Learning in Time-Frequency Domain for Fractional Delay-Doppler Channel Estimation in OTFS

Sandesh Rao Mattu and A. Chockalingam

***Abstract*—In this letter, we propose a learning-based approach for estimation of fractional delay-Doppler (DD) channel in orthogonal time frequency space (OTFS) systems. A key novelty in the proposed approach is that learning is done in the time-frequency (TF) domain for DD domain channel estimation. Learning in the TF domain is motivated by the fact that the range of values in the TF channel matrix is favorable for training as opposed to the large swing of values in the DD channel matrix which is not favourable for training. A key beneficial outcome of the proposed approach is its low complexity along with very good performance. Specifically, it drastically reduces the complexity of the computation of a constituent DD parameter matrix (CDDPM) in a state-of-the-art algorithm. Simulation results show that the proposed TF learning-based algorithm achieves almost the same performance as that of the state-of-the-art algorithm, while being drastically less complex making it practically appealing.**

***Index Terms*—OTFS, machine learning, channel estimation, delay-Doppler domain, time-frequency domain, low-complexity.**

I. Introduction

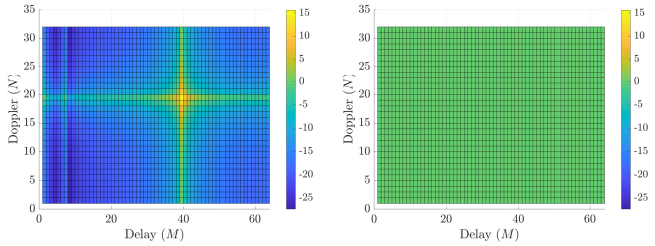
Orthogonal time frequency space (OTFS), a modulation scheme that multiplexes information symbols in the delay-Doppler (DD) domain has been shown to be well suited for high-Doppler channels [1],[2]. Signal detection and channel estimation are crucial OTFS receiver functions [2]. A computationally efficient OTFS signal detector for integer DD channels assuming perfect channel knowledge has been proposed in [3], where it is emphasized that fractional DD channel estimation is an important research topic. Several traditional techniques and a few machine learning-based techniques have been investigated for the purpose of DD channel estimation [4]-[11]. Machine-learning techniques implemented using deep neural networks (DNNs) have been widely used for realizing various functional blocks in wireless transceivers. These learning networks have shown robustness along with complexity advantages [12], [13]. Several early works on DD channel estimation for OTFS have considered integer delays and Dopplers in the channel response. More recent works have considered fractional delays and Dopplers, which are more practical [7]-[10]. A key issue with the estimation of fractional DDs is the relatively high complexity compared to that with the estimation of integer DDs. This letter considers estimation of fractional DDs in OTFS and proposes a novel machine learning-based approach that achieves very good performance at low complexity. A key novelty in the proposed approach is that learning is carried out in the time-frequency (TF) domain for DD domain channel estimation. The rationale behind the proposed learning in the TF domain is explained as follows.

In [8], the authors have considered the problem of fractional DD channel estimation and proposed an algorithm, called modified maximum-likelihood estimation (M-MLE) algorithm, which was shown to perform better than the off-grid sparse Bayesian learning (SBL) based algorithm reported in [7]. Subsequently, the authors in [9] proposed a DD domain inter-path interference cancellation (DDIPIC) algorithm which was shown to perform better than the M-MLE algorithm in [8]. However, the complexity of the DDIPIC algorithm is high. A sizable part of this high complexity is due to the computation of a constituent DD parameter matrix (CDDPM) in the DDIPIC algorithm. .We propose to learn this matrix (CDDPM) rather then explicitly computing it with a motivation to reduce complexity. The way we learn this matrix forms a key novelty in the letter. Specifically, we observed that a direct learning of the CDDPM in the DD domain is not effective, which is attributed to the large swing in the values of the elements of the CDDPM (see Fig. 1a), which is not favorable for training. On the other hand, the TF domain matrix corresponding to the CDDPM has a more even spread of the values of the elements in the TF matrix (see Fig. 1b), which is more favorable for training using a simple network. Symplectic finite Fourier transform (SFFT) is applied on the trained TF matrix to obtain the CDDPM, which is then used in the signal detection. Simulation results demonstrate that the proposed TF learning approach achieves almost the same normalized mean square error (NMSE) and bit error rate (BER) performance as that of the DDIPIC algorithm in [9], at a drastically reduced complexity.

