Unit 4. Coding and Capacity on Fading Channels

ECE-GY 6023. WIRELESS COMMUNICATIONS

PROF. SUNDEEP RANGAN

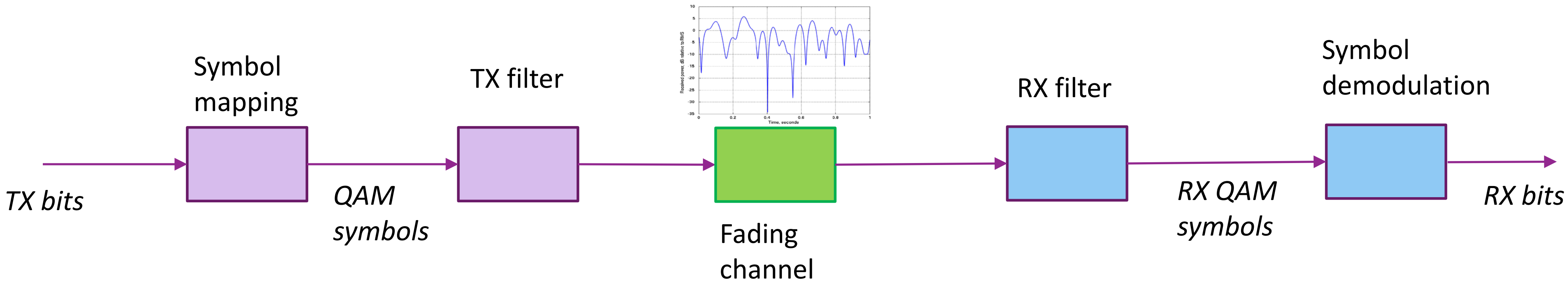
Learning Objectives

- Describe symbol mapping for QAM constellations
- Implement symbol detection for faded symbols
  - Compute average BER and SER on AWGN and flat channels and compare
- Identify if a system can be model as slow or fast fading
- For slow fading, compute outage probability and capacity under a fading model
- For fast fading, compute the ergodic capacity
- Create a TX and RX chain for flat and fading channels with given components
  - Symbol equalization, soft symbol detection, interleaving, channel decoder
- Use MATLAB tools for common channel encoders and decoders
  - Convolutional, turbo codes and LDPC codes

Outline

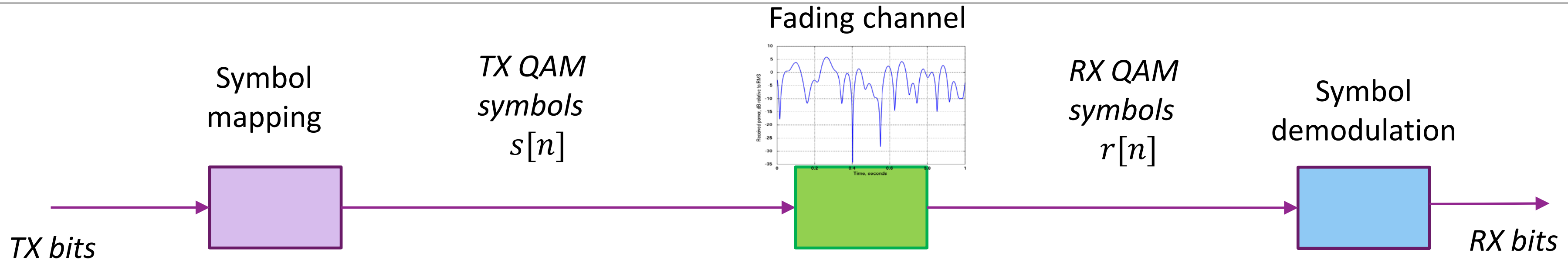
- ❑Uncoded Modulation over Fading Channels
- ❑Outage Probability and Ergodic Capacity
- ❑Review: Coding over an AWGN Channel
- ❑Coding over Fading Channels

Uncoded Modulation



- This section: Uncoded modulation over fading channels
  - That is, communication with no channel encoding and decoding
- We will show uncoded modulation works very poorly
- Virtually all practical wireless systems use coding of some form

Mathematical Model



- **Simple memoryless model**:

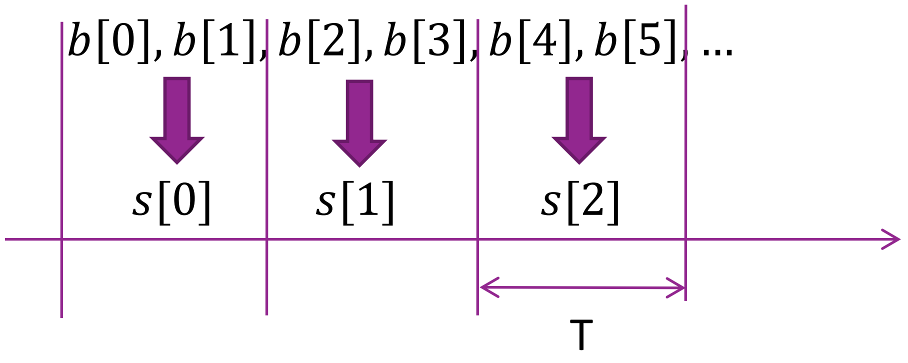
$$\boxed{r[n] = h[n]s[n] + w[n]}$$

  - $s[n]$ and $r[n]$: TX and RX QAM symbols
  - $h[n]$: Fading channel gain, $w[n]$ Noise

- **Assumptions**:
  - Perfect synchronization
  - No ISI in the channel (or the equalizer has removed the effect of the ISI, more on this later)
  - We can look at one symbol at a time

Review: Bit to Symbol Mapping

- $b[k] \in \{0,1\}$ = sequence of bits.
- $s[n] \in \{s_1, \ldots, s_M\}$ = sequence of complex symbols
  - Each symbol has one of M possible values
- Modulation rate: $R_{mod} = \log_2 M$ bits per symbol
  - Each R_{mod} bits gets mapped to one symbol
- Symbol period: One symbol every T seconds.
- Bit rate of $R = R_{mod}/T$ bits per second

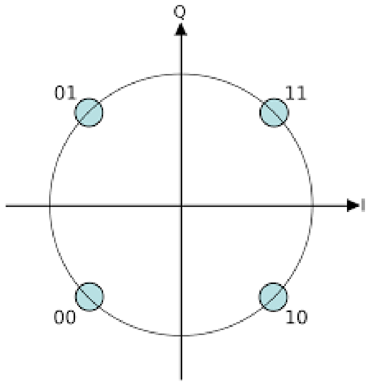


Ex. with M=4 symbols
R_{mod}=2 bits per symbol

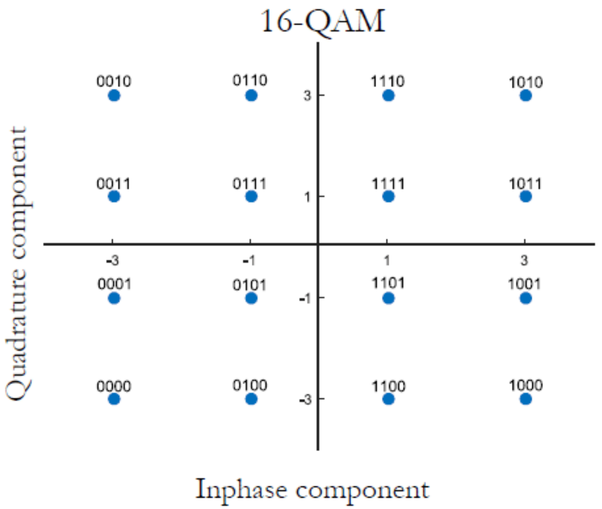
Review: QAM Modulation

- M −QAM: Most common bit to symbol mapping in wireless system
  - $R/2$ bits mapped to I and $R/2$ bits mapped to Q
  - Each dimension is mapped uniformly

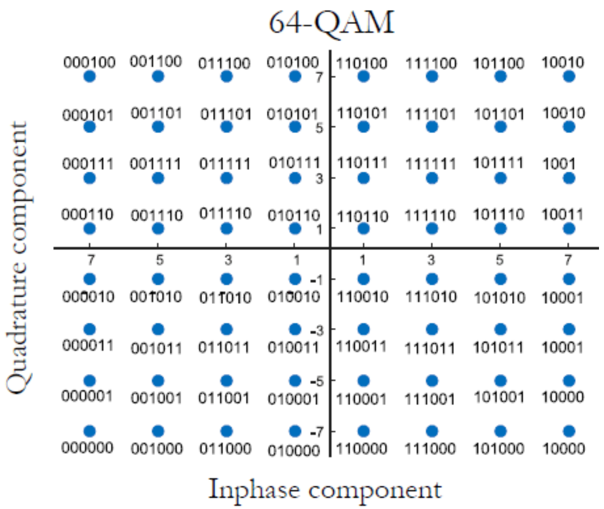
QPSK



$R = 2$ bits/sym



$R = 4$ bits/sym



$R = 6$ bits/sym

ML Estimation for Symbol Demodulation

- Consider single symbol: $r = hs + w,\ \ w \sim CN(0, N_0),\ \ s \in \{s_1, \ldots, s_M\}$
  - Drop the sample index n
  - s is a QAM symbol
- Maximum likelihood estimation:

$$\hat{s} = \arg \max_{s=s_1,\ldots s_M} p(r|s = s_m)$$

- Given s and h: $r \sim CN(hs, N_0)$
- Hence,

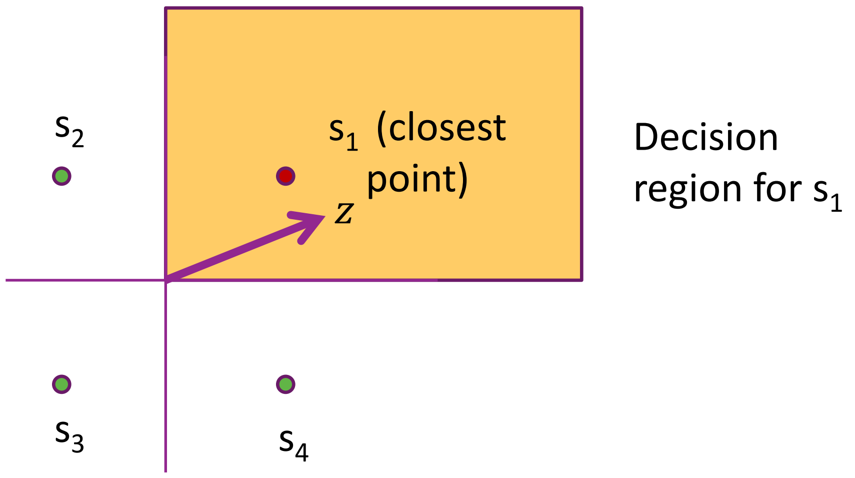
$$p(r|s) = \frac{1}{\pi\, N_0} \exp\left(-\frac{|r - hs|^2}{N_0}\right)$$

Equalization and Nearest Symbol Detection

- Likelihood: $p(r|s) = \frac{1}{\pi N_0} \exp\left(-\frac{|r-hs|^2}{N_0}\right)$
- MLE is: $\hat{s} = \arg\max_s p(r|s) = \arg\min_s |r-hs|^2 = \arg\min_s |z-s|^2$
- Here, $z = \frac{r}{h}$ = equalized symbol.

