Coded Demixing for Unsourced Random Access

Jamison R. Ebert, *Student Member, IEEE*, Vamsi K. Amalladinne, *Member, IEEE*,
Stefano Rini, *Member, IEEE*, Jean-Francois Chamberland, *Senior Member, IEEE*,
Krishna R. Narayanan, *Fellow, IEEE*

Abstract—Unsourced random access (URA) is a recently proposed multiple access paradigm tailored to the uplink channel of machine-type communication networks. By exploiting a strong connection between URA and compressed sensing, the massive multiple access problem may be cast as a compressed sensing (CS) problem, albeit one in exceedingly large dimensions. To efficiently handle the dimensionality of the problem, coded compressed sensing (CCS) has emerged as a pragmatic signal processing tool that, when applied to URA, offers good performance at low complexity. While CCS is effective at recovering a signal that is sparse with respect to a single basis, it is unable to jointly recover signals that are sparse with respect to separate bases. In this article, the CCS framework is extended to the demixing setting, yielding a novel technique called coded demixing. A generalized framework for coded demixing is presented and a low-complexity recovery algorithm based on approximate message passing (AMP) is developed. Coded demixing is applied to heterogeneous multi-class URA networks and traditional single-class networks. Its performance is analyzed and numerical simulations are presented to highlight the benefits of coded demixing.

Index Terms—Wireless communication; unsourced random access; coded compressed sensing; convex demixing; approximate message passing.

I. INTRODUCTION

The number of unattended devices communicating over wireless networks is expected to increase drastically over the next decades. These machine-type devices, unlike their human counterparts, tend to infrequently transmit very short payloads over the network medium. The handling of such sporadic transmissions is highly inefficient under traditional uplink multiuser coordination processes and thus necessitates the design of novel data-link layer protocols.

An increasingly popular paradigm for such machine-type communication is that of unsourced random access (URA), described by Polyanskiy [1]. In the URA formulation, only a small percentage of the total device population is envisioned to be actively communicating with the base station at a time and their messages are envisioned to be small, i.e., on the order of one hundred bits. Each active user simultaneously transmits its message over regularly scheduled time slots

J. R. Ebert, J.-F. Chamberland, and K. R. Narayanan are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843, USA (emails: {jrebert,chmbrlnd,krn}@tamu.edu).

V. K. Amalladinne is with Qualcomm Research, Qualcomm Technologies, Inc., San Diego, CA, 92121, USA (email: vamsia@qti.qualcomm.com).

S. Rini is with the Department of Electrical and Computer Engineering, National Yang-Ming Chiao-Tong (NYCU), Hsinchu, 300, TW (email: stefano.rini@nycu.edu.tw).

using a common codebook, thus eliminating the overhead and latency associated with multiuser coordination. The receiver is then tasked with producing an unordered list of transmitted messages. We note that, under this framework, the receiver does not know the origin of a message unless a unique identifier is embedded within the message itself.

The predominant performance criterion for URA schemes is the per-user probability of error (PUPE), or the probability that a transmitted message is not recovered by the receiver. The number of active devices is typically assumed to be known at the access point, and the output list size is correspondingly constrained. Since URA devices operate with short payloads, finite block length bounds provide better insight than asymptotic information-theoretic results in this context. Polyanskiy [1] offers finite block length achievability bounds based on random Gaussian coding and maximum likelihood (ML) decoding. While these benchmarks show enviable performance, the ensuing schemes are not computationally tractable in scenarios of practical interest. Over the past few years, significant research effort has gone into developing pragmatic algorithms that approach Polyanskiy's benchmarks with manageable complexity [2]–[12]. We expound on several of these schemes below.

A. Existing URA Schemes

Ordentlich and Polyanskiy authored the first low-complexity scheme for URA in [2]. Therein, they propose dividing the transmission period of n channel uses into slots. Each active user then randomly selects a slot to transmit its message over. A concatenated coding structure is employed wherein the inner linear code recovers the modulo-2 sum of transmitted codewords within a slot, and the outer BCH code recovers the constituent codewords. As long as fewer than T users select a slot, decoding is feasible. In [3], Vem et al. retain the slotted structure proposed in [2], but transmit each message over multiple slots. Furthermore, a preamble section is carved off from every message; this initial portion is transmitted via compressed sensing (CS) and its content determines the repetition pattern for the corresponding payload. The second part is then encoded using an LDPC code and decoded using a message-passing decoder within a slot. Successive interference cancellation (SIC) is leveraged to remove the contribution of decoded messages across slots. This latter scheme provides a sizeable performance improvement over [2]. The authors in [4] introduce sparse interleave division multiple access (IDMA), which employs a signal sparse graph across all transmissions and results in a further performance improvement for systems with a large number of active devices.

In [5], Marshakov et al. show that replacing LDPC codes with polar codes in T-fold irregular repetition slotted ALOHA (IRSA) can result in substantial performance benefits. Their scheme outperforms sparse IDMA when the number of active users is low, i.e., less than 125 or so. Instead of using time-division for separating codewords, Pradhan et al. propose using random spreading in [6]. There, the message is again broken into two parts where the first part is used to select a spreading sequence and the second part is encoded using a polar code. The polar-encoded codeword is spread using the selected spreading sequence. At the receiver, an energy detector is employed to identify active sequences and the codewords are recovered using a list decoder. In [7], the polar coding scheme is improved significantly when the number of active users is large by allowing the set of active users to use different power levels. In [11], Han et al. present a sparse Kronecker product (SKP) based scheme wherein a user's message is encoded as the Kronecker product of a sparse vector and an FEC-encoded vector. The sparse component enables the receiver to use CS techniques to perform multiuser detection and the FEC portion provides significant performance benefits. An iterative algorithm is employed at the receiver. This latter scheme uniformly outperforms [6] and nearly achieves Polyanskiy's finite block length achievability bound when the number of active users is low.

While significant progress was being made with traditional channel coding schemes for URA, the connection between URA and compressed sensing was being explored further. This connection is made by considering a simple bijection on each user's B-bit message. Specifically, each user's B-bit message $\mathbf{u} \in \mathbb{F}_2^B$ may be transformed into a 1-sparse vector $\mathbf{x} \in \mathbb{F}_2^{2^B}$, where the single unity entry in $\mathbf{x}$ is at index $[\mathbf{u}]_2$, which represents an integer under radix-2. After converting its payload to a sparse vector $\mathbf{x}$, each user compresses its sparsified payload $\mathbf{x}$ using a known sensing matrix $\mathbf{A} \in \mathbb{R}^{n \times 2^B}$ into vector $\mathbf{y}$ and then transmits $\mathbf{y}$ over the Gaussian multiple access channel (GMAC). At the receiver, standard CS support recovery is performed to recover the index vector corresponding to the sent payload. This scheme easily extends to the multiple access case: if K users are active during a given time slot, the receiver simply performs K-sparse recovery and is able to recover all sent messages, provided that no two users select the same message to transmit.

A challenging aspect of the compressed sensing view for URA lies in the sheer dimensionality of the problem, often exceeding 2^{100} in practice. At such dimensions, it becomes impractical to utilize standard CS solvers. To circumvent this complexity issue, Amalladinne et al. propose a divide-and-conquer approach wherein each information message is broken up into fragments which are linked together via a tree-based outer code [8]. The coded message components are then converted into a sparse index representation, compressed using the common codebook $\mathbf{A}$, transmitted over the AWGN channel, and individually recovered using standard CS techniques such as non-negative least-squares (NNLS) or LASSO. After each section has been processed, a graph-based decoder connects recovered message fragments back together in a process called stitching [8]. This scheme was enhanced by Fengler, Jung,

and Caire in [9] through the joint encoding of codeword sections at the transmitter and the application of approximate message passing (AMP) as a CS decoder at the receiver. The AMP framework was modified in [10] by allowing for soft information to be dynamically shared between the inner CS decoder and the outer LDPC decoder, resulting in 1 dB gain when $K = 100$. In [13], Ebert et al. show that the weak form of spatial coupling induced by the AMP denoiser in [10] is sufficiently strong to harness the benefits of spatial coupling within the system, thus allowing for the independent encoding of codeword sections. This results in a roughly $1/2$ reduction in computational complexity while maintaining nearly the same level of performance. The scheme presented in [10] represents the state of the art in compressed sensing-based schemes for URA. While this latter scheme does not outperform certain channel coding based approaches such as [6], it offers much lower computational complexity. An even lower-complexity CS-based scheme is described by Calderbank and Thompson in [12]; however, the complexity reduction comes at the expense of PUPE performance and it requires additional channel uses. It should be noted that other URA schemes have been presented, e.g., [14]–[17], yet they are only peripherally connected to our contributions. Indeed, the ideas presented in this work extend some of the notions introduced by Amalladinne et al. [10] and strengthen the connection between URA and the field of sparse recovery.

B. Main Contributions

From a signal processing perspective, one reason for the recent interest in CCS is that it is a computationally efficient tool for recovering signals of exceedingly high dimensions (on the order of 2^{100}) that are sparse in a given domain. Given the connection between URA and compressed sensing, CCS is a natural solution to the massive random access problem.

Independent of CCS, demixing (or convex demixing) has emerged as a tool for recovering constituent signals $\mathbf{s}_1$ and $\mathbf{s}_2$ from their sum $\mathbf{s}_1 + \mathbf{s}_2$, given that $\mathbf{s}_1$ and $\mathbf{s}_2$ are sufficiently sparse in incoherent bases [18]. In this paper, we draw connections between CCS and demixing to allow for the efficient recovery of very high dimensional signals that are sparse in different bases, where the separate bases exhibit low cross-coherence. Due to the extreme dimensions of the problem, the divide and conquer approach of CCS is extended to the demixing problem, thus yielding a novel framework that we call *coded demixing*.

When applied to URA, coded demixing provides the best known performance to multiple heterogeneous classes of users that simultaneously utilize limited channel resources. Furthermore, coded demixing provides significant gains over CCS in the single-class network by reducing the number of users present on a single outer factor graph. We summarize our main contributions as follows.

1) In Section II, a framework for coded demixing is carefully developed that allows for the compression and simultaneous transmission of multiple sparse signals from incoherent bases over fixed channel resources. At the receiver, a low-complexity AMP-based recovery algorithm performs demixing at scale.

