

Estimating the volume of a convex body using the sampling technique

Problem:

- The problem is to estimate the volume of a convex body K which is a subset of \mathbb{R}^n .
- Since we are going to use a sampling strategy assume for now that there exists a membership *oracle* that answers questions “ $X \in K?$ ” for any point X in \mathbb{R}^n .
- For the problem to be solvable we need to know at least one point P that belongs to K
- Define $B(P, r)$ as a ball in \mathbb{R}^n that is centered at point P and has a radius of r . and let R be defined such that:

$$B(P, 1) \text{ is in } K \text{ which is in } B(P, R)$$

Note that:

For a small number of dimensions we can easily estimate the volume of K by sampling uniformly from $B(P, R)$ and observing the frequency of the sample points locating in K . this works well because the unit ball, whose volume is 1, is in K and thus:

$$\frac{\text{Vol}(K)}{\text{Vol}(B(P, R))} \geq \frac{1}{R^n}$$

So $\Theta(R^n)$ samples are sufficient for a good estimate. However, for fixed R , the sample size becomes exponential in n so the method fails when n is variable.

It has been shown that a deterministic algorithm that is restricted to a sub-exponential (in n) number of oracle calls can not approximate volume within a factor of 2, even if R is a constant. However, randomization allows us to do much better.

The sampling problem:

We define the following random walk on K :

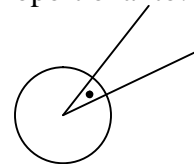
Let δ be a positive constant. Given a point $X_t \in K$, we draw the next point X_{t+1} uniformly at random from $B(X_t, \delta) \cap K$.

In order to sample from $B(X_t, \delta) \cap K$, use *rejection sampling*:

Keep drawing points uniformly at random from $B(X_t, \delta)$ and stop when you get a point in K . Unfortunately, the expected number of trials will be inversely proportional to:

$$\text{Vol}[B(X_t, \delta) \cap K] / \text{Vol}[B(X_t, \delta)]$$

Even for small values of δ , this ratio can be tiny, if we are close to a corner of K , as the figure illustrates:



Thus, we must distinguish between the *speedy walk*, in which we count each accepted sample as one step, and the *full walk*, in which we count both accepted and rejected steps so that every oracle call is a step.

The speedy walk has the advantage of having a stationary distribution that is uniform over K , but the step count for the full walk is the right measure of complexity since it counts all oracle calls.

The distinction between the speedy walk and the full walk becomes unimportant if we assume that K is *smooth* meaning that for every point x in K there is a unit ball B such

that x belongs to B and B is a subset of K . Unfortunately this condition rules out polyhedra, an important case.

Theorem: there exists a constant C such that for a smooth K and $\delta=C/\sqrt{n}$

$$X \in K \Rightarrow \text{Vol}[B(X, \delta) \cap K] / \text{Vol}[B(X, \delta)] \geq .4$$

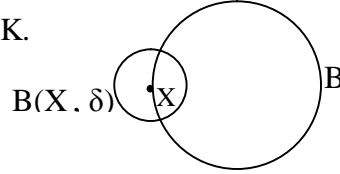
Proof sketch:

Let B be a ball of radius 1 containing X and contained in K .

It can be shown that

$$\text{Vol}[B(X, \delta) \cap B] / \text{Vol}[B(X, \delta)] \geq .4$$

The worst case is when X is on the boundary of B :



Thus the full walk is only 2.5 times slower than the speedy walk and we can focus on the speedy walk whose stationary distribution is uniform.

The mixing time ψ_ϵ is defined as the least t such that for all $t' \geq t$ and all measurable sets A in K , the probability that the walk starting at P lies at a point in A after t' steps is between $(1-\epsilon) \text{Vol}[A] / \text{Vol}[K]$ and $(1+\epsilon) \text{Vol}[A] / \text{Vol}[K]$

Formally:

$$\psi_\epsilon = \min\{t \mid t' \geq t \text{ \& measurable } A \text{ in } K \Rightarrow (1-\epsilon) \text{Vol}[A] / \text{Vol}[K] \leq \Pr(X_{t'} \in A) \leq (1+\epsilon) \text{Vol}[A] / \text{Vol}[K]\}$$

Theorem: if K is smooth then

$$\psi_\epsilon = O\left(\frac{R^2 n}{\delta^2} \left(\log(1/\epsilon) + n \frac{\log(R)}{\delta}\right)\right)$$

Since δ is constant \sqrt{n} we have $\psi_\epsilon = O(R^2 n^4)$

The proof is technical and long so it is omitted in this note. See Jerum Ch. 6 for the proof.

