Deep Learning for Optimal Energy-Efficient Power Control in Wireless Interference Networks

Bho Matthiesen, *Student Member, IEEE,* Alessio Zappone, *Senior Member, IEEE,* Eduard A. Jorswieck, *Senior Member, IEEE,* Merouane Debbah, *Fellow, IEEE*

Abstract—This work develops a deep learning power control framework for energy efficiency maximization in wireless interference networks. Rather than relying on suboptimal power allocation policies, the training of the deep neural network is based on the globally optimal power allocation rule, leveraging a newly proposed branch-and-bound procedure with a complexity affordable for the offline generation of large training sets. In addition, no initial power vector is required as input of the proposed neural network architecture, which further reduces the overall complexity. As a benchmark, we also develop a first-order optimal power allocation algorithm. Numerical results show that the neural network solution is virtually optimal, outperforming the more complex first-order optimal method, while requiring an extremely small online complexity.

Index Terms—Energy Efficiency, Deep Learning, Artificial Neural Networks, Power Control, Interference Networks.

I. INTRODUCTION

Energy management is known to be one of the crucial issues for the sustainability of future wireless communication networks, whose bit-per-Joule energy efficiency (EE) is required to increase by a factor 2,000 compared to present networks [2]. To this end, several energy management techniques have been proposed, such as energy-efficient network deployment, the use of renewable energy sources, as well as the development of resource allocation techniques aimed at EE maximization [3]. This work focuses on the last of these energy management approaches, and in particular on the issue of energy-efficient power control.

Due to the fractional nature of energy-efficient performance metrics, traditional convex optimization theory can not directly handle energy-efficient power control problems. Instead, the mathematical frameworks of generalized concavity theory and fractional programming provide a suitable set of optimization methods. However, these optimization tools come with a limited complexity only when the fractional function to maximize fulfills certain mathematical assumptions, such as the concavity of the numerator [4]. Unfortunately, this requirement is not fulfilled whenever an interference-limited network needs to be optimized, and indeed in these cases EE maximization problems are typically NP-hard [5]. Thus, several suboptimal methods have been proposed for energy-efficient resource allocation in wireless interference networks. The simplest approach is to resort to interference cancelation techniques or to orthogonal transmission schemes, thus falling back into the noise-limited regime [6]–[8]. However, this either leads to a poor resource efficiency, or to noise enhancement effects and/or non-linear receive schemes. A more recent approach is instead that of trying to develop energy-efficient power control algorithms that, although not provably optimal, enjoy limited (typically polynomial) complexity. In [9], the maximization of the system global energy efficiency (GEE) is pursued, by merging fractional programming with alternating optimization, decomposing the problem into a series of simpler sub-problems. A similar approach is used in [10], where the minimum of the users' EEs is maximized and in [11], where the sum of the individual users' EEs is considered. In [12], fractional programming is merged with sequential optimization theory, to develop power control algorithms for the maximization of the system GEE or minimum of the users' EEs. Unlike previous contributions, the method proposed there guarantees first-order optimality, and has been numerically shown to achieve global optimality in [13]. Instead, a similar result is not available when the sum of the users' EEs is to be maximized, which is a less investigated scenario, and acknowledged as the hardest energy-efficient power control problem [4].

Nevertheless, regardless of the performance function to maximize, all available approaches based on traditional optimization theory do not lead to a simple online implementation. Indeed, in fading-impaired wireless networks, the optimal power allocation will inevitably depend on the propagation channels, which requires to solve the optimization problem anew anytime the propagation scenario changes. This is especially problematic when the optimal allocation is based on the instantaneous channel realizations, which can vary quite rapidly in mobile environments. In these cases, online power control causes a significant complexity overhead, since even suboptimal power control methods that enjoy polynomial complexity are based on involved numerical algorithms and do not provide a closed-form expression for the optimal power allocation as a function

B. Matthiesen and E.A. Jorswieck are with the Chair for Communications Theory, Communications Laboratory, Technische Universität Dresden, Dresden, Germany (e-mail: bho.matthiesen@tu-dresden.de, eduard.jorswieck@tu-dresden.de).

A. Zappone and M. Debbah are with the Large Networks and Systems Group, CentraleSupélec, Gif-sur-Yvette, 91192, France (e-mail: alessio.zappone@l2s.centralesupelec.fr, merouane.debbah@l2s.centralesupelec.fr). M. Debbah is also with the Mathematical and Algorithmic Sciences Laboratory, France Research Center, Huawei Technologies, Paris, France.

of the system propagation channels.

The aim of this work is to show that this drawback can be overcome leveraging the framework of artificial intelligence (AI), and in particular deep learning [14], which has been recently proposed for wireless communication systems [15]–[17]. Focusing on the problem of EE maximization, we show that it is possible to train a deep artificial neural network (ANN) to provide near-optimal energy-efficient power allocations, with a complexity limited to the computation of simple algebraic operations. The possibility of AI-based resource allocation in wireless networks was recently put forth in [18], and power control through deep learning has been proposed in [19]–[24]. In [19], [20], power control for weighted sum-rate maximization in interference channels is performed employing the tool of deep reinforcement learning. However, deep reinforcement learning is not suitable for quasi-static block fading channels, where the fading realizations vary independently from one fading block to the next. For such a scenario, [21] proposes to use an ANN to emulate a given iterative power control algorithm, and demonstrates the method with reference to the weighted MMSE algorithm for sum-rate maximization [25]. It is shown that a limited gap with respect to the weighted MMSE algorithm can be obtained, while at the same time significantly reducing the computational complexity. Sum-rate maximization by a series of deep neural network is considered in [22], where it is proposed to use the sum-rate function as cost function directly during the training phase of the neural networks. The proposed method is shown to outperform the weighted MMSE method in some configurations. In [23], a decentralized robust precoding scheme in a network MIMO system is developed by ANNs, aimed at optimizing the users' transmission rates. Instead, [24] proposes to merge deep learning and transfer learning for resource allocation in wireless interference networks. It is shown that using synthetic data generated from mathematical models to augment datasets based on true, measured data, can significantly simplify the training procedure of ANNs.

While all above works on AI-based power control are focused on sum-rate maximization, with the exception of [24], which considers GEE maximization, this work deals with the development of a generic ANN-based power control algorithm that is able to maximize all most relevant energy-efficient metrics, considering the weighted sum energy efficiency (WSEE) as a specific case-study. This choice is motivated by the fact that, as already remarked, WSEE maximization is considered the hardest among existing energy-efficient maximization problems, which poses complexity issues even in some noise-limited setups, due to the fact that the sum of pseudo-concave functions is not guaranteed to be pseudo-concave. Demonstrating the effectiveness of ANNs in maximizing the WSEE appears as the most compelling argument to motivate ANN-based approaches also for other energy-efficient problems. Specifically, the main contributions of the work are as follows:

- We develop an ANN-based approach for energy-efficient power control in wireless interference networks. The proposed approach has an extremely low complexity, essentially providing a closed-form, near-optimal solution. Unlike previously mentioned works that train an ANN to emulate a given, suboptimal algorithm, the proposed method trains an ANN directly based on the optimal power allocation rule. As a result, globally optimal performance can be achieved, in contrast to previous approaches that are intrinsically limited by the suboptimality of the algorithm that is emulated. Moreover, previous related works require to specify an initialization power vector, which is instead not required by the proposed ANN architecture, thus further reducing the overall complexity. The analysis is carried out focusing on the case-study of WSEE maximization, but the proposed method is general enough to apply to all major energy-efficient performance metrics.
- We develop a novel branch-and-bound (BB) algorithm to globally solve energy-efficient power control problems with a complexity that is significantly lower than standard global optimization approaches to these kind of problems. The proposed BB algorithm significantly simplifies the generation of the training set required to implement the ANN-based approach introduced above. The proposed method is general enough to apply to all major energy-efficient metrics.
- We also propose a new power control framework that is guaranteed to obtain a first-order optimal solution of the WSEE maximization problem with polynomial complexity. Besides having its own merits, this novel first-order optimal power control method also provides a meaningful benchmark for the proposed ANN-based power control method.
- Extensive numerical results are provided to assess the performance of the proposed ANN-based approach. It is found that, despite having an extremely limited computational complexity, the ANN-based approach outperforms the first-order optimal approach, achieving near-optimal performance. Moreover, our numerical analysis shows that the ANN-based method is even robust to mismatches in the channel statistics between the training data and the actual scenario in which the ANN is tested.

The rest of the work is organized as follows. Section II describes the system model and formulates the power control problem. Section III develops the proposed ANN-based power control method, while Section IV and Section V introduce the proposed globally optimal and first-order optimal power control algorithms, respectively. Section VI illustrates the numerical performance assessment of the proposed algorithms, while concluding remarks are provided in Section VII.

A. Notation

Vectors $\boldsymbol{a}$ and matrices $\boldsymbol{A}$ are typeset in bold face, sets and maps in calligraphic letters $\mathcal{A}$. The sets of real and complex numbers are denoted by $\mathbb{R}$ and $\mathbb{C}$, respectively. We define the sets $\{a_i\}_{i=1}^n = \{a_1, a_2, \ldots, a_n\}$ and vectors $(a_i)_{i=1}^n = (a_1, a_2, \ldots, a_n)$ where we might omit the index bounds if clear from the context. The closed $[a, b]$ and open (a, b) intervals are generalized to n-dimensional vectors as $[\boldsymbol{a}, \boldsymbol{b}] = \{\boldsymbol{x} \mid a_i \leq x_i \leq b_i, \text{ for all } i = 1, \ldots, n\}$. In the same way, vector inequalities $\boldsymbol{a} \leq \boldsymbol{b}$ are element-wise, i.e., $a_i \leq b_i$ for all $i = 1, \ldots, n$, $\boldsymbol{0}$ denotes a zero vector of appropriate

dimension, and scalar functions applied to vectors are to be applied element-wise, e.g., $\log \boldsymbol{a} = (\log a_i)_{i=1}^n$. The operators $\|\cdot\|$, $|\cdot|$, $(\cdot)^T$, and $(\cdot)^H$ denote the L^2-norm, absolute value, transpose, and conjugate transpose, respectively. Logarithms are, unless noted otherwise, to the base 2, and $\ln$ denotes the natural logarithm. The diameter $\operatorname{diam}\mathcal{A}$ of a set $\mathcal{A}$ is the maximum distance between two points in $\mathcal{A}$. The derivative, partial derivative, and gradient of a function f are written as $\frac{d}{dx}f(x)$, $\frac{\partial}{\partial x_i}f(\boldsymbol{x})$, and $\nabla f(\boldsymbol{x}) = (\frac{\partial}{\partial x_i}f(\boldsymbol{x}))_i$, respectively. Finally, definitions and approximations are marked by $:=$ and $\approx$, respectively.

