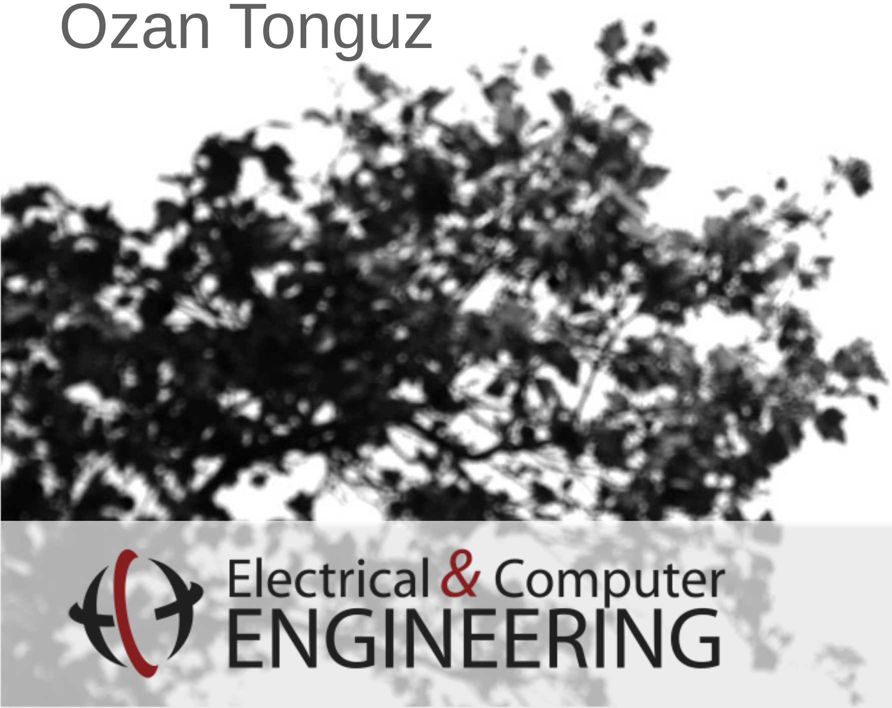
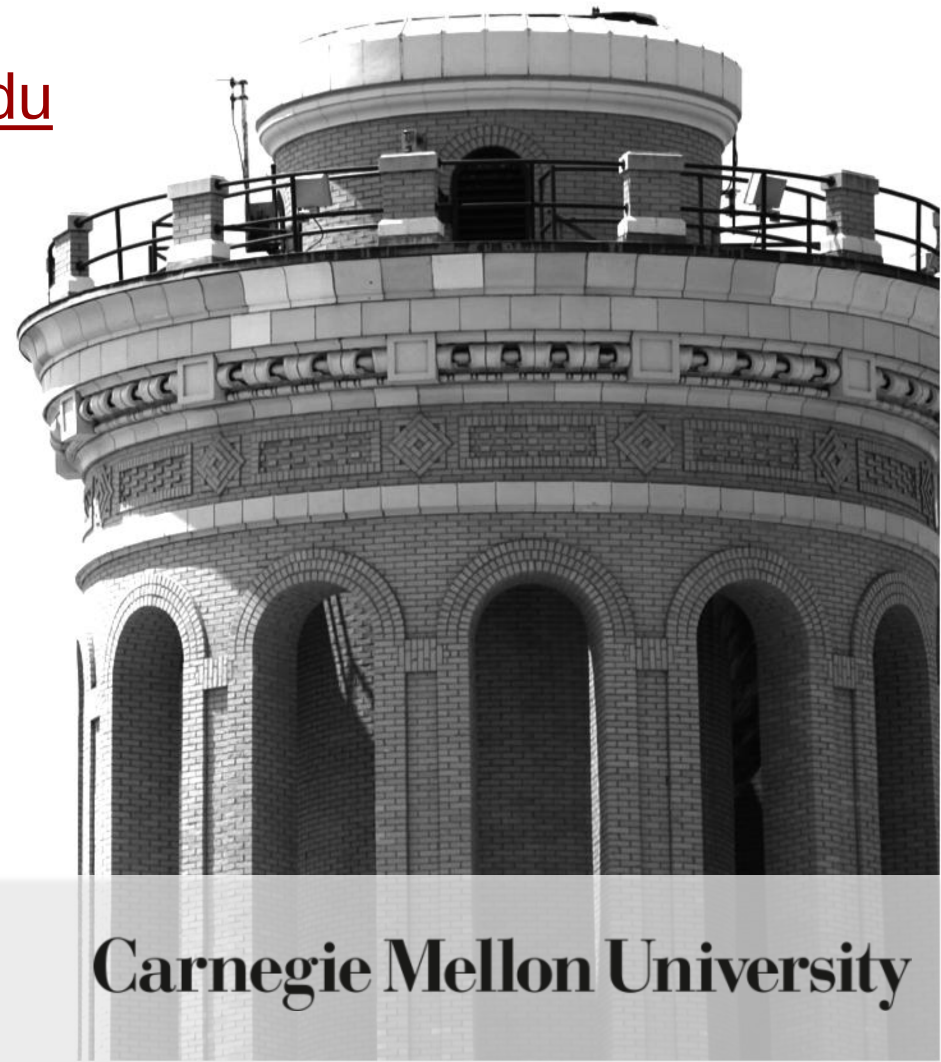
Distributed Optimization: How to Bridge the Gap between Theory and Application?

