Journal od SDR Communications (2025), 1–7
doi:02.09

Progress Report

Working Principles of an M-PSK communication system on SDR

Mihir Divyansh , Dhanush Nayak , Kaustubh Khachane , Siddhartha Vemula

Abstract

This guide is to aid attempts to establish digital communications on a SDR, using PSK based modulation. We provide the math and intuition behind particular design decisions at the transmitter and receiver end, in practical scenarios.

Keywords: SDR, PSK

1. Scrambling in Digital Communications

Raw binary sequences frequently contain undesirable structures such as long runs of identical symbols. These patterns can impair the operation of synchronization circuits, degrade spectral efficiency, and hinder adaptive receiver algorithms.

- In wireless channels, signals travel over multiple paths, causing Inter-Symbol Interference (ISI). Equalizers are filters that adapt to undo this effect. Long streams ensure that the equalizer does not learn.
- Repetitive or constant input sequences concentrate energy at discrete frequencies,producing a line spectrum that is spectrally inefficient.

1.1. Implementation Mechanism:LFSR

The most common mechanism used to implement scramblers is the **Linear Feedback Shift Register (LFSR)**. An LFSR is a sequential circuit consisting of a shift register whose input is formed by a xor function of selected stages of the register.

The properties of an LFSR are determined by its **generator polynomial**. For example, the polynomial specifies feedback from the second and third and fourth stages.

$$G(x) = x^4 + x^3 + x^2 + 1$$

- If the generator polynomial is **primitive**, the LFSR produces a **maximal-length sequence (m-sequence)**.
- For L stage LFSR we have $2^L - 1$ possible states because we exclude all zeros state. The all zeros state is a locked state.
- Such sequences exhibit pseudo-random statistical behavior, making them highly suitable for scrambling.

1.2. Scrambling Techniques

Two principal categories of scramblers are employed in digital communication systems: additive (synchronous) scramblers and multiplicative (self-synchronizing) scramblers.

Author for correspondence:
Cite this article:

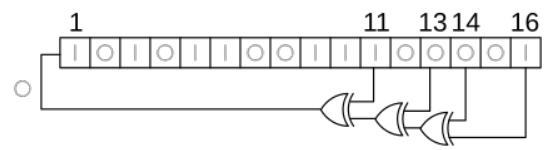

Figure 1. A 16-bit Fibonacci LFSR. The feedback tap numbers shown correspond to a primitive polynomial

1.2.1. Additive (Synchronous) Scrambler

- **Operation:** The scrambler sequence C_n generated by the LFSR is independent of the input data. The scrambled output is obtained as

$$S_n = I_n \oplus C_n,$$

  where I_n is the input data bit and S_n is the scrambled output.
- **Synchronization:** The receiver employs an identical LFSR initialized with the same seed. Correct operation requires strict synchronization between the transmitter and receiver.
- **Error Propagation:** A single bit error in the received scrambled data results in exactly one error after descrambling.

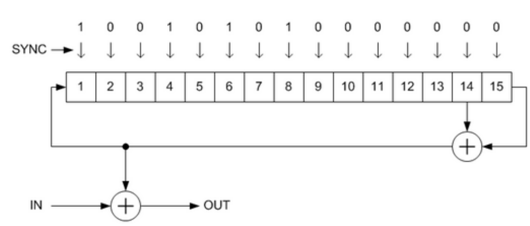

Figure 2. An additive scrambler (descrambler)

1.3. Multiplicative (Self-Synchronizing) Scrambler

- **Operation:** The input data is introduced directly into the feedback path of the LFSR. The scrambled output is formed by combining the input bit with delayed versions of past scrambled bits through modulo-2 addition.

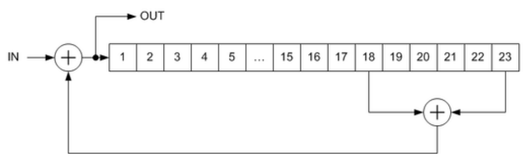

Figure 3. A multiplicative scrambler

$$S_n = I_n \oplus C_n$$
$$D_{in} = I_{in}$$

- **Self-Synchronization:** At the receiver, the same LFSR structure is used. Synchronization is automatically restored within L bits, irrespective of the initial state.
- **Error Propagation:** A single channel error causes a burst of up to L errors during descrambling, as the error propagates through the feedback register.

