Randomized Algorithms for Rare Events, Combinatorial Optimization and Counting

Technion, 2008

Reuven Rubinstein

Faculty of Industrial Engineering and Management,
Technion, Israel

Contents

1. Introduction
2. CE for Rare Events and Combinatorial Optimization
3. MinxEnt for Rare Events and Combinatorial Optimization
4. The Cloning Method for Rare Events, Combinatorial Optimization and Counting
5. The Gibbs Sampler
6. Integer Programs, Multiple Events and the Satisfiability problem
7. Why and when it Works
8. Generating Points Uniformly on Different Bodies
9. Convergence and Numerical Results.
10. Open Problems.

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering

Applications

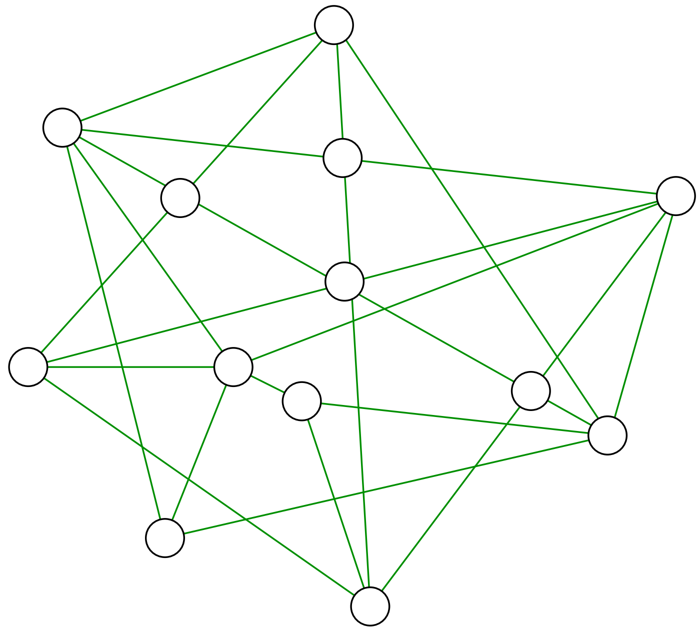
- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering
- Multi-extremal Continuous Optimization

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering
- Multi-extremal Continuous Optimization
- NP- hard Counting problems: Hamiltonian Cycles, SAW's, calculation the Permanent, Satisfiability Problem, etc.

Combinatorial Optimization: A Coloring Problem

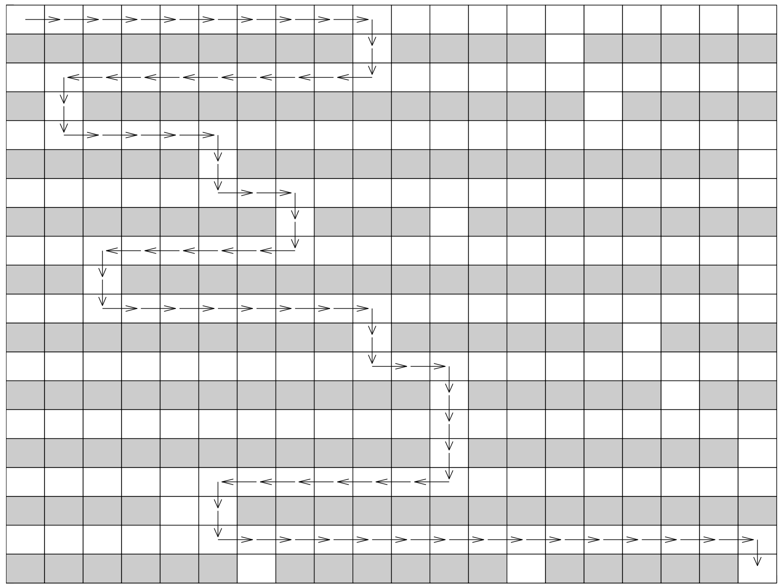
We wish to color the nodes white and black.



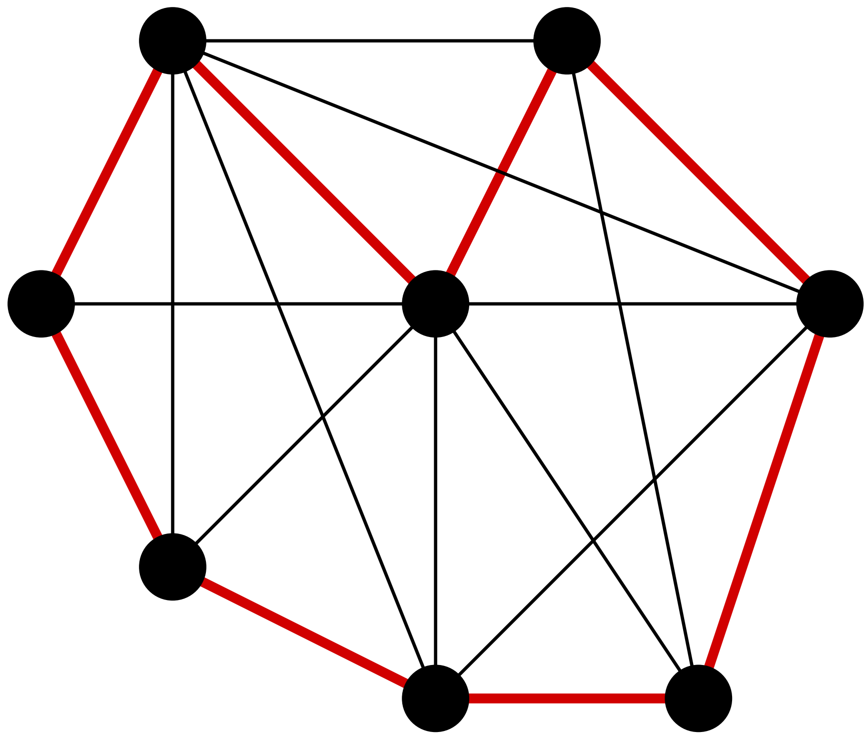
How should we color so that the total number of links between the two groups is maximized? This problem is known as *Maximal Cut* problem.

