HAL
open science

Deep Learning Based Power Control for Cell-Free Massive MIMO with MRT

Lou Salaun, Hong Yang

To cite this version:

Lou Salaun, Hong Yang. Deep Learning Based Power Control for Cell-Free Massive MIMO with MRT. IEEE Global Communications Conference (GLOBECOM 2021), Dec 2021, Madrid, Spain. 10.1109/GLOBECOM46510.2021.9685229 . hal-03736680

HAL Id: hal-03736680

https://hal.science/hal-03736680v1

Submitted on 22 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Deep Learning Based Power Control for Cell-Free Massive MIMO with MRT

Lou Salaün
Nokia Bell Labs, 1 Route de Villejust
Nozay, 91620 France
lou.salaun@nokia-bell-labs.com

Hong Yang
Nokia Bell Labs, 600 Mountain Avenue
Murray Hill, NJ 07974 USA
h.yang@nokia-bell-labs.com

***Abstract*—Cell-Free Massive MIMO with MRT (Maximum-Ratio Transmission) has the advantage of decentralized beamforming with the smallest front-haul overhead. Its downlink power control plays a dual role of fair power distribution among users and interference mitigation. It is well-known that finding the optimal max-min power control relies on SOCP (Second Order Cone Programming) feasibility bisection search, whose large computational delay is not suitable for practical implementation. In this paper, we devise a deep learning approach for finding a practical near-optimal power control. Specifically, we propose a convolutional neural network that takes as input the channel matrix of large-scale fading coefficients and outputs the total transmit power of each AP (access point). Using this information, the downlink power control for each user is then computed by a low-complexity convex program. Our approach requires to generate far fewer training examples than existing schemes. The reason is that we augment the training dataset with magnitudes larger number of artificial examples by exploiting the special structure of the problem. The resulting deep learning model not only provides a near-optimal solution to the original problem, but also generalizes well for problems with different number of users and different propagation morphologies, without the need to retrain it. Numerical simulations validate the near optimality of our solution with a significant reduction in computational burden.**

***Index Terms*—Cell-Free, distributed, Massive MIMO, power control, deep learning, convolutional neural network, maximum ratio, MRT, conjugate beamforming, max-min.**

I. Introduction

Since the seminal work [1], Massive MIMO has become a core technology for 5G. Traditional cellular Massive MIMO employs many service antennas in each base station tower to accomplish precision beamforming. The intended coverage area is divided into multiple cells. Users in each cell area are served by a base station. Base stations do not cooperate in beamforming related signal processing but may cooperate in power control. Cell-Free Massive MIMO is a new wireless technology that is likely to play a key role for beyond 5G [2]. It promises ubiquitous coverage and high data rate to all users by leveraging the superior pathloss advantage furnished by the many APs distributed throughout the coverage area, and the precision beamforming and channel hardening benefits of Massive MIMO.

For Cell-Free Massive MIMO to become a reality, a major challenge is its fronthaul demand for the necessary beamforming signal processing [3]. Among all common precoding schemes, MRT has the lowest fronthaul overhead, but it requires sophisticated downlink power control to mitigate inter-user interference [4]. It is well-known that with MRT, downlink max-min power control, which maximizes the minimum SINR among all active users, can be obtained with SOCP feasibility bisection search. However, such algorithm comes with a large computational delay that is not suitable for practical implementation. The challenging goal of finding an approach for a practical downlink max-min power control has been pursued in the past with limited success [5], [6]. Paper [7] considers a Cell-Free Massive MIMO with multiple antenna APs, and proposes a deep learning method to approximate a high complexity heuristic algorithm for max-min power allocation. Deep learning based uplink power controls for Cell-Free Massive MIMO are studied in [8], [9].

In this paper, we aim to train a deep learning model to find a practical near-optimal max-min downlink power control. Our contributions are as follows.

1) We propose a deep learning model with 2 convolution layers and 4 fully connected layers to approximate closely the optimal power control. To our knowledge, it is the best in terms of performance among all practical solutions.
2) The neural network can be executed on different number of users thanks to its structure and the specific data preprocessing. Furthermore, we show by simulations that it generalizes well for different number of users and propagation morphologies, without the need to retrain it.
3) We exploit the special structure of the problem and propose a procedure to augment the training dataset with artificial examples. This greatly reduces the number of examples required to train the neural network.

II. Preliminaries

A. Notation

Superscripts ^{T}, *, denote transpose, complex conjugate transpose respectively. Thus double superscripts $^{\mathrm{T}*}$ and $^{*\mathrm{T}}$ denote un-transposed conjugate. $\mathbb{E}(\cdot)$ is the expectation. All vectors are column vectors. For $\mathbf{v} \in \mathbb{C}^n$, $\mathrm{diag}(\mathbf{v})$ denotes the diagonal matrix with $\mathbf{v}$ as its diagonal elements. For $\mathbf{A} \in \mathbb{C}^{n\times n}$, $\mathrm{diag}(\mathbf{A})$ denotes the vector formed by the diagonal elements of $\mathbf{A}$. $[\cdot]_{l,l}$ denotes the lth diagonal element of a matrix. $[\cdot]_l$ denotes the lth element of a vector. $\odot$ denotes the element-wise multiplication. $\mathbf{I}_n$ denotes the n-dimensional identity matrix.

