Aug 8 – 11AM ET | ACADEMIC WEBINAR

The AC Optimal Power Flow Problem – Part 2

Diving Deeper into the AC Optimal Power Flow Problem and Showcasing the New Power Flow OptiMod Python Optimization Use Case

Dr. Daniel Bienstock
Columbia University

GUROBI OPTIMIZATION

Once again: Practical Optimization at a Crossroads

- Current and **past** areas of interest: logistics, transportation, supply chain, advertising, pricing, yield management, market analytics, mechanism design, ...
- These areas will remain relevant, but ...
- The **future**: **heavy engineering, hard science.**
- Very nonlinear, complex models that embody hard, inflexible rules.
- Very large scale, high level of modeling detail, myriad details in complex systems.
- **Demanding** performance requirements: must get **good** solutions **fast.**
- Traditional optimization and Operations Research must have a stake in this domain.

Outline For This Talk

- Review of basic AC and DC Optimal Power flow.
- Relaxations and linear relaxations* – computational experiments! *
- GO competitions run by the Department of Energy. *
- Gurobi Optimods demonstration. *

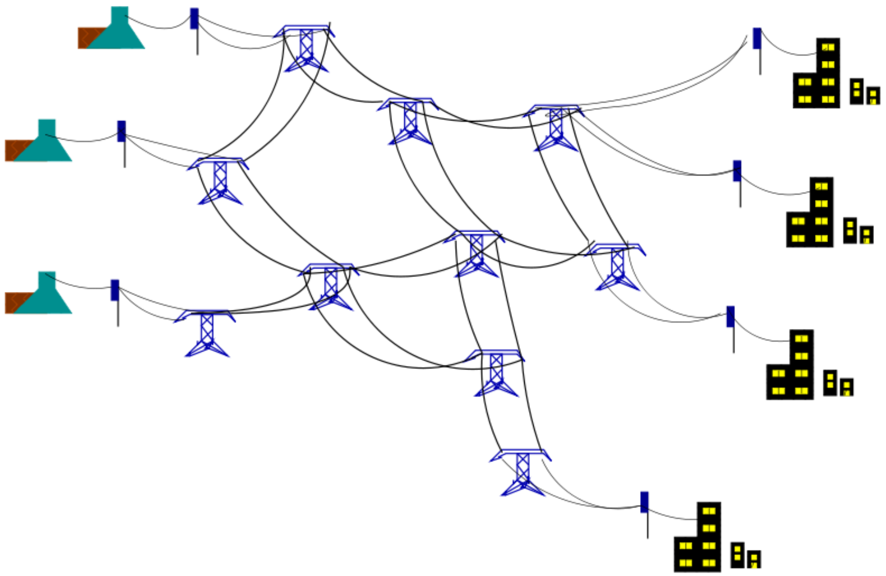
* = new content as compared to first webinar

GUROBI OPTIMIZATION

Standard ACOPF

We are given a power system, i.e., a network of

- Generators
- Power lines and transformers
- Buses (nodes)
- Each bus has a load, i.e., numerical demand for power generators, lines, transformers and buses (nodes) with power demands



Objective: meet demands at minimum cost

Cost incurred at generators

Note: power flows following **laws of physics**

GUROBI OPTIMIZATION

A formal textbook statement of standard ACOPF

Minimize cost of generation: $\sum_{g \in \mathcal{G}} F_g(P^g)$

- Here, $\mathcal{G}$ is the set of generators
- P^g is the (active) power generated at g
- F_g is generation cost at g – convex, piecewise-linear or quadratic
  Example: $F_g(P) = 3P^2 + 2P$

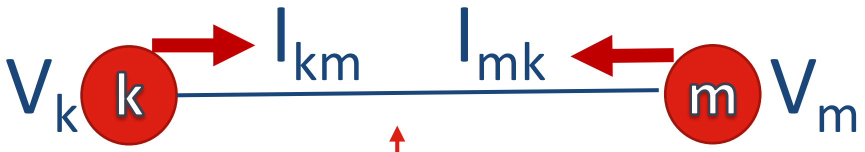
Constraints:

- PF (power flow) constraints: choose voltages so that network delivers power from generators to the loads, following **AC power flow laws**

- Voltage magnitudes are constrained

- Power flow on any line km cannot be too large
- The output of any generator is limited

GUROBI OPTIMIZATION



$$Y = \begin{pmatrix} G_{kk} + jB_{kk} & G_{km} + jB_{km} \\ G_{mk} + jB_{mk} & G_{mm} + jB_{mm} \end{pmatrix}$$

Admittance matrix for line km

$$I_{km} = Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \rightarrow S_{km} = V_k I_{km}^*$$

Ohm's Law

Complex current

Complex power

S_{km} != $-S_{mk}$

S_{km} = **50.0** + **20.0** j
S_{mk} = - 49.9 – 21.02 j
i.e.,
Complex power **arriving** at m =
$-S_{mk}$ = **49.9 + 21.02 j**
0.1 **loss** in active power
1.02 **gain** in reactive

$$V_k = |V_k| e^{j\theta_k}$$

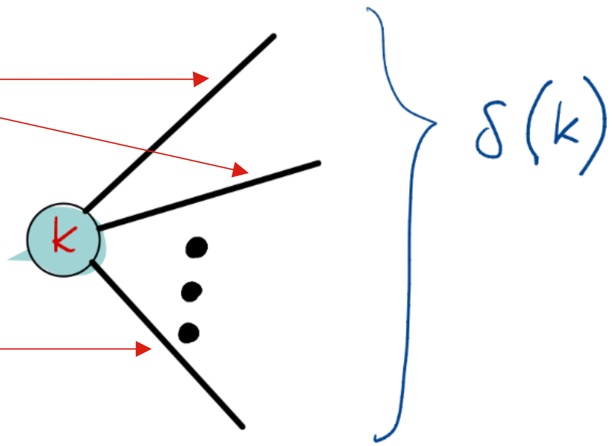
$$S_{km} = V_k \left[Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \right]^*$$

$$\theta_{km} \doteq \theta_k - \theta_m$$

$$V_m = |V_m| e^{j\theta_m}$$

$$S_{km} = (G_{kk} - jB_{kk})\,|V_k|^2 + (G_{km} - jB_{km})\,|V_k||V_m|(\cos\theta_{km} + j\sin\theta_{km})$$

GUROBI OPTIMIZATION



"Reactive" power generation/demand at **k**

$$\sum_{km\in\delta(k)} \boldsymbol{S_{km}} = \left(\sum_{i\in\mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d\right) + j\left(\sum_{i\in\mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d\right)$$

LHS = complex power injected into grid at **k**

Real power demand at **k**

Total real ("active") power generated at **k**

GUROBI OPTIMIZATION

Basic ACOPF

Minimize $\sum_{i\in\mathcal{G}} F_i(\boldsymbol{P_i^g})$

with constraints:

$\forall$ branch km: $\boldsymbol{S_{km}} =$
$(G_{kk} - jB_{kk})\ |\boldsymbol{V_k}|^{\boldsymbol{2}} + (G_{km} - jB_{km})\ |\boldsymbol{V_k}||\boldsymbol{V_m}|(\boldsymbol{\cos\theta_{km}} + \boldsymbol{j}\ \boldsymbol{\sin\theta_{km}})$

$\sum_{km\in\delta(k)} \boldsymbol{S_{km}} = (\sum_{i\in\mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d) + j(\sum_{i\in\mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d)$

