Randomized Algorithms

Randomized Rounding

- Brief Introduction to Linear Programming and Its Usage in Combinatorial Optimization
- Randomized Rounding for Cut Problems
- Randomized Rounding for Covering Problems
- Randomized Rounding for Satisfiability Problems
- **Randomized Rounding and Semi-definite Programming**

Approximate Sampling and Counting

- ...

MAX-CUT and MAX-2SAT

MAX-CUT

Input: graph $G = (V, E)$, $w : E \to \mathbb{N}$
Output: a cut $(S, \bar{S})$, $S \subset V$, with maximum total weight of edges crossing the cut.

MAX-2SAT

Input: a 2-CNF formula φ, n variables, m clauses, clause j is "weighted" with $w_j \in \mathbb{N}$
Output: a truth assignment maximizing the total weight of satisfied clauses

QCQP and Strict QCQP

Definition (Quadratically Constrained Quadratic Program – QCQP)

Optimize a quadratic function subject to quadratic constraints.

Definition (Strict QCQP)

Optimize a quadratic function subject to quadratic constraints. The monomials in the objective function and in the constraints are all of degrees 2 or 0.

MAX-2SAT as a QCQP

Think: $y_i = 1/0$ iff $x_i = $ TRUE/FALSE

Example:

$$\varphi = \underbrace{(\bar{x}_1 \vee x_2)}_{w_1} \wedge \underbrace{(x_3)}_{w_2} \wedge \underbrace{(x_1 \vee \bar{x}_3)}_{w_3}$$

$$\begin{array}{ll} \max & w_1(1 - y_1(1-y_2)) + w_2(1-(1-y_3)) + w_3(1-(1-y_1)y_3) \\ \text{subject to} & y_i^2 = y_i, \quad \forall i \\ & y_i \in \mathbb{R}, \quad \forall i \end{array}$$

MAX-CUT as a Strict QCQP

Think: $x_i = 1$ or -1 iff vertex $i \in$ or $\notin S$

$$\begin{array}{lll} \max & \displaystyle\frac{1}{2}\sum_{ij \in E} w_{ij}(1 - x_i x_j) & \\ \text{subject to} & x_i^2 = 1, \quad \forall i \in V & x_i \in \mathbb{R}, \quad \forall i \in V \end{array}$$

Vector Program

Definition (Vector Program)

Variables: n vectors $\mathbf{v}_1, \dots, \mathbf{v}_n$ in $\mathbb{R}^n$
Objective and Constraints: linear in the inner products $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$

The general form of a vector program is

$$\begin{array}{lll} \max & \displaystyle\sum_{1 \le i,j \le n} c_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle & \\ \text{subject to} & \displaystyle\sum_{1 \le i,j \le n} a_{ij}^{(k)} \langle \mathbf{v}_i, \mathbf{v}_j \rangle = b_k & 1 \le k \le m \\ & \mathbf{v}_i \in \mathbb{R}^n & \forall i \in [n] \end{array}$$

From Strict QCQP to Vector Program

From a Strict QCQP, we easily get a “relaxed” vector program by replacing each variable with a vector, and a product of two variables with the inner product of the corresponding vectors

$$
\begin{aligned}
\max \quad & \frac{1}{2} \sum_{ij \in E} w_{ij}(1 - \langle \mathbf{v}_i, \mathbf{v}_j \rangle) \\
\text{subject to} \quad & \langle \mathbf{v}_i, \mathbf{v}_i \rangle = 1, \quad \forall i \in V \\
& \mathbf{v}_i \in \mathbb{R}^n, \quad \forall i \in V
\end{aligned}
$$

Why Vector Programs?

- A vector program (VP) can be solved to within $\pm\epsilon$ of optimality in time polynomial in the input size and $\log(1/\epsilon)$
- Reason: vector program is equivalent to semidefinite program
- After getting a (near) optimal solution $\mathbf{v}_1^*, \dots, \mathbf{v}_n^*$ to the vector program, we can (randomly) "round" back to a feasible solution $\mathbf{x}^A$ of the original optimization problem.
- Sometime, a problem can be relaxed directly to a semidefinite program (SDP)
- Thus, need to know SDP and its equivalence with VP

Positive Semidefinite Matrices

Definition/Characterization: given a real and symmetric $n \times n$ matrix $\mathbf{A}$, the following are equivalent

