Learned Trimmed-Ridge Regression for Channel Estimation in Millimeter-Wave Massive MIMO

Pengxia Wu, Julian Cheng, *Fellow, IEEE*, Yonina C. Eldar, *Fellow, IEEE*,
and John M. Cioffi, *Life Fellow, IEEE,*

Abstract—Channel estimation poses significant challenges in millimeter-wave massive multiple-input multiple-output systems, especially when the base station has fewer radio-frequency chains than antennas. To address this challenge, one promising solution exploits the beamspace channel sparsity to reconstruct full-dimensional channels from incomplete measurements. This paper presents a model-based deep learning method to reconstruct sparse, as well as approximately sparse, vectors fast and accurately. To implement this method, we propose a trimmed-ridge regression that transforms the sparse-reconstruction problem into a least-squares problem regularized by a nonconvex penalty term, and then derive an iterative solution. We then unfold the iterations into a deep network that can be implemented in online applications to realize real-time computations. To this end, an unfolded trimmed-ridge regression model is constructed using a structural configuration to reduce computational complexity and a model ensemble strategy to improve accuracy. Compared with other state-of-the-art deep learning models, the proposed learning scheme achieves better accuracy and supports higher downlink sum rates.

I. INTRODUCTION

Millimeter Wave (mmWave) massive multiple-input multiple-output (MIMO) has become a key technology in wireless communications by offering a high data rate and spectrum efficiency for wide frequency bands and large antenna arrays [1]. Accurate channel state information (CSI) is crucial for achieving the expected array gains, as beamforming relies on CSI to direct narrow beams precisely towards the intended users [2]. To reduce the hardware and energy cost, current mmWave massive MIMO systems often adopt a hybrid analog digital (AD) architecture having a few radio frequency (RF) chains whose number is less than the number of antennas [2]. Under such an architecture, conventional channel estimation schemes can be time-costly and computational-intensive because they require many pilots to acquire enough channel observations. On the other hand, if the number of pilots is insufficient, conventional channel estimation schemes often fail to estimate channels accurately. By contrast, beamspace channel estimation methods are desirable when the angular-domain (beamspace) channels are

approximately sparse and thus can be reconstructed from a few pilot observations [3].

Existing beamspace channel estimation approaches can be categorized into compressive-sensing-based and deep-learning-based schemes. The compressive-sensing-based techniques regard the pilot training and channel reconstruction process as a compressive-sensing process and reconstruct the beamspace channels using compressive-sensing algorithms [4], [5]. Many compressive-sensing-based channel-estimation schemes were proposed for beamspace channel estimation in massive MIMO [6]–[14] and mmWave massive MIMO systems [15]–[17], including methods developed to reduce feedback overhead [8], [10], [13]. Greedy algorithms such as OMP [18] were used for massive MIMO channel estimation in [11], [19], [20]. Some OMP variants exploited the common-support feature between subcarriers or different users [12], [16], [21]. Sparse Bayesian learning (SBL) was explored for massive MIMO channel estimation and several variants exploited channel temporary correlations and common-sparsity structures in different subcarriers [22], [23]. An online Bayesian learning algorithm was proposed to relieve the computational burden [24]. The approximate message passing (AMP) algorithm [25] was also used for massive MIMO channel estimation. Many of these existing methods rely on the common-support sparsity structure that exist in different subcarrier channels or different users' channels. Their performance degrades in practice when the assumptions of channel sparsity or common-support sparsity structure are damaged by power leakage and beam squint effects [26], [27].

Deep-learning-based schemes are data-driven and establish models based on empirical experience and then use massive data to train the model and obtain an approximate solution. Many deep-learning-based channel estimation schemes have shown satisfactory performance using general-purpose deep-network models [28]–[34], including the fully-connected deep neural network (DNN), the convolutional neural network (CNN), and the recurrent neural network (RNN). However, one major shortcoming of the deep-learning-based approach is its 'black box' nature, and as a result, the developed channel-estimation schemes are difficult to interpret and analyze. Also, the built models often have large size and require massive training data and extensive parameter tuning by trial and error.

Model-based deep-learning schemes have drawn increasing attention because they leverage the well-established theoretical foundations—such as a proper system model, the known signal statistics, or existing estimation and detection algorithms—into designing a more efficient deep-learning network [35],

P. Wu was with the School of Engineering, The University of British Columbia, Kelowna, BC V1X 1V7, Canada (email: pengxia.wu@alumni.ubc.ca).

J. Cheng is with the School of Engineering, The University of British Columbia, Kelowna, BC V1X 1V7, Canada (email: julian.cheng@ubc.ca).

Yonina C. Eldar is with the Math and CS Faculty, Weizmann Institute of Science, Rehovot 7610001, Israel (e-mail: yonina.eldar@weizmann.ac.il).

John M. Cioffi is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: cioffi@stanford.edu).

[36]. The model-based deep-learning models not only result in reduced sizes, better interpretation, and smaller training data sets, they also achieve higher accuracy and faster convergence [37]. Deep unfolding is one such promising method that transforms a traditional iterative algorithm into a trainable deep-learning network [38], [39]. Various deep unfolding algorithms were developed for sparse reconstruction, such as the learned iterative shrinkage-thresholding algorithm (LISTA) [40] and the learned denoising-based approximate message passing network (LDAMP) [41]. Variant algorithms were developed for CSI acquisition applications, such as the LDAMP and learned AMP (LAMP) [42] for beamspace channel estimation [28], [42] and the basis-pursuit autoencoder for pilot matrix design [43].

This work proposes a deep unfolding algorithm to provide fast and accurate beamspace channel estimation for mmWave massive MIMO. The top-$\ell_{K,2}$ norm [44] is integrated into deep-unfolding networks to promote reconstruction sparsity and accuracy. To design such an unfolding network, we first leverage the top-$\ell_{K,2}$ norm to derive an exact sparse reconstruction solution which works perfectly with noiseless measurements and known sparsity level. However, implementing such an iterative algorithm is challenging in practice because it has high computational complexity and degraded performance when the sparsity level is unknown and the measurements are noisy. To address the challenges in practical implementation, we use deep unfolding to transfer these iterations into a more robust and lower-computational learnable algorithm. The proposed deep unfolding algorithm does not rely on a Gaussian statistical prior as in SBL algorithms, nor the common-support structural sparsity assumptions as in block-sparse-recovery algorithms. Instead, the proposed method treats beamspace channels as generic sparse vectors with unknown sparsity levels, and then specifies the top-$\ell_{K,2}$ norm parameters to promote different sparsity levels in different models, and finally leverages model-averaging ensemble to boost the reconstruction accuracy. Our contributions are summarized as follows:

- The trimmed-ridge regression is proposed to formulate the sparse-reconstruction problem, leading to more accurate solutions than the Lasso and ridge regression. An iterative algorithm is derived to reconstruct exact-sparse samples precisely.
- A deep unfolding network is proposed to improve the robustness and reduce the computational time in practical applications. With an interpretable structure of only a few layers, the constructed deep network can reconstruct beamspace channels faster and more accurately than the iteraive algorithms.
- A specific configuration is proposed to reduce computational complexity without degrading the model-prediction accuracy. A model ensemble strategy is proposed to improve model-prediction accuracy. Numerical results confirm the superior performance of the proposed deep-learning scheme in terms of both channel-estimation error and downlink sum rate.

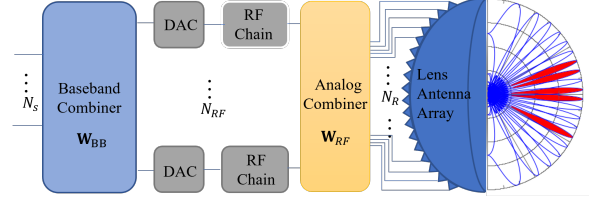This paper is organized into six sections. Section II formu-

Fig. 1: A mmWave massive MIMO system with a lens antenna array.

lates the problem of channel estimation in a mmWave massive MIMO system. Section III introduces the iterative trimmed regression algorithm, which is further expanded upon in Section IV, where the algorithm evolves into the learned trimmed ridge regression variant. Section V is dedicated to the exposition of numerical findings, while Section VI encapsulates conclusions.

II. SYSTEM MODEL

A. Channel Estimation in a MmWave Massive MIMO System

The system model is a mmWave massive MIMO system operating in time division duplex mode. As in Fig. 1, the base station (BS) has a lens antenna array with N antennas and N_{RF} RF chains, where $N_{RF} \ll N$. The BS serves U users and each user has a single antenna. For the uplink transmission, the model adopts a hybrid combiner as $\mathbf{W} = \mathbf{W}_{BB}\mathbf{W}_{RF} \in \mathbb{C}^{N_S \times N}$, where $\mathbf{W}_{BB} \in \mathbb{C}^{N_S \times N_{RF}}$ is the digital baseband combining matrix, $\mathbf{W}_{RF} \in \mathbb{C}^{N_{RF} \times N}$ is the analog combining matrix, and N_S denotes the number of independent data streams. The model includes the case where $N_S = N_{RF}$ and then sets $\mathbf{W}_{BB} = \mathbf{I}_{N_{RF}}$; thus, the combiner becomes $\mathbf{W} = \mathbf{W}_{RF} \in \mathbb{C}^{N_{RF} \times N}$, where the analog combiner network implements one-bit phase shifters. To estimate channels for all U users, each user repeatedly transmits a known pilot sequence $\mathbf{p}_u \in \mathbb{C}^U$ to the BS across B time blocks. Without loss of generality, the received pilots for a user at the mth time block are

$$\mathbf{r}_m = \mathbf{W}_m \sum_{u=1}^{U} \mathbf{h}_u \mathbf{p}_u^T \mathbf{p}_u^* + \mathbf{n}_{m,u} \qquad (1)$$
$$= \mathbf{W}_m \mathbf{h}_u + \mathbf{n}_{m,u},$$

where $\mathbf{r}_m$ is the pilot observation for the a user at the mth time block, $\mathbf{h}$ is the channel vector, and $\mathbf{n}_m$ is the effective noise vector.

For narrowband block fading channels, a spatial-domain channel can be expressed as [15]

$$\mathbf{h} = \sqrt{\frac{N_{BS}}{N_p}} \sum_{l=1}^{N_p} \beta^{(l)} \boldsymbol{\alpha}(\phi^{(l)}), \qquad (2)$$

where N_p is the number of scattering clusters, $l = 1$ is the index for the line-of-sight path, $2 \le l \le N_p$ is the index for non-line-of-sight (NLoS) paths, $\beta^{(l)}$ is the complex path gain, and $\boldsymbol{\alpha}(\phi^{(l)})$ is the array steering vector at the spatial direction $\phi^{(l)}$ for the lth path. The array steering vector contains a list of complex spatial sinusoids representing the relative phase shifts

of the incident far-field waveform across the array elements. For an N-element uniform linear array, $\boldsymbol{\alpha}(\phi^{(l)})$ is

$$\boldsymbol{\alpha}(\phi^{(l)}) = \frac{1}{\sqrt{N_{BS}}}[1, e^{-j2\pi\phi^{(l)}}, ..., e^{-j2\pi\phi^{(l)}(N_{BS}-1)}]^T, \quad (3)$$

where $\phi^{(l)}$ denotes the spatial direction of the lth path and is related to the physical angle $\theta^{(l)}$ by $\phi^{(l)} = d/\lambda \sin\theta^{(l)}$, where λ is the wavelength, and d is the antenna spacing. For $d = \lambda/2$, we have $-\frac{1}{2} \leq \phi^{(l)} \leq \frac{1}{2}$ and $-\frac{\pi}{2} \leq \theta^{(l)} \leq \frac{\pi}{2}$. The physical angle $\theta^{(l)}$ for the lth path is assumed to be uniformly distributed over $[-\pi/2, \pi/2]$. The NLoS path gain $\beta^{(l)}$ for $2 \leq l \leq N_p$ is assumed to follow the standard complex Gaussian distribution [7], [45].

