CHALMERS
UNIVERSITY OF TECHNOLOGY

Reduced-latency DL-based Fractional Channel Estimation in OTFS Receivers

Citation for the original published paper (version of record):
Marchese, M., Wymeersch, H., Spallaccini, P. et al (2025). Reduced-latency DL-based Fractional Channel Estimation in OTFS Receivers. 2025 IEEE International Conference on Machine Learning for Communication and Networking Icmlcn 2025.
http://dx.doi.org/10.1109/ICMLCN64995.2025.11140575

Reduced-latency DL-based Fractional Channel Estimation in OTFS Receivers

Mauro Marchese*, Henk Wymeersch†, Paolo Spallaccini‡, Stefano Chinnici‡, Pietro Savazzi*§
*University of Pavia, Italy, ‡HCL Software, Vimodrone, Milan, Italy, †Chalmers University of Technology, Sweden,
§CNIT Consorzio Nazionale Interuniversitario per le Telecomunicazioni, Pavia, Italy
E-mail: mauro.marchese01@universitadipavia.it

***Abstract*—In this work, we propose a deep learning (DL)-based approach that integrates a state-of-the-art algorithm with a time-frequency (TF) learning framework to minimize overall latency. Meeting the stringent latency requirements of 6G orthogonal time-frequency space (OTFS) systems necessitates low-latency designs. The performance of the proposed approach is evaluated under challenging conditions: low delay and Doppler resolutions caused by limited time and frequency resources, and significant interpath interference (IPI) due to poor separability of propagation paths in the delay-Doppler (DD) domain. Simulation results demonstrate that the proposed method achieves high estimation accuracy while reducing latency by approximately 55% during the maximization process. However, a performance trade-off is observed, with a maximum loss of 3 dB at high pilot SNR values.**

***Index Terms*—Deep learning (DL), interpath interference (IPI), channel estimation, fractional channel parameters.**

I. Introduction

Channel estimation in orthogonal time-frequency space (OTFS) [1]–[3] systems plays a crucial role in future 6G networks, as data detection heavily relies on accurate channel state information (CSI) at the receiver. In OTFS systems, the channel parameters often assume fractional delay and Doppler values when delay and Doppler resolutions are insufficient. This leads to the spreading of received pilot replicas into adjacent bins, causing interpath interference (IPI) [4]. To address this, multigrid algorithms that account for fractional delays and Doppler shifts must be employed to improve estimation accuracy. Additionally, refinement steps are needed to enhance detection capabilities and eliminate false paths introduced by IPI [5]. Thus, an increase in complexity is unavoidable to guarantee high estimation quality. Therefore, novel solutions should be developed to reduce latency due to receiver processing when such computationally demanding algorithms must be used.

To mitigate the impact of IPI, the progressive interpath interference cancellation (P-IPIC) algorithm [5] was proposed as a low-latency alternative to the state-of-the-art delay-Doppler interpath interference cancellation (DDIPIC) algorithm [4]. P-IPIC achieves better performance than DDIPIC, with gains of approximately $1-2$ dB in high-IPI scenarios while significantly reducing latency. The latency reduction stems from (i) a single global refinement step in P-IPIC, compared to multiple refinements required by DDIPIC which cancels IPI through these refinement procedures; (ii) less complexity needed for cost function evaluation since the P-IPIC cancels IPI by computing a residue vector and estimates channel parameters by maximizing a residue-based cost fucntion [5]. While other low-complexity methods, such as the modified maximum likelihood estimator (M-MLE) [6], have been explored, DDIPIC is shown to outperform these methods at the expense of higher computational complexity [4].

Deep learning has emerged as a transformative tool in wireless communications [7], providing solutions to challenges in 6G technologies by enhancing performance and reducing complexity. For example, [8] demonstrates that a deep neural network (DNN) can directly estimate channel parameters from received frames, outperforming conventional threshold-based methods [9]. Similarly, [10] highlights the suitability of deep learning for feature extraction in the delay-Doppler domain. In [11], a DNN-based approach is proposed to reduce the latency of the DDIPIC algorithm [4]. This approach approximates the relationship between delay-Doppler pairs and corresponding columns of a constituent delay-Doppler parameter matrix (CDDPM), with training in the time-frequency domain achieving superior performance due to the narrower value range of unvectorized CDDPM columns.

