The Cross-Entropy Method

A Unified Approach to Rare Event Simulation and Stochastic Optimization

*Dirk P. Kroese   Reuven Y. Rubinstein

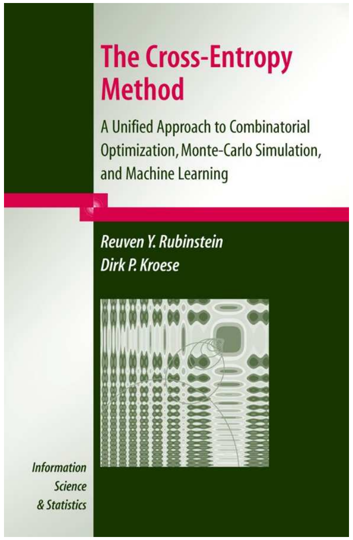
*Department of Mathematics, The University of Queensland, Australia
Faculty of Industrial Engineering and Management, Technion, Israel

Contents

1. Introduction
2. CE Methodology
3. Application: Max-Cut Problem, etc.
4. Some Theory on CE
5. Conclusion

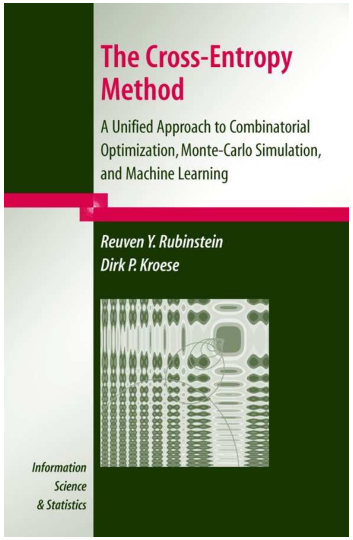
CE Matters

Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.



CE Matters

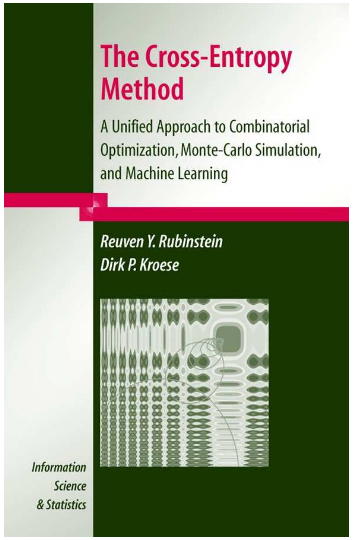
Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.



Special Issue: *Annals of Operations Research* (Jan 2005).

CE Matters

Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.



Special Issue: *Annals of Operations Research* (Jan 2005).

The CE home page:

`http://www.cemethod.org`

Introduction

The Cross-Entropy Method was originally developed as a simulation method for the estimation of *rare event* probabilities:

$$\boxed{\text{Estimate} \quad \mathbb{P}(S(\boldsymbol{X}) \geq \gamma)}$$

$\boldsymbol{X}$: random vector/process taking values in some set $\mathcal{X}$.

S: real-values function on $\mathcal{X}$.

Introduction

The Cross-Entropy Method was originally developed as a simulation method for the estimation of *rare event* probabilities:

$$\boxed{\text{Estimate} \quad \mathbb{P}(S(\boldsymbol{X}) \geq \gamma)}$$

$\boldsymbol{X}$: random vector/process taking values in some set $\mathcal{X}$.
S: real-values function on $\mathcal{X}$.

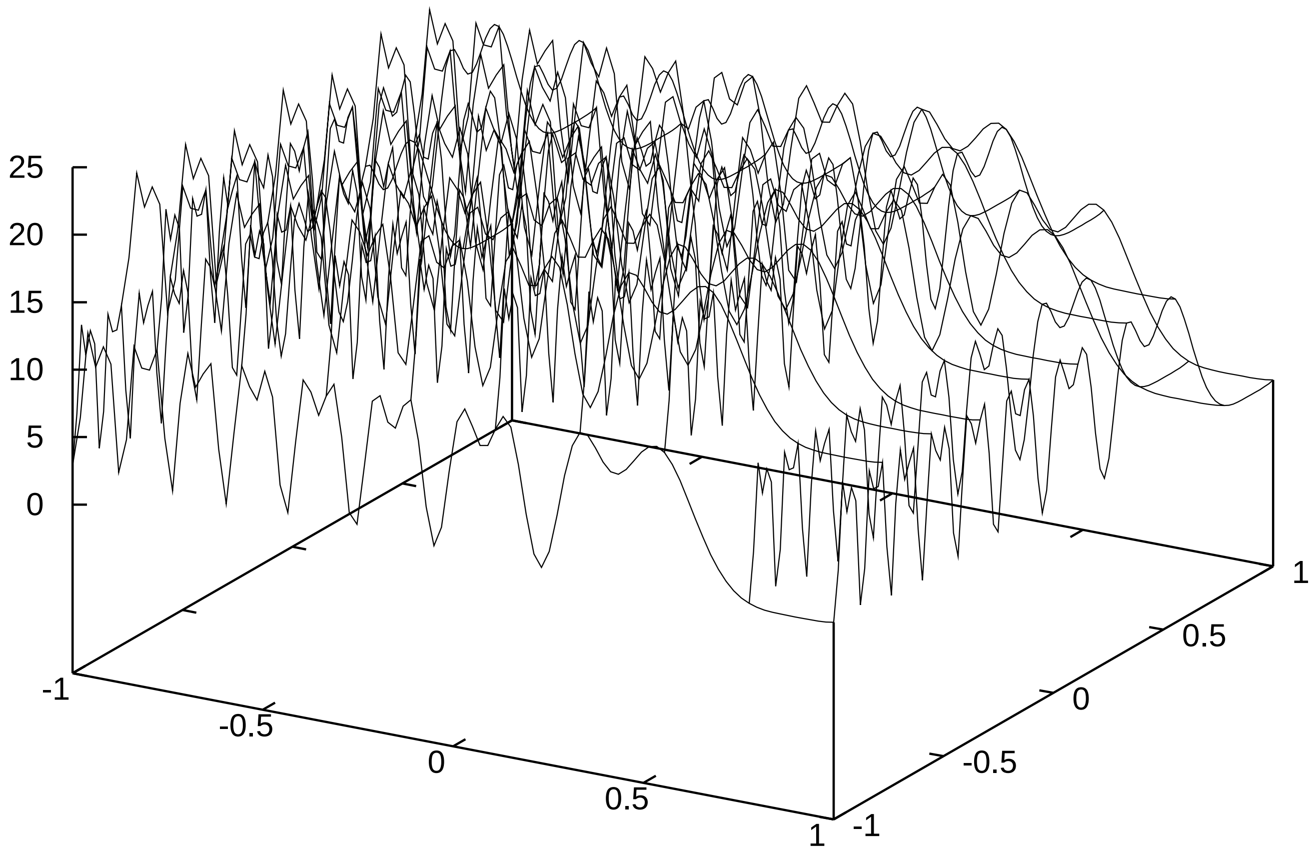
It was soon realised that the CE Method could also be used as an *optimization* method:

$$\boxed{\text{Determine} \quad \max_{\boldsymbol{x} \in \mathcal{X}} \; S(\boldsymbol{x})}$$

Some Applications

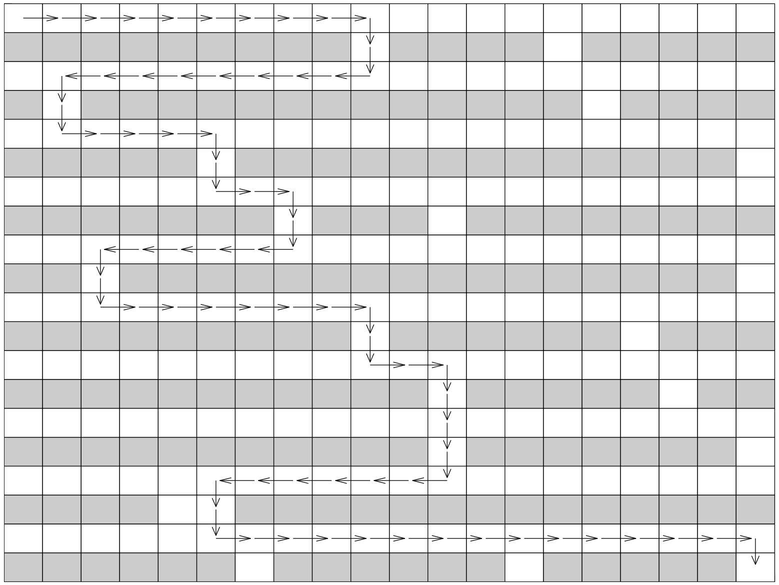
- Combinatorial Optimization (e.g., Travelling Salesman, Maximal Cut and Quadratic Assignment Problems)
- Noisy Optimization (e.g., Buffer Allocation, Financial Engineering)
- Multi-Extremal Continuous Optimization
- Pattern Recognition, Clustering and Image Analysis
- Production Lines and Project Management
- Network Reliability Estimation
- Vehicle Routing and Scheduling
- DNA Sequence Alignment

A Multi-extremal function



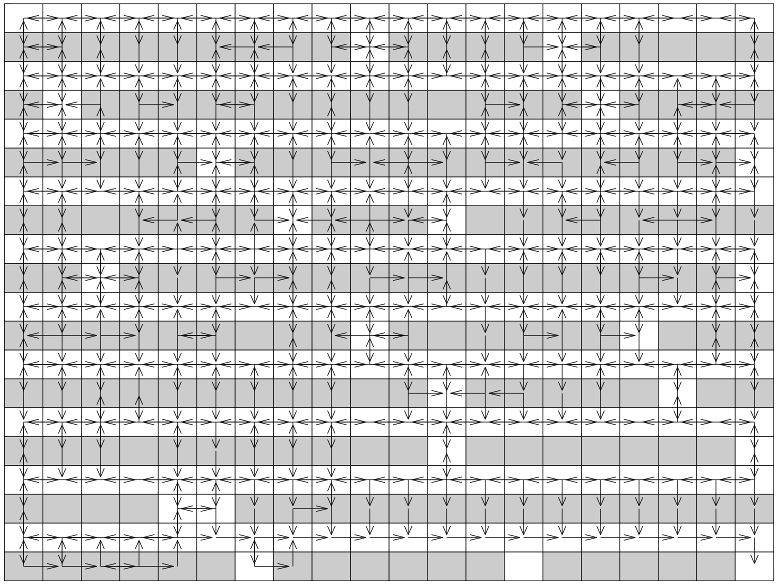
A Maze Problem

The Optimal Trajectory



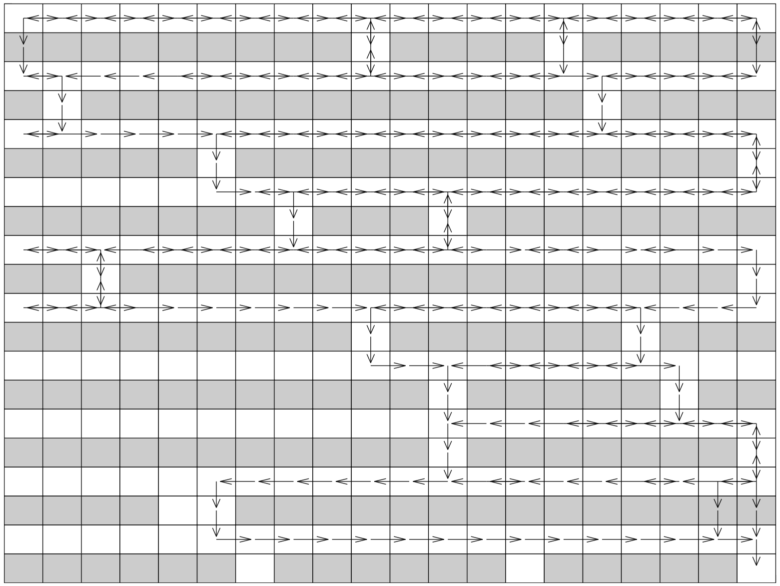
A Maze Problem

Iteration 1:



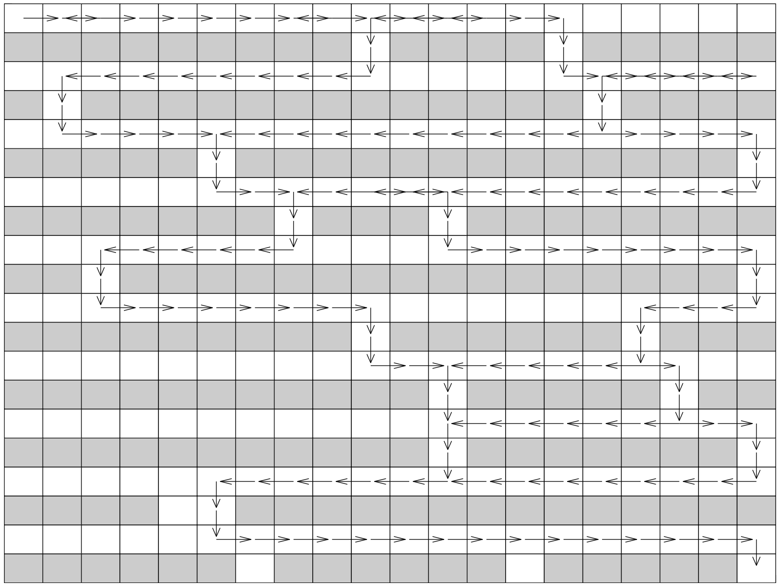
A Maze Problem

Iteration 2:



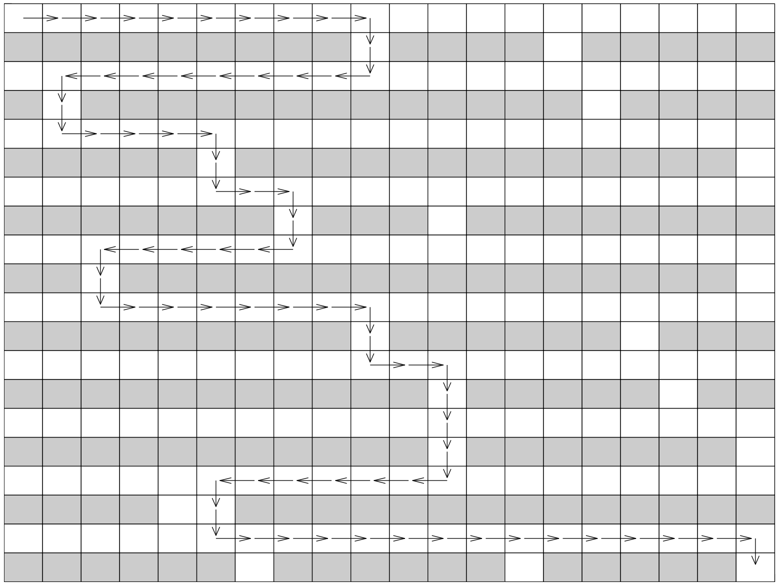
A Maze Problem

Iteration 3:



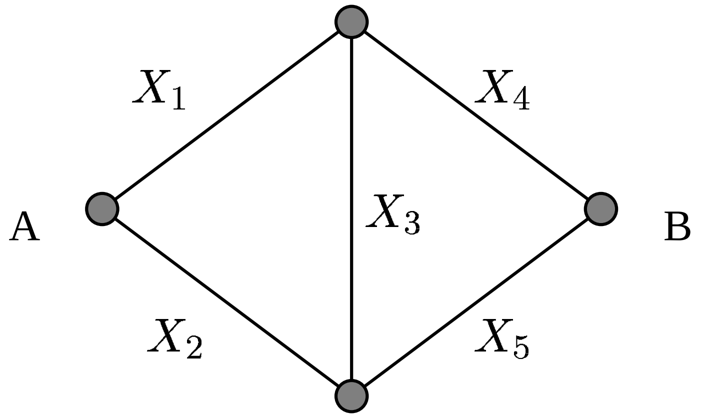
A Maze Problem

Iteration 4:



Example 1: Rare Event Simulation

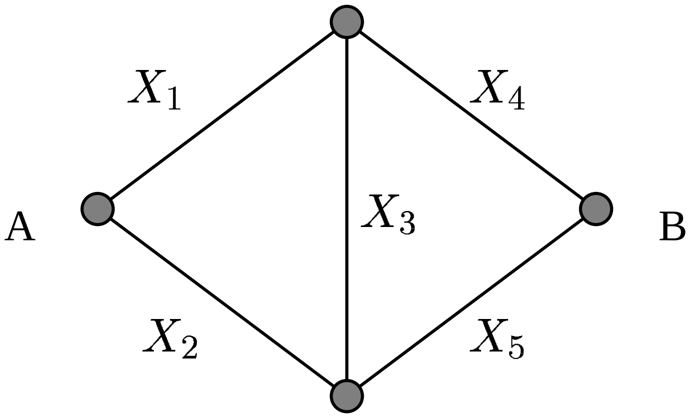
Consider a randomly weighted graph:



The random weights $X_1, \ldots, X_5$ are independent and exponentially distributed with means $u_1, \ldots, u_5$.

