Minimal thermodynamic cost of communication

Abhishek Yadav [1,2,*] and David Wolpert [1,3,4,5]
[1] Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, New Mexico 87501, USA
[2] Department of Physical Sciences, IISER Kolkata, Mohanpur 741246, India
[3] Complexity Science Hub, Vienna, Austria
[4] Arizona State University, Tempe, Arizona 85281, USA
[5] International Center for Theoretical Physics, Trieste 34151, Italy

(Received 21 December 2024; accepted 23 November 2025; published 22 December 2025; corrected 30 October 2026)

Thermodynamic cost of communication is a major factor in the thermodynamic cost of real-world computers, both biological and digital. Despite its importance, the fundamental principles underlying this cost remain poorly understood. This paper makes two major contributions to addressing this gap. First, we derive a universal relationship between information transmission rate and minimal entropy production by focusing on the mismatch cost component of thermodynamic cost. The resulting relationship holds independently of the underlying physical dynamics, making it broadly applicable. We discuss the implications of the minimal communication cost for work extraction in measurement-and-feedback protocols, and through examples involving binary channels, we show that the relationship between transmission rate and minimal thermodynamic cost can exhibit diminishing returns in certain scenarios. Second, we extend this thermodynamic analysis to the computational front and back ends critical to communication—namely, encoding and decoding to reduce errors in noisy transmission. Using the framework of periodic machines, we establish strictly positive minimal costs for implementing linear error-correcting codes. We compare these costs with end-to-end error rates, highlighting trade-offs between thermodynamic efficiency and decoding accuracy.

DOI: 10.1103/qvc2-32xr

I. INTRODUCTION

A. Background and scope

Modern computers rely on decomposing large-scale, complex problems into simpler calculations handled by subcomponents that are reused iteratively throughout a computation. These components, such as logic gates within circuits, must communicate to execute desired computation. Given that communication is integral to nearly all forms of computation, it becomes imperative to understand the energetic costs associated with communication to fully understand the energetic costs of computation itself. Even if individual computing units are thermodynamically efficient, the overall system can still incur a high thermodynamic cost due to the energy required for communication.

Early work by Landauer and Bennett investigated the energetic costs of computation [1–3], arriving at a semiformal understanding of minimal heat generation in logically irreversible processes such as bit erasure. These early analyses were based on the idea that a change in the Shannon entropy of a system during computation results in exchange of heat with the environment, quantified by $k_B T \Delta S$, where k_B is the Boltzmann constant, T is the temperature of the thermal bath, and ΔS is the change in the system's entropy. While this intuition has been validated by more recent developments in stochastic thermodynamics, it was incomplete [4].

In particular, it failed to distinguish between reversible and irreversible components of total heat flow. The entropy change ΔS contributes only to the reversible part of heat transfer—heat that can, in principle, be recovered through a time-reversed protocol [5]. However, it is now understood that an additional, inherently irreversible contribution exists: the entropy production (EP), which accounts for heat that is unavoidably lost to the environment [5,6].

This important distinction was not fully appreciated in early work, leading to the inaccurate view that logical reversibility implies thermodynamic reversibility. In reality, these are distinct concepts. A computation is logically reversible if, given any output, the input can be uniquely reconstructed—this condition ensures no loss of information and implies $\Delta S = 0$, yielding zero Landauer cost. However, thermodynamic reversibility requires that the EP be zero. Thus, while logical reversibility eliminates the reversible (Landauer) component of heat, it does not necessarily eliminate the total energetic cost: Entropy production may still be present. Logical reversibility alone is therefore insufficient to guarantee thermodynamic reversibility [4].

This misconception influenced early views on communication as well. In earlier literature, it was argued that communication—viewed as a copy operation—could be achieved without any fundamental energetic cost [7]. For instance, consider the following operation:

$$00 \mapsto 00 \quad \text{and} \quad 10 \mapsto 11. \tag{1}$$

*Contact author: abhi.yadav.28k@gmail.com

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

Here, the left bit is copied to the right bit, and the right bit always starts in state 0. This operation is logically reversible, and thus was claimed to have no unavoidable thermodynamic cost. But such reasoning only applies to the reversible component of heat flow and ignores EP.

This paper has two main goals. First, we derive a strictly positive lower bound on the thermodynamic cost associated with communication through a noisy channel, thereby establishing that communication—contrary to some earlier beliefs—has an unavoidable thermodynamic cost. We model communication as a noisy copy operation, analogous to the idealized example in Eq. (1), with the channel consisting of an input and an output node. Crucially, the operation of a communication channel involves copying the value of input node to the output node with some noise and subsequently resetting the value of the input node with a new value drawn from a source. This cycle—reset, copy, reset—repeats with each new message.

Second, we extend this thermodynamic analysis to the computational front and back ends critical to communication, i.e., encoding and decoding to reduce errors in noisy transmission [8]. Subsequently, we present a comprehensive analysis of the thermodynamic costs associated with the complete end-to-end communication system, encompassing the encoder, channel, and decoder.

Although prior work has examined energetic costs of encoding and decoding—particularly in the context of very large scale integration (VLSI) circuits [9,10]—our approach differs in scope and methodology. We use tools from stochastic thermodynamics and analyze abstract, algorithmic implementations of encoding and decoding, rather than specific hardware realizations. As such, the bounds we present are theoretical, and offer insights into energetic costs of encoding and decoding algorithms.

Our analysis is grounded in the mismatch cost (MMC) decomposition of thermodynamic cost. For any process on state space $\mathcal{X}$ that evolves an initial distribution p_0 to a final distribution $p_1 = Gp_0$, a broad class of thermodynamic cost functions can be written in the form

$$\mathcal{C}(p_0) = \langle f \rangle_{p_0} + S(Gp_0) - S(p_0), \tag{2}$$

where $f : \mathcal{X} \to \mathbb{R}$ is a finite real-valued function and $S(\cdot)$ denotes Shannon entropy.

Such a cost admits the following decomposition [11,12]:

$$\mathcal{C}(p_0) = D(p_0 \| q_0) - D(Gp_0 \| Gq_0) + \mathcal{C}(q_0), \tag{3}$$

where $D(\cdot\|\cdot)$ is the Kullback-Leibler (KL) divergence and $q_0 = \arg\min_{r_0} \mathcal{C}(r_0)$ is known as the prior associated with the cost function $\mathcal{C}$. The quantity $D(p_0\|q_0) - D(Gp_0\|Gq_0)$ is referred to as the MMC. It quantifies the portion of the total cost attributable to a mismatch between the actual initial distribution p_0 and the prior q_0. This decomposition holds for any choice of G and f, and importantly, MMC provides a non-negative contribution for any $p_0 \in \Delta_{\mathcal{X}}$.

Several thermodynamic cost measures—including total entropy production, the change in nonequilibrium free energy, and nonadiabatic entropy production—can be expressed in the form of Eq. (3) and therefore MMC gives a non-negative contribution in those cases [11]. Recent work has shown that the MMC can account for a substantial portion of the total thermodynamic cost in certain scenarios [13].

In this paper, we specifically focus on the MMC contribution to the total thermodynamic cost. It is important to emphasize that we do not assume or specify any particular physical model for the components of the communication system. Instead, our analysis centers on how the probability distribution over system states evolves during each fundamental step of the communication process. For instance, in the case of encoders and decoders, each computational step induces a change in the system's state distribution, resulting in a sequence $\{p_0, p_1, \ldots, p_n\}$.

It is true that, given a function $f : \mathcal{X} \to \mathbb{R}$ and map G, the corresponding prior distribution can be derived [13]. The function f captures some physical details of the process that implements the map G. For instance, in the case of total EP, $f(x)$ represents the average heat exchanged between the system and its thermal environment when the system begins in state x.

In this work though, we do not derive the prior from a specific physical model of a communication system. Instead, to gain concrete and interpretable insights into the thermodynamics of communication processes, we adopt simple and illustrative choices for the prior distributions.

Nonetheless, the minimal thermodynamic cost that we derive for communication channels holds independently of the specific choice of prior or the physical implementation of the communication channel. The resulting bound is universal and applies to any communication channel.

In the case of encoder and decoder components, however, our chosen priors are not intended to imply generality across all possible priors. Instead, they serve to illustrate that—even in the absence of a detailed physical model—each algorithmic step can contribute a nonzero thermodynamic cost, and MMC can be used to analyze it once the prior is known.

B. Roadmap and summary

The paper is organized as follows: We briefly review Shannon's communication theory in Sec. II A. We restrict our attention to independent and identically distributed (i.i.d.) information sources, and discrete memoryless communication channels. We present entropy of the source as an information measure and mutual information across the channel as the rate of information transfer through a channel. Subsequently, we give a brief account of error-correcting codes, outlining how they enable the detection and correction of message errors, with a specific focus on linear encoders and syndrome decoders.

Section II B provides an overview of stochastic thermodynamics. We review the MMC lower bound on EP, laying the foundation for later discussions. In Sec. III, we present our first main result: an MMC lower bound on the thermodynamic cost of communication through a noisy channel [Eqs. (37) and (38)]. We then highlight measurement-and-feedback scenarios where communication cost directly impacts work extraction, noting that measurement is essentially a communication process, and as a result, the minimal cost associated with communication plays a crucial role in such settings.

In Sec. III C, we examine a concrete example involving binary channels. For a specific choice of prior, we demon-

TABLE I. Table of notation for the main symbols used in the paper.

Table of notation	
Symbol	Definition
$\pi_{Y\|X}$	Conditional distribution representing the noise in the channel
p_X	Input or source distribution
p^a_{XY}	Distribution over joint state of channel before copy operation
p^b_{XY}	Distribution over joint state of channel after copy operation
q^a_{XY}	Prior distribution of process of overwriting of input node
q^b_{XY}	Prior distribution of process of copying input state to output node
$\tilde{q}^a_{XY}$	Distribution resulting form prior q^a_{XY} after overwriting of input node
$\tilde{q}^b_{XY}$	Distribution resulting form prior q^b_{XY} after copy operation
MC^a	Mismatch cost associated with the process of overwriting of input node
MC^b	Mismatch cost associated with the process of copying of input state to output node.

strate that the relationship between the rate of information transmission and the corresponding minimal cost is concave, reflecting diminishing returns. This implies that splitting the information rate across multiple channels can yield a lower total minimal cost than transmitting the same rate through a single channel.

Section IV shifts the focus to the encoding and decoding procedures. These algorithms consist of a sequence of computational steps. In Sec. IV A, we introduce a formal framework to model the implementation of such algorithms, grounded in the periodic nature of modern computer architectures and the essential role of the instruction pointer. This framework was initially developed in Ref. [13].

Using this framework, Secs. IV B and IV C analyze the stepwise MMC incurred by linear encoders and syndrome decoders. Importantly, we make some rather straightforward and illustrative choices for the priors, usually assuming that they are uniform distribution. This allows us to compare the minimal thermodynamic cost across various linear error-correcting codes, each characterized by different encoding lengths and decoding error rates. Finally, we combine the thermodynamic cost of encoding and decoding with the minimal cost associated with the communication channel, yielding a total end-to-end cost of the full communication system—comprising encoding, channel transmission, and decoding. We compare the associated MMC cost of the system with the corresponding end-to-end error rate, and highlight the trade-offs between decoding error rates and the minimal cost. We conclude with a broader discussion in Sec. V. The main symbols used in the paper are presented in Table I.

II. PRELIMINARIES

A. Shannon's theory of communication

The fundamental problem of communication involves transmitting information from a source to a receiver, a process that is almost always subject to unavoidable noise. At an abstract level, communication can be viewed as a copying operation, where symbols generated by a source are reproduced, often imperfectly, at the receiving end.

Claude Shannon laid the foundation of modern information theory by quantifying the information content of a symbol emitted by a source, or more precisely, the average amount of information generated per symbol [8]. For an i.i.d. source defined by a set of possible messages and a corresponding probability distribution, the average information content per symbol is quantified by the Shannon entropy:

$$S(p_X) = -\sum_{x\in\mathcal{X}} p_X(x)\log p_X(x), \tag{4}$$

where $\mathcal{X}$ is the set of possible messages, X is the random variable associated with the input state, and p_X is the distribution over the messages. The source is maximally informative when p_X is uniformly distributed over X. Conversely, the source produces zero information when the probability distribution is degenerate (i.e., when all probability mass is concentrated on a single symbol).

If the channel, or the copy operation, is noisy, then the information received is not identical to the information produced at the source; some of it is inevitably lost during transmission. Let the noise in the channel be characterized by the conditional distribution $\pi_{Y|X}$, where Y is the output variable with state space $\mathcal{Y}$. On average, the amount of information lost due to noise is given by the conditional entropy:

$$S(X|Y) = -\sum_{\substack{x\in\mathcal{X},\\ y\in\mathcal{Y}}} p_{XY}(x,y)\log \pi_{X|Y}(x|y), \tag{5}$$

where p_{XY} is the joint distribution obtained by $p_{XY}(x,y) = \pi_{Y|X}(y)p_X(x)$. The amount of information successfully transmitted from source to receiver is then given by the difference between the total information generated at the source and the information lost in the channel:

$$I(X;Y) = S(X) - S(X|Y), \tag{6}$$

known as the mutual information between X and Y.

This mutual information represents the average information transmitted per use of the channel, and for that reason it is called the rate of information transmission or rate of communication.

Error correcting codes

In modern communication protocols, the focus is on accurately predicting the original message despite the errors introduced by the channel noise. For this purpose, a communication channel is often supplemented with an encoder to

encode the messages before sending them through the channel and a decoder at the other end of the channel [14]. Assuming that the source messages are optimally represented as binary strings of length k, an (n, k) binary encoder is a function that maps these messages to binary strings of length $n > k$:

$$\mathrm{E} : \{0, 1\}^k \longrightarrow \{0, 1\}^n. \tag{7}$$

Each binary string $\mathbf{s} \in \{0, 1\}^k$ is mapped to a *valid codeword* $\mathrm{E}(\mathbf{s})$ in $\{0, 1\}^n$. *Hamming distance* between two codewords is defined as the number of positions at which their bits differ. The *minimum distance* $d_{\min}$ of a code E is the smallest Hamming distance between any pair of distinct valid codewords:

$$d_{\min} := \min_{\{\mathbf{s} \neq \mathbf{s}'\}} d(\mathrm{E}(\mathbf{s}), \mathrm{E}(\mathbf{s}')), \tag{8}$$

where $d(\mathbf{t}, \mathbf{t}')$ is the Hamming distance between $\mathbf{t}$ and $\mathbf{t}'$.

