Lecture 8, Tuesday w5, 2014-10-14

Things we covered:

- Motivation for stream codes: optimal predictions incorporate learning and knowledge of any dependencies in the source.
- Product rule, and its generalization to many variables, sometimes called the chain rule.
- Arithmetic coding:
  - Imagine all possible terminated strings in a binary tree, where nodes correspond to some interval on the number line [0,1).
  - The interval for a particular string can be identified by consulting a series of predictive distributions for each symbol in turn, given the previous. Linear computation in length of string.
  - Binary strings of length l also define intervals on the number-line, with heights 2^{-l}.
  - Encoding: find a binary string whose interval is unambiguously inside the interval for the target string.
  - Decoding: find the unique string that entirely surrounds the given binary interval.

I forgot to mention the material covered at the top of p111 of MacKay, which I normally explain in class (Sorry!). It's a way of turning the guessing game we played into a compression system, if the guesses are made by a reproducible machine instead of an audience. Non-examinable, but a way of understanding how sequential guesses could be useful, which you may find interesting.

The slides also reference some of the history of such guessing games. Shannon used such games to estimate the entropy of the English language (possibly as low as 1 bit / character!).

Check your progress

Things to try to work out after the lecture:

- Imagine encoding a pair of symbols with dependency described by some joint distribution, $P(x_1, x_2)$. If we encoded the symbols independently, implicitly assuming $P(x_1, x_2) = P(x_1)P(x_2)$, what would be the penalty? (We covered something last time on how to answer this question.)
- Does arithmetic coding define an instantaneous code?

- Does arithmetic coding define a complete code?
- In lectures, I skimmed over the details of how an interval was defined, and precisely what was required for a binary string to encode an interval defined by the model. The book (see reading, below) is more careful, representing an interval from the model like this: $[a, b)$. Why is that necessary, and how precisely are the intervals matched together?

Recommended reading

We've nearly finished the 'week 4' slides. I'll probably go over bits I missed as part of a quick recap next time. Otherwise they may be self-explanatory(?).

Read **MacKay pp110–116**.