SignalNet: A Low Resolution Sinusoid Decomposition and Estimation Network

Ryan M. Dreifuerst, *Student Member, IEEE,* Robert W. Heath Jr. *Fellow, IEEE*[*]

August 21, 2022

***Abstract*—The detection and estimation of sinusoids is a fundamental signal processing task for many applications related to sensing and communications. While algorithms have been proposed for this setting, quantization is a critical, but often ignored modeling effect. In wireless communications, estimation with low resolution data converters is relevant for reduced power consumption in wideband receivers. Similarly, low resolution sampling in imaging and spectrum sensing allows for efficient data collection. In this work, we propose SignalNet, a neural network architecture that detects the number of sinusoids and estimates their parameters from quantized in-phase and quadrature samples. We incorporate signal reconstruction internally as domain knowledge within the network to enhance learning and surpass traditional algorithms in mean squared error and Chamfer error. We introduce a worst-case learning threshold for comparing the results of our network relative to the underlying data distributions. This threshold provides insight into why neural networks tend to outperform traditional methods and into the learned relationships between the input and output distributions. In simulation, we find that our algorithm is always able to surpass the threshold for three-bit data but often cannot exceed the threshold for one-bit data. We use the learning threshold to explain, in the one-bit case, how our estimators learn to minimize the distributional loss, rather than learn features from the data.**

***Index Terms*—Sinusoid decomposition, estimation, deep-learning, low-resolution, detection**

I. Introduction

Detecting the number of sinusoids and estimating the amplitude, frequency, and phase is a common problem in signal processing [1]–[3]. In these settings, information is contained in the sinusoidal parameters of the data, i.e. in the Doppler shift from a radar signal or the angles of arrival and departure in the spatial domain. As bandwidth and array size continue to increase, e.g. in mmWave communications or wideband automotive radar, traditional systems require analog-to-digital converters (ADCs) with high resolution and rates approaching 100 Gbps or more. High resolution data converters become a limiting factor for low power consumption [4]. One solution to reduce power consumption is to reduce the resolution in ADCs, transferring the difficulty of the problem from the replaced power ADCs to practical digital signal processing algorithms [4]–[7].

[*]Ryan M. Dreifuerst and Robert W. Heath Jr. are with North Carolina State University, Raleigh, NC 27695 (rmdreifu@ncsu.edu, rwheathjr@ncsu.edu). This work is based upon work supported in part by Samsung Research America and the National Science Foundation under Grant No. ECCS-1711702.

In principle, low resolution techniques improve efficiency–computationally, economically, or in power consumption–at the cost of introducing quantization error. With high resolution sampling the quantization error is often modeled as an additive noise term [5], [8]. This is motivated by techniques to linearize the quantization, such as Bussgang Decomposition [9], [10]. Ultimately, the linearization has more error for 1-3 bit quantizers, which are the most desirable from an efficiency perspective. To overcome this limitation, we investigate neural network techniques for low resolution signal processing because of their ability to learn analytical models directly from data. The use cases for such algorithms are broad, with varying compute constraints (server, cloud, deployed), but our algorithm is primarily intended for edge computer signal processing, so we also consider important characteristics such as memory, training sample size, and execution time in our investigation.

There is, to the best of our knowledge, little prior work that has focused directly on the joint task of detection and estimation of sinusoids from quantized data. In contrast, the field of *unquantized* detection and estimation of sinusoids has a rich history [11]–[15]. These techniques tend to divide the problem into separate, but related, tasks for detection and estimation. Detection is usually performed using model order estimators such as Minimum Description Length (MDL) [13], [16] or Akaike Information Criterion (AIC) [12], [17]. Then, given a prediction of the number of sinusoids, the estimation is typically solved using non-parametric or parametric methods [15], [18], [19]. Alternatively, the problem can be approached using compressive sensing [20] techniques. In this paradigm, the recovered signal vector is the frequency representation of the multiple sinusoids, enabling sparse reconstruction techniques. Notable algorithms in this domain are based on the message passing algoritm [21], [22]. These methods are restricted to on-grid measurements, however, which affects the accuracy for limited samples. More recently, an additional class of algorithms based on neural networks have also been tasked with the same problem [23]. This has led to new state of the art results for the detection and estimation of sinusoid frequencies [23]. Neural networks have the advantage of learning directly from training data, so no assumptions regarding the SNR or model order are necessary. Instead, the model learns assumptions and distributions from the training data, possibly leading to inadvertent bias. While inductive bias is necessary for successful learning [24], it can have adverse effects when data distribution changes. Our results specifically

look at the effects of data distribution and distribution shift for model validation.

More prior work has focused on the estimation of a single quantized sinusoid in noise [25]–[27]. The Cramér Rao Bound was derived in [25]. Algorithms using correlation [25], dithering [26], and time-varying thresholding [27] have been proposed for quantized sinusoid estimation. Dithering and time-varying thresholding are related by effectively adjusting the sampling comparator by either a known (time-varying thresholds) or unknown factor (dithering). These methods rely on knowledge of the amplitude of the signal, or the ability to estimate by iterative grid search. Unlike the unquantized case, non-parametric methods, such as the Periodogram, are often the best estimators with extremely low-resolution sinusoid estimation [26], [28]. All of these techniques (non-parametric estimators [19], correlation-based estimators [25], dithering [26] and time-varying thresholds [27]) do not make use of the quantization noise correlation [10]. This correlation is not easily modeled in accurate and tractable forms. Instead, our neural network attempts to learn the underlying relationships directly from data, bypassing the need for explicit models. We also contend with new concepts such as the minimum recognizable separation between signals, which strongly limits the performance of the previously mentioned estimators.

Our paper is a rigorous investigation into deep learning for sinusoidal parameter estimation from low resolution sampling. We propose and analyze a state of the art architecture, SignalNet, that relies on internal reconstruction to successfully estimate multi-sinusoidal representations. Traditional algorithms do not account for quantization effects and instead are applied with Bussgang Decomposition to linearize the nonlinear distortion. This is the defacto approach for other low resolution problems like channel estimation [7], [10]. While it may not be surprising that neural networks are well-suited to handle nonlinear systems, we show that the use of domain knowledge within SignalNet outperforms other neural networks. Finally, we analyze the proposed network with both statistical measures through a new learning threshold and out-of-distribution data, which is testing data drawn from a different distribution than the training data. These tests show that the SignalNet architecture is able to learn meaningful features that generalize well and ensure the neural model is capable of handling crucial signal processing tasks across many domains. Our algorithm follows traditional approaches by dividing the task into two separate networks, which we identify as the *detection module* and the *estimation module*. The focal point of our algorithm is the estimation module, which sequentially estimates sinusoid parameters. Within the estimation module, we configure the network to reconstruct the input sequences and feed the error back into the estimation module, thereby explicitly defining the relationship between the output estimates and input sequences. This method instills domain knowledge within the network and helps the model directly capture the input-output relationship for the sinusoid detection and estimation problem. We then benchmark our algorithm against classical methods for each module, as well as define and derive learning thresholds for the problem based on estimation theory. In simulation, we show that both of our modules outperform traditional algorithms for quantized data. In spite of this, our detection module and amplitude estimation algorithms are not able to surpass the learning threshold for one-bit resolution data. Our contributions can be summarized as follows:

- We propose a novel deep learning algorithm for sinusoid parameter estimation. Our algorithm uses reconstruction as an internal mechanism for learning the relationship between sinusoidal parameters and input data. We show that this method is able to estimate sinusoids significantly more accurately than traditional algorithms for quantized data. We further show that our network generalizes nearly ideally to out-of-distribution datasets.
- We extend our sinusoid parameter estimator to include multiple sinusoid estimation and detection. We find that the overall network, SignalNet, is able to accurately detect the number of sinusoidal signals and their associated parameters from limited observations of quantized data. The proposed algorithm achieves state of the art results and universally improves upon the benchmark algorithms.
- We define a new benchmark for comparing neural network algorithms and traditional algorithms based on the loss function and distribution of the output distribution. The resulting threshold defines the worst-case error expected for non-adversarial data, and provides insight into why neural networks tend to outperform other algorithms within signal processing–even for nearly random data. Applying this threshold to our simulations shows that our one-bit sinusoid detection algorithm and one-bit amplitude estimates are not able to learn meaningful input-output relationships. All other networks surpass the learning threshold and traditional algorithms, suggesting our algorithms are learning substantial information from the data.

Notation: $\mathbf{A}$ is a matrix, $\mathbf{a}$ and $\{a[i]\}_{i=1}^{N}$ are column vectors and a, A denote scalars. $\mathbf{A}^T$, $\overline{\mathbf{A}}$ and $\mathbf{A}^*$ represent the transpose, conjugate, and conjugate transpose of $\mathbf{A}$. The real and imaginary parts of $\mathbf{A}$ are denoted by $\mathrm{Re}(\mathbf{A})$ and $\mathrm{Im}(\mathbf{A})$. $A[k,\ell]$ denotes the entry of $\mathbf{A}$ in the k^{th} row and the ℓ^{th} column. a_ℓ refers to the ℓ^{th} element of $\mathbf{a}$ and $\mathbf{a}_\ell$ refers to the ℓ^{th} column of $\mathbf{A}$. Similarly, $\mathbf{A}[:,k]$ refers to the k^{th} column of $\mathbf{A}$. Unspecified norm equations are $\|\mathbf{a}\|_2 = \mathbf{a}^*\mathbf{a}$ for vectors, and the Frobenius norm $\|\mathbf{A}\|_F = \sqrt{\mathrm{Tr}(\mathbf{A}\mathbf{A}^*)}$ for matrices. We define $j = \sqrt{-1}$. $\lfloor a \rfloor$ is the nearest integer from truncating a and $\lceil a \rceil$ is the nearest integer greater than or equal a.

II. System model

We begin by defining the classical representation of a received sample set, comprised of multiple sinusoids, in noise. We then clarify how quantization and normalization are performed, based on common hardware considerations for gain control. Afterwards, we summarize the learning objects of each subtask and present two loss functions based on prior work and the role of our algorithm in processing signals.