Example 1: Rare Event Simulation

Consider a randomly weighted graph:



The random weights $X_1, \ldots, X_5$ are independent and exponentially distributed with means $u_1, \ldots, u_5$.

Find the probability that the length of the shortest path from A to B is greater than or equal to γ.

Crude Monte Carlo (CMC)

Define $\boldsymbol{X} = (X_1, \ldots, X_5)$ and $\boldsymbol{u} = (u_1, \ldots, u_5)$. Let $S(\boldsymbol{X})$ be the length of the shortest path from node A to node B.

Crude Monte Carlo (CMC)

Define $\boldsymbol{X} = (X_1, \ldots, X_5)$ and $\boldsymbol{u} = (u_1, \ldots, u_5)$. Let $S(\boldsymbol{X})$ be the length of the shortest path from node A to node B.

We wish to estimate

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E} I_{\{S(\boldsymbol{X}) \geq \gamma\}} \ .$$

Crude Monte Carlo (CMC)

Define $\boldsymbol{X} = (X_1, \ldots, X_5)$ and $\boldsymbol{u} = (u_1, \ldots, u_5)$. Let $S(\boldsymbol{X})$ be the length of the shortest path from node A to node B.

We wish to estimate

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E} I_{\{S(\boldsymbol{X}) \geq \gamma\}} \ .$$

This can be done via Crude Monte Carlo: sample independent vectors from density $f(\boldsymbol{x}; \boldsymbol{u}) = \prod_{j=1}^{5} \exp(-x_j/u_j)/u_j$, and estimate ℓ via

$$\frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \ .$$

Importance Sampling (IS)

However, for small ℓ this requires a very large simulation effort.

Importance Sampling (IS)

However, for small ℓ this requires a very large simulation effort.

A better way is to use Importance Sampling: draw $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from a different density g, and estimate ℓ via the estimator

$$\widehat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \, W(\boldsymbol{X}_i) \, ,$$

where $W(\boldsymbol{X}) = f(\boldsymbol{X})/g(\boldsymbol{X})$ is called the likelihood ratio.

Which Change of Measure?

If we restrict ourselves to g such that $X_1, \ldots, X_5$ are independent and exponentially distributed with means $v_1, \ldots, v_5$, then

$$W(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}) := \frac{f(\boldsymbol{x}; \boldsymbol{u})}{f(\boldsymbol{x}; \boldsymbol{v})} = \exp\left(-\sum_{j=1}^{5} x_j \left(\frac{1}{u_j} - \frac{1}{v_j}\right)\right) \prod_{j=1}^{5} \frac{v_j}{u_j} \, .$$

In this case the "change of measure" is determined by the reference vector $\boldsymbol{v} = (v_1, \ldots, v_5)$.

Which Change of Measure?

If we restrict ourselves to g such that $X_1, \ldots, X_5$ are independent and exponentially distributed with means $v_1, \ldots, v_5$, then

$$W(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}) := \frac{f(\boldsymbol{x}; \boldsymbol{u})}{f(\boldsymbol{x}; \boldsymbol{v})} = \exp\left(-\sum_{j=1}^{5} x_j \left(\frac{1}{u_j} - \frac{1}{v_j}\right)\right) \prod_{j=1}^{5} \frac{v_j}{u_j} \; .$$

In this case the "change of measure" is determined by the reference vector $\boldsymbol{v} = (v_1, \ldots, v_5)$.

Question: How do we find the optimal $\boldsymbol{v} = \boldsymbol{v}^*$?

Which Change of Measure?

If we restrict ourselves to g such that $X_1, \ldots, X_5$ are independent and exponentially distributed with means $v_1, \ldots, v_5$, then

$$W(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}) := \frac{f(\boldsymbol{x}; \boldsymbol{u})}{f(\boldsymbol{x}; \boldsymbol{v})} = \exp\left(-\sum_{j=1}^{5} x_j \left(\frac{1}{u_j} - \frac{1}{v_j}\right)\right) \prod_{j=1}^{5} \frac{v_j}{u_j} \,.$$

In this case the "change of measure" is determined by the reference vector $\boldsymbol{v} = (v_1, \ldots, v_5)$.

Question: How do we find the optimal $\boldsymbol{v} = \boldsymbol{v}^*$?

Answer: Let CE find it adaptively!

CE Algorithm

1 Define $\hat{\boldsymbol{v}}_0 := \boldsymbol{u}$. Set $t := 1$ (iteration counter).

CE Algorithm

1 Define $\hat{\boldsymbol{v}}_0 := \boldsymbol{u}$. Set $t := 1$ (iteration counter).

2 Update $\hat{\gamma}_t$: Generate $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ according to $f(\cdot; \hat{\boldsymbol{v}}_{t-1})$. Let $\hat{\gamma}_t$ be the worst of the $\rho \times N$ best performances, provided this is less than γ. Else $\hat{\gamma}_t := \gamma$.

CE Algorithm

1 Define $\hat{\boldsymbol{v}}_0 := \boldsymbol{u}$. Set $t := 1$ (iteration counter).

2 Update $\hat{\gamma}_t$: Generate $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ according to $f(\cdot; \hat{\boldsymbol{v}}_{t-1})$. Let $\hat{\gamma}_t$ be the worst of the $\rho \times N$ best performances, provided this is less than γ. Else $\hat{\gamma}_t := \gamma$.

3 Update $\hat{\boldsymbol{v}}_t$: Use the **same** sample to calculate, for $j = 1, \ldots, n$,

$$\hat{v}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1}) \, X_{ij}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1})} \; .$$

THIS UPDATING IS BASED ON CE.

CE Algorithm

1 Define $\hat{\boldsymbol{v}}_0 := \boldsymbol{u}$. Set $t := 1$ (iteration counter).

2 Update $\hat{\gamma}_t$: Generate $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ according to $f(\cdot; \hat{\boldsymbol{v}}_{t-1})$. Let $\hat{\gamma}_t$ be the worst of the $\rho \times N$ best performances, provided this is less than γ. Else $\hat{\gamma}_t := \gamma$.

3 Update $\hat{\boldsymbol{v}}_t$: Use the **same** sample to calculate, for $j = 1, \ldots, n$,

$$\hat{v}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1})\, X_{ij}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1})} \; .$$

THIS UPDATING IS BASED ON CE.

4 If $\hat{\gamma}_t = \gamma$ then proceed to step 5; otherwise set $t := t + 1$ and reiterate from step 2.

CE Algorithm

1. Define $\hat{\boldsymbol{v}}_0 := \boldsymbol{u}$. Set $t := 1$ (iteration counter).
2. Update $\hat{\gamma}_t$: Generate $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ according to $f(\cdot; \hat{\boldsymbol{v}}_{t-1})$. Let $\hat{\gamma}_t$ be the worst of the $\rho \times N$ best performances, provided this is less than γ. Else $\hat{\gamma}_t := \gamma$.
3. Update $\hat{\boldsymbol{v}}_t$: Use the **same** sample to calculate, for $j = 1, \ldots, n$,

$$\hat{v}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1})\, X_{ij}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1})} .$$

   THIS UPDATING IS BASED ON CE.

4. If $\hat{\gamma}_t = \gamma$ then proceed to step 5; otherwise set $t := t + 1$ and reiterate from step 2.
5. Estimate ℓ via the LR estimator, using the final $\hat{\boldsymbol{v}}_T$.

Example

Level: $\gamma = 2$. Fraction of best performances: $\rho = 0.1$. Sample size in steps 2 – 4: $N = 1000$. Final sample size: $N_1 = 10^5$.

t	$\widehat{\gamma}_t$	$\widehat{\boldsymbol{v}}_t$				
0		0.250	0.400	0.100	0.300	0.200
1	0.575	0.513	0.718	0.122	0.474	0.335
2	1.032	0.873	1.057	0.120	0.550	0.436
3	1.502	1.221	1.419	0.121	0.707	0.533
4	1.917	1.681	1.803	0.132	0.638	0.523
5	2.000	1.692	1.901	0.129	0.712	0.564

Example (cont.)

- The estimate was $1.34 \cdot 10^{-5}$,

Example (cont.)

- The estimate was $1.34 \cdot 10^{-5}$,
- with an estimated relative error (that is, Std$(\widehat{\ell})/\mathbb{E}\widehat{\ell}$) of 0.03.

Example (cont.)

- The estimate was $1.34 \cdot 10^{-5}$,
- with an estimated relative error (that is, Std$(\widehat{\ell})/\mathbb{E}\widehat{\ell}$) of 0.03.
- The simulation time was only 3 seconds (1/2 second for table).

Example (cont.)

- The estimate was $1.34 \cdot 10^{-5}$,
- with an estimated relative error (that is, Std$(\widehat{\ell})/\mathbb{E}\widehat{\ell}$) of 0.03.
- The simulation time was only 3 seconds (1/2 second for table).
- CMC with $N_1 = 10^8$ samples gave an estimate $1.30 \cdot 10^{-5}$ with the same RE (0.03). The simulation time was 1875 seconds.

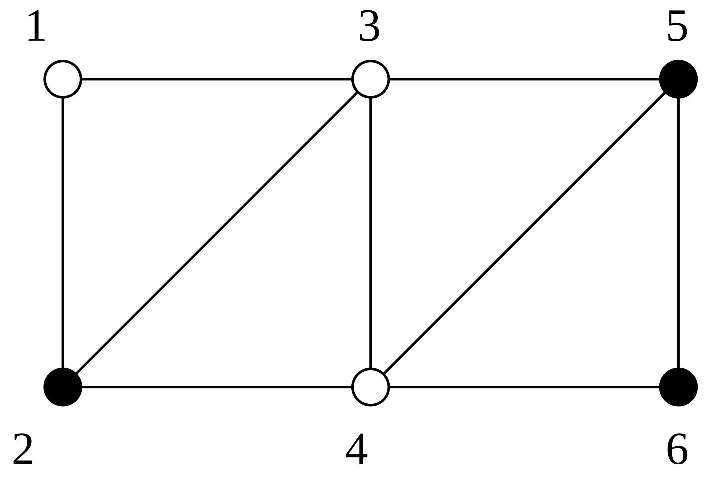
Example (cont.)

- The estimate was $1.34 \cdot 10^{-5}$,
- with an estimated relative error (that is, Std$(\widehat{\ell})/\mathbb{E}\widehat{\ell}$) of 0.03.
- The simulation time was only 3 seconds (1/2 second for table).
- CMC with $N_1 = 10^8$ samples gave an estimate $1.30 \cdot 10^{-5}$ with the same RE (0.03). The simulation time was 1875 seconds.
- With minimal effort we reduced our simulation time by a factor of 625.

Example 2: The Max-Cut Problem

Consider a weighted graph G with node set $V = \{1, \ldots, n\}$. Partition the nodes of the graph into two subsets V_1 and V_2 such that the sum of the weights of the edges going from one subset to the other is maximised.

Example



Cost matrix:

$$C = \begin{pmatrix} 0 & c_{12} & c_{13} & 0 & 0 & 0 \\ c_{21} & 0 & c_{23} & c_{24} & 0 & 0 \\ c_{31} & c_{32} & 0 & c_{34} & c_{35} & 0 \\ 0 & c_{42} & c_{43} & 0 & c_{45} & c_{46} \\ 0 & 0 & c_{53} & c_{54} & 0 & c_{56} \\ 0 & 0 & 0 & c_{64} & c_{65} & 0 \end{pmatrix}.$$

$\{V_1, V_2\} = \{\{1,3,4\},\{2,5,6\}\}$ is a possible cut. The cost of the cut is

$$c_{12} + c_{32} + c_{35} + c_{42} + c_{45} + c_{46}.$$

Random Cut Vector

We can represent a cut via a cut vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, where $x_i = 1$ if node i belongs to same partition as 1, and 0 else.

Random Cut Vector

We can represent a cut via a cut vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, where $x_i = 1$ if node i belongs to same partition as 1, and 0 else.

For example, the cut $\{\{1,3,4\},\{2,5,6\}\}$ can be represented via the cut vector $(1,0,1,1,0,0)$.

Random Cut Vector

We can represent a cut via a cut vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, where $x_i = 1$ if node i belongs to same partition as 1, and 0 else.

For example, the cut $\{\{1, 3, 4\}, \{2, 5, 6\}\}$ can be represented via the cut vector $(1, 0, 1, 1, 0, 0)$.

Let $\mathcal{X}$ be the set of all cut vectors $\boldsymbol{x} = (1, x_2, \ldots, x_n)$.

Random Cut Vector

We can represent a cut via a cut vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, where $x_i = 1$ if node i belongs to same partition as 1, and 0 else.

For example, the cut $\{\{1, 3, 4\}, \{2, 5, 6\}\}$ can be represented via the cut vector $(1, 0, 1, 1, 0, 0)$.

Let $\mathcal{X}$ be the set of all cut vectors $\boldsymbol{x} = (1, x_2, \ldots, x_n)$.

Let $S(\boldsymbol{x})$ be the corresponding cost of the cut.

Random Cut Vector

We can represent a cut via a cut vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, where $x_i = 1$ if node i belongs to same partition as 1, and 0 else.

For example, the cut $\{\{1,3,4\},\{2,5,6\}\}$ can be represented via the cut vector $(1,0,1,1,0,0)$.

Let $\mathcal{X}$ be the set of all cut vectors $\boldsymbol{x} = (1, x_2, \ldots, x_n)$.

Let $S(\boldsymbol{x})$ be the corresponding cost of the cut.

We wish to maximise $S(\boldsymbol{x})$ via the CE method.

General CE Procedure

First, cast the original optimization problem of $S(\boldsymbol{x})$ into an associated rare-events estimation problem: the estimation of

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E} I_{\{S(\boldsymbol{X}) \geq \gamma\}} \ .$$

Second, formulate a parameterized random mechanism to generate objects $\boldsymbol{X} \in \mathcal{X}$. Then, iterate the following steps:

General CE Procedure

First, cast the original optimization problem of $S(\boldsymbol{x})$ into an associated rare-events estimation problem: the estimation of

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E} I_{\{S(\boldsymbol{X}) \geq \gamma\}} \; .$$

Second, formulate a parameterized random mechanism to generate objects $\boldsymbol{X} \in \mathcal{X}$. Then, iterate the following steps:

• *Generate a random sample* of objects $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N \in \mathcal{X}$ (e.g., cut vectors).

General CE Procedure

First, cast the original optimization problem of $S(\boldsymbol{x})$ into an associated rare-events estimation problem: the estimation of

$$\ell = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E} I_{\{S(\boldsymbol{X}) \geq \gamma\}} \;.$$

Second, formulate a parameterized random mechanism to generate objects $\boldsymbol{X} \in \mathcal{X}$. Then, iterate the following steps:

- *Generate a random sample* of objects $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N \in \mathcal{X}$ (e.g., cut vectors).

- *Update the parameters* of the random mechanism (obtained via CE minimization), in order to produce a better sample in the next iteration.

Generation and Updating Formulas

Generation of cut vectors: The most natural and easiest way to generate the cut vectors is to let $X_2, \ldots, X_n$ be independent Bernoulli random variables with success probabilities $p_2, \ldots, p_n$.

Generation and Updating Formulas

Generation of cut vectors: The most natural and easiest way to generate the cut vectors is to let $X_2, \ldots, X_n$ be independent Bernoulli random variables with success probabilities $p_2, \ldots, p_n$.

Updating formulas: From CE minimization: the updated probabilities are the maximum likelihood estimates of the ρN best samples:

$$\hat{p}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} \, I_{\{X_{ij}=1\}}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}}}, \quad j = 2, \ldots, n \; .$$

Algorithm

1 Start with $\hat{\boldsymbol{p}}_0 = (1/2, \ldots, 1/2)$. Let $t := 1$.

Algorithm

1 **Start with $\hat{\boldsymbol{p}}_0 = (1/2, \ldots, 1/2)$. Let $t := 1$.**

2 **Update $\hat{\gamma}_t$: Draw $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from $\mathsf{Ber}(\hat{\boldsymbol{p}}_t)$. Let $\hat{\gamma}_t$ be the worst performance of the $\rho \times 100\%$ best performances.**

Algorithm

1 **Start with** $\hat{\boldsymbol{p}}_0 = (1/2, \ldots, 1/2)$**. Let** $t := 1$**.**

2 **Update** $\hat{\gamma}_t$**: Draw** $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ **from** $\mathsf{Ber}(\hat{\boldsymbol{p}}_t)$**. Let** $\hat{\gamma}_t$ **be the worst performance of the** $\rho \times 100\%$ **best performances.**

3 **Update** $\hat{\boldsymbol{p}}_t$**: Use the same sample to calculate**

$$\hat{p}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} \, I_{\{X_{ij}=1\}}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}}} \; ,$$

$j = 1, \ldots, n$**, where** $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{in})$**, and increase** t **by 1.**

Algorithm

1 **Start with $\hat{\boldsymbol{p}}_0 = (1/2, \ldots, 1/2)$. Let $t := 1$.**

2 **Update $\hat{\gamma}_t$: Draw $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from $\mathsf{Ber}(\hat{\boldsymbol{p}}_t)$. Let $\hat{\gamma}_t$ be the worst performance of the $\rho \times 100\%$ best performances.**

3 **Update $\hat{\boldsymbol{p}}_t$: Use the same sample to calculate**

$$\hat{p}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}} \, I_{\{X_{ij}=1\}}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \hat{\gamma}_t\}}} \ ,$$

$j = 1, \ldots, n$, where $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{in})$, and increase t by 1.

