Smoothed Analysis
With Applications in Machine Learning

Contents

1 Acknowledgements

I would like to thank Felix Fischer for setting this essay, for his time and his advice. I would also like to thank Bhargav Narayanan for organising my Part III seminar on this topic, and everyone who attended it and asked thoughtful questions. I thank Daniel Spielman for kindly granting permission to use his diagrams (Figure 1).

2 Purpose of the Essay

The purpose of this essay is to give an accessible introduction to smoothed analysis, including a complete presentation of two smoothed analysis results relevant to machine learning:

- D. Arthur and S. Vassilvitskii. Worst-case and smoothed analysis of the ICP algorithm, with an application to the k-means method. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 153–164. IEEE Computer Society Press, 2006.

- A. Blum and J. Dunagan. Smoothed analysis of the perceptron algorithm for linear programming. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 905–914. Society for Industrial and Applied Mathematics, 2002.

This essay seeks to simplify the presentation of these results by refocussing on machine learning and including more explanation, commentary and diagrams. All diagrams are by me unless otherwise stated. I have also fixed what appear to be minor errors in the original work (see Appendix A). The essay is therefore of interest to anyone who wishes to acquaint themselves with smoothed analysis, anyone with an interest in theoretical machine learning, or anyone who would like to understand the above two results more easily than by reading the original papers. All sources used are listed in the bibliography.

Overview: In Section 3 we motivate and define smoothed analysis, discuss the definition, and give some history. We also discuss why machine learning is a particularly suitable area for smoothed analysis. In Sections 4 and 5 we present smoothed analyses of the k-means method and perceptron algorithm respectively. In Section 6 we discuss some of the subtleties and open problems of smoothed analysis.

3 Introduction to Smoothed Analysis

3.1 The Problem: A Gap Between Theory and Practice

Sometimes there are gaps between theory and practice. For example, the *simplex algorithm* solves linear programs (LPs) of the form

$$\begin{aligned} &\max c^T x \\ &\text{subject to } Ax \leq y \end{aligned} \tag{1}$$

where c, x and y are real vectors and A a real matrix. The algorithm, invented in 1947 by Dantzig [11], works by walking along the edges of the feasible region, which is a *convex polytope*, until the optimal vertex is found. In 1972 Klee and Minty [19] proved that the simplex algorithm has exponential worst-case complexity. Despite this, the simplex algorithm is very efficient in practice and, despite decades of effort to oust it, remains the most widely-used linear program solving algorithm today [30].

This situation is not uncommon; there are many other algorithms whose practical usefulness contradicts their poor worst-case complexity (see Section 3.3 for examples). Ideally, a measure of complexity should *reflect* an algorithm's practical usefulness, to aid our design and selection of better algorithms for real world applications. We use worst-case complexity as our traditional measure of complexity for good reason; we would like guarantees on the running time. However it seems that worst-case complexity is overly pessimistic, declaring many algorithms intractable when in reality the worst case input occurs rarely, or never, in practice. In fact, many proofs of bad worst-case complexity tend to use contrived constructions of 'bad' input, whose naturality in any real-life setting is dubious.

One alternative is *average-case complexity*. While worst-case complexity asks 'what is the *longest* possible running time?', average-case asks 'what is the *expected* running time?' While this definition avoids the problem of overemphasis on unnatural worst-case examples, it implicitly assumes some probability distribution of the input. This can be problematic, as it is often unclear or unknown what distribution is representative for a particular application. In addition, average-case complexity results are not transferable: a result with respect to one distribution does not apply for another distribution. Therefore a single algorithm may need to be analysed many times, once for each distribution deemed relevant. This can be time-consuming.

3.2 A Solution: Smoothed Complexity

Imagine you are playing a game with an adversary. You have an algorithm, which you want to run quickly; your adversary wants it to run slowly. Your adversary may choose any input and you must run your algorithm on it. Worst-case complexity asks 'what is the longest running time, given that your adversary can choose any input?' Smoothed complexity weakens the adversary — they now have a *trembling hand* (to quote [18]). That is, your adversary can still choose whatever input they like, but a small random perturbation (outside the adversary's control) is applied to their choice. Smoothed complexity then asks 'what is the longest *expected* running time, given that your adversary can choose any input which will then be perturbed?'

Formally, suppose A is an algorithm with input space X. Let X_n denote inputs of size n. There are a number of ways of measuring 'size of input', but for our purposes we take this to mean the number of real variables in the input, so $X_n = \mathbb{R}^n$. For $\mathbf{x} \in X$, let $C_A(\mathbf{x})$ denote the running time of A on $\mathbf{x}$. The *worst-case complexity* of A is

$$\max_{\mathbf{x} \in X_n} C_A(\mathbf{x}). \tag{2}$$

The *average-case complexity* of A (given some distribution μ_n) is

$$\mathop{\mathbb{E}}_{\mathbf{x} \xleftarrow{\mu_n} X_n} C_A(\mathbf{x}). \tag{3}$$

Note that worst-case and average-case complexities are functions of n, the input size. The *smoothed complexity* [30] of A is

$$\max_{\mathbf{x} \in X_n} \mathop{\mathbb{E}}_{\mathbf{g}}(C_A(\mathbf{x} + \sigma \|\mathbf{x}\| \mathbf{g})) \tag{4}$$

where $\mathbf{g}$ is a vector of n Gaussian random variables with mean 0 and variance 1, σ is a positive scalar, and $\|x\| = \max\{|x_1|, \ldots, |x_n|\}$ is the uniform norm for $X_n = \mathbb{R}^n$. Note that smoothed complexity is a function of n and σ, taking into account the size of the input and the size of the perturbation. The $\|\mathbf{x}\|$ in Equation (4) is necessary to prevent the adversary from choosing arbitrarily large input, rendering the effect of the perturbation negligible. If A satisfies $C_A(s\mathbf{x}) = C_A(\mathbf{x})$ for all scalars $s > 0$ (that is, the algorithm's running time is unaffected by scaling the input) then Equation (4) is equal to

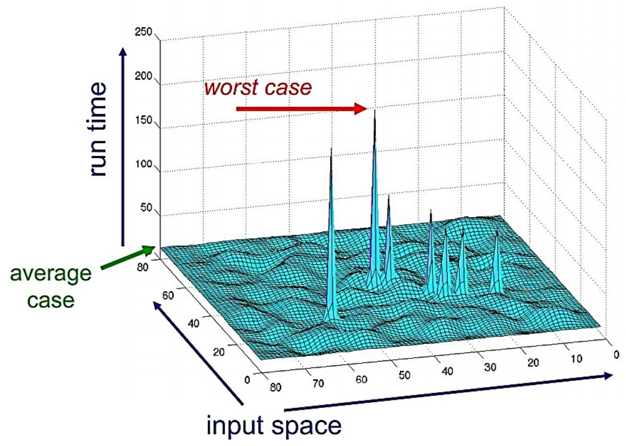
$$\max_{\mathbf{x} \in [-1,1]^n} \mathop{\mathbb{E}}_{\mathbf{g}}(C_A(\mathbf{x} + \sigma \mathbf{g})). \tag{5}$$

That is, by restricting the magnitude of the real variables of the input, we do not need to make the perturbation proportional to the input [33]. Another equivalent formulation is to consider

$$\max_{\mathbf{x} \in X^n} \mathop{\mathbb{E}}_{\mathbf{g}}(C_A(\mathbf{x} + \sigma \mathbf{g})) \tag{6}$$

as a function of n and $\frac{\|x\|}{\sigma}$. As the running times of both the k-means method and perceptron algorithm are unaffected by scaling, we use the more convenient definitions given by Equations (5) and (6).

We say A has *polynomial smoothed complexity* [30] if Equation (4) or (5) is bounded above by a polynomial in n and $1/\sigma$, or equivalently if (6) is bounded above by a polynomial in n and $\frac{\|x\|}{\sigma}$. Following the Equation (5) definition, we say A has *probably polynomial smoothed complexity* [7, 33] if there exists



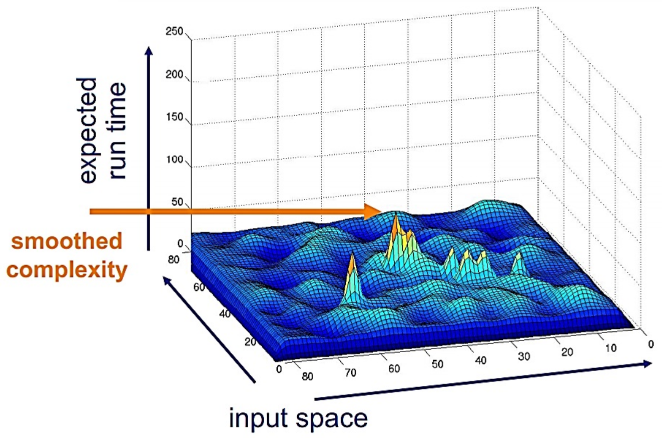


Figure 1: Top: worst-case and average-case complexity. Bottom: smoothed complexity. Pictures by Daniel Spielman, used with permission.

a polynomial $p(n, 1/\sigma, 1/\delta)$ such that for all $n \in \mathbb{N}$, $\sigma > 0$, $\delta \in [0, 1]$, and $\mathbf{x} \in [-1, 1]^n$, the probability that $C_A(\mathbf{x}+\sigma\mathbf{g})$ exceeds $p(n, 1/\sigma, 1/\delta)$ is at most δ. Probably polynomial smoothed complexity is weaker than polynomial smoothed complexity.

The typical effect of smoothed complexity can be seen in Figure 1. On top is a plot of the running time of an algorithm for different inputs. The running time is mostly small, except for some isolated steep spikes, where the running time is large. The worst-case complexity is the height of the tallest spike. The average-case complexity is low (assuming, say, uniform distribution over the input space). Neither of these are satisfactory: the worst-case complexity is too pessimistic, and the average-case does not pay enough attention to the spikes. In the bottom plot, the vertical axis measures expected running time, with the expectation taken over a small Gaussian perturbation. The perturbation 'smooths down' the steep spikes. The smoothed complexity of the algorithm is then the maximum height of the plot on the bottom (much smaller than the worst-case complexity). Note that it is possible that the input achieving the worst-case complexity is not the same as the input achieving the smoothed complexity.

Why is Equation (4) a good definition? Firstly, smoothed complexity is a hybrid of worst- and average-case complexity, achieving the best of both. As σ tends to 0, Equation (4) becomes worst-case complexity, and as σ tends to ∞, Equation (4) becomes average-case complexity (with respect to a Gaussian distribution centred at 0 with large variance). Smoothed complexity avoids the pessimism of worst-case complexity by lessening the influence of isolated worst-case examples, but avoids the arbitrary nature of average-case complexity by considering the maximum (i.e. worst case) over all unperturbed inputs. Secondly, smoothed complexity can help us to distinguish between 'bad' and 'really bad' worst-case complexity. That is, if an algorithm has isolated worst-case spikes as in Figure 1, the smoothed complexity will be much less than the worst-case complexity. However, if there is a whole dense area of worst-case behaviour (unlike Figure 1), the smoothed complexity will be similar to the worst-case complexity. Lastly, smoothed complexity is especially suitable in settings where the input to an algorithm is subject to noise, because in such cases we would be unconcerned by isolated worst-case inputs.

3.3 History of Smoothed Analysis

Smoothed analysis was invented in 2001 by Spielman and Teng in order to explain the good practical performance of the simplex algorithm (described in Section 3.1). They showed that the algorithm has polynomial smoothed complexity [30] and won the Godel Prize for their work. Since then, smoothed analysis has been used to explain the good performance of many other algorithms such as the Nemhauser-Ullmann heuristic for the knapsack problem [5, 6, 25], quicksort [4, 13, 21], the 2-opt heuristic for TSP [12], Goldberg's algorithm for single shortest path [4], binary programming [7], and integer programming [26]. See [33] for a recent summary article.

3.4 Machine Learning and Smoothed Analysis

Machine learning is a particularly suitable area for smoothed analysis for two reasons. Firstly, it is an area where theory lags behind practice. Experimentalists have made great progress in recent years, but the good performance of many machine learning algorithms are yet to be explained. Secondly, in many machine learning applications such as machine vision, natural language processing and medical diagnosis, the input, which involves real-world measurements, contains inherent noise.

