LT-AF Codes: LT Codes with Alternating Feedback

Ali Talari and Nazanin Rahnavard
Oklahoma State University, Stillwater, OK 74078,
Emails: {ali.talari, nazanin.rahnavard}@okstate.edu

Abstract—LT codes are capacity achieving and flexible rateless codes that need a single-bit feedback to inform the encoder of the successful decoding. However, this weak feedback channel remains unused when the transmission is in progress. In addition, although LT codes are asymptotically capacity achieving, their performance significantly degrades for short block lengths. Consequently, we propose \textit{LT Codes with Alternating Feedback} (LT-AF Codes) that considerably improve the performance of LT codes for short-block lengths when 	extit{belief propagation decoder} is in use. In our proposed scheme, the decoder alternatively issues two types of feedbacks based on the dependencies of the still undecoded received output symbols and the number of decoded input symbols. We propose two methods to form the latter type of feedback with a trade-off in their complexity and performance.

I. INTRODUCTION

LT codes [1] are the first practical realization of Rateless codes [1–3] with capacity-achieving performance on erasure channels. LT codes are universal in the sense that they are simultaneously near optimal for every erasure channel with varying or unknown erasure rate \( \epsilon \in [0,1) \) [1]. These codes require only one feedback that is issued by the decoder (receiver) to inform the encoder (transmitter) of a successful LT decoding. However, in LT coding the available feedback channel remains under utilized during the transmission.

Further, as the data-block length decreases the performance of LT codes significantly deteriorates [1, 4, 5]. Therefore, existing work have proposed to employ feedbacks to inform the encoder of the number of successfully decoded input symbols (source packets) [6–8], a suitable input symbol to be sent for enhancing the decoding [9], or the index of some recovered input symbols [10]. In this paper, we take a step further and propose to alternatively generate two types of aforementioned feedbacks and propose \textit{LT codes with alternating feedbacks} (LT-AF codes). We propose two novel methods to analyze the decoders buffer and to request a suitable input symbol that greedily makes the highest progress in the decoding process as a feedback. In contrast to other existing work, we design LT-AF codes with a realistic feedback channel assumption, where the feedback channel can have unknown or varying erasure rate \( \epsilon_{fb} \in [0,1) \).

The paper is organized as follows. In Section II, we provide a brief review on LT codes. In Section III we review the existing rateless codes with feedback. In Section IV, we propose and analyze LT-AF codes. Section V reports the performance of LT-AF codes. Finally, Section VI concludes the paper.

II. REVIEW ON LT CODES

First, without loss of generality and for simplicity, let us assume that the input and output symbols are binary symbols, while they may contain thousands of bits. LT codes [1] have simple encoding and decoding procedures as follows.

\textbf{LT encoding:} In LT encoding of \( k \) input symbols, first an output symbol degree \( d \) is chosen from the Robust-Soliton (RS) degree distribution [1] \( \{ \mu_1, \mu_2, \ldots, \mu_k \} \), where \( \mu_i \) is the probability that \( d = i \) (also shown by its generator polynomial \( \mu(x) = \sum_{i=1}^{k} \mu_i x^i \)). Next, \( d \) input symbols are chosen uniformly at random from \( k \) input symbols and are XORed to generate an output symbol. We refer to the \( d \) contributing input symbols in forming an output symbol as its neighbors. This procedure can be potentially repeated infinite number of times; however, it is stopped upon the reception of the single-bit feedback. The RS distribution \( \mu(.) \) is obtained by combining the ideal-Soliton (IS) distribution \( \rho(.) \) and distribution \( \tau(.) \) given by

\begin{equation}
\rho(i) = \left\{ \begin{array}{ll}
\frac{1}{k} & i = 1, \\
\frac{1}{n(n+1)} & i = 2, \ldots, k,
\end{array} \right.
\end{equation}

and

\begin{equation}
\tau(i) = \left\{ \begin{array}{ll}
\frac{R}{\delta} \ln\left(\frac{\delta}{R}\right) & i = 1, \ldots, \frac{k}{R} - 1, \\
0 & i = \frac{k}{R} + 1, \ldots, k,
\end{array} \right.
\end{equation}

respectively, where \( R = c \ln \left( \frac{\delta}{R}\right) \sqrt{k} \), and \( \delta \) and \( c \) are two tuneable parameters [1]. It is easy to see that the average degree of output symbols with IS distribution is \( \sum_{i=1}^{k} i \rho(i) = \rho'(1) = H(k) \approx \ln k \), where \( \rho'(x) \) is the first derivative of \( \rho(x) \) with respect to its variable \( x \), and \( H(k) \) is the \( k^{th} \) Harmonic number [1]. Finally, RS degree distribution \( \mu(.) \) is obtained by

\begin{equation}
\mu(i) = \frac{\rho(i) + \tau(i)}{\beta}, i = 1, \ldots, k,
\end{equation}

where \( \beta = \sum_{i=1}^{k} \rho(i) + \tau(i) \).

