Learning the MMSE Channel Predictor

Nurettin Turan and Wolfgang Utschick
Professur für Methoden der Signalverarbeitung, Technische Universität München, 80290 Munich, Germany
Email: {nurettin.turan,utschick}@tum.de

***Abstract*—In this work a feed-forward neural network-based channel predictor is derived, where assumptions on a physical wave propagation channel model in a fading scenario are incorporated into the design procedure of the predictor. We start with the general expression of an approximated minimum mean squared error (MMSE) predictor and derive a predictor having the structure of a feed-forward neural network by making two key assumptions. By properly training this neural network it is possible to compensate the approximation errors due to these assumptions. It is further possible to outperform the linear MMSE (LMMSE) predictor with perfect knowledge of the statistical moments of second order based on the covariance function for specific channel model assumptions, especially for low SNR values.**

***Index Terms*—time-variant channel state information, minimum mean squared error prediction, machine learning, neural networks**

I. Introduction

Channel state information (CSI) estimation and prediction is one major task in wireless communication systems. It is beneficial to have CSI at the transmitter, i.e., at the base station (BS) to increase the achievable transmission rate in a wireless communication system [1]. In high mobility scenarios, where the users are moving with relatively high velocities, the CSI knowledge at the transmitter side may get outdated rapidly. This problem can be tackled with accurate channel prediction.

In [2], a low-complexity convolutional neural network (CNN)-based channel estimator was derived, where assumptions on the 3GPP spatial channel model were incorporated into the design procedure of the estimator [3], [4]. Following the derivation of the CNN-based channel estimator from [2], a similar approach was followed in [5] to derive a feed-forward neural network-based channel predictor, where assumptions on a physical wave propagation channel model in a fading scenario were incorporated into the design procedure of the predictor. The derivation of the neural network based channel predictor starts with a reformulation of the general expression of the linear minimum mean squared error (LMMSE) predictor. By making two key assumptions, it is possible to derive a predictor, which has the structure of a feed-forward neural network. In this way, we obtain the initialization weights and biases of the neural network predictor, which is then further trained offline to achieve a performance enhancement as compared to the untrained case.

In [5], the neural network predictor performance was evaluated using a ray-tracing-based indoor scenario of the generic DeepMIMO dataset [6]. However, in the following we want to focus more on the derivation of the offline learning-based feed-forward neural network predictor and its capability to serve as a blueprint predictor for specific communication scenarios, e.g, line of sight (LOS) scenarios as in [5]. We further present simulation results using the physical wave propagation channel model, with which the neural network-based predictor is derived. With the offline-learned neural network predictors it is possible to outperform the LMMSE predictor based on the Jakes assumption of the underlying Doppler spectrum.

A. Notation:

Given a vector $\mathbf{x} \in \mathbb{C}^K$, its transpose and the conjugate transpose are denoted by $\mathbf{x}^T$ and $\mathbf{x}^H$, respectively. The modulus $|\mathbf{x}|$ and the exponential function $\exp(\mathbf{x})$ are applied element-wise. With $\mathrm{diag}(\mathbf{x})$ we denote the square matrix with the entries of $\mathbf{x}$ on its main diagonal and zero elsewhere. The circular convolution of two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^K$ is given by $\mathbf{x} * \mathbf{y} \in \mathbb{C}^K$. The $K \times K$ identity matrix is $\mathbf{I}_K$ and the all ones vector is denoted by $\mathbf{1}$.

II. Neural Network-Based Channel Prediction

The physical wave propagation channel model used to derive the feed-forward neural network-based predictor is constructed by the superposition of P plane-waves impinging at a user, which moves with a constant velocity v [1], [7]. These plane-waves correspond to paths, which are mainly determined by path-specific Doppler shifts $f_p = \cos(\delta_p) f_c v/c$, and phases ψ_p, where f_c is the carrier frequency, c the speed of light and δ_p the direction of arrival (DoA) of path p. The Doppler bandwidth is defined as $B_D = f_c v/c$ and is the maximum possible Doppler shift. We assume that each path-phase ψ_p and each path specific DoA δ_p are uniformly distributed over the interval $[-\pi, \pi)$ and that the path specific Doppler shifts and phases do not change over a block of $M_o + N_p$ symbols, where M_o is the *observation length*, N_p is the *prediction length* [1], [7]. The *symbol duration* T_s, is much longer than the delay spread of the channel, thus, we have a frequency-flat channel [1]. Following the argumentation of [1], [7] the channel coefficients $h[m]$ are constructed by:

$$h[m] = \sum_{p=0}^{P-1} \frac{1}{\sqrt{P}} e^{j\psi_p} e^{j2\pi f_p T_s m} = \sum_{p=0}^{P-1} a_p e^{j2\pi f_p T_s m}, \quad (1)$$

with $m = 0, \ldots, M_o + N_p - 1$. An example of the fading process with three propagation paths ($P = 3$) is depicted in Fig. 1, where the black dots represent channel coefficients in the *observation interval* $\mathcal{I}_{M_o} = \{0, 1, \ldots, M_o - 1\}$ and the red dots represent channel coefficients in the *prediction interval*

978-1-7281-7440-2/20/$31.00 ©2020 IEEE

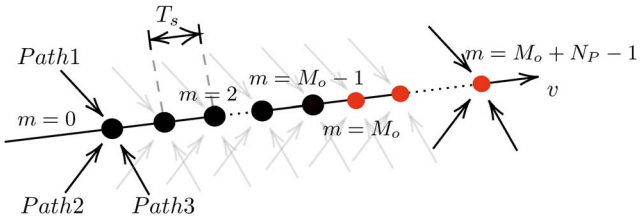

Fig. 1. Fading process channel construction example with $P = 3$.