II. System Model & Problem Statement

We consider the uplink of a multi-cell interference network in which L single antenna user equipments (UEs) are served by M base stations (BSs), equipped with n_R antennas at each BS. The received signal at BS $a(i)$ is

$$\boldsymbol{y}_{a(i)} = \sum_{j=1}^{L} \boldsymbol{h}_{a(i),j} x_j + \boldsymbol{z}_{a(i)} \tag{1}$$

wherein $\boldsymbol{h}_{a(i),j} \in \mathbb{C}^{n_R}$ is the channel from UE j to BS $a(i)$, $x_j \in \mathbb{C}$ the symbol transmitted by UE j, and $\boldsymbol{z}_{a(i)}$ zero-mean circularly symmetric complex Gaussian noise with power σ_i^2. Each UE is subject to an average transmit power constraint, i.e., $p_j \leq P_j$ where p_j is the average power of x_j, and the BSs use matched filtering. Thus, the signal (1) is processed by computing

$$\frac{\boldsymbol{h}_{a(i),i}^H \boldsymbol{y}_{a(i)}}{\|\boldsymbol{h}_{a(i),i}\|} = \|\boldsymbol{h}_{a(i),i}\| x_i + \sum_{j \neq i} \frac{\boldsymbol{h}_{a(i),i}^H \boldsymbol{h}_{a(i),j}}{\|\boldsymbol{h}_{a(i),i}\|} x_j + \frac{\boldsymbol{h}_{a(i),i}^H}{\|\boldsymbol{h}_{a(i),i}\|} \boldsymbol{z}_{a(i)}.$$

Then, under the assumption of Gaussian codebooks, the achievable rate from UE i to its intended BS is

$$R_i = B \log \left(1 + \frac{\|\boldsymbol{h}_{a(i),i}\|^2 p_i}{\sigma_i^2 + \sum_{j \neq i} \frac{\left|\boldsymbol{h}_{a(i),i}^H \boldsymbol{h}_{a(i),j}\right|^2}{\|\boldsymbol{h}_{a(i),i}\|^2} p_j} \right), \tag{2}$$

where B is the communication bandwidth. This rate expression is equivalent to

$$R_i = B \log \left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j} \right) \tag{3}$$

with $\alpha_i = \frac{\|\boldsymbol{h}_{a(i),i}\|^2}{\sigma_i^2}$ and $\beta_{i,j} = \frac{\left|\boldsymbol{h}_{a(i),i}^H \boldsymbol{h}_{a(i),j}\right|^2}{\sigma_i^2 \|\boldsymbol{h}_{a(i),i}\|^2}$.

In this context, the EE of the link between UE i and its intended BS $a(i)$ is defined as the benefit-cost ratio in terms of the link's achievable rate and power consumption necessary to operate the link, i.e.,

$$\mathrm{EE}_i = \frac{B \log \left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j} \right)}{\mu_i p_i + P_{c,i}} \tag{4}$$

where μ_i is the inefficiency of UE i's power amplifier and $P_{c,i}$ is the total static power consumption of UE i and its associated BS.

A. Problem formulation and motivation

The goal of this work is to develop an efficient and effective algorithm to solve energy-efficient power control problems. As a specific case-study, the main focus of the work will be on the maximization of the WSEE, subject to maximum power constraints. Namely, the problem at hand is formulated as

$$\begin{cases} \max_{\boldsymbol{p}} & \sum_{i=1}^{L} w_i \dfrac{\log \left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j} \right)}{\mu_i p_i + P_{c,i}} \\ \text{s.t.} & 0 \leq p_i \leq P_i, \quad \text{for all } i = 1, 2, \ldots, L, \end{cases} \tag{P1}$$

Regarding the motivation for considering the WSEE as main case-study, several arguments can be made. First of all, it is well-known that the WSEE is the hardest to maximize among energy-efficient metrics. Thus, showing that an ANN is indeed able to learn the structure of the optimal solution of (P1), is a very strong motivation for the use of ANNs to tackle other energy-efficient power control problems, too. Among the reasons that make Problem (P1) so challenging to handle by traditional optimization techniques, the following observations hold:

- The objective of (P1) is a sum of fractions, a functional form that is NP-hard in general [26], and which indeed can not be tackled with polynomial complexity by any available fractional programming technique.
- Each summand of the objective of (P1) is a fraction with a non-concave numerator, which would make (P1) NP-hard even if only the weighted sum-rate were to be maximized (i.e. if $\mu_i = 0$ for all $i = 1, \ldots, L$) [5].
- Finally, even suboptimal methods to solve (P1) that are based on convex or pseudo-convex relaxations become problematic when the goal is to deploy the optimized power allocation in real-time, i.e. updating the optimal power allocation policy any time the realizations of the propagation channels $\{\boldsymbol{h}_{i,j}\}_{i,j}$ changes. Indeed, any change in the channel realizations will result in different parameters $\{\alpha_i, \beta_{i,j}\}_{i,j}$, thus modifying the optimal solution of (P1). As a consequence, in order to develop an online power control algorithm that is able to track the fading channel fluctuations, it would be needed to solve (P1) within the channels coherence time.

Another reason to motivate the consideration of the WSEE lies in its different operational meaning compared to the more widely-considered GEE metric. While the latter is meant to optimize the EE of the network as a whole, the former enables to balance the EE levels among the UEs, thanks to the use of the weights $w_i \geq 0$, prioritizing the EE of some links over others. For example, some terminals might not have a reliable energy source, and might, for example, be powered by renewable energy sources. In this case, it might be useful to prioritize the individual EEs of these users over that of the other users.

Further elaborating on the prioritization of the users' individual EEs, it is important to mention that Problem (P1) can be cast into the framework of multi-objective optimization. Formally

speaking, let us consider the problem of jointly maximizing all of the users' individual EEs, namely

$$\max_{\boldsymbol{0} \leq \boldsymbol{p} \leq \boldsymbol{P}}[\mathrm{EE}_1(\boldsymbol{p}), \mathrm{EE}_2(\boldsymbol{p}), \ldots, \mathrm{EE}_L(\boldsymbol{p})] \qquad (5)$$

with $\boldsymbol{p} = [p_1, \ldots, p_L]$ and $\boldsymbol{P} = [P_1, \ldots, P_L]$. A moment's thought shows that the individual EEs are conflicting objectives, since, for all i, (4) is strictly decreasing in p_j for all $j \neq i$, i.e., $p_j = 0$ for all $j \neq i$ maximizes (4), but the optimal p_i is strictly positive. Thus, no power allocation vector exists that simultaneously maximizes all individual EEs. In this context, in order to define a suitable solution concept for (5), the notion of energy-efficient Pareto region is defined as the set of all EE vectors $[\mathrm{EE}_1(\boldsymbol{p}), \mathrm{EE}_2(\boldsymbol{p}), \ldots, \mathrm{EE}_L(\boldsymbol{p})]$ which can be attained by a feasible power allocation vector $\boldsymbol{p}$. The outer boundary of the Pareto region is called Pareto boundary, and provides all feasible operating points at which it is not possible to improve the EE of one user, without decreasing the EE of another user [27], [28]. For this reason, the operating points on the Pareto-boundary are called Pareto-optimal and they are commonly understood as the solutions of the multi-objective optimization problem.

Having said this, a popular method of determining Pareto-optimal solutions is by the so-called scalarization approach, which consists of maximizing a weighted sum of the objectives. Therefore, Problem (P1) is the scalarized version of the multi-objective problem (5), and thus solving (P1) yields a Pareto-optimal[1] solution of (5).

A third motivation to focus on the WSEE is that it is a direct generalization of the system weighted sum-rate

$$\mathrm{WSR} = \sum_{i=1}^{L} w_i \log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right), \qquad (6)$$

obtained from the WSEE by setting $\mu_i = 0$ and $P_{c,i} = 1$ for all $i = 1, \ldots, L$. In addition, we emphasize that the ANN-based approach and the global optimization approach to be developed in Section III and Section IV, respectively, are not restricted to the WSEE, and will be shown to encompass all major energy-efficient metrics, namely the weighted product and minimum of the individual EEs

$$\mathrm{WPEE} = \prod_{i=1}^{L} \left(\frac{\log\left(1 + \frac{\alpha_i p_i}{1+\sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}\right)^{w_i} \qquad (7)$$

$$\mathrm{WMEE} = \min_{1 \leq i \leq L} w_i \frac{\log\left(1 + \frac{\alpha_i p_i}{1+\sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}, \qquad (8)$$

as well as the system GEE

$$\mathrm{GEE} = \frac{\sum_{i=1}^{L} \log\left(1 + \frac{\alpha_i p_i}{1+\sum_{j \neq i} \beta_{i,j} p_j}\right)}{\sum_{i=1}^{L} \mu_i p_i + P_{c,i}}. \qquad (9)$$

The following section introduces the ANN architecture used by the proposed energy-efficient power control approach.

---

[1]It can be proved that solving a scalarized problem for varying combinations of the weights enables to obtain the complete convex hull of the Pareto-boundary.
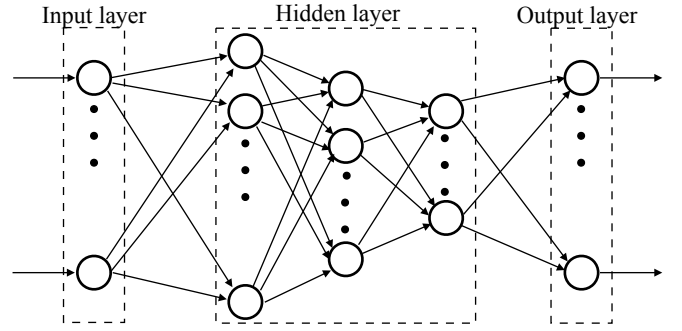

Fig. 1. General scheme of a deep feedforward ANN with fully-connected layers.

III. ANN-BASED POWER CONTROL

The key challenge to overcome towards the development of a low-complexity, near-optimal power allocation algorithm is the complex structure of Problem (P1). Specifically, all available optimization approaches rely on numerical procedures and fail to provide the optimal power allocation in closed-form, as a function of the propagation channels $\{\alpha_i, \beta_{i,j}\}_{i,j}$ and system parameters $\{P_i\}_i$. If such a closed-form were available, then computing the optimal power allocation would have a negligible complexity, and it would be possible to update the power allocation policy in real-time, i.e. tracking the fading channel variations. Thus, we aim at determining an expression for the map:

$$\mathcal{F} : \mathbf{a} = (\alpha_i, \beta_{i,j}, P_i)_{i,j} \in \mathbb{R}^{L(L+1)} \mapsto \boldsymbol{p}^* \in \mathbb{R}^L, \qquad (10)$$

with $\boldsymbol{p}^*$ the optimal power allocation corresponding to $\mathbf{a}$. Such a reformulation of the power allocation problem motivates the use of an ANN to learn the map (10), since ANNs, and in particular feedforward neural networks with fully-connected layers, are universal function approximators [29].