Gabriela Hug, ghug@ece.cmu.edu

partially in collaboration with
Soummya Kar and partially with
Ozan Tonguz





Electrical & Computer ENGINEERING

Carnegie Mellon University

Purpose of Talk

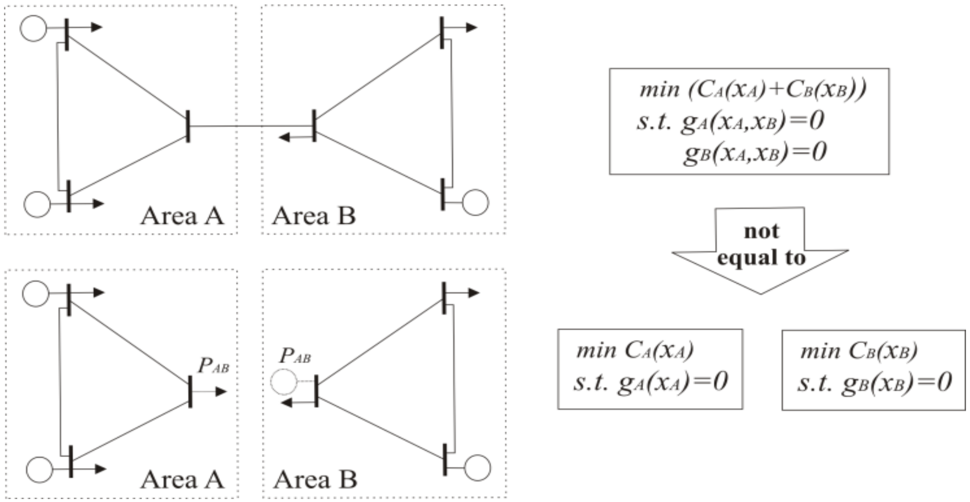
- Part I – “Soul Searching”
  - Initiate discussion on how gap between theoretic development of distributed algorithms and practical application in the “real world” can be closed
  - Discuss measures of performance for distributed algorithm beyond number of iterations
- Part II - Research
  - Introduce distributed DC Optimal Power Flow method

Introduction

- Motivation for Distributed Approaches
  - *Distributed Infrastructure:* Transition to distributed power plants and intelligent demand
  - *Computational Efficiency:* Stochastic / predictive optimization leads to computationally intensive calculations
  - *Communication:* Ubiquitous communication capabilities expected to be available
  - *Coordination:* Information sharing is crucial for enabling optimal coordination and monitoring across system

Introduction

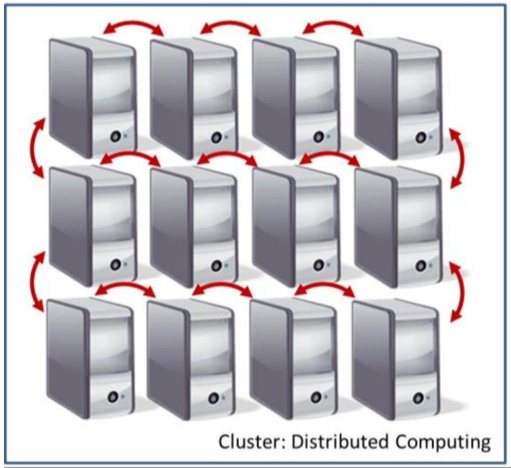
- Definition of "Distributed"
  - No need for centralized coordinator
  - Limited communication among distributed entities
  - Has to converge to same solution achieved by centralized optimization



