QPSK Modem for Satellite Communication Link

Advised by: Professor David Koilpillai, Professor Lakshmi N. Theagarajan

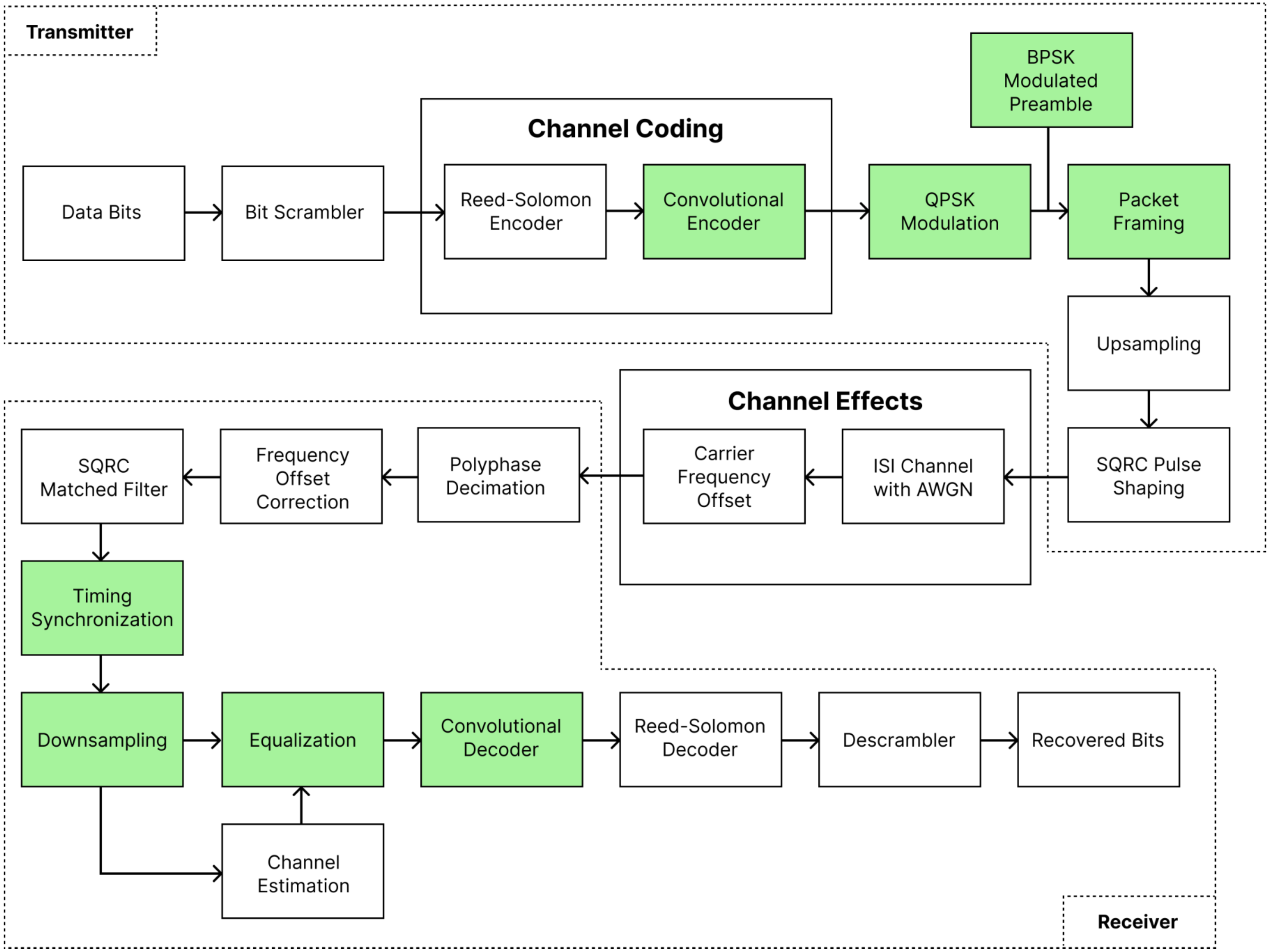
EE4901 UGRC Project

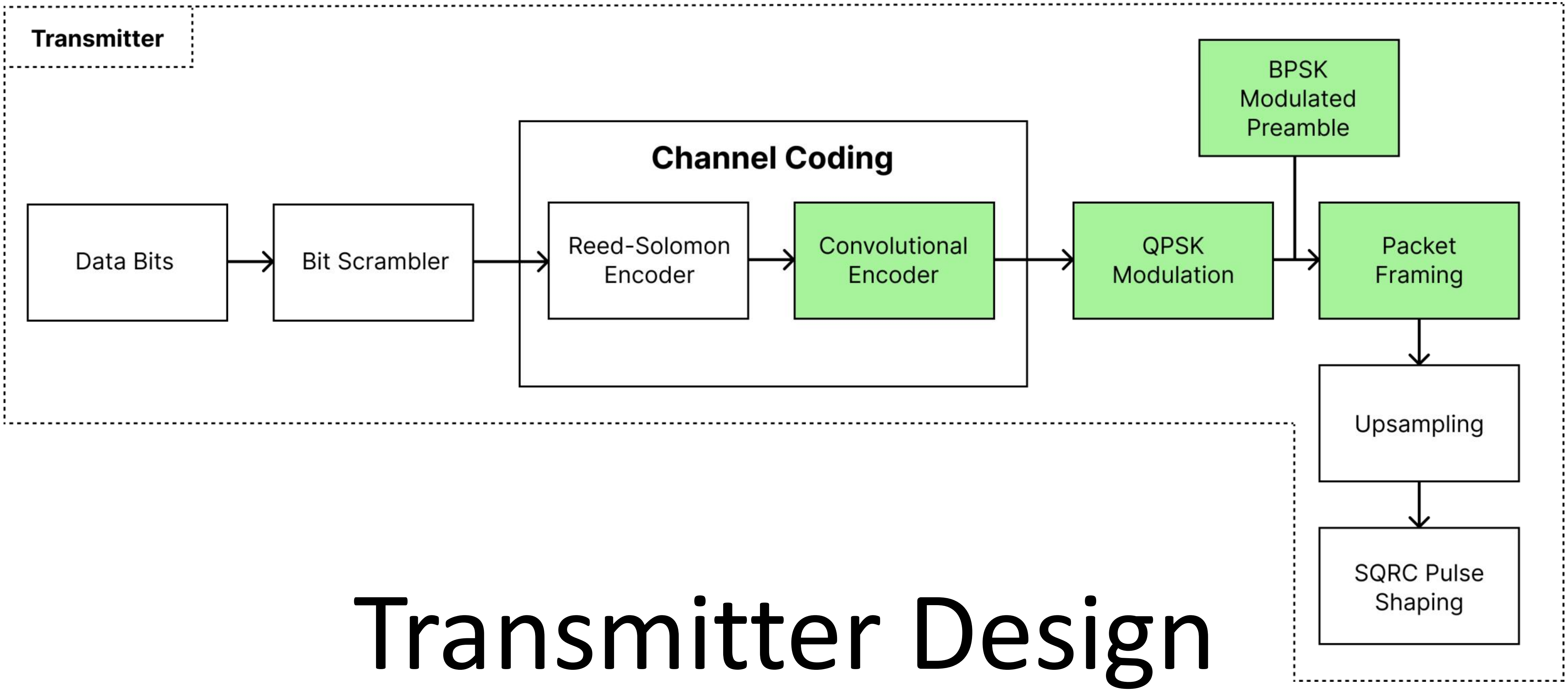
Aditya Mallick

Outline

- Design Choices
  - Transmitter
  - Receiver
- Simulation results
- Independent research – ML for equalization

