ChirpSmoother: Eliminating Noise Amplification for Robust LoRa Weak-Signal Reception

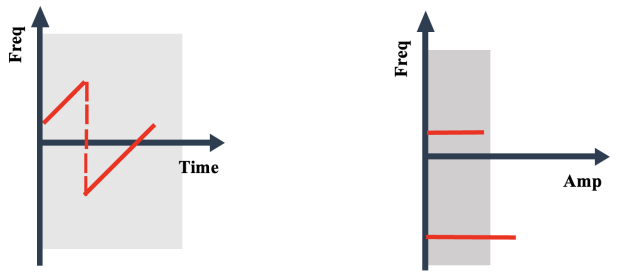
Abstract

LoRa has emerged as a leading Low-Power Wide-Area Network (LPWAN) technology for Internet of Things (IoT) applications due to its exceptional range and energy efficiency. However, the inherent frequency and phase discontinuities in LoRa's Chirp Spread Spectrum (CSS) modulation fundamentally limit weak signal decoding performance. This paper presents ChirpSmoother, a novel approach that eliminates frequency jumps entirely rather than attempting to mitigate their effects. By leveraging LoRa's frequency hopping mechanism and encoding mechanism in an innovative way, ChirpSmoother transmits frequency-jump-free chirps. This fundamental redesign avoids the phase misalignment and frequency discontinuity problem while preventing the noise amplification inherent in peak merging approaches. We address the challenge of cross-channel interference in multi-channel deployments through a temporal segmentation algorithm, enabling robust operation in dense LoRaWAN environments. We implemented ChirpSmoother on COTS LoRa radios and evaluated its performance. Experimental results demonstrate that ChirpSmoother achieves 0.68 to 1.72 dB gain over the state-of-the-art and 3.01 to 4.27 dB gain over peak merging methods, enabling reliable communication at signal levels previously considered undecodable. Crucially, ChirpSmoother achieves this gain while maintaining the O(n log n) complexity of standard LoRa, ensuring its feasibility on resource-constrained devices.

1 Introduction

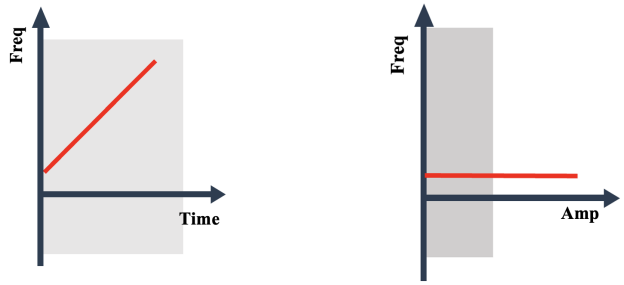
Low-Power Wide-Area Networks (LPWANs) have emerged as a transformative technology for connecting billions of Internet of Things (IoT) devices across diverse applications ranging from smart cities to industrial monitoring and agricultural sensing [3, 23, 29, 34, 35]. Among various LPWAN technologies, LoRa[21] has gained widespread adoption due to its operation in unlicensed spectrum bands and its excellent performance in terms of communication range and energy efficiency. By employing CSS [2] modulation, LoRa enables sensor nodes to communicate over distances of several kilometers while maintaining battery lifetimes of up to ten years [5, 12, 26].

However, the practical deployment of LoRa networks faces significant challenges in achieving the advertised communication ranges, particularly in urban and indoor environments. Recent measurement studies reveal that the actual coverage of LoRa gateways can be dramatically reduced from the theoretical tens of kilometers to merely 1-2 kilometers in dense urban areas due to signal blockage, multipath propagation, and interference [4, 11, 15, 18, 22, 27]. This severe range reduction forces a choice between costly dense gateway deployments and power-hungry transmissions that shorten battery life. The fundamental issue is that LoRa signals operating at the edge of coverage areas suffer from extremely low Signal-to-Noise Ratios (SNRs), often falling below the demodulation threshold even with the maximum spreading factor configuration.



(a) Chirp (b) Chirp after FFT

Figure 1: Noise Amplification in LoRa Demodulation



(a) Chirp (b) Chirp after FFT

Figure 2: ChirpSmoother Demodulation without Noise

The root cause of LoRa's limited weak signal performance lies in a fundamental characteristic of its CSS modulation that has been largely overlooked: the frequency discontinuity inherent in practical LoRa chirp signals. Unlike ideal chirps that sweep continuously across the frequency band,

real LoRa transmitters generate chirps with an abrupt frequency jump from the maximum frequency B back to 0 Hz. This frequency wrapping, combined with hardware-induced phase misalignment between chirp segments, creates a critical bottleneck in the demodulation process. As shown in Fig.1, the standard LoRa decoder needs to merge the energy from the two disconnected frequency components, but this peak merging process fundamentally amplifies noise alongside the signal, resulting in a theoretical 6 dB SNR penalty that severely limits weak signal reception.

The common thread across all existing approaches[5, 6, 8, 10, 30, 32] is that they attempt to work around the frequency discontinuity problem rather than addressing it directly. Whether through spatial diversity, computational complexity, or transmitter coordination, these methods add substantial overhead while achieving only incremental improvements because they cannot eliminate the fundamental noise amplification caused by peak merging.

In this paper, we present ChirpSmoother, a novel LoRa enhancement technique that fundamentally eliminates frequency jumps in chirp signals, thereby avoiding the noise amplification penalty entirely. Unlike existing approaches that attempt to mitigate the effects of frequency discontinuities through increasingly complex algorithms, ChirpSmoother prevents these discontinuities from occurring in the first place.

Our key insight is that LoRa's encoding mechanism and frequency hopping capability can be repurposed to generate frequency-jump-free chirps, as shown in Fig.2, while maintaining full compatibility with the LoRa standard. By transmitting the whitening sequence as payload data, we exploit the deterministic encoding pipeline to generate symbols that, after standard LoRa processing, produce chirps without frequency wrapping. We then leverage LoRa's built-in frequency hopping mechanism—originally designed for regulatory compliance—to encode information through controlled channel frequency offsets rather than varying the initial frequency within a single channel.

This fundamental redesign offers critical advantages by eliminating frequency jumps to eradicate the severe noise amplification penalty from peak merging, thereby maximizing the signal-to-noise ratio. Crucially, it achieves this while preserving the computational simplicity of conventional LoRa, maintaining an O(n log n) complexity suitable for resource-constrained gateways. This software-based approach ensures seamless integration with commercial COTS LoRa radios, enabling widespread deployment via simple Over-the-Air (OTA) firmware updates without physical modifications.

Implementing ChirpSmoother presents several technical challenges that we address through novel algorithmic innovations:

Cross-channel interference emerges as a unique challenge because ChirpSmoother's frequency-hopped symbols may extend beyond the nominal channel bandwidth. In dense LoRaWAN deployments with multiple adjacent channels, this can cause interference between simultaneous transmissions. We develop a temporal segmentation algorithm that exploits the time-domain characteristics of interference patterns to identify and cancel cross-channel interference while preserving full signal energy.

Compatibility with LoRaWAN infrastructure requires careful consideration of how frequency-jump-free chirps interact with existing network protocols and channel allocation schemes. We design ChirpSmoother to maintain standard LoRaWAN channel spacing and timing requirements while adapting the decoder to handle the extended frequency occupancy of our modified chirps.

These substantial performance gains are achieved while maintaining full compatibility with existing LoRa infrastructure and without increasing computational complexity, making ChirpSmoother immediately deployable for enhancing IoT connectivity in challenging propagation environments.

