Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Monte Carlo Algorithms for Two-Dimensional Channels

Mehdi Molkaraie and Hans-Andrea Loeliger

Outline

Part I

- Monte Carlo methods to compute the capacity of noiseless constrained 2D channels.
- Tree-Based Gibbs sampling.

Part II

- Extensions to compute information rates of noisy constrained 2D source/channel models.
- Multilayer importance sampling.

Both problems reduce to computing the partition function of graphical models with cycles

The Partition Function

Problem setting:

- finite sets $\mathcal{X}_1, \ldots, \mathcal{X}_N$ and $\mathcal{X} \triangleq \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_N$
- function $f : \mathcal{X} \to \mathbb{R}$ with $f(x) \geq 0$ for all $x \in \mathcal{X}$

Compute the partition function

$$Z_f \triangleq \sum_{x \in \mathcal{X}} f(x),$$

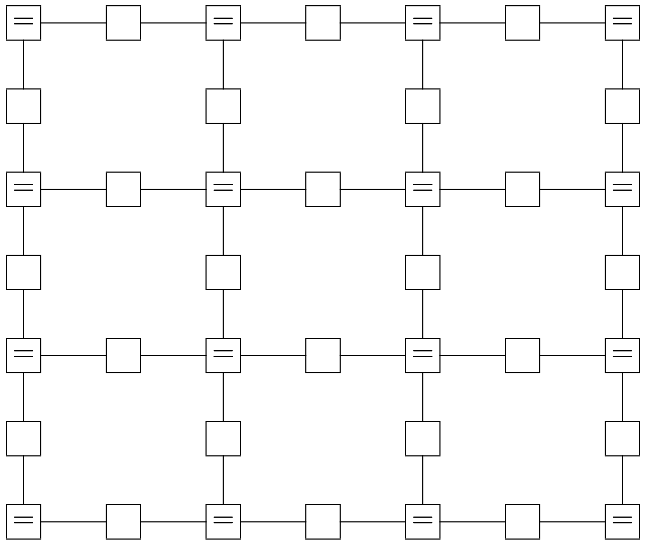
where

- $\mathcal{X}_1, \ldots, \mathcal{X}_N$ are "small" sets (e.g., $|\mathcal{X}_1| = \ldots = |\mathcal{X}_N| = 2$)
- N is large
- f has a "useful" factorization (factor graph) - but not cycle-free.

Also define

$$\mathcal{S}_f \triangleq \{x \in \mathcal{X} : f(x) > 0\}.$$

Part I: Noiseless Constrained 2D Channels



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

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

Also known as 2D $(1, \infty)$-RLL channel.

Noiseless Constrained 2D Channels

In this case

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

$$C_N \stackrel{\triangle}{=} \frac{1}{N} \log Z_f = \text{noiseless capacity} \quad \text{(for } N \to \infty \text{ called } C_\infty \text{ the Shannon capacity)}$$

For a 2D $(1, \infty)$-RLL [CW98,NZ00]

$$C_\infty \approx 0.587891\ldots$$

Tight bounds for C_∞ are available for a few special cases, while our method works for various generalizations of this example.

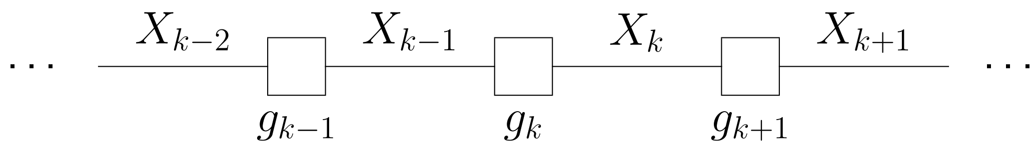
Noiseless Constrained 1D Channels

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].

Estimating $1/Z_f$ (Ogata-Tanemura)

Algorithm:

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

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

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

Issues:

1. How draw samples? Gibbs sampling: highly dependent samples, prone to slow mixing.
2. Not applicable to previous example since $Z_f = |\mathcal{S}_f|$.

