Signal Processing using Machine Learning Techniques

Naman Gupta

BT/EE/220686

Supervisor: Prof. Rohit Budhiraja

Undergraduate Project Report
April 24, 2025

Abstract

This project delves into the critical signal processing challenges presented by Cell-Free (CF) massive Multiple-Input Multiple-Output (mMIMO) systems, particularly when tailored for massive Machine-Type Communication (mMTC) applications. Such environments involve numerous distributed access points cooperatively serving a potentially vast number of devices, many exhibiting sporadic activity and operating under power constraints. The inherent complexity arising from large-scale interference, potential channel uncertainties, and the sheer number of devices necessitates advanced signal detection techniques. We frame the detection problem within a Bayesian probabilistic framework, aiming to infer the transmitted signals or user activity by calculating posterior distributions, which naturally accommodates the system's inherent uncertainties and noise characteristics.

Our investigation begins by tackling the fundamental task of joint channel estimation and data detection, specifically within the context of an underdetermined Cell-Free massive MIMO setting. This challenging scenario arises when the number of users or transmitted symbols surpasses the effective information capacity provided by the receiving antennas, a condition often intensified by hardware constraints like low-resolution quantization which further limits the available information per measurement, resulting in more unknowns than effective equations. To navigate this, assuming initially that user activity is known or all potential users are actively transmitting, we first explore a Support Vector Machine (SVM) based method, framing the detection problem through the lens of binary classification. Building upon this, we then examine Partitioned Variational Inference (PVI), recognizing its strengths as a decentralized, probabilistic framework well-suited to the distributed architecture of CF systems and capable of iterative refinement via local computations. Finally, we investigate the capabilities of Automatic Differentiation Variational Inference (ADVI), a sophisticated approach that leverages flexible variational approximations guided by efficient gradient computations through automatic differentiation to more accurately capture the complex underlying posterior distributions.

Subsequently, we tackle the more intricate and practical mMTC problem, where user activity is unknown and sporadic, necessitating joint Active User Detection (AUD), Channel Estimation (CE), and eventual Data Detection (DD). This investigation explicitly considers the significant hardware constraint of employing 1-bit Analog-to-Digital Converters (ADCs) at the receivers. These introduce severe non-linear distortion, making conventional linear estimation methods ineffective. For this challenging joint task under harsh quantization, we propose and adapt the ADVI framework, leveraging its ability to model complex, non-Gaussian posteriors arising from the combination of sparsity (inactive users) and the non-linear quantization effects.

The primary contribution of this work lies in demonstrating the effectiveness of the ADVI-based approach for robust signal processing in these demanding scenarios. Implemented in PyTorch and evaluated on synthetic datasets mimicking CF-mMIMO mMTC conditions with 1-bit ADCs, the proposed ADVI solution consistently achieves superior detection and estimation accuracy (lower Bit Error Rate and Normalized Mean Squared Error) compared to the baseline methods across various Signal-to-Noise Ratios. It proves particularly adept at handling the system's underdetermined nature, the sparsity induced by mMTC traffic patterns, and the severe non-linearities imposed by 1-bit quantization, offering a unified and powerful machine learning-based solution for next-generation massive access systems.

Table of Contents

Chapter 1:
Massive MIMO (mMIMO)

Massive multiple-input multiple-output (mMIMO) is a transformative wireless communication technology that leverages a large number of antennas at the base station to serve multiple users simultaneously. However, one of the challenges that arises in mMIMO systems is dealing with underdetermined systems, where the number of antennas exceeds the number of users or transmitted symbols. In such cases, solving for the transmitted symbols becomes more complex because there are more unknowns than equations, making the system harder to resolve. This becomes much harder to solve when we introduce a 1-bit ADC. Furthermore, it becomes particularly challenging in noisy environments, where accurate symbol detection is critical. In this chapter, we start with Support Vector Machine (SVM) based algorithm for 'Joint Channel Estimation and Data Detection'. We also implement a decentralized algorithm - 'Partitioned Variational Inference'. Later we propose an algorithm based on 'Automatic Variational Variational Inference' which gives results better than many available algorithms. We will end the chapter by comparing results of some of the already developed algorithms with the ones we propose.

1.1 System Model

We consider a cell-free (CF) massive MIMO (mMIMO) system with M single-antenna users and an N-antenna base station, where $N \geq M$ is assumed. The transmitted signal vector $\mathbf{x}^\top = [x_1, x_2, \ldots, x_M]^\top \in \mathbb{C}^M$ consists of the signals transmitted by the users. Each user's signal x_m is subject to the power constraint $\mathbb{E}[|x_m|^2] = 1$ for all $m \in \{1, 2, \ldots, M\}$.

Let $\mathbf{H} \in \mathbb{C}^{N \times K}$ denote the channel matrix, where for now, we assume block-flat fading. To enable accurate channel estimation, the users transmit pilot sequences, which are typically known at the base station. Denote the pilot sequence matrix as $\mathbf{X}_p \in \mathbb{C}^{M \times T_p}$, where each user m transmits a T_p-length pilot sequence during the channel estimation phase. The pilot sequences are used by the base station to estimate the channels to the users.

The received signal matrix $\mathbf{Y}_p$ during the pilot phase can be modeled as:

$$\mathbf{Y}_p = \text{sign}(\mathbf{H}\mathbf{X}_p + \mathbf{N}_p) \tag{1}$$

where $\mathbf{X}_p = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]^\top$. Once the pilot sequences are transmitted and the channels are estimated, the users switch to transmitting data. The data transmission is performed using data sequences, denoted as $\mathbf{X}_d \in \mathbb{C}^{M \times T_d}$, where each user transmits a T_d-length data sequence during the data phase. Let $\mathbf{X}_d = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]^\top \in \mathbb{C}^{M \times T_d}$ denote the data signal matrix transmitted by the M users.

The total received signal at all the APs can now be modeled as:

$$\mathbf{Y}_d = \text{sign}(\mathbf{H}\mathbf{X}_d + \mathbf{N}_d) \tag{2}$$

where $\mathbf{H} = [\mathbf{H}_1; \mathbf{H}_2; \ldots; \mathbf{H}_L] \in \mathbb{C}^{N \times M}$ is the complete channel matrix, and $\mathbf{N}_d$ is the complete noise matrix, where each column of $\mathbf{N}_d$ which is distributed as $\mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I}_N)$.

Thus, the system model in (2) becomes underdetermined when $N \ll M$. In upcoming subsections, we discuss three different algorithms: (i) SVM-based algorithm, (ii) PVI-based algorithm and (iii) ADVI-based algorithm, for data detection in the above-defined under defined system model, leveraging the channel estimates obtained using pilot sequences and the data transmitted during the data phase.

1.2 SVM Based Algorithm

Support Vector Machines (SVMs), renowned for their robustness in binary classification tasks, can be effectively adapted for joint channel estimation and data detection (CE-DD) in cell-free massive MIMO systems equipped with 1-bit ADCs. In this approach, the CE-DD problem is reformulated as a series of binary classification problems, leveraging the margin-maximizing property of SVMs to perform inference. We now describe the SVM-based method in detail.

1.2.1 Transformation to Real for Channel Estimation

Since SVM solvers operate in the real domain, we first convert the original complex-valued system into an equivalent real-valued formulation. Given a pilot sequence $\tilde{\mathbf{X}}_p$ of length T_p, the training data is generated as:

$$\tilde{\mathbf{Y}}_p = \text{sign}(\tilde{\mathbf{H}}\tilde{\mathbf{X}}_p + \tilde{\mathbf{Z}}_p) \tag{3}$$

The transformation to the real domain proceeds by separating the real and imaginary parts and stacking them appropriately:

$$\mathbf{Y}_p = \left[\Re\{\tilde{\mathbf{Y}}_p\}, \Im\{\tilde{\mathbf{Y}}_p\}\right], \tag{4}$$

$$\mathbf{X}_p = \begin{bmatrix} \Re\{\tilde{\mathbf{X}}_p\} & \Im\{\tilde{\mathbf{X}}_p\} \\ -\Im\{\tilde{\mathbf{X}}_p\} & \Re\{\tilde{\mathbf{X}}_p\} \end{bmatrix}, \tag{5}$$

$$\mathbf{N}_p = \left[\Re\{\tilde{\mathbf{N}}_p\}, \Im\{\tilde{\mathbf{N}}_p\}\right], \tag{6}$$

$$\mathbf{H} = \left[\Re\{\tilde{\mathbf{H}}\}, \Im\{\tilde{\mathbf{H}}\}\right]. \tag{7}$$

Thus, the real-valued system model becomes:

$$\mathbf{Y}_p = \text{sign}(\mathbf{H}\mathbf{X}_p + \mathbf{N}_p) \tag{8}$$

enabling the application of standard SVM classifiers.

1.2.2 Channel Estimation

The objective is to independently estimate each row of the real-valued channel matrix $\mathbf{H}$ by posing a binary classification problem. For each receive antenna i, we solve the following soft-margin SVM optimization:

$$\begin{aligned} \min_{\mathbf{h}_i, \{\xi_n\}} \quad & \frac{1}{2}\|\mathbf{h}_i\|^2 + C\sum_{n=1}^{T_p} \xi_n \\ \text{s.t.} \quad & y_{i,n}(\mathbf{h}_i^\top \mathbf{x}_n) \geq 1 - \xi_n, \quad \xi_n \geq 0, \end{aligned} \tag{9}$$

where $\mathbf{h}_i^\top$ is the i-th row of $\mathbf{H}$, $\mathbf{x}_n$ is the n-th pilot vector (column of $\mathbf{X}_p$), and $y_{i,n} \in \{-1, +1\}$ is the 1-bit quantized received label at antenna i and time n. The slack variables ξ_n accommodate classification errors, and C controls the regularization strength.

After optimization, the estimated channel vector is normalized to satisfy power constraints:

$$\widehat{\mathbf{h}}_i = \sqrt{K} \cdot \frac{\tilde{\mathbf{h}}_i}{\|\tilde{\mathbf{h}}_i\|} \tag{10}$$

where $\tilde{\mathbf{h}}_i$ is the raw SVM output before normalization. Finally, stacking the recovered rows and applying the inverse of the transformation in (7) yields the complex-valued channel estimate $\widehat{\mathbf{H}} \in \mathbb{C}^{N \times M}$.

1.2.3 Transformation to Real for Data Detection

For data detection, a slightly different real-valued transformation is adopted. The complex domain model during data transmission is:

$$\bar{\mathbf{Y}}_d = \text{sign}(\widehat{\mathbf{H}}\bar{\mathbf{X}}_d + \bar{\mathbf{N}}_d) \tag{11}$$

The corresponding real-domain transformations are:

$$\mathbf{Y}_d = \begin{bmatrix} \Re\{\bar{\mathbf{Y}}_d\} \\ \Im\{\bar{\mathbf{Y}}_d\} \end{bmatrix}, \quad \mathbf{N}_d = \begin{bmatrix} \Re\{\bar{\mathbf{N}}_d\} \\ \Im\{\bar{\mathbf{N}}_d\} \end{bmatrix}, \tag{12}$$

$$\mathbf{X}_d = \begin{bmatrix} \Re\{\bar{\mathbf{X}}_d\} \\ \Im\{\bar{\mathbf{X}}_d\} \end{bmatrix}, \quad \mathbf{H}_d = \begin{bmatrix} \Re\{\widehat{\mathbf{H}}\} & -\Im\{\widehat{\mathbf{H}}\} \\ \Im\{\widehat{\mathbf{H}}\} & \Re\{\widehat{\mathbf{H}}\} \end{bmatrix}. \tag{13}$$

This ensures that the model $\mathbf{Y}_d = \text{sign}(\mathbf{H}_d\mathbf{X}_d + \mathbf{N}_d)$ aligns with standard real-valued SVM formulations.

1.2.4 Data Detection

Given the estimated real-valued channel matrix $\mathbf{H}_d$, the transmitted data vectors are detected by solving a real-valued SVM problem for each time instant m:

$$\min_{\mathbf{x}_m, \{\xi_i\}} \quad \frac{1}{2}\|\mathbf{x}_m\|^2 + C\sum_{i'=1}^{2N} \xi_i \tag{14}$$

$$\text{s.t.} \quad y_{i,m}\left(\hat{\mathbf{h}}_i^\top \mathbf{x}_m\right) \geq 1 - \xi_i, \quad \xi_i \geq 0, \tag{15}$$

where $\mathbf{x}_m$ is the m-th column of $\mathbf{X}_d$, $y_{i,m}$ is the i'-th entry of the m-th received vector $\mathbf{Y}_d$, and $\hat{\mathbf{h}}_i^\top$ is the i'-th row of $\mathbf{H}_d$.

