Power Grids: Controlling Cascades and Assessing Vulnerability

Dan Bienstock

Columbia

DOE PI, October 2011

Outline

- **Part I**. The N-K problem in AC power flows
- **Part II**. Cascades

Joint with Sean Harnett (Columbia), Taedong Kim and Steve Wright (Wisconsin).

Supported by DOE DE-SC0002283.

AC Model for Power Grids

Power flows and voltages in a power grid are dictated by elementary physical laws (Kirchoff), along with

- physical attributes (e.g. impedances) of the lines connecting nodes;
- (complex) power supplied at generator nodes
- (complex) power consumed at demand nodes

Leads to a system of quadratic equations in complex variables, namely

- complex voltages $V_i = |V_i|e^{j\theta_i}$ at the nodes;
- complex powers $P_i + jQ_i$ at the nodes.

Half of the variables are given:

- P_i, Q_i at the demand nodes;
- P_i, $|V_i|$ at the generator nodes.

Eliminate P_i and Q_i to obtain a nonlinear system $F(V) = 0$

Background: AC power transmission equaitons

→ Voltage at bus (node) $\mathbf{k} = \mathbf{V_k} = |\mathbf{V_k}|\, \mathbf{e}^{\mathbf{j}\theta_\mathbf{k}}$, where $\mathbf{j} = \sqrt{-1}$

Background: AC power transmission equaitons

→ Voltage at bus (node) $\mathbf{k} = \mathbf{V_k} = |\mathbf{V_k}|\, \mathbf{e}^{\mathbf{j}\theta_\mathbf{k}}$, where $\mathbf{j} = \sqrt{-1}$

→ Current injected by $\mathbf{k}$ into line between $\mathbf{k}$ and $\mathbf{m} = \mathbf{I_{km}} =$

$$\frac{V_k - V_m}{Z_{km}}$$

where $\mathbf{Z_{km}}$ = line impedance = $\mathbf{R_{km}} + \mathbf{j}\,\mathbf{X_{km}}$

($\mathbf{R_{km}}$ = resistance, $\mathbf{X_{km}}$ = inductance)

Background: AC power transmission equaitons

→ Voltage at bus (node) $\mathbf{k} = \mathbf{V_k} = |\mathbf{V_k}|\, \mathbf{e}^{\mathbf{j}\theta_\mathbf{k}}$, where $\mathbf{j} = \sqrt{-1}$

→ Current injected by $\mathbf{k}$ into line between $\mathbf{k}$ and $\mathbf{m} = \mathbf{I_{km}} =$

$$\frac{V_k - V_m}{Z_{km}}$$

where $\mathbf{Z_{km}}$ = line impedance = $\mathbf{R_{km}} + \mathbf{j}\,\mathbf{X_{km}}$

($\mathbf{R_{km}}$ = resistance, $\mathbf{X_{km}}$ = inductance)

or, current injected by $\mathbf{k} = (V_k - V_m) Y_{km}$, where

$\mathbf{Y_{km}}$ = line admittance = $\mathbf{Z_{km}^{-1}}$

(Π-model, simple version)

General case

$$\begin{bmatrix} \mathbf{I_{km}} \\ \mathbf{I_{mk}} \end{bmatrix} = \mathcal{Y}_{\mathbf{km}} \begin{bmatrix} \mathbf{V_k} \\ \mathbf{V_m} \end{bmatrix}$$

where $\mathcal{Y}_{\mathbf{km}}$ = admittance matrix for line **km**

General case

$$\begin{bmatrix} \mathbf{I_{km}} \\ \mathbf{I_{mk}} \end{bmatrix} = \mathcal{Y}_{\mathbf{km}} \begin{bmatrix} \mathbf{V_k} \\ \mathbf{V_m} \end{bmatrix}$$

where $\mathcal{Y}_{\mathbf{km}}$ = admittance matrix for line **km**

Finally, complex **power** injected by **k** into line **km** = $\mathbf{V_k I^*_{km}}$

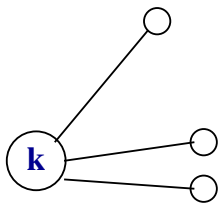
General case

$$\begin{bmatrix} \mathbf{I_{km}} \\ \mathbf{I_{mk}} \end{bmatrix} = \mathcal{Y}_{\mathbf{km}} \begin{bmatrix} \mathbf{V_k} \\ \mathbf{V_m} \end{bmatrix}$$

where $\mathcal{Y}_{\mathbf{km}}$ = admittance matrix for line **km**

Finally, complex **power** injected by **k** into line **km** = $\mathbf{V_k I^*_{km}}$

and, **total** power injected by **k** into the network = $\sum_{\mathbf{km}} \mathbf{V_k I^*_{km}}$



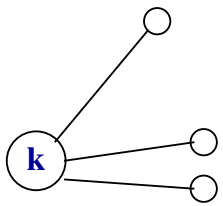
General case

$$\begin{bmatrix} \mathbf{I_{km}} \\ \mathbf{I_{mk}} \end{bmatrix} = \mathcal{Y}_{\mathbf{km}} \begin{bmatrix} \mathbf{V_k} \\ \mathbf{V_m} \end{bmatrix}$$

where $\mathcal{Y}_{\mathbf{km}}$ = admittance matrix for line **km**

Finally, complex **power** injected by **k** into line **km** = $\mathbf{V_k I^*_{km}}$

and, **total** power injected by **k** into the network = $\sum_{\mathbf{km}} \mathbf{V_k I^*_{km}}$



$= \mathbf{U^T}\, \mathbf{Y_k}\, \mathbf{U}$, where $\mathbf{U^T} = (\mathbf{Re(V_1)}, \ldots, \mathbf{Re(V_n)}, \mathbf{Im(V_1)}, \ldots, \mathbf{Im(V_n)})$

Summary:

where $\mathbf{U}^{\mathbf{T}} = (\mathbf{Re}(\mathbf{V_1}), \ldots, \mathbf{Re}(\mathbf{V_n}), \mathbf{Im}(\mathbf{V_1}), \ldots, \mathbf{Im}(\mathbf{V_n}))$,

power injected by **k** into network $= \mathbf{U}^{\mathbf{T}}\, \mathbf{Y_k}\, \mathbf{U}$

Summary:

where $\mathbf{U^T = (Re(V_1), \ldots, Re(V_n), Im(V_1), \ldots, Im(V_n))}$,

power injected by **k** into network $= \mathbf{U^T\, Y_k\, U} \;=\; \mathbf{P_k + j\, Q_j}$

($\mathbf{P_k}$ = active power, $\mathbf{Q_k}$ = reactive power)

Summary:

where $\mathbf{U}^{\mathbf{T}} = (\mathbf{Re}(\mathbf{V}_1), \ldots, \mathbf{Re}(\mathbf{V}_n), \mathbf{Im}(\mathbf{V}_1), \ldots, \mathbf{Im}(\mathbf{V}_n))$,

power injected by $\mathbf{k}$ into network $= \mathbf{U}^{\mathbf{T}}\, \mathbf{Y}_{\mathbf{k}}\, \mathbf{U} \;=\; \mathbf{P}_{\mathbf{k}} + \mathbf{j}\, \mathbf{Q}_{\mathbf{j}}$

($\mathbf{P}_{\mathbf{k}}$ = active power, $\mathbf{Q}_{\mathbf{k}}$ = reactive power)

Power flow problem

Solve $\mathbf{P}_{\mathbf{k}} + \mathbf{j}\, \mathbf{Q}_{\mathbf{k}} \;=\; \mathbf{U}^{\mathbf{T}}\, \mathbf{Y}_{\mathbf{k}}\, \mathbf{U}$, for all $\mathbf{k}$,

- $\mathbf{P}_{\mathbf{k}}$ and $|\mathbf{V}_{\mathbf{k}}|$ given for each generator $\mathbf{k}$
- $\mathbf{P}_{\mathbf{k}}$ and $\mathbf{Q}_{\mathbf{k}}$ given for each non-generator $\mathbf{k}$

Summary:

where $\mathbf{U}^{\mathbf{T}} = (\mathbf{Re}(\mathbf{V_1}), \ldots, \mathbf{Re}(\mathbf{V_n}), \mathbf{Im}(\mathbf{V_1}), \ldots, \mathbf{Im}(\mathbf{V_n}))$,

power injected by $\mathbf{k}$ into network $= \mathbf{U}^{\mathbf{T}}\, \mathbf{Y_k}\, \mathbf{U} \;=\; \mathbf{P_k} + \mathbf{j}\,\mathbf{Q_j}$

($\mathbf{P_k}$ = active power, $\mathbf{Q_k}$ = reactive power)

Power flow problem

Solve $\mathbf{P_k} + \mathbf{j}\,\mathbf{Q_k} \;=\; \mathbf{U}^{\mathbf{T}}\, \mathbf{Y_k}\, \mathbf{U}$, for all $\mathbf{k}$,

- $\mathbf{P_k}$ and $|\mathbf{V_k}|$ given for each generator $\mathbf{k}$
- $\mathbf{P_k}$ and $\mathbf{Q_k}$ given for each non-generator $\mathbf{k}$

Ideally, should meet additional conditions:

Summary:

where $\mathbf{U}^T = (\mathrm{Re}(V_1), \ldots, \mathrm{Re}(V_n), \mathrm{Im}(V_1), \ldots, \mathrm{Im}(V_n))$,

power injected by $\mathbf{k}$ into network $= \mathbf{U}^T \mathbf{Y}_k \mathbf{U} \;=\; P_k + j\,Q_j$

(P_k = active power, Q_k = reactive power)

Power flow problem

Solve $P_k + j\,Q_k \;=\; \mathbf{U}^T \mathbf{Y}_k \mathbf{U}$, for all $\mathbf{k}$,

- P_k and $|V_k|$ given for each generator $\mathbf{k}$
- P_k and Q_k given for each non-generator $\mathbf{k}$

Ideally, should meet additional conditions:

$|V_k|$ approximately constant, and high; line phase angle differences small

Summary:

We need to solve a system of quadratic equations

- An algorithmicist says: I want a method that is guaranteed to work

Summary:

We need to solve a system of quadratic equations

- An algorithmicist says: I want a method that is guaranteed to work
- The engineers say: to solve today's problem, we will use yesterday's solution and apply Newton-Raphson

$\longrightarrow$ The iterative approach works extremely well

Summary:

We need to solve a system of quadratic equations

- An algorithmicist says: I want a method that is guaranteed to work
- The engineers say: to solve today's problem, we will use yesterday's solution and apply Newton-Raphson

→ The iterative approach works extremely well

→ but only under normal conditions.
→ The rest of the time it's a disaster

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)
- The system may have **many** solutions, and Newton may converge to one that is not **practicable**.