A higher minimum distance is desirable because it increases the capability to detect and correct errors. If the Hamming distance between codewords is very small, even minor errors can turn one valid codeword into another, making it impossible to detect error. Goal of an encoder is to maximize the minimum distance for a given n and k. In fact, given n and k, the maximum achievable minimum distance is bounded by [14]

$$d_{\min} \leqslant n - k + 1. \tag{9}$$

Increasing n generally helps achieve a larger minimum distance, which improves error detection and correction. However, a larger n also means that each codeword occupies more bits, requiring an increased number of channel use to send a message, and therefore slowing down the communication.

We briefly introduce a simple and widely used class of error-correcting codes known as linear codes. In a linear code, the encoding function is a linear transformation represented by a matrix of the form

$$\mathrm{E}^T = \begin{bmatrix} I_k \\ P \end{bmatrix}, \tag{10}$$

where I_k is the $k \times k$ identity matrix and P is a $(n - k) \times k$ matrix known as the parity matrix. Given an input message $\mathbf{s} \in \{0, 1\}^k$, represented as a column vector, the corresponding codeword $\mathbf{t} \in \{0, 1\}^n$ is obtained by

$$\mathbf{t} = \mathrm{E}^T \mathbf{s}. \tag{11}$$

The first k bits of the codeword are a direct copy of the input, while the remaining $n - k$ bits—called parity bits—are linear combinations of the input bits as specified by P. These parity bits encode redundancy that enables error detection and correction. A commonly used decoding strategy for linear codes is the syndrome decoder.

Syndrome decoding relies on a key property of linear codes known as the syndrome. The syndrome is a binary vector associated with each codeword: It is zero if the word is a valid codeword, and nonzero if an error has occurred during transmission. Moreover, the value of the syndrome often contains information about the location of the error, making it useful for correction. For a linear code $\mathrm{E} = [I_{k\times k}, P^T]$, the corresponding *parity check matrix* H is defined as

$$\mathrm{H} = [P \quad I_{n-k}]. \tag{12}$$

The syndrome associated with a received codeword $\mathbf{r} \in \{0, 1\}^n$ is defined as $\mathbf{z} = \mathrm{H}\mathbf{r}$, where H is the parity-check matrix. A key property of linear codes is that all valid codewords lie in the kernel of H, i.e.,

$$\mathrm{H}\mathbf{t} = \mathbf{0}, \quad \text{for all valid codewords } \mathbf{t} = \mathrm{E}^T\mathbf{s}. \tag{13}$$

Thus, any received codeword with a nonzero syndrome indicates that an error has occurred during transmission. The syndrome not only detects the presence of an error but can also be used to locate it. Let $\mathbf{e}$ denote the binary error vector introduced by the channel. Then the received string $\mathbf{r}$ can be expressed as

$$\mathbf{r} = \mathbf{t} \oplus \mathbf{e}, \tag{14}$$

where $\mathbf{t}$ is the transmitted codeword and $\oplus$ indicates a modulo 2 addition. Using the property in Eq. (11), the syndrome of the received string becomes

$$\mathbf{z} = \mathrm{H}\mathbf{r} = \mathrm{H}(\mathbf{t} \oplus \mathbf{e}) = \mathrm{H}\mathbf{e}. \tag{15}$$

This shows that the syndrome depends solely on the error vector $\mathbf{e}$, and not on the original codeword $\mathbf{t}$, and this can be used to detect and correct error. The third equality, $\mathbf{z} = \mathrm{H}\mathbf{e}$, establishes a relationship between the syndrome of a received codeword and the error vector $\mathbf{e}$ that might have occurred during transmission. With a precomputed lookup table that maps all possible error vectors $\mathbf{e}$ to their corresponding syndromes $\mathbf{z} = \mathrm{H}\mathbf{e}$, one can effectively decode the received message by (1) observing the syndrome $\mathbf{z}$ of the received vector $\mathbf{r}$, and then (2) using the lookup table to identify the most probable error vector $\tilde{\mathbf{e}}$ corresponding to the observed syndrome $\mathbf{z}$, and finally, (3) predict the originally transmitted valid codeword by adding this error vector (modulo 2) to the received codeword $\mathbf{r}$

$$\hat{\mathbf{t}} = \mathbf{r} \oplus \tilde{\mathbf{e}}. \tag{16}$$

It is important to emphasize that despite all the syndrome decoding, there is always a nonzero possibility of making an incorrect prediction of $\tilde{\mathbf{e}}$ since there could be more than one $\mathbf{e}$ associated with a single $\mathbf{z}$. In that case, the decoder has to make the most probable estimate of $\mathbf{e}$. An incorrect estimate $\tilde{\mathbf{e}}$ leads to a decoding error. The block error rate is the probability that an entire transmitted codeword (or block) is decoded incorrectly, i.e., $P(\hat{\mathbf{t}} \neq \mathbf{t})$. Unlike bit error rate, which measures errors per individual bit, block error rate captures whether the decoder successfully recovers the full original message block without any errors. Reducing the block error rate generally requires increasing the redundancy of the code by making the codewords longer, which also consequently leads to a decrease in the communication rate [14].

B. Relevant stochastic thermodynamics

Equilibrium statistical mechanics derives thermodynamics properties of a system from the underlying equilibrium statistics. Stochastic thermodynamics is an extension that defines thermodynamic quantities such as heat, work, and entropy to

the level of stochastic trajectories in processes evolving arbitrarily far from equilibrium. The key assumption in stochastic thermodynamics is that any degree of freedom not explicitly described by the system's dynamics—such as the heat or particle reservoirs—remains in equilibrium. This allows us to connect thermodynamic quantities and dynamics, through the principle of local detailed balance [15–18].

One of the central quantities in stochastic thermodynamics is EP, which quantifies the irreversible heat dissipation during a process. Consider a system with a discrete state space $\mathcal{X}$ that evolves over a time interval $[0, \tau]$, transforming an initial distribution p_{X_0} into a final distribution p_{X_τ}. Suppose that the system is coupled to several heat reservoirs indexed by i, each characterized by an inverse temperature β_i. Let $\mathcal{Q}_i(x_0)$ denote the expected heat flow from the system into the ith reservoir, conditioned on the system starting in state x_0. The expected total heat flow into the ith reservoir over the process is given by

$$\langle \mathcal{Q}_i \rangle_{p_{X_0}} = \sum_{x \in \mathcal{X}} p_{X_0}(x) \mathcal{Q}_i(x). \tag{17}$$

With this, the total entropy production during the process is expressed as

$$\mathcal{C}(p_{X_0}) = \sum_i \beta_i \langle \mathcal{Q}_i \rangle_{p_{X_0}} - [S(p_{X_0}) - S(p_{X_\tau})], \tag{18}$$

where $S(p) = -\sum_x p(x) \log p(x)$ is the Shannon entropy. The first sum in the expression is also called the *total entropy flow* and it corresponds to the change in the thermodynamic entropy of the environments during the process. Furthermore, the total entropy flow from the system to the reservoirs can be written compactly as

$$\sum_i \beta_i \langle \mathcal{Q}_i \rangle_{p_{X_0}} = \sum_{x \in \mathcal{X}} p_{X_0}(x) f(x) = \langle f \rangle_{p_{X_0}}, \tag{19}$$

where $f(x) := \sum_i \beta_i \mathcal{Q}_i(x)$ is the expected change in the thermodynamic entropy of all the reservoirs combined if the system starts in state x.

Thus, $\mathcal{C}(p_{X_0})$ is the part of the total entropy flow that is irreversibly lost into the environment, called the entropy production, whereas $[S(p_{X_0}) - S(p_{X_1})]$ is the reversible part of the total heat flow, called Landauer cost. According to the second law of thermodynamics, entropy production is always non-negative:

$$\mathcal{C}(p_{X_0}) \geqslant 0. \tag{20}$$

The expected entropy flow $\langle f \rangle_{p_{X_0}}$, which captures the average heat dumped into the reservoirs, is therefore always lower bounded by the Landauer cost:

$$\langle f \rangle_{p_{X_0}} \geqslant S(p_{X_0}) - S(p_{X_1}). \tag{21}$$

This lower bound is saturated in the limit of quasistatic (infinitely slow) or near-equilibrium processes, where $\mathcal{C}(p_{X_0}) \to 0$. Moreover, the Landauer cost itself is upper bounded by $\ln |\mathcal{X}|$. In contrast, for processes that are far from equilibrium, the total entropy flow $\langle f \rangle_{p_{X_0}}$ can far exceed this bound, meaning that EP dominates the total thermodynamic cost.

Advances in stochastic thermodynamics have led to the discovery of various contributions to total EP based on key properties of a system's dynamics. Notably, thermodynamic uncertainty relations (TURs) establish a contribution to EP that arises from the precision of currents involved in the system's evolution [19–21]. Another significant class of results, known as speed limit theorems (SLTs), provides a lower bound on EP in terms of the average number of state transitions during the system's evolution [22,23]. These results require some key details of systems dynamics to lower bound EP; for instance, SLTs require knowledge of the average number of transitions in the underlying dynamics, while TURs depend on the precision of currents involved in the process.

A distinct and complementary contribution to EP is known as the mismatch cost. Consider the initial distribution that minimizes EP in Eq. (18), denoted as q_{X_0}, which we refer to as the prior distribution of the process. When the process starts in a distribution different from q_{X_0}, the resulting excess EP is given by the mismatch cost. $\mathcal{C}(p_0)$ can be decomposed as [11,24]

$$\mathcal{C}(p_0) = [D(p_{X_0} \| q_{X_0}) - D(p_{X_\tau} \| q_{X_\tau})] + \mathcal{C}_{\text{res}}, \tag{22}$$

where $D(p\|q)$ is the KL divergence between p and q, and $\mathcal{C}_{\text{res}}$, known as the residual EP, represents the minimum EP of the process, defined as $\mathcal{C}_{\text{res}} := \mathcal{C}(q_0)$. The drop in KL divergence $D(p_{X_0} \| q_{X_0}) - D(p_{X_\tau} \| q_{X_\tau})$ is called the MMC. Moreover, the mapping of any initial distribution of systems state to final distribution under a dynamics can be compactly described by a stochastic map G such that $p_{X_\tau}(x) = \sum_{x' \in \mathcal{X}} G(x|x') p_{X_0}(x')$ or $p_{X_\tau} = G p_{X_0}$. Then, MMC can be written as

$$\text{MC}(p_{X_0}) = D(p_{X_0} \| q_{X_0}) - D(G p_{X_0} \| G q_{X_0}). \tag{23}$$

Due to the data processing inequality for KL divergence, MMC is always non-negative [25]. By the second law, the residual EP satisfies $\mathcal{C}_{\text{res}} \geqslant 0$, and consequently MMC provides a non-negative lower bound on total cost. The detailed physical implementation of the process is encoded in the residual cost and the prior distribution. However, once the prior is known, the MMC gives a guaranteed lower bound on EP for any arbitrary initial distribution. Crucially, this bound applies broadly to a general Markovian or non-Markovian process.

Furthermore, in Ref. [13], it has been shown that MMC can be a significant contribution to the total cost, especially when total heat generation during the process is significant. To sketch the intuition, recall that $f(x)$ represents the expected entropy flow to the reservoirs when the system starts in state x. One can rewrite Eq. (18) as

$$\mathcal{C}(p_{X_0}) = \sum_x f(x) p_{X_0}(x) - [S(p_{X_0}) - S(G p_{X_0})]. \tag{24}$$

Prior distribution is entirely determined by $f(x)$ and the map G [13]. In a scenario, where the total heat generation is large on a scale of $\ln |\mathcal{X}|$ and $f(x)$ is not uniform across all states $x \in \mathcal{X}$, the resulting prior distribution gets closer to the edges of the probability simplex. In other words, the prior becomes increasingly peaked. This peaking of the prior results in a high value of KL divergence between a typical actual distribution p_{X_0} and the prior q_{X_0}, thereby increasing the MMC. More precisely, if $\max_{x \in \mathcal{X}} f(x) - f(c) > \ln |\mathcal{X}|$, then the mismatch cost for the worst-case input distribution, denoted as $\text{MC}^* =$

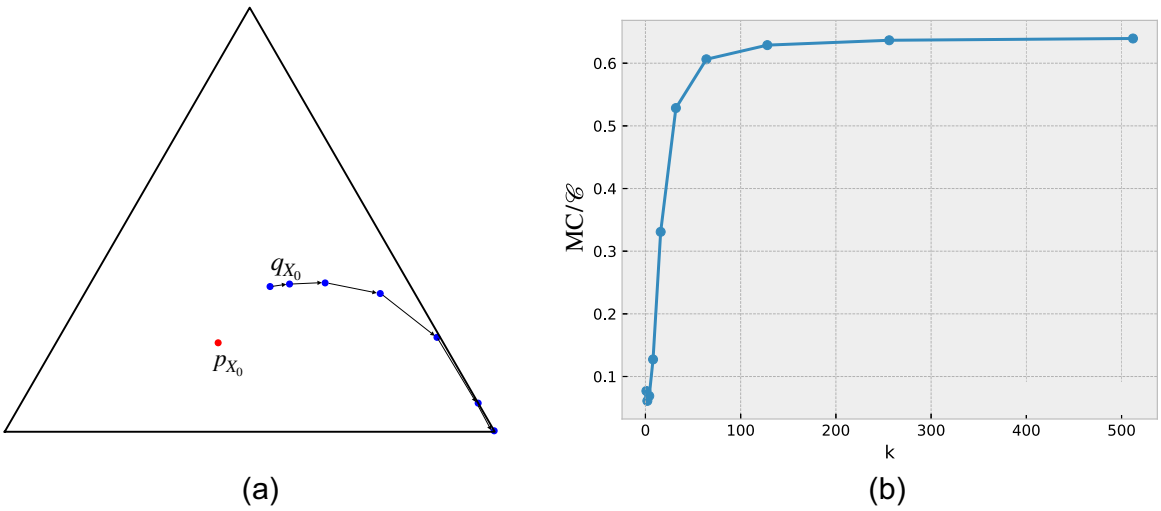


FIG. 1. Mismatch cost increases significantly with entropy flow scaling k, demonstrating its dominant contribution in high-EP scenarios. Starting with arbitrary chosen values $f(1) = 0.5$, $f(2) = 0.1$, and $f(3) = 0.2$, we scale all $f(x)$ by a factor k, resulting in new values $kf(x)$. (a) As k increases, the associated prior distribution $q_{X_0} = \arg\min_{r_{X_0}} \mathcal{C}(r_{X_0})$ [as defined by Eq. (24)] shifts closer to the boundary of the 2-simplex. (b) Consequently, the mismatch cost for a typical initial distribution (here, $p_{X_0} = \{0.46, 0.33, 0.21\}$) becomes a larger fraction of the total EP, eventually exceeding 60% of the total EP. The calculations are performed using the map G defined in Eq. (26), with ϕ set to 0.1. The total entropy production is computed from $f(x)$ and G using Eq. (24), while the corresponding mismatch cost for each resulting prior is calculated using Eq. (23).