This paper makes three key contributions:

- We identify and theoretically analyze the fundamental limitation of LoRa's CSS modulation—the noise amplification caused by frequency discontinuities and the subsequent peak merging process—showing that even optimal phase alignment incurs a 6 dB SNR penalty.
- We present ChirpSmoother, the first approach that eliminates frequency jumps entirely by innovatively repurposing LoRa's whitening and frequency hopping mechanisms, achieving significant SNR improvements without computational overhead or hardware modifications.
- We implement and validate ChirpSmoother on commercial COTS LoRa hardware (SX1276 [25]) with a USRP X410 [9] gateway. The real-world implementation delivers 0.68 to 1.72 dB improvement over state-of-the-art methods while maintaining remarkably low computational overhead.

2 Background

In this section, we analyze the LoRa modulation and demodulation, and reveal the underlying causes that constrain LoRa demodulation performance.

LoRa CSS Modulation. LoRa employs Chirp Spread Spectrum (CSS) modulation, where information is encoded in chirp signals whose frequency varies linearly with time. A LoRa symbol is fundamentally a chirp signal that sweeps across the entire bandwidth B during a symbol period T. Each symbol encodes information by offsetting the initial frequency of the chirp. The fundamental base up-chirp, with an initial frequency of zero, can be mathematically expressed

as:

$$u_b(t;\phi) = e^{j(2\pi\cdot\frac{B}{2T}t^2+\phi)} \tag{1}$$

The term ϕ represents the initial phase of a LoRa chirp. The symbol duration T is defined as:

$$T = \frac{2^{SF}}{B} \tag{2}$$

where SF denotes the spreading factor, a fundamental parameter in LoRa modulation.

Each chirp symbol encodes SF bits of information, resulting in 2^{SF} distinct symbols that can be represented by integers ranging from 0 to $2^{SF}-1$. The symbol 0 corresponds to the base up-chirp with an initial frequency of 0 Hz. For general chirp symbols with non-zero initial frequencies, the signal can be characterized as $x_s(t)$:

$$u(t;f;\phi) = e^{j(2\pi\cdot(f+\frac{B}{2T}t)\cdot t+\phi)} \tag{3}$$

$$x_s(t) = \begin{cases} A\cdot u(t;f_0,\phi_0), & \text{if } 0 \le t < t_0 \\ A\cdot u(t;f_0-B,\phi_1), & \text{if } t_0 \le t < T \end{cases} \tag{4}$$

where the transition time t_0 is given by:

$$t_0 = (1-\frac{f_0}{B})T \tag{5}$$

The inherent frequency wrapping in LoRa chirps (sweeping 0 to B then jumping to 0) splits the signal into two temporal segments. This transition, particularly in low-cost hardware, creates a phase discontinuity and subsequent misalignment ($\phi_0 \neq \phi_1$), which critically degrades demodulation performance.

LoRa Demodulation Process. Let the noise be represented as $x_n(t)$. The received signal can be expressed as:

$$x_r(t) = x_s(t) + x_n(t) \tag{6}$$

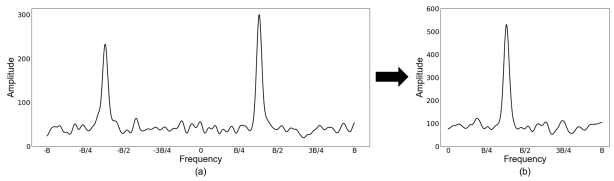


Figure 3: Peak merging in LoRaPHY. (a) Original Fourier spectrum after dechirp. (b) Spectrum after peak merge. Peak merging enhances both the signal and the noise.

The standard LoRa demodulation process begins with dechirp, where the received signal is multiplied by a base down-chirp:

$$x_d(t) = e^{j(2\pi\cdot\frac{B}{2T}t^2)} \tag{7}$$

The dechirped signal becomes:

$$x_r'(t) = x_r(t)\cdot x_d \tag{8}$$

$$x_r'(t) = \begin{cases} A\cdot e^{j2\pi f_0 t} + x_n'(t), & \text{if } 0 \le t < t_0 \\ A\cdot e^{j2\pi(f_0-B)t} + x_n'(t), & \text{if } t_0 \le t < T \end{cases} \tag{9}$$

After dechirp, Fast Fourier Transform (FFT) is applied to convert the time-domain signal into the frequency domain for symbol detection.

Let f_0 denote the initial frequency; the LoRa decoding process estimates it as:

$$\hat{f_0} = \arg\max_f |X_r(f)| \tag{10}$$

$\hat{f_0}$ represents the frequency corresponding to the maximum peak in the Fourier transform. f_0 denotes the actual initial frequency. Correct decoding is achieved when both are equal.

Core Limitation. After dechirping, the signal energy splits into two frequency components at f_0 and f_0-B. Consequently, peak merging process is necessary to combine these components and recover the full signal energy. However, this process is fundamentally flawed. The phase misalignment between the two signal segments can cause destructive interference, potentially canceling out signal energy, as illustrated by the combined peak magnitude:

$$X(f_0) = t_0\cdot Ae^{j\phi_0} + (T-t_0)\cdot Ae^{j\phi_1} + X_n'(f_0) \tag{11}$$

Even if phase alignment techniques (often requiring more computation) perfectly superimpose the signal energy, they inevitably also superimpose noise from both frequency regions, as shown in Fig. 3. Because the essence of peak merging is to superimpose two frequency spectrum segments (-B,0) and (0,B), not only is the signal enhanced in this process, but the noise is also enhanced. This inescapable noise amplification during peak merging is the core limitation that degrades LoRa's weak-signal demodulation performance.

3 ChirpSmoother Overview

ChirpSmoother enhances weak signal decoding performance by addressing the noise amplification that occurs during signal energy combination. ChirpSmoother consists of two main components: an encoder at the transmitter and a decoder at the receiver.

The encoder is responsible for generating signals on COTS, while the decoder is responsible for decoding data at the gateway and resolving potential adjacent signal interference issues efficiently.

4 ChirpSmoother Encoder

At the transmitter, we first obtain chirps without frequency jumps, then encode information by modifying the initial frequency on these frequency-jump-free chirps.

4.1 Frequency Jump Elimination

LoRa employs CSS modulation technology, where the frequency increases linearly from f_0 to B, then jumps to 0, and subsequently increases linearly from 0 to f_0. The frequency jump from B to 0 limits decoding performance. Through analysis of the LoRa encoding mechanism, we observe that when the initial frequency is 0, no frequency jump occurs, and the frequency increases linearly from 0 to B. This type of chirp is free from frequency jumps and can be leveraged for our approach.

However, how can we obtain such chirps? LoRa encoding includes whitening, Hamming encoding, interleaving, and inverse Gray decoding. Due to the LoRa encoding process, complete control over the encoded information is impossible (e.g., with Hamming encoding, the parity bits are entirely determined by the data bits, making it impossible to modify parity bits when data bits are fixed). Although complete manipulation of the encoded content is not feasible, certain specific encoding sequences are attainable. We need a base up-chirp without frequency jumps.

We noticed that by setting the payload data to the whitening sequence, during the first whitening operation, the payload content becomes the whitening sequence itself. Whitening essentially performs an XOR operation with the whitening sequence. When the whitening sequence is XORed with itself, it generates an all-zero sequence. After Hamming encoding operations on all-zero data, the added parity bits remain zero. Interleaving only changes the order of data sequence bits; when all bits are zero, reordering still results in all zeros. Gray decoding essentially involves numerous XOR operations, and identical binary bits remain zero after XOR operations. By transmitting the whitening sequence, it is theoretically possible to obtain all-zero data.