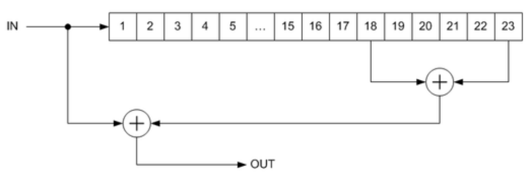

Figure 4. A multiplicative descrambler

1.4. The MATH

1.4.1. System Model and Assumptions

We consider a digital scrambler using a LFSR. The system is defined by:

- **Input process:** $\{I_n\}$ is i.i.d. Bernoulli(p), so $P(I_n = 1) = p$, $P(I_n = 0) = 1 - p$.
- **LFSR process:** $\{C_n\}$ is a m-sequence generated by an LFSR of length L with a primitive polynomial. Its period is $N = 2^L - 1$.
- **Independence:** $\{I_n\}$ and $\{C_n\}$ are statistically independent.
- **Scrambler output:** $S_n = I_n \oplus C_n$.

1.4.2. Mean of the Scrambler Output

The expected value of S_n is:

$$\begin{aligned} E[S_n] &= P(S_n = 1) = P(I_n \oplus C_n = 1) \\ &= P(I_n = 1, C_n = 0) + P(I_n = 0, C_n = 1) \\ &= p \cdot P(C_n = 0) + (1-p) \cdot P(C_n = 1) \end{aligned}$$

For an m-sequence:

$$\begin{aligned} P(C_n = 1) &= \frac{2^{L-1}}{2^L - 1}, \quad P(C_n = 0) = \frac{2^{L-1} - 1}{2^L - 1} \\ E[S_n] &= p \cdot \frac{2^{L-1} - 1}{2^L - 1} + (1-p) \cdot \frac{2^{L-1}}{2^L - 1} \\ &= \frac{p(2^{L-1} - 1) + (1-p)2^{L-1}}{2^L - 1} \\ &= \frac{2^{L-1} - p}{2^L - 1} \end{aligned}$$

For large L, $2^L - 1 \approx 2^L$, thus:

$$E[S_n] \approx \frac{2^{L-1}}{2^L} - \frac{p}{2^L} = \frac{1}{2} - \frac{p}{2^L} \to \frac{1}{2}$$

1.4.3. Autocorrelation Function

The autocorrelation function is defined as:

$$R_S(k) = E[S_n S_{n+k}]$$

Since $S_n \in \{0, 1\}$, $S_n S_{n+k} = 1$ if and only if $S_n = 1$ and $S_{n+k} = 1$. Thus:

$$R_S(k) = P(S_n = 1, S_{n+k} = 1) = P(I_n \oplus C_n = 1, I_{n+k} \oplus C_{n+k} = 1)$$

- **Case 1:** $k = 0$

$$R_S(0) = E[S_n^2] = E[S_n] = \frac{2^{L-1} - p}{2^L - 1}$$

- **Case 2:** $k \neq 0$

We condition on the LFSR sequence:

$$\begin{aligned} R_S(k) &= \sum_{a=0}^{1} \sum_{b=0}^{1} P(C_n = a, C_{n+k} = b) \cdot P(I_n \oplus a = 1, I_{n+k} \oplus b = 1) \\ &= \sum_{a,b} P(C_n = a, C_{n+k} = b) \cdot P(I_n \neq a) \cdot P(I_{n+k} \neq b) \end{aligned}$$

Define $f(x) = P(I_n \neq x) = \begin{cases} p & \text{if } x = 0 \\ 1-p & \text{if } x = 1 \end{cases}$. Then:

$$R_S(k) = \sum_{a,b} P(C_n = a, C_{n+k} = b) \cdot f(a) \cdot f(b)$$

For an m-sequence, the joint distribution for $k \neq 0$ is:

$$P(0,0) = \frac{2^{L-2} - 1}{2^L - 1}$$
$$P(0,1) = P(1,0) = P(1,1) = \frac{2^{L-2}}{2^L - 1}$$

Substituting :

$$\begin{aligned} R_S(k) &= P(0,0)f(0)f(0) + P(0,1)f(0)f(1) \\ &\quad + P(1,0)f(1)f(0) + P(1,1)f(1)f(1) \\ &= \frac{2^{L-2} - 1}{N} p^2 + \frac{2^{L-2}}{N} p(1-p) \\ &\quad + \frac{2^{L-2}}{N}(1-p)p + \frac{2^{L-2}}{N}(1-p)^2 \\ &= \frac{1}{N}\left[(2^{L-2} - 1)p^2 + 2^{L-2} \cdot 2p(1-p) + 2^{L-2}(1-p)^2\right] \\ &= \frac{1}{N}\left[2^{L-2}(p^2 + 2p(1-p) + (1-p)^2) - p^2\right] \end{aligned}$$

Note that $p^2 + 2p(1-p) + (1-p)^2 = (p + (1-p))^2 = 1$.

$$R_S(k) = \frac{1}{N}\left(2^{L-2} - p^2\right) = \frac{2^{L-2} - p^2}{2^L - 1}$$

For large L, $2^L - 1 \approx 2^L$:

$$R_S(0) = \frac{2^{L-1} - p}{2^L - 1} \approx \frac{2^{L-1}}{2^L} = \frac{1}{2}$$
$$R_S(k) = \frac{2^{L-2} - p^2}{2^L - 1} \approx \frac{2^{L-2}}{2^L} = \frac{1}{4} \quad \text{for } k \neq 0$$

Thus:

$$R_S(k) \approx \begin{cases} \frac{1}{2} & k = 0 \\ \frac{1}{4} & k \neq 0 \end{cases}$$

Therefore it boils down to autocorrelation function of an i.i.d. Bernoulli($\frac{1}{2}$) process.

