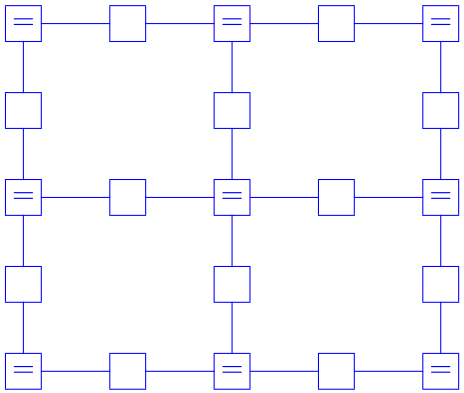
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Importance Sampling to Estimate the Entropy Function in Graphical Models with Cycles

Mehdi Molkaraie and Hans-Andrea Loeliger

Information and Signal Processing Laboratory ETH Zurich



Workshop on Graphical Models: The Fields Institute, 16–18 April 2012

Problem Setting

Let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of N binary RV's ie each with alphabet $\mathcal{X} = \{0, 1\}$

Let $f(x_1, x_2, \ldots, x_N)$ be a nonnegative function $f : \mathcal{X}^N \rightarrow \mathbb{R}$

The support of f

$$\mathcal{S}_f = \{x \in \mathcal{X}^N : f(x) > 0\}$$

Suppose $f(x)$ factors into several nonnegative "local functions" each having some subset of $\{x_1, x_2, \ldots, x_N\}$ as arguments ie

$$f(x) = \prod_{R \in \mathcal{R}} f_R(x_R)$$

Example:
Suppose $N = 5$ and $\mathcal{R} = \big\{\{1, 2, 3\}, \{3, 4, 5\}, \{5\}\big\}$

$$f(x) = f_{\{1,2,3\}}(x_1, x_2, x_3) f_{\{3,4,5\}}(x_3, x_4, x_5) f_{\{5\}}(x_5)$$

Quantities of Interest

We can define/compute

- A probability mass function $p(x)$ on $\mathcal{X}^N$ as

$$p(x) = \frac{f(x)}{Z_f}$$

- The normalization constant, the partition function, Z_f

$$Z_f = \sum_{x \in \mathcal{X}^N} f(x)$$

- The marginals of $p(x)$ on R

$$p_R(x_R) = \sum_{x \setminus x_R} p(x)$$

- The entropy function

$$H(X) = -\sum_{x \in \mathcal{X}^N} p(x) \log_2 p(x) = -\mathrm{E}_p \left[\log_2 p(X) \right]$$

Graphical Models: Factor Graphs

We use (Forney) factor graphs [KFL01] to express factorizations as

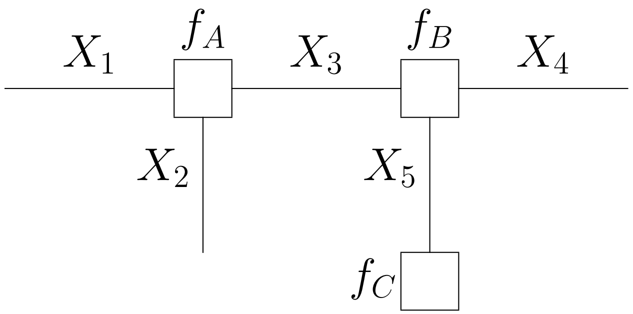
$$f(x) = \prod_{R \in \mathcal{R}} f_R(x_R)$$

A factor graphs consists of

- nodes (representing factors), and
- edges/half edges (representing variables).

Example 1:

$$f(x) = f_A(x_1, x_2, x_3) f_B(x_3, x_4, x_5) f_C(x_5)$$



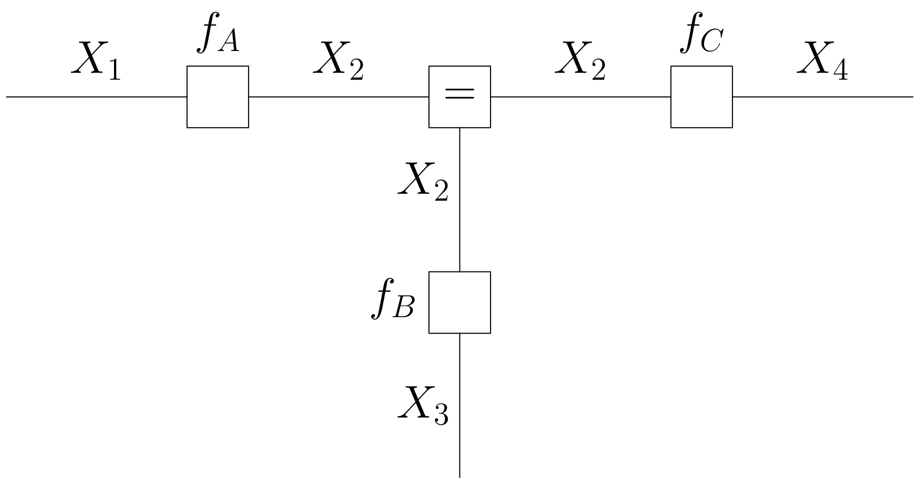
Factor Graphs : Cloning

If a variable appears in more than two factors, we clone the variable.

$$f_1(x)f_2(x)f_3(x) = f_1(x)f_2(x')f_3(x'')\delta(x - x')\delta(x - x'')$$

Example 2:

