ECE 598: Computational Power Systems **Winter 2025**

Lecture 4: ADMM for grid optimization

Lecturer: Vladimir Dvorkin *Scribe(s): Yanyong Mao and Zhehua Xia*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

4.1 Penalized economic dispatch

- Economic dispatch (ED) problem review

$$\begin{aligned} \underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad & c(\mathbf{p}) \\ \text{subject to} \quad & \mathbf{1}^\top \mathbf{p} = d \quad : \nu \end{aligned}$$

  where $c : \mathbb{R}^n \to \mathbb{R}$ is a convex cost function, $p \in \mathbb{R}^n$ is generator dispatch.

- **Proposition**: Let's solve an unconstrained problem for some large value ν

$$\underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad c(\mathbf{p}) + \frac{\nu}{2} \|\mathbf{1}^\top \mathbf{p} - \mathbf{d}\|_2^2$$

  Pros: As $\nu \to \infty$, the penalize is really heavy. The solution approaches that of ED
  Cons: Difficult to choose the initial penalty value. The larger the penalty, the more difficult for algorithm to converge. The rate of convergence is proportional to Lipschitz constant $\nu\sqrt{n}$[1]

- We will work with Lagrange duality to find a better approach to solving ED

4.2 Dual ascent algorithm

- Partial Lagrangian function of ED

$$\max_{\nu} \min_{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}} \quad \mathcal{L}(\mathbf{p}, \nu) = c(\mathbf{p}) + \nu \left(d - \mathbf{1}^\top p \right)$$

- Finding the optimal solution of ED is the same as finding the *saddle point* of $\mathcal{L}$

$$\mathcal{L}(\mathbf{p}^*, \nu^*) = c(\mathbf{p}^*) + \nu^*(d - \mathbf{1}^\top \mathbf{p}^*) = c(\mathbf{p}^*)$$

- One way to optimize Lagrangian function is via dual ascent
- Consider a sequence of iterations k =1,...,K starting from $(\mathbf{p}^0, \nu^0)$

[1] $\nabla_p(\frac{\nu}{2}\|\mathbf{1}^\top \mathbf{p} = \mathbf{d}\|_2^2)$ grows linearly in $\boldsymbol{p}$, the Lipschitz constant $\nu\|\mathbf{1}\| \;= \nu\sqrt{n}$

- At every iteration k, the primal-dual update for some $\rho > 0$:

$$\mathbf{p}^{k+1} = \arg\min_{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}} \mathcal{L}(\mathbf{p}, \nu^k) \qquad \textit{primal update}$$

$$\nu^{k+1} = \nu^k + \rho(d - \mathbf{1}^\top \mathbf{p}^{k+1}) \qquad \textit{dual ascent}$$

- $\mathbf{p}$: power injection. $c(\mathbf{p})$: cost. ν: price. $\mathbf{p^{k+1}}$: minimize the total cost $c(\mathbf{p}^*) + \nu^*(d - \mathbf{1}^\top \mathbf{p}^*)$ under a given electricity price ν^k. ρ: sensitivity of dual variable to the violation of constraint.
- If the demand exceeds generation, the price will increased in the next iteration. Conversely, if the supply surpasses demand, the price will drop in the next iteration
- For certain problem classes, dual ascent yields efficient, convergent algorithms to an optimal primal-dual solution $(\mathbf{p}^*, \nu^*)$
- However, it may fail for some problems in power systems
- **Example**: in ED with linear cost $c(\mathbf{p}) = \mathbf{c}^\top \mathbf{p}$, the dual function

$$\phi(\nu) = \mathcal{L}(\mathbf{p}^*, \nu) = \begin{cases} d\nu, & \text{if} \quad \mathbf{c} - \mathbf{1}\nu = \mathbf{0} \\ -\infty, & \text{if otherwise} \end{cases}$$

  is unbounded without additional constraints $\Rightarrow$ no meaningful primal update

4.3 Augmented Lagrangian method

- Intuitively, the dual ascent fails to converge because the Lagrangian does not penalize the power balance constraint strongly enough
- Remedy: **Augmented Lagrangian**

$$\mathcal{L}_\rho(\mathbf{p}, \nu) = c(\mathbf{p}) + \nu(d - \mathbf{1}^\top \mathbf{p}) + \frac{\rho}{2}\|d - \mathbf{1}^\top \mathbf{p}\|_2^2$$

  where $\rho > 0$ is a penalty parameter
- The augmented Lagrangian can be regarded as the Lagrangian function for

$$\begin{aligned} \underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad & c(\mathbf{p}) + \frac{\rho}{2}\|d - \mathbf{1}^\top \mathbf{p}\|_2^2 \\ \text{subject to} \quad & \mathbf{1}^\top \mathbf{p} = d \quad : \nu \end{aligned}$$

