Computation over Multiple-Access Channels

Bobak Nazer, *Student Member, IEEE* and Michael Gastpar, *Member, IEEE*

***Abstract*— The problem of reliably reconstructing a function of sources over a multiple-access channel is considered. It is shown that there is no source-channel separation theorem even when the individual sources are independent. Joint source-channel strategies are developed that are optimal when the structure of the channel probability transition matrix and the function are appropriately matched. Even when the channel and function are mismatched, these computation codes often outperform separation-based strategies. Achievable distortions are given for the distributed refinement of the sum of Gaussian sources over a Gaussian multiple-access channel with a joint source-channel lattice code. Finally, computation codes are used to determine the multicast capacity of finite field multiple-access networks, thus linking them to network coding.**

***Index Terms*— Distributed computation, joint source-channel coding, lattice codes, linear codes, multiple-access, network coding, separation theorem.**

I. Introduction

Computation and communication are often viewed as distinct problems. A communications engineer, tasked to design a multi-user system for performing computations while facing communication constraints, would almost certainly employ a version of the "separation principle." The system would employ a (distributed) source code to compress the sources into bits and a channel code to losslessly convey these bits over the noisy channel. The perceived reason for this design choice is two-fold. First, the abstraction of the sources and channel to bits lends itself to a universal, modular design. Second, it seems that the only gain from a joint source-channel design stems from exploiting the correlations between the sources as in [1].

In this paper, we study the problem of computing functions over multiple-access channels (MACs) and show that in many cases of interest, a joint design can exploit a match between the structure of the channel and the function to be computed. This *structural gain* does not hinge on the correlations between the sources and, with a perfect matching, increases the *computation rate* proportionally to the number of users. Furthermore, our underlying schemes are modular and depend primarily on coding techniques originally developed for their lower complexity.

Instead of fighting the interference caused by other users, our codes exploit channel collisions to compute functions efficiently. This can be thought of as a form of passive cooperation between transmitting terminals. More precisely, in the standard literature, cooperation is often considered in terms of the correlations (and more generally, dependence) it creates between transmitted signals, thus permitting it to outperform the communication performance attainable without cooperation. It should be clear that correlated signals only result in improved performance if the correlation between the signals is appropriately matched to the *structure* of the multiple-access channel. In our considerations, the goal is no longer to communicate messages, but a function thereof. By using appropriate codes, the transmitters cooperate to realize an enhanced communication performance. Again, it should be clear that this only results in a gain if the desired function is appropriately matched to the *structure* of the multiple-access channel. In this paper, we provide a partial characterization of the necessary structural match.

A. Related Work

Shannon showed in his landmark paper that separate source and channel code design is asymptotically optimal in a point-to-point setting [2, Theorem 21]. This insight has fueled a design philosophy based completely on bits. Although in many cases of interest, such an approach is optimal, it is well-known that in certain scenarios separation fails. For instance, Cover, El Gamal, and Salehi demonstrated that separation is suboptimal for transmitting correlated sources over a MAC in [1]. Their joint source-channel scheme uses the source correlations to create channel input probability distributions unavailable to a separation-based scheme. Exploiting the source correlations in this fashion is sometimes known as *collaborative gain*. Ahlswede and Han continued work on the problem of sending correlated sources over a MAC in [3]. In particular, they considered a variant of the problem in which only one of the sources had to be recovered.

In [4], [5], an uncoded joint source-channel scheme is shown to be optimal (and significantly better than separation) for estimating a remote source from multiple observations. Although at a first glance, the scheme seems to benefit only from the correlations between the observations, it also exploits an ideal structural match between the channel, a Gaussian MAC, and the sufficient statistic, the sum of the observations. This uncoded transmission framework has been extended to more general sensor network estimation problems in [6], [7].

In [8], function properties are used to reduce the amount of required communication in a large sensor network. For many functions, the sensors can process incoming data before sending it along to the fusion center, thus reducing the communications overhead.

Reliable distributed computation has been studied from the source coding perspective. The general problem is still

This work was supported by the National Science Foundation under CAREER Grant CCF-0347298 and NeTS-ProWin Grant CCF-0627024 as well a Graduate Research Fellowship. The material in this paper was presented in part at the 43rd Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, September 2005, at the IEEE International Symposium on Information Theory, Seattle, WA, July 2006, and at the IEEE International Symposium on Information Theory, Nice, France, June 2007.

The authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, 94720-1770, USA (email: bobak@eecs.berkeley.edu; gastpar@eecs.berkeley.edu)

open and seems prohibitively difficult with current techniques. Körner and Marton found the rate region for distributed compression of the parity of two correlated uniform binary sources in [9]. Their proof relies on random *linear* codes and their gains come entirely from the correlation between the sources. The seeming necessity of linear codes for this simple problem implies that random coding techniques are inadequate for the general problem.

In [10], Orlitsky and Roche determined the required rate for sending X to a decoder with side information Y that must reliably compute $f(X, Y)$. This is essentially a generalization of the Körner-Marton parity problem to any function except that the decoder gets Y for free. The basic result is that in most cases of interest, we must send X in its entirety to the decoder; further compression is only possible if for some x and x', $f(x, Y) = f(x', Y)$ with probability 1. In many cases, the gains enabled by requiring only a function of the sources at the decoder versus the sources themselves are marginal.

Earlier work by Yamamato established the rate-distortion function for sending X to a decoder that must reconstruct $f(X, Y)$ up to a given fidelity given Y as side-information [11]. In [12], the authors extend the rate-distortion function to the case where only a noisy version of X is available at the encoder.

Recently, there has been a great deal of interest in network coding [13]–[15]. The key idea is that routing is suboptimal for multicasting over networks: intermediate nodes may only need to send a function of incoming messages. For networks of point-to-point channels, network coding can be implemented separately from channel coding, i.e. there is a separation theorem [16]. In more general scenarios, such as networks that include deterministic broadcast channels, channel and network coding cannot be separated [17].

B. Summary of Paper Results

First, we will bound the performance of separation-based schemes. For many functions, if the sources are independent, then the best a separation-based scheme can do is have each encoder send its source in its entirety. Our main theorems are summarized below:

- Theorem 1 gives the maximum achievable rate (or computation capacity) for reliably sending linear functions over linear MACs. Essentially, we employ the same linear source code and linear channel code at each encoder. When the codewords collide on the channel, the codeword for our desired function is computed. In many cases, the gains over separation are proportional to the number of users even when the sources are independent.
- Theorem 2 gives achievable rates for reliably sending arbitrary functions over arbitrary MACs. Since codewords cannot always be reliably merged on the channel, we use a systematic scheme with an uncoded phase and a separation-based refinement phase. This scheme outperforms separation in some surprising cases.
- We use some of the recent lattice constructions from [18] to create lattice computation codes for sending the sum of Gaussian sources over a Gaussian MAC with Theorem 3. Uncoded transmission is exactly optimal when the source and channel bandwidths are equal. When there are more channel uses than source symbols, our codes continue to reap some of the gains of uncoded transmission.
- Through the study of a multicasting problem, we show that computation codes are useful even when the evaluation of functions is not called for in the problem statement. Theorem 4 gives the multicast capacity of a class of MAC networks. The MACs in the network are basically noisy adders over a finite field. Computation codes are used to harness these channels for part of the overall network code.

Appendix I gives inner bounds to the source coding region for distributed computation. Some of these bounds rely on conditional graph entropy results by Orlitsky and Roche in [10] which are also summarized in the appendix. Upper bounds on the computation capacity for a MAC appear in Appendix II. Note that some of these results were reported in the conference papers [19]–[21].

II. Problem Statement: Reliable Computation

We explore distributed computation through a variation on the standard multiple-access problem. Our distributed computation system (Figure 1) consists of the following basic elements: a set of M sources and a function, $f(\cdot)$, taken over those sources, a multiple-access channel, a joint source-channel encoder for each source, and a decoder. We now give mathematical definitions for each element.

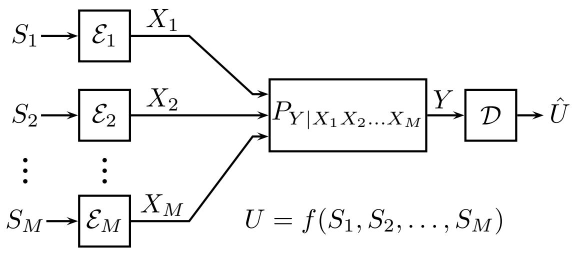

Fig. 1. Reliable Computation over a MAC. The decoder only reconstructs a function of the sources.

Remark 1: We assume that time is discrete. This can easily be justified by the well-known fact that any continuous-time system with finite bandwidth can be reduced to a discrete-time system with Shannon's sampling theorem [22, p.248-250]. In nearly any practical setting, finite bandwidth is assured.

Definition 1 (Sources): Let $\{(S_1[i], S_2[i], \ldots, S_M[i])\}_{i=1}^{\infty}$ be a sequence of independent drawings of an M-tuple of possibly dependent random variables (rvs) $S_1, S_2, \ldots, S_M$ which take values in the alphabets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$, respectively. The random variables are drawn according to the probability distribution function (pdf) $p_{S_1S_2\cdots S_M}(s_1, s_2, \ldots, s_M)$. As a shorthand, we sometimes write the pdf as $p_{S_1S_2\cdots S_M}$. Also, we may sometimes write the M-tuple of sources as simply $\mathbf{S}$.

Remark 2: We use superscripts to denote vectors of rvs. For example, $U^k = (U[1], U[2], \ldots, U[k])$ and $X_j^n =$

$(X_j[1], X_j[2], \dots, X_j[n])$. To simplify notation we may also denote a vector with a bold, lowercase version of the random variable, where the length can always be inferred from context. For example, $\mathbf{u} = (U[1], U[2], \dots, U[k])$.

Definition 2 (Desired Function): Let $\mathcal{U}$ be a discrete alphabet and f a fixed many-to-one function:

$$f : \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \mathcal{S}_M \to \mathcal{U}. \tag{1}$$

We will refer to f as the *desired function*.

Remark 3: We may want to recover more than one function of the sources at the decoder. This can easily be accommodated in our framework. For instance, we can define f from (1) to have a vector output with each element corresponding to the output of a single function.

Definition 3 (MAC): There is a conditional pdf

$$p_{Y|X_1X_2\cdots X_M}(y|x_1, x_2, \dots, x_M) \tag{2}$$

with $x_j \in \mathcal{X}_j,\ j \in \{1, 2, \dots, M\}$, and $y \in \mathcal{Y}$. We will refer to the X_j as the *channel inputs*, $p_{Y|X_1X_2\cdots X_M}$ as the *multiple-access channel* (MAC), and Y as the *channel output*.

Definition 4 (Computation Code): A (k, n, ϵ) code is specified by M *encoders*:

$$\mathcal{E}_j : \mathcal{S}_j^k \to \mathcal{X}_j^n, \tag{3}$$

for $j = 1, 2, \dots, M$, as well as a *decoder*:

$$\mathcal{D} : \mathcal{Y}^n \to \mathcal{U}^k, \tag{4}$$

such that:

$$\begin{aligned} X_j^n &= \mathcal{E}_j(S_j^k) \\ \hat{U}^k &= \mathcal{D}(Y^n) \\ \Pr(\hat{U}^k &\neq U^k) \leq \epsilon. \end{aligned} \tag{5}$$

Definition 5 (Average Cost): Each encoder may be subject to an *average cost constraint* over a block specified by a *cost function*, $\rho_j(x_j)$:

$$\begin{aligned} &\rho_j : \mathcal{X}_j \to \mathbb{R}_+ \\ &\rho_j(x_j^n) = \frac{1}{n}\sum_{i=1}^{n} \rho_j(x_{ji}) \leq \Gamma_j, \quad \Gamma_j \in \mathbb{R}_+, \end{aligned} \tag{6}$$

for $j = 1, 2, \dots, M$.

Remark 4: The cost constraint can be used to model and restrict energy consumption at the encoders. Note that not specifying a cost constraint is equivalent to assigning the same cost to each input symbol.

Definition 6 (Computation Rate): We say a *computation rate*, $\kappa = \frac{k}{n}$, is achievable if $\forall \epsilon \in (0, 1)$ there exists a $(\kappa n, n, \epsilon)$ code for some $n \in \mathbb{Z}_+$.

Remark 5: The computation rate is where we break with the standard information theoretic framework. Usually, we require that all M sources be transmitted across the channel losslessly. Here, we only penalize ourselves when the function $U = f(S_1, S_2, \dots, S_M)$ is incorrectly evaluated. Also note that our framework is general enough that through proper selection of $f(\cdot)$ it can become the standard multiple-access problem [22, p.388-407] as well as the multiple-access with correlated sources problem [1].

Definition 7 (Computation Capacity): The *computation capacity* is the supremum of all achievable computation rates.

III. Separation-Based Computation

In this section, we formally define what we mean by a separation-based scheme for computation over a MAC. Informally, a separation-based scheme consists of a set of source encoders and channel encoders as well as a source decoder and a channel decoder. Each source encoder must output a representation of its source in bits. Given these bit representations of the sources, the source decoder must be able to reconstruct the desired function with a vanishing probability of error.

Definition 8: The *distributed compression rate region*, $\mathbf{R}_f$, is the set of all rate vectors $(R_1, R_2, \dots, R_M)$ such that for all $\epsilon > 0$ and k large enough there are M source encoders and a source decoder of the form:

$$\mathcal{E}_j^S : \mathcal{S}_j^k \to \{0, 1\}^{kR_j} \tag{7}$$
$$\mathcal{D}^S : \{0, 1\}^{kR_1} \times \cdots \times \{0, 1\}^{kR_M} \to \mathcal{U}^k, \tag{8}$$

for $j = 1, 2, \dots, M$ such that the desired function $U = f(S_1, S_2, \dots, S_M)$ can be recovered with probability of error at most ϵ:

$$\begin{aligned} &\hat{U}^k = \mathcal{D}^S(\mathcal{E}_1^S(S_1^k), \dots, \mathcal{E}_M^S(S_M^k)) \\ &\Pr(\hat{U}^k \neq U^k) < \epsilon. \end{aligned} \tag{9}$$

Unfortunately, as of the writing of this paper, there is no single letter characterization for distributed compression of an arbitrary many-to-one function. Körner and Marton solved the special case where there are two correlated, uniform, binary sources and we want to recover their parity [9]. See Section IV-D.1 for more details. Orlitsky and Roche solved a related problem where a decoder must recover a function of the source and a side information random variable [10]. The required rate is given by a graph entropy characterization and is reviewed in detail in Appendix I-A. We will use their result to establish the distributed compression rate region for a restricted class of functions with independent sources as inputs. Essentially, if no input symbols can be merged without incurring errors and the sources are independent, then the sources must be sent in their entirety.