After solving, $\mathbf{x}_m$ is normalized and mapped to the nearest modulation constellation point (e.g., QAM symbols), completing the detection process.

1.2.5 Iterative Refinement

A key advantage of the SVM-based framework is its ability to iteratively refine the channel estimates. After initial data detection, the detected symbols can be treated as pseudo-pilots and incorporated into the channel estimation process. This pseudo-labeling approach enhances the quality of channel estimates without requiring additional pilot overhead.

The feedback loop alternates between data detection and channel refinement until convergence criteria (e.g., stability of channel estimates) are met, significantly improving performance at moderate and high SNR regimes.

While the SVM-based approach offers a simple and computationally attractive solution for CE-DD, it falls short in performance under low SNR regimes and highly underdetermined conditions. Moreover, it does not naturally extend to decentralized setups.

To address these limitations, we now explore Partitioned Variational Inference (PVI), a probabilistic, decentralized inference method that better suits large-scale and federated mMIMO systems.

1.3 Partitioned Variational Inference Based Algorithm

Partitioned Variational Inference (PVI) provides a probabilistic framework for federated learning, where a central server interacts with multiple clients, each possessing a different subset of the data. In PVI, each client independently updates the posterior distribution of a global random variable, based on its local data, without sharing raw data with others. This ensures both data privacy and decentralization of the training process. We adapt PVI to the signal detection setting where each client observes a subset of the channel, while all transmit the same signal to the server.

1.3.1 Generalized Algorithm

The system equation be given as:

$$\mathbf{y} = \text{sign}(\mathbf{H}\mathbf{x} + \mathbf{n}) \tag{16}$$

where $\mathbf{y}$ is the received signal vector of dimension $(N \times 1)$, $\mathbf{H}$ is a matrix of dimension $(N \times M)$, $\mathbf{x}$ is the unknown vector of dimension $(M \times 1)$, $\mathbf{n}$ is the AWGN noise vector with variance N_0, and each client has access to exactly (N/K) rows of $\mathbf{H}$, where K is the number of clients.

The channel entries are modeled as i.i.d. $\mathcal{N}(0, 1)$. Each client works with a partial observation of $\mathbf{H}$ but the same transmitted $\mathbf{x}$.

The PVI algorithm, in the above setup, proceeds iteratively between the server and the clients. It operates under three possible modes: sequential, synchronous, and asynchronous updates.

The server maintains a global approximate posterior $q(\theta)$ over the random variable θ (e.g., $\mathbf{x}$). The algorithm steps are:

1. **Client Selection:** The server selects a set of clients b_i whose local likelihood approximations need refinement. The current global posterior of θ, ie. $q^{(i-1)}(\theta)$ is sent to the selected clients.

2. **Client Update:** Each selected client k updates its local approximate likelihood $t_k(\theta)$ by solving:

$$q_k^{(i)}(\theta) = \underset{q(\theta)\in\mathcal{Q}}{\text{argmax}} \int q(\theta) \log\left(\frac{q^{(i-1)}(\theta)p(y_k|\theta)}{q_k^{(i-1)}(\theta)}\right) d\theta \tag{17}$$

   Subsequently, it computes the update factor $\Delta_k^{(i)}(\theta)$ and sends it to the server:

$$\Delta_k^{(i)}(\theta) \propto \frac{t_k^{(i)}(\theta)}{t_k^{(i-1)}(\theta)} \tag{18}$$

3. **Server Aggregation:** The server updates the global posterior using the received updates:

$$q^{(i)}(\theta) \propto q^{(i-1)}(\theta) \prod_{k\in b_i} \Delta_k^{(i)}(\theta) \tag{19}$$

This iterative procedure continues until convergence criteria are met.

1.3.2 Modes of Operation

Three variations of the update mechanism are possible:

- **Sequential:** A single client is selected and updated at each iteration.
- **Synchronous:** All clients update their approximate likelihoods simultaneously in each iteration.
- **Asynchronous:** Clients update whenever they become available, enabling flexibility in computation.

1.3.3 Application to Signal Detection

In our setup, we apply the PVI framework to the signal detection problem in CF-mMIMO systems under the following transformations:

1. **System Transformation to Real Domain:** The complex-valued system model is first transformed into an equivalent real-valued representation by stacking the real and imaginary parts. This simplifies the probabilistic modeling and inference steps.
2. **Channel Estimation (Row-wise):** Each client estimates its portion of the channel matrix $\mathbf{H}$ based only on its locally observed pilots. Since each client sees a disjoint subset of antennas, channel estimation is performed row-wise.
3. **Channel Normalization (Transformation):** After estimation, each client's local channel is normalized to facilitate stable detection performance. This involves normalizing each row of $\mathbf{H}$ individually.
4. **Data Detection (Column-wise):** Given the normalized partial channel observations and the quantized received signals, clients perform local approximate inference to detect the transmitted symbols. Since the signal vector $\mathbf{x}$ is common across all clients, detection operates column wise across all transmit symbols.
5. **Iterative Improvement:** The local updates are communicated back to the server, which aggregates them to refine the global posterior. This iterative refinement continues over multiple PVI rounds, progressively improving the accuracy of the detected signal.

This partitioned approach ensures decentralized operation, data privacy, and efficient scalability to large system dimensions.

Although PVI significantly improves over SVM by offering a decentralized and probabilistic framework, its modeling assumptions and inference granularity are relatively coarse.

To further enhance detection performance, especially in challenging noise and quantization settings, we propose leveraging Automatic Differentiation Variational Inference (ADVI) — a more flexible and scalable approach capable of approximating complex posteriors via automatic differentiation.

1.4 ADVI Based Algorithm

In highly under-determined Cell-Free massive MIMO (CF-mMIMO) systems with one-bit analog-to-digital converters (ADCs), traditional inference techniques struggle with the joint channel estimation and data detection (CE-DD) problem due to non-linear and quantized measurements. Automatic Differentiation Variational Inference (ADVI) offers a scalable and flexible alternative by approximating the posterior distribution of the latent variables using variational inference combined with automatic differentiation. In this section, we describe the generalized ADVI algorithm and its application to CE-DD.

1.4.1 Generalized ADVI Algorithm

We consider a real-valued vector $\mathbf{u} \in \mathbb{R}^M$ for which we aim to estimate the posterior distribution given an observed vector $\mathbf{v}$ and a known matrix $\mathbf{W} \in \mathbb{R}^{N \times M}$. We know, $\mathbf{W}_{i \in 1:N}$ is distributed as $\mathcal{N}(\mathbf{0}, \mathbf{I}_M)$. We define the relation between $\mathbf{v}$ and $\mathbf{u}$ as -

$$\mathbf{v} = \text{sign}(\mathbf{W}\mathbf{u} + \boldsymbol{\varepsilon}) \tag{20}$$

where $\mathbf{v} \in \{-1, +1\}^N$ and the noise vector $\boldsymbol{\varepsilon} \in \mathbb{R}^N$ is distributed as $\mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_N)$.

We now propose the use of Automatic Differentiation Variational Inference (ADVI) algorithm in the above setting.

- ***Prior and Likelihood***

  We assume a Gaussian prior on the unknown vector,

  $$p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_M) \tag{21}$$

  $\mathbf{w}_n^\top = \mathbf{W}_{n,:}$ and so the likelihood for each observation $\mathbf{v}_n$ given $\mathbf{u}$ can be approximated by the sigmoid distribution as

  $$p(\mathrm{v}_n|\mathbf{u}) = \sigma(v_n \mathbf{w}_n^\top \mathbf{u}) \tag{22}$$

  where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function. The joint likelihood is -

  $$p(\mathbf{v}|\mathbf{u}) = \prod_{n=1}^{N} p(\mathbf{v}_n|\mathbf{u}) \tag{23}$$

- ***Posterior Inference via Variational Approximation***

The posterior distribution of interest is given by

$$p(\mathbf{u}|\mathbf{v}) = \frac{p(\mathbf{v}|\mathbf{u})p(\mathbf{u})}{p(\mathbf{v})} \tag{24}$$

Due to the non-conjugacy of the prior and the likelihood, an analytical solution is intractable. We thus resort to variational inference by approximating the true posterior with a tractable distribution $q_{\boldsymbol{\xi}}(\mathbf{u})$. We adopt a mean-field Gaussian approximation:

$$q_{\boldsymbol{\xi}}(\mathbf{u}) = \prod_{m=1}^{M} \mathcal{N}(\mathbf{u}_m|\mu_m, \sigma_m^2) \tag{25}$$

where $\boldsymbol{\xi} = \{\mu_1, ..., \mu_M, \sigma_1, ..., \sigma_M\}$.

To get the optimum posterior distribution $q_{\boldsymbol{\xi}}(\mathbf{u})$, we can minimize the KL divergence between the approximate posterior $q_{\boldsymbol{\xi}}(\mathbf{u})$ and the true posterior $p(\mathbf{u}|\mathbf{v})$. We can write this KL divergence as -

$$\begin{aligned}
&\mathrm{KL}[q_{\boldsymbol{\xi}}(\mathbf{u})||p(\mathbf{u}|\mathbf{v})] = \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}\left[\log \frac{q_{\boldsymbol{\xi}}(\mathbf{u})}{p(\mathbf{u}|\mathbf{v})}\right] \\
&= \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[\log q_{\boldsymbol{\xi}}(\mathbf{u}) - \log p(\mathbf{u}, \mathbf{v}) + \log p(\mathbf{v})] \\
&\overset{(a)}{=} \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[\log q_{\boldsymbol{\xi}}(\mathbf{u}) - \log p(\mathbf{u}, \mathbf{v})] + \log p(\mathbf{v}) \\
&\overset{(b)}{=} -\mathcal{L}(\boldsymbol{\xi}) + \log p(\mathbf{v})
\end{aligned} \tag{26}$$

where equality (a) is obtained by taking $p(\mathbf{v})$ out of the expectation, since it is independent of $q_{\boldsymbol{\xi}}(\mathbf{u})$. Also, we can see clearly from (26) that minimizing KL divergence is equivalent to maximizing $\mathcal{L}(\boldsymbol{\xi})$, the Evidence Lower Bound (ELBO). From equality (b) of (26) -

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{\xi}) = \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[\log p(\mathbf{v}, \mathbf{u}) - \log q_{\boldsymbol{\xi}}(\mathbf{u})] \\
&= \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[\log p(\mathbf{v}, \mathbf{u})] + \mathbb{H}(q_{\boldsymbol{\xi}}(\mathbf{u})) \\
&= \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[\log p(\mathbf{v}|\mathbf{u})] + \mathbb{E}_{q_{\boldsymbol{\xi}}(\mathbf{u})}[p(\mathbf{u})] + \mathbb{H}(q_{\boldsymbol{\xi}}(\mathbf{u}))
\end{aligned} \tag{27}$$

where $\mathbb{H}(q)$ is the entropy of $q(\mathbf{u})$ and because we have mean-field gaussian it can be expressed as -

$$\begin{aligned}
&\mathbb{H}(q_{\boldsymbol{\xi}}(\mathbf{u})) = \sum_{m=1}^{M} \mathbb{E}_{q_{\boldsymbol{\xi}}(u_m)}[-\log q_{\boldsymbol{\xi}}(u_m)] \\
&= \sum_{m=1}^{M} -\mathbb{E}[\log \mathcal{N}(\mu_m, \sigma_m)] \\
&= \sum_{m=1}^{M} -\mathbb{E}\left[\log(2\pi\sigma_m^2)^{-1/2} \exp\left\{\frac{-(u_m - \mu_m)^2}{2\sigma_m^2}\right\}\right] \\
&= \sum_{m=1}^{M} \frac{1}{2}(\log(2\pi\sigma_m^2)) + \frac{1}{2\sigma_m^2}\mathbb{E}[(u_m - \mu_m)^2] \\
&= \frac{M}{2}[1 + \log(2\pi)] + \sum_{m=1}^{M} \log(\sigma_m)
\end{aligned} \tag{28}$$

because, $\mathbb{E}[(u_m - \mu_m)^2] = \sigma_m^2$.

