
Graph-Based Approximate Nearest Neighbor Search Revisited: Theoretical Analysis and Optimization

Xinran Ma

New York University
maxinran22@mails.ucas.ac.cn

Zhaoqi Zhou

Huawei Technologies Co., Ltd.
zhouzhaoqi1@huawei.com

Chuan Zhou

Academy of Mathematics
and Systems Science, CAS
zhouchuan@amss.ac.cn

Qi Meng

Academy of Mathematics
and Systems Science, CAS
meq@amss.ac.cn

Zaijiu Shang

Shanghai Institute for Mathematics
and Interdisciplinary Sciences
zaijiu@simis.cn

Guoliang Li

Tsinghua University
liguoliang@tsinghua.edu.cn

Zhiming Ma

Academy of Mathematics
and Systems Science, CAS
mazm@amt.ac.cn

Abstract

Graph-based approaches to approximate nearest neighbor search (ANNS) have achieved remarkable success in enabling fast, high-recall retrieval on billion-scale vector datasets. Among them, the Sparse Neighborhood Graph (SNG) has emerged as a widely adopted graph structure due to its superior search performance. However, the theoretical understanding of SNG remains limited, leading to reliance on heuristic-based and often suboptimal truncation strategies. In this work, we aim to bridge the gap between theory and practice by providing formal guarantees for graph-based ANNS methods and proposing principled optimization strategies for the truncation parameter. By characterizing the index construction process through martingale-based analysis, we show that the degree of the index graph is $O(n^{2/3+\epsilon})$, where ϵ is an arbitrarily small constant. Furthermore, we prove that the expected search path length during query processing is $O(\log n)$. Based on these theoretical insights, we introduce a novel and principled method for selecting the truncation parameter R in SNG. Experimental results demonstrate that our method achieves comparable or superior performance in terms of query latency and Recall@10 compared to commonly used binary search heuristics, while yielding 2x to 9x speedups in overall index construction.

1 Introduction

NNS-ANNS Problems. Nearest neighbor search (NNS) is a fundamental problem in computer science [16, 25] and has become a fundamental research topic across various fields such as information retrieval [28, 27], machine learning [36, 22] and data mining [31]. Formally, given a dataset $P = \{p_1, \dots, p_n\} \subset \mathbb{R}^d$ of cardinality n and query point $q \in \mathbb{R}^d$, the NNS problem requires finding the k closest points to q under an appropriate distance metric $\|\cdot\|$. The computational complexity of exact NNS grows prohibitively with both dimensionality d due to the well-documented curse of dimensionality [17] and the expanding dataset size n . This fundamental limitation has motivated the development of Approximate Nearest Neighbor Search (ANNS) algorithms, which achieve significant efficiency improvements while sacrificing minimal accuracy.

Graph-Based ANNS. Modern ANNS approaches achieve remarkable efficiency-accuracy trade-offs and can be categorized into four categories: tree-based approaches [32, 5], hashing-based approaches [8, 34, 21, 6], vector quantization-based approaches [10, 14], and graph-based approaches [3, 33, 37, 11]. Among these, graph-based approaches have demonstrated particular success in practical applications [7, 37], owing to their ability to capture local neighborhood structures. The graph-based ANNS framework operates through two sequential phases: an *index construction phase* that builds a graph preserving neighborhood relationships, followed by an *query processing phase* that employs the *GreedySearch* algorithm [12, 30, 24] to iteratively traverse the graph along paths until convergence to the query point. Numerous graph construction strategies have been proposed, each differing in how neighbors are selected and edges are formed.

Theoretical Limitations. Despite their empirical success, graph-based ANNS methods remain theoretically underexplored, primarily due to the stochastic nature of high-dimensional data and the combinatorial complexity of graph construction. Existing theoretical studies often rely on simplified assumptions and focus on analyzing key structural properties such as degree, query complexity, and search path length. For instance, sublinear query complexity has been established for threshold-based nearest neighbor graphs¹ under spherical data models [26], while navigable graphs² constructed under a binomial distribution exhibit an average degree of $O(n^\beta)$ for $\beta < 1/2$ [9], although such assumptions are often unrealistic in practical settings. The Monotonic Search Network (MSNET)³ is known to achieve a logarithmic search path length [12]. The Sparse Neighborhood Graph (SNG) [1] is a variant designed to further reduce graph density while preserving search quality in practice. It has been broadly adopted in modern ANNS systems, including DiskANN [30], HNSW [24], and NSG [12]. The worst-case search analysis of SNG [18] relies on dataset aspect ratios⁴, yielding bounds that are less interpretable compared to those based on cardinality.

Practical Limitations and Parameter Tuning. To reduce index construction time, a common practice in SNG construction is to impose a maximum degree of node through a *truncation parameter* R . However, existing methods typically rely on parameter sweeping [30], most commonly implemented as a binary search guided by the downstream search performance of the constructed graph. This tuning process for R is computationally expensive, highlighting the need for a theoretically grounded approach to its selection—one that ensures high search performance while reducing tuning overhead.

1.1 Main Results

To address the dual challenges of theoretical understanding and practical parameter tuning in SNG-based approximate nearest neighbor search, this study develops a rigorous theoretical framework for graph construction under realistic random data assumptions, with a focus on the SNG. Building on this foundation, we further derive a marginal optimal principle-based strategy for parameter optimization. Specifically, we provide tighter bounds for both the indexing and query processes and offer an analytical expression for the key graph truncation parameter in SNG, enabling direct performance gains during index construction.

Degree Bound of the SNG Graph. We establish a tighter degree bound for SNG, improving upon the prior rough bound of $O(n)$ from [30]:

Theorem 1. *Given a dataset P of n points in d -dimensional space, with probability 1, the maximum out-degree in the constructed SNG is bounded by $O(n^{2/3+\epsilon})$ for any small constant $\epsilon > 0$.*

This result highlights the sparsity of the graph and the effectiveness of pruning during index construction. It implies that the SNG remains sparser than a complete graph. The proof idea and sketch are provided in Section 3, and the full proof is deferred to Appendix C.4.

Search Path Length on SNG. Building on the degree bound, we further demonstrate that the expected search path length in SNG is logarithmic in the dataset size:

¹A threshold-based nearest neighbor graph connects two nodes if the distance between them is below a specified threshold value.

²A graph is navigable if the search algorithm can find a path from any starting vertex to any target vertex in the dataset.

³The Monotonic Search Network is a navigable graph where the path between any two nodes follows a monotonic trajectory.

⁴The aspect ratio is the ratio of the maximum to minimum distance between any pair of points in the dataset.

Theorem 2. *Given a dataset P with n points in d -dimensional space, with probability 1, the SNG search converges in $O(\log n)$ steps.*

The sketch of the proof is provided in Section 4, and the full argument appears in Appendix C.5.

Optimizing Parameter R . Based on these findings, we analyze the graph construction complexity under the truncated setting and derive an analytical expression for the optimal truncation parameter R , as presented in Section 5.2. This theoretically grounded formulation enables principled and efficient parameter selection, removing the need for manual tuning. Empirically, in the search phase, our optimized R achieves comparable or improved latency and Recall@10 compared to traditional binary search-based heuristics. In the index construction phase, our optimized approach delivers 2x to 9x acceleration across three standard benchmark datasets.

2 Preliminaries

2.1 Overview of SNG Construction Algorithm

The construction process of SNG proceeds as follows. Given a dataset $P = \{p_1, \dots, p_n\} \subset \mathbb{R}^d$, for each point $p \in P$, a candidate set $S = P \setminus \{p\}$ is initialized. While S is non-empty, the algorithm selects the nearest neighbor $p^* = \arg \min_{s \in S} \|p - s\|$, adds a directed edge from p to p^* , and prunes the candidate set by removing all points $p' \in S$ that satisfy the condition $\|p - p'\| > \|p^* - p'\|$. In particular, under the truncated setting, the iterative process terminates after at most R iterations.

To prevent pathological cases, such as points aligned on a straight line that could result in a graph diameter of $O(n)$, Vamana algorithm [30] introduces a relaxation parameter α , modifying the pruning condition to $\|p - p'\| > \alpha \cdot \|p^* - p'\|$ ⁵. In practice, α is often set to a fixed value slightly greater than 1 and does not require tuning [19].

2.2 Probabilistic Framework

Stochastic Modeling of the Pruning Process. To analyze the SNG construction, we establish the following probabilistic framework. Assume that p is the point being processed. Let $t \in \{0, 1, 2, \dots\}$ index the iterations, and at iteration t , the *candidate set* is denoted S_t , initialized as $S_0 = P \setminus \{p\}$. The corresponding *processed points set* is defined as $S'_t = P \setminus (S_t \cup \{p\})$, i.e., S'_t contains all pruned points and previously selected nearest neighbors. In the non-truncated case, the process terminates when $S_t = \emptyset$, or equivalently, when $S'_t = P \setminus \{p\}$. Due to the randomness of point distributions in high-dimensional space, the number of pruned points at each iteration is also stochastic, resulting in randomness in the graph structure. Consequently, the pruning process is inherently random. This correspondence between iteration counts and the evolution of S_t and S'_t allows us to treat them as two *stochastic processes*. We formalize this with the following probabilistic setup. For definitions of foundational probability theory concepts, we refer readers to Appendix D.

Probability Space for SNG construction. We introduce the *probability space* $(\Omega, \mathcal{F}, \mathbb{P})$, where Ω denotes the sample space containing all possible sequences of *nearest neighbor selections* and *pruning steps* during **overall** SNG construction. The set \mathcal{F} is the corresponding σ -algebra, capturing all measurable events in Ω , and \mathbb{P} is the associated probability measure. We further define the *natural filtration* $\{\mathcal{F}_t\}_{t \geq 1}$, representing the information available up to iteration t . That is, \mathcal{F}_t is generated by the sequence of nearest neighbor selections and candidate sets:

$$\mathcal{F}_t = \sigma((p_1^*, S_1), (p_2^*, S_2), \dots, (p_t, S_t))$$

where $p_i^* = \arg \min_{s \in S_i} \|p - s\|$ is the nearest neighbor selected at iteration i . The update rule for candidate set is given by: $S_{t+1} = S_t \setminus (\{p' \in S_t : \|p - p'\| > \alpha \cdot \|p^* - p'\|\} \cup \{p_{t+1}^*\})$. This formulation models the pruning process as a sequence of progressively revealed decisions. In each iteration, the state of the system is updated based on local geometric information, following a stochastic rule defined by the data distribution and the pruning condition.

⁵In SNG constructions used in HNSW[24] and NSG[12] algorithm, the α parameter is not included and can be interpreted as being implicitly set to 1. The results of this paper also hold when incorporating parameter α , for a comprehensive discussion, please refer to the end of Appendix C.4.

We now show that the construction process exhibits a consistent dynamic trend. Intuitively, in each iteration, the number of processed points strictly increases, while the size of the candidate set strictly decreases. This monotonic behavior can be formally captured through the martingale framework.

Definition 3 (Martingale). *A stochastic process $\{X_t\}_{t \geq 0}$ is a martingale with respect to the natural filtration $\{\mathcal{F}_t\}$ if, for all $t \geq 0$:*

$$\mathbb{E}[|X_t|] < \infty \quad \text{and} \quad \mathbb{E}[X_{t+1}|\mathcal{F}_t] = X_t.$$

It is a supermartingale if $\mathbb{E}[X_{t+1}|\mathcal{F}_t] \leq X_t$, and a submartingale if $\mathbb{E}[X_{t+1}|\mathcal{F}_t] \geq X_t$.

