Faster Relative Entropy Coding with Greedy Rejection Coding

Anonymous Author(s)
Affiliation
Address
email

Abstract

Relative entropy coding (REC) algorithms encode a sample from a target distribution Q using a proposal distribution P using as few bits as possible. Unlike entropy coding, REC does not assume discrete distributions or require quantisation. As such, it can be naturally integrated into communication pipelines such as learnt compression and differentially private federated learning. Unfortunately, despite their practical benefits, REC algorithms have not seen widespread application, due to their prohibitively slow runtimes or restrictive assumptions. In this paper, we make progress towards addressing these issues. We introduce Greedy Rejection Coding (GRC), which generalises the rejection based-algorithm of Harsha et al. (2007) to arbitrary probability spaces and partitioning schemes. We first show that GRC terminates almost surely and returns unbiased samples from Q, after which we focus on two of its variants: GRCS and GRCD. We show that for continuous Q and P over $\mathbb{R}$ with unimodal density ratio dQ/dP, the expected runtime of GRCS is upper bounded by $\beta D_{\mathrm{KL}}[Q\|P] + \mathcal{O}(1)$ where $\beta \approx 4.82$, and its expected codelength is optimal. This makes GRCS the first REC algorithm with guaranteed optimal runtime for this class of distributions, up to the multiplicative constant β. This significantly improves upon the previous state-of-the-art method, A* coding (Flamich et al., 2022). Under the same assumptions, we experimentally observe and conjecture that the expected runtime and codelength of GRCD are upper bounded by $D_{\mathrm{KL}}[Q\|P] + \mathcal{O}(1)$. Finally, we evaluate GRC in a variational autoencoder-based compression pipeline on MNIST, and show that a modified ELBO and an index-compression method can further improve compression efficiency.

1 Introduction and motivation

Over the past decade, the development of excellent deep generative models (DGMs) such as variational autoencoders (VAEs; Vahdat & Kautz, 2020; Child, 2020), normalising flows (Kingma et al., 2016) and diffusion models (Ho et al., 2020) demonstrated great promise in leveraging machine learning (ML) for data compression. Many recent learnt compression approaches have significantly outperformed the best classical hand-crafted codecs across a range of domains including, for example, lossless and lossy compression of images and video (Zhang et al., 2021; Mentzer et al., 2020, 2022).

Transform coding. Most learnt compression algorithms are *transform coding* methods: they first map a datum to a latent variable using a learnt transform, and encode it using entropy coding (Ballé et al., 2020). Entropy coding assumes discrete variables while the latent variables in DGMs are typically continuous, so most transform coding methods quantize the latent variable prior to entropy coding. Unfortunately, quantization is a non-differentiable operation. Thus, state-of-the-art DGMs trained with gradient-based optimisation must resort to some continuous approximation to quantisation during training and switch to hard quantisation for compression. Previous works have argued that using quantisation within learnt compression is restrictive or otherwise harmful, and that

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

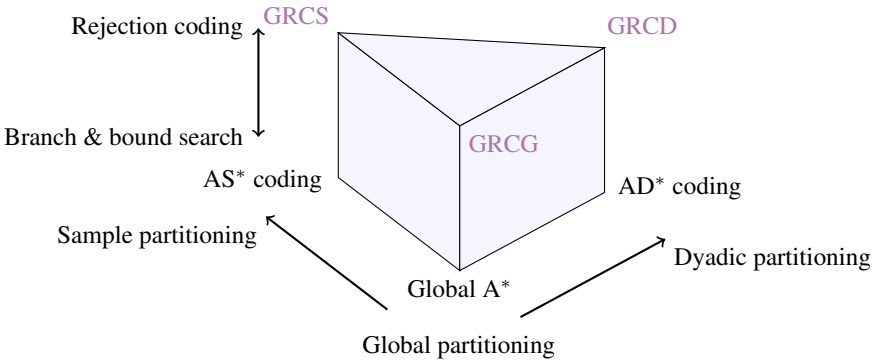


Figure 1: An illustration of the relations between the variants of GRC, introduced in this work, and the variants of A* coding. Algorithms in purple are introduced in this work. The algorithms of Harsha et al. (2007) and Li & El Gamal (2018) are equivalent to GRCG and Global A* coding respectively.

a method which naturally interfaces with continuous latent variables is needed (Havasi et al., 2018; Flamich et al., 2020; Theis & Agustsson, 2021; Flamich et al., 2022).

Relative entropy coding. In this paper, we study *relative entropy coding* (REC; Havasi et al., 2018; Flamich et al., 2020), an alternative to quantization and entropy coding. A REC algorithm uses a proposal distribution P, and a public source of randomness S, to produce a random code which represents a *single sample* from a target distribution Q. Thus REC does not assume discrete distributions and interfaces naturally with continuous variables. Remarkably, REC has fundamental advantages over quantization in lossy compression with realism constraints (Theis & Agustsson; Theis et al., 2022). More generally, it finds application across a range of settings including, for example, differentially private compression for federated learning (Shah et al., 2022).

Limitations of existing REC algorithms. While algorithms for solving REC problems already exist, most of them suffer from limitations that render them impractical. These limitations fall into three categories: prohibitively long runtimes, overly restrictive assumptions, or additional coding overheads. In this work, we study and make progress towards addressing these limitations.

General-purpose REC algorithms. On the one hand, some REC algorithms make very mild assumptions and are therefore applicable in a wide range of REC problems (Harsha et al., 2007; Li & El Gamal, 2018). Unfortunately, these algorithms have prohibitively long runtimes. This is perhaps unsurprising in light of a result by Agustsson & Theis (2020), who showed that without additional assumptions on Q and P, the worst-case expected runtime of any general-purpose REC algorithm scales as $2^{D_{\mathrm{KL}}[Q\|P]}$, which is impractically slow. There are also REC algorithms which accept a desired runtime as a user-specified parameter, at the expense of introducing bias in their samples (Havasi et al., 2018; Theis & Yosri, 2022). Unfortunately, in order to reduce this bias to acceptable levels, these algorithms require runtimes of an order of $2^{D_{\mathrm{KL}}[Q\|P]}$, and are therefore also impractical.

Faster algorithms with additional assumptions. On the other hand, there exist algorithms which make additional assumptions in order to achieve faster runtimes. For example, dithered quantisation (Ziv, 1985; Agustsson & Theis, 2020) achieves an expected runtime of $D_{\mathrm{KL}}[Q\|P]$, which is optimal since any REC algorithm has an expected runtime of at least $D_{\mathrm{KL}}[Q\|P]$. However, it requires both Q and P to be uniform distributions, which limits its applicability. Recently, Flamich et al. (2022) introduced A* coding, an algorithm based on A* sampling (Maddison et al., 2014) which, under assumptions satisfied in practice, achieves an expected runtime of $D_{\infty}[Q\|P]$. Unfortunately, this runtime is sub-optimal and is not always practically fast, since $D_{\infty}[Q\|P]$ can be arbitrarily large for fixed $D_{\mathrm{KL}}[Q\|P]$. Further, as discussed in Flamich et al. (2022) this runtime also comes at a cost of an additional, substantial, overhead in codelength, which limits the applicability of A* coding.

Our contributions. In this work, we address some of these limitations. First, we propose *greedy rejection coding* (GRC), a REC algorithm based on rejection sampling. Then, inspired by A* coding (Flamich et al., 2022), we develop GRCS and GRCD, two variants of GRC that partition the sample space to dramatically speed up termination. Figure 1 illustrates the relations between GRC and its variants with existing algorithms. We analyze the correctness and the runtime of these algorithms and, in particular, prove that GRCS has an optimal codelength and order-optimal runtime on a wide class of one-dimensional problems. In more detail, our contributions are:

- We introduce Greedy Rejection Coding (GRC), which generalises the algorithm of Harsha et al. (2007) to arbitrary probability spaces and partitioning schemes. We prove that under mild conditions, GRC terminates almost surely and returns an unbiased sample from Q.

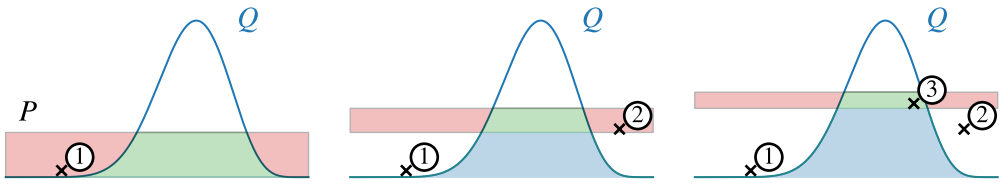


Figure 2: Example run of Harsha et al. (2007), for a pair of continuous Q and P over $[0, 1]$. The green and red regions correspond to acceptance and rejection regions at each step. Here the algorithm rejects the first two samples and accepts the third one, terminating at the third step.

- We introduce GRCS and GRCD, two variants of GRC for continuous distributions over $\mathbb{R}$, which adaptively partition the sample space to dramatically improve their convergence, inspired by AS* and AD* coding (Flamich et al., 2022), respectively.
- We prove that whenever dQ/dP is unimodal, the expected runtime and codelength of GRCS is $\mathcal{O}(D_{\mathrm{KL}}[Q\|P])$. This significantly improves upon the $\mathcal{O}(D_\infty[Q\|P])$ runtime of AS* coding, which is always larger than that of GRCS. This runtime is order-optimal, while making far milder assumptions than, for example, ditered quantization.
- We provide clear experimental evidence for and conjecture that whenever dQ/dP is unimodal, the expected runtime and codelength of GRCD are $D_{\mathrm{KL}}[Q\|P]$. This also significantly improves over the $D_\infty[Q\|P]$ empirically observed runtime of AD* coding.
- We implement a compression pipeline with VAEs, using GRC to compress MNIST images. We propose a modified ELBO objective and show that this, together with a practical method for compressing the indices returned by GRC further improve compression efficiency.

2 Background and related work

Relative entropy coding. First, we define REC algorithms. Definition 1 is stricter than the one given by Flamich et al. (2022), as it has a stronger condition on the the algorithms' expected codelength. In this paper, all logarithms are to the base 2, and all divergences are measured in bits.

Definition 1 (REC algorithm)**.** *Let $(\mathcal{X}, \Sigma)$ be a measurable space, let $\mathcal{R}$ be a set of pairs of distributions (Q, P) over $(\mathcal{X}, \Sigma)$ such that $D_{\mathrm{KL}}[Q\|P] < \infty$ and $\mathcal{P}$ be the set of all distributions P such that $(Q, P) \in \mathcal{R}$ for some distribution Q. Let $S = (S_1, S_2, \dots)$ be a publicly available sequence of independent and fair coin tosses, with corresponding probability space $(\mathcal{B}, \mathcal{F}, \mathbb{P})$ and let $\mathcal{C} = \{0, 1\}^*$ be the set of all finite binary sequences. A REC algorithm is a pair of functions $\mathtt{enc} : \mathcal{R} \times \mathcal{B} \to \mathcal{C}$ and $\mathtt{dec} : \mathcal{C} \times \mathcal{P} \times \mathcal{B} \to \mathcal{X}$, such that for each $(Q, P) \in \mathcal{R}$, the outputs of the encoder $C = \mathtt{enc}(Q, P, S)$ and the decoder $X = \mathtt{dec}(P, C, S)$ satisfy*

$$X \sim Q \quad \textit{and} \quad \mathbb{E}_S[|C|] = D_{\mathrm{KL}}[Q\|P] + \mathcal{O}(\log D_{\mathrm{KL}}[Q\|P]), \tag{1}$$

where $|C|$ is the length of the string C. We call `enc` *the encoder and* `dec` *the decoder.*

In practice, S is implemented with a pseudo-random number generator (PRNG) with a public seed. In the remainder of this section, we discuss relevant REC algorithms, building up to GRC in section 3.

Existing REC algorithms. While there are many REC algorithms already, they suffer from various issues limiting their applicability in practice. Our proposed algorithm, Greedy Rejection Coding (GRC), is based on and generalises the rejection-based algorithm of Harsha et al. (2007), by drawing inspiration from A* coding (Flamich et al., 2022). Specifically, A* coding can be viewed as a generalisation of an algorithm due to Li & El Gamal (2018). The former generalises the latter by introducing a partitioning scheme to speed up termination. In an analogous fashion, GRC generalises Harsha et al. (2007) by also introducing partitioning schemes, to speed up termination and achieve optimal runtimes. Here we discuss relevant algorithms, building up to GRC in section 3.

REC with rejection sampling. Harsha et al. (2007) introduced a REC algorithm based on rejection sampling, which we generalise and extend in this work. While this algorithm was originally presented for discrete Q and P, we will show that it can be generalised to arbitrary probability spaces. In this section, we present this generalised version and in section 3 we further extend it to arbitrary partitioning schemes (see definition 5). The generalisation to arbitrary probability spaces relies on the Radon-Nikodym derivative dQ/dP, which is guaranteed to exist since $Q \ll P$ by definition 1. When Q and P both have densities, dQ/dP coincides with the density ratio.

At each step, the algorithm draws a sample from P and performs an accept-reject step, as illustrated in fig. 2. If it rejects the sample, it rules out part of Q corresponding to the acceptance region, adjusts

the proposal to account for the removed mass, and repeats until acceptance. More formally, define T_0 to be the zero-measure on $(\mathcal{X}, \Sigma)$, and recursively for $d \in \mathbb{N}$, set:

$$T_{d+1}(S) \stackrel{\text{def}}{=} T_d(S) + A_{d+1}(S), \qquad A_{d+1}(S) \stackrel{\text{def}}{=} \int_S \alpha_{d+1}(x)\, dP(x), \tag{2}$$

$$t_d(x) \stackrel{\text{def}}{=} \frac{dT_d}{dP}(x), \qquad \alpha_{d+1}(x) \stackrel{\text{def}}{=} \min\left\{ \frac{dQ}{dP}(x) - t_d(x), (1 - T_d(\mathcal{X})) \right\}, \tag{3}$$

$$X_d \sim P,\ U_d \sim \text{Uniform}(0,1) \qquad \beta_{d+1}(x) \stackrel{\text{def}}{=} \frac{\alpha_{d+1}(x)}{1 - T_d(\mathcal{X})}, \tag{4}$$

for all $x \in \mathcal{X}, S \in \Sigma$. The algorithm terminates at the first occurrence of $U_d \leq \beta_{d+1}(X_d)$. The T_d measure corresponds to the mass that has been ruled off up to and including the d^{th} rejection: $T_1(\mathcal{X}), T_2(\mathcal{X})$ and $T_3(\mathcal{X})$ are the sums of the blue and green masses in the left, middle and right plots of fig. 2 respectively. The A_d measure corresponds to the acceptance mass at the d^{th} step: $A_1(\mathcal{X}), A_2(\mathcal{X})$ and $A_3(\mathcal{X})$ are the masses of the green regions in the left, middle and right plots of fig. 2 respectively. Lastly, t_d, α_d are the Radon-Nikodym derivatives i.e., roughly speaking, the densities, of T_d, A_d with respect to P, and $\beta_{d+1}(X_d)$ is the probability of accepting the sample X_d. Here, the encoder `enc` amounts to keeping count of the number of rejections that occur up to the first acceptance, setting C equal to this count and returning X and C. The decoder `dec` amounts to drawing $C + 1$ samples from P, using the same seed as the encoder, and returning the last of these samples. While this algorithm is elegantly simple and achieves optimal codelengths, Flamich & Theis (2023) showed its expected runtime is $2^{D_\infty[Q\|P]}$, where $D_\infty[Q\|P] = \sup_{x\in\mathcal{X}} \log(dQ/dP)(x)$ is the Rényi ∞-divergence. Unfortunately, this is prohibitively slow in most practical cases.