- Procedure:
  - Step 1: Equalize the symbol: $z = \frac{r}{h}$
  - Step 2: Find $s = s_1, \ldots, s_M$ closest to z in the complex plane

Decision Regions



Example: Decision region in QPSK

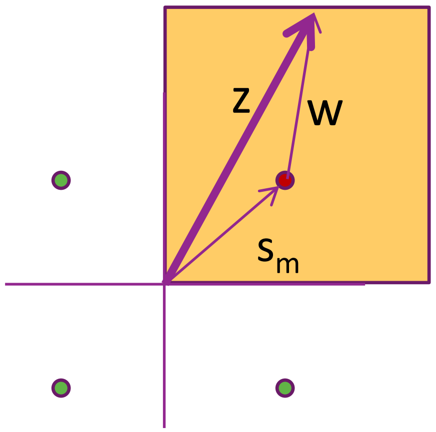
- ML estimate is closest point in constellation to z: $\hat{s} = \arg\min_i \|z - s_i\|$
- Decision region for a point s_m:
  - set of points r where s_m is the closest point: $D_m = \{\boldsymbol{r} | \hat{s} = s_m\}$

Error Probabilities on an AWGN Channel

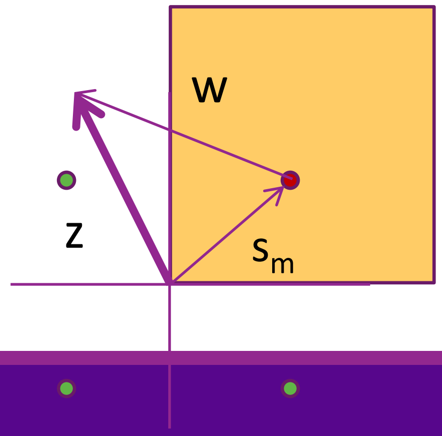
- Error probabilities:
  - Symbol error rate, SER: Prob symbol is misdetected
  - Bit error rate, BER: Probability of a bit is in error
  - Assume TX symbols are uniformly distributed
- First consider AWGN model: $z = s + v$
  - No fading
- SER for QPSK can be shown to be:

$$SER = 1 - \left(1 - Q(\sqrt{\gamma_s})\right)^2 \approx 2Q(\sqrt{\gamma_s})$$

  - SNR = $\gamma_s = \frac{E_s}{N_0} = \frac{E|s|^2}{E|v|^2}$



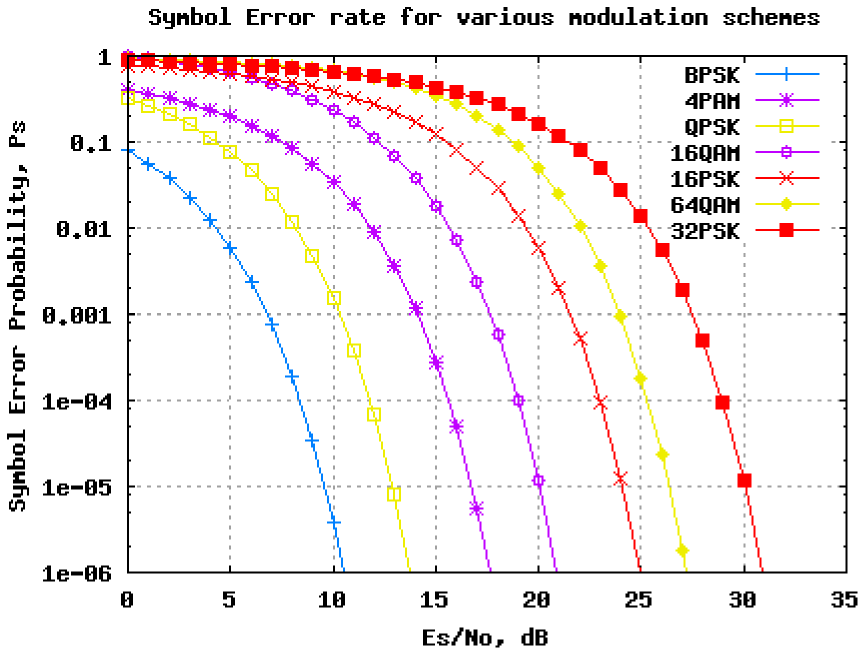
No error
z in correct decision region



Error
z not in
correct decision region

SER for AWGN Modulation

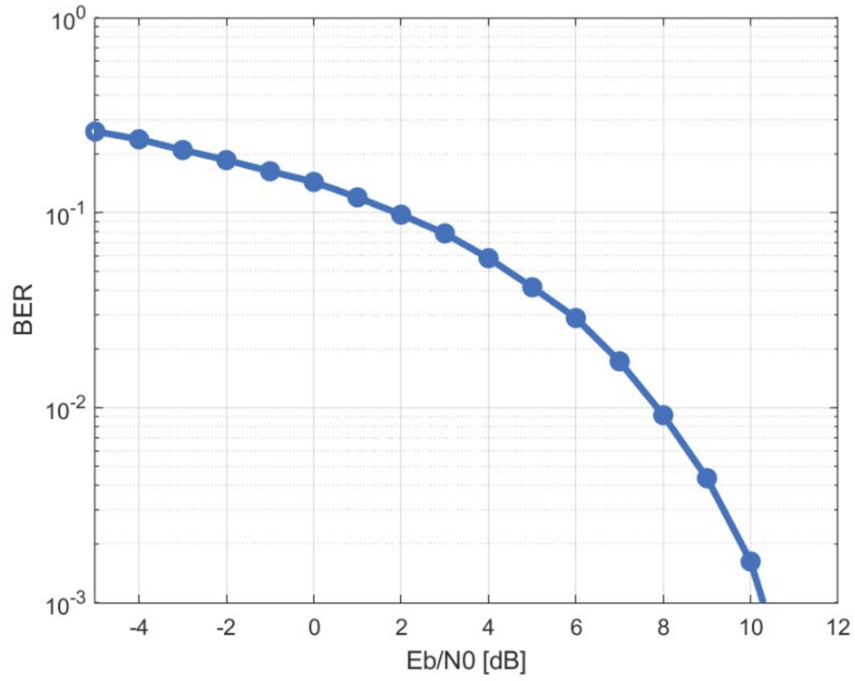
- Error formula can be derived for most QAM mappings
  - See, e.g., Proakis
- For an AWGN channel:
  - SER typically decays exponentially with SNR
  - Ex: for QPSK



Ex: BER Simulation for 16-QAM

```
% SNR levels to test
EbN0Test = (-5:11)';
ntest = length(EbN0Test);

% TX symbol energy
Es = mean(abs(s).^2);
ber = zeros(ntest,1);

for i = 1:ntest

    % Add the noise
    EbN0 = EbN0Test(i);
    chan = comm.AWGNChannel("BitsPerSymbol", bitsPerSym, 'EbNo', EbN0, ...
        'SignalPower', Es);
    r = chan.step(s);

    % Demodulate
    bitsEst = qamdemod(r,M,'UnitAveragePower',true,'Output','bit');

    % Measure the BER
    ber(i) = mean(bitsEst ~= bits);
    fprintf(1, 'EbN0=%7.2f BER=%12.4e\n', EbN0, ber(i));

    % Break if zero since higher SNRs also be zero
    if (ber(i) == 0)
        break
    end
end
```

- See demo
- Easy to do in MATLAB



SNR on a Fading Channel

- Now return to a fading channel:

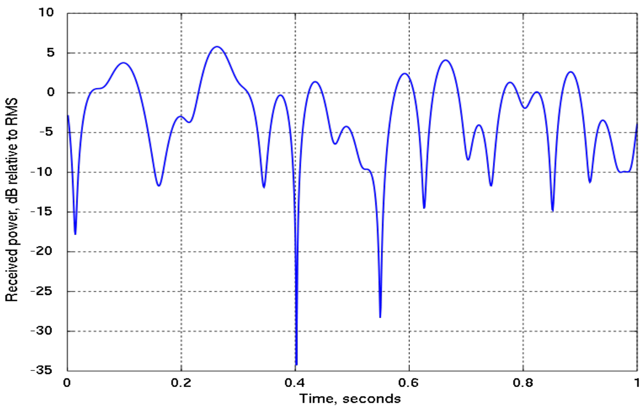
$$r = hs + w, \qquad w \sim CN(0, N_0),$$

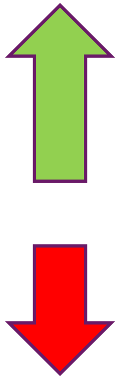
- Equalization: $z = \frac{r}{h} = s + v,$
  - $v = \frac{w}{h}$ Effective noise after equalization
- SNR after equalization:
  - Noise energy after equalization:

$$E|v|^2 = \frac{1}{|h|^2} E|w|^2 = \frac{N_0}{|h|^2}$$

  - SNR is $\gamma_s = \frac{E|s|^2}{E|v|^2} = |h|^2 \frac{E_s}{N_0}$
  - SNR varies with the fading h
- Average SNR is: $\bar{\gamma}_s = E[\gamma_s] = E|h|^2 \frac{E_s}{N_0}$

SNR: $\gamma_s = \frac{|h|^2 E_s}{N_0}$





High SNR

Low SNR

SER on a Fading Channel

- Fading channel: $r = hs + w$
- With fading, SNR is random ,. SNR is $\gamma_s = |h|^2 \frac{E_s}{N_0}$
- Define the average SER:

$$\overline{SER}(\bar{\gamma}_s) = E[SER(\gamma_s)] = \int_0^\infty p(\gamma_s)\, SER(\gamma_s) d\gamma_s$$

  - A function of the average SER
  - Represents the average over independent channel realizations
- If h is Rayleigh distributed, γ_s is exponential with $\bar{\gamma}_s = E[\gamma_s] = E|h|^2 \frac{E_s}{N_0}$

$$\overline{SER}(\bar{\gamma}_s) = \frac{1}{\bar{\gamma}_s} \int_0^\infty e^{-\gamma_s/\bar{\gamma}_s}\, SER(\gamma_s) d\gamma_s$$

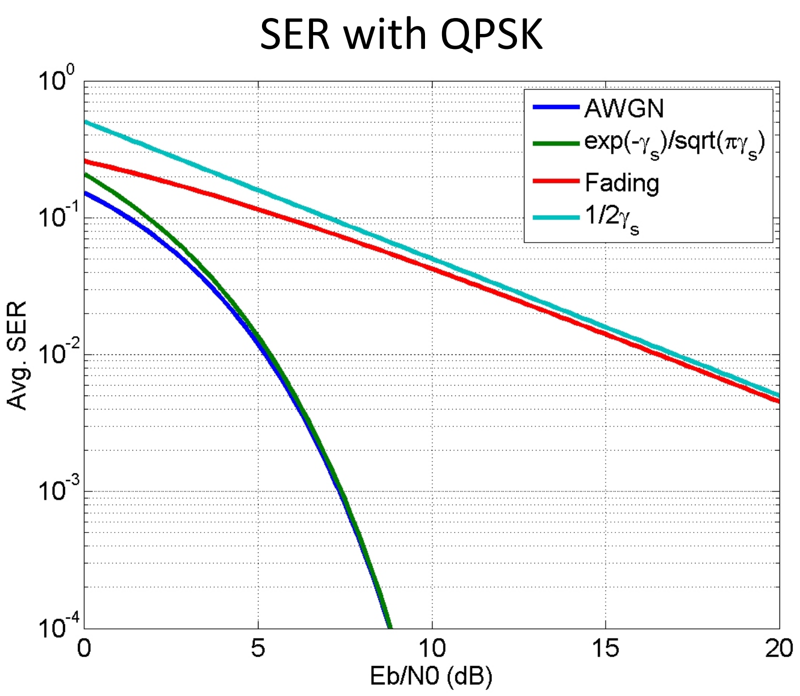
Example: SER on QPSK with Rayleigh Fading

- Rayleigh fading: γ_s is exponential $E(\gamma_s) = \bar{\gamma}_s$
- QPSK: $SER(\gamma_s) \approx 2Q\left(\sqrt{2\gamma_s}\right)$ for large γ_s
- Lemma: Suppose that γ is exponential $E(\gamma) = \bar{\gamma}$, $E\left(Q(\sqrt{\alpha\gamma})\right) = \frac{1}{2}\left[1 - \sqrt{\frac{\alpha\bar{\gamma}}{2+\alpha\bar{\gamma}}}\right] \approx \frac{1}{2\alpha\bar{\gamma}}$
  - Detailed proof below. Write
- Average SER: From Lemma

$$\overline{SER} = E[SER(\gamma_s)] \approx \frac{2}{2(2)\bar{\gamma}} = \frac{1}{2\bar{\gamma}}$$

- Average SER decays as $\propto 1/\bar{\gamma}_s$
- In AWGN channel, SER decays as $Q\left(\sqrt{2\gamma_s}\right) \propto e^{-\gamma_s}$
- Much slower decay

Comparison of Fading vs. AWGN



- Error rate with fading is dramatically higher.
- Ex. for QPSK:
  - No fading, SER decays exponentially
  - With fading, SER decays with inverse SNR
- Similar relations for most other constellations
- Need much higher SNR

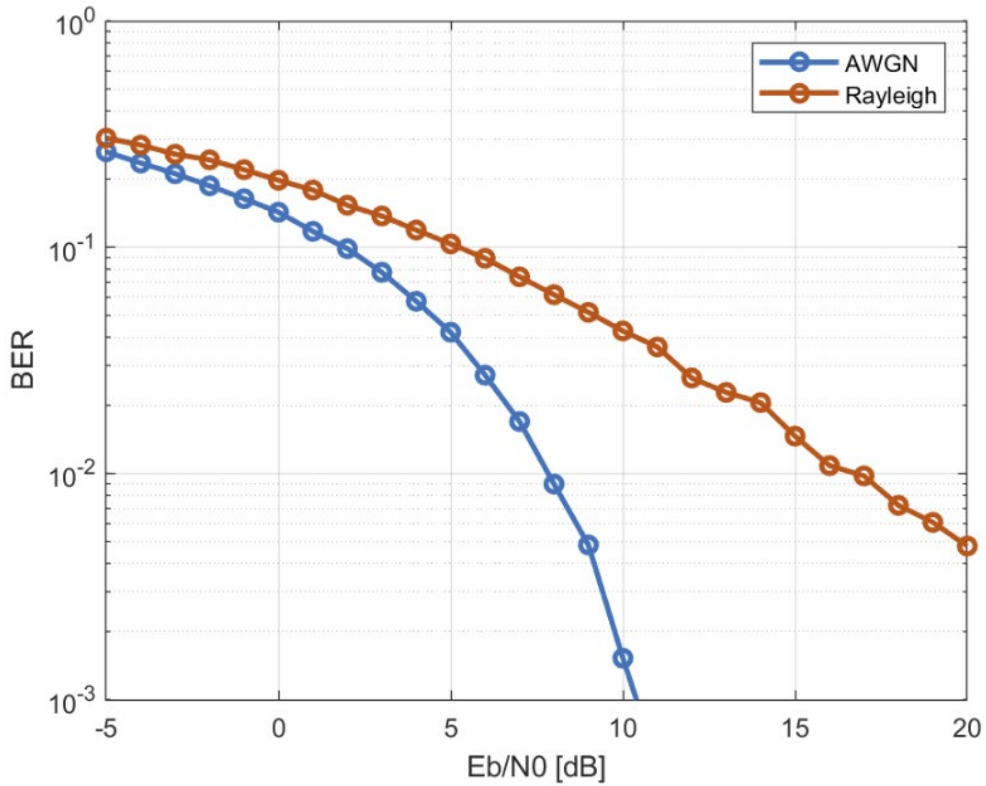
No fading

$$\overline{SER}(\bar{\gamma}_s) \approx \frac{e^{-\bar{\gamma}_s}}{\sqrt{\pi \bar{\gamma}_s}}$$

Rayleigh fading

$$\overline{SER}(\bar{\gamma}_s) \approx \frac{1}{2\bar{\gamma}_s}$$

16-QAM Example



- See demo
- Large gap between AWGN and Rayleigh

Lemma for Average of Q function

- **Lemma:** Suppose that γ is exponential $E(\gamma) = \bar{\gamma}$.

