Non-asymptotic Coded Slotted ALOHA

Mohammad Fereydounian
University of Pennsylvania
3401 Walnut St, Philadelphia, PA 19104
mferey@seas.upenn.edu

Xingran Chen
University of Pennsylvania
3330 Walnut St, Philadelphia, PA 19104
mvjamali@umich.edu

Hamed Hassani
University of Pennsylvania
3401 Walnut St, Philadelphia, PA 19104
hassani@seas.upenn.edu

Shirin Saeedi Bidokhti
University of Pennsylvania
3330 Walnut St, Philadelphia, PA 19104
saeedi@seas.upenn.edu

Abstract

Coding for random access communication is a key challenge in Internet of Things applications. In this paper, the well-known scheme of Coded Slotted Aloha (CSA) is considered and its performance is analyzed in the non-asymptotic regime where the frame length and the number of users are finite. A density evolution framework is provided to describe the dynamics of decoding, and fundamental limits are found on the maximum channel load (i.e., the number of active users per time slot) that allows reliable communication (successful decoding). Finally, scaling laws are established, describing the non-asymptotic relation between the probability of error, number of users, and the channel load.

1 Introduction

The technology of Internet of Things (IoT) has brought new challenges in the design of multi-access communication systems. In traditional networks, the number of users is small and it is hence practical to coordinate them for transmission. For example, this can be done with the help of a common clock that can be implemented through a low rate (vanishing by blocklength) communication link. In IoT application, however, the number of users is very large, orders of magnitude larger than the blocklength. For example, in Low-Power Wide-Area Networks (P-WANs), the number of users is in the order of tens of millions. Clearly, coordinating all the users is infeasible in such scenarios and hence communication should be assumed uncoordinated. This has motivated the key challenge of coding for massive random access communication, where senders communicate their packets in a bursty manner at random times.

The literature on massive random access communciation ranges from traditional (slotted) ALOHA-type protocols [1–3], and their coded variants [4–6], to recent information theoretic frameworks and code designs in the regime of operation where the total number of users scale linearly with the blocklength [7–10]. Slotted ALOHA (SA) [1–3] is one of the first random-access protocols that deals with collision through random retransmission of packets and it is still in use for cellular and satellite communication. In SA protocols, time is slotted and users randomly choose slots to send their packets. Naturally, without coordination, packets that are transmitted may collide in which case each involved users will send its packet later in another random timeslot. In multi-access channels in which the output is a superposition of the inputs, however, successive interference cancellation (SIC) can be implemented through iterative algorithms and it offers significant gains in spectral efficiency [4,5]. For a thorough review of random-access protocols and their challenges we refer to [6] and the references therein.

In [5,11], the problem of recovering from collisions has be tackled from the perspective of error correcting codes. The idea is to encode redundancy into the transmitted packets via repetition coding, leading to the class of Irregular Repetition Slotted ALOHA (IRSA) schemes [5], or more generally via an error correcting code, leading to the class of Coded Slotted ALOHA (CSA) schemes [11]. The redundancy is then exploited in decoding by successive cancellation and a corresponding iterative message passing algorithm. Exploiting a bipartite graph representation, [11] derives density evolution equations for CSA and analyzes the SIC process in an asymptotic setting where the frame length and the number of users both tend to infinity (their ratio remaining constant).

In this work, we provide a non-asymptotic analysis of CSA. The non-asymptotic regime is arguably the practical regime of interest especially in applications where delay is of importance. Previous work on random access schemes, including coded variants of SA, have mostly relied on simulations to address the non-asymptotic performance of the schemes. Some notable exceptions are [12–16]. The work in [12] gives an accurate analysis of frameless ALOHA (parallel to rateless codes) in the finite-length regime. The works in [13–15] analyze CSA in the error floor region, and [16] provides a finite-length analysis for IRSA in the waterfall region. To the best of our knowledge, a finite-length analysis of the more general class of CSA protocols in the waterfall region has been missing from the literature.

In summary, our contributions are as follows:

- We use connections between CSA and LDPC ensembles that were established in [11] and build on methods that were originally developed in [17,18] to analyze LDPC ensembles in the finite blocklength regime. In a recent work, [16] uses a similar approach and provides the non-asymptotic analysis of the special class of IRSA random access codes. While the existing analytical methods in LDPC ensembles are directly applicable to the analysis of IRSA, they turn out to be insufficient for the analysis of CSA . Assuming a regular CSA ensemble in which (active) users have the same rate and blocklength, we generalize the methods of LDPC ensembles to analyze the performance of regular CSA in the non-asymptotic regime. This generalization includes the derivation of a new density evolution as well as the components of the so-called *scaling law* which describe the probability of error.
- Channel load G is defined to be the number of active users per time slot. It turns out that CSA ensembles encounter a thresholding effect, meaning that there is a maximum achievable channel load G^* such that for $G > G^*$, it is almost surely impossible to decode all messages and for $G < G^*$ it is almost surely possible. We provide an analytical approach to obtain G^* as a root of an algebraic equation.
- Finally, we simulate the CSA ensemles we defined earlier and verify our results with the simulation outcomes.

The paper is organized as follows. Section 2 presents the modeling and an overview on the the SIC decoding process that is used throughout the paper. Section 3.1 derives the differential equations that describe this process. Section 3.2, solves the differential equations and Section 3.3, finds the maximum achievable channel load as a root of an algebraic equation. Section 4 provides the probability of error in the waterfall region for the non-asymptotic settings. Section 5, presents simulation results that validates our derivations.

2 Modeling

Suppose we have a multi-access channel with N users, each of which is active with probability p. An active user is one that has a message to send at the current time. Let N_a denote the expected number of active users; i.e., $N_a = pN$. In Slotted Aloha (SA) protocols, channel resources (e.g., time) are divided into slots. Let M denote the number of time slots. Channel load G is then defined as

$$G = \frac{N_a}{M} = \frac{pN}{M}.$$

Each user chooses a set of time slots to send its message. If two users transmit during the same time slot, a *collision* occurs. In case of collision, the receiver has only access to the summation of all the collided packets (at the time slot in which collision occurs). We also assume that a packet cannot be recovered in a time slot unless all the other collided packets are previously recovered and hence can be subtracted from the summation.

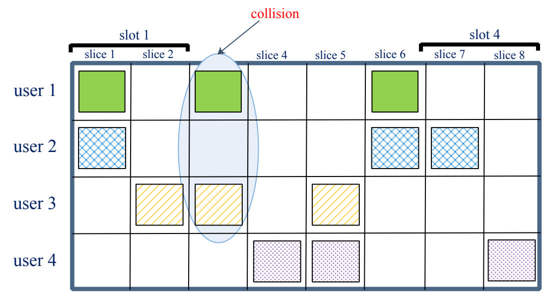
If all active users encode their messages as one packet to be sent in one time slot, then it is very likely that the entire message is lost due to collision. Therefore, to increase reliability, one can encode a message in one packet but repeat this packet n times. This way, the information rate is reduced to $R = 1/n$ but recovering only one of these packets is enough to recover the message.

This is called Repetition Slotted ALOHA (RSA). More generally, one can use different repetition lengths for users and design what is known as an Irregular Repetition Slotted ALOHA scheme [5]. Note that even under full recovery, the information rate for each user in RSA is $R = 1/n$ i.e., each packet is repeated n times. This means higher reliability is attained at the expense of lower rates. Replacing repetition coding with a general coding scheme is a fundamental way to increase the rate while ensuring high reliability in communication systems. This approach leads to Coded Slotted Aloha (CSA) introduced in [11]. CSA is clearly more expensive than RSA in terms of system complexity (leading to more expensive equipments, shorter battery life, larger processing time), however, CSA provides higher throughput for each user.

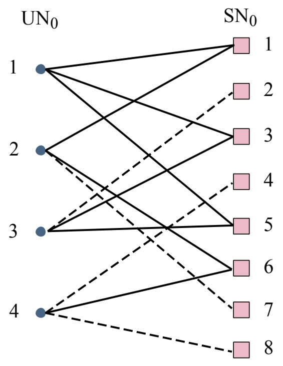
The CSA scheme is described as follows: Each time slot is divided into k smaller time slices. We thus have $m = kM$ time slices in total. Also, each message is divided into k smaller packets. Now, a coding scheme is used to encode these k packets into n coded packets. Each user chooses n distinct time slices out of m time slices uniformly at random to send its n coded packets. The setting is illustrated in Figure **1**a for 4 users with $n = 3, k = 2$, and $m = 8$. We assume that fully recovering an arbitrary subset of k coded packets out of n coded packets is enough to recover the original message. Therefore, the resulting rate is $R = k/n$ (note that the special case $k = 1$ is equivalent to RSA). For simplicity, we consider a regular scenario in which all users utilize the same blocklength n. We refer to such a CSA scheme by CSA(n, k, N_a, M).

For decoding, we use a successive interference cancelation (SIC) procedure: At each step q, $q = 0, 1, 2, \dots$, we first find a collision-free time slice containing only one packet and then we fully recover that packet. From this packet we identify its user. We mark this packet as "decoded". For any user, when k out of n packets are marked as decoded, then the user's original message can be fully recovered, and as a result, all of its n packets are known. We thus can After that, we subtract those n packets from their corresponding time slices, remove the user and we say the user is *resolved*. This elimination will result in more collision-free slices with only one packet inside. We then continue decoding by finding the next collision-free slice. The decoding stops when there is no collision-free slice. At this point, if all users are resolved, decoding is successful, otherwise, we declare a *B-error* (in analogy to block error in LDPC ensembles). If a B-error occurs, the fraction of unresolved users is statistically known as *packet loss probability* (PLP). In terms of analysis, PLP can be simply computed in terms of the probability of B-error (as we will see in the following).

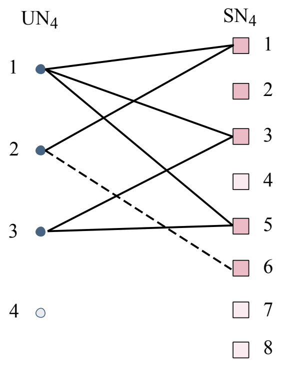
For a refined analysis of the SIC process discussed above, we need a more structured modeling based on a bipartite graph which we call the decoding graph. This is analogous to the so-called Tanner graph used for the analysis of LDPC codes [19]. Define a bipartite graph K with two sets of nodes: A set of N_a user nodes UN



(a) Time slice allocation



(b) K_0



(c) K_4

Figure 1: A realization of CSA scheme with parameters $N_a = 4, M = 4, n = 3, k = 2, m = kM = 8$. Figure **1**a represents the initial time slice allocation. Figures **1**b and **1**c illustrate the residual graphs at the beginning of iterations $q = 0$ and $q = 4$, respectively. Note that user node 4 is removed at iteration $q = 3$. In both graphs, the edges connected to singletons depicted in dashed lines

and a set of m slice nodes SN. The i^{th} user node represents the i^{th} active user, where $i \in \{1, \ldots, N_a\}$ and the j^{th} slice node represents the j^{th} time slice, where $j \in \{1, \ldots, m\}$. For each user, there is an edge to the n slice nodes in which its packets are sent. As a result, each user node has degree n. However, the degree of a slice node is potentially between 0 and N_a. We recall that neighbours of a user node are chosen uniformly at random among slice nodes. A slice node with degree 1 is called a *singleton*.
The SIC decoding process discussed above can be expressed in terms of the decoding graph as follows: At each step q, we find a singleton, and remove from the graph the singleton along with the edge connected to it. We then consider the user connected to the singleton. If the corresponding user node has degree $n - k$, that user node together with all connecting edges are removed. We continue until no singleton is found. At each step, by the graph resulted from the previous step is called *residual graph*. Figure **1**a shows the initial time slice allocation while Figures **1**a, **1**b, and **1**c illustrate the SIC process for a CSA realization with 4 users and parameters $n = 3, k = 2$, and $m = 8$. Figures **1**b and **1**c represent the residual graphs at iterations $q = 0$ and $q = 4$. As it can be seen, the initial graph K_0 includes 4 singletons namely, slice nodes $2, 4, 7$, and 8. The slice nodes $2, 4, 7$, will be removed in iterations $q = 0, 1, 2$, respectively. In iteration $q = 3$, we first remove the slice node 8. At this point, 2 out 3 slice nodes connecting to user node 4 are decoded. This means user node 4 is resolved and it must be removed from the graph. This makes a new singleton i.e., slice node 6 and the process goes to iteration 4. We refer to this SIC decoding process as the *peeling* process. Algorithm 1 presents the pseudo code of the peeling process.