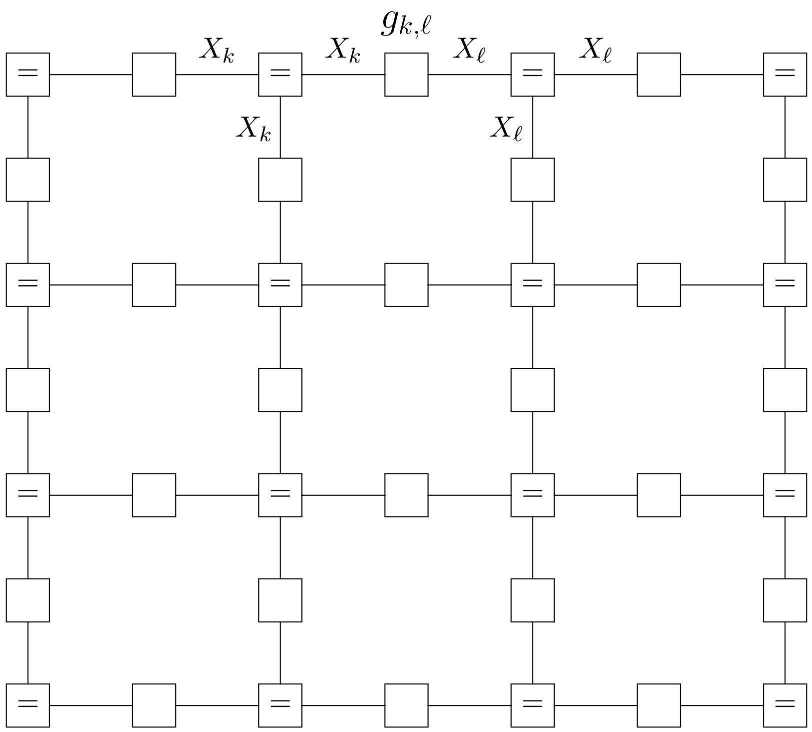
$$f(x) = f_A(x_1, x_2)f_B(x_2, x_3)f_C(x_2, x_4)$$



Factor Graphs : 2D Model

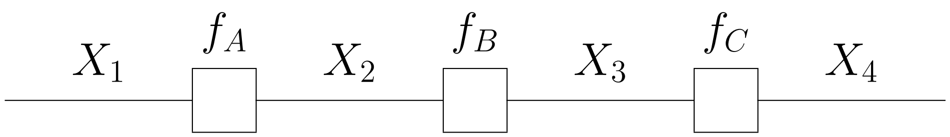
Example 3:

$$f(x_1, \ldots, x_N) \; = \prod_{\text{neighbors } (x_k, x_\ell)} g_{k,\ell}(x_k, x_\ell)$$



Cycle-Free Factor Graphs: The Sum-Product Algorithm

If f has a cycle-free factor graph representation, the sum-product algorithm can compute Z_f and the marginals $p_R(x_R)$ efficiently (after a finite number of steps).



Similar algorithms:

The sum-product algorithm on factor graphs.
J. Pearl's belief propagation algorithm.
Forward/Backward algorithm.
BCJR on trellises.

Cycle-Free Factor Graphs: Sampling $\&$ $H(X)$

Consider

$$p(x) \propto f_A(x_1, x_2) f_B(x_2, x_3) f_C(x_3, x_4)$$

By reparameterization

$$p(x) \; = \; \frac{p(x_1, x_2)\, p(x_2, x_3)\, p(x_3, x_4)}{p(x_2)\, p(x_3)}$$

$$p(x) \; = \; p(x_1)\, p(x_2|x_1)\, p(x_3|x_2)\, p(x_4|x_3)$$

In a cycle-free graph

- It is easy to draw samples according to $p(x)$

- Entropy decomposes

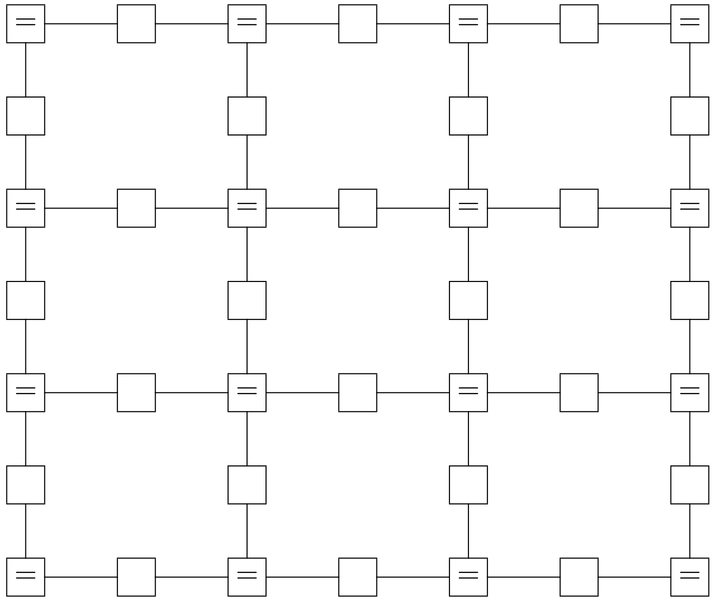
$$H(X) = H(X_1, X_2) + H(X_2, X_3) + H(X_3, X_4) - H(X_2) - H(X_3)$$

Constrained 2D Model with Cycles: Z_f

Consider a constrained 2D model as of size $N = m \times m$

$$f(x_1, \ldots, x_N) = \prod_{\text{neighbors } (x_k, x_\ell)} g(x_k, x_\ell)$$

$$g(x_k, x_\ell) = \begin{cases} 0, & \text{if } x_k = x_\ell = 1 \\ 1, & \text{else} \end{cases}$$



Constrained 2D Model

In this case

$$Z_f = \sum_{x \in \mathcal{X}^N} f(x) = \text{number of valid configurations} = |\mathcal{S}_f|$$

The entropy rate $\frac{1}{N} H(X) = \frac{1}{N} \log_2 Z_f$

Example 4

For a 2×2 model

0	0	1	0	0	1	0	0	0	0	1	0	0	1
0	0	0	0	0	0	1	0	0	1	0	1	1	0

The entropy rate is

$$\frac{1}{4} \log_2 7 = 0.701 \quad \text{bits/symbol}$$

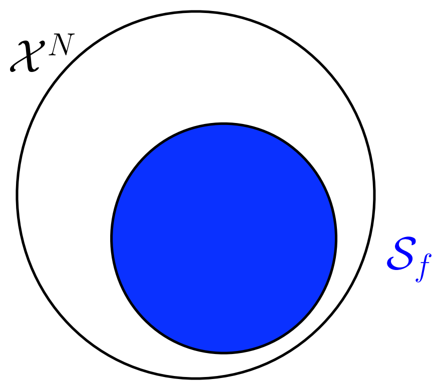
Estimating $1/Z_f$

Gibbs Sampling

1. Draw samples $x^{(1)}, x^{(2)}, \ldots, x^{(K)} \in \mathcal{S}_f$ according to $p(x)$
2. Compute:

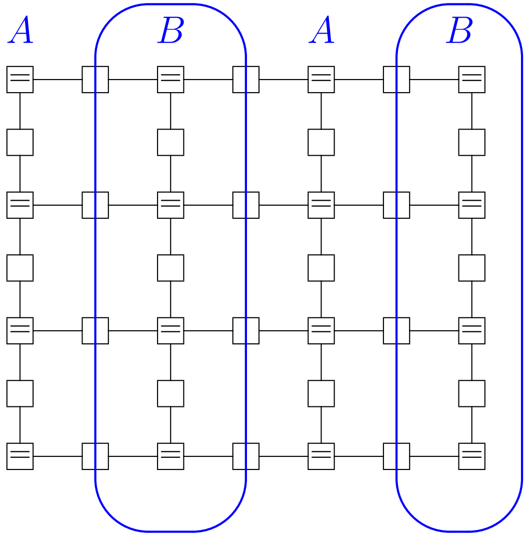
$$\hat{\Gamma} = \frac{1}{K \cdot |\mathcal{S}_f|} \sum_{k=1}^{K} \frac{1}{f(x^{(k)})}$$