\textbf{LT decoding:} Rateless decoding is iteratively performed upon arrival of a new output symbols. The decoder finds an output symbol such that the value of all but one of its neighboring input symbols is known. It recovers the value of the unknown input symbol by bitwise XOR operations and removes all the edges incident to that output symbol. This process is repeated until no such an output symbol exists. The set of output symbols that are reduced to degree one is called the ripple. If the ripple becomes empty the decoding stops and waits for new output symbols of degree one to proceed the decoding. If all \( k \) input symbols are recovered, it issues a single-bit feedback indicating success of the decoding.

As shown in [1] the only condition for a successful LT decoding is the delivery of a certain number of output
symbols since they are statistically independent. Let \( \gamma_{\text{suc}} \) be the required coding overhead to have a successful decoding with high probability (w.h.p.), i.e., \( \gamma_{\text{suc}} \times k \) coded symbols are enough to decode \( k \) input symbols w.h.p. Further, let \( \gamma, 0 \leq \gamma \leq \gamma_{\text{suc}} \), denote the received coding overhead (meanwhile the transmission is in progress), i.e., \( \gamma \times k \) is the number of received output symbols at receiver.

Although LT codes with RS distribution \( \mu(\cdot) \) are asymptotically capacity achieving, i.e., \( \gamma_{\text{suc}} \to 1 \) as \( k \to \infty \) [1], when \( k \) is finite \( \gamma_{\text{suc}} \) becomes significantly larger than 1 [1, 4, 5], which may result in an inefficient FEC coding. Therefore, we exploit the feedback channel to obtain a much smaller \( \gamma_{\text{suc}} \) for a finite \( k \) in LT-AF coding.

### III. RELATED WORK

Previously, many works have investigated LT codes with feedback [6–9, 11, 12]. Authors in [6] proposed shifted LT (SLT) codes and have shown that when \( n \) input symbols have been recovered at the decoder, the degree of each arriving output symbol decreases by an expected \( \frac{k-n}{k} \) fraction (due to earlier recovery of their neighboring input symbol). Therefore, they propose to shift the RS distribution such that its average degree \( \mu'(1) \) is increased by \( \frac{k-n}{k} \), where \( \mu'(x) \) is the first derivative of \( \mu(x) \) with respect to its variable \( x \). With this setup, arriving output symbols at decoder always maintain an RS degree distribution regardless of the value of \( n \). SLT codes considerably improve the performance of LT codes. We make some changes to the idea of distribution shifting proposed in SLT codes and employ it in the design of LT-AF codes, while showing that LT-AF codes considerably outperform SLT codes.

In contributions [7] and [8], Growth codes and RT-oblivious codes have been proposed, respectively, which have basically the same structure. In these algorithms, as \( n \) increases and reaches to certain thresholds a feedback indicating that decoder has achieved the corresponding threshold is initiated. Therefore, the encoder gradually increases the degree of output symbols on-the-fly based on the feedbacks such that the instantaneous decoding probability of each delivered output symbol is maximized. Since Growth and RT-oblivious codes only consider the instantaneous recovery probability of each output symbol upon reception, they do not perform as well as SLT and LT-AF codes.

Authors in [9] propose to employ IS degree distribution \( \rho(\cdot) \) for LT coding. They have proposed to start decoding when an overhead of \( \gamma = 1 \) has been delivered to the decoder. When the decoding halts during the decoding process and some input symbols are remaining unrecovered, a randomly selected input symbol that is a neighbor of an output symbol of degree two is requested from the encoder. Despite the advantages of algorithm proposed in [9], in this scheme many feedbacks are issued back-to-back as soon as \( \gamma \) exceeds 1.

Authors in [11] redesigned the LT coding degree distributions when a few feedback opportunities are available based on \( k \). They show that incorporating feedback to LT codes can significantly decrease both the coding overhead and the encoding/decoding complexity. In [11], the point at which the feedback is initiated should be optimized, while in LT-AF codes no optimization is required. Further, new optimization of degree distribution is not necessary in LT-AF codes.