1.4.4. The Takeaway

1. For large LFSR length L, the scrambler output $\{S_n\}$ has:
   - Mean $E[S_n] \to \frac{1}{2}$
   - Autocorrelation $R_S(k) \to \begin{cases} \frac{1}{2} & k = 0 \\ \frac{1}{4} & k \neq 0 \end{cases}$
2. These are the properties of an i.i.d. Bernoulli($\frac{1}{2}$) process, verifying the fact that the scrambler successfully whitens the input spectrum.

2. Why use filter at T_x and R_x and RRC

We will be discussing an M-PSK (M-ary Phase Shift Keying) system. In such a system, symbols are mapped onto a unit circle with M equally spaced phase values. But just mapping and transmitting is not enough, the physical channel and practical constraints force us to use filters both at the transmitter and at the receiver.

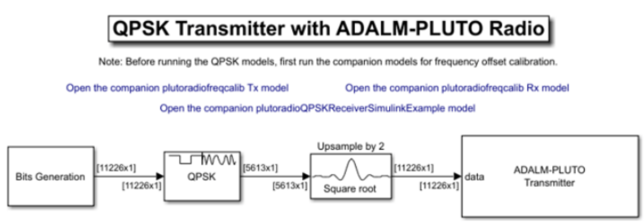

Figure 5. Transmitter Design

2.1. The Symbol Pulse

In linear modulation, the transmitted baseband signal is

$$s(t) = \sum_k a_k \, p(t - kT_s),$$

where a_k are constellation symbols and $p(t)$ is the pulse shape. If no pulse shaping is applied, the natural choice is simply

$$p(t) = \begin{cases} 1, & 0 \leq t < T_s, \\ 0, & \text{otherwise}, \end{cases}$$

i.e., a rectangular pulse. This arises because:

1. Symbols are transmitted over intervals of duration T_s.
2. The transmitter holds each symbol constant during T_s.
3. This is mathematically just a rectangular window.
4. The receiver samples once per symbol, thus it is consistent with this structure.
5. Hence, rectangular pulses emerge naturally before filtering.

2.2. Need for Filtering at the Transmitter

The rectangular pulse $p(t)$ has the spectrum

$$P(f) = T_s \, \mathrm{sinc}(fT_s),$$

which decays slowly and has an infinite bandwidth. This creates two major issues:

1. **Bandwidth inefficiency:** The sinc spectrum spreads energy far outside the intended bandwidth.
2. **ISI potential:** The rectangular pulse has a sinc shaped spectrum. When transmitted over a bandlimited channel, high frequency components of the sinc are attenuated, so the received pulse is no longer perfectly rectangular. Equivalently, in time domain terms, the ideal sinc , which should have zero crossings at integer multiples of T_s to ensure ISI-free detection, is truncated and distorted. As a result, the samples at $t = kT_s$ contain residual contributions from neighboring symbols, i.e.,
   $$r(kT_s) = a_k + \sum_{m \neq k} a_m \, h((k-m)T_s),$$
   where $h(t)$ is the effective channel response. The second term represents intersymbol interference (ISI), which grows more severe as the channel bandwidth becomes narrower.
3. **Non-ideal hardware:** Practical RF amplifiers and mixers cannot handle infinite bandwidth signals, requiring bandlimiting.
4. **Matched detection:** Optimal receiver detection (via matched filtering) requires a well-defined pulse shape, an unbounded sinc makes it difficult to detect.

Therefore, a pulse-shaping filter is applied at the transmitter to limit bandwidth and control ISI. Here , a root-raised cosine (RRC) filter is used so that, combined with an identical RRC at the receiver, the overall response is raised cosine, which satisfies the Nyquist ISI-free criterion.

2.3. Transmitter Design: Upsampling and Root-Raised-Cosine Pulse Shaping

After M-PSK symbol mapping, the symbols $\{a_k\}$ are generated at the symbol rate

$$R_s = \frac{1}{T_s},$$

where T_s is the symbol duration. Direct transmission of these symbols as rectangular pulses leads to sinc shaped spectra, bandwidth inefficiency, and intersymbol interference (ISI), as discussed previously. To address this, two processing steps are introduced at the transmitter: **upsampling** and **pulse shaping** with a root-raised cosine (RRC) filter.

