SCALABLE DISCRETE-TO-CONTINUOUS CHANNEL SIMULATION FOR COMPRESSION AND PRIVACY

Anonymous authors
Paper under double-blind review

ABSTRACT

Channel simulation has recently emerged as a useful component in machine learning systems where samples from a prescribed probability distribution are to be compressed. We study the simulation of discrete-to-continuous channels and present a scheme which performs exact simulation using a fixed-size shared random pool containing one or more samples from each potential target distribution. Unlike sampling-based schemes which generate a potentially unbounded sequence of independent samples from a proposal distribution, the number of samples used by our algorithm is finite and fixed in advance. To the best of our knowledge, this is the first exact scheme with such a property for general discrete-to-continuous channels. The key ingredient in our approach is a latent permutation which hides the correspondence between the shared random samples and their generating distributions. We also provide a flexible tradeoff between the number of generated samples and the compression rate. Exploiting the structure exposed by our algorithm, we use polar and multilevel coding to scale to long blocklengths in $O(n \log n)$ time to benefit from reduced per-symbol overhead. In our experiments, exact simulation remains fast at large blocklengths, and for input alphabet sizes up to $N = 16$; for larger alphabets our approach further admits a Sinkhorn-Knopp approximation of the posterior induced by the latent permutation. We give applications to variable-rate compression with stochastic VQ-VAEs and differentially private distributed mean estimation via exact simulation of the Gaussian mechanism, showing gains over baselines in both wall-clock speed and communication rate.

1 INTRODUCTION

In channel simulation, one party, termed the *encoder*, observes a source $X \in \mathcal{X}$, $X \sim P_X$, and sends a message M to another party, the *decoder*, which uses the message to output a random variable $Y \in \mathcal{Y}$. Here, Y must follow a prescribed conditional distribution $P_{Y|X}(\cdot \mid X)$, called the *channel*, while the expected bit-length of M, $\mathbb{E}[\ell(M)]$, should be made as small as possible. To assist in reducing the communication cost, we assume that shared randomness W is available at both the encoder and decoder. Viewed in this way, channel simulation provides a general method to compress a noisy version of X into a binary representation, where the amount and type of noise added to X is controlled by the choice of $P_{Y|X}$. It may also be regarded as a stochastic generalization of lossy compression methods such as quantization, which similarly perturb a source and produce discrete representations. As such, it has emerged as a key building block in neural compression architectures, where the deterministic nature of quantization can pose difficulties during training (Flamich & Gündüz, 2026), and has also been applied to privacy (Liu et al., 2024).

The nature of X and Y has a direct bearing on the difficulty of channel simulation. While many constructions have been proposed for arbitrary probability spaces, notably the Poisson functional representation (PFR, Li & Gamal, 2018), rejection-sampling based schemes (Flamich et al., 2023; Phan & Khisti, 2025) and approximate methods leveraging importance sampling (Havasi et al., 2019; Phan et al., 2024), these may either be slow without additional restrictions on $P_{Y|X}$, suffer from highly variable running times, or produce biased outputs at smaller sample counts. Most critically, known exact general methods require generating infinitely many samples in the worst case, making them less attractive for real-world applications. In contrast, when Y is discrete, the exponential functional representation (EFR, Li & Gamal, 2018) can be used to simulate any channel in $O(|\mathcal{Y}|)$ time with the shared randomness consisting of $|\mathcal{Y}|$ exponential random variables. We are interested in

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the case where X is from a finite set but $\mathcal{Y}$ is continuous, which arises in situations where continuous noise perturbs discrete input data, and ask whether this second problem may also be solved with a *fixed number* of shared randomness samples. Our first goal in this paper is to answer in the affirmative by introducing a novel approach we term the *permuted scheme*. In contrast to existing sampling-based schemes which rely on a shared sequence of independent samples and use this common randomness to reduce the communication rate, our approach generates one sample from each possible conditional output distribution, resulting in a sample pool of size $|\mathcal{X}|$. To achieve compression, it adds additional randomness by shuffling the sample pool with a latent permutation, resulting in a non-independent sequence. While the fixed-size sample pool leads to some communication overhead above $I(X;Y)$, this can be reduced by increasing the number of samples per output mode.

In machine learning applications, a further challenge is that the channels to be simulated are often high-dimensional, involving for example vectors of latents, gradients or network weights. In this *multi-shot* setting we are interested in simulating n independent uses of the channel; the alphabet size $|\mathcal{X}|$ remains fixed while n may be large. Increasing n, also known as the blocklength, has the benefit of amortizing communication overheads. For example, one-shot PFR achieves an expected message length at most $I(X;Y) + \log(I(X;Y) + 2) + 3$ bits (Li, 2024); extending to blocklength n yields $I(X;Y) + \log(nI(X;Y) + 2)/n + 3/n$ bits per sample, which approaches $I(X;Y)$ as $n \to \infty$. Unfortunately, the time complexity of PFR and similar general channel simulation schemes increases exponentially with n, rendering their use on high-dimensional channels impractical. Recent work has turned to techniques from coding theory to accelerate channel simulation across many independent dimensions (Sriramu et al., 2024; Zhao & Li, 2026; Ozyilkan et al., 2026), yet their applicability remains limited to binary or other discrete outputs. Our second objective is therefore to demonstrate how a polar coding framework can similarly be incorporated into the permuted scheme, scaling discrete-to-continuous channel simulation to long blocklengths. In summary, our contributions are:

1. We introduce the permuted scheme for general discrete-input, continuous-output channels, using a novel random permutation to achieve exact simulation with a pool of random samples whose size is finite and fixed ahead of time. We further show how an approximation based on the Sinkhorn-Knopp algorithm (Sinkhorn, 1964; Knight, 2008) can be used to scale our approach to large input alphabets in applications such as neural compression where an exact output distribution may not be necessary.
2. By adopting a construction based on polar codes (Arıkan, 2009) and multilevel coding (MLC, Wachsmann et al., 1999), we scale the dimension of channels which can be simulated with the permuted scheme. For a given alphabet size, the complexity of our algorithm scales with the channel dimension as $O(n \log n)$, making it practical even when n is large and allowing operation at long blocklengths where per-symbol overhead is reduced.
3. We deploy our scheme to two applications: first, we demonstrate how our channel simulator can be used to achieve variable-rate lossy compression from VQ-VAEs (van den Oord et al., 2017) without retraining for different rates; second, we present a scheme for communication-efficient distributed mean estimation (DME) with central differential privacy (CDP) via exact simulation of the Gaussian mechanism (Dwork & Roth, 2014).

Throughout, logarithms are base 2 unless otherwise stated. Entropies, mutual information and KL divergence are to the same base. We write x^n for a sequence $\{x_i\}_{i=1}^n$ and let $[n] = \{1, \dots, n\}$.

2 Background and Related Work

One-Shot Channel Simulation. In the *one-shot* setting, the encoder observes a single symbol X and the decoder outputs a corresponding Y. Since X and Y are not required to be scalars, but may instead be random sequences, vectors, or any other object, the one-shot setting is the most general. Bennett et al. (2002) and Winter (2002) established theoretical performance bounds for channel simulation under unlimited common randomness, with the lower bound being $I(X;Y)$. Harsha et al. (2010) introduced an early simulation algorithm for discrete channels known as greedy rejection sampling (GRS) which achieves a near-optimal coding cost, and was later extended to general probability spaces (Flamich et al., 2023; Flamich & Theis, 2023). Adopting a shared Poisson point process as the common randomness, Li & Gamal (2018) introduced PFR, a popular exact channel simulation algorithm whose rate also attains the mutual information with logarithmic redundancy.

More recently, schemes based on standard rejection sampling have also emerged as alternatives with similar theoretical guarantees (Phan & Khisti, 2025; Hill et al., 2026).

Multi-Shot Channel Simulation. By the multi-shot setting, we refer to simulating n iid copies of a channel, i.e. the product channel $P_{Y|X}^{\times n}$. In principle, any one-shot simulation algorithm can be scaled to the multi-shot case by treating the input X^n as one high-dimensional symbol. However, while algorithms have been proposed which improve the performance of one-shot simulation in special cases (Flamich et al., 2022; Flamich, 2023; Hegazy & Li, 2022), existing general sampling-based schemes do not scale well to large n as their sample complexity is exponential in the joint mutual information (Li, 2024). For binary-output channels, Sriramu et al. (2024) devised an algorithm based on polar coding (Arıkan, 2009) which runs in $O(n \log n)$ time and can hence take advantage of decreased per-symbol overhead at long blocklengths. Subsequently, Zhao & Li (2026) introduced methods based on polar and other linear codes extending multi-shot simulation to additive exchangeable noise channels over finite fields, while Ozyilkan et al. (2026) generalized to a non-stationary setting. However, these approaches are currently limited to discrete or, in the case of long-blocklength polar code constructions, binary outputs, and cannot be used when a richer output space is desired.

Applications to Neural Compression and Privacy. The connection between channel simulation and compression was established by Winter (2002); any channel simulation algorithm can be converted into a lossy compressor (Li & Gamal, 2018). On the practical side, Havasi et al. (2019) used approximate simulation to compress network parameters, while Flamich et al. (2020) applied a similar approach to images. Theis et al. (2022) defined a channel from a diffusion model and simulated it with PFR to achieve image compression under a realism constraint, which Vonderfecht & Feng (2025) made practical using a faster approximate construction. Phan et al. (2024) adopted channel simulation as the basis for image compression with side information at the decoder. Continuous latent spaces derived from β-VAEs or Gaussian diffusion have typically been used in these approaches, which consequently suffer from the high computational cost of simulating general channels and are practically limited to short blocklengths. By instead using our discrete-to-continuous scheme and the discrete latent space of a VQ-VAE, we retain the flexibility of channel simulation-based compression while reducing coding overhead by jointly processing blocks of up to 8192 latent variables.

Channel simulation has also found applications in differential privacy (DP, Dwork et al., 2006; Dwork & Roth, 2014), which offers a principled way to protect users' data released to an untrusted party. A DP mechanism applies a randomized privacy-preserving perturbation to the data X, canonical examples being additive Gaussian or Laplacian noise, and can therefore be construed as a channel (Li, 2024). Feldman & Talwar (2021) used approximate simulation via rejection sampling and pseudorandomness to reduce the communication cost of locally differentially private (LDP) mechanisms. Shah et al. (2022) developed an approach based on importance sampling (IS). However, these methods are approximate in the sense that they do not preserve the target noise distribution. Under a trusted aggregator, Hasırcıoğlu & Gündüz (2024) and Hegazy et al. (2024) used dithered quantization to achieve exact realizations of 1D additive noise mechanisms. Liu et al. (2024) proposed a general method extending PFR to exactly simulate any LDP or CDP mechanism, at the expense of inheriting PFR's cost and non-deterministic stopping time. We offer a complementary approach that can be used to exactly and quickly simulate CDP mechanisms acting on discrete inputs up to 16-ary alphabets.

3 One-Shot Discrete-to-Continuous Channel Simulation

We are interested in simulating a channel $P_{Y|X}$ from $X \in \mathcal{X}$ to $Y \in \mathcal{Y}$, where $\mathcal{X}$ is a finite set, while using a fixed number of shared random samples. Suppose without loss of generality that $\mathcal{X} = [N]$; we assume that for all $x \in \mathcal{X}$, the density $p_{Y|X}(\cdot \mid x)$ exists. Our key idea is to take one sample from each potential output distribution $P_{Y|X}(\cdot \mid x)$. A naive possibility that achieves exact simulation is:

1. Generate shared randomness $\bar{U}^N$, where $\bar{U}_i \sim P_{Y|X}(\cdot \mid i)$.
2. At the encoder, observe $X = x$ and select the index $K = x$.
3. Send K to the decoder, which will output $Y = \bar{U}_K$.

However, since this scheme directly transmits X to the decoder without compression, the communication cost is at least the source entropy $H(X)$, which may be much larger than $I(X;Y)$. An intuitive

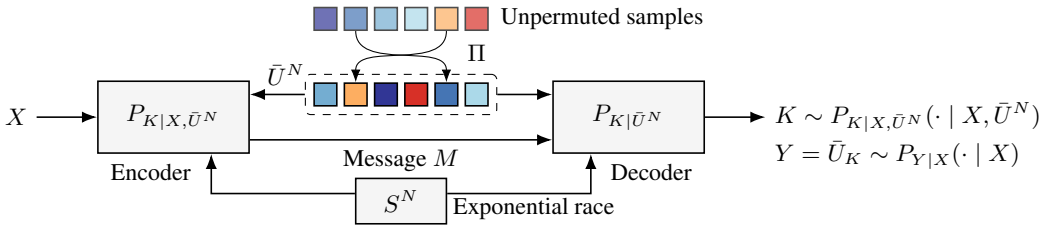


Figure 1: Permuted scheme for one-shot channel simulation. After being used to scramble the random samples, the permutation Π does not play any further role in the encoder's target distribution $P_{K|X,\bar{U}^N}(\cdot \mid X, \bar{U}^N)$ or the decoder's prior $P_{K|\bar{U}^N}(\cdot \mid \bar{U}^N)$.

idea to reduce the amount of communication might be to introduce source-dependent randomness into the index K such that it follows a conditional distribution $P_{K|X}$, while retaining the same sample pool $\bar{U}^N$. Unfortunately, as we show in Appendix A using a simple counterexample on a BI-AWGN channel, the only general solution preserving the output distribution under this construction is $P_{K|X}(k \mid x) = \delta(k - x)$, which brings us back to the naive scheme and a communication cost of $H(X)$. Somewhat counterintuitively, to reduce the rate below the entropy while simulating $P_{Y|X}$ exactly, our proposal instead works by throwing away some of the information contained in the shared randomness, shuffling $\bar{U}^N$ so that the encoder and decoder no longer know a priori which sample belongs to which mode. This is the basis of our permuted scheme, which we consider next.

3.1 Permuted Scheme for Channel Simulation with Finitely Many Samples

To overcome the naive scheme's failure to drive the communication cost closer to $I(X;Y)$, our key insight is to break the deterministic mapping between the index of each sample $\bar{U}_i$ and the distribution which generated it via a random permutation. Importantly, the permutation is ignored after being used to shuffle the samples, which are treated thereafter as marginally identically distributed. This deliberate ignorance on the part of the encoder and decoder is what allows the rate to drop below $H(X)$ in our scheme. The key steps are as below:

1. Generate a uniform random permutation Π of $[N]$ and, given $\Pi = \pi$, generate shared randomness $\bar{U}^N$, where $\bar{U}_{\pi(i)} \sim P_{Y|X}(\cdot \mid i)$.
2. At the encoder, observe $X = x$ and select K according to the target distribution $P_{K|X,\bar{U}^N}(k \mid x, u^N) = \Pr[\Pi(x) = k \mid \bar{U}^N = u^N]$. Note that K is conditionally independent of Π given $\bar{U}^N$ by construction.
3. Losslessly encode K into the message M and send it to the decoder to output $Y = \bar{U}_K$.

The problem is now reduced to communicating a sample from $P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$. We highlight that the random permutation induces a non-independent sequence of samples, unlike prior channel simulation schemes where the samples are independent. By sampling K after the permutation, our algorithm exploits the additional uncertainty injected into the posterior about which sample was generated from which target distribution given X and $\bar{U}^N$. We continue our earlier BI-AWGN example in Appendix A to motivate the use of the random permutation and give a concrete demonstration of the permuted scheme. Next, and before proceeding to the rate analysis, we provide a guarantee that our approach performs exact channel simulation. The proof is found in Appendix B.1.

Proposition 1. *The permuted scheme exactly simulates $P_{Y|X}$.*

To efficiently send the index $K \sim P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$ to the decoder and establish the communication cost, we can use the well-established strategy of coordinated sampling with a shared exponential race, also known as EFR (Li & Gamal, 2018). In particular, a sequence S^N of iid $\mathrm{Exp}(1)$ random variables is generated and shared between the encoder and the decoder. After sorting S^N to take into account the decoder's knowledge of the prior $P_{K|\bar{U}^N}(\cdot \mid u^N)$, K is chosen to be the winning index of the exponential race, and its position $\bar{K}$ in the *sorted* list is communicated using a suitable lossless entropy code. This approach is detailed in Algorithm 1 and visualized in Fig. 1. The upper bound on the expected length achieved by EFR (Li, 2024, Theorem 14) then applies directly to the permuted scheme, giving us the following one-shot achievability result. A proof is given in Appendix B.2.

Algorithm 1 One-Shot Discrete to Continuous Channel Simulation

function ENCODE($x, \mathfrak{G}, \texttt{enc}$)
Input: Channel input x, PRNG $\mathfrak{G}$, lossless encoder `enc`. **Output:** Message M.
Generate Π, a uniform random permutation of $[N]$, using $\mathfrak{G}$.
Given $\Pi = \pi$, sample $\bar{U}_{\pi(i)} \sim P_{Y|X}(\cdot \mid i)$ using $\mathfrak{G}$, along with $S^N \sim \text{Exp}(1)$ iid.
Given $\bar{U}^N = u^N$, $p_i \leftarrow P_{K|X,\bar{U}^N}(i \mid x, u^N)$, $q_i \leftarrow P_{K|\bar{U}^N}(i \mid u^N)$. ▷ See (3).
Compute a permutation σ such that $S_{\sigma(1)}/q_{\sigma(1)} \leq S_{\sigma(2)}/q_{\sigma(2)} \leq \cdots \leq S_{\sigma(N)}/q_{\sigma(N)}$.
$K \leftarrow \arg\min_{1 \leq i \leq N} S_i/p_i$, $\bar{K} \leftarrow \sigma^{-1}(K)$, $M \leftarrow \texttt{enc}(\bar{K})$
return M

function DECODE($M, \mathfrak{G}, \texttt{dec}$)
Input: Message M, PRNG $\mathfrak{G}$, lossless decoder `dec`. **Output:** Channel output Y.
Generate Π, a uniform random permutation of $[N]$, using $\mathfrak{G}$.
Given $\Pi = \pi$, sample $\bar{U}_{\pi(i)} \sim P_{Y|X}(\cdot \mid i)$ using $\mathfrak{G}$, along with $S^N \sim \text{Exp}(1)$ iid.
Given $\bar{U}^N = u^N$, $q_i \leftarrow P_{K|\bar{U}^N}(i \mid u^N)$. ▷ See (3).
Compute a permutation σ such that $S_{\sigma(1)}/q_{\sigma(1)} \leq S_{\sigma(2)}/q_{\sigma(2)} \leq \cdots \leq S_{\sigma(N)}/q_{\sigma(N)}$.
$\bar{K} \leftarrow \texttt{dec}(M)$, $K \leftarrow \sigma(\bar{K})$, $Y \leftarrow \bar{U}_K$
return Y

Theorem 1. *The index transmitted to the decoder as part of the permuted scheme satisfies* $\mathbb{E}[\log \bar{K}] \leq I(X;Y \mid \bar{U}^N) + 1$*, and it achieves an expected message length in bits of*

$$\mathbb{E}[\ell(M)] \leq I(X;Y \mid \bar{U}^N) + \log(I(X;Y \mid \bar{U}^N) + 2) + 3. \tag{1}$$

In Appendix C, we provide context on the differences between the upper bound in Theorem 1 and similar achievability results for other channel simulation algorithms such as PFR and GRS, especially the role of the conditional mutual information $I(X;Y \mid \bar{U}^N)$, and prove a related lower bound.

3.2 COMPUTING THE POSTERIOR DISTRIBUTION

To run Algorithm 1 in practice, the encoder needs to compute the posterior $P_{K|X,\bar{U}^N}(k \mid x, u^N)$ for all $k \in [N]$, while the decoder needs to calculate $P_{K|\bar{U}^N}(k \mid u^N)$. From Bayes' rule, we have

$$P_{K|X,\bar{U}^N}(k \mid x, u^N) = \frac{p_{\bar{U}^N}(u^N \mid \Pi(x) = k)}{N p_{\bar{U}^N}(u^N)}. \tag{2}$$

The prior $P_{K|\bar{U}^N}(k \mid u^N)$ can in turn be found from (2), and evaluates to a constant if X is uniform. However, computing the terms on the right-hand side is difficult due to the introduction of the latent permutation Π and the dependence between the shared random samples, distinct from existing channel simulation schemes where these are independent. To derive an exact expression for the posterior, we introduce the $N \times N$ matrix of likelihoods $L(u^N)$, where $L_{ij} = p_{Y|X}(u_i \mid j)$. Then,

$$P_{K|X,\bar{U}^N}(k \mid x, u^N) = \frac{p_{Y|X}(u_k \mid x)\,\text{perm}(L(u^N)_{\{k,x\}})}{\text{perm}(L(u^N))} \tag{3}$$

where perm denotes the matrix permanent and $L(u^N)_{\{k,x\}}$ is formed by removing row k and column x from $L(u^N)$. While computing the permanent is #P-complete, it remains fast up to around $N = 16$. This is the method we adopt for the CDP experiment in Section 5.2, which requires exact simulation to maintain privacy. For large N, as is the case for our VQ-VAE image compression experiment in Section 5.1 which uses $N = 256$, we turn to an approximation based on Sinkhorn-Knopp iterations which has previously been adopted in machine learning for approximating posteriors over latent permutations (Mena et al., 2018; 2020). Although this no longer offers exact channel simulation, we find the approximation sufficient to achieve competitive end-to-end rate-distortion performance. Additional details concerning the approximation and its empirical performance are provided in Appendix E.1. Moreover, the rate analysis in Theorem 1 still applies if $I(X;Y \mid \bar{U}^N)$ is instead defined with respect to the joint distribution induced by the approximate channel. We emphasize that while we use an approxmation for large *alphabets* in our experiments, exact simulation remains practical for large *blocklengths* if N is small enough, as we will demonstrate in Section 5.2.

3.3 Communication Cost with Multiple Samples per Mode

A natural question following from the theoretical discussion in Section 3.1 is whether the structure of the shared randomness in the permuted scheme can be modified to recover the mutual information $I(X;Y)$ in place of $I(X;Y \mid \bar{U}^N)$. Since $I(X;Y \mid \bar{U}^N) = I(X;Y) + I(X;\bar{U}^N \mid Y)$, this implies reducing the dependence between X and the set of rejected samples given $Y = \bar{U}_K$. As this dependence arises due to the finite size of the sample pool, it is reasonable to expect that increasing the total number of samples beyond N may serve to reduce the size of the penalty. Following this idea, we now consider a generalized setting with $m > 1$ iid samples per mode; as m becomes large, knowing X and Y reveals relatively less information about the remaining $Nm - 1$ samples. The target distribution for the index $K_m \in [Nm]$ becomes

$$P_{K_m|X,\bar{U}^{Nm}}(k \mid x, u^{Nm}) = \frac{1}{m} \Pr[k \in \Pi(x) \mid \bar{U}^{Nm} = u^{Nm}] \tag{4}$$

where Π is now a uniform random mapping from each mode x to a set $\Pi(x)$ containing the m unique indices assigned to it. First, we show that this scheme also performs exact channel simulation, with the proof following in Appendix B.3.