Estimating $1/Z_f$ (Ogata-Tanemura)

Algorithm:

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

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

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

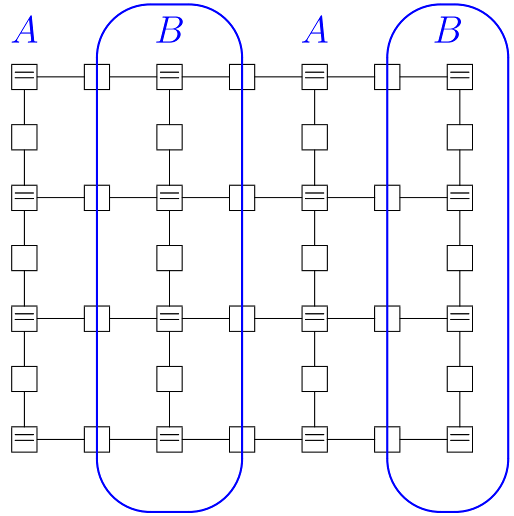
1. How draw samples? Gibbs sampling: highly dependent samples, prone to slow mixing.
2. Not applicable to previous example since $Z_f = |\mathcal{S}_f|$.

We will address both issues by tree-based Gibbs sampling and tree-based estimation of $1/Z_f$.

Tree-Based Gibbs Sampling

(Hamze & de Freitas, 2004)

Partition the index set $\{1, \ldots, N\}$ into two parts (A, B) such that fixing either x_A or x_B breaks all cycles in the remaining factor graph.



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 $p(x_B | x_A = x_A^{(k)}) \propto f(x_A^{(k)}, x_B)$

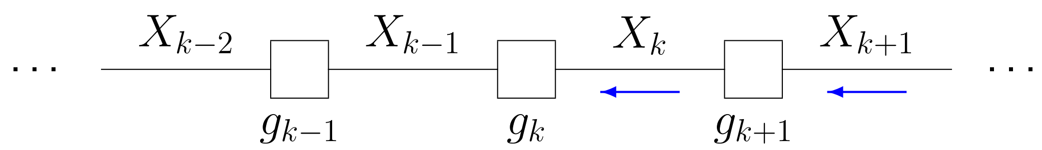
Much faster mixing than naive Gibbs sampling.

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 $\overleftarrow{\mu}$.

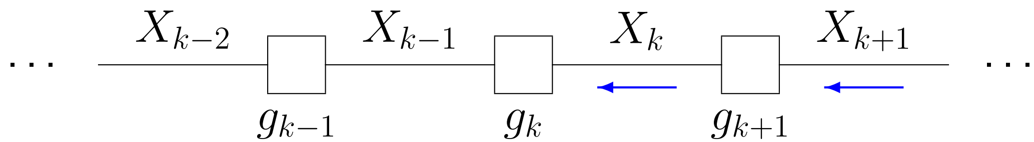
$\Longrightarrow$ "backward filtering forward sampling" (or the other way round)

Sampling from Cycle-Free Factor Graphs

(demonstrated for Markov chains)

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

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



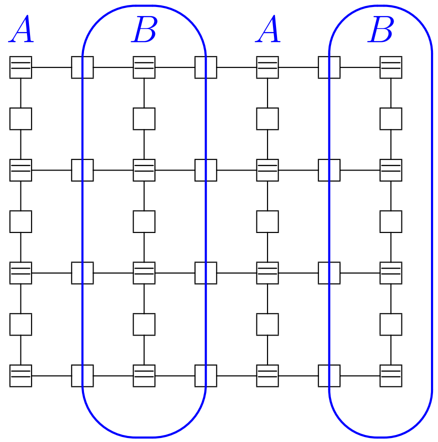
Reparameterize using $p(x_k | x_{x-1}) = \dfrac{g_k(x_{k-1}, x_k) \overleftarrow{\mu}_{X_k}(x_k)}{\overleftarrow{\mu}_{X_{k-1}}(x_{k-1})}$

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

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

Yields $Z_g = \sum_{x_1, \ldots, x_n} \prod_{k=2}^{n} g_k(x_{k-1}, x_k) = \sum_{x_1} \overleftarrow{\mu}_{X_1}(x_1)$ as a byproduct.