The spatial channel vector $\mathbf{h}$ can be transformed into the beamspace channel representation $\mathbf{h}_b$ using a discrete Fourier transform (DFT) as

$$\mathbf{h}_b = \mathbf{U}\mathbf{h}, \quad (4)$$

where the DFT matrix $\mathbf{U} \in \mathbb{C}^{N \times N}$ is represented using a set of orthogonal array steering vectors as $\mathbf{U} = [\boldsymbol{\alpha}(\phi_1), \boldsymbol{\alpha}(\phi_2), ..., \boldsymbol{\alpha}(\phi_N)]^H$ [46], where $\phi_i = \frac{1}{N}(i - \frac{N+1}{2})$ for $i = 1, 2, ..., N$ is the virtual spatial direction predefined by the array having half-wavelength spaced antennas. The beamspace channel $\mathbf{h}_b$ is approximately sparse, which means that most of entries in $\mathbf{h}_b$ are nearly zeros. Since the lens antenna array can be designed to approximate an ideal DFT transmission [46], (1) can be written as

$$\begin{aligned} \mathbf{r}_m &= \mathbf{W}_m\mathbf{U}\mathbf{h} + \mathbf{n}_m \\ &= \mathbf{W}_m\mathbf{h}_b + \mathbf{n}_m, \end{aligned} \quad (5)$$

where $\mathbf{h}_b$ is the beamspace channel.

The pilot observations over B time blocks are

$$\mathbf{z} = \begin{bmatrix} \mathbf{r}_1 \\ ... \\ \mathbf{r}_B \end{bmatrix} = \begin{bmatrix} \mathbf{W}_1 \\ ... \\ \mathbf{W}_B \end{bmatrix} \mathbf{h}_b + \begin{bmatrix} \bar{\mathbf{n}}_1 \\ ... \\ \bar{\mathbf{n}}_B \end{bmatrix}. \quad (6)$$

By denoting $\mathbf{W} = [\mathbf{W}_1^T, ..., \mathbf{W}_B^T]^T$ and $\mathbf{n} = [\mathbf{n}_1^T, ..., \mathbf{n}_B^T]^T$, the pilot observation in (6) is compactly expressed as

$$\mathbf{z} = \boldsymbol{\Phi}\mathbf{h}_b + \mathbf{n}, \quad (7)$$

where $\mathbf{z} \in \mathbb{C}^M$ and $M = BN_{RF}$ is the total number of measurements. The measurement matrix $\boldsymbol{\Phi} \in \mathbb{C}^{M \times N}$ consists of the analog combiners $\{\mathbf{W}_m \in \mathbb{C}^{N_{RF} \times N}, 1 \leq m \leq B\}$, which are enabled by one-bit phase shifters, and therefore, entries of $\boldsymbol{\Phi}$ can be modeled to follow a Bernoulli distribution and are randomly selected from $\{-\frac{1}{\sqrt{M}}, \frac{1}{\sqrt{M}}\}$.

The beamspace channel estimation in (7) is equivalent to the two real-valued equations:

$$\begin{aligned} \Re(\mathbf{z}) &= \boldsymbol{\Phi}\Re(\mathbf{h}_b) + \Re(\mathbf{n}) \\ \Im(\mathbf{z}) &= \boldsymbol{\Phi}\Im(\mathbf{h}_b) + \Im(\mathbf{n}). \end{aligned} \quad (8)$$

We use the vector $\mathbf{x}$ to denote an arbitrary real-valued sample, i.e., $\mathbf{x} = \Re(\mathbf{h}_b)$ or $\mathbf{x} = \Im(\mathbf{h}_b)$, and use the vector $\mathbf{y}$ to denote a real-valued measurement vector, i.e., $\mathbf{y} = \Re(\mathbf{z})$ or $\mathbf{y} = \Im(\mathbf{z})$. The sparse channel reconstruction problem is then expressed as

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{x} + \mathbf{n}, \quad (9)$$

where $\mathbf{y} \in \mathbb{R}^M$, $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N}$, $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{n} \in \mathbb{R}^M$, and $M \ll N$. Since $\mathbf{x}$ is sparse, we can find it by solving

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \leq K. \end{aligned} \quad (10)$$

The ℓ_0-norm constraint $\|\mathbf{x}\|_0 \leq K$ in (10) results the problem NP-hard. A maximum a posteriori (MAP) estimator with a Laplacian prior, which is also known as the Lasso regression, i.e., $\min_{\mathbf{x}} \|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|_2^2 + \lambda_1\|\mathbf{x}\|_1$, is a widely-used approximate to (10). Another approximate solution is the ridge regression, i.e., $\min \frac{1}{2}\|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|_2^2 + \lambda_2\|\mathbf{x}\|_2^2$, which is a MAP estimator with a Gaussian prior but does not provide sparsity in general. The λ_1 and λ_2 denote regularization parameters on the ℓ_1- and ℓ_2-regularizers $\lambda_1\|\mathbf{x}\|_1$ and $\lambda_2\|\mathbf{x}\|_2^2$, respectively. A proposed solution here is a trimmed ℓ_2-norm regularizer to remove the penalties on a few large-magnitude elements and promote sparse solutions.

Since the large bandwidth is the main advantage of mmWave frequency band, it is reasonable to adopt the wide-band channel model and system model for mmWave massive MIMO systems [27], [47]. Although this work focuses on narrowband block-fading channels of massive MIMO, the proposed algorithms are generic sparse reconstruction algorithms and can conveniently extend to the wideband massive MIMO channels which have sparse representations in a virtual angular-delay-Doppler domain.

B. Trimmed-Ridge Regression

The top-$\ell_{K,2}$ norm permits reformulation of the sparse-reconstruction problem in (10). The top-$\ell_{K,2}$ norm [1] is the square sum of the K elements having the largest magnitudes in the vector $\mathbf{x}$, i.e., $\|\mathbf{x}\|_{K,2}^2 := x_{(1)}^2 + x_{(2)}^2 + \cdots + x_{(K)}^2$, where $x_{(i)}$ denotes the element whose magnitude is the ith-largest among the N elements of the vector $\mathbf{x}$, i.e., $|x_{(1)}| \geq |x_{(2)}| \geq \cdots \geq |x_{(K)}|$. Since $\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{K,2}^2 = 0$ is equivalent to $\|\mathbf{x}\|_0 \leq K$, the sparse reconstruction optimization in (10) can be rewritten as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{K,2}^2 = 0. \end{aligned} \quad (11)$$

Use of a Lagrange multiplier ρ transforms the constrained optimization (11) into an unconstrained optimization as

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|_2^2 + \rho(\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{K,2}^2) := F(\mathbf{x}). \quad (12)$$

The optimization in (12) is referred to as the trimmed-ridge regression. This regularizer in (12) helps achieve sparse and accurate solutions.

III. ITERATIVE TRIMMED-RIDGE REGRESSION ALGORITHM

This section describes a trimmed-ridge regression algorithm that uses DC programming to derive an iterative solution. A discussion on determining the algorithm step size then follows.

[1]In our prior work [14], the top-(K, 1) norm was proposed to develop an iterative algorithm for sparse channel reconstruction. However, unfolding the iterative algorithm that contains top-(K, 1) norm led to unsatisfying performance.

A. Deriving an Iteration Solution Using DC Programming

The trimmed-ridge regression (12) is a non-convex optimization. The objective $F(\mathbf{x})$ in (12) can be equivalently written as

$$F(\mathbf{x}) = \frac{k}{2}\|\mathbf{x}\|_2^2 - \frac{k}{2}\|\mathbf{x}\|_2^2 + \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \rho(\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{K,2}^2),$$
(13)

where $k \geq 0$ is a Lipschitz constant of the least squares objective $\frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$. We decompose the objective function $F(\mathbf{x})$ into convex functions $f(\mathbf{x})$ and $g(\mathbf{x})$ [2],

$$\min_{\mathbf{x}} \underbrace{\frac{k}{2}\|\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_2^2}_{f(\mathbf{x})} - \underbrace{\left(\frac{k}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_{K,2}^2\right)}_{g(\mathbf{x})}.$$
(14)

Next, the solution splits the positive and negative parts of $\mathbf{x}$ by letting $\mathbf{u} = (\mathbf{x})_+$ and $\mathbf{v} = (-\mathbf{x})_+$, where $(\mathbf{x})_+$ represents the operation of $(x)_+ = \max\{0, x\}$ for each element x in vector $\mathbf{x}$; $(-\mathbf{x})_+$ represents the operation of $(-x)_+ = \max\{0, -x\}$ for each element $-x$ in vector $-\mathbf{x}$. Hence, $\mathbf{u}$ and $\mathbf{v}$ are nonnegative vectors. Let $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$, $\mathbf{A} = [\mathbf{\Phi}, -\mathbf{\Phi}]$, and express (14) as

$$\min_{\mathbf{z} \succeq \mathbf{0}} \underbrace{\frac{k}{2}\|\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_2^2}_{f(\mathbf{z})} - \underbrace{\left(\frac{k}{2}\|\mathbf{z}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_{K,2}^2\right)}_{g(\mathbf{z})},$$
(15)

where $\mathbf{z} \succeq \mathbf{0}$ represents that $\mathbf{z}$ is a non-negative vector. DC programming solves (15) by repeating the following two steps until convergence: (a) compute the gradient (or subgradient) $\partial g(\mathbf{z}^{(t)})$ for an arbitrary tth iteration; (b) solve the convex subproblem $\min_{\mathbf{z} \succeq \mathbf{0}} f(\mathbf{z}) - \mathbf{z}^T \partial g(\mathbf{z}^{(t)})$ to obtain $\mathbf{z}^{(t+1)}$. The two steps specifically are

(a) $\quad \partial g(\mathbf{z}^{(t)}) = k\mathbf{z}^{(t)} - (\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\bar{\mathbf{z}}_K^{(t)}$

(b) $\quad \mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z} \succeq \mathbf{0}} \left\{\frac{k}{2}\|\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_2^2 - \mathbf{z}^T \partial g(\mathbf{z}^{(t)})\right\},$
(16)

where $\bar{\mathbf{z}}_K^{(t)}$ is the half of the gradient of top-$\ell_{K,2}$ norm while noting that $\bar{\mathbf{z}}_K^{(t)} = z_{(1)} + z_{(2)} + \cdots + z_{(K)}$, where $z_{(i)}$ denotes the element whose magnitude is the ith-largest among the elements of the vector $\mathbf{z}$, i.e., $z_{(1)} \geq z_{(2)} \geq \cdots \geq z_{(K)}$. The step (b) in (16) can be expressed as

$$\min_{\mathbf{z} \succeq \mathbf{0}} \quad \frac{k + 2\rho}{2}\left\|\mathbf{z} - \frac{1}{k + 2\rho}\partial g(\mathbf{z}^{(t)})\right\|_2^2.$$
(17)

A closed-form solution to (17) is the Euclidean projection of $\frac{1}{k+2\rho}\partial g(\mathbf{z}^{(t)})$ onto the nonnegative orthant, i.e.,

$$\mathbf{z}^* = \left(\frac{1}{k + 2\rho}\partial g(\mathbf{z}^{(t)})\right)_+,$$
(18)

[2] The convexity of $h(\mathbf{x})$ can be proved by showing it's Hessian matrix is semidefinite, i.e., $k\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi} \succeq \mathbf{0}$, for $k = \lambda_{max}(\mathbf{\Phi}^T\mathbf{\Phi})$, where $\lambda_{max}(\cdot)$ denotes the largest eigenvalue of a matrix [48].

where $(\cdot)_+$ denotes the positive-retaining operation that sets the negative elements to zeros. The DC programming procedure in (16) reduces to

(a) $\quad \partial g(\mathbf{z}^{(t)}) = k\mathbf{z}^{(t)} - (\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\bar{\mathbf{z}}_K^{(t)}$

(b) $\quad \mathbf{z}^{(t+1)} = \left(\frac{1}{k + 2\rho}\partial g(\mathbf{z}^{(t)})\right)_+.$
(19)

Thus, (19) is an iterative solution to the trimmed-ridge regression in (12). Algorithm 1 summarizes the iterations in (19). Since the iterations are derived under a DC programming framework, Algorithm 1 shares the same global convergence property as the general DC programming [49].