The two algorithms analysed in the remainder of this essay, the k-means method for clustering and the perceptron algorithm for perceptron learning, are central to machine learning. Other applications of smoothed analysis in machine learning include Kalai and Teng's proof that all decision trees are PAC-learnable from most product distributions [18], and the work of Kalai et al. on PAC-learning DNFs and agnostically learning decision trees [17].

4 The k-Means Method

Clustering is a vital machine learning task, used in diverse applications such as social network analysis, recommender systems, image segmentation and medical imaging. In particular, we have the following problem:

The k-Means Clustering Problem: Given a set of n points $\mathcal{X} = \{x_1, \ldots, x_n\}$ in $\mathbb{R}^d$, find k cluster centres $\{c_1, \ldots, c_k\}$ minimising the *potential function*

$$\phi = \sum_{x \in \mathcal{X}} \min_{1 \le i \le k} \|x - c_i\|^2. \tag{7}$$

That is, we want to partition n points into k clusters, each with a centre, so that each point is closest to its cluster centre, and the sum of the squared distance between each point and its cluster centre is minimised.

4.1 Description of the k-Means Method

The k-means method, sometimes called *Lloyd's algorithm* (developed by Lloyd in [20]), proceeds as follows.

The k-Means Method

1. Choose an arbitrary set of k cluster centres.
2. Set the cluster $\mathcal{C}_i$ to be the set of points of $\mathcal{X}$ closer to c_i than any other c_j.
3. Recompute the optimal centres for these clusters by setting $c_i = \frac{1}{|\mathcal{C}_i|} \sum_{x \in C_i} x$, the centre of mass of $\mathcal{C}_i$.
4. Repeat steps 2 and 3 until the partitioning stabilises.

See Figure 2 for a demonstration of the k-means method in action. The dotted lines are perpendicular bisectors between the cluster centres that help us perform Step 2 (assigning points to closest centres). These hyperplanes are called *Voronoi boundaries* [3], dividing the space into *Voronoi cells*. Note that we will always denote a cluster by $\mathcal{C}$ and a fixed subset of $\mathcal{X}$ by S. That is, $\mathcal{C}_i$ may change value many times over the course of the algorithm. For example, $\mathcal{C}_3$ is 'all points of $\mathcal{X}$ shaped like a square' in Figure 2.

The k-means method takes advantage of the following facts:

- If the partition is fixed, then it is easy to choose optimal cluster centres (just take centres of mass; see Lemma 4.6 for proof).

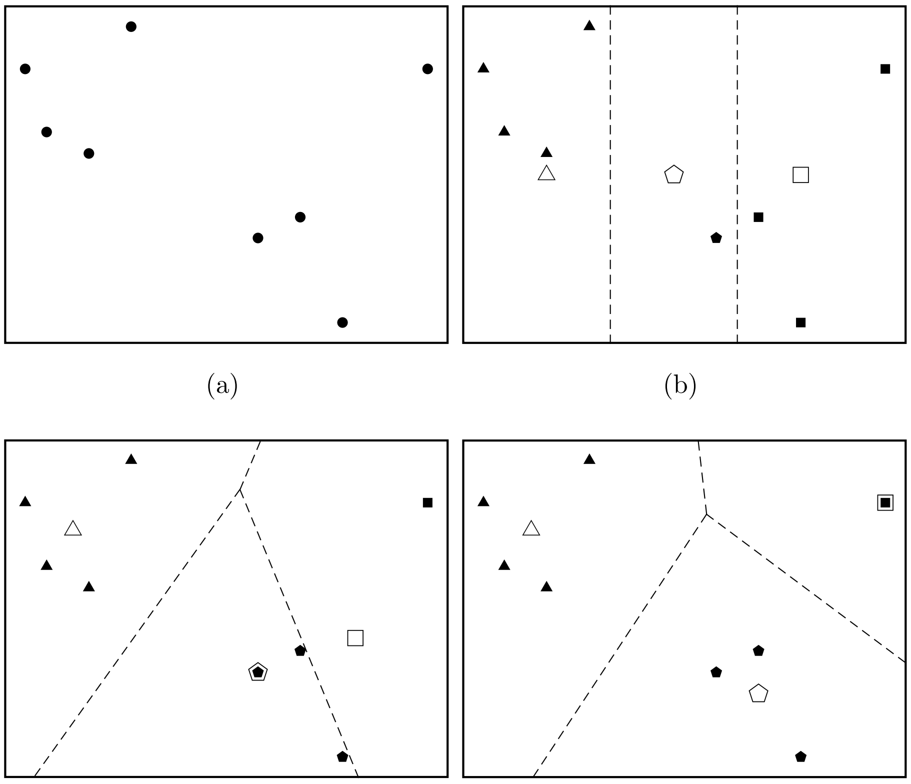


Figure 2: A demonstration of the k-means method with $d = 2$ and $k = 3$. (a): The point set $\mathcal{X}$. (b): Choose arbitrary centres and assign points to nearest centres. (c): Recalculate the centres and assign points to nearest centres. (d): Recalculate the centres. Algorithm terminates because points are already assigned to nearest centres.

- If the cluster centres are fixed, then it is easy to choose an optimal partition (just assign each point to its nearest centre).

The algorithm simply iterates between these two actions until there are no changes to be made. This is guaranteed: each step strictly decreases the potential ϕ, therefore no partition can be repeated. As there are only finitely many possible k-partitions, the k-means method always terminates.

4.2 Analysis of the k-Means Method

The algorithm trivially terminates in k^n iterations, as this is the number of ways to k-partition n objects. This bound was improved by [16], who showed that the k-means method never takes more than $O(n^{kd})$ iterations. The proof works by counting the number of ways to partition n points into k clusters using Voronoi boundaries in d-dimensional space. However this bound is not $\textsc{Poly}(n)$ when k or d is unfixed. As for lower bounds: the k-means method has been shown to have superpolynomial worst-case complexity of $2^{\Omega(\sqrt{n})}$ [2]. In addition, the k-means method is not optimal — the solution it finds is only locally optimal.

Despite these major theoretical drawbacks, the k-means method is widely used for its simplicity and its good practical performance [8]. This motivated a smoothed analysis of the algorithm, which emerged over a series of papers [1, 3, 22]. In 2006, Arthur and Vassilvitskii proved that

Theorem 4.1 (Main Theorem (Arthur, Vassilvitskii [3])). *Fix an arbitrary set $\mathcal{X}' \subset \mathbb{R}^d$ of n points and assume that each point in $\mathcal{X}'$ is independently perturbed by a Gaussian distribution with mean 0 and variance σ^2, yielding a new set $\mathcal{X}$ of points, and let D denote the diameter of $\mathcal{X}$. Then the expected running time of the k-means method on $\mathcal{X}$ is bounded above by a polynomial in n^k and D/σ.*

If we had n instead of n^k in this statement, we would have polynomial smoothed complexity in the sense of Equation (6). However, n^k is still an improvement on the worst-case bound of $O(n^{kd})$, as it is independent of the dimension d. Theorem 4.1 was later improved slightly by Manthey and Roglin [22]. Finally, in 2009 Arthur, Manthey and Roglin showed that

Theorem 4.2 (Arthur, Manthey, Roglin [1]). *Fix an arbitrary set $\mathcal{X}' \subseteq [0,1]^d$ of n points and assume that each point in $\mathcal{X}'$ is independently perturbed by a Gaussian distribution with mean 0 and variance σ^2, yielding a new set $\mathcal{X}$ of points. Then the expected running time of k-means on $\mathcal{X}$ is bounded above by a polynomial in n and $1/\sigma$.*

Thus the k-means method has polynomial smoothed complexity in the sense of Equation (5). Both Theorems 4.1 and 4.2 have intricate proofs, but the earlier result is significantly simpler. We will therefore prove Theorem 4.1 in full (excepting one technical lemma) and then give a very high level sketch of Theorem 4.2.

4.3 Overview of the Proof

All proofs in the rest of this section are due to [3] unless otherwise stated, though all diagrams are by me. The most significant difference between my presentation and that of [3] is the proof of Theorem 4.14, which appeared to have an error in the original paper. I have resolved the apparent error, and my corrections make no difference to the main theorem. See Appendix A for details.

The proof proceeds as follows. After giving some preliminary lemmas and definitions (Section 4.4), we consider two cases: either some cluster makes a 'large change' (meaning it gains $\geq 2kd$ points in a single iteration or loses $\geq 2kd$ points in a single iteration), or no cluster makes a 'large change'. In the 'large change' scenario (Section 4.6), we find that if $\mathcal{X}$ is 'separated' then there is a large potential change during the iteration with the 'large change' of points, and that furthermore $\mathcal{X}$ is likely to be 'separated'. In the 'small change' scenario (Section 4.5), we find that if $\mathcal{X}$ is 'sparse' then there is a large potential change within a certain timeframe, and that furthermore $\mathcal{X}$ is likely to be 'sparse'. We put these results together to obtain the main bound (Section 4.7). In Section 4.8 we briefly sketch the more advanced result showing that k-means has polynomial smoothed complexity.

4.4 Preliminaries

The following preliminary lemma gives a simple upper bound on the probability that a variable with Gaussian distribution is in any fixed ball.

Lemma 4.3 ('Gaussian variable unlikely to be in a fixed ball')**.** *Suppose y has d-dimensional Gaussian distribution with mean $\mu \in \mathbb{R}^d$ and variance σ^2. Then, given any fixed ball of radius ϵ, y is in the ball with probability at most $(\epsilon/\sigma)^d$.*

Proof. The probability distribution function for y has maximum value $1/(\sqrt{2\pi}\sigma)^d$. The volume of the ball is at most $(2\epsilon)^d$ (by considering the hypercube containing the ball), so the probability that y is in the ball is at most $(2\epsilon)^d/(\sqrt{2\pi}\sigma)^d < (\epsilon/\sigma)^d$. □

Definition 4.4. *Given a point x and hyperplane $\mathcal{H}$, $d(x, \mathcal{H})$ is the minimum distance from x to $\mathcal{H}$.*

Definition 4.5. *Given a set of points S, the* centre of mass *of S is*

$$c(S) = \frac{1}{|S|} \sum_{s \in S} s \tag{8}$$

The next lemma proves that the centre of mass is the optimal choice for a cluster centre. It will also help us measure the change in potential when a cluster centre is changed.

Lemma 4.6 ('Potential change wrt centre of mass')**.** *For any set of points S and arbitrary point x,*

$$\sum_{s \in S} \|s - x\|^2 = \sum_{s \in S} \|s - c(S)\|^2 + |S| \cdot \|c(S) - x\|^2. \tag{9}$$

Proof.

$$\begin{aligned}
\sum_{s \in S} \|s - x\|^2 &= \sum_{s \in S} (s - x) \cdot (s - x) \\
&= \sum_{s \in S} (s - c(S) + c(S) - x) \cdot (s - c(S) + c(S) - x) \\
&= \sum_{s \in S} (s - c(S)) \cdot (s - c(S)) + \sum_{s \in S} (c(S) - x) \cdot (c(S) - x) \\
&\quad + 2 \sum_{s \in S} (c(S) - x) \cdot (s - c(S)) \\
&= \sum_{s \in S} \|s - c(S)\|^2 + \sum_{s \in S} \|c(S) - x\|^2 + 2(c(S) - x) \cdot \sum_{s \in S} (s - c(S))
\end{aligned}$$

Lastly note that $\sum_{s \in S}(s - c(S)) = 0$. $\square$

4.5 Small Cluster Changes

We begin the 'small changes' case with the following definition.

Definition 4.7. *Given a set of points $\mathcal{X} \subset \mathbb{R}^d$ with $|\mathcal{X}| = n$, a* key-value *is any expression of the form*

$$\frac{n_1}{n_2} c(S), \tag{10}$$

where $S \subset \mathcal{X}$ has at most $4kd$ points, and where n_1 and n_2 are coprime positive integers satisfying $n_1 \leq n^2$ and $n_2 < n$.

A key-value is a special kind of linear combination of points in $\mathcal{X}$. Given any two linear combinations a and b of points of $\mathcal{X}$, we write $a \equiv b$ to mean that the coefficients for each point of $\mathcal{X}$ are the same.