4 **If the stopping criterion is met, then stop; otherwise set $t := t + 1$ and reiterate from step 2.**

Example

- Results for the case with $n = 400, m = 200$ nodes are given next.

Example

- Results for the case with $n = 400, m = 200$ nodes are given next.
- Parameters: $\rho = 0.1$, $N = 1000$.

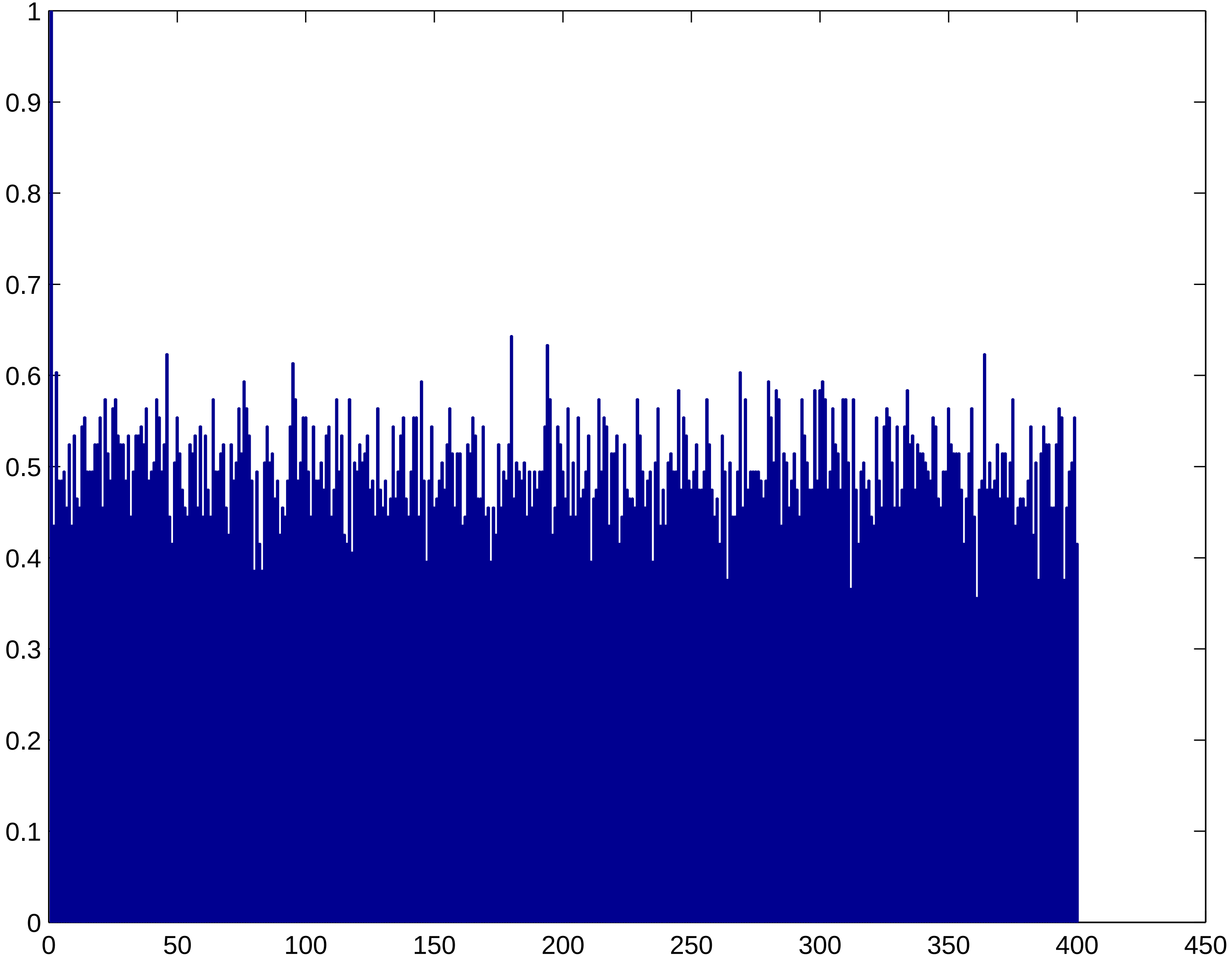
Example

- Results for the case with $n = 400, m = 200$ nodes are given next.
- Parameters: $\rho = 0.1$, $N = 1000$.
- The CPU time was only 100 seconds (Matlab, pentium III, 500 Mhz).

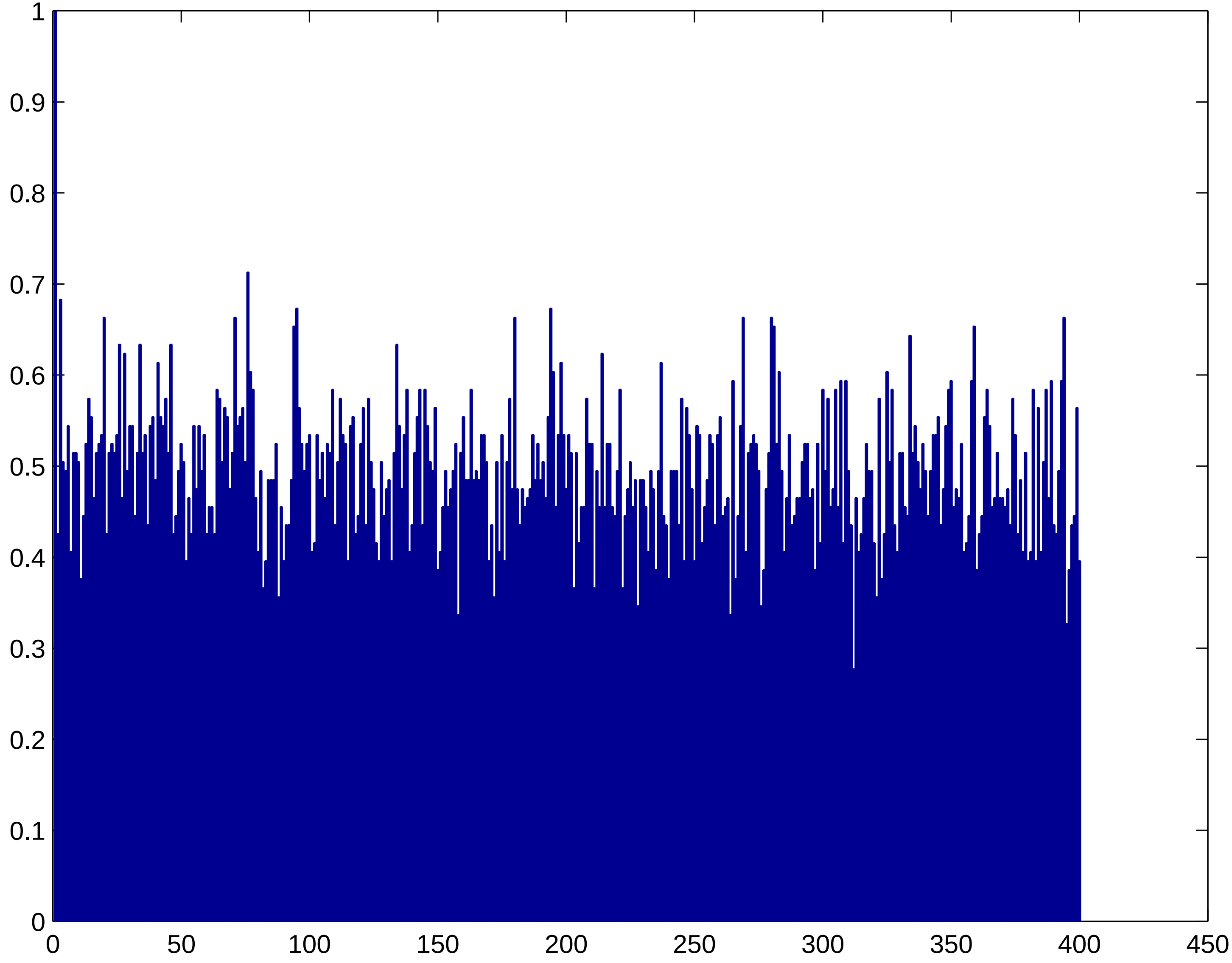
Example

- Results for the case with $n = 400, m = 200$ nodes are given next.
- Parameters: $\rho = 0.1$, $N = 1000$.
- The CPU time was only 100 seconds (Matlab, pentium III, 500 Mhz).
- The CE algorithm converges quickly, yielding the exact optimal solution 40000 in 22 iterations.

