_c$: upper bounds for penalty coefficients k_d and k_c.

- Benefit: balance the impact of different terms in the loss functions to avoid one dominates the other two
- Training time is roughly the same as supervised training

[1] W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth", In Proceedings of the 38th International Conference on Machine Learning Workshop, virtual conference, Jul. 23, 2021.

Unsupervised Learning Works

- IEEE Case118 test case; training/testing samples: 600/40,000
- Training time: 3 hrs for DeepOPF-V, 1 hr for DeepOPF-NGT, 10 min for EACOPF

Metric	DeepOPF-NGT	DeepOPF-V	DeepOPF-AC	EACOPF
$\eta_{opt}(\%)$	<0.1	<0.4	<0.3	<6.0
$\eta_{V}(\%)$	-	-	99.81	96.57
$\eta_{P_g}(\%)$	100.00	100.00	100.00	99.20
$\eta_{Q_g}(\%)$	100.00	99.98	100.00	100.00
$\eta_{S_l}(\%)$	99.28	100.00	100.00	99.46
$\eta_{\theta_l}(\%)$	100.00	100.00	100.00	100.00
$\eta_{P_d}(\%)$	99.93	99.91	-	-
$\eta_{Q_d}(\%)$	99.80	99.70	-	-
η_{S_p}	x1e3	x1e3	x1e2	x1e2

[1] X. Pan, M. Chen, T. Zhao, and S. H. Low, “DeepOPF: A feasibility-optimized Deep Neural Network Approach for AC Optimal Power Flow Problems,” arXiv preprint arXiv:2007.01002, 2020.
[2] A. Zamzam and K. Baker, “Learning optimal solutions for extremely fast AC optimal power flow, IEEE SmartGridComm, 2020.

Ground Truth Data Help

- A small amount of ground truth data can be exploited in training to further improve the performance

Metric	DeepOPF-NGT		DeepOPF-SSL				
N_{label}	0		50	100	150	200	250
Epoch	10000	3000	3000				
$\eta_{opt}(\%)$	0.33	-0.65	-1.69	-0.44	-0.37	-0.33	-0.02
$\eta_{P_g}(\%)$	99.75	99.39	98.03	99.49	96.37	99.22	99.99
$\eta_{Q_g}(\%)$	99.90	99.96	99.25	99.78	99.12	99.99	99.96
$\eta_{S_l}(\%)$	99.92	99.76	99.89	99.97	99.80	100.00	100.00
$\eta_{\theta_l}(\%)$	100.00	100.00	100.00	100.00	100.00	100.00	100.00
$\eta_{P_d}(\%)$	99.79	98.73	97.61	99.10	98.58	99.24	99.51
$\eta_{Q_d}(\%)$	99.44	99.99	97.17	98.13	98.01	99.06	99.38
η_{S_p}	1048	1042	934	947	977	1047	1058

Observation

- DeepOPF can speedup AC-OPF solving time by two orders of magnitudes with <0.2% optimality loss
  - PR2 with a penalty approach can guarantee equality constraints and promote inequality constraint feasibility
- Limitation #1: The speedup is lower than DC-OPF
  - The PR2 design requires solving nonlinear AC-PF
- Limitation #2: Preparing training data for AC-OPF is time-consuming
- Limitation #3: training complexity is high for large-scale AC-OPF problems

High Training Complexity for Large AC-OPF

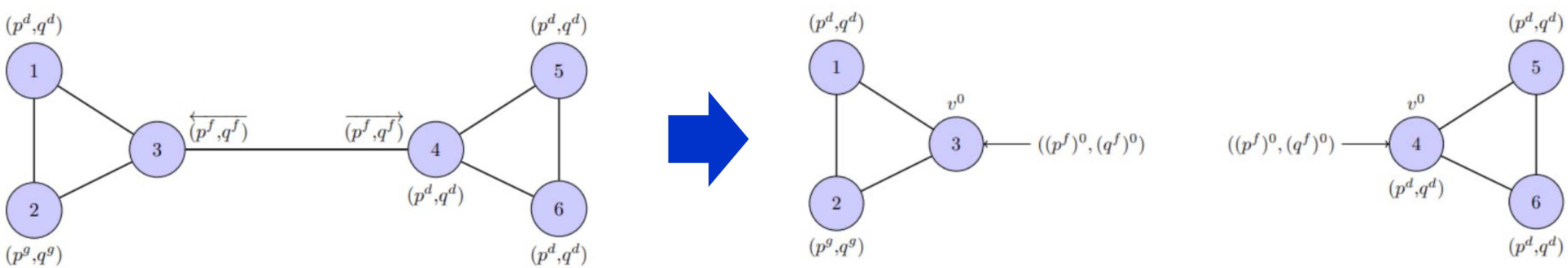
- Training DNN to solve large-scale AC-OPF problems incurs high complexity [1]
  - Large DNN output dimension: 28,180 for a 9241-bus system
  - Long training time: 7 hours for AC-OPF problems over a 3500-bus system

- Complexity may increase exponentially in the grid size

[1] M. Chatzos, T. W. K. Mak and P. Vanhentenryck, "Spatial Network Decomposition for Fast and Scalable AC-OPF Learning," in *IEEE Trans. on Power Systems, 2022,*

Grid Decomposition

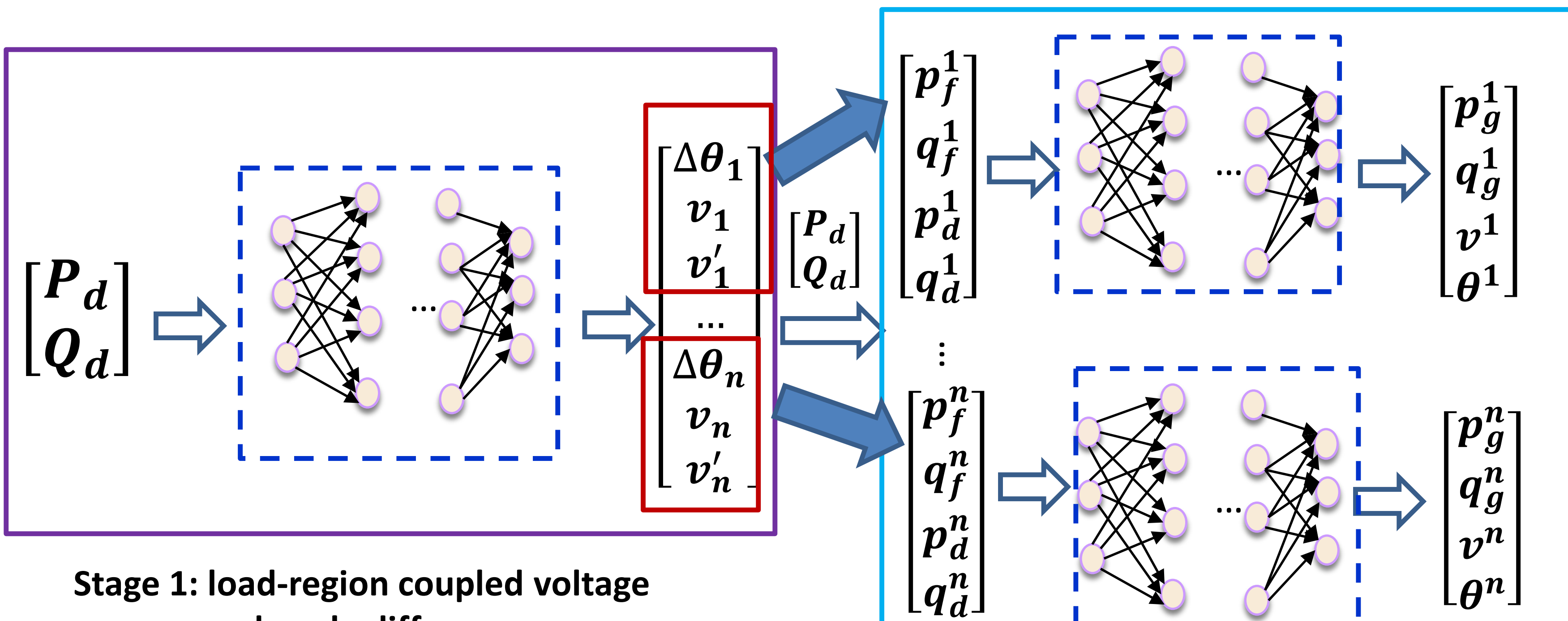
- Decompose a power grid into disjoint regions [1]
  - Regions are connected via coupling branches
  - The coupling branch flows are sufficient statistics to separate regions
- Keep the training complexity linear in the grid size



[1] M. Chatzos, T. W. K. Mak and P. Vanhentenryck, "Spatial Network Decomposition for Fast and Scalable AC-OPF Learning," in *IEEE Trans. on Power Systems, 2022*

Two-stage Learning for AC-OPF

- Use a two-stage approach to solve large-scale AC-OPF problems
  - Stage 1: predict load to coupled voltage and angle
  - Stage 2: predict (load, coupled flow) to OPF solutions in each region



Stage 1: load-region coupled voltage and angle difference

Stage 2: (load, region coupled flow)-OPF solution

Evaluation over an RTE 9241-bus Network

Speedup	Optimality gap	Feasibility rate
x10	0.03%	$> 98.5\%$

- □ 9,241 buses, 16,049 branches, 4,895 load and 1,445 generator buses
- □ Load profile: $\pm 7.75\%$ variation
- □ Training/test dataset: 8K/2K samples
- □ Training time: 30/60 minutes for $1^{st}/2^{nd}$ stage
- □ Baseline: IPOPT solver

[1] C.Josz, S. Fliscounakis, J. Maeght, P. Panciatici, “AC power flow data in MATPOWER and QCQP format: iTesla, RTE snapshots, and PEGASE”, *arXiv preprint arXiv:1603.01533*.

Summary of Using NN for AC-OPF

- Predict-and-reconstruct (PR2) works for AC-OPF [1, 3-5]
  - 15,000x over a 2000-bus network [5]
- May also predict AC-OPF solutions directly [2]
- Unsupervised learning to solve AC-OPF problems without the need of preparing AC-OPF solutions for training [4, 6]
- Grid decomposition to speedup DNN training for large problems [7]

[1] X. Pan, M. Chen, T. Zhao and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv preprint arXiv:2007.01002, 2020.

[2] F. Fioretto, T. Mak and P. V. Hentenryck, "Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods", AAAI, 2020.

[3] A. Zamzam and K. Baker, "Learning Optimal Solutions for Extremely Fast AC Optimal Power Flow", SmartGridComm, 2020.

[4] P. L. Donti, D. Rolnick and J. Z. Kolter, "DC3: a learning method for optimization with hard constraints", ICLR, 2021.

[5] W. Huang, X. Pan, M. Chen and S. H. Low, "DeepOPF-V: Solving AC-OPF Problems Efficiently", IEEE Transactions on Power Systems, vol. 37, no. 1, pp. 800 - 803, Jan. 2022.

[6] W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth", In Proceedings of the 38th International Conference on Machine Learning Workshop, virtual conference, Jul. 23, 2021.

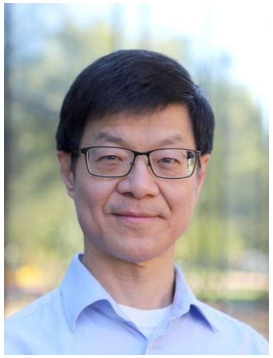
[7] M. Chatzos, T. W. K. Mak and P. Vanhentenryck, "Spatial Network Decomposition for Fast and Scalable AC-OPF Learning," in *IEEE Trans. on Power Systems, 2022*

Outline

- □ Optimal power flow (OPF) problems
- □ Example applications
- □ Recent advances and future challenges

- □ Machine learning for constrained optimization
- □ Machine learning for solving OPF problems: overview
- □ Machine learning for DC-OPF and SC-DCOPF problems
- □ Machine learning for standard AC-OPF problems

- □ Ensuring DNN feasibility for constrained optimization
- □ Graph neural network approach
- □ Solving AC-OPF with multiple load-solution mappings
- □ Concluding Remarks



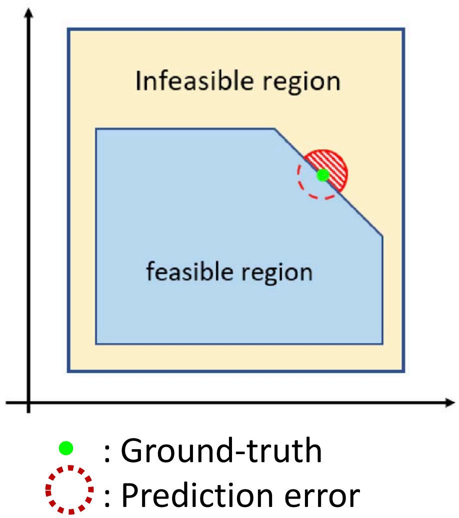
NN solution feasibility

$$\min_{u} f(u)$$

$$\text{s.t.} \quad g(x, y, u) = 0 \leftarrow \text{Equality constraints}$$

$$h(x, y, u) \geq 0 \leftarrow \text{Inequality constraints}$$

Problem, Landscape, Contributions



Existing Study	Solution Feasibility Guarantee	Bounded Optimality Loss	Low Run-Time Complexity
Penalty approach	✗	✗	✓
Projection approach	✓	✓	✗
Sampling approach	✓	✓	✗
Preventive learning	✓	✗	✓
Gauge mapping	✓	✗	✓
Homeomorphic Projection	✓	✓	✓

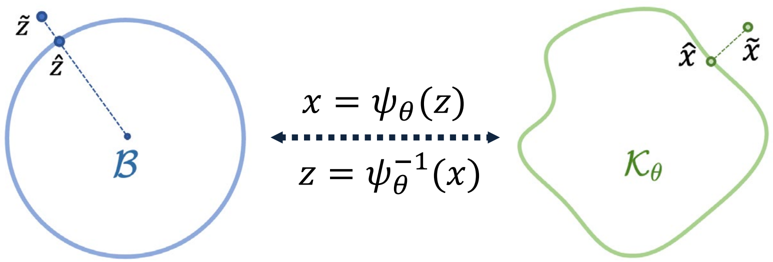
- Our homeomorphic projection [1] recovers solution feasibility with
  - Feasibility guarantee
  - Bounded optimality loss
  - Low run-time complexity

  for optimization over ball-homeomorphic set (covering all compact convex sets and any non-convex sets satisfying certain conditions)

[1] E. Liang, M. Chen, and S. H. Low, "Low Complexity Homeomorphic Projection to Ensure Neural-Network Solution Feasibility for Optimization over (Non-)Convex Set", ICML, 2023.