$\max_{p_{X_0}} \mathrm{MC}(p_{X_0})$), is lower bounded by [13]

$$\mathrm{MC}^* \geqslant \max_x f(x) - \min_x f(x) - \ln|\mathcal{X}|. \tag{25}$$

Figure 1 illustrates this effect for a three-state system under a map G,

$$G = \begin{bmatrix} 1-\phi & \phi & 0 \\ 0 & 1-\phi & \phi \\ \phi & 0 & 1-\phi \end{bmatrix}, \tag{26}$$

where $\phi \in [0, 1]$. Starting from an arbitrary function $f(x)$, scaling its values by a factor of k causes the prior distribution to shift toward the edges of the 2-simplex. As this scaling increases, the associated MMC becomes an increasingly significant contributor to the total EP for a typical initial distribution, as shown in Fig. 1.

In the context of communication channels, MMC is central to understanding how the rate of information transmission relates to its cost. As discussed earlier, the information transmitted from X to Y during each copy operation is quantified by the mutual information $I(X;Y)$, which depends on the input distribution p_X for a fixed channel. From the decomposition of total thermodynamic cost in Eq. (22), the MMC is the component of the cost that explicitly depends on p_X, while the residual term remains fixed across all input distributions. The input distribution to a communication channel determines the initial joint distribution over the channel's state. Since MMC captures the part of EP that depends on this initial distribution, it effectively isolates the component of EP that varies with the input. Consequently, when analyzing how changes in information transmission affect the total thermodynamic cost, MMC identifies precisely the portion that covaries with mutual information.

III. MINIMAL THERMODYNAMIC COST IN COMMUNICATION CHANNELS

The model of communication channel that we are going to consider in this section is inspired by many real-world channels in which the source end is continuously and repeatedly updated with new values, and each of which is copied to the other end of the channel, the receiver end. Crucially, it is not a one-time process but it is continuously repeated—source values are repeatedly updated and copied to the receiver in an ongoing cycle. Such dynamics arise in a variety of communication channels—for example, in cellular signaling pathways, where ligand concentrations at the membrane are continuously sensed and propagated downstream [26,27] and in digital devices, where sensors or processors continuously write new values to shared memory or communication buses for retrieval by downstream components.

We represent a noisy communication channel as a two-node graph with an input node X and an output node Y, and a conditional distribution $\pi_{Y|X}$ that specifies the probability of output given input (see Figs. 2 and 3). The joint distribution p_{XY} defines the state of the channel. Communication involves two key steps:

(a) Overwriting of input: A value is assigned to the input node X, drawn randomly from a distribution p_X.

(b) Copying input value to output: The input value is then copied to the output node Y, subject to noise governed $\pi_{Y|X}$.

Importantly, as in many practical scenarios where channels are repeatedly reused, in this model, steps (a) and (b) are applied in succession over and over again. More specifically, after the end of step (b) in each iteration of the channel, the output and input node are correlated. Let p^b_{XY} denote the joint state of the channel at the end of step (b) from a previous cycle. It can be written as

$$p^b_{XY}(x, y) = \pi_{Y|X}(y|x)\, p_X(x). \tag{27}$$

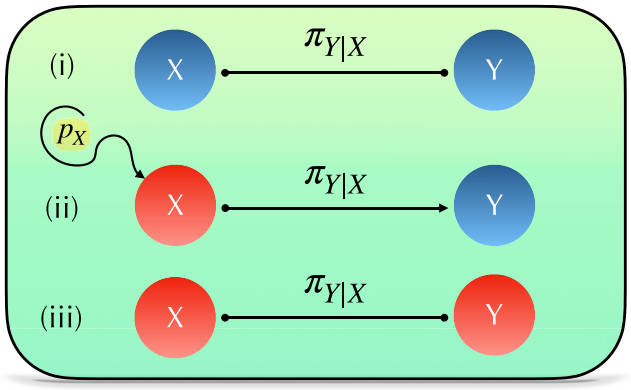


FIG. 2. Schematic of a communication channel: (1) The state of the input node X is updated with a new value drawn from the information source. (2) This input state is then copied to the output node Y, subject to noise characterized by the conditional distribution $\pi_{Y|X}$.

For subsequent use of the channel, input node is overwritten with a new value that is randomly sampled from the distribution p_X. Let p^a_{XY} denote the joint state of the channel after overwriting process (a). Since the new input value is sampled randomly, p^a_{XY} is given by

$$p^a_{XY}(x, y) = p_X(x)\, p_Y(y), \tag{28}$$

where $p_Y(y) = \sum_x p^b_{XY}(x, y)$ is the marginal distribution over Y. The next step, (b), involves copying the new input value of X to the output node Y subject to the noise $\pi_{Y|X}$. After this copy process, the joint distribution returns to $p^b_{XY}(x, y)$.

In this way, each iteration of sending a bit through the channel causes the system to cycle from the state p^b_{XY} to p^a_{XY}, and then back to its initial form p^b_{XY} (see Fig. 4).

A. Mismatch cost in communication channel

The joint state of the channel consists of the input node X and the output node Y. The process of input overwriting can be viewed as the evolution of this joint system, where the state of X changes independently of Y, while the state of Y remains unchanged. This type of evolution is known as a subsystem

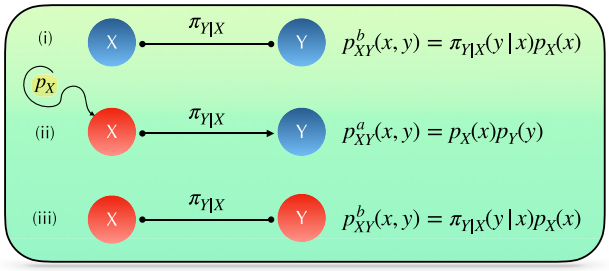


FIG. 3. We model the communication channel as consisting of an input node X and an output node Y. At each iteration, a value is sampled from the input distribution p_X and copied to the output according to the conditional distribution $\pi_{Y|X}$. Following this copy operation, the input is updated with a new value, independently drawn from p_X, thereby resetting the joint state of (X, Y) from correlated to uncorrelated.

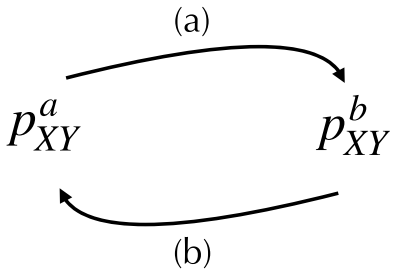


FIG. 4. Diagram illustrating how the joint distribution of the channel evolves during (a) the overwriting of the input and (b) the noisy copying of the input state to the output. Here, $p^a_{XY}(x, y) = p_X(x)p_Y(y)$ and $p^b_{XY}(x, y) = \pi_{Y|X}(y|x)p_X(x)$. With each use of the channel, the joint distribution cycles from p^b_{XY} to p^a_{XY} and back again.

process (see Appendix 1). In such processes, evolution of a joint system can be broken down into independent evolution of multiple subsystems [28,29]. Importantly, in subsystem processes the prior distribution over the joint system factorizes into the product of the marginals of the two subsystems. Specifically, let q^a_{XY} denote the prior for the input-overwriting process. Then, it can be expressed as a product distribution:

$$q^a_{XY}(x, y) = q^a_X(x)q^a_Y(y). \tag{29}$$

The new value of X is independently sampled from p_X. If the channel starts in q^a_{XY} before the input node is overwritten, then after the overwriting, the distribution of X is p_X and the resulting joint distribution is

$$\tilde{q}^a_{XY}(x, y) = p_X(x)q^a_Y(y). \tag{30}$$

The actual distribution over the states of the channel before and after the overwriting process are p^b_{XY} [Eq. (27)] and p^a_{XY} [Eq. (28)], respectively, while the prior distribution changes from q^a_{XY} to $\tilde{q}^a_{XY}$ as depicted in Fig. 5. Therefore, the mismatch cost associated with this overwriting process is given by

$$\mathrm{MC}^a(p_X) = D\big(p^b_{XY} \| q^a_{XY}\big) - D\big(p^a_{XY} \| \tilde{q}^a_{XY}\big), \tag{31}$$

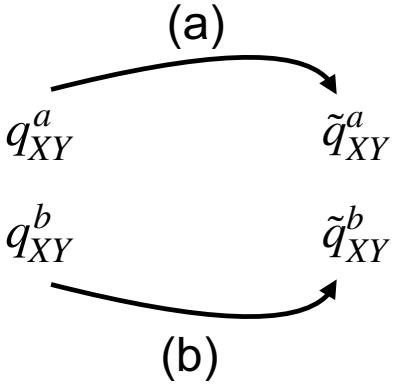


FIG. 5. Diagram illustrating how the prior distributions associated with (a) the overwriting process and (b) the noisy copying of the input state to the output evolve through their respective transformations. In panel (a), the prior changes from $q^a_{XY}(x, y) = q^a_X(x)q^a_Y(y)$ to $\tilde{q}^a_{XY}(x, y) = p_X(x)q^a_Y(y)$. In panel (b), the prior changes from $q^b_{XY}(x, y)$ to $\tilde{q}^b_{XY}(x, y) = \pi_{Y|X}(y|x)q^b_X(x)$.

which simplifies to

$$\mathrm{MC}^a(p_X) = \mathrm{I}(X;Y) + D\big(p_X \| q_X^a\big). \tag{32}$$

where $\mathrm{I}(X;Y)$ is the mutual information between X and Y before overwriting.

$$\mathrm{I}(X;Y) = \sum_{x,y\in X} p_{XY}^b(x,y) \log \frac{p_{XY}^b(x,y)}{p_X(x)p_Y(y)}. \tag{33}$$

The derivation is provided in Appendix 2.

Now, let us consider the mismatch cost associated with copying process. The value of the output depends on the value of the input $\pi_{Y|X}$. Therefore, although Y evolves while the state of X does not change, by definition it is not a subsystem process and the prior distribution cannot be expressed as a product distribution. Let q_{XY}^b denote some arbitrary joint prior distribution for the copying process. After the copy operation, q_{XY}^b evolves to

$$\tilde{q}_{XY}^b(x,y) = \pi_{Y|X}(y|x) q_X^b(x), \tag{34}$$

where $q_X^b(x) = \sum_y q_{XY}^b(x,y)$ is the marginal prior distribution of X. The actual distribution over the states of the channel before and after the copying process is p_{XY}^a [Eq. (28)] and p_{XY}^b [Eq. (27)], respectively. Therefore, the mismatch cost associated with copying operation is

$$\mathrm{MC}^b = D\big(p_{XY}^a \| q_{XY}^b\big) - D\big(p_{XY}^b \| \tilde{q}_{XY}^b\big), \tag{35}$$

which further simplifies to

$$\mathrm{MC}^b = D\big(p_X p_Y \| q_{XY}^b\big) - D\big(p_X \| q_X^b\big). \tag{36}$$

Combining Eqs. (32) and (36), the total mismatch cost is

$$\mathrm{MC}(p_X) = \mathrm{I}(X;Y) + D\big(p_X \| q_X^a\big) + D\big(p_X p_Y \| q_{XY}^b\big) - D\big(p_X \| q_X^b\big). \tag{37}$$

While $I(X;Y)$ quantifies the rate of information transmitted per channel use, $\mathrm{MC}(p_X)$ in Eq. (37) expresses the MMC per use of the communication channel, representing the unavoidable EP incurred with each symbol transmission.

The derivation of Eq. (37) does not rely on any specific assumptions about the physical details of the communication channel's implementation. In particular, it holds regardless of whether the underlying dynamics of steps (a) or (b) are Markovian or non-Markovian, discrete time or continuous time. The result follows directly from the general structure of the communication model, which consists of two fundamental operations: (1) overwriting the input (a subsystem process) and (2) performing a noisy copy to the output.

The only model-dependent quantities in Eq. (37) are the prior distributions, q_X^a and q_{XY}^b, which encapsulate the physical characteristics of the process. As described earlier in Eq. (24), these priors are determined by the entropy flow functions $f(x)$ and the computational map G. Once G is specified, assumptions about the entropy flow $f(x)$ allow one to determine the corresponding priors. In the next section, we illustrate this by deriving priors from given entropy flow profiles.

Furthermore, since KL divergence is always non-negative, $D(p_X \| q_X^a) \geqslant 0$, and by the data processing inequality, $D(p_X p_Y \| q_{XY}^b) - D(p_X \| q_X^b) \geqslant 0$ for all p_X, q_X^a, and q_{XY}^b. Therefore,

$$\mathrm{MC}(p_X) \geqslant \mathrm{I}(X;Y). \tag{38}$$

This inequality holds irrespective of the prior of the underlying process implementing the communication channel. The total cost in the communication channel is lower bounded by the mutual information, which quantifies the rate of information transmission.

B. Implications for work extraction via measurement and feedback

We briefly talk about the role of the minimal cost of communication in Eq. (38) to the work extraction under measurement-and-feedback control.

A generalized maximum work extraction formulation of the second law states that in an isothermal process the maximum extractable work between two distributions p_{X_0} and p_{X_1} is given by the change in nonequilibrium free energy [30],

$$\langle W \rangle \leqslant \mathcal{F}(p_{X_0}) - \mathcal{F}(p_{X_1}), \tag{39}$$

where $\mathcal{F}(p_{X_0}) = \langle E \rangle_{p_{X_0}} - k_B T S(p_{X_0})$. However, in the presence of measurement-and-feedback protocols; i.e., when there is a measurement of system's state involved before applying the protocol and the protocol depends on the outcome of the measurement, the maximum extractable work is the change in free energy plus the mutual information between system's state and the memory [31,32].

$$\langle W \rangle \leqslant \mathcal{F}_{E_0}(p_{X_0}) - \mathcal{F}_{E_1}(p_{X_1}) + \mathrm{I}(X;M), \tag{40}$$

where $\mathrm{I}(X;M)$ is the mutual information between system's state and the memory of the measuring device. If the accuracy of a measurement is described by the conditional distribution $\pi_{M|X}$, which gives the probability of observing outcome M given the system's state X, then the measurement process can be viewed as a communication (or observational) channel $\pi_{M|X}$. Equation (40) implies that it is possible to extract work exceeding the bound imposed by the second law [as given in Eq. (39)] by performing a measurement on the system and applying a feedback protocol that depends on the measurement outcome.

A canonical example of this phenomenon is Szilard's engine, which consists of a single-particle gas in a box. In this setup, a demon measures whether the particle is on the left or right side of a partition. Based on this information, the demon allows the piston to expand, extracting an amount of work equal to $k_B T \log 2$ [33,34].

By repeating this cycle, the demon can extract $k_B T \log 2$ of work per iteration—or more generally, $k_B T \mathrm{I}(X;M)$, where $\mathrm{I}(X;M)$ is the mutual information between the system and measurement outcomes. This seems to suggest the possibility of a perpetual motion machine of the second kind, apparently in violation of the second law.