Lemma 1: Assume that the sources are independent and the desired function, f, is chosen such that for each pair of possible source symbols at an encoder, $s_j, s_j^* \in \mathcal{S}_j$, there is a choice of $s_1, s_2, \dots, s_{j-1}, s_{j+1}, \dots, s_M$ such that:

$$\Pr(f(s_1, \dots, s_j, \dots, s_m) \neq f(s_1, \dots, s_j^*, \dots, s_m)) > 0.$$

Then, the rate required for each decoder for distributed compression of $f(\cdot)$ is $R_j \geq H(S_j)$.

See Appendix I-A for a proof.

Example 1: Let $S_1, S_2, \ldots, S_M$ be independent sources drawn uniformly from the same alphabet. Then, real addition, $U_1 = \sum_{j=1}^{M} S_j$, and multiplication, $U_2 = S_1 \cdot S_2 \cdot \ldots \cdot S_M$, satisfy the conditions of Lemma 1.

The distributed compression rate region for complete recovery of the sources was characterized by Slepian and Wolf in [23]. Their classic result shows that the distributed encoders need only the sum rate of a joint encoder. The Slepian-Wolf rate region, $\mathbf{R}_{SW}$, is the set of all rate vectors, $(R_1, R_2, \ldots, R_M)$, satisfying:

$$R(\mathbf{T}) \leq H(\mathbf{T}|\mathbf{T}^C) \quad \forall \mathbf{T} \subseteq \mathbf{S}, \tag{10}$$

where $R(\mathbf{T}) = \sum_{j \in \mathbf{T}} R_j$ (see [22, p.415, Thm.14.4.2]).

Ahlswede and Liao concurrently determined the MAC capacity region for independent messages [24], [25]. The MAC capacity region, $\mathbf{R}_{\text{MAC}}$, is the closure of the convex hull of the set of all rate vectors, $(R_1, R_2, \ldots, R_M)$, satisfying:

$$R(\mathbf{T}) \leq I(X(\mathbf{T}); Y | X(\mathbf{T}^C)) \quad \forall \mathbf{T} \subseteq \mathbf{S}, \tag{11}$$

for some product distribution $p(x_1, x_2, \ldots, x_M) = \prod_{j=1}^{M} p(x_j)$ where $X(\mathbf{T}) = \{X_j : j \in \mathbf{T}\}$ (see [22, p.403, Thm.14.3.5]).

Definition 9: The *maximum sum rate* of a MAC is:

$$C_{\text{MAC}} = \max_{(R_1, R_2, \ldots, R_M) \in \mathbf{R}_{\text{MAC}}} \sum_{j=1}^{M} R_j. \tag{12}$$

Definition 10: We say that the maximum sum rate of a MAC is *symmetric* if $R_1^* = R_2^* = \cdots = R_M^*$ where

$$(R_1^*, R_2^*, \ldots, R_M^*) \in \operatorname*{arg\,max}_{(R_1, R_2, \ldots, R_M) \in \mathbf{R}_{\text{MAC}}} \sum_{j=1}^{M} R_j. \tag{13}$$

A separation-based code consists of a distributed compression code concatenated with a MAC code. Note that in many cases, we will choose a ratio of source symbols per channel use such that the two rate regions intersect and communication is possible.

Definition 11: A computation rate $\kappa = \frac{k}{n}$ is *achievable with separation* if:

$$\mathbf{R}_{\text{MAC}}(\kappa) = \left\{ \left(\frac{R_1}{\kappa}, \ldots, \frac{R_M}{\kappa} \right) : (R_1, \ldots, R_M) \in \mathbf{R}_{\text{MAC}} \right\}$$
$$\mathbf{R}_f \cap \mathbf{R}_{\text{MAC}}(\kappa) \neq \emptyset. \tag{14}$$

As shown in [1], when we want to send correlated sources over a MAC, separation is not optimal. Clearly, if we allow our sources to be correlated but only require a function of these sources at the decoder, a separation-based scheme may not be optimal for the same reasons. However, even if we assume that the sources are independent, we still do not get a separation theorem as shown in the following example, taken from Problem 1.1 in [26].

Example 2: Let S_1 and S_2 be independent $\mathcal{B}(\frac{1}{2})$ sources. Each source is seen by a separate encoder with access to one terminal of a MAC. The MAC input alphabets are $\mathcal{X}_1 = \mathcal{X}_2 = \{0, 1\}$ and the output is $Y = X_1 \oplus X_2$. The maximum sum rate of this MAC is clearly $C_{\text{MAC}} = 1$. At the decoder, we would like to losslessly compute $U = S_1 \oplus S_2$. Using Lemma 1 and the data processing inequality, it can be shown that the best separation-based scheme achieves a computation rate of $\kappa_{\text{SEP}} = \frac{1}{2}$. The separation-based scheme just amounts to using two channel uses, one for each source. However, simultaneously placing the sources on the channel (or uncoded transmission), achieves a computation rate of $\kappa_{\text{JOINT}} = 1$ which is clearly optimal by the data processing inequality.

As the example demonstrates, sometimes we can compute the desired function using the channel. In these cases, joint source-channel schemes can achieve a much higher computation rate than separation-based schemes, sometimes a factor of M higher. Of course, the above example is somewhat contrived, as the channel performs exactly the operation we desire. Furthermore, the channel is not noisy, so interference is the only issue. Our results show that using the channel's natural operation to compute a function can give us boosts over separation-based schemes, even when the channel is noisy.

IV. LINEAR COMPUTATION CODING

In this section, we develop a class of MACs for which we can find the computation capacity for linear functions.

A. Definitions

Our achievable rates coincide with our upper bounds when there is an ideal structural match between the channel and the desired function. More specifically, we require that our MAC can be written as a linear function over a finite field of its inputs followed by a symmetric discrete memoryless channel (DMC) as illustrated in Figure 2. Furthermore, there must be a one-to-one map from our desired function to a linear function over the same finite field. If these conditions are met, there is a joint source-channel code that achieves the computation capacity. In these perfectly matched cases, separation-based schemes will fall far short of the optimal performance. For many of these scenarios, such as sending a sum over a noisy adder (as in Figure 2), the gap will be proportional to M, the number of users.

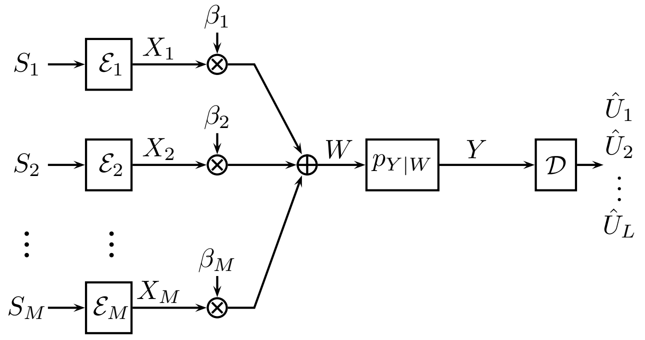

Fig. 2. Discrete Linear Multiple-Access Channel

Definition 12: We call a MAC *linear* with respect to $\mathbb{F}$ if its channel inputs take values on a Galois field $\mathbb{F}$ and we can represent the channel output Y as coming from a discrete memoryless channel (DMC), $p_{Y|W}$, where:

$$W = \sum_{j=1}^{M} \beta_j X_j \tag{15}$$

for some $\beta_j \in \mathbb{F} \setminus \{0\}$ where 0 is the zero symbol in $\mathbb{F}$. Note that addition and multiplication are performed over $\mathbb{F}$. Also, note that if any β_j could take a zero value, then that encoder would effectively have no channel input. See Figure 2.

In [27], it was shown that linear codes are sufficient for achieving the maximum sum rate of a linear multiple-access channel.

We will also need the following definition from [28, p.94]:

Definition 13: We say that a DMC is *symmetric* if the output symbols can be placed into subsets such that for each subset the probability transition matrix satisfies the following two conditions:

1) Each row is a permutation of every other row.
2) Each column is a permutation of every other column.

Remark 6: It can be shown that the uniform distribution achieves capacity on symmetric DMCs [24], [29]. This concept can also be extended to channels with discrete inputs and continuous outputs such as the binary-input Gaussian channel. However, in the interests of space, we limit ourselves to discrete alphabets.

B. Results

We will need a result by Csiszár for linear Slepian-Wolf coding [30].

Lemma 2 (Csiszár): Let $(U_1, U_2, \ldots, U_L)$ be a vector source generated i.i.d. according to some joint probabilty mass function (pmf) on a discrete alphabet. For any point in the Slepian-Wolf rate region and k large enough, there are matrices $\mathbf{H_1}, \mathbf{H_2}, ..., \mathbf{H_L}$ of size $k \times m_\ell$, respectively, taking values over a Galois field with associated decoding function $b(\cdot)$ that can be used to compress the sources in a distributed fashion with $\Pr((\hat{U}_1^k, \hat{U}_2^k, \ldots, \hat{U}_L^k) \neq (U_1^k, U_2^k, \ldots, U_L^k)) < \epsilon \quad \forall \epsilon > 0$.

The proof relies on showing that multiplying by a random matrix is equivalent to random binning. For a full proof, see [30]. The following lemma appears as Problem 2.1.11 in [31].

Lemma 3: Consider a symmetric DMC with encoder input W, channel input X, channel output Y, and capacity C. Both W and X take values on Galois field $\mathcal{X}$. For any $\epsilon > 0$ and n large enough, there exists a matrix $\mathbf{G} \in \mathcal{X}^{m \times n}$ with associated decoding function $c(\cdot)$ such that when $\mathbf{x} = \mathbf{wG}$, $\Pr(c(\mathbf{y}) \neq \mathbf{w}) < \epsilon$ if $m \log|\mathcal{X}| < nC$.

The basic proof idea is that using a random generator matrix results in pairwise independent codewords whose entries are i.i.d. according to a uniform distribution. See [28, §6.2] for the binary case.

Corollary 1: Lemma 3 also holds for asymmetric DMCs so long as C, the channel capacity, is replaced with $I(W;Y)$ where $p(W)$ is taken to be uniform.

Theorem 1: Let $f_1, f_2, \ldots, f_L$ be linear functions with respect to $\mathbb{F}$ and let $U_\ell = f_\ell(S_1, S_2, \ldots, S_M)$. For a linear MAC with respect to $\mathbb{F}$ with symmetric $p_{Y|W}$ and capacity $C = \max_{p(w)} I(W;Y)$,

$$\kappa = \frac{C}{H(U_1, U_2, \ldots, U_L)} \tag{16}$$

is the computation capacity for the vector of desired functions, $U = (U_1, U_2, \ldots, U_L)$.

Proof: (*Achievability.*) Using Lemma 2, we choose matrices $\mathbf{H_1}, \mathbf{H_2}, ..., \mathbf{H_L}$ of size $k \times m_\ell$, respectively, to get $(U_1, U_2, \ldots, U_L)$ to some point in the Slepian-Wolf rate region with sum rate $H(U_1, U_2, \ldots, U_L)$.

Using Lemma 3, we choose a matrix $\mathbf{G}$ of size $(\sum_{\ell=1}^{L} m_\ell) \times n$ to communicate over $p_{Y|W}$ at capacity. We note that each β_j has a multiplicative inverse β_j^{-1}.

At each encoder we use the following encoding rule:

$$\mathbf{t_j} = [\alpha_{1j}\mathbf{s_j}\mathbf{H_1} \; \alpha_j\mathbf{s_j}\mathbf{H_2} \; \cdots \; \alpha_{Lj}\mathbf{s_j}\mathbf{H_L}]$$
$$\mathbf{x_j} = \beta_j^{-1}\mathbf{t_j}\mathbf{G}.$$

After the linear operation performed by the channel, we get:

$$\mathbf{w} = \beta_1\mathbf{x_1} + \beta_2\mathbf{x_2} + \cdots + \beta_M\mathbf{x_M}$$
$$\mathbf{w} = [\mathbf{u_1}\mathbf{H_1} \; \mathbf{u_2}\mathbf{H_2} \; \cdots \; \mathbf{u_L}\mathbf{H_L}]\,\mathbf{G}.$$

The received sequence, $\mathbf{y}$, is just $\mathbf{w}$ corrupted by symmetric noise. Using Lemma 3, we can recover $[\mathbf{u_1}\mathbf{H_1} \; \mathbf{u_2}\mathbf{H_2} \; \cdots \; \mathbf{u_L}\mathbf{H_L}]$ from $\mathbf{y}$ for any block error probability $\epsilon > 0$ for n large enough so long as $\sum_{\ell=1}^{L} m_\ell < \frac{nC}{\log|\mathcal{X}|}$. Using Lemma 2, we can recover $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_L}$ for any block error probability $\epsilon > 0$ for k large enough so long as $\frac{kH(U_1,U_2,\ldots,U_M)}{\log|\mathcal{X}|} < \sum_{\ell=1}^{L} m_\ell$ (and the appropriate side rate constraints are met). This succeeds with probability greater than $1-\epsilon$ for k large enough so long as $kH(U_1, U_2, \ldots, U_L) < nC$.

(*Converse.*) For this class of MACs, we can simply allow the encoders to completely collaborate and get a tight upper bound. This reduces our problem to a point-to-point problem and we can invoke the separation theorem to get that $kH(U_1, U_2, \ldots, U_L) \leq nC$. It immediately follows that $\kappa \leq \frac{C}{H(U_1,U_2,\ldots,U_L)}$. ■

If our MAC is linear but its DMC is asymmetric, we can use the strategy used in Theorem 1 to give an achievable computation rate.

