Solving a Class of Optimal Power Flow Problems in Tree Networks

Amir Beck$^1$  Yuval Beck$^2$  Yoash Levron$^3$  Alex Shtof$^3$
Luba Tetruashvili$^3$

$^1$Tel Aviv University
$^2$Holon Institute of Technology
$^3$Technion - Israel Institute of Technology

July, 2018
Power networks and problems
Power network (AC)

"A mathematical model describing problems on a large electrical circuit which conveys power from generators to consumers."

Ingredients

- Model data - an augmented graph
- Decision variables
- Constraints

Two fundamental problems

Feasibility  Power Flow
Optimization  Optimal Power Flow
Model data: \( P = (G, z, C_1, \ldots, C_n) \).
- Topology graph \( G = (V, E) \).
- Edge impedances \( 0 \neq z_{ij} \in \mathbb{C} \).
- Nodal constraints \( C_i \in \mathbb{R} \times \mathbb{C} \).

Decision variables:
- Voltage - \( v \in \mathbb{C}^n \), Power - \( s \in \mathbb{C}^n \).

Constraints:
\[
\begin{align*}
  s_i &= \sum_{j \in N(i)} v_i \frac{v_j^* - v_i^*}{z_{ij}^*} & i & \in V \\
  \arg(v_1) &= 0 \\
  (|v_i|, s_i) &\in C_i & i & \in V
\end{align*}
\]
Model data: \( P = (G, z, C_1, \ldots, C_n) \).
- Topology graph \( G = (V, E) \).
- Edge impedances \( 0 \neq z_{ij} \in \mathbb{C} \).
- Nodal constraints \( C_i \in \mathbb{R} \times \mathbb{C} \).

Decision variables:
- Voltage \(- v \in \mathbb{C}^n \), Power \(- s \in \mathbb{C}^n \).

Constraints:
\[
s_i = \sum_{j \in N(i)} v_i \frac{v_i^* - v_j^*}{z_{ij}^*} \quad i \in V
\]
\[
\arg(v_1) = 0
\]
\[
(|v_i|, s_i) \in C_i \quad i \in V
\]
Power network (AC)

Model data: $P = (G, z, C_1, \ldots, C_n)$.
- Topology graph $G = (V, E)$.
- Edge impedances $0 \neq z_{ij} \in \mathbb{C}$.
- Nodal constraints $C_i \in \mathbb{R} \times \mathbb{C}$.

Decision variables:
Voltage - $v \in \mathbb{C}^n$, Power - $s \in \mathbb{C}^n$.

Constraints:

$$s_i = \sum_{j \in N(i)} v_i^* \frac{v_j^* - v_i^*}{z_{ij}^*} \quad i \in V$$

$$\arg(v_1) = 0$$

$$(|v_i|, s_i) \in C_i \quad i \in V$$
Model data: $P = (G, z, C_1, \ldots, C_n)$.
- Topology graph $G = (V, E)$.
- Edge impedances $0 \neq z_{ij} \in \mathbb{C}$.
- Nodal constraints $C_i \in \mathbb{R} \times \mathbb{C}$.

Decision variables:
Voltage - $v \in \mathbb{C}^n$, Power - $s \in \mathbb{C}^n$.

Constraints:

$\mathcal{FF}(P) = \{ (v, s) :$

$$s_i = \sum_{j \in N(i)} v_i \frac{v_i^* - v_j^*}{z_{ij}^*} \quad i \in V$$

$$\arg(v_1) = 0$$

$$\quad (|v_i|, s_i) \in C_i \quad i \in V$$

$\}$.
Common constraints

PQ constraint
- known power demand
\[ C_i = [u_i, u_i] \times \{ \hat{s}_i \} . \]
Examples:
\[ C_i = [0.9, 1.1] \times \{-5 - ı \} . \]
\[ C_i = [0.9, +\infty] \times \{-5 - ı \} . \]