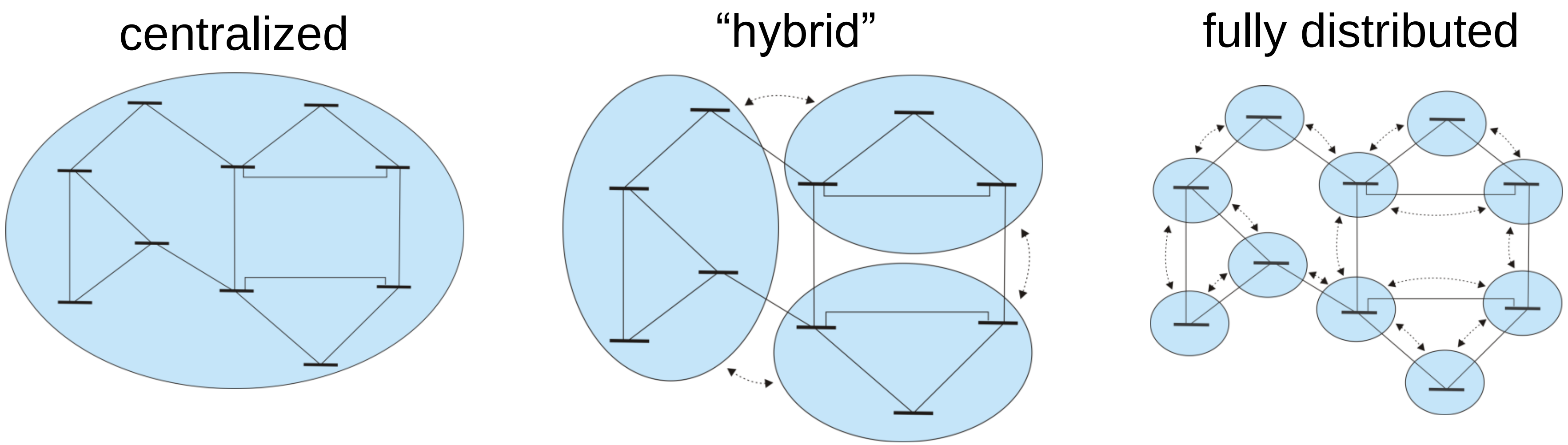
Introduction

- Advantages
  - improved computational efficiency to because of the possibility for parallel computations,
  - increased resilience against outages of computational entities,
  - opportunity to achieve system wide optimal performance with limited data exchange among controlling entities
- Disadvantages
  - Communication needs
  - Vulnerable to communication failures

vs.