REC with Poisson & Gumbel processes. Li & El Gamal (2018) introduced a REC algorithm based on Poisson processes, referred to as Poisson Functional Representation (PFR). PFR assumes that dQ/dP is bounded above, and relies on the fact that (Kingman, 1992), if T_n are the ordered arrival times of a homogeneous Poisson process on $\mathbb{R}^+$ and $X_n \sim P$, then

$$N \stackrel{\text{def}}{=} \arg\min_{n\in\mathbb{N}} \left\{ T_n \frac{dP}{dQ}(X_n) \right\} \implies X_N \sim Q, \tag{5}$$

Therefore, PFR casts the REC problem into an optimisation, or search, problem, which can be solved in finite time almost surely. The PFR encoder draws pairs of samples T_n, X_n, until it solves the search problem in eq. (5), and returns $X = X_N, C = N - 1$. The decoder can recover X_N from (P, C, S), by drawing N samples from P, using the same random seed, and keeping the last sample. While, like the algorithm of Harsha et al. (2007), PFR is elegantly simple and achieves optimal codelengths, its expected runtime is also $2^{D_\infty[Q\|P]}$ unfortunately (Maddison, 2016).

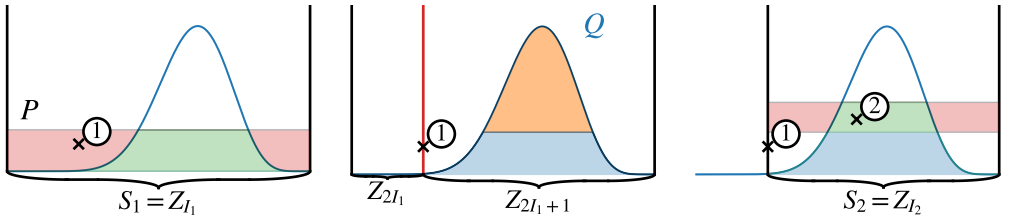
Fast REC requires additional assumptions. These algorithms' slow runtimes are perhaps unsurprising considering Agustsson & Theis's result, which shows under the computational hardness assumption $\text{RP} \neq \text{NP}$ that without making additional assumptions on Q and P, there is no REC algorithm whose expected runtime scales *polynomially* in $D_{\text{KL}}[Q\|P]$. Therefore, in order achieve faster runtimes, a REC algorithm must make additional assumptions on Q and P.

A* coding. To this end, Flamich et al. (2022) proposed: (1) a set of appropriate assumptions which are satisfied by many deep latent variable models in practice and (2) a REC algorithm, referred to as A* coding, which leverages these assumptions to achieve a substantial speed-up over existing methods. In particular, A* coding generalizes PFR by introducing a partitioning scheme, which splits the sample space $\mathcal{X}$ in nested partitioning subsets, to speed up the solution of eq. (5). Drawing inspiration from this, our proposed algorithm generalises eqs. (2) to (4) in an analogous manner (see fig. 1), introducing partitioning processes (definition 2) to speed up the algorithm's termination.

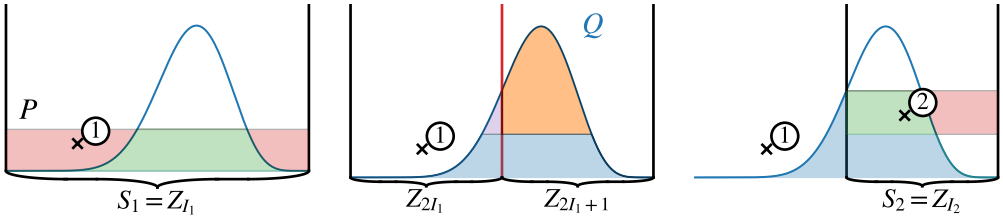
Definition 2 (Partitioning process)**.** *A partitioning process is a process $Z : \mathbb{N}^+ \to \Sigma$ such that*

$$Z_1 = \mathcal{X}, \ \ Z_{2n} \cap Z_{2n+1} = \emptyset, \ \ Z_{2n} \cup Z_{2n+1} = Z_n. \tag{6}$$

In other words, a partitioning process Z is a process indexed by the heap indices of an infinite binary tree, where the root node is $\mathcal{X}$ and any two children nodes Z_{2n}, Z_{2n+1} partition their parent node Z_n. In section 3 we present specific choices of partitioning processes which dramatically speed up GRC.



(a) Sample & accept or reject (b) Partition & sample $b_1 \in \{0, 1\}$ (c) Sample & accept or reject



(d) Sample & accept or reject (e) Partition & sample $b_1 \in \{0, 1\}$ (f) Sample & accept or reject

Figure 3: Illustrations of the two variants of GRC considered in this work. (a) to (c) show GRC with the *sample-splitting* partitioning process (GRCS). (d) to (f) show GRC with the dyadic partition process (GRCD). GRC interleaves accept-reject steps with partitioning steps. In the former, it draws a sample and either accepts or rejects it. In the latter, it partitions the sample space and randomly chooses one of the partitions, ruling out large parts of the sample space and speeding up termination.

Greedy Poisson Rejection Sampling. Contemporary to our work, Anonymous (2023) introduces a rejection sampler based on Poisson processes, which can be used as a REC algorithm referred to as Greedy Poisson Rejection Sampling (GPRS). Similar to GRC and A* coding, GPRS partitions the sample space to speed up the convergence to the accepted sample. Furthermore, a variant of GPRS also achieves order-optimal runtime for one-dimensional distribution pairs with a unimodal density ratio. However, the construction of their method is significantly different from ours, relying entirely on Poisson processes. Moreover, GPRS requires numerically solving a certain ODE, while our method does not, making it potentially more favourable in practice. We believe establishing a closer connection between GPRS and GRC is a promising future research direction.

3 Greedy Rejection Coding

Generalising Harsha et al. (2007). In this section we introduce Greedy Rejection Coding (GRC; definition 5), which generalises the algorithm of Harsha et al. (2007) in two ways. First, GRC can be used with distributions over arbitrary probability spaces. Therefore, it is applicable to arbitrary REC problems, including REC with continuous distributions. Second, similar to A* coding, GRC can be combined with arbitrary partitioning processes, allowing it to achieve optimal runtimes given additional assumptions on the REC problem, and an appropriate choice of partitioning process.

3.1 Algorithm definition

Overview. Before specifying GRC, we summarise its operation with an accompanying illustration. On a high level, GRC interleaves accept-reject steps with partitioning steps, where the latter are determined by a partitioning process. Specifically, consider the example in figs. 3d to 3f, where Q and P are distributions over $\mathcal{X} = [0, 1]$, and Z is the partitioning process defined by

$$Z_n = [L, R] \implies Z_{2n} = [L, M), Z_{2n+1} = [M, R], \text{ where } M = (L + R)/2. \tag{7}$$

In each step $d = 1, 2, \ldots$, GRC maintains a heap index I_d of an infinite binary tree, and an active subset $S_d = Z_{I_d} \subseteq \mathcal{X}$ of the sample space, initialised as $I_0 = 1$ and $S_1 = Z_1 = \mathcal{X}$ respectively.

Accept-reject step. In each accept-reject step, GRC draws a sample from the restriction of P to S_d, namely $P|_{S_d}/P(S_d)$. If the sample is accepted, the algorithm terminates. Otherwise, GRC performs a partitioning step as shown in fig. 3d

Partitioning step. In each partitioning step, GRC partitions $S_d = Z_{I_d}$ into Z_{2I_d} and Z_{2I_d+1}, as specified by the partitioning process Z. It then samples a Bernoulli random variable b_d, whose outcomes have probabilities proportional to the mass of Q which has not been accounted for, up to and including step d, within the partitions Z_{2I_d} and Z_{2I_d+1} respectively. In fig. 3e, these two masses correspond to the purple and orange areas, and the algorithm has sampled $b_d = 1$. Last, GRC

Algorithm 1 Harsha et al.'s rejection algorithm; equivalent to GRC with a global partition

Require: Target Q, proposal P, space $\mathcal{X}$

$d \leftarrow 0, T_0 \leftarrow 0$

while `True` **do**
  $X_{d+1} \sim P$
  $U_{d+1} \sim \text{Uniform}(0,1)$
  $\beta_{d+1} \leftarrow \texttt{AcceptProb}(Q, P, X_{d+1}, T_d)$
  if $U_{d+1} \leq \beta_{d+1}$ **then**
    return X_{d+1}, d
  end if
  $T_{d+1} \leftarrow \texttt{RuledOutMass}(Q, P, T_d)$
  $d \leftarrow d+1$
end while

Algorithm 2 GRC with partition process Z; differences to Harsha et al.'s algorithm shown in green

Require: Target Q, proposal P, space $\mathcal{X}$, partition Z

$d \leftarrow 0, T_0 \leftarrow 0$
$I_0 \leftarrow 1, S_1 \leftarrow \mathcal{X}$
while `True` **do**
  $X_{I_d} \sim P|_{S_d}/P(S_d)$
  $U_{I_d} \sim \text{Uniform}(0,1)$
  $\beta_{I_d} \leftarrow \texttt{AcceptProb}(Q, P, X_{I_d}, T_d)$
  if $U_{I_d} \leq \beta_{d+1}$ `or` $d = D_{\max}$ **then**
    return X_{I_d}, I_d
  end if
  $p \leftarrow \texttt{PartitionProb}(Q, P, T_d, Z_{2d}, Z_{2d+1})$
  $b_d \sim \text{Bernoulli}(p)$
  $I_{d+1} \leftarrow 2I_d + b_d$ and $S_{d+1} \leftarrow Z_{I_{d+1}}$
  $T_{d+1} \leftarrow \texttt{RuledOutMass}(Q, P, T_d, S_{d+1})$
  $d \leftarrow d+1$
end while

updates the heap index to $I_{d+1} = 2I_d + b_d$ and the active subset to $S_{d+1} = Z_{I_{d+1}}$. GRC proceeds by interleaving accept-reject and partitioning steps until an acceptance occurs.

Algorithm specification. The aforementioned algorithm can be formalised in terms of probability measures over arbitrary spaces and arbitrary partitioning processes. Above, algorithms 1 and 2 describe Harsha et al.'s rejection sampler and our generalisation of it, respectively. For the sake of keeping the exposition lightweight, we defer the formal measure-theoretic definition of GRC to the appendix (see definition 5 in appendix A.1), and refer to algorithm 2 as a working definition here.

Comparison to Harsha et al. While algorithms 1 and 2 are similar, they differ in two notable ways. First, rather than drawing a sample from P, GRC draws a sample from the restriction of P to an active subset $S_d = Z_d \subseteq \mathcal{X}$, namely $P|_{S_d}/P(S_d)$. Second, GRC updates its active subset $S_d = Z_d$ at each step, setting it to one of the children of Z_d, namely either Z_{2d} or Z_{2d+1}, by drawing $b_d \sim$ Bernoulli, and setting Z_{2d+b_d}. This partitioning mechanism, which does not appear in algorithm 1, yields a different variant of GRC for each choice of partitioning process Z. In fact, as shown in Proposition 1 below, algorithm 1 is a special case of GRC with $S_d = \mathcal{X}$ for all d. See appendix A.2 for the proof.

Proposition 1 (Harsha et al. (2007) is a special case of GRC)**.** *Let Z be the global partitioning process over Σ, defined as*

$$Z_1 = \mathcal{X}, \quad Z_{2n} = Z_n, \quad Z_{2n+1} = \emptyset, \quad \textit{for all } \; n = 1, 2, \ldots. \tag{8}$$

*Harsha et al. (2007) is equivalent to GRC using this Z and setting $C = D^*$ instead of $C = I_{D^*}$. We refer to this algorithm as Global GRC, or **GRCG** for short.*

Partitioning processes and additional assumptions. While Proposition 1 shows that Harsha et al.'s algorithm is equivalent to GRC with a particular choice of Z, a range of other choices of Z is possible, and this is where we can leverage additional structure. In particular, we show that when Q and P are continuous distributions over $\mathbb{R}$ with a unimodal density ratio dQ/dP, we can dramatically speed up GRC with an appropriate choice of Z. In particular, we will consider the sample-splitting and dyadic partitioning processes from Flamich et al. (2022), given in Definitions 3 and 4.

Definition 3 (Sample-splitting partitioning process)**.** *Let $\mathcal{X} = \mathbb{R} \cup \{-\infty, \infty\}$ and P a continuous distribution. The sample-splitting partitioning process is defined as*

$$Z_n = [a, b], a, b \in \mathcal{X} \implies Z_{2n} = [a, X_n], \;\; Z_{2n+1} = [X_n, b], \; \textit{where } X_n \sim P|_{Z_n}/P(Z_n).$$

In other words, in the sample-splitting process, Z_n are intervals of $\mathbb{R}$, each of which is partitioned into sub-intervals Z_{2n} and Z_{2n+1} by splitting at the sample X_n drawn from $P|_{Z_n}/P(Z_n)$. We refer to GRC with the sample-splitting partitioning process as **GRCS**.

Definition 4 (Dyadic partitioning process)**.** *Let $\mathcal{X} = \mathbb{R} \cup \{-\infty, \infty\}$ and P a continuous distribution. The dyadic partitioning process is defined as*

$$Z_n = [a, b], a, b \in \mathcal{X} \implies Z_{2n} = [a, c], \;\; Z_{2n+1} = [c, b], \; \textit{such that } P(Z_{2n}) = P(Z_{2n+1}).$$

Similar to the sample-splitting process, in the dyadic process Z_n are intervals of $\mathbb{R}$. However, in the dyadic process, Z_n is partitioned into sub-intervals Z_{2n} and Z_{2n+1} such that $P(Z_{2n}) = P(Z_{2n+1})$. We refer to GRC with the dyadic partitioning process as **GRCD**.

GRC with a tunable codelength. Flamich et al. presented a depth-limited variant of AD* coding, DAD* coding, in which the codelength $|C|$ can be provided as a tunable input to the algorithm. Fixed-codelength REC algorithms are typically approximate because they introduce bias in their samples, but are nevertheless useful in certain contexts, such as for coding a group of random variables with the same fixed codelength. GRCD can be similarly modified to accept $|C|$ as an input, by limiting the maximum steps of the algorithm by $D_{\max}$ (see algorithm 2). Setting $D_{\max} = \infty$ in algorithm 2 corresponds to exact GRC, while setting $D_{\max} < \infty$ corresponds to depth-limited GRC.

3.2 Theoretical results

Correctness of GRC. In theorem 1 we show that GRC terminates almost surely and produces unbiased samples from Q, given interchangeable mild assumptions on Q, P and Z. Assumption 1 is the most general, since it holds for any Q and P over arbitrary probability spaces, and can be used to apply GRC to arbitrary coding settings.

