Adaptive Neural Network for Deconstructing Multi-dimensional Channel Kernels

Iresha Amarasekara, Zhibin Zou and Aveek Dutta
Department of Electrical and Computer Engineering
University at Albany SUNY, Albany, NY 12222 USA
{iamarasekara, zzou2, adutta}@albany.edu

***Abstract*—Waveform design using Eigenfunctions is proven to be optimal with respect to interference cancellation. However, ideal eigenfunctions are undesirable in real-time systems due to the channel kernel constraint and the computational complexity. In the machine learning community, the investigations on the eigenfunction decomposition are based on Mercer's theorem which requires the kernel to be symmetric which is hard to satisfy in communication systems. High Order Generalized Mercer's Theorem (HOGMT) shows the ability to decompose the multi-dimensional asymmetric kernel into eigenfunctions. Based on this, we proposed an equivalent Neural Network (NN) for the general channel kernel decomposition. Then Augmented Lagrangian Method (ALM) is applied to solve the constrained optimization problem as an unconstrained optimization problem, which avoids additional tuning rounds when the size of the kernel or the number of eigencomponents changes. We show the output of the proposed NNs convergences to eigenfunctions by both theory and simulations. The code is available at [].**

***Keywords*—Eigen-decomposition, Multi-dimensional channel decomposition, Adaptive Neural Networks.**

I. Introduction

The eigen decomposition is at the core of many communication systems such as equalization, channel characterization and Channel State Information (CSI) feedback. In the Multiple-Input Multiple-Output (MIMO) system, Singular Value Decomposition (SVD)-based precoding decomposes the spatial channel matrix into eigenvectors, parallel transmitting signals over which, the sum rate is achieved [1]. However, treating the channel as spatial channel matrices fails to capture the time, frequency and delay-Doppler dependence, thereby incapable of canceling the interference in these domains. Therefore, in Linear Time-Varying (LTV) channels *designing waveforms using eigenfunctions of the channel is optimal [2]*. However, eigenfunction decomposition by Mercer's theorem [3] is only applicable to the symmetric kernel, which is not always satisfied as the channel is a random operator [2], [4].

Multi-dimensional kernels and its decomposition: The general wireless channel can be represented as an *asymmetric kernel* from the first principle [5]–[7], which reflects the mapping relation from the transmitter and the receiver. High Order Generalized Mercer's Theorem (HOGMT) [8], [9] is a principled method to decompose a multi-dimensional[1] asymmetric kernel into jointly orthogonal eigenfunctions. [9], [10] show that precoding and modulation using eigenfunctions decomposed by HOGMT result in transmission over independent orthogonal subchannels in the eigen-domain, thereby avoiding interference from other symbols across all the DoF. Figure 1 shows partial representations of a spatio-temporal channel, where the time domain shows the time-delay profile, the space domain shows the channel gains between MIMO antennas and the space-time domain shows the delay profile across antennas. Unlike 1-dimensional eigenvectors in the MIMO channel matrix, the decomposed spatio-temporal eigenfunctions are represented by the 2-dimensional waveforms which achieve orthogonality in the time-space domain. However, in contrast to Fourier bases, commonly used in OFDM and OTFS, eigenfunctions are not pre-defined resulting in an open problem in the literature [4, Chapter 2.5]: "*Eigenfunctions are strictly unstructured, which makes them computationally expensive for practical purposes.*"

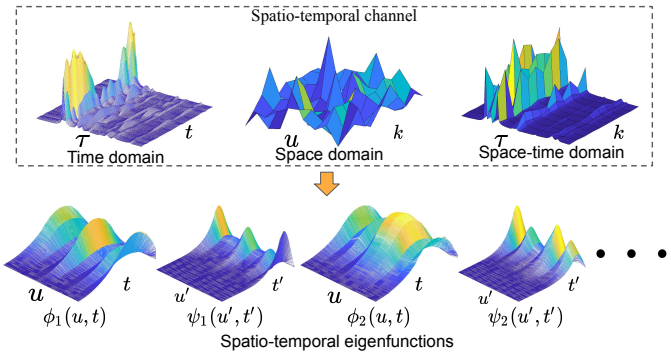


Figure 1: The time, space and space-time domain representations of the spatio-temporal channel, and the decomposed spatio-temporal eigenfunctions

Neural Network (NN) offers a distinct edge over linear methods as it inherently captures non-linear relationships among hidden variables with practical computational complexity. This encourages us to adopt NNs for the implementation of eigenfunction decomposition. Current NN-based eigen decomposition methods are based on Mercer's theorem or the SVD method which decompose eigenfunctions from the symmetric kernel [11] or the eigenvectors from the spatial channel matrix [12], incapable of decomposing multi-dimensional asymmetric channel kernels, such as rapidly time-varying MIMO channel kernels, MIMO-OFDM channel kernels, MIMO-OTFS channel kernels. Meanwhile, the existing NNs treat the constraint as a fixed penalty, which suffers

[1]In the context of the channel representation, multi-dimension includes time, space, frequency, delay-Doppler domains, etc.

from exhaustive tuning when the size of the channel kernel and the number of eigenfunctions change. We applied the ALM method to the optimization problem, where the Lagrange multiplier is automatically updated during the training process, thereby being adaptive to variant scenarios. The contributions of this paper are summarized as follows:

- We present a baseline NN for multi-dimensional asymmetrical kernel decomposition based on HOGMT, which solves the most general eigen decomposition problem as the symmetric kernel can be seen as the degeneration case of the asymmetric kernel.
- We improve the proposed NN by Augmented Lagrangian Method. The constrained optimization is converted to unconstrained optimization. The penalty coefficient is automatically updated during the training process, avoiding exhaustive tuning for different kernel types and the number of output eigenfunctions. Further, the proposed methods achieve lower complexity compared with SoTA.
- We validate the adaptivity, accuracy, the orthogonality and the convergence time of the proposed two NNs by extensive simulations. We further evaluate the output using eigenfunction-based waveform design methods.

