2018 IEEE Conference on Decision and Control (CDC)
Miami Beach, FL, USA, Dec. 17-19, 2018

Direct Data-Driven Methods for Risk Limiting Dispatch

Junjie Qin, Kameshwar Poolla and Pravin Varaiya

Abstract— **In the classical risk limiting dispatch (RLD) formulation, the system operator dispatches generators relying on information about the distribution of demand. In practice, such information is not readily available and therefore is estimated using historical demand and auxiliary information (or features) such as weather forecasts. In this paper, instead of using a separated estimation and optimization procedure, we propose learning methods that directly compute the RLD decision rule based on historical data. Using tools from statistical learning theory, we then develop generalization bounds and sample complexity results of the proposed methods. These algorithms and performance guarantees, developed for the single-bus network, are then extended to a general network setting for the uniform reserve case.**

I. Introduction

With the deepening penetration of renewable energy resources, it becomes increasingly important for electric power system operators to manage the uncertainty associated with these energy resources. This requires modeling the uncertainty using historical data and controlling resources in a way that balances economic costs and system risks. In practice, this process has been treated in two separate steps:

- *estimation* or *learning* step that identifies a model of the uncertain parameters, and
- *optimization* or *control* step that uses the uncertainty model to select the dispatch decisions by optimizing certain criteria subject to power network constraints.

The merits of this *separated estimation and optimization* procedure include the fact that it leads to tractable formulations and efficient algorithms for each subproblem. However this procedure may result in inefficient dispatch decision because of potential inconsistencies between these two steps. In particular, without using the actual cost information for the system, the estimation step cannot weight different errors properly (e.g. forecast errors that lead to insufficient generation v.s. excessive generation); without knowing how the estimation step obtains the model of uncertainty, a control procedure that takes the model as given can only perform well with respect to the model, without guarantees with respect to the actual system that generates the data used to produce the model.

Meanwhile, with the recent data deluge and interest in applying machine learning and artificial intelligence techniques for engineering problems, a question of debate is whether these learning techniques are capable of controlling the most complex man-made machine, namely, the electric power grid [1], [2]. A basic step in answering this question is to understand how learning methods can be rigorously applied in the context of controlling engineering systems, in a way that gives *end-to-end* performance guarantees for the resulting control decisions based on attributes of the initial dataset. To this end, we will need to merge the estimation and optimization steps, and directly obtain the decisions from the data.

In this paper, we propose such a direct data-driven method for dispatching generators in a power network under uncertainty. In particular, we consider the problem of *risk limiting dispatch* (RLD), a stochastic control formulation that optimally balances the economic costs and system risks. The classical RLD formulation focuses on the optimization step, *assuming* a model of the uncertainty. Thus we first extend the formulation to a data-driven setting, where the system operator is given the initial data that are used to obtain the uncertainty model instead of the model itself. This dataset contains historical net demand (demand less renewable generation) and auxiliary information (a.k.a. features or covariates) such as weather forecasts. We then propose learning algorithms for the data-driven RLD problem that directly map the data into the optimal dispatch decisions. This is done by modifying existing learning algorithms (including regularized linear regression, kernel regression and neural networks) and can be implemented in practice. Utilizing recent finite-sample tools from statistical learning theory, we develop end-to-end performance guarantees for using these learning algorithms to solve data-driven RLD. For ease of exposition, we focus most of our development on a single-bus network setup without power network constraints. The single-bus results are then extended to general network settings under the restriction that the reserve levels are identical across the network.

A. Contribution and paper organization

This paper contributes to the literature in the following ways. First, it formulates the data-driven RLD as a learning problem (Section II). Second, it introduces a systematic procedure for modifying standard learning algorithms to solve the data-driven RLD problem (Section III). Third, it develops end-to-end performance guarantees for the proposed procedure, establishing that the resulting learning algorithms are *probably approximated correct* (PAC) with respect to the RLD costs by bounding their sample complexities (Section IV). Finally, these results are extended to the setting with general power network for the case of uniform reserve, providing efficient and rigorous approaches for dispatching generators under uncertainty and power network constraints (Section V).

J. Qin, K. Poolla and P. Varaiya are with the Department of Electrical Engineering and Computer Science, University of California at Berkeley {qinj, poolla, varaiya}@berkeley.edu

978-1-5386-1395-5/18/$31.00 ©2018 IEEE

B. Literature

Many papers address either the estimation step or the optimization step. For the estimation step, see [3], [4] and references therein on load forecasting, and [5], [6], [7] and references therein on wind and solar forecasting. For the optimization step, different methods have been proposed based on different types of uncertainty models produced in the estimation step. For example, see [8] for model predictive control, [9] for stochastic programming based economic dispatch, [10] for robust optimization based approaches.

Risk limiting dispatch [11] is a stochastic optimal control formulation for the optimization step. It leads to efficient and easy-to-implement dispatch rules that balances generation costs and the operation risk of potential loss of load. It has been generalized to incorporate multiple forward markets [12], stochastic prices [13], storage [14], and network constraints [15]. All prior work on this topic has assumed the knowledge of the probability distributions of the demand.

Our performance guarantees are related to but different from those of methods based on sample average approximation (SAA), cf. [16] and [17]. Performance guarantees for SAA are usually asymptotic, while we obtain bounds for the performance of our algorithms that holds with a finite number of sample points.

II. Formulation

A. Notations

For an Euclidean space $\mathbb{R}^n$, we use $\mathbf{1} \in \mathbb{R}^n$ to denote the all-one vector. Given arbitrary sets $\mathcal{X}$ and $\mathcal{Y}$, we use $\mathcal{Y}^{\mathcal{X}}$ to denote the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$.

B. Classical risk limiting dispatch

In the classical risk limiting dispatch setting [11], the system operator dispatches generation one day ahead to meet an unknown net demand defined to be the actual demand less the renewable generation. As the day-ahead forecast errors for renewable generation are usually substantial, explicitly accounting for the uncertainty in the dispatch process is necessary.