However, in practice, we observe that LoRa's inverse Gray decoding is not standard Gray decoding. The standard Gray coding is:

$$v = v_0 \oplus (v_0 >> 1) \tag{12}$$

LoRa's Gray coding is:

$$v = (v_0 - 1) \oplus ((v_0 - 1) >> 1) \tag{13}$$

LoRa's Gray decoding is:

$$v_0 = v \oplus (v >> 1) \oplus (v >> 2) \ldots + 1 \tag{14}$$

Consequently, the final result obtained through LoRa's Gray decoding is an all-ones sequence. All-zero sequences and all-ones sequences have very similar symbol characteristics. The performance of all-zero and all-one symbols is comparable. While the all-one symbol chirp is technically segmented into two temporal regions, the energy distribution between these segments is highly asymmetric. For SF=7, which represents the case with the smallest energy asymmetry, the first segment contains 127/128 of the total chirp energy, with only 1/128 residing in the second segment. This results in a negligible energy loss of approximately 0.034 dB when utilizing only the dominant segment. As the spreading factor increases, this energy asymmetry becomes even more pronounced. For instance, at SF=10, the energy ratio reaches 1023:1, corresponding to an energy loss of merely 1/1024 (approximately 0.0042 dB) when discarding the second segment. Therefore, for practical purposes, all-one symbols can be treated as effectively equivalent to frequency-jump-free chirps, exhibiting similar weak signal decoding performance to all-zero symbols. In the remainder of this paper, we conduct analysis using all-zero symbols.

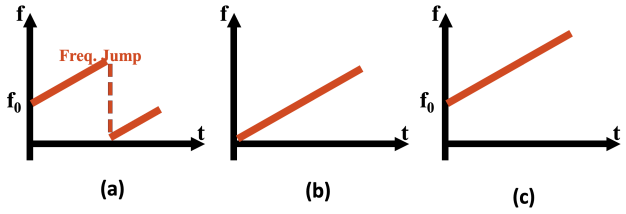


Figure 4: ChirpSmoother encoder. (a) Traditional LoRa chirp. (b) Chirp without frequency jump. (c) Frequency hopping on a frequency-jump-free chirp.

4.2 Frequency Hopping in ChirpSmoother

By sending the whitening sequence, we can obtain symbols similar to base up-chirps. However, these symbols have identical initial frequencies. Symbols without variation cannot encode data. To introduce differences among these symbols, we discovered that LoRa has a frequency hopping mechanism that allows different symbols to operate on different channels.

Frequency hopping [17] is primarily designed to meet regulatory requirements, ensuring compliance with regulatory agencies' limits on maximum channel dwell time. When the transmission time of a single data packet may exceed the regulatory upper limit for channel occupancy time, frequency hopping technology must be used to meet compliance requirements.

Unlike Bluetooth's packet-level frequency hopping, LoRa supports symbol-level frequency hopping. LoRa supports frequency hopping with a period defined by HoppingPeriod:

$$HoppingPeriod = T_s \times FreqHoppingPeriod \quad (15)$$

where T_s represents the symbol duration. When FreqHoppingPeriod is set to 1, the system hops to a new frequency after every single symbol, enabling precise per-symbol frequency control.

ChirpSmoother repurposes the LoRa frequency hopping mechanism for a novel encoding scheme. During standard operation, the transmitter sequentially cycles through a predefined frequency table managed by the host microcontroller. At each hop, the FhssPresentChannel counter increments and triggers the FhssChangeChannel interrupt, signaling the need for a frequency update. We exploit this mechanism by constructing a custom frequency hopping table where each entry corresponds to a specific symbol value. The frequency offset for symbol **s** is calculated as:

$$f_{hop}(s) = f_{base} + s \cdot \Delta f \quad (16)$$

where f_{base} is the base channel frequency, $s \in [0, 2^{SF}-1]$ is the symbol value, and $\Delta f = \frac{B}{2^{SF}}$ is the frequency resolution.

Fig.4 illustrates the transformation from traditional LoRa chirps with frequency jumps to ChirpSmoother's frequency-jump-free chirps achieved through controlled frequency hopping. The resulting signal maintains the chirp spread spectrum characteristics while eliminating the discontinuities that limit weak signal performance.

5 ChirpSmoother Decoder

The ChirpSmoother decoder is designed to efficiently decode frequency-jump-free chirp signals while handling potential cross-channel interference in dense LoRaWAN deployments. We present both the standard decoding process and our novel interference cancellation mechanism.

In the standard decoding process, ChirpSmoother follows the same fundamental pipeline as traditional LoRa: dechirp operation, FFT processing, and peak detection. The key difference lies in the signal characteristics after dechirping. Unlike traditional LoRa, where frequency-wrapped chirps produce two distinct frequency components, ChirpSmoother's frequency-jump-free signals result in a single, concentrated frequency peak rather than two peaks.

5.1 Cross-Channel Interference Analysis

In practical LoRaWAN deployments, multiple devices transmit simultaneously on adjacent channels. ChirpSmoother maintains compatibility with standard LoRaWAN channel allocation while addressing unique challenges arising from its extended frequency occupancy.

ChirpSmoother adopts standard LoRaWAN channel spacing: center frequency spacing of 200 kHz, channel bandwidth of 125 kHz, and a 75 kHz guard band between adjacent channels.

However, unlike traditional LoRa where symbols occupy a fixed frequency range (0, B), ChirpSmoother symbols may occupy different ranges depending on their initial frequency. For example, symbol 0 occupies (0, B), symbol 2^{SF-1} occupies (B/2, 3B/2), and a general symbol s occupies (f_0, $f_0 + B$) where $f_0 = s \cdot B/2^{SF}$.

This variable frequency occupancy necessitates oversampling at a rate of 2B to capture all possible signal components.

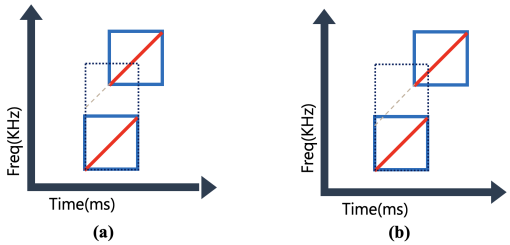


Figure 5: (a) Without interference. (b) With interference.

While ChirpSmoother's variable frequency occupancy enables frequency-jump-free transmission, it introduces unique interference patterns when adjacent channel signals overlap. We analyze the interference mechanisms and identify exploitable characteristics for effective mitigation.

The interference characteristics are determined by three key factors: **symbol initial frequency**, which determines the frequency overlap region; **relative time offset**, which controls the temporal overlap between signals; and the **dechirp transformation**, after which the frequency-domain interference from adjacent channels becomes temporally localized, appearing only in specific time segments rather than throughout the entire symbol period.

Different data packets have different reception time differences. Taking the case where the initial frequency is 0 as an example, we can divide this into two scenarios:

5.2 Cross-Channel Interference Cancellation

Case 1: Receiving adjacent channel signals without interference. As shown in Fig 5(a), this situation does not require interference cancellation. Although adjacent channels intrude into the current channel, after dechirp and FFT the peaks generated by adjacent channels exceed B and fall outside the (0, B) range. Since signals from the current channel cannot produce peaks exceeding B, these can be excluded.

Case 2: Receiving adjacent channel signals with interference. As shown in Fig. 5(b), when the time difference

between two chirp windows is small, interference does not occur. Only when the time difference between two chirp windows is large do the dechirp and FFT peaks of adjacent chirp signals potentially fall within the decoding range of the current channel (intersection of the dashed extension line of adjacent channel chirp signals with the start time of the current channel window). When the signal window time difference is large, there is less interference signal from adjacent channels (adjacent signals enclosed in the dashed rectangle in Fig.5).

Observing the interference signal situation, we find that in the received signal the first half has no signal interference. Leveraging this discovery, we perform interference cancellation.