A Maze Problem

The Optimal Trajectory

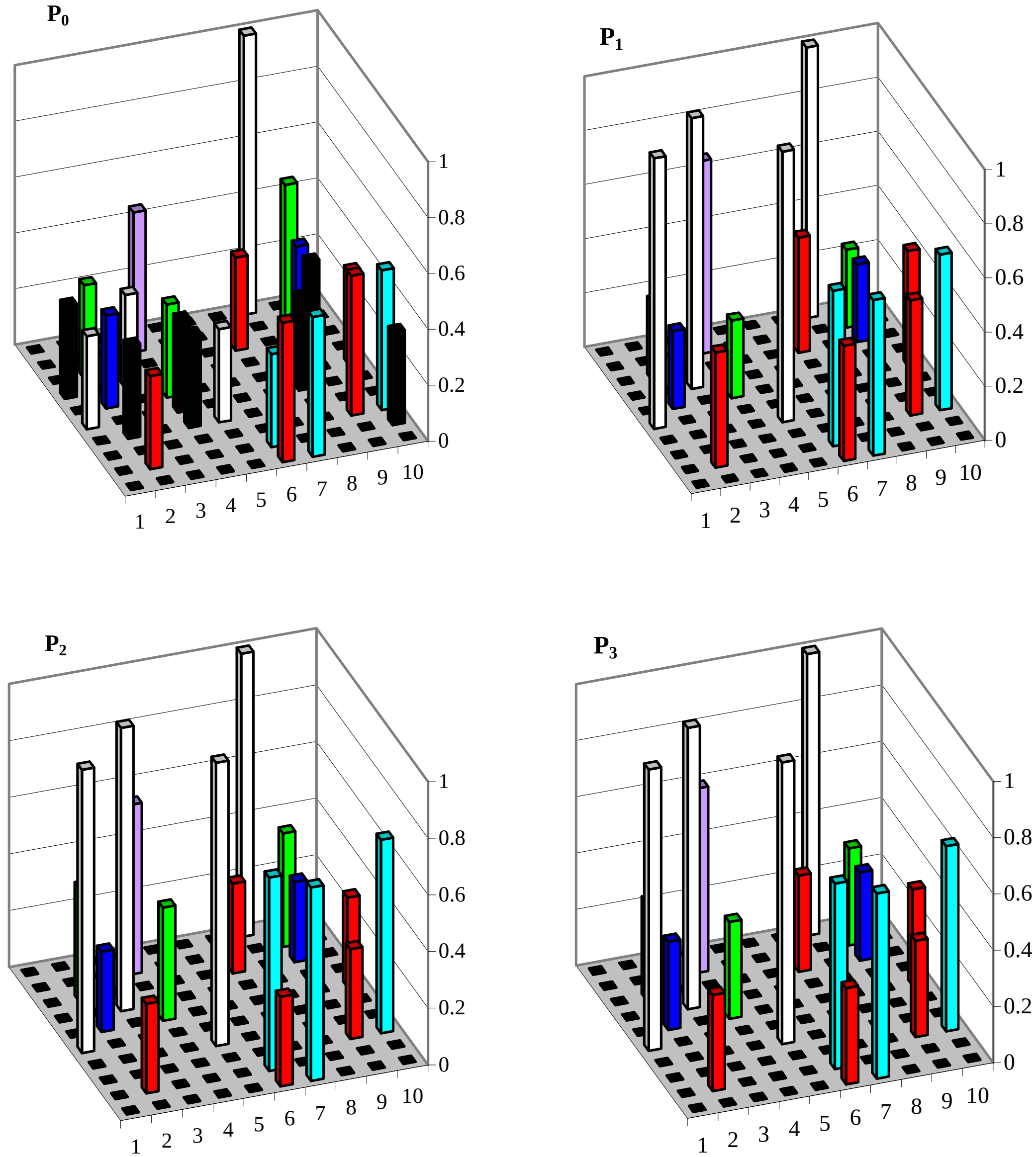


Counting Hamiltonian Cycles



How many Hamiltonian cycles does this graph have?

Calculating the Number of HC's



General Procedure

We cast cast the original optimization problem of $S(\boldsymbol{x})$ and counting into an associated rare-events probability estimation problem, that estimation of

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq m) = \mathbb{E}\left[I_{\{S(\boldsymbol{X}) \geq m\}}\right] \ .$$

and involves the following iterative steps:

- Formulate a random mechanism to *generate* the objects $\boldsymbol{x} \in \mathcal{X}$.
- Give the *updating formulas* (parametric or non parametric), in order to produce a better sample in the next iteration.

Generating Tuples

In our randomized algorithms we shall generate either an adaptive parametric sequence of tpuples

$$\{(m_0, \boldsymbol{v}_0),\ (m_1, \boldsymbol{v}_1), \ldots, (m_T, \boldsymbol{v}_T)\}$$

or non-parametric one

$$\{(m_0, f(\boldsymbol{x}, \boldsymbol{v}_0)),\ (m_1, g^*(\boldsymbol{x}, m_0)), \ldots, (m_T, g^*(\boldsymbol{x}, m_{T-1}))\}.$$

A Randomized Algorithm for Optimization

1 **Starting:** **Start with the proposal pdf, like** $f(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{p})$**. Set** $t := 1$**.**

A Randomized Algorithm for Optimization

1 **Starting:** **Start with the proposal pdf, like** $f(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{p})$. **Set** $t := 1$.