Earlier efforts to reconcile Szilard's engine with the second law emphasized that, in a repeated setting, the step of resetting the demon's memory to a blank state is logically irreversible and incurs an associated Landauer's cost [35]. This cost precisely offsets the additional work extracted, thereby restoring consistency with the second law. More recent work

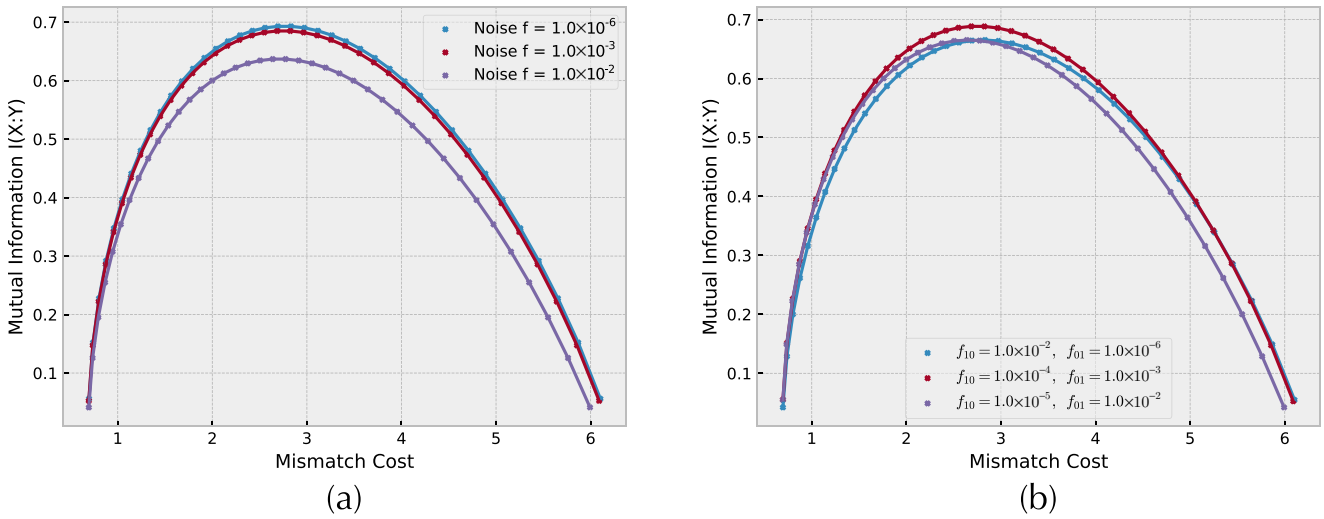


FIG. 6. Relationship between mismatch cost and transmitted information in a binary channel. The graphs are obtained by using Eq. (37). For the binary channel, the input distribution is parameterized as $p_X(x=0) = r$ and $p_X(x=1) = 1-r$. Each curve is generated by varying r from 0 to 1, with the resulting mutual information between X and Y plotted on the y axis and the corresponding total mismatch cost from Eq. (37) on the x axis. The overwriting prior q_X^a is taken to be uniform, while the prior for the copying process is set as $q_{XY}^b(0,0) = q_{XY}^b(0,1) = 0.498$ and $q_{XY}^b(1,0) = q_{XY}^b(1,1) = 0.002$. Distinct curves correspond to different noise levels in panel (a) symmetric channels and panel (b) asymmetric channels, parameterized by the conditional probabilities $\pi_{Y|X}(0|1) = f_{01}$ and $\pi_{Y|X}(1|0) = f_{10}$. In both cases, mutual information initially increases with mismatch cost, peaks at channel capacity, and then decreases. This concave-down relationship reflects diminishing returns: higher thermodynamic cost yields progressively smaller gains in information transmission.

has further deepened our understanding by showing that a loss of correlation between components of a system during a process contributes to the total EP. Specifically, the decrease in mutual information between subsystems during a process corresponds exactly to a positive contribution to the entropy production [29,36].

Consistent with prior work, Eq. (38) shows that there is a minimal thermodynamic cost associated with measurement in each cycle, and this cost is precisely given by the mutual information between the system's state X and the measurement device's memory M. Furthermore, Eq. (37) provides an even tighter lower bound on the thermodynamic cost—one that can exceed the mutual information. Together, these results demonstrate that a measurement-and-feedback control process necessarily dissipates at least as much, and potentially more, work than it extracts. This reinforces the thermodynamic consistency of information-driven processes and precludes the possibility of a perpetual motion machine of the second kind.

In the next section, we examine a binary communication channel and begin by making assumptions about the associated priors. These assumptions allow us to analyze the channel's thermodynamic cost without needing to explicitly define the underlying physical process. We will demonstrate that, for certain kinds of priors, the observed relationship aligns with the concave relationship between EP and communication rate that has been found in concrete physical models [37,38]. However, we also show that this concavity is not universal, as other relationships can arise depending on the prior.

C. Example of a binary channel

We consider a binary communication channel with input and output state spaces $\mathcal{X} = \mathcal{Y} = \{0,1\}$. For simplicity, we first assume that the prior distribution associated with the overwriting process, q_X^a in Eq. (37), is uniform. We begin by assuming that the prior associated with the copying operation satisfies $q_{XY}^b(0,0) = q_{XY}^b(0,1) \ll q_{XY}^b(1,0) = q_{XY}^b(1,1)$. As shown in Fig. 1, such an asymmetric prior-positioned near the edge of the simplex—reflects large values of the entropy flow terms. A sufficient condition for obtaining this type of prior is a communication channel where $f(x,y)$ values are significantly large, with $f(0,0) = f(0,1) \geqslant f(1,0) = f(1,1)$. The source distribution $p_X(x)$ is parameterized by a single variable p, where $p_X(0) = p$ and $p_X(1) = 1-p$. By varying p over the interval [0,1], we explore different source distributions. For each $p_X(x)$, we compute the mutual information between X and Y after the copy operation, and the mismatch cost using Eq. (37).

Figure 6 illustrates the relationship between mutual information and mismatch cost for both symmetric and asymmetric binary channels. Mutual information initially increases with mismatch cost, peaks at channel capacity, and then declines. This behavior arises because of how mutual information and mismatch cost vary with the input distribution parameter $p_X(x=1) = r$. As r varies from 0 to 1, the mutual information begins at zero, increases to a maximum at $r = r^*$ (with $r^* = 0.5$ for a binary symmetric channel), and then decreases again, reaching zero at $r = 1$. In contrast, the mismatch cost is minimized at an intermediate value of r depending on the prior distribution, but maximized at the boundaries (either at $r = 1$ or $r = 1$). Thus, when r approaches either boundary, the mutual information goes to zero while the mismatch cost is maximized. This behavior results in concave-down curves that reflect the law of diminishing returns: The increase in mutual information with increasing thermodynamic cost gets progressively smaller.

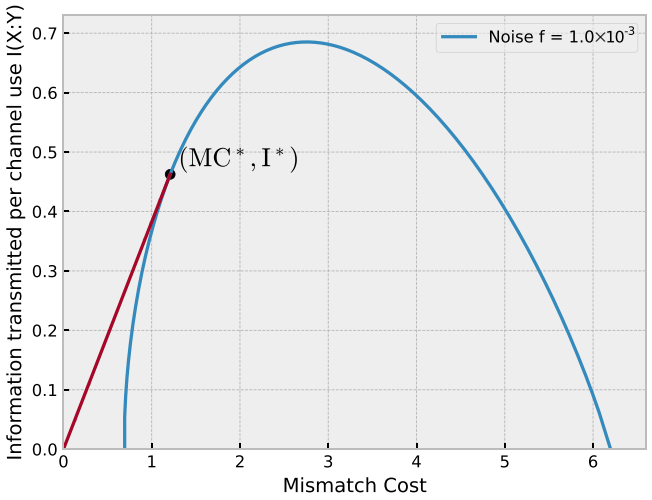


FIG. 7. Relationship between information transmitted and mismatch cost per channel use is concave downward, illustrating the law of diminishing returns. The point (MC*, I*) on the curve, determined by the tangent from the origin, is where the transmission rate to cost ratio is maximum.

It suggests that operating at channel capacity is thermodynamically inefficient. This principle—known as reverse multiplexing—shows that distributing cost across multiple channels can enhance overall information transfer efficiency [39,40].

For example, Fig. 7 shows an operating point (MC*, I*) where the ratio of information transmitted to mismatch cost is maximized. Beyond this point, the curve exhibits diminishing returns—any further increase in information transmission rate $\tilde{\mathrm{I}} > \mathrm{I}^*$ requires disproportionately higher mismatch cost per additional bit of information. In such cases, it is more thermodynamically efficient to split the communication across two identical channels, (X_1, Y_1) and (X_2, Y_2), rather than using a single channel at a higher rate. If the values of inputs X_1 and X_2 are sampled independently from p_{X_1} and p_{X_2}, respectively, the combined information transmitted across the two channels is simply the sum of the information transmitted through each channel individually:

$$\mathrm{I}(X_1, X_2; Y_1, Y_2) = \mathrm{I}(X_1; Y_1) + \mathrm{I}(X_2; Y_2). \tag{41}$$

In Appendix 3, we provide a formal proof that the MMC associated with communication over a joint channel—comprising two input nodes and two output nodes—is equal to the sum of the MMC incurred when each of the two input-output channels operates independently. That is, if the values of input nodes are sampled independently from distributions p_{X_1} and p_{X_2}, then the MMC incurred during communication over joint channel, $\mathrm{MC}_{12}(p_{X_1} p_{X_2}) = \mathrm{MC}_1(p_{X_1}) + \mathrm{MC}_2(p_{X_1})$. This result demonstrates the additivity of MMC under parallel composition of communication channels.

Since each channel operates at or below I*, where the information-to-cost ratio is higher, the combined mismatch cost $\mathrm{MC}_1 + \mathrm{MC}_2$ will be lower than the mismatch cost required to transmit the same total information through a single high-rate channel. This reflects the principle of reverse multiplexing, where distributing the load across multiple lower-rate channels reduces the overall thermodynamic cost.

However, this concave downward relationship does not hold for all choices of prior q^b_{XY}. For example, when both q^b_{XY} and q^a_X are uniform, the relationship between mutual information and mismatch cost is qualitatively different. As shown in Fig. 8(a), mismatch cost and rate of information transmission become inversely related: MMC is minimized when the rate of information transmission reaches its maximum (i.e., channel capacity), indicating that operating at capacity is thermodynamically efficient in this case. Conversely, mismatch cost is highest when information transmitted is minimal. The reason behind this is that in a symmetric noisy channel, information transmission is maximized when the source distribution is uniform. On the other hand, when the priors q^a_X and q^b_{XY} are uniform, mismatch cost is minimized, since $D(p_X \| q^a_X) = 0$ and $D(p_{XY} \| q^b_{XY})$ is minimum. As the source distribution deviates from uniformity, mutual information decreases while mismatch cost increases, yielding the inverse relationship shown in Fig. 8(a).

Interestingly, the relationship can behave quite differently under other priors. Figure 8(b), generated with an arbitrary choice of q^b_{XY} (while keeping q^a_X uniform), shows a different relationship between mismatch cost and mutual information.

We now turn to the mismatch cost associated with encoding and decoding protocols in communication processes. As discussed earlier, encoding and decoding form the backbone of modern communication theory and practice. Thus, a comprehensive understanding of the thermodynamic cost of communication must also account for the energetic costs of encoding and decoding.

Unlike the simple yet broadly applicable model of a noisy communication channel considered earlier, encoding and decoding are inherently complex computational tasks. There exist multiple ways to implement them—for instance, linear encoders can be realized through Boolean circuits, while even basic implementations of syndrome decoders involve nontrivial circuit architectures. This complexity makes it challenging to account for the thermodynamic cost incurred at every computational step.

To circumvent these difficulties, we adopt the framework of periodic machines, as introduced in Ref. [13]. Periodic machines provide a way to implement algorithms at either a high-level or low-level abstraction, designed to reflect the periodic, clock-driven nature of modern computing architectures (e.g., random access stored program (RASP)-based models). By design, periodic machines repeat a fixed physical process—analogous to the fetch-decode-execute cycles in stored-program computers. Importantly, these machines are tailored to the specific algorithm they implement. In the next section, we briefly review the properties of periodic mismatch cost and employ this framework to derive lower bounds on EP associated with encoding and decoding algorithms implemented on a periodic machine.

IV. MINIMAL THERMODYNAMIC COST OF LINEAR ERROR-CORRECTING CODES

A. Mismatch cost of algorithmic computation

In this section, we formally define the concept of a periodic machine [13], which iteratively executes the steps of a

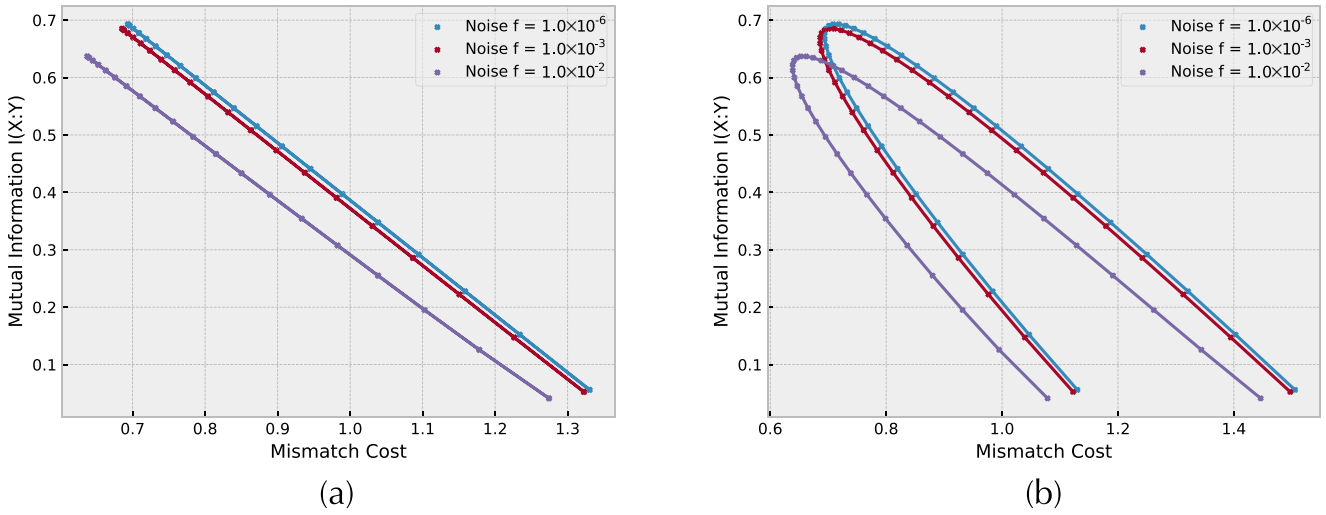


FIG. 8. Relationship between mismatch cost and mutual information in a binary channel for different prior distributions. (a) Both q_X^a and q_{XY}^b are uniform, leading to an inverse relationship: Mismatch cost is minimized when mutual information is maximized (at channel capacity). (b) q_X^a remains uniform, but q_{XY}^b is chosen asymmetrically with $q_{XY}^b(0,0) = 0.35$, $q_{XY}^b(0,1) = 0.22$, $q_{XY}^b(1,0) = 0.25$, and $q_{XY}^b(1,1) = 0.18$, resulting in a qualitatively different relationship. Once again, each curve is generated by varying input distribution parameter $p_X(x = 1) = r$ from 0 to 1, with the resulting mutual information between X and Y plotted on the y axis and the corresponding total mismatch cost from Eq. (37) on the x axis.

program. A program or algorithm is a sequence of instructions that systematically operate on variables, updating their values step by step as the computation progresses. All variables of an algorithm include input variables, internal variables such as flags, loop counters, and other control variables. A central component is the program counter, which tracks the current instruction being executed at any given moment. The state of the algorithm at any point during its execution is defined by the values of all its variables, including the current value of the program counter. Let X_i denote the random variable representing the state of the algorithm immediately after the ith algorithmic step, with possible joint values $\boldsymbol{x}_i \in \mathcal{X}_\mathcal{A}$. Here, $\mathcal{X}_\mathcal{A}$ is the set of all allowed joint states of the variables in the algorithm $\mathcal{A}$. Additionally, X_0 represents the random variable corresponding to the initial state $\boldsymbol{x}_0$ of the algorithm before execution begins.