To elaborate, let us briefly introduce the architecture of the considered ANN. We employ a fully-connected feedforward network which takes as input a realization of the parameter vector $\mathbf{a}$, producing as output a power allocation vector $\hat{\boldsymbol{p}}$, which estimates the optimal power allocation vector $\mathbf{p}^*$ corresponding to $\mathbf{a}$. Between the input and output layer, K hidden layers are present. For all $k = 1, \ldots, K + 1$, the k-th layer has N_k neurons, with neuron n computing

$$\zeta_k(n) = f_{n,k}\big(\boldsymbol{\gamma}_{n,k}^T \boldsymbol{\zeta}_{k-1} + \delta_{n,k}\big) \qquad (11)$$

wherein $\boldsymbol{\zeta}_k = (\zeta_k(1), \ldots, \zeta_k(N_{k+1}))$ denotes the $N_{k+1} \times 1$ output vector of layer k, $\boldsymbol{\gamma}_{n,k} \in \mathbb{R}^{N_{k-1}}$ and $\delta_{n,k} \in \mathbb{R}$ are neuron-dependent weights and bias terms, respectively, while $f_{n,k}$ is the activation function[2] of neuron n in layer k. Apparently, each neuron performs quite simple operations. Nevertheless, combining the processing of multiple neurons, ANNs can perform very complex tasks and obtain an overall input-output map that emulates virtually any function. Formally, the following universal approximation result is proved in [29, Theorem 1].

---

[2]In principle, any function can be considered as activation function, even though widely accepted choices are sigmoidal functions, rectified linear units (ReLU), and generalized ReLU functions. The specific choices considered in this work are discussed in Section VI.

Proposition 1: Consider a single layer of an ANN with $n \times 1$ input vector $\boldsymbol{\xi}$, $m \times n$ weight matrix $\widetilde{\boldsymbol{\Gamma}}$, $m \times 1$ bias vector $\widetilde{\boldsymbol{\delta}}$, and activation functions $\boldsymbol{f} = [f_1, \ldots, f_m]$. Let us define the set of all input-output maps that can be obtained by the ANN, i.e. the set

$$\mathcal{A}_n = \left\{ \boldsymbol{f}\left(\widetilde{\boldsymbol{\Gamma}}\boldsymbol{\xi} + \widetilde{\boldsymbol{\delta}}\right) \mid \widetilde{\boldsymbol{\Gamma}} \in \mathbb{R}^{n \times m}, \widetilde{\boldsymbol{\delta}} \in \mathbb{R}^m \right\}. \quad (12)$$

Then, $\mathcal{A}_n$ is dense in the set of continuous functions if and only if $\boldsymbol{f}$ is not an algebraic polynomial.

Proposition 1 formally proves that the input-output relationship of an ANN can emulate any continuous map.[3] In addition to Proposition 1, [30] provides bounds for the number of neurons to be used to obtain a given approximation accuracy. However, although of great theoretical importance, neither of these results is constructive, in the sense that they do not provide any guidance as to the topology of the ANN to use and how to configure the weights and biases to achieve a desired approximation accuracy. In practice, it has been empirically shown that deep architectures, i.e. ANN with multiple layers, tend to require less neurons [14, Sec. 6.4.1] to achieve the same level of accuracy, which motivates us to employ a deep ANN. Moreover, it is intuitively clear that the number of neurons to employ increases with the size of the problem, i.e. with the dimension of the domain and co-domain of the map to estimate. For the case of (10), it is needed to estimate a map from an $L(L+1)$-dimensional space to an L-dimensional space.

At this point, the problem remains of how to tune the weights and biases to reliably estimate (10). To this end, the weights $\boldsymbol{\Gamma} = \{\gamma_{n,k}\}_{n,k}$, and the biases $\boldsymbol{\delta} = \{\delta_{n,k}\}_{n,k}$ must be adjusted in a supervised learning fashion by training the ANN. This requires the use of a so-called training set, i.e. a set of the form $\{(\boldsymbol{a}_n, \boldsymbol{p}_n^*) \mid n = 1, \ldots, N_T\}$ with N_T training tuples $(\boldsymbol{a}_n, \boldsymbol{p}_n^*)$, wherein $\boldsymbol{p}_n^*$ is the optimal power allocation vector corresponding to $\boldsymbol{a}_n$. In other words, the training set contains examples of desired power allocation vectors corresponding to some possible configurations of system parameters $\boldsymbol{a}_n$. By exploiting these examples, the ANN learns to predict the power allocation also for new realizations of $\boldsymbol{a}_n$ that are not contained in the training set. Mathematically speaking, the training process consists of adjusting the weights and biases of the ANN in order to minimize the loss between actual and desired output, namely considering the problem:

$$\min_{\boldsymbol{\Gamma}, \boldsymbol{\delta}} \frac{1}{N_T} \sum_{n=1}^{N_T} \mathcal{L}(\hat{\boldsymbol{p}}_n(\boldsymbol{\Gamma}, \boldsymbol{\delta}), \boldsymbol{p}_n^*) \quad (13)$$

with $\mathcal{L}(\cdot, \cdot)$ any suitable measure of the error incurred when the actual output of the ANN corresponding to the n-th training input is $\hat{\boldsymbol{p}}_n$, while the desired output was $\mathbf{p}_n^*$. A widely-used example of error measure is the squared error $\|\hat{\boldsymbol{p}}_n(\boldsymbol{\Gamma}, \boldsymbol{\delta}) - \boldsymbol{p}_n^*\|^2$ [14]. The minimization of (13) can be tackled by state-of-the-art, off-the-shelf stochastic gradient descent methods specifically developed for ANNs training [14], and therefore will not be discussed here. Instead, it is interesting to note that the learning process can be simplified by normalizing the transmit powers before running the stochastic

gradient descent training algorithm. Specifically, applying the variable change $p_i \to \tilde{p}_i P_i$, for all $i = 1, \ldots, L$, we normalize the transmit power to lie in the interval $[0, 1]$, which leads to the following equivalent reformulation of Problem (P1)

$$\begin{cases} \max_{\tilde{\boldsymbol{p}}} & \sum_{i=1}^{L} w_i \dfrac{\log\left(1 + \dfrac{\tilde{\alpha}_i \tilde{p}_i}{1 + \sum_{j \neq i} \tilde{\beta}_{i,j} \tilde{p}_j}\right)}{\mu_i P_i \tilde{p}_i + P_{c,i}} & \text{(P2)} \\ \text{s.t.} & 0 \leq \tilde{p}_i \leq 1, \quad \text{for all } i = 1, 2, \ldots, L, \end{cases}$$

wherein $\tilde{\alpha}_i = \alpha_i P_i$, $\tilde{\beta}_i = \beta_i P_i$, $\tilde{\mu}_i = \mu_i P_i$, for all $i = 1, \ldots, L$. Then, the normalized training set then is $\mathcal{S}_T = \{(\tilde{\boldsymbol{a}}_n, \tilde{\boldsymbol{p}}_n^*) \mid n = 1, \ldots, N_T\}$ with parameter vector $\tilde{\boldsymbol{a}} = (\tilde{\alpha}_i, \tilde{\beta}_{i,j}, P_i)_{i,j}$. The advantage of this reformulation is that, despite the values $P_1, \ldots, P_L$, the transmit powers are always in the set $[0, 1]$. Intuitively, this simplifies the dependence of the optimal power allocation on the maximum power constraints, thereby making it easier for the ANN to grasp the optimal power allocation structure as a function of the maximum power constraints.

After the training phase, all weights and biases of the ANN are configured and the ANN provides the desired closed-form estimate of the map (10). Indeed, once the weights and biases have been set, the input-output relationship of the ANN can be written in closed-form as the composition of the affine combinations and activation functions of the neurons in the ANN. This effectively provides a closed-form expression for the map (10), within an approximation accuracy that can be made small at will by properly designing and training the ANN. Thus, as $\mathbf{a}$ changes due to channel fading, the corresponding power allocation can be obtained without having to solve Problem (P1) again, but simply computing the output of the ANN when the input is the new realization of $\mathbf{a}$. As further analyzed in the rest of this section, this grants a huge complexity reduction over traditional optimization methods, which instead require to tackle (P1) by numerical methods each time the realization of $\mathbf{a}$ changes.

A. Computational complexity

The complexity of the proposed ANN-based power allocation can be divided into two main components:

(a) Using the trained ANN for the online computation of the power allocation vector.
(b) Building the training set and implement the training procedure.

The first task requires computing the output $\zeta_k(n)$ of each neuron in the ANN, moving from the first layer to the output layer, which in turn requires $\sum_{k=1}^{K+1} N_{k-1} N_k$ real multiplications,[4] and evaluating $\sum_{k=1}^{K+1} N_k$ scalar activation functions $f_{n,k}$. Despite being typically non-linear, the activation functions are elementary functions whose computation does not pose any significant computational issue. Thus, obtaining the output of the trained ANN for any given input vector entails a negligible complexity.

---

[3]The continuity of (10) will be analyzed in Section IV.

[4]The complexity related to additions is neglected as it is much smaller than that required for multiplications.

As for the second task, most of the complexity lies in the generation of the training set,[5] which requires actually solving the NP-hard Problem (P1) N_T times, i.e. for N_T different realizations of the system parameter vector $\mathbf{a}$. At a first sight, this might seem to defeat the purpose of using the proposed ANN-based approach, but actually this is not the case for three main reasons:

- The training set can be generated *offline*. Thus, a much higher complexity can be afforded, and it is not needed to obtain the solution within the channels coherence time.
- The training set can be updated at a *much longer time-scale* compared with the channels coherence time. In other words, the training set can be updated at a much longer time-scale than that at which Problem (P1) should be solved if traditional optimization approaches were used.
- Despite the first two points, it can be argued that Problem (P1) is NP-hard, and thus generating a large training set appears a daunting task even if it can be performed offline. In order to address this issue, the next section develops a novel and efficient BB method which makes it possible to obtain the global solution of (P1) with a complexity affordable by offline implementations.

It is also important to remark that the proposed method does not need to feed the ANN with any initialization power vector, as it would be the case if a suboptimal, iterative power allocation algorithm were emulated. This has allowed us to reduce the size of the ANN input from $L(L+2)$ to $L(L+1)$, further limiting the overall complexity and simplifying the training procedure.

Finally, we explicitly observe that, as anticipated, the proposed ANN-based approach is not restricted to the maximization of the WSEE. Indeed, any power allocation problem can be cast as in (10) and the BB method to be developed in the coming Section IV is not limited to the maximization of the WSEE, but encompasses all major energy-efficient metrics, as addressed in detail in Section IV-A.

IV. GLOBALLY OPTIMAL POWER CONTROL

Due to the fact that Problem (P1) belongs to the class of NP-hard problems, it has in general multiple locally optimal solutions. Thus, traditional optimization approaches like gradient descent or interior-point methods are not able to globally solve (P1). Instead, this section develops a novel BB procedure to solve (P1) with guaranteed global optimality, while at the same time lowering the complexity with respect to BB methods using general-purpose bounds.