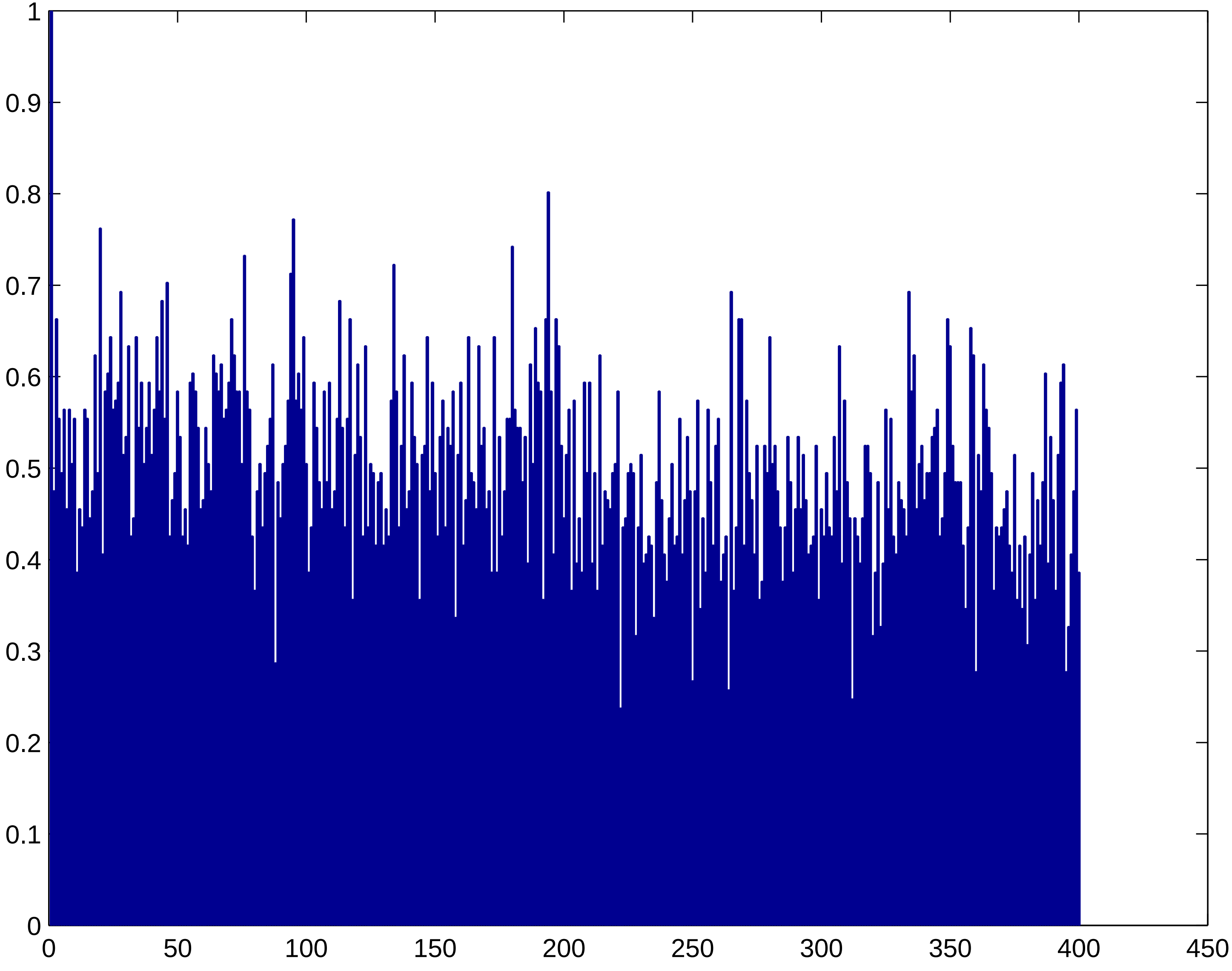
Max-Cut



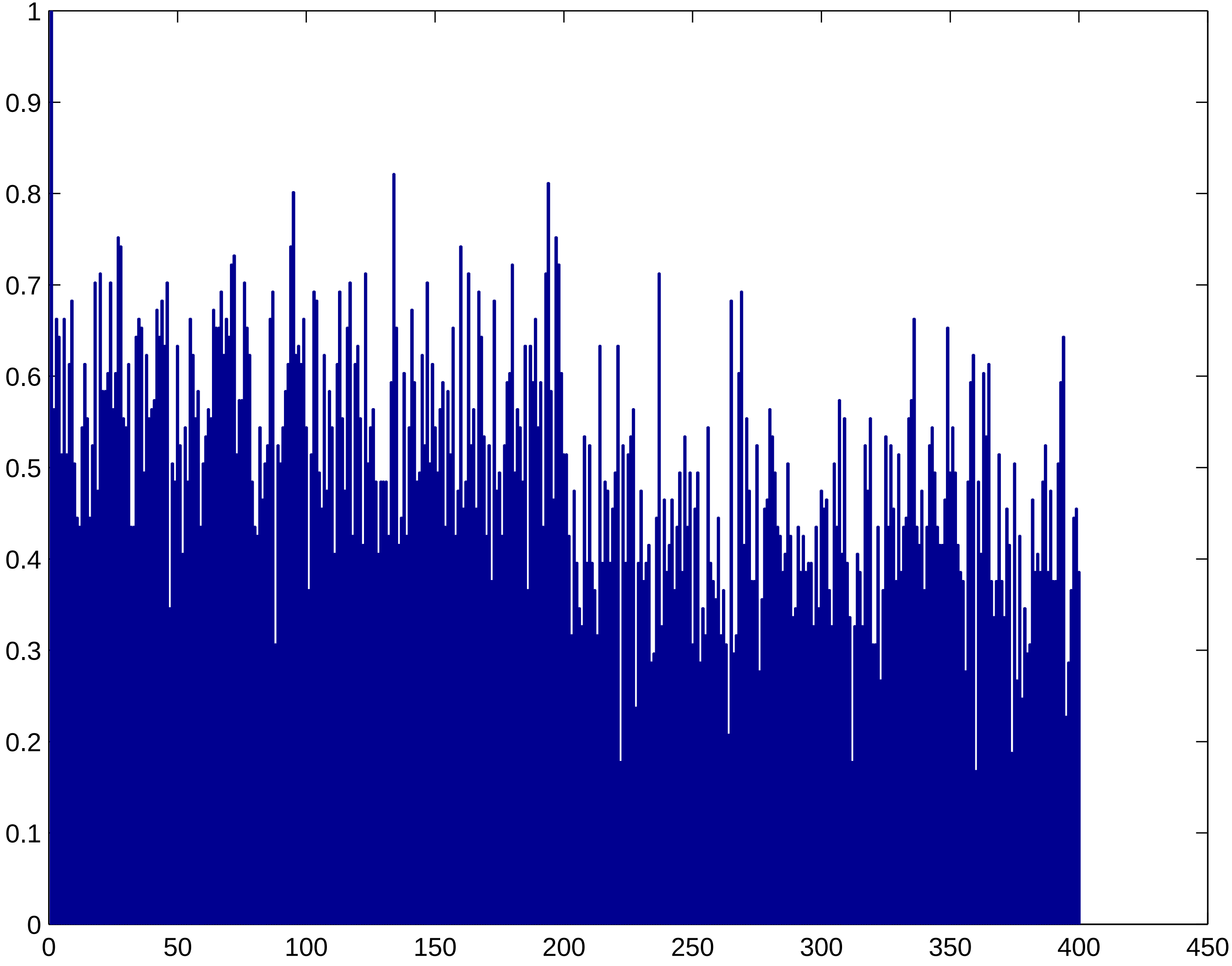
Max-Cut



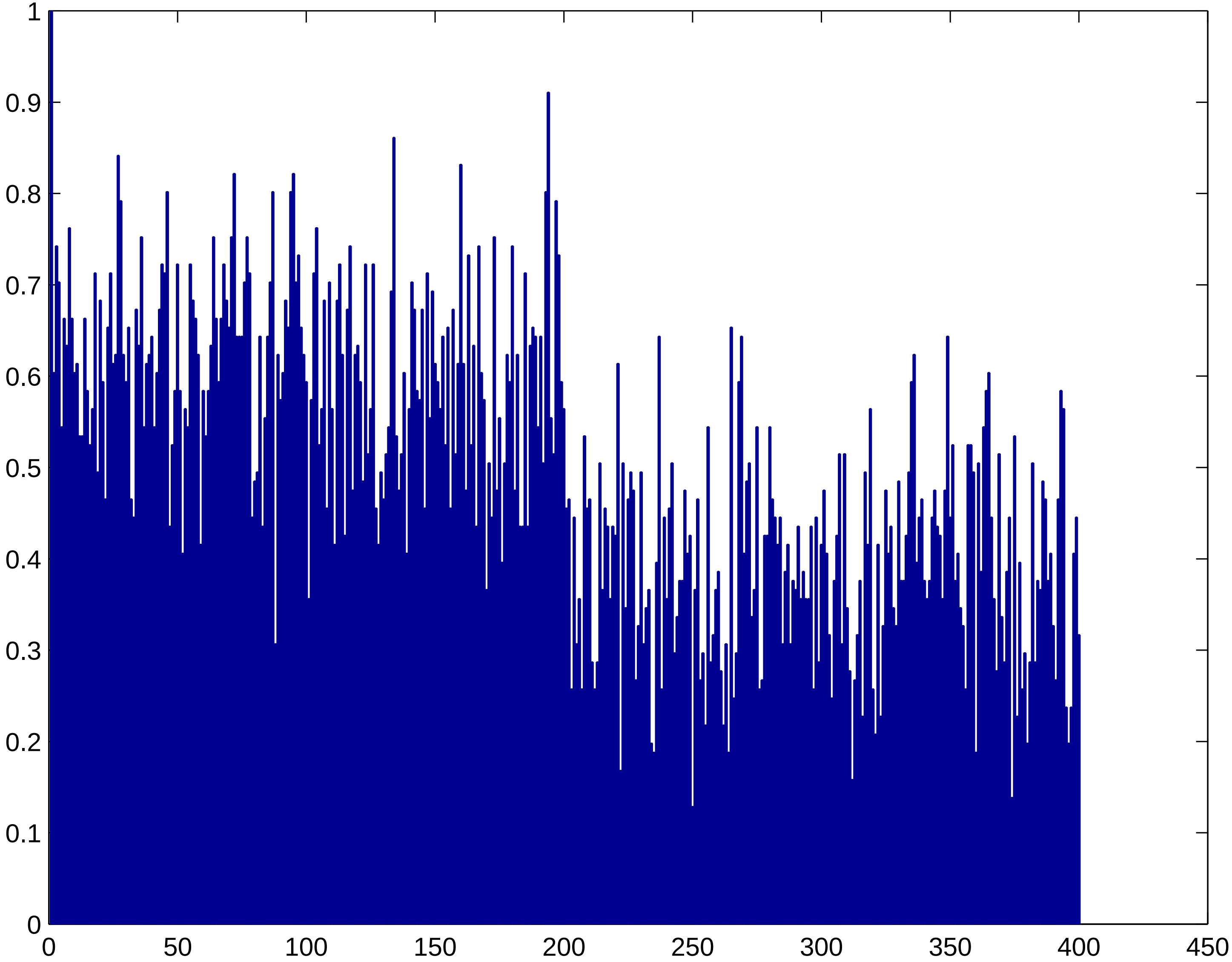
Max-Cut



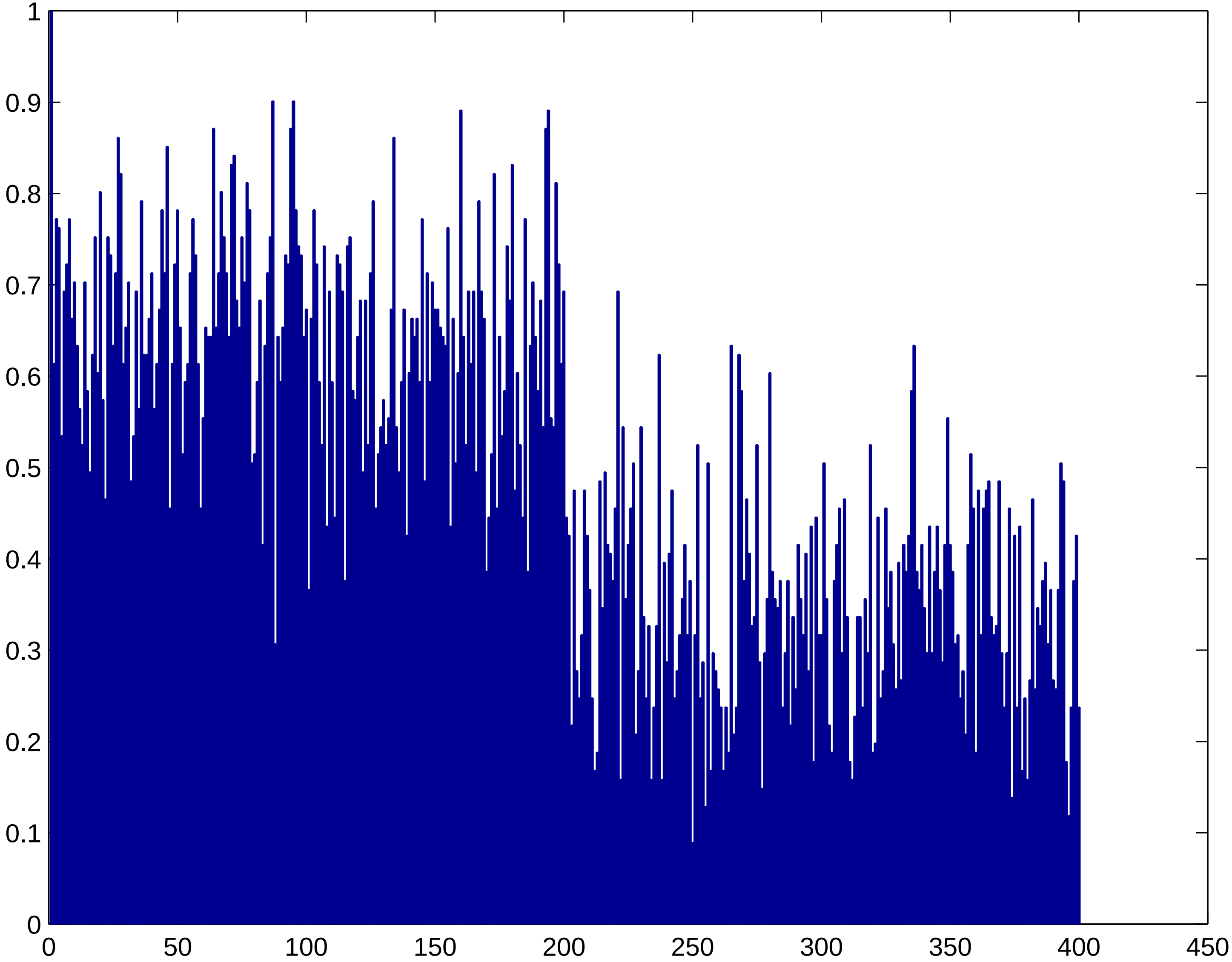
Max-Cut



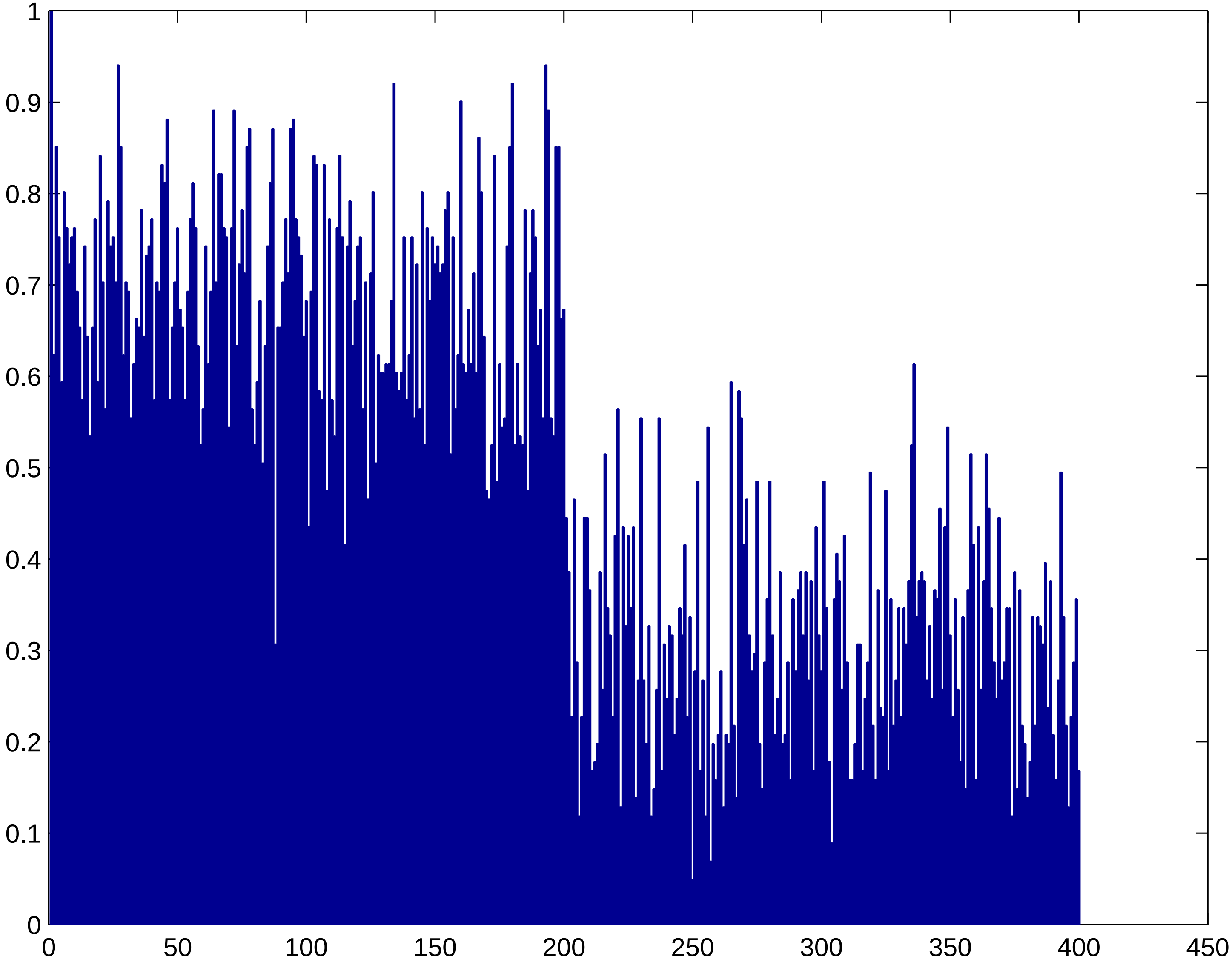
Max-Cut



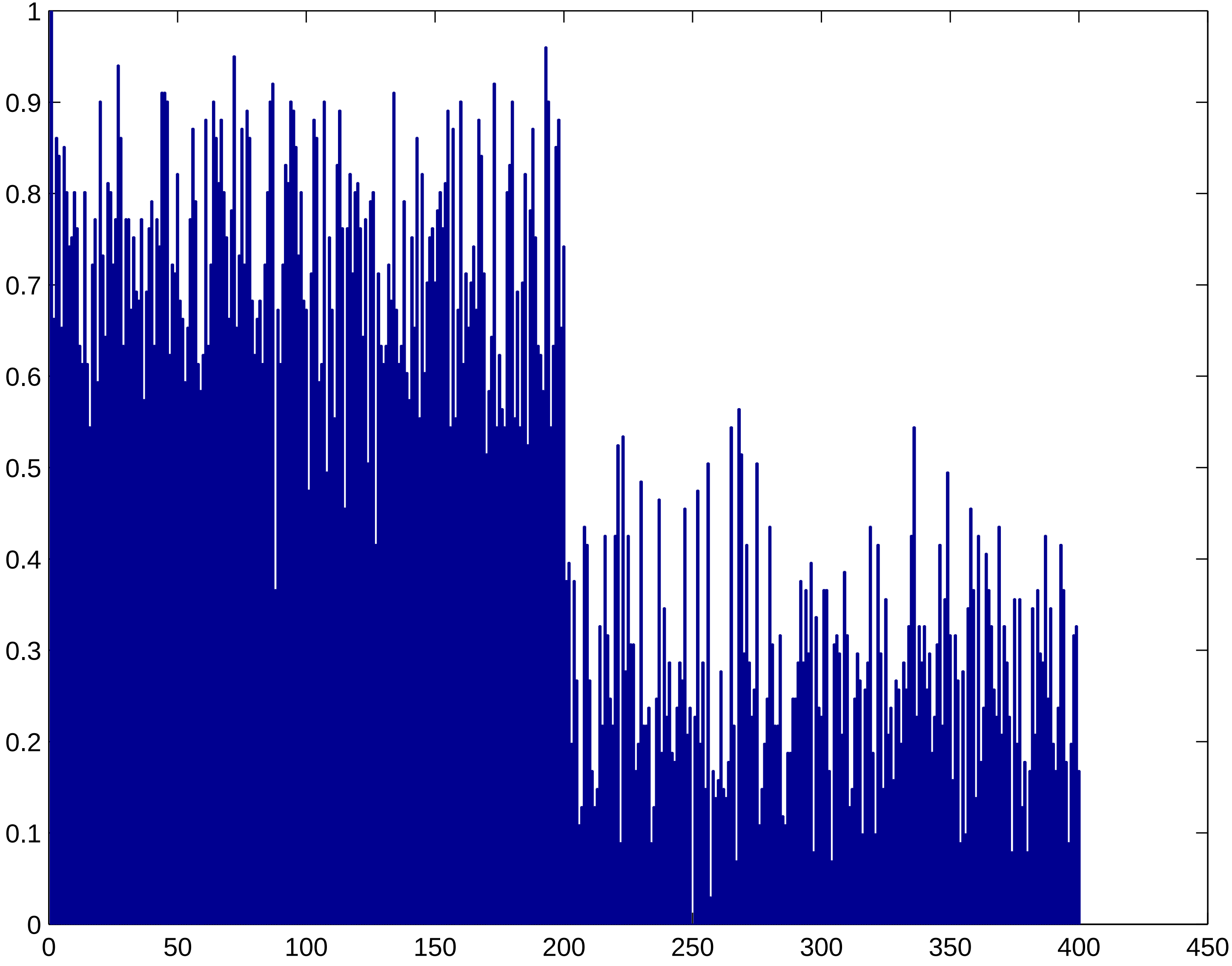
Max-Cut



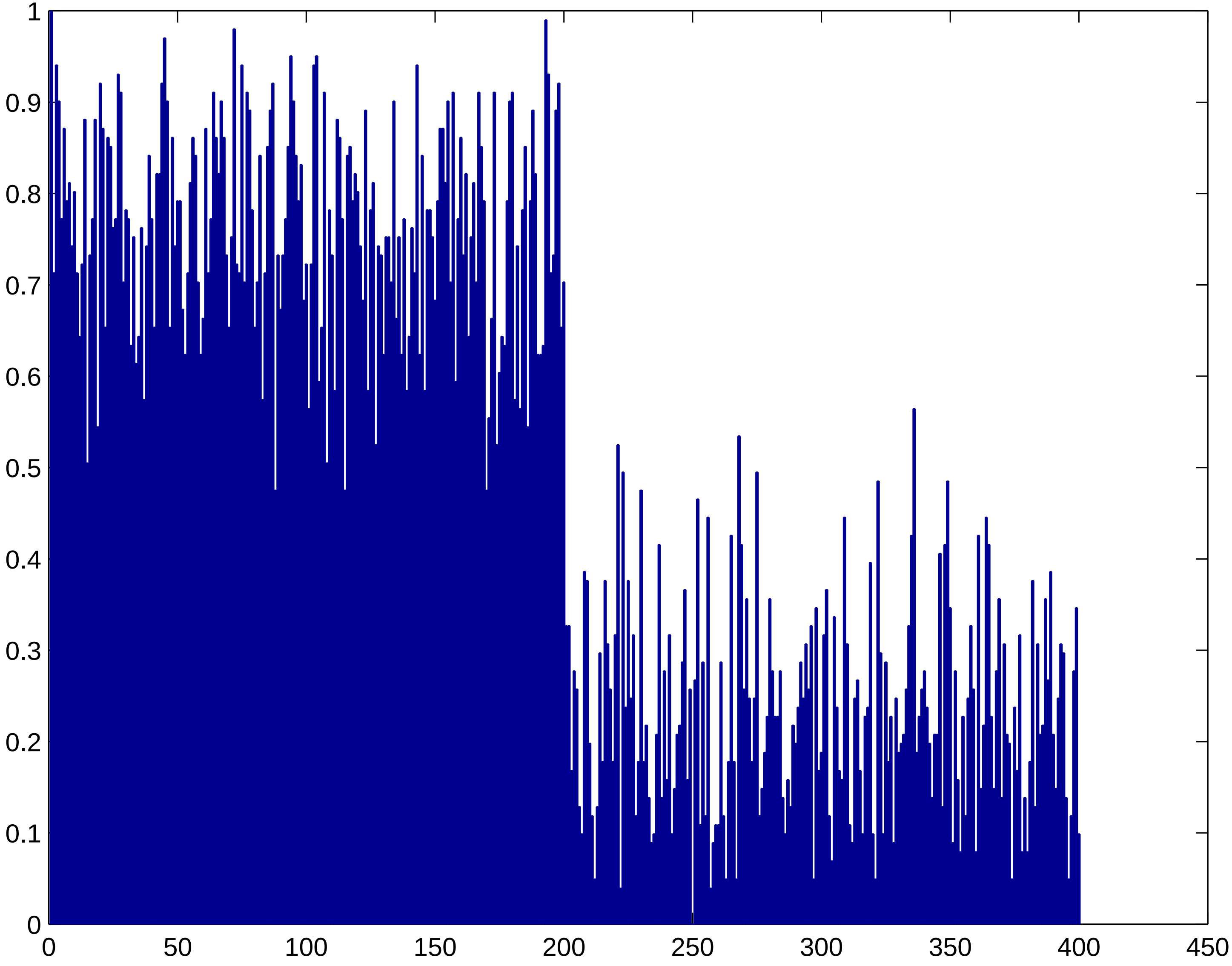
Max-Cut



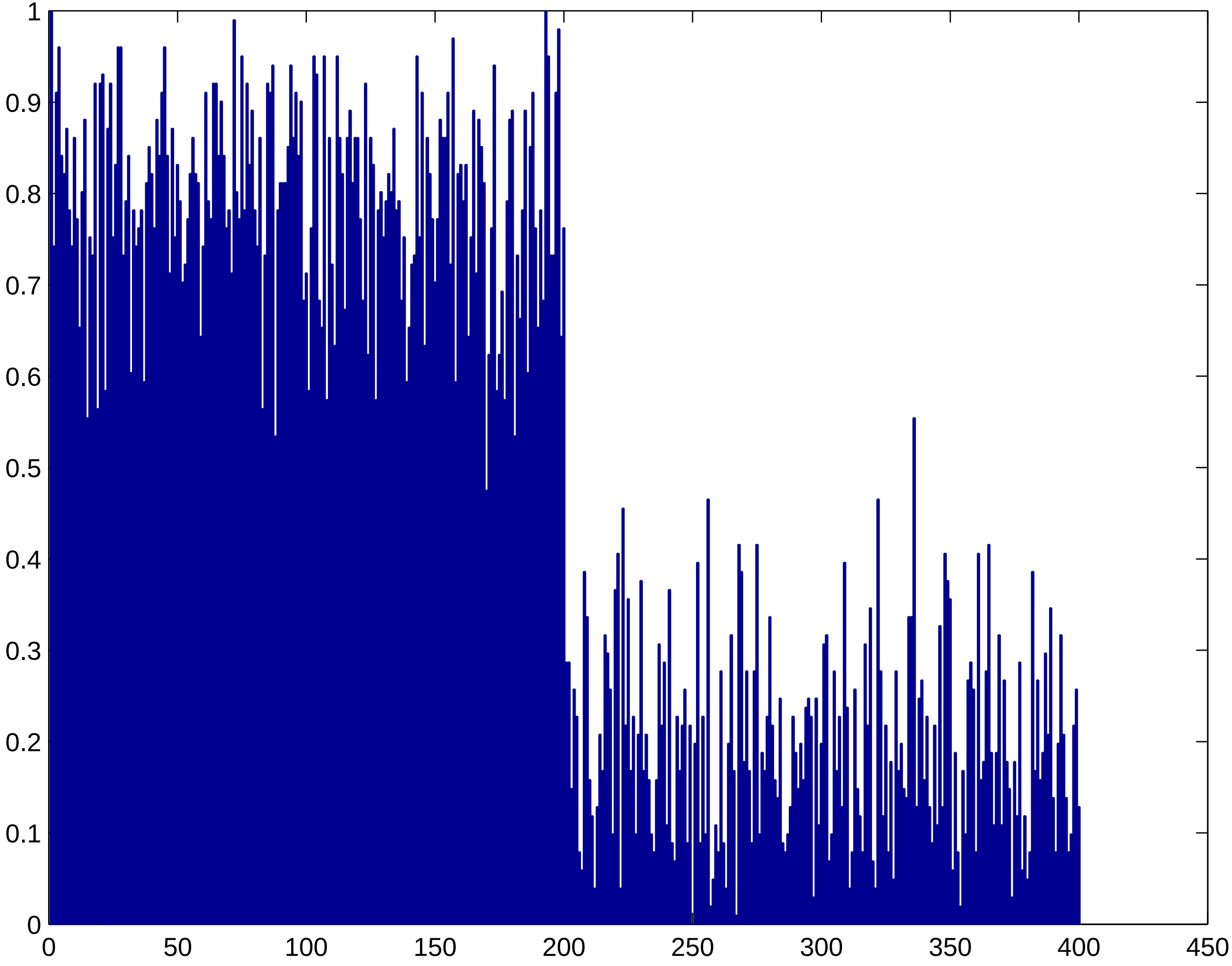
Max-Cut



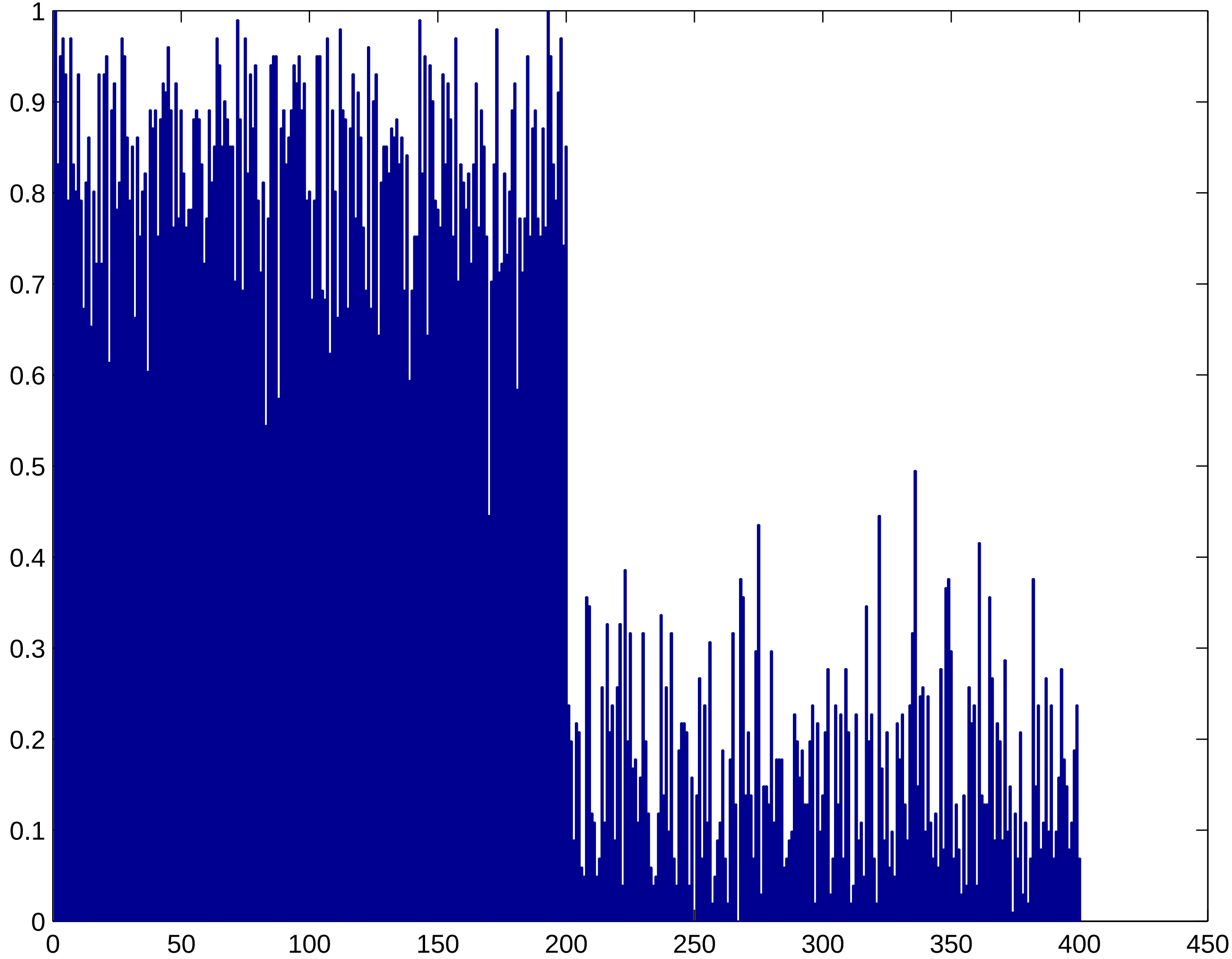
Max-Cut



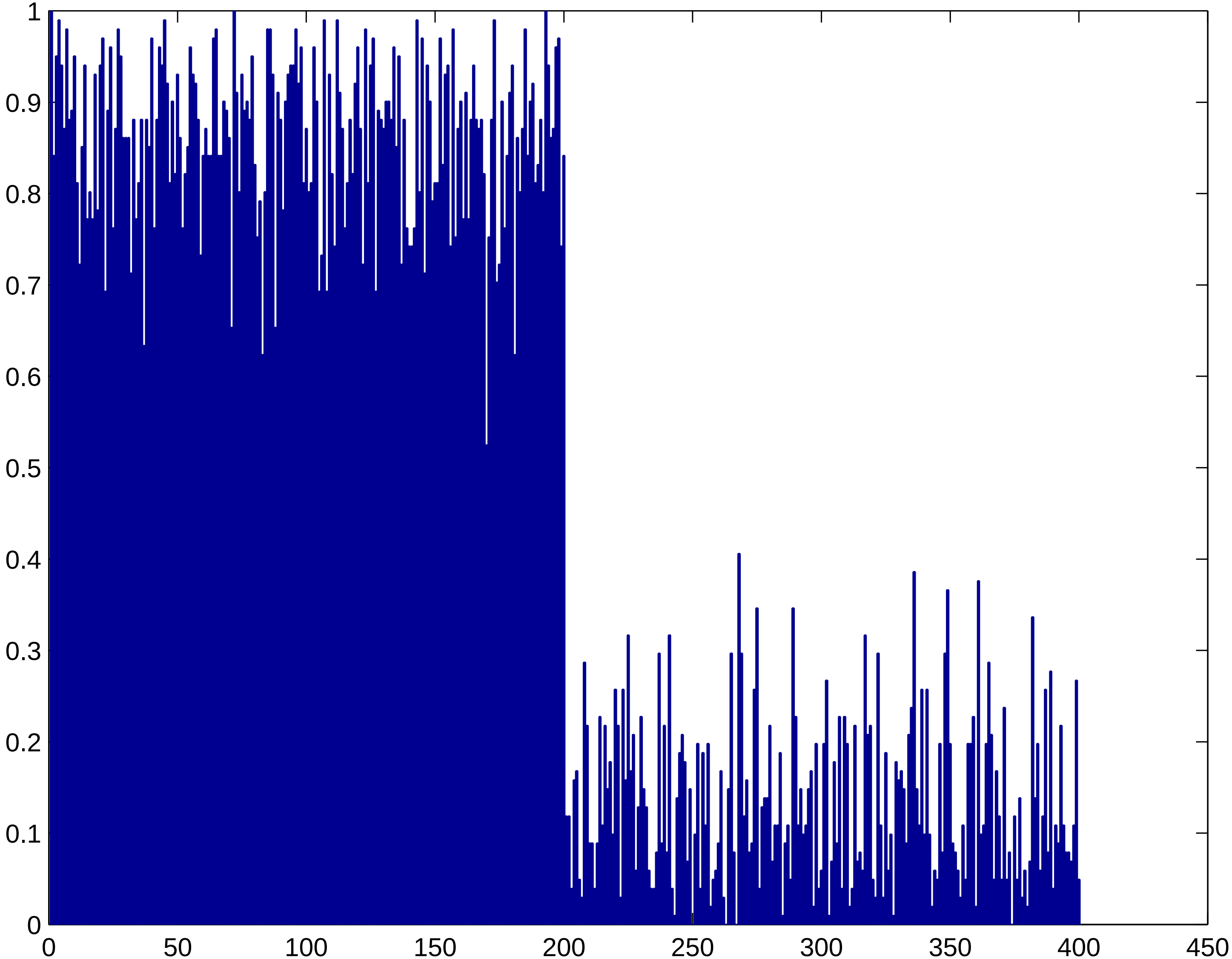
Max-Cut



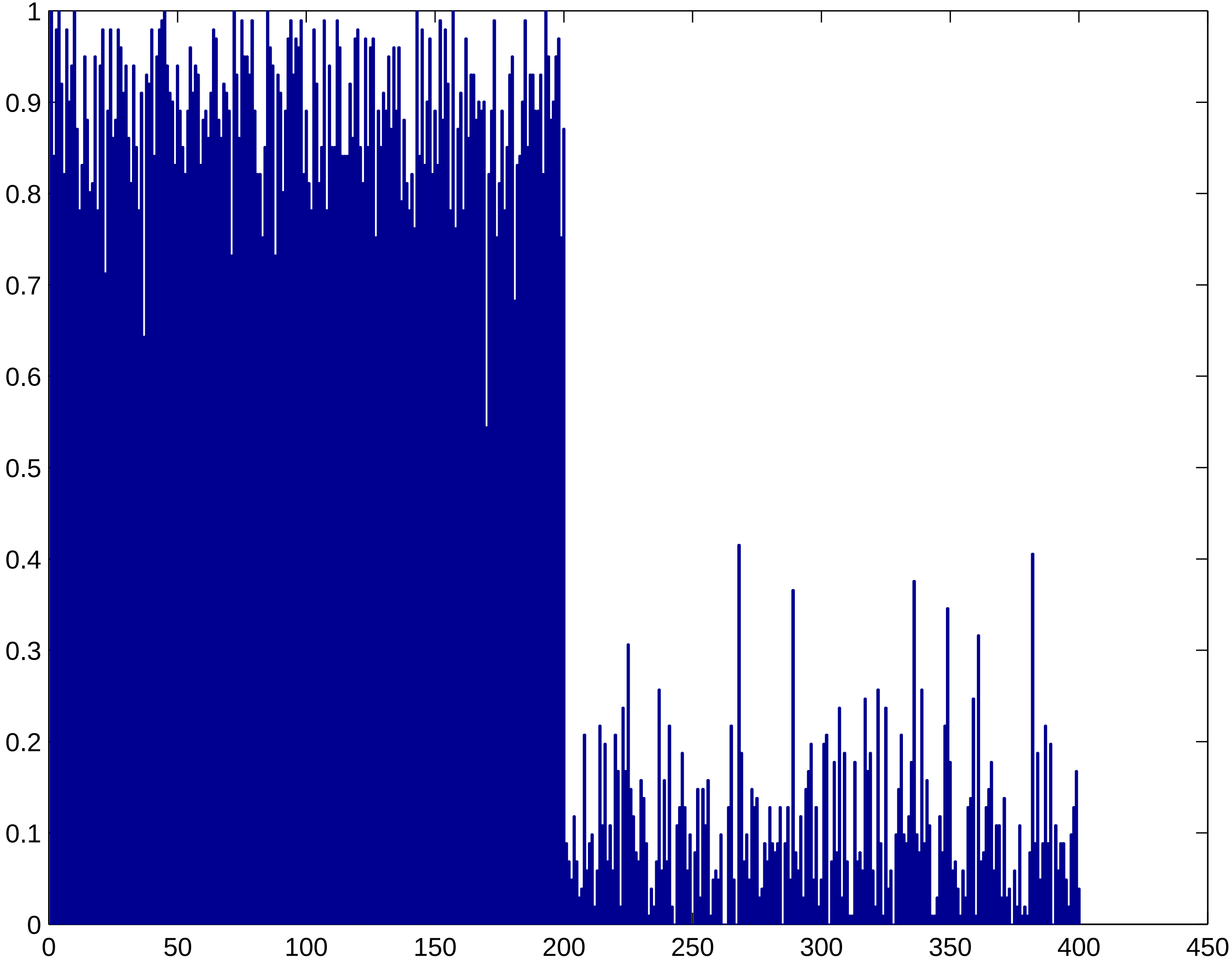
Max-Cut



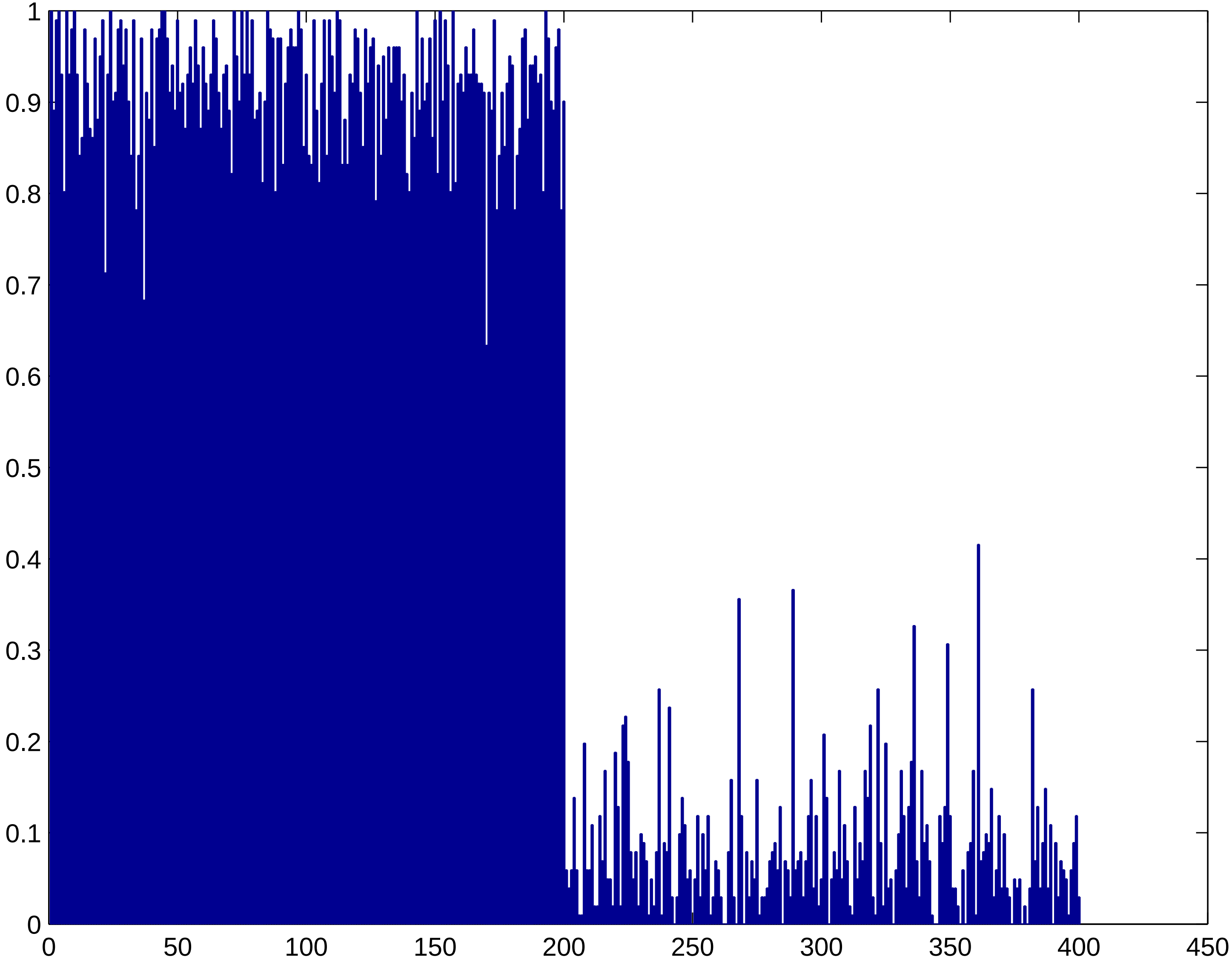
Max-Cut



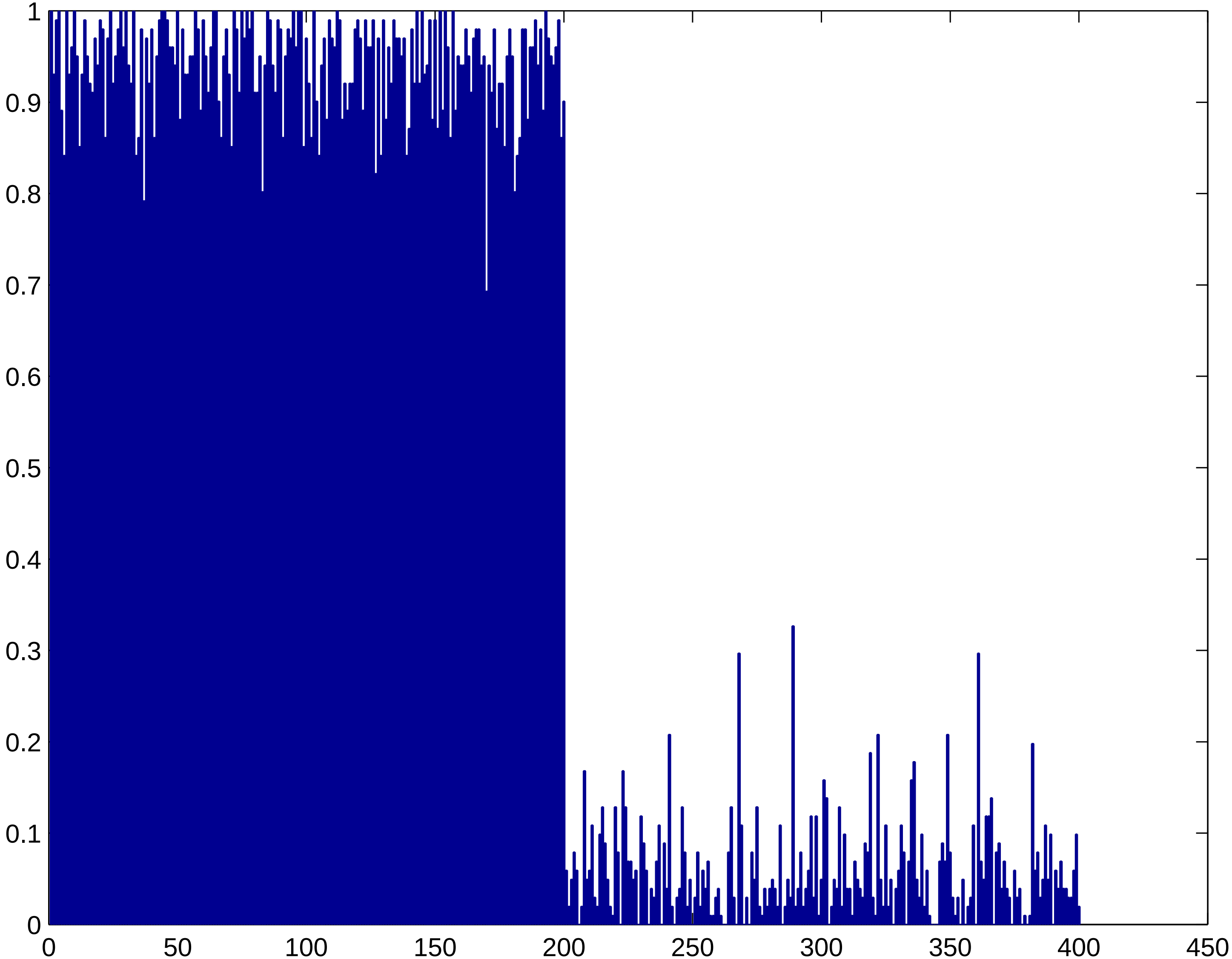
Max-Cut



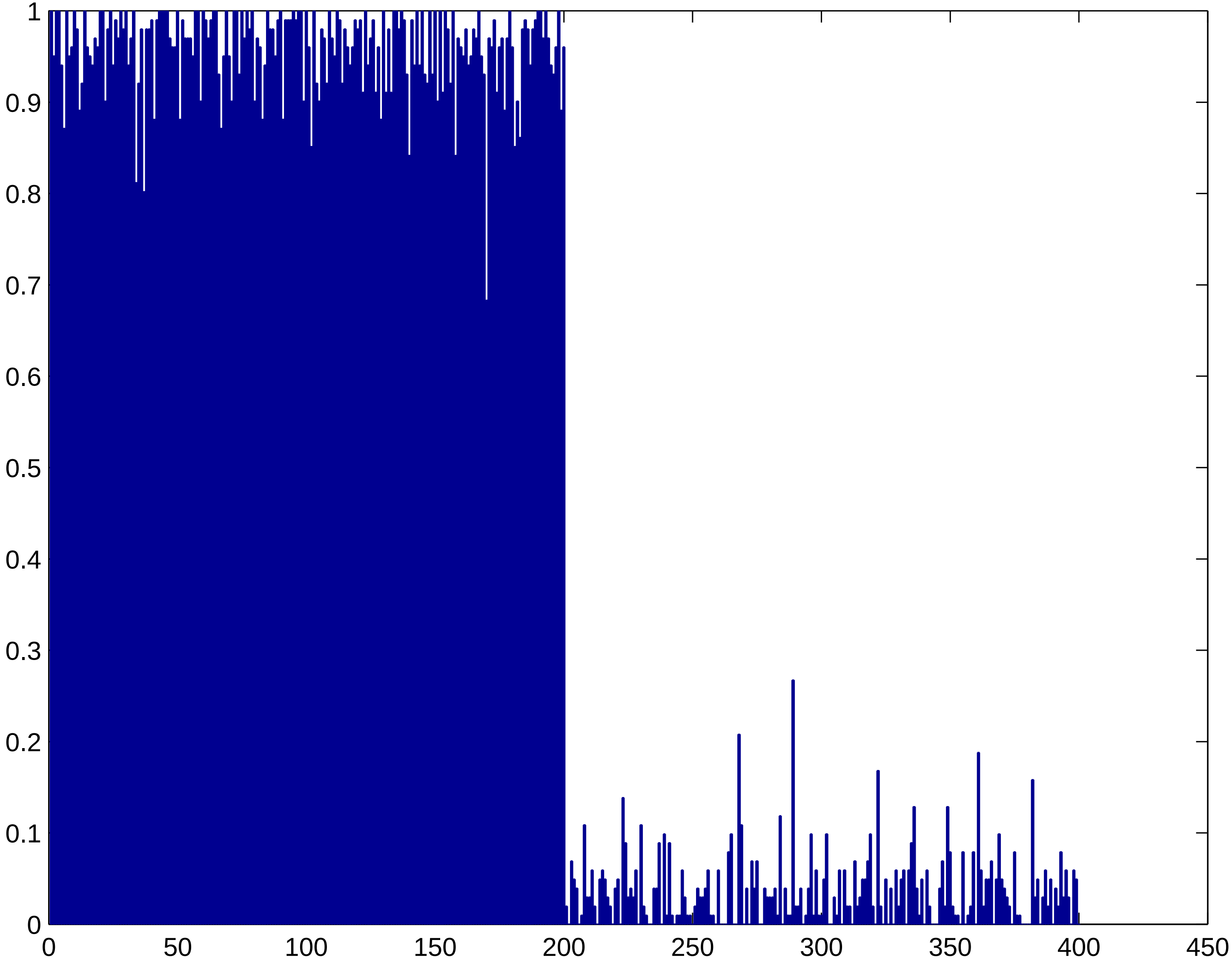
Max-Cut



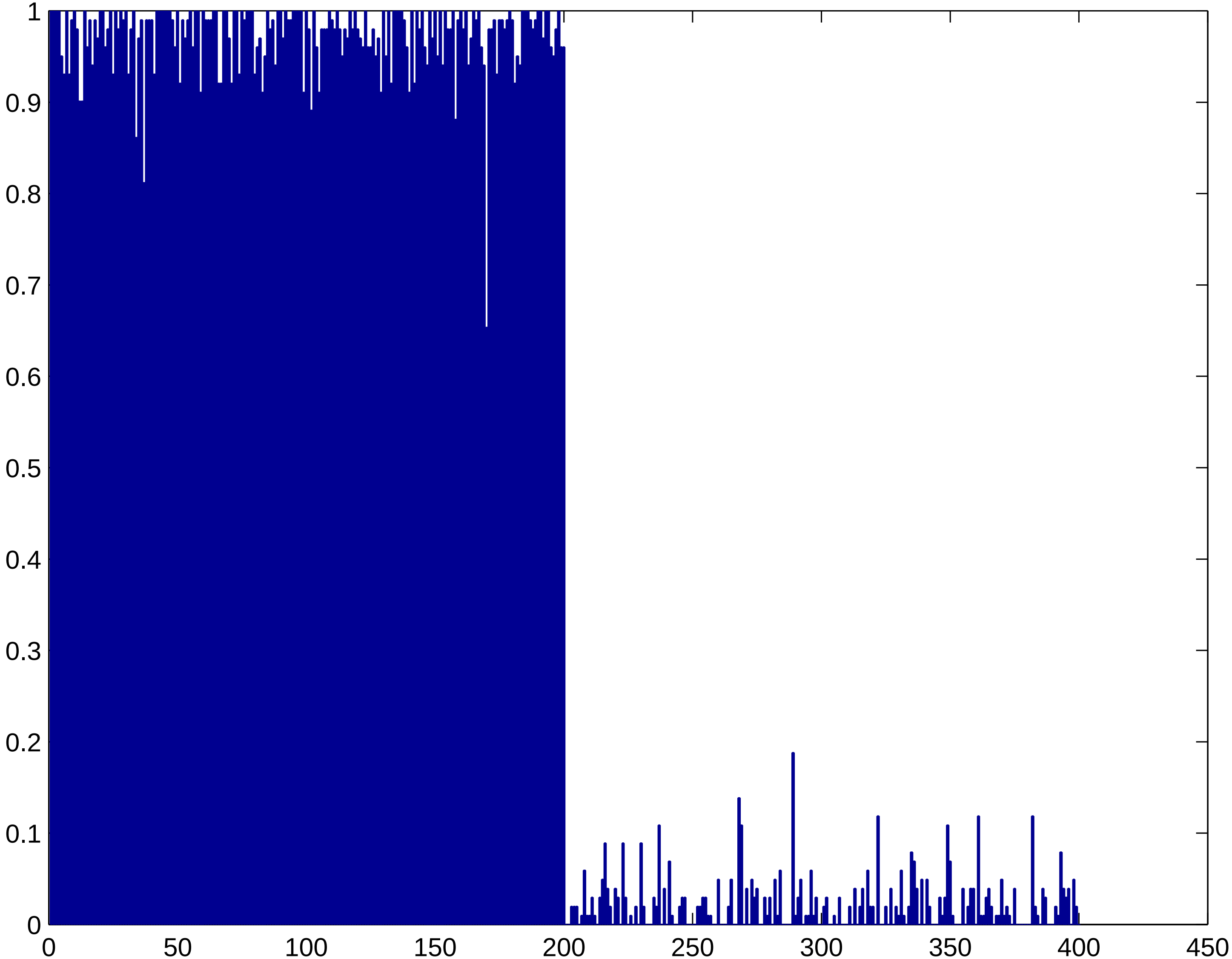
Max-Cut



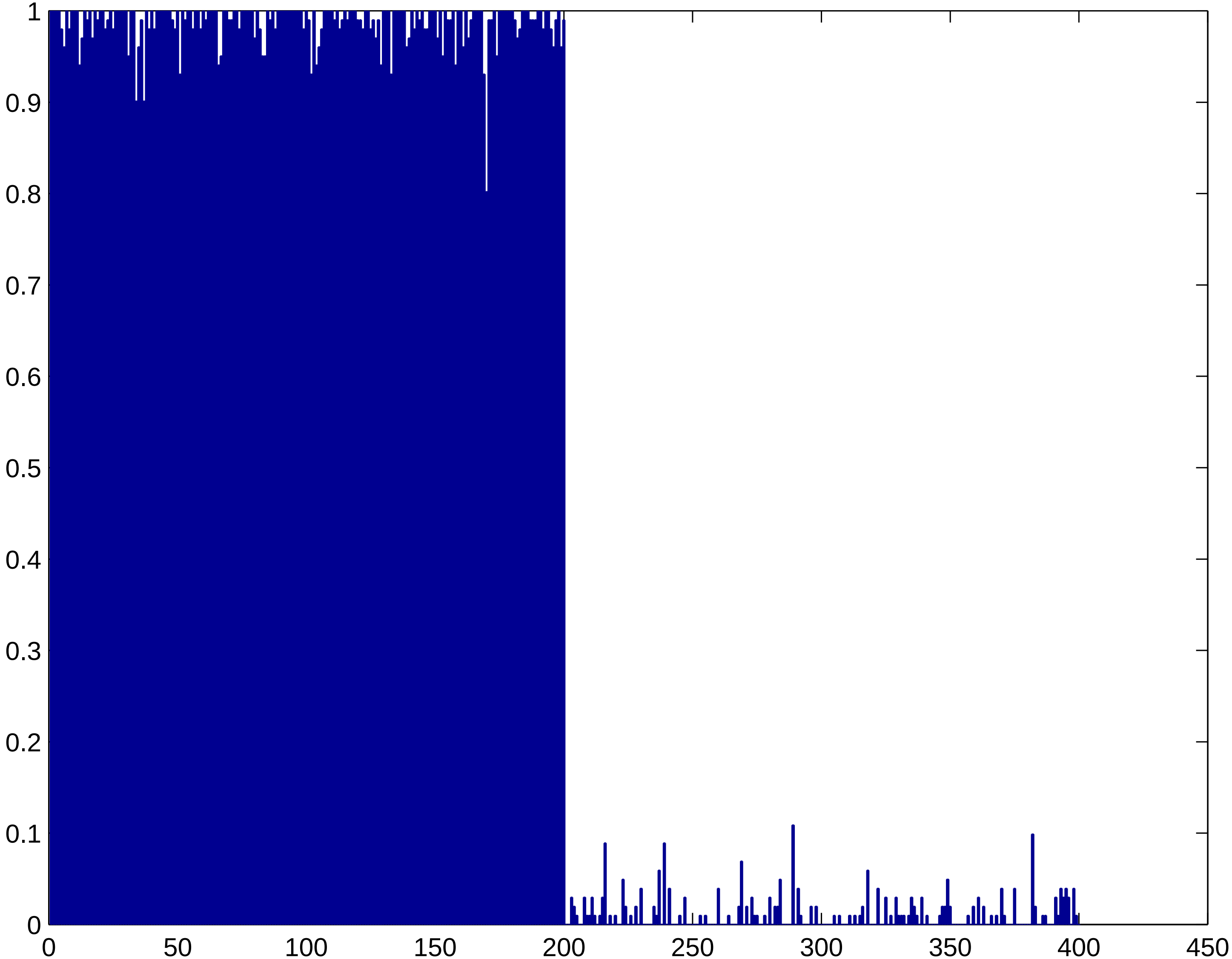
Max-Cut



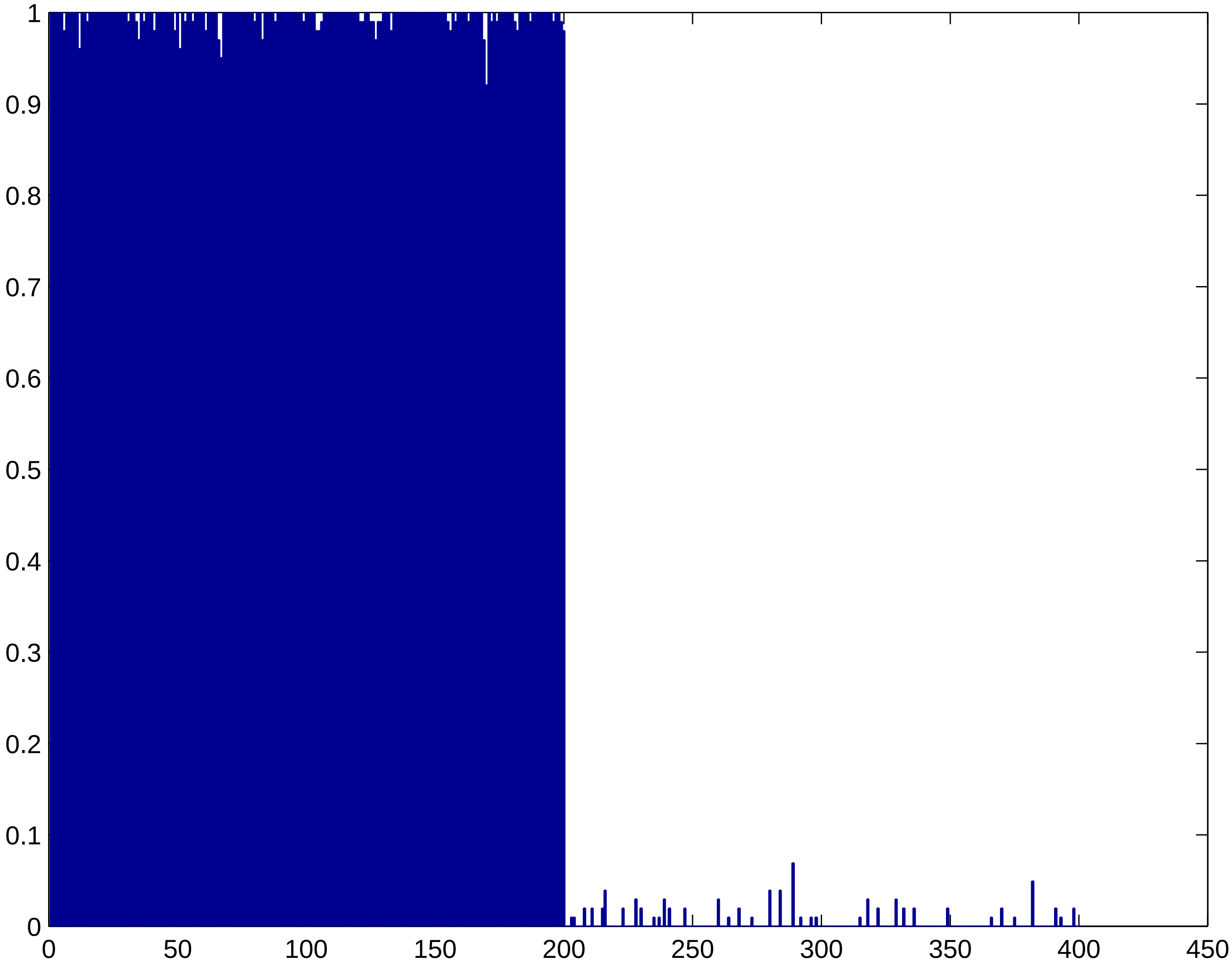
Max-Cut



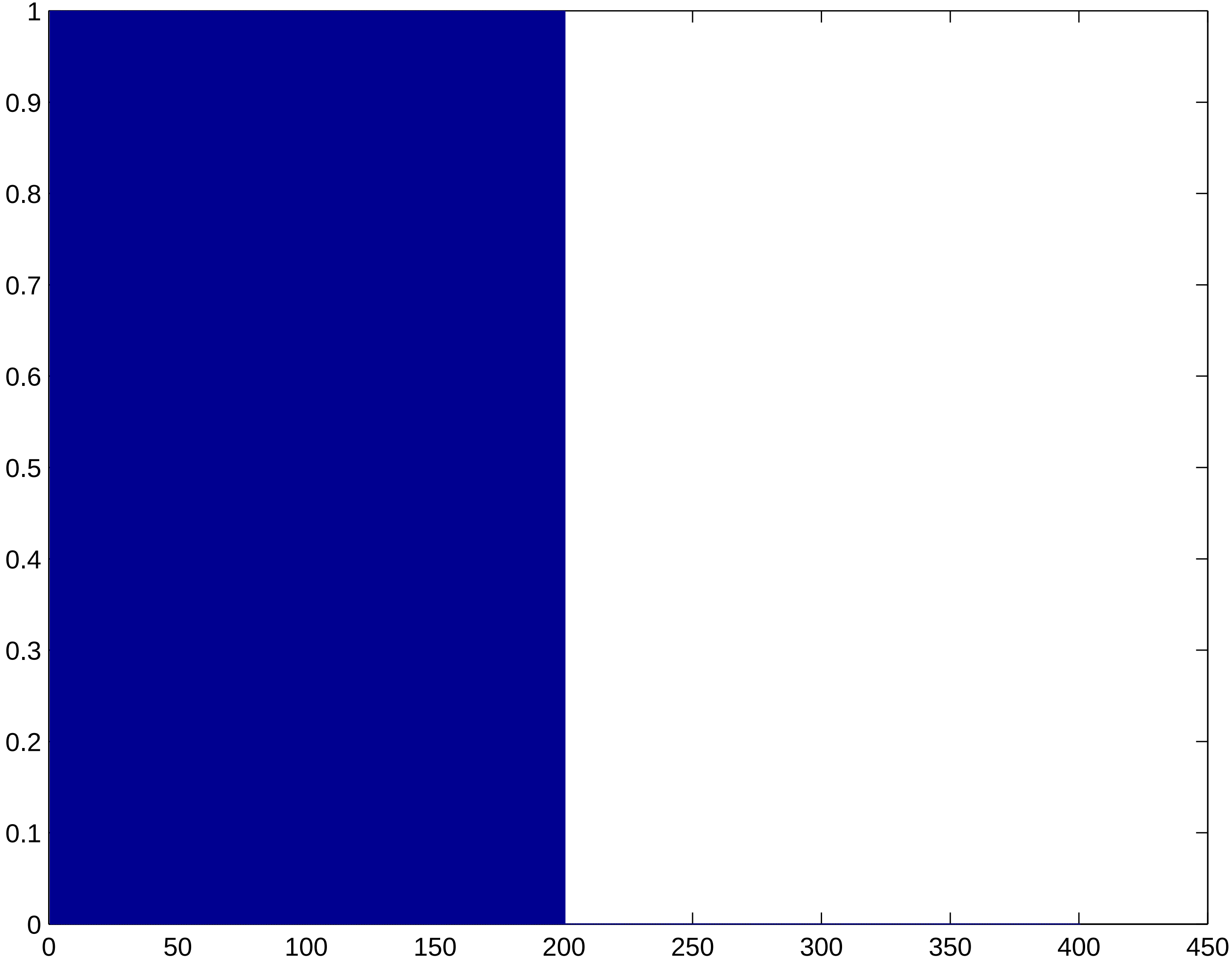
Max-Cut



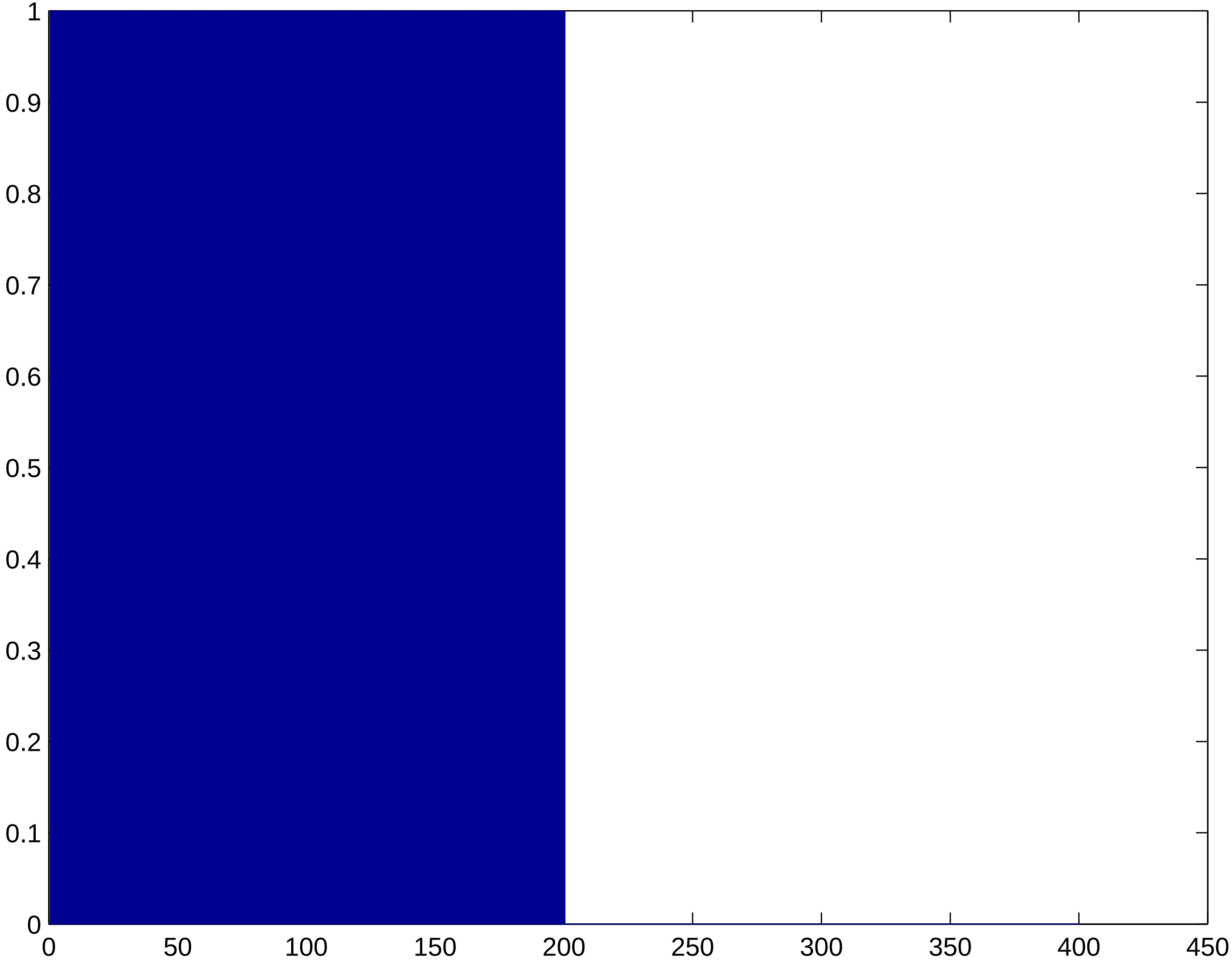
Max-Cut



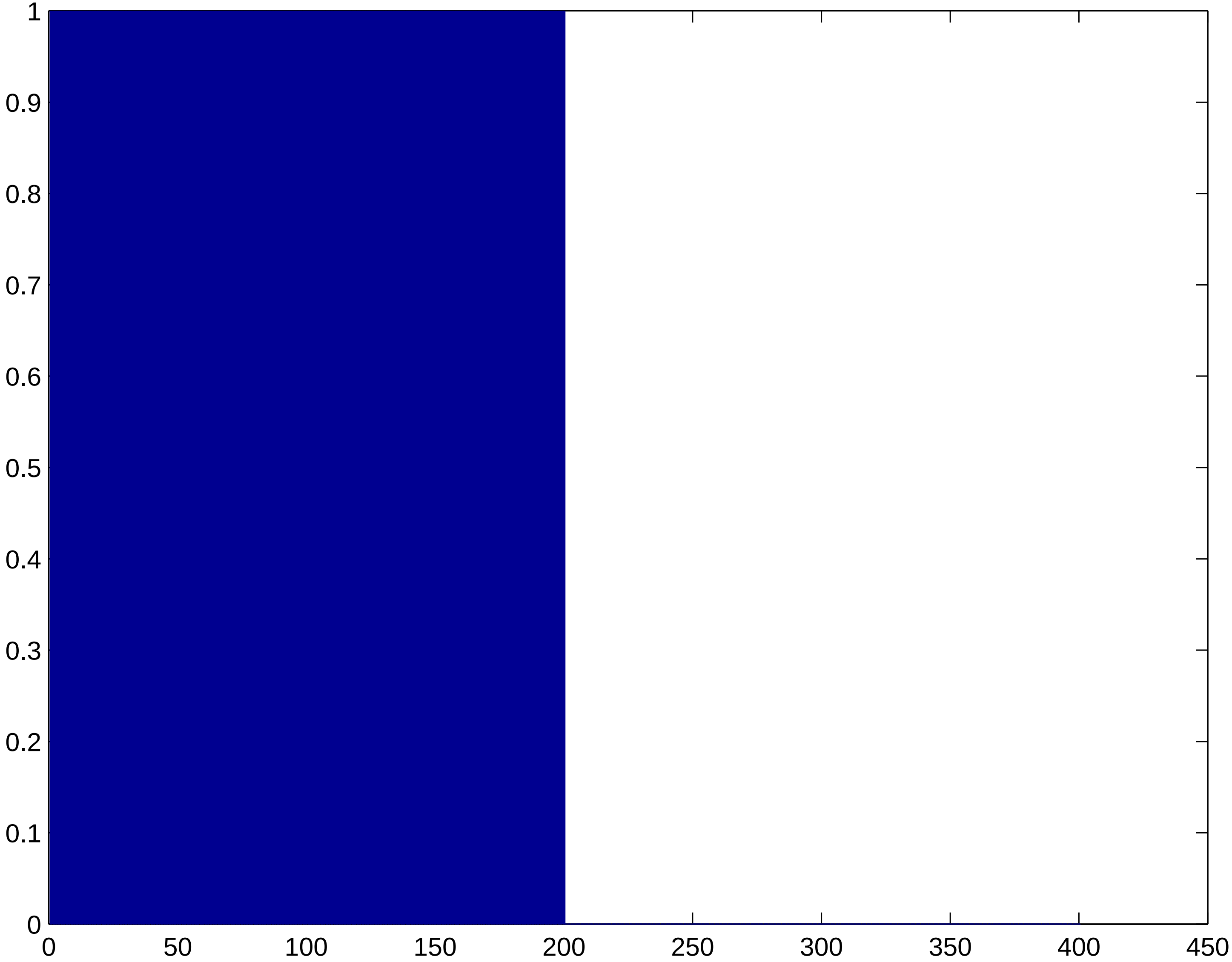
Max-Cut



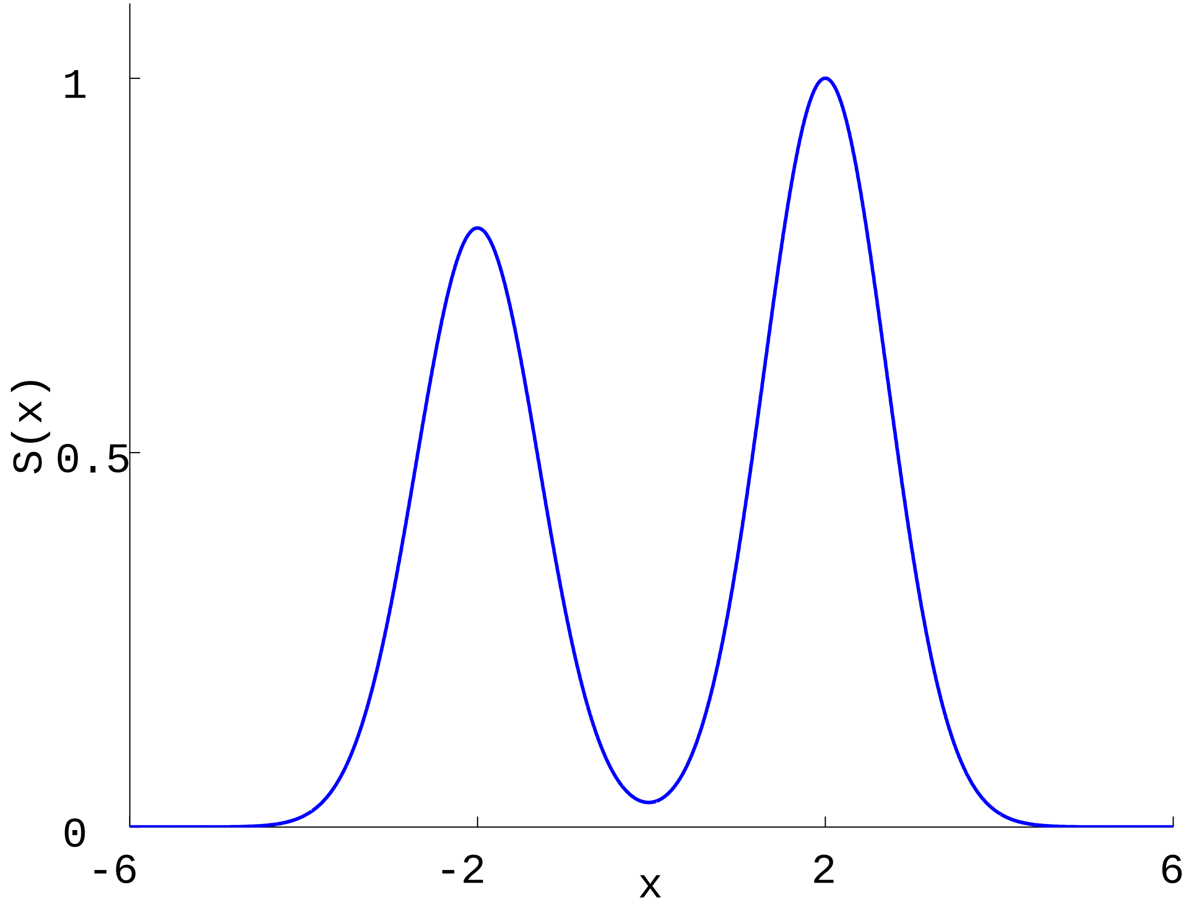
Max-Cut



Max-Cut



Example: Continuous Optimization

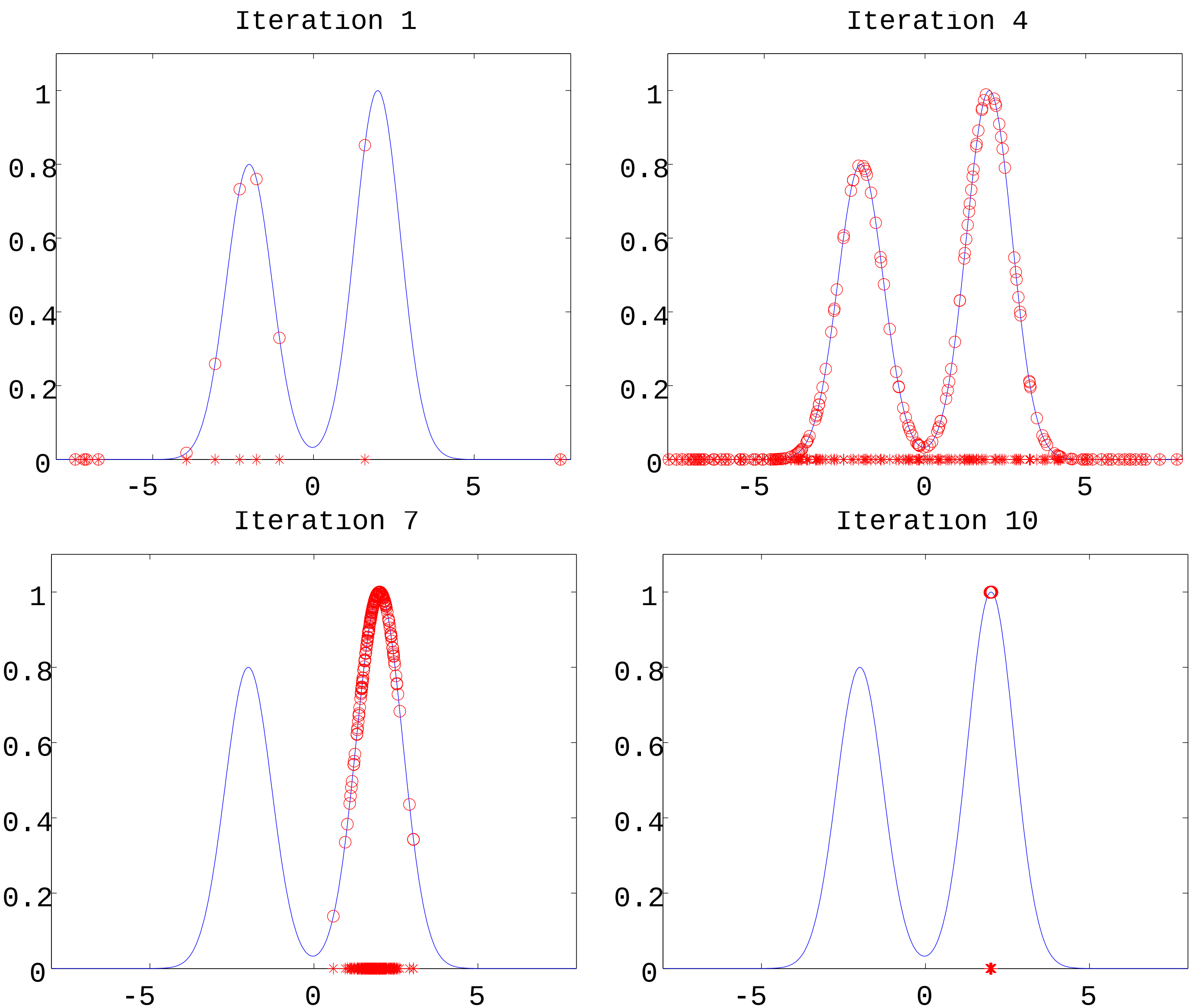


$$S(x) = \mathrm{e}^{-(x-2)^2} + 0.8\,\mathrm{e}^{-(x+2)^2}$$

Matlab Program

```
S = inline('exp(-(x-2).^2) + 0.8*exp(-(x+2).^2)');
mu = -10; sigma = 10; rho = 0.1; N = 100; eps = 1E-3;
t=0;  % iteration counter
while sigma > eps
  t = t+1;
  x = mu + sigma*randn(N,1);
  SX = S(x); % Compute the performance.
  sortSX = sortrows([x SX],2);
  mu = mean(sortSX((1-rho)*N:N,1));
  sigma = std(sortSX((1-rho)*N:N,1));
  fprintf('%g  %6.9f  %6.9f  %6.9f \n', t, S(mu),mu, sigma
```

Numerical Result



Cross-Entropy: Some Theory

Estimate $\ell := \mathbb{P}_{\boldsymbol{u}}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \geq \gamma\}}$ via

Cross-Entropy: Some Theory