$\mathcal{I}_{N_p} = \{M_o, M_o+1, \ldots, M_o+N_p-1\}$. If $P \to \infty$, the channel coefficients follow a Gaussian distribution based on the central limit theorem. However, if the channel is constructed with a few paths only, the obtained channel coefficients are distributed non-Gaussian.

The time-variant block-fading model is a zero mean and unit variance process, which is wide-sense stationary over a block consisting of the union of the observation interval $\mathcal{I}_{M_o}$ and the prediction interval $\mathcal{I}_{N_p}$ [1], [8]. After observing all channel coefficients in the observation interval $\mathcal{I}_{M_o}$, a pre-selected channel coefficient out of the prediction interval $\mathcal{I}_{N_p}$ is predicted. This can be achieved by exploiting the correlation properties between all channel coefficients within the considered block. The power spectral density (PSD) of the process is given by [1], [9]: $S_h(f) = \sum_{p=0}^{P-1} |a_p|^2 \delta(f - f_p)$, and the covariance function $R_h[k]$ can be obtained by sampling the inverese Fourier transform of the PSD [9], [10], viz., $R_h[k] = \sum_{p=0}^{P-1} |a_p|^2 e^{j2\pi f_p T_s k}$ at $k = 0, 1, \ldots, M_o + N_p - 1$. If $P \to \infty$, the limit of the discrete covariance function $R_h[k]$ is equal to $J_0(2\pi k T_s f_c v/c)$ [1], [9], [11]. We now collect the channel coefficients $h[m]$ of the observation interval $\mathcal{I}_{M_o}$ in a vector $\mathbf{h} = [h[M_o - 1], h[M_o - 2], \ldots, h[1], h[0]]^T$. The corresponding covariance matrix $\mathbf{\Sigma_h}$ [1] is:

$$\mathbf{\Sigma_h} = \begin{bmatrix} R_h[0] & R_h[1] & \ldots & R_h[M_o-1] \\ R_h^*[1] & R_h[0] & \ldots & R_h[M_o-2] \\ \vdots & \vdots & \ddots & \vdots \\ R_h^*[M_o-2] & R_h^*[M_o-3] & \ldots & R_h[1] \\ R_h^*[M_o-1] & R_h^*[M_o-2] & \ldots & R_h[0] \end{bmatrix}. \quad (2)$$

A. LMMSE Predictor

At the BS, we do only have access to noisy observations of channel coefficients within the observation interval $\mathcal{I}_{M_o}$. These are collected in a vector

$$\mathbf{y} = \mathbf{h} + \mathbf{n}, \quad (3)$$

where the complex additive white Gaussian noise (AWGN) is described by $\mathbf{n} \sim \mathcal{N}_{\mathbb{C}}(\mathbf{0}, \mathbf{\Sigma_n} = \sigma_n^2 \mathbf{I}_{M_o})$. The covariance matrix of the noisy observations $\mathbf{y}$ is $\mathbf{\Sigma_y} = \mathbf{\Sigma_h} + \sigma_n^2 \mathbf{I}_{M_o}$. Channel coefficients of the prediction interval $\mathcal{I}_{N_p}$ can be obtained with the $l-$step LMMSE predictor [1], [12]:

$$\hat{h}[m] = \hat{h}_m = \mathbf{c}_{h_m\mathbf{y}}^H \mathbf{\Sigma_y}^{-1} \mathbf{y}, \quad (4)$$

with $m \in \mathcal{I}_{N_p}$ and $l = m - (M_o - 1)$ and the correlation vector $\mathbf{c}_{h_m\mathbf{y}}^H$ equal to

$$\mathbf{c}_{h_m\mathbf{y}}^H = [R_h[l], R_h[1+l], \ldots, R_h[M_o - 1 + l]]. \quad (5)$$

A reformulated version of the LMMSE predictor is derived in the following, with the ultimate goal to derive the so-called *Gridded Predictor*, the *Structured Predictor* and eventually the *Neural Network Predictor*. The derivations of these predictors can also be found in [5].

First, we fix a desired step length l and extend the vector of channel coefficients $\mathbf{h}$ of the observation interval $\mathcal{I}_{M_o}$ artificially by l channel coefficients of the prediction interval $\mathcal{I}_{N_p}$: $\mathbf{h}^{l-\text{ext}} = [h[m], h[m-1] \ldots, h[M_o], \mathbf{h}^T]^T$, with $m \in \mathcal{I}_{N_p}$ and $m = l + (M_o - 1)$. The extended covariance matrix $\mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}}$ is constructed analogous to the covariance matrix $\mathbf{\Sigma_h}$ from (2). The covariance matrix $\mathbf{\Sigma_h}$ is embedded in the bottom right part of $\mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}}$. The correlation vector $\mathbf{c}_{h_m\mathbf{y}}^H$ is identical to the zeroth row starting from the $l-$th column of the extended covariance matrix $\mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}}$. With the following definitions,

$$\mathbf{e}_1^T = [1, 0, \ldots, 0] \qquad (1 \times M_o + l) \quad (6)$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_{M_o} \end{bmatrix} \qquad (M_o + l \times M_o). \quad (7)$$

the correlation vector $\mathbf{c}_{h_m\mathbf{y}}^H$ and the covariance matrix $\mathbf{\Sigma_h}$ can be extracted from the extended covariance matrix $\mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}}$ with:

$$\mathbf{c}_{h_m\mathbf{y}}^H = \mathbf{e}_1^T \mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}} \mathbf{S} \quad \text{and} \quad \mathbf{\Sigma_h} = \mathbf{S}^T \mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}} \mathbf{S}. \quad (8)$$

