Lab 7 - Digital Modulation: Frame Synchronization [1]

ECE531 – Software Defined Radio

Spring 2025

Daniel Gallagher, danielgallagher@arizona.edu
Department of Electrical & Computer Engineering
University of Arizona, Tucson AZ 85721

[1] Acknowledgement: This laboratory assignment is based on material provided by the *Software-Defined Radio for Engineers* book course material which was previously taught at Worcester Polytechnic Institute.

1 Overview and Objectives

This laboratory will introduce the concept of frame synchronization between transmitting and receiving nodes. Specifically, a simplified error model will be discussed along with efficient methods with various implementations.

Note: A summary of the Matlab Communication Toolbox functions are available at: https://www.mathworks.com/help/comm/synchronization-and-receiver-design.html

2 Introduction to Frame Synchronization

In previous labs, we have discussed frequency correction, timing compensation, and matched filtering. The final aspect of synchronization is frame synchronization. At this point, it is assumed that the available samples represent single symbols and are corrected for timing, frequency, and phase offsets. However, since in a realistic system the start of a frame will still be unknown, we need to perform an additional correction. We demonstrate this issue visually in Figure 1, which contains a sample synchronized frame with an unknown offset of p samples.

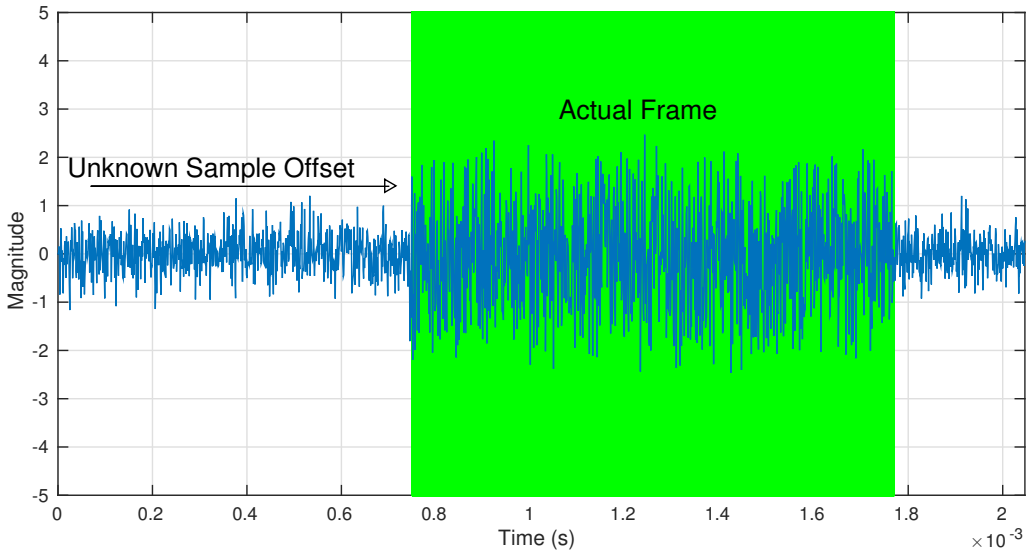

Figure 1: Example received frame in AWGN with an unknown sample offset.

Once we have an estimate $\hat{p}$, we can extract data from the desired frame, demodulate to bits, and perform any additional channel decoding or source decoding originally applied to the signal. There are various ways to accomplish this estimation, but the implementation outlined in this lab is based around cross-correlation.

3 Frame Synchronization

A common method of determining the start of a given frame is with the use of markers, even in wired networking. However, in the case of wireless signals, this problem becomes more difficult as seen in Figure 1 which actually uses a marker. Due to the high degree of noise in the signal, specifically designed preamble sequences are appended to frames before modulation. Such sequences are typically known exactly at the receiver and have certain qualities that make frame estimation accurate.

In Figure 2, we outline a typical frame ordering containing: preamble, header, and payload data. Header and payloads are unknown at the receiver, but will maintain some structure so they can be decoded correctly.

Figure 2: Common frame structure of wireless packet with preamble followed by header and payload data.