Power flow limit on branch km:
$|\boldsymbol{S_{km}}|^{\boldsymbol{2}} = \mathcal{R}\mathrm{e}(\boldsymbol{S_{km}})^2 + Im(\boldsymbol{S_{km}})^2 \leq U_{km}$

Voltage limit on bus k:
$V_k^{\min} \leq |\boldsymbol{V_k}| \leq V_k^{\max}$

Generator output limits:
$P_i^{\min} \leq \boldsymbol{P_i^g} \leq P_i^{\max}$

GUROBI OPTIMIZATION

But there is an equivalent formulation as a **QCQP**
(Quadratically Constrained Quadratic Program)

$V_k = |V_k| e^{j\theta_k}$

$V_m = |V_m| e^{j\theta_m}$

$\theta_{km} \doteq \theta_k - \theta_m$

$$S_{km} = (G_{kk} - jB_{kk})\,|V_k|^2 + (G_{km} - jB_{km})\,|V_k||V_m|(\cos\theta_{km} + j\sin\theta_{km})$$

$$S_{km} = V_k \left[Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \right]^*$$

$$Y = \begin{pmatrix} G_{kk} + jB_{kk} & G_{km} + jB_{km} \\ G_{mk} + jB_{mk} & G_{mm} + jB_{mm} \end{pmatrix}$$

Admittance matrix for line km

$$V_i = e_i + j f_i$$

Use rectangular coordinates for voltages

GUROBI OPTIMIZATION

ACOPF as a **QCQP**

$$S_{km} = (G_{kk} - jB_{kk})\,|V_k|^2 + (G_{km} - jB_{km})\,|V_k||V_m|(\cos\theta_{km} + j\sin\theta_{km})$$

$$V_i = e_i + j f_i$$

Use rectangular coordinates for voltages

Real ("active") part

$$P_{km} = G_{kk}(e_k^2 + f_k^2) + G_{km}(e_k e_m + f_k e_m) + B_{km}(-e_k f_m + f_k e_m)$$

Imaginary ("reactive") part

$$Q_{km} = -B_{kk}(e_k^2 + f_k^2) - B_{km}(e_k e_m + f_k e_m) + G_{km}(-e_k f_m + f_k e_m)$$

GUROBI OPTIMIZATION

A common simplification: the DC approximation

$\forall$ branch km: $\boldsymbol{S_{km}} =$
$(G_{kk} - jB_{kk})\ |\boldsymbol{V_k}|^2 + (G_{km} - jB_{km})\ |\boldsymbol{V_k}||\boldsymbol{V_m}|(\cos \boldsymbol{\theta_{km}} + \boldsymbol{j} \sin \boldsymbol{\theta_{km}})$

- $|\boldsymbol{V_k}| = \mathbf{1}$ for all buses $\boldsymbol{k}$. Why?
- $\cos \boldsymbol{\theta_{km}} = \mathbf{1}$ and $\sin \boldsymbol{\theta_{km}} = \boldsymbol{\theta_k} - \boldsymbol{\theta_m}$ for all branches $\boldsymbol{km}$.
- Ignore reactive power.

GUROBI OPTIMIZATION

DC approximation

Minimize $\sum_{i \in \mathcal{G}} F_i(\,\boldsymbol{P_i^g})$

with constraints:

$\forall$ branch km: $\quad \boldsymbol{P_{km}} = y_{km}(\,\boldsymbol{\theta_k - \theta_m})$

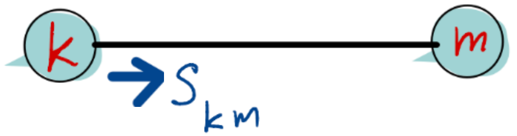
$\sum_{km \in \delta(k)} \boldsymbol{P_{km}} = (\sum_{i \in \mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d)$

Power flow limit on branch km:
$|\boldsymbol{P_{km}}| \leq U_{km}$

Generator output limit on bus k:
$P_k^{\min} \leq \boldsymbol{P_k^g} \leq P_k^{\max}$

GUROBI OPTIMIZATION

ACOPF as a **QCQP**
(Quadratically Constrained Quadratic Program)



complex power **injected** into **km** at **k**

$$V_k = |V_k| e^{j\theta_k}$$

$$V_m = |V_m| e^{j\theta_m}$$

$$\theta_{km} \doteq \theta_k - \theta_m$$

$$\boldsymbol{S_{km}} = (G_{kk} - jB_{kk})\, |\boldsymbol{V_k}|^2 + (G_{km} - jB_{km})\, |\boldsymbol{V_k}||\boldsymbol{V_m}|(\cos \boldsymbol{\theta_{km}} + \boldsymbol{j} \sin \boldsymbol{\theta_{km}})$$

$$S_{km} = V_k \left[Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \right]^*$$

$$Y = \begin{pmatrix} G_{kk} + jB_{kk} & G_{km} + jB_{km} \\ G_{mk} + jB_{mk} & G_{mm} + jB_{mm} \end{pmatrix}$$

Admittance matrix for line km

$$V_i = e_i + j f_i$$

Use rectangular coordinates for voltages

GUROBI OPTIMIZATION

Solving convex relaxations of ACOPF

- McCormick relaxation
- Spatial branching
- Outer approximation of convex quadratics
- Cuts

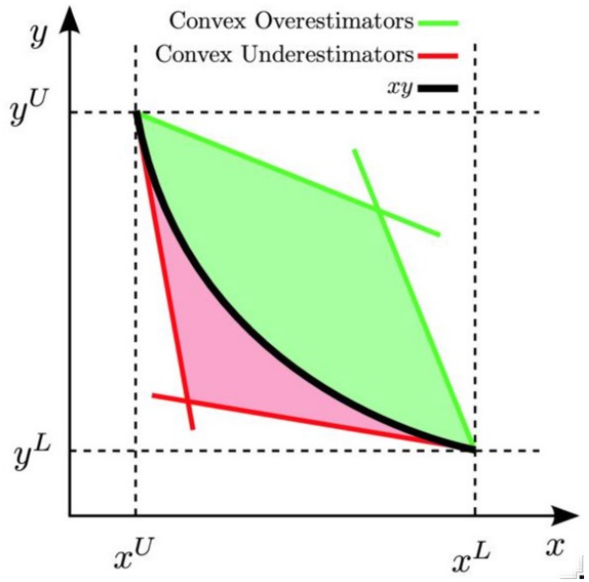
GUROBI OPTIMIZATION

McCormick relaxation - an important workhorse

Suppose a formulation has a bilinear expression, xy

$$x^L \leq x \leq x^U, \quad y^L \leq y \leq y^U$$

Then: introduce a new variable, w. Convex hull of all points (x,y,w} provided by under/over estimators



The underestimators of the function are represented by:

$$w \geq x^L y + x y^L - x^L y^L \;;\; w \geq x^U y + x y^U - x^U y^U$$

The overestimators of the function are represented by:

$$w \leq x^U y + x y^L - x^U y^L \;;\; w \leq x y^U + x^L y - x^L y^U$$

In formulation, replace each occurrence of xy with the variable w.
And write the four inequalities involving w, x and y.

(source: Wikipedia)

Works well in tandem with ***spatial branching.***

GUROBI OPTIMIZATION

Spatial branching

- An adaptation of classical branch-and-bound.