2 **Update** $\hat{m}_t$**:** **Draw** $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ **from parametric** $f(\boldsymbol{x}, \widehat{\boldsymbol{p}}_t)$ **or non-parametric pdf** $g_t = g(\boldsymbol{x}, \widehat{m}_t)$. **Find the elite sampling based on** $\hat{m}_t$, **which is the worst performance of the** $\rho \times 100\%$ **best performances.**

A Randomized Algorithm for Optimization

1 **Starting:** **Start with the proposal pdf, like** $f(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{p})$**. Set** $t := 1$**.**

2 **Update** $\hat{m}_t$**: Draw** $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ **from parametric** $f(\boldsymbol{x}, \widehat{\boldsymbol{p}}_t)$ **or non-parametric pdf** $g_t = g(\boldsymbol{x}, \widehat{m}_t)$**. Find the elite sampling based on** $\hat{m}_t$**, which is the worst performance of the** $\rho \times 100\%$ **best performances.**

3 **Update** $\hat{\boldsymbol{p}}_t$ **or** $g_t = g(\boldsymbol{x}, \widehat{m}_t$**:. For a parametric method update the parameter** $\hat{\boldsymbol{p}}_t$ **and for a non-parametric one update the pdf** $g_t = g(\boldsymbol{x}, \widehat{m}_t)$ **and increase** t **by 1.**

A Randomized Algorithm for Optimization

1. **Starting:** **Start with the proposal pdf, like** $f(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{p})$**. Set** $t := 1$**.**
2. **Update $\hat{m}_t$:** **Draw** $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ **from parametric** $f(\boldsymbol{x}, \widehat{\boldsymbol{p}}_t)$ **or non-parametric pdf** $g_t = g(\boldsymbol{x}, \widehat{m}_t)$**. Find the elite sampling based on** $\hat{m}_t$**, which is the worst performance of the** $\rho \times 100\%$ **best performances.**
3. **Update $\hat{\boldsymbol{p}}_t$ or $g_t = g(\boldsymbol{x}, \widehat{m}_t$:.** **For a parametric method update the parameter** $\hat{\boldsymbol{p}}_t$ **and for a non-parametric one update the pdf** $g_t = g(\boldsymbol{x}, \widehat{m}_t)$ **and increase** t **by 1.**
4. **Stopping:** **If the stopping criterion is met, then stop; otherwise set** $t := t + 1$ **and reiterate from step 2.**

Non-Parametric MinxEnt

$$\min_g \left\{ \mathcal{D}(g|h) = \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})} g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_g \ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right\}$$

$$(\mathrm{P}_0) \qquad \text{s.t.} \quad \int S_j(\boldsymbol{x}) g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_g S_j(\boldsymbol{X}) = b_j, \; j = 1, \ldots, k,$$

$$\int g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1. \tag{1}$$

Here g and h are joint n-dimensional pdf's or n-dimensional pmf's, $S_j(\boldsymbol{x}), \; j = 1, \ldots, k$, are known functions of an n-dimensional vector $\boldsymbol{x}$ and h is a known pdf, called the prior pdf.

Single Constraint MinxEnt Program

When we have only a single constraint

$$\mathbb{E}_g S(\boldsymbol{X}) = b, \ (\int g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1)$$

the solution of the program (P_0) is

$$g(\boldsymbol{x}) = \frac{h(\boldsymbol{x}) \exp\{-S(\boldsymbol{x})\lambda\}}{\mathbb{E}_h \exp\{-S(\boldsymbol{X})\lambda\}}$$

and

$$\frac{\mathbb{E}_h S(\boldsymbol{X}) \exp\{-\lambda S(\boldsymbol{X})\}}{\mathbb{E}_h \exp\{-\lambda S(\boldsymbol{X})\}} = b,$$

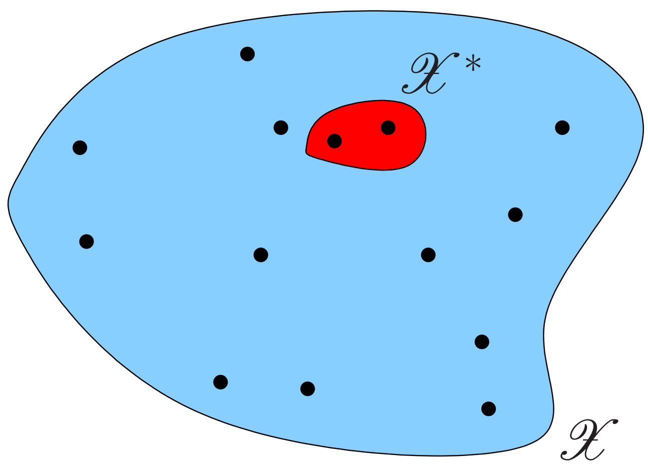
respectively.

Counting via Monte Carlo

We start with the following basic

Example.

Assume we want to calculate an area of same “irregular" region $\mathcal{X}^*$. The Monte-Carlo method suggests inserting the ”irregular" region $\mathcal{X}^*$ into a nice “regular" one $\mathcal{X}$ as per figure below



$\mathscr{X}$: Set of objects (paths in a graph, colorings of a graph, etc.)

$\mathscr{X}^*$: Subset of special objects (cycles in a graph, colorings of a certain type, etc).

Counting via Monte Carlo

To calculate $|\mathcal{X}^*|$ we apply the following sampling procedure:

(i) Generate a random sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$, *uniformly* distributed over the "regular" region $\mathcal{X}$.

(ii) Estimate the desired area $|\mathcal{X}^*|$ as

$$\widehat{|\mathcal{X}^*|} = \widehat{\ell}|\mathcal{X}|,$$

where

$$\widehat{\ell} = \frac{N_{\mathcal{X}^*}}{N_{\mathcal{X}}} = \frac{1}{N}\sum_{k=1}^{N} I_{\{\boldsymbol{X}_k \in \mathcal{X}^*\}},$$

$I_{\{\boldsymbol{X}_k \in \mathcal{X}^*\}}$ denotes the indicator of the event $\{\boldsymbol{X}_k \in \mathcal{X}^*\}$ and $\{\boldsymbol{X}_k\}$ is a sample from $f(\boldsymbol{x})$ over $\mathcal{X}$, where $f(\boldsymbol{x}) = \frac{1}{|\mathcal{X}|}$.