Before we discuss the typical sequences utilized, we will introduce a technique for estimating the start of a known sequence at an unknown sample position. Let us consider a set of N different binary sequences y_n, where $n \in [1, ..., N]$, each of length L. Given an additional binary sequence x, we want to determine how similar x is to the existing N sequences. The cross-correlation provides us with the appropriate estimate, which we perform as:

$$C_{xy}(k) = \sum_{m} x^*(m) y_n(m+k), \tag{1}$$

which is identical to a convolution without a time reversal on the second term. When $x = y_n$ for a given n, C_{xy} will be maximized compared with the other $n-1$ sequences, and produce a peak at the L^{th} index at least. We can use this concept to help build our frame start estimator, which, as discussed, will contain a known sequence called the preamble.

Common sequences utilized in preambles for narrowband communications are Barker Codes [2]. Barker Codes are utilized since they have unique auto-correlation properties that have minimal or ideal off-peak correlation. Specifically, such codes or sequences $a(i)$ have autocorrelation functions defined as:

$$c(k) = \sum_{i=1}^{N-k} a(i)a(i+k), \tag{2}$$

such that:

$$|c(v)| \leq 1, \quad 1 \leq v < N. \tag{3}$$

However, only nine sequences are known for $N \in [1, 2, 3, 4, 5, 7, 11, 13]$, provided in Table 1. We provide a visualization of these auto-correlations in Figure 3 for a select set of lengths. As the sequence becomes longer, the central peak becomes more pronounced. For communication

systems, we typically append multiple such codes together to produce longer sequences for better performance, as well as for other identifications.

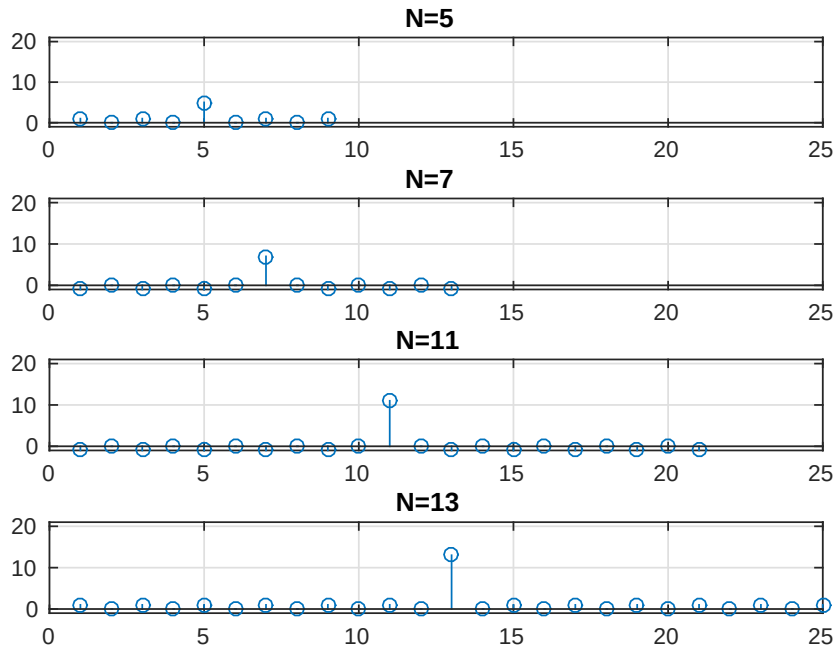

Figure 3: Barker code autocorrelations for different lengths.

Table 1: Barker Codes From `comm.BarkerCode`

N	Code
2	$-1, +1$
3	$-1, -1, +1$
4	$-1, -1, +1, -1$
5	$-1, -1, -1, +1, -1$
7	$-1, -1, -1, +1, +1, -1, +1$
11	$-1, -1, -1, +1, +1, +1, -1, +1, +1, -1, +1$
13	$-1, -1, -1, -1, -1, +1, +1, -1, -1, +1, -1, +1, -1$

Using these codes, we have implemented a small example to show how a barker sequence $a(k)$ can be used to locate sequences in a larger set of data $r(k)$, which we have provided in `lab7part1.m`. In this example, we insert a barker code within a larger random sequence at an unknown position p, similar to our original error model in Section 2, shown as the top plot in Figure 4.

