POLITECNICO DI TORINO

Master's Degree in Physics of Complex systems

Master's Degree Thesis

A coding approach to Statistical Inference

Supervisors

Prof. Matteo MARSILI

Prof. Luca DALL'ASTA

Candidate

Emanuele MASSA

September 2020

Acknowledgements

First I would like to express my gratitude to my supervisor Prof.M.Marsili for the technical support and his patience in following the developments of the project.

I am extremely grateful to my family, for their always present support and to my friends for their unconditioned and always sincere presence in my life.

Finally I would like to thank all my colleagues that have become friends during these last years.

Summary

A collection of data points, or a *sample*, is a sequence of N identical and independently distributed realizations $\hat{s} = (s_1, \ldots, s_N)$ of the random variable S. The latter is defined over a finite alphabet $\mathcal{S}$ and is distributed according to $p(s)$, which we are going to call the *generative model*.

Throughout the text upper case letters refer to random variables while low case ones are used to label the values a random variable can assume.

A representation is a probabilistic mapping from the space of outcomes $s \in \mathcal{S}$, to the space of the *labels* $\underline{b} \in \mathcal{B}^*$,

$$p(s) = \sum_{\underline{b} \in \mathcal{B}^*} p(s|\underline{b}) p(\underline{b}) \tag{1}$$

Our goal is to devise a mapping $p(s|\underline{b})$ and infer a probability distribution over the labels $p(\underline{b})$ such that eq.1 is satisfied [1].

Since the true generative distribution is not known,the latter is approximated by the empirical distribution computed on the basis of a sample of N identical and independently distributed draws

$$p(s) \approx \hat{p}(s) = \frac{\hat{k}_s}{N} \tag{2}$$

$$k_s = \sum_{i=1}^{N} \delta_{s^i,s} \tag{3}$$

We refer to k_s as the frequency of outcome s, since it counts the number of times s is observed in a sample of size N. At this point we must also distinguish between the set of different symbols $\mathcal{S}$ which is in general not known, and the set of different symbols that are observed $\mathcal{S}'$.The latter depend on the sample and is thus a random variable itself.

In our study the *labels* $\underline{b}$ are inferred *after* observing the data, by considering only the information contained in the sample $\hat{s}$. This latter setting is known as *unsupervised* in the literature [2]. In this context the labels are then equivalent

to *hidden variables*, since their values are computed by those of the *observable* variables (i.e $s \in \mathcal{S}$).

We propose to generate *representations* by employing a *coding* procedure.

Coding procedures construct *codes* for a set of symbols $s \in \mathcal{S}$ on the basis of a probability distribution $p(s)$.

A code is a dictionary: for each symbol $s \in \mathcal{S}$ there is an associated code-word. The latter is a sequence of characters $\underline{b} = (b_1, b_2, \dots)$, each extracted from an alphabet $\mathcal{B}$. In order for the code to be *decodable*, symbol and code-word must be associated in such a way that knowing the code-word we can retrieve easily the corresponding symbol $b \to s$.

By the definition given above then, every code is also a representation.This is said to be *deterministic* because to each symbol encoded corresponds one and only one code-word.

Among the many *coding* procedures available, Huffman coding is in particular an *optimal* and simple one [3],[4].

We then investigated the possibility of employing *Binary* Huffman coding, i.e $\mathcal{B} = \{0,1\}$, as an unsupervised manner of building representations.

The aim of the following work is to study:

- whether the number of Huffman codes can be taken as a *measure* of the information carried by the sample on its generative process.
- the properties of the representation extracted, following previous works [5][2]. This is defined over a set of binary strings that constitute the code-words of the Huffman codes $b_1, b_2, \dots$ and are seen *hidden* variables in this context.

The thesis is divided into 4 parts:

chapter 1 We present key concepts in coding theory, properties of codes and the optimal code-length theorem [3]. Huffman codes[4] are then introduced and some of their useful properties are discussed, together with their use as generative algorithm for discrete distributions.

chapter 2 After an introduction to the concepts and meaning of Relevance and Resolution [6], we consider the definition of Most Informative Samples as stated in the articles by Marsili et al[2]. We then argue why the number of Huffman codes can be taken as a measure of the uncertainty regarding the generative model of the data. The relationship between the latter quantities and $\mathcal{R}[\hat{k}]$, the logarithm of the number of Huffman trees is investigated and the numerical results are presented and discussed.

chapter 3 A probabilistic representation $\pi(s|\underline{b})$ based on a *mixture* of different Huffman codes is explicitly constructed and the maximum entropy distribution $p(\underline{b})$

satisfying eq.1 is derived.We then focused on individuating a subset of *relevant* hidden variables.In order to do so we propose an iterative procedure to integrate out bits and compute the mutual information $I[S, \underline{B}]$ between the latter $\underline{B}$ and the source S. This allows us to define *robust* bits as those containing the largest fraction of mutual information as opposed to *noisy* bits, i.e the remaining ones.

chapter 3 As a benchmark example the ideas developed in the previous part are employed on the data-set regarding the decisions of the Supreme Court of the United States [7].

Chapter 1

Codes,Huffman coding and generative models

1.1 A short introduction to coding theory and data compression

For data compression we consider the problem of *encoding* the values assumed by a random variable S that we will call the *source*. The latter S takes values $s \in \mathcal{S}$ with probability $P(s) \in [0,1] \subset \mathbb{R}$.

A *code* is a function that associates to each symbol $s \in \mathcal{S}$ a code-word $\underline{b} \in \mathcal{B}^*$

$$\mathcal{C} : s \in \mathcal{S} \to \underline{b} \in \mathcal{B}^*$$

where $\mathcal{B}$ is the alphabet, the set of *characters* available, and $\mathcal{B}^* = \{(b_1, \ldots, b_n), b_i \in \mathcal{B}, n \in \mathbb{N}\}$ is the set of all possible sequences made of characters.

Our aim is to compress a given sequence of symbols $\hat{s} = (s_1, s_2, \ldots, s_N)$, also known as a *message*, as much as possible.

Since each message $\hat{s}$ is observed with a certain probability, the length of the encoded message is a random variable.

We define the *code-length* function:

$$L(s) : \mathcal{S} \to \mathbb{N}$$

that is induced by the code over the symbols.

The length of the encoded message $L(\hat{s})$ is the sum of the lengths of the encoded symbols that are observed in $\hat{s} = (s_1, s_2, \ldots, s_N)$

$$L(\hat{s}) = \sum_{s \in \mathcal{S}} L(s) k_s$$

This quantity, that we are going to call *message length*, grows at least linearly in N (the number of draws in the sample), so diverges as $N \to +\infty$.

Instead the message length per data-point

$$\frac{L(\hat{s})}{N} = \sum_{s \in \mathcal{S}} L(s) \frac{k_s}{N}$$

is always a finite quantity and in the limit of long $N \to +\infty$ messages converges to the average message length

$$\mathbb{E}\Big[L\Big] = \sum_{s} p(s) L(s)$$

We then look for the code that achieve the minimum of the latter quantity:

$$L^*(s) = \arg\min_{L(s)} \left\{ \sum_{s} p(s) L(s) \right\}$$

Intuitively an *optimal* code should assign shorter code-words to more probable symbols.

When compressing messages $\hat{s}$, which are sequences of symbols, we produce sequences of code-words.

In order to achieve the optimal code-length function we should then avoid to employ an additional character to signal the separation between different code-words.

This leads to the definition of *Prefix-code* or *instantaneous code* [8].

Definition 1 (Prefix-code) *A code is called prefix-code if no code-word is prefix of any other code-word.*

In other words prefix-codes are such that encoding a message $\hat{s} = (s^1, \ldots, s^N)$ by concatenating code-words

$$\hat{b} = (\underline{b}_1, \underline{b}_2, \ldots, \underline{b}_N) = (\mathcal{C}(s^1), \mathcal{C}(s^2), \ldots, \mathcal{C}(s^N))$$

is possible without any ambiguity since

$$\mathcal{C}^*(\hat{s}) = (C(s^{(1)}), C(s^{(2)}), \ldots, C(s^{(N)})) : \hat{s} \in \mathcal{S}^N \to \underline{b} \in \mathcal{B}^*$$

is a bijective function [8].

The following theorem imposes a constraint on the possible code-length functions of a prefix-code and is a cornerstone of coding theory, the interested reader can find a proof in [3].

Theorem 1 (Kraft Inequality) *For any* prefix-code *over a finite alphabet $\mathcal{B}$, the code-length function $L(s), s \in \mathcal{S}$ must be such that:*

$\sum_s |\mathcal{B}|^{-L(s)} \leq 1$

Conversely for each sequence of integers $l_1, \ldots, l_N$ that satisfies the above inequality exists a prefix-code *with code-length function taking values $L(s) = l_s$.*

Optimal code-length

Finding the optimal code for a given source can then be stated formally as a constrained minimization problem,

$$L^* = \arg\min \left\{ \sum_s p(s)L(s) + \lambda(\sum_s 2^{-L(s)}) \right\}$$

where the constraint is due to the Kraf Inequality 1.

Ignoring the restriction of integer code-length the solution reads

$$L^*(s) = -\log_2(p(s))$$

so that the minimum expected average message length is given by the entropy of the source

$$H[S] = \sum_s -p(s)\log_2(p(s))$$

Note that if we take the ceiling of $\lceil -\log_2(p(s)) \rceil$ in order to have integer code-lengths,this also satisfies the Kraft Inequality:

$$\sum_s 2^{-\lceil -\log_2(p(s)) \rceil} \leq \sum_s 2^{-\log_2(p(s))} \leq 1$$

and $\log_2(\frac{1}{p(s)}) \leq \lceil -\log_2(p(s)) \rceil < 1 + \log_2(\frac{1}{p(s)})$,this implies

$$H[S] \leq \mathbb{E}[L(s)] < H[S] + 1$$

so the average message length of an optimal code will exceed the entropy of the source by at most 1 bit.

1.2 Huffman codes

So far we have not mentioned how to retrieve an actual optimal code. Many procedures exists generating codes that satisfy the bound on the average message length [3], examples are Arithmetic coding by Rissanen [9] and Lempel-Ziv-Welch coding by the homonym authors[10],[11].

An elegant algorithmic procedure constructing *optimal* prefix-codes was invented by David Huffman [4], while still a student at MIT in the 50′ and is known as *minimum redundancy* coding or *Huffman* coding.

1.2.1 Constructing minimum redundancy codes

Given a source S emitting symbols $s \in \mathcal{S}$,$|\mathcal{S}| = Q$, with probability $p(s)$ the coding algorithm [4] for a *binary* alphabet $\mathcal{B} = \{0,1\}$ goes on as follows:

1 sort $p(s)$ in increasing order $p(s_i) > p(s_{i-1})$, store this in a list $q_k = p(s_k)$

2 take the first 2 elements q_1, q_2 (which will always be the least probable 2) assign bit 0 to the first and 1 to the second.This identifies q_1 and q_2 with the two nodes of a binary tree.

3 define $q_1 \leftarrow q_1 + q_2$ and $q_2 \leftarrow q_3 \dots q_{Q-1} \leftarrow q_Q$, delete q_Q.

4 repeat until the length of the list is 1, or equivalently, after $Q - 1$ iterations.

In other words at each step we are defining a new sample where two outcomes are merged together defining a node with frequency the sum of the frequencies.Thus at each iteration the list becomes shorter of one unit.

If the above list is stored in a heap structure, which automatically grants an ordered sequence at each of the above steps, then the algorithm runs in $\Theta(Q \log Q)$.

Consider employing the empirical distribution $\hat{p}(s)$ computed on a sample of N observations $\hat{s} = (s_1, s_2, \dots, s_N)$. This is defined over the states that have been observed at least once $\mathcal{S}'$,$M = |\mathcal{S}'| \leq N$. So in this case the algorithm runs in $\Theta(N \log N)$.

Huffman codes and binary trees

Each Huffman code uniquely identifies a *complete binary tree*.The latter is iteratively generated during the creation of the Huffman code.

A *binary tree* is a tree where each internal node has at most 2 leaves.The latter is said to be *complete* if every internal node has *exactly* two leaves.

A code-word $\underline{b} = (b_1, b_2, \dots)$ specifies a path linking the root of a tree to a leaf.

This is useful in the decoding procedure since we just need to control the bits sequentially from the left-most: when a leaf of the Huffman tree is reached we write the correspondent character,then we restart from the root and repeat the same procedure until the last bit is reached.

An example presenting a possible binary tree generated by the empirical distribution of the letters in *Politecnico di Torino* is presented in fig 1.1.

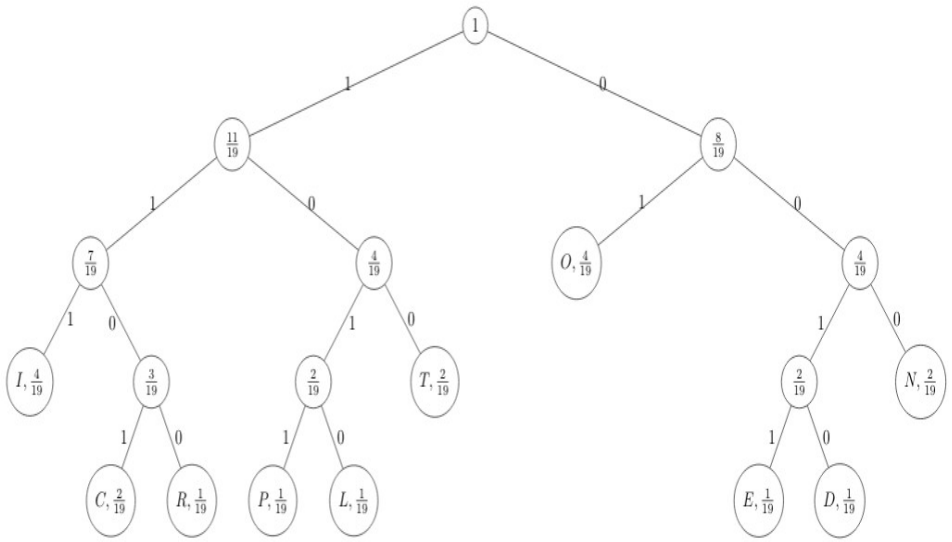


Figure 1.1: One of the possible Huffman tree/code for the empirical distribution of the letters in the string *Politecnico di Torino*

1.3 Generating Discrete Distributions by Huffman codes

The Huffman coding routine produces a complete binary tree that can be regarded as an *algorithm* to generate symbols $s \in \mathcal{S}$ distributed according to a discrete probability distribution $p(s)$ [3].

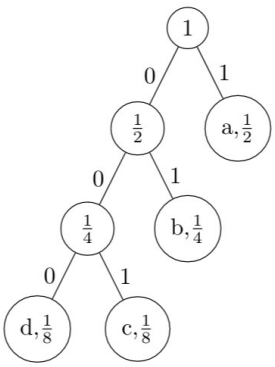
We produce binary sequences $\underline{b} = (b_1, \ldots, b_{d_T})$ (d_T being the depth of the tree) with a certain probability $p(\underline{b})$, that is dictated by $p(s)$ and by the Huffman tree, and then we generate a symbol by decoding the binary sequence.