The Approach

Each problem will be casted into the problem of estimation of the rare event probability of the type

$$\ell(m) = \mathbb{E}_f \left[I_{\{S(\boldsymbol{X}) \geq m\}} \right].$$

Here $S(\boldsymbol{X})$ is the sample performance, $\boldsymbol{X} \sim f(\boldsymbol{x})$ and m is fixed, called, the *level* chosen such that $\ell(m)$ is very small.

Approach

To estimate $\ell(m) = \mathbb{E}_f \left[I_{\{S(\boldsymbol{X}) \geq m\}} \right]$ we define a fixed grid $\{m_t, \ t = 0, 1, \ldots, T\}$ satisfying $-\infty < m_0 < m_1 < \ldots m_T = m$ and then use for $\ell(m)$ the well known chain (nested events) rule

$$\ell(m) = \mathbb{E}_f [I_{\{S(\boldsymbol{X}) \geq m_0\}}] \prod_{t=1}^{T} \mathbb{E}_f [I_{\{S(\boldsymbol{X}) \geq m_t\}} | I_{\{S(\boldsymbol{X}) \geq m_{t-1}\}}] = c_0 \prod_{t=1}^{T} c_t,$$

or as

$$\ell(m) = \mathbb{E}_f [I_{\{S(\boldsymbol{X}) \geq m_0\}}] \prod_{t=1}^{T} \mathbb{E}_{g^*_{t-1}} [I_{\{S(\boldsymbol{X}) \geq m_t\}}] = c_0 \prod_{t=1}^{T} c_t,$$

$$c_t = \mathbb{E}_f [I_{\{S(\boldsymbol{X}) \geq m_t\}} | I_{\{S(\boldsymbol{X}) \geq m_{t-1}\}}] = \mathbb{E}_{g^*_{t-1}} [I_{\{S(\boldsymbol{X}) \geq m_t\}}].$$

Approach

Here f denotes the proposal pdf $f = f(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{v}_0)$; and $g^*_{t-1} = g^*(\boldsymbol{x}, m_{t-1}) = \ell^{-1}_{t-1} f(\boldsymbol{x}) I_{\{S(\boldsymbol{x}) \geq m_{t-1}\}}$,denotes the zero variance importance sampling (IS) pdf at iteration $t-1$, where $\ell_{t-1} = \ell(m_{t-1}) = \mathbb{E}_f \left[I_{\{S(\boldsymbol{X}) \geq m_{t-1}\}} \right]$ is the normalization constant.

Note that the sequence c_t in the product formula for ℓ will be used only for counting.

Approach

The estimator of $\ell(m)$ is

$$\hat{\ell}(m) = \prod_{t=0}^{T} \hat{c}_t, \; \hat{c}_t = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq m_t\}},$$

where $\boldsymbol{X}_i \sim g^*_{t-1}$.
It is readily seen that if the proposal density $f(\boldsymbol{x})$ is *uniformly* distributed on the original set $\mathcal{X} = \{\boldsymbol{x} : S(\boldsymbol{x}) \geq m_{-1}\}$, than g^*_{t-1} is *uniformly* distributed on the set $\mathcal{X}_{t-1} = \{\boldsymbol{x} : S(\boldsymbol{x}) \geq m_{t-1}\}$.
The main trick of this work is to show how to sample from the IS pdf $g^*(\boldsymbol{x}, m_{t-1})$ *without knowing the normalization constant* $\ell_{t-1} = \ell(m_{t-1})$.

Approach

For such an estimator to be useful, the levels m_t should be chosen such that each quantity $\mathbb{E}_f\left[I_{\{S(\boldsymbol{X})\geq m_t\}} \middle| I_{\{S(\boldsymbol{X})\geq m_{t-1}\}}\right]$ is not too small, say approximately equal to 10^{-2}. In our approach we shall estimate each $\mathbb{E}_f\left[I_{\{S(\boldsymbol{X})\geq m_t\}} \middle| I_{\{S(\boldsymbol{X})\geq m_{t-1}\}}\right] = \rho$ by using the Gibbs sampler.

As mentioned, we shall generate here an adaptive sequence of tuples

$$\{(m_0, f(\boldsymbol{x}, \boldsymbol{v}_0)),\ (m_1, g^*(\boldsymbol{x}, m_0)), \ldots, (m_T, g^*(\boldsymbol{x}, m_{T-1}))\}$$

instead of the sequence

$$\{(m_0, \boldsymbol{v}_0),\ (m_1, \boldsymbol{v}_1), \ldots, (m_T, \boldsymbol{v}_T)\}.$$

Quick Glance

Consider

$$\ell(m) = \mathbb{E}_f \left[I_{\{\sum_{i=1}^{n} X_i \geq m\}} \right],$$

where all X_i's are iid Ber$(p = 1/2)$ random variables. Assume that we want to count the number of outcomes on the set