End-to-end Block Diagram

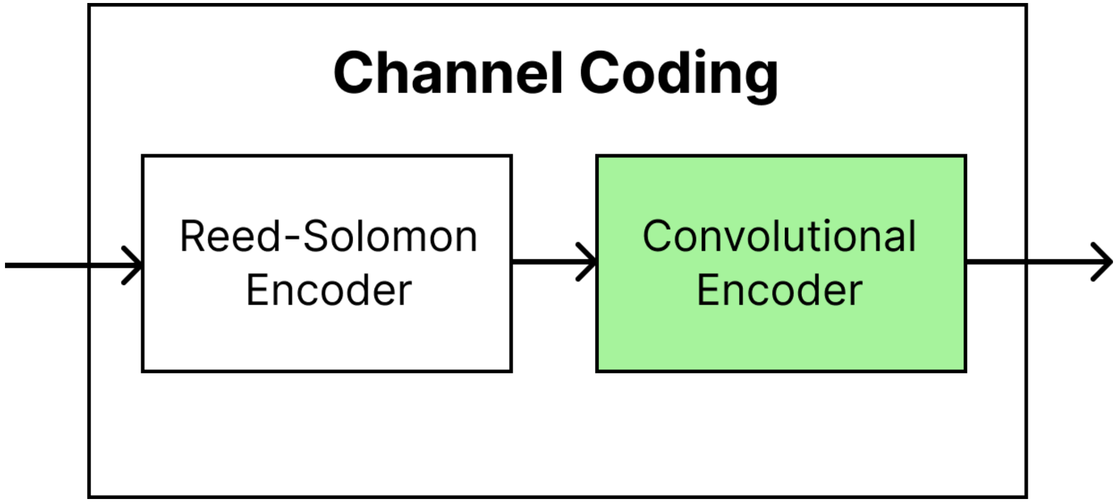




Transmitter Design

Channel Coding

- we have opted to use a concatenated code
- as block length increases, ensures exponentially decreasing error probability
- maintains polynomial time decoding complexity
- RS outer code + inner Viterbi conv code (RSV code)
- first used in Voyager 2 (1977)

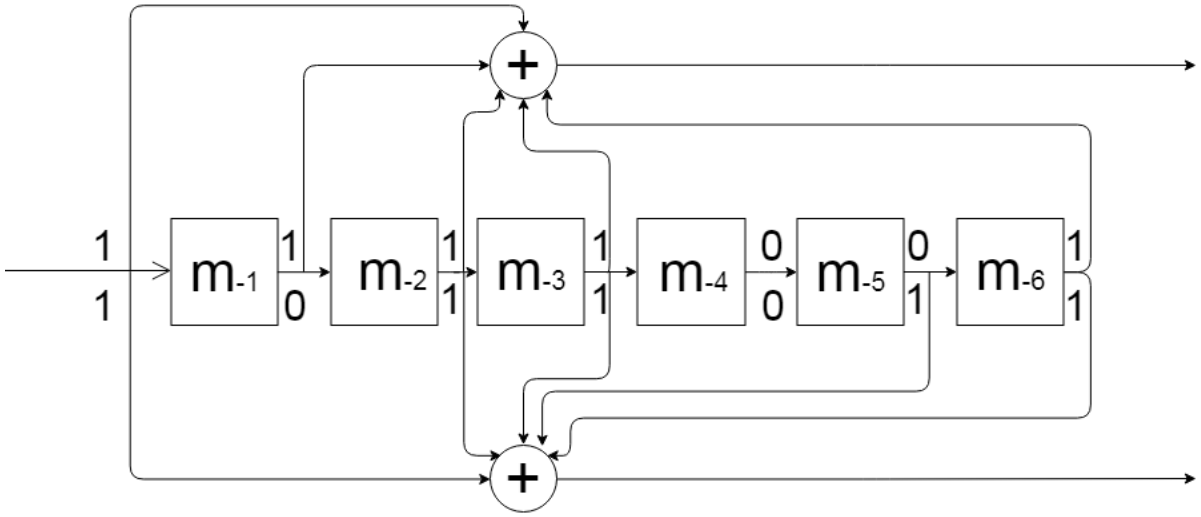


Convolutional Coding

- used in a variety of systems such as 802.11 (Wi-fi) and in satellite communication
- involves transmission of parity bits computed from message bits using generator polynomials
- unlike block codes like RS, only the parity bits are transmitted

Design Choice - Convolutional Encoder

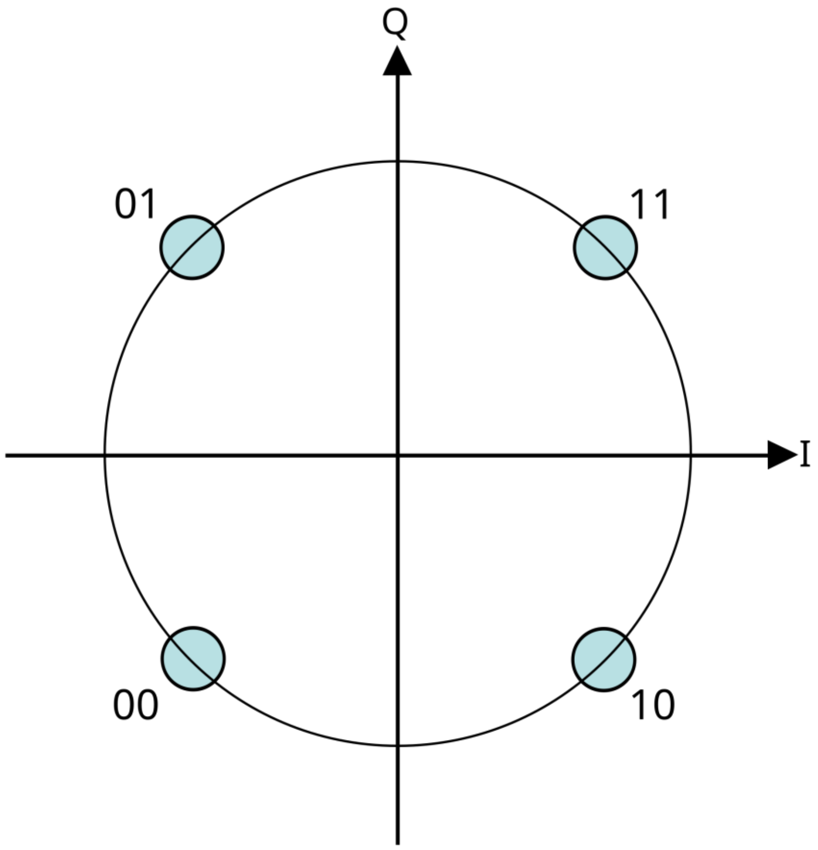
- implemented using a shift register and modulo 2 addition
- size of the SR is known as constraint length (l)
- constraint length decides the complexity of decoding
- for l = 7, and r = 1/2, (g1,g2) = [171_8 , 133_8] have been shown to maximize the free distance of the code



Shift-register for the (7, [171, 133]) convolutional code polynomial

Design Choice - QPSK Modulation

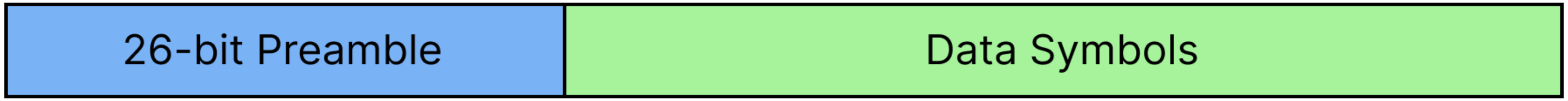
- quadrature phase-shift keying, each symbol is represented by 2 bits
- both bits modulated at same time, choose from 4 possible carrier phase shifts
- QPSK transmits at the same data rate in half the bandwidth compared to BPSK while maintaining same BER