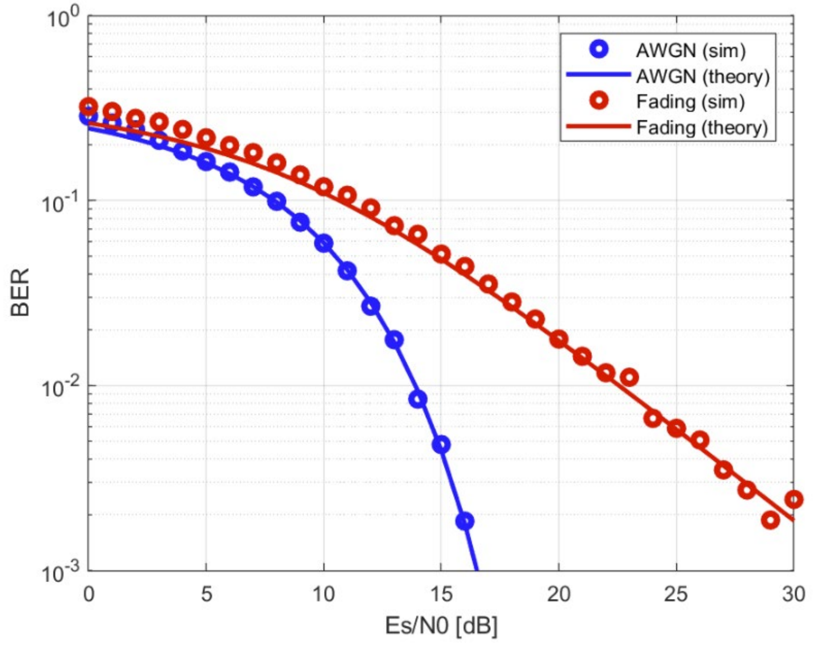
$$E\left(Q(\sqrt{\alpha\gamma})\right) = \frac{1}{2}\left[1 - \sqrt{\frac{\alpha\bar{\gamma}}{2+\alpha\bar{\gamma}}}\right] \approx \frac{1}{2\alpha\bar{\gamma}}$$

- Proof:
  - $E\left(Q(\sqrt{\alpha\gamma})\right) = \frac{1}{\bar{\gamma}}\int_0^\infty Q(\sqrt{\alpha\gamma})e^{-\gamma/\bar{\gamma}}d\gamma$
  - $Q(\sqrt{\alpha\gamma}) = \frac{1}{\sqrt{2\pi}}\int_{\sqrt{\alpha\gamma}}^\infty e^{-u^2/2}du$
  - Change order of integral

In-Class Exercise

Problem 1: Theoretical Error Rate Probability

Modify the code in the demo to compute the `BER` vs. `Es/N0` for 16-QAM for AWGN and fading channel. (Recall the demo measured the `BER` vs. `Eb/N0`).

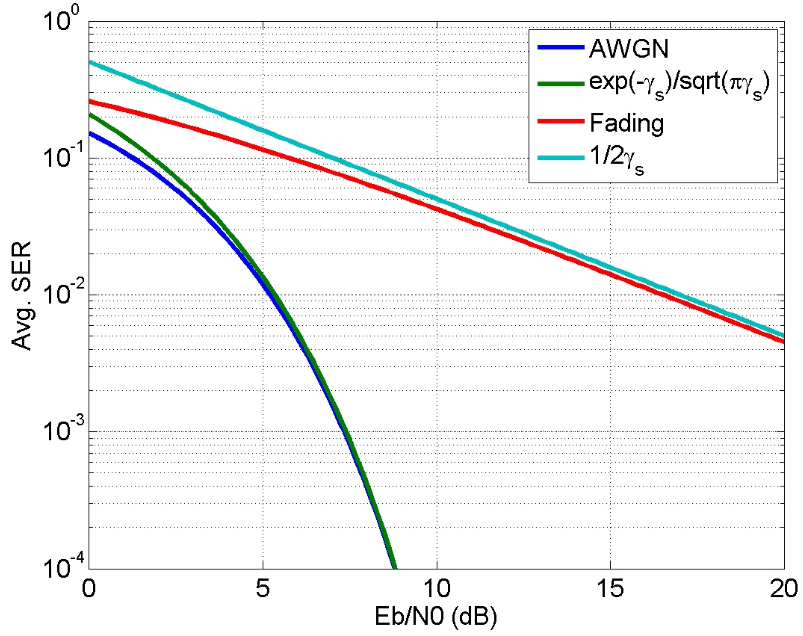


Outline

- Uncoded Modulation over Fading Channels
- Outage Probability and Ergodic Capacity
- Review: Coding over an AWGN Channel
- Coding over Fading Channels

Coding Over Fading Channels

- Lesson from previous section:
  - With fading, uncoded modulation cannot provided sufficient reliability
  - Error rate decays slowly with SNR

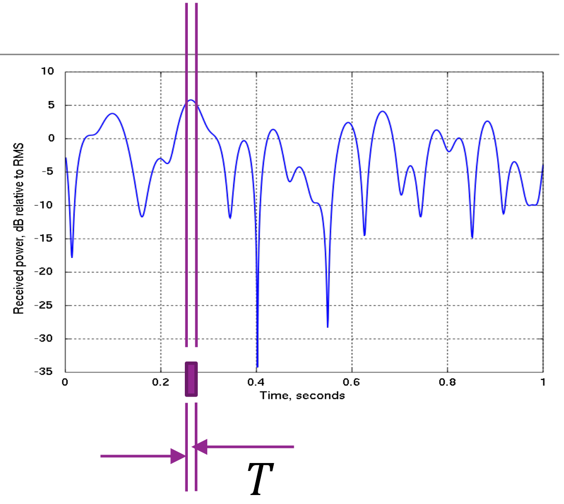


- Channel coding:
  - Send data in blocks
  - Block contains redundancy
  - If some parts fade, can still recover block

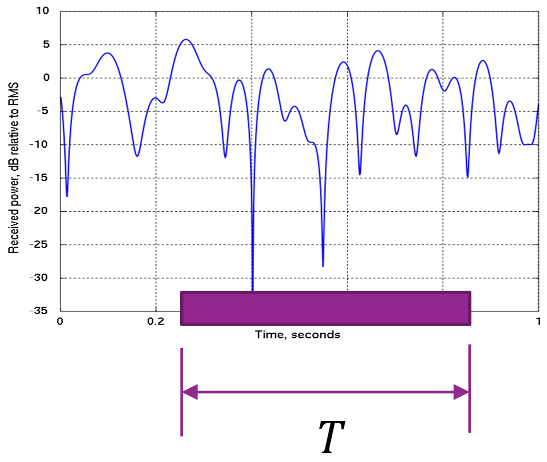
- All commercial wireless systems use coding!

Slow Fading vs. Fast Fading

- To analyze coding over a fading channel, consider two extreme cases
- Slow fading
  - Block length $T \ll T_{coh}$, T_{coh} channel coherence time
  - All symbols in block are faded equally
  - Entire packet sees same SNR
  - Performance measured by outage probability and outage capacity

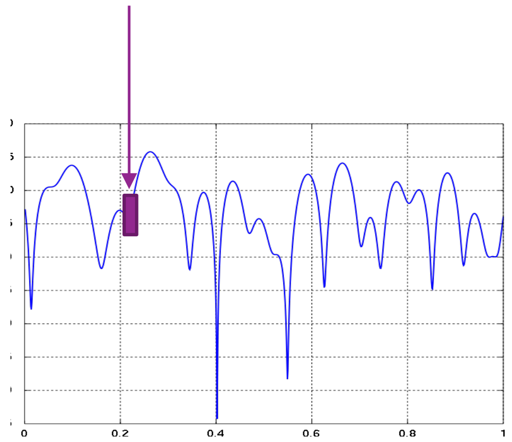


- Fast fading
  - Block length $T \gg T_{coh}$
  - Symbols in packet see many realizations of channel
  - Performance measured by ergodic capacity



Slow Fading and Outage Probability

- Coding block sees an SNR γ
- SNR γ varies but is constant over each block
  - Transmission time $\ll$ Coherence time
- Code has some target SNR γ_{tgt}
  - Target could be based on some block error probability
- Assume γ has some distribution
- Outage probability: $\mathrm{P}_{out} = P(\gamma \leq \gamma_{tgt})$
  - The fraction of time target is not met
- Can be computed from the distribution of γ



Outage Probability for Rayleigh Fading

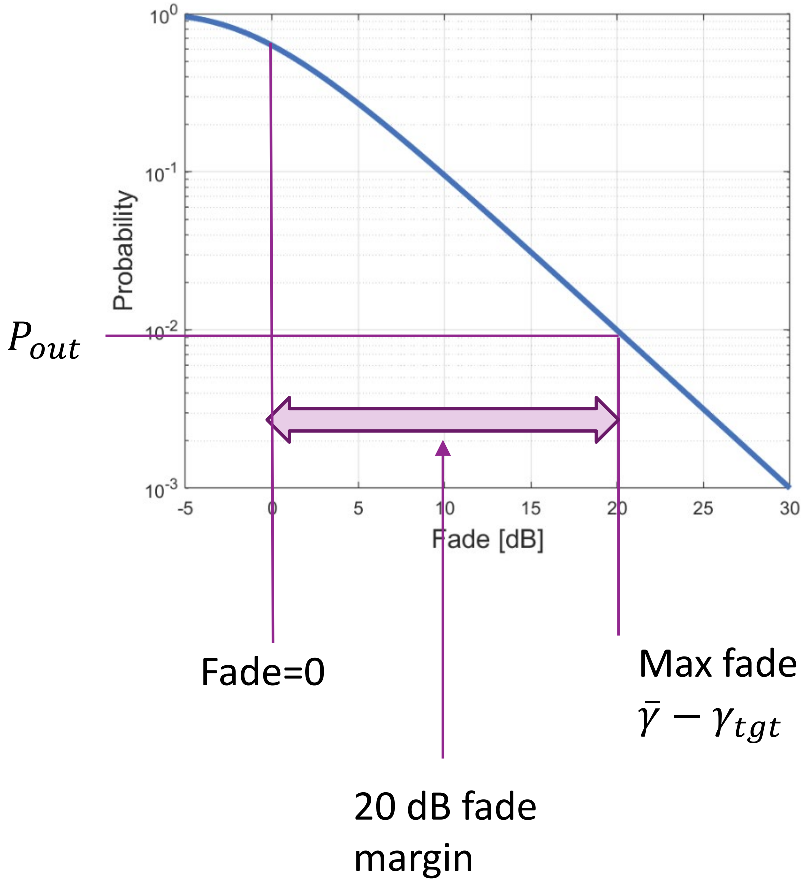
- Suppose a channel is Rayleigh fading
- SNR γ is exponentially distributed with some mean $\bar{\gamma}$
- Outage probability: $P_{out} = P(\gamma < \gamma_{tgt}) = 1 - e^{-\frac{\gamma_{tgt}}{\bar{\gamma}}}$
- Average SNR for a given outage probability: $\bar{\gamma} = -\frac{\gamma_{tgt}}{\ln(1-P_{out})}$
- Fade margin: Additional SNR needed above target for a given outage probability:
  - In linear scale: $\frac{\bar{\gamma}}{\gamma_{tgt}} = -\frac{\gamma_{tgt}}{\ln(1-P_{out})} \approx \frac{1}{P_{out}}$
  - In dB: $\bar{\gamma} \approx \gamma_{tgt} - 10\log_{10}(P_{out})$

Fade Margin Example

- Example:
  - Target SNR is $\gamma_{tgt} = 10$ dB
  - Outage probability: $P_{out} = 0.01$
- From previous slide, necessary average SNR is:

$$\bar{\gamma} \approx \gamma_{tgt} - 10\log_{10}(P_{out})$$
$$= 10 - 10\log_{10}(0.01) = 30 \text{ dB}$$

- The average SNR needs to be 20 dB above target!
- Plot: Fade margin vs. outage
- Fade margins with Rayleigh fading can be enormous!