Definition 4.8. $\mathcal{X} \subset \mathbb{R}^d$ *is* δ-sparse *if any key-values (a, b, c, d) that satisfy $\|a + b - c - d\| \leq \delta$ also satisfy $a + b \equiv c + d$.*

Intuitively, δ-sparseness is a notion of sparseness because any linear combinations of points that are close enough must be the same.

Definition 4.9. *Given a constant C, a* C-epoch *is a sequence of iterations of the k-means method during which the potential decreases by a total of at most C.*

Our overall goal in this section is to show that under certain conditions, C-epochs are short. The next theorem shows that in the 'small changes' case, if $\mathcal{X}$ is $2n^2\sqrt{C}$-sparse, then any cluster can have at most two different values during one C-epoch. We will then deduce that a C-epoch lasts at most 2^k iterations.

Theorem 4.10. *Suppose the k-means method is run on a $2n^2\sqrt{C}$-sparse set $\mathcal{X} \subset \mathbb{R}^d$ of size n. Take some cluster $\mathcal{C}$, and suppose $\mathcal{C}$ never gains at least $2kd$ points in a single iteration, and never loses at least $2kd$ points in a single iteration. Then $\mathcal{C}$ can have at most two different values during one C-epoch.*

Proof. Suppose not: $\mathcal{C}$ takes on at least three different values in a single epoch. Then there exist $S_1, S_2, S_3 \subset \mathcal{X}$, all distinct, that occur as consecutive values of $\mathcal{C}$. Let $A = S_1 \cap S_2 \cap S_3$ and let $B_i = S_i - A$ for $i = 1, 2, 3$. Then

$$\begin{aligned} |B_1| &= |S_1 - S_1 \cap S_2 \cap S_3| \\ &= |S_1 - S_1 \cap S_2| + |S_1 \cap S_2 - S_1 \cap S_2 \cap S_3| \\ &\leq |S_1 - S_1 \cap S_2| + |S_2 - S_2 \cap S_3| \end{aligned} \tag{11}$$

which is $< 4kd$ as $\mathcal{C}$ never loses $\geq 2kd$ points in a single iteration. We similarly show that $|B_2|, |B_3| < 4kd$.

Now consider $\|c(S_1) - c(S_2)\|$. When we recalculate the centre of $\mathcal{C}$ and change it from $c(S_1)$ to $c(S_2)$, by Lemma 4.6 ('potential change wrt centre of mass') the potential drop is at least $|S_2|.\|c(S_2) - c(S_1)\|^2 \geq \|c(S_2) - c(S_1)\|^2$.

As the transition from S_1 to S_2 happens within a single C-epoch, then $\|c(S_2) - c(S_1)\| \leq \sqrt{C}$. We also have

$$c(S_2) - c(S_1) = \frac{|A|c(A) + |B_2|c(B_2)}{|A| + |B_2|} + \frac{|A|c(A) + |B_1|c(B_1)}{|A| + |B_1|}. \tag{12}$$

Equivalently,

$$\begin{aligned} &|A|(|B_1| - |B_2|)c(A) = \\ &|S_1|.|S_2|(c(S_2) - c(S_1)) + |B_1|.|S_2|c(B_1) - |B_2|.|S_1|c(B_2). \end{aligned} \tag{13}$$

Furthermore note that $|S_1|.|S_2|\|c(S_2) - c(S_1)\| \leq n^2\sqrt{C}$.

We wish to divide by $|B_1| - |B_2|$ in Equation (13). Note that $|B_1| \neq |B_2|$, because if $|B_1| = |B_2|$, Equation (13) tells us that $a = |B_1|.|S_2|c(B_1)$ and $b = |B_2|.|S_1|c(B_2)$ are within distance $n^2\sqrt{C}$, then as $\mathcal{X}$ is $2n^2\sqrt{C}$-sparse, $a \equiv b$, and hence $B_1 = B_2$, a contradiction as $S_1 \neq S_2$.

Dividing by $|B_1| - |B_2|$ in Equation (13) we find that

$$x = \frac{|B_1|.|S_2|}{|B_1| - |B_2|}c(B_1) - \frac{|B_2|.|S_1|}{|B_1| - |B_2|}c(B_2) \tag{14}$$

is within $n^2\sqrt{C}$ of $|A|c(A)$. Applying all the above reasoning to the transition from S_2 to S_3, we find that

$$y = \frac{|B_2|.|S_3|}{|B_2| - |B_3|}c(B_2) - \frac{|B_3|.|S_2|}{|B_2| - |B_3|}c(B_3) \tag{15}$$

is also within $n^2\sqrt{C}$ of $|A|c(A)$. So $\|x - y\| \leq 2n^2\sqrt{C}$. Since x and y are both differences of two key-values (as $|B_i| < 4kd$), and $\mathcal{X}$ is $2n^2\sqrt{C}$-sparse, we have $x \equiv y$.

Now, we show $B_1 \cap B_2 = \emptyset$. Suppose there exists $p \in B_1 \cap B_2$. Then the coefficient of p in x is -1 (by rearranging), and if $p \notin B_3$, its coefficient in y is $\frac{|A|+|B_3|}{|B_2|-|B_3|} \neq -1$. So $p \in B_1 \cap B_2 \cap B_3$, contradicting the definition of the B_i. Similarly $B_2 \cap B_3 = \emptyset$.

Therefore the set of points of $\mathcal{X}$ for which x has nonzero coefficient is precisely $B_1 \cup B_2$, and the points for which y has nonzero coefficient is $B_2 \cup B_3$. Then $B_1 \cup B_2 = B_2 \cup B_3$ so $B_1 = B_3$, contradicting $S_1 \neq S_3$. □

We can now deduce that C-epochs are short, under the 'small changes' condition.

Corollary 4.11. *Suppose the k-means method is run on a $2n^2\sqrt{C}$-sparse set $\mathcal{X} \subset \mathbb{R}^d$ of size n. Then, during any 2^k consecutive iterations, either some cluster has gained at least $2kd$ points in a single iteration, or has lost at least $2kd$ points in a single iteration, or the potential has decreased by a total of at least C.*

Proof. If no cluster gains/loses $2kd$ points in a single iteration, then Theorem 4.10 says that all clusters have at most two values in a single C-epoch. As the k-means method can never repeat configurations, this means that there are at most $2^k - 1$ iterations in a single epoch, i.e. the potential must decrease by at least C during the 2^k consecutive iterations. $\square$

It remains to show that $\mathcal{X}$ is likely to be δ-sparse.

Theorem 4.12. *Let $\mathcal{X}$ be a σ-perturbed set of n points in $\mathbb{R}^d$. Then $\mathcal{X}$ is δ-sparse with probability at least $1 - n^{16kd+12}\left(\frac{n^4\delta}{\sigma}\right)^d$.*

Proof. First consider key-values a, b, c, d satisfying $a + b \not\equiv c + d$. We can write $a + b - c - d = \sum_{y\in\mathcal{X}} k_y y$ for some rationals k_y with denominator at most n^4. Since $a + b - c - d \not\equiv 0$, $\exists x \in \mathcal{X}$ with $k_x \neq 0$. Then $\|a + b - c - d\| \leq \delta$ iff x is in some ball of radius $\delta/|k_x|$, which by Lemma 4.3 ('Gaussian variable unlikely to be in a fixed ball') happens with probability at most $\left(\frac{\delta}{|k_x|\sigma}\right)^d \leq \left(\frac{n^4\delta}{\sigma}\right)^d$. Now,

$$\begin{aligned}
\mathbb{P}(\mathcal{X} \text{ is } \delta\text{-sparse}) &= \mathbb{P}\left(\bigcap_{a,b,c,d} \|a+b-c-d\| \leq \delta \implies a+b \equiv c+d\right) \\
&= 1 - \mathbb{P}\left(\bigcup_{a,b,c,d} \|a+b-c-d\| \leq \delta \text{ and } a+b \not\equiv c+d\right) \\
&\geq 1 - \sum_{\substack{a,b,c,d \\ a+b\not\equiv c+d}} \mathbb{P}(\|a+b-c-d\| \leq \delta) \\
&\geq 1 - \left(\binom{n}{4kd} n^3\right)^4 \left(\frac{n^4\delta}{\sigma}\right)^d \\
&\geq 1 - n^{16kd+12}\left(\frac{n^4\delta}{\sigma}\right)^d
\end{aligned}$$

The penultimate inequality is because there are $\leq \binom{n}{4kd}n^3$ ways to choose a key-value. $\square$

4.6 Large Cluster Changes

We begin the 'large changes' case with the following definition, which is illustrated in Figure 3.

Definition 4.13. *Take a point set $\mathcal{P}$ in $\mathbb{R}^d$ and $\epsilon > 0$. We say $\mathcal{P}$ is* ϵ-separated *if for any hyperplane $\mathcal{H}$, there are at most $2d$ points in $\mathcal{P}$ within distance ϵ of $\mathcal{H}$.*

The next theorem shows that if $\mathcal{X}$ is ϵ-separated, then the 'large change' causes a large potential drop. This theorem appeared to have an error in the original paper; the version below features alterations by me; see Appendix A for details.

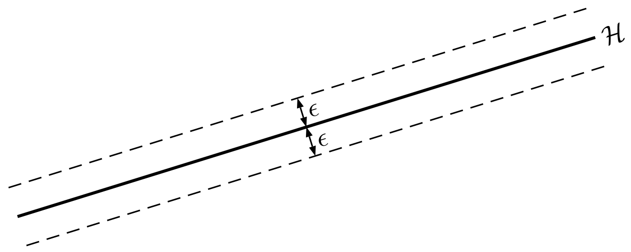


Figure 3: A point set $\mathcal{P}$ is ϵ-separated if for all hyperplanes $\mathcal{H}$ there are at most $2d$ points in the dotted area.

Theorem 4.14. *Suppose the k-means algorithm is run on an ϵ-separated point set $\mathcal{X} \subset \mathbb{R}^d$. Suppose that in a single iteration, one cluster loses at least $2kd$ points. Then the potential drops by at least $2\epsilon^2/n$ on that iteration. Similarly replacing 'loses' with 'gains'.*

Proof ([3] with corrections). Suppose a cluster $\mathcal{C}$ loses at least $2kd$ points in a single iteration. Then $\exists$ another cluster $\mathcal{C}'$ which gains at least $2d+1$ points from $\mathcal{C}$. Let S_1 and S_2 be the values of $\mathcal{C}$ before and after the iteration respectively, and similarly S_1' and S_2' for $\mathcal{C}'$. Let c and c' be the centres of S_1 and S_1'. Let $\mathcal{H}_2$ be the hyperplane bisecting c and c'. Let $\mathcal{H}_1$ be the previous Voronoi boundary between $\mathcal{C}$ and $\mathcal{C}'$. To clarify, the sequence of events is this:

(i) Draw the Voronoi boundary $\mathcal{H}_1$.

(ii) Assign points according to $\mathcal{H}_1$. $\mathcal{C} \leftarrow S_1$; $\mathcal{C}' \leftarrow S_1'$.

(iii) Recalculate centres. $c \leftarrow c(S_1)$; $c' \leftarrow c(S_1')$.

(iv) Draw the Voronoi boundary $\mathcal{H}_2$ which bisects c and c'.

(v) Assign points according to $\mathcal{H}_2$. $\mathcal{C} \leftarrow S_2$; $\mathcal{C}' \leftarrow S_2'$.

We know at least $2d+1$ points transfer from $\mathcal{C}$ to $\mathcal{C}'$ on step (v). As $\mathcal{X}$ is ϵ-separated, one of those points x is at least ϵ from $\mathcal{H}_2$ (see Figure 4). Therefore on step (v), the potential ϕ decreases by at least $\|x-c\|^2 - \|x-c'\|^2 = (2x - c - c').(c'-c)$. Now note x is at least ϵ from $\mathcal{H}_2$, and on the same side as c'; equivalently $(2x-c-c').(c'-c) \geq 2\epsilon\|c'-c\|$. So suffices to show $\|c'-c\| \geq \epsilon/n$.