When cross-channel interference occurs, the time-domain characteristics of adjacent channel interference signals exhibit distinctive temporal localization patterns. As illustrated in Fig. 6, when signals from higher-frequency adjacent channels interfere with the current channel, the interference manifests exclusively in the latter portion of the time domain, leaving the initial segment completely interference-free. This temporal segregation property forms the foundation of our cancellation approach.

The temporal localization occurs because the interfering chirp's instantaneous frequency only enters the current channel's band after a specific time offset. For an interfering signal from an adjacent channel with center frequency offset Δf_c, the interference begins at time t_{int} when its instantaneous frequency crosses into the current channel's upper band edge. This creates a guaranteed clean window of duration $T_{clean} = t_{int}$ at the beginning of each symbol.

Simply performing segmented Fourier transforms and utilizing only the interference-free segment would result in significant signal energy loss, as we would discard the energy from the interference-affected portion. Our approach instead aims to preserve the full signal energy while selectively removing interference components.

Our cross-channel interference cancellation process operates through systematic detection and mitigation phases. Initially, we perform FFT on the entire dechirped signal to identify the primary peak location. If this peak falls within a region where interference is impossible—specifically when $f_{peak} < B - \Delta f_c$—we directly output the result without further processing, as interference from adjacent channels cannot affect these frequencies.

However, when the FFT peak falls within the potential interference zone, we proceed with temporal segmentation analysis. The critical insight is that the maximum temporal extent of interference corresponds to the base up-chirp case (symbol 0), while other symbols exhibit smaller interference windows. By calculating the division point based on the worst-case scenario, we guarantee complete isolation of the clean signal portion:

$$\frac{B}{T} = \frac{B - f_0 + B_{int}}{T_{sep}} \tag{17}$$

Rearranging this equation yields the temporal division point:

$$T_{sep} = T \cdot \left(1 - \frac{f_0 - B_{int}}{B}\right) \tag{18}$$

where B_{int} represents the frequency offset at which interference begins, typically equal to the guard band width.

We then perform separate FFT operations on the segments $[0, T_{sep}]$ and $[T_{sep}, T]$. The front segment spectrum X_{front} contains only the desired signal, while the rear segment spectrum X_{rear} contains both signal and interference components. By comparing these spectra, we can identify interference peaks—these appear as frequency components with significantly higher magnitude in X_{rear} compared to X_{front}.

Upon detecting interference peaks, we synthesize a cancellation signal specifically for the affected time segment. The cancellation process involves generating an anti-phase replica of the identified interference component:

$$x_{cancel}(t) = \sum_{i \in \mathcal{I}} A_i \cdot e^{-j(2\pi f_i t + \phi_i)} \cdot w(t - T_{sep}) \tag{19}$$

where $\mathcal{I}$ denotes the set of identified interference frequencies, A_i and ϕ_i are the amplitude and phase extracted from X_{rear}, and $w(t)$ is a smooth windowing function that prevents discontinuities at the segment boundary.

The cancellation signal is applied only to the affected portion:

$$x'_{cleaned}(t) = \begin{cases} x'r(t), & 0 \le t < T_{sep} \\ x'r(t) - x_{cancel}(t), & T_{sep} \le t < T \end{cases} \tag{20}$$

After applying the cancellation signal, we perform a final FFT on the entire cleaned signal, ensuring full utilization of the symbol energy while effectively suppressing interference. This approach maintains the 6 dB advantage of ChirpSmoother over peak-merging methods while enabling robust operation in dense multi-channel deployments.

6 Theoretical Performance Analysis

6.1 Algorithm Complexity

ChirpSmoother maintains low computational complexity while achieving significant performance gains. The fundamental decoding pipeline comprises three standard operations identical to conventional LoRa demodulation: dechirp,

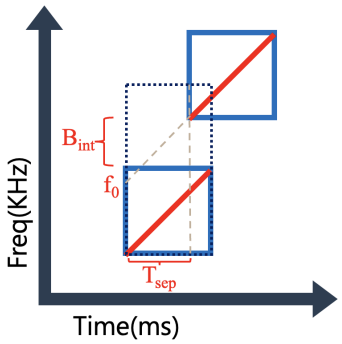


Figure 6: Interference condition.

FFT, and peak detection. For a LoRa symbol with n samples, the dechirp operation requires n complex multiplications to multiply the received signal with the base downchirp, yielding $O(n)$ time complexity. The subsequent Fast Fourier Transform converts the dechirped signal to the frequency domain with $O(n \log n)$ complexity using standard FFT algorithms. Finally, peak detection scans through n frequency bins to identify the maximum magnitude, contributing $O(n)$ complexity. Consequently, the total time complexity of ChirpSmoother remains $O(n \log n)$, identical to standard LoRa demodulation methods.

The interference cancellation mechanism, when triggered, introduces additional segmented FFT operations on windows of size $m = n \cdot (T - T_{sep})/T$, where $m < n/2$ in typical scenarios. These segmented operations add complexity $O(m \log m)$, but since the primary FFT operation dominates with $O(n \log n)$ complexity, the asymptotic complexity remains unchanged. Furthermore, interference cancellation is only activated when peaks fall within the potential interference zone and actual interference is detected through spectral comparison, significantly reducing the average computational burden. The segmented FFTs operate on smaller data sets, making the incremental cost negligible compared to the primary decoding operations. This computational efficiency ensures that ChirpSmoother remains suitable for real-time implementation on resource-constrained LoRa gateways while delivering substantial weak signal enhancement. Unlike deep learning-based approaches that require extensive computational resources and model storage, ChirpSmoother's lightweight design aligns with the fundamental LPWAN principle of minimizing device complexity.

6.2 Power Loss Due to Phase Misalignment

We have two assumptions for our theoretical model:

Assumption 1: The phases ϕ_0 and ϕ_1 (before and after the frequency jump) follow uniform distributions over $[0, 2\pi)$ and are statistically independent.

Assumption 2: The noise components $n'(t)$ follow complex Gaussian distributions with zero mean and variance σ^2, i.e., $n'(t) \sim \mathrm{CN}(0, \sigma^2)$.

Traditional LoRa signals form two single-tone signals after the dechirp operation. The dechirped signal can be described in Eq. 9. After FFT operation, the two signal segments are merged at frequency bin f_0:

$$X(f_0) = t_0 \cdot Ae^{j\phi_0} + (T - t_0) \cdot Ae^{j\phi_1} + X'_n(f_0) \tag{21}$$

Let $\alpha = \frac{t_0}{T}$, then

$$X(f_0) = A \cdot T(\alpha e^{j\phi_0} + (1-\alpha) \cdot e^{j\phi_1}) + X'_n(f_0) \tag{22}$$

In the ideal case without a phase jump ($\phi_0 = \phi_1$):

$$|X_{ideal(f_0)}| = A^2T^2(\alpha + (1-\alpha))^2 = A^2T^2 \tag{23}$$

In the practical case:

$$\begin{aligned} |X_{\text{real}}(f_0)|^2 &= A^2 \cdot T^2 |\alpha \cdot e^{j\phi_0} + (1-\alpha) \cdot e^{j(\phi_0+\Delta\phi)}|^2 \\ &= A^2 \cdot T^2 |\alpha + (1-\alpha) \cdot e^{j\Delta\phi}|^2 \end{aligned} \tag{24}$$

We define the power loss factor η as

$$\begin{aligned} \eta(\alpha, \Delta\phi) &= |\alpha + (1-\alpha) \cdot e^{j\Delta\phi}|^2 \\ &= \alpha^2 + (1-\alpha)^2 + 2\alpha(1-\alpha)\cos(\Delta\phi) \end{aligned} \tag{25}$$

In the worst case, when $\Delta\phi = \pi$ and $\alpha = \frac{1}{2}$, the phases are completely opposite and the energy cancels out, making the symbol impossible to demodulate.