2.3.1. Upsampling

The discrete-time symbols a_k occur once every T_s seconds. In order to implement digital filtering and later DAC, we need to increase the sampling rate to be an integer multiple of the symbol rate.

1. Mathematically, linear interpolation upsampling by a factor L constructs intermediate samples as weighted

averages of neighboring symbols:

$$a_u[n] = (1-\alpha)\,a_k + \alpha a_{k+1}, \quad n = kL + \alpha L,\ \alpha \in [0,1].$$

2. This increases the sampling rate from R_s to LR_s, ensuring L samples per symbol instead of one.
3. The interpolated samples provide a smoother approximation of the continuous-time signal compared to simple zero-insertion.
4. This smoothness improves the accuracy of subsequent pulse shaping.
5. Without linear interpolation, filtering would not have sufficient resolution to enforce the desired continuous-time pulse shape.
6. Also as we are using Gardeners Algorithm at the receiver , we require a number of intermediate points which makes upsampling necessary

Thus, upsampling via linear interpolation prepares the data for pulse shaping by generating intermediate points between symbols.

2.3.2. RRC Pulse Shaping Filter

The Nyquist criterion for zero ISI requires that the overall channel response $h(t)$ satisfy

$$h(kT_s) = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0, \end{cases}$$

so that at sampling instants, each symbol is recovered without interference from others.

2.3.3. Raised Cosine (RC) Response

The raised cosine pulse in frequency domain is defined as

$$H_{\mathrm{RC}}(f) = \begin{cases} T_s, & |f| \le \frac{1-\beta}{2T_s}, \\ \frac{T_s}{2}\left[1+\cos\left(\frac{\pi T_s}{\beta}\left(|f| - \frac{1-\beta}{2T_s}\right)\right)\right], & \frac{1-\beta}{2T_s} < |f| \le \frac{1+\beta}{2T_s}, \\ 0, & |f| > \frac{1+\beta}{2T_s}, \end{cases}$$

where $0 \le \alpha \le 1$ is the roll-off factor.

$$h(t) = \begin{cases} \frac{\pi}{4T}\,\mathrm{sinc}\left(\frac{1}{2\beta}\right), & t = \pm\frac{T}{2\beta} \\ \frac{1}{T}\,\mathrm{sinc}\left(\frac{t}{T}\right)\frac{\cos\left(\frac{\pi\beta t}{T}\right)}{1-\left(\frac{2\beta t}{T}\right)^2}, & \text{otherwise} \end{cases}$$

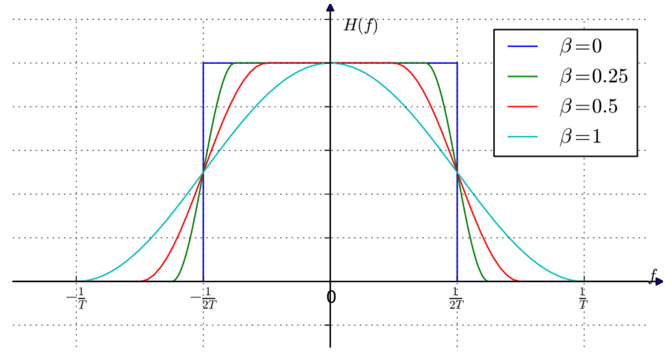

Figure 6. RC filter frequency response

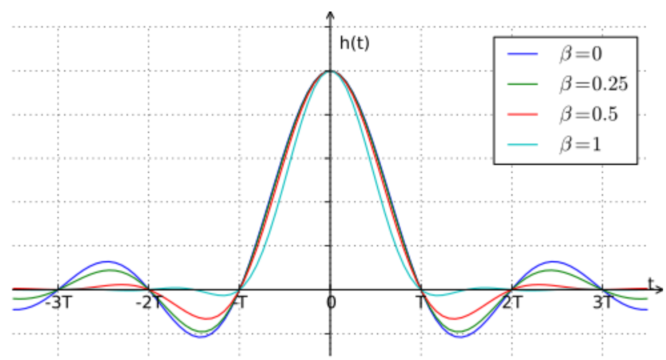

Figure 7. RC filter impulse response

Key properties:

- Bandwidth is limited to $\frac{1+\beta}{2T_s}$.
- Satisfies the Nyquist ISI-free condition.
- Smooth spectral roll-off reduces adjacent-channel interference.

