Problems and solutions in nonlinear mixed-integer programming

Daniel Bienstock
Columbia University

IMA, 2016

Talk outline

1. Some light entertainment

2. Some mathematics

3. Additional entertainment

Why we should study polynomial optimization:

cascading failures of power grids

- In August 2003, a cascading failure of the Eastern Interconnect caused a large and long-lasting blackout
- The Eastern Interconnect is the electrical circuit that we are in
- The blackout affected some fifty million people for several days and cost a lot of money
- In September, 2003, a similar blackout affected most of Italy

Recent cascades

- U.S. Northeast and Canada; Italy, 2003
- San Diego, 2011
- India, 2012

Rising concerns

- Increasing demand, increasing scope and complexity of grids
- Too expensive to add extensive capacity
- Use of renewables desirable but adds stochastic risk
- Malevolent action (?)

Cascade dynamics

(0) Stuff happens ("act of God"): some network elements

(power lines, generators, transformers, etc)

are **disabled**

Cascade dynamics

(0) Stuff happens ("act of God"): some network elements

(power lines, generators, transformers, etc)

are **disabled**

(1) Power flows are rearranged: mostly due to physics

Cascade dynamics

(0) Stuff happens ("act of God"): some network elements

(power lines, generators, transformers, etc)

are **disabled**

(1) Power flows are rearranged: mostly due to physics

(2) As a result some network elements become overloaded

Cascade dynamics

(0) Stuff happens ("act of God"): some network elements

(power lines, generators, transformers, etc)

are **disabled**

(1) Power flows are rearranged: mostly due to physics

(2) As a result some network elements become overloaded

(3) At a later time, some of these become tripped or outaged

Cascade dynamics

(0) Stuff happens ("act of God"): some network elements

(power lines, generators, transformers, etc)

are **disabled**

(1) Power flows are rearranged: mostly due to physics

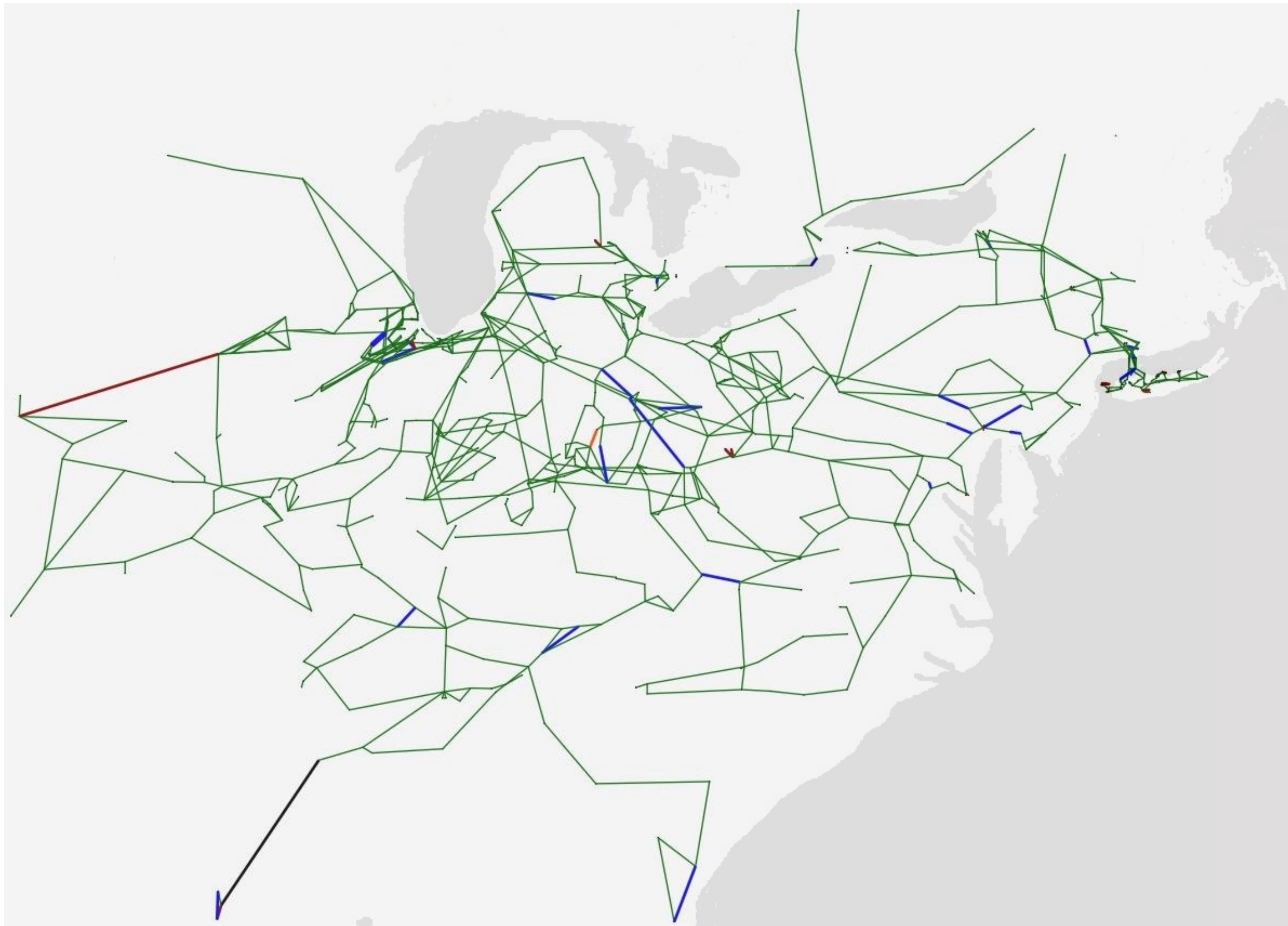
(2) As a result some network elements become overloaded

(3) At a later time, some of these become tripped or outaged

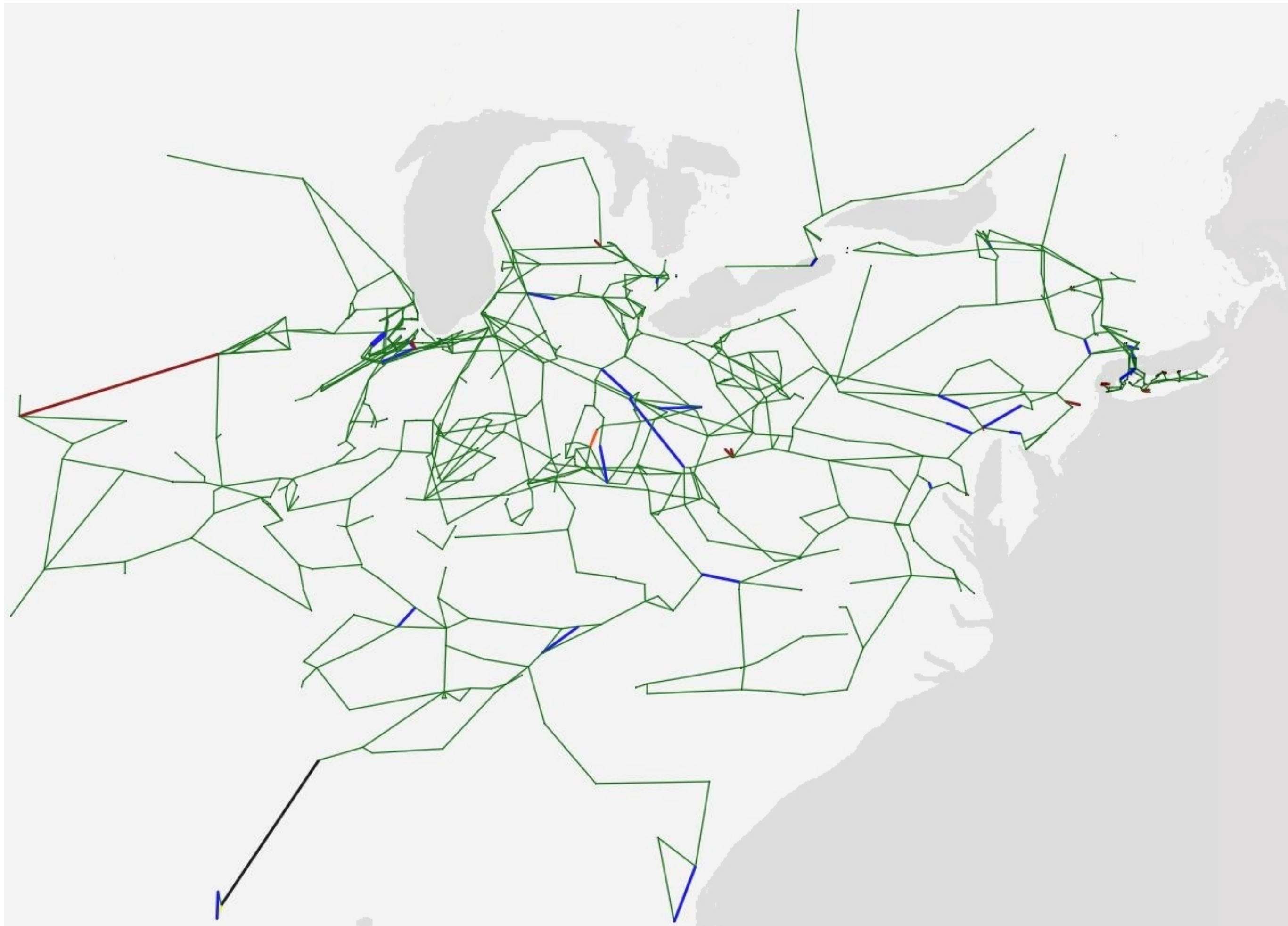
(4) Go to **(1)**.

Let's go to the movies

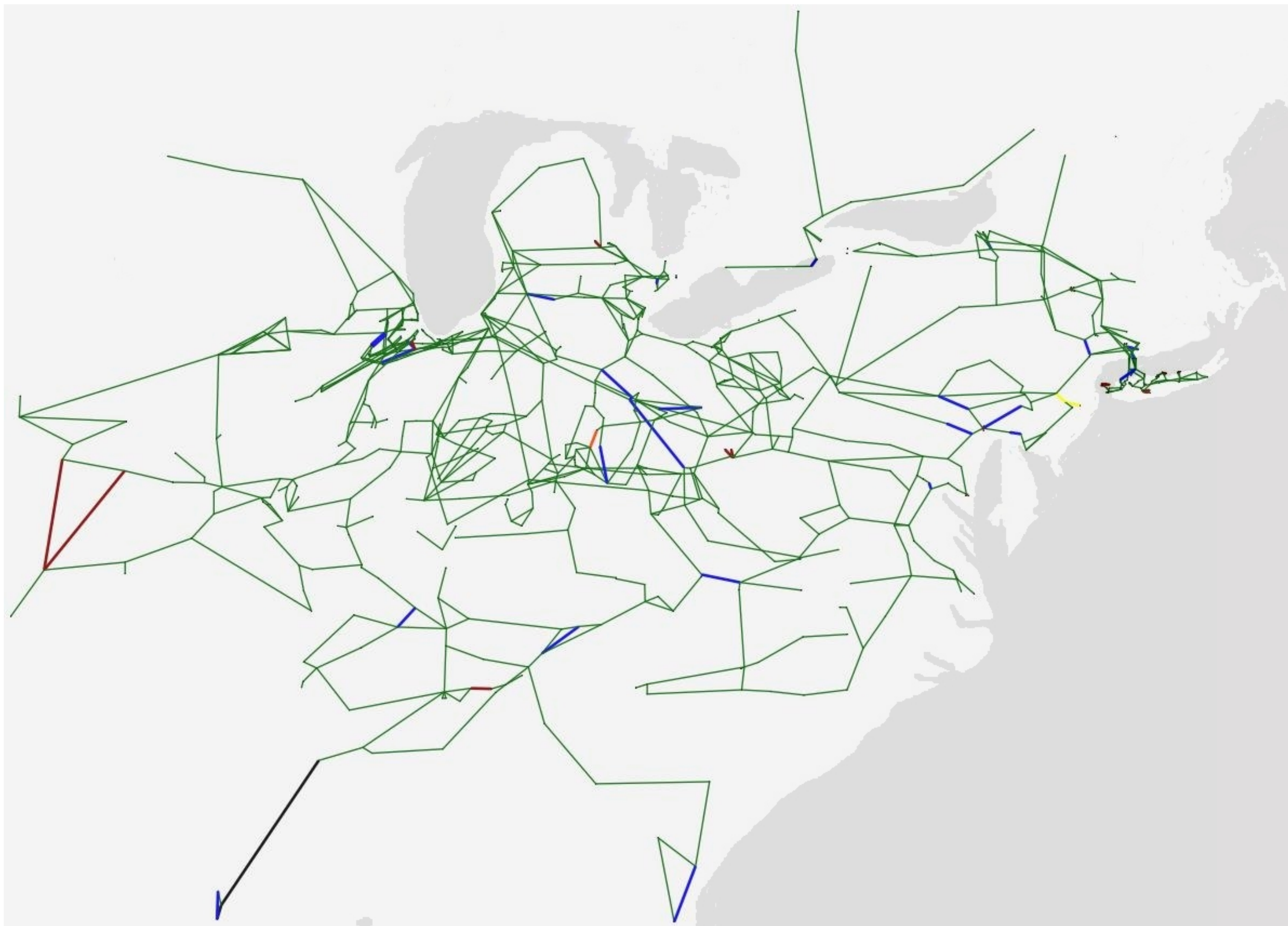
Simulated cascade of Eastern Interconnect