When the distribution is *dyadic*, i.e the probability of each outcome is a negative power of 2, the probability assigned to each leaf on the Huffman tree will be the *actual* probability of the correspondent symbol.

It is then sufficient to generate random bit-strings of the appropriate length and to follow the correspondent path starting from the root of the tree until a leaf is found,the latter corresponds to a symbol in $\mathcal{S}$.

Example 1 (Huffman tree as generative algorithm) *Consider the trivial case of a dyadic distribution where $M = 4$ and $p(a) > p(b) > p(c) > p(d)$ then a tree that can be generated by Huffman procedure is in fig1.2.*

The depth of the corresponding tree, equal to the number of bits, is 3*. This means that we will have $2^3 = 8$ bit-strings and the generative model can be visualized as a table 1.2.*



b_1	b_2	b_3	s
0	0	0	d
0	0	1	c
0	1	0	b
0	1	1	b
1	0	0	a
1	0	1	a
1	1	0	a
1	1	1	a

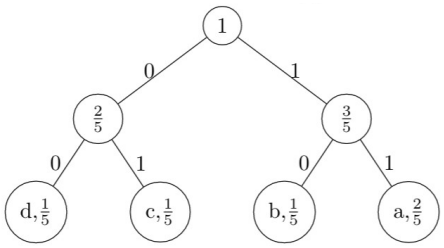
Figure 1.2: One of the Huffman trees generated by the dyadic distribution $p(a) = \frac{1}{2}$,$p(b) = \frac{1}{4}$ and $p(c) = p(d) = \frac{1}{8}$ and its corresponding table

In the case where the probability distribution over $\mathcal{S}$ is not dyadic,see fig 1.3, Huffman coding will generate a binary tree that is in one to one correspondence with a dyadic distribution over $\mathcal{S}$. In order to generate $s \sim p(s)$ a non-uniform probability distribution $p(\underline{b})$ over the binary sequences $b_1, \ldots, b_{d_T}$ must then be considered.

We introduce a way of retrieving $p(\underline{b})$ in the 4th chapter.This leads us to define a representation matrix whcih is obtained as an average over *all* the Huffman trees.

1.3.1 Degeneracy of Huffman codes

At each step of the algorithm to construct an Huffman tree there could happen ties in the frequencies,i.e more then one node having the same frequency. In this



b_1	b_2	s
0	0	d
0	1	c
1	0	b
1	1	a

Figure 1.3: One of the Huffman trees generated by the non-dyadic distribution $p(a) = \frac{2}{5}$,$p(b) = p(c) = p(d) = \frac{1}{5}$ and its corresponding table

case the choice of which couple of nodes to merge is *arbitrary*,so Huffman coding does not provide a *unique* optimal code.

To count the number of different Huffman trees we need to consider all the different choices that can be made in carrying out the construction of a tree.

It is important to note that when merging 2 nodes with *distinct* frequencies the choice is *not* arbitrary, and the bit 0 must be assigned to one with lower frequency. The latter corresponds to an arbitrary rule that must be established before counting the degeneracy and must be observed during all the procedure. In this aspect our analysis departs clearly from that in [4] where this additional symmetry is kept.

When applied on the empirical distribution of a sample $\hat{p}(s)$, we define the histogram of the frequencies

$$m_k = \sum_{s \in \mathcal{S}} \delta_{k,k_s}$$

for a given sample size $\mathcal{S}$ is the set of distinct states that have been observed.

Note that the number of different Huffman trees $NHT(\hat{s})$ is bounded by:

$$NHT(\hat{s}) \leq e^{\mathcal{R}_{max}}$$

where $\mathcal{R}_{max}$ is computed on the sample where $m_1 = N$,i.e exactly N different symbols are observed in a sample of size N.

We can compute this number by noting that at the first step we can choose among N possible states of frequency 1,then we can choose among $N-1$ states one to be merged with the previously chosen, and so on, until there will be $\lfloor N/2 \rfloor$ with frequency 2 and at most 1 state with frequency one. This reasoning can be repeated until we reach the point where only 1 state is left with frequency N.

If we approximate $N \approx 2^d$, then

$$\log(NHT(\hat{s})) \leq \log\left(\prod_{k=0}^{d}(2^k)!\right) = \sum_{k=0}^{d}\log\left((2^k)!\right) = \mathcal{R}_{max} \tag{1.1}$$

$$\mathcal{R}_{max} \sim \sum_{k=0}^{d} k2^k \sim N\log N \tag{1.2}$$

On the other hand consider carrying out the procedure explained in the previous paragraph,all the symbols that bear the same frequency can be permuted leading to the same initial condition, thus resulting in an equally optimal code. This means that $NHT \geq \prod_k m_k!$.

It is then useful to consider

$$\mathcal{R}[k] = \log(NHT)$$

The procedure outlined in the previous paragraph can be employed to compute $\mathcal{R}[k]$, with some simple modifications.

First compute $\underline{m} = \{m_k, k = 1,2,\ldots,N\}$, then start to create the binary tree. At each step there are 2 possibilities:

1 $m_k \geq 2$ then we can pick the 2 elements to merge in this subset, and we have $m_k(m_k-1)$ ways of doing this.

2 $m_k = 1$, thus we must pick the other element in the subset $m_{k'}$ and there are exactly $m_{k'}$ possible choices.

The procedure is delineated as a pseudocode in 1 Since at each step we are merging 2 symbols this is equivalent to reduce $|S|$ of one,thus the running time of the procedure is $\Theta(N)$.

1.3.2 Are all Huffman trees created equal?

Definition 2 *$\mathcal{HT}$ is the set of trees T that can be produced by Huffman coding procedure when applied on $\hat{p}(s)$*

We can ask if the depth of an Huffman tree $T \in \mathcal{HT}$ is the same for every element of this set or not, the importance of this question will become clear in the last chapter.

Algorithm 1 Compute $\mathcal{R}[k]$

procedure $\mathcal{R}[k](m_k)$
    $m_k = \sum_{s\in\mathcal{S}} \delta_{k_s,k} \quad k \in \{1,\dots,N\}, \quad N =$ of draws
    $k = 0$
    $\mathcal{R}[\hat{k}]$=0
    while $k \leq N$ **do**
        if $m_k \geq 2$ **then**
            $\mathcal{R}[\hat{k}] += \log(m_k) + \log(m_k - 1)$
            $m_k \leftarrow m_k - 2$
            $m_{2k} \leftarrow m_{2k} + 1$
        else if $m_k = 1$ **then**
            $\mathcal{R}[\hat{k}] += \log(m_k) + \log(m_{k'})$ ▷ k' is the smallest frequency $> k$ such that $m'_k \geq 1$
            $m_k \leftarrow m_k - 1$
            $m'_k \leftarrow m'_k - 1$
            $m_{k+k'} \leftarrow m_{k+k'} + 1$
        else
            $k = k'$

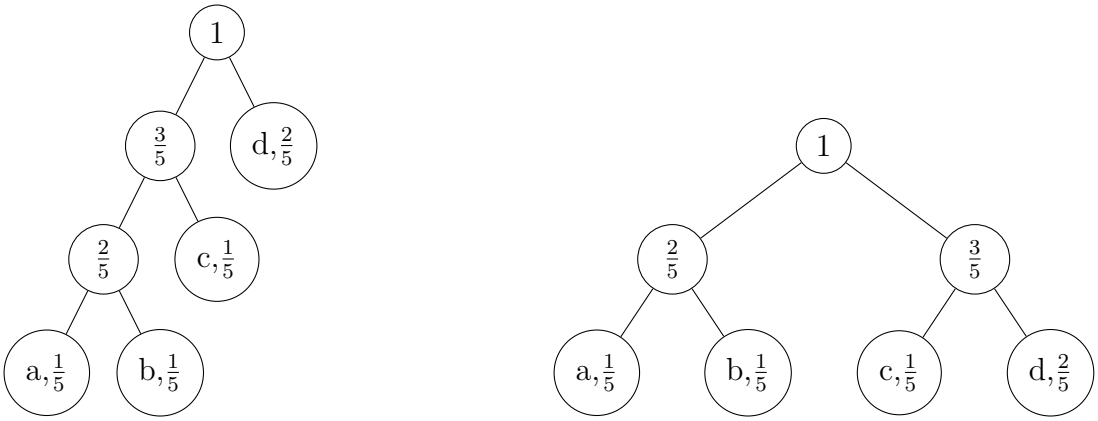


Figure 1.4: In this example, for instance, there are 4 symbols $\mathcal{S} = \{a, b, c, d\}$ and $p_a = p_b = p_c$ while $p_d = 2p_a$. $\mathcal{R} = 12$: 6 trees have depth $d_T = 3$ and the remaining have depth $d_T = 2$.

To answer this question, let's concentrate on the algorithm described above to count the degeneracy. At each step a slot of the array m_k can contain binary trees with different depths.

Depending on how we choose among different nodes when there are ties,we will obtain different trees of different depths 1.4.

In Figure 1.4 two different trees are presented, all the remaining trees can be obtained by symmetry re-shuffling the labels of the symbols with same frequency.

Thus we conclude that d_T is not the same over all the elements of $\mathcal{HT}$.

1.3.3 Storing the Huffman code

Note that we can always add a level to a given binary tree by substituting a leaf with a binary tree of depth 1 with 2 leaves, each with frequency half the one of the original leaf.

This *extended* Huffman code can be constructed by padding shorter code-words with dummy bits until the target depth is reached.

This is useful since we can picture the output of the coding procedure as a mapping

$$p(s|\underline{b}, T) = \delta_{s,s_T(\underline{b})}$$

where we stressed that several Huffman trees exists and each one bears its own code, which corresponds to a probability distribution over the data.

In order to actually store such a matrix in a computer we must know the cardinality of the code-words space. The latter is fixed by the depth of tree considered, and is thus a random variable.

For a given dataset $\hat{s}$ we can still investigate what is the largest depth achievable by a Huffman tree. The deepest tree will be the one that assigns the *longest code-word* to the *least probable symbol*.

Since all Huffman trees are *complete* binary trees, i.e each internal node (all the nodes apart the leaves which are defined as external nodes) posses 2 children,the number of internal nodes is constrained by the number of external nodes [12]:

if there are N external nodes there are exactly $N-1$ internal nodes.

This can be proven by induction [12] and implies that

$$d = \max_{T \in \mathcal{HT}} \left\{ d(T) \right\} \leq N - 1$$

with equality only when all the probabilities are such that $p_i \geq p_{i-1} + p_{i-2}$. This resembles a Fibonacci series and indeed bounds have been derived by exploiting this property[13].

Thus *without* having previously observed the frequency of the outcomes, we know that the number of bits constituting the code-words is $d \leq N - 1$.

This seems to be a useful bound but is in practice a *loose* one, just think that in order to encode 100 symbols we would need at most 99 bits which give $2^{99} \sim 10^{30}$ states which correspond to the number of columns of the $p(s|\underline{b}, T)$ matrix.

Computing the max depth

To compute the maximum depth among all the Huffman trees

$$d = \max_{T : T \in \mathcal{HT}} \{ d(T) \}$$

we employ Huffman procedure and bias the choice of the nodes while constructing the tree: in case of ties we always choose the node which corresponds to the deepest sub-tree.

This procedure is presented as a pseudo-code 2.

This procedure is polynomial in N since we must go through N merging steps and for each of these we search for the max in a finite sequence of size smaller than N.

An upper bound on the running time is given by the worst case scenario in which all the symbols have the same frequency, in this case the running time is $\Theta(N^2)$:

$$T_N = N + (N-1) + (N-1) + (N-2) + \cdots \sim \Theta(N^2)$$

Algorithm 2 Compute max depth of Huffman trees a given sample

procedure MAX-DEPTH(m_k)
    create a list $l_k = \{s : k_s = k\}$
    assign depth $d_s = 0$ to all nodes
    k = 0
    while $k \leq N$ **do**
        if $length(l_k) \geq 2$ **then**
            select $\arg\max\{d(s) : s \in l_k\}$
            select $\arg\max\{d(s') : s' \in l_k, s' \neq s\}$
            delete s, s' from l_k
            add a node s'' to l_{2k}, with $d_{s''} = \max(d_{s'}, ds) + 1$
        else if $length(m_k) = 1$ **then**
            select the only $s \in l_k$
            select $\arg\max\{d(s') : s' \in l_{k'}\}$ ▷ k' is the smallest frequency such that $length(l'_k) > 0$
            delete s from l_k and s' from l'_k
            add a node s'' to $l_{k+k'}$, with $d_{s''} = \max(d_{s'}, ds) + 1$
        else
            $k = k'$

Chapter 2

Information content of a sample and degeneracy of Huffman trees

2.1 Resolution, Relevance and Most Informative Samples

2.1.1 Information contained in a sample:Resolution and Relevance

We go trough a brief review of the key concepts present in [6],[2],[14] by Marsili et al. In these articles the quantities *Resolution* and *Relevance* have been introduced and their meaning was investigated under different point of view.

Consider a complex system which optimazes a utility function $U(\vec{s})$, $\vec{s} = (s, \bar{s})$, defined over a set of known $s \in \mathcal{S}$ and unknown $\bar{s} \in \mathcal{Q}$ variables.

The observed variables $s \in \mathcal{S}$ are defined/chosen by the modeler,in the sense that they must be regarded as abstract labels that identify a cluster of observations of the system in object.The latter can be chosen in different way and account for a description of the system at given level of detail.

The modeler also attempts to approximate the true utility function $U(\vec{s})$ with another one $u(s)$ defined solely on the known variables.

We can then rewrite

$$U(\vec{s}) = u(s) + v(s|\bar{s}), \qquad v(s|\bar{s}) = U(\vec{s}) - u(s) \tag{2.1}$$

The state assumed by the system will be

$$\vec{s}^* = \arg\max_{\vec{s}}\{U(\vec{s})\} = (s^*, \bar{s}^*) \tag{2.2}$$

and the authors showed that the probability that a known state s is the observable part of the optimal state s^* is of the form

$$p(s) = \frac{e^{\beta u(s)}}{\mathcal{Z}(\beta)} \tag{2.3}$$

where β is a function of the number of unknown variables [6].

For a sample, the frequencies k_s provide a noisy estimate of $p(s) \approx \frac{k_s}{N} = \hat{p}(s)$.

In this context the quantity

$$\hat{H}[s] = -\sum_s \hat{p}(s) \log_2 \hat{p}(s)$$

which is the *minimum* number of bits necessary to encode a symbol generated by a random variable $S \sim \hat{p}(s)$, is a measure of the *resolution* of the description given by the labels $s \in \mathcal{S}$, hence the name *Resolution*.

