ESTIMATION OF SIGNAL PARAMETERS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

S. Schieler*, S. Semper*, R. Faramarzahangari*, M. Döbereiner†, C. Schneider†

*Technische Universität Ilmenau
†Fraunhofer Institute of Integrated Circuits

ABSTRACT

This paper introduces a Deep Learning approach for signal parameter estimation in the context of wireless channel modeling. Our work is capable of multidimensional parameter estimation from a signal containing an unknown number of paths. The signal parameters are estimated relative to a predefined grid, providing quasi grid-free, hence, more accurate estimates than previous grid-limited approaches. It requires no prior knowledge of the number of paths, giving it an advantage in terms of complexity compared to existing solutions. Along with the description, we provide an initial performance analysis and a comparison with State-of-the-Art techniques and discuss future research directions.

Index Terms— Parameter Estimation, Convolutional Neural Networks, Delay-Doppler Estimation, Harmonic Retrieval

1. INTRODUCTION

Estimating radio signal propagation parameters is a problem encountered in many signal processing tasks, e.g., channel estimation, radar localization, and direction finding. Until recently, available solutions for the task could be loosely divided into three groups: subspace algorithms, iterative maximum likelihood (ML), and Sparse Signal Recovery (SSR). Recent publications [1–6] added a fourth category: Deep Neural Networks (DNNs)-based algorithms.

1.1. State-of-the-Art

In [1] a Convolutional Neural Network (CNN) is used to estimate frequencies by predicting a super-resolution pseudo-spectrum from a superposition of up to ten, complex-valued sinusoids. A second network predicts the number of sinusoids. The results show a performance improvement when compared to Multiple Signal Classification (MUSIC), especially in the low-Signal-to-Noise Ratio (SNR) domain. In [2] a CNN is trained to perform Direction of Arrival (DoA) estimation of up to three unknown sources on a grid. Similarly, the authors of [3] also tackle the problem of DoA estimation by combining a denoising autoencoder with a DNN. Both show performance improvement in the low-SNR domain compared to MUSIC. In [4] a DNN is combined with gradient steps on the likelihood function to perform DoA estimation. The results highlight the decrease in complexity and improvements in performance when both approaches are paired.

Except for [6], all previous works use a grid-based representation of the signal parameters, which limits the achievable accuracy to the grid resolution. As [2] notes, to obtain better estimates from the DNNs directly, a better grid resolution is required. However, the higher memory footprint associated with grid scaling inevitably limits usability, particularly in multi-dimensional tasks.

1.2. Our contributions

Compared to previous works, our approach uses a CNN to estimate signal parameters relative to a grid. This relative representation decouples the achievable accuracy from the grid size, allowing our approach to reach super-resolution. We show, by simulation, that our CNN architecture can directly estimate the signal parameters and the model order (number of paths). Due to the CNN architecture, the number of weights does not scale exponentially with the input dimensionality compared to fully-connected approaches, as in [6]. Therefore, we demonstrate a 2D delay and Doppler-shift estimation task encountered in wireless channel estimation and radar localization. According to our literature study, this is the first demonstration of multi-dimensional signal parameter estimation with a DNN. Furthermore, we introduce a multi-window preprocessing with the Discrete Fourier Transform (DFT) to provide more features for the learning process.

2. SIGNAL MODEL

Our task is estimating specular paths by their signal parameters, namely their propagation delays \( \tau \) and Doppler-shifts \( \alpha \), from a wireless channel measurement. We use the signal model introduced in this section to obtain training- and label data for our approach. Please note that our task is similar to the DoA estimation tackled by the previous works [1–6]

