

## 1 Hybrid Lemma

The hybrid lemma is a statement of continuity among distributions.

**Theorem 1** (Hybrid Lemma). *Suppose there is a sequence of distributions  $X_1, X_2, X_3, \dots, X_{n-1}, X_m$  and an algorithm  $A$  that distinguishes  $X_1$  from  $X_m$  with probability  $p$ , meaning  $|\Pr_{x \leftarrow X_1}\{A(x) = 1\} - \Pr_{x \leftarrow X_m}\{A(x) = 1\}| < p$ . Then for some  $1 \leq i \leq m-1$  there exists an algorithm that distinguishes  $X_i$  from  $X_{i+1}$  with probability at least  $\frac{p}{m-1}$ .*

*Proof (by contradiction).* Define  $p_i = \Pr_{x \leftarrow X_i}\{A(x) = 1\}$ . Suppose for each  $1 \leq i \leq m-1$ , we have  $|p_i - p_{i+1}| < \frac{p}{m-1}$ . Then, adding together distances from 1 all the way to  $m$ , we have

$$\begin{aligned} |p_i - p_{i+1}| &\geq \frac{p}{m-1} \\ |p_1 - p_2| + |p_2 - p_3| + \dots + |p_{m-1} - p_m| &< (m-1) \cdot \frac{p}{m-1} \\ |p_1 - p_2| + |p_2 - p_3| + \dots + |p_{m-1} - p_m| &< p \end{aligned}$$

Using triangle inequality,

$$|p_1 - p_m| \leq |p_1 - p_2| + |p_2 - p_3| + \dots + |p_{m-1} - p_m| < p$$

However, by premise, we have  $|p_1 - p_m| \geq p$

$$p \leq |p_1 - p_m| < p$$

This is a contradiction. □

**Corollary 2.** *If  $\mathcal{X} \approx_c \mathcal{Y}$  and  $\mathcal{Y} \approx_c \mathcal{Z}$ , then  $\mathcal{X} \approx_c \mathcal{Z}$*

The Prediction Lemma gives an alternative definition for computational indistinguishability. The idea is, you have any algorithm  $A$  which distinguishes between two ensembles. If the algorithm's accuracy can't be non-negligibly better than  $\frac{1}{2}$ , the ensembles are computationally indistinguishable.

**Theorem 3** (Prediction Lemma). *Two ensembles  $\mathcal{X}^{(0)}$  and  $\mathcal{X}^{(1)}$  are computationally indistinguishable iff for every NUPPT algorithm  $A$ , there exists a negligible  $\varepsilon(\cdot)$  s.t. for each  $n \in \mathbb{N}$ ,*

$$\Pr_{b \leftarrow U\{0,1\}}[A(1^n, t) = b | t \leftarrow X_n^{(b)}] < \frac{1}{2} + \varepsilon(n)$$

## 2 PRGs

We improve on the one-time pad by using a shorter key length. Instead of encoding  $m \oplus k_1$  where  $k_1$  is long, we encode  $m \oplus g(k_2)$ , where  $k_2$  is short, but the function  $g$  turns it into a random string. PRGs (pseudo-random generators) are functions  $g$  such that

1.  $g$  is a function (1 input to only 1 output)
2.  $g$  takes in a binary string and returns another binary string  $g : \{0, 1\}^* \rightarrow \{0, 1\}^*$
3.  $g$  is efficiently computable (polynomial time) and deterministic
4.  $g$ 's output is longer than the input  $|g(x)| > |x|$  for all  $x \in \{0, 1\}^*$
5. pseudo-randomness:  $\{g(x) | x \leftarrow U\{0, 1\}^n\} \approx_c \{U\{0, 1\}^{n+1}\}$

The last condition states that if  $g$  constructs a  $n + 1$ -bit uniform random binary string, the output should be computationally indistinguishable from a uniform random  $(n + 1)$ -bit binary string. This defines a single-bit expansion PRG.

### Example

Polynomial:  $x_{i+1} = (a \times x_i + b) \bmod p$  is not a **PRG**. As we can predict the value of next state based on the value of current state. And we can't distinguish this with true random with non-negligible probability through frequency analysis.

**Lemma 4.** *If there exists an **PRG**, it holds that  $NP \neq P$ . (Shows great impact to those cryptographic objects, e.g. secure encryption where length of key is less than length of text  $|K| < |M|$ )*

*Proof.* We want to find a set  $L$ , where  $L \in NP$ , but  $L \notin P$ . And the

$$L := \{\text{all strings outputted by PRG}\} = \{g(x) : x \in \{0, 1\}^*\}$$

If function  $g$  is a PRG,  $L \in NP$  by for  $\forall y \in L$ , there  $\exists x$  s.t.  $g(x) = y$  is the witness  $\rightarrow L \in NP$

Assume for contradiction, if language  $L \in P : \exists A$  s.t.  $A(y) = 1$  (if  $y \in L$ ) where  $A$  is polynomial time computable, and could determine whether given  $y$  is in  $L$ . We use  $A$  as a distinguisher.

Then we have  $Pr_{t \in L_n}[A[t] = 1] = 1$  and  $Pr_{t \in U\{0, 1\}^{n+1}}[A[t] = 1] < \frac{1}{2}$ , which shows that the language  $L$  is not computationally distinguishable, thus it holds that  $g$  is not PRG.  $\square$