2) In Section III, the developed coded demixing framework is applied to a network containing several classes of devices, where a class of devices is a collection of devices with a fixed message length, power budget, and set of coding requirements. Numerical simulations are presented to show that, in the multi-class context, coded demixing outperforms treating interference as noise (TIN) and successive interference cancellation (SIC) schemes with comparable computational complexity.

3) In Section IV, the developed coding demixing framework is applied to a network consisting of a single class of users. There, the set of active users is randomly partitioned into distinct, yet homogeneous, groups during message transmission. Indeed, this reduces the expected number of users present on a single factor graph and therefore improves the performance of the belief propagation (BP) decoder. It is shown that coded demixing with stochastic binning provides nearly state-of-the-art performance when the number of users is large in the single-antenna URA network at reduced complexity.

C. Notation

Unless specifically noted, all variables are real-valued. Matrices are denoted by bold capital letters such as $\mathbf{A}$, and column vectors are denoted by bold lower case letters such as $\mathbf{x}$. The transpose of $\mathbf{x}$ is denoted by $\mathbf{x}^\mathsf{T}$ and the ℓ_p-norm of $\mathbf{x}$ is denoted $\|\mathbf{x}\|_p$. Furthermore, the concatenation of two vectors is written as $\mathbf{uv}$, where $\mathbf{uv} \triangleq [\mathbf{u}^\mathsf{T}\mathbf{v}^\mathsf{T}]^\mathsf{T}$. An $n \times n$ diagonal matrix with entries $d_1, d_2, \ldots d_n$ along its main diagonal is denoted as $\mathrm{diag}\,(d_1, d_2, \ldots d_n)$. In general, sets are denoted by calligraphic letters such as $\mathcal{G}$; two exceptions to this rule are $\mathbb{R}$, which denotes the set of real numbers; and $[N]$, which denotes the set of integers $\{1, 2, \ldots, N\}$. The cardinality of a set is denoted by $|\mathcal{G}|$. Finally, we denote the probability of event $\mathcal{X}$ by $\mathbb{P}\,(\mathcal{X})$.

II. System Model and Coded Demixing

In this section, we present the system under consideration and introduce coded demixing both as a conceptual framework and as a solution to various URA scenarios. From an abstract point of view, coded demixing extends the CCS family of algorithms [8]–[10] to situations where the received signal is the superposition of multiple components that are known to be sparse in distinct domains. The presence of multiple sparse domains introduces new challenges including the eventual need to estimate the number of messages in each domain and the construction of an efficient recovery scheme tailored to this more general framework. In doing so, we leverage several notions from CCS and, whenever appropriate, we draw distinctions about the unique character of coded demixing compared to prior art. Initially, we consider a case where side information is available in terms of number of active devices per sparse domain; this assumption will be relaxed in Sec. IV.

A. System Model

Consider a URA system in which K active devices out of K_{tot} total devices wish to communicate with a base station over a single-input single-output (SISO) uncoordinated GMAC. Each of the K active devices belongs to one of G groups; the manner in which these groups are created will be described later in this article. Devices within a group are homogeneous, but differences in power, data, and/or coding requirements may exist across groups. For convenience, we assign a unique label $j \in [K]$ to every currently active device. Although these labels appear in some equations, they are immaterial to exchanges between devices and the access point. Furthermore, we emphasize that these generic labels are employed solely for the purpose of exposition; they do not reveal anything about the true identities of the active users. The notation $\mathcal{G}_g$ refers to the set of active users that belong to group $g \in [G]$ and $K_g = |\mathcal{G}_g|$ denotes the number of active users in that group. User j encodes its message $\mathbf{w}_j$ into the signal $\mathbf{x}_j \in \mathbb{R}^n$ and then transmits $\mathbf{x}_j$ over n channel uses. The signal received at the access point is given by

$$\mathbf{y} = \sum_{g \in [G]} \sum_{j \in \mathcal{G}_g} d_g \mathbf{x}_j + \mathbf{z}, \tag{1}$$

where d_g represents the amplitude scaling within group g and $\mathbf{z} \in \mathbb{R}^n$ is additive white Gaussian noise with i.i.d. standard normal components. The signal sent by user $j \in \mathcal{G}_g$, denoted $\mathbf{x}_j$, is produced via a codebook $\mathcal{C}_g$ common to all users in $\mathcal{G}_g$. We emphasize that the transmitted signal is only a function of the user's data and group, and *not* its identity. It is also noted that the signal $\mathbf{x}_j \in \mathcal{C}_g \subset \mathbb{R}^n$ satisfies the URA power constraint $\mathbb{E}\left[\|\mathbf{x}_j\|_2^2\right] \leq nP_g$.

The receiver is then tasked with recovering an unordered list of transmitted messages $\hat{\mathcal{W}}(\mathbf{y})$, where $|\hat{\mathcal{W}}(\mathbf{y})| \leq K$. The performance of the scheme is evaluated via the per user probability of error P_e,

$$P_e = \frac{1}{K} \sum_{j \in [K]} \mathbb{P}\left(\mathbf{w}_j \notin \hat{\mathcal{W}}(\mathbf{y})\right). \tag{2}$$

As is customary in existing URA literature, the total number of active users K is given as side information at the receiver. In contrast, the number of active users contained within each group is unknown and therefore must be estimated.

Aside from the presence of multiple groups, this system model is equivalent to the model adopted in other URA articles when the access point features a single receive antenna. This enables the fair comparison of our proposed scheme with prior art, using a *de facto* common task framework.

B. From CCS to Coded Demixing

The theory of coded demixing introduced in this article is closely connected to CCS, and especially the variant of CCS found in [10]. Still, the presence of multiple sparse domains prevents the direct application of CCS to the URA formulation in (1). Below, we discuss the innovations needed to enable coded demixing, focusing primarily on the message encoding and the decoding processes. For the time being, we assume that the values of $\{K_g\}$ are known; we will eventually circle back and relax this requirement. We delay applications and performance characterization to subsequent sections (see Sec. III and Sec. IV).

1) Encoding Process: Without loss of generality, we present the encoding process for user j in group g. We note that the structure of the encoding process is the same across groups, but specific parameters may vary from group to group. The message encoding process utilizes the concatenated coding structure established in [8]. This features a SPARC-like CS inner code similar to the approach proposed in [9], and an LDPC outer code akin to those in found [10].

The first step in the encoding process is to add redundancy to user j's w_g-bit message $\mathbf{w}_j$ via a rate R_g LDPC code. Explicitly, the message $\mathbf{w}_j \in \{0,1\}^{w_g}$ is broken into $\varkappa_g$ segments each of length v_g such that

$$\mathbf{w}_j = \mathbf{w}_j(1)\mathbf{w}_j(2)\cdots\mathbf{w}_j(\varkappa_g).$$

Each section $\mathbf{w}_j(i)$ may be viewed as an element of GF(2^{v_g}) and thus the entire message $\mathbf{w}_j$ may be encoded using a non-binary LDPC code, yielding a codeword $\mathbf{v}_j$. The codeword $\mathbf{v}_j$ can likewise be broken into L_g segments, each of length v_g:

$$\mathbf{v}_j = \mathbf{v}_j(1)\mathbf{v}_j(2)\cdots\mathbf{v}_j(L_g). \tag{3}$$

At this point, each section of the codeword is converted into a 1-sparse message vector via the bijection $\mathbf{m}_j(i) = f_g(\mathbf{v}_j(i))$, where $f_g(\mathbf{v}(i)) : \mathbb{R}^{v_g} \to \mathbb{R}^{2^{v_g}}$. The single unitary entry of $\mathbf{m}_j(i)$ is at location $[\mathbf{v}_j(i)]_2$, which is interpreted as an integer expressed under radix-2. Once every section is encoded, the resulting L_g-sparse vector $\mathbf{m}_j = \mathbf{m}_j(1)\cdots\mathbf{m}_j(L_g)$ becomes similar to a SPARC codeword; although it features fewer sections, each is of much greater length [9]. Attached to group g is a Gaussian sensing matrix $\mathbf{A}_g \in \mathbb{R}^{n \times L_g 2^{v_g}}$ with independent entries distributed as $\mathcal{N}(0, 1/n)$, so that the expected ℓ_2-norm of each column is one. Message $\mathbf{m}_j$ is converted into a signal $\mathbf{x}_j = \mathbf{A}_g\mathbf{m}_j$ and transmitted over the channel. Fig. 1 graphically summarizes this part of the encoding process. From the perspective of a device, these steps are equivalent to the encoding process within CCS.

The distinction between CCS and coded demixing becomes more apparent when the distributed encoding of all active devices is considered together. We let vector $\mathbf{s}_g$, $g \in [G]$, be the sum of all the signals sent by active devices within group g. That is,

$$\mathbf{s}_g = \sum_{j \in \mathcal{G}_g} \mathbf{m}_j.$$

We allow devices across groups to use different power levels, which leads to the scaling of signal amplitudes, with $\Phi_g = d_g\mathbf{A}_g$. Under these aggregates, the signal received at the base station can be expressed as

$$\mathbf{y} = \Phi_1\mathbf{s}_1 + \Phi_2\mathbf{s}_2 + \cdots + \Phi_G\mathbf{s}_G + \mathbf{z}, \tag{4}$$

where $\mathbf{s}_g$ is L_gK_g-sparse. The random nature of each $\mathbf{A}_g$ ensures that the set of sensing matrices $\{\mathbf{A}_g : g \in [G]\}$ exhibits low cross-coherence with high probability; this is key for the successful application of demixing techniques [19]. With the conditions of sparsity and low coherence met, (4) assumes a canonical form for demixing problems.