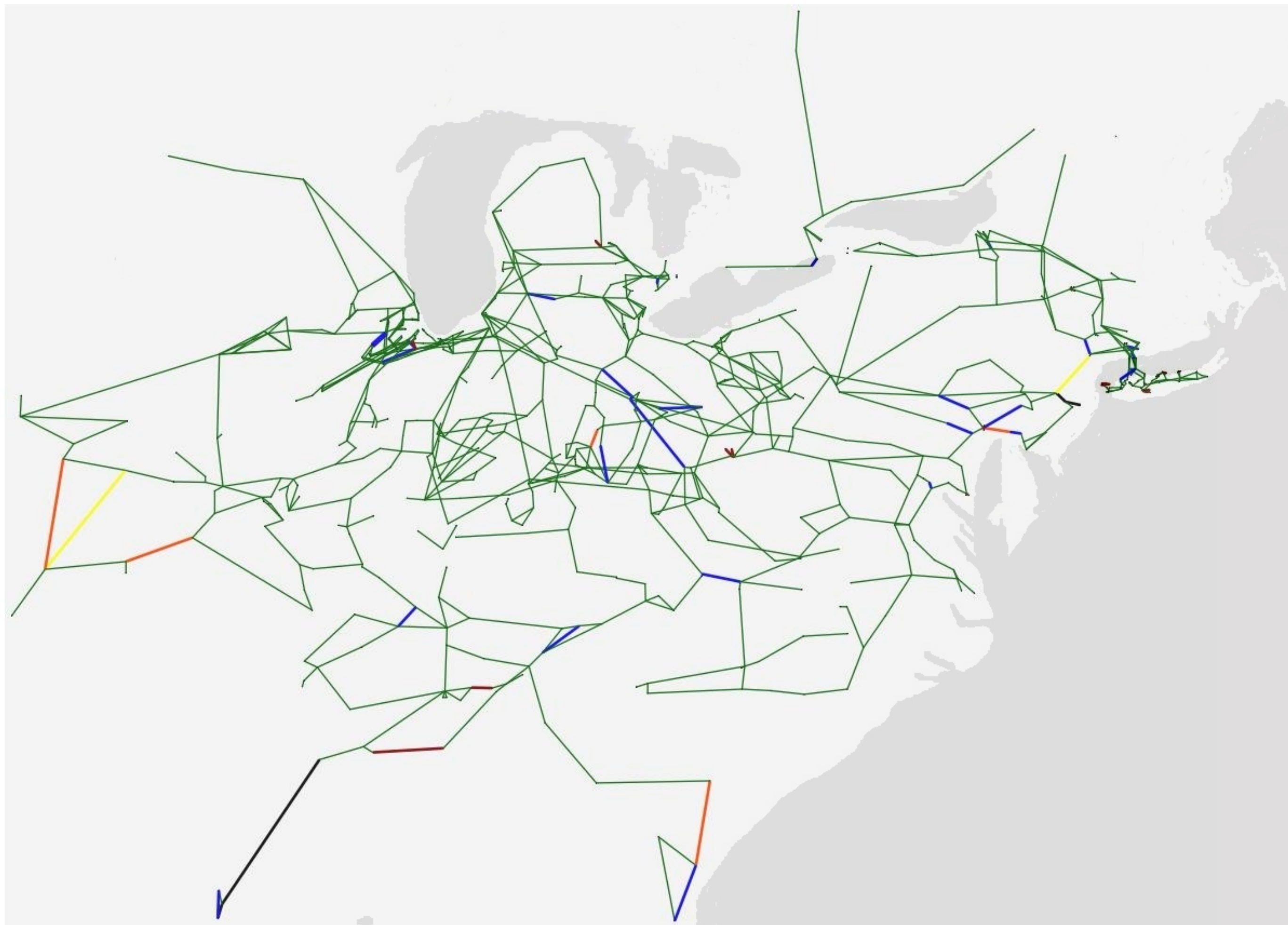
Simulated cascade of Eastern Interconnect



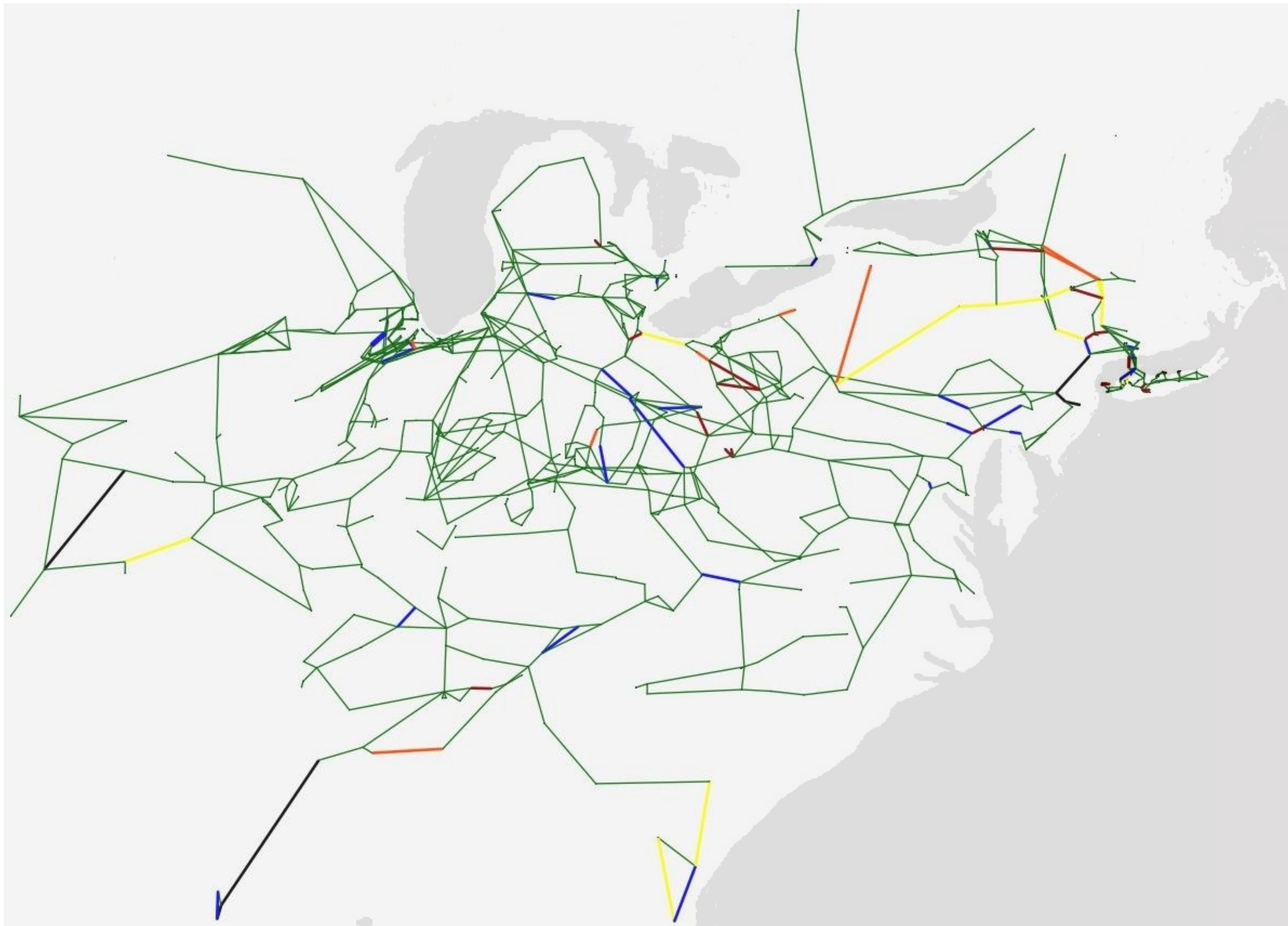
Simulated cascade of Eastern Interconnect



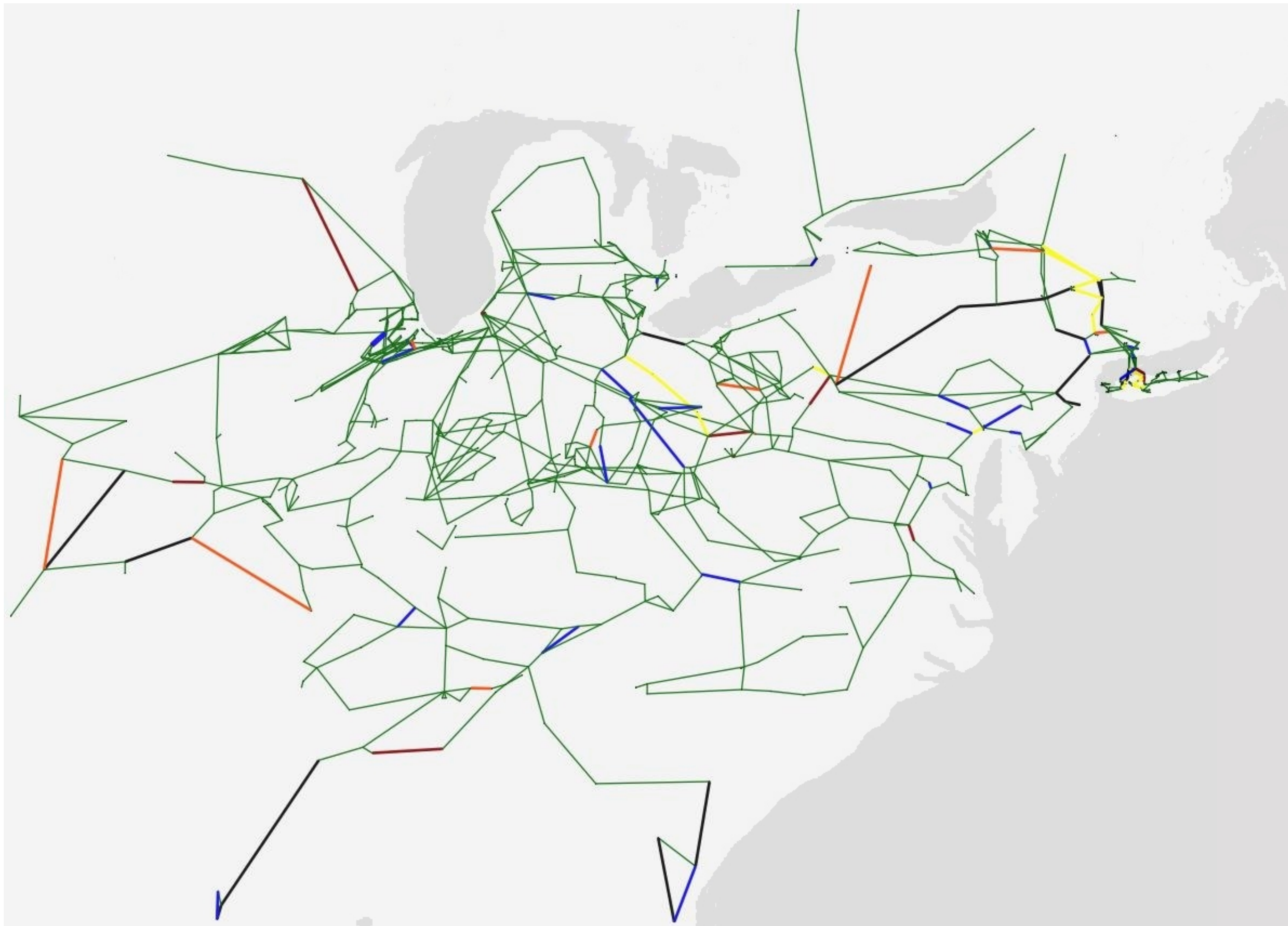
Simulated cascade of Eastern Interconnect