PV constraint
- known voltage and power generation
\[ C_i = \{ \hat{u}_i \} \times \{ \hat{p}_i + ı[q_i, q_i] \} . \]
Example:
\[ C_i = [1.05] \times \{10 + ı[-5, +\infty] \} . \]
Common constraints

PQ constraint - known power demand

\[ C_i = [\underline{u}_i, \overline{u}_i] \times \{ \hat{s}_i \}. \]

Examples:

\[ C_i = [0.9, 1.1] \times \{ -5 - 1 i \}. \]

\[ C_i = [0.9, +\infty] \times \{ -5 - 1 i \}. \]
Common constraints

**PQ constraint** - known power demand

\[ C_i = [u_i, \bar{u}_i] \times \{\hat{s}_i\}. \]

Examples:

\[ C_i = [0.9, 1.1] \times \{-5 - 1\nu\}. \]
\[ C_i = [0.9, +\infty] \times \{-5 - 1\nu\}. \]

**PV constraint** - known voltage and power generation

\[ C_i = \{\hat{u}_i\} \times \{\hat{p}_i + \nu[q_i, \bar{q}_i]\}. \]

Example:

\[ C_i = [1.05] \times \{10 + \nu[-5, +\infty]\}. \]
The optimal power flow (OPF) problem

**Input:** A power network $P = (G, z, C_1, \ldots, C_n)$, and a cost function $f : \mathbb{C}^n \times \mathbb{C}^n \to \mathbb{R}$

**Objective:** Solve the optimization problem

$$\min_{v, s} \quad f(v, s)$$

$$\text{s.t.} \quad (v, s) \in \mathcal{FF}(P)$$
The optimal power flow (OPF) problem

**Input:** A power network $P = (G, z, C_1, \ldots, C_n)$, and a cost function $f : \mathbb{C}^n \times \mathbb{C}^n \to \mathbb{R}$

**Objective:** Solve the optimization problem

$$\min_{v, s} \quad f(v, s)$$

s.t. $\quad (v, s) \in \mathcal{FF}(P)$

**Cost function examples:**

$$f(v, s) = \sum_{i \in \text{Gen}} \text{re}(s_i), \quad f(v, s) = \sum_{i \in \text{PQ}} ||v_i| - 0.5 \cdot (u_i + \bar{u}_i)||$$
The optimal power flow (OPF) problem

**Input:** A power network \( P = (G, z, C_1, \ldots, C_n) \), and a cost function \( f : \mathbb{C}^n \times \mathbb{C}^n \to \mathbb{R} \)

**Objective:** Solve the optimization problem

\[
\begin{align*}
\min_{v, s} & \quad f(v, s) \\
\text{s.t.} & \quad (v, s) \in \mathcal{FF}(P)
\end{align*}
\]

**Cost function examples:**

\[
\begin{align*}
f(v, s) &= \sum_{i \in \text{Gen}} \text{re}(s_i), \quad f(v, s) = \sum_{i \in \text{PQ}} |v_i| - 0.5 \cdot (u_i + \bar{u}_i)|
\end{align*}
\]

**A common setup**

- Consumers have PQ constraints
- Generators have box constraints.
A non-convex and potentially non-smooth problem.
The bad news

Karsten Lehmann, Alban Grastien, Pascal Van Hentenryck. AC-Feasibility on Tree Networks is NP-Hard

*IEEE Transactions on Power Systems, 2015*
Approaches

- Heuristics\(^1\)
  - Interior point methods
  - Convex relaxations
- Global solution methods (e.g. branch and bound)
- Specialized solution\(^2\) methods for sub-classes
  - Tight convex relaxations
  - Our method

\(^1\)Known to work well in practice, but no theory explaining why.