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)
- The system may have **many** solutions, and Newton may converge to one that is not **practicable**.
- Distant starting point may case Newton's method to "get lost"

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)
- The system may have **many** solutions, and Newton may converge to one that is not **practicable**.
- Distant starting point may case Newton's method to "get lost"

Enhancing Newton's method can help a little (e.g. line search, damping).

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)
- The system may have **many** solutions, and Newton may converge to one that is not **practicable**.
- Distant starting point may case Newton's method to "get lost"

Enhancing Newton's method can help a little (e.g. line search, damping).

Fancier algorithms (e.g. continuation / homotopy) are too expensive.

Solving the AC Equations

Newton's method for nonlinear equations $F(V) = 0$ (or an expanded formulation) is the "workhorse." Codes available e.g. MATPOWER, Powerworld.

Possible issues with Newton's method:

- The system may be **infeasible**! (Physically, the network is not viable.)
- The system may have **many** solutions, and Newton may converge to one that is not **practicable**.
- Distant starting point may case Newton's method to "get lost"

Enhancing Newton's method can help a little (e.g. line search, damping).

Fancier algorithms (e.g. continuation / homotopy) are too expensive.

Lavaei-Low (2010): Semi-definite programming formulation can yield reliable solution (under some assumptions).

Measuring Viability of a Grid

Not all solutions of the AC equations are equally practicable.

- There are often limits on the capacity of transmission lines. Can enforce as constraints on the flows.
- Desirable solutions have voltage magnitudes $|V_i|$ close to 1 for all i (in the range $[.93, 1.07]$, say). after scaling

Measuring Viability of a Grid

Not all solutions of the AC equations are equally practicable.

- There are often limits on the capacity of transmission lines. Can enforce as constraints on the flows.
- Desirable solutions have voltage magnitudes $|V_i|$ close to 1 for all i (in the range $[.93, 1.07]$, say). after scaling

If system not viable, find the smallest total change to the loads that restores feasibility — subject to line limits and voltage magnitudes staying near 1.

Observation: solutions are "sparse"

ℓ_1 Formulation

One possible formulation of the "feasibility restoration" problem:

$$
\begin{aligned}
\min_{V,\sigma,\rho} \; & \sum_{i \in PQ} |P_i^D| \sigma_i \\
\text{s.t. } & F_i^P(V) + |P_i^G| \rho_i = 0, \quad \forall i \in PV, \\
& F_i^P(V) + |P_i^D| \sigma_i = 0, \quad \forall i \in PQ, \\
& F_i^Q(V) + |Q_i^D| \sigma_i = 0, \quad \forall i \in PQ, \\
& V_{\min} \leq |V_i| \leq V_{\max}, \quad \forall i \in PV \cup PQ, \\
& 0 \leq \sigma_i \leq 1, \quad \forall i \in PQ. \\
& \rho_i \text{ free}, \quad \forall i \in PV,
\end{aligned}
$$

where σ_i denote (relative) changes to the loads, ρ_i denotes changes to generation capacity, etc.

Vulnerability Analysis – "N-K" problem

Identify K $(= 2, 3, ...)$ lines in the network whose removal has the most dramatic effect on viability.

Vulnerability Analysis – "N-K" problem

Identify K $(= 2, 3, ...)$ lines in the network whose removal has the most dramatic effect on viability.

Line removal $\Rightarrow$ binary variables (one for each line), very difficult.

Vulnerability Analysis – "N-K" problem

Identify K $(= 2, 3, ...)$ lines in the network whose removal has the most dramatic effect on viability.

Line removal $\Rightarrow$ binary variables (one for each line), very difficult.

$\rightarrow$ Use continuous variables $\gamma_i \geq 0$ to denote relative amount by which impedance is increased on each line. (Bienstock and Verma, SIOPT 2010)

Vulnerability Analysis – "N-K" problem

Identify K $(= 2, 3, ...)$ lines in the network whose removal has the most dramatic effect on viability.

Line removal $\Rightarrow$ binary variables (one for each line), very difficult.

$\rightarrow$ Use continuous variables $\gamma_i \geq 0$ to denote relative amount by which impedance is increased on each line. (Bienstock and Verma, SIOPT 2010)

Problem: Maximize the degradation of the network $\mathcal{G}(\gamma)$ when impedances are changed according to $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_n)$.

$$\max \mathcal{G}(\gamma) \ \text{ s.t. } \ e^T \gamma \leq B,$$

where $e = (1, 1, \ldots, 1)^T$ and B denotes the "impedance increase" budget.

"Maximize the damage for a given budget of impedance increases."

Large γ_i = critical lines.

Defining $\mathcal{G}$

Possible definitions of degradation measure $\mathcal{G}(\gamma)$:

- Objective of the ℓ_1 formulation: load that needs to be shed to regain feasibility

Defining $\mathcal{G}$

Possible definitions of degradation measure $\mathcal{G}(\gamma)$:

- Objective of the ℓ_1 formulation: load that needs to be shed to regain feasibility
- Voltage collapse: $\sum_{=1}^{n}(1-|V_i|^2)^2$. (Valid only when the network is stil feasible despite the increase in impedances.)

Defining $\mathcal{G}$

Possible definitions of degradation measure $\mathcal{G}(\gamma)$:

- Objective of the ℓ_1 formulation: load that needs to be shed to regain feasibility
- Voltage collapse: $\sum_{=1}^{n}(1-|V_i|^2)^2$. (Valid only when the network is stil feasible despite the increase in impedances.)

In either case, get a two-level optimization problem:

- **Upper level:** variables γ, objective $\mathcal{G}$;
- **Lower level:** solving AC power flow equations (or ℓ_1 extension) for given γ.

Defining $\mathcal{G}$

Possible definitions of degradation measure $\mathcal{G}(\gamma)$:

- Objective of the ℓ_1 formulation: load that needs to be shed to regain feasibility
- Voltage collapse: $\sum_{=1}^{n}(1-|V_i|^2)^2$. (Valid only when the network is stil feasible despite the increase in impedances.)

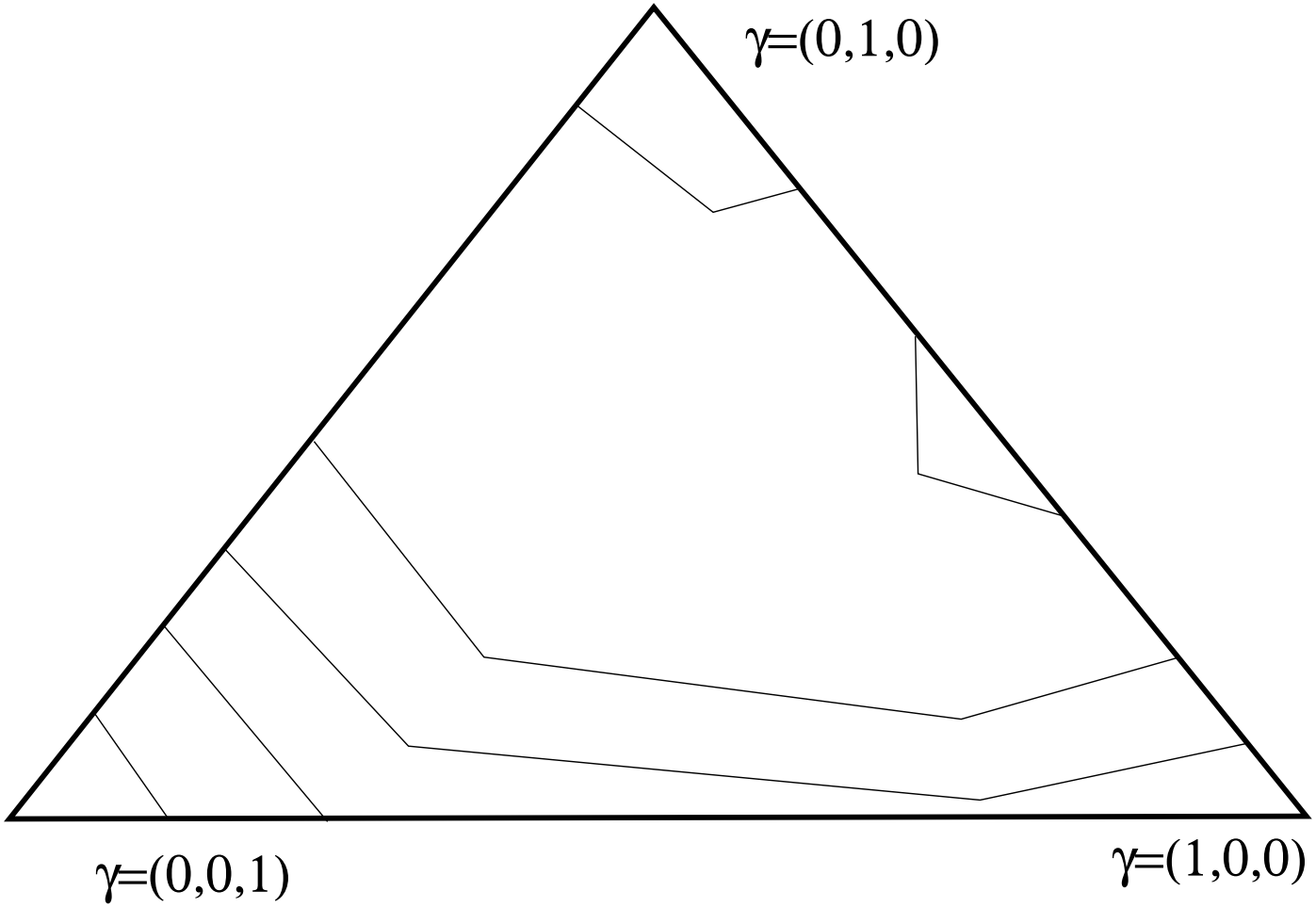
In either case, get a two-level optimization problem:

- **Upper level:** variables γ, objective $\mathcal{G}$;
- **Lower level:** solving AC power flow equations (or ℓ_1 extension) for given γ.