Remark 1. *The sparse domains in coded demixing are defined by the column spans of $\{\mathbf{A}_g : g \in [G]\}$. Each group typically uses a separate LDPC outer code, and the number*
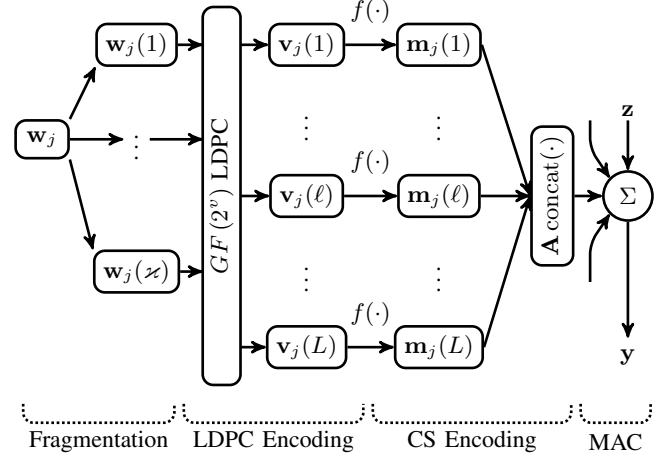

Fig. 1. This figure illustrates the encoding process for user j in group g. For simplicity of notation, the group subscript g has been dropped in this figure. User j's message $\mathbf{w}_j$ is first broken into $\varkappa_g$ fragments; then, these fragments are encoded using an LDPC code over $GF(2^{v_g})$. Each of the LDPC-encoded fragments are subsequently converted into 1-sparse vectors through the bijection $f_g(\cdot)$ and the resulting sparse vectors are compressed through multiplication with $\mathbf{A}_g$. This results in signal $\mathbf{x}_j$, which is combined with signals from other users (possibly from other groups) and AWGN to form the received signal $\mathbf{y}$.

of sections and their lengths may vary from group to group. The introduction of multiple groups creates the need to track $\{K_g : g \in [G]\}$; these quantities may not be available as side information. This, in turn, may require the modification of the transmitted signals, with possibly the addition of group specific pilots, to facilitate this task.

Remark 2. *The presence of multiple sparse domains invites the rethinking of the decoder because the naive application of parallel CCS decoders (one for each sparse domain) to the received vector $\mathbf{y}$ will not work. Intuition can be drawn from the many algorithms designed for convex demixing [18]–[23], yet a suitable demixing solver must have manageable complexity for the problem dimensions relevant to modern communication scenarios and should take advantage of the signal structure described above. These requirements preclude the use of classical regularized convex optimization approaches; consequently, we resort to a first-order iterative solution.*

2) Decoding Process: The decoder is tasked with recovering the set $\{\mathbf{s}_g : g \in [G]\}$ and subsequently, disambiguating the messages contained within each $\mathbf{s}_g$. This two-step process is inspired by CCS decoding as presented in [10], albeit with key modifications to support multiple groups of users. A diagram of the iterative decoding procedure, with its main components, appears in Fig. 2.

Equation (4) admits an alternate representation that gives insight into how to develop an iterative procedure for support recovery. If the matrices $\{\Phi_g : g \in [G]\}$ are concatenated to form $\Phi = [\Phi_1, \Phi_2, \ldots, \Phi_G] \in \mathbb{R}^{n \times \gamma}$, where $\gamma = \sum_g L_g 2^{v_g}$, and the vectors $\{\mathbf{s}_g : g \in [G]\}$ are similarly stacked into
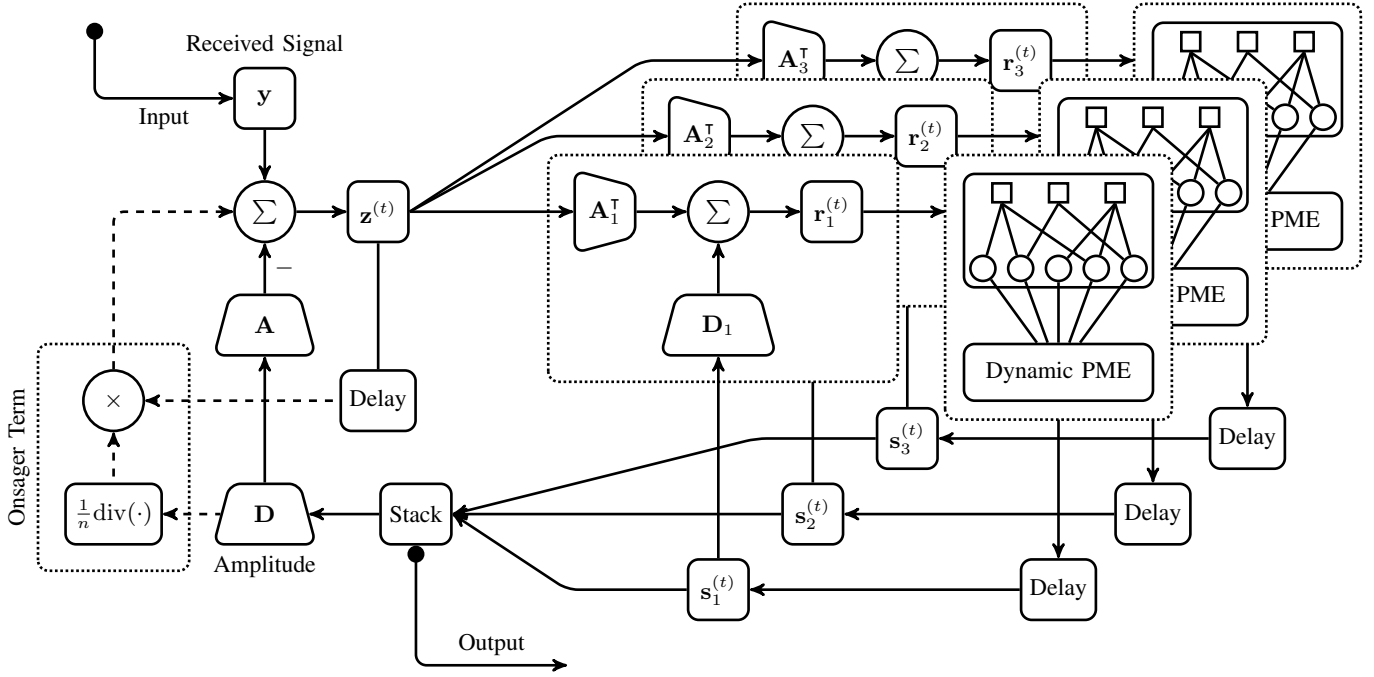
Fig. 2. This notional diagram depicts the operation of the coded demixing recovery algorithm. The input comes in the form of observation $\mathbf{y}$ at the top left. The contributions of the state estimate (minus the Onsager term) are subtracted from $\mathbf{y}$, leading to the residual $\mathbf{z}$. This residual is then processed through demixing and turned into an effective observation for each group. Denoising is performed as a means to update the state estimates. The computation of the Onsager contribution, which is intrinsic to AMP, is highlighted using dashed lines. This iterative process continues until convergence is reached. At that point, the state estimates $\{\mathbf{s}_g : g \in [G]\}$ are taken as the output of the algorithm.

$\mathbf{s} = \mathbf{s}_1 \mathbf{s}_2 \ldots \mathbf{s}_G \in \mathbb{R}^\gamma$, then (4) can be rewritten as

$$\mathbf{y} = \mathbf{\Phi}\mathbf{s} + \mathbf{z} = \mathbf{A}\mathbf{D}\begin{bmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_G \end{bmatrix} + \mathbf{z}, \qquad (5)$$

where $\mathbf{D} = \operatorname{diag}(d_1, d_2, \ldots, d_G)$ and $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_G]$. We note that (5) is in standard CS form, and possibly amenable to decoding via approximate message passing (AMP). Of course, the rewriting of this equation does not simplify the intricate sparse structure of vectors $\{\mathbf{s}_g : g \in [G]\}$. However, it offers a conceptual bridge that has been leveraged in the past to relate demixing to CS. At this point, we turn to the AMP composite iterative structure and to the design of suitable denoiser for coded demixing. The specific AMP algorithm we have in mind takes the form

$$\mathbf{z}^{(t)} = \mathbf{y} - \mathbf{A}\mathbf{D}\begin{bmatrix} \mathbf{s}_1^{(t)} \\ \vdots \\ \mathbf{s}_G^{(t)} \end{bmatrix} + \frac{\mathbf{z}^{(t-1)}}{n}\operatorname{div}\mathbf{D}\boldsymbol{\eta}^{(t-1)}\left(\mathbf{r}^{(t-1)}\right) \quad (6)$$

$$\begin{bmatrix} \mathbf{s}_1^{(t+1)} \\ \vdots \\ \mathbf{s}_G^{(t+1)} \end{bmatrix} = \boldsymbol{\eta}^{(t)}\left(\mathbf{r}^{(t)}\right) = \begin{bmatrix} \boldsymbol{\eta}_1^{(t)}\left(\mathbf{r}_1^{(t)}\right) \\ \vdots \\ \boldsymbol{\eta}_G^{(t)}\left(\mathbf{r}_G^{(t)}\right) \end{bmatrix}, \quad (7)$$

where the superscript t denotes the iteration number, $\mathbf{s}^{(0)} = \mathbf{0}$, and $\mathbf{z}^{(0)} = \mathbf{y}$. Although more involved, this is the natural multi-group extension of the AMP algorithm applied to CCS

in [10]. In the present context, we define the effective observation by

$$\mathbf{r}^{(t)} = \mathbf{A}^\mathsf{T}\mathbf{z}^{(t)} + \mathbf{D}\begin{bmatrix} \mathbf{s}_1^{(t)} \\ \vdots \\ \mathbf{s}_G^{(t)} \end{bmatrix}. \qquad (8)$$

Equation (6) can be interpreted as the residual enhanced with an Onsager correction term [24], [25]. The AMP state is updated in (7) using denoiser $\eta_g^{(t)}(\cdot)$, which seeks to take advantage of the block structure of the problem. Essentially, the denoiser should promote structured sparsity while also accounting for the presence of the outer code and, as such, it must be designed carefully. The two iterative steps are linked through the computation of the effective observation in (8).

