Information Theory and Networks

Lecture 10: Sampling with Fair Coins

Matthew Roughan
<matthew.roughan@adelaide.edu.au>
http://www.maths.adelaide.edu.au/matthew.roughan/Lecture_notes/InformationTheory/

School of Mathematical Sciences,
University of Adelaide

September 18, 2013

Part I

Sampling with Fair Coins

USA Today has come out with a new survey: Apparently three out of four people make up 75 percent of the population.

David Letterman

Problem

Imagine you have a fair coin, but you want to sample from an arbitrary distribution, how would you do it?

Example 1

Example: use a sequence of fair coin tosses to generate a random variable X with PMF

$$X = \begin{cases} a, & \text{with probability } 1/2, \\ b, & \text{with probability } 1/4, \\ c, & \text{with probability } 1/4, \end{cases}$$

Example 1

Obvious solution:

1. Toss the coin once:
   1. If its a H, then $X = a$
   2. If its a T, toss it again
      1. If its a H, then $X = b$
      2. If its a T, then $X = c$

$$X = \begin{cases} a, & \text{with probability } 1/2, \\ b, & \text{with probability } 1/4, \\ c, & \text{with probability } 1/4, \end{cases}$$

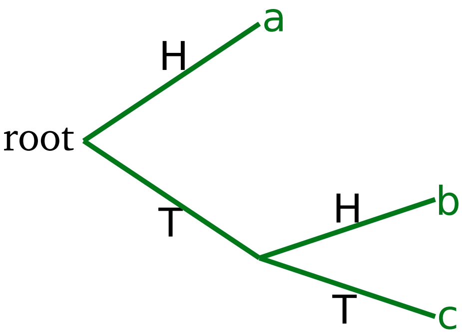
General Problem

- We want to generate a random varible $X \in \Omega = \{1, 2, \cdots, m\}$
  - X has PMF $\{p_1, p_2, \ldots, p_m\}$
- We have a series of (independent) fair coin tosses $Z_1, Z_2, \ldots$
  - let T denote the number of coin tosses (which is potentially a RV)
  - we'd like methods that minmise $E[T]$

Example 1

First thing to note is that the coin tosses define a binary tree:
For example, given the solution for

$$X = \begin{cases} a, & \text{with probability } 1/2, \\ b, & \text{with probability } 1/4, \\ c, & \text{with probability } 1/4, \end{cases}$$



So consider the tree we want to generate:

- The tree should be complete: every node should be a leaf, or have two descendents.
  - i.e., at a node, we stop, or toss another coin
  - probability of a leaf at depth k is 2^{-k}
- The leaves correspond to outcomes for X
  - more than one leaf could be labelled with the same outcome
  - the total probability of all leaves with the same outcome i should be p_i
- The tree could be infinite
- There are multiple possible trees for some problems
  - lets aim for the most efficient
  - minimise the expected depth

Problem

How could we go about designing such a tree for a dyadic distribution (one whose probabilities are powers of two)?

Is it related to entropy?

Solution: use the Huffman tree we get from treating it like a coding problem.
Why?

- Tree generates dyadic probabilities
- Probability of leaf for code k is $2^{-\ell_k}$ where ℓ_k is the length of the code (depth of the tree)
- Earlier we showed that for dyadic probabilities the optimal code lengths were

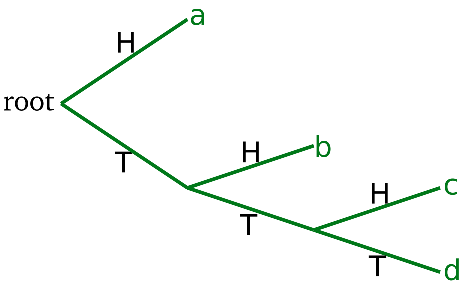
$$\ell_k = \left\lceil \log_D \left(\frac{1}{p_k} \right) \right\rceil$$

  For dyadic probabilities, this gives integer lengths.
- Huffman code tree generates optimal codes
- So the probabilities the Huffman code generates are the same as the ones we need
  - and the expected number of coin tosses will be minimised
  - the expected number will be the entropy

Example 2

$$X = \begin{cases} a, & \text{with probability } 1/2, & \text{has code } 0 \\ b, & \text{with probability } 1/4, & \text{has code } 10 \\ c, & \text{with probability } 1/8, & \text{has code } 110 \\ d, & \text{with probability } 1/8, & \text{has code } 111 \end{cases}$$

Code tree as a set of trials: $0 = H$, $1 = T$



Problem

What about non-dyadic probabilities?

Solution:

- Break probabilities into dyadic atoms
    - Write out the probability in binary (decimal) notation
    - You may need to approximate at some point
- Combine back to the original probabilities by applying the same label to the appropriate leaves.
    - sum leafs with same labels

Example 3

$$X = \begin{cases} a, & \text{with probability } 1/3 \\ b, & \text{with probability } 2/3 \end{cases}$$

Binary expansions:

$$\begin{aligned} \frac{2}{3} &= 0.101010101\ldots &= 2^{-1} + 2^{-3} + \cdots \\ \frac{1}{3} &= 0.010101010\ldots &= 2^{-2} + 2^{-4} + \cdots \end{aligned}$$

So the atoms we need are

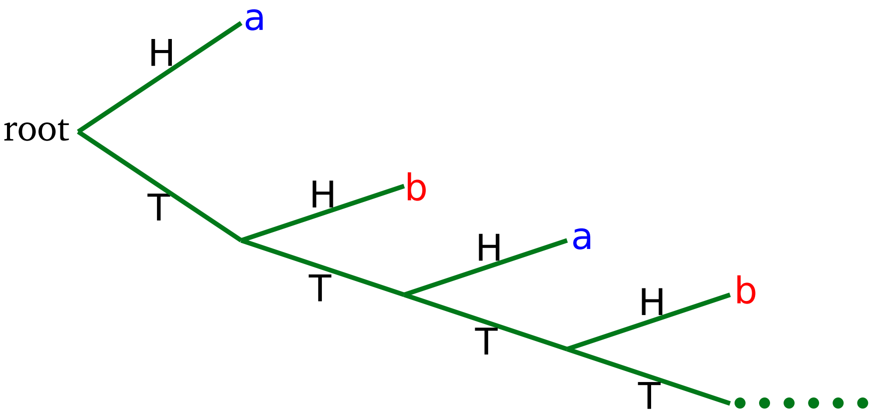
$$(p_1, p_2, \ldots) = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots \right)$$

and colour indicates the label.

Example 3

We need a Huffman code tree for the PMF

$$(p_1, p_2, \ldots) = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots \right)$$



Problem

What if you don't even know if your coin is fair?

Solution: You can always get a fair $p = 1/2$ Bernoulli trial with a biased coin by

1. Toss the coin twice
2. If you get two heads or two tails, repeat until you get HT or TH
3. If you get
   - HT call that a head
   - TH call that a tail

The two events HT and TH have equal probability, by construction.

Source Coding and 20 Questions

Yet another way to think about coding

- 20 questions:
  - Want to guess a 'fact' — say an experiment's outcome
  - Only allowed Yes/No questions
  - Want to find the most efficient set of questions
- Obviously, Huffman code is optimal way of generating questions if we know the PMF

Further reading I

Thomas M. Cover and Joy A. Thomas, *Elements of information theory*, John Wiley and Sons, 1991.