A cross-correlation is performed using MATLAB®'s `xcorr` function, providing the lower plot in Figure 4. The cross-correlation will be of length $2L_r - 1$, where L_r is the length of r. Since

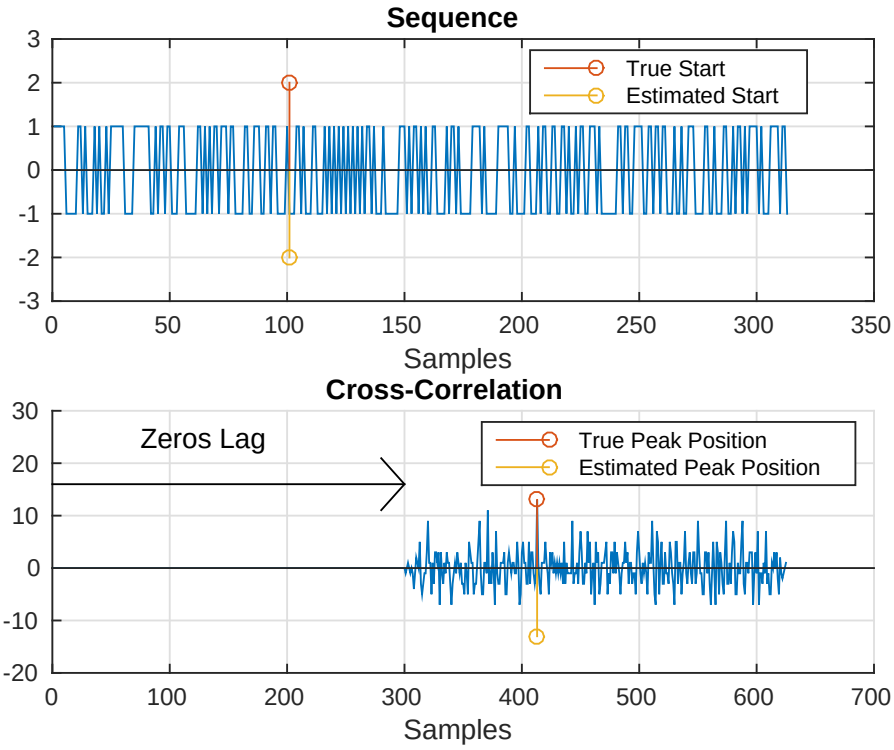

Figure 4: Sample Barker Code implementation for sequence offset determination.

`xcorr` will pad zeros to a so its length is equal to L_r [3], this will result in at least $L_r - L_a$ zeros to appear in the correlation where L_a is the original length of a. From Figure 3, we know that the peak will appear at L_a samples from the start of the sequence. Taking this into account, we can directly determine the offset position of our desired sequence:

$$\hat{p} = \underset{k}{\text{argmax}}\ C_{ra}(k) - L_r, \tag{4}$$

which is what we observe from our estimates in the top plot of Figure 4.

Now that we have a method for estimating the start of a frame, let us consider a slightly simpler problem: Can we determine that a frame exists in the correlation? This can be useful if we wish to handle data in smaller pieces rather than working with complete frames, or as a mechanism for determining if a channel is occupied.

When we consider signal detection, we typically define this feature as a power sensitivity, or the minimum received power at the receiver to be detected. However, this sensitivity will be based on some source waveform and cannot be generalized. Therefore, such a value should never be given on its own, unless provided with respect to some standard transmission. Even when considering formal methods of detection theory, such as Neyman-Pearson or Bayesian approaches, you must have some knowledge or reference to the source signal [4]. The receiver sensitivity requirement for 802.11ac specifically is defined as the minimum received signal power to maintain a packet error rate of 10% for a given modulation and coding scheme [1].

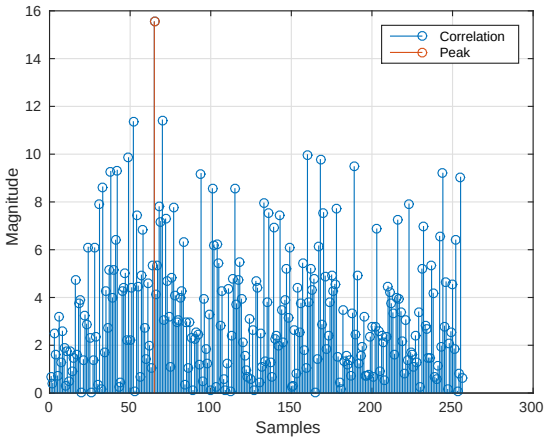

Figure 5: Correlation without signal present.