- ***Transforming parameters to being unconstrained***

  So far, we have variational parameters $\boldsymbol{\xi} \in \{\mathbb{R}^M, \mathbb{R}^M_{>0}\}$. In order to optimize the ELBO in the real coordinate space without the need to explicitly enforce positivity constraint during optimization, we transform the mean-field Gaussian variance as -

  $$\omega_m = \log(\sigma_m) \tag{29}$$

  The optimization problem now becomes -

  $$\boldsymbol{\phi}^* = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}\, \mathcal{L}(\boldsymbol{\phi}) \tag{30}$$

  where $\boldsymbol{\phi} = \{\mu_1, ..., \mu_M, \omega_1, ..., \omega_M\}$ and $\boldsymbol{\phi} \in \mathbb{R}^{2M}$.

- ***Gradient Ascent***

  Now the task at hand is to solve (30). Expanding the likelihood term of $\mathcal{L}(\boldsymbol{\phi})$ in (27) we get,

  $$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{u})}[\log p(\mathbf{v}|\mathbf{u})] = \sum_{n=1}^{N} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{u})}[\log \sigma(v_n \mathbf{w}_n^\top \mathbf{u})] \tag{31}$$

  Since expectation of sigmoid with respect to gaussian doesn't have closed form, we use gradient ascent to maximize $\mathcal{L}(\boldsymbol{\phi})$. This involves iteratively updating the parameters in the direction of the gradient. The general update rule for parameter $\boldsymbol{\phi}$ at iteration 'i' is given by -

  $$\boldsymbol{\phi}^{(i+1)} \leftarrow \boldsymbol{\phi}^{(i)} + \boldsymbol{\alpha}\nabla_{\boldsymbol{\phi}}\mathrm{ELBO}(\boldsymbol{\phi}^{(i)}) \tag{32}$$

  where α is the learning rate. To apply this update, we must compute the gradient of the ELBO, $\mathcal{L}(\boldsymbol{\phi})$ in our case, with respect to $\boldsymbol{\phi}$ -

  $$\nabla_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi}) = \nabla_{\boldsymbol{\phi}}\{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{u})}[\log p(\mathbf{v}, \mathbf{u})] + \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{u}))\} \tag{33}$$

  We leverage automatic differentiation (AD) to compute these gradients efficiently. AD decomposes functions into elementary operations and applies the chain rule algorithmically, allowing for precise and scalable derivative computations. However, direct differentiation w.r.t. $\boldsymbol{\phi}$ is intractable due to the expectation also being over $q_\phi(\mathbf{u})$, ie. dependent on $\boldsymbol{\phi}$. Thus to apply AD, we push the gradient inside the expectation, which is valid if $q_\phi(\mathbf{u})$ is reparameterizable.

- ***Reparameterization Trick and Computing Gradients***

  In order to make the expectation in (33) independent of $\boldsymbol{\phi}$, we do the following re-parameterization. Let

  $$\boldsymbol{\eta}_m := S_{\boldsymbol{\phi}}(\mathbf{u}) = \frac{\mathbf{u}_m - \mu_m}{e^{\omega_m}} \quad \forall m = 1, ..., M \tag{34}$$

So, we can write the posterior on u_m as -

$$\implies q_{\phi_m}(u_m) = \frac{1}{\sqrt{2\pi\sigma_m^2}} \exp\left(\frac{1}{2}\left(\frac{u_m - \mu_m}{e^{\omega_m}}\right)^2\right)$$
$$\implies q_{\phi_m}(u_m) = e^{-\omega_m} q(\eta_m) \tag{35}$$

where $\eta_m \sim \mathcal{N}(0,1)$. And so,

$$q_{\boldsymbol{\phi}}(\mathbf{u}) = \text{diag}(e^{-\boldsymbol{\omega}}) q(\boldsymbol{\eta}) \tag{36}$$

From (34), we can write -

$$\implies d\eta_m = e^{-\omega_m} du_m$$
$$\implies d\boldsymbol{\eta} = \text{diag}(e^{-\boldsymbol{\omega}}) d\mathbf{u} \tag{37}$$

since, for $i \neq j, du_i/du_j = 0$
We can also express $\mathbf{u}$ as -

$$\mathbf{u} := S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}) = \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta} + \boldsymbol{\mu} \tag{38}$$

which gives us -

$$\frac{d\mathbf{u}}{d\boldsymbol{\mu}} = \mathbf{I}_M \quad \text{and} \quad \frac{d\mathbf{u}}{d\boldsymbol{\omega}} = \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}}) \tag{39}$$

Thus, we can transform the expectation in $\mathcal{L}(\boldsymbol{\phi})$ as -

$$\mathbb{E}_{q_\phi(\mathbf{u})}[\log p(\mathbf{v},\mathbf{u})] = \int q_{\boldsymbol{\phi}}(\mathbf{u}) \log(p(\mathbf{v},\mathbf{u})) d\mathbf{u}$$
$$\overset{(a)}{=} \int \left(\text{diag}(e^{-\omega_m}) q(\boldsymbol{\eta})\right) \log(p(\mathbf{v},\mathbf{u})) d\mathbf{u}$$
$$= \int q(\boldsymbol{\eta}) \log[p(\mathbf{v}, S^{-1}(\boldsymbol{\eta}))] \left(\text{diag}(e^{-\omega_m}) d\mathbf{u}\right)$$
$$\overset{(b)}{=} \int q(\boldsymbol{\eta}) \log[p(\mathbf{v}, S^{-1}(\boldsymbol{\eta}))] d\boldsymbol{\eta}$$
$$= \mathbb{E}_{q(\boldsymbol{\eta})}\left[\log p(\mathbf{v}, S^{-1}(\boldsymbol{\eta}))\right] \tag{40}$$

where equality (a) is obtained by substituting value of $q_{\boldsymbol{\phi}}(\mathbf{u})$ as in equation (36). Equality (b) is obtained using equation (37).

And so, we can rewrite (27) as -

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{\mathcal{N}(\eta)}[\log p(\mathbf{v}, S^{-1}(\boldsymbol{\eta}))] + \mathbb{H}[q(S^{-1}(\boldsymbol{\eta}))]$$
$$= \mathbb{E}_{\mathcal{N}(\eta)}[f(\mathbf{u})] + \mathbb{H}[q(\mathbf{u})] \tag{41}$$

where $\mathbf{u} = S^{-1}(\boldsymbol{\eta})$.

Now that expectation depends on $\boldsymbol{\eta}$ only, so we can switch the expectation and gradient -

$$\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\phi}} f(\mathbf{u})] + \nabla_{\boldsymbol{\phi}} \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{u})) \tag{42}$$

From (28), $\nabla_{\boldsymbol{\mu}} \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{u})) = \mathbf{0}$ and $\nabla_{\boldsymbol{\omega}} \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{u})) = \mathbf{1}_M$. So, we can finally write -

$$\nabla_{\boldsymbol{\mu}} \mathcal{L} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \text{log} p(\mathbf{v},\mathbf{u})]$$
$$= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\left(\sum_{n=1}^{N} \sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n\right) - \mathbf{u}\right] \tag{43}$$

(Derivations in Appendix A.)
Similarly,

$$\nabla_{\boldsymbol{\omega}} \mathcal{L} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \text{log} p(\mathbf{v},\mathbf{u})] + \mathbf{1}_M$$
$$= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\left(\sum_{n=1}^{N} [\sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n] - \mathbf{u}\right) \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right] + \mathbf{1}_M \tag{44}$$

(Derivations in Appendix B.)

- ***Monte Carlo Integration***

  We can now use MC integration to find the expectations in (43) and (44). For this, We draw S samples $\{\eta^{(s)}\}_{s=1}^S$ from $\mathcal{N}(\mathbf{0}, \mathbf{I}_M)$. We estimate $\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi})$ by averaging over these S samples as -

$$\nabla_{\boldsymbol{\mu}} \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_{\mathbf{u}} \text{log} p(\mathbf{v}, \mathbf{u}^{(s)}) \tag{45}$$

$$\nabla_{\boldsymbol{\omega}} \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \left[\nabla_{\mathbf{u}} \text{log} p(\mathbf{v},\mathbf{u})\right] \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}}) + \mathbf{1}_M \tag{46}$$

  with $\mathbf{u}^{(s)} = \boldsymbol{\mu} + \text{diag}(\exp(\boldsymbol{\omega})) * \boldsymbol{\eta}$

- ***Updating Parameters***

  So far, we have approximated the gradient of ELBO using 'Automatic Differentiation' and Monte Carlo averaging. So using that, we iteratively update the variational parameters at each step as -

$$\boldsymbol{\phi}^{(t+1)} \leftarrow \boldsymbol{\phi}^{(t)} + \boldsymbol{\alpha} \nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}^{(t)}) \tag{47}$$

  where $\boldsymbol{\alpha}$ is the learning rate and $t = 1, 2, ...$ until convergence. The final posterior estimate of $\mathbf{u}$ is therefore -

$$\mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, diag(exp(\boldsymbol{\omega}))) \tag{48}$$

Algorithm 1 Proposed ADVI Algorithm

Input: $\mathbf{v}, \mathbf{W}$, tolerance$(\tau) = 0.01$, step-size$(\boldsymbol{\alpha})$
Initialize: $\boldsymbol{\mu}^{(0)} = \mathbf{0}, \boldsymbol{\omega}^{(0)} = \mathbf{0}$
while *change in* $\mathcal{L}$ *is above a threshold* **do**
  Draw S samples $\{\eta\}_{s=1}^S$ from the standard multivariate Gaussian.
  Approximate $\nabla_{\boldsymbol{\mu}} \mathcal{L}$ and $\nabla_{\boldsymbol{\omega}} \mathcal{L}$ using MC integration (Equation (45),(46))
  Update $\boldsymbol{\mu}^{(i+1)} \leftarrow \boldsymbol{\mu}^{(i)} + \boldsymbol{\alpha} \nabla_{\boldsymbol{\mu}} \mathcal{L}$
  Update $\boldsymbol{\omega}^{(i+1)} \leftarrow \boldsymbol{\omega}^{(i)} + \boldsymbol{\alpha} \nabla_{\boldsymbol{\omega}} \mathcal{L}$
  Increment i as $i \leftarrow i + 1$
return *Estimated posterior mean of* $\mathbf{u} = \boldsymbol{\mu}$

1.4.2 Channel Estimation

In Cell-Free massive MIMO systems, estimating the channel matrix $\mathbf{H}$ is essential for reliable data recovery. During the training phase, pilot symbols are transmitted from the users, and the access points (APs) receive a noisy and quantized version of the response -

$$\mathbf{Y}_p = \text{sign}(\mathbf{H}\mathbf{X}_p + \text{noise}) \tag{49}$$

To apply the ADVI algorithm, we convert the system to a real-valued domain as in subsection (1.2.1), since the ADVI formulation assumes real variables, and then decouple the problem row-wise:

$$\mathbf{y}_i = \text{sign}(\mathbf{X}_p^\top \mathbf{h}_i + \mathbf{z}_i), \quad \forall i \in \{1, \ldots, N\} \tag{50}$$

Here, each row $\mathbf{h}_i$ of the real-valued channel matrix $\mathbf{H}$ is estimated independently using the generalized ADVI algorithm where $\mathbf{W} = \mathbf{X}_p^\top$, $\mathbf{u} = \mathbf{h}_i$, and $\mathbf{v} = \mathbf{y}_i$.

The motivation behind this decomposition is that the received vector at each antenna (row-wise) can be treated as a separate probabilistic model, making the estimation scalable and parallelizable.

Algorithm 2 Channel Estimation using ADVI

Input: $\mathbf{Y}_p, \mathbf{X}_p, \sigma^2$
for $i \leftarrow 1$ *to* N **do**
  Let $\mathbf{v} \leftarrow \mathbf{y}_i$, $\mathbf{W} \leftarrow \mathbf{X}_p^\top$
  Using inputs $\mathbf{v}$ and $\mathbf{W}$ as inputs to Algorithm 1, let the returned vector is $\mathbf{u}$.
  Normalize $\mathbf{u}$ as $\hat{\mathbf{u}} \leftarrow (\sqrt{M})(\mathbf{u}/||\mathbf{u}||^2)$
  $\bar{\mathbf{H}}[:, i] \leftarrow \hat{\mathbf{u}}$
$\hat{\mathbf{H}} = \bar{\mathbf{H}}[:, :M] + j\bar{\mathbf{H}}[:, M:]$
return *Estimated complex channel matrix* $\hat{\mathbf{H}}$

1.4.3 Data Detection

Once the channel matrix $\hat{\mathbf{H}}$ is estimated, it is used for data detection during the payload phase. The observed signal is:

$$\mathbf{Y}_d = \text{sign}(\mathbf{H}\mathbf{x}_d + \text{noise}) \tag{51}$$

As in channel estimation, we move to the real domain. However, in this case, we treat each column $\mathbf{x}_t$ of the transmitted data matrix as the latent variable, and the rows of $\hat{\mathbf{H}}$ form the design matrix $\mathbf{W}$. Thus, we rewrite the model as:

$$\mathbf{y}_t = \text{sign}(\mathbf{H}\mathbf{x}_t + \mathbf{z}_t) \tag{52}$$

Here, $\mathbf{x}_t$ is detected by solving the ADVI inference problem with $\mathbf{W} = \hat{\mathbf{H}}$, $\mathbf{u} = \mathbf{x}_t$, and $\mathbf{v} = \mathbf{y}_t$.