The approach is based on successively partitioning the set $[\mathbf{0}, \mathbf{P}] = [0, P_i]^L$ into L-dimensional hyper-rectangles of the form

$$\mathcal{M}^k = \{\mathbf{p} \,:\, r_i^{(k)} \leq p_i \leq s_i^{(k)}, \forall\, i = 1, \ldots, L\} \triangleq [\mathbf{r}^{(k)}, \mathbf{s}^{(k)}] . \tag{14}$$

---

[5]As mentioned, the execution of the training algorithm is conveniently performed by off-the-shelf stochastic gradient descent algorithms, which ensure a fast convergence. Moreover gradient computation is performed by the backpropagation algorithm, which further reduces the computational complexity [14]. The resulting complexity is negligible compared to the generation of the training set.

In each hyper-rectangle $\mathcal{M}^k$, we require an upper-bound $\beta(\mathcal{M}^k)$ of (P1) on $\mathcal{M}^k$, i.e., a power vector $\widetilde{\mathbf{p}}^{(k)} \in \mathcal{M}^k$ feasible for (P1) with WSEE greater than or equal to the WSEE achieved by any other power vector in $\mathcal{M}^k$. In order to ensure convergence to the global optimum, it is essential that $\beta(\mathcal{M}^k) = \infty$ if (P1) is infeasible for all $\mathbf{p} \in \mathcal{M}^k$ and that bounding is consistent with branching, i.e.,

$$\beta(\mathcal{M}^k) - \max_{\mathbf{p} \in \mathcal{M}^k} f(\mathbf{p}) \to 0 \;\; \text{as} \;\; \operatorname{diam} \mathcal{M}^k \to 0, \tag{15}$$

where $f(\mathbf{p})$ is the objective of (P1). Equipped with this notation, the branching procedure operates as follows. In each iteration k, the hyper-rectangle $\mathcal{M}^k = [\mathbf{r}^{(k)}, \mathbf{s}^{(k)}]$ with the best bound is selected and then bisected via $(\mathbf{v}^{(k)}, j_k)$ where

$$\mathbf{v}^{(k)} = \frac{1}{2}(\widetilde{\mathbf{p}}^{(k)} + \mathbf{r}^{(k)}), \quad j_k = \arg\max_j \left| \widetilde{p}_j^{(k)} - r_j^{(k)} \right|. \tag{16}$$

The partition sets are given by the subrectangles $\mathcal{M}_-^k$ and $\mathcal{M}_+^k$ determined by the hyperplane $p_{j_k} = v_{j_k}$ as

$$\mathcal{M}_-^k = \{\mathbf{x} \,|\, r_{j_k}^{(k)} \leq x_{j_k} \leq v_{j_k}^{(k)}, \, r_i^{(k)} \leq x_i \leq s_i^{(k)} \,(i \neq j_k)\}$$
$$\mathcal{M}_+^k = \{\mathbf{x} \,|\, v_{j_k}^{(k)} \leq x_{j_k} \leq s_{j_k}^{(k)}, \, r_i^{(k)} \leq x_i \leq s_i^{(k)} \,(i \neq j_k)\}. \tag{17}$$

Then, given $\mathcal{M}^k = [\mathbf{r}^{(k)}, \mathbf{s}^{(k)}]$, observe that a bound $\beta(\mathcal{M}^k)$ can be found as:

$$\max_{\mathbf{p} \in \mathcal{M}^k} \sum_{i=1}^{L} w_i \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}$$

$$\leq \sum_{i=1}^{L} w_i \max_{\mathbf{p} \in \mathcal{M}^k} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}$$

$$= \sum_{i=1}^{L} w_i \max_{r_i^{(k)} \leq p_i \leq s_i^{(k)}} \underbrace{\frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} r_j^{(k)}}\right)}{\mu_i p_i + P_{c,i}}}_{:= \overline{\mathrm{EE}}_i(p_i, \mathcal{M}^k)} \tag{18}$$

$$=: \beta(\mathcal{M}^k) \tag{19}$$

where the last step is due to EE_i being decreasing in p_j for all $j \neq i$. Thus, we select the bound by maximizing EE_i with respect to $\mathbf{p}$ over $\mathcal{M}^k$. This provides an excellent accuracy-complexity trade-off and leads to fast convergence, as confirmed by the numerical analysis reported in Section VI. At the same time, the bound can be computed with reasonably low computational complexity. Indeed, $\overline{\mathrm{EE}}_i(p_i, \mathcal{M}^k)$ is a strictly pseudo-concave function of p_i, being the ratio of a strictly concave over an affine function [4]. Thus, its global maximizer is obtained as the unique zero of its derivative, namely the unique solution of the equation:

$$\frac{\alpha_i(\mu_i p_i + P_{c,i})}{1 + \alpha_i p_i + \sum_{j \neq i} \beta_{i,j} r_j^{(k)}} = \mu_i \ln\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} r_j^{(k)}}\right), \tag{20}$$

which is denoted by $\widehat{p}_i^{(k)}$. Equation (20) can be solved numerically with any root finding algorithm, e.g. with Newton-Raphson's or Halley's method. However, due to $\frac{d}{dp_i}\overline{\mathrm{EE}}_i$ approaching zero quickly as $p_i \to \infty$ these methods might suffer

from numerical problems. Instead, a more stable numerical solution of (20) is seen to be computed as

$$\widetilde{p}_i^{(k)} = \frac{1}{\psi_i} \left(\frac{\frac{\psi_i}{\mu_i} P_{c,i} - 1}{W_0 \left(\left(\frac{\psi_i}{\mu_i} P_{c,i} - 1 \right) e^{-1} \right)} - 1 \right), \quad (21)$$

where $\psi_i = \frac{\alpha_i}{1+\sum_{j \neq i} \beta_{i,j} r_j^{(k)}}$ and $W_0(\cdot)$ is the principal branch of the Lambert W function.

The final procedure is stated in Algorithm 1. The set $\mathscr{R}$ holds all hyper-rectangles to be examined and γ is the current best value (CBV). In lines 3–5, the box $\mathcal{M}^k$ with the best bound is selected and bisected. Later, in line 16, this hyper-rectangle is removed from $\mathscr{R}$. Each of the resulting subrectangles is examined in lines 6–15. If its bound is not better than the CBV (plus the absolute tolerance ε), it is ignored. Otherwise, it is added to $\mathscr{R}$ in line 16. Then, the algorithm selects a feasible point from $\mathcal{M}^k$ and updates the current best solution (CBS) if the objective value for this point is better than the CBV. When $\mathscr{R}$ is empty, there are no further hyper-rectangles to examine and the problem is solved.

Algorithm 1 Global optimal solution of (P1)

1: Initialize $\mathscr{R}_0 = \{[\mathbf{0}, \mathbf{P}]\}$, $\gamma = -\infty$, $k = 0$

2: **repeat**
3:     Select $\mathcal{M}^k \in \arg\max\{\beta(\mathcal{M}) \,|\, \mathcal{M} \in \mathscr{R}_k\}$.
4:     Bisect $\mathcal{M}^k$ via $(\mathbf{v}^k, j_k)$ with $(\mathbf{v}^k, j_k)$ as in (16) and
5:     Let $\mathscr{P}_k = \{\mathcal{M}_-^k, \mathcal{M}_+^k\}$ with $\mathcal{M}_-^k, \mathcal{M}_+^k$ as in (17).

6:     **for all** $\mathcal{M} \in \mathscr{P}_k$ **do**
7:         **if** $\beta(\mathcal{M}) > \gamma + \varepsilon$ **then**
8:             Add $\mathcal{M}$ to $\mathscr{S}_k$.
9:             Let $\mathbf{r}$ such that $\mathcal{M} = [\mathbf{r}, \mathbf{s}]$.
10:            **if** $f(\mathbf{r}) > \gamma$ **then**
11:                $\bar{\mathbf{p}} \leftarrow \mathbf{r}$
12:                $\gamma \leftarrow f(\mathbf{r})$
13:            **end if**
14:         **end if**
15:     **end for**

16:     $\mathscr{R}_{k+1} \leftarrow \mathscr{S}_k \cup \mathscr{R}_k \setminus \{\mathcal{M}^k\}$
17:     $k \leftarrow k + 1$
18: **until** $\mathscr{R}_k = \emptyset$

19: **return** $\bar{\mathbf{p}}$ as the optimal solution

Convergence of Algorithm 1 is stated formally below.

Proposition 2: For every $\varepsilon > 0$, Algorithm 1 converges in a finite number of iterations towards a point with objective value within an ε-region of the global optimal value of (P1).

Proof: First, observe that the branching procedure does not generate any box containing infeasible points. Hence, no feasibility checks are necessary during Algorithm 1.

After each iteration, $\mathcal{R}_k$ contains all boxes that might hold a better solution than the CBS $\bar{\mathbf{p}}$. If the algorithm terminates, then all boxes $\mathcal{M}$ generated from $\mathcal{R}_k$ after the last update of γ had a bound $\beta(\mathcal{M}) \leq \gamma + \varepsilon$. Since $\bar{\mathbf{p}}$ is feasible and satisfies $f(\bar{\mathbf{p}}) > \beta(\mathcal{M}) - \varepsilon \geq \max_{\mathbf{p} \in \mathcal{M}} f(\mathbf{p}) - \varepsilon$ for every $\mathcal{M}$, $\bar{\mathbf{p}}$ is a global ε-optimal solution. Thus, it remains to show that the algorithm is finite, i.e., that the termination criterion $\mathcal{R}_k = \emptyset$ occurs at some point. This is equivalent to showing that the

bound approaches the optimal value of (P1). To this end, we leverage [31, Prop. 6.2], which ensures the convergence with the employed bisection rule. Indeed, by virtue of [31, Prop. 6.2], there exists a subsequence $\{k_v\}_v$ such that $\mathbf{r}^{(k_v)}$ and $\widetilde{\mathbf{p}}^{(k_v)}$ approach a common limit $\mathbf{x}$ as $v \to \infty$. Then,

$$f(\mathbf{r}^{(k_v)}) - \beta(\mathcal{M}^{k_v})$$
$$= \sum_{i=1}^{L} w_i \overline{\mathrm{EE}}_i(r_i^{(k_v)}, \mathcal{M}^{k_v}) - \sum_{i=1}^{L} w_i \overline{\mathrm{EE}}_i(\widetilde{p}_i^{(k_v)}, \mathcal{M}^{k_v}) \to 0,$$

and, thus, $\beta(\mathcal{M}^{k_v}) \to f(\mathbf{x})$. Moreover, this point is feasible for (P1) and, hence, $\beta(\mathcal{M}^{k_v}) \geq f(\mathbf{x})$. Therefore, $\beta(\mathcal{M}^{k_v}) \to f(\mathbf{p}^*)$ (with $\mathbf{p}^*$ the optimal solution of (P1)) as $v \to \infty$. ∎

The following remarks are in order.