Objective $\mathcal{G}$ usually non-concave, possibly nonsmooth.

- Frank-Wolfe method with estimated gradient (smooth objectives);
- Randomized search algorithm for nondifferentiable optimization (described below).

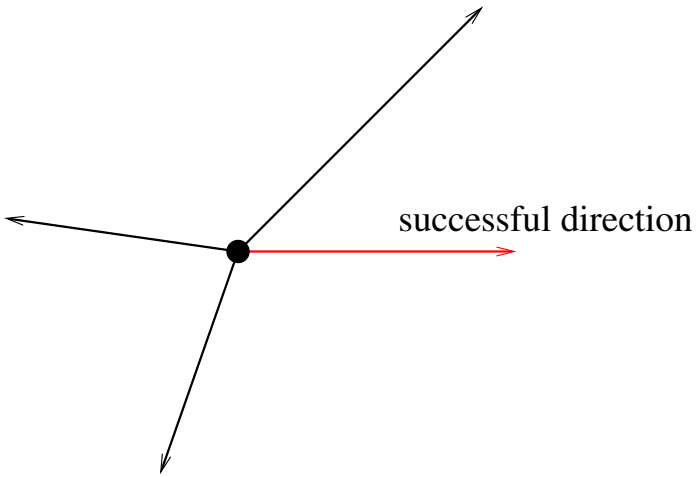
Possible Contours of $\mathcal{G}(\gamma)$

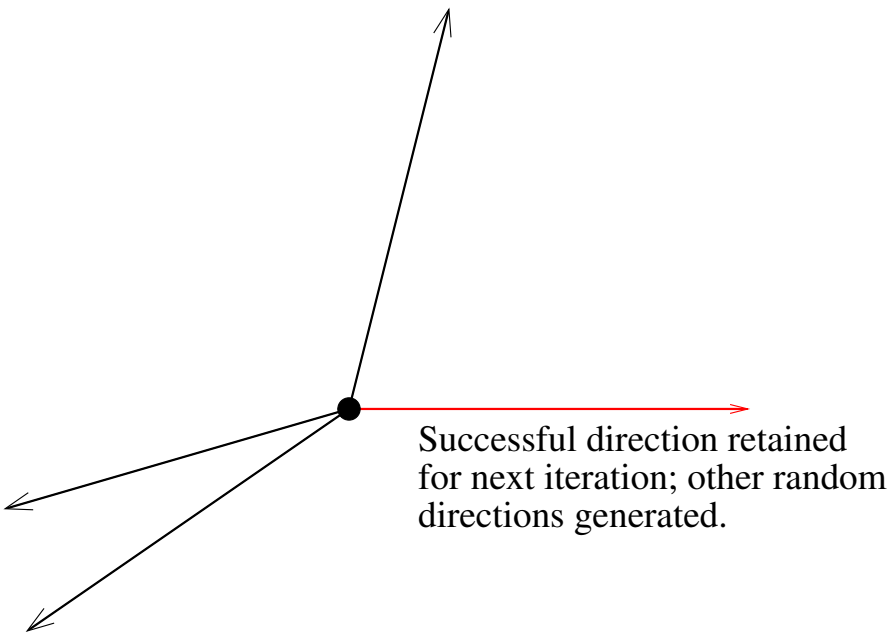


Possible countour map for G in case n=3.
Multiple local maxima; global max is at (0,0,1)

Randomized Search

- Take latest successful search direction and calculate some random new directions (10, say);
- Evaluate function at step length α along each direction;
- Take step with the best decrease.
- If all steps increase f, shrink α and try again with a new set of random directions.





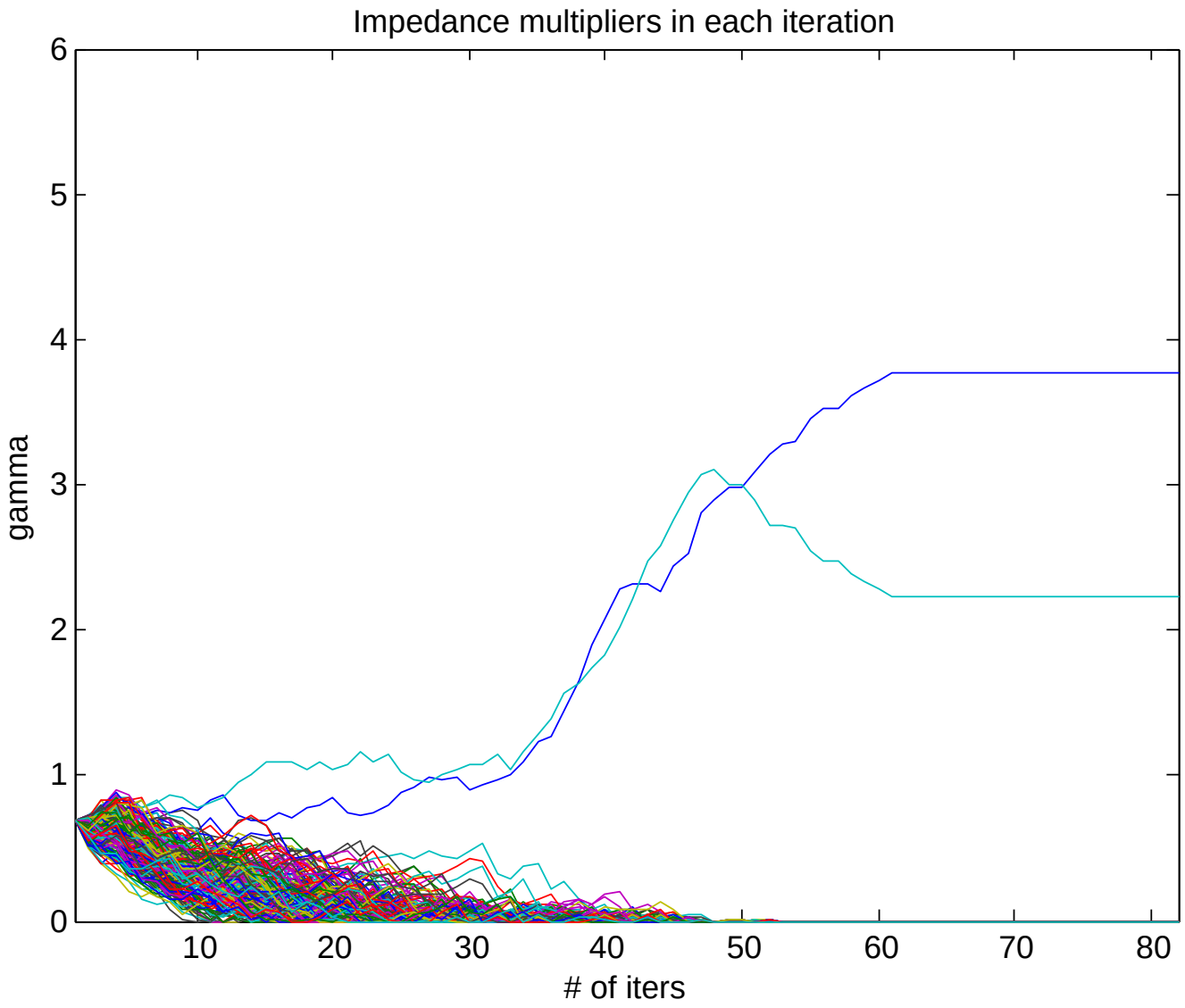
Run 10 times on the IEEE 118 Bus Case

10 random search directions at each step; budget $B = 6$; uses a continuation heuristic (start with large B and decrease to 6).