Estimate $\ell := \mathbb{P}_{\boldsymbol{u}}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \geq \gamma\}}$ via

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \frac{f(\boldsymbol{X}_i; \boldsymbol{u})}{g(\boldsymbol{X}_i)} \; .$$

Cross-Entropy: Some Theory

Estimate $\ell := \mathbb{P}_{\boldsymbol{u}}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \geq \gamma\}}$ via

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \frac{f(\boldsymbol{X}_i; \boldsymbol{u})}{g(\boldsymbol{X}_i)} .$$

The best density (zero variance estimator!) is

$$g^*(\boldsymbol{x}) := \frac{I_{\{S(\boldsymbol{x}) \geq \gamma\}} f(\boldsymbol{x}; \boldsymbol{u})}{\ell} .$$

Cross-Entropy: Some Theory

Estimate $\ell := \mathbb{P}_{\boldsymbol{u}}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \geq \gamma\}}$ via

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \frac{f(\boldsymbol{X}_i; \boldsymbol{u})}{g(\boldsymbol{X}_i)} \, .$$

The best density (zero variance estimator!) is

$$g^*(\boldsymbol{x}) := \frac{I_{\{S(\boldsymbol{x}) \geq \gamma\}} f(\boldsymbol{x}; \boldsymbol{u})}{\ell} \, .$$

Problem: g^* depends on the unknown ℓ.

Cross-Entropy: Some Theory

Estimate $\ell := \mathbb{P}_{\boldsymbol{u}}(S(\boldsymbol{X}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \geq \gamma\}}$ via

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \frac{f(\boldsymbol{X}_i; \boldsymbol{u})}{g(\boldsymbol{X}_i)} .$$

The best density (zero variance estimator!) is

$$g^*(\boldsymbol{x}) := \frac{I_{\{S(\boldsymbol{x}) \geq \gamma\}} f(\boldsymbol{x}; \boldsymbol{u})}{\ell} .$$

Problem: g^* depends on the unknown ℓ.

Idea: choose $g = f(\cdot; \boldsymbol{v})$ such that the "distance" between the densities g^* and $f(\cdot; \boldsymbol{v})$ is minimal.

Idea: choose $g = f(\cdot;\boldsymbol{v})$ such that the "distance" between the densities g^* and $f(\cdot;\boldsymbol{v})$ is minimal.