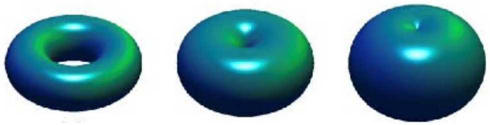
Motivation and Homeomorphism

- Projection for feasibility
  - Over a general set: **hard**
  - Over a ball: **easy**



Projection over sets “topologically equivalent” to a ball should be easy too.

- **Homeomorphic mapping**: one-to-one mapping between two sets that is continuous



[1] Geschke, S. (2012). Convex open subsets of Rn are homeomorphic to n-dimensional open balls. Hausdorff Center for Mathematics, Endenicher Allee, 62, 53115.

Ball-Homeomorphism Sets

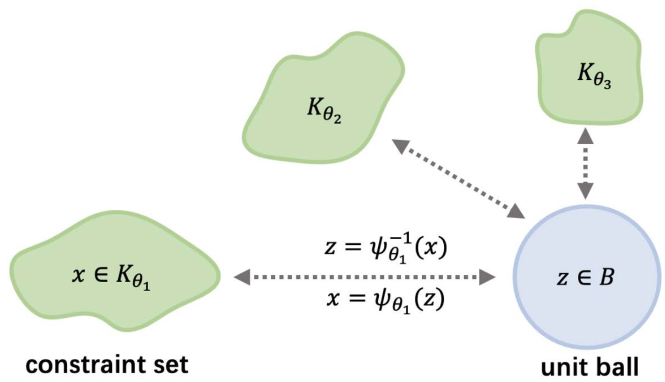
- All compact convex sets [1]
- All compact and differentiable (6 or higher)-dimension manifolds with simply-connected surface [2]
- All compact differentiable 5-dimension manifolds with boundary diffeomorphic to a 4-dimension sphere [2]
- All simply-connected sets in R^2

[1] Geschke, S. (2012). Convex open subsets of Rn are homeomorphic to n-dimensional open balls. Hausdorff Center for Mathematics, Endenicher Allee, 62, 53115.
[2] Smale, S. (1962). On the structure of manifolds. American Journal of Mathematics, 84(3), 387-399

Our Homeomorphic Projection Framework

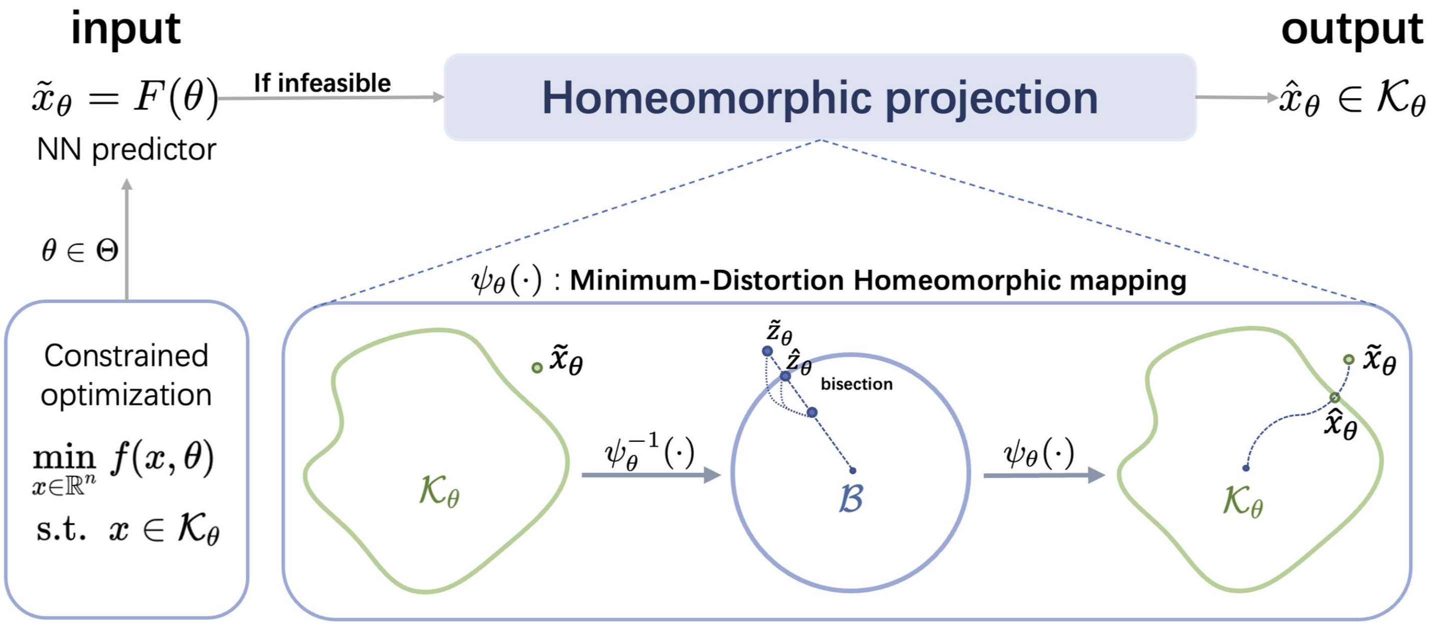
Constrained optimization

$$\min_{x \in \mathbb{R}^n} f(x, \theta)$$
$$\text{s.t.} \quad x \in \mathcal{K}_\theta$$



Setting: recover feasibility w.r.t. a ball-homeomorphic set

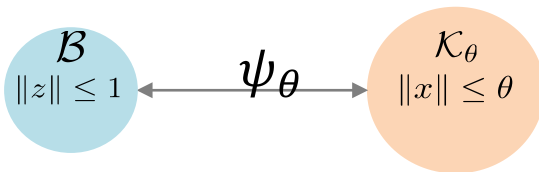
Our Homeomorphic Projection Framework



Setting: recover feasibility w.r.t. a ball-homeomorphic set

1. Learn a minimum distortion homeomorphic (MDH) mapping between the constraint set and a unit ball
2. Perform bisection over the ball so the mapped solution is feasible respect to the ball-homeomorphic constraint set

MDH Mapping

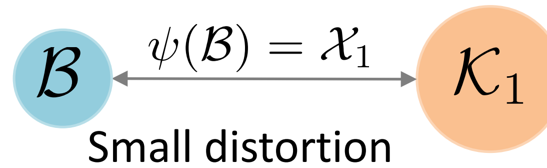


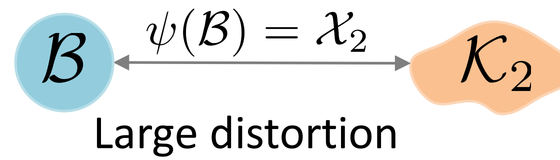
Small-distortion HM $\mathrm{D}(\psi_\theta^1) = 1$

- ψ_θ^1: $x = \theta z$

Large-distortion HM $\mathrm{D}(\psi_\theta^2) \approx 2.5$

- ψ_θ^2: $x = \theta \mathrm{R}(||z||)z$



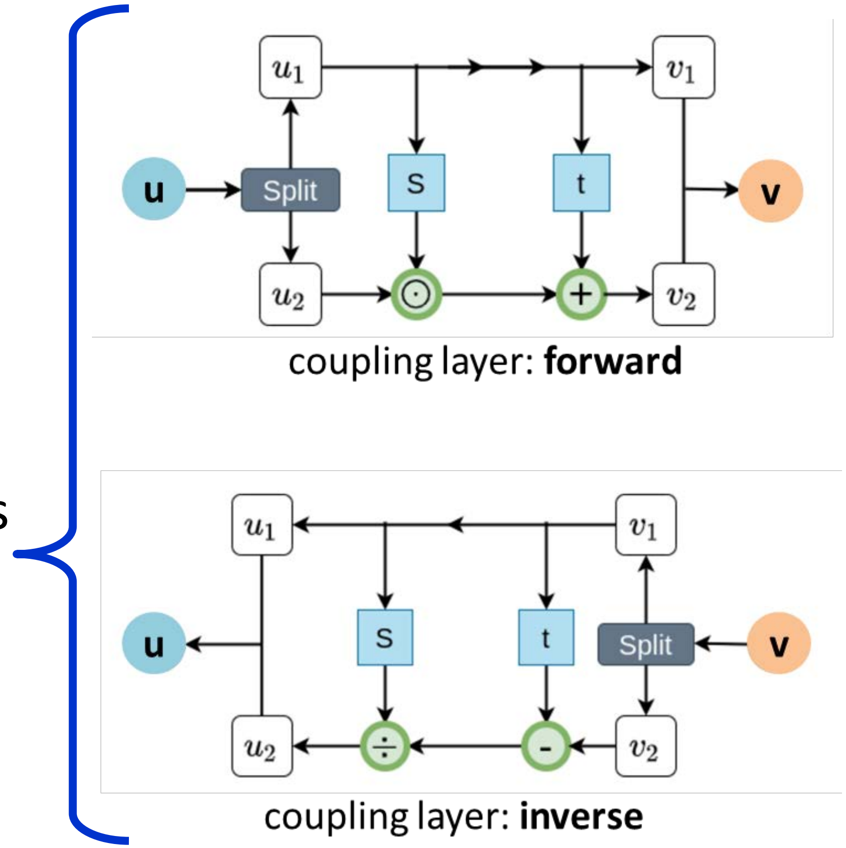


- Multiple homeomorphic mappings between two sets
- We prefer the one with minimum distortion $\mathrm{D}(\psi) = k_2/k_1 \geq 1$
  - $k_2 = \sup_{z_1, z_2}\{||\psi(z_1) - \psi(z_2)||/||z_1 - z_2||\}$
  - $k_1 = \inf_{z_1, z_2}\{||\psi(z_1) - \psi(z_2)||/||z_1 - z_2||\}$

- Different set-pairs have different minimum distortions
- Small distortion leads to minor projection-induced optimality loss

INN Can Approximate MDH Mapping

- Finding MDH mapping is hard
  - Infinite dimensional optimization
  - No closed-form in general
- **INN**: invertible NN for learning one-to-one mapping
  - Example: multiple coupling layers [13], each is an affine mapping
- INN is a universal approximator for differentiable homeo-morphic mapping [13-15]



[1] Lyu, J., Chen, Z., Feng, C., Cun, W., Zhu, S., Geng, Y., ... & Chen, Y. (2022). Universality of parametric Coupling Flows over parametric diffeomorphisms. arXiv preprint arXiv:2202.02906.
[2] Teshima, T., Ishikawa, I., Tojo, K., Oono, K., Ikeda, M., & Sugiyama, M. (2020). Coupling-based invertible neural networks are universal diffeomorphism approximators. Advances in Neural Information Processing Systems, 33, 3362-3373.
[3] Ishikawa, I., Teshima, T., Tojo, K., Oono, K., Ikeda, M., & Sugiyama, M. (2022). Universal approximation property of invertible neural networks. arXiv preprint arXiv:2204.07415.

Unsupervised INN Training

- Finding MDH mapping: $\min_{\psi_\theta \in \mathcal{H}^n} \log \mathrm{D}(\psi_\theta^{-1}, \mathcal{X}_\theta) \quad \text{s.t.} \; \mathcal{K}_\theta = \psi_\theta(\mathcal{B})$

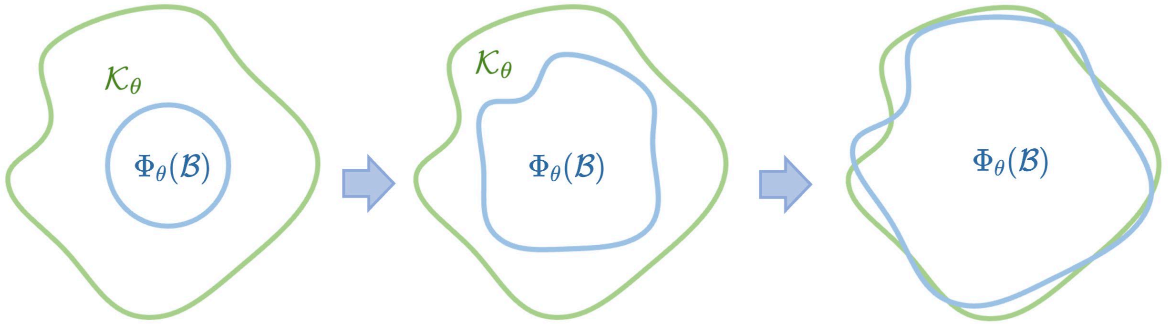
- INN loss function based on equivalent formulation and approximation $\mathcal{L}(\Phi_\theta) = \widehat{\mathrm{V}}(\Phi_\theta(\mathcal{B})) - \lambda_1 \mathrm{P}(\Phi_\theta(\mathcal{B})) - \lambda_2 \widehat{\mathrm{D}}(\Phi_\theta^{-1}, \mathcal{X}_\theta)$

  Volume maximization

  Penalty for $\Phi_\theta(\mathcal{B}) \subseteq \mathcal{K}_\theta$

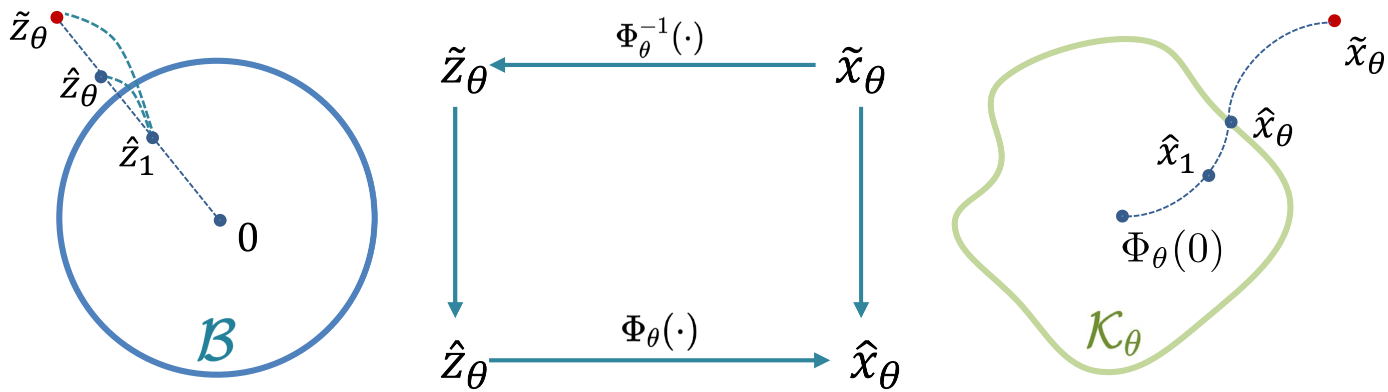
  Distortion minimization

- Training illustration



- Training **requirement**: the trained INN must be **valid,** i.e., mapping the ball center to a feasible point, i.e., $\Phi_\theta(0) \in K_\theta$