In this work, the goal is to develop a reduced-latency algorithm for channel estimation in high-IPI regime where higher complexity is needed to counteract the effects if IPI. The P-IPIC algorithm from [5] is considered as the basline method as it is a state-of-the art algorithm for channel estimaiton in high IPI regimes. To achieve this latency reduction, the computation of the cost function, which is inevitably evaluated multiple times during the estimation procedure, should be simplified. To do so, the deep learning approach from [11] is adapted and combined with the recently proposed P-IPIC algorithm in [5] that leverages a global refinement to reduce IPI. Therefore, two neural networks are designed to predict the time-frequency CDDPM required for residue-based cost function computation. Simulation results demonstrate a significant reduction in latency, approximately halving the time required for cost function maximization, while maintaining high estimation accuracy. The proposed DL-based P-IPIC is therefore shown to be a good candidate as low-latency channel estimation algorithm for high-IPI scenarios.

Notation: $\mathbf{X}$ is a matrix, $\mathbf{x}$ is a vector, and x is a scalar. $X[m,n]$ represents the (m,n)-th element of the matrix $\mathbf{X}$, and $\hat{x}$ denotes the estimate of x. $\|\mathbf{X}\|_F$, $\mathbf{X}^T$, $\mathbf{X}^H$, and $\mathbf{X}^{-1}$

represent the Frobenius norm, transpose, Hermitian (conjugate transpose), and inverse of $\mathbf{X}$, respectively. $\mathbf{I}$ is the identity matrix, $\|\mathbf{x}\|$ is the norm of the vector $\mathbf{x}$, and $|x|$ denotes the absolute value of scalar x. The operators $\text{vec}(\mathbf{X})$ and $\text{vec}_{M,N}^{-1}(\mathbf{x})$ represent the vectorization of matrix $\mathbf{X}$ and the reshaping of vector $\mathbf{x}$ back into a $M \times N$ matrix, respectively. $\mathbb{E}\{\cdot\}$ denotes the expected value and $\mathcal{CN}(0, \sigma^2\mathbf{I})$ represents a complex Gaussian random variable with zero mean and covariance matrix $\sigma^2\mathbf{I}$. Finally, $\mathbb{C}$ denotes the set of complex numbers.

II. OTFS Model and Channel Estimation

A. System Model

In this section, both pilot and observation models for fractional channel estimation are presented in OTFS systems with rectangular pulse shaping. In the following, an OTFS system with M subcarriers with spacing Δf and N time slots of duration T is considered.

1) Pilot model: The single-antenna transmitter sends a pilot symbol in the delay-Doppler (DD) domain. The DD pilot frame is given by

$$X[m,n] = \begin{cases} \sqrt{E_p} & m = m_p,\ n = n_p, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where (m_p, n_p) is the delay-Doppler cell in which the pilot is sent and E_p is the energy of the transmitted pilot signal.

2) Observation model: the received vectorized pilot frame is given by

$$\mathbf{y} = \mathbf{H}_{dd}\text{vec}(\mathbf{X}) + \mathbf{n}, \tag{2}$$

where $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2\mathbf{I}) \in \mathbb{C}^{MN\times 1}$ is the AWGN noise vector and $\mathbf{H}_{dd} \in \mathbb{C}^{MN\times MN}$ is the delay-Doppler domain channel matrix given by

$$\mathbf{H}_{dd} = \sum_{i=1}^{P} \alpha_i \boldsymbol{\Upsilon}_i(\tau_i, \nu_i), \tag{3}$$

where P is the number of propagation paths, α_i is the complex gain of the i-th path and τ_i and ν_i are the propagation delay and the Doppler shift associated with the i-th path, respectively. Moreover, $\boldsymbol{\Upsilon}_i(\tau_i, \nu_i) \in \mathbb{C}^{MN\times MN}$ is a matrix that captures the effect of the propagation delay and the Doppler shift of the i-th path and for $l', l'' = 0, 1, .., M-1$, $k', k'' = 0, 1, ..., N-1$ is computed according to equations reported at the bottom of the page (see also [4]–[6], [11], [12]). Channel estimation is performed considering the input-output relation rewritten as

$$\mathbf{y} = \mathbf{R}\big(\boldsymbol{\tau}, \boldsymbol{\nu}\big)\boldsymbol{\alpha} + \mathbf{n}, \tag{4}$$

where $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2\ ...\ \alpha_P]^T \in \mathbb{C}^{P\times 1}$ is the channel gains vector and $\mathbf{R}\big(\boldsymbol{\tau}, \boldsymbol{\nu}\big) = [\mathbf{r}_1\ \mathbf{r}_2\ ...\ \mathbf{r}_P] \in \mathbb{C}^{MN\times P}$ is the CDDPM matrix. Each column of $\mathbf{R}\big(\boldsymbol{\tau}, \boldsymbol{\nu}\big)$ is related to a different path and is given by

$$\mathbf{r}_i(\tau_i, \nu_i) = \boldsymbol{\Upsilon}_i(\tau_i, \nu_i)\text{vec}(\mathbf{X}). \tag{5}$$

B. The P-IPIC Algorithm

In this section the P-IPIC algorithm proposed in [5] is summarized. This channel estimation algorithm operates in two distinct phases.