- Example: suppose that we have a variable x with

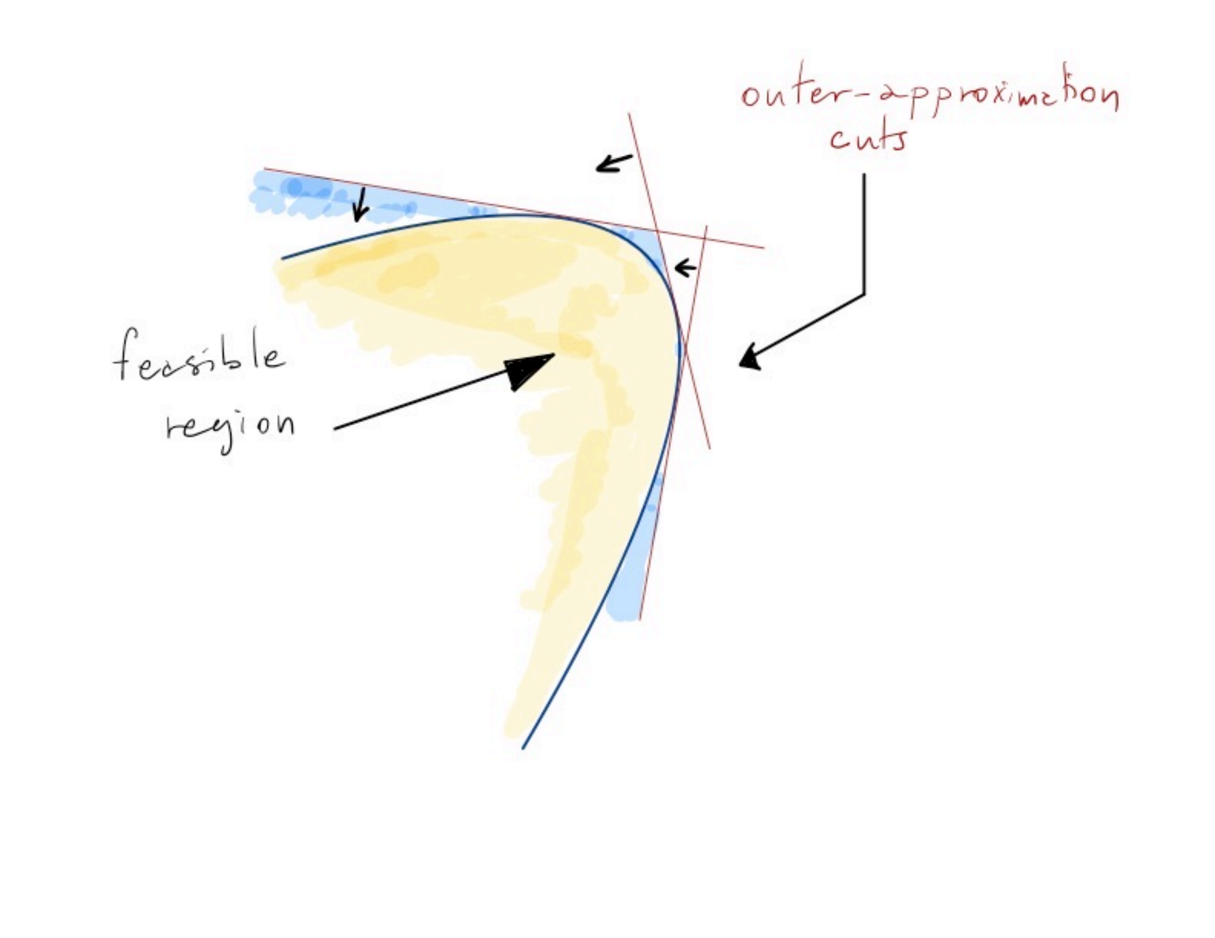
  Suppose that the solution to a relaxation has x = 0.65

$$0.5 \leq x \leq 1.0$$

- Then we *branch*:
  - Branch 1:
  - Branch 2: $0.5 \leq x \leq 0.65$
  - McCormick inequalities tightened in both branches.

$$0.65 \leq x \leq 1.0$$

Outer envelope approximation of convex quadratics



Why needed?

Potential pitfalls:

- Many cuts may be needed to accurately approximate a quadratic.
- Cuts may be nearly parallel
- Many quadratics may need to be accurately approximated

- We may succeed in revealing a global feature, at a cost

- Consequently, the resulting linearly constrained problem may prove challenging

And how well does spatial branching work on ACOPF?

Case	Root relaxation	300 seconds	Interior Point Knitro	Interior point time (s)
9	2264.30	5301.40*	5296.69	0.24
30	0.00	154.08	576.89	0.47
118	0.00	0.00	129660.69	0.24
1354pegase	23037.69	23037.69	74069.35	2.45
ACTIVSg2000	649917.91	649917.91	1228892.08	3.01

(Gurobi 10 on QCQP)

*Why is **lower bound** so bad?*

How about upper bounds … using spatial branching?

GUROBI OPTIMIZATION

A critical observation

$$\boldsymbol{S_{km}} = (G_{kk} - jB_{kk})\ |\boldsymbol{V_k}|^2 + (G_{km} - jB_{km})\ |\boldsymbol{V_k}||\boldsymbol{V_m}|(\cos\boldsymbol{\theta_{km}} + \boldsymbol{j}\sin\boldsymbol{\theta_{km}})$$

Real ("active") part

$$\boldsymbol{P_{km}} = G_{kk}\ |\boldsymbol{V_k}|^2 + G_{km}\ |\boldsymbol{V_k}||\boldsymbol{V_m}|\cos\boldsymbol{\theta_{km}} + B_{km}\ |\boldsymbol{V_k}||\boldsymbol{V_m}|\sin\boldsymbol{\theta_{km}}$$

Introduce new variables: $\boldsymbol{v_k^{(2)}} = |\boldsymbol{V_k}|^2,\quad \boldsymbol{c_{km}} = |\boldsymbol{V_k}||\boldsymbol{V_m}|\cos\boldsymbol{\theta_{km}},\quad \boldsymbol{s_{km}} = |\boldsymbol{V_k}||\boldsymbol{V_m}|\sin\boldsymbol{\theta_{km}}$

$$\boldsymbol{P_{km}} = G_{kk}\ \boldsymbol{v_k^{(2)}} + G_{km}\ \boldsymbol{c_{km}} + B_{km}\ \boldsymbol{s_{km}}$$

And! $$\boldsymbol{c_{km}^2} + \boldsymbol{s_{km}^2} \leq \boldsymbol{v_k^{(2)}}\boldsymbol{v_m^{(2)}}$$ **Jabr inequality**

New variables can be related to rectangular coordinates for voltages

$$\boldsymbol{P_{km}} = G_{kk}(\ \boldsymbol{e_k^2} + \boldsymbol{f_k^2}) + G_{km}(\ \boldsymbol{e_k}\boldsymbol{e_m} + \boldsymbol{f_k}\boldsymbol{e_m}) + B_{km}(\ -\boldsymbol{e_k}\boldsymbol{f_m} + \boldsymbol{f_k}\boldsymbol{e_m})$$

Minimize $\sum_{i \in \mathcal{G}} F_i(\boldsymbol{P_i^g})$

with constraints:

$$\sum_{km \in \delta(k)} \boldsymbol{S_{km}} = (\sum_{i \in \mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d) + j(\sum_{i \in \mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d)$$