This process yields soft estimates of the transmitted symbols which are then normalized and quantized to recover the binary data.

1.4.4 Iterative Refinement

While a single round of channel estimation followed by data detection can work, the quantized nature of observations often limits the accuracy of channel estimates. To address this, an iterative refinement loop is employed.

Algorithm 3 Data Detection using ADVI

Input: $\mathbf{Y}_d, \hat{\mathbf{H}}, \sigma^2$
for $t \leftarrow 1$ *to* T_d **do**
  Let $\mathbf{v} \leftarrow \mathbf{y}_t$, $\mathbf{W} \leftarrow \hat{\mathbf{H}}$
  $\mathbf{u} = Algorithm1(\mathbf{v}, \mathbf{W})$
  Normalize $\mathbf{u}$ as $\hat{\mathbf{u}} \leftarrow (\sqrt{M})(\mathbf{u}/||\mathbf{u}||^2)$
  $\bar{\mathbf{X}}[:, t] \leftarrow \hat{\mathbf{u}}$
$\check{\mathbf{X}}_d = \bar{\mathbf{X}}_d[:M, :] + j\bar{\mathbf{X}}_d[M:, :]$
$\hat{\mathbf{X}}_d = \text{sign}(\Re\{\check{\mathbf{X}}_d\}) + j * \text{sign}(\Im\{\check{\mathbf{X}}_d\})$
return *Estimated complex data matrix* $\hat{\mathbf{X}}_d$

After an initial data detection step, the detected data $\hat{\mathbf{x}}_d$ is treated as additional training data. The complete augmented data becomes:

$$\mathbf{X}_r = [\mathbf{X}_p, \hat{\mathbf{X}}_d], \quad \mathbf{Y}_r = [\mathbf{Y}_p, \mathbf{Y}_d] \tag{53}$$

This augmented dataset is used to re-estimate $\mathbf{H}$ using the ADVI-based channel estimation procedure. The process is repeated until the change in $\mathbf{H}$ across iterations falls below a threshold.

Iterative refinement improves the model accuracy and enables performance comparable to systems with higher-resolution ADCs, all while maintaining low training overhead.

1.4.5 Summary of ADVI-Based CE-DD

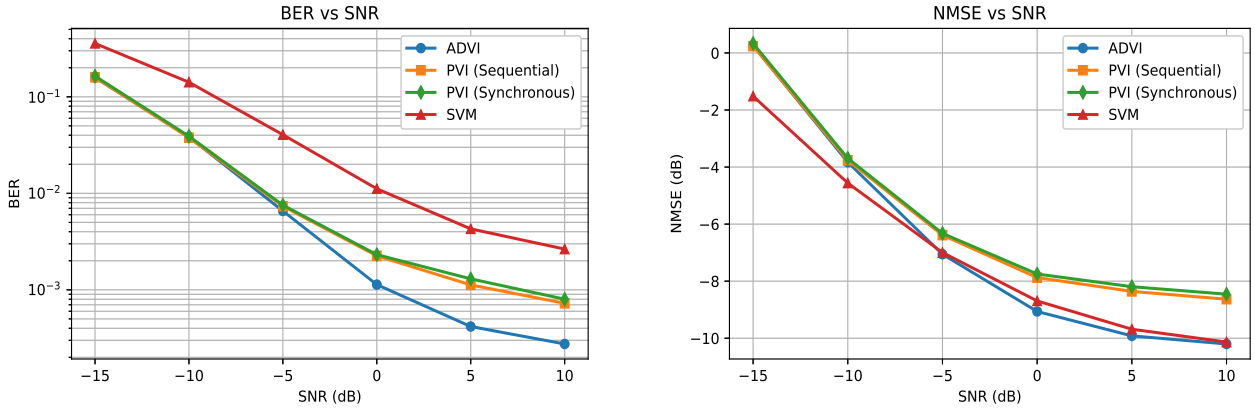
The full CE-DD pipeline using ADVI consists of the following steps:

1. Convert all signals to their real-valued equivalents.
2. Apply ADVI row-wise to estimate the channel matrix from pilot data.
3. Apply ADVI column-wise to detect transmitted data using the estimated channel.
4. Iteratively refine the channel estimate using detected data as pseudo-pilots.
5. Continue until convergence criteria (e.g., channel estimate stability) is met.

Having introduced and described the three algorithms — SVM, PVI, and ADVI — we now present a comparative analysis based on their detection performance and channel estimation accuracy across a wide range of SNR values.

1.5 Results

We compare the performance of the SVM, PVI (Sequential and Synchronous modes), and ADVI algorithms for joint channel estimation and data detection (CE-DD) across a wide range of SNR values from -15 dB to 10 dB. These simulations were conducted using number of receiver antennas $N = 128$, number of users $M = 20$ (all assumed active in this scenario), pilot sequence length $T_p = 80$, and data sequence length $T_d = 200$.



(a) BER vs SNR

(b) NMSE vs SNR

Figure 1: Performance comparison for Joint CE-DD with QAM symbol signals of different algorithms: (a) BER vs SNR and (b) NMSE vs SNR, evaluated from −15dB to 10dB for SVM, PVI (Sequential and Synchronous), and ADVI algorithm.

At low SNRs (e.g., −15 dB), all algorithms experience relatively high BER, with SVM performing the worst at approximately 0.36, while PVI and ADVI achieve considerably lower BER around 0.16. As SNR improves, the performance gap widens. For example, at 10 dB, SVM achieves a BER of 2.6×10^{-3}, whereas PVI (Sequential) and PVI (Synchronous) reduce the BER to approximately 7.2×10^{-4} and 8.0×10^{-4} respectively. ADVI outperforms all, achieving an even lower BER of 3.0×10^{-4} at the same SNR.

A similar trend is observed in NMSE performance. At 10 dB, SVM achieves an NMSE of −10.13 dB, while PVI (Sequential) and PVI (Synchronous) deliver NMSEs around −8.63 dB and −8.45 dB respectively. ADVI again leads, achieving an NMSE of −10.20 dB, indicating more accurate channel estimation and data detection.

Overall, ADVI demonstrates superior performance consistently across all SNRs, especially at moderate and high SNRs. PVI significantly improves over SVM, with the Synchronous mode offering marginally better performance than the Sequential mode. However, SVM, while computationally simpler, consistently underperforms in both BER and NMSE compared to the other two methods.

From the results obtained, it becomes clear that while all methods provide useful trade-offs, ADVI consistently achieves the best performance. We summarize the key takeaways from this comparative study below.

1.6 Conclusion

In this work, we compared three algorithms — SVM, PVI, and ADVI — for the task of joint channel estimation and data detection (CE-DD) in 1-bit quantized Cell-Free massive MIMO systems.

Support Vector Machine (SVM) is the simplest and most computationally efficient method among the three. It requires minimal iterations and achieves fast inference times, making it attractive for extremely low-latency or resource-constrained systems. However, it suffers from poor performance, especially at low and moderate SNRs, and struggles when the system is highly underdetermined.

Partitioned Variational Inference (PVI) offers a significant improvement over SVM by adopting a probabilistic framework. PVI is particularly attractive for decentralized and federated learning scenarios, where full centralization of data or computations is impractical. While its performance does not match ADVI, it provides a good trade-off between accuracy and decentralization. Furthermore, the Synchronous variant of PVI generally converges faster and yields slightly better accuracy than the Sequential mode, making it preferable in settings where coordination among nodes is feasible.

Automatic Differentiation Variational Inference consistently outperforms both SVM and PVI across all SNR regimes. It achieves the lowest Bit Error Rate (BER) and Normalized Mean Squared Error (NMSE), demonstrating strong robustness even under severe quantization and noisy environments. ADVI is particularly effective in highly underdetermined settings where the number of antennas is much smaller than the number of users, owing to its ability to model posterior distributions flexibly. However, the computational overhead associated with variational updates and Monte Carlo sampling must be considered, especially for large-scale deployments.

In conclusion, ADVI emerges as the best-performing algorithm for CE-DD in the considered setting, offering superior accuracy and robustness. PVI remains a strong candidate when decentralization and communication constraints are priorities, while SVM is suitable for extremely fast, lightweight deployments where some performance degradation is acceptable.

Chapter 2:
Massive Access Systems

Massive Access Systems (MAS) are crucial for 5G and future 6G networks, enabling the simultaneous connection of a massive number of devices in a grant-free, random access manner. In 5G, massive machine-type communication (mMTC) allows devices to sporadically transmit data without dedicated scheduling, improving efficiency. With the advent of 6G, the number of connected devices will further increase, making efficient active user detection (AUD) and channel estimation (CE) even more critical. However, the sparse nature of the user activity and variable channel sparsity pose significant challenges in reliably detecting active users and estimating their channels. To reduce system costs, we implement a 1-bit ADC at the receiver. While this simplifies hardware requirements, it makes channel estimation and user detection more difficult, as 1-bit quantization introduces significant distortion. Additionally, the underdetermined nature of the system, where the number of unknowns exceeds the number of equations, exacerbates the challenge, especially with a large number of devices and sporadic activity. We address these challenges by employing an Automatic Variational Inference (ADVI)-based algorithm. This approach effectively handles the sparsity of both the user activity and the channel, using a probabilistic model to perform joint AUD and CE. Our solution outperforms traditional methods, offering robust and accurate results despite the complexities introduced by 1-bit ADC and large-scale device activity.

2.1 System Model

We consider an uplink mMTC system with M single-antenna users and a base station (BS) equipped with N antennas, where only $M_a \ll M$ users are active at any given time. These active users simultaneously transmit pilots (or data) over T_p time slots, and the BS utilizes these signals for active user detection (AUD) and channel estimation (CE).

The received signal at the BS during the t-th time slot is given by:

$$\mathbf{y}_t = \mathbf{H}\,\mathrm{diag}(\alpha)\,\mathbf{x}_t + \mathbf{n}_t, \tag{1}$$

where:

- $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_M] \in \mathbb{C}^{N\times M}$ is the channel matrix, with each $\mathbf{h}_m \in \mathbb{C}^{N\times 1}$ representing the channel from the m-th user to the BS.
- $\mathrm{diag}(\alpha) = \mathrm{diag}(\alpha_1, \alpha_2, \ldots, \alpha_M) \in \{0,1\}^{M\times M}$ is a diagonal matrix that indicates user activity, where $\alpha_m = 1$ if the m-th user is active and $\alpha_m = 0$ if the user is inactive.
- $\mathbf{x}_t = [x_{1,t}, x_{2,t}, \ldots, x_{M,t}]^\top \in \mathbb{C}^{M\times 1}$ is the vector of transmitted signals, where $x_{m,t}$ denotes the pilot (or data symbol) transmitted by the m-th user during the t-th time slot.
- $\mathbf{n}_t \sim \mathcal{CN}(0, \sigma^2 I_N)$ is the additive white Gaussian noise (AWGN) vector at the BS, where σ^2 is the noise power and I_N is the identity matrix of size $N \times N$.

The system utilizes pilot sequences for channel estimation. Let $\mathbf{X}_p \in \mathbb{C}^{M\times T_p}$ be the pilot sequence matrix, where each user m transmits a T_p-length pilot sequence during the channel estimation phase. The pilot sequence matrix is used by the BS to estimate the channels to the active users.

During the data phase, the users transmit data symbols. Let $\mathbf{X}_d \in \mathbb{C}^{M\times T_d}$ represent the data transmission matrix, where each user transmits a T_d-length data sequence during the data phase.

The total received signal at the BS during the pilot phase can be expressed as:

$$\mathbf{Y}_p = \mathbf{H}\,\mathrm{diag}(\alpha)\,\mathbf{X}_p + \mathbf{N}_p$$

where $\mathbf{Y}_p \in \mathbb{C}^{N\times T_p}$ is the received signal matrix during the pilot phase and $\mathbf{N}_p \sim \mathcal{CN}(0, \sigma^2 I_N)$ is the noise matrix corresponding to the pilot phase.