Mathematically, given the probability distribution of net demand $D \sim \mathbb{P}_D$ supported on $[D^{\min}, D^{\max}]$, the classical risk limiting dispatch solves

$$u^\star(\mathbb{P}_D) = \operatorname*{argmin}_{u \in \mathcal{U}} \ \mathbb{E}_{D\sim\mathbb{P}_D}[c(u, D)], \tag{1}$$

where the function

$$c(u, d) = \alpha u + \beta(d - u)_+$$

captures the cost of scheduling generation in the day-ahead market αu and the operational risk of not having enough generation to cover the real-time net demand $\beta(d - u)_+$. The constant coefficient $\alpha > 0$ models the average price of purchasing from day-ahead market, and $\beta > \alpha$ models the value of loss load in real time[1]. For simplicity, we assume $\mathcal{U} = \mathbb{R}$.

[1]Parameter β may also be interpreted as the price for purchasing power from fast ramping generators in real time.

In practice, $\mathbb{P}_D$ is not known a priori and needs to be learned from data. Thus in a *separated estimation and optimization* (SEO) paradigm, one first estimates a distribution $\widehat{\mathbb{P}}_D$ from data, and then solves (1) with $\widehat{\mathbb{P}}_D$ in place of the actual but unknown distribution $\mathbb{P}_D$. While the solution $u^\star(\widehat{\mathbb{P}}_D)$ is the optimal dispatch for demand distribution $\widehat{\mathbb{P}}_D$, it is in general suboptimal for $\mathbb{P}_D$. Furthermore, it is unclear in SEO how to gauge the true performance of the dispatch, i.e.,

$$\mathbb{E}_{D\sim\mathbb{P}_D}\left[c\left(u^\star(\widehat{\mathbb{P}}_D), D\right)\right],$$

as $\mathbb{P}_D$ will be different from $\widehat{\mathbb{P}}_D$ when the sample size is limited.

C. Data-driven risk limiting dispatch

In the data-driven setting, we aim to directly determine the dispatch with the given dataset. For this purpose, we consider a rather general and practical setup, where we are given observations of the net demand over historical hours, together with other relevant data that we refer to as features. For instance, for a historical hour i, we may have records of the net demand D_i and a vector of features X_i with its entries recording forecasts of the temperature, wind speed, and other relevant information about hour i, as well as some nonlinear transformations of certain features. As the features and the net demand are generally correlated, these features could contain useful information about the net demand. We emphasize that for the purpose of our dispatch problem, the feature X_i is available at the time when dispatch decision regarding D_i is made.

Denote the historical data set by

$$S_n = \{Z_1 = (X_1, D_1), \ldots, Z_n = (X_n, D_n)\},$$

where $Z_i \in \mathcal{Z} := \mathcal{X} \times \mathcal{D}$ with $\mathcal{X} \subset \mathbb{R}^p$ with p being the number of features, and $\mathcal{D} = [D^{\min}, D^{\max}]$. We also refer to S_n as the *training set*, because it is the dataset that is used to "train" the dispatch decision rule.

We are also given features of the delivery hour for which we are making a dispatch decision. We denote these features by $X_* \in \mathbb{R}^p$. In the learning theory terminology, X_* is referred to as the *test inputs/features*. The net demand D_* for the delivery hour is not known at the time of dispatch.

In the data-driven risk limiting dispatch problem, we aim to identify a dispatch rule h, a.k.a. *hypothesis* in learning theory, that is a mapping from $\mathcal{X}$ to $\mathcal{U}$ using the historical data set S_n that minimizes the RLD cost:

$$h^\star = \operatorname*{argmin}_{h \in \mathcal{H}} \ \mathbb{E}_{D_*\sim\mathbb{P}_D}\left[c(h(X_*), D_*)\right], \tag{2}$$

where $\mathcal{H} \subset \mathcal{U}^{\mathcal{X}}$ is the *hypothesis class*, i.e, a subset of all functions from $\mathcal{X}$ to $\mathcal{U}$ in which we are searching for a good dispatch rule. The resulting dispatch is then $u_* = h^\star(X_*)$.

Following assumptions are in force for the rest of the paper.

A1 S_n contains i.i.d. samples from an unknown distribution $\mathbb{P}_Z$.

A2 $\|X_i\|_\infty \le X^{\max}$, $i \in \{1, \ldots, n, *\}$.

Assumption **A**1 can be relieved but not completely removed. In particular, if process $\{Z_i : i = 1, \ldots, n\}$ exhibits periodic structures corresponding to e.g. different times of the day, similar treatment in this paper may be carried out on samples grouped according to the periodic cycles. Fundamentally we need a form of stationarity so that past data indeed contains distributional information for the future. Assumption **A**1 is the simplest such assumption. Assumption **A**2 is made without loss of practicality as all features are bounded physical quantities in our application.

Problem (2) is challenging for two reasons. First, since the objective function involves taking expectation with respect to an unknown distribution for which we only have a finite sample, it is unrealistic to hope for identifying the exact minimizer of (2). We thus seek to identify an *approximate* solution h such that, for some small $\epsilon > 0$,

$$\mathbb{E}_{D_*}\left[c(h(X_*), D_*)\right] \leq \mathbb{E}_{D_*}\left[c(h^\star(X_*), D_*)\right] + \epsilon. \quad (3)$$

Furthermore, since our solution h depends on dataset S_n which is a random sample of size n from distribution $\mathbb{P}_Z$, there is always a chance of getting a bad sample that does not represent the population $\mathbb{P}_Z$. Thus any solution can only be *probably* correct so that there will be a small probability $\delta > 0$ of failure (i.e., resulting in high costs). In summary, an efficient algorithm for (2) identifies a hypothesis h that is *probably approximately correct* (PAC), i.e. h satisfies (3) with a large probability $1 - \delta$.

III. Learning algorithms

A learning algorithm determines the dispatch rule h (i.e., a map from features in the testing hour to the amount of generation to dispatch for that hour) based on the given training set S_n. In this section, we will provide specialized learning algorithms for the data-driven risk limiting dispatch problem. Just like when these learning algorithms are applied to common machine learning tasks, these specialized learning algorithms are efficient as they rely only on solving simple convex programs (except neural networks) and are easy-to-implement. We will also show in Section IV that these algorithms have nice theoretical performance guarantees as they are PAC.