### IV. LT-AF CODES

Let \( \Omega_{k,n}(\cdot) \) denote the degree distribution of LT-AF codes for a data-block of length \( k \) when \( n \) input symbols are already recovered at decoder. We adopt the idea of SLT codes [6], and propose to shift \( \Omega_{k,n}(\cdot) \) based on \( n \) (see Section III). Therefore, we allow the decoder to issue the first type of feedback referred to as \( fb_1 \), which is used to keep the encoder updated with the current value of \( n \). The encoder also generates an output symbol of degree-one (containing a randomly selected input symbol) as acknowledgment as described in detail later.

Although IS distribution given by (1) is solely designed for the theoretical analysis of RS distribution, we slightly modify and employ it at the encoding phase of LT-AF codes in combination with two types of feedback. The IS distribution is tuned for \( \gamma_{\text{suc}} = 1 \) such that at each decoding iteration in expectation exactly one input symbol is recovered and only one output symbols is reduced to degree 1 and is added to the ripple. The single output symbol in the ripple with degree one can decode one input symbol in the next iteration. Since on average only a single degree one output symbol is generated for \( k \) output symbols (note that \( \rho(1) = \frac{1}{2} \)), the IS distribution would ideally realize an optimal coding/decoding, i.e., complete recovery of \( k \) input symbols from \( k \) output symbols and \( \gamma_{\text{suc}} = 1 \).

However, due to inherent randomness and uncertainties in the output symbol generation there is a high probability that an output symbol does not reduce to degree one when an input symbol is recovered. Consequently, the ripple becomes empty and the decoding stops although undecoded output and unrecovered input symbols are remaining. Despite this, we propose to employ IS distribution in LT-AF coding and exploit the feedback channel and request a suitable input symbol (which is an output symbol of degree 1) so that the decoding can continue. Therefore, we allow the decoder to request desired input symbols employing the second type of feedback referred to as \( fb_2 \).

Moreover, to design \( \Omega_{k,n}(\cdot) \) we propose to modify the IS distribution such that the encoder does not generate any degree-one output symbol. With this setup, we may exploit the degree-one output symbols as acknowledgments from the encoder to the reception of feedbacks. Therefore, the encoder generates a degree-one output symbol if and only if it has received a \( fb_1 \) or \( fb_2 \). Consequently, the lack of the arrival of an output symbol at the decoder with degree one after issuing a \( fb_1 \) or \( fb_2 \) clearly indicates a feedback loss. Consequently, all feedback packet losses are identified by the decoder and a feedback retransmission is performed. Therefore, the decoding recovery rate of LT-AF codes does not considerably degrade at high feedback channel loss rates \( \varepsilon_{fb} \in [0, 1] \) in contrast to existing work [6–10]. We should note that a degree-one output symbol generated at the encoder after a \( fb_1 \) contains a randomly selected input symbol. However, such an output
symbol would contain the requested input symbol (selected by decoder) after a \( f_{b_2} \).

Let \( \Omega_{k,n}(x) = \sum_{i=1}^{k} \Omega_{k,n,i} x^i \), where \( \Omega_{k,n,d} \) is the probability of selecting degree \( d \) to generate an LT-AF output symbol. Since we do not allow the encoder to generate any degree-one symbol, we set \( \Omega_{k,n,1} = 0 \). Further, considering the distribution shifting idea from [6] we define \( \Omega_{k,n,d} \) as follows.

\[
\Omega_{k,n,d} = \begin{cases} 
0 & \text{if } d = 1, \\
\frac{1}{k-d+1} \rho_{k-n}(i) & \text{if } d = 2, 3, \ldots, k, \left\lfloor \frac{i}{k-n} \right\rfloor = d,
\end{cases}
\]

where \( \left\lfloor \cdot \right\rfloor \) returns the closest integer to its argument, \( \rho_n(.) \) is the IS distribution for a data-block of length \( k \) given by (1), and \( \frac{1}{k-d+1} \) is the normalizing factor to have \( \sum_d \Omega_{k,n,d} = 1 \).

**Lemma 1:** The average degree of a check node generated employing \( \Omega_{k,n}(.) \) distribution is

\[
\sum_i \Omega(i) = \Omega'_{k,n}(1) = k^2 \ln (k-n) \ln (k-n-1),
\]

where \( \Omega'_{k,n}(x) \) is the first derivative of \( \Omega_{k,n}(x) \) with respect to its argument \( x \).