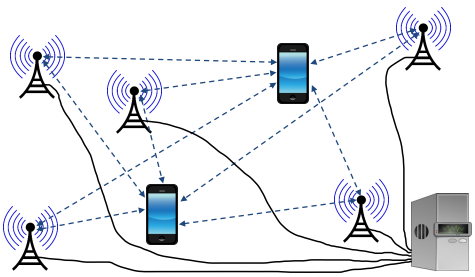

Fig. 1: A cell-free system with a CPU. All APs serve all users.

B. Cell-Free Massive MIMO

A "cell-free" system does not divide coverage area into cells, as in a traditional cellular system. The entire coverage area is served by M APs (access points) that are deployed throughout the coverage area. Each AP can have one or more service antennas. We assume that each AP has one service antenna, and the system is to serve K single antenna UEs (user equipments) simultaneously, and $M \gg K$. Fig. 1 shows an example of a cell-free system. In a cell-free system, each user is served by all APs simultaneously. All APs are connected to a CPU (central processing unit) for precoding and decoding processing, and for power control coordination.

Let

$$\mathbf{G} = (\mathbf{g}_1 \quad \cdots \quad \mathbf{g}_K) = \begin{pmatrix} \bar{\mathbf{g}}_1^{\mathrm{T}} \\ \vdots \\ \bar{\mathbf{g}}_M^{\mathrm{T}} \end{pmatrix} \in \mathbb{C}^{M \times K}$$

be the channel matrix between the M AP antennas and the K user antennas, where $\mathbf{g}_k \in \mathbb{C}^M$ is the channel vector between the kth user and the M AP antennas, and $\bar{\mathbf{g}}_m \in \mathbb{C}^K$ is the channel vector between the mth AP antenna and the K users.

The downlink data channel is modeled as

$$\mathbf{x} = \mathbf{G}^{\mathrm{T}}(\sqrt{\rho_{\mathrm{d}}}\mathbf{s}) + \mathbf{w} \tag{1}$$

where $\mathbf{x} \in \mathbb{C}^K$ is the received signal vector at the K user terminals, ρ_{d} is the downlink SNR (signal to noise ratio) for each AP, $\mathbf{s} \in \mathbb{C}^M$ is M precoded inputs to the M antenna ports at the M APs, and $\mathbf{w} \in \mathbb{C}^K$ is a circularly-symmetric Gaussian noise vector. Downlink power is subject to the power constraint for each access point, and can be specified as

$$\|\mathbb{E}(\mathbf{s}^{*\mathrm{T}} \odot \mathbf{s})\|_\infty \leq 1. \tag{2}$$

C. Downlink SINR with MRT

A Cell-Free Massive MIMO system distributes M service antennas throughout an intended coverage area. These service antennas serve as access points to provide data service to K autonomous users, where M is much larger than K. This is in contrast to a cellular Massive MIMO system in which an intended coverage area is divided into cells, and users located in each cell is served by an M antenna array that is located at the center of the cell.

For a Cell-Free Massive MIMO, the channel between the mth service antenna and the kth user is modeled as

$$g_{m,k} = \sqrt{\beta_{m,k}} h_{m,k}, \tag{3}$$

where $\beta_{m,k}$ models the large-scale fading that accounts for geometric attenuation and shadow fading; $h_{m,k}$ models the small-scale fading that accounts for random scattering.

In rich scattering propagation environment, the magnitude of the signal typically varies randomly according to the Rayleigh distribution, thus the small-scaling fading $h_{m,k}, \forall m, k$ are modeled as circularly symmetric complex Gaussian, independent and identically distributed random variables.

Under these assumptions, and with MMSE (minimum mean square error) channel estimation based on orthogonal uplink pilot sequence, the ergodic downlink effective SINR for the Cell-Free Massive MIMO system with MRT precoding is given by [5]

$$\mathrm{SINR}_k^{\text{cell-free}} = \frac{\rho_{\mathrm{d}} \left(\sum_{m=1}^{M} \sqrt{\alpha_{m,k}\eta_{m,k}} \right)^2}{1 + \rho_{\mathrm{d}} \sum_{m=1}^{M} \beta_{m,k} \sum_{k'=1}^{K} \eta_{m,k'}} \tag{4}$$

where

$$\alpha_{m,k} = \frac{\rho_{\mathrm{u}} \tau \beta_{m,k}^2}{1 + \rho_{\mathrm{u}} \tau \beta_{m,k}} \tag{5}$$