In statistical learning theory, the principled approach to tackling problems similar to (2) is through *empirical risk minimization* (ERM). The idea is to replace the population mean in (2) with the empirical mean, and then solve the following optimization

$$\widehat{h} = \operatorname*{argmin}_{h \in \mathcal{H}} \ \frac{1}{n}\sum_{i=1}^{n} c(h(X_i), D_i). \quad (4)$$

Although all algorithms that we introduce in this section take the form of (4), their implementation can be rather different due to differences in the hypothesis classes. Compared to their standard learning counterparts, the algorithms below use a specialized cost function that arises from the RLD application. Therefore, these algorithms can also be implemented by simply customizing existing learning solvers.

1) ℓ_2 regularized linear regression (ERM-L2): Consider linear hypotheses with weights that have bounded ℓ_2 norm: $\mathcal{H}^{\mathrm{L2}} = \{x \mapsto w^\top x : w \in \mathbb{R}^p,\ \|w\|_2 \leq W_2^{\max}\}$. Without the norm constraint on weights, this is equivalent to linear regression with RLD cost in places of the sum of squared residuals. The norm constraint on weights serves as a form a regularization, which is especially useful when the number of features p is large. The resulting optimization is

$$\begin{aligned} \min_{w \in \mathbb{R}^p} \quad & \frac{1}{n}\sum_{i=1}^{n} c(w^\top X_i, D_i) \\ \text{s.t.} \quad & \|w\|_2 \leq W_2^{\max}. \end{aligned}$$

2) ℓ_1 regularized linear regression (ERM-L1): Consider linear hypotheses with weights that have bounded ℓ_1 norm: $\mathcal{H}^{\mathrm{L1}} = \{x \mapsto w^\top x : w \in \mathbb{R}^p,\ \|w\|_1 \leq W_1^{\max}\}$. This is the same as ERM-L2 except the ℓ_1 norm bound encourages sparsity in the weights therefore it performs a form of automatic *feature selection*.

3) Kernel method (ERM-K): For a Reproducing Kernel Hilbert Space (RKHS)[2] $\bar{\mathcal{H}}$ with its associated kernel function $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ that is bounded $\sup_{x,x' \in \mathcal{X}} \sqrt{k(x,x')} \leq K^{\max}$, we consider hypothesis class $\mathcal{H}^{\mathrm{K}} = \{h \in \bar{\mathcal{H}} : \|h\|_{\bar{\mathcal{H}}} \leq V^{\max}\}$. Let $K \in \mathbb{R}^{n \times n}$ be the kernel matrix defined for the training set S_n, i.e., $K_{ij} = k(X_i, X_j)$. Then problem (4) with $\mathcal{H} = \mathcal{H}^{\mathrm{K}}$ takes the form of

$$\begin{aligned} \min_{v \in \mathbb{R}^n} \quad & \frac{1}{n}\sum_{i=1}^{n} c\left(\sum_{j=1}^{n} v_j K_{ij}, D_i\right) \\ \text{s.t.} \quad & v^\top K v \leq \left(V^{\max}\right)^2. \end{aligned}$$

Given the solution $v^\star$, the resulting dispatch is $u = K_*^\top v^\star$, where $K_* \in \mathbb{R}^n$ is defined as $(K_*)_i = k(X_*, X_i)$.

4) ℓ_2 regularized neural networks (ERM-NN2): Consider densely connected deep neural networks [19] with m layers and d_j units in the jth layer. Such neural networks can represent functions of the form

$$\begin{aligned} & h_W(x) \\ & = \sigma^{(m)}\left(W^{(m)}\sigma^{(m-1)}\left(\ldots\left(W^{(2)}\sigma^{(1)}\left(W^{(1)}x\right)\right)\ldots\right)\right), \end{aligned}$$

where $W = \{W^{(j)}_{j \in [m]}\}$, $W^{(j)} \in \mathbb{R}^{d_j \times d_{j-1}}$, $j = 1, \ldots, m$, with $d_0 := p$ and $d_m := 1$, and $\sigma^{(j)} : \mathbb{R}^{d_j} \mapsto \mathbb{R}^{d_j}$ is defined as

$$\sigma^{(j)}\left(\begin{bmatrix} y_{11} & \ldots & y_{1d_j} \\ \vdots & \ddots & \vdots \\ y_{d_j 1} & \ldots & y_{d_j d_j} \end{bmatrix}\right) = \begin{bmatrix} \sigma(y_{11}) & \ldots & \sigma(y_{1d_j}) \\ \vdots & \ddots & \vdots \\ \sigma(y_{d_j 1}) & \ldots & \sigma(y_{d_j d_j}) \end{bmatrix},$$

with σ being a 1-Lipschitz activation function such that $\sigma(0) = 0$ (e.g. $\sigma(y) = \tanh(y)$ and $\sigma(y) = \max\{y, 0\}$, i.e., the rectified linear unit (ReLU)). We consider hypothesis class $\mathcal{H}^{\mathrm{NN2}} = \{x \mapsto h_W(x) : \|W_r^{(j)}\|_2 \leq W_2^{\max},\ j = 1, \ldots, m,\ r = 1, \ldots, d_j\}$, where $\left(W_r^{(j)}\right)^\top$ is the rth row

[2] See [18] for a nice review of kernel methods.

of matrix $W^{(j)}$. The optimization for finding the best neural network coefficients takes the form of

$$\min_{W} \quad \frac{1}{n}\sum_{i=1}^{n} c(h_W(X_i), D_i)$$
$$\text{s.t.} \quad \|W_r^{(j)}\|_2 \le W_2^{\max},\ j=1,\dots,m,\ r=1,\dots,d_j.$$

This optimization is nonconvex but empirical success has been observed in using stochastic gradient descent for problems of this form [20].

5) ℓ_1 regularized neural networks (ERM-NN1): We consider the same model as in ERM-NN2, except that the ℓ_2 norm constraints are replaced by ℓ_1 norm constraints to promote sparsity in the coefficients: $\mathcal{H}^{\mathrm{NN1}} = \{x \mapsto h_W(x) : \|W_r^{(j)}\|_1 \le W_1^{\max},\ j=1,\dots,m,\ r=1,\dots,d_j\}$.

IV. Performance

A. Preliminaries

We start by introducing basic notations and concepts in learning theory that are essential for developing our analysis of the learning algorithms' performance.