Proposition 2. *The multi-sample permuted scheme exactly simulates $P_{Y|X}$.*

If we use an exponential race to communicate K_m, Theorem 1 also applies to the multi-sample extension and shows that the communication cost is determined by $I(X;Y \mid \bar{U}^{Nm})$. Summarized below and in Appendix B.4, we show that this approaches $I(X;Y)$ as m increases. A numerical evaluation is in Appendix E.2. For technical reasons we restrict our analysis to the binary-input case, though we expect the result to extend to larger N. Note also that in the polar coding extension and experiments to follow in Sections 4 and 5, we focus on $m = 1$, which we find already sufficient to obtain competitive compression rates.

Theorem 2. *Let $\mathbb{E}[\ell_m(M)]$ be the expected message length when m samples are taken per mode. When $N = 2$, the expected message length using the multi-sample permuted scheme satisfies*

$$\limsup_{m\to\infty} \mathbb{E}[\ell_m(M)] \leq I(X;Y) + \log(I(X;Y) + 2) + 3. \tag{5}$$

4 Scaling the Permuted Scheme with Polar Coding

As mentioned in Section 1, in applications such as the transmission of an n-dimensional vector of independent latent variables, it may be necessary to simulate multiple instances of one channel. When considering a source vector $X^n \sim P_X$ iid and the product channel $P_{Y|X}^{\times n}$, the one-shot scheme could be applied to n independent blocks of N permuted samples to produce an index sequence K^n. As $P_{K^n|X^n,\bar{U}^{Nn}}$ factorizes across channel uses, using a joint exponential race to encode K^n gives

$$\mathbb{E}[\ell(M)]/n \leq I(X;Y \mid \bar{U}^N) + \log(nI(X;Y \mid \bar{U}^N) + 2)/n + 3/n, \tag{6}$$

analogous to (1), which approaches $I(X;Y \mid \bar{U}^N)$ as n becomes large. The exponential race would however contain N^n candidates, which quickly becomes infeasible. The one-shot permuted scheme therefore suffers from the rapid growth in complexity with the source dimension common to existing protocols like PFR and GRS. Nevertheless, a secondary benefit of the permuted scheme is that its fixed sample count construction, with only N samples to choose between per channel use, introduces structure allowing us to adapt the PolarSim algorithm (Sriramu et al., 2024), which operates on iid copies of binary-output channels in $O(n \log n)$ time, to devise an efficient multi-shot version.

Owing to PolarSim's binary-output requirement, which is also inherited by recent extensions (Ozyilkan et al., 2026), it is not immediately obvious how to incorporate its framework into our discrete-to-continuous setting. While we have shown in Section 3.1 that the permuted scheme reduces to communicating the N-ary index K after conditioning on the shared random samples, even if K were binary, constructing a new polar code for each instance of the shared randomness would be impractical. Moreover, the resulting channel instances would not be identically distributed. In the remainder of this section, we show how to overcome these challenges. First, the permuted scheme with $N = 2$ can be reformulated as simulating a binary-symmetric channel (BSC) with side information, which can be solved efficiently by a simple generalization of PolarSim. Second, we generalize to larger alphabets via multilevel coding (MLC, Wachsmann et al., 1999).

4.1 Applying Polar Coding for a Binary Alphabet

To begin, we consider $N = 2$ as a special case, where we can evaluate (2) directly to get

$$P_{K|X,\bar{U}^2}(1 \mid 1, u^2) = P_{K|X,\bar{U}^2}(2 \mid 2, u^2) = 1 - q(u^2) \tag{7}$$

$$P_{K|X,\bar{U}^2}(1 \mid 2, u^2) = P_{K|X,\bar{U}^2}(2 \mid 1, u^2) = q(u^2) \tag{8}$$

where we have introduced the state-dependent BSC crossover probability

$$q(u^2) = \frac{p_{Y|X}(u_1 \mid 2)p_{Y|X}(u_2 \mid 1)}{p_{Y|X}(u_1 \mid 1)p_{Y|X}(u_2 \mid 2) + p_{Y|X}(u_1 \mid 2)p_{Y|X}(u_2 \mid 1)}. \tag{9}$$

For n independent channel uses,

$$P_{K^n|X^n,\bar{U}^{2n}}(k^n \mid x^n, u^{2n}) = \prod_{i=1}^{n} \mathrm{BSC}(k_i \mid x_i, q(u_i^2)) = \delta(k_i - x_i)(1 - 2q(u_i^2)) + q(u_i^2) \tag{10}$$

which represents a nonstationary BSC where the crossover probability changes at each timestep i. While Ozyilkan et al. (2026) recently proposed a generalization of PolarSim to simulate such a time-varying channel, their approach relies on interleaving custom permutations inside the polar transform, introducing additional overhead and requiring a more complicated theoretical analysis.

Instead, we propose viewing the channel as a stationary channel with side information, which is possible due to the specific structure of the shared randomness in the permuted scheme. Let $w_i = u_i^2$ be the side information at timestep i, which is known to both the encoder and the decoder. We have that (x_i, w_i) are iid draws from the joint distribution $P_X \times P_{\bar{U}^2}$. Then, following the discussion in Appendix D, and in particular Theorem 4, the modified PolarSim framework which we present there in Algorithm 2 can be used to simulate the channel with a communication cost approaching $I(X; Y \mid \bar{U}^2)$ bits per channel use as n grows.

4.2 Increasing the Alphabet Size via Multilevel Coding

For real-world applications like those we will pursue experimentally in Section 5, it is important to be able to use alphabets larger than $N = 2$. Fortunately, the binary scheme provides a foundation for efficient simulation at long blocklengths when N is increased, so long as the posterior distribution in (2) or its approximation remains tractable. We base our method on MLC (Wachsmann et al., 1999), which is a classical approach for constructing capacity-achieving codes for 2^L-ary transmission, $L \geq 1$, from existing binary codes. Beginning with the channel $P_{K|X,\bar{U}^N}$, we identify K with a binary sequence B^L, $L = \log N$. Define the bijection $f(b^L) = 1 + \sum_{l=1}^{L} 2^{l-1} b_l$ so that $K = f(B^L)$. Applying the chain rule, $P_{K|X,\bar{U}^N}(k \mid x, u^N) = \prod_{l=1}^{L} P_{B_l|B^{l-1},X,\bar{U}^N}(b_l \mid b^{l-1}, x, u^N)$, where $b^L = f^{-1}(k)$. This decomposes the task of sampling K into L sequential binary sampling steps.

To find the bit-wise conditional distributions, we take the posterior distribution in (2) and let $q_{jk}(u^N) = P_{K|X,\bar{U}^N}(k \mid j, u^N)$ be the crossover probability from $X = j$ to $K = k$. Next, we divide the set of binary sequences $A = \{0,1\}^L$ into subsets $A(b^l) = \{a^L \in A : a^l = b^l\}$. Then,

$$P_{B_l|B^{l-1},X,\bar{U}^N}(b_l \mid b^{l-1}, x, u^N) = \frac{\sum_{k \in f(A(b^l))} q_{xk}(u^N)}{\sum_{k \in f(A(b^{l-1}))} q_{xk}(u^N)}. \tag{11}$$

We now have a binary-output channel at each level, which can be efficiently simulated for n independent uses using the polar coding method introduced in Section 4.1. For level l, the side information at timestep i is $w_i = (u_i^N, b_i^{l-1})$, and (x_i, w_i) are iid draws from $(P_X \times P_{\bar{U}^N}) P_{B^{l-1}|X,\bar{U}^N}$. Again referring to Appendix D and Theorem 4, for each level and as the blocklength grows, the extended PolarSim scheme in Algorithm 2 achieves

$$\lim_{n\to\infty} \mathbb{E}[\ell(M_l)]/n = I(X; B_l \mid B^{l-1}, \bar{U}^N) \tag{12}$$

where M_l is the transmitted message at level l. The total cost to communicate K in the limit is then

$$\lim_{n\to\infty} \frac{1}{n} \sum_{l=1}^{L} \mathbb{E}[\ell(M_l)] = \sum_{l=1}^{L} I(X; B_l \mid B^{l-1}, \bar{U}^N) = I(X; B^L \mid \bar{U}^N) = I(X; Y \mid \bar{U}^N) \tag{13}$$

where the last equality follows as for continuous outputs and given $\bar{U}^N$, Y is uniquely determined by $K = f(B^L)$ and vice versa.

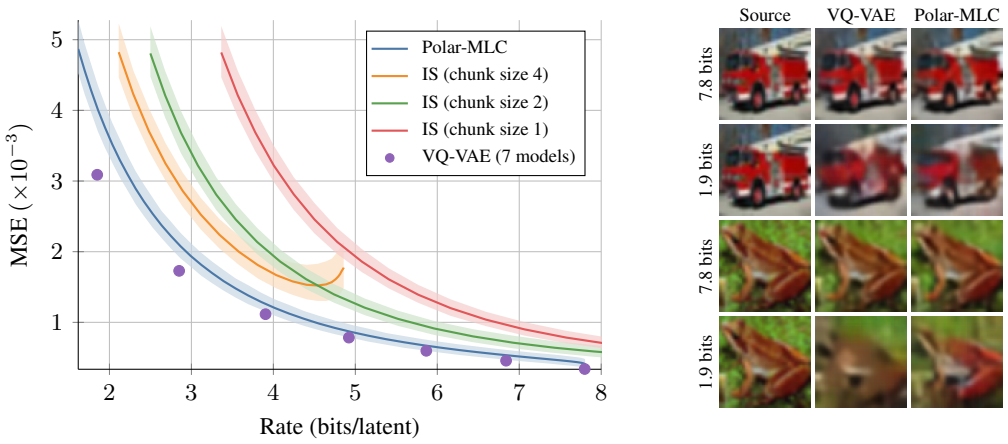


Figure 2: Rate-distortion performance. *Left:* Using our channel simulator, a single stochastic model with the rate chosen at inference time can approach the performance of individually trained deterministic models with fixed codebooks, especially at high rates. P05–P95 MSE is shown. *Right:* CIFAR-10 images compressed using fixed-rate VQ-VAEs and our variable-rate stochastic model.

5 Experiments

5.1 Variable-Rate Compression with Stochastic VQ-VAEs

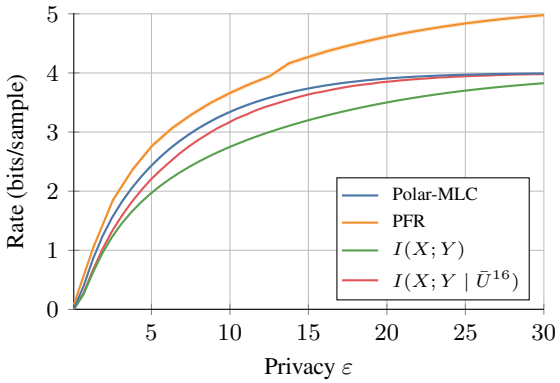
VQ-VAEs (van den Oord et al., 2017) are autoencoders which compress high-dimensional data such as images into discrete tokens. Letting W be the input image, a standard VQ-VAE first consists of an encoder network producing a sequence of n latent vectors $z^n = f_{\phi_e}(W) \in \mathbb{R}^{n \times D}$, typically arranged in a grid, where D is the dimension of the latent space. A vector quantizer then snaps each latent to the nearest codeword from a learned codebook c^N, producing

$$k_i^* = \underset{1 \le k \le N}{\arg\min} \|f_{\phi_e}(W)_i - c_k\|_2, \ \hat{z}_i = c_{k_i^*}. \tag{14}$$

Finally, a decoder network takes the quantized latents and outputs a reconstruction $\hat{W} = g_{\phi_d}(\hat{z}^n)$.

Unlike β-VAEs (Higgins et al., 2017) which offer inherent rate-distortion flexibility, the rate of a basic VQ-VAE is fixed at approximately $\log N$ bits/latent, as the index assignment in (14) does not provide a differentiable rate objective. However, if $\hat{z}_i$ is replaced by a sample from a learned Gaussian posterior, creating a *stochastic* variant of the VQ-VAE, our permuted scheme can take the indices returned by the argmin and reduce the communication cost below $\log N$ by adding noise to the latents. To this end, we learn an isotropic Gaussian with mean and variance vectors $\mu_k, \sigma_k^2 \in \mathbb{R}^D$ in place of the codeword c_k for each codebook index. While related stochastic formulations have previously been explored to enhance the training of VQ-VAEs (Takida et al., 2022; Xu et al., 2026), they have not to the best our knowledge been adopted at inference time to reduce the rate. By adopting the loss-conditioning technique of Dosovitskiy & Djolonga (2020) to scale the variances and condition the decoder based on a scalar rate parameter λ, our approach can train a single VQ-VAE checkpoint to sweep a continuous range of rate-distortion operating points without retraining.

We demonstrate our approach for image compression on CIFAR-10 upscaled to 64×64, with the model architecture from van den Oord et al. (2017). The channel input symbols are $X_i = k_i^*$ as in (14) and the alphabet size is $N = 256$. Each batch of 32 images yields 8192 discrete latents which are jointly compressed by simulating $P_{Y|X}(\cdot \mid x) = \mathcal{N}(\mu_x, \mathrm{diag}(\sigma_x^2))$ using the polar-MLC implementation of the permuted scheme from Section 4, where the Sinkhorn-Knopp approximation is used for the discrete posterior. In Fig. 2 we compare against individually trained deterministic VQ-VAEs with smaller codebooks. By way of ablation, we also evaluate the same variable-rate stochastic models replacing our algorithm with coupled IS (Phan et al., 2024), closely related to ordered random coding (Theis & Ahmed, 2022). Our method smoothly achieves rates between 1.6 and 7.8 bits/latent while outperforming the IS baseline with varying chunk sizes and incurring only a small distortion penalty over individually trained fixed-rate models. The experimental setup is discussed in Appendix E.3, where we also show decreased encoding times versus the IS baseline.



ε	Polar-MLC (s)	PFR (s)
0.05	3.06 ± 0.32	58.7 ± 7.8
2.19	2.96 ± 0.12	182.2 ± 40.0
4.33	2.93 ± 0.03	1852.8 ± 75.5
6.47	3.03 ± 0.24	$13\,148.9 \pm 299.0$
8.61	2.93 ± 0.03	$59\,890.6 \pm 1801.4$

Figure 3: DME using the Gaussian mechanism. *Left:* Communication rate as a function of the privacy ε. P05–P95 rates are shown. *Right:* Encoding time in seconds and standard deviation for 128 blocks of 4096 samples. PFR splits the client vector into chunks of either 8 samples during the benchmark, or 4–12 samples as a function of ε for the rate comparison. More details are in Appendix E.4.

5.2 Distributed Mean Estimation with Central Differential Privacy

DME is a fundamental primitive in private distributed learning and optimization (Abadi et al., 2016; McMahan et al., 2017). In a general DME setting under DP, C clients each hold data $X_i \in \mathcal{X}$ and an aggregator aims to estimate the mean $\mu = \sum_{i=1}^{C} X_i/C$ subject to privacy constraints. A DP mechanism $P_{Y|X}$ is applied identically to each X_i to give a randomized output Y_i, yielding the estimate $\hat{\mu} = \sum_{i=1}^{C} Y_i/C$. Given the desired privacy ε, the aim is to minimize the expected squared error $\mathbb{E}[\|\hat{\mu} - \mu\|_2^2]$. We consider CDP, where the aggregator is trusted and releases $\hat{\mu}$ to an untrusted party. Although this setting is weaker than LDP, where the aggregator itself is also untrusted, in practice it allows for significantly more accurate estimators (Chen et al., 2023).

Our experiments adopt a similar setting to Chen et al. (2023) and Liu et al. (2024), using the Gaussian mechanism for (ε, δ)-CDP with $\delta = 10^{-6}$. We have $C = 10^5$ clients and the dimension of a local vector X_i is $D = 512$. Each coordinate X_{ij} is independent and uniform on a set of 16 values evenly spaced in $[-1, 1]$. Compared to the binary vectors in Liu et al. (2024), this larger alphabet size has relevance to emerging applications in ultra-low-precision, e.g. INT4, federated learning (Ding et al., 2025). To reduce communication overhead, we assume $B = 8$ instances of the DME problem can be batched and encoded jointly, giving a coding blocklength up to $BD = 4096$. In Fig. 3, we plot the bits per sample for $\varepsilon \in [0.05, 30]$ and provide wall-clock encoding times, excluding entropy coding for ease of comparison. We take PFR as a channel simulation baseline; note that the Poisson private representation (Liu et al., 2024) has a higher expected rate than PFR and is unnecessary for CDP. Both algorithms tested perform exact simulation and therefore provably achieve the same expected squared error as the uncompressed Gaussian mechanism at a given ε. Our polar-MLC permuted scheme achieves lower rates across the operating range, largely thanks to its ability to jointly encode 4096 samples, and is also faster. More details are in Appendix E.4, including a comparison with dithered quantization which, while not a general-purpose channel simulator, has also been used for compressed DME (Hasırcıoğlu & Gündüz, 2024).

6 Conclusion

In this work, we have considered exact and approximate channel simulation with a discrete input alphabet and introduced the permuted scheme to solve both variants while using a fixed number of shared random samples. Following this, we applied techniques from coding theory, namely polar and multi-level codes, to scale the permuted scheme to long blocklengths. Our channel simulator was demonstrated in two relevant machine learning applications: variable-rate neural image compression with VQ-VAEs and communication-efficient DME with privacy guarantees via the Gaussian mechanism. While the general channel simulation problem remains computationally difficult, our work shows that searching for scalable constructions on narrower yet practically relevant classes of channels may constitute a promising direction for continued research.

AI USAGE STATEMENT

In this work, we used generative AI tools to explore several ideas and alternative representations while developing the technical proof of Theorem 2 in the supplementary material. The overall argument, proof writing, as well as the other proofs in the paper, were not assisted by AI. We did not use generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, design or provide feedback on the methodology or experiments, implement methods, and interpret results. Other required-disclosure categories including generation of datasets, qualitative analysis and translation are not applicable. Additionally, among recommended-disclosure categories, we used AI tools for brainstorming, minor assistance with coding and LaTeX syntax, information search regarding known results from literature, and proofreading. We have reviewed all work assisted by AI tools. More precisely, any mathematical ideas proposed or retrieved by AI were checked against the appropriate sources for applicability, correctness and attribution, and all AI-assisted code was reviewed and tested by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we provide anonymized source code along with our submission as supplementary material, which allows the experiments in the main paper to be reproduced. This supplementary material also contains the trained model checkpoints used to obtain the neural compression results in Section 5.1, along with data-processing scripts used in the process of creating the various figures in the paper. Concerning the theoretical results, complete proofs stating any necessary assumptions are in Appendix B. The experimental setups for the empirical sections, Sections 5.1 and 5.2, are also documented extensively in Appendix E to aid reproducibility and provide additional details about how and under what settings our results are reported.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Zhang Li. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016.

Jason Altschuler, Jonathan Niles-Weed, and Philippe Rigollet. Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Erdal Arıkan. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7): 3051–3073, 2009.

Erdal Arıkan. Source polarization. In *IEEE International Symposium on Information Theory*, pp. 899–903, 2010.

Charles H. Bennett, Peter W. Shor, John A. Smolin, and Ashish V. Thapliyal. Entanglement-assisted capacity of a quantum channel and the reverse Shannon theorem. *IEEE Transactions on Information Theory*, 48(10):2637–2655, 2002.

Sourav Chatterjee and Persi Diaconis. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

Wei-Ning Chen, Dan Song, Ayfer Özgür, and Peter Kairouz. Privacy amplification via compression: achieving the optimal privacy-accuracy-communication trade-off in distributed mean estimation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 69202–69227, 2023.

W. J. Conover. *Practical nonparametric statistics*. Wiley, New York, 3 edition, 1999.

Li Ding, Hao Zhang, Wenrui Dai, Chenglin Li, Weijia Lu, Zhifei Yang, Xiaodong Zhang, Xiaofeng Ma, Junni Zou, and Hongkai Xiong. LBI-FL: low-bit integerized federated learning with temporally dynamic bit-width allocation. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 13885–13899, 2025.

Alexey Dosovitskiy and Josip Djolonga. You only train once: loss-conditional training of deep networks. In *International Conference on Learning Representations*, 2020.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pp. 265–284, 2006.

Vitaly Feldman and Kunal Talwar. Lossless compression of efficient private local randomizers. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 3208–3219, 2021.

Gergely Flamich. Greedy Poisson rejection sampling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 37089–37127, 2023.

Gergely Flamich and Deniz Gündüz. Data compression with stochastic codes. *IEEE BITS the Information Theory Magazine*, pp. 1–14, 2026.

Gergely Flamich and Lucas Theis. Adaptive greedy rejection sampling. In *IEEE International Symposium on Information Theory*, pp. 454–459. IEEE, 2023.

Gergely Flamich, Marton Havasi, and José Miguel Hernández-Lobato. Compressing images by encoding their latent representations with relative entropy coding. In *Advances in Neural Information Processing Systems*, volume 33, pp. 16131–16141, 2020.

Gergely Flamich, Stratis Markou, and José Miguel Hernández-Lobato. Fast relative entropy coding with A* coding. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 6548–6577, 2022.

Gergely Flamich, Stratis Markou, and José Miguel Hernández-Lobato. Faster relative entropy coding with greedy rejection coding. *Advances in Neural Information Processing Systems*, 36: 50558–50569, 2023.

Robert M. Gray and Thomas G. Stockham. Dithered quantizers. *IEEE Transactions on Information Theory*, 39(3):805–812, 1993.

Prahladh Harsha, Rahul Jain, David McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. *IEEE Transactions on Information Theory*, 56(1):438–449, 2010.

Burak Hasırcıoğlu and Deniz Gündüz. Communication efficient private federated learning using dithering. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7575–7579, 2024.

Marton Havasi, Robert Peharz, and José Miguel Hernández-Lobato. Minimal random code learning: getting bits back from compressed model parameters. In *International Conference on Learning Representations*, 2019.

Mahmoud Hegazy and Cheuk Ting Li. Randomized quantization with exact error distribution. In *2022 IEEE Information Theory Workshop*, pp. 350–355, 2022.

Mahmoud Hegazy, Rémi Leluc, Cheuk Ting Li, and Aymeric Dieuleveut. Compression with exact error distribution for federated learning. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, pp. 613–621, 2024.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Spencer Hill, Fady Alajaji, Tamás Linder, and Gergely Flamich. Rejection sampling is optimal for relative entropy coding, 2026. URL `https://arxiv.org/abs/2604.23076`.