Run	Number of Iterations	Objective $\mathcal{G}(\gamma)$	Lines with non-zero γ_i			
			71	72	74	82
1	77	1.3309e-01	3.7685	0	2.2315	0
2	109	1.3464e-01	3.1652	0	2.7205	0.1143
3	152	1.1230e-01	2.0779	0.6844	3.2257	0.0120
4	147	1.2033e-01	2.4403	0	3.5597	0
5	71	1.3309e-01	3.7637	0	2.2361	0
6	120	1.2197e-01	2.4565	0.2271	3.3163	0
7	88	1.3309e-01	3.7685	0	2.2315	0
8	109	1.2133e-01	2.3508	0.2450	3.4043	0
9	96	1.3309e-01	3.7683	0	2.2317	0
10	82	1.3309e-01	3.7683	0	2.2317	0

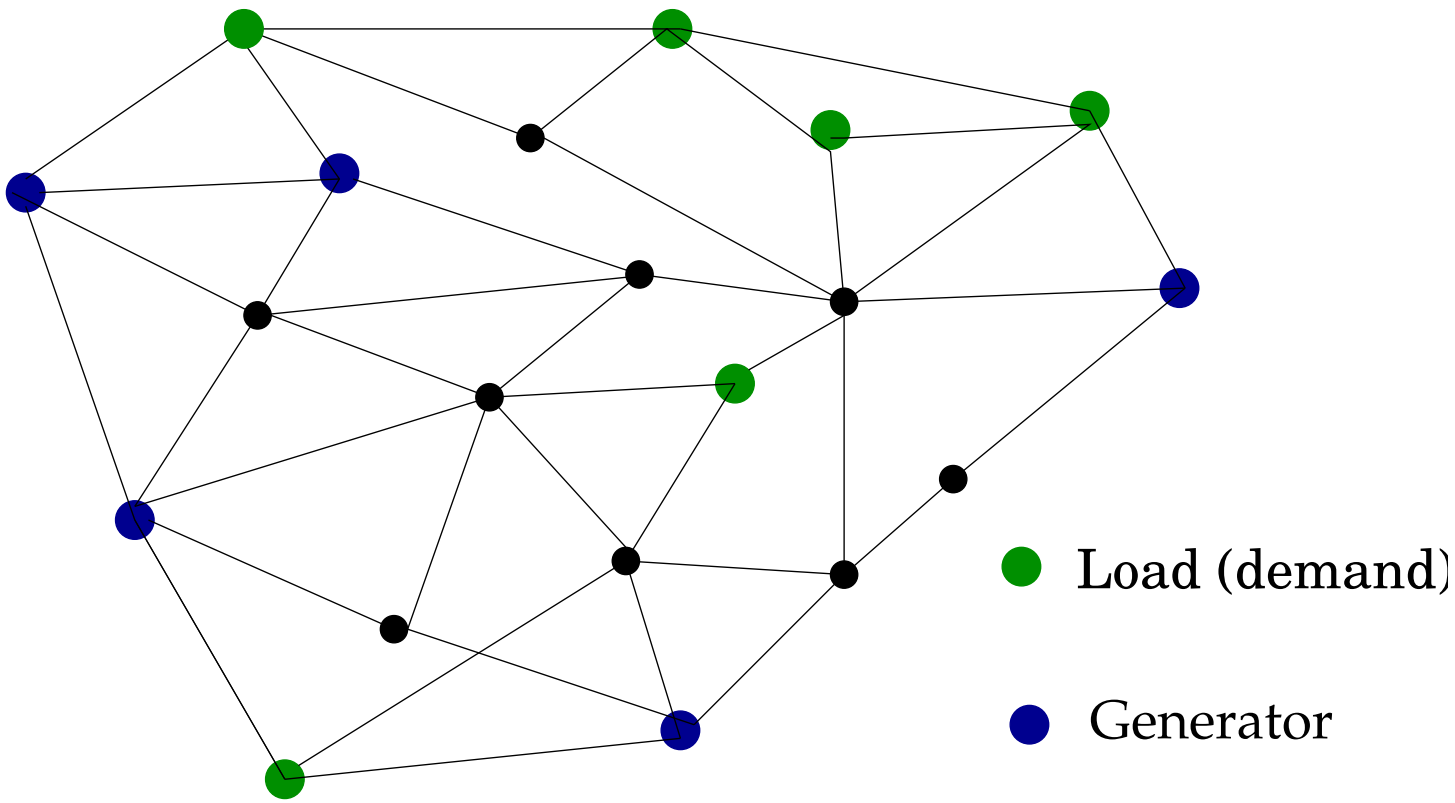
Final solutions similar, but not the same, because of randomness in search.