In the context of SNG construction, the size of candidate set $|S_t|$ forms a supermartingale, while the size of the processed set $|S'_t|$ forms a submartingale, both with respect to the natural filtration $\{\mathcal{F}_t\}$. This is consistent with the fact that at each iteration, the candidate set S_t can only decrease in expectation in each step, whereas the processed set S'_t increases monotonically as the algorithm processes. From the SNG pruning rule and the probability framework, we summarize the following properties of processed sets S'_t , which will facilitate the analysis of the construction process. These properties directly follow from our previous results and do not require separate proof.

Lemma 4 (Dynamic Property of S'_t). *Given a dataset $P \subset \mathbb{R}^d$ consisting of n points, the SNG construction of $p \in P$ satisfies the following properties:*

- (1) **Monotonicity:** *The processed sets form a nested sequence $\emptyset = S'_0 \subset S'_1 \subset S'_2 \subset \dots \subset S'_T$, with a submartingale $|S'_t|$.*
- (2) **Termination:** *The final processed set includes $n - 1$ points, excluding only the current point p .*
- (3) **Degree property:** *Out-degree of p equals the number of iterations, as each iteration adds exactly one directed edge from p to its nearest neighbor in the current candidate set.*

Definition 5 (M - t Level). *The SNG construction is said to reach the M - t level at iteration t if t is the smallest iteration that $|S'_t| \geq M$. This represents the first passage time at which at least M points have been processed.*

The M - t metric serves as a tool to track the progress of graph construction. For a fixed iteration count t , a larger value of M indicates faster pruning and more rapid expansion of the processed set. For a fixed M , a smaller value of t implies that the algorithm has processed M points more quickly.

2.3 Distribution Assumption

For our theoretical analysis, we assume that large-scale, high-dimensional data points are independently and uniformly distributed. This assumption is also adopted in [12], where it serves as the basis for the theoretical analysis of MSNET. Although real-world datasets may exhibit non-uniformity, our core probabilistic framework remains valid and offers potential for broader generalization. A more detailed discussion of this assumption is provided in Appendix G.

3 Degree Bounds of SNG

We focus on bounding the degree of SNG. As shown in Lemma 4, the out-degree of each point equals the number of pruning iterations during construction. Thus, our analysis shifts to examining the dynamics of the pruning process and estimating the total number of iterations required and the total number of pruning iterations required during index construction.

First, we aim to demonstrate the existence of *an initial fast pruning phase* that significantly reduces the candidate set within a few iterations. To this end, we begin by characterizing the pruning probability at each step of the SNG construction in the following Lemma 6. This is a key component in quantifying how many points are removed from the candidate set in expectation. The proof is deferred to Appendix C.1. In particular, Appendix C.2 provides a detailed discussion of the applicability of Lemma 6.

Lemma 6 (Pruning Probability in SNG Construction). *Given a dataset $P \subset \mathbb{R}^d$ consisting of n points independently and uniformly distributed within a ball of radius ρ_0 , centered at p , we consider the SNG construction processed on the center point p with $\alpha = 1$. The probability π_t that a point p' is pruned in the t -th iteration is given by $\pi_t = \frac{I_{1-(\rho_t/\rho_0)^2}(\frac{d+1}{2}, \frac{1}{2}) - (\rho_t/\rho_0)^d \cdot I_{0.75}(\frac{d+1}{2}, \frac{1}{2})}{2(1-(\rho_t/\rho_0)^d)}$, where $I_x(a, b)$ denotes the regularized incomplete Beta function, and ρ_t is the distance between the nearest neighbor and p in t -th iteration. For large d , this probability satisfies $\frac{I_{0.75}(\frac{d+1}{2}, \frac{1}{2})}{2} < \pi_t < \frac{1}{2}$.*

To motivate the study of the fast pruning behavior, we examine a low-dimensional setting using the M - t level. As a concrete example, in two dimensions, we show that **only four iterations** are sufficient to prune over 80% of the candidate points.

Lemma 7 (Two-Dimensional Case). *Given a dataset $P \subset \mathbb{R}^2$ with n points independently and uniformly distributed within a ball of radius ρ_0 . Then, with probability $1 - o(1)$, the SNG construction process reaches $(0.8(n-1))$ -4 level.*

Proof. Recall that in the SNG construction of $p \in P$, we iteratively prune a candidate set S_i , starting with $S_0 = P \setminus p$. At iteration $i = 1, 2, \dots$, nearest neighbor is selected from S_{i-1} , and prune other points with probability π_i . Specifically, the number of points pruned in iteration i , denoted by ΔS_i , follows a binomial distribution $\Delta S_i \sim \text{Bin}(|S_{i-1}| - 1, \pi_i)$, where the subtraction accounts for the exclusion of the selected nearest neighbor from the pruning. After t iterations, S'_t includes all pruned points and selected nearest neighbors. Thus, we have $|S'_t| = \sum_{i=1}^t (\Delta S_i + 1)$ for $t = 2, 3, \dots$, and for $t = 1$, $|S'_1| = \Delta S_1 + 1$, since each iteration contributes one nearest neighbor and ΔS_i pruned points.

From Lemma 6, we know that for $d = 2$, the pruning probability at each iteration satisfies $\pi_i > 1/3$ (see the end of Appendix C.1 for details). To estimate the pruning progress, we assume a minimal pruning rate of $\pi_{\min} = \frac{1}{3}$ per iteration. Expected number of points pruned in iteration i is $E[\Delta S_i] = \pi_i(|S_{i-1}| - 1)$, and the variance is $\text{Var}(\Delta S_i) = \pi_i(1 - \pi_i)(|S_{i-1}| - 1)$. Since $\pi_i(1 - \pi_i) < \frac{1}{3}$, we bound the normalized variance $\text{Var}(\frac{\Delta S_i}{n-1}) = \pi_i(1 - \pi_i)(|S_{i-1}| - 1)/(n-1)^2 \leq \frac{1}{3(n-1)}$.

If each iteration prunes at least a fraction π_{\min}^i , where $\pi_{\min}^1 = 1/3$, $\pi_{\min}^2 = 2/9$, $\pi_{\min}^3 = 4/27$, and $\pi_{\min}^4 = 8/81$, the cumulative expected fraction pruned after four iterations is 0.802.

To ensure this holds with high probability, we control the deviation of ΔS_i from its expectation by selecting constants ϵ_i , such that:

$$\epsilon_1 < \frac{E[\Delta S_1]}{n-1} - \frac{1}{3}, \quad \epsilon_2 < \frac{E[\Delta S_2]}{n-1} - \frac{2}{9}, \quad \epsilon_3 < \frac{E[\Delta S_3]}{n-1} - \frac{4}{27}, \quad \epsilon_4 < \frac{E[\Delta S_4]}{n-1} - \frac{8}{81}.$$

By Chebyshev's inequality, we obtain the following concentration bound for each iteration:

$$\Pr\left(\left|\frac{\Delta S_i}{n-1} - \frac{p_i(|S_{i-1}| - 1)}{n-1}\right| \leq \epsilon_i\right) \geq 1 - \frac{\text{Var}\left(\frac{\Delta S_i}{n-1}\right)}{\epsilon_i^2} \geq 1 - \frac{1}{3(n-1)^2 \epsilon_i^2}.$$

Assuming the deviations remain within ϵ_i for all four iterations, the overall probability that cumulative number of processed points satisfies $|S'_4| \geq 0.8(n-1)$ is at least

$$\Pr(|S'_4| \geq 0.8(n-1)) \geq \prod_{i=1}^4 \left(1 - \frac{1}{3(n-1)^2 \epsilon_i^2}\right) = 1 - o(1).$$

Therefore, with probability $1 - o(1)$, the SNG construction reaches the $(0.8(n-1))$ -4 level. \square

In high-dimensional settings, the same intuition is reflected in the concept of *sublinear-time progress*: a linear number of points is processed within sublinear iterations. This is formalized below.

Lemma 8 (Sublinear Time Progress). *Given a dataset $P \subset \mathbb{R}^d$ with n points distributed uniformly:*
(1) *For any constants $\nu_1, \nu_2 \in (0, 1)$ with $\nu_1 > \nu_2$, with probability 1, the SNG construction reaches the $n^{\nu_1} \cdot O(n^{\nu_2})$ level.*
(2) *For any constant $\nu \in (0, 1)$, with probability 1, the SNG construction reaches the $(n - n^{1-\nu}) \cdot O(n^\nu)$ level.*

Lemma 8 is proven in Appendix C.3, and will be critical in bounding the total number of pruning iterations in Theorem 1. This naturally raises the question of whether the pruning process continues to progress rapidly beyond the initial phase. Empirical results presented in Appendix H.2 demonstrate that, in practice, the pruning process undergoes a *significant slowdown* after the initial fast pruning phase, often reaching a *plateau phase*.

To analyze the plateau phase, we observe that the submartingale S'_t exhibits bounded per-step variance throughout the pruning process. Leveraging this property, we introduce the Differential Equation Method (DEM) (see Lemma 11), a standard tool for approximating the evolution of bounded-variance

submartingales. By combining the analyses of both the initial fast pruning phase and the plateau phase, we obtain a comprehensive characterization of the pruning dynamics.

We now revisit our main result, Theorem 1, and provide a sketch of the proof. The full proof is provided in Appendix C.4.

Theorem 1. *Given a dataset P of n points in d -dimensional space, with probability 1, the maximum out-degree in the constructed SNG is bounded by $O(n^{2/3+\epsilon})$ for any small constant $\epsilon > 0$.*

Sketch Proof. As outlined at the beginning of this section, our goal is to show that the total number of iterations T , which corresponds to the degree, satisfies $T = O(n^{2/3+\epsilon})$. We divide the SNG construction into two parts, based on the number of processed points S'_t : (1) When $|S'_t| < n - n^{1-\nu}$, for some $\nu \in (2/3, 1)$, Lemma 8 ensures that the number of iterations to reach this stage is $t_1 = O(n^\nu)$ with probability 1. (2) When $|S'_t| \geq n - n^{1-\nu}$, the process is within the plateau phase. Applying DEM, the discrete process S'_t can be approximated by the solution to an ODE. Solving this ODE, the number of iterations required to reach $n - 1$ processed points in this phase is $O(n^{1-\nu})$. Adding both phases gives $T = O(n^\nu) + O(n^{1-\nu}) = O(n^\nu)$, yielding the bound $O(n^{2/3+\epsilon})$ for any small $\epsilon > 0$. \square

4 Search Bounds

We now turn to the analysis of search on the constructed SNG. Our goal is to show that with high probability, GreedySearch converges to an approximate nearest neighbor in $O(\log n)$ steps. The full proof of Theorem 2 is provided in Appendix C.5.

Theorem 2. *Given a dataset P with n points in d -dimensional space, with high probability 1, the SNG search converges in $O(\log n)$ steps.*

Sketch Proof. Let $\{v_0, v_1, \dots, v_k\}$ denote a greedy search path from the start node $p = v_0$ to the approximate nearest neighbor of the query q . At each step, GreedySearch moves to the neighbor closest to q , ensuring that $\|v_i - q\| > \|v_{i+1} - q\|$.

To analyze the expected path length k , we analyze the distribution of neighbors around the destination v_k , assuming they are uniformly distributed in a d -dimensional ball of radius ρ_0 . We partition this ball into concentric annular layers and let η_i denote the number of neighbors lying in the i -th layer. Each η_i follows a binomial distribution proportional to the shell volume, and the total number of neighbors is bounded by $\eta = O(n^{2/3+\epsilon})$ from Theorem 1.

