Smoothed Analysis of Local Search Algorithms

Bodo Manthey

University of Twente, Department of Applied Mathematics
Enschede, The Netherlands, b.manthey@utwente.nl

Abstract. Smoothed analysis is a method for analyzing the performance of algorithms for which classical worst-case analysis fails to explain the performance observed in practice. Smoothed analysis has been applied to explain the performance of a variety of algorithms in the last years.
One particular class of algorithms where smoothed analysis has been used successfully are local search algorithms. We give a survey of smoothed analysis, in particular applied to local search algorithms.

1 Smoothed Analysis

1.1 Motivation

The goal of the analysis of algorithms is to provide measures for the performance of algorithms. In this way, it helps to compare algorithms and to understand their behavior. The most commonly used method for the performance of algorithms is *worst-case analysis*. If an algorithm has a good worst-case performance, then this is a very strong statement and, up to constants and lower order terms, the algorithm should also perform well in practice. However, there are many algorithms that work surprisingly well in practice although they have a very poor worst-case performance. The reason for this is that the worst-case performance can be dominated by a few pathological instances that hardly or never occur in practice.

A frequently used alternative to worst-case analysis is *average-case analysis*. In average-case analysis, the expected performance is measured with respect to some fixed probability distribution. Many algorithms with poor worst-case but good practical performance show a good average-case performance. However, the drawback of average-case analysis is that random instances drawn according to some fixed probability distribution often have very special properties with high probability. These properties of random instances distinguish them from typical instances. Thus, a good average-case running-time does not necessarily explain a good practical performance.

In order to get a more realistic measure for the performance of algorithms in cases where worst-case analysis is too pessimistic, Spielman and Teng [56] proposed *smoothed analysis* as a new paradigm to analyze algorithms. The key idea is that practical inputs are often not pathological, but are subject to a small amount of random noise. This random noise can, for instance, stem from measurement errors. It can also come from numerical imprecision or other circumstances, where we have no reason to believe that these influences change the input in a worst-case manner.

1.2 Definition

In smoothed analysis, we measure the maximum expected running-time, where the maximum is taken over the (adversarial) choices of the adversary, and the expected value is taken over the random perturbation of the input. The random perturbation is controlled by some perturbation parameter.

In almost all cases, this *perturbation parameter* is either the standard deviation σ of the perturbation or an upper bound ϕ on the density of the underlying probability distributions. In the former case, larger σ means more randomness, and the analysis approaches the worst-case analysis for very small σ. This model is also called the *two-step model* of smoothed analysis. The most commonly used type of perturbations are Gaussian distributions of standard deviation σ.

In the latter case, smaller ϕ means more randomness, and the analysis approaches the worst-case analysis for large ϕ. This model is also called the *one-step model* of smoothed analysis.

We restrict ourselves here to the two-step model with Gaussian noise, and we define this model in the following. We assume that our instances $X = \{x_1, \ldots, x_n\}$ of size n consist of n points $x_i \in \mathbb{R}^d$ $(1 \leq i \leq n)$. We denote by $\mathcal{N}(\mu, \sigma^2)$ a d-dimensional Gaussian distribution with mean $\mu \in \mathbb{R}^d$ and variance σ^2 (more precisely, its covariance matrix is a diagonal matrix with σ^2 on all diagonal entries).

Assume that we have a performance measure m that maps instances to, e.g., the number of iterations that the algorithm under consideration needs on an instance X or the approximation ratio that the algorithm achieves on X. Then the worst-case performance as a function of the input size is given as

$$M_{\text{worst}}(n) = \max_{\substack{X = \{x_1, \ldots, x_n\} \\ \subseteq [0,1]^d}} \big(m(X)\big). \tag{1}$$

The average-case performance is given by

$$M_{\text{average}}(n) = \mathop{\mathbb{E}}_{\substack{Y = \{y_1, \ldots, y_n\} \\ y_i \sim \mathcal{N}(0,1)}} \big(m(Y)\big).$$

Here, the points y_i (for $1 \leq i \leq n$) are drawn according to independent d-dimensional Gaussian distributions with mean 0 and standard deviation 1. Another probability distribution that is frequently used is drawing the points independently and uniformly from the unit hypercube $[0,1]^d$.