Corollary 2: If $p_{Y|W}$ is asymmetric then the following computation rate is achievable for sending our vector of desired linear functions, $(U_1, U_2, \ldots, U_L)$:

$$\kappa = \frac{I(W;Y)}{H(U_1, U_2, \ldots, U_L)}, \tag{17}$$

where $I(W;Y)$ is evaluated using a uniform distribution on W.

Remark 7: Theorem 1 can be easily extended to functions that have an invertible, entropy-preserving map to linear functions.

If the sources satisfy the conditions of Lemma 1, then we must transmit them in their entirety to reconstruct our desired linear functions with a separation-based scheme. Thus, $\kappa_{\text{SEP}} = \frac{C}{\sum_{j=1}^{M} H(S_j)}$ is the highest computation rate achievable with separation. By Theorem 1, $\kappa_{\text{COMP}} = \frac{C}{H(U_1,\ldots,U_L)}$ is the optimal computation rate. If $H(U_1,\ldots,U_L) < \sum_{j=1}^{M} H(S_j)$, then $\kappa_{\text{SEP}} < \kappa_{\text{COMP}}$. For example, if the sources are independent and our desired function is simply their sum, then the conditions of Lemma 1 are satisfied and $H(U) < \sum_{j=1}^{M} H(S_j)$ so $\kappa_{\text{SEP}} < \kappa_{\text{COMP}}$.

C. Simple Examples

Our strategy, computation coding, is optimal for linear MACs when $p_{Y|W}$ is symmetric. If $p_{Y|W}$ is asymmetric, computation coding performs better than separation in some cases and separation performs better in others. We show this by means of two examples. In the first, computation coding outperforms separation.

Example 3: Let S_1 and S_2 be independent $\mathcal{B}(\frac{1}{2})$ sources. The channel inputs, X_1 and X_2, also take values on $\{0,1\}$ and the channel output, Y, is formed by passing $W = X_1 \oplus X_2$ through an asymmetric DMC whose probability transition matrix is given by the following table:

$P(Y=y\|W=w)$	$y=0$	$y=1$
$w=0$	0.9	0.1
$w=1$	0.5	0.5

We want to send $U = S_1 \oplus S_2$ over the channel. The MAC's maximum sum rate is $C_{\text{MAC}} = 0.148$ whereas the mutual information induced by a uniform distribution is 0.147. We get that $\kappa_{\text{SEP}} = 0.074$ by Lemma 1 and $\kappa_{\text{COMP}} = 0.147$ by Corollary 2. Note that the achieved computation rate is close to the upper bound, $\kappa_{\text{JOINT}} = 0.148$, from Lemma 12.

We now give an example with an asymmetric DMC where separation outperforms computation coding.

Example 4: Let S_1 and S_2 be independent sources drawn uniformly over GF(5). The channel inputs, X_1 and X_2, also take values on GF(5) and the channel output, Y, is formed by passing $W = X_1 \oplus_5 X_2$ through an asymmetric DMC whose probability transition matrix is given by the following table:

$P(Y=y\|W=w)$	$y=0$	$y=1$
$w=0$	1	0
$w=1$	0.5	0.5
$w=2$	0.5	0.5
$w=3$	0.5	0.5
$w=4$	0	1

We want to send $U = S_1 \oplus_5 S_2$ over the channel. The MAC's maximum sum rate is $C_{\text{MAC}} = 1$ and is achieved by only using input symbols 0 and 4. A uniform distribution induces a mutual information of $\frac{2}{5}$. We get that $\kappa_{\text{SEP}} = \frac{1}{2}\frac{1}{\log 5}$ from Lemma 1 and that $\kappa_{\text{COMP}} = \frac{2}{5}\frac{1}{\log 5}$ from Corollary 2. In this scenario, $\kappa_{\text{SEP}} > \kappa_{\text{COMP}}$. The best upper bound we have on the computation rate is $\kappa_{\text{JOINT}} = \frac{1}{\log 5}$ using Lemma 12.

This shows that the channel symmetry condition cannot be removed from the statement of Theorem 1. Unlike the point-to-point setting, we are not free to create a linear code and map different length subsequences to channel inputs unevenly to achieve any distribution (see [28, p.208]). The distributed nature of computation codes prohibits this kind of non-linear mapping.

D. Extended Example: Mod-2 Adder MAC

We now explore an example to illustrate some of the key principles at work in computation coding. Our example centers on the mod-2 adder MAC (M2MAC) (Figure 3). All operations are done in GF(2). There are two sources, S_1 and S_2, generated from the following joint pdf:

$$\begin{aligned} \Pr(S_1=0,S_2=0) &= \Pr(S_1=1,S_2=1) = \frac{1-p}{2} \\ \Pr(S_1=0,S_2=1) &= \Pr(S_1=1,S_2=0) = \frac{p}{2}. \end{aligned} \quad (18)$$

A simple calculation will show that S_1 and S_2 have uniform marginal distributions. Our goal is to losslessly transmit $U = S_1 \oplus S_2$ across the channel at the highest computation rate $\kappa = \frac{k}{n}$. The entropy of U is given by the binary entropy function:

$$h_B(p) = -p\log p - (1-p)\log(1-p). \quad (19)$$

Note that all logarithms in this paper are in base 2. The channel input and output alphabets are identically given by $\mathcal{X}_1 = \mathcal{X}_2 = \mathcal{Y} = \{0,1\}$. The channel inputs are added mod-2 to yield $W = X_1 \oplus X_2$ which is passed through a binary symmetric channel (BSC) with crossover probability q to give Y (see Figure 3).

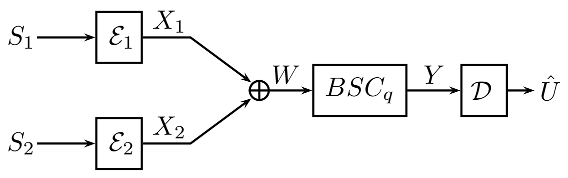

Fig. 3. Mod-2 Adder Multiple-Access Channel (M2MAC)

1) Separation: Körner-Marton Revisited: Our sources and desired function are identical to those from the Körner-Marton problem [9]. By combining the Körner-Marton source coding scheme with an appropriate MAC code, we will get the optimal separation-based scheme.

Lemma 4 (Körner-Marton): S_1 and S_2 are separately encoded by two source coders at rates R_1 and R_2. The mod-2 sum, U, can be reconstructed with $\Pr(\hat{U}^k \neq U^k) < \epsilon,\ \forall \epsilon > 0$ iff $R_1 > h_B(p)$ and $R_2 > h_B(p)$.

For the M2MAC, the capacity region has only a single constraint:

$$R_1 + R_2 < 1 - h_B(q). \tag{20}$$

Note that this implies that time-sharing is optimal for the M2MAC.

We can now give the best possible computation rate available using separation. The sum source coding rate required is $2h_B(p)$ and the MAC sum capacity is $1 - h_B(q)$. Reliable communication requires that $k(2h_B(p)) < n(1-h_B(q))$. This gives the optimal separation computation rate of:

$$\kappa_{\text{SEP}} = \frac{1}{2}\left(\frac{1-h_B(q)}{h_B(p)}\right). \tag{21}$$

Remark 8: The Körner-Marton scheme allows for a strictly lower sum source coding rate and thus, a higher computation rate than Slepian-Wolf coding of S_1 and S_2.

2) Computation Coding: The best separation-based scheme for the M2MAC uses structured source coding to exploit the source correlations. The channel coding strategy focuses on avoiding the interference caused by the other user. Yet, the interference is due to the summation taken by the MAC. Computation coding exploits this summation by using both a structured source code and a structured channel code. In doing so, it can optimally exploit both the source correlations and the structure of the MAC. An application of Theorem 1 to this scenario yields the following corollary.

Corollary 3: The optimal reliable computation rate for sending $U = S_1 \oplus S_2$ over the M2MAC is

$$\kappa = \frac{1-h_B(q)}{h_B(p)}. \tag{22}$$

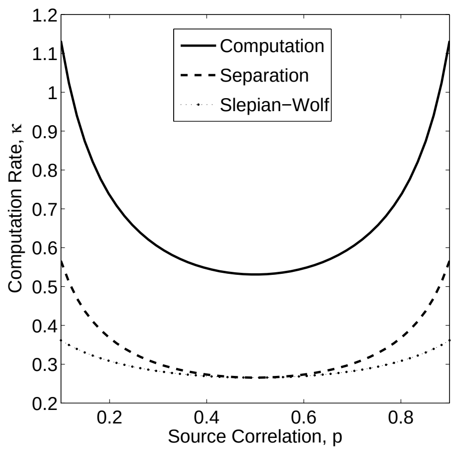

Fig. 4. Comparison of schemes for computing parity over a noisy modulo-2 adder MAC (M2MAC).

Somewhat surprisingly, this strategy allows for a computation rate twice that of the separation scheme, regardless of the source statistics. The computation rates for computation coding (Corollary 3), the best separation-based scheme (21), and a suboptimal separation-based scheme that uses Slepian-Wolf source coding over an M2MAC with crossover probability $q = 0.1$ are shown in Figure 4.

3) Discussion: Both our computation coding scheme and the best separation-based scheme take advantage of the structure of the function for source coding. The computation coding scheme goes one step further and takes advantage of structure of channel. The computation rate is doubled by this structural gain. This shows that the MAC rate region is an insufficient characterization of the channel for distributed computation.

The symmetric source pdf (see (18)) used for the M2MAC example can be changed to any joint pdf and the computation capacity will still be achieved by the scheme put forth in Theorem 1. However, for an asymmetric pdf, the Körner-Marton scheme may not be the best separation-based strategy. It is only known to be optimal for the symmetric pdf in (18), as this is the most general pdf that results in uniform marginal pdfs. Ahlswede and Han showed that if the marginals are not uniform, there are achievable points outside the Körner-Marton region [3].

To be more specific, both the Körner-Marton scheme and computation coding calibrate their codes using the entropy of the desired function. Körner-Marton fails as a general solution as it then converts the linear representation into bits, which destroys the code's match with the function. The function-channel match in computation coding allows for a continuous abstraction of the problem in terms of the underlying finite field. This is why we are able to meet our upper bounds in matched cases.

The most interesting aspect of our strategy is that it depends entirely on codes that were originally intended to reduce system complexity. Elias' random linear coding proof was meant to show that the search for implementable codes is not futile; all of the benefits of Shannon's random codebooks can be transferred into random generator matrices [32]. The Körner-Marton result and our computation code show that structured codes can enable rate gains. In particular, structured codes allow redundancy to be added in a distributed, yet structured, fashion.

Symmetric, linear MACs seem to be the largest class of MACs for which our computation codes are optimal. In the next section, we explore strategies for sending arbitrary functions over arbitrary MACs.

V. Systematic Computation Coding

We now develop computation codes for sending arbitrary functions over arbitrary MACs. Our main idea is to use uncoded transmission followed by an update phase, which can be thought of as a systematic code.

A. Arbitrary Functions over Arbitrary MACs

In the point-to-point setting, systematic transmission refers to first sending a block of the source uncoded across the channel and then using a code to refine the noisy version of the source [33], [34]. The decoder uses the uncoded block as side information to infer the source from the received codeword. Systematic transmission is a good framework for the digital upgrade of analog systems. We propose a systematic computation coding scheme that first uses uncoded transmission to send a noisy function to the decoder and then refines this function with a separation-based scheme.

We briefly consider the code used in Section IV-D for sending the parity of binary sources over the M2MAC. Assume the sources are independent and the channel code is written in systematic form. This computation coding scheme is also systematic in that the encoders first send a noisy version of the desired sum and then refine it with parity-check bits. In this setting, we allow the channel to merge both the information bits and the parity-check bits to give a codeword that describes the sum of the sources. However, for an arbitrary MAC, we may not able to use the channel to combine our codewords. Therefore, we only use a joint source-channel code to send a noisy version of the function. We will then switch over to a separation-based scheme that uses a linear source code and a capacity-achieving MAC code at each encoder to refine the noisy function.

Theorem 2: Let $f_1, f_2, \ldots, f_L$ be arbitrary functions and let $U_\ell = f_\ell(S_1, \ldots, S_M)$. Choose a Galois field $\mathbb{F}$, linear functions $g_1, g_2, \ldots, g_L$ over $\mathbb{F}$ and mappings $c_j : \mathcal{S}_j \to \mathbb{F}$ for $j = 1, \ldots, M$ and $d_1, \ldots, d_L$ such that:

$$\Pr(d_\ell(g_\ell(c_1(S_1), \ldots, c_M(S_M))) = f_\ell(S_1, \ldots, S_M)) = 1$$

for $\ell = 1, 2, \ldots, L$. Let $V_\ell = g_\ell(c_1(S_1), \ldots, c_M(S_M))$. If the maximum sum rate of the MAC is symmetric[1], then the computation rate

$$\kappa = \frac{C_{\text{MAC}}}{C_{\text{MAC}} + MH(V_1, V_2, \ldots, V_L|T)} \tag{23}$$

is achievable for any joint pdf of the form:

$$p_{T|X_1\cdots X_M}(t|x_1, \ldots, x_M) \left(\prod_{j=1}^{M} p_{X_j|S_j}(x_j|s_j) \right) \cdots \\ \cdots (p_{S_1\cdots S_M}(s_1, \ldots, s_m)) \tag{24}$$

where

$$p_{T|X_1\cdots X_M}(t|x_1, \ldots, x_M) = p_{Y|X_1\cdots X_M}(t|x_1, \ldots, x_M)$$

Proof: First, let $C_{\ell j} = c_{\ell j}(S_j)$ and let $V_\ell = g_\ell(C_{\ell 1}, \ldots, C_{\ell M})$.

(*Uncoded Transmission.*) At time step i for $1 \le i \le k$, encoder j maps $S_j[i]$ into a channel input, $X_j[i]$, according to $p_{X_j|S_j}(x_j|s_j)$. The decoder collects the channel outputs to use as side information in the next phase, $T^k = Y^k$.

