MIMO CHANNEL ESTIMATION WITH NON-IDEAL ADCS: DEEP LEARNING VERSUS GAMP

Marcos Yuichi Takeda, Aldebaro Klautau[*]

Federal University of Pará
LASSE - 5G Research Group
Belem, Pará, 66075-110, Brazil
marcos.takeda@itec.ufpa.br, aldebaro@ufpa.br

Amine Mezghani and Robert W. Heath Jr.[†]

The University of Texas, Austin
Dep. of Elec. and Computer Eng.
Austin, Texas, 78712-1084, USA
{amine.mezghani,rheath}@utexas.edu

ABSTRACT

Channel estimation for massive MIMO using coarse quantizers is nontrivial due to severe nonlinear distortions caused by quantization and the large dimensional MIMO channel. The best solutions to this problem nowadays are based on the generalized approximate message passing (GAMP) and its variations. However, there are practical issues such as nonideal quantizers that may violate the assumptions in which GAMP algorithms rely. This motivates research on methods based on deep learning (DL), which provides a framework for designing new communication systems that embrace practical impairments. We explore DL applied to channel estimation of MIMO systems with low resolution analog-to-digital converters (ADCs). Assuming millimeter wave MIMO, the channel estimation results indicate that a single neural network trained in a range of practical conditions is more robust to ADC impairments than a GAMP variant. For example, for a fixed wireless scenario with channels obtained from ray-tracing, DL achieved a normalized mean-square error lower than GAMP's by more than 5 dB.

1. INTRODUCTION

Deep learning can be used to solve problems in communications systems design whenever a standard approach is difficult [1–3]. In this paper, we apply deep learning to channel estimation of MIMO systems with 1-bit analog-to-digital converters (ADCs). Due to the number of antennas in massive MIMO, the use of low power ADCs is a sensible alternative to reduce the power consumption. This comes at the cost of a reduced resolution, which imposes challenges to receiver algorithms due to the non-linearity of the quantization operation and the massive number of antennas.

State-of-art algorithms for MIMO with low resolution ADCs are based on *message passing* (MP) [4], especially the *generalized approximate message passing* (GAMP) [5] and variations. The bilinear variants [6, 7] are successfully used on joint channel and detection (JCD) estimation. Moreover, [8, 9] provide an analysis of GAMP performance when used on JCD estimation with low resolution ADCs. The results in [8–10] show that, when GAMP assumptions hold, it performs close to the optimum at reasonable complexity. The contributions of this papers are two-fold. First we investigate DL architectures for 1-bit MIMO channel estimation, using distinct datasets. Having channels obtained from both statistical models and ray-tracing simulations allow us to perceive the impact of the datasets on performance. This is sensible given that machine learning algorithms can lead to biased conclusions simply by the use of an evaluation methodology based on restricted datasets. The second contribution is a direct comparison between DL methods and EM-BG-GAMP [11], a state-of-art variant of GAMP that outperforms previous algorithms (such as, e. g. [12]). We adopt millimeter wave (mmWave) MIMO systems and investigate the hypothesis that, in cases that GAMP assumptions do not hold, DL-based approaches using neural networks (NN) can outperform GAMP.

Channel estimation for low-resolution mmWave MIMO has been tackled with techniques that do not use DL, such as in [13, 14], which adopt *orthogonal matching pursuit* (OMP) and variational Bayesian line spectral estimation (VALSE) to leverage sparsity, respectively. Some recent papers emphasize advantages of DL for channel estimation (not necessarily with low-resolution ADCs) [15–18]. DL is used for channel estimation in [16], which proposes a pre-processing ("tentative estimation") to alleviate the task of the subsequent convolutional deep neural network (DNN). The authors in [16] use DL to explore the time and frequency correlations in mmWave wideband channels. In [18], the DNN is used to denoise the data, which is then processed by a conventional estimator. This requires adjusting the DNN weights for each

[*] Supported in part by, respectively, CAPES and CNPq, Brazil.

[†] This material is based upon work supported in part by the National Science Foundation under Grant No. ECCS-1711702 and Grant No. CNS-1731658.

978-1-7281-0824-7/19/$31.00 ©2019 IEEE

coherence block, but avoids the conventional offline training stage. Another online channel estimation method is proposed in [19], which avoids the cost of offline DNN training by using the extreme learning machine (ELM) architecture. Different from these papers, we investigate channel estimation with DL applied to systems with 1-bit ADCs and contrast it with GAMP.