$$\mathcal{X}^* = \{x : \sum_{i=1}^{n} X_i \geq m\}.$$

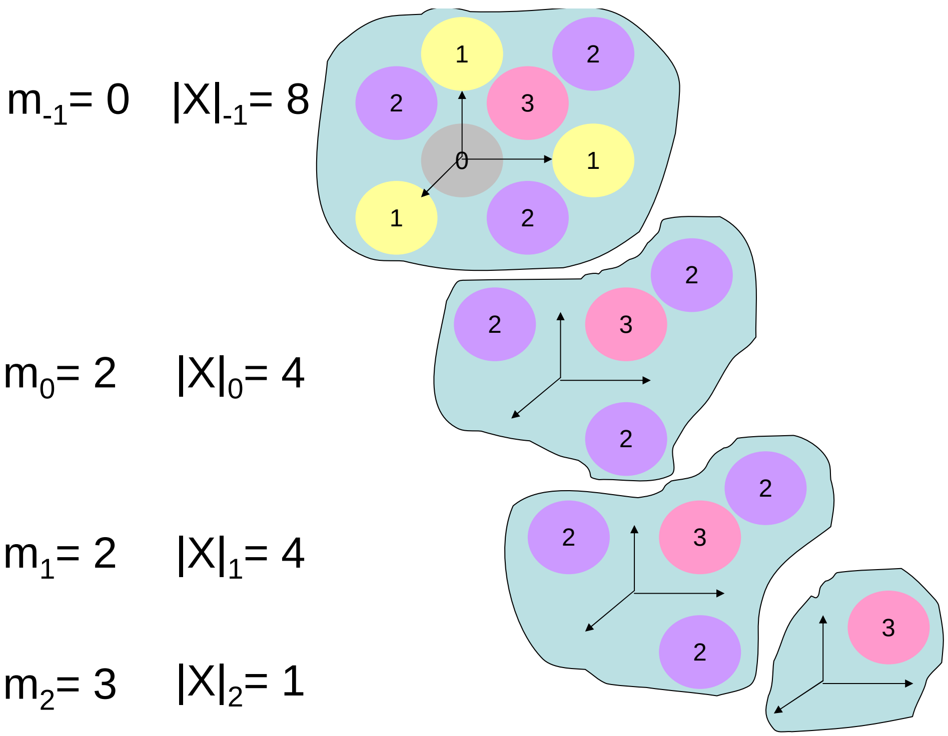
Let $n = m = 3$. Although it is obvious that $|\mathcal{X}^*| = 1$, we demonstrate the sampling mechanism in the product formula

$$\ell(m) = \mathbb{E}_f[I_{\{S(\boldsymbol{X}) \geq m_0\}}] \prod_{t=1}^{T} \mathbb{E}_{g^*_{t-1}}[I_{\{S(\boldsymbol{X}) \geq m_t\}}] = c_0 \prod_{t=1}^{T} c_t.$$

Quick Glance: Flipping 3 Coins

Possible dynamic of the evolution of the sequence of levels m_t and cardinalities $|\mathcal{X}_t|$, that is tuples

$$\{(m_{-1}, |\mathcal{X}_{-1}|),\ (m_0, |\mathcal{X}_0|), \ldots, (m, |\mathcal{X}_m|)\}.$$



Quick Glance

According to Figure we obtain $m_0 = 2$ after the first iteration, which means that while flipping 3 symmetric coins $\sum_{i=1}^{3} X_i = m_0 = 2$, (2 coins resulted to 1 and one coin resulted to 0). As soon as we obtain $m_0 = 2$ we reduce the original sample space $\mathcal{X}_{-1}$ containing 8 points to the one $\mathcal{X}_0$ containing 4 points. This is done by eliminating 4 outcomes corresponding to events $\{\sum_{i=1}^{3} X_i = 0\}$ and $\{\sum_{i=1}^{3} X_i = 1\}$ from the space $\mathcal{X}_{-1} = \{\boldsymbol{X} : \sum_{i=1}^{3} X_i \geq 0\}$. In other words, as soon as we obtain an outcome, such that $\sum_{i=1}^{3} X_i = 2$ we truncate the sample space $\mathcal{X}_{-1}$ by excluding from it all points corresponding to the event $\{\sum_{i=1}^{3} X_i \leq 1\}$, etc.

General Case: Multiple Constraints

Consider a set containing both equality and inequality constraints of an integer program, that is

$$\sum_{k=1}^{n} a_{ik} x_k = b_i, \; i = 1, \ldots, m_1,$$

$$\sum_{k=1}^{n} a_{jk} x_k \geq b_j, \; j = m_1 + 1, \ldots, m_1 + m_2,$$

$$\boldsymbol{x} \geq \mathbf{0}, \quad x_k \text{ integer } \forall k = 1, \ldots, n.$$

General Case: Multiple Constraints

It can be shown that in order to count the number of points (feasible solutions) of the above set one can consider the following associated rare-event probability problem

$$\ell(m) = \mathbb{E}_{\boldsymbol{u}} \left[I_{\{\sum_{i=1}^{m} C_i(\boldsymbol{X}) \geq m\}} \right],$$

where the first m_1 terms $C_i(\boldsymbol{X})$'s are

$$C_i(\boldsymbol{X}) = I_{\{\sum_{k=1}^{n} a_{ik} X_k = b_i\}}, \; i = 1, \ldots, m_1,$$