Outage Capacity

- Suppose we can achieve some rate $R(\gamma)$ as a function of SNR γ
- When SNR γ is random, so is the rate $R(\gamma)$
- Outage capacity: Rate, R_{out}, we can achieve with a probability P_{out}

$$P_{out} = P(R(\gamma) \le R_{out})$$

- Example:
  - Suppose system has 20 MHz bandwidth and the rate is 60% of Shannon capacity
  - The average SNR is 20 dB.
  - What is the outage capacity for 1% outage assuming Rayleigh fading?
- Solution:
  - From earlier, for Rayleigh fading, the SNR achievable at the outage probability is

$$\gamma \approx \bar{\gamma} + 10\log_{10}(P_{out}) = 20 + 10\log_{10}(0.01) = 20 - 20 = 0$$

  - In linear scale, $\gamma = 1$
  - Outage capacity: $R_{out} = 0.6(20)\log_2(1+1) = 12$ Mbps

System Implications for Outage

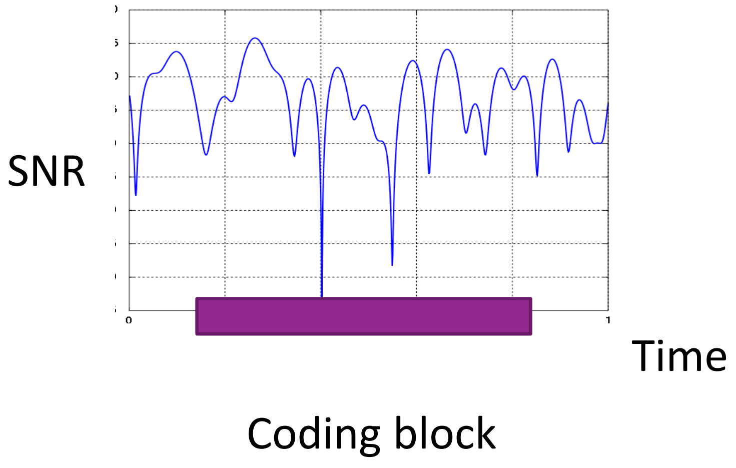
- With slow Rayleigh fading, need to add large fade margin
- Channel coding does not mitigate fading
  - Fading causes all bits to fail
  - Still may be useful to use channel coding (e.g., for noise across the symbols)
- Possible solutions?
  - If there is motion, perhaps we can retransmit later
  - Go to a lower rate (needs less SNR)
  - Just accept that some locations are in outage
- Some of these solutions are discussed in the next unit

Fast Fading Model

- Coding block much longer than coherence time
- Simple information theoretic model

$$r[n] = h[n]s[n] + w[n], \qquad n = 1, \ldots, N$$

  - Channel gains $h[n]$ are i.i.d. with some distribution
  - $w[n] \sim CN(0, N_0)$ and $E|s[n]|^2 = E_s$
  - Each symbol experiences an SNR $\gamma_s[n] = \frac{|h[n]|^2 E_s}{N_0}$
  - Blocklength $N \to \infty$
- Assumption implicitly assumes:
  - We have a very long blocklength N
  - Can experience many independent fades



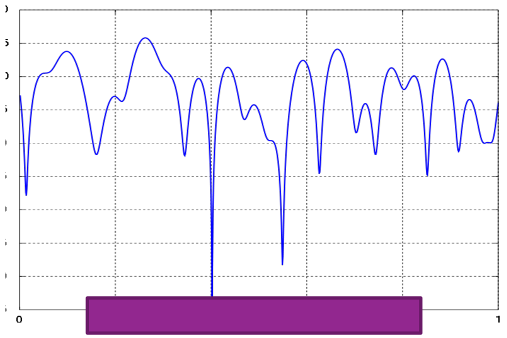
Ergodic Capacity

- Fast fading model: $r[n] = h[n]s[n] + w[n],\ \ w[n] \sim CN(0, N_0)$
  - Channel gains $h[n]$ are i.i.d. with some distribution
- Ergodic capacity: Theoretical maximum rate per symbol
  - Assume average transmit power limit $E|s[n]|^2 = E_s$
  - Maximum taken over all codes and blocklength
  - No computational limits
- Theorem: Ergodic capacity of a fast fading channel is:

$$\boxed{C = E[\log(1+\gamma)], \qquad \gamma = \frac{|h|^2 E_s}{N_0}}$$

  - Value is in bits per symbol
  - Expectation is over channel distribution h

SNR



Time

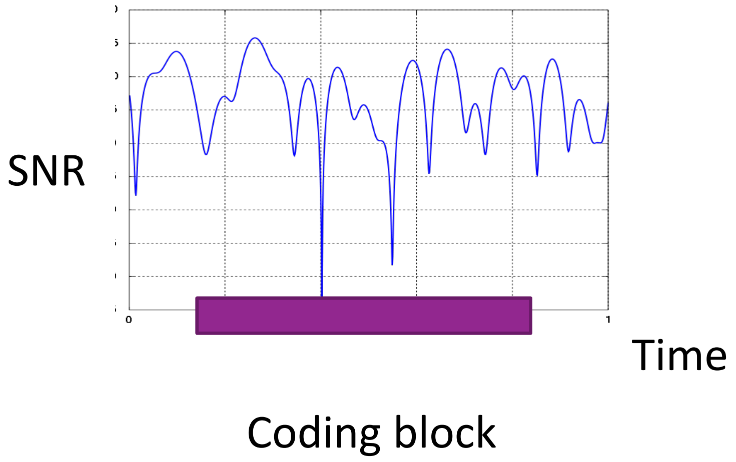
Coding block

Shannon Ergodic Capacity Key Remarks

- From previous slide, ergodic capacity is:

$$C = E[\log(1+\gamma)], \qquad \gamma = \frac{|h|^2 E_s}{N_0}$$

- Theoretical result: Needs infinite computation and delay
  - We will look at performance of real codes next
- TX does not need to know channel h!
  - But RX must estimate and use this channel.
  - We will see RX design is critical
- If TX knew the channel, it could get theoretically get slightly higher rate
  - Uses a method called water-filling
  - Place more power on symbols with better SNR.
  - Gain is not typically large and rarely used in practical wireless systems

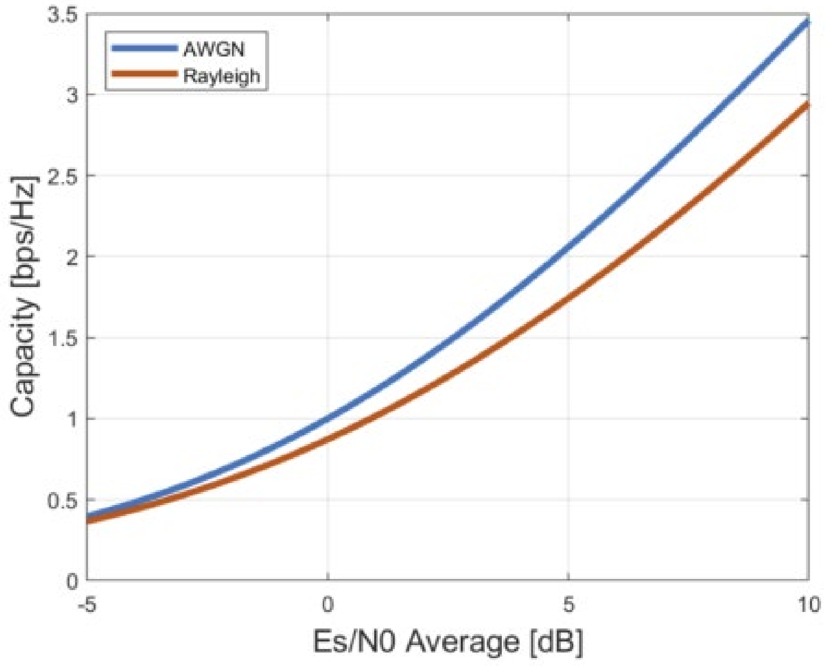


Comparing Ergodic and Flat Capacity

- Fading capacity is always lower than flat fading
  - Keeping the same average SNR the same
  - This fact follows from Jensen's inequality:

$$C = E[\log(1+\gamma)] \le \log\big(1 + E(\gamma)\big) = C_{AWGN}$$

- But gap is not that large at low to moderate SNRs
  - See graph to the right. Loss of only 1-2 dB in
- Conclusions:
  - We should try to code over large number of fading realizations
  - In this case, the capacity loss is theoretically small
  - Much better than the case of uncoded modulation
- We will look at practical codes next



In-Class Exercise

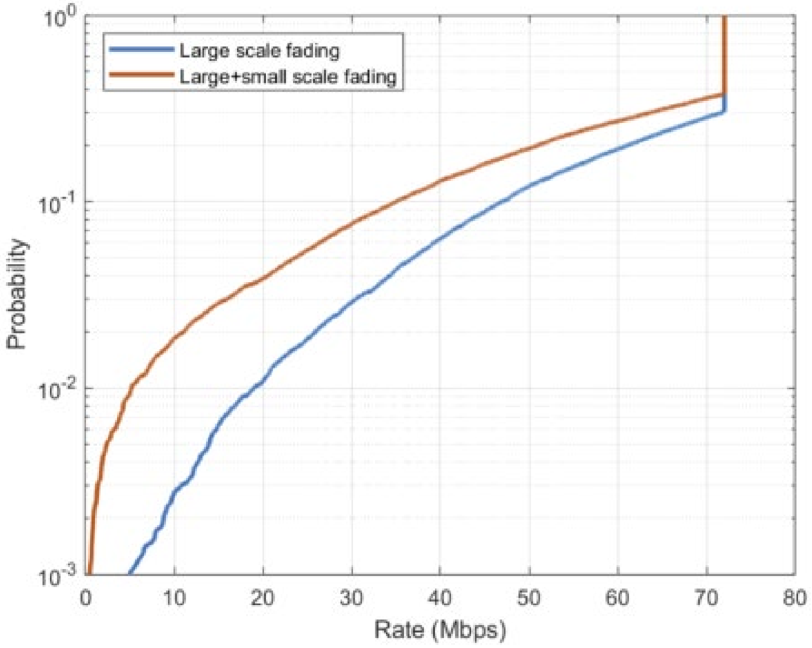
Problem 2: Outage Capacity in an Indoor Environment

In this problem, we will estimate the outage capacity in an indoor setting. Our goal is to look at the effects of both large-scale and small-scale fading.