1) Search phase: The algorithm iteratively searches for possible paths maximizing a residue-based cost function given by

$$\Phi_e\big(\mathbf{r}(\tau,\nu)\big) = \frac{|\mathbf{r}^H(\tau,\nu)\mathbf{e}^{(i-1)}|^2}{\|\mathbf{r}(\tau,\nu)\|^2}, \tag{6}$$

where the residue vector at the i-th iteration is given as

$$\mathbf{e}^{(i)} = \mathbf{y} - \mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\hat{\boldsymbol{\alpha}}. \tag{7}$$

During this phase, the propagation delay and Doppler shift of the i-th path are estimated by maximizing $\Phi_e\big(\mathbf{r}(\tau,\nu)\big)$ over a search area comprising delays and Doppler shifts that are multiples of the DD resolutions. It can be observed that the delay resolution is $\Delta\tau = \frac{1}{M\Delta f}$, and the Doppler resolution is $\Delta\nu = \frac{1}{NT}$. In this way, a coarse estimate is obtained.

Afterward, an iterative fine estimation procedure is performed to refine the estimated channel parameters. This is achieved by maximizing the residue-based cost function over a search area comprising fractional DD parameters centered at the estimate obtained in the previous step. The search area is narrowed down during each iteration, reducing the DD spacings until the desired accuracy is achieved or a maximum number of iterations is reached. Further details can be found in Section III-C and in [5]. Once the DD pair associated with the i-th path is estimated, the channel gains vector is updated using the regularized least squares (RLS) formula

$$\hat{\boldsymbol{\alpha}} = \Big(\mathbf{R}^H(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}}) + \lambda\mathbf{I}\Big)^{-1}\mathbf{R}^H(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\mathbf{y}, \tag{8}$$

where λ is a sufficiently small number. After that, once the residue vector is obtained, if $\|\mathbf{e}^{(i)}\| < \epsilon$, where ϵ is the convergence tolerance parameter, the *search phase* ends and a number of paths equal to $\hat{P} = i$ is detected. Otherwise, a new iteration is run to search for a new path. The threshold for the stopping criterion is chosen, assuming a Gaussian stationary noise source, as $\epsilon = 3\sqrt{MN\sigma^2}$.

2) Refinement phase: During this second phase, the algorithm iteratively refines the estimated paths by performing a coarse and fine estimate, during which the DD pair associated

$$\Upsilon_i[k'M + l', k''M + l''] = \frac{e^{-j2\pi\tau_i\nu_i}}{MN}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} f_{k'',l'}(m)e^{j2\pi\left(\frac{m}{M}\left(l'-l''-\frac{M\tau_i}{T}\right)-\frac{n}{N}\left(k'-k''-\frac{N\nu_i}{\Delta f}\right)\right)},$$

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

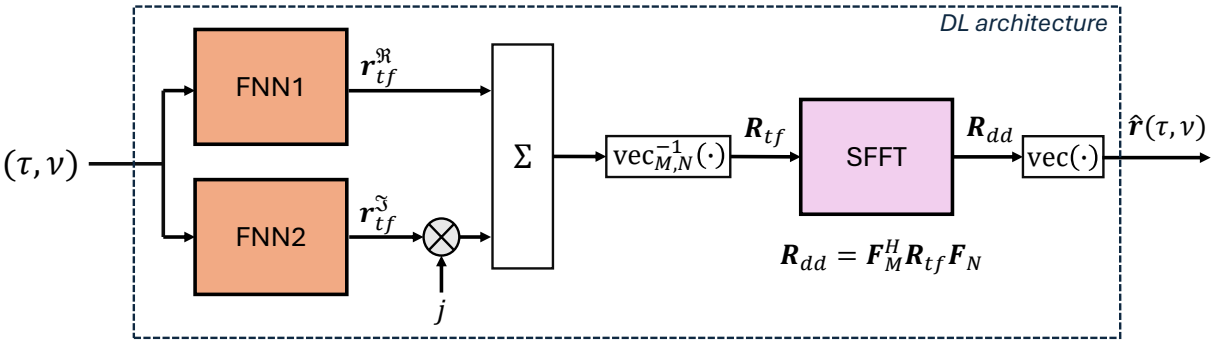

Fig. 1. The architecture of the TF learning approach: a delay-Doppler pair is mapped to the corresponding real and imaginary parts of the TF CDDPM column. After that proper reshaping and conversion to the DD domain is applied to obtain the desired CDDPM column.