$\forall$ branch km : $\boldsymbol{S_{km}} = \boldsymbol{P_{km}} + \boldsymbol{jQ_{km}}$

$$\boldsymbol{P_{km}} = G_{kk}\, \boldsymbol{v_k^{(2)}} + G_{km}\, \boldsymbol{c_{km}} + B_{km}\, \boldsymbol{s_{km}}$$

$$\boldsymbol{Q_{km}} = -B_{kk}\, \boldsymbol{v_k^{(2)}} - B_{km}\, \boldsymbol{c_{km}} + G_{km}\, \boldsymbol{s_{km}}$$

$$\rightarrow \boldsymbol{v_k^{(2)}} = \boldsymbol{e_k^2} + \boldsymbol{f_k^2},\ \boldsymbol{c_{km}} = \boldsymbol{e_k e_m} + \boldsymbol{f_k f_m},\ \boldsymbol{s_{km}} = -\boldsymbol{e_k f_m} + \boldsymbol{f_k e_m}$$

$$\boldsymbol{P_{km}}^2 + \boldsymbol{Q_{km}}^2 \leq U_{km}$$

$V_k^{\min} \leq \boldsymbol{v_k^{(2)}} \leq V_k^{\max}$ $\forall$ bus k

$P_k^{\min} \leq \boldsymbol{P_k^g} \leq P_k^{\max}$ $\forall$ generator k

Minimize $\sum_{i \in \mathcal{G}} F_i(\boldsymbol{P_i^g})$

with constraints:

$$\sum_{km \in \delta(k)} \boldsymbol{S_{km}} = \left(\sum_{i \in \mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d\right) + j\left(\sum_{i \in \mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d\right)$$

$\forall$ branch km : $\boldsymbol{S_{km}} = \boldsymbol{P_{km}} + \boldsymbol{jQ_{km}}$

$$\boldsymbol{P_{km}} = G_{kk}\, \boldsymbol{v_k^{(2)}} + G_{km}\, \boldsymbol{c_{km}} + B_{km}\, \boldsymbol{s_{km}}$$

$$\boldsymbol{Q_{km}} = -B_{kk}\, \boldsymbol{v_k^{(2)}} - B_{km}\, \boldsymbol{c_{km}} + G_{km}\, \boldsymbol{s_{km}}$$

$$\rightarrow \boldsymbol{c_{km}^2} + \boldsymbol{s_{km}^2} \leq \boldsymbol{v_k^{(2)}} \boldsymbol{v_m^{(2)}}$$

$$\boldsymbol{P_{km}}^2 + \boldsymbol{Q_{km}}^2 \leq U_{km}$$

$V_k^{\min} \leq \boldsymbol{v_k^{(2)}} \leq V_k^{\max}$ $\forall$ bus k

$P_i^{\min} \leq \boldsymbol{P_i^g} \leq P_i^{\max}$ $\forall$ generator i

We can see an improvement

Case	Jabr relaxation value	Relaxation time (s)	Interior point value	Interior point time (s)
9	5296.67	0.00	5296.69	0.24
30	573.58	0.03	576.89	0.47
118	129297.41	0.32	129660.69	0.24
1354pegase	740092.83	2.02	74069.35	2.45
ACTIVSg2000	1226328.77	4.29	1228892.08	3.01
3120sp	2130950.72	53.01	2142703.77	5.24
9241pegase	309238.37	31.00	315912.43	161.29
9241pegase Jabr + non-convex	**84371.82**	**Root time: 400 s**		
ACTIVSg10k	1337732.48 premature termination	64.00 Ten cores	2485898.75	223.88 Two cores
13659pegase	361664.26 premature termination	59.00 Ten cores	386107.52	228.02 Two cores

(Gurobi 10 on SOCP,   Knitro from Matlab as interior point solver)

Why this behavior?

Lessons to learn (so far)

- Large SOCP relaxations are difficult for our solvers. Why?
- Many of the Jabr (SOC) constraints are tight at optimum – they are needed.
- Outer approximations, if accurate, require many linear inequalities.
- All are needed, or else relaxation inaccurate.
- If too many outer envelop inequalities are used, relaxation becomes hard.

- A **challenge** for spatial branch-and-bound:
  - Either start from weak relaxation, or use a very heavy formulation at every node
  - A challenge for heuristics

- What is needed? **An effective, compact** linear **relaxation.**

Minimize $\sum_{i \in \mathcal{G}} F_i(\boldsymbol{P_i^g})$

with constraints:

$$\sum_{km \in \delta(k)} \boldsymbol{S_{km}} = (\sum_{i \in \mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d) + j(\sum_{i \in \mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d)$$

$\forall$ branch km : $\boldsymbol{S_{km}} = \boldsymbol{P_{km}} + \boldsymbol{jQ_{km}}$

$$\boldsymbol{P_{km}} = G_{kk}\, \boldsymbol{v_k^{(2)}} + G_{km}\, \boldsymbol{c_{km}} + B_{km}\, \boldsymbol{s_{km}}$$

$$\boldsymbol{Q_{km}} = -B_{kk}\, \boldsymbol{v_k^{(2)}} - B_{km}\, \boldsymbol{c_{km}} - G_{km}\, \boldsymbol{s_{km}}$$

$$\rightarrow \boldsymbol{c_{km}^2} + \boldsymbol{s_{km}^2} \leq \boldsymbol{v_k^{(2)}} \boldsymbol{v_m^{(2)}}$$

$$\boldsymbol{P_{km}}^2 + \boldsymbol{Q_{km}}^2 \leq U_{km}$$

$V_k^{\min} \leq \boldsymbol{v_k^{(2)}} \leq V_k^{\max} \quad \forall$ bus k

$P_i^{\min} \leq \boldsymbol{P_i^g} \leq P_i^{\max} \quad \forall$ generator i

Another critical observation

$$S_{km} = (G_{kk} - jB_{kk})\,|V_k|^2 + (G_{km} - jB_{km})\,|V_k||V_m|(\cos\theta_{km} + j\sin\theta_{km})$$

Real ("active") part:

$$P_{km} = G_{kk}\,|V_k|^2 + G_{km}\,|V_k||V_m|\cos\theta_{km} + B_{km}\,|V_k||V_m|\sin\theta_{km}$$

$$P_{km} = G_{kk}\,v_k^{(2)} + G_{km}\,c_{km} + B_{km}\,s_{km}$$

Observation:

(+ G. Munoz, 2014)

$$P_{km} + P_{mk} = \text{power “loss”} \geq 0$$

Using this inequality, and **foregoing the (rotated cone) Jabr ineq**., already yields a **very tight relaxation (linear?)**

Minimize $\sum_{i \in \mathcal{G}} F_i(\ \boldsymbol{P_i^g})$

with constraints:

$$\sum_{km \in \delta(k)} \boldsymbol{S_{km}} \ = \ (\sum_{i \in \mathcal{G}(k)} \boldsymbol{P_i^g} - P_k^d) \ + \ j(\sum_{i \in \mathcal{G}(k)} \boldsymbol{Q_i^g} - Q_k^d)$$

$$\forall \text{ branch } km : \quad \boldsymbol{S_{km}} = \ \boldsymbol{P_{km}} \ + \ \boldsymbol{jQ_{km}}$$

$$\boldsymbol{P_{km}} = G_{kk} \ \boldsymbol{v_k^{(2)}} + G_{km} \ \boldsymbol{c_{km}} + B_{km} \ \boldsymbol{s_{km}}$$

$$\boldsymbol{Q_{km}} = -B_{kk} \ \boldsymbol{v_k^{(2)}} \ -B_{km} \ \boldsymbol{c_{km}} \ +G_{km} \ \boldsymbol{s_{km}}$$

$$\boldsymbol{P_{km}} + \ \boldsymbol{P_{mk}} \ \geq 0 \qquad \leftarrow$$

$$\boldsymbol{P_{km}}^2 + \ \boldsymbol{Q_{km}}^2 \ \leq \ U_{km}$$

$$V_k^{\min} \leq \boldsymbol{v_k^{(2)}} \leq V_k^{\max} \quad \forall \text{ bus } k$$

$$P_i^{\min} \leq \boldsymbol{P_i^g} \leq P_i^{\max} \quad \forall \text{ generator } i$$