II. Preliminary

A. General Channel Kernel

The time and space domain continuous input-output relations for LTV channel are shown as,

$$r(t)=\int g(t,\tau)s(t-\tau)\ d\tau \text{ and } r(u)=\int h(u,k)s(u-k)\ d\tau \tag{1}$$

where $g(t,\tau)$ and $h(u,k)$ are transfer functions in the time and space domains. By Fourier transform, the time domain can also be transferred to the frequency domain, resulting in the transfer function in the OFDM system. However, it would not change the general form of the input-output relation, which can be expressed as

$$r(Z)=\int H(Z;\Gamma)s(Z-\Gamma)\ d\Gamma=\int K(Z;Z')s(Z')\ dZ' \tag{2}$$

where $H(Z;\Gamma)$ is the multi-dimensional transfer function (in the time-varying MIMO channel $(Z;\Gamma)=(u,t;k,\tau)$, while in the MIMO-OFDM system, $(Z;\Gamma)=(u,f;k,\nu)$). $K(Z,Z')=H(Z,Z-Z')$ is the general channel kernel following the definition of kernel in [5], [7]. Notice $K(Z,Z')$ is asymmetric for the general channel ($K(X,Y)\neq K(Y,X)$).

B. High Order Generalized Mercer's Theorem

Mercer's theorem decomposes the symmetric kernel into eigenfunctions as

$$K(t,t')=\sum_{n=1}^{\infty}\lambda_n\phi_n(t)\phi_n(t') \tag{3}$$

where ϕ_n denotes the eigenfunction and α_n is the corresponding eigenvalue. A generalized version of Mercer's Theorem, called High Order Generalized Mercer's Theorem (HOGMT) has been recently proposed in [9]. This presents a mathematically principled approach to decompose multi-dimensional asymmetric channel kernels like in (2), into low-dimension, jointly orthogonal eigenfunctions, which is expressed as,

$$K(Z;Z')=\sum_{n=1}^{\infty}\sigma_n\psi_n(Z)\phi_n(Z') \tag{4}$$

where $\mathbb{E}\{\sigma_n\sigma_n'\}=\lambda_n\delta_{nn'}$. λ_n is the n-th eigenvalue and $\psi_n(Z)$ and $\phi_n(Z')$ are orthonormal eigenfunctions, i,e.,

$$\int\phi_n(Z')\phi_{n'}^*(Z')dZ'=\delta_{nn'} \text{ and } \int\psi_n(Z)\psi_{n'}^*(Z)dZ=\delta_{nn'} \tag{5}$$

These eigenfunctions are referred as *dual eigenfunctions* that exhibit the important *duality* property,

$$\int K(Z;Z')\phi_n^*(Z')\ dZ'=\sigma_n\psi_n(Z) \tag{6}$$

This property indicates that the eigenfunctions are transferred to their dual eigenfunction scaled only by the channel gains when transmitting over the channel.

C. Augmented Lagrangian Method

In this work, we adopt the Augmented Lagrangian Method (ALM) [13] to solve an equality-constrained optimization problem and present the formulation here for clarity. Consider an optimization problem of the form,

$$\text{minimize } F(x) \qquad \text{s.t. : } c_i(x)=0,\ \forall i=1,2,\dots,m, \tag{7}$$

where $x\in\mathbb{R}^n$. For notational convenience, we denote $c(x)\triangleq[c_1(x),c_2(x),\dots,c_m(x)]^\top\in\mathbb{R}^m$. Therefore, the Lagrangian function of (7) is defined as

$$L(x,\alpha)\triangleq F(x)+\sum_{i=1}^{m}\alpha_i c_i(x), \tag{8}$$

where $\alpha\triangleq[\alpha_1,\alpha_2,...,\alpha_m]^\top\in\mathbb{R}^m$ are the Lagrange multipliers. To solve (7) using the ALM [13], (7) is modified to

$$\begin{aligned}&\text{minimize } F(x)+\frac{\mu}{2}\|c(x)\|^2\\ &\text{subject to: } c_i(x)=0,\ \forall i=1,2,\dots,m,\end{aligned} \tag{9}$$

where μ is referred to as the penalty parameter. The corresponding Lagrangian function is defined as

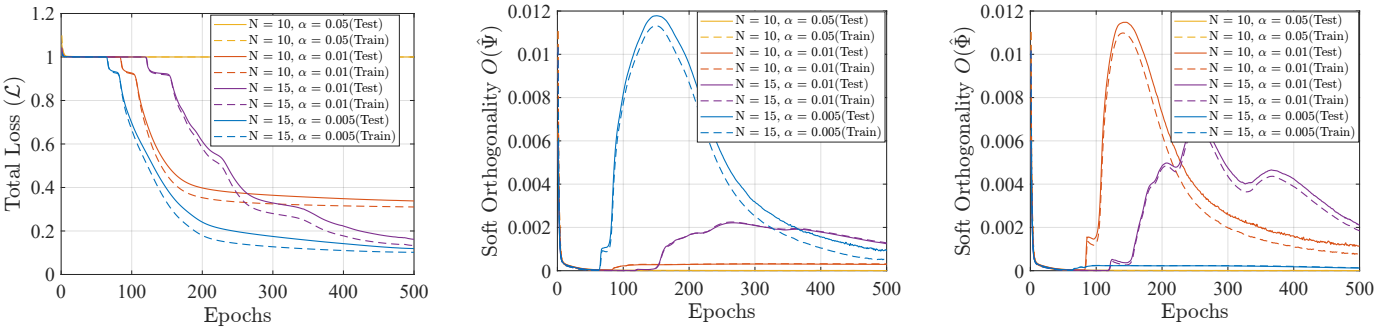
$$L_\mu(x,\alpha)\triangleq F(x)+\frac{\mu}{2}\|c(x)\|^2+\sum_{i=1}^{m}\alpha_i c_i(x), \tag{10}$$

which is referred to as the augmented Lagrange function. In all, the constrained optimization (7) is transformed into an unconstrained optimization of (10).