A jump from any state $\boldsymbol{x}_i$ to a next $\boldsymbol{x}_{i+1}$ is fully specified by the algorithm. The transition is many-to-one or one-to-one in the case of a deterministic algorithm, whereas in a probabilistic algorithm, this transition is stochastic, and the next state is drawn from a probability distribution specified by the algorithm. In both cases, the progression from $\boldsymbol{x}_i$ to $\boldsymbol{x}_{i+1}$ can be captured by a map $G_\mathcal{A} : \mathcal{X}_\mathcal{A} \to \mathcal{X}_\mathcal{A}$, corresponding to algorithm which, given the current state $\boldsymbol{x}_i$, returns the next state $\boldsymbol{x}_{i+1}$:

$$\boldsymbol{x}_{i+1} = G_\mathcal{A}(\boldsymbol{x}_i). \tag{42}$$

When the algorithm halts, $G_\mathcal{A}$ reaches a fixed point. The dynamics of the algorithm induce a corresponding dynamics on the probability distribution over the state space $\mathcal{A}$. To describe this formally, we index each state $\boldsymbol{x}_i \in \mathcal{X}_\mathcal{A}$ and define a transition matrix G, where each element G_{nm} specifies the probability of transitioning from the state with index m to the state with index n. If state m does not transition to state n, set $G_{nm} = 0$. For deterministic algorithms, G_{nm} will be either 0 or 1, reflecting the deterministic nature of the state transitions. The discrete-time evolution of the probability distribution over the state space is governed by the stochastic map G:

$$p_{X_{i+1}} = G p_{X_i}, \tag{43}$$

where p_{X_i} represents the probability distribution of X_i. Equation (43) defines a discrete-time evolution of the probability distribution, where the same stochastic map G is applied at each iteration. Therefore, one can write

$$p_{X_i} = G^i p_{X_0}. \tag{44}$$

As the algorithm progresses and eventually halts, the distribution converges to a steady-state distribution.

Assuming that the underlying physical process implementing G remains unchanged across iterations, this corresponds to repeatedly applying the same physical process, driving the system through a sequence of probability distributions. Consequently, the prior distribution associated with the underlying process implementing the stochastic map G also remains the same across iterations. Let q_X denote the prior. After each iteration of the process, the ending distribution associated with the prior is $\tilde{q}_X = G q_X$, whereas the actual distribution after ith iteration is given by Eq. (44). Therefore, the MMC in ith iteration is given by

$$\mathrm{MC}^i(p_{X_0}) = D(G^{i-1} p_{X_0} \| q_X) - D(G^i p_{X_0} \| G q_X) \tag{45}$$

for any $i \in \{0, 1, ...\}$. Note that the prior distribution q_0 remains same across iterations. However, the actual state distribution evolves with each iteration. Therefore, if the algorithm takes r iteration to halt for all it's inputs, the total mismatch cost associated with an initial distribution p_{X_0} is

given by

$$\mathrm{MC}(p_{X_0}) = \sum_{i=1}^{r} \mathrm{MC}^i(p_{X_0}) = \sum_{i=1}^{r} [D(G^{i-1} p_{X_0} \| q_X) - D(G^i p_{X_0} \| G q_X)]. \quad (46)$$

Even if the process starts at the prior q_0, after the first iteration, the distribution becomes $p_1 = Gq_0$, which differs from q_0. This deviation from the prior distribution results in a strictly positive MMC in the next iteration, and the same holds for subsequent iterations [41].

Initial state of the algorithm and resetting cost

Before going further, let us briefly discuss how the initial distribution for an algorithm is properly defined. When assigning an initial distribution p_{X_0}, it is crucial to understand the nature of the variables involved. An algorithm starts with its program counter set to 0 and includes input and noninput variables. Noninput variables may consist of loop counters, flags, and other special variables that are initialized to specific starting values. Let X_{in}, X_{nin}, and X_{sp} denote the random variables representing the states of input variables, noninput variables, and special variables, with values x_{in}, x_{nin}, and x_{sp}, respectively. We assume that the input variables are freshly initialized before each run by sampling from a distribution $p_{X_{\mathrm{in}}}$. The special variables—such as the program counter and loop counters—are always reset to their initialized states at the start of each run. This is represented by a deterministic distribution over their initial values:

$$p_{X_{\mathrm{sp}}}(x_{\mathrm{sp}}) = \delta_{X_{\mathrm{sp}}}(x_{\mathrm{sp}}), \quad (47)$$

where $\delta_{X_{\mathrm{sp}}}(x_{\mathrm{sp}})$ is the Kronecker delta, ensuring that the special variables are set to their predefined initial states with probability 1. In contrast, the noninput variables are assumed to retain their values from the end of the previous execution of the algorithm. That is,

$$p_{X_{\mathrm{nin}}}(x_{\mathrm{nin}}) = \sum_{x_{\mathrm{in}}, x_{\mathrm{sp}}} p_{X_r}(\boldsymbol{x}), \quad (48)$$

where $p_{X_r}(\boldsymbol{x})$ is the joint distribution over all variables at the end of the previous run. Thus, after reinitialization, the joint distribution over the algorithm's state is given by

$$p_{X_0}(\boldsymbol{x}) = p_{X_{\mathrm{in}}}(x_{\mathrm{in}})\, \delta_{X_{\mathrm{sp}}}(x_{\mathrm{sp}})\, p_{X_{\mathrm{nin}}}(x_{\mathrm{nin}}). \quad (49)$$

This transition of the distribution from p_{X_r} to p_{X_0}, which occurs due to the reinitialization before the start of a new run, incurs a mismatch cost. The MMC associated with this reinitialization step is given by

$$\mathrm{MC}^a = D\big(p_{X_r} \| q^a_{X_f}\big) - D\big(p_{X_0} \| \tilde{q}^a_{X_f}\big), \quad (50)$$

where $q^a_{X_f}$ is the prior distribution associated with the overwriting process. After the input and special variables are reinitialized, the prior evolves to

$$\tilde{q}^a_{X_f}(\boldsymbol{x}) = p_{X_{\mathrm{in}}}(x_{\mathrm{in}}) \delta_{X_{\mathrm{sp}}}(x_{\mathrm{sp}}) q^a_{X_{\mathrm{nin}}}(x_{\mathrm{nin}}), \quad (51)$$

where $q^a_{X_{\mathrm{nin}}}(x_{\mathrm{nin}}) = \sum_{x_{\mathrm{in}}, x_{\mathrm{sp}}} q^a_{X_f}(\boldsymbol{x})$ is the marginal distribution of X_{nin}.

```
def LinearEncoder(t):
  for c in range(0, n-k):
    t[c+k] = (P[c]@ t[:k])%2
  return t
```

FIG. 9. Python script of a linear encoding program. The first k bits of the vector $\mathbf{t}$ correspond to the input string, while the remaining bits are parity bits computed during program execution using the parity check matrix P. The variable i serves as a loop counter, initialized to $i = 0$ at the start of the program.

Given this, the mismatch cost simplifies to

$$\mathrm{MC}^a = D\big(p_{X_r} \| q^a_{X_r}\big) - D\big(p_{X_{\mathrm{nin}}} \| q^a_{X_{\mathrm{nin}}}\big). \quad (52)$$

The above consideration of mismatch cost of setting new values before reusing the computational machine extends to any physical device that involves overwriting.

In the next couple of sections, we compute the mismatch cost associated with linear encoding and syndrome decoding algorithm.

B. Mismatch cost of linear encoding algorithm

Consider the linear encoding algorithm illustrated in Fig. 9. It is an (n, k) linear encoder with an associated parity-check matrix P,

$$P = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}. \quad (53)$$

The algorithm takes as input an n-bit tape $\mathbf{t}$, where the first k bits represent the input message $\mathbf{s}$, and the remaining $n - k$ bits are reserved for the parity bits. Inside the for-loop, the algorithm iteratively computes the cth parity bit and writes it to the $(c + k)$th position of $\mathbf{t}$. Values of $\mathbf{t}$ and the loop counter c together constitute the state of the algorithm.

To compute the MMC of this encoding algorithm, we construct the stochastic map G_{E} associated with the algorithm, as described in Sec. IV A. We assume that the k-bit input message $\mathbf{s}$ is sampled from an i.i.d. Bernoulli distribution with bias α. That is,

$$p_{X_{\mathrm{in}}}(\mathbf{s}) = \alpha^{\#_1(s)}(1 - \alpha)^{\#_0(s)}, \quad (54)$$

where $\#_1(s)$ and $\#_0(s)$ denote the number of 1s and 0s in $\mathbf{s}$, respectively. The remaining $n - k$ bits of $\mathbf{t}$ retain their values from the previous execution of the algorithm, while the loop counter is reinitialized to $c = 0$. This defines the initial distribution p_{X_0}, as discussed in Sec. IV A 1.

After the ith step in the program, which corresponds to ith iteration of the for-loop that computes the ith parity bit, the distribution over the state of the algorithm is given by

$$p_{X_i} = G^i_{\mathrm{E}} p_{X_0}. \quad (55)$$

To compute the mismatch cost at each step, we assume that the prior distribution associated with the physical process implementing G_{E} is uniform. Then, the mismatch cost incurred

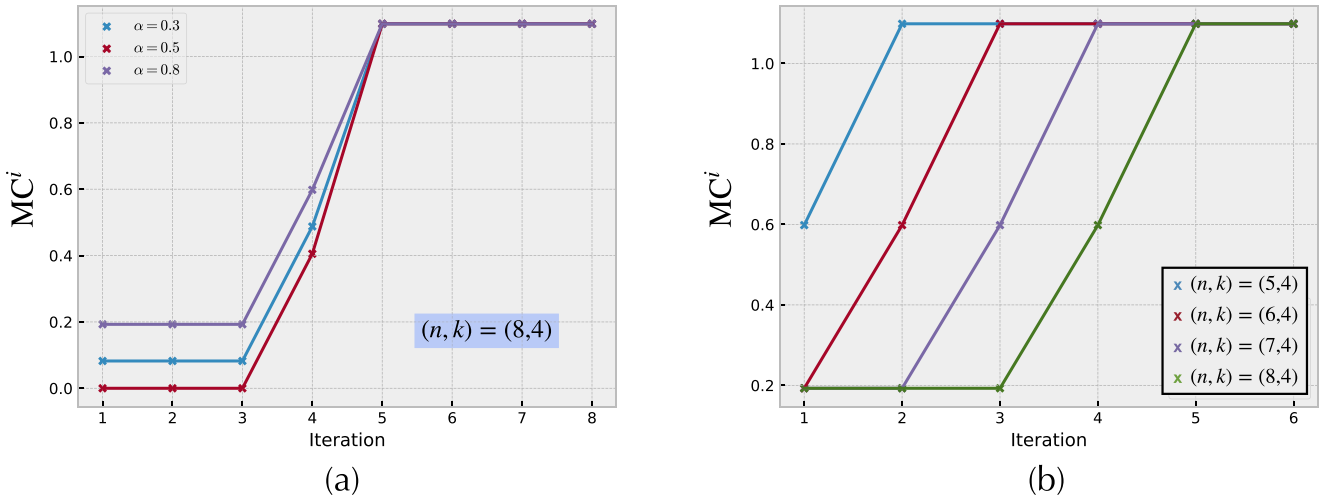


FIG. 10. Mismatch cost incurred in each iteration of the map G_{E} for panel (a) an (8, 4) encoder. The input strings to the encoder are drawn from an i.i.d. Bernoulli distribution given in Eq. (54), with Bernoulli parameter α. (b) Comparison of MMC incurred in each iteration across different encoders, where the encoding algorithms vary by the number of additional parity check bits. The input strings are sampled from a distribution with parameter $\alpha = 0.80$ in panel (b).

in the ith iteration is given by

$$\mathrm{MC}^i(p_{X_0}) = D\big(G_{\mathrm{E}}^{i-1} p_{X_0} \| q_X\big) - D\big(G_{\mathrm{E}}^{i} p_{X_0} \| G_{\mathrm{E}} q_X\big). \tag{56}$$

Completing the program requires $n-k$ iterations of the map G_{E}. Therefore, the total MMC incurred by an (n, k) linear encoding algorithm is given by

$$\mathrm{MC}_E(p_{X_0}) = \sum_{i=1}^{n-k} \big[D\big(G_{\mathrm{E}}^{i-1} p_{X_0} \,\|\, q_X\big) - D\big(G_{\mathrm{E}}^{i} p_{X_0} \,\|\, G_{\mathrm{E}} q_X\big)\big]. \tag{57}$$

Figure 10(a) shows the mismatch cost incurred in each iteration of a $(8, 4)$ linear encoding algorithm for various values of the Bernoulli parameter α of the input distribution. For a given α, the mismatch cost remains constant across the first three iterations, which correspond to the computation of the first three parity bits. However, it increases during the computation of the fourth parity bit. In the fifth iteration, the algorithm halts, the system's dynamics reach a steady state, and the mismatch cost stabilizes.