is the mean-square of the channel estimate, and ρ_{d} and ρ_{u} are the normalized downlink and uplink SNR (Signal-to-Noise Ratio) respectively. τ is the length of the uplink pilot sequence that is used for channel estimation. $\boldsymbol{\eta} = \{\eta_{m,k}\}$ is the downlink power control which is subject to per AP power constraint:

$$\sum_{k'=1}^{K} \eta_{m,k'} \leq 1, \quad \forall m. \tag{6}$$

The max-min power control optimization problem can be formulated as

$$\begin{aligned} \max_{\boldsymbol{\eta}} \quad & \min_k \quad \mathrm{SINR}_k^{\text{cell-free}}, \\ \text{subject to} \quad & \sum_{k'=1}^{K} \eta_{m,k'} \leq 1, \quad \forall m, \\ & \eta_{m,k'} \geq 0, \quad \forall m,\ k'. \end{aligned} \tag{$\mathcal{P}$}$$

An optimal solution to problem $\mathcal{P}$ can be obtained with SOCP feasibility bisection search [10], [5], [11]. However, SOCP's computational complexity becomes impractical as M and K increase. Hence, we use deep learning to simplify the problem by finding near-optimal values for intermediate variables. Suppose that the optimal transmit power $p_m^{\text{opt}} = \sum_{k'=1}^{K} \eta_{m,k'}$ for the m-th AP can be computed by a neural network, then the problem can be simplified as follows

$$\begin{aligned} \max_{\boldsymbol{\eta}} \quad & \min_k \quad \frac{\rho_{\mathrm{d}} \left(\sum_{m=1}^{M} \sqrt{\alpha_{m,k}\eta_{m,k}} \right)^2}{1 + \rho_{\mathrm{d}} \sum_{m=1}^{M} \beta_{m,k} p_m^{\text{opt}}}, \\ \text{subject to} \quad & \sum_{k'=1}^{K} \eta_{m,k'} = p_m^{\text{opt}}, \quad \forall m, \\ & \eta_{m,k'} \geq 0, \quad \forall m,\ k'. \end{aligned} \tag{$\mathcal{P}'$}$$

In this paper, we develop and train a neural network to find the value of p_m^{opt} for all m. It takes as input the large-scale fading coefficients matrix $\mathbf{B} = (\beta_{m,k})_{m=1\ldots M,\, k=1\ldots K}$

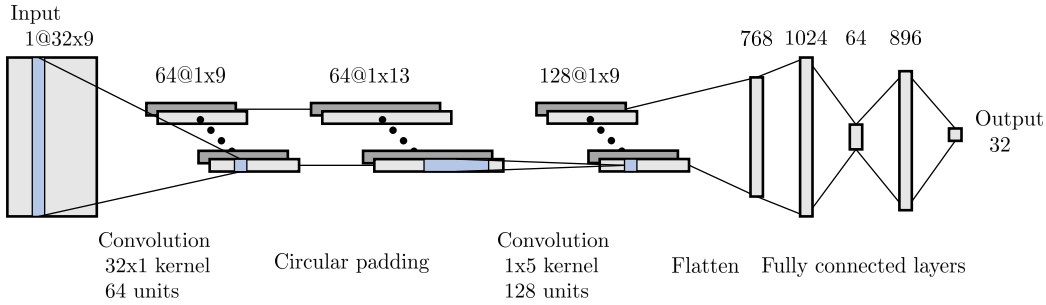

Fig. 2: Structure of the neural network

and outputs the per AP power vector $\mathbf{p} = (p_1^{\text{opt}} \cdots p_M^{\text{opt}})$. The resulting problem $\mathcal{P}'$ is known to be convex, see references [5], [6]. Therefore it can be readily solved by standard convex programming tools with low complexity.

III. Deep Learning Power Control

A. Datasets

To train and evaluate the neural network, we generate datasets composed of optimal $(\mathbf{B}, \mathbf{p})$ pairs. Each dataset corresponds to a different simulation scenario characterized by a triplet (M, K, mor), where $mor \in \{urban, suburban, rural\}$ denotes the propagation morphology. For each example of the dataset, we generate randomly the channel between the M APs and the K users according to the propagation morphology mor. The corresponding optimal $\mathbf{p}$ is obtained by a second-order cone program (SOCP).

The neural network proposed in this paper takes as input a matrix of size 32x9 and outputs a vector of size 32, which means that it accepts any data with $M \leq 32$ and $K \leq 9$. These values are chosen arbitrarily here, nevertheless the input size, output size and hidden layers size of the neural network can be scaled to match any practical system's requirements in M and K.

If an input matrix $\mathbf{B}$ is smaller than 32x9, the missing elements are padded with a well chosen constant value. We choose this constant to be lower than any other values in the dataset, i.e., 10^{-20} in our simulations. Similarly, if the size of $\mathbf{p}$ is less than 32, we pad the missing elements with 0. The idea is to train the neural network to interpret these padded values as missing elements both in the input and the output. We will see in the next subsection that a $\log_2$ transformation is applied to $\mathbf{B}$ during the preprocessing step. For this reason, we choose to pad the input matrix with 10^{-20} instead of 0 to avoid numerical issues.

B. Preprocessing

The initial values of $\mathbf{B}$ in the datasets are not suitable for neural networks such as multi-layer perceptron and convolutional neural networks. Indeed, the large-scale fading values vary by many orders of magnitude, while the connections between neurons remain within the same order of magnitudes. To solve this issue, we first apply a $\log_2$ transformation to all the $\beta_{m,k}$. As an example, if $\beta_{m,k}$ takes values in $\left[10^{-15}, 10^{-5}\right]$, then $-50 < \log_2 \beta_{m,k} < -16$. With this transformation, the neural network is now able to extract useful information from the dataset. Note that, it does not matter which logarithm base we use as they are all equal up to a constant factor.

Another common practice is to normalize each input feature $\log_2 \beta_{m,k}$ and output p_m^{opt} to have zero mean and unit standard deviation over all examples of the dataset. This helps speed up the training, as the model does not have to learn the statistics of the data.

When multiple datasets are used by the same neural network –regardless of whether it is for training or during execution– they are normalized jointly with the same mean and standard deviation values. This ensures that the input and output values are consistent between any two examples that go through the neural network.

C. Structure of the Neural Network

The structure of the neural network is illustrated in figure 2. We use the notation "depth @ height x width" to indicate the size of the convolution layers. The numbers on top of the fully connected layers represent their number of neurons.

A preliminary model investigated is a multi-layer perceptron with only fully connected layers. It works well for small instances like 10x2. However, the size of such fully connected layers do not scale well when M and K increases. For this reason, we incorporated convolution layers in the current model, which are suitable for image-like inputs like $\mathbf{B}$.

The first convolution layer has a kernel of size 32x1, i.e., for each user k, the 32 large-scale fading coefficients $\beta_{1,k} \cdots \beta_{M,k}$ are connected to 64 units. The ouput of this layer is a tensor of size 64@1x9, where the k-th column has 64 neurons processing the information of user k only.

Let $(\mathbf{c}_1 \cdots \mathbf{c}_9)$ be the columns of the 64@1x9 tensor. The circular padding consists in repeating the columns periodically to get a 64@1x13 tensor of the form $(\mathbf{c}_1 \cdots \mathbf{c}_9 \mathbf{c}_1 \mathbf{c}_2 \mathbf{c}_3 \mathbf{c}_4)$. Since the second convolution has a kernel of size 1x5, the circular padding ensures that each column $\mathbf{c}_k$ is connected to the same number of neurons. Without circular padding, columns $\mathbf{c}_1$ and $\mathbf{c}_9$ would be connected to 128 neurons, while column $\mathbf{c}_5$ would be connected to $128 \times 5 = 640$ neurons.

The output of this second convolution layer is flatten into $128 \times 1 \times 9 = 1152$ neurons, then followed by 4 fully connected layers of size 768, 1024, 64 and 896. The ELU (Exponential Linear Unit) activation function is used on all hidden layers, followed by a layer normalization [12]. The output layer is a simple linear activation.

D. Training and Data Augmentation

For training, we use the Nadam optimizer [13] with a learning rate of 7×10^{-4}. The batch size is 1024, and the training is stopped after 200 epochs.

The training dataset is composed of $18,000$ examples from the $(M, K, mor) = (32, 6, urban)$ scenario, and $18,000$ examples from the $(M, K, mor) = (32, 9, urban)$ scenario. By training the network on these two scenarios, it learns to generalize. Indeed, we will see in Section IV that the network generalizes well even for values of K and morphologies it did not trained for.

One can observe that the problem formulation $\mathcal{P}$ does not depend on the choice of indices for the users and APs. In other words, permuting the columns of $\mathbf{B}$, and permuting the rows of $\mathbf{B}$ and $\mathbf{p}$ produces another valid and optimal instance of $\mathcal{P}$. The row permutation (i.e., permutation of the APs' indices) applied to $\mathbf{B}$ and $\mathbf{p}$ has to be the same. Using this idea, we augment the dataset by creating new examples. The procedure is as follows: each example of the dataset is duplicated 60 times; each duplicated example is randomly permuted. We make sure that the 60 permutations applied on the same example are all different to avoid creating identical examples. The augmented training dataset now contains $36,000 \times 60 = 2.16 \times 10^6$ examples.

This augmentation is interesting for training as it achieves efficient regularization without the need to generate a massive number of "real" examples, which would take considerable time due to the complexity of SOCP. In addition, the augmentation helps the neural network learn the symmetries of the problem. It is important to note that this augmentation will only be used for training and not for performance evaluation as it produces redundant examples.

IV. Numerical Results

In this section, we demonstrate the usage and effectiveness of our deep learning model for obtaining near-optimal max-min power control.

In our simulations, maximum 32 APs and maximum 9 users are randomly distributed in a circular area within a radius of 500 meters for the urban scenario, 1 km for suburban and 4 km for rural. Both service antennas and user antennas have 0 dBi gain. Both AP receiver and mobile receiver have a 9 dB noise figure. The transmit power of each AP is 200 mW. Noise power calculation assumes 20 MHz carrier spectral bandwidth.

For each simulation scenario (M, K, mor), we generate a dataset of $2,000$ examples which are used to obtain the figures in this section.

A. Propagation Models

We use the "NLoS" propagation models specified in [14], for which the path loss in dB is given by

$$\begin{aligned}\mathrm{PL}(d) = {} & 161.04 - 7.1\log_{10}(W) + 7.5\log_{10}(h) \\ & -[24.37 - 3.7(h/h_{\mathrm{AP}})^2]\log_{10}(h_{\mathrm{AP}}) \\ & +[43.42 - 3.1\log_{10}(h_{\mathrm{AP}})][\log_{10}(d) - 3] \\ & +20\log_{10}(f_{\mathrm{c}}) - (3.2[\log_{10}(11.75h_{\mathrm{AT}})]^2 - 4.97)\end{aligned}$$

	Urban	Suburban	Rural
W (in meters)	20	20	20
h (in meters)	20	10	5
h_{AP} (in meters)	20	20	40
h_{AT} (in meters)	1.5	1.5	1.5
f_{c} (in GHz)	2	2	0.45
σ (in dB)	6	8	8

TABLE I: Parameters for path loss models

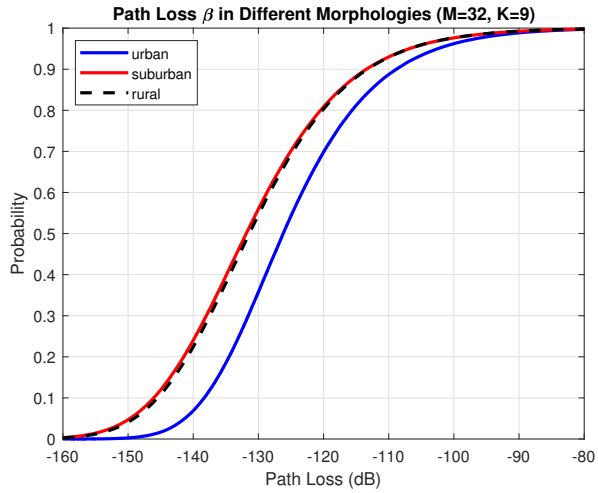

Fig. 3: Distributions of β in different morphologies

where W is the street width (in meters); h is the average building height (in meters); h_{AP} is the AP antenna height (in meters); h_{AT} is the user AT antenna height (in meters); f_{c} is the carrier frequency in GHz; and d is the distance between transmitter antenna and receiver antenna, also in meters.

Our simulation parameters for the propagation models are compatible with [14] and are summarized in Table I, where σ is the standard deviation of the lognormal shadow fading.

We consider three propagation morphologies, namely urban, suburban and rural. From Fig. 3, we see that there is substantial difference in the distributions of path loss β between urban and suburban. 2 GHz band is assumed for both urban and suburban. Distributions for suburban and rural are similar due to the fact that 450 MHz band is used in rural.

Fig. 4 plots the CDF of β for different numbers of APs and UEs in a fixed area (disk with radius = 500 m) for the urban morphology. We see that the distributions of path loss β are essentially the same for the Ms and Ks we considered.

B. Normalized Downlink and Uplink SNR

The normalized downlink SNR (Signal-to-Noise Ratio) ρ_{d} is calculated as follows.

$$\rho_{\mathrm{d}} = P_{\mathrm{d}} \times \mathrm{Gain}_{\mathrm{AP}} \times \mathrm{Gain}_{\mathrm{AT}}/(B \times N_0 \times \mathrm{NF}_{\mathrm{AT}}),$$

where P_{d} is the full downlink radiated power for each AP; $\mathrm{Gain}_{\mathrm{AP}}$ is the antenna gain of the service antenna at the AP minus associated cable loss, and $\mathrm{Gain}_{\mathrm{AT}}$ is the user AT antenna gain minus associated cable loss; B is the carrier spectral bandwidth; N_0 is the spectral density of the thermal noise power, which can be calculated as $N_0 = kT$, where

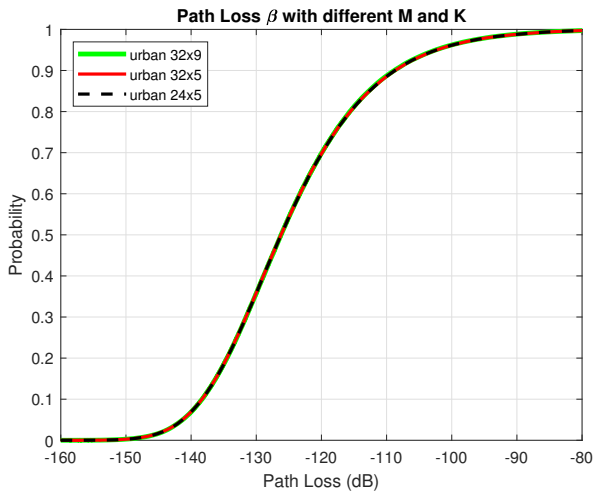

Fig. 4: Distributions of β with different M and K

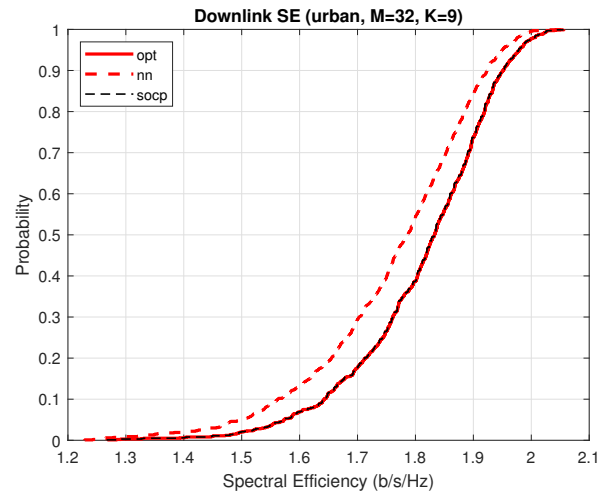

Fig. 5: CDF of per user SE for $(M, K, mor) = (32, 9, urban)$

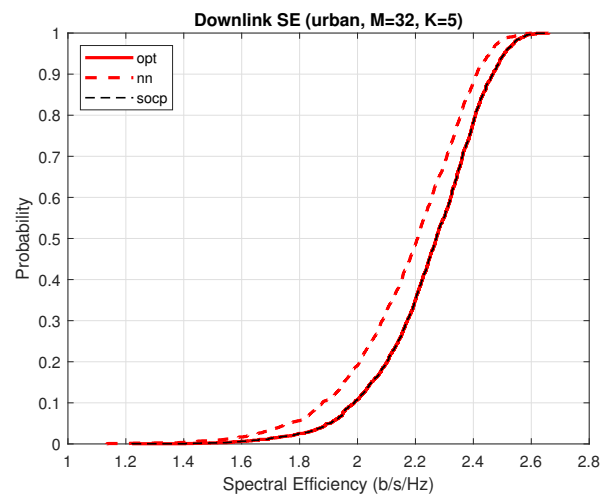

Fig. 6: CDF of per user SE for $(M, K, mor) = (32, 5, urban)$

$k = 1.3807 \times 10^{-23}$ is Boltzmann's constant in joules per kelvin, and T is the receiver system noise temperature in kelvins; $\mathrm{NF_{AT}}$ is the user access terminal receiver noise figure.

Similarly, the normalized uplink SNR ρ_{u} is calculated as

$$\rho_{\mathrm{u}} = P_{\mathrm{u}} \times \mathrm{Gain_{AP}} \times \mathrm{Gain_{AT}}/(B \times N_0 \times \mathrm{NF_{AP}}),$$

where P_{u} is the full uplink radiated power for each AT; $\mathrm{NF_{AP}}$ is the AP receiver noise figure.

The uplink SNR ρ_{u} is relevant because the downlink throughput performance depends on the quality of the channel estimation, which in turn depends on ρ_{u} as shown in (5). As mentioned previously, in our simulation, we assume 200 mW radiated power for both AP and UE amplifiers, resulting in $\rho_{\mathrm{d}} = \rho_{\mathrm{u}} \approx 3.144 \times 10^{11}$. We use K orthogonal pilots for channel estimation, i.e., $\tau = K$ in (5).

C. Near Optimal Spectral Efficiency

The figures in this section present the spectral efficiency CDF of our deep-learning based solution compared to the optimal for different simulation scenarios. The curve "nn" shows the performance of our model. Curve "socp" is obtained by solving SOCP on problem $\mathcal{P}$, while "opt" is obtained by solving $\mathcal{P}'$ using a convex solver given that the optimal p_m^{opt} are given. As expected, curves "socp" and "opt" are identical since problems $\mathcal{P}$ and $\mathcal{P}'$ are equivalent.

We recall that our model is trained on the urban scenario with $M = 32$ and $K = 6, 9$. The performance of this model in terms of per user SE is shown in Fig. 5. At median, the deep-learning based solution is about 0.05 b/s/Hz away from optimal, which corresponds to around 2.7% loss in per user spectral efficiency.

To see how our model generalizes to different K, we run a simulation with $(M, K, mor) = (32, 5, urban)$. Fig. 6 shows that the deep-learning based solution is about 0.065 b/s/Hz away from optimal at median.

An interesting question is whether our model can generalize to different M and K. To verify this we run a simulation with $(M, K, mor) = (24, 5, urban)$. Fig. 7 shows that the deep-learning based solution loses about 0.28 b/s/Hz from optimal at median, that is approximately 14.6% loss in per user spectral efficiency. The quality of the neural network inference degrades compared to the previous scenarios with $M = 32$. Further tuning of the model, with additional training on different number of APs, may be able to reduce this loss.

We also validate the performance of our model with other morphologies. Fig. 8 represents a suburban scenario, in which the deep-learning based solution is approximately 0.05 b/s/Hz away from optimal at median. It is also 0.05 b/s/Hz away from optimal at median in the rural scenario shown in Fig. 9. This represents about 2.8% loss in per user spectral efficiency.

In summary, our deep-learning based solution generalizes very well to different number of users and different propagation morphologies, without the need to retrain it specifically for these new cases. However, it does not perform as well when the number of APs changes. We offer two possible causes as follows. First, the convolution layers in Fig. 2 can efficiently adapt to missing users in the input when $K < 9$, while it is less suited for missing APs when $M < 32$. Secondly, the model has only been trained on examples with $M = 32$ APs. It may

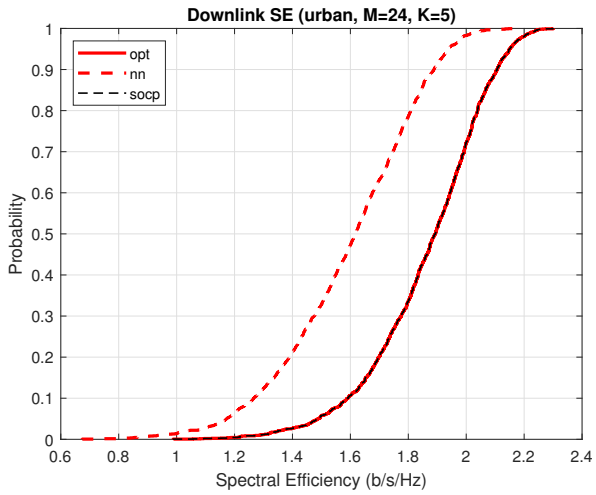

Fig. 7: CDF of per user SE for $(M, K, mor) = (24, 5, urban)$

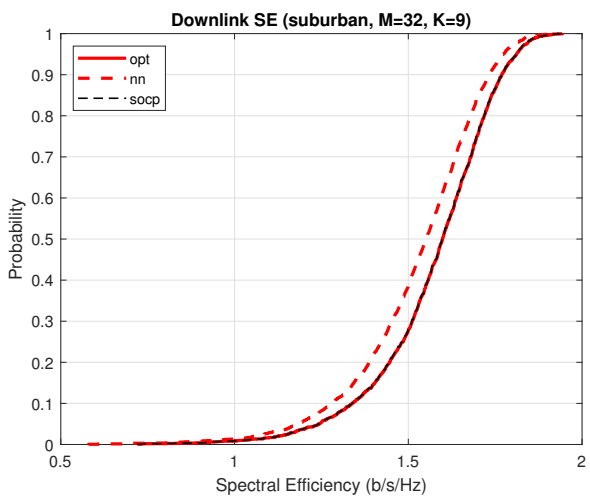

Fig. 8: CDF of per user SE for $(M, K, mor) = (32, 9, suburban)$

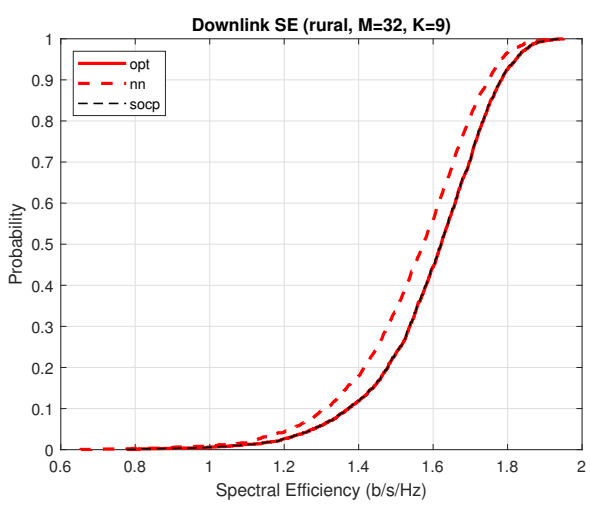

Fig. 9: CDF of per user SE for $(M, K, mor) = (32, 9, rural)$

	SOCP	NN
Urban, $M = 24, K = 5$	1.92×10^8	3.74×10^6
Urban, $M = 32, K = 5$	2.76×10^8	
Urban, $M = 32, K = 9$	3.46×10^8	
Suburban, $M = 32, K = 9$	3.55×10^8	
Rural, $M = 32, K = 9$	3.79×10^8	

TABLE II: FLOPs comparison between SOCP and the proposed neural network for different number of users and APs

be able to generalize better with further training on different values of M, as well as improvements on the structure of the neural network.

D. Computational Complexity

To compare the computational complexity of SOCP and our solution, we count their number of floating point operations (FLOPs) during execution. Each multiplication or addition counts as one FLOP. We use the MOSEK solver [15] in Python 3 to solve the SOCP problem. The SINR accuracy at termination of the bisection search is set to 0.01. The neural network is implemented with Tensorflow in Python 3. We show the number of FLOPs in Table II for different number of users and APs. Each value in this table is obtained by averaging the FLOPs count over $2,000$ random instances.

The complexity of the neural network depends on its number of parameters, the number of layers and their sizes. However, it does not vary with the input size since $\mathbf{B}$ is padded to be consistently of size 32×9. Therefore, the execution complexity of our solution remains constant, equal to 3.74×10^6 FLOPs, in all scenarios.

The complexity of SOCP increases with M and K, and varies slightly between different propagation morphologies. We can see in Table II that SOCP requires 50 times more FLOPs than our solution when $M = 24$ and $K = 5$. For scenarios with 32 APs and 5 to 9 UEs, the neural network performs 70 to 100 times fewer FLOPs than SOCP.

The significant reduction in computational complexity can also be seen by their run-times. Indeed, when both algorithms are executed on the same computer[1], we observe that SOCP takes around 5s, while our model terminates under 50ms. Hence, the proposed deep learning solution achieves low run-time suitable for practical implementations.

V. Conclusions

Cell-Free Massive MIMO is a beyond 5G technology that promises ubiquitous coverage and high data rate to all users. A major challenge for such a system to become reality is its fronthaul overhead and power control complexity. MRT has the lowest fronthaul overhead, but it requires sophisticated downlink power control to mitigate inter-user interference. We employ a deep learning approach to the problem and demonstrate its effectiveness. We exploit the special structure of the problem and come up with a method to easily augment

[1] with the following specifications: Intel Core i5 CPU with 8 GB of RAM, Windows 10, 64 bits. GPU is not used by the neural network to keep the comparison fair.

the number of training examples by magnitudes. Our trained model generalizes well for different number of users, and different propagation morphologies. Simulations show that the model is practically optimal in the sense that it delivers spectral efficiency that is less than 3% away from optimal, leaving little room for further improvement for all practical purposes. An added advantage is the resulting near optimal power control is also energy efficient [11].

Currently, our model does not generalize to different number of APs well. Possible further investigation includes: 1) Extend the model to handle different number of APs. 2) Learning optimal linear precoding where either the whole precoding matrix is learned or some intermediate structures are learned to speed up the computation of the optimal precoding matrix.

References

[1] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 9, no. 11, pp. 3590–3600, 2010.

[2] J. Zhang, E. Björnson, M. Matthaiou, D. W. K. Ng, H. Yang, and D. J. Love, "Prospective multiple antenna technologies for beyond 5G," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 8, pp. 1637–1660, 2020.

[3] E. Björnson and L. Sanguinetti, "Making cell-free Massive MIMO competitive with MMSE processing and centralized implementation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 77–90, 2020.

[4] G. Interdonato, H. Q. Ngo, and E. G. Larsson, "Enhanced normalized conjugate beamforming for cell-free Massive MIMO," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 2863–2877, 2021.

[5] E. Nayebi, A. Ashikhmin, T. L. Marzetta, H. Yang, and B. D. Rao, "Precoding and power optimization in cell-free Massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 7, pp. 4445–4459, 2017.

[6] H. Yan, A. Ashikhmin, and H. Yang, "A scalable and energy efficient IoT system supported by cell-free Massive MIMO," *IEEE Internet Things J.*, 2021.

[7] Y. Zhao, I. G. Niemegeers, and S. Heemstra de Groot, "Power allocation in cell-free Massive MIMO: A deep learning method," *IEEE Access*, vol. 8, no. 5, pp. 87 185–87 200, 2020.

[8] C. D'Andrea, A. Zappone, S. Buzzi, and M. Debbah, "Uplink power control in cell-free Massive MIMO via deep learning," in *IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2019.

[9] N. Rajapaksha, K. B. S. Manosha, N. Rajatheva, and M. Latva-aho, "Deep learning-based power control for cell-free Massive MIMO networks," *arXiv preprint arXiv:2102.10366*, 2021.

[10] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free Massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1834–1850, 2017.

[11] H. Yang and T. L. Marzetta, "Energy efficiency of Massive MIMO: Cell-free vs. cellular," in *IEEE 87th Veh. Technol. Conf.*, 2018.

[12] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[13] T. Dozat, "Incorporating Nesterov momentum into Adam," 2016.

[14] M. Series, "Guidelines for evaluation of radio interface technologies for IMT-Advanced," *Report ITU M.2135-1*, 2009. [Online]. Available: https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2135-1-2009-PDF-E.pdf

[15] E. D. Andersen, C. Roos, and T. Terlaky, "On implementing a primal-dual interior-point method for conic quadratic optimization," *Mathematical Programming*, vol. 95, no. 2, pp. 249–277, 2003.