Tree-Based Estimation of $1/Z_f$ (ISIT 2008)



Suppose

$$f_A(x_A) \stackrel{\triangle}{=} \sum_{x_B} f(x_A, x_B).$$

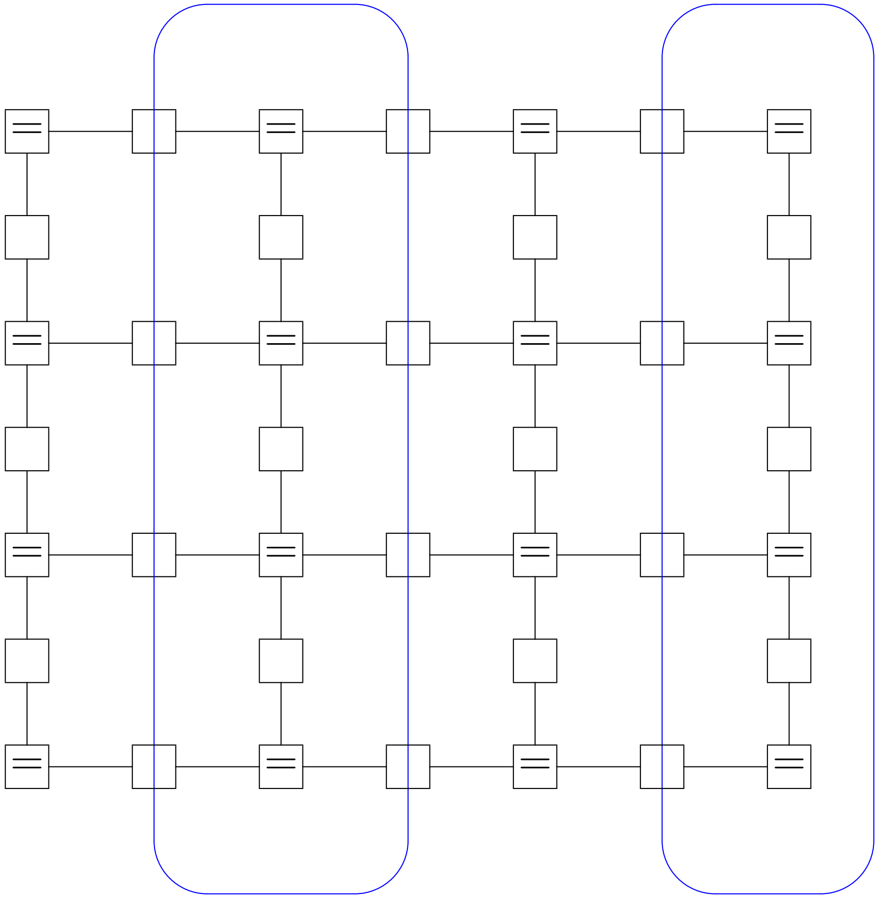
Therefore $$Z_{f_A} = \textstyle\sum_{x_A} f_A(x_A) = Z_f.$$

$\Longrightarrow$ Can modify the "first method" to estimate $1/Z_f = 1/Z_{f_A}$ by:

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

Get $f_A(x_A^{(k)})$ and $|\mathcal{S}_{f_A}|$ as byproducts of tree-based Gibbs sampling.