An m multi-sinusoidal signal is represented by vectors of amplitudes, phases, and frequencies $\mathbf{a}, \boldsymbol{\phi}, \mathbf{f} \in \mathbb{R}^M$. The

sinusoidal components are summed and sampled at sampling intervals T as

$$u[n] = \sum_{i=1}^{m} a_i \exp\left(j2\pi f_i nT + \phi_i\right). \tag{1}$$

Given an infinite number of samples, the exact signal components can be resolved from this model, so long as the sampling period is small enough such that the sampling theorem [29] is maintained. In realistic settings, the number of samples is finite and the desired signal is obstructed by noise $v[k]$, which is often modeled as additive, independent, and identically distributed (IID), Gaussian noise. For brevity of notation, we define the $m \times N$-dimensional matrix $\mathbf{G}$ such that $\mathbf{g}_i = \{2\pi f_i nT + \phi_i\}_{n=0}^{N-1}$. Now (1) can be represented as a finite set of N real and imaginary components with complex-valued noise

$$\mathbf{u} = \sum_{i=1}^{m} a_i \Big(\cos\left(\mathbf{g}_i\right) + j \sin\left(\mathbf{g}_i\right) \Big) + \mathbf{v}\,. \tag{2}$$

This model aligns with classic [11]–[15] and recent [23] approaches. We now extend the system model to include quantization effects.

Prior work with quantization has generally considered the estimation of only a single sinusoid [25], [30], [31] or one-bit resolution [26], [27], [32], thus defining the quantization levels to maximize the dynamic range (and limit clipping) is straightforward. Modeling quantization effects for multiple bit quantizers with varying signals is not as simple, and is not done consistently in literature [5], [7], [25], [33]. Yet, defining the quantization levels in a realistic manner is crucial for the system model. As a result, we define the quantization model for unit-norm power, based on ideal traditional gain control hardware found in wireless systems. Specifically, the signal model is normalized and quantized according to a b bit uniform quantizer $\mathbb{Q}_b$ as

$$\mathbf{z} = \mathbb{Q}_b\left(N \frac{\mathbf{u}}{\|\mathbf{u}\|^2} \right) \tag{3}$$

$$\mathbb{Q}_b(A + jB) = \mathbb{Q}_b(A) + j\mathbb{Q}_b(B) \tag{4}$$

$$\mathbb{Q}_b(A) = q_i \quad \text{if } A \in (q_{i-1}, q_i]. \tag{5}$$

The assumption of ideal gain control is a simplification, however, the nonlinear effects of a realistic gain controller would almost certainly be eclipsed by the subsequent quantization function. There are 2^b quantization levels $\mathbf{q} := \{q_i\}_{i=0}^{2^b}$ that can be output from $\mathbb{Q}_b$ covering $2^b + 1$ bins in the range of $[-1, 1]$. The output of the quantization function is assigned q_i if the input is in the range $(q_{i-1}, q_i]$, and the first and last bins extend to $\pm\infty$. We are interested in low resolution settings, where $B \in \{1, 2, 3\}$. Higher resolutions are often not significantly different from 3 bit resolution with idealized gain control, as we assume in (3). We further restrict our system to the two edge cases, $B = \{1, 3\}$ bits to highlight the key differences between extremely low resolution (one-bit) and modestly low resolution (three-bit) sampling.

Complex number support is limited in many deep learning frameworks, so we vectorize the real and imaginary components to form the $\mathbb{R}^{N\times 2}$ received signal matrix as

$$\mathbf{X} = \Big[\mathrm{Re}(\mathbf{z}), \mathrm{Im}(\mathbf{z}) \Big]^T. \tag{6}$$

It is noteworthy that an alternative approach could be to use a polar representation, or define complex layers that operate using complex-valued optimization [34]. Such architectures are an interesting topic of future investigation.

Given the input to the system from (6), we mathematically summarize the goal of our neural network from a general learning perspective. First, we divide the problem into two tasks, detection and estimation, to be learned by two different neural network architectures. Detection is performed first. The goal is for our algorithm to learn the parameters $\mathbf{W}_1$ for a function $\beta(\mathbf{X}; \mathbf{W}_1)$ that relates a set of observations $\mathbf{X}$ to the true number of sinusoidal components m in the original $\mathbf{u}$, based on a loss function $\mathrm{L}_{\mathrm{det}}$. The estimation network function, $\Theta(\mathbf{X}; \mathbf{W}_2, m)$, has parameters $\mathbf{W}_2$ and is trained to predict $\{\hat{\mathbf{a}},\ \hat{\mathbf{f}},\ \hat{\boldsymbol{\phi}}\}$, assuming perfect knowledge of m, with a loss function $\mathrm{L}_{\mathrm{est}}$. The parameters, which are the neural network weights, are learned from feasible sets $\mathbf{W}_{\mathrm{det}}, \mathbf{W}_{\mathrm{est}}$ for the corresponding tasks and neural architectures. The training can be summarized as solving the problem

$$\mathbf{W}_1 = \underset{\mathbf{W}\in\mathbf{W}_{\mathrm{det}}}{\arg\min}\, \mathbb{E}_{\mathbf{X}}[\mathrm{L}_{\mathrm{det}}(m, \beta(\mathbf{X}; \mathbf{W})] \tag{7}$$

$$\mathbf{W}_2 = \underset{\mathbf{W}\in\mathbf{W}_{\mathrm{est}}}{\arg\min}\, \mathbb{E}_{\mathbf{X}}[\mathrm{L}_{\mathrm{est}}(\{\mathbf{a},\ \mathbf{f},\ \boldsymbol{\phi}\}, \Theta(\mathbf{X}; \mathbf{W}, m))|m]. \tag{8}$$

The networks are combined and the overall estimate is produced according to

$$\hat{m} = \beta(\mathbf{X}; \mathbf{W}_1) \tag{9}$$

$$\{\hat{\mathbf{a}},\ \hat{\mathbf{f}},\ \hat{\boldsymbol{\phi}}\} = \Theta(\mathbf{X}; \mathbf{W}_2, \hat{m}). \tag{10}$$

All of the outputs in (10) are real vectors of size $\hat{m}$. Additionally, (10) and (8) imply that the estimator $\Theta(\mathbf{X}; \mathbf{W}, \hat{m})$ is specific to the value of $\hat{m}$ provided during detection. We make use of this in Section III, where internal reconstruction is used to estimate the $\hat{m}$ signal set.

We note that (7)-(10) can be applied to the traditional algorithms without quantization by reframing the training steps and meaning of $\mathbf{W}_1, \mathbf{W}_2$. By regarding the "learning" phase as a period of collecting sample statistics, (7) and (8) can be thought of as a tuning process for determining Bussgang gain [10] and algorithm parameters such as dither scaling [26]. The minimization is performed using known equations, such as the multi-level quantization presented in [7]. Then the functions β, θ can be regarded as the combination of the Bussgang Decomposition, a detection algorithm (e.g. AIC) and an estimation algorithm (non-parametric/parametric).

The goal of our network is to estimate the relevant sinusoidal components as an initial signal processing task, leaving possible filtering or decoding to subsequent algorithms. As a result, the detection algorithm should learn to overestimate the number of signals during uncertainty, rather than underestimate. This way higher level algorithms can filter out small or irrelevant components rather than miss signals altogether. To

instill this knowledge, we define a heavy-sided loss function for the detection network

$$\mathrm{L}_{\mathrm{det}}(m,\hat{m}) = \begin{cases} e^{m-\hat{m}} - 1 & \text{if } m \geq \hat{m} \\ \frac{1}{2}(m-\hat{m})^2 & \text{otherwise.} \end{cases} \tag{11}$$

The specific internal functions comprising (11) are chosen to be differentiable, smooth, and ensuring that the loss has the relationship $\mathrm{L}_{\mathrm{det}}(m, m+1) < \mathrm{L}_{\mathrm{det}}(m, m-1) < \mathrm{L}_{\mathrm{det}}(m, m+2)$, which we make use of in our analysis of the learning threshold. Note that differentiability and smoothness are determined by the functions, rather than the input space.

In the estimation module, we will train the network based on mean squared error (MSE), assuming the true number of signals m is known. Because the final model could potentially have different model order predictions, a similar loss function cannot be used for the overall SignalNet. Instead, when both modules are combined, we evaluate the network according to the Chamfer distance [35], which is used in [23] and defined as

$$L_{\text{chamfer}} = \sum_{f_i \in \mathbf{f}} \min_{\hat{\mathrm{f}}_k \in \hat{\mathbf{f}}} |f_i - \hat{\mathrm{f}}_k| + \sum_{\hat{\mathrm{f}}_i \in \hat{\mathbf{f}}} \min_{f_k \in \mathbf{f}} |\hat{\mathrm{f}}_i - f_k|. \tag{12}$$

The comparison can be applied to the frequency estimates as shown, or the other desired quantities. Effectively, this loss function penalizes for detecting the incorrect number of signals $\hat{m} \neq m$ as well as for incorrect estimates. Poor estimates are penalized twice, however, due to the two summations. An alternative loss function could be based on cosine similarity or other metrics such as the optimal subpattern assignment metric [36]. We believe that the correct approach, however, is to use the Chamfer distance between the vectors of different sizes. The reason is that the complexity is very simple and the parameter values can not necessarily be chosen to indicate model order. For example, zeros in the amplitude vector indicate the absence of a sinusoid while while zeros in the phase vector imply the presence of a sinusoid with zero phase.

III. SignalNet

In this section, we define the SignalNet architecture, comprised of two sub-networks for detection and estimation. Readers interested in learning more about the basics of deep learning are encouraged to review [37]. We focus heavily on the estimation side of the problem, because extraneous signals can be eliminated at later points, but poorly estimated signals are detrimental to application-specific information. To improve the learning of our estimation module, we include domain knowledge through reconstruction and cancellation, similar to successive interference cancellation [38], with similar algorithms also based on it such as [39]. We provide a description, figures, and a table summarizing the operation of SignalNet, as well as source code[1].