III. Neural Network for Asymmetric Kernels Decomposition

A. Approximating the Kernel with Finite Eigenfunctions

Theoretically, both Mercer's Theorem and HOGMT decompose the kernel into infinite eigenfunctions. However, we can only operate a finite number of eigenfunctions in reality.



(a) Total Loss $\mathcal{L}$ (b) Soft Orthogonality $O(\hat{\Phi})$ (c) Soft Orthogonality $O(\hat{\Phi})$

Figure 2: Baseline for N=10 and N=15 cases with different penalties.

Therefore, we convert the kernel decomposition problem to the approximation problem by N eigenfunctions as

$$\arg\min_{\hat{K}} \quad \mathbb{E}\{\|K(Z;Z')-\hat{K}(Z;Z')\|^2\} \tag{11a}$$

where $\hat{K}(Z;Z')$ is the approximated kernel as

$$\hat{K}(Z;Z')=\sum_{n=1}^{N}\sigma_n\hat{\psi}_n(Z)\hat{\phi}_n(Z') \tag{11b}$$

with the orthogonal constraint

$$\int\hat{\phi}_n(Z')\hat{\phi}^*_{n'}(Z')dZ'=\delta_{nn'} \text{ and } \int\hat{\psi}_n(Z)\hat{\psi}^*_{n'}(Z)dZ=\delta_{nn'} \tag{11c}$$

In this work, we focus on asymmetric kernels unless otherwise specified, as the symmetric kernel is just a special case of asymmetric kernel. Unlike Fourier methods provide complex exponentials as subcarriers, which are only independent in the time-invariant channels, eigenfunctions decomposed from the channel kernel remain independent over all kinds of channels. However, these eigenfunctions are not pre-defined, which suffers from high computational complexity, thereby limiting their practical implementation [14]. We postulate that *a neural network that solves the MMSE problem* (11a) *makes the outputs converge to eigenfunctions*.

B. HOGMT based Neural Network

We design an equivalent NN with equality-constrained objective function (12a) for solving (11a), whose outputs convergence to eigenfunctions decomposed by HOGMT from the universal approximation theorem [17], [18]

$$\mathcal{L}=\mathcal{J}+\sum_{i=1}^{2}\alpha_i\Omega_i \tag{12a}$$

where $\mathcal{J}$ is the MSE optimization

$$\mathcal{J}=\frac{1}{B}\sum_{b=1}^{B}\frac{\|\mathbf{K}_b-\sum_{n=1}^{N}\hat{\sigma}_{n,b}\hat{\Psi}_{n,b}\otimes\hat{\Phi}_{n,b}\|^2}{\|\mathbf{K}_b\|^2} \tag{12b}$$

where Ω_1 and Ω_2 are the regularizations to ensure the orthogonality constraint

$$\Omega_1=\frac{1}{B}\sum_{b=1}^{B}\sum_{n=1}^{N}\sum_{n'\neq n}^{N}\|\langle\hat{\Phi}_{n,b},\hat{\Phi}_{n',b}\rangle\| \tag{12c}$$

$$\Omega_2=\frac{1}{B}\sum_{b=1}^{B}\sum_{n=1}^{N}\sum_{n'\neq n}^{N}\|\langle\hat{\Psi}_{n,b},\hat{\Psi}_{n',b}\rangle\| \tag{12d}$$

It is easy to observe that the designed NN is equivalent to the MMSE optimization (11a). α_i is the penalty for the constraint Ω_i. Comparing (12a) and (8), we can see $\mathcal{L}$ is also a Lagrange function where α_i is the Lagrange multiplier.

The total loss is a general evaluation for NNs, which basically depends on the total eigenvalue for eigen decomposition. However, in the communication system, orthogonality is most critical for interference cancellation. The NN-based method is incapable of achieving strict orthogonality as the hard decision will limit the freedom of the output space and bring overfitting problems. Therefore, we define the *soft orthogonality* as

Definition 1. *Soft orthogonality of the eigenfunction set* $\{\hat{\Phi}\}$ *is defined as*

$$O(\hat{\Phi})=\frac{1}{N(N-1)}\sum_{n=1}^{N}\sum_{n'\neq n}^{N}\|\langle\hat{\Phi}_n,\hat{\Phi}_{n'}\rangle\| \tag{13}$$

where $O(\hat{\Phi})$=0 *means eigenfunctions* $\{\hat{\Phi}\}$ *are strictly orthogonal. Similarly,* $O(\hat{\Psi})$ *denotes the soft orthogonality for eigenfunctions* $\{\hat{\Psi}\}$.

We validate the baseline method using spatio-temporal channel kernels. More details about the NN architecture and the dataset are described in Section V-B. Figure 2 shows that with an appropriate penalty, the baseline shows the convergence with respect to both the total loss and the soft orthogonality. However, it suffers from two main drawbacks:

- The choice of penalty will dramatically affect the performance of baseline even in the same scenario. From Figure 2a, for $N=10$ case, $\alpha=0.01$ is much better than $\alpha=0.05$, where the latter is not even convergent.
- When changing the number of output, the same penalty can not ensure the same result, Figure 2a, 2b and 2c show

Table I: Comparing complexity of eigen decomposition

Eigen decomposition	Methods	Time complexity	Parameters
Spatial channel matrix $H \in \mathbb{C}^{N_t \times N_r}$	SVD [15]	$\mathcal{O}(\min(N_t N_r^2, N_t^2 N_r))$	N: number of eigen components; N_d: the order of dimensions; N_L: number of layers
	SVD-DNN [12]	$\mathcal{O}(\max(2N^2 N_t N_r, 2N^2 N_t^2, 2N^2 N_r^2))$	
Spatio-temporal channel tensor $K \in \mathbb{C}^{L_u \times L_t \times L_{u'} \times L_{t'}}$	HOSVD [16]	$\mathcal{O}(N_d \max(L_u L_t L_{u'}, L_u L_t L_{t'}, L_t L_{u'} L_{t'})^3)$	
	HOGMT [9]	$\mathcal{O}(\min(L_u L_t (L_{u'} L_{t'})^2, (L_u L_t)^2 L_{u'} L_{t'}))$	
	Baseline	$\mathcal{O}(2N N_L L_u L_t L_{u'} L_{t'})$	
	HOGMT-ALM	$\mathcal{O}(2N N_L L_u L_t L_{u'} L_{t'})$	

that, for the same $\alpha = 0.01$, $N = 15$ is not comparable with $N = 10$ case. It requires extra tuning of the penalty to $\alpha = 0.005$.

