

## Handout 3: Problem Set 1, Due Wed, Jan 21st

Prof: Jeff A. Bilmes <bilmes@ee.washington.edu>

Lecture 3, Jan 12, 2004

### 3.1 Problems from Text

Do problems 2.16, 2.3, 2.5, 2.6, 2.12, 2.10, 2.20 in Cover and Thomas.

### 3.2 Additional Problems

**Problem 0.** The rest of Jensen.

In class, we proved part of Jensen's inequality. Specifically, we proved that if  $f$  is convex and  $X$  is a random variable, then  $Ef(X) \geq f(EX)$ .

We also mentioned but did not prove that if  $f$  is strictly convex, then if  $Ef(X) = f(EX)$ , then  $X = EX$  meaning that  $X$  is a constant.

In this problem, your task is to prove this second condition.

**Problem 1.** Matlab, and entropy of images.

Consider a probability distribution  $p(X)$  over a  $32 \times 32$  pixel image, where  $X$  is a  $32 \times 32$  random matrix. Assume, for this discussion, that there are 8 bit pixels which means that in total there are a large number  $256^{32 \times 32} = 10^{2466}$  possible images that are representable (this is more than the number of atoms in the universe which is estimated to be about  $10^{78}$ , and certainly more than the entire human population has viewed throughout the course of its existence).<sup>1</sup>

When  $p(X)$  is a uniform distribution, then when we sample from such a distribution, the images would look entirely random — the likelihood of ever encountering an “real” image (say of birds, bees, trees, or industrial manufacturing plants), one we would likely see in our world, is exceedingly small. You can create such images in matlab with the following command `imagesc(rand(32,32))`. Note that the probability of seeing the image that you see is the same as that of seeing an image of yourself, when generating such image samples.

Since all the images are equally likely, the entropy of such a distribution is  $H = 8192$  (why?).<sup>2</sup>

Your goal is to design a matlab program which 1) generates random  $32 \times 32$  pixel images, 2) has as low entropy as possible, and 3) still appears to the human perceptual system as random pixels (much like the matlab command `imagesc(rand(32,32))` appears). Argue why the entropy for your scheme must be low. Include in your HW a fair number of samples from your distribution. Also include the matlab code, and make sure to comment the code well. What can you say about the compressibility of such images as compared to `imagesc(rand(32,32))`? Optional and extra credit: Can you compute the entropy for your distribution?

The last part of the question is to imagine that you had a random image generator that generates scenes from the natural world (trees, mountains, etc.). Would you think the entropy of the underlying distribution would be low or high relative to 8192? Explain why?

<sup>1</sup>While the matlab function surely uses more than 8 bits, we describe it in terms of 8 bits just for simplicity. In fact there are many more possible images when the random number generator uses more than 8 bits.

<sup>2</sup>Assume in this problem that `rand()` is a real and not a pseudo random number generator.

**Problem 2.**

This problem should get you thinking about probability as events in event space.

You have a choice of three doors, and you get whatever prize exists behind the one door you choose. Behind one door is that expensive car you always wanted, and behind the other two doors are goats (which you presumably do not want). You choose the first door. Then, a goat is shown to exist behind the third door. You then have the opportunity to change your choice of door. Is it possible to calculate the probability that the car lies behind the second door and should that affect your choice? Explain your answer.

**3.3 Entropy and such.**

**Problem 3.** Probabilities of probabilities:

1) Let  $X \in \{x_1, x_2, x_3\}$  be a random variable with probability mass function  $p(x_1) = 0.1, p(x_2) = 0.2, p(x_3) = 0.7$ . What is the probability that  $p(X) \in [0.15, 0.5]$ ? What is:

$$p\left(\left|\log \frac{p(X)}{0.2}\right| > 0.05\right)$$

**Problem 4.** Expectations of probabilities:

Recall the idea of computing the expectation of a function of  $x$ :

$$E[f(x)] = \sum_x f(x)p(x)$$

As we discussed in class,  $f(x)$  can be any function including one that might even use probability function  $p(x)$  itself.

In class we said the information of an event  $x$  should correspond roughly inversely to  $p(x)$ . We defined the information of an event as  $I(x) = \log 1/p(x)$  and the entropy as the average (or expected) information  $H = E[I(X)]$  in the random variable  $X$ .

1) Give at least one reason why you would *not* want to make the definition  $I(x) \triangleq 1/p(x)$ ?

2) optional and extra credit: suggest another function that might be used for  $I(x)$ . Justify why it might be useful as a measure of the information provided by the event  $x$  that occurs with probability  $p(x)$ .

**Problem 5a.**

You are given  $n$  coins, all of which are equal in weight except for one that is either heavier or lighter. You are also given a two-pan balance to use. In each use of the balance you may put any number of the  $n$  coins on the left pan, and the same number on the right pan, and push a button to initiate the weighing. There are three possible outcomes: either the weights are equal, or the coins on the left are heavier, or the coins on the left are lighter.

Come up with an upper bound on the number of coins  $n$  that can be weighed in  $k$  separate weighings. In other words, if you are allowed to do  $k$  separate weighings, then your bound should say that you can discover, out of a set of no more than  $f(k)$  coins for some  $f(\cdot)$ , which of the coins is odd, and if it is heavier or lighter.

**Problem 5b.** extra credit (but at least read and think about):

For this part of the problem, there are  $n = 12$  coins. Your task is to design a strategy to determine which is the odd coin and whether it is heavier or lighter than the others *in as few uses of the balance as possible* (the fewest number is three by the way).

While thinking about this problem, you should consider (and will probably find helpful) the following questions, some of which were discussed in class.

- (1) How can one measure information?
- (2) What is the most information you can get from a single weighing?
- (3) When you have identified the odd coin and whether it is heavy or light, how much information have you gained?
- (4) What is the smallest number of weighings that might conceivably be sufficient to always identify the odd coin and whether it is heavy or light? Note the similarity of this question to problem 5a.
- (5) As you design a strategy you can draw a tree showing for each of the three outcomes of a weighing what weighing you perform next. What is the probability of each of the possible outcomes of the first weighing, assuming uniform initial placement of the coins and selection of if the odd coin is heavier or lighter?
- (6) How much information is gained on the first step of the weighing problem if 6 coins are weighed against the other 6? How much is gained if 4 are weighed against 4 on the first step, leaving the other 4 aside? Can you apply similar ideas to subsequent weighings?