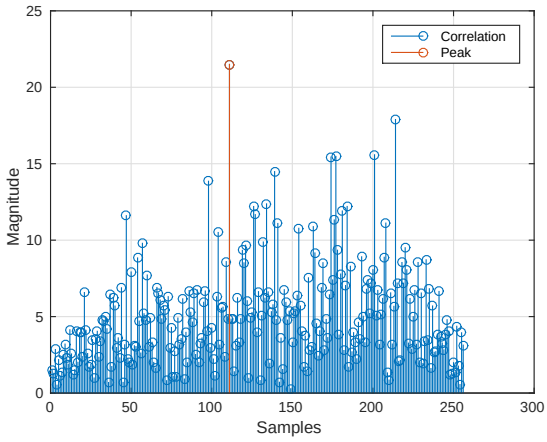

Figure 6: Correlation with signal present.

When we consider the implementation consequences of detecting a signal, our design becomes more complicated than the example in `lab7part1.m`. In the most basic sense, detection becomes a thresholding problem for our correlator. Therefore, the objective becomes determining a reference or criteria for validating a peak, which can vary significantly over time depending on channel noise and the automatic gain control of the Pluto.

Even in simulations, appropriate thresholding becomes non-trivial, which we can demonstrate with Figures 5 and 6. In these figures, the peak appears larger relative to the rest of the correlation in the case where no frame exists in the receive signal, compared to the condition when a frame exists. Therefore, an implementation that performs well should handle such conditions and operate regardless of the input scaling.

In `lab7part2.m`, we have enhanced the baseline bit generation to include support for ASCII characters to bits and back to ASCII. Included as well is an example demodulation and packet error rate measurement. These will be used to determine the overall performance of your receiver design and frame synchronizer.

4 Questions

1. Based on `lab7part1.m`, implement an alternative scheme that utilizes the `filter` function for estimation rather than `xcorr`. (`filter` is generally much faster than `xcorr` in interpreted MATLAB.) Provide a benchmark using `tic` and `toc` for their relative performance for different sequence lengths.
2. Based on the template provided in `lab7part2.m` and the example in `lab7part1.m`, implement a frame synchronization scheme.
3. Test your frame synchronization implementation over the SNR range $[0, 10]$ dB and provide the resulting plot for detection probability and packet error rate.

5 Lab Report Preparation & Submission Instructions

Laboratory reports should be uploaded as PDF or Word documents outside of a zip archive and formatted as follows:

- Cover page: includes course number, laboratory title, name, submission date.
- Table of Contents, List of Tables, List of Figures. (Optional)
- Lab Results:
  - Introduction to the laboratory experiment, including a brief description of the objectives and goals.
  - Detailed explanation of the laboratory experiment, including the design, implementation, and testing of the system.
  - Results and discussion of the laboratory experiment, including captured outputs, observations, and responses to laboratory questions.
  - Conclusions to the overall lab that discuss meaningful lessons learned and other takeaways from the assignment. (Important)
- Upload source files with report submission. You may additionally include select code source listings in your report to highlight techniques used.
  - Note: Python files autogenerated from GNURadio do not need to be uploaded if the .grc files are included.

Remember to write your laboratory report in a detailed and descriptive manner, explaining your experience and observations in such a way that it provides the reader with some insight as to what you have accomplished. Furthermore, please include images and outputs wherever possible in your laboratory report document.

References

[1] IEEE Standard for Information technology– Telecommunications and information exchange between systemsLocal and metropolitan area networks– Specific requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz. *IEEE Std 802.11ac-2013 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012, IEEE Std 802.11aa-2012, and IEEE Std 802.11ad-2012)*, pages 1–425, Dec 2013.

[2] RH Barker. Group synchronizing of binary digital sequences. *Communication theory. Butterworth, London*, pages 273–287, 1953.

[3] The MathWorks Inc. xcorr. [Online]: https://www.mathworks.com/help/signal/ref/xcorr.html, 2017.

[4] Steven M Kay. Fundamentals of statistical signal processing, vol. ii: Detection theory. *Signal Processing. Upper Saddle River, NJ: Prentice Hall*, 1998.