These motivate us to propose an adaptive HOGMT-NN, which can ensure the convergence of the output and avoid exhaustive tuning for different scenarios.

IV. Adaptive Neural Network for Eigen Decomposition

A. HOGMT-ALM: ALM Based Neural Network for HOGMT

In practice we may encounter different kernel size (N) depending on the communication environment. In such cases, Baseline will require exhaustive tuning for α_i, which is critical to maintain the orthogonality constraint in (12a). It is evident that the optimal Lagrange multiplier values α^* are tightly coupled with the optimal solution x^* resulting from optimizing (10) [13]. Note that, in neural networks the role of the program variable x is played by the weights of the neural network w. When it comes to neural networks-based kernel decomposition methods, for kernels of different dimensions and different numbers of eigenfunctions, neural networks will have different input layer sizes and output layer sizes. Besides such structural factors, the eventual optimal weights (x^*) of the neural network will also depend on the input-output data used to train the network. Consequently, the dimensions of x^* as well as the numerical values of x^* are both subject to changes even if we keep the structure of the hidden layers of the neural networks unchanged during such scenarios. Therefore, due to the said coupling between x^* and α^*, it is not guaranteed that α^* will be the same for all possible such scenarios. In all, we can't train neural networks while keeping the corresponding Lagrange multipliers fixed at some arbitrarily chosen predefined value. This motivates dynamic adaptation of the used Lagrange multipliers during the training process.

Therefore, we incorporate the ALM method and the objective function (12a) is modified as

$$\mathcal{L} = \mathcal{J} + \sum_{i=1}^{2} A_i^T \Omega_i' + \frac{\mu}{2} \sum_{i=1}^{2} \|\Omega_i'\|^2 \tag{14a}$$

where, μ is the penalty parameter,

$$A_i \triangleq [\alpha_{i,1}, \alpha_{i,2}, ..., \alpha_{i,K}]^T \in \mathbb{R}^{2K}, \tag{14b}$$

is the vector containing the Lagrange multipliers, and

$$\Omega_1' \triangleq \frac{1}{B} \sum_{b=1}^{B} [\mathrm{Re}(\tilde{\Phi}_{1,b}), \mathrm{Im}(\tilde{\Phi}_{1,b}), ..., \mathrm{Re}(\tilde{\Phi}_{K,b}), \mathrm{Im}(\tilde{\Phi}_{K,b})]^T,$$

$$\Omega_2' \triangleq \frac{1}{B} \sum_{b=1}^{B} [\mathrm{Re}(\tilde{\Psi}_{1,b}), \mathrm{Im}(\tilde{\Psi}_{1,b}), ..., \mathrm{Re}(\tilde{\Psi}_{K,b}), \mathrm{Im}(\tilde{\Psi}_{K,b})]^T, \tag{14c}$$

where $\tilde{\Phi}_{k,b}$ and $\tilde{\Psi}_{k,b}$ is the inner product of one pair $(\hat{\Phi}_{n,b}, \hat{\Phi}_{n',b})$ and $(\hat{\Psi}_{n,b}, \hat{\Psi}_{n',b})$ for $n \neq n'$, respectively. There are $K = N(N-1)$ pairs for each eigenfunction set.

The main idea behind HOGMT-ALM is to eliminate the constraints of the constrained optimization problem (7) and get x closer to x^*. Here x^* is an unconstrained local minimum of $L_\mu(\cdot, \alpha)$ under two conditions [13]:

- Choosing α closer to α^*: If $\mu > \bar{\mu}$, by taking α closer to α^*, x can be approximated to x^* by solving the unconstrained minimization of $L_\mu(\cdot, \alpha)$. To achieve this condition, α can be updated iteratively based on the following rule (inspired by the gradient ascent, or, in other words, to enforce each product $\alpha_i c_i(x) > 0, \forall i$):
$$\alpha^{k+1} = \alpha^k + \mu^k c(x^k). \tag{15}$$
- Choosing a sufficiently large value for μ: Using a high μ takes an unconstrained local minimum of $L_\mu(x, \alpha)$ closer to a feasible point as larger μ means higher cost for infeasibility. Consequently, this leads to a good approximation to x^*. To achieve this condition, similar to α, μ can also be updated based on the following rule (as proposed in [13]):
$$\mu^{k+1} = \begin{cases} \beta \mu^k & \text{if } \|c(x^k)\| > \gamma \|c(x^{k-1})\|, \\ \mu^k & \text{else.} \end{cases} \tag{16}$$
where $\beta > 1$ and $\gamma < 1$ are predefined design parameters that characterize the ascent and descent of μ^k and $c(x^k)$ over the iterations (k), respectively.

Finally, we point out that, to get x closer to x^*, A_i is updated during the training process according to Ω_i' and the μ, as shown in Algorithm 1.

B. Complexity Analysis

The complexity for eigenvector and eigenfunction decomposition is shown in Table I. Both SVD and DNN-SVD are designed to decompose 2-D matrices only. While High-Order SVD (HOSVD), HOGMT, and proposed methods decompose multi-dimensional tensors. The relationship between SVD, HOSVD and HOGMT can be found in Lemma 3 of [9].