γ vs iteration count on Run #10

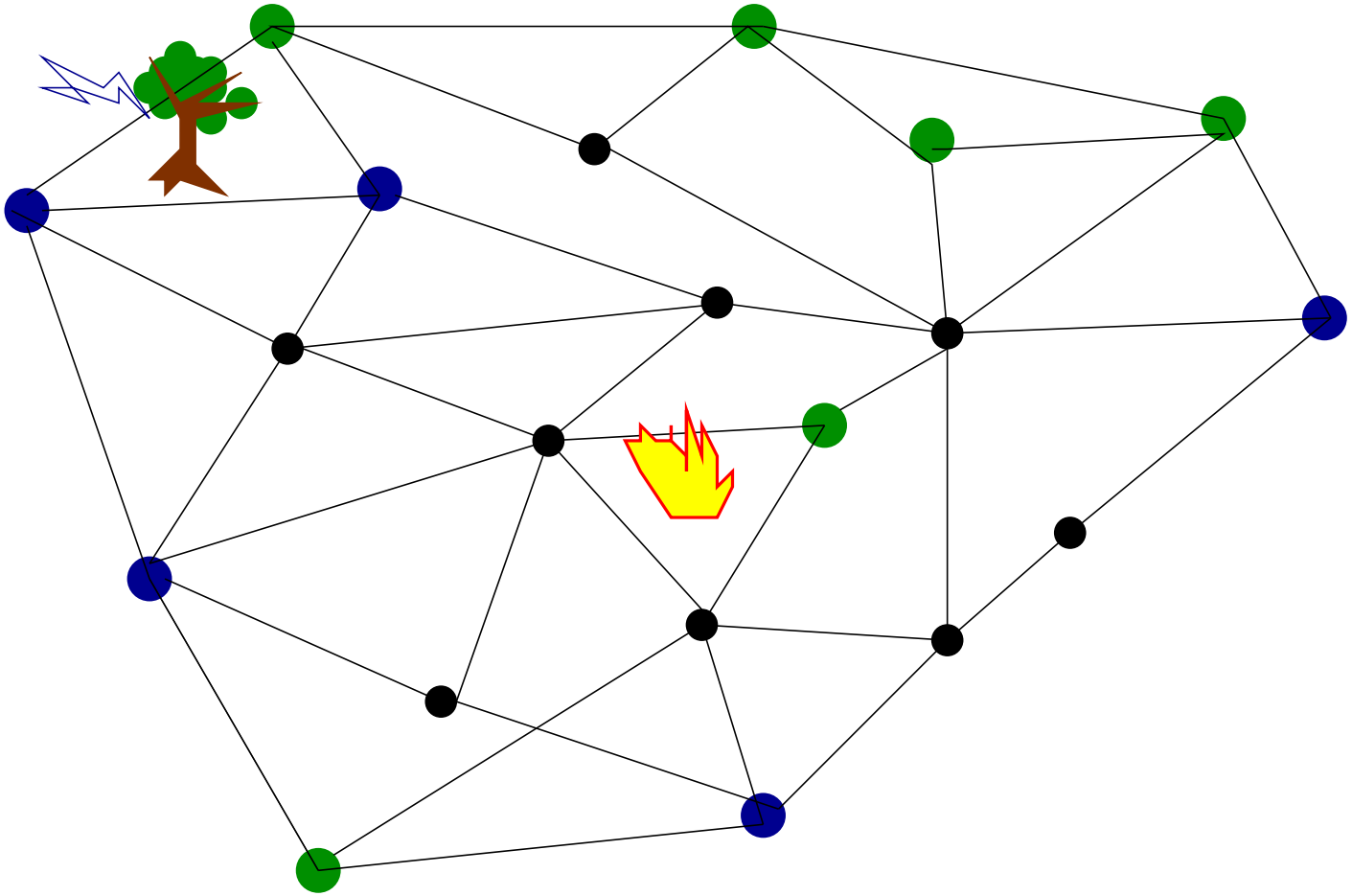


Part II. Cascades

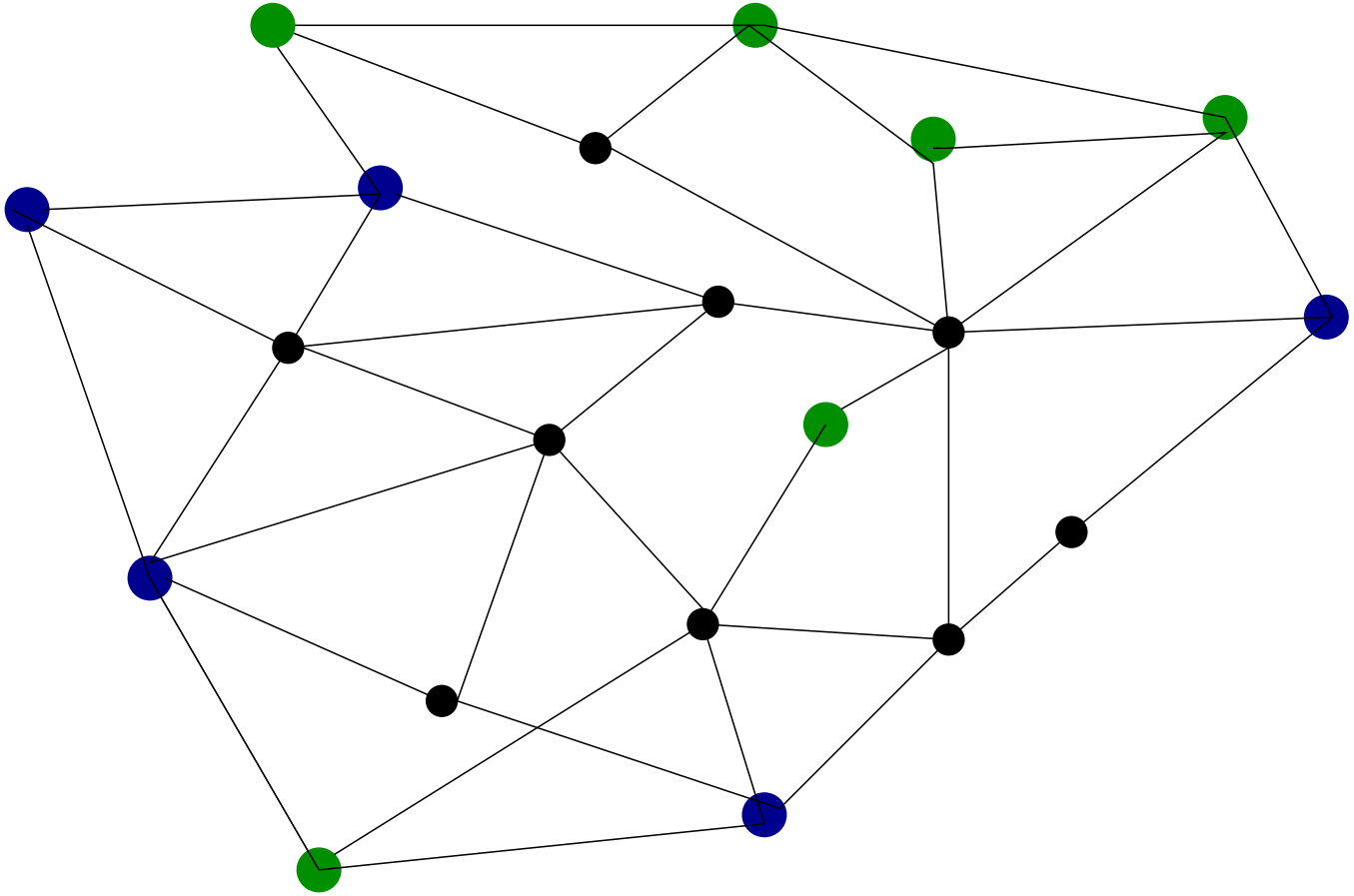
Cascade cartoon - Dobson et al



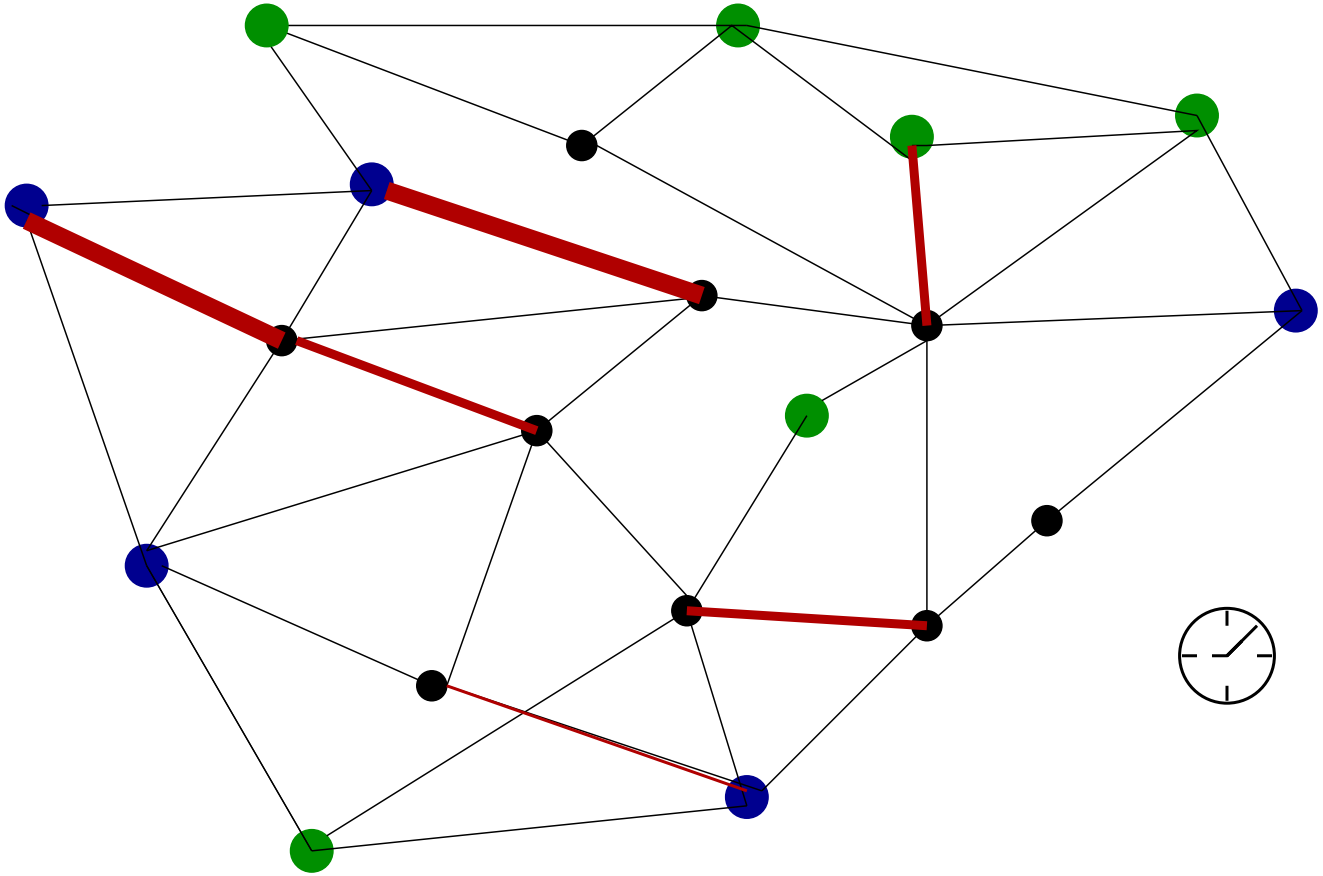
Cascade cartoon



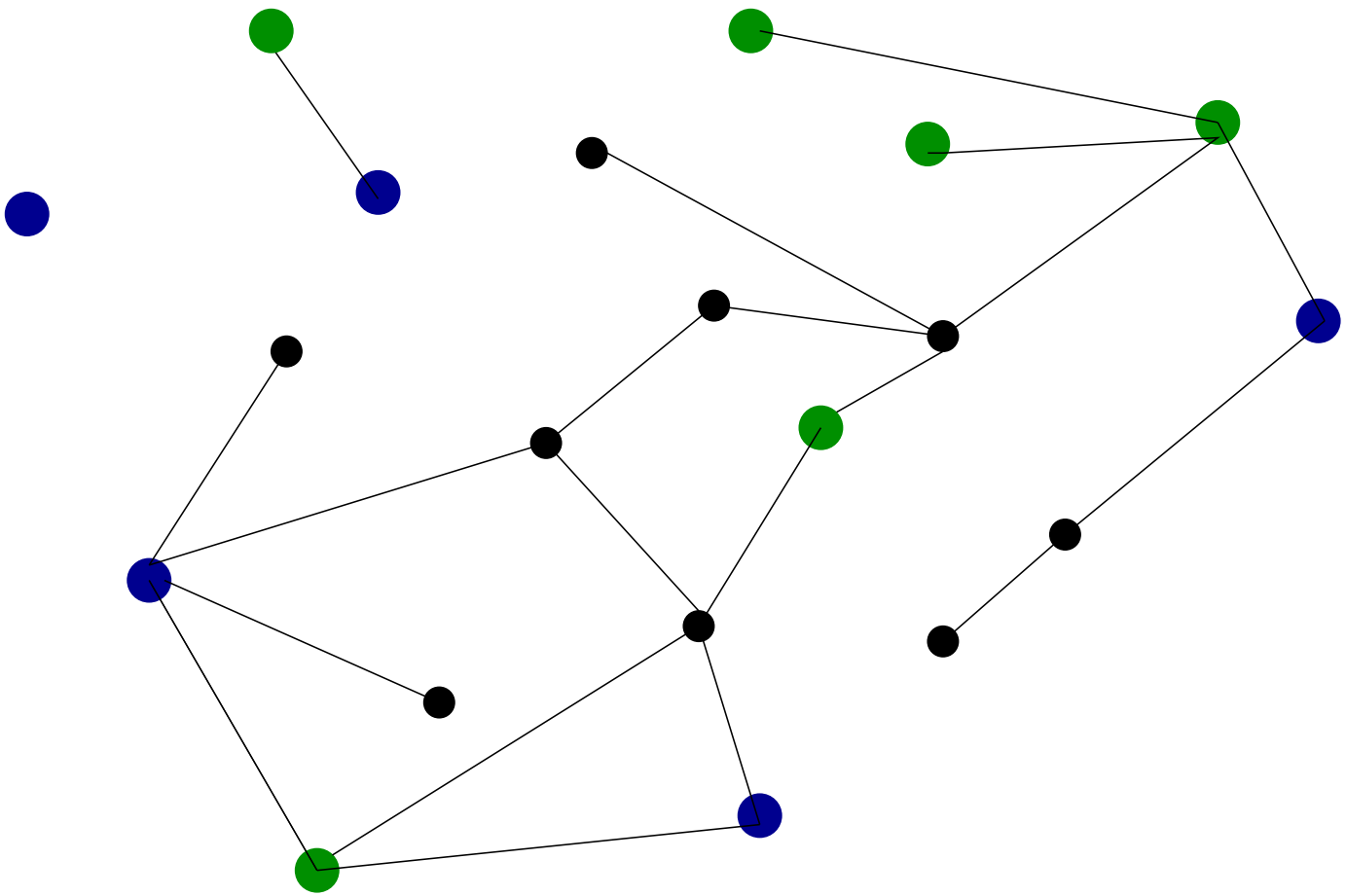
Cascade cartoon



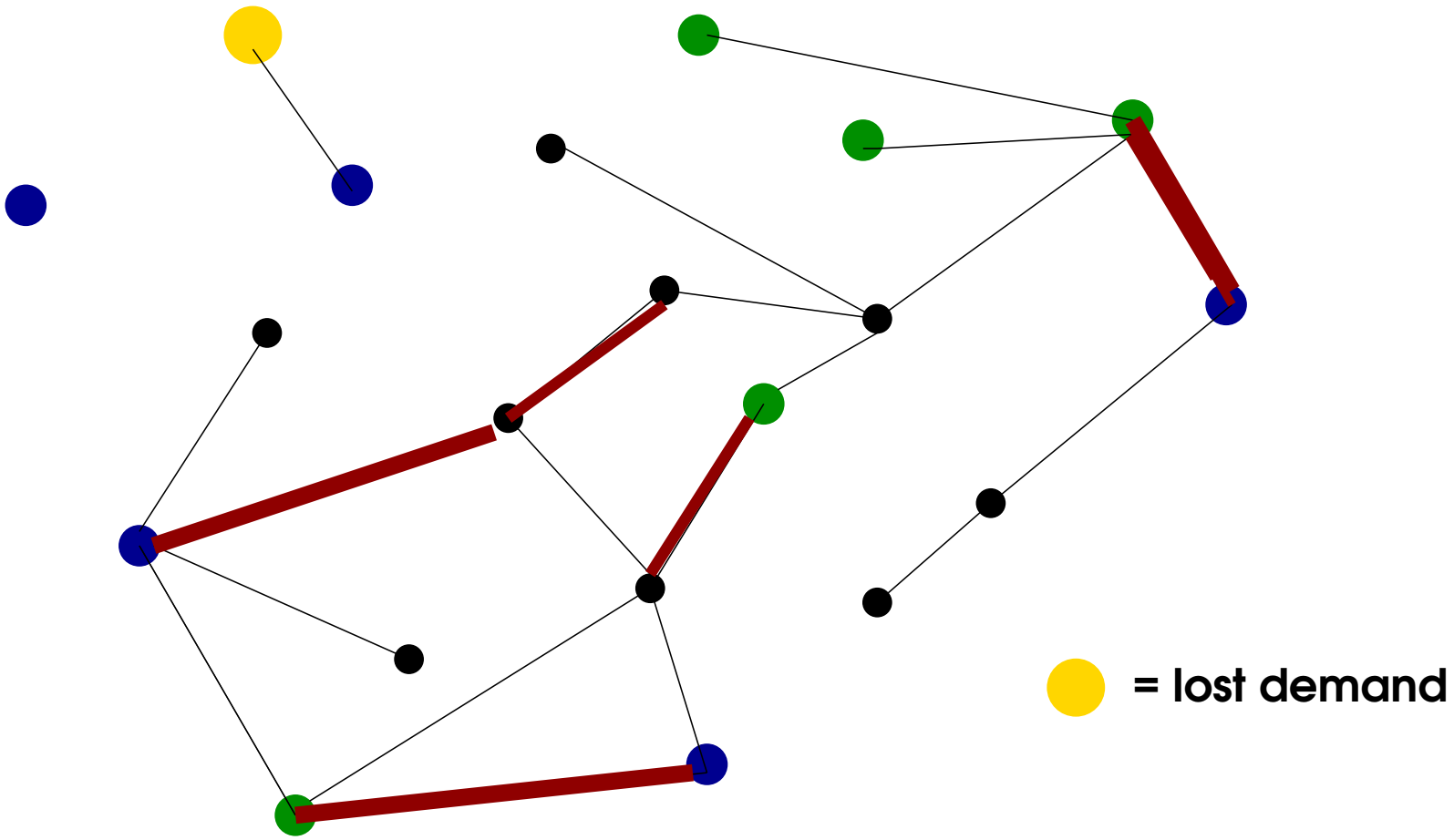
Cascade cartoon



Cascade cartoon



Cascade cartoon



Controlled cascades + detailed cascade model

→ Initial outage event takes place.

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.

2. New power flows are instantiated.

3a. **Take measurements and apply control to shed demand.**

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3a. **Take measurements and apply control to shed demand.**

3b. **Reconfigure generator outputs;**

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3a. **Take measurements and apply control to shed demand.**

3b. **Reconfigure generator outputs; get new power flows.**

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3a. **Take measurements and apply control to shed demand.**

3b. **Reconfigure generator outputs; get new power flows.**

4. The next set of outages takes place.

Controlled cascades + detailed cascade model

$\longrightarrow$ Initial outage event takes place. **Compute control.**

For $\mathbf{r = 1, 2, \ldots, R - 1}$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3a. **Take measurements and apply control to shed demand.**

3b. **Reconfigure generator outputs; get new power flows.**

4. The next set of outages takes place.

At round $\mathbf{R}$, reduce demands so as to remove any line overloads.

Robust control

- Partial information
- Noisy or incorrect information
- Delays

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

$\longrightarrow$ Parameterized by integers $\mathbf{r} > \mathbf{0}$ and $\delta > \mathbf{0}$.

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

$\longrightarrow$ Parameterized by integers $\mathbf{r} > \mathbf{0}$ and $\boldsymbol{\delta} > \mathbf{0}$.

At round $\mathbf{r}$,

- Let $\boldsymbol{\kappa}^{\boldsymbol{\delta}}$ = max **overload** of any line within radius $\boldsymbol{\delta}$ of $\mathbf{v}$

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

$\longrightarrow$ Parameterized by integers $\mathbf{r > 0}$ and $\boldsymbol{\delta} > \mathbf{0}$.

At round $\mathbf{r}$,

- Let $\boldsymbol{\kappa}^{\boldsymbol{\delta}}$ = max **overload** of any line within radius $\boldsymbol{\delta}$ of $\mathbf{v}$