$\Rightarrow E[\hat{\Gamma}] = 1/Z_f$



Tree-Based Gibbs Sampling

Partition the index set $\{1, \ldots, N\}$ into two parts (A, B) such that by fixing x_A or x_B the remaining factor graph is cycle-free.



Generate samples $(x_A^{(1)}, x_B^{(1)}), (x_A^{(2)}, x_B^{(2)}), \ldots$ by alternating between

- sampling $x_A^{(k)}$ according to $p(x_A | x_B = x_B^{(k-1)}) \propto f(x_A, x_B^{(k-1)})$
- sampling $x_B^{(k)}$ according to $p(x_B | x_A = x_A^{(k)}) \propto f(x_A^{(k)}, x_B)$

Mixes faster than Gibbs sampling

Tree-Based Estimation of $1/Z_f$

Suppose

$$f_A(x_A) \triangleq \sum_{x_B} f(x_A, x_B)$$

Therefore

$$\begin{aligned} Z_{f_A} &= \sum_{x_A} f_A(x_A) \\ &= Z_f \end{aligned}$$

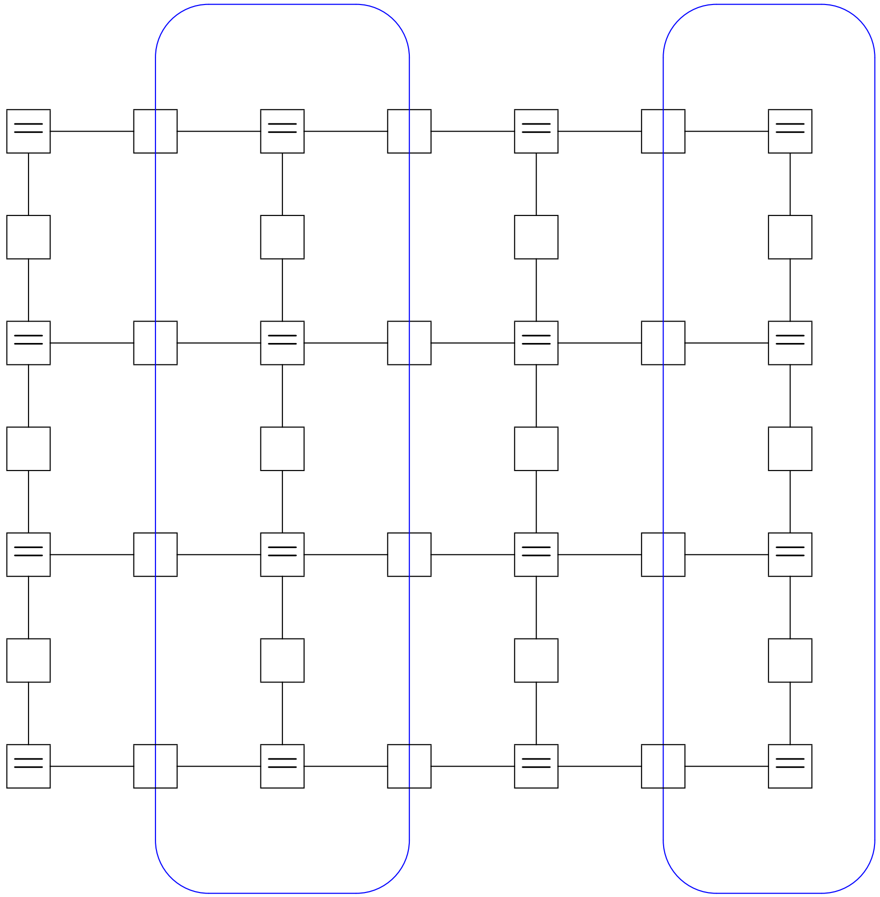
$\Longrightarrow$ Modified Gibbs sampler to estimate $1/Z_{f_A}$ by:

$$\hat{\Gamma}_A = \frac{1}{K \cdot |\mathcal{S}_{f_A}|} \sum_{k=1}^{K} \frac{1}{f_A(x_A^{(k)})}$$