Figure 10(b) compares the per-iteration MMC across different linear encoders, each with the same input length $k = 4$ but varying numbers of appended parity bits. Interestingly, despite differences in total iterations and internal state space (e.g., tape length and loop counter size), certain patterns persist. For example, the MMC associated with computing the final parity bit is identical across encoders, whether it occurs in the first iteration of a $(5, 4)$ encoder or the fourth in a $(8, 4)$ encoder. Similar consistency appears for the second-to-last parity bit, third-to-last, and so on. Moreover, once the steady state is reached, all encoders exhibit the same constant MMC.

Figure 11 shows the total mismatch cost, computed by summing over all iterations before the algorithm halts [as defined in Eq. (57)], plotted against the number of parity bits for different values of the Bernoulli parameter α of the input distribution. The results indicate that the total MMC grows linearly with the number of parity bits. This linear trend arises because each additional parity bit contributes a roughly constant mismatch cost, as observed in Fig. 10(b). The magnitude of this cost depends on the input distribution: The further α deviates from the uniform distribution, the higher the mismatch cost, as shown in Fig. 10(a).

In the next section, we perform a similar analysis for syndrome decoders.

C. Mismatch cost of syndrome decoding algorithm

We provide a high-level pseudocode in Fig. 12 for a syndrome decoder, which operates on three main variables: **y**, **z**, and **e**, along with a program counter denoted as pc. Here, **y** is the n-bit string received from the channel, **z** is the computed syndrome, and **e** is the error vector. The program counter pc takes values 1, 2, and 3, corresponding to the three main steps of the decoding process: (1) computing the syndrome,

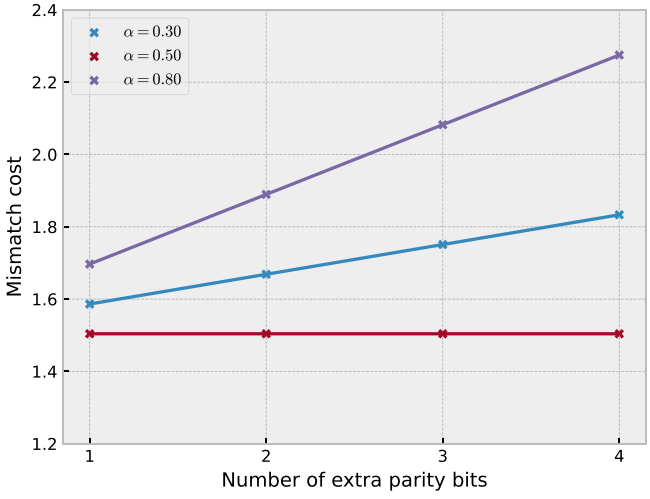


FIG. 11. Total mismatch cost ($\mathrm{MC}_E = \sum_i \mathrm{MC}_E^i$) of the encoding algorithm for different encoding lengths. Here again, $k = 4$ is fixed and n varies from 5 to 8. α is again the Bernoulli parameter of the probability distribution of the input strings [see Eq. (54)].

```python
def SyndromeDecoder(y):
  z = H@y          #Calculating syndrome [Code line 1: Matrix Multiplication]
  e = Table[z]     #look-up table [Code line 2: Searching an element in the table]
  y = (y+e)%2      #estimated output [Code line 3]
  return y
```

FIG. 12. Python script for syndrome decoder. For a received $\mathbf{y}$, the first step involves computing its syndrome $\mathbf{s}$ by using the parity check matrix H. Based on the syndrome, error vector $\mathbf{e}$ is estimated from a precomputed lookup table in the second step. The third step involves correcting the received vector by adding the estimated error vector.

(2) using a precomputed lookup table to identify the most probable error vector, and (3) correcting the received vector by adding the estimated error vector. Together, these variables define the state of the syndrome decoding algorithm, denoted as $(\mathbf{y}, \mathbf{z}, \mathbf{e}, \text{pc})$.

The stochastic map G_D associated with the decoding algorithm is constructed, as described in Sec. IV A. We use Y_i to denote the random variable representing the state of the decoder immediately after the ith iteration of the map. Additionally, Y_0 denotes the random variable representing the initial state of syndrome decoder, which is chosen as described in Sec. IV A 1: The program counter is set to 1 with probability 1; the noninput variables $\mathbf{z}$ and $\mathbf{e}$ retain their values from the end of the previous execution; and the input variable $\mathbf{y}$ is freshly initialized by sampling from $p_{Y_{\text{in}}}(\mathbf{y})$, the distribution over received strings induced by the channel. $p_{Y_{\text{in}}}(\mathbf{y})$ is obtained from the input distribution to the encoder. Specifically, $p_{X_{\text{in}}}(\mathbf{s})$, defined in Eq. (54) for encoder, induces a distribution over the encoded codewords, which are transmitted through the noisy channel, thereby inducing a corresponding distribution over the received strings at the decoder input. Let $p_{Y_{\text{in}}}(\mathbf{y})$ denote the induced distribution over input to the decoder.

To compute MMC asscoiated with the decoder, we assume that the prior distribution q_X for the decoding algorithm is uniform. MMC incurred in ith step of the decoding algorithm is given by

$$\text{MC}_D^i(p_{Y_0}) = D\big(G_D^{i-1} p_{Y_0} | q_Y\big) - D\big(G_D^i p_{Y_0} | G_D q_Y\big). \tag{58}$$

Figure 13(a) illustrates the mismatch cost in each iteration of a $(7, 4)$ syndrome decoder for various values of the Bernoulli parameter α of the input distribution. Since the decoding algorithm halts after three steps and the distribution p_{Y_i} reaches a steady state, the mismatch cost stabilizes beyond the third iteration. In contrast to the encoding algorithm—where each step simply computes a parity bit—the computations performed during decoding differ across steps, resulting in significant variation in the mismatch cost. In particular, the second step, which updates $\mathbf{e}$ using a lookup table, incurs the highest mismatch cost.

Figure 13(b) compares the per-iteration MMC across various $(n, 4)$ syndrome decoders, which differ in the value of n—that is, the length of the received n-bit string after a $k = 4$ bit input was encoded into an n-bit codeword and transmitted through a noisy channel. All decoders reach a steady state after three iterations, independent of n. For larger n, the MMC in each decoding step is higher due to the greater number of parity bits involved. However, once the steady state is reached, the per-iteration MMC becomes identical across decoders.

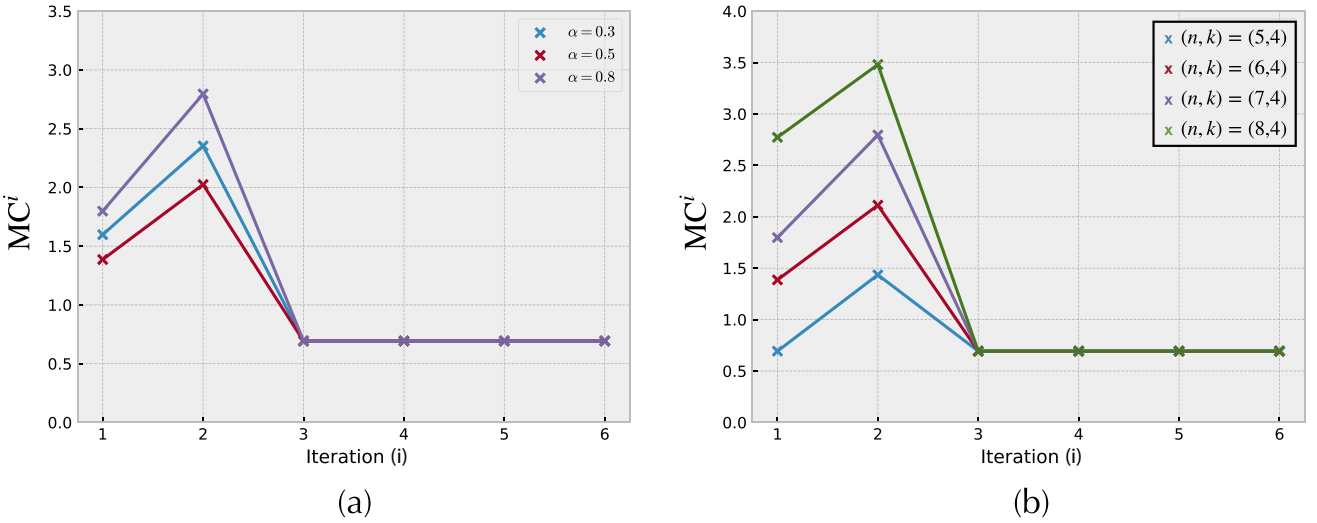


FIG. 13. (a) Mismatch cost in each step of a $(7, 4)$ syndrome decoder. The ith iteration of G_D corresponds to the execution of the ith step of the syndrome decoding algorithm [Eq. (12)]. The input string to the encoder is sampled from a distribution with Bernoulli parameter α. The encoded strings are transmitted through a binary symmetric noisy channel with noise level $f = 0.001$. The strings $\mathbf{y}$ received at the decoder have a distribution that varies with changes in α. (b) Mismatch cost in each iteration of the map G_D across decoders differing in the number of extra parity check bits $n - k$ used for decoding. Here, $k = 4$ and $\alpha = 0.80$.

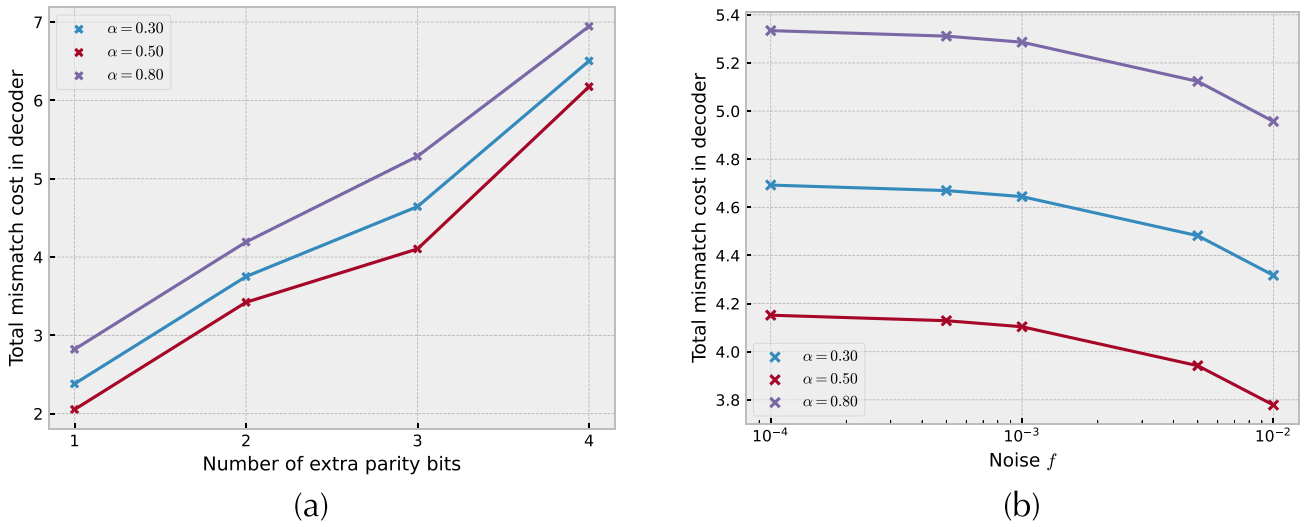


FIG. 14. (a) Total mismatch cost, $\mathrm{MC} = \sum_{i=1}^{3} \mathrm{MC}^i$, plotted against the number of parity bits $n-k$ in the codewords that the syndrome receives and decodes for different values of Bernoulli parameter α. Here, $k=4$ and noise in the channel is symmetric with $\pi_{0|1} = \pi_{1|0} = 0.001$. (b) Total mismatch cost of a (7, 4) syndrome decoder for various noise in the binary symmetric channel.

The total mismatch cost of syndrome decoder is obtained by summing over the three steps:

$$\mathrm{MC}_D(p_{Y_0}) = \sum_{k=1}^{3} D\big(G_D^{k-1} p_{Y_0} \| q_Y\big) - D\big(G_D^k p_{Y_0} \| G_D q_Y\big). \tag{59}$$

Figure 14(a) plots the total mismatch cost as a function of the number of parity bits $n-k$ in the codewords that the syndrome decoder receives and decodes. As expected, the total mismatch cost increases with the number of parity bits the decoder operates on. However, unlike the encoder case, this increase is not linear in n.

We also examine how the mismatch cost varies with channel noise. Figure 14(b) shows that the total mismatch cost of a (7, 4) syndrome decoder decreases as the noise in a binary symmetric channel increases. This is because higher noise leads to a broader distribution over received strings $P_{Y_{\mathrm{in}}}(\mathbf{y})$, making the induced distribution over the decoder's states closer to the assumed uniform prior. As a result, the mismatch cost, which quantifies deviation from the prior, decreases.

Finally, we combine the MMC of the encoder, the channel, and the decoder. The process begins with a source that i.i.d. generates messages from distribution $p_{X_{\mathrm{in}}}$. These messages are passed through the encoder, producing encoded bit-strings that are transmitted through the channel one bit at a time. The encoded strings induce a distribution over the input node of the channel, denoted p_X. After transmission, the channel outputs distorted strings, which are received by the decoder and follow a distribution $p_{Y_{\mathrm{in}}}$, which are then decoded (see Fig. 15). Three sequential three sequential components in the protocol—encoder, channel, and decoder—each repeatedly reused and incurring a mismatch cost: $\mathrm{MC}_E(p_{X_{\mathrm{in}}})$, $\mathrm{MC}_C(p_X)$, and $\mathrm{MC}_D(p_{Y_{\mathrm{in}}})$, respectively.

The protocol operates cyclically: Bit-strings generated by the source are passed through the encoder, transmitted through the channel, and then processed by the decoder. After each run, new bit-strings are generated, and the cycle repeats, with the inputs to the encoder, channel, and decoder being reinitialized sequentially. This repeated use introduces correlations between adjacent components (e.g., between the encoder output and the channel input) that are created during the forward pass and then broken during reinitialization. The resulting loss of mutual information between components in each cycle contributes to the total EP, over and above the sum of the mismatch costs of individual components. However, in this work, we do not analyze this intercomponent contribution and focus solely on the sum of the mismatch costs from the encoder, channel, and decoder: $\mathrm{MC}_E(p_{X_{\mathrm{in}}}) + \mathrm{MC}_C(p_X) + \mathrm{MC}_D(p_{Y_{\mathrm{in}}})$.