- If $\boldsymbol{\kappa}^{\boldsymbol{\delta}} > \mathbf{c_v^r}$, demand at $\mathbf{v}$ reduced (scaled) by a factor

$$\min\left\{\mathbf{1}, \left\{\mathbf{b_v^r} + \mathbf{s_v^r}\left(\mathbf{c_v^r} - \boldsymbol{\kappa}^{\boldsymbol{\delta}}\right)\right\}^{+}\right\}.$$

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

$\longrightarrow$ Parameterized by integers $\mathbf{r > 0}$ and $\boldsymbol{\delta} > \mathbf{0}$.

At round $\mathbf{r}$,

- Let $\boldsymbol{\kappa}^{\boldsymbol{\delta}}$ = max **overload** of any line within radius $\boldsymbol{\delta}$ of $\mathbf{v}$
- If $\boldsymbol{\kappa}^{\boldsymbol{\delta}} > \mathbf{c_v^r}$, demand at $\mathbf{v}$ reduced (scaled) by a factor

$$\min\left\{1, \left\{b_v^r + s_v^r\left(c_v^r - \kappa^\delta\right)\right\}^+\right\}.$$

Example: $(1, 1, s)$ control; scale = $\min\left\{1, \{1 + s\,(1 - \kappa)\}^+\right\}$.

Adaptive affine controls

For each demand bus n $\mathbf{v}$, and round $\mathbf{r}$, control triple $\mathbf{c_v^r}$, $\mathbf{b_v^r}$, $\mathbf{s_v^r}$

$\longrightarrow$ Parameterized by integers $\mathbf{r > 0}$ and $\boldsymbol{\delta} > \mathbf{0}$.

At round $\mathbf{r}$,

- Let $\boldsymbol{\kappa}^{\boldsymbol{\delta}}$ = max **overload** of any line within radius $\boldsymbol{\delta}$ of $\mathbf{v}$
- If $\boldsymbol{\kappa}^{\boldsymbol{\delta}} > \mathbf{c_v^r}$, demand at $\mathbf{v}$ reduced (scaled) by a factor

$$\min\left\{1, \left\{b_v^r + s_v^r\left(c_v^r - \kappa^\delta\right)\right\}^+\right\}.$$

Example: $(1, 1, s)$ control; scale = $\min\left\{1, \{1 + s\,(1-\kappa)\}^+\right\}$.

Problem: choose control so as to maximize demand at end of round $\mathbf{R}$.

Methodology

- General approach: simulation-based optimization
- Better fundamentals: stochastic gradient algorithm applied to (randomly) smoothed version of the problem
- Parallel implementation on 40 i7 cores using Unix sockets
- Each gradient computation takes a few seconds (per core) on real grids with tens of thousands of lines

Sept 2011 San Diego blackout

Joint work with A. Bernstein, D. Hay, G. Zussman, M. Uzunoglu (EE Dept. Columbia)

Sept 2011 San Diego blackout

Joint work with A. Bernstein, D. Hay, G. Zussman, M. Uzunoglu (EE Dept. Columbia)

- Act of god: human error + still unknown

Sept 2011 San Diego blackout

Joint work with A. Bernstein, D. Hay, G. Zussman, M. Uzunoglu (EE Dept. Columbia)

- Act of god: human error + still unknown
- **We do not have complete or exact data**

Sept 2011 San Diego blackout

Joint work with A. Bernstein, D. Hay, G. Zussman, M. Uzunoglu (EE Dept. Columbia)