2.3.4. Root Raised Cosine (RRC) Filter

In practice, the RC pulse shaping is split equally between transmitter and receiver, each implementing a root-raised cosine (RRC) filter:

$$H_{\mathrm{RRC}}(f) = \sqrt{H_{\mathrm{RC}}(f)}.$$

Thus, the cascaded response at Tx and Rx is

$$H_{\mathrm{Tx}}(f)\cdot H_{\mathrm{Rx}}(f) = H_{\mathrm{RRC}}(f)\cdot H_{\mathrm{RRC}}(f) = H_{\mathrm{RC}}(f),$$

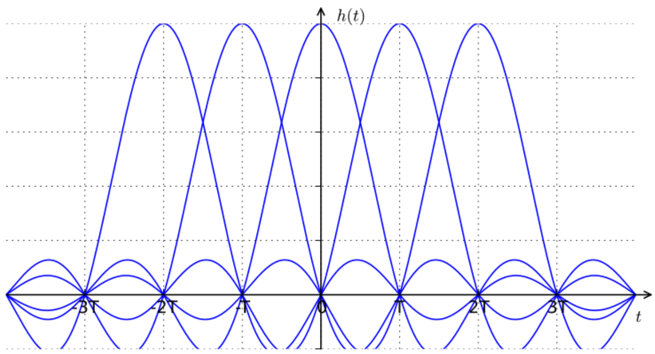

Figure 9. Zero ISI property of RC

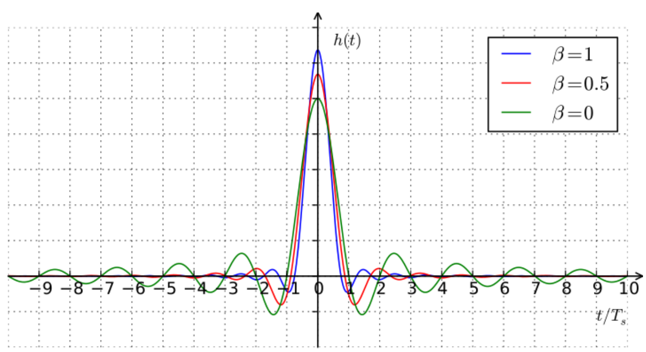

Figure 8. RRC filter

which achieves zero-ISI while also enabling matched filtering at the receiver.

2.4. Matched Filter

[H] In digital communication, the received signal is corrupted by additive white Gaussian noise (AWGN). The matched filter is chosen at the receiver because it:

1. **Maximizes SNR:** For an input signal $s(t)$ observed in presence of AWGN $n(t)$, the matched filter
$$h(t) = p^*(T - t)$$
maximizes the signal-to-noise ratio at the decision instant T.
2. **Eliminates ISI with Nyquist Pulses:** When the transmit filter is Root-Raised-Cosine (RRC), the matched filter is another RRC. Their combined response is a Raised Cosine (RC) filter, which satisfies the Nyquist criterion:
$$p(t)\big|_{t=kT_s} = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}$$
ensuring zero inter-symbol interference (ISI) at the symbol sampling instants.

2.4.1. How Matched Filtering Works with RRC

1. The received signal is noisy and unknown, but the expected transmitted pulse $p(t)$ (RRC) is known.
2. The matched filter is designed as $h(t) = p^*(-T + t)$, the time-reversed and conjugated version of the RRC pulse.
3. Convolving the received signal with this filter maximizes the SNR at the symbol sampling instants. The reason is that the convolution with the conjugated time reversed version is basically the correlation.

2.4.2. Intuition with RRC Pulses

1. The matched filter "looks" for the known RRC shape in the incoming signal, correlating the two.
2. Noise and interference are suppressed because they are uncorrelated with the RRC pulse.
3. ISI is minimized since the matched filter preserves the zero-crossing property of RRC pulses at symbol intervals.

2.5. Trade-offs in RRC Filter Design

1. **Roll-off factor α:** Smaller α gives higher bandwidth efficiency but requires sharper filter transitions (more taps, higher complexity). Larger α eases implementation but uses more bandwidth.
2. **Filter length:** Longer filters approximate the ideal response better, but increase latency and computation.
3. **ISI performance:** Properly designed RRC ensures that ISI is minimized, satisfying the Nyquist criterion after matched filtering.
4. **Spectral containment:** Smooth RRC roll-off avoids spectral leakage, important in practical RF systems with adjacent channels.

3. Automatic Gain Control (AGC)

In this section, we describe the automatic gain control (AGC) loop used in the MATLAB implementation. The algorithm maintains the average output signal power at a desired reference level while adapting to changes in the input signal amplitude.

1. **Signal Model:** The input signal is denoted by $x(n)$, and the controlled output is
$$y(n) = x(n)\, e^{g(n-1)}, \tag{1}$$
where $g(n)$ is the adaptive gain parameter.
2. **Power Estimation:** The instantaneous input power is estimated as
$$P_x(n) = f\left(|x(n)|\right), \tag{2}$$
where $f(\cdot)$ is a smoothing operator, typically implemented as a first-order IIR filter.
3. **Predicted Log-Power:** After gain adjustment, the predicted output power is
$$z(n) = P_x(n)\, e^{2g(n-1)}. \tag{3}$$
Taking the logarithm,
$$\log z(n) = \log P_x(n) + 2g(n-1). \tag{4}$$
4. **Error Signal and Adaptation:** A reference power level `ref` is chosen as the target output power. The error signal is then defined as
$$e(n) = \text{ref} - \log z(n) = \text{ref} - \big(\log P_x(n) + 2g(n-1)\big). \tag{5}$$
The log-domain gain is updated according to
$$g(n) = g(n-1) + K\, e(n), \tag{6}$$

where $K > 0$ is the adaptation step size (loop gain).