The reformulated $l-$step LMMSE predictor is then [5], [13]:

$$\hat{h}_m = \mathbf{e}_1^T \mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}} \mathbf{S} (\mathbf{S}^T \mathbf{\Sigma}_{\mathbf{h}}^{l-\text{ext}} \mathbf{S} + \sigma_n^2 \mathbf{I}_{M_o})^{-1} \mathbf{y} \quad (9)$$

$$= \mathbf{e}_1^T \mathbf{W}^{l-\text{ext}} \mathbf{y}. \quad (10)$$

B. Gridded Predictor

Using Bayes' approach of [2] an approximated minimum mean squared error (MMSE) predictor is derived in the following. We assume to have a random variable $\boldsymbol{\delta}$, corresponding to the DoAs of a sampled scenario that determines the path-specific Doppler shifts, which are characterized by a prior $p(\boldsymbol{\delta})$. Accordingly, we require for each sample, that the closed-form solution $\mathbf{W}_{\boldsymbol{\delta}}$ of the LMMSE predictor according to $\mathbf{W}^{l-\text{ext}}$ as in (10) is available [2], [3], [14]:

$$\hat{\mathbf{W}}_{\text{MMSE}} = \int p(\boldsymbol{\delta}|\mathbf{y}) \mathbf{W}_{\boldsymbol{\delta}} d\boldsymbol{\delta}. \quad (11)$$

The estimated filter can be reformulated to:

$$\hat{\mathbf{W}}_{\text{MMSE}} = \frac{\int p(\mathbf{y}|\boldsymbol{\delta}) \mathbf{W}_{\boldsymbol{\delta}} p(\boldsymbol{\delta}) d\boldsymbol{\delta}}{\int p(\mathbf{y}|\boldsymbol{\delta}) p(\boldsymbol{\delta}) d\boldsymbol{\delta}}, \quad (12)$$

where, the likelihood $p(\mathbf{y}|\boldsymbol{\delta})$ is assumed to be Gaussian:[1]

$$p(\mathbf{y}|\boldsymbol{\delta}) \propto \frac{\exp\left(-\mathbf{y}^H \mathbf{\Sigma}_{\mathbf{y}_{\boldsymbol{\delta}}}^{-1} \mathbf{y}\right)}{|\mathbf{\Sigma}_{\mathbf{y}_{\boldsymbol{\delta}}}|}. \quad (13)$$

[1]The second order statistical moments are indexed with $\boldsymbol{\delta}$ in the following, to express the dependency on the selected sample of a specific realization.

$\mathbf{\Sigma}_{\mathbf{y}\boldsymbol{\delta}}^{-1}$ is re-expressed in terms of $\mathbf{W}_{\boldsymbol{\delta}}$. To this end, we identify $\mathbf{W}_{\boldsymbol{\delta}}$ with the predictor in (9) [13]:

$$\mathbf{\Sigma}_{\mathbf{h}_{\boldsymbol{\delta}}}^{l-\text{ext}}\mathbf{S}(\mathbf{S}^T\mathbf{\Sigma}_{\mathbf{h}_{\boldsymbol{\delta}}}^{l-\text{ext}}\mathbf{S}+\sigma_n^2\mathbf{I}_{M_o})^{-1}=\mathbf{W}_{\boldsymbol{\delta}} \tag{14}$$

$$\mathbf{\Sigma}_{\mathbf{h}_{\boldsymbol{\delta}}}^{l-\text{ext}}\mathbf{S}=\mathbf{W}_{\boldsymbol{\delta}}(\mathbf{S}^T\mathbf{\Sigma}_{\mathbf{h}_{\boldsymbol{\delta}}}^{l-\text{ext}}\mathbf{S}+\sigma_n^2\mathbf{I}_{M_o}) \tag{15}$$

$$\mathbf{I}_{M_o}=\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}+\sigma_n^2\mathbf{I}_{M_o}(\mathbf{S}^T\mathbf{\Sigma}_{\mathbf{h}_{\boldsymbol{\delta}}}^{l-\text{ext}}\mathbf{S}+\sigma_n^2\mathbf{I}_{M_o})^{-1} \tag{16}$$

$$\mathbf{\Sigma}_{\mathbf{y}_{\boldsymbol{\delta}}}^{-1}=\mathbf{\Sigma}_{\mathbf{n}}^{-1}(\mathbf{I}_{M_o}-\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}). \tag{17}$$

Incorporating this result into the expression for the likelihood yields:

$$p(\mathbf{y}|\boldsymbol{\delta})\propto\exp\left(\sigma_n^{-2}\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\mathbf{y}\mathbf{y}^H)\right)|\mathbf{I}_{M_o}-\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}|. \tag{18}$$

We further define $\hat{\mathbf{C}}=\sigma_n^{-2}\mathbf{y}\mathbf{y}^H$ and

$$b_{\boldsymbol{\delta}}=\log|\mathbf{I}_{M_o}-\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}|, \tag{19}$$

such that we can reformulate the likelihood $p(\mathbf{y}|\boldsymbol{\delta})$ as:

$$p(\mathbf{y}|\boldsymbol{\delta})\propto\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}}\right). \tag{20}$$

Inserting the above expression into (12) yields:

$$\hat{\mathbf{W}}_{\text{MMSE}}=\frac{\int\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}}\right)\mathbf{W}_{\boldsymbol{\delta}}p(\boldsymbol{\delta})d\boldsymbol{\delta}}{\int\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}}\right)p(\boldsymbol{\delta})d\boldsymbol{\delta}}. \tag{21}$$

The overall approximated MMSE predictor is (we still have to multiply with $\mathbf{e}_1^T$ from the left, see (9)):

$$\hat{\mathbf{w}}^T(\hat{\mathbf{C}})=\mathbf{e}_1^T\frac{\int\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}}\right)\mathbf{W}_{\boldsymbol{\delta}}p(\boldsymbol{\delta})d\boldsymbol{\delta}}{\int\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}}\right)p(\boldsymbol{\delta})d\boldsymbol{\delta}}. \tag{22}$$

For arbitrary priors $p(\boldsymbol{\delta})$ a closed form solution for this filter does not exist. Nevertheless, with following assumption it is possible to obtain a computable expression [2], [13].