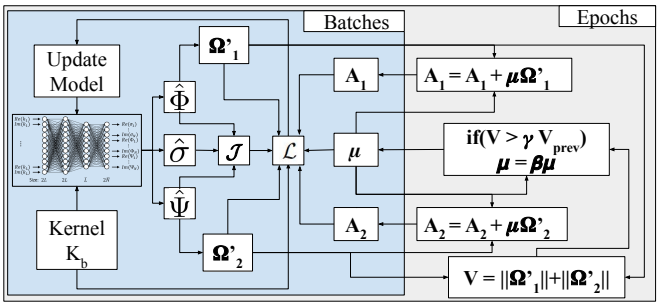
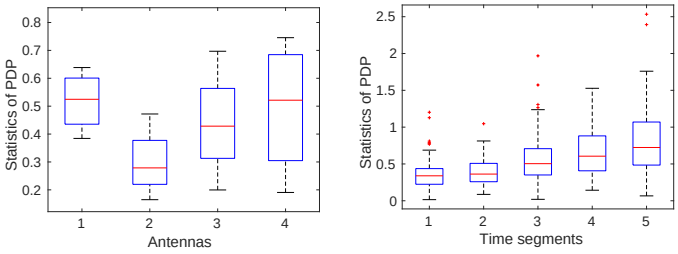


Figure 3: HOGMT−ALM Adaptive Learning Architecture



(a) PDP distribution in space domain at $t = t' = 1$

(b) PDP distribution in time domain at $u = u' = 1$

Figure 4: Statistics of the channel kernel over space and time

Since both baseline and HOGMT−ALM use fully connected architecture (Section V), their time complexities are the same, which depends on the number of layers (N_L), the size of the input ($2L_uL_tL_{u'}L_{t'}$ for complex-valued NN) and the size of the output (N). It's clear that proposed NNs have less complexity than HOSVD. Meanwhile, for a fixed number N and N_L, both baseline and HOGMT−ALM increase at a much slower rate than HOGMT with respect to the increasing size of the input tensor.

V. HOGMT−ALM Implementation

A. Neural Network Architecture

In order to decompose the spatio-temporal channel kernel $K(u,t;u',t') \in \mathbb{C}^{L_u \times L_t \times L_{u'} \times L_{t'}}$ that has $L \triangleq L_uL_tL_{u'}L_{t'}$ parameters, the baseline comprises a 4-layer fully connected feed-forward neural network. However, this neural network handles only real inputs and outputs. Because of that, the parameters of the kernel were split into real and imaginary parts. The size of each layer is as follows: input and output layers have $2L$ and $\tilde{N} \triangleq N(1+L_uL_t+L_{u'}L_{t'})$ nodes, respectively, while two hidden layers have $2L$ and $\tilde{L} \triangleq L+\tilde{N}$ nodes. As the activation function of hidden linear layers, the LeakyRelu activation function with a negative slope of 0.1 is used. The neural network architecture of HOGMT−ALM is the same as the architecture of baseline. However, training is different for both.

B. Data Generation

We generated 5120 time-varying MIMO channel kernels. Each of them is with $L_u \times L_{u'}$ antennas for $L_t = L_{t'}$ moments (time slots), where $L_u = L_{u'} = 4$ and $L_t = L_{t'} = 5$ in our work. The delay taps are randomly generated in the interval $[3 \quad 5]$. The distribution of Power Delay Profile (PDP) at each moment of the generated kernel changes over space and time as shown in Figure 4a and Figure 4b, respectively. A training-test split of 80%-20% is used to evaluate the model.

C. Training

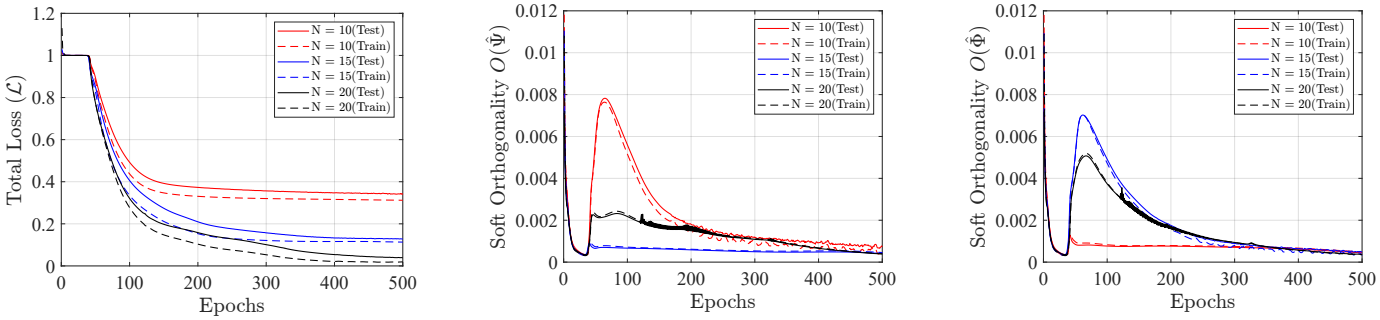
Figure 3 shows how different training-related parameters are updated batch-wise and epochs-wise during the training process of HOGMT−ALM. The training is done as mini-batches where the chosen mini-batch size is 16. The Adam optimizer with a learning rate of 1×10^{-5} is used for training. As shown in figure 3, there are a few adaptive parameters involved in the training, such as A_1, A_2 and μ, that depend on Ω'_1 and Ω'_2. These parameters are updated during each epoch, relative to the Ω'_1 and Ω'_2 of the last mini-batch of the previous epoch using (15) for A_1 and A_2 and (16) for μ where γ = 0.75 and β = 1.01.