To gain some intuition we can think of two limiting cases:

- at a sufficient level of detail every data point corresponds to a different label, $H[s]$ is maximal and takes value $\log N$
- when the description is too coarse the same symbol s' is observed N times, $k_s = N\delta_{s,s'}$ in a sample of size N, this implies $H[s] = 0$

Introducing the *degeneracy* histogram

$$\underline{m} = (m_1, m_2, \ldots, m_N) \qquad m_k = \sum_{s \in \mathcal{S}} \delta_{k_s,k} \tag{2.4}$$

the quantity

$$\hat{H}[k] = -\sum_k \frac{k m_k}{N} \log_2(\frac{k m_k}{N}) \tag{2.5}$$

is the *minimum* number of bits that are needed to encode an outcome of the random variable k_{s_i} [6], where s_i is a randomly chosen point from the sample $\hat{s}$. We can also use the logarithm in base e to define the above quantities, in this case the units of information will be bits nats as opposed to the former bits. So we will drop the index being aware that both convention are equivalent apart for a scaling factor needed to change the log basis.

Since the frequencies $\underline{k} = \{k_s, s \in \mathcal{S}\}$ give also an estimate of the part of the utility function that depends on the known variables

$$u_s = \frac{1}{\beta} \log(p(s)) - \log(\mathcal{Z}(\beta)) \approx c + \frac{1}{\beta} \log(\frac{k_s}{N}) \tag{2.6}$$

the conclusion of the authors is that the latter quantity, called *Relevance*, quantifies (in bit) the amount of information carried by the sample on the structure of the generative model $u(s)$ [2],[6].

Note that

$$\hat{H}[s] = \hat{H}[s|k] + \hat{H}[k] = -\sum_k \frac{km_k}{N} \log(\frac{1}{m_k}) - \sum_k \frac{km_k}{N} \log(\frac{km_k}{N}) \tag{2.7}$$

The number of classifications of the same data-set $\hat{s}$ that yield the same number of classes $\sum_k m_k$, each with the same size k, is [2]

$$\prod_k \frac{km_k!}{(k!)^{m_k}} \sim e^{N\hat{H}[s|k]} \tag{2.8}$$

so $\hat{H}[s|k]$ is a measure of the ambiguity of the description $s \in \mathcal{S}$ adopted.

Thus the information content (per data point) of the sample $\hat{H}[s]$ consists of two parts:

- $\hat{H}[k]$ which is the *useful* number of bits that can be utilized to estimate the generative process
- $\hat{H}[s|k]$ which is the number of remaining bits that account for the *noise* in the sample, see explanation below.

2.1.2 Most Informative Samples

The previous considerations suggest that samples that are *most informative* about their generative process are such that the *Relevance* is maximal at fixed *Resolution*.

Mathematically *Most Informative Samples* (MIS) are the solution of the optimization problem

$$\underline{m}^{MIS} = \arg\max \left\{ \hat{H}[\hat{k}] + \beta \hat{H}[\hat{s}] + \mu \hat{N} \right\} \tag{2.9}$$

$$= \arg\max_{\underline{m}} \left\{ -\sum_k \frac{km_k}{N} \log(\frac{km_k}{N}) - \beta \sum_k \frac{km_k}{N} \log(\frac{k}{N}) + \mu \sum_k km_k \right\} \tag{2.10}$$

the latter is non trivial, given that it is defined over integer variables $m_k \in \{0, N\} \quad k = 1, \ldots, N$.

These must be compared to random/structure-less samples which corresponds to a choice of the labels that do not provide a meaningful description of the system observed.

As explained in detail in appendix A.1 the MIS optimization problem can be mapped into a statistical mechanics system in the *zero temperature* limit. One can then think to the m_k as being drawn from a probability distribution: when the inverse temperature gets large, the latter is localized around the positive integer values that are solution of the optimization problem 2.9.

In this case the averaged version of the MIS equation is considered.

An upper bound can be obtained as argued in [6], this *approximate* solution reads

$$m_k = ck^{-\beta-1}$$

where

- β is fixed by the *Resolution* constraint
- c is fixed by the constraint over $\sum_k km_k$

A similar result can be reached in a different way as described in A.2.1.

The latter is displayed as an upper bound on the MIS curve in the following pictures,see fig 2.1 triangle curve, fig2.9 yellow curve.

In the article [14] the authors corroborated the result previously obtained by presenting a lower-bound on the MIS curve by taking a Poisson distribution for the m_k with average n_k, see fig 2.1 dotted curve. The approximate result still decreases as a power law in k, with an exponent that is function of β.

These results lead to the conclusion that Most Informative Samples, in the under-sampled regime, have a power law distribution with an exponent which depends on the multiplier β .

The latter can then be used to explain the trade-off between *Resolution* and *Relevance*.

As β varies from large and positive to negative values the solution of (2.9) describes a curve in the plane $\hat{H}[\hat{s}], \hat{H}[\hat{k}]$, starting from the point $\hat{H}[\hat{s}] = 1, \hat{H}[\hat{k}] = 0$ up to the point where the *data processing inequality*[3] is met.

The latter states that a function of a random variable cannot carry more information then the original variable itself:

$$\hat{H}[s] = \hat{H}[s|k] + \hat{H}[k] = -\sum_k \frac{km_k}{N} \log(\frac{1}{m_k}) - \sum_k \frac{km_k}{N} \log(\frac{km_k}{N}) \geq \hat{H}[\hat{k}] \quad (2.11)$$

The figure obtained by numerical simulation 2.1 2.2 can be explained by an approximate analysis. Following [6] and [2] we divide the curve in two parts:

- $\beta \geq 1$ loss-less compression part of the curve, here the slope of the curve is negative then a reduction of one bit in $\hat{H}[\hat{s}]$ is balanced with a gain in $\hat{H}[\hat{k}]$ and the ratio $-\frac{\Delta\hat{H}[\hat{k}]}{\Delta\hat{H}[\hat{s}]} \geq 1$

- $\beta < 1$ lossy compression part where $-\frac{\Delta \hat{H}[\hat{k}]}{\Delta \hat{H}[\hat{s}]} < 1$

Our solution based on the result of a MCMC simulation technique are in agreement with these previous results see fig 2.1,2.2.

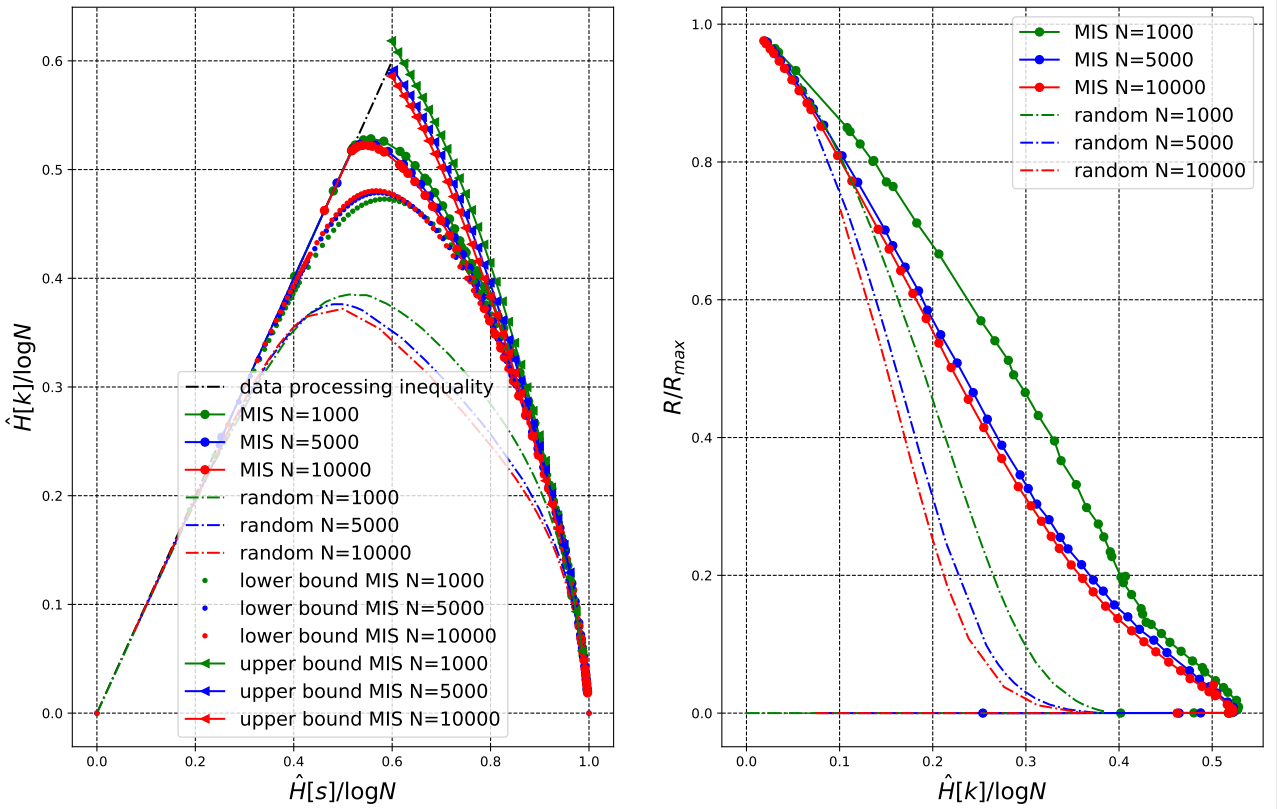


Figure 2.1: The numerical solution of the MIS equation 2.9 are presented for N = 1000,5000,10000, together with the correspondent curve for structure-less samples and the lower bound on the MIS curve obtained by M.Marsili and A.Haimovici in [14] by using Poisson distributed m_k.

The details of the simulation are explained in the appendix A.1.

2.1.3 Representations and Huffman codes

Since each Huffman code corresponds to a deterministic representation relating the observed states $s \in \mathcal{S}'$ to the bit-strings $\underline{b} \in \{0,1\}^d$, $\mathcal{R}[\hat{k}]$ is a measure of how many such representations are compatible with the empirical evidence.

The main intuition is that:

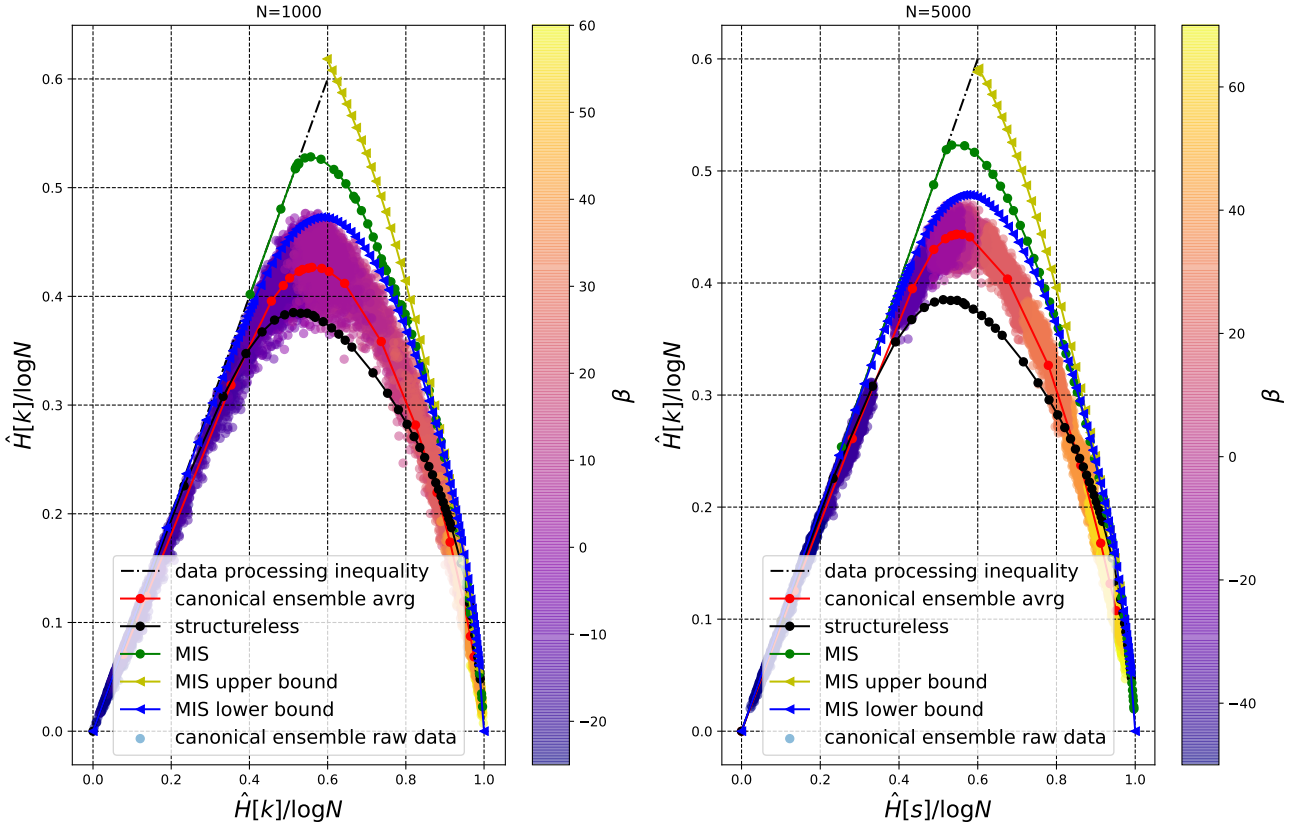


Figure 2.2: Numerical results for N=1000,N=5000. The data marked as *canonical ensemble* are extracted from an approximate measure explained in A.2.1.The *structure-less* curve is obtained as the average over 10^5 realizations of Nrandom tosses in M boxes, with M varying from $10N$ to 2. The upper bound on the MIS curve (on the left) is derived and obtained in A.2.1.The lower bound is the one presented in [14] taking m_k as Poisson R.Vs.

- in the *under-sampled* regime,i.e when the ratio the ratio $\mathcal{S}/N \gg 1$,the number of ties in the frequencies is expected to be huge $m_1 \gg 1, m_2 < m_1, \ldots$, this implies $\mathcal{R}[\hat{k}] \gg 1$
- in the *well-sampled* regime each state is expected to be seen with a different frequency resulting in $\mathcal{R}[\hat{k}] \approx 0$

So that $\mathcal{R}[\hat{k}]$ is expected to vary from close to 0 to $O(N \log N)$, in the case each state is observed once.