We model the wireless channel transfer-function measurement of bandwidth \( B \) with \( N_f \in \mathbb{N} \) frequency samples and \( N_t \in \mathbb{N} \) snapshots based on the narrowband assumption \( B \ll f_0 \). We denote the sampled observation in complex baseband \( (f_c = 0) \) by \( S \). The sampling process is characterized by the sampling intervals in frequency \( \Delta f > 0 \) and time \( \Delta t > 0 \) with \( S \) sampled \( N_f, N_t \in \mathbb{N} \) times at

\[
\begin{align*}
  f_k &= f_0 + k \cdot \Delta f \\
  t_l &= t_0 + l \cdot \Delta t,
\end{align*}
\]

where \( k = 0, ..., N_f - 1 \), \( l = 0, ..., N_t - 1 \), \( f_0 = -\frac{B}{2} \), and \( t_0 = 0 \). Therefore, the discrete signal model \( S \in \mathbb{C}^{N_f \times N_t} \) is formulated as

\[
S_{k,l}(\gamma, \tau, \alpha) = \sum_{p=1}^{P} \gamma_p \exp(-2j\pi f_k \tau_p) \exp(2j\pi t_l \alpha_p),
\] (1)
where the index $p = 1, \ldots, P$ denotes the path index and $\gamma \in \mathbb{C}^P$, $\tau \in \mathbb{R}^P$, $\alpha \in \mathbb{R}^P$ contain the corresponding complex path weights, delays and Doppler-shifts, respectively. The noisy observation $\eta \in \mathbb{C}^{\tau \times \alpha \times N}$ is then formulated as

$$Y = S(\gamma, \tau, \alpha) + N,$$

where $N \in \mathbb{C}^{N \times N}$ is a complex, zero-mean Gaussian noise vector.

The goal of the presented approach is to use a neural network to estimate $\tau$ and $\alpha$ from the sampled observations $\gamma$ and $\beta$. We neglect the direct estimation of the amplitudes in $\gamma$, as they can be efficiently reconstructed from the estimates and the observation eq. (2), e.g., via Least Squares (LS).

### 3. NEURAL NETWORK

#### 3.1. Preprocessing

Prior to the CNN, the data $Y$ is preprocessed using a multi-window 2D-DFT approach. As shown in [I], self-learned transformations are akin to the DFT. Hence, preprocessing with the DFT reduces the number of trainable parameters in the network while efficiently transforming the data to the target observation domain. The motivation for the multi-window approach is to obtain different data views. Rectangular windows, for example, achieve the maximum SNR and provide a narrow pulse shape but also result in high sidelobes after the DFT, potentially causing masking of close-by paths. Narrower windows, such as Hann-windows, reduce the sidelobes but increase the pulse width and degrade the SNR, reducing the detection probability of power-weak paths. Hence, the choice for a window function is usually application specific and bears the presented trade-offs. However, as a CNN can process multiple views of the same data in parallel (i.e., by interpreting them as color channels of an image), the diverse views can be exploited for more robust detection and estimation. We used $N_W = 8$ different windowing functions, i.e., a Tukey, Taylor, Chebyshev, Blackman, Flat Top, Cosine, Hann, and Rectangular window. The choice was made to ensure diverse data views (i.e., combining narrow and wide windows). We note that a more analytical study of the impact is required in future research.

The result after the 2D-DFT is denoted as $V \in \mathbb{C}^{NW \times N \times N}$, where $N_W$ denotes the number of windows. To obtain real-valued numbers for the training, we employ four mapping functions

$$
\begin{align*}
    f_1(Y_i) &= \text{Re}(Y_i) \\
    f_2(Y_i) &= \text{Im}(Y_i) \\
    f_3(Y_i) &= \log(|Y_i|) \\
    f_4(Y_i) &= \angle(Y_i)
\end{align*}
$$

to map the complex $Y_i$ to the CNN input data $Y_2 \in \mathbb{R}^{NW \times N \times N}$. Though the first two provide the same information as the latter, our experiments revealed that the additional diversity helped the learning.