We design SignalNet, shown by the modules and overall architecture in Figures 1-3, according to the general structure in (6)-(10), where one module detects the number of sinusoids in the signal and another estimates the sinusoids parameter from an observed sequence $\mathbf{X}$. The detection module is used

[1]https://github.com/Ryandry1st/Carrier-Frequency-Offset

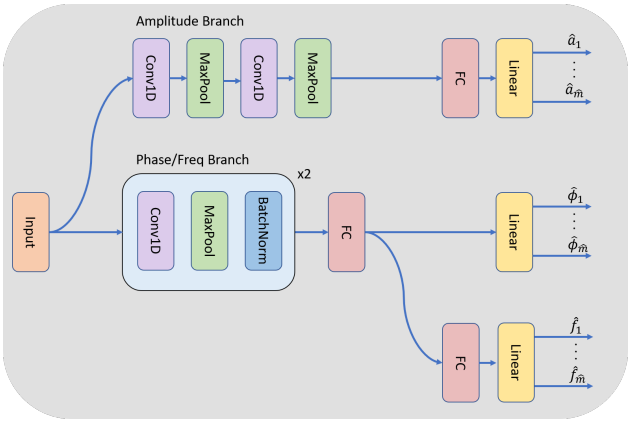


Fig. 1. A diagram of a block estimator used to determine the parameters for a specific number of sinusoids. The two path network is made up of an unnormalized branch for amplitude estimation and a normalized branch for frequency and phase estimation. Frequency and phase estimation are linked because of the natural linear relationship between frequency and change in phase over a set of N samples. This is equivalent to determining the frequency as the average phase change over the N samples.

to estimate the number of sinusoids present in a signal, and is modeled using a three layer convolutional neural network with a softmax output. We use a softmax activation function on the output, which outputs a one-hot vector representing the most likely number of sinusoids detected. Although a real-valued system could be used with rounding, the domain of the output is significantly larger (i.e. the outputs are no longer a discrete set of M values), causing the performance to be negatively impacted.

The sinusoid estimation module, however, requires more direct knowledge of the relationship between the output parameters and the input $\mathbf{X}$ to effectively capture the information. To do this, we design the estimator around the idea of successive-estimation and cancellation, similar to interference cancellation methods in non-orthogonal communications [38]. The network architecture first estimates the parameters for a single sinusoid, reconstructs the time-domain signal, and compares it with the original input. Then, it estimates the parameters for the next sinusoid from the difference. This knowledge helps our neural architecture better handle multiple frequencies. A block diagram of the sinusoid estimator is shown in Figure 2. Internally, each sinusoid estimator has two branches of two convolutional layers where one branch has batch normalization for estimating frequency and phase components and the other does not to retain amplitude information. Hyperparameters are tuned using iterative grid search for both the sinusoid estimator and detection module, and layer specifics can be found in Table 1 in the supplementary material.

Recalling (10), we train all of the networks independently, where each $m \in \{1, ...M\}$ possible number of outputs and each quantization resolution $b \in \{1, ...B\}$ result in a different network, totaling $M \times B$ networks in our investigation. In this setup, we build the b-bit SignalNet architecture from M different sinusoid estimator networks. The need for having separate estimators for each number of sinusoids is based on two considerations: real-time feasibility and our choice of internal reconstruction. The real-time feasibility is a re-

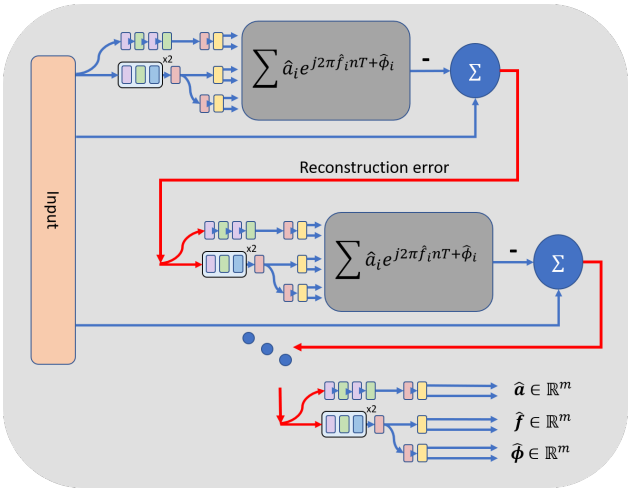


Fig. 2. The sinusoid estimator is comprised of multiple block estimators. These networks successively estimate $\{1, ...m\}$ sinusoids by estimating and reconstructing each number of sinusoids and recomputing on the error. Internal reconstruction is used to provide clear relationships between the estimated outputs $[\mathbf{a}, \mathbf{f}, \boldsymbol{\phi}]$ and the input $\mathbf{X}$. This formulation results in each $m \in 1, ...M$ estimator producing different outputs and learning different features.

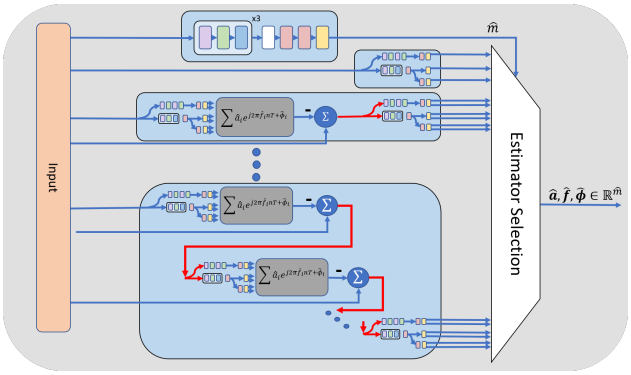


Fig. 3. A block diagram of SignalNet. The architecture includes M sinusoid estimator modules and one detection module, all trained for a specific quantization resolution b. Each subnetwork is highlighted, with the top nextwork being the detection module, and the following networks are the $\{1, 2, \text{and} M\}$ sinusoid estimators. In our investigation, we develop a SignalNet variant for one-bit and three-bit resolutions, resulting in $(M + 1) \times 2$ total networks.

sult of graph-based optimization, which does not allow for dynamically-sized outputs without sacrificing inference speed. Offline training time evaluation is not considered in this paper, as our algorithm only needs to be trained and then deployed. Along with computational performance, our use of internal reconstruction requires separate estimators for each number of sinusoids. When estimating and reconstructing an unknown number of sinusoids, the first sinusoid estimated will be different depending on the total number of sinusoids because of the goal of reconstructing a similar set of samples as the network's input. In other words, our network does not attempt to find the peaks of DFT bins. Instead, it finds the parameters that generate a signal most closely approximating the input, while gradients are only updated from the parameter error. This is best understood from Fig. 10 in the supplementary section where we show a two-sinusoid signal with the best reconstruction using $\{1, 2, 3\}$ sinusoids. The output of the single sinusoid estimator will not be either of the two underlying sinusoids, but some sinusoid between the two. After training, the detection and estimation modules are joined according to Figure 3.

IV. Simulation Setup

We consider a simulation setup similar to [23], with the notable exception of considering a reduced range of sinusoids $M = 5$. The most important aspect is the non-uniform frequency distribution of our data generation, as outlined in Algorithm 1. The frequency generation is unusual by necessity due to limitations on the frequency spacing and to remove unintended bias from the network performance. One might consider using frequency components drawn from an independently sampled uniform distribution. Unfortunately, it has been shown that resolving frequency content with spacing equal or less than $1/N$ from N discrete, noisy samples is non-convergent and may result in excessive errors [40], [41]. Although off-grid techniques can be used to resolve sparse signals (for some appropriate basis), resolving two signals with less separation and in the presence of noise is provably inconsistent [41]. In contrast, we could use uniformly spaced frequency content, however, this would introduce significantly more structure to the data that would unintentionally bias the neural network. For example, rather than estimate each sinusoid, the network would simply need to learn to estimate the spacing and a single frequency with high accuracy, thereby changing the structure and goal of the system. A neural network trained in such a way would be unlikely to generalize well to less structured settings.

Instead, we use a similar setting to [23], where we start by selecting a random number of sinusoids to be present. Then the uncertainty in the spacing is selected by a folded normal distribution along with the initial frequency offset from a uniform distribution. Finally, the maximum frequency is compared with 0.5, to ensure that no frequency content is undersampled according to the Nyquist criterion [29]. The frequency distribution for each m is shown in Fig. 11. Then amplitudes and phases are drawn from uniform distributions and the signal is constructed by summing over all of the sinusoids and adding Gaussian noise to obtain $\{y\}_{n=0}^{N-1}$ for a desired SNR. In the final steps, the signal is normalized to unity power norm, representative of an ideal automatic gain controller or similar control systems at the input to the data converter. Finally, quantization $\mathbb{Q}_b$ is applied according to the number of bits b.

We select the simulation setup based on two considerations for the difficulty (i.e. challenge in achieving accurate estimates) of the problem: the length of the data and the separation of the frequencies. For length N discrete observations, resolving unquantized data with separation of $1/N$ is nearly impossible in the presence of noise [40], [41], and in particular when the signal to quantization noise ratio (SQNR) is below 20 [42] as we have here. We do not strictly enforce this separation, but use the folded normal distribution to discourage it, resulting in some situations that are nearly unresolvable. This leads to estimators that may have overlapping frequency

Algorithm 1 Simulation Data Generation

Draw a random integer, m from $[0,\ M-1]$
do
  $w_0 \sim \mathcal{U}(0, 0.25)$
  $f_1 \leftarrow w_0$
  for $i \in [1, ...m-1]$ **do**
    $w_i \sim |\mathcal{N}(0, 2.5/N)|$
    $f_{i+1} \leftarrow w_0 + i/N + w_i$
  end for
while $\max f_i > 0.5$
$a_i \sim \mathcal{U}(0.1, 1.0) \quad i \in [1, ...m]$
$\phi_i \sim \mathcal{U}(0, 2\pi) \quad i \in [1, ...m]$
$\mathbf{y} \leftarrow \sum_{i=0}^{m} a_i \exp(2\pi j f_i n + \phi_i) + \mathbf{v}$
$y_i \leftarrow \frac{y_i}{\|\mathbf{y}\|} \quad i \in \{0, 1, ..N-1\}$
$\mathbf{x}_i \leftarrow \mathbb{Q}_b(y_i)$

content that could represent interference. Additionally, the amplitude of the signal directly contributes to the estimation error of the problem, so we constrain the amplitudes to the range of $[0.1, 1.0]$. These choices create a challenging simulation setup, as seen by the simulation results in Section VIII, especially the amplitude estimation results in Figure 6.

When testing the results, the data is drawn again from the same distributions according to Algorithm 1. This is not the same as if the training data is used for testing, only that the distributions the values are drawn from are the same, which is not a particularly strict assumption considering train-test splits are often done so that the distributions remain the same. That being said, we also evaluate the case where the frequencies are drawn from a different distribution in Section VIII-F.