Note that all points of S_1 are on one side of $\mathcal{H}_1$, so $c = c(S_1)$ is on the same side of $\mathcal{H}_1$. Similarly all points of S_1' are on the other side of $\mathcal{H}_1$, so c' is too. So $\mathcal{H}_1$ divides c and c' (see Figure 4). Suppose $\|c'-c\| < \epsilon/n$. Then $d(c, \mathcal{H}_1) < \epsilon/n$. But c is the centre of S_1 which has size $\leq n$ and is all on one side of $\mathcal{H}_1$. So all points of S_1 are within ϵ of $\mathcal{H}_1$ (if not, c must be at least ϵ/n from $\mathcal{H}_1$). See Figure 5. Then as $\mathcal{X}$ is ϵ-separated, $|S_1| \leq 2d$. Contradiction, as $|S_1| \geq 2d+1$ (because $\mathcal{C}$ loses at least $2d+1$ points to $\mathcal{C}'$ in the iteration).

In the case that $\mathcal{C}$ *gains* at least $2kd$ points, we proceed similarly, finding some $\mathcal{C}'$ that loses at least $2d+1$ points to $\mathcal{C}$, and one of those points x is at

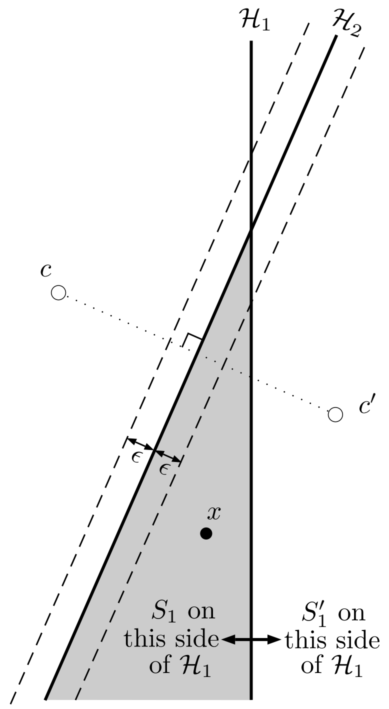

$\mathcal{H}_1$
$\mathcal{H}_2$
c
c'
ϵ
ϵ
x
S_1 on this side of $\mathcal{H}_1$
S_1' on this side of $\mathcal{H}_1$

Figure 4: There are at least $2d+1$ points of S_1 in the shaded area, one of which is x.

least ϵ from $\mathcal{H}_2$ (but on the same side as c). Then the potential decrease is at least $\|x-c'\|^2 - \|x-c\|^2 = (2x-c'-c).(c-c') \geq 2\epsilon\|c'-c\|$. If $\|c'-c\| < \epsilon/n$ then $d(c',\mathcal{H}_1) < \epsilon/n$, so all points of S_1' are within ϵ of $\mathcal{H}_2$, so $|S_1'| \leq 2d$. Contradiction, as $|S_1'| \geq 2d+1$ (as $\mathcal{C}'$ loses at least $2d+1$ points in the iteration). □

It remains to show $\mathcal{X}$ is likely to be ϵ-separated. We make use of the following technical lemma (a proof can be found in [3]).

Lemma 4.15. *Let $\mathcal{P}$ be a set of at least d points in $\mathbb{R}^d$, and let $\mathcal{H}$ be an arbitrary hyperplane. Then there exists a hyperplane $\mathcal{H}'$ passing through d points of $\mathcal{P}$ that satisfies*

$$\max_{p\in\mathcal{P}} \left(dist(p,\mathcal{H}')\right) \leq 2d \cdot \max_{p\in\mathcal{P}} \left(dist(p,\mathcal{H})\right). \tag{16}$$

Theorem 4.16. *Let $\mathcal{X}$ be a σ-perturbed set of n points in $\mathbb{R}^d$. Then $\mathcal{X}$ is ϵ-separated with probability at least $1 - n^{2d}\left(\frac{4d\epsilon}{\sigma}\right)^d$.*

Proof. First we show that if $\mathcal{X}$ is not ϵ-separated, then $\exists$ a hyperplane $\mathcal{H}$ passing through d points in $\mathcal{X}$ and within distance $2d\epsilon$ of d other points in $\mathcal{X}$: we have some hyperplane $\mathcal{J}$ s.t. $2d+1$ points of $\mathcal{X}$ are within ϵ of $\mathcal{J}$. Let $\mathcal{P}$ denote

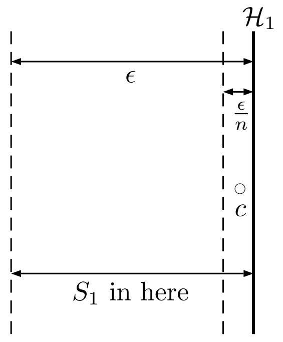


Figure 5: If $d(c, \mathcal{H}_1) < \frac{\epsilon}{n}$ then all points of S_1 are within ϵ of $\mathcal{H}_1$.

those $2d+1$ points. Then by Lemma 4.15, $\exists$ some $\mathcal{H}$ passing through d points of $\mathcal{P}$ such that $\max_{p\in\mathcal{P}}(\text{dist}(p,\mathcal{H})) \le 2d \cdot \max_{p\in\mathcal{P}}(\text{dist}(p,\mathcal{J})) \le 2d\epsilon$.

Therefore suffices to prove that there exists such a hyperplane with probability at most $n^{2d}\left(\frac{4d\epsilon}{\sigma}\right)^d$. First consider any disjoint $P_1, P_2 \subset \mathcal{X}$ with $|P_1| = |P_2| = d$, and let $\mathcal{H}$ be the hyperplane passing through all points of P_1. The probability that some $x \in P_2$ is within $2d\epsilon$ of $\mathcal{H}$ is at most $2d\epsilon/\sigma < 4d\epsilon/\sigma$ by applying Lemma 4.3 ('Gaussian variable unlikely to be in a fixed ball') in the one-dimensional case. So the probability that $\mathcal{H}$ is within $2d\epsilon$ of all points of P_2 is at most $(4d\epsilon/\sigma)^d$. So $\mathbb{P}(\exists$ such a hyperplane) is

$$\begin{aligned}
&\mathbb{P}\left(\bigcup_{\substack{\text{disj.}P_1,P_2\subset\mathcal{X}\\|P_1|=|P_2|=d}} \text{The } \mathcal{H} \text{ passing through } P_1 \text{ is within } 2d\epsilon \text{ of } P_2\right)\\
&\le \sum_{\substack{\text{disj.}P_1,P_2\subset\mathcal{X}\\|P_1|=|P_2|=d}} \left(\frac{4d\epsilon}{\sigma}\right)^d\\
&= \binom{n}{d}\binom{n-d}{d}\left(\frac{4d\epsilon}{\sigma}\right)^d\\
&\le n^d(n-d)^d\left(\frac{4d\epsilon}{\sigma}\right)^d\\
&\le n^{2d}\left(\frac{4d\epsilon}{\sigma}\right)^d.
\end{aligned}$$

□

4.7 Establishing the Bound

We are almost ready to prove the main theorem.

Theorem 4.17. *Let $\mathcal{X}$ be a σ-perturbed set of points in $\mathbb{R}^d$. Let D denote the* *diameter of $\mathcal{X}$. Then there exists a polynomial in n^k, $p^{-1/d}$, and D/σ which, with probability at least $1-2p$, is an upper bound on the number of iterations taken by the k-means method on $\mathcal{X}$.*

Proof. Take $C = \frac{\sigma^2 p^{2/d}}{4n^{32k+36}}$, $\delta = 2n^2\sqrt{C}$, and $\epsilon = \frac{\sigma p^{1/d}}{4dn^2}$. Then Theorem 4.16 says that

$$\mathbb{P}(\mathcal{X} \text{ is } \epsilon\text{-separated }) \geq 1 - n^{2d}\left(\frac{4d\epsilon}{\sigma}\right)^d = 1-p$$

and Theorem 4.12 says that

$$\mathbb{P}(\mathcal{X} \text{ is } \delta\text{-sparse}) \geq 1 - n^{16kd+12}\left(\frac{n^4\delta}{\sigma}\right)^d = 1 - n^{12(1-d)}p \geq 1-p.$$

So $\mathbb{P}(\mathcal{X}$ is δ-sparse and ϵ-separated$) \geq 1-2p$. In that case, we show that the potential drops by at least C every 2^k iterations. Given any sequence of 2^k iterations, either:

(i) No cluster gains or loses at least $2kd$ points in a single iteration, then Corollary 4.11 tells us that the potential drops by at least C over those 2^k iterations.

(ii) Otherwise some cluster gains or loses at least $2kd$ points in a single iteration, then Theorem 4.14 tells us that the potential drops by at least $\frac{2\epsilon^2}{n} = \frac{Cn^{32k+31}}{2d^2} \geq C$ on that iteration.

Then as the potential is at most nD^2 after the first iteration, the number of iterations is at most

$$\frac{2^k nD^2}{C} = \frac{2^{k+2}n^{32k+37}D^2}{\sigma^2 p^{2/d}} \leq n^{33k+39}(p^{-\frac{1}{d}})^2\left(\frac{D}{\sigma}\right)^2.$$

□

The main theorem follows by appropriate choice of p:

Theorem 4.18 (Main Theorem (Arthur, Vassilvitskii [3]))**.** *Fix an arbitrary set $\mathcal{X}' \subset \mathbb{R}^d$ of n points and assume that each point in $\mathcal{X}'$ is independently perturbed by a Gaussian distribution with mean 0 and variance σ^2, yielding a new set $\mathcal{X}$ of points, and let D denote the diameter of $\mathcal{X}$. Then the expected running time of the k-means method on $\mathcal{X}$ is bounded above by a polynomial in n^k and D/σ.*

Proof. The k-means method never takes more than $O(n^{kd})$ iterations (see Section 4.2), so taking $p = 1/O(n^{kd})$ we obtain, using the previous theorem,

$$\begin{aligned}\mathbb{E}(\#\text{iterations}) &\leq (1-2p)\textsc{Poly}(n^k, p^{-1/d}, D/\sigma) + 2p \cdot O(n^{kd})\\ &\leq \textsc{Poly}(n^k, D/\sigma).\end{aligned}$$

□

4.8 Polynomial Smoothed Complexity of k-Means Method

As noted in Section 4.2, the result we have just proved is a precursor to the proof that the k-means has polynomial smoothed complexity [1]. In this section we briefly indicate how the improvement was achieved. In [1], Arthur et al. write that Theorem 4.18's use of the $O(n^{kd})$ upper bound on the number of iterations is 'quite wasteful'. The problem is resolved by introducing the notion of *transition blueprints.*

> 'Such a blueprint is a description of an iteration of k-means that *almost* uniquely determines everything that happens during the iterations, which will dramatically reduce the number of cases that have to be considered'.

The idea is that these blueprints capture all the important information about an iteration needed for analysis, but the reduction in number of cases leads to the tighter bound given by Theorem 4.2.

5 The Perceptron Algorithm

The perceptron algorithm solves the following problem:

Problem A: Given a set of points $a_1, \dots, a_m \in \mathbb{R}^d$ and labels $l_1, \dots, l_m \in \{-1, +1\}$, find (if they exist) $w \in \mathbb{R}^d$ and $w_0 \in \mathbb{R}$ s.t. $a_i^T w > w_0$ for all i s.t. $l_i = +1$ and $a_i^T w < w_0$ for all i s.t. $l_i = -1$.

That is, given a set of points in d-dimensional space, each labelled 'positive' or 'negative', find a hyperplane separating the positive from the negative points, if one exists. This is a classic task in machine learning — the labelled points can be thought of as *training data*, which we use to construct a *perceptron* (a function that classifies future points as 'positive' or 'negative'). In this case, the perceptron we seek is a *linear classifier*. The perceptron algorithm can also be used to approximately solve linear programs: Suppose we want to maximise $c^T x$. We perform binary search on c_0, repeatedly using the perceptron algorithm to find feasible solutions that also satisfy $c^T x > c_0$. See [10] for details.

We can simplify the formulation of Problem A. First, we may write the constraint $a_i^T w > w_0$ as $(a_i, 1)^T (w, -w_0) > 0$, and similarly for the 'negative' constraints. Second, we may replace the 'negative' constraints $a_i^T w < 0$ with $(-a_i)^T w > 0$, making them 'positive' constraints. Therefore Problem A reduces to:

Problem B: Given a set of points $a_1, \dots, a_m \in \mathbb{R}^d$, find $w \in \mathbb{R}^d$ s.t. $a_i^T w > 0$ for all i, if it exists.