Using volume arguments, we upper bound the expected total number of neighbors located within the ball of radius $\|v_k - v_0\|$ by the expected number of dataset points that fall into the same region. Specifically, we show $E[k \cdot \eta] \leq \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1} \cdot E[(\frac{(\rho_0 - \Delta r)}{\rho_0})^{d(k-1)}]}{\Delta r^{d+1} \cdot d}$, where Δr is a lower bound on per-step radial improvement. Applying Jensen's inequality, we define an auxiliary function $g(x)$ whose root bounds $E[k]$. We demonstrate $g(k) < 0$, note that $g(x)$ is monotonically increasing, and $g(\log n) > 0$, which together imply that $k < \log n$, hence $E[k] = O(\log n)$. \square

5 Truncation Parameter Optimization

In the preceding sections, we establish theoretical guarantees for the SNG structure. A practical and widely adopted realization of SNG is implemented in the Vamana algorithm, which serves as the core indexing component in DiskANN system [30]. In the following, we focus on optimizing the truncation parameter specifically for Vamana algorithm (see Appendix E for algorithm pseudo code).

5.1 Complexity Analysis

Total Complexity of Vamana. The following are the graph construction steps of the Vamana algorithm. It begins by constructing an initial randomized R -regular graph with truncation parameter R . For each data point p , GreedySearch is performed from the dataset centroid to gather a candidate set for subsequent pruning. Following candidate collection, the graph is refined through iterative SNG pruning, truncation limits the number of pruning iterations to R . Additionally, reverse edges are

added to neighbors of each point to improve search connectivity, which may cause degree overruns. Such overruns trigger further pruning. Detailed complexity analyses for GreedySearch and pruning are deferred to Appendix F, the *overall construction complexity* of the Vamana algorithm is expressed as:

$$n(C_1 \cdot R \log n + b_1 + C_2(R \cdot R \log n/\alpha) + b_2 + C_3(\alpha \cdot R \cdot R^2) + b_3).$$

where C_i, b_i are implementation-dependent constants.

5.2 Optimization Principle

As discussed in Section 2.1, the pruning parameter α is typically predefined. In contrast, the truncation parameter R introduces an intrinsic trade-off: increasing R improves search accuracy but also increases construction complexity. The optimal balance between these objectives varies across applications, depending on the tolerance of each application. This reflects the absence of a universal standard trade-off.

Interestingly, our empirical observations reveal that the construction complexity exhibits a non-monotonic relationship with respect to α : it initially decreases and then increases. This suggests the existence of an optimal α that minimizes construction cost for a given R . Motivated by this, we propose computing the value of R that achieves optimal efficiency with respect to a target α , leading to improvements beyond the naive trade-off. We refer to this as the *marginal optimality principle*.

Algorithm 1 Optimization of Truncation Parameter R

- 1: **Input:** Test pruning parameter α_1 , target pruning parameter α_2 , number of data points n
 - 2: **Output:** Optimized truncation parameter R^*
 - 3: Initialize $R \leftarrow n^{2/3}$
 - 4: Build graph index using Vamana with RobustPrune(α_1, R)
 - 5: Compute average degree \bar{R} from the resulting graph
 - 6: Compute scaling constant $K' \leftarrow \frac{\alpha_1^2 \cdot \bar{R}}{\log n}$
 - 7: Compute optimized truncation $R^* \leftarrow \frac{K' \cdot \log n}{\alpha_2^2}$
 - 8: **Return** R^*
-

Our theoretical analysis, by setting the derivative of the construction complexity with respect to α to zero, yields the relationship $R^* \propto \log n / \alpha^2$. This approach is inspired by the complexity-aware tuning strategies used in prior work, such as [13]. The goal of our algorithm is to determine the constant of proportionality. To this end, we begin with an initial construction using $R = n^{2/3}$ under a test pruning parameter α_1 . This choice corresponds to a known theoretical upper bound and approximates the structure of a non-truncated graph. We then measure the average degree \bar{R} of the constructed graph to estimate the scaling coefficient K' in the formula of R^* . Having obtained this coefficient, we directly compute the optimal truncation parameter R^* for the target pruning parameter α_2 .

6 Experiment

Experiment Setup. We evaluate our parameter optimization method against the traditional binary search baseline through two sets of experiments: (1) End-to-end parameter selection and graph construction, where we measure the total index construction time; and (2) Query-time performance evaluation, where we assess the resulting indices using both latency and Recall@10 metrics⁶. All the experiments are conducted on a machine with an Intel i7-14700KF CPU (20 cores), 64GB DDR4 RAM, and a Samsung 990 EVO Plus 2TB SSD. Codes are available at <https://anonymous.4open.science/r/Graph-based-ANNS-F168>.

⁶Recall@10 measures the proportion of queries for which at least one true nearest neighbor appears among the top-10 retrieved candidates. Formally, $\text{Recall}@k = \frac{|S \cap T|}{|T|}$, where S is the retrieved set and T is the ground truth.

We use Vamana [30] as the baseline—a generalization of Sparse Neighborhood Graphs with a tunable pruning parameter α —due to its strong empirical performance on large-scale datasets. All experiments are conducted on three standard public benchmarks: SIFT1M, GIST1M [20], and DEEP1M [4]. Dataset statistics are summarized in Table 2.

To further assess the distributional robustness of our theoretical results, we conduct an experiment on a synthetic dataset generated from a Gaussian Mixture Model (GMM). GMMs are known to approximate a broad class of real-world distributions effectively [15]. We create a dataset with 100,000 points in 8 dimensions, construct an SNG on it, and analyze the resulting degree distribution. The goal is to empirically verify whether the sparsity property and degree bound established under uniform distributions (Theorem 1) is scalable for non-uniform distributions.

6.1 Index Construction Time Overhead

In the truncated version of the SNG, graph construction consists of two phases: parameter selection and index building. We compare our analytical parameter optimization method with the traditional binary search approach by separately measuring the parameter selection time (denoted as tuning time) and the graph construction time using the Vamana algorithm (denoted as construction time) across various values of the pruning parameter α . As shown in Figure 1, our method consistently accelerates the overall construction pipeline. Specifically, it reduces the total construction time by over 47% in all tested settings, with the majority of the gain attributed to the significant reduction in tuning time. In particular, the tuning time is reduced by more than 50%, demonstrating the efficiency of our theoretical optimization approach.

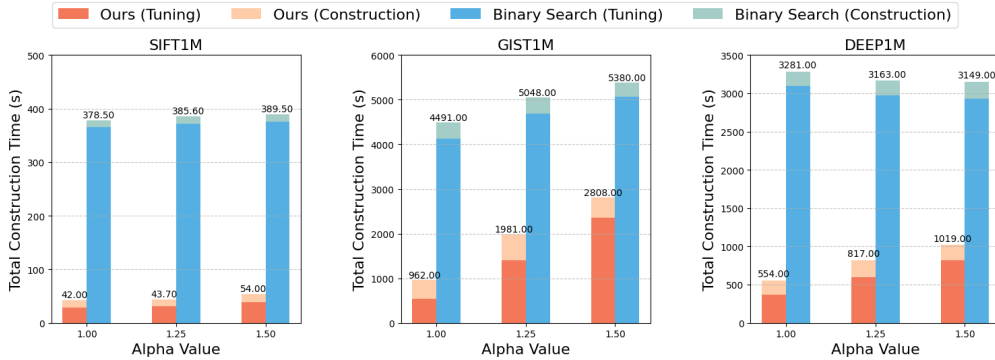


Figure 1: Comparison of total graph construction time

6.2 Search Performance

We compare the optimized truncation parameter R , derived from our analytical method, against the baseline values obtained via binary search across three benchmark datasets: SIFT1M, GIST1M, and DEEP1M. Evaluation is conducted using recall and query latency. The results are presented in Figures 2, 3, and 4, respectively. Across all datasets, our method consistently achieves higher recall at comparable latency levels. For example, on GIST1M, our method maintains similar recall under $\alpha = 1.00$ and $\alpha = 1.25$, and achieves a 1.52% absolute improvement in recall at 228 ms latency under $\alpha = 1.50$. On DEEP1M, under $\alpha = 1.00$, our approach improves recall by at least 1.9% at just 5 ms latency. These results demonstrate that our parameter optimization method matches or outperforms binary search-based heuristics across datasets with varying characteristics, while being significantly more efficient in tuning.

6.3 Scalability Evaluation

To evaluate the scalability of our theoretical results—particularly the sparsity and degree bounds established under uniform distributions in Section 3—we extend our analysis to non-uniform data settings. Specifically, we generate a synthetic dataset consisting of $n = 100,000$ points drawn from a Gaussian Mixture Model (GMM) with 8 dimensions and 10 clusters. We use 90% of the data (90,000

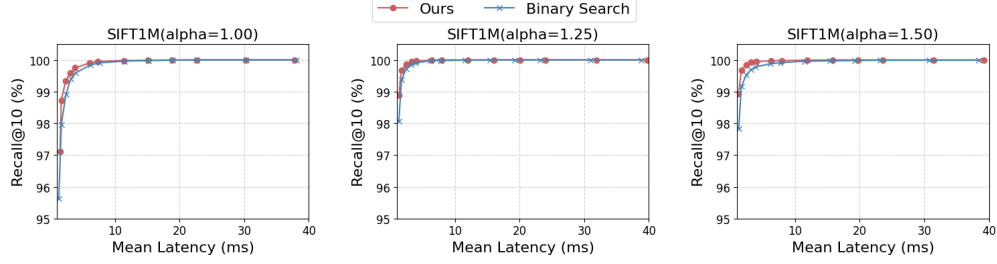


Figure 2: Query performance across varying α values on SIFT1M

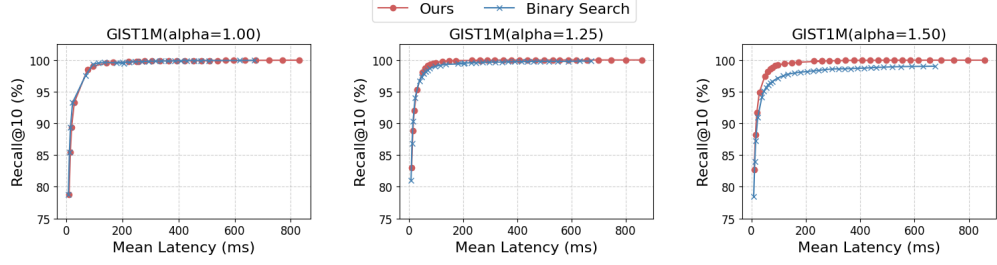


Figure 3: Query performance across varying α values on GIST1M

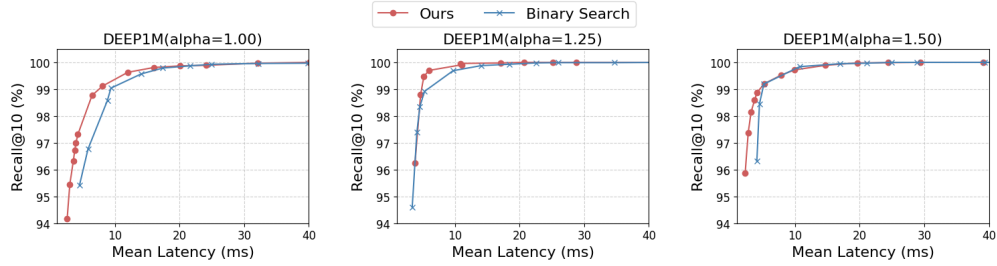


Figure 4: Query performance across varying α values on DEEP1M

points) as the base set to construct an SNG under parameter $\alpha = 1.2$. Results under other values of α are also provided in Appendix H.4.

