# Notes on Generating Probability Distributions for a given Entropy value

Kornilios Kourtis
kkourt@cslab.ece.ntua.gr

## 1 Motivation

The goal is to develop an algorithm for creating a probability distribution that results in a given entropy value. Ideally, the algorithm should be able to generate all possible probability distributions, with the same probability. Nevertheless, since it is not trivial to guarantee this property, we will insert some randomness to our algorithm to ensure some variation in the results.

## 2 Introduction

Assuming a discrete probability distribution $p_i$, it stands that:

$$\sum_{i=1}^{N} p_i = 1 \tag{1}$$

The expression for the entropy $E$ is:

$$E = -\sum_{i=1}^{N} p_i \log p_i \tag{2}$$

[1]

We will use 2 for the base of the logarithm, which corresponds to the choice of `bit` as the unit of measurement for the value of entropy.

## 3 A pair for a pair

A simple approach for the generation of a probability distribution with a specified entropy value is to create an algorighm that, given an the target entropy $E_t$, a probability distribution $pd_\rho$ and its entropy $E_\rho$, calculates a new probability distribution $pd_{\rho+1}$ with entropy $E_{\rho+1}$ such that:

$$|E_t - E_\rho| > |E_t - E_{\rho+1}| \tag{3}$$

[1] $p = 0 \Rightarrow p \log p = 0$

We consider the following approach: out of the $pd_\rho$ distribution we pick two probabilities $p_k$, $p_l$ and replace them with two different probabilities $p'_k$, $p'_l$ to form $pd_{\rho+1}$. Since $pd_{\rho+i}$ needs to satisfy (1):

$$p_k + p_l = p'_k + p'_l = \sigma \Rightarrow \left\{ \begin{array}{lcl} p_k &=& \alpha\,\sigma \\ p_l &=& (1-\alpha)\,\sigma \\ & & \\ p'_k &=& \beta\,\sigma \\ p'_l &=& (1-\beta)\,\sigma \end{array} \right\} \tag{4}$$

From (2) the difference in entropy is:

$$\begin{aligned} \Delta E =& E_{\rho+1} - E_\rho \\ =& -\sum_{i=1}^{N} p'_i \log p'_i + \sum_{i=1}^{N} p_i \log p_i \\ =& -p'_k \log p'_k - p'_l \log p'_l + p_k \log p_k + p_l \log p_l \\ \overset{(4)}{=}& -\beta\sigma \log \beta\sigma - (1-\beta)\sigma \log (1-\beta)\sigma + \alpha\sigma \log \alpha\sigma + (1-\alpha)\sigma \log (1-\alpha)\sigma \end{aligned}$$

Which after some simple calculations results in:

$$\Delta E = \sigma\Big(\alpha \log \alpha + (1-\alpha)\log(1-\alpha) - \beta \log \beta - (1-\beta)\log(1-\beta)\Big) \tag{5}$$

Hence an equivalent declaration of the problem is to find an appropriate $\beta$ so (3) is satisfied, given $\sigma$, $\alpha$. We consider and study the function $h$ for $\alpha, \beta \in (0,1)$:

$$\begin{aligned} h_\alpha(\beta) &= \alpha \log \alpha + (1-\alpha)\log(1-\alpha) - \beta \log \beta - (1-\beta)\log(1-\beta) \\ &= g(\alpha) - g(\beta), \quad g(x) = x \log x + (1-x)\log(1-x) \end{aligned}$$

**Remark 3.1** *$\beta = \alpha$ and $\beta = 1-\alpha$ are roots of $h_\alpha(\beta)$*

**Lemma 3.2** *$h_\alpha(\beta)$ is strictly increasing for $\beta \in (0, \frac{1}{2})$, strictly decreasing for $\beta \in (\frac{1}{2}, 1)$, has a maximum for $\beta = \frac{1}{2}$ and .*

We use the first derivative of the function:

$$\begin{aligned} h'_\alpha(\beta) &= 0 + \Big(-\beta \log \beta - (1-\beta)\log(1-\beta)\Big)' \\ &= -\frac{1}{\ln 2}\Big(\beta' \ln \beta + \beta \ln' \beta + (1-\beta)' \ln(1-\beta) + (1-\beta)\ln'(1-\beta)\Big) \\ &= -\frac{1}{\ln 2}\Big(\ln \beta + \frac{\beta}{\beta} - \ln(1-\beta) + \frac{1-\beta}{-(1-\beta)} \\ &= -\frac{1}{\ln 2}\Big(\ln \beta + 1 - \ln(1-\beta) - 1\Big) \end{aligned}$$