2. Bisection with Valid INN



- Given a **valid** INN and an infeasible solution
  - Step 1: map it to H-space

$$\tilde{z}_\theta = \Phi_\theta^{-1}(\tilde{x}_\theta)$$

  - Step 2: bisection for α

$$\alpha^* = \sup_{\alpha \in [0,1]} \{\Phi_\theta(\alpha \cdot \tilde{z}_\theta) \in \mathcal{K}_\theta\}$$

  - Step 3: map it back

$$\hat{x}_\theta = \Phi_\theta(\alpha^* \cdot \tilde{z}_\theta)$$

Feasibility, Optimality, & Run-Time Complexity

- **Theorem 1:** Given a valid m-layer INN and an infeasible n-dim solution, the k-step bisection will return a solution with:
  - Feasibility guarantee
  - An optimality loss bounded by $\epsilon_{\text{pre}} + \epsilon_{\text{bis}} + \epsilon_{\text{hom}}$
  - A run-time complexity of $O(kmn^2)$

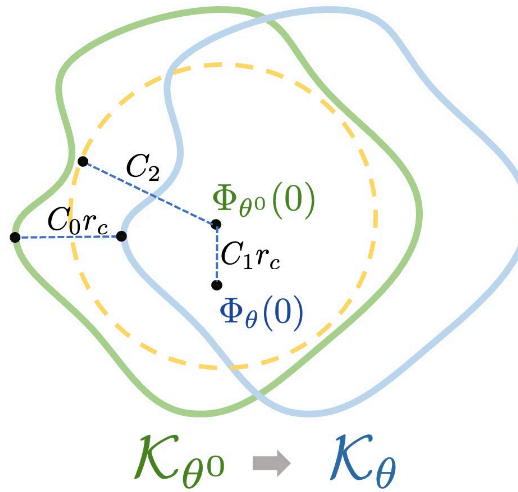
- ϵ_{pre}: NN Prediction error
- $\epsilon_{\text{bis}} = O(2^{-k})$: bisection-induced optimal loss
- $\epsilon_{\text{hom}} \leq \mathrm{D}(\Phi_\theta)(2\epsilon_{\text{inn}} + \epsilon_{\text{pre}})$: homeomorphism-induced optimality loss

- Trading run-time complexity with optimality by tuning m

Suff. Condition for Universally Valid INN

- **Theorem 2**: Consider the r_c-covering dataset $D = \{\theta_i, i = 1, \ldots, M\} \subseteq \Theta = [0,1]^d$, suppose the trained INN is valid over this training set. If $(C_0 + C_1) r_c \leq C_2$, then the INN will be valid for any input parameter, i.e., $\forall \theta \in \Theta, \Phi_\theta(0) \in \mathcal{K}_\theta$.

- A trained INN is universally-valid over the entire input region if
  - It is valid over the "dense" training set
  - The worst-case relative center-to-boundary movement is less than the conservative center-to-boundary distance



- Tune r_c to satisfy the condition (may need more samples)

Numerical Experiments

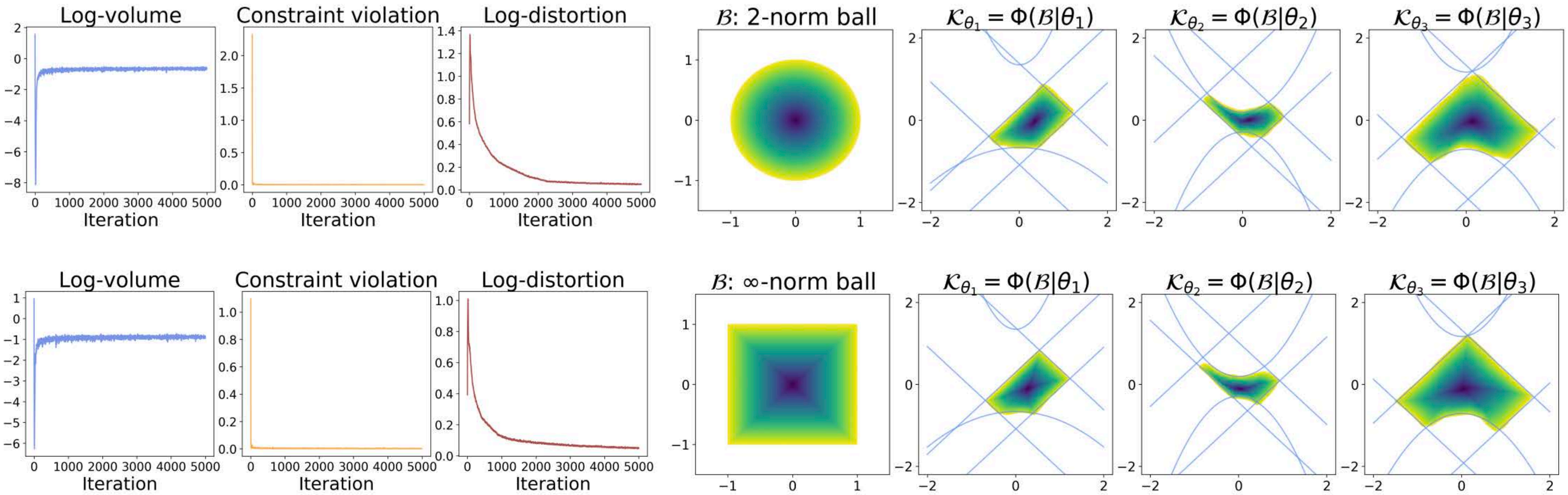
- □ Train an MDH mapping in a 2-dim toy example
  - – Evaluate the unsupervised training approach
  - – Visualize the approximated constraint sets

- □ Recover feasibility for convex and non-convex optimization problems
  - – Evaluate the feasibility, optimality, and run-time complexity of homeomorphic projection

INN Indeed Learns MDH Mappings

- Learning the MDH mapping between a unit ball and a quadratic constraint set (with different parameter θ)

$$\mathcal{K}_\theta = \{x \in \mathbb{R}^2 \mid x^\top Q x + q^\top x + b \le 0, \quad \theta = [Q, q, b]\}\}$$



Recovering Feasibility for QCQP, SDP, & AC-OPF

- 100% feasibility, minor optimality loss, substantial speedup

	Feasibility			Optimality				Speedup	
	feas. rate %	ineq. vio. 1-norm	eq. vio. 1-norm	sol. err. %	infeas. sol. err. %	obj. err. %	infeas. obj. err. %	Total ×	Post. ×
	Convex QCQP: $n = 200,\ d = 100,\ n_{\text{eq}} = 100,\ n_{\text{ineq}} = 100$								
NN	54.49	0.163	0	8.16	8.23	3.05	2.96	795657.1	–
NN+WS	100	0	0	4.41	0	1.7	0	2.1	1
NN+Proj	100	0	0	8.15	8.23	3.07	3	2.1	1
NN+D-Proj	56.54	0.023	0	8.15	8.21	3.06	2.98	10.8	4.9
NN+H-Proj	100	0	0	8.36	8.67	3.33	3.58	1618.5	738.8
	SDP: $n = 15 \times 15,\ d = 100,\ n_{\text{eq}} = 100,\ n_{\text{ineq}} = 1$								
NN	74.02	11.43	0	6.77	6.99	4.08	3.7	21440.2	–
NN+WS	100	0	0	4.96	0	3.12	0	1.5	0.4
NN+Proj	100	0	0	6.60	6.31	4.43	5.06	1.5	0.4
NN+D-Proj	87.7	5.69	0	6.76	6.94	4.08	3.7	2.6	0.7
NN+H-Proj	100	0	0	7.49	9.76	4.94	7.03	87.6	22.8
	118-node AC-OPF: $n = 344,\ d = 236,\ n_{\text{eq}} = 236,\ n_{\text{ineq}} = 452$								
NN	73.24	0.006	0	1.27	1.23	0.24	0.23	178.2	–
NN+WS	100	0	0	0.94	0	0.18	0	3.6	1
NN+Proj	100	0	0	1.55	2.31	0.24	0.23	3.8	1
NN+D-Proj	87.79	0.0001	0	1.26	1.23	0.24	0.23	4.9	1.4
NN+H-Proj	100	0	0	1.41	1.78	0.34	0.63	24.6	7.6

Summary

- **Homeomorphic projection** recovers solution feasibility with
  - Feasibility guarantee
  - Bounded optimality loss
  - Low run-time complexity

  for optimization over ball-homeomorphic set (covering all compact convex sets and any non-convex sets satisfying certain conditions)

- **Message**: ball-homeomorphism suggests a line between "easy" and "hard" projections

- Open questions
  - How to characterize INN's universal approximation capability
  - Achieving better/difference performance on optimality and complexity?

One DNN for Multiple AC-OPF Problems with Flexible Topology

- M. Zhou, M. Chen, and S. H. Low, “DeepOPF-FT: One Deep Neural Network for Multiple AC-OPF Problems with Flexible Topology”, IEEE Transactions on Power Systems, vol. 38, issue 1, pp. 964 - 967, January 2023.
- Chen Y, Lakshminarayana S, Maple C, et al. “A meta-learning approach to the optimal power flow problem under topology reconfigurations”. IEEE Open Access Journal of Power and Energy, 2022, 9: 109-120.
- M Gao, J Yu, Z Yang, J Zhao, “A Physics-Guided Graph Convolution Neural Network for Optimal Power Flow”, IEEE Transactions on Power Systems, 2023.
- S. Liu, C. Wu, and H. Zhu, “Topology-Aware Graph Neural Networks for Learning Feasible and Adaptive AC-OPF Solutions”, IEEE Transactions on Power Systems, 2023.

Motivation

- Stand-alone methods
  - Key idea: learning a mapping from load to AC-OPF optimal solutions [1] –[5]
  - Limitation: Hard to generalize to power systems with flexible network topology or line admittance
- Flexible topology and admittance in power systems
  - N-k contingency
  - Topology reconfiguration
  - Line admittance variation with temperature
- A learning-based AC-OPF solver for power systems with flexible topology and admittance is needed.

[1] X. Pan, T. Zhao, and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", in Proceedings of the 10th IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (IEEE SmartGridComm 2019), Beijing, China, October 21 - 24, 2019.
[2] X. Pan, T. Zhao, M. Chen, and S. Zhang, "DeepOPF: A Deep Neural Network Approach for Security-Constrained DC Optimal Power Flow", IEEE Transactions on Power Systems, vol. 36, issue 3, pp. 1725 - 1735, May 2021.
[3] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems", arXiv preprint arXiv:2007.01002, 2020.
[4] A. S. Zamzam and K. Baker, "Learning optimal solutions for extremely fast ac optimal power flow," in Proc. IEEE SmartGridComm, 2020.
[5] Fioretto F, Mak T W K, Van Hentenryck P. Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(01): 630-637.

Training one DNN per power network

- One alternative approach for solving AC-OPF problems with flexible topology is training one DNN per power network
  - Limitation: the computation burden is extremely high

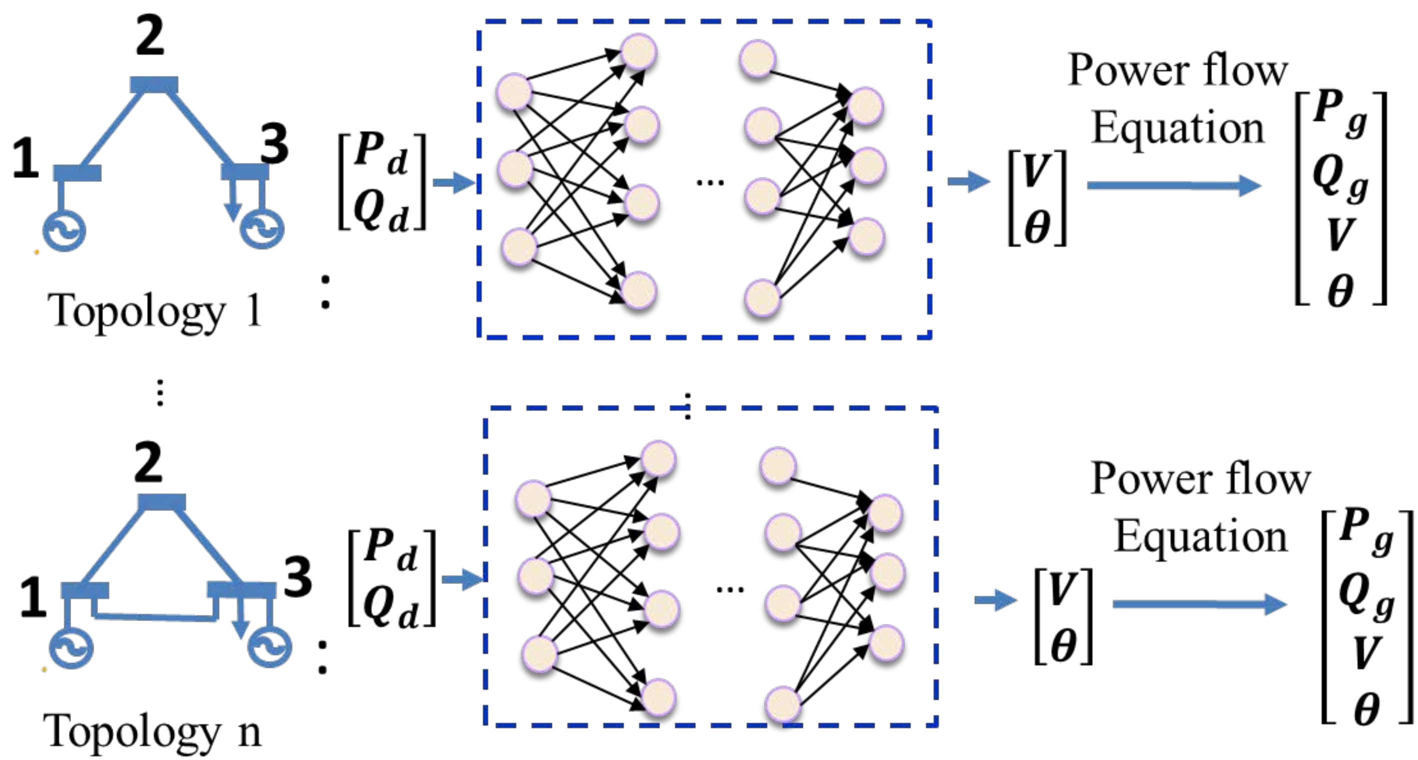


Figure 1: Schematic of training one DNN per power network

DNN re-training

- Another alternative approach for solving AC-OPF problems with flexible topology is re-training the DNN when encountering a new topology
  - Limitation: high computation burden and operation delay