The expected power loss is:

$$\begin{aligned} E[\eta] &= E[\alpha^2 + (1-\alpha)^2 + 2\alpha(1-\alpha)\cos(\Delta\phi)] \\ &= \alpha^2 + (1-\alpha)^2 \end{aligned} \tag{26}$$

$$E[\eta] = \int_0^1 [\alpha^2 + (1-\alpha)^2] d\alpha = \int_0^1 [2\alpha^2 - 2\alpha + 1] d\alpha = \frac{2}{3} \tag{27}$$

The average power loss is $10\log_{10}(\frac{2}{3}) \approx -1.76$. Phase misalignment introduces an average energy loss of 1.76 dB.

While the average SNR degradation of 1.76 dB may appear modest, this average value conceals the severe impact on packet-level performance. The critical issue lies in the variance of power loss across different chirps within a single LoRa packet. From Equation (18), the power loss factor $\eta(\alpha, \Delta\phi)$ varies significantly depending on the phase difference $\Delta\phi$ and the timing parameter α. When $\Delta\phi$ approaches π and α is near 0.5, the power loss can approach complete signal cancellation, far exceeding the average loss. A small number of symbol decoding errors can render the entire data packet unusable. For example, when CR is equal to 4/5 (error detection only, no correction), LoRa supports a maximum of 255 bytes per single data packet. In this case there may be hundreds of symbols, and the phase difference follows the random distribution. It is extremely unlikely that there will be no symbol with large energy loss among the hundreds of

symbols, so a single symbol error can invalidate the entire packet.

In a typical LoRa packet containing multiple chirps, each chirp experiences different phase misalignment conditions due to varying symbol values and random phase offsets. The distribution of η has a variance of:

$$Var[\eta] = E[\eta^2] - (E[\eta])^2 \tag{28}$$

For uniformly distributed α and $\Delta\phi$, the worst-case chirps can experience SNR degradation of 10-20 dB or more, making them undecodable even when the average degradation is only 1.76 dB. Since LoRa packets require successful decoding of all symbols for packet recovery, even a single severely degraded chirp can cause the entire packet to fail. This explains why the seemingly small average degradation has a disproportionately large impact on packet error rates, particularly in weak signal scenarios where the margin for error is minimal.

6.3 Noise Power Amplification

To address the energy loss caused by phase misalignment, oversampling is employed to separate the peaks generated by the front and rear signal segments. This approach prevents the automatic peak merging that occurs under standard sampling rates and avoids energy loss caused by phase misalignment. The next step is to fuse the energy of these two peaks. The essence of fusion is to superimpose the FFT spectrum results of the front and rear segments of the dechirp signal. During this process, energy can be effectively superimposed, but we notice that noise is also superimposed, which limits the SNR improvement after peak merging.

Let N represent the entire spectrum. N_1 represents the front spectrum in the range (-B,0), N_2 represents the rear spectrum in the range (0,B), and the noise spectrum and the two segment spectra follow a complex normal distribution. For a complex normal distribution, the real and imaginary parts are independent Gaussian random variables. Both real and imaginary parts have the same variance, defined as $\sigma^2/2$, and the total power of the complex noise is σ^2. Therefore, the noise can be expressed as:

$$N \sim CN(0, \sigma^2) \tag{29}$$

$$N_1 \sim CN(0, \sigma^2), \quad N_2 \sim CN(0, \sigma^2) \tag{30}$$

There are two algorithms for implementing peak merging. One is simple spectral magnitude extraction, which eliminates the effects of phase, allowing signal energy to be superimposed without loss. However, noise is also greatly enhanced.

The magnitude of noise follows a Rayleigh distribution, which can be expressed as:

$$|N| \sim \text{Rayleigh}(\sigma), \quad |N_1| \sim \text{Rayleigh}(\sigma), \quad |N_2| \sim \text{Rayleigh}(\sigma) \tag{31}$$

$$E(|N|) = \sigma\sqrt{\frac{\pi}{2}}, \quad Var(|N|) = \sigma^2\left(2 - \frac{\pi}{2}\right) \tag{32}$$

$$E(|N|^2) = 2\sigma^2 \tag{33}$$

Then N_1 and N_2 will be merged, which follow:

$$E(|N_1| + |N_2|) = 2\sigma\sqrt{\frac{\pi}{2}} \tag{34}$$

$$Var(|N_1| + |N_2|) = 2\sigma^2(2 - \frac{\pi}{2}) \tag{35}$$

The power of the merged noise can be expressed:

$$P_{noise} = Var(|N_1| + |N_2|) + E(|N_1| + |N_2|)^2 = 4 \cdot \sigma^2 \tag{36}$$

For ChirpSmoother, which uses an ideal chirp without a frequency jump, the SNR can be defined as:

$$SNR_0 = 10 \cdot log\frac{A^2}{\sigma^2} = 20 \cdot log\frac{A}{\sigma} \tag{37}$$

And the SNR of peak merging using spectral magnitude extraction, can be defined as:

$$SNR_1 = 10 \cdot log\frac{A^2}{4 \cdot \sigma^2} = 20 \cdot log\frac{A}{4 \cdot \sigma} \tag{38}$$

The SNR degradation can be expressed as :

$$\Delta SNR = SNR_1 - SNR_0 = -20 \cdot log4 \approx -6.02 \text{ dB} \tag{39}$$

The peak merging using spectral magnitude extraction results in 6.02 dB of energy loss, which is a disaster for wireless communications.

The second approach involves phase alignment, which we can prove is equivalent to the first method in terms of performance. Phase alignment requires calculating the phase difference between the front and rear signal segments, followed by phase rotation to achieve coherence. However, computing the phase difference relies on exhaustive phase search algorithms.

When the searched phase approaches the true phase difference, the resulting signal peak becomes higher. Therefore, the algorithm searches for the bin with the highest peak value, considering it as the coherent superposition result after phase alignment. However, during the search process, noise-induced phase alignments are also discovered. After coherent superposition, the noise level is also enhanced.

We can derive that while signal energy can be perfectly fused under ideal phase alignment conditions, practical phase search precision limitations prevent exhaustive enumeration of all possible phases, resulting in minor energy losses after

merging. The same principle applies to noise. For noise components, the magnitude summation of front and rear segment spectra is equivalent to first performing phase alignment on noise and then taking the magnitude. Therefore, the two approaches are fundamentally equivalent.

Mathematically, we can express this equivalence as:

$$|N_1| + |N_2| \equiv |N_1 \cdot e^{j\Delta\theta} + N_2| \tag{40}$$

where $\Delta\theta$ represents the optimal phase rotation angle found through exhaustive search.

In conclusion, peak merging schemes, whether based on phase alignment or magnitude-based combination, are fundamentally equivalent in nature. While these approaches can address the destructive interference caused by phase misalignment, they introduce substantial energy losses of approximately 6 dB due to noise amplification during the merging process.

This significant SNR degradation occurs because peak merging inherently combines noise from both signal segments along with the desired signal components. Although phase alignment techniques can theoretically achieve perfect signal coherence, the practical noise amplification fundamentally limits their effectiveness in weak signal scenarios. ChirpSmoother eliminates this limitation entirely by avoiding frequency jumps and the subsequent need for peak merging, thereby maintaining the original signal integrity without introducing additional noise amplification.

7 Implementation and Evaluation

7.1 Implementation Platform

We implement ChirpSmoother on commercial LoRa hardware nodes and conduct comprehensive evaluations to validate its effectiveness. We deploy Semtech SX1276 LoRa [25] nodes as transmitters and utilize USRP X410 Software Defined Radios [9] as LoRa gateways for signal reception and processing. The experimental setup operates at a center frequency of 915 MHz with spreading factors ranging from SF7 to SF12 (default SF7) and bandwidth configurations of 125 kHz, 250 kHz, and 500 kHz (default 125 kHz).

