

Homework Help - Problem Set 3

Coherence Time This problem attempts to familiarize the use of basic fading parameters for a mobile wireless channel, and build some intuition on moving receivers.

- a. The data rate calculation is the same as always (b/T where $b = \log_2(|M|)$), so this should be trivial for you at this point.
- b. The average error probability needs to incorporate the average over the fading distribution. The non-fading P_e is the simple Q-function expression for SQ QAM for which comfort grows at this point. The other formula comes from L5:14 or from Section 1.6.2.1 where you should recognize that $\mathcal{E}_{x,g} \rightarrow \kappa \cdot SNR$, where the κ is the SQ QAM scaling constant. Your intuition should tell you which one is worst, but get both numbers and see by how much. Don't be surprised if it is significant.

side comment: Even though significantly worse, it could well be that the outage probability (not computed here and depends upon specified acceptable gain level) might be acceptable with just a little more SNR (or a reduction of rate to 16 QAM). What this suggests is that time-varying systems may benefit from an outer concatenation of another code that is often able to correct any data lost to outages, but may well be working well (like your no-fading answer) a significant fraction of the time.

- c. The max Doppler frequency is the fraction of (vehicle speed/speed of light) times the carrier frequency. Faster speed, or higher carrier, create a faster time variation.
- d. The coherence time is basically the time that we can trust the channel to be roughly stationary. There is a formula for this (an approximation) on L5:17 as well as in Section 1.6. We probably want to have many transmissions within the coherence time, which then allows us an ability to learn the roughly stationary channel gain/SNR over that period. That means wider bandwidth (more symbols per second) can be an advantage - this ultimately was a component in wireless systems' increasingly using more bandwidth per user, although perhaps multiplexing several into a common wider band in ways that allow each gain to be learned over some good choice of packet period. Your answers here might suggest why a 6 GHz carrier might use such a large symbol rate of 300 MHz.
- e. This kind of puts your intuition to work. $27 \text{ dB} = 500$. A very rudimentary Chapter 9 has been added to web site. Look at the channel ID

section if you want the full theory. However, for here, we should understand that $\sigma^2 = \frac{\mathcal{E}_{x,g}}{500}$ on average. Since it varies, we must estimate this gain. That estimate will be $\mathcal{E}_{x,g} + \Delta\mathcal{E}_{x,g}$. The estimate's offset $\Delta\mathcal{E}_{x,g}$ will basically reduce as $\mathcal{E}_{x,g}/L$ where L is number of measurements of g (see Section 9.1.1 if you want more complete explanation of gain estimation). We want $\bar{\mathcal{E}}_{x,g}/(\sigma^2 + \Delta\mathcal{E}_{x,g}) \geq 10^{2.6}$ or 26 dB (so less than 1 dB loss). Approximating $\sigma^2 = 10^{-2.7} \cdot \bar{\mathcal{E}}_{x,g}$, then basically $1/500 + 1/L \leq 10^{-2.6}$. The solution is roughly $L = 2000$. Now you can compare this to the coherence time.

- f. The first term is kind of a gross gain based on distance and obstacles through which waves must pass - any movement is caught in the variance of the fading parameters. This should help you think through it.
- g. This is plugging into the formula given for a_k .
- h. The power delay profile just sums the values in each of the table's columns. There is a normalization to a probability-distribution-like quantity, for which it is ok to use a sum of squared values x probability to get rms delay spread.
- i. if your gut tells you that this should vary over such a wide transmission bandwidth, it is correct. However, you will need to use the formulas from L5:17 on rms delay spread and coherence bandwidth to compare to the 300 MHz symbol rate and see how much. Is the coherence bandwidth less than 300 MHz? If so, this is frequency-selective fading.

Explore Fading Impact This problem starts like the previous one in PS3, but with 16 QAM. Parts a and b then refer to the previous help.