V. Benchmarks

Because there are, to the best of our knowledge, no other algorithms designed for sinusoidal decomposition and estimation from quantized data, we limit the benchmarks to well-established estimators, in combination with Bussgang Decomposition [10]. We first provide a brief recount of Bussgang Decomposition and how it applies to our system model. Afterwards, we highlight the two common detection methods, AIC and MDL and the limitations that minimizing model order has on the detection loss function, (11). An alternative method, BIC [43] also has potential, though it is known to penalize over-estimation even more than AIC, which will inherently perform worse for our loss function, and can be shown to have equal performance to MDL in our scenario [24, pg. 236]. We also investigate EM-VAMP, a message passing algorithm, as a method for determining the model order in a compressed sensing form. EM-VAMP is an excellent bridge between traditional methods like AIC, and data driven methods like deep learning. In fact, AMP-based methods have been shown to outperform neural networks in similar scenarios such as low-resolution channel estimation [44]. Finally, we suggest the non-parametric methods we compare against based on the Fast Fourier Transform (FFT) and Periodogram.

The Bussgang Decomposition is a method for linearizing a function by computing the linear minimum mean squared error estimator. In the case of quantization, it has been shown to be a useful tool to separate the signal and quantization noise into two uncorrelated terms [7], [10] assuming a Gaussian signal. The essence of the decomposition is to calculate a gain factor G such that, for a nonlinear function $y = U(x)$, the result is

$$\begin{aligned} y &= Gx + \eta \\ \mathbb{E}[y\eta] &= 0 \\ \mathbb{E}[x\eta] &= 0 \end{aligned}$$

which is the linear minimum mean squared error estimator (LMMSE) of x from y. This formulation extends to vector signals, causing G to become $\mathbf{G}$ which is a diagonal matrix. The exact formulation for multi-bit quantization, $U = \mathbb{Q}_b$, is provided in [7]. One of the key components in the formulation is knowledge of the first and second moments of $\mathbf{x}$. Because of our power normalization and gaussian noise, we observe that the complex observation vector $\mathbf{X}[:, 0] + j\mathbf{X}[:, 1] \sim \mathbb{CN}(\mathbf{0}, \mathbf{1})$. Then, by using the inverse of $\mathbf{G}$ and neglecting the distortion, we can obtain the linearized recovery of $\mathbf{x}$

$$\tilde{\mathbf{x}} = \mathbf{G}^{-1}(\mathbf{X}[:, 0] + j\mathbf{X}[:, 1]). \tag{13}$$

We will apply this decomposition to the following benchmark detection and estimation algorithms in our simulation results.

We evaluate our detection module against AIC and MDL, although both of these methods are suboptimal due to misaligned goals. Fundamentally, these methods attempt to minimize the complexity while capturing the number of spectral components. In contrast, the loss function (11), penalizes underestimates more than over-estimates. This, in part, leads to significantly under-performing detection results for AIC and MDL. Additionally, AIC and MDL are not well-suited for low resolution sampling, so in the case of 1-bit data, the results are non-convergent and have loss values far greater than other algorithms. To keep the scaling of the graphs meaningful, we instead show the 2- and 3− bit results for these two algorithms. We also show the EM-VAMP method, using a learned threshold for determining the model order from the recovered vector support. While not following the exact same structure as we outlined in Section II, this benchmark is similar to data driven algorithms, but is more structured than neural networks. The EM-VAMP method recovers the frequency-grid-aligned sparse signal $\mathbf{x}$ (the sinusoidal components in the frequency domain) from a 2x oversampled DFT observation matrix $\mathbf{A}$. Mathematically, the EM-VAMP method works to minimize the error from observations $\mathbf{y}$ for the model

$$\mathbf{y} = \text{sign}(\mathbf{A}\,\mathbf{x} + w) \tag{14}$$

by recovering the sinusoidal amplitudes and phases in the sparse coefficients of $\mathbf{x}$. This framework is limited due to its on-grid nature, however, so we only consider it for detection. For more information on EM-VAMP and GAMP methods, see [21], [22].

To benchmark our multi-sinusoid estimator, we employ the Periodogram, because it has been shown to produce better estimates for low resolution sinusoidal data than eigendecomposition and dithering methods [26], [28]. The Periodogram is calculated from the scaled-and-squared, zero-padded Fast

Fourier Transform (FFT) with $N_0 = 2^{16}$ to ensure that grid resolution is not a limiting factor. Although amplitude and frequency information can be obtained from the Periodogram, the phase information is recovered from the inverse tangent of the ratio between the imaginary and real components of the FFT. Precisely, the phase estimate is calculated first by the FFT of the samples, then selecting the m-largest magnitude of the FFT, and finally applying the inverse tangent (this would be atan2 in many programming languages). The steps can be mathematically expressed as

$$\mathbf{r} = \text{FFT}([\mathbf{x}, \mathbf{0}_{N_{\text{FFT}}-N}]) \in \mathbb{C}^{N_{\text{FFT}}} \tag{15}$$

$$\mathbf{p} = m - \text{argmax}(\mathbf{r}) \tag{16}$$

$$\equiv \{i \text{ for } p_i \in \text{abs}(\mathbf{r}) : |\text{abs}(\mathbf{r}) \cap (-\infty, p_i)| \leq m\} \tag{17}$$

$$\hat{\boldsymbol{\phi}}_{\text{FFT}} = \arctan\left(\text{Im}(\mathbf{r}[\mathbf{p}]), \text{Re}(\mathbf{r}[\mathbf{P}])\right). \tag{18}$$

We note two important clarifications in (15)-(18). First, the peak finding algorithm in step 2 simply finds the m largest maxima of the oversampled N_{FFT} FFT. Second, we make use of of both the absolute value and the size of a vector in the definitions, so we use $abs()$ to refer to the magnitude of the complex values and $|\ldots|$ to refer to the cardinality of the set.

VI. Learning Threshold

Before looking at the simulation results, we first introduce a baseline for algorithmic comparison, particularly for neural network evaluation. Because machine learning algorithms learn from data, there are inherent constraints and underlying distributions that algorithms can optimize for, creating unfair bias when compared to traditional algorithms. For this reason, we also define a learning threshold based on statistical estimation theory and Empirical Risk Minimization [24]. First, let $\mathbf{a} \in \mathbb{R}^p$ be the input and $b \in \mathbb{R}$ be the true output, with joint distribution $P(\mathbf{a}, b)$. The goal is for an estimator, $g_\theta(\mathbf{a})$ that is defined by parameters θ, to predict b. We wish to minimize a loss function, and for simplicity we start with mean squared error (MSE), for K predictions. Note, sample-wise MSE is defined as $\hat{L}_{\text{MSE}}(b, g_\theta(\mathbf{a})) = 1/K \sum_{i=1}^{K} (b_i - g_\theta(\mathbf{a}_i))^2$, where the parameters θ are updated based on training data. If the dataset is representative of the distribution, which occurs for sufficiently large numbers of samples in expectation, the empirical formulation matches the theoretical formulation as

$$\min_\theta L_{\text{MSE}}(b, g_\theta(\mathbf{a})) = \min_\theta \mathbb{E}_{\mathbf{a}} \mathbb{E}_{b|\,\mathbf{a}}[(y - g_\theta(\mathbf{a}))^2 |\, \mathbf{a}] \tag{19}$$

$$g_\theta(\mathbf{a}) = \mathbb{E}[b|\,\mathbf{a}]. \tag{20}$$

Now, we define the learning threshold as the worst-case training error, in the non-adversarial case, that occurs when $\mathbf{a}$ is independent from b. That is, when $\mathbf{a}$ is both independent from b and not chosen by an adversary to intentionally mislead an algorithm to the wrong outcome. This simplifies (19)-(20) to

$$L_{\text{th}}(b, g_\theta(\mathbf{a})) = \min_\theta \mathbb{E}_b[(b - g_\theta(\mathbf{a}))^2] \tag{21}$$

$$g_\theta(\mathbf{a}) = \mathbb{E}[b] \quad \forall\, \mathbf{a}\,. \tag{22}$$

The result, for this trivial example, is that the estimator should simply estimate the mean of the output variable, and the resulting error will be the variance of b. This well-known relationship is a foundational concept of our learning threshold, and is directly used for our algorithms trained with MSE loss. It is similar to other established estimation techniques such as the minimum variance unbiased estimator or the minimum mean squared error estimator when the loss function is the mean squared error. The formulation extends beyond mean squared error though, and can be applied to more interesting cases, in particular the detection module with loss function defined by (11). This distribution is heavy-sided and the output variable can only take integer values, leading to more extensive analysis.

The threshold analysis is not just an interesting derivation. We show in Section VIII that our algorithms for both detection and amplitude estimation converge to this threshold for one-bit data. This implies that the learning done by the neural networks is limited to *distributional learning*, which is simply learning to minimize the loss for an output variable's distribution and loss function. This can be thought of as "blind optimization" in the sense that the neural network achieves performance equivalent to a constant output function that ignores the input. Therefore, we can also see the inherent limitations of the threshold estimator: the output distribution and loss function must be known, and the constant estimator is only reasonable in expectation, with potentially poor results. Its intent is to judge whether a basic level of learning has occurred, rather than act as a true estimator.

The learning threshold will show that there is an inherent bias (in a learning theory sense) that is observable just from the distributions and loss function that can often surpass traditional methods. The threshold is relevant for two reasons: 1) it provides a scalar metric for the uncertainty in the output variable, with respect to the loss function, and 2) comparing the neural network's loss with this threshold provides clarity about whether a neural network is learning features or just optimizing for the output distribution. In general, we expect our neural network estimators to outperform the learning threshold, suggesting some relationship is learned between the input and the output. Because the quantization effects are not independent from the noise however, low SNR data can be misleading or adversarial. In contrast, the threshold does not change with the input SNR, so we would expect to see results that are similar to or worse than the learning threshold for sufficiently low signal to noise ratios and quantized signals. In the subsequent sections we will derive learning thresholds for the detection and estimation problems based on our simulation setting to lend some intuition to why neural networks outperform traditional methods. In the final section, we will evaluate our network using data from a different distribution, showing the generalizability of our model and the relationship to the learning threshold.