We conduct experiments in both indoor and outdoor environments to evaluate ChirpSmoother's performance across diverse propagation conditions. The indoor environment consists of office spaces filled with desks, computers, and other equipment that create complex multipath propagation. The outdoor environment spans university campus areas with varying line-of-sight conditions and terrain characteristics. To obtain signals at different SNR levels for comprehensive evaluation, we synthesize signals with controlled power levels.

7.2 Baseline Methods for Comparison

We compare ChirpSmoother against several state-of-the-art approaches representing different enhancement strategies and traditional approaches:

Highest FFT Bin (HFFT): This baseline method employs oversampling to address frequency-wrapped chirps. After dechirp operation and FFT transformation, the signal energy forms two distinct peaks in the frequency domain. HFFT simply selects the peak with the highest energy magnitude for symbol demodulation, effectively discarding the energy from the secondary peak.

Multiple FFT (MFFT): MFFT [24] decomposes the samples into several signals with sampling rate B, corrects for phase differences between them, and sums their FFT outputs.

LoRaPHY: LoRaPHY [32] addresses the phase misalignment in oversampling scenarios through peak merging techniques. The approach includes two primary methods: (1) coarse-grained phase alignment (CPA) that converts frequency-domain components from vectors (with both magnitude and phase) to scalars (magnitude only), then performs spectral fusion by summing the magnitudes of the front and rear spectrum segments, and (2) fine-grained phase alignment (FPA) that searches for the phase difference between the two signal segments through exhaustive enumeration, followed by phase rotation to align the front and rear segments before superposition. As demonstrated in our theoretical analysis in Section 6.3, these two methods are fundamentally equivalent in performance.

NELoRa: NELoRa [16] employs neural-enhanced demodulation technology using dual-channel spectrograms that combine amplitude and phase information as input features. The method utilizes a dual Deep Neural Network (DNN) architecture: the first DNN serves as a noise filter to recover clean chirp patterns, simulating real spectrograms under high SNR conditions; the second DNN performs decoding classification based on spectrograms, exploring multi-dimensional feature spaces to achieve ultra-low SNR demodulation. However, this approach requires substantial computational overhead and extensive model training datasets.

LoRaTrimmer: LoRaTrimmer [6] addresses frequency jump effects in real LoRa signals through adaptive chirp trimming design. The method adaptively adjusts FFT perception ranges for different frequencies, trimming the FFT window to the actual signal duration, thereby reducing accumulation of extraneous noise. While this approach mitigates some noise amplification effects, it does not fundamentally address the phase misalignment issues inherent in frequency-wrapped chirps.

ChirpSmoother: Our proposed ChirpSmoother method enhances LoRa weak signal decoding performance by eliminating frequency jumps entirely. Unlike existing methods

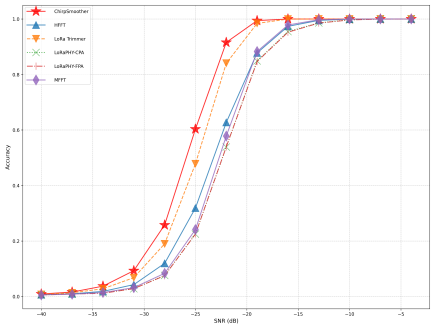


Figure 7: Decoding performance in SF8

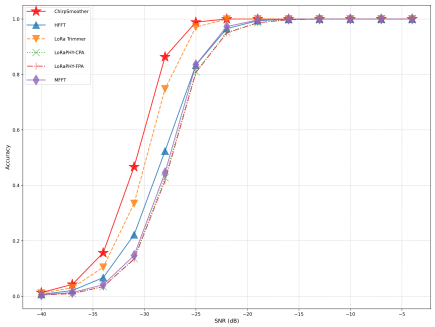


Figure 8: Decoding performance in SF10

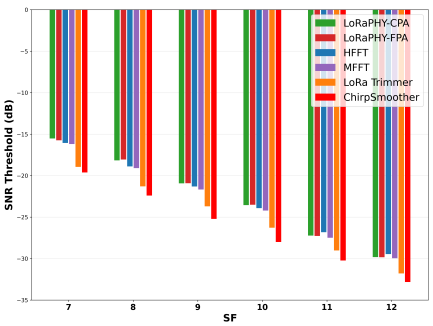


Figure 9: SNR across SF

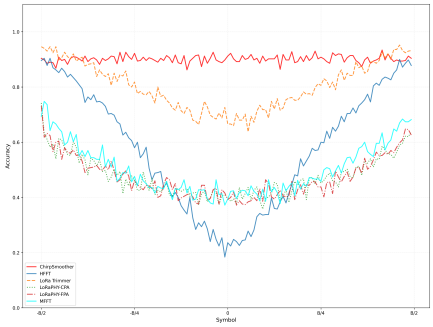


Figure 10: Decoding Performance Across Symbol Variations

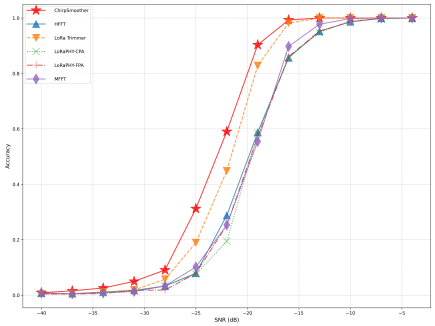


Figure 11: Decoding performance in 250 kHz bandwidth

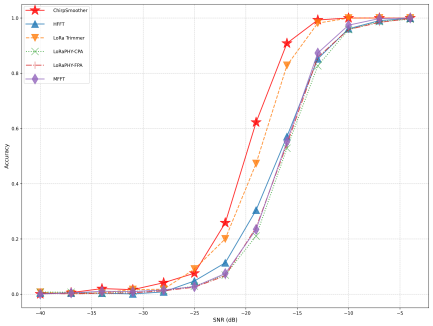


Figure 12: Decoding performance in 500 kHz bandwidth

that focus on peak merging or phase alignment, ChirpSmoother fundamentally resolves the frequency discontinuity problem in chirp signals. This approach avoids the noise amplification issues that plague traditional peak merging techniques while maintaining computational efficiency suitable for real-time processing.

7.3 Overall Performance in Different SF and BW

We comprehensively evaluated ChirpSmoother's performance against existing methods under varying SNR conditions, spreading factors, and bandwidths. Fig. 7 and Fig. 8 present the decoding performance results for SF=7 and SF=10, respectively. Using 90% accuracy to determine the minimum operational SNR (Fig. 9), ChirpSmoother consistently demonstrates superior decoding performance across all tested scenarios. We also evaluated ChirpSmoother's performance under different bandwidths, as shown in Fig. 11 and Fig. 12.

Several key observations emerge from our experimental results. First, the two peak merging variants, LoRaPHY-CPA and LoRaPHY-FPA, exhibit nearly identical decoding performance, empirically validating our theoretical analysis. Second, and perhaps more surprisingly, HFFT outperforms both peak merging schemes despite discarding half of the signal energy. This counterintuitive result confirms that the noise amplification penalty from peak merging exceeds the potential gains from energy consolidation, effectively demonstrating the severity of noise superposition effects in practical deployments.

In terms of quantitative improvements, ChirpSmoother achieves 3.01-4.49 dB SNR gains compared to peak merging schemes (LoRaPHY-CPA), 3.38-4.09 dB improvement over HFFT, and 0.68-1.72 dB enhancement compared to LoRaTrimmer. These substantial performance gains demonstrate ChirpSmoother's effectiveness in enhancing weak signal reception while maintaining computational efficiency suitable for IoT deployments.