- c. At this point, this problem deviates to simulate no fading and fading and checks the formulas versus simulation. You may use the `comm.ErrorRate` facility. You may need to `reset(errrate)` if you reuse this label from earlier assignments. Do not use the `awgn.m` program. Just set up the `Ex` and `N0` directly and create an AWGN like in L3:26. Same commands basically.
- c. The doppler this time requires you to convert units and compute.
- c. The Rayleigh command object's use appears in L5:21 "To generate Rayleigh Fading Outputs" slide. Remember to `reset(rayleighchan)` or whatever your name for the object when you run simulations that have conditions that have changed. You'll need to use the `qammod` function to generate points and then scale them by the `rayleighchan` object's appropriate output. I did 10k channel simulations, each with 1000 transmissions. That did not take very long in matlab. Again, you may use the `comm.ErrorRate` to help count errors. This should match pretty closely Part b's theoretical answer.
- c. The doppler frequency follows the formula $f_d = (\frac{v}{c}) \cdot f_c$. This is easy.
- c. Now we repeat Part c but need to make the channel gain random using the `rayleighchan` program, so rather than just adding noise we take the faded output of `rayleigh chan` and add noise. Try 100k runs instead of

10k, this improves the results without taking too much more time. You should get a match to theory in part b to within $\pm 10 - 15\%$. More accuracy requires even longer runs, which would take too much time.

LLR calculation This problem directly minimizes bit-error probability by computing log likelihood ratios for the 3 bits of input in 8 PAM.

- a. Start by finding $p_{u_i, y}$ in terms of the channel distribution $p_{y/(u_1, u_2, u_3)}$, recognizing that it is the margin distribution that occurs by summing $p_{y, (u_1, u_2, u_3)}$ over the other two bits possible values. What is the input distribution if uniform with 8 points? Use that to simplify.
- b. Since the detector is for a single bit, then how many values can it have? (This is that simple.)
- c. If we sum over the other two bits, how many possibilities? (This is almost as simple.)
- d. Look at the argument in the given expression's exponent term? How many values can there be for this term - compute them. Then add various combinations thereof to get the values for each bit's possible values. You can now see which is largest and implement the MAP/ML detector.
- e. The LLRs simply follow from the ratios of the probabilities just computed for each of the bits. It is possible to have an infinite LLR, which means confidence is very high for the decision. We are now starting to see not only a decision, but also soft information about the decision's reliability for each of the bits.
- f. Investigate or think about these differences in terms of their being exponentiated. What does that mean for some of the terms?
- g. This part of the problem is emphasizing the Gray coding's improvement in that most errors are because the decoder picked the correct PAM value's nearest neighbor.
- h. Try to keep in mind that a binary-bit code would correct up to $\lfloor \frac{d_{free}-1}{2} \rfloor$ errors - However if the code corrected only symbols and an error was made, we're back to the potential of bit errors associated with the PAM mapping instead of the code.

Mapping QAM to DMC This problem uses your established familiarity with an AWGN to investigate its after-decoded use in an outer code through symmetric DMC modelling.

- a. This progresses easily based on the SQ QAM P_e formula that depends on SNR and constellation size. Clearly the 16QAM has more subsymbol values and should have a larger P_e if all are used, which is assumed in this first part.
- b. For the QAM system, it is basically uncoded with 4 bits mapped into 16 symbol values. This basically asks how many real dimensions and views QAM as trivial two-dimensional code. For the bit-level coded system,

each bit is a subsymbol so your answers will change, illustrating different coding views of essentially the same electronics.