The **Kullback-Leibler** or **cross-entropy** distance is defined as:

$$
\begin{aligned}
\mathcal{D}(g,h) &= \mathbb{E}_g \log \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \\
&= \int g(\boldsymbol{x}) \log g(\boldsymbol{x})\, d\boldsymbol{x} - \int g(\boldsymbol{x}) \log h(\boldsymbol{x})\, d\boldsymbol{x}\ .
\end{aligned}
$$

Idea: choose $g = f(\cdot; \boldsymbol{v})$ such that the "distance" between the densities g^* and $f(\cdot; \boldsymbol{v})$ is minimal.

The **Kullback-Leibler** or **cross-entropy** distance is defined as:

$$\mathcal{D}(g, h) = \mathbb{E}_g \log \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})}$$
$$= \int g(\boldsymbol{x}) \log g(\boldsymbol{x})\, d\boldsymbol{x} - \int g(\boldsymbol{x}) \log h(\boldsymbol{x})\, d\boldsymbol{x}\ .$$

Determine the optimal $\boldsymbol{v}^*$ from $\min_{\boldsymbol{v}} \mathcal{D}(g^*, f(\cdot; \boldsymbol{v}))$.

This is equivalent to solving

$$\max_{\boldsymbol{v}} \mathbb{E}_{\boldsymbol{u}} \, I_{\{S(\boldsymbol{X}) \geq \gamma\}} \log f(\boldsymbol{X}; \boldsymbol{v}) \ .$$

Using again IS, we can rewrite this as

$$\max_{\boldsymbol{v}} \ \mathbb{E}_{\boldsymbol{w}} \, I_{\{S(\boldsymbol{X}) \geq \gamma\}} \, W(\boldsymbol{X}; \boldsymbol{u}, \boldsymbol{w}) \log f(\boldsymbol{X}; \boldsymbol{v}),$$

for *any* reference parameter $\boldsymbol{w}$, where

$$W(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{w}) = \frac{f(\boldsymbol{x}; .\boldsymbol{u})}{f(\boldsymbol{x}; \boldsymbol{w})}$$

We may *estimate* the optimal solution $\boldsymbol{v}^*$ by solving the following stochastic counterpart:

$$\max_{\boldsymbol{v}} \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i)\geq\gamma\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \boldsymbol{w}) \log f(\boldsymbol{X}_i; \boldsymbol{v}) ,$$