Algorithm 1 Iterative trimmed-ridge regression (ITRR)

Input: measurements $\mathbf{y}$, matrix $\mathbf{A}$ and a small number ϵ
Output: reconstructed $\hat{\mathbf{x}}$
Initialization: $\mathbf{u}^{(0)}, \mathbf{v}^{(0)}, \mathbf{z}^{(0)} \leftarrow [(\mathbf{u}^{(0)})^T, (\mathbf{v}^{(0)})^T]^T$

1: **for** $t = 0, 1, \ldots$ **do**
2: $\quad \nabla F(\mathbf{z}^{(t)}) \leftarrow \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho(\mathbf{z}^{(t)} - \bar{\mathbf{z}}_K^{(t)})$
3: $\quad \mathbf{z}^{(t+1)} \leftarrow \left(\mathbf{z}^{(t)} - \frac{1}{k+2\rho}\nabla F(\mathbf{z}^{(t)})\right)_+$
4: $\quad$ Check termination condition $\|\mathbf{z}^{(t+1)} - \mathbf{z}^{(t)}\|_2 \leq \epsilon$, return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^{(t+1)} = [(\mathbf{u}^{(t+1)})^T, (\mathbf{v}^{(t+1)})^T]^T$, and return $\hat{\mathbf{x}} = \mathbf{u}^{(t+1)} - \mathbf{v}^{(t+1)}$.
5: **end for**

B. Relaxing Step Sizes

Step 2 and Step 3 in Algorithm 1 can be combined to write the update for $\mathbf{z}^{(t+1)}$ as

$$\mathbf{z}^{(t+1)} = \left(\mathbf{z}^{(t)} - \frac{1}{k + 2\rho}\underbrace{\left(\mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho(\mathbf{z}^{(t)} - \bar{\mathbf{z}}_K^{(t)})\right)}_{\nabla F(\mathbf{z}^{(t)})}\right)_+.$$
(20)

We observe that (20) is a gradient-projection descent update with a step size of $1/(k + 2\rho)$ by defining $F(\mathbf{z}^{(t)}) = \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2^2 + \rho(\|\mathbf{z}\|_2^2 - \|\mathbf{z}\|_{K,2}^2)$. The fixed step size $1/(k + 2\rho)$, which is tied to the Lipschitz constant k, can lead to slow convergence. This motivates use of an accelerated step size strategy [50]. The monotonic Barzilai-Borwein (BB) step size is adopted, where the tth-step update becomes

$$\mathbf{w}^{(t+1)} = \left(\mathbf{z}^{(t)} - \alpha^{(t)}\nabla F(\mathbf{z}^{(t)})\right)_+,$$
$$\mathbf{z}^{(t+1)} = \mathbf{z}^{(t)} + \beta^{(t)}(\mathbf{w}^{(t+1)} - \mathbf{z}^{(t)}).$$
(21)

Here $\alpha^{(t)} > 0$ is the step size for the tth update and $\beta^{(t)} \in (0, 1]$ is the extrapolation parameter that prevents the step size from being too large to ensure the objective value descends monotonically. The step size $\alpha^{(t)}$ is explicitly computed by

$$\alpha^t = \frac{\|\mathbf{z}^t - \mathbf{z}^{t-1}\|^2}{(\mathbf{z}^t - \mathbf{z}^{t-1})^T (F(\mathbf{z}^t) - F(\mathbf{z}^{t-1}))}.$$
(22)

The extrapolation parameter β^t is often empirically set as $t/(t + 3)$ [50]. Due to the nonconvex property of $F(\mathbf{z}^{(t)})$, the iteration described in (21) lacks a theoretical guarantee

of convergence to the optimal solution. However, Section V's simulations demonstrate that the iterative algorithm with the BB step sizes achieves a consistently satisfactory level of reconstruction accuracy.

C. ITRR Performance on Reconstructing a Sparse Sample

This section shows the convergence property and the reconstruction accuracy of the ITRR algorithm when reconstructing a sparse sample. Simulations artificially produced a sparse sample, $\mathbf{h}_s$ with 16 nonzero entries from an arbitrary beamspace channel vector, by setting all other elements to zero except for the 16 largest-magnitude entries. The measurements $\mathbf{\Phi}\mathbf{h}_s$ were the input to obtain the reconstruction $\hat{\mathbf{h}}_b$. Both the ITRR using a fixed step size and the ITRR using the BB step size (ITRR-BB) were tested. The Lasso and ridge regressions were performed to reconstruct the same sparse sample using a similar iterative algorithm, i.e., the gradient-projection descent using the BB step size (PGD-BB). Only the results of the real part of $\mathbf{h}_s$ will be demonstrated. The results for the imaginary part are similar, because we treat the real part the imaginary part of a channel as two independent real-valued vectors as in (8) and (9).

Figure 2 shows the reconstructions against the true sparse sample. The ridge regression produces a noisy and inaccurate reconstruction having the normalized ℓ_2-error in the order of magnitude of -1. The Lasso regression produces a sparse reconstruction having a normalized ℓ_2-error in the order of magnitude of -6. The proposed ITRR algorithm produces an exact sparse reconstruction in the order of magnitude of -30. Both the ITRR and the ITRR-BB algorithms achieved similar reconstruction results in our numerical experiments. Optimization theory explains the difference in reconstruction accuracy: The ℓ_1-regularizer encourages sparsity but the same penalty on all element degrades accuracy. On the contrary, ℓ_2-regularizer discourages sparsity. Since ℓ_2-regularizer gradient is $\nabla\|\mathbf{x}\|_2^2 = 2\mathbf{x}$, the gradients of small absolute-value elements are small. Therefore, ℓ_2-regularizer yields diminishing gradients when elements values reduce to nearly zero, thus, the algorithm is more likely to converge before achieving a sparse solution. The proposed trimmed-ridge regression removes the penalties on large-magnitude elements to promote both the sparsity and accuracy.

Figures 3–5 demonstrate the convergence properties of the iterative algorithms when constructing the real part of a sparse sample. Fig. 3 shows the objective values versus iterations for ITRR and ITRR-BB algorithms. Fig. 4 shows the normalized ℓ_2-error of reconstructions over iterations. The ITRR and ITRR-BB reached lower reconstruction errors on the orders of magnitude of -30 and -31, while the Lasso and ridge regression algorithms achieved errors on the orders of magnitude of -6 and -1. The objective evolution in Fig. 3 and the reconstruction error evolution in Fig. 4 together suggest that using the BB steps size in an ITRR algorithm can accelerate the convergence without degrading the reconstruction accuracy. Fig. 5 shows the ℓ_2-norm values of the reconstruction vectors versus iterations, where the true ℓ_2-norm value is also plotted for comparison. The reconstructions by the ITRR-BB, the PGD-BB for Lasso regression, and the ITRR in turn

achieved the optimal ℓ_2-norm, whereas the PGD-BB for ridge regression left a constant gap between the reconstruction ℓ_2-norm and the optimal ℓ_2-norm. The ridge regression has an ℓ_2-regularizer that discourages sparsity and thus leads to an inaccurate reconstruction, while the proposed ITRR and ITRR-BB algorithm achieves an accurate solution. These results demonstrate the advantage of using the proposed top-$\ell_{K,2}$ to remove the penalties on K large magnitude elements from the conventional ℓ_2-regularizer.

IV. LEARNED TRIMMED-RIDGE REGRESSION ALGORITHM

While the ITRR algorithm offers the potential for direct beamspace channel estimation, its practical implementation can pose significant challenges. First, the computational complexity of ITRR increases rapidly with the problem dimension, i.e., the number of BS antennas of a massive MIMO system. Second, the regularization parameter ρ and the top-K parameter ideally should be tuned specifically for each channel realization, because using a fixed parameter setting can be suboptimal and consequently result in a degradation of the reconstruction performance. These two properties, which are not exclusive to ITRR but are commonly found in iterative and regularized regression algorithms, can result in slow convergence, making them unsuitable for real-time applications. A mmWave massive MIMO system has limited channel coherence time and thus requires low latency for channel-estimation algorithms. A deep-learning-based method is a promising alternative for practical online implementation due to its non-iterative and the inherent parallel computation ability. This motivates development of a model-based deep-learning approach that emulates the proposed ITRR algorithm, primarily because the ITRR-iteration qualities make it easy to comprehend, exhibit a straightforward computational structure, and demonstrate competitive reconstruction accuracy. The following sections first present the basic deep-network structure and then propose important configurations to reduce the computational complexity and a model-ensemble method to improve the reconstruction accuracy.

A. Unfolding Trimmed-Ridge Regression (UTRR) Structure

As in Fig. 6, the UTRR framework consists of a single-layer encoder from $\mathbf{x}$ to $\mathbf{y}$ and a multi-layer decoder from $\mathbf{y}$ to $\hat{\mathbf{x}}$. The encoder $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$ performs a linear dimension reduction. The input sparse vector $\mathbf{x}$ is projected onto the subspace spanned by the columns of the measurement matrix $\mathbf{\Phi}$ to obtain the compressed measurement vector $\mathbf{y}$. The decoder plays the role of the sparse reconstruction from $\mathbf{y}$ to $\hat{\mathbf{x}}$ in a feed-forward manner. The decoder is constructed to mimic the ITRR to obtain the reconstructed $\hat{\mathbf{x}}$. We can interpret the decoder by three functional parts, including the initializing layer, the unfolding layers and the output layer.