2.1 Notation and Definitions

Throughout this paper, the set of edges of a graph K is denoted by $E(K)$, the number of elements of a set S is denoted by $|S|$, and the expectation of a random variable X is denoted by $\mathbb{E}[X]$. Consider a continuous time t. Suppose step q of the decoding process, where $q \in \{0, 1, 2, \ldots\}$, happens at $t = q\Delta t$ where $\Delta t = 1/E$ and $E = nN_a$ is the total number of edges in K. Also, let t_f be the time at which decoding stops. Let K_t be the residual graph at time t and UN_t and SN_t be the corresponding sets of user nodes and slice nodes of K_t. Note that $K_0 = K$, $UN_0 = UN$, $SN_0 = SN$ and thus $|UN_0| = N_a$ and $|SN_0| = m = kM$. Now for $i \in \{n - k + 1, \ldots, n\}$ and $t \in [0, t_f]$ define

$$L_i(t) := \mathbb{E}\big[\,|\{e \in E(K_t) : \; e \text{ is connected to } u \in UN_t \text{ with } \deg(u) = i\}|\,\big],$$

Algorithm 1 Peeling process for CSA(n, k, N_a, M)

```
    Input: The bipartite graph $K$ with two sets of nodes $UN$ and $SN$ connected by CSA($n, k, N_a, M$) protocol.
    Output: B-error $\in \{0, 1\}$.
 1: top:
 2: if $UN = \emptyset$ then
 3:     B error $= 0$
 4:     Break
 5: end if
 6: if $\exists s \in SN : \deg(s) = 1$ then
 7:     $N(s) :=$ the neighbour of $s$ in $UN$
 8:     Remove the node $s$ and the edge connected to $s$
 9:     if $\deg(N(s)) = n - k$ then
10:         remove $N(s)$ and the edges connected to $N(s)$
11:     end if
12:     goto top
13: end if
14: B-error =1
```

$$l_i(t) := \frac{L_i(t)}{E} = L_i(t)\Delta t,$$

$$\lambda_i := l_i(t)\Big|_{t=0}.$$

Similarly for $j \in \{0, \ldots, N_a\}$ and $t \in [0, t_f]$ define

$$R_j(t) := \mathbb{E}\left[\, |\{e \in E(K_t) : \ e \text{ is connected to } s \in SN_t \text{ where } \deg(s) = j\}| \,\right], \tag{1}$$

$$r_j(t) := \frac{R_j(t)}{E} = R_j(t)\Delta t, \tag{2}$$

$$\rho_j := r_j(t)\Big|_{t=0}, \tag{3}$$

$$P_j := \Pr\{\text{a random } s \in SN_a \text{ satisfies } \deg(s) = j\}. \tag{4}$$

Also define

$$e(t) := \frac{\mathbb{E}\left[|E(K_t)|\right]}{E}. \tag{5}$$

Note that

$$e(t) = \sum_{i=1}^{N_a} l_i(t) = \sum_{j=1}^{m} r_j(t), \tag{6}$$

$$\rho_j = \frac{jP_j}{\sum_j jP_j}. \tag{7}$$

3 Asymptotic Analysis

3.1 Differential Equations for Density Evolution

Consider the CSA(n, k, N_a, M) model defined in Section 2. By applying the peeling process to the elements of this model, as N_a increases, the sequence of residual graphs closely follows a "typical path". We will describe this path, characterize the typical deviation from it, and derive a set of differential equations describing this path. Note that the peeling process can be viewed as a stochastic process with small increments, i.e., at each step a singleton is chosen and is peeled from the graph along with the connecting user node. Hence, by using standard arguments based the Wormald's method [20,21], one can show that for large values of N_a, the behavior of individual instances follow a "typical behavior" or an "expected behavior" with high probability and such typical behavior can be expressed as the solution of a differential equation (see [22], [23, Appendix C]).

We now construct a set of coupled differential equations describing the typical behavior discussed above. We call the sequence $\{l_i(t), r_j(t)\}_{i,j}$, the degree distribution of residual graph at time t. The degree distribution of the residual graph constitutes a sufficient statistic for tracking the distribution of the residual graph. This is because given this degree distribution all residual graphs which are compatible with this degree distribution are equally likely. Therefore, in order to analyze the behavior of the decoder, it suffices to analyze the evolution of this degree distribution. In order to obtain differential equations describing the dynamic of $l_i(t)$ and $r_j(t)$, our first step is to compute the change in $L_i(t)$ at each step of the peeling process i.e., computing $L_i(t+\Delta t) - L_i(t)$. As defined earlier, $L_i(t)$ denotes the expected number of edges which are connected to degree i user nodes at time t. Consider a step of peeling process at time t. First, a singleton (a degree 1 slice node) $s \in SN_t$ is chosen arbitrarily. Let e be the connecting edge. Then, s and e are removed. By this removal, a change in $L_i(t)$ happens only when e is connected to a degree i or $i+1$ user node. With probability $l_i(t)/e(t)$, e is connected to a degree i user node u. In this case, by removing e, u is not of degree i anymore and thus $L_i(t)$ will be decreased by i units. Also, with probability $l_{i+1}(t)/e(t)$, e is connected to a degree $i+1$ user node u'. In this case, by removing e, u' becomes a degree i node and thus $L_i(t)$ will be increased by i units. Note that the latter case is valid when $i < n$ since $i+1$ must exist. Thus, based on this argument, we have the following equations for case $n-k+1 \le i \le n-1$ and case $i = n$.

$$\begin{cases} L_i(t+\Delta t) - L_i(t) = -i \cdot \dfrac{l_i(t)}{e(t)} + i \cdot \dfrac{l_{i+1}(t)}{e(t)}, & i \in \{n-k+1, \ldots, n-1\}, \\ L_n(t+\Delta t) - L_n(t) = -n \cdot \dfrac{l_n(t)}{e(t)}. \end{cases}$$

Replacing $l_i(t) = L_i(t)\Delta t$ results in

$$\begin{cases} \dfrac{l_i(t+\Delta t) - l_i(t)}{\Delta t} = -i \cdot \dfrac{l_i(t)}{e(t)} + i \cdot \dfrac{l_{i+1}(t)}{e(t)}, & i \in \{n-k+1, \ldots, n-1\}, \\ \dfrac{l_n(t+\Delta t) - l_n(t)}{\Delta t} = -n \cdot \dfrac{l_n(t)}{e(t)}. \end{cases} \tag{8}$$

Now, as $N_a \to \infty$, we have $E = nN_a \to \infty$ and as a result, $\Delta t = 1/E \to 0$. Therefore, (8) becomes

$$\begin{cases} \dfrac{dl_i(t)}{dt} = i \cdot \dfrac{l_{i+1}(t) - l_i(t)}{e(t)}, & i \in \{n-k+1, \ldots, n-1\}, \\ \dfrac{dl_n(t)}{dt} = -n \cdot \dfrac{l_n(t)}{e(t)}. \end{cases} \tag{9}$$

The approach for deriving similar equations for r_j is quite the same but needs further explanation. Consider the decoding step happening at time t. Define $a(t)$ to be the expected number of the edges that are removed in this step. Let $s \in SN_t$ be the singleton that is removed and e be the connecting edge. With probability $l_{n-k+1}(t)/e(t)$, e is connected to a degree $n-k+1$ user node u. In this case, after peeling s and e, u becomes of degree $n-k$. This results in removal of u which leads to removing $n-k$ other edges (connected to u) and in total $n-k+1$ edges will be removed in this step. Otherwise, if e is not connected to a degree $n-k+1$ user node, only e will be removed and decoding process goes to the next step. Thus, we have

$$a(t) = (n-k+1)\cdot\frac{l_{n-k+1}(t)}{e(t)} + \sum_{i=n-k+2}^{n} 1\cdot\frac{l_i(t)}{e(t)}. \tag{10}$$

Note that the expected number of deleted edges other than e, is $a(t)-1$. Now, consider one of these deleted edges, namely, e'. Then with probability $r_j(t)/e(t)$, e' is connected to a degree j slice node. In this case, $R_j(t)$ will be decreased by j units. Also, with probability $r_{j+1}(t)/e(t)$, e' is connected to a degree $j+1$ slice node and in this case, R_j will be increased by j units. This is true for any such e' and expected number of them is $a(t)-1$. Here, it is important to note that for j, the degree of a slice node, we have $j\in\{0,\dots,N_a\}$. Thus, the above argument is valid when $j<N_a$ since $j+1$ must exist. This argument results in the following equations:

$$\begin{cases} R_j(t+\Delta t) - R_j(t) = \left(-j\cdot\dfrac{r_j(t)}{e(t)} + j\cdot\dfrac{r_{j+1}(t)}{e(t)}\right)\cdot(a(t)-1), \qquad j\in\{2,\dots,N_a-1\}, \\ R_{N_a}(t+\Delta t) - R_{N_a}(t) = -N_a\cdot\dfrac{r_{N_a}(t)}{e(t)}\cdot(a(t)-1). \end{cases}$$

Substituting $r_j(t) = R_j(t)\Delta t$, gives

$$\begin{cases} \dfrac{r_j(t+\Delta t) - r_j(t)}{\Delta t} = \left(-j\cdot\dfrac{r_j(t)}{e(t)} + j\cdot\dfrac{r_{j+1}(t)}{e(t)}\right)\cdot(a(t)-1), \qquad j\in\{2,\dots,N_a-1\}, \\ \dfrac{r_{N_a}(t+\Delta t) - r_{N_a}(t)}{\Delta t} = -N_a\cdot\dfrac{r_{N_a}(t)}{e(t)}\cdot(a(t)-1). \end{cases} \tag{11}$$

Now, as $N_a\to\infty$, we have $E=nN_a\to\infty$ and as a result, $\Delta t = 1/E\to 0$ and only the first equation in (11) matters. Therefore, (11) becomes

$$\frac{dr_j(t)}{dt} = j\cdot\left(r_{j+1}(t) - r_j(t)\right)\cdot\frac{a(t)-1}{e(t)}, \qquad j\geq 2. \tag{12}$$

Moreover, using (6), for $j=1$, we have

$$r_1(t) = e(t) - \sum_{j\geq 2} r_j(t). \tag{13}$$

In order to solve (9) and (12) analytically, the following change of variable turns out to be useful since it eliminates $e(t)$:

$$t \longmapsto x = \exp\left(\int_0^t \frac{d\tau}{e(\tau)}\right). \tag{14}$$

As a result, for the initial point, we have

$$t=0\longmapsto x=1.$$

Now, taking derivatives yields

$$\frac{dx}{x} = \frac{dt}{e(t)}.$$

For simplicity, we consider this new variable x as "time". By applying this change of variable to (9) and (12), we have

$$\begin{cases} \dfrac{dl_i(x)}{dx} = i\cdot\dfrac{l_{i+1}(x) - l_i(x)}{x}, \qquad i\in\{n-k+1,\dots,n-1\}, \\ \dfrac{dl_n(x)}{dx} = -n\cdot\dfrac{l_n(x)}{x}, \end{cases} \tag{15}$$

and

$$\frac{dr_j(x)}{dx} = -j \cdot \left(r_{j+1}(x) - r_j(x)\right) \cdot \frac{a(x)-1}{x}, \qquad j \geq 2. \tag{16}$$

Note that by using (13) and the change of variable $t \mapsto x$ described in (14), as $N_a \to \infty$, we have

$$e(x) = \sum_{i=n-k+1}^{n} l_i(x) = \sum_{j \geq 1} r_j(x). \tag{17}$$

As a result, we have

$$r_1(x) = e(x) - \sum_{j \geq 2} r_j(x). \tag{18}$$

In order to solve these equations, we need to determine their initial conditions. Recall that we defined

$$\lambda_i = l_i(t=0) = l_i(x=1)$$

$$\rho_j = r_j(t=0) = r_j(x=1),$$

and that λ_i denotes the fraction of edges connected to a degree i user node in the initial graph K_0. As we discussed earlier, initially each user node has degree n which means

$$\lambda_i = \begin{cases} 0, & n-k+1 \leq i < n, \\ 1, & i = n. \end{cases} \tag{19}$$

Determining ρ_j needs further computations. We ignore empty time slices (corresponding to $j=0$) and focus on $j \geq 1$. The following lemma gives ρ_j for $j \geq 1$.

Lemma 1. *Consider CSA(n, k, N_a, M) and ρ_j as defined in* (3). *Suppose $N_a \to \infty$ and $M \to \infty$ with $G = \frac{N_a}{M}$ to be a fixed constant. Then for $j \geq 1$, we have*

$$\rho_j = \frac{1}{(j-1)!}\left(\frac{G}{R}\right)^{j-1} \exp\left(-\frac{G}{R}\right), \tag{20}$$

where $R = k/n$.