with the i-th path is refined by maximizing the regularized observation-based cost function

$$\Phi_y(\mathbf{R}) = \mathbf{y}^H \mathbf{R}\left(\mathbf{R}^H \mathbf{R} + \lambda \mathbf{I}\right)^{-1} \mathbf{R}^H \mathbf{y}, \tag{9}$$

where the dependence on the DD vectors is omitted for simplicity. This is done since $\Phi_y(\mathbf{R})$ takes into account all the estimated paths. During the refinement phase, the residue vector is computed in each iteration, and the stopping criterion is checked. If the stopping criterion is met before all the paths are refined, the remaining unrefined paths are declared false alarms and discarded. Thus, during this phase, a number of paths equal to $\hat{\hat{P}} \leq \hat{P}$ is detected.

III. DL-based Approach

During the cost function maximization, the computational complexity of the P-IPIC algorithm is dominated by the computation of the CDDPM column required to evaluate the residue-based cost function in (5). A reduced-latency approach is to replace the direct computation of (5) with a DL architecture that learns and predicts the desired CDDPM column [11]. In this case, a performance loss is unavoidable due to the limited capabilities of a neural network to learn the mathematical relation between a DD pair and the corresponding CDDPM column. Moreover, the DL architecture outputs the TF version of the desired CDDPM column to facilitate training [11].

A. The Architecture

Fig. 1 shows the adopted DL architecture. It consists of two distinct fully connected feedforward neural networks (FNNs) that learn the real and imaginary parts of the TF CDDPM column, respectively. In particular, the FNN is characterized by:

- *Two inputs*: the network receives as input a normalized DD pair. Normalization is performed to ensure that the delay values are in the range $[0, 1]$ and the Doppler shift values are in the range $[-1, 1]$.
- *Concatenation layer*: this layer is used to concatenate the two inputs.
- *Fully connected layers*: a certain number of fully connected layers are used. The activation function is ReLU, except for the output layer, which adopts a linear activation function.
- *Output layer*: it provides the final output of the network. Its dimension is MN, as the TF CDDPM column must be provided as output.

As depicted in Fig. 1, FNN1 learns the real part of the TF CDDPM column, denoted as $\mathbf{r}_{tf}^{\Re} = \Re\{\mathbf{r}_{tf}\}$, while FNN2 learns the imaginary part $\mathbf{r}_{tf}^{\Im} = \Im\{\mathbf{r}_{tf}\}$. The TF CDDPM column is given by the inverse symplectic finite Fourier transform (ISFFT) of the unvectorized CDDPM column $\mathbf{R}_{dd} = \text{vec}_{M,N}^{-1}(\hat{\mathbf{r}}(\tau, \nu))$ as

$$\mathbf{r}_{tf} = \text{vec}(\mathbf{R}_{tf}) = \text{vec}(\mathbf{F}_M \mathbf{R}_{dd} \mathbf{F}_N^H), \tag{10}$$

where $\mathbf{F}_N = \frac{1}{\sqrt{N}} \left\{ e^{-j2\pi \frac{mq}{N}} \right\}_{m,q=0}^{N-1}$ is a N-point discrete Fourier transform (DFT) matrix. Hermitian conjugate operation is applied to perform inverse DFT.

Hence, the two network outputs are combined and reshaped to obtain $\mathbf{R}_{tf}$. Then, the SFFT is applied to obtain the DD matrix, which, once vectorized, provides the desired CDDPM column.

The number of hidden layers is 2. Increasing the number of hidden layers does not enhance performance but instead increases latency. The number of neurons per hidden layer is denoted as L_1 for the first hidden layer and L_2 for the second hidden layer. The input dimension of the network is 2, corresponding to a delay-Doppler pair. The last layer, with MN neurons, is used to obtain the desired output dimension, and a linear activation function is used to span the real set.

B. Offline Training

The FNN-based architecture is trained to learn the desired relation. Training is performed using the backpropagation algorithm [13] and the Adam optimizer. The hyperparameters used during the training phase are reported in Table I. The learning rate is halved every 10 epochs. The objective function is the L1 loss function. Denoting $r_{tf}[n]$ the output of the n-th neuron in the output layer and $r_{tf}^{d}[n]$ the desired value for that

TABLE I
HYPER-PARAMETERS FOR TRAINING.

Number of training samples	200000
Mini batch size	1000
Number of epochs	100
Initial learning rate	0.001
Learning rate drop factor	0.5
Learning rate drop period	10

output, the L1 loss function is defined as

$$\mathscr{E} = \sum_{n=1}^{MN} \left| r_{tf}^{d}[n] - r_{tf}[n] \right|. \tag{11}$$

Training is carried out offline as described above and in the same way for the two networks.