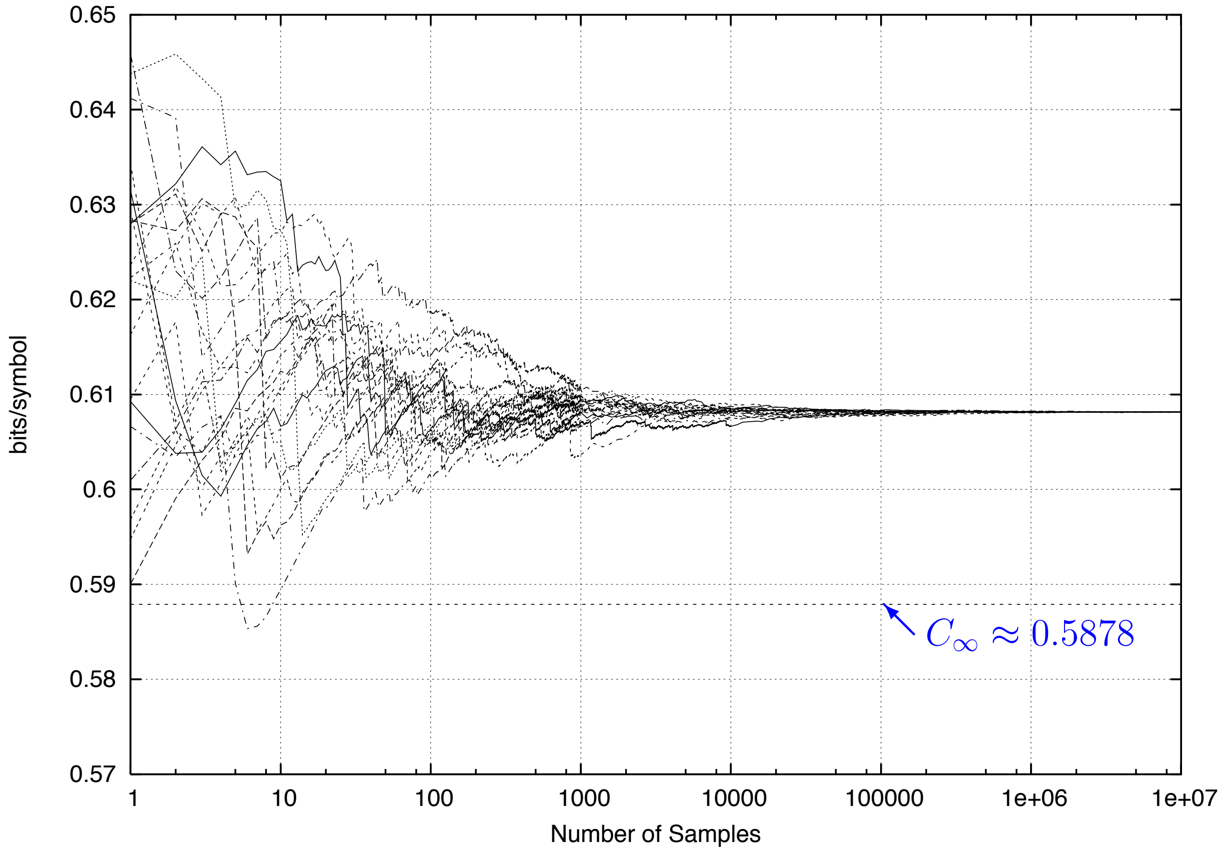
$$\Gamma_A = \frac{1}{K|\mathcal{S}_{f_A}|}\sum_{k=1}^{K}\frac{1}{f_A(\mathbf{x}_A^{(k)})} \qquad \Gamma_B = \frac{1}{K|\mathcal{S}_{f_B}|}\sum_{k=1}^{K}\frac{1}{f_B(\mathbf{x}_B^{(k)})}$$



Numerical Example: (ITW 2009)

2D $(1,\infty)$-RLL constraint, $N = 10 \times 10$.