\(^2\)A method equipped with a theory explaining why it works.
Approaches

- Heuristics\(^1\)
  - Interior point methods
  - Convex relaxations

- Global solution methods (e.g. branch and bound)

- Specialized solution\(^2\) methods for sub-classes
  - Tight convex relaxations
  - Our method

---

1. Known to work well in practice, but no theory explaining why.
2. A method equipped with a theory explaining why it works.
Our setup
Assumptions

Network assumptions

- The topology is a tree \( T = (V, E) \) with root\( (T) = 1 \) and \(|V| > 1\).
- Leaf constraints: \( C_i \) is a compact PQ or PV constraint.
- Root constraint: \( C_1 = [u_r, \bar{u}_r] \times \mathbb{C} \).
- Remaining constraints: \( C_i \) is a PQ constraint.
- Non-zero voltages: \((u, s) \in C_i \implies u > 0\).
Assumptions

Network assumptions

- The topology is a tree $T = (V, E)$ with $\text{root}(T) = 1$ and $|V| > 1$.
- Leaf constraints: $C_i$ is a compact PQ or PV constraint.
- Root constraint: $C_1 = [u_r, \bar{u}_r] \times \mathbb{C}$.
- Remaining constraints: $C_i$ is a PQ constraint.
- Non-zero voltages: $(u, s) \in C_i \implies u > 0$.

Problem assumptions

- The objective function $f$ is continuous.
Visualization
Example

\[ z_{12} = 0.02 + 0.005 \}
\[ z_{23} = 0.02 + 0.01i \]
\[ z_{24} = 0.04 + 0.06i \]

\[ C_1 = [0.97, \infty] \times \mathbb{C} \]  
Root constraint

\[ C_2 = [0.9, 1.1] \times \{-0.2 - 0.1i\} \]  
PQ constraint

\[ C_3 = [0.9, 1.1] \times \{-0.4 - 0.3i\} \]  
PQ constraint

\[ C_4 = \{1\} \times (0.25 + i[-1, 1]) \]  
PV constraint
Motivation

Only one degree of freedom. So why the effort?
Motivation

- It is challenging - to the best of our knowledge, no known efficient solution.
- Useful as a computational step.

https://flic.kr/p/5jLLgR
The tree reduction / expansion method
The main result

- A decomposition of the feasible set into a finite union of parameterized curves:

\[ \mathcal{F}\mathcal{F}(P) = \bigcup_{i=1}^{m} \text{image}(\gamma_i), \]

where \( \gamma_i : [0, 1] \rightarrow \mathbb{C}^n \times \mathbb{C}^n \) are continuous and piecewise smooth functions.

- An algorithm to compute a representation of \( \{\gamma_i\}_{i=1}^{m} \).

- An efficient algorithm to compute \( \gamma_i(t) \) given \( i \) and \( t \).
Solving OPF

- Compute a representation of the curves: \( \{ \gamma_i \}_{i=1}^m \).
- Choose sampling density \( N \in \mathbb{N} \).
- Grid search:

\[
(\hat{\mathbf{v}}, \hat{s}) \in \arg\min_{(\mathbf{v}, s)} \{ f(\mathbf{v}, s) : (\mathbf{v}, s) = \gamma_i(j/(N - 1)), i = 1, \ldots, m, j = 0, \ldots, N - 1 \}\]

Observation: For any \((v, s) \in \mathcal{FF}(P)\), the vector \(s\) is redundant:

\[
s_i = \sum_{j \in N(i)} v_i \frac{v_i^* - v_j^*}{z_{ij}^*}, \quad i = 1, \ldots, n.
\]

We work with \(\mathcal{FF}_v(P) = \{v : (v, s) \in \mathcal{FF}(P)\}\).
Observation: Compact PQ and PV constraints are line segments in \( \mathbb{R} \times \mathbb{C} \).

Example: \([0.9, 1.1] \times \{-0.4 - 0.3i\}\) is the line segment between \((0.9, -0.4 - 0.3i)\) and \((1.1, -0.4 - 0.3i)\).
Observation: Compact PQ and PV constraints are line segments in \( \mathbb{R} \times \mathbb{C} \).

Example: \([0.9, 1.1] \times \{ -0.4 - 0.3i \} \) is the line segment between \((0.9, -0.4 - 0.3i)\) and \((1.1, -0.4 - 0.3i)\).