Figure 16(a) shows how the total end-to-end mismatch cost increases with the number of parity bits used in the encoding and decoding process, for a fixed noisy channel and a given Bernoulli input distribution with parameter α [see Eq. (54)]. As expected, using more parity bits improves error correction,

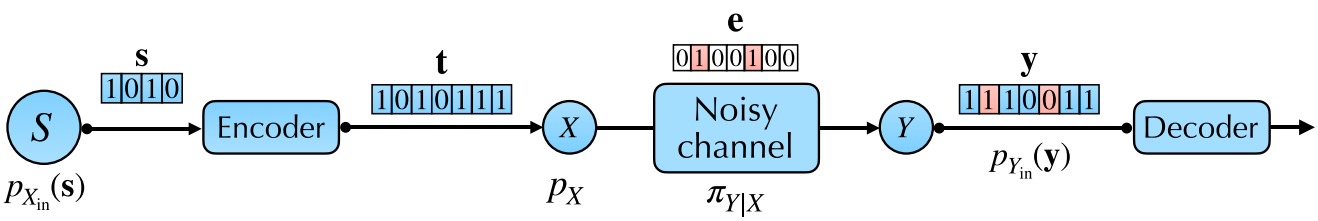


FIG. 15. Full communication protocol: The source S independently generates k-bit strings from a distribution $p_{X_{\mathrm{in}}}$, which are encoded into n-bit codewords. Each encoded string is transmitted through a noisy channel one bit at a time, inducing a distribution p_X at the channel input.

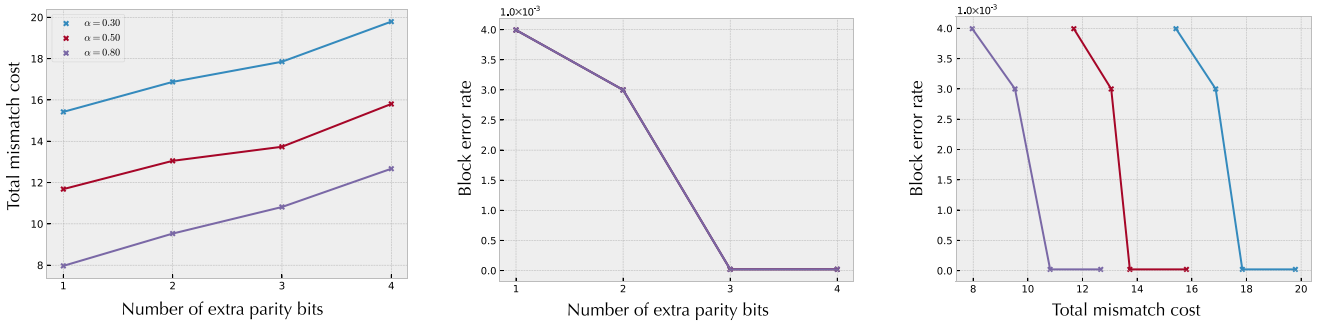


FIG. 16. (a) Total mismatch cost in an end-to-end communication protocol: $\mathrm{MC}_E(p_{X_{\mathrm{in}}}) + \mathrm{MC}_C(p_X) + \mathrm{MC}_D(p_{Y_{\mathrm{in}}})$. Here, $k = 4$ and noise in the channel is symmetric with $\pi_{0|1} = \pi_{1|0} = 0.001$ and α is the Bernoulli parameter of input distribution $p_{X_{\mathrm{in}}}$. (b) End-to-end block error rate decreases with the increasing number of parity check bits. (c) End-to-end block error rate vs total mismatch cost. Lowering block error rate requires a higher total MMC.

resulting in a lower end-to-end block error rate [Fig. 16(b)]. Consequently, Fig. 16(c) shows that block error rate decreases as total mismatch cost increases.

V. DISCUSSION AND FUTURE DIRECTIONS

In this paper, we have examined the MMC contribution to the thermodynamic cost of communication and, in doing so, established a minimal cost associated with communication channels. Our analysis centers on a general model of channel operation—namely, a repeated cycle of copy-reset-copy steps—which is both natural and ubiquitous across physical and engineered systems. Crucially, we showed that the minimal thermodynamic cost per communication cycle is precisely equal to the amount of information transmitted in that cycle. This result is universal: It holds independently of the specific physical implementation of the communication process.

Given the generality of the framework, we believe that the methods and results developed in this work can offer insight into a wide range of communication systems beyond engineered channels. In particular, they may help illuminate the fundamental thermodynamic costs of communication in biological systems—such as neural information processing in the brain, where neurons continuously exchange signals to process information [39,40], or in cellular signaling networks that transmit biochemical messages [27,42], often at considerable thermodynamic cost.

In addition, our analysis may have implications for communication complexity theory and the study of distributed systems. Traditionally, communication complexity is quantified in terms of the number of messages exchanged or the number of communication rounds required to perform a distributed task [43–46]. We suggest that incorporating thermodynamic cost—as formalized through MMC—into this framework could enrich the field, potentially leading to new insights into the design of communication-efficient and energy-efficient distributed algorithms, as well as optimal network topologies. Additionally, we hope that our analysis will be relevant to the field of neuromorphic computing—an emerging computational architecture that, among other aims, seeks to overcome the substantial energetic cost associated with communication between memory and processor that traditional von Neumann architectures suffer from Refs. [47,48].

With that said, we believe that there is substantial room for future exploration and refinement. In this paper, we have exclusively focused on i.i.d. information sources, and the minimal thermodynamic costs we derive apply strictly under that assumption. However, many scenarios involve correlated signals, such as those generated by first- or second-order Markov processes. A natural question then arises: How does the minimal cost of communicating each bit change when the source exhibits memory or temporal correlations? Understanding this could yield a deeper theory of thermodynamic costs for correlated or memory-bearing sources.

Furthermore, as emphasized in our discussion of reverse multiplexing, the relationship between mismatch cost and communication rate is not universally concave—it depends critically on the choice of prior distribution. This observation opens up two broad avenues for future work: (1) For what classes of prior distributions does the mismatch cost exhibit concavity with respect to communication rate, and what constraints does this impose on the detail of the underlying process implementing that communication channel? (2) Beyond concavity, what other forms can this relationship take for different priors, and what novel communication strategies—analogous to, but distinct from, reverse multiplexing—do these suggest?

We observed that the stepwise MMC incurred in both the linear encoder and the syndrome decoder increases with the length of the encoded string. However, this growth behaves differently for the two components: It is linear for the encoder, but nonlinear for the decoder.

The linear growth in the encoder's MMC reveals an important insight: As the encoding length increases, so does the number of computational steps—each parity bit adds a step. The fact that each additional step contributes a roughly constant increase in MMC, regardless of the expanding state space, suggests that the cost is primarily dictated by the nature of the computational step itself, which in this case is the computation of the parity check bit, rather than the size of the underlying state space.

The decoding algorithm exhibits more complex behavior. As the encoding length grows, not only does the state space of the decoder grow, but so does the nature of its computation. For example, when calculating the syndrome using matrix multiplication, the dimensions of the matrices change with

the length of the received string **y**, resulting in variations in the underlying computation. Consequently, the total MMC of syndrome decoder does not exhibit a linear growth with increasing length of the encoded string.

Additionally, while our numerical results consistently show that the second step of the decoding process dominates the total mismatch cost, we currently lack a clear analytical intuition for this behavior. This remains an interesting open question and points to a need for further investigation into the fine-grained structure of the decoder's thermodynamic costs.

As emphasized earlier, the stepwise MMC for the encoder and decoder presented in this work is derived under the assumption of a uniform prior distribution. This choice is intended purely for illustrative purposes; the results are not meant to imply universality, but rather to demonstrate that a detailed step-by-step analysis of minimal thermodynamic cost is feasible—even in the absence of an explicit physical model of the computational process.

Importantly, the periodic machine framework employed in this study can be readily applied to analyze the minimal costs associated with other encoding and decoding protocols, such as turbo codes or low density parity check (LDPC) codes. However, a significant open question remains: Are there universal features of minimal thermodynamic costs for encoding and decoding that hold irrespective of the prior distribution, depending only on the structure or complexity of the protocol itself?

A potential direction for discovering prior-independent minimal costs involves analyzing the structure of the periodic mismatch cost itself. Consider Eq. (46), which defines the total mismatch cost for a given prior q_X:

$$\mathrm{MC}_{q_X}(p_{X_0}) = \sum_{i=1}^{r} [D(G^{i-1}p_{X_0}\|q_X) - D(G^i p_{X_0}\|Gq_X)]. \quad (60)$$

As the system evolves through a sequence of state distributions $\{p_{X_0}, p_{X_1}, \ldots, p_{X_r}\}$ under the update rule $p_{X_{t+1}} = Gp_{X_t}$, one can find a distribution $\hat{q}_X$ that minimizes of the sum on the right-hand side, and yields a special MMC value, denoted $\mathrm{MC}_{\hat{q}_X}(p_{X_0})$, that serves as a strictly positive lower bound on the MMC incurred for any other choice of prior q_X.

While this observation suggests a promising direction for deriving prior-independent minimal thermodynamic cost, we do not pursue it further in this paper and leave a detailed development for future work.

ACKNOWLEDGMENTS

This work was supported by the U.S. National Science Foundation (NSF) Grant No. 2221345. We would like to thank Francesco Caravelli for helpful discussions. We would also like to thank the Santa Fe Institute for helping to support this research.

DATA AVAILABILITY

The data that support the findings of this article are openly available [49].

APPENDIX: DERIVATIONS

1. Subsystem processes

Consider a system composed of two subsystems A and B with state space $\mathcal{X}$ and $\mathcal{Y}$, respectively. A process is called a *subsystem process* [28,29] if (1) for any initial distribution $p_{XY}(x, y)$ over the joint state space, the following two conditions are satified:

(1) Two subsystems evolve independently of each other, i.e.,

$$G_{AB}(x_1, y_1|x_0, y_0) = G_A(x_1|x_0)G_B(y_1|y_0). \quad (A1)$$

(2) Total entropy flow of the joint system, denoted as Q_{AB} is the sum of the entropy flow of the subsystems,

$$Q_{AB}(p_{XY}) = Q_A(p_X) + Q_B(p_B). \quad (A2)$$

According to Eq. (18), the total EP of the joint system is given by

$$\mathcal{C}(p_{XY}) = Q(p_{XY}) + S(G_{AB}p_{XY}) - S(p_{XY}). \quad (A3)$$

On the other hand, EP of individual subsystems can be written as

$$\mathcal{C}_A(p_X) = Q^A(p_0^A) + S(G^A p_0^A) - S(p_0^A), \quad (A4)$$

and likewise for $\mathcal{C}_B$. Therefore, we can write Eq. (A3) as

$$\begin{aligned}\mathcal{C} &= \mathcal{C}_A + \mathcal{C}_B + (S(p_X) + S(p_Y) - S(p_{XY})) \\ &\quad - (S(G_A p_X) + S(G_B p_Y) - S(G_{AB}p_{XY})), \quad (A5)\\ &= \mathcal{C}_A + \mathcal{C}_B + I_0(X;Y) - I_1(X;Y), \quad (A6)\\ &= \mathcal{C}_A + \mathcal{C}_B + \Delta\mathrm{I}(X;Y), \quad (A7)\end{aligned}$$

where $\Delta\mathrm{I}(X;Y) = \mathrm{I}_0(X;Y) - \mathrm{I}_1(X;Y)$ denotes the drop in mutual information between X and Y from the initial to the final distribution. By the data processing inequality, mutual information cannot increase under local processing: $\mathrm{I}_0(X;Y) \geqslant \mathrm{I}_1(X;Y)$. Consequently, in a subsystem process, even if each subsystem evolves thermodynamically reversibly—i.e., with zero entropy production in each part ($\mathcal{C}_X = \mathcal{C}_Y = 0$)—the joint system may still exhibit a positive total entropy production due to a loss in mutual information between the subsystems. Specifically, if the initial correlation between subsystems X and Y is lost during the process, then the drop in mutual information contributes positively to total EP:

$$\mathcal{C} \geqslant \Delta\mathrm{I}(X : Y). \quad (A8)$$

However, if the initial joint distribution is uncorrelated—i.e., a product distribution—then $\mathrm{I}_0(X;Y) = 0$, and hence $\Delta\mathrm{I}(X;Y) = 0$. In this case, if q_X and q_Y are the prior distributions minimizing the entropy production in the X and Y subsystems, respectively, then the joint prior minimizing the total EP is simply their product:

$$q_{XY}(x, y) = q_X(x)q_Y(y). \quad (A9)$$

■

2. Derivation of Eqs. (32) and (36)

During the input-overwriting process, the actual joint distribution evolves from the correlated form $p^b_{XY}(x, y)$

$= \pi_{Y|X}(y|x)p_X(x)$ to the product form $p^a_{XY}(x,y) = p_X(x)p_Y(y)$:

$$\pi_{Y|X}(y|x)p_X(x) \xrightarrow{(a)} p_X(x)p_Y(y). \tag{A10}$$

In parallel, the prior distribution evolves from $q^a_{XY}(x,y) = q^a_X(x)q^a_Y(y)$ to $\tilde{q}^a_{XY}(x,y) = p_X(x)q^a_Y(y)$:

$$q^a_X(x)q^a_Y(y) \xrightarrow{(a)} p_X(x)q^a_Y(y). \tag{A11}$$

Thus, the corresponding drop in KL divergence during the process is given by

$$\mathrm{MC}^a(p_X) = D\big(p^b_{XY}\|q^a_{XY}\big) - D\big(p^a_{XY}\|\tilde{q}^a_{XY}\big), \tag{A12}$$

$$= \sum_{x,y} \pi_{Y|X}(y|x)p_X(x) \ln \frac{\pi_{Y|X}(y|x)p_X(x)}{q^a_X(x)q^a_Y(y)}$$

$$- \sum_{x,y} p_X(x)p_Y(y) \ln \frac{p_X(x)p_Y(y)}{p_X(x)q^a_Y(y)}, \tag{A13}$$

$$= \sum_{x,y} \pi_{Y|X}(y|x)p_X(x) \ln \frac{\pi_{Y|X}(y|x)p_X(x)}{p_X(x)p_Y(y)}$$

$$+ \sum_{x,y} \pi_{Y|X}(y|x)p_X(x) \ln \frac{p_X(x)p_Y(y)}{q^a_X(x)q^a_Y(y)}$$

$$- \sum_{x,y} p_X(x)p_Y(y) \ln \frac{p_Y(y)}{q^a_Y(y)}. \tag{A14}$$

In the third equality, the first term corresponds to the mutual information between X and Y, while the second term decomposes into the sum of two KL divergences: one between p_X and q^a_X, and the other between p_Y and q^a_Y. The third term is itself a KL divergence between p_Y and q^a_Y. Therefore, the contribution involving p_Y and q^a_Y cancels out, leaving,

$$\mathrm{MC}^a(p_X) = \mathrm{I}(X;Y) + D\big(p_X\|q^a_X\big) + D\big(p_Y\|q^a_Y\big) - D\big(p_Y\|q^a_Y\big), \tag{A15}$$