First generate `nx` locations in a box of size 30 x 40 m representing locations in some large indoor environment. Assume an access point is located at the origin and compute a vector, `dist`, representing the distance in meters from the AP to each location.

```
% Parameters
nx = 10000;
xmax = [30,40];
d = 2;
```

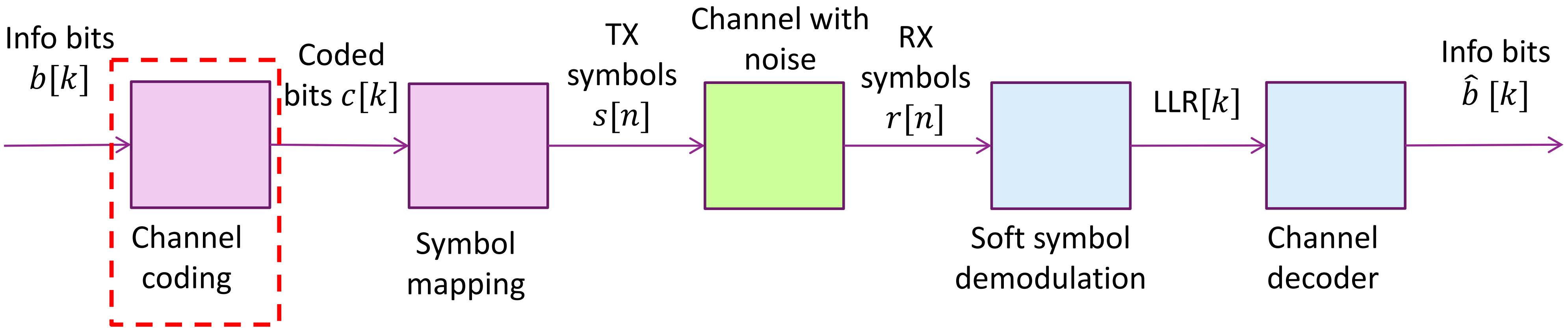
- Indoor environment
- Look at large scale and small-scale fading



Outline

- Uncoded Modulation over Fading Channels
- Outage Probability and Ergodic Capacity
- Review: Coding over an AWGN Channel
- Coding over Fading Channels

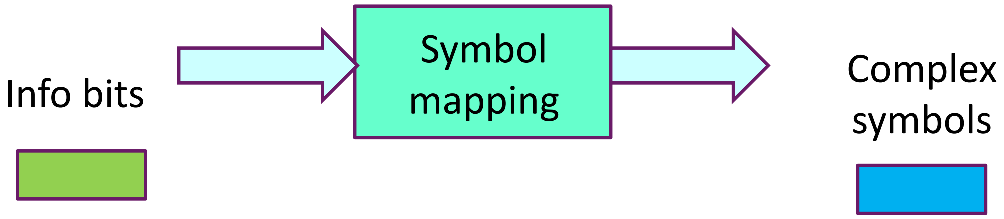
Coded Communication on an AWGN Channel



- We first review channel coding on a flat channel:

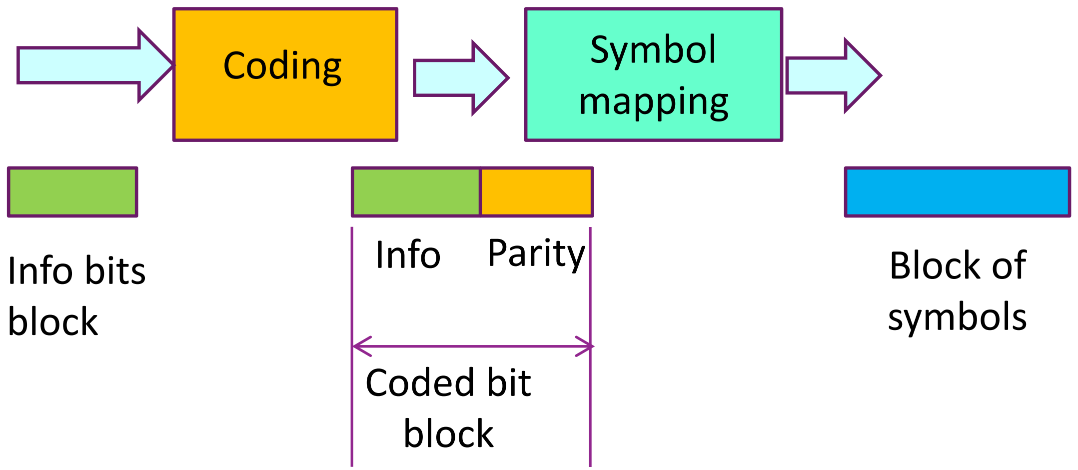
$$r[n] = s[n] + w[n], \qquad w[n] \sim CN(0, N_0)$$

Uncoded vs. Coded Modulation



Uncoded Modulation:

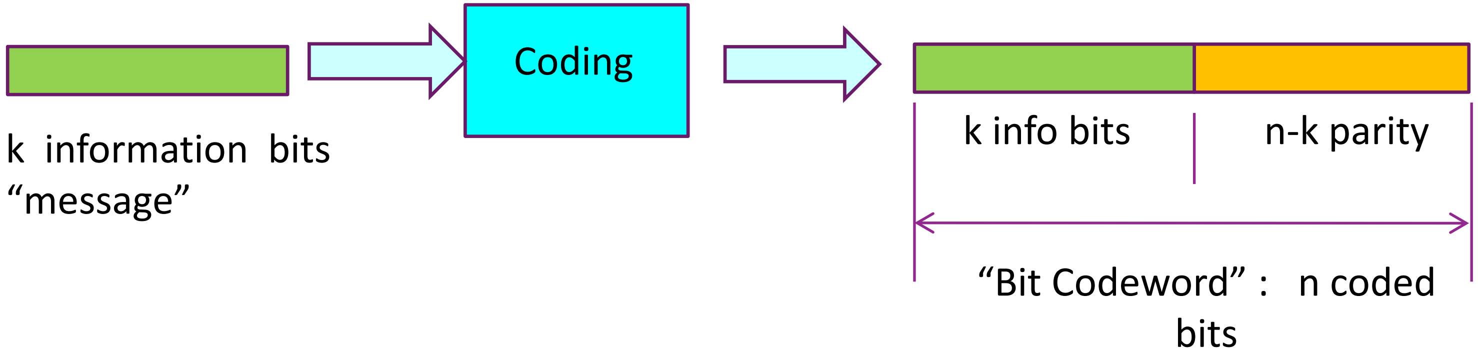
- Modulate raw information bits
- One symbol at a time.
- Any symbol is in error,   data packet is lost!



Coded modulation:

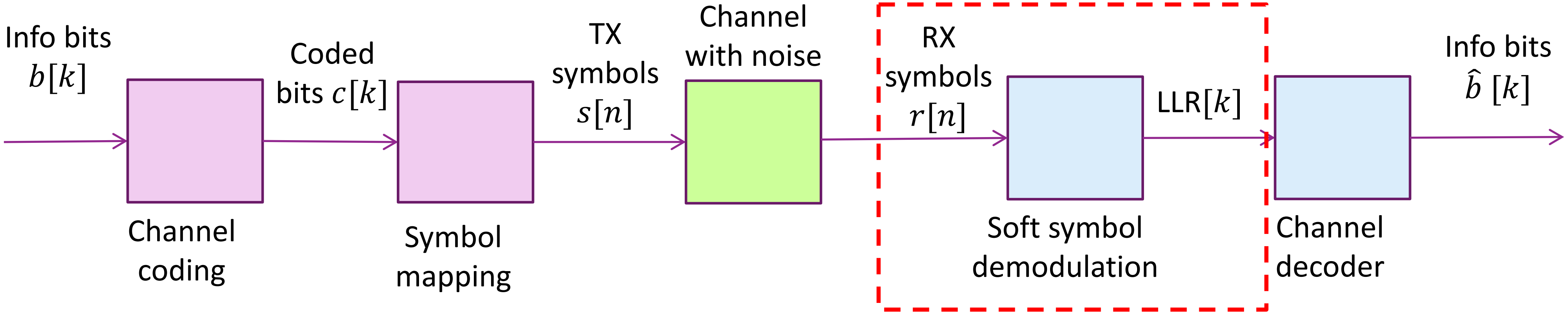
- Transmit in blocks (also called frames)
- Add extra parity bits to each block for reliability
- Decode entire block together

Key Parameters of Block Codes



- An (n, k) block code has:
  - k = number of information bits (input block size)
  - n = number of coded bits (output block size)
  - $n - k$ = number of additional bits, typically parity
  - R_{cod} = coding rate = k/n.
- Typical values in wireless:
  - Block size: $k = 100$ to 10000
  - Code rate: $\frac{1}{3}$ to $\frac{5}{6}$

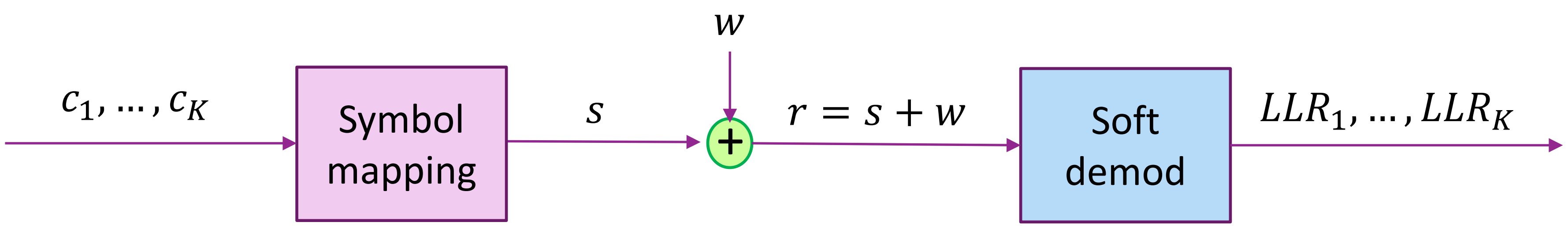
Coded Communication on an AWGN Channel



- We first review channel coding on a flat channel:

$$r[n] = s[n] + w[n], \qquad w[n] \sim CN(0, N_0)$$

Soft Symbol Demodulation

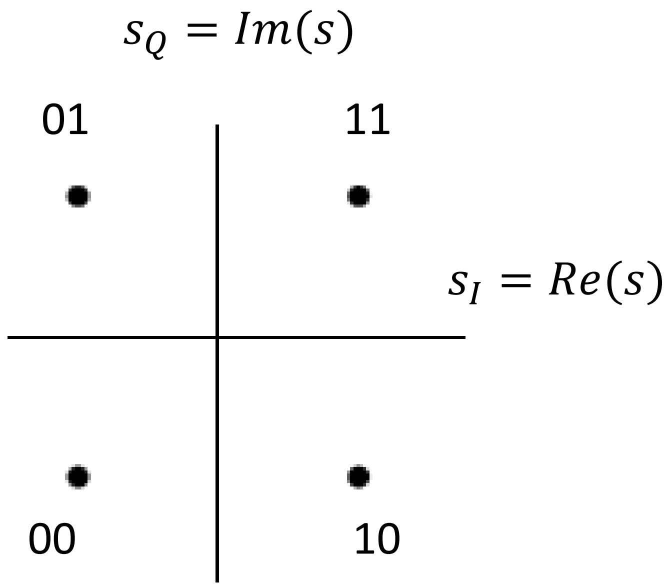


- Set-up: Coded bits $(c_1, \ldots, c_K)$ get mapped to symbol s
  - Receive $r = s + w, \quad w \sim CN(0, N_0)$
- Uncoded systems use hard decision detection:
  - Estimate bits $(\hat{c}_1, \ldots, \hat{c}_K)$ from symbol s
  - Makes a discrete decision.
- Coded systems generally use soft decision demodulation:
  - Output log likelihood ratios: $LLR_k = \ln \frac{P(r|c_k=1)}{P(r|c_k=0)}$
  - LLR_k positive $\Rightarrow c_k = 1$ more likely
  - LLR_k negative $\Rightarrow c_k = 0$ more likely

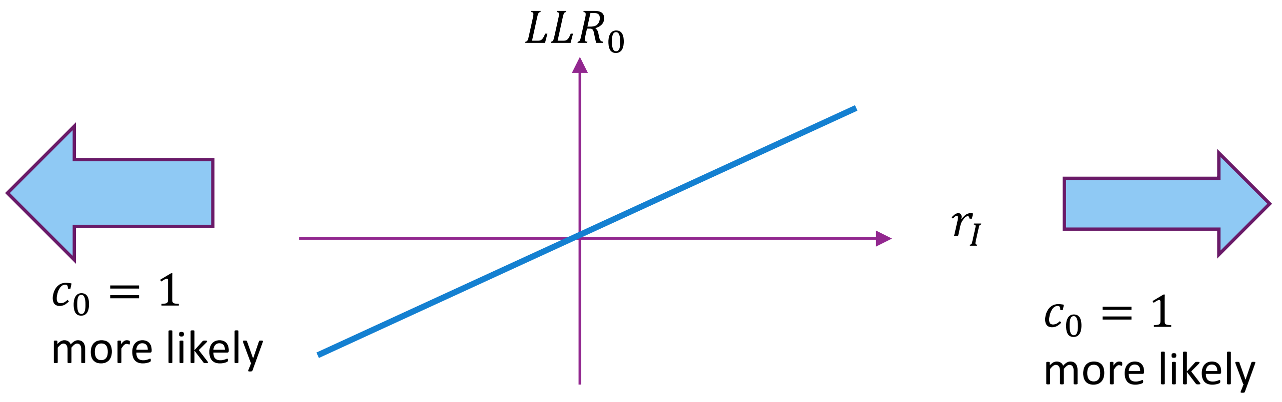
LLR for QPSK

- TX symbol: $s = \pm A \pm iA,\ \ A = \sqrt{\frac{E_s}{2}}$
- RX symbol: $r = s + w,\ \ \ w \sim CN(0, N_0)$
- LLR for bit c_0
  - $s_I = Re(s) = \begin{cases} A & c_0 = 1 \\ -A & c_0 = 0 \end{cases}$
  - $r_I = s_I + w_I,\ \ \ w_I \sim N(0, \frac{N_0}{2})$ [Each dim has $\frac{N_0}{2}$]
  - Likelihood: $p(r_I|s_I) = \frac{1}{\sqrt{\pi N_0}} \exp\left[-\frac{1}{N_0}(r_I - s_I)^2\right]$
  - $LLR_0 = \ln \frac{p(r_I|c_0 = 1)}{p(r_I|c_0 = 0)} = \ln \frac{p(r_I|s_I = A)}{p(r_I|s_I = -A)}$
  - With some algebra: $LLR_0 = \frac{4Ar_I}{N_0} = \frac{4}{N_0}\sqrt{\frac{E_s}{2}}\, r_I$