5.1 Description of the Perceptron Algorithm

The perceptron algorithm (often called the *perceptron learning algorithm* in the literature) solves Problem B as follows:

The Perceptron Algorithm

1. Initialise $\omega = \mathbf{0}$.
2. Pick some a_i such that $a_i^T \omega \leq 0$ and update ω by
$$\omega \leftarrow \omega + \frac{a_i}{|a_i|} \tag{17}$$
3. If we do not have $a_i^T \omega > 0$ for all i, go back to step 2.

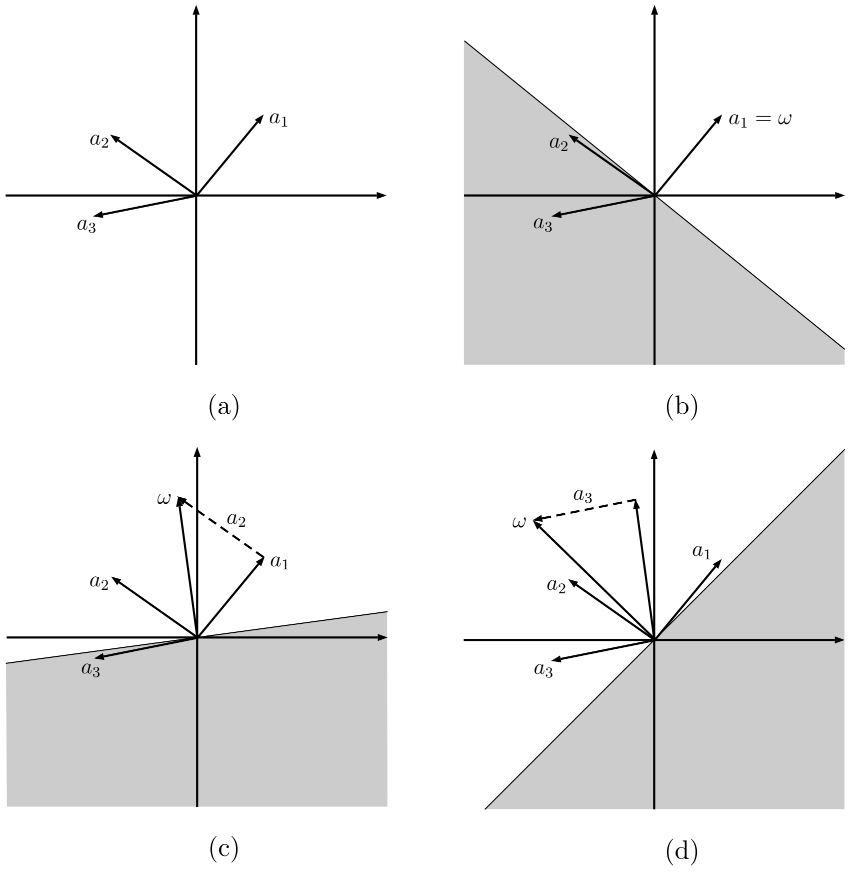


Figure 6: A demonstration of the perceptron algorithm with $d = 2$ and $m = 3$. In this example, $|a_i| = 1$ for all i. (a): The points a_1, a_2 and a_3. $\omega = \mathbf{0}$. (b): After updating with respect to a_1. (c): After updating with respect to a_2. (d): After updating with respect to a_3. Algorithm terminates as $a_i^T \omega > 0$ for all i.

See Figure 6 for a demonstration of the algorithm in action. The algorithm clearly doesn't terminate in the case when there is no solution. However in that case it is known that some value of ω must occur twice during the algorithm, provided the a_i have rational coefficients [23, Section 11.9]. Conversely, we will see from the proof of Theorem 5.2 that if there is a solution, ω cannot repeat a value. Therefore, assuming the a_i are rational, we may modify the algorithm so that it terminates, reporting 'no solution', when the problem is unsolvable. This is also useful for general $a_i \in \mathbb{R}^d$ which can be approximated by rationals.

It is also non-obvious from the definition that the algorithm terminates for solvable problems. This is proved in the next section by the Perceptron Convergence Theorem. This may be surprising, as the perceptron algorithm is a simple greedy algorithm, using only local information on each step to update

ω. Minsky and Papert offer an explanation in [23, Section 11.6] by likening the perceptron algorithm to a hill-climbing algorithm. Let ω^* be any solution. Then the function to be maximised (the hill function) is $\frac{\omega^T\omega^*}{|\omega|}$. Though we do not know ω^* when we are running the algorithm, it can be shown that our update step 'climbs the hill'. Minsky and Papert argue that (restricting ω to the unit ball) the hill is well-behaved: 'there are no false local maxima, no ridges, no plateaus', accounting for the effectiveness of this hill-climbing algorithm. In fact, this is the essence of the proof of the Perceptron Convergence Theorem.

One advantage of being a local greedy algorithm is that the perceptron algorithm can be used for *online learning* (if a new point is added during the running of the algorithm, the algorithm can continue).

5.2 Wiggle Room and the Perceptron Convergence Theorem

It turns out that in the solvable case, the running time of the perceptron algorithm can be bounded in terms of the problem's *wiggle room*, which essentially measures how much room there is to 'wiggle' inside the feasible region.

Definition 5.1. *Given some feasible ω, define the* wiggle room *of ω to be*

$$\nu(\omega) = \min_i \frac{a_i^T w}{|a_i||w|}. \tag{18}$$

That is, not only is ω feasible, but every ω' within an angle $\sin^{-1}(\nu(\omega))$ of ω is feasible. Given an instance of the perceptron learning problem, the *wiggle room of the problem* is the wiggle room of the feasible solution with maximal wiggle room. The following theorem, proved independently by Block and Novikoff in 1962 [9, 24], shows that problems with larger wiggle room are solved more quickly.

Theorem 5.2 (Perceptron Convergence Theorem (Block-Novikoff 1962))**.** *When the perceptron algorithm is run on a solvable problem with wiggle room ν, it terminates in at most $1/\nu^2$ iterations.*

Proof [10]. Take some ω^* with wiggle room ν. We may assume ω^* is a unit vector (as scaling ω^* does not change the value of ν). Consider the value of $\omega^T\omega^*$ on each step. It starts as 0, and on each step it increases by at least ν, as $\left(\omega + \frac{a_i}{|a_i|}\right)^T \omega^* = \omega^T\omega^* + \frac{a_i^T\omega^*}{|a_i|} \geq \omega^T\omega^* + \nu$. However by Cauchy-Schwarz $\omega^T\omega^* \leq |\omega|$ for all ω, as ω^* is a unit vector.

Now consider the value of $|\omega|^2$ on each step. It never increases by more than 1 in a single step because $\left(\omega + \frac{a_i}{|a_i|}\right)^2 = \omega^2 + 2\frac{a_i^T\omega}{|a_i|} + 1 \leq \omega^2 + 1$, as $a_i^T\omega \leq 0$ in order for i to have been chosen for the update step. Putting these observations together, we see that after the first t iterations,

$$\nu t \leq \omega^T\omega^* \leq |\omega| \leq \sqrt{t} \tag{19}$$

so $t \leq 1/\nu^2$. □

5.3 Analysis of the Perceptron Algorithm

The perceptron algorithm was invented in 1957 by Rosenblatt [28]. It is known to have exponential worst-case complexity but 'fairly efficient' average-case complexity (under some sensible distribution) [27, Section 4.4]. That, and the algorithm's wide use in machine learning, motivates a smoothed analysis. We will present Blum and Dunagan's 2002 smoothed analysis of the perceptron algorithm [10].

Blum and Dunagan use the following perturbation model. Let M be Problem B (given on page 21) with the extra assumption that $|a_i| \leq 1$ for all i. Let $d_i = a_i + \sigma g_i$, where each g_i is chosen independently according to a d-dimensional Gaussian distribution of mean 0 and variance 1. Then $\tilde{M}$ is Problem B but with input points d_i. This is the obvious choice of perturbation model for this problem.

Theorem 5.3 (Smoothed Complexity of Perceptron [10])**.** *Let M be a perceptron problem with $|a_i| \leq 1$ and let $\tilde{M}$ be M under a Gaussian perturbation of variance σ^2, where $\sigma^2 \leq 1/2d$. For any δ, with probability at least $1-\delta$, either*

(i) the perceptron algorithm finds a solution to $\tilde{M}$ in $O\left(\frac{d^3 m^8 \log^8(m/\delta)}{\sigma^2 \delta^8}\right)$ iterations, or

(ii) $\tilde{M}$ is infeasible.

Theorem 5.3 shows that the perceptron algorithm has *probably polynomial smoothed complexity* in the sense of Equation (5). The condition $\sigma^2 \leq 1/2d$ is another limitation of the result; as the bound only tells us about close to worst-case behaviour, and not close to average-case behaviour. However, we are more interested in behaviour as we approach worst-case.

5.4 Overview of the Proof

We will prove Theorem 5.3 in full, excepting some technical lemmas that are not of interest to the main argument. All proofs are due to [10] unless otherwise stated, though all diagrams are by me. Blum and Dunagan's original paper analysed the perceptron algorithm in the context of solving linear programs in order that their result could be directly compared to Spielman and Teng's simplex result [30]. Consequently their proof is tailored to that setting, considering only perceptron problems with the structure of a LP. Here we present a smoothed analysis of the algorithm for the original perceptron problem, significantly simplifying the exposition, and the proofs of Lemmas 5.10 and 5.11. Lastly, there appeared to be a small error in the original proof of Lemma 5.10. My correction has resulted in slight differences in the bounds in the statements of Lemmas 5.10, 5.11 and Theorem 5.12. However, the bound of Theorem 5.12 is still polynomial and so is essentially unchanged. See Appendix A for details.

The proof proceeds as follows. First we give two preliminary lemmas (Section 5.5), then we prove the Brunn-Minkowski Theorem from the Brunn-Minkowski

Inequality (Section 5.6). In Section 5.7 we prove two lemmas (one using the Brunn-Minkowski Theorem), showing that the wiggle room is unlikely to be small. The main result follows by the Perceptron Convergence Theorem.

5.5 Preliminaries

This section may be skipped now and referred to later as needed. We will state without proof the following technical lemma, which states that 'small boundaries are easily missed' by a variable with Gaussian distribution. While the statement of this lemma appears easily believable, the proof (which is given in the appendix of [10]) is surprisingly long. However, no details of the proof are relevant to our main argument.

Lemma 5.4 ('Small boundaries are easily missed')**.** *Let K be an arbitrary convex body, and let $\Delta(K,\epsilon)$ denote the ϵ-boundary of K, i.e.*

$$\Delta(K,\epsilon) = \{x : \exists x' \in K, |x - x'| \leq \epsilon\} \setminus K. \tag{20}$$

Let g have d-dimensional Gaussian distribution with mean $\overline{g}$ and variance σ^2. Then

$$\mathbb{P}(g \in \Delta(K,\epsilon)) = O\left(\frac{\epsilon\sqrt{d}}{\sigma}\right). \tag{21}$$

We will also require the following bound on the sum of squared Gaussians.