(*Refinement.*) Using Lemma 2, we choose matrices $\mathbf{H_1}, \mathbf{H_2}, ..., \mathbf{H_L}$ of size $k \times m_\ell$, respectively, to get $(V_1, V_2, \ldots, V_L)$ to some point in the Slepian-Wolf rate region of $(V_1, V_2, \ldots, V_L, T)$ with sum rate $H(V_1, V_2, \ldots, V_L|T)$.

At each encoder we compute:

$$\mathbf{r_j} = [\alpha_{1j}\mathbf{c_{1j}}\mathbf{H_1} \ \alpha_{2j}\mathbf{c_{2j}}\mathbf{H_2} \ \cdots \ \alpha_{Lj}\mathbf{c_{Lj}}\mathbf{H_L}] \, .$$

[1]We assume that maximum sum rate of the MAC is symmetric according to Definition 10 to simplify the statement of the theorem. This can be removed for a more general (but more cumbersome) theorem statement.

Each encoder then transmits $\mathbf{r_j}$ to the decoder using a multiple-access channel code:

$$\mathcal{E}_j^C : \mathbb{F}^{\sum_{\ell=1}^L m_\ell} \to \mathcal{X}_j^n,$$

targeted at the symmetric maximum sum rate, C_{MAC}. Choose $\epsilon > 0$. We can recover $\mathbf{r_1}, \mathbf{r_2}, \ldots, \mathbf{r_M}$ at the decoder with probability of error less than $\frac{\epsilon}{2}$ for n large enough if $\log|\mathbb{F}| \sum_{\ell=1}^L m_\ell < nC_{\text{MAC}}$. The decoder then computes:

$$\mathbf{w} = \mathbf{r_1} + \mathbf{r_2} + \cdots + \mathbf{r_M}$$
$$\mathbf{w} = [\mathbf{v_1}\mathbf{H_1} \ \mathbf{v_2}\mathbf{H_2} \ \cdots \ \mathbf{v_L}\mathbf{H_L}] \, ,$$

where addition is over $\mathbb{F}$. Using Lemma 2, we can recover $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_L}$ from $\mathbf{w}$ and $\mathbf{t}$ with probability of error less than $\frac{\epsilon}{2}$ for $m_1, m_2, \ldots, m_L$ large enough that satisfy $kH(V_1, V_2, \ldots, V_M|T) < \log|\mathbb{F}| \sum_{\ell=1}^L m_\ell$ (and the appropriate Slepian-Wolf side rate constraints are met). Finally, we apply the functions $d_1, d_2, \ldots, d_L$ to recover our desired function sequences $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_L}$. The probability of error is upper bounded by ϵ.

The uncoded transmission phase requires k channel uses and the refinement phase requires at least $\frac{kMH(V_1,V_2,\ldots,V_L|T)}{C_{\text{MAC}}}$ channel uses. Thus, we can achieve any computation rate satisfying: $\kappa = \frac{k}{n} < \frac{C_{\text{MAC}}}{C_{\text{MAC}}+MH(V_1,V_2,\ldots,V_L|T)}$. ■

Remark 9: In some cases, we will have $H(S_j) < H(V_1, V_2, \ldots, V_L|T)$. In this case, encoder j can just send S_j in its entirety to the decoder to lower the overall computation rate.

Remark 10: We can recover the classical MAC capacity region by setting $L = M$ and setting $U_j = S_j$ for $j = 1, 2, \ldots, M$. These sources are independent and take values on the input alphabets of the MAC. Their marginal distributions assigned by the maximum sum rate achieving distribution for the MAC.

Remark 11: Theorem 2 can be further generalized by allowing for a different ratio of source symbols to channel symbols in the uncoded phase. As it is currently stated, Theorem 2 uses one channel symbol per source symbol in the uncoded phase. This causes the computation rate to be upper bounded by 1.

Corollary 4: If the mappings $d_1, \ldots, d_L$ in Theorem 2 are invertible and entropy-preserving, then $H(V_1, \ldots, V_L|T) = H(U_1, \ldots, U_L|T)$.

We show that systematic computation coding can outperform separation-based coding with the following example.

Example 5: Our setting is basically the same as the M2MAC (see Section IV-D). For simplicity, we make S_1 and S_2 independent $\mathcal{B}(\frac{1}{2})$ processes. The only difference is the channel performs a real addition, $W = S_1 + S_2$, and then noise is added mod-3 to get the output: $Y = W \oplus_3 Z$. The additive noise Z is distributed according to $P(Z = 0) = .8$ and $P(Z = 1) = P(Z = 2) = .1$. Our desired function is the parity of the sources, $U = S_1 \oplus S_2$. We would like to apply Theorem 2 for computing U. In the first phase, we just transmit the sources uncoded ($p_{X_j|S_j}(x_j|s_j) = \delta(x_j - s_j)$) to get side information T at the decoder. The conditional entropy of our desired function, $U = S_1 \oplus S_2$, is $H(U|T) = 0.60$ and the maximum sum rate of the MAC is $C_{\text{MAC}} = 0.66$. With this method, we can achieve $\kappa_{\text{COMP}} = 0.35$. From Lemma 1, we get that the best separation-based scheme gives a computation rate of $\kappa_{\text{SEP}} = 0.33$. Interestingly, we can outperform our systematic scheme by using the quasi-linearity of the channel to merge codewords. We simply employ the computation code for the M2MAC from Corollary 3 directly and map the output symbol 2 to 0 at the decoder. This gives us an improved computation rate of $\kappa = 0.40$. None of these schemes meet the upper bound given by Lemma 13: $\kappa_{\text{JOINT}} = 0.66$.

The above example shows that our systematic scheme can be beaten by other strategies. In this case, we mapped our problem into an M2MAC problem. In general, we can take an arbitrary MAC and map it to a linear MAC in order to employ a computation code.

In the next example, we compute the parity of independent binary sources over a binary multiplying channel.

Example 6: S_1 and S_2 are $\mathcal{B}(\alpha)$ sources. We are interested in sending the mod-2 sum $U = S_1 \oplus S_2$ over the binary multiplying channel (BMC) $Y = X_1 \cdot X_2$ where $\mathcal{X}_1 = \mathcal{X}_2 = \{0, 1\}$. In the first phase, we send the sources uncoded across the channel and in the second phase, we use a MAC code to send our update bins. The computation rates for both separation-based coding (Lemma 1) and our scheme (Theorem 2) are plotted in Figure 5. The upper bound is significantly higher than both achievable rates and is not shown on the plot. Our scheme outperforms separation-based coding for α between approximately 0.65 and 0.85 (and between 0.15 and 0.35 by symmetry). The underlying reason is that these input distributions get close to the maximum mutual information for the MAC, resulting in good side information for the second phase. It is quite surprising that our scheme even moderately outperforms separation as there is almost no structural match between the channel and the desired function.

The following example is the dual of the last one: we compute the product of binary independent sources over a mod-2 adder.

Example 7: S_1 and S_2 are independent $\mathcal{B}(\alpha)$ sources. We want to send $U = S_1 \cdot S_2$ over a mod-2 adder. The channel output is given by $Y = X_1 \oplus X_2$, $\mathcal{X}_1 = \mathcal{X}_2 = \{0, 1\}$. We can losslessly recover U from $V = S_1 \oplus_3 S_2$. Our scheme is to send S_1 and S_2 uncoded over the channel for phase one and then use this as side information to send V. The computation rates for both separation-based coding (Lemma 1) and our scheme (Theorem 2) are plotted in Figure 6. Again, the upper bound is significantly higher than both achievable rates and is not shown on the plot. Although the gains are marginal, that any gains are possible is surprising.

The next example demonstrates that there exists cases where computation coding is useful for sending a non-linear function over a noisy non-linear channel.

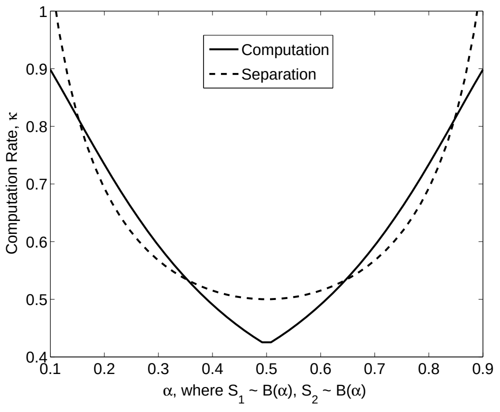

Fig. 5. Computing parity over a binary multiplying channel

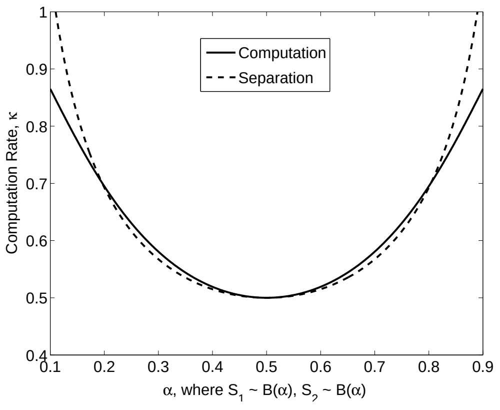

Fig. 6. Computing a binary product over a mod-2 adder

Example 8: S_1 and S_2 are independent sources drawn uniformly from $\{0, 1, 2\}$. We would like to know whether or not S_1 and S_2 are equal: $U = \mathbb{1}(S_1 \neq S_2)$. We can losslessly recover U from the linear function $V = S_1 \oplus_3 2S_2$ over GF(3). The channel is just $W = \mathbb{1}(S_1 \neq S_2)$ followed by a BSC with transition probability 0.1 to give Y. We employ Theorem 2. Our uncoded phase uses the sources directly; there is no remapping. In the update phase, we send V. With this strategy, we get a computation rate of $\kappa_{\text{COMP}} = 0.194$. Lemma 1 gives that the best separation-based computation rate is $\kappa_{\text{SEP}} = 0.168$. Finally, using Lemma 13 we get an upper bound of $\kappa_{\text{JOINT}} = 0.578$.

B. Detailed Example: Simple Sensor Network

We now give an example that demonstrates the usefulness of systematic computation coding for computing over a Gaussian MAC. By using a systematic computation code, we can balance between using the structure of the MAC to compute functions and applying the optimal input distribution. The former is achieved in the uncoded phase and the latter is a result of the separation-based update phase.

There are M sources, each drawn uniformly and independently from $\{-1, 1\}$. The encoders must satisfy average power

	$N = 0.25$			$N = 1$			$N = 10$		
M	COMP	SEP	UPPER	COMP	SEP	UPPER	COMP	SEP	UPPER
2	0.8663	0.7925	1.0566	0.3555	0.3962	0.5263	0.0458	0.0658	0.0877
10	0.7200	0.2679	0.9898	0.1692	0.1730	0.6391	0.0222	0.0500	0.1847
20	0.6583	0.1585	0.9882	0.1307	0.1098	0.6846	0.0161	0.0396	0.2471
30	0.6226	0.1153	0.9883	0.1138	0.0826	0.7077	0.0132	0.0333	0.2857
40	0.5972	0.0916	0.9885	0.1036	0.0670	0.7224	0.0114	0.0290	0.3131

TABLE I
COMPUTATION RATES FOR RELIABLE SUMMATION OF BINARY SOURCES OVER A GAUSSIAN MAC

constraints:

$$\frac{1}{n}\sum_{i=1}^{n} x_j[i]^2 \leq 1 \tag{25}$$

for $j = 1, 2, \ldots, M$. The channel output is just the sum of the channel inputs plus independent Gaussian noise:

$$Y[i] = \sum_{j=1}^{M} X_j[i] + Z[i], \tag{26}$$

where $\{Z[i]\}_{i=1}^{\infty}$ is an i.i.d. Gaussian sequence with mean 0 and variance N. We would like to reliably compute the real sum of the sources $U = \sum_{j=1}^{M} S_j$ at the decoder. This example can be viewed as a simple sensor network model.

For the uncoded part of our scheme, we simply plug the sources in as channel inputs and collect the channel outputs at the decoder as T^k. Choose $\mathbb{F}$ such that $|\mathbb{F}| > M$. Let $c_j(s_j) = \frac{(s_j+1)}{2}$ $\forall j$ and $g(b_1, b_2, \ldots, b_M) = b_1 \oplus b_2 \oplus \cdots \oplus b_M$ where addition is over $\mathbb{F}$. Let $d(v) = 2v - M$. By applying the results of Theorem 2, we can get the computation rates given in Table I. The best separation-based performance is determined using Lemma 1 and the upper bound is from Lemma 12.

For $N = 0.25$, our scheme outperforms separation-based coding by a nice margin. For $N = 1$, our scheme only outperforms separation as the number of users becomes large. Finally, for $N = 10$, our scheme is always dominated by separation. Note that these are only achievable rates for one particular systematic scheme. By adjusting the constellation used in the uncoded phase, we can certainly create a better code. We could also adjust the power allocation between the first and second phases of our systematic scheme to improve performance. In the above example, we used the same average power per symbol across the codeword.

Remark 12: As shown in Example 5, sometimes remapping the MAC to a linear MAC outperforms systematic coding. This does not appear to be the case for the Gaussian MAC. Converting the Gaussian MAC to a discrete linear MAC requires using an input pdf that is far from capacity achieving. In many regimes, the structural gains obtained from computation coding over a transformed Gaussian MAC are overshadowed by the rate penalty due to the bad input distribution.

The above examples show that computation coding is useful even when the desired function and the channel are not addition over a finite field. However, it is clear that our systematic scheme is not ideally matched to the Gaussian MAC, even if it can give significant gains. This is partially because our sources are binary but the channel has a continuous input alphabet. A more natural problem to consider for the Gaussian MAC is summing Gaussian sources with a mean-squared error criterion. In the next section, we develop a computation coding scheme based on lattices for this problem.

VI. LATTICE COMPUTATION CODING

Channel structure can also be exploited for communicating continuous-valued functions over MACs. One natural example is sending the sum of Gaussian sources over a Gaussian MAC. Here, as in the finite field case, we can use the addition performed by the MAC to our advantage with structured codes. Our problem statement is nearly identical to that in Section II except that we are willing to tolerate some distortion in our reconstructed function.