The degree distribution of all nodes is visualized in Figure 5. Notably, approximately 20,000 points have degrees concentrated in the range $[50, 60]$, and the maximum observed degree does not exceed 100. This empirical evidence supports our theoretical prediction that the graph degree remains bounded by $O(n^{2/3})$, even under complex, non-uniform distributions. The corresponding query performance on this dataset is reported in Table 3.

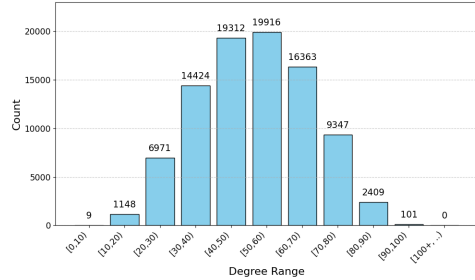


Figure 5: Degree distribution of GMM dataset

7 Conclusions

In this paper, we revisited the design and analysis of graph-based ANNS methods, with a focus on SNG). Our work bridges the gap between theoretical understanding and practical deployment. We provide rigorous theoretical guarantees on the structure of SNG by proving that the graph degree is bounded by $O(n^{2/3})$ and that the search path length is $O(\log n)$. Building on this theoretical foundation, we introduced a novel parameter optimization strategy based on the marginal optimality principle. Unlike existing heuristic methods such as binary search, our approach yields a principled estimate for the truncation parameter R , significantly reducing the tuning overhead while maintaining

or even improving search performance. Experimental results on a GMM dataset further validate the scalability of our theoretical analysis of degree.

References

- [1] Sunil Arya and David M. Mount. Approximate nearest neighbor queries in fixed dimensions. In *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 271–280, Philadelphia, PA, USA, 1993. SIAM.
- [2] ROBERT B. ASH. 2 - further results in measure and integration theory. In *Real Analysis and Probability*, Probability and Mathematical Statistics: A Series of Monographs and Textbooks, pages 66–68. Academic Press, 1972.
- [3] Ilias Azizi, Karima Echihabi, and Themis Palpanas. Elpis: Graph-based similarity search for scalable data science. *Proc. VLDB Endow.*, 16(6):1548–1559, 2023.
- [4] Artem Babenko and Victor S. Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2055–2063. IEEE Computer Society, 2016.
- [5] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.
- [6] Deng Cai. A revisit of hashing algorithms for approximate nearest neighbor search. *IEEE Trans. Knowl. Data Eng.*, 33(6):2337–2348, 2021.
- [7] Meng Chen, Kai Zhang, Zhenying He, Yinan Jing, and X. Sean Wang. Roargraph: A projected bipartite graph for efficient cross-modal approximate nearest neighbor search. *Proc. VLDB Endow.*, 17(11):2735–2749, 2024.
- [8] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the 20th ACM Symposium on Computational Geometry, Brooklyn, New York, USA, June 8-11, 2004*, pages 253–262. ACM, 2004.
- [9] Haya Diwan, Jinrui Gou, Cameron Musco, Christopher Musco, and Torsten Suel. Navigable graphs for high-dimensional nearest neighbor search: Constructions and limits. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.
- [10] Jingya Fan, Yang Wang, Wenwen Song, and Zhibin Pan. Flexible product quantization for fast approximate nearest neighbor search. *Multim. Tools Appl.*, 83(18):53243–53261, 2024.
- [11] Cong Fu, Changxu Wang, and Deng Cai. High dimensional similarity search with satellite system graph: Efficiency, scalability, and unindexed query compatibility. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(8):4139–4150, 2022.
- [12] Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proc. VLDB Endow.*, 12(5):461–474, 2019.
- [13] Yujian Fu, Cheng Chen, Xiaohui Chen, Weng-Fai Wong, and Bingsheng He. Optimizing the number of clusters for billion-scale quantization-based nearest neighbor search. *IEEE Trans. Knowl. Data Eng.*, 36(11):6786–6800, 2024.
- [14] Jianyang Gao and Cheng Long. Rabbitq: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search. *Proc. ACM Manag. Data*, 2(3):167, 2024.
- [15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- [16] Qiang Huang and Anthony Kum Hoe Tung. Lightweight-yet-efficient: Revitalizing ball-tree for point-to-hyperplane nearest neighbor search. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*, pages 436–449. IEEE, 2023.

- [17] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.
- [18] Piotr Indyk and Haike Xu. Worst-case performance of popular approximate nearest neighbor search implementations: Guarantees and limitations. *Advances in Neural Information Processing Systems*, 36:66239–66256, 2023.
- [19] Shikhar Jaiswal, Ravishankar Krishnaswamy, Ankit Garg, Harsha Vardhan Simhadri, and Sheshansh Agrawal. Ood-diskann: Efficient and scalable graph anns for out-of-distribution queries. *arXiv preprint arXiv:2211.12850*, 2022.
- [20] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.
- [21] Yifan Lei, Qiang Huang, Mohan S. Kankanhalli, and Anthony K. H. Tung. Locality-sensitive hashing scheme based on longest circular co-substring. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 2589–2599. ACM, 2020.
- [22] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [23] Shengqiao Li. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics & Statistics*, 4(1):66–70, 2010.
- [24] Yury A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(4):824–836, 2020.
- [25] Cun Matthew Mu, Jun Raymond Zhao, Guang Yang, Binwei Yang, and Zheng John Yan. Fast and exact nearest neighbor search in hamming space on full-text search engines. In *Similarity Search and Applications - 12th International Conference, SISAP 2019, Newark, NJ, USA, October 2-4, 2019, Proceedings*, volume 11807 of *Lecture Notes in Computer Science*, pages 49–56. Springer, 2019.
- [26] Liudmila Prokhorenkova and Aleksandr Shekhovtsov. Graph-based nearest neighbor search: From practice to theory. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7803–7813. PMLR, 2020.
- [27] M. M. Mahabubur Rahman and Jelena Tesic. Evaluating hybrid approximate nearest neighbor indexing and search (HANNIS) for high-dimensional image feature search. In *IEEE International Conference on Big Data, Big Data 2022, Osaka, Japan, December 17-20, 2022*, pages 6802–6804. IEEE, 2022.
- [28] Susav Shrestha, Narasimha Reddy, and Zongwang Li. ESPN: memory-efficient multi-vector information retrieval. In *Proceedings of the 2024 ACM SIGPLAN International Symposium on Memory Management, ISMM 2024, Copenhagen, Denmark, 25 June 2024*, pages 95–107. ACM, 2024.
- [29] Sunil Srinivasa and Martin Haenggi. Distance distributions in finite uniformly random networks: Theory and applications. *IEEE Trans. Veh. Technol.*, 59(2):940–949, 2010.
- [30] Suhas Jayaram Subramanya, Devvrit Fnu, Harsha Vardhan Simhadri, Ravishankar Krishnaswamy, and Rohan Kadekodi. Diskann: Fast accurate billion-point nearest neighbor search on a single node. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

- [31] Yukihiro Tagami. Annexml: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 455–464. ACM, 2017.
- [32] Yufei Tao, Ke Yi, Cheng Sheng, and Panos Kalnis. Quality and efficiency in high dimensional nearest neighbor search. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, pages 563–576. ACM, 2009.
- [33] Mengzhao Wang, Lingwei Lv, Xiaoliang Xu, Yuxiang Wang, Qiang Yue, and Jiongkang Ni. An efficient and robust framework for approximate nearest neighbor search with attribute constraint. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.
- [34] Jiuqi Wei, Botao Peng, Xiaodong Lee, and Themis Palpanas. DET-LSH: A locality-sensitive hashing scheme with dynamic encoding tree for approximate nearest neighbor search. *Proc. VLDB Endow.*, 17(9):2241–2254, 2024.
- [35] Nicholas Wormald. Differential equations for random processes and random graphs. *The annals of applied probability*, pages 1217–1235, 1995.
- [36] Qiang Yue, Xiaoliang Xu, Yuxiang Wang, Yikun Tao, and Xuliyuan Luo. Routing-guided learned product quantization for graph-based approximate nearest neighbor search. In *40th IEEE International Conference on Data Engineering, ICDE 2024, Utrecht, The Netherlands, May 13-16, 2024*, pages 4870–4883. IEEE, 2024.
- [37] Chaoji Zuo, Miao Qiao, Wenchao Zhou, Feifei Li, and Dong Deng. Serf: Segment graph for range-filtering approximate nearest neighbor search. *Proc. ACM Manag. Data*, 2(1):69:1–69:26, 2024.

A Supplementary Lemmas

We first introduce a commonly used lemma in probability theory, which determines the probability of the limsup of events by analyzing the convergence of the series.

Lemma 9 (Borel–Cantelli lemma). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, where: Ω is the set of all possible outcomes, \mathcal{F} is a sigma-algebra that contains all the information about events, and \mathbb{P} is the associated probability measure. Given a sequence of events $A_1, A_2, \dots \in \mathcal{F}$ such that $\sum_{n=1}^{\infty} \mathbb{P}(A_n) < \infty$, it follows that:*

$$\mathbb{P} \left(\bigcap_{N=1}^{\infty} \left(\bigcup_{n \geq N} A_n \right) \right) = 0.$$

Here, $\bigcap_{N=1}^{\infty} \left(\bigcup_{n \geq N} A_n \right)$ is also commonly denoted as $\limsup A_n$.

The following Chernoff bound provides a probabilistic guarantee on the deviation of a binomial random variable from its expected value.

Lemma 10 (Chernoff Bound of binomial variable). *Let $X \sim \text{Bin}(n, p)$ and let $\mu = \mathbb{E}[X]$. For any $0 < \delta < 1$:*

$$\mathbb{P}(X \leq (1 - \delta)\mu) \leq \exp \left(-\frac{\delta^2 \mu}{2} \right).$$

The frame of Wormald’s differential equation method (DEM) was developed by Wormald in 1990s [35] as a powerful tool for analyzing the discrete-time randomized graph processes and algorithms. Given a discrete-time stochastic process, in particular, consider a submartingale, if the one-step differences are bounded and the expected differences are well-approximated by a Lipschitz function, it follows that the process closely aligns with the corresponding differential equation with high probability.

Lemma 11 (Differential Equation method [35]). *Given integers $n \geq 1$, a bounded domain $\mathcal{D} \subseteq \mathbb{R}^{a+1}$, a function $F : \mathcal{D} \rightarrow \mathbb{R}$, and a sequence of sigma algebras $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \dots$. Define random variables $Y(i)$ such that $Y(i)$ is \mathcal{F}_i -measurable for $i \geq 0$.*

Assume that for all $i \geq 0$, when $(i/n, Y(i)/n) \in \mathcal{D}$, the following conditions are satisfied. Additionally, let \mathcal{D} contain the closure of the set $\{(0, z_0) : \mathbb{P}(Y(0) = z_0 n) \neq 0\}$ for some n , meaning that \mathcal{D} includes all possible starting points.