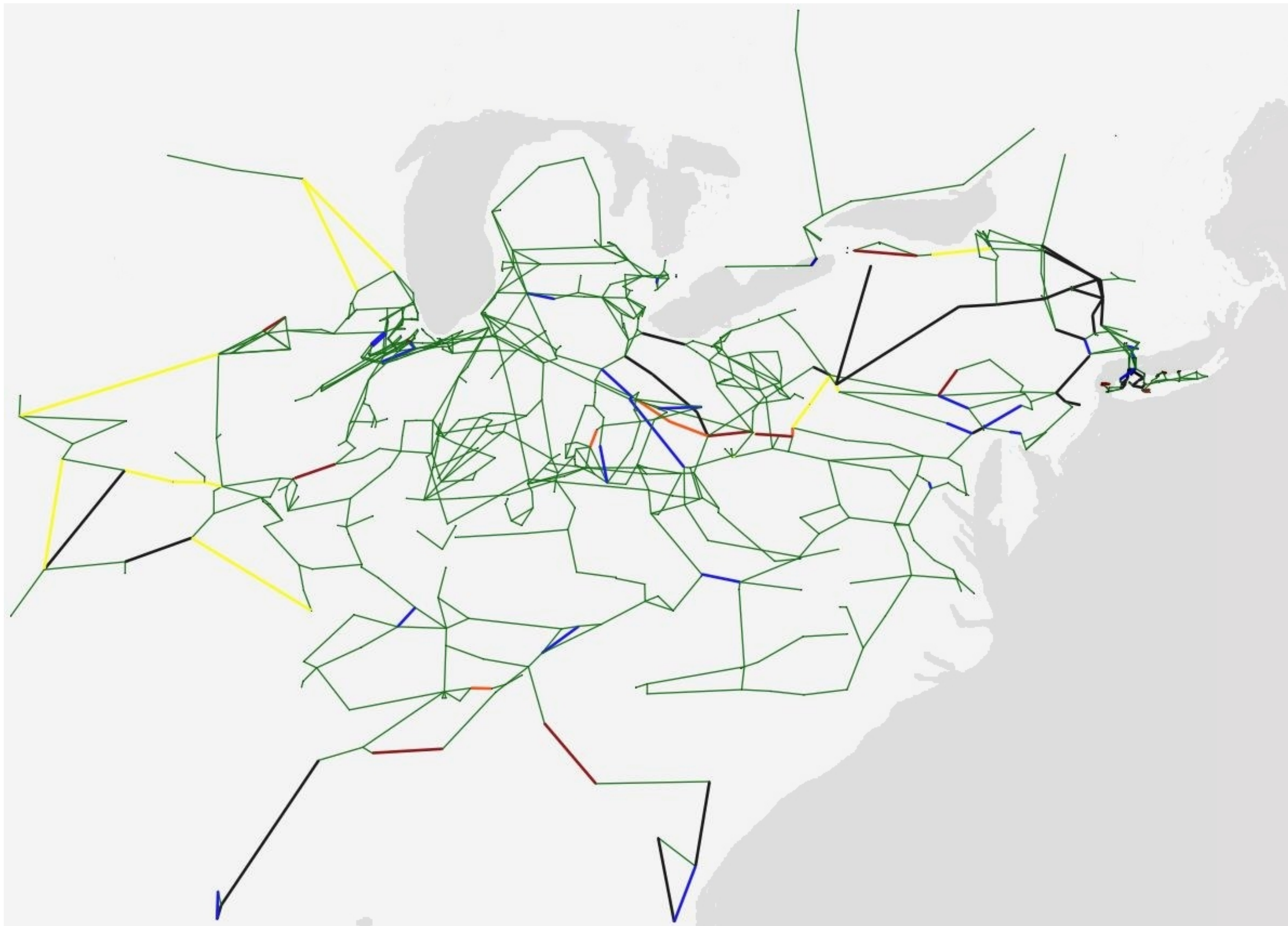
Simulated cascade of Eastern Interconnect



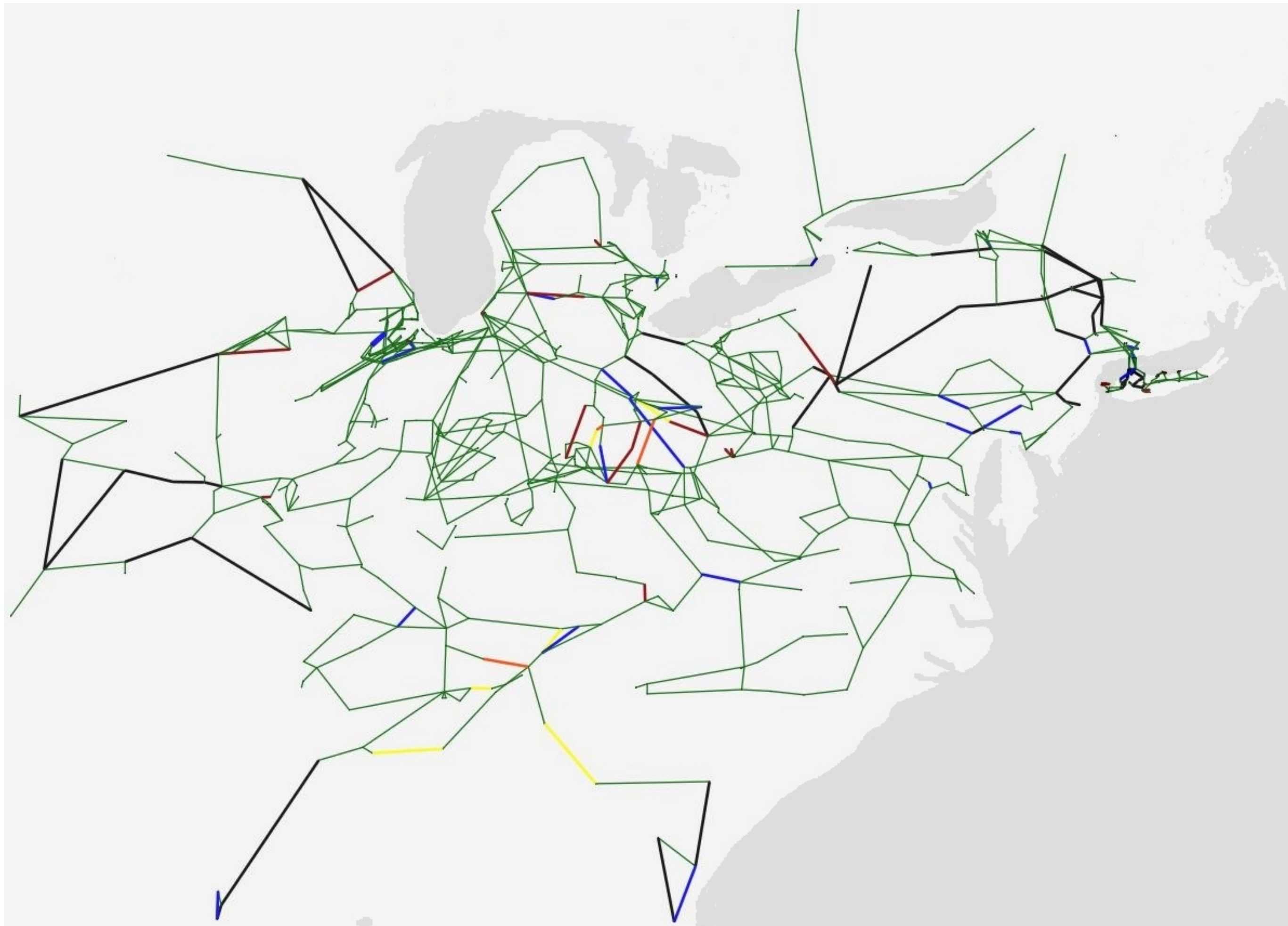
Simulated cascade of Eastern Interconnect



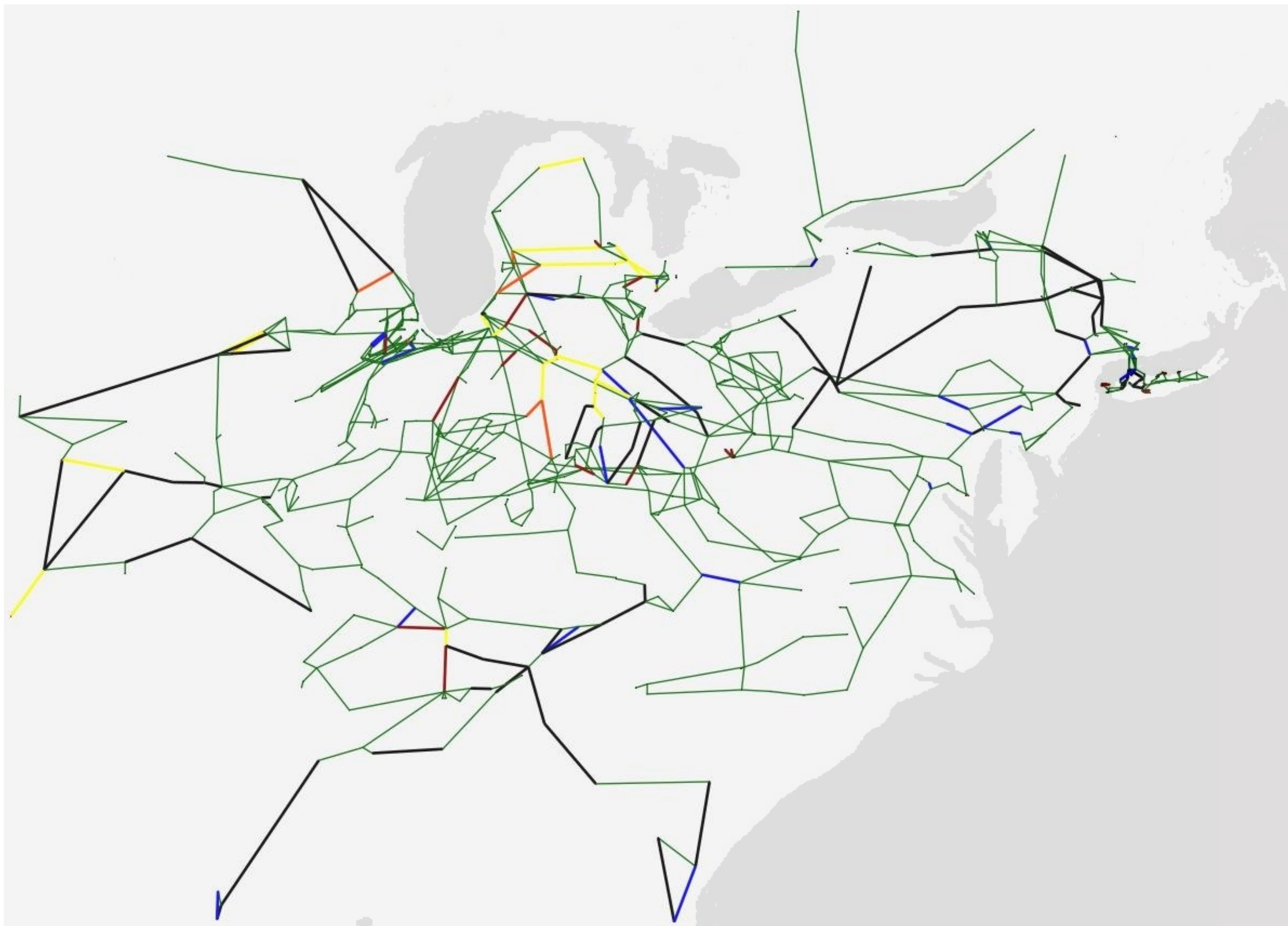
Simulated cascade of Eastern Interconnect



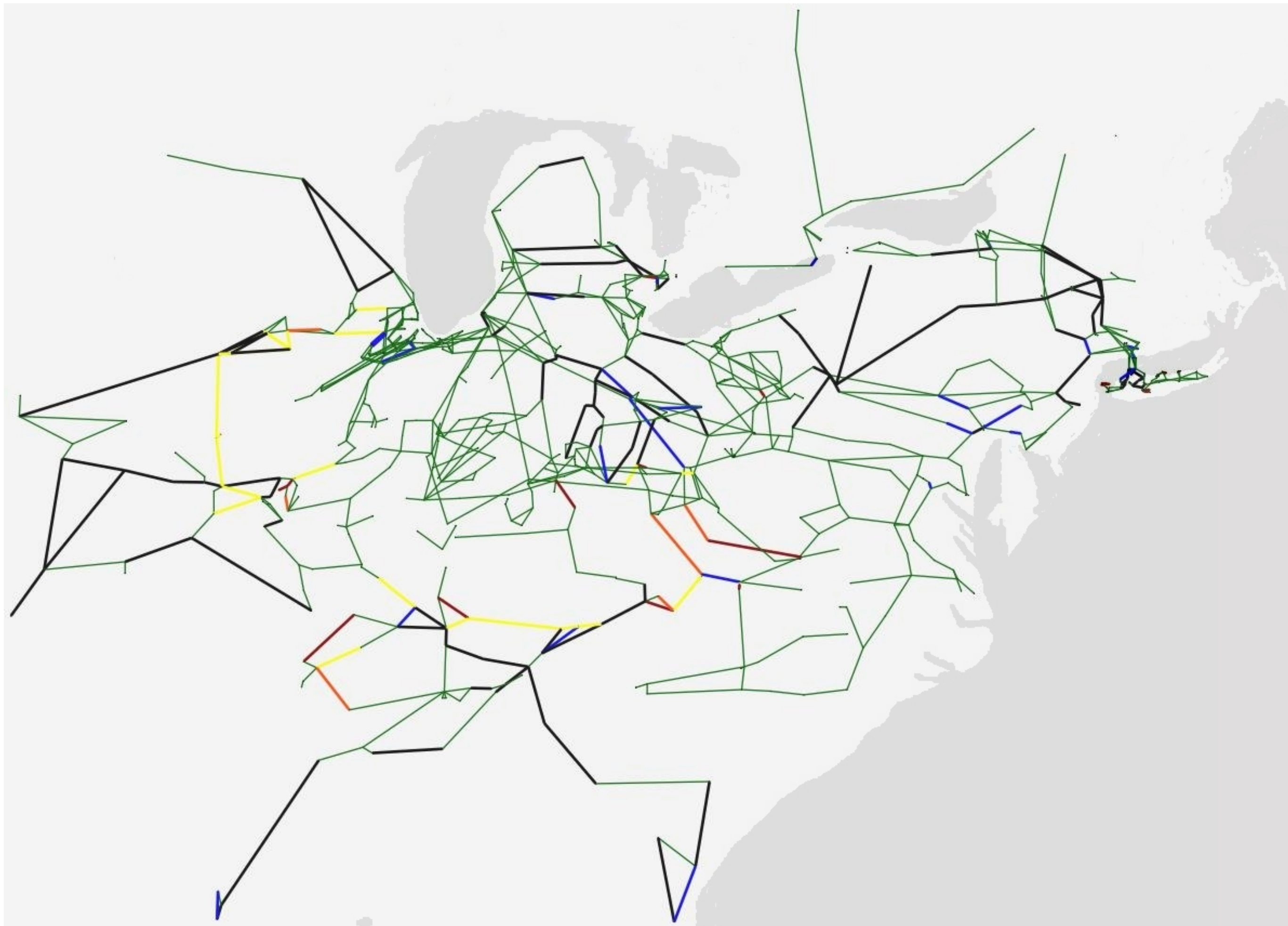
Simulated cascade of Eastern Interconnect