where

$$f_A(x_A^{(k)}) = \sum_{x_B} f(x_A^{(k)}, x_B)$$

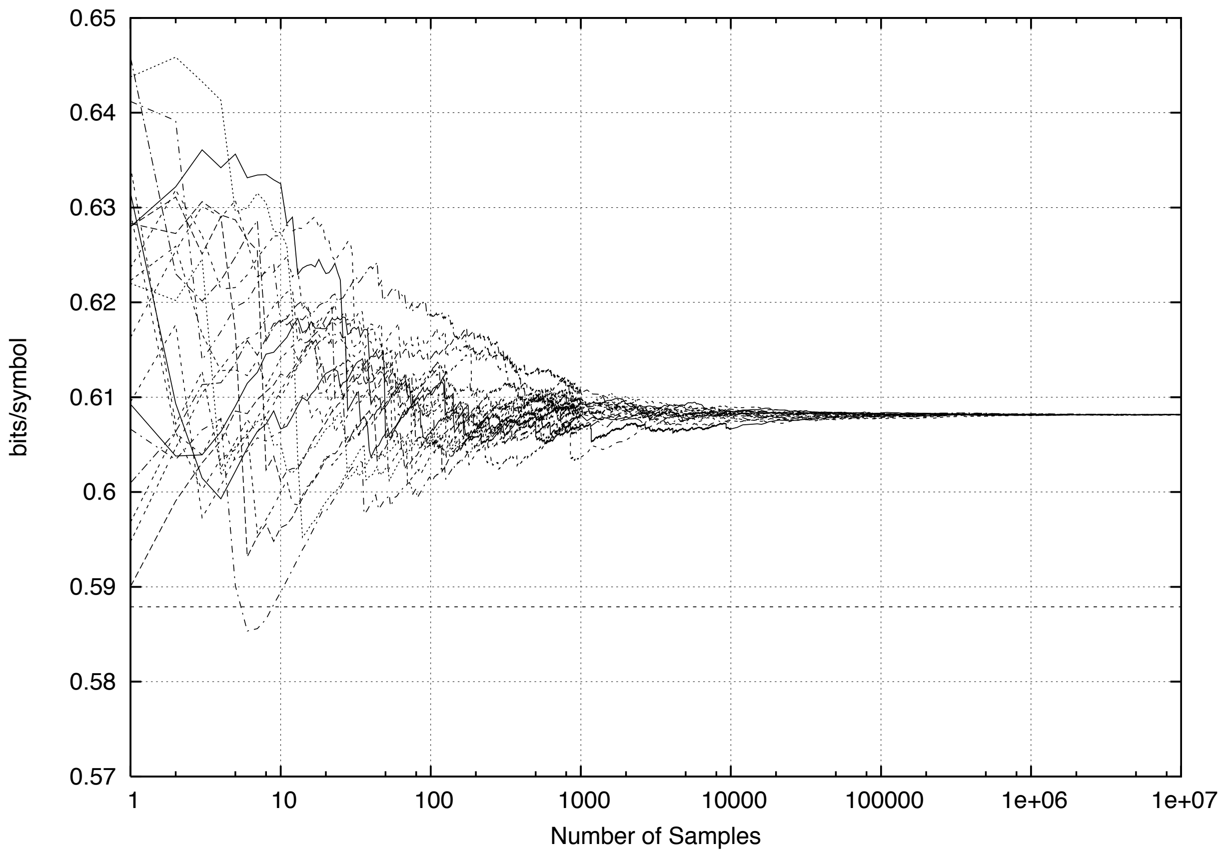
$$\hat{\Gamma}_A = \frac{1}{K|\mathcal{S}_{f_A}|}\sum_{k=1}^{K}\frac{1}{f_A(\mathbf{x}_A^{(k)})} \qquad \hat{\Gamma}_B = \frac{1}{K|\mathcal{S}_{f_B}|}\sum_{k=1}^{K}\frac{1}{f_B(\mathbf{x}_B^{(k)})}$$



Numerical Example:

Size $N = 10 \times 10$.

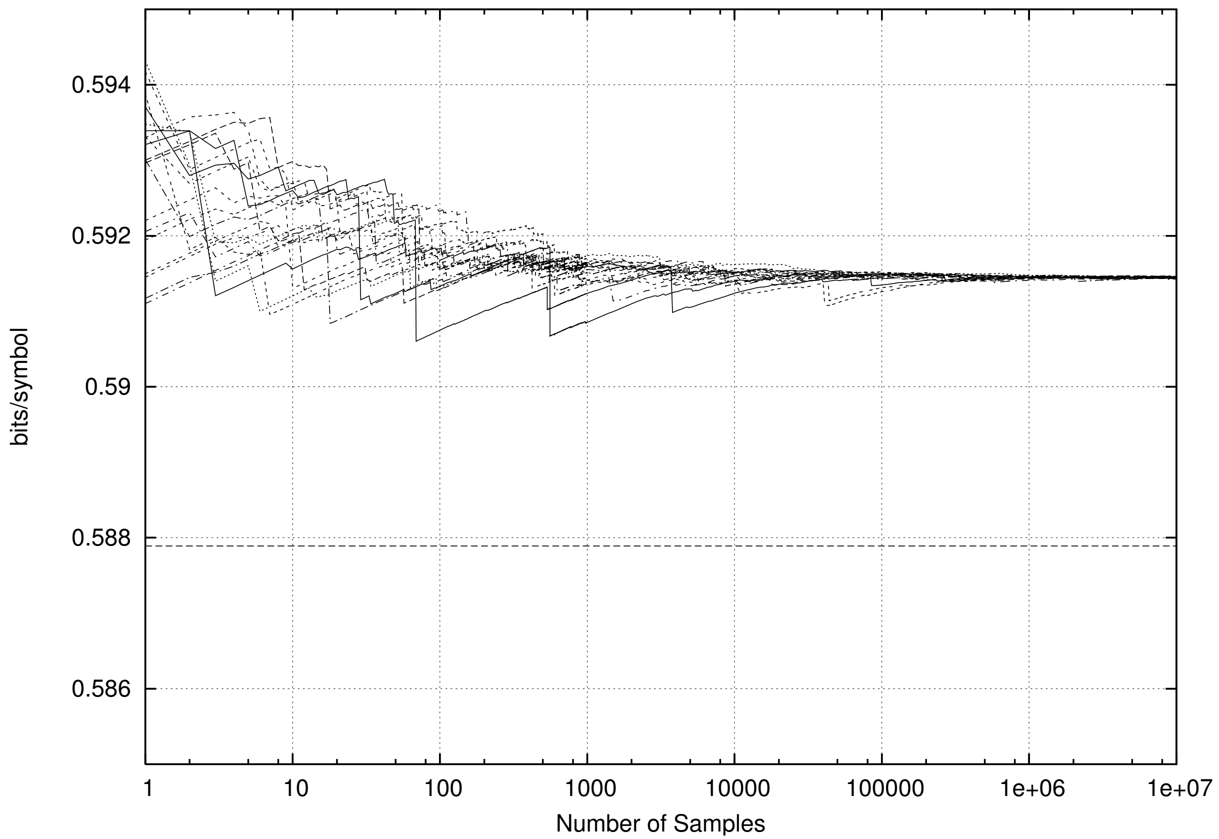
Estimated $\frac{1}{N} \log \hat{Z}_f$ vs. number of samples K



Numerical Example:

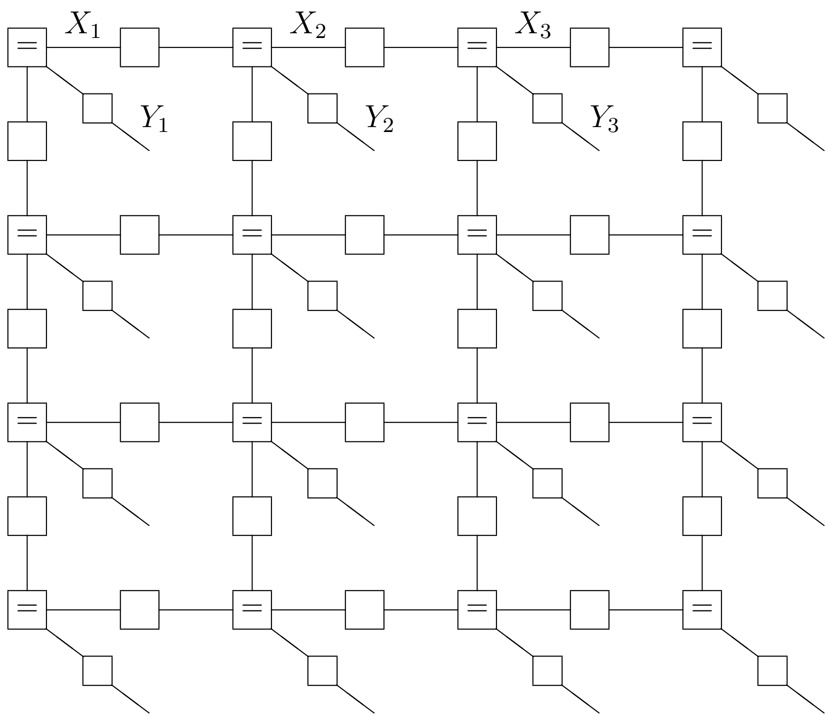
Size $N = 60 \times 60.$

Estimated $\frac{1}{N} \log \hat{Z}_f$ vs. number of samples K



Source/Channel Models with Cycles

- Channel output Y and channel input X
  with two-dimensional factor graph for $p(x)$ (up to a scale factor)
- Memoryless channel $p(y|x) = \prod_{k=1}^{N} p(y_k|x_k)$
- Goal: estimating $H(Y)$



Estimating the Mutual Information Rate $I(X;Y)$

The mutual information rate between the input process and the output process is

$$\frac{1}{N}I(X;Y) = \frac{1}{N}\big(H(Y) - H(Y|X)\big)$$

In many cases of interest, $H(Y|X)$ is analytically available eg when noise is additive white Gaussian (AWGN), independent of the input

$$H(Y|X) = \frac{N}{2}\log(2\pi e\sigma^2)$$

$\Longrightarrow$ By estimating $H(Y)$, we will have an estimate of $I(X;Y)$.

Source/Channel Models: $H(Y)$

$$H(Y) = -\,\mathrm{E}\left[\log p(Y)\right] \approx -\frac{1}{L}\sum_{\ell=1}^{L} \log p(y^{(\ell)})$$

Algorithm

1. Create samples $y^{(1)}, \ldots, y^{(L)}$ by
   a) Generating samples $x^{(1)}, \ldots, x^{(L)}$ by simulating the input.
   b) Generating $y^{(1)}, \ldots, y^{(L)}$ from $x^{(1)}, \ldots, x^{(L)}$ by channel simulation.
2. Estimate $p(y^{(\ell)})$ for $\ell = 1, 2, \ldots, L$.

- We can generate input samples at step (1.a) using MCMC.
- We concentrate on step (2): estimating

$$p(y^{(\ell)}) = \sum_{x \in \mathcal{X}^N} p(x, y^{(\ell)})$$

(Computing $p(y^{(\ell)})$ needs a sum with an exponential number of terms).

Estimating $p(y^{(\ell)})$

Clearly, $p(y^{(\ell)})$ is the partition function of $p(x, y^{(\ell)})$

$$p(y^{(\ell)}) = \sum_{x \in \mathcal{X}^N} p(x, y^{(\ell)})$$

We can estimate $p(y^{(\ell)})$ using Gibbs sampling.

We also have

$$\begin{aligned} p(y^{(\ell)}) &= \sum_{x \in \mathcal{X}^N} p(x) p(y^{(\ell)} | x) \\ &= \mathrm{E}\left[p(y^{(\ell)} | X)\right] \end{aligned}$$

But ...

Estimating $p(y^{(\ell)})$

Previous method has slow/erratic convergence at SNR $\gtrsim -4$ dB.

$$\text{SNR} \stackrel{\triangle}{=} 10 \log_{10}(\frac{1}{\sigma^2})$$

Analogy with statistical physics: $Z = \sum_s e^{-E(s)/k_B T}$

$$\begin{array}{ccc} \text{High temperature (easy)} & \Longleftrightarrow & \text{Low SNR} \\ \text{Low temperature (hard)} & \Longleftrightarrow & \text{High SNR} \end{array}$$

Let us define

$$f_\ell(x) \stackrel{\triangle}{=} p(x)\, p(y^{(\ell)}|x)$$

The desired quantity $p(y^{(\ell)})$ is Z_{f_ℓ}, the partition function of $f_\ell(x)$.

Estimating $p(y^{(\ell)})$

Importance sampling

1. Draw samples $x^{(1)}, x^{(2)}, \ldots, x^{(K)}$ from $\mathcal{X}^N$ according to some auxiliary probability distribution $q(x) = \frac{1}{Z_g} g(x)$,
2. Compute

$$\hat{R} = \frac{1}{K} \sum_{k=1}^{K} \frac{f(x^{(k)})}{g(x^{(k)})}$$

$\Longrightarrow \mathrm{E}(\hat{R}) = Z_f / Z_g$.

One (obvious) choice for $g(x)$ is

$$g(x) \triangleq f(x)^{\alpha}, \quad \text{for } 0 < \alpha < 1$$

With this choice, $g(x)$ and $f(x)$ have the same factor graph structure.

Estimating $p(y^{(\ell)})$

Use J parallel versions of importance sampling as
For $j = 0, 1, \ldots, J$ let

$$g_j(x) \triangleq f(x)^{\alpha_j}$$

with $0 < \alpha_J < \ldots < \alpha_1 < \alpha_0 = 1$.