Distributedness



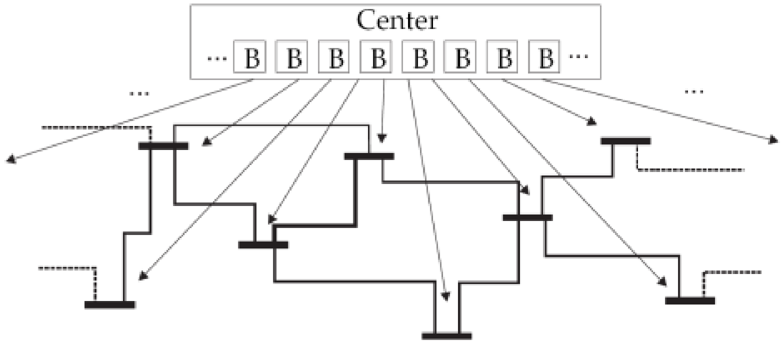
Increase in information exchange →

Increase in iterations until convergence →

← Increase in individual problem size

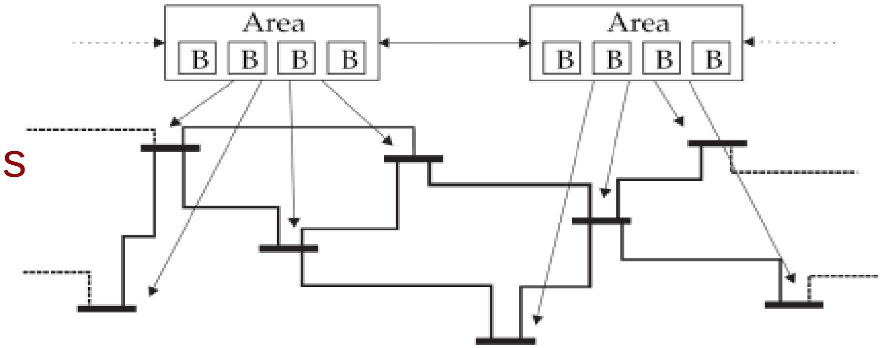
What is the optimal level of distributedness?

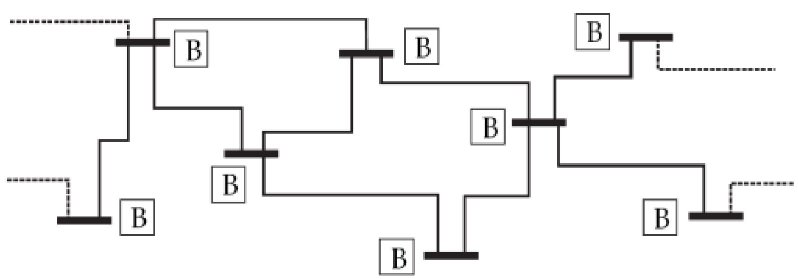
Implementation



Centralized Implementation
Goal: Computational Efficiency

Hybrid Implementation
Goal: Area Coordination plus Computational Efficiency

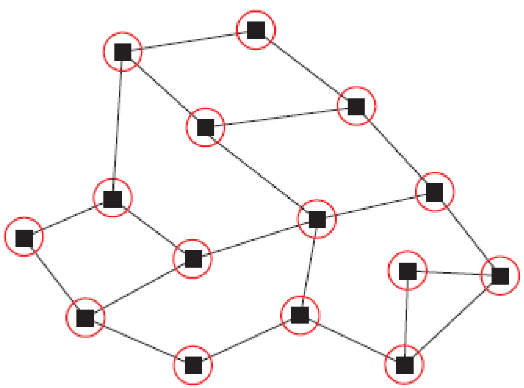




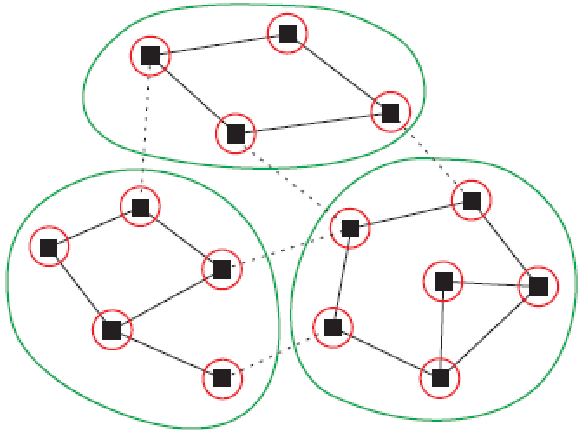
Fully Distributed Implementation
Goal: Resiliency plus Computational Efficiency

Implementation

- Synchronous vs. Asynchronous



Synchronous:
Less overall iterations
but communication after
every single iteration



Asynchronous:
More overall iterations but less
communication between “areas”

How to Benchmark Algorithms?

- Measures of Performance
  - Number of iterations => only useful if methods follow similar principles
  - Required data to be exchanged (amount and type)
  - Computation time
  - Robustness with respect to communication failures and with respect to computational entities
  - Computational complexity at each iteration
  - Accuracy of solution

How to weigh different objectives?
Should that be dependent on system state?

Barriers for Implementation

- Missing communication infrastructure?
- Trust in computational methods?
- Missing willingness to coordinate?
- History of centralized control/optimization?
- Vendor dominance in energy management systems?