Remark 1: Inspecting Algorithm 1 it can be seen that each step towards the computation of the optimal power allocation is continuous with respect to the channel parameters $\{\alpha_i, \beta_{i,j}\}_{i,j}$, with the exception of (21), which has a discontinuity in $\alpha_i = 0$. Therefore, in order to claim the continuity of the map (10), that is required by Proposition 1, it is necessary to assume that α_i is bounded below by a strictly positive quantity, i.e. $\alpha_i > \omega_i$, for some $\omega_i > 0$ and $i = 1, \ldots, L$. This assumption does not seem restrictive for any practical system, recalling that α_i is the channel-to-noise ratio between the i-th UE and the associated BS.

Remark 2: If it is desired to use a relative tolerance instead of an absolute tolerance, it is sufficient to replace "$\beta(\mathcal{M}) > \gamma + \varepsilon$" by "$\beta(\mathcal{M}) > (1 + \varepsilon)\gamma$" in line 7 of Algorithm 1.

A. Application to other Energy Efficiency Metrics

This section shows how to apply the proposed framework to other performance metrics. It was already observed in Section II-A that the WSEE generalizes the system weighted sum-rate defined in (6), and thus the proposed approach naturally applies to the maximization of the weighted sum-rate with $\widetilde{\mathbf{p}}^{(k)} = \mathbf{s}^{(k)}$ as can be easily seen from (18). Instead, the application of our framework to other EE metrics requires more effort. Specifically, we will consider the WPEE, WMEE, and GEE functions, as defined in (7), (8), and (9), respectively.

1) WPEE and WMEE maximization: Let us consider the WPEE function. The main issue regarding the applicability of the proposed BB procedure is the derivation of a bound for the WPEE over the generic hyper-rectangle $\mathcal{M}^k$, that is tight, consistent with branching, and simple to maximize. To this end, observe that we can write:

$$\max_{\mathbf{p} \in \mathcal{M}^k} \prod_{i=1}^{L} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}$$

$$\leq \prod_{i=1}^{L} \max_{\mathbf{p} \in \mathcal{M}^k} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{\mu_i p_i + P_{c,i}}$$

$$= \prod_{i=1}^{L} \max_{r_i^{(k)} \leq p_i \leq s_i^{(k)}} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} r_j^{(k)}}\right)}{\mu_i p_i + P_{c,i}} \quad (22)$$

$$=: \beta(\mathcal{M}^k). \quad (23)$$

Note that, for any $i = 1, \ldots, L$, the upper-bound $\beta(\mathcal{M}^k)$ in (22) coincides with that obtained for the WSEE function in (18). Thus, the same approach used for WSEE maximization applies also to the maximization of the WPEE.

Moreover, it can be seen that the same bounding technique applies also to the WMEE function. Indeed, all steps above can be made also if the product of the EEs is replaced by the minimum of the EEs. Indeed, both are increasing functions of the individual EEs and no differentiability assumption is required by the proposed BB procedure.

2) GEE maximization: The GEE function does not explicitly depend on the individual EEs, which slightly complicates the bounding technique. Nevertheless, a similar approach applies. Defining $P_c = \sum_{i=1}^{L} P_{c,i}$, it holds

$$
\max_{\boldsymbol{p} \in \mathcal{M}^k} \frac{\sum_{i=1}^{L} \log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{P_c + \sum_{i=1}^{L} \mu_i p_i}
$$

$$
\leq \sum_{i=1}^{L} \max_{\boldsymbol{p} \in \mathcal{M}^k} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{P_c + \sum_{i=1}^{L} \mu_i p_i}
$$

$$
= \sum_{i=1}^{L} \max_{s_i^{(k)} \leq p_i \leq r_i^{(k)}} \frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} r_j^{(k)}}\right)}{\mu_i p_i + P_c + \sum_{j \neq i} \mu_j r_j^{(k)}} \quad (24)
$$

$$
=: \beta(\mathcal{M}^k), \quad (25)
$$

where the last step is due to $\frac{\log\left(1 + \frac{\alpha_i p_i}{1 + \sum_{j \neq i} \beta_{i,j} p_j}\right)}{P_c + \sum_{\ell=1}^{L} \mu_\ell p_\ell}$ being decreasing[6] in p_j for all $j \neq i$. Although slightly different from that obtained for the WSEE, WPEE, and WMEE, the bound in (24) is formally equivalent to (18) as a function of p_i. Thus, the same maximization procedure applies to GEE maximization, too.

V. FIRST-ORDER OPTIMAL POWER CONTROL

This section is devoted to developing a power control algorithm with guaranteed convergence to a first-order optimal point of Problem (P1). Besides providing an alternative power control approach, the method developed here also provides a theoretically solid benchmark for the proposed ANN-based algorithm developed in Section III.

The algorithm proposed in this section is inspired by the successive pseudo-concave framework from [32], which tackles the maximization of a function f by maximizing a sequence of approximate functions $\{\widetilde{f}_t\}_t$ fulfilling the following properties for all j:

1) $\widetilde{f}_t(\boldsymbol{p}; \boldsymbol{p}^{(t)})$ is pseudo-concave in $\boldsymbol{p}$ for any feasible $\boldsymbol{p}^{(t)}$.
2) $\widetilde{f}_t(\boldsymbol{p}; \boldsymbol{p}^{(t)})$ is continuously differentiable in $\boldsymbol{p}$ for any feasible $\boldsymbol{p}^{(t)}$ and continuous in $\boldsymbol{p}^{(t)}$ for any feasible $\boldsymbol{p}$.
3) $\nabla_{\boldsymbol{p}} \widetilde{f}_t(\boldsymbol{p}^{(t)}; \boldsymbol{p}^{(t)}) = \nabla_{\boldsymbol{p}} f(\boldsymbol{p}^{(t)})$.
4) $\widetilde{f}_t(\boldsymbol{p}; \boldsymbol{p}^{(t)})$ has a non-empty set of maximizers in the feasible set.
5) Given any convergent subsequence $\{\boldsymbol{p}^{(t)}\}_{t \in \mathcal{T}}$ where $\mathcal{T} \subseteq \{1, 2, \ldots\}$, the sequence $\{\arg\max_{\boldsymbol{p} \in \mathcal{P}} \widetilde{f}_t(\boldsymbol{p}; \boldsymbol{p}^{(t)})\}_{t \in \mathcal{T}}$ is bounded.

[6]This can be verified easily from its first-order derivative.

It is seen that the approximate functions are parametrized by $\boldsymbol{p}^{(t)}$ and the properties above need to hold for any feasible $\boldsymbol{p}^{(t)}$. In practice, $\boldsymbol{p}^{(t)}$ is updated according to a specific rule after each iteration, as it will be explained in the sequel.

Under the assumptions above, [32] shows that every limit point of the sequence $\boldsymbol{p}_t^\star$ of maximizers of $\widetilde{f}_t$ with respect to $\boldsymbol{p}$, converges to a first-order optimal point for the original problem of maximizing f. Moreover, due to the first property above, for all t, the maximization of $\widetilde{f}_t$ can be accomplished with polynomial complexity by standard optimization methods.[7] Thus, the crucial point when employing the above framework is about finding suitable approximate functions $\{\widetilde{f}_t\}_t$ that fulfill all above assumptions. The rest of this section shows how this can be accomplished for the WSEE maximization problem (P1).

To elaborate, recalling (3), and for a given point $\boldsymbol{p}^{(t)}$, we propose the following approximation for the EE of user i in (4),

$$
w_i \text{EE}_i \approx \widetilde{\text{EE}}_i(\boldsymbol{p}; \boldsymbol{p}^{(t)}) = w_i \frac{R_i(p_i, \boldsymbol{p}_{-i}^{(t)})}{\mu_i p_i^{(t)} + P_{c,i}} + (p_i - p_i^{(t)})
$$
$$
\cdot \left(w_i \frac{-\mu_i R_i(\boldsymbol{p}^{(t)})}{(\mu_i p_i^{(t)} + P_{c,i})^2} + \sum_{j \neq i} w_j \frac{\frac{\partial}{\partial p_i} R_j(\boldsymbol{p}^{(t)})}{\mu_i p_i^{(t)} + P_{c,i}} \right), \quad (26)
$$

wherein $R_i(p_i, \boldsymbol{p}_{-i}^{(t)})$ denotes the i-th user's rate as a function of the i-th user's power p_i, while all other powers are fixed to $\boldsymbol{p}_{-i}^{(t)} = \{p_j^{(t)}\}_{j \neq i}$.

In the Appendix it is shown that the approximation (26) fulfills all five properties required by the successive pseudo-concave approximation framework. Here, we remark that, for all $i = 1, \ldots, L$, the approximate function in (26) is constructed in such a way to be concave with respect to p_i, since the i-th user's rate R_i is concave in p_i, while the other terms in the EE have been linearized around $\boldsymbol{p} = \boldsymbol{p}^{(t)}$. Replacing each EE in the WSEE maximization problem (P1) by (26), we obtain the approximate problem:

$$
\begin{cases} \max_{\boldsymbol{p}} & \sum_{i=1}^{L} \widetilde{\text{EE}}_i(\boldsymbol{p}; \boldsymbol{p}^{(t)}) \\ \text{s.t.} & 0 \leq p_i \leq P_i, \quad \text{for all } i = 1, 2, \ldots, L, \end{cases} \quad \text{(P3)}
$$

which is a concave maximization problem and can therefore be solved in polynomial time using standard convex optimization tools [33]. Moreover, since $\widetilde{\text{EE}}_i$ depends only on p_i for all i, Problem (P3) can be decoupled over the users, and each transmit power p_i can be optimized separately by solving the scalar problem:

$$
\begin{cases} \max_{p_i} & \widetilde{\text{EE}}_i(p_i; \boldsymbol{p}^{(t)}) \\ \text{s.t.} & 0 \leq p_i \leq P_i, \end{cases} \quad \text{(P4)}
$$

Thus, by the successive pseudo-concave optimization framework, the original Problem (P1) is tackled by solving a sequence

[7]Recall that pseudo-concave functions are differentiable by definition and every stationary point is a global maximizer.

of problems of the form of (P3), updating the point $\boldsymbol{p}^{(t)}$ after each iteration according to the formula

$$\boldsymbol{p}^{(t+1)} = \boldsymbol{p}^{(t)} + \gamma^{(t)}(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)}), \qquad (27)$$

with $\mathbb{B}\boldsymbol{p}^t$ an optimal solution of (P3) and $\gamma^t = \beta^{m_t}$ to be determined by the Armijo rule, where m^t is the smallest nonnegative integer such that

$$f(\boldsymbol{p}^{(t)} + \beta^{m_t}(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)})) \geq f(\boldsymbol{p}^{(t)}) + \alpha\beta^{m_t}\nabla f(\boldsymbol{x}^{(t)})^T(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)})$$

with $0 < \alpha < 1$ and $0 < \beta < 1$ being scalar constants.[8]

The overall procedure is summarized in Algorithm 2, whose convergence is formally stated below.