Philip A. Knight. The Sinkhorn–Knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008.

Szymon Kobus, Lucas Theis, and Deniz Gündüz. Gaussian channel simulation with rotated dithered quantization. In *IEEE International Symposium on Information Theory*, pp. 1907–1912, 2024.

Cheuk Ting Li. Channel simulation: theory and applications to lossy compression and differential privacy. *Foundations and Trends in Communications and Information Theory*, 21(6):847–1106, 2024.

Cheuk Ting Li and Abbas El Gamal. Strong functional representation lemma and applications to coding theorems. *IEEE Transactions on Information Theory*, 64(11):6967–6978, 2018.

Chih Wei Ling, Chun Hei Michael Shiu, Youqi Wu, Jiande Sun, Cheuk Ting Li, Linqi Song, and Weitao Xu. Communication-efficient and privacy-adaptable mechanism for federated learning, 2025. URL https://arxiv.org/abs/2501.12046.

Yanxiao Liu, Wei-Ning Chen, Ayfer Özgür, and Cheuk Ting Li. Universal exact compression of differentially private mechanisms. In *Advances in Neural Information Processing Systems*, volume 37, pp. 91492–91531, 2024.

M. M. Mamatov. Estimation of the remainder term of the generalized Moivre-Laplace local limit theorem. *Fergan. Gos. Ped. Inst. Učen. Zap. Ser. Mat.*, 1:63–66, 1965. MR0200962 (34 847).

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with Gumbel-Sinkhorn networks. In *International Conference on Learning Representations*, 2018.

Gonzalo Mena, Erdem Varol, Amin Nejatbakhsh, Eviatar Yemini, and Liam Paninski. Sinkhorn permutation variational marginal inference. In *Proceedings of the 2nd Symposium on Advances in Approximate Bayesian Inference*, pp. 1–9, 2020.

Ezgi Ozyilkan, Sharang M. Sriramu, Elza Erkip, Aaron B. Wagner, and Jona Ballé. SoftBinary coding: a new information-theoretic neural compression paradigm. In *43rd International Conference on Machine Learning*, 2026.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: visual reasoning with a general conditioning layer. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

Buu Phan and Ashish Khisti. Channel simulation and distributed compression with ensemble rejection sampling. In *Advances in Neural Information Processing Systems*, volume 38, pp. 43066–43123, 2025.

Buu Phan, Ashish Khisti, and Christos Louizos. Importance matching lemma for lossy compression with side information. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, pp. 1387–1395, 2024.

Yury Polyanskiy and Yihong Wu. *Information theory: from coding to learning*. Cambridge University Press, Cambridge, 2025.

Jorma Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, 20(3):198–203, 1976.

Lawrence Roberts. Picture coding using pseudo-random noise. *IRE Transactions on Information Theory*, 8(2):145–154, 1962.

Abhin Shah, Wei-Ning Chen, Johannes Ballé, Peter Kairouz, and Lucas Theis. Optimal compression of locally differentially private mechanisms. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, pp. 7680–7723, 2022.

Ali Moradi Shahmiri, Chih Wei Ling, and Cheuk Ting Li. Communication-efficient Laplace mechanism for differential privacy via random quantization. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4550–4554, 2024.

Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964.

Sharang M. Sriramu, Rochelle Barsz, Elizabeth Polito, and Aaron B. Wagner. Fast channel simulation via error-correcting codes. In *Advances in Neural Information Processing Systems*, volume 37, pp. 107932–107959, 2024.

Yuhta Takida, Takashi Shibuya, Weihsiang Liao, Chieh-Hsin Lai, Junki Ohmura, Toshimitsu Uesaka, Naoki Murata, Shusuke Takahashi, Toshiyuki Kumakura, and Yuki Mitsufuji. SQ-VAE: variational Bayes on discrete representation with self-annealed stochastic quantization. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 20987–21012, 2022.

Wenpin Tang and Fengmin Tang. The Poisson binomial distribution – old and new. *Statistical Science*, 38(1):108–119, 2023.

Lucas Theis and Noureldin Y. Ahmed. Algorithms for the communication of samples. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 21308–21328, 2022.

Lucas Theis, Tim Salimans, Matthew D. Hoffman, and Fabian Mentzer. Lossy compression with Gaussian diffusion, 2022. URL `https://arxiv.org/abs/2206.08889`.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jeremy Vonderfecht and Liu Feng. Lossy compression with pretrained diffusion models. In *International Conference on Learning Representations*, pp. 687–702, 2025.

Udo Wachsmann, Robert F. H. Fischer, and Johannes B. Huber. Multilevel codes: theoretical concepts and practical design rules. *IEEE Transactions on Information Theory*, 45(5):1361–1391, 1999.

Andreas Winter. Compression of sources of probability distributions and density operators, 2002. URL `https://arxiv.org/abs/quant-ph/0208131`.

Tongda Xu, Wendi Zheng, Jiajun He, José Miguel Hernández-Lobato, Yan Wang, Ya-Qin Zhang, and Jie Tang. Training-free vector quantization via Gaussian VAEs. In *43rd International Conference on Machine Learning*, 2026.

Jianguo Zhao and Cheuk Ting Li. Rejection-sampled linear codes for channel simulation. In *IEEE International Symposium on Information Theory*, pp. 1–6, 2026.

A BI-AWGN EXAMPLE

Here, we consider a concrete worked example focusing on simulating a BI-AWGN channel, as pictured in Fig. 4, to motivate the use of the permuted scheme to reduce the channel simulation rate while maintaining the correct output distribution using only $N = 2$ shared Gaussian samples. In this example, the physical channel input $W \in \{-1, 1\}$ is uniformly distributed, and the channel noise Z is a 1D Gaussian with variance σ^2. To harmonize the problem with the indexing convention used in Section 3, we map W to a canonical channel input $X \in \{1, 2\}$, with $X = (W + 1)/2 + 1$. We have $H(X) = 1$, and the channel capacity can be shown to be, see e.g. Polyanskiy & Wu (2025, Example 3.4),

$$I(X;Y) = 1 - \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-z^2/2} \log(1 + e^{2z/\sigma - 2/\sigma^2})\, dz. \tag{15}$$

We consider the one-shot channel simulation methods proposed in Section 3 in turn, beginning with the naive scheme, continuing to the source-dependent index randomization idea, and concluding by demonstrating how introducing a random permutation finally succeeds in driving the rate below $H(X)$.

Naive Scheme. In this particular example, the naive scheme consists of sampling

$$\bar{U}_1 \sim P_{Y|X}(\cdot \mid 1) = \mathcal{N}(-1, \sigma^2) \text{ and } \bar{U}_2 \sim P_{Y|X}(\cdot \mid 2) = \mathcal{N}(1, \sigma^2). \tag{16}$$

Upon observing the index $X = x$, the encoder would send $K = x$ to the decoder, which would then output $\bar{U}_K$. Clearly, if $X = 1$ the decoder's output correctly follows the density $p_{Y|X}(y \mid 1) = \mathcal{N}(y; -1, \sigma^2)$, and if $X = 2$ it follows $p_{Y|X}(y \mid 2) = \mathcal{N}(y; 1, \sigma^2)$. However, since K is always equal to X, which in turn is independent of $\bar{U}^2$, it is not possible to transmit K to the decoder using less than $H(X) = 1$ bits. Thus, while the naive scheme produces correctly distributed output samples, it does not achieve any compression whatsoever compared to directly sending X and later adding noise at the decoder.

Randomizing the Index Based on the Source. As briefly mentioned in Section 3, a first idea is to continue generating $\bar{U}_1 \sim \mathcal{N}(-1, \sigma^2)$ and $\bar{U}_2 \sim \mathcal{N}(1, \sigma^2)$, but additionally randomize the index K given X at the encoder in an attempt to reduce the communication cost. To formalize this, we allow the encoder, after observing $X = x$, to sample K from an arbitrary conditional distribution $P_{K|X}(\cdot \mid x)$. Without loss of generality, assume $X = 1$. Let $\alpha = P_{K|X}(1 \mid 1)$. The density of the output, which we now write as $\tilde{Y}$ to avoid confusion with the target distribution $P_{Y|X}$, is then

$$\begin{aligned} p_{\tilde{Y}|X}(y \mid 1) &= \iint (\alpha\delta(y - u_1) + (1 - \alpha)\delta(y - u_2)) p_{\bar{U}^2}(u^2)\, du_1 du_2 && (17) \\ &= \alpha \int \delta(y - u_1)\mathcal{N}(u_1; -1, \sigma^2)\, du_1 \int \mathcal{N}(u_2; 1, \sigma^2)\, du_2 && (18) \\ &\quad + (1 - \alpha) \int \delta(y - u_2)\mathcal{N}(u_2; 1, \sigma^2)\, du_2 \int \mathcal{N}(u_1; -1, \sigma^2)\, du_1 \\ &= \alpha\mathcal{N}(y; -1, \sigma^2) + (1 - \alpha)\mathcal{N}(y; 1, \sigma^2). && (19) \end{aligned}$$

The only solution yielding the desired output density $p_{\tilde{Y}|X}(y \mid 1) = \mathcal{N}(y; -1, \sigma^2)$ for all y is the one-hot solution with $\alpha = 1$. The same argument can be repeated for $X = 2$, showing that an exact simulation requires $P_{K|X}(k \mid x) = \delta(k - x)$, i.e. $K = X$ just as in the naive scheme. As a result, the best possible rate with this setup is again $H(X) = 1$, and the index randomization gives no advantage over directly sending X losslessly and adding noise after the fact.

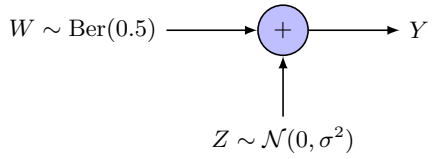


Figure 4: System model for the BI-AWGN channel.

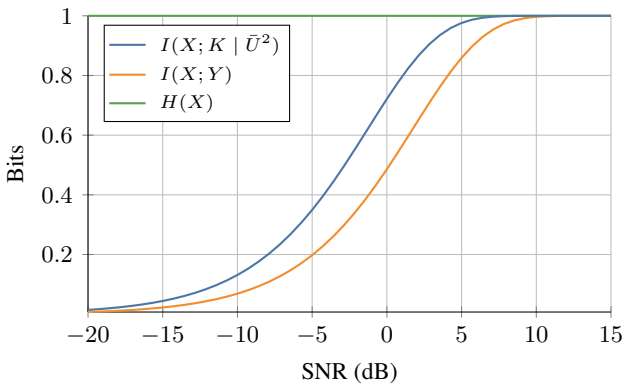


Figure 5: Conditional mutual information, capacity and source entropy for a BI-AWGN channel. The conditional mutual information $I(X;K \mid \bar{U}^2)$ is a lower bound on the permuted scheme's rate.

Introducing a Random Permutation. We now show how our alternative idea of randomly shuffling $\bar{U}^2$ allows us to achieve a communication cost less than the entropy. As discussed in Section 3.1, we introduce a random permutation of the samples which hides the assignment between the sample indices and the two output modes. After this, the samples are marginally identically distributed but not independent. We then have $P_{K|X}(k \mid x) = 1/2$ for all k and x as, under the random permutation, $\bar{U}_1$ is equally likely to have been sampled from mode 1 or 2, and likewise for $\bar{U}_2$. In this particular binary-input case, Π is a uniform permutation of $\{1, 2\}$, meaning:

1. With probability $1/2$, $\bar{U}_1 \sim \mathcal{N}(-1, \sigma^2)$ and $\bar{U}_2 \sim \mathcal{N}(1, \sigma^2)$.
2. With probability $1/2$, $\bar{U}_1 \sim \mathcal{N}(1, \sigma^2)$ and $\bar{U}_2 \sim \mathcal{N}(-1, \sigma^2)$.

We visualize this procedure in Fig. 6. To sample K, the permuted scheme as described in Section 3.1 uses the Bayesian posterior, the probability of $K = k$ being set equal to the probability given $\bar{U}^2$ and X that $\bar{U}_k$ was the sample taken from the conditional distribution $P_{Y|X}(\cdot \mid X)$. For our BI-AWGN simulation, this is

$$P_{K|X,\bar{U}^2}(k \mid x, u^2) = \Pr[\Pi(x) = k \mid \bar{U}^2 = u^2] = \frac{p_{\bar{U}^2}(u^2 \mid \Pi(x) = k) \Pr[\Pi(x) = k]}{p_{\bar{U}^2}(u^2)}. \quad (20)$$

Since Π is uniform, $\Pr[\Pi(x) = k] = 1/2$, and the unconditional distribution of $\bar{U}^2$ is the same as that induced by sampling without replacement from the two modes. As before, assuming $X = 1$, we evaluate

$$P_{K|X,\bar{U}^2}(1 \mid 1, u^2) = \frac{\mathcal{N}(u_1; -1, \sigma^2)\mathcal{N}(u_2; 1, \sigma^2)}{\mathcal{N}(u_1; -1, \sigma^2)\mathcal{N}(u_2; 1, \sigma^2) + \mathcal{N}(u_1; 1, \sigma^2)\mathcal{N}(u_2; -1, \sigma^2)} \quad (21)$$

$$P_{K|X,\bar{U}^2}(2 \mid 1, u^2) = \frac{\mathcal{N}(u_1; 1, \sigma^2)\mathcal{N}(u_2; -1, \sigma^2)}{\mathcal{N}(u_1; -1, \sigma^2)\mathcal{N}(u_2; 1, \sigma^2) + \mathcal{N}(u_1; 1, \sigma^2)\mathcal{N}(u_2; -1, \sigma^2)}. \quad (22)$$

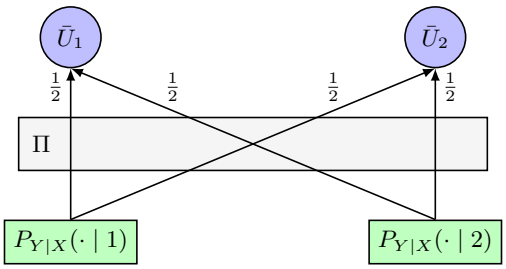


Figure 6: Illustration of the random permutation Π used by our scheme on the BI-AWGN channel.

Substituting (21), (22) into the expression for the algorithm's output distribution,

$$p_{\tilde{Y}|X}(y \mid 1) = \iint (\delta(y-u_1)P_{K|X,\bar{U}^2}(1 \mid 1, u^2) + \delta(y-u_2)P_{K|X,\bar{U}^2}(2 \mid 1, u^2))p_{\bar{U}^2}(u^2)\, du_1 du_2 \tag{23}$$

confirms that the permuted approach indeed generates an output following $\mathcal{N}(-1, \sigma^2)$ when $X = 1$, and by repeating the same steps, $\mathcal{N}(1, \sigma^2)$ when $X = 2$. The lower bound on the rate is the conditional mutual information $I(X; K \mid \bar{U}^2)$, and this may now be lower than $H(X)$. We prove this lower bound later in Proposition 4, noting that for continuous output distributions $I(X; K \mid \bar{U}^2) = I(X; Y \mid \bar{U}^2)$. For the BI-AWGN channel in particular, using the posterior distributions in (21), (22) and computing the mutual information gives

$$I(X; K \mid \bar{U}^2) = 1 - \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-z^2/2} \log(1 + e^{2\sqrt{2}z/\sigma - 4/\sigma^2})\, dz. \tag{24}$$

To give an idea of the relative size of these quantities, we plot $I(X; K \mid \bar{U}^2)$, $I(X; Y)$ and $H(X)$ in Fig. 5, for signal-to-noise ratios (SNRs) between -20 and 15, where $\mathrm{SNR} = -10\log_{10}(\sigma^2)$. At both low and high SNRs, $I(X; K \mid \bar{U}^2)$ becomes very close to the channel capacity.

B Proof of Results in the Main Paper

B.1 Proof of Proposition 1

Suppose $X = 1$ without loss of generality and let the output be denoted $\tilde{Y}$ throughout this section so as not to confuse its distribution with $P_{Y|X}$. The density function of $\tilde{Y}$ conditioned on $X = 1$ and the shared samples being $\bar{U}^N = u^N$ is

$$p_{\tilde{Y}|X,\bar{U}^N}(y \mid 1, u^N) = \sum_{k=1}^{N} \delta(y - u_k) \Pr[\Pi(1) = k \mid \bar{U}^N = u^N]. \tag{25}$$

Integrating out the joint density of the samples,

$$p_{\tilde{Y}|X}(y \mid 1) = \int \cdots \int \sum_{k=1}^{N} \delta(y - u_k) \Pr[\Pi(1) = k \mid \bar{U}^N = u^N] p_{\bar{U}^N}(u^N)\, du_1 \cdots du_N. \tag{26}$$

Applying Bayes' rule,

$$\Pr[\Pi(1) = k \mid \bar{U}^N = u^N] = \frac{p_{\bar{U}^N}(u^N \mid \Pi(1) = k) \Pr[\Pi(1) = k]}{p_{\bar{U}^N}(u^N)} = \frac{p_{\bar{U}^N}(u^N \mid \Pi(1) = k)}{N p_{\bar{U}^N}(u^N)}. \tag{27}$$

From this,

$$p_{\tilde{Y}|X}(y \mid 1) = \frac{1}{N} \int \cdots \int \sum_{k=1}^{N} \delta(y - u_k) p_{\bar{U}^N}(u^N \mid \Pi(1) = k)\, du_1 \cdots du_N \tag{28}$$

$$= \frac{1}{N} \sum_{k=1}^{N} \int \cdots \int \delta(y - u_k) p_{\bar{U}^N}(u^N \mid \Pi(1) = k)\, du_1 \cdots du_N. \tag{29}$$

Now, as the analysis of the inner nested integral will be the same for any k, let us fix $k = 1$ without loss of generality. We find

$$\int \cdots \int \delta(y - u_1) p_{\bar{U}^N}(u^N \mid \Pi(1) = 1)\, du_1 \cdots du_N$$

$$= \int \delta(y - u_1) p_{\bar{U}_1}(u_1 \mid \Pi(1) = 1)\, du_1 \int \cdots \int p_{\bar{U}^{N\setminus\{1\}}}(u^{N\setminus\{1\}} \mid \Pi(1) = 1)\, du_2 \cdots du_N \tag{30}$$

$$= p_{\bar{U}_1}(y \mid \Pi(1) = 1) \tag{31}$$

$$= p_{Y|X}(y \mid 1) \tag{32}$$

where $\bar{U}^{N\setminus\{1\}}$ denotes the set of all samples excluding $\bar{U}_1$. Thus, since the same simplification works for any k,

$$p_{\tilde{Y}|X}(y \mid 1) = \frac{1}{N} \sum_{k=1}^{N} p_{Y|X}(y \mid 1) = p_{Y|X}(y \mid 1). \tag{33}$$

As the same argument holds for any $X \in [N]$, the output follows the required distribution. ■

B.2 Proof of Theorem 1

We begin by proving the first part of the theorem by bounding the expected size of the index $\bar{K}$ into the sorted list in Algorithm 1. Applying EFR to communicate the sampled index with fixed target and prior distributions P and Q respectively yields the bound $\mathbb{E}[\log \bar{K}] \leq D_{\mathrm{KL}}(P\|Q) + 1$ (Li, 2024, Lemma 12). In our case, given $X = x$ and $\bar{U}^N = u^N$, we have $P(k) = P_{K|X,\bar{U}^N}(k \mid x, u^N)$ and $Q(k) = P_{K|\bar{U}^N}(k \mid u^N)$. This gives

$$\mathbb{E}[\log \bar{K} \mid X, \bar{U}^N] \leq D_{\mathrm{KL}}(P_{K|X,\bar{U}^N}(\cdot \mid X, \bar{U}^N)\|P_{K|\bar{U}^N}(\cdot \mid \bar{U}^N)) + 1. \tag{34}$$

It then only remains to take the expectation over X and $\bar{U}^N$ in order to bound the unconditional expected size. In particular,

$$\mathbb{E}_{X,\bar{U}^N}[D_{\mathrm{KL}}(P_{K|X,\bar{U}^N}(\cdot \mid X, \bar{U}^N)\|P_{K|\bar{U}^N}(\cdot \mid \bar{U}^N))] = I(X; K \mid \bar{U}^N). \tag{35}$$

Thus,

$$\mathbb{E}[\log \bar{K}] = \mathbb{E}_{X,\bar{U}^N}[\mathbb{E}[\log \bar{K} \mid X, \bar{U}^N]] \leq I(X; K \mid \bar{U}^N) + 1 = I(X; Y \mid \bar{U}^N) + 1 \tag{36}$$

where the last equality follows as given the set of samples $\bar{U}^N$, the mapping from K to Y is a bijection with probability one; as our setting assumes a continuous output distribution, ties within $\bar{U}^N$ occur with probability zero.

Concerning the second part of the theorem which deals with the expected message length, we make use of the following result which bounds the cross-entropy with a carefully-chosen Zipf distribution.