If $Y(t)$ is a submartingale and the following three conditions hold:

(1) (Boundedness) *For some $\beta = \beta(n) \geq 1$ and $\gamma = \gamma(n)$, the probability that $|Y(i+1) - Y(i)| \leq \beta$ given \mathcal{F}_i is at least $1 - \gamma$. This ensures that the change in $Y(i)$ over one step is controlled.*

(2) (Trend) *For some $\lambda_1 = \lambda_1(n) = o(1)$, the gap between the conditional expectation of the change and the normalized value of the function F is insignificant:*

$$\left| \mathbb{E}(Y(i+1) - Y(i) \mid \mathcal{F}_i) - F \left(\frac{i}{n}, \frac{Y(i)}{n} \right) \right| \leq \lambda_1.$$

(3) (Lipschitz) *F is continuous and Lipschitz with constant L on \mathcal{D} .*

Then we have:

(a) For $(0, \hat{z}_0) \in \mathcal{D}$, the differential equation

$$\frac{dz}{dx} = F(x, z)$$

has a unique solution in \mathcal{D} with the initial condition $z(0) = \hat{z}_0$. This solution can be extended close to the boundary of \mathcal{D} .

Let $\lambda > \lambda_1 + C_0 n \gamma$, where $\lambda = o(1)$. Then, with probability

$$1 - O \left(n \gamma + \frac{\beta}{\lambda} \exp \left(-\frac{n \lambda^3}{\beta^3} \right) \right),$$

the following holds:

$$Y(i) = nz \left(\frac{i}{n} \right) + O(\lambda n),$$

where $z(x)$ is the solution from part (a) with the initial condition $\hat{z}_0 = \frac{1}{n}Y(0)$. This result is valid uniformly for $0 \leq i \leq \sigma n$, where $\sigma = \sigma(n)$ is the supremum of x such that the solution $z(x)$ remains within \mathcal{D} .

We will also use Jensen's inequality in the proof. It establishes a fundamental relationship between the expectation of a convex transformation of a random variable and the transformation of its expectation.

Lemma 12 (Jensen's Inequality). *Suppose g is convex and X and $g(X)$ are both integrable. Then*

$$g(\mathbb{E}X) \leq \mathbb{E}g(X)$$

B Summary of Notations

The following is a summary of the main symbols used in the paper.

Table 1: Summary of Notation

Symbol	Description
$P = \{p_1, \dots, p_n\}$	Dataset of n points in \mathbb{R}^d
S_t	Candidate set at iteration t
S'_t	Processed set at iteration t : $S'_t = P \setminus (S_t \cup \{p\})$
p_t^*	Nearest neighbor of p in S_t at iteration t
\mathcal{F}_t	Natural filtration up to iteration t
ΔS_t	Number of points pruned in iteration t
π_t	Pruning probability at iteration t
R	Truncation parameter (maximum degree limit)
M	Number of processed points at a given level
t	Iteration index (time step)

C Omitted Proofs

C.1 Proof of Lemma 6

The following is the complete proof of Lemma 6.

Lemma 6. *Given a dataset $P \subset \mathbb{R}^d$ consisting of n points independently and uniformly distributed within a ball of radius ρ_0 , centered at p . we consider the SNG construction processed on the center point p with $\alpha = 1$. The probability π_t that a point p' is pruned in the t -th iteration is given by
$$\pi_t = \frac{I_{1-(\rho_t/\rho_0)^2}(\frac{d+1}{2}, \frac{1}{2}) - (\rho_t/\rho_0)^d \cdot I_{0.75}(\frac{d+1}{2}, \frac{1}{2})}{2(1-(\rho_t/\rho_0)^d)},$$
 where $I_x(a, b)$ denotes the regularized incomplete Beta function, and ρ_t is the distance between the nearest neighbor and p in t -th iteration. For large d , this probability satisfies $\frac{I_{0.75}(\frac{d+1}{2}, \frac{1}{2})}{2} < \pi_t < \frac{1}{2}$.*

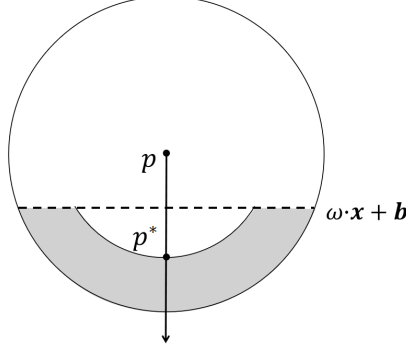


Figure 6: Given that p^* is the nearest point to p in t -th iteration and $\|p - p^*\| = \rho_t$. The light grey region represents all p' satisfied pruning condition $\|p - p'\| > \|p^* - p'\|$.

Proof of Lemma 6. Assuming a uniform distribution over the search space, the probability that a randomly sampled point falls within a region is proportional to the region's volume. Without loss of generality, we place the center point p at the origin and align the vector $\overrightarrow{pp^*}$ with the last coordinate axis, as shown in Figure 6. Under this coordinate system, the nearest neighbor p^* lies at the position $(0, 0, \dots, 0, \rho_t) \in \mathbb{R}^d$. Let p^* denote the nearest point to p in t -th iteration and $\|p - p^*\| = \rho_t$, then the hyperplane is defined by:

$$\omega \cdot x + b = 0, \quad \text{where } \omega = (0, \dots, 0, 1), \quad b = -\frac{\rho_t}{2}.$$

The pruned region at iteration t , denoted by $D(t)$, is defined as the intersection of the ball $B(p, \rho_0)$ and the half-space $\omega \cdot x > \frac{\rho_t}{2}$. Let $V(\cdot)$ denote the Lebesgue volume in the appropriate dimension. For example, $V(B_d(r))$ is the volume of a d -dimensional ball of radius r , and $V(B_{d-1}(r))$ is the volume of a $(d-1)$ -dimensional ball of radius r . In particular, $V(B_d(1))$ denotes the volume of the d -dimensional unit ball:

$$V(B_d(1)) = \frac{\pi^{d/2}}{\Gamma(1 + d/2)}.$$

To compute $V(D(t))$, we slice the ball orthogonally along the last coordinate x_d , and integrate over the $(d-1)$ -dimensional cross-sectional volumes. The volume of the pruned region is given by:

$$V(D(t)) = \int_{x_d = \frac{\rho_t}{2}}^{\rho_0} V\left(B_{d-1}\left(\sqrt{\rho_0^2 - x_d^2}\right)\right) dx_d,$$

where $B_{d-1}(\sqrt{\rho_0^2 - x_d^2})$ is the $(d-1)$ -dimensional ball of radius $\sqrt{\rho_0^2 - x_d^2}$, corresponding to the horizontal slice at height x_d . Using the change of variables $t = x_d/\rho_0$, we have $x_d = \rho_0 t$ and $dx_d = \rho_0 dt$. Substituting this into the integral yields:

$$V(D(t)) = V(B_{d-1}(1)) \cdot \rho_0^d \int_{\frac{\rho_t}{2\rho_0}}^1 (1 - t^2)^{\frac{d-1}{2}} dt.$$

Next, we apply the substitution $z = 1 - t^2$ to convert the integral into a standard form. Under this change of variables, we obtain:

$$\int_{\frac{\rho_t}{2\rho_0}}^1 (1 - t^2)^{\frac{d-1}{2}} dt = \frac{1}{2} \int_0^{1 - \left(\frac{\rho_t}{2\rho_0}\right)^2} z^{\frac{d-1}{2}} (1 - z)^{-1/2} dz.$$

This integral corresponds to the incomplete Beta function. After normalization, it becomes proportional to the *regularized incomplete Beta function*, also known as the cumulative distribution function of the Beta distribution, denoted by $I_x(a, b)$. Thus, we have:

$$\int_{\frac{\rho_t}{2\rho_0}}^1 (1 - t^2)^{\frac{d-1}{2}} dt = \frac{\Gamma\left(\frac{d+1}{2}\right) \Gamma\left(\frac{1}{2}\right)}{2\Gamma\left(\frac{d}{2} + 1\right)} \cdot I_{1 - \left(\frac{\rho_t}{2\rho_0}\right)^2}\left(\frac{d+1}{2}, \frac{1}{2}\right),$$

where $\Gamma(\cdot)$ denotes the Gamma function. Hence, the volume of the pruned region is:

$$V(D(t)) = \frac{\Gamma(\frac{d+1}{2})\Gamma(\frac{1}{2})}{2\Gamma(\frac{d}{2}+1)} V(B_{d-1}(1)) \cdot \rho_0^d \cdot I_{1-(\frac{\rho_t}{2\rho_0})^2} \left(\frac{d+1}{2}, \frac{1}{2} \right).$$

However, since this includes points within the inner ball $B(\rho_t)$, and we subtract the volume within $B(\rho_t)$, denoted V_{inner} , is:

$$V_{\text{inner}} = \frac{\Gamma(\frac{d+1}{2})\Gamma(\frac{1}{2})}{2\Gamma(\frac{d}{2}+1)} V(B_{d-1}(1)) \cdot \rho_t^d \cdot I_{3/4} \left(\frac{d+1}{2}, \frac{1}{2} \right),$$

where $3/4 = 1 - (1/2)^2$ accounts for the halfway-positioned hyperplane. The probability that a randomly sampled point is pruned in iteration t is:

$$\pi_t = \frac{V(D(t)) - V_{\text{inner}}}{V(B_d(\rho_0)) - V(B_d(\rho_t))}.$$

Substituting the expressions above and simplifying using $V(B_d(1)) = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}$, we get:

$$\pi_t = \frac{I_{1-(\frac{\rho_t}{2\rho_0})^2} - \left(\frac{\rho_t}{\rho_0}\right)^d \cdot I_{3/4}}{2 \left(1 - \left(\frac{\rho_t}{\rho_0}\right)^d\right)}.$$

Since $\frac{\rho_t}{\rho_0} \in (0, 1)$ and $1 - \left(\frac{\rho_t}{2\rho_0}\right)^2 \in (3/4, 1)$, the ratio is always decreases with respect to $\frac{\rho_t}{\rho_0}$. Specifically, noting that $I_{3/4}(a, b) < 1$, for large d , we derive:

$$\frac{1}{2} I_{3/4} \left(\frac{d+1}{2}, \frac{1}{2} \right) < \pi_t < \frac{1}{2}.$$

2-dimensional case. In the two-dimensional case, we can leverage geometric relationships to derive a lower bound of $\frac{1}{3}$ for the pruning probability. This bound corresponds to the scenario where the pruning probability decreases as ρ_t/ρ_0 increases, reaching its minimum when $\rho_t/\rho_0 = 1$ —that is, when the distance from the center point p to its nearest neighbor p^* approaches the radius of the disk. In this situation, the pruning decision boundary can be approximated by the perpendicular bisector of the segment connecting p and p^* . Geometrically, this bisector divides the circular region into two sectors, and the prunable region corresponds to the sector opposite p . As the angle subtended by this region exceeds $\frac{2\pi}{3}$, it follows that the fraction of the total area being pruned is at least $\frac{1}{3}$. Thus, the pruning probability is bounded below by $\frac{1}{3}$ in the two-dimensional setting. \square

C.2 Conservative Probability Estimation

Applicability of Pruning Probability to General Points. Our probability calculations are specifically designed for the scenarios where the reference point is located at the center of the sphere, ensuring uniform density in all directions. Under the assumption of a uniform distribution, each point in the dataset can be locally approximated as having uniform density in its neighborhood. Consequently, the pruning probability derived in Lemma 6 extends naturally to arbitrary points in the dataset, justifying its applicability throughout the SNG construction process.

Overlap. We also note that in later iterations, the pruning region at step t may overlap with those from earlier steps. This geometric overlap does not invalidate our degree bound—in fact, it tends to have faster elimination of candidates and thus requires fewer steps (i.e., fewer edges). In the proof of Lemma 6, any point that lies within the overlapping regions of multiple pruning areas remains eligible for removal, effectively undergoing repeated pruning. Therefore, our earlier volume-based estimate, which assumes disjoint pruning regions and ignores such overlaps, is conservative.