Proof. See Appendix A.2. □

The set of differential equations given by (15), (16), and, (18) together with their initial conditions given by (19) and (20) characterize the evolution of degree distribution. In the rest of this manuscript, we refer to them as *density evolution*. As we discussed earlier density evolution is statistically sufficient to analyze the peeling process. Note that $r_1(x)$, starting from $\rho_1 = \exp(-G/R)$ at $x=1$, shows the fraction of singletons at time x. Considering x_f as the time at which decoding stops, we have $r_1(x_f) = 0$. Therefore, decoding is successful if and only if the residual graph K_{t_f} is empty or equivalently, $e(x_f) = 0$. Otherwise, a B-error is declared. Hence, evolution of $r_1(x)$ is critical to analyze the peeling process.

In order to find a closed form for $r_1(x)$, we need to analytically solve $r_j(x)$ for $j \geq 2$ and then replace them into (18). Solving $r_j(x)$ requires having $a(x)$ which itself depends on $l_i(x)$, $i \in \{n-k+1, \ldots, n\}$. Note that (15) shows a finite sequence of recursive differential equations which goes backward. One challenge to find an analytical solution here is that the length of this recursion varies with n. The other challenge is that even if we knew $l_i(x)$, $i \in \{n-k+1, \ldots, n\}$ (and hence $a(x)$), the differential equations for $r_j(x)$ in (16) form an infinite sequence of recursive differential equations which go backward. This means that each r_j has a coupled differential equation with the next element $r_{j+1}(x)$. At first glance, solving such a system seems challenging even numerically because we do not have any initial differential equation to start with. This is also the case in LDPC ensembles but the difference here is that unlike LDPC, guessing the solutions does not seem straightforward which may reduce the chance for finding closed form solutions. We will address these challenges in the next section. As we will see, using some tricks, both of these challenges will be resolved and closed form solutions for $l_i(x)$, $r_j(x)$, $e(x)$ and most importantly, $r_1(x)$, will be found. Moreover, based on these results, we also find an algebraic equation whose root is the maximum achievable channel load G^*. This equation can be solved numerically and G^* can be computed algebraically up to any desired precision. This approach gives a precise analytic way of determining G^*.

3.2 Solving Density Evolution

In this section, we provide solutions for the density evolution discussed in the previous section i.e., the differential equations (15), (16), and equation (18) with initial conditions given in (19) and (20). Solving these, requires some intermediate steps. Our method for solving such a recursive sequence of coupled differential equations, is first converting this sequence into a recursive sequence of numbers and then solving this sequence of numbers, using some combinatorial tricks. We used this general method for guessing the solution of l_i and r_j. Note that providing the details of how the solutions are being guessed is unnecessary and it suffices to only prove that these solutions satisfy the density evolution. All proofs are provided in Appendix.

Lemma 2. *The finite recursive sequence of differential equations defined in* (15) *together with initial conditions $l_i(1) = \lambda_i$, where λ_i is given by* (19)*, result in the following recursive sequence of solutions:*

$$\begin{cases} l_i(x) = \dfrac{i}{x^i}\displaystyle\int_1^x y^{i-1} l_{i+1}(y)dy, & i \in \{n-k+1, \dots, n-1\}, \\ l_n(x) = \dfrac{1}{x^n}. \end{cases} \tag{21}$$

Proof. See Appendix A.1. □

Theorem 1. *The finite recursive sequence of functions given by* (21)*, has the following solution:*

$$l_i(x) = \sum_{j=n-k+1}^{n} \frac{\alpha_j^{(i)}}{x^j}, \qquad i \in \{n-k+1, \cdots, n\}, \tag{22}$$

where $\left\{\alpha_j^{(i)}\right\}_{i,j}$, $n-k+1 \le i \le j \le n$, is a finite 2-dimensional recursive sequence of integers which is (uniquely) determined by the following equations:

$$\begin{cases} \alpha_n^{(n)} = 1 \\ \alpha_i^{(i)} = \displaystyle\sum_{j=i+1}^{n} (-1)^{j-i+1} \binom{j-1}{i-1} \alpha_j^{(j)}, & i \in \{n-1, n-2, \dots, n-k+1\} \\ \alpha_j^{(i)} = (-1)^{j-i} \dbinom{j-1}{i-1} \alpha_j^{(j)}, & j \in \{i+1, \cdots, n\} \end{cases} \tag{23}$$

Proof. See Appendix A.1. □

Remark 1. *Note that the elements in the sequence $\left\{\alpha_j^{(i)}\right\}_{i,j}$ are uniquely determined by* (23) *as follows: From the first equation, we have $\alpha_n^{(n)} = 1$. Then second equation gives $\alpha_{n-1}^{(n-1)}$ and then $\alpha_{n-2}^{(n-2)}$. Continue these derivations until $\alpha_1^{(1)}$ is obtained. Then all diagonal elements i.e., $\alpha_j^{(j)}$ are known. At this point, the third equation directly relates all other elements to the diagonal elements.*

Lemma 3. *Consider $e(x)$ as given in* (17)*, then we have*

$$e(x) = \sum_{j=n-k+1}^{n} \frac{\beta_j}{x^j}, \tag{24}$$

where β_j is a finite sequence of integers defined as the following:

$$\beta_j := \alpha_j^{(j)} (-1)^j \sum_{i=n-k+1}^{j} (-1)^i \binom{j-1}{i-1}, \tag{25}$$

where $\alpha_j^{(j)}$ given by (23)*.*

Proof. See Appendix A.2. □

In order to find the solution of the infinite recursive sequence of differential equations described in (16), we use a function $\lambda(x)$. Let $\lambda(x)$ be the function satisfying

$$\frac{\lambda'(x)}{\lambda(x)} = \frac{a(x)-1}{x}, \qquad \lambda(1) = 1, \tag{26}$$

where $a(x)$ is obtained from (10) by applying the change of variable (14).

Lemma 4. *Consider $\lambda(x)$ defined in* (26)*, then we have*

$$\lambda(x) = \exp\left((n-k) \int_1^x \frac{\sum_{j=0}^{k-1} \alpha_{n-j}^{(n-k+1)} y^j}{\sum_{j=0}^{k-1} \beta_{n-j} y^{j+1}} dy \right), \tag{27}$$

where $\alpha_j^{(j)}$ determined by (23) *and β_j defined as* (25)*.*

Proof. See Appendix A.2. □

Lemma 5. *The infinite recursive sequence of differential equations defined in* (16) *together with initial conditions $r_j(1) = \rho_j$ where ρ_j is given by* (20)*, result in the following infinite recursive sequence of solutions:*

$$r_j(x) = \frac{1}{\lambda^j(x)} \left(j \int_1^x r_{j+1}(y) \lambda^j(y) \frac{\lambda'(y)}{\lambda(y)} dy + \rho_j \right), \qquad j \geq 2, \tag{28}$$

where $\lambda(x)$ satisfies (26) *and is computed in* (27)*.*

Proof. See Appendix A.2. □

Theorem 2. *The infinite recursive sequence of functions given by* (28)*, has the following solution:*

$$r_j(x) = \frac{1}{(j-1)!\lambda^j(x)} \left(\frac{G}{R}\right)^{j-1} \exp\left(-\frac{G}{R\lambda(x)}\right), \quad j \geq 2. \tag{29}$$

Moreover, this result, together with (18) *and* (25)*, imply*

$$r_1(x) = \left[\sum_{j=n-k+1}^{n} \frac{\beta_j}{x^j} \right] - \frac{1}{\lambda(x)} \left(1 - \exp\left(-\frac{G}{R\lambda(x)}\right) \right), \tag{30}$$

where β_j defined by (25) *and $\lambda(x)$ is given by* (27)*.*

Proof. See Appendix A.2. □

Remark 2. *Theorem 2 is one the main results of this paper. Indeed,* (30) *is the key component in analysis of CSA(n, k, N_a, M) ensembles which reveals the dynamic of the number of singletons throughout peeling process. Moreover, as we will see in the next subsection,* (30) *makes it possible to compute G^* algebraically and directly from parameters of the system.*

3.3 Finding Maximum Achievable Channel Load G^*

The dynamic of decoding process is strongly based on $r_1(x)$. The analytical solution of $r_1(x)$ in (30) shows the dependency of $r_1(x)$ on G. For emphasizing on this dependency, we denote $r_1(x)$ by $r_1(x;G)$ whenever needed. $r_1(x)$ represents the expected value of normalized number of singletons at time x through peeling process. Suppose decoding stops at time x_f. As discussed earlier, the decoding is successful if and only if the residual graph at time x_f i.e., K_{x_f} is empty, otherwise, there are some unresolved user nodes left. Note that regardless of success or failure of the decoding, $r_1(x_f) = 0$.

It is known that nature of CSA ensembles yields a thresholding effect on channel load in an analogy to the thresholding effect on information rate of a communication system which leads to definition of capacity. In other words, there exists a threshold G^* such that if $G < G^*$, then the decoding succeeds with high probability and if $G > G^*$, it fails with high probability. In terms of density evolution, $G < G^*$ if and only if K_{x_f} is empty with high probability.

A visualization for such a transition is illustrated in Figure **2**. This figure shows three plots corresponding to three different channel loads $G_1 < G^* < G_2$. The plots visualize the appearance of typical shapes of $r_1(x)$ in CSA ensembles. It can be seen that as G increases, $r_1(x;G)$ goes downward. Starting from a very small channel load G_1, where the decoding is expected to be successful, $r_1(x;G_1)$ lies above the x-axis. As G increases, $r_1(x;G)$ goes downward all the way until it becomes tangent to x-axis. Let x^* be the tangent point, then this means that decoding stops at $r_1(x^*) = 0$ and because this happens at very early stages (corresponding to converted time x), this is the moment that decoding fails. Let G^* be the corresponding channel load at this moment. Keeping increasing G, results in $r_1(x;G)$ to further going down and thus the remaining in the state of decoding failure. Note that $r_1(x^*;G^*) = 0$. This equation does not distinguish between x^* and x_f and hence is not enough for computing G^*. A sharper argument for finding G^* is as follows: Start increasing G and plotting $r_1(x)$ for each G as illustrated for three values in Figure **2**. Consider $r_1(x;G_1)$ for some $G_1 < G^*$ and fix a point $x_2 \in (0, x_f)$. As G gets larger, there exists a moment (i.e., some G_2) where $r_1(x;G_2)$ hits x-axis at $x = x_2$ as shown in Figure **2**. Define function $\tilde{G}(x)$ that for a given x, returns such G corresponding to the hitting point. Therefore, for $x = x_2$, we have $\tilde{G}(x_2) = G_2$. In other words, for each x, we define $\tilde{G}(x)$ as the unique value satisfying

$$r_1\left(x;\tilde{G}(x)\right) = 0.$$

Computing $\tilde{G}(x)$ using (30) imply

$$\begin{aligned} r_1\left(x;\tilde{G}(x)\right) &= e(x) - \frac{1}{\lambda(x)}\left(1 - \exp\left(-\frac{\tilde{G}(x)}{R\lambda(x)}\right)\right) = 0, \\ \Leftrightarrow \tilde{G}(x) &= -R\lambda(x)\log\left(1 - e(x)\lambda(x)\right). \end{aligned} \tag{31}$$

Now, we argue that x^* is a local minimizer for $\tilde{G}(x)$. Consider a small neighborhood $N(x^*)$ around x^* in Figure **2**. In this neighborhood, by increasing G, x^* is the first point at which r_1 hits the x-axis by definition of x^*. This means $\forall x \in N(x^*) : \tilde{G}(x) \geq \tilde{G}(x^*)$. Thus, x^* is local minimizer of $\tilde{G}(x)$. Also, analytical expressions of $e(x)$ and $\lambda(x)$, given in (24) and (27), imply that $e(x)$ and $\lambda(x)$ are differentiable and as a result $\tilde{G}(x)$ is differentiable. Hence, this argument results in

$$\left.\frac{d\tilde{G}(x)}{dx}\right|_{x=x^*} = 0, \tag{32}$$

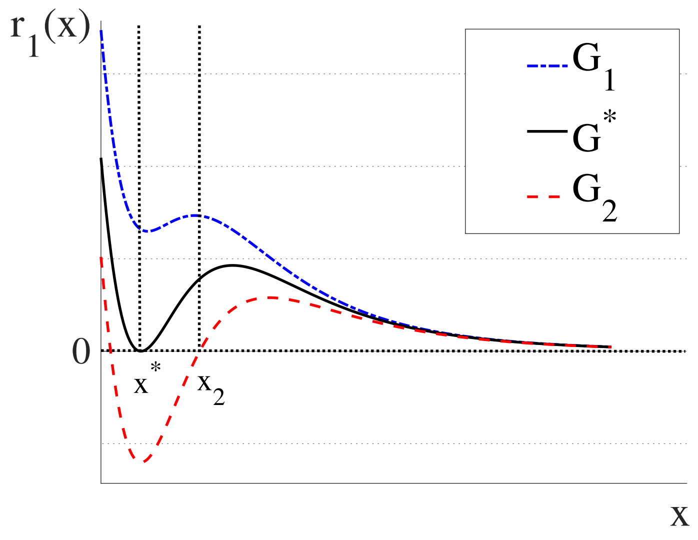


Figure 2: This figure illustrates the argument of derivation of x^* and G^*. Values of G satisfy $G_1 < G^* < G_2$ and we have $\tilde{G}(x_2) = G_2$, $\tilde{G}(x^*) = G^*$.

and we know that

$$G^* = \tilde{G}(x^*). \tag{33}$$

Thus, in order to obtain G^*, we first find x^* by solving (32) and then we replace x^* into (33). Since we have the analytical form of $e(x)$ and $\lambda(x)$, derivative of $\tilde{G}$ in (31) can be computed analytically. This results in an algebraic equation whose root is G^*. Theorem 3 finds this equation.