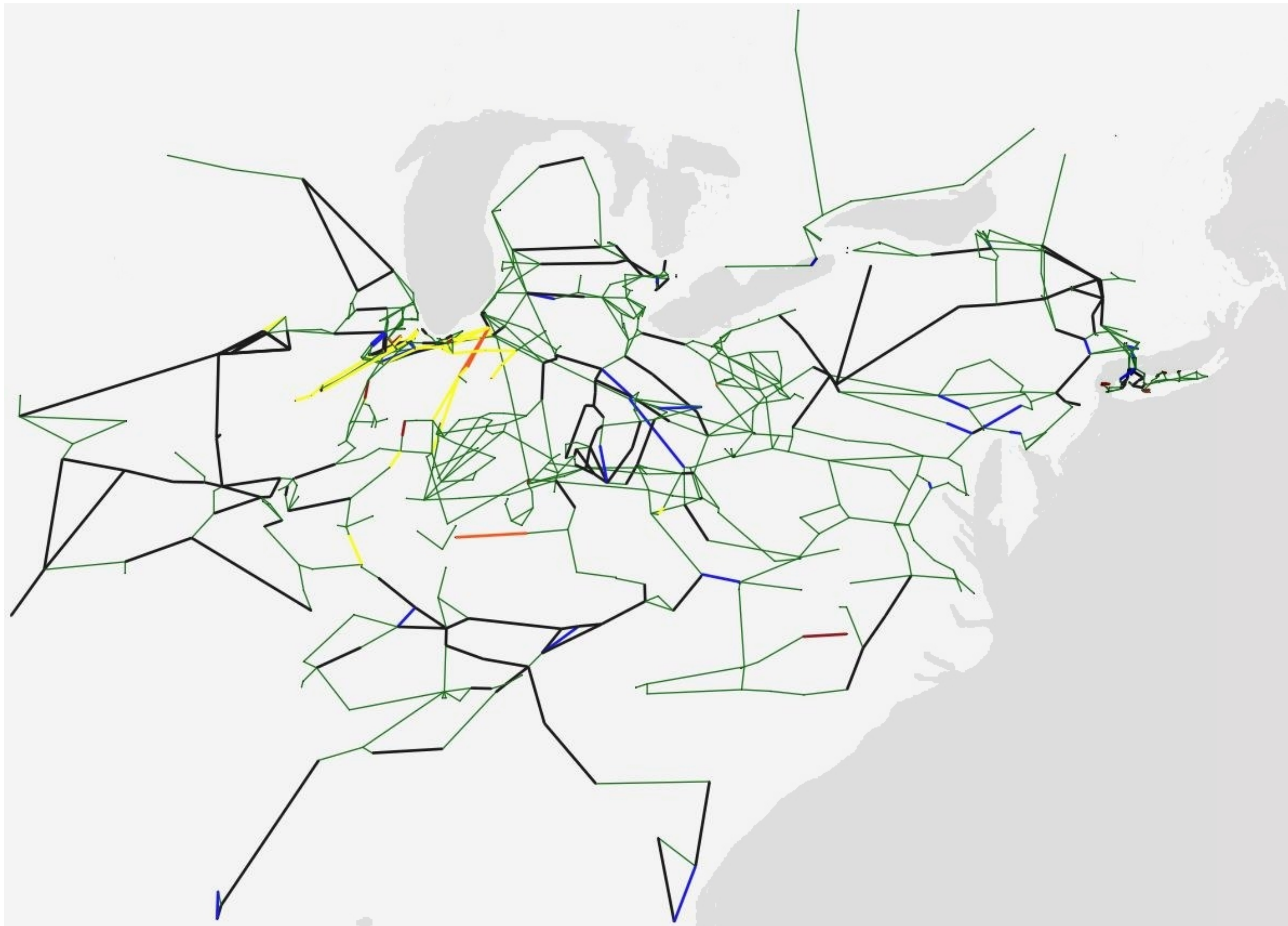
Simulated cascade of Eastern Interconnect



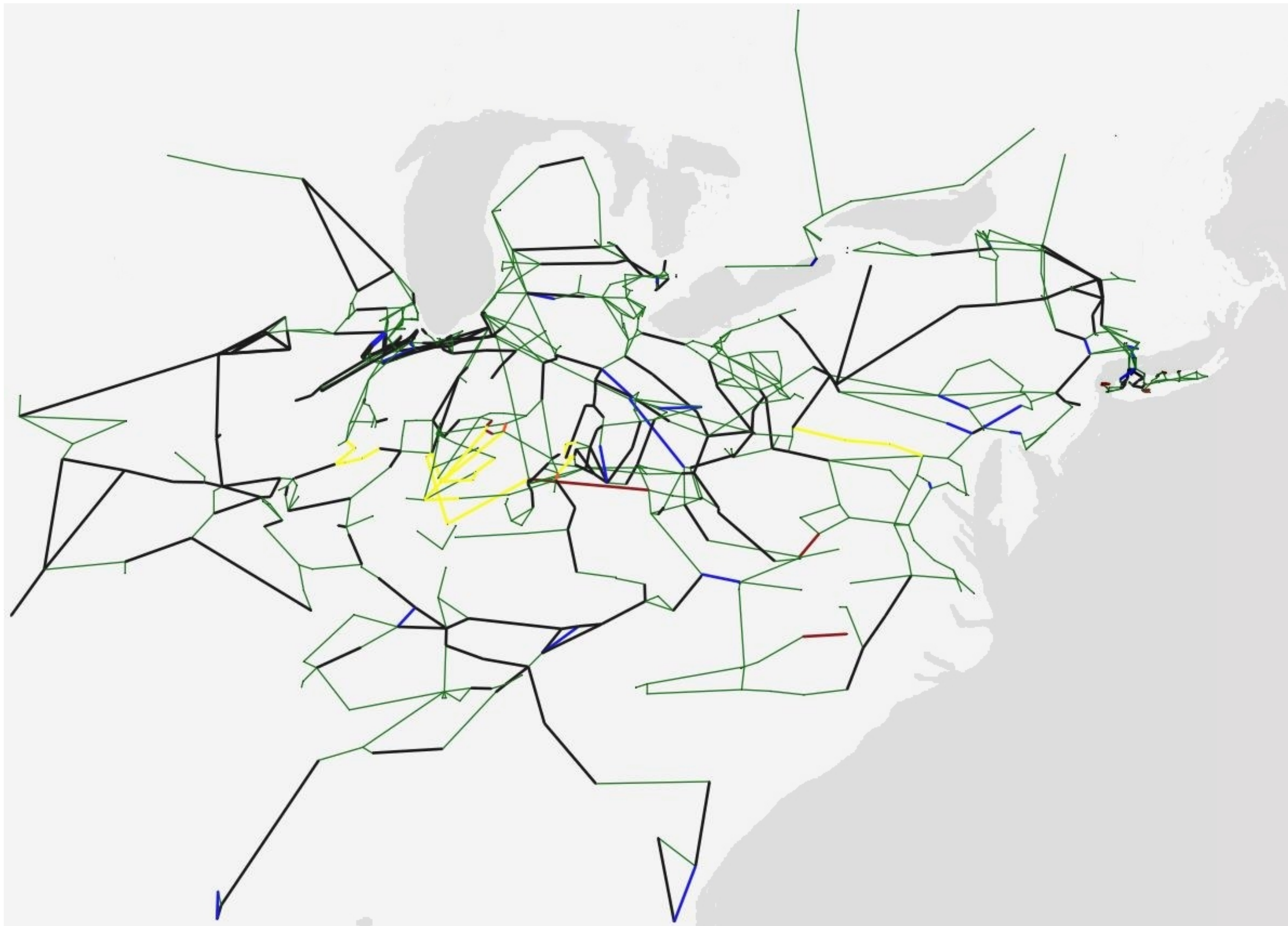
Simulated cascade of Eastern Interconnect



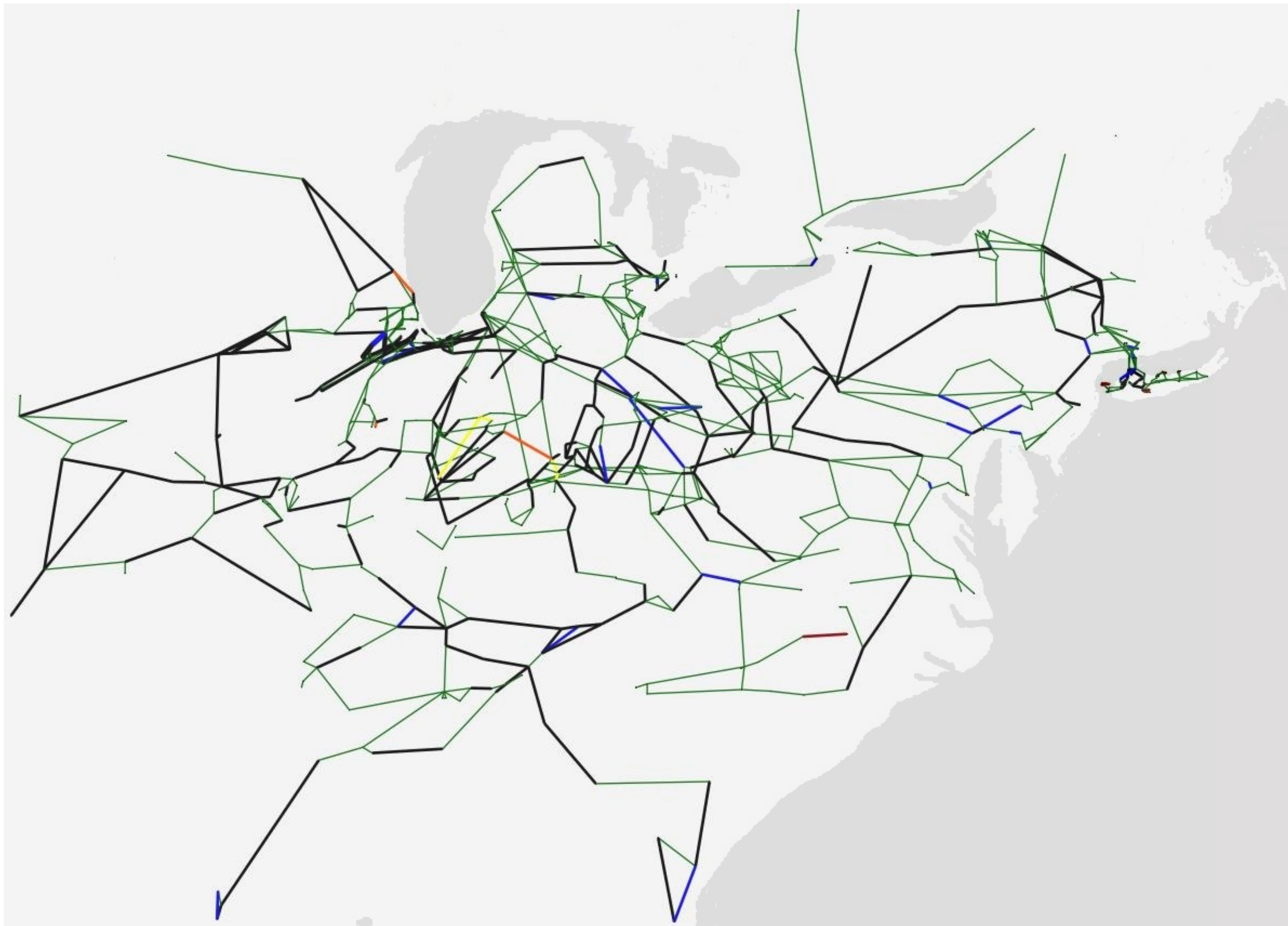
Simulated cascade of Eastern Interconnect