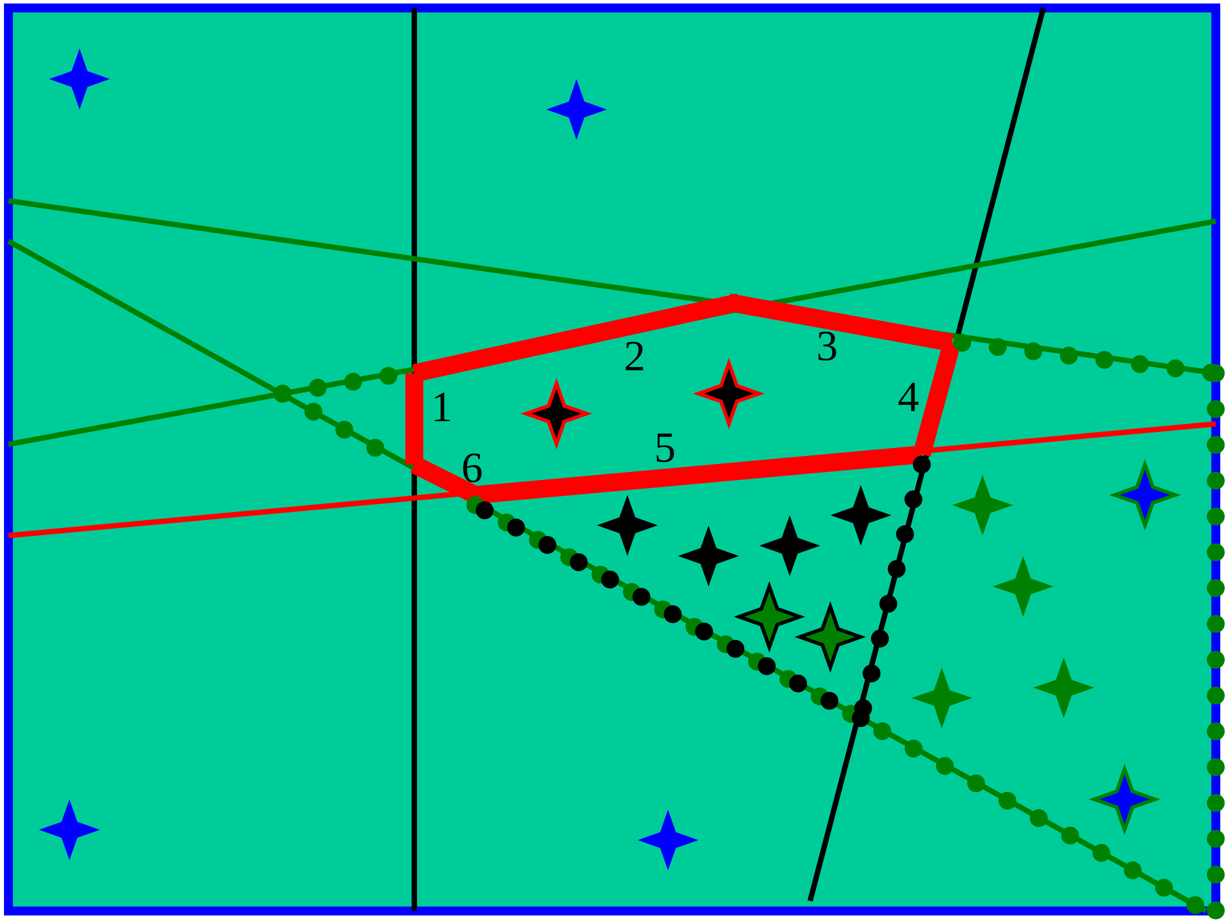
while the remaining m_2 ones are

$$C_i(\boldsymbol{X}) = I_{\{\sum_{k=1}^{n} a_{ik} X_k \geq b_i\}}, \; i = m_1 + 1, \ldots, m_1 + m_2.$$

General Case: Multiple Constraints

Thus, in order to count the the number of feasible solution on the above set we shall consider an associated rare event probability estimation problem involving a *sum of dependent Bernoulli random variables*. Such representation is crucial for a large set of counting problems.

Polytop



The Gibbs Sampler

Our goal is sample from the IS pdf $g^*(\boldsymbol{x})$ or any other pdf $g(\boldsymbol{x})$.
It is assumed that generating from the conditional pdfs
$g(X_i|X_1,\ldots,X_{i-1},X_{i+1},\ldots,X_n),\ i=1,\ldots,n$ is simple.
In Gibbs sampler for any given vector $\boldsymbol{X}=(X_1,\ldots,X_n)\in\mathcal{X}$
one generates a *new* vector $\widetilde{\boldsymbol{X}}=(\widetilde{X}_1,\ldots,\widetilde{X}_n)$ as:

Algorithm: The Gibbs Sampler

1. Draw $\widetilde{X}_1$ from the conditional pdf $g(X_1|X_2,\ldots,X_n)$.
2. Draw $\widetilde{X}_i$ from the conditional pdf $g(X_i|\widetilde{X}_1,\ldots,\widetilde{X}_{i-1},X_{i+1},\ldots,X_n),\ i=2,\ldots,n-1$.
3. Draw $\widetilde{X}_n$ from the conditional pdf $g(X_n|\widetilde{X}_1,\ldots,\widetilde{X}_{n-1})$.

After many *burn-in* periods $\widetilde{\boldsymbol{X}}$ is distributed $g(\boldsymbol{x})$.

The Gibbs Sampler: Example

Consider estimation

$$\ell(m) = \mathbb{E}_f \left[I_{\{\sum_{i=1}^n X_i \geq m\}} \right].$$

The Gibbs sampler for generating variables $X_i, \ i = 1, \ldots, N$ is

$$g^*(x_i, m | \boldsymbol{x}_{-i}) = c_i(m) f_i(x_i) I_{\{x_i \geq m - \sum_{j \neq i} x_j\}},$$

where $|\boldsymbol{x}_{-i}$ denotes conditioning on all random variables but *excluding* the remaining ones and $c_i(m)$ is the normalization constant. Sampling a random variable $\widetilde{X}_i$ can be performed as follows. Generate $Y \sim$ Ber $(1/2)$. If $I_{\{\widetilde{Y} \geq m - \sum_{j \neq i} x_j\}}$, then set $\widetilde{X}_i = Y$, oterwise set set $\widetilde{X}_i = 1 - Y$.

Cloning Algorithm for Counting

Given ρ, say $\rho = 0.1$, the sample size N, the burn in period b, say $3 \leq b \leq 10$ execute the following steps:

1. **Acceptance-Rejection**

Set a counter $t = 1$. Generate a sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from the proposal density $f(\boldsymbol{x})$. Let $\widetilde{\mathcal{X}}_0 = \{\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_0}\}$ be the largest subset of the population $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$, called the elite samples for which $S(\boldsymbol{X}_i) \geq m_0$. Note that $\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_0} \sim g^*(\boldsymbol{x}, m_0)$ and that

$$\hat{\ell}(m_0) = \hat{c}_0 = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq m_0\}} = \frac{N_0}{N}$$

is an *unbiased* estimator of $\ell(m_0)$.