Theorem 3. *Consider $l_{n-k+1}(x)$, $e(x)$, $\lambda(x)$, and $\tilde{G}(x)$ which are given by* (22), (24), (27), *and* (31), *respectively, and define $h(x) = e(x)\lambda(x)$. Then the maximum achievable channel load G^* satisfies $G^* = \tilde{G}(x^*)$, where x^* is the solution of the following algebraic equation:*

$$\log(1 - h(x)) = \frac{1 - h(x)}{h(x)} \left(1 + \frac{xe'(x)}{(n-k)l_{n-k+1}(x)}\right). \tag{34}$$

Proof. See Appendix A.3. □

Note that alculation of the root of the algebraic equation (34) can be easily implemented by a single MATLAB command. Furthermore, in Section 5, comparing computed values of G^* from (34) to simulated values of G^* from non-asymptotic realizations of CSA shows that (34) provides an extremely accurate prediction of G^*. Indeed, they show an accuracy more than 99%.

4 Non-asymptotic Analysis: Probability of Error

Consider the definition of B-error defined in Section 2. Let $P_B(N_a, G)$ denote the probability of occurring B-error in the CSA(n, k, N_a, M) scheme under the discussion with N_a active users and $M = N_a/G$ time slots. As we discussed in the previous sections, it turns out that asymptotically i.e., as $N_a \to \infty$, $P_B(N_a, G)$ encounters a thresholding effect in terms of the channel load G. In other words, there is a threshold G^* that if $G > G^*$, then $P_B(N_a, G) \to 1$ and if $G < G^*$, $P_B(N_a, G) \to 0$. In the non-asymptotic case, however, this transition of P_B from 0 to 1 is not sharp and forms a smooth curve in the region where values of G are less than and close enough to the threshold G^*. This region is called the *waterfall* region. The behavior of P_B in the waterfall region turns out to be governed by the so-called *scaling law*. Results for the scaling law from statistical physics [24] suggest that there exist a non-negative constant μ and a non-negative function f such that the smooth transition in non-asymptotic case is expected to be governed as follows:

$$\lim_{\substack{N_a \to \infty \\ \text{s.t. } N_a^{1/\mu}(G^* - G) = z}} P_B(N_a, G) = f(z).$$

Inspired by the general scaling law, [18] analyzed the non-asymptotic behavior of LDPC ensembles. In analogy to that analysis, we will have the following formula for our setting:

$$P_B(N_a, G) = Q\left(\frac{\sqrt{N_a}}{\alpha}\left(G^* - \beta N_a^{-2/3} - G\right)\right). \tag{35}$$

where $Q(.)$ is the tail probability of standard normal distribution and the scaling parameters α and β will be obtained later in this section based on the asymptotic analysis. Note that in [18], the effect of the scaling parameter β has been proved numerically and analytically has been left as a conjecture.
Our approach for derivation of α and β will be generally based on [18]. Let $d = k + 1$ and define the $(d + 1)$-dimensional vector $z = (z_0, z_1, \cdots, z_d) = (r_1, r_2, l_{n-k+1}, \cdots, l_n)$. Note that $r_j(x)$ is related to two different notions, one is the number of edges connected to degree j slice nodes at time x and the other is the number of degree j slice nodes at time x. We refer to these two related values as corresponding edge-based and node-base quantities of r_j, respectively. We exploit similar definitions for $l_i(x)$. Now define $\delta^{(z_i z_j)}(x)$ to be the normalized covariance between corresponding node-base quantities of z_i and z_j at time x. The analysis of finite dimensional Markov processes over z then leads to the following set of $\binom{d+1}{2} + d + 1$ coupled differential equations for $\delta^{(z_i z_j)}$, where $i, j \in \{0, \cdots, d\}, i \leq j$:

$$\frac{d\delta^{(z_i z_j)}(x)}{dx} = \frac{e(x)}{x}\left[\frac{\hat{f}^{(z_i z_j)}(x)}{n} + \sum_{k=0}^{d} \delta^{(z_i z_k)}(x)\frac{\partial \hat{f}^{(z_j)}(x)}{\partial z_k} + \frac{\partial \hat{f}^{(z_i)}(x)}{\partial z_k}\delta^{(z_k z_j)}(x)\right], \tag{36}$$

where $\hat{f}^{(z_i)}(x)$ represents the expected change of the corresponding edge-based quantity of z_i and $\hat{f}^{(z_i z_j)}(x)$ represents the covariance between the corresponding edge-based quantities of z_i and z_j at time x. The functions $\hat{f}^{(z_i)}$ and $\hat{f}^{(z_i z_j)}$ are generally called local drifts and local covariances and we refer to the differential system (36) as *covariance evolution*. Note that (36) is obtained after applying the change of variable in (14). In order to solve covariance evolution, we need to first derive local drifts and local covariances for the formulation of CSA that we discussed in Section 2. The following theorem computes these values:
Theorem 4. *The local drifts and local covariances corresponding to the CSA scheme defined in Section 2, satisfy*

$$\begin{aligned}
\hat{f}^{(r_1)}(x) &= (n-k)\frac{l_{n-k+1}(x)}{e^2(x)}\left(r_2(x) - r_1(x)\right) - 1, \\
\hat{f}^{(r_2)}(x) &= -2(n-k)\frac{l_{n-k+1}(x)}{e^{2(x)}}r_2(x), \\
\hat{f}^{(l_i)}(x) &= \frac{i}{e(x)}(l_{i+1}(x) - l_i(x)), \qquad i \in \{n-k+1, \ldots, n\}, \\
\hat{f}^{(l_n)}(x) &= -n\frac{l_n(x)}{e(x)}, \\
\hat{f}^{(r_1 r_1)}(x) &= (n-k)\frac{l_{n-k+1}(x)}{e^3(x)}\left((n-k-1)(r_1(x) - r_2(x))^2 + (3r_1(x) - r_2(x))e(x)\right) + 1 - \left(\hat{f}^{(r_1)}(x)\right)^2, \\
\hat{f}^{(r_1 r_2)}(x) &= 2(n-k)(n-k-1)\frac{l_{n-k+1}(x)}{e^3(x)}r_2(x)\left(r_1(x) - r_2(x)\right) - \hat{f}^{(r_1)}(x)\hat{f}^{(r_2)}(x),
\end{aligned}$$

$$\hat{f}^{(r_1 l_{n-k+1})}(x) = -\frac{n-k+1}{e(x)}\left[l_{n-k+1}(x)((n-k)(r_2(x)-r_1(x))-e(x))+e(x)l_{n-k+2}(x)\right] - \hat{f}^{(r_1)}(x)\hat{f}^{(l_{n-k+1})}(x),$$

$$\hat{f}^{(r_1 l_i)}(x) = -\frac{i}{e(x)}\left(l_{i+1}(x)-l_i(x)\right) - \hat{f}^{(r_1)}(x)\hat{f}^{(l_i)}(x), \qquad i \in \{n-k+2,\dots,n-1\},$$

$$\hat{f}^{(r_1 l_n)}(x) = \frac{n l_n(x)}{e(x)} - \hat{f}^{(r_1)}(x)\hat{f}^{(l_n)}(x),$$

$$\hat{f}^{(r_2 r_2)}(x) = 4(n-k)\frac{l_{n-k+1}(x)}{e^3(x)}r_2(x)\left(e(x)+(n-k-1)r_2(x)\right) - \left(\hat{f}^{(r_2)}(x)\right)^2,$$

$$\hat{f}^{(r_2 l_{n-k+1})}(x) = 2(n-k)(n-k+1)\frac{l_{n-k+1}(x)}{e^2(x)}r_2(x) - \hat{f}^{(r_2)}(x)\hat{f}^{(l_{n-k+1})}(x),$$

$$\hat{f}^{(r_2 l_i)}(x) = -\hat{f}^{(r_2)}(x)\hat{f}^{(l_i)}(x), \qquad i \in \{n-k+2,\dots,n\},$$

$$\hat{f}^{(l_i l_i)}(x) = \frac{i^2}{e(x)}\left(l_i(x)+l_{i+1}(x)\right) - \left(\hat{f}^{(l_i)}(x)\right)^2, \qquad i \in \{n-k+1,\dots,n-1\},$$

$$\hat{f}^{(l_i l_{i+1})}(x) = -i(i+1)\frac{l_{i+1}(x)}{e(x)} - \hat{f}^{(l_i)}(x)\hat{f}^{(l_{i+1})}(x), \qquad i \in \{n-k+1,\dots,n-1\},$$

$$\hat{f}^{(l_i l_j)}(x) = -\hat{f}^{(l_i)}(x)\hat{f}^{(l_j)}(x), \qquad i \in \{n-k+1,\dots,n-2\},\, j > i+1,$$

$$\hat{f}^{(l_n l_n)} = n^2\frac{l_n(x)}{e(x)} - \left(\hat{f}^{(l_n)}(x)\right)^2,$$

where the functions $l_i(x)$, $r_j(x)$, and $e(x)$ are from Theorem 1, Theorem 2, and Lemma 3, respectively.

Proof. See AppendixA.4. □

In order to solve the covariance evolution in (36), we also need to compute the initial conditions of $\delta^{z_i z_j}(x)$. Note that due to the change of variable (14), the initial point is $x = 1$. The following lemma obtains these quantities:

Lemma 6. *The initial conditions for the covariance evolution* (36) *are*

$$\begin{aligned}
\delta^{(z_i z_j)}(1) &= 0, \qquad \text{if } i \notin \{0,1\} \text{ or } j \notin \{0,1\},\\
\delta^{(r_1 r_1)}(1) &= n\rho_1(1-\rho_1),\\
\delta^{(r_2 r_2)}(1) &= n\rho_2(1-\rho_2),\\
\delta^{(r_1 r_2)}(1) &= -n\rho_1\rho_2,
\end{aligned}$$

where the values of ρ_j are given as (20).

Proof. See AppendixA.4. □

In order to solve the covariance evolution (36), one needs to first compute local drifts and local covariances from Theorem 4 and intial conditions from Lemma 6. Solving covariance evolution reveals the functions $\delta^{(z_i z_j)}$ for all i, j including $\delta^{(r_1 r_1)}$. Using this result together with the components computed in Theorem 4, the scaling parameters α and β can be calculated as follows:

$$\begin{aligned}
\alpha &= -\sqrt{\frac{\delta^{(r_1 r_1)}(x)}{n}}\left(\frac{\partial r_1(x;G)}{\partial G}\right)^{-1}\Bigg|_{x=x^*,G=G^*},\\
\beta &= -\left(\frac{\hat{f}^{(r_1 r_1)}(x)}{n}\right)^{2/3}\left[\sum_{k=1}^{d}\frac{\partial \hat{f}^{(r_1)}(x)}{\partial z_k}\hat{f}^{(z_k)}(x)\right]^{-1/3}\left(\frac{\partial r_1(x;G)}{\partial G}\right)^{-1}\Bigg|_{x=x^*,G=G^*},
\end{aligned} \tag{37}$$

where $r_1(x;G)$ is given as (30). Note that the covariance evolution needs to be computed numerically to obtain α, however, β can be directly achieved from the values in Theorem 4. Furthermore, the derivative of $r_1(x;G)$ can be taken analytically from (30). In the next section, we provide simulation results in comparison to our asymptotic and non-asymptotic predictions.