Initializing layer ($\mathbf{y}$ to $\mathbf{z}^{(0)}$): The first two layers can be understood as the initialization of ITRR, which originates from the input, i.e., the measurements $\mathbf{y}$ and obtain the initialized positive vector $\mathbf{z}^{(0)}$. We first initialize $\mathbf{x}^{(0)}$ to be $\mathbf{x}^{(0)} = \mathbf{\Phi}^T\mathbf{y}$. Then, we prepare an equivalent positive initialization vector $\mathbf{z}^{(0)}$ by separating and concatenating the positive and negative
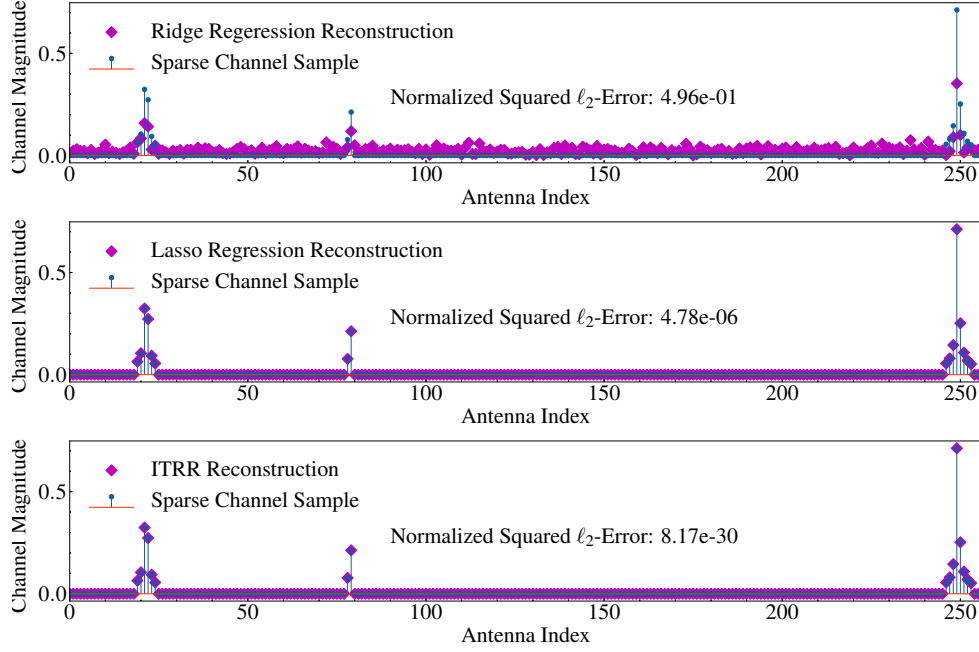
Fig. 2: Channel magnitudes of a sparse sample $\mathbf{h}_s$ and the reconstruction $\hat{\mathbf{h}}_s$ by the ridge regression, Lasso regression and the proposed ITRR algorithm, wherein the normalized squared ℓ_2-error is computed as $\|\Re(\mathbf{h}_s) - \Re(\hat{\mathbf{h}}_s)\|_2^2 / \|\Re(\mathbf{h}_s)\|_2^2$.
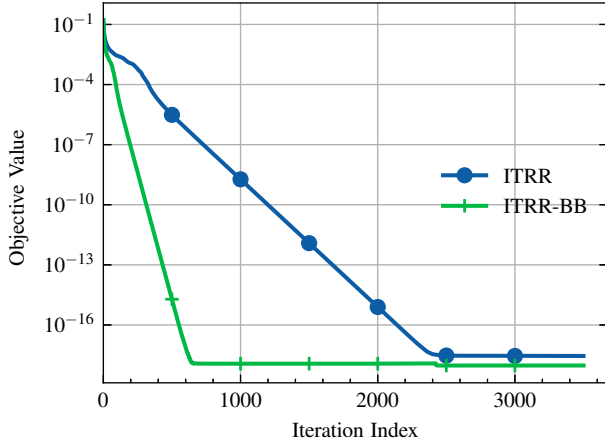

Fig. 3: Objective evolution when reconstructing the real part of the sparse sample $\Re(\mathbf{h}_s)$.

Fig. 4: Normalized reconstruction error evolution (i.e., $\|\Re(\hat{\mathbf{h}}_s) - \Re(\mathbf{h}_s)\|_2^2 / \|\Re(\mathbf{h}_s)\|_2^2$) when reconstructing the real part of the sparse sample

part of $\mathbf{x}^{(0)}$, that is $\mathbf{z}^{(0)} = [(\mathbf{u}^{(0)})^T, (\mathbf{v}^{(0)})^T]^T$, where $\mathbf{u}^{(0)} = \text{ReLU}(\mathbf{x}^{(0)})$ and $\mathbf{v}^{(0)} = \text{ReLU}(-\mathbf{x}^{(0)})$.

Unfolding layers ($\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L)}$): In unfolding layers, the iterations of the ITRR algorithm are stacked into multiple layers of the deep network by treating each iterative update step as a deep-network layer. The unfolding layers perform a forward update from $\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L)}$, which consists of L structural-identical blocks. Each block is a one-step iteration of the ITRR algorithm using the BB step-size structure. However, the proposed algorithm sets the step size $\alpha^{(t)}$ as a trainable parameter and sets the extrapolation parameter to be $\beta^{(t)} = 1/2$. This setting is equivalent to designing a residual learning unit [51] within each unfolding layer. The trainable weights of the tth unfolding layer are in $\mathbf{A}^{(t)} \in \mathbb{R}^{M \times 2N}$,
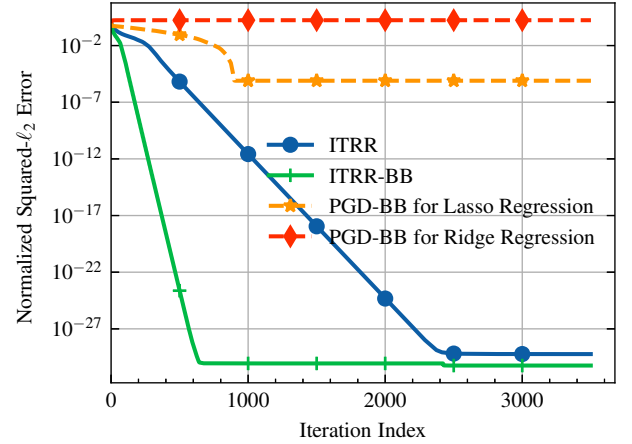
which is initialized as $[\boldsymbol{\Phi}, -\boldsymbol{\Phi}]$. The ReLU activation function in the unfolding layers is notably equivalent to the projection operations $(\cdot)_+$ in the ITRR updates.

Output layer ($\mathbf{z}^{(L)}$ to $\hat{\mathbf{x}}$): The last layer is the output layer to obtain the reconstructed vector $\hat{\mathbf{x}}$. Recalling that $\mathbf{z}^{(L)} = [z_1^{(L)}, ..., z_{2N}^{(L)}]$; therefore, we have $\hat{\mathbf{x}} = \mathbf{u}^{(L)} - \mathbf{v}^{(L)}$, where $\mathbf{u}^{(L)} = [z_1^{(L)}, ..., z_N^{(L)}]$ and $\mathbf{v}^{(L)} = [z_{N+1}^{(L)}, ..., z_{2N}^{(L)}]$.

B. Reducing Computational Complexity

The basic UTRR model structure in Fig. 6 helps specify the configurations in the unfolding layers to reduce computational complexity and improve model prediction accuracy. A UTRR
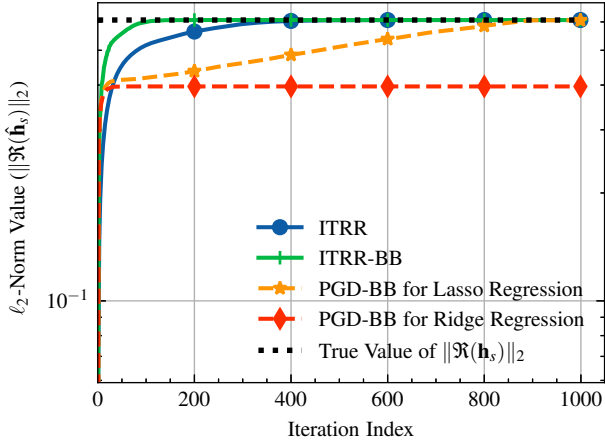
Fig. 5: The ℓ_2-norm value evolution when reconstructing the real part of the sparse sample

model has multiple identical unfolding layers from $\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L)}$, and each layer contains a $\bar{\mathbf{z}}_K^{(t)}$ term which is the gradient of the top-$\ell_{K,2}$ norm $\|\mathbf{z}\|_{K,2}^2$. The $\bar{\mathbf{z}}_K^{(t)}$ calculation must sort all elements of the vector $\mathbf{z}^{(t)}$. If such sorting operation is performed on every training sample within each layer, it causes a high computational complexity. To reduce this computational complexity, the top-K parameter is set to zero for all the intermediate layers from $\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L-1)}$ (except for the last layer $\mathbf{z}^{(L)}$). Thus, the unfolding iterations for the intermediate layers simplify to

$$\mathbf{w}^{(t+1)} = \text{ReLU}\big(\mathbf{z}^{(t)} - \alpha^{(t)}\big(\mathbf{A}^{(t)T}(\mathbf{y} - \mathbf{A}^{(t)}\mathbf{z}^{(t)}) + \rho\mathbf{z}^{(t)}\big)\big)$$
$$\mathbf{z}^{(t+1)} = (\mathbf{w}^{(t+1)} + \mathbf{z}^{(t)})/2$$
(23)

for $0 \leq t \leq L - 2$. Eq. (23) shows the $\bar{\mathbf{z}}_K^{(t)}$ term has been directly removed for the intermediate layers. Thus the computation of $\mathbf{w}^{(t+1)}$ for $t \in \{0, 1, ..., L-2\}$ can be simply interpreted as unfolding the projected gradient descent on traditional ridge regression. This configuration has negligible impact on the model-prediction performance and can reduce the computational complexity significantly (as shown in Section V-C).

C. Training Procedure

The goal of training a UTRR model is to produce an output vector $\hat{\mathbf{x}}$ from the noisy measurement vector $\mathbf{y}$ such that $\hat{\mathbf{x}} \approx \mathbf{x}$, where $\mathbf{x}$ represents the true channel vector. To this end, we should use a supervised learning training technique and prepare a dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$, where $\mathcal{D}_y = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$ include the noisy measurements and act as the training inputs; $\mathcal{D}_x = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ are the original channel samples and act as training labels. Given the dataset, the training process minimizes a loss function by optimizing the network weights (trainable parameters) using a backpropagation method. The loss function and trainable parameters are defined as follows:

Loss function: The loss function is defined by the mean square error of the reconstruction as

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \mathbb{E}[\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2].$$
(24)

Trainable parameters: The trainable parameters are $\{\mathbf{A}^{(t)}, \rho^{(t)}, \alpha^{(t)}\}$ for $t \in \{0, ..., L - 1\}$. The number of trainable parameters for a UTRR model would be $L(2MN + 2)$. Here, $\mathbf{A}^{(t)}$ is initialized to $[\mathbf{\Phi}, -\mathbf{\Phi}]$, where $\mathbf{\Phi}$ is the measurement matrix. In this work, we simply initialize the parameter $\rho^{(t)}$ to 1.0 and initialize the parameter $\alpha^{(t)}$ to 0.1 for all layers and for all models. Note that those initialization values for $\rho^{(t)}$ and $\alpha^{(t)}$ may not be the optimal. Tunning techniques for those hyperparameters to achieve their optimality are out of the scope of this paper. Practical cross validation strategies can help to choose proper the initialization values for different dataset. Given the training dataset, the trainable parameters use backpropagation to minimize the loss function in (24).

Algorithm 2 summarizes a UTRR model training procedure. The Step 3 to Step 11 perform forward computations of the UTRR model, and Step 12 performs the backward propagation to optimize the network trainable parameters, which can be performed automatically by deep learning platforms (e.g., Tensorflow and Pytorch).