II. OTFS system model

A total of MN information symbols are multiplexed in the DD domain to generate the symbol matrix denoted by $\mathbf{A}_{\rm DD} \in \mathbb{A}^{M\times N}$, where $\mathbb{A}$ signifies the modulation alphabet from which these information symbols are drawn. These symbols are distributed along the delay and Doppler axes with spacings of T/M and $\Delta f/N$, respectively, where Δf equals $1/T$ is the subcarrier spacing, $B = M\Delta f$ is the bandwidth, and M and N are the number of delay and Doppler bins, respectively. Through the use of inverse symplectic finite Fourier transform (ISFFT), the symbols in the DD domain are transformed into the time-frequency (TF) domain to obtain $\mathbf{A}_{\rm TF} \in \mathbb{C}^{M\times N}$. $\mathbf{A}_{\rm TF}$ is converted into a continuous time domain signal $a(t)$ using Heisenberg transform. The transmit signal $a(t)$ passes through a channel comprising of P paths in the DD domain. Each path is associated with a delay τ_p (with $0 < \tau_p < T$) and a Doppler shift ν_p, which are assumed to take fractional values. The received signal $b(t)$ is converted back to DD domain, first by transforming it to TF domain

Manuscript submitted October 19, 2023; revised December 06, 2023 and January 09, 2024; accepted February 13, 2024. This work was supported in part by the J. C. Bose National Fellowship, Department of Science and Technology, Government of India. The associate editor coordinating the review of this paper and approving it for publication was Dr. Tongyang Xu (*Corresponding author: A. Chockalingam*). The authors are with the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore-560012, India. (e-mail: sandeshmattu@iisc.ac.in, achockal@iisc.ac.in).



(a) DD domain (b) TF domain

Fig. 1. Absolute values of training data in DD and TF domains in dB scale.

using Wigner transform to obtain $\mathbf{B}_{\text{TF}} \in \mathbb{C}^{M\times N}$, and then using symplectic finite Fourier transform (SFFT) to obtain $\mathbf{B}_{\text{DD}} \in \mathbb{C}^{M\times N}$ in the DD domain. The transmit and receive pulses are assumed to be rectangular pulses with a duration of T and amplitude $1/\sqrt{T}$. The system input-output relation can be written in the form [8], [9]

$$\mathbf{b}_{\text{DD}} = \sum_{p=1}^{P} \alpha_p \mathbf{E}_p(\tau_p, \nu_p)\mathbf{a}_{\text{DD}} + \mathbf{w}, \tag{1}$$

where $\mathbf{w} \in \mathbb{C}^{MN\times 1}$ is the additive noise samples distributed as i.i.d. $\mathcal{CN}(0,\sigma^2)$ and $\mathbf{G} = \sum_{p=1}^{P} \alpha_p \mathbf{E}_p(\tau_p,\nu_p)$ is the channel matrix. Additionally, $\mathbf{b}_{\text{DD}} \in \mathbb{C}^{MN\times 1}$, $\mathbf{a}_{\text{DD}} \in \mathbb{A}^{MN\times 1}$ are vectorized forms of $\mathbf{B}_{\text{DD}}$ and $\mathbf{A}_{\text{DD}}$, respectively, i.e., $\mathbf{b}_{\text{DD}}[d'] = \mathbf{b}_{\text{DD}}[k'M + l'] = \mathbf{B}_{\text{DD}}[l', k']$, $\mathbf{a}_{\text{DD}}[d] = \mathbf{a}_{\text{DD}}[kM + l] = \mathbf{A}_{\text{DD}}[l, k]$, $l', l = 0, 1, \cdots M-1$, $k', k = 0, 1, \cdots, N-1$, and $d', d = 0, 1, \cdots, MN-1$, and $\mathbf{E}_p$ is an $MN \times MN$ matrix whose entries are given by

$$\mathbf{E}_p[d', d] = e^{-j2\pi\tau_p\nu_p} ee', \tag{2}$$

where $e = \frac{1}{N}\sum_{n=0}^{N-1} e^{-j2\pi n\left(\frac{k'-k}{N} - \frac{\nu_p}{\Delta f}\right)}$, $e' = \frac{1}{M}\sum_{m=0}^{M-1} e^{j2\pi\frac{m}{M}\left(l'-l-M\frac{\tau_p}{T}\right)} f_{\tau_p,\nu_p,k,l'}(m)$, and $f_{\tau_p,\nu_p,k,l'}(m)$ is evaluated using (3) given at the bottom of this page.

III. TF LEARNING BASED DD CHANNEL ESTIMATION

In this section, we present the pilot frame structure, briefly describe the channel estimation algorithm in [9] for reference, and then present the proposed learning approach. As in [8], an exclusive pilot frame, $\mathbf{A}_p \in \mathbb{R}^{M\times N}$, given by

$$\mathbf{A}_p[k,l] = \begin{cases} \sqrt{MNE_p}, & \text{if } k = k_p, l = l_p, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

is transmitted for channel estimation, where $k = k_p, l = l_p$ is the DD resource element (DDRE) in which the pilot symbol is transmitted and E_p is the energy of the transmitted time domain pilot symbol. The received OTFS frame corresponding to the pilot frame is used to estimate the channel at the receiver. The estimation algorithm obtains estimates of the three tuple $(\hat{\tau}_p, \hat{\nu}_p, \hat{\alpha}_p)$, $p = 1, 2, \cdots, P'$, where P' is the number of estimated paths. These estimates are used to construct the estimated channel matrix, $\hat{\mathbf{G}} = \sum_{p=1}^{P'} \hat{\alpha}_p \mathbf{E}_p(\hat{\tau}_p, \hat{\nu}_p) \in \mathbb{C}^{MN\times MN}$ (refer (1)), which is then used for the detection of data frames that follow the pilot frame.