Observation: Line segments are curves.

Idea: Replace leaf constraints with parameterized curves - functions from \([0, 1]\) to \( \mathbb{R} \times \mathbb{C} \).
Example

Input network

\[ C_1 = [0.97, \infty] \times \mathbb{C} \]
\[ C_2 = [0.9, 1.1] \times \{-0.2 - 0.1i\} \]
\[ C_3 = [0.9, 1.1] \times \{-0.4 - 0.3i\} \]
\[ C_4 = \{1\} \times (0.25 + i[-1, 1]) \]

Curved network

\[ C_1 = [0.97, \infty] \times \mathbb{C} \]
\[ C_2 = [0.9, 1.1] \times \{-0.2 - 0.1i\} \]
\[ C_3 = \text{image}(\nu_3, \sigma_3) \]
\[ \nu_3(t) = 0.9 + 0.2t, \quad \sigma_3(t) = -0.4 - 0.3i \]
\[ C_4 = \text{image}(\nu_4, \sigma_4) \]
\[ \nu_4(t) = 1, \quad \sigma_4(t) = 0.25 + i(-1 + 2t) \]
Curved network

- The topology is a tree $T = (V, E)$ with root($T$) = 1.
- Leaf constraints: $C_i = \text{image}(\nu_i, \sigma_i)$ given continuous $\nu_i: [0, 1] \rightarrow \mathbb{R}$ and $\sigma_i: [0, 1] \rightarrow \mathbb{C}$.
- Root constraint:
  - If $|V| > 1$ then $C_1 = [u_r, \bar{u}_r] \times \mathbb{C}$.
  - If $|V| = 1$ then $C_1 = [u_r, \bar{u}_r] \times \{0\}$.
- Remaining constraints: $C_i$ is a PQ constraint.
- Non-zero voltages: $(u, s) \in C_i \implies u > 0$.
Curved network

- The topology is a tree $T = (V, E)$ with root($T$) = 1.
- Leaf constraints: $C_i = \text{image}(\nu_i, \sigma_i)$ given continuous $\nu_i : [0, 1] \rightarrow \mathbb{R}$ and $\sigma_i : [0, 1] \rightarrow \mathbb{C}$.
- Root constraint:
  - If $|V| > 1$ then $C_1 = [u_r, \overline{u}_r] \times \mathbb{C}$.
  - If $|V| = 1$ then $C_1 = [u_r, \overline{u}_r] \times \{0\}$.
- Remaining constraints: $C_i$ is a PQ constraint.
- Non-zero voltages: $(u, s) \in C_i \implies u > 0$.

Next: find a relationship between $\mathcal{F}\mathcal{F}_v(P)$ and $\mathcal{F}\mathcal{F}_v(P')$ for a smaller network $P'$. 
Tree reduction

- Reducible node - a non-leaf whose children are all leaves
- Tree reduction - the operation of removing all children of some reducible node.

Example

\[
T
\]

\[
\begin{array}{c}
A \\
B \\
C \\
E \\
F \\
G \\
H \\
\end{array}
\]

\[
T' \quad T
\]

\[
\begin{array}{c}
A \\
B \\
C \\
E \\
F \\
G \\
H \\
\end{array}
\]
Tree-Reduction Theorem

Let $P = (T, z, C_1, \ldots, C_n)$ be a curved network.
Let \( P = (T, z, C_1, \ldots, C_n) \) be a curved network. Let \( j \) be a reducible node, let \( T' \) be the resulting reduction, and w.l.o.g its children are \( \{j + 1, \ldots, n\} \).
Tree-Reduction Theorem

Let \( P = (T, z, C_1, \ldots, C_n) \) be a curved network. Let \( j \) be a reducible node, let \( T' \) be the resulting reduction, and w.l.o.g its children are \( \{j + 1, \ldots, n\} \). Define

\[
\tilde{\nu}_k(t) = \left| \nu_k(t) - z_{kj}(\sigma_k(t))^*/\nu_k(t) \right|, \quad k = j + 1, \ldots, n
\]

assume \( \tilde{\nu}_k \) are invertible, and let

\[
U_j = [u_j, \bar{u}_j] \cap \text{image}(\tilde{\nu}_{j+1}) \cap \cdots \cap \text{image}(\tilde{\nu}_n).
\]
Tree-Reduction Theorem