5 Simulation Results

In this section, we will compare our predictions of maximum achievable channel load G^* and probability of B-error P_B with the outcomes of simulations.
Table 1, shows computation results of G^* from Theorem 3 versus values of G^* which are obtained from simulating the corresponding CSA scheme under various settings of n and k. These simulations are obtained with $N_a = 20000$ and by averaging over 2000 trials. This table actually reveals that true optimal channel load G^* can be approximated by Theorem 3 with a very high precision i.e., an accuracy more than 99%. Moreover, note that the implementation of (34) is also straightforward since $e'(x)$ can be determined analytically using (3) and as a result, (34) is a simple algebraic equation which can be solved simply by a single MATLAB commend.
In Figure **3**, we compare our non-asymptotic prediction from (35) to simulation results in the waterfall region. All plots sharing the same setting where $n = 5$ and $k = 3$ with solid lines corresponding to the computation of (35) using the scaling parameters $\alpha = 0.42362$ and $\beta = 0.8629$ obtained from (37) and dashed lines corresponding to simulations. The curves represent the plots for $N_a = 1000, 2000, 4000, 8000, 16000, 32000$ with the most left curve corresponding to $N_a = 1000$ and the most right curve corresponding to $N_a = 32000$. The curves in between, correspond to the other values of N_a in an order. The results from simulations (i.e., dashed lines) are acquired by averaging over 200000 trials. Moreover, the vertical dashed line accord with $G^* = 0.5840$ which is coming from (34). Several observations can be taken from Figure **3**. First, noting that the region for G is very small and the curves are plotted in a semi-log configuration, it can be seen that P_B from (35) accurately predicts the actual probability or B-error up to a very high precision. Second, it shows that as N_a increases, the non-asymptotic prediction from (35) and the true probability of B-error become closer and for $N_a = 32000$, they are almost coincided. In addition to being closer, they also converge to the vertical line as expected. Third, it can be seen that the thresholding effect is actually happening for the CSA scheme defined in Section 2. Forth, as it can be seen in the figure, the G^* computed from (34) is well approximating the true value of the thresholding effect.
Figure **4** illustrates the dynamic of the solution of density evolution for $r_1(x)$ as the values of G are increasing to reach G^*. The curves of $r_1(x)$ in Figure **4** are plotted based on (30) for $G = 0.45, 0.55, 0.65$ and $G^* = 0.7253$. From (34), the stopping time of the decoding process when $G = G^*$, is acquired as $x^* = 1.2822$. Replacing this x^* in (33), results in $G^* = 0.7253$.

A Appendix: Proofs

A.1 Proofs for Evolution of User Nodes

Proof of Lemma 2. First, from (15), for $l_n(x)$ we have

$$\frac{dl_n(x)}{dx} = -\frac{nl_n(x)}{x} \Rightarrow \log(l_n(x)) = -n\log x + c \Rightarrow l_n(x) = e^c x^{-n}.$$

Note that $e^c = l_n(1) = \lambda_n = 1$. Thus,

$$l_n(x) = \frac{1}{x^n}.$$

Table 1: Computed G^* versus simulated G^* under $N_a = 20000$ by averaging over 2000 trials.

Parameters	Simulated G^*	Computed G^* from Theorem 3
n = 5, k =2	0.737	0.7388
n = 5, k =3	0.582	0.5840
n = 6, k =2	0.724	0.7253
n = 6, k =3	0.669	0.6699
n = 8, k =2	0.659	0.6602
n = 8, k =5	0.545	0.5458
n = 12, k =4	0.636	0.6372
n = 12, k =10	0.266	0.2664
n = 25, k =4	0.459	0.4595

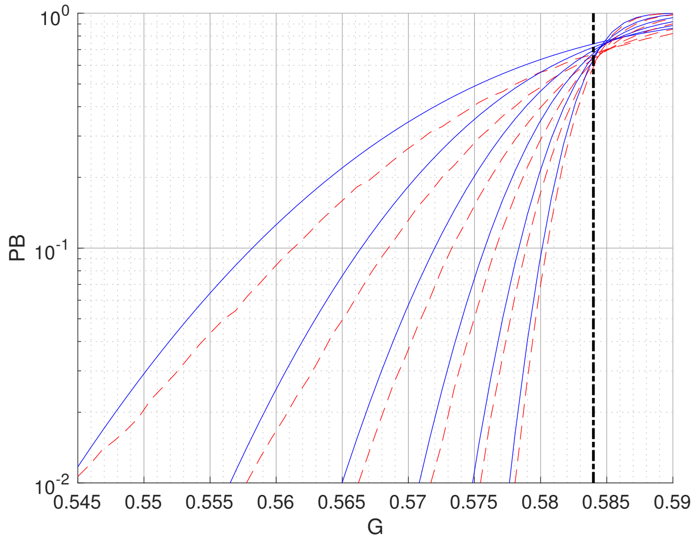


Figure 3: The probability of B-error P_B in terms of G for $N_a = 1000, 2000, 4000, 8000, 16000, 32000$. All curves are obtained using the parameters $n = 5$, $k = 3$. The solid and dashed lines correspond to results of computations and simulations respectively. The vertical dashed line shows $G^* = 0.5840$. Furthermore, the scaling parameters are $\alpha = 0.42362$ and $\beta = 0.8629$.

Now, for $i \in \{n-k+1, \ldots, n-1\}$, we have

$$\begin{aligned}\frac{dl_i(x)}{dx} = i \cdot \frac{l_{i+1}(x) - l_i(x)}{x} &\Rightarrow x^i\left(\frac{dl_i(x)}{dx} + \frac{il_i(x)}{x}\right) = ix^{i-1}l_{i+1}(x)\\ &\Rightarrow (x^i l_i(x))' = ix^{i-1}l_{i+1}(x)\\ &\Rightarrow x^i l_i(x) = \int_1^x iy^{i-1}l_{i+1}(y)dy + c_i\end{aligned}$$

Also for $i \in \{n-k+1, \ldots, n-1\}$, $c_i = l_i(1) = \lambda_i = 0$. Hence,

$$l_i(x) = \frac{i}{x^i}\int_1^x y^{i-1}l_{i+1}(y)dy.$$

□

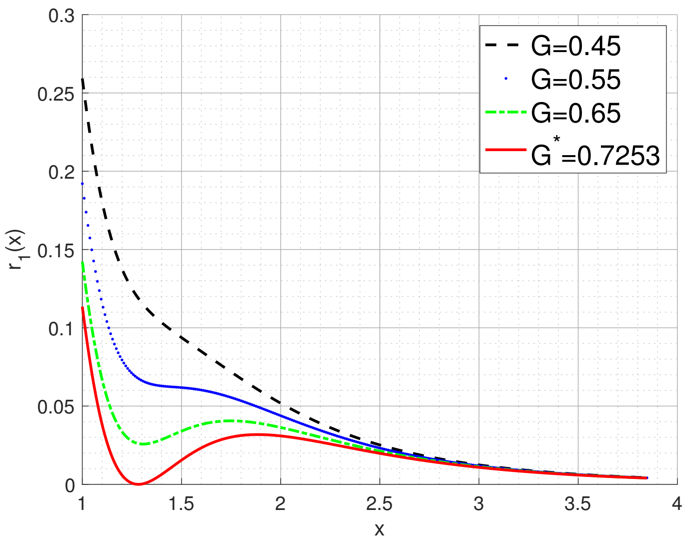


Figure 4: Illustrating the behavior of the solution of the density evolution for r_1 with respect to increasing G under the setting $n = 6$, $k = 2$.

Proof of Theorem 1. We prove the statement using backward induction. If $i = n$, from (21), we have $l_n(x) = 1/x^n$, which gives $\alpha_n^{(n)} = 1$. Now, for $i \in \{n-k+1, \cdots, n-1\}$, suppose $l_{i+1}(x) = \sum_{j=1}^{n} \alpha_j^{(i+1)}/x^j$ for some given numbers $\alpha_j^{(i+1)}$. Then, replacing it into (21) implies

$$\begin{aligned} l_i(x) =& \frac{i}{x^i} \int_1^x y^{i-1} \sum_{j=i+1}^{n} \frac{\alpha_j^{(i+1)}}{y^j} dy \\ =& \frac{i}{x^i} \sum_{j=i+1}^{n} \alpha_j^{(i+1)} \int_1^x y^{i-j-1} dy \\ =& \frac{i}{x^i} \sum_{j=i+1}^{n} \alpha_j^{(i+1)} \left(\frac{x^{i-j}}{i-j} - \frac{1}{i-j} \right) \\ =& \left(\sum_{j=i+1}^{n} \frac{i\alpha_j^{(i+1)}}{j-i} \right) \frac{1}{x^i} + \sum_{j=i+1}^{n} \frac{-i\alpha_j^{(i+1)}}{j-i} \cdot \frac{1}{x^j}. \end{aligned}$$

Now, by defining

$$\begin{cases} \alpha_i^{(i)} := \sum_{j=i+1}^{n} \dfrac{i\alpha_j^{(i+1)}}{j-i}, \\ \alpha_j^{(i)} := \dfrac{-i\alpha_j^{(i+1)}}{j-i}, \quad j \in \{i+1, \ldots, n\}, \end{cases} \tag{38}$$

it suffices to prove that (38) is equivalent to (21). From (38), for $j \in \{i+1, \cdots, n\}$ and $i \in \{n-k+1, \cdots, n-1\}$, we have

$$\alpha_j^{(i)} = \frac{-i\alpha_j^{(i+1)}}{j-i} = \frac{(-1)^2 i(i+1)}{(j-i)(j-i-1)} \alpha_j^{(i+2)} = \cdots = \frac{(-1)^{j-i} i(i+1) \cdots (j-1)}{(j-i)(j-i-1) \cdots 1} \alpha_j^{(j)} = (-1)^{j-i} \binom{j-1}{i-1} \alpha_j^{(j)}.$$

Thus, (38) is equivalent to

$$\begin{cases} \alpha_i^{(i)} = \sum_{j=i+1}^{n} \frac{i\alpha_j^{(i+1)}}{j-i}, & i \in \{n-k+1, \cdots, n-1\}, \\ \alpha_j^{(i)} = (-1)^{j-i}\binom{j-1}{i-1}\alpha_j^{(j)}, & j \in \{i+1, \cdots, n\}. \end{cases} \tag{39}$$

Now, from (39), we have

$$\begin{aligned} \alpha_i^{(i)} &= \sum_{j=i+1}^{n} \frac{i\alpha_j^{(i+1)}}{j-i} \\ &= \sum_{j=i+1}^{n} \frac{i}{j-i}(-1)^{j-i-1}\binom{j-1}{i}\alpha_j^{(j)} \\ &= \sum_{j=i+1}^{n} (-1)^{j-i-1}\binom{j-1}{i-1}\alpha_j^{(j)}, \end{aligned}$$

which implies that (39) is equivalent to

$$\begin{cases} \alpha_i^{(i)} = \sum_{j=i+1}^{n} (-1)^{j-i+1}\binom{j-1}{i-1}\alpha_j^{(j)}, & i \in \{n-k+1, \cdots, n-1\}, \\ \alpha_j^{(i)} = (-1)^{j-i}\binom{j-1}{i-1}\alpha_j^{(j)}, & j \in \{i+1, \cdots, n\}. \end{cases}$$

Also, as discussed earlier $\alpha_n^{(n)} = 1$. Hence, (38) is equivalent to (21). □

A.2 Proofs for Evolution of Slice Nodes

Proof of Lemma 1. Note that in K_0, each user node is connected to n slice nodes which are chosen uniformly at random. When $N_a \to \infty$, this choosing scheme is equivalent to the situation where each edge is choosing a slice node uniformly at random i.e., with probability $1/kM$. Moreover, in this way, possible range of j will become $j \in \{0, \ldots, nN_a\}$ but as $N_a \to \infty$, these two schemes are equivalent. Consider P_j as defined in (4). As stated earlier in problem statement, we remove all zero degree slice nodes prior to decoding. Let P'_j defined in a same way as P_j under consideration of these zero degree slice nodes. Then, for $j \in \{0, \ldots, nN_a\}$, we can write

$$P'_j = \binom{nN_a}{j}\left(\frac{1}{kM}\right)^j\left(1 - \frac{1}{kM}\right)^{nN_a - j}.$$