Channel estimation algorithm: Equation (1) can be written in an alternate form as [9]

$$\mathbf{b} = \sum_{p=1}^{P} \mathbf{r}(\tau_p, \nu_p)\alpha_p + \mathbf{w} = \mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})\boldsymbol{\alpha} + \mathbf{w}, \tag{5}$$

where $\mathbf{r}(\tau_p, \nu_p) = \mathbf{E}_p\mathbf{a}_{\text{DD}} \in \mathbb{C}^{MN\times 1}$, $\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu}) = [\mathbf{r}(\tau_1, \nu_1) \ \cdots \ \mathbf{r}(\tau_P, \nu_P)] \in \mathbb{C}^{MN\times P}$, $\boldsymbol{\alpha} = [\alpha_1 \ \cdots \ \alpha_P]^T \in \mathbb{C}^{P\times 1}$, and $\mathbf{w} \sim \mathcal{CN}(0, \sigma^2) \in \mathbb{C}^{MN\times 1}$. We refer to the matrix $\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})$ as the *constituent DD parameter matrix* (CDDPM) because it captures the effects of delay and Doppler of each path in the channel on the transmitted OTFS frame. The maximum-likelihood solution for the three tuple estimation is

$$(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}}) = \underset{\boldsymbol{\alpha},\boldsymbol{\tau},\boldsymbol{\nu}}{\arg\min} \|\mathbf{b} - \mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})\boldsymbol{\alpha}\|^2, \tag{6}$$

which is an estimation problem in three variables. To reduce the estimation complexity, $\boldsymbol{\alpha}$ can be solved given $(\boldsymbol{\tau}, \boldsymbol{\nu})$ as

$$\boldsymbol{\alpha} = \left[\mathbf{R}^H(\boldsymbol{\tau}, \boldsymbol{\nu})\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})\right]^{-1} \mathbf{R}^H(\boldsymbol{\tau}, \boldsymbol{\nu})\mathbf{b}. \tag{7}$$

To estimate $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$, given $\boldsymbol{\alpha}$, (6) can be solved to obtain

$$[\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}}] = \underset{\boldsymbol{\tau},\boldsymbol{\nu}}{\arg\max} \left[\boldsymbol{\Theta}(\mathbf{R})\right], \tag{8}$$

where $\boldsymbol{\Theta}(\mathbf{R}) = \mathbf{b}^H\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})(\mathbf{R}^H(\boldsymbol{\tau}, \boldsymbol{\nu})\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu}))^{-1}\mathbf{R}^H(\boldsymbol{\tau}, \boldsymbol{\nu})\mathbf{b}$. Substituting $\boldsymbol{\tau} = \hat{\boldsymbol{\tau}}$ and $\boldsymbol{\nu} = \hat{\boldsymbol{\nu}}$ in (7), we obtain the estimate of the channel coefficient $\hat{\boldsymbol{\alpha}}$.

The algorithm proceeds in a path-wise fashion, i.e., the delay and Doppler values of pth path $(1 \le p \le P_{\max})$ are estimated before values of $(p+1)$th path values are estimated. Since the knowledge of the number of paths is not assumed to be known, a maximum of $P_{\max}$ paths are estimated. The estimation of τ_p and ν_p for the pth path is carried out in three steps. First, a coarse estimation (integer estimation) is carried out to obtain τ_p', ν_p'. This is followed by an iterative fine estimation step, where the fractional estimation of the delay and Doppler is carried out to obtain $\hat{\tau}_p, \hat{\nu}_p$. Finally, a refinement step refines the estimates obtained till the pth path. In each of the steps, the cost function in (8) is maximized over different search areas as described below. The algorithm begins with initializing $\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu}) = [\mathbf{r}(\tau_1, \nu_1) \ \mathbf{r}(\tau_2, \nu_2) \ \cdots \ \mathbf{r}(\tau_{P_{\max}}, \nu_{P_{\max}})] = \mathbf{0}_{MN\times P_{\max}}$.

Coarse estimation: The search area in this step is defined as $\mathcal{G} = \left\{\frac{0}{M\Delta f}, \frac{1}{M\Delta f}, \cdots, \frac{L}{M\Delta f}\right\} \times \left\{-\frac{K}{NT}, \cdots, \frac{0}{NT}, \cdots, \frac{K}{NT}\right\}$, where $L = \lceil \tau_{\max} M\Delta f \rceil$, $K = \lceil \nu_{\max} NT \rceil$, $\tau_{\max}$ and $\nu_{\max}$ denote the maximum delay and Doppler, respectively, and $\times$ denotes Cartesian product of two sets. For estimating the parameters for the pth path, $\mathbf{r}(\tau_p, \nu_p)$ is computed for all (τ_p, ν_p) in $\mathcal{G}$ and the coarse estimates are obtained from (8) by maximizing the cost function over the search area.