Given a data point $z=(x,d)$ and a hypothesis h, the *loss function* is defined as

$$\ell(h,z) = c(h(x),d),$$

which characterizes the cost of the dispatch rule h for a given realization of feature x and demand d in our setting. The main quantity that we are interested in is the *generalization cost* of an algorithm A acting on dataset S, defined as

$$L(A,S) = \mathbb{E}_Z\left[\ell(A_S, Z)\right],$$

where the expectation is taken over the population distribution of Z, $\mathbb{P}_Z$. Also of interest is the *empirical cost*, defined as

$$\widehat{L}(A,S) = \frac{1}{n}\sum_{i=1}^{n}\ell(A_S, Z_i),$$

for dataset S containing Z_i, $i=1,\dots,n$. When the training dataset S is clear from the context, we will simply write $L(A)$ and $\widehat{L}(A)$. When the algorithm is clear from context or is not of concern, we also denote the generalization cost and empirical cost of a given hypothesis h by, respectively,

$$L(h) = \mathbb{E}_Z\left[\ell(h,Z)\right] \quad \text{and} \quad \widehat{L}(h) = \frac{1}{n}\sum_{i=1}^{n}\ell(h,Z_i).$$

Most of our theoretical analysis centers around the notion of *excess cost* of a hypothesis $h = A_S$, which is the additional generalization cost of h compared to the optimal one:

$$L(h) - L(h^\star).$$

Bounds on this quantity control the suboptimality of h. Such bounds, as mentioned in Section II, cannot hold with probability one. Thus we hope to show that the proposed algorithms are probably approximately correct, defined as follows.

Definition 1 (PAC): A learning algorithm is probably approximately correct with parameters $\delta \in (0,1)$ and $\epsilon > 0$ if the hypothesis produced by it, denoted by h, satisfies

$$L(h) - L(h^\star) \le \epsilon,$$

with probability at least $1-\delta$.

B. Main results

We now provide the performance guarantees for the learning algorithms proposed in Section III, showing that they are PAC. The tools for establishing these results are introduced in Section IV-C. For convenience, we denote $C^{\max} = \max\{\alpha, \beta-\alpha\}$, which is the (globally) Lipschitz coefficient for the mapping $h \mapsto \ell(\cdot,h)$ in the RLD loss function $\ell(z,h) = \alpha h(x) + \beta(d-h(x))_+$.

Theorem 1 (PAC for ERM): Every algorithm introduced in Section III is PAC, with any $\delta \in (0,1)$ and ϵ no larger than the corresponding bound in Table I.

TABLE I
Excess cost bounds for ERM algorithms

ERM	Bound
L2	$\frac{4C^{\max}X^{\max}W_2^{\max}\sqrt{p}}{\sqrt{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}$
L1	$\frac{4C^{\max}X^{\max}W_1^{\max}\sqrt{2\log(2p)}}{\sqrt{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}$
K	$\frac{4C^{\max}V^{\max}K^{\max}}{\sqrt{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}$
NN2	$\prod_{j=1}^{m}\sqrt{d_j}\frac{2^{m+1}(W_2^{\max})^m C^{\max}X^{\max}\sqrt{p}}{\sqrt{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}$
NN1	$\frac{2^{m+1}(W_1^{\max})^m C^{\max}X^{\max}\sqrt{2\log(2p)}}{\sqrt{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}$

Many observations can be made regarding Theorem 1. Denote the excess cost bound in Theorem 1 for each algorithm by $\overline{\epsilon}(n,\delta,p)$. The first observation is that for any fixed p and δ (even if δ is arbitrarily close to 0),

$$\lim_{n\to\infty}\overline{\epsilon}(n,\delta,p) = 0.$$

We thus have the following corollary.

Corollary 1: Every algorithm introduced in Section III is asymptotically optimal.

Furthermore, for all algorithms except ERM-K, we have $\overline{\epsilon}(n,\delta,p)$ increasing with the number of features p as it requires more samples to fit a richer model with more parameters to achieve the same level of accuracy. Dependence on p is not explicit in the bound for the kernel method as in kernel method the underlying feature space may be infinite dimensional, whose richness is controlled by parameters $K^{\max}$ and $V^{\max}$.

Finally, we can invert the mapping $\overline{\epsilon}(n,\delta,p)$ to find the number of samples required for each algorithm to achieve a desired level of accuracy (δ,ϵ). We summarize these *sample complexity* results in the following corollary.

Corollary 2: With the number of samples listed in Table II, algorithms introduced in Section III have excess costs no larger than $\epsilon > 0$ with probability at least $1-\delta \in (0,1)$.

We note that although with the big-O notation linear regression and neural networks have the same expression for

TABLE II
SAMPLE COMPLEXITIES FOR ERM ALGORITHMS

ERM	Sample complexity
L2, NN2	$O\left(\frac{p+\log(1/\delta)}{\epsilon^2}\right)$
L1, NN1	$O\left(\frac{\log p+\log(1/\delta)}{\epsilon^2}\right)$
K	$O\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$

sample complexity, the hidden constants for these methods are significantly different when the neural networks are deep.

C. Uniform convergence

The classical approach to establishing bounds for the excess cost is called *uniform convergence*. That is, if for any hypothesis $h \in \mathcal{H}$ (including the hypothesis induced by an algorithm A and the optimal hypothesis $h^\star$) we can bound the right-hand-side quantity of

$$L(h) - \widehat{L}(h) \leq \sup_{h\in\mathcal{H}} \left(L(h) - \widehat{L}(h)\right) = G,$$

then we can bound the excess cost of ERM $\widehat{h}$ by

$$\begin{aligned}\mathbb{P}\left[L(\widehat{h}) - \widehat{L}(h^\star) \geq \epsilon\right] &\leq \mathbb{P}\left[\sup_{h\in\mathcal{H}} \left|L(h) - \widehat{L}(h)\right| \geq \frac{\epsilon}{2}\right] \\ &\leq \mathbb{P}\left[G \geq \frac{\epsilon}{2}\right] + \mathbb{P}\left[G' \geq \frac{\epsilon}{2}\right],\end{aligned}$$

where G' is defined analogously for the negative loss $\ell' = -\ell$.

The following quantity plays a key role in bounding G for a wide range of loss functions and hypothesis classes.