C. DL-based P-IPIC

The DL-based P-IPIC channel estimation algorithm works as follows: during the cost function maximization, the required CDDPM column is obtained using FNN predictions. After that, before computing the channel gains vector, in order to maintain high estimation accuracy, the CDDPM column corresponding to the detected path is evaluated according to (5) and stored. Hence, the maximum number of evaluations of (5), considering both *search phase* and *refinement phase*, is $2P_{\max}$. The algorithm flow is summarized in Algorithm 1.

The DL-based P-IPIC requires the following parameters: $P_{\max}$: maximum number of detectable paths; $s_{\max}$: maximum number of iterations during the fine estimation phase; $\epsilon_\tau, \epsilon_\nu$: convergence tolerance parameters for stopping criteria during fine estimate phase [4], [5]; and m_τ, n_ν: parameters that are used to define the search space dimension along delay and Doppler axes during the fine estimation. In particular the number of grid points is $\big(2\lfloor \frac{m_\tau}{2} \rfloor + 1\big)\big(2\lfloor \frac{n_\nu}{2} \rfloor + 1\big)$. Moreover, I_{dd} indicates the integer DD grid used during the coarse estimate phase while $F_{dd}^{(s)}$ indicates the fractional DD grid used during the s-th iteration of the fine estimation phase. In particular, $F_{dd}^{(s)} = \left\{ W_\tau^{(s)} \Gamma + \hat{\tau}_{\text{fine}}^{(s-1)} \right\} \times \left\{ W_\nu^{(s)} \Lambda + \hat{\nu}_{\text{fine}}^{(s-1)} \right\}$ ($\times$ denots Cartesian product) where $\hat{\tau}_{\text{fine}}^{(s-1)}$,$\hat{\nu}_{\text{fine}}^{(s-1)}$ are the DD parameter estimates obtained in the previous iteration, $\Gamma = \{-\lfloor \frac{m_\tau}{2} \rfloor, ..., 0, ..., \lfloor \frac{m_\tau}{2} \rfloor\}$ and $\Lambda = \{-\lfloor \frac{n_\nu}{2} \rfloor, ..., 0, ..., \lfloor \frac{n_\nu}{2} \rfloor\}$ are the spanning sets, and $W_\tau^{(s)} = \frac{\Delta\tau}{m_\tau^{s-1}}$,$W_\nu^{(s)} = \frac{\Delta\nu}{n_\nu^{s-1}}$ are the fine delay and Doppler resolutions, respectively. It should be noted that, when $s = 1$, $\hat{\tau}_{\text{fine}}^{(s-1)} = \hat{\tau}_{\text{coarse}}$, $\hat{\nu}_{\text{fine}}^{(s-1)} = \hat{\nu}_{\text{coarse}}$. In Algorithm 1 the DD parameters of the search areas are fed as inputs to the DL architecture in Fig. 1 to obtain $\mathbf{R}_{tf} = \text{vec}_{M,N}^{-1}\big(\text{FNN1}(\tau,\nu) + j\text{FNN2}(\tau,\nu)\big)\big)$ and then the estimate of the corresponding CDDPM vector as $\hat{\mathbf{r}}(\tau,\nu) = \text{vec}(\text{SFFT}(\mathbf{R}_{tf}))$. The estimated CDDPM column $\hat{\mathbf{r}}(\tau,\nu)$ is then used to evaluate the cost function during each phase.

D. Latency Reduction

The two FNNs must be designed properly in order to guarantee latency reduction by selecting a proper dimension for each layer. In particular, the complexity of the brute force evaluation of (5) is of the order of $\mathcal{O}(N^3M^4)$. On the other hand, the latency of the FNNs, assuming $L_1, L_2 > MN$, is dominated by the second hidden layer and is approximated by $2L_1L_2$. With this in mind, the number of neurons per layer can be selected to ensure a latency gain. Specifically, the layer dimensions should satisfy $L_1L_2 < \frac{N^3M^4}{2}$. In the following two cases are considered: $L_1 = L_2$ and $L_2 = 4L_1$. Assuming $L_1 = L_2$, the number of neurons per layer should be $L_1 < \sqrt{\frac{N^3M^4}{2}}$. Conversely, when $L_2 = 4L_1$ the dimension of the first hidden layer should satisfy $L_1 < \sqrt{\frac{N^3M^4}{8}}$. Moreover, in practical hardware implementation the use of FNNs facilitates the incorporation of parallelization, thus contributing to further reduction in latency.

