Exploiting Dense Sensitivity Matrices in Linear Optimal Power Flow

Brent Eldridge, Johns Hopkins University
Anya Castillo, NextEra Analytics
Ben Knueven, National Renewable Energy Laboratory
Manuel Garcia, Sandia National Laboratories
Outline

Background

Linear OPF formulations

Why are dense constraints better for large-scale DCOPF?

Modeling simplifications

Preliminary results

Funding from U.S. Department of Energy’s Office of Electricity through the Advanced Grid Modeling program under project title “Reassessing the Market—Computation Interface to Enhance Grid Security and Efficiency.”
Optimal Power Flow underlies many market applications.

- Unit commitment
- Contingency analysis
- Real time dispatch
- Transmission switching
Comparison of AC and DC approaches

<table>
<thead>
<tr>
<th>AC OPF-based models</th>
<th>DC OPF-based models</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Physics is correct</td>
<td>• Physics is approximated</td>
</tr>
<tr>
<td>• Nonlinear, nonconvex</td>
<td>• Linear, convex</td>
</tr>
<tr>
<td>• Uses real and reactive power</td>
<td>• Only considers real power</td>
</tr>
<tr>
<td>• Relaxation (SDR, SOCR, QCR) to</td>
<td>• B-theta and distribution factor</td>
</tr>
<tr>
<td>convex program</td>
<td>(PTDF) formulations</td>
</tr>
<tr>
<td>• Promising approach, <em>but...</em></td>
<td>• B-theta common in academic</td>
</tr>
<tr>
<td>• Poor scaling in large systems</td>
<td>literature</td>
</tr>
<tr>
<td>• Nonlinear</td>
<td>• <em>PTDF used in ISO applications</em></td>
</tr>
<tr>
<td>• Unsuitable for many applications</td>
<td></td>
</tr>
</tbody>
</table>
Optimal power flow is not solved in isolation

• Consider the “outer loop”
• PTDF advantages:
  • Linear
  • Compact formulation
  • Nonbinding constraints can be dropped
• AC OPF and B-theta require the full explicit formulations
  • Huge model size for AC OPF and AC OPF relaxations (SDP, etc.)
Research Questions for Linear OPF

How accurate can linear OPF models be?

What is the computational “cost” of more accurate OPF approximations?

What kind of simplifications can improve computational performance of dense OPF formulations?

What is the impact on solution accuracy?
Dense Optimal Power Flow: Distribution factor (PTDF) derivation

- Standard DC power flow assumptions result in:
  \[ p^f = -BA\theta \]

- Substitution into linear power balance constraints:
  \[ p^d - p^g - A^\top BA\theta = 0 \]

- Define the PTDF as:
  \[ \text{PTDF} := -BA(A^\top BA)^{-1} \Rightarrow p^f = \text{PTDF}(p^d - p^g) \]

**Issue**: the B-theta power flow constraint matrix \(-BA\) is *sparse*, so inverting and calculating \(\text{PTDF}\) results in a *dense* constraint matrix.

**Notation:**
- Power flow: \( p^f \in \mathbb{R}^K \)
- Power demand: \( p^d \in \mathbb{R}^N \)
- Power generation: \( p^g \in \mathbb{R}^N \)
- Voltage angle: \( \theta \in \mathbb{R}^N \)
- Adjacency matrix: \( A \in \mathbb{R}^{K \times N} \)
- Susceptance mat.: \( B \in \mathbb{R}^{K \times K} \)
Four proposed models: Getting more accuracy out of linear OPF

**Sparse (S-LOPF)**
- First-order Taylor series approximation of decoupled AC power flow
- Mid-line power flow (Garcia et al., 2019)

**Dense (D-LOPF)**
- Dense factorization of the S-LOPF

**Condensed (C-LOPF)**
- Summation of branch losses to a system-wide loss variable
- Loss distribution factors to approximate effect of losses on branch flows

**Real power (P-LOPF)**
- Assume fixed voltages and reactive power flow
- Basically AC-linearized DC OPF
Sparse Linear OPF model (S-LOPF)

\[
\begin{align*}
\min & \sum_i C_i(p_i^g), \\
\text{s.t.:} & \\
p^d - p^g + A^\top p^f + \frac{1}{2} |A|^\top p^\ell &= 0 \\
q^d - q^g + A^\top q^f + \frac{1}{2} |A|^\top q^\ell &= 0 \\
p^f &= F\theta + F^0 \\
q^f &= G\nu + G^0 \\
p^\ell &= L\theta + L^0 \\
q^\ell &= K\nu + K^0 \\
(p^f, q^f, p^\ell, q^\ell, \nu) &\in \mathcal{I} \\
(p^g, q^g) &\in \mathcal{G}
\end{align*}
\]