Algorithm 2 UTRR Model Training Procedure

    Input: Dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$
    Output: Optimized $\{\mathbf{A}^{(t)}, \rho^{(t)}, \alpha^{(t)}\}$ for $t \in \{0, ..., L-1\}$
    Initialize: Set top-K parameter, initilize trainable parameters

1: **for** epoch $= 1, 2, ...$ **do**
2:     **for** batch $= 1, 2, ...$ **do**
3:         $\mathbf{x}^{(0)} \leftarrow \mathbf{\Phi}^T \mathbf{y}$
4:         $\mathbf{z}^{(0)} \leftarrow [(\mathbf{x}^{(0)})_+^T, (-\mathbf{x}^{(0)})_+^T]^T$
5:         **for** $t = 0, 1, ..., L-1$ **do**
6:             $\nabla F(\mathbf{z}^{(t)}) = \mathbf{A}^{(t)T}(\mathbf{y} - \mathbf{A}^{(t)}\mathbf{z}^{(t)}) + \rho(\mathbf{z}^{(t)} - \bar{\mathbf{z}}_K^{(t)})$
7:             $\mathbf{w}^{(t+1)} = \text{ReLU}\big[\mathbf{z}^{(t)} - \alpha^{(t)}\nabla F(\mathbf{z}^{(t)})\big]$
8:             $\mathbf{z}^{(t+1)} = (\mathbf{w}^{(t+1)} + \mathbf{z}^{(t)})/2$
9:         **end for**
10:        For $\mathbf{z}^{(L)} = [z_1^{(L)}, ..., z_{2N}^{(L)}]$, set $\mathbf{u}^{(L)} = [z_1^{(L)}, ..., z_N^{(L)}], \mathbf{v}^{(L)} = [z_{N+1}^{(L)}, ..., z_{2N}^{(L)}]$
11:        $\hat{\mathbf{x}} \leftarrow \mathbf{u}^{(L)} - \mathbf{v}^{(L)}$
12:        Minimize $\mathcal{L}$ in (24) and update parameters $\{\mathbf{A}^{(t)}, \rho^{(t)}, \alpha^{(t)}\}$ for $t \in \{0, ..., L-1\}$
13:     **end for**
14: **end for**

D. Model-Averaging Ensemble

The previous sections have explained the model structure and training procedure for a UTRR model. This section proposes averaging an ensemble of multiple UTRR models to improve further the reconstruction accuracy. Model-averaging ensemble was originally proposed to train multiple models over smaller-size subsets of data, and then combine the multiple prediction results as the final prediction result [52]. Nowadays, machine-learning engineers use model-averaging ensemble to reduce model uncertainty (to reduce prediction variance without increasing prediction bias) [53], because a model ensemble often performs better than any individual model (intuitively, errors from various models are averaged).
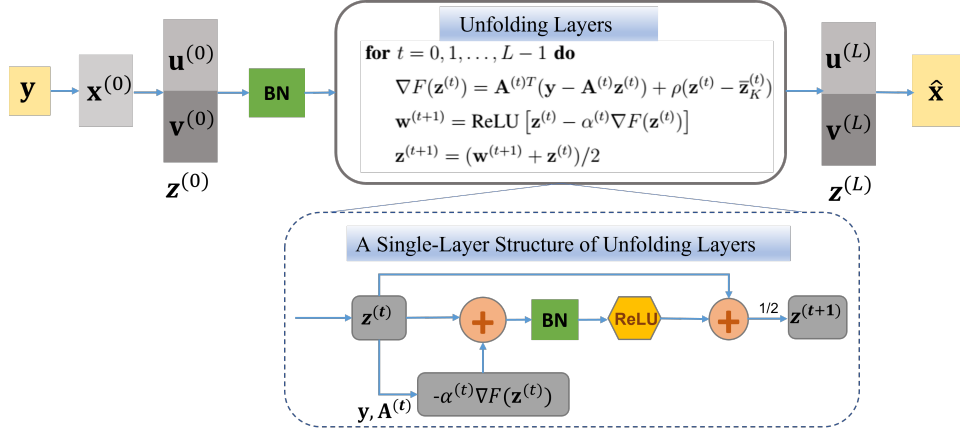
Fig. 6: The model structure of UTRR. BN represents a layer of Batch Normalization.

The simplest ensemble method averages multiple models' prediction results; alternatively, a weighted-average method can weigh models unequally according to their performance. The model weights can be optimized by a basic cross-validation method or by advanced algorithms. For example, a "meta-network" can be trained to treat each individual model as a "neuron" and serve the synaptic weights of the trained network as the optimized ensemble weight [54]. This work explores the simplest strategy, i.e., averaging the sum of each model reconstruction result by using constant and equal weights. Extending our work to find the optimal model-averaging-ensemble strategy remains an interesting research direction.

Since multiple diverse models are required to develop a model-averaging ensemble, we should first prepare several models that produce reconstructions having small bias and large variance. The UTRR is readily available to develop diverse models because the $\bar{\mathbf{z}}_K^{(t)}$ term in the last unfolding layer (i.e., $t = L - 1$) can be assigned with different values on the top-K parameter. The top-K parameter implies the assumed sparsity level on the trimmed-ridge regression problem. Then, multiple UTRR models with different top-K parameters can be assembled for final reconstruction. Recall that the top-K parameters are zeroed for the intermediate unfolding layers to avoid computing the top-$\ell_{K,2}$ norm derivative term, whereas the term $\bar{\mathbf{z}}_K^{(t)}$ in the last unfolding layer (i.e., from $\mathbf{z}^{(L-1)}$ to $\mathbf{z}^{(L)}$) plays an important role in a model-ensemble approach. In summary, a model-averaging ensemble includes three steps: 1) develop several UTRR models that have different values of the top-K parameters; 2) train each of the UTRR models using the same set of training data; 3) reconstruct the unseen beamspace channel by using each model and average their reconstructions. Fig. 7 shows the pipeline of using the UTRR model-averaging ensemble for beamspace channel estimation. The proposed UTRR is first trained offline to acquire the optimized model, and the optimized models are ensembled and implemented online to estimate the beamspace channels in real time. The third step should perform parallel computations on all the model reconstructions to avoid extra latency in a practical implementation.
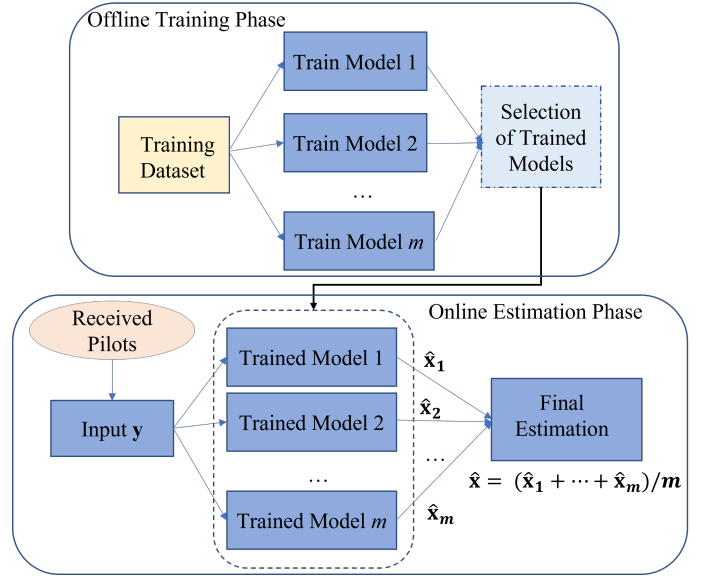

Fig. 7: Pipeline of using UTRR model-averaging ensemble for beamspace channel estimation

V. NUMERICAL RESULTS

This section reports the main numerical results relevant to the proposed algorithms. Simulation setups and performance metrics will be first discussed. Then, performance of the ITRR algorithm and the UTRR models will be evaluated and compared with other methods. Also, this section evaluates the UTRR performance enhancement by the proposed computational-complexity reducing strategy and the model-averaging ensemble.

A. Simulation Setup

Simulations consider a mmWave massive MIMO system that has $U = 16$ single-antenna users and a BS equipped with $N = 256$ antennas and $N_{RF} = 16$ RF chains. To estimate the channels for all users, each user repeatedly transmits a pilot sequence having the length of $T = 16$ over $B = 8$ blocks. Thus, the BS receives $M = 128$ measurements for each user, as described in (7). Note that $M = 128$ is a

default setting in our simulation unless explicitly denoting different values of M. Following these system parameters, we first use the Saleh-Valenzuela channel model in (2) to generate the spatial channel samples $\{\mathbf{h}_1, ..., \mathbf{h}_n\}$, and then use a DFT in (4) to transform the spatial channel samples into the beamspace channel samples $\{\mathbf{h}_{b,1}, ..., \mathbf{h}_{b,n}\}$. Each channel sample is normalized to eliminate the effect of large-scale fading. The measurements $\{\mathbf{z}_1, ..., \mathbf{z}_n\}$ are produced according to (7) by $\mathbf{z}_i = \mathbf{\Phi}\mathbf{h}_{b,i} + \mathbf{n}_i$ for $1 \leq i \leq n$, where $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ is a random matrix whose entries follow a Bernoulli distribution and are randomly selected from $\{-\frac{1}{\sqrt{M}}, \frac{1}{\sqrt{M}}\}$, and $\mathbf{n}_i$ is a Gaussian noise vector with a zero mean and a variance that is associated with the channel estimation signal-to-noise ratio (SNR), which is defined by $SNR = \mathbb{E}\left[\|\mathbf{\Phi}\mathbf{h}_b\|^2/\|\mathbf{n}\|^2\right]$. The training dataset, validation dataset and testing dataset contain $80,000$, $2,000$, and $2,000$ beamspace channels and measurements.

Experiments trained upon several UTRR models using different top-K parameters. The models were trained and evaluated individually, and then the trained models are used with model-averaging ensemble. In practice, if the sparsity level is known or estimatable, it is suggested to select different top-K parameters around the known/estimated sparsity level. In our simulations, we do not know the sparsity level of the beamspace channels since all the channels are approximately sparse instead of exact sparse. We randomly selected four top-K parameters $K \in \{0, 16, 64, 256\}$ for four UTRR models. Note that those model selections may not be the optimal choices. All UTRR models trained in this paper have 30 unfolding layers with the same training hyperparameters as follows. The diminishing leaning rates are adopted in the training process, and the learning rates are set to be 0.005, 0.001, 0.0005, 0.0002, and 0.0001. The maximum training epoch is set to be 300. Early stopping is adopted to avoid overfitting, and the tolerate epoch number is set to 10, which means training stops if the validation error has not decreased for 10 epoches. The batch size is set to be 128.

Analyses use the normalized mean square error (NMSE) to evaluate channel estimation accuracy and compute the downlink sum rate to evaluate the influence of imperfect channel estimation on the communication-link capacity. The channel estimation NMSE is defined by $\mathbb{E}\left[\frac{(\|\mathbf{h}_b - \hat{\mathbf{h}}_b\|_2^2)}{\mathbf{h}_b}\right]$, where $\mathbf{h}_b$ denotes the true beamspace channel and $\hat{\mathbf{h}}_b$ denotes the estimated beamspace channel. The downlink sum rate is $\mathbb{E}\left[\sum_{u=1}^{U} \log_2(1 + \frac{\|\mathbf{h}_{b,u}^H \mathbf{f}_u\|^2}{\sum_{i \neq u} \|\mathbf{h}_{b,u}^H \mathbf{f}_i\|^2 + 1/SNR_{dl}})\right]$, where $\mathbf{f}_u$ denotes the precoder for the uth user, and SNR_{dl} denotes the downlink transmit SNR. For the downlink transmission that uses zero-forcing precoding, the transmission model can be represented as $y_u = \sqrt{P_u}\mathbf{h}_u^T \mathbf{f}_u \mathbf{s}_u + n_u$, where $u = 1, 2, ...U$ represents the uth user, The downlink SNR is defined as P_t/σ_u^2, where the P_t is the transmission power and σ_u^2 is noise power spectrum. A zero-forcing precoder is adopted and defined as $\tilde{\mathbf{F}} = \hat{\mathbf{H}}_b(\hat{\mathbf{H}}_b^H \hat{\mathbf{H}}_b)^{-1}$, where $\hat{\mathbf{H}}_b$ is the estimated uplink channels for U users. The analysis assumes an ideal channel reciprocity and uses the estimated uplink channels as the downlink channels without considering a channel-reciprocity

error. For simplicity, the power allocation is not considered in simulating the sum rates; instead, each user's precoder uses matrix normalization to meet a total power constraint, i.e., $\mathbf{f}_u = \tilde{\mathbf{f}}_u / \|\mathbf{F}\|_F$, where $\tilde{\mathbf{f}}_u$ is the uth column of the precoder matrix $\tilde{\mathbf{F}}$. Performance comparison with the proposed methods is of two iterative algorithms including the OMP [18] and AMP [25] algorithms, and three deep-learning methods including a model-based network GM-LAMP [45], a model-free CNN-based network [33] and a sparse Bayesian learning unfolding network SBL-CE-Net [26]. Note that the GM-LAMP has superior performance than LAMP [45], and LAMP has superior performance than LISTA [55]. The SBL-CE-Net method has demonstrated superior performance than a group of other methods including the MMV-LAMP [56] and the SBL family [22], [23]. Furthermore, results in [56] and [22] have already demonstrated significant performance superiority than some other greedy algorithms such as SOMP, group-LASSO, and MMV-AMP [26].