Assumption 1: The prior $p(\boldsymbol{\delta})$ is discrete and uniform:

$$p(\boldsymbol{\delta}_i)=1/N,\forall i=1,\dots N. \tag{23}$$

We obtain the *Gridded Predictor* [13], by replacing the prior in (22) by $1/N$ and the integrals by sums:

$$\hat{\mathbf{w}}^T(\hat{\mathbf{C}})=\mathbf{e}_1^T\frac{(1/N)\sum_{i=1}^N\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}_i}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}_i}\right)\mathbf{W}_{\boldsymbol{\delta}_i}}{(1/N)\sum_{i=1}^N\exp\left(\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}_i}\hat{\mathbf{C}})+b_{\boldsymbol{\delta}_i}\right)}, \tag{24}$$

where each sample specific filter $\mathbf{W}_{\boldsymbol{\delta}_i}$ is calculated according to (9) and $b_{\boldsymbol{\delta}_i}$ is evaluated by (19). The Gridded Predictor allows to predict channel coefficients, without any knowledge on the true PSD of a specific scenario. With an increasing number of samples N the approximation error decreases. Nevertheless, there will be a gap compared to the LMMSE predictor with perfect knowledge of the statistical moments of second order based on the coveriance function $R_h[k]$, because of a finite N. This gap is even more significant if the $\boldsymbol{\delta}_i$ are sampled from a prior $p(\boldsymbol{\delta})$, with more than one propagation path. In such a case, many combinations of DoAs are possible, which can not be fully captured by the Gridded Predictor with a finite number of samples N.

C. Structured Predictor

The drawbacks of the Gridded Predictor are the numerical complexity and a large memory requirement, due to the storage of a filter for each sample $\mathbf{W}_{\boldsymbol{\delta}_i}$. With the following assumption, it is further possible to simplify the predictor and to reduce the memory overhead [13]:

Assumption 2: $\forall i=1,\dots,N$ the filters $\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}_i}$ can be decomposed as:

$$\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}_i}=\mathbf{Q}^H\text{diag}(\mathbf{w}_{\boldsymbol{\delta}_i})\mathbf{Q}, \tag{25}$$

with $\mathbf{w}_{\boldsymbol{\delta}_i}\in\mathbb{R}^K$ and a common matrix $\mathbf{Q}\in\mathbb{C}^{K\times M}$.

Instead of storing a matrix for each sample $\mathbf{W}_{\boldsymbol{\delta}_i}$, it is now sufficient to store a vector $\mathbf{w}_{\boldsymbol{\delta}_i}$ for each sample, which reduces the memory overhead. Similar as in [2], possible candidates for $\mathbf{Q}$ are either the $M_o\times M_o$ DFT matrix, $\mathbf{Q}=\mathbf{F}_1\in\mathbb{C}^{M_o\times M_o}$ (Circulant approx.) or the first M_o columns of the $2M_o\times 2M_o$ DFT matrix, $\mathbf{Q}=\mathbf{F}_2\in\mathbb{C}^{2M_o\times M_o}$ (Toeplitz approx.). By defining:

$$\hat{\mathbf{c}}=\sigma_n^{-2}|\mathbf{Q}\mathbf{y}|^2, \tag{26}$$

and using (25), the trace expressions in (24) are reformulated:

$$\begin{aligned}\text{tr}(\mathbf{S}^T\mathbf{W}_{\boldsymbol{\delta}_i}\hat{\mathbf{C}})&=\text{tr}(\mathbf{Q}^H\text{diag}(\mathbf{w}_{\boldsymbol{\delta}_i})\mathbf{Q}\sigma_n^{-2}\mathbf{y}\mathbf{y}^H) &(27)\\ &=\text{tr}(\text{diag}(\mathbf{w}_{\boldsymbol{\delta}_i})\sigma_n^{-2}\mathbf{Q}\mathbf{y}\mathbf{y}^H\mathbf{Q}^H) &(28)\\ &=\mathbf{w}_{\boldsymbol{\delta}_i}^T\hat{\mathbf{c}}, &(29)\end{aligned}$$

since $\hat{\mathbf{c}}$ contains the diagonal entries of $\sigma_n^{-2}\mathbf{Q}\mathbf{y}\mathbf{y}^H\mathbf{Q}^H$. Inserting this result into (24), simplifies the Gridded Predictor:

$$\hat{\mathbf{w}}^T(\hat{\mathbf{c}})=\frac{\sum_{i=1}^N\exp\left(\mathbf{w}_{\boldsymbol{\delta}_i}^T\hat{\mathbf{c}}+b_{\boldsymbol{\delta}_i}\right)\mathbf{e}_1^T\mathbf{W}_{\boldsymbol{\delta}_i}}{\sum_{i=1}^N\exp\left(\mathbf{w}_{\boldsymbol{\delta}_i}^T\hat{\mathbf{c}}+b_{\boldsymbol{\delta}_i}\right)}. \tag{30}$$