Assumption 1. *GRC has a finite ratio mode if $dQ/dP(x) < M$ for all $x \in \mathcal{X}$, for some $M \in \mathbb{R}$.*

Assumption 1 holds for GRCG, GRCS and GRCD, so long as dQ/dP is bounded. While this assumption is very general, in some cases we may want to consider Q, P with unbounded dQ/dP. To this end, we show that it can be replaced by alternative assumptions, such as assumptions 2 and 3.

Assumption 2. *GRC is single-branch if for each d, $b_d = 0$ or $b_d = 1$ almost surely.*

GRC with the global partitioning process (eq. 8) satisfies assumption 2. In addition, if Q and P are distributions over $\mathbb{R}$ and dQ/dP is unimodal, GRCS also satisfies assumption 2.

Assumption 3. *Suppose $\mathcal{X} \subseteq \mathbb{R}^N$. GRC has nicely shrinking Z if, almost surely, the following holds. For each $x \in \mathcal{X}$ which is in a nested sequence of partitions $x \in Z_1 \supseteq \cdots \supseteq Z_{k_d} \supseteq \ldots$ with $P(Z_{k_d}) \to 0$, there exist $\gamma, r_1, r_2, ... \in \mathbb{R}_{>0}$ such that*

$$r_d \to 0,\ Z_{k_d} \subseteq B_{r_d}(x) \text{ and } P(Z_{k_d}) \geq \gamma P(B_{r_d}(x)). \tag{9}$$

If Q and P are distributions over $\mathbb{R}$, GRCD satisfies assumption 3. Theorem 1 shows that if any of the above assumptions hold, then GRC terminates almost surely and yields unbiased samples from Q. We provide the proof of the theorem in appendix B.

Theorem 1 (Correctness of GRC)**.** *Suppose Q, P and Z satisfy any one of assumptions 1 to 3. Then, algorithm 2 terminates with probability* 1*, and its returned sample X has law $X \sim Q$.*

Expected runtime and codelength of GRCS. Now we turn to the expected runtime and codelength of GRCS. Theorem 2 shows that the expected codelength of GRCS is optimal, while Theorem 3 establishes that its runtime is order-optimal. We present the proofs of the theorems in appendix C.

Theorem 2 (GRCS codelength)**.** *Let Q and P be continuous distributions over $\mathbb{R}$ such that $Q \ll P$ and with unimodal dQ/dP. Let Z be the sample-splitting process, and X its returned sample. Then,*

$$\mathbb{H}[X|Z] \leq D_{\mathrm{KL}}[Q\|P] + 2\log\left(D_{\mathrm{KL}}[Q\|P] + 1\right) + \mathcal{O}(1). \tag{10}$$

Theorem 3 (GRCS runtime)**.** *Let Q and P be continuous distributions over $\mathbb{R}$ such that $Q \ll P$ and with unimodal dQ/dP. Let Z be the sample-splitting process and D the number of steps the algorithm takes before accepting a sample. Then, for $\beta = 2/\log(4/3) \approx 4.82$ we have*

$$\mathbb{E}[D] \leq \beta\ D_{\mathrm{KL}}[Q\|P] + \mathcal{O}(1) \tag{11}$$

Improving the codelength of GRCD. In Theorem 2 we state the bound for the REC setting, where we make no further assumptions on Q and P. However, we can improve the bound if we consider the *reverse channel coding* (RCC) setting (Theis & Yosri, 2022). In RCC, we have a pair of correlated random random variables $X, Y \sim P_{X,Y}$. During one round of communication, the encoder receives $Y \sim P_Y$ and needs to encode a sample $X \sim P_{X|Y}$ from the posterior using P_X as the proposal distribution. Thus, RCC can be thought of as the average-case version of REC, where the encoder sets $Q \leftarrow P_{X|Y}$ and $P \leftarrow P_X$. In this case, when the conditions of Theorem 2 hold for every $(P_{X|Y}, P_X)$ pair, in appendix C we show that the bound can be improved to $\mathbb{I}[X;Y] + \log(\mathbb{I}[X;Y] + 1) + \mathcal{O}(1)$, where $\mathbb{I}[X;Y] = \mathbb{E}_{Y\sim P_Y}\left[D_{\mathrm{KL}}[P_{X|Y}\|P_Y]\right]$ is the mutual information between X and Y.

GRCS runtime is order-optimal. Theorem 3 substantially improves upon the runtime of A* coding, which is the current fastest REC algorithm with similar assumptions. In particular, AS* coding has $\mathcal{O}(D_\infty[Q\|P])$ expected runtime, which can be arbitrarily larger than that of GRCS. Remarkably, the runtime of GRCS is optimal up to the multiplicative factor β. This term arises from the fact the sample-splitting process may occasionally rule out a small part of the sample space at a given step.

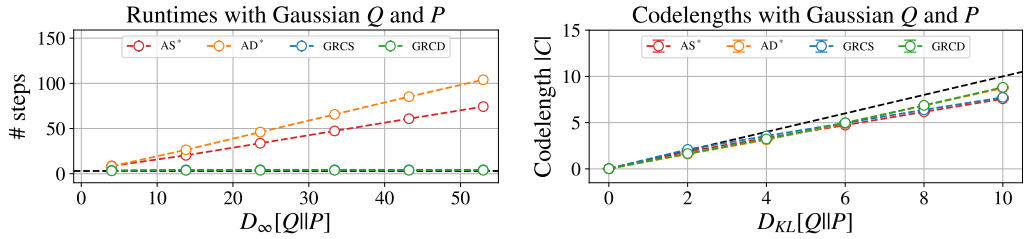


Figure 4: Comparison between GRC and A* coding on synthetic REC problems with Gaussian Q and P. *Left:* we fix $D_{\mathrm{KL}}[Q\|P] = 3$ and vary $D_\infty[Q\|P]$, measuring the number of steps taken by each algorithm. *Right:* we fix $D_\infty[Q\|P] = D_{\mathrm{KL}}[Q\|P] + 2$ and vary $D_{\mathrm{KL}}[Q\|P]$, plotting the codelengths produced by each algorithm. Reported codelengths do not include additional logarithmic overhead terms. Results are averaged over 4×10^3 different random seeds for each datapoint. We have included error-bars in both plots but these are too small to see compared to the plot scales.

4 Experiments

We conducted two sets of experiments: one on controlled synthetic REC problems to check the predictions of our theorems numerically, and another using VAEs trained on MNIST to study how the performance of GRC-based compression pipelines can be improved in practice. We conducted all our experiments under fair and reproducible conditions and make our source code public.[1]

4.1 Synthetic Experiments

Synthetic REC experiments. First, we compare GRCS and GRCD, against AS* and AD* coding, on a range of synthetic REC problems. We systematically vary distribution parameters to adjust the difficulty of the REC problems. Figure 4 shows the results of our synthetic experiments.

Partitioning processes improve the runtime of GRC. First, we observe that, assuming that dQ/dP is unimodal, introducing an appropriate partitioning process such as the sample-splitting or the dyadic process, dramatically speeds up GRC. In particular, fig. 4 shows that increasing the infinity divergence $D_\infty[Q\|P]$ (for a fixed $D_{\mathrm{KL}}[Q\|P]$) does not affect the runtimes of GRCS and GRCD, which remain constant and small. This is a remarkable speed-up over the exponential expected runtime of GRCG.

GRC is faster than A* coding. Further, we observe that GRC significantly improves upon the runtime of A* coding, which is the fastest previously known algorithm with similar assumptions. In particular, Figure 4 shows that increasing the infinity divergence $D_\infty[Q\|P]$, while keeping the KL divergence $D_{\mathrm{KL}}[Q\|P]$ fixed, increases the runtime of both AS* and AD* coding, while the runtimes of GRCS and GRCD remain constant. More generally, for a fixed KL divergence, the infinity divergence can be arbitrarily large or even infinite. In such cases, A* coding would be impractically slow or even inapplicable, while GRCS and GRCD remain practically fast.

GRCD improves on GRCS. In our experiments, we observe that the performance of GRCD (green in fig. 4) matches that of GRCS (blue in fig. 4) in terms of runtime and codelength. While in our experiments, GRCD does not yield an improvement over GRCS, we note the following behaviour. The sample-splitting process may occasionally rule out a only a small part of space, which can slow down convergence. In particular, in appendix C we show that on average, the sample-splitting process rules out $1/2$ of the active sample space in the best case at each step, and $3/4$ in the worst case. By contrast, the dyadic process always rules out $1/2$ of the sample space, potentially speeding up termination. We conjecture that GRCD achieves an optimal expected runtime with $\beta = 1$.

4.2 Compression with Variational Autoencoders

Compressing images with VAEs and REC. One of the most promising applications of REC is in learnt compression. Here, we implement a proof-of-concept lossless neural compression pipeline using a VAE with a factorized Gaussian posterior on MNIST and take the architecture used by Townsend et al. (2018). To compress an image Y, we encode a latent sample X from the VAE posterior $q(X \mid Y)$ by applying GRCD dimensionwise after which we encode the image Y with entropy coding using the VAE's conditional likelihood $p(Y \mid X)$ as the coding distribution. Unfortunately, in addition to the $D_{\mathrm{KL}}[q(X_d \mid Y)\|p(X_d)]$ bits coding cost for latent dimension d, this incurs an overhead of $\log(D_{\mathrm{KL}}[q(X_d \mid Y)\|p(X_d)] + 1) + \mathcal{O}(1)$ bits, analogously to how a symbol code, like Huffman coding, incurs a constant overhead per symbol (MacKay, 2003). However, since $\log(1 + x) \approx x$ when $x \approx 0$, the logarithmic overhead of GRC can become significant when the KL over Hence, we now investigate two approaches to mitigate this issue.

[1] Source code to be published with the camera-ready version: `https://github.com/source-code`.

Training objective	# latent	Total BPP with ζ coding	Total BPP with δ coding	Neg. ELBO per pixel	Overhead BPP with δ coding
ELBO	20	1.472 ± 0.004	1.482 ± 0.004	1.391 ± 0.004	0.091 ± 0.000
	50	1.511 ± 0.003	1.530 ± 0.003	1.357 ± 0.003	0.172 ± 0.000
	100	1.523 ± 0.003	1.600 ± 0.003	1.362 ± 0.003	0.238 ± 0.000
Modified ELBO	20	1.470 ± 0.004	1.478 ± 0.004	1.393 ± 0.004	0.085 ± 0.000
	50	1.484 ± 0.003	1.514 ± 0.003	1.373 ± 0.003	0.141 ± 0.000
	100	1.485 ± 0.003	1.579 ± 0.003	1.373 ± 0.003	0.205 ± 0.000

Table 1: Lossless compression performance comparison on the MNIST test set of a small VAE with different latent space sizes, optimized using either the ELBO or the modified ELBO in eq. (12). We report the bits per pixel (BPP) attained using different coding methods, averaged over the 10,000 test images, along with the standard error, using GRCD. See section 4.2 for further details.

Modified ELBO for REC. A principled approach to optimizing our neural compression pipeline is to minimize its expected codelength. For bits-back methods (Townsend et al., 2018, 2019), the negative ELBO indeed expresses their expected codelength, but in REC's case, it does not take into account the additional dimensionwise logarithmic overhead we discussed above. Thus, we propose to minimize a modified negative ELBO to account for this (assuming that we have D latent dimensions):

$$\underbrace{\mathbb{E}_{X\sim q(X|Y)}[-\log p(Y|X)] + D_{\mathrm{KL}}[q(X|Y)\|p(X)]}_{\text{Regular ELBO}} + \sum_{d=1}^{D} \underbrace{\log\left(D_{\mathrm{KL}}[q(X_d|Y)\|p(X_d)] + 1\right)}_{\text{Logarithmic overhead per dimension}}. \quad (12)$$

Coding the latent indices. As the final step during the encoding process, we need a prefix code to encode the heap indices I_d returned by GRCD for each d. Without any further information, the best we can do is use Elias δ coding (Elias, 1975), which, assuming our conjecture on the expected runtime of GRCD holds, yields an expected codelength of $\mathbb{I}[Y;X] + 2\log(\mathbb{I}[Y;X]+1) + \mathcal{O}(1)$. However, we can improve this if we can estimate $\mathbb{E}[\log I_d]$ for each d: it can be shown, that the maximum entropy distribution of a positive integer-valued random variable with under a constraint on the expectation on its logarithm is $\zeta(n|\lambda) \propto n^{-\lambda}$, with $\lambda^{-1} = \mathbb{E}[\log I_d] + 1$. In this case, entropy coding I_d using this ζ distribution yields improves the expected codelength to $\mathbb{I}[Y;X] + \log(\mathbb{I}[Y;X]+1) + \mathcal{O}(1)$.

Experimental results. We trained our VAE with $L \in \{20, 50, 100\}$ latent dimensions optimized using the negative ELBO and its modified version in Equation (12), and experimented with encoding the heap indices of GRCD with both δ and ζ coding. We report the results of our in Table 1 on the MNIST test set in bits per pixel. In addition to the total coding cost, we report the negative ELBO per pixel, which is the fundamental lower bound on the compression efficiency of REC with each VAE. Finally, we report the logarithmic overhead due to δ coding. We find that both the modified ELBO and ζ coding prove beneficial, especially as the dimensionality of the latent space increases. This is expected, since the overhead is most significant for latent dimensions with small KLs, which becomes more likely as the dimension of the latent space grows. The improvements yielded by each of the two methods are significant, with ζ coding leading to a consistent $1-7\%$ gain compared to δ coding and the modified objective resulting in up to 2% gain in coding performance.

5 Conclusion and Future Work

Summary. In this work, we introduced Greedy Rejection Coding (GRC), a REC algorithm which generalises the rejection algorithm of Harsha et al. to arbitrary probability spaces and partitioning processes. We proved the correctness of our algorithm under mild assumptions, and introduced GRCS and GRCD, two variants of GRC. We showed that the runtimes of GRCS and GRCD significantly improve upon the runtime of A* coding, which can be arbitrarily larger. We evaluated our algorithms empirically, verifying our theory and conducted a proof-of-concept learnt compression experiment on MNIST using VAEs. We demonstrated that a principled modification to the ELBO and entropy coding GRCD's indices using a ζ distribution can further improve compression efficiency.

Limitations and Further work. One limitation of GRC is that, unlike A* coding, it requires us to be able to evaluate the CDF of Q. While in some settings this CDF may be intractable, this assumption is satisfied by most latent variable generative models, and is not restrictive in practice. However, one practical limitation of GRCS and GRCD, as well as AS* and AD* , is that they assume target-proposal pairs over $\mathbb{R}$. For multivariate distributions, we can decompose them into univariate conditionals and apply GRC dimensionwise, however this incurs an additional coding overhead per dimension, resulting in a non-negligible cost. Thus, an important direction is to investigate whether fast REC algorithms for multivariate distributions can be devised, to circumvent this challenge.

References

Eirikur Agustsson and Lucas Theis. Universally quantized neural compression. *Advances in Neural Information Processing Systems*, 33, 2020.

Anonymous. Greedy Poisson rejection sampling. *See the Supplementary Materials*, 2023.