A crucial fact about AMP is that under suitable conditions, such as $\mathbf{A}$ having i.i.d. Gaussian entries and the denoiser being Lipschitz-continuous [24], [26], $\mathbf{r}^{(t)}$ is asymptotically distributed as

$$\mathbf{D}\begin{bmatrix} \mathbf{s}_1^{(t)} \\ \vdots \\ \mathbf{s}_G^{(t)} \end{bmatrix} + \tau_t \boldsymbol{\zeta}_t, \qquad (9)$$

where $\boldsymbol{\zeta}_t \sim \mathcal{N}(0, \mathbf{I})$ and τ_t is a deterministic scalar quantity. It is also noted that, though τ_t may be computed through joint state evolution, it is often approximated by $\tau_t^2 \approx \|\mathbf{z}^{(t)}\|_2^2/n$. Naturally, it may be tempting to apply a minimum mean square error (MMSE) estimator to each section ℓ

$$\mathbb{E}[\mathbf{s}_g(\ell)|d_g\mathbf{s}_g + \tau_t\boldsymbol{\zeta}_t = \mathbf{r}_g]$$
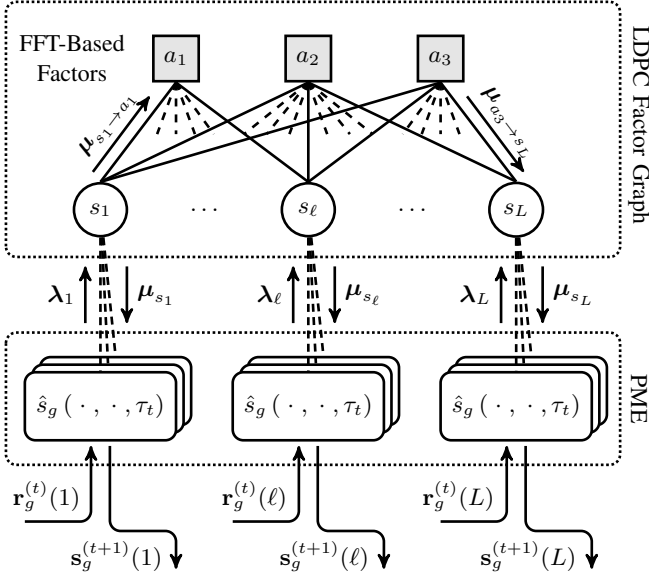
Fig. 3. This figure illustrates the denoiser employed for group g within the AMP iterate for coded demixing. The PME of (10) is applied to the effective observation $\mathbf{r}$ using uninformative priors to obtain local observations $\{\boldsymbol{\lambda}_\ell : \ell \in [L]\}$, which initialize the outer LDPC factor graph. After running belief propagation, messages $\boldsymbol{\mu}_{s_\ell}$ are sent from the variable nodes in the factor graph to the PME for each section; these messages are used to generate informative priors. Equipped with more accurate priors, the PME is run on the effective observation $\mathbf{r}$ once more, yielding $\boldsymbol{\eta}(\mathbf{r})$.

as the denoiser. However, because each $\mathbf{s}_g(\ell)$ is K_g-sparse, the MMSE estimator would have to consider $\binom{2^{v_g}}{K_g}$ different possibilities for every section. Clearly, this becomes combinatorially intractable for section sizes of practical interest. We therefore adopt the strategy proposed by Fengler et al. [9], and we approximate the MMSE denoiser using the posterior mean estimator (PME),

$$\hat{s}_g(q, r, \tau_t) =$$
$$\frac{q \exp\left(-\frac{(r - d_g)^2}{2\tau_t^2}\right)}{q \exp\left(-\frac{(r - d_g)^2}{2\tau_t^2}\right) + (1 - q)\exp\left(-\frac{r^2}{2\tau_t^2}\right)}, \quad (10)$$

where r is an entry in the effective observation $\mathbf{r}$, q denotes the prior probability of an entry of $\mathbf{s}_g$ being equal to one, and τ_t is the scaling factor discussed above. In [9], the authors focus on the single-group scenario and they employ an uninformative prior, which would be equivalent to $q_g = 1 - (1 - 2^{-v_g})^{K_g}$ for group g.

In [10], Amalladinne et al. show that, when there is only one group, the structure of the outer LDPC code can be exploited to improve the PME denoiser. Specifically, one round of BP may be run on a suitably designed outer factor graph to generate more accurate priors. This helps AMP converge to a parity-consistent solution, and it has been shown to improve the performance of CCS by 1 dB when $K = 100$. We wish to extend these developments to coded demixing by generating priors for each group through message-passing on each group's outer LDPC factor graph.

We only briefly summarize the message passing rules associated with the LDPC outer codes because the details of message passing on factor graphs are well established (e.g.,

[27]). We also omit the group subscript g for this part to lighten notation. While every group has its own LDPC outer code, belief propagation acts in a similar fashion within each group, with no explicit interactions between groups. The fact that the LDPC factor graphs are disjoint across groups should prevent any confusion about group indexing. Variable nodes are initialized with local estimates $\boldsymbol{\lambda}_\ell(k) = \hat{s}_g(q_g, \mathbf{r}_g(\ell, k), \tau_t)$. One round of belief propagation is then performed with check to variable messages given by

$$\boldsymbol{\mu}_{a \to s_\ell}(k) = \sum_{\mathbf{k}_a : k_\ell = k} \Psi_a(\mathbf{k}_a) \prod_{s_j \in N(a) \backslash s} \boldsymbol{\mu}_{s_j \to a}(k_j), \quad (11)$$

where Ψ_a is an indicator function that enforces parity consistency, k is an index within a section, and $\mathbf{k}_a$ is a collection of section indices for the neighbors $N(a)$ of check node a. Messages from variable nodes to check nodes are given by

$$\boldsymbol{\mu}_{s_\ell \to a}(k) \propto \boldsymbol{\lambda}_\ell(k) \prod_{a_p \in N(s_\ell) \backslash a} \boldsymbol{\mu}_{a_p \to s_\ell}(k), \quad (12)$$

where the '$\propto$' indicates a normalization of the message. After running one round of BP, the belief associated with section ℓ may be obtained as follows

$$\boldsymbol{\mu}_{s_\ell}(k) = \prod_{a \in N(s_\ell)} \boldsymbol{\mu}_{a \to s_\ell}(k). \quad (13)$$

Once normalized, (13) may be used as a more informative prior q in the PME of (10). Explicitly,

$$\mathbf{q}_g(\ell, k) = 1 - \left(1 - \frac{\boldsymbol{\mu}_{s_\ell}(k)}{\|\boldsymbol{\mu}_{s_\ell}\|_1}\right)^{K_g}. \quad (14)$$

Note that log-domain Fourier transform decoding [28], [29] is employed for efficient computation of BP. Accordingly FFT-based factors are employed in the bipartite graph representing the LDPC code. An illustration of message passing on a generic bipartite graph is provided in Fig. 3.

Combining the PME of (10) with the priors computed dynamically using BP results in the smooth denoiser found in [10, Definition 6]. Accounting for the various sparsity domains in coded demixing and their distinct outer LDPC codes, we get group-specific non-separable denoisers of the form

$$\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g) = \hat{\mathbf{s}}_g(1, \mathbf{r}_g, \tau_t) \cdots \hat{\mathbf{s}}_g(L_g, \mathbf{r}_g, \tau_t), \quad (15)$$

where individual components are based on (10) with

$$\hat{\mathbf{s}}_g(\ell, \mathbf{r}_g, \tau_t) = (\hat{s}_g(\mathbf{q}_g(\ell, k), \mathbf{r}_g(\ell, k), \tau_t) : k \in [2^{v_g}])$$

and priors are obtained from (14) with one round of message passing on the LDPC factor graph. This same approach is used for all groups $\{g \in [G]\}$, though each group runs BP on its own factor graph and power levels may vary across groups. This explains why the subscript g must be included in the definition of $\boldsymbol{\eta}_g^{(t)}(\mathbf{r})$. It has been shown that this denoiser is Lipschitz continuous; thus, it is well-suited for AMP. In computing the AMP residual in (6), the Onsager correction term requires the divergence of the denoiser, which is provided below,

$$\mathrm{div}\mathbf{D}\boldsymbol{\eta}^{(t)} = \frac{1}{\tau_t^2}\left(\|\mathbf{D}^2\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_1 - \|\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_2^2\right). \quad (16)$$

Note that the form of the Onsager term remains remarkably simple despite the presence of several groups. This stems from the additive structure of both $\|\cdot\|_1$ and $\|\cdot\|_2^2$, which yields

$$\sum_{g\in[G]} \operatorname{div}\left(d_g \boldsymbol{\eta}_g^{(t)}\right) = \operatorname{div}\mathbf{D}\boldsymbol{\eta}^{(t)}.$$

In this sense, the compact notation of (16) is due, not only to the demixing formulation, but also to the specifics of our denoiser. It is noted that (16) holds only when the number of BP iterations per AMP step is strictly less than the length of the shortest cycle on the outer graph. The derivation of (16) is provided in Appendix A.

After running several iterations of the AMP algorithm via (6) and (7), the decoder produces estimates for $\{\mathbf{s}_g : g \in [G]\}$. To get the transmitted messages, the receiver must then extract valid codewords by finding parity-consistent indices across the L_g recovered sections of $\mathbf{s}_g$. The indices associated with the $K_g + \delta$ largest values in the first section of $\mathbf{s}_g$ are identified and the following actions are repeated for each of the selected indices. The root section of the LDPC factor graph is initialized with the standard basis vector $\mathbf{e}_i$ where i is the selected index and the other $L_g - 1$ sections are initialized with the output of AMP. Several BP iterations on the graph of the outer code are then performed, which yields candidate sections. If a parity-consistent codeword is obtained, that codeword is added to the list of recovered codewords along with its associated likelihood. Once this process has been performed for all G bins, the recovered codewords from all bins are collected and ordered in terms of decreasing likelihood. Finally, the top K codewords are retained as $\hat{\mathcal{W}}(\mathbf{y})$.

Remark 3. *While the theory of AMP and state evolution applies to random Gaussian matrices, it is common in practice to generate each $\mathbf{A}_g$ by randomly sampling the rows of a $2^{v_g} \times 2^{v_g}$ Hadamard matrix (excluding the row of all ones), instead of generating the sensing matrices $\{\mathbf{A}_g : g \in [G]\}$ with random Gaussian entries. This enables the use of Fast Walsh-Hadamard Transform techniques within the AMP iterations (6) and (7), which decreases both the computational complexity and the required memory. The fact that AMP works well for certain random matrices with non-Gaussian entries was shown empirically in [30].*

Having established a framework for coded demixing, we proceed with the description of two application scenarios: accommodating multiple classes of users within a network, and improving the PUPE performance of a single-class network via stochastic binning of users. We begin with an examination of the multi-class URA network.

III. MULTI-CLASS NETWORKS VIA CODED DEMIXING

In its original definition, the URA model is composed of K_{tot} *homogeneous* users. That is, all K_{tot} users have the exact same power budget, message length, and code requirements. An interesting and highly applicable twist to the original URA setting is that of a *heterogeneous* URA model (HetURA), as described in [31], where devices are partitioned into classes with varying power, data, and code requirements.

These classes may either be completely independent or connected together in some fashion. As an example of independent classes, consider an industrial setup where a central base station receives sensor data from various sections of a warehouse, and each section is located at a different distance from the base station. To offset the free space path loss incurred by the additional distance, sensors in further away sections employ a lower code rate when communicating with the base station. In this example, each class consists of the group of sensors that employ a specific code rate.