Here $Z_{g_0} = Z_f$ and

$$\frac{Z_f}{Z_{g_J}} = \frac{Z_{g_0}}{Z_{g_1}} \frac{Z_{g_1}}{Z_{g_2}} \cdots \frac{Z_{g_{J-1}}}{Z_{g_J}}$$

Multilayer importance sampling

1. For $j = 1, 2 \ldots, J$ compute $Z_{g_{j-1}}/Z_{g_j}$ by importance sampling.
2. Use $\prod_{j=1}^{J} \hat{R}_j$ as an estimate of Z_f/Z_{g_J}, since $\mathrm{E}(\hat{R}_j) = Z_{g_{j-1}}/Z_{g_j}$

Estimating $p(y^{(\ell)})$: Remarks

Algorithm

1. For $j = 1, 2, \ldots, J$ compute $Z_{g_{j-1}}/Z_{g_j}$ by importance sampling.
2. Use $\prod_{j=1}^{J} \hat{R}_j$ as an estimate of Z_f/Z_{g_J}.

Estimating Z_{g_J} easier than $Z_f \Longrightarrow$ High temperature.

If J is large, $g_j(x)$ is a good approximation of $g_{j-1}(x)$, at each layer j.

Larger values of J are required for higher values of SNR.

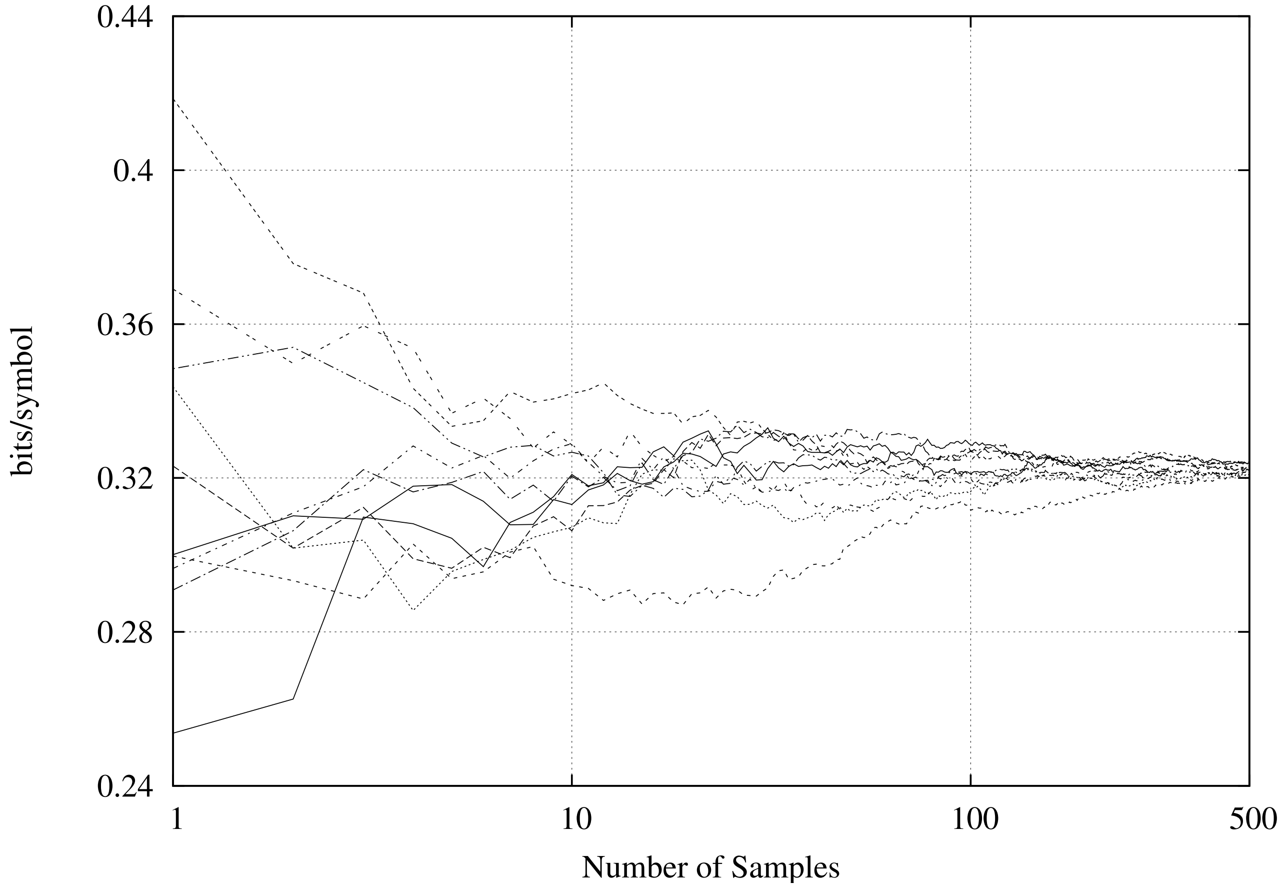
Some choices of $\{\alpha_0, \ldots, \alpha_J\}$ might result in faster convergence.

Similar ideas: Annealed importance sampling [Neal98], Equilibrium free energy differences [Jarzynski97].

Numerical Example: $I(X;Y)$ at zero dB

Channel size $N = 24 \times 24$.
AWGN channel, $p(x)$ uniform over valid configurations, and $J = 4$.