Johannes Ballé, Philip A Chou, David Minnen, Saurabh Singh, Nick Johnston, Eirikur Agustsson, Sung Jin Hwang, and George Toderici. Nonlinear transform coding. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):339–353, 2020.

Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. *CoRR*, abs/2011.10650, 2020. URL `https://arxiv.org/abs/2011.10650`.

Richard M Dudley. *Real analysis and probability*. CRC Press, 2018.

Nelson Dunford and Jacob T Schwartz. *Linear operators, part 1: general theory*, volume 10. John Wiley & Sons, 1988.

Peter Elias. Universal codeword sets and representations of the integers. *IEEE transactions on information theory*, 21(2):194–203, 1975.

Gergely Flamich and Lucas Theis. Adaptive greedy rejection sampling. *arXiv preprint arXiv:2304.10407*, 2023.

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

Gergely Flamich, Stratis Markou, and Jose Miguel Hernandez-Lobato. Fast relative entropy coding with A* coding. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 6548–6577. PMLR, 17–23 Jul 2022.

Prahladh Harsha, Rahul Jain, David McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)*, pp. 10–23. IEEE, 2007.

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*, 2018.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020. URL `https://arxiv.org/abs/2006.11239`.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

J.F.C. Kingman. *Poisson Processes*. Oxford Studies in Probability. Clarendon Press, 1992. ISBN 9780191591242. URL `https://books.google.co.uk/books?id=VEiM-OtwDHkC`.

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.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

CA Maddison. Poisson process model for Monte Carlo. *Perturbation, Optimization, and Statistics*, pp. 193–232, 2016.

Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. *Advances in Neural Information Processing Systems*, 27:3086–3094, 2014.

Fabian Mentzer, George D Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression. *Advances in Neural Information Processing Systems*, 33:11913–11924, 2020.

Fabian Mentzer, George Toderici, David Minnen, Sergi Caelles, Sung Jin Hwang, Mario Lucic, and Eirikur Agustsson. Vct: A video compression transformer. In *Advances in Neural Information Processing Systems*, 2022.

Fernando Pérez-Cruz. Kullback-Leibler divergence estimation of continuous distributions. In *2008 IEEE international symposium on information theory*, pp. 1666–1670. IEEE, 2008.

Walter Rudin. *Real and Complex Analysis*. McGraw-Hill Science/Engineering/Math, 1986.

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

L. Theis and E. Agustsson. On the advantages of stochastic encoders. In *Neural Compression Workshop at ICLR*, 2021. URL `https://arxiv.org/abs/2102.09270`.

Lucas Theis and Eirikur Agustsson. On the advantages of stochastic encoders. In *Neural Compression: From Information Theory to Applications–Workshop@ ICLR 2021*.

Lucas Theis and Noureldin Yosri. Algorithms for the communication of samples. In *International Conference on Machine Learning*, 2022.

Lucas Theis, Tim Salimans, Matthew D Hoffman, and Fabian Mentzer. Lossy compression with gaussian diffusion. *arXiv preprint arXiv:2206.08889*, 2022.

James Townsend, Thomas Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. In *International Conference on Learning Representations*, 2018.

James Townsend, Thomas Bird, Julius Kunze, and David Barber. Hilloc: lossless image compression with hierarchical latent variable models. In *International Conference on Learning Representations*, 2019.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in neural information processing systems*, 33:19667–19679, 2020.

Shifeng Zhang, Ning Kang, Tom Ryder, and Zhenguo Li. iflow: Numerically invertible flows for efficient lossless compression via a uniform coder. *CoRR*, abs/2111.00965, 2021. URL `https://arxiv.org/abs/2111.00965`.

Jacob Ziv. On universal quantization. *IEEE Transactions on Information Theory*, 31(3):344–347, 1985.

A Formal definition of Greedy Rejection Coding

A.1 Formal definition

Here we give a formal definition of GRC in terms of measures. We chose to omit this from the main text for the sake of exposition, and instead formally define GRC in definition 5 below.

Definition 5 (Greedy Rejection Coding)**.** *Let Z be a partitioning process on Σ, and $I_0 = 1$, $S_0 = Z_{I_0}$. Let $T_0(\cdot, S_0)$ be the zero-measure on $(\mathcal{X}, \Sigma)$. Then for $d = 0, 1, \ldots$ define*

$$t_d(x, S_{0:d}) \stackrel{\text{def}}{=} \frac{dT_d(\cdot, S_{0:d})}{dP(\cdot)}(x), \tag{13}$$

$$\alpha_{d+1}(x, S_{0:d}) \stackrel{\text{def}}{=} \min\left\{\frac{dQ}{dP}(x) - t_d(x, S_{0:d}), \frac{1 - T_d(\mathcal{X}, S_{0:d})}{P(S_d)}\right\} \tag{14}$$

$$A_{d+1}(S, S_{0:d}) \stackrel{\text{def}}{=} \int_S dP(x)\ \alpha_{d+1}(x, S_{0:d}), \tag{15}$$

$$\beta_{d+1}(x, S_{0:d}) \stackrel{\text{def}}{=} \alpha_{d+1}(x, S_{0:d})\ \frac{P(S_d)}{1 - T_d(\mathcal{X}, S_{0:d})}, \tag{16}$$

$$X_{I_d} \sim \frac{P|_{S_d}}{P(S_d)}, \tag{17}$$

$$U_{I_d} \sim \text{Uniform}(0, 1), \tag{18}$$

$$b_d \sim \text{Bernoulli}\left(\frac{Q(Z_{2I_d+1}) - T_d(Z_{2I_d+1}, S_{0:d}) - A_{d+1}(Z_{2I_d+1}, S_{0:d})}{Q(S_d) - T_d(S_d, S_{0:d}) - A_{d+1}(S_d, S_{0:d})}\right), \tag{19}$$

$$I_{d+1} \stackrel{\text{def}}{=} 2I_d + b_d, \tag{20}$$

$$S_{d+1} \stackrel{\text{def}}{=} Z_{I_{d+1}}, \tag{21}$$

$$T_{d+1}(S, S_{0:d+1}) \stackrel{\text{def}}{=} T_d(S \cap S_{d+1}, S_{0:d}) + A_{d+1}(S \cap S_{d+1}, S_{0:d}) + Q(S \cap S'_{d+1}), \tag{22}$$

where $S \in \Sigma$ and $P|_{Z_d}$ denotes the restriction of the measure P to the set Z_d. Generalised Greedy Rejection Coding (GRC) amounts to running this recursion, computing

$$D^* = \min\{d \in \mathbb{N} : U_{I_d} \leq \beta_{d+1}(X_{I_d}, S_{0:d})\}, \tag{23}$$

and returning $X = X_{I_{D^*}}$ and $C = I_{D^*}$.

The functions `AcceptProb` and `RuledOutMass` in algorithm 2 correspond to calculating the quantities in eq. (16) and eq. (22). The function `PartitionProb` corresponds to computing the success probability of the Bernoulli coin toss in eq. (19).

A.2 Harsha et al.'s algorithm is a special case of GRC

Here we show that the algorithm of Harsha et al. is a special case of GRC which assumes discrete P and Q distributions and uses the global partitioning process, which we refer to as GRCG. Note that the original algorithm described by Harsha et al. assumes discrete P and Q distributions, whereas GRCG does not make this assumption.

Proposition 2 (Harsha et al. (2007) is a special case of GRC)**.** *Let Z be the global partitioning process over Σ, defined as*

$$Z_1 = \mathcal{X}, \quad Z_{2n} = Z_n, \quad Z_{2n+1} = \emptyset, \quad \textit{for all } \ n = 1, 2, \ldots. \tag{24}$$

Harsha et al. (2007) is equivalent to GRC using this Z and setting $C = D^*$ instead of $C = I_{D^*}$. We refer to this variant of GRC as Global GRC, or GRCG for short.

Proof. With Z defined as in eq. (24), we have $b_d \sim \text{Bernoulli}(0)$ by eq. (19), so $b_d = 0$ almost surely. Therefore $S_d = \mathcal{X}$ for all $d \in \mathbb{N}^+$. From this, we have $T_{d+1}(S, S_{0:d}) = T_d(S, S_{0:d}) + A_d(S, S_{0:d})$ and also $P(S_d) = P(\mathcal{X}) = 1$ for all $d \in \mathbb{N}^+$. Substituting these in the equations of definition 5, we recover eqs. (2) to (4). Setting $C = D^*$ instead of $C = I_{D^*}$ makes the two algorithms identical. □

B Proof of correctness of GRC: Theorem 1

In this section we give a proof for the correctness of GRC. Before going into the proof, we outline our approach and the organisation of the proof.

Proof outline. To prove theorem 1, we consider running GRC for a finite number of d steps. We consider the measure $\tau_d : \Sigma \to [0, 1]$, defined such that for any $S \in \Sigma$, the quantity $\tau_d(S)$ is equal to the probability that GRC terminates within d steps and returns a sample $X \in S \subseteq \Sigma$. We then show that $\tau_d \to Q$ in total variation as $d \to \infty$, which proves theorem 1.

Organisation of the proof. First, in section B.1 we introduce some preliminary definitions, assumptions and notation on partitioning processes, which we will use in later sections. Then, in B.2 we derive the τ_d measure, and prove some intermediate results about it. Specifically, proposition 3 shows that the measures A_d and T_d from the definition of GRC (definition 5) correspond to probabilities describing the termination of the algorithm, and lemma 1 uses these facts to derive the form of τ_d in terms of A_d. Then, lemma 2 shows that the measure τ_d is no larger than the measure Q and lemma 3 shows that the limit of τ_d as $d \to \infty$ is also a measure. Lastly lemma 4 shows that T_d and τ_d are equal on the active sets of the partition process followed within a run of GRC, and then lemma 5 uses that result to derive the subsets of the sample space on which τ_d is equal to Q and τ is equal to Q.

Then, in appendix B.3 we break down the proof of theorem 1 in four cases. First, we consider the probability p_d that GRC terminates at step d, given that it has not terminated up to and including step $d - 1$. Lemma 7 shows that if $p_d \not\to 0$, then $\tau_d \to Q$ in total variation. Then we consider the case $p_d \to 0$ and show that in this case, if any of assumptions 1, 2 or 3 hold, then again $\tau_d \to Q$ in total variation. Putting these results together proves theorem 1.

B.1 Preliminary definitions, assumptions and notation

For the sake of completeness, we restate relevant definitions and assumptions. Definition 6 restates our notation on the target Q and proposal P measures and assumption 4 emphasises our assumption that $Q \ll P$. Definition 7 restates the definition of partitioning processes.

Definition 6 (Target Q and proposal P distributions)**.** *Let Q and P be probability measures on a measurable space $(\mathcal{X}, \Sigma)$. We refer to Q and P as the target and proposal measures respectively.*

Assumption 4 ($Q \ll P$)**.** *We assume Q is absolutely continuous w.r.t. P, that is $Q \ll P$. Under this assumption, the Radon-Nikodym derivative of Q w.r.t. P exists and is denoted as $dQ/dP : \mathcal{X} \to \mathbb{R}^+$.*

Definition 7 (Partitioning process)**.** *A random process $Z : \mathbb{N}^+ \to \Sigma$ which satisfies*

$$Z_1 = \mathcal{X}, \;\; Z_{2n} \cap Z_{2n+1} = \emptyset, \;\; Z_{2n} \cup Z_{2n+1} = Z_n. \tag{25}$$

is called a partitioning process.

That is, a partitioning process Z is a random process indexed by the heap indices of an infinite binary tree, where the root node is $\mathcal{X}$ and any two children nodes Z_{2n} and Z_{2n+1} partition their parent node Z_n. Note that by definition, a partitioning process takes values which are measurable sets in $(\mathcal{X}, \Sigma)$.

Because GRC operates on an binary tree, we find it useful to define some appropriate notation. Definition 8 specifies the ancestors of a node in a binary tree. Notation 1 gives some useful indexing notation for denoting different elements of the partitioning process Z, as well as for denoting the branch of ancestors of an element in a partitioning process.

Definition 8 (Ancestors)**.** *We define the one-step ancestor function $A_1 : 2^{\mathbb{N}^+} \to 2^{\mathbb{N}^+}$ as*

$$A_1(N) = N \cup \{n \in \mathbb{N}^+ : n' = 2n \text{ or } n' = 2n + 1, \text{ for some } n' \in N\}, \tag{26}$$

and the ancestor function $A : 2^{\mathbb{N}^+} \to 2^{\mathbb{N}^+}$ as

$$A(N) = \left\{n \in \mathbb{N}^+ : n \in A_1^k(\{n'\}) \text{ for some } n' \in N, k \in \mathbb{N}^+\right\}. \tag{27}$$

where A_1^k denotes the composition of A_1 with itself k times.

Viewing $\mathbb{N}^+$ as the set of heap indices of an infinite binary tree, A maps a set $N \subseteq \mathbb{N}$ of natural numbers (nodes) to the set of all elements of N and their ancestors.

Notation 1 (Double indexing for Z, ancestor branch)**.** *Given a partitioning process Z, we use the notation $Z_{d,k}$, where $d = 1, 2, \dots$ and $k = 1, \dots, 2^{d-1}$ to denote the k^{th} node at depth d, that is*

$$Z_{d,k} := Z_{2^{d-1}-1+k}. \tag{28}$$

We use the hat notation $\hat{Z}_{d,k}$ to denote the sequence of nodes consisting of $Z_{d,k}$ and all its ancestors

$$\hat{Z}_{d,k} := (Z_n : n \in A(\{2^{d-1} - 1 + k\})), \tag{29}$$

and call $\hat{Z}_{d,k}$ the ancestor branch of $Z_{d,k}$.

Notation 2 ($\mathbb{P}$ measure)**.** *In definition 5, we defined $\mathbb{P}$ to be the measure associated with an infinite sequence of independent fair coin tosses over a measurable space $(\Omega, \mathcal{S})$. To avoid heavy notation, for the rest of the proof we will overload this symbol as follows: if F is a random variable from Ω to some measurable space, we will abbreviate $\mathbb{P} \circ F^{-1}$ by simply $\mathbb{P}(F)$.*

B.2 Deriving the measure of samples returned by GRC

For the remainder of the proof, we condition on a fixed partitioning process sample Z. For brevity, we omit this conditioning which, from here on is understood to be implied. Proposition 3 shows that the measures A_d and T_d correspond to the probabilities that GRC picks a particular branch of the binary tree and terminates at step d, or does not terminate up to and including step d, respectively.