GUROBI OPTIMIZATION

The almost linear formulation (gray = conic)

Case	relaxation	Relax time (s)	Interior Point	IPM time (s)
1354pegase	730269.92 740092.83	0.81 2.02	74069.35	2.45
ACTIVSg2000	1201333.04 1226328.77	1.93 4.29	1228892.08	3.01
3120sp	2061763.91 2130950.72	2.60 53.01	2142703.77	5.24
3375wp	7267498.66 7393015.73	2.00 5.00	7412072.19	5.66
6468rte	84117.84 unable to converge	2.29 48.00	unable to converge	long
9241pegase	304392.01 309238.37	7.32 31.00	315912.43	161.29
ACTIVSg10k	2364513.20 unable to converge	7.23 **146.94**	2485898.75	129.54
13569pegase	372046.88 unable to converge	6.98 76.00	386107.52	320.00
ACTIVSg25k	5802299.18 unable to converge	125 **332**	6017830.61	86.07
ACTIVSg70k	15305723.20 unable to converge	162.10 684.05	16439499.87	450.55

$$S_{km} = (G_{kk} - jB_{kk})\,|V_k|^2 + (G_{km} - jB_{km})\,|V_k||V_m|(\cos\theta_{km} + j\sin\theta_{km})$$

$$G_{kk} > 0 > G_{km} = G_{mk} \geq -G_{kk}, \quad B_{km} = B_{mk}$$

$$P_{km} = G_{kk}\,|V_k|^2 + G_{km}\,|V_k||V_m|\cos\theta_{km} + B_{km}\,|V_k||V_m|\sin\theta_{km}$$

$$P_{km} = G_{kk}\,v_k^{(2)} + G_{km}\,c_{km} + B_{km}\,s_{km}$$

$$P_{mk} = G_{mm}\,v_m^{(2)} + G_{mk}\,c_{km} - B_{km}\,s_{km}$$

$$P_{km} + P_{mk} = G_{kk}\,v_k^{(2)} + G_{mm}\,v_m^{(2)} + 2G_{km}\,c_{km} \geq \min\{G_{kk}, G_{mm}\}(\,v_k^{(2)} + v_m^{(2)} - 2c_{km})$$

$$c_{km}^2 + s_{km}^2 \leq v_k^{(2)} v_m^{(2)}$$

implies

$$v_k^{(2)} + v_m^{(2)} - 2c_{km} \geq 0$$

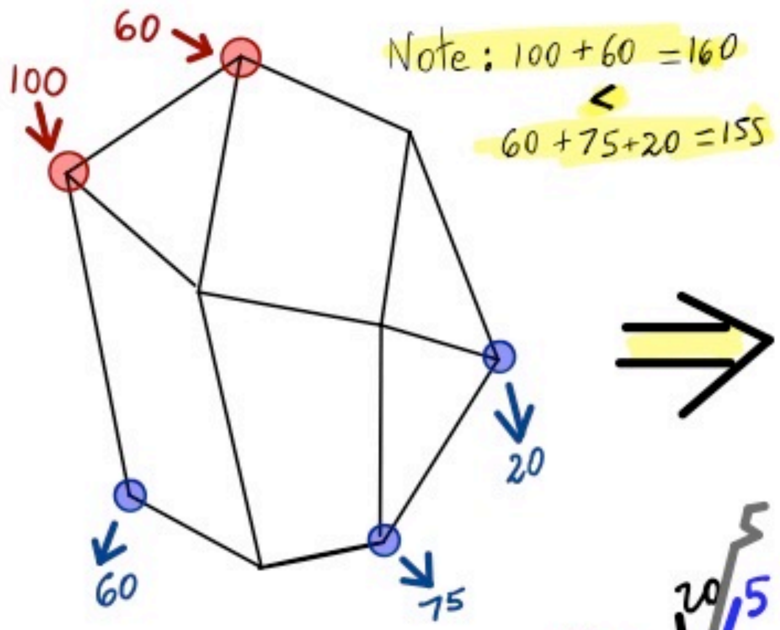
So:

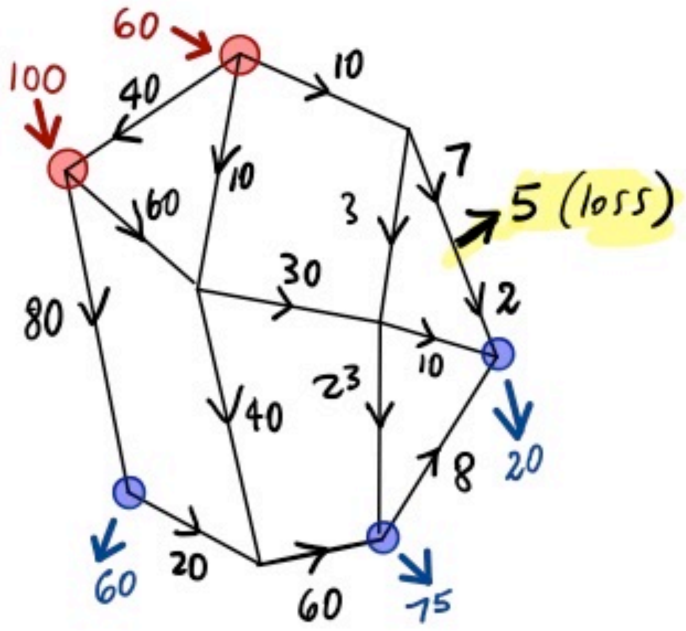
$$P_{km} + P_{mk} \geq 0$$

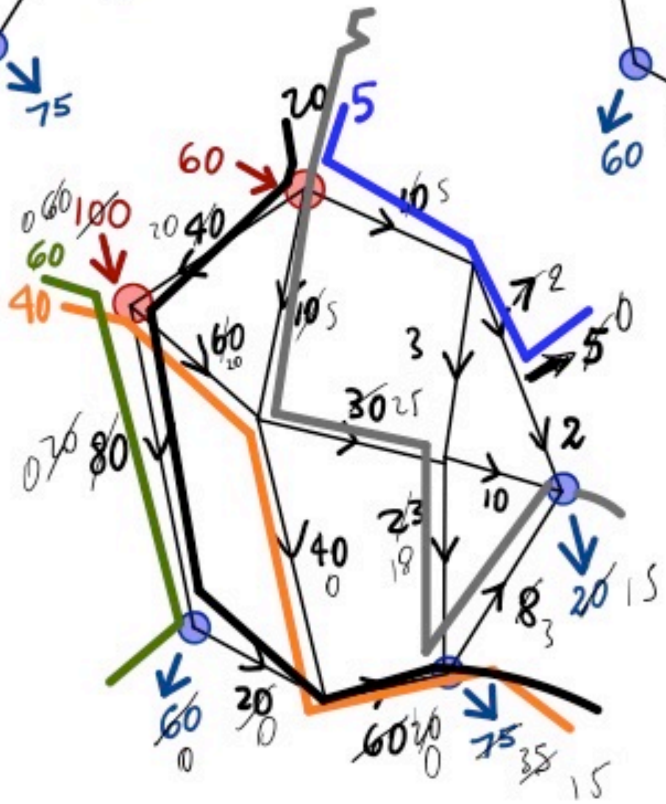
Lemma: this implies source-destination flow paths