Definition 2 (Rademacher complexity): Let $\mathcal{F}$ be a class of real-valued functions $f : \mathcal{Z} \mapsto \mathbb{R}$. The Rademacher complexity of $\mathcal{F}$ is defined as

$$R_n(\mathcal{F}) = \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \xi_i f(Z_i)\right], \tag{5}$$

where $Z_1, \ldots, Z_n$ are i.i.d. samples from $\mathbb{P}_Z$, and $\xi_1, \ldots \xi_n$ are drawn i.i.d. from the uniform distribution over $\{-1, +1\}$.

Remark 1 (Complexity measure): Rademacher complexity is a measure of the richness of a function class with respect to a data generating probability distribution. It is closely related with other well-known notions of complexity such as Vapnik-Chervonenkis dimension and covering numbers [21]. Its properties allow us to give a unified treatment of many distinct learning algorithms.

We will develop most of our results based on the following key theorem [22] from statistical learning theory that bounds the excess cost of ERM using the Rademacher complexity of the *loss class*, which is the composition of the loss function with each of the hypothesis:

$$\mathcal{L}_\mathcal{H} = \{z \mapsto \ell(z, h) : h \in \mathcal{H}\}. \tag{6}$$

Theorem 2: With probability at least $1-\delta$, the excess cost of ERM $\widehat{h}$ is bounded as

$$L(\widehat{h}) - L(h^\star) \leq 4R_n(\mathcal{L}_\mathcal{H}) + \sqrt{\frac{2\log(2/\delta)}{n}}. \tag{7}$$

To apply this theorem to the learning algorithms for data-driven risk limiting dispatch problem, we need to bound the Rademacher complexity for the loss classes induced by the hypothesis class and the RLD cost. This is done in the following lemma.

Lemma 1: Given the loss function $\ell(z, h) = \alpha h(x) + \beta(d - h(x))_+$ with Lipschitz coefficient $C^{\max} = \max\{\alpha, \beta - \alpha\}$ for the mapping $h \mapsto \ell(\cdot, h)$, and training set S_n, the following bounds for the Rademacher complexity hold.

1) For the loss class $\mathcal{L}_{\mathcal{H}^{\mathrm{L2}}}$ defined for the linear hypothesis class with bounded ℓ_2 norm $\mathcal{H}^{\mathrm{L2}} = \{x \mapsto w^\top x : w \in \mathbb{R}^p,\ \|w\|_2 \leq W_2^{\max}\}$,
$$R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{L2}}}\right) \leq \frac{C^{\max} X^{\max} W_2^{\max}\sqrt{p}}{\sqrt{n}}.$$
2) For the loss class $\mathcal{L}_{\mathcal{H}^{\mathrm{L1}}}$ defined for the linear hypothesis class with bounded ℓ_1 norm $\mathcal{H}^{\mathrm{L1}} = \{x \mapsto w^\top x : w \in \mathbb{R}^p,\ \|w\|_1 \leq W_1^{\max}\}$,
$$R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{L1}}}\right) \leq \frac{C^{\max} X^{\max} W_1^{\max}\sqrt{2\log(2p)}}{\sqrt{n}}.$$
3) For the loss class $\mathcal{L}_{\mathcal{H}^{\mathrm{K}}}$ defined for the RKHS hypothesis class $\mathcal{H}^{\mathrm{K}} = \{h \in \bar{\mathcal{H}} : \|h\|_{\bar{\mathcal{H}}} \leq V^{\max}\}$,
$$R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{K}}}\right) \leq \frac{C^{\max} V^{\max} K^{\max}}{\sqrt{n}}.$$
4) For the loss class $\mathcal{L}_{\mathcal{H}^{\mathrm{NN2}}}$ defined for deep neural networks with ℓ_2 bounded weights: $\mathcal{H}^{\mathrm{NN2}} = \{x \mapsto h_W(x) : \|W_r^{(j)}\|_2 \leq W_2^{\max},\ j = 1, \ldots, m,\ r = 1, \ldots, d_j\}$,
$$R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{NN2}}}\right) \leq \prod_{j=1}^{m}\sqrt{d_j}\,\frac{2^{m-1}\left(W_2^{\max}\right)^m C^{\max} X^{\max}\sqrt{p}}{\sqrt{n}}.$$
5) For the loss class $\mathcal{L}_{\mathcal{H}^{\mathrm{NN1}}}$ defined for deep neural networks with ℓ_1 bounded weights: $\mathcal{H}^{\mathrm{NN1}} = \{x \mapsto h_W(x) : \|W_r^{(j)}\|_1 \leq W_1^{\max},\ j = 1, \ldots, m,\ r = 1, \ldots, d_j\}$,
$$R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{NN1}}}\right) \leq \frac{2^{m-1}\left(W_1^{\max}\right)^m C^{\max} X^{\max}\sqrt{2\log(2p)}}{\sqrt{n}}.$$

Substituting Rademacher complexity bounds in Lemma 1 into Theorem 2 gives the PAC bounds in Theorem 1 stated in Section IV-B.

V. OPTIMAL UNIFORM RESERVE FOR POWER NETWORK

In this section, we present a simple extension of single-bus results to a general network setting. In particular, the power network has N buses with the power injection region denoted by

$$\mathcal{P} = \{P \in \mathbb{R}^N : \mathbf{1}^\top P = 0,\ HP \leq \bar{f}\},$$

where the first constraint models real power balance, and the second constraint models line flow limit with matrix H being the shift-factor matrix and $\bar{f} \geq 0$ modeling line capacities. We assume $\mathcal{P}$ is compact and has non-empty interior.

In the data-driven risk limiting dispatch problem for a general network, we have a set of p features for each bus of

the network. We can stack all these features and obtain our training set

$$S_n = \{Z_1 = (X_1, D_1), \ldots, Z_n = (X_n, D_n)\},$$

where $X_i \in \mathbb{R}^{Np}$ and $D_i \in \mathbb{R}^N$.