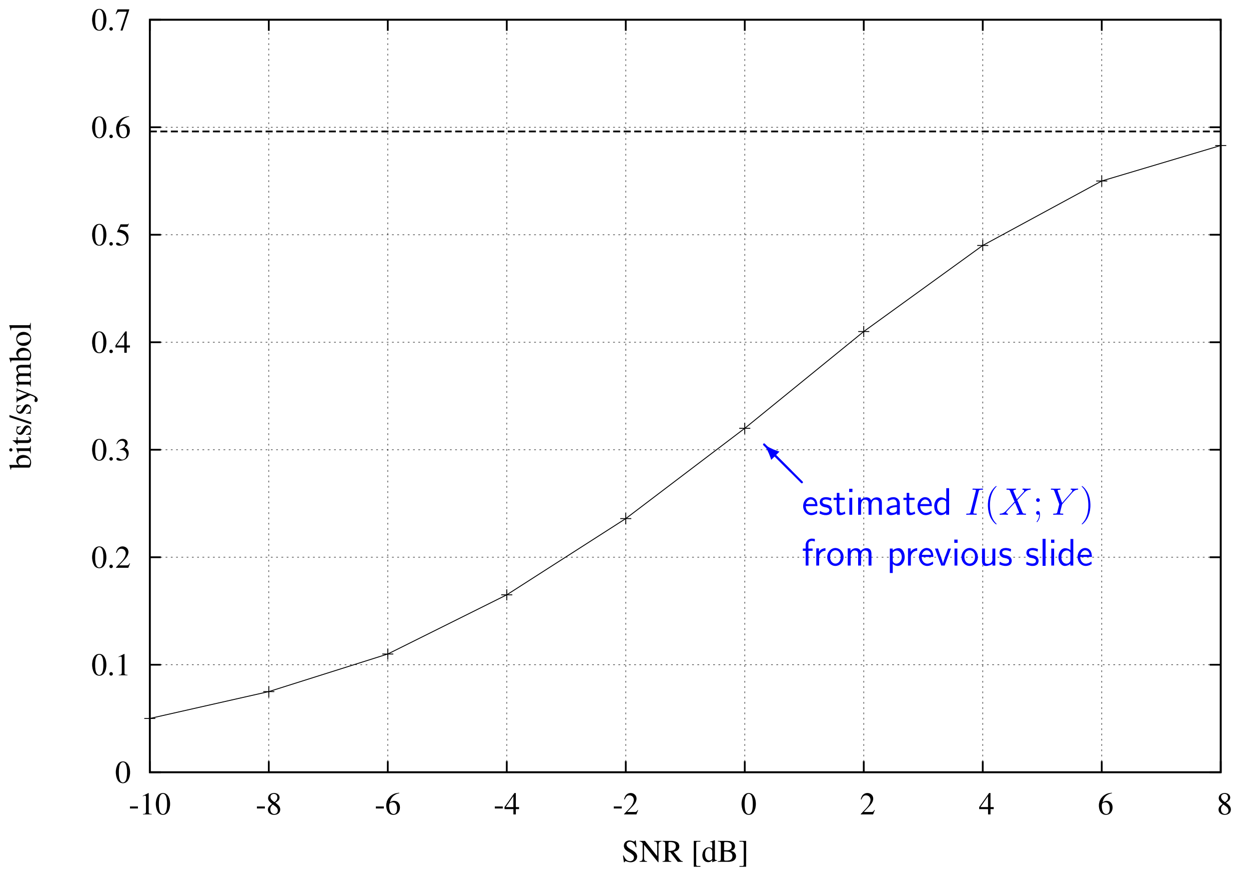
Estimated information rate at zero dB vs. number of samples L.



Numerical Example: $I(X;Y)$ vs. SNR

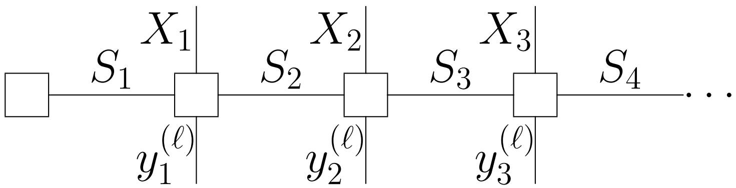
Channel size $N = 24 \times 24$.
AWGN channel, $p(x)$ uniform over valid configurations.

Estimated i.u.d. information rate vs. SNR



Concluding Remarks

We proposed a sampling-based method to estimate the entropy of the input/output process of source/channel models, (in particular) information rates of 2D source/channel models:



$$p(x, y, s) = p(s_1) \prod_{k=1}^{N} p(x_k, y_k, s_{k+1} | s_k)$$

Shannon-McMillan Theorem:
For a finite-valued ergodic process $\{X_N\}$

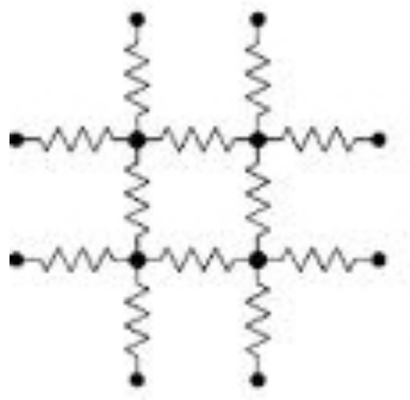
$$-\frac{1}{N} \log p(X_1, X_2, \ldots, X_N) \to H \quad \text{with probability 1}$$

Papers available online: http://people.ee.ethz.ch/~loeliger

Thank You!

Other Constraints

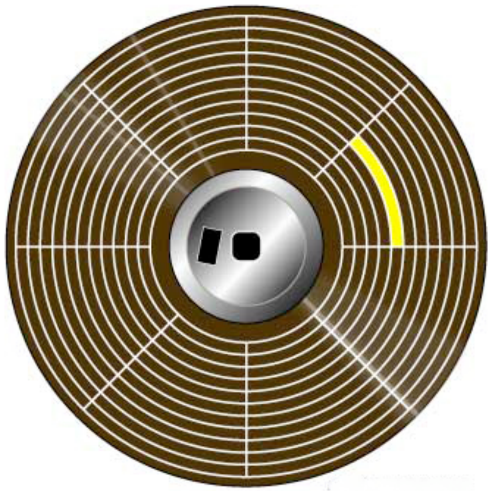
- DC-free (Spectral-Null Constraints)
  Bipolar $\{-1, +1\}$ alphabet, number of $+1$'s and -1's are equal.
- No Isolated Bit. Bits agree with at least one of their neighbors.
- Channels with prescribed number of 1 and 0. Number of 1's in each row/column is at most $n/2$.



(Memory coding for limiting current).

RLL Constraints Applications

Track-oriented magnetic recording (1D): in DVDs, hard disks, to reduce interference, improve synchronization, time-control, etc.



Page-oriented magnetic recording (2D): in holograhic memory, to increase capacity per surface.

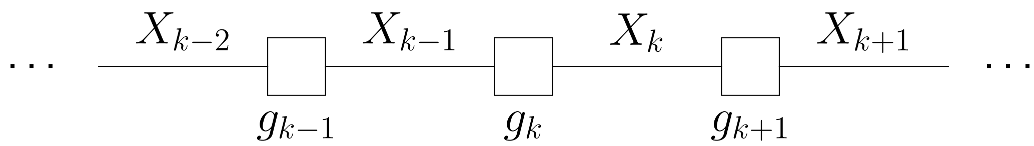
Noiseless Constrained 1D Channels

Consider a 1D $(1, \infty)$-RLL constraint

$$f(x_1, \ldots, x_N) = \prod_{k=2}^{N} g_k(x_{k-1}, x_k)$$

$$Z_f = \sum_{x \in \mathcal{X}} f(x) = \sum_{x \in \mathcal{X}} \prod_{k=2}^{N} g_k(x_{k-1}, x_k)$$

Computing Z_f is straightforward



with sum-product message passing on a cycle-free factor graph.