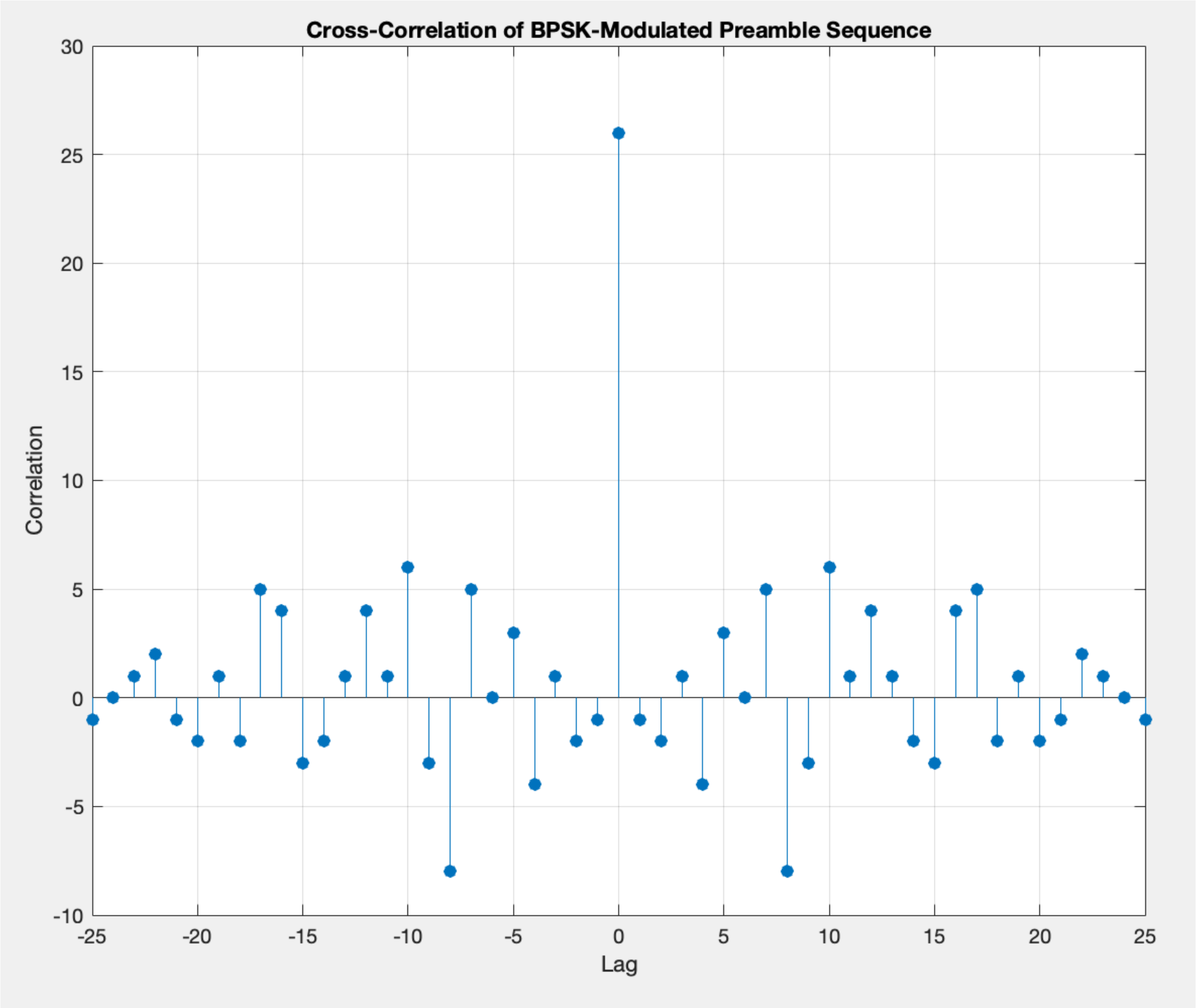
QPSK constellation with gray coding

Design Choice - Packet Framing

- 26 bit BPSK modulated preamble appended to every data frame
- chosen to have good autocorrelation properties and low probability of occurring in data
- used for
  - sample synchronization between transmitter and receiver
  - channel coefficient estimation
  - carrier frequency offset estimation



Packet Structure

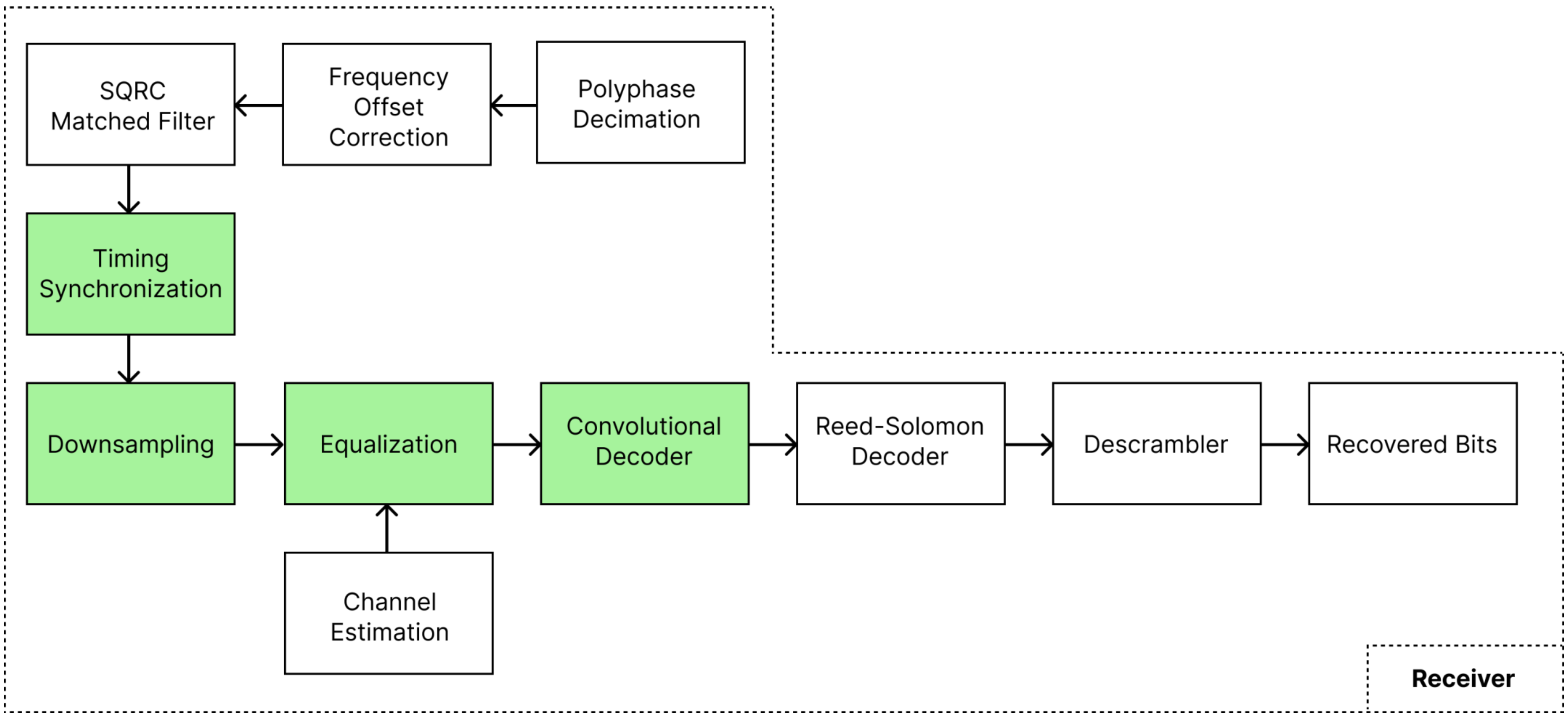


Preamble autocorrelation

Channel Model

- multipath complex fading model with AWGN
- exponentially decaying channel coefficient amplitudes generated using Gaussian distribution
- channel impulse response normalized to unit energy

$$r(k) = \sum_{l=0}^{L-1} f_l I(k-l) + v(k)$$



Receiver Design

Timing Synchronization & Downsampling

- correlate received signal with preamble to identify correct sampling point at the peaks
- ensure each peak crosses certain threshold
- assuming bursty transmission, check that at least a certain number of peaks have been observed consecutively to start downsampling