Let $P = (T, z, C_1, \ldots, C_n)$ be a curved network. Let $j$ be a reducible node, let $T'$ be the resulting reduction, and w.l.o.g its children are $\{j + 1, \ldots, n\}$. Define

$$\tilde{\nu}_k(t) = |\nu_k(t) - z_{kj}(\sigma_k(t))^*/\nu_k(t)|, \quad k = j + 1, \ldots, n$$

assume $\tilde{\nu}_k$ are invertible, and let

$$U_j = [u_j, u_j] \cap \text{image}(\tilde{\nu}_{j+1}) \cap \cdots \cap \text{image}(\tilde{\nu}_n).$$

Then,

- If $U_j = \emptyset$ then $\mathcal{FF}(P) = \emptyset$. 
Tree-Reduction Theorem

Let $P = (T, z, C_1, \ldots, C_n)$ be a curved network. Let $j$ be a reducible node, let $T'$ be the resulting reduction, and w.l.o.g its children are $\{j + 1, \ldots, n\}$. Define

$$\tilde{\nu}_k(t) = |\nu_k(t) - z_{kj}(\sigma_k(t))^*/\nu_k(t)|, \quad k = j + 1, \ldots, n$$

assume $\tilde{\nu}_k$ are invertible, and let

$$U_j = [u_j, \bar{u}_j] \cap \text{image}(\tilde{\nu}_{j+1}) \cap \cdots \cap \text{image}(\tilde{\nu}_n).$$

Then,

- If $U_j = \emptyset$ then $\mathcal{FF}(P) = \emptyset$.
- Otherwise, the there exist functions $h_{j+1}, \ldots, h_n$, and a curve $C'_j$, such that $v \in \mathcal{FF}_v(P)$ if and only if

$$ (v_1, \ldots, v_j) \in \mathcal{FF}_v(T', z', C_1, \ldots, C_{j-1}, C'_j),$$

$$v_k = h_k(v_j), \quad k = j + 1, \ldots, n$$
Tree-Reduction Theorem - the definition of \( h_k \) and \( C'_j \)

Let

\[
\tilde{\sigma}_k(t) = \sigma_k(t) - z_{kj} \left( \frac{\vert \sigma_k(t) \vert^2}{(\nu_k(t))^2} \right),
\]

\[
\phi_k = \tilde{\sigma}_k \circ \tilde{\nu}_k^{-1},
\]

Then,

\[
h_k(v_j) = v_j - z_{kj} \frac{\phi_k^*(\vert v_j \vert)}{v_j^*},
\]

and \( C'_j = \text{image}(\nu_j, \sigma_j) \) with

\[
\nu_j(t) = (1 - t) \cdot (\min U_j) + t \cdot (\max U_j),
\]

\[
\sigma_j(t) = \begin{cases} 
\hat{s}_j + \sum_{k=j+1}^{n} (\phi_k \circ \nu_j)(t), & j \neq 1, \\
0 & j = 1.
\end{cases}
\]
Tree reduction theorem - example

The big picture ($j = 2$)

$$(v_1, v_2, v_3, v_4)^T \in \mathcal{F}\mathcal{F}_v(P)$$

$\iff$

$$(v_1, v_2)^T \in \mathcal{F}\mathcal{F}_v(P'),$$

$v_3 = h_3(v_2)$,

$v_4 = h_4(v_2)$. 
Tree reduction theorem - example

The big picture \( (j = 2) \)

\[
(v_1, v_2, v_3, v_4)^T \in \mathcal{FF}_v(P) \iff (v_1, v_2)^T \in \mathcal{FF}_v(P'),
\]

\[
v_3 = h_3(v_2),
\]

\[
v_4 = h_4(v_2).
\]