Every unit of load is accounted for by a unit of generation

Flow decomposition







Each unit of load and loss accounted for by a corresponding unit of generation

$$P_{km} + P_{mk} = \sigma_{km}\, v^{(2)}_{km} + g_{km}(\, v^{(2)}_{k} + v^{(2)}_{m} - 2c_{km}) \qquad (\sigma_{km} \geq 0, \quad g_{km} > 0)$$

$$P_{km} + P_{mk} = r_{km}\, |I_{km}|^2$$ (physics)

Use $i^{(2)}_{km}$ to model $|I_{km}|^2$. So: $i^{(2)}_{km} = \dfrac{\sigma_{km}\, v^{(2)}_{km} + g_{km}(\, v^{(2)}_{k} + v^{(2)}_{m} - 2c_{km})}{r_{km}}$

$$P^2_{km} + Q^2_{km} = |V_k|^2 |I_{km}|^2$$ (more physics)

$$P^2_{km} + Q^2_{km} \leq v^{(2)}_{k} i^{(2)}_{km}$$

but don't: we do not want too many tight SOC constraints

(Castillo et al, Coffrin et al)

$$2P_{km} \leq v^{(2)}_{k} + i^{(2)}_{km}$$

(for example)

LP-based cutting plane algorithm

- **Matias Villagra** (PhD student)
- **Cut management**

$$\min \sum_{k \in \mathcal{G}} F_k(P_k^g)$$

subject to:

$$\sum_{km \in \delta(k)} S_{km} = (P_k^g - P_k^d) + j(Q_k^g - Q_k^d) \qquad \forall \text{ bus } k$$

$\forall$ branch km:

$$S_{km} = P_{km} + jQ_{km}$$

$$P_{km} = G_{kk} v_k^{(2)} + G_{km} c_{km} + B_{km} s_{km}$$

$$Q_{km} = -B_{kk} v_k^{(2)} + B_{km} c_{km} - G_{km} s_{km}$$

$$Q_{km} = -B_{kk} v_k^{(2)} + B_{km} c_{km} - G_{km} s_{km}$$

$$i_{km}^{(2)} = (G_{kk}^2 + B_{kk}^2)(v_k^{(2)} + v_m^{(2)} - 2c_{km})$$

<— key variable !

$$V_k^{\min} \leq v_k^{(2)} \leq V_k^{\max} \qquad \forall \text{ bus } k$$

$$P_k^{\min} \leq P_k^g \leq P_k^{\max} \qquad \forall k \in \mathcal{G}$$

Outline

- Start from **basic formulation (an LP?).**
- **Iterate**:
  - **Solve** linearly constrained problem.
  - **Separate** outer-envelope cuts for Jabr constraints (within tolerance).
  - **Separate** outer-envelope cuts for I2 constraints (within tolerance).
  - **Discard** old enough cuts, if slack enough.
  - **Reject** new cuts if too parallel to existing cuts.

LP-based cutting plane algorithm: details

→ **Matias Villagra** (PhD student)

$$c^t x \geq 0, \;\; d^t x \geq 0 \text{ parallel if } \cos(\text{angle}_{c,d}) > 1 - \frac{10^{-5}}{2}$$

$$\min \sum_{k \in \mathcal{G}} F_k(P_k^g)$$

subject to:

$$\sum_{km \in \delta(k)} S_{km} = (P_k^g - P_k^d) + j(Q_k^g - Q_k^d) \qquad \forall \text{ bus k}$$

$\forall$ branch km:

$$S_{km} = P_{km} + jQ_{km}$$

$$P_{km} = G_{kk} v_k^{(2)} + G_{km} c_{km} + B_{km} s_{km}$$

$$Q_{km} = -B_{kk} v_k^{(2)} + B_{km} c_{km} - G_{km} s_{km}$$

$$Q_{km} = -B_{kk} v_k^{(2)} + B_{km} c_{km} - G_{km} s_{km}$$

$$i_{km}^{(2)} = (G_{kk}^2 + B_{kk}^2)(v_k^{(2)} + v_m^{(2)} - 2c_{km})$$ <— key variable !

$$V_k^{\min} \leq v_k^{(2)} \leq V_k^{\max} \qquad \forall \text{ bus k}$$

$$P_k^{\min} \leq P_k^g \leq P_k^{\max} \qquad \forall k \in \mathcal{G}$$

- Start from **basic formulation.**

- **Iterate**:
  - Solve linearly constrained problem.
  - Compute the top 55% most violated **Jabr-envelope** cuts in current solution. **Add cuts** if non-parallel.
  - Compute the top 15% most violated **i2-envelope** cuts in current solution. **Add cuts** if non-parallel.
  - **Discard** old (5 or more iterations) cuts which are slack (violation > 1e-5).

Performance of cutting-plane Algorithm (gray = Gurobi on conic)

Case	Cutting-plane	Time (s)	Log barrier	Log barrier time (s)
1354pegase	74000.3 74006.95	3.68 1.20	74069.35	2.45
ACTIVSg2000	1225960 1226328.77	6.52 4.29	1228892.08	3.01
2869pegase	133863 133875.022	16.51 2.61	133999.29	2.99
3120sp	2131250 unable to converge	5.5 2.82	2142703.77	5.24
3375wp	7393140 unable to converge	25.13 5.21	7412072.19	5.66
9241pegase	309297 309040.38	67.55 13.20	315912.43	161.29
ACTIVSg10k	2471240 unable to converge	16.31 20.20	2485898.75	129.54
13569pegase	379167 unable to converge	79.88 34.46	386107.52	320.00
ACTIVSg25k	5979740 unable to converge	51.32 61.11	6017830.61	86.07
ACTIVSg70k	16300200 unable to converge	205.9 282.29	16439499.87*	450.55

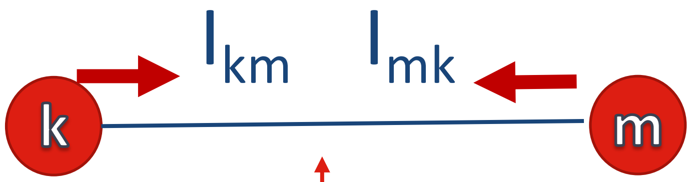
Gurobi parameters: method = Barrier, BarHomogeneous = 1, NumericFocus = 1, BarConvTol = 1e-6, FeasibilityTol = 1e-6, OptimalityTol = 1e-6
Knitro used as interior point solver.