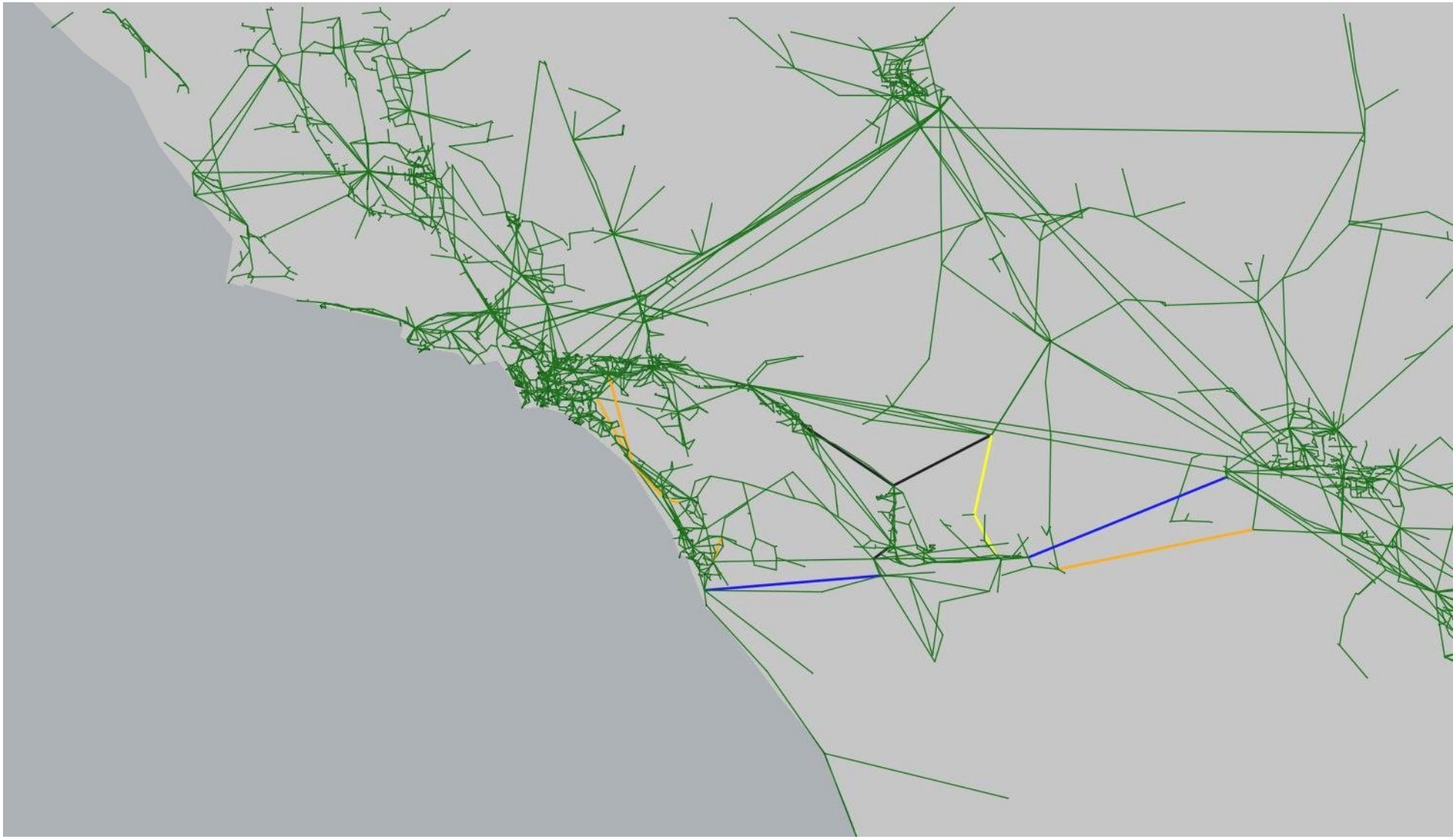
- Act of god: human error + still unknown
- **We do not have complete or exact data**
- Nevertheless, in our simulations a cascade takes place, with similar characteristics at the initial stages (rounds)

Sept 2011 San Diego blackout

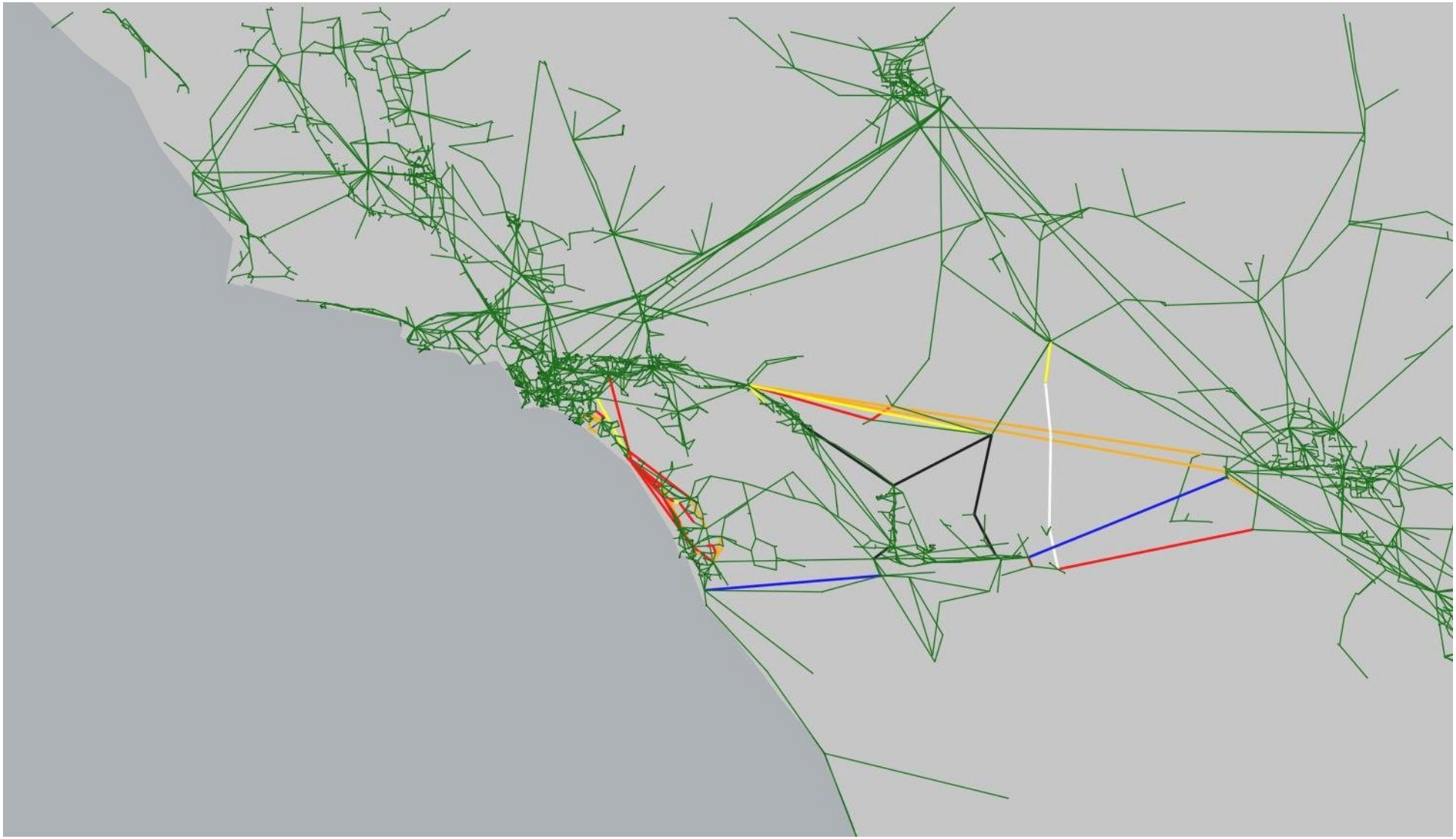
Joint work with A. Bernstein, D. Hay, G. Zussman, M. Uzunoglu (EE Dept. Columbia)

- Act of god: human error + still unknown
- **We do not have complete or exact data**
- Nevertheless, in our simulations a cascade takes place, with similar characteristics at the initial stages (rounds)
- Data should be made available to **all** researchers