C.3 Proof of Lemma 8

Proof of Lemma 8. Part (i): We aim to show that $|S'_{t_1}| \geq n^{\nu_1}$ for $t_1 = O(n^{\nu_2})$. To that end, define the first passage time t_1 as the smallest t such that $|S'_t| \geq n^{\nu_1}$. Our goal to prove that this event occurs with probability one, i.e.,

$$\Pr(t_1 \leq Cn^{\nu_2}) = 1$$

for some $C > 0$. Formally:

$$\Pr(t_1 = O(n^{\nu_2})) = \Pr\left(\bigcup_{C>0} \bigcup_{n_0>0} \bigcap_{n>n_0} \{t_1 \leq Cn^{\nu_2}\}\right) = \Pr\left(\bigcup_{C>0} \bigcup_{n_0>0} \bigcap_{n>n_0} \{|S'_{Cn^{\nu_2}}| \geq n^{\nu_1}\}\right).$$

This is equivalent to demonstrating that:

$$\Pr\left(\bigcap_{C>0} \bigcap_{n_0>0} \bigcup_{n>n_0} \{|S'_{Cn^{\nu_2}}| < n^{\nu_1}\}\right) = 0.$$

Now, consider the probability $\Pr(|S'_{Cn^{\nu_2}}| < n^{\nu_1})$. Since $|S'_t| = \sum_{i=1}^t (\Delta S_i + 1)$, for $t = Cn^{\nu_2}$, we have:

$$|S'_{Cn^{\nu_2}}| = \sum_{i=1}^{Cn^{\nu_2}} (\Delta S_i + 1) \geq Cn^{\nu_2} \cdot \min_{1 \leq i \leq Cn^{\nu_2}} (\Delta S_i + 1).$$

Thus, we have:

$$\begin{aligned} \Pr(|S'_{Cn^{\nu_2}}| < n^{\nu_1}) &\leq \Pr\left(Cn^{\nu_2} \cdot \min_{1 \leq i \leq Cn^{\nu_2}} (\Delta S_i + 1) < n^{\nu_1}\right) \\ &= \Pr\left(\min_{1 \leq i \leq Cn^{\nu_2}} \Delta S_i < \frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}}\right). \end{aligned}$$

Since $\nu_1 > \nu_2$, for large n , we have $n^{\nu_1} \gg Cn^{\nu_2}$, which implies that $\frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}} \approx \frac{n^{\nu_1}}{Cn^{\nu_2}}$. We apply the Chernoff bound (Lemma 10) to estimate $\Pr(\Delta S_i \leq (1 - \delta)E[\Delta S_i])$:

$$\Pr(\Delta S_i \leq (1 - \delta)E[\Delta S_i]) \leq \exp\left(-\frac{\delta^2 E[\Delta S_i]}{2}\right) \leq \exp\left(-\frac{\delta^2 (|S_{i-1}| - 1) I_{0.75}\left(\frac{d+1}{2}, \frac{1}{2}\right)}{4}\right),$$

where $E[\Delta S_i] = \pi_i(|S_{i-1}| - 1)$ and $\pi_i > \frac{I_{0.75}\left(\frac{d+1}{2}, \frac{1}{2}\right)}{2}$. We set $1 - \delta = \frac{\frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}}}{E[\Delta S_i]} \geq \frac{\frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}}}{n/2}$, noting that $E[\Delta S_i] \leq |S_{i-1}| - 1 \approx n$ early in the process. Thus, we obtain:

$$\Pr(|S'_{Cn^{\nu_2}}| < n^{\nu_1}) \leq \exp\left(-\frac{\left(1 - \frac{\frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}}}{n/2}\right)^2 (n - n^{\nu_2} - 1) I_{0.75}\left(\frac{d+1}{2}, \frac{1}{2}\right)}{4}\right).$$

The series $\sum_{n>0} \exp\left(-\frac{\left(1 - \frac{\frac{n^{\nu_1} - Cn^{\nu_2}}{Cn^{\nu_2}}}{n/2}\right)^2 (n - n^{\nu_2} - 1) I_{0.75}\left(\frac{d+1}{2}, \frac{1}{2}\right)}{4}\right)$ converges, as the exponent becomes increasingly negative with n . By the Borel-Cantelli lemma (Lemma 9), we have $\Pr\left(\bigcap_{C>0} \bigcap_{n_0>0} \bigcup_{n>n_0} \{|S'_{Cn^{\nu_2}}| < n^{\nu_1}\}\right) = 0$, which implies that $\Pr(t_1 = O(n^{\nu_2})) = 1$.

Part (ii): The proof follows a similar approach. We need to demonstrate that $|S'_{t_2}| \geq n - n^{1-\nu}$ for $t_2 = O(n^\nu)$. Let t_2 be the first passage such that $|S'_t| \geq n - n^{1-\nu}$. We compute:

$$\Pr(t_2 = O(n^\nu)) = \Pr\left(\bigcup_{C>0} \bigcup_{n_0>0} \bigcap_{n>n_0} \{|S'_{Cn^\nu}| \geq n - n^{1-\nu}\}\right).$$

Similarly, we have:

$$\Pr(|S'_{Cn^\nu}| < n - n^{1-\nu}) \leq \Pr\left(\min_{1 \leq i \leq Cn^\nu} \Delta S_i < \frac{n - n^{1-\nu} - Cn^\nu}{Cn^\nu}\right).$$

Let $1 - \delta \geq \frac{n - n^{1-\nu} - Cn^\nu}{n/2}$. The Chernoff bound provides a convergent series:

$$\sum_{n>0} \exp\left(-\frac{\left(1 - \frac{n - n^{1-\nu} - Cn^\nu}{n/2}\right)^2 (n - n^\nu - 1) I_{0.75}\left(\frac{d+1}{2}, \frac{1}{2}\right)}{4}\right) < \infty.$$

By the Borel-Cantelli lemma, $\Pr\left(\bigcap_{C>0} \bigcap_{n_0>0} \bigcup_{n>n_0} \{|S'_{Cn^\nu}| < n - n^{1-\nu}\}\right) = 0$, which implies $\Pr(t_2 = O(n^\nu)) = 1$. \square

Part (i) indicates that the initial phase of SNG construction exhibits rapid growth, with the number of processed points scaling exponentially faster than the number of iterations. In contrast, part (ii) demonstrates that a sublinear number of iterations, on the order of $O(n^\nu)$, is sufficient to process nearly all points, achieving a growth magnitude close to n .

C.4 Proof of Theorem 1

Proof of Theorem 1. Let T denote the total number of iterations. Recall from Lemma 4 (3) that the maximum out-degree is T . Then our goal is to show $T = O(n^{2/3+\epsilon})$ with probability $1 - o(1)$. To analyze this, we divide the construction process into two phases based on the number of processed points $|S'_t|$. Specifically, we define the first phase as $t \leq t_1$, during which $|S'_t| < n - n^{1-\nu}$, and the second phase as $t_1 < t \leq T$, where $|S'_t| \geq n - n^{1-\nu}$, for any $\nu \in (2/3, 1)$.

First Phase ($t \leq t_1$): From Lemma 8 (part ii), for any $\nu \in (0, 1)$, with probability 1, the SNG construction reaches the $(n - n^{1-\nu})$ - $O(n^\nu)$ level. Thus, the number of iterations t_1 to process $n - n^{1-\nu}$ points is:

$$t_1 = O(n^\nu).$$

Second Phase ($t_1 < t \leq T$): When $|S'_t| \geq n - n^{1-\nu}$, the remaining points are $|S_t| = n - |S'_t| \leq n^{1-\nu}$. We denote the number of iterations in the second phase as $T_2 = T - t_1$, and aim to estimate how long it takes to complete the construction, i.e., to reach $|S'_T| = n - 1$.

Since at each step at least one point (the nearest neighbor) is added to the processed set, we trivially have $|S_i| \leq n^{1-\nu}$. However, to obtain a tighter estimate, we analyze the dynamics more precisely.

During this actual plateau phase, most iterations do not satisfy the pruning condition, and only remove the nearest neighbor. Significant pruning occurs in only a small fraction of steps. Intuitively, this is because the candidate set becomes small and the pruning probability π_t decays with it. Applying Markov's inequality, we can bound the probability that more than k points are pruned at iteration t :

$$\Pr(\Delta S_i \geq k) \leq \frac{\mathbb{E}[\Delta S_i]}{k} = \frac{\pi_i(|S_i| - 1)}{k} \leq \frac{n\pi_i}{k} = O(1),$$

for constant $k \in \mathbb{N}$. This justifies that $\pi_t = O(1/n)$ in this phase. The update rule for the processed set is:

$$|S'_{t+1}| - |S'_t| = \Delta S_{t+1} + 1, \quad \mathbb{E}[|S'_{t+1}| - |S'_t| \mid \mathcal{G}_t] = 1 + \pi_t(|S_t| - 1),$$

where \mathcal{F}_t is the natural filtration up to iteration t (defined in Section 2.2). Let $z(t/n) := |S'_t|/n$ be the normalized process, and approximate the expected increment by:

$$f(t, z) = 1 + \pi_t \cdot n(1 - z), \quad \text{with } \pi_t \cdot n = O(1).$$

Thus, f is Lipschitz in z , with constant $\theta = O(1)$, satisfying:

$$|f(t, z_1) - f(t, z_2)| \leq \theta |z_1 - z_2|.$$

This verifies the Lipschitz condition for the application of Lemma 11. The corresponding diffeential equation is:

$$\frac{dz}{dx} = 1 + \theta(1 - z), \quad z(0) = 1 - n^{-\nu},$$

whose solution is:

$$z(x) = 1 + \frac{1}{\theta} - \left(\frac{1}{\theta} + n^{-\nu} \right) e^{-\theta x}.$$

To determine when $|S'_t| = n - 1$, set $z(t/n) = 1 - \frac{1}{n}$ and solve:

$$1 - \frac{1}{n} = 1 + \frac{1}{\theta} - \left(\frac{1}{\theta} + n^{-\nu} \right) e^{-\theta \cdot t/n}.$$

which leads to:

$$e^{-\theta \cdot t/n} = \frac{\frac{1}{\theta} + \frac{1}{n}}{\frac{1}{\theta} + n^{-\nu}}, \quad \Rightarrow \quad t = \frac{n}{\theta} \ln \left(1 + \frac{n^{-\nu} - \frac{1}{n}}{\frac{1}{\theta} + \frac{1}{n}} \right).$$

Using the approximation $\ln(1 + x) \approx x$ for small x , we conclude:

$$t \approx \frac{n}{\theta} \cdot \frac{n^{-\nu} - \frac{1}{n}}{\frac{1}{\theta} + \frac{1}{n}} = O(n^{1-\nu}).$$

Thus:

$$T_2 = O(n^{1-\nu}).$$

Lemma 11 guarantees that this approximation holds with high probability. Specifically, for $\lambda = O(n^{-m})$, where $m \in (0, 3\nu - 2)$, the probability is:

$$1 - O\left(\frac{n^{1-\nu} + 1}{n^{-m}} \exp\{-n^{-3m-2+3\nu}\}\right) \rightarrow 1 \text{ as } n \rightarrow \infty.$$

Total Iterations: Combining both phases:

$$T = t_1 + T_2 = O(n^\nu) + O(n^{1-\nu}) = O(n^\nu),$$

for $\nu \in (2/3, 1)$, thus for any $\epsilon > 0$, we conclude:

$$T = O(n^{2/3+\epsilon}).$$

with probability 1.