Algorithm 1 HOGMT−ALM Training

Inputs $A_1^{[0]}, A_2^{[0]}, \mu^{[0]}, \gamma$;
for $k \leftarrow 0$ to T **do**
  for *mini batches* $\leftarrow 0$ to *data size/batch size* **do**
    $x_b \leftarrow$ split *real* and *imaginary* parts of X_b
    $Y_b = NN\_Model(x_b)$
    Derive $\hat{\sigma}_{n,b}$, $\hat{\Phi}_{n,b}$ and $\hat{\Psi}_{n,b}$ for each eigenfunction
    Calculate $\mathcal{J}$ according to (12b)
    Calculate Ω'_1 and Ω'_2 according to (14c)
  end for
  Update Lagrange multipliers
  $A_1^{[k+1]} \leftarrow A_1^{[k]} + \mu\Omega'_1$
  $A_2^{[k+1]} \leftarrow A_2^{[k]} + \mu\Omega'_2$
  if $\|\Omega'_1\| + \|\Omega'_2\| > \gamma(\|\Omega'_1\|^{[k-1]} + \|\Omega'_2\|^{[k-1]})$ **then**
    $\mu^{[k+1]} \leftarrow \beta\mu^{[k]}$
  else
    $\mu^{[k+1]} \leftarrow \mu^{[k]}$
  end if
end for

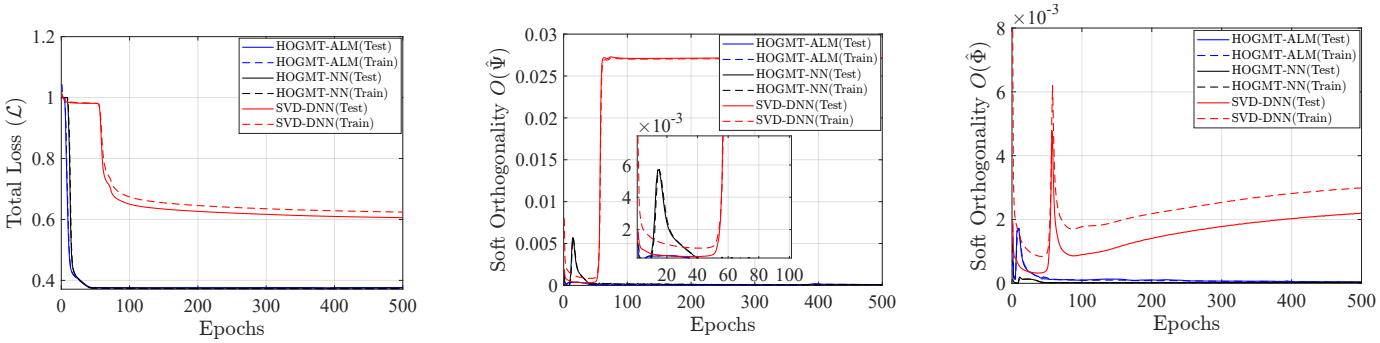
VI. Results

Adaptive to different number of eigenfunctions: We validate the HOGMT−ALM using the above dataset for $N = 10$, $N = 15$ and $N = 20$ cases with the same initialization of A_1, A_2, μ and γ. Figure 10 shows that both total loss and soft orthogonality converge for different N, which indicates HOGMT−ALM is adaptive to the required number of eigenfunctions without changing the architecture and tuning the hyper-parameters. Further, unlike the baseline method unable to ensure the convergent behavior, the outputs of HOGMT−ALM start to converge around the same time (number of epochs), which makes the convergent performance more predictable. From the practical and system views, this property provides a time reference for the validation of training, meaning it does



(a) Total Loss $\mathcal{L}$ (b) Soft Orthogonality $O(\hat{\Phi})$ (c) Soft Orthogonality $O(\hat{\Psi})$

Figure 5: Performance of HOGMT−ALM for 4-D channel kernels with N=10, 15, and 20



(a) Total Loss $\mathcal{L}$ (b) Soft Orthogonality $O(\hat{\Psi})$ (c) Soft Orthogonality $O(\hat{\Phi})$

Figure 6: HOGMT−ALM, baseline, and SVD-DNN for 2-D channel kernels with N=10

not require waiting for *uncertain* epochs to check if the result is convergent. Figure 10a shows larger N results in less total loss. Because with more eigenfunctions, the kernel can be approximated more accurately. Figure 10b and 10b show that two soft orthogonalities converge to less than 0.001 within 500 epochs, which is not guaranteed by baseline.

Adaptive to the degeneration case: The above results validates the performance of HOGMT−ALM for multi-dimensional channels. To compare with the state-of-the-art method, SVD-DNN [12], we generate 16×16 spatial channel matrices as the dataset, since SVD is designed only for 2-D matrices. We still use the same architecture and hyper-paprameters for HOGMT−ALM as in the 4-D channel kernel case. Both baseline and SVD-NN set penalty as α=0.01. The baseline achieves similar performance as HOGMT−ALM in the degeneration case as the 2-D kernel is less complex than the 4-D kernels. However, it still requires setting a proper penalty. HOGMT−ALM outperforms SVD-DNN, which indicates HOGMT−ALM is adaptive to different channel types without changing the penalty. More comparisons are shown in Appendix A [19]

Other Applications of HOGMT−ALM: One common application of eigenfunctions in wireless communications is multiplexing. Unlike kernel approximation and Principal Component Analysis (PCA), which care about the MSE and orthogonality, multiplexing in communications requires the decomposed eigenfunctions to have duality (6) as it significantly affects the demultiplexing performance at the receiver. Therefore, we further proposed an algorithm to ensure the duality and implement the multiplexing by output eigenfunctions. The result is shown in Appendix B [19]

VII. Conclusion

In this work, we proposed an equivalent NN for implementing HOGMT as a baseline, which decomposes the multi-dimensional channel kernel into eigenfunctions. Then we proposed an adaptive NN called HOGMT−ALM based on both HOGMT and ALM methods, which is adaptive to the different channel kernels, number of users and eigenfunctions, avoiding the exhaustive tuning for the penalty in the baseline method.

References

[1] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO-OFDM wireless communications with MATLAB*. John Wiley & Sons, 2010.

[2] K. Liu, T. Kadous, and A. Sayeed, "Orthogonal time-frequency signaling over doubly dispersive channels," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2583–2603, 2004.

[3] J. Mercer, "Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 209, pp. 415–446, 1909. [Online]. Available: http://www.jstor.org/stable/91043

[4] S. S. Das and R. Prasad, *Orthogonal Time Frequency Space Modulation: OTFS a Waveform for 6G*. CRC Press, 2022.

[5] P. Bello, "Characterization of Randomly Time-Variant Linear Channels," *IEEE Transactions on Communications Systems*, vol. 11, no. 4, pp. 360–393, 1963.