- Even though the dispatch cost can be linear, the overall objective is quadratic
- Hence, the dual function is always bounded $\Rightarrow$ primal update $\mathbf{p}^{k+1}$ exists
- At every iteration k, the primal-dual update:

$$\mathbf{p}^{k+1} = \arg\min_{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}} \mathcal{L}_\rho(\mathbf{p}, \nu^k) \qquad \textit{primal update}$$

$$\nu^{k+1} = \nu^k + \rho(d - \mathbf{1}^\top \mathbf{p}^{k+1}) \qquad \textit{dual ascent}$$

- Difference from dual ascent algorithm: the Lagrangian function

- Working with augmented Lagrangian, we end up solving the original problem

$$\begin{aligned}\mathbf{0} &= \nabla_{\mathbf{p}} \mathcal{L}_\rho(\mathbf{p}^{k+1}, \nu^k) && \textit{primal update for augmented problem}\\ &= \nabla_{\mathbf{p}} c(\mathbf{p}^{k+1}) + \mathbf{1}\nu^k + \rho \mathbf{1}(d - \mathbf{1}^\top \mathbf{p}^{k+1})\\ &= \nabla_{\mathbf{p}} c(\mathbf{p}^{k+1}) + \mathbf{1}\nu^{k+1} = \nabla_{\mathbf{p}} \mathcal{L}(\mathbf{p}^{k+1}, \nu^{k+1}) && \textit{Lagrangian of the original problem}\end{aligned}$$

- Thus, $\mathbf{p}^{k+1}$ minimizes the Lagrangian function of the original problem

4.4 Alternating direction method of multipliers (ADMM)

4.4.1 Motivation of ADMM

The augmented Lagrangian method has limitations:

1. It is computationally heavy because it has to solve the optimization problem for every update.
2. It requires a centralized controller to exchange information between generators, which is hard to achieve in reality due to a lot of reasons such as privacy and scalability.

4.4.2 ADMM Framework

ADMM addresses the above issues by splitting the original problem into sub-problems that are easier to solve. The basic idea is to perform a series of updates on each primal variable and then update the dual variable:

1. **Update p_1:** Fix ν and $p_2, \ldots, p_n$; update:

$$p_1^{k+1} = \arg \min_{\underline{p}_1 \leqslant p_1 \leqslant \overline{p}_1} \mathcal{L}_\rho(p_1, p_2^k, \ldots, p_n^k, \nu^k)$$

2. **Update p_i $(i = 2, \ldots, n)$:** With ν and all other $p_j (j \neq i)$ fixed, update:

$$p_i^{k+1} = \arg \min_{\underline{p}_i \leqslant p_i \leqslant \overline{p}_i} \mathcal{L}_\rho(p_1^{k+1}, \ldots, p_i, \ldots, p_n^k, \nu^k)$$

3. **Dual Update:** After updating all primal variables, update the dual variable ν by:

$$\nu^{k+1} = \nu^k + \rho \left(d - \sum_{i=1}^n p_i^{k+1} \right)$$

4.4.3 Closed-Form Solution for Sub-Problems

For each sub-problem, the update for p_i can be formulated as

$$p_i^{k+1} = \arg \min_{\underline{p}_i \leqslant p_i \leqslant \overline{p}_i} \left(c_i(p_i) - \nu^k p_i + \frac{\rho}{2} \|\Delta p_i^k - p_i\|_2^2 \right)$$

where

$$\Delta p_i^k = d - \sum_{j=1}^{i-1} p_j^{k+1} - \sum_{j=i+1}^{n} p_j^k$$

which is the power imbalance between demand and all other generators' generation.

The Lagrangian optimality condition gives that

$$\nabla_{p_i} c_i(p_i) - \nu^k - \rho(\Delta p_i^k - p_i) = 0$$

The ADMM method can be very efficient because these sub-problems have closed-form solutions due to the fixation of all other variables:

1. For **quadratic cost function**:

   $$c_i(p_i) = c_0 + c_{1,i} p_i + c_{2,i} p_i^2$$

   the solution is

   $$p_i^{k+1} = \frac{\nu^k + \rho\, \Delta p_i^k - c_{1,i}}{2c_{2,i} + \rho}$$

   followed by a projection onto $[\underline{p}_i, \overline{p}_i]$

2. For **linear cost function**:

   $$c_i(p_i) = c_0 + c_{1i} p_i$$

   the solution becomes

   $$p_i^{k+1} = \Delta p_i^k - \frac{c_{1i} + \nu^k}{\rho}$$

   again followed by the necessary projection.

- **C**losed-form solutions exist when the cost functions have a low-order structure (e.g., quadratic or linear) and the constraints are not complex (such as box constraints here), so that the derivative equations can be solved directly.
- **T**his avoids the need for repeating optimization within each ADMM step, resulting in significant computational savings, especially important in real-time grid dispatch scenarios.