- $\mathbf{A}$ is positive semidefinite
- $\mathbf{x}^T \mathbf{A} \mathbf{x} \geq 0$, for all $\mathbf{x} \in \mathbb{R}^n$
- all eigenvalues of $\mathbf{A}$ are non-negative
- $\mathbf{A} = \mathbf{W}^T \mathbf{W}$ for some real matrix $\mathbf{W}$ (not necessarily square)
- $\mathbf{A}$ is a nonnegative linear combination of matrices of the type $\mathbf{x}\mathbf{x}^T$
- the determinant of all symmetric minor of $\mathbf{A}$ is non-negative

More notations

- Use $\mathbf{A} \in \mathbb{R}^{n \times n}$ to denote "$\mathbf{A}$ is an $n \times n$ real matrix"
- Use $\mathbf{A} \succeq 0$ to denote "$\mathbf{A}$ is positive semidefinite" (PSD)
- Use S_n to denote the set of all symmetric matrices in $\mathbb{R}^{n \times n}$
- For $\mathbf{C}, \mathbf{X} \in S_n$, the Frobenius inner product of them is

$$\mathbf{C} \bullet \mathbf{X} := \operatorname{tr} \mathbf{C}^T \mathbf{X} = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}.$$

Definition (Semidefinite Program – SDP)

Optimizing a linear function of the x_{ij} subject to linear constraints on them, and subject to $\mathbf{X} = (x_{ij}) \succeq 0$

In particular, let $\mathbf{C}, \mathbf{A}_1, \dots, \mathbf{A}_m \in S_n$, and $b_1, \dots, b_m \in \mathbb{R}$. The following is a general SDP:

$$\begin{array}{rcl} \max & \mathbf{C} \bullet \mathbf{X} & \\ \text{subject to} & \mathbf{A}_i \bullet \mathbf{X} = b_i & 1 \leq i \leq m \\ & \mathbf{X} \succeq 0 & \end{array}$$

If all $\mathbf{C}, \mathbf{A}_1, \dots, \mathbf{A}_m$ are diagonal matrices, then the SDP is an LP.

Solving Semidefinite Programs

Theorem

A semidefinite program can be solved to within an additive factor ϵ of optimality in time polynomial in n and $\log(1/\epsilon)$

Two basic methods:

- Ellipsoid
- Interior point

Vector Program $\equiv$ Semidefinite Program

Vector Program

$$\begin{aligned} \max \quad & \sum_{1 \le i,j \le n} c_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \\ \text{subject to} \quad & \sum_{1 \le i,j \le n} a_{ij}^{(k)} \langle \mathbf{v}_i, \mathbf{v}_j \rangle = b_k \quad 1 \le k \le m \\ & \mathbf{v}_i \in \mathbb{R}^n \qquad \forall i \in [n] \end{aligned} \tag{1}$$

Semidefinite Program

$$\begin{aligned} \max \quad & \mathbf{C} \bullet \mathbf{X} \\ \text{subject to} \quad & \mathbf{A}_k \bullet \mathbf{X} = b_k \quad 1 \le k \le m \\ & \mathbf{X} \succeq 0 \end{aligned} \tag{2}$$

- From (1) to (2), set $x_{ij} = \langle \mathbf{v}_i, \mathbf{v}_j \rangle$
- From (2) to (1), write $\mathbf{X} = \mathbf{W}^T \mathbf{W}$ (possible since $\mathbf{X}$ is PSD), then set $\mathbf{v}_i$ to be the ith column of $\mathbf{W}$

The Vector Program (i.e. the SDP) for MAX-CUT

$$
\begin{aligned}
\max \quad & \frac{1}{2}\sum_{ij\in E} w_{ij}(1-\langle \mathbf{v}_i, \mathbf{v}_j\rangle) \\
\text{subject to} \quad & \langle \mathbf{v}_i, \mathbf{v}_i\rangle = 1, \quad \forall i \in V \\
& \mathbf{v}_i \in \mathbb{R}^n, \quad \forall i \in V
\end{aligned}
$$

Intuitions:

- A feasible solution maps each vertex to a point on the n-dimensional unit sphere $\mathbb{S}_{n-1}$
- Let θ_{ij} be the angle between $\mathbf{v}_i, \mathbf{v}_j$, the contribution of edge ij is $\frac{1}{2}(1-\langle \mathbf{v}_i, \mathbf{v}_j\rangle) = \frac{1}{2}(1-\cos\theta_{ij})$
- The wider separated the $\mathbf{v}_i, \mathbf{v}_j$, the larger the contribution
- A hyperplane (through the origin) will likely separate $\mathbf{v}_i, \mathbf{v}_j$ if they are widely separated
- Thus, pick a random hyperplane and "use" it as a cut