The numerical results support our intuitions:

- $\mathcal{R}[\hat{k}]/R_{max}$ is a strictly increasing function of $\hat{H}[\hat{s}]$ in the under-sampling

regime, see fig 2.4

- $\mathcal{R}[\hat{k}]/R_{max}$ is a strictly decreasing function of $\hat{H}[\hat{k}]$ in the under-sampling regime, see fig 2.3,2.2
- $\mathcal{R}[\hat{k}]/R_{max}$ is close to 0 in the well-sampled regime, see figs 2.4 and 2.3

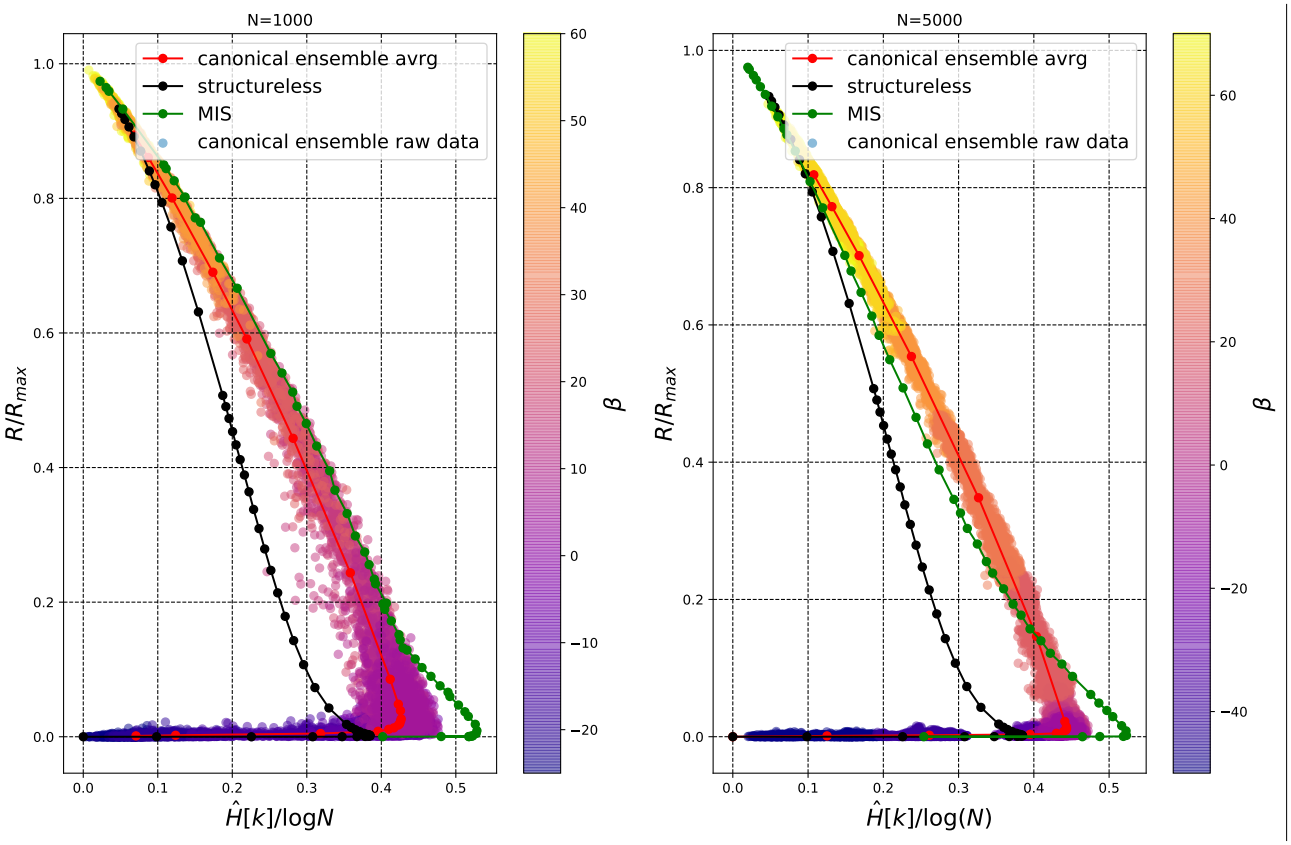


Figure 2.3: Numerical results for N = 1000,5000. Raw data are scatter-plotter together with the MIS curve and the structure-less curve.

One can also argue that the observed dependency in the $\mathcal{R}[k], \hat{H}[k]$ plane can be simply explained by the first observation only. This would not be consistent with fig 2.3. If $\mathcal{R}[k]$ depended solely on $\hat{H}[s]$ then, because of the strictly increasing relation between these 2 quantities, the samples that maximaze $\hat{H}[k]$ at fixed $\hat{H}[s]$ should be coincident with those that maximaze $\hat{H}[k]$ at fixed $\mathcal{R}[k]$.

To further verify weather or not $\mathcal{R}[k]$ depends on both the quantities, we proceed to fit the data $\mathcal{R}[\hat{k}], \hat{H}[\hat{k}], \hat{H}[\hat{s}]$ in the *under-sampled* regime, i.e $\beta > 0$,with a linear relationship. We present the results for the MIS points

$$\mathcal{R}[\hat{k}] = \alpha_s \hat{H}[\hat{s}] + \alpha_k \hat{H}[\hat{k}] \tag{2.12}$$

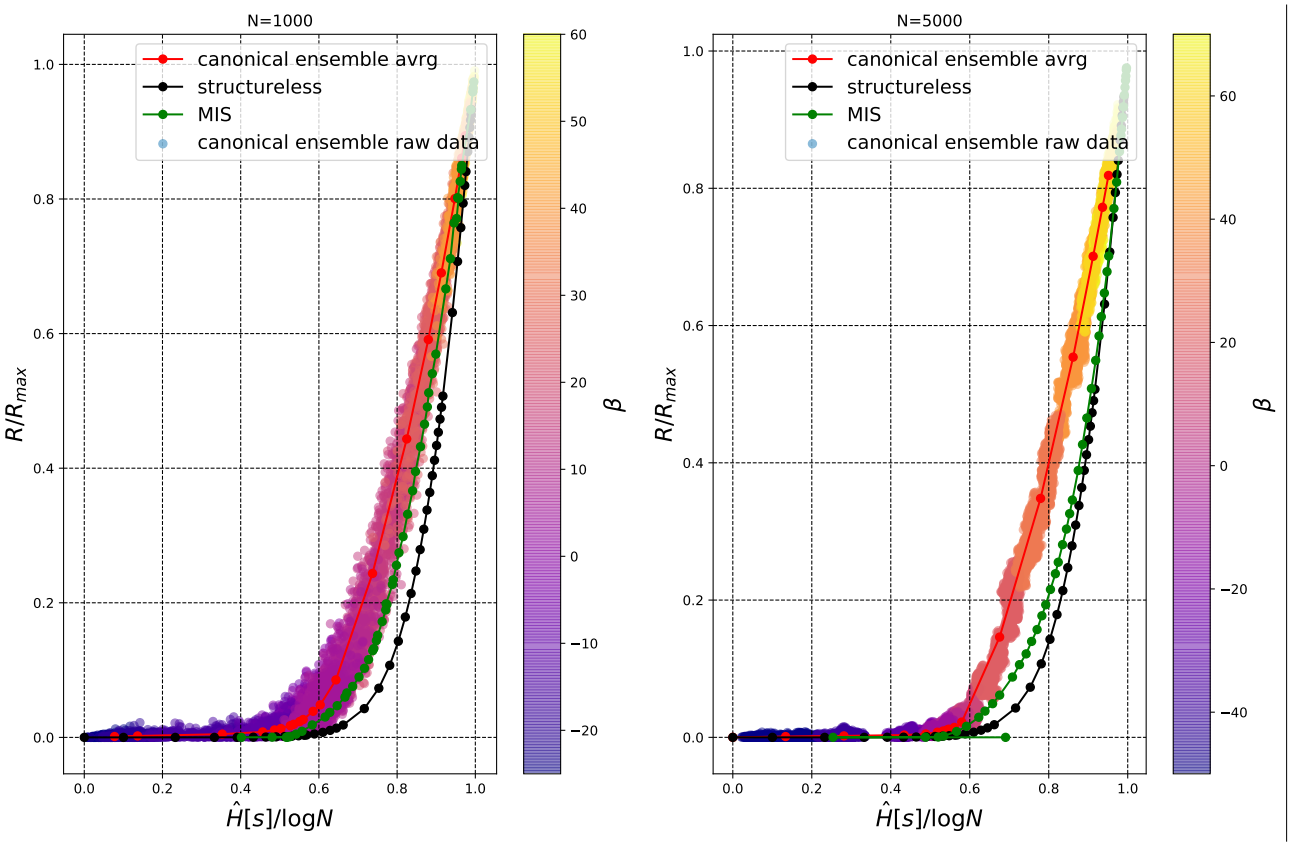


Figure 2.4: Numerical results for N = 1000,5000. Raw data are scatter-plotter together with the MIS curve and the structure-less curve.

If $\mathcal{R}[k]$ depends only on $\hat{H}[s]$, then we expect to find α_s to be significantly larger then $\alpha_k \approx 0$. We found that this is not the case.

N	$\hat{\alpha}_s^{est}$	$\hat{\alpha}_k^{est}$	$\hat{\sigma}_s$	$\hat{\sigma}_k$	$\hat{\rho}_{s,k}$
1000	$9.6 \cdot 10^{-1}$	-1.15	$2 \cdot 10^{-2}$	$4 \cdot 10^{-2}$	$-8.5 \cdot 10^{-1}$
5000	$9.3 \cdot 10^{-1}$	-1.23	$2 \cdot 10^{-2}$	$5 \cdot 10^{-2}$	$-7.7 \cdot 10^{-1}$
10000	$9.0 \cdot 10^{-1}$	-1.14	$2 \cdot 10^{-2}$	$5 \cdot 10^{-2}$	$-7.6 \cdot 10^{-1}$

Table 2.1: Least square fitting results for the MIS data-points when $\beta \geq 0$.The symbol $\hat{\sigma}$ refers to the estimated Mean Squared Error, $\hat{\rho} = \frac{cov(\hat{\alpha}_s, \hat{\alpha}_k)}{\hat{\sigma}_s \hat{\sigma}_k}$ is the estimated Pearson coefficient.The details of the computation are in the appendix A.3.

These results reveal that $\mathcal{R}[k]$, in the under-sampled regime, is a linear function of both $\hat{H}[s]$ *and* $\hat{H}[k]$ and corroborate what we see in figure 2.3.

Our conclusion is that $\mathcal{R}[k]$ can be seen as a measure of the uncertainty on *which*

is the generative model. Thus the *useful* information content $\hat{H}[k]$ is anti-correlated with the uncertainty regarding the generative algorithm $\mathcal{R}[k]$. This conclusion is general and does not depend on the fact that samples lie on the MIS curve, since the same holds for sample with fixed *Resolution* and sample size as one can see in figure 2.3.

Chapter 3

The generative model in the Huffman representation and robust bits

Each Huffman tree defines a deterministic representation $p(s|\underline{b}, T)$ which relates one or more bit-strings to a symbol s. We have argued in chapter 2 that each tree corresponds to a generative algorithm as well, when provided with a generator of binary sequences.

A probabilistic representation can then be obtained if we think that the tree itself is a random variable T:

$$\hat{p}(s) = \sum_{T \in \mathcal{T}} \sum_{\underline{b} \in \mathcal{B}} p(s|\underline{b}, T) p(\underline{b}, T)$$

It must be stressed here that the trees contained in $\mathcal{HT}$ are the those that can be generated from $\hat{p}(s)$ by Huffman coding when a convention is fixed:

when merging two nodes, the one with lower frequency is assigned bit 0.

This convention is of fundamental importance: without the latter $\pi(s|\underline{b})$ would result in a degenerate matrix with all identical entries equal to $2^{-|\mathcal{S}'|}$.

Since we can indeed distinguish between 2 nodes with different frequencies, the additional symmetry we would account for by not following this convention is broken.

The resulting generative model is specified by the conditional probability distribution:

$$\pi(s|\underline{b}) = \sum_{T \in \mathcal{HT}} p(s|\underline{b}, T) p(T|\underline{b})$$

where

- $p(s|\underline{b}, T) = \delta_{s,s_T(\underline{b})}$, since given a tree T and a code-word $\underline{b}$ we can easily retrieve the corresponding symbol $s_T(\underline{b})$.

- $p(T|\underline{b}) = \frac{p(\underline{b}|T)p(T)}{\sum_{T\in\mathcal{HT}} p(\underline{b}|T)p(T)}$.

Given a tree T, this uniquely defines a dyadic distribution $q(s) \propto 2^{-L(s)}$ over $s \in \mathcal{S}$.

If this were case, the distribution over the bit strings would be $p(\underline{b}|T) = 2^{-d} = p(\underline{b})$ and $p(T|\underline{b}) = p(T)$.

So the assumption

$$p(T|\underline{b}) \approx p(T) \tag{3.1}$$

is equivalent to approximate $\hat{p}(s)$ with the dyadic distribution dictated by the tree T.

In order to simplify our calculations we will then use the above approximation. Then

$$\pi(s|\underline{b}) \approx \sum_{T\in\mathcal{HT}} p(s|\underline{b}, T)p(T) = \sum_{T\in\mathcal{HT}} \frac{1}{|\mathcal{HT}|} p(s|\underline{b}, T) \tag{3.2}$$

where we have assumed a uniform distribution over the Huffman trees since they all provide an optimal generative algorithm as described in chapter 2.

Given $\pi(s|\underline{b})$, our aim is to retrieve the correct distribution over the binary sequences $p(\underline{b})$:

$$\hat{p}(s) = \sum_{\underline{b}} \pi(s|\underline{b})p(\underline{b})$$

Since the variables $\underline{b}$ are inferred by the evidence $\hat{s}$, they are called *hidden* or *latent* variables.

The idea of using a *mixture* of Huffman codes is related to the concept of :

- *noisy* bits $\rightarrow$ represents different symbols in different trees.They should correspond to hidden variables which do not contain much useful information because they depend on the tree, which is a particular result of Huffman coding. Such bits are *not* consistent with the symmetries dictated by the observed frequencies.

- *robust* bits $\rightarrow$ code for the same symbol in several different trees. They should correspond to meaningful hidden variables, since they do not depend on the particular code and are thus consistent with the symmetries dictated by the observed frequencies.

Construction of $\pi(s|\underline{b})$

Since $\log(|\mathcal{HT}|) \lesssim N \log N$ and there is no way, to the author knowledge, of producing all the Huffman trees consistent with a sample, $\pi(s|\underline{b})$ cannot be computed exactly.

One can then resort to exact enumeration for small sample sizes, but this approach is doomed to be applicable only in the well-sampled regime where the number of ties in the frequency is small compared to number of observed states, and when the number of states itself is small.

Nevertheless the mixture model can be estimated by means of montecarlo sampling as the empirical average

$$\hat{\pi}(s|\underline{b}) = \sum_{t=1}^{M} \frac{1}{M} p(s|\underline{b}, T^t)$$

of a sufficiently large sample extracted from $p(T)$.

We can use the same algorithm used to build a single Huffman tree, with the variant that whenever there is a tie in the frequency we choose *randomly* the 2 nodes to merge.

When the frequency index is N and a tree has been constructed, then $p(s|\underline{b}, T)$ can be stored as a 2-D array in which to each row corresponds a symbol, and to each column a bit-string.

This can be easily done because there exists a bijection $\underline{b} \in \{0,1\}^d \to n \in \{0,1,\dots,2^d-1\}$, i.e each $\underline{b}$ is in unique correspondence with the binary representation of a natural number.