VII. Model training

In our setup, we train the two modules, the detection module and the sinusoid estimation module, separately. We employ the loss function in (11) to train the detection module for 20 epochs with $50,000$ realizations of one-hot encoded signal

counts. We evaluate the estimation module using the MSE in the training data, assuming that the true number of signals is known. This way each amplitude-frequency-phase estimate corresponds to a true set of parameters. Note that the mean-squared-error is defined for multi-output systems as the mean over the stacked vector of outputs, which is

$$L_{\text{MSE}}(\mathbf{c}, \hat{\mathbf{c}}) = \frac{1}{p} \| \mathbf{c} - \hat{\mathbf{c}} \|^2 \quad \mathbf{c}, \hat{\mathbf{c}} \in \mathbb{R}^p. \tag{23}$$

Unfortunately, not all parameters have the same scale or degree of randomness, relative to the loss function. Classically, this is mitigated by scaling all of the sample outputs to roughly mimic a standard normal distribution, or to be within the same range i.e. $[0, 1]$ [45]. In our case, because the output variables are not drawn from the same distribution, but the distribution is known, this technique does not appropriately scale the outputs. The learning threshold, as we defined in Section VI, is used to solve this problem by measuring the loss function with respect to the distribution of the output variables. We define the overall normalization used to achieve a unified scalar loss function as

$$\boldsymbol{\ell} = \left[L(\mathbf{a}, \hat{\mathbf{a}}), L(\mathbf{f}, \hat{\mathbf{f}}), L(\boldsymbol{\phi}, \hat{\boldsymbol{\phi}}) \right]^T \tag{24}$$

$$\boldsymbol{\ell}_{\text{th}} = \left[\frac{1}{\text{L}_{\text{th},\mathbf{a}}}, \frac{1}{\text{L}_{\text{th},\mathbf{f}}}, \frac{1}{\text{L}_{\text{th},\boldsymbol{\phi}}} \right]^T \tag{25}$$

$$\ell_{\text{eff}} = \frac{1}{m} \boldsymbol{\ell}^T \boldsymbol{\ell}_{\text{th}}. \tag{26}$$

In this case, each of the respective estimated parameters is stacked into a vector, and compared to the true parameters in mean squared error and represented by $L(\mathbf{x}, \hat{\mathbf{x}})$.Then the scaling (26) is applied to prevent the gradients from being dominated by parameters that have larger variance. We then train on $100,000$ data samples, using gradient updates from a scaled sum of loss functions based on the learning thresholds. We also use (26) again for computing the overall chamfer loss as a unified metric for SignalNet. The specific learning thresholds are derived in Section VIII. While training, we use learning rate reduction and early stopping based on a separate validation set to optimize the gradient learning. Batch sizes are set to 32, and networks are all trained using an Adam optimizer [51], with an initial learning rate of 0.001. When evaluating our algorithms, we generate $8,000$ samples of new data following the same algorithm for our test data.

Before evaluating our algorithm, we briefly remark on data size and memorization. The size of the training dataset, while initially appearing large, is extremely small compared to the input data space. For example, in the smallest case, $b = 1$, the size of the input data space is 2^{2N}, where $N = 64$ each taking a value of ± 1. Similarly, the size of our neural networks is much smaller than the data dimension as well, with the total number of parameters for SignalNet being $300,000$ parameters, but only at most $120,000$ parameters in any individual network. As a result, our algorithm is not capable of memorizing the data and could potentially benefit from increasing the dataset size by orders of magnitude. Furthermore, near real-time inference is achievable with modern processing hardware. With an Nvidia 3090 GPU and limited optimization, we were able to achieve inferences in under 1 ms. Our results in Section VIII show the efficacy of our algorithm to learn under these strict data and size constraints.

VIII. Simulation results

In this section, we evaluate each SignalNet component, first individually, then as a whole. We derive the learning threshold for each subtask, shown by dashed lines in each plot, and compare the simulation results of our algorithm along with the benchmark traditional algorithm. In the final setting, we combine the two modules and compare it with different combinations of our modules and the traditional methods.

A. Detection results

First, we evaluate the detection module, which is compared with EM-VAMP, AIC and MDL algorithms in Fig. 4. In this situation, the loss function is the detection loss defined in (11), and the possible outputs are $\hat{m} \in \{1, 2, ..5\}$. Then, the learning threshold is defined as

$$\text{L}_{\text{th,m}}(m, g_\theta(\mathbf{X})) = \min_{\hat{m}} \mathbb{E}_m[\text{L}_{\text{det}}(m, \hat{m})] \tag{27}$$

where $\hat{m} = g_\theta(\mathbf{X}) \ \forall \ \mathbf{X}$ is a constant estimator. Because the function is smooth, differentiable, and convex, it is a simple step to go from (27) to solving for $\hat{m}$ from

$$0 = \frac{\partial}{\partial m} \mathbb{E}_m[\text{L}_{\text{det}}(m, \hat{m})]. \tag{28}$$

Then, because L_{det} is bounded over the inputs, the derivative and expectation can be interchanged by the bounded convergence theorem. Following from (28),

$$0 = \mathbb{E}_m \left[\frac{\partial}{\partial m} \text{L}_{\text{det}}(m, \hat{m}) \right] \tag{29}$$

$$= \frac{1}{M} \sum_{m=1}^{M} \frac{\partial}{\partial m} \text{L}_{\text{det}}(m, \hat{m}). \tag{30}$$

The step from (29) to (30) is because m is uniformly distributed and discrete, so the expectation is the arithmetic mean. In our definition of L_{det}, we made the important restriction that $\text{L}_{\text{det}}(m, m+1) < \text{L}_{\text{det}}(m, m-1) < \text{L}_{\text{det}}(m, m+2)$, so the resulting threshold estimator will be $\hat{m} \in (\lceil M/2 \rceil, \lceil M/2+1 \rceil)$ for our simulation setting. We now ignore the constant $1/M$ factor, replace $\frac{\partial}{\partial m} \text{L}_{\text{det}}$ with its derivative components, and solve for the threshold estimator

$$0 = \sum_{m=1}^{\lfloor \hat{m} \rfloor} (m - \hat{m}) + \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m - \hat{m}} \tag{31}$$

$$= \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=1}^{\lfloor \hat{m} \rfloor} (m) - \hat{m} + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m - \hat{m}} \tag{32}$$

$$= \frac{\lfloor \hat{m} \rfloor + 1}{2} - \hat{m} + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m - \hat{m}}. \tag{33}$$

Defining $\alpha = \frac{\lfloor \hat{m} \rfloor + 1}{2}$ simplifies the results to achieve a expression for the constant estimator:

$$0 = \alpha - \hat{m} + \frac{e^{-\hat{m}}}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m} \tag{34}$$

$$= -e^{\hat{m}}(\hat{m} - \alpha) + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m} \tag{35}$$

$$= -e^{\hat{m}-\alpha}(\hat{m} - \alpha) + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\alpha} \tag{36}$$

$$\hat{m} = W\left(\frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\alpha} \right) + \alpha \tag{37}$$

$$= W\left(\frac{1}{\lceil M/2 \rceil} \sum_{m=\lceil M/2+1 \rceil}^{M} e^{m-\alpha} \right) + \alpha. \tag{38}$$

Here, $W()$ is the Lambert W function, and the final expression comes from the choice of $\mathrm{L}_{\mathrm{det}}$ and $\hat{m} \in (\lceil M/2 \rceil, \lceil M/2+1 \rceil)$. Evaluating for $M = 5$ leads to $\hat{m} \approx 3.69$, $\mathrm{L}_{\mathrm{th,m}} \approx 1.67$. We note that although the definition of m and $\hat{m}$ must be integer values, fractional amounts can be effectively obtained by choosing between the floor and ceiling values with the appropriate regularity. In this setting, the threshold estimator is not truly constant, but the estimator does not depend on the input. The threshold is shown in Figure 4 by the dashed line and provides a metric for the maximum expected error if the noise and quantization effects are negligible.

Evaluating the estimators, we first start with the simplest consideration: can the estimators perform better than a blind estimator? Most importantly, all of the estimators fail to reach the threshold for SNR < -5dB, showing that the noise and quantization effects cause the algorithms to perform worse than if the data $\mathbf{X}$ was ignored. Additionally, the one-bit detection module is never able to surpass the threshold, suggesting that the multiple sinusoid detection algorithm is not able to learn meaningful results from one-bit data. The three bit results are able to improve upon the threshold for SNR > -4dB, and perform better than higher resolution versions of AIC and MDL across the entire SNR range. Further, it can be observed that extremely low SNR data with greater variation (higher bit resolution) results in worse estimators.

We can see that the neural network algorithms show a significant advantage over the traditional algorithms. It might be assumed this is due to the choice of loss function, which is different than the benchmark algorithms are designed for. This is not the case, however, as we show the results using simple mean-squared-error in Figure 12 in the supplementary results. While our algorithm's performance is not entirely due to the choice of loss function, we have already seen that the learning threshold is quite low, so we know that a simple estimator that estimates $m = 3, 4$ with the correct regularity will produce effective results.

B. Frequency estimation

Next, we evaluate the frequency estimation performance of the estimation module. The distribution of the output vector

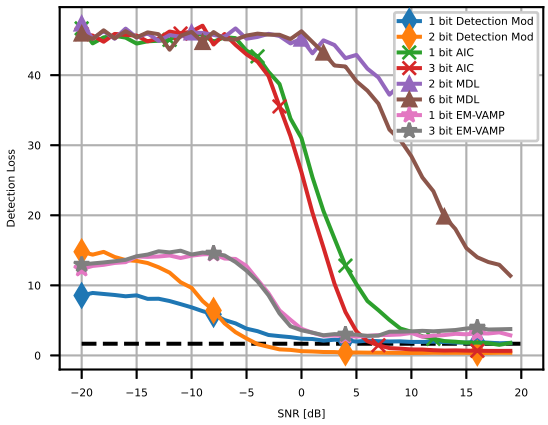


Fig. 4. Our detection module compared to EM-VAMP, MDL and AIC with the proposed detection loss metric. We do not show the results of 1-bit data resolution for MDL and AIC, due to a lack of convergence, which cause the detection loss to be too high and makes it harder to see the differences between the other performance results. Instead, we show the 2 and 3 bit results for comparison. It can be seen that the one-bit detection module is unable to surpass the learning threshold. The threshold is also far below either MDL or AIC until 7dB, showing the inherent advantage data driven techniques have, even for relatively uninformative distributions like a discrete uniform.

$\mathbf{f}$ comes from both the initial w_0 and the offsets, w_i. While the compared algorithms must search for each point with no prior, the actual frequency distributions have structure that can be exploited for better estimation at low SNR. This is one reason why the Periodogram estimator performs significantly worse than the neural architecture in Figure 5, and does not reach the threshold for SNR < -5dB. Following Algorithm 1, the learning threshold is calculated for each number of sinusoidal components shown by assuming m is known and computing the expected mean squared error. The estimator, $g_\theta(\mathbf{X})$ is simply the mean estimator because the loss function is the mean squared error, exactly as shown in Section VI.