Instead of considering the general data-driven dispatch rule, we focus on a *uniform reserve* case. In particular, we assume that among the Np features there are conventional demand forecasts $\widehat{d}_i \in \mathbb{R}^N$. These demand forecasts may be obtained through simple methods such as autoregressive models or produced by existing forecast software. They are treated as exogenous variables for our purpose. Furthermore, we focus on dispatch rule of the form

$$u = \widehat{d} + \Delta \mathbf{1},$$

where $u \in \mathbb{R}^N$, and $\Delta \in \mathbb{R}$ is the uniform reserve level the system operator purchases for each bus of the network. We note that although such a dispatch rule looks restrictive, it is similar to what is implemented in today's system to cope with a limited amount of renewable generation.

The uniform reserve case of data-driven risk limiting dispatch problem for general network takes the form of

$$\min_{h \in \mathcal{H}} \quad \mathbb{E}_{D_* \sim \mathbb{P}_D} \left[c\left(\widehat{d}_* + h(X_*)\mathbf{1}, D_* \right) \right], \tag{8}$$

where $X_* \in \mathbb{R}^{Np}$ is the test input vector, whose elements include $\widehat{d}_* \in \mathbb{R}^N$. The cost function takes the form of

$$c(u, d) = \min_{g \in \mathbb{R}^N} \quad \alpha^\top u + \beta^\top (g)_+ \tag{9}$$

$$\text{s.t.} \quad g + u - d \in \mathcal{P}, \tag{10}$$

where $\alpha \in \mathbb{R}^N_+$ are the nodal prices of purchasing from day-ahead market, while $\beta \in \mathbb{R}^N_+$ are the nodal prices of purchasing from real-time market.

All algorithms introduced in Section III extends to this setting in a straightforward manner. We give the details of the network case of ERM-L2 as an example below. Other algorithms, i.e., ERM-L1, ERM-K, ERM-NN2, ERM-NN1 can be similarly defined:

$$\begin{aligned} \min_{w \in \mathbb{R}^{Np}} \quad & \frac{1}{n} \sum_{i=1}^n c\left(\widehat{d}_i + \left(w^\top X_i \right) \mathbf{1}, D_i \right) \\ \text{s.t.} \quad & \|w\|_2 \le W_2^{\max}, \end{aligned}$$

which is equivalent to

$$\begin{aligned} \min_{\substack{w \in \mathbb{R}^{Np} \\ \forall i,\ g_i \in \mathbb{R}^N}} \quad & \frac{1}{n} \sum_{i=1}^n \alpha^\top \widehat{d}_i + \left(w^\top X_i \right) \alpha^\top \mathbf{1} + \beta^\top (g_i)_+ \\ \text{s.t.} \quad & g_i + \widehat{d}_i + (w^\top X_i)\mathbf{1} - D_i \in \mathcal{P}, \quad i \in [n], \\ & \|w\|_2 \le W_2^{\max}. \end{aligned}$$

We proceed to show that all performance guarantees presented in Section IV-B hold for the network setting. To this end, we first show that the following function, defined for any given $\widehat{d}$ and d, is Lipschitz continuous:

$$\tilde{c}(\Delta) = c(\widehat{d} + \Delta \mathbf{1}, d).$$

Proposition 1: There exists a finite constant $\widetilde{C}^{\max}$ such that for any $\Delta_1, \Delta_2 \in \mathbb{R}$,

$$|\tilde{c}(\Delta_1) - \tilde{c}(\Delta_2)| \le \widetilde{C}^{\max} |\Delta_1 - \Delta_2|.$$

A way to obtain a value for $\widetilde{C}^{\max}$ is described in the proof.

We can now state our main results regarding using proposed learning algorithms for network data-driven RLD problem.

Theorem 3: Theorem 1 holds for the network data-driven risk limiting dispatch problem (8), with the number of features replaced with Np and the Lipschitz coefficient $C^{\max}$ replaced with $\widetilde{C}^{\max}$.

VI. Concluding remarks

In this paper we study the data-driven risk limiting dispatch problem, where the system operator learns the optimal dispatch rule directly from the data. We present learning algorithms for this problem and establish that they are probably approximately correct. In this process, we introduce a procedure for modifying learning methods to solve data-driven stochastic optimization problems and tools from statistical learning theory for rigorously analyzing the performance of the resulting algorithms. We then extend our learning algorithms and their performance guarantees to a general network setting with power network constraints.

The performance guarantees that we obtain are in the form of probabilistic bounds on $L(\widehat{h}) - \inf_{h \in \mathcal{H}} L(h)$, that is common in learning theory setting but may not be seen as the most ideal bounds one can obtain from a control perspective. The ideal bound would be on the quantity $L(\widehat{h}) - \inf_h L(h)$ without the a priori limitation from the hypothesis class. However, bounds on such quantity may not always be obtainable. Moreover, in principle when we only have a finite sample about the underlying uncertainty model, optimizing in a hypothesis class that is too large may result in overfitting. In fact, we can write

$$L(\widehat{h}) - \inf_h L(h) = \underbrace{L(\widehat{h}) - \inf_{h \in \mathcal{H}} L(h)}_{\text{generalization error}} + \underbrace{\inf_{h \in \mathcal{H}} L(h) - \inf_h L(h)}_{\text{approximation error}},$$

and observe that with a fixed sample size and a hypothesis class of increasing richness, the generalization error increases while the approximation error decreases. It is sometimes possible to bound the approximation error for certain $\mathcal{H}$, see [23] for kernel methods with certain kernels (e.g. Gaussian kernel) and [24] for neural networks.

Our treatment on the network data-driven risk limiting dispatch problem demonstrates that our results can be carried over to the network setting under the restriction that the decision variable is a scalar. A complete solution to the network version of the problem is an interesting direction for future research.