Algorithm 1 Correlation Peak-Based Timing Synchronization

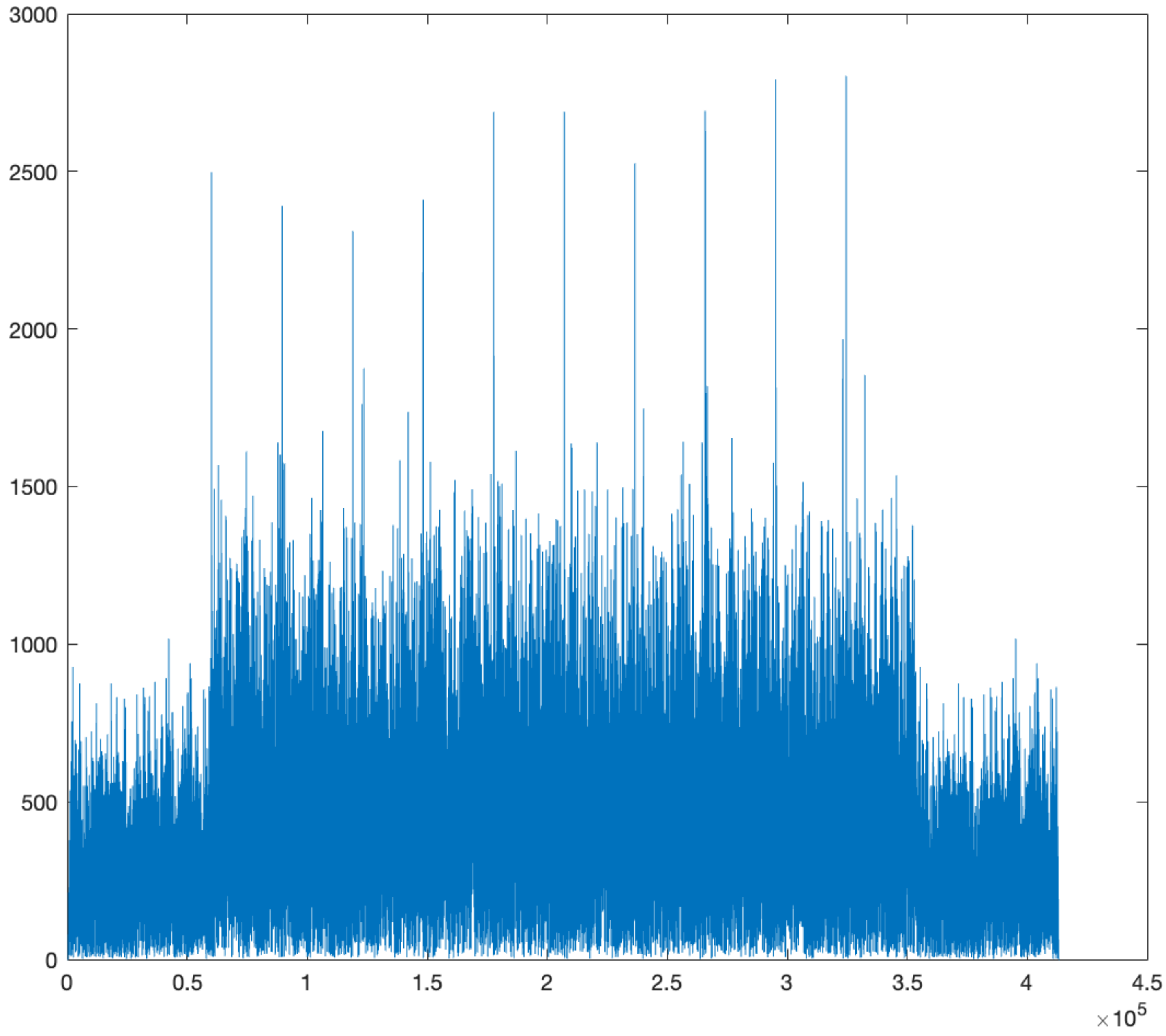
Input: Received signal, preamble sequence, packet size, synchronization criteria

Output: Downsampled preamble and data sequences

```
Compute correlation of received signal with preamble.
Slide a window over the correlation in steps of packet size.

for each window do
    Update peaks, indices, and thresholds.
    if synchronization achieved then
        Find sampling point.
        Extract preamble, data.
    end if
    Repeat until all packets are processed.
end for
return  Extracted preambles, data.
```

Correlation of preamble with received data – 10 packet burst



Channel Equalization

- necessary to mitigate the effects of fading, ISI and recover the original signal accurately at the receiver.
- methods include zero forcing (ZF), MMSE equalizers, MLSE equalization and recently, deep learning-based equalizers.
- choice depends on channel characteristics and performance requirements.

Review of Viterbi Algorithm

- one of the most common symbol detection algorithms
- optimal symbol detector for channels obeying a Markovian input-output stochastic relationship, which is encountered in many practical channels.
- requires channel information (CSI)
- works by recursively calculating path metrics and selecting the sequence with the minimum cumulative metric at each state.