Now, in order to compute P_j, we must disregard zero degree slice nodes which results in

$$P_j = \frac{P'_j}{1 - P'_0}.$$

We have $\frac{N_a}{M} = G$, where G is a given constant and thus, as $N_a \to \infty$, we have $M \to \infty$. Also note that $\frac{nN_a}{kM} = \frac{G}{R}$. Therefore, as $N_a \to \infty$, we have

$$P'_j = \lim_{N_a \to \infty} \binom{nN_a}{j} \left(\frac{1}{kM}\right)^j \left(1 - \frac{1}{kM}\right)^{nN_a - j} = \frac{1}{j} \left(\frac{G}{R}\right)^j e^{-\frac{G}{R}}.$$

As a result, for $j = 0$

$$P'_0 = e^{-\frac{G}{R}}.$$

Therefore,

$$P_j = \frac{1}{j} \left(\frac{G}{R}\right)^j \frac{e^{-\frac{G}{R}}}{1 - e^{-\frac{G}{R}}}.$$

Hence, from (7), we have

$$\rho_j = \frac{jP_j}{\sum_j jP_j} = \frac{1}{(j-1)!} \left(\frac{G}{R}\right)^{j-1} e^{-\frac{G}{R}}.$$

□

Proof of Lemma 3. Using (17) and (22), we can write

$$\begin{aligned}
e(x) &= \sum_{i=n-k+1}^{n} l_i(x) \\
&= \sum_{i=n-k+1}^{n} \sum_{j=i}^{n} \frac{\alpha_j^{(i)}}{x^j} \\
&= \sum_{i=n-k+1}^{n} \sum_{j=i}^{n} \frac{(-1)^{j-i}}{x^j} \binom{j-1}{i-1} \alpha_j^{(j)} \\
&= \sum_{j=n-k+1}^{n} \sum_{i=n-k+1}^{j} \frac{(-1)^{j-i}}{x^j} \binom{j-1}{i-1} \alpha_j^{(j)} \\
&= \sum_{j=n-k+1}^{n} \frac{(-1)^j}{x^j} \alpha_j^{(j)} \sum_{i=n-k+1}^{j} (-1)^i \binom{j-1}{i-1} \\
&= \sum_{j=n-k+1}^{n} \frac{\beta_j}{x^j}.
\end{aligned}$$

□

Proof of Lemma 4. Using (10), the change of variable described in (14), and (17) imply

$$\frac{a(x)-1}{x} = \frac{1}{x} \left(\frac{(n-k+1)l_{n-k+1}(x) + \sum_{i=n-k+2}^{n} l_i(x)}{e(x)} - 1 \right) = \frac{(n-k)l_{n-k+1}(x)}{xe(x)}. \tag{40}$$

Now, replacing values from (22) and (24) results in

$$\frac{a(x)-1}{x} = (n-k) \frac{\sum_{j=n-k+1}^{n} x^{-j} \alpha_j^{(n-k+1)}}{x \sum_{j=n-k+1}^{n} x^{-j} \beta_j} = (n-k) \frac{\sum_{j=0}^{k-1} \alpha_{n-j}^{(n-k+1)} x^j}{\sum_{j=0}^{k-1} \beta_{n-j} x^{j+1}}.$$

Note that from (26), we can simply conclude that

$$\frac{a(x)-1}{x} = \frac{\lambda'(x)}{\lambda(x)} \Rightarrow \log(\lambda(x)) = \int_1^x \frac{a(y)-1}{y} dy + c$$
$$\Rightarrow \lambda(x) = e^c \exp\left(\int_1^x \frac{a(y)-1}{y} dy\right).$$

Also, $e^c = \lambda(1) = 1$. Hence,

$$\lambda(x) = \exp\left((n-k)\int_1^x \frac{\sum_{j=0}^{k-1} \alpha_{n-j}^{(n-k+1)} y^j}{\sum_{j=0}^{k-1} \beta_{n-j} y^{j+1}} dy\right).$$

□

Proof of Lemma 5. Replacing (26) into (16) gives

$$\frac{dr_j(x)}{dx} = j(r_{j+1}(x) - r_j(x))\frac{\lambda'(x)}{\lambda(x)}, \qquad j \geq 2.$$

Multiply both sides by $\lambda^j(x)$ to get

$$\lambda^j(x)\frac{dr_j(x)}{dx} + j\lambda^j(x) r_j(x)\frac{\lambda'(x)}{\lambda(x)} = jr_{j+1}(x)\lambda^j(x)\frac{\lambda'(x)}{\lambda(x)}$$
$$\Rightarrow \frac{d}{dx}\left(\lambda^j(x) r_j(x)\right) = jr_{j+1}(x)\lambda^j(x)\frac{\lambda'(x)}{\lambda(x)}.$$

Taking integral from both sides gives

$$r_j(x) = \frac{1}{\lambda^j(x)}\left(j\int_1^x r_{i+1}(y)\lambda^j(y)\frac{\lambda'(y)}{\lambda(y)} dy + c_j\right).$$

Note that $c_j = r_j(1)\cdot\lambda^j(1) = \rho_j \cdot 1 = \rho_j$. Hence

$$r_j(x) = \frac{1}{\lambda^j(x)}\left(j\int_1^x r_{j+1}(y)\lambda^j(y)\frac{\lambda'(y)}{\lambda(y)} dy + \rho_j\right).$$

□

Proof of Theorem 2. For $j \geq 2$, it suffices to prove that $r_j(x)$ given by (29) satisfies (28). By replacing $r_{j+1}(x)$ from (29) into the right hand side of (28) and using (20), we get $r_j(x)$ defined in (29), as follows:

$$\begin{aligned}
&\frac{1}{\lambda^j(x)}\left(j\int_1^x r_{j+1}(y)\lambda^j(y)\frac{\lambda'(y)}{\lambda(y)} dy + \rho_j\right) \\
=&\frac{1}{\lambda^j(x)}\left(j\int_1^x \frac{1}{j!\lambda^{j+1}(y)}\left(\frac{G}{R}\right)^j \exp\left(-\frac{G}{R\lambda(y)}\right)\lambda^j(y)\frac{\lambda'(y)}{\lambda(y)} dy + \rho_j\right) \\
=&\frac{1}{\lambda^j(x)}\left(\frac{1}{(j-1)!}\left(\frac{G}{R}\right)^j \int_1^x \exp\left(-\frac{G}{R\lambda(y)}\right)\frac{\lambda'(y)}{\lambda^2(y)} dy + \rho_j\right) \\
=&\frac{1}{\lambda^j(x)}\left(\frac{1}{(j-1)!}\left(\frac{G}{R}\right)^{j-1}\left[\exp\left(-\frac{G}{R\lambda(x)}\right) - \exp\left(-\frac{G}{R\lambda(1)}\right)\right] + \rho_j\right) \\
=&\frac{1}{\lambda^j(x)}\left(\frac{1}{(j-1)!}\left(\frac{G}{R}\right)^{j-1}\exp\left(-\frac{G}{R\lambda(x)}\right) - \rho_j + \rho_j\right) \\
=&r_j(x),
\end{aligned}$$

Therefore, for $j \geq 2$, $r_j(x)$ given by (29) satisfies (28). Now, for $r_1(x)$, substituting (29) into (18) implies

$$\begin{aligned}
r_1(x) &= e(x) - \sum_{j \geq 2} r_j(x) \\
&= e(x) - \sum_{j \geq 2} \frac{1}{\lambda^j(x)} \left(\frac{1}{(j-1)!} \left(\frac{G}{R} \right)^{j-1} \exp\left(-\frac{G}{R\lambda(x)} \right) \right) \\
&= e(x) - \frac{1}{\lambda(x)} \exp\left(-\frac{G}{R\lambda(x)} \right) \sum_{j \geq 2} \frac{1}{(j-1)!} \left(\frac{G}{R\lambda(x)} \right)^{j-1} \\
&= e(x) - \frac{1}{\lambda(x)} \exp\left(-\frac{G}{R\lambda(x)} \right) \left(\exp\left(\frac{G}{R\lambda(x)} \right) - 1 \right) \\
&= e(x) - \frac{1}{\lambda(x)} \left(1 - \exp\left(-\frac{G}{R\lambda(x)} \right) \right).
\end{aligned}$$

Then, replacing $e(x)$ from (24) results in

$$r_1(x) = \left[\sum_{j=n-k+1}^{n} \frac{\beta_j}{x^j} \right] - \frac{1}{\lambda(x)} \left(1 - \exp\left(-\frac{G}{R\lambda(x)} \right) \right)$$

□

A.3 Proofs for G^*

Proof of Theorem 3. As discussed earlier in the paper, we have

$$\left. \frac{d\tilde{G}(x)}{dx} \right|_{x=x^*} = 0.$$

Taking derivative of $\tilde{G}$ gives

$$\frac{d\tilde{G}(x)}{dx} = -R \left[\lambda'(x) \log(1 - e(x)\lambda(x)) - \lambda(x) \frac{e'(x)\lambda(x) + e(x)\lambda'(x)}{1 - e(x)\lambda(x)} \right]. \tag{41}$$

Now, (26) and (40) together imply

$$\frac{\lambda'(x)}{\lambda(x)} = \frac{a(x) - 1}{x} = \frac{(n-k)l_{n-k+1}(x)}{xe(x)}.$$

Therefore,

$$\lambda'(x) = \lambda(x) \frac{(n-k)l_{n-k+1}(x)}{xe(x)}. \tag{42}$$

Replacing (42) into (41) results in

$$\frac{d\tilde{G}(x)}{dx} = -R\lambda^2(x)x \left[\frac{(n-k)l_{n-k+1}(x)}{h(x)} \log(1 - h(x)) - \frac{xe'(x) + (n-k)l_{n-k+1}(x)}{1 - h(x)} \right],$$

where $h(x) = e(x)\lambda(x)$. Now, noting that we have $x > 0$ and $\lambda(x) > 0$, the following can be concluded:

$$\left. \frac{d\tilde{G}(x)}{dx} \right|_{x=x^*} = 0 \Rightarrow \log(1 - h(x^*)) = \frac{1 - h(x^*)}{h(x^*)} \left(1 + \frac{x^* e'(x^*)}{(n-k)l_{n-k+1}(x^*)} \right).$$

□

A.4 Proofs for Scaling Law

Proof of Theorem 4. For notation simplicity, we omit the argument x from all functions, particularly, we considered $e = e(x)$. Note that local drift $\hat{f}^{(z_i)}$ represents the expected change of the corresponding edge-based quantity of z_i and local covariance $\hat{f}^{(z_i z_j)}$ represents the covariance between the corresponding edge-based quantities of z_i and z_j. Obtaining these local drift and local covariance only need some straightforward probabilistic evaluations of the problem. We include all the computations without detail explanations. We also include computations of the derivatives of the resulted values which are needed to form (36) as well as (37). First, for $i \in \{n-k+1, \dots, n\}$, define

$$w_i(u_1, u_2) = \binom{i-1}{u_1-1, u_2} q_1^{u_1-1} q_2 (1-q_1-q_2)^{i-u_1-u_2},$$

where we used the following notation:

$$\binom{m}{r, s} := \frac{m!}{r!s!}.$$

Note that

$$\sum_{u_1,u_2} w_i(u_1, u_2) = \sum_{u_1,u_2} \binom{i-1}{u_1-1, u_2} q_1^{u_1-1} q_2^{u_2} (1-q_1-q_2)^{i-u_1-u_2} = 1.$$

Also define

$$W_i(x, y) := \sum_{u_1,u_2} w_i(u_1, u_2) x^{u_1} y^{u_2}.$$

Then we can write

$$\begin{aligned} W_i(x,y) &= x \sum_{u_1,u_2} \binom{i-1}{u_1-1, u_2} (xq_1)^{u_1-1} (yq_2)^{u_2} (1-q_1-q_2)^{i-u_1-u_2} \\ &= x(xq_1 + yq_2 + 1 - q_1 - q_2)^{i-1}. \end{aligned}$$