$$= \mathrm{I}(X;Y) + D\big(p_X\|q^a_X\big). \tag{A16}$$

Now, during the noisy copy operation, the actual distribution transforms from $p^a_{XY}(x,y) = p_X(x)p_Y(y)$ to $p^b_{XY}(x,y) = \pi_{Y|X}(y|x)p_X(x)$,

$$p_X(x)p_Y(y) \xrightarrow{(b)} \pi_{Y|X}(y|x)p_X(x). \tag{A17}$$

while the prior distribution evolves from $q^b_{XY}(x,y)$ to $\tilde{q}^b_{XY}(x,y) = \pi_{Y|X}(y|x)q^b_Y(y)$,

$$q^b_{XY}(x,y) \xrightarrow{(b)} \pi_{Y|X}(y|x)q^b_Y(y). \tag{A18}$$

Therefore, the corresponding drop in KL divergence during the copying process is given by

$$\mathrm{MC}^b(p_X) = D\big(p^aXY\|q^b_{XY}\big) - D\big(p^b_{XY}\|\tilde{q}^b_{XY}\big), \tag{A19}$$

$$= \sum_{x,y} p_X(x)p_Y(y) \ln \frac{p_X(x)p_Y(y)}{q^b_{XY}(x,y)}$$

$$- \sum_{x,y} \pi_{Y|X}(y|x)p_X(x) \ln \frac{\pi_{Y|X}(y|x)p_X(x)}{\pi_{Y|X}(y|x)q^b_Y(y)}, \tag{A20}$$

$$= D\big(p_Xp_Y\|q^b_{XY}\big) - D\big(p_X\|q^b_X\big). \tag{A21}$$

■

3. Additivity MMC for multiple channels

We focus on the case of two channel but the analysis below can be generalized to more than two channels. Consider two channels with input nodes X_1 and X_2, and output nodes Y_1 and Y_2, and each with condition distributions $\pi_{Y_1|X_1}$ and $\pi_{Y_1|X_1}$. We assume that for the process of overwriting of new input values, the input nodes X_1 and X_2 are independently sampled from p_{X_1} and p_{X_2}, respectively. We aim to show that in this case, the MMC associated with the communication across the joint channel $((X_1,X_2),(Y_1,Y_2))$ is the sum of MMC associated with the communication across the individual channels. We begin with the MMC of overwriting process. From Eq. (32), we have

$$\mathrm{MC}^a(p_{X_1X_2}) = \mathrm{I}(X_1,X_2;Y_1,Y_2) + D\big(p_{X_1X_2}\|q^a_{X_1X_2}\big). \tag{A22}$$

where, since inputs are sampled independently from p_{X_1} and p_{X_2}, the joint input distribution is a product distribution, $p_{X_1X_2}(x_1,x_2) = p_{X_1}(x_1)p_{X_2}(x_2)$. This allows us to write down

$$\mathrm{I}(X_1,X_2;Y_1,Y_2) = \mathrm{I}(X_1;Y_1) + \mathrm{I}(X_2;Y_2). \tag{A23}$$

Now, since the two input nodes X_1 and X_2 evolve independently during the overwriting process, as we discussed in Eq. (A 1), the prior distribution $q^a_{X_1X_2}$ is a product distribution of the form $q^a_{X_1X_2}(x_1,x_2) = q^a_{X_1}(x_1)q^a_{X_2}(x_2)$. Therefore, the KL-divergence term decomposes into two KL divergences:

$$D\big(p_{X_1X_2}\|q^a_{X_1X_2}\big) = D\big(p_{X_1}\|q^a_{X_1}\big) + D\big(p_{X_2}\|q^a_{X_2}\big). \tag{A24}$$

Using Eqs. (A23) and (A24), we get a decomposition of MMC of joint overwriting process into sum of MMC of overwriting processes for each input node,

$$\mathrm{MC}^a(p_{X_1X_2}) = \mathrm{MC}^a(p_{X_1}) + \mathrm{MC}^a(p_{X_2}). \tag{A25}$$

Now, according to Eq. (36), MMC of copy operation is given by

$$\mathrm{MC}^b(p_{X_1X_2}) = D\big(p_{X_1X_2}p_{Y_1Y_2}\|q^b_{X_1X_2Y_1Y_2}\big) - D\big(p_{X_1X_2}\|q^b_{X_1X_2}\big). \tag{A26}$$

Since the output values update according to conditional distributions $\pi_{Y_1|X_1}$ and $\pi_{Y_2|X_2}$ separately, therefore,

$$p_{Y_1Y_2}(y_1,y_2) = \sum_{x_1,x_2} \pi_{Y_1|X_2}(y_1|x_1)\pi_{Y_2|X_2}(y_2|x_2)p_{x_1}(x_1)p_{x_2}(x_2) \tag{A27}$$

$$= \left(\sum_{x_1} \pi_{Y_1|X_1}(y_1|x_1)p_{X_1}(x_1)\right) \times \left(\sum_{x_2} \pi_{Y_2|X_2}(y_2|x_2)p_{X_2}(x_2)\right) \tag{A28}$$

$$= p_{Y_1}(y_1)p_{Y_2}(y_2). \tag{A29}$$

Now, since the two joint systems (X_1,Y_1) and (X_2,Y_2) evolve independently during the copy operation according to the conditional distribution $\pi_{Y_1|X_1}$ and $\pi_{Y_2|X_2}$, therefore, the prior

distribution $q^b_{X_1X_2Y_1Y_2}$ decomposes into product distributions:

$$q^b_{X_1X_2Y_1Y_2}(x_1, x_2, y_1, y_2) = q^b_{X_1Y_1}(x_1, y_1)q^b_{X_2Y_2}(x_2, y_2). \quad \text{(A30)}$$

Using Eqs. (A29) and (A30), the first KL-divergence term in Eq. (A26) decomposes into two KL divergences:

$$D\big(p_{X_1X_2}p_{Y_1Y_2}\|q^b_{X_1X_2Y_1Y_2}\big) = D\big(p_{X_1}p_{Y_1}\|q^b_{X_1Y_1}\big) + D\big(p_{X_2}p_{Y_2}\|q^b_{X_2Y_2}\big). \quad \text{(A31)}$$

Now, $q^b_{X_1X_2}(x_1, x_2) = \sum_{y_1,y_2} = q^b_{X_1Y_1}(x_1, y_1)q^b_{X_2Y_2}(x_2, y_2) = q^b_{X_1}(x_1)q^b_{X_1}(x_2)$, and therefore the second KL divergence in Eq. (A26) also decomposes into sum of two terms,

$$D\big(p_{X_1X_2}\|q^b_{X_1X_2}\big) = D\big(p_{X_1}\|q^b_{X_1}\big) + D\big(p_{X_2}\|q^b_{X_2}\big). \quad \text{(A32)}$$

Equations (A31) and (A32) allow us to write the MMC of the joint copy operation as a sum of MMC of individual copy operations:

$$\mathrm{MC}^b(p_{X_1X_2}) = \mathrm{MC}^b(p_{X_1}) + \mathrm{MC}^b(p_{X_2}). \quad \text{(A33)}$$

Hence, Eqs. (A33) and (A25) tell us that when the values of the inputs of the two channels are drawn independently and when the outputs each channel evolves according to $\pi_{Y_1|X_1}$ and $\pi_{Y_2|X_2}$, then the MMC of communication across the joint channel is the sum of the MMC of communication across each channel considered separately:

$$\mathrm{MC}(p_{X_1}p_{X_2}) = \mathrm{MC}(p_{X_1}) + \mathrm{MC}(p_{X_2}). \quad \text{(A34)}$$

■

[1] R. Landauer, Irreversibility and heat generation in the computing process, IBM J. Res. Dev. **5**, 183 (1961).

[2] C. H. Bennett, The thermodynamics of computation—A review, Int. J. Theor. Phys. **21**, 905 (1982).

[3] R. Landauer, The physical nature of information, Phys. Lett. A **217**, 188 (1996).

[4] T. Sagawa, Thermodynamic and logical reversibilities revisited, J. Stat. Mech. (2014) P03025.

[5] C. van den Broeck and M. Esposito, Ensemble and trajectory thermodynamics: A brief introduction, Phys. A **418**, 6 (2015).

[6] U. Seifert, Stochastic thermodynamics: Principles and perspectives, Eur. Phys. J. B **64**, 423 (2008).

[7] R. Landauer, Minimal energy requirements in communication, Science **272**, 1914 (1996).

[8] C. E. Shannon, A mathematical theory of communication, Bell Sys. Tech. J. **27**, 379 (1948).

[9] P. Grover, K. Woyach, and A. Sahai, Towards a communication-theoretic understanding of system-level power consumption, IEEE J. Sel. Areas Commun. **29**, 1744 (2011).

[10] P. Grover, A. Goldsmith, and A. Sahai, Fundamental limits on the power consumption of encoding and decoding, in *2012 IEEE International Symposium on Information Theory Proceedings* (2012), pp. 2716–2720.

[11] A. Kolchinsky and D. H. Wolpert, Dependence of integrated, instantaneous, and fluctuating entropy production on the initial state in quantum and classical processes, Phys. Rev. E **104**, 054107 (2021).

[12] D. H. Wolpert, and A. Kolchinsky, Thermodynamics of computing with circuits, New J. Phys. **22**, 063047 (2020).

[13] A. Yadav, F. Caravelli, and D. Wolpert, System-independent lower bounds on entropy production incurred by running a computer program, arXiv:2411.16088

[14] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms* (Cambridge University Press, USA, 2003).

[15] U. Seifert, Stochastic thermodynamics, fluctuation theorems and molecular machines, Rep. Prog. Phys. **75**, 126001 (2012).

[16] U. Seifert, Stochastic thermodynamics: From principles to the cost of precision, Phys. A **504**, 176 (2018).

[17] M. Esposito, K. Lindenberg, and Christian V. den Broeck, Entropy production as correlation between system and reservoir, New J. Phys. **12**, 013013 (2010).

[18] M. Esposito, Stochastic thermodynamics under coarse graining, Phys. Rev. E **85**, 041125 (2012).

[19] A. C. Barato, and U. Seifert, Thermodynamic uncertainty relation for biomolecular processes, Phys. Rev. Lett. **114**, 158101 (2015).

[20] T. R. Gingrich, J. M. Horowitz, N. Perunov, and J. L. England, Dissipation bounds all steady-state current fluctuations, Phys. Rev. Lett. **116**, 120601 (2016).

[21] J. M. Horowitz, and T. R. Gingrich, Thermodynamic uncertainty relations constrain non-equilibrium fluctuations, Nat. Phys. **16**, 15 (2020).

[22] N. Shiraishi, K. Funo, and K. Saito, Speed limit for classical stochastic processes, Phys. Rev. Lett. **121**, 070601 (2018).

[23] V. Tuan Vo, T. V. Vu, and Y. Hasegawa, Unified approach to classical speed limit and thermodynamic uncertainty relation, Phys. Rev. E **102**, 062132 (2020).

[24] A. Kolchinsky and D. H. Wolpert, Dependence of dissipation on the initial distribution over states, J. Stat. Mech. (2017) 083202.

[25] Y. Polyanskiy and Y. Wu, *Information Theory From Coding to Learning* (Cambridge University Press, 2025).

[26] T. E. Ouldridge, C. C. Govern, and P. R. ten Wolde, Thermodynamics of computational copying in biochemical systems, Phys. Rev. X **7**, 021004 (2017).

[27] P. R. ten Wolde, N. B. Becker, T. E. Ouldridge, and A. Mugler, Fundamental limits to cellular sensing, J. Stat. Phys. **162**, 1395 (2016).

[28] D. H. Wolpert, The stochastic thermodynamics of computation, J. Phys. A: Math. Theor. **52**, 193001 (2019).

[29] D. H. Wolpert, Uncertainty relations and fluctuation theorems for Bayes nets, Phys. Rev. Lett. **125**, 200602 (2020).

[30] H.-H. Hasegawa, J. Ishikawa, K. Takara, and D. J. Driebe, Generalization of the second law for a nonequilibrium initial state, Phys. Lett. A **374**, 1001 (2010).

[31] T. Sagawa and M. Ueda, Nonequilibrium thermodynamics of feedback control, Phys. Rev. E **85**, 021104 (2012).

[32] T. Sagawa and M. Ueda, Generalized Jarzynski equality under nonequilibrium feedback control, Phys. Rev. Lett. **104**, 090602 (2010).

[33] L. Szilard, On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings, Behav. Sci. **9**, 301 (1964).

[34] T. Sagawa, Thermodynamics of information processing in small systems, Prog. Theor. Phys. **127**, 1 (2012).
[35] C. H. Bennett, Demons, engines and the second law, Sci. Am. **257**, 108 (1987).
[36] S. Ito and T. Sagawa, Information thermodynamics on causal networks, Phys. Rev. Lett. **111**, 180603 (2013).
[37] F. Tasnim, N. Freitas, and D. H. Wolpert, Entropy production in communication channels, Phys. Rev. E **110**, 034101 (2024).
[38] L. Yan and X. Ge, Entropy production-based energy efficiency optimization for wireless communication systems, IEEE Open J. Commun. Soc. **5**, 6482 (2024).
[39] V. Balasubramanian, Heterogeneity and efficiency in the brain, Proc. IEEE **103**, 1346 (2015).
[40] V. Balasubramanian, Brain power, Proc. Natl. Acad. Sci. USA **118**, e2107022118 (2021).
[41] T. E. Ouldridge and D. H. Wolpert, Thermodynamics of deterministic finite automata operating locally and periodically, New Journal of Physics **25**, 123013 (2023).
[42] C. Waltermann and E. Klipp, Information theory based approaches to cellular signaling, Biochim. Biophys. Acta **1810**, 924 (2011).
[43] A. C.-C. Yao, Some complexity questions related to distributive computing (preliminary report), in *Proceedings of the 11th Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 1979), pp. 209–213.
[44] C. H. Papadimitriou and M. Sipser, Communication complexity, in *Proceedings of the 14th Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 1982), pp. 196–200.
[45] I. Newman, Private vs common random bits in communication complexity, Inf. Process. Lett. **39**, 67 (1991).
[46] A. C.-C. Yao, How to generate and exchange secrets, in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science* (IEEE Computer Society, USA, 1986), pp. 162–167.
[47] G. W. Burr, A. Sebastian, E. Vianello, R. Waser, and S. Parkin, Emerging materials in neuromorphic computing: Guest editorial, APL Mater. **8**, 010401 (2020).
[48] D. Marković, A. Mizrahi, D. Querlioz, and J. Grollier, Physics for neuromorphic computing, Nat. Rev. Phys. **2**, 499 (2020).
[49] A. Yadav, Communication, GitHub (2025), https://github.com/Kensho28/Communication.

Correction: Minor errors were fixed in an expression following Eq. (3), in a character following Eq. (22), in a phrase in the text preceding Eq. (25), and in the summation limits in Eqs. (46) and (60).