Proposition 3 (Acceptance and rejection probabilities)**.** *Let V_d be the event that GRC does not terminate up to and including step d and W_d be the event that it terminates at step d. Let $S_{0:d} = B_{0:d}$ denote the event that the sequence of the first d bounds produced is $B_{0:d}$. Then*

$$\mathbb{P}(V_d, S_{0:d} = B_{0:d}) = 1 - T_d(\mathcal{X}, B_{0:d}), \qquad \text{for } d = 0, 1, \dots, \tag{30}$$

$$\mathbb{P}(W_{d+1}, S_{0:d} = B_{0:d}) = A_{d+1}(\mathcal{X}, B_{0:d}), \qquad \text{for } d = 0, 1, \dots. \tag{31}$$

Proof. First we consider the probability that GRC terminates at step $k + 1$ given that it has not terminated up to and including step d, that is the quantity $\mathbb{P}(W_{k+1} \mid V_k, S_{0:k} = B_{0:k})$. By definition 5, this probability is given by integrating the acceptance probability $\beta_{k+1}(x, B_{0:k})$ over $x \in \mathcal{X}$, with respect to the measure $P|_{B_k}/P(B_k)$, that is

$$\mathbb{P}(W_{k+1} \mid V_k, S_{0:k} = B_{0:k}) = \int_{x \in B_k} dP(x) \frac{\beta_{k+1}(x, B_{0:k})}{P(B_k)} \tag{32}$$

$$= \int_{x \in \mathcal{X}} dP(x) \frac{\beta_{k+1}(x, B_{0:k})}{P(B_k)} \tag{33}$$

$$= \int_{x \in \mathcal{X}} dP(x) \frac{\alpha_{k+1}(x, B_{0:k})}{1 - T_k(\mathcal{X}, B_{0:k})} \tag{34}$$

$$= \frac{A_{k+1}(\mathcal{X}, B_{0:k})}{1 - T_k(\mathcal{X}, B_{0:k})}, \tag{35}$$

Now, we show the result by induction on d, starting from the base case of $d = 0$.

Base case: For $d = 0$, by the definition of GRC (definition 5) $S_0 = Z_{I_0} = \mathcal{X}$, so

$$\mathbb{P}\left(V_0, S_0 = B_0\right) = 1 \;\text{ and }\; T_0(\mathcal{X}, B_0) = 0, \tag{36}$$

which show the base case for eq. (30). Now, plugging in $k = 0$ in eq. (35) we obtain

$$\mathbb{P}(W_1, S_0 = B_0) = \mathbb{P}(W_1 \mid V_0, S_0 = B_0) = \frac{A_1(\mathcal{X}, B_0)}{1 - T_0(\mathcal{X}, B_0)} = A_1(\mathcal{X}, B_0) \tag{37}$$

where we have used the fact that $T_0(\mathcal{X}, B_0) = 0$, showing the base case for eq. (31).

Inductive step: Suppose that for all $k = 0, 1, 2, \dots, d$ it holds that

$$\mathbb{P}\left(V_d, S_{0:k} = B_{0:k}\right) = 1 - T_d(\mathcal{X}, B_{0:k}) \quad \text{and} \quad \mathbb{P}\left(W_{k+1}, S_{0:k} = B_{0:k}\right) = A_{k+1}(\mathcal{X}, B_{0:k}). \tag{38}$$

Setting $k = d$ in eq. (35), we obtain

$$\mathbb{P}(W'_{d+1} \mid V_d, S_{0:d} = B_{0:d}) = \frac{1 - T_d(\mathcal{X}, B_{0:d}) - A_{d+1}(\mathcal{X}, B_{0:d})}{1 - T_d(\mathcal{X}.B_{0:d})}, \tag{39}$$

and using the inductive hypothesis from eq. (38), we have

$$\mathbb{P}(V_{d+1}, S_{0:d} = B_{0:d}) = \mathbb{P}(W'_{d+1}, V_d, S_{0:d} = B_{0:d}) = 1 - T_d(\mathcal{X}, B_{0:d}) - A_{d+1}(\mathcal{X}, B_{0:d}). \tag{40}$$

Now, $B_d = Z_n$ for some $n \in \mathbb{N}^+$. Denote $B_d^L := Z_{2n}$ and $B_d^R := Z_{2n+1}$. Then, by the product rule

$$\mathbb{P}(V_{d+1}, S_{0:d} = B_{0:d}, S_{d+1} = B_d^R) = \tag{41}$$

$$= \mathbb{P}(S_{d+1} = B_d^R \mid V_{d+1}, S_{0:d} = B_{0:d})\mathbb{P}(V_{d+1}, S_{0:d} = B_{0:d}) \tag{42}$$

$$= \frac{Q(B_d^R) - T_d(B_d^R, B_{0:d}) - A_{d+1}(B_d^R, B_{0:d})}{Q(B_d) - T_d(B_d, B_{0:d}) - A_{d+1}(B_d, B_{0:d})}\mathbb{P}(V_{d+1}, S_{0:d} = B_{0:d}) \tag{43}$$

$$= \frac{Q(B_d^R) - T_d(B_d^R, B_{0:d}) - A_{d+1}(B_d^R, B_{0:d})}{\underbrace{Q(\mathcal{X})}_{=1} - T_d(\mathcal{X}, B_{0:d}) - A_{d+1}(\mathcal{X}, B_{0:d})}\mathbb{P}(V_{d+1}, B_{0:d} = B_{0:d}) \tag{44}$$

$$= Q(B_d^R) - T_d(B_d^R, B_{0:d}) - A_{d+1}(B_d^R, B_{0:d}) \tag{45}$$

$$= 1 - T_{d+1}(\mathcal{X}, B_{0:d+1}) \tag{46}$$

where we have written $B_{0:d+1} = (B_0, \ldots, B_d, B_d^R)$. Above, to go from 41 to 42 we used the definition of conditional probability, to go from 42 to 43 we used the definition in 19, to go from 43 to 44 we used the fact that for $k = 0, 1, 2, \ldots$, it holds that

$$\begin{aligned} Q(\mathcal{X}) - T_k(\mathcal{X}, B_{0:k}) - A_{k+1}(\mathcal{X}, B_{0:k}) &= Q(B_k) - T_k(B_k, B_{0:k}) - A_{k+1}(B_k, B_{0:k}) + \\ &\quad + Q(B'_k) - \underbrace{T_k(B'_k, B_{0:k})}_{= Q(B'_k)} - \underbrace{A_{k+1}(B'_k, B_{0:k})}_{= 0} \qquad (47) \\ &= Q(B_k) - T_d(B_k, B_{0:k}) - A_{k+1}(B_k, B_{0:k}), \qquad (48) \end{aligned}$$

from 44 to 45 we have used eq. (40), and lastly from 45 to 46 we have again used eq. (48). Equation (46) similarly holds if $B_{d+1} = B_d^R$ by $B_{d+1} = B_d^L$, so we arrive at

$$\mathbb{P}(V_{d+1}, B_{0:d+1} = B_{0:d+1}) = 1 - T_{d+1}(\mathcal{X}, B_{0:d+1}), \tag{49}$$

which shows the inductive step for eq. (30). Further, we have

$$\mathbb{P}(W_{d+2}, B_{0:d+1} = B_{0:d+1}) = \mathbb{P}(W_{d+2} \mid V_{d+1}, B_{0:d+1} = B_{0:d+1})\mathbb{P}(V_{d+1}, B_{0:d+1} = B_{0:d+1}) \tag{50}$$

and also by setting $k = d + 1$ in eq. (35) we have

$$\mathbb{P}(W_{d+2} \mid V_{d+1}, B_{0:d+1} = B_{0:d+1}) = \frac{A_{d+2}(\mathcal{X}, B_{0:d+1})}{1 - T_{d+1}(\mathcal{X}, B_{0:d+1})}. \tag{51}$$

Combining eq. (49) and eq. (51) we arrive at

$$\mathbb{P}(W_{d+2}, B_{0:d+1} = B_{0:d+1}) = A_{d+2}(\mathcal{X}, B_{0:d+1}), \tag{52}$$

which is the inductive step for eq. (31). Putting eqs. (49) and (52) together shows the result. □

We now turn to defining and deriving the form of the measure τ_D. We will define τ_D to be the measure such that for any $S \in \Sigma$, the probability that GRC terminates up to and including step D and returns a sample within S is given by $\tau_D(S)$. We will also show that τ_D is non-increasing in D.

Lemma 1 (Density of samples generated by GRC)**.** *The probability that GRC terminates by step $D \geq 1$ and produces a sample in S is given by the measure*

$$\tau_D(S) = \sum_{d=1}^{D} \sum_{k=1}^{2^{d-1}} A_d(S, \hat{Z}_{d,k}), \tag{53}$$

where $\hat{Z}_{D,k}$ is the ancestor branch of $Z_{D,k}$ as defined in eq. (29)*. Further, τ_D is non-decreasing in D, that is if $n \leq m$, then $\tau_n(S) \leq \tau_m(S)$ for all $S \in \Sigma$.*

Proof. Let V_d be the event that GRC does not terminate up to and including step d and let $W_d(S)$ be the event that GRC terminates at step d and returns a sample in S. Then

$$\begin{aligned}
\tau_D(S) &= \sum_{d=1}^{D} \mathbb{P}(W_d(S)) && (54)\\
&= \sum_{d=1}^{D} \mathbb{P}(W_d(S), V_{d-1}) && (55)\\
&= \sum_{d=1}^{D} \sum_{k=1}^{2^{d-1}} \mathbb{P}(W_d(S), V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k}) && (56)\\
&= \sum_{d=1}^{D} \sum_{k=1}^{2^{d-1}} \mathbb{P}(W_d(S) \mid V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k})\, \mathbb{P}(V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k}). && (57)
\end{aligned}$$

Further, the terms in the summand can be expressed as

$$\begin{aligned}
\mathbb{P}(V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k}) &= 1 - T_{d-1}(\mathcal{X}, \hat{Z}_{d,k}), && (58)\\
\mathbb{P}(W_d(S) \mid V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k}) &= \int_{x \in S} dP(x) \frac{\beta_d(x, \hat{Z}_{d,k})}{P(Z_{d,k})} && (59)\\
&= \int_{x \in S} dP(x) \frac{\alpha_d(x, \hat{Z}_{d,k})}{1 - T_{d-1}(\mathcal{X}, \hat{Z}_{d,k})} && (60)\\
&= \frac{A_d(S, \hat{Z}_{d,k})}{1 - T_{d-1}(\mathcal{X}, \hat{Z}_{d,k})}, && (61)
\end{aligned}$$

and substituting eqs. (58) and (61) into the sum in eq. (57), we obtain eq. (53). Further, since the inner summand is always non-negative, increasing D adds more non-negative terms to the sum, so τ_D is also non-decreasing in D. □

Now we turn to proving a few results about the measure τ_D. Lemma 2 shows that $\tau_D \leq Q$ for all D. This result implies that $||Q - \tau_D||_{TV} = Q(\mathcal{X}) - \tau_D(\mathcal{X})$, which we will use later.

Lemma 2 ($Q - \tau_D$ is non-negative)**.** *Let $D \in \mathbb{N}^+$. Then $Q - \tau_D$ is a positive measure, that is*

$$Q(S) - \tau_D(S) \geq 0 \text{ for any } S \in \Sigma. \tag{62}$$

Proof. Let $S \in \Sigma$ and write

$$\begin{aligned}
Q(S) - \tau_D(S) &= \sum_{k=1}^{2^{D-1}} Q(S \cap Z_{D,k}) - \tau_D(S \cap Z_{D,k}) && (63)\\
&= \sum_{k=1}^{2^{D-1}} \left[Q(S \cap Z_{D,k}) - \sum_{d=1}^{D} \sum_{k'=1}^{2^{D-1}} A_d(S \cap Z_{D,k}, \hat{Z}_{D,k'}) \right] && (64)\\
&= \sum_{k=1}^{2^{D-1}} \left[Q(S \cap Z_{D,k}) - \sum_{d=1}^{D} A_d(S \cap Z_{D,k}, \hat{Z}_{D,k}) \right] && (65)\\
&= \sum_{k=1}^{2^{D-1}} \left[Q(S \cap Z_{D,k}) - T_{D-1}(S \cap Z_{D,k}, \hat{Z}_{D,k}) - A_D(S \cap Z_{D,k}, \hat{Z}_{D,k}) \right] && (66)
\end{aligned}$$

We will show that the summand in eq. (66) is non-negative. From the definition in eq. (14) we have

$$\alpha_D(x, \hat{Z}_{D,k}) = \min\left\{\frac{dQ}{dP}(x) - t_{D-1}(x, \hat{Z}_{D,k}), \frac{1 - T_{D-1}(\mathcal{X}, \hat{Z}_{D,k})}{P(Z_{D,k})}\right\} \tag{67}$$

$$\leq \frac{dQ}{dP}(x) - t_{D-1}(x, \hat{Z}_{D,k}) \tag{68}$$

and integrating both sides of eq. (68) over $S \cap Z_{D,k}$, we obtain

$$A_D(S \cap Z_{D,k}, \hat{Z}_{D,k}) \leq Q(S \cap Z_{D,k}) - T_{D-1}(S \cap Z_{D,k}, \hat{Z}_{D,k}) \tag{69}$$

Putting this together with eq. (66) we arrive at

$$Q(S) - \tau_D(S) \geq 0, \tag{70}$$

which is the required result. □

Thus far we have derived the form of τ_D, shown that it is non-decreasing in D and that it is no greater than Q. As we are interested in the limiting behaviour of τ_D, we next show that its limit, $\tau = \lim_{D\to\infty} \tau_D$, is also a measure. Further, it also holds that $\tau \leq Q$.

Lemma 3 (Measures τ_D converge to a measure $\tau \leq Q$)**.** *For each $S \in \Sigma$, $\tau_D(S)$ converges to a limit. Further, the function $\tau : \Sigma \to [0, 1]$ defined as*

$$\tau(S) = \lim_{D\to\infty} \tau_D(S) \tag{71}$$

is a measure on $(\mathcal{X}, \Sigma)$ and $\tau(S) \leq Q(S)$ for all $S \in \Sigma$.