Now, consider the following computations:

$$\begin{aligned} \sum_{u_1,u_2} w_i(u_1,u_2)u_1 &= \left.\frac{\partial W_i(x,y)}{\partial x}\right|_{x=y=1} \\ &= (xq_1 + yq_2 + 1 - q_1 - q_2)^{i-1} + xq_1(i-1)(xq_1 + yq_2 + 1 - q_1 - q_2)^{i-2}\Big|_{x=y=1} \\ &= 1 + q_1(i-1), \end{aligned}$$

$$\begin{aligned} \sum_{u_1,u_2} w_i(u_1,u_2)u_2 &= \left.\frac{\partial W_i(x,y)}{\partial y}\right|_{x=y=1} \\ &= xq_2(i-1)(xq_1 + yq_2 + 1 - q_1 - q_2)^{i-2}\Big|_{x=y=1} \\ &= q_2(i-1), \end{aligned}$$

$$\begin{aligned}
\sum_{u_1,u_2} w_i(u_1,u_2)u_1^2 &= \sum_{u_1,u_2} w_i(u_1,u_2)u_1(u_1-1) + \sum_{u_1,u_2} w_i(u_1,u_2)u_1 \\
&= \frac{\partial^2 W_i(x,y)}{\partial x^2}\bigg|_{x=y=1} + (1+q_1(i-1)) \\
&= q_1(i-1)(xq_1+yq_2+1-q_1-q_2)^{i-2} + q_1(i-1)(xq_1+yq_2+1-q_1-q_2)^{i-2} \\
&+ xq_1^2(i-1)(i-2)(xq_1+yq_2+1-q_1-q_2)^{i-3}\bigg|_{x=y=1} + (1+q_1(i-1)) \\
&= q_1(i-1)(2+q_1(i-2)) + 1 + q_1(i-1) \\
&= q_1(i-1)(3+q_1(i-2)) + 1,
\end{aligned}$$

$$\begin{aligned}
\sum_{u_1,u_2} w_i(u_1,u_2)u_2^2 &= \sum_{u_1,u_2} w_i(u_1,u_2)u_2(u_2-1) + \sum_{u_1,u_2} w_i(u_1,u_2)u_2 \\
&= \frac{\partial^2 W_i(x,y)}{\partial y^2}\bigg|_{x=y=1} + q_2(i-1) \\
&= xq_2^2(i-1)(i-2)(xq_1+yq_2+1-q_1-q_2)^{i-3}\bigg|_{x=y=1} + q_2(i-1) \\
&= q_2^2(i-1)(i-2) + q_2(i-1) \\
&= q_2(i-1)(q_2(i-2)+1),
\end{aligned}$$

$$\begin{aligned}
\sum_{u_1,u_2} w_i(u_1,u_2)u_1u_2 &= \frac{\partial^2 W_i(x,y)}{\partial x \partial y}\bigg|_{x=y=1} \\
&= q_2(i-1)(xq_1+yq_2+1-q_1-q_2)^{i-2} \\
&+ xq_1q_2(i-1)(i-2)(xq_1+yq_2+1-q_1-q_2)^{i-3}\bigg|_{x=y=1} \\
&= q_2(i-1) + q_1q_2(i-1)(i-2).
\end{aligned}$$

Using these computaions, we can now calculate the components appeared in the differential equations of covarianve evolution as follows:

$$\begin{aligned}
\hat{f}^{(r_1)} &= \frac{l_{n-k+1}}{e}\sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(u_2-u_1) - \sum_{i=n-k+2}^{n}\frac{l_i}{e} \\
&= \frac{l_{n-k+1}}{e}\left((n-k)q_2-(n-k)q_1-1\right) - \sum_{i=n-k+2}^{n}\frac{l_i}{e} \\
&= (n-k)\frac{l_{n-k+1}}{e^2}(r_2-r_1) - 1,
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(r_2)} &= \frac{l_{n-k+1}}{e}\sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(-2u_2) \\
&= -2(n-k)\frac{l_{n-k+1}}{e}q_2 \\
&= -2(n-k)\frac{l_{n-k+1}}{e^2}r_2,
\end{aligned}$$

$$\hat{f}^{(l_i)} = \frac{i}{e}\left(l_{i+1} - l_i\right), \qquad i \in \{n-k+1, \ldots, n\},$$

$$\hat{f}^{(l_n)} = -\frac{nl_n}{e},$$

Therfore, for derivatives of these values which apeare in covariance evolution, we have

$$\begin{aligned}
\frac{\partial \hat{f}^{(r_1)}}{\partial r_1} &= -(n-k)\frac{l_{n-k+1}}{e^2},\\
\frac{\partial \hat{f}^{(r_1)}}{\partial r_2} &= (n-k)\frac{l_{n-k+1}}{e^2},\\
\frac{\partial \hat{f}^{(r_1)}}{\partial l_{n-k+1}} &= (n-k)(r_2-r_1)\frac{e-2l_{n-k+1}}{e^3},\\
\frac{\partial \hat{f}^{(r_1)}}{\partial l_i} &= -2(n-k)(r_2-r_1)\frac{l_{n-k+1}}{e^3}, \qquad i \in \{n-k+2, \ldots, n\},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \hat{f}^{(r_2)}}{\partial r_1} &= 0,\\
\frac{\partial \hat{f}^{(r_2)}}{\partial r_2} &= -2(n-k)\frac{l_{n-k+1}}{e^2},\\
\frac{\partial \hat{f}^{(r_2)}}{\partial l_{n-k+1}} &= -2(n-k)\frac{e-2l_{n-k+1}}{e^3}r_2,\\
\frac{\partial \hat{f}^{(r_2)}}{\partial l_i} &= 4(n-k)\frac{l_{n-k+1}}{e^3}r_2, \qquad i \in \{n-k+2, \ldots, n\},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \hat{f}^{(l_i)}}{\partial r_1} &= 0, \qquad i \in \{n-k+1, \ldots, n\},\\
\frac{\partial \hat{f}^{(l_i)}}{\partial r_2} &= 0, \qquad i \in \{n-k+1, \ldots, n\},\\
\frac{\partial \hat{f}^{(l_i)}}{\partial l_i} &= -\frac{i}{e^2}\left(l_{i+1} - l_i + e\right), \qquad i \in \{n-k+1, \ldots, n\},\\
\frac{\partial \hat{f}^{(l_i)}}{\partial l_{i+1}} &= -\frac{i}{e^2}\left(l_{i+1} - l_i - e\right), \qquad i \in \{n-k+1, \ldots, n\},\\
\frac{\partial \hat{f}^{(l_i)}}{\partial l_j} &= -\frac{i}{e^2}\left(l_{i+1} - l_i\right), \qquad i \in \{n-k+1, \ldots, n\}, j \notin \{i, i+1\},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \hat{f}^{(l_n)}}{\partial r_1} &= 0,\\
\frac{\partial \hat{f}^{(l_n)}}{\partial r_2} &= 0,
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial \hat{f}^{(l_n)}}{\partial l_i} &= \frac{n l_n}{e^2}, \qquad i \in \{n-k+1, \dots, n-1\}, \\
\frac{\partial \hat{f}^{(l_n)}}{\partial l_n} &= -\frac{n}{e^2}(e - l_n).
\end{aligned}
$$

Based on these computaions of $\hat{f}^{z_i}$, $\hat{f}^{z_i z_j}$ can be computed as follows:

$$
\begin{aligned}
\hat{f}^{(r_1 r_1)} + \left(\hat{f}^{(r_1)}\right)^2 &= \frac{l_{n-k+1}}{e} \sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(u_2-u_1)^2 + \frac{l_{n-k+2}}{e}(-1)^2 + \cdots + \frac{l_n}{e}(-1)^2 \\
&= \frac{l_{n-k+1}}{e} \sum_{u_1,u_2} (u_1,u_2)\left(u_1^2 + u_2^2 - 2u_1u_2\right) + \sum_{i=n-k+2}^{n} \frac{l_i}{e} \\
&= \frac{l_{n-k+1}}{e}(q_1(n-k)(3+(n-k+1)q_1) + 1 + q_2(n-k)(q_2(n-k-1)+1) \\
&\quad - 2q_2(n-k) + q_1q_2(n-k)(n-k-1)) + \sum_{i=n-k+1}^{n} \frac{l_i}{e} \\
&= (n-k)\frac{l_{n-k+1}}{e}((n-k-1)(q_1-q_2)^2 + 3q_1 - q_2) + 1 \\
&= (n-k)\frac{l_{n-k+1}}{e^3}((n-k-1)(r_1-r_2)^2 + (3r_1-r_2)e) + 1, \\
\Rightarrow \hat{f}^{(r_1 r_1)} &= (n-k)\frac{l_{n-k+1}}{e^3}((n-k-1)(r_1-r_2)^2 + (3r_1-r_2)e) + 1 - \left(\hat{f}^{(r_1)}\right)^2,
\end{aligned}
$$

$$
\begin{aligned}
\hat{f}^{(r_1 r_2)} + \hat{f}^{(r_1)}\hat{f}^{(r_2)} &= \frac{l_{n-k+1}}{e} \sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(u_2-u_1)(-2u_2) \\
&= \frac{l_{n-k+1}}{e} \sum_{u_1,u_2} (u_1,u_2)(-2u_2^2 + 2u_1u_2) \\
&= \frac{l_{n-k+1}}{e}(-2q_2(n-k)(q_2(n-k+1)+1) + 2(q_2(n-k) + q_1q_2(n-k)(n-k-1))) \\
&= 2(n-k)(n-k-1)\frac{l_{n-k+1}}{e} q_2(q_1-q_2) \\
&= 2(n-k)(n-k-1)\frac{l_{n-k+1}}{e^3} r_2(r_1-r_2), \\
\Rightarrow \hat{f}^{(r_1 r_2)} &= 2(n-k)(n-k-1)\frac{l_{n-k+1}}{e^3} r_2(r_1-r_2) - \hat{f}^{(r_1)}\hat{f}^{(r_2)},
\end{aligned}
$$

$$
\begin{aligned}
\hat{f}^{(r_1 l_{n-k+1})} + \hat{f}^{(r_1)}\hat{f}^{(l_{n-k+1})} &= \frac{l_{n-k+1}}{e} \sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(u_2-u_1)(-1)(n-k+1) \\
&\quad + \frac{l_{n-k+2}}{e} \sum_{u_1,u_2} w_{n-k+2}(u_1,u_2)(-1)(n-k+1) \\
&= -(n-k+1)\frac{l_{n-k+1}}{e}((n-k)q_2 - (n-k)q_1 - 1) - (n-k+1)\frac{l_{n-k+2}}{e} \\
&= -\frac{n-k+1}{e}(l_{n-k+1}((n-k)(q_2-q_1) - 1) + l_{n-k+2}) \\
&= -\frac{n-k+1}{e^2}(l_{n-k+1}((n-k)(r_2-r_1) - e) + e l_{n-k+2})
\end{aligned}
$$

$$\Rightarrow \hat{f}^{(r_1 l_{n-k+1})} = -\frac{n-k+1}{e^2}\left(l_{n-k+1}((n-k)(r_2-r_1)-e)+el_{n-k+2}\right) - \hat{f}^{(r_1)}\hat{f}^{(l_{n-k+1})},$$

$$\begin{aligned}
\hat{f}^{(r_1 l_i)} + \hat{f}^{(r_1)}\hat{f}^{(l_i)} &= \frac{l_i}{e}\sum_{u_1,u_2} w_i(u_1,u_2)(-1)(-i) + \frac{l_{i+1}}{e}\sum_{u_1,u_2} w_{i+1}(u_1,u_2)(-1)i \\
&= -\frac{i}{e}(l_{i+1}-l_i), \qquad i \in \{n-k+2,\dots,n-1\}, \\
\Rightarrow \hat{f}^{(r_1 l_i)} &= -\frac{i}{e}(l_{i+1}-l_i) - \hat{f}^{(r_1)}\hat{f}^{(l_i)}, \qquad i \in \{n-k+2,\dots,n-1\},
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(r_1 l_n)} + \hat{f}^{(r_1)}\hat{f}^{(l_n)} &= \frac{l_n}{e}\sum_{u_1,u_2} w_n(u_1,u_2)(-1)(-n) \\
\Rightarrow \hat{f}^{(r_1 l_n)} &= \frac{nl_n}{e} - \hat{f}^{(r_1)}\hat{f}^{(l_n)},
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(r_2 r_2)} + \left(\hat{f}^{(r_2)}\right)^2 &= \frac{l_{n-k+1}}{e}\sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(-2u_2)^2 \\
&= \frac{4l_{n-k+1}}{e}q_2(n-k)(1+(n-k-1)q_2) \\
&= 4(n-k)\frac{l_{n-k+1}}{e^3}r_2(e+(n-k-1)r_2), \\
\Rightarrow \hat{f}^{(r_2 r_2)} &= 4(n-k)\frac{l_{n-k+1}}{e^3}r_2(e+(n-k-1)r_2) - (\hat{f}^{(r_2)})^2,
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(r_2 l_{n-k+1})} + \hat{f}^{(r_2)}\hat{f}^{(l_{n-k+1})} &= \frac{l_{n-k+1}}{e}\sum_{u_1,u_2} w_{n-k+1}(u_1,u_2)(-2u_2)(-1)(n-k+1) \\
&= 2(n-k+1)\frac{l_{n-k+1}}{e}(n-k)q_2 \\
&= 2(n-k)(n-k+1)\frac{l_{n-k+1}}{e^2}r_2 \\
\Rightarrow \hat{f}^{(r_2 l_{n-k+1})} &= 2(n-k)(n-k+1)\frac{l_{n-k+1}}{e^2}r_2 - \hat{f}^{(r_2)}\hat{f}^{(l_{n-k+1})},
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(r_2 l_i)} + \hat{f}^{(r_2)}\hat{f}^{(l_i)} &= 0, \qquad i \in \{n-k+2,\dots,n\}, \\
\Rightarrow \hat{f}^{(r_2 l_i)} &= -\hat{f}^{(r_2)}\hat{f}^{(l_i)}, \qquad i \in \{n-k+2,\dots,n\},
\end{aligned}$$