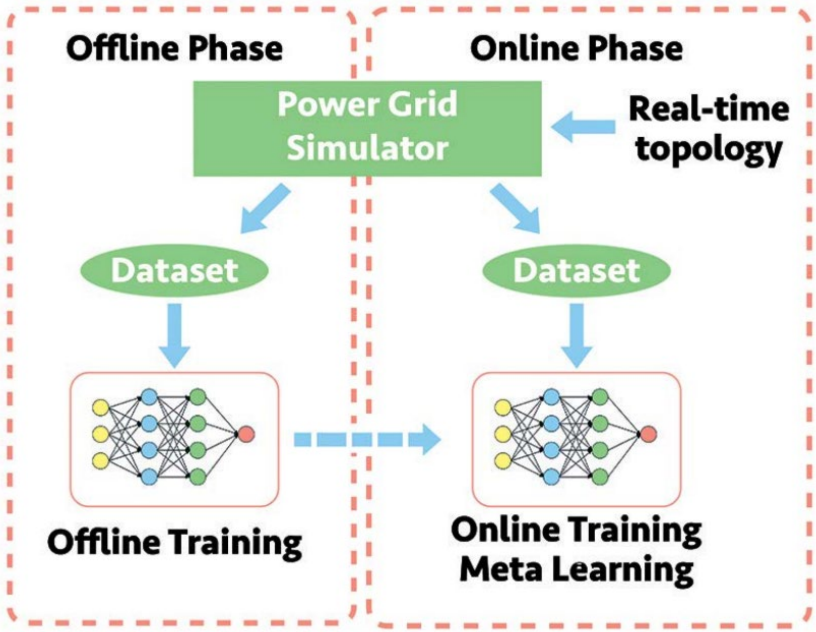
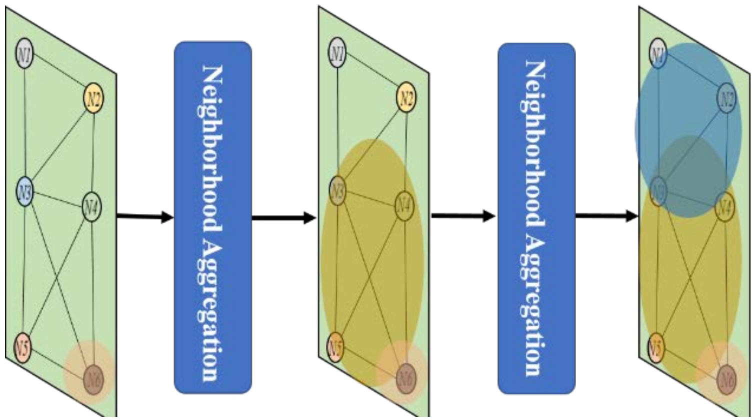


Figure 2: Schematic of DNN re-training [1]

[1] Chen Y, Lakshminarayana S, Maple C, et al. A meta-learning approach to the optimal power flow problem under topology reconfigurations[J]. IEEE Open Access Journal of Power and Energy, 2022, 9: 109-120.

GNN Approach

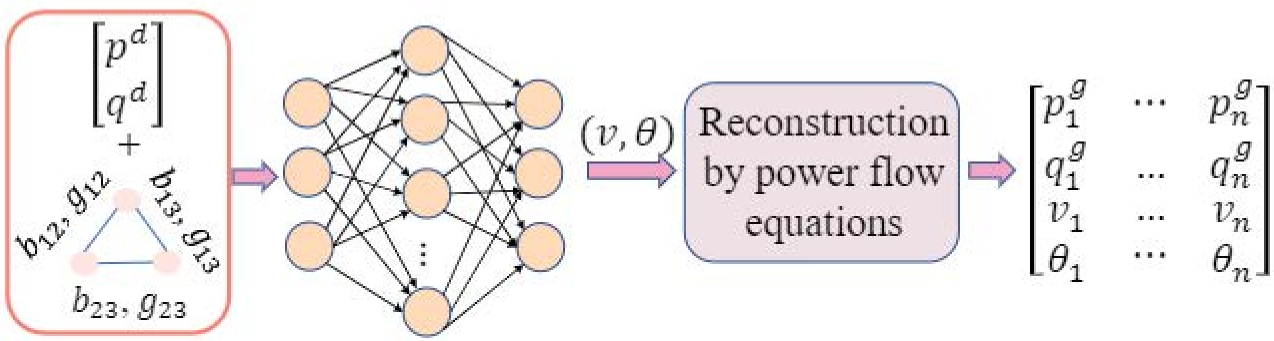


- Basic idea:
  - Each node on the NN captures a set of features of the corresponding node on the grid
  - Each iteration passes messages according to the actual connectivity
  - After K iterations output the OPF solutions
- Pros: explore the topology structure; can be robust to topology change
- Cons: no strong performance guarantee (yet)

This slide is adapted from those by M Gao, J Yu, Z Yang, J Zhao, “A Physics-Guided Graph Convolution Neural Network for Optimal Power Flow”, IEEE Transactions on Power Systems, 2023.

Embedding Training Design

- Main idea of embedded training
  - Embed the discrete network representation into the continuous admittance space
  - Use DNN to learn the mapping from (load, admittance) to bus voltages of an AC-OPF solution.



Simulation result: N-k contingency

- We test DeepOPF-FT and discrete training over the IEEE 57-bus test system in the N-4/5/6 contingency (each accounts for 1/3 of the data) with flexible topology.

Metric	DeepOPF-FT	DIS-V1 (50,000)	DIS-V2 (50,000)	DIS-V1 (150,000)	DIS-V2 (150,000)
η_{opt} (%)	0.14	-4.29	-1.31	-4.79	1.07
η_V / η_θ (%)	-	-	-	-	-
η_{Pg} (%)	95.0	94.3	93.3	97.0	96.1
η_{Qg} (%)	96.0	92.4	95.6	96.3	94.0
η_{Sl} (%)	>99.9	>99.9	>99.9	>99.9	>99.9
η_{Pd} (%)	97.2	92.7	95.2	95.8	95.8
η_{Qd} (%)	94.3	87.5	91.2	93.0	91.6
η_{sp}	x129	x130	x132	x130	x130

Simulation result: arbitrary topology

- We test DeepOPF-FT and DeepOPF-V for single topology over the IEEE 9-bus test system over arbitrary topology.

Metric	**DeepOPF-FT**		**DeepOPF-V for single topology**		
	(FT, -)	**(FT, FA)**	**(FT, -)**	**(FT, FA)**	**(-, -)**
η_{opt} (%)	0.84	0.92	94.60	95.23	-0.95
η_V / η_θ (%)	-	-	-	-	-
η_{Pg} (%)	>99.9	>99.9	53.6	53.6	100
η_{Qg} (%)	>99.9	100	97.8	97.8	>99.9
η_{Sl} (%)	>99.9	>99.9	96.6	96.3	100
η_{Pd} (%)	97.4	97.3	74.8	74.6	97.0
η_{Qd} (%)	95.3	95.0	57.0	56.8	91.8
η_{sp}	x124	x122	x88	x86	x133

Generalization

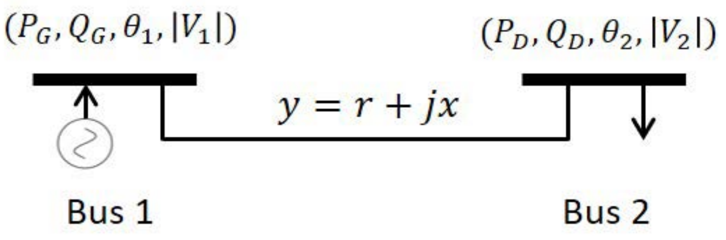
- The idea of DeepOPF-FT can be generalized to more flexible optimal power flow settings by including the corresponding parameters as DNN inputs:
  - Flexible line capacity
  - Flexible generation coefficients
  - Flexible generator capacity

Machine Learning for AC-OPF Problems with Multiple Solutions

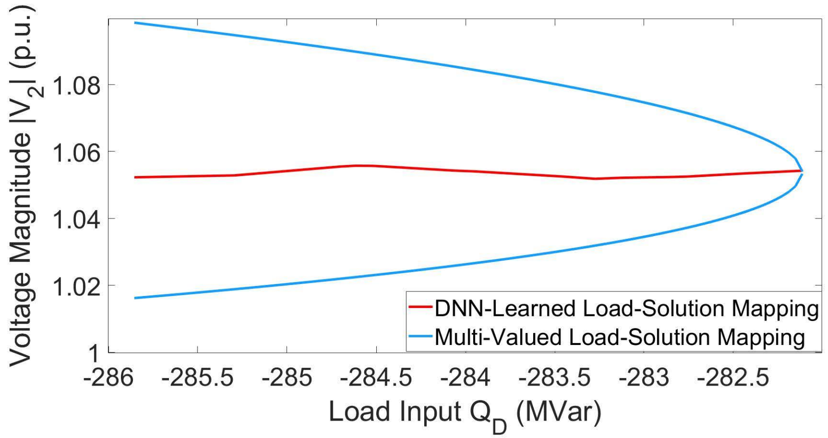
- X. Pan, W. Huang, M. Chen and S. H. Low, "DeepOPF-AL: Augmented Learning for Solving AC-OPF Problems with Multiple Load-Solution Mappings", arXiv preprint arXiv:2206.03365, 2022.
- J. Kotary, F. Fioretto, and P. Van Hentenryck, “Learning hard optimization problems: A data generation perspective,” NeurIPS 2021.

Issue of Learning Multi-Valued Mapping

- AC-OPF problems may admit a multi-valued load-solution mapping [1]
- A well-trained DNN with (standard) supervised learning fails to learn a target mapping
- Data-generation or unsupervised learning approaches [2,3] has no guarantee of learning one mapping correctly



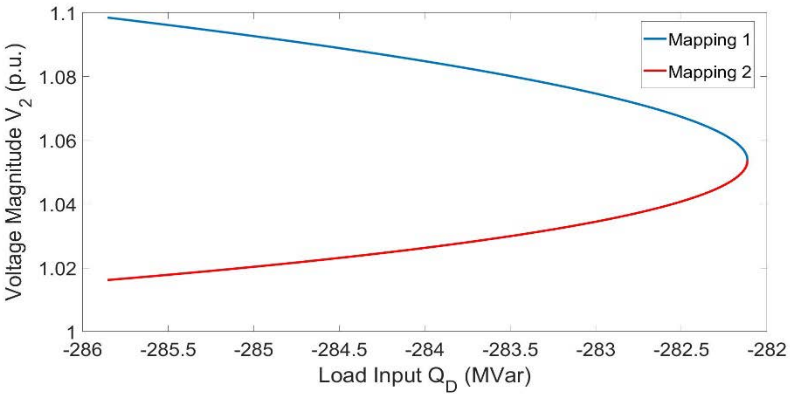
A toy 2-bus example.



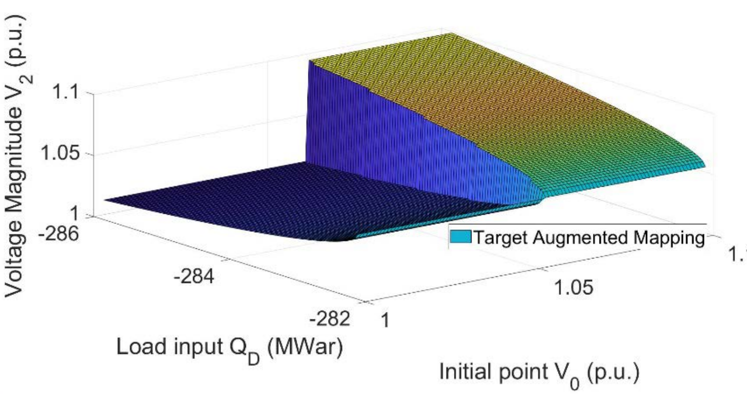
DNN's mapping vs. target mapping.

[1] W. A. Bukhsh, A. Grothey, K. I. McKinnon, and P. A. Trodden, "Local solutions of the optimal power flow problem," IEEE Trans. Power Syst., vol. 28, no. 4, 2013.
[2] J. Kotary, F. Fioretto, and P. Van Hentenryck, "Learning hard optimization problems: A data generation perspective," NeurIPS 2021.
[3] W. Huang and M. Chen, "DeepOPF-NGT: A Fast Unsupervised Learning Approach for Solving AC-OPF Problems without Ground Truth," ICML Workshop, 2021.