As an example of connected classes, consider the task of sending a long URA message over the channel, say on the order of 256 bits. Unfortunately, off-the-shelf CS is not able to handle the dimensionality of this problem. One option to circumvent this issue is to apply the technique presented in [13]; another option is to employ multiple, concatenated classes. Specifically, let the transmitter divide its message into sections and link each section together with an additional outer code. Then, let the transmitter send the first part of its message as part of class one, the second as part of class two, and so on until the final portion of its message is sent via the class ϱ. After recovering all the received messages, the decoder may use side information about classes $1, \dots, \varrho$ and the outermost linking parities to properly stitch the very large message together.

For ease of exposition, we study only the case of independent classes in our simulations, but the results presented in this article may be extended to the connected class case with minimal modifications.

A. Prior Work

After introducing the HetURA model in [31], Hao et al. propose a scheme for the case when two classes of users exist with different power levels. Therein, the high-energy users employ a two-layer superposition modulation and the low-energy users employ the same base constellation as the high-energy users. The top level of the superposition constellation is decoded, followed by the lower level with the help of SIC. Finally, a CCS-style tree code is employed as an outer code to disambiguate messages. A version of this problem is also studied by Huang et al. in [32], wherein each user transmits its message L times in random slots. At the receiver, power-domain NOMA techniques are leveraged to handle collisions within a single slot and inter-slot SIC is applied to improve performance. In [33], this problem is considered from a coded demixing perspective. It is upon the work of Amalladinne et al. [33] that we wish to expound in this article.

B. Proposed Approach

The coded demixing model presented in Section II-A may be applied to the HetURA channel in the following manner. Let the K_{tot} total users be clustered into G groups, where each group consists of devices from the same class. We note that a class of users can be broken up into multiple groups, but a single group cannot have devices from multiple classes. Thus, each group $g \in [G]$ has an associated amplitude level d_g, message length w_g, coding rate R_g, codeword length v_g,
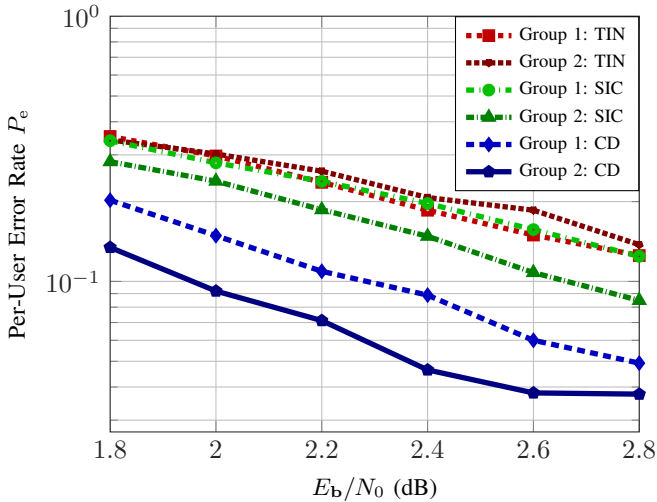
Fig. 4. This plot compares the PUPE of various multi-class URA schemes. Two classes are considered, each with 25 active devices. In group 1, $w_1 = 128$ bits and $R_1 = 1/2$. In group 2, $w_2 = 96$ bits and $R_2 = 3/8$. Coded demixing outperforms other known communication schemes.

and sensing matrix $\mathbf{A}_g$. Furthermore, to conform to a widely adopted convention, it is assumed that the number of active users per group is known to the receiver.

C. Two-Class System Model

For illustrative purposes, we consider a scheme with two groups, each containing 25 active users. Group 1 features a message length of 128 bits and a coding rate of $1/2$, and group 2 features a message length of 96 bits with a coding rate of $3/8$. The total number of sections (parity + information) within each graph is 16, and the length of each section is 2^{16}. The power budget P is obtained via the relationship $\frac{E_{\mathrm{b}}}{N_0} = \frac{nP}{2w_g}$. The transmitter outer/inner encoding operations proceed exactly as outlined in Section II-A. At the receiver, AMP decoding and message disambiguation proceed as outlined above, with the exception that codewords recovered from group 1 are kept separate from codewords recovered from group 2. The PUPE is computed as in (2) for each of the two groups individually.

D. Two-Class Simulation Results

We compare the performance of multi-class CCS via coded demixing to other existing communication schemes, including treating interference as noise (TIN) and single-class decoders with SIC. In TIN, the decoder for group 1 operates on received vector $\mathbf{y}$, and views the contribution of group 2 as additional noise. Similarly, the decoder for group 2 operates on $\mathbf{y}$ and treats interference from group 1 as noise. In SIC, the decoder first produces an estimate $\hat{\mathbf{s}}_1$ of the signal sent by group 1 based on observation $\mathbf{y}$. It then subtracts the estimated contribution of group 1 from $\mathbf{y}$, yielding the residual $\tilde{\mathbf{y}} = \mathbf{y} - \Phi_1\hat{\mathbf{s}}_1$. The decoder then produces an estimate $\hat{\mathbf{s}}_2$ of the signal transmitted by group 2 from residual $\tilde{\mathbf{y}}$.

Fig. 4 shows that coded demixing outperforms both SIC and TIN in both groups by up to 0.6 dB. In addition to providing a striking PUPE improvement, the complexity of the coded demixing solution is very comparable to the complexity of SIC or TIN. Altogether, a judicious use of coded demixing forms a promising paradigm for multi-class URA, and it currently offers state-of-the-art performance.

IV. ENHANCED SINGLE-CLASS CCS VIA CODED DEMIXING AND STOCHASTIC BINNING

In Section III, the coded demixing framework was considered as a means to enable multiple classes of devices to simultaneously use the same network resources. In this section, we demonstrate how the coded demixing framework may be used to enhance the PUPE performance of a network with a single class of users by randomly partitioning active devices into groups.

Within CCS, K codewords are present on the outer LDPC factor graph at once during message passing [10]. This results in a form of mixing that makes it difficult for belief propagation to distinguish between codewords and degrades performance. Though the belief propagation algorithm may be modified slightly to facilitate multi-user decoding, reducing the number of simultaneous codewords on the factor graph seems to constitute a better way to improve the performance of the outer decoder. As a means to accomplish this goal, the collection of active devices may stochastically partition themselves into groups, and subsequently adopt the coded demixing framework introduced above (see also [34]). We elaborate on this option below.

A. Stochastic Binning

We propose a scheme in which there are G homogenous groups with separate sensing matrices and outer factor graphs. We stress that these groups share the same transmit power, message length, number of sections, and coding rate; that is, $d_g = d_h, w_g = w_h, v_g = v_h, L_g = L_h, R_g = R_h$ for all $g, h \in [G]$. To emphasize the fact that the groups are homogeneous, we employ the term *bin* instead of *group*.

After the K active users have generated their messages, every user selects a bin based on the first $w_0 = \log_2(G)$ bits of its information message. We restrict our treatment of this approach to values of G that are powers of two; other values of G are possible, but would require slight adjustments to the proposed algorithm. The distribution of users across bins $\{K_g : g \in [G]\}$ assumes a multinomial distribution with K trials and probabilities $p_g = 1/G$ for all $g \in [G]$. Fig. 5 depicts the bin selection and encoding processes.

We stress that the process of assigning active users to bins must be performed stochastically since there is no a-priori knowledge on what users are currently active.

Although within the URA common task framework the total number of active users K is known to the receiver, the number of users per bin $\{K_g : g \in [G]\}$ is unknown. Rather, this information must be inferred at the destination. To enable accurate bin occupancy estimation, every active user simultaneously transmits a length-G standard basis vector $\mathbf{e}_g$, where the index of the single unitary entry in $\mathbf{e}_g$ corresponds to the bin that the user has selected. Within the URA framework, the expected total power used per device is upper-bounded
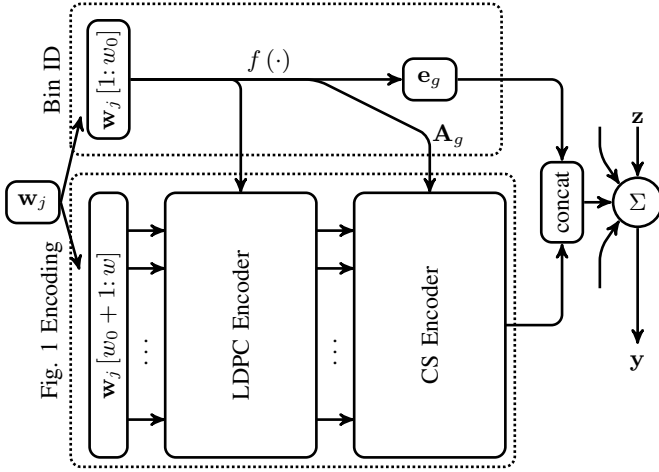
Fig. 5. This figure illustrates the encoding process for coded demixing with stochastic binning. Each user's message is separated into two parts: the first w_0 bits are used to select a bin and the remaining $(w - w_0)$ bits are encoded as described in Fig. 1. We note that the choice of bin determines the outer LDPC code employed as well as the sensing matrix $\mathbf{A}_g$ used for CS encoding. Because the receiver does not know how many users have selected each bin a priori, a bin identification sequence must be sent so the distribution of active users across bins can be estimated.

by nP. Consequently, the power devoted to transmitting the bin identification sequence must be subtracted from the power allocated to sending the information message. Numerical simulations indicate that allocating 0.2% of the total power to the bin occupancy estimation task allows for good estimates $\{\hat{K}_g : g \in [G]\}$ without significantly degrading the PUPE performance. At the receiver, a standard linear minimum mean square error (LMMSE) estimator is used to jointly estimate the number of users per bin $\{\hat{K}_g : g \in [G]\}$. These estimates are used throughout the AMP decoding algorithm, as presented in Section II-A.

After transmitting their bin identification sequences, the set of active users transmit their encoded messages over the GMAC. After estimating the number of users per bin, the AMP-based coded demixing algorithm of Section. II is employed and $\hat{K}_g + \delta$ potential messages are retained within bin g. At this stage, we point out that w_0 bits of information are conveyed per message via the choice of a bin; simply stated, every user within bin g must have the same first w_0 bits, which must equal the binary representation of g. The bin identity can therefore be leveraged as an additional error-detecting code and the list of candidate messages within each bin may be pruned by removing all messages whose first w_0 bits do not correspond to g. After removing parity-inconsistent messages within each bin, the G lists of candidate messages, one for each bin, are merged into a single list of recovered messages. The aggregate list is sorted in terms of decreasing likelihood, and the top K messages from this sorted list are taken as the K reported messages by the base station. To further improve performance, one outer loop of SIC may be employed, wherein approximately 70% of the recovered messages are retained during the first round of SIC. After decoding these messages, their contributions are subtracted from the received vector $\mathbf{y}$, and the decoding algorithm is run one more time to recover the remaining 30% of the messages.