Proof. First, by lemma 1, $\tau_D(S)$ is non-decreasing in D, and bounded above by $Q(S)$ for all $S \in \Sigma$. Therefore, for each $S \in \Sigma$, $\tau_D(S)$ converges to some limit as $D \to \infty$. Define $\tau : \Sigma \to [0, 1]$ as

$$\tau(S) = \lim_{D\to\infty} \tau_D(S), \tag{72}$$

and note that τ is a non-negative set function for which $\tau(\emptyset) = 0$. By the Vitali-Hahn-Saks theorem (see Corollary 4, p. 160; Dunford & Schwartz, 1988), τ is also countably additive, so it is a measure. Also, by lemma 2, $\tau_D(S) \leq Q(S)$ for all $D \in \mathbb{N}^+$ and all $S \in \Sigma$, so $\tau(S) \leq Q(S)$ for all $S \in \Sigma$. □

Definition 9 ($H_{d,k}$, H_d and H)**.** *For $d = 1, 2, \ldots$ and $k = 1, \ldots, 2^{d-1}$, we define the sets $H_{d,k}$ as*

$$H_{d,k} = \left\{x \in Z_{d,k} \;\middle|\; \frac{dQ}{dP}(x) - t_{d-1}(x, \hat{Z}_{d,k}) \geq \frac{1 - T_{d-1}(\mathcal{X}, \hat{Z}_{d,k})}{P(Z_{d,k})}\right\}. \tag{73}$$

Also, define the sets H_d and H as

$$H_d = \bigcup_{k=1}^{2^{d-1}} H_{d,k} \;\; \textit{and} \;\; H = \bigcap_{d=1}^{\infty} H_d. \tag{74}$$

Lemma 4 ($T_D(\cdot, \hat{Z}_{D+1,k})$ and τ_D agree in $Z_{D+1,k}$)**.** *Let $R \in \Sigma$. If $R \subseteq Z_{D+1,k}$, then*

$$\tau_D(R) = T_D(R, \hat{Z}_{D+1,k}). \tag{75}$$

Proof. Suppose $R \subseteq Z_{D+1,k}$. First, we have

$$\tau_D(R) = \sum_{d=1}^{D} \sum_{k'=1}^{2^{d-1}} A_d(R, \hat{Z}_{d,k'}) = \sum_{d=1}^{D} A_d(R, (\hat{Z}_{D+1,k})_{1:d}). \tag{76}$$

From the definition of T_D in eq. (22), we have

$$T_D(R, \hat{Z}_{D+1,k}) = T_{D-1}(R \cap Z_{D+1,k}, (\hat{Z}_{D+1,k})_{1:D}) + A_D(R \cap Z_{D+1,k}, (\hat{Z}_{D+1,k})_{1:D}) + \tag{77}$$

$$+ \underbrace{Q(R \cap Z'_{D+1,k})}_{=0}$$

$$= T_{D-1}(R \cap Z_{D+1,k}, (\hat{Z}_{D+1,k})_{1:D}) + A_D(R \cap Z_{D+1,k}, (\hat{Z}_{D+1,k})_{1:D}) \tag{78}$$

$$= T_{D-1}(R, (\hat{Z}_{D+1,k})_{1:D}) + A_D(R, (\hat{Z}_{D+1,k})_{1:D}) \tag{79}$$

where we have used the assumption that $R \subseteq Z_{D+1,k}$. In a similar manner, applying eq. (79) recursively $D-1$ more times, we obtain

$$T_D(R, \hat{Z}_{D+1,k}) = \sum_{d=1}^{D} A_d(R, (\hat{Z}_{D+1,k})_{1:d}) = \tau_D(R). \tag{80}$$

which is the required result. □

Lemma 5 (Equalities with Q, τ_D and τ)**.** *The following two equalities hold*

$$Q(\mathcal{X} \setminus H_D) = \tau_D(\mathcal{X} \setminus H_D) \;\; \textit{and} \;\; Q(\mathcal{X} \setminus H) = \tau(\mathcal{X} \setminus H). \tag{81}$$

Proof. Let $R = Z_{D+1,k} \setminus H_{D,k}$. Then, by similar reasoning used to prove eq. (77), we have

$$T_D(R, \hat{Z}_{D+1,k}) = T_{D-1}(R, (\hat{Z}_{D+1,k})_{1:D}) + A_D(R, (\hat{Z}_{D+1,k})_{1:D}) \tag{82}$$

Further, we also have

$$\begin{aligned}
A_D(R, \hat{Z}_{D,k}) &= \int_R dP(x)\, \alpha_D(x, \hat{Z}_{D,k}) && (83)\\
&= \int_R dP(x)\; \min\left\{ \frac{dQ}{dP}(x) - t_{D-1}(x, \hat{Z}_{D,k}), \frac{1 - T_{D-1}(\mathcal{X}, \hat{Z}_{D,k})}{P(Z_{D,k})} \right\} && (84)\\
&= \int_R dP(x)\; \left(\frac{dQ}{dP}(x) - t_{D-1}(x, \hat{Z}_{D,k}) \right) && (85)\\
&= Q(R) - T_{D-1}(R, \hat{Z}_{D,k}) && (86)
\end{aligned}$$

where from eq. (84) to eq. (85) we have used the definition of $H_{D,k}$. Then, combining eqs. (82) and (86) and using lemma 4, we arrive at

$$Q(Z_{D+1,k} \setminus H_{D,k}) = T_D(Z_{D+1,k} \setminus H_{D,k}, \hat{Z}_{D+1,k}) = \tau_D(Z_{D+1,k} \setminus H_{D,k}). \tag{87}$$

Now, using the equation above, we have that

$$\tau_D(\mathcal{X} \setminus H_D) = \sum_{k=1}^{2^D} \tau_D(Z_{D+1,k} \setminus H_D) = \sum_{k=1}^{2^D} Q(Z_{D+1,k} \setminus H_D) = Q(\mathcal{X} \setminus H_D). \tag{88}$$

Now, using $\tau_D \leq \tau \leq Q$ and $\tau_D(\mathcal{X} \setminus H_D) = Q(\mathcal{X} \setminus H_D)$, we have that $\tau(\mathcal{X} \setminus H_D) = Q(\mathcal{X} \setminus H_D)$, which is the first part of the result we wanted to show. Taking limits, we obtain

$$Q(\mathcal{X} \setminus H) = \lim_{D \to \infty} Q(\mathcal{X} \setminus H_D) = \lim_{D \to \infty} \tau(\mathcal{X} \setminus H_D) = \tau(\mathcal{X} \setminus H), \tag{89}$$

which is the second part of the required result. □

B.3 Breaking down the proof of Theorem 1 in five cases

In definition 10 we introduce the quantities $w_d = Q(\mathcal{X}) - \tau_d(\mathcal{X})$ and $p_d = \mathbb{P}(W_d \mid V_{d-1})$. Then we break down the proof of theorem 1 in five cases. First, in lemma 7 we show that if $p_d \not\to 0$, then $w_d \to 0$. Second, in lemma 8 we show that if $P(H_d) \to 0$, then $w_d \to 0$. In lemma 9 we show an intermediate result, used in the other three cases, which we consider in lemmas 10, 11 and 12. Specifically, in these three cases we show that if $p_d \to 0$ and $P(H_d) \not\to 0$, and assumption 1, 2 or 3 hold respectively, we have $w_d \to 0$. Putting these results together shows theorem 1.

Definition 10 (p_d, $w_{d,k}$ and w_d)**.** *Define $p_d = \mathbb{P}(W_d \mid V_{d-1})$. Also define $w_{d,k}$ and w_d as*

$$w_{d,k} \stackrel{\text{def}}{=} Q(Z_{d,k}) - \tau_d(Z_{d,k}), \tag{90}$$

$$w_d \stackrel{\text{def}}{=} \sum_{k=1}^{2^{d-1}} w_{d,k}. \tag{91}$$

Lemma 6 (w_d non-increasing in d)**.** *The sequence w_d is non-negative and non-increasing in d.*

Proof. Since τ_d is non-decreasing in d (from lemma 5) and

$$w_d = \sum_{k=1}^{2^{d-1}} Q(Z_{d,k}) - \tau_d(Z_{d,k}) = Q(\mathcal{X}) - \tau_d(\mathcal{X}), \tag{92}$$

it follows that w_d is a non-increasing and non-negative sequence. □

Lemma 7 (Case 1)**.** *If $p_d \not\to 0$, then $w_d \to 0$.*

Proof. Let $p_d = \mathbb{P}(W_d \mid V_{d-1})$ and suppose $p_d \not\to 0$. Then, there exists $\epsilon > 0$ such that $p_d > \epsilon$ occurs infinitely often. Therefore, there exists an increasing sequence of integers $a_d \in \mathbb{N}$ such that $p_{a_d} > \epsilon$ for all $d \in \mathbb{N}$. Then

$$\tau_{a_d}(\mathcal{X}) = \mathbb{P}\left(\bigcup_{d=1}^{a_d} W_d\right) \tag{93}$$

$$= 1 - \mathbb{P}\left(V_{a_d}\right), \tag{94}$$

$$= 1 - \prod_{d=1}^{a_d} \mathbb{P}\left(V_d \mid V_{d-1}\right), \tag{95}$$

$$= 1 - \prod_{d=1}^{a_d} (1 - p_d), \tag{96}$$

$$\geq 1 - (1-\epsilon)^d \to 1 \text{ as } d \to \infty. \tag{97}$$

Therefore, $\tau_d(\mathcal{X}) \to 1$ as $d \to \infty$, which implies that $||Q - \tau_d||_{TV} \to 0$. □

Lemma 8 (Case 2)**.** *If $P(H_d) \to 0$, then $w_d \to 0$.*

Proof. Suppose $P(H_d) \to 0$. Since $Q \ll P$, we have $Q(H) = 0$, and since $Q \geq \tau \geq 0$ (by lemma 3), we also have $\tau(H) = 0$. Therefore

$$\lim_{d\to\infty} w_d = \lim_{d\to\infty} ||Q - \tau_d||_{TV} \tag{98}$$

$$= Q(\mathcal{X}) - \tau(\mathcal{X}) \tag{99}$$

$$= \underbrace{Q(\mathcal{X} \setminus H) - \tau(\mathcal{X} \setminus H)}_{=0 \text{ from lemma 5}} + \underbrace{Q(H)}_{=0} - \underbrace{\tau(H)}_{=0} \tag{100}$$

$$= 0 \tag{101}$$

which is the required result. □

Lemma 9 (An intermediate result)**.** *If $p_d \to 0$ and $w_d \not\to 0$ as $d \to \infty$, then*

$$\sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})}\, w_{d,k} \to 0 \text{ as } d \to \infty. \tag{102}$$

Proof. Suppose that $p_d = \mathbb{P}(W_d \mid V_{d-1}) \to 0$ and $w_d \not\to 0$. Then

$$\begin{aligned}
\mathbb{P}(W_d \mid V_{d-1}) &\geq \mathbb{P}(W_d(H_d) \mid V_{d-1}) && (103)\\
&= \sum_{k=1}^{2^{d-1}} \mathbb{P}\left(W_d(H_{d,k}) \mid V_{d-1}\right) && (104)\\
&= \sum_{k=1}^{2^{d-1}} \mathbb{P}\left(W_d(H_{d,k}), S_{0:d-1} = \hat{Z}_{d,k} \mid V_{d-1}\right) && (105)\\
&= \sum_{k=1}^{2^{d-1}} \mathbb{P}\left(W_d(H_{d,k}) \mid V_{d-1}, S_{0:d-1} = \hat{Z}_{d,k}\right) \mathbb{P}\left(S_{0:d-1} = \hat{Z}_{d,k} \mid V_{d-1}\right) && (106)\\
&= \sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})} \mathbb{P}\left(S_{0:d-1} = \hat{Z}_{d,k} \mid V_{d-1}\right) && (107)\\
&= \sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})} \frac{w_{d,k}}{w_d} \to 0. && (108)
\end{aligned}$$

In addition, if $w_d \not\to 0$, then since $0 \leq w_d \leq 1$ we have

$$\sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})} w_{d,k} \to 0. \tag{109}$$

which is the required result. □

Lemma 10 (Case 3)**.** *Suppose that $p_d \to 0$, $P(H_d) \not\to 0$ and assumption 1 holds. Then $w_d \to 0$.*

Proof. Suppose that $p_d \to 0$, $P(H_d) \not\to 0$. Suppose also that assumption 1 holds, meaning there exists $M \in \mathbb{R}$ such that $dQ/dP(x) < M$ for all $x \in \mathcal{X}$. Then for any $S \in \Sigma$, we have

$$\frac{Q(S) - \tau(S)}{P(S)} \leq \frac{Q(S)}{P(S)} = \frac{\int_S \frac{dQ}{dP} dP}{P(S)} \leq M\, \frac{\int_S dP}{P(S)} = M \implies \frac{Q(S) - \tau(S)}{M} \leq P(S). \tag{110}$$

Further, we have

$$\begin{aligned}
\sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})}\, w_{d,k} &\geq \sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})} \left(Q(H_{d,k}) - \tau(H_{d,k})\right) && (111)\\
&\geq \frac{1}{M} \sum_{k=1}^{2^{d-1}} \frac{\left(Q(H_{d,k}) - \tau(H_{d,k})\right)^2}{P(Z_{d,k})} && (112)\\
&\geq \frac{1}{M} \sum_{k=1}^{2^{d-1}} \frac{\left(Q(H \cap H_{d,k}) - \tau(H \cap H_{d,k})\right)^2}{P(Z_{d,k})} && (113)\\
&\geq \frac{1}{M} \sum_{k=1}^{2^{d-1}} \frac{\Delta_{d,k}^2}{P(Z_{d,k})} && (114)\\
&= \frac{1}{M}\, \Phi_d && (115)\\
&\to 0, && (116)
\end{aligned}$$

where in the second inequality we have used eq. (110) and we have defined

$$\Delta_{d,k} \stackrel{\text{def}}{=} Q(H \cap H_{d,k}) - \tau(H \cap H_{d,k}), \tag{117}$$

$$\Phi_d \stackrel{\text{def}}{=} \sum_{k=1}^{2^{d-1}} \frac{\Delta_{d,k}^2}{P(Z_{d,k})}. \tag{118}$$

Now note that the sets $H \cap H_{d+1,2k}$ and $H \cap H_{d+1,2k+1}$ partition the set $H \cap H_{d,k}$. Therefore

$$\Delta_{d,k} = \Delta_{d+1,2k} + \Delta_{d+1,2k+1}. \tag{119}$$

By the definition of Φ_d in eq. (118), we can write

$$\Phi_{d+1} = \sum_{k=1}^{2^d} \frac{\Delta_{d,k}^2}{P(Z_{d+1,k})} = \sum_{k=1}^{2^{d-1}} \left[\frac{\Delta_{d+1,2k}^2}{P(Z_{d+1,2k})} + \frac{\Delta_{d+1,2k+1}^2}{P(Z_{d+1,2k+1})} \right], \tag{120}$$

where we have written the sum over 2^d terms as a sum over 2^{d-1} pairs of terms. We can rewrite the summand on the right hand side as

$$\frac{\Delta_{d+1,2k}^2}{P(Z_{d+1,2k})} + \frac{\Delta_{d+1,2k+1}^2}{P(Z_{d+1,2k+1})} = \frac{\Delta_{d+1,2k}^2}{P(Z_{d+1,2k})} + \frac{(\Delta_{d,k} - \Delta_{d+1,2k})^2}{P(Z_{d+1,2k+1})} \tag{121}$$

$$= \Delta_{d,k}^2 \left[\frac{\rho^2}{P(Z_{d+1,2k-1})} + \frac{(1-\rho)^2}{P(Z_{d+1,2k})} \right] \tag{122}$$

$$= \Delta_{d,k}^2 \, g(\rho) \tag{123}$$

where in eq. (121) we have used eq. (119), from eq. (121) to eq. (122) we defined the quantity $\rho = \Delta_{d+1,2k}/\Delta_{d,k}$, and from eq. (122) to eq. (123) we have defined $g : [0,1] \to \mathbb{R}$ as

$$g(r) \stackrel{\text{def}}{=} \frac{r^2}{P(Z_{d+1,2k})} + \frac{(1-r)^2}{P(Z_{d+1,2k+1})}. \tag{124}$$