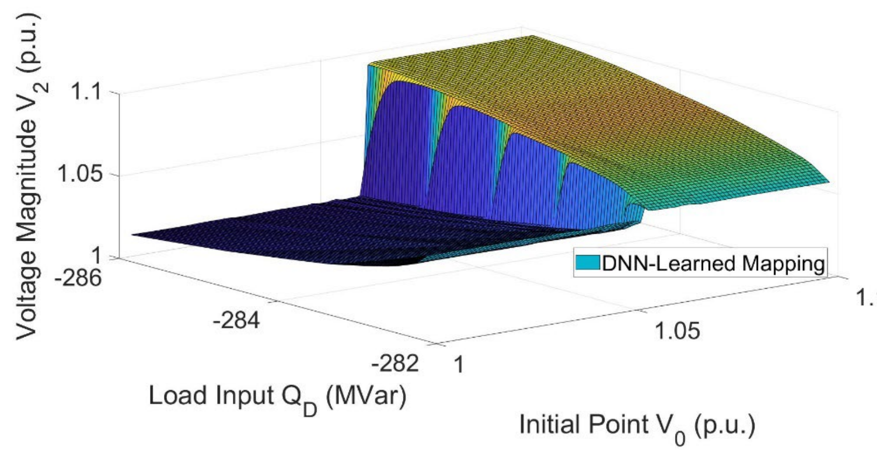
Our Approach: Augmented Learning



Embedding

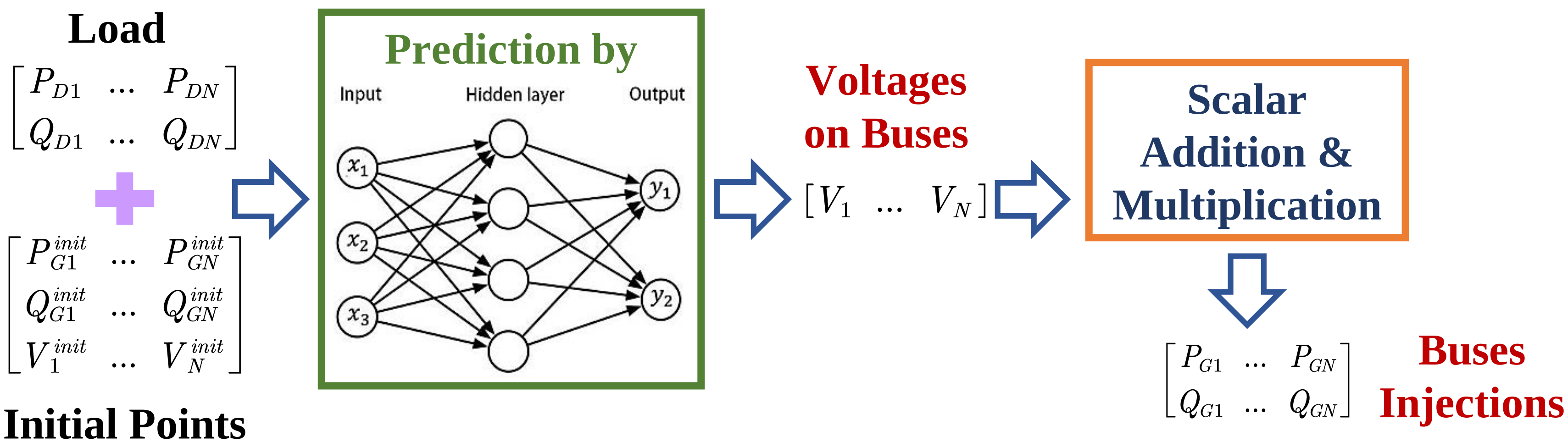


- Augment the load with the initial point in training data generation
- The augmented mapping is unique and can be learned by DNN



DeepOPF-AL: A Simple Design

- Follow prediction-and-reconstruction in DeepOPF-V [1]
  - Learn the unique augmented mapping from (load, initial point) to optimal solution



[1] W. Huang, X. Pan, M. Chen, and S. H. Low, "Deepopf-V: Solving AC-OPF problems efficiently," IEEE Trans. Power Syst., 2021

Simulation for Learning 2-valued Mapping

- Compare DeepOPF-AL with DeepOPF-V over IEEE Case39 with realistic load profile (40% variation)
  - Each load corresponds to two solutions

Metric	Case39-V2 with Avg. Cost Diff. = 30%			
	Balanced Dataset		Unbalanced Dataset	
	DeepOPF-AL	DeepOPF-V	DeepOPF-AL	DeepOPF-V
$\eta_{opt}(\%)$	**0.48**	-8.56	**0.66**	-5.98
$\eta_{P_G}(\%)/\eta_{Q_G}(\%)$	97.5/**94.6**	99.3/91.7	97.4/**98.4**	99.8/96.7
$\eta_{S_l}(\%)$	100	100	100	100
$\eta_{P_D}(\%)/\eta_{Q_D}(\%)$	**0.19/6.47**	0.61/27.6	**0.09/0.49**	0.32/12.9
t_{mips} (ms)	2808	2808	2676	2676
t_{dnn} (ms)	1.4	1.3	1.3	1.2
η_{speed}	×2006	×2160	×2058	×2230

Summary

- Standard supervised learning fails to solve AC-OPF with multi-valued load-solution mapping
- DeepOPF-AL generates quality solutions by learning a unique augmented mapping for AC-OPF with multi-valued mapping
- Augmented learning applicable to general constrained problems with multi-valued mapping
- Future work: Reduce complexity for augmented learning

Hybrid and Other Approaches

Predicting Active Constraints for Solving OPF Problems

- Y. Ng, S. Misra, L. A. Roald, and S. Backhaus, "Statistical learning for DC optimal power flow", in PSCC. Dublin, Ireland, 2018
- D. Deka and S. Misra, "Learning for DC-OPF: Classifying active sets using neural nets", in Proc. IEEE Milan PowerTech. 2019
- A. Robson, M. Jamei, C. Ududec, and L. Mones, "Learning an optimally reduced formulation of OPF through meta-optimization," arXiv:1911.06784, 2019.
- S. Misra, L. Roald and Y. Ng, "Learning for constrained optimization: Identifying optimal active constraint sets", INFORMS Journal on Computing, 34(1), 463-480, 2022.
- Y. Chen and B. Zhang, "Learning to solve network flow problems via neural decoding," arXiv:2002.04091, 2020.
- L. Zhang, Y. Chen, and B. Zhang, "A convex neural network solver for DCOPF with generalization guarantees," IEEE TCNS, 2021.

Idea and Existing Works

- **Idea**: Predict active constraints upon the given load
  - Reduce problem size
  - May directly solve (active-constrained) equations for solutions

- **Existing works**:
  - Classify active/inactive constraints by learning techniques [1-4]
  - Predict dual variables and then derive the active constraints[5-6]
  - Pros: optimal and feasible solutions if all active constraints found
  - Cons: no guarantee; limited speedup; do not work well for AC-OPF

[1] Y. Ng, S. Misra, L. A. Roald, and S. Backhaus, "Statistical learning for DC optimal power flow", in PSCC. Dublin, Ireland, 2018.
[2] D. Deka and S. Misra, "Learning for DC-OPF: Classifying active sets using neural nets", in Proc. IEEE Milan PowerTech, 2019.
[3] A. Robson, M. Jamei, C. Ududec, and L. Mones, "Learning an optimally reduced formulation of OPF through meta-optimization," arXiv:1911.06784, 2019.
[4] S. Misra, L. Roald and Y. Ng, "Learning for constrained optimization: Identifying optimal active constraint sets", INFORMS Journal on Computing, 34(1), 463-480, 2022.
[5] Y. Chen and B. Zhang, "Learning to solve network flow problems via neural decoding," arXiv:2002.04091, 2020.
[6] L. Zhang, Y. Chen, and B. Zhang, "A convex neural network solver for DCOPF with generalization guarantees," IEEE TCNS, 2021.

Performance on IEEE 14-/39-bus Networks

- Load profile and training/test dataset :
  - IEEE 14-bus case: $\pm 30/50\%$ variation, 40K/10K samples for training/test
  - IEEE 118-bus case: $\pm 9\%$ variation, 48K/12K samples for training/test

- Baseline: CVXOPT solver

Cases	Speedup	Optimality Gap (%)	Feasibility Rate (%)
IEEE 14-bus	x10	0	93.1
IEEE 118-bus	x10	0	89.0

[1] Y. Chen and B. Zhang, "Learning to solve network flow problems via neural decoding," arXiv preprint arXiv:2002.04091, 2020.
[2] L. Zhang, Y. Chen, and B. Zhang, "A convex neural network solver for dcopf with generalization guarantees," IEEE TCNS, 2021.

Learning-Boosted Iterative Schemes for OPF Problems (along the line of learn-to-optimize)

- Baker K. A learning-boosted quasi-newton method for ac optimal power flow. arXiv preprint arXiv:2007.06074, 2020.
- D. Biagioni, P. Graf, X. Zhang, A. Zamzam, K. Baker, and J. King. Learning-accelerated ADMM for distributed DC optimal power flow. IEEE Control Systems Letters, 2020

Learning-Boosted Quasi-Newton

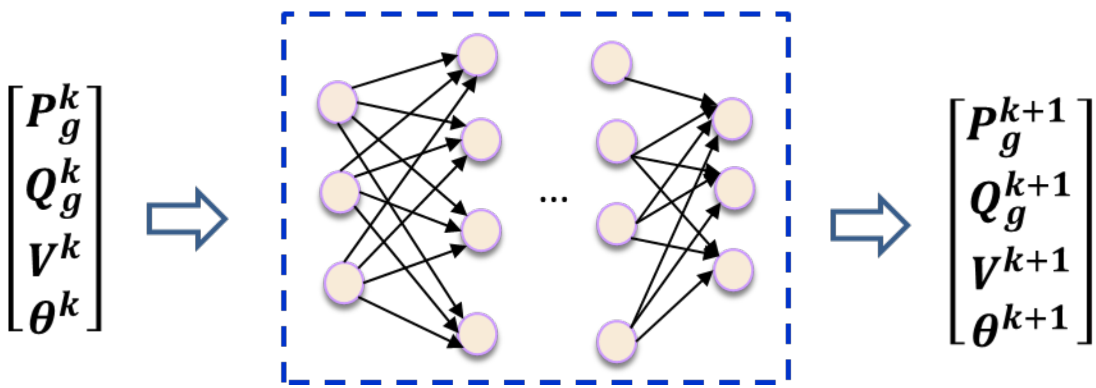
- Newton's method: update x according to KKT vector $d()$ and the Jacobian matrix of the KKT conditions $J()$:

$$x^{k+1} = x^k - \alpha^k J^{-1}(x^k) d(x^k)$$

- Quasi-Newton method: approximate the Jacobian matrix or its inverse for low complexity

$$x^{k+1} = x^k - \alpha^k H^{-1}(x^k) d(x^k)$$

- Learning-boosted method:
  - **Key idea:** replace Newton's update step with DNN [1], for AC-OPF



[1] Baker K. A learning-boosted quasi-newton method for ac optimal power flow. arXiv preprint arXiv:2007.06074, 2020.

Performance of Learning-Boosted Scheme

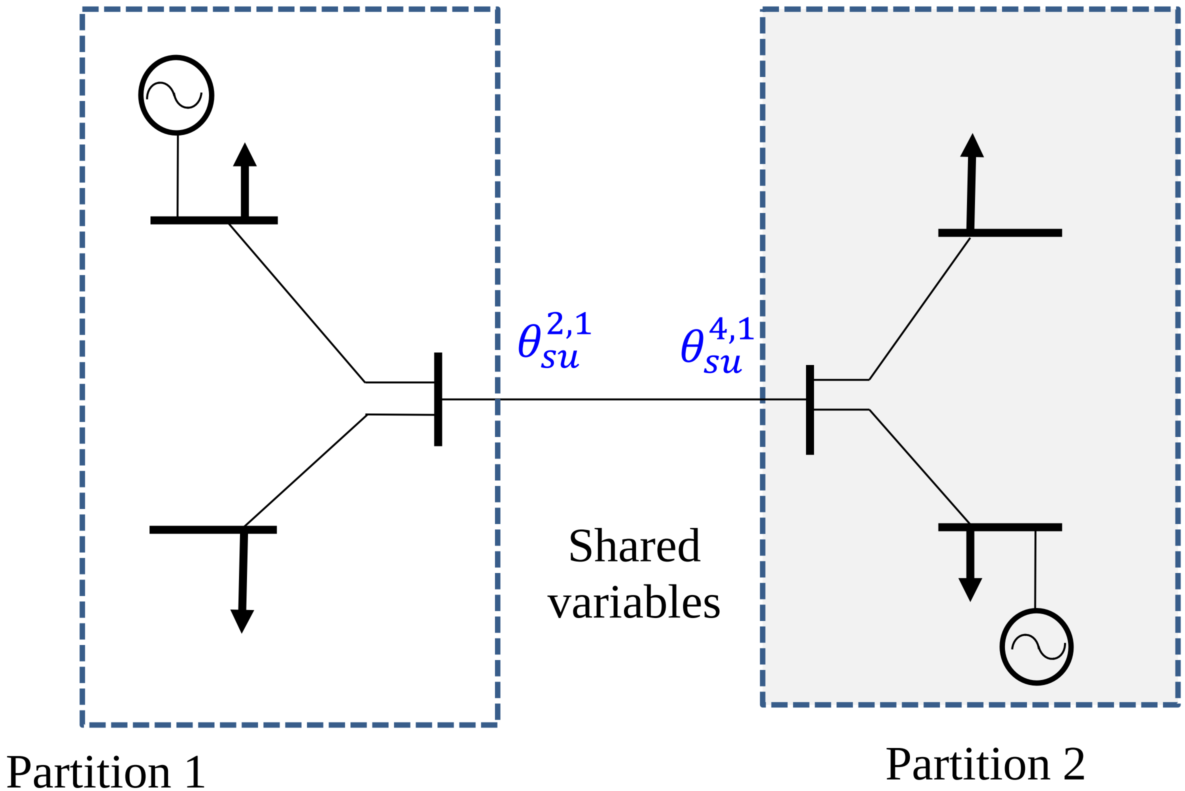
- Compare against the MATPower solver, over IEEE 30-/300-bus, PG-lib 500-bus, and 1,354-bus system

Metric	30-bus	300-bus	500-bus	1354-bus
MAE_{vm} (p.u.)	0.004	0.009	0.099	0.019
MAE_{pg} (MW)	0.64	1.47	0.62	7.55
MAE_{cost} (%)	0.29	0.65	0.66	1.16
Speed up	-x0.66	x36.6	x18.3	x22.5
Mean constraint violation (p.u.)	0.05	0.43	0.32	9.95

[1] Baker K. A learning-boosted quasi-newton method for ac optimal power flow. arXiv preprint arXiv:2007.06074, 2020.