Case of $\alpha > 1$. When $\|p - p'\| > \alpha \cdot \|p^* - p'\|$, the pruning region is no longer a simple intersection of a hyperplane and a sphere, but in fact corresponds to the intersection of two spheres. The resulting volume is given by:

$$\frac{1}{2} \frac{\pi^{n/2}}{\Gamma(\frac{n}{2} + 1)} r^n I_{1-h^2/(\rho_0)^2} \left(\frac{n+1}{2}, \frac{1}{2} \right),$$

where ρ_0 is the radius of the ball, h is the distance between centers, and $I_x(a, b)$ denotes the regularized incomplete Beta function, with the detailed derivation provided in [23]. Due to the computability of this volume, the pruning probability can be updated and **remains bounded**. The proof technique can be adapted accordingly, thus preserving the conclusions regarding the degree bound. \square

C.5 Full Proof to Theorem 2

Proof of Theorem 2. We analyze the expected length k of a GreedySearch path from p to the query $v_k \{p = v_0, v_1, \dots, v_k\}$, where each node selects its neighbor closest to the query point. Let η denote the out-degree (number of neighbors) of v_k , and assume that points in P are independently and uniformly distributed in a d -dimensional ball of radius ρ_0 .

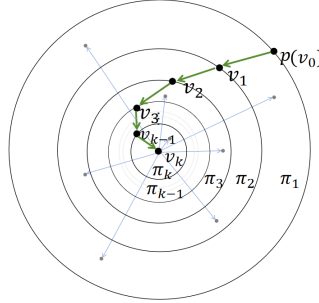


Figure 7: Path and neighbor distribution diagram: Blue lines represent edges from v_k to its neighbors, and green lines represent the search path. π_i denotes the number of neighbors in the i -th annular layer from the outermost to the innermost.

Define Δr as the minimum distinguishable pairwise distance difference among all point triples:

$$\Delta r = \min\{\|a - b\| - \|a - c\|, \|a - b\| - \|b - c\|, \|a - c\| - \|b - c\| \mid a, b, c \in P, \text{ distinct}\}$$

as introduced in [12]. Let v_k be the final node of the search path. We construct concentric balls centered at v_k with radii $\|v_k - v_i\|$ for $i = 0, 1, \dots, k-1$, and define the annular layer $A_i = B(v_k, \|v_k - v_{i-1}\|) \setminus B(v_k, \|v_k - v_i\|)$. Let η_i be the number of neighbors in each layer A_i . Then $\eta_i \sim \text{Bin}\left(\eta, \frac{V(A_i)}{V(B(v_k, \rho_0))}\right)$, and the expected number in each layer is proportional to the volume difference:

$$\mathbb{E}[\eta_i] = \eta \cdot \frac{V(B(v_k, \|v_k - v_{i-1}\|)) - V(B(v_k, \|v_k - v_i\|))}{V(B(v_k, \rho_0))}.$$

Summing over all k layers, we have:

$$E \left[\sum_{i=1}^k \eta \cdot \frac{V(B(v_k, \|v_k - v_{i-1}\|)) - V(B(v_k, \|v_k - v_i\|))}{V(B(v_k, R))} \right] \leq n \cdot \frac{V(B(v_k, \|p - v_k\|))}{V(B(v_k, R))}.$$

This upper bound is based on the fact that all points in the ball may serve as neighbors. Using $V(B(v_k, r)) \propto r^d$, we obtain:

$$E \left[\sum_{i=1}^k \eta \cdot \frac{\|v_k - v_{i-1}\|^d - \|v_k - v_i\|^d}{\rho_0^d} \right] \leq n \cdot E \left[\frac{\|p - v_k\|^d}{\rho_0^d} \right]. \quad (1)$$

Now, note that

$$\|v_k - v_{i-1}\|^d - \|v_k - v_i\|^d = (\|v_k - v_{i-1}\| - \|v_k - v_i\|) \sum_{j=0}^{d-1} \|v_k - v_{i-1}\|^j \|v_k - v_i\|^{d-1-j}.$$

Since $\|v_k - v_{i-1}\| - \|v_k - v_i\| \geq \Delta r$ and $\|v_k - v_i\| \geq \|v_k - v_{k-1}\|$ for all i , we have:

$$\|v_k - v_{i-1}\|^d - \|v_k - v_i\|^d \geq d \cdot \Delta r \cdot \|v_k - v_{k-1}\|^{d-1}.$$

Substituting into Equation 1, we obtain:

$$\mathbb{E}[k \cdot \eta \cdot d \cdot \Delta r \cdot \|v_k - v_{k-1}\|^{d-1}] \leq n \cdot \mathbb{E}[\|p - v_k\|^d].$$

Assuming independence between path length and distances, we rearrange:

$$E[k \cdot \eta] \leq n \cdot \frac{E[\|p - v_k\|^d]}{\Delta r \cdot d \cdot E[\|v_k - v_{k-1}\|^{d-1}]}. \quad (2)$$

Next, we upper bound $\|p - v_k\|$. Since the search path is monotonic:

$$\|p - v_k\| = \|v_0 - v_k\| \leq \|v_0 - v_k\| \cdot \prod_{i=1}^{k-1} \frac{\|v_i - v_k\|}{\|v_{i-1} - v_k\|} \cdot \frac{\|p - v_k\|}{\|v_{k-1} - v_k\|}.$$

Because $\frac{\|v_i - v_k\|}{\|v_{i-1} - v_k\|} \leq \frac{\rho_0 - \Delta r}{\rho_0}$ and $\frac{\|p - v_k\|}{\|v_{k-1} - v_k\|} \leq \frac{\rho_0}{\Delta r}$, we have:

$$\|p - v_k\|^d \leq \frac{\rho_0^{2d}}{\Delta r^d} \cdot \left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d(k-1)}.$$

Substituting this into Equation 2, we obtain:

$$\mathbb{E}[k \cdot \eta] \leq \frac{n \cdot \rho_0^{2d} \cdot \mathbb{E} \left[\left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d(k-1)} \right]}{d \cdot \Delta r^{d+1} \cdot \mathbb{E}[\|v_k - v_{k-1}\|^{d-1}]}.$$

Using the bound $\mathbb{E}[\|v_k - v_{k-1}\|] \geq \frac{\rho_0}{(n+1)^{1/d}}$ from [29] and Jensen's inequality (Lemma 12), we find:

$$\mathbb{E}[\|v_k - v_{k-1}\|^{d-1}] \geq \left(\frac{\rho_0}{(n+1)^{1/d}} \right)^{d-1}.$$

Thus:

$$\begin{aligned} E[k \cdot \eta] &\leq \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1} \cdot E \left[\left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d \cdot (k-1)} \right]}{\Delta r^{d+1} \cdot d} \\ &\leq \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1} \cdot E \left[- \left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d \cdot (k-1)} + 2 \right]}{\Delta r^{d+1} \cdot d}. \end{aligned}$$

Applying Jensen's inequality again:

$$E[k \cdot \eta] \leq \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1} \cdot \left(- \left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d \cdot \mathbb{E}[k-1]} + 2 \right)}{\Delta r^{d+1} \cdot d}$$

Define the function:

$$g(x) := \eta \cdot x - \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1}}{d \cdot \Delta r^{d+1}} \left[- \left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d(x-1)} + 2 \right].$$

We have $E(k) < 0$. Evaluating:

$$g(0) < 0,$$

$$g'(x) = \eta + \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1}}{\Delta r^{d+1} \cdot d} \cdot d \cdot \ln \left(\frac{R}{R - \Delta r} \right) \cdot \left(\frac{R - \Delta r}{R} \right)^{d \cdot x} > 0,$$

$$g(\log n) = \eta \log n + \frac{n(n+1)^{(d-1)/d} \rho_0^{d+1}}{\Delta r^{d+1} \cdot d} \left[\left(\frac{R - \Delta r}{R} \right)^{d \cdot (\log n - 1)} - 2 \right].$$

we have $\left(\frac{\rho_0 - \Delta r}{\rho_0} \right)^{d \cdot \log n} = e^{d \cdot \log n \cdot \ln \frac{\rho_0 - \Delta r}{\rho_0}} = n^{C' \cdot d \cdot \ln \frac{\rho_0 - \Delta r}{\rho_0}} > 2$. Thus:

$$g(\log n) > 0.$$

Since $g(x)$ is monotonically increasing and transitions from negative to positive, we conclude that $g(\log n) > 0$ and $g(E(k)) < 0$. Hence, $E(k) < \log n$, leading to:

$$E[k] = O(\log n).$$

□

D Foundations of Probability Spaces

In this section, we provide formal definitions of the foundational concepts used in our probabilistic analysis, including σ -algebras, probability spaces, random variables, and adapted processes. For more detailed content related to probability, refer to [2].

Definition 13 (σ -algebra). Ω is the sample space. A nonempty class \mathcal{F} of subsets of Ω is a σ -algebra on Ω if

- (i). $A^c \in \mathcal{F}$ whenever $A \in \mathcal{F}$,
- (ii). $\cup_{n=1}^{\infty} A_n \in \mathcal{F}$ whenever $A_n \in \mathcal{F}, n \geq 1$.

(i.e. \mathcal{F} is closed under complement and countable union.) The pair (Ω, \mathcal{F}) is called a measurable space. The sets of \mathcal{F} are called measurable sets.

Definition 14 (Probability Space). A probability space is a triple $(\Omega, \mathcal{F}, \mathbb{P})$, where:

- (i). Ω is the sample space (set of all possible outcomes).
- (ii). \mathcal{F} is a σ -algebra over Ω .
- (iii). $\mathbb{P} : \mathcal{F} \rightarrow [0, 1]$ is a probability measure.

Definition 15 (Random Variable). $(\Omega, \mathcal{F}, \mathbb{P})$ is a probability space. A random variable X is a measurable function $X : \Omega \rightarrow \mathbb{R}$, meaning that for any Borel set $B \subseteq \mathbb{R}$, the preimage $X^{-1}(B) \in \mathcal{F}$. This ensures that we can assign probabilities to events of the form $X \in B$.

Definition 16 (Adapted Process). Let $\{\mathcal{F}_t\}_{t \geq 0}$ be a filtration (i.e., an increasing sequence of σ -algebras) on $(\Omega, \mathcal{F}, \mathbb{P})$. A stochastic process $\{X_t\}_{t \geq 0}$ is said to be adapted to the filtration $\{\mathcal{F}_t\}$ if, for every t , the random variable X_t is \mathcal{F}_t -measurable. This means that the value of X_t is known given the information available up to time t .

E Algorithms

Below is the detailed RobustPrune algorithm used to do truncated SNG pruning for points in the datasets [30], corresponding to the graph construction process in Section 2.1.

Algorithm 2 RobustPrune(p, S, α, R)

Require: Graph G , point $p \in P$, candidate set S , relaxation parameter α , truncation parameter R

Ensure: G with neighbor set $N_{\text{out}}(p)$

- 1: Set $N_{\text{out}}(p) \leftarrow \emptyset$
 - 2: **while** $S \neq \emptyset$ **do**
 - 3: $p^* \leftarrow \arg \min_{s \in S} \|s - p\|$
 - 4: Add p^* to $N_{\text{out}}(p)$
 - 5: Remove p^* from S
 - 6: **if** $|N_{\text{out}}(p)| = R$ **then**
 - 7: break
 - 8: **end if**
 - 9: **for** $p' \in S$ **do**
 - 10: **if** $\|p - p'\| \geq \alpha \cdot \|p^* - p'\|$ **then**
 - 11: Remove p' from S
 - 12: **end if**
 - 13: **end for**
 - 14: **end while**
 - 15: **return** G with the neighbor points of p
-

Below, we present the overall construction algorithm—Vamana—which consists of two pruning steps and one search step. The algorithm iterates through all points $p \in P$ in a random order to construct the graph. Specifically, in line 11 of the Vamana algorithm, reverse edges are added to enhance the graph's connectivity.