Scheme	G	K	L	w	n
CCS-AMP	1	25-175	16	128	38400
CCS-AMP	1	200-275	18	128	38400
Coded Demixing	2	25-275	16	128	38400
Coded Demixing	8	25-275	16	128	38400

Remark 4. *As will be shown shortly, in some regimes, increasing the number of bins tends to improve performance. However, one must tread carefully because AMP is known to only converge within certain regions in terms of undersampling and sparsity ratios [30]. This depends on the problem formulation and the denoiser, yet traditional settings offer a cautionary tale as to what can be accomplished. For our scheme, the sparsity is equal to $\sum_{g \in [G]} K_g L/n = KL/n$ and the undersampling ratio is given by $n/GL2^v$. As the number bins G is increased, the sparsity level remains constant but the undersampling ratio goes to zero. This can affect the ability of AMP to handle noise and, at worst, can cause the operating point of the system to fall out of the convergence region.*

Furthermore, low cross-coherence among $\{\mathbf{A}_g : g \in [G]\}$ is a critical underlying assumption of the theory of demixing and, as G increases, the cross-coherence among $\{\mathbf{A}_g : g \in [G]\}$ is bound to increase. Thus, there exists an upper bound on G for which decoding succeeds with high probability. An exact characterization of such a bound remains an open research question. Still, these notions provide guiding principles on the selection of G.

Remark 5. *When $G = 1$, every active user belongs to the same bin. Thus, under the URA common task framework, K is known and there is no need to perform occupancy estimation. In this situation, active users would refrain from sending a bin identification sequence and would instead employ 100% of their power for transmitting their payloads. Under these modifications, the $G = 1$ coded demixing scheme reduces to the CCS-AMP scheme of Amalladinne et al. in [10].*

B. Single-Class Simulation Results

We now characterize the performance of a single-class URA system that employs coded demixing and stochastic binning. We consider competing implementations with the parameters summarized in Table I. The power budget P is obtained via the relationship $\frac{E_\mathrm{b}}{N_0} = \frac{nP}{2w}$ and, as discussed in Remark 3, the sensing matrices $\{\mathbf{A}_g : g \in [G]\}$ are generated by randomly sampling the rows of a $2^v \times 2^v$ Hadamard matrix, excluding the row of all ones.

Fig. 6 compares the required E_b/N_0 to obtain a PUPE of 0.05 as a function of K for $G = 1, 2, 8$. The curve for $G = 1$ corresponds to the original CCS-AMP scheme presented in [10]. Fig. 6 empirically reveals that the $G = 2$ curve uniformly outperforms CCS-AMP for all K, and that the $G = 8$ curve outperforms both CCS-AMP and $G = 2$ for moderate K (i.e., $75 \leq K \leq 200$). However, when $G = 8$ and K is large, the performance of the $G = 8$ curve decays sharply.

There are some nuances to Fig. 6 that need to be considered when interpreting the results. Recall that, as part of the decoding process, BP is run on the outer factor graph to
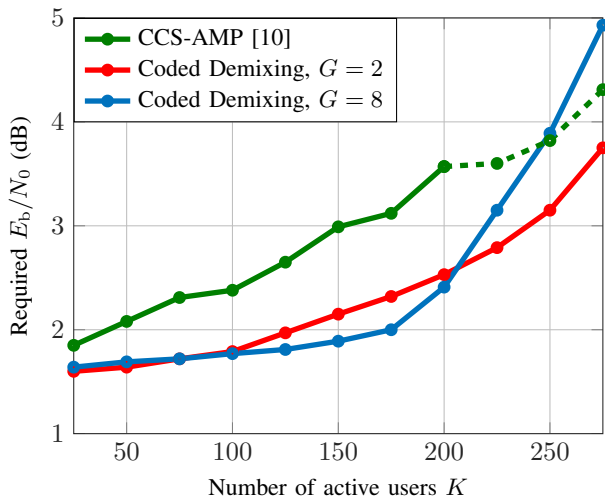
Fig. 6. Required E_b/N_0 for PUPE = 0.05 vs number of active users K for $G = 1, 2, 8$. Under coded demixing, adding bins improves the performance of the URA system, up to a point (see Remark 4).

Fig. 7. This plot shows required E_b/N_0 for PUPE = 0.05 vs number of active users K. The performance of coded demixing with $G = 2$ is compared to other URA schemes.

disentangle the K users present on that graph. Empirical results evidence an upper limit on K for which the decoding process performs arbitrarily well as $E_b/N_0 \to \infty$; in our case, this upper limit is around 200. Thus, when generating the CCS-AMP curve, the outer graph has to be extended to 18 sections for $K \geq 200$ in order to obtain a PUPE of 0.05. Furthermore, the extended graph is characterized by a crossover SNR when compared to the original graph. For the region of K and E_b/N_0 considered in Fig. 6, the extended graph did not improve the performance of the $G = 2$ or $G = 8$ bin curves and, therefore, it was not used for $K \geq 200$. The CCS-AMP curve is displayed, partially, as a dotted line on this figure to highlight the region where the extended outer graph is being used. Altogether, the parameter space is vast and the performance of coded demixing appears to be sensitive to the choice of parameters. The optimal choice of parameters, including the optimal design of the outer graph, remains an open research problem.

Fig. 7 compares the required E_b/N_0 to obtain a PUPE of 0.05 as a function of K for coded demixing to Polyanskiy's random coding benchmark [1], CCS-AMP [10], Sparse Kronecker-Product coding [11], Sparse IDMA [4], and polar coding with spreading [6]. When $G = 2$ and $K \geq 225$, coded demixing outperforms all referenced schemes in terms of required E_b/N_0.

C. Performance with K Unknown at Receiver

Throughout this article, we have assumed that the total number of active devices K is known at the receiver. This assumption is in line with existing URA literature and allows for a fair comparison with existing schemes. However, in many scenarios of practical interest, this assumption is not properly justified; therefore, we briefly consider the case when K is unknown a priori and must be estimated. Interestingly, under minor modifications, coded demixing still performs well in this alternate scenario. This should not be too surprising in view of recently published results on URA without side
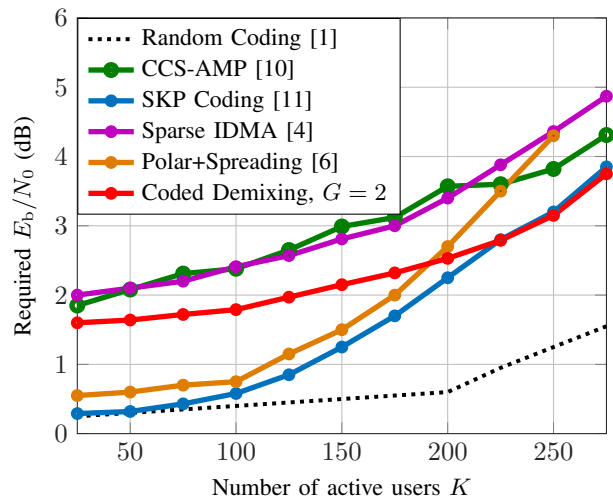
information [35], a study which concludes that the lack of knowledge in the number of active devices should entail only a small penalty in power efficiency.

When K is not known at the receiver, the total number of active devices can be inferred from the bin identification sequences. Because our proposed LMMSE estimator relies on K as known parameter, a different estimator should be employed. To obtain $\{\hat{K}_g : g \in [G]\}$, the receiver may simply divide the received bin identification sequence by the users' transmit power and then round the result to the nearest integer. The estimated total number of active users $\hat{K}$ is then given by $\hat{K} = \sum_{g \in [G]} \hat{K}_g$. The rest of the coded demixing algorithm proceeds exactly as has been described in this article, albeit using $\hat{K}$ instead of K.

When K is unknown, we consider both the probability of missed detection $\Pr(\text{MD})$, defined as the probability of a user's message not appearing in $\hat{\mathcal{W}}(\mathbf{y})$, and the probability of false alarm $\Pr(\text{FA})$, defined as the probability of a codeword appearing in $\hat{\mathcal{W}}(\mathbf{y})$ that does not correspond to a transmitted message. In contrast, when K is known and used as a constraint on the size of the output list, $\Pr(\text{MD}) = P_e$ and $\Pr(\text{FA}) \leq P_e$. Fig. 8 shows the performance of coded demixing when K is unknown at the receiver for $G = 2$ at the E_b/N_0 values identified in Fig. 6. From this figure, it is clear that not knowing K incurs only a small performance penalty in terms of $\Pr(\text{MD})$ and $\Pr(\text{FA})$. We point out that, despite the fact that the receiver does not have exact knowledge of K, the devices themselves implicitly assume that $K \leq 275$. If K were to grow much larger, the code rate would eventually have to decrease as it did for the CCS-AMP curve in Fig. 6. This broad assumption about user density is akin to those employed in cellular systems.

V. CONCLUSION

In this article, a novel signal processing tool called coded demixing is presented that extends existing notions of convex demixing and coded compressed sensing (CCS) to enable the
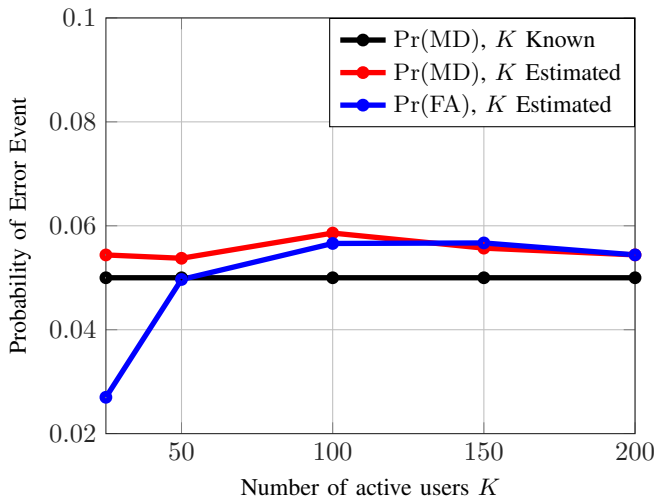
Fig. 8. This figure showcases the $\Pr(\mathrm{MD})$ and $\Pr(\mathrm{FA})$ of coded demixing when $G = 2$ for the case when K is unknown at the receiver. Transmit powers are selected to achieve $\Pr(\mathrm{MD}) = 0.05$ when K is known, serving as a performance benchmark. Clearly, not knowing K at the receiver incurs only a small performance penalty.

joint recovery of very high dimensional signals that are sparse with respect to separate bases. A pragmatic framework featuring an approximate message passing (AMP) based decoder is developed that enables all sent signals to be disentangled and for individual sparsity patterns to be recovered accurately with low-complexity.