Randomized Rounding for MAX-CUT

1. Let $\mathbf{v}_1^*, \dots, \mathbf{v}_n^*$ be a (near) optimal solution to the vector program
2. Choose a unit vector $\mathbf{r}$ uniformly at random from the unit sphere $\mathbb{S}_{n-1}$ (think of it as the normal vector of the random hyperplane)
3. Output the cut $(S, \bar{S})$, where

$$\begin{aligned} S &= \{i \in V \mid \langle \mathbf{v}_i^*, \mathbf{r} \rangle \geq 0\} \\ \bar{S} &= \{i \in V \mid \langle \mathbf{v}_i^*, \mathbf{r} \rangle < 0\} \end{aligned}$$

Analysis

- For any edge $ij \in E$,

$$\text{Prob}[\mathbf{v}_i, \mathbf{v}_j \text{ are separated by } \mathbf{r}] = \frac{\theta_{ij}}{\pi} = \frac{\arccos(\langle \mathbf{v}_i, \mathbf{v}_j \rangle)}{\pi}$$

- Expected cut capacity is thus

$$\begin{aligned}
& \sum_{ij \in E} w_{ij} \frac{\arccos(\langle \mathbf{v}_i, \mathbf{v}_j \rangle)}{\pi} \\
= & \sum_{ij \in E} \left(\frac{\frac{\arccos(\langle \mathbf{v}_i, \mathbf{v}_j \rangle)}{\pi}}{\frac{1-\langle \mathbf{v}_i, \mathbf{v}_j \rangle}{2}} \right) w_{ij} \left(\frac{1 - \langle \mathbf{v}_i, \mathbf{v}_j \rangle}{2} \right) \\
\geq & \min_{x \in [-1,1]} \left(\frac{\frac{\arccos(x)}{\pi}}{\frac{1-x}{2}} \right) \cdot \text{OPT}(\text{Vector Program}) \\
\geq & \ 0.87856 \cdot \text{MAX-CUT CAPACITY}
\end{aligned}$$

How to choose $\mathbf{r}$ uniformly on the sphere?

- What do we mean by "uniform on the sphere anyway?
  - The uniform distribution of a bounded set $B \subset \mathbb{R}^k$ is the distribution whose density is

$$f(x_1, \dots, x_k) = \begin{cases} \frac{1}{V} & \mathbf{x} \in B \\ 0 & \text{otherwise} \end{cases}$$

    where V is the k-dimensional volume (or *Lebesgue measure*) of B.
- Consider $\mathbb{S}_1$, the 2-dimensional circle. One way to pick $\mathbf{r}$ uniformly is to pick $\theta \in [0, 2\pi]$ uniformly at random.
- This is a continuous distribution, which we have not really talked about

PTCF: Continuous Random Variable

- A r.v. X taking on uncountably many possible values is a *continuous random variable* if there exists a function $f : \mathbb{R} \to \mathbb{R}$, having the property that for every $B \subseteq \mathbb{R}$:

$$\text{Prob}[X \in B] = \int_B f(x)dx$$

- f is called the *(probability) density function* (PDF) of X. We must have

$$1 = \text{Prob}[X \in (-\infty, \infty)] = \int_{-\infty}^{\infty} f(x)dx.$$

- The *(cumulative) distribution function* (CDF) $F(\cdot)$ of X is defined by

$$F(a) = \text{Prob}\big[X \in (-\infty, a]\big] = \int_{-\infty}^{a} f(x)dx.$$

- Note that

$$\frac{d}{da}F(a) = f(a)$$

PTCF: Continuous Uniform Distribution

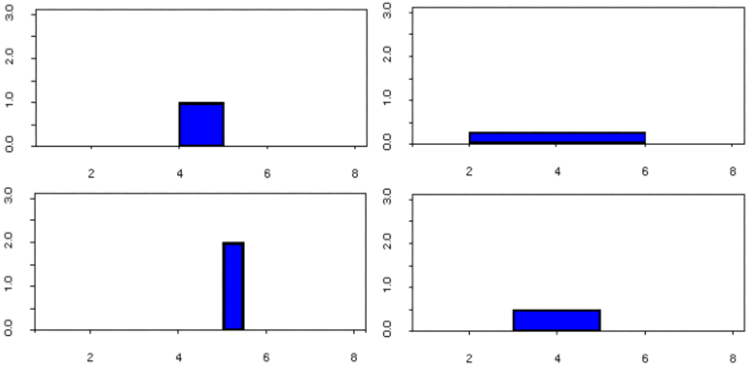
X is said to be *uniformly distributed* on the interval $[\alpha, \beta]$ if its density is

$$f(x) = \begin{cases} \frac{1}{\beta-\alpha} & \text{if } x \in [\alpha, \beta] \\ 0 & \text{otherwise} \end{cases}$$