Iterative fine estimation: Following the coarse estimation step,

$$f_{\tau_p,\nu_p,k,l'}(m) = \sum_{s=-m}^{M-1-m} e^{j2\pi\frac{sl'}{M}} \left[\left(1-\frac{\tau_p}{T}\right) e^{j\pi\left(1+\frac{\tau_p}{T}\right)\left(\frac{\nu_p}{\Delta f}-s\right)} \text{sinc}\left(\left(1-\frac{\tau_p}{T}\right)\left(\frac{\nu_p}{\Delta f}-s\right)\right) + e^{-j2\pi\frac{k}{N}} \left(\frac{\tau_p}{T}\right) e^{\frac{j\pi\tau_p}{T}\left(\frac{\nu_p}{\Delta f}-s\right)} \text{sinc}\left(\left(\frac{\tau_p}{T}\right)\left(\frac{\nu_p}{\Delta f}-s\right)\right)\right]. \tag{3}$$

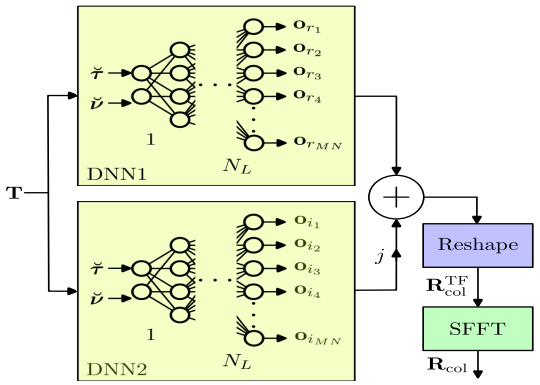


Fig. 2. Proposed TF learning network architecture for learning $\mathbf{R}(\boldsymbol{\tau}, \boldsymbol{\nu})$.

the search area is now defined around the optimal coarse value (for $s = 1$) or the fine estimate obtained in the previous iteration of the fine estimation step (for $s > 1$) given by $\mathcal{F}^{(s)} = \left\{w_\tau^{(s)}\Gamma + \hat{\tau}^{(s)}\right\} \otimes \left\{w_\nu^{(s)}\Lambda + \hat{\nu}^{(s)}\right\}$, where s is the iteration number in the fine estimation step, $w_\tau^{(s)} = \frac{1}{M\Delta f m_\tau^{s-1}}$ and $w_\nu^{(s)} = \frac{1}{NTn_\nu^{s-1}}$ are the resolution along Doppler and delay, respectively, $\Gamma = \left\{0, \cdots, \lfloor \frac{m_\tau}{2} \rfloor\right\}$ for $\hat{\tau}_s = 0$ and $\Gamma = \left\{-\lfloor \frac{m_\tau}{2} \rfloor, \cdots, 0, \cdots, \lfloor \frac{m_\tau}{2} \rfloor\right\}$ for $\hat{\tau}_s > 0$, and $\Lambda = \left\{-\lfloor \frac{n_\nu}{2} \rfloor, \cdots, 0, \cdots, \lfloor \frac{n_\nu}{2} \rfloor\right\}$. For $s = 1$, $\hat{\tau}^{(1)}$ and $\hat{\nu}^{(1)}$ are initialized to the coarse estimates obtained earlier. A similar procedure as in coarse estimation step is followed using $\mathcal{F}^{(s)}$ as the search area for obtaining the first fine estimate $(\hat{\tau}^{(2)}, \hat{\nu}^{(2)})$, following which s in incremented by 1. For $s > 1$, the search area is centered over the newly obtained fine estimate with finer resolution. This is carried out until a maximum number of iterations $(s_{\max})$ is reached or the stopping criterion given by $\left(|\hat{\tau}^{(s+1)} - \hat{\tau}^{(s)}| < \epsilon_\tau \text{ and } |\hat{\nu}^{(s+1)} - \hat{\nu}^{(s)}| < \epsilon_\nu\right)$ is met.

Refinement step: To further improve the accuracy of the estimates, the refinement of the estimated parameters are carried out. After the tth path estimation, with $1 < t < P_{\max}$, before estimation of the $(t+1)$th path, the refinement of all the paths till t are carried out. For refining the zth path $(1 < z < t)$, all the t columns in $\mathbf{R}$ are filled except the zth column. Next, coarse and iterative fine estimation are carried for the zth path. After refinement of all the paths, the algorithm proceeds to estimate the parameters of the $(t+1)$th path.

Stopping criterion: After estimating t paths, the matrix $\mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})$ is obtained and $\hat{\boldsymbol{\alpha}}$ is obtained from (7). If $\|\mathcal{E}^{(t)} - \mathcal{E}^{(t-1)}\|^2 < \epsilon$, where $\mathcal{E}^{(t)} = \mathbf{b} - \mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\hat{\boldsymbol{\alpha}}$ then the algorithm stops. If the criterion is not met until $t = P_{\max}$, then the algorithm is terminated at $t = P_{\max}$.