Other approaches: combinatorial and algebraic [Shannon48].

Capacity of 1D $(1, \infty)$-RLL

1-D $(1, \infty)$-RLL means adjacent bits can not both have value 1.

- $N = 1$, $Z = 2$, valid sequences: 0, 1
- $N = 2$, $Z = 3$, valid sequences: 00, 10, 01, not: 11
- $N = 3$, $Z = 5$, valid sequences: 000, 100, 010, 001, 101.
- Valid sequences of length N: $0\underbrace{\boxed{}}_{N-1}$ or $10\underbrace{\boxed{}}_{N-2}$

$$Z(N) = Z(N-1) + Z(N-2)$$

Easy to prove

$$C_\infty^{(1,\infty)} = \lim_{N\to\infty} \frac{\log_2 Z(N)}{N} = \log_2 \frac{1+\sqrt{5}}{2} \approx 0.6942 \text{ bits}$$

In statistical physics: transfer matrix method.

1D Numerical Approach

- By increasing the size of the factor graph

N	$Z(N)$	$\frac{1}{N}\log_2 Z(N)$
1	2	1
2	3	0.79
3	5	0.77
4	8	0.75
5	13	0.74
10	144	0.72
100	9×10^{20}	0.70
400	5×10^{83}	0.69

We know

$$C_{1D}^{(1,\infty)} = 0.6942 \text{ bits}$$

Bounds for Noiseless Constrained 2D Channels

In 2D, C_∞ is known (tightly bounded) only for a few special cases:

- For 2D $(1,\infty)$-RLL, [CW98]

$$0.587891... \leq C_\infty \leq 0.587891...$$

- For 2D (d,k)-RLL, [KZ00]

$$C_\infty = 0 \quad \Leftrightarrow \quad k = d+1$$

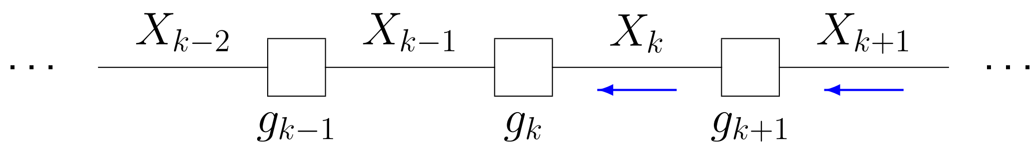
We propose a general method based on Gibbs sampling to compute a Monte Carlo estimate of the capacity of noiseless 2D RLL constraints.

Sampling from Cycle-Free Factor Graphs

(demonstrated for Markov chains)

Sampling from $\quad p(x_1, \ldots, x_n) = p(x_1) \prod_{k=2}^{n} p(x_k | x_{k-1}) \quad$ is straightforward.

What if $\quad p(x_1, \ldots, x_n) \propto \prod_{k=2}^{n} g_k(x_{k-1}, x_k) \;$?



Reparameterize using $\quad p(x_k | x_{x-1}) = \dfrac{g_k(x_{k-1}, x_k) \overleftarrow{\mu}_{X_k}(x_k)}{\overleftarrow{\mu}_{X_{k-1}}(x_{k-1})}$

with sum-product messages $\overleftarrow{\mu}$.

$\Longrightarrow$ "backward filtering forward sampling" (or the other way round)

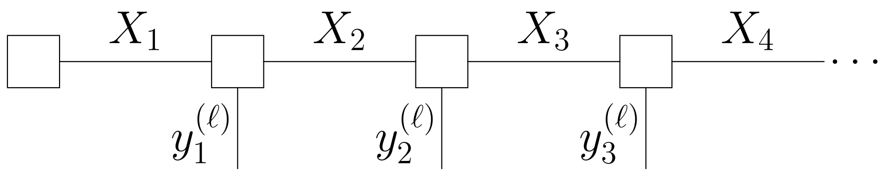
Estimating $H(Y)$

In the following, we consider

- Source/Channel models with the input process X and the output process Y.

We are primarily interested in

- Estimating $H(Y)$ in source/channel models where Y is a noisy observation of X.



Source/Channel Models: $H(Y)$

Suppose the input process of the source/channel model is X and the output process is Y.

We want to compute

$$H(Y) = -\,\mathrm{E}\left[\log p(Y)\right] \approx -\frac{1}{L}\sum_{\ell=1}^{L} \log p(y^{(\ell)})$$

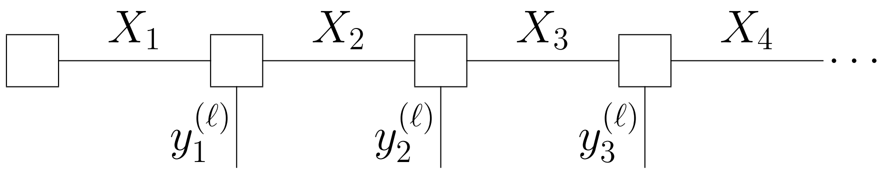
for samples $y^{(1)}, y^{(2)}, \ldots, y^{(L)}$ from $p(y)$.

Algorithm

1. Create samples $y^{(1)}, \ldots, y^{(L)}$ by
   a) Generating samples $x^{(1)}, \ldots, x^{(L)}$ by simulating the input.
   b) Generating $y^{(1)}, \ldots, y^{(L)}$ from $x^{(1)}, \ldots, x^{(L)}$ by channel simulation.
2. Estimate $p(y^{(\ell)})$ for $\ell = 1, 2, \ldots, L$.

Cycle-Free Source/Channel Models: $p(y)$

-Hidden Markov models



In this case

$$\begin{aligned} p(x,y) &= p(x_1)\prod_{k=1}^{N} p(x_{k+1}, y_k | x_k) \\ p(y^{(\ell)}) &= \sum_{x \in \mathcal{X}^N} p(x, y^{(\ell)}) \end{aligned}$$

-Memoryless source/channel models

$$\begin{aligned} p(x,y) &= p(x)\prod_{k=1}^{N} p(y_k | x_k) \\ p(y^{(\ell)}) &= \sum_{x \in \mathcal{X}^N} p(x, y^{(\ell)}) \end{aligned}$$