5. **Stability Constraint:** To avoid runaway gain growth, the adaptive parameter is constrained by

$$g(n) = \min\left(g(n), g_{\max}\right), \tag{7}$$

where $g_{\max}$ is a design parameter that sets the maximum allowed gain.

4. Frequency Offset Correction

Due to mismatches, the Local Oscillators (LOs) at the transmitter and receiver have slight differences in their generated frequency. This difference, known as the Carrier Frequency Offset (CFO), introduces a time-varying phase rotation on the received baseband signal. If left uncompensated, the constellation points in an M-PSK system will rotate continuously, which is a problem. We tackle this by splitting the offset into coarse and fine components, so we can use simple methods to compensate the bulk of the frequency offset, and then more nuanced techniques to deal with the fine offset.

4.1. Coarse Correction : The M-th Power Method

The M-th Power Method is a frequency estimation technique that exploits the rotational symmetry of M-PSK constellations.

Consider the received baseband signal with CFO, after being downconverted crudely, and (re)sampled:

$$r[n] = \alpha s[n]\, e^{j2\pi\Delta f n T_s} + w[n], \tag{8}$$

where $s[n] \in \{e^{j2\pi k/M},\ k = 0, 1, \ldots, M-1\}$ are the transmitted M-PSK symbols, Δf is the frequency offset, T_s is the sampling period, and $w[n]$ is noise.

Raising $r[n]$ to the M-th power yields:

$$y[n] = \left(r[n]\right)^M = \left(\alpha s[n]\right)^M e^{j2\pi M\Delta f n T_s}. \tag{9}$$

Since $\left(s[n]\right)^M = 1$ for all M-PSK symbols, the resulting signal reduces to:

$$y[n] = \alpha^M e^{j2\pi M\Delta f n T_s}. \tag{10}$$

Thus, the modulation is completely removed, leaving a pure tone whose frequency is M times the original CFO. Estimating the frequency of $y[n]$ (using an FFT, or autocorrelation) gives the following:

$$\hat{\Delta f} = \frac{1}{M}\hat{f}_y, \tag{11}$$

where $\hat{f}_y$ is the estimated frequency of $y[n]$.

Finally, the coarse CFO correction is applied by rotating the received samples:

$$r'[n] = r[n]\, e^{-j2\pi\hat{\Delta f} n T_s}. \tag{12}$$

- Its resolution is limited by the FFT size or averaging window.
- Residual frequency offset must still be tracked with a fine synchronization method, such as a Costas loop.

4.2. Fine Correction: ϕ-Tracking Loops

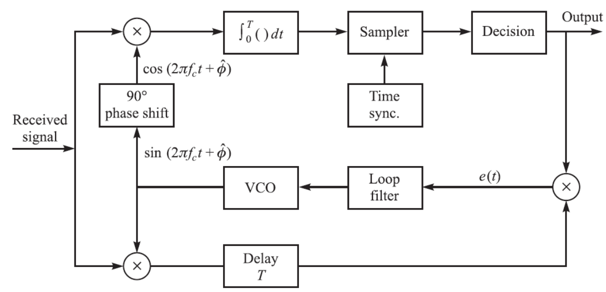

Figure 10. Carrier recovery with a decision-feedback PLL

We assume coarse frequency correction has been applied and only a small residual phase error $\phi[n]$ (rad) remains. Let the complex baseband received sample be

$$r[n] = s[n]e^{j\phi[n]} + w[n], \qquad s[n] \in S_{M\text{-PSK}},\ w[n] \sim \mathcal{N}(0, N_0), \tag{13}$$

and symbol energy $E_s = \mathbb{E}[|s[n]|^2]$. The DD-PLL proceeds by:

1. Making a hard decision on the received symbol:
$$\hat{s}[n] = \mathscr{D}\{r[n]\},$$
where $\mathscr{D}\{\cdot\}$ denotes the decision device mapping to the nearest M-PSK constellation point. Generally, the Decision function is
$$\mathscr{D} = \operatorname{argmin}_{s\in S} |\mathbf{r} - \mathbf{s}|$$
2. Forming the phase detector (PD) output using the received sample and the decision:
$$e[n] = \arg\{r[n]\hat{s}[n]^*\}.$$
3. Filtering and integrating the error signal $e[n]$ through the loop filter and NCO to update the carrier phase estimate:
$$\hat{\phi}[n+1] = \hat{\phi}[n] + K_0\, u[n],$$
where $u[n]$ is the loop filter output.