Learning-Accelerated ADMM for Distributed DC-OPF

- Partition the OPF problem into subproblems with shared boundary variables and apply consensus ADMM



[1] D. Biagioni, P. Graf, X. Zhang, A. Zamzam, K. Baker, and J. King. Learning-accelerated ADMM for distributed DC optimal power flow. IEEE Control Systems Letters, 2020

Learning-Accelerated ADMM

- Recurrent neural networks (RNN)-based ADMM [1]
  - Inputs: shared variable θ_{su}^{k} and dual variables λ^{k} of previous K steps (sequentially)
  - Outputs: the optimal $(\theta_{su}^{*}, \lambda^{*})$

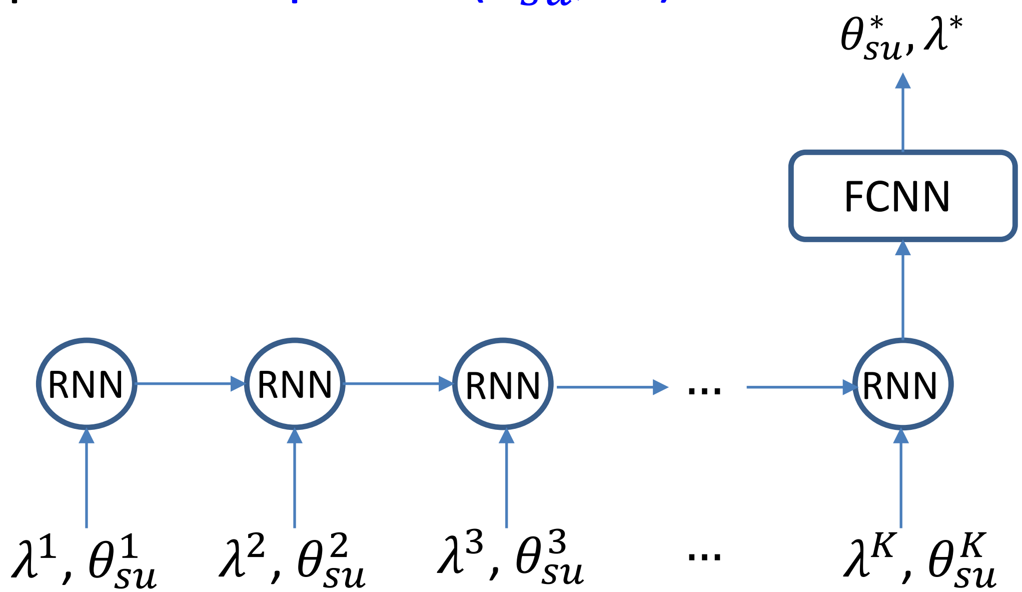


Figure: Schematic learning-accelerated ADMM method [1]

[1] D. Biagioni, P. Graf, X. Zhang, A. Zamzam, K. Baker, and J. King. Learning-accelerated ADMM for distributed DC optimal power flow. IEEE Control Systems Letters, 2020

Learning-Accelerated ADMM

- The performance of LA-ADMM is tested on the IEEE 14-/118- bus and RTE 2848-bus system

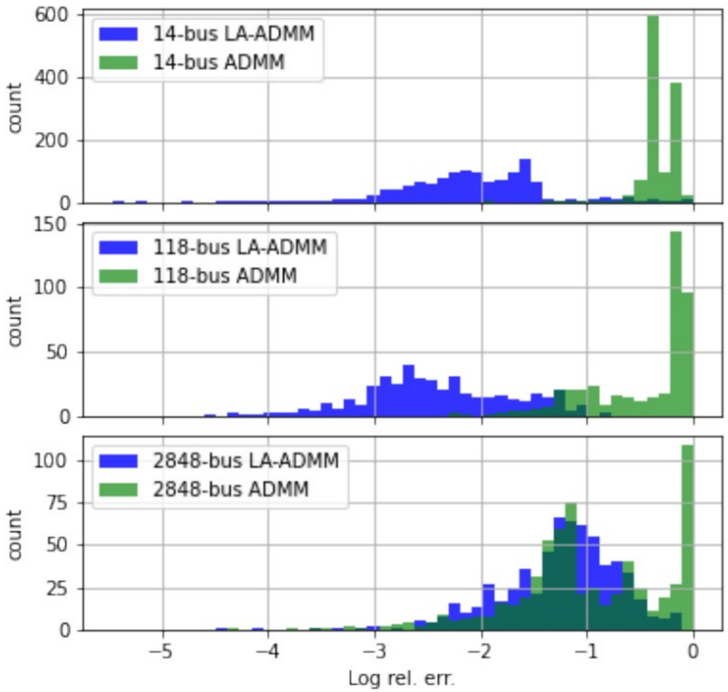


Figure: Histograms of log10 relative error in the objective cost of standard ADMM and LA-ADMM after 4 iterations.

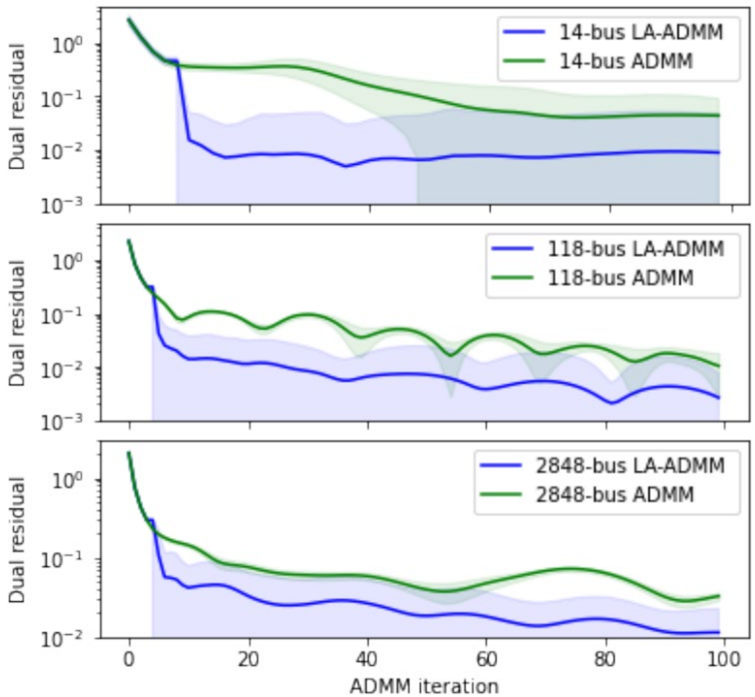


Figure: Residual error as function of ADMM iteration averaged over all test cases.

[1] D. Biagioni, P. Graf, X. Zhang, A. Zamzam, K. Baker, and J. King. Learning-accelerated ADMM for distributed DC optimal power flow. IEEE Control Systems Letters, 2020

A Physics-Guided Graph Convolution Neural Network for Optimal Power Flow

Slides based on those provided by Maosheng Gao, Juan Yu, Zhifang Yang, and Junbo Zhao

CHONGQING UNIVERSITY

UNIVERSITY OF CONNECTICUT

… Requires Solving OPF Frequently

$$\min_{u} \; f(u)$$ ← Total generation cost

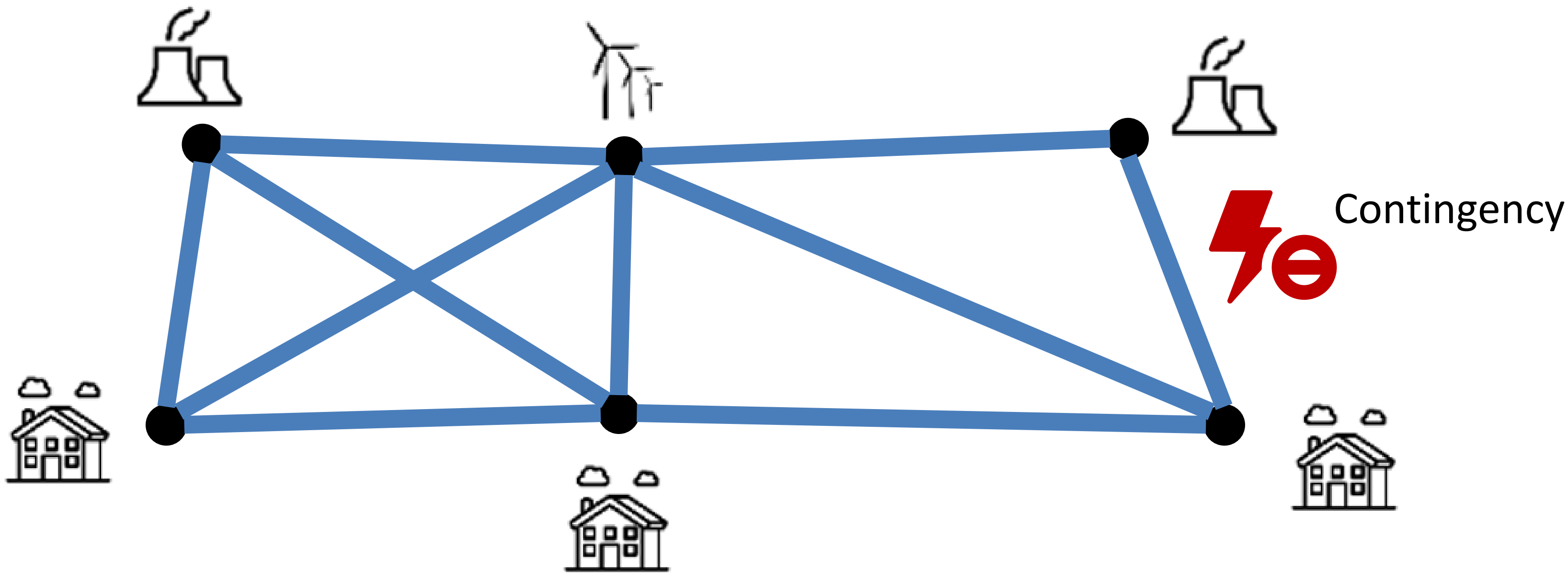
$$\text{s.t. } g(x,u) = 0$$ ← Non-convex power balance constraints

$$h(x,u) \geq 0$$ ← Non-convex Physical and operational constraints (e.g., branch limits)

- The uncertainty of renewable energy forces solving OPF frequently
- Data-driven methods using neural networks can yield 20-100 times faster than conventional optimization-based methods

[1] X. Pan, T. Zhao and M. Chen, "DeepOPF: Deep Neural Network for DC Optimal Power Flow", SmartGridComm, 2019. (arXiv:1905.04479, May 11th, 2019)
The Journal version for SC-DCOPF appears in IEEE Transactions on Power Systems in 2021.

Varying Operating Conditions of AC-OPF

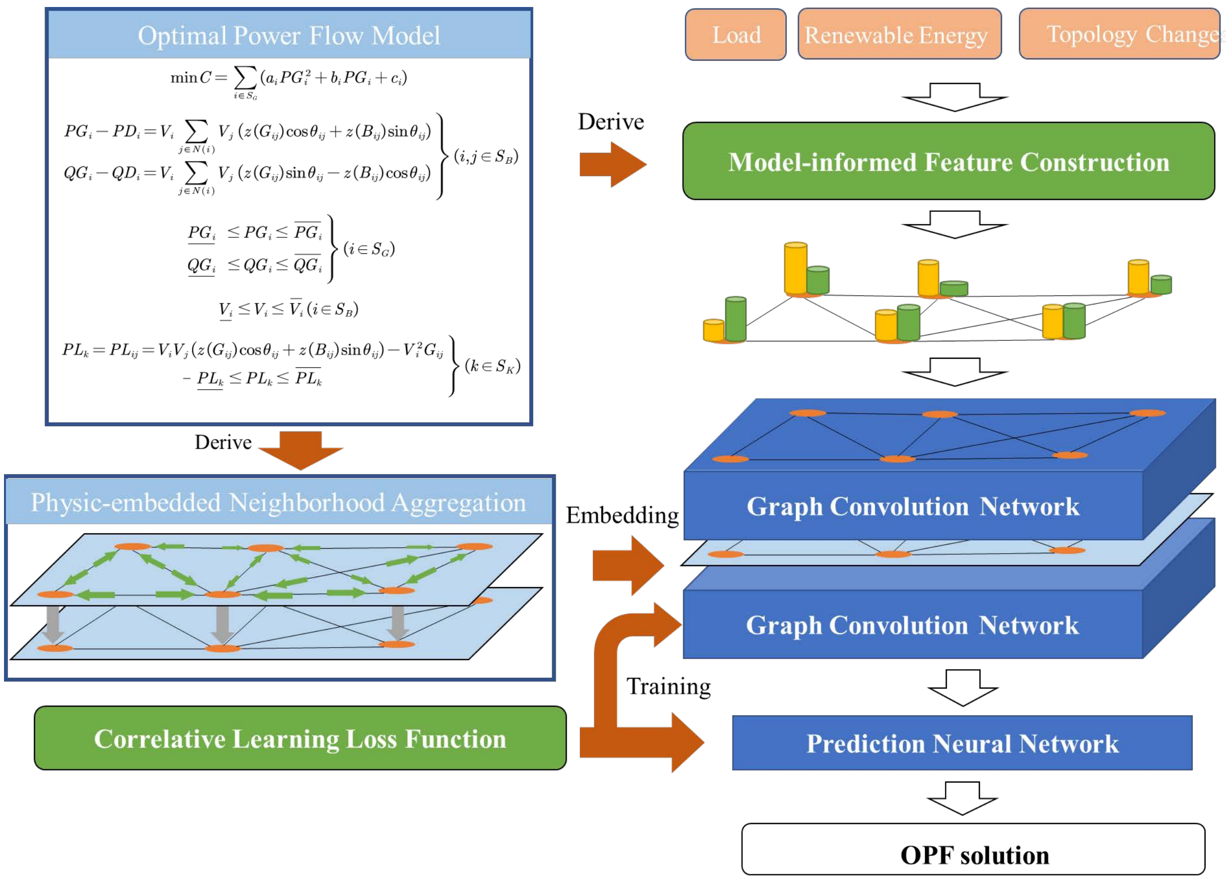


- Contingencies of generator, transformer, etc.
- Topology feature is complex
- All operation conditions satisfy the identical physical model