A. Problem Statement

Each encoder, $\mathcal{E}_j$, sees an i.i.d. Gaussian source $S_j \sim \mathcal{N}(0, \sigma_S^2)$ and faces one terminal of a Gaussian MAC. The encoders must satisfy average power constraints:

$$\frac{1}{n}\sum_{i=1}^{n} x_j[i]^2 \leq P \quad \forall j \in \{1, 2, \ldots, M\}. \tag{27}$$

The channel output is just the sum of the channel inputs plus independent Gaussian noise:

$$Y[i] = \sum_{j=1}^{M} X_j[i] + Z[i], \tag{28}$$

where $\{Z[i]\}_{i=1}^{\infty}$ is an i.i.d. Gaussian sequence with mean 0 and variance N. For every k source symbols, we are allocated $n = \ell k + r$ channel uses where $\ell, r \in \mathbb{Z}_+$ and $r < k$. Recall that $\kappa = \frac{k}{n}$.

Our goal is to reconstruct the sum of the sources, $U = S_1 + S_2 + \cdots + S_M$, at the decoder with the lowest possible distortion. Distortion is measured by the usual mean-squared error criterion:

$$D = \frac{1}{k}\sum_{i=1}^{k} E[(U_i - \hat{U}_i)^2]. \tag{29}$$

Lemma 5: If $\kappa = 1$, then uncoded transmission is optimal for sending the sum of i.i.d. Gaussian sources over a Gaussian MAC and achieves distortion

$$D_{\text{UNC}} = M\sigma_S^2 \frac{N}{N + MP}. \tag{30}$$

Proof: (*Achievability.*) At each encoder, simply feed a source symbol, scaled to meet the power constraint, into the channel at each time step. At the decoder, we compute the minimum-mean squared error (MMSE) estimate of U. This results in the following distortion:

$$\begin{aligned} D &= E[(U-\hat{U})^2] \\ &= M\sigma_S^2 - \frac{(E[UY])^2}{E[Y^2]} \\ &= M\sigma_S^2 \frac{N}{N+MP} \end{aligned}$$

(*Converse*.) We use the upper bound from Lemma 14 in Appendix II. Recall that $\kappa = 1$.

$$\begin{aligned} \kappa &\leq \frac{C_{\text{MAC}}}{R(D)} \\ \log\left(\frac{M\sigma_S^2}{D}\right) &\leq \log\left(1+\frac{MP}{N}\right) \\ D &\geq M\sigma_S^2 \frac{N}{N+MP} \end{aligned}$$

■

Our uncoded scheme achieves the lower bound on distortion. Therefore, uncoded transmission is optimal for sending sums of Gaussians over a Gaussian MAC. Essentially, this is the distributed computation extension of the famous fact that uncoded transmission is optimal for sending a Gaussian source over an additive white noise Gaussian (AWGN) channel. In the point-to-point setting, we are able to use a separation theorem to send a Gaussian source over an AWGN channel optimally for any ratio of channels uses to source symbols. However, for computing a sum over a Gaussian MAC, there is no separation theorem. As in the discrete case, we can improve our communication system by taking advantage of the structure of the MAC.

We now give a lower bound on the achievable distortion for any joint source-channel scheme for this problem.

Corollary 5: The achievable distortion for sending a Gaussian sum over a Gaussian MAC is lower bounded by

$$D_{\text{LOWER}} = M\sigma_S^2 \left(\frac{N}{N+MP}\right)^{\frac{1}{\kappa}}. \quad (31)$$

This bound is an immediate consequence of Lemma 14 in Appendix II.

B. Lattices for Computation

Lattice codes are the AWGN equivalent of linear codes for DMCs. A great deal work has gone into showing that lattice codes can achieve capacity on an AWGN channel and the rate distortion bound for a Gaussian source [18], [35]–[39]. Like linear codes, the appeal of lattices was their lower complexity compared to purely random coding strategies. However, their structural properties can also be exploited for distributed computation. First, we will need some definitions from [18].

Definition 14: An n-dimensional *lattice*, Λ, is a set of points in $\mathbb{R}^n$ such that if $\mathbf{x},\mathbf{y} \in \Lambda$, then $\mathbf{x}+\mathbf{y} \in \Lambda$, and if $\mathbf{x} \in \Lambda$, then $-\mathbf{x} \in \Lambda$. A lattice can always be written in terms of a generator matrix $\mathbf{G} \in \mathbb{R}^{n \times n}$:

$$\Lambda = \{\mathbf{x} = \mathbf{z}\mathbf{G} : \mathbf{z} \in \mathbb{Z}^n\}, \quad (32)$$

where $\mathbb{Z}$ represents the integers.

Definition 15: A *lattice quantizer* is a map, $Q : \mathbb{R}^n \to \Lambda$, that sends a point, $\mathbf{x}$, to the nearest lattice point in Euclidean distance:

$$\mathbf{x_q} = Q(\mathbf{x}) = \arg\min_{\mathbf{v}\in\Lambda} ||\mathbf{x}-\mathbf{v}||_2. \quad (33)$$

Definition 16: Let $\mathbf{x} \bmod \Lambda = \mathbf{x} - Q(\mathbf{x})$. For all $\mathbf{x},\mathbf{y} \in \mathbb{R}^n$, the mod Λ operation satisfies:

$$((\mathbf{x} \bmod \Lambda) + \mathbf{y}) \bmod \Lambda = (\mathbf{x}+\mathbf{y}) \bmod \Lambda. \quad (34)$$

Definition 17: The *fundamental Voronoi region*, $\mathcal{V}$, of a lattice, is the set of all points that are closest to the zero vector: $\mathcal{V}_0 = \{\mathbf{x} : Q(\mathbf{x}) = \mathbf{0}\}$.

Definition 18: The *normalized second moment* of a lattice is:

$$G(\Lambda) = \frac{1}{k}\frac{\int_{\mathcal{V}} ||\mathbf{x}||^2 d\mathbf{x}}{\left[\int_{\mathcal{V}} d\mathbf{x}\right]^{1+\frac{2}{k}}}. \quad (35)$$

Erez, Litsyn and Zamir showed in [38] that there exist lattices that are simultaneously good source codes and good channel codes. These will be extremely useful in our distributed refinement scheme.

Lemma 6 (Erez-Litsyn-Zamir): There exist sequences of lattices, Λ_k, such that the normalized second moment, $G(\Lambda_k)$, is asymptotically optimal:

$$\lim_{k\to\infty} G(\Lambda_k) = \frac{1}{2\pi e}, \quad (36)$$

and with high probability an i.i.d. Gaussian sequence, $\{S[i]\}_{i=1}^k$, with mean zero and variance $G(\Lambda_k)$ falls within the fundamental Voronoi region, $\mathcal{V}_{0,k}$:

$$\lim_{k\to\infty} \Pr\left(\{S[i]\}_{i=1}^k \in \mathcal{V}_{0,k}\right) = 1. \quad (37)$$

See [38] for a full proof.

In [18], Kochman and Zamir develop an elegant joint source-channel lattice scheme for sending a Wyner-Ziv Gaussian source over a dirty paper channel. Our distributed refinement scheme consists of two main steps. First, we use uncoded transmission to send a noisy sum to the decoder. Then, we have each encoder run a version of the Kochman-Zamir scheme targeted at the desired sum, U. Unfortunately, there is a penalty for this form of distributedness. The lattice at each encoder results in channel outputs that violate the power constraint by a factor of M. Therefore, we must scale down our inputs to meet the power constraint and accept the resulting increase in distortion at the decoder. Still, in many cases of interest, our scheme outperforms separation as we will see in Section VI-C.

Theorem 3: For $n = \ell k,\ \ell \in \mathbb{Z}_+$, the following distortion is achievable for sending a Gaussian sum over a Gaussian MAC so long as $P > \frac{M-1}{M}N$:

$$D_{\text{LAT}} = M\sigma_S^2 \left(\frac{N}{N+MP}\right)\left(\frac{MN}{N+MP}\right)^{\ell-1}. \tag{38}$$

Proof: We will first show the achievable scheme for $\ell = 2$. We thus have $2k$ channel uses to convey k sums. We will use the first k channel uses for an uncoded transmission phase as in Lemma 5. The decoder will then form an MMSE estimate $\hat{U}$ of the sum $U = S_1 + \cdots + S_M$ and use this as side information for the next phase. Thus, $U = Q + \hat{U}$ where Q is an i.i.d. Gaussian sequence with mean 0 and variance σ_Q^2 where

$$\sigma_Q^2 = M\sigma_S^2 \frac{N}{N+MP}. \tag{39}$$

Choose a sequence of good lattices, Λ_k, using Lemma 6 and scale them such that the normalized second moment of the lattice is MP. Let $\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_M$ be independent dither vectors drawn uniformly over the fundamental Voronoi region, $\mathbf{d}_j \sim \text{Unif}(\mathcal{V}_{0,k})$, and made available to the encoders and decoder.

Each encoder transmits $\frac{1}{\sqrt{M}}\mathbf{x}_j$ where:

$$\mathbf{x}_j = [\gamma \mathbf{s}_j + \mathbf{d}_j] \bmod \Lambda_k. \tag{40}$$

The channel output is given by:

$$\mathbf{y} = \frac{1}{\sqrt{M}} \sum_{j=1}^{M} \mathbf{x}_j + \mathbf{z}.$$

The decoder then computes:

$$\begin{aligned}
\mathbf{t} &= \alpha \mathbf{y} - \left(\sum_{j=1}^{M} \mathbf{d}_j + \gamma \hat{\mathbf{u}}\right) \\
\mathbf{r} &= \mathbf{t} \bmod \Lambda_k \\
&= \left[\frac{\alpha}{\sqrt{M}} \sum_{j=1}^{M} \mathbf{x}_j + \alpha \mathbf{z} - \sum_{j=1}^{M} (\mathbf{d}_j + \gamma \mathbf{s}_j) + \gamma \mathbf{q}\right] \bmod \Lambda_k \\
&= \left[\left(\frac{\alpha}{\sqrt{M}} - 1\right) \sum_{j=1}^{M} \mathbf{x}_j + \alpha \mathbf{z} + \gamma \mathbf{q}\right] \bmod \Lambda_k.
\end{aligned}$$

If the second moment of the term inside the modulo operation does not exceed MP, the second moment of the lattice, then we can guarantee that:

$$\lim_{k\to\infty} \Pr\left(\mathbf{r} = \left(\frac{\alpha}{\sqrt{M}} - 1\right) \sum_{j=1}^{M} \mathbf{x}_j + \alpha \mathbf{z} + \gamma \mathbf{q}\right) = 1. \tag{41}$$

See [36] for a detailed discussion of the effect of the dither in this step. The second moment can be controlled by requiring that:

$$\left(\frac{\alpha}{\sqrt{M}} - 1\right)^2 (M^2 P) + \alpha^2 N + \gamma^2 \sigma_Q^2 \le MP. \tag{42}$$

This equation will be satisfied by our final choice of the constants α and γ. The decoder's estimate of the sum is given by:

$$\begin{aligned}
\hat{\hat{\mathbf{u}}} &= \beta \mathbf{r} + \hat{\mathbf{u}} \\
&= \beta \left(\left(\frac{\alpha}{\sqrt{M}} - 1\right) \sum_{j=1}^{M} \mathbf{x}_j + \alpha \mathbf{z} + \gamma \mathbf{q}\right) + \hat{\mathbf{u}} \\
&= \beta \left(\left(\frac{\alpha}{\sqrt{M}} - 1\right) \sum_{j=1}^{M} \mathbf{x}_j + \alpha \mathbf{z}\right) - (1 - \beta\gamma)\mathbf{q} + \mathbf{u}.
\end{aligned}$$

This estimate gives the following mean-squared error:

$$D = \beta^2 \left(\left(\frac{\alpha}{\sqrt{M}} - 1\right)^2 M^2 P + \alpha^2 N\right) + (1 - \beta\gamma)^2 \sigma_Q^2.$$

We define the following constants:

$$\begin{aligned}
\alpha &= \frac{MP\sqrt{M}}{MP+N} \\
\gamma_0 &= \sqrt{\frac{MP}{\sigma_Q^2}\left(1 - \frac{MN}{MP+N}\right)},
\end{aligned}$$

and let $\gamma \to \gamma_0$ from below as $k \to \infty$. This ensures that Equation (42) is always satisfied. We also set:

$$\beta = \frac{\sigma_Q^2 \gamma}{MP}.$$

As $k \to \infty$, we get that the achieved distortion is:

$$D = M\sigma_S^2 \frac{N}{N+MP} \frac{MN}{N+MP}. \tag{43}$$

This proves the theorem for $\ell = 2$. For all higher values of ℓ, the scheme can be repeated with the final estimate from the last refinement taken as side information for the next stage. ■

Remark 13: A simple achievable scheme for situations where we do not have an integer number of channel uses per source symbol is to time share between two integers whose average gives the proper ratio.

Remark 14: Our scheme is only applicable to the "bandwidth expansion" case; i.e. we have more channel uses than source symbols.

Remark 15: This joint source-channel scheme can be easily generalized so that we can send a linear function of Gaussian sources instead of just a sum. Essentially, the γ coefficient in (40) should be replaced by $\phi_j \gamma$ at encoder j where ϕ_j is the desired coefficient for that source ($U = \sum_{j=1}^{M} \phi_j S_J$).

C. Separation-Based Scheme

We now give the best separation-based scheme for sending a Gaussian sum over a Gaussian MAC.

Corollary 6: The best achievable distortion for a separation-based scheme for sending a Gaussian sum over a Gaussian MAC is given by:

$$D_{\text{SEP}} = M\sigma_S^2 \left(\frac{N}{N+MP}\right)^{\frac{1}{\kappa M}}. \tag{44}$$

This is an immediate consequence of Lemma 11 in Appendix I-B.

For comparison, note that with repetition coding we can achieve the following distortion:

$$D_{\text{REP}} = M\sigma_S^2 \frac{N}{N + \ell MP}, \tag{45}$$

where $n = \ell k$ for some $\ell \in \mathbb{Z}_+$.

Example 9: Let $M = 5$, $P = 3$, $N = 1$, and $\sigma_S^2 = 1$. The end-to-end distortions for repetition coding (45), separation-based coding (Corollary 6), computation coding (Theorem 3), and our lower bound (Corollary 5) are given in Figure 7. Note that as the number of channel uses per source symbol increases, our scheme performs exponentially better than separation-based coding.