Details:

- Compute \( \tilde{\nu}_3 \) and \( \tilde{\nu}_4 \). Verify invertability.
- Compute \( U_2 = [u_2, \overline{u}_2] \cap \text{image}(\tilde{\nu}_3) \cap \text{image}(\tilde{\nu}_4) \). Verify \( U_2 \neq \emptyset \).
- Compute the functions \( h_3 \) and \( h_4 \), and the curve \( C_2' \).
Tree reduction theorem - example

The big picture \((j = 2)\)

\[
(v_1, v_2, v_3, v_4)^T \in \mathcal{F}\mathcal{F}_v(P) \iff (v_1, v_2)^T \in \mathcal{F}\mathcal{F}_v(P'),
\]

\[
v_3 = h_3(v_2), \\
v_4 = h_4(v_2).
\]

Details:

- Compute \(\tilde{\nu}_3\) and \(\tilde{\nu}_4\). Verify invertability.
- Compute \(U_2 = [u_2, \bar{u}_2] \cap \text{image}(\tilde{\nu}_3) \cap \text{image}(\tilde{\nu}_4)\). Verify \(U_2 \neq \emptyset\).
- Compute the functions \(h_3\) and \(h_4\), and the curve \(C_2'\).
Tree reduction theorem - example

\[(v_1, v_2, v_3, v_4)^T \in \mathcal{FF}_v(P) \]

\[
\iff
\]

\[v_1 \in \mathcal{FF}_v(P'')\),

\[v_2 = h_2(v_1),

\[v_3 = h_3(v_2),

\[v_4 = h_4(v_2).

\[\begin{matrix}
1 & 2 & 3 & 4 \\
\end{matrix}
\]

\[P: \]

\[P'': \]

\[\begin{matrix}
1 & 2 & 3 & 4 \\
\end{matrix}
\]

\[\begin{matrix}
c_1 & c_2 & c_3 & c_4 \\
\end{matrix}
\]

\[\begin{matrix}
z_{12} & z_{23} \\
\end{matrix}
\]

\[\begin{matrix}
c_1' \\
\end{matrix}
\]
Tree reduction theorem - example

\[(v_1, v_2, v_3, v_4)^T \in \mathcal{FF}(P) \iff v_1 \in \mathcal{FF}(P'') \land v_2 = h_2(v_1) \land v_3 = h_3(v_2) \land v_4 = h_4(v_2).\]

Observation

\[C_1' = U_1 \times \{0\}\]
Tree reduction theorem - example

\[(v_1, v_2, v_3, v_4)^T \in \mathcal{F\mathcal{F}}_v(P)\]

\[\iff\]

\[v_1 \in \mathcal{F\mathcal{F}}_v(P''),\]

\[v_2 = h_2(v_1),\]

\[v_3 = h_3(v_2),\]

\[v_4 = h_4(v_2).\]

Observation

\[\Rightarrow C'_1 = U_1 \times \{0\}\]

\[\Rightarrow \mathcal{F\mathcal{F}}(P'') = \{(v_1, s_1) : |v_1| \in U_1, s_1 = 0, \arg(v_1) = 0\}\]
Tree reduction theorem - example

\[(v_1, v_2, v_3, v_4)^T \in \mathcal{FF}_v(P) \iff v_1 \in \mathcal{FF}_v(P'') \]
\[v_2 = h_2(v_1),\]
\[v_3 = h_3(v_2),\]
\[v_4 = h_4(v_2).\]

Observation

\[C'_1 = U_1 \times \{0\}\]
\[\implies \mathcal{FF}(P'') = \{(v_1, s_1) : |v_1| \in U_1, s_1 = 0, \ \text{arg}(v_1) = 0\}\]
\[\implies \mathcal{FF}_v(P'') = U_1.\]
Conclusion

\[(v_1, v_2, v_3, v_4)^T \in \mathcal{FF}_v(P) \iff v_1 \in U_1, v_2 = h_2(v_1), v_3 = h_3(v_2), v_4 = h_4(v_2).\]