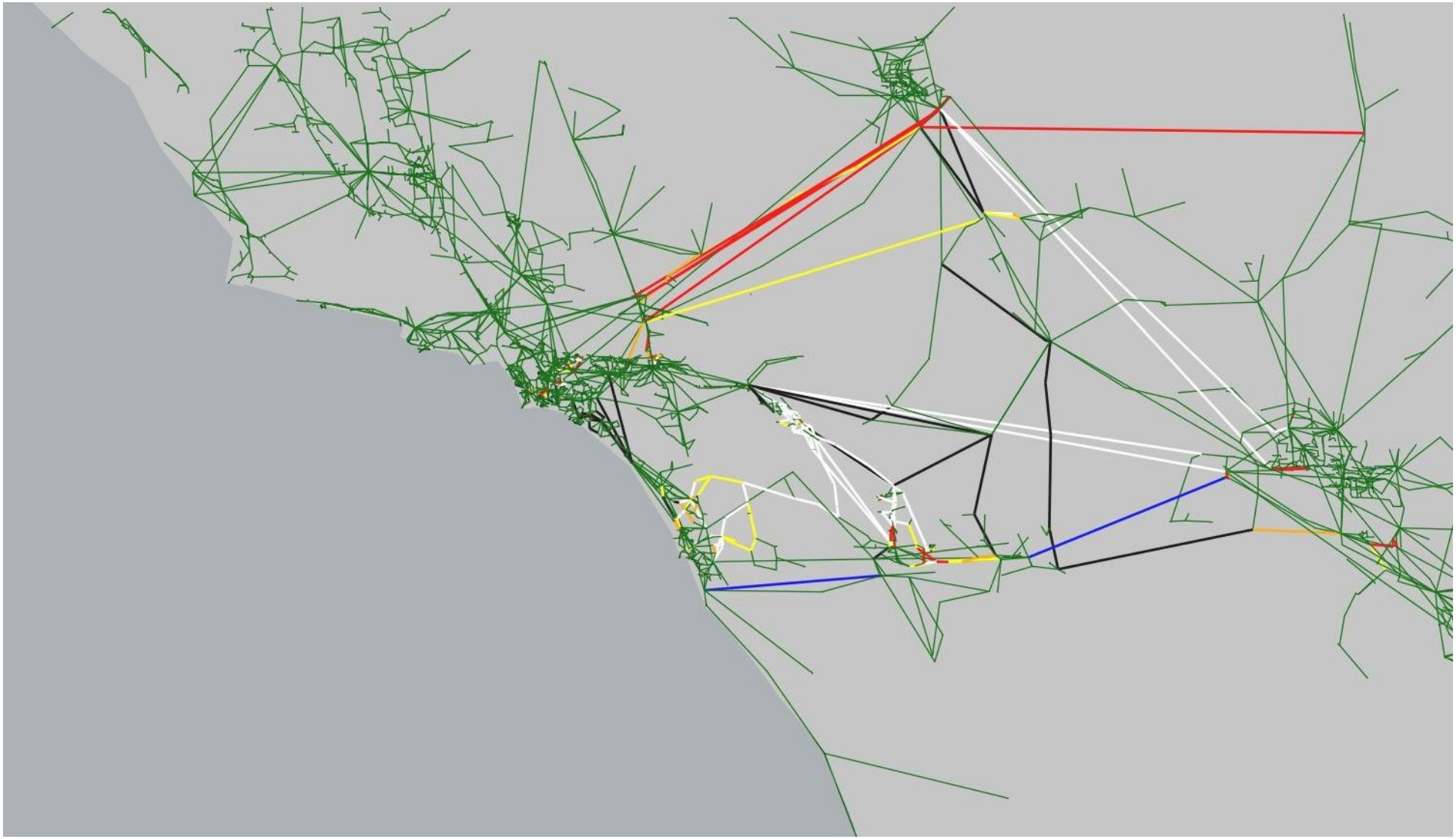
Beginning



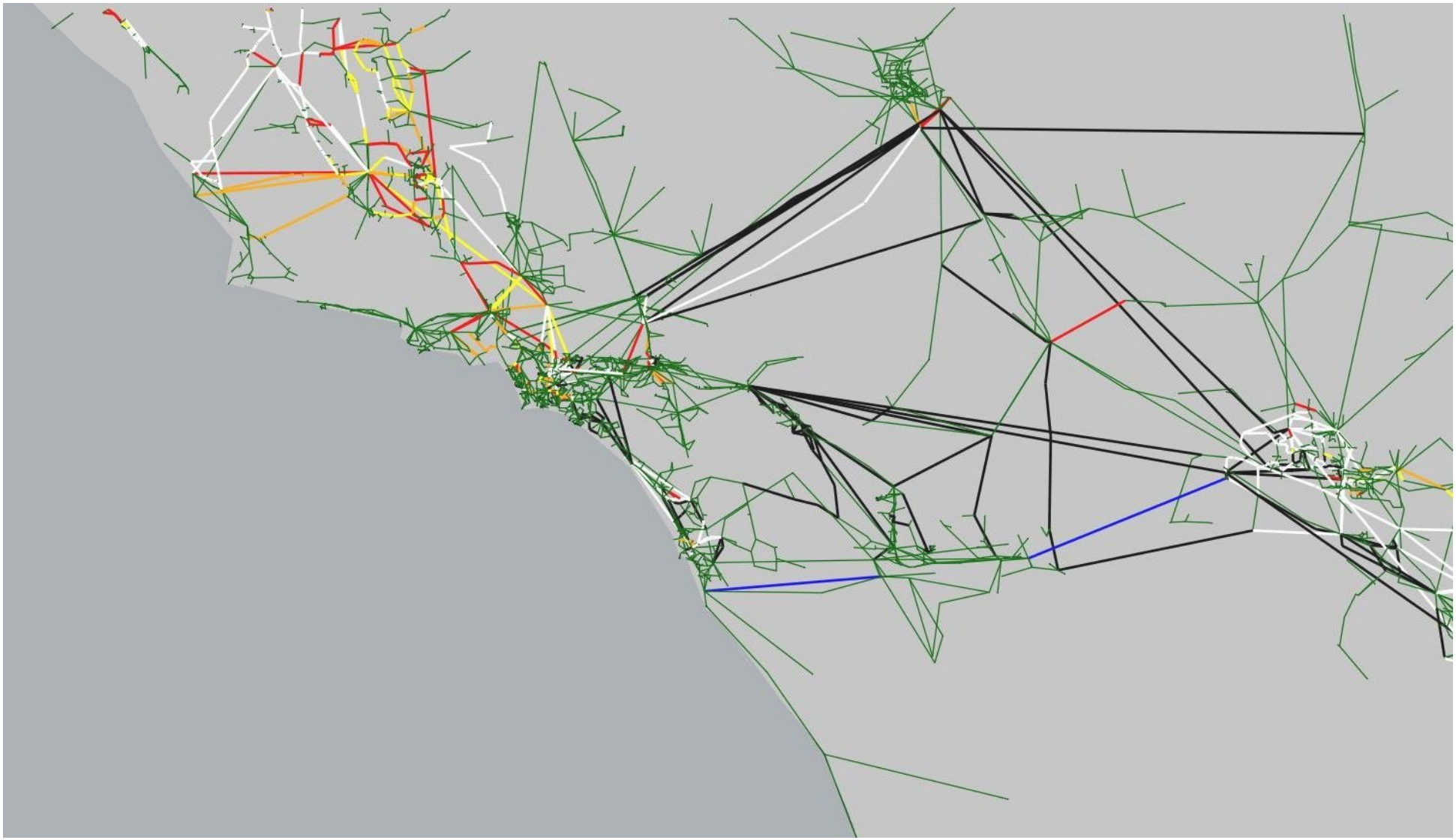
More



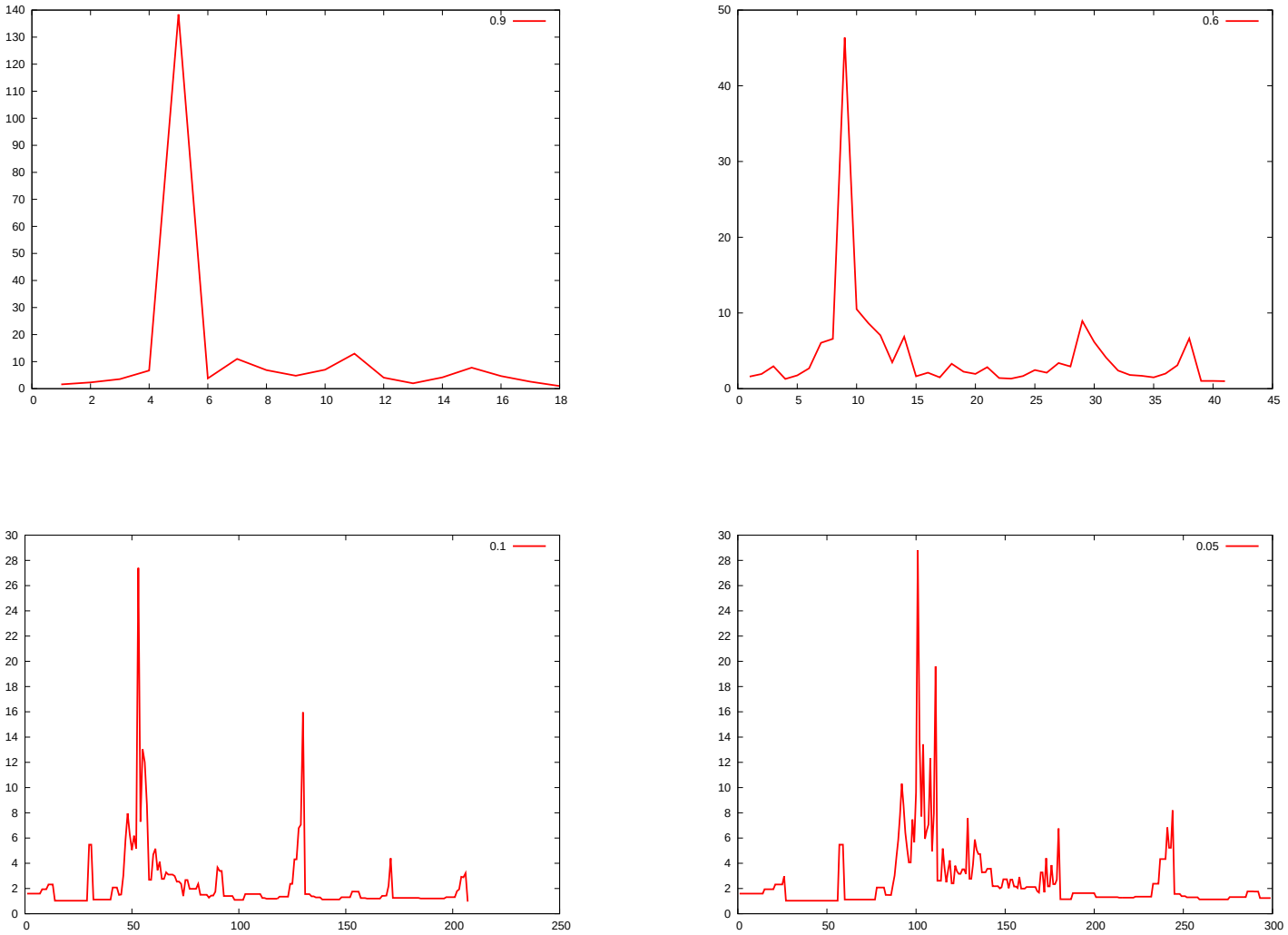
More



More



Max line overload during cascade with increasing resolution



"Ideal" control

Stop cascade by shedding a minimum amount of load

$\longrightarrow$ Remove all line overloads
$\longrightarrow$ System limitations (e.g. generator "ramp up/down" speeds)

(solve an optimization problem)

"Ideal" control

Stop cascade by shedding a minimum amount of load

$\longrightarrow$ Remove all line overloads
$\longrightarrow$ System limitations (e.g. generator "ramp up/down" speeds)

(solve an optimization problem)

- Round 1, yield = **71.74%**,
- Round 20, yield = **51.50%**,
- Round 30, yield = **98.44%**,
- Round 60, yield = **87.88%**,
- Round 110, yield = **66.59%**,

"Ideal" control

Stop cascade by shedding a minimum amount of load

$\longrightarrow$ Remove all line overloads
$\longrightarrow$ System limitations (e.g. generator "ramp up/down" speeds)

(solve an optimization problem)

- Round 1, yield = **71.74%**,
- Round 20, yield = **51.50%**,
- Round 30, yield = **98.44%**,
- Round 60, yield = **87.88%**,
- Round 110, yield = **66.59%**,

But: very nonrobust and requires much coordination

Near optimal scaling control

Simple, robust

- Applies control around round 28, only
- Yield = **95.10%**, **8** outaged lines
- Cascade stops around round **29**

Near optimal scaling control

Simple, robust

- Applies control around round 28, only
- Yield = **95.10%**, **8** outaged lines
- Cascade stops around round **29**

No-control approach:

- yield = **61.47%**,
- $>$ **300** rounds, $>$ **800** outaged lines