Algorithm 1: DL-based P-IPIC

Input: $P_{\max},\ \epsilon,\ s_{\max},\ \epsilon_\tau,\ \epsilon_\nu,\ m_\tau,\ n_\nu,\ \mathbf{y}$
Initialize: $\mathbf{e}^{(0)} = \mathbf{y}$;
Search phase:
for $i = 1$ **to** $P_{\max}$ **do**
  Coarse estimate:
  $\hat{\tau}_{\text{coarse}}, \hat{\tau}_{\text{coarse}} = \arg\max_{(\tau,\nu)\in I_{dd}} \Phi_e\big(\hat{\mathbf{r}}(\tau,\nu)\big)$;
  Iterative fine estimate ($s = 1 : s_{\max}$):
  $\hat{\tau}_{\text{fine}}^{(s)}, \hat{\nu}_{\text{fine}}^{(s)} = \arg\max_{(\tau,\nu)\in F_{dd}^{(s)}} \Phi_e\big(\hat{\mathbf{r}}(\tau,\nu)\big)$
  $\hat{\tau}_i, \hat{\nu}_i = \hat{\tau}_{\text{fine}}^{(s)}, \hat{\nu}_{\text{fine}}^{(s)}$;
  Compute CDDPM column:
  $\mathbf{r}_i(\hat{\tau}_i, \hat{\nu}_i) = \boldsymbol{\Upsilon}_i(\hat{\tau}_i, \hat{\nu}_i)\text{vec}(\mathbf{X})$;
  Update channel gains vector:
  $\hat{\boldsymbol{\alpha}} = \Big(\mathbf{R}^H(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}}) + \lambda\mathbf{I}\Big)^{-1}\mathbf{R}^H(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\mathbf{y}$;
  Compute residue vector:
  $\mathbf{e}^{(i)} = \mathbf{y} - \mathbf{R}(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\nu}})\hat{\boldsymbol{\alpha}}$;
  Check stopping criterion:
  if $\|\boldsymbol{e}^{(i)}\| < \epsilon$ **then**
    $\hat{P} = i$;
    break;
Refinement phase ($i_{\text{ref}} = 1 : \hat{P}$) $\rightarrow \hat{\hat{P}}$;
Output: $\hat{\mathbf{H}}_{dd} = \sum_{i=1}^{\hat{\hat{P}}} \hat{\alpha}_i \boldsymbol{\Upsilon}_i(\hat{\tau}_i, \hat{\nu}_i)$;

IV. SIMULATION RESULTS

In this section, the simulation results comparing the DL-based approach with the model-based approaches are presented. In the following, two different cases are considered: $L_1 = L_2 = 8MN$ and $L_2 = 4L_1$ with $L_1 = 16MN$, where the dimension of the layers is selected according to the conditions reported in Section III-D. This analysis aims to investigate the learning properties of FNNs and the latency-performance trade-off.

A. Simulation Parameters

OTFS modulation with $M = N = 16$ with subcarrier spacing of $\Delta f = \frac{1}{T} = 25$ kHz and carrier frequency $f_c = 5.1$ GHz is considered. Thus, delay and Doppler resolutions are $\tau_{\text{res}} = 2.5\ \mu$s, and $\nu_{\text{res}} = 1.5625$ kHz. As in [4], [5], [11], the parameters for the channel estimation algorithms are $P_{\max} = 15,\ s_{\max} = 10,\ \epsilon_\tau = 10^{-10},\ \epsilon_\nu = 10^{-2},\ m_\tau =$

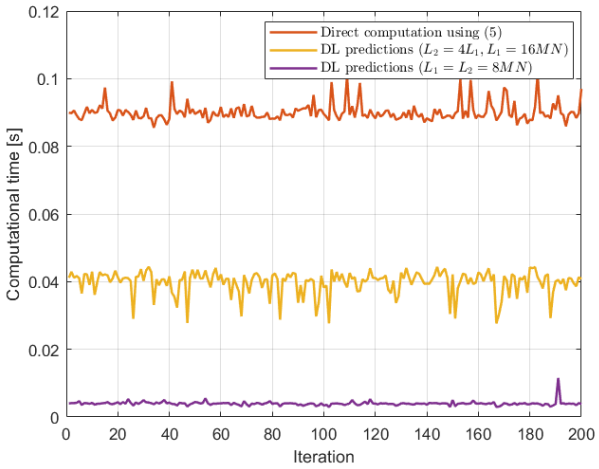

Fig. 2. Computational time for different inputs.

$n_\nu = 10$. Finally, as discussed above, the threshold ϵ for the stopping criterion is set according to the noise variance and OTFS frame parameters.