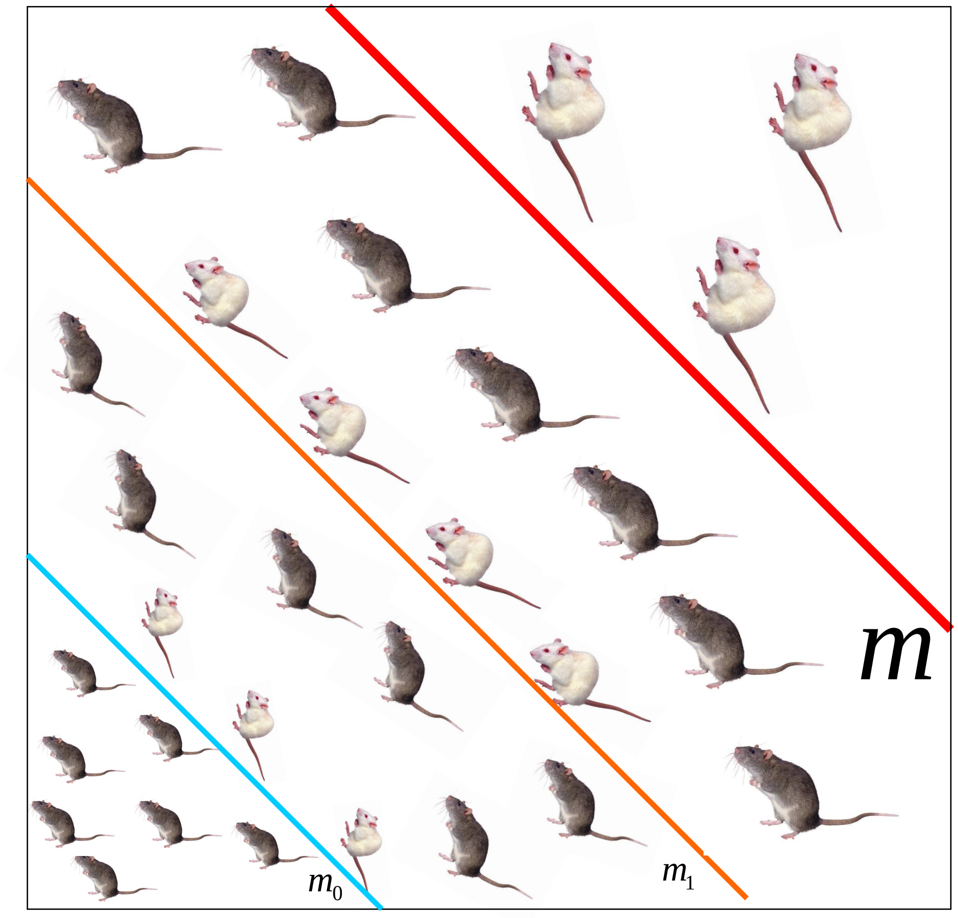
The Cloning Mechanism

The goal of the cloning parameter η is to reproduce η times the N_{t-1} elites at iteration $t-1$. After that we apply the burn-in period of length b the total ηN_{t-1} samples, such that $b\eta N_{t-1} = N$, that is

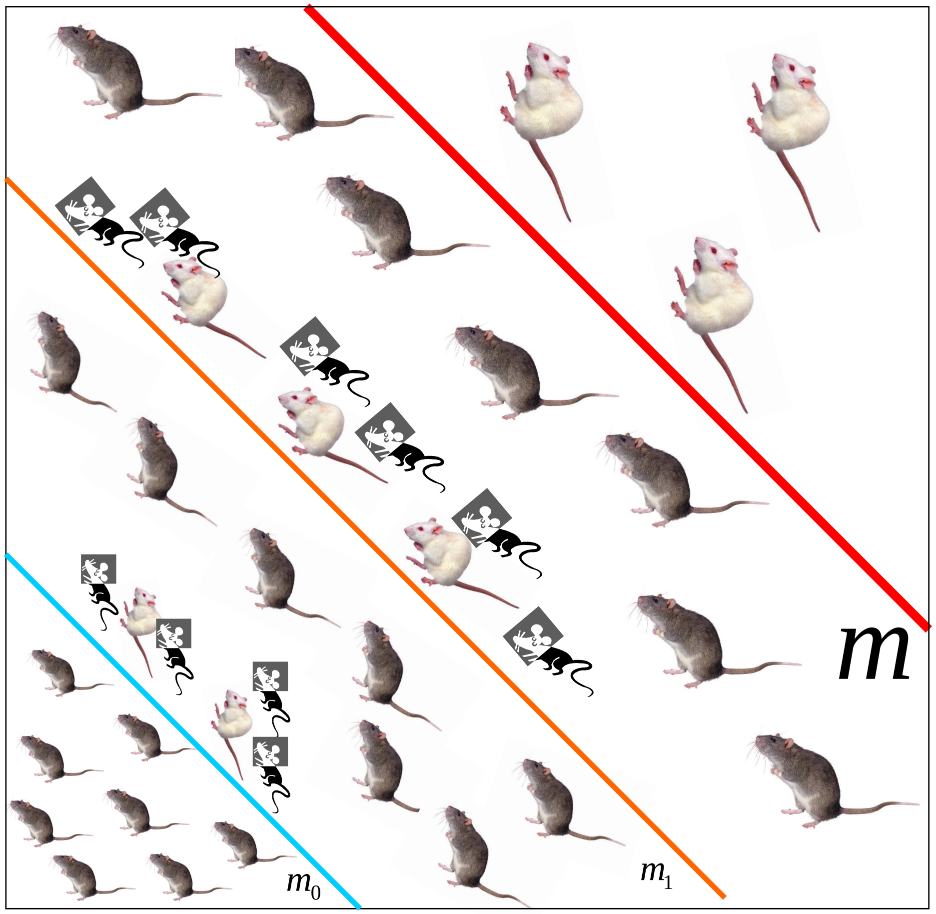
$$b_{t-1} = \left\lceil \frac{N}{\eta N_{t-1}} \right\rceil.$$

The goal of the cloning mechanism is to find a good balance in the Gibbs sampler in terms of bias-variance using N, N_{t-1}, η, b. As an example, let $N = 1,000$, $N_{t-1} = 20$, $\eta = 5$. We obtain $b = 10$. Our numerical studies show that it is quite reasonable to choose $3 \leq \eta \leq 5$.