Nonlinearity

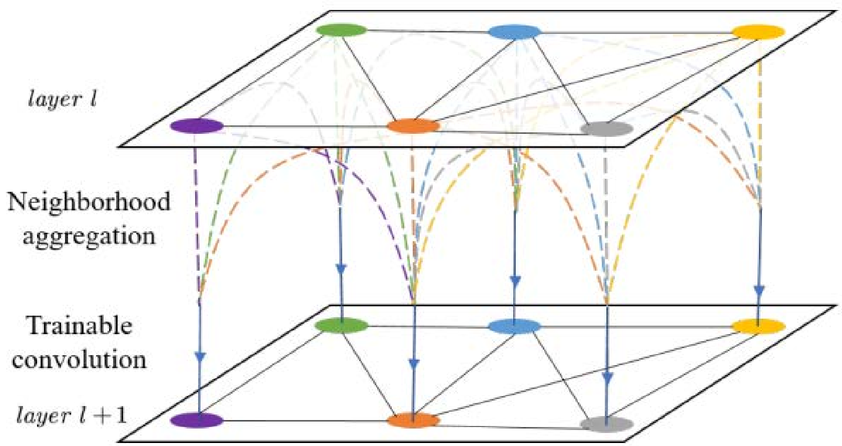
Discreteness

Our Approach: Physics-Guided Graph Convolution Neural Networks



- □ *Physics-embedded graph convolution* is derived by decomposing the AC power flow equations based on Gaussian-Seidel iteration
- □ *Model-informed feature construction* is proposed by aggregating neighbor node features
- □ *A new constrained loss function* is proposed to consider the physical correlations among the outputs

Physics-embedded Graph Convolution



$$\begin{cases} PG_i - PD_i = e_i \sum_{j\in N(i)} \left(z(G)_{ij} e_j - z(B)_{ij} f_j \right) + \\ \qquad f_i \sum_{j\in N(i)} \left(z(G)_{ij} f_j + z(B)_{ij} e_j \right) \\ QG_i - QD_i = f_i \sum_{j\in N(i)} \left(z(G)_{ij} e_j - z(B)_{ij} f_j \right) - \\ \qquad e_i \sum_{j\in N(i)} \left(z(G)_{ij} f_j + z(B)_{ij} e_j \right) \end{cases}$$

Aggregating the power flow feature

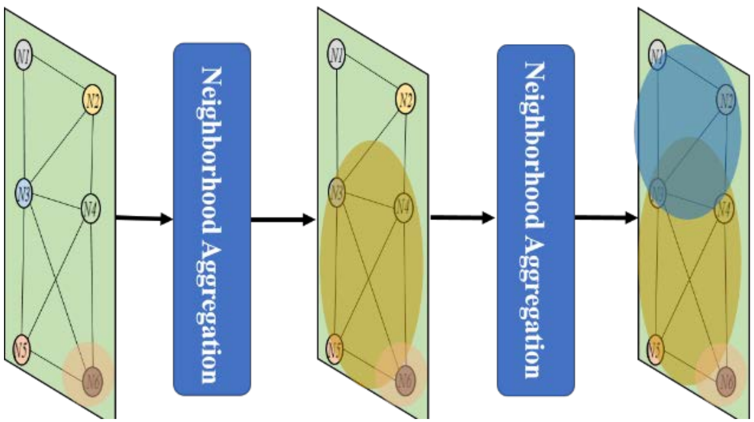
$$e_i^{l+1} = \frac{\delta_i \alpha_i - \lambda_i \beta_i}{\alpha_i^2 + \beta_i^2},\ f_i^{l+1} = \frac{\delta_i \beta_i + \lambda_i \alpha_i}{\alpha_i^2 + \beta_i^2} \quad where \begin{cases} \alpha_i = \sum_{j\in N(i), j\neq i} \left(z(G)_{ij} e_j - z(B)_{ij} f_j \right), \delta_i = PG_i - PD_i - \left(e_i^2 + f_i^2 \right) z(G)_{ii} \\ \beta_i = \sum_{j\in N(i), j\neq i} \left(z(G)_{ij} f_j + z(B)_{ij} e_j \right), \lambda_i = QG_i - QD_i + \left(e_i^2 + f_i^2 \right) z(B)_{ii} \end{cases}$$

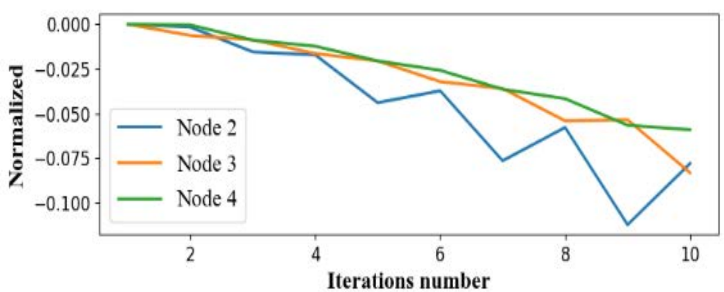
- □ Replacing the graph convolution kernel with a physics-embedded aggregation function

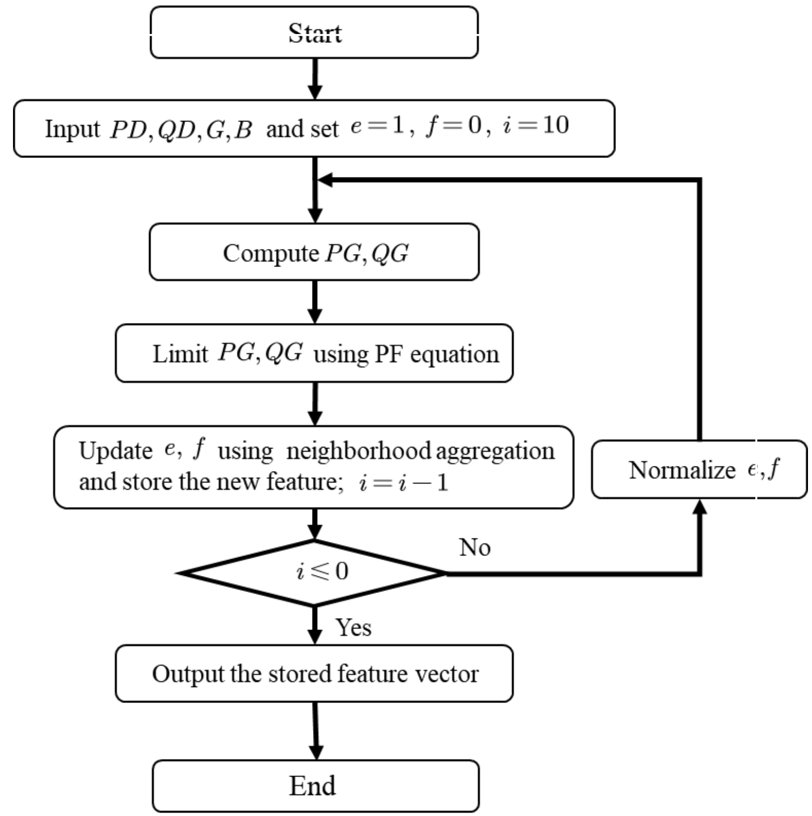
Physics-embedded Graph Convolution

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{e}^{l+1} \\ \boldsymbol{f}^{l+1} \end{bmatrix} = f\left(\begin{bmatrix} \frac{\boldsymbol{\delta} \odot \boldsymbol{\alpha} - \boldsymbol{\lambda} \odot \boldsymbol{\beta}}{\boldsymbol{\alpha} \odot \boldsymbol{\alpha} + \boldsymbol{\beta} \odot \boldsymbol{\beta}} & 0 \\ 0 & \frac{\boldsymbol{\delta} \odot \boldsymbol{\beta} + \boldsymbol{\lambda} \odot \boldsymbol{\alpha}}{\boldsymbol{\alpha} \odot \boldsymbol{\alpha} + \boldsymbol{\beta} \odot \boldsymbol{\beta}} \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_1 \\ \boldsymbol{W}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \end{bmatrix} \right),$$

Model-informed Feature Construction

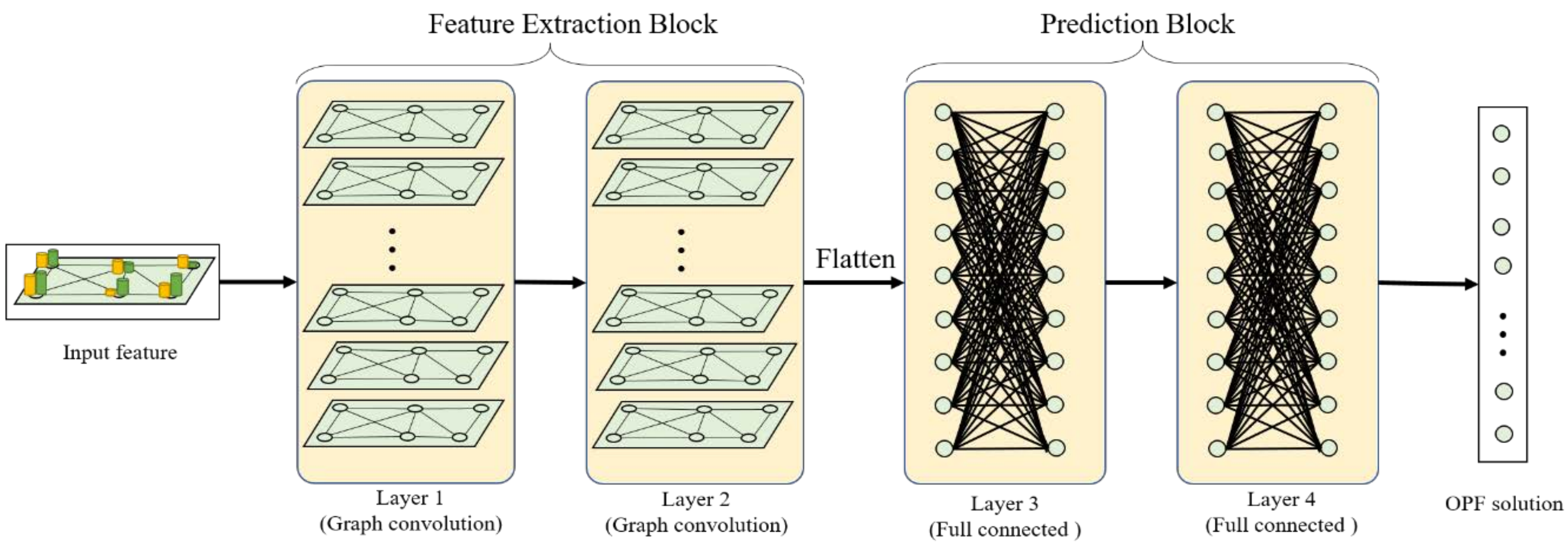






- Iterative aggregating of the node feature based on physics-embedded convolution and feature constraints

Physics-Guided Graph Convolution Neural Networks for OPF



- The *complete architecture of physics-guided GCNN* for the OPF problem combining the feature extraction block with several graph convolution layers and prediction block with several fully connected layers
- It could be the basic structure of neural networks for other applications, such as the real-time OPF calculation using reinforcement learning techniques

Simulation under Fixed Topology

- Compare different GCNNs over IEEE Case57 with 10% load variation
  - Physics-embedded GCNN has better convergence in training and lower mean absolute error in testing

		Physics-embedded graph convolution kernel	Model-guided feature construction	Correlative learning loss function
M1	Original GCNN [1]	×	×	×
M2	GCNN [2]	×	×	×
M3	Physics-guided GCNN	○	×	×
M4		○	○	×
M5		○	○	○

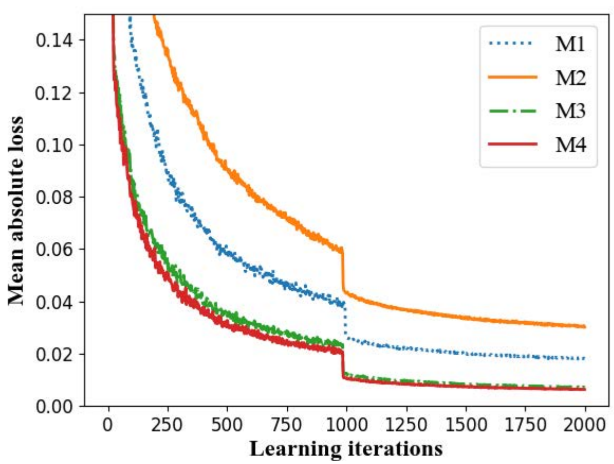


Fig. 1. The loss curve of the four neural networks.

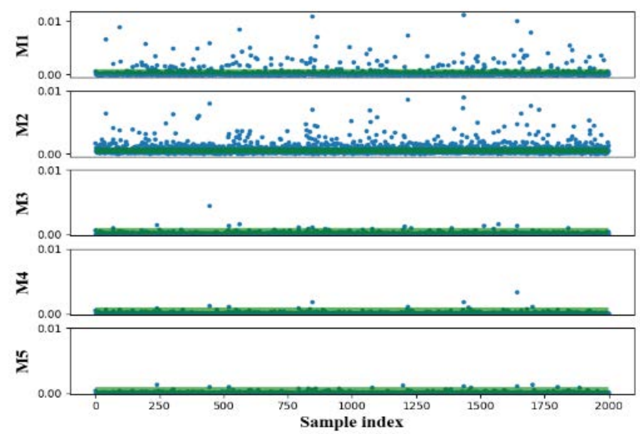


Fig. 2. The mean absolute error of testing samples.

[1] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *ICLR*, 2017, pp. 1–14.
[2] D. Owerko, F. Gama and A. Ribeiro, “Optimal power flow using graph neural networks,” ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 5930-5934.

Simulation under Varying Topologies

- Compare different data-driven OPF methods over different IEEE systems with 10% load variation and five topologies
  - Physics-embedded GCNN has enhanced topology feature extraction ability under varying topologies in power systems

M6	CNN [1]	The diagonal elements of admittance matric is regarded as topology input feature.
M7	CNN [2]	The $\lvert Y_{\text{diag}} \rvert$ and $\angle Y_{\text{diag}}$ is regarded as topology input feature.
M8	DNN [3]	The voltage difference of each bus are used as the topology feature.
M9	DNN	The diagonal elements of B are used as the topology feature.