B. Simulation Scenario

The high mobility multipath channel is assumed to have P Rayleigh faded paths with exponential Power Delay Profile (PDP) with decaying time constant of 10 μs and maximum delay of 10 μs. In the considered case study, the propagation paths are close enough so that high IPI occurs. The channel is made of $P = 4$ propagation paths with delays $\boldsymbol{\tau} = [0 \quad 2.3 \quad 3.15 \quad 7.7]$ μs. The maximum User Equipment (UE) speed is $v_{UE} = 190$ $\frac{\text{m}}{\text{s}}(684\frac{\text{km}}{\text{h}})$ resulting into a maximum Doppler spread of $\nu_{\max} = \frac{v_{UE}}{c} f_c = 3.23$ kHz, where $c = 3 \cdot 10^8$ $\frac{\text{m}}{\text{s}}$ is the speed of light. The Doppler shifts of all paths are generated assuming Jakes' Doppler power spectrum (DPS) using $\nu_i = \nu_{\max} \cos(\theta_i)$ where $\theta_i \sim \mathcal{U}[0, 2\pi]$. For numerical results, a set of 10^2 channel realizations has been considered.

Performance metrics are evaluated as functions of the pilot signal-to-noise ratio (PSNR). The pilot SNR is given as

$$\text{PSNR} = \frac{E_p}{MN\sigma^2}. \tag{12}$$

C. Computational Time

Fig. 2 shows a comparison in terms of computational time between DL- and model-based approaches. The computational time obtained in simulation is shown for different inputs (200 DD pairs, randomly generated). It can be observed that the FNN-based architecture in Fig. 1 effectively enables latency reduction. In particular, in the case $L_1 = L_2 = 8MN$, the computational time is almost reduced by a factor of ten. On the other hand, in the case $L_2 = 4L_1$ with $L_1 = 16MN$, latency increases, as expected. In this case, the computational time is approximately halved (latency reduction of 55%). It is worth mentioning that the effective latency reduction depends on the practical hardware implementation, meaning that its actual advantage may vary. However, the simulation results demonstrate the potential of the proposed approach.

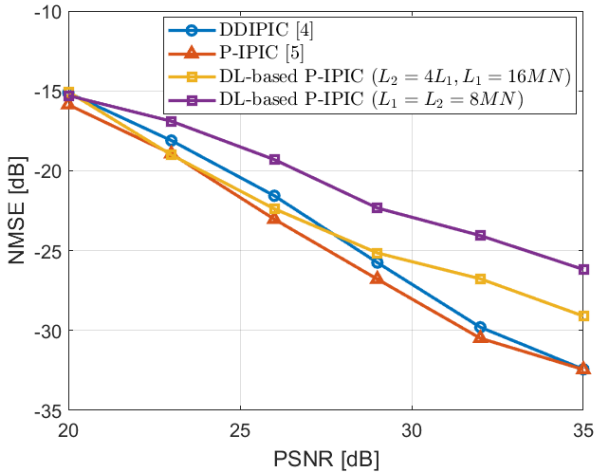

Fig. 3. NMSE as function of pilot SNR.

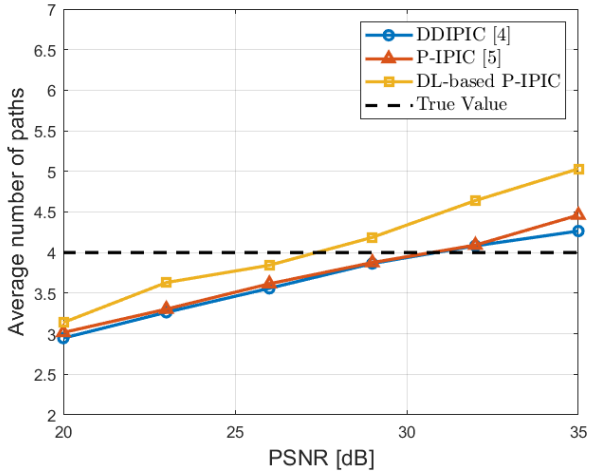

Fig. 4. Average number pf paths as function of pilot SNR.