Proposition 3 (Li & Gamal, 2018, Proposition 4)**.** *For a random variable Z following the distribution P_Z and satisfying $\mathbb{E}[\log Z] \leq \ell$, its cross-entropy with $\mathrm{Zipf}(1 + 1/\ell)$ satisfies*

$$H(Z) \leq H(P_Z, \mathrm{Zipf}(1 + 1/\ell)) \leq \ell + \log(\ell + 1) + 1. \tag{37}$$

As a result, if we use a Shannon code designed for $\mathrm{Zipf}(1 + 1/(I(X; Y \mid \bar{U}^N) + 1))$ as the `enc`, `dec` pair in Algorithm 1, it satisfies the upper bound on the encoding length in (1). ■

B.3 Proof of Proposition 2

In this section, we prove that the multi-sample permuted scheme performs exact channel simulation. The approach is essentially the same as that used in Appendix B.1 apart from the slight modification to the posterior distribution over K_m, which accounts for the presence of several samples per mode. Supposing $X = 1$ without loss of generality and letting the sample output by the algorithm be $\tilde{Y}$, the conditional density function of $\tilde{Y}$ is

$$p_{\tilde{Y}|X,\bar{U}^{Nm}}(y \mid 1, u^{Nm}) = \sum_{k=1}^{Nm} \delta(y - u_k) P_{K_m|X,\bar{U}^{Nm}}(k \mid 1, u^{Nm}) \tag{38}$$

$$= \frac{1}{m} \sum_{k=1}^{Nm} \delta(y - u_k) \Pr[k \in \Pi(1) \mid \bar{U}^{Nm} = u^{Nm}]. \tag{39}$$

Then, as in the proof of Proposition 1, we integrate out the joint density of the Nm samples to get

$$p_{\tilde{Y}|X}(y \mid 1) = \int \cdots \int \frac{1}{m} \sum_{k=1}^{Nm} \delta(y - u_k) \Pr[k \in \Pi(1) \mid \bar{U}^{Nm} = u^{Nm}] \, du_1 \cdots du_{Nm} \tag{40}$$

and apply Bayes' rule to write

$$\Pr[k \in \Pi(1) \mid \bar{U}^{Nm} = u^{Nm}] = \frac{p_{\bar{U}^{Nm}}(u^{Nm} \mid k \in \Pi(1))}{N p_{\bar{U}^{Nm}}(u^{Nm})} \tag{41}$$

in order to make the substitution

$$p_{\tilde{Y}|X}(y \mid 1) = \frac{1}{Nm} \sum_{k=1}^{Nm} \int \cdots \int \delta(y - u_k) p_{\bar{U}^{Nm}}(u^{Nm} \mid k \in \Pi(1)) \, du_1 \cdots du_{Nm}. \tag{42}$$

Analyzing the inner integral when $k = 1$ without loss of generality,

$$\begin{aligned}
&\int \cdots \int \delta(y - u_1) p_{\bar{U}^{Nm}}(u^{Nm} \mid 1 \in \Pi(1))\, du_1 \cdots du_{Nm} \\
&= \int \delta(y - u_1) p_{\bar{U}_1}(u_1 \mid 1 \in \Pi(1))\, du_1 \\
&\quad \times \int \cdots \int p_{\bar{U}^{Nm\setminus\{1\}}}(u^{Nm\setminus\{1\}} \mid 1 \in \Pi(1))\, du_2 \cdots du_{Nm} && (43) \\
&= p_{\bar{U}_1}(y \mid 1 \in \Pi(1)) && (44) \\
&= p_{Y|X}(y \mid 1). && (45)
\end{aligned}$$

As the same simplification holds for any k, we find

$$p_{\tilde{Y}|X}(y \mid 1) = \frac{1}{Nm} \sum_{k=1}^{Nm} p_{Y|X}(y \mid 1) = p_{Y|X}(y \mid 1). \tag{46}$$

The same argument holds for any input $X \in [N]$. ■

B.4 Proof of Theorem 2

As mentioned in the theorem statement, we limit our attention to the case of $N = 2$. Following the steps used to prove Theorem 1 in Appendix B.2, with the target distribution given $X = x$ and $\bar{U}^{2m} = u^{2m}$ being $P(k) = P_{K_m|X,\bar{U}^{2m}}(k \mid x, u^{2m})$ as in (4), and the prior being $Q(k) = P_{K_m|\bar{U}^{2m}}(k \mid u^{2m})$, the expected message length when there are m samples per mode satisfies

$$\mathbb{E}[\ell_m(M)] \leq I(X; K_m \mid \bar{U}^{2m}) + \log(I(X; K_m \mid \bar{U}^{2m}) + 2) + 3. \tag{47}$$

Expanding, we note that

$$I(X; K_m \mid \bar{U}^{2m}) = H(X) - H(X \mid K_m, \bar{U}^{2m}) \tag{48}$$

as X is independent of $\bar{U}^{2m}$. Then, the term depending on m is

$$H(X \mid K_m, \bar{U}^{2m}) = \mathbb{E}[h_B(p_m(K_m, \bar{U}^{2m}))] \tag{49}$$

where $h_B(p) = -p \log p - (1 - p) \log(1 - p)$ is the binary entropy function on $p \in [0, 1]$ and $p_m(k, u^{2m}) = P_{X|K_m,\bar{U}^{2m}}(1 \mid k, u^{2m})$. Also define $q_m(k, u^{2m}) = P_{X|Y}(1 \mid u_k)$. To continue, we make use of the following lemma, the proof of which is deferred to Appendix B.5.

Lemma 1. *We have that*

$$\lim_{m\to\infty} |p_m(K_m, \bar{U}^{2m}) - q_m(K_m, \bar{U}^{2m})| = 0 \text{ a.s.} \tag{50}$$

As $h_B(p)$ is uniformly continuous on $[0, 1]$, Lemma 1 implies that

$$\lim_{m\to\infty} |h_B(p_m(K_m, \bar{U}^{2m})) - h_B(q_m(K_m, \bar{U}^{2m}))| = 0 \text{ a.s.} \tag{51}$$

Furthermore, as $h_B(p)$ is bounded between 0 and 1, the difference is also bounded. Applying the bounded convergence theorem then gives

$$\begin{aligned}
&\lim_{m\to\infty} \mathbb{E}[h_B(p_m(K_m, \bar{U}^{2m})) - h_B(q_m(K_m, \bar{U}^{2m}))] \\
&= \mathbb{E}\Big[\lim_{m\to\infty} (h_B(p_m(K_m, \bar{U}^{2m})) - h_B(q_m(K_m, \bar{U}^{2m})))\Big] && (52) \\
&= 0. && (53)
\end{aligned}$$

But, for arbitrary m,

$$\begin{aligned}
\mathbb{E}[h_B(q_m(K_m, \bar{U}^{2m}))] &= \mathbb{E}_{K_m,\bar{U}^{2m}}[h_B(P_{X|Y}(1 \mid \bar{U}_{K_m}))] && (54) \\
&= \mathbb{E}_{\bar{U}_{K_m}}[h_B(P_{X|Y}(1 \mid \bar{U}_{K_m}))] && (55) \\
&= \mathbb{E}_{Y\sim P_Y}[h_B(P_{X|Y}(1 \mid Y))] && (56) \\
&= H(X \mid Y) && (57)
\end{aligned}$$

where in (56) we have used the fact that, as $\bar{U}_{K_m}$ is the output selected by the multi-sample permuted scheme when there are m samples per mode, the guarantee of exact channel simulation from Proposition 2 ensures that $\bar{U}_{K_m} \sim P_Y$ for any m. It follows that

$$\lim_{m\to\infty} \mathbb{E}[h_B(p_m(K_m, \bar{U}^{2m}))] = H(X \mid Y). \tag{58}$$

Finally, substituting into (48), then (47), and using continuity, we arrive at the desired result. ■

B.5 Proof of Lemma 1

We start by dispensing with a trivial case. If the intersection of the supports of the conditional densities at the two modes, $p_{Y|X}(\cdot \mid 1)$ and $p_{Y|X}(\cdot \mid 2)$, has zero probability, then the output $Y = \bar{U}_{K_m}$ immediately determines the input X. In this case,

$$P_{X|K_m,\bar{U}^{2m}}(1 \mid K_m, \bar{U}^{2m}) = P_{X|Y}(1 \mid \bar{U}_{K_m}) = \mathbb{1}\{\bar{U}_{K_m} \in \operatorname{supp} p_{Y|X}(\cdot \mid 1)\} \tag{59}$$

and the required equality holds trivially for all m.

Otherwise, we start by evaluating the pointwise posterior probability for a given $k \in [2m]$, and shared randomness realization $\bar{U}^{2m} = u^{2m}$. Using Bayes' rule,

$$P_{X|K_m,\bar{U}^{2m}}(1 \mid k, u^{2m}) = \frac{P_{K_m|X,\bar{U}^{2m}}(k \mid 1, u^{2m}) P_X(1)}{\sum_{x=1}^{2} P_{K_m|X,\bar{U}^{2m}}(k \mid x, u^{2m}) P_X(x)}. \tag{60}$$

Substituting the posterior distribution as defined in (4), we get

$$P_{X|K_m,\bar{U}^{2m}}(1 \mid k, u^{2m}) = \frac{\Pr[k \in \Pi(1) \mid \bar{U}^{2m} = u^{2m}] P_X(1)}{\sum_{x=1}^{2} \Pr[k \in \Pi(x) \mid \bar{U}^{2m} = u^{2m}] P_X(x)}. \tag{61}$$

We next turn our attention to computing $\Pr[k \in \Pi(x) \mid \bar{U}^{2m} = u^{2m}]$ which, as detailed in Section 3.2 for the single-sample permuted scheme, involves the computation of likelihood matrix permanents. To make this precise, by applying Bayes' rule a second time after noting that $\Pr[k \in \Pi(x)] = 1/2$ for any k and x, we see that

$$\Pr[k \in \Pi(x) \mid \bar{U}^{2m} = u^{2m}] = \frac{p_{\bar{U}^{2m}}(u^{2m} \mid k \in \Pi(x))}{\sum_{x'=1}^{2} p_{\bar{U}^{2m}}(u^{2m} \mid k \in \Pi(x'))}. \tag{62}$$

As in Appendix B.3, let $\bar{U}^{2m\setminus\{k\}}$ denote the set of all samples excluding $\bar{U}_k$. We can then express the conditional density function as

$$p_{\bar{U}^{2m}}(u^{2m} \mid k \in \Pi(x)) = p_{Y|X}(u_k \mid x) p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(x)). \tag{63}$$

Going back to (61), we get

$$P_{X|K_m,\bar{U}^{2m}}(1 \mid k, u^{2m}) = \frac{p_{\bar{U}^{2m}}(u^{2m} \mid k \in \Pi(1)) P_X(1)}{\sum_{x=1}^{2} p_{\bar{U}^{2m}}(u^{2m} \mid k \in \Pi(x)) P_X(x)} \tag{64}$$

$$= \frac{p_{Y|X}(u_k \mid 1) p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(1)) P_X(1)}{\sum_{x=1}^{2} p_{Y|X}(u_k \mid x) p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(x)) P_X(x)}. \tag{65}$$

Next define the likelihood ratio of the rejected samples as

$$\Lambda_m(k, u^{2m}) = \frac{p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(1))}{p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(2))}. \tag{66}$$

To analyze this ratio, we introduce the $2m \times 2m$ likelihood matrix $L(u^{2m})$ where $L_{ij} = p_{Y|X}(u_i \mid 1)$ for $1 \le j \le m$, and $p_{Y|X}(u_i \mid 2)$ for $m+1 \le j \le 2m$. This matrix contains only two unique columns, let these be $z^{(1)}$ and $z^{(2)}$, with

$$z^{(1)} = \begin{bmatrix} p_{Y|X}(u_1 \mid 1) \\ \vdots \\ p_{Y|X}(u_{2m} \mid 1) \end{bmatrix} \text{ and } z^{(2)} = \begin{bmatrix} p_{Y|X}(u_1 \mid 2) \\ \vdots \\ p_{Y|X}(u_{2m} \mid 2) \end{bmatrix}. \tag{67}$$

With this established, note that

1. $p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(1))$ is proportional to the permanent of the sub-matrix with row k and one of the columns equal to $z^{(1)}$ removed.
2. $p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(2))$ is proportional to the permanent of the sub-matrix with row k and one of the columns equal to $z^{(2)}$ removed.

Since the matrix permanent is a polynomial of the elements of the matrix, we can also express the desired probabilities in terms of a polynomial. Define

$$P_k(w) = \prod_{i\in[2m]\setminus\{k\}} (wz_i^{(1)} + z_i^{(2)}) = \sum_{j=0}^{2m-1} C_j^{(k)} w^j. \tag{68}$$

Each term after expanding the product represents the joint likelihood of a given assignment of all $2m-1$ remaining samples between the two modes; $p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(1))$ is proportional to the sum of all those which correspond to selecting $m-1$ likelihoods from $z^{(1)}$ and m from $z^{(2)}$. This sum is the coefficient $C_{m-1}^{(k)}$ in the polynomial. Similarly, $p_{\bar{U}^{2m\setminus\{k\}}}(u^{2m\setminus\{k\}} \mid k \in \Pi(2))$ is proportional to $C_m^{(k)}$, with the same constant of proportionality. Thus, the probability ratio becomes $\Lambda_m(k, u^{2m}) = C_{m-1}^{(k)}/C_m^{(k)}$.

To continue, we normalize each term of the polynomial to get

$$\hat{P}_k(w) = \prod_{i\in[2m]\setminus\{k\}} (wp_i + (1-p_i)) = \sum_{j=0}^{2m-1} \hat{C}_j^{(k)} w^j \tag{69}$$

where

$$p_i = \frac{z_i^{(1)}}{z_i^{(1)} + z_i^{(2)}} \text{ and } 1 - p_i = \frac{z_i^{(2)}}{z_i^{(1)} + z_i^{(2)}}. \tag{70}$$

We note that when the p_i's are derived from our random sample pool $\bar{U}^{2m}$, the event $\{z_i^{(1)} = z_i^{(2)} = 0\}$ has probability zero, since each sample is sourced from one of the modes and has a strictly positive density evaluated under the mode that produced it. Furthermore, as the intersection of the supports of $p_{Y|X}(\cdot \mid 1)$ and $p_{Y|X}(\cdot \mid 2)$ has nonzero probability outside of the trivial case already dealt with, we will have $\Pr[0 < p_i < 1] > 0$.

Now, we can analyze $\hat{P}_k(w)$ by recognizing that it is the probability generating function of a Poisson binomial random variable with $n = 2m-1$ trials (Tang & Tang, 2023). In particular, for $1 \le i \le 2m$, let X_i be a Bernoulli random variable with success probability p_i, and let

$$S_k(u^{2m}) = \sum_{i\in[2m]\setminus\{k\}} X_i. \tag{71}$$

Then $S_k(u^{2m})$ is a Poisson binomial random variable for each k, and its probability generating function is $\hat{P}_k(w)$. For $0 \le j \le n$, we have that $\hat{C}_j^{(k)} = \Pr[S_k(u^{2m}) = j]$ and so

$$\Lambda_m(k, u^{2m}) = \frac{\Pr[S_k(u^{2m}) = m-1]}{\Pr[S_k(u^{2m}) = m]}. \tag{72}$$

While (72) holds for a particular realization of $(K_m, \bar{U}^{2m})$, we are interested in the ensemble behavior when the full pool of $2m$ shared random samples is generated following $P_{\bar{U}^{2m}}$ and the index K_m is also random. The following lemma establishes the asymptotic properties that will allow us to adapt our results to this context.

Lemma 2. *Assume* $\Pr[0 < p_i < 1] > 0$. *Then,*

$$\lim_{m\to\infty} \sup_{1\le k\le 2m} \left| \frac{\Pr[S_k(\bar{U}^{2m}) = m-1]}{\Pr[S_k(\bar{U}^{2m}) = m]} - 1 \right| = 0 \text{ a.s.} \tag{73}$$

The uniform convergence result from Lemma 2 implies that $\lim_{m\to\infty} \Lambda_m(K_m, \bar{U}^{2m}) = 1$ almost surely. Consider an arbitrary sample path $\{u^{2m}, k_m\}_{m\ge1}$ for which $\lim_{m\to\infty} \Lambda_m(k_m, u^{2m}) = 1$, and define

$$p_m = p_m(k_m, u^{2m}) = P_{X|K_m,\bar{U}^{2m}}(1 \mid k_m, u^{2m}) \tag{74}$$

$$q_m = q_m(k_m, u^{2m}) = P_{X|Y}(1 \mid u_{k_m}) \tag{75}$$

$$a_m = p_{Y|X}(u_{k_m} \mid 1) P_X(1) \tag{76}$$

$$b_m = p_{Y|X}(u_{k_m} \mid 2) P_X(2). \tag{77}$$

We also drop the arguments of $\Lambda_m(k_m, u^{2m})$. Using (65) and applying Bayes' rule to q_m, we get

$$0 \leq \lim_{m\to\infty} |p_m - q_m| \tag{78}$$

$$= \lim_{m\to\infty} \left| \frac{a_m \Lambda_m}{a_m \Lambda_m + b_m} - \frac{a_m}{a_m + b_m} \right| \tag{79}$$

$$= \lim_{m\to\infty} \left| \frac{q_m \Lambda_m}{q_m \Lambda_m + 1 - q_m} - q_m \right| \tag{80}$$

$$= \lim_{m\to\infty} \frac{q_m(1-q_m)|\Lambda_m - 1|}{1 + q_m(\Lambda_m - 1)} \tag{81}$$

$$\leq \lim_{m\to\infty} \frac{1}{2}|\Lambda_m - 1| \tag{82}$$

where in (82) we have used the fact that as $\Lambda_m \to 1$, there exists some M_0 for which $\Lambda_m \geq 1/2$ for all $m > M_0$, and therefore for which the denominator $1 + q_m(\Lambda_m - 1) \geq 1/2$. Also, $q_m(1-q_m) \leq 1/4$. Taking the limit of Λ_m and noting that the argument holds, by Lemma 2, on a set of sample paths of probability one, we complete the proof. ■

B.6 Proof of Lemma 2

The main difficulty in proving Lemma 2 is that the success probabilities p_i making up the Poisson binomial distribution in question are not fixed, but are instead functions of the random variables $\bar{U}^{2m}$, which themselves are not independent. Consequently, there are two distinct layers of randomness to deal with. Our first step is to recognize that, from (70), p_i is a function of $\bar{U}_i$ only, and we can group $\bar{U}^{2m}$ into m blocks $\bar{U}_1^2, \ldots, \bar{U}_m^2$, each of which is indeed iid from $P_{\bar{U}^2}$. Here, $P_{\bar{U}^2}$ is the distribution of two samples where exactly one is from $P_{Y|X}(\cdot \mid 1)$ and the other is from $P_{Y|X}(\cdot \mid 2)$. Let $\mathcal{S}_j$ be the set of two indices corresponding to the locations of the samples forming block $\bar{U}_j^2$, where $j \in [m]$. With this structure established, we present some results downstream of the p_i's which will be useful later.

Lemma 3. *Define*

$$Y_j = \sum_{i\in\mathcal{S}_j} \mathbb{E}[X_i \mid \bar{U}_i] \text{ and } V_j = \sum_{i\in\mathcal{S}_j} \mathrm{Var}(X_i \mid \bar{U}_i) \tag{83}$$

and note that $\mathbb{E}[X_i \mid \bar{U}_i] = p_i$ *and* $\mathrm{Var}(X_i \mid \bar{U}_i) = p_i(1-p_i)$. *Then, the following hold almost surely:*

1. $\limsup_{m\to\infty} \dfrac{\left|\sum_{j=1}^m Y_j - m\right|}{\sqrt{2m \ln\ln m}} = \sigma_Y$, *where* $\sigma_Y^2 = \mathrm{Var}(Y_j)$.
2. $\lim_{m\to\infty} \dfrac{\sum_{j=1}^m V_j}{m} = \mu_V$, *where* $\mu_V = \mathbb{E}[V_j]$.

Proof: Note that Y_j and V_j are both functions of $\bar{U}_j^2$ only, and as mentioned previously, $\bar{U}_1^2, \ldots, \bar{U}_m^2$ are iid draws from $P_{\bar{U}^2}$. Moreover, the X_i's are Bernoulli random variables which can only take values in $\{0, 1\}$. Consequently, from the definitions in (83), Y^m and V^m are iid sequences of bounded random variables.

Expanding $\mathcal{S}_j = \{j_1, j_2\}$, we find the mean of Y_j to be

$$\mathbb{E}[Y_j] = \mathbb{E}\left[\frac{p_{Y|X}(\bar{U}_{j_1} \mid 1)}{p_{Y|X}(\bar{U}_{j_1} \mid 1) + p_{Y|X}(\bar{U}_{j_1} \mid 2)} + \frac{p_{Y|X}(\bar{U}_{j_2} \mid 1)}{p_{Y|X}(\bar{U}_{j_2} \mid 1) + p_{Y|X}(\bar{U}_{j_2} \mid 2)} \right]. \tag{84}$$

As dictated by the common randomness structure, exactly one of $\bar{U}_{j_1}$ and $\bar{U}_{j_2}$ is sampled from $P_{Y|X}(\cdot \mid 1)$ and the other is sampled from $P_{Y|X}(\cdot \mid 2)$. Assume without loss of generality that $\bar{U}_{j_1} \sim P_{Y|X}(\cdot \mid 1)$ and $\bar{U}_{j_2} \sim P_{Y|X}(\cdot \mid 2)$. We have

$$\mathbb{E}[Y_j] = \int \frac{p_{Y|X}(u \mid 1)^2}{p_{Y|X}(u \mid 1) + p_{Y|X}(u \mid 2)}\, du + \int \frac{p_{Y|X}(u \mid 1) p_{Y|X}(u \mid 2)}{p_{Y|X}(u \mid 1) + p_{Y|X}(u \mid 2)}\, du \tag{85}$$

$$= \int \frac{p_{Y|X}(u \mid 1)(p_{Y|X}(u \mid 1) + p_{Y|X}(u \mid 2))}{p_{Y|X}(u \mid 1) + p_{Y|X}(u \mid 2)}\, du \tag{86}$$

$$= \int p_{Y|X}(u \mid 1)\, du \tag{87}$$

$$= 1. \tag{88}$$

Thus $\mathbb{E}[\sum_{j=1}^{m} Y_j] = m$, and applying the law of the iterated logarithm and the strong law of large numbers yields the desired results. ■

Now we turn to analyzing the Poisson binomial random variable $S_k(\bar{U}^{2m})$ itself. Define

$$\mu_m^{(k)} = \sum_{i \in [2m] \setminus \{k\}} \mathbb{E}[X_i \mid \bar{U}_i] = \sum_{j=1}^{m} Y_j - p_k \tag{89}$$

$$B_m^{(k)} = \sum_{i \in [2m] \setminus \{k\}} \mathrm{Var}(X_i \mid \bar{U}_i) = \sum_{j=1}^{m} V_j - p_k(1 - p_k). \tag{90}$$

As a corollary to Lemma 3, we have the following.

Corollary 1. *Define*

$$R_m^* = \sup_{1 \le k \le 2m} |\mu_m^{(k)} - m| \text{ and } B_m^* = \inf_{1 \le k \le 2m} B_m^{(k)}. \tag{91}$$

Then, the following hold almost surely:

1. $\limsup_{m \to \infty} \dfrac{R_m^*}{\sqrt{2m \ln \ln m}} = \sigma_Y.$
2. $\lim_{m \to \infty} \dfrac{B_m^*}{m} = \mu_V.$

Proof: For the first part, using (89) and the triangle inequality, for all k,

$$\left|\sum_{j=1}^{m} Y_j - m\right| - p_k \le |\mu_m^{(k)} - m| \le \left|\sum_{j=1}^{m} Y_j - m\right| + p_k$$

$$\implies \left|\sum_{j=1}^{m} Y_j - m\right| - 1 \le R_m^* \le \left|\sum_{j=1}^{m} Y_j - m\right| + 1 \tag{92}$$

as $0 \le p_k \le 1$. Therefore,

$$\limsup_{m \to \infty} \frac{\left|\sum_{j=1}^{m} Y_j - m\right| - 1}{\sqrt{2m \ln \ln m}} \le \limsup_{m \to \infty} \frac{R_m^*}{\sqrt{2m \ln \ln m}} \le \limsup_{m \to \infty} \frac{\left|\sum_{j=1}^{m} Y_j - m\right| + 1}{\sqrt{2m \ln \ln m}} \tag{93}$$

after which the desired result follows from Lemma 3.