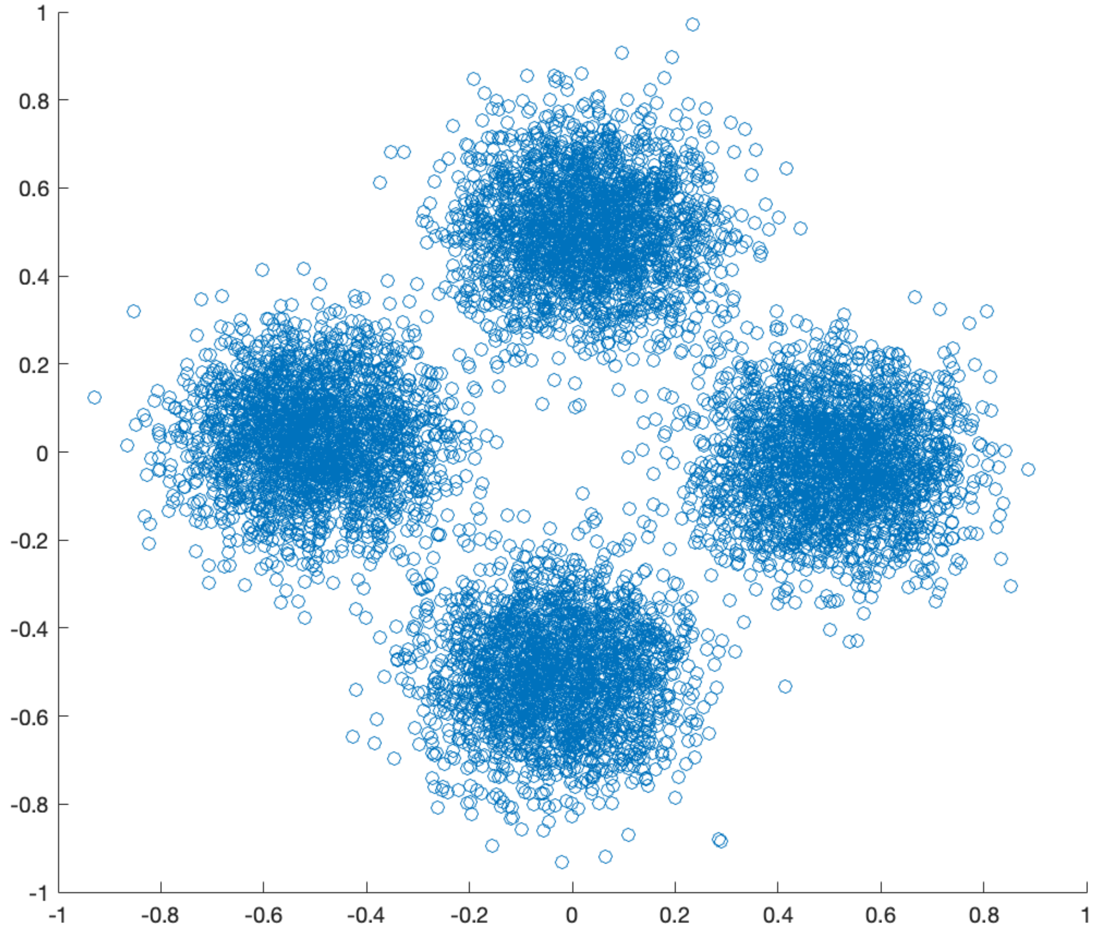
Design Choice – SOVA Equalizer

- we have implemented a modified version of the MLSE equalizer
- Viterbi algorithm provides optimal performance
- branch metric is distance between expected channel output and received symbol
- soft outputs are necessary for input to the convolutional decoder

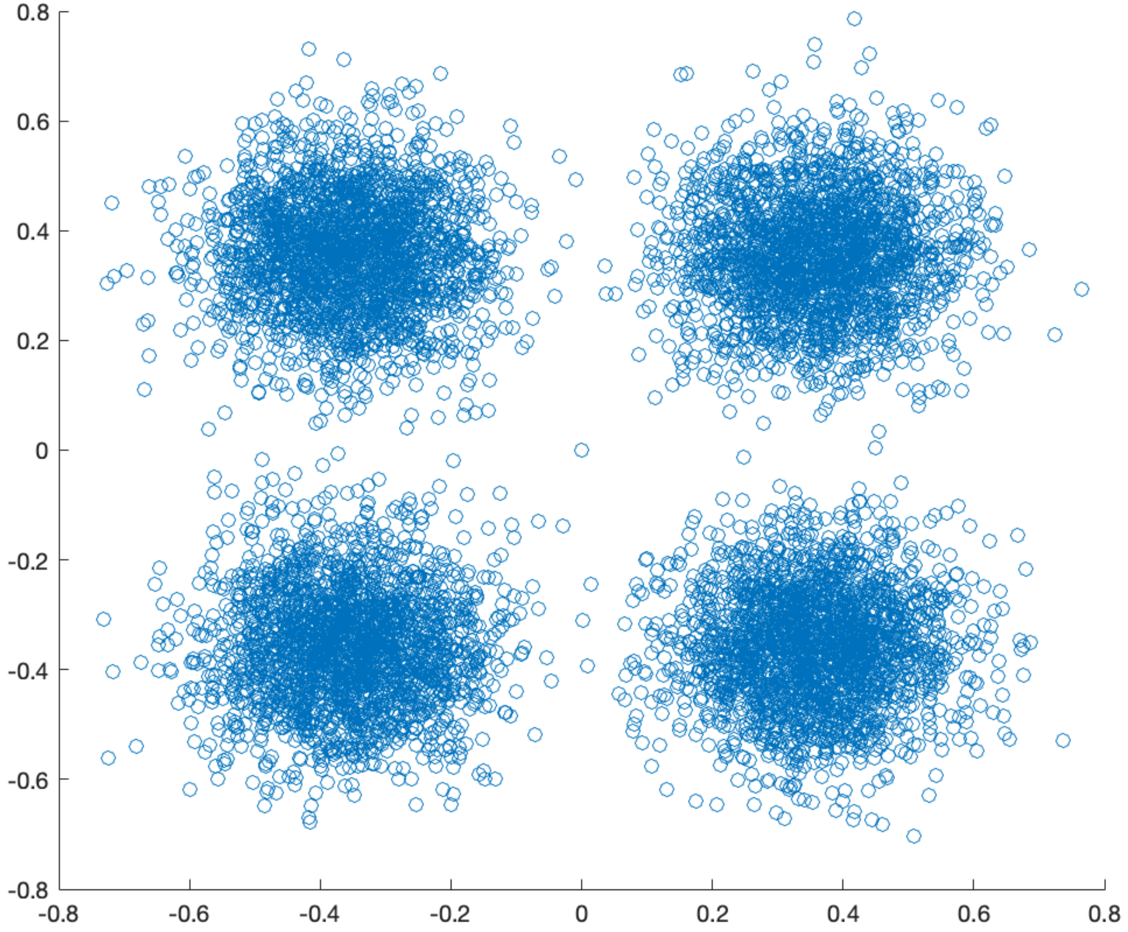
Algorithm 2 Soft-Output Viterbi Equalizer

Input: Received signal, packet size, channel taps, preamble sequence, traceback length.
Output: Equalized soft outputs.

```
Initialization: Trellis, survivors, precompute state transition mappings, set initial state to preamble.
for each received symbol do
    for each prior channel state do
        Compute expected channel output.
        Calculate branch metric as squared error.
        if path metric improved for next state then
            Update path metric for the next state.
            Set survivor of next state as prior state.
        end if
    end for
    if sufficient symbols have been processed then
        Perform traceback using minimum path metric to
        equalize oldest symbol (use soft output formula).
    end if
end for
Perform traceback to recover final symbols.
return  Equalized symbols.
```



Points before equalization



Points after SOVA equalization

Convolutional Decoder

- also uses the Viterbi algorithm
- similar approach to equalization
- output of the conv code can be represented as a QPSK symbol (2 parity bits)
- distance between received symbol and expected output symbol is used as branch metric

Algorithm 3 QPSK Viterbi Decoder

Input: Soft equalized symbols, traceback length, generator polynomials (for conv code).

Output: Decoded bits.