During the data phase, the received signal at the BS is:

$$\mathbf{Y}_d = \mathbf{H}\,\mathrm{diag}(\alpha)\,\mathbf{X}_d + \mathbf{N}_d$$

where $\mathbf{Y}_d \in \mathbb{C}^{N\times T_d}$ is the received signal matrix during the data phase and $\mathbf{N}_d \sim \mathcal{CN}(0, \sigma^2 I_N)$ is the noise matrix corresponding to the data phase.

This system model captures the uplink communication, where the BS receives a mixture of signals from active users, each weighted by its respective channel response. The signal is corrupted by noise, and the BS must estimate the active users and their respective channels based on the received pilot signals and the transmitted data symbols.

We now proceed with the ADVI-based approach in the system model defined, starting with joint channel estimation and activity detection first.

2.2 Activity Detection and Channel Estimation using ADVI

In order to estimate the channel and activity vector, a pilot sequence $\tilde{\mathbf{X}}_p$ of length T_p is used to generate the training data as -

$$\tilde{\mathbf{Y}}_p = \mathrm{sign}(\tilde{\mathbf{H}}(\tilde{\boldsymbol{\alpha}})\tilde{\mathbf{X}}_p + \tilde{\mathbf{Z}}_p) \tag{54}$$

where $\boldsymbol{\alpha} \in \{0,1\}^M$ is the user activity vector, $\tilde{\mathbf{H}} \in \mathbb{R}^{N\times M}$ is the channel matrix, and $\tilde{\mathbf{Y}}_p \in \{-1,+1\}^{N\times T_p}$ is the observed signal matrix, when the pilot matrix $\tilde{\mathbf{X}}_p$ is transmitted over the channel.

We now propose a method for activity detection and channel estimation using the ADVI algorithm.

2.2.1 Transforming the System into Real Space

First, the complex valued matrices $\tilde{\mathbf{Y}}_p, \tilde{\mathbf{X}}_p, \tilde{\mathbf{Z}}_p$ and $\tilde{\mathbf{H}}$ are converted into real valued matrices $\mathbf{Y}_p, \mathbf{X}_p, \mathbf{Z}_p$ and $\mathbf{H}$ respectively as follows -

$$\mathbf{Y}_p = [\Re\{\tilde{\mathbf{Y}}_p\}, \Im\{\tilde{\mathbf{Y}}_p\}] = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N]^\top, \tag{55}$$

$$\mathbf{X}_p = \begin{bmatrix} \Re\{\tilde{\mathbf{X}}_p\} & \Im\{\tilde{\mathbf{X}}_p\} \\ -\Im\{\tilde{\mathbf{X}}_p\} & \Re\{\tilde{\mathbf{X}}_p\} \end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{2T_p}], \tag{56}$$

$$\mathbf{Z}_p = [\Re\{\tilde{\mathbf{Z}}_p\}, \Im\{\tilde{\mathbf{Z}}_p\}] = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]^\top, \text{and} \tag{57}$$

$$\mathbf{H} = [\Re\{\tilde{\mathbf{H}}\}, \Im\{\tilde{\mathbf{H}}\}] = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N]^\top \tag{58}$$

and, $\boldsymbol{\alpha} = [\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\alpha}}]$. Note that $\mathbf{y}_i^\top \in \{\pm 1\}^{1\times 2T_p}$, $\mathbf{h}_i^\top \in \mathbb{R}^{1\times 2M}$ and $\mathbf{z}_i^\top \in \mathbb{R}^{1\times 2T_p}$ with $i \in \{1, 2, \ldots, N\}$ represents rows of $\mathbf{Y}_p$, $\mathbf{H}$ and $\mathbf{Z}_p$ respectively. However, $\mathbf{x}_t \in \mathbb{R}^{2M\times 1}$ with $t \in \{1, 2, \ldots, 2T_p\}$ is the tth column of $\mathbf{X}_p$. So, equation (54) gets transformed to -

$$\mathbf{Y}_p = \mathrm{sign}(\mathbf{H}(\boldsymbol{\alpha})\mathbf{X}_p + \mathbf{Z}_p) \tag{59}$$

where $\mathbf{Y}_p \in \{-1,+1\}^{N\times 2T_p}$ and matrix $\mathbf{X}_p \in \mathbb{R}^{2M\times 2T_p}$. Now, we proceed to predicting the channel matrix $\mathbf{H}$ and activity vector $\boldsymbol{\alpha}$ in the real space.

2.2.2 ADVI algorithm

Using the relation between $\mathbf{Y}_p$ and $\mathbf{X}_p$ as defined in (59), we can also write -

$$\mathbf{y}_n = \mathrm{sign}(\mathbf{h}_n^\top \mathrm{diag}(\boldsymbol{\alpha})\mathbf{X}_p + \mathbf{z}_n) \tag{60}$$

where, $\mathbf{h}_n^\top = \mathbf{H}_{n,:}$, $\mathbf{y}_n = \mathbf{Y}_{n,:}$ and $\mathbf{z}_n = \mathbf{Z}_{n,:}$ be the rows of $\mathbf{H}, \mathbf{Y}_p$ and $\mathbf{Z}_p$.

- ***Prior and Likelihood Distribution***

  We assume a double exponential (Laplace) prior on the unknown activity vector $\boldsymbol{\alpha}$ and gaussian prior on the channel $\mathbf{H}$

$$p(\boldsymbol{\alpha}) = \frac{1}{2}\exp(-|\boldsymbol{\alpha}|) \tag{61}$$

$$p(\mathbf{h}_n) = \mathcal{N}(\mathbf{0}, \mathbf{I}_{2M}), \text{where} \quad \mathbf{h}_n = \mathbf{H}_{n,:} \tag{62}$$

  We can approximate the likelihood distribution using sigmoid as -

$$p(y_{nt}|\boldsymbol{\alpha}, \mathbf{h}_n) = \sigma(y_{nt}\mathbf{h}_n^\top \text{diag}(\boldsymbol{\alpha})\mathbf{x}_t) \tag{63}$$

  where $y_{nt} = (\mathbf{Y}_p)_{n,t}$ and $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function.

- ***Posterior Inference via Variational Approximation***

  The joint posterior distribution of interest is given by

$$p(\boldsymbol{\alpha}, \mathbf{H}|\mathbf{Y}_p) = \frac{p(\mathbf{Y}_p|\boldsymbol{\alpha}, \mathbf{H})p(\boldsymbol{\alpha}, \mathbf{H})}{p(\mathbf{Y}_p)} \tag{64}$$

  Due to the non-conjugacy of the prior and the likelihood, an analytical solution is intractable. We thus resort to variational inference by approximating the true posterior of $\boldsymbol{\alpha}$ and $\mathbf{H}$ with a tractable distributions $q(\boldsymbol{\alpha})$ and $q(\mathbf{H})$. We adopt a mean-field Gaussian approximation:

$$q(\boldsymbol{\alpha}) = \prod_{m=1}^{2M} \mathcal{N}(\alpha_m|\mu_{\alpha_m}, \sigma^2_{\alpha_m}) \tag{65}$$

$$\text{and,} \quad q(\mathbf{H}) = \prod_{n=1}^{N}\prod_{m=1}^{2M} \mathcal{N}(w_{nm}|\mu_{w_{nm}}, \sigma^2_{w_{nm}}) \tag{66}$$

  To get the optimum posterior distribution estimate $q(\boldsymbol{\alpha}, \mathbf{H})$, we can minimize the KL divergence between the approximate joint posterior $q(\boldsymbol{\alpha}, \mathbf{H})$ and the true joint posterior $p(\boldsymbol{\alpha}, \mathbf{H}|\mathbf{Y}_p)$, which is the same as maximizing the Evidence Lower Bound ($\mathcal{L}$).

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\alpha},\mathbf{H})}[\log p(\boldsymbol{\alpha}, \mathbf{H}, \mathbf{Y}_p)] + \mathbb{H}(q(\boldsymbol{\alpha})) + \mathbb{H}(q(\mathbf{H})) \tag{67}$$

  where $\mathbb{H}$ denotes the entropy of the respective distributions. We can write the entropy $\mathbb{H}$ as -

$$\mathbb{H}(q(\boldsymbol{\alpha})) = M[1 + \log(2\pi)] + \sum_{m=1}^{2M} \log(\sigma_{\alpha_m}) \tag{68}$$

$$\mathbb{H}(q(\mathbf{H})) = NM[1 + \log(2\pi)] + \sum_{n=1}^{N}\sum_{m=1}^{2M} \log(\sigma_{\omega_{nm}}) \tag{69}$$

- ***Transforming parameters to being unconstrained***

  So far, we have variational parameters $\{\sigma_{\alpha_m}\}_{m=1}^{2M} \in \mathbb{R}_+^{2M}$ and $\{\sigma_{w_{nm}}\}_{n,m=1}^{N,2M} \in \mathbb{R}_+^{N\times 2M}$. In order to optimize the ELBO($\mathcal{L}$) in the real coordinate space without the need to explicitly enforce positivity constraint during optimization, we transform the mean-field Gaussian variance as -

$$\omega_{\alpha_m} = \log(\sigma_{\alpha_m}); \boldsymbol{\omega}_{w_{nm}} = \log(\sigma_{w_{nm}}) \tag{70}$$

  The optimization problem now becomes -

$$\boldsymbol{\phi}^* = \underset{\boldsymbol{\phi}}{\text{argmax}}\, \mathcal{L}(\boldsymbol{\phi}) \tag{71}$$

  where we define the set of parameters $\boldsymbol{\phi} = \{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\mu}_{\mathbf{h}_1}, \ldots, \boldsymbol{\mu}_{\mathbf{h}_N}, \boldsymbol{\omega}_{\boldsymbol{\alpha}}, \boldsymbol{\omega}_{\mathbf{h}_1}, \ldots, \boldsymbol{\omega}_{\mathbf{h}_N}\}$.

- ***Gradient Ascent***

  Since expectation of sigmoid with respect to gaussian doesn't have closed form, we use gradient ascent to maximize $\mathcal{L}(\boldsymbol{\phi})$. This involves iteratively updating the parameters in the direction of the gradient. The general update rule for parameter $\boldsymbol{\phi}$ at iteration 'i' is given by -

$$\boldsymbol{\phi}^{(i+1)} \leftarrow \boldsymbol{\phi}^{(i)} + \lambda \nabla_{\boldsymbol{\phi}} \text{ELBO}(\boldsymbol{\phi}^{(i)}) \tag{72}$$

  where λ is the learning rate. To apply this update, we must compute the gradient of the ELBO ($\mathcal{L}(\boldsymbol{\phi})$) with respect to $\boldsymbol{\phi}$ -

$$\nabla_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi}) = \nabla_{\boldsymbol{\phi}}\{\mathbb{E}_{q(\boldsymbol{\alpha},\mathbf{H})}[\log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H})] + \mathbb{H}(q(\boldsymbol{\alpha})) + \mathbb{H}(q(\mathbf{H}))\} \tag{73}$$

  We leverage automatic differentiation (AD) to compute these gradients efficiently. AD decomposes functions into elementary operations and applies the chain rule algorithmically, allowing for precise and scalable derivative computations.

  However, direct differentiation w.r.t. $\boldsymbol{\phi}$ is intractable due to the expectation also being over $q(\boldsymbol{\alpha}, \mathbf{H})$, ie. dependent on $\boldsymbol{\phi}$. Thus to apply AD, we push the gradient inside the expectation, which is valid if $q(\boldsymbol{\alpha}, \mathbf{H})$ is reparameterizable.

- ***Reparameterization Trick and Computing Gradients***

  Let $\mathcal{S} = \{\mathcal{S}_n\ ; 0 \le n \le N\}$ be such that $\mathcal{S}_0 = \boldsymbol{\alpha}$ and $\mathcal{S}_n = \mathbf{h}_n\ \forall\ n = 1, 2, \ldots, N$. Now, in order to make the expectation in (73) independent of $\boldsymbol{\phi}$, we do the following re-parameterization. Let,

$$\mathcal{S}_n := \mathcal{F}(\boldsymbol{\eta}_{\mathcal{S}_n}) = \boldsymbol{\mu}_{\mathcal{S}_n} + \boldsymbol{\sigma}_{\mathcal{S}_n}\boldsymbol{\eta}_{\mathcal{S}_n} \tag{74}$$

  such that $\eta_{\mathcal{S}_n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2M})$ for each $n = 0, 1, \ldots, N$.