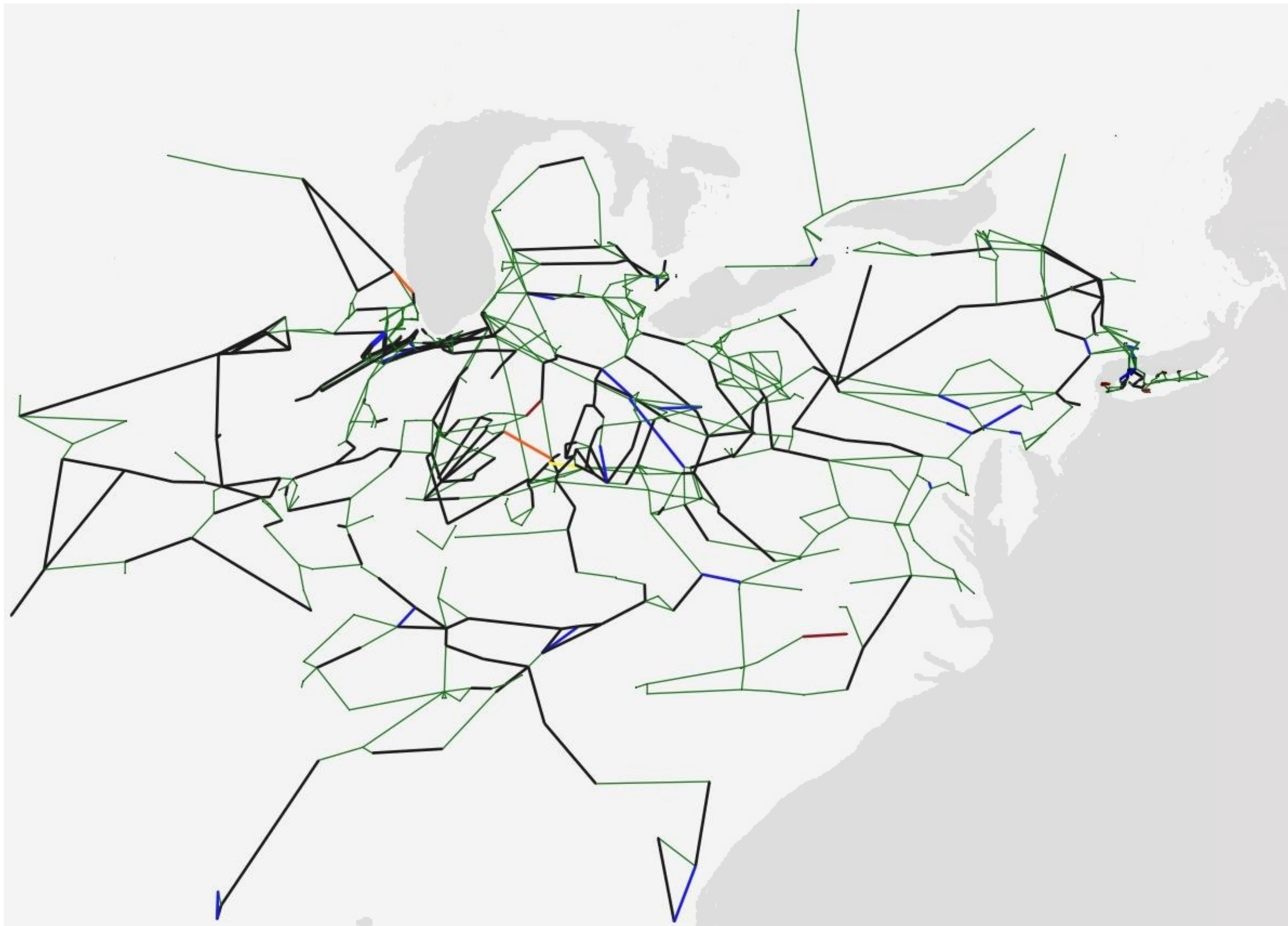
Simulated cascade of Eastern Interconnect



Simulated cascade of Eastern Interconnect



Simulated cascade of Eastern Interconnect



Simulated cascade of Eastern Interconnect

Back to reality:

why it is hard to simulate a power grid under distress

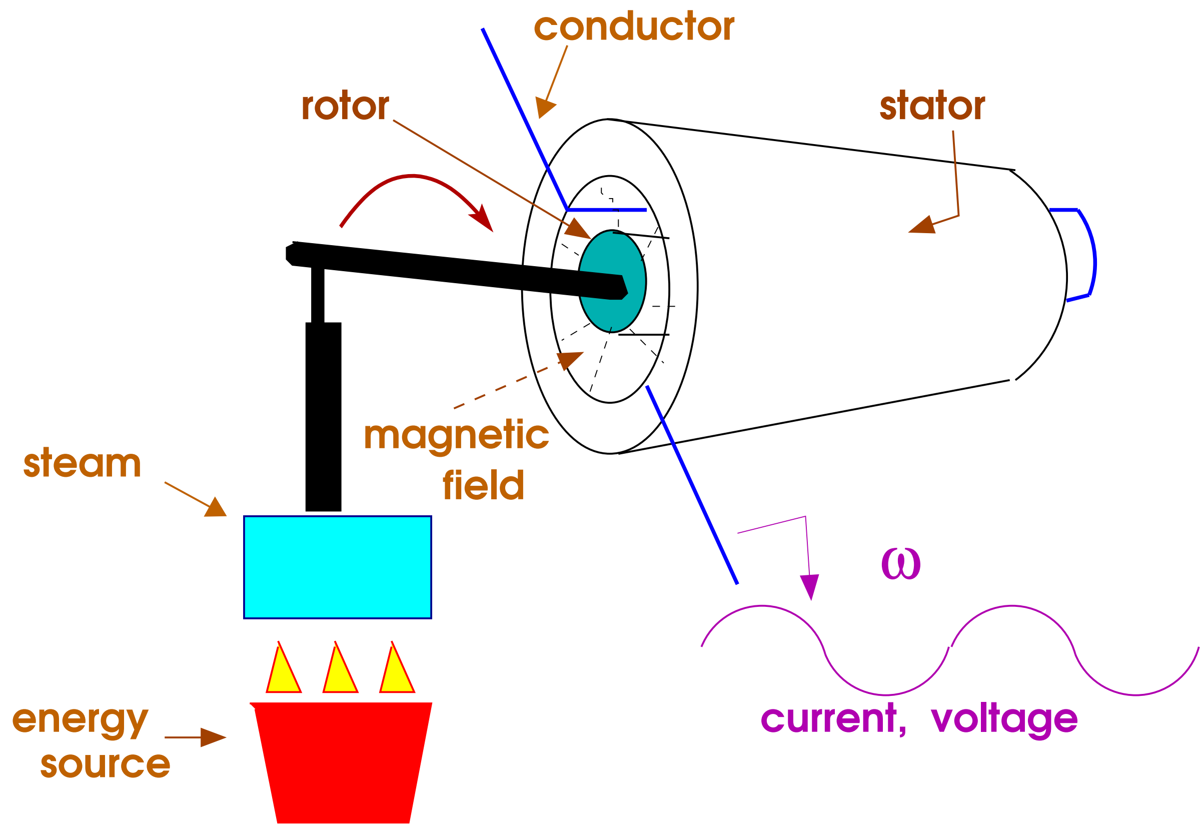
(1) We have to explain when and why equipment will fail

(2) This requires an understanding of the physics of power flows

(3) Additionally, there is noise, missing information, and more

$\rightarrow$ let's begin with **(2)**.

The Grid


conductor
rotor
stator
steam
magnetic
field
ω
energy
source
current, voltage

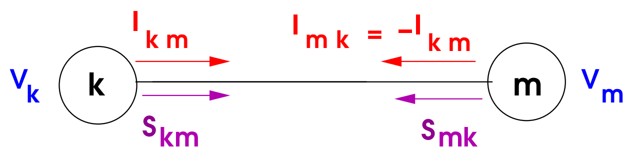
Voltage, Power, Current

Real-time voltage (potential energy) at bus (node) k:

$$V_k(t) = \hat{V}_k \cos(\omega t + \theta_k)$$

Steady-state (time average over one period of length $2\pi/\omega$): voltage

at bus k represented as: $= \hat{V}_k e^{j\theta_k} = \hat{V}_k(\cos\theta_k + j\sin\theta_k)$



- I_{km} = (complex) **current** injected into km at k

- S_{km} = (complex) **power** injected into km at k

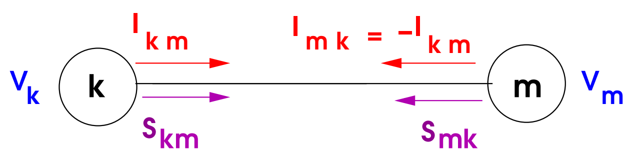
Voltage, Power, Current

Real-time **voltage** (potential energy) at bus (node) $\boldsymbol{k}$:

$$V_k(t) \;=\; \hat{V}_k \cos(\omega t + \theta_k)$$

Steady-state (time average over one period of length $2\pi/\omega$): **voltage**

at bus $\boldsymbol{k}$ represented as: $= \hat{V}_k e^{j\theta_k} \;=\; \hat{V}_k(\cos\theta_k \;+\; j\sin\theta_k)$



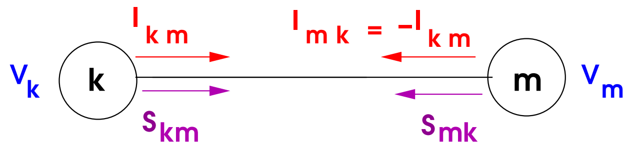
- $\boldsymbol{I_{km}}$ = (complex) **current** injected into $\boldsymbol{km}$ at $\boldsymbol{k}$

- $\boldsymbol{S_{km}}$ = (complex) **power** injected into $\boldsymbol{km}$ at $\boldsymbol{k}$

- **Ohm's law:** $I_{km} \;=\; y_{km}(V_k - V_m)$ (y_{km} = admittance)

- $S_{km} \;=\; V_k I_{km}^*$

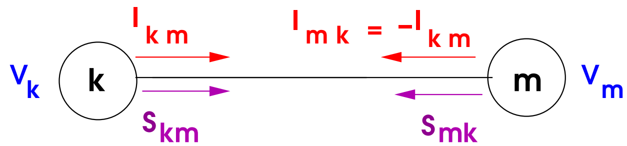
Steady-state (time average over one period of length $2\pi/\omega$): **voltage** at bus k represented as:
$= \hat{V}_k e^{j\theta_k} = \hat{V}_k(\cos\theta_k + j\sin\theta_k)$



- I_{km} = (complex) **current** injected into km at k
- S_{km} = (complex) **power** injected into km at k
- **Ohm's law:** $I_{km} = y_{km}(V_k - V_m)$ (y_{km} = admittance)
- $S_{km} = V_k I^*_{km}$

These are all **complex** quantities, but all are "real"

Steady-state (time average over one period of length $2\pi/\omega$): **voltage** at bus k represented as:
$= \hat{V}_k e^{j\theta_k} = \hat{V}_k(\cos\theta_k + j\sin\theta_k)$

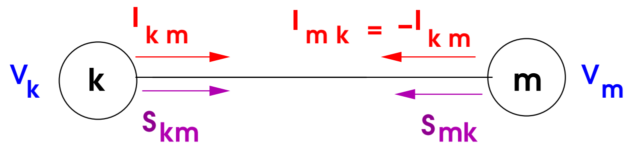


- I_{km} = (complex) **current** injected into km at k
- S_{km} = (complex) **power** injected into km at k
- **Ohm's law:** $I_{km} = y_{km}(V_k - V_m)$ (y_{km} = admittance)
- $S_{km} = V_k I^*_{km}$

These are all **complex** quantities, but all are "real"

- **Real** part of $S_{km} = P_{km} =$ "active" power
- **Imaginary** part of $S_{km} = Q_{km} =$ "reactive" power

Steady-state (time average over one period of length $2\pi/\omega$): **voltage** at bus k represented as:
$= \hat{V}_k e^{j\theta_k} = \hat{V}_k(\cos\theta_k + j\sin\theta_k)$



- I_{km} = (complex) **current** injected into km at k
- S_{km} = (complex) **power** injected into km at k
- **Ohm's law:** $I_{km} = y_{km}(V_k - V_m)$ (y_{km} = admittance)
- $S_{km} = V_k I^*_{km}$

These are all **complex** quantities, but all are "real"

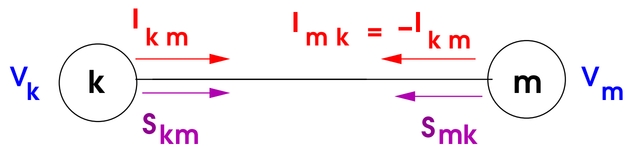
- **Real** part of $S_{km} = P_{km} =$ "active" power
- **Imaginary** part of $S_{km} = Q_{km} =$ "reactive" power
- If we write $V_k = e_k + j f_k$, then

$$P_{km} = (e_k - e_m)(g\,,\, b)\binom{e_k}{f_k} + (f_k - f_m)(-b\,,\, g)\binom{e_k}{f_k}.$$

(Here, $y_{km} = g + jb$), a **quadratic** expression on e_k, e_m, f_k, f_m.

- A similar quadratic yields Q_{km}

Steady-state (time average over one period of length $2\pi/\omega$): **voltage** at bus k represented as: $= \hat{V}_k e^{j\theta_k} = \hat{V}_k(\cos\theta_k + j\sin\theta_k)$



- I_{km} = (complex) **current** injected into km at k
- S_{km} = (complex) **power** injected into km at k
- **Ohm's law:** $I_{km} = y_{km}(V_k - V_m)$ (y_{km} = admittance)
- $S_{km} = V_k I^*_{km}$

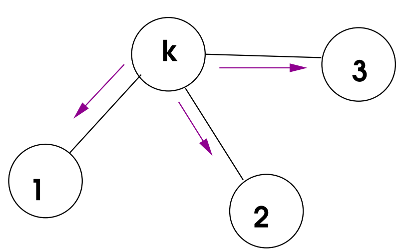
These are all **complex** quantities, but all are "real"

- **Real** part of $S_{km} = P_{km} =$ "active" power
- **Imaginary** part of $S_{km} = Q_{km} =$ "reactive" power
- If we write $V_k = e_k + j f_k$, then

$$P_{km} = (e_k - e_m)(g\,,\, b)\binom{e_k}{f_k} + (f_k - f_m)(-b\,,\, g)\binom{e_k}{f_k}.$$

(Here, $y_{km} = g + jb$), a **quadratic** expression on e_k, e_m, f_k, f_m.

- A similar quadratic yields Q_{km}
- What do we have at a given bus k?