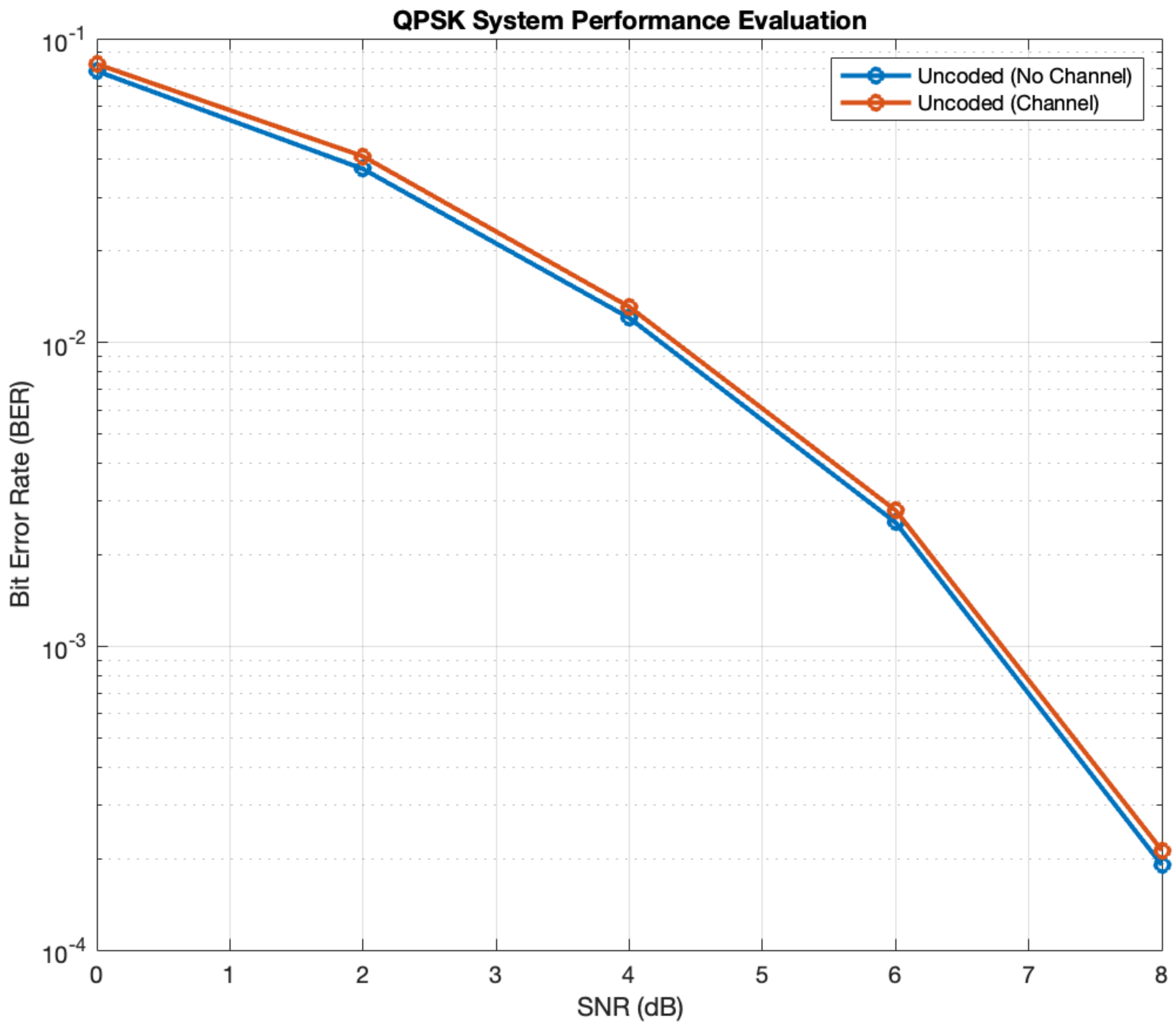
Initialization: Trellis, survivors, precompute state transition mappings.

```
for each received symbol do
    for each future state do
        Compute branch metrics for all possible transitions to future state.
        Add these to path metric of previous state and compare.
        Update future state in trellis with the smallest path metric.
        Store the corresponding survivor state.
    end for
    if sufficient symbols have been processed then
        Perform traceback starting from state with minimum path metric to decode oldest symbol.
        Shift the trellis window forward.
    end if
end for
Perform traceback to recover final symbols.
return Decoded bits.
```

Results

Performance of SOVA equalizer in ISI channel

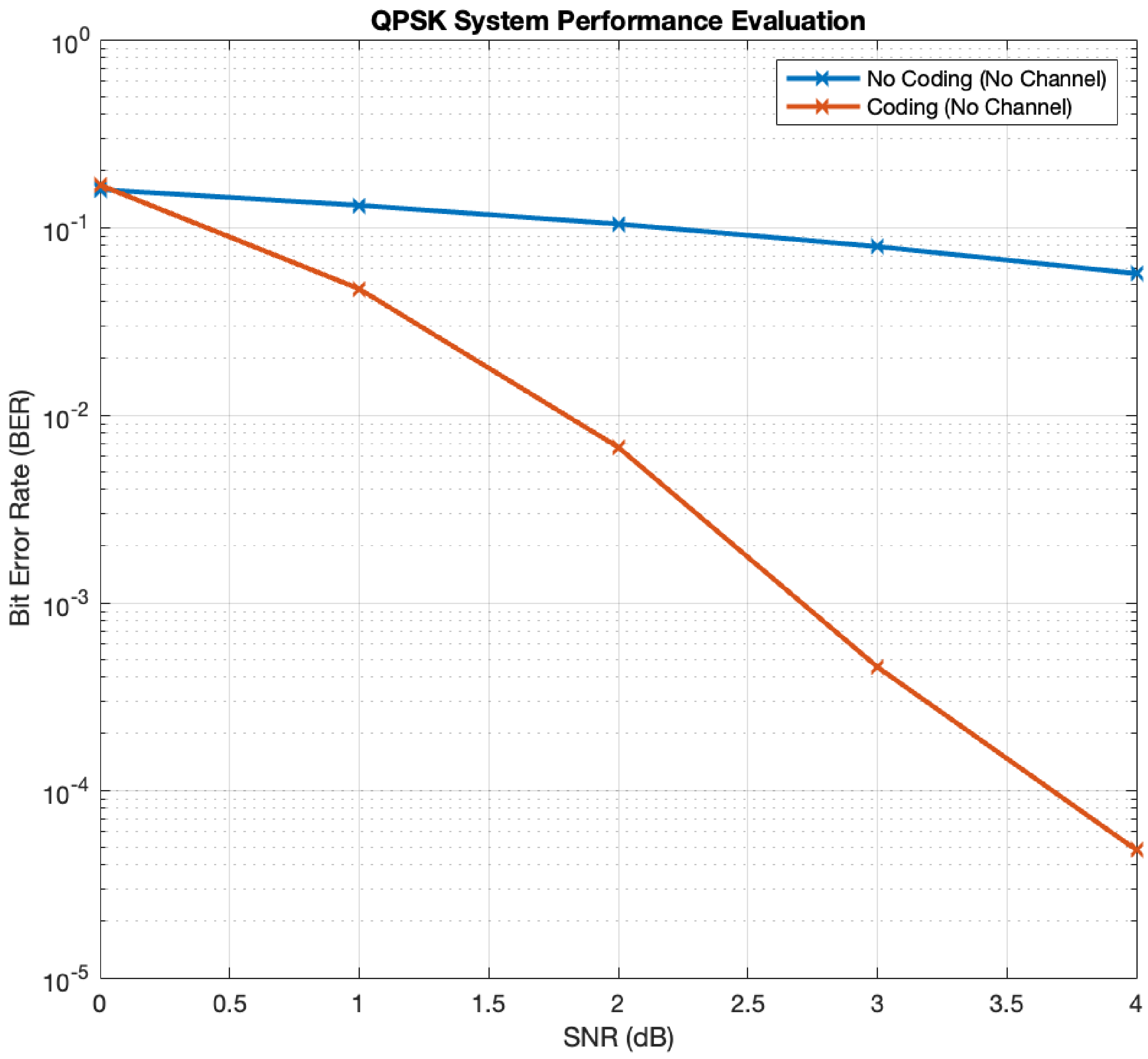
- SOVA can perfectly invert the effect of the channel, close to no ISI case.