As $F(a) = \int_{-\infty}^{a} f(x)dx$, we get

$$F(a) = \begin{cases} 0 & a < \alpha \\ \frac{a-\alpha}{\beta-\alpha} & a \in [\alpha, \beta] \\ 1 & a > \beta \end{cases}$$

PTCF: Continuous Unif. Dist., Some Density Plots



PTCF: Exponential Distribution

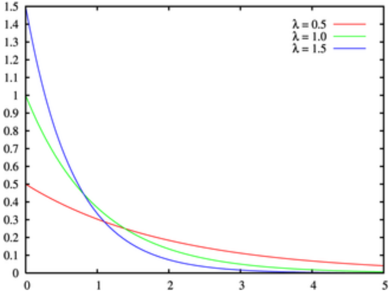
X is said to be exponentially distributed with parameter λ if its density is

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

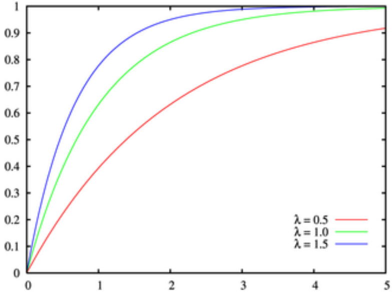
Its cdf F is

$$F(a) = \int_{-\infty}^{a} f(x)dx = 1 - e^{-\lambda a}, \ a \geq 0.$$

PTCF: Exponential Dist., Some Plots



Densities



Distributions

PTCF: Normal Distribution

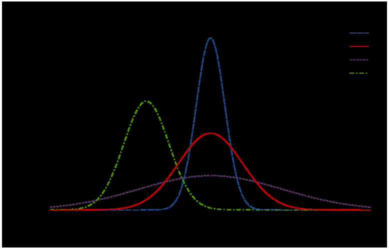
- A continuous r.v. X is *normally distributed* with parameters μ and σ^2 if the density of X is given by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \ x \in \mathbb{R}.$$

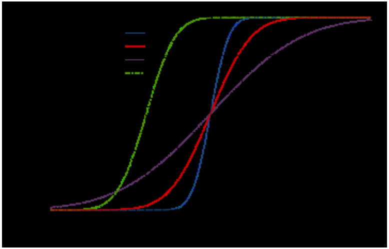
  Normal variables are also called *Gaussian* variables.

- If X is normally distributed with parameters μ and σ^2, then $Y = \alpha X + \beta$ is normally distributed with parameters $\alpha\mu + \beta$ and $(\alpha\sigma)^2$
- When $\mu = 0$ and $\sigma^2 = 1$, X is said to have *standard normal* distribution.

PTCF: Normal Distribution, Some Plots



Densities



Distributions

PTCF: Continuous Distribution Random Number Generation

- Uniform distribution: discretize it, then use some *pseudo-random number generator*
- Let's assume we can generate a uniform number $X \in [0, 1)$.

Question:

- How to generate $Y \in \text{Normal}(\mu, \sigma)$?
- It is actually sufficient to generate $Y \in \text{Normal}(0, 1)$
- How to generate a point on an n-sphere uniformly at random?

PTCF: Normal Distribution Random Number Generator

The Polar Method (for Normal$(0, 1)$)

1. Generate $V_1, V_2 \in [-1, 1]$ uniformly
2. $S = V_1^2 + V_2^2$
3. If $S \geq 1$, go back to step 1
4. Set $X_1 = V_1\sqrt{\frac{-2\ln S}{S}}$ and $X_2 = V_2\sqrt{\frac{-2\ln S}{S}}$
   Then, X_1 and X_2 are independent standard normal variables

For Normal(μ, σ))

1. Let X be a standard normal variable
2. Then, $Y = \mu + \sigma X$ is Normal(μ, σ)

PTCF: Generating a Random Point on an n-Sphere

- Generate $X_1, \dots, X_n$ independently from Normal$(0, 1)$
- Let $r = (r_1, \dots, r_n)$ be defined by

$$r_i = \frac{X_i}{\sqrt{X_1^2 + \cdots + X_n^2}}$$

- The joint density of the X_i only depends on $\sqrt{X_1^2 + \cdots + X_n^2}$, so the distribution is spherically symmetric, and thus its projection on to the sphere (i.e. $\mathbf{r}$) is uniformly distributed on the surface of the sphere!