$$\begin{aligned}
\hat{f}^{(l_i l_i)} + \left(\hat{f}^{(l_i)}\right)^2 &= (-i)^2\frac{l_i}{e} + i^2\frac{l_{i+1}}{e}, \qquad i \in \{n-k+1,\dots,n-1\}, \\
\Rightarrow \hat{f}^{(l_i l_i)} &= \frac{i^2}{e}(l_i + l_{i+1}) - \left(\hat{f}^{(l_i)}\right)^2, \qquad i \in \{n-k+1,\dots,n-1\},
\end{aligned}$$

$$\hat{f}^{(l_i l_{i+1})} + \hat{f}^{(l_i)}\hat{f}^{(l_{i+1})} = -i(i+1)\frac{l_{i+1}}{e}, \qquad i \in \{n-k+1,\dots,n-1\},$$

$$\Rightarrow \hat{f}^{(l_i l_{i+1})} = -i(i+1)\frac{l_{i+1}}{e} - \hat{f}^{(l_i)}\hat{f}^{(l_{i+1})}, \qquad i \in \{n-k+1,\dots,n-1\},$$

$$\begin{aligned}\hat{f}^{(l_i l_j)} + \hat{f}^{(l_i)}\hat{f}^{(l_j)} &= 0, \qquad i \in \{n-k+1,\dots,n-1\},\\ \hat{f}^{(l_i l_j)} &= -\hat{f}^{(l_i)}\hat{f}^{(l_j)}, \qquad i \in \{n-k+1,\dots,n-2\}, j > i+1\end{aligned}$$

$$\begin{aligned}\hat{f}^{(l_n l_n)} + \left(\hat{f}^{(l_n)}\right)^2 &= n^2\frac{l_n}{e},\\ \Rightarrow \hat{f}^{(l_n l_n)} &= n^2\frac{l_n}{e} - \left(\hat{f}^{(l_n)}\right)^2.\end{aligned}$$

□

Proof of Lemma 6. Using the same idea as in the proof of Theoram 4, first define

$$d(u_1,u_2) = \binom{nN_a}{u_1,u_2}\rho_1^{u_1}\rho_2^{u_2}(1-\rho_1-\rho_2)^{nN_a-u_1-u_2}.$$

where we used the following notation:

$$\binom{m}{r,s} := \frac{m!}{r!s!}.$$

Note that

$$\sum_{u_1,u_2} d(u_1,u_2) = \sum_{u_1,u_2}\binom{nN_a}{u_1,u_2}\rho_1^{u_1}\rho_2^{u_2}(1-\rho_1-\rho_2)^{nN_a-u_1-u_2} = 1.$$

Also define

$$D(x,y) := \sum_{u_1,u_2} d(u_1,u_2)x^{u_1}y^{u_2} = (x\rho_1 + y\rho_2 + 1 - \rho_1 - \rho_2)^{nN_a}.$$

Now, consider the following computations:

$$\begin{aligned}\sum_{u_1,u_2} d(u_1,u_2)u_1 &= \left.\frac{\partial D(x,y)}{\partial x}\right|_{x=y=1}\\ &= \left.\rho_1 nN_a(x\rho_1 + y\rho_2 + 1 - \rho_1 - \rho_2)^{nN_a-1}\right|_{x=y=1}\\ &= \rho_1 nN_a,\end{aligned}$$

$$\begin{aligned}\sum_{u_1,u_2} d(u_1,u_2)u_2 &= \left.\frac{\partial D(x,y)}{\partial y}\right|_{x=y=1}\\ &= \left.\rho_2 nN_a(x\rho_1 + y\rho_2 + 1 - \rho_1 - \rho_2)^{nN_a-1}\right|_{x=y=1}\\ &= \rho_2 nN_a,\end{aligned}$$

$$\begin{aligned}\sum_{u_1,u_2} d(u_1,u_2)u_1^2 &= \sum_{u_1,u_2} d(u_1,u_2)u_1(u_1-1) + \sum_{u_1,u_2} d(u_1,u_2)u_1\\ &= \left.\frac{\partial^2 D(x,y)}{\partial x^2}\right|_{x=y=1} + \rho_1 nN_a\end{aligned}$$

$$
\begin{aligned}
&= \rho_1^2 nN_a(nN_a-1)(x\rho_1+y\rho_2+1-\rho_1-\rho_2)^{nN_a-2}\Big|_{x=y=1} + \rho_1 nN_a \\
&= \rho_1^2 nN_a(nN_a-1) + \rho_1 nN_a \\
&= \rho_1 nN_a(1+\rho_1(nN_a-1)),
\end{aligned}
$$

$$
\begin{aligned}
\sum_{u_1,u_2} d(u_1,u_2)u_2^2 &= \sum_{u_1,u_2} d(u_1,u_2)u_2(u_2-1) + \sum_{u_1,u_2} d(u_1,u_2)u_2 \\
&= \frac{\partial^2 D(x,y)}{\partial y^2}\Big|_{x=y=1} + \rho_2 nN_a \\
&= \rho_2^2 nN_a(nN_a-1)(x\rho_1+y\rho_2+1-\rho_1-\rho_2)^{nN_a-2}\Big|_{x=y=1} + \rho_2 nN_a \\
&= \rho_2^2 nN_a(nN_a-1) + \rho_2 nN_a \\
&= \rho_2 nN_a(1+\rho_2(nN_a-1)),
\end{aligned}
$$

$$
\begin{aligned}
\sum_{u_1,u_2} d(u_1,u_2)u_1u_2 &= \frac{\partial D(x,y)}{\partial x \partial y}\Big|_{x=y=1} \\
&= \rho_1 nN_a(nN_a-1)\rho_2(x\rho_1+y\rho_2+1-\rho_1-\rho_2)^{nN_a-2}\Big|_{x=y=1} \\
&= nN_a(nN_a-1)\rho_1\rho_2.
\end{aligned}
$$

Based on these computations, we can now calculate the initial conditions for $\delta^{(ij)}$. Firstly, note that for $i \in \{n-k+1,\dots,n\}$, $l_i(1)$ is deterministic. Thus,

$$
\delta^{(ij)}(1) = 0, \qquad \text{if } i \notin \{0,1\} \text{ or } j \notin \{0,1\}.
$$

For $i,j \in \{0,1\}$, initial conditions for $\delta^{(ij)}$ can be computed as follows:

$$
\begin{aligned}
N_a\delta^{(00)}(1) &= \sum_{u_1,u_2} d(u_1,u_2)u_1^2 - \left(\sum_{u_1,u_2} d(u_1,u_2)u_1\right)^2 \\
&= \rho_1 nN_a\left(1+\rho_1(nN_a-1)\right) - (\rho_1 nN_a)^2 \\
&= nN_a\rho_1(1-\rho_1), \\
\Rightarrow \delta^{(00)}(1) &= n\rho_1(1-\rho_1),
\end{aligned}
$$

$$
\begin{aligned}
N_a\delta^{(11)}(1) &= \sum_{u_1,u_2} d(u_1,u_2)u_2^2 - \left(\sum_{u_1,u_2} d(u_1,u_2)u_2\right)^2 \\
&= \rho_2 nN_a\left(1+\rho_2(nN_a-1)\right) - (\rho_2 nN_a)^2 \\
&= nN_a\rho_2(1-\rho_2), \\
\Rightarrow \delta^{(11)}(1) &= n\rho_2(1-\rho_2),
\end{aligned}
$$

$$
N_a\delta^{(01)}(1) = \sum_{u_1,u_2} d(u_1,u_2)u_1u_2 - \left(\sum_{u_1,u_2} d(u_1,u_2)u_1\right)\left(\sum_{u_1,u_2} d(u_1,u_2)u_2\right)
$$

$$
\begin{aligned}
&= nN_a(nN_a - 1)\rho_1\rho_2 - (\rho_1 nN_a)(\rho_2 nN_a) \\
&= -nN_a\rho_1\rho_2 \\
\Rightarrow \delta^{(01)}(1) &= -n\rho_1\rho_2.
\end{aligned}
$$

□

References

[1] N. Abramson, "The aloha system: another alternative for computer communications," in *1970 Fall Joint Computer Conf.*, Houston, Texas, USA, November 1970, pp. 281–285.

[2] L. G. Roberts, "Aloha packet system with and without slots and capture," *ACM SIGCOMM Computer Communication Review Newsletter*, vol. 5, no. 2, pp. 28–42, Apr. 1975.

[3] D. Bertsekas, D. Bertsekas, and R. Gallager, *Data Networks*, ser. Prentice-Hall international editions. Prentice-Hall, 1987.

[4] E. Casini, R. D. Gaudenzi, and O. D. R. Herrero, "Contention resolution diversity slotted aloha (crdsa): An enhanced random access schemefor satellite access packet networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1408–1419, April 2007.

[5] G. Liva, "Graph-based analysis and optimization of contention resolution diversity slotted aloha," *IEEE Transactions on Communications*, vol. 59, no. 2, pp. 477–487, February 2011.

[6] M. Berioli, G. Cocco, G. Liva, and A. Munari, "Modern random access protocols," *Foundations and Trends® in Networking*, vol. 10, no. 4, pp. 317–446, 2016.

[7] X. Chen and D. Guo, "Many-access channels: The gaussian case with random user activities," in *2014 IEEE International Symposium on Information Theory*, June 2014, pp. 3127–3131.

[8] X. Chen, T. Chen, and D. Guo, "Capacity of gaussian many-access channels," *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 3516–3539, June 2017.

[9] Y. Polyanskiy, "A perspective on massive random-access," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2523–2527.

[10] M. Effros, V. Kostina, and R. C. Yavas, "Random access channel coding in the finite blocklength regime," Jan. 2018, https://arxiv.org/abs/1801.09018.

[11] E. Paolini, G. Liva, and M. Chiani, "Coded slotted aloha: A graph-based method for uncoordinated multiple access," *IEEE Trans. Inform. Theory*, vol. 61.

[12] F. Lázaro and u. Stefanović, "Finite-length analysis of frameless aloha with multi-user detection," *IEEE Communications Letters*, vol. 21, no. 4, pp. 769–772, April 2017.

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

[14] M. Ivanov, F. Brännström, A. G. i Amat, and P. Popovski, "Broadcast coded slotted aloha: A finite frame length analysis," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 651–662, Feb 2017.

[15] E. Sandgren, A. G. i Amat, and F. Brännström, "On frame asynchronous coded slotted aloha: Asymptotic, finite length, and delay analysis," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 691–704, Feb 2017.

[16] A. Graell i Amat and G. Liva, "Finite length analysis of irregular repetition slotted aloha in the waterfall region," *IEEE Communications Letters*, vol. 22, pp. 886–889, 05 2018.

[17] A. Amraoui, "Asymptotic and finite-length optimization of ldpc codes," Ph.D. dissertation, EPFL, Lausanne, Switzerland, 2006, number 3558.

[18] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke, "Finite-length scaling for iteratively decoded LDPC ensembles," *IEEE Trans. Inform. Theory*, vol. 55, no. 2, pp. 473–498, Feb. 2009.

[19] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, Sep. 1981.

[20] N. C. Wormald, "Differential equations for random processes and random graphs," *The annals of applied probability*, pp. 1217–1235, 1995.

[21] ——, "The differential equation method for random graph processes and greedy algorithms," *Lectures on approximation and randomized algorithms*, vol. 73, p. 155, 1999.

[22] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. ACM, 1997, pp. 150–159.

[23] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[24] V. Privman, *Finite Size Scaling and Numerical Simulation of Statistical Systems*. World Scientific, Singapour, 1990.