[6] L. A. Zadeh, "Frequency analysis of variable networks," *Proceedings of the IRE*, vol. 38, no. 3, pp. 291–299, 1950.

[7] F. Hlawatsch and G. Matz, *Wireless Communications Over Rapidly Time-Varying Channels*, 1st ed. USA: Academic Press, Inc., 2011.

[8] Z. Zou, M. Careem, A. Dutta, and N. Thawdar, "Unified Characterization and Precoding for Non-Stationary Channels," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 5140–5146.

[9] Z. Zou, M. Careem, A. Dutta, and N. Thawdar, "Joint Spatio-Temporal Precoding for Practical Non-Stationary Wireless Channels," *IEEE Transactions on Communications*, vol. 71, no. 4, pp. 2396–2409, 2023.

[10] Z. Zou and A. Dutta, "Multidimensional eigenwave multiplexing modulation for non-stationary channels," in *Globecom 2023 (Accepted)*.

[11] Z. Deng, J. Shi, and J. Zhu, "Neuralef: Deconstructing kernels by deep neural networks," in *International Conference on Machine Learning*. PMLR, 2022, pp. 4976–4992.

[12] T. Peken, S. Adiga, R. Tandon, and T. Bose, "Deep learning for SVD and hybrid beamforming," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6621–6642, 2020.

[13] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2016.

[14] K. Liu, T. Kadous, and A. Sayeed, "Orthogonal time-frequency signaling over doubly dispersive channels," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2583–2603, 2004.

[15] G. H. Golub and C. F. van Loan, *Matrix Computations*, 4th ed. JHU Press, 2013. [Online]. Available: http://www.cs.cornell.edu/cv/GVL4/golubandvanloan.htm

[16] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[17] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, no. 3, pp. 183–192, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900038

[18] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[19] I. Amarasekara, Z. Zou and A. Dutta, "Supplementary Material: Adaptive Neural Network for Deconstructing Multi-dimensional Channel Kernels." [Online]. Available: https://www.dropbox.com/s/mjlowl23ciav03x/Appendix_MEM.pdf?dl=0

APPENDIX A
ADAPTIVE TO DIFFERENT KERNEL SIZES

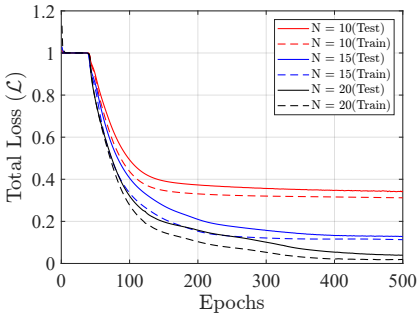
We further compare HOGMT−ALM, baseline and SVD-DNN for different 2-D channel kernels without changing the hyper-parameters.

Compare HOGMT−ALM, baseline and SVD-DNN for 8x8, 12x12, 24x24 and 32x32 matrices

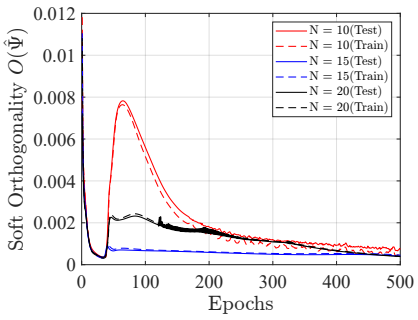
APPENDIX B
HOGMT−ALM FOR MULTIPLEXING

Algorithm 2 change it to duality algorithm

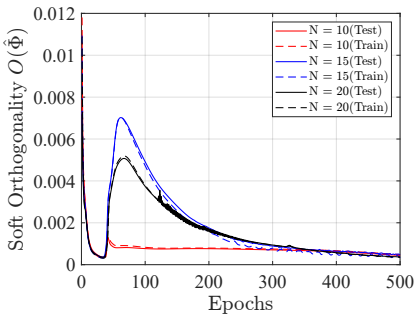
Inputs $A_1^{[0]}, A_2^{[0]}, \mu^{[0]}, \gamma$;
for $k \leftarrow 0$ to T **do**
  for $mini\ batches \leftarrow 0$ to $data\ size/batch\ size$ **do**
    $x_b \leftarrow$ split $real$ and $imaginary$ parts of X_b
    $Y_b = NN\_Model(x_b)$
    Derive $\hat{\sigma}_{n,b}$, $\hat{\Phi}_{n,b}$ and $\hat{\Psi}_{n,b}$ for each eigenfunction
    Calculate $\mathcal{J}$ according to (12b)
    Calculate Ω_1' and Ω_2' according to (14c)
  end for
  Update Lagrange multipliers
  $A_1^{[k+1]} \leftarrow A_1^{[k]} + \mu\Omega_1'$
  $A_2^{[k+1]} \leftarrow A_2^{[k]} + \mu\Omega_2'$
  if $\|\Omega_1'\| + \|\Omega_2'\| > \gamma(\|\Omega_1'\|^{[k-1]} + \|\Omega_2'\|^{[k-1]})$ **then**
    $\mu^{[k+1]} \leftarrow \beta\mu^{[k]}$
  else
    $\mu^{[k+1]} \leftarrow \mu^{[k]}$
  end if
end for