Similarly for the second part, using (90) and as $0 \le p_k(1 - p_k) \le 1/4$ for all k,

$$\sum_{j=1}^{m} V_j - \frac{1}{4} \le B_m^* \le \sum_{j=1}^{m} V_j. \tag{94}$$

From this,

$$\lim_{m \to \infty} \frac{\sum_{j=1}^{m} V_j - 1/4}{m} \le \lim_{m \to \infty} \frac{B_m^*}{m} \le \lim_{m \to \infty} \frac{\sum_{j=1}^{m} V_j}{m}. \tag{95}$$

As before, the result is then obtained by applying Lemma 3. ■

Remark. The condition $\Pr[0 < p_i < 1] > 0$ ensures that the probability of each Bernoulli trial being deterministic is strictly less than one. Concretely, this gives us $\mu_V > 0$, and Corollary 1 then implies that $B_m^* \to \infty$ almost surely.

In what follows, we focus on analyzing the convergence of the Poisson binomial probability ratio for a generic sample path of the outer randomness $\{u^{2m}\}_{m\geq 1}$ on which the properties claimed in Lemma 3 and Corollary 1 hold. As stated there, the set of such sample paths has probability one under the law of $\{\bar{U}^{2m}\}_{m\geq 1}$. Establishing the desired result for $\{u^{2m}\}_{m\geq 1}$ is then enough to show almost sure convergence. Continuing requires the following local limit theorem for sums of independent binary random variables.

Theorem 3 (Mamatov, 1965)**.** *Let W^n be a sequence of independent random variables taking on values 0 and 1. Define*

$$q_i = \Pr[W_i = 1],\ T_n = \sum_{i=1}^{n} W_i,\ a_n = \sum_{i=1}^{n} \mathbb{E}[W_i],\ b_n = \sum_{i=1}^{n} \mathrm{Var}(W_i), \tag{96}$$

$$L_n = b_n^{-3/2} \sum_{i=1}^{n} q_i(1-q_i)(q_i^2 + (1-q_i)^2). \tag{97}$$

Suppose $0 < q_i < 1$, then there exists an absolute constant C such that

$$\left| \sqrt{b_n} \Pr[T_n = j] - \frac{1}{\sqrt{2\pi}} e^{-(j-a_n)^2/2b_n} \right| < CL_n. \tag{98}$$

Given our realization $\bar{U}^{2m} = u^{2m}$, let $\mathcal{I}_m \subseteq [2m]$ be the set of indices for which $0 < p_i < 1$. The zero-variance trials at a given k and m, which are $\{X_i\}_{i\in[2m]\setminus(\mathcal{I}_m\cup\{k\})}$, are all deterministic and contribute a constant total of

$$G_m^{(k)} = \sum_{i\in[2m]\setminus(\mathcal{I}_m\cup\{k\})} p_i \tag{99}$$

to $S_k(u^{2m})$. To use Theorem 3, we therefore define the restricted sum

$$\tilde{S}_k(u^{2m}) = \sum_{i\in\mathcal{I}_m\setminus\{k\}} X_i = S_k(u^{2m}) - G_m^{(k)}. \tag{100}$$

The sums of the means and variances for the new sequence, with reference to (89) and (90), are respectively

$$\tilde{\mu}_m^{(k)} = \sum_{i\in\mathcal{I}_m\setminus\{k\}} \mathbb{E}[X_i \mid \bar{U}_i = u_i] = \mu_m^{(k)} - G_m^{(k)} \tag{101}$$

$$\tilde{B}_m^{(k)} = \sum_{i\in\mathcal{I}_m\setminus\{k\}} \mathrm{Var}(X_i \mid \bar{U}_i = u_i) = B_m^{(k)}. \tag{102}$$

The support of $\tilde{S}_k(u^{2m})$ is $\{0, 1, \ldots, r_m^{(k)}\}$ where $r_m^{(k)} = |\mathcal{I}_m \setminus \{k\}|$. As we wish to evaluate the Theorem 3 bound for $\tilde{S}_k(u^{2m})$ at the points $j_1 = m - G_m^{(k)}$ and $j_2 = m - 1 - G_m^{(k)}$, we first check that for large enough m, these points fall inside the support, clearing the way for a straightforward application of the theorem. First, since $B_m^{(k)} = \sum_{i\in\mathcal{I}_m\setminus\{k\}} p_i(1-p_i)$ we have

$$B_m^{(k)} \leq \sum_{i\in\mathcal{I}_m\setminus\{k\}} p_i = \tilde{\mu}_m^{(k)} \tag{103}$$

$$B_m^{(k)} \leq \sum_{i\in\mathcal{I}_m\setminus\{k\}} (1-p_i) = r_m^{(k)} - \tilde{\mu}_m^{(k)}. \tag{104}$$

Therefore $B_m^{(k)} \leq \min\{\tilde{\mu}_m^{(k)}, r_m^{(k)} - \tilde{\mu}_m^{(k)}\}$. In other words, the distance of the restricted sum's mean $\tilde{\mu}_m^{(k)}$ from the endpoints of its support is at least $B_m^{(k)}$, and this holds for any m. Concerning our specific evaluation points,

$$j_1 - \tilde{\mu}_m^{(k)} = m - \mu_m^{(k)} \text{ and } j_2 - \tilde{\mu}_m^{(k)} = m - 1 - \mu_m^{(k)} \tag{105}$$

we have that

$$\frac{1}{B_m^*} \max\{|j_1 - \tilde{\mu}_m^{(k)}|, |j_2 - \tilde{\mu}_m^{(k)}|\} \leq \frac{R_m^* + 1}{B_m^*}. \tag{106}$$

From Corollary 1, for any $\varepsilon_1 > 0$, there exists some M_1 such that for all $m > M_1$,

$$\frac{R_m^*}{\sqrt{2m \ln \ln m}} \leq \sigma_Y + \varepsilon_1 \tag{107}$$

and also for any $\varepsilon_2 > 0$ there exists some M_2 such that for all $m > M_2$,

$$\frac{m}{B_m^*} \leq \frac{1}{\mu_V} + \varepsilon_2. \tag{108}$$

Thus for any $\varepsilon_1, \varepsilon_2 > 0$ there exists some M_3 such that for all $m > M_3$,

$$\frac{R_m^*}{B_m^*} = \frac{R_m^*}{\sqrt{2m \ln \ln m}} \frac{\sqrt{2 \ln \ln m/m}}{B_m^*/m} \leq D(\varepsilon_1, \varepsilon_2) \sqrt{\frac{2 \ln \ln m}{m}} \tag{109}$$

where $D(\varepsilon_1, \varepsilon_2) = (\sigma_Y + \varepsilon_1)(\mu_V^{-1} + \varepsilon_2) > 0$. Thus, fixing ε_1 and ε_2,

$$0 \leq \lim_{m \to \infty} \frac{R_m^*}{B_m^*} \leq \lim_{m \to \infty} D(\varepsilon_1, \varepsilon_2) \sqrt{\frac{2 \ln \ln m}{m}} = 0. \tag{110}$$

Combining, taking the supremum, and as $B_m^* \to \infty$ by Corollary 1, we get

$$\lim_{m \to \infty} \frac{1}{B_m^*} \sup_{1 \leq k \leq 2m} \max\{|j_1 - \tilde{\mu}_m^{(k)}|, |j_2 - \tilde{\mu}_m^{(k)}|\} = 0. \tag{111}$$

Therefore there exists some M_4 such that for all $m > M_4$, the distance of both evaluation points j_1 and j_2 from the restricted sum's mean is less than B_m^*, and consequently within the support. In what follows, we assume $m > M_4$ throughout so that Theorem 3 can safely be used to analyze the restricted sequence of Bernoulli trials, which is $\{X_i\}_{i \in \mathcal{I}_m \setminus \{k\}}$.

Applying Theorem 3 to this restricted sequence gives us for each m, k and at our evaluation points $j \in \{j_1, j_2\}$,

$$\sqrt{B_m^{(k)}} \Pr[\tilde{S}_k(u^{2m}) = j] = A_m^{(k)}(j) + \varepsilon_m^{(k)}(j) \tag{112}$$

where $A_m^{(k)}(j) = (2\pi)^{-1/2} e^{-(j - \mu_m^{(k)} + G_m^{(k)})^2 / 2B_m^{(k)}}$ and $|\varepsilon_m^{(k)}(j)| \leq C/\sqrt{B_m^{(k)}}$, using the fact that, as $p_i^2 + (1-p_i)^2 \leq 1$,

$$(B_m^{(k)})^{-3/2} \sum_{i \in \mathcal{I}_m \setminus \{k\}} p_i(1-p_i)(p_i^2 + (1-p_i)^2) \leq (B_m^{(k)})^{-1/2}. \tag{113}$$

The probability ratio of interest can be written as

$$\frac{\Pr[S_k(u^{2m}) = m-1]}{\Pr[S_k(u^{2m}) = m]} = \frac{\Pr[\tilde{S}_k(u^{2m}) = m - 1 - G_m^{(k)}]}{\Pr[\tilde{S}_k(u^{2m}) = m - G_m^{(k)}]} \tag{114}$$

$$= \frac{A_m^{(k)}(m - 1 - G_m^{(k)}) + \varepsilon_m^{(k)}(m - 1 - G_m^{(k)})}{A_m^{(k)}(m - G_m^{(k)}) + \varepsilon_m^{(k)}(m - G_m^{(k)})} \tag{115}$$

$$= \frac{A_m^{(k)}(m - 1 - G_m^{(k)})}{A_m^{(k)}(m - G_m^{(k)})} \frac{1 + \Delta_m^{(k)}(m - 1 - G_m^{(k)})}{1 + \Delta_m^{(k)}(m - G_m^{(k)})}. \tag{116}$$

where we have defined $\Delta_m^{(k)}(j) = \varepsilon_m^{(k)}(j)/A_m^{(k)}(j)$.

To analyze the probability ratio, we begin with the error sequences. Using Theorem 3, for any m and $k \in [2m]$,

$$|\Delta_m^{(k)}(m - G_m^{(k)})| \leq \frac{C\sqrt{2\pi}}{(B_m^{(k)})^{1/2}} \exp\left(\frac{(\mu_m^{(k)} - m)^2}{2B_m^{(k)}}\right) \tag{117}$$

$$\leq \frac{C\sqrt{2\pi}}{(B_m^*)^{1/2}} \exp\left(\frac{(R_m^*)^2}{2B_m^*}\right) \tag{118}$$

$$:= d_m(m) \tag{119}$$

$$|\Delta_m^{(k)}(m-1-G_m^{(k)})| \le \frac{C\sqrt{2\pi}}{(B_m^{(k)})^{1/2}} \exp\left(\frac{(\mu_m^{(k)}-m+1)^2}{2B_m^{(k)}}\right) \tag{120}$$

$$= \frac{C\sqrt{2\pi}}{(B_m^{(k)})^{1/2}} \exp\left(\frac{(\mu_m^{(k)}-m)^2}{2B_m^{(k)}}\right) \exp\left(\frac{\mu_m^{(k)}-m}{B_m^{(k)}}\right) \exp\left(\frac{1}{2B_m^{(k)}}\right) \tag{121}$$

$$\le \frac{C\sqrt{2\pi}}{(B_m^{*})^{1/2}} \exp\left(\frac{(R_m^{*})^2}{2B_m^{*}}\right) \exp\left(\frac{R_m^{*}}{B_m^{*}}\right) \exp\left(\frac{1}{2B_m^{*}}\right) \tag{122}$$

$$\coloneqq d_m(m-1). \tag{123}$$

As a result,

$$\max\{|\Delta_m^{(k)}(m-G_m^{(k)})|, |\Delta_m^{(k)}(m-1-G_m^{(k)})|\} \le \max\{d_m(m), d_m(m-1)\} \coloneqq d_m. \tag{124}$$

We now proceed to analyze the sequence d_m by looking first at $d_m(m)$ and then at $d_m(m-1)$. First, for all $m \ge 3$,

$$d_m(m) = \frac{Cm^{-1/2}\sqrt{2\pi}}{(B_m^{*}/m)^{1/2}} \exp\left(\left(\frac{R_m^{*}}{\sqrt{2m\ln\ln m}}\right)^2 \frac{m\ln\ln m}{B_m^{*}}\right). \tag{125}$$

By (107) and (108), for any $\varepsilon_1, \varepsilon_2 > 0$, there exists some M_5 such that for any $m > M_5$,

$$\left(\frac{R_m^{*}}{\sqrt{2m\ln\ln m}}\right)^2 \frac{m}{B_m^{*}} \le E(\varepsilon_1, \varepsilon_2) \tag{126}$$

where $E(\varepsilon_1, \varepsilon_2) = (\sigma_Y + \varepsilon_1)^2(\mu_V^{-1} + \varepsilon_2) > 0$. With the required preliminaries out of the way, and fixing ε_1 and ε_2, we evaluate the limit as follows.

$$0 \le \lim_{m\to\infty} d_m(m) \tag{127}$$

$$\le \lim_{m\to\infty} \frac{Cm^{-1/2}\sqrt{2\pi}}{(B_m^{*}/m)^{1/2}} (\ln m)^{E(\varepsilon_1,\varepsilon_2)} \tag{128}$$

$$= \frac{C\sqrt{2\pi}}{\sqrt{\mu_V}} \lim_{m\to\infty} \frac{(\ln m)^{E(\varepsilon_1,\varepsilon_2)}}{\sqrt{m}} \tag{129}$$

where we have also applied Corollary 1 to evaluate $(B_m^{*}/m)^{1/2}$ in the limit. Since for any positive constant ρ, $\lim_{m\to\infty} m^{-1/2}(\ln m)^{\rho} = 0$, it follows that $\lim_{m\to\infty} d_m(m) = 0$.

Moving on to $d_m(m-1)$, we reuse (109) and (126), fix ε_1 and ε_2, and account for the final exponential term in (122) as $B_m^{*} \to \infty$ by Corollary 1. We get

$$0 \le \lim_{m\to\infty} d_m(m-1)$$

$$\le \lim_{m\to\infty} \frac{Cm^{-1/2}\sqrt{2\pi}}{(B_m^{*}/m)^{1/2}} (\ln m)^{E(\varepsilon_1,\varepsilon_2)} \exp\left(D(\varepsilon_1,\varepsilon_2)\sqrt{\frac{2\ln\ln m}{m}}\right) \tag{130}$$

$$= \frac{C\sqrt{2\pi}}{\sqrt{\mu_V}} \lim_{m\to\infty} \frac{(\ln m)^{E(\varepsilon_1,\varepsilon_2)}}{\sqrt{m}} \exp\left(D(\varepsilon_1,\varepsilon_2)\sqrt{\frac{2\ln\ln m}{m}}\right). \tag{131}$$

As the limit of the exponential term evaluates to one, we again get $\lim_{m\to\infty} d_m(m-1) = 0$. Consequently, $\lim_{m\to\infty} d_m = 0$.

Next, we examine the Gaussian ratio. Expanding gives

$$\ln \frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})} = \frac{m-\mu_m^{(k)}}{B_m^{(k)}} - \frac{1}{2B_m^{(k)}} \tag{132}$$

$$\implies -\frac{|\mu_m^{(k)}-m|}{B_m^{(k)}} - \frac{1}{2B_m^{(k)}} \le \ln \frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})} \le \frac{|\mu_m^{(k)}-m|}{B_m^{(k)}} + \frac{1}{2B_m^{(k)}} \tag{133}$$

$$\implies -\frac{R_m^{*}}{B_m^{*}} - \frac{1}{2B_m^{*}} \le \ln \frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})} \le \frac{R_m^{*}}{B_m^{*}} + \frac{1}{2B_m^{*}}. \tag{134}$$

Let $c_m = (R_m^*/B_m^*) + 1/(2B_m^*)$, we then have that for any m and $k \in [2m]$,

$$e^{-c_m} \leq \frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})} \leq e^{c_m}. \tag{135}$$

Turning our attention to the sequence c_m, using the result in (110) and the fact that $B_m^* \to \infty$ by Corollary 1, we obtain $\lim_{m\to\infty} c_m = 0$.

Using these results on d_m and c_m, we can proceed to examine (116). As $d_m \to 0$, there exists some M_6 such that for all $m > M_6$, $d_m < 1$ and hence for any $k \in [2m]$,

$$\frac{1-d_m}{1+d_m} \leq \frac{1+\Delta_m^{(k)}(m-1-G_m^{(k)})}{1+\Delta_m^{(k)}(m-G_m^{(k)})} \leq \frac{1+d_m}{1-d_m}. \tag{136}$$

Then, using (135),

$$e^{-c_m}\frac{1-d_m}{1+d_m} \leq \frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})}\frac{1+\Delta_m^{(k)}(m-1-G_m^{(k)})}{1+\Delta_m^{(k)}(m-G_m^{(k)})} \leq e^{c_m}\frac{1+d_m}{1-d_m}. \tag{137}$$

As the bounding sequences do not depend on k, we can move to the supremum and get

$$0 \leq \sup_{1\leq k\leq 2m}\left|\frac{A_m^{(k)}(m-1-G_m^{(k)})}{A_m^{(k)}(m-G_m^{(k)})}\frac{1+\Delta_m^{(k)}(m-1-G_m^{(k)})}{1+\Delta_m^{(k)}(m-G_m^{(k)})} - 1\right| \tag{138}$$

$$\leq \max\left\{e^{c_m}\frac{1+d_m}{1-d_m} - 1, 1 - e^{-c_m}\frac{1-d_m}{1+d_m}\right\}. \tag{139}$$

As we have already established that $\lim_{m\to\infty} c_m = \lim_{m\to\infty} d_m = 0$, we obtain the desired result for the sample path $\{u^{2m}\}_{m\geq 1}$, from which almost sure convergence follows. ■

C Theoretical Comparison and Lower Bound

Recall that from Theorem 1, the one-shot permuted scheme achieves

$$\mathbb{E}[\ell(M)] \leq I(X;Y \mid \bar{U}^N) + \log(I(X;Y \mid \bar{U}^N) + 2) + 3. \tag{140}$$

We can also establish the equality

$$I(X;Y) + I(X;\bar{U}^N \mid Y) = H(X) - H(X \mid Y) + H(X \mid Y) - H(X \mid Y, \bar{U}^N) \tag{141}$$

$$\overset{(a)}{=} H(X \mid \bar{U}^N) - H(X \mid Y, \bar{U}^N) \tag{142}$$

$$= I(X;Y \mid \bar{U}^N) \tag{143}$$

where in (a) we have used the fact that $\bar{U}^N$ is generated independently of X,. Therefore, the leading term in the achievability bound in (140) contains a penalty, namely $I(X;\bar{U}^N \mid Y)$, above the mutual information $I(X;Y)$.

This may be surprising when compared against other well-known channel simulation schemes including PFR (Li & Gamal, 2018), which achieves

$$\mathbb{E}[\ell(M)] \leq I(X;Y) + \log(I(X;Y) + 2) + 3 \tag{144}$$

or GRS (Harsha et al., 2010; Flamich & Theis, 2023), which satisfies

$$\mathbb{E}[\ell(M)] \leq I(X;Y) + \log(I(X;Y) + \log(4e)) + \log(8e). \tag{145}$$

Neither of these expressions contain such a penalty; extending the PFR and GRS bounds to n independent uses of the channel with an iid source X^n leaves $I(X;Y)$ as the only term which does not vanish as n becomes large, when considering the number of bits per sample $\mathbb{E}[\ell(M)]/n$.

We now present a lower bound on the communication cost, which shows that the penalty is inherent to the structure of the finite sample pool $\bar{U}^N$ used as part of the shared randomness in the permuted scheme, rather than being an artifact of the implementation or a loose upper bound in Theorem 1.

Proposition 4. *The communication cost of the permuted scheme is bound below by*

$$\mathbb{E}[\ell(M)] \geq I(X;Y \mid \bar{U}^N). \tag{146}$$

Proof: The cost of transmitting any message M with shared randomness W is bound below by

$$H(M \mid W) \geq H(M \mid W) - H(M \mid X, W) = I(X; M \mid W). \tag{147}$$

Since W is generated independent of the input X,

$$I(X; M \mid W) = I(X; M \mid W) + I(X; W) = I(X; M, W). \tag{148}$$

Since the decoder outputs Y as a function of M and W, adding Y does not change the mutual information. We get

$$\begin{aligned} I(X; M, W) &= I(X; M, W, Y) && (149)\\ &= I(X; Y) + I(X; W \mid Y) + I(X; M \mid Y, W) && (150)\\ &\geq I(X; Y) + I(X; W \mid Y). && (151) \end{aligned}$$

Finally,

$$\mathbb{E}[\ell(M)] \geq H(M \mid W) \geq I(X; Y) + I(X; W \mid Y). \tag{152}$$

For the permuted scheme in particular, the shared randomness comprises $\bar{U}^N$, S^N and Π. Substituting for W gives

$$\mathbb{E}[\ell(M)] \geq I(X; Y) + I(X; \bar{U}^N, S^N, \Pi \mid Y) \geq I(X; Y) + I(X; \bar{U}^N \mid Y). \tag{153}$$

which yields the desired lower bound after using (143). ■

D Non-Uniform PolarSim with Side Information

In this section, we present a modified version of PolarSim (Sriramu et al., 2024) that allows for non-uniformly distributed inputs as well as the use of side information available at both the encoder and decoder. While the generalized PolarSim algorithm presented in Ozyilkan et al. (2026) covers this setting and further allows for nonstationary channels, it relies on interleaving additional random permutations inside the polar transform, and its theoretical analysis is complicated by the fact that the number of recursion levels may not equal $\log n$ for n channel uses. In contrast, our method makes use of the standard, unmodified polar transform (Arıkan, 2009; 2010), considerably simplifying the theory and implementation. We emphasize that our modified PolarSim is still limited to binary-output channels, and is paired with MLC in Section 4 of the main paper to handle larger output alphabets.

The complete algorithm, comprising an encoder and a decoder sharing common randomness via the PRNG $\mathfrak{G}$, is presented in Algorithm 2, where G_n is the polar transform matrix as defined in Arıkan (2010). Note that $G_n^{-1} = G_n$, and the polar transform can in practice be implemented recursively in $O(n \log n)$ time. The goal of Algorithm 2 is to simulate n iid copies of a binary-output channel $P_{Y|X,W}$ when provided input sequences x^n and w^n. Here, x^n is the channel input and w^n is the side information, with (x^n, w^n) comprising n iid draws from a joint distribution $P_{X,W}$. As w^n is known to both the encoder and the decoder, the conditional mutual information $I(X; Y \mid W)$ is a lower bound on the communication cost per channel use.

We now present two results on our variant of the PolarSim algorithm, first dealing with its correctness in terms of the output distribution and second with the rate performance in the limit of large block-lengths. Note that in practice, we implement `enc`, `dec` using arithmetic coding (Rissanen, 1976) with empirically estimated probabilities, therefore leading to some rate overhead above the theoretical bounds, which imply knowledge of the exact probabilities.