By further collecting the sample specific vectors and biases in matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ and the vector $\mathbf{b}$:

$$\mathbf{A}_1=\begin{bmatrix}\mathbf{w}_{\boldsymbol{\delta}_1}^T\\ \vdots\\ \mathbf{w}_{\boldsymbol{\delta}_N}^T\end{bmatrix}\quad\mathbf{A}_2=\begin{bmatrix}\mathbf{e}_1^T\mathbf{W}_{\boldsymbol{\delta}_1}\\ \vdots\\ \mathbf{e}_1^T\mathbf{W}_{\boldsymbol{\delta}_N}\end{bmatrix}^T\quad\mathbf{b}=\begin{bmatrix}b_{\boldsymbol{\delta}_1}\\ \vdots\\ b_{\boldsymbol{\delta}_N}\end{bmatrix}, \tag{31}$$

we end up with the *Structured Predictor* [13]:

$$\hat{\mathbf{w}}(\hat{\mathbf{c}})=\mathbf{A}_2\frac{\exp\left(\mathbf{A}_1\hat{\mathbf{c}}+\mathbf{b}\right)}{\mathbf{1}^T\exp\left(\mathbf{A}_1\hat{\mathbf{c}}+\mathbf{b}\right)}, \tag{32}$$

D. Neural Network Predictor

An expert observation yields that a feed-forward neural network with one hidden layer and the softmax activation function, has the same structure as the Structured Predictor. Therefore, we define a neural network which is depicted in Fig. 2 as [13]:

$$\hat{\mathbf{w}}_{\text{NN}}(\hat{\mathbf{c}})=\mathbf{A}_{(2)}\frac{\exp\left(\mathbf{A}_{(1)}\hat{\mathbf{c}}+\mathbf{b}_{(1)}\right)}{\mathbf{1}^T\exp\left(\mathbf{A}_{(1)}\hat{\mathbf{c}}+\mathbf{b}_{(1)}\right)}+\mathbf{b}_{(2)}. \tag{33}$$

The matrix $\mathbf{A}_1$ of the Structured Predictor from (32), which comprises the sample specific filter vectors $\mathbf{w}_{\boldsymbol{\delta}_i}\in\mathbb{R}^K$, equals to the weight matrix $\mathbf{A}_{(1)}$ of the first layer of the neural network: $\mathbf{A}_{(1)}=\mathbf{A}_1$. The vector $\mathbf{b}$ is the bias vector of the first layer, thus $\mathbf{b}_{(1)}=\mathbf{b}$. If we carefully consider (31), we can

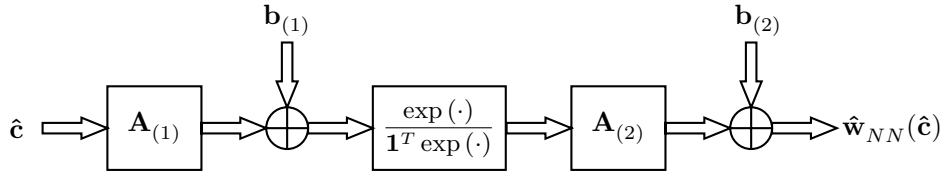

Fig. 2. Feed-forward neural network with one hidden layer and softmax activation function.

see that the entries of the second matrix $\mathbf{A}_2$ consist of sample specific filter vectors $\mathbf{e}_1^T \mathbf{W}_{\boldsymbol{\delta}_i} \in \mathbb{C}^{1\times M}$. Thus, the matrix $\mathbf{A}_2$ is complex. We split the matrix $\mathbf{A}_2$ into its real and imaginary part and define

$$\mathbf{A}_{(2)} = \begin{bmatrix} \Re(\mathbf{A}_2) \\ \Im(\mathbf{A}_2) \end{bmatrix}. \tag{34}$$

We further define a bias term for the second layer and the Structured Predictor suggests: $\mathbf{b}_{(2)} = \mathbf{0}$. Accordingly, the output of the neural network $\hat{\mathbf{w}}_{\text{NN}}(\hat{\mathbf{c}})$ is the concatenation of the real and imaginary parts of the Structured Predictor $\hat{\mathbf{w}}(\hat{\mathbf{c}})$:

$$\hat{\mathbf{w}}_{\text{NN}}(\hat{\mathbf{c}}) = \begin{bmatrix} \hat{\mathbf{w}}_{\text{NN},\Re}(\hat{\mathbf{c}}) \\ \hat{\mathbf{w}}_{\text{NN},\Im}(\hat{\mathbf{c}}) \end{bmatrix} = \begin{bmatrix} \Re(\hat{\mathbf{w}}(\hat{\mathbf{c}})) \\ \Im(\hat{\mathbf{w}}(\hat{\mathbf{c}})) \end{bmatrix}. \tag{35}$$

In this way, the Structured predictor can serve as a blueprint predictor for specific system models, by initializing the weights $\mathbf{A}_{(1)}$ and $\mathbf{A}_{(2)}$ and biases $\mathbf{b}_{(1)}$ and $\mathbf{b}_{(2)}$ of the Neural Network Predictor with the parameters of the Structured Predictor as explained above. By further training the Neural Network Predictor offline, we wish to achieve a better performance as compared to the predictors described above. Therefore, a predefined number of mini-batches are generated. A mini-batch consists of B channel realizations $\mathbf{h}_{b,\mathcal{I}_{M_o}}$, where each comprises M_o channel coefficients of the observation interval $\mathcal{I}_{M_o}$, and corresponding channel coefficients $h_{b,\mathcal{I}_{N_p}}$ of the prediction interval $\mathcal{I}_{N_p}$ (for the desired step l), with $b = 1, 2, \ldots, B$. For each channel realization $\mathbf{h}_{b,\mathcal{I}_{M_o}}$ a noisy version $\mathbf{y}_{b,\mathcal{I}_{M_o}}$ is generated by adding complex AWGN with known variance σ_n^2. According to (26), the input of the neural network $\hat{\mathbf{c}}_b$ for each $\mathbf{y}_{b,\mathcal{I}_{M_o}}$ can be evaluated depending on $\mathbf{Q}$ (Toeplitz or Circular). For each input $\hat{\mathbf{c}}_b$ a specific filter $\hat{\mathbf{w}}_{NN}(\hat{\mathbf{c}}_b)$ is present at the output, which can be further processed to obtain an estimate $\hat{h}_{b,\mathcal{I}_N}$ by calculating:

$$\hat{h}_{b,\mathcal{I}_N} = [\hat{\mathbf{w}}_{NN,\Re}(\hat{\mathbf{c}}_b) + j\hat{\mathbf{w}}_{NN,\Im}(\hat{\mathbf{c}}_b)]^T \mathbf{y}_{b,\mathcal{I}_M} \tag{36}$$