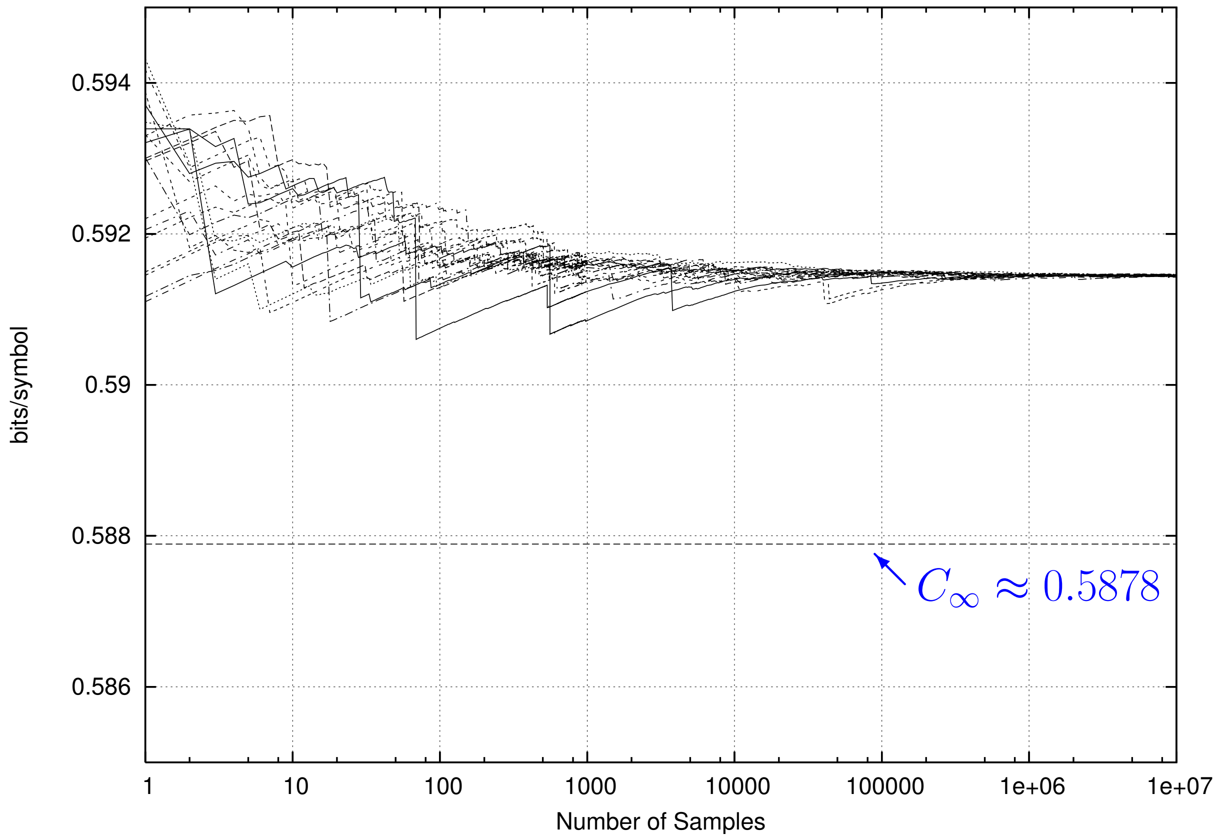
Estimated noiseless capacity $\frac{1}{N}\log \hat{Z}_f$ vs. number of samples K



Numerical Example: (ITW 2009)