B. ITRR and UTRR Performance Evaluations on Reconstructing Beamspace Channels

In practice, beamspace channels are approximately sparse, containing most small-valued entries instead of zeros. It is of interest to examine the ITRR performance on reconstructing the testing dataset, which contains $2,000$ approximately beamspace channels, and compare with other widely-used iterative sparse reconstruction algorithms, including the OMP [18] and AMP [25] algorithms. Fig. 8 shows the results of channel estimation NMSE versus SNR. The ITRR algorithm shows a better accuracy compared with the OMP algorithm over a wide SNR range, and has similar performance as the AMP algorithm. Since the AMP and OMP algorithms have been successfully used in various compressive-sensing tasks, this result confirms the ITRR algorithm also has an competitive accuracy and can also be applied in similar applications. However, it is worth to mention the limitation of ITRR in the real-time applications that have low latency requirements. First, the iterative mechanism requires sufficient computation time to achieve expected accuracy, but additional delay is undesirable. Second, the top-K parameter as well as the regularization parameter ρ in an ITRR algorithm needs to be manually evaluated and selected for better performance. Ideally, those parameters should be adaptively tuned according to the real channels to achieve best performance. Certain model-selection strategies such as cross validation can help to ensure overall performance for slowing changing environment. However, it is unrealistic to obtain the K and ρ that are ideal for all individual channel realizations.

The proposed deep-learning method can overcome the above limitations. The UTRR models used in our simulation have only 30 unfolding layers, which is 20 to 100 times less than the number of ITRR iterations. Moreover, the trained models can be directly used to reconstruct a batch of channels simultaneously in parallel without tuning any parameters. For example, is our simulation, we set the batch size as 128, which
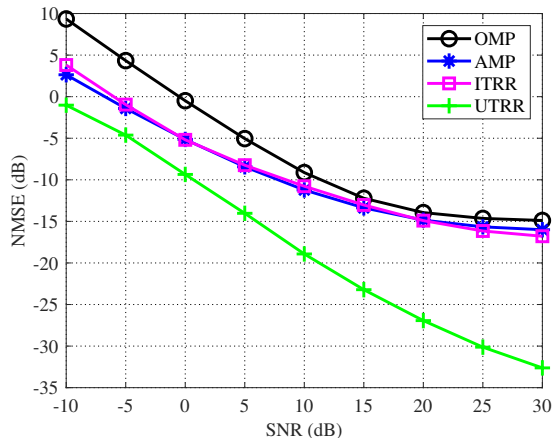
Fig. 8: Reconstruction NMSE of the iterative algorithms and the UTRR

TABLE I: Comparisons of the training time and reconstruction errors between a UTRR model with and without computational-complexity reduction (CCR). Here, the channel estimation SNR is 20 dB, and the top-K parameter of the UTRR model is 16.

Models	NMSE	Training Time
UTRR$_{K=16}$ w/o CCR	-26.78 dB	548 min
UTRR$_{K=16}$ w/ CCR	-26.93 dB	255 min

means 64 [3] users' channels can be estimated simultaneously at one-time computation. It is of interest to evaluate the UTRR performance and compare with the ITRR algorithm over the same testing dataset. Fig. 8 includes the reconstruction NMSE by a trained UTRR model, where the top-K parameter of the model is set to be $K = 16$. Compared with the iterative algorithms, the UTRR reduces the NMSE about 5 dB at a low SNR, about 10 dB at a moderate SNR, and about 15 dB at a high SNR.

C. UTRR Performance Evaluation on Reducing Computational Complexity

Section IV-B proposed the strategy of reducing computational complexity (RCC) for the UTRR model. The computational complexity can be reduced by simply setting the parameter $K = 0$ within the intermediate unfolding layers. This section uses Table I to present the training time and reconstruction NMSE at $SNR = 20$ dB for a UTRR model with the RCC configuration and compare with the results of the UTRR model without the RCC configuration. In Table I, the UTRR$_{K=16}$ without RCC is a UTRR model that contains 30 layers of identical unfolding layers, where each layer has the top-K parameter as $K = 16$; the UTRR$_{K=16}$ with RCC is a UTRR model that sets the top-K parameter $K = 0$ for the 29 layers of intermediate unfolding layers, while the top-K parameter $K = 16$ is for the last unfolding layer. The UTRR model with a RCC and the UTRR model without a RCC have a similar NMSE but the former only costs about 50% training time [4]. Similar results have been observed in other SNR range and for other UTRR models having different the top-K parameters. These results support our claim that the proposed RCC strategy does not degrade the model-prediction performance but can reduce the training time significantly.

D. UTRR Performance Evaluation on Model-Averaging Ensemble

Section IV-D proposed the model-averaging ensemble to improve the reconstruction accuracy. In a model-averaging-ensemble approach, several UTRR models having different top-K parameters are trained individually, and then in the inference phase, channel estimations from different UTRR models are averaged as a final estimation. Table II lists the NMSE results for different UTRR models having different top-K parameters as well as the NMSE results for the final ensemble model UTRR-E at $SNR = 10$ dB and $SNR = 20$ dB. All the UTRR models with different top-K parameters produced a similar NMSE, but by averaging the predictions from individual models, the UTRR-E produces a lower NMSE by around 4 dB. Table III measures the ratio of accurately reconstructed channels in the testing dataset, where an accurate reconstruction is counted when its normalized ℓ_2 error is below a threshold, which is set as $Th = 0.001$ for $SNR = 10$ dB and $Th = 0.0001$ for $SNR = 20$ dB. At both SNRs, Table III shows the model-averaging ensemble improved the accurate reconstruction ratio by around 50%. The results of NMSE and the accurate reconstruction ratio from Table II and Table III prove that the proposed model-averaging ensemble can improve the reconstruction accuracy significantly. Note that assembling multiple models that have the same the top-K parameter would not produce such improvement. In practical applications, since the computational complexity from the summation and average operations are negligible, the predictions from each trained model can be computed in parallel to avoid adding additional delays.

E. Performance Comparison for Different Pilot Lengths

The performance of UTRR models for pilot length $M = 64$ are also added in Table II and Table III to show the pilot length influence. Table II shows when using the pilot length of $M = 64$, the NMSE performance degrades around 5 dB compared with the pilot length $M = 128$. Table III shows the model-averaging ensemble improved the accurate reconstruction ratio by around 50% when $M = 128$. When $M = 64$, the accurate reconstruction ratio improves by 10%–15% by model-averaging ensemble compared to the results from each individual model.

Figure 9 compares the NMSE performance for the proposed approaches for the pilot lengths $M = 128$ and $M = 64$. Iterative aglorithms ITRR and ITRR-BB have similar accuracy performance, therefore, we only demonstrate ITRR-BB accuracy in this figure. It shows that more pilots lead to higher accuracy for all approaches. Compared with ITRR-BB, the

---

[3]The number is half of the batch size because we stack the real part and column part of a beamspace channels in column wise. Therefore, one beamspace channel is equivalent to two parallel real-valued vectors.

[4]The training was performed by a Nvidia RTX A6000 GPU.

TABLE II: Reconstruction NMSEs of different UTRR models and the ensembled model.

NMSE(dB)＼Model SNR	$\text{UTRR}_{K=0}$	$\text{UTRR}_{K=16}$	$\text{UTRR}_{K=64}$	$\text{UTRR}_{K=256}$	UTRR-E
$M = 128, SNR = 10$ dB	-18.92	-18.92	-19.00	-18.77	-22.06
$M = 128, SNR = 20$ dB	-26.97	-26.93	-26.73	-26.94	-30.18
$M = 64, SNR = 20$ dB	-21.89	-21.22	-21.72	-21.45	-24.51

TABLE III: The accurate reconstruction ratios of different UTRR models and the ensembled model. Here, an accurate reconstruction is counted if its normalized ℓ_2-error below the threshold Th.

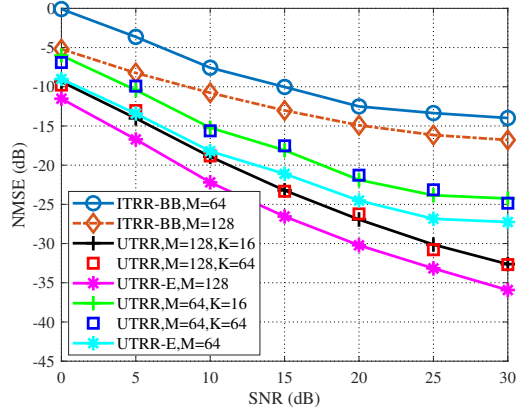| Ratio (%)＼Model Threshold	$\text{UTRR}_{K=0}$	$\text{UTRR}_{K=16}$	$\text{UTRR}_{K=64}$	$\text{UTRR}_{K=256}$	UTRR-E
$M = 128, Th = 0.01, SNR = 10$ dB	39.1	38.65	40.75	34.6	85.15
$M = 128, Th = 0.001, SNR = 20$ dB	10.85	10.6	8.15	11.75	67.4
$M = 64, Th = 0.01, SNR = 20$ dB	85.8	81.4	85.2	83.1	95.6

Fig. 9: Reconstruction NMSEs of deep-learning methods.
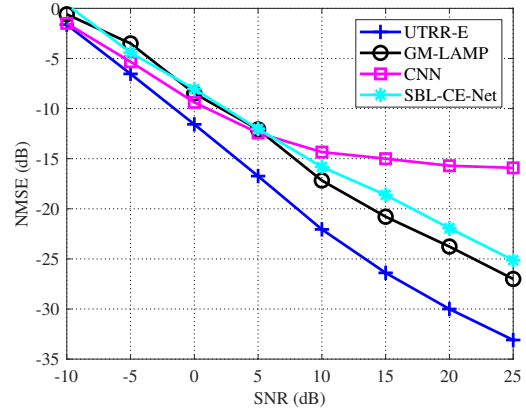

Fig. 10: Reconstruction NMSEs of deep-learning methods.

unfolded networks UTRR and UTRR-E using the same pilot length achieved higher accuracy. For UTRR and UTRR-E, more pilots lead to more noticeable NMSE reduction at high SNR range between 20–30 dB.

F. Comparisons of UTRR Performance With Other Deep-Learning Methods

Next, we compare the UTRR-E performance with the GM-LAMP model [45], the CNN-based model [33], and the SBL-CE-Net [26]. Fig. 9 plots the channel estimation NMSE over the SNR in the range of $[-10, 25]$ dB. Compared with the GM-LAMP and the CNN-based model, the proposed UTRR-E achieved lower NMSEs for the entire SNR range. The CNN-based model has a slightly lower NMSE than the GM-LAMP for the low SNR range of $[-10, 5]$ dB, but at a high $SNR \in [10, 25]$ dB, the NMSE of the CNN-based model stuck at around -15 dB while other methods start showing noticeable advantages. For example, at the high $SNR = 25$ dB, the SBL-CE-Net has the NMSE around -25 dB, the GM-LAMP achieves the NMSE around -26 dB, and the proposed UTRR-E achieves the NMSE below -33 dB.