As performance metric (cost function), we choose the mean squared error (MSE). The stochastic gradient is then:

$$\mathbf{g} = \frac{1}{B}\sum_{b=1}^{B} \frac{\partial}{\partial[\mathbf{A}_{(i)}; \mathbf{b}_{(i)}]} \|h_{b,\mathcal{I}_N} - \hat{h}_{b,\mathcal{I}_N}\|_2^2, \tag{37}$$

with $i = 1, 2$. Then, the variables of the neural network are updated with a desired gradient algorithm (e.g., [15]). The described procedure is repeated until a convergence criterion is fulfilled.

The learning procedure is summarized in the following [13]:

Algorithm 1 Learning the MMSE Channel Predictor

1: Init. the Neural Network with the Structured Predictor
2: Generate a mini-batch of in total B channel realizations, of the observation interval $\mathbf{h}_{b,\mathcal{I}_{M_o}}$ and corresponding channel coefficients of the prediction interval (of desired prediction step l) $h_{b,\mathcal{I}_{N_p}}$, for $b = 1, 2, \ldots, B$.
3: Generate noisy version $\mathbf{y}_{b,\mathcal{I}_{M_o}}$ of $\mathbf{h}_{b,\mathcal{I}_{M_o}}$ and calculate $\hat{\mathbf{c}}_b$ (input of the neural network), for $b = 1, 2, \ldots, B$.
4: Calculate the stochastic gradient ($i = 1, 2$):

$$\mathbf{g} = \frac{1}{B}\sum_{b=1}^{B} \frac{\partial}{\partial[\mathbf{A}_{(i)}; \mathbf{b}_{(i)}]} \|h_{b,\mathcal{I}_{N_p}} - \hat{h}_{b,\mathcal{I}_{N_p}}\|_2^2,$$

5: Update the variables of the neural network with a desired gradient algorithm (e.g., [15])
6: Repeat steps 2-5 until a convergence criterion is fulfilled.

III. Simulation Results

In this Section, we discuss the performances of the previously described predictors. As baseline we use the LMMSE predictor with perfect knowledge of the statistical moments of second order based on the coveriance function $R_h[k]$ (i.e., the DoAs are known) and denote it as *LMMSE Perfect*. The LMMSE predictor with the assumption of $P \to \infty$, is denoted as *LMMSE Jakes*, Clearly, assuming infinitely many paths is not true for specific cases with a finite number of paths. Nevertheless, constructing the LMMSE predictor with the assumption of having infinitely many paths is straightforward, since in this case the covariance function is equal to the zeroth order Bessel function. The Gridded Predictor is simply denoted as *Gridded*, and the Structured Predictor as *Structured Toep* (Toepltiz approx.) or as *Structured Circ* (circulant approx.). The Neural Network Predictor is denoted as *NN Toep* or as *NN Circ*. Table I summarizes all considered predictors.

TABLE I
Considered Predictors in the Simulations

LMMSE Perfect	with perfect knowledge of the statistical moments of second order based on $R_h[k]$
LMMSE Jakes	LMMSE predictor with assumption $P \to \infty$
Gridded	Gridded Predictor
Structured Toep	Structured Predictor with $\mathbf{Q} = \mathbf{F}_2$
Structured Circ	Structured Predictor with $\mathbf{Q} = \mathbf{F}_1$
NN Toep	Neural Network Predictor with $\mathbf{Q} = \mathbf{F}_2$
NN Circ	Neural Network Predictor with $\mathbf{Q} = \mathbf{F}_1$

For all simulations in the following the symbol duration $T_s = 20.57\,\mu\text{s}$ and the carrier frequency $f_c = 2\,\text{GHz}$ as in [1].

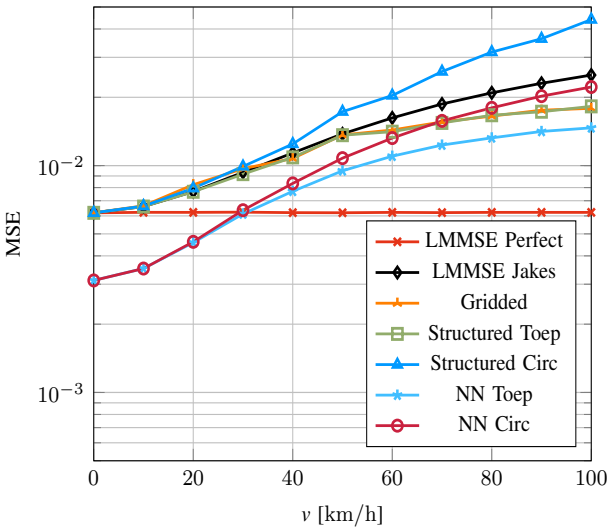

Fig. 3. MSE at prediction step $l = 4$, $M_o = 16$, SNR 10 dB, $P = 1$, $f_c = 2$ GHz, $T_s = 20.57$ µs, $N = 16$ or $N = 32$

