VI : Distribution Simulation

We are going to mainly focus on the approximate distribution simulation setup, but as an intro we'll start by examining the exact simulation case.

Going forward, we will see how approximate distribution simulation is related to information-theoretic security and wiretap channels.

Exact Simulation Channels

Goal: We are interested in understanding how many random bits (fair coin flips) are needed to simulate a random variable $V \sim P_V \in \mathcal{P}(\mathcal{V})$

What do we mean by simulation?

Given random bits $W_1, W_2, \ldots \overset{iid}{\sim} \text{Ber}(\frac{1}{2})$ we want to design an algorithm $A : \{0,1\}^* \to \mathcal{V}$ such that the distribution of $A(W_1, W_2, \ldots)$ equals P_V.

Definition (Simulation): Let $\underline{W} = \{W_i\}_{i=1}^{\infty}$ be iid $\text{Ber}(\frac{1}{2})$ r.v's. An algorithm A that takes $\underline{W}$ as an input and gives values in $\mathcal{V}$ as outputs simulated $V \sim P_V \in \mathcal{P}(\mathcal{V})$ if $\text{Law}(A(\underline{W})) = P_V$.

Example: Let

$$V = \begin{cases} a & \text{w.p. } 1/2 \\ b & \text{w.p. } 1/4 \\ c & \text{w.p. } 1/4 \end{cases}$$

How many bits are needed to simulate V and how can we do that?

Define

$$A(\underline{W}) = \begin{cases} a & \text{if } W_1 = 0 \quad \leftarrow 1/2 \\ b & \text{if } W_1 W_2 = 10 \quad \leftarrow 1/4 \\ c & \text{if } W_1 W_2 = 11 \quad \leftarrow 1/4 \end{cases}$$

Clearly,

$$\text{Law}(A(\underline{W})) = P_V .$$

Observe that the expected number of bits the algorithm uses is

$$1 \cdot \tfrac{1}{2} + 2 \cdot \tfrac{1}{2} = 1.5 \text{ bits}$$

- 1: # bits needed to simulate a
- $\tfrac{1}{2}$: $\mathbb{P}(a)$
- 2: # bits needed to sim b or c
- $\tfrac{1}{2}$: $\mathbb{P}(B) + \mathbb{P}(C)$

Observe that

$$H(V) = H(P_V) = \frac{1}{2} \cdot \log_2 2 + \frac{1}{4} \log_2 4 = 1.5 \text{ bits}$$

This is **NOT** a coincedence!

To show the fundamental connection between the expected # bits needed to simulate a distribution and the entropy of the simulated distribution, we need to formulate the problem.

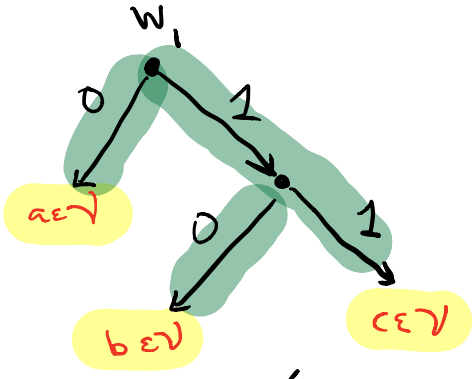
Formal Setup

Resources: $\underline{W} = (W_i)_{i=1}^{\infty}$ (iid $\mathrm{Ber}(\frac{1}{2})$ coin flips)

Target: Simulate $V \sim \underline{P}_V = (p_1, \ldots, p_m) \in \mathcal{P}(\{1, \ldots, m\})$

Algorithm: We describe the algorithm mapping strings of bits $W_1, W_2, \ldots$ to possible outcomes in $\mathcal{V}$ via a binary tree.

The leaves of the tree are marked by output letters (in the alphabet $\mathcal{V}$), and the path to the leaves is given by a sequence of bits produced by our fair coin flips.

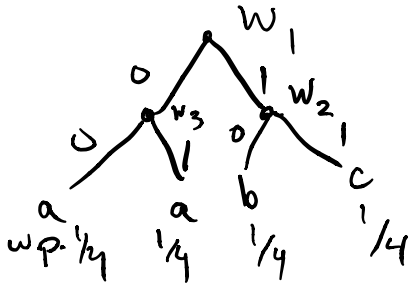


The tree corresponding to the algorithm satisfies

1) No Ambiguity: The tree should be complete (i.e. every node is either a leaf or has two children). The tree may be infinite.

2) Fair iid Coins: The probability of a leaf at depth k is 2^{-k}. Many leaves may be labeled w/ the same output symbol - therefore the total probability of all these leaves should equal the desired probability of the output of the corresponding output symbol.