where $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ is a random sample from $f(\cdot; \boldsymbol{w})$. Alternatively, solve:

$$\frac{1}{N} \sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i)\geq\gamma\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \boldsymbol{w}) \nabla \log f(\boldsymbol{X}_i; \boldsymbol{v}) = \boldsymbol{0},$$

where the gradient is with respect to $\boldsymbol{v}$.

- The solution to the CE program can often be calculated analytically.

- The solution to the CE program can often be calculated analytically.
- Note that the CE program is useful only when under $\boldsymbol{w}$ the event $\{S(\boldsymbol{X}) \geq \gamma\}$ is not too rare, say $\geq 10^{-5}$.

- The solution to the CE program can often be calculated analytically.
- Note that the CE program is useful only when under $\boldsymbol{w}$ the event $\{S(\boldsymbol{X}) \geq \gamma\}$ is not too rare, say $\geq 10^{-5}$.

Question: how to choose $\boldsymbol{w}$ so that this is indeed the case?

- The solution to the CE program can often be calculated analytically.
- Note that the CE program is useful only when under $\boldsymbol{w}$ the event $\{S(\boldsymbol{X}) \geq \gamma\}$ is not too rare, say $\geq 10^{-5}$.

Question: how to choose $\boldsymbol{w}$ so that this is indeed the case?

Answer: use a multi-level approach.

- The solution to the CE program can often be calculated analytically.
- Note that the CE program is useful only when under $\boldsymbol{w}$ the event $\{S(\boldsymbol{X}) \geq \gamma\}$ is not too rare, say $\geq 10^{-5}$.

Question: how to choose $\boldsymbol{w}$ so that this is indeed the case?

Answer: use a multi-level approach.

Introduce a sequence of reference parameters $\{\boldsymbol{v}_t,\ t \geq 0\}$ and a sequence of levels $\{\gamma_t,\ t \geq 1\}$, and iterate in both γ_t and $\boldsymbol{v}_t$.

Toy example 1 (continued)

Recall