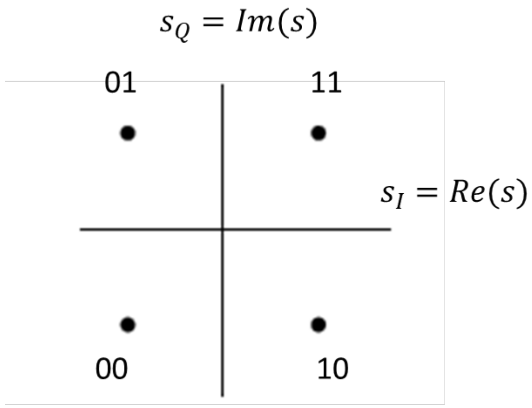
Mapping of bits (c_0, c_1)



QPSK LLR Visualized



Mapping of bits (c_0, c_1)

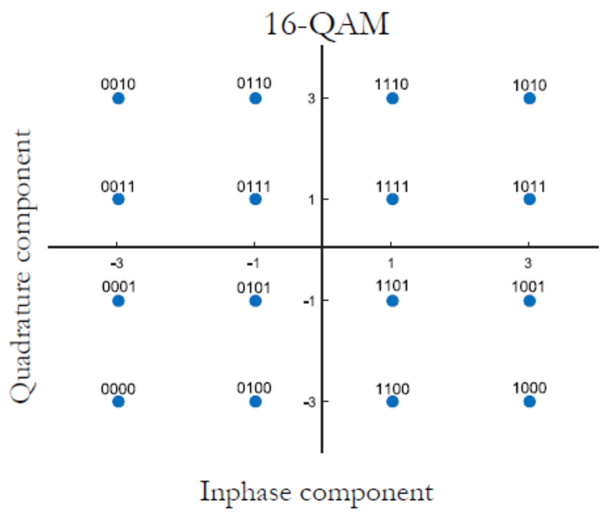


- LLR for c_0 is: $LLR_0 = \frac{4}{N_0}\sqrt{\frac{E_s}{2}}r_I$
- LLR for c_1 is: $LLR_1 = \frac{4}{N_0}\sqrt{\frac{E_s}{2}}r_Q$

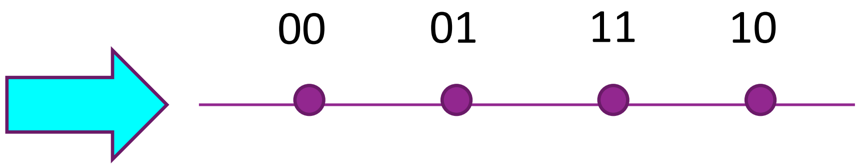
High Order Constellations

- ❑Higher order constellations (eg. 16- or 64-QAM)
- ❑Each constellation r is a point is a function of multiple bits.
- ❑Example: For 16-QAM
  - In phase dimension r_I depends on bits (c_0, c_1)
- ❑Cannot compute LLR on an individual bit directly

Mapping of bits (c_1, c_2, c_3, c_4)



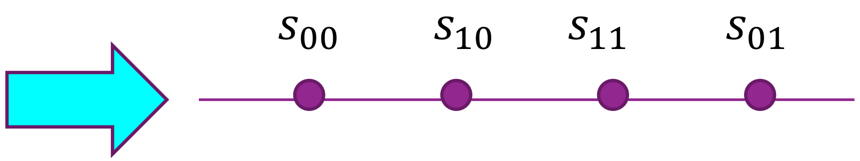
Two bits: (c_1, c_2)



$r = s + n$

High Order Constellations

Two bits:
(c_1, c_2)



$r = s + n$

- To create LLRs for individual bits use total probability rule:

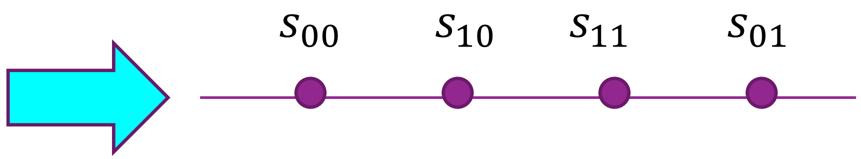
$$p(r|c_1) = \frac{1}{2}\big(p(r|c_1, c_2 = 0) + p(r|c_1, c_2 = 1)\big)$$

- Resulting bitwise LLR:

$$LLR \text{ for } c_1 = \log \frac{p(r|c_1, c_2 = 1{,}0) + p(r|c_1, c_2 = 1{,}1)}{p(r|c_1, c_2 = 0{,}0) + p(r|c_1, c_2 = 0{,}1)}$$

High Order Constellations

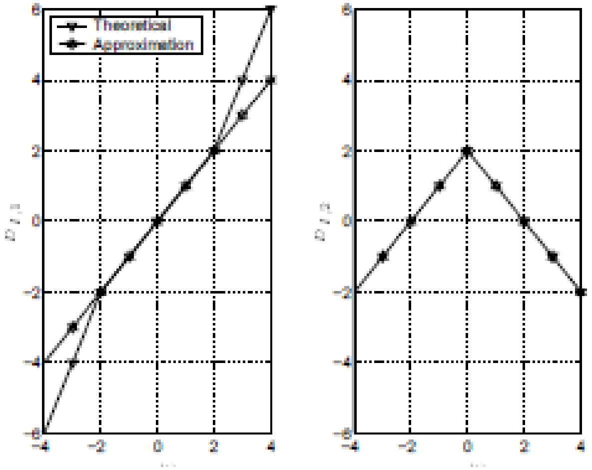
Two bits:
(c_1, c_2)



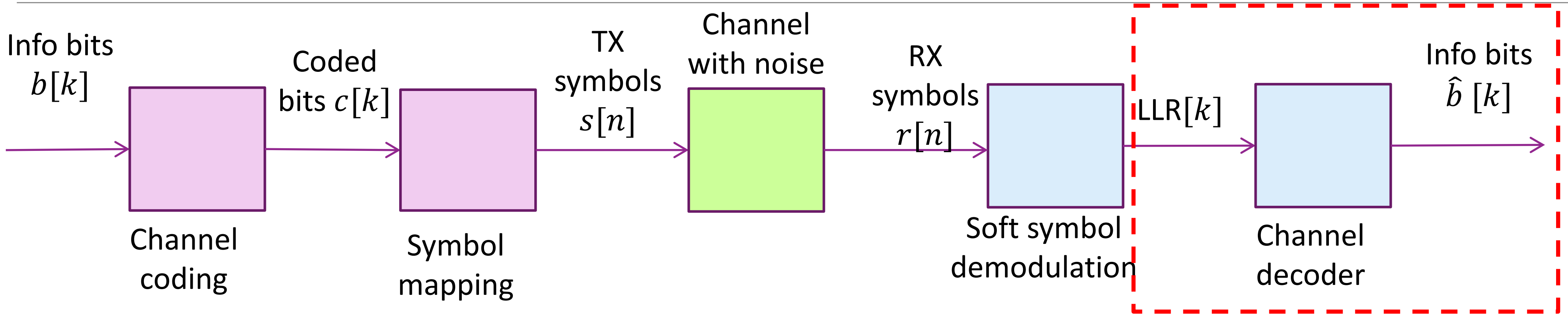
$r = s + n$

- ❑LLRs can have irregular shapes
- ❑Not simple linear function as in BPSK / QPSK case
- ❑Often use approximations
- ❑More info: Caire, Taricco and Biglieri, “Bit-Interleaved Coded Modulation," 1998.

LLR for c2 LLR for c1



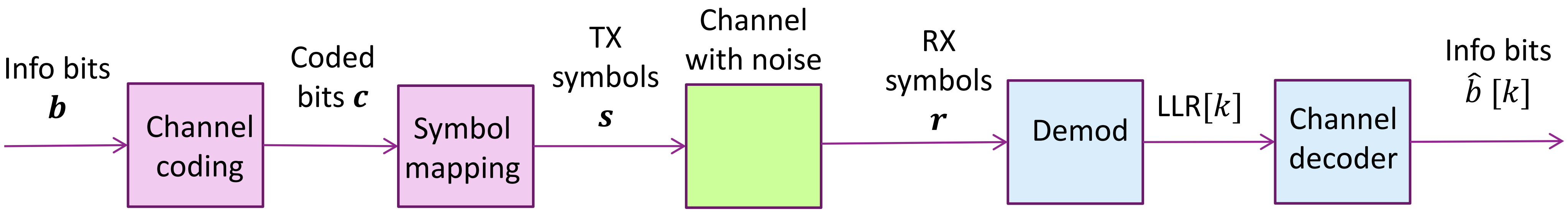
Coded Communication on an AWGN Channel



- We first review channel coding on a flat channel:

$$r[n] = s[n] + w[n], \qquad w[n] \sim CN(0, N_0)$$

Maximum Likelihood Channel Decoding



- Channel coding: Information block: $\boldsymbol{b} = (b_1, \ldots, b_K)$ generates a codeword $\boldsymbol{c} = (c_1, \ldots, c_N)$
- Receiver gets a vector $\boldsymbol{r} = (r_1, \ldots, r_L)$, $L =$ number of complex modulation symbols
- Channel decoder: Goal is to estimate $\boldsymbol{b}$ (or equivalently $\boldsymbol{c}$) from $\boldsymbol{r}$.
- Ideally will use maximum likelihood decoding:

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \log p(\boldsymbol{r}|\boldsymbol{c})$$

  - Finds the codeword that is most likely given the receive vector

Decoding via the LLRs, Part 1

- Channel decoding: Ideally select codeword to maximize $\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \log p(\boldsymbol{r}|\boldsymbol{c})$
- Equivalently we can maximize: $\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} [\log p(\boldsymbol{r}|\boldsymbol{c}) - \log p(\boldsymbol{r}|\boldsymbol{c} = \boldsymbol{0})]$
  - Highest likelihood relative to the all zero codeword
- Assume likelihood factors as: $\log p(\boldsymbol{r}|\boldsymbol{c}) = \sum_{n=1}^{N} \log p(r_{\sigma(n)}|c_n)$
  - $r_{\sigma(n)}$ is the complex modulation symbol containing coded bit c_n
- Hence, objective is:
  - $\log p(\boldsymbol{r}|\boldsymbol{c}) - \log p(\boldsymbol{r}|\boldsymbol{c} = \boldsymbol{0})$

$$= \sum_{n=1}^{N} \left[\log p\left(r_{\sigma(n)}\middle|c_n\right) - \log p\left(r_{\sigma(n)}\middle|c_n = 0\right)\right] = \sum_{n=1}^{N} \log \frac{p\left(r_{\sigma(n)}\middle|c_n\right)}{p\left(r_{\sigma(n)}\middle|c_n = 0\right)}$$

Decoding via the LLRs, Part 2

- From previous slide $\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \sum_{n=1}^{N} \log \frac{p\left(r_{\sigma(n)} \middle| c_n\right)}{p\left(r_{\sigma(n)} \middle| c_n = 0\right)}$
- But since $c_n = 0$ or $c_n = 1$:

$$\log \frac{p\left(r_{\sigma(n)} \middle| c_n\right)}{p\left(r_{\sigma(n)} \middle| c_n = 0\right)} = c_n \log \frac{p\left(r_{\sigma(n)} \middle| c_n = 1\right)}{p\left(r_{\sigma(n)} \middle| c_n = 0\right)} = LLR_n$$

- Hence, the channel decoder can find the codeword by maximizing:

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \sum_{n=1}^{N} c_n LLR_n$$

Decoding Complexity

- Channel decoding ideally selects codeword

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \sum_{n=1}^{N} c_n LLR_n$$

- Brute force optimization is exponentially difficult:
  - Suppose the information block is $\boldsymbol{b} = (b_1, \ldots, b_K)$
  - Each $\boldsymbol{b}$ generates one codeword $\boldsymbol{c} = (c_1, \ldots, c_N)$
  - Optimization must, ideally, search over 2^K possible codewords $\boldsymbol{c}$
  - Computationally impossible