As a widely used metric to evaluate the channel estimation accuracy, the NMSE itself is insufficient to measure the impact of imperfect channel estimation on the communication system. Because the estimated channels from two different channel

estimation approaches can have the same NMSE but have significantly distinct channel realizations. Different channel realizations having different values in each antenna entry can lead to different beamforming directions and thus cause a significantly different impact on the achievable rate of the massive MIMO system.

Figure 10 plots the sum rate of using the estimated beamspace channels by the proposed UTRR-E and also plots the sum rate of using the Gaussian noise-contaminated CSI. The x-axis is the channel estimation NMSE to ensure the sum rates are computed at the same NMSE for two types of imperfect CSI. Large gaps exist between the sum rates of using the UTRR-E estimated CSI and using the Gaussian noise contaminated CSI. This result affirms that different channel estimation schemes can have the same NMSE but have a significantly different impact on the sum rate. Therefore, the sum rate is a necessary metric to access the direct impact of channel estimation schemes on a communication system.

Figure 11 compares the downlink sum rates versus the transmit SNR for different deep-learning channel-estimation methods, where the ideal channel reciprocity is assumed and the uplink channels were estimated at $SNR = 15, 0, -10$ dB. When the uplink channels are estimated at a high SNR, the channel estimation errors are small enough. Thus, the downlink sum rates are dominated by the downlink SNR. Therefore, all three deep-learning methods show little differ-
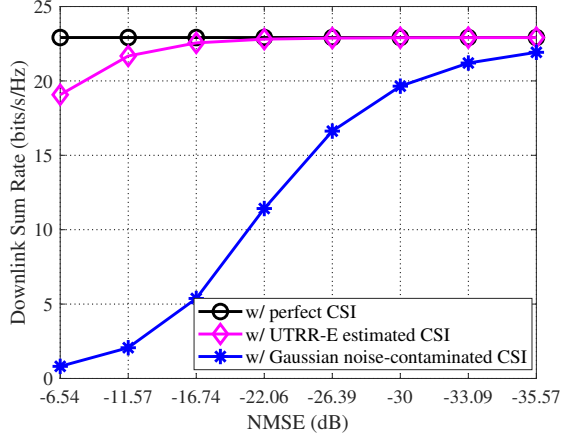
Fig. 11: Downlink sum rate at $SNR_{dl} = 15$ dB versus different channel estimation NMSE

Fig. 13: Downlink sum rate at $SNR_{dl} = 20$ dB versus uplink channel estimation SNR
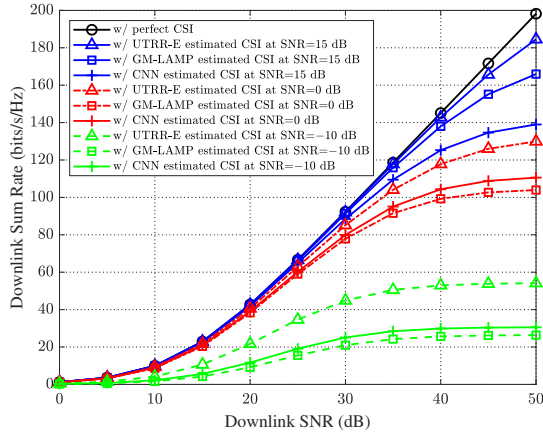

Fig. 12: Downlink Sum rate versus downlink SNR when uplink channels are estimated at $SNR = -10, 0, 15$ dB

ence in the sum rates. For example, for the uplink channels are estimated when the channel estimation SNR is 15 dB, the average channel estimation errors are -26.39 dB, -20.79 dB, and -15 dB for the UTRR-E, the GM-LAMP and the CNN-based model, respectively; their sum rates are 143, 138, and 125 bits/s/Hz for the downlink $SNR = 40$ dB, while the sum rate with perfect CSI is 145 bits/s/Hz that only has little difference with the sum rates using imperfect CSI. The uplink channel-estimation errors increase when the uplink SNR decreases, and thus the downlink sum rates degrade more obviously because of the imperfect CSI. As shown in the sum-rate curves when the channel estimation $SNR = -10$ dB, larger gaps exist between the sum rates using imperfect CSI and the sum rate using perfect CSI. For all the cases of different channel estimation SNRs, the proposed UTRR-E model achieved higher sum rate over the entire downlink SNR range. Fig. 12 shows the impact of the channel estimation SNR on the downlink sum rate, where the downlink transmit SNR is $SNR_{dl} = 20$ dB. When the channel estimation SNR increases, the sum rates of all three deep-learning methods increase and gradually approach their maximum sum rates. The optimal
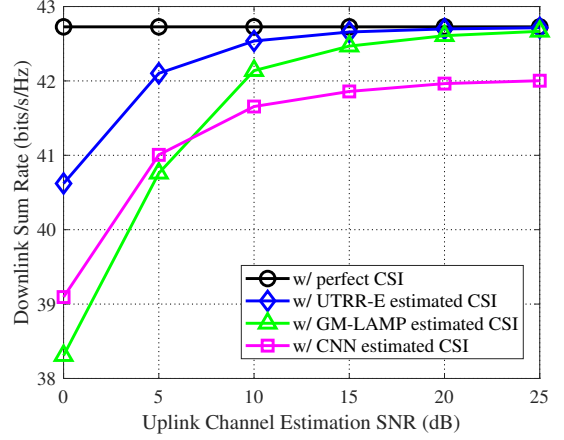
sum rate with perfect CSI is 43 bits/s/Hz, which is drawn as a straight line for comparison. The UTRR-E achieves a 41 bits/s/Hz sum rate, which is 95% of the optimal sum rate at $SNR = 0$ dB; at $SNR = 10$ dB, the UTRR-E achieves the optimal sum rate. For the entire uplink channel estimation SNR range, the UTRR-E shows higher sum rates than the other two deep-learning methods. The sum rate results of Fig. 11 and Fig. 12 conclude that the proposed UTRR models with model-averaging ensemble can estimate the channels more accurately and provide higher sum rate than the state-of-the-art other deep-learning methods.

G. Computational Complexity

The computational complexity of the ITRR algorithm is $O(TMN^2)$, where T represents the number of iterations before convergence. The forward-pass computational complexity of a UTRR network is $O(LMN^2)$, where L represents the number of layers of the decoder. Although the computational complexities of the ITRR and the UTRR decoder appear similar, the number of layers L is usually much smaller than the number of iterations I. Additionally, the sorting operation of finding largest K elements has computational complexity $O(N \log K)$. ITRR algorithm requires to perform sorting operation in every iteration while the UTRR with CRR only requires once in the last layer of the network. By contrast, the GM-LAMP network are constructed on the AMP algorithm, the computational complexity of the AMP algorithm and the GM-LAMP network is the same, i.e., $O(TMN)$ [45]. The computational complexity of the CNN-based model is $O(NM)$ for each fully connected layer and $O(Nk^2 \Sigma_{l=1}^{N_{re}} n_{l-1} n_l)$ for each CNN layer, where k is the side length of the convolutional filters, n_{l-1} and n_l denote the numbers of input and output feature maps of the $l$the convolutional layer, $1 \leq l \leq N_{re}$ [33]; seven CNN layers were used with the kernel size 3×3 [33]. The computational complexity of the OMP algorithm can be represented by $O(SMN) + O(S^3 M)$, where S is the sparsity level of the beamspace channel vector [57]. The computational complexity for SBL family is $O(TMN^2)$. Note that the computational

complexity is quite high because calculating matrix inverse is required for two times per iteration, and the computational complexity for matrix inverse is $O(M^3)$. The proposed SBL-CE-Net introduces extra computational complexity by the 3D convolutional layers is $O(NN_F F_S^3)$, where N_F is the number of filters NF filters and the filter size is F_S [26].

VI. CONCLUSION

This paper presented an iterative algorithm, i.e., the ITRR, and then unfolded it into a deep-learning model, i.e., the UTRR, for the beamspace channel estimation in mmWave massive MIMO systems. The ITRR algorithm has interpretable structure and shows competitive performance, while the practical implemetation is challenging due to high computational complexity and suboptimal parameter tunning. By unfolding the ITRR into a learnable algorithm which can be trained offline, the UTRR addressed the practical implementation challenges with faster, more robust and accurate sparse reconstructions. Furthermore, this paper proposed the model-averaging ensemble to improve the model prediction accuracy. Numerical results conclude that the proposed learning scheme can reconstruct beamspace channels with a higher accuracy and thus prompts the mmWave massive MIMO system to achieve larger sum rates. This work demonstrated a complete design of leveraging traditional model-driven optimizations to develop a deep-learning method for an applied problem. The proposed UTRR models can be applied to other sparse reconstruction applications.

REFERENCES

[1] R. W. Heath Jr., N. González-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 3, pp. 436–453, Apr. 2016.

[2] S. A. Busari, K. M. S. Huq, S. Mumtaz, L. Dai, and J. Rodriguez, "Millimeter-wave massive MIMO communication for future wireless systems: A survey," *IEEE Commun. Surveys Tut.*, vol. 20, no. 2, pp. 836–869, Second Quart. 2018.

[3] C. R. Berger, Z. Wang, J. Huang, and S. Zhou, "Application of compressive sensing to sparse channel estimation," *IEEE Commun. Mag.*, vol. 48, no. 11, pp. 164–174, Nov. 2010.

[4] Y. C. Eldar and G. Kutyniok, *Compressed Sensing: Theory and Applications*. New York, NY, USA: Cambridge University Press, 2012.

[5] Y. C. Eldar, *Sampling Theory: Beyond Bandlimited Systems*. New York, NY, USA: Cambridge University Press, 2015.

[6] Z. Gao, L. Dai, Z. Wang, and S. Chen, "Spatially common sparsity based adaptive channel estimation and feedback for FDD massive MIMO," *IEEE Trans. Signal Process.*, vol. 63, no. 23, pp. 6169–6183, Dec. 2015.

[7] Z. Gao, L. Dai, W. Dai, B. Shim, and Z. Wang, "Structured compressive sensing-based spatio-temporal joint channel estimation for FDD massive MIMO," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 601–617, Feb. 2016.

[8] M. E. Eltayeb, T. Y. Al-Naffouri, and H. R. Bahrami, "Compressive sensing for feedback reduction in MIMO broadcast channels," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3209–3222, Sept. 2014.

[9] J. W. Choi, B. Shim, and S. Chang, "Downlink pilot reduction for massive MIMO systems via compressed sensing," *IEEE Commun. Lett.*, vol. 19, no. 11, pp. 1889–1892, Nov. 2015.

[10] C.-C. Tseng, J.-Y. Wu, and T.-S. Lee, "Enhanced compressive downlink CSI recovery for FDD massive MIMO systems using weighted block l_1-minimization," *IEEE Trans. Commun.*, vol. 64, no. 3, pp. 1055–1067, Mar. 2016.

[11] W. Shen, L. Dai, Y. Shi, B. Shim, and Z. Wang, "Joint channel training and feedback for FDD massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 65, no. 10, pp. 8762–8767, Oct. 2016.

[12] X. Rao and V. K. N. Lau, "Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3261–3271, June 2014.

[13] A. Liu, F. Zhu, and V. K. N. Lau, "Closed-loop autonomous pilot and compressive CSIT feedback resource adaptation in multi-user FDD massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 65, no. 1, pp. 173–183, Jan. 2017.

[14] P. Wu and J. Cheng, "Nonconvex regularized gradient projection sparse reconstruction for massive MIMO channel estimation," *IEEE Trans. Commun*, vol. 69, no. 11, pp. 7722–7735, Nov. 2021.

[15] X. Gao, L. Dai, S. Han, C. I, and X. Wang, "Reliable beamspace channel estimation for millimeter-wave massive MIMO systems with lens antenna array," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 6010–6021, Sept. 2017.