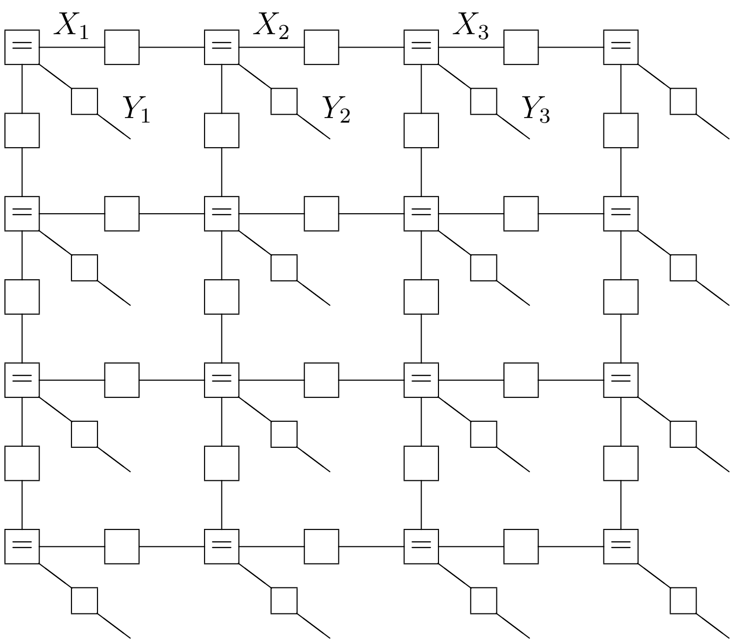
2D $(1, \infty)$-RLL constraint, $N = 60 \times 60$.

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



Part II: Extension to Information Rate of Noisy Constrained 2-D Channels

- Constrained channel input $X_1, \ldots, X_N$ with 2-D factor graph for $p(x_1, \ldots, x_N)$ (up to a scale factor)
- Memoryless channel $p(y|x) = \prod_{k=1}^{N} p(y_k|x_k)$



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

Estimating $I(X;Y)$

Want to estimate

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

Suppose $H(Y|X)$ is analytically available, for example, if the noise is additive white Gaussian (AWGN) independent of the input

$$\frac{1}{N} H(Y|X) = \frac{1}{2}\log(2\pi e \sigma^2)$$

We will focus on estimating $H(Y)$ (next slides).

Estimating $H(Y)$

by a double-loop algorithm.

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

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

Issues:

1. How to generate samples $y^{(1)}, \ldots, y^{(L)}$?
   - Generate samples $x^{(1)}, \ldots, x^{(L)}$ from $p(x)$ by tree-based Gibbs sampling.
   - Generate $y^{(1)}, \ldots, y^{(L)}$ from $x^{(1)}, \ldots, x^{(L)}$ by channel simulation.
2. Remaining problem: how to estimate $p(y^{(\ell)})$?
   $\Longrightarrow$ inner loop (next slides).

Estimating $p(y^{(\ell)})$: Method 1

Clearly, the partition function of $p_{X,Y}(x, y^{(\ell)})$ (as a function of x) is $p(y^{(\ell)})$

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

$\Longrightarrow$ Can estimate $p(y^{(\ell)})$ by
tree-based Gibbs sampling on $p_{X,Y}(x, y^{(\ell)})$.

Estimating $p(y^{(\ell)})$: Method 1

Clearly, the partition function of $p_{X,Y}(x, y^{(\ell)})$ (as a function of x) is $p(y^{(\ell)})$

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

$\Longrightarrow$ Can estimate $p(y^{(\ell)})$ by
tree-based Gibbs sampling on $p_{X,Y}(x, y^{(\ell)})$.

Convergence too slow/erratic at SNR $\gtrsim$-4 dB, (SNR $\triangleq 10 \log_{10} \frac{1}{\sigma^2}$)

Estimating $p(y^{(\ell)})$: Method 2

Importance sampling

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

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

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

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

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

With this choice, the structure of the factor graph is preserved $\Rightarrow$ Can sample from $q(x)$ with tree-based Gibbs sampling.

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

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

$$g_j(x) \stackrel{\triangle}{=} f(x)^{\alpha_j}$$

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

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

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}.

Estimating Z_{g_J} easier than Z_f $\Rightarrow$ tree-based Ogata-Tanemura.
In particular, we have $Z_{g_J} = |S_f|$ if $\alpha_J = 0$.