No Cloning ($\eta = 1$) for $P(X_1 + X_2 \geq m)$



Cloning ($\eta = 2$) for $P(X_1 + X_2 \geq m)$



Cloning Algorithm for Counting

Given ρ, say $\rho = 0.1$, the sample size N, the burn in period b, say $3 \leq b \leq 10$ execute the following steps:

1. **Acceptance-Rejection**

Set a counter $t = 1$. Generate a sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from the proposal density $f(\boldsymbol{x})$. Let $\widetilde{\mathcal{X}}_0 = \{\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_0}\}$ be the largest subset of the population $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$, called the elite samples for which $S(\boldsymbol{X}_i) \geq m_0$. Note that $\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_0} \sim g^*(\boldsymbol{x}, m_0)$ and that

$$\hat{\ell}(m_0) = \hat{c}_0 = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq m_0\}} = \frac{N_0}{N}$$

is an *unbiased* estimator of $\ell(m_0)$.

The Cloning Algorithm for Counting

2. **Cloning** Given b and the number of elites N_{t-1} find the cloning parameter η_{t-1} according to $\eta_{t-1} = \left\lceil \frac{N}{bN_{t-1}} \right\rceil - 1$. Reproduce η_{t-1} times each vector $\widetilde{\boldsymbol{X}}_k = (\widetilde{X}_{1k}, \ldots, \widetilde{X}_{nk})$ of the elite sample $\{\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_{t-1}}\}$. Denote the entire new population by $\mathcal{X}_{cl} = \{(\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_1), \ldots, (\widetilde{\boldsymbol{X}}_{N_{t-1}}, \ldots, \widetilde{\boldsymbol{X}}_{N_{t-1}})\}$. To each of the cloned vectors of the population $\mathcal{X}_{cl}$ apply the Gibbs sampler for b_{t-1} burn-in periods. Denote the *new entire* population by $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$. Observe that each component of $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$ is distributed approximately $g^*(\boldsymbol{x}, \hat{m}_{t-1})$.

The Cloning Algorithm for Counting

3. **Estimating** $c_t = \mathbb{E}_f\left[I_{\{S(\boldsymbol{X})\geq m_t\}} \middle| I_{\{S(\boldsymbol{X})\geq m_{t-1}\}}\right]$. Let $\widetilde{\mathcal{X}}_t = \{\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_t}\}$ be the subset of the population $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$ for which $S(\boldsymbol{X}_i) \geq m_t$. Take

$$\hat{c}_t = \frac{1}{N}\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i)\geq m_t\}} = \frac{N_t}{N}$$

is an estimator of c_t. Note that $\widetilde{\boldsymbol{X}}_1, \ldots, \widetilde{\boldsymbol{X}}_{N_t}$ is distributed only *approximately* $g^*(\boldsymbol{x}, m_t)$.

The Cloning Algorithm for Counting

4. **Stopping Rule** If $t = T$ go to step 5, otherwise set $t = t + 1$ and repeat from step 2.

5. **Estimating $\ell(m)$.** Deliver

$$\hat{\ell}(m) = \prod_{t=0}^{T} \hat{c}_t = \frac{1}{N^T} \prod_{t=0}^{T} N_t$$

as an estimator of $\ell(m)$.

The Direct Estimator

3-SAT with Matrix $A = (75 \times 325)$, $N = 10,000$ and $\rho = 0.1$

	$\lvert\mathscr{X}^*\rvert$			Empirical			
t	Mean	Max	Min	Mean	Max	Min	m_t
1	5.4e+020	5.6e+020	5.1e+020	0.0	0.0	0.0	292
4	1.2e+018	1.3e+018	1.1e+018	0.0	0.0	0.0	304
7	6.1e+015	6.8e+015	5.7e+015	0.0	0.0	0.0	310
10	5.0e+012	5.7e+012	4.4e+012	0.0	0.0	0.0	315
13	2.5e+010	2.8e+010	2.1e+010	0.0	0.0	0.0	318
16	3.5e+008	4.7e+008	4.2e+007	0.0	0.0	0.0	321
20	2341.2	2924.0	1749.9	2203.5	2224.0	2181.0	325
21	2341.2	2924.0	1749.9	2225.0	2247.0	2197.0	325

Dynamics for 3-SAT with Matrix $A = (75 \times 325)$

t	$\lvert \mathscr{X}^* \rvert$	Empirical	$N_{t,e}$	$N_{t,e}^{(s)}$	m_t^*	m_{*t}	ρ_t
1	5.4e+020	0.0	1020	1020	305	292	0.11
4	1.2e+018	0.0	1462	1462	310	304	0.12
7	6.1e+015	0.0	1501	1501	316	310	0.12
10	5.0e+012	0.0	2213	2213	320	315	0.23
13	2.5e+010	0.0	1962	1962	321	318	0.17
16	3.5e+008	0.0	1437	1437	324	321	0.12
20	2341	2203	196	187	325	325	0.01
21	2341	2225	10472	2199	325	325	1.00

Complexity of the $(N = 1)$-policy Algorithm

According to the $(N = 1)$-policy algorithm, at each fixed level m_{t-1} we use the acceptance-rejection (single trial) method, until for the first time we hit a higher level $m_t > m_{t-1}$.

Theorem. Under some mild conditions, the average number of iterations and the associated variance to hit the desired level m while estimating

$$\ell(m) = \mathbb{E}_{\boldsymbol{u}}\left[I_{\{\sum_{i=1}^{m} C_i(\boldsymbol{X}) \geq m\}}\right]$$

by using the $(N = 1)$-policy algorithm is at most

$$O(n^b \ln \frac{n}{n+1-m}) \;\; \text{and } \mathrm{O}(\mathrm{n}^{2\mathrm{b}}),$$

where $1 \leq b = b(p) \leq 2$.

Further Research