Optimal Power Flow Problem

- Economic dispatch with line constraints

Minimize generation cost	$\Longleftrightarrow$	$\min \sum_{i=1}^{N_G} \left(a_i P_{G_i}^2 + b_i P_{G_i} + c_i \right)$
s.t. power balance equations	$\Longleftrightarrow$	$s.t.\quad P_{G_i} - P_{L_i} = \sum_{j \in \Omega_i} \frac{\theta_i - \theta_j}{X_{ij}} \qquad i = 1, \dots, N_G$
line flow constraints	$\Longleftrightarrow$	$-\overline{P}_{ij} \leq \frac{\theta_i - \theta_j}{X_{ij}} \leq \overline{P}_{ij} \qquad ij \in \Omega_L$
generation constraints	$\Longleftrightarrow$	$0 \leq P_{G_i} \leq \overline{P}_{G_i} \qquad i = 1, \dots, N_G$

$\Rightarrow$ derive first order optimality conditions resulting in system of equality and inequality constraints

$$\boxed{\begin{aligned} g(P, \theta, \lambda, \mu) &= 0 \\ h(P, \theta, \lambda, \mu) &\leq 0 \end{aligned}}$$

Distributed Optimal Power Flow

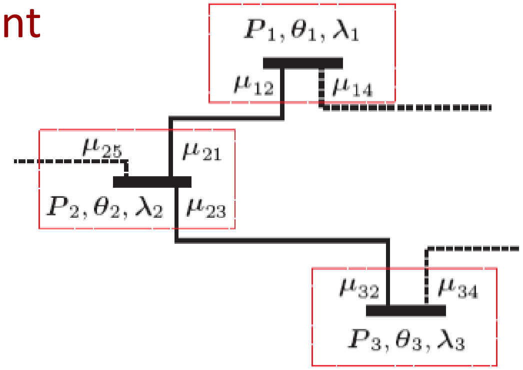
- Iterative procedure
  - update local variables

tuning parameter

$$x_i(k+1) = x_i(k) + \Phi^T \cdot g_i(x(k))$$

optimality constraint

$x_1 = [\theta_1, P_1, \lambda_1, \mu_{12}, \mu_{14}]$



  - project variables into feasible space
  - exchange updated values with neighbors
  - Locational Marginal Prices

$$\lambda_i(k+1) = \lambda_i(k) - \beta \cdot \left(\lambda_i(k) \sum_{j \in \Omega_i} \frac{1}{X_{ij}} - \sum_{j \in \Omega_i} \lambda_j(k) \frac{1}{X_{ij}} + \sum_{j \in \Omega_i} (\mu_{ij}(k) - \mu_{ji}(k)) \frac{1}{X_{ij}} \right)$$

$dL/d\theta_j$

$$-\alpha \cdot \left(\sum_{n \in \Omega_{G_i}} P_{G_n}(k) - P_{L_i} - \sum_{j \in \Omega_i} \frac{\theta_i(k) - \theta_j(k)}{X_{ij}} \right)$$

$dL/d\lambda_j$: power balance equation

Distributed Optimal Power Flow

- Power generation output

$$P_{G_n}(k+1) = \frac{\lambda_i(k+1) - b_n}{2a_n}$$

project into feasible space

- Bus angle update

$$\theta_i(k+1) = \theta_i(k) - \gamma\left(-\sum_{n\in\Omega_{G_i}} P_{G_n}(k) + P_{L_i} + \sum_{j\in\Omega_i}\frac{\theta_i(k)-\theta_j(k)}{X_{ij}}\right)$$