Thus, we can transform the expectation in $\mathcal{L}(\boldsymbol{\phi})$ as -

$$\mathbb{E}_{q(\mathcal{S})}[\log p(\mathbf{Y}_p, \mathcal{S})] = \mathbb{E}_{q(\boldsymbol{\eta}_S)}\Big[\log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H})\Big] \tag{75}$$

And so, we can rewrite $\mathcal{L}(\boldsymbol{\phi})$ as -

$$\begin{aligned}\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{\eta}_S)}[\log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H}))] \\ + \mathbb{H}[q(\boldsymbol{\alpha})] + \mathbb{H}[q(\mathbf{H})]\end{aligned} \tag{76}$$

Now that expectation depends only on $\boldsymbol{\eta}_S$, we can switch the expectation and gradient -

$$\begin{aligned}\nabla_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta}_S)}[\nabla_{\boldsymbol{\phi}} \log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H}))] \\ + \nabla_{\boldsymbol{\phi}}\mathbb{H}(q(\boldsymbol{\alpha})) + \nabla_{\boldsymbol{\phi}}\mathbb{H}(q(\mathbf{H}))\end{aligned} \tag{77}$$

From (68) and (69),

$$\nabla_{\mu_{\mathcal{S}_n}}\mathbb{H}(q_{\boldsymbol{\phi}}(\mathcal{S}_n)) = \mathbf{0}\ ;\ \nabla_{\boldsymbol{\omega}_{\mathcal{S}_n}}\mathbb{H}(q_{\boldsymbol{\phi}}(\mathcal{S}_n)) = \mathbf{I}_{2M} \tag{78}$$

for $n = 1, 2, ..., N$. So, we can finally write -

$$\nabla_{\boldsymbol{\mu}_{\mathcal{S}_n}}\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta}_S)}[\nabla_{\boldsymbol{\mu}_{\mathcal{S}_n}} \log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H})] \tag{79}$$

And so,

$$\nabla_{\boldsymbol{\mu}_{\mathcal{S}_0}}\mathcal{L}(\boldsymbol{\phi}) = \sum_{n=1}^{N}\sum_{t=1}^{T}\sigma\Big(-y_{nt}\mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \boldsymbol{\alpha} \tag{80}$$

$$\nabla_{\boldsymbol{\mu}_{\mathcal{S}_n}}\mathcal{L}(\boldsymbol{\phi}) = \sum_{n=1}^{N}\sum_{t=1}^{T}\sigma\Big(-y_{nt}\mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \mathbf{h}_n \tag{81}$$

(See Appendix C and D).
Similarly,

$$\nabla_{\boldsymbol{\omega}_{\mathcal{S}_n}}\mathcal{L}(\boldsymbol{\phi}) = \sum_{n=1}^{N}\mathbb{E}_{\mathcal{N}(\boldsymbol{\eta}_S)}[\nabla_{\boldsymbol{\omega}_{\mathcal{S}_n}} \log p(\mathbf{Y}_p, \boldsymbol{\alpha}, \mathbf{H})] + \mathbf{I}_{2M} \tag{82}$$

And so,

$$\begin{aligned}\nabla_{\boldsymbol{\omega}_{\mathcal{S}_0}}\mathcal{L}(\boldsymbol{\phi}) = \sum_{n=1}^{N}\sum_{t=1}^{T}\Big[\sigma\Big(-y_{nt}\mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \boldsymbol{\alpha}\Big] \\ \boldsymbol{\eta}_{S_0}^\top(\boldsymbol{\omega}_{S_0}) + +\mathbf{I}_{2M}\end{aligned} \tag{83}$$

$$\begin{aligned}\nabla_{\boldsymbol{\omega}_{\mathcal{S}_n}}\mathcal{L}(\boldsymbol{\phi}) = \sum_{n=1}^{N}\sum_{t=1}^{T}\Big[\sigma\Big(-y_{nt}\mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \mathbf{h}_n\Big] \\ \boldsymbol{\eta}_{S_n}^\top(\boldsymbol{\omega}_{S_n}) + \mathbf{I}_{2M}\end{aligned} \tag{84}$$

(See Appendix C and E).

- ***Monte Carlo Integration***

  We can now use MC integration to find the expectations in (79) and (82). For this, We draw K samples $\{\boldsymbol{\eta}_{\mathcal{S}_n}^{(s)}\}_{k=1}^{K}$ from $\mathcal{N}(\mathbf{0}, \mathbf{I}_{2M})$ for each $n = 0, 1, 2, ..., N$. We estimate $\nabla_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi})$ by averaging over these K samples as -

$$\nabla_{\boldsymbol{\mu}_{\mathcal{S}_n}}\mathcal{L} \approx \frac{1}{K}\sum_{k=1}^{K}\nabla_{\mathcal{S}_n}[\log p(\mathbf{Y}_p, \mathcal{F}(\mathcal{S}^{(k)}))] \tag{85}$$

$$\begin{aligned}\nabla_{\boldsymbol{\omega}_{\mathcal{S}_n}}\mathcal{L} \approx \Bigg(\frac{1}{K}\sum_{k=1}^{K}\Big[\nabla_{\mathcal{S}_n}[\log p(\mathbf{Y}_p, \mathcal{F}(\mathcal{S}^{(k)}))]\Big] \\ (\boldsymbol{\eta}_{\mathcal{S}_n}^{(k)})^\top \mathrm{diag}(e^{\boldsymbol{\omega}_{\mathcal{S}_n}})\Bigg) + \mathbf{I}_{2M}\end{aligned} \tag{86}$$

  with $\mathcal{S}^{(s)} = \boldsymbol{\mu}_{\mathcal{S}} + \mathrm{diag}(\exp(\boldsymbol{\omega}_{\mathcal{S}})) * \boldsymbol{\eta}_{\mathcal{S}}^{(k)}$

- ***Updating Parameters***

  So far, we have approximated the gradient of ELBO using 'Automatic Differentiation' and Monte Carlo averaging. So using that, we iteratively update the variational parameters at each step as -

$$\boldsymbol{\phi}^{(t+1)} \leftarrow \boldsymbol{\phi}^{(t)} + \lambda\nabla_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi}^{(t)}) \tag{87}$$

  where λ is the learning rate and $t = 1, 2, ...$ until convergence.

  The final posterior estimate of $\boldsymbol{\alpha}$ is therefore -

$$\check{\boldsymbol{\alpha}} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\alpha}}, diag(exp(\boldsymbol{\omega}_{\boldsymbol{\alpha}}))) \tag{88}$$

  and final posterior estimate of $\mathbf{H}$ is -

$$\check{\mathbf{h}}_n = \check{\mathbf{H}}_{n,;} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{h}_n}, diag(exp(\boldsymbol{\omega}_{\mathbf{h}_n}))) \tag{89}$$

2.2.3 Estimated Activity Vector and Channel Matrix

Now, that we've got the posterior estimates for both the activity vector $\boldsymbol{\alpha}$ and the channel matrix $\mathbf{H}$. We use the posterior means of $\check{\boldsymbol{\alpha}}$ to get an ideal estimated activity vector $\hat{\boldsymbol{\alpha}} \in \{0, 1\}^M$ as -

$$\hat{\alpha}_i = \begin{cases} 1, & \text{if } |\check{\alpha}_i| \geq 0.5 \text{ and } |\check{\alpha}_i| = |\check{\alpha}_{M+i}| \\ 0, & \text{if } |\check{\alpha}_i| < 0.5 \text{ and } |\check{\alpha}_i| = |\check{\alpha}_{M+i}| \\ 0, & \text{if and } |\check{\alpha}_i| \neq |\check{\alpha}_{M+i}| \end{cases} \tag{90}$$

for all $1 \leq i \leq M$. We also get the complex channel estimate $\hat{\mathbf{H}} \in \mathbb{C}^{N\times M}$ by reversing the transformation done in (58) as -

$$\hat{\mathbf{H}} = \check{\mathbf{H}}[:, :M] + j\check{\mathbf{H}}[:, M:] \tag{91}$$

In the next section, we use the estimates $\hat{\boldsymbol{\alpha}}$ and $\hat{\mathbf{H}}$ for data detection in the same system model.

Algorithm 4 ADVI-Based Activity Detection and Channel Estimation

Input: $\mathbf{Y}_p, \mathbf{X}_p$, tolerance($\tau$) = 0.01, step-size($\lambda$)
Initialize: $\boldsymbol{\mu}_{\mathcal{S}}^{(0)} = \mathbf{0}, \boldsymbol{\omega}_{\mathcal{S}}^{(0)} = \mathbf{0}$
while *change in* $\mathcal{L}$ *is above a threshold* **do**
  Draw K samples $\{\boldsymbol{\eta}_{\mathcal{S}}\}_{k=1}^{K}$ from the standard multivariate Gaussians.
  Approximate $\nabla_{\boldsymbol{\mu}_{\mathcal{S}}}\mathcal{L}$ and $\nabla_{\boldsymbol{\omega}_{\mathcal{S}}}\mathcal{L}$ using Equation (85) and (86).
  Update $\boldsymbol{\mu}_{\mathcal{S}}^{(i+1)} \leftarrow \boldsymbol{\mu}_{\mathcal{S}}^{(i)} + \lambda\nabla_{\boldsymbol{\mu}_{\mathcal{S}}}\mathcal{L}$
  Update $\boldsymbol{\omega}_{\mathcal{S}}^{(i+1)} \leftarrow \boldsymbol{\omega}_{\mathcal{S}}^{(i)} + \lambda\nabla_{\boldsymbol{\omega}_{\mathcal{S}}}\mathcal{L}$
  Increment i as $i \leftarrow i+1$
Using $\check{\boldsymbol{\alpha}} = \boldsymbol{\mu}_{\mathcal{S}_0}$, get $\hat{\boldsymbol{\alpha}}$, using Equation (90).
$\check{\mathbf{H}}[i,:] = \boldsymbol{\mu}_{\mathcal{S}_n}$, for $1 \le n \le N$.
$\hat{\mathbf{H}} = \check{\mathbf{H}}[:, :M] + j\check{\mathbf{H}}[:, M:]$
return *Estimated activity vector* $\hat{\boldsymbol{\alpha}}$ *and channel matrix* $\hat{\mathbf{H}}$.

2.3 Data Detection using ADVI

Now, that we have successfully done activity detection, we know which of the users are active and transmitting data. Let the set of active users be $\mathcal{U} = \{i\ ;\ \hat{\alpha}_i = 1\} \in \mathbb{R}^{M_a}$, where $M_a << M$. We write the effective channel matrix $\tilde{\mathbf{H}}_a \in \mathbb{C}^{N\times M_a}$ as -

$$\tilde{\mathbf{H}}_a = \hat{\mathbf{H}}_a(\boldsymbol{\alpha}_a) \tag{92}$$

where, $\hat{\mathbf{H}}_a = [\mathbf{h}_{\mathcal{U}_1}, \mathbf{h}_{\mathcal{U}_2}, \ldots, \mathbf{h}_{\mathcal{U}_{M_a}}] \in \mathbb{C}^{N\times M_a}$, $\mathbf{h}_i = \hat{\mathbf{H}}_{:,i}$, and $\boldsymbol{\alpha}_a = [\alpha_{\mathcal{U}_1}, \alpha_{\mathcal{U}_2}, \ldots, \alpha_{\mathcal{U}_{M_a}}] \in \mathbb{C}^{M_a\times 1}$. With the effective channel matrix $\tilde{\mathbf{H}}_a$ in complex domain and the received signal $\tilde{\mathbf{Y}}_d$ known to us, we aim to detect the data matrix $\tilde{\mathbf{X}}_d$ transmitted. Note that $\tilde{\mathbf{X}}_d \in \mathbb{C}^{M_a\times T_d}$ is the transmitted data sequence of length T_d. The received data signal is given as:

$$\tilde{\mathbf{Y}}_d = \text{sign}(\tilde{\mathbf{H}}_a\tilde{\mathbf{X}}_d + \tilde{\mathbf{Z}}_d) \tag{93}$$

where $\tilde{\mathbf{Y}}_d \in \{+1, -1\}^{N\times T_d}$ and $\tilde{\mathbf{Z}}_d \in \mathbb{R}^{N\times T_d}$. To bring this to real domain, we do the following transformation -

$$\mathbf{Y}_d = \begin{bmatrix}\Re\{\tilde{\mathbf{Y}}_d\}\\ \Im\{\tilde{\mathbf{Y}}_d\}\end{bmatrix} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{T_d}], \tag{94}$$

$$\mathbf{X}_d = \begin{bmatrix}\Re\{\tilde{\mathbf{X}}_d\}\\ \Im\{\tilde{\mathbf{X}}_d\}\end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{T_d}], \tag{95}$$

$$\mathbf{Z}_d = \begin{bmatrix}\Re\{\tilde{\mathbf{Z}}_d\}\\ \Im\{\tilde{\mathbf{Z}}_d\}\end{bmatrix} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{T_d}], \tag{96}$$

$$\mathbf{H} = \begin{bmatrix}\Re\{\tilde{\mathbf{H}}_a\} & -\Im\{\tilde{\mathbf{H}}_a\}\\ \Im\{\tilde{\mathbf{H}}_a\} & \Re\{\tilde{\mathbf{H}}_a\}\end{bmatrix} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{2M_a}]^\top \tag{97}$$