Proposition 3: Any limit point $\boldsymbol{p}_t^\star$ of $\{\boldsymbol{p}^{(t)}\}_t$ obtained by Algorithm 2 is a stationary point of (P1).

Proof: Please refer to the Appendix. ∎

Algorithm 2 Successive convex approximation algorithm

1: Initialize $t = 0$, $\boldsymbol{p}^{(0)} \in [\mathbf{0}, \boldsymbol{P}]$, $\alpha, \beta \in (0,1)$.
2: **repeat**
3: $\mathbb{B}_i\boldsymbol{p}^{(t)} \leftarrow \arg\max_{0 \leq p_i \leq P_i} \widetilde{r}_i(p_i; \boldsymbol{p}^{(t)}), \quad i = 1, 2, \ldots, L.$
4: $\gamma^{(t)} \leftarrow 1$
5: **while** $f(\boldsymbol{p}^{(t)} + \gamma^{(t)}(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)})) < f(\boldsymbol{p}^{(t)}) +$
       $\alpha\gamma^{(t)}\nabla f(\boldsymbol{p}^{(t)})^T(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)})$ **do**
6:   $\gamma^{(t)} \leftarrow \beta\gamma^{(t)}$
7: **end while**
8: $\boldsymbol{p}^{(t+1)} \leftarrow \boldsymbol{p}^{(t)} + \gamma^{(t)}(\mathbb{B}\boldsymbol{p}^{(t)} - \boldsymbol{p}^{(t)})$
9: $t \leftarrow t + 1$
10: **until** convergence.

Remark 3: It can be seen that, despite requiring only the solution of convex problems in each iteration, the iterative Algorithm 2 is more complex than the ANN-based algorithm from Section III, which basically provides a closed-form expression for optimal power allocation as a function of the system propagation channels. Despite the higher complexity, the numerical performance assessment presented in the following Section VI shows that Algorithm 2 is outperformed by the proposed ANN-based algorithm.

VI. NUMERICAL EVALUATION

We consider the uplink of a wireless interference network in which $L = 4$ single-antenna UEs are placed in a square area with edge $2\,\mathrm{km}$ and communicate with 4 access points placed at coordinates $(0.5, 0.5)\,\mathrm{km}$, $(0.5, 1.5)\,\mathrm{km}$, $(1.5, 0.5)\,\mathrm{km}$, $(1.5, 1.5)\,\mathrm{km}$, and equipped with $n_R = 2$ antennas each. The path-loss is modeled following [35], with carrier frequency $1.8\,\mathrm{GHz}$ and power decay factor equal to 4.5, while fast fading terms are modeled as realizations of zero-mean, unit-variance circularly symmetric complex Gaussian random variables. The circuit power consumption and power amplifier inefficiency terms are equal to $P_{c,i} = 1\,\mathrm{W}$ and $\mu_i = 4$ for all $i = 1, \ldots, L$, respectively. The noise power at each receiver is generated as $\sigma^2 = F\mathcal{N}_0 B$, wherein $F = 3\,\mathrm{dB}$ is the receiver noise figure, $B = 180\,\mathrm{kHz}$ is the communication bandwidth, and $\mathcal{N}_0 = -174\,\mathrm{dBm/Hz}$ is the noise spectral density. All users have the same maximum transmit powers $P_1, \ldots, P_L = P_{\max}$.

---

[8]Please refer to [34, Sec. 1.2.1] for more details on the Armijo rule and how to choose α and β properly.

The proposed ANN-based solution of Problem (P1) is implemented through a feedforward ANN with $K + 1$ fully-connected layers, with the $K = 5$ hidden layers having 128, 64, 32, 16, 8 neurons, respectively. In order to generate the training set, Problem (P1) needs to be solved for different realizations of the vector $\tilde{\boldsymbol{a}} = (\tilde{\alpha}_i, \tilde{\beta}_{i,j}, P_{\max})_{i,j}$. When doing this, the use of realistic numbers for the receive noise power and propagation channels might lead to coefficients $\{\tilde{\alpha}_i, \tilde{\beta}_{i,j}\}_{i,j}$ with quite a large magnitude, which can cause numerical problems to the gradient-descent training algorithm. We have observed that this issue is solved by expressing the parameter vectors $\tilde{\boldsymbol{a}}$ in the training set in logarithmic units rather than in a linear scale. A similar problem occurs for the output powers, which in some cases might be close to zero due to the normalization by $P_{\max}$. This issue is also resolved by expressing the output powers in logarithmic scale. On the other hand, logarithms cause numerical problems when the optimal transmit powers are very close to zero. In order to avoid this issue, a suitable approach is to clip logarithmic values approaching $-\infty$ at $-M$ for $M > 0$. In our experiments, $M = 20$ worked well.[9] Thus, the considered normalized training set is

$$\mathcal{S}_T = \{(\log_{10}\tilde{\boldsymbol{a}}_n, \max\{-20, \log_{10}\tilde{\boldsymbol{p}}_n^*\}) \mid n = 1, \ldots, N_T\},$$

where all functions are applied element-wise to the vectors in the training set.

As for the activation functions, ReLU and its generalizations are the most widely used choice. Our experiments verify that they also perform well in this application. Specifically, the first hidden layer has an exponential linear unit (ELU) activation, motivated by the need to compensate for the logarithmic conversion in the training set. This choice, together with the logarithmic normalization of the data set, has proven itself essential for good training performance. Next, the other hidden layers alternate ReLU and ELU activation functions, while the output layer deploys a linear activation function. The use of a linear activation in the output layer seems to contrast with the fact that the transmit powers need to be constrained in the interval $[0, 1]$. However, enforcing this constraint directly in the output activation function might mislead the ANN. Indeed, it could lead to low training errors simply thanks to the use of cut-off levels in the activation function, instead of being the result of proper adjustment of the hidden layer weights and biases. In this case, the ANN would not be able to learn that the training and validation errors are acceptable only because the desired power level is close to either 1 or 0, and the clipping at the output layer provides by construction such a power level, regardless of the configuration adopted in the hidden layers. Instead, a linear output activation function allows the ANN to learn whether the present configuration of weights and biases is truly leading to a small error.

A. Training Performance

The ANN is implemented in Keras 2.2.4 [36] with TensorFlow 1.12.0 [37] as backend. Training is performed on a

---

[9]Note that, although using a logarithmic scale, the transmit powers are not expressed in dBW, since the logarithmic values are not multiplied by 10. Thus $-M = -20$, corresponds to -200 dBW.

Nvidia GeForce GTX 1080 Ti over 100 epochs with batches of size 128 and shuffling of the training data before each epoch. Initialization is performed by Keras with default parameters, i.e., Glorot uniform initialization [38] for the kernel and zero biases. The optimization problem (13) is solved by the Adam optimizer with Nesterov momentum [39], initialized by the default parameters from Keras, and using the squared error as the loss function in (13). Source code and data sets are available from https://github.com/bmatthiesen/deep-EE-opt.

The training set is generated from 2,000 independent and identically distributed (i.i.d.) realizations of UEs' positions and propagation channels. Each user is randomly placed in the service area and channels are then generated according to the channel model described above. Each UE i is associated to the access point towards which it enjoys the strongest effective channel α_i. For each channel realization, we apply Algorithm 1 to solve (P1) for $P_{\max} = -30, \ldots, 20$ dB in 1 dB steps with relative tolerance $\varepsilon = 0.01$. This yields a training set of 102,000 samples.

Besides the training set, also a validation set and a test set are required. The validation set is used during training to estimate the generalization performance of the ANN, i.e., the performance on a data set the ANN was not trained on. The validation loss is the central metric for hyperparameter tuning, i.e., choosing all parameters of the ANN other than weights and biases (e.g. number of layers, activation functions, batch size). Since during this process information about the validation set leaks into the ANN model, another set for the final testing of the ANN is required, the test set. It is essential that the test set is never used during training and tuning of the ANN [40]. The validation and test sets have been independently generated from 200 and 10,000 i.i.d. channel realizations, respectively, with the same procedure used for the training set. This results in 10,200 samples for the validation set, i.e. 10% of the training set, and 510,000 samples for the test set. The final performance to be shown in Section VI-B will be averaged over the test set samples. Thus, using a test set based on 10,000 channel scenarios means that the performance presented in Section VI-B is what is obtained by using the trained ANN for 10,000 channel coherence times. This confirms that the training phase needs to be performed only sporadically, as argued in Section III-A.

Considering training, validation, and test sets, 622,200 data samples were generated, which required solving the NP-hard problem (P1) 622,200 times. This has been accomplished by the newly proposed BB method developed in Algorithm 1, which has been implemented in C++ employing the Intel® MKL, OpenMP, and the Lambert W library published at https://github.com/CzB404/lambert_w. Computing the complete data set (622,200 samples including training, validation, and test sets) took 8.4 CPU hours on Intel Haswell nodes with Xeon E5-2680 v3 CPUs running at 2.50 GHz. The mean and median times per sample are 48.7 ms and 4.8 ms, respectively, which shows the effectiveness of the proposed Algorithm 1, and in turn supports the argument that the offline generation of a suitable training set for the proposed ANN-based power control method is quite affordable.

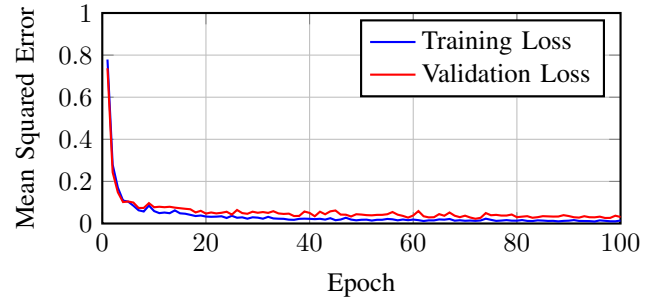Due to the random initialization, shuffling of the training data,

Fig. 2. Training and validation loss.

and the inherent randomness of the optimizer, the weights and biases of the ANN are realizations of a random process. Thus, all performance results reported for the ANN are averaged over 10 realizations of the network obtained by training the ANN *on the same training set* with different initialization of the underlying random number generator.[10] The average training and validation losses for the final ANN are shown in Fig. 2. It can be observed that both errors quickly approach a very small value that is of the same order of magnitude as the tolerance of Algorithm 1. Moreover, neither of the losses increases over time which leads to the conclusion, that the adopted training procedure fits the training data well, without underfitting or overfitting.

B. Testing Performance

The average performance of the final ANN on the test set is reported in Fig. 3. Recall that this test set is never used during training and, thus, the ANN has no information about it except for its statistical properties gathered from the training set (and, possibly, the validation set due to hyperparameter tuning). It can be seen from Fig. 3 that the gap to the optimal value is virtually non-existent which is confirmed by the empirical cumulative distribution function (CDF) of the relative approximation error displayed in Fig. 4. Its mean and median values are 0.0133 and 0.00739 respectively.