- Cost minimization
- Linear balance constraints
- First-order Taylor series expansion of power flow and branch losses
  - \(F, G, L, K\) and offsets are calculated from the AC power flow equations
  - Analogous to \(PTDF\)
- Transmission constraints \(\mathcal{I}\)
  - Piece-wise linear apparent power limits
  - Upper/lower voltage magnitude limits
- Generator constraints \(\mathcal{G}\)
Dense Linear OPF model (D-LOPF)

\[
\begin{align*}
\min & \sum_i c_i(p^g_i), \\
\text{s.t.:} & \\
1^\top(p^d - p^g) + 1^\top p^\ell = 0 \\
p^f &= \tilde{F}(p^d - p^g) + \tilde{F}^0 \\
q^f &= \tilde{G}(q^d - q^g) + \tilde{G}^0 \\
p^\ell &= \tilde{L}(p^d - p^g) + \tilde{L}^0 \\
q^\ell &= \tilde{K}(q^d - q^g) + \tilde{K}^0 \\
v &= \tilde{S}(q^d - q^g) + \tilde{S}^0 \\
(p^f, q^f, p^\ell, q^\ell, v) &\in \mathcal{T} \\
(p^g, q^g) &\in \mathcal{G}
\end{align*}
\]

- Cost minimization
- System real power balance
  - Reactive balance is unnecessary
- Redefine \( p^f, q^f, p^\ell, q^\ell, \) and \( v \) using PTDF-like distribution factors

\[
\tilde{F} := -F \left( A^\top F + \frac{1}{2} |A|^\top L \right)^{-1}
\]

- Other constraints stay the same
Condensed Linear OPF model (C-LOPF)

\[
\text{min } \sum \limits_i C_i(p^g_i), \quad \text{s.t.:}
\]
\[
1^\top (p^d - p^g) + p^\ell = 0
\]
\[
p^f = \hat{F}(p^d - p^g) + \hat{F}^0
\]
\[
q^f = \hat{G}(q^d - q^g) + \hat{G}^0
\]
\[
p^\ell = \hat{L}(p^d - p^g) + \hat{L}^0
\]
\[
q^\ell = \hat{K}(q^d - q^g) + \hat{K}^0
\]
\[
v = \hat{S}(q^d - q^g) + \hat{S}^0
\]
\[
(p^f, q^f, p^\ell, q^\ell, v) \in \mathcal{T}
\]
\[
(p^g, q^g) \in \mathcal{G}
\]

- Cost minimization
- System power balance with system power losses
- Reduce number of constraints by summing \( p^\ell \) and \( q^\ell \) constraints
  \[
  \bar{L} := 1^\top \hat{L}, \quad \bar{L}^0 := 1^\top \hat{L}^0
  \]
- Approximate branch losses with loss distribution factor in \( \mathcal{T} \) constraints
- Generator constraints stay the same
Real Power Linear OPF model (P-LOPF)

\[
\begin{align*}
\min \sum_i C_i(p_i^g), & & s.t.: \\
1^\top (p^d - p^g) + p^\ell = 0 \\
p^f = \hat{F}(p^d - p^g) + \hat{F}^0 \\
p^\ell = \bar{L}(p^d - p^g) + \bar{L}^0 \\
(p^f, q^f, p^\ell, q^\ell, v) & \in \mathcal{T} \\
(p^g, q^g) & \in \mathcal{G}
\end{align*}
\]

- Cost minimization
- System power balance with system power losses
- Remove reactive power and voltage magnitude constraints
- Fix reactive power and voltage variables in $\mathcal{T}$ constraints
- Generator constraints stay the same
Methodology

Solve AC OPF
- Basepoint solution
- Solved using IPOPT

Calculate Sensitivities
- Real/Reactive power flow and line losses
- Voltage magnitude
- Sparse and dense calculations

Change Demand
- Uniform multiplier from 0.9 to 1.1
- 0.01 increments

Solve LOPF
- S/D/C/P-LOPF
- B-theta/PTDF
- Solved using Gurobi

Solve AC Power Flow
- Calculate power flow errors and violations
- Simulates “actual” dispatch from OPF solution

Extensive set of test cases:

PRELIMINARY results use full methodology to case588_sdet, nominal demand only to case6515_rte