#### 3.2. Off-Grid Parameter Encoding

The parameters are encoded relative to predefined grid regions to preserve their spatial relation and enable off-grid estimates with the CNN. Our encoding has similarity to [7] but is modified to suit task-specific needs. The encoding consists of three steps: parameter normalization, cell assignment, and relative parameter encoding. The normalization maps the parameters into a range between 0 and 1, such that $\tau$ and $\alpha$ are in the interval of $[0, 1]$ and $[0, 1]$, respectively. Next, we define a regular 2D grid of size $J \in \mathbb{N} \times J \in \mathbb{N}$ and define cells $\eta_{ij}$, which each span non-overlapping rectangular regions on the grid. Paths are mapped to the cells based on the shortest Euclidean distance to the cell centroids. Each cell is represented by a vector with three elements per path such that

$$\eta_{ij} = 
\begin{bmatrix}
    \mu_{[i,j]} \quad \Delta \tau_{[i,j]} \quad \Delta \alpha_{[i,j]} \quad \ldots \quad \mu_{[i,j]} \quad \Delta \tau_{[i,j]} \quad \Delta \alpha_{[i,j]}
\end{bmatrix}^T.
$$

![Fig. 1](image-url) The architecture of our CNN uses convolutional layers to perform upscaling, downscaling and downscaling. The encoded parameters in $\eta$ and model order $\rho$ (see Section 3.2 and Section 3.3, respectively) are estimated from dense layers to the downsampled or downsampled result.
\[ \Delta \tau_c \text{ and } \Delta \alpha_c \] denote the Euclidean distance between the cell centroid \( \eta_{i,j} \) and the true parameter. In line with best-practices for training, \( \Delta \tau_c \) and \( \Delta \alpha_c \) are normalized by the cell width and offsetted, such that \( \Delta \tau_c, \Delta \alpha_c \subset [0, 1) \). As the number of paths in each cell can vary, \( \mu^{[i,j]} = \{0, 1\} \) indicates if the path with parameters \( \Delta \tau_{c[i,j]} \) and \( \Delta \alpha_{c[i,j]} \) is assigned (1) or not (0). This enables computing a weighted loss during training (detailed in Section 3.4). For a learnable ordering of paths in each cell, the paths are assigned from \( c = 1 \ldots C \) in descending magnitude order, and unassigned parameters are labeled as 0 (see Figure 2).

The result of the encoding is a 3D array \( \eta \in \mathbb{R}^{I \times J \times C} \), which is the prediction task for the third stage of our CNN.

### 3.3. Network Architecture

Our network architecture is split into four stages represented in Figure 1. The first stage passes the input through 5 blocks of 2D convolutional layers. Each block consists of a 2D convolutional layer, followed by Batch-Normalization and a ReLU activation function. The convolutional layers are parameterized to preserve the data shape but double the number of channels after each block.

The second stage performs downsampling via convolutional layers with a stride of 2, which reduces the data shape by 2 with each block. In this stage, the number of channels is preserved.

Stage three achieves the parameter predictions. First, the number of channels is reduced by two blocks of convolutional layers, followed by two fully-connected (FC) layers interleaved by a ReLU activation function. The relative parameter estimates \( \eta \) (more details in Section 3.2) are encoded in the output of the final FC layer.

The fourth stage is used to predict the number of paths \( P \) (model order) based on the results of the second stage. It uses a single convolutional block followed by two FC layers interleaved by a ReLU activation function. Its output is \( \rho \), a one-hot encoded vector of the model order estimate \( P \).

### 3.4. Loss Functions and Training

Our approach uses multiple loss functions combined in a weighted sum. The first part is the loss for the model order estimate \( \rho \). It uses the well-known Binary Crossentropy (BCE) loss for the one-hot encoded values.

\[
L_0 = \hat{\rho} \cdot \log(\rho) + (1 - \hat{\rho}) \cdot \log(1 - \rho)
\]  

(4)

The loss for the parameter estimates \( \eta \) utilizes a masked squared error (MSE) loss function

\[
L_1 = \sum_{i,j=1}^{I,J} \sum_{c=1}^{C} \left(\sigma(\hat{\mu}^{[i,j]}_{c}) \cdot \left[ \frac{\Delta \tau_{c[i,j]}}{\Delta \alpha_{c[i,j]}} \right] - \left[ \frac{\Delta \tau_{c[i,j]}}{\Delta \alpha_{c[i,j]}} \right] \right)^2
\]  