In addition to near-optimal approximation performance and low computational complexity, the proposed ANN-based approach also outperforms several baseline approaches. Specifically, we have included a comparison with the following benchmarks:

- **SCAos:** The first-order optimal method based on the sequential convex approximation method developed in Section V. For each value of $P_{\max}$, the algorithm initializes the transmit power to $p_i = P_{\max}$, for all $i = 1, \ldots, L$.
- **SCA:** This is again the first-order optimal method based on sequential convex approximation developed in Section V, but in this case a double-initialization approach is used. Specifically, at $P_{\max} = -30$ dBW once again maximum power initialization is used. However, for all values of $P_{\max} > -30$ dBW, the algorithm is run twice, once with the maximum power initialization, and once initializing the transmit powers with the optimal solution obtained

---

[10]Note that this is not equivalent to *model ensembling* [40, Sect. 7.3.3] or *bagging* [14, Sect. 7.1].
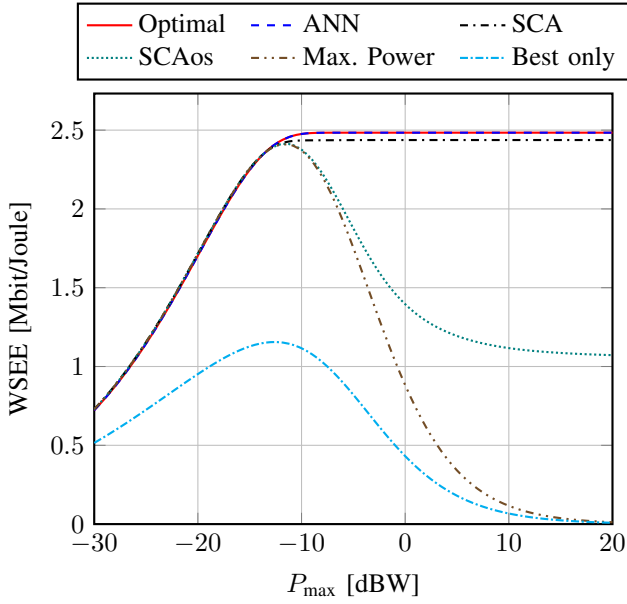
Fig. 3. Performance on the test set compared to the global optimal solution, first-order optimal solutions, and fixed power allocations.
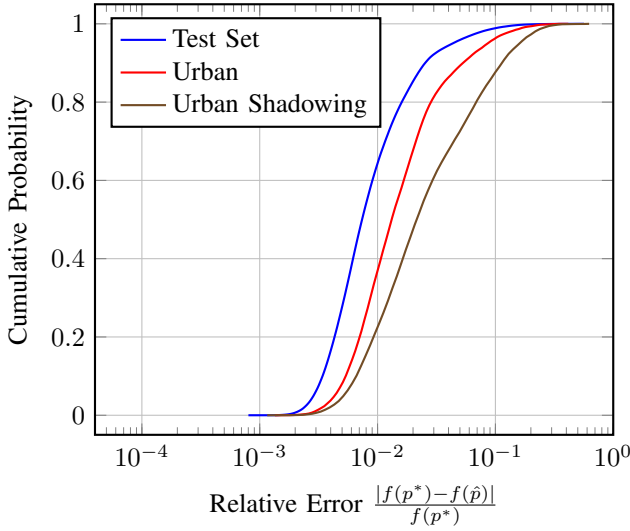

Fig. 4. Empirical CDF of the relative approximation error.

for the previous $P_{\max}$ value. Then, the power allocation achieving the better WSEE value is retained.

- **Max. Power:** All UEs transmit at maximum power, i.e. $p_i = P_{\max}$, for all $i = 1, \ldots, L$. This strategy is known to perform well in interference networks for low $P_{\max}$ values.
- **Best only:** Only one UE is allowed to transmit, specifically that with the best effective channel. This approach is motivated for high $P_{\max}$ values, as a naive way of nulling out multi-user interference.

The results show that the proposed ANN-based approach outperforms all benchmarks. The only benchmark that performs comparably with the ANN-based approach is the SCA approach. However, as described above, this method relies on a sophisticated initialization rule, which requires to solve the WSEE maximization problem twice and for the complete range of $P_{\max}$ values. This is clearly not suitable for obtaining a "one-shot" solution, i.e. when the WSEE needs to be maximized

only for one specific value of $P_{\max}$. Moreover, it requires some calibration depending on the channel statistics since it performs well provided the $P_{\max}$ range is sufficiently far away from the WSEE saturation region, i.e. the range of $P_{\max}$ values for which the WSEE keeps constant, starting from $P_{\max} \approx -10\,\mathrm{dBW}$ in Fig. 3. Thus, the SCA approach has a quite higher complexity than the ANN-based method, but, despite this, it performs slightly worse. In conclusion, we can argue that the ANN approach is much better suited to online power allocation than state-of-the-art approaches, including Algorithm 2.

Previous results consider a test set whose samples are independently generated from the training and validation sets, but following the same statistical distribution. Instead, now we wish to analyze how robust the ANN performance is to changes in the channel statistics. To this end, in the following we consider a new test set, whose samples are generated according to a different statistical distribution. Specifically, we generate path-loss effects according to the Hata-COST231 propagation model [41], [42] for urban (non-metropolitan) areas with carrier frequency 1.9 GHz and base station height 30 m. However, we do not repeat the training based on the new channel model, but instead use again the same ANN trained as described above. Remarkably, as indicated by the prediction performance reported in Fig. 5 and verified by the distribution of the relative error in Fig. 4 under the label "Urban," the performance is still quite good and degrades only slightly compared to the case in which the test set and training set contain samples taken from the same distribution.

Next, we further modify the channel generation procedure of the test set, by also introducing a log-normal shadowing [42] with 8 dB standard deviation. It can be observed from Fig. 4 that the median relative error increases by roughly half an order of magnitude. Given that the underlying channel distribution is quite different from the trained channel model, the performance can still be considered good. This is also verified by the WSEE performance shown in Fig. 5. Indeed, the performance is still better than it could be expected by the SCAos method. Based on these observations, we can conclude that training the ANN on synthetic data based on simple channel models is quite robust and performs well also in more sophisticated channels scenarios. Of course, the performance tends to degrade as the difference between the distributions of the training and test set increases.

C. Performance of an ANN with reduced size

We conclude this section by evaluating the performance of a much smaller ANN trained with the same data as before. Specifically, we only consider 2 hidden layers having 16 and 8 neurons, respectively, with activation functions ELU and ReLU. Again, the output layer has 4 nodes and a linear activation function. Training is performed in batches of size 128. While the counterpart of Fig. 3 looks identical (and is, therefore, not reproduced), the difference between the two ANNs is best studied from the training loss in Fig. 6 and the distribution of the relative error in Fig. 7. First, observe from Fig. 6 that the training and validation losses stall on a value clearly greater than those of the original (larger) ANN. The CDF of the relative
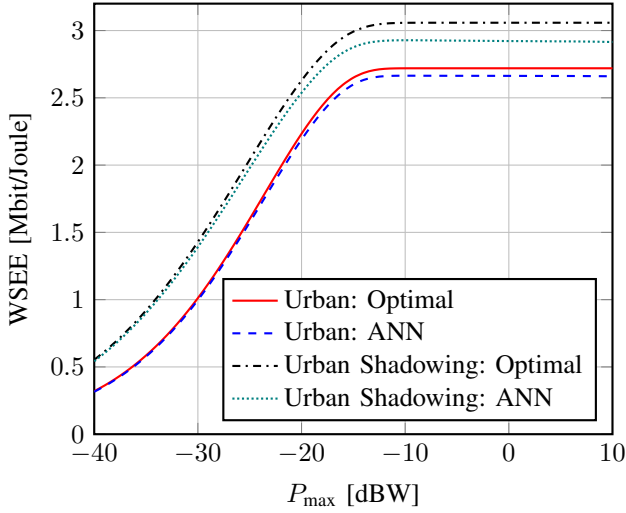
Fig. 5. Performance on different channel distributions: Hata-COST231 Urban propagation model with and without 8 dB shadowing.
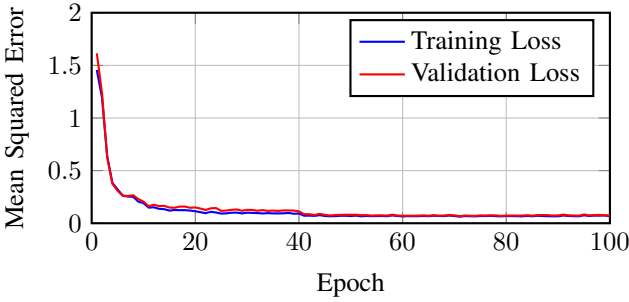

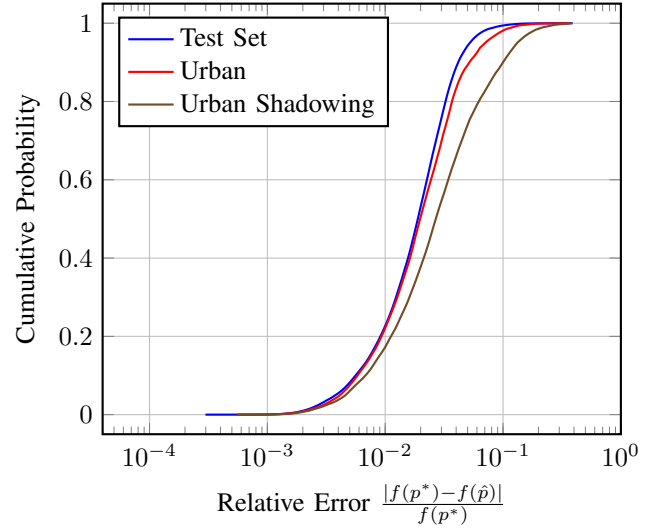Fig. 6. Training and validation loss of the smaller ANN.

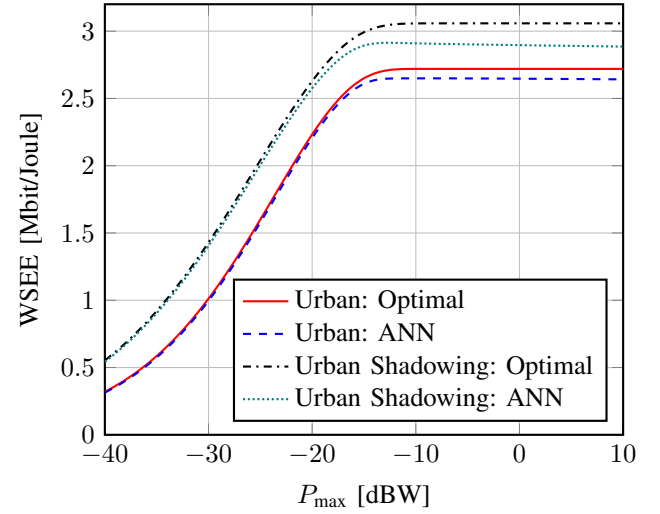Fig. 7. Empirical CDF of the relative approximation error made by the smaller ANN.