A. Proposed TF learning based approach using DNN

In the above channel estimation algorithm, all the coarse estimation, fine estimation, and refinement steps require multiple computations of the cost function in (8) which requires generation of the CDDPM, $\mathbf{R}$. Computing the columns of $\mathbf{R}$, $\mathbf{r}(\tau_i, \nu_i)$, for each path involves high complexity. Therefore, to reduce the complexity and for practicality, we propose to devise and train a network for learning the columns of $\mathbf{R}$. Specifically, we note that the CDDPM is a function of delay and Doppler (i.e., each column of CDDPM has a one-to-one relation to (τ, ν)-tuple, as $\mathbf{r}(\tau, \nu) = \mathbf{E}(\tau, \nu)\mathbf{a}_{\text{DD}}\big)$, and we use DNNs to learn this one-to-one relation. It turns out that the proposed learning/training architecture is able to effectively learn this relation accurately, offering complexity benefit in the process. The proposed DNN architecture and training methodology are presented below.

TABLE I
PARAMETERS OF DNN1/DNN2

Parameter	Value
Architecture	Fully connected neural network
Input dimension	2
Output dimension	$MN = 2048$
Number of layers (N_L)	10
Activation function	ReLU for all layers except last layer and linear activation function for last layer
κ	1000

1) Architecture: Figure 2 shows the architecture of the proposed approach. The input to the DNN is the matrix $\mathbf{T} = [\check{\boldsymbol{\tau}}\ \check{\boldsymbol{\nu}}] \in \mathbb{R}^{S\times 2}$, where S is the cardinality of $\mathcal{G}$ (for coarse estimation) or $\mathcal{F}^{(s)}$ (for the sth iteration of the fine estimation) and $\check{\boldsymbol{\tau}} = \kappa\boldsymbol{\tau}/\tau_{\max}$, $\check{\boldsymbol{\nu}} = \kappa\boldsymbol{\nu}/\nu_{\max}$. The division by $\tau_{\max}$ $(\nu_{\max})$ is carried out to normalize the values of delay (Doppler) between 0 and 1 (-1 and 1)[1]. Further, the multiplication by κ is done to magnify small changes in delay and Doppler values in the training data. The vectors $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ are obtained from the search area $\mathcal{G}$ or $\mathcal{F}^{(s)}$. This matrix is passed through two networks, DNN1 and DNN2. DNN1 outputs the real part, $[\mathbf{o}_{r_1}\ \cdots\ \mathbf{o}_{r_{MN}}] \in \mathbb{R}^{S\times MN}$, while DNN2 outputs the imaginary part, $[\mathbf{o}_{i_1}\ \cdots\ \mathbf{o}_{i_{MN}}] \in \mathbb{R}^{S\times MN}$. The real and imaginary parts are combined and reshaped to obtain $\mathbf{R}_{\text{col}}^{\text{TF}} \in \mathbb{R}^{S\times M\times N}$. Each $M \times N$ matrix in $\mathbf{R}_{\text{col}}^{\text{TF}}$ is then converted to DD domain from TF domain[2] using SFFT and vectorized to obtain an MN-length vector. These vectors form the rows of $\mathbf{R}_{\text{col}} \in \mathbb{C}^{S\times MN}$. The DNN1 and DNN2 are trained so as to provide $\mathbf{r}(\boldsymbol{\tau}(t), \boldsymbol{\nu}(t)) \in \mathbb{C}^{1\times MN}$ as the tth row of $\mathbf{R}_{\text{col}}$ for tth row in the input, $\mathbf{T}(t) = [\boldsymbol{\tau}(t)\ \boldsymbol{\nu}(t)]$.

The architectures of DNN1 and DNN2 are identical and are comprised of fully connected layers as shown in Fig. 2. The output dimension is twice the input dimension, i.e., the ith layer $(i = 1, 2, \cdots)$ of DNN1 (DNN2) has input and output dimensions 2^i and $2^{(i+1)}$, respectively. The number of layers, N_L, in DNN1 (DNN2) is determined by the choices of M and N, such that the last layer has input dimension 2^{N_L} and output dimension $\min(2^{N_L+1}, MN)$, with $2^{N_L} < MN$ and $2^{N_L+1} \geq MN$. For each fully connected layer except the last layer, a rectified linear unit (ReLU) activation function is used, and a linear activation function is used for the last layer to allow the output of DNN1 and DNN2 to span $\mathbb{R}$. The parameters of DNN1 (DNN2) are listed in Table I.

2) Training methodology: Training data is obtained by generating (τ, ν) tuples and corresponding $\mathbf{r}(\tau, \nu)$ vectors using $\mathbf{r}(\tau, \nu) = \mathbf{E}\mathbf{a}_{\text{DD}}$ (see (5)). The vectors $\mathbf{r}^T(\tau, \nu) \in \mathbb{C}^{1\times MN}$ are

[1]The normalization to values between 0 and 1, and -1 and 1 for delay and Doppler, respectively, are done so that the ranges of both the inputs are similar, which aids training. Without this normalization, delay would be in the order of 10^{-6}, while Doppler would be in 10^3.

[2]We note that the training is carried out in the TF domain instead of DD domain. This is done because the ratio of the absolute values of the highest value to the lowest value in the vector $\mathbf{r}(\tau, \nu)$ in the DD domain is huge which is detrimental to training. In the TF domain, however, this ratio is quite reasonable and therefore favorable for training (see Fig. 1).