4.4.4 Exchange ADMM for Parallel Computations

In the sequential ADMM, updates are performed one after another. The Exchange ADMM achieves parallelization by:

1. Calculate the total power mismatch at iteration k:

   $$\Delta p^k = d - \sum_{i=1}^{n} p_i^k$$

2. Update any p_i in parallel:

   $$p_i^{k+1} = \arg \min_{\underline{p}_i \leqslant p_i \leqslant \overline{p}_i} \left(c_i(p_i) - \nu^k p_i + \frac{\rho}{2} \| p_i - (p_i^k - \Delta p^k) \|^2 \right)$$

3. Update the dual variable:

$$\nu^{k+1} = \nu^k + \rho\, \Delta p^k$$

This approach eliminates the sequential limitation, allowing each generator to update its decision in parallel. Hence, it realizes partial decentralization because the only thing each generator owner need to do to begin the alternating direction iteration is to publish their electricity price, instead of doing a system wide optimization. However, a central entity is still required to gather price information and demand mismatch and update dual variable.

4.4.5 Consensus ADMM for Fully Decentralized Coordination

When coordinating power flows among different regions (or areas), a fully decentralized approach is desirable. Tp achieve this goal, consensus ADMM is proposed:

Regional Optimization Problem Formulation

The global optimization problem is formulated as:

$$\begin{aligned} \underset{\mathbf{p},\mathbf{f}}{\text{minimize}} \quad & c(\mathbf{p}) \\ \text{subject to} \quad & \mathbf{p} - \mathbf{d} = \mathbf{A}\mathbf{f} \\ & \underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}} \\ & \underline{\mathbf{f}} \leqslant \mathbf{f} \leqslant \overline{\mathbf{f}} \end{aligned}$$

where the adjacency matrix $\mathbf{A}$ is defined by

$$a_{ij} = \begin{cases} +1, & \text{if node } i \text{ exports flow } j \\ -1, & \text{if node } i \text{ imports flow } j \\ 0, & \text{otherwise} \end{cases}$$

Inter-regional Consensus Optimization Formulation

The consensus ADMM formulation duplicates the power flow variables to enforce agreement (consensus) among neighboring areas:

$$\begin{aligned} \underset{\mathbf{p}\in\mathcal{P},\mathbf{f}\in\mathcal{F}}{\text{minimize}} \quad & \sum_{i=1}^{n} c_i(p_i) & \\ \text{subject to} \quad & p_i - d_i = \mathbf{a}_{i*}^{\top}\mathbf{f}_i, & \forall i \\ & \mathbf{f}_{i*}^{i} - \mathbf{f}_{*i}^{*} = \mathbf{0} : \boldsymbol{\nu}_i, & \forall i \\ & \underline{\mathbf{f}} \leqslant \mathbf{f} \leqslant \overline{\mathbf{f}} & \end{aligned}$$

The consensus ADMM can be solved by iterating between these two problems:

1. **Local Update (Private Variables):** For each area i,

$$\begin{aligned} \mathbf{f}_i^{k+1} = \arg\min_{p_i,\mathbf{f}_i} \quad & \left(c_i(p_i) + {\boldsymbol{\nu}_i^k}^{\top}\mathbf{f}_i + \frac{\rho}{2}\|\mathbf{f}_i - \mathbf{f}^k\|^2 \right) \\ \text{subject to} \quad & p_i - d_i = \mathbf{a}_i^{\top}\mathbf{f}_i \\ & \underline{p}_i \leqslant p \leqslant \overline{p}_i \end{aligned}$$

2. **Consensus Variable Update:**

$$\begin{aligned}\mathbf{f}^{k+1} = \arg\min_{\mathbf{f}} \quad & \left(-\boldsymbol{\nu}_i^{k\top} \mathbf{f} + \sum_{i=1}^{n} \frac{\rho}{2} \|\mathbf{f}_i^{k+1} - \mathbf{f}\| \right) \\ \text{subject to} \quad & \underline{\mathbf{f}} \leqslant \mathbf{f} \leqslant \overline{\mathbf{f}}\end{aligned}$$

3. **Dual Update:**

$$\boldsymbol{\nu}_i^{k+1} = \boldsymbol{\nu}_i^{k} + \rho(\mathbf{f}_i^{k+1} - \mathbf{f}^{k+1})$$

This method requires communication only among neighboring regions, so the central coordinator is no more needed. And the iteration process can be seen as the negotiating between neighboring regions. As demand changes, these neighbors will coordinate by observing the price and adjacent flows to reach a power balance and market equilibrium.

4.4.6 Conclusion

ADMM provides an effective way to overcome the computational and communication challenges in centralized optimization methods. The closed-form solutions for sub-problems allows simple and quick updating by avoiding costly iterative optimization steps. The Exchange ADMM enables parallel computations, while the Consensus ADMM enables fully decentralized coordination. All of these characteristics make this methods suitable for large-scale and privacy-concerned applications.