Example of (2) $V = \begin{cases} a & \text{w.p. } 1/2 \\ b & \text{w.p. } 1/4 \\ c & \text{w.p. } 1/4 \end{cases}$



3) The expected number of fair bits $\mathbb{E}[T]$ that the algorithm uses equals the expected depth of the tree (where each edge is chosen w.p. $1/2$)

Lower Bound

If the output of an algorithm is $V \sim P_V \in \mathcal{P}(\mathcal{V})$ with $H(V)$, which is measured in bits (i.e $\log_2$), we would expect $H(V)$ bits are necessary to simulate that output.

Indeed, we will see that $\boxed{\mathbb{E}[T] \geq H(V)}$. But first we need a lemma.

To state it, let us describe $\mathbb{E}[T]$ in slightly more explicit notation:

- Let $\mathcal{L}$ be the set of leaves of a binary tree T_2.
- Let $Q \in \mathcal{P}(\mathcal{L})$ be $Q(l) = 2^{-d(l)}$, where $l \in \mathcal{L}$ and $d(l)$ is the depth of l.

$\Rightarrow$ The expected depth of tree is given by

$$\mathbb{E}[T] = \mathbb{E}_Q[d(L)]$$

"L is a rv" $L \sim Q$

Lemma: $\mathbb{E}[T] = \mathbb{E}_Q[d(L)] = H(Q) \geq H(P_V)$

Proof:

$$\mathbb{E}_Q[d(L)] = \sum_{l \in \mathcal{L}} Q(l)\, d(l) = -\sum_{l \in \mathcal{L}} \underbrace{2^{-d(l)}}_{Q(l)} \log_2 2^{-dl} = H(Q)$$

Q.E.D

Theorem (Lower Bound on $\mathbb{E}[T]$): For any algorithm $\mathcal{A}$, generating $V \sim P_V \in \mathcal{P}(\{1,\dots,m\})$ from W, we have $\mathbb{E}[T] \geq H(V)$, where T is the expected number of bits/coin flips $\mathcal{A}$ uses.

Proof:

$\mathcal{A}(W)$ can be represented by a complete binary tree. Label each $l \in \mathcal{L}$ be a **distinct** symbol $y(l)$, and set $\mathcal{Y} := \{y(l)\}_{l \in \mathcal{L}}$ $(y(l) \neq y(l') \Leftrightarrow l \neq l')$.

Let Y be a random variable with distribution

$$Q_Y(y) := 2^{-d(l^{-1}(y))}$$

where $l^{-1}(y)$ is the unique leaf labeled as y.

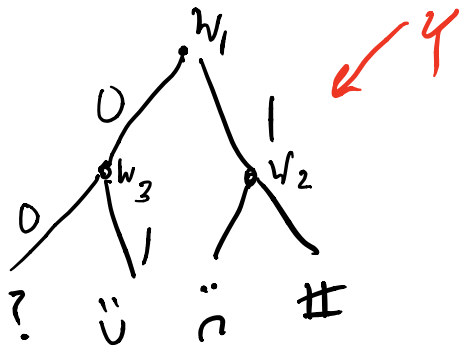
By the lemma $\Rightarrow$

$$\mathbb{E}[T] = H(Y)$$

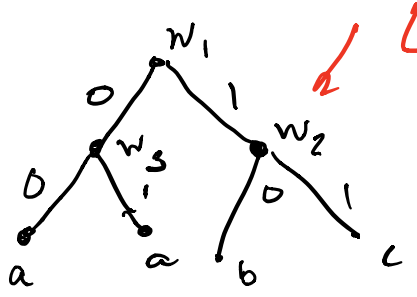
The tree simulates V. Since one or more of the leaves are mapped onto the same output symbols in V, we have that V is a deterministic function of Y, and thus

$$\mathbb{E}[T] = H(Y) \geq H(f(Y)) = H(V)$$

$$V = \begin{cases} a & \text{w.p } 1/2 \\ b & \text{w.p. } 1/4 \\ c & \text{w.p. } 1/4 \end{cases}$$



Y



$Law(A(w)) = P_V$

$$f(y) = \begin{cases} a, & y = ?, \ddot{u} \\ b, & y = \ddot{n} \\ c, & y = \# \end{cases}$$

The entropy lower bound is achievable in some cases, but a general upper bound for any P_V distribution is $H(V) + 2$.

Theorem: For any $P_V \in \mathcal{P}(\{1,2,\ldots,m\})$ there exists an optimal algorithm $A(W)$ for simulating P_V with

$$H(V) \leq \mathbb{E}(T) \leq H(V)+2$$

[result NOT proved, but result shows entropy plays a key role in describing a solution to an operational problem]

Approximate Distribution Simulation - Soft Covering Lemma

We just saw that simulating a distribution exactly requires # of random bits/coin flips that equals the entropy of that distribution.

In particular, to simulate a product distribution $P_V^{\otimes n}$ we need $H(P_V^{\otimes n}) = n \cdot H(V)$ random bits (equivalently $W \sim \text{Unif}(\{1,\ldots,2^{nH(V)}\})$).