Theorem 4. *Algorithm 2 simulates the channel $P_{Y|X,W}$ exactly, in the sense that given inputs x^n and w^n, the output $y^n \in \{0,1\}^n$ is a draw from $P_{Y^n|X^n,W^n}(\cdot \mid x^n, w^n) = \prod_{i=1}^n P_{Y|X,W}(\cdot \mid x_i, w_i)$. Moreover, it is rate-optimal in the sense that there exists a choice of* `enc`, `dec` *such that*

$$\lim_{n\to\infty} \frac{1}{n} \mathbb{E}[\ell(M)] = I(X; Y \mid W). \tag{154}$$

Proof: Beginning with the claim of correctness and following the argument in Ozyilkan et al. (2026), the polar transform $y^n = z^n G_n$ is a bijection, and by construction z^n is a draw from $P_{Z^n|X^n,W^n}(\cdot \mid x^n, w^n)$. As the decoder reconstructs z^n exactly, it follows that y^n is a draw from $P_{Y^n|X^n,W^n}(\cdot \mid x^n, w^n)$, which is the desired output distribution.

Moving on to consider the rate, we again follow a slight modification of the proof given in Ozyilkan et al. (2026); the full argument is presented here for completeness using the same notation as that in Algorithm 2. We have

$$\begin{aligned}
&\Pr[\Delta_i = 1] \\
&= \mathbb{E}_{Z^{i-1},X^n,W^n}[\Pr[\Delta_i = 1 \mid Z^{i-1}, X^n, W^n]] && (155)\\
&= \mathbb{E}_{Z^{i-1},X^n,W^n}[|\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] - \Pr[Z_i = 1 \mid Z^{i-1}, W^n]|] && (156)\\
&\leq (\mathbb{E}_{Z^{i-1},X^n,W^n}[(\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] - \Pr[Z_i = 1 \mid Z^{i-1}, W^n])^2])^{1/2} && (157)\\
&= (\mathbb{E}_{Z^{i-1},W^n}[\mathbb{E}[(\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] \\
&\quad - \mathbb{E}[\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] \mid Z^{i-1}, W^n])^2 \mid Z^{i-1}, W^n]])^{1/2} && (158)\\
&= (\mathbb{E}_{Z^{i-1},W^n}[\mathrm{Var}(\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] \mid Z^{i-1}, W^n)])^{1/2} && (159)\\
&= (\mathrm{Var}(\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n]) \\
&\quad - \mathrm{Var}(\mathbb{E}[\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n] \mid Z^{i-1}, W^n]))^{1/2} && (160)\\
&= (\mathrm{Var}(\Pr[Z_i = 1 \mid Z^{i-1}, X^n, W^n]) - \mathrm{Var}(\Pr[Z_i = 1 \mid Z^{i-1}, W^n]))^{1/2} && (161)
\end{aligned}$$

where we have followed the steps in Ozyilkan et al. (2026, Appendix C), while removing the permutation randomness present in their construction and inserting the shared side information sequence W^n. Then, following Lemma C.1 in the same document, we get

$$\Pr[\Delta_i = 1] \leq \frac{1}{2}(H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2)^{1/2} \leq \frac{1}{2}. \tag{162}$$

If `enc`, `dec` are chosen to be an arithmetic encoder-decoder pair furnished with the appropriate probabilities, the expected message length satisfies (Rissanen, 1976)

$$\begin{aligned}
\frac{\mathbb{E}[\ell(M)]}{n} &\leq \frac{1}{n}\sum_{i=1}^{n} h_B(\Pr[\Delta_i = 1]) + \frac{2}{n} && (163)\\
&\leq \frac{1}{n}\sum_{i=1}^{n} h_B\left(\frac{1}{2}(H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2)^{1/2}\right) + \frac{2}{n}. && (164)
\end{aligned}$$

For any $\delta \in (0, 1/2)$, create three partitions,

$$A_L = \{i \in [n] : H(Z_i \mid Z^{i-1}, W^n) \in [0, \delta)\} \tag{165}$$

Algorithm 2 Modified PolarSim (Sriramu et al., 2024; Ozyilkan et al., 2026).

function ENCODE($x^n, w^n, \mathfrak{G}$, `enc`)
- **Input:** Channel input x^n, side information w^n, PRNG $\mathfrak{G}$, lossless encoder `enc`.
- **Output:** Message M.
- Generate $S^n \sim \mathrm{Unif}[0,1]$ using $\mathfrak{G}$.
- **for** $i = 1, \ldots, n$ **do**
  - $p_i \leftarrow \Pr[Z_i = 0 \mid Z^{i-1} = z^{i-1}, X^n = x^n, W^n = w^n]$
  - $q_i \leftarrow \Pr[Z_i = 0 \mid Z^{i-1} = z^{i-1}, W^n = w^n]$
  - $z_i \leftarrow \mathbb{1}\{S_i > p_i\}$
  - $\tilde{z}_i \leftarrow \mathbb{1}\{S_i > q_i\}$
  - $\Delta_i \leftarrow z_i \oplus \tilde{z}_i$
- $M \leftarrow \texttt{enc}(\Delta^n)$
- **return** M

function DECODE($M, w^n, \mathfrak{G}$, `dec`)
- **Input:** Message M, side information w^n, PRNG $\mathfrak{G}$, lossless decoder `dec`.
- **Output:** Channel output $y^n \in \{0,1\}^n$.
- Generate $S^n \sim \mathrm{Unif}[0,1]$ using $\mathfrak{G}$.
- $\Delta^n \leftarrow \texttt{dec}(M)$
- **for** $i = 1, \ldots, n$ **do**
  - $q_i \leftarrow \Pr[Z_i = 0 \mid Z^{i-1} = z^{i-1}, W^n = w^n]$
  - $\tilde{z}_i \leftarrow \mathbb{1}\{S_i > q_i\}$
  - $z_i \leftarrow \Delta_i \oplus \tilde{z}_i$
- $y^n \leftarrow z^n G_n$
- **return** y^n

$$A_M = \{i \in [n] : H(Z_i \mid Z^{i-1}, W^n) \in [\delta, 1-\delta]\} \tag{166}$$
$$A_H = \{i \in [n] : H(Z_i \mid Z^{i-1}, W^n) \in (1-\delta, 1]\} \tag{167}$$

which form the low, medium and high-entropy sets respectively. Then,

$$\frac{\mathbb{E}[\ell(M)]}{n} \leq \frac{1}{n} \sum_{i \in A_H} h_B\left(\frac{1}{2}(H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2)^{1/2}\right) + h_B\left(\frac{\sqrt{\delta}}{2}\right) + \frac{|A_M|}{n} + \frac{2}{n}. \tag{168}$$

To continue with the proof, which will involve bounding the sizes of A_L, A_M and A_H, we recall the standard source polarization theorem.

Lemma 4 (Arıkan, 2010, Theorem 1)**.** *Let $(X, Y) \sim P_{X,Y}$ be an arbitrary pair of random variables over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} = \{0, 1\}$. Consider $\{(X_i, Y_i)\}_{i \geq 1}$ a sequence of independent draws from (X, Y) and write (X^n, Y^n) to denote the first n elements of this sequence. For any $n = 2^m$, $m \geq 1$, let $Z^n = X^n G_n$. Then, for any $\delta \in (0, 1)$,*

$$\lim_{n\to\infty} \frac{1}{n} |\{i \in [n] : H(Z_i \mid Z^{i-1}, Y^n) \in (1-\delta, 1]\}| = H(X \mid Y) \tag{169}$$
$$\lim_{n\to\infty} \frac{1}{n} |\{i \in [n] : H(Z_i \mid Z^{i-1}, Y^n) \in [0, \delta)\}| = 1 - H(X \mid Y). \tag{170}$$

Remark. Note that while the result in Arıkan (2010) is stated for countable $\mathcal{Y}$ for simplicity, the same arguments, in particular regarding the convergence of the Bhattacharyya supermartingale and its connection to the entropy martingale, also work with the side information spaces considered here if the source Bhattacharyya parameters are defined appropriately as

$$Z(X \mid Y) = 2\,\mathbb{E}_Y\left[\sqrt{P_{X|Y}(0 \mid Y) P_{X|Y}(1 \mid Y)}\right]. \tag{171}$$

More specifically, our MLC construction in Section 4.2 uses the side information $(\bar{U}^N, B^{l-1})$ at level l, where $\bar{U}^N$ is continuous and B^{l-1} is discrete. Thus we state Lemma 4 in this setting. See also Theorem B.2 of Ozyilkan et al. (2026) for a related analysis. In what follows we assume that n is a valid polar blocklength throughout, i.e. $n = 2^m$ for some integer $m \geq 1$ as required by the source polarization theorem.

Since $(X^n, W^n) \sim P_{X,W}$ iid and the channel $P_{Y|X,W}$ is memoryless, we note

$$P_{Y^n|W^n}(y^n \mid w^n) = \mathbb{E}_{X^n|W^n=w^n}[P_{Y^n|X^n,W^n}(y^n \mid X^n, w^n)] \tag{172}$$
$$= \mathbb{E}_{X^n|W^n=w^n}\left[\prod_{i=1}^n P_{Y|X,W}(y_i \mid X_i, w_i)\right] \tag{173}$$
$$= \mathbb{E}_{X_1|W_1=w_1} \cdots \mathbb{E}_{X_n|W_n=w_n}\left[\prod_{i=1}^n P_{Y|X,W}(y_i \mid X_i, w_i)\right] \tag{174}$$
$$= \prod_{i=1}^n \mathbb{E}_{X_i|W_i=w_i}[P_{Y|X,W}(y_i \mid X_i, w_i)] \tag{175}$$
$$= \prod_{i=1}^n P_{Y|W}(y_i \mid w_i). \tag{176}$$

Therefore, $(Y^n, W^n) \sim P_{Y|W} P_W$ iid. Applying Lemma 4 with $X^n \leftarrow Y^n$ and $Y^n \leftarrow W^n$, we then have that $\lim_{n\to\infty} |A_M|/n = 0$. From this, there exists some N_A such that for all $n > N_A$, $|A_M|/n < \delta$. Further using the bound $h_B(\sqrt{\delta}/2) \leq 2\delta^{1/4}$ (Ozyilkan et al., 2026), we have for any $n > N_A$,

$$\frac{\mathbb{E}[\ell(M)]}{n} \leq \frac{1}{n} \sum_{i \in A_H} h_B\left(\frac{1}{2}(H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2)^{1/2}\right) + 2\delta^{1/4} + \delta + \frac{2}{n}. \tag{177}$$

Under the same condition on n, and again following Ozyilkan et al. (2026) with the necessary modifications,

$$(1-\delta)|A_H| \leq \sum_{i=1}^{n} H(Z_i \mid Z^{i-1}, W^n) \leq |A_H| + (1-\delta)|A_M| + \delta|A_L| \tag{178}$$

$$\implies (1-\delta)|A_H| \leq nH(Y \mid W) \leq |A_H| + (1-\delta)|A_M| + \delta|A_L| \tag{179}$$

$$\implies (1-\delta)\frac{|A_H|}{n} \leq H(Y \mid W) \leq \frac{|A_H|}{n} + 2\delta \tag{180}$$

$$\implies -\delta \leq H(Y \mid W) - \frac{|A_H|}{n} \leq 2\delta \tag{181}$$

$$\implies \frac{|A_H|}{n} \leq H(Y \mid W) + \delta. \tag{182}$$

Here we have used the fact that, as the polar transform is a bijection, $H(Z^n \mid W^n) = H(Y^n \mid W^n)$. Also, as we previously established that (Y^n, W^n) is iid,

$$\sum_{i=1}^{n} H(Z_i \mid Z^{i-1}, W^n) = H(Y^n \mid W^n) = nH(Y \mid W). \tag{183}$$

Next defining

$$\tilde{R} = \frac{1}{n} \sum_{i \in A_H} h_B\left(\frac{1}{2}(H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2)^{1/2}\right) \tag{184}$$

and the partitions

$$B_L = \{i \in [n] : H(Z_i \mid Z^{i-1}, X^n, W^n) \in [0, \delta)\} \tag{185}$$

$$B_M = \{i \in [n] : H(Z_i \mid Z^{i-1}, X^n, W^n) \in [\delta, 1-\delta]\} \tag{186}$$

$$B_H = \{i \in [n] : H(Z_i \mid Z^{i-1}, X^n, W^n) \in (1-\delta, 1]\} \tag{187}$$

we see that

$$\tilde{R} \leq \frac{|A_H \cap B_L|}{n} + \frac{|A_H \cap B_M|}{n} + h_B\left(\sqrt{\frac{\delta}{2}}\right) \tag{188}$$

$$\leq \frac{|A_H \cap B_L|}{n} + \frac{|B_M|}{n} + h_B\left(\sqrt{\frac{\delta}{2}}\right) \tag{189}$$

where we have used the fact that for any $i \in A_H \cap B_H$,

$$H(Z_i \mid Z^{i-1}, W^n) - (H(Z_i \mid Z^{i-1}, X^n, W^n))^2 \leq 1 - (1-\delta)^2 \leq 2\delta. \tag{190}$$

As $(Y^n, X^n, W^n) \sim P_{Y|X,W} P_{X,W}$ iid, we can apply Lemma 4 again, this time with $X^n \leftarrow Y^n$ and $Y^n \leftarrow (X^n, W^n)$. We then have $\lim_{n\to\infty} |B_M|/n = 0$, and so there exists some N_B such that for all $n > N_B$, $|B_M|/n < \delta$. For such n, following a similar approach to (178)–(182),

$$(1-\delta)|B_H| \leq \sum_{i=1}^{n} H(Z_i \mid Z^{i-1}, X^n, W^n) \leq |B_H| + (1-\delta)|B_M| + \delta|B_L| \tag{191}$$

$$\implies (1-\delta)|B_H| \leq nH(Y \mid X, W) \leq |B_H| + (1-\delta)|B_M| + \delta|B_L| \tag{192}$$

$$\implies (1-\delta)\frac{|B_H|}{n} \leq H(Y \mid X, W) \leq \frac{|B_H|}{n} + 2\delta \tag{193}$$

$$\implies -\delta \leq H(Y \mid X, W) - \frac{|B_H|}{n} \leq 2\delta \tag{194}$$

$$\implies \frac{|B_H|}{n} \geq H(Y \mid X, W) - 2\delta. \tag{195}$$

Again, similar to (183), we have made use of the simplification

$$\sum_{i=1}^{n} H(Z_i \mid Z^{i-1}, X^n, W^n) = H(Y^n \mid X^n, W^n) = nH(Y \mid X, W). \tag{196}$$

Thus, for any $n > N_0 = \max\{N_A, N_B\}$, and noting that $B_H \subseteq A_H$ we have

$$\frac{|A_H \cap B_L|}{n} \leq \frac{|A_H|}{n} - \frac{|A_H \cap B_H|}{n} \tag{197}$$

$$= \frac{|A_H|}{n} - \frac{|B_H|}{n} \tag{198}$$

$$\leq H(Y \mid W) - H(Y \mid X, W) + 3\delta \tag{199}$$

$$= I(X; Y \mid W) + 3\delta. \tag{200}$$

Substituting into (189), we obtain for all such n,

$$\tilde{R} \leq I(X; Y \mid W) + 4\delta + 2\delta^{1/4} \tag{201}$$

where we have applied the inequality $h_B(\sqrt{\delta/2}) \leq 2\delta^{1/4}$ (Ozyilkan et al., 2026). Finally, let $N_1 = \max\{N_0, 2/\delta\}$. For any $n > N_1$, after substituting into (177), we get

$$\frac{\mathbb{E}[\ell(M)]}{n} \leq I(X; Y \mid W) + 6\delta + 4\delta^{1/4}. \tag{202}$$

Since δ can be made arbitrarily small, after combining with the fact that $I(X; Y \mid W)$ is a lower bound on the communication cost per sample, we have shown that

$$\lim_{n\to\infty} \frac{1}{n} \mathbb{E}[\ell(M)] = I(X; Y \mid W) \tag{203}$$

which completes the proof. ■

E Additional Experimental Results and Details

E.1 Approximating the Posterior Distribution

As briefly mentioned in Section 3.2, computing the exact posterior distribution $P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$ rapidly becomes impractical at larger alphabet sizes as it involves calculating matrix permanents. When the alphabet size grows beyond $N = 16$ in our image compression application, we therefore adopt an iterative approximation of the posterior via a Sinkhorn-Knopp-based algorithm introduced to machine learning in Mena et al. (2018; 2020). In this section, we first briefly describe how the approximation is implemented in the context of the permuted scheme as introduced in Section 3, and then provide empirical results on the suitability of its output distribution for applied channel simulation tasks.

Implementation Details. Mena et al. (2020) describe an algorithm for approximating posterior distributions over a permutation matrix P, i.e. a random $N \times N$ binary matrix with a unique 1 in each row and column representing a permutation of $[N]$, where the distribution of P is

$$P_{P|A}(P \mid A) = \frac{1}{Z_A} \exp(\langle \ln A, P \rangle_F). \tag{204}$$

Here, $\langle \ln A, P \rangle_F$ is the Frobenius matrix inner product, A is a parameter matrix and Z_A is a normalizing constant. In Mena et al. (2020) it is shown and theoretically justified that the matrix of expectations $\mathbb{E}[P]$ can be well-approximated by applying the Sinkhorn operator (Sinkhorn, 1964) to A. The Sinkhorn operator is defined to be the infinite successive row-then-column normalization of A and produces a doubly stochastic matrix, which can be viewed as the solution to an instance of optimal transport with entropic regularization (Altschuler et al., 2017). The basic iterative implementation of this successive normalization idea is sometimes referred to as the Sinkhorn-Knopp algorithm (Knight, 2008). In practice, a finite number of row and column normalization steps are performed, and the algorithm is carried out in log-space to increase numerical stability.

We briefly outline how this framework allows us to compute an approximation for $P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$. The likelihood of observing $\bar{U}^N = u^N$ in the course of the permuted scheme given the underlying permutation is $\Pi = \pi$ is

$$p_{\bar{U}^N|\Pi}(u^N \mid \pi) = \prod_{i=1}^{N} p_{Y|X}(u_i \mid \pi^{-1}(i)). \tag{205}$$

```python
def sinkhorn(L, num_iters=20):
    for _ in range(num_iters):
        # Row normalization
        L = L - torch.logsumexp(L, dim=1, keepdim=True)
        # Column normalization
        L = L - torch.logsumexp(L, dim=0, keepdim=True)
    return L
```

Listing 1: Sinkhorn-Knopp algorithm in PyTorch.

Applying Bayes' rule, the posterior induced on the uniformly distributed random permutation Π is therefore

$$\Pr[\Pi = \pi \mid \bar{U}^N = u^N] = \frac{p_{\bar{U}^N|\Pi}(u^N \mid \pi)}{N! p_{\bar{U}^N}(u^N)} = \frac{p_{\bar{U}^N|\Pi}(u^N \mid \pi)}{\operatorname{perm}(L(u^N))}. \tag{206}$$

We can map Π to a permutation matrix $P(\Pi) \in \{0,1\}^{N\times N}$; given a realization π, we have that $\pi(j) = i$ if and only if $P_{ij} = 1$ or, in the context of the permuted scheme, u_i was drawn from $P_{Y|X}(\cdot \mid j)$. Then, the posterior can be written as a Frobenius inner product involving the log-likelihood matrix $\ln L(u^N)$, where $L(u^N)$ is as defined in Section 3.2,

$$\Pr[\Pi = \pi \mid \bar{U}^N = u^N] = \frac{\exp(\langle \ln L(u^N), P(\pi)\rangle_F)}{\operatorname{perm}(L(u^N))}. \tag{207}$$

This is in the form of (204), where $L(u^N)$ is the parameter matrix. Taking the expectation of each element yields

$$\begin{aligned}
\mathbb{E}[P_{ij}] &= \sum_{\pi\in S_N} \Pr[\Pi = \pi \mid \bar{U}^N = u^N]\, \mathbb{1}\{\pi(j) = i\} && (208)\\
&= \mathbb{E}[\mathbb{1}\{\Pi(j) = i\} \mid \bar{U}^N = u^N] && (209)\\
&= \Pr[\Pi(j) = i \mid \bar{U}^N = u^N]. && (210)
\end{aligned}$$

Note that (210) is exactly the posterior distribution required to run the permuted scheme as described in Section 3.1, that is $\Pr[\Pi(j) = i \mid \bar{U}^N = u^N] = P_{K|X,\bar{U}^N}(i \mid j, u^N)$. Thus we can obtain the required posterior probabilities given $X = x$ and $\bar{U}^N = u^N$, namely $\Pr[\Pi(x) = k \mid \bar{U}^N = u^N]$ for all $1 \le k \le N$, by taking the expectation and extracting column x of $\mathbb{E}[P(\Pi)]$. Consequently, to get an approximation, we can follow the approach in Mena et al. (2020) by computing $\ln L(u^N)$, performing Sinkhorn-Knopp iterations on this log-likelihood matrix, and taking column x from the result. We show a simple PyTorch implementation in Listing 1. In practice, for our experiments in Section 5.1, we use a custom Triton implementation which avoids instantiating the entire likelihood matrix in GPU memory to increase throughput.

Performance of the Approximation. To measure the suitability of the approximate Sinkhorn-Knopp posterior, we run a small additional experiment on basic discrete-input N-ary PAM channels with additive Gaussian noise. In particular, we first test the marginal output distribution produced by the channel simulation for $N \in \{32, 64, 128, 256, 512\}$ by generating 10^5 samples uniformly from the N-ary input X, then simulating $P_{Y|X}$ with the permuted scheme for each input sample, incorporating the Sinkhorn-Knopp algorithm to approximate $P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$ as required. The inputs come from a constellation $\mathcal{X} = \{2m - (N-1) : m \in [N]\}$ evenly spaced around the origin, and we use $P_{Y|X}(\cdot \mid x) = \mathcal{N}(x, \sigma^2)$ with $\sigma^2 = 10^{-\mathrm{SNR}/10}$ and low, moderate and high signal-to-noise ratios (SNRs) in $\{-10, 1, 10\}$.