First, we note the first and second moments of the folded normal distribution, used for the frequency components beyond the initial one and approximated for our simulation settings

$$\mathbb{E}[w_i] = \sqrt{\frac{5}{N\pi}} \approx 0.1577 \quad i \in \{1, \ldots m\} \tag{39}$$

$$\sigma^2_{w_i} = \frac{2.5}{N} - E[w_i]^2 = \frac{5}{2N}(1 - \frac{2}{\pi}) \approx 0.0142. \tag{40}$$

Based on these statistics and similar measures associated with uniform variables, we can get the threshold and constant vector estimator for the frequency estimation results

$$g_\theta(\mathbf{X}) = \left[\mathbb{E}[w_0], \ldots, \mathbb{E}[w_0] + \frac{m-1}{N} + \mathbb{E}[w_m] \right] \tag{41}$$

$$= \left[0.125, \ \ldots, \ 0.125 + \frac{(m-1)}{N} + \sqrt{\frac{5}{N\pi}} \right]. \tag{42}$$

The independence of w_i is used in (41) to separate the expectations. Because $g_\theta(\mathbf{X})$ is an unbiased estimator, the threshold only depends the variance of the random vector f_i.

We use $\sigma^2()$ to refer to the variance of a random variable, so the threshold is

$$\mathrm{L}_{\mathrm{th},\mathbf{f}}(m) = \frac{1}{m}\sum_{i=1}^{m}\sigma^2(f_i) \tag{43}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\frac{1}{64} + \frac{1}{m}\sum_{i=2}^{m}\frac{5}{2N}\left(1-\frac{2}{\pi}\right) \tag{44}$$

$$= \frac{1}{64} + \left(1-\frac{1}{m}\right)\frac{5}{2N}\left(1-\frac{2}{\pi}\right). \tag{45}$$

Here we average over all samples and outputs for multiple output mean squared error to produce a scalar loss value. This results in $\mathrm{L}_{\mathrm{th},\mathbf{f}} \approx [-18\text{dB}, -16.4\text{dB}, -16\text{dB}, -15.8\text{dB}, -15.7\text{dB}]$ for each value of $1 \leq m \leq M$. While there is potential for the max-cutoff at $f = 0.5$ to cause the distributions to be heavy-tailed (see e.g. Fig. 11), this does not occur in expectation, so we do not include it in the simplified learning threshold. It can be seen, however, that it does have an effect on the distributions for $m = \{4, 5\}$ in Figure 11 within the supplementary materials.

Our results show that, especially for $m \in \{2, 3\}$, our algorithm consistently outperforms the Periodogram by $3-8$dB, and is only surpassed for the $m = 1$ case with SNR > -4dB. Given more data, it is likely that our algorithm would reach or surpass the Periodogram consistently, however we only train on a small dataset to focus on the insights and learning of our architecture. Our results are especially interesting, because the potential gain of using our algorithm for two- or three-sinusoid signals is an order of magnitude better performance.

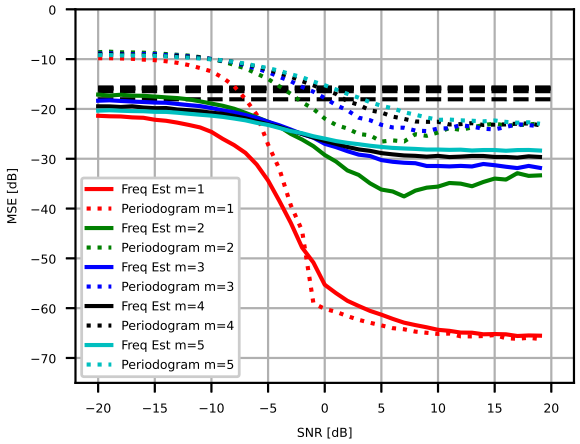
C. Amplitude estimation

Next, we consider the amplitude estimates of our algorithm. While the frequency estimates are resolvable for any bit resolution with sufficiently many samples N and reasonable spacing, the amplitude estimates can be particularly challenging near critical frequencies [25]. For example, consider the following: a single sinusoid with normalized frequency of $f = 0.25 \pm \epsilon$, where $\epsilon < 1/N$, $\phi = 0.1$, and one-bit quantization. In this setting, every subset of $1/f = 4$ points are identical and only contain $\{1+j, -1+j, -1-j, 1-j\}$. Thus the received sequence, without noise, will be $N/4$ repetitions of that sequence, which makes accurate amplitudes estimation infeasible. While this is true for frequency estimation as well, the repetition improves the frequency estimate, and as $N \to \infty$ the frequency estimate will converge to f. There is no similar guarantee for the amplitude estimates, because regardless of the amplitude, the observation sequence is still the same. Even with infinite SNR very little amplitude information is recoverable from such limited data. Nevertheless, the learning threshold does not depend on the input data, so we can still analyze the expected worst case non-adversarial results. Because a_i are independent, the threshold is simply

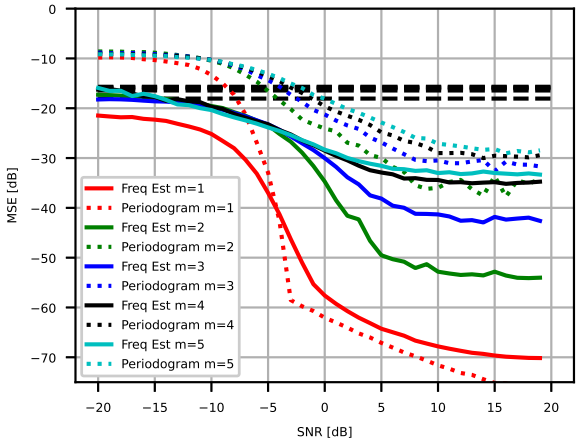
$$\mathbb{E}[a_i] = 0.55$$

$$\mathrm{L}_{\mathrm{th},\mathbf{a}}(m) = \frac{1}{m}\sum_{i=1}^{m}\sigma^2(a_i)$$

$$= \sigma^2(a_1) = 0.0675.$$



(a) one-bit

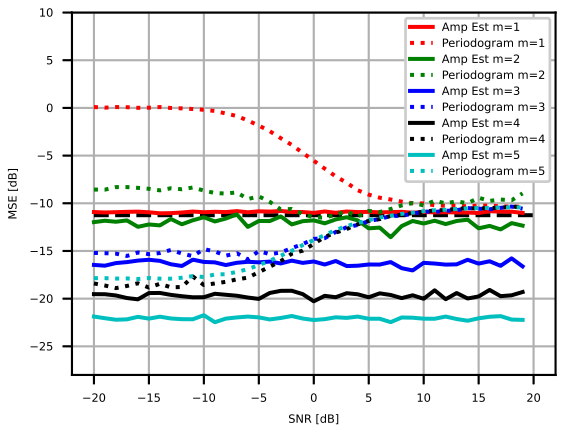


(b) three-bit

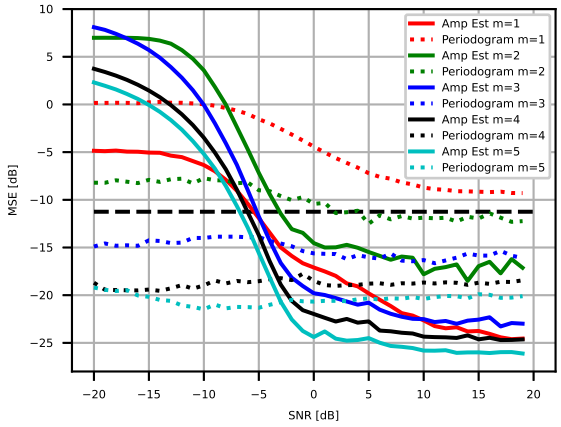
Fig. 5. Two plots of the results of our sinusoidal estimation module's frequency outputs (Freq Mod –) for a given m. Our estimator outperforms the threshold universally and the periodogram (. . .) except in the $m = 1$ case for SNR ≥ -4dB. For $m > 1$, our estimator is consistently better than the periodogram. In (a) the Periodogram results in worsening performance at high SNR, which we also observed in our past work with one-bit quantization [28]. Our results only show the performance loss for $m = 2$. The plot in (b) no longer shows regressing performance and the increased data resolution especially improves the $m = 2, 3$ cases, where our estimator improves upon the Periodogram by 11-15dB.

While the amplitude learning threshold is on a similar scale to the frequency learning threshold, actually achieving this value is challenging for low resolution due to quantization effects directly impacting the amplitude of the signal. We will again rely on the Periodogram as a benchmark for amplitude estimation.

From Figure 6 we obtain insight into what is happening between the one- and three-bit versions. In 6a most of the estimators are not learning any meaningful features from the input and are simply minimizing the loss over the distribution, based on the lack of improvement with increasing SNR. In 6b, the estimators perform worse initially because of the quantization and noise effects, but outperform the threshold for SNR > -3dB. Thus our three-bit results are much more useful and generalizable than the one-bit case, even if the performance



(a) one-bit



(b) three-bit

Fig. 6. A comparison of our sinusoid estimator's amplitude results (Amp Est –). Unexpectedly, our estimator appears to perform better for one-bit data than three-bit at low SNR. Identifying the learning threshold, however, shows the Periodogram and our $m = \{1, 2\}$ estimators converging to a similar level in (a). In contrast, all of our estimators surpass the threshold for three-bit data in (b).

is initially worse for low SNR. From these plots in particular, we can see the value of defining the learning threshold when evaluating deep learning algorithms. Interestingly, increasing the number of sinusoids to $m = \{3, 4, 5\}$ leads to better results, even though the true amplitudes are independent. We believe this is because as m increases, the average error tends toward the arithmetic mean of each the amplitude estimates, but in the case of $m = \{2, 3\}$, if the minimum separation is not sufficient, the estimation error is dominated by the overlapping components.