Table I. The testing accuracy of different methods under varying topologies

System		M5	M6	M7	M8	M9
39-bus	P_{VG}	**97.76%**	86.06%	86.15%	93.09%	88.28%
	P_{PG}	**98.05%**	84.39%	86.31%	92.72%	89.82%
57-bus	P_{VG}	**99.99%**	84.43%	71.02%	82.24%	77.57%
	P_{PG}	**99.76%**	93.36%	82.51%	96.37%	94.35%
118-bus	P_{VG}	**93.49%**	83.29%	70.05%	82.47%	83.65%
	P_{PG}	**98.47%**	92.89%	85.45%	93.18%	93.05%
300-bus	P_{VG}	**92.72%**	69.83%	77.46%	60.09%	71.72%
	P_{PG}	**94.05%**	70.85%	78.26%	70.26%	72.63%

[1] Y. Du, F. Li, J. Li, and T. Zheng, "Achieving 100x acceleration for n-1 contingency screening with uncertain scenarios using deep convolutional neural network," IEEE Trans. Power Syst., vol. 34, no. 4, pp. 3303–3305, 2019.
[2] Y. Zhou, W. -J. Lee, R. Diao and D. Shi, "Deep reinforcement learning based real-time AC optimal power flow considering uncertainties," J. Mod. Power Syst. Clean Energy, vol. 10, no. 5, pp. 1098-1109, Sep. 2022.
[3] F. Hasan, A. Kargarian and J. Mohammadi, "Hybrid learning aided inactive constraints filtering algorithm to enhance ac OPF solution time," IEEE Trans. Ind. Appl., vol. 57, no. 2, pp. 1325-1334, March-April 2021.

Simulation for Potential Applications

- Apply the physics-embedded GCNN in PPO reinforcement learning for real-time OPF problems with the constraint violation penalty as the reward function [1]
  - Physics-embedded GCNN has better convergence compared with CNN and DNN

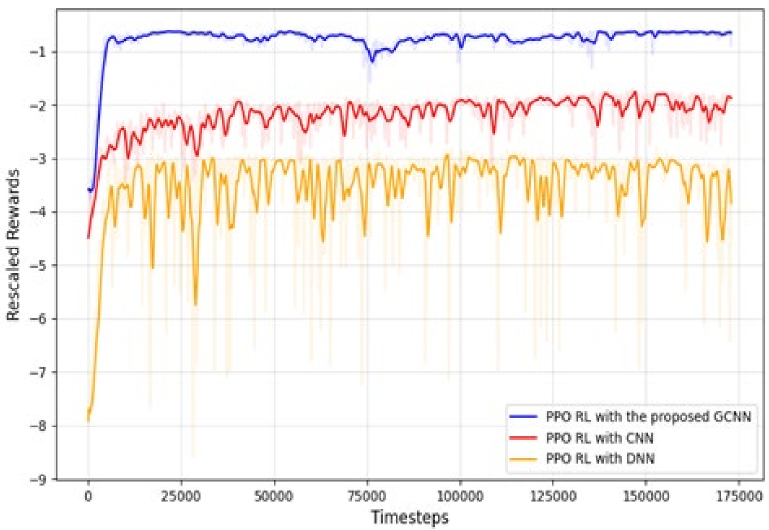


Fig. 1. The reward curves of different neural networks when training with the PPO RL algorithm.

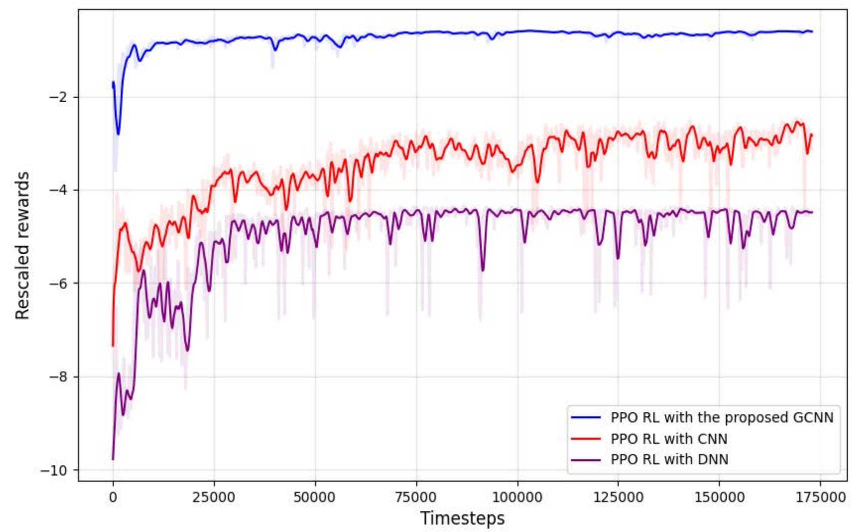


Fig. 2. The reward curves of different neural networks when considering the ramp rate.

[1] Y. Zhou, W. -J. Lee, R. Diao and D. Shi, "Deep reinforcement learning based real-time AC optimal power flow considering uncertainties," *J. Mod. Power Syst. Clean Energy*, vol. 10, no. 5, pp. 1098-1109, September 2022.

Summary

- □ Physics-embedded design can enhance the topologies feature extraction ability of GCNN
- □ The physics-embedded GCNN has potential application in other varying topologies problems in power systems
- □ Future work: address complicated OPF problems, such as the multiple-period co-optimization

Concluding Remarks

OPF is Critical for Power System Operation

- OPF is to minimize the cost of serving load subject to physical and operational constraints
  - KCL and KVL physical constraints
  - Voltage, generation, and branch flow limits
  - Other operational constraints

- OPF underpins various important power system applications
  - Demand response
  - Economic dispatch
  - Unit commitment
  - Electricity market clearing
  - Security and reliability assessment

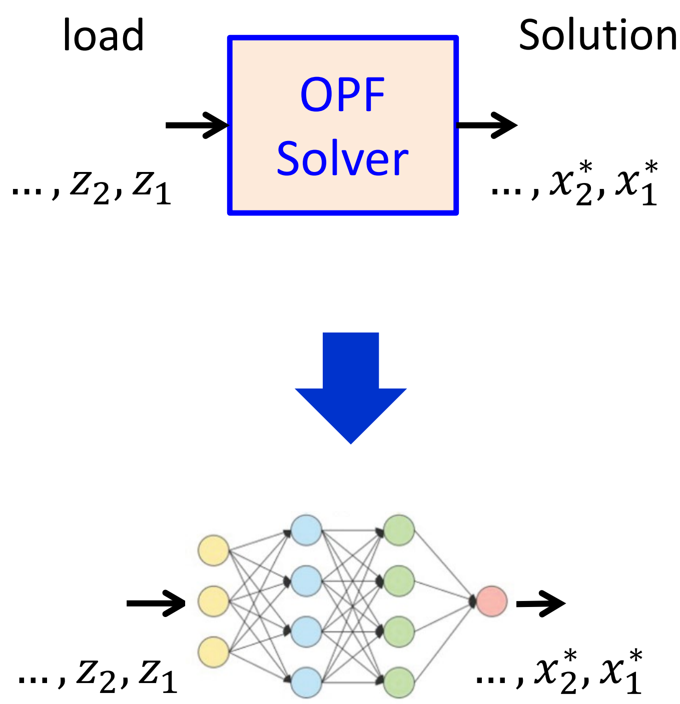
Solving OPF Efficiently is Important

- AC-OPF problem is non-convex and NP-hard, hard to solve in real-time
  - Practical OPF can involve more than 1M variables

- Penetration of renewable requires solving OPF frequently
  - Early termination of algorithms gives suboptimal solution
  - 5% saving amounts to 36 billion USD/year globally

- General Newton-like iterative algorithms — Slow
- Linearization to solve OPF approximately — Inaccurate
- Convexification to solve OPF optimality — Only applicable to special case

Directly Solving OPF by NN Works

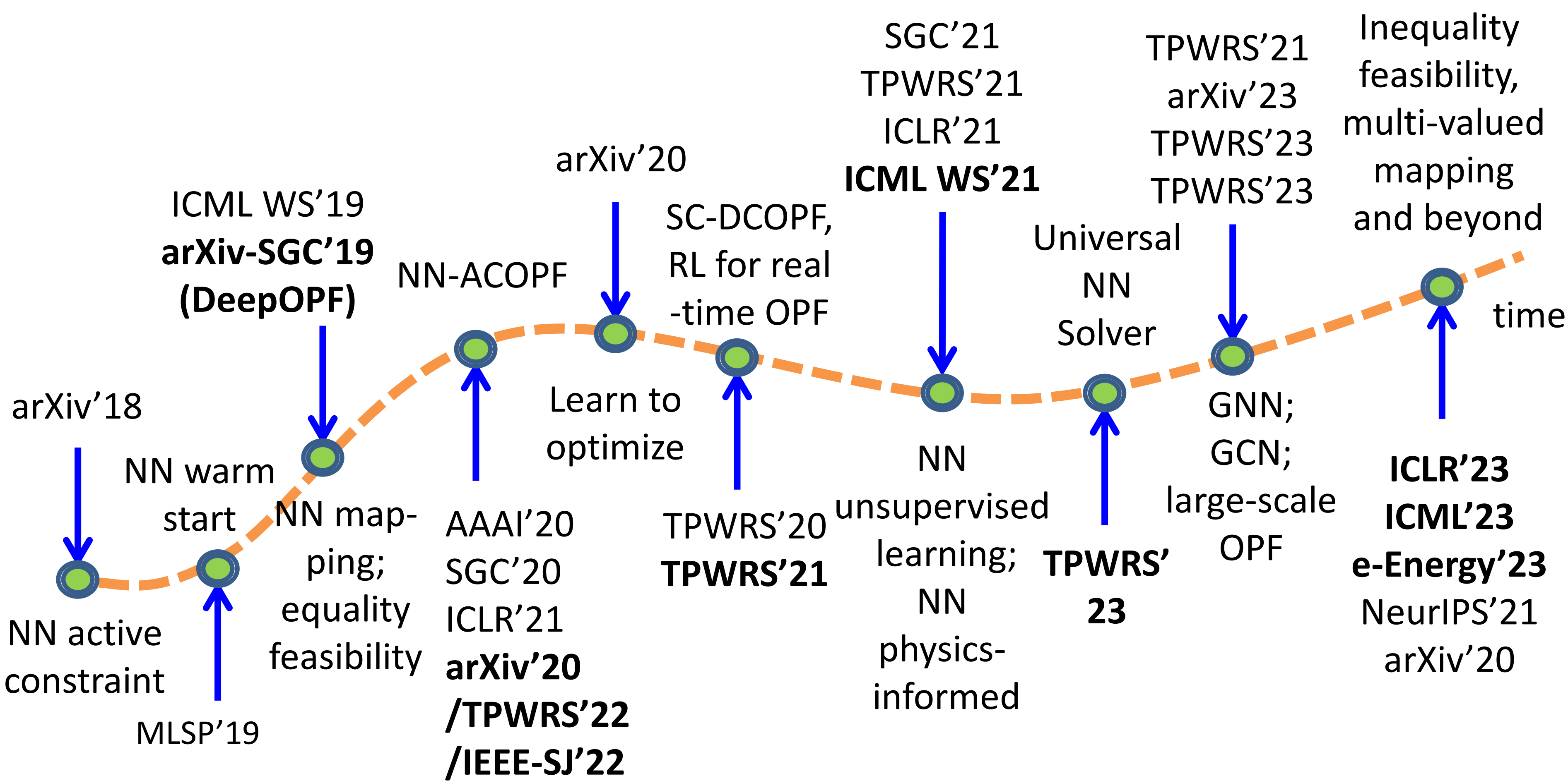


- <0.2% optimality loss in AC-OPF simulations over IEEE cases, real-world topology, and loads
  - With theoretical justification
  - 15,000x speedup over a 2000-bus network
- Generalizable approaches for ML solving constrained problems
  - Predict-and-Reconstruct to ensure equality feasibility
  - Preventive learning to ensure inequality constraints
  - Augmented learning to learn a legitimate mapping (when multiple mappings exists)

Open Issues

- Tighter NN size bounds for learning multi-dimension mapping
- Ensuring NN solution feasibility for inequality constraints beyond ball-homeomorphism
- Learning latent variables to reduce NN size and improve learning efficiency
- Other OPF formulations: real-time OPF, stochastic OPF, security-constrained AC-OPF

The Path Ahead is Still Unfolding



- Our works in bold font
- https://energy.hosting.acm.org/wiki/index.php/ML_OPF_wiki

Wiki and Overview Webpage

- A wiki page hosted by ACM SIGEnergy
  - https://energy.hosting.acm.org/wiki/index.php/ML_OPF_wiki
- A wiki page hosted by Climate Change AI (on more general topics)
  - https://wiki.climatechange.ai/wiki/Welcome_to_the_Climate_Change_AI_Wiki
- An overview webpage by Letif Mones
  - https://invenia.github.io/blog/2021/10/11/opf-nn/
- Dataset or data generators for training NN for OPF problems
  - https://github.com/NREL/OPFLearn.jl
  - https://github.com/invenia/OPFSampler.jl/

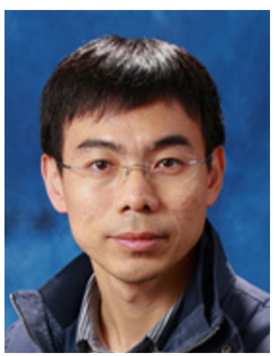
Acknowledgements

Xiang Pan
(CUHK; Tencent)

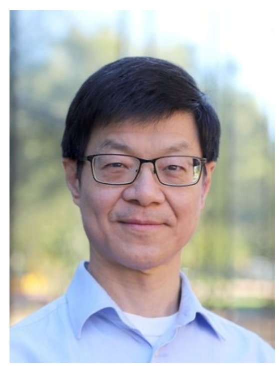
Tianyu Zhao
(CUHK; Lenovo Research)

Min Zhou
(CityU)

Enming Liang
(CityU)



Shengyu Zhang
(Tencent)



Steven Low
(Caltech)

Wanjun Huang
(CityU; Beihang Univ)

Thank You!

Minghua Chen and Steven Low

香港城市大學
City University of Hong Kong

Caltech