- Line flow Lagrange Multipliers

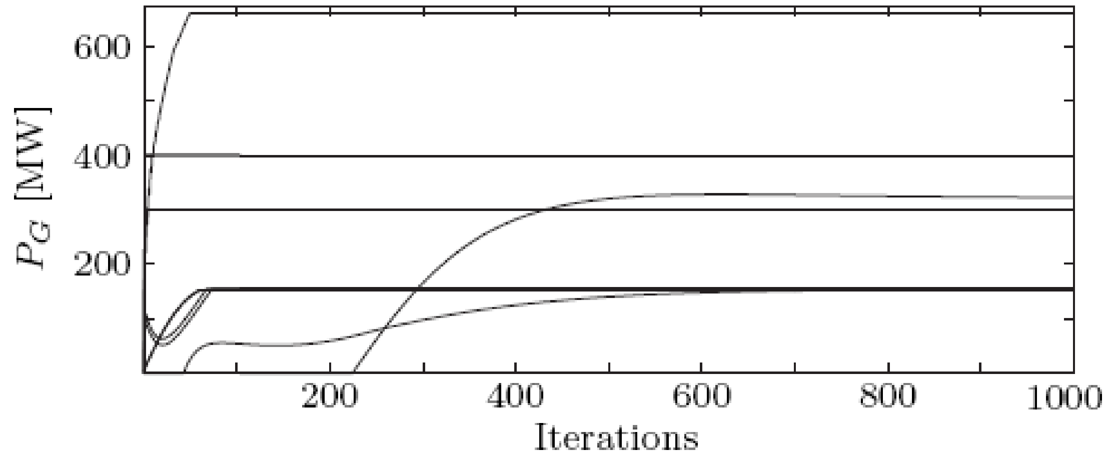
$$\mu_{ij}(k+1) = \mu_{ij}(k) - \delta\cdot\left(\overline{P}_{ij} - \frac{\theta_i(k)-\theta_j(k)}{X_{ij}}\right)$$

$$\mu_{ji}(k+1) = \mu_{ji}(k) - \delta\cdot\left(\overline{P}_{ij} + \frac{\theta_i(k)-\theta_j(k)}{X_{ij}}\right)$$

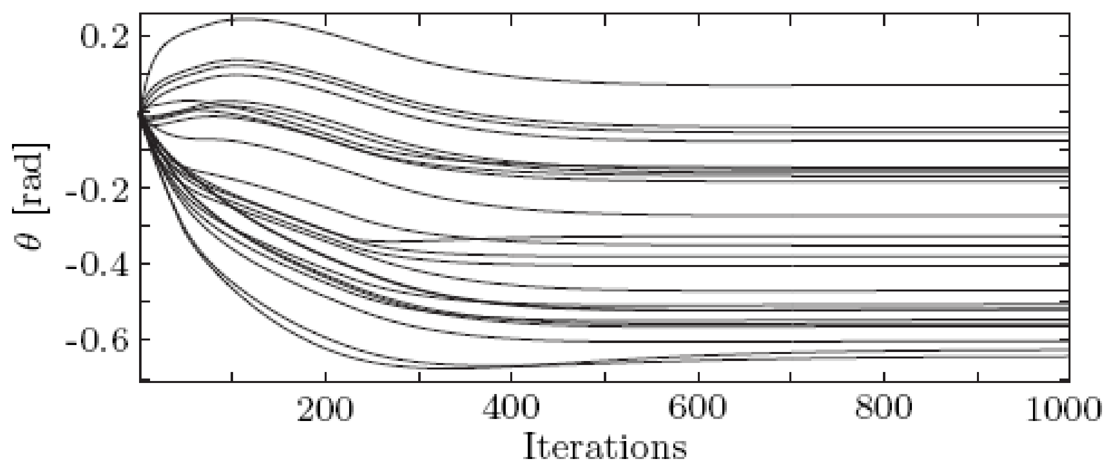
project into feasible space

Simulation Results

- Synchronous Update, uncongested IEEE RTS



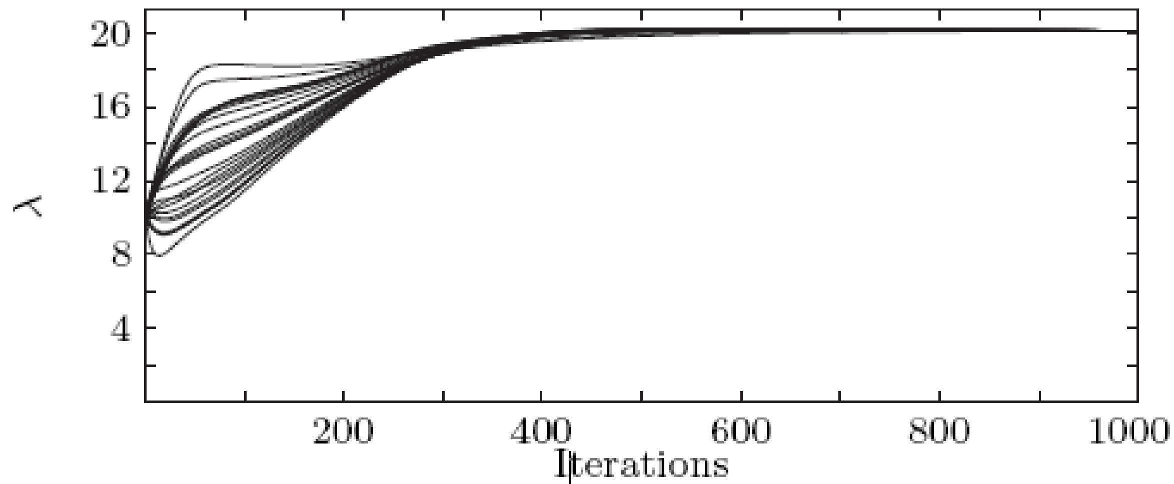
(a)