Invertability \(\implies\) the feasible set is a curve \(\gamma_1 : [0, 1] \rightarrow \mathbb{R} \times \mathbb{C}\)

The above is an algorithm to evaluate \(\gamma_1(t)\)
Two-phase meta-algorithm

Phase 1 - tree reduction
- Perform a sequence of reductions, until one-node network is obtained.
- Compute $h_k$ functions ($\phi_k$ in practice) and $U_j$ intervals.
- If any $U_j = \emptyset$ - return ”problem infeasible”
- If any $\tilde{\nu}_k$ non-invertible - return ”error”

Phase 2 - tree expansion
- Take any $\nu_1 \in U_1$.
- Use expand $\nu_1$ to the full $\nu$ vector using $h_k$ functions.
- Compute the $s$ vector.
Inverting $\tilde{\nu}_k$
The remedy - spline approximation

Choose approx. density $d$. Let $t = \frac{1}{d-1}(0, 1, 2, \ldots, d - 1)$.

- $\nu_k$ and $\sigma_k$ are approximated by vectors:

$$\nu_k = (\nu_k(t_1), \ldots, \nu_k(t_d)),$$
$$\sigma_k = (\sigma_k(t_1), \ldots, \sigma_k(t_d)).$$

- Approximations of $\tilde{\nu}_k$ and $\tilde{\sigma}_k$ are computed componentwise:

$$\tilde{\nu}_k = \left| \nu_k - z_{kj} \frac{(\sigma_k)^*}{\nu_k} \right|,$$
$$\tilde{\sigma}_k = \ldots$$

- $\phi_k = \tilde{\sigma}_k \circ \tilde{\nu}_k^{-1}$ is approximated by a cubic spline interpolant mapping the components of $\tilde{\nu}_k$ to $\tilde{\sigma}_k$. 
Multiple curves

\( \tilde{\nu}_k(t) \) invertible \( \iff \) it is strictly monotone.

Clearly, \( \tilde{\nu}_k(t) \) is invertible on \( I_1, I_2, \) and \( I_3. \)

Solution: Split the network \( P \) into 3 networks.
Multiple curves

$\tilde{\nu}_k(t)$ invertible $\iff$ it is strictly monotone.

Clearly, $\tilde{\nu}_k$ is invertible on $I_1$, $I_2$, and $I_3$. 

Solution: Split the network $P$ into 3 networks.
Multiple curves

\[ \tilde{\nu}_k(t) \text{ invertible } \iff \text{ it is strictly monotone.} \]

Clearly, \( \tilde{\nu}_k \) is invertible on \( l_1, l_2, \) and \( l_3. \)
Solution: Split the network \( P \) into 3 networks.
Multiple curves - cont.

Eventually: multiple terminal single-node networks
Multiple curves - cont.

Eventually: multiple terminal single-node networks
Multiple curves - cont.

Eventually: multiple terminal single-node networks

Reduction Split
Multiple curves - cont.

Eventually: multiple terminal single-node networks
Eventually: multiple terminal single-node networks
Multiple curves - cont.

Eventually: multiple terminal single-node networks
Numerical results
Experiment setup

- Modified to conform to the assumptions
  - PV constraints on leaves only.
  - $z_{ij}$ for each edge $(i,j) \in E$.
- Several tests
  - Accuracy - distance from the feasible set w.r.t the approximation density.
  - Reliability - comparison with MATPOWER\textsuperscript{3}.
  - Number of parallel networks in practice.

\textsuperscript{3}A well-known PF and OPF solver. Stability function can be specified using the extension mechanism.
Accuracy

Test setup

▶ Check several approximation densities $d \in [2^3, 2^{12}]$.
▶ For each approximation density, sample each curve at $N = 1000$ points.
▶ For each deviation measure, report the highest value among the samples.