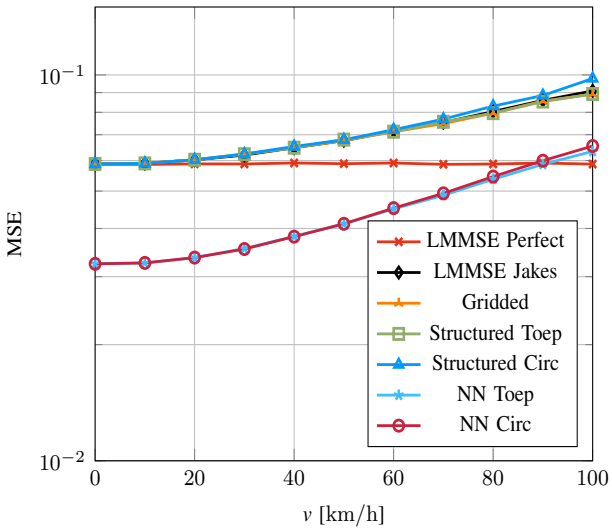

Fig. 4. MSE at prediction step $l = 4$, $M_o = 16$, SNR 0 dB, $P = 1$, $f_c = 2$GHz, $T_s = 20.57$ µs, $N = 16$ or $N = 32$

For the construction of the Gridded Predictor, the Structured Predictors and the Neural Network Predictors, a fixed number of samples is needed, which is predefined depending on the number of observed symbols M_o, in order to achieve easier interpretation in terms of computational complexity. For the cases, where $\mathbf{Q} = \mathbf{F}_2$ (Toeplitz assumption) the number of samples N is doubled (the input of the Structured Predictor and the Neural Network Predictor with $\mathbf{Q} = \mathbf{F}_2$ is twice as long as for the case $\mathbf{Q} = \mathbf{F}_1$). All of the predictors are specifically constructed for each simulated velocity, i.e., we have to construct and train the Neural Network Predictors for each velocity separately. The performances of the predictors are evaluated by calculating the MSE of 200.000 predictions.

In the first simulation (Fig. 3), the number of observed symbols $M_o = 16$ and the prediction step $l = 4$. The SNR is 10 dB and the number of impinging plane-waves at the user is one, i.e., $P = 1$. Thus, we have one randomly generated DoA for each channel realization, which remains constant over the considered block. The number of samples for the construction of the predictors is set to $N = 16$ or $N = 32$ (depending on $\mathbf{Q}$). The MSE of the LMMSE Perfect predictor remains constant for all velocities. The LMMSE Perfect predictor outperforms the LMMSE Jakes predictor for all velocities, since the LMMSE Perfect predictor has perfect knowledge of the spectrum, whereas the LMMSE Jakes predictor assumes $P \to \infty$. As compared to the LMMSE Jakes predictor, the Structured Circ predictor performs worse, whereas the Gridded and Structured Toep predictor outperform the LMMSE Jakes predictor for velocities higher than 50 km/h (Fig. 3). As explained above the NN Toep is initialized with the Structured Toep predictor and the NN Circ predictor is initialized with the Structured Circ predictor. For training the Neural Network Predictors 3000 mini-batches, each of size $B = 50$, were used. After training the Neural Network Predictors, both of them outperform the LMMSE Jakes predictor for all considered velocities. The NN Circ predictor has a lower MSE as compared to the Gridded Predictor for velocities smaller than 70 km/h (Fig. 3). The NN Toep predictor outperforms the Gridded predictor for all velocities. We can conclude that the the Neural Network Predictors are able to compensate the approximation error of Assumption 1 with a finite number of samples N. Already the differences in the MSEs for the Structured Toep and Structured Circ predictors suggest that the Toeplitz assumption ($\mathbf{Q} = \mathbf{F}_2$) is a better approximation. In addition to that, the neural network size with Toeplitz assumption is twice as large as compared to the Circular case. This may also explain the gap between the two Neural Network Predictor performances. For the velocity range from 0 km/h to 30 km/h (Fig. 3) the Neural Network Predictors even outperform the LMMSE Perfect predictor based on the knowledge of the covariance function. This is the consequence of the assumed channel model, i.e., channel coefficients, constructed with a low number of paths (propagation channel models with specular geometry), which are not Gaussian distributed render the LMMSE predictor to be not optimal and therefore to be outperformed by other approaches that take into account the actual underlying distribution of channel coefficients or their respective samples.

In the next simulation setting (Fig. 4), the number of observed symbols remains $M_o = 16$ and the prediction step remains as well unchanged, $l = 4$. The SNR is now 0 dB and the number of impinging plane-waves at the user is again one, i.e., $P = 1$. The number of samples for the construction of the predictors is set to $N = 16$ or $N = 32$ (depending on $\mathbf{Q}$). As in the previous simulation, the MSE of the LMMSE Perfect predictor remains constant for all velocities. Obviously, the LMMSE Perfect predictor again outperforms the LMMSE Jakes predictor for all velocities. The Structured Circ predictor, the Gridded predictor, the Structured Toep predictor and the LMMSE Jakes predictor perform equally well. The Neural

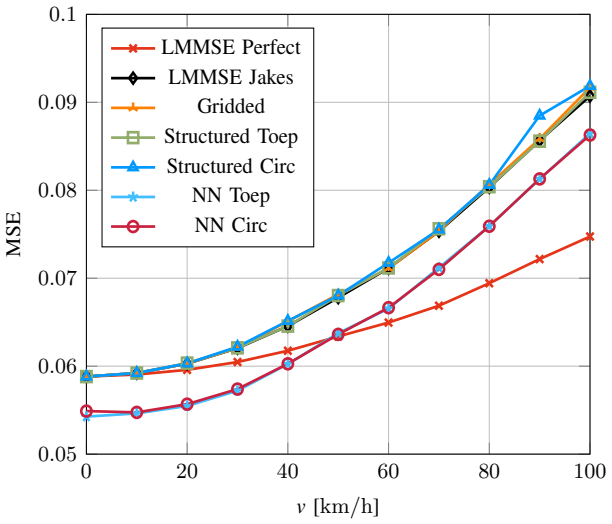

Fig. 5. MSE at prediction step $l = 4$, $M_o = 16$, SNR 0 dB, $P = 2$, $f_c = 2\,\text{GHz}$, $T_s = 20.57\,\mu\text{s}$, $N = 16$ or $N = 32$

Network Predictors are again initialized as described above and for training them again 3000 mini-batches, each of size $B = 50$, were used. After training the Neural Network Predictors, both of them outperform the LMMSE Jakes predictor and the Gridded predictor for all considered velocities. The NN Toep predictor has a slightly lower MSE as compared to the NN Circ predictor for velocities larger than $60\,\text{km/h}$ (Fig. 4). Both of the Neural Network Predictors outperform the LMMSE Perfect predictor for the velocity range from $0\,\text{km/h}$ to $90\,\text{km/h}$ (Fig. 4).