(b)

Simulation Results

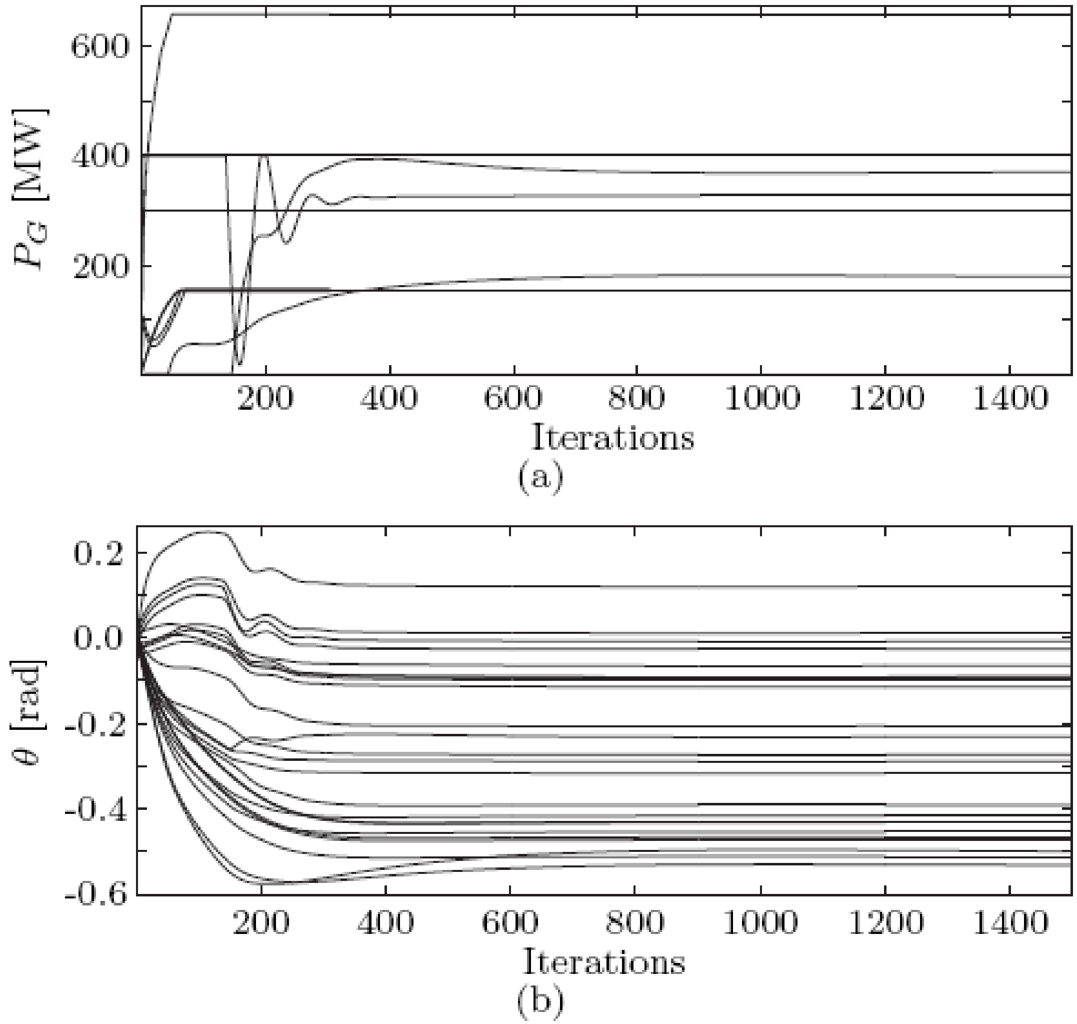
- Synchronous Update, uncongested IEEE RTS