References

[1] DeepMind and National Grid in AI talks to balance energy supply. [Online]. Available: https://www.ft.com/content/27c8aea0-06a9-11e7-97d1-5e720a26771b

[2] C. Rudin, D. Waltz, R. N. Anderson, A. Boulanger, A. Salleb-Aouissi, M. Chow, H. Dutta, P. N. Gross, B. Huang, S. Ierome *et al.*, "Machine learning for the new york city power grid," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 2, pp. 328–345, 2012.
[3] K. Liu, S. Subbarayan, R. Shoults, M. Manry, C. Kwan, F. Lewis, and J. Naccarino, "Comparison of very short-term load forecasting techniques," *IEEE Transactions on power systems*, vol. 11, no. 2, pp. 877–882, 1996.
[4] T. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," *International Journal of Forecasting*, vol. 32, no. 3, pp. 914–938, 2016.
[5] S. S. Soman, H. Zareipour, O. Malik, and P. Mandal, "A review of wind power and wind speed forecasting methods with different time horizons," in *North American power symposium (NAPS), 2010*. IEEE, 2010, pp. 1–8.
[6] P. Pinson, H. A. Nielsen, J. K. Møller, H. Madsen, and G. N. Kariniotakis, "Non-parametric probabilistic forecasts of wind power: required properties and evaluation," *Wind Energy*, vol. 10, no. 6, pp. 497–516, 2007.
[7] R. H. Inman, H. T. Pedro, and C. F. Coimbra, "Solar forecasting methods for renewable energy integration," *Progress in energy and combustion science*, vol. 39, no. 6, pp. 535–576, 2013.
[8] M. D. Ilic, L. Xie, and J.-Y. Joo, "Efficient coordination of wind power and price-responsive demandpart i: Theoretical foundations," *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 1875–1884, 2011.
[9] A. Papavasiliou, S. S. Oren, and R. P. O'Neill, "Reserve requirements for wind power integration: A scenario-based stochastic programming framework," *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 2197–2206, 2011.
[10] R. Jiang, J. Wang, and Y. Guan, "Robust unit commitment with wind power and pumped storage hydro," *IEEE Transactions on Power Systems*, vol. 27, no. 2, pp. 800–810, 2012.
[11] P. Varaiya, F. Wu, and J. Bialek, "Smart Operation of Smart Grid: Risk-Limiting Dispatch," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 40–57, Jan 2011.
[12] R. Rajagopal, E. Bitar, F. F. Wu, and P. Varaiya, "Risk-Limiting Dispatch for Integrating Renewable Power," *International Journal of Electrical Power and Energy Systems, to appear*, 2012.
[13] J. Qin and R. Rajagopal, "Price of uncertainty in multistage stochastic power dispatch," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 4065–4070.
[14] J. Qin, H.-I. Su, and R. Rajagopal, "Storage in risk limiting dispatch: Control and approximation," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 4202–4208.
[15] B. Zhang, R. Rajagopal, and D. Tse, "Network risk limiting dispatch: Optimal control and price of uncertainty," *IEEE Transactions on Automatic Control*, vol. 59, no. 9, pp. 2442–2456, 2014.
[16] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello, "The sample average approximation method for stochastic discrete optimization," *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 479–502, 2002.
[17] C. Zhao, Q. Wang, J. Wang, and Y. Guan, "Expected value and chance constrained stochastic unit commitment ensuring wind power utilization," *IEEE Transactions on Power Systems*, vol. 29, no. 6, pp. 2696–2705, 2014.
[18] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *The annals of statistics*, pp. 1171–1220, 2008.
[19] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.
[20] K. Kawaguchi, "Deep learning without poor local minima," in *Advances in Neural Information Processing Systems*, 2016, pp. 586–594.
[21] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.
[22] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 463–482, 2002.
[23] C. Carmeli, E. De Vito, A. Toigo, and V. Umanitá, "Vector valued reproducing kernel hilbert spaces and universality," *Analysis and Applications*, vol. 8, no. 01, pp. 19–61, 2010.
[24] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.
[25] S. M. Kakade, K. Sridharan, and A. Tewari, "On the complexity of linear prediction: Risk bounds, margin bounds, and regularization," in *Advances in neural information processing systems*, 2009, pp. 793–800.
[26] O. L. Mangasarian and T.-H. Shiau, "Lipschitz continuity of solutions of linear inequalities, programs and complementarity problems," *SIAM Journal on Control and Optimization*, vol. 25, no. 3, pp. 583–595, 1987. [Online]. Available: https://doi.org/10.1137/0325033

APPENDIX I
PROOF OF LEMMA 1

As the loss class is a composition of the hypothesis class and the loss function modeling RLD cost, we use the following composition property of Rademacher complexity, adapted from [22]:

Proposition 2: Given a function class $\mathcal{F}$ and a Lipschitz continuous function ϕ with Lipschitz constant C_ϕ, we have

$$R_n(\phi \circ \mathcal{F}) \leq C_\phi R_n(\mathcal{F}),$$

where $\phi \circ \mathcal{F} = \{z \mapsto \phi(f(z)) : f \in \mathcal{F}\}$.

It remains to bound the Rademacher complexity of the hypothesis classes for the proposed algorithms.

1) ERM-L2, L1: The Rademacher complexity bounds for linear functions with bounded ℓ_2 or ℓ_1 norm are established in [25].

2) ERM-K: For $\mathcal{H} = \mathcal{H}^{\mathrm{K}}$, we have

$$\begin{aligned}
R_n(\mathcal{H}) =& \mathbb{E}_\xi \left[\frac{1}{n} \sup_{h \in \mathcal{H}} \sum_{i=1}^n \xi_i h(X_i) \right] \\
=& \mathbb{E}_\xi \left[\frac{1}{n} \sup_{h \in \mathcal{H}} \left\langle h, \sum_{i=1}^n \xi_i k(\cdot, X_i) \right\rangle \right] \\
=& \mathbb{E}_\xi \left[\frac{V^{\max}}{n} \left\langle \sum_{i=1}^n \xi_i k(\cdot, X_i), \sum_{i=1}^n \xi_i k(\cdot, X_i) \right\rangle \right] \\
\leq& \frac{V^{\max}}{n} \sqrt{\mathbb{E}_\xi \left[\left\langle \sum_{i=1}^n \xi_i k(\cdot, X_i), \sum_{i=1}^n \xi_i k(\cdot, X_i) \right\rangle \right]} \\
=& \frac{V^{\max}}{n} \sqrt{\sum_{i=1}^n \langle k(\cdot, X_i), k(\cdot, X_i) \rangle} \\
=& \frac{V^{\max}}{n} \sqrt{\sum_{i=1}^n k(X_i, X_i)} \\
\leq& \frac{V^{\max}}{n} \sqrt{n \left(K^{\max}\right)^2} = \frac{V^{\max} K^{\max}}{\sqrt{n}},
\end{aligned}$$

where line 3 is due to the Cauchy-Schwartz condition for equality.