**Proof:** The average degree of IS distribution \( \rho(\cdot) \) is \( \ln k \) (see Section II). Therefore, it is easy to see that \( \Omega_{k,0}(1) = \frac{\ln k}{k-1} + \ln k \), since for \( n = 0 \) no shift occurs in IS distribution and degree-one check nodes are not generated. Generalization for \( \Omega'_{k,0}(1) \) is straightforward. Considering that the average degree of IS degree distribution for a data-block length of \( k-n \) is \( \ln (k-n) \) and the shift of degree distribution increases the average degree by a factor of \( \frac{k-n}{k-n-1} \), (4) is obtained.

### A. Generating \( f_{b_1} \)

Obviously, the decoder is not always aware of \( n \) unless its knowledge about \( n \) is updated by a \( f_{b_1} \). Initially, the encoder assumes \( n = 0 \) and employs the degree distribution \( \Omega_{k,\cdot}(\cdot) \) to generate output symbols. Let \( n_i \) denote the most recent reported value of \( n \) employing a \( f_{b_1} \). Similar to [6], we propose the encoder to generate a \( f_{b_1} \) when \( \Omega'_{k,n}(1) - \Omega'_{k,n,1}(1) = \sqrt{k} \ln k \), i.e., average degree of \( \Omega_{k,n}(\cdot) \) increases by at least \( \sqrt{k} \ln k \). Let \( n_i \) be the threshold that for \( n \geq n_i \) the \( i^{th} \) \( f_{b_1} \) is generated. In the following lemma we give the expression for \( n_i \).

**Lemma 2:** In LT-AF codes with data-block length \( k \), \( n_i \) the threshold of \( n \) for which \( i^{th} f_{b_1} \) is issued is recursively obtained as follows.

\[
n_i = \left[ k + \frac{W_{-1}(-A_i(k))}{A_i(k)} \right], i > 0,
\]

where \( A_i(k) = \frac{1}{k} \left( k \frac{1}{k} \sqrt{k} + \frac{k}{k-n_i} \ln (k-n_i-1) \right) \), \( W_m(.) \) is the \( m^{th} \) root of Lambert W-Function (the Lambert W-Function is defined as the inverse function of \( f(x) = x e^x \) [13]).

**Proof:** Let us first analyze \( n_{11} \), the value of \( n \) that initiates the first \( f_{b_1} \). Since before the first \( f_{b_1} \) no distribution shifting occurs we have \( \Omega'_{k,n_0}(1) = \frac{\ln k}{k-n_0} \ln k \).

Therefore, the first \( f_{b_1} \) is issued for a value of \( n_1 \) that \( \Omega'_{k,n_1}(1) - \Omega'_{k,n_0}(1) = \sqrt{k} \ln k \). Using Lemma 1 we have

\[
\frac{k}{k-n_1} \ln (k-n_1) - \frac{k}{k-n_0} \ln (k-n_0) = \sqrt{k} \ln k,
\]

which gives

\[
\ln (k-n_1) = \frac{1}{k} \left( \frac{k-1}{k} \sqrt{k} \ln k + \frac{k}{k-n_0} \ln (k-n_0) \right).
\]

Next, let \( A_i(k) = \frac{1}{k} \left( k \frac{1}{k} \sqrt{k} + \frac{k}{k-n_i} \ln (k-n_i-1) \right) \). Since \( A_i(k) > -1, \forall k, i \) employing Lambert’s W function, we have

\[
k - n_1 = \frac{W_{-1}(-A_i(k))}{A_i(k)},
\]

which gives

\[
n_1 = \left[ k + \frac{W_{-1}(-A_i(k))}{A_i(k)} \right].
\]

Finally, we have \( \Omega'_{k,n_1}(1) - \Omega'_{k,n_i}(1) = \sqrt{k} \ln k \) that proves the lemma.

**Lemma 2** gives the value of \( n \) for which \( f_{b_1} \)'s are generated.

In Figure 1, we have depicted \( \frac{n_i}{k}, i \in \{1, 2, \ldots, 5\} \) versus \( k \). From Figure 1, we can see that \( \frac{n_i}{k} \) decreases as \( k \) increases. As an example, we can see that at \( k = 10^2 \) the first and the second \( f_{b_1} \)'s are issued at \( n \geq 39 \) and \( n \geq 58 \), respectively. Further, for \( k = 10^4 \) the first and the second \( f_{b_1} \)'s are issued at \( n \geq 2740 \) and \( n \geq 4346 \), respectively.