(5)

where \( \sigma(\cdot) \) represents the sigmoid function, and \( \cdot \) marks the predictions. The sum over the two-element vector inside the \( \cdot \) is omitted for readability. As mentioned earlier, \( \mu^{[i,j]} \) is used to weight the parameter estimates \( \Delta \tau^{[i,j]} \) and \( \Delta \alpha^{[i,j]} \). The weighting is required to account for a varying number of paths in each cell vector. Its effect becomes apparent by inspecting the limits of the \( \sigma(\cdot) \) function, as

1. \( \lim_{x \to -\infty} \sigma(x) = 1 \), causing the MSE of the corresponding predictions to contribute to \( L_1 \).
2. \( \lim_{x \to +\infty} \sigma(x) = 0 \), causing the MSE of the corresponding predictions to not contribute to \( L_1 \).

In other words, predicting a negative value for \( \mu^{[i,j]} \) causes \( \sigma(\mu^{[i,j]} \Delta \tau^{[i,j]}) \approx 0 \) and \( \sigma(\mu^{[i,j]} \Delta \alpha^{[i,j]}) \approx 0 \) and hence close to the corresponding 0 in the labels.

Finally, both loss components are combined in a weighted sum such that

\[
L_{total} = \beta \cdot L_0 + \hat{\beta} \cdot L_1.
\]  

(6)

For our experiments, we chose \( \beta = 4 \) to ensure both losses equally contribute to the learning.

### 3.5. Training

The three synthetic datasets, a training-, validation-, and testset, were created by sampling random values from differently seeded pseudo-random number generators. Table 1 contains a comprehensive summary of the respective settings for the dataset and training hyperparameters.

Each sample in the dataset contains a random number of 1 to 20 specular paths. The complex path amplitudes \( \gamma \) contain random phases, and their magnitudes are uniformly spread across the range
of 0 dB to −30 dB. To prevent overfitting, the measurement noise \( N \) is generated randomly for every snapshot with a varying noise power level, such that the SNR is in the range of 0 dB to 50 dB. We use a uniform distribution in the linear domain, such that 90% of samples have a SNR < 10 dB.

4. ANALYSIS

To provide as initial assessment of the parameter estimation performance of our approach, we compare it to a periodogram-based peak search (grid-limited), and two high-resolution algorithms: ESPRIT ([9], with 6x6 spatial smoothing and LS) and ML-based Rimax estimation [10]. The model order required for the peak-search and ESPRIT is obtained from the EDC [11]. We provide a comparison of the MSE for the estimated parameters and the model order error.

The MSE is computed only for those estimated parameters with a match in the groundtruth. Figure 3 shows that our approach can break the grid limitation and outperform the periodogram-based method. In the low-SNR domain below <15 dB it also performs slightly better than ESPRIT. As expected, the results of the SOTA estimators improve with increasing SNR, while our approach saturates for SNRs >20 dB. This might indicate the architecture is unsuitable to achieve better accuracies, or is related to the rare occurrence of high SNR samples \( P(\sigma > 20 \text{ dB}) = 0.01 \) during the training. Another possible reason, also noted in [9], could be the result of predicted outliers, which at some point in the training cause a more significant error than further optimization of the MSE. However, further study is required to understand the effect and improve the results.

Apart from the accuracy, the computational complexity of the algorithms is also of interest. As an assessment of the computational complexity of an ML estimator is not straightforward due to the use of iterative numerical methods, we assess the runtime per sample of each estimator on an identical system. The periodogram approach ranks fastest with an average of 3 ms, followed by our approach with 19 ms. ESPRIT takes third with 127 ms, while the ML algorithm requires around 11.9 s on average. It highlights that our approach can be helpful in applications requiring fixed-clock, near-real-time estimates. However, if better accuracy is desired, our approach can also be combined with additional gradient steps on the likelihood function, as demonstrated in [6].

Regarding the model order estimation, we compare our approach to EDC and the model order extracted from the ML estimates. The results are illustrated in Figure 4 and reaffirm the results of previous publications [1, 6]. Our approach consistently achieves the best results across the studied SNR range. EDC and achieves similar performance at SNRs >20 dB, but underestimates the model order for smaller SNR.