The running time is $\Theta(M \cdot 2^d \cdot n)$, n being the total number of MC samples.

At this point one should know how many steps are needed for the procedure to converge. We propose to probe the convergence in two ways:

1 by monitoring the differences between the elements of the matrix π as the number of steps is increased

2 by comparing the obtained matrix with one that satisfy a *minimal symmetry requirement*, that is explained below

The differences between the elements of two matrices can be summed up in one number, the norm of the difference matrix.

We employed the Frobenius norm

$$\|M\|_F = \sum_i \sum_j M_{i,j}^2 \tag{3.3}$$

So we first computed π for increasing values of the number of Montecarlo steps $\underline{\pi} = (\pi_{t^1}, \pi_{t^2}, \dots)$, where the list is ordered such that $t^{i+1} > t^i$, and then we checked

the value

$$\|\Delta\pi_i\|_F\,, \quad \Delta\pi_i = \pi_{t^i} - \pi_{t^{i-1}} \tag{3.4}$$

when the convergence is reached we expect to find $\Delta\pi_i = 0$.

A *minimal symmetry requirement* for π is to be consistent with the symmetries of the sample.This amount to satisfy

$$\pi(s|\underline{b}) = \pi(s'|\underline{b}) \quad \forall \underline{b} \quad k_s = k'_s \tag{3.5}$$

because if 2 symbols have the same frequency they can be permuted and the resulting tree will still be an Huffman tree.

We can explicitly construct a matrix that satisfies the *minimal symmetry requirement* by enforcing the constraint 3.5. If we do so for $\hat{\pi}$, this gives us a matrix that we are going to call $\hat{\pi}_S$.

One can then also observe how the difference between the latter two matrices changes as the number of Montecarlo steps gets larger.

We expect the difference to get closer and closer to 0 as the average converges toward its actual value.

3.0.1 $p(\underline{b})$, Maximum entropy principle and Spin models

Maximum entropy distribution

We look for the *least constrained* probability distribution $p(\underline{b})$ compatible with $\hat{p}(s) = \sum_{\underline{b}\in\{0,1\}^n} \pi(s|\underline{b})p(\underline{b})$, this is given by the *maximum entropy* distribution [15].

The latter is equivalent to find the maximum of the Entropy function taking into account the desired constraints

$$\Gamma(\underline{p}) = \sum_{\underline{b}} \left\{ -p(\underline{b})\log(p(\underline{b})) + \sum_s \theta_s \left(\sum_{\underline{b}\in\{0,1\}^n} \pi(s|\underline{b})p(\underline{b}) - \hat{p}(s) \right) + \eta \left(\sum_{\underline{b}\in\{0,1\}^n} p(\underline{b}) - 1 \right) \right\}$$

the result is

$$p(\underline{b}) = e^{\eta - 1 + \sum_s \theta_s \pi(s|\underline{b})} = \frac{e^{\sum_s \theta_s \pi(s|\underline{b})}}{Z(\underline{\theta})}$$

where $\{\theta_s\}$ and η are Lagrange multipliers fixed by the conditions

$$\sum_{\underline{b}} \pi(s|\underline{b})q(\underline{b}) = \frac{\partial}{\partial\theta_s}\log\mathcal{Z}(\theta) = \hat{p}(s) \tag{3.6}$$

Integrating in θ_s we obtain that these conditions are equivalent to solve

$$\underline{\theta}^* = \arg\max\left\{ \sum_s \theta_s \hat{p}(s) - \log(Z(\underline{\theta})) \right\} = \arg\max\left\{ \mathcal{F}(\underline{\theta}) \right\}$$

because the Hessian matrix of $\mathcal{F}(\underline{\theta})$ is negative semi-definite for every value of $\underline{\theta}$ being minus the co-variance matrix of $\pi(s|\underline{b})$.

The argument of the maximum θ^*, can be found using a gradient ascent procedure

$$\theta_s^{(t+1)} = \theta_s^{(t)} + \delta\theta_s \qquad \delta\theta_s \propto \left\{\hat{p}(s) - \mathbb{E}_{\underline{\theta}^{(t)}}[\pi(s|\underline{b})]\right\}$$

the exit condition $\Delta < \xi$ is fixed by the accuracy chosen ξ and the norm employed to compute Δ.

Spin models

Since we want to retrieve a probability distribution over a set of binary variables, the natural model that seems a good candidate is the *Ising* model with heterogeneous interactions of possibly arbitrary order [5].

In the following we indicate the sequence corresponding to the binary conversion of n over d variables with $bin(n,d) = (b_1, b_2, \ldots, b_d)$. For instance

$$bin(10{,}4) = 1010 \qquad bin(10{,}4)_1 = 1 \quad bin(10{,}4)_2 = 0\ldots \tag{3.7}$$

The set of product operators on a set of d binary variables $\underline{b} = (b_1, b_2, \ldots, b_d)$

$$\phi^\mu(\underline{b}) = \prod_{i:bin(\mu)_i \neq 0} (2b_i - 1) \tag{3.8}$$

satisfy the relations

$$\sum_{\underline{b}} \phi^\mu(\underline{b}) = 0 \quad \forall\mu \tag{3.9}$$

$$\sum_{\underline{b}} \phi^\mu(\underline{b})\phi^\nu(\underline{b}) = 2^d\delta_{\mu,\nu} \tag{3.10}$$

and constitute thus a basis for the binary functions of d bits [5],[16].

The problem of retrieving $p(\underline{b})$ such that

$$\hat{p}(s) = \sum_{\underline{b}} \pi(s|\underline{b})p(\underline{b}) \tag{3.11}$$

can be rewritten as

$$\hat{p}(s) = \sum_{\mu} \Lambda^\mu(s)\mathbb{E}\left[\phi^\mu(\underline{b})\right] \tag{3.12}$$

where

$$\Lambda^\mu(s) = \frac{1}{2^d}\sum_{\underline{b}} \pi(s|\underline{b})\phi^\mu(\underline{b}) \tag{3.13}$$

This means that the probability distribution $\hat{p}(s)$ induces a set of constraints on the average values of the operators $\phi^\mu(\underline{b})$. The *least square* solution of the latter system can be obtained by pseudo-inverse matrix or gradient descent using the quadratic error as the function to be minimazed.

The maximum entropy principle then dictates that the probability distribution over the bits is of the form

$$p(\underline{b}) = \frac{e^{\sum_\mu g^\mu \phi^\mu(\underline{b})}}{\mathcal{Z}(\underline{g})} \tag{3.14}$$

where the coupling g^μ are fixed to enforce the constraints on the average values of $\phi^\mu(\underline{b})$.

With the same exact reasoning one can project the maximum entropy distribution obtained in the previous paragraph onto a fully connected heterogeneous spin model by means of the operators $\phi^\mu(\underline{b})$

$$\log p(\underline{b}) = \sum_s \theta_s \pi(s|\underline{b}) - \log \mathcal{Z}(\underline{\theta}) \tag{3.15}$$

$$= \sum_\mu g^\mu \phi^\mu(\underline{b}) \tag{3.16}$$

where we define

$$g^\mu = \sum_s \theta_s \Lambda^\mu(s) \tag{3.17}$$

It is useful to observe that obtaining $p(\underline{b})$ with gradient descent over $\{\theta_s, \quad s \in \mathcal{S}\}$ parameters and then projecting the log-probability onto the basis $\{\phi^\mu(\underline{b})\}$ is faster and easier than doing gradient descent with parameters $\{g^\mu\}$, because the number of parameters is larger in the latter case.

This approach is useful in spotting the strength of the various interactions among the variables, and gives us an explicit energy function.

On the basis of this result one can also consider different models with a subset of interactions and learn the couplings of such theories.

3.0.2 Information content of the binary variables $\underline{B}$

We briefly review the definition and meaning of the mutual information for *discrete* random variables.

Mutual information

Definition 3 (Mutual information) *The mutual information $I[X, Y]$ between two variables X, Y taking values in respectively $\mathcal{X}, \mathcal{Y}$, is defined [3] as*

$$I[X, Y] = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log\left(\frac{p(x, y)}{p(x)p(y)}\right) \tag{3.18}$$

If one recalls the definition of Shannon entropy $H[X]$ of a random variable X, taking values in $\mathcal{X}$, as

$$H[X] = \sum_{x \in \mathcal{X}} -p(x) \log p(x) \tag{3.19}$$

and of *conditional* entropy as

$$H[X|Y] = \sum_{y \in \mathcal{Y}} H[X|Y = y]p(y) = \sum_{y \in \mathcal{Y}} -p(x|y) \log(p(x|y))p(y) \tag{3.20}$$

Then the mutual information can be rewritten as

$$I[X, Y] = H[X] - H[X|Y] = H[Y] - H[Y|X] \tag{3.21}$$

Thus $I[X, Y]$ quantifies the reduction in the entropy regarding the random variable X due to the knowledge of Y, and vice-versa [3].

Since $H[X]$ corresponds to the optimal number of bits needed to encode the outcome X, we can also interpret the latter conclusion in term of *description length*.

$I[X, Y]$ is equal to the difference between the number of bits needed to encode the outcome of X without the knowledge of Y and without.

Consider for instance X, Y such that they are independent, then the value assumed by X is not influenced by the value assumed by Y.This means that $p(x, y) = p(x)p(y)$ and finally $I[X, Y] = 0$.This is consistent with the intuition that X carries no information on Y and vice-versa.

Robust and noisy bits

Since $\pi(s|\underline{b})$ is a rectangular matrix, with a number of columns $2^d \geq |\mathcal{S}| = M$, the number of bits necessary to generate the empirical distribution $\hat{p}(s)$ is *larger* than actually needed.

We argue that there exist *noisy* and *robust* bits and attempt to define quantitatively these adjectives.

To quantify the information content of each subset of bits we look at the *mutual information* $I[S, \underline{B}]$, while the number of bits d is varies from 10 to 1.

In this context

$$I[S, \underline{B}_{\leq d}] = \sum_{s \in \mathcal{S}, \underline{b} \in \{0,1\}^d} \pi(s|\underline{b})p(\underline{b}) \log(\frac{\pi(s|\underline{b})}{\hat{p}(s)}) \tag{3.22}$$

As the number of marginalization steps increase, the number of bits will decrease and the mutual information of the restricted sequence $\underline{b}_{\leq k}$ will trace a curve $I[S, \underline{B}_{\leq k}]$ for $k = 1, \ldots, d$. We expect the latter to be a decreasing function of the number of coarse-graining iterations k, or an increasing function of the number of bits.

We define *robust* bits as those that account for the *largest* fraction of mutual information.

In particular a bit is *robust* if when it is integrated out results in a significant reduction of the mutual information, as opposed to *noisy* bits which causes no variation of the latter.

If we define

$$\Delta I_k = I[S, \underline{B}_{\leq k}] - I[S, \underline{B}_{\leq k-1}] \tag{3.23}$$

then the bit k is *robust* when $\Delta I_k > 0$, *noisy* else.

Incidentally this also allow us to investigate if there is a hierarchy of *relevance* between *robust* bits:

a bit k is more *relevant* than another one k' if $\Delta I_k < \Delta I_{k'}$

Chapter 4

Application on the Us Supreme court data set

The Us Supreme Court "is the highest court in the US government,consisting of nine justices who vote on the constitutionality of legislative and executive actions " [7], to which we refer the interested reader and for a more detailed description of the data set employed.

The data set is composed by $N = 895$ court votes and have been selected being the largest available relative to a *natural* court, i.e the set of justices is the same throughout the whole data-set.

"although opinions can be nuanced, each justice casts a yes (i = +1) or no (i = 1) vote, and the majority of votes decides the fate of the case" [7]. A court decision is thus uniquely identified by a sequence of 9 bits, each of which represents the decision of a single justice. The number of possible court decisions is therefore $|\mathcal{S}| = 2^9 = 512$ while the number of distinct observed states is $|\mathcal{S}'| = M = 128 = 2^7$.

Our aim is to extract a set of binary variables by means of Huffman coding and to quantify their information content.

4.0.1 Construction of $\hat{\pi}$

The maximum depth reached by an Huffman tree compatible with the data-set is 10, so $\underline{b} \in \{0,1\}^{10}$. The matrix π will then have shape (M,2^{10}). We compute $\hat{\pi}$ for different numbers of Montecarlo steps $t^{(1)}, t^{(2)}, \ldots$, we indicate the corresponding matrices as $\hat{\pi}^{(1)}, \hat{\pi}^{(2)}, \ldots$.

To check the convergence we considered

$$\Delta\pi^{(i)} = \hat{\pi}^i - \hat{\pi}^{i-1} \tag{4.1}$$

The results of such procedure are presented in fig 4.1.

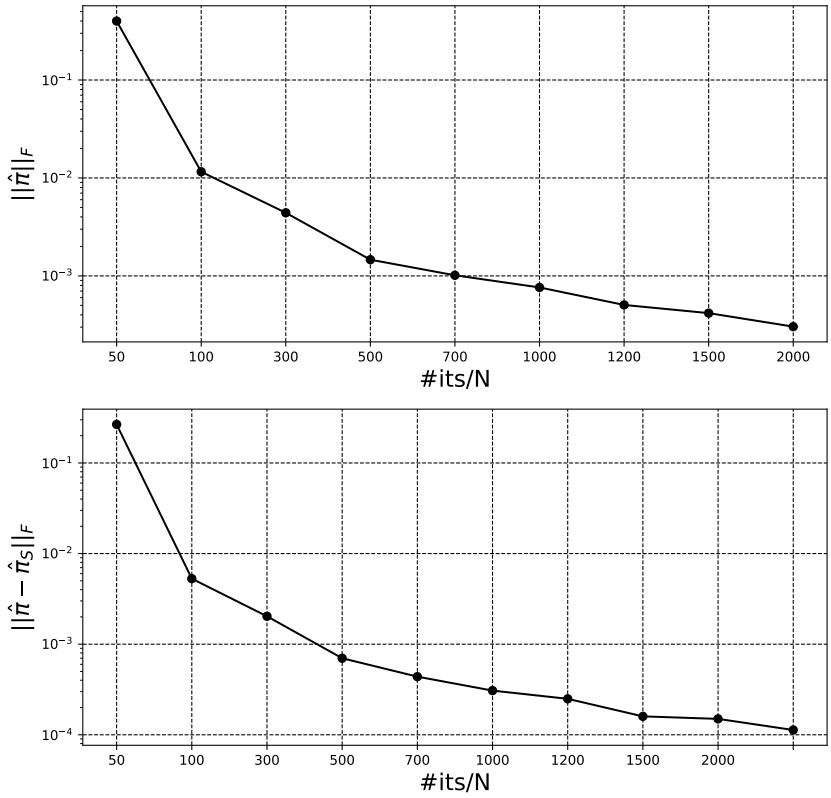


Figure 4.1: Frobenius norm of $\Delta\pi^{(t)}$ vs number of Montecarlo iterations t. The number of Montecarlo steps is rescaled by the the sample size N.The Frobenius norm of the difference matrix tend to saturate to a value close to 0 as the number of iterations increases. The absolute difference between matrix $\hat{\pi}$ and its symmetrized version $\hat{\pi}_{MS}$ rapidly decays as the number of Montecarlo steps grows.