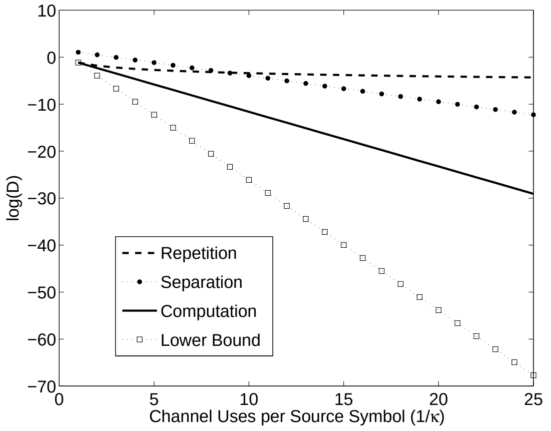

Fig. 7. Refining the Sum of Gaussian Sources over a Gaussian MAC, $M = 5$, $P = 3$, $N = 1$, $\sigma_S^2 = 1$

D. Computation Rate

We now examine the performance of our scheme in terms of its computation rate. The results in the previous two sections were given in terms of distortion per number of channel uses. This can be thought of as a type of distortion-rate function. For a better comparison to our results on reliable computation, we now define a rate-distortion function for distributed refinement.

Definition 19: The *computation rate-distortion function*, $\kappa^*(D)$, is the highest number of functions per channel use that can be conveyed to the decoder with average distortion D. Let $d(u, \hat{u})$ be the distortion measure of interest. A computation rate $\kappa(D) = \frac{k}{n}$ is achievable if for k large enough there exist encoders and a decoder:

$$\mathcal{E}_j : \mathcal{S}_j^k \to \mathcal{X}_j^n \tag{46}$$
$$\mathcal{D} : \mathcal{Y}^n \to \mathcal{U}^k, \tag{47}$$

such that:

$$\hat{U}^k = \mathcal{D}(Y^n)$$
$$\frac{1}{k}\sum_{i=1}^{k} E[d(U[i], \hat{U}[i])] = D. \tag{48}$$

An upper bound on the joint computation rate-distortion function and the separation-based computation rate-distortion function are given below:

$$\kappa^*_{\text{UPPER}}(D) = \frac{\log\left(1 + \frac{MP}{N}\right)}{\log\left(\frac{M\sigma_S^2}{D}\right)} \tag{49}$$
$$\kappa^*_{\text{SEP}}(D) = \frac{\log\left(1 + \frac{MP}{N}\right)}{M\log\left(\frac{M\sigma_S^2}{D}\right)}. \tag{50}$$

Note that our upper bound is M times higher that the best separation-based rate just as in the reliable case. We now give the computation rates for computation coding and repetition coding for the case when $n = \ell k$ for some $\ell \in \mathbb{Z}_+$:

$$\kappa_{\text{COMP}}(D) = \frac{\log\left(1 + \frac{MP}{N}\right) - \log M}{\log\left(\frac{M\sigma_S^2}{D}\right) - \log M} \tag{51}$$
$$\kappa_{\text{REP}}(D) = \frac{DMP}{N(M\sigma_S^2 - D)}. \tag{52}$$

For rates that do not satisfy $n = \ell k$ we can find the computation-rate distortion function by time-sharing between points that do satisfy the constraint.

Example 10: Let $M = 5$, $P = 4$, $N = 1$, and $\sigma_S^2 = 1$. The computation rates for our upper bound (49), computation coding (51), separation-based coding (50), and repetition (52) are given in Figure 8. We only plot computations rates lower than 1 as our scheme is only useful in the bandwidth expansion regime ($0 \leq \kappa \leq 1$.)

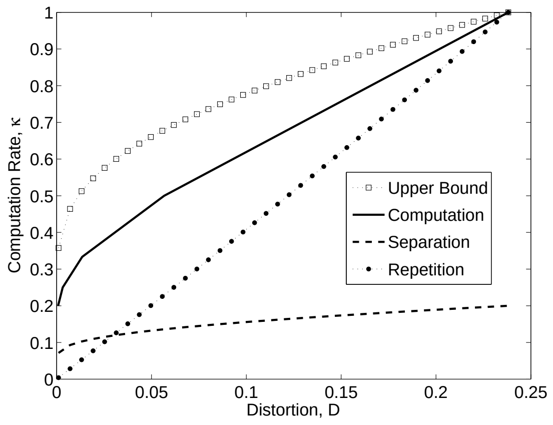

Fig. 8. Computation Rates for Sending the Sum of Gaussian Sources over a Gaussian MAC, $M = 5$, $P = 4$, $N = 1$, $\sigma_S^2 = 1$

Our lattice computation coding scheme performs significantly better than the best separation-based scheme so long as the power per user exceeds the noise variance. However, the distortion achieved by our scheme is also much higher than that given by the lower bound. Future work will focus on closing this gap. We now turn to an application of computation coding to give the capacity for a particular network coding problem.

VII. Multicasting over Finite Field MAC Networks

In this section, we demonstrate that computation coding is not only useful for problems that explicitly require the distributed computation of functions; it is also useful for achieving the capacity of certain networks. We look at a class of networks that include linear MACs for which computation coding is a natural way to find the multicast capacity. The linear MACs act as coding points when needed and computation coding is used to ensure that the resulting outputs are reliable.

Our setup is very similar to the network coding problem [13], with the addition of MACs. The usual network, $\mathcal{G}$, considered in a network coding problem consists of the following elements:

1) $\mathcal{V}$: the encoder/decoder nodes in the network. Each node, v, has a unique label taken from the integers.
2) $\mathcal{E}$: the directed, point-to-point, noiseless links. Each link is labelled by a triple (v_1, v_2, r) where v_1 and v_2 are the source and destination node labels and $r \in \mathbb{R}_+$ is the capacity.
3) v^S: the source node. One element of $\mathcal{V}$.
4) $(v_1^R, v_2^R, \ldots, v_L^R)$: the receiver nodes. Each one is an element of $\mathcal{V}$.

See Figure 9(b) for an example of such a network.

Definition 20: A multicast rate, R, is *achievable* if $\forall \epsilon \in (0,1)$ and n large enough there exist encoding and decoding functions for the network such that:

$$\begin{aligned} &\hat{W}_\ell = f_{v_\ell^R}(Y^n_{v_\ell^R}) \\ &\Pr\Big(\{\hat{W}_1 \neq W\} \cup \cdots \cup \{\hat{W}_L \neq W\}\Big) < \epsilon, \end{aligned} \tag{53}$$

where $W \in \{1, 2, \ldots, 2^{nR}\}$.

Definition 21: The *multicast capacity* is the supremum of all achievable multicast rates.

A simple upper bound is the max-flow min-cut theorem of Ford and Fulkerson. In [13], it was shown that for a network of point-to-point channels the multicast capacity is given by the max-flow min-cut theorem. For each receiver, calculate the maximum information flow across all cuts that separate the source node from that receiver. The multicast capacity is the minimum of all these max-flow values taken over all cuts and receivers. In [14] and [15], it was shown that linear encoding and decoding over a finite field is sufficient to achieve the multicast capacity. Bounds are also given on the required field size. For our computation coding scheme to work, we will need that the field used by the network code is identical to the field of the MACs. It was independently and concurrently shown by Ho et al. in [40], Jaggi et al. in [41], and Sanders et al. [42] that the field size only needs to be larger than the number of receivers. We reproduce the version from [40] below.

Lemma 7 (Ho et al.): Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a standard network of noiseless, directed, rate-limited links with a single source and L receivers. The multicast capacity is given by the max-flow min-cut bound and can be achieved by an algebraic network code over any finite field larger than L ($\mathbb{F}_q, \;\; q > L$). The encoding function of each node (except the receivers) is a memoryless, linear combination of all its inputs:

$$X_v[i] = \sum_j^M \alpha_{vj} Y_{vj}[i]. \tag{54}$$

where $Y_{vj}[i]$ is the value seen by node v at time i on its j^{th} incoming edge and $Y_{vj}[i], \alpha_{vj} \in \mathbb{F}_q$ for all $v \in \mathcal{V} \setminus \{v_1^R, v_2^R, \ldots, v_L^R\}$.

For a full proof, see [43].

A. Problem Setup

First, we constrain all MACs in our network to be linear with respect to the same field, $\mathbb{F}$. We also assume that the MAC noise processes are independent of each other. This restriction on the MACs is necessary to give tight upper bounds on the multicast capacity. Note that computation coding can be beneficial even if some of the MACs are nonlinear [20]. However, our primary objective in this section is to show that computation coding is sometimes a useful tool to get all the way to capacity.

A finite field MAC network, $\mathcal{G}_{\text{MAC}}$, consists of the following elements:

1) $\mathcal{V}_N$: the encoder/decoder nodes of the network. Each node, v, has a unique label taken from the integers ($v \in \mathbb{Z}_+$) and consists of a decoding function $g_{v_j v}$ for each incoming edge (v_j, v) and an encoding function $f_{v v_k}$ for each outgoing edge (v, v_k).
2) v^S: the source node. One element of $\mathcal{V}_N$.
3) $(v_1^R, v_2^R, \ldots, v_L^R)$: the receiver nodes. Each one is an element of V.
4) $\mathcal{V}_{\text{MAC}}$: the MACs in the network. Each MAC, m, has a unique label (even from the elements of $\mathcal{V}_N$) taken from the integers ($m \in \mathbb{Z}_+$). We also define a function $c(\cdot)$ such that $c(m)$ equals the maximum sum rate of MAC m.
5) $\mathcal{E}_{NN}$: the directed point-to-point channels in the network. Each channel is labelled by a triple (v_1, v_2, r) where v_1 and v_2 are the source and destination labels and r is the capacity.
6) $\mathcal{E}_{NM}$: the input edges from nodes to MACs. Each edge is represented by a pair (v, m) which means that node v has an input into MAC m.
7) $\mathcal{E}_{MN}$: the output edges from a MAC to a node. To simplify our main proof, we assume that each MAC only has an input into one node. Note that by introducing intermediate nodes in the graph we can exactly simulate the effect of the MAC output broadcasting to multiple

nodes. Each edge is given by a pair (m, v) which means that the output of MAC m is fed into node v.

8) $X_{v_j v_k}[i]$: the channel input on the edge (v_j, v_k) at time i.
9) $Y_{v_j v_k}[i]$: the channel output on the edge (v_j, v_k) at time i.

See Figure 9(a) for an example of such a network. Achievability and capacity for multicasting over a finite field MAC network are defined identically as in the original network coding problem (Definitions 20 and 21).

Remark 16: Note that the linear MACs that are used in this network have a capacity region that can be described by a simplex. In other words, a single user can reach the optimal input distribution on its own. Thus, all of the benefits from using our codes in this problem are due to structural gain not collaborative gain.

B. Multicast Capacity

We will now show that the max-flow min-cut bound is achievable even if there are linear MACs in the network. Note that in the cut between each receiver and the source, each MAC is evaluated based on its sum rate. At first, this suggests that our outer bound is loose since the receivers may compete for part of each MAC's sum rate. However, by incorporating computation coding into an overall network code, we can exploit the operation of each MAC for our network code. If we assume that the MAC field size q is larger than the number of receivers in the network L, then we can find an achievable scheme that meets the outer bound.

Theorem 4: The max-flow min-cut bound for multicasting is achievable for a finite field MAC network, $\mathcal{G}_{\text{MAC}}$, if $|\mathbb{F}| > L$, where $|\mathbb{F}|$ is the MAC field size and L is the number of receivers in the network.

Proof: (*Achievability.*) First, we will transform our network of noisy MACs and point-to-point channels, $\mathcal{G}_{\text{MAC}}$, into a network of noiseless, point-to-point links, $\mathcal{G}'$. Then, we will find an appropriate network code using Lemma 7. Finally, we will map this network code to our original network using computation codes and classical channel codes.

The nodes, $\mathcal{V}'$, of our transformed network are the original encoder/decoder nodes plus one new encoder/decoder node for each original MAC:

$$\mathcal{V}' = \mathcal{V}_N \cup \mathcal{V}_{\text{MAC}}.$$

The noiseless links, $\mathcal{E}'$, of our transformed network come in three types. First, we have links between nodes that were originally connected by the point-to-point channels, $\mathcal{E}_{NN}$. The capacity of each of these links is given by the capacity of the original point-to-point channel. Second, we take the original MAC inputs from nodes, $\mathcal{E}_{NM}$, and convert these into links between the inputting nodes and the new stand-in MAC nodes. These links have infinite capacities. Finally, we take the original MAC outputs to nodes, $\mathcal{E}_{MN}$, and convert these into links between the new stand-in MAC nodes and the observing node. These links have capacities given by the sum rate capacity of the original MAC. This transformation can be summarized as:

$$\begin{aligned}\mathcal{E}' = \mathcal{E}_{NN} \cup \{(v, m, \infty) | (v, m) \in \mathcal{E}_{NM}\} \cdots \\ \cdots \cup \{(m, v, c(m)) | (m, v) \in \mathcal{E}_{MN}\}.\end{aligned}$$

The new network, $\mathcal{G}'$ has the same max-flow min-cut characterization as our original network, $\mathcal{G}_{\text{MAC}}$. Choose $R < C$ where C is the multicast capacity of $\mathcal{G}'$ and choose $\epsilon > 0$. Using Lemma 7, we can find an algebraic network code over the field $\mathbb{F}$ that achieves a multicast rate R over $\mathcal{G}'$ with an error probability less than $\frac{\epsilon}{2}$. This algebraic network code describes an input-output relationship for each node that we will duplicate on our original MAC network over a long block length.

Let $K = |\mathcal{E}_{NN}| + |\mathcal{E}_{MN}|$ be the number of channels in the original network, $\mathcal{G}_{\text{MAC}}$. For each stand-in MAC node in the transformed network we create a computation code for the MAC, $m \in \mathcal{V}_{\text{MAC}}$, in the original network. The computation code is targeted at the linear function over $\mathbb{F}$ used to process inputs in the stand-in MAC node. This linear function, U_m, can be sent over the MAC at any computation rate $\kappa < \frac{c(m)}{H(U_m)}$ with an error probability less than $\frac{\epsilon}{2K}$ using Theorem 1.