$P_V^{\otimes n} = H(V^n)$ (i.i.d), $n \cdot H(V) = H(P_V)$

approx: $\delta_{TV}(Q_{V^n}^{(alg)}, P_V^{\otimes n}) \xrightarrow{n\to\infty} 0$

} aside, makes sense later

What if we back off from the exact simulation requirement to an approximate one? Can we do w/ less than $n \cdot H(V)$ bits?

Turns out that provided a noisy channel, we can! The quantification of this fact is known in the IT literature as the soft-covering lemma.

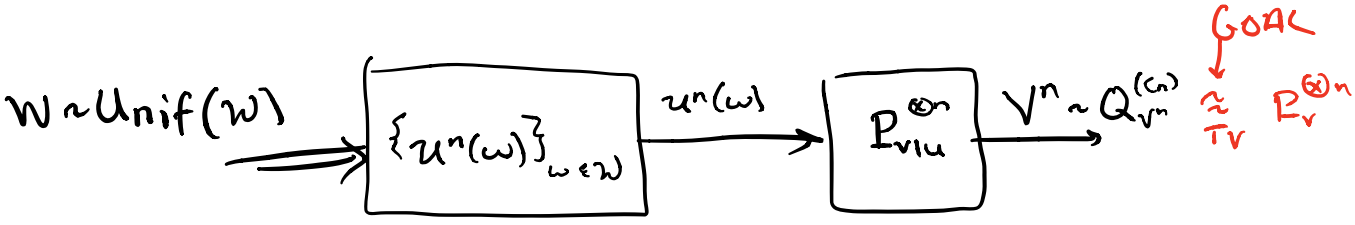
Setup: Our goal is to simulate $P_V^{\otimes n}$, where $P_V \in \mathcal{P}(\mathcal{V})$, approximately, e.g., in total variation or KL-divergence or...

Resources:

- A uniform rv $W \sim \text{Unif}(\mathcal{W})$, $\mathcal{W} = \{1, 2, \ldots, 2^{nR}\}$,
- A codebook of sequences $\{u^n(w)\}_{w \in \mathcal{W}} \in \mathcal{U}^n$,
- A noisy channel $P_{V|U}$ from $\mathcal{U}$ to $\mathcal{V}$.

We assume that the channel is such that $\exists P_U \in \mathcal{P}(\mathcal{U})$ with $\sum_{u \in \mathcal{U}} P_U(u) \cdot P_{V|U}(v|u) = P_V(v)$, $\forall v \in \mathcal{V}$

Strategy: We will uniformly pick a codeword $u^n(w)$ from the codebook and pass it through the $P_{V|U}^{\otimes n}$ channel to get an output sequence V^n with distribution $Q_{V^n}^{(c_n)}$. We will try to design the codebook (choosing the rate AND the codewords) to make $Q_{V^n}^{(c_n)}$ approximate $P_V^{\otimes n}$ in total variation.



Discussion Section [continuation of above]

The output of above channel is

$$Q_{V^n}^{(c_n)} \cdot Q_{V_1,V_2,\ldots,V_n}^{(c_n)}$$ ← NOT a product measure, there's correlation b/t symbols

Induced Joint Distribution

$$\sum_{w,u^n} Q_{W,U^n,V^n}^{(c_n)}(w,u^n,v^n) = \frac{1}{|\mathcal{W}|} \mathbb{1}_{\{u^n = u^n(w)\}} P_{V|U}^{\otimes n}(v^n|u^n)$$

$$Q_{V^n}^{(c_n)}(v^n) = \frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} P_{V|U}^{\otimes n}(v^n|u^n(w))$$

Q: For which case can the upper bound be improved to $H(V)$?

$$H(V) \leq \mathbb{E}[T] \leq H(V)$$

Def (Dyadic Distribution): We say that $P \in \mathcal{P}(\mathcal{V})$ is dyadic if $\log_2 \frac{1}{P(v)} \in \mathbb{N} \ \forall v \in \mathcal{V}$.

Equivalently, $P(v) = 2^{-k(v)}$, $k(v) \in \mathbb{N}$, $\forall v \in \mathcal{V}$.

Theorem: If $P_V \in \mathcal{P}(\{1,2,\ldots,m\})$ is dyadic then there exists an optimal algorithm for simulating $V \sim P_V$ such that

$$\mathbb{E}[T] = H(V)$$

Proof Outline: We'll describe how to build a tree with $H(V)$ coins used on average that represents $\mathbb{P}_V$ exactly.

Note that since $\mathbb{P}_V(v) = 2^{-k(v)}$ for some $k: V \to \mathbb{N}$, each $v \in V$ corresponds to a path in a complete binary tree leading to a leaf of depth $k(v)$.