In our numerical experiments

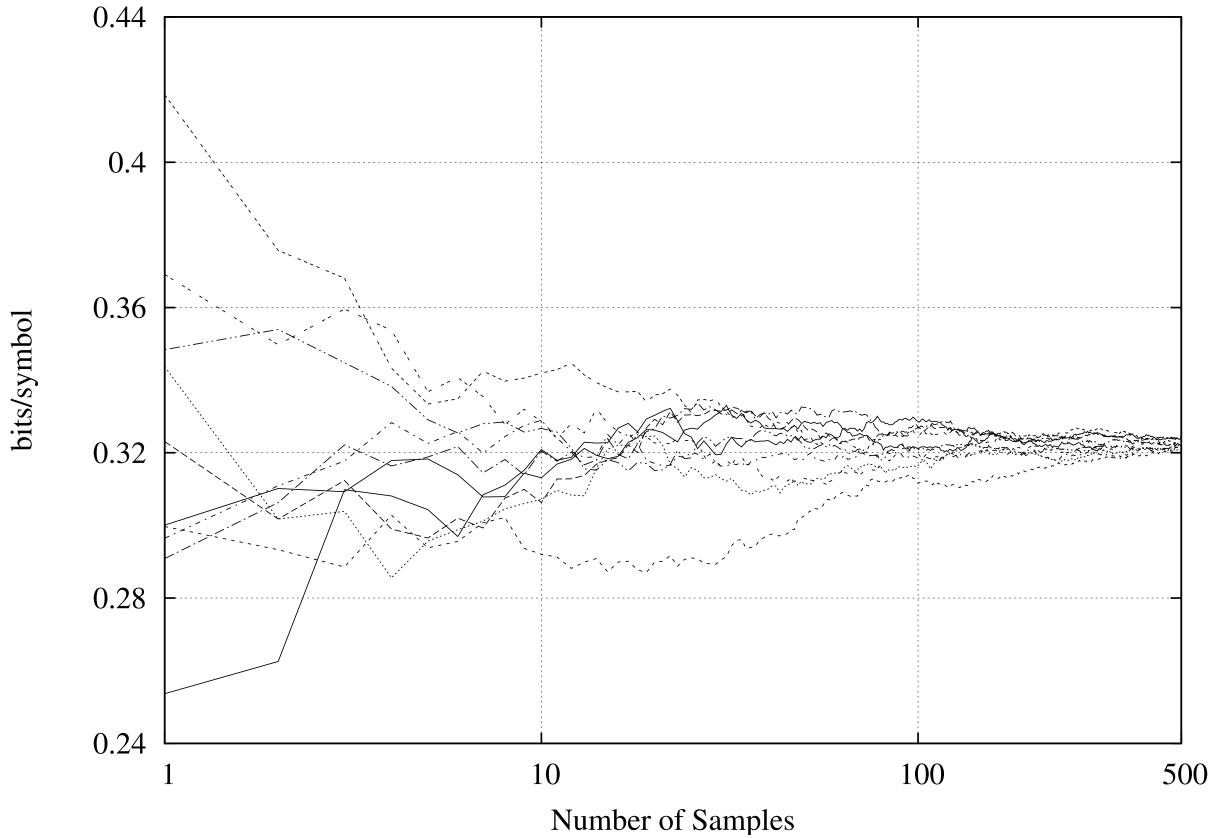
$$f_\ell(x) \triangleq p(x) p_{Y|X}(y^{(\ell)}|x)$$

And Z_{f_ℓ} is the desired quantity.

Numerical Example:

Noisy 2D $(1, \infty)$-RLL constraint, $N = 24 \times 24$.
AWGN channel, $p(x)$ uniform over valid configurations, and $J = 4$.