This novel framework may be utilized in a heterogeneous multi-class URA setting where each class features distinct power, coding, and data requirements. In such a network, coded demixing is shown to significantly outperform traditional techniques such as treating interference as noise and successive interference cancellation. Somewhat surprisingly, coded demixing with stochastic binning may also be used to enhance single-class URA networks. This approach significantly improves PUPE performance because the number of users present on a single bipartite factor graph is dramatically reduced, which in turns boosts the performance of the BP decoder. It is shown that increasing the number of bins of users improves performance, up to a point. When the number of bins is too large, the operating point of the system may fall out of AMP's convergence region and/or the cross-coherence of the sensing matrices may grow too large for reliable demixing.

Coded demixing has a promising future within URA. Open research challenges include precisely characterizing the upper bound on the number of bins that may be employed within a single-class system and finding an optimal scheme for the heterogenous multi-class URA setting. Furthermore, incorporating fading into the proposed framework as well as extending the framework to accommodate multiple input multiple output (MIMO) scenarios remain open research directions.

APPENDIX A
DERIVATION OF ONSAGER TERM

In this section, we derive the form of the Onsager term for the coded demixing algorithm of Section II. Recall that the dynamic denoiser used within AMP is defined as

$$\boldsymbol{\eta}^{(t)}\left(\mathbf{r}^{(t)}\right) = \begin{bmatrix} \boldsymbol{\eta}_1^{(t)}\left(\mathbf{r}_1^{(t)}\right) \\ \vdots \\ \boldsymbol{\eta}_G^{(t)}\left(\mathbf{r}_G^{(t)}\right) \end{bmatrix}, \tag{17}$$

where the group denoisers are of the form

$$\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g) = \hat{\mathbf{s}}_g(1, \mathbf{r}_g, \tau_t) \cdots \hat{\mathbf{s}}_g(L_g, \mathbf{r}_g, \tau_t). \tag{18}$$

Elements within the denoiser for group g are given by

$$\hat{\mathbf{s}}_g(\ell, \mathbf{r}_g(\ell), \tau_t) = (\hat{s}_g(\mathbf{q}_g(\ell, k), \mathbf{r}_g(\ell, k), \tau_t) : k \in [2^{v_g}]) \tag{19}$$

and $\hat{s}_g(q, r, \tau_t)$ is the posterior mean estimate (PME) equal to

$$\hat{s}_g(q, r, \tau_t) =$$

$$\frac{q \exp\left(-\frac{(r - d_g)^2}{2\tau_t^2}\right)}{q \exp\left(-\frac{(r - d_g)^2}{2\tau_t^2}\right) + (1 - q) \exp\left(-\frac{r^2}{2\tau_t^2}\right)}, \tag{20}$$

We emphasize that this denoiser is utilized twice within the AMP iterate: once during the state update to promote sparsity, and once during the residual computation as part of the Onsager correction term. Below, we formally derive the needed expression for the Onsager term,

$$\frac{\mathbf{z}}{n} \mathrm{div} \mathbf{D} \boldsymbol{\eta}^{(t)} = \frac{\mathbf{z}}{n\tau_t^2} \left(\|\mathbf{D}^2 \boldsymbol{\eta}^{(t)}(\mathbf{r})\|_1 - \|\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_2^2 \right). \tag{21}$$

Before doing so, we first consider the partial derivative of (20), which is required in the computation of the Onsager term.

Lemma 6 (Adapted from [10], Lemma 7). *If the number of BP iterations on the outer factor graph is strictly less than the length of its shortest cycle, then the partial derivative of $\hat{s}_g(\mathbf{q}(\ell, k), \mathbf{r}(\ell, k), \tau_t)$ with respect to $\mathbf{r}(\ell, k)$ is equal to*

$$\frac{d_g}{\tau_t^2} \hat{s}_g\left(\mathbf{q}(\ell, k), \mathbf{r}(\ell, k), \tau_t\right) \left(1 - \hat{s}_g\left(\mathbf{q}(\ell, k), \mathbf{r}(\ell, k), \tau_t\right)\right). \tag{22}$$

Proof: For simplicity of notation, we use $q = \mathbf{q}(\ell, k)$, $r = \mathbf{r}(\ell, k)$, and $\tau = \tau_t$. Then, we can write

$$\hat{s}_g(q, r, \tau) = \frac{q \exp\left(-\frac{(r - d_g)^2}{2\tau^2}\right)}{q \exp\left(-\frac{(r - d_g)^2}{2\tau^2}\right) + (1 - q) \exp\left(-\frac{r^2}{2\tau^2}\right)}$$

$$= \frac{q}{q + (1 - q) \exp\left(\frac{d_g^2 - 2rd_g}{2\tau^2}\right)}.$$

Because the number of BP iterations is less than the length of the shortest cycle, $\partial q / \partial r = 0$. Thus, using the chain rule of differentiation, we obtain

$$\frac{\partial \hat{s}_g(q, r, \tau)}{\partial r} = \frac{d_g}{\tau^2} \frac{q(1 - q) \exp\left(\frac{d_g^2 - 2rd_g}{2\tau^2}\right)}{\left(q + (1 - q) \exp\left(\frac{d_g^2 - 2rd_g}{2\tau^2}\right)\right)^2}$$

$$= \frac{d_g}{\tau^2} \hat{s}_g(q, r, \tau)(1 - \hat{s}_g(q, r, \tau)).$$

In this context, q acts as a prior or belief inherited from neighboring sections. When the number of BP iterations on the LDPC factor graph is strictly less than the length of its shortest cycle, the computation of $\mathbf{q}(\ell, k)$ does not depend on

the value of $\mathbf{r}(\ell, k)$. This assumption is crucial in obtaining the expression above, else the partial derivative of $\mathbf{q}(\ell, k)$ with respect to $\mathbf{r}(\ell, k)$ would have to be considered as well. ∎

Having established Lemma 6, we are ready to compute the divergence of the denoiser.

Proposition 7. *The divergence of $\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})$ with respect to $\mathbf{r}$ is given by*

$$\text{div}\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r}) = \frac{1}{\tau_t^2}\left(\|\mathbf{D}^2\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_1 - \|\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_2^2 \right), \quad (23)$$

provided that the number of BP iterations on the outer factor graph is strictly less than the length of the shortest cycle.

Proof: The divergence of $\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})$ may be expressed as

$$\text{div}\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r}) = \sum_{g=1}^{G} d_g \text{div}\,\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)$$
$$= \sum_{g=1}^{G} d_g \text{div}\left(\hat{\mathbf{s}}_g(1,\mathbf{r}_g,\tau_t)\cdots\hat{\mathbf{s}}_g(L_g,\mathbf{r}_g,\tau_t)\right)$$
$$= \sum_{g=1}^{G} d_g \sum_{\ell=1}^{L_g} \sum_{k=1}^{2^v} \frac{\partial\hat{\mathbf{s}}_{g,k}(\ell,\mathbf{r}_g,\tau_t)}{\partial\mathbf{r}_g(\ell,k)}.$$

Under the assumption that the number of BP iterations on the outer factor graph is strictly less than the length of the shortest cycle and using the result of Lemma 6, we get

$$\text{div}\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})$$
$$= \sum_{g=1}^{G} \sum_{\ell=1}^{L_g} \sum_{k=1}^{2^v} \frac{d_g^2}{\tau_t^2}\hat{\mathbf{s}}_{g,k}\left(\mathbf{r}_g(\ell),\tau_t\right)\left(1 - \hat{\mathbf{s}}_{g,k}\left(\mathbf{r}_g(\ell),\tau_t\right)\right).$$

The above expression may be represented as the sum of G inner products,

$$\text{div}\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r}) = \sum_{g=1}^{G} \frac{1}{\tau_t^2}\langle d_g\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g), d_g\mathbf{1} - d_g\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)\rangle$$
$$= \sum_{g=1}^{G} \frac{1}{\tau_t^2}\left(\|d_g^2\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)\|_1 - \|d_g\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)\|_2^2 \right).$$

Leveraging the short-hand forms

$$\left\|\mathbf{D}^2\boldsymbol{\eta}^{(t)}(\mathbf{r})\right\|_1 = \sum_{g=1}^{G} \left\|d_g^2\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)\right\|_1$$
$$\left\|\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})\right\|_2^2 = \sum_{g=1}^{G} \left\|d_g\boldsymbol{\eta}_g^{(t)}(\mathbf{r}_g)\right\|_2^2,$$

the divergence of $\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})$ simplifies to

$$\text{div}\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r}) = \frac{1}{\tau_t^2}\left(\|\mathbf{D}^2\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_1 - \|\mathbf{D}\boldsymbol{\eta}^{(t)}(\mathbf{r})\|_2^2 \right).$$

This completes the proof. ∎

The full Onsager term in (21) follows directly from the results of Proposition 7.

References

[1] Y. Polyanskiy, "A perspective on massive random-access," in *Proc. Int. Symp. on Information Theory*. IEEE, 2017, pp. 2523–2527.

[2] O. Ordentlich and Y. Polyanskiy, "Low complexity schemes for the random access Gaussian channel," in *Proc. Int. Symp. on Information Theory*. IEEE, 2017, pp. 2528–2532.

[3] A. Vem, K. R. Narayanan, J. Cheng, and J.-F. Chamberland, "A user-independent serial interference cancellation based coding scheme for the unsourced random access Gaussian channel," in *Information Theory Workshop (ITW)*. IEEE, 2017, pp. 121–125.

[4] A. Pradhan, V. Amalladinne, A. Vem, K. R. Narayanan, and J.-F. Chamberland, "A joint graph based coding scheme for the unsourced random access gaussian channel," in *Global Communications Conference (GLOBECOM)*. IEEE, 2019.