![Fig. 1. Values of \( \frac{n_i}{k} \) versus \( k \) for \( f_{b_1} \) generation.](image)

### B. Generating \( f_{b_2} \)

Since in LT-AF coding no degree-one output symbol is generated, no decoding is performed and we have \( n = 0 \) until some degree-one output symbols are requested employing \( f_{b_2} \)'s. The idea to generate \( f_{b_2} \) is to smartly and greedily choose and request an input symbol that makes the largest
progress toward decoding completion. It is well-known that LT codes have all-or-nothing decoding property (also called waterfall phenomenon) [1], where an abrupt jump in the ratio of decoded input symbols occurs at a γ close to $\gamma_{\text{succ}} > 1$. Therefore, transmission of $fb_2$'s before γ = 1 does not considerably contribute to decoding progress. Therefore, we propose to generate $fb_2$'s only when γ surpasses 1. To have uniformly distributed $fb_2$'s and to avoid feedback channel congestion, an LT-AF decoder issues a $fb_2$ on the reception of every $(\ln k)^{th}$ output symbol.

Let us first describe the structure of input and output symbols in the buffer of a decoder. Input and received output symbols of an LT code at a decoder can be viewed as vertices of a bipartite graph $G$. The input symbols are the variable nodes $v_i, i \in \{1, \ldots, k\}$ and the output symbols are the check nodes $c_j, j \in \{1, 2, \ldots, \gamma k\}$ [3, 14], and they are connected to their neighbors denoted by $\mathcal{N}(v_i)$ and $\mathcal{N}(c_j)$, respectively, with undirected edges.

During data transmission some variable nodes $v_i, i \in \{1, \ldots, k\}$ are decoded and some check nodes $c_j, j \in \{1, 2, \ldots, \gamma k\}$ are reduced to degree zero and are both removed from the decoding graph $G$. Let us refer to the set of remaining undecoded variable nodes by $\mathcal{V}_{\text{un}}$ and the set of buffered check nodes with a degree higher than one by $\mathcal{C}_{\text{buff}}$.

We remind that the check nodes with degree 1 are called the ripple. Figure 2 illustrates such a graph $G$ at a decoder at γ = 1 for $k = 7$. Note that we interchangeably employ the terms variable and check nodes for input and output symbols, respectively, in the rest of the paper.

![Fig. 2. The bipartite graph representing the input and the output symbols of an LT-AF code at the buffer of a decoder.](image)

It is important to note that the design of $fb_2$ is to greedily decode as many input symbols as possible so the decoding succeeds at a smaller $\gamma_{\text{succ}}$. However, as discussed earlier as $n$ increases closer to the end of decoding the average degree of check nodes should be increased to decrease the probability that they become redundant due to earlier recovery of all their neighboring variable nodes. This is the rationale to employ the distribution shifting and $fb_1$ along with $fb_2$. In the next sections, we devise three algorithms to analyze the graph $G$ at decoder and greedily select suitable variable nodes to generate $fb_2$'s.

1) Generating $fb_2$ Based on Variable Node with Maximum Degree (VMD): One insight into choosing a suitable variable node is requesting the variable node $v_1 \in \mathcal{V}_{\text{un}}$ with the maximum degree. Such a selection greedily removes the highest number of edges in the first step of decoding after the delivery of the respective input symbol. Based on this idea we propose an algorithm called “Variable Node with Maximum Degree” (VMD), where the decoder requests the variable node with the highest degree in its current decoding graph to issue a $fb_2$. For instance, in Figure 2 VMD would request $v_5$. On the arrival of $c_8$ containing only $v_5$, the value of $v_5$ will become known. This removes all the edges emanating from $v_5$ to all other check nodes and reduces some to degree 1 (they are added to the ripple). For instance, $c_7$ is added to the ripple, which recovers $v_7$ in the next decoding iteration. Note that at this step the ripple becomes empty and decoding stalls; hence we have $\mathcal{C}_{\text{buff}} = \{c_1, c_2, \ldots, c_6\}$ and $\mathcal{V}_{\text{un}} = \{v_1, v_2, v_3, v_4, v_6\}$.

We can see that VMD greedily removes the largest possible number of edges from $G$ and decreases the degree of many check nodes.

2) Generating $fb_2$ Based on Full Variable Node Decoding (FVD): A more complex method to generate $fb_2$ is to run a dummy decoding for all unrecovered input symbols. Next, the single input symbol whose delivery results in the highest number of decoded input symbols is requested by a $fb_2$. We refer to this method by “Full Variable Node Decoding” (FVD), which has a much higher complexity than VMD.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of LT-AF codes employing numerical simulations. Our results are obtained employing Monte-Carlo method by averaging over the results of at least $10^7$ numerical simulations.