Finding feasible solutions: the GO2 competition

- Whereas developing strong relaxations is important from a fundamental standpoint …
  the industry is focused on computing good solutions, fast **enough**.
- The **GO (Grid Optimization) Competition** has been run by the U.S. Department of Energy for several years.
- It is currently in its third iteration, or "Challenge".
- Here we will talk on our experience in Challenge 2 (with Richard Waltz, Knitro/Artelys). We placed #2. The winner was **Hassan Hijazi** (LANL).
- **All competitions have focused on finding good solutions within time limits.**
- **Many important industry features** that go beyond standard ACOPF.
- One **particular feature** is especially relevant …

GUROBI OPTIMIZATION

Feature: configurable admittance matrices



$$Y = \begin{pmatrix} G_{kk} + j B_{kk} & G_{km} + j B_{km} \\ G_{mk} + j B_{mk} & G_{mm} + j B_{mm} \end{pmatrix}$$

Admittance matrix for line km

$$I_{km} = Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \rightarrow S_{km} = V_k I_{km}^*$$

Ohm's Law

Complex power

Complex current

$$S_{km} = V_k \left[Y \begin{pmatrix} V_k \\ V_m \end{pmatrix} \right]^*$$

GUROBI
OPTIMIZATION

Feature: configurable admittance matrices

$$Y = \begin{pmatrix} G_{kk} + j\,B_{kk} & G_{km} + j\,B_{km} \\ G_{mk} + j\,B_{mk} & G_{mm} + j\,B_{mm} \end{pmatrix}$$

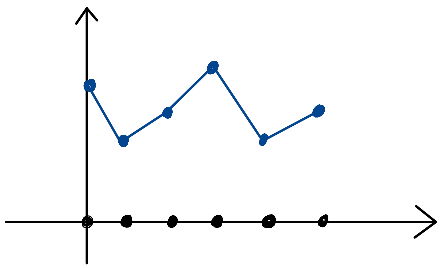
Admittance matrices include information about physical attributes such as:

- Pure impedance (resistance and reactance)
- Transformer properties (voltage magnitude and phase shifting)
- Shunts (line-charging)
- All are configurable!

$$\boldsymbol{S_{km}} = (G_{kk} - j B_{kk})\ |\boldsymbol{V_k}|^2 + (G_{km} - j B_{km})\ |\boldsymbol{V_k}||\boldsymbol{V_m}|(\cos\boldsymbol{\theta_{km}} + \boldsymbol{j}\sin\boldsymbol{\theta_{km}})$$

Example: impedance correction for transformers

Inversely proportional to:



GUROBI OPTIMIZATION

Feature: contingencies

Optimization problem (in GO2) seeks a **minimum-cost solution**. However,

- This solution corresponds to operation under a **"base case"**
- But this solution must **survive a number of contingencies**
- Each contingency corresponds to **losing a branch or a generator**
- Definition of survivable: we must be able to reset variables, within bounds, so as to become feasible or nearly feasible
- **We pay a penalty for each infeasibility**
- **Total score: value of base-case solution + average continency penalty**
- **Thousands** of contingencies
- Industry standard: "N-1" criterion

Related feature: penalized slacks

The nominal optimization problem includes **nonlinear equations**.

- Typical example: load balance equations.
- Nonlinear solvers are unlikely to satisfy these **exactly**.
- What would that mean in terms of the application? **Insufficient or excessive** energy.
- In fact, it is unlikely that loads will be known **exactly** at the time of the computation.
- And even if known, they will **change** shortly after.
- **How does the grid even work???**
  - **Reserves and real-time balancing take care of mismatches.**
  - **Power mismatches sensed** through AC grid physics.
- GO competition: **allow** small errors, but **penalize** them in the objective.

GUROBI OPTIMIZATION

Most important feature: the prior solution

ACOPF problems, in the real-world are ~~never~~ rarely run “from scratch”.

- Rather, they are run with a **prior solution vector available**.
- This vector corresponds to a solution previously (recently) computed for a prior (recent) data realization.
- Examples: some loads have changed (a bit). A few generators may be turned off, and a few, turned on. Some branches may be turned off/on.
- **The total change will not be large.**
- New solution amounts to a **migration from the prior solution.** The migration is constrained.
- Can we take this information into account, in computing a new solution?
- **YES!**

How do we find solutions?

Today, there is no question as to what kind of algorithm to use in order to compute solutions: interior-point methods. Knitro, IPOPT, a few others.

- Several closely related methodologies based on nonlinear programming.
- Incredibly effective for ACOPF. Very flexible. There is no competition.
- But …
- **Interior point methods are "local solvers".**
- There is no guarantee of … anything, basically.
- On very large models arising from the GO competition, the algorithms need a lot of help from heuristics.
- In the next few slides, we will discuss what we (R. Waltz and I) did, using Knitro.

GUROBI OPTIMIZATION

Basic approach:

1. **Dimensionality reduction**
   - Fact: real-world power systems are **very robust**
   - Transition from previous solution, to base solution **should require minor adjustments**
   - Transition from base solution, to contingency solutions **should require minor adjustments**
2. **Progressive rounding**
   Common-sense iterative rounding of integer variables – **all integer variables modeled as sum of binaries**
3. **Numerical stability**
   Some numerical parameters in GO setup are too large.
   Slack penalty function in particular. Replaced with simple linear penalty plus bounds.

GUROBI OPTIMIZATION

Dimensionality reduction

Example: transition from previous solution, to base solution

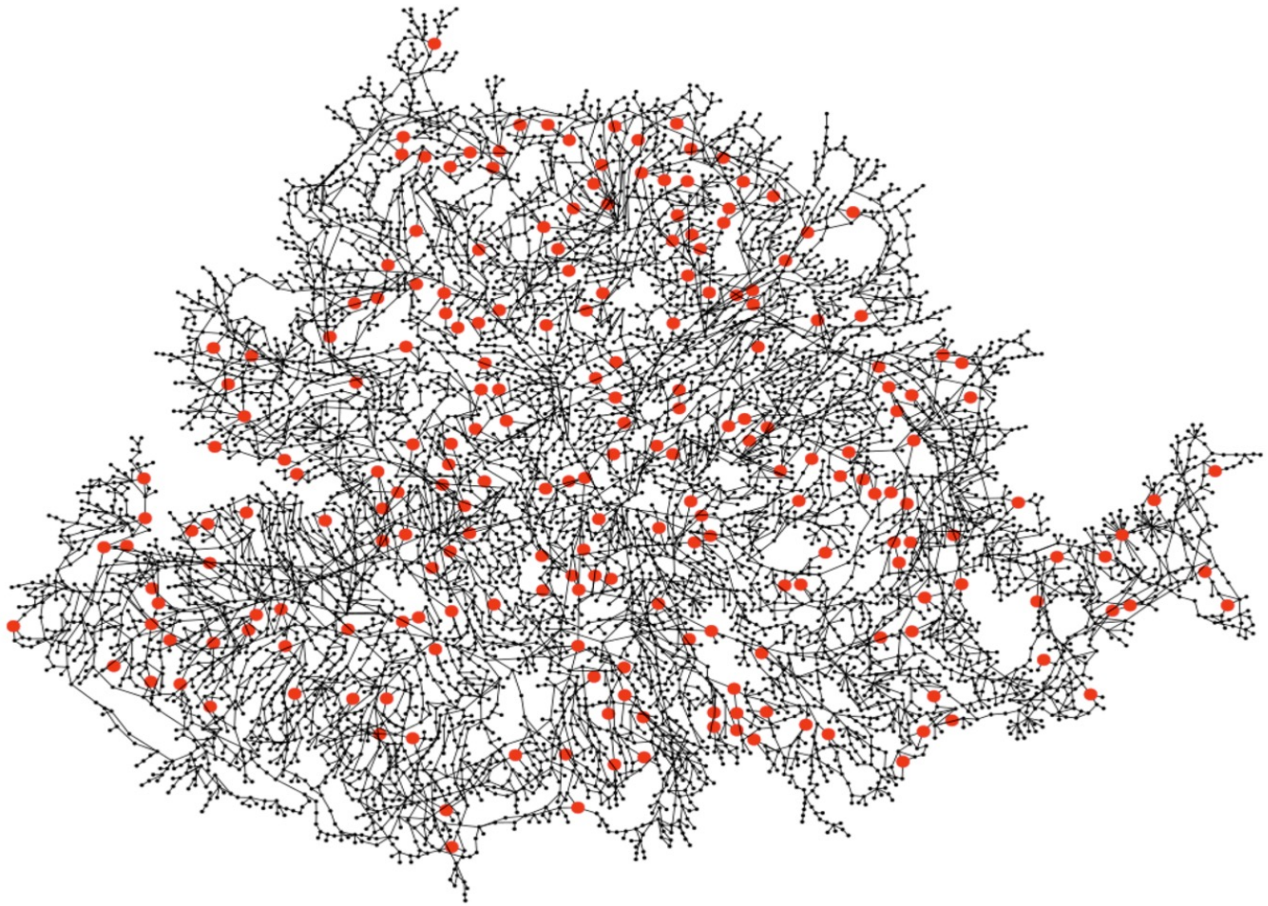
- Most changes in **data** from previous network, to new one, involve changes in **demands**
- Most demand changes are **very small**
- **What we did:**