Lemma 5.5 ('Sum of squared Gaussians')**.** *Let $X_1, \ldots, X_d$ be independent $N(0,\sigma)$ random variables. Then*

$$\mathbb{P}\left(\sum_{i=1}^{d} X_i^2 \geq \kappa^2\right) \leq e^{\frac{d}{2}\left(1-\frac{\kappa^2}{d\sigma^2}+\log\frac{\kappa^2}{d\sigma^2}\right)}. \tag{22}$$

Proof. We derive a standard Chernoff bound. First consider $Y_i \sim N(0,1)$. For $Y \sim N(0,1)$, a simple integration shows that $\mathbb{E}(e^{tY^2}) = (1-2t)^{-1/2}$ for all $t > 1/2$. Now

$$\begin{aligned}
\mathbb{P}\left(\sum_{i=1}^{d} Y_i^2 \geq k\right) &= \mathbb{P}\left(e^{t\sum_{i=1}^{d} Y_i^2} \geq e^{tk}\right) && \forall t > 0\\
&\leq \mathbb{E}\left(e^{t\sum_{i=1}^{d} Y_i^2}\right)e^{-tk} && \text{by Markov's Inequality}\\
&= (1-2t)^{-d/2}e^{-tk}\\
&\leq \left(\frac{k}{d}\right)^{d/2} e^{-\frac{k}{2}+\frac{d}{2}} && \text{taking optimal } t = \frac{1}{2}-\frac{d}{2k}\\
&= e^{\frac{d}{2}\left(1-\frac{k}{d}+\log\frac{k}{d}\right)}.
\end{aligned}$$

Then as $\mathbb{P}(\sum_{i=1}^{d} X_i^2 \geq \kappa^2) = \mathbb{P}(\sum_{i=1}^{d} Y_i^2 \geq \frac{\kappa^2}{\sigma^2})$, we obtain the desired bound. □

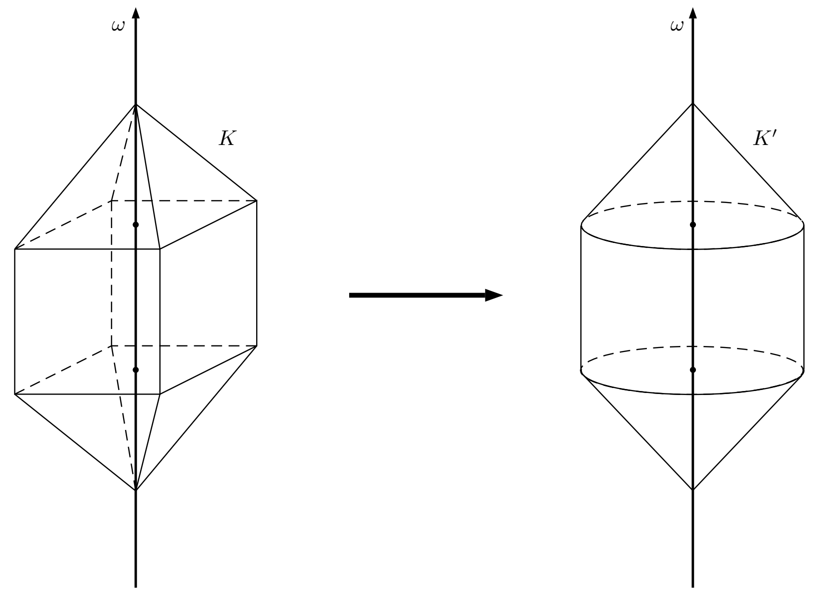


Figure 7: An example Schwarz rounding for $d = 3$.

5.6 The Brunn-Minkowski Theorem

We now turn to the Brunn-Minkowski Theorem, which we will need for our main proof. It will follow from the well-known Brunn-Minkowski Inequality, given below.

Theorem 5.6 (The Brunn-Minkowski Inequality)**.** *For convex bodies A, B in $\mathbb{R}^n$ and for $0 \le \lambda \le 1$,*

$$(vol_n((1-\lambda)A + \lambda B))^{1/n} \ge (1-\lambda)(vol_n(A))^{1/n} + \lambda(vol_n(B))^{1/n}$$

where $+$ denotes the vector sum*: $X + Y = \{x + y : x \in X, y \in Y\}$.*

A proof of the inequality as stated here can be found in [29, Section 7.1]. The inequality is a powerful fundamental tool for both geometry and analysis; see [14] for an extensive survey.

Theorem 5.7 (The Brunn-Minkowski Theorem)**.** *Let K be a d-dimensional convex body, and let $\overline{x}$ denote the centre of mass of K, $\overline{x} = \mathbb{E}_{x \in K} x$. Then $\forall w \in \mathbb{R}^d$,*

$$\frac{\max_{x \in K} w^T(x - \overline{x})}{\max_{x \in K} w^T(\overline{x} - x)} \le d. \tag{23}$$

Proof [10]. Let K and ω be fixed. Wlog ω is a unit vector, and $\overline{x}$ is the origin. Let K' be the body that is rotationally symmetric about ω and has the same $(d-1)$-dimensional volume for every cross-section $K_r = \{x : x \in K, w^T x = r\}$.

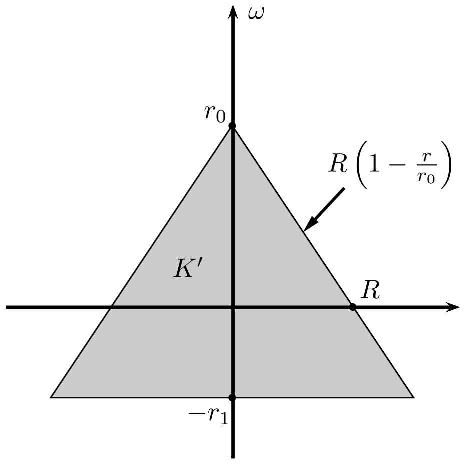


Figure 8: The smallest possible value of r_1 is given by this situation, where K' is a cone.

That is, $vol_{d-1}(K_r) = vol_{d-1}(K'_r)$ for all r. This transformation is called a *Schwarz rounding*; see Figure 7. Clearly K' has the same centre of mass as K.

We claim K' is convex. To that end take $r_1, r_2 \in \mathbb{R}$. Suffices to show

$$radius\left(K'_{\frac{r_1+r_2}{2}}\right) \le \frac{1}{2} radius(K'_{r_1}) + \frac{1}{2} radius(K'_{r_2}). \tag{24}$$

Let $A = K_{r_1}$, $B = K_{r_2}$. As K is convex, $\frac{1}{2}A + \frac{1}{2}B \subseteq K_{\frac{r_1+r_2}{2}}$ (where $+$ denotes the vector sum). So

$$\begin{aligned} vol_{d-1}\left(K_{\frac{r_1+r_2}{2}}\right)^{\frac{1}{d-1}} &\ge vol_{d-1}\left(\frac{1}{2}A + \frac{1}{2}B\right)^{\frac{1}{d-1}} \\ &\ge \frac{1}{2} vol_{d-1}(A)^{\frac{1}{d-1}} + \frac{1}{2} vol_{d-1}(B)^{\frac{1}{d-1}} && \text{(B-M ineq)} \\ &= \frac{1}{2} vol_{d-1}(K'_{r_1})^{\frac{1}{d-1}} + \frac{1}{2} vol_{d-1}(K'_{r_2})^{\frac{1}{d-1}} && \text{(def of } K'). \end{aligned}$$

Then as the radius of a $(d-1)$-dimensional sphere is proportional to the $\frac{1}{d-1}$th power of its volume, we deduce Equation (24). So K' is convex.

Let $r_0 = \max_{x \in K} w^T x$, $r_1 = \max_{x \in K}(-w^T x)$. We wish to show that $r_0/r_1 \le d$, equivalently, $r_1 \ge r_0/d$. Let $R = radius(K'_0)$. By convexity of K', $radius(K'_r) \ge R(1 - \frac{r}{r_0})$ for all $r \in [0, r_0]$. Similarly $radius(K'_r) \le R(1 - \frac{r}{r_0})$ for all $r \in [-r_1, 0]$. See Figure 8.

However, K' has the origin as its centre of mass, so the least possible value for r_1 is given by

$$\int_{r=0}^{r_1} r\left(1 + \frac{r}{r_0}\right)^{d-1} dr = \int_{r=0}^{r_0} r\left(1 - \frac{r}{r_0}\right)^{d-1} dr. \tag{25}$$

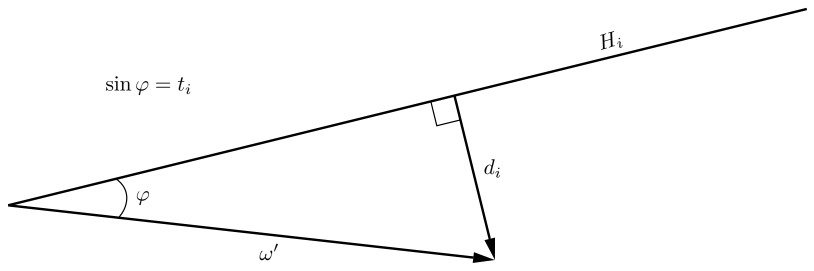


Figure 9: Illustration of the definition of t_i in two dimensions. H_i is the hyperplane representing the ith constraint. φ is maximum possible for feasible ω'.

Integrating by parts yields $r_1 = r_0/d$ as desired. □

The above theorem is very similar to [15, Theorem 5.7, p.56], which states that the Schwarz rounding of a convex body is convex. Interestingly, in [15], that fact is used to prove the Brunn-Minkowski Inequality (not the reverse). We will use the following corollary of the Brunn-Minkowski Theorem in our main proof. It says that given a convex body and a tangent hyperplane, if the body contains a point far from the hyperplane then the centre of mass of the body is also far from the hyperplane.

Corollary 5.8 ('Tangent hyperplane to a convex body')**.** *Let K be a d-dimensional convex body with centre of mass $\overline{x}$, and H a $(d-1)$-dimensional hyperplane tangent to K. If $\max_{x\in K} d(H,x) \geq t$, then $d(\overline{x}, H) \geq \frac{t}{d+1}$.*

Proof. Wlog we may assume H is of the form $\{x : a^T x = 0\}$ with a a unit vector, and that $a^T x \geq 0$ for all $x \in K$. The Brunn-Minkowski Theorem tells us that

$$\frac{\max_{x\in K} a^T(x-\overline{x})}{\max_{x\in K} a^T(\overline{x}-x)} \leq d$$

or equivalently,

$$(d+1)a^T\overline{x} \geq \max_{x\in K} a^T x - d\max_{x\in K}(-a^T x).$$

Note $\max_{x\in K} a^T x = \max_{x\in K} d(H,x) \geq t$ by assumption and $\max_{x\in K}(-a^T x) = 0$ as K is tangent to H. So $d(\overline{x}, H) = a^T\overline{x} \geq \frac{t}{d+1}$. □

5.7 Proof of the Main Theorem

Recall that $\tilde{M}$ is the perturbed version of the perceptron problem M, with input points $d_i = a_i + \sigma g_i$.

Definition 5.9. *When $\tilde{M}$ is feasible, we define*

$$t_i = \max_{w' \text{ feasible for } \tilde{M}} \frac{d_i^T w'}{|d_i||w'|}. \tag{26}$$

That is, t_i is the cosine between d_i and ω', or the sine of the maximum angle between a feasible point and the hyperplane H_i representing the i^{th} constraint. See Figure 9. The following lemma shows that for feasible $\tilde{M}$, t_i is unlikely to be small. The proof works by showing that for t_i to be small, d_i must hit some small boundary, then applying Lemma 5.4 ('small boundaries are easily missed'). This lemma appeared to have a small error in the original paper which has been corrected here; see Appendix A for details.

Lemma 5.10 ('t_i is unlikely to be small')**.** *Fix* $i \in \{1, \dots, m\}$. *Then*

$$\mathbb{P}(\tilde{M} \textit{ is feasible and } t_i \leq \epsilon) = O\left(\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4} \log \frac{\sigma}{\epsilon\sqrt{d}}\right) \tag{27}$$

as $\epsilon \to 0$.

Proof ([10] with corrections). Suppose all a_j, $j \neq i$, have already been perturbed, i.e. we have d_j for all $j \neq i$. Let $R = \{w : d_j^T w > 0 \,\forall j \neq i\}$, the set of points satisfying these constraints. We show that no matter what R is, the random perturbation of a_i will be enough to prove the lemma. If R is empty then $\tilde{M}$ is infeasible no matter what d_i is, so we may assume R is nonempty. Let $D = \{d_i : d_i^T w \leq 0 \,\forall w \in R\}$, the set of values d_i could take on so that $\tilde{M}$ is infeasible. Note that D is a convex cone from the origin. Let $F = \{d_i : \exists d_i' \in D : \frac{d_i^T d_i'}{|d_i|.|d_i'|} \geq \sqrt{1-\epsilon^2}\} \setminus D$. F is an 'ϵ-boundary' to D in the *angle* sense: F consists of points d_i which are at an angle with sine less than ϵ from a point d_i' of D. That is, the points of F are close to choices of d_i making $\tilde{M}$ infeasible. We claim that F is precisely those d_i that make $t_i \leq \epsilon$. The proof is supplied in the appendix of [10].