A. LT-AF Decoding Error Rate and Runtime

![Fig. 3. Decoding error rate comparison for $k = 1000$.](image)

![Fig. 4. Decoding success rate comparison for $k = 1000$.](image)

We plot the decoding bit-error-rate (BER) (average ratio of unrecovered input symbols to total number of input symbols $1 - E\left[\hat{c}\right]$) and the ratio of successful decodings versus received overhead γ in Figures 3 and 4, respectively, for $k = 1000$. Note that we set $c = 0.9$ and $\delta = 0.1$ for SLT and LT codes as proposed in [6].
Figures 3 and 4 show that LT-AF codes significantly surpass SLT codes. Figure 3 shows that SLT codes require $\gamma_{\text{succ}} = 1.31$ for decoding completion (BER $\leq 10^{-8}$), which has been reduced to $\gamma_{\text{succ}} = 1.14$ and $\gamma_{\text{succ}} = 1.09$ using FVD and VMD. However, we should note that in LT-AF coding the average degree of output symbols is much higher than that of regular LT codes, which causes a higher encoding/decoding complexity.

B. Number of Feedbacks

In this section, we compare the total number of feedbacks issued by LT-AF codes and compare it to that of SLT codes. We emphasize that other proposed LT codes with feedback cannot achieve the performance of SLT and LT-AF codes. The expected number of feedbacks for LT-AF and SLT codes are summarized in Table I. From Table I, we can interestingly observe that not only LT-AF codes decrease the required coding overhead for a successful decoding $\gamma_{\text{succ}}$, but also they need slightly smaller number of feedbacks compared to SLT codes. Further, we should note that the total number of feedbacks is much smaller than $\gamma_{\text{succ}}$.  

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$N = 1000$</th>
<th>$f_{b1}$</th>
<th>$f_{b2}$</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>LT-AF,VMD</td>
<td>2.68</td>
<td>9.29</td>
<td>11.97</td>
<td></td>
</tr>
<tr>
<td>LT-AF,FVD</td>
<td>3.58</td>
<td>6.92</td>
<td>10.5</td>
<td></td>
</tr>
<tr>
<td>SLT</td>
<td>-</td>
<td>-</td>
<td>12.27</td>
<td></td>
</tr>
</tbody>
</table>

C. Robustness to Erasure in Feedback Channel

We evaluate the effect of feedback loss on the performance of LT-AF and SLT codes. Assume that the loss rate of the feedback channel is $\epsilon_{fb} = 0.9$ (which is not known to encoder and decoder), hence 90% of the feedbacks are lost in transmission. Note, that in a lossy forward channel the degree-one acknowledgements may also be dropped while $f_{b1}$ or $f_{b2}$ may have already been delivered. In the case of $f_{b2}$ loss, the retransmission compensates this loss. However, in case of $f_{b1}$ loss, the encoder shifts the degree distribution accordingly while the decoder remains unaware of this shift. In this case, feedback retransmission is not even required since the degree distribution shift has already occurred. Therefore, we consider the worst case in our simulations and assume that if an acknowledgement is lost the distribution shifting does not occur as well. Figure 5 shows the performance of LT-AF and SLT codes for $k = 1000$ and $\epsilon_{fb} = 0.9$.

Figure 5 shows the excellent resilience of LT-AF codes to feedback loss in contrast to SLT codes. In practice, the performance of SLT codes approach that of regular LT codes as the feedback loss increases. To the best of our knowledge robustness against feedback loss had not been considered in any existing work and this significantly distinguishes LT-AF codes.

VI. CONCLUSION

In this paper, we proposed LT-AF codes that are LT codes with two types of feedback, which alleviate the low performance of LT codes for short data-block lengths. In LT-AF codes, the decoder may inform the encoder with the total number of decoded input symbols by the first type of feedback or request a certain input symbol from the encoder employing second type of feedback. We showed that LT-AF codes require lower coding overhead for successful decoding and lower number of feedback compared to existing work. Finally and most importantly, LT-AF codes’ performance does not considerably degrade at large loss rates in the feedback channel. On the other hand, LT-AF codes generate output symbols with a higher average degree resulting in coding/decong with higher complexity.

VII. ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grants ECCS-1056065 and CCF-0915994.

REFERENCES