Thus:

$$h'_\alpha(\beta) = \log\frac{1-\beta}{\beta} \Rightarrow \begin{cases} 0 < \beta < \frac{1}{2} & \Rightarrow & h'_\alpha(\beta) > 0 \\ \beta = \frac{1}{2} & \Rightarrow & h'_\alpha(\beta) = 0 \\ \frac{1}{2} < \beta < 1 & \Rightarrow & h'_\alpha(\beta) < 0 \end{cases} \tag{6}$$

It should be noted that we can focus on area $(0, \frac{1}{2}]$ for $\alpha, \beta$, since we can choose $p_k < p_l$ and $p'_k < p'_l$ without loss of generality. This is reflected in the $h$ function by the symmetry resulting from: $h_a(b) = h_{1-a}(b) = h_a(1-b) = h_{1-a}(1-b)$. Figure 1 presents plots for a number of $h$ functions with different $\alpha$ values. Note that for $\alpha = 0.5$ the entropy value can't be increased since the two symbols have the same probability and we can't insert more randomness.

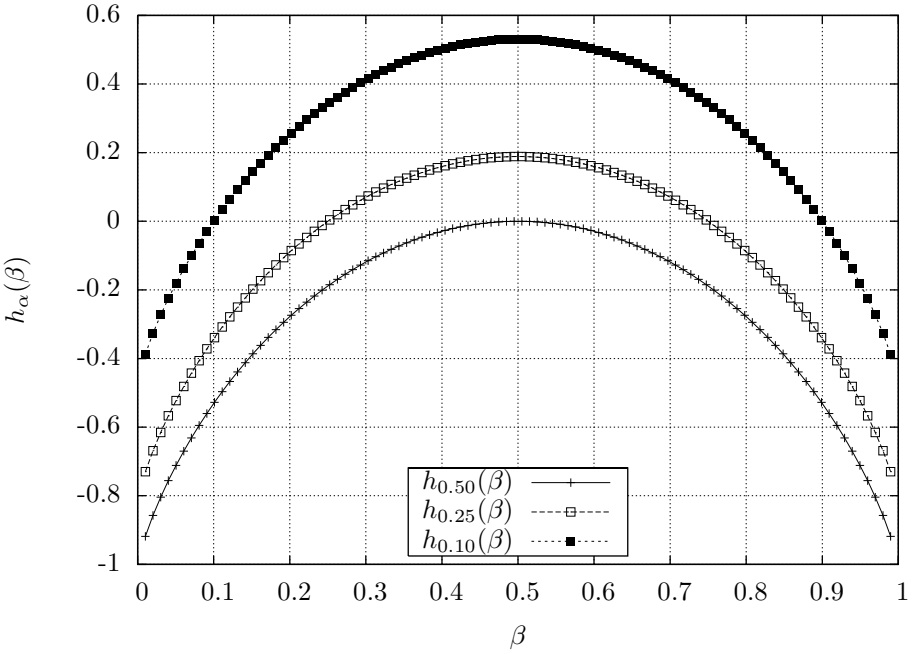


Figure 1: Plots of the $h_\alpha$ function for different values of $\alpha$

The range of the entropy difference that results from a pair substition is

$$\lim_{\beta\to 0} h_a(\beta) < \frac{\Delta E}{\sigma} < h_a(0.5) \Rightarrow g(\alpha) < \frac{\Delta E}{\sigma} < g(\alpha) + 1 \tag{7}$$

In theory, at each step we can choose a new entropy value $E_{\rho+1}$ that satisfies both (3) (convergence) and (7) (existence) and calculate the corresponding $\beta$ value. To compute the $\beta$ value we need to solve $h_a(\beta) = \Delta E/\sigma$. A simple approach is to use binary search, since $h_a$ is strictly increasing.

In practice, however, we use a more efficient approach and choose a $\beta$ value from the range $(0, \alpha)$ if $E_\rho > E_t$ and from the range $(\alpha, 0.5)$ if $E_\rho < E_t$. The choice of the $\beta$ value is random (uniformly), in order to add some randomness to the results.

An important aspect of this method is the choice of the initial pair to replace. There are two contradictory properties of the selection process: convergence and variation. The process should be made fast enough to be of practical value and slow enough to allow for variation inserted in the results. Intuitively, probabilities close arithmetically are good candidates for decreasing the entropy, while the opposite stands for probabilities with large difference.

The (current) implementation is done (mostly) in python and can be found in `http://www.cslab.ece.ntua.gr/~kkourt/src/entropy/`. It uses the Beta distribution to select the initial pair. The parameters of the distribution $(\alpha, \beta)$ are chosen randomly for a number of iterations.