5. CONCLUSION

Our work introduces a new approach for multi-dimensional estimation of signal parameters and the model order using a CNN. Compared to recent approaches in the field, it uses a grid-relative representation of signal parameters for the prediction, and, therefore, can estimate parameters directly without typical on-grid limitations. Due to the CNN our approach scales well to multiple dimensions and requires no separate pairing for the parameter estimates. Regarding estimation accuracy, our approach performs similarly to ESPRIT in the low-SNR domain and outperforms on-grid periodogram-based approaches. In line with previous work, it demonstrates superior model order estimation, especially in the low-SNR domain. Our approach is well-suited for time-constraint parameter estimation tasks because of its comparably low-computational complexity compared to high-resolution methods. The opportunities for future work are manifold. A more detailed study of the accuracy-SNR relationship is required, and an ablation study should quantify the performance impacts of architecture blocks. Further opportunities are the applicability to wideband channel data to assess the performance under dispersion-related effects. Similarly, it must be tested with measurement data to understand if the results are consistent under the influence of system imperfections. However, as Neural Networks are well known to account for nonlinearities, the approach has the potential to combat otherwise difficult model system imperfections.

![Fig. 3. Comparison of the MSE. Our method can not achieve the high accuracy of SOTA algorithms but shows on-par performance to ESPRIT in low-SNR conditions while outperforming the Periodogram.](image1)

![Fig. 4. Average Error of the model order. Our method outperforms both EDC and the maximum-likelihood approach. Note that the results are averaged across mini-batches (32 samples each) to avoid spikes in the Violin plot.](image2)
6. REFERENCES

[1] Gautier Izacard, Sreyas Mohan, and Carlos Fernandez-Granda, “Data-driven estimation of sinusoid frequencies,” in Advances in Neural Information Processing Systems, 2019, vol. 32, pp. 5127–5137.

[2] Georgios K. Papageorgiou, Mathini Sellathurai, and Yonina C. Eldar, “Deep networks for direction-of-arrival estimation in low snr,” IEEE Transactions on Signal Processing, vol. 69, pp. 3714–3729, 2021.

[3] Dawei Chen, Shuo Shi, Xuemai Gu, and Byonghyo Shim, “Robust DoA Estimation Using Denoising Autoencoder and Deep Neural Networks,” IEEE Access, pp. 1–1, 2022, Conference Name: IEEE Access.

[4] Mostafa Naseri, Adnan Shahid, Gert-Jan Gordebeke, Sam Lemey, Michiel Boes, Samuel Van de Velde, and Eli De Poorter, “Machine Learning-Based Angle of Arrival Estimation for Ultra-Wide Band Radios,” IEEE Communications Letters, pp. 1–1, 2022, Conference Name: IEEE Communications Letters.

[5] Wanli Liu, “Super resolution DOA estimation based on deep neural network,” Scientific Reports, vol. 10, no. 1, pp. 19859, Dec. 2020.

[6] Andreas Barthelme and Wolfgang Utschick, “A Machine Learning Approach to DoA Estimation and Model Order Selection for Antenna Arrays With Subarray Sampling,” IEEE Transactions on Signal Processing, vol. 69, pp. 3075–3087, 2021.

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 2016, pp. 779–788, IEEE.

[8] Diederik P. Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” CoRR, vol. abs/1412.6980, 2015.

[9] H. Krim and M. Viberg, “Two decades of array signal processing research: the parametric approach,” IEEE Signal Processing Magazine, vol. 13, no. 4, pp. 67–94, July 1996.

[10] Andreas Richter, Estimation of Radio Channel Parameters: Models and Algorithms, Thesis, Technische Universität Ilmenau, Ilmenau, 2005.

[11] L.C. Zhao, P.R. Krishnaiah, and Z.D. Bai, “On detection of the number of signals in presence of white noise,” Journal of Multivariate Analysis, vol. 20, no. 1, pp. 1–25, Oct. 1986.