It now suffices to show that $\mathbb{P}(d_i \in F) = O\left(\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4} \log \frac{\sigma}{\epsilon\sqrt{d}}\right)$. In order to use Lemma 5.4 ('small boundaries are easily missed'), we wish to show that F is contained in an 'ϵ-boundary' of D in a *distance* sense. As small angle implies small distance only for small values of $|d_i|$, we will condition on the probability that $|d_i|$ is large. See Figure 10.

Take any $\kappa \geq 1$. We have $d_i = a_i + \sigma g_i$ where $|a_i| \leq 1$. So by Lemma 5.5 ('sum of squared Gaussians'):

$$\begin{aligned} \mathbb{P}(|d_i| \geq \kappa + 1) &\leq \mathbb{P}(|\sigma g_i| \geq \kappa) && \text{as } |a_i| \leq 1 \\ &= \mathbb{P}(|\sigma g_i|^2 \geq \kappa^2) \\ &\leq e^{\frac{d}{2}\left(1 - \frac{\kappa^2}{d\sigma^2} + \log \frac{\kappa^2}{d\sigma^2}\right)} && \text{by Lemma 5.5.} \end{aligned} \tag{28}$$

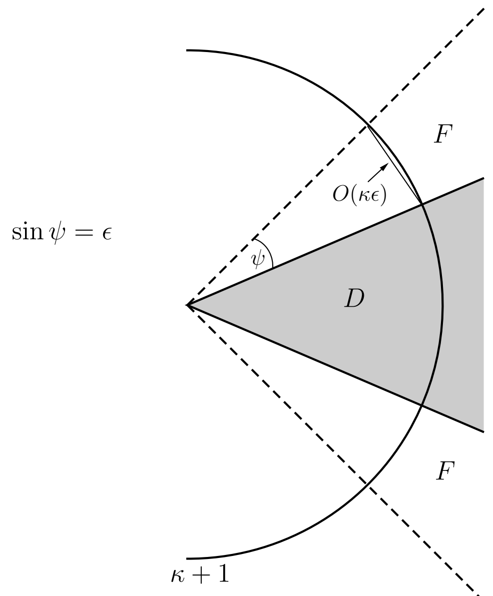


Figure 10: The shaded area D is a convex cone. F consists of points which are at an angle with sine less than ϵ from a point of D. Points of F with magnitude less than $\kappa+1$ are within $O(\kappa\epsilon)$ of a point of D.

Now note that

$$\begin{aligned}\frac{d}{2}\left(1-\frac{\kappa^2}{d\sigma^2}+\log\frac{\kappa^2}{d\sigma^2}\right) &\leq \frac{d}{2}\left(1-2\kappa^2+\log 2\kappa^2\right) && \text{(see explanation below)}\\ &\leq \frac{d}{2}\left(-2\kappa^2+\frac{3}{2}\kappa^2\right) && \text{for } \kappa\geq 1 \text{ (computation)}\\ &\leq \frac{1}{2}\left(-\frac{\kappa^2}{2}\right) && \text{as } d\geq 1\\ &= -\frac{\kappa^2}{4}\end{aligned} \tag{29}$$

where the first inequality is because the function $f(x)=x-\log x$ is increasing for $x\geq 1$, and $\sigma^2\leq 1/2d$ so $\frac{\kappa^2}{d\sigma^2}\geq 2\kappa^2>1$. Putting together Equations (28) and (29), we obtain $\mathbb{P}(|d_i|\geq\kappa+1)\leq e^{-\kappa^2/4}$ for all $\kappa\geq 1$.

Now, we show that if $|d_i|\leq\kappa+1$ and $d_i\in F$, then d_i is within $O(\kappa\epsilon)$ of a point of D. See Figure 10. There exists a point $d_i'\in D$ at an angle with sine at most ϵ from d_i. Wlog we may assume $|d_i'|=|d_i|$. Then by the cosine rule, $|d_i-d_i'|^2\leq 2(\kappa+1)^2(1-\sqrt{1-\epsilon^2})=O(\kappa^2\epsilon^2)$ so $|d_i-d_i'|=O(\kappa\epsilon)$. Now let H be the event that $d_i\in F$. By Lemma 5.4 ('small boundaries are easily missed'),

$\mathbb{P}(H|d_i \le \kappa+1) = O\left(\frac{\kappa\epsilon\sqrt{d}}{\sigma}\right)$. So

$$\begin{aligned}\mathbb{P}(H) &= \mathbb{P}(H|d_i \le \kappa+1)\mathbb{P}(d_i \le \kappa+1) + \mathbb{P}(H|d_i > \kappa+1)\mathbb{P}(d_i > \kappa+1)\\ &\le O\left(\frac{\kappa\epsilon\sqrt{d}}{\sigma}\right)\cdot 1 + 1\cdot e^{-\frac{\kappa^2}{4}}.\end{aligned}$$

We now choose $\kappa = \log(\sigma/\epsilon\sqrt{d}) \ge 1$ and note $e^{-\frac{\kappa^2}{4}} \le e^{-\frac{\kappa}{4}}$ to obtain

$$O\left(\frac{\epsilon\sqrt{d}}{\sigma}\log\left(\frac{\sigma}{\epsilon\sqrt{d}}\right)\right) + \left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4} = O\left(\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4}\log\frac{\sigma}{\epsilon\sqrt{d}}\right)$$

as $\epsilon \to 0$. □

We have shown that t_i is unlikely to be small. By showing that if all t_i are large, then $\tilde{M}$ has large wiggle room, we can now prove that the wiggle room is unlikely to be small. This proof will use the corollary of the Brunn-Minkowski Theorem from Section 5.6.

Lemma 5.11 ('Wiggle room is unlikely to be small')**.** *Take any $\nu > 0$. Let E be the event that $\tilde{M}$ is feasible yet contains no solution of wiggle room ν. Then*

$$\mathbb{P}(E) = O\left(m\left(\frac{d^{1.5}\nu}{\sigma}\right)^{1/4}\log\frac{\sigma}{d^{1.5}\nu}\right) \tag{30}$$

as $\nu \to 0$.

Proof. Setting $\epsilon = 2(d+1)\nu$, we find

$$\begin{aligned}\mathbb{P}(\tilde{M} \text{ is feasible and } \exists i\colon t_i \le \epsilon) &\le \sum_{i=1}^{m}\mathbb{P}(\tilde{M} \text{ is feasible and } t_i \le \epsilon)\\ &\le m\,O\left(\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4}\log\frac{\sigma}{\epsilon\sqrt{d}}\right) \qquad \text{(Lemma 5.10)}\\ &= O\left(m\left(\frac{d^{1.5}\nu}{\sigma}\right)^{1/4}\log\frac{\sigma}{d^{1.5}\nu}\right).\end{aligned}$$

Therefore it suffices to show that if $\tilde{M}$ is feasible and there is no solution of wiggle room ν, then $t_i \le \epsilon$ for some i. To prove the contrapositive, suppose $t_i > \epsilon$ for all i. Let ω^* be the unit vector that satisfies $\tilde{M}$ with maximum wiggle room ν', and suppose for contradiction that $\nu' < \nu$.

Recall that the feasible region $\tilde{M}$ is a d-dimensional cone. Let $\tilde{M}' = \tilde{M} \cap \{\omega : \omega^T\omega^* = 1\}$, a $(d-1)$-dimensional hyperplane. $\tilde{M}'$ is clearly convex. Recall that ω^* has wiggle room ν'. This means that any vector at an angle with sine less than ν' from ω^* is feasible. Therefore in $\tilde{M}'$, ω^* is the centre of a $(d-1)$-dimensional

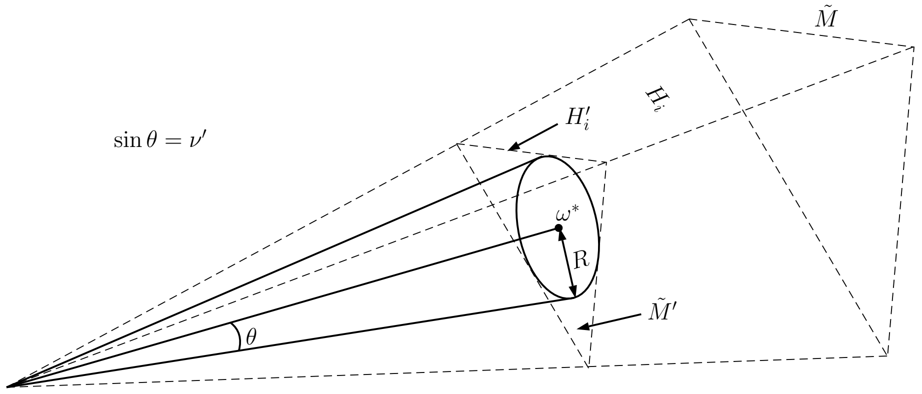


Figure 11: An example with $d = 3$ and three constraints ($m = 3$).

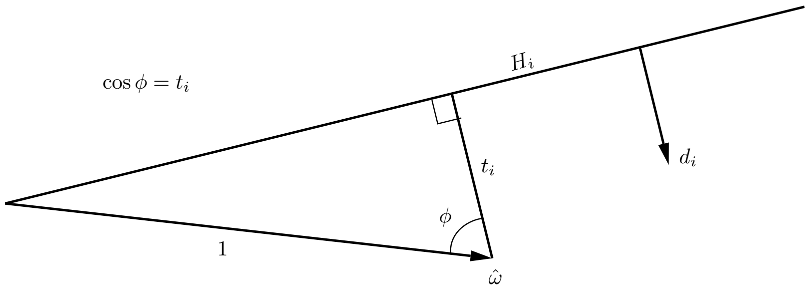


Figure 12: $\hat{\omega}$ is distance exactly t_i from H_i.

sphere of radius $R = \frac{\nu'}{\sqrt{1-\nu'^2}} \leq 2\nu'$ for $\nu' \leq 1/2$ (we may assume $\nu' \leq 1/2$ as we are considering $\nu \to 0$). The sphere is in contact with the bounding constraints so ω^* forms the centre of a sphere of *maximum* radius over all spheres lying within $\tilde{M}'$. Let $H_i = \{\omega : d_i^T\omega = 0\}$ denote the hyperplane representing the i^{th} constraint, and $H_i' = H_i \cap \{\omega : \omega^T\omega^* = 1\}$. See Figure 11 for an illustration of the construction so far.

Define $s_i = \max_{\omega' \in \tilde{M}'} d(\omega', H_i')$, the furthest distance of a point in $\tilde{M}'$ from H_i'. We claim $s_i \geq t_i$ for all i. To that end, fix i and take $\hat{\omega} \in \tilde{M}$ a unit vector such that $\frac{d_i^T\hat{\omega}}{|d_i|} = t_i$ (we can do this by definition of t_i). Then $\hat{\omega}$ is distance exactly t_i from H_i; see Figure 12 to justify this. Now take $\hat{\omega}'$ a scalar multiple of $\hat{\omega}$ such that $\hat{\omega}' \in \tilde{M}'$. Then $|\hat{\omega'}| \geq |\hat{\omega}|$ so $\hat{\omega}'$ is distance at least t_i from H_i. So

$$t_i \leq d(\hat{\omega}', H_i) \leq d(\hat{\omega}', H_i') \leq s_i. \tag{31}$$

Let $\overline{\omega} = \mathbb{E}_{\omega \in \tilde{M}'}(\omega)$, the centre of mass of $\tilde{M}'$. We will show that $\overline{\omega}$ is the centre of a sphere of radius $> 2\nu'$ contained in $\tilde{M}'$, deriving a contradiction. We apply Corollary 5.8 ('tangent hyperplane to a convex body') to the $(d-1)$-dimensional convex body $\tilde{M}'$ with tangent hyperplane H_i'. As

$\max_{\omega' \in \tilde{M}'} d(H_i', \omega') = s_i \geq t_i$, then $d(\overline{\omega}, H_i') \geq \frac{t_i}{d} > \frac{\epsilon}{d} = \frac{4(d+1)\nu}{d} > 4\nu > 4\nu'$. So $\overline{\omega}$ is the centre of a sphere of radius $4\nu'$ contained in $\tilde{M}'$, a contradiction. $\square$

We can now prove the main theorem.