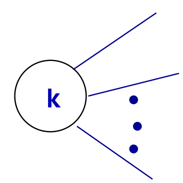
Putting it all together: power flow problem

$$V_k = \hat{V}_k e^{j\theta_k^V} = e_k + j f_k, \quad (1)$$

$$I_{km} = \boldsymbol{y_{\{k,m\}}}(V_k - V_m), \quad \boldsymbol{y_{\{k,m\}}} = \textit{admittance} \text{ of } km. \quad (2)$$

$$p_{km} = \mathcal{R}\mathrm{e}(V_k I_{km}^*), \quad q_{km} = Im(V_{km} I_{km}^*) \quad (3)$$

Network Equations



$$\sum_{km \in \delta(k)} p_{km} = \hat{P}_k, \quad \sum_{km \in \delta(k)} q_{km} = \hat{Q}_k \quad \forall\, k \quad (4)$$

Generator: $\hat{P}_k, |V_k| \ (= \hat{V}_k)$ given. Other buses: $\hat{P}_k, \hat{Q}_k$ given.

Problem. Compute a solution of this system of quadratic equations.

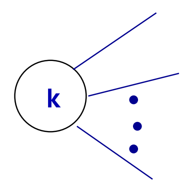
More general problem: ACOPF

$$V_k \;=\; \hat{V}_k e^{j\theta_k^V} \;=\; e_k + j f_k, \tag{5}$$

$$I_{km} \;=\; \boldsymbol{y_{\{k,m\}}}(V_k - V_m), \quad \boldsymbol{y_{\{k,m\}}} = \textit{admittance} \text{ of } km. \tag{6}$$

$$p_{km} \;=\; \mathcal{R}\text{e}(V_k I_{km}^*), \quad q_{km} \;=\; Im(V_{km} I_{km}^*) \tag{7}$$

Network Inequalities



$$\hat{P}_k^{\min} \le \sum_{km \in \delta(k)} p_{km} \le \hat{P}_k^{\max}, \quad \hat{Q}_k^{\min} \le \sum_{km \in \delta(k)} q_{km} \le \hat{Q}_k^{\max} \quad \forall\, k \tag{8}$$

$$\hat{V}_k^{\min} \le |V_k| \le \hat{V}_k^{\max} \quad \forall\, k \tag{9}$$

Problem

Solve an **optimization problem** subject to these quadratic **inequalities**.

How is ACOPF solved in industrial practice?

- **Best practice #1**:

How is ACOPF solved in industrial practice?

- **Best practice #1**: ~~Don't solve it and go for a beer instead~~

  Solve a **linearized** version.
  Why?
  Should be that: $|V_k| \approx 1$ for all k, so assume $|V_k| = 1$
  and: $\theta_k \approx \theta_m$, so $\sin(\theta_k - \theta_m) \rightarrow \theta_k - \theta_m$ and $\cos(\theta_k - \theta_m) \rightarrow 1$

How is ACOPF solved in industrial practice?

- **Best practice #1**: ~~Don't solve it and go for a beer instead~~

  Solve a **linearized** version.
  Why?
  Should be that: $|\boldsymbol{V_k}| \approx \mathbf{1}$ for all k, so assume $|\boldsymbol{V_k}| = \mathbf{1}$
  and: $\boldsymbol{\theta_k} \approx \boldsymbol{\theta_m}$, so $\mathbf{sin}(\boldsymbol{\theta_k} - \boldsymbol{\theta_m}) \rightarrow \boldsymbol{\theta_k} - \boldsymbol{\theta_m}$ and $\mathbf{cos}(\boldsymbol{\theta_k} - \boldsymbol{\theta_m}) \rightarrow \mathbf{1}$

- **Sequential linearization**. Replace all active constraints with their linearizations, and iterate.

How is ACOPF solved in industrial practice?

- **Best practice #1**: ~~Don't solve it and go for a beer instead~~

  Solve a **linearized** version.
  Why?
  Should be that: $|V_k| \approx 1$ for all k, so assume $|V_k| = 1$
  and: $\theta_k \approx \theta_m$, so $\sin(\theta_k - \theta_m) \rightarrow \theta_k - \theta_m$ and $\cos(\theta_k - \theta_m) \rightarrow 1$

- **Sequential linearization**. Replace all active constraints with their linearizations, and iterate.

- **IPOPT, et al**. Use interior point (e.g. barrier) methods to obtain a **locally optimal** solution.

→ But can we "certify" optimality?

→ But can we "certify" ***infeasibility***?

Quadratically constrained, quadratic programming problems (QCQPs):

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leq 0, \quad 1 \leq i \leq m \\ & x \in \mathbb{R}^n \end{array}$$

Here,

$$f_i(x) = x^T M_i x + c_i^T x + d_i$$

is a general quadratic, with M_i $n \times n$, wlog symmetric

Quadratically constrained, quadratic programming problems (QCQPs):

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \ \leq \ 0, \quad 1 \leq i \leq m \\ & x \in \mathbb{R}^n \end{array}$$

Here,

$$f_i(x) \ = \ x^T M_i x + c_i^T x + d_i$$

is a general quadratic, with M_i $n \times n$, wlog symmetric

- **Special case: Linear Programming**
  $M_i = 0, \quad 0 \leq i \leq m$

- **Special case: Convex Quadratic Programming:**
  $M_i \succeq 0, \quad 0 \leq i \leq m$

- **Unfortunately**, QCQP is NP-hard

Quadratically constrained, quadratic programming problems (QCQPs):

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \ \leq \ 0, \quad 1 \leq i \leq m \\ & x \in \mathbb{R}^n \end{aligned}$$

Here,

$$f_i(x) \ = \ x^T M_i x + c_i^T x + d_i$$

is a general quadratic, with M_i $n \times n$, wlog symmetric

- **Special case: Linear Programming**
  $M_i = 0, \quad 0 \leq i \leq m$

- **Special case: Convex Quadratic Programming:**
  $M_i \succeq 0, \quad 0 \leq i \leq m$

- **Unfortunately**, QCQP is NP-hard

  a deep fact: $x_j(1 - x_j) \ = \ 0$ is a quadratic constraint

OK, let's take a step waaaaay back: the trust-region (sub)problem

$$\begin{aligned} \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & \|x - \mu\|_2 \le r \end{aligned}$$

OK, let's take a step waaaaay back: the trust-region (sub)problem

$$\begin{aligned} \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & \|x - \mu\|_2 \le r \end{aligned}$$

- Control Theory
- Dynamical Systems
- Robust error estimation
- Robust optimization
- ~~Olympic swimming~~

OK, let's take a step waaaaay back: the trust-region (sub)problem

$$\begin{aligned} \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & \|x - \mu\|_2 \leq r \end{aligned}$$

- Control Theory
- Dynamical Systems
- Robust error estimation
- Robust optimization
- ...
- How about the **antitrust region (sub)problem**

$$\begin{aligned} \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & \|x - \mu\|_2 \geq r \end{aligned}$$

Digression: application of trust-region subproblem

→ Unconstrained optimization $\min\{f(x) : x \in \mathbb{R}^n\}$

Algorithm

- Given an iterate x^t, construct a **quadratic** "model" for $f(x)$ which is approximately valid in a neighborhood $\|x - x^t\| \leq \Delta$.

- For example, use

$$f(x^k) \ + \ \tfrac{1}{2}(x - x^t)^T H(x^t)(x - x^t)$$

  where $H(x^t)$ is the Hessian of f at x^t.

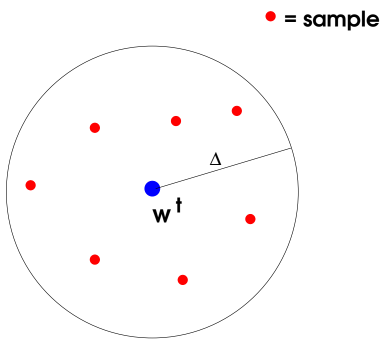
Digression: application of trust-region subproblem

$\rightarrow$ Unconstrained optimization $\min\{f(x) \,:\, x \in \mathbb{R}^n\}$

Algorithm

- Given an iterate x^t, construct a **quadratic** "model" for $f(x)$ which is approximately valid in a neighborhood $\|x - x^t\| \leq \Delta$.

- For example, get pairs $(y^1, f(y^1)), (y^2, f(y^2)), \ldots, (y^m, f(y^m))$



- Using these samples, construct an approximation to $f(x)$ (model = spline, least squares estimate, etc).

- Call this model: $Q(x)$

- **Solve:** $\min\{\, Q(x) \,:\, \|x - x^t\| \leq \Delta\}$. This is the trust-region subproblem.

- The solution becomes w^{t+1}.
  Or **(better)**: conduct a line-search from w^t to the solution so as to compute w^{t+1}.

- General purpose codes: **KNITRO**, **LOQO** have been used on OPF.

Summary

→ Unconstrained optimization $\min\{f(x) : x \in \mathbb{R}^n\}$

Algorithm

- Given an iterate x^t, construct a **quadratic** "model" for $f(x)$ which is approximately valid in a neighborhood $\|x - x^t\| \leq \Delta$.

- Call this model: $Q(x)$

- **Solve:** $\min\{ Q(x) : \|x - x^t\| \leq \Delta\}$.
  This is the trust-region subproblem.

- The solution becomes w^{t+1}.
  Or **(better)**: conduct a line-search from w^t to the solution so as to compute w^{t+1}.

- **What does this algorithm produce?**

- Does it solve the problem? Approximately?

How do we solve the trust region subproblem?

- Fast solution is crucial for the application
- This is a very mature problem that is considered well-solved
- Let us look at the problem from a broader perspective

Want to solve:

$$\begin{aligned} \boldsymbol{f*} \;=\; \min \quad f(x) &\doteq x^T A x + 2a^T x + a_0 \\ \text{s.t.} \quad g(x) &\doteq x^T B x + 2b^T x + b_0 \;\geq\; 0 \end{aligned}$$

Want to solve:

$$\begin{aligned} \boldsymbol{f*} \;=\; \min \quad f(x) &\doteq x^T A x + 2a^T x + a_0 \\ \text{s.t.} \quad g(x) &\doteq x^T B x + 2b^T x + b_0 \;\geq\; 0 \end{aligned}$$

Easier question:

Given a real $\boldsymbol{\theta}$, is it the the case that $\boldsymbol{f}^* \geq \boldsymbol{\theta}$?

Want to solve:

$$\boldsymbol{f*} \;=\; \min \quad f(x) \doteq x^T A x + 2a^T x + a_0$$
$$\text{s.t.} \quad g(x) \doteq x^T B x + 2b^T x + b_0 \;\geq\; 0$$

Easier question:

Given a real $\boldsymbol{\theta}$, is it the the case that $\boldsymbol{f^* \geq \theta}$?

Duality: true, iff there exists real $\boldsymbol{\gamma \geq 0}$ s.t. $\boldsymbol{f(x) - \theta - \gamma\, g(x) \geq 0 \quad \forall\, x}$

(this is MAGIC)

Want to solve:

$$\boldsymbol{f*} = \min \quad f(x) \doteq x^T Ax + 2a^T x + a_0$$
$$\text{s.t.} \quad g(x) \doteq x^T Bx + 2b^T x + b_0 \geq 0$$

Easier question:

Given a real $\boldsymbol{\theta}$, is it the the case that $\boldsymbol{f^* \geq \theta}$?

Duality: true, iff there exists real $\boldsymbol{\gamma \geq 0}$ s.t. $\boldsymbol{f(x) - \theta - \gamma\, g(x) \geq 0} \quad \forall\, \boldsymbol{x}$

(this is MAGIC) **i.e., iff** there exists $\boldsymbol{\gamma \geq 0}$ with:

$$(x^T, 1) \begin{pmatrix} A - \gamma B & a - \gamma b \\ (a - \gamma b)^T & a_0 - \gamma b_0 - \theta \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} \geq 0 \qquad \forall\, x$$

Want to solve:

$$\boldsymbol{f*} \; = \; \min \quad f(x) \doteq x^T A x + 2a^T x + a_0$$
$$\text{s.t.} \quad g(x) \doteq x^T B x + 2b^T x + b_0 \; \geq \; 0$$

Easier question:

Given a real $\boldsymbol{\theta}$, is it the the case that $\boldsymbol{f^* \geq \theta}$?

Duality: true, iff there exists real $\boldsymbol{\gamma \geq 0}$ s.t. $\boldsymbol{f(x) - \theta - \gamma\, g(x) \geq 0} \quad \forall\, \boldsymbol{x}$

(this is MAGIC) **i.e., iff** there exists $\boldsymbol{\gamma \geq 0}$ with:

$$(x^T, 1) \begin{pmatrix} A - \gamma B & a - \gamma b \\ (a - \gamma b)^T & a_0 - \gamma b_0 - \theta \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} \quad \geq \quad 0 \qquad \forall\, x$$

and it turns out that this is equivalent to:

$$\begin{pmatrix} A - \gamma B & a - \gamma b \\ (a - \gamma b)^T & a_0 - \gamma b_0 - \theta \end{pmatrix} \quad \succeq \quad 0 \qquad \text{(proof?)}$$

$$\boldsymbol{f*} \;=\; \min \quad f(x) \doteq x^T A x + 2a^T x + a_0$$
$$\text{s.t.} \quad g(x) \doteq x^T B x + 2b^T x + b_0 \;\geq\; 0$$

Rewrite it as:

$$\max \quad \theta$$
$$\text{s.t.} \quad f^* \;\geq\; \theta$$

Duality:

$$\mathbf{max}_{\boldsymbol{\theta},\boldsymbol{\gamma}}\; \theta$$

$$\text{s.t.} \quad \begin{pmatrix} A - \gamma B & a - \gamma b \\ & \\ (a - \gamma b)^T & a_0 - \gamma b_0 - \theta \end{pmatrix} \;\succeq\; 0$$

Back to general QCQP

$$
\begin{array}{rll}
\textbf{(QCQP):} & \min\ x^TQx\ +\ 2c^Tx & \\
\text{s.t.} & x^TA_ix\ +\ 2b_i^Tx\ +\ r_i\ \geq 0 & i = 1, \ldots, m \\
 & x \in \mathbb{R}^n. &
\end{array}
$$

Back to general QCQP

$$\begin{array}{lll} \textbf{(QCQP):} & \min \ x^T Q x \, + \, 2c^T x & \\ \text{s.t.} & x^T A_i x \, + \, 2b_i^T x \, + \, r_i \ \geq 0 & i = 1, \ldots, m \\ & x \in \mathbb{R}^n. & \end{array}$$

$\rightarrow$ form the **semidefinite relaxation**

$$\begin{array}{lll} \textbf{(SR):} & \min \ \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X & \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \ \geq 0 & i = 1, \ldots, m \\ & X \succeq 0, \quad X_{11} = 1. & \end{array}$$

Here, for symmetric matrices $\boldsymbol{M}$, $\boldsymbol{N}$,

$$M \bullet N \ = \ \sum_{h,k} M_{hk} N_{hk}$$

Why do we call it a relaxation?