Here, $\mathbf{y}_t \in \{\pm 1\}^{2N\times 1}$, $\mathbf{x}_t \in \mathbb{R}^{2M_a\times 1}$ and $\mathbf{z}_t \in \mathbb{R}^{2N\times 1}$ with $t \in \{1, 2, \ldots, T_d\}$ are the columns of $\mathbf{Y}_d, \mathbf{X}_d$, and $\mathbf{Z}_d$ respectively corresponding to the tth time instant. However, $\mathbf{h}_i \in \mathbb{R}^{2M_a\times 1}$ with $i \in \{1, 2, \ldots 2N\}$ represents the ith row of $\mathbf{H}$. Using the above transformation into real space, equation (93) gets transformed to -

$$\begin{aligned}&\implies \mathbf{Y}_d = \text{sign}(\mathbf{H}\mathbf{X}_d + \mathbf{Z}_d)\\ &\implies \mathbf{y}_t = \text{sign}(\mathbf{H}\mathbf{x}_t + \mathbf{z}_t) \qquad \forall t \in \{1, 2, \ldots, T_d\}\end{aligned} \tag{98}$$

We proceed with ADVI to get the posterior estimate for the transmitted data column-wise. We tend to solve the problem of estimating $\mathbf{x}_t$ in -

$$\mathbf{y}_t = \text{sign}(\mathbf{H}\mathbf{x}_t + \mathbf{z}_t) \tag{99}$$

for each of $1 \le t \le T_d$ again using the proposed ADVI algorithm that follows.

2.3.1 ADVI Algorithm

We assume a Gaussian prior on the vector $\mathbf{x}_t$ as -

$$p(\mathbf{x}_t) = \mathcal{N}(\mathbf{0}, \mathbf{I}_{2M_a}) \tag{100}$$

and approximate the likelihood for each observation $\mathbf{y}_t$ given $\mathbf{x}_t$ by the sigmoid distribution as -

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{n=1}^{2N}\begin{cases}\sigma(\mathbf{h}_n^\top\mathbf{x}_t), & \text{for } y_{nt} = 1,\\ 1-\sigma(\mathbf{h}_n^\top\mathbf{x}_t), & \text{for } y_{nt} = -1\end{cases} \tag{101}$$

where $y_{nt} = (\mathbf{Y}_d)_{nt}$, $\mathbf{w}_n^\top = \mathbf{W}_{n,:}$ and $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function.

Now, the posterior distribution of interest is given by -

$$p(\mathbf{x}_t|\mathbf{y}_t) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{y}_t)} \tag{102}$$

Due to the non-conjugacy of the prior and the likelihood, an analytical solution is intractable. We thus resort to variational inference by approximating the true posterior with a tractable distribution $q(\mathbf{x}_t)$. We adopt a mean-field Gaussian approximation:

$$q(\mathbf{x}_t) = \prod_{m=1}^{M}\mathcal{N}(\mathbf{x}_{tm}|\mu_m, \sigma_m^2) \tag{103}$$

We know that minimizing KL divergence is equivalent to maximizing $\mathcal{L}$, the Evidence Lower Bound (ELBO).

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(\mathbf{x}_t)}[\log p(\mathbf{y}_t, \mathbf{x}_t) - \log q(\mathbf{x}_t)]\\ &= \mathbb{E}_{q(\mathbf{x}_t)}[\log p(\mathbf{y}_t, \mathbf{x}_t)] + \mathbb{H}(q(\mathbf{x}_t))\\ &= \mathbb{E}_{q(\mathbf{x}_t)}[\log p(\mathbf{y}_t|\mathbf{x}_t)] + \mathbb{E}_{q(\mathbf{x}_t)}[p(\mathbf{x}_t)] + \mathbb{H}(q(\mathbf{x}_t))\end{aligned} \tag{104}$$

where $\mathbb{H}(q)$ is the entropy of $q(\mathbf{x}_t)$ and because we have mean-field gaussian it can be expressed as -

$$\mathbb{H}(q(\mathbf{x}_t)) = \frac{M}{2}[1 + \log(2\pi)] + \sum_{m=1}^{M}\log(\sigma_m) \tag{105}$$

So far, we have variational parameters $\{\mu_m\}_{m=1}^M \in \mathbb{R}^M$ and $\{\sigma_m\}_{m=1}^M \in \mathbb{R}_{>0}^M$. In order to optimize the ELBO in the real co-ordinate space without the need to explicitly enforce positivity constraint during optimization, we transform the mean-field Gaussian variance $\{\sigma_m\}_{m=1}^M$ as -

$$\omega_m = \log(\sigma_m) \tag{106}$$

The optimization problem now becomes -

$$\boldsymbol{\xi}^* = \underset{\boldsymbol{\xi}}{\text{argmax}}\, \mathcal{L}(\boldsymbol{\xi}) \tag{107}$$

where $\boldsymbol{\xi} = \{\mu_1, ..., \mu_M, \omega_1, ..., \omega_M\}$ and $\boldsymbol{\xi} \in \mathbb{R}^{2M}$. Now the task at hand is to solve (107). Since in the likelihood term of $\mathcal{L}(\boldsymbol{\xi})$, the expectation of sigmoid with respect to Gaussian does not have closed form, we use gradient ascent to maximize $\mathcal{L}(\boldsymbol{\xi})$. This involves iteratively updating the parameters in the direction of the gradient. The general update rule for parameter $\boldsymbol{\xi}$ at iteration 'i' is given by -

$$\boldsymbol{\xi}^{(i+1)} \leftarrow \boldsymbol{\xi}^{(i)} + \lambda \nabla_{\boldsymbol{\xi}} \text{ELBO}(\boldsymbol{\xi}^{(i)}) \tag{108}$$

where λ is the learning rate. To apply this update, we must compute the gradient of the ELBO, $\mathcal{L}(\boldsymbol{\xi})$ in our case, with respect to $\boldsymbol{\xi}$ -

$$\nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi}) = \nabla_{\boldsymbol{\xi}} \{\mathbb{E}_{q(\mathbf{x}_t)}[\log p(\mathbf{y}_t, \mathbf{x}_t)] + \mathbb{H}(q(\mathbf{x}_t))\} \tag{109}$$

In order to make the expectation in (109) independent of $\boldsymbol{\xi}$, we do the following re-parameterization. Let

$$\boldsymbol{\eta}_m := \mathcal{G}(\mathbf{x}_t) = \frac{\mathbf{x}_{tm} - \mu_m}{e^{\omega_m}} \quad \forall m = 1, ..., M \tag{110}$$

such that $\boldsymbol{\eta}_m \sim \mathcal{N}(0, 1)$. Thus, we can transform the expectation in $\mathcal{L}(\boldsymbol{\xi})$ as -

$$\mathbb{E}_{q(\mathbf{x}_t)}[\log p(\mathbf{y}_t, \mathbf{x}_t)] = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\Big[\log p(\mathbf{y}_t, \mathcal{G}^{-1}(\boldsymbol{\eta}))\Big] \tag{111}$$

And so, we can write -

$$\begin{aligned}\mathcal{L}(\boldsymbol{\xi}) &= \mathbb{E}_{\mathcal{N}(\eta)}[\log p(\mathbf{y}_t, \mathcal{G}^{-1}(\boldsymbol{\eta}))] + \mathbb{H}[q(\mathcal{G}^{-1}(\boldsymbol{\eta}))] \\ &= \mathbb{E}_{\mathcal{N}(\eta)}[f(\mathbf{x}_t)] + \mathbb{H}[q(\mathbf{x}_t)]\end{aligned} \tag{112}$$

where $\mathbf{x}_t = \mathcal{G}^{-1}(\boldsymbol{\eta})$.

Now that expectation depends on $\boldsymbol{\eta}$ only, so we can switch the expectation and gradient as -

$$\nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi}) = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\xi}} f(\mathbf{x}_t)] + \nabla_{\boldsymbol{\xi}} \mathbb{H}(q(\mathbf{x}_t)) \tag{113}$$

From (105), $\nabla_{\boldsymbol{\mu}} \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{x}_t)) = \mathbf{0}$ and $\nabla_{\boldsymbol{\omega}} \mathbb{H}(q_{\boldsymbol{\phi}}(\mathbf{x}_t)) = \mathbf{I}_M$. So, we can finally write -

$$\begin{aligned}\nabla_{\boldsymbol{\mu}} \mathcal{L} &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \text{log} p(\mathbf{y}_t, \mathbf{x}_t)] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t) \frac{d\mathbf{x}_t}{d\boldsymbol{\mu}}\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t)]\end{aligned} \tag{114}$$

since $d\mathbf{x}_t / d\boldsymbol{\mu} = \mathbf{I}_M$. Similarly,

$$\begin{aligned}&\nabla_{\boldsymbol{\omega}} \mathcal{L} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \text{log} p(\mathbf{y}_t, \mathbf{x}_t)] + \mathbf{I}_M \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t) \frac{d\mathbf{x}_t}{d\boldsymbol{\omega}}\right] + \mathbf{I}_M \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[(\nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t)) \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right] + \mathbf{I}_M\end{aligned} \tag{115}$$

We can now use Monte Carlo integration to find the expectations in (114) and (115). For this, We draw S samples $\{\eta^{(s)}\}_{s=1}^S$ from $\mathcal{N}(\mathbf{0}, \mathbf{I}_M)$. We estimate $\nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi})$ by averaging over these S samples as -

$$\nabla_{\boldsymbol{\mu}} \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t^{(s)}) \tag{116}$$

$$\nabla_{\boldsymbol{\omega}} \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \left[\nabla_{\mathbf{x}_t} \text{log} p(\mathbf{y}_t, \mathbf{x}_t)\right] \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}}) + \mathbf{I}_M \tag{117}$$

with $\mathbf{x}_t^{(s)} = \boldsymbol{\mu} + \text{diag}(\exp(\boldsymbol{\omega})) * \boldsymbol{\eta}^{(s)}$ So far, we have approximated the gradient of ELBO using Automatic Differentiation and Monte Carlo integration. So using that, we iteratively update the variational parameters at each step as -

$$\boldsymbol{\xi}^{(t+1)} \leftarrow \boldsymbol{\xi}^{(t)} + \lambda \nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi}^{(t)}) \tag{118}$$

where λ is the learning rate and $t = 1, 2, ...$ until convergence. The final posterior estimate of $\mathbf{x}_t$ is therefore -

$$\tilde{\mathbf{x}}_t \sim \mathcal{N}(\boldsymbol{\mu}, diag(exp(\boldsymbol{\omega}))) \tag{119}$$

Using the algorithm defined in this section we can get the posterior estimates for $\tilde{\mathbf{x}}_t$ for each $t = 1, 2, \ldots, T_d$. Thus we get the posterior estimate of the entire data matrix $\tilde{\mathbf{X}}_d = [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_{T_d}]$ in real space. Now, to get to complex domain, we do the reverse transformation of (95) as -

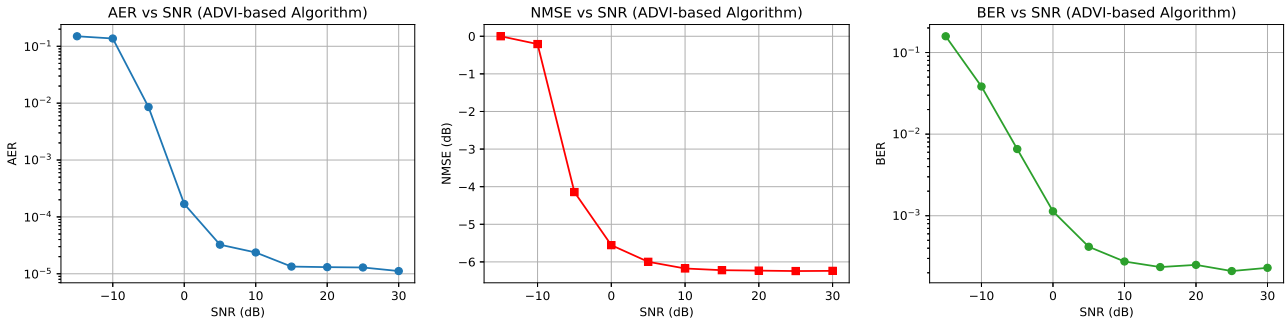
$$\hat{\mathbf{X}}_d = \tilde{\mathbf{X}}_d[: M, :] + j\tilde{\mathbf{X}}_d[M :, :] \tag{120}$$