TABLE II
HYPER-PARAMETERS USED WHILE TRAINING

Hyper-parameter	Value
Batch size	40000
Mini batch size	8000
Number of epochs	40000
Learning rate	0.001, divide by 2 every 4000 epochs
Number of training samples	325000

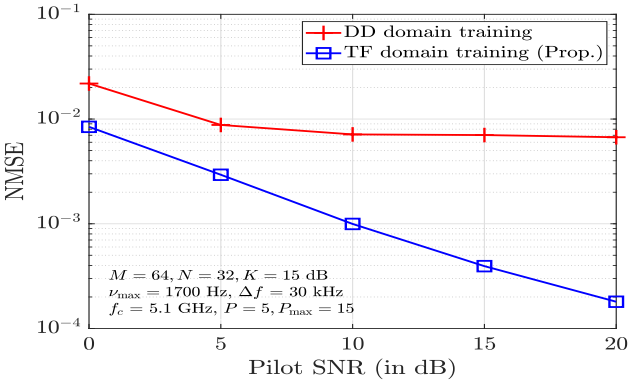


Fig. 3. NMSE performance comparison between training carried out in DD domain and TF domain.

reshaped into matrices of size $M \times N$ and converted to TF domain using ISFFT, following which they are converted back to vectors to obtain $\mathbf{r}^{\mathrm{TF}}(\tau,\nu) \in \mathbb{C}^{1\times MN}$. To train the network, the tuples are fed as input to the DNN1 and DNN2 to generate the output. Training is carried out using an Adam optimizer to minimize the L1 loss function evaluated between the output of the DNN1 (DNN2) and $\Re\{\mathbf{r}^{\mathrm{TF}}(\tau,\nu)\}$ ($\Im\{\mathbf{r}^{\mathrm{TF}}(\tau,\nu)\}$) values, where $\Re\{\cdot\}$ and $\Im\{\cdot\}$, represent the real part and imaginary part, respectively. The other hyper parameters used while training are presented in Table II. We note that this training has to be carried out offline, only once. Subsequently, the network weights are frozen. During test phase, DNN1 and DNN2 with the trained weights are used for both coarse and fine estimation steps in the estimation algorithm.

IV. Results and Discussions

In this section, we present the results obtained using the proposed low-complexity TF learning based channel estimation algorithm. The following parameters are considered. $M = 64$, $N = 32$, $\Delta f = 30$ kHz, and $f_c = 5.1$ GHz. Two power delay profiles (PDPs) are considered. First, as in [8], the channel is considered to have $P = 5$ paths with a line of sight (LOS) path having a Rice factor of 15 dB. The first and second paths have fixed delays given by 0.667μs, 0.867μs, respectively, and for the other paths, the delays are uniformly distributed in $(0.867\mu\text{s}, 7\mu\text{s}]$. For all the paths, the Dopplers are generated using the Jake's Doppler spectrum, i.e., $\nu_p = \nu_{\max}\cos(\theta_p)$, with $\theta_p \sim \mathrm{U}(-\pi,\pi]$, where $\mathrm{U}\{\cdot\}$ denotes uniform distribution. As in [8], the LOS path gain is determined by a fixed absolute squared value, according to the Rice factor. For the remaining paths, an exponential PDP is considered. Second, the Vehicular A (VehA) channel model with $P = 6$ paths defined in [14] is considered. Further, $m_\tau = n_\nu = 10$, $\epsilon_\tau = 10^{-10}$, $\epsilon_\nu = 10^{-2}$, $P_{\max} = 15$, $s_{\max} = 10$, and $\epsilon = 0.01MN\sigma^2$.

A. Training in DD domain vs TF domain

The advantage of training the network in TF domain is demonstrated in the NMSE plots shown in Fig. 3. The DNN1 and DNN2 with the same parameters and hyper-parameters are trained using data i) in DD domain and ii) in TF domain. In the DD domain, the swing in the values in CDDPM is quite high (15 dB to -25 dB, as seen in Fig. 1a). In DD domain training, DNN1/DNN2 are trained to output the learnt CDDPM directly in the DD domain (without the use of Reshape and SFFT blocks in Fig. 2). Due to the presence of high and low values in DD domain training data (i.e., CDDPM), the weights/biases in the network need to be high (to be able to output high values) and also low (to be able to output low values). Learning to output both very high and very low values leads to poor weight/bias updates, leading to poorly learnt CDDPM. This remains so even at high pilot SNRs, because the relative swing in the values remain the same. This issue is alleviated in the TF domain training, thanks to the much smaller swing in the TF training data values (0 to 5 dB, as seen in Fig. 1b). This is reflected in Fig. 3, where the NMSE performance with DD domain training is found to floor, whereas that with TF domain training does not.