1. Select a subset $\boldsymbol{S}$ of buses with **large enough** demand changes
2. **Restrict** changes in prior solution to buses that are **electrically close enough** to $\boldsymbol{S}$
3. Apply our rounding heuristic to the correspondingly restricted ACOPF problem

N06100 scenario 115

6476 buses, 3371 loads, 406 generators, 5337 lines, 3086 transformers, 2467 contingencies

- About 100,000 variables and constraints
- Picture shows buses where prior solution has infeasibility greater than 1e-3: about 200



obj=712281.64

total_bus_cost 1.85847217e-02
total_load_benefit 1.26257799e+06
total_gen_cost 5.50296330e+05
total_line_cost 0.00000000e+00
total_xfmr_cost 0.00000000e+00

(Knitro feaserror 2.481e-09)

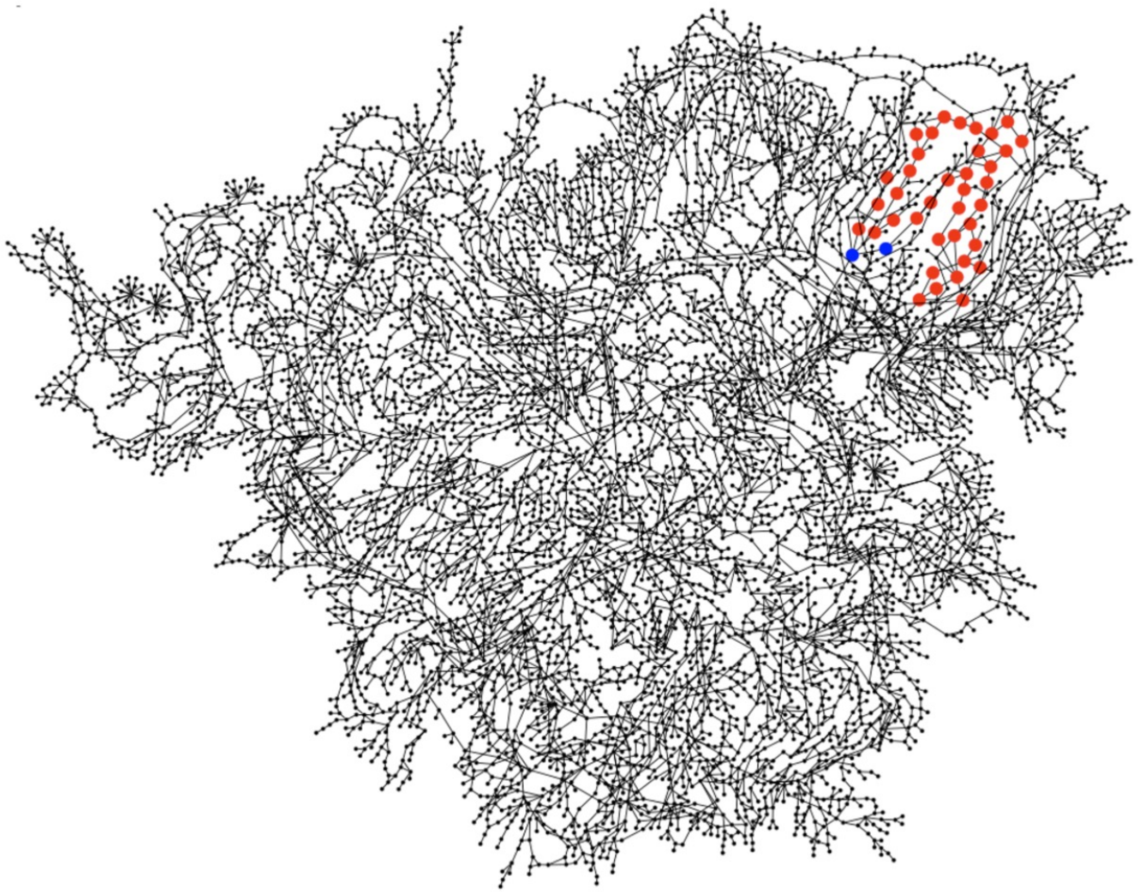
169.05 sec 450 iterations

GUROBI OPTIMIZATION

N06100 scenario 115

6476 buses, 3371 loads, 406 generators, 5337 lines, 3086 transformers, 2467 contingencies

Contingency 3 (line down)



- **Base case** obj: **712281.64**
- **Heuristic** solution:

total_bus_cost 1.65362151e-02
total_load_benefit 1.25961077e+06
total_gen_cost 5.44767392e+05
total_line_cost 0.00000000e+00
total_xfmr_cost 0.00000000e+00

objective 714843.36

31.30 seconds 227 iterations
+ 6.8 seconds + 27 iterations (rounding)

GUROBI OPTIMIZATION

Moving forward …

1. **Heuristics** for finding feasible solutions for ACOPF that do **not** rely on interior point methods. Existing generic heuristics do not work (well, or at all). **LP-based** heuristics would be ideal.

2. ACOPF in **energy markets**. Current power grid operations are supported by the day-ahead, or **SCUC** (Security-Constrained Unit Commitment) computation. This a large-scale, mixed-integer, multiple scenario problem. It is linear – using the (linear) DC approximation to ACOPF. Moving forward, it is expected that the AC power flow model will become more prevalent. GO3 competition points in that direction.

GUROBI OPTIMIZATION

ACOPF Gurobi Optimod

Published last week: https://gurobi-optimization-gurobi-optimods.readthedocs-hosted.com/en/stable/mods/opf/opf.html

GUROBI OPTIMIZATION

GUROBI: ALWAYS FREE FOR ACADEMICS & RECENT GRADUATES

OPTIMIZATION SUCCESS STARTS HERE

Special opportunities for the academic community:

- Gurobi: Always free for academics
- Gurobi user community page
- Educational Resources:
    - www.gurobi.com/academia

Email academicprogram@gurobi.com to learn more.

TAKE GUROBI WITH YOU

Gurobi is also available to recent graduates through our Take Gurobi With You Program

Introducing:

www.BurritoOptimizationGame.com

BURRITO© OPTIMIZATION GAME

GUROBI OPTIMIZATION

QUESTIONS?

Thank You

For more information: gurobi.com

Dr. Daniel Bienstock
Columbia University

daniel.bienstock@gurobi.com