7.4 Performance across Different Symbols

We observe that standard LoRa signals, regardless of the decoding method employed, exhibit a consistent pattern where different chirps demonstrate varying decoding performance. Specifically, chirps with initial frequencies approaching 0 or B achieve optimal decoding performance, while chirps with initial frequencies near B/2 experience significant performance degradation across all decoding methods, as shown

in Fig. 10. This phenomenon occurs because standard LoRa transmission splits signals into two discontinuous segments: the first segment spans (f_0, B) and the second segment covers (0, f_0).

This performance variation fundamentally stems from chirp energy being divided across two segments. The HFFT method selects only the bin with the highest peak as the signal bin, resulting in significant performance variations as the initial frequency changes, as clearly demonstrated in our results.

Other decoding methods attempt peak merging by combining energy from both signal segments, but this approach introduces additional challenges. As shown in Fig.10, both LoRaPHY-CPA and LoRaPHY-FPA exhibit varying performance levels across symbols. During peak merging, noise consolidation occurs, amplifying the overall noise level. Originally, noise is distributed across (-B, B) with relatively low overall levels. After peak merging, noise from both segments (-B, 0) and (0, B) is combined, further intensifying the noise level.

The results clearly demonstrate that when chirp energy distribution between the two signal segments is uneven, peak merging algorithms like LoRaPHY perform significantly worse than HFFT. Additionally, the phase search-based FPA method achieves performance similar to CPA under low SNR conditions. Phase search algorithms can coherently combine both signal and noise components during the search process. Consequently, both noise and signal experience coherent superposition during the search, amplifying the noise level. This explains why CPA and FPA demonstrate similar performance under low SNR conditions in our results.

Therefore, both phase search-based methods and magnitude-based approaches face the fundamental challenge of noise superposition. As illustrated in Fig.10, peak merging algorithms demonstrate inferior decoding performance compared to HFFT for most symbols, indicating that the signal gain achieved through peak merging typically cannot compensate for the noise superposition losses in most scenarios.

7.5 Computational Cost

ChirpSmoother's computational approach closely resembles traditional LoRa decoding processes. After the dechirp operation, an FFT is performed to identify the maximum frequency component, which is identical to the HFFT computational methodology. Theoretically, ChirpSmoother's computational overhead equals that of HFFT. For LoRaTrimmer, we used the open-source decoding method from the paper, which has computational complexity of $O(n^2)$, to illustrate the cost of higher complexity. We conducted symbol-level decoding performance measurements on a MacBook (M1 Pro chip). Table 1 presents the decoding results, which align with our theoretical analysis.

Table 1: Symbol Decoding Time Comparison (ms)

Method	SF=7	SF=8	SF=9	SF=10	SF=11	SF=12
LoRaPHY-FPA	2.11	4.61	9.74	20.47	48.02	103.99
LoRaPHY-CPA	0.91	1.95	5.27	9.36	22.58	51.45
MFFT	0.11	0.13	0.19	0.29	0.53	0.98
HFFT	0.04	0.06	0.11	0.20	0.42	0.83
LoRaTrimmer	0.56	2.26	10.36	36.78	145.92	566.17
ChirpSmoother	0.06	0.06	0.15	0.20	0.41	0.87

The extremely low computational complexity enables broader compatibility and deployment scenarios. As a representative LPWAN technology, LoRa's widespread adoption stems from its adherence to LPWAN characteristics: low power consumption and long-range communication. ChirpSmoother's minimal computational complexity and excellent weak signal decoding performance align perfectly with current LoRa LPWAN attributes. Algorithms with higher computational complexity fail to satisfy LPWAN requirements and have limited application scenarios.

This computational efficiency is particularly crucial for edge deployment scenarios where processing resources are constrained, like tinySDR [13]. Unlike deep learning-based approaches that require substantial computational overhead and extensive training datasets, ChirpSmoother maintains the simplicity and efficiency essential for IoT applications while delivering significant performance improvements.

8 Related Work

Multi-gateway collaborative enhancement. Multi-gateway systems leverage spatial diversity by combining signals from multiple distributed receivers to improve decoding performance. For instance, Charm [5] coherently combines raw physical layer samples to boost SNR, while Chime [10] focuses on selecting optimal frequencies. OPR [1] recovers corrupted packets through cloud-based bit error correction using spatial diversity and RSSI analysis. Despite their effectiveness, these methods are hampered by the practical difficulty of achieving the required sample-level time synchronization (MALoRa [14], PCube [31]) and the substantial network overhead from transmitting large volumes of data. Nephalai [19] jointly demodulates samples and uses compressive sensing to reduce LPWAN bandwidth.

Transmitter-Side Collaborative Enhancement. Transmitter-side approaches coordinate multiple transmissions to enhance signal strength at the source. Choir [8] achieves this through constructive interference from multiple co-located nodes transmitting in unison, while XCopy [30] coherently

combines retransmitted packets from a single device for temporal diversity. Falcon [28] employs a technique that selectively interferes with other LoRa transmissions to enable data transmission over low SNR links. Although these techniques avoid receiver-side complexity, they impose significant costs in terms of hardware deployment and energy consumption, contradicting the low-power ethos of IoT networks.

Signal Processing Optimization. This category of research focuses on improving the fundamental demodulation algorithms without requiring additional hardware. Methods like LoRaPHY [32] and LoRaTrimmer introduce advanced phase alignment and adaptive FFT windowing, respectively, to better consolidate signal energy. Others use symbol repetition (Ostinato [33]) or channel analysis (Chime [10]). However, these approaches primarily mitigate the effects of the frequency jump; they cannot escape the fundamental tradeoff where merging signal peaks also amplifies the underlying noise.

Deep Learning-Based Enhancement. Recently, deep learning has been applied to augment the traditional LoRa signal processing pipeline. NELoRa [16] proposes a DNN-based demodulator to operate at lower SNRs, while DeepLoRa [20] uses LSTMs for path loss modeling, and SRLoRa [7] applies DNNs to multi-gateway scenarios. Although powerful, these methods suffer from major practical drawbacks: they demand significant computational resources unsuitable for constrained edge devices, require large and diverse training datasets, and often fail to generalize across different configurations without retraining.

9 Conclusion

In this paper, we analyze how frequency jumps limit LoRa decoding performance. Previous approaches have focused on minimizing the impact of frequency jumps, including phase misalignment, energy dispersion, and noise enhancement. We leverage the supporting characteristics of commercial LoRa nodes to completely resolve the frequency jump issue. Compared to state-of-the-art methods, our approach achieves a signal-to-noise ratio improvement of 0.68-1.72 dB, and compared to peak merging methods, we achieve an improvement of 3.01-4.27 dB. Moreover, our computational complexity is extremely low, making it suitable for a wide range of edge nodes with limited computational capabilities. We also address the unique cross-channel interference problem at the receiver end.

References

[1] Artur Balanuta, Nuno Pereira, Swarun Kumar, and Anthony Rowe. 2020. A cloud-optimized link layer for low-power wide-area networks. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 247–259.

[2] Albert Berni and WO Gregg. 2003. On the utility of chirp modulation for digital signaling. *IEEE Transactions on Communications* 21, 6 (2003), 748–751.

[3] Tusher Chakraborty, Heping Shi, Zerina Kapetanovic, Bodhi Priyantha, Deepak Vasisht, Binh Vu, Parag Pandit, Prasad Pillai, Yaswant Chabria, Andrew Nelson, et al. 2023. Whisper: IoT in the TV white space spectrum. *GetMobile: Mobile Computing and Communications* 26, 4 (2023), 32–35.

[4] Silvia Demetri, Marco Zúñiga, Gian Pietro Picco, Fernando Kuipers, Lorenzo Bruzzone, and Thomas Telkamp. 2019. Automated estimation of link quality for LoRa: A remote sensing approach. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*. 145–156.