- c. Redundancy measures the extra bits in a constellation. It's easy when viewing the code at bit level. It is possible to also view a series of 16QAM choices as subsymbols in a code, and then we would get different values for the redundancy, but that is not what is happening here because the code extends only over 4 bits.
- d. Simple parity forces any codeword to have an even number of 1's in it, so what is d_{free} . Adding an overall parity bit to any code extends it to be an even code where all codewords have even parity.
- e. If this code with described hard ML decoder cannot correct errors, do you expect it to perform better?
- f. This is for the uncoded system, so we count real dimensions for N always, which should make it easy to see for this one symbol to find values.
- g. The previous question's answer is irrelevant, so this tests if you can see that the SDMC inner model's internal parameters are not pertinent to the outer code in this problem.
- h. You may use the symbol rule that $\bar{P}_b \simeq b \cdot P_e$ for this part.
- i. The original system has 6 information bits per byte, so this will limit the redundancy that the outer system can use. Clearly smaller N in the outer system with a large P that does not cause the data rate to decrease.
- j. We can try a few N 's with implied P 's for maximum correct, but not exceeding the redundancy limit. I was able to get close in 4 guesses by just using the function provided. Try a few values and evaluate the P_e . Nearest neighbors are important, which is why the matlab routine is suggested. You may want to visit the expressions in Section 2.2.2.2 for the nearest neighbors $N_e = N_0$ of an MDS code - just use first term here. (Probably using additional terms increases the P_e somewhat although distances are increasing and reduce those additional terms also, but we ignore that in this problem.)
- k. The data rate can be increased by increasing the codeword length by more than 4/3 times the parity increase. Try large N and $P = 16$, see what you get.

Bandwidth Expansion This problem tests the trade between equal factor increases in energy versus bandwidth, as well as design with QAM on AWGN.

- a. The problem statement's hint almost gives away what we want you to do already.
- b. Compare to BPSK with larger symbol rate. You should get a much better design because there is no bandwidth limit, even though the power is limited.
- c. We just ask here for you to compare the BPSK data rate to the 128SQ data rate.

- d. This just restates the rule that you should be by now well understanding.
- e. The G matrix is always $k \times n$, so this should be easy and it is the Hamming code you've seen in class as an example.
- f. The d_{free} calculation for this code is in Lecture 7.
- g. Since this code is being used with a binary system, we will need to expand the clock rate. You may presume the BPSK system is running at some symbol rate. Use of the unused quadrature dimension would reduce the d_{min} by $\sqrt{2}$ so it is equivalent to doubling the bandwidth if the distance is maintained. In other words, coding gain $r \cdot d_{free}$ applies directly to BPSK. Thus, since we have more output bits than input bits, how much does this sampling rate have to increase if we maintain the same data rate R ? This is easy, it is n/k for any binary code.
- h. At constant power, the energy is spread over wider frequency band so the loss is $10 \cdot \log_{10}(r)$.
- i. Fortunately the coding gain more than offsets this in that $d - free > 1/r$ for this code. That is $\gamma = r \cdot d_{free} > 1$.
- j. We're looking for the simple understanding here that it must be possible to increase the clock rate - that is, the extra bandwidth is available for use.
- k. This question causes thought on how far can we extend this "free bandwidth" concept. You know r for any code must not exceed the \bar{C} , and as we increase bandwidth sufficiently even $\bar{C} < 1$. However, R is increasing linearly with $1/T = 2W$ while loss of data rate is only logarithmic with reduced power (SNR). Write the \bar{E}_x in terms of \bar{C} and σ^2 , and divide by \bar{C} - this is lowest energy/bit possible. Let $\bar{C} \rightarrow 0$ as the bandwidth increases to infinity. This will be a finite limit of how much energy must be spent to send a bit.
- l. There are 7 output bits on this code, so we use 128SQ. Unfortunately, the gray code does not work for odd SQ constellations, but the "ideally" here means that the nearest neighbor count can be assumed to be 1-bit error. It can be made close to this, but not actually achieve in practice, but I don't want that detail to get in the way of this problem's analysis. This problem instead wants to make you think, which 16 point subset might the design pick? How much does their intra-set distance increase? It's ok to subtract any nonzero mean from any such subset. What are we left with as best choice? If you are not finding any advantage, then you are on the right track. Just explain it.
- m. Now, you are an interleaver designer. The Gray Coding has a shot now because of the 3 independent codes.
- n. Basically, the entire coding gain of the Hamming code should be possible if we can truly make each use equal to an AWGN. There is still an energy loss from using 128-point constellations, but we are effectively designing a different type of code with the interleaving that indeed can obtain in the limit the 2.4 dB. Even better binary codes can gain more with BICM.