Algorithm 3 Vamana Algorithm

Require: Database P with n points where i -th point has coordinates x_i , parameters α , truncation parameter R

Ensure: Directed graph G over P with out-degree $\leq R$

```
1: Randomly initialize  $G$  to a random  $R$ -regular directed graph
2: Let  $s$  denote the medoid of dataset  $P$ 
3: Let  $\sigma$  denote a random permutation of  $1, 2, \dots, n$ 
4: for  $i = 1$  to  $n$  do
5:   Perform a GreedySearch from  $s$  to  $x_{\sigma(i)}$ ,  $\mathcal{V} \leftarrow$  set of all points visited along the search path
6:   Run RobustPrune( $\sigma(i)$ ,  $\mathcal{V}$ ,  $\alpha$ ,  $R$ ) using  $\mathcal{V}$  as the candidate set
7:   for all points  $j \in N_{\text{out}}(\sigma(i))$  do
8:     if  $|N_{\text{out}}(j) \cup \{\sigma(i)\}| > R$  then
9:       Run RobustPrune( $j$ ,  $N_{\text{out}}(j) \cup \{\sigma(i)\}$ ,  $\alpha$ ,  $R$ )
10:    else
11:      Update  $N_{\text{out}}(j) \leftarrow N_{\text{out}}(j) \cup \{\sigma(i)\}$ 
12:    end if
13:  end for
14: end for
```

F Complexity Analysis Details

Below is a detailed complexity analysis of each component in the Vamana algorithm. **Complexity of GreedySearch.** The GreedySearch algorithm iteratively selects the closest neighbor to the query and follows a path to an approximate nearest neighbor, evaluating all neighbors at each step. According to [12], the complexity of GreedySearch is determined by the product of the average path length and the average degree. Theorem 2 establishes that the path length is $O(\log n)$. Thus, for a graph with degree K , the search complexity is $O(K \cdot \log n)$.

Complexity of Pruning. In the SNG pruning algorithm, the outer loop checks if the candidate set S is empty. If not, it adds an edge to the current nearest neighbor. The inner loop evaluates all remaining points in S , determining whether to prune them based on the pruning rule. In the non-truncated SNG pruning process, the outer loop iterates at most $O(n^{2/3+\epsilon})$ times, as established by Theorem 1. The inner loop evaluates all points in the candidate set S , excluding the nearest neighbor, checking up to $n - 2$ points per iteration. This yields a complexity upper bound of $O(n^{5/3+\epsilon})$, tighter than the $O(n^2)$ estimate in [30]. For the truncated variant, RobustPrune, the outer loop runs at most R times, where R is the optimized truncation parameter. The inner loop evaluates up to $n - 2$ points, resulting in a worst-case complexity of $O(R \cdot n)$.

Total Complexity of Vamana. The following are the graph construction steps of the Vamana algorithm used in DiskANN. To begin with, the graph is randomly initialized as an R -regular graph. R is the maximum degree specified by the parameter sweep. To reduce the indexing time, for each point p in the dataset, GreedySearch is first performed starting from the centroid s to obtain the set of volunteer points, which are then used as candidate points for pruning, the complexity can be approximate by $O(R \cdot \log n)$ and written as $C_1 \cdot R \log n + b_1$ [12]. Next, we apply RobustPrune to prune the randomized graph. This process runs for at most R iterations, and in each iteration, the number of points to be checked is approximately $O(R \log n)$. Considering the parameter α , as α increases, the graph has higher degree, hence the complexity can be regarded as inversely proportional to α . Therefore, the algorithm's complexity is estimated to be $O(R \cdot R \log n / \alpha)$ and can be expressed as $C_2(R \cdot R \log n / \alpha) + b_2$. To enhance GreedySearch convergence during the search phase, we add reverse edges for each neighbor of point p . This may cause a degree overrun beyond R . In such cases, RobustPrune is reapplied. For each point p with at most R possible neighbors, we add reverse edges and reconstruct the graph. The candidate set size does not exceed $R + 1$, and the inner loop iterates at most R times. A larger α increases the number of neighbors, raising the probability of a degree overrun, which we model as a proportional relationship, so the complexity is expressed as $C_3(\alpha \cdot R \cdot R^2) + b_3$.

G Discussion

Point Processes and Uniformity: A New Perspective on Assumptions. Dataset point distributions can often be modeled by a *spatial point process*, which is a random process used to describe the distribution of points in a given space. Formally, a spatial point process is a measurable mapping from a probability space into the space of locally finite point configurations in \mathbb{R}^d .

As elaborated in [26], a widely used model is the *Poisson point process*, which assumes that the number of points in any bounded region follows a Poisson distribution and that point locations are independent and identically distributed. Crucially, given a fixed number of points within a bounded region, the conditional distribution of those points under a Poisson point process is uniform. This property provides a natural justification for analyzing nearest neighbor graph constructions under the uniform distribution assumption, as the conditioning step aligns with the typical setup in ANNS frameworks such as DiskANN and SNG.

Future Research Directions. An important avenue for future research is to theoretically characterize the behavior of the SNG algorithm under diverse data distributions. Such analysis is crucial to ensure the robustness and effectiveness of SNG across datasets with varying structural properties. Advancing this line of work would further enhance the adaptability and generalizability of the proposed probability method, making it more suitable for real-world applications involving complex, non-uniform, and multimodal data distributions.

H Extended Results

H.1 Informations of Datasets

Below are the detailed specifications of the three datasets used, including their size, the size of the query set, and their dimension.

Table 2: Datasets

Dataset	NB Base Vectors	NB Query Vectors	Dimension
SIFT1M	1,000,000	10,000	128
GIST1M	1,000,000	1,000	960
DEEP1M	1,000,000	1,000	256

H.2 Pruning process on SIFT1M

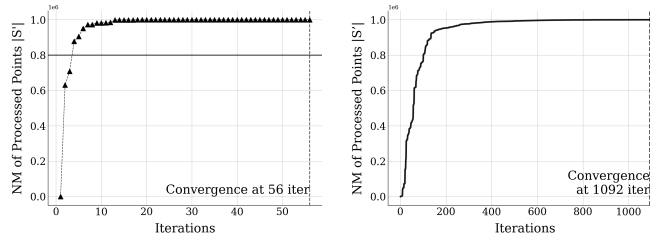


Figure 8: **Left Figure (General Pruning Process):** The pruning process for a sample point from SIFT1M is illustrated in the left figure. The horizontal axis represents the number of iteration steps, while the vertical axis indicates the number of processed points. Notably, only 4 steps are required to exceed the 80% threshold. However, the remaining pruning iterations are significantly prolonged, with the final 20% of points requiring 13 times the iterations of the initial points. This indicates that the algorithm’s speed diminishes quickly, leading to a slower rate of subsequent point scanning, with later stages consuming the majority of the scanning time. **Right Figure (An Extreme Case):** The pruning process for the mean point shows a similar trend of rapid initial progress followed by a slow phase. However, the slower pruning phase dominates, requiring 1092 steps to complete.

H.3 Query performance on GMM

Below are the test results for the search performance on the GMM dataset, where latency is measured in microseconds. It can be observed that the latency corresponding to a 99% recall rate does not exceed 3 ms.

Table 3: Query Processing Performance on GMM

L / Beamwidth	QPS	Mean Latency	99.9 Latency	Mean IO (us)	CPU (s)	Recall@10
10 / 2	35,187.99	768.96	2,115.00	746.02	3.90	97.23
20 / 2	19,995.63	1,442.69	2,676.00	1,403.40	6.15	99.96
30 / 2	13,505.18	2,048.59	3,706.00	1,992.13	10.46	100.00
40 / 2	10,163.05	2,723.53	4,740.00	2,652.44	11.80	100.00
50 / 2	8,135.30	3,375.69	5,658.00	3,289.35	14.62	100.00
100 / 2	4,058.43	6,859.58	24,449.00	6,695.53	29.03	100.00

H.4 Degree Distribution of GMM for other α values

Below, we present the degree distribution of the GMM dataset under different α values, all of which conform to the theoretical bound.

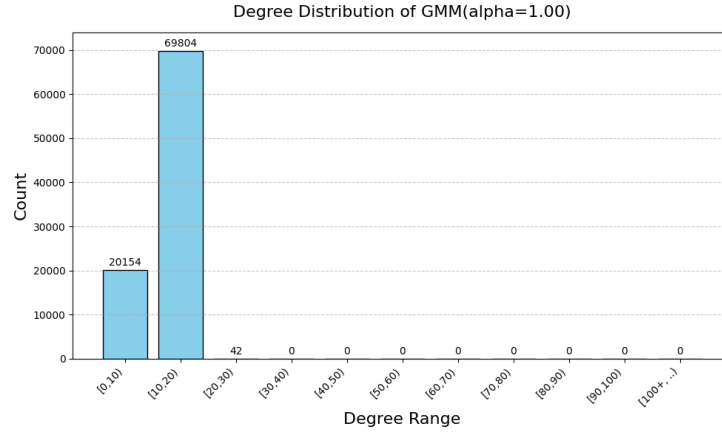


Figure 9: Degree distribution of GMM when $\alpha = 1.00$

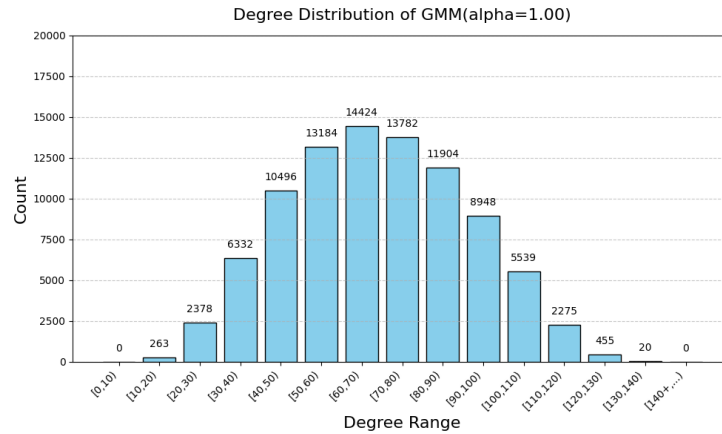


Figure 10: Degree distribution of GMM when $\alpha = 1.25$

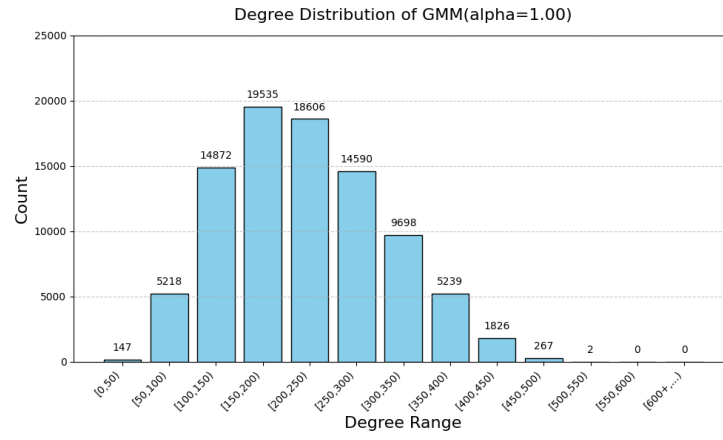


Figure 11: Degree distribution of GMM when $\alpha = 1.50$