The first and second derivatives of g are

$$\frac{dg}{dr} = \frac{2r}{P(Z_{d+1,2k})} - \frac{2(1-r)}{P(Z_{d+1,2k+1})}, \tag{125}$$

$$\frac{d^2 g}{dr^2} = \frac{2}{P(Z_{d+1,2k})} + \frac{2}{P(Z_{d+1,2k+1})} > 0, \tag{126}$$

so g has a single stationary point that is a minimum, at $r = r_{\min}$, which is given by

$$r_{\min} := \frac{P(Z_{d+1,2k})}{P(Z_{d+1,2k}) + P(Z_{d+1,2k+1})}. \tag{127}$$

Plugging this back in g, we obtain

$$g(r_{\min}) = \frac{1}{P(Z_{d+1,2k}) + P(Z_{d+1,2k+1})} = \frac{1}{P(Z_{d,k})}, \tag{128}$$

which implies that

$$\frac{\Delta_{d+1,2k}^2}{P(Z_{d+1,2k})} + \frac{\Delta_{d+1,2k+1}^2}{P(Z_{d+1,2k+1})} \geq \frac{\Delta_{d,k}^2}{P(Z_{d,k})}. \tag{129}$$

Therefore

$$\Phi_{d+1} = \sum_{k=1}^{2^d} \frac{\Delta_{d,k}^2}{P(Z_{d+1,k})} \geq \sum_{k=1}^{2^{d-1}} \frac{\Delta_{d,k}^2}{P(Z_{d,k})} = \Phi_d, \tag{130}$$

but since $\Phi_d \to 0$, this is only possible if $\Phi_d = 0$ for all d, including $d = 1$, which would imply that

$$\Delta_{1,1} = Q(H \cap H_{1,1}) - \tau(H \cap H_{1,1}) = Q(H) - \tau(H) = 0, \tag{131}$$

which, together with lemma 5, implies that

$$Q(\mathcal{X}) - \tau(\mathcal{X}) = Q(H) - \tau(H) = 0, \tag{132}$$

and therefore $w_d = ||Q - \tau_d||_{TV} \to 0$. □

Lemma 11 (Case 4)**.** *Suppose that $p_d \to 0$, $P(H_d) \not\to 0$ and assumption 3 holds. Then $w_d \to 0$.*

Proof. Suppose that $p_d \to 0$, $P(H_d) \not\to 0$. Suppose also that assumption that assumption 3 holds, meaning that for each d, we have $w_{d,k} > 0$ for exactly one value of $k = k_d$, and $w_{d,k} = 0$ for all other $k \neq k_d$. In this case, it holds that $H_{d,k} = \emptyset$ for all $k \neq k_d$ and $H_d = H_{d,k_d}$. Since $P(H_d) \not\to 0$ and $P(H_d)$ is a decreasing sequence, it converges to some positive constant. We also have

$$p_d \geq \sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})}\, w_{d,k} = \frac{P(H_{d,k_d})}{P(Z_{d,k_d})}\, w_{d,k_d} = \frac{P(H_{d,k_d})}{P(Z_{d,k_d})}\, w_d \geq P(H_d)\, w_d \to 0, \tag{133}$$

which can only hold if $w_d \to 0$, arriving at the result. □

Lemma 12 (Case 5)**.** *Suppose that $p_d \to 0$, $P(H_d) \not\to 0$ and assumption 3 holds. Then $w_d \to 0$.*

Proof. Suppose that $p_d \to 0$, $P(H_d) \not\to 0$ and assumption 3 holds. Since each $x \in \mathcal{X}$ belongs to exactly one $Z_{d,k}$ we can define the function $B_d : \mathcal{X} \to \Sigma$ as

$$B_d(x) = Z_{d,k} \text{ such that } x \in Z_{d,k}. \tag{134}$$

Using this function we can write

$$p_d \geq \sum_{k=1}^{2^{d-1}} \frac{P(H_{d,k})}{P(Z_{d,k})}\, w_{d,k} = \sum_{k=1}^{2^{d-1}} P(H_{d,k})\, \frac{Q(Z_{d,k}) - \tau_d(Z_{d,k})}{P(Z_{d,k})} = \int_{H_d} dP\, \frac{Q(B_d(x)) - \tau_d(B_d(x))}{P(B_d(x))}.$$

Now, because the sets H_d are measurable, their intersection $H := \cap_{d=1}^{\infty} H_d$ is also measurable. We can therefore lower bound the integral above as follows

$$\int_{H_d} dP\, \frac{Q(B_d(x)) - \tau_d(B_d(x))}{P(B_d(x))} \geq \int_{H} dP\, \frac{Q(B_d(x)) - \tau_d(B_d(x))}{P(B_d(x))} \tag{135}$$

$$\geq \int_{H} dP\, \frac{Q(B_d(x)) - \tau(B_d(x))}{P(B_d(x))}, \tag{136}$$

where the first inequality holds as the integrand is non-negative and we are constraining the integration domain to $H \subseteq H_d$, and the second inequality holds because $\tau_d(S) \leq \tau(S)$ for any $S \in \Sigma$. Define $\mathcal{C}$ to be the set of all intersections of nested partitions, with non-zero mass under P

$$\mathcal{C} = \left\{ \bigcap_{d=0}^{\infty} Z_{d,k_d} : P\left(\bigcap_{d=0}^{\infty} Z_{d,k_d} \right) > 0, k_0 = 1, k_{d+1} = 2k_d \text{ or } k_{d+1} = 2k_d + 1 \right\}, \tag{137}$$

and note that all of its elements are pairwise disjoint. Each of the elements of $\mathcal{C}$ is a measurable set because it is a countable intersection of measurable sets. In addition, $\mathcal{C}$ is a countable set, which can be shown as follows. Define the sets $\mathcal{C}_n$ as

$$\mathcal{C}_n = \left\{ E \in \mathcal{C} : 2^{-n-1} < P(E) \leq 2^{-n} \right\} \text{ for } n = 0, 1, \ldots \tag{138}$$

and note that their union equals $\mathcal{C}$. Further, note that each $\mathcal{C}_n$ must contain a finite number of elements. That is because if $\mathcal{C}_n$ contained an infinite number of elements, say $E_1, E_2, \cdots \in \mathcal{C}_n$, then

$$P(\mathcal{X}) \geq P\left(\bigcup_{k=1}^{\infty} E_k \right) = \sum_{k=1}^{\infty} P(E_k) > \sum_{k=1}^{\infty} 2^{-n-1} \to \infty, \tag{139}$$

where the first equality holds because P is an additive measure and the E_n terms are disjoint, and the second inequality follows because $E_k \in \mathcal{C}_n$ so $P(E_k) > 2^{-n-1}$. This results in a contradiction because $P(\mathcal{X}) = 1$, so each $\mathcal{C}_n$ must contain a finite number of terms. Therefore, $\mathcal{C}$ is a countable union of finite sets, which is also countable. This implies that the union of the elements of $\mathcal{C}$, namely $C = \cup_{C' \in \mathcal{C}} C'$ is a countable union of measurable sets and therefore also measurable. Since C is measurable, $H \setminus C$ is also measurable and we can rewrite the integral in eq. (135) as

$$p_d \geq \int_{H} dP\, \frac{Q(B_d(x)) - \tau(B_d(x))}{P(B_d(x))} \tag{140}$$

$$= \int_{H \cap C} dP\, \frac{Q(B_d(x)) - \tau(B_d(x))}{P(B_d(x))} + \int_{H \setminus C} dP\, \frac{Q(B_d(x)) - \tau(B_d(x))}{P(B_d(x))} \tag{141}$$

$$\to 0 \tag{142}$$

Since both terms above are non-negative and their sum converges to 0, the terms must also individually converge to 0. Therefore, for the first term, we can write

$$\lim_{d\to\infty}\int_{H\cap C} dP\,\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} = \liminf_{d\to\infty}\int_{H\cap C} dP\,\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} = 0. \quad (143)$$

Similarly to B_d defined in eq. (134), let us define $B : C \to \Sigma$ as

$$B(x) = C' \in \mathcal{C} \text{ such that } x \in C'. \quad (144)$$

Applying Fatou's lemma (4.3.3, p. 131; Dudley, 2018) to eq. (143), we obtain

$$\begin{aligned}\liminf_{d\to\infty}\int_{H\cap C} dP\,\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} &\geq \int_{H\cap C} dP\,\liminf_{d\to\infty}\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} &(145)\\ &= \int_{H\cap C} dP\,\frac{Q(B(x))-\tau(B(x))}{P(B(x))} &(146)\\ &= 0, &(147)\end{aligned}$$

where from eq. (145) to eq. (146) we have used the fact that $P(B_d(x)) > 0$ whenever $x \in C$ and also that $B_1(x) \supseteq B_2(x) \supseteq \ldots$. Now we can re-write this integral as a sum, as follows. Let the elements of $\mathcal{C}$, which we earlier showed is countable, be $C_1, C_2, \ldots$ and write

$$\begin{aligned}\int_{H\cap C} dP\,\frac{Q(B(x))-\tau(B(x))}{P(B(x))} &= \sum_{n=1}^{\infty}\int_{H\cap C_n} dP\,\frac{Q(B(x))-\tau(B(x))}{P(B(x))} &(148)\\ &= \sum_{n=1}^{\infty}\frac{P(H\cap C_n)}{P(C_n)}\left(Q(C_n)-\tau(C_n)\right) &(149)\\ &= 0. &(150)\end{aligned}$$

Now, from lemma 5, we have

$$\sum_{n=1}^{\infty}\frac{P(H\cap C_n)}{P(C_n)}\left(Q(C_n)-\tau(C_n)\right) = \sum_{n=1}^{\infty}\frac{P(H\cap C_n)}{P(C_n)}\left(Q(H\cap C_n)-\tau(H\cap C_n)\right) = 0, \quad (151)$$

which in turn implies that for each $n = 1, 2, \ldots$, we have either $Q(H \cap C_n) - \tau(H \cap C_n) = 0$ or $P(H \cap C_n) = 0$. However, the latter case also implies $Q(H \cap C_n) - \tau(H \cap C_n) = 0$ because $Q \ll P$, so $Q(H \cap C_n) - \tau(H \cap C_n) = 0$ holds for all n. Therefore

$$\tau(H\cap C) = \sum_{n=1}^{\infty}\tau(H\cap C_n) = \sum_{n=1}^{\infty}Q(H\cap C_n) = Q(H\cap C). \quad (152)$$

Returning to the second term in the right hand of eq. (141), and again applying Fatou's lemma

$$\liminf_{d\to\infty}\int_{H\setminus C} dP\,\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} \geq \int_{H\setminus C} dP\,\liminf_{d\to\infty}\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))}. \quad (153)$$

Now, since Z has the nice-shrinking property from assumption 3, we can apply a standard result from measure theory and integration Rudin (1986, given in Theorem 7.10, p. 140), to show that the following limit exists and the following equalities are satisfied

$$\begin{aligned}\lim_{d\to\infty}\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} &= \lim_{d\to\infty}\frac{1}{P(B_d(x))}\int_{B_d} dP\left(\frac{dQ}{dP}(x)-\frac{d\tau}{dP}(x)\right) &(154)\\ &= \frac{dQ}{dP}(x)-\frac{d\tau}{dP}(x) &(155)\end{aligned}$$

Inserting 155 into eq. (153), we obtain

$$\liminf_{d\to\infty}\int_{H\setminus C} dP\,\frac{Q(B_d(x))-\tau(B_d(x))}{P(B_d(x))} \geq \int_{H\setminus C} dP\,\left(\frac{dQ}{dP}(x)-\frac{d\tau}{dP}(x)\right) = 0, \quad (156)$$

which in turn implies that

$$\frac{dQ}{dP}(x)-\frac{d\tau}{dP}(x) = 0 \;\; P\text{-almost-everywhere on } H\setminus C, \quad (157)$$

or equivalently that $Q(H \setminus C) = \tau(H \setminus C)$. Combining this with the fact that $Q(\mathcal{X} \setminus H) = \tau(\mathcal{X} \setminus H)$ and our earlier result that $Q(H \cap C) = \tau(H \cap C)$, we have

$$||Q - \tau||_{TV} = Q(\mathcal{X} \setminus H) - \tau(\mathcal{X} \setminus H) + Q(H \setminus C) - \tau(H \setminus C) + Q(H \cap C) - \tau(H \cap C) = 0,$$

which is equivalent to $w_d = ||Q - \tau_d||_{TV} \to 0$, that is the required result. □

Theorem (Correcness of GRC)**.** *If any one of the assumptions 1, 2 or 3 holds, then*

$$||Q - \tau_d||_{TV} \to 0 \ \textit{as} \ \ d \to \infty. \tag{158}$$

Proof. If $p_d \to 0$, then $w_d \to 0$ by lemma 7. If $P(H_d) \to 0$, then $w_d \to 0$ by lemma 8. Therefore suppose that $p_d \not\to 0$ and $P(H_d) \not\to 0$. Then if any one of assumptions 1, 2 or 3 holds, we can conclude from lemma 10, 11 or 12 respectively, that $||Q - \tau_d||_{TV} \to 0$. □

C Optimality of GRCS

Algorithm 3 GRCS with arthmetic coding for the heap index.

Require: Target Q, proposal P over $\mathbb{R}$ with unimodal density ratio $r = dQ/dP$ with mode μ.

$d \leftarrow 0, T_0 \leftarrow 0, L_0 \leftarrow 0$
$I_0 \leftarrow 1, S_1 \leftarrow \mathbb{R}$
while `True` **do**
  $X_{I_d} \sim P|_{S_d}/P(S_d)$
  $U_{I_d} \sim \text{Uniform}(0,1)$
  $\beta_{I_d} \leftarrow \texttt{clip}\left(P(S_d) \cdot \frac{r(X_{I_d}) - L_d}{1 - T_d}, 0, 1\right)$ ▷ $\texttt{clip}(y,a,b) \overset{def}{=} \max\{\min\{y,b\},a\}$
  if $U_{I_d} \leq \beta_{d+1}$ **then**
    return X_{I_d}, I_d
  end if
  if $X_{I_d} > \mu$ **then**
    $I_{d+1} \leftarrow 2I_d$
    $S_{d+1} \leftarrow S_d \cap (-\infty, X_{I_d})$
  else
    $I_{d+1} \leftarrow 2I_d + 1$
    $S_{d+1} \leftarrow S_d \cap (X_{I_d}, \infty)$
  end if
  $L_{d+1} \leftarrow L_d + T_d/P(S_d)$
  $T_{d+1} \leftarrow \mathbb{P}_{Y\sim Q}[r(Y) \geq L_{d+1}] - L_{d+1} \cdot \mathbb{P}_{Y\sim P}[r(Y) \geq L_{d+1}]$
  $d \leftarrow d + 1$
end while

In this section, we prove Theorems 2 and 3. We are only interested in continuous distributions over $\mathbb{R}$ with unimodal density ratio dQ/dP for these theorems. Hence, we begin by specializing Algorithm 2 to this setting, shown in Algorithm 3. For simplicity, we also dispense with the abstraction of partitioning processes and show the bound update process directly. Furthermore, we also provide an explicit form for the `AcceptProb` and `RuledOutMass` functions.

Before we move on to proving our proposed theorems, we first prove two useful results. First, we bound the negative log P-mass of the bounds with which Algorithm 3 terminates.