Phase detector (PD).. A common PD is the arg (angle) of the product $r[n]\hat{s}[n]^*$:

$$z[n] \;=\; r[n]\,\hat{s}[n]^*, \qquad e[n] \;=\; \arg\{z[n]\}. \tag{14}$$

When the decision is correct ($\hat{s}[n] = s[n]$) this becomes

$$z[n] = s[n]s[n]^* e^{j\phi[n]} + w[n]s[n]^* = E_s e^{j\phi[n]} + \tilde{w}[n], \tag{15}$$

where $\tilde{w}[n] = w[n]s[n]^*$ is complex Gaussian with $\mathbb{E}\{|\tilde{w}|^2\} = N_0 E_s$.

Small-angle linearization and detector gain.. For small residual phase $|\phi[n]| \ll 1$ we linearize $e[n]$. Using $\arg(u) \approx \Im\{u\}/\Re\{u\}$ for u near the positive real axis,

$$e[n] \approx \frac{\Im\{E_s e^{j\phi[n]} + \tilde{w}[n]\}}{\Re\{E_s e^{j\phi[n]} + \tilde{w}[n]\}} \approx \frac{E_s\phi[n] + \Im\{\tilde{w}[n]\}}{E_s} = \phi[n] + \underbrace{\frac{\Im\{\tilde{w}[n]\}}{E_s}}_{\text{PD noise}}.$$

Hence the small-signal *detector gain* (slope) is

$$K_d \;=\; \frac{d\,\mathbb{E}\{e[n]\}}{d\phi}\Big|_{\phi=0} \;=\; 1 \quad \text{(radians per rad)}.$$

The PD noise has variance

$$\mathrm{Var}\{e[n]\} \approx \frac{1}{E_s^2}\,\mathrm{Var}\{\Im\{\tilde{w}[n]\}\} = \frac{1}{E_s^2}\cdot\frac{1}{2}\,\mathbb{E}\{|\tilde{w}[n]|^2\} \tag{16}$$

$$= \frac{1}{E_s^2}\cdot\frac{1}{2}\,(N_0 E_s) = \frac{N_0}{2E_s}. \tag{17}$$

(If symbols are normalized $E_s = 1$ this reduces to $N_0/2$.)

Effect of decision errors.. If $\hat{s}[n] \neq s[n]$ (an incorrect hard decision), the PD output is generally biased and can produce a large erroneous $e[n]$. Thus DD-PLL requires sufficiently low symbol-error rate (or an initial blind estimator) to avoid bias/instability.

The loop model is similar to the one used in Symbol timing recovery. The digital update equations follow from the similar setup.

Cycle slips and phase ambiguity.. For pure M-PSK there is an intrinsic phase ambiguity of $2\pi/M$. Differential encoding or pilot symbols resolve this ambiguity. Cycle slips occur when the instantaneous accumulated phase error exceeds the decision sector (roughly π/M) before the loop corrects it. We can ensure $\sigma_\phi \ll \pi/M$ to make slips negligible.

Practically.

1. We ensure coarse freq correction so the residual $|\phi| \ll \pi/M$.
2. Measure or compute detector noise variance $\sigma_e^2 = N_0/(2E_s)$.
3. Choose loop bandwidth B_n to trade off tracking vs noise: typically B_n a few $\times 10^{-3}$–10^{-2} cycles/symbol for stable links.
4. Set damping $\zeta \approx 1$ (or $\zeta \in [0.7, 1]$.
5. Compute ω_n and then K_p, K_i using K_d and K_0 as above.
6. Discretize with bilinear transform to obtain the recursive update and implement the NCO as a phase accumulator.

Remarks..

- Decision errors bias the PD and act as an impulsive disturbance; practical implementations use decision averaging or loop gain scheduling to mitigate this.
- For high-order PSK (large M) the decision regions are small; DD-PLL requires higher E_s/N_0 or lower loop bandwidth to maintain lock without slips.

5. Symbol Synchronization : Timing Recovery

Consider the matched-filter output

$$y(t) = \sum_k a_k\, h(t - kT - \tau) + v(t), \tag{18}$$

where a_k are data symbols, $h(t)$ is the Nyquist pulse, T is the symbol period, and τ is the unknown timing offset.