$$f(\boldsymbol{x};\boldsymbol{v}) = \exp\left(-\sum_{j=1}^{5}\frac{x_j}{v_j}\right)\prod_{j=1}^{5}\frac{1}{v_j}\,.$$

The optimal $\boldsymbol{v}$ follows from the system of equations

$$\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i)\geq\gamma\}}\, W(\boldsymbol{X}_i;\boldsymbol{u},\boldsymbol{w})\,\nabla\log f(\boldsymbol{X}_i;\boldsymbol{v}) = \boldsymbol{0}.$$

Since

$$\frac{\partial}{\partial v_j}\log f(\boldsymbol{x};\boldsymbol{v}) = \frac{x_j}{v_j^2} - \frac{1}{v_j},$$

we have for the jth equation

$$\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} \, W(\boldsymbol{X}_i; \boldsymbol{u}, \boldsymbol{w}) \, \left(\frac{X_{ij}}{v_j^2} - \frac{1}{v_j} \right) = 0 \; ,$$

whence,

$$v_j = \frac{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \boldsymbol{w}) \, X_{ij}}{\sum_{i=1}^{N} I_{\{S(\boldsymbol{X}_i) \geq \gamma\}} W(\boldsymbol{X}_i; \boldsymbol{u}, \boldsymbol{w})} \; ,$$

which leads to the updating formula in step 3 of the Algorithm.

Further research

- Multi-extremal constrained continuous optimization.

Further research

- Multi-extremal constrained continuous optimization.
- Noisy optimization.

Further research

- Multi-extremal constrained continuous optimization.
- Noisy optimization.
- Simulation with heavy tail distributions.

Further research

- Multi-extremal constrained continuous optimization.
- Noisy optimization.
- Simulation with heavy tail distributions.
- Incorporating MaxEnt (MinxEnt), e.g. MCE

Further research

- Multi-extremal constrained continuous optimization.
- Noisy optimization.
- Simulation with heavy tail distributions.
- Incorporating MaxEnt (MinxEnt), e.g. MCE
- Multi-actor games

Further research

- Multi-extremal constrained continuous optimization.
- Noisy optimization.
- Simulation with heavy tail distributions.
- Incorporating MaxEnt (MinxEnt), e.g. MCE
- Multi-actor games
- Convergence of CE algorithm.