Fig. 8. Performance of the smaller ANN on different channel distributions: Hata-COST231 Urban propagation model with and without 8 dB shadowing.

error reflects this as well, being shifted significantly to the right. Still, the mean error on the test set is so small that no difference would be observed in terms of achieved WSEE value. Instead, the performance on the test set generated from the Hata-COST231 Urban model with shadowing differs noticeably from the original ANN as can be seen from Fig. 8. However, the performance can still be considered good. Thus, although the smaller ANN has worse performance than the original, the practical implications are limited and its reduced complexity might be worth the downsides.

VII. CONCLUSIONS

This work has developed an ANN-based power control framework for EE maximization in wireless interference networks. The proposed approach shows an extremely low-complexity, essentially providing a closed-form expression for the optimal power allocation as a function of system and problem parameters. Despite the very low complexity, the resulting performance is virtually optimal, thanks to the fact that the ANN can be trained directly on the globally optimal power allocation. This is made possible by the development of a novel branch-and-bound procedure that is able to efficiently compute the globally optimal energy-efficient power allocation rule. Moreover, the proposed ANN-based method outperforms other competing solutions, including a newly proposed first-order optimal power allocation method.

In addition, the numerical analysis shows that the ANN performance remains stable also when averaged over many test samples, thus showing that the training phase needs to be performed only sporadically. Also, the proposed ANN-based approach exhibits a remarkable resilience to mismatches between the statistical distribution of the training and test sets. Furthermore, it still features satisfactory performance when only a relatively small number of neurons and layers are deployed.

APPENDIX A
PROOF OF THEOREM 3

Problem (P1) has a closed convex feasible set and its objective is a proper, continuously differentiable function. Thus, the result follows from [32, Thm. 1] if we can show that (26) satisfies all five technical conditions stated in Section V.

Conditions 1) and 2) are clearly satisfied, and the boundedness of the feasible set $[0, P]$ is sufficient for conditions 4) and 5) to hold [32].

Condition 3) is equivalent to

$$\frac{\partial}{\partial p_k}\widetilde{\mathrm{EE}}_i(\boldsymbol{p};\boldsymbol{p}^{(t)})\big|_{\boldsymbol{p}=\boldsymbol{p}^{(t)}} = \frac{\partial}{\partial p_k}\mathrm{EE}_i(\boldsymbol{p})\big|_{\boldsymbol{p}=\boldsymbol{p}^{(t)}}$$

for all $k = 1, \ldots, K$. The left-hand side (LHS) can be expressed as

$$
\frac{\partial}{\partial p_i} \widetilde{\mathrm{EE}}_i(\boldsymbol{p}; \boldsymbol{p}^{(t)})\big|_{\boldsymbol{p}=\boldsymbol{p}^{(t)}}
$$

$$
= \left[w_i \frac{\partial}{\partial p_i} \frac{R_i(p_i, \boldsymbol{p}_{-i}^{(t)})}{\phi_i q_i + P_{c,i}} - w_i \frac{\phi_i R_i(\boldsymbol{p}^{(t)})}{(\phi_i q_i + P_{c,i})^2} \right.
$$

$$
\left. + \sum_{j \neq i} w_j \frac{\frac{\partial}{\partial p_i} R_j(\boldsymbol{p}^{(t)})}{\phi_j q_j + P_{c,j}} \right]_{\boldsymbol{p}=\boldsymbol{p}^{(t)}}
$$

$$
= w_i \left(\frac{\frac{\partial}{\partial p_i} R_i(\boldsymbol{p}^{(t)})}{\phi_i q_i + P_{c,i}} - \frac{\phi_i R_i(\boldsymbol{p}^{(t)})}{(\phi_i q_i + P_{c,i})^2} \right)
$$

$$
+ \sum_{j \neq i} w_j \frac{\frac{\partial}{\partial p_i} R_j(\boldsymbol{p}^{(t)})}{\phi_j q_j + P_{c,j}}. \tag{28}
$$

The right-hand side (RHS) can be expressed as

$$
\frac{\partial}{\partial p_i} \mathrm{EE}_i(\boldsymbol{p}) = w_i \frac{\partial}{\partial p_i} \left(\frac{R_i(\boldsymbol{p})}{\phi_i p_i + P_{c,i}} \right) + \sum_{j \neq i} w_j \frac{\frac{\partial}{\partial p_i} R_j(\boldsymbol{p})}{\phi_j p_j + P_{c,j}}
$$

$$
= w_i \left(\frac{\frac{\partial}{\partial p_i} R_i(\boldsymbol{p})}{\phi_i p_i + P_{c,i}} - \frac{\phi_i R_i(\boldsymbol{p})}{(\phi_i p_i + P_{c,i})^2} \right)
$$

$$
+ \sum_{j \neq i} w_j \frac{\frac{\partial}{\partial p_i} R_j(\boldsymbol{p})}{\phi_j p_j + P_{c,j}}. \tag{29}
$$

When evaluated at $\boldsymbol{p} = \boldsymbol{p}^t$, the LHS and RHS are equal, and thus Condition 3) is satisfied.

REFERENCES

[1] B. Matthiesen, Y. Yang, and E. A. Jorswieck, "Optimization of weighted individual energy efficiencies in interference networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, Apr. 2018.

[2] "NGMN 5G white paper," NGMN Alliance, Tech. Rep., Mar. 2015.

[3] S. Buzzi, C.-L. I, T. E. Klein, H. V. Poor, C. Yang, and A. Zappone, "A survey of energy-efficient techniques for 5G networks and challenges ahead," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, 2016.

[4] A. Zappone and E. Jorswieck, *Energy Efficiency in Wireless Networks via Fractional Programming Theory*, ser. Found. Trends Commun. Inf. Theory. Now Publishers, 2015, vol. 11, no. 3-4.

[5] Z.-Q. Luo and S. Zhang, "Dynamic spectrum management: Complexity and duality," *IEEE J. Sel. Areas Commun.*, vol. 2, no. 1, pp. 57–73, Feb. 2008.

[6] D. W. K. Ng, E. S. Lo, and R. Schober, "Energy-efficient resource allocation in multi-cell OFDMA systems with limited backhaul capacity," *IEEE Trans. Wireless Commun.*, vol. 11, no. 10, pp. 3618–3631, Oct. 2012.

[7] Q. Xu, X. Li, H. Ji, and X. Du, "Energy-efficient resource allocation for heterogeneous services in OFDMA downlink networks: Systematic perspective," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2071–2082, June 2014.

[8] J. Xu and L. Qiu, "Energy efficiency optimization for MIMO broadcast channels," *IEEE Trans. Wireless Commun.*, vol. 12, no. 2, pp. 690–701, Feb. 2013.

[9] S. He, Y. Huang, S. Jin, and L. Yang, "Coordinated beamforming for energy efficient transmission in multicell multiuser systems," *IEEE Trans. Commun.*, vol. 61, no. 12, pp. 4961–4971, Dec. 2013.

[10] B. Du, C. Pan, W. Zhang, and M. Chen, "Distributed energy-efficient power optimization for CoMP systems with max-min fairness," *IEEE Commun. Lett.*, vol. 18, no. 6, pp. 999–1002, 2014.

[11] S. He, Y. Huang, L. Yang, and B. Ottersten, "Coordinated multicell multiuser precoding for maximizing weighted sum energy efficiency," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 741–751, Feb. 2014.

[12] A. Zappone, L. Sanguinetti, G. Bacci, E. A. Jorswieck, and M. Debbah, "Energy-efficient power control: A look at 5G wireless technologies," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1668–1683, Apr. 2016.

[13] A. Zappone, E. Björnson, L. Sanguinetti, and E. Jorswieck, "Globally optimal energy-efficient power control and receiver design in wireless networks," *IEEE Trans. Signal Process.*, vol. 65, no. 11, pp. 2844–2859, Jun. 2017.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[15] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[16] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Machine learning for wireless networks with artificial intelligence: A tutorial on neural networks," 2017. [Online]. Available: http://arxiv.org/pdf/1710.02913.pdf

[17] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3520–3535, June 2017.

[18] F. D. Calabrese, L. Wang, E. Ghadimi, G. Peters, and P. Soldati, "Learning radio resource management in 5G networks: Framework, opportunities and challenges," 2017. [Online]. Available: http://arxiv.org/abs/1611.10253

[19] Y. S. Nasir and D. Guo, "Deep reinforcement learning for distributed dynamic power allocation in wireless networks," 2018. [Online]. Available: https://arxiv.org/abs/1808.00490

[20] J. Fang, X. Li, W. Cheng, Z. Chen, and H. Li, "Intelligent power control for spectrum sharing: A deep reinforcement learning approach," 2017. [Online]. Available: https://arxiv.org/abs/1712.07365

[21] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.

[22] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," 2018. [Online]. Available: http://arxiv.org/abs/1807.10025

[23] P. de Kerret and D. Gesbert, "Robust decentralized joint precoding using team deep neural network," in *Proc. 15th Int. Symp. Wireless Commun. Syst. (ISWCS)*, 2018.

[24] A. Zappone, M. Di Renzo, M. Debbah, T. T. Lam, and X. Qian, "Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks towards wireless systems optimization," 2018. [Online]. Available: https://arxiv.org/abs/1808.01672

[25] Q. Shi, M. Razaviyayn, Z. Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[26] R. W. Freund and F. Jarre, "Solving the sum-of-ratios problem by an interior-point method," *J. Global Optim.*, vol. 19, no. 1, pp. 83–102, 2001.

[27] L. A. Zadeh, "Optimality and non-scalar-valued performance criteria," *IEEE Trans. Autom. Control*, vol. 8, no. 1, pp. 59–60, Jan. 1963.

[28] K. Miettinen, *Nonlinear Multiobjective Optimization*. Springer, 1999.

[29] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Netw.*, vol. 6, pp. 861–867, 1993.

[30] A. E. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 930–945, 1993.

[31] H. Tuy, *Convex Analysis and Global Optimization*. Springer International Publishing, 2016.

[32] Y. Yang and M. Pesavento, "A unified successive pseudoconvex approximation framework," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3313–3328, Jul. 2017.

[33] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*, ser. SIAM studies in applied mathematics 13. SIAM, 1994.

[34] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.

[35] G. Calcev *et al.*, "A wideband spatial channel model for system-wide simulations," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, March 2007.

[36] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[37] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," https://tensorflow.org, 2015.

[38] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artificial Intell. Statis.*, 2010, pp. 249–256.

[39] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.

[40] F. Chollet, *Deep Learning with Python*. Manning, Nov. 2017.
[41] 3GPP, "Digital cellular telecommunications systems (phase 2+); radio network planning aspects," ETSI, Tech. Rep. TR 43.030 V9.0.0 R9, Feb. 2010.
[42] T. S. Rappaport, *Wirless Communications*, 2nd ed. Prentice-Hall, 2002.