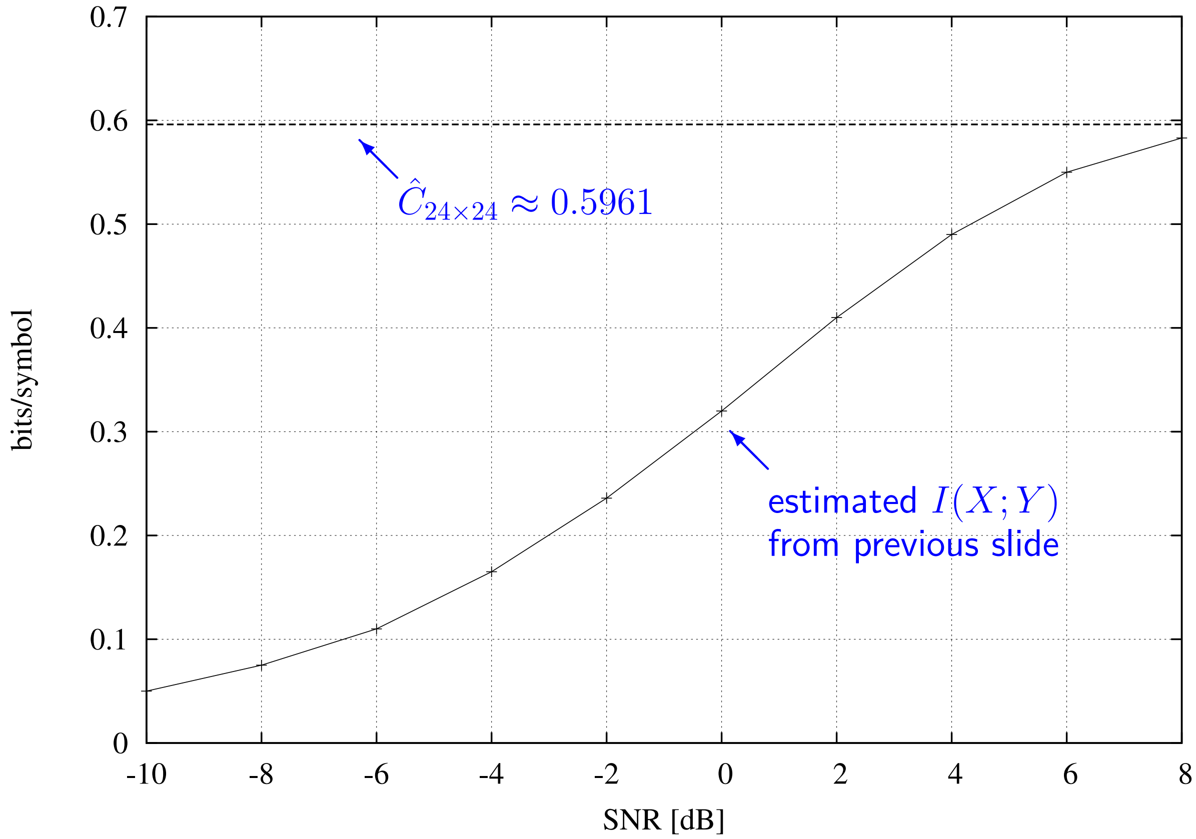
Estimated information rate at zero dB vs. number of samples L.



Numerical Example:

Noisy 2D $(1, \infty)$-RLL constraint, $N = 24 \times 24$.
AWGN channel, $p(x)$ uniform over valid configurations.

Estimated information rate vs. SNR



Conclusion

We proposed new sampling-based methods to estimate

- the partition function (normalization constant) of unnormalized 2D probability distributions and
- the information rates of 2D source/channel models

The methods can handle other 2D constraints and other noisy 2D channels, like ISI channels.

The proposed methods are guaranteed to asymptotically converge to the desired quantity, in contrast to approximate GBP-based methods [SSKWW08, SM10].

Applications: 2D storage such as holographic data storage.

Thank You!