Indeed we still cannot be sure that the average computed over $2000N$ trees is a sufficient number of steps to obtain a correct estimate of π. What we can do is to check a posteriori if our result is consistent with the *minimal symmetry* requirement dictated by the observed frequency

$$\pi(s|\underline{b}) = \pi(s'|\underline{b}), \quad \forall \underline{b}, \quad k_s = k_{s'} \tag{4.2}$$

To do so we considered the difference between the matrix $\hat{\pi}$ and its *symmetrized* version $\hat{\pi}_{MS}$, see fig 4.1.

4.0.2 How many bits are needed to generate the decision of the United States Supreme Court

We carried out the procedure delineated in the previous chapter for $\hat{\pi}$. The mutual information between the decision of the judges S and the bits $\underline{B}$ has been computed as the number of the latter varies from 10 to 1. The result is presented in fig 4.2.

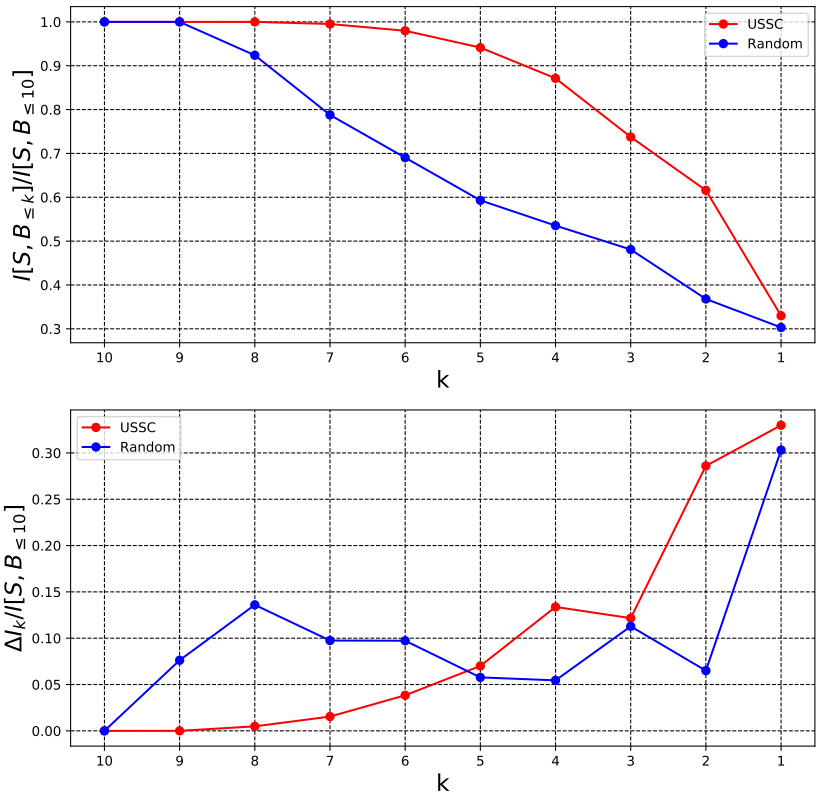


Figure 4.2: $I[\underline{B}_{\leq k}, S]/I[\underline{B}_{\leq 10}, S]$ as k goes from 10 to 1. The *random* curve is relative to π_S computed for a sample obtained by randomly tossing $N = 895$ balls in $M = 512$ boxes.

For the USSC data-points, the picture 4.2 shows that we can easily distinguish between *robust* and *noisy* bits. Furthermore *robust* bits show different degrees of *relevance*, in the sense that they account for different fractions of the total information content.

First we note that the number of bits which contain the largest fraction of the mutual information is ≤ 7. This is consistent with $M = 2^7$,since to generate M states we need at least 7 bits,i.e to each binary string $\underline{b} \in \mathcal{B}$ must be associated at least a symbol $s \in \mathcal{S}$.

We additionally expect the number of robust bits to be $\geq \hat{H}[s] \simeq 5$ [7], since this is the optimal description length for an outcome drawn from $\hat{p}(s)$ [2].

This is confirmed by observing that more than the 90% of the mutual information is contained in the first 5 bits.

We confronted these results with those obtained from the same procedure applied to a synthetic data set, that we called *Random*. The latter is obtained by randomly throwing $N = 895$ balls in 512 boxes and mimics a set of data-points resulting from a structure-less generative model.

In this case we can still observe that there is a subset of 9 bits which accounts for the whole information content.This is consistent with the fact that the number of distinct states that are observed is $M = |\mathcal{S}'| = 512 = 2^9$.

Both the data-sets resulted in ΔI_k getting larger as k gets smaller, but for *Random* the information content is spread almost evenly on all the bits.

This is consequence of the fact that the latter is a *randomly* generated sequence: the resolution $\hat{H}[s]$ is larger. Thus the number of bits necessary to encode an outcome of $S \sim \hat{p}(s)$ will be larger.

Figure 4.3 shows the mutual information between the system and the single bits,together with

$$\Delta p(b_i = 1) = p(b_i = 1) - \frac{1}{2} \tag{4.3}$$

for the different bits. What we realize from this picture is that *robust* bits are characterized by:

- $\Delta p(b_k) \neq 0$
- $I[S, B_k] > 0$

thus *noisy* bits account for the noise in the sample in a precise sense:

$\Delta p(b_k) = 0$ so they carry *no* information on the generative model.

4.0.3 The Hamiltonian of the bit-system

As explained in the previous chapter one can re-write the probability distribution $p(\underline{b})$ as a Boltzmann weight

$$p(\underline{b}) = \frac{e^{-\mathcal{H}(\underline{b})}}{\mathcal{Z}(\underline{g})}, \qquad \mathcal{H}(\underline{b}) = -\sum_{\mu=1}^{2^d-1} g^\mu \phi^\mu(\underline{b}) \tag{4.4}$$

so the couplings $\{g^\mu\}$ specify how much a given interaction contributes to shape the probability distribution.

For the USSC data-set, when only the *robust* bits are considered, the largest part of the coupling are close to zero, while 9 peaks are clearly distinguishable, see fig 4.4.

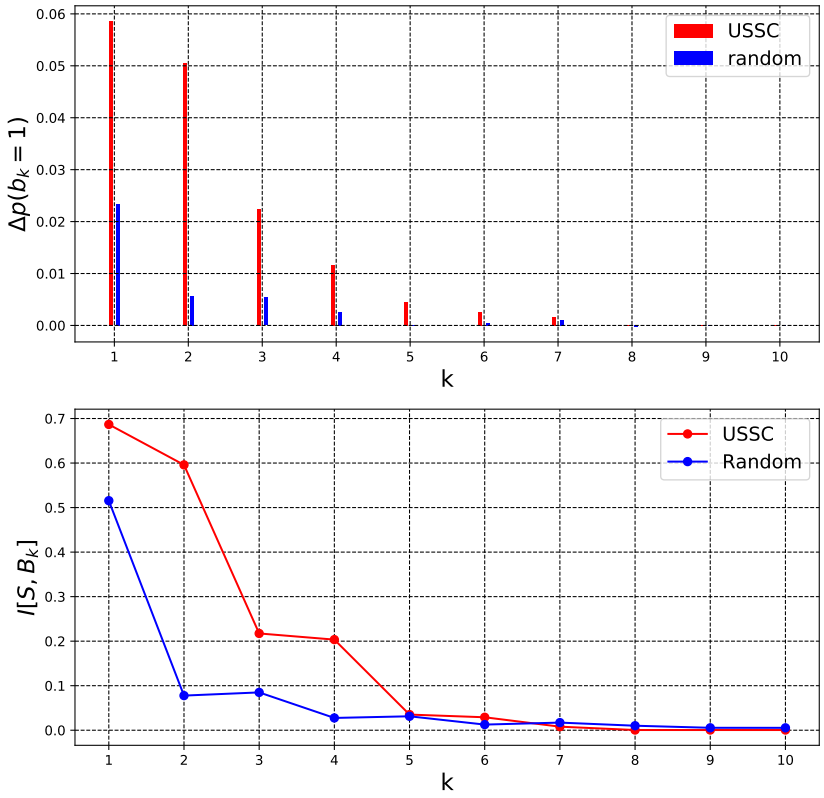


Figure 4.3: $I[S, B_k]$ as k varies

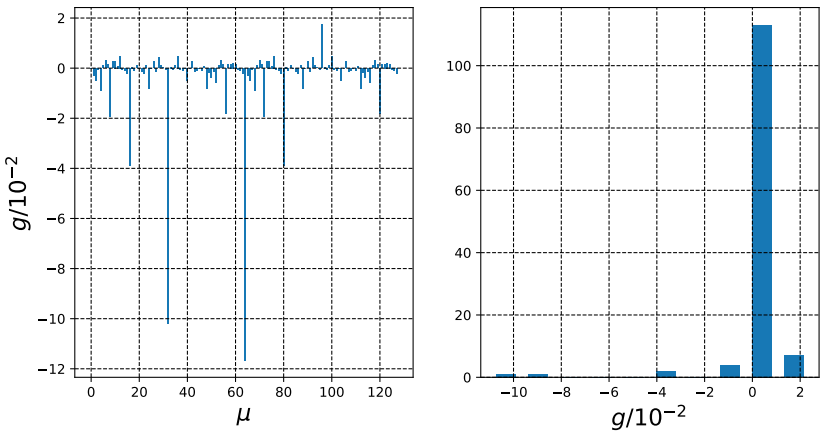


Figure 4.4: bar-plot of g^μ and corresponding histrogram.

These are 1 or 2 body interactions, while only a 3-body and 4-body operators are present. Most importantly they involve only the first four bits b_1, b_2, b_3, b_4.

The $\{g^\mu\}$ are mainly negative for *relevant* interactions as among the 9 largest only one is positive, the latter being the interaction between b_0, b_1.

An hypergraph picturing the network of interactions relative to the 9 largest couplings is in fig 4.5.

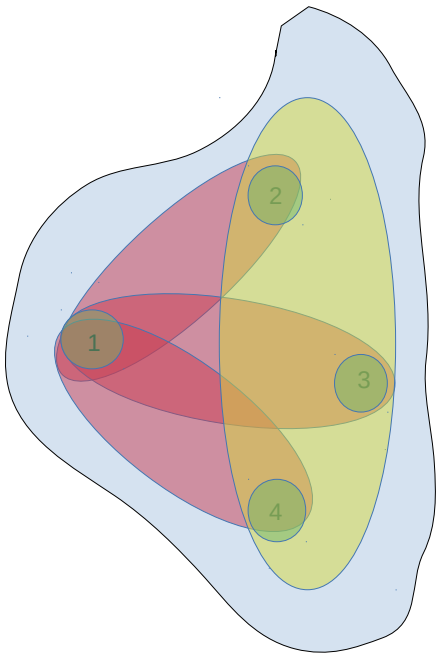


Figure 4.5: Hypergraph picturing the network of interactions for the 9 largest couplings.

We observe a hierarchical structure across $\{g^\mu\}$.

Bits can be distinguished on the basis of their position from the Most Significant Bit, which is the left most in our notation, to the Least Significant Bit.

If one order the couplings according to their magnitude, then as the number of coupling considered grows,the more less significant bits start to interact.

For instance if one considers

- the first 9 couplings $\rightarrow (b_1, b_2, b_3, b_4)$
- the first 21 couplings $\rightarrow (b_1, b_2, b_3, b_4, b_5)$
- the first 41 couplings $\rightarrow (b_1, b_2, b_3, b_4, b_5, b_6)$

This is consistent with the conclusion that the *robust* bits are characterized by different degrees of *relevance* and bits that are more significant have also an higher degree of *relevance*.

Conclusions

To summarize the original contribution of the above work, we have shown that :

- $\mathcal{R}[\hat{k}]$, defined as the logarithm of the number of Huffman trees, can be taken as a quantitative measure of the uncertainty regarding the generative model.This quantity depends both on $\hat{H}[s]$ and $\hat{H}[k]$, it is *positively-correlated* with the second, while it is *negatively-correlated* with the first.This is consistent with the definition and meaning of $\hat{H}[k]$ as the useful information that can be used to estimate the generative model [2],[6]. To show this we produced synthetic samples at fixed *average* resolution for different samples sizes. We additionally computed the *exact* solution of the MIS equation by employing the saddle point identity, thus obtaining the result by simulating a fictitious physical system in the *zero* temperature limit as explained in appendix A.1.

- when Huffman coding is exploited as an unsupervised procedure to generate representations, a set of *hidden binary* variables is extracted. The latter are organized in a hierarchy, that is quantified in a precise sense by means of the mutual information. We distinguished between *robust* and *noisy* bits on the basis of the $I[S, \underline{B}]$, the former being those that contain a non-zero fraction of mutual information. We also observe, for the data-set studied, that *robust* bits carry different amount of information and are thus organized in a hierarchy of *relevance*.

- the distribution induced over the labels of the Huffman representation (i.e the bit-strings) can be represented as a spin model [5]. The structure of the Hamiltonian is dictated by the couplings of the different interactions among the variables. We argue that the relevant interactions can be understood by estimating the couplings through the use of appropriately defined projection

operators $\phi^{\mu}(\underline{b})$ [16].For the data-set analyzed, the g^{μ} result to be organized in a hierarchy.

We conclude by remarking some problems that afflicts our analysis together with additional topics that could be worth investigating in the future.

The generative model build with the procedure explained in the text is *not* capable of generating *new* outputs, in the sense that the probability distribution is set to 0 for not-yet-seen states. This is because the symbol space $\mathcal{S}$ corresponds to the set of already-observed states, i.e those with frequency $k_s > 0$.

A possible way to solve this problem, when the *real* symbol space $\mathcal{S}^0$ is known, could be to include in the sample also those symbols with frequency 0.The Huffman procedure would theoretically work, with the only drawback of generating very deep-trees, as the number of additional bits would be at least as large as $\log(|\mathcal{S}^0 \setminus \mathcal{S}|)$. Provided that a small amount of probability is assigned to yet-unseen states, this would define a model where every state as a defined probability.But further investigations are needed in this direction.

A rigorous proof of the convergence of $\hat{\pi}$ would be preferable, even though our approximate analysis of the Frobenius norm and the confrontation with its *minimally symmetric* version seems to point in the right direction.

For what concerns the hidden variables $\underline{b}$ an important point is to understand their meaning, up to now we do not know what features the various bits code for.

Looking at the conditional probability π for the USSC, we inferred for instance that the first bit only is capable of distinguishing between the 2 most probable events, i.e the unanimous decisions of the court, and the remaining split votes.But also in this case additional investigations are needed.