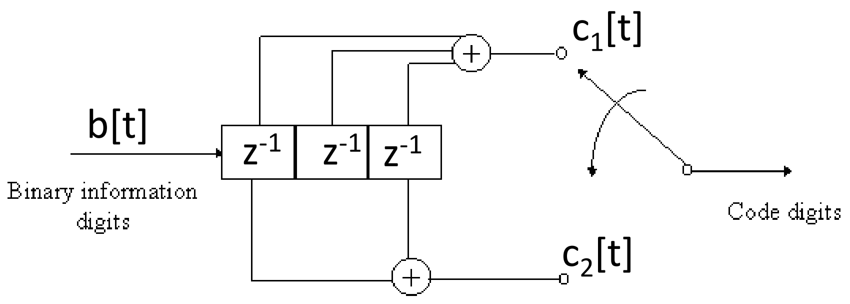
- Coding design requires searching over coding mechanisms with:
  - Computationally tractable decoding
  - But still have good performance

Quest for the Shannon Limit

- Shannon capacity formula and random codes, 1948.
  - Determines the capacity, but no practical code to achieve it.
- Hamming (7,4) code, 1950
- Reed-Solomon codes based on polynomials over finite fields, 1960
  - Used in Voyager program, 1977. CD players, 1982.
- Convolutional codes.
  - Viterbi algorithm, 1969. Widely used in cellular systems. (Viterbi later invents CDMA and founds Qualcomm)
  - Typically, within 3-4 dB of capacity
- Turbo codes, Berrou, Glavieux, Thitimajshima, 1993.
  - Able to achieve capacity within a fraction of dB.
  - Adopted as standard in all 4G and 5G cellular systems by the late 1990s.
- LDPC codes
  - Similar iterative technique as turbo codes. Re-discovered in 1996.
  - Used in 5G systems

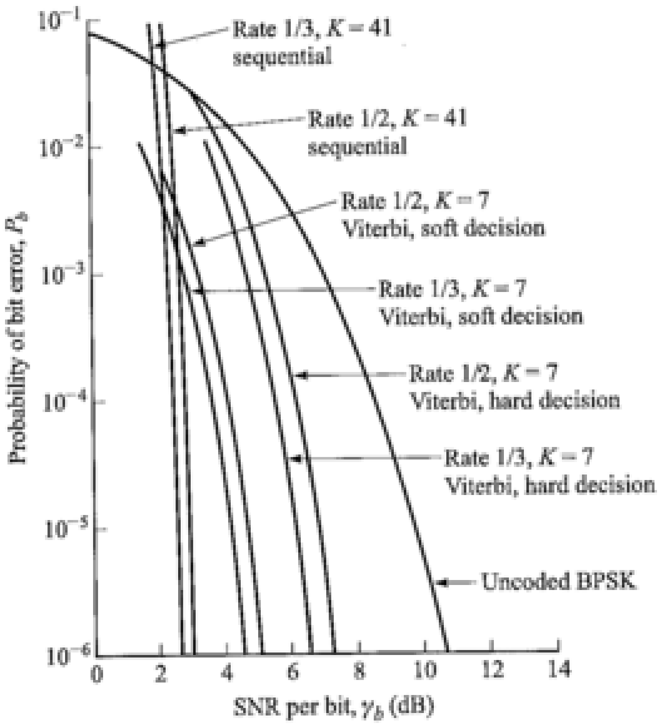
Convolutional Codes

- Encode data through parallel binary (usu. FIR) filters
- Example convolutional code:
  - Rate = ½ (two output bits ($c1[t], c2[t]$) for each input bit $b[t]$.
  - Constraint length *K=3* (size of shift register)
  - Additions are modulo two
- Benefits:
  - Easy to implement, good performance
  - Can be decoded with Viterbi algorithm
    Iterative procedure similar to dynamic programming procedure
  - See digital comm class for more details



$$c_1[t] = b[t] + b[t-1] + b[t-2]$$
$$c_2[t] = b[t] + b[t-2]$$

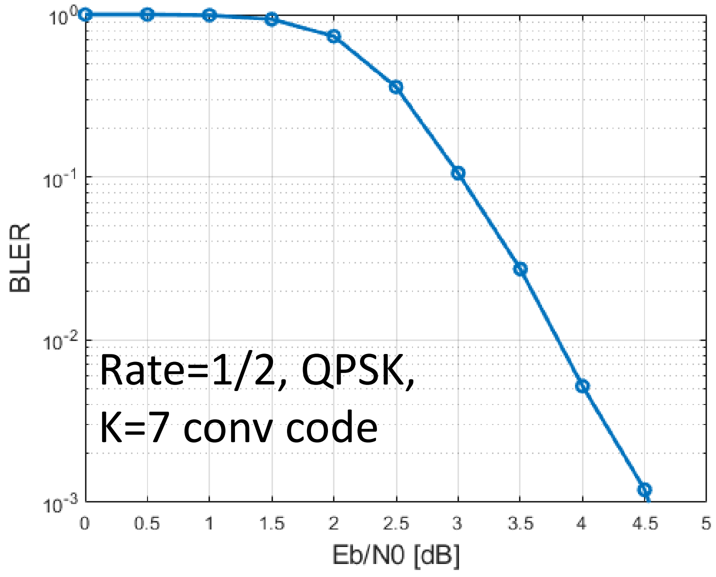
Convolutional Code Performance



- Convolutional codes performance:
  - > 5 dB better than uncoded BPSK at low BER
- Only moderate constraint length (K=7) needed
- Source: Proakis, "Digital communications"

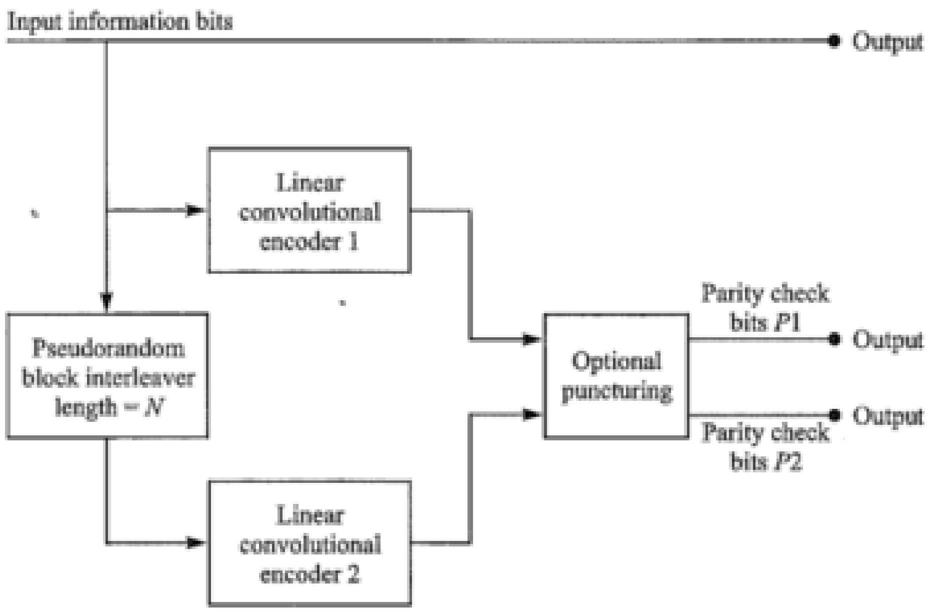
Simulation in MATLAB

- MATLAB has excellent tools
  - Conv encoder / decoder
  - LLR
- See demo



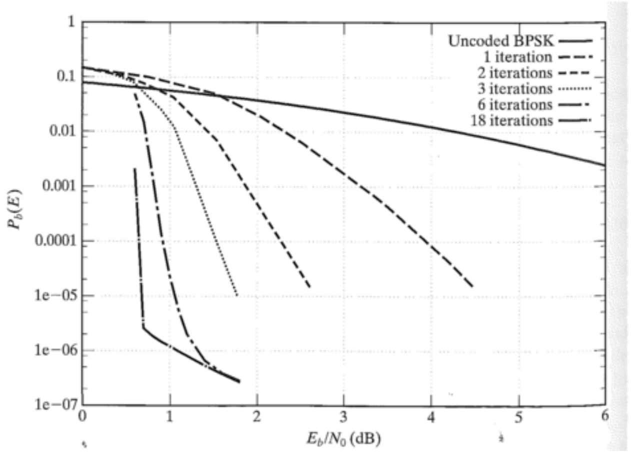

```
% Generate random bits
bitsIn = randi([0,1], nbits, 1);

% Convolutionally encode the data
bitsEnc = convEnc.step(bitsIn);

% QAM modulate
txSig = qammod(bitsEnc, M,'InputType','bit','UnitAveragePower',true);

% Add noise
rate = nbits/length(bitsEnc);
Es = mean(abs(txSig).^2);
EsN0 = EbN0 + 10*log10(rate*bitsPerSym);
chan = comm.AWGNChannel('NoiseMethod', 'Signal to noise ratio (Es/No)', ...
    'EsNo', EsN0, 'SignalPower', Es);
rxSig = chan.step(txSig);

% Compute LLRs
noiseVar = Es*db2pow(-EsN0);
llr = qamdemod(rxSig,M,'OutputType','approxllr', ...
    'UnitAveragePower',true,'NoiseVariance', noiseVar);

% Run Viterbi decoder.  We remove the tail bits
bitsOut = convDec.step(llr);
bitsOut = bitsOut(1:nbits);
```

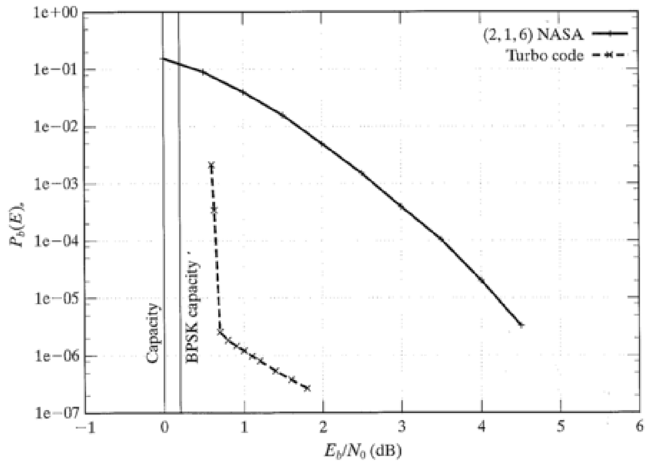
Turbo Codes



- Turbo codes:
  - Concatenation of two convolutional codes Typically IIR and short (K=3)
  - Interleaver: Randomly permutes the input bits
- Output
  - Input bit, and
  - Parity bits from each convolutional encoder
  - With no puncturing R=1/3
- Discovered in 1993, ,
  - Berrou, Glavieux, Thitimajshima, 1993.
  - Able to achieve capacity within a fraction of dB.
- Used in 3G and 4G standards

Turbo Code Iterative Decoding



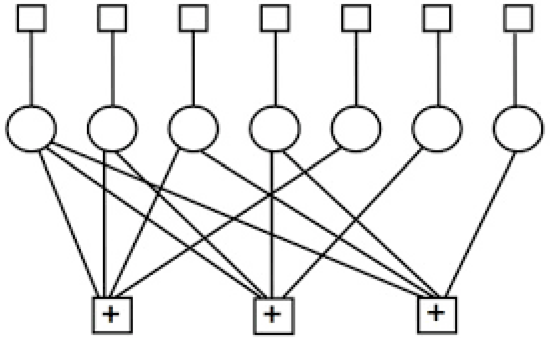


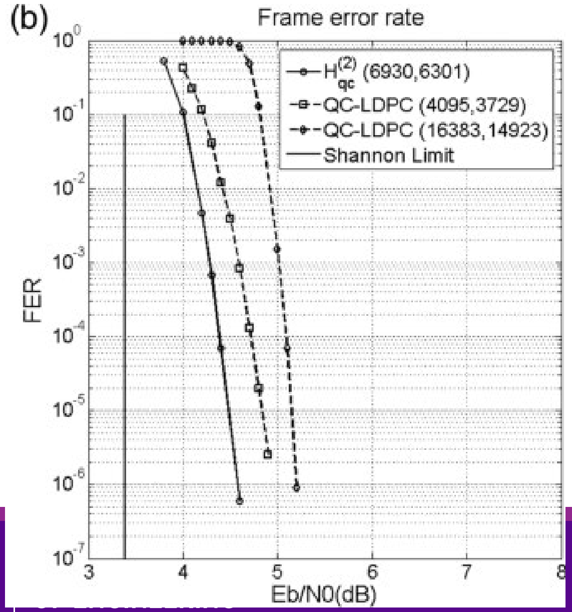
- Turbo decoder uses an iterative message passing
  - Decode each convolutional coder one at a time
  - Use posterior information of one code as prior for the other
- Good performance in small number (usu. ~8) iterations
  - Typically use short codes (K=3).
  - Complexity similar to convolutional codes
- Close to Shannon capacity
  - Much better than convolution codes

Source: Lin, Costello, “Error Control Coding”

LDPC Codes

LDPC Graph





- Code defined by a bipartite graph
  - Connects n coded bits and $n-k$ parity bits
  - Data k information bits
- Also use a message passing decoder
  - Based on graphical models
- Obtains excellent performance
  - Lower complexity than turbo decoder
  - Good for very high data rate applications
- Used in 802.11ad and 5G New Radio