(a) Total Loss $\mathcal{L}$

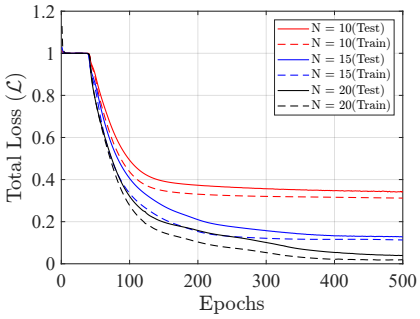


(b) Soft Orthogonality $O(\hat{\Phi})$

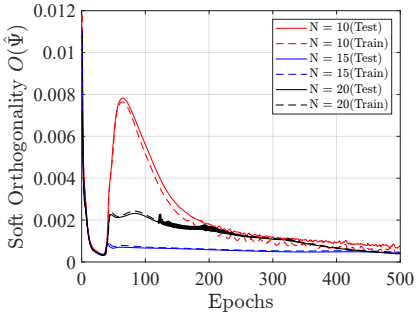


(c) Soft Orthogonality $O(\hat{\Psi})$

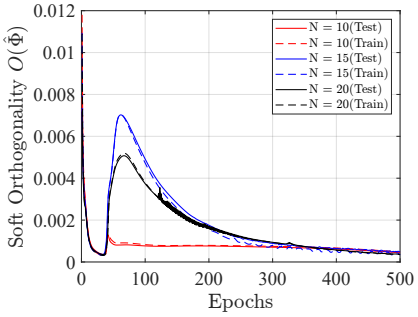
Figure 7: 8x8 2-D kernel



(a) Total Loss $\mathcal{L}$

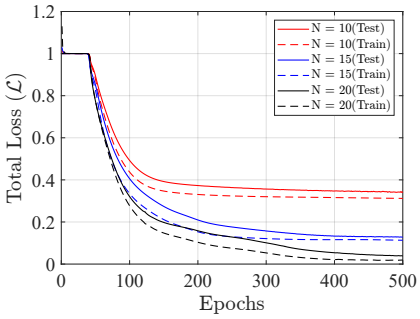


(b) Soft Orthogonality $O(\hat{\Phi})$

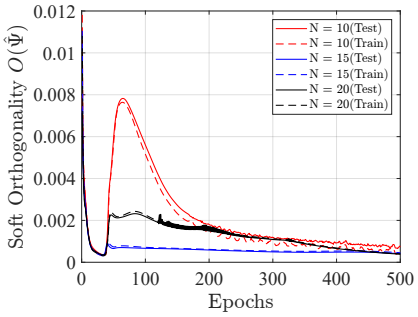


(c) Soft Orthogonality $O(\hat{\Psi})$

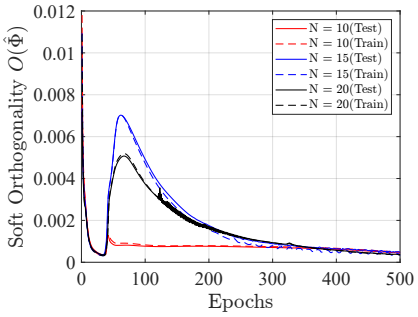
Figure 8: 12x12 2-D kernel



(a) Total Loss $\mathcal{L}$

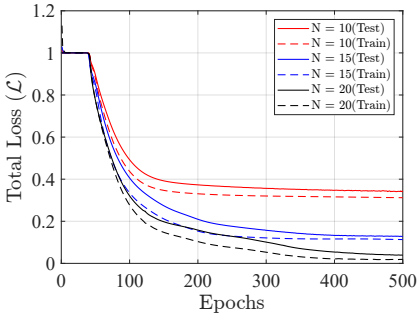


(b) Soft Orthogonality $O(\hat{\Phi})$

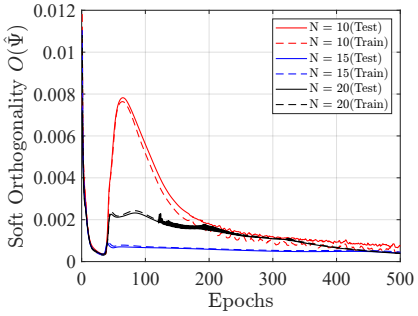


(c) Soft Orthogonality $O(\hat{\Psi})$

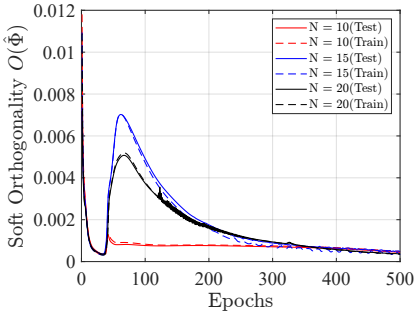
Figure 9: 24x24 2-D kernel



(a) Total Loss $\mathcal{L}$

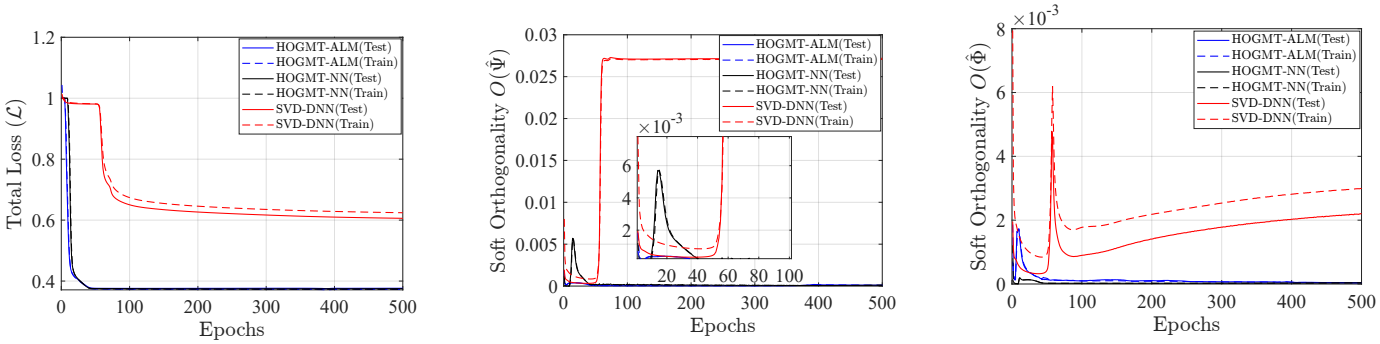


(b) Soft Orthogonality $O(\hat{\Phi})$



(c) Soft Orthogonality $O(\hat{\Psi})$

Figure 10: 32x32 2-D kernel



(a) Total Loss $\mathcal{L}$ (b) Soft Orthogonality $O(\hat{\Psi})$ (c) Soft Orthogonality $O(\hat{\Phi})$

Figure 11: HOGMT−ALM, baseline, and SVD-DNN for 2-D channel kernels with N=10 Compare with duality-NN