Appendix A

Numerical simulations

A.1 Solving the MIS optimization problem

Through the saddle point approximation we can map the optimization problem into a statistical mechanics problem

$$\max\left\{\hat{H}[\hat{k}]+\beta\hat{H}[\hat{s}]+\mu\hat{N}\right\}=\lim_{\gamma\to+\infty}\frac{1}{\gamma}\left\{\log\left(\sum_{\underline{m}}e^{\gamma(\hat{H}[\hat{k}]+\beta\hat{H}[\hat{s}]+\mu\hat{N})}\right)\right\} \tag{A.1}$$

Note that the constraints written as

$$\hat{H}[\hat{s}]=\epsilon\sim O(\log N) \tag{A.2}$$

$$\hat{N}=N\sim O(N) \tag{A.3}$$

imply that $\hat{H}[\hat{s}],\hat{H}[\hat{k}]$ are intensive quantities, while $\hat{N}$ is extensive, thus is better to re-write the constraints as

$$N\hat{H}[\hat{s}]=E\sim O(N\log N) \tag{A.4}$$

$$\hat{N}=N\sim O(N) \tag{A.5}$$

and rescale the multipliers thus setting the scaling of the various term contributing to the *Energy* to $O(N)$.

This leads to

$$\max\left\{\hat{N}\hat{H}[\hat{k}]+\beta\hat{N}\hat{H}[\hat{s}]+\mu\hat{N}\right\}=\lim_{\gamma\to+\infty}\left\{\frac{1}{\gamma}\log\left(\sum_{\underline{m}}e^{\gamma(\hat{N}\hat{H}[\hat{k}]+\beta\hat{N}\hat{H}[\hat{s}]+\mu\hat{N})}\right)\right\} \tag{A.6}$$

Now we can identify

$$\mathcal{Z}(\beta,\gamma,\mu)=\sum_{\underline{m}}e^{\gamma(\hat{N}\hat{H}[\hat{k}]+\beta\hat{N}\hat{H}[\hat{s}]+\mu\hat{N})}$$

and the Boltzmann weight reads

$$\mathcal{P}_{\beta,\gamma,\mu}(\underline{m}) \propto e^{\gamma(\hat{N}\hat{H}[\hat{k}]+\beta\hat{N}\hat{H}[\hat{s}]+\mu\hat{N})} \tag{A.7}$$

This means that the solution of the problem is accessible trough simulations techniques in the appropriate limit.

The value of A.6 is exactly the free energy of this statistical mechanics model in the '*zero-temperature*' limit

$$\lim_{\gamma\to+\infty} \mathcal{F}(\beta,\mu,\gamma) = \lim_{\gamma\to+\infty} \frac{1}{\gamma} \log\left(\sum_{\underline{m}} e^{\gamma(\hat{N}\hat{H}[\hat{k}]+\beta\hat{N}\hat{H}[\hat{s}]+\mu\hat{N})}\right) \tag{A.8}$$

A.1.1 Montecarlo simulation

We set up a Montecarlo Markov Chain procedure in order to sample configurations $\underline{m}$ according to the measure A.7.

Two options are available :

- propose moves which conserve the number of samples,thus sampling at fixed N,hence $\mu = 0$
- propose moves that *do not* conserve the number of samples,thus $\mu \neq 0$

Proposal

If the first option is preferred, as in this case, then a possible way of proposing the moves is to consider a combination of 2 moves,which separately do not conserve N:

evaporation pick one frequency at random with $m_k > 0$

$$m_k \leftarrow m_k - 1 \tag{A.9}$$
$$m_{k-1} \leftarrow m_{k-1} + 1 \tag{A.10}$$

deposition pick one frequency at random with $m_k > 0$ and

move a

$$m_k \leftarrow m_k - 1 \tag{A.11}$$
$$m_{k+1} \leftarrow m_{k+1} + 1 \tag{A.12}$$

or move b

$$m_k \leftarrow m_k \tag{A.13}$$
$$m_1 \leftarrow m_1 + 1 \tag{A.14}$$

The probability with which the 2 possible moves in *deposition* are chosen is fixed so that each final state is reached with the same probability. If we indicate with $l(\underline{m}) = \sum_{k=1}^{N} 1[m_k > 0]$ then

move a with probability $p_a = \frac{l}{l+1}$

move b with probability $p_b = \frac{1}{l+1}$

Even though detailed balance is satisfied, this way of proposing moves is not symmetrical : $l(\underline{m})$ can be different in the initial$\underline{m}$ and final $\underline{m}'$ configurations.

Acceptance

This in turn means that the acceptance of the MCMC, fixed by the Metropolis-Hastings rule [17], is

$$\min\left(e^{n(\Delta\hat{H[k]}_{\underline{m},\underline{m}'}+\nu\Delta\hat{H[s]}_{\underline{m},\underline{m}'})+\log(\frac{l(\underline{m})}{l(\underline{m}')})},1\right)$$

where the correction term $e^{\log(\frac{l(\underline{m})}{l(\underline{m}')})}$ is equal to the ratio of the forward and backward proposal probability $\frac{Q_{\underline{m}\to\underline{m}'}}{Q_{\underline{m}'\to\underline{m}}}$.

A.2 Synthetic data production

We would like to produce samples $\underline{m} = (m_1, \ldots, m_N)$ such that

$$\hat{N} = \sum_k k m_k = N \tag{A.15}$$

$$N\hat{H}[\hat{s}] = -\sum_k k m_k \log\frac{k}{N} = E \tag{A.16}$$

$$N\hat{H}[\hat{k}] = -\sum_k k m_k \log\frac{k m_k}{N} = S \tag{A.17}$$

One would thus want to sample from the distribution

$$p(\underline{m})_{E,S,N} = \frac{1}{\Omega(E,S,N)}\delta(N\hat{H}[\hat{s}] - E)\delta(N\hat{H}[\hat{k}] - S)\delta_{\sum_k k m_k, N}$$

unfortunately sampling from this distribution is extremely inefficient and depend on the tolerance allowed on the values E, S,thus a *smoothed* version must be considered. *Information theory* tells us that the least constrained distribution consistent with the requirements above is the one that maximaze

$$\Gamma_{\beta,\lambda,\mu}(q) = \sum_{\underline{m}} -q(\underline{m})\Big\{\log q(\underline{m}) - \beta(\sum_k k m_k \log\frac{k}{N}) - \lambda(\sum_k k m_k \log\frac{k m_k}{N}) - \mu(\sum_k k m_k)\Big\} \tag{A.18}$$

this is the *maximum entropy distribution*

$$\mathcal{P}_{\beta,\lambda,\mu} = \frac{1}{\mathcal{Z}(\beta,\lambda,\mu)} e^{+\beta N \hat{H}[\hat{s}] + \lambda N \hat{H}[\hat{k}] + \mu \hat{N}} \tag{A.19}$$

the parameters β, λ, μ are fixed by the equations

$$E = \frac{\partial}{\partial \beta} \log Z(\beta,\lambda,\mu) = \mathbb{E}_{\beta,\lambda,\mu}\left[N \hat{H}[\hat{s}] \right] \tag{A.20}$$

$$S = \frac{\partial}{\partial \lambda} \log Z(\beta,\lambda,\mu) = \mathbb{E}_{\beta,\lambda,\mu}\left[N \hat{H}[\hat{k}] \right] \tag{A.21}$$

$$N = \frac{\partial}{\partial \mu} \log Z(\beta,\lambda,\mu) = \mathbb{E}_{\beta,\lambda,\mu}\left[\hat{N} \right] \tag{A.22}$$

Consider then sampling from this *smoothed* distribution.

At this point it is not clear what parameters should we use and to gain some intuition on at least the order of magnitude of the parameters controlling $\mathcal{P}_{\beta,\lambda}$ we can search for approximate solutions .

A.2.1 β, μ Ensemble

First we consider the distribution obtained by fixing $\mathbb{E}\left[N \hat{H}[\hat{s}] \right]$ and $\mathbb{E}\left[\hat{N} \right]$,

We will thus consider the measure of a grand-canonical ensemble

$$\mathcal{P}_{\beta,\mu}(\underline{m}) = \frac{e^{(+\beta N \hat{H}[\hat{s}] + \mu \hat{N})}}{\mathcal{Z}(\beta,\mu)}$$

Note that

$$\beta N \hat{H}[\hat{s}] + \mu \hat{N} = -\beta \sum_k m_k k \log \frac{k}{\hat{N}} + \mu \hat{N} \tag{A.23}$$

$$= -\beta \sum_k m_k k \log \frac{k}{N} - \beta \sum_k m_k k \log \frac{N}{\hat{N}} + \mu \hat{N} \tag{A.24}$$

$$= \beta N \sum_k \epsilon_k m_k + \mu N \sum_k \frac{k m_k}{N} + \beta N \frac{\hat{N}}{N} \log \frac{\hat{N}}{N} \tag{A.25}$$

$$= \beta' \tilde{H}[\hat{s}] + \mu' \frac{\hat{N}}{N} + \beta' \frac{\hat{N}}{N} \log \frac{\hat{N}}{N} \tag{A.26}$$

where we defined

$$\epsilon_k = -\frac{k}{N} \log \frac{k}{N} \tag{A.27}$$

we can then re-write the normalization factor as

$$\mathcal{Z}(\beta',\mu') = \sum_{\underline{m}} e^{+\beta' \tilde{H}[\hat{s}]+\mu' \frac{\hat{N}}{N}+\beta' \frac{\hat{N}}{N}\log\frac{\hat{N}}{N}} = \mathbb{E}^0_{\beta',\mu'}\left[e^{+\beta' \frac{\hat{N}}{N}\log\frac{\hat{N}}{N}}\right]\mathcal{Z}(\beta',\mu')^0$$

where $\mathbb{E}^0_{\beta',\mu'}\left[\right]$ is intended as the expectation value using the measure

$$\mathcal{P}^0_{\beta',\mu'} = \frac{e^{+\beta' \tilde{H}[\hat{s}]+\mu' \frac{\hat{N}}{N}}}{\mathcal{Z}^0(\beta,\mu)}$$

As a *rough* approximation we can expand in power series, akin to a high temperature expansion in stat. mechanics

$$\mathcal{Z}(\beta,\mu) = \mathcal{Z}_0(\beta,\mu)\sum_{k=0}^{+\infty}\frac{\beta'^k}{k!}\mathbb{E}^0_{\beta',\mu'}\left[\left(\frac{\hat{N}}{N}\log\frac{\hat{N}}{N}\right)^k\right] \tag{A.28}$$

the *lowest* order of approximation is valid when

$$\beta'\mathbb{E}^0_{\beta',\mu'}\left[\frac{\hat{N}}{N}\log\frac{\hat{N}}{N}\right] \ll 1$$

this is valid when $\hat{N} \approx N$ and must be checked a-posteriori.

If $\mathbb{E}^0_{\beta',\mu'}\left[\frac{\hat{N}}{N}\right] \approx 1$ then

$$\mathbb{E}^0_{\beta',\mu'}\left[\frac{\hat{N}}{N}\log\frac{\hat{N}}{N}\right] = \mathbb{E}^0_{\beta',\mu'}\left[\frac{\hat{N}}{N}-1\right] + \frac{1}{2}\mathbb{E}^0_{\beta',\mu'}\left[\left(\frac{\hat{N}}{N}-1\right)^2\right] + \dots \tag{A.29}$$

$$\approx \frac{1}{2N^2}\mathbb{V}^0_{\beta',\mu'}\left[\hat{N}\right] \tag{A.30}$$

so it is sufficient to check the variance of $\hat{N}$ to have an estimate of the validity of the approximation.

In the regime of validity of the approximation explained above, we have

$$1 = \frac{\partial}{\partial\mu'}\log\mathcal{Z}^0(\beta',\mu') = \mathbb{E}^0_{\bar{\beta},\bar{\mu}}\left[\frac{\hat{N}}{N}\right] = \sum_{k=0}^{+\infty}\frac{k}{N}\frac{1}{e^{(\beta\epsilon_k+\mu k)}-1} = \sum_{k=0}^{+\infty}\frac{k}{N}n_k \tag{A.31}$$

$$e = \frac{\partial}{\partial\beta'}\log\mathcal{Z}^0(\beta',\mu') = \mathbb{E}^0_{\bar{\beta},\bar{\mu}}\left[\tilde{H}[\hat{s}]\right] = \sum_{k=0}^{+\infty}\epsilon_k\frac{1}{e^{(\beta\epsilon_k+\mu k)}-1} = \sum_{k=0}^{+\infty}\epsilon_k n_k \tag{A.32}$$

Solving this set of coupled non-linear equations is equivalent to find the maximum of

$$\mathcal{F}^0(\beta',\mu') = \beta e + \mu - \log\mathcal{Z}^0(\beta,\mu)$$

since the Hessian of $\mathcal{F}^0$ is negative semi-definite for every value of β', μ' being equal to the co-variance matrix of $\hat{H}[\hat{s}], \hat{N}$.

Gradient ascent has been employed, with

$$\delta\beta' = \alpha\left(1 - \mathbb{E}^0_{\beta,\mu}\left[\frac{\hat{N}}{N}\right]\right) \tag{A.33}$$

$$\delta\mu' = \alpha\left(e - \mathbb{E}^0_{\beta,\mu}\left[\tilde{H}[\hat{s}]\right]\right) \tag{A.34}$$

The variances can then be easily computed using the fluctuation dissipation theorem as

$$\mathbb{V}^0_{\beta,\mu}\left[\hat{N}\right] = \frac{\partial}{\partial\mu'}\mathbb{E}^0_{\beta,\mu}\left[\hat{N}\right] = \sum_{k=0}^{+\infty} k^2(1+n_k)n_k \tag{A.35}$$

$$\mathbb{V}^0_{\beta,\mu}\left[\hat{H}[\hat{s}]\right] = \frac{\partial}{\partial\beta'}\mathbb{E}^0_{\beta,\mu}\left[\tilde{H}[\hat{s}]\right] = \sum_{k=0}^{+\infty} \epsilon_k^2(1+n_k)n_k \tag{A.36}$$

The numerical solutions for different values of N all result in the fluctuations growing with β and being a finite fraction of the theoretical mean.

This implies that our result should only be trusted for small positive values of β.Indeed we observe by confrontation with the simulations that this is not the case: there is an almost exact agreement between the theoretical average and the average of the simulation.

Alternative derivation of the upper bound for the MIS curve

An upper-bound on $\mathbb{E}^0_{\beta,\mu}\left[\hat{H}[\hat{k}]\right]$ can be derived as a consequence of the well known Jensen's inequality

$$\mathbb{E}\left[X \log X\right] \geq \mathbb{E}\left[X\right] \log \mathbb{E}\left[X\right]$$

which implies

$$\mathbb{E}^0_{\beta,\mu}\left[\hat{H}[\hat{k}]\right] \leq -\sum_k \frac{k}{N}\mathbb{E}^0_{\beta,\mu}\left[m_k\right] \log(\frac{k}{N}\mathbb{E}^0_{\beta,\mu}\left[m_k\right]) = \tilde{\mathbb{E}}^0_{\beta,\mu}\left[\hat{H}[\hat{k}]\right]$$

It must additionally be remembered that $\hat{H}[\hat{k}]$ is bounded from above by the *data processing inequality* and thus $\hat{H}[\hat{k}] \leq \hat{H}[\hat{s}]$.