Results

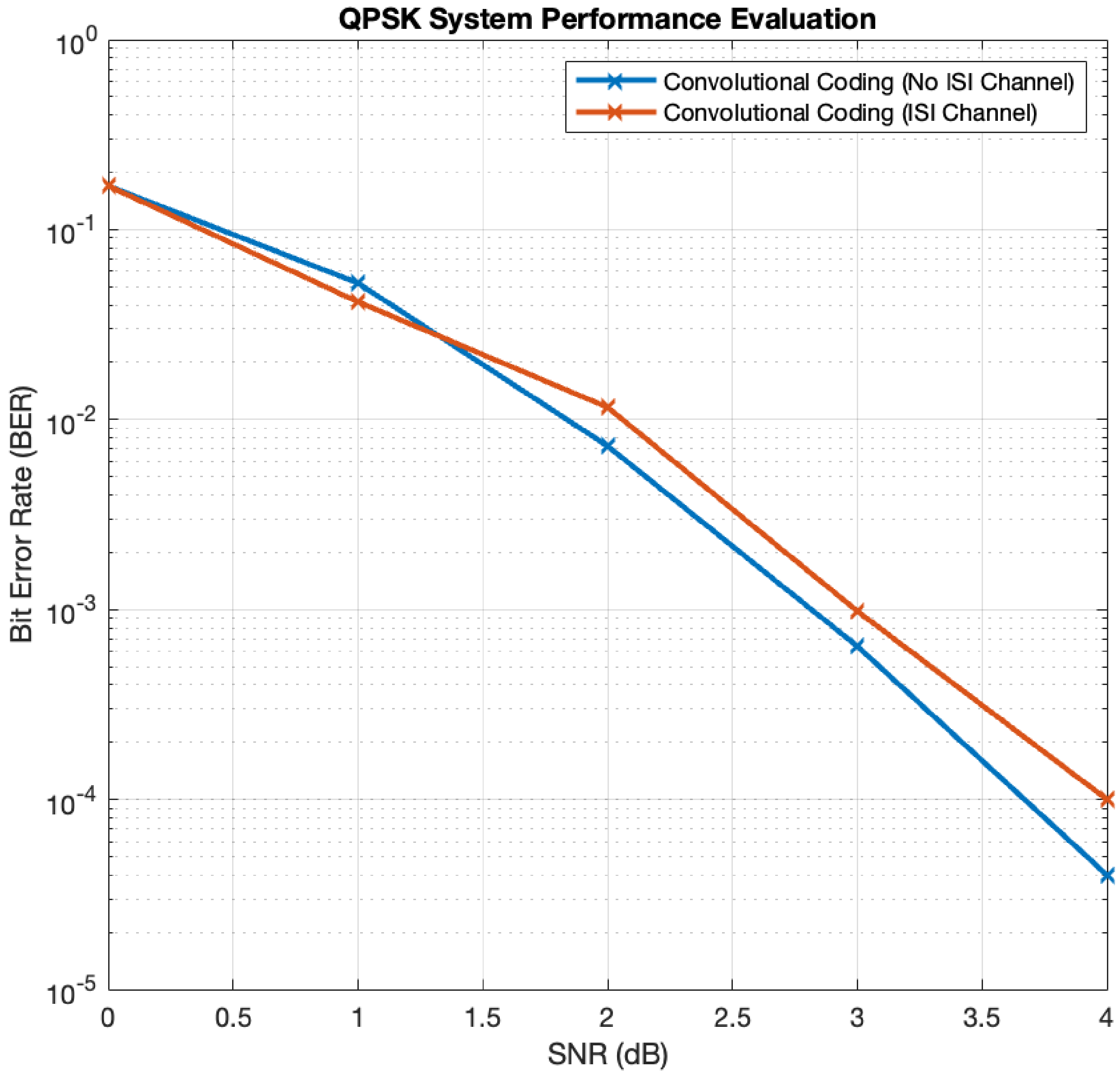
Performance of Viterbi Decoder (no channel)

- Convolutional coding brings down the BER to around 10^-4 at 4 dB.



Results

Combined Performance of SOVA and Viterbi Decoding

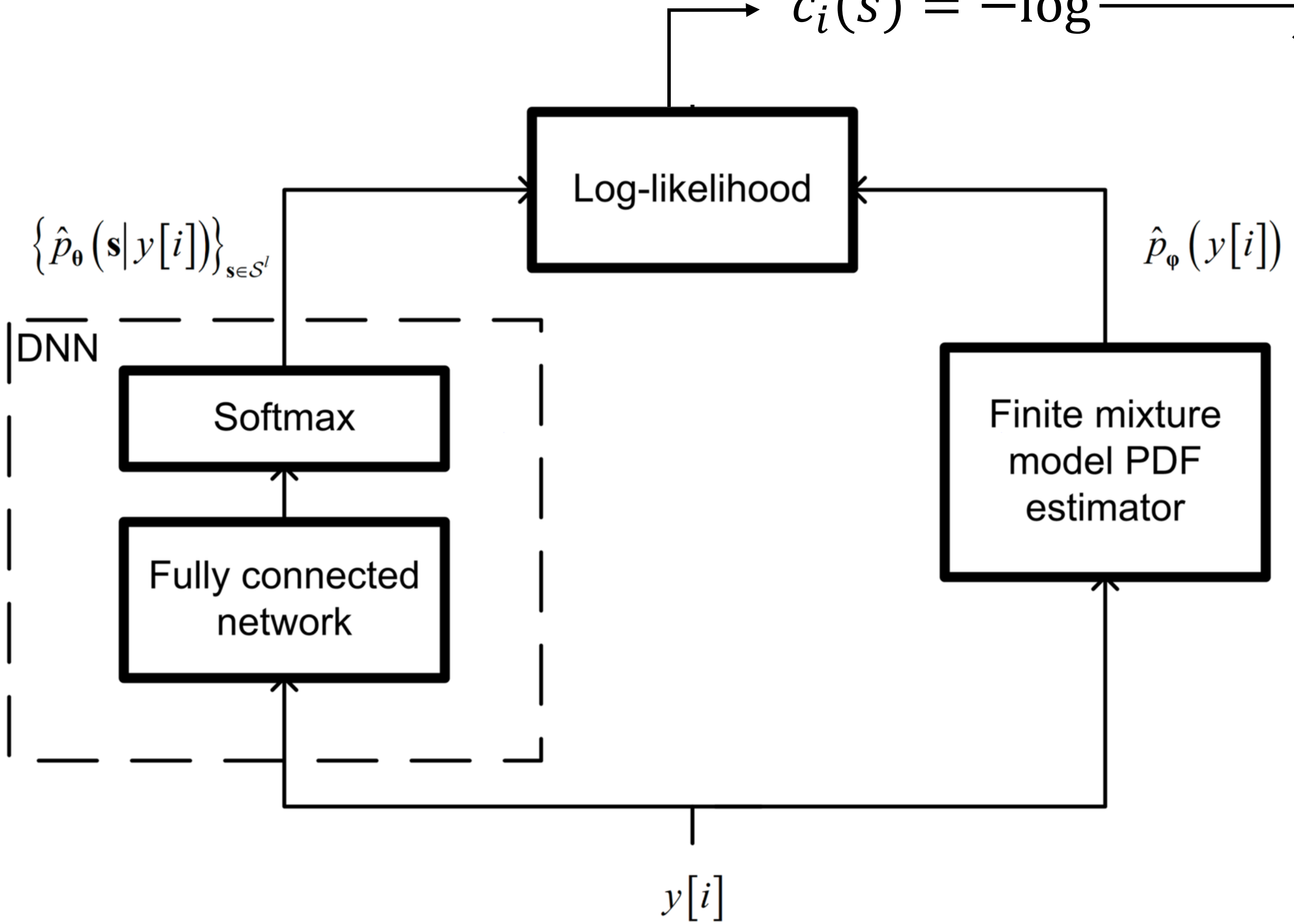


Additional Research – ML for Equalization

- I have studied the feasibility of ML for channel equalization
- experimented with the ViterbiNet algorithm
- integrates ML into the VA by estimating the log-likelihood using deep learning techniques

ViterbiNet Workflow

$$\hat{c}_i(s) = -\log \frac{\hat{P}_\theta(s|y[i]) \cdot \hat{P}_\varphi(y[i])}{m^{-l}}$$