We use the Kolmogorov-Smirnov (KS) statistic (Conover, 1999) to measure the closeness to the target marginal with

$$p_Y(y) = \frac{1}{N}\sum_{x\in\mathcal{X}} \mathcal{N}(y; x, \sigma^2) \tag{211}$$

and present the results in Table 1. For scale, we compare against the same test run on samples generated directly from the true marginal in (211). The mean of the KS statistic is displayed over

	SNR	−10		1		10	
N	Iters.	Approx.	True	Approx.	True	Approx.	True
32	20	3.08 ± 0.64	3.00 ± 0.81	2.96 ± 0.67	3.03 ± 0.84	3.01 ± 0.62	2.98 ± 0.79
	50	2.67 ± 0.52		2.43 ± 0.52		2.43 ± 0.54	
	100	2.43 ± 0.27		2.36 ± 0.59		2.35 ± 0.60	
64	20	2.55 ± 0.57	2.84 ± 0.78	2.57 ± 0.57	2.93 ± 0.87	2.54 ± 0.54	2.94 ± 0.83
	50	2.76 ± 0.64		2.67 ± 0.63		2.70 ± 0.66	
	100	3.41 ± 1.03		3.31 ± 0.92		3.25 ± 0.96	
128	20	2.79 ± 0.68	3.15 ± 1.02	2.80 ± 0.69	3.20 ± 1.06	2.82 ± 0.70	3.19 ± 1.05
	50	3.10 ± 0.88		3.18 ± 0.85		3.16 ± 0.85	
	100	2.46 ± 0.77		2.43 ± 0.65		2.44 ± 0.68	
256	20	2.87 ± 0.65	2.51 ± 0.67	2.92 ± 0.65	2.51 ± 0.70	2.92 ± 0.65	2.51 ± 0.69
	50	2.93 ± 0.74		2.95 ± 0.73		2.94 ± 0.74	
	100	2.68 ± 0.44		2.66 ± 0.37		2.68 ± 0.38	
512	20	2.81 ± 0.42	3.03 ± 1.30	2.83 ± 0.45	3.01 ± 1.26	2.82 ± 0.41	3.00 ± 1.26
	50	2.52 ± 0.84		2.48 ± 0.81		2.48 ± 0.79	
	100	2.39 ± 0.56		2.44 ± 0.51		2.44 ± 0.54	

Table 1: KS statistics for the marginal distributions produced for N-ary Gaussian channels using the Sinkhorn-Knopp approximation (Approx.) with varying iteration counts and ground-truth samples (True). All values have been scaled up by 10^3 and one standard deviation is shown.

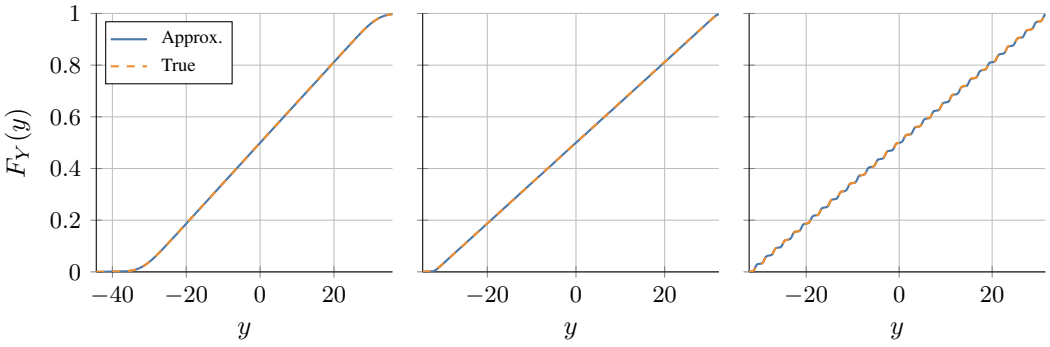


Figure 7: Empirical marginal CDFs for the output of a 1D Gaussian channel with 32-ary input constructed using the Sinkhorn-Knopp approximation (Approx.) and the ground-truth CDF (True) at SNRs, from left to right -10, 1 and 10.

10 independent runs with 10^5 samples each, along with its standard deviation. For all iteration counts, the mean KS statistic for our approximate sampler falls within the uncertainty region. We also plot the empirical marginal CDF for $N = 32$ derived from our approximate sampler using the Sinkhorn-Knopp posterior alongside the analytical Gaussian mixture CDF in Fig. 7.

As the noise is additive, we may also test the fit of the approximation to $P_{Y|X}$ via the residual $Z = Y - X$, which we desire to be a zero-mean Gaussian. We generate 10^5 samples each from 10 runs using 20 Sinkhorn-Knopp iterations and compare against the CDF of $\mathcal{N}(0, \sigma^2)$. To account for a simple calibration of the mechanism, we also estimate the achieved, as opposed to nominal, standard deviation across all runs as half the spread between the 16th and 84th quantiles, denoting it by $\hat{\sigma}$, and compare to $\mathcal{N}(0, \hat{\sigma}^2)$. This is shown in Fig. 8 for $N = 256$. Mean KS statistics against the adjusted target range from 2.41×10^{-3} to 4.99×10^{-3} across $N \in \{32, 64, 128, 256, 512\}$. The closer agreement observable in Fig. 8 after calibration indicates that much of the discrepancy from the nominal channel can be attributed to a slightly increased noise scale, while the shape remains visibly close to Gaussian.

As final test, we also consider the quality of the conditional distributions, namely $P_{Y|X}(\cdot \mid x)$ for each x, in the specific context of the VQ-VAE experiment where we apply the Sinkhorn-Knopp approximation. As set out in Appendix E.3, the experiment uses $N = 256$ and the target distribution

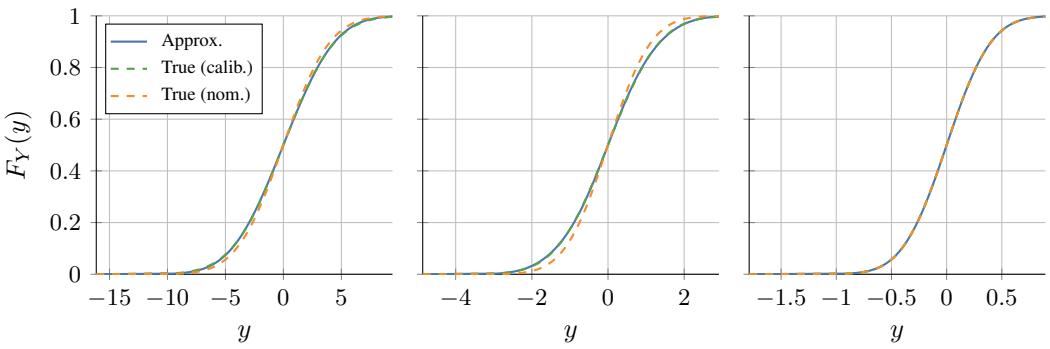


Figure 8: Empirical CDFs for the noise residual of a 1D Gaussian channel using the Sinkhorn-Knopp approximation (Approx.) and the ground-truth CDF (True) at SNRs, from left to right, -10, 1 and 10. The calibrated (calib.) and nominal (nom.) true CDFs are shown.

	Approx.			True		
λ	Mean	P05	P95	Mean	P05	P95
1.0×10^{-1}	2.74	2.27	3.23	2.66	2.26	3.08
2.7×10^{-2}	2.95	2.29	3.96	2.66	2.27	3.10
7.1×10^{-3}	3.71	2.32	7.94	2.66	2.26	3.10
1.9×10^{-3}	3.21	2.31	4.62	2.66	2.27	3.09
5.0×10^{-4}	2.66	2.27	3.09	2.66	2.27	3.09

Table 2: KS test statistics for the conditional output distributions produced from our trained stochastic VQ-VAE using the Sinkhorn-Knopp approximation (Approx.) and ground-truth samples (True). All values have been scaled up by 10^3.

for each x is an isotropic Gaussian with $D = 64$ dimensions. We run each mode through our variable-rate model for 5 different rate parameters λ to generate 10^5 samples each from the learned posterior and record the closeness to the target conditional distribution

$$P_{Y|X}(\cdot \mid x) = \mathcal{N}(\mu_x, \mathrm{diag}(\sigma_x^2)) \tag{212}$$

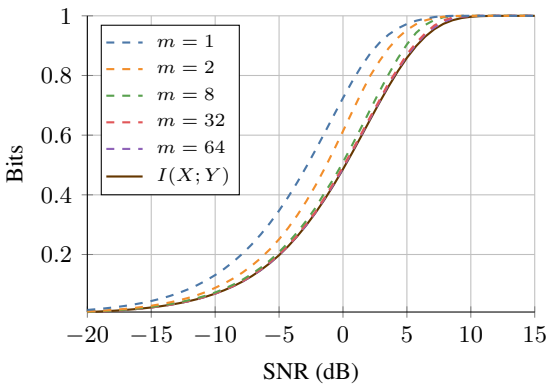
marginally for each of the 64 dimensions over 10 runs using 20 iterations. These results are recorded in Table 2, where we show the mean, 5th and 95th-percentile KS statistics across all modes and all dimensions. While the approximation error in the upper quantiles is somewhat larger than that observed for the marginal KS statistics in Table 1, the downstream rate-distortion results reported in Section 5.1 show that the approximate channel remains effective in the learned end-to-end system.

E.2 Conditional Mutual Information with Multiple Samples Per Mode

We offer some numerical results to illustrate the behavior of the conditional mutual information, $I(X;Y \mid \bar{U}^{Nm})$, which governs the communication cost of the multi-sample permuted scheme as discussed in Section 3.3. As in the setting of the asymptotic result in Theorem 2, we consider $N = 2$. Recall from (47) in the proof of Theorem 2 that for finite m, the expected message length when there are m samples per mode satisfies

$$\mathbb{E}[\ell_m(M)] \leq I(X; K_m \mid \bar{U}^{2m}) + \log(I(X; K_m \mid \bar{U}^{2m}) + 2) + 3. \tag{213}$$

We evaluate a representative example on a BI-AWGN channel with uniformly distributed input $X \in \{-1, 1\}$ and $P_{Y|X}(\cdot \mid x) = \mathcal{N}(x, \sigma^2)$. We plot a Monte Carlo estimate of $I(X;Y \mid \bar{U}^{2m})$ in Fig. 9 for practically computable values of m, using the Poisson binomial expressions developed in Appendix B.4. The number of trials is 1000. One standard error of the mean is shown in the table. By $m = 32$, the conditional mutual information is already close to $I(X;Y)$.



m	$I(X;Y \mid \bar{U}^{2m})$
1	0.7246 ± 0.0098
2	0.6134 ± 0.0075
4	0.5436 ± 0.0045
8	0.5094 ± 0.0028
16	0.4966 ± 0.0019
32	0.4945 ± 0.0013
64	0.4885 ± 0.0009

Figure 9: Conditional mutual information $I(X;Y \mid \bar{U}^{2m})$ for a BI-AWGN channel using m samples per mode. *Left:* Information in bits as a function of SNR. *Right:* Details at $\mathrm{SNR} = 0\,\mathrm{dB}$. For this SNR, $I(X;Y) = 0.4859$.

	IS ($B = 1$)	IS ($B = 2$)	IS ($B = 4$)	Polar-MLC
Time/batch (s)	19.90 ± 0.01	19.67 ± 0.01	19.57 ± 0.03	0.59 ± 0.18

Table 3: Time per batch in seconds for the polar-MLC permuted scheme and IS at different chunk sizes. One batch contains 32 images and IS uses 2^{16} importance samples. Both methods are evaluated on one L40S GPU.

E.3 Variable-Rate Image Compression

In this section, we provide additional details concerning the implementation of the experiments in Section 5.1, dealing in turn with our proposed stochastic VQ-VAE with the polar-MLC implementation of the permuted scheme, then the standard deterministic VQ-VAE baseline, and finally the IS baseline, thus covering all the methods presented in Fig. 2 in the main paper. First, we consider some elements of the setting common to all approaches. We evaluate all the models on the CIFAR-10 dataset upsampled to 64×64 using bilinear interpolation, and adopt the standard split of $50\,000$ train and $10\,000$ test samples. The batch size during training is 64; for testing it is 32, as this corresponds to 8192 latent variables to be transmitted per batch using either our polar-MLC permuted scheme or IS.

Our results report both the mean rate-distortion curve as well as the 5th and 95th-percentile distortions for each rate operating point, where the distortion is the average MSE across a batch of 32 images. The deterministic models are trained on a single L40S GPU with $48\,\mathrm{GB}$ of VRAM, while the variable-rate stochastic models are trained on two such GPUs. Evaluation is carried out on one GPU. In Table 3 we show wall-clock per-batch inference timings measured on the test set, demonstrating an improvement over the IS baseline. Timings are taken under the same conditions as evaluation, to be described in detail below; the rate parameter used is $\lambda = 5 \times 10^{-2}$. In Fig. 10, we provide some additional example output images from our proposed variable-rate stochastic VQ-VAE and the deterministic baseline in low, medium and high-rate regimes.

Stochastic VQ-VAE with Polar-MLC Permuted Scheme. The stochastic VQ-VAE uses the network architecture proposed in van den Oord et al. (2017), based on the publicly available reference implementation.[1] This architecture is also used for the deterministic VQ-VAE and IS baselines. In particular, the model consists of an encoder, a vector quantizer and a decoder, all of which are trained jointly. The encoder contains 2 convolutional downsampling layers with stride 2 and kernel size 4×4, with 2 residual blocks following after. The downsampling layers all have 256 hidden units, while the residual layers contain 64. Each residual block contains a 3×3 convolution followed by a

[1] `github.com/deepmind/sonnet/blob/v2/examples/vqvae_example.ipynb`

1×1 convolution, and ReLU activations are adopted throughout. The decoder mirrors the encoder, with the downsampling layers replaced by the corresponding upsampling operations.

Our codebook for the stochastic models has size $N = 256$ and the dimension of each codeword is $D = 64$. The codebook means are trained using exponential moving average (EMA) updates with a decay of 0.99 where codewords are selected at training and inference time based on the minimum-distance rule in (14); gradients flow back to the encoder through a straight-through estimator (STE). Codebook variances are trained using standard gradient descent.

To achieve rate-conditional training, we condition the decoder network and the learned codebook variances on a scalar input λ using the strategy from Dosovitskiy & Djolonga (2020). Each convolutional and residual layer is followed by a feature-wise linear modulation (FiLM, Perez et al., 2018) block, implemented as a multi-layer perceptron (MLP) with one hidden layer, a hidden dimension of 256, and SiLU activation. If a layer outputs a feature map F with dimensions $W \times H \times C$, each FiLM block learns a modulation $(a(\lambda), b(\lambda))$ where both a and b are vectors of dimension C. Then, the feature map is transformed element-wise as

$$\hat{F}_{ijk} = a_k(\lambda) F_{ijk} + b_k(\lambda). \tag{214}$$

During training, λ is sampled log-uniformly and independently for each sample within a batch between 10^{-5} and 10^{-1}. Models are initialized from a converged deterministic checkpoint. Following this, all parameters apart from the EMA-updated weights are trained using the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and a learning rate of 10^{-4} for 200 epochs, followed by a cosine decay down to 10^{-6} over a further 100 epochs. The decoder is trained on both ground truth samples and approximate samples generated from the Sinkhorn-Knopp approximation introduced in Section 3.2.

To obtain the rate measurements in Fig. 2 under joint coding of the 8192 latents using the polar-MLC implementation of the permuted scheme from Section 4, we report the length of the bitstream returned by arithmetic coding (Rissanen, 1976) divided by the number of latents n. The Sinkhorn-Knopp approximation is used to estimate the discrete posterior $P_{K|X,\bar{U}^N}(\cdot \mid x, u^N)$, and the prior on the codebook is treated as uniform during training but learned across the training set for inference. We denote this learned prior as $\hat{P}_X$ in what follows. Lastly, following the exposition of the modified PolarSim algorithm in Appendix D, the arithmetic coder requires an estimate of the probability $\Pr[\Delta_i = 1]$ for each $i \in [n]$ and, in the MLC case for a 256-ary alphabet, for each level $l \in [8]$. These are computed using 2000 Monte Carlo iterations with the inputs being synthetic discrete latents sampled from the learned prior. During inference, the latents are shuffled using additional shared randomness before being fed to the channel simulator to better satisfy the iid input assumption of our algorithm. The original order is recovered before passing the results to the decoder network. We evaluate on 3 random seeds.

Deterministic VQ-VAE Baseline. The deterministic VQ-VAEs use the same neural architecture, except they do not use rate-conditional training and each codebook entry is simply a fixed vector $c_k \in \mathbb{R}^D$. To achieve different rates, we separately train 7 independent VQ-VAEs with the codebook sizes being $N \in \{4, 8, 16, 32, 64, 128, 256\}$. Again, we use the Adam optimizer, this time with a base learning rate of 2×10^{-4} over 300 epochs, before applying a cosine decay to 10^{-6} for an additional 100 epochs. To estimate the communication cost for the deterministic models, we assume an ideal entropy code given access to the learned prior on the codebook obtained from the training set, giving a lower bound of $\ell(M) \geq \sum_{i=1}^{n} -\log \hat{P}_X(x_i)$ bits for a realization x^n of the full latent vector.

Importance Sampling Baseline. Our implementation follows the communication-efficient IS scheme discussed in Phan et al. (2024), which is a variant of ordered random coding (Theis & Ahmed, 2022), and is evaluated on the same trained $N = 256$ rate-conditional checkpoints used to test our polar-MLC scheme. Thus, we treat IS as an alternative approximate channel simulator, which does not require modification to any other aspects of the concept, network architecture, or training.

We refer the reader to Phan et al. (2024) for full details regarding the IS algorithm and outline the specifics relevant to our experiment here. For a single sample, our proposal distribution is the learned marginal with

$$\hat{p}_Y(y) = \sum_{x=1}^{N} \hat{P}_X(x) \mathcal{N}(y; \mu_x, \mathrm{diag}(\sigma_x^2)) \tag{215}$$

and the target distribution is the specific mode, $P_{Y|X}(\cdot \mid x) = \mathcal{N}(\mu_x, \mathrm{diag}(\sigma_x^2))$. To keep the computational complexity of IS manageable, the 8192 latents are split into chunks of size B and each chunk is accorded 2^{16} iid samples from the marginal. This sample count allows one operating point to be evaluated in approximately 1.5 hours; our polar-MLC method can do the same in 2–3 minutes. In Fig. 2, we evaluate chunk sizes $B \in \{1, 2, 4\}$. While larger chunk sizes reduce the bitrate via amortization, the number of samples becomes insufficient to faithfully reproduce the output distribution at higher rates if B is too large, leading to increased distortion. We report the bitrate using the cross-entropy with the specific Zipf distribution described in Li (2024), which provides a slightly tighter bound than that used in Phan et al. (2024). If an entropy code is constructed using this distribution, a lower bound on the bitrate is then $\ell(M) \geq \sum_{i=1}^{n/B} -\log P_{\mathrm{Zipf}(\alpha)}(k_i)$ where k_i is the sample index returned by importance sampling for chunk i, and $P_{\mathrm{Zipf}(\alpha)}$ is the PMF of a Zipf distribution with parameter $\alpha = 1 + 1/(I(X;Y) + 1)$. We evaluate on 3 random seeds.

E.4 Distributed Mean Estimation

Here, we provide some further information on the DME experiment in Section 5.2, including privacy calibration, simulation with the polar-MLC permuted scheme, the PFR implementation, and finally a comparison to dithered quantization.

Privacy Details and Calibration. First, we go into some further detail on the CDP setting and how privacy budgets are accounted for in order to determine the amount of Gaussian noise to be added. Once the appropriate noise standard deviation has been determined by the privacy accountant, any exact channel simulator may be used to simulate such noise and faithfully reproduce the exact Gaussian mechanism with the associated privacy-utility guarantees and tradeoffs.

We start by giving some more specifics of our DME setup. Recall that the vector X_i held locally by client $i \in [C]$ has dimension $D = 512$, and each coordinate X_{ij}, $j \in [D]$ is independent and uniformly distributed on a set of $N = 16$ points spaced evenly on $[-1, 1]$. More specifically, this input constellation is $\mathcal{X} = \{2(m-1)/15 - 1 : m \in [N]\}$. We therefore have that the client vectors live in the ℓ_2 ball $\mathcal{B}_D(L) = \{v \in \mathbb{R}^D : \|v\|_2 \leq L\}$ with $L = \sqrt{D}$.

Given the desired privacy ε, we apply the Gaussian mechanism by having each client add isotropic Gaussian noise $Z_i \sim \mathcal{N}(0, \sigma_\varepsilon^2 I_D)$ to its local vector. Then, the noisy sample $Y_i = X_i + Z_i$ can be communicated to the server using a channel simulator. The server in turn calculates and later releases the estimate $\hat{\mu} = \sum_{i=1}^{C} Y_i/C = \mu + Z$, where $Z = \sum_{i=1}^{C} Z_i/C$ is the aggregate added noise.

The standard definition of (ε, δ)-CDP with respect to the released mean, see for example Chen et al. (2023), is provided below.

Definition 1 ((ε, δ)-CDP for DME)**.** The mean $\hat{\mu}$ satisfies (ε, δ)-central DP if for any neighboring datasets $x^C = (x_1, \ldots, x_i, \ldots, x_C)$ and $\tilde{x}^C = (x_1, \ldots, \tilde{x}_i, \ldots, x_C)$ and measurable S,

$$\Pr[\hat{\mu} \in S \mid X^C = x^C] \leq e^\varepsilon \Pr[\hat{\mu} \in S \mid X^C = \tilde{x}^C] + \delta. \tag{216}$$

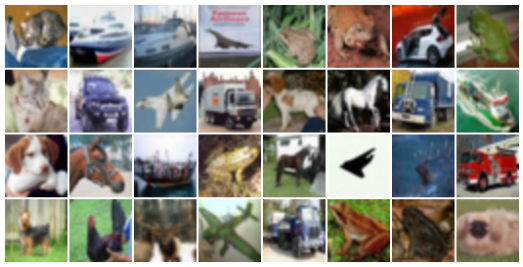
Provided σ_ε is chosen appropriately given ε and δ, the Gaussian mechanism is known to satisfy (216) (Dwork & Roth, 2014). To numerically obtain the required σ_ε, following Liu et al. (2024) we use a Rényi DP accountant. Our implementation uses Google's differential privacy library,[2] which provides a `GaussianDpEvent` that can be calibrated against a desired (ε, δ) pair to return the ratio ν between the standard deviation and the ℓ_2 sensitivity of the target function. With reference to the notion of neighboring datasets in Definition 1, the ℓ_2 sensitivity of the mean is

$$\Delta_2 = \sup_{x^C, \tilde{x}^C} \|\mu(x^C) - \mu(\tilde{x}^C)\|_2 = \frac{1}{C} \sup_{x_i, \tilde{x}_i} \|x_i - \tilde{x}_i\|_2 = \frac{2\sqrt{D}}{C}. \tag{217}$$

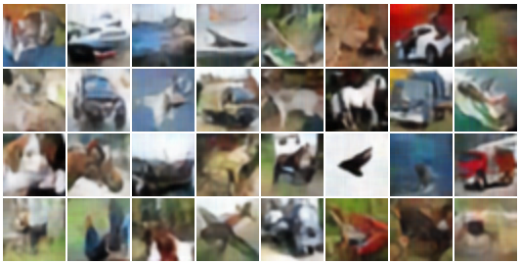
Therefore, upon finding ν from the privacy accountant, the aggregate noise should follow $Z \sim \mathcal{N}(0, (4\nu^2 D/C^2) I_D)$ and the noise added at each client should have $\sigma_\varepsilon = 2\nu\sqrt{D/C}$.