[5] E. Marshakov, G. Balitskiy, K. Andreev, and A. Frolov, "A polar code based unsourced random access for the Gaussian MAC," in *Proc. Vehicular Tech. Conf.* IEEE, 2019.

[6] A. K. Pradhan, V. K. Amalladinne, K. R. Narayanan, and J.-F. Chamberland, "Polar coding and random spreading for unsourced multiple access," in *International Conference on Communications (ICC)*. IEEE, 2020.

[7] M. J. Ahmadi and T. M. Duman, "Random spreading for unsourced MAC with power diversity," *IEEE Commun. Letters*, vol. 25, no. 12, pp. 3995–3999, 2021.

[8] V. K. Amalladinne, J.-F. Chamberland, and K. R. Narayanan, "A coded compressed sensing scheme for unsourced multiple access," *IEEE Trans. Inform. Theory*, vol. 66, no. 10, pp. 6509–6533, 2020.

[9] A. Fengler, P. Jung, and G. Caire, "SPARCs for unsourced random access," *IEEE Trans. Inform. Theory*, vol. 67, no. 10, pp. 6894–6915, 2021.

[10] V. K. Amalladinne, A. K. Pradhan, C. Rush, J.-F. Chamberland, and K. R. Narayanan, "Unsourced random access with coded compressed sensing: Integrating AMP and belief propagation," *IEEE Trans. Inform. Theory*, vol. 68, no. 4, pp. 2384–2409, 2021.

[11] Z. Han, X. Yuan, C. Xu, S. Jiang, and X. Wang, "Sparse Kronecker-product coding for unsourced multiple access," *IEEE Wireless Commun. Letters*, vol. 10, no. 10, pp. 2274–2278, 2021.

[12] R. Calderbank and A. Thompson, "CHIRRUP: A practical algorithm for unsourced multiple access," *Information and Inference*, 2019, iaz029.

[13] J. R. Ebert, V. K. Amalladinne, J.-F. Chamberland, and K. R. Narayanan, "A hybrid approach to coded compressed sensing where coupling takes place via the outer code," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 4770–4774.

[14] G. Kasper Facenda and D. Silva, "Efficient scheduling for the massive random access Gaussian channel," *IEEE Trans. on Wireless Comm.*, vol. 19, no. 11, pp. 7598–7609, 2020.

[15] A. Decurninge, I. Land, and M. Guillaud, "Tensor-based modulation for unsourced massive random access," *IEEE Wireless Commun. Letters*, vol. 10, no. 3, pp. 552–556, 2021.

[16] V. Shyianov, F. Bellili, A. Mezghani, and E. Hossain, "Massive unsourced random access based on uncoupled compressive sensing: Another blessing of massive MIMO," *IEEE J. Select. Areas Commun.*, vol. 39, no. 3, pp. 820–834, 2021.

[17] E. Nassaji, M. Bashir, and D. Truhachev, "Unsourced random access over fading channels via data repetition, permutation, and scrambling," *IEEE Trans. on Commun.*, vol. 70, no. 2, pp. 1029–1042, 2022.

[18] M. B. McCoy, V. Cevher, Q. T. Dinh, A. Asaei, and L. Baldassarre, "Convexity in source separation: Models, geometry, and algorithms," *IEEE Signal Processing Mag.*, vol. 31, no. 3, pp. 87–95, 2014.

[19] M. B. McCoy and J. A. Tropp, "Sharp recovery bounds for convex demixing, with applications," *Foundations of Computational Mathematics*, vol. 14, no. 3, pp. 503–567, 2014.

[20] D. Amelunxen, M. Lotz, M. B. McCoy, and J. A. Tropp, "Living on the edge: phase transitions in convex programs with random data," *Information and Inference: A Journal of the IMA*, vol. 3, no. 3, pp. 224–294, 2014.

[21] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky, "The convex geometry of linear inverse problems," *Foundations of Computational Mathematics*, vol. 12, no. 6, pp. 805–849, 2012.

[22] C. Hegde and R. G. Baraniuk, "Signal recovery on incoherent manifolds," *IEEE Trans. Inform. Theory*, vol. 58, no. 12, pp. 7204–7214, 2012.

[23] B. N. Bhaskar, G. Tang, and B. Recht, "Atomic norm denoising with applications to line spectral estimation," *IEEE Trans. Signal Processing*, vol. 61, no. 23, pp. 5987–5999, 2013.

[24] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Trans. Inform. Theory*, vol. 57, no. 2, pp. 764–785, 2011.

[25] D. L. Donoho, A. Javanmard, and A. Montanari, "Information-theoretically optimal compressed sensing via spatial coupling and approximate message passing," *IEEE Trans. Inform. Theory*, vol. 59, no. 11, pp. 7434–7464, 2013.

[26] R. Berthier, A. Montanari, and P.-M. Nguyen, "State evolution for approximate message passing with non-separable functions," *Information and Inference*, vol. 9, no. 1, pp. 33–79, 2020.

[27] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[28] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," in *Information Theory Workshop (ITW)*. IEEE, 1998, pp. 70–71.

[29] H. Song and J. R. Cruz, "Reduced-complexity decoding of Q-ary LDPC codes for magnetic recording," *IEEE Trans. on Magnetics*, vol. 39, no. 2, pp. 1081–1087, 2003.

[30] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. National Academy of Sciences*, vol. 106, no. 45, pp. 18914–18919, 2009.

[31] A. Hao, S. Rini, V. K. Amalladinne, A. K. Pradhan, and J.-F. Chamberland, "An exploration of the heterogeneous unsourced MAC," *arXiv preprint arXiv:2011.10900*, 2020.

[32] Y.-C. Huang, S.-L. Shieh, Y.-P. Hsu, and H.-P. Cheng, "Iterative collision resolution for slotted ALOHA with NOMA for heterogeneous devices," *IEEE Trans. on Commun.*, vol. 69, no. 5, pp. 2948–2961, 2021.

[33] V. K. Amalladinne, A. Hao, S. Rini, and J.-F. Chamberland, "Multi-class unsourced random access via coded demixing," in *Proc. Int. Symp. on Information Theory*. IEEE, 2021, pp. 3080–3085.

[34] J. R. Ebert, V. K. Amalladinne, Stefano Rini, J.-F. Chamberland, and K. R. Narayanan, "Stochastic binning and coded demixing for unsourced random access," in *International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2021, pp. 351–355.

[35] K.-H. Ngo, A. Lancho, G. Durisi, and A. Graell i Amat, "Massive uncoordinated access with random user activity," in *Proc. Int. Symp. on Information Theory*. IEEE, 2021.

Stefano Rini is an Associate Professor in the Department of Electrical and Computer Engineering at the National Yang Ming Chiao Tung University (NYCU) (previously National Chiao-Tung University (NCTU)), Hsinchu, Taiwan. He received the B.A. degree in computer science from the Politecnico di Milano, Italy, in 2005, the M.S. degree in both Electrical and Computer Engineering and Statistics from the University of Illinois Chicago (UIC) in 2009. He earned his doctoral degree in Electrical and Computer Engineering at the University of Illinois Chicago (UIC), USA, in 2011 under the supervision of Professor Tuninetti and co-adviser Professor Devroye, with a thesis titled "Cognition and Cooperation in Wireless Networks: an Information Theoretic Perspective". In 2012, he was a postdoctoral fellow at the Institute for Communications Engineering at the Technical University of Munich (TUM), Germany with Professor Kramer. In 2013 he was a postdoctoral fellow at the Department of Electrical Engineering at Stanford University, USA, with Professor Goldsmith. In 2014, he joined NYCU.

Jean-Francois Chamberland is a professor in the Department of Electrical and Computer Engineering at Texas A&M University. He completed the B.Eng. degree at McGill University, the M.S. degree at Cornell University, and the Ph.D. degree at the University of Illinois, Urbana-Champaign. His research interests are in the areas of communications and information theory, statistical inference, computer systems and networks, and learning. Recently, he has been studying the efficient design of wireless systems and the fundamental limits of communication networks. His contributions have been recognized through an IEEE Young Author Best Paper Award from the IEEE Signal Processing Society, a Faculty Early Career Development (CAREER) Award from the National Science Foundation (NSF), and an IEEE Communications Society & Information Theory Society Joint Paper Award. He currently serves as a senior area editor for the IEEE Open Journal of Signal Processing, and he was an associate editor for the IEEE Transactions on Information Theory.

Jamison R. Ebert received the B.S. degree in electrical engineering from Brigham Young University in 2020. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering at Texas A&M University in College Station, TX, USA. His research interests include digital communications, coding theory, and compressed sensing.

Krishna R. Narayanan received the B.E. degree from Coimbatore Institute of Technology, M.S. degree from Iowa State University and Ph.D. degree in Electrical Engineering from Georgia Institute of Technology in 1998. Since 1998, he has been with the Department of Electrical and Computer Engineering at Texas A&M University, where he is currently the Eric D. Rubin '06 professor. His current research interests are in the design of massive uncoordinated multiple access schemes, coded distributed computing, exploring connections between coding theory and sparse signal recovery and in analyzing data defined on graphs. He was the recipient of the NSF career award in 2001. He also received the 2006 and 2020 best paper awards from the IEEE technical committee for signal processing for data storage. In 2014, he received the Professional Progress in Engineering award given to an outstanding alumnus of Iowa State University. He was awarded a university level distinguished teaching award in 2018 at Texas A&M University. He has served as an associate editor for coding techniques for the IEEE Transactions on Information Theory and as the area editor (and editor) for the coding theory and applications area of the IEEE Transactions on Communications. He was elected as Fellow of the IEEE for contributions to coding for wireless communications and data storage.

Vamsi K. Amalladinne received the B.Tech. degree in electronics and communication engineering from the LNM Institute of Information Technology, Jaipur, India, in 2012, the M.Tech. degree in signal processing and communications from IIT Kanpur, India, in 2014, and the Ph.D. degree in electrical and computer engineering from Texas A&M University, College Station, USA, in 2021. From June 2014 to August 2016, he was employed as a DSP Firmware Developer for CDMA systems with Qualcomm, Hyderabad, India. He is currently employed as a Senior Research Engineer with the Wireless Research and Development Division, Qualcomm, San Diego, USA, where he is contributing to the research and standardization of next-generation Wi-Fi networks. His research interests are in wireless communication, signal processing, error control coding, compressed sensing, and machine learning.