Back to general QCQP

$$
\begin{array}{lll}
\textbf{(QCQP):} & \min \; x^T Q x \, + \, 2 c^T x & \\
\text{s.t.} & x^T A_i x \, + \, 2 b_i^T x \, + \, r_i \; \geq 0 & i = 1, \ldots, m \\
 & x \in \mathbb{R}^n. &
\end{array}
$$

$\rightarrow$ form the **semidefinite relaxation**

$$
\begin{array}{lll}
\textbf{(SR):} & \min \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X & \\
\text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \; \geq 0 & i = 1, \ldots, m \\
 & X \succeq 0, \quad X_{11} = 1. &
\end{array}
$$

Here, for symmetric matrices $\boldsymbol{M}$, $\boldsymbol{N}$,

$$
M \bullet N \; = \; \sum_{h,k} M_{hk} N_{hk}
$$

Why do we call it a relaxation?

Given $\boldsymbol{x}$ feasible for **QCQP**, the matrix $\boldsymbol{X} \; = \; (\mathbf{1}, \boldsymbol{x}^T) \begin{pmatrix} \mathbf{1} \\ \boldsymbol{x} \end{pmatrix}$ feasible for **SR** and with the same value

So the value of problem **SR** is a **lower bound** for **QCQP**

Back to general QCQP

$$\textbf{(QCQP):} \quad \begin{array}{ll} \min & x^T Q x \,+\, 2c^T x \\ \text{s.t.} & x^T A_i x \,+\, 2b_i^T x \,+\, r_i \;\geq 0 \qquad i = 1, \ldots, m \\ & x \in \mathbb{R}^n. \end{array}$$

→ form the **semidefinite relaxation**

$$\textbf{(SR):} \quad \begin{array}{ll} \min & \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \;\geq 0 \qquad i = 1, \ldots, m \\ & X \succeq 0, \quad X_{11} = 1. \end{array}$$

Here, for symmetric matrices $\boldsymbol{M}$, $\boldsymbol{N}$,

$$M \bullet N \;=\; \sum_{h,k} M_{hk} N_{hk}$$

Why do we call it a relaxation?

Given $\boldsymbol{x}$ feasible for **QCQP**, the matrix $\boldsymbol{X} \;=\; (\mathbf{1}, \boldsymbol{x}^T) \begin{pmatrix} \mathbf{1} \\ \boldsymbol{x} \end{pmatrix}$ feasible for **SR** and with the same value

So the value of problem **SR** is a **lower bound** for **QCQP**

But we need to go backwards: given a solution $\boldsymbol{X}$ to **SR**, does it give us a solution to **QCQP**?

Only if $\boldsymbol{X}$ has rank-1. Unfortunately, **SR** **typically does not** have a rank-1 solution.

It's pretty bad ...

Theorem (Pataki, 1998):

An SDP

$$\begin{aligned}
\textbf{(SR):} \quad & \min \ M \bullet X \\
\text{s.t.} \quad & N^i \bullet X \ \geq b_i \qquad i = 1, \ldots, m \\
& X \succeq 0, \quad X \text{ an } n \times n \text{ matrix,}
\end{aligned}$$

always has a solution of rank $\approx \boldsymbol{m^{1/2}}$, and this bound is attained.

Observation (Lavaei and Low):
The SDP relaxation of practical AC-OPF instances can have a rank-1 solution, or the solution can be relatively easy to massage into rank-1 solutions (also see earlier work of Bai et al)
Current research thrust: Can we leverage this observation into practical, globally optimal algorithms for AC-OPF?

I need to solve a complicated QCQP

$$\textbf{(QCQP):} \quad \begin{array}{lll} \min & x^T Q x + 2c^T x & \\ \text{s.t.} & x^T A_i x + 2b_i^T x + r_i \geq 0 & i = 1, \ldots, m \\ & x \in \mathbb{R}^n. & \end{array}$$

... what do I do?

I need to solve a complicated QCQP

$$
\begin{aligned}
\textbf{(QCQP):} \quad & \min \; x^T Q x \; + \; 2c^T x \\
\text{s.t.} \quad & x^T A_i x \; + \; 2b_i^T x \; + \; r_i \; \geq 0 \qquad i = 1, \ldots, m \\
& x \in \mathbb{R}^n.
\end{aligned}
$$

... what do I do? ~~run away~~

General techniques

- McCormick reformulation.
  Each $\boldsymbol{x_i x_j}$, where $\boldsymbol{x_i^L \leq x_i \leq x_i^U}$ and $\boldsymbol{x_j^L \leq x_j \leq x_j^U}$ is **replaced** by $\boldsymbol{X_{ij}}$ plus

$$
\begin{aligned}
X_{ij} & \geq x_i^L x_j + x_j^L x_i - x_i^L x_j^L \\
X_{ij} & \geq x_i^U x_j + x_j^U x_i - x_i^U x_j^U \\
X_{ij} & \leq x_i^U x_j + x_j^L x_i - x_i^U x_j^L \\
X_{ij} & \leq x_i^L x_j + x_j^U x_i - x_i^L x_j^U
\end{aligned}
$$

  Yields a **linear** programming relaxation

- Spatial branching, e.g. if $\boldsymbol{0 \leq x_j \leq 1}$ you branch as: $\boldsymbol{0 \leq x_j \leq 1/2}$ and $\boldsymbol{1/2 \leq x_j \leq 1}$.

- Widely implemented in many high-quality codes.

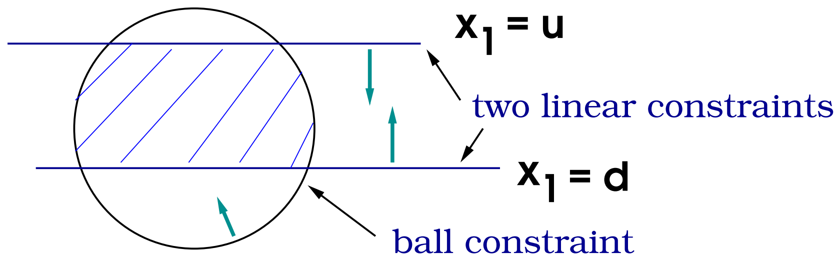
Let's take a computing break

A nice generalization of the trust-region subproblem

Solve a problem of the form

$$\begin{array}{rrcl} \min & x^T Q x + c^T x & & \\ \text{s.t.} & \|x\|_2 & \leq & 1 \\ & a_i^T x & \leq & b_i \quad i = 1, 2 \end{array}$$

provided the two **(two!)** linear constraints are parallel:

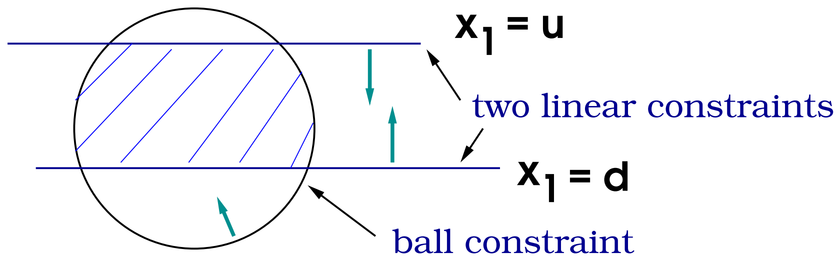


A nice generalization of the trust-region subproblem

Solve a problem of the form

$$\begin{aligned} \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & \|x\|_2 \le 1 \\ & a_i^T x \le b_i \quad i = 1, 2 \end{aligned}$$

provided the two **(two!)** linear constraints are parallel:



$$\rightarrow \min \{ x^T Q x + c^T x \, : \, d \le x_1 \le u, \, \|x\| \le 1 \}$$

$$\begin{aligned} \text{restate as:} \quad \min \quad & \sum_{i,j} q_{ij} X_{ij} + c^T x \\ \text{s.t.} \quad & X_{11} + du \le (d+u) x_1 \\ & \|X_{.1} - dx\| \le x_1 - d \\ & \|ux - X_{.1}\| \le u - x_1 \\ & \sum_j X_{jj} \le 1 \\ & X \succeq xx^T \end{aligned}$$

Lemma: This problem has an optimal solution with $X = xx^T$, i.e. a **rank-1** solution.

Many theoretically nice generalizations

- More than one ball constraint (but not too many) and more than one linear inequality (but not too many)
- A "small" number of general quadratic constraints
- The algorithms are theoretically efficient but computationally very challenging
- ~~I did some of this, so let's move on~~

Back to semidefinite relaxation

$$\begin{array}{lll} \textbf{(QCQP):} & \min \ x^T Q x \ + \ 2c^T x & \\ \text{s.t.} & x^T A_i x \ + \ 2b_i^T x \ + \ r_i \ \geq 0 & i = 1, \ldots, m \\ & x \in \mathbb{R}^n. & \end{array}$$

$\rightarrow$ form the **semidefinite relaxation**

$$\begin{array}{lll} \textbf{(SR):} & \min \ \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix} \bullet X & \\ \text{s.t.} & \begin{pmatrix} r_i & b_i^T \\ b_i & A^i \end{pmatrix} \bullet X \ \geq 0 & i = 1, \ldots, m \\ & X \succeq 0, \quad X_{11} = 1. & \end{array}$$

And let's make it worse. How about the **moment relaxation**?

Higher-order SDP relaxations

Consider the polynomial optimization problem

$$f_0^* \doteq \min\{ f_0(x) : f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} x^{\pi}$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^{\pi} \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$
- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

Higher-order SDP relaxations

Consider the polynomial optimization problem

$$f_0^* \doteq \min \{ f_0(x) \; : \; f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n \},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^\pi$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^\pi \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$
- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

Moment Relaxations

- Introduce a variable X_π used to represent each monomial x^π of order $\leq d$, for some integer d.
- This set of monomials includes all of those appearing in the polynomial optimization problem as well as $x^0 = 1$.

Higher-order SDP relaxations

Consider the polynomial optimization problem

$$f_0^* \doteq \min\{ f_0(x) : f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} x^\pi$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^\pi \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$

- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

Moment Relaxations

- Introduce a variable X_π used to represent each monomial x^π of order $\leq d$, for some integer d.
- This set of monomials includes all of those appearing in the polynomial optimization problem as well as $x^0 = 1$.
- If we replace each x^π in the formulation with the corresponding X_π we obtain a *linear* relaxation.

Higher-order SDP relaxations

Consider the polynomial optimization problem

$$f_0^* \doteq \min\{ f_0(x) \, : \, f_i(x) \geq 0, \quad 1 \leq i \leq m, \quad x \in \mathbb{R}^n\},$$

where each $f_i(x)$ is a **polynomial** i.e. $f_i(x) = \sum_{\pi \in S(i)} a_{i,\pi} \, x^\pi$.

- Each π is a tuple $\pi_1, \pi_2, \ldots, \pi_n$ of **nonnegative integers**, and $x^\pi \doteq x_1^{\pi_1} x_2^{\pi_2} \ldots x_n^{\pi_n}$
- Each $S(i)$ is a finite set of **tuples**, and the $a_{i,\pi}$ are reals.

Moment Relaxations

- Introduce a variable X_π used to represent each monomial x^π of order $\leq d$, for some integer d.
- This set of monomials includes all of those appearing in the polynomial optimization problem as well as $x^0 = 1$.
- If we replace each x^π in the formulation with the corresponding X_π we obtain a *linear* relaxation.
- Let X denote the vector of all such monomials. Then $X X^T \succeq 0$ and of rank one. The semidefinite constraint strengthens the formulation.
- Further semidefinite constraints are obtained from the constraints.

I need to solve a large nontrivial SDP

$$\begin{array}{llll} \textbf{(SDP):} & \min & F_0 \bullet X & \\ & \text{s.t.} & F_i \bullet X \geq b_i & i = 1, \ldots, m \\ & & X \succeq 0 & \end{array}$$

... what do I do?