So far we have only considered the case of having only one impinging plane-wave at the user. In the next simulation setting (Fig. 5), we now increase the number of impinging plane-waves at the user to two, i.e., $P = 2$. The number of observed symbols remains $M_o = 16$ and the prediction step remains as well unchanged, $l = 4$. The SNR is 0 dB. The number of samples for the construction of the predictors is still set to $N = 16$ or $N = 32$ (depending on $\mathbf{Q}$). In contrast to the previous simulations, the MSE of the LMMSE Perfect predictor does not remain constant for all velocities due to the increased number of paths to two. The LMMSE Jakes predictor with $P \to \infty$ suggests that the predictor performance heavily depends on the Doppler bandwidth B_D of the system setup, which depends on the velocity of the user. The Gridded predictor, the Structured Circ predictor, the Structured Toep predictor and the LMMSE Jakes predictor perform again almost equally well. We train the neural networks with 3000 mini-batches, each of size $B = 50$, where each channel realization is constructed with $P = 2$, i.e., two DoAs are randomly generated for each channel realization and remain constant over the considered block. In this simulation setting, the trained Neural Network Predictors outperform all other predictors for the velocity range from $0\,\text{km/h}$ to $50\,\text{km/h}$ (Fig. 5).

IV. Conclusion

A feed-forward neural network channel predictor, which is trained offline, was presented in this paper. We started with the general expression of an approximated MMSE predictor and derived a predictor having the structure of a feed-forward neural network by making two key assumptions. By using the Structured Predictor as a blueprint an by further training the Neural Network Predictor, it was possible to compensate the approximation errors due to the assumptions. The Neural Network Predictor outperformed the LMMSE Perfect predictor for the specific channel models with low path numbers, especailly for low SNR values. We considered simulation settings with low path numbers, since with a relatively high number of propagation paths the channel is similar to Jakes model. In [5] the performance of the Neural Network Predictor was evaluated by using an indoor LOS scenario of the DeepMIMO dataset. The Structured Predictor which served as a blueprint predictor was constructed by creating scenarios with $P = 1$ as above. However, the dynamic construction procedure from above can be also used for other scenarios, where there is no dominating LOS component, e.g., in scenarios with two dominant paths.

References

[1] T. Zemen, C. F. Mecklenbrauker, F. Kaltenberger, and B. H. Fleury, "Minimum-Energy Band-Limited Predictor with Dynamic Subspace Selection for Time-Variant Flat-Fading Channels," *IEEE Trans. on Signal Process.*, vol. 55, pp. 4534–4548, 2007.

[2] D. Neumann, T. Wiese, and W. Utschick, "Learning the MMSE Channel Estimator," *IEEE Trans. on Signal Process.*, vol. 66, pp. 2905–2917, 2018.

[3] C. Hellings, A. Dehmani, S. Wesemann, M. Koller, and W. Utschick, "Evaluation of Neural-Network-Based Channel Estimators Using Measurement Data," in *WSA 2019; 23rd Int. ITG Workshop on Smart Antennas*, April 2019, pp. 1–5.

[4] 3GPP, "Spatial channel model for multiple input multiple output (MIMO) simulations (release 12)," 3rd Generation Partnership Project (3GPP), TR 25.996 V12.0.0, 2014.

[5] N. Turan and W. Utschick, "Reproducible Evaluation of Neural Network based Channel Estimators and Predictors Using a Generic Dataset," in *WSA 2020; 24th Intern. ITG Workshop on Smart Antennas*, February 2020, pp. 1–6, accepted, arXiv preprint available: 1912.00005.

[6] A. Alkhateeb, "DeepMIMO: A Generic Deep Learning Dataset for Millimeter Wave and Massive MIMO Applications," in *Proc. of Inf. Theory and Appl. Workshop (ITA)*, San Diego, CA, Feb 2019, pp. 1–8.

[7] R. H. Clarke, "A statistical theory of mobile-radio reception," *Bell Syst. Tech. J.*, vol. 47, pp. 957–1000, 1968.

[8] T. Zemen, C. F. Mecklenbrauker, and B. H. Fleury, "Time-Variant Channel Prediction using Time-Concentrated and Band-Limited Sequences," in *2006 IEEE Int. Conf. on Commun.*, vol. 12, June 2006, pp. 5660–5665.

[9] A. Goldsmith, *Wireless Communications*. Cambridge Univ. Press, 2005.

[10] R. Bracewell, *The Fourier Transform and Its Appl.* New York: McGraw Hill, 2000.

[11] W. Jakes, *Microwave Mobile Communications*. Wiley, 1974.

[12] S. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.

[13] N. Turan and W. Utschick, "Learning The MMSE Channel Predictor," arXiv 1911.07256, 2019.

[14] M. Koller, C. Hellings, and W. Utschick, "Learning-Based Channel Estimation for Various Antenna Array Configurations," in *2019 IEEE 20th Int. Workshop on Signal Process. Advances in Wireless Commun. (SPAWC)*, July 2019, pp. 1–5.

[15] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *Int. Conf. on Learn. Representations*, 12 2014.