Continuum limit approximation Since

$$\mathbb{E}^0_{\beta,\mu}\Big[\hat{H}[\hat{k}]\Big] = \sum_k \frac{k}{N}\mathbb{E}^0_{\beta,\mu}\Big[m_k \log m_k\Big] - \mathbb{E}^0_{\beta,\mu}\Big[\hat{H}[\hat{s}]\Big]$$

and

$$m_k \sim \frac{e^{-(\beta\epsilon_k+\mu k)m_k}}{\mathcal{Z}_k(\beta,\mu)}$$

consider the expectation

$$\mathbb{E}\Big[-X\log X\Big] = -\sum_{X=0}^{\infty}\frac{e^{-\alpha x}}{\mathcal{Z}(\alpha)}x\log x \approx \int_0^{\infty} dy \frac{1}{\alpha^2}\frac{e^{-y}}{\mathcal{N}(\alpha)}y(\log\alpha - \log y) = \quad \text{(A.37)}$$

$$= \int_{-\infty}^{+\infty} dz \frac{1}{\alpha^2\mathcal{N}(\alpha)} e^{-(e^{-z}+z)}e^{-z}(z+\log\alpha) \quad \text{(A.38)}$$

noting that $Z \sim e^{-(e^{-z}+z)}$ is a Gumbel distributed random variable, we can rewrite the above integral as a function of its moment generating function $\phi_Z(t)$

$$\phi(t) = \int_{-\infty}^{+\infty} dz e^{-(e^{-z}+z)}e^{tz} = \int_0^{+\infty} ds e^{-s}s^{1-t-1} = \Gamma(1-t)$$

as

$$\mathbb{E}\Big[-X\log X\Big] \approx \frac{1}{\alpha^2\mathcal{N}(\alpha)}\left(\log\alpha\phi(t)|_{t=-1} - \frac{\partial}{\partial t}\phi(t)|_{t=-1}\right) = \frac{\log\alpha+1-\gamma}{\alpha^2\mathcal{N}(\alpha)} = \boxed{\frac{\log\alpha+1-\gamma}{\alpha}}$$

the normalization factor $\mathcal{N}(\alpha) = \frac{1}{\alpha}$ is fixed so that the continuous approximation is correctly normalized.

Thus asymptotically as $N \to \infty$

$$\mathbb{E}^0_{\beta,\mu}\Big[\hat{H}[\hat{k}]\Big] \sim \sum_{k=0}^{N}\frac{\log\alpha_k+1-\gamma}{\alpha_k}$$

this means that in the continuous approximation $\mathbb{E}\Big[-X\log X\Big] \simeq -\mathbb{E}\Big[X\Big]\log\mathbb{E}\Big[X\Big]$.

This result suggests that as $N \to +\infty$ the value of $\hat{H}[\hat{k}]$ should approach its upperbound.

A.2.2 β, N ensemble

Thanks to the insight gained in the previous paragraph we already know the order of magnitude of the multipliers and their relation with the quantities they tune. We then use our educated guesses on the parameters to simulate the system at fixed

$\hat{N}$ with the MCMC developed to solve the MIS equation 2.9 setting the multiplier of $\hat{H}[\hat{k}]$ to 0.

The samples extracted are in *quantitative* agreement with the approximate results of the previous section (see figA.1), even though we stressed that our intent was only to estimate the order of magnitude of the parameters β, μ: we expected large errors due to the fluctuations of $\hat{N}$. A possible explanation of the latter fact is that the function $f(x) = e^{-x \log x}$ is well approximated by $f(x) \simeq 1 - (x - 1)$ when $x \simeq 1$, thus the average effect of the fluctuations around 1 sum up to be 0.

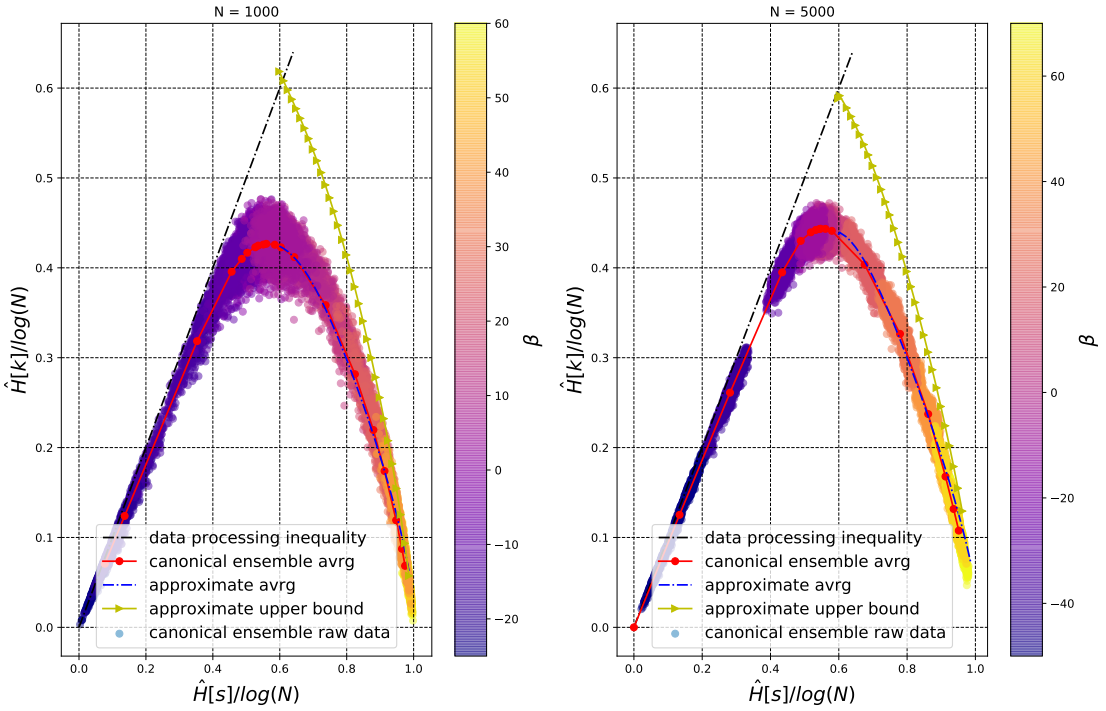


Figure A.1: Numerical results for N = 1000, N = 5000.The raw data from the β, N ensemble, here called *canonical*, are presented together with the theoretical average computed with the approximated measure of the β, μ ensemble of previous section.We include also the upper bound on the MIS curve computed in the previous section.

A.3 Least square fitting of the linear relation $\mathcal{R}[\hat{k}], \hat{H}[\hat{s}], \hat{H}[\hat{k}]$

In order to search for the parameters α_s, α_k such that

$$\mathcal{R}[\hat{k}] = \alpha_s \hat{H}[\hat{s}] + \alpha_k \hat{H}[\hat{k}] \tag{A.39}$$

defining $\mathcal{R}[\hat{k}]_i = Y_i$ and $X_i = (\hat{H}[\hat{s}]_i, \hat{H}[\hat{k}]_i)$, $1 \leq i \leq N$ the above problem can be stated as

$$\underline{Y} = X \cdot \underline{\alpha} + \underline{\epsilon} \tag{A.40}$$

where $\underline{\epsilon}$ is a set of N independent and identically distributed draws of a random variable with finite support,i.e the noise.

As a measure of goodness of fit we take the mean squared error

$$\Delta(\underline{\alpha}) = (\underline{Y} - X \cdot \underline{\alpha})^T (\underline{Y} - X \cdot \underline{\alpha}) \tag{A.41}$$

then

$$\hat{\underline{\alpha}} = \arg\min_{\underline{\alpha}} \Delta(\underline{\alpha}) = (X^T X)^{-1} X^T \underline{Y} = M \underline{Y} \tag{A.42}$$

the matrix $M = (X^T X)^{-1} X^T$ is known in the literature as Moore-Penrose pseudo-inverse, since it generalizes the concept of inverse for non-square matrices.

Since the Least Square estimator $\hat{\underline{\alpha}}$ depends on the data $\underline{Y}$, it is a random variable. We can quantify the error on the fitting coefficient by the variance of $\hat{\underline{\alpha}}$

$$\mathbb{V}\left[\hat{\alpha}_i\right] = \sum_{k,j} M_{ik} M^T_{ji} \mathbb{E}\left[(Y_k - \mathbb{E}[Y_k])(Y_j - \mathbb{E}[Y_j])\right] = (M^T M)_{i,i} \mathbb{V}[\epsilon] \tag{A.43}$$

where we used the fact that the errors are uncorrelated and identically distributed

$$\mathbb{E}[(Y_k - \mathbb{E}[Y_k]) \cdot (Y_j - \mathbb{E}[Y_j])] = \mathbb{V}[\epsilon] \delta_{k,j}$$

with the same reasoning we obtain that the co-variance

$$Cov\left[\hat{\alpha}_i \hat{\alpha}_j\right] \mathbb{E}\left[(\hat{\alpha}_i - \mathbb{E}[\hat{\alpha}_i])(\hat{\alpha}_j - \mathbb{E}[\hat{\alpha}_j])\right] = (M^T M)_{i,j} \mathbb{V}[\epsilon] \tag{A.44}$$

The (unbiased) estimator for the variance of the error is

$$\hat{\sigma}^2_\epsilon = \sum_{i=1}^{N} \frac{(Y_i - \underline{X}_i \cdot \underline{\alpha})^2}{N-1} \tag{A.45}$$

So the errors on $\hat{\alpha}_i$ are given by

$$\hat{\sigma}^2_i = (M^T M)_{i,i} \hat{\sigma}^2_\epsilon \tag{A.46}$$

and the Pearson correlation coefficient $\hat{\rho}_{i,j}$ is given by

$$\hat{\rho}_{i,j} = \frac{(M^T M)_{i,j}}{\sqrt{(M^T M)_{i,i}} \sqrt{(M^T M)_{j,j}}} \tag{A.47}$$

A.4 Marginalization of π

The marginalization procedure can be carried out in 2 directions :

1 integrate out the Least Significant Bit (LSB):$p_{\underline{b}} \to p_{\underline{b}'}$ where $\underline{b} = (b_1, \ldots, b_d)$ and $\underline{b}' = (b_1, \ldots, b_{d-1})$.

2 integrate out the Most Significant Bit(MSB):$p_{\underline{b}} \to p_{\underline{b}'}$ where $\underline{b} = (b_1, \ldots, b_d)$ and $\underline{b}' = (b_2, \ldots, b_d)$.

LSB-Coarse Graining is implemented by observing that configurations $\underline{b}$ which are binary numbers corresponds to natural numbers in base 10 and these number corresponds to column induces of $\pi(s|\underline{b})$. Two binary numbers which differ only by the LSB correspond to natural numbers whose difference is 1(we use the Python notation for concatenation as '+').

$$\pi(s|\underline{b}') \leftarrow \pi(s|\underline{b}' +' 0')p(b_d = 0|\underline{b}') + \pi(s|\underline{b}' +' 1')p(b_d = 1|\underline{b}')$$

MSB-Coarse Graining is similarly implemented by noting that binary numbers that differ by the MSB are natural numbers in base 10 that differ by 2^{d-1} where d is the length of the binary sequence,formally:

$$\pi(s|\underline{b}') \leftarrow \pi(s|'0' + \underline{b}')p(b_1 = 0|\underline{b}') + \pi(s|'1' + \underline{b}')p(b_1 = 1|\underline{b}')$$

When reducing the number of bits we will always employ the LSB method.The MSB method is useful in combination with the LSB to compute marginals, as one can sequentially integrate first the bits that are less significant and then those that are more significant.

Bibliography

[1] M.Marsili O.Duranthon and R.Xie. «Maximal Relevance and Optimal LearningMachines». In: *arXiv:1909.12792 [physics.data-an]* () (cit. on p. iv).

[2] Ryan John Cubero Junghyo Jo Matteo Marsili Yasser Roudi and Juyong Song. «Statistical criticality arises in Most Informative Rapresentations». In: *J.Stat.Mech.(2019)063402* (). DOI: 10.1088/1742-5468/ab16c8 (cit. on pp. iv, v, 13, 15, 16, 33, 36).

[3] Thomas M.Cover Joy A.Thomas. *Element of information theory*. Wiley (cit. on pp. v, 2, 3, 5, 16, 27, 28).

[4] David A.Huffman. «A method for the construction of minimum-redundancy codes». In: *Proceedings of the IRE Volume: 40 Issue: 9 Sept. 1952* (1952), pp. 891–921. DOI: 10.1109/JRPROC.1952.273898 (cit. on pp. v, 3, 4, 7).

[5] Luigi Gresele Matteo Marsili. «On Maximum Entropy and inference». In: *Entropy 2017 19 642* (). DOI: 10.3390/e19120642 (cit. on pp. v, 26, 36).

[6] Matteo Marsili. «On sampling and modelling complex systems». In: *J.Stat.Mech.(2013)P0900* () (cit. on pp. v, 13–16, 36).

[7] William Bialek Edward D. Lee Chase P. Broedersz. «Statistical physics of the Us supreme court». In: *Journal of Statistical Physics, Volume 160, Issue 2, pp 275â301* (). DOI: 10.1007/s10955-015-1253-6 (cit. on pp. vi, 30, 33).

[8] Peter Grunwald. «A tutorial introduction to Minimum Description Length principle». In: *arXiv:math/0406077 [math.ST]* () (cit. on p. 2).

[9] J.Rissanen and G.G.Langdon. «Arithmetic coding». In: *IBM J.RES. DEVELOP.;VOL.23;NO.2;MARCH 1979* () (cit. on p. 3).

[10] J.Ziv and A.Lempel. «Compression of Individual Sequences via Variable-Rate Coding». In: *IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-24, NO. 5, SEPTEMBER 1978* () (cit. on p. 3).

[11] T.Welch. «A technique for High-Performance data compression». In: *Computer. 17 (6): 8–19.* () (cit. on p. 3).

[12] D.E.Knuth. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*. Addison-Wesley (cit. on p. 11).

[13] M.Buro. «On the maximum length of Huffman codes». In: *Information Processing Letters 45 (1993),Elsevier* () (cit. on p. 11).

[14] A.Haimovici and M.Marsili. «Criticality of mostly informative samples: A Bayesian model selection approach». In: *arXiv:1502.00356* () (cit. on pp. 13, 16–18).

[15] E.T.Jaynes. «Information theory and statistical mechanics». In: *The Physical Review,Vol 106,No 4,620-630,May 15,1957* () (cit. on p. 25).

[16] Shun-ichi Amari. *Information Geometry and Its Applications*. Springer (cit. on pp. 26, 37).

[17] W.K.Hastings. «Monte Carlo sampling methods using Markovchains and their applications». In: *Biometrika(1970), 57, 1, p.97* () (cit. on p. 40).