Algorithm 5 ADVI-Based Data Detection Algorithm

Input: $\mathbf{Y}_d, \mathbf{H}$, tolerance(τ) = 0.01, step-size($\boldsymbol{\alpha}$)
for $t = 0$ *to* T_d **do**
  Initialize: $\boldsymbol{\mu}_t^{(0)} = \mathbf{0}, \boldsymbol{\omega}_t^{(0)} = \mathbf{0}$
  while *change in* $\mathcal{L}$ *is above a threshold* **do**
    Draw S samples $\{\eta\}_{s=1}^S$ from the standard multivariate Gaussian.
    Approximate $\nabla_{\boldsymbol{\mu}_t} \mathcal{L}$ and $\nabla_{\boldsymbol{\omega}_t} \mathcal{L}$ using Equations (116) and (117)
    Update $\boldsymbol{\mu}_t^{(i+1)} \leftarrow \boldsymbol{\mu}_t^{(i)} + \boldsymbol{\alpha} \nabla_{\boldsymbol{\mu}_t} \mathcal{L}$
    Update $\boldsymbol{\omega}_t^{(i+1)} \leftarrow \boldsymbol{\omega}_t^{(i)} + \boldsymbol{\alpha} \nabla_{\boldsymbol{\omega}_t} \mathcal{L}$
    Increment i as $i \leftarrow i + 1$
  Let $\tilde{\mathbf{X}}_d[:, t] = \boldsymbol{\mu}_t$
$\tilde{\mathbf{X}}_d = \tilde{\mathbf{X}}_d[: M, :] + j\tilde{\mathbf{X}}_d[M :, :]$
$\hat{\mathbf{X}}_d = \text{sign}(\Re\{\tilde{\mathbf{X}}_d\}) + j * \text{sign}(\Im\{\tilde{\mathbf{X}}_d\})$
return *Estimated data matrix* $\hat{\mathbf{X}}_d$

2.4 Results

We evaluate the performance of the proposed ADVI-based algorithm for joint active user detection, channel estimation, and data detection (CE-DD) in an uplink mMTC system across a wide range of SNR values from -15 dB to 30 dB. The specific results presented correspond to a configuration with a number of receiver antennas ($N = 64$) and total potential users ($M = 100$),



(a) (b) (c)

Figure 2: Performance of the ADVI-based algorithm at 15% sparsity across SNR levels: (a) Activity Error Rate (AER) versus SNR; (b) Channel Estimation NMSE versus SNR; (c) Data Detection BER.

operating at a sparsity level of 15% (i.e., 15% of users are active at any given time). Also, pilots of length $T_p = 40$, and data of length $T_d = 100$ were used in simulations.

At low SNRs (e.g., -15 dB), the algorithm experiences a relatively high Active Error Rate (AER), which steadily decreases as the SNR improves. The AER exhibits a rapid decline between -10 dB and 0 dB, highlighting the robustness of the ADVI-based method to noise levels. Beyond 5 dB, the AER stabilizes at very low values, demonstrating the effectiveness of the proposed approach even in moderate-to-high SNR regimes.

A similar trend is observed for the normalized mean squared error (NMSE) of the channel estimation. Initially, at very low SNRs, the NMSE values are relatively high, but as the SNR increases, the NMSE decreases significantly and remains consistently low beyond 10 dB. This indicates that the ADVI-based method provides highly accurate channel estimation as the signal quality improves.

Furthermore, the results show that as the sparsity level of the mMTC system increases (i.e., fewer users are active simultaneously), the AER improves substantially. With fewer active devices, the algorithm benefits from reduced interference, enabling more reliable active user detection and further enhancing the overall estimation performance.

Overall, the results demonstrate that the ADVI-based algorithm performs robustly across different SNR and sparsity conditions. It effectively reduces active user detection errors and improves channel estimation accuracy even in challenging low-SNR scenarios, while achieving near-optimal performance at moderate and high SNRs. The observed trends in AER and NMSE clearly highlight the potential of the ADVI-based approach for reliable and efficient active user detection and channel estimation in large-scale mMTC systems.

2.5 Conclusion

In this work, we proposed an ADVI-based algorithm for joint active user detection and channel estimation in uplink mMTC systems. We formulated the problem in a probabilistic framework and applied the ADVI method to approximate the intractable posterior distributions efficiently.

Extensive simulation results demonstrated that the proposed algorithm achieves excellent performance in terms of both active error rate (AER) and normalized mean squared error (NMSE) across a wide range of SNR values. The method remains robust under low-SNR conditions and achieves near-optimal detection and estimation performance at moderate to high SNRs. Additionally, it was observed that higher sparsity levels lead to further improvements in AER, highlighting the algorithm's suitability for practical mMTC scenarios characterized by sporadic user activity.

Overall, the ADVI-based approach provides a scalable, efficient, and accurate solution for active user detection and channel estimation in large-scale grant-free random access systems.

APPENDIX

APPENDIX A

Gradient of ELBO with respect to $\boldsymbol{\mu}$ in (43):

$$\begin{aligned}\nabla_{\boldsymbol{\mu}}\mathcal{L} &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(\mathbf{v},\mathbf{u})] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(\mathbf{v}, S^{-1}(\boldsymbol{\eta}))] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(\mathbf{v}|S^{-1}(\boldsymbol{\eta}))] + \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(S^{-1}(\boldsymbol{\eta}))]\end{aligned} \tag{121}$$

$$\begin{aligned}\text{Now,}\quad \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(\mathbf{v}|S^{-1}(\boldsymbol{\eta}))] &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \nabla_{\boldsymbol{\mu}} \log p(\mathbf{v}_n|S^{-1}(\boldsymbol{\eta}))\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \nabla_{\boldsymbol{\mu}} \log \sigma\left(v_n \mathbf{w}_n^\top \Big(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}\Big)\right)\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \sigma\Big(-v_n \mathbf{w}_n^\top[\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}]\Big) v_n \mathbf{w}_n\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n\right]\end{aligned} \tag{122}$$

where $\mathbf{u} = \boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}$. And,

$$\begin{aligned}&\mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\mu}} \log p(S^{-1}(\boldsymbol{\eta}))] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\boldsymbol{\mu}} \log\left(\frac{|\text{diag}(e^{\boldsymbol{\omega}})|^{-1/2}}{(2\pi)^{M/2}}\right)\right] + \\ &\quad \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\boldsymbol{\mu}}\left(\frac{-1}{2}(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta})^\top(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta})\right)\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[-\Big(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}\Big)\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[-\mathbf{u}]\end{aligned} \tag{123}$$

Using (122) and (123), equation (121) can now be written as -

$$\nabla_{\boldsymbol{\mu}}\mathcal{L} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\left(\sum_{n=1}^{N} \sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n\right) - \mathbf{u}\right] \tag{124}$$

APPENDIX B

Gradient of ELBO with respect to $\boldsymbol{\omega}$ in (44):

$$\begin{aligned}\nabla_{\boldsymbol{\omega}}\mathcal{L} &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \log p(\mathbf{v},\mathbf{u})] + \mathbf{1}_M \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \log p(\mathbf{v}|S^{-1}(\boldsymbol{\eta}))] \\ &\quad + \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \log p(S^{-1}(\boldsymbol{\eta}))] + \mathbf{1}_M\end{aligned} \tag{125}$$

$$\begin{aligned}\text{Now,}\quad \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \log p(\mathbf{v}|S^{-1}(\boldsymbol{\eta}))] &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \nabla_{\boldsymbol{\omega}} \log p(\mathbf{v}_n|S^{-1}(\boldsymbol{\eta}))\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \nabla_{\boldsymbol{\omega}} \log \sigma\left(v_n \mathbf{w}_n^\top \Big(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}\Big)\right)\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \sigma\Big(-v_n \mathbf{w}_n^\top[\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}]\Big) v_n \mathbf{w}_n \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\sum_{n=1}^{N} \sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right]\end{aligned} \tag{126}$$

where $\mathbf{u} = \boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}$. And,

$$\begin{aligned}&\mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[\nabla_{\boldsymbol{\omega}} \log p(S^{-1}(\boldsymbol{\eta}))] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\boldsymbol{\omega}} \log\left(\frac{|\text{diag}(e^{\boldsymbol{\omega}})|^{-1/2}}{(2\pi)^{M/2}}\right)\right] + \\ &\quad \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\nabla_{\boldsymbol{\omega}}\left(\frac{-1}{2}(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta})^\top(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta})\right)\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[-\Big(\boldsymbol{\mu} + \text{diag}(e^{\boldsymbol{\omega}})\boldsymbol{\eta}\Big)\boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right] \\ &= \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}[-\mathbf{u}\boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})]\end{aligned} \tag{127}$$

Using (126) and (127), equation (125) can now be written as -

$$\nabla_{\boldsymbol{\mu}}\mathcal{L} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\eta})}\left[\left(\sum_{n=1}^{N}[\sigma(-v_n \mathbf{w}_n^\top \mathbf{u}) v_n \mathbf{w}_n] - \mathbf{u}\right) \boldsymbol{\eta}^\top \text{diag}(e^{\boldsymbol{\omega}})\right] + \mathbf{1}_M \tag{128}$$

APPENDIX C

Simplifying the ELBO in (77) :

$$\begin{aligned}\mathcal{L}(\boldsymbol{\phi}) &= \sum_{n=1}^{N} \log p(\mathbf{y}_n|\boldsymbol{\alpha}, \mathbf{h}_n) + \log p(\boldsymbol{\alpha}) + \sum_{n=1}^{N} \log p(\mathbf{h}_n) \\ &= \left[\sum_{n=1}^{N}\sum_{t=1}^{T} \log \sigma\Big(y_{nt} \mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big)\right] - \boldsymbol{\alpha}^\top\boldsymbol{\alpha} - \sum_{n=1}^{N} \frac{1}{2}\mathbf{h}_n^\top \mathbf{h}_n\end{aligned} \tag{129}$$

APPENDIX D

Gradient of ELBO with respect to $\boldsymbol{\mu}_{S_0}$ in (79) using simplification in Appendix (A):

$$\begin{aligned}\nabla_{\boldsymbol{\mu}_{S_0}}\mathcal{L}(\boldsymbol{\phi}) &= \nabla_{\boldsymbol{\mu}_{S_0}}\left[\sum_{n=1}^{N}\sum_{t=1}^{T} \log \sigma\Big(y_{nt} \mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big)\right] \\ &\quad - \nabla_{\boldsymbol{\mu}_{S_0}}\left[\boldsymbol{\alpha}^\top\boldsymbol{\alpha} + \sum_{n=1}^{N} \frac{1}{2}\mathbf{h}_n^\top \mathbf{h}_n\right] \\ &= \sum_{n=1}^{N}\sum_{t=1}^{T} \sigma\Big(-y_{nt} \mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \boldsymbol{\alpha}\end{aligned} \tag{130}$$

APPENDIX E

Gradient of ELBO with respect to $\boldsymbol{\mu}_{S_n}$ for each $n = 1, 2, \ldots, N$ in (79) using simplification in Appendix (A):

$$\begin{aligned}\nabla_{\boldsymbol{\mu}_{S_n}}\mathcal{L}(\boldsymbol{\phi}) &= \nabla_{\boldsymbol{\mu}_{S_n}}\left[\sum_{n=1}^{N}\sum_{t=1}^{T} \log \sigma\Big(y_{nt} \mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big)\right] \\ &\quad - \nabla_{\boldsymbol{\mu}_{S_n}}\left[\boldsymbol{\alpha}^\top\boldsymbol{\alpha} + \sum_{n=1}^{N} \frac{1}{2}\mathbf{h}_n^\top \mathbf{h}_n\right] \\ &= \sum_{n=1}^{N}\sum_{t=1}^{T} \sigma\Big(-y_{nt} \mathbf{h}_n^\top(\boldsymbol{\alpha})\mathbf{x}_t\Big) - \mathbf{h}_n\end{aligned} \tag{131}$$