D. Phase estimation

In the final evaluation of the sinusoid estimator, we look at the phase estimates. We begin in a similar fashion to the amplitude and phase results by first solving for the learning threshold. Similar to the amplitude distribution, the phases are uniformly distributed, $\phi_i \in [0, 2\pi]$, so the constant estimator and learning threshold are simply

$$\mathbb{E}[\phi_i] = \pi$$

$$\begin{aligned} \mathrm{L}_{\mathrm{th},\boldsymbol{\phi}}(m) =& \frac{1}{m}\sum_{i=1}^{m}\sigma^2(\phi_i) \\ =& \sigma^2(\phi_1) = \frac{\pi^2}{3}. \end{aligned}$$

From the learning thresholds, we expect the phase error to be significantly higher than the other estimator losses, which is why we scale the sum training loss according to the learning thresholds in Section IV. This should help ensure none of the losses affect the model significantly more than the others, based on the simulation setup.

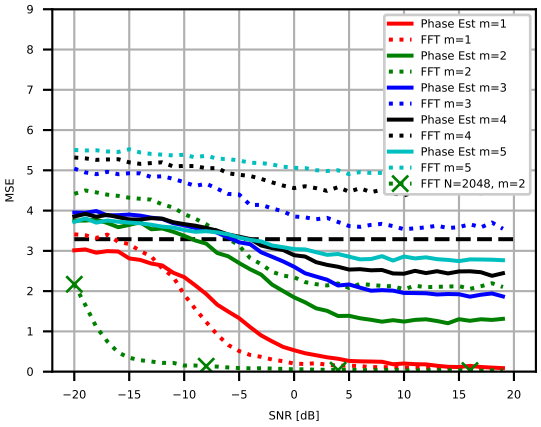
As mentioned in Section V, the Periodogram is replaced with the FFT as a benchmark for estimating the phase of a signal. We show the results of this algorithm with ours in Figure 7, this time without dB scaling. We can see that the one-bit results in Fig. 7a are better than the amplitude estimates, in the sense that every estimator is able to learn and improve with increasing SNR. Similar to the results from Figure 5b, only the $m = 1$ case shows the benchmark (FFT) performing better than our estimator. In the three-bit results, Fig. 7b, we see that the FFT results are largely unchanged from the one-bit version, but our estimator now shows a noticeable improvement with increasing SNR. We also show the case where we increase $N \to 2048$ for the two-sinusoid FFT to show that sample length is the limiting factor.

E. SignalNet results

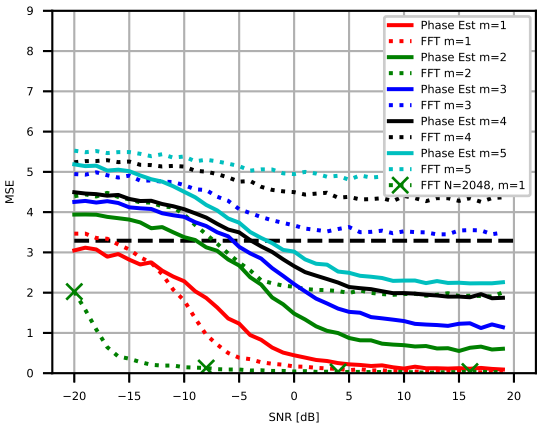
In the final results, we join the two modules, and consider pairings with different traditional estimators. We no longer show the learning threshold, instead comparing across different combinations of algorithms to see how effective our algorithm is in the overall performance. We start by demonstrating the effects of normalization on SignalNet. As we previously found, the phase estimates tend to have much higher error, simply because the phase values are drawn from a distribution with a higher variance. Because the Chamfer loss, assuming the correct number of sinusoids ($m = \hat{m}$), is just two times the mean absolute error, we can approximate the thresholds by the root mean squared error, which is an upper bound on the mean absolute error. Thus, we normalize by the square root of the learning thresholds from Section VIII.

After normalizing, we evaluate benchmarks using AIC with the Periodogram/FFT against SignalNet. AIC is chosen because of its success over MDL in Figure 4. We interchange AIC with our detection module as well and interchange the Periodogram/FFT with our sinusoidal estimator to identify which components provide the most gain in Figure 8. We color in the two regions corresponding to the gain from using our sinusoidal estimator (red \\ shading) and our detection module (blue // shading).

The final results in Fig. 8, show the sinusoid estimation module providing the most noticeable gain. We can also see that the combination is not uniform: switching to our detection module from AIC provides significant gain, but so does switching to our sinusoid estimator from the Periodogram/FFT. This is because our sinusoid estimator has learned to fit the best estimates for a given number of sinusoids, so even with poor detection results, it still has reasonable Chamfer loss.



(a) one-bit



(b) three-bit

Fig. 7. A comparison of the estimation performance of our estimator and the FFT for one-bit data and three-bit data. The chart in (a) shows all of the estimators achieving and passing the threshold beyond SNR $= -5$dB, however most of the estimators have only a slight improvement over the threshold. In contrast, our estimator in (b) is able to successfully surpass the threshold for SNR > -3dB for every value of m. Interestingly, the FFT does not noticeably improve with the data resolution, but does improve with longer sequences.

This explains the asymmetric gain from adding or subtracting one of our modules between the AIC+FFT curve and our SignalNet curve. Recall however, that the learning threshold normalization has been applied to the estimation module, but not to the detection module, as there is no intuitive way to apply that with the Chamfer loss definition. Based on the results from the one-bit detection module in Fig. 4, a significant portion of the gain being seen by the detection module can be attributed to learning the distribution, rather than learning that generalizes beyond our simulation. Ultimately, our SignalNet architecture appears to provide a valuable improvement over other techniques, when compared in our simulation setup. However, we cannot give conclusive statements until we also evaluate how our algorithm generalizes to different data distributions.

F. Generalizability

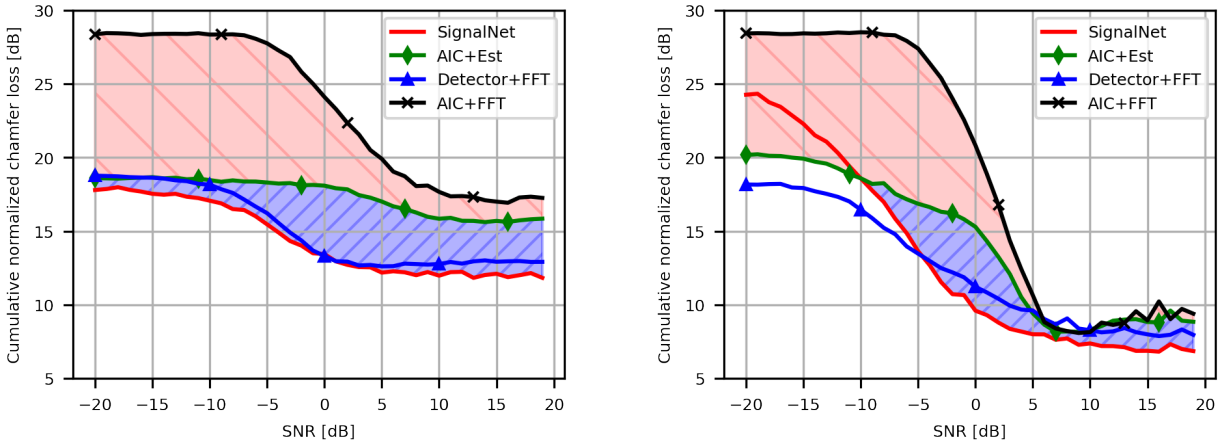
In our final results, we evaluate the out-of-distribution (OOD) results of our algorithm, determined from uniformly distributed frequency content. Given the structure of our generation process in Algorithm 1, we wish to also have a measure of the performance on more general data settings. As we mentioned in Section IV, it is crucial that we generate frequency content that is separated by at least $1/N$, but also should not be equally spaced. In this section, we generate the sinusoidal frequencies from a uniform distribution, with the restriction that no frequency content can be within $1/N$ of any other. If there is insufficient spacing, we regenerate an entirely new set. For each value of m, we draw frequencies from a uniform distribution with the same domain as our original estimator, to ensure that the learned domain remains consistent.

We show the results of all three estimators for the $m = 2$ case, which is the distribution furthest from a uniform distribution, and therefore experiences the greatest distribution shift. We can determine which case has the greatest distribution shift through a variety of metrics, but it is most intuitive to compare the distributions in Fig. 11 with a uniform distribution over the same domain in the supplementary material. We can see from the results in Figure 9, that there is almost no performance impact due to the change in distribution. Based on the ability of our neural network to generalize and outperform the benchmarks, we can conclude that the SignalNet architecture is a valuable improvement over other techniques for the detection and estimation task.

IX. Conclusion

In this paper, we described and evaluated SignalNet, a novel deep neural network for multi-sinusoidal decomposition from low resolution sampling. While no other work has considered quantized, multi-sinusoidal decomposition, our network follows traditional setups by dividing the problem into the subtasks of detection and estimation. We describe a distinct architecture for each subtask and specifically focus on developing a novel estimation network. Our estimation network uses internal reconstruction to explicitly learn the input-output relationship. We saw the benefit that instilling this domain knowledge provided to our estimator, allowing it to efficiently learn features from the data in most settings. While our results strictly consider the case of estimating all sinusoidal parameters, in the case that a specific parameter is not necessary or is known a priori, a constant vector can be inserted in its place during reconstruction.

We also proposed a theoretical tool for comparing our networks and normalizing across distributions by defining a learning threshold. The threshold is used to express the distribution of the output variable for a specific loss function. We used the insight from the learning threshold to understand why the performance of our model did not universally increase going from the one-bit to the three-bit versions and often had better low SNR performance with one-bit data–by not learning meaningful features. This prevented the estimator from performing well, but also made it robust to noise. We



(a) one-bit

(b) three-bit

Fig. 8. The resulting chamfer loss for each of the algorithms with one-bit data (a) and three-bit data (b). We color two regions: one region to show the gain from our detector (blue //) and the other to show the gain from our sinusoid estimator (red \\). At low SNR, the estimator provides significant gains, in part from distributional learning, while at high SNR, the detector provides a larger gain due to the limitations of low resolution sampling for AIC (Fig. 4).

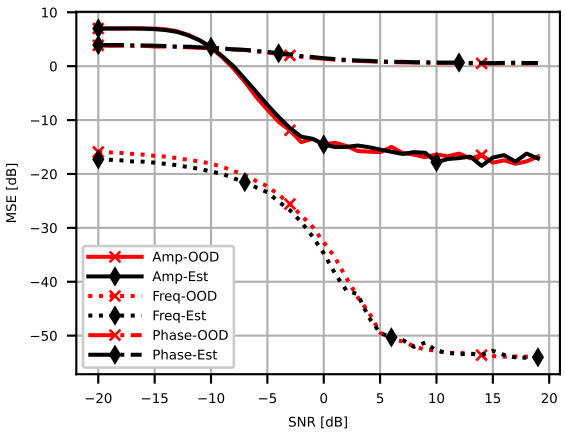


Fig. 9. A comparison of our proposed network testing with the training distribution (black) and out-of-distribution samples (red) for the $m = 2$ case. Frequency content is drawn from a uniform distribution with the same domain as the training data. The results are almost identical, showing that the features learned by our model are effective, even under distribution shift.

found that our learning threshold is a useful metric to consider when comparing neural networks with broader algorithms and determining the success and generalizability of learning algorithms.