- `pglib_opf_case3_lmbd`
- `pglib_opf_case5_pjm`
- `pglib_opf_case14_ieee`
- `pglib_opf_case24_ieee_rts`
- `pglib_opf_case30_as`
- `pglib_opf_case30_fsr`
- `pglib_opf_case30_ieee`
- `pglib_opf_case39_epri`
- `pglib_opf_case57_ieee`
- `pglib_opf_case73_ieee_rts`
- `pglib_opf_case89_pegase`
- `pglib_opf_case118_ieee`
- `pglib_opf_case162_ieee_dtc`
- `pglib_opf_case179_goc`
- `pglib_opf_case200_tamu`
- `pglib_opf_case240_pserc`
- `pglib_opf_case300_ieee`
- `pglib_opf_case500_tamu`
- `pglib_opf_case588_sdet`
- `pglib_opf_case934_sdet`
- `pglib_opf_case1354_pjm`
- `pglib_opf_case1790_pjm`
- `pglib_opf_case1951_rte`
- `pglib_opf_case2000_tamu`
- `pglib_opf_case2000_tamu_dtc`
- `pglib_opf_case2000_tamu_k`
- `pglib_opf_case2160_pserc`
“Vanilla” Implementations: DC OPF / LOPF / AC OPF

Solution Speed - cases with 3-588 buses

AC Power Flow Error - case2383wp_k

PRELIMINARY RESULTS
The lazy constraint algorithm can be implemented on “dense” OPFs

- Implemented with Gurobi’s persistent solver
- D/C/P-LOPF and PTDF formulation of the DC OPF
- Reduces transmission constraints to around 5% of full model
- Reduces overall model size to about 50%

Not possible on sparse (S-LOPF/B-theta) formulations

- Removing line results in a new topology
Lazy algorithm benefits cases with >100 buses

The lazy algorithm only provides improvement on larger cases
Hybrid line loss constraints for D-LOPF

• Lazy algorithm does NOT allow us to ignore branch loss constraints

\[ \sum_i (p_i^d - p_i^g) + \sum_{k \in \mathcal{K}'} p_k^l + \sum_{k \in \mathcal{K} \setminus \mathcal{K}'} (p_k^l = 0) = 0 \]

• Solution: implement a hybrid model with residual losses

\[ \sum_i (p_i^d - p_i^g) + \sum_{k \in \mathcal{K}'} p_k^l + p_{\text{resid}} = 0 \]

Result is similar to C-LOPF formulation

(case3375wp_k)

PRELIMINARY RESULTS
Loss summation in C-LOPF and hybrid D-LOPF can cause accumulation of power flow errors.
Factor truncation improves constraint sparsity but can also accumulate power flow errors

• Dense models can be made more “sparse” by eliminating small coefficients

\[
\hat{F}_{ik}^\varepsilon = \hat{F}_{ik} \{ \hat{F}_{ik} < \varepsilon \} \\
\hat{F}_{ik}^{0,\varepsilon} = \hat{F}_{ik}^0 + \sum_{\{ i : \hat{F}_{ik} \geq \varepsilon \}} \hat{F}_{ik} (p_i^d - p_i^g)
\]

• Errors are usually small

PRELIMINARY RESULTS
Factor truncation improves constraint sparsity but can also accumulate power flow errors

- Dense models can be made more “sparse” by eliminating small coefficients
  
  \[
  \hat{F}_{ik}^\varepsilon = \hat{F}_{ik}1\{\hat{F}_{ik} < \varepsilon\} \\
  \hat{F}_{ik}^{0,\varepsilon} = \hat{F}_{ik}^0 + \sum_{i: \hat{F}_{ik} \geq \varepsilon} \hat{F}_{ik}(p_i^d - p_i^g)
  \]

- Errors are usually small

PRELIMINARY RESULTS
Conclusions
How accurate can LOPF results be?

Normalized total cost with nominal demand (up to case6515_rte)

A few cases with >1% gap

*Infeasible

PRELIMINARY RESULTS
What is the computational “cost”? 

Normalized solution time at nominal demand (up to case6515_rte) 

Numerical (?) difficulties in specific cases

*Infeasible
Conclusions

• Benefits of AC-linearized optimal power flow models
  • vs DC OPF: power flow accuracy
  • vs AC OPF: computational speed
  • Based on use of state estimator data in real-world applications

• Simplifications required for good “PTDF” implementations
  • Active set algorithms
  • Line loss relaxation (system/residual losses)
  • Factor truncation
  • Not trivial... Lots of trial and error

• Future work:
  • Preprocessing memory & speed improvements
  • More intelligent active set implementation
  • Computational performance in unit commitment models
Questions