(c)

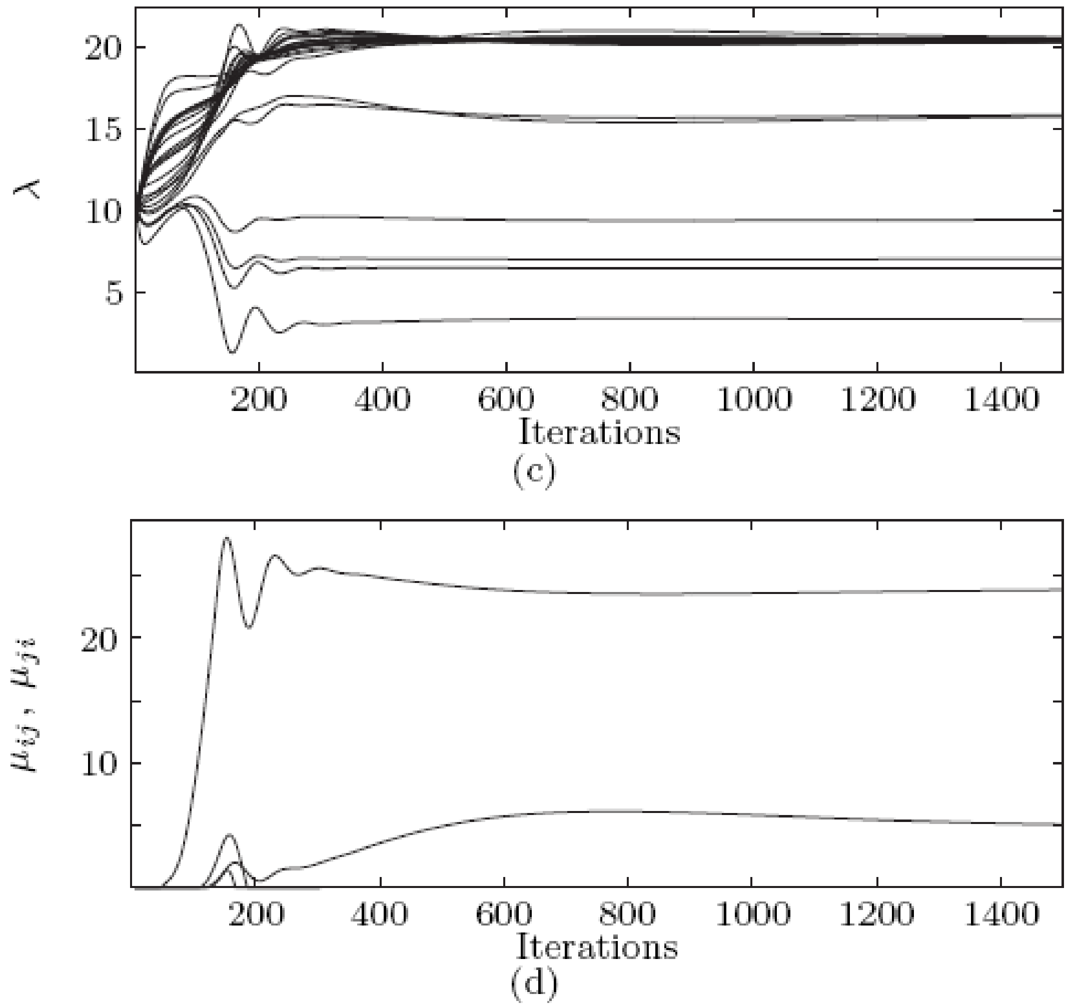
Simulation Results

- Synchronous Update, congested IEEE RTS



Simulation Results

- Synchronous Update, congested IEEE RTS



Conclusions

- Soul searching:
  - Still unclear how to close the gap between theory and implementation
  - Performance measures are necessary to benchmark algorithms => use these adaptively?
  - Trade-offs between performance measures
- Algorithm:
  - Innovation based method for distributed Optimal Power Flow
  - Very low complexity but comparably large number of iterations required