The smoothed performance is a combination of both:

$$M_{\text{smoothed}}(n, \sigma) = \max_{\substack{X = \{x_1, \ldots, x_n\} \\ \subseteq [0,1]^d}} \mathop{\mathbb{E}}_{\substack{Y = \{y_1, \ldots, y_n\} \\ y_i \sim \mathcal{N}(x_i, \sigma^2)}} \big(m(Y)\big). \tag{2}$$

An adversary specifies the instance X, and then Y is obtained by perturbing the points in X.

Note that M_{smoothed} depends also on the perturbation parameter σ: For very small σ, we have $Y \approx X$ and the smoothed performance approaches the worst-case performance. For large σ, the influence of X is negligible compared to the perturbation, and the smoothed performance approaches the average-case performance.

Note further that we have restricted the choices of the adversary to points in $[0, 1]^d$. Assuming scale-invariance of the underlying problem, this is no restriction and makes no difference for worst-case analysis. For smoothed analysis, however, we would have to scale σ in the same way.

Moreover, we observe that the (adversarial) choice of X in (2) can be different from the choice of X in (1). In worst-case analysis, the adversary picks an instance with worst performance. In smoothed analysis, the adversary chooses an instance X that maximizes the expected performance subject to the perturbation.

Finally, we remark that we do not require that a feasible or optimal solution for X remains a feasible or an optimal solution for Y, respectively. Roughly speaking, we are interested in the distribution of difficult instances and if difficult instances are isolated. This does not require that we can obtain a solution for X from a solution for the instance Y obtained by perturbing X.

1.3 Overview of Results Besides Local Search

Since its invention, smoothed analysis has been applied to a variety of algorithms and problems using a variety of perturbation models. We do not discuss the models here, but only give an overview to which algorithms and problems smoothed analysis has been applied. We also refer to two surveys about smoothed analysis that highlight different perspectives of smoothed analysis [46, 57].

Linear programming and matrix problems. Smoothed analysis has originally been applied to the simplex method [56]. This analysis has subsequently been improved and simplified significantly [28, 59]. Besides this, smoothed analysis has been applied to a variety of related algorithms and problems such as the perceptron algorithm [13], interior point methods [55], and condition numbers [19, 20, 29, 52, 58].

Integer programming and multi-criteria optimization. Starting with a smoothed analysis of the knapsack problem [7], a significant amount of research has been dedicated to understanding the solvability of integer programming problems and the size of Pareto curves in multi-criteria optimization problems [6, 8, 10, 16, 17, 49–51]. Beier and Vöcking's characterization of integer programming problems that can be solved in smoothed polynomial time [8] inspired an embedding of smoothed analysis into the existing worst-case and average-case complexity theory [11].

Graphs and formulas. Smoothed analysis can also be applied to purely discrete problems such as satisfiability of Boolean formulas [24, 34, 41] or graph problems [35, 41, 44, 54]. However, it is much less obvious what a meaningful perturbation model is than in problems involving numbers.

Sorting and searching. Smoothed analysis has been applied to analyze problems based on permutations, most notably the quicksort algorithm [4, 27, 36, 45].

Approximation ratios. Smoothed analysis has mostly been applied to analyze the running-time of algorithms, but there are also a few analyses of approximation ratios for Euclidean optimization problems [12, 26] and packing problems [26, 39].

Other applications. Other applications of smoothed analysis to concrete algorithms include online algorithms [5, 53], algorithms for computing minimum cost flows [15, 25], computational geometry [9, 22], finding Nash equilibria [23], PAC learning [38], computing the edit distance [1], minimizing concave functions [40], balancing networks [37], and belief propagation for discrete optimization problems [14].

2 Local Search Algorithms

Local search algorithms are often very powerful tools to compute near-optimal solutions for hard combinatorial optimization problems. Starting from an initial solution, they iteratively try to improve the solution by small changes, until they terminate in a local optimum. While often showing a surprisingly good performance in practice, the theoretical performance of many local search heuristics is poor.

Smoothed Analysis has successfully been used to bridge the gap between the theoretical prediction of performance and the performance observed in practice and to explain the practical performance of a couple of local search algorithms. In most cases, the number of iterations until a local optimum is reached has been analyzed. Examples of local search algorithms whose running-time has been analyzed in the framework of smoothed analysis include the 2-opt heuristic for the traveling salesman problem (TSP) [31, 48], the iterative closest point (ICP) algorithm to match point clouds [3], the k-means method for clustering [2, 3, 47], and the flip heuristic for the maximum cut problem [30, 33].

Only a few results are known about the smoothed approximation ratio of local search algorithms. Examples are the 2-opt heuristic for the TSP [31, 42] and the jump and lex-jump heuristic in scheduling [18, 32].