2. SYSTEM MODEL

The system is modeled as depicted in Fig. 1. The number of transmit and receive antennas are N_t and N_r, respectively. The single-user MIMO matrix $\mathbf{H} \in \mathbb{C}_{N_\mathrm{r} \times N_\mathrm{t}}$ represents a block fading channel, such that the channel remains constant over a period of T symbols. Assuming ideal synchronism, one can write

$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{W}, \tag{1}$$

where $\mathbf{X}_{N_\mathrm{t} \times T}$ contains the transmit symbols, $\mathbf{Y}_{N_\mathrm{r} \times T}$ has the received symbols and $\mathbf{W}_{N_\mathrm{r} \times T}$ is element-wise additive white Gaussian noise. The received samples are further quantized upon reception, as

$$\tilde{\mathbf{Y}} = Q(\mathbf{Y}) \tag{2}$$

where $Q(x)$ is a function representing the quantization operation on each real and imaginary components. For an ideal 1-bit quantizer, the operation corresponds to $Q(x) = \mathrm{sgn}(x)$, where $\mathrm{sgn}(\cdot)$ is the signum function.

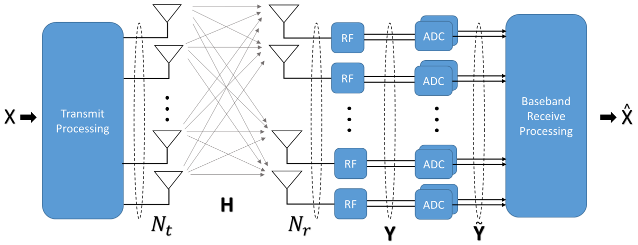

Fig. 1. System model.

We are interested in mmWave MIMO systems with uniformly-spaced antenna arrays. Therefore, we conveniently use the two-dimensional fast Fourier transform (FFT) of $\mathbf{H}$, the so-called *virtual* (or *angular*) channel representation [20]

$$\mathbf{H_v} = \mathrm{FFT}_{\mathrm{2D}}\{\mathbf{H}\}, \tag{3}$$

proposed in [21]. The virtual channel $\mathbf{H_v}$ has a sparse structure that favors channel estimation algorithms [20].

We assume channel estimation based on known pilot symbols, which are transmitted at the beginning of each block of length T. A transmission block is divided in two parts: training (or pilot) symbols and data symbols, represented by

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{\mathrm{pilot}} & \mathbf{X}_{\mathrm{data}} \end{bmatrix}, \tag{4}$$

$$\tilde{\mathbf{Y}} = \begin{bmatrix} \tilde{\mathbf{Y}}_{\mathrm{pilot}} & \tilde{\mathbf{Y}}_{\mathrm{data}} \end{bmatrix}. \tag{5}$$

At the receiver, $\tilde{\mathbf{Y}}_{\mathrm{pilot}}$ and $\tilde{\mathbf{Y}}_{\mathrm{data}}$ are observed at each block interval, while $\mathbf{X}_{\mathrm{pilot}}$ is known *a priori*.

In this paper we model a non-ideal 1-bit quantizer as a hysteresis based on two thresholds t_{down} and t_{up}, both distributed according to a zero-mean Gaussian $\mathcal{N}(0, \sigma^2)$ with variance σ^2. We enforce that $t_{\mathrm{down}} \leq t_{\mathrm{up}}$ and, if we draw values for which this is not true, these values are swapped. Fig. 2 illustrates the hysteresis model used. Imperfections are inevitable on ADCs implementations [22], and, for 1-bit quantizers, a prominent one is the threshold hysteresis.

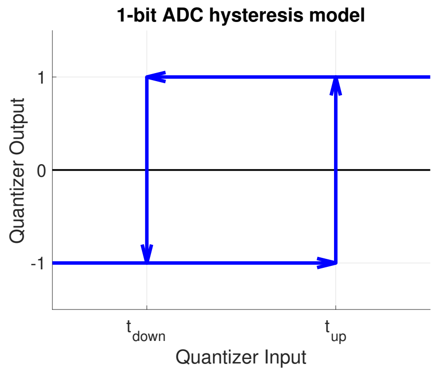

Fig. 2. Non-ideal ADC hysteresis.

3. CHANNEL ESTIMATION

An extra difficulty in this channel estimation problem comes from the non-linearity resulting from the 1-bit ADC quantization. GAMP-based algorithms model the non-linearity resulting from the 1-bit ADC quantization. Therefore, it performs extremely well when the model assumptions are met. Also, GAMP-based approaches can leverage even information from $\tilde{\mathbf{Y}}_{\mathrm{data}}$. On the other hand, NNs are universal function approximators that can effectively handle non-linearities.

Before delving into the GAMP and DL-based methods, it should be noticed that for 1-bit MIMO there are $M = 2^{2N_\mathrm{r}T_{\mathrm{pilot}}}$ distinct received pilots $\tilde{\mathbf{Y}}_{\mathrm{pilot}}$, where T_{pilot} is the number of pilot symbols. Therefore, channel estimation could be eventually implemented as a look-up table. The input would be the index corresponding to $\tilde{\mathbf{Y}}_{\mathrm{pilot}}$, and $\hat{\mathbf{H}}$ the complex-valued output with dimension $N_\mathrm{t}N_\mathrm{r}$. There are methods based on look-up tables [23], but the tables may not be feasible when large dimensions are involved. We use GAMP or NNs to implement this mapping with reasonable complexity.

3.1. Channel estimation using deep learning

In our ML-based approach, the channel estimation problem is posed as a multivariate regression problem [24]. In this paper, the output is the estimated channel $\hat{\mathbf{H}}$ such that a single NN performs multivariate regression. There are several alternatives to the NN input and overall architecture. The quantized pilot samples $\tilde{\mathbf{Y}}_{\text{pilot}}$ are the most prominent source of information regarding the channel and, consequently, are part of the NN input. The NN input may include $\mathbf{X}_{\text{pilot}}$ or not. Because $\mathbf{X}_{\text{pilot}}$ does not change over time in our simulations, it is not part of the input. For optimizing the model weights (i. e., for training the NN), we adopt the mean squared error (MSE) as loss function.

The input $\tilde{\mathbf{Y}}_{\text{pilot}}$ is organized as a matrix of dimension $2N_{\text{r}} \times T$, where the 2 factor accounts for the real and imaginary parts. The output is the virtual channel matrix $\mathbf{H_v}$, with the same dimension $N_{\text{r}} \times N_{\text{t}}$ of $\mathbf{H}$. The output size is then $2N_{\text{r}} \times N_{\text{t}}$ to account for the complex numbers. The next paragraphs describe the three adopted NN architectures. As a preliminar study, a thorough hyperparameter optimization was avoided so that these values were selected manually.

A simple dense network architecture with 8 layers is used. The activations for all layers but the last are rectified linear units (ReLU). Each layer, except the last, has 128 outputs which are fed to the next layers. The last layer uses a linear activation. Neither *dropout* nor *batch normalization* was used.

The convolutional network model used in this paper has 3 convolutional layers and 1 dense layer as the output. The convolutional layers configuration is described in Table 1. Each convolutional layer uses ReLU activation and is followed by a batch normalization layer and a dropout layer with dropout rate $r_{\text{drop}} = 0.3$. No padding is used on the convolution. The output layer uses a linear activation as the other models.

Table 1. Topology of the convolutional network.

Conv. layer	Outputs	Window size	Stride
1	192	5	1
2	384	5	1
3	192	3	2

In this paper we use a residual network architecture with four layers, where each layer output is connected to the next two layers. Each layer, except the last, has 40 outputs and uses ReLU activation. No dropout or batch normalization is used as in the dense architecture. The skip connections are concatenated to the input of the layer they are arriving.

4. EVALUATION METHODOLOGY

We now briefly discuss the importance of the training and test sets when contrasting channel estimation using DL and GAMP. State-of-art GAMP-based algorithms consider the receiver knows the distribution $p(\mathbf{H})$ of channels but not its realizations. Knowing distributions for GAMP (even if not their parameters) and having large datasets for DL, are similar in the sense that both are manifestations of access to a potentially infinite amount of data. One distinction is that GAMP variants leverage the analytical expression of $p(\mathbf{H})$ as a highly compact representation of knowledge about the channels. In contrast, the NN is expected to find its own way of representing all relevant information contained in $p(\mathbf{H})$. To accomplish that, the ML-based method would ideally rely on access to a reasonable number of realizations (composing a rich training set from e. g. measurement data) or a software routine to draw samples from this distribution on-the-fly. If this is not the case, and the available data is restricted, it is important to clarify the scope in which the simulations are valid, specially when using ML-based approaches. For example, if the assessment methodology uses only matched conditions, in which the test and training sets are derived under the same circumstances (e. g., a statistical model or ray-tracing simulations from the same 3D scenario as in [25]), this restriction must be highlighted.

In this paper we train an NN under different noise conditions (*multi-condition* training) and different σ values, aiming that the NN will be capable of generalizing. Both noise multi-condition training and the time-variant channel are challenging for the stochastic gradient descent (SGD) used in DL. The network training with SGD may not converge even with advanced Keras' optimizers such as *Adam*. Most SGD routines obtain the gradient estimate by averaging the individual gradients of a set of B examples called *mini batch*. Having $B > 1$ often helps convergence by averaging the noise out and may be essential when the SNR imposed during training is low. But in the case of time-varying channels (examples corresponding to eventually distinct channels $\mathbf{H_v}$) and noisy measurements, SGD may perform better by not changing the channel within a mini batch, but keeping it constant over $B \times T$ consecutive symbols.

The simulation results are measured in normalized mean squared error (NMSE), defined as

$$\text{NMSE} = \frac{\mathbb{E}\left[\left\|\mathbf{H} - \hat{\mathbf{H}}\right\|^2\right]}{\mathbb{E}\left[\|\mathbf{H}\|^2\right]}, \qquad (6)$$

where $\mathbf{H}$ is the channel and $\hat{\mathbf{H}}$ is its estimation. The software used for DL simulations is written in the *Python* programming language using the *Keras* API with a *Tensorflow* backend.

Training is done by minimizing the MSE loss function using the *Adam* optimizer. We use a maximum number of epochs $N_e = 100$ with an early stopping mechanism for the case when the validation error starts to increase in order to avoid overfitting. Often, the training stops before reaching maximum epochs. This indicates a local minimum for the validation loss, and further training would potentially cause overfitting.

4.1. Random sparse MIMO channels

We used the stochastic channel model adopted in [12], which allows controlling the sparsity level and is useful in simulations of mmWave massive MIMO. The L multipath components (MPCs) in the virtual domain $\mathbf{H_v}$ are assumed to coincide with DFT bins (there is no leakage). Then, as indicated by Eq. (3), an inverse 2D DFT generates the corresponding $\mathbf{H}$ [21]. We control the sparsity of $\mathbf{H_v}$ by controlling the range of values of L. In this case, the signals (e. g. $\mathbf{X}$) and channels ($\mathbf{H_v}$ and $\mathbf{H}$) are generated on-the-fly by implementing the Keras Sequence interface, eliminating the need for storage of large datasets.

4.2. MIMO Channels from ray-tracing simulations

We used the methodology proposed in [26] to generate MIMO channels with Remcom's Wireless InSite ray-tracing simulator. The 3D scenario represented a 337×202 m^2 area of Rosslyn, Virginia, which is part of Wireless InSite's examples. The open source *Simulation of Urban MObility* (SUMO) [27] was repeatedly used to generate at each time instant t, the positions of all moving objects (cars and trucks) along the streets with a sampling interval T_s.

For each scene, Wireless InSite ray-tracing simulator is executed and information about the $L = 100$ strongest rays (gain, phase, angles, etc.) to each receiver is stored in a database. The process is repeated until the target number of simulations is reached. We simulated a *fixed wireless* scenario, in which the receivers are not mobile but located in houses and business offices. The adopted carrier frequency was 60 GHz, $T_s = 1$ sec, the transmitter antenna height was 5 m and *diffuse scattering* was enabled.

In order to implement the *Keras Sequence* interface, we load a dataset containing the simulated ray-tracing channels and generate $\mathbf{X}_{\text{pilot}}$ and $\tilde{\mathbf{Y}}_{\text{pilot}}$, so we also eliminate the need to store them.

5. SIMULATION RESULTS

Given $\mathbf{H}$, from the ray-tracing or random datasets, we obtain $\tilde{\mathbf{Y}}_{\text{pilot}}$ from $\mathbf{X}_{\text{pilot}}$. The implemented Keras Sequence interface generates a sample with $\tilde{\mathbf{Y}}_{\text{pilot}}$ as the input and $\mathbf{H}$ as the output. An epoch uses 1000 mini-batches, where each batch contains 5 samples. The validation set contains 200 mini-batches previously extracted from the Sequence interface. For the test, a set of size 500 is used. $N_e = 100$ epochs were used to train each model. Typical training times on a 8-core machine with a GPU were less than an hour.

In order to assess the performance of EM-BG-GAMP under conditions where its assumptions do not hold, we simulated the case where the quantizer follows our non-ideal model varying the σ of the Gaussian distribution of t_{down} and t_{up} from $\sigma = 0$ (ideal quantizer) to $\sigma = 0.5$. Fig. 3 shows that for $\sigma = 0$ is the ideal case with no hysteresis, where the EM-BG-GAMP performs at its best. Using the non-ideal quantizer model, EM-BG-GAMP suffers performance losses due to the assumptions not being true anymore. As σ increases, the performance decreases as there is a model mismatch for the quantization probability distribution. The NMSE increase on high SNRs is a well-known phenomenon on GAMP context called "stochastic resonance", which is described in detail in [28].

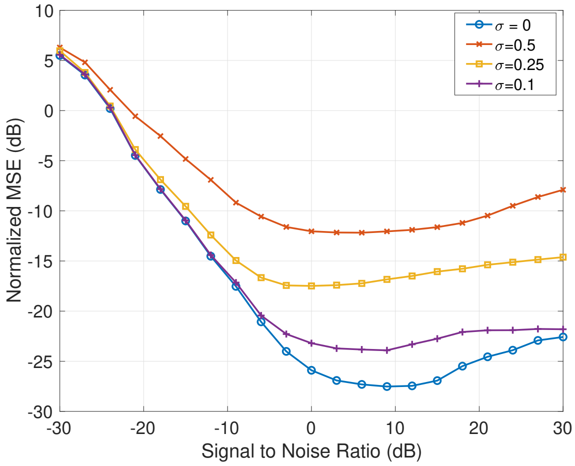

Fig. 3. Impact of ADC hysteresis on EM-BG-GAMP MSE.

5.1. Random sparse dataset results

Using the residual network model proposed in [29], we achieved the results in Fig. 4. A single network was trained on a multicondition style, with respect to both SNR and σ. We used a batch size $B = 5$, an SNR range uniformly distributed between $[-1, 1]$, and a σ uniformly distributed in the range $[0, 0.5]$. We can note that the increase in σ greatly impairs EM-BG-GAMP performance, while the residual network is nearly unaffected. In fact, for $\sigma = 0.5$, the residual network performed better than for $\sigma = 0$.

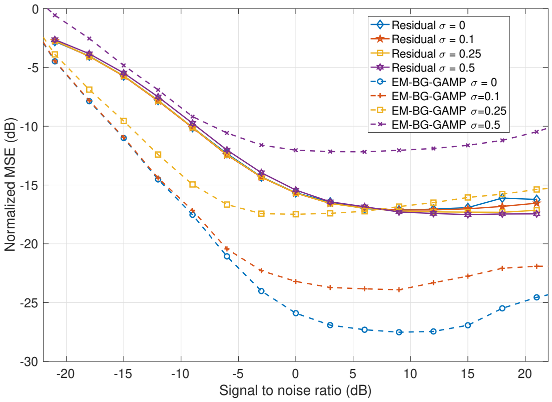

Fig. 4. Comparison of the residual network in the presence of hysteresis quantizers.

Fig. 5 shows the NMSE of the proposed models when trained with a ray-tracing dataset for $\sigma = 0$ to $\sigma = 0.5$. The dense network performed better in a wider SNR range. On the other hand, the convolutional network achieved lower NMSE on the high SNR range. Although the residual network could not outperform the others, it has half of the layers of the dense network and also less neurons per layers and requires fewer training epochs to converge.

The better performance of the NNs when compared to GAMP is due to less variability in the dataset channels. We want to avoid situations in which the NNs simply "memorize" the training data. In order to enforce that, we use different episodes for training and test, avoiding that the mobile objects in the 3D scenarios are located in similar positions.

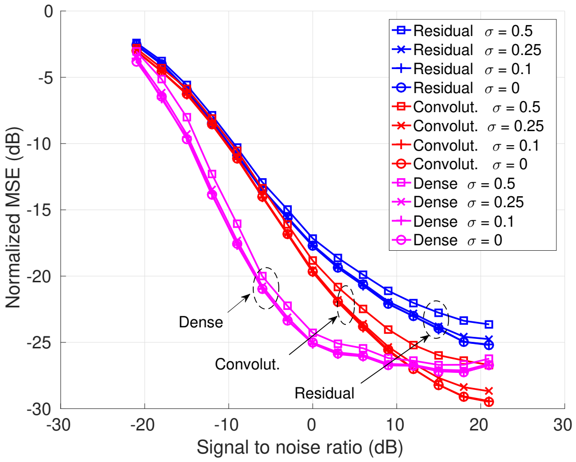

Fig. 5. Comparison of DL models in the presence of 1-bit ADCs with hysteresis for ray-tracing channels.

Fig. 6 compares the EM-BG-GAMP results with the dense network. For the ray-tracing case, the dense network outperforms EM-BG-GAMP by more than 5 dB at high SNRs. EM-BG-GAMP underperformed on the ray-tracing channels when compared to the random sparse channels.

6. CONCLUSIONS

In this paper, we compared the performance of state-of-art GAMP based algorithm with an NN approach in the presence of a hysteresis quantizer that does not meet GAMP assumptions. In contrast, the DL-based methods can create an internal model for this impairment during training. For the random channels, the results show that when the impairment level increases, in our case $\sigma = 0.25$ and $\sigma = 0.5$, EM-BG-GAMP suffers performance reduction and is outperformed by the residual network. For the ray-tracing datasets, the NNs benefited from less channel variability and achieved better performance than for the random sparse dataset. On the other hand, EM-BG-GAMP had poor performance for the used ray-tracing channels, even with the ideal 1-bit ADC case. This performance reduction is under further investigation.

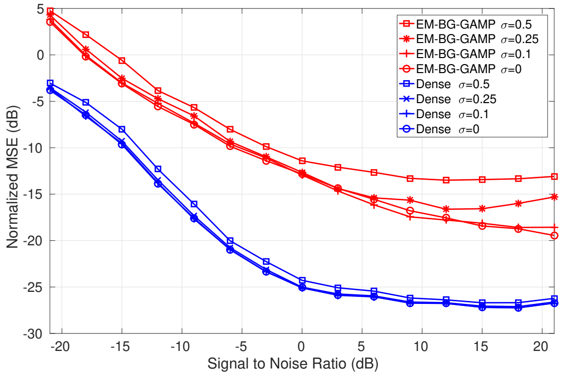

Fig. 6. Comparing EM-BG-GAMP with dense network for ray-tracing channels.

7. REFERENCES

[1] Y. Yang, Y. Li, K. Li, S. Zhao, R. Chen, J. Wang, and S. Ci, "DECCO: Deep-Learning Enabled Coverage and Capacity Optimization for Massive MIMO Systems," *IEEE Access*, vol. 6, pp. 23361–23371, 2018.

[2] T. Wang, C. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep Learning for Wireless Physical Layer: Opportunities and Challenges," *China Communications*, vol. 14, no. 11, pp. 92–111, Nov 2017.

[3] K. Kim, J. Lee, and J. Choi, "Deep Learning Based Pilot Allocation Scheme (DL-PAS) for 5G Massive MIMO System," *IEEE Communications Letters*, vol. 22, no. 4, pp. 828–831, April 2018.

[4] F. Krzakala et al, *Statistical Physics, Optimization, Inference, and Message-Passing Algorithms*, Oxford University Press, 2016.

[5] S. Rangan, "Generalized Approximate Message Passing for Estimation with Random Linear Mixing," in *2011 IEEE International Symposium on Information Theory Proceedings*, July 2011, pp. 2168–2172.

[6] J. T. Parker, P. Schniter, and V. Cevher, "Bilinear Generalized Approximate Message Passing—Part I: Derivation," *IEEE Transactions on Signal Processing*, vol. 62, no. 22, pp. 5839–5853, Nov 2014.

[7] J. T. Parker and P. Schniter, "Parametric Bilinear Generalized Approximate Message Passing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 795–808, June 2016.

[8] C. Wen, C. Wang, S. Jin, K. Wong, and P. Ting, "Bayes-Optimal Joint Channel-and-Data Estimation for Massive MIMO With Low-Precision ADCs," *IEEE Transactions on Signal Processing*, vol. 64, no. 10, pp. 2541–2556, May 2016.

[9] P. Sun, Z. Wang, R. W. Heath, and P. Schniter, "Joint Channel-Estimation/ Decoding With Frequency-Selective Channels and Few-Bit ADCs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 899–914, Feb 2019.

[10] J. Mo, P. Schniter, and R. W. Heath, "Channel Estimation in Broadband Millimeter Wave MIMO Systems With Few-Bit ADCs," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1141–1154, March 2018.

[11] J. Mo, P. Schniter, and R. W. Heath, "Channel Estimation in Broadband Millimeter Wave MIMO Systems With Few-Bit ADCs," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1141–1154, Mar. 2018.

[12] J. Mo, P. Schniter, N. G. Prelcic, and R. W. Heath, "Channel Estimation in Millimeter Wave MIMO Systems with One-Bit Quantization," in *Proc. 48th Asilomar Conference on Signals, Systems and Computers*, Nov. 2014, pp. 957–961.

[13] Yucheng Wang, Wei Xu, Hua Zhang, and Xiaohu You, "Wideband mmWave Channel Estimation for Hybrid Massive MIMO with Low-Precision ADCs," *arXiv e-prints*, p. arXiv:1809.04454, Sep 2018.

[14] Jiang Zhu, Chao-kai Wen, Jun Tong, Chongbin Xu, and Shi Jin, "Gridless Variational Bayesian Channel Estimation for Antenna Array Systems with Low Resolution ADCs," *arXiv e-prints*, p. arXiv:1906.00576, Jun 2019.

[15] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li, "Deep Learning-based Channel Estimation for Beamspace mmWave Massive MIMO Systems," *arXiv:1802.01290 [cs, math]*, Feb. 2018.

[16] P. Dong, H. Zhang, G. Y. Li, N. NaderiAlizadeh, and I. S. Gaspar, "Deep CNN for Wideband Mmwave Massive Mimo Channel Estimation Using Frequency Correlation," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 4529–4533.

[17] Peihao Dong, Hua Zhang, Geoffrey Ye Li, Navid NaderiAlizadeh, and Ivan Simoes Gaspar, "Deep CNN based Channel Estimation for mmWave Massive MIMO Systems," *arXiv e-prints*, p. arXiv:1904.06761, Apr 2019.

[18] Eren Balevi and Jeffrey G. Andrews, "Deep Learning-Based Channel Estimation for High-Dimensional Signals," *arXiv e-prints*, p. arXiv:1904.09346, Apr 2019.

[19] J. Liu, K. Mei, X. Zhang, D. Ma, and J. Wei, "Online Extreme Learning Machine-based Channel Estimation and Equalization for OFDM Systems," *IEEE Communications Letters*, pp. 1–1, 2019.

[20] R. W. Heath, N. González-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An Overview of Signal Processing Techniques for Millimeter Wave MIMO Systems," vol. 10, no. 3, pp. 436–453, Apr. 2016.

[21] Akbar M Sayeed, "Deconstructing Multiantenna Fading Channels," *IEEE Transactions on Signal Processing*, vol. 50, no. 10, pp. 2563–2579, 2002.

[22] Texas Instruments, "SNAA077C AN-156 Specifying A/D and D/A Converters," May 2004.

[23] Y. S. Jeon, S. N. Hong, and N. Lee, "Blind Detection for MIMO Systems with Low-Resolution ADCs Using Supervised Learning," in *Proc. IEEE Int. Conf. on Communications (ICC)*, May 2017, pp. 1–6.

[24] François Chollet, *Deep Learning with Python*, Manning Publications, 2017.

[25] Xiaofeng Li, Ahmed Alkhateeb, and Cihan Tepedelenlioğlu, "Generative Adversarial Estimation of Channel Covariance in Vehicular Millimeter Wave Systems," Aug. 2018.

[26] Aldebaro Klautau, Pedro Batista, Nuria González-Prelcic, Yuyang Wang, and Robert W. Heath, "5G MIMO Data for Machine Learning: Application to Beam-Selection using Deep Learning," in *Proc. Inf. Theory Appl. Workshop (ITA)*, 2018.

[27] Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker, "Recent Development and Applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, Dec. 2012.

[28] L. Gammaitoni, "Stochastic resonance and the dithering effect in threshold physical systems," vol. 52, pp. 4691–4698, Nov. 1995.

[29] A. Klautau, N. González-Prelcic, A. Mezghani, and R. W. Heath, "Detection and Channel Equalization with Deep Learning for Low Resolution MIMO Systems," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Oct 2018, pp. 1836–1840.