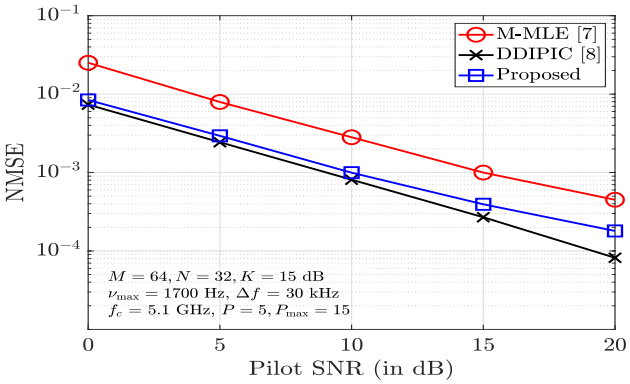


Fig. 4. NMSE performance comparison between the proposed approach and DDIPIC [9] and M-MLE [8].

B. NMSE as a function of pilot SNR

Figure 4 shows the NMSE performance comparison between the proposed approach, the DDIPIC algorithm in [9], and the M-MLE algorithm in [8]. It is observed that the NMSE performance of the proposed approach is better than the M-MLE scheme. For example, an NMSE of 10^{-3} is attained at a pilot SNR of 15 dB for M-MLE, whereas it is achieved at about 10 dB in the proposed approach. Also, the performance of the proposed approach is similar to that of DDIPIC.

C. BER as a function of SNR

For computing the BER, we assume that the channel remains constant for two OTFS frame duration. The first frame is the pilot frame given in (4), which is used for channel estimation, and the second frame is an OTFS frame with data symbols drawn from a constellation, detected using the estimated channel. Figures 5 and 6 show the BER performance of the proposed approach as a function of SNR using 64-QAM and minimum mean square error (MMSE) detection for $P = 5$ with pilot SNR 10 dB and $P = 6$ with pilot SNR 15 dB, respectively. Additionally, the performances of DDIPIC and M-MLE algorithms are also presented. Perfect channel state information (CSI) performance is also added as a benchmark.

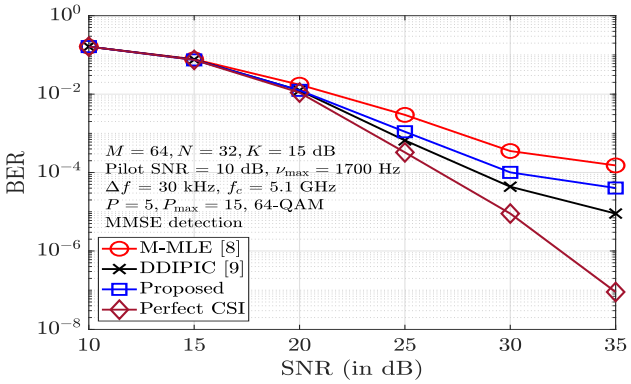


Fig. 5. BER performance comparison between the proposed approach, DDIPIC [9], M-MLE [8], and perfect CSI.

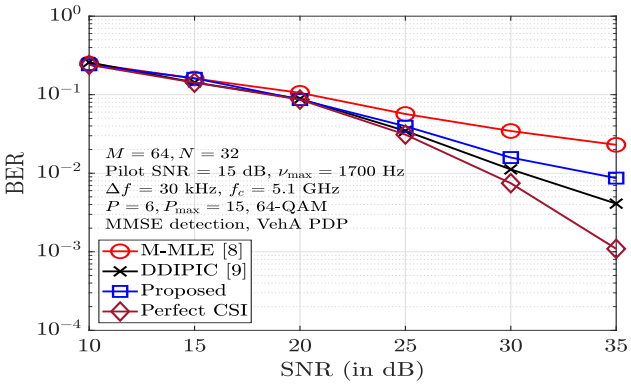


Fig. 6. BER performance comparison between the proposed approach, DDIPIC [9], M-MLE [8], and perfect CSI for VehA channel model.

It is observed that for both the considered channel models, the performance of M-MLE scheme is inferior to both the proposed approach as well as the DDIPIC scheme, owing to the inferior NMSE performance. Also, the proposed approach performs very close to the DDIPIC algorithm[3], which is in turn close to the perfect CSI performance. For example, in Fig. 5, for a BER of 10^{-3}, the proposed approach has an advantage of about 2.5 dB, while the DDIPIC algorithm has an advantage of less than a dB when compared with the proposed algorithm.