Polar-MLC Permuted Coding Scheme for DME. Once we have obtained the required σ_ε that each client should use, communicating the noisy data Y_i to the server can be framed as a multi-shot

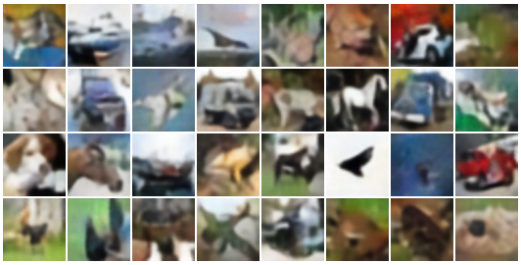
[2] `github.com/google/differential-privacy/`



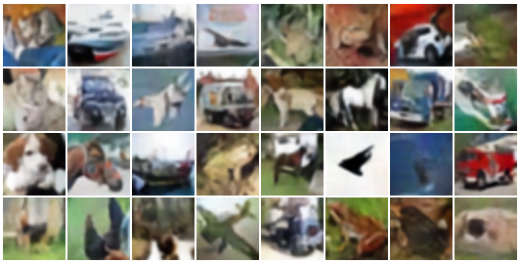
(a) Ground truth



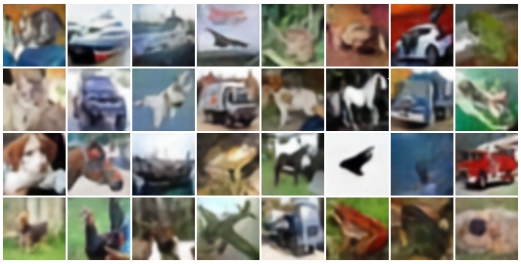
(b) Polar-MLC, $\lambda = 7.2 \times 10^{-2}$, $R = 1.9$



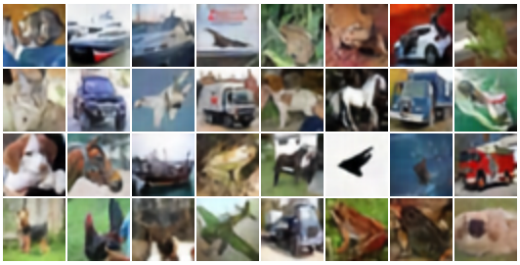
(c) VQ-VAE, $N = 4$, $R = 1.9$



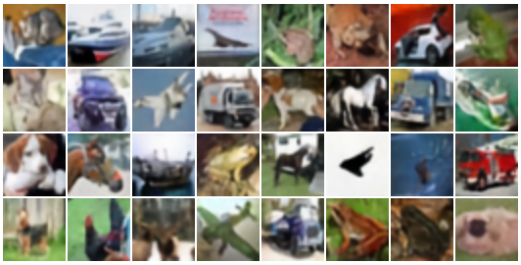
(d) Polar-MLC, $\lambda = 2.5 \times 10^{-2}$, $R = 2.9$



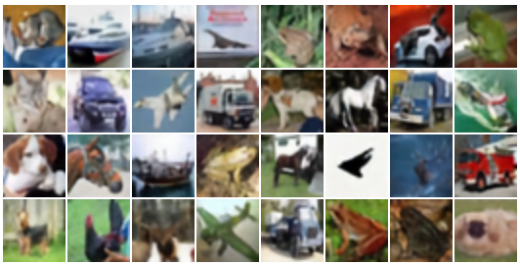
(e) VQ-VAE, $N = 8$, $R = 2.9$



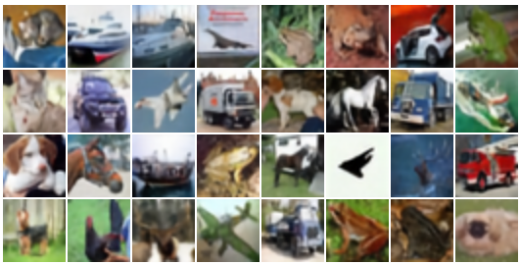
(f) Polar-MLC, $\lambda = 6.0 \times 10^{-3}$, $R = 5.0$



(g) VQ-VAE, $N = 32$, $R = 4.9$



(h) Polar-MLC, $\lambda = 5.0 \times 10^{-4}$, $R = 7.8$



(i) VQ-VAE, $N = 256$, $R = 7.8$

Figure 10: Additional example outputs from the stochastic VQ-VAE experiment using our channel simulation scheme, along with reconstructions from deterministic VQ-VAE baselines achieving similar coding rates. The rate R is in bits/latent.

Index	0	1	2–3	4–10	11–24
B	12	10	8	6	4

Table 4: Chunk size B used for the PFR experiment. The test index refers to evenly spaced values of ε between 0.05 and 30. As ε increases, so does the mutual information and the computational cost at a given chunk size.

channel simulation task with $P_{Y|X}(\cdot \mid x) = \mathcal{N}(x, \sigma_\varepsilon^2)$ and D independent uses of this channel. The alphabet size is $N = 16$, and the exact permuted scheme using matrix permanents and polar-MLC encoding can be used to this end. Because polar coding benefits from longer blocklengths, we jointly encode $B = 8$ instances of the DME problem at a time by concatenating the noisy vectors to be transmitted, giving a coding blocklength of $n = 4096$. We emphasize that this does not affect the privacy or utility properties of the Gaussian mechanism under consideration, the only disadvantage of batched communication being greater latency at the server; privacy guarantees remain on the individual length-D vectors rather than being jointly composed across a batch.

Concerning the implementation, as in Appendix E.3, probability tables for arithmetic coding are obtained from 2000 Monte Carlo iterations. To align with the PFR baseline to be discussed next, we report a tight upper bound on the arithmetic coding cost, $\ell(M) \le \sum_{i=1}^{n} -\log \hat{P}_{\Delta_i}(\Delta_i) + 2$, where Δ^n is the binary correction vector that appears in the modified PolarSim, Algorithm 2, and $\hat{P}_{\Delta_i}(\Delta_i)$ comes from the estimated probability table. The number of bits per sample in Fig. 3 then refers directly to this upper bound applied to the normalized coding cost $\ell(M)/n$ and added across all 4 levels of the multi-level code. We test 50 values of ε and run 200 trials for each.

PFR Baseline. For PFR, we implement the basic scheme from Li & Gamal (2018). We refer the reader to Algorithm 2 in Li (2024) for detailed pseudocode outlining the method. The proposal distribution is the marginal, which in our case is a standard mixture of Gaussians with

$$p_Y(y) = \frac{1}{N} \sum_{x \in \mathcal{X}} \mathcal{N}(y; x, \sigma_\varepsilon^2) \tag{218}$$

and the target distribution is the specific conditional Gaussian, $P_{Y|X}(\cdot \mid x) = \mathcal{N}(x, \sigma_\varepsilon^2)$. To terminate, PFR requires an upper bound on the density ratio for each possible x, $g_x^* \ge \sup_y p_{Y|X}(y \mid x)/p_Y(y)$. Fortunately, such a bound exists for our Gaussian mixture. To enhance the speed of PFR, which becomes slower if a loose stopping bound is used, we precompute the tightest bound numerically for each $x \in \mathcal{X}$ using an optimization routine before entering the sampling loop.

While we have above described the algorithm in scalar terms for simplicity, to achieve a competitive coding cost, we apply PFR jointly on chunks of B samples at a time. As the computational complexity and expected number of samples to be generated increases exponentially in B and also in $I(X;Y)$ (Li, 2024), B cannot be made too large. We therefore sweep B in increments of 2 and choose the largest feasible value of B for each ε, reported in Table 4. Our criteria is that for each ε, a block of 4096 samples can be communicated in less than 10–20 s on an L40S GPU; with reference to Table 5, we emphasize that our exact simulator can process 128 such blocks in approximately 3–19 s on a 32-core CPU. We evaluate 25 values of ε and run 200 trials for each. Following Algorithm 2 in Li (2024), we assume the integer indices returned by PFR are to be encoded using an entropy code designed for a specific Zipf distribution with parameter $\alpha = 1 + 1/(I(X;Y) + 1)$, and report the lower bound $\ell(M) \ge \sum_{i=1}^{n/B} -\log P_{\mathrm{Zipf}(\alpha)}(k_i)$ where k_i is the sample index returned by PFR for block i and $P_{\mathrm{Zipf}(\alpha)}$ is the PMF of the relevant Zipf distribution. While PFR requires a sequential loop to test the stopping condition, see Algorithm 2 in Li (2024), we parallelize by evaluating 2^{16} Poisson arrivals at once.

Performance Benchmarks. We briefly mention the benchmarking setup used to produce the table in Fig. 3. For the polar-MLC permuted scheme, 128 blocks of 4096 samples are transmitted, using one 32-core Intel Xeon Gold 6338 CPU running a C++ backend. We run the benchmark 20 times and report the mean and standard deviation of the wall-clock time. PFR runs on a single L40S GPU, and due to its higher computational cost, we run the benchmark 5 times on 8 blocks, then scale up to estimate the time for 128 blocks. We emphasize that, due to its larger workload per block, PFR does

ε	B	Polar-MLC (s)	PFR (s)	ε	B	Polar-MLC (s)	PFR (s)
0.05	12	3.13 ± 0.31	1353.96 ± 1056.32	16.27	4	4.38 ± 0.26	120.22 ± 0.53
1.30	10	3.00 ± 0.21	748.23 ± 636.66	17.52	4	6.82 ± 0.53	125.47 ± 0.50
2.55	8	2.96 ± 0.16	285.96 ± 19.78	18.77	4	9.04 ± 0.44	129.81 ± 0.63
3.79	8	3.04 ± 0.26	1164.51 ± 37.78	20.02	4	10.80 ± 0.82	133.23 ± 0.72
5.04	6	2.97 ± 0.16	147.62 ± 1.72	21.26	4	10.16 ± 0.65	136.10 ± 0.45
6.29	6	3.02 ± 0.23	283.41 ± 3.02	22.51	4	13.94 ± 0.89	137.35 ± 0.41
7.54	6	3.06 ± 0.27	534.45 ± 3.84	23.76	4	10.81 ± 0.76	138.23 ± 0.46
8.79	6	2.92 ± 0.01	948.25 ± 7.54	25.01	4	15.86 ± 1.13	139.44 ± 0.61
10.03	6	2.97 ± 0.13	1614.22 ± 15.08	26.26	4	12.63 ± 0.83	140.03 ± 0.71
11.28	6	3.06 ± 0.25	2533.94 ± 28.77	27.50	4	13.26 ± 0.96	140.31 ± 0.55
12.53	6	3.00 ± 0.13	3764.37 ± 45.37	28.75	4	18.68 ± 1.25	140.53 ± 1.09
13.78	4	3.04 ± 0.16	107.86 ± 0.46	30.00	4	12.38 ± 0.88	140.69 ± 0.82
15.02	4	3.12 ± 0.19	113.68 ± 0.31				

Table 5: Encoding times in seconds and standard deviation for 128 blocks of 4096 samples, when PFR uses the chunk sizes specified in Table 4.

not perform additional batching at the block level. For ease of comparison between the two schemes, entropy coding, whether using the Zipf distribution for PFR or arithmetic coding for polar-MLC, is not included in the timing data. All tests, including the benchmark, use 3 random seeds. The chunk size for PFR is kept constant at $B = 8$ during the benchmark.

In addition to the benchmark table in Fig. 3, we also provide timings using the varying chunk sizes given in Table 4. Recall that these chunk sizes were chosen such that the time per block for PFR does not become too slow, whereas the Fig. 3 table used a constant chunk size to illustrate the scaling characteristics more clearly. Here we run the benchmark 5 times on 32 blocks for PFR and scale up, while our method again uses 128 blocks and 20 benchmark repeats. Results are shown in Table 5. We suspect the increase in our algorithm's runtime at large ε values may be due to more frequent occurrence of subnormal likelihoods when the noise scale is small.

Comparison with Dithered Quantization. Subtractively dithered quantization (Roberts, 1962; Gray & Stockham, 1993) is a widely-known technique which can exactly simulate additive noise channels, and as such has also been used to achieve DP with additive noise mechanisms, including the Gaussian mechanism which is of interest here (Hasırcıoğlu & Gündüz, 2024; Hegazy et al., 2024). However, as it can only realize additive noise channels, it is not considered a general-purpose channel simulation scheme in the same sense as PFR, GRS or our permuted construction.

Nevertheless, for completeness we provide in this section a comparison with the algorithm from Hasırcıoğlu & Gündüz (2024), which is specifically designed to solve the CDP DME problem using the Gaussian mechanism. Given a scalar $X = x$, to transmit a sample $Y \sim \mathcal{N}(x, \sigma_\varepsilon^2)$, the scheme performs the following:

1. Generate a sample $V \sim \Gamma(3/2, 1/2)$, where $\Gamma(3/2, 1/2)$ is the gamma distribution with shape and rate parameters $3/2$ and $1/2$ respectively.
2. Given $V = v$, compute the quantization step size as $\Delta = 2\sigma_\varepsilon\sqrt{v}$ and sample $U \sim \mathrm{Unif}[-\Delta/2, \Delta/2]$.
3. Apply the quantization function $Q(w) = \lfloor (x - \Delta/2)/\Delta \rceil \Delta + \Delta/2$ to $x + U$.
4. Transmit the result using a fixed-length code consuming $b = \lceil \log_2(2\lfloor 1/\Delta + 1 \rceil) \rceil$ bits.

The above description is for a scalar channel. For a higher-dimensional channel such as ours, where 4096 samples must be transmitted, the scalar procedure is simply repeated and the resulting codewords concatenated. In Fig. 11, we show the results obtained using this scheme and the fixed-length code described above in the same setting as that of the PFR comparison in Fig. 3. We note that, unlike the polar-MLC permuted scheme or PFR, the fixed-length construction does not approach zero bits/sample as $\varepsilon \to 0$, as it incurs at least 1 bit of communication for each coordinate.

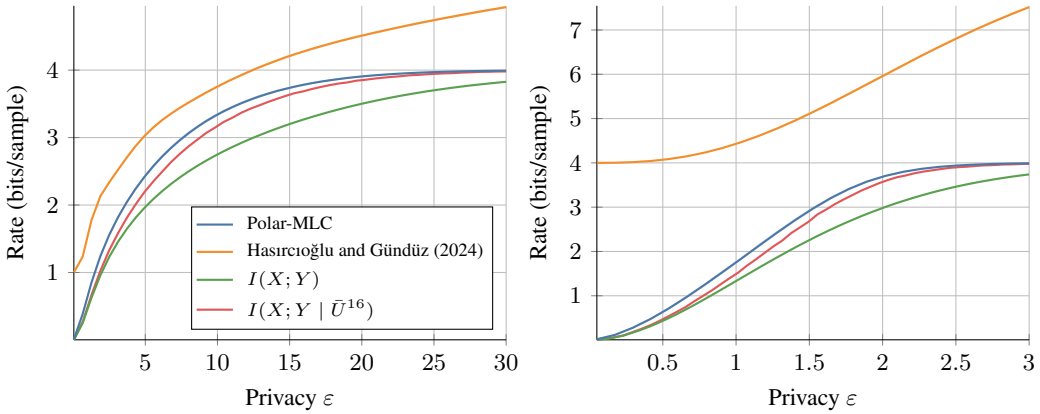


Figure 11: Comparison with dithered quantization for distributed mean estimation under the Gaussian mechanism. *Left:* Scalar 16-ary channel. *Right:* 4D 16-ary channel. P05–P95 rates are shown.

As the scheme in Hasırcıoğlu & Gündüz (2024) operates coordinate-wise, it may perform more poorly when the dimension of each sample is increased. To illustrate this, we also consider a setting where the alphabet size is still $N = 16$, but the constellation $\mathcal{X}$ is now the set of vertices of a 4D hypercube with side length 1. This maintains the same ℓ_2 sensitivity of the mean despite the change in geometry, and the experimental results are shown in Fig. 11 for $\varepsilon \in [0.05, 3]$. As each dimension of the constellation must be simulated individually using the scalar protocol, the dithered quantization scheme incurs additional overhead while our more general-purpose polar-MLC permuted scheme, which is agnostic to the geometry of the constellation, readily generalizes. We use 3 random seeds.

As an aside, we note that while variants on dithered quantizers directly simulating Gaussian noise in higher dimensions have recently been considered (Kobus et al., 2024), they currently do not achieve exact simulation and therefore cannot guarantee privacy via the usual properties of the Gaussian mechanism. Achieving exact simulation currently requires, for example, more complex schemes including an inner rejection sampling loop (Ling et al., 2025), which incurs greater computational complexity and is beyond the scope of our experiment. For scalar approaches, variable-length codes may also be used to encode the quantized index instead of a fixed-length code. For example, Hegazy et al. (2024) and Shahmiri et al. (2024) adopt Elias gamma coding, which we find underperforms the fixed-length code in our problem setting. Entropy codes conditioned on the shared randomness can theoretically obtain even better rate performance but generally require constructing a new code for each realization, which may be infeasible in practice, as noted in Hegazy et al. (2024). This is outside the scope of the present comparison; we leave the consideration of such coding schemes and any potential simplifications in the discrete-to-continuous setting to future work.

E.5 Alternative Approximate Scheme via Polar-Coded Importance Sampling

Following from the discussion of importance sampling in Appendix E.3, we briefly propose an alternative scheme borrowing some of the machinery from Section 4 to perform approximate channel simulation via IS. Instead of using the approach in Phan et al. (2024) and sending an index into a sorted list whose order is determined using the shared randomness, by noticing that IS induces a explicit posterior and prior distribution on the selected index K, we can directly encode samples from this distribution using polar codes.

In particular, let the number of importance samples be m and the channel $P_{Y|X}$. Fixing a proposal distribution P_Y, we sample $\bar{U}^m \sim P_Y$ iid using shared randomness, thus creating an iid sample pool. As in Phan et al. (2024), selecting the index K following

$$P_{K|X,\bar{U}^m}(k \mid x, u^m) = \frac{\lambda_k(x, u^m)}{\sum_{k'=1}^{m} \lambda_{k'}(x, u^m)}, \ \lambda_k(x, u^m) = \frac{p_{Y|X}(u_k \mid x)}{p_Y(u_k)} \tag{219}$$

given $X = x$ serves to produce $Y = \bar{U}_K$ as an approximate sample from $P_{Y|X}(\cdot \mid x)$. Assuming P_X is known, the corresponding prior $P_{K|\bar{U}^m}$ for use at the decoder can also be computed from

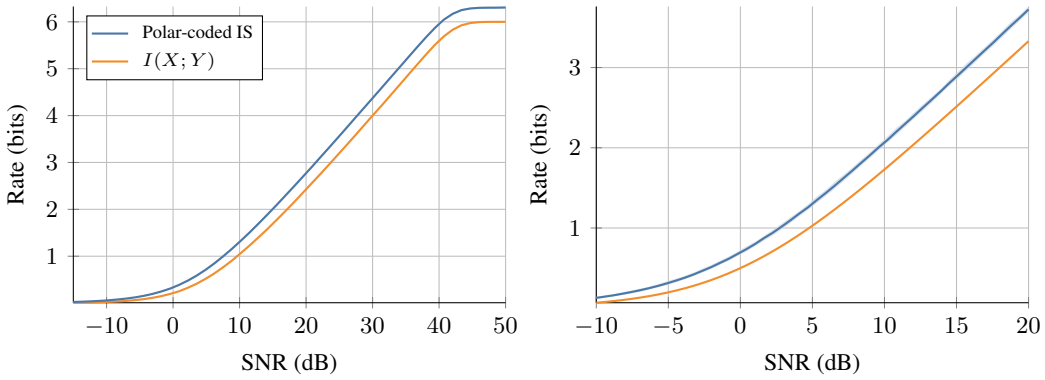


Figure 12: Rate of polar-coded IS for approximate channel simulation. *Left:* 64-ary input Gaussian channel. *Right:* 1D AWGN channel with unit signal power. P05–P95 rates are shown.

(219). We emphasize that unlike the permuted scheme, IS with finite m is fundamentally limited to approximate channel simulation.

Considering n uses of the channel with a source $X^n \sim P_X$ iid, we can perform the IS procedure identically across each time index, with m fresh importance samples generated independently within each step, to generate a sequence K^n and corresponding approximate outputs Y^n. Given this memoryless structure and assuming m is a power of 2, the multi-level procedure described in Section 4.2 can be used to communicate a sample from $P_{K|X,\bar{U}^m}$.

While we leave more detailed theoretical and empirical study of this scheme to future work, we provide two simple examples here to demonstrate its rate performance, as its implementation reuses in large part the polar and multi-level coding techniques already presented in this paper. We note that the approximation quality of IS has been analyzed extensively in prior work, e.g. Phan et al. (2024); Theis & Ahmed (2022); Chatterjee & Diaconis (2018). The application of polar codes in IS-based approximate channel simulation is however new to the best of our knowledge.

N-ary Gaussian Channel. First, we consider a N-ary input channel with 1D Gaussian noise and $N = 64$. We place the 64 input modes evenly in $[-1, 1]$. The noise variance is $\sigma^2 = 10^{-\mathrm{SNR}/10}$, where $\mathrm{SNR} \in [-15, 50]$. We use $m = 2^{10}$ importance samples at each timestep and report the rate achieved by arithmetic coding in Fig. 12.

Continuous-Input AWGN Channel. Polar-coded IS can in principle be extended to approximate channel simulation with a continuous input. As a simple example, we consider a setting similar to the approximate AWGN simulation in Sriramu et al. (2024, Appendix B), where an input $X^n \sim \mathcal{N}(0, 1)$ iid is to be simulated with additive Gaussian noise. For this preliminary experiment, we restrict our attention to 1D noise. This noise is $Z^n \sim \mathcal{N}(0, \sigma^2)$ iid such that $P_Y = \mathcal{N}(0, 1+\sigma^2)$, which we take as the proposal distribution for the $m = 2^{10}$ importance samples. We have $I(X;Y) = \log(1 + \sigma^{-2})/2$. Unlike for the N-ary input channel, marginalizing over X to compute $P_{K|\bar{U}^m}$ is not convenient for general continuous inputs. Therefore, noting that the proof of correctness of the modified PolarSim in Appendix D does not depend on the prior used at the decoder, we instead let the decoder assume a uniform distribution over K. We note that the rate analysis from Appendix D, which does depend on the true prior on K being available, no longer applies; extending this result may be of future interest along with evaluation of the approximation quality at different sample counts. The empirically observed rate using arithmetic coding is shown in Fig. 12.