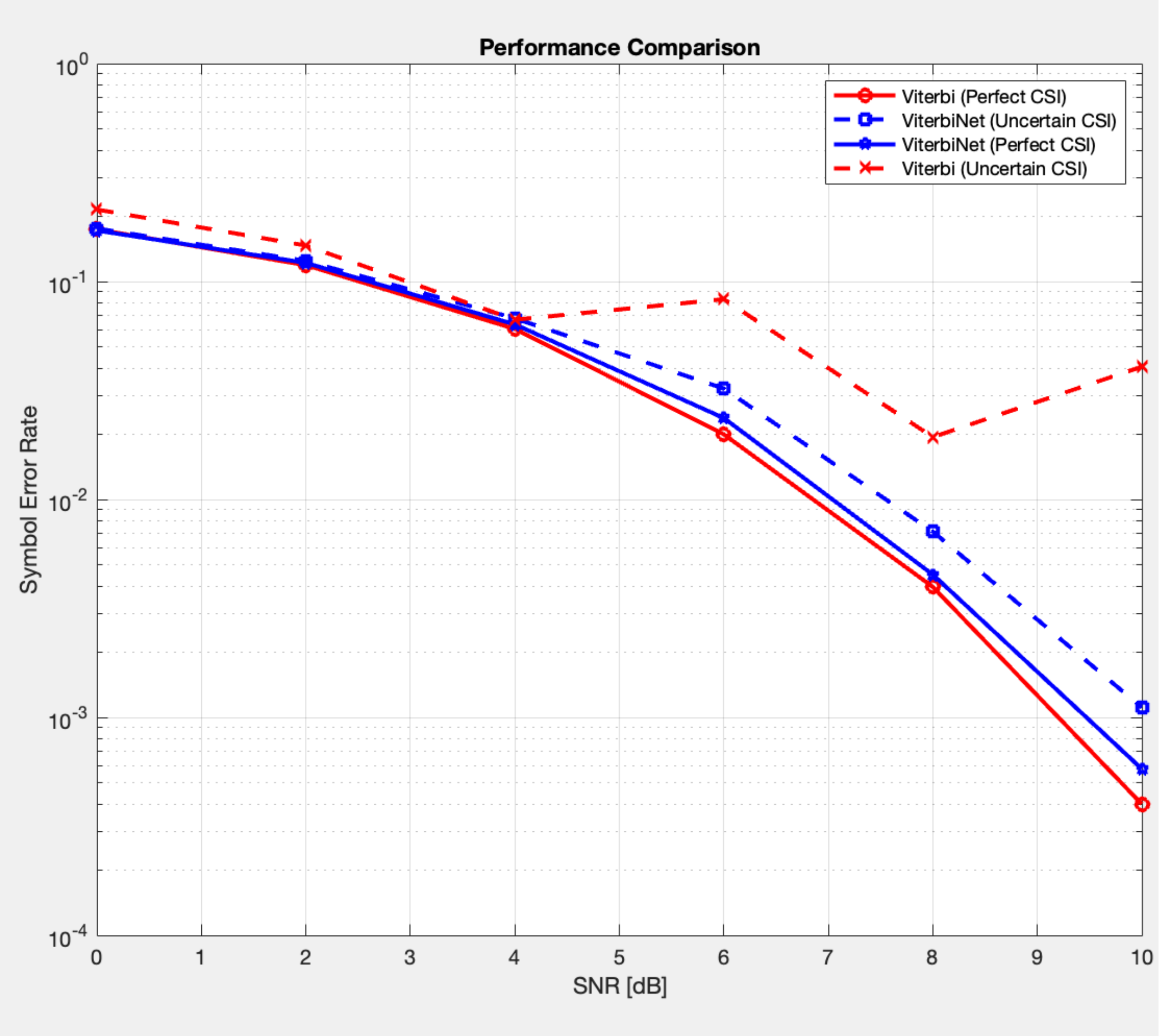


Courtesy of: Shlezinger et Al., “ViterbiNet: A deep learning based Viterbi algorithm for symbol detection,” IEEE Trans. Wireless Commun., vol. 19, no. 5, pp. 3319–3331, 2020.

Experiments

Comparison of ViterbiNet and Viterbi Algorithm in 2 settings:

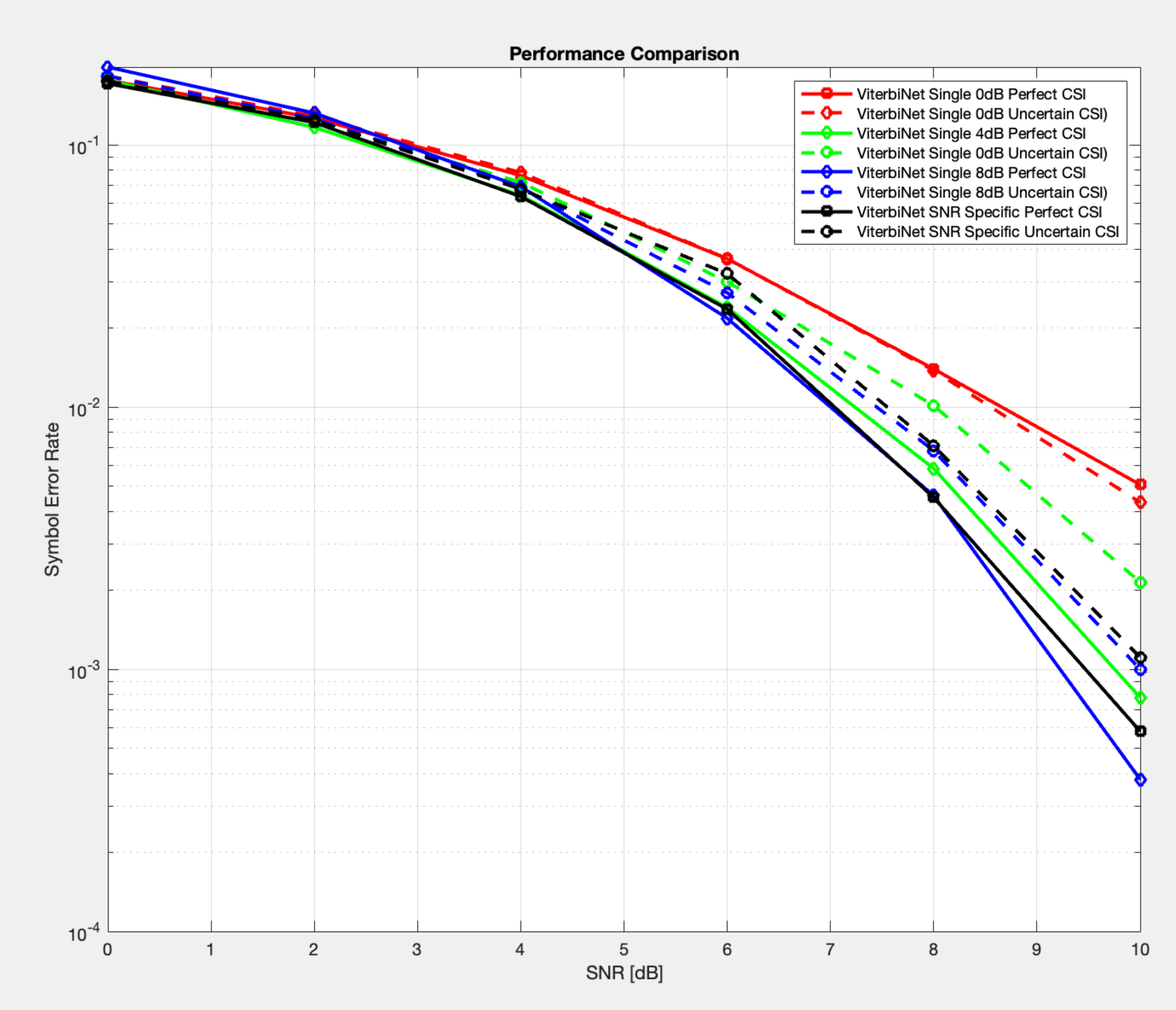
- Perfect CSI
- Uncertain CSI – noisy channel estimate for VA, VN trained using multiple noisy channel estimates.
- VN almost matches VA performance in full CSI setting.
- VN performs much better than VA in uncertain CSI scenario, only slightly worse than Perfect CSI.



Experiments

Comparison of ViterbiNet using general SNR training and specific SNR training:

- ViterbiNet generalizes well to multiple SNR levels even when trained with single SNR data, sometimes even better than SNR specific training!
- Lesser training cost for same performance



Q&A

Code for all experiments available at https://github.com/aditya2331

Thank You!