[5] Adwait Dongare, Revathy Narayanan, Akshay Gadre, Anh Luong, Artur Balanuta, Swarun Kumar, Bob Iannucci, and Anthony Rowe. 2018. Charm: Exploiting geographical diversity through coherent combining in low-power wide-area networks. In *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 60–71.

[6] Jialuo Du, Yunhao Liu, Yidong Ren, Li Liu, and Zhichao Cao. 2024. Loratrimmer: Optimal energy condensation with chirp trimming for lora weak signal decoding. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 1104–1118.

[7] Jialuo Du, Yidong Ren, Zhui Zhu, Chenning Li, Zhichao Cao, Qiang Ma, and Yunhao Liu. 2023. Srlora: Neural-enhanced lora weak signal decoding with multi-gateway super resolution. In *Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*. 270–279.

[8] Rashad Eletreby, Diana Zhang, Swarun Kumar, and Osman Yağan. 2017. Empowering low-power wide area networks in urban settings. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 309–321.

[9] EttusResearch. 2024. USRP X410. https://www.ettus.com/all-products/usrp-x410 Accessed 11-Aug-2024.

[10] Akshay Gadre, Revathy Narayanan, Anh Luong, Anthony Rowe, Bob Iannucci, and Swarun Kumar. 2020. Frequency Configuration for {Low-Power}{Wide-Area} Networks in a Heartbeat. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. 339–352.

[11] Akshay Gadre, Fan Yi, Anthony Rowe, Bob Iannucci, and Swarun Kumar. 2020. Quick (and dirty) aggregate queries on low-power WANs. In *2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, Vol. 2020.

[12] Jetmir Haxhibeqiri, Eli De Poorter, Ingrid Moerman, and Jeroen Hoebeke. 2018. A survey of LoRaWAN for IoT: From technology to application. *Sensors* 18, 11 (2018), 3995.

[13] Mehrdad Hessar, Ali Najafi, Vikram Iyer, and Shyamnath Gollakota. 2020. {TinySDR}:{Low-Power}{SDR} platform for {Over-the-Air} programmable {IoT} testbeds. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. 1031–1046.

[14] Ningning Hou, Xianjin Xia, and Yuanqing Zheng. 2023. Don't miss weak packets: Boosting LoRa reception with antenna diversities. *ACM Transactions on Sensor Networks* 19, 2 (2023), 1–25.

[15] Oana Iova, Amy L Murphy, Gian Pietro Picco, Lorenzo Ghiro, Davide Molteni, Federico Ossi, Francesca Cagnacci, et al. 2017. LoRa from the city to the mountains: Exploration of hardware and environmental factors. In *International Conference on Embedded Wireless Systems and Networks (EWSN) 2017, Uppsala, sweden, 20-22 February 2017*. Uppsala University, 317–322.

[16] Chenning Li, Hanqing Guo, Shuai Tong, Xiao Zeng, Zhichao Cao, Mi Zhang, Qiben Yan, Li Xiao, Jiliang Wang, and Yunhao Liu. 2021.

NELoRa: Towards ultra-low SNR LoRa communication with neural-enhanced demodulation. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 56–68.

[17] Chenning Li, Yidong Ren, Shuai Tong, Shakhrul Iman Siam, Mi Zhang, Jiliang Wang, Yunhao Liu, and Zhichao Cao. 2024. Chirptransformer: Versatile lora encoding for low-power wide-area iot. In *Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services*. 479–491.

[18] Jansen C Liando, Amalinda Gamage, Agustinus W Tengourtius, and Mo Li. 2019. Known and unknown facts of LoRa: Experiences from a large-scale measurement study. *ACM Transactions on Sensor Networks (TOSN)* 15, 2 (2019), 1–35.

[19] Jun Liu, Weitao Xu, Sanjay Jha, and Wen Hu. 2020. Nephalai: towards LPWAN C-RAN with physical layer compression. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–12.

[20] Li Liu, Yuguang Yao, Zhichao Cao, and Mi Zhang. 2021. Deeplora: Learning accurate path loss model for long distance links in lpwan. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.

[21] LoRa Alliance. 2020. A technical overview of LoRa and LoRaWAN. https://lora-alliance.org/resource-hub/what-lorawanr. Accessed: 2020-11-19.

[22] Spyridon Mitropoulos, Vasilios A Orfanos, Dimitrios Rimpas, and Ioannis Christakis. 2023. LoRa Radius Coverage Map on Urban and Rural Areas: Case Study of Athens' Northern Suburbs and Tinos Island, Greece. *Engineering Proceedings* 58, 1 (2023), 19.

[23] Yidong Ren, Wei Sun, Jialuo Du, Huaili Zeng, Younsuk Dong, Mi Zhang, Shigang Chen, Yunhao Liu, Tianxing Li, and Zhichao Cao. 2024. Demeter: Reliable cross-soil lpwan with low-cost signal polarization alignment. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 230–245.

[24] Vincent Savaux. 2022. A low-complexity demodulation for oversampled LoRa signal. *International Journal of Mobile Network Design and Innovation* 10, 3 (2022), 148–153.

[25] Semtech. 2024. Semtech SX1276. https://www.semtech.com/products/wireless-rf/lora-connect/sx1276 Accessed 11-Aug-2024.

[26] Ritesh Kumar Singh, Priyesh Pappinisseri Puluckul, Rafael Berkvens, and Maarten Weyn. 2020. Energy consumption analysis of LPWAN technologies and lifetime estimation for IoT application. *Sensors* 20, 17 (2020), 4794.

[27] Wenxin Tang, Honggang Zhao, Sitong Chen, Aoyang Li, Wenjing Yang, Wei Cheng, Yong Li, and Limeng Dong. 2025. Measurement of LoRa signal propagation in urban areas utilizing aerial gateway and ground gateway. *Scientific Data* 12, 1 (2025), 1464.

[28] Shuai Tong, Zilin Shen, Yunhao Liu, and Jiliang Wang. 2021. Combating link dynamics for reliable lora connection in urban settings. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 642–655.

[29] Shuai Tong, Jiliang Wang, Jing Yang, Yunhao Liu, and Jun Zhang. 2023. Citywide lora network deployment and operation: Measurements, analysis, and implications. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*. 362–375.

[30] Xianjin Xia, Qianwu Chen, Ningning Hou, Yuanqing Zheng, and Mo Li. 2023. Xcopy: Boosting weak links for reliable lora communication. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.

[31] Xianjin Xia, Ningning Hou, Yuanqing Zheng, and Tao Gu. 2023. PCube: Scaling LoRa concurrent transmissions with reception diversities. *ACM Transactions on Sensor Networks* 18, 4 (2023), 1–25.

[32] Zhenqiang Xu, Shuai Tong, Pengjin Xie, and Jiliang Wang. 2023. From demodulation to decoding: Toward complete lora phy understanding and implementation. *ACM Transactions on Sensor Networks* 18, 4 (2023), 1–27.

[33] Zhenqiang Xu, Pengjin Xie, Jiliang Wang, and Yunhao Liu. 2022. Ostinato: Combating lora weak links in real deployments. In *2022 IEEE 30th International Conference on Network Protocols (ICNP)*. IEEE, 1–11.

[34] Kang Yang, Yuning Chen, Tingruixiang Su, and Wan Du. 2023. Link quality modeling for lora networks in orchards. In *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*. 27–39.

[35] Xinghua Yang, Zheyu Liu, Kechao Tang, Xunzhao Yin, Cheng Zhuo, Qi Wei, and Fei Qiao. 2023. Breaking the energy-efficiency barriers for smart sensing applications with "Sensing with Computing" architectures. *Science China Information Sciences* 66, 10 (2023), 200409.