Deviation measures

<table>
<thead>
<tr>
<th>PQ constraints</th>
<th>PV constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_{PQ,v}(v, s) = \max_{j \in PQ} d \left(</td>
<td>v_j</td>
</tr>
<tr>
<td>$E_{PQ,s}(v, s) = \max_{j \in PQ}</td>
<td>s_j - \hat{s}_j</td>
</tr>
<tr>
<td></td>
<td>$E_{PV,q}(v, s) = \max_{j \in PV} d \left( \text{im}(s_j), q_j, \bar{q}_j \right)$</td>
</tr>
</tbody>
</table>
Accuracy

$E_{PQ,s}$

$E_{PQ,s}$, $v$ and $E_{PV,q}$ were zero in all our experiments.
Accuracy

$E_{PV,p}$

$EPV,p$

$13b$

$34b$

$37b$

$47b$

$69b$

$123b$

$13  24  25  26  27  28  29  30  31  32$

$-45$

$-40$

$-35$

$-30$

$-25$

$-20$

$-15$

$-10$

$E_{PQ}$, $v$ and $E_{PV,q}$ were zero in all our experiments.
Accuracy

$E_{PV,v}$

$\text{EPV,v}$, $\text{v}$ and $E_{PV,q}$ were zero in all our experiments.
Accuracy

$E_{PV,v}$ and $E_{PV,q}$ were zero in all our experiments.
Reliability

Test setup

- Perturb the PQ constraints of each network:

\[ [u_j, \bar{u}_j] \times \{ \text{re}(\hat{s}_j) \cdot \alpha_j + (\text{im}(\hat{s}_j) \cdot \beta_j)\iota \}, \]

where \( \alpha_j, \beta_j \sim U[0, 2] \).

- Generate 5000 random networks from each existing network.

- Solve OPF with the “stability” objective function:

\[ f(v, s) = \sum_{j \in PQ} \left| v_j - \frac{1}{2}(u_j + \bar{u}_j) \right| \]

- Solve using both MATPOWER and our solver.

- Gather statistics.
Reliability

<table>
<thead>
<tr>
<th>Network Size</th>
<th>% of Random Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>13-node</td>
<td>6.46%</td>
</tr>
<tr>
<td>34-node</td>
<td>5.22%</td>
</tr>
<tr>
<td>37-node</td>
<td>2.56%</td>
</tr>
<tr>
<td>47-node</td>
<td>2.78%</td>
</tr>
<tr>
<td>69-node</td>
<td>2.34%</td>
</tr>
<tr>
<td>123-node</td>
<td>39.64%</td>
</tr>
</tbody>
</table>

The % of random networks, generated from each original network, for which our method found a solution while MATPOWER did not.
Number of networks in parallel

**Observation:** $\tilde{\nu}_k$ becomes more ‘wild’ when $|z|$ increases:

$$\tilde{\nu}_k(t) = \left| \nu_k(t) - z_{kj}(\sigma_k(t))^*/\nu_k(t) \right|$$
Number of networks in parallel

Observation: \( \tilde{\nu}_k \) becomes more ‘wild’ when \(|z|\) increases:

\[
\tilde{\nu}_k(t) = |\nu_k(t) - z_{kj}(\sigma_k(t))^*/\nu_k(t)|
\]

Setup: Make \( \tilde{\nu}_k \) non-invertible my replacing \( z \) with \( \alpha z \) for \( \alpha \in [1, 10] \).
Number of networks in parallel

Observation: \( \tilde{\nu}_k \) becomes more ‘wild’ when \(|z|\) increases:

\[
\tilde{\nu}_k(t) = |\nu_k(t) - z_{kj}(\sigma_k(t))^* / \nu_k(t)|
\]

Setup: Make \( \tilde{\nu}_k \) non-invertible my replacing \( z \) with \( \alpha z \) for \( \alpha \in [1, 10] \).

Results - Maximum (worst) number of networks in parallel

<table>
<thead>
<tr>
<th>13-node</th>
<th>34-node</th>
<th>37-node</th>
<th>47-node</th>
<th>69-node</th>
<th>123-node</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Questions?