Lemma 13. *Let Q and P be distributions over $\mathbb{R}$ with unimodal density ratio $r = dQ/dP$, given to Algorithm 3 as the target and proposal distribution as input, respectively. Let $d \geq 0$ and let $X_{1:d} \overset{def}{=} X_1, \ldots, X_d$ denote the samples simulated by Algorithm 3 up to step $d+1$, where for $d = 0$ we define the empty list as $X_{1:0} = \emptyset$. Let S_d denote the bounds at step $d+1$. Then,*

$$-\sum_{j=0}^{d} A_{j+1}(\mathbb{R}, S_{0:d}) \cdot \log P(S_j) \leq D_{\mathrm{KL}}[Q\|P] + \log e. \tag{159}$$

Proof. For brevity, we will write $A_d = A_d(\mathbb{R}, S_{0:d})$ and $T_d = T_d(\mathbb{R}, S_{0:d})$. Furthermore, as in Algorithm 3, we define

$$L_d \overset{def}{=} \sum_{j=0}^{d-1} \frac{1 - T_j}{P(S_j)} \quad \text{with} \quad L_0 = 0. \tag{160}$$

Note that $X_{1:d}$ is well-defined for all $d \geq 0$ since we could remove the return statement from the algorithm to simulate the bounds it would produce up to an arbitrary step d. Now, note that by Proposition 3 we have $\mathbb{P}[D = d \mid X_{1:d}] = A_{d+1}(\mathbb{R}, S_{0:d})$. Now, fix $d \geq 0$ and bounds $S_{0:d}$, and let $x \in \mathbb{R}$ be such that $\alpha_{d+1}(x) > 0$ which holds whenever $r(x) \geq L_d$. From this, for $d \geq 1$ we find

$$r(x) \geq \sum_{j=0}^{d-1} \frac{1 - T_j}{P(S_j)} \tag{161}$$

$$\geq \frac{1 - T_{d-1}}{P(S_{d-1})}, \tag{162}$$

where the second inequality follows from the fact that the $(1 - T_j)/P(S_j)$ terms are all positive. taking logs, we get

$$\log r(x) - \log(1 - T_{d-1}) \geq -\log P(S_{d-1}). \tag{163}$$

Now, we consider the expectation of interest:

$$\sum_{j=0}^{d} -A_{j+1} \cdot \log P(S_j) = -\sum_{j=0}^{d} \int_{\mathbb{R}} \alpha_{j+1}(x) \log P(S_j)\, dx \tag{164}$$

$$\overset{\text{eq. (163)}}{\leq} \sum_{j=0}^{d} \int_{\mathbb{R}} \alpha_{j+1}(x)(\log(r(x)) - \log(1 - T_j))\, dx \tag{165}$$

$$\overset{(a)}{\leq} \int_{\mathbb{R}} \sum_{j=0}^{\infty} \alpha_{j+1}(x) \log r(x)\, dx + \sum_{j=0}^{\infty} A_{j+1} \log \frac{1}{1 - T_j} \tag{166}$$

$$\overset{(b)}{=} \int_{\mathbb{R}} q(x) \log r(x)\, dx + \sum_{j=0}^{\infty} (T_{j+1} - T_j) \log \frac{1}{1 - T_j} \tag{167}$$

$$= D_{\mathrm{KL}}[Q\|P] + \sum_{j=0}^{\infty} (T_{j+1} - T_j) \log \frac{1}{1 - T_j} \tag{168}$$

$$\overset{(c)}{\leq} D_{\mathrm{KL}}[Q\|P] \cdot \log 2 + \int_0^1 \log \frac{1}{1 - t}\, dt \tag{169}$$

$$= D_{\mathrm{KL}}[Q\|P] + \log e. \tag{170}$$

Inequality (a) holds because all terms are positive. This is guaranteed by the fact that for $d \geq 1$, we have $L_d \geq 1$, hence $0 \leq \log L_d \leq r(x)$ whenever Equation (163) holds. Equality (b) follows by the correctness of GRC (Theorem 1), which implies that for all $x \in \mathbb{R}$ we have $\sum_{j=0}^{\infty} \alpha_d(x) = q(x)$, and inequality (c) follows from the facts that $0 \leq T_d \leq 1$ for all d and that the summand in the second term forms a lower-Riemann sum approximation to $-\log(1 - t)$. □

Second, we consider the contraction rate of the bounds $S_{0:d}$, considered by Algorithm 3.

Lemma 14. *Let Q and P be distributions over $\mathbb{R}$ with unimodal density ratio $r = dQ/dP$, given to Algorithm 3 as the target and proposal distribution as input, respectively. Assume P has CDF F_P and the mode of r is at μ. Fix $d \geq 0$ and let $X_{1:d}$ be the samples considered by Algorithm 3 and S_d the bounds at step $d + 1$. Then,*

$$\mathbb{E}_{X_{1:d}}[P(S_d)] \leq \left(\frac{3}{4}\right)^d \tag{171}$$

Proof. We prove the claim by induction. For $d = 0$ the claim holds trivially, since $S_0 = \mathbb{R}$, hence $P(S_0) = 1$. Assume now that the claim holds for $d = k - 1$, and we prove the statement for $d = k$. By the law of iterated expectations, we have

$$\mathbb{E}_{X_{1:k}}[P(S_k)] = \mathbb{E}_{X_{1:k-1}}[\mathbb{E}_{X_k|X_{1:k-1}}[P(S_k)]]. \tag{172}$$

Let us now examine the inner expectation. First, assume that $S_{k-1} = (a, b)$ for some real numbers $a < b$ and define $A = F_P(a), B = F_P(B), M = F_P(\mu)$ and $U = F_P(X_k)$. Since $X_k \mid X_{1:k-1} \sim P|_{S_{k-1}}$, by the probability integral transform we have $U \sim \mathrm{Unif}(A, B)$, where $\mathrm{Unif}(A, B)$ denotes the uniform distribution on the interval (A, B). The two possible intervals from which Algorithm 3 will choose are (a, X_k) and (X_k, b), whose measures are $P((a, X_k)) = F_P(X_k) - F_P(a) = U - A$ and similarly $P((X_k, b)) = B - U$. Then, $P(S_k) \leq \max\{U - A, B - U\}$, from which we obtain the bound

$$\mathbb{E}_{X_k|X_{1:k-1}}[P(S_k)] \leq \mathbb{E}_U[\max\{U - A, B - U\}] = \frac{3}{4}(B - A) = \frac{3}{4}P(S_{k-1}). \tag{173}$$

Plugging this into Equation (172), we get

$$\mathbb{E}_{X_{1:k}}[P(S_k)] \leq \frac{3}{4}\mathbb{E}_{X_{1:k-1}}\left[P(S_{k-1})\right] \tag{174}$$

$$\leq \frac{3}{4} \cdot \left(\frac{3}{4}\right)^{k-1}, \tag{175}$$

where the second inequality follows from the inductive hypothesis, which finishes the proof. □

The proof of Theorem 3: We prove our bound on the runtime of Algorithm 3 first, as this will be necessary for the proof of the bound on the codelength. First, let D be the number of steps Algorithm 3 takes before it terminates minus 1. Then, we will show that

$$\mathbb{E}[D] \leq \frac{1}{\log(4/3)} D_{\mathrm{KL}}[Q\|P] + 4 \tag{176}$$

We tackle this directly. Hence, let

$$\mathbb{E}_D[D] = \lim_{d\to\infty} \mathbb{E}_{X_{1:j}}\left[\sum_{j=1}^{d} j \cdot A_{j+1}\right] \tag{177}$$

$$= \lim_{d\to\infty} \mathbb{E}_{X_{1:j}}\left[\sum_{j=1}^{d} \frac{-j}{\log P(S_j)} \cdot -A_{j+1} \log P(S_j)\right] \tag{178}$$

$$\leq \lim_{d\to\infty} \mathbb{E}_{X_{1:j}}\left[\max_{j\in[1:d]}\left\{\frac{-j}{\log P(S_j)}\right\} \cdot \sum_{j=1}^{d} -A_{j+1} \log P(S_j)\right] \tag{179}$$

$$\overset{\text{lemma 13}}{\leq} (D_{\mathrm{KL}}[Q\|P] + \log e) \cdot \lim_{d\to\infty} \mathbb{E}_{X_{1:j}}\left[\max_{j\in[1:d]}\left\{\frac{-j}{\log P(S_j)}\right\}\right]. \tag{180}$$

To finish the proof, we will now bound the term involving the limit. To do this, note, that for any finite collection of reals F, we have $\max_{x\in F}\{x\} = -\min_{x\in F}\{-x\}$, and that for a finite collection of real-valued random variables $\hat{F}$ we have $\mathbb{E}[\min_{\mathbf{x}\in\hat{F}}\{\mathbf{x}\}] \leq \min_{\mathbf{x}\in\hat{F}}\{\mathbb{E}[\mathbf{x}]\}$. Now, we have

$$\lim_{d\to\infty} \mathbb{E}_{X_{1:j}}\left[\max_{j\in[1:d]}\left\{\frac{-j}{\log P(S_j)}\right\}\right] = \lim_{d\to\infty} -\mathbb{E}_{X_{1:j}}\left[\min_{j\in[1:d]}\left\{\frac{j}{\log P(S_j)}\right\}\right] \tag{181}$$

$$\leq \lim_{d\to\infty}\left(-\min_{j\in[1:d]}\left\{\mathbb{E}_{X_{1:j}}\left[\frac{j}{\log P(S_j)}\right]\right\}\right) \tag{182}$$

$$\overset{\text{(a)}}{\leq} \lim_{d\to\infty}\left(-\min_{j\in[1:d]}\left\{\frac{j}{\log \mathbb{E}_{X_{1:j}}\left[P(S_j)\right]}\right\}\right) \tag{183}$$

$$\overset{\text{lemma 14}}{\leq} \lim_{d\to\infty}\left(-\min_{j\in[1:d]}\left\{\frac{-j}{j\log(4/3)}\right\}\right) \tag{184}$$

$$= \lim_{d\to\infty}\left(\max_{j\in[1:d]}\left\{\frac{1}{\log(4/3)}\right\}\right) \tag{185}$$

$$= \frac{1}{\log(4/3)} \tag{186}$$

Inequality (a) follows from Jensen's inequality. Finally, plugging this back into the previous equation, we get

$$\mathbb{E}[D] \leq \frac{D_{\mathrm{KL}}[Q\|P] + \log e}{\log 4/3} \leq \frac{D_{\mathrm{KL}}[Q\|P]}{\log 4/3} + 4 \tag{187}$$

Proof of Theorem 2: For the codelength result, we need to encode the length of the search path and the search path itself. More formally, since the returned sample X is a function of the partition process Z, the search path length D and search path $S_{0:D}$, we have

$$\mathbb{H}[X \mid Z] \leq \mathbb{H}[D, S_{0:D}] = \mathbb{H}[D] + \mathbb{H}[S_{0:D} \mid D]. \tag{188}$$

we can encode D using Elias γ-coding, from which we get

$$\begin{aligned}
\mathbb{H}[D] &\leq \mathbb{E}_D[2\log(D+1)] + 1 && (189)\\
&\leq 2\log(\mathbb{E}[D]+1) + 1 && (190)\\
&\leq 2\log\left(\frac{D_{\mathrm{KL}}[Q\|P] + \log e}{\log(4/3)} + 1\right) + 1 && (191)\\
&\leq 2\log\left(D_{\mathrm{KL}}[Q\|P] + \log e + \log(4/3)\right) + 1 - 2\log\left(\log(4/3)\right) && (192)\\
&\leq 2\log\left(D_{\mathrm{KL}}[Q\|P] + 1\right) + 1 - 2\log\left(\log(4/3)\right) + 2\log(\log e + \log(4/3)) && (193)\\
&\leq 2\log\left(D_{\mathrm{KL}}[Q\|P] + 1\right) + 6. && (194)
\end{aligned}$$

Given the search path length D, we can use arithmetic coding (AC) to encode the sequence of bounds $S_{0:D}$ using $-\log P(S_D) + 2$ bits (assuming infinite precision AC). Hence, we have that the average coding cost is upper bounded by

$$\mathbb{H}[S_{0:D} \mid D] \leq \mathbb{E}_D[-\log P(S_D)] + 2 \overset{\text{lemma 13}}{\leq} D_{\mathrm{KL}}[Q\|P] + 5. \tag{195}$$

Putting everything together, we find

$$\mathbb{H}[D, S_{0:D}] \leq D_{\mathrm{KL}}[Q\|P] + 2\log(D_{\mathrm{KL}}[Q\|P] + 1) + 11, \tag{196}$$

as required.

D Additional experiments with depth-limited GRC

In this section we show the results of some experiments comparing the approximation bias of depth limited GRCD, to that of depth limited AD* , following the setup of Flamich et al. (2022). Limiting the depth of each algorithm introduces bias in the resulting samples, as these are not guaranteed to be distributed from the target distribution Q, but rather from a different distribution $\hat{Q}$. Figure 5 quantifies the effect of limiting the depth on the bias of the resulting samples.

In our experiment we take Q and P to be Gaussian and we fix $D_{\mathrm{KL}}[Q\|P] = 3$ (bits), and consider three different settings of $D_\infty[Q\|P] = 5, 7$ or 9 (bits), corresponding to each of the panes in fig. 5. For each such setting, we set the depth limit of each of the two algorithms to $D_{\max} = D_{\mathrm{KL}}[Q\|P] + d$ bits, and refer to d as the *number of additional bits*. We then vary the number of additional bits allowed for each algorithm, and estimate the bias of the resulting samples by evaluating the KL divergence between the empirical and the exact target distribution, that is $D_{\mathrm{KL}}[\hat{Q}\|Q]$. To estimate this bias, we follow the method of Pérez-Cruz (2008). For each datapoint shown we draw 200 samples $X \sim \hat{Q}$ and use these to estimate $D_{\mathrm{KL}}[\hat{Q}\|Q]$. We then repeat this for 10 different random seeds, reporting the mean bias and standard error in the bias, across these 10 seeds.

Generally we find that the bias of GRCD is higher than that of AD* . This is likely because AD* is implicitly performing importance sampling over a set of $2^{D_{\max}+d} - 1$ samples, and returning the one with the highest importance weight. By contrast, GRCD is running rejection sampling up to a maximum of $D_{\max} + d$ steps, returning its last sample if it has not terminated by its $(D_{\max} + d)^{\text{th}}$ step. While it might be possible to improve the bias of depth limited GRCD by considering an alternative way of choosing which sample to return, using for example an importance weighting criterion, we do not examine this here and leave this possibility for future work.

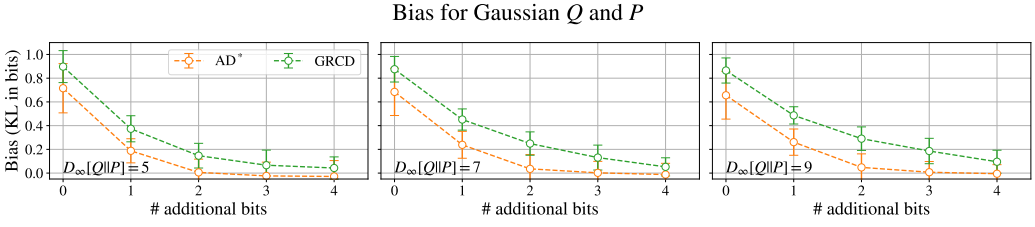


Figure 5: Bias of depth-limited AD* and GRCD, as a function of the number of additional bit budget given to each algorithm. See text above for discussion.