In-Class Problem

Problem 3: NR LDPC Coding

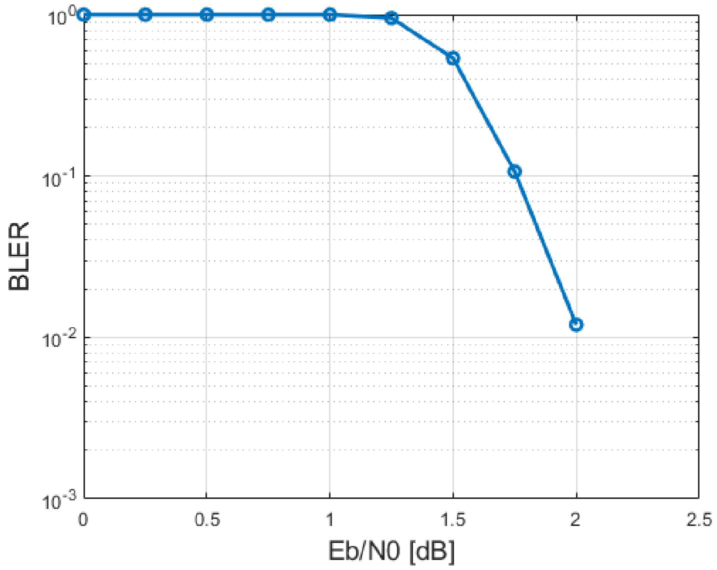
In this problem, we will simulate the LDPC encoding and decoding used in the 5G NR standard. The MATLAB 5G Toolbox has an amazingly good implementation of this code, so we can just call it. In the NR LDPC code, the number of input bits is given by:

```
nbitsIn = nrows*nlift;
```

where nrows is the number of rows in the LDPC base graph, and `nlift` is the so-called lifting factor which expands the graph to different block sizes. We will use the following parameters:

```
bgn = 1;      % LDPC base graph number (1 or 2)
nrows = 22;  % number of rows in the base graph
nlift = 128; % lifting factor
nbitsIn = nrows*nlift;  % number of input bits
maxNumIters = 8;        % max number of LDPC decode iterations
```

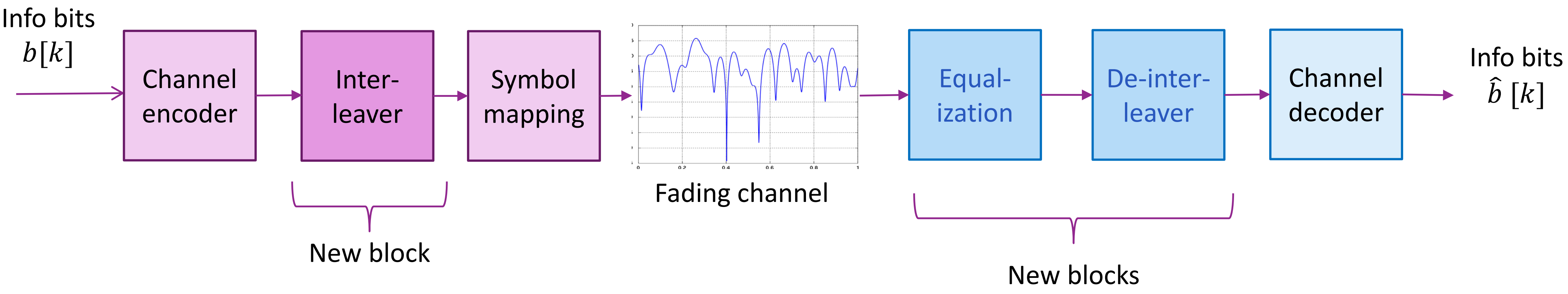
- Simulate the commercial 5G NR LDPC code
  - A rate 1/3 code
  - Much better performance than convolutional code



Outline

- Uncoded Modulation over Fading Channels
- Outage Probability and Ergodic Capacity
- Review: Coding over an AWGN Channel
- Coding over Fading Channels

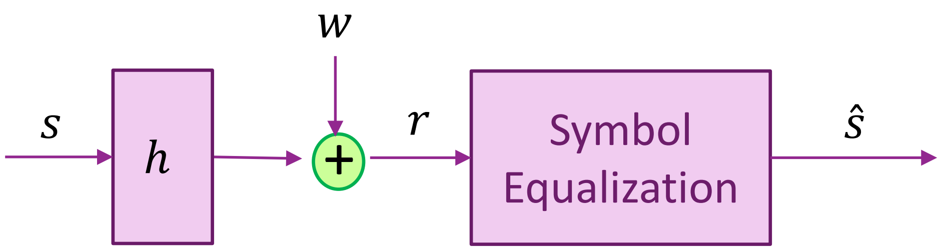
Coded Communication on a Fading Channel



- Now consider fading channel: $r[n] = h[n]s[n] + w[n], \quad w[n] \sim CN(0, N_0)$
- To handle fading we need to introduce a few new blocks
  - Interleaving and de-interleaving
  - Equalization

MMSE Symbol Equalization

- Received noisy symbol:
$$r = hs + w, \qquad w \sim CN(0, N_0)$$
- MMSE estimation:
  - Use linear estimate $\hat{s} = \alpha r$
  - Select α to minimize $E|s - \hat{s}|^2 = E|s - \alpha r|^2$
- Resulting estimate (shown with simple algebra):
  - Estimate: $\hat{s} = \alpha r, \quad \alpha = \frac{E_s h^*}{|h|^2 E_s + N_0}$
  - Noise variance: $E|s - \hat{s}|^2 = \frac{E_s N_0}{|h|^2 E_s + N_0}$
- Provides lower noise estimate than channel inversion

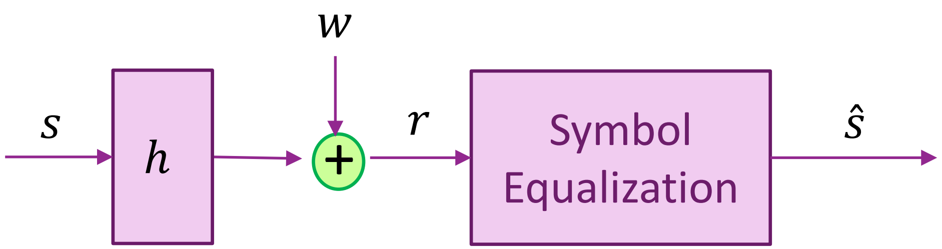


Symbol Equalization via Inversion

- Received noisy symbol:

$$r = hs + w, \qquad w \sim CN(0, N_0)$$

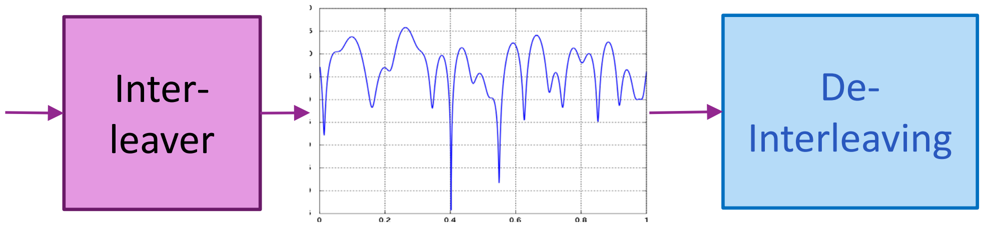
- Symbol equalization:
  - Estimate s from r
  - Also obtain a noise estimate (needed for LLR)



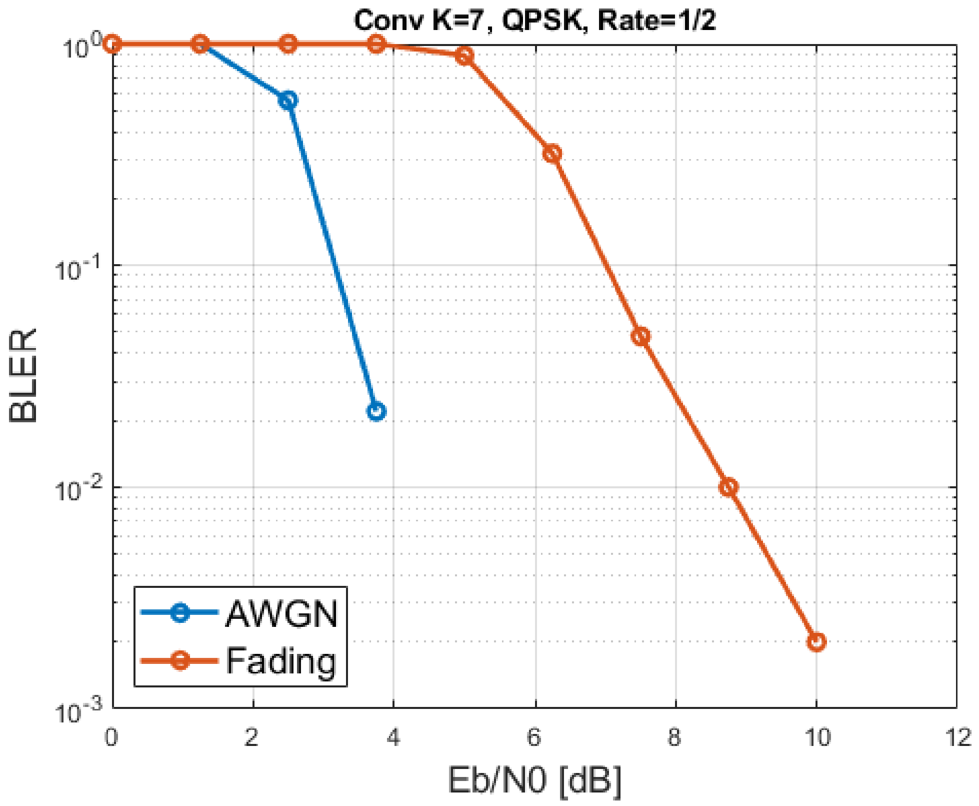
- Channel inversion:
  - Symbol estimate $\hat{s} = \frac{r}{h} = s + v, \;\; v = \frac{w}{h}$
  - Noise estimate: $E|v|^2 = \frac{1}{|h|^2} E|w|^2 = \frac{N_0}{|h|^2}$

Interleaving and De-Interleaving

- Problem: Fading is correlated in time
  - Will result in many consecutive faded bits
  - Many codes perform poorly if errors are together
- Interleaver
  - Shuffles the bits before symbol mapping
  - De-interleaving is performed on LLRs
  - Randomizes locations of errors
  - Removes time correlations
- Many interleavers used in practice
  - Random interleaver (with some seed at TX and RX)
  - Row-column interleavers…



Simulation



- Simulation:
  - Convolutional code, rate $\frac{1}{2}$ with QPSK
  - Constraint length $K = 7$
  - Plotted is block error rate (BLER) vs. $\frac{E_b}{N_0}$
- Gap between AWGN and fading:
  - Approximately 4 dB at BLER = 10^{-2}
  - Much smaller gap than uncoded modulation

Simulating in MATLAB

Transmitter and Channel Fading

```
% Generate random bits
bitsIn = randi([0,1], nbits, 1);

% Convolutionally encode the data
bitsEnc = convEnc.step(bitsIn);

% Random interleaver
state = randi(2^16,1);
bitsInt = randintrlv(bitsEnc,state);

% QAM modulate
txSig = qammod(bitsInt,M,'InputType','bit','UnitAveragePower',true);

% Add fading
nout = length(txSig);
if fading
    h = sqrt(1/2)*(randn(nout,1) +1i*randn(nout,1));
else
    h = ones(nout,1);
end
rxSig0 = h.*txSig;
```

Channel Noise and Receiver

```
% Pass through AWGN channel
rxSig = chan.step(rxSig0);

% MMSE Equalize
wvar = Es*10.^(-0.1*EsN0);
z = conj(h).*rxSig ./ (abs(h).^2 + wvar);
svar = 1;
noiseVar = wvar*svar./(wvar + svar*abs(h).^2);

% Compute LLRs
llrInt = qamdemod(z,M,'OutputType','approxllr', ...
    'UnitAveragePower',true,'NoiseVariance', noiseVar);

% De-interleave
llr = randdeintrlv(llrInt,state);

% Run Viterbi decoder
bitsOut = convDec.step(llr);
bitsOut = bitsOut(1:nbits);
```

Summary

- **Fading**: Causes variations in SNR
- **Uncoded modulation**:
  - Dramatically increases error rate
  - Must add significant fade margin
- **Coding with slow fading**
  - All symbols are faded together
  - Fade margin still necessary
- **Coding with fast fading**
  - Can greatly mitigate fading
  - Recover faded bits with redundancy
  - But needs to encoded over many independent fades
  - Transmit over many coherence or bandwidth