Theorem 5.12 (Smoothed Complexity of Perceptron [10]). *Let M be a perceptron problem with $|a_i| \leq 1$ and let $\tilde{M}$ be M under a Gaussian perturbation of variance σ^2, where $\sigma^2 \leq 1/2d$. For any δ, with probability at least $1-\delta$, either*

(i) the perceptron algorithm finds a solution to $\tilde{M}$ in $O\left(\frac{d^3 m^8 \log^8(m/\delta)}{\sigma^2 \delta^8}\right)$ iterations, or

(ii) $\tilde{M}$ is infeasible.

Proof ([10] with alterations). Set $\nu = O\left(\frac{\sigma}{d^{1.5}}\left(\frac{\delta}{m}\right)^4 \frac{1}{\log^4(m/\delta)}\right)$. Lemma 5.11 ('wiggle room is unlikely to be small') tells us that the probability that $\tilde{M}$ is feasible yet has no solution of wiggle room ν is at most

$$\begin{aligned} O\left(m\left(\frac{d^{1.5}\nu}{\sigma}\right)^{1/4} \log \frac{\sigma}{d^{1.5}\nu}\right) &= O\left(\delta \cdot \frac{\log\left(\left(\frac{m}{\delta}\right)^4 \log^4\left(\frac{m}{\delta}\right)\right)}{\log\left(\frac{m}{\delta}\right)}\right) \\ &= O(5\delta) = O(\delta) \end{aligned}$$

where the second equality is because

$$\frac{\log(x^4 \log^4 x)}{\log x} = \frac{4\log x + 4\log\log x}{\log x} = 4 + \frac{4\log\log x}{\log x}$$

and this is ≤ 5 for x sufficiently large, as the second term tends to zero as $x \to \infty$. Lastly, the Perceptron Convergence Theorem tells us that if $\tilde{M}$ has wiggle room ν, then the perceptron algorithm terminates in $O(1/\nu^2) = \left(\frac{d^3 m^8 \log^8(m/\delta)}{\sigma^2 \delta^8}\right)$ iterations. $\square$

6 Closing Remarks

In this essay we have seen the need for smoothed analysis as a tool to close the observed gaps between theory and practice, as well as its application to two problems in machine learning. In this section we briefly discuss some of the pitfalls to avoid when performing smoothed analysis, some interesting future work, and reflect on what we have seen.

6.1 Subtleties of Smoothed Analysis

Before performing smoothed analysis, one needs to choose the perturbation model carefully. In some settings, it may be appropriate to use a non-Gaussian distribution, or to apply the perturbation multiplicatively rather than additively (Spielman and Teng call the latter a *relative perturbation* [30]). There is also the question of what parts of the input to perturb. For example, the input of a linear program (see Equation (1) on p4) is the vectors c and y and the matrix A. Which of these should be perturbed? In [30], Spielman and Teng choose to perturb y and A.

When we apply our perturbation, we want to avoid destroying important properties of the input e.g. we may wish to preserve the zeros of a matrix, the feasibility of a linear program, or the planarity of a graph. These are called *property-preserving perturbations* [31]. Lastly, what is a sensible perturbation model for discrete problems e.g. sorting algorithms where the input is an ordered list? The choice of model is important, as shown by Fouz et al. [13] who consider two different perturbation models (*additive noise* and *partial permutations*) for a smoothed analysis of quicksort, and obtain differing results for each. Spielman and Teng discuss perturbation models for discrete problems in [31].

6.2 Open Problems and Future Work

There are a number of open problems in smoothed analysis; see [32] for an overview. Spielman and Teng's original result [30] on the simplex algorithm applied to the *shadow pivot rule* version of the algorithm. What is the smoothed complexity under other pivot rules? Many existing smoothed analyses do not use property-preserving perturbations when it may be appropriate. Do these results still hold under property-preserving perturbations?

Smoothed analysis appears to be very difficult to carry out; most papers proving the smoothed complexity of an algorithm are quite long and intricate. It would be easier if smoothed complexity was more integrated into existing complexity theory. For example, is there a notion of completeness for smoothed complexity? Can we relate smoothed complexity to hardness of approximation? The only general theoretical result I have seen is Beier and Vöcking's result that a binary optimisation problem has polynomial smoothed complexity iff it has pseudopolynomial (worst-case) complexity [7].

6.3 Conclusion

Smoothed analysis is a valuable tool to help us explain and predict the behaviour of algorithms. As a hybrid of worst- and average-case complexity, it has the advantages of both. It has successfully explained the good performance of the simplex algorithm, the k-means method, the perceptron algorithm, and many others. In particular, smoothed analysis has the potential to aid the understanding of many other machine learning algorithms.

References

[1] D. Arthur, B. Manthey, and H. Roglin. k-means has polynomial smoothed complexity. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 405–414. IEEE Computer Society Press, 2009.

[2] D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *Proceedings of the 22nd Annual Symposium on Computational Geometry*, pages 144–153. ACM, 2006.

[3] D. Arthur and S. Vassilvitskii. Worst-case and smoothed analysis of the ICP algorithm, with an application to the k-means method. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 153–164. IEEE Computer Society Press, 2006.

[4] C. Banderier, R. Beier, and K. Mehlhorn. Smoothed analysis of three combinatorial problems. In *Mathematical Foundations of Computer Science 2003*, pages 198–207. Springer, 2003.

[5] R. Beier, H. Röglin, and B. Vöcking. The smoothed number of Pareto optimal solutions in bicriteria integer optimization. In *Integer Programming and Combinatorial Optimization*, pages 53–67. Springer, 2007.

[6] R. Beier and B. Vöcking. Random knapsack in expected polynomial time. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pages 232–241. ACM, 2003.

[7] R. Beier and B. Vöcking. Typical properties of winners and losers in discrete optimization. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 343–352. ACM, 2004.

[8] P. Berkhin. A survey of clustering data mining techniques. In *Grouping Multidimensional Data*, pages 25–71. Springer, 2006.

[9] H. Block. The perceptron: a model for brain functioning. I. *Reviews of Modern Physics*, 34(1):123, 1962.

[10] A. Blum and J. Dunagan. Smoothed analysis of the perceptron algorithm for linear programming. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 905–914. Society for Industrial and Applied Mathematics, 2002.

[11] G. B. Dantzig. Maximization of a linear function of variables subject to linear inequalities, in activity analysis of production and allocation. 1951.

[12] M. Englert, H. Röglin, and B. Vöcking. Worst case and probabilistic analysis of the 2-opt algorithm for the TSP. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1295–1304. Society for Industrial and Applied Mathematics, 2007.

[13] M. Fouz, M. Kufleitner, B. Manthey, and N. Z. Jahromi. On smoothed analysis of quicksort and Hoare's find. *Algorithmica*, 62(3-4):879–905, 2012.

[14] R. Gardner. The Brunn-Minkowski inequality. *Bulletin of the American Mathematical Society*, 39(3):355–405, 2002.

[15] P. M. Gruber and J. M. Wills. *Handbook of Convex Geometry*, volume A. North Holland, 1993.

[16] M. Inaba, N. Katoh, and H. Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th Annual Symposium on Computational Geometry*, pages 332–339. ACM, 1994.

[17] A. T. Kalai, A. Samorodnitsky, and S.-H. Teng. Learning and smoothed analysis. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 395–404. IEEE Computer Society Press, 2009.

[18] A. T. Kalai and S.-H. Teng. Decision trees are PAC-learnable from most product distributions: a smoothed analysis. *arXiv preprint arXiv:0812.0933*, 2008.

[19] V. Klee and G. J. Minty. How good is the simplex algorithm? *Inequalities-III: Proceedings*, page 159, 1972.

[20] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[21] B. Manthey and R. Reischuk. Smoothed analysis of binary search trees. *Theoretical Computer Science*, 378(3):292–315, 2007.

[22] B. Manthey and H. Röglin. Improved smoothed analysis of the k-means method. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 461–470. Society for Industrial and Applied Mathematics, 2009.

[23] M. Minsky and S. Papert. Perceptron: an introduction to computational geometry. *The MIT Press, Cambridge, expanded edition*, 19:88, 1969.

[24] A. B. Novikoff. On convergence proofs for perceptrons. Technical report, DTIC Document, 1963.

[25] H. Röglin. Minicourse on smoothed analysis. *Lecture Notes*, 2007.

[26] H. Röglin and B. Vöcking. Smoothed analysis of integer programming. *Mathematical Programming*, 110(1):21–56, 2007.

[27] R. Rojas. *Neural Networks: A Systematic Introduction.* Springer, 1996.

[28] F. Rosenblatt. The perceptron, a perceiving and recognizing automaton. Technical report, Report 85-460-1, Cornell Aeronautical Laboratory, 1957.

[29] R. Schneider. *Convex Bodies: The Brunn–Minkowski Theory*, volume 151. Cambridge University Press, 2013.

[30] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 296–305. ACM, 2001.

[31] D. A. Spielman and S.-H. Teng. Smoothed analysis: motivation and discrete models. In *Algorithms and Data Structures*, pages 256–270. Springer, 2003.

[32] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms and heuristics: progress and open questions. 2006.

[33] D. A. Spielman and S.-H. Teng. Smoothed analysis: an attempt to explain the behavior of algorithms in practice. *Communications of the ACM*, 52(10):76–84, 2009.

A Corrections

This appendix contains the details of the corrections made to apparent errors in [3] and [10].

A.1 Correction to Theorem 4.14 in k-Means Proof

Theorem 4.14 corresponds to Proposition 5.4 in [3] (or Proposition 5.7 in the journal version). In the original paper, the proposition states that the potential drops by at least $\frac{4\epsilon^2}{n}$. The proof gets to the point where it suffices to show $\|c - c'\| \geq \frac{2\epsilon}{n}$, then says: 'On the other hand, since $x \in \mathcal{C}$ when c was calculated, and since $|\mathcal{C}| \leq n$, the distance from c to $\mathcal{H}$ is at least ϵ/n.' ($\mathcal{H}$ is $\mathcal{H}_2$ in our notation). This statement does not appear to follow: all we know is that c is the centre of mass of S_1, which contains x, which is at least ϵ from $\mathcal{H}_2$ on the opposite side from c. We know nothing about S_1 in relation to $\mathcal{H}_2$.

The key to the correction is considering the *preceding* Voronoi boundary $\mathcal{H}_1$ to $\mathcal{H}_2$. This results in changing the statement of the theorem from $\frac{4\epsilon^2}{n}$ to $\frac{2\epsilon^2}{n}$. This factor of two makes no difference to the proof of the main theorem.

A.2 Correction to Lemma 5.10 in Perceptron Proof

Lemma 5.10 corresponds to Lemma 6.1 in [10]. In the original paper, the lemma gives a bound of $O\left(\frac{\epsilon\sqrt{d}}{\sigma}\log\frac{\sigma}{\epsilon\sqrt{d}}\right)$. The proof gets to the point where we have

$$O\left(\frac{\epsilon\sqrt{d}}{\sigma}\log\frac{\sigma}{\epsilon\sqrt{d}}\right) + \left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4} \tag{32}$$

and concludes that this is $O\left(\frac{\epsilon\sqrt{d}}{\sigma}\log\frac{\sigma}{\epsilon\sqrt{d}}\right)$. However, we are considering arbitrarily small ϵ, so in fact we have $\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4} \geq \left(\frac{\epsilon\sqrt{d}}{\sigma}\right)$, so the bound should be $O\left(\left(\frac{\epsilon\sqrt{d}}{\sigma}\right)^{1/4}\log\frac{\sigma}{\epsilon\sqrt{d}}\right)$. The power of 1/4 carries through the statements of Lemma 5.11 and Theorem 5.12, causing us to choose a slightly different value of ν in the proof of Theorem 5.12, making our final bound $O\left(\frac{d^3m^8\log^8(m/\delta)}{\sigma^2\delta^8}\right)$ instead of $O\left(\frac{d^3m^2\log^2(m/\delta)}{\sigma^2\delta^2}\right)$, but this is still polynomial in d, m, $1/\sigma$ and $1/\delta$. Note: any other differences in the proof of Lemma 5.10 (for example we choose $\kappa = \log(\sigma/\epsilon\sqrt{d})$ and the original chooses $\kappa^2 = \log(\sigma/\epsilon\sqrt{d})$) are due to our presentation for the perceptron problem and the original paper's presentation for linear programming.