Our scheme uses the network Bn times, divided into B blocks each of length n. In a block of length n, each node, $v \in \mathcal{V}_N$, takes the received sequence of channel symbols from the previous block and decodes them to determine the messages sent from all incoming links. The messages are assumed to be sequences of values over $\mathbb{F}$. It then takes these messages and computes the linear functions assigned to it in the transformed network. The output of these functions are then sent over the appropriate point-to-point channels in $\mathcal{E}_{NN}$ using a capacity-achieving code with error probability $\frac{\epsilon}{2K}$. The functions intended for stand-in MAC nodes in the transformed network are mapped to the appropriate computation codes designed above. Thus, over a long block, each node and MAC emulates the function assigned to it in the transformed network. Since the rate assignments in the transformed network are appropriately chosen, all functions can be sent without violating their respective channel capacities. However, in the initial blocks, not every node sees incoming symbols owing to delays in the network. By choosing B (and n) large enough, we can overcome this delay and approach the target rate.

Finally, we get that the network code is successful if the algebraic network code is successful and no block errors are made on any channel in the network. By the union bound, we get that this error probability is less than ϵ which completes the proof.

(*Converse.*) Since our network transformation can only increase the multicast capacity of the network and we can achieve any rate less than the transformed capacity, we get that our scheme meets the max-flow min-cut bound. ■

We have shown that for channel networks that include MACs, computation coding is helpful for multicasting. Specifically, for certain links, we are only interested in sending functions of the input bits. If the communications bottleneck on that link is a MAC, then computation coding can increase the overall network throughput. If all of the MACs in our network are perfectly matched to linear functions, which is the case with linear MACs, then we can give an overall network code that achieves the max-flow min-cut bound.

Overall, our results indicate that structured joint source-channel codes are necessary for the analysis of general network capacity, even if we are only interested in communicating bits. This is similar to the conclusion reached by Körner and Marton in [9] for the many-help-one source-coding problem. This may indicate that purely random coding techniques will not be able to achieve the capacity of a general network, some structure is needed. For a general network of multi-user channels, a complete analysis may be intractable due to the complications introduced by the various channel structures and the seeming necessity of structured codes.

C. Detailed Example: Butterfly Network

We now develop a simple example for which computation-based network coding provides a clear advantage over separation-based network coding. Consider the channel network in Figure 9(a). Each vertex on the graph represents a decoder/encoder pair. The labeled edges represent noiseless bit pipes each with capacity C. At the center of the graph is an linear MAC with inputs X_1 (from the left) and X_2 (from the right) and output Y given by $Y = X_1 \oplus_3 X_2 \oplus_3 Z$ where $\Pr(Z = 0) = 1 - q$ and $\Pr(Z = 1) = q$. We would like to determine the multicast capacity, C_N, of this network.

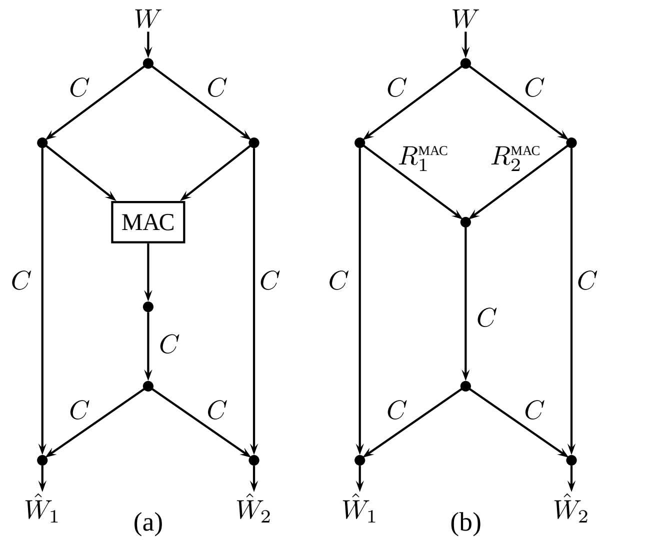

Fig. 9. (a) Multiple-Access Network Coding Example (b) Converting MAC into bit pipes

1) Separation-Based Network Coding: Given the network in Figure 9(a), one might choose to use standard channel coding strategies coupled with network coding. Using a multiple-access code, we can convert the network of noisy channels into a network of noiseless bit pipes (see Figure 9(b)). Then, we can find a network code for our network of bit pipes.

Using a standard MAC code, we can allocate a rate R_1^{MAC} for the left user and a rate R_2^{MAC} for the right user. The capacity region, $\mathcal{R}_{\text{MAC}}$, is the set of all rate pairs, (R_1, R_2), satisfying $R_1 + R_2 < 1 - h_B(q)$.

Lemma 8: For the channel network in Figure 9(b), the multicast capacity is

$$C_{\text{SEP}} = \min\left(2C, C + \frac{\log 3 - h_B(q)}{2}\right). \qquad (55)$$

The proof follows from a simple application of the max-flow min-cut bound and network coding. The key is to send the parity of the left and right bitstreams down the center path. See [13] for more details.

2) Computation Coding: If we only consider the structure of the function for source compression and treat the MAC like two bit pipes we cannot achieve the optimum performance. By using a linear network code and a linear channel code, we can take advantage of the channel's natural operation to reliably compute a function for the network code.

Corollary 7: For the channel graph from Figure 9(a) with a mod-3 adder MAC, the multicast capacity is

$$C_N = \min\left(2C, C + \log 3 - h_B(q)\right). \qquad (56)$$

This is an immediate consequence of Theorem 4. Thus, incorporating the natural function computed by the MAC into a network code can outperform a separation-based scheme.

Remark 17: The field size requirements in Theorem 4 are not tight. If we placed an M2MAC in the center of Figure 9 (a), we could still achieve the max-flow min-cut bound by sending mod-2 sums down the center path using the code from Corollary 3.

Remark 18: We can find more examples of computation coding beating separation by incorporating the MACs from Example 5 or Section V-B and determining the computation rate for sending a mod-2 sum over these channels. See [20] for more details.

D. Discussion

We now have that channel-network separation cannot completely characterize the multicast capacity of networks that includes MACs. A similar conclusion was reached for networks that include deterministic broadcast channels by Ratnakar and Kramer in [17]. Ramamoorthy et al. showed that source-channel separation does not hold for multicasting more than one source to multiple receivers in [44]. All of these results imply that both structural considerations and source dependencies are necessary to characterize the capacity of channel networks. For general channel models, these considerations may put the optimal solution out of reach given current tools. However, for certain classes of channel models it should be possible to give capacity results by choosing codes that are appropriately matched as was done for linear MACs in this section.

VIII. Conclusions

We have developed a tool, computation coding, that is useful for reliable distributed computation over MACs. This coding technique essentially consists of using the same structured source code (targeted at the desired function) at each encoder followed by a capacity-achieving channel code. For certain classes of channels, such as discrete linear MACs and the Gaussian MAC, structured channel codes can achieve the computation capacity. In these cases, our computation codes can take full advantage of the channel operation to compute the desired function: individual codewords can actually be merged directly on the channel. In these cases, computation coding does much better than separation, sometimes with gains that are proportional to the number of users. In other cases, we do not exploit the channel operation for the full duration of our scheme, only in an initial uncoded phase. The resulting systematic computation code shows that there are gains to be had even in mismatched cases such as binary addition over a binary multiplying channel in Example 6.

The underlying codes that make up our computation codes are essentially the same as the structured codes being developed for practical use. Furthermore, our schemes can be viewed as having a source coding part followed by a channel coding part; the main difference from separation-based coding is that we do not force a representation in bits, we only use the underlying alphabet of our desired function. The channel coding part of our computation codes remains unchanged if we change the desired function but leave the channel the same. Thus, our strategies are of relatively low complexity and are, in a sense, modular.

We have also shown through a case study that these coding strategies can be useful in analyzing the capacity of networks that include MACs. We suspect that similar applications of an optimal computation code for Gaussian MACs may yield insights into the capacity of AWGN networks. At the very least, it seems clear that structural considerations cannot be ignored if we want to characterize the capacity of large networks.

APPENDIX I
BOUNDS FOR SEPARATION-BASED SCHEMES

In this appendix, we will develop two inner bounds on the distributed compression rate region. The first is a bound on distributed, reliable computation. The second is a bound for computing the sum of independent Gaussian sources as in Section VI-C.

A. Separation-Based Reliable Computation

For the first bound, we will need a result of Orlitsky and Roche for computing with side information [10].

Let S_1 and S_2 be sources according to Definition 1 and let $f : \mathcal{S}_1 \times \mathcal{S}_2 \rightarrow \mathcal{U}$ be the desired function.

Definition 22: The elements of $\mathcal{S}_1$ are the vertices of the *characteristic graph*, G, of S_1, S_2, and f. Two distinct vertices, a and b, are connected if there is a $c \in \mathcal{S}_2$ such that $p_{S_1S_2}(a,c),\ p_{S_1S_2}(b,c) > 0$ and $f(a,c) \neq f(b,c)$. We say the graph is *complete* if each vertex is connected to every other vertex.

We say a set of vertices is independent if no two are connected. Let $\Gamma(G)$ be the collection of independent sets of the graph G.

Definition 23: The *conditional graph entropy* is given by:

$$H_G(S_1|S_2) \triangleq \min_{\substack{W - S_1 - S_2 \\ S_1 \in W \in \Gamma(G)}} I(W; S_1|S_2), \tag{57}$$

where $W - S_1 - S_2$ signifies a Markov chain.

Lemma 9 (Orlitsky-Roche): Two sources, S_1 and S_2, are generated from the joint pmf $p_{S_1S_2}$. An encoder observes S_1 and must send enough bits to a decoder that sees S_2, such that the decoder can reconstruct $U = f(S_1, S_2)$ with a vanishing probability of error:

$$\mathcal{E} : \mathcal{S}_1^k \rightarrow \{0,1\}^{kR} \tag{58}$$
$$\mathcal{D} : \{0,1\}^{kR} \times \mathcal{S}_2^k \rightarrow \mathcal{U}^k \tag{59}$$
$$\hat{U}^k = \mathcal{D}(\mathcal{E}(S_1^k), S_2^k)$$
$$\lim_{k\rightarrow\infty} P(\hat{U}^k \neq U^k) = 0. \tag{60}$$

This is possible iff:

$$R > H_G(S_1|S_2). \tag{61}$$

We will use this side information result to generate individual rate constraints on separation-basd schemes for distributed compression. There are M sources and a desired function $f(\cdot)$. Let $\mathbf{S}_j^C = (S_1, S_2, \ldots, S_{j-1}, S_{j+1}, \ldots, S_M)$.

Lemma 10: The rate required for each encoder of a separation-based scheme for distributed compression of $U = f(S_1, S_2, \ldots, S_M)$ is lower bounded by

$$R_j \geq H_G(S_j|\mathbf{S}_j^C) \quad \forall j \in \{1, 2, \ldots, M\}. \tag{62}$$

Proof: At encoder j, assume that all other sources are available at the decoder. Clearly, this can only decrease the rate required of encoder j. An application of Lemma 9 gives that a rate of $H_G(S_j|\mathbf{S}_j^C)$ is required from each encoder to reconstruct $f(S_1, S_2, \ldots, S_M)$ losslessly at the decoder. ■

Proof of Lemma 1: We need the conditional graph entropy at each encoder used in the proof for Lemma 10 above. The characteristic graph for each encoder is complete. Therefore, the independent sets are the singletons and $W = S_j$. It follows that $H_G(S_j|\mathbf{S}_j^C) = I(W; S_j|\mathbf{S}_j^C)) = H(S_j|\mathbf{S}_j^C) = H(S_j)$.

B. Separation-Based Gaussian Summation

We now show that if we want to reconstruct the sum of independent Gaussian sources by separation, the encoders can do no better than send their sources to the decoder.

Lemma 11: Let $S_1, S_2, \ldots, S_M$ be i.i.d. Gaussian sources with mean 0 and variance σ_S^2. There are M source encoders each observing one of the sources and conveying bits to a decoder that must reconstruct $U = S_1 + S_2 + \cdots + S_M$. Distortion is measured by the usual mean-squared error criterion: $D = E[(\hat{U} - U)^2]$. The sum rate distortion function is

$$R(D) = \frac{M}{2} \log\left(\frac{M\sigma_S^2}{D}\right). \tag{63}$$

Proof: (*Converse.*) Let $f_j : \mathbb{R}^k \to \{1, 2, \ldots, 2^{kR_j}\}$ be the j^{th} source encoding function and let $W_j = f_j\left(\{S_j[i]\}_{i=1}^k\right)$ be the message output by this encoder for a length k block of source symbols. Given $\mathbf{W} = (W_1, W_2, \ldots, W_M)$ at the decoder, the minimum-mean squared estimate (MMSE) of U is given by the conditional expectation.

$$\begin{aligned}
D &= \frac{1}{k}\sum_{i=1}^{k} E[(U[i] - \hat{U}[i])^2] \\
&\geq \frac{1}{k}\sum_{i=1}^{k} E[(U[i] - E[U(i)|\mathbf{W}])^2] \\
&\overset{(a)}{=} \frac{1}{k}\sum_{i=1}^{k} E\left[\left(\sum_{j=1}^{M} S_j[i] - E\left[\sum_{j=1}^{M} S_j[i]\Big|\mathbf{W}\right]\right)^2\right] \\
&\overset{(b)}{=} \frac{1}{k}\sum_{i=1}^{k} E\left[\left(\sum_{j=1}^{M} S_j[i] - E\left[\sum_{j=1}^{M} S_j[i]\Big|W_j\right]\right)^2\right] \\
&\overset{(c)}{=} \frac{1}{k}\sum_{j=1}^{M}\sum_{i=1}^{k} E[(S_j[i] - E[S_j[i]|W_j])^2] \\
&\overset{(d)}{\geq} \sum_{j=1}^{M} \sigma_S^2 2^{-2R_j}
\end{aligned}$$

(a) by linearity of expectation
(b), (c) by independence of S_i and S_j for all $i \neq j$
(d) by the single source rate distortion converse (see [22, pp. 350-351])

Minimizing the function $\sum_{j=1}^{M} \sigma_S^2 2^{-2R_j}$ is just a convex optimization problem subject to the convex constraint $\sum_{j=1}^{m} R_j = R$. It easily follows that the minimizing solution satisfies $R_1 = R_2 = \cdots = R_M$. We obtain:

$$\begin{aligned}
D &\geq M\sigma_S^2 2^{-2\frac{R}{M}} \\
R(D) &\geq \frac{M}{2}\log\left(\frac{M\sigma_S^2}{D}\right).
\end{aligned}$$

(*Achievability.*) Each encoder simply uses a standard Gaussian rate distortion code for its source with distortion target $D_j = \frac{D}{M}$. Such a code requires a rate of at least $\frac{1}{2}\log\left(\frac{M\sigma_S^2}{D}\right)$ per encoder. See [22, pp. 351-358] for the derivation of such a code. The decoder recovers each source and sums the individual estimates to get an estimate of the desired sum at distortion D. ■

APPENDIX II
COMPUTATION CAPACITY UPPER BOUNDS

In this appendix, we give two upper bounds on the computation capacity and one upper bound on the computation rate-distortion function. Our first bound comes from joining the encoders and reducing our problem to a point-to-point problem.