The theorem can be extended to the case where K is not smooth, at the cost of increasing the bound on mixing time by a factor of n . in this case the full walk can stall for a long time if it gets extremely close to a corner but it is unlikely to reach such a troublesome point within $O(R^2 n^5)$ steps.

Using samples to estimate volume

The idea is simple. We define a series of concentric balls where:

$$B_0 \subset B_1 \subset \dots \subset B_t \quad B_0 = B(p, 1); B_t = B(p, R) \text{ and } \text{Vol}[B_{i+1}] \leq 2\text{Vol}[B_i].$$

This can be done with $t = O(n \log R)$

We have $\text{Vol}[B_0 \cap K] = \text{Vol}[B_0]$ and can estimate $\text{Vol}[B_i \cap K] / \text{Vol}[B_{i+1} \cap K]$ by sampling almost uniformly from $[B_{i+1} \cap K]$ and observing the frequency with which the samples lie in $[B_i \cap K]$.

Let S_i be the estimate of $\text{Vol}[B_i \cap K] / \text{Vol}[B_{i+1} \cap K]$. Then $\text{Vol}[B_0] \prod_{i=1}^t (1/S_i)$ is an estimate of $\text{Vol}[K]$.

Online Computational and Competitive Analysis

Example: Ski rental problem

Suppose it costs 1 unit of money to rent skis for a ski trip and C units to buy a pair of skis. Should we buy skis or should we rent each time?

Case I: off-line algorithm) t , the number of ski trips is known. In this case, we should buy if $t \geq c$ and rent if $t < c$. Our optimal cost for this case would be $\text{opt} = \min(t, c)$.

Case II: on-line algorithm) t is unknown. Consider the policy of renting for the first s trips and buying before $(s+1)^{\text{th}}$ trip. Then our cost would equal t if $t \leq s$ and $t+c$ if $t > s$.

One way to evaluate this policy is to calculate the largest possible ratio of its cost to the cost that we would have if we knew t in advance.

Then we need to minimize $\text{Max}(\text{on-line cost}/\text{off-line cost})$ over all values of t , that is called the **competitive ratio** of the on-line algorithm. The following table shows the costs after each trip for $s=4$ and $c=6$:

t	On-line cost	Off-line cost
1	1	1
2	2	2
3	3	3
4	4	4
5	10	5
6	10	6

∴ Ratio is 10/5

Now assume that $s=8$ and $c=6$:

T	On-line cost	Off-line cost
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	6
8	8	6
9	14	6
10	14	6

∴ Ratio is 14/6

Observe that the worst-case ratio occurs when $t = s + 1$ and equals $(s+c)/\min(s+1, c)$

If c is an integer, this ratio is minimized when $s = c-1$ giving the ratio $(2c-1)/c$

Thus the best competitive ratio achievable for the ski rental problem is $(2c-1)/c$ and it is obtained by the rule: Buy skis on the c^{th} trip.

These results are very conservative for the on-line algorithm. However, for randomized values of s the on-line algorithm would work better and the result would be more realistic.

Another example: List accessing

Assume that a linked list data structure supports the operations $\text{INSERT}(x)$, $\text{DELETE}(x)$, and $\text{ACCESS}(x)$, where x is a key. The cost of each operation is 1 plus the number of keys preceding x or 1 plus the length of the list if x was not on the list before the operation.

Once a key has been inserted or accessed in a step, it can be moved ahead to any earlier place in the list at no cost, since the algorithm can remember the links it traversed in accessing or inserting a key.

We will show in the next section that MTF policy –which moves the recently accessed or inserted key to front– achieves a competitive ratio of 2.