D. Complexity comparison

In this subsection, we compare the complexity of generating $\mathbf{r}(\tau,\nu)$, which are columns of the $\mathbf{R}$ matrix, using the following approaches. The first method is the brute force computation using the equivalence $\mathbf{r}(\tau,\nu) = \mathbf{E}\mathbf{a}_{\rm DD}$. In the second method, we compute this equivalence in TF domain to simplify calculations, i.e., $\mathbf{a}_{\rm DD}$ is converted to $\mathbf{a}^{\rm TF}$ using ISFFT and $\mathbf{r}^{\rm TF}$ is obtained using an analytical input-output relation which requires much less computational complexity, followed by $\mathbf{r}^{\rm TF}$ to $\mathbf{r}(\tau,\nu)$ conversion using SFFT. The third method is from [10], which gives a low-complexity method for generation of the matrix $\mathbf{G}$. Here, we computed the (k_pM+l_p)th column of $\mathbf{G}$ using the method in [10] by substituting $P = \alpha = 1$, which gives $\mathbf{r}(\tau,\nu)$. The fourth method is the proposed TF learning method, where the $\mathbf{R}_{\rm col}^{\rm TF}$ learnt in TF domain is converted to DD domain through SFFT to obtain $\mathbf{r}(\tau,\nu)$. We obtained the run time complexity of computing $\mathbf{r}(\tau,\nu)$ of all the methods on the same machine for fair comparison (see Table III). It is seen that the naive brute force way of computing $\mathbf{r}(\tau,\nu)$ requires 60 s of run time, while the second method requires just about 1 s. The third method in [10] requires 10 s, which is not as good as the second method. The run time of the proposed method is the best amongst all (better by a factor of at least 10), which is practically very attractive. While the other three methods give exact values of $\mathbf{r}(\tau,\nu)$ (due to their computation using analytical expressions in the system model), the proposed method gives $\mathbf{r}(\tau,\nu)$ through learning. Yet, the performance achieved by the proposed learning is quite close to those obtained using exact analytical computations. This shows that a judicious adoption of learning approach for the problem at hand can yield efficient solutions. The proposed learning approach for embedded pilot frames [4] and general pulse shapes can be carried out as future work.

[3]The performance of the proposed approach is slightly inferior compared to that of DDIPIC in Figs. 4, 5, 6. This is because the DDIPIC uses exact computation of the CDDPM, whereas the proposed approach learns the CDDPM, which is a close approximation of the exact CDDPM. This difference between the exact and learnt CDDPM translates to a slight performance degradation.

TABLE III
RUN TIME COMPLEXITIES OF COMPUTING $\mathbf{r}(\tau,\nu)$

Method	Description	Average time (in sec)
1	Brute force	60
2	TF domain processing	1
3	Method in [10]	10
4	TF learning (Prop.)	0.1

REFERENCES

[1] R. Hadani et al., "Orthogonal time frequency space modulation," *Proc. IEEE WCNC'2017*, pp. 1-6, Mar. 2017.
[2] Z. Wei et al., "Orthogonal time-frequency space modulation: a promising next-generation waveform," *IEEE Wireless Commun. Mag.*, vol. 28, no. 4, pp. 136-144, Aug. 2021.
[3] Z. Gong, S. Liu, and Y. Huang, "Doppler diversity reception for OTFS modulation," *Proc. IEEE VTC'2022-Spring*, pp. 1-5, Jun. 2022.
[4] P. Raviteja, K. T. Phan, Y. Hong, and E. Viterbo, "Embedded pilot-aided channel estimation for OTFS in delay-Doppler channels," *IEEE Trans. Veh. Tech.*, vol. 68, no. 5, pp. 4906-4917, May 2019.
[5] L. Zhao, W. J. Gao, and W. Guo, "Sparse Bayesian learning of delay-Doppler channel for OTFS system," *IEEE Commun. Lett.*, vol. 24, no. 12, pp. 2766-2769, Dec. 2020.
[6] D. Shi, W. Wang, L. You, X. Song, Y. Hong, X. Gao, and G. Fettweis, "Deterministic pilot design and channel estimation for downlink massive MIMO-OTFS systems in presence of the fractional Doppler," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7151-7165, Nov. 2021.
[7] Z. Wei, W. Yuan, S. Li, J. Yuan, and D. W. K. Ng, "Off-grid channel estimation with sparse Bayesian learning for OTFS systems," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7407-7426, Sep. 2022.
[8] I. A. Khan and S. K. Mohammed, "Low complexity channel estimation for OTFS modulation with fractional delay and Doppler," arXiv:2111.06009 [cs.IT], Nov. 2021.
[9] S. P. Muppaneni, S. R. Mattu, and A. Chockalingam, "Channel and radar parameter estimation with fractional delay-Doppler using OTFS," *IEEE Commun. Letters*, vol. 27, no. 5, pp. 1392-1396, May 2023.
[10] Z. Wang, L. Liu, Y. Yi, R. Calderbank, and J. Zhang, "Low-complexity channel matrix calculation for OTFS systems with fractional delay and Doppler," *Proc. IEEE MILCOM'2022*, pp. 787-792, Nov. 2022.
[11] S. R. Mattu and A. Chockalingam, "Learning based delay-Doppler channel estimation with interleaved pilots in OTFS," *Proc. IEEE VTC'2022-Fall*, pp. 1-6, Sep. 2022.
[12] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: an emerging interdisciplinary paradigm" *IEEE Wireless Commun.*, vol. 27, no. 4, pp. 133-139, Aug. 2020.
[13] Y. I. Tek, A. T. Dogukan, and E. Basar, "Autoencoder-based enhanced orthogonal time frequency space modulation," *IEEE Commun. Lett.*, vol. 27, no. 10, pp. 2628-2632, Oct. 2023.
[14] ITU-R M.1225, "Guidelines for the evaluation of radio transmission technologies for IMT-2000," International Telecommunication Union Radio communication, 1997.