Definition 24: The *maximum joint sum rate* is the highest sum rate one can achieve on a MAC if the encoders are allowed to cooperate completely. It is given by:

$$C_{\text{JOINT}} = \max_{p(x_1, x_2, \ldots, x_M)} I(X_1, X_2, \ldots, X_M; Y). \tag{64}$$

Lemma 12: The reliable computation rate is upper bounded:

$$\kappa_{\text{JOINT}} \leq \frac{C_{\text{JOINT}}}{H(U)}. \tag{65}$$

The proof follows immediately from joining the encoders and applying the point-to-point separation theorem. See [22, p. 216] for a full proof of the point-to-point separation theorem.

Our second bound is for the case when the sources are independent. We assume that the multiple-access channel has a symmetric maximum sum rate, C_{MAC}, according to Definition 10. This assumption can be removed for a more general statement of the lemma below.

Lemma 13: If the sources are independent and the maximum sum rate of the MAC is symmetric then the reliable computation rate is upper bounded by

$$\kappa_{\text{IND}} \leq \frac{C_{\text{MAC}}}{H(U)}. \tag{66}$$

Proof: Let $P_e = \Pr(\hat{U}^k \neq U^k)$. By Fano's inequality, we can show that $H(U^k|Y^n) \leq 1 + kP_e \log|\mathcal{U}|$. Now, set $\lambda_k = \frac{1}{k} + P_e \log|\mathcal{U}|$.

$$\begin{aligned}
H(U) &= \frac{1}{k}H(U^k) \\
&= \frac{1}{k}(H(U^k) - H(U^k|Y^n) + H(U^k|Y^n)) \\
&= \frac{1}{k}(I(U^k; Y^n) + H(U^k|Y^n)) \\
&\leq \frac{1}{k}I(U^k; Y^n) + \lambda_k \\
&\leq \frac{1}{k}I(X_1^n, X_2^n, \ldots, X_M^n; Y^n) + \lambda_k
\end{aligned}$$

where the last step is due to the data processing inequality. From here we are free to apply the standard MAC converse (see [22, pp.399-402]):

$$\frac{k}{n} \leq \frac{I(X_1, X_2, \ldots, X_M; Y)}{H(U)}.$$

for some pdf of the form $\prod_{j=1}^{M} p_{X_j}(x_j)$. The result follows immediately. ■

It is also possible to give an upper bound that factors in the exact nature of the source correlations as in [45]. However, the focus of this paper is on the gains that can be achieved by exploiting the structure rather than the correlations. All of our examples have independent sources so such a bound is unnecessary for the scope of this paper.

Finally, we upper bound the computation rate-distortion function for the case when the sources are independent and the MAC has a symmetric maximum sum rate.

Lemma 14: If the sources are independent and the maximum sum rate of the MAC is symmetric the computation rate-distortion function is upper bounded:

$$\kappa(D) \leq \frac{C_{\text{MAC}}}{R_U(D)}. \tag{67}$$

where $R_U(D)$ is the rate-distortion function of the desired function, $U = f(S_1, S_2, \ldots, S_M)$.

Proof: Let $d(u, \hat{u})$ be the distortion measure and $\mathbf{S}[i] = (S_1[i], S_2[i], \ldots, S_M[i])$.

$$\begin{aligned}
kR_U(D) &= kR(E[d(U^k, \hat{U}^k)]) \\
&= kR_U\left(\frac{1}{k}\sum_{i=1}^{k} E[d(U[i], \hat{U}[i])]\right) \\
&\overset{(a)}{\leq} k\sum_{i=1}^{k}\frac{1}{k}R_U(E[d(U[i], \hat{U}[i])]) \\
&= \sum_{i=1}^{k} R_U(E[d(U[i], \hat{U}[i])]) \\
&\overset{(b)}{\leq} \sum_{i=1}^{k} I(\mathbf{S}[i]; \hat{U}_i) \\
&= \sum_{i=1}^{k} h(\mathbf{S}[i]) - \sum_{i=1}^{k} h(\mathbf{S}[i]|\hat{U}[i]) \\
&\overset{(c)}{\leq} \sum_{i=1}^{k} h(\mathbf{S}[i]) - \sum_{i=1}^{k} h(\mathbf{S}[i]|\hat{U}^k, \mathbf{S}[i-1], \ldots, \mathbf{S}[1]) \\
&\overset{(d)}{=} h(\mathbf{S}^k) - h(\mathbf{S}^k|\hat{U}^k) \\
&= I(\mathbf{S}^k; \hat{U}^k) \\
&\overset{(e)}{\leq} I(\mathbf{X}^n; Y^n).
\end{aligned}$$

(a) by convexity of $R(D)$
(b) by definition of $R(D)$
(c) since conditioning reduces entropy
(d) by the chain rule for entropy
(e) by the data processing inequality
From here we are free to apply the standard MAC converse (see [22, pp.399-402]):

$$\frac{k}{n} \leq \frac{I(X_1, X_2, \ldots, X_M; Y)}{R_U(D)}.$$

for some pdf of the form $\prod_{j=1}^{M} p_{X_j}(x_j)$. The result follows immediately. ∎

Acknowledgment

The authors would like to thank K. Eswaran for providing the proof of Lemma 11 and A. D. Sarwate for a careful reading of the manuscript. The authors would also like to thank K. Eswaran, V. Prabhakaran, and A. D. Sarwate for extremely valuable discussions as well as the anonymous reviewers for their helpful comments.

References

[1] T. Cover, A. El Gamal, and M. Salehi, "Multiple access channels with arbitrarily correlated sources," *IEEE Transactions on Information Theory*, vol. 26, pp. 648–657, November 1980.
[2] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.
[3] R. Ahlswede and T. S. Han, "On source coding with side information via a multiple-access channel and related problems in multi-user information theory," *IEEE Transactions on Information Theory*, vol. 29, pp. 396–412, May 1983.
[4] M. Gastpar and M. Vetterli, "Source-channel communication in sensor networks," in *2nd Intl. Symp. on Info. Proc. in Sensor Networks (IPSN '03)* (L. J. Guibas and F. Zhao, eds.), Lecture Notes in Computer Science, (New York, NY), pp. 167–177, Springer, April 2003.
[5] M. Gastpar, "Uncoded transmission is exactly optimal for a simple Gaussian "sensor" network," in *2nd Annual Workshop on Information Theory and its Applications, UCSD*, (La Jolla, CA), January 2007.
[6] G. Mergen and L. Tong, "Type based estimation over multiaccess channels," *IEEE Transactions on Signal Processing*, vol. 54, pp. 613–626, February 2006.
[7] W. U. Bajwa, A. M. Sayeed, and R. Nowak, "Matched source-channel communication for field estimation in wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, (Los Angeles, CA), pp. 332–339, April 2005.
[8] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 755–764, April 2005.
[9] J. Körner and K. Marton, "How to encode the modulo-two sum of binary sources," *IEEE Transactions on Information Theory*, vol. 25, pp. 219–221, March 1979.
[10] A. Orlitsky and J. R. Roche, "Coding for computing," *IEEE Transactions on Information Theory*, vol. 47, pp. 903–917, March 2001.
[11] H. Yamamoto, "Wyner-Ziv theory for a general function of the correlated sources," *IEEE Transactions on Information Theory*, vol. 28, pp. 803–807, September 1982.
[12] H. Feng, M. Effros, and S. Savari, "Functional source coding for networks with receiver side information," in *42rd Annual Allerton Conference on Communication, Control, and Computing*, (Monticello, IL), pp. 1419–1427, September 2004.
[13] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.
[14] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, pp. 371–381, February 2003.
[15] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 782–795, October 2003.
[16] L. Song, R. W. Yeung, and N. Cai, "A separation theorem for single-source network coding," *IEEE Transactions on Information Theory*, vol. 52, pp. 1861–1871, May 2006.
[17] N. Ratnakar and G. Kramer, "The multicast capacity of deterministic relay networks with no interference," *IEEE Transactions on Information Theory*, vol. 52, pp. 2425–2432, June 2006.
[18] Y. Kochman and R. Zamir, "Analog matching of colored sources to colored channels," in *Proceedings of the IEEE International Symposium on Information Theory*, (Seattle, WA), 2006.
[19] B. Nazer and M. Gastpar, "Reliable computation over multiple-access channels," in *43rd Annual Allerton Conference on Communication, Control, and Computing*, (Monticello, IL), September 2005.
[20] B. Nazer and M. Gastpar, "Computing over multiple-access channels with connections to wireless network coding," in *IEEE International Symposium on Information Theory*, (Seattle, WA), July 2006.
[21] B. Nazer and M. Gastpar, "Computation over Gaussian multiple-access channels," in *IEEE International Symposium on Information Theory*, (Nice, France), 2007.
[22] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley-Interscience, 1991.
[23] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, July 1973.
[24] R. Ahlswede, "Multi-way communication channels," in *Proceedings of 2nd International Symposium on Information Theory*, (Thakadsor, Armenian SSR), pp. 23–52, Publishing House of the Hungarian Academy of Sciences, 1971.

[25] H. Liao, *Multiple access channels*. PhD thesis, University of Hawaii, Honolulu, HI, September 1972.
[26] M. Gastpar, *To Code or Not To Code*. PhD thesis, Ecole Polytecnique Fédérale (EPFL), Lausanne, Switzerland, November 2002.
[27] M. Effros, M. Médard, T. Ho, S. Ray, D. R. Karger, R. Koetter, and B. Hassibi, “Linear network codes: A unified framework for source, channel, and network coding,” in *DIMACS Workshop on Network Information Theory*, (Piscataway, NJ), 2003.
[28] R. Gallager, *Information Theory and Reliable Communication*. New York: John Wiley and Sons, Inc., 1968.
[29] R. L. Dobrushin, “Asymptotic optimality of group and systematic codes for some channels,” *Theory of Probability and its Applications*, vol. 8, no. 1, pp. 47–59, 1963.
[30] I. Csiszár, “Linear codes for sources and source networks: Error exponents, universal coding,” *IEEE Transactions on Information Theory*, vol. 28, pp. 585–592, July 1982.
[31] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. New York: Academic Press, 1982.
[32] P. Elias, “Coding for noisy channels,” *IRE Convention Record*, vol. 4, pp. 37–46, 1955.
[33] S. Shamai and S. Verdú, “Capacity of channels with uncoded side information,” *European Transactions on Telecommunications*, vol. 6, pp. 587–600, Sept.-Oct. 1995.
[34] S. Shamai, S. Verdú, and R. Zamir, “Systematic lossy source/channel coding,” *IEEE Transactions on Information Theory*, vol. 44, pp. 564–579, March 1998.
[35] R. Urbanke and B. Rimoldi, “Lattice codes can achieve capacity on the AWGN channel,” *IEEE Transactions on Information Theory*, vol. 44, pp. 273–278, January 1998.
[36] U. Erez and R. Zamir, “Achieving $\frac{1}{2}\log(1+\mathrm{SNR})$ on the AWGN channel with lattice encoding and decoding,” *IEEE Transactions on Information Theory*, vol. 50, pp. 2293–2314, October 2004.
[37] R. Zamir, S. Shamai, and U. Erez, “Nested linear/lattice codes for structured multiterminal binning,” *IEEE Transactions on Information Theory*, vol. 48, pp. 1250–1276, June 2002.
[38] U. Erez, S. Litsyn, and R. Zamir, “Lattices which are good for (almost) everything,” *IEEE Transactions on Information Theory*, vol. 51, pp. 3401–3416, October 2005.
[39] S. Servetto, “Lattice quantization with side information,” in *Proceedings of the IEEE Data Compression Conference (DCC)*, (Snowbird, UT), 2000.
[40] T. Ho, D. R. Karger, M. Médard, and R. Koetter, “Network coding from a network flow perspective,” in *Proceedings of the IEEE International Symposium on Information Theory*, (Yokohama, Japan), 2003.
[41] S. Jaggi, P. Chou, and K. Jain, “Low complexity algebraic network codes,” in *Proceedings of the IEEE International Symposium on Information Theory, Yokohama, Japan*, 2003.
[42] P. Sanders, S. Egner, and L. Tolhuizen, “Polynomial time algorithms for network information flow,” in *15th ACM Symposium on Parallel Algorithms and Architectures*, (San Diego, CA), pp. 286–294, 2003.
[43] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, “A random linear network coding approach to multicast,” *IEEE Transactions on Information Theory*, vol. 52, pp. 4413–4430, October 2006.
[44] A. Ramamoorthy, K. Jain, P. A. Chou, and M. Effros, “Separating distributed source coding from network coding,” *IEEE Transactions on Information Theory*, vol. 52, pp. 2785–2795, June 2006.
[45] S. Ulukus and W. Kang, “A single-letter upper bound for the sum rate of multiple access channels with correlated sources,” in *Proceedings of the 39th Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), 2005.