I need to solve a large nontrivial SDP

$$
\begin{aligned}
\textbf{(SDP):} \quad & \min \; F_0 \bullet X \\
\text{s.t.} \quad & F_i \bullet X \; \geq b_i \qquad i = 1, \ldots, m \\
& X \succeq 0
\end{aligned}
$$

... what do I do? ~~run away even faster~~

Answer: use **structured sparsity**, if you can

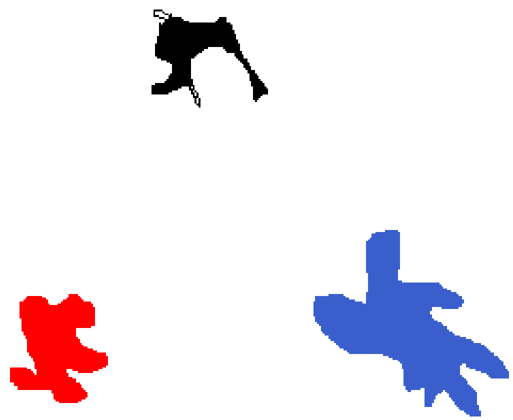
I need to solve a large nontrivial SDP

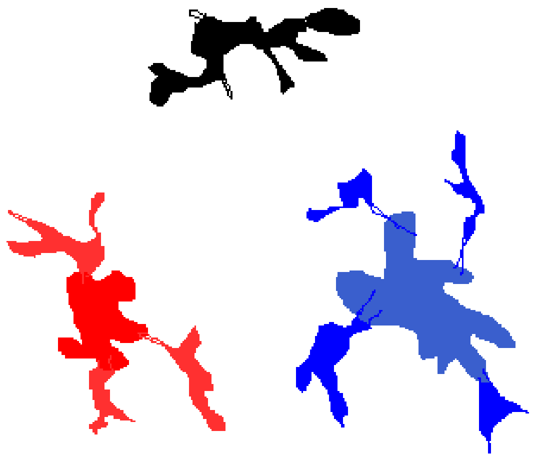
$$
\textbf{(SDP):} \quad \begin{array}{rll} \min & F_0 \bullet X & \\ \text{s.t.} & F_i \bullet X \ \geq b_i & \quad i = 1, \ldots, m \\ & X \succeq 0 & \end{array}
$$

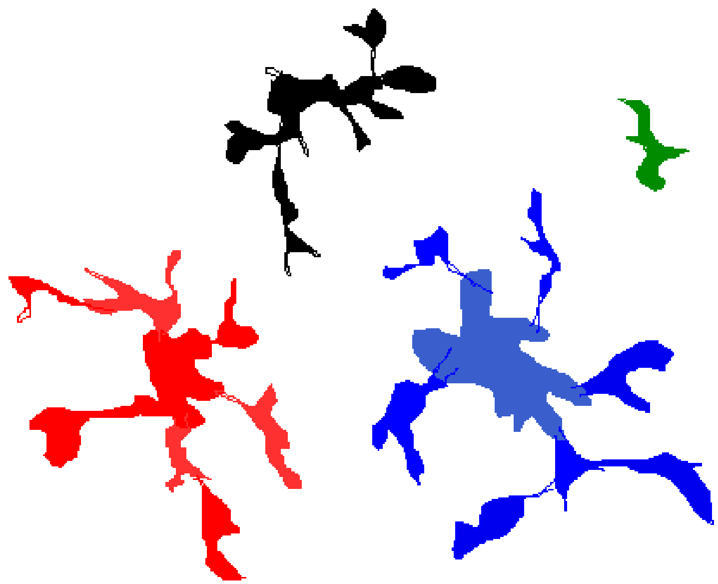
... what do I do? ~~run away even faster~~

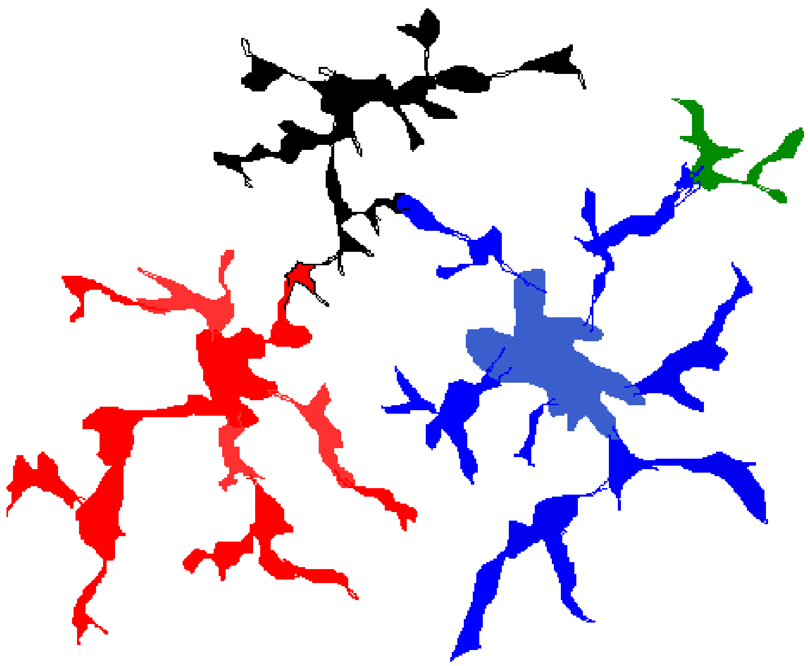
Answer: use **structured sparsity**, if you can

$\rightarrow$ How did power grids develop over time?





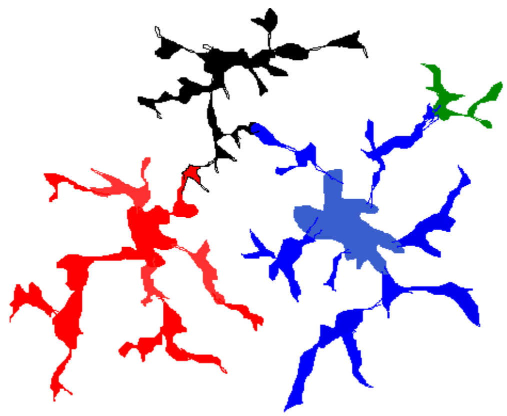




→ Modern grids are very sparse, and “tree-like”

Informal definition

A graph has small *treewidth* if it can be formed by glueing together small blobs (subnetworks) in a tree-like fashion.



- Modern grids have “small” tree-width

- SDP relaxations reflect this fact

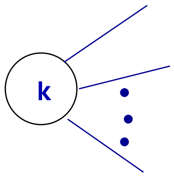
Back to ACOPF

$$V_k \;=\; \hat{V}_k e^{j\theta_k^V} \;=\; e_k + j f_k,$$

$$I_{km} \;=\; \boldsymbol{y_{\{k,m\}}}(V_k - V_m), \quad \boldsymbol{y_{\{k,m\}}} = \textit{admittance} \text{ of } km.$$

$$p_{km} \;=\; \mathcal{R}\mathrm{e}(V_k I_{km}^*), \quad q_{km} \;=\; Im(V_{km} I_{km}^*)$$
$$\hat{V}_k^{\min} \;\leq\; |V_k| \;\leq\; \hat{V}_k^{\max} \quad \forall\, k$$

Network Inequalities

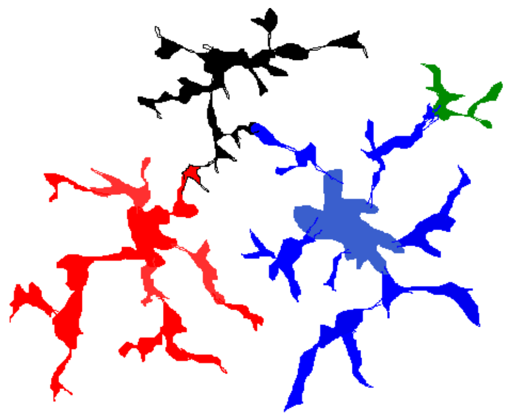


$$\hat{P}_k^{\min} \;\leq\; \sum_{km \in \delta(k)} p_{km} \;\leq\; \hat{P}_k^{\max} \qquad \forall\, k$$

$$\hat{Q}_k^{\min} \;\leq\; \sum_{km \in \delta(k)} q_{km} \;\leq\; \hat{Q}_k^{\max} \qquad \forall\, k$$

Informal definition

A graph has small *treewidth* if it can be formed by glueing together small blobs (subnetworks) in a tree-like fashion.



- Modern grids have "small" tree-width

- SDP relaxations reflect this fact

- SDP algorithms can leverage this fact

Crimes against computers

$$
\begin{aligned}
& \max \ y \\
\text{s.t.} \quad & 1000\,y + x \le 1000 && \text{(10a)} \\
& 10000\,\delta \ge 1 && \text{(10b)} \\
& \delta \le 10\,a && \text{(10c)} \\
& a \le 10\,b && \text{(10d)} \\
& b \le 10\,c && \text{(10e)} \\
& c \le 10\,d && \text{(10f)} \\
& d \le 10\,x && \text{(10g)} \\
& y \ \textbf{binary}, \text{ all other variables} \ge 0
\end{aligned}
$$

Crimes against computers

$$\begin{aligned}
& \max\ y \\
\text{s.t.} \quad & 1000\,y + x \le 1000 && \text{(11a)} \\
& 10000\,\delta \ge 1 && \text{(11b)} \\
& \delta \le 10\,a && \text{(11c)} \\
& a \le 10\,b && \text{(11d)} \\
& b \le 10\,c && \text{(11e)} \\
& c \le 10\,d && \text{(11f)} \\
& d \le 10\,x && \text{(11g)} \\
& y \textbf{ binary}, \text{ all other variables} \ge 0
\end{aligned}$$

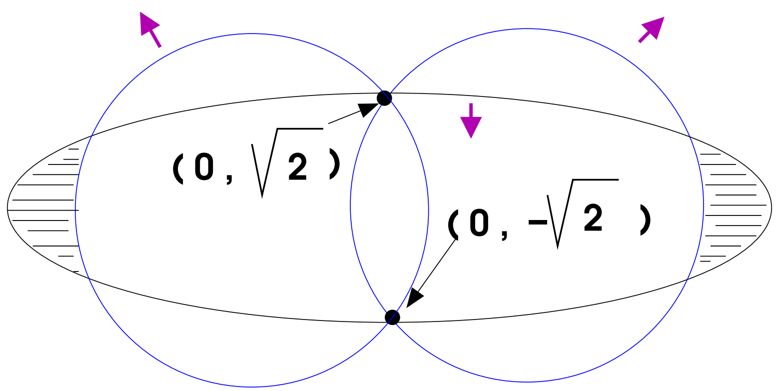
Value = 0

More crimes against computers

$$
\begin{aligned}
\max \quad & 20x_2 - 20s_5 - 20s_6 + 2s_7 + s_5^2 \\
\text{s.t.} \quad & (x_1 - 1)^2 + x_2^2 \geq 3 + \frac{\phi}{10} && \text{(12a)} \\
& (x_1 + 1)^2 + x_2^2 \geq 3 && \text{(12b)} \\
& \frac{1}{10}x_1^2 + x_2^2 \leq 2 && \text{(12c)} \\
& 10\,\delta + 10\,\phi^2 \geq 1 && \text{(12d)} \\
& -10\,a + \delta + 10\,\phi^2 \leq 0 \\
& -10\,b + a + 10\,\phi^2 \leq 0 \\
& -10\,c + b + 10\,\phi^2 \leq 0 \\
& -10\,d + c + 10\,\phi^2 \leq 0 \\
& -10\,e + d + 10\,\phi^2 + 10\,s_5^2 = 0 && \text{(12e)} \\
& -10\,f + e + 10\,\phi^2 + 10\,s_6^2 = 0 \\
& -10\,g + f + 10\,\phi^2 + 10\,s_7^2 = 0 \\
& -10\,\phi + g + 10\,\phi^2 \leq 0 && \text{(12f)}
\end{aligned}
$$

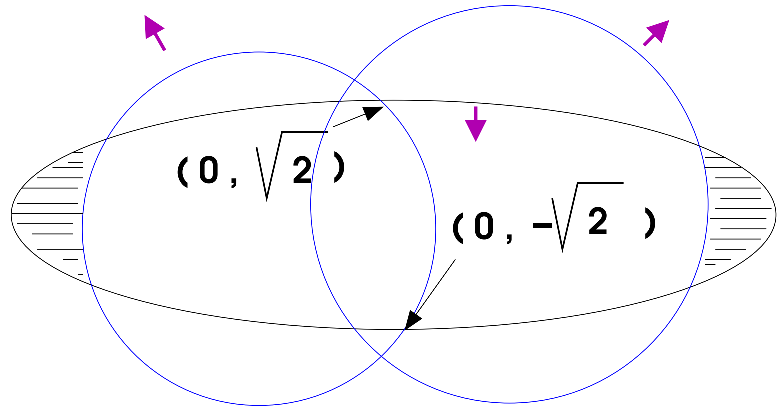
What's going on?

$$\begin{array}{rrcl} & \max & & x_2 \\ \text{s.t.} & (x_1-1)^2 & + & x_2^2 \geq 3 \\ & (x_1+1)^2 & + & x_2^2 \geq 3 \\ & & \dfrac{x_1^2}{10} + & x_2^2 \leq 2 \end{array}$$



What's going on?

$$\begin{aligned} \max \quad & x_2 \\ \text{s.t.} \quad (x_1 - 1)^2 \; + \; & x_2^2 \geq 3 + \phi \qquad (\phi > 0) \\ (x_1 + 1)^2 \; + \; & x_2^2 \geq 3 \\ \frac{x_1^2}{10} + & x_2^2 \leq 2 \end{aligned}$$



-> Thu.Aug.11.190441.2016@rabbitchaser