Finally, we combined our networks together to complete the SignalNet architecture. Our unified network is able to surpass the benchmark algorithms universally. We were able to directly quantify the improvement from our sinusoid estimator which provided between 2-10dB improvement. We conclude that a domain-aware detection module could improve the results further, as well as additional investigation into appropriate loss functions. Our results suggest that multi-sinusoid estimation can be performed even with extremely low resolution quantization using our SignalNet architecture.

References

[1] L. G. Beatty, J. D. George, and A. Z. Robinson, "Use of the complex exponential expansion as a signal representation for underwater acoustic calibration," *The Journal of the Acoustical Society of America*, vol. 63, no. 6, pp. 1782–1794, 1978.

[2] A. Ghasemi and E. S. Sousa, "Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs," *IEEE Communications Magazine*, vol. 46, no. 4, pp. 32–39, 2008.

[3] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, 1969.

[4] R. W. Heath *et al.*, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. Select. Areas Commun.*, vol. 10, no. 3, pp. 436–453, April 2016.

[5] J. Liu, Z. Luo, and X. Xiong, "Low-resolution ADCs for wireless communication: A comprehensive survey," *IEEE Access*, vol. 7, pp. 91 291–91 324, 2019.

[6] J. Mo and R. W. Heath, "Capacity analysis of one-bit quantized MIMO systems with transmitter channel state information," *IEEE Trans. Signal Processing*, vol. 63, pp. 5498–5512, 2015.

[7] S. Jacobsson *et al.*, "Throughput analysis of massive MIMO uplink with low-resolution ADCs," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 4038–4051, 2017.

[8] Y. Zhou and W. Yu, "Optimized backhaul compression for uplink cloud radio access network," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1295–1307, 2014.

[9] J. J. Bussgang, "Crosscorrelation functions of amplitude-distorted gaussian signals," 1952.

[10] O. T. Demir and E. Bjornson, "The Bussgang decomposition of nonlinear systems: Basic theory and MIMO extensions," *IEEE Signal Processing Magazine*, vol. 38, no. 1, pp. 131–136, 2021.

[11] A. Fredriksen, D. Middleton, and V. VandeLinde, "Simultaneous signal detection and estimation under multiple hypotheses," *IEEE Trans. Inform. Theory*, vol. 18, no. 5, pp. 607–614, 1972.

[12] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, no. 2, pp. 387–392, 1985.

[13] M. Wax and I. Ziskind, "Detection of the number of coherent signals by the MDL principle," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 8, pp. 1190–1196, 1989.

[14] P. Stoica, R. L. Moses, B. Friedlander, and T. Soderstrom, "Maximum likelihood estimation of the parameters of multiple sinusoids from noisy measurements," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 3, pp. 378–392, 1989.

[15] P. M. Djuric, "Simultaneous detection and frequency estimation of sinusoidal signals," in *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, 1993, pp. 53–56.

[16] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, pp. 465–471, 1978.

[17] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Proc. of International Symposium on Information Theory (ISIT)*, 1973, pp. 267–281.

[18] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propagat.*, vol. 34, no. 3, pp. 276–280, March 1986.

[19] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, 1st ed. John Wiley and Sons, Inc., 1996.

[20] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. of the National Academy of Sciences*, vol. 106, no. 45, pp. 18 914–18 919, 2009.

[21] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. of International Symposium on Information Theory (ISIT)*, 2011.

[22] A. K. Fletcher and P. Schniter, "Learning and free energies for vector approximate message passing," in *Proc. of International Conference on Acoustics, Speech and Signal Processing*, 2017.

[23] G. Izacard, S. Mohan, and C. Fernandez-Granda, "Data-driven estimation of sinusoid frequencies," in *Proc. of Adv. Neural Inf. Process. Syst.*, Jun. 2019, pp. 5127–5137.

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer New York Inc., 2001.

[25] A. Høst-Madsen and P. Händel, "Effects of sampling and quantization on single-tone frequency estimation," *IEEE Trans. Signal Processing*, vol. 48, no. 3, pp. 650–662, 2000.

[26] O. Dabeer and A. Karnik, "Signal parameter estimation using 1-bit dithered quantization," *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5389–5405, Dec. 2006.

[27] C. Gianelli, L. Xu, J. Li, and P. Stoica, "One-bit compressive sampling with time-varying thresholds: Maximum likelihood and the Cramér-Rao bound," in *Proc. of Asilomar Conf. on Signals, Systems and Computers*, 2017, pp. 399–403.

[28] R. M. Dreifuerst, R. W. Heath, M. N. Kulkarni, and J. Zhang, "Deep learning-based carrier frequency offset estimation with one-bit ADCs," in *Proc. of Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5.

[29] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[30] A. Wadhwa and U. Madhow, "Blind phase/frequency synchronization with low-precision ADC: A bayesian approach," in *Proc. of Allerton Conference on Communication, Control, and Computing (Allerton)*, 2013, pp. 181–188.

[31] M. Abdizadeh, J. T. Curran, and G. Lachapelle, "New decision variables for GNSS acquisition in the presence of CW interference," *IEEE Trans. Aerosp. Electron. Syst.*, 2014.

[32] G. Zeitler, G. Kramer, and A. C. Singer, "Bayesian parameter estimation using single-bit dithered quantization," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2713–2726, jun 2012.

[33] J. Mo, P. Schniter, and R. W. Heath, "Channel estimation in broadband millimeter wave MIMO systems with few-bit ADCs," in *IEEE Trans. Signal Process.*, vol. 66, no. 5, Mar. 2018.

[34] J. Bassey, L. Qian, and X. Li, "A survey of complex-valued neural networks," 2021.

[35] G. Borgefors, "Distance transformations in arbitrary dimensions," *Computer Vision, Graphics, and Image Processing*, vol. 27, pp. 321–345, 1984.

[36] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3447–3457, 2008.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[38] T. Cover, "Broadcast channels," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 2–14, 1972.

[39] B. Fleury *et al.*, "Channel parameter estimation in mobile radio environments using the sage algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 3, 1999.

[40] E. J. Candès and C. Fernandez-Granda, "Towards a mathematical theory of super-resolution," *Communications on Pure and Applied Mathematics*, vol. 67, no. 6, pp. 906–956, 2014.

[41] A. Moitra, "Super-resolution, extremal functions and the condition number of vandermonde matrices," in *Proc. of ACM Symposium on Theory of Computing*, 2015, p. 821–830.

[42] W. Liao and A. Fannjiang, "Music for single-snapshot spectral estimation: Stability and super-resolution," 2014.

[43] G. Schwarz, "Estimating the Dimension of a Model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461 – 464, 1978.

[44] M. Y. Takeda, A. Klautau, A. Mezghani, and R. W. Heath, "MIMO channel estimation with non-ideal ADCS: Deep learning versus GAMP," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2019.

[45] D. Xu *et al.*, "Survey on multi-output learning," *IEEE Trans. Neural Networks*, vol. 31, no. 7, pp. 2409–2429, 2020.

Ryan Dreifuerst Ryan Dreifuerst is a PhD student at North Carolina State University. He obtained his B.S. in Electrical Engineering from Milwaukee School of Engineering with minors in Mathematics and Physics. He also received his B.S. in Electrical and Communications Engineering from Technische Hochschule Lübeck. He received his M.S in Electrical Engineering at The University of Texas at Austin (UT Austin). Ryan has worked with industry partners on advancing the Open Radio Access Network (ORAN) initiative and spent the last two summers researching 5G and 6G innovations with Qualcomm and Samsung Research America. His research lies at the intersection of machine learning and signal processing. Specifically, Ryan has focused on augmenting machine learning with domain knowledge for physical layer processing in wireless communications and signal processing.

Robert W. Heath Jr. (S'96 - M'01 - SM'06 - F'11) received the B.S. and M.S. degrees from the University of Virginia, Charlottesville, VA, in 1996 and 1997 respectively, and the Ph.D. from Stanford University, Stanford, CA, in 2002, all in electrical engineering. From 1998 to 2001, he was a Senior Member of the Technical Staff then a Senior Consultant at Iospan Wireless Inc, San Jose, CA where he worked on the design and implementation of the physical and link layers of the first commercial MIMO-OFDM communication system. From 2002-2020 he was with The University of Texas at Austin, most recently as Cockrell Family Regents Chair in Engineering and Director of UT SAVES. He is presently the Lampe Distinguished Professor at North Carolina State University and co-founder of 6GNC. He is also President and CEO of MIMO Wireless Inc. He authored "Introduction to Wireless Digital Communication" (Prentice Hall, 2017) and "Digital Wireless Communication: Physical Layer Exploration Lab Using the NI USRP" (National Technology and Science Press, 2012), and co-authored "Millimeter Wave Wireless Communications" (Prentice Hall, 2014) and "Foundations of MIMO Communication" (Cambridge University Press, 2018). Dr. Heath has been a co-author of a number award winning conference and journal papers including recently the 2017 Marconi Prize Paper Award, the 2019 IEEE Communications Society Stephen O. Rice Prize, the 2020 IEEE Signal Processing Society Overview Paper Award and the 2021 IEEE Vehicular Technology Society Neal Shepherd Memorial Best Propagation Paper Award. Other notable awards include the 2017 EURASIP Technical Achievement award, the 2019 IEEE Kiyo Tomiyasu Award and the 2021 IEEE Vehicular Technology Society James Evans Avant Garde Award. In 2017, he was selected as a Fellow of the National Academy of Inventors. He is a member-at-large on the IEEE Communications Society Board-of-Governors (2020-2022) and is a past member-at-large on the IEEE Signal Processing Society Board-of-Governors (2016-2018). He was Editor-in-Chief of IEEE Signal Processing Magazine from 2018-2020. He is also a licensed Amateur Radio Operator, a Private Pilot, a registered Professional Engineer in Texas.