3) ERM-NN2, NN1: Let $\mathcal{H}^{(1)} = \mathcal{H}^{\mathrm{L2}}$. Define recursively

$$\mathcal{H}^{(j+1)} = \left\{ h^{(j+1)}(x) = \sum_{s=1}^{d_j} W_{rs}^{(j+1)} \sigma\left(h_s^{(j)}(x)\right) : \begin{matrix} \|W_r^{(j+1)}\|_2 \leq W_2^{\max} \\ \forall s,\ h_s^{(j)} \in \mathcal{H}^{(j)} \end{matrix} \right\}$$

for any $r = 1, \ldots, d_{j+1}$. Then

$$
\begin{aligned}
&R_n\left(\mathcal{H}^{(j+1)}\right) \\
&= \mathbb{E}_\xi\left[\frac{1}{n}\sup_{h^{(j+1)}\in\mathcal{H}^{(j+1)}}\sum_{i=1}^{n}\xi_i h^{(j+1)}(X_i)\right] \\
&= \mathbb{E}_\xi\left[\frac{1}{n}\sup_{\substack{\|W_r^{(j+1)}\|_2\le W_2^{\max}\\ \forall s,\ h_s^{(j)}\in\mathcal{H}^{(j)}}}\sum_{i=1}^{n}\sum_{s=1}^{d_j}\xi_i W_{rs}^{(j+1)}\sigma\left(h_s^{(j)}(X_i)\right)\right] \\
&\le \mathbb{E}_\xi\left[\frac{W_2^{\max}}{n}\sup_{\forall s,\ h_s^{(j)}\in\mathcal{H}^{(j)}}\left\|\left\{\sum_{i=1}^{n}\xi_i\sigma\left(h_s^{(j)}(X_i)\right)\right\}_{s\in[d_j]}\right\|_2\right] \\
&\le \sqrt{d_j}\mathbb{E}_\xi\left[\frac{W_2^{\max}}{n}\max_{s\in[d_j]}\sup_{\forall s,\ h_s^{(j)}\in\mathcal{H}^{(j)}}\left|\sum_{i=1}^{n}\xi_i\sigma\left(h_s^{(j)}(X_i)\right)\right|\right] \\
&= W_2^{\max}\sqrt{d_j}\mathbb{E}_\xi\left[\frac{1}{n}\sup_{h^{(j)}\in\mathcal{H}^{(j)}}\left|\sum_{i=1}^{n}\xi_i\sigma\left(h^{(j)}(X_i)\right)\right|\right] \\
&\le 2W_2^{\max}\sqrt{d_j}R_n\left(\mathcal{H}^{(j)}\right).
\end{aligned}
$$

Thus

$$
R_n\left(\mathcal{H}^{\mathrm{NN2}}\right) \le (2W_2^{\max})^{m-1}\left(\prod_{j=1}^{m}\sqrt{d_j}\right)R_n\left(\mathcal{H}^{\mathrm{L2}}\right),
$$

and so

$$
R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{NN2}}}\right) \le (2W_2^{\max})^{m-1}\prod_{j=1}^{m}\sqrt{d_j}\frac{C^{\max}X^{\max}W_2^{\max}\sqrt{p}}{\sqrt{n}}.
$$

Similarly, we can derive

$$
R_n\left(\mathcal{L}_{\mathcal{H}^{\mathrm{NN1}}}\right) \le (2W_1^{\max})^{m-1}\frac{C^{\max}X^{\max}W_1^{\max}\sqrt{2\log(2p)}}{\sqrt{n}}.
$$

Appendix II
Proof of Proposition 1

Consider the optimization parameterized by Δ,

$$
\min_{g\in\mathbb{R}^N} \quad \beta^\top(g)_+ \tag{11}
$$

$$
\text{s.t.} \quad g + \widehat{d} + \Delta\mathbf{1} - d \in \mathcal{P}, \tag{12}
$$

which is equivalent to (9) as the first term in the objective function of (9) does not depend on the decision of the optimization. Optimization (11) can be converted into a linear program, by introducing variables for positive and negative parts of g. By [26], we have for $\Delta_1, \Delta_2 \in \mathbb{R}$, the corresponding solution of (11), denoted by g_1 and g_2, satisfies

$$
\|g_1 - g_2\|_\infty \le C_g^{\max}|\Delta_1 - \Delta_2|,
$$

with $C_g^{\max}$ finite defined as equation (2.19) of [26]. As function $g \mapsto \beta^\top(g)_+$ is $\|\beta\|_2$-Lipschitz, we have

$$
\begin{aligned}
|\beta^\top(g_1)_+ - \beta^\top(g_2)_+| &\le \|\beta\|_2\|g_1 - g_2\|_2 \\
&\le \sqrt{N}\|\beta\|_2 C_g^{\max}|\Delta_1 - \Delta_2|.
\end{aligned}
$$

It follows that

$$
\begin{aligned}
&|\tilde{c}(\Delta_1) - \tilde{c}(\Delta_2)| \\
&= \left|\left(\Delta_2\alpha^\top\mathbf{1} + \beta^\top(g_2)_+\right) - \left(\Delta_1\alpha^\top\mathbf{1} + \beta^\top(g_1)_+\right)\right| \\
&\le \underbrace{\left(\alpha^\top\mathbf{1} + \sqrt{N}\|\beta\|_2 C_g^{\max}\right)}_{\widetilde{C}^{\max}}|\Delta_2 - \Delta_1|.
\end{aligned}
$$

Appendix III
Proof of Theorem 3

For the network setting, the loss function is

$$
\ell(h, z) = c(\widehat{d} + h\mathbf{1}, d),
$$

where $z = (x, d)$ with $x \in \mathbb{R}^{Np}$ and $d \in \mathbb{R}^N$, and $\widehat{d}$ is part of the entries of x. By Proposition 1, the loss class $\mathcal{L}_\mathcal{H} = \{z \mapsto \ell(z, h) : h \in \mathcal{H}\}$ is a composition of a $\widetilde{C}^{\max}$-Lipschitz function with the hypothesis class. Thus using the same lines of argument in the proof of Lemma 1, we can bound the Rademacher complexity of the loss classes associated with all the proposed algorithms. Invoking Theorem 2 completes the proof.