D. NMSE vs Pilot SNR

The accuracy of channel estimation algorithms is measured through the Normalized Mean Squared Error (NMSE) computed as

$$\text{NMSE} = \mathbb{E}\left[\frac{\|\mathbf{H}_{dd} - \hat{\mathbf{H}}_{dd}\|_F^2}{\|\mathbf{H}_{dd}\|_F^2}\right]. \tag{13}$$

Fig. 3 shows the NMSE as a function of the pilot SNR. Considering the first case ($L_1 = L_2 = 8MN$), it can be observed that a performance loss of 6 dB is experienced at most, but with a drastically reduced latency. This is due to the limited learning capabilities of the FNNs. On the other hand, when the number of neurons per layer is increased, it can be seen that the DL-based approach offers, for PSNR values ranging from almost 22 dB to 26 dB, performance comparable to that of the P-IPIC algorithm. Thus, a performance gain of 1 dB is achieved compared to DDIPIC. On the contrary, at higher PSNR values, the DL-based approach exhibits a performance loss compared to model-based approaches. This performance loss is at most 3 dB and can be considered acceptable given the

advantage in terms of latency achieved using DL predictions. Moreover, this performance reduction can be attributed to the limited capabilities of neural networks to learn the desired relation. In particular, since P-IPIC is based on residuals, it is prone to error propagation, and at higher PSNR values, errors in the estimate lead to residual peaks that are interpreted as small amplitude paths by the algorithm [5].

E. Average Number of Paths vs Pilot SNR

Detection capabilities are investigated looking at the average number of paths detected by the algorithms as a function of the pilot SNR. Simulation results are shown in Fig. 4. It can be noticed that the DL-based approach ($L_2 = 4L_1$ with $L_1 = 16MN$) detects, on average, a higher number of propagation paths compared to model-based approaches. Moreover, as expected, at high PSNR values the DL-based P-IPIC overestimates the number of propagation paths by one. This results is consistent with the performance loss experienced in the NMSE at high PSNR values.

V. Conclusions and Future Works

In this work, a DL-based P-IPIC algorithm is proposed to perform low-latency channel estimation for the fractional DD case. Simulation results show that the DL-based P-IPIC can achieve good estimation performance at drastically reduced latency for low to mid PSNR values. In contrast, at higher PSNR values, an acceptable performance loss is unavoidable.

In future work, a DL-based single-step refinement procedure will be investigated to further reduce the latency and complexity of fractional channel estimation algorithms in high-IPI scenarios.

Acknowledgment

This study was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART").

References

[1] R. Hadani, S. Rakib, M. Tsatsanis, A. Monk, A. J. Goldsmith, A. F. Molisch, and R. Calderbank, "Orthogonal time frequency space modulation," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6.

[2] R. Hadani and A. Monk, "Otfs: A new generation of modulation addressing the challenges of 5g," 2018.

[3] Y. Hong, T. Thaj, and E. Viterbo, *Delay-Doppler Communications: Principles and Applications*. Netherlands: Elsevier, 2022, publisher Copyright: © 2022 Elsevier Inc. All rights reserved.

[4] S. P. Muppaneni, S. R. Mattu, and A. Chockalingam, "Channel and radar parameter estimation with fractional delay-doppler using otfs," *IEEE Communications Letters*, vol. 27, no. 5, pp. 1392–1396, 2023.

[5] M. Marchese, H. Wymeersch, P. Spallaccini, and P. Savazzi, "Progressive inter-path interference cancellation algorithm for channel estimation using orthogonal time–frequency space," *Sensors*, vol. 24, no. 4414, 2024. [Online]. Available: https://doi.org/10.3390/s24134414

[6] I. A. Khan and S. K. Mohammed, "Low complexity channel estimation for otfs modulation with fractional delay and doppler," 2021.

[7] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: An emerging interdisciplinary paradigm," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 133–139, 2020.

[8] L. Guo, P. Gu, J. Zou, G. Liu, and F. Shu, "Dnn-based fractional doppler channel estimation for otfs modulation," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 11, pp. 15 062–15 067, 2023.

[9] P. Raviteja, K. T. Phan, and Y. Hong, "Embedded pilot-aided channel estimation for otfs in delay–doppler channels," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4906–4917, 2019.

[10] Y. Gao, X. Xu, B. Hu, J. Du, W. Yang, and Y. Jin, "Channel estimation for mimo-otfs satellite communication: a deep learning-based approach," in *2024 33rd International Conference on Computer Communications and Networks (ICCCN)*, 2024, pp. 1–6.

[11] S. R. Mattu and A. Chockalingam, "Learning in time-frequency domain for fractional delay-doppler channel estimation in otfs," *IEEE Wireless Communications Letters*, vol. 13, no. 5, pp. 1245–1249, 2024.

[12] L. Gaudio, M. Kobayashi, G. Caire, and G. Colavolpe, "On the effectiveness of otfs for joint radar parameter estimation and communication," *IEEE Transactions on Wireless Communications*, vol. 19, no. 9, sep 2020.

[13] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. McMaster University, Hamilton: Pearson Education, Inc., 2009.