- Coarse carrier frequency offset correction has been completed in the previous step, so we can assume that the constellation doesn't rotate much over the period of a few symbols.
- Let us assume that a_k are i.i.d, to simplify further calculations

5.1. Gardner Timing-Error Detector (TED)

Let the symbol instants be $t_k = kT + \hat{\tau}$. Define the on-time and early/late samples:

$$y_k = y(t_k), \tag{19}$$

$$y_{k,E} = y\big(t_k - \tfrac{T}{2}\big)\,, \tag{20}$$

$$y_{k,L} = y\big(t_k + \tfrac{T}{2}\big)\,. \tag{21}$$

The Gardner error is defined as

$$e_k = \Re\left\{\big(y_{k,L} - y_{k,E}\big)\, y_k^*\right\}. \tag{22}$$

This is non-data-aided and phase-insensitive.

5.2. S-Curve and Detector Gain

Let $\varepsilon = \hat{\tau} - \tau$ denote the timing error. With $R_h(\tau)$ the autocorrelation of $h(t)$,

$$\mathbb{E}[e_k] = \sigma_a^2\Big(R_h'\big(\tfrac{T}{2} + \varepsilon\big) - R_h'\big(\tfrac{T}{2} - \varepsilon\big)\Big), \tag{23}$$

$$\approx 2\,\sigma_a^2\, R_h''\big(\tfrac{T}{2}\big)\ \varepsilon\ = K_d\,\varepsilon, \tag{24}$$

where K_d is the detector gain. This linear approximation holds for small ε.

5.3. Timing-PLL Model

The loop consists of TED, loop filter $F(s)$, and NCO:

$$\varepsilon(s) \xrightarrow{K_d} e(s) \xrightarrow{F(s)} u(s) \xrightarrow{K_0=1} \dot{\varepsilon}(s). \tag{25}$$

A PI filter

$$F(s) = K_p + \frac{K_i}{s} \tag{26}$$

followed by the NCO (which is basically an integrator) gives the closed-loop characteristic equation

$$s^2 + K_0 K_d K_p\, s + K_0 K_d K_i = 0. \tag{27}$$

Matching to $s^2 + 2\zeta\omega_n s + \omega_n^2$ gives

$$K_p = \frac{2\zeta\omega_n}{K_d}, \qquad K_i = \frac{\omega_n^2}{K_d}, \tag{28}$$

with damping factor ζ and natural frequency ω_n in rad/symbol.

6. Implementation in MATLAB

While the theoretical framework provides an accurate enough description of the processing chain, practical implementation requires specific design choices and parameter settings. MATLAB and Simulink provide specialized blocks that realize the functions of Automatic Gain Control (AGC), synchronization, filtering, and frame alignment. In this section, we document the practical considerations taken in building the transmitter and receiver model.

6.1. Automatic Gain Control (AGC)

Parameters such as the step size, desired output power, averaging length, and maximum allowed gain are available to tune. For an initial demonstration, we chose typical values close to defaults.

- The maxGain is set to $60dB$, so as to mimic realistic gain blocks in RF chains, which have gains from $(40 - 70dB)$.

- Output power just needs to be consistent throughout. (Typical values are 1, 2, 3..)
- Averaging size is taken to be 50, and is a tradeoff between noise tolerance, and speed.

6.2. Coarse Frequency Compensation

Carrier frequency offset due to oscillator mismatch or Doppler is initially corrected using a coarse frequency compensation block. The frequency is estimated using an autocorrelation based technique. For a pure tone :

$$R(k) = \sum_{n=0}^{N-k+1} x[n]x^*[n+k] \tag{29}$$

$$R(k) = \sum_{n=0}^{N-k+1} e^{j2\pi fTn} e^{-j2\pi fT(n+k)} \tag{30}$$

$$\angle R = -2\pi fTk \tag{31}$$

6.3. Symbol Synchronization

The symbol synchronizer employs a timing error detector (e.g., Gardner detector) and a digital timing recovery loop. Using bilinear transform, and simplifying, the update equation is

$$u_k = u_{k-1} + \left(K_p + \tfrac{K_i}{2}\right)e_k + \left(-K_p + \tfrac{K_i}{2}\right)e_{k-1}, \tag{32}$$

and the fractional delay state μ_k (in the NCO) is updated as

$$\mu_{k+1} = \mu_k + u_k \quad \text{(wrapped into } [0,1)\text{)}. \tag{33}$$

An interpolator then generates $y_k, y_{k,E}, y_{k,L}$ at arbitrary μ_k.

6.4. Frame Synchronization and Decoding

Finally, the frame synchronizer locates the start of valid frames using a known preamble sequence. The BER decoding block then compares the decoded bitstream against a reference, providing a direct performance measure of the entire chain.

In summary, the MATLAB/Simulink implementation translates theoretical algorithms into parameterized blocks whose behavior is shaped by engineering trade-offs. The choice of loop gains, filter lengths, and averaging windows determines whether the receiver achieves robust lock and reliable symbol recovery under practical noise and impairment conditions.

7. Running the Code

Clone from github:
https://github.com/vs00007/EE3701-Communication-Systems-Lab