[16] Z. Gao, C. Hu, L. Dai, and Z. Wang, "Channel estimation for millimeter-wave massive MIMO with hybrid precoding over frequency-selective fading channels," *IEEE Commun. Lett.*, vol. 20, no. 6, pp. 1259–1262, June 2016.

[17] P. S. Jianhua Mo and R. W. Heath Jr., "Channel estimation in broadband millimeter wave MIMO systems with few-bit ADCs," *IEEE Trans. Signal Process.*, vol. 66, no. 5, pp. 1141–1154, Mar. 2018.

[18] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inform. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[19] K. Venugopal, A. Alkhateeb, N. G. Prelcic, and R. W. Heath Jr., "Channel estimation for hybrid architecture-based wideband millimeter wave systems," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 9, pp. 1996–2009, Sept. 2017.

[20] J. P. González-Coma, J. Rodríguez-Fernández, N. González-Prelcic, L. Castedo, and R. W. Heath Jr., "Channel estimation and hybrid precoding for frequency selective multiuser mmWave MIMO systems," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 2, pp. 353–367, May 2018.

[21] J. Rodríguez-Fernández, N. González-Prelcic, K. Venugopal, and R. W. Heath Jr., "Frequency-domain compressive channel estimation for frequency-selective hybrid millimeter wave MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, pp. 2946–2960, May 2018.

[22] S. Srivastava, A. Mishra, A. Rajoriya, A. K. Jagannatham, and G. Ascheid, "Quasi-static and time-selective channel estimation for block-sparse millimeter wave hybrid MIMO systems: Sparse bayesian learning (SBL) based approaches," *IEEE Trans. Signal Process.*, vol. 67, no. 5, pp. 1251–1266, Mar. 2019.

[23] P. Wang, M. Pajovic, P. V. Orlik, T. Koike-Akino, K. J. Kim, and J. Fang, "Sparse channel estimation in millimeter wave communications: Exploiting joint AoD-AoA angular spread," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Taipei, Taiwan, May 2017, pp. 1–6.

[24] G. Joseph and C. R. Murthy, "Channel estimation for hybrid architecture based wideband millimeter wave systems," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5510–5525, Otc. 2017.

[25] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing," *Natl. Acad. Sci.*, vol. 106, no. 45, pp. 18 914–18 919, 2009.

[26] J. Gao, C. Zhong, G. Y. Li, and A. B. J. B. Soriaga, "Deep learning-based channel estimation for wideband hybrid mmwave massive MIMO," *IEEE Trans. Commun.*, vol. 71, no. 106, pp. 3679–3693, June 2023.

[27] Y. Chen, Y. Xiong, D. Chen, T. Jiang, S. X. Ng, and L. Hanzo, "Hybrid precoding for wideband millimeter wave MIMO systems in the face of beam squint," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 1847–1859, Mar. 2021.

[28] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmWave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, Oct. 2018.

[29] C.-J. Chun, J.-M. Kang, and I.-M. Kim, "Deep learning-based channel estimation for massive MIMO systems," *IEEE Commun. Lett.*, vol. 8, no. 4, pp. 1228–1231, Aug. 2019.

[30] C. K. Wen, W. T. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, Oct. 2018.

[31] T. Wang, C. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 416–419, Apr. 2019.

[32] Q. Yang, M. B. Mashhadi, and D. Gündüz, "Deep convolutional compression for massive MIMO CSI feedback," in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, Oct. 2019, pp. 1–6.

[33] X. Ma and Z. Gao, "Data-driven deep learning to design pilot and channel estimator for massive MIMO," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5677–5682, May 2020.

[34] M. B. Mashhadi and D. Gündüz, "Pruning the pilots: Deep learning-based pilot design and channel estimation for MIMO-OFDM systems,"

IEEE Trans. Wireless Commun, vol. 20, no. 10, pp. 6315–6328, Oct. 2021.

[35] N. Shlezinger, Y. C. Eldar, and S. P. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115 384–115 398, Nov. 2022.

[36] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *Proceedings of the IEEE*, vol. 111, no. 5, pp. 465–499, May 2023.

[37] H. He, S. Jin, C. Wen, F. Gao, G. Y. Li, and Z. Xu, "Model-driven deep learning for physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 77–83, Oct. 2019.

[38] J. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *Mitsubishi Electric Research Laboratories, Tech. Rep. TR2014-117*, Aug. 2014.

[39] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, Mar. 2021.

[40] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010, pp. 399–406.

[41] C. Metzler, A. Mousavi, and R. Baraniuk, "Learned D-AMP: Principled neural network based compressive image recovery," in *Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 2017, pp. 1772–1783.

[42] X. Ma, Z. Gao, F. Gao, and M. D. Renzo, "Model-driven deep learning based channel estimation and feedback for millimeter-wave massive hybrid MIMO systems," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2388–2406, 2021.

[43] P. Wu and J. Cheng, "Deep unfolding basis pursuit: Improving sparse channel reconstruction via data-driven measurement matrices," *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8090–8150, Oct. 2022.

[44] K. Tono, A. Takeda, and J. Gotoh., "Efficient DC algorithm for constrained sparse optimization," 2017. [Online]. Available: arXiv:1701.08498.

[45] X. Wei, C. Hu, and L. Dai, "Deep learning for beamspace channel estimation in millimeter-wave massive MIMO systems," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 182–193, Jan. 2021.

[46] J. Brady, N. Behdad, and A. M. Sayeed, "Beamspace MIMO for millimeter-wave communications: System architecture, modeling, analysis, and measurements," *IEEE Trans. Antennas Propag.*, vol. 61, no. 7, pp. 3814–3827, July 2013.

[47] Y. Chen, D. Chen, and L. Hanzo, "Channel-covariance and angle-of-departure aided hybrid precoding for wideband multiuser millimeter wave MIMO systems," *IEEE Trans. Commun.*, vol. 67, no. 12, pp. 8315–8327, Dec. 2019.

[48] P. Wu, "Sparse channel estimation for massive MIMO via deep compressive sensing," Ph.D. dissertation, University of British Columbia, 2021.

[49] H. A. L. Thi and T. P. Dinh, "DC programming and DCA: Thirty years of developments," *Math. Program.*, vol. 169, pp. 5–68, 2018.

[50] N. Parikh and S. P. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 123–231, 2014.

[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, Las Vegas, NV, USA, June 2016, pp. 770–778.

[52] P. Sollich and A. Krogh, "Learning with ensembles: How over-fitting can be useful," in *Advances in Neural Information Processing Systems (NIPS)*, 1995, pp. 190–196.

[53] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Networks*, vol. 12, no. 10, pp. 1399–1404, Dec. 1999.

[54] D. M. Jose, A. M. Vincent, and G. S. Dwarakish, "Improving multiple model ensemble predictions of daily precipitation and temperature through machine learning techniques," *Scientific Reports*, vol. 12, no. 4678, Mar. 2022.

[55] M. Borgerding, P. Schniter, and S. Rangan, "AMP-inspired deep networks for sparse linear inverse problems," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4293–4308, Aug. 2017.

[56] X. Ma, Z. Gao, F. Gao, and M. D. Renzo, "Model-driven deep learning based channel estimation and feedback for millimeter-wave massive hybrid MIMO systems," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2388–2406, Aug. 2021.

[57] K. Venugopal, A. Alkhateeb, N. G. Prelcic, and R. W. Heath, "Channel estimation for hybrid architecture based wideband millimeter wave systems," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 9, pp. 1996–2009, Sept. 2017.

Pengxia Wu received the B.S. degree in communication engineering in 2013 and the M.S degree in communication and information system in 2016 from Jilin University, Changchun, China. She received the Ph.D. degree in electrical engineering from the University of British Columbia, School of Engineering, Kelowna, BC, Canada, in 2022. She is currently a research scientist at Rockwell Automation, Milwaukee, WI, USA. Her research interests include wireless communication, robotics, and artificial intelligence in industrial automation.

Julian Cheng (S'96–M'04–SM'13–F'23) received a B.Eng. degree (Hons.) in electrical engineering from the University of Victoria, Victoria, BC, Canada, in 1995, the M.Sc.(Eng.) degree in mathematics and engineering from Queen's University, Kingston, ON, Canada, in 1997, and a Ph.D. in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2003. He is a Full Professor in the School of Engineering, Faculty of Applied Science, The University of British Columbia, Kelowna, BC, Canada. He was with Bell Northern Research and NORTEL Networks. His research interests include machine learning and deep learning for wireless communications, wireless optical technology, and quantum communications. He is a Registered Professional Engineer in British Columbia, Canada. He served as the President of the Canadian Society of Information Theory (2017-2021). He was the Co-Chair of the 12th Canadian Workshop on Information Theory in 2011, the 28th Biennial Symposium on Communications in 2016, and the General Co-Chair of the 2021 and 2024 IEEE Communication Theory Workshop. He is the Chair of the Radio Communications Technical Committee of the IEEE Communications Society. He was a past Associate Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE COMMUNICATIONS LETTERS, and the IEEE ACCESS, as well as an Area Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS. Dr. Cheng served as a Guest Editor for Special Issues of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS on Optical Wireless Communications and Positioning and Sensing Over Wireless Networks.

Yonina C. Eldar is a Professor in the Department of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel where she heads the center for Biomedical Engineering and Signal Processing and holds the Dorothy and Patrick Gorman Professorial Chair. She is also a Visiting Professor at MIT, a Visiting Scientist at the Broad Institute, and an Adjunct Professor at Duke University and was a Visiting Professor at Stanford. She is a member of the Israel Academy of Sciences and Humanities, an IEEE Fellow and a EURASIP Fellow. She received the B.Sc. degree in physics and the B.Sc. degree in electrical engineering from Tel-Aviv University, and the Ph.D. degree in electrical engineering and computer science from MIT, in 2002. She has received many awards for excellence in research and teaching, including the IEEE Signal Processing Society Technical Achievement Award (2013), the IEEE/AESS Fred Nathanson Memorial Radar Award (2014) and the IEEE Kiyo Tomiyasu Award (2016). She was a Horev Fellow of the Leaders in Science and Technology program at the Technion and an Alon Fellow. She received the Michael Bruno Memorial Award from the Rothschild Foundation, the Weizmann Prize for Exact Sciences, the Wolf Foundation Krill Prize for Excellence in Scientific Research, the Henry Taub Prize for Excellence in Research (twice), the Hershel Rich Innovation Award (three times), and the Award for Women with Distinguished Contributions. She received several best paper awards and best demo awards together with her research students and colleagues, was selected as one of the 50 most influential women in Israel, and was a member of the Israel Committee for Higher Education. She is the Editor in Chief of Foundations and Trends in Signal Processing, a member of several IEEE Technical Committees and Award Committees, and heads the Committee for Promoting Gender Fairness in Higher Education Institutions in Israel.

John M. Cioffi – Illinois-BSEE: 1978, Stanford-PhDEE: 1984; Prof. EE, Stanford, 1986-present, now recalled emeritus. Bell Labs 1978-84; IBM Research 84-86. Founder Amati 1991 (1997 purchased by TI); Chairman and CEO ASSIA Inc. Cioffi's specific interests are in the area of high-performance digital transmission. Awards include USA's Medal of Technology (2023), IEEE AG Bell (2010), Kirchmayer (2014) and Millennium Medals; Member Internet (2014) and Consumer-Electronics (2018) Halls of Fame; Marconi Fellow (2006); Member, US National (2001), UK Royal (2009), and Scottish Royal (2024) Engineering Academies; IEEE Kobayashi (2001), Armstrong (2014), and Women-In-Coms-Mentor (2018) Awards. Cioffi has served over a dozen boards of directors, presently PhyTunes (Chairman) and Marconi Society, along with Advisory Boards of VectorML and Zenith Aerospace. 800+ papers and 150+ heavily licensed patents.