In the following, we briefly sketch the main ideas how these results have been obtained.

2.1 Smoothed Analysis of the Running-Time

The key idea of all smoothed analyses of running-times of local search heuristics is the following: we use the objective function to measure progress. Then we show that, after perturbation, the objective function decreases (in case of a minimization problem) significantly with high probability either in every iteration or in every sequence of iterations.

More precisely: assume that the objective value of our initial solution is at most I, and assume further that the objective value decreases by at least δ in every iteration of the local search algorithm. Then (assuming that the objective value cannot become negative) we must reach a local optimum within at most I/δ iterations. An upper bound I for the initial solution is often relatively easy to get, and usually one that holds with high probability suffices. Thus, the main task is to analyze the minimal improvement δ.

The general outline to analyze δ is as follows: often, it is quite straightforward to show that the probability that some fixed iteration yields a small improvement is small. Then a simple union bound over all possible iterations yields a first bound for the probability that δ is small. However, the number of possible iterations can be quite large, which renders this bound useless. Thus, the goal is to analyze similar iterations together to avoid the wasteful and naive union bound. Hence, we want to come up with as few classes as possible such that for every class, we can show that it is unlikely that it contains an iteration that yields only a small improvement.

For the 2-opt heuristic for the TSP, one can get polynomial bounds for the smoothed running-time by considering single iterations, although better bounds can be obtained by considering pairs of iterations that share an edge [31].

For the k-means method for clustering, considering single iterations does not seem to be sufficient. In the case that the clustering does not change much from iteration to iteration, it seems to be possible that very small improvements occur. However, it is unlikely that a short sequence of such iterations yields only very small improvements [2]. Even more iterations have been considered together to analyze the flip heuristic for the maximum cut problem [33].

2.2 Smoothed Analysis of the Approximation Ratio

Much less is known about the smoothed approximation ratios of local search algorithms than about their smoothed running-time. This might be because the approximation ratio depends heavily on the initialization, and the worst local optima are often quite robust against slight perturbations. In light of this, the running-time becomes crucial for the approximation performance: if the local search heuristic terminates very quickly, we can afford to run it many times with different initializations. Hopefully, at least one initialization yields a good solution.

One way to get rid of the dependency of the initialization in the analysis is to compare the worst local optimum to the global optimum [31, 42]. This keeps the analysis tractable, but still often leads to results that are too pessimistic to reflect the performance observed in practice. In the following, we denote by WLO the objective value of the worst local optimum and by OPT the objective value of a global optimum.

A second technical difficulty is that WLO and OPT are not independent, and we would like to analyze their ratio. The simplest approach to circumvent this challenge is to replace WLO by a worst-case upper bound. While this again seems too pessimistic, it simplifies the analysis a lot: we are only left with analyzing $\mathbb{E}(\frac{1}{\text{OPT}})$ instead of $\mathbb{E}(\frac{\text{WLO}}{\text{OPT}})$. This approach has in particular been used for the 2-opt heuristic for the TSP. For the 2-opt heuristic, it is known that WLO = $O(n^{\frac{d-1}{d}})$ for tours of n points in $[0,1]^d$ [21]. This has been exploited by Englert et al. [31] to prove a bound on the smoothed approximation ratio of the 2-opt heuristic.

However, ignoring the dependency between global and local optimum has significant limitations. What is bad for the approximation ratio is a large WLO together with a small OPT. Intuitively, in terms of the TSP, we get a very short optimal tour if the points are very close. But then also WLO should be small. The other way around, if there is a locally optimal TSP tour that is very long, then the points cannot be too close to each other. Hence, OPT cannot be too small. This information has been exploited to prove that the 2-opt heuristic achieves smoothed approximation ratio of $O(\log(1/\sigma))$ [42].

Still, simple construction heuristics for the TSP achieve approximation ratios of 2. Thus, the obvious open problem concerning smoothed approximation ratios is to analyze hybrid heuristics consisting of a clever initialization together with local search (see also Section 3). (It has been shown that using the nearest-neighbor heuristic to initialize 2-opt does not yield a better bound than $\Omega(\log n/\log\log n)$ for sufficiently small σ [42].)

3 Open Problems

To conclude, we list three open problems concerning smoothed analysis of local search algorithms.

Lin-Kernighan heuristic for the TSP. The Lin-Kernighan heuristic [43] is an extremely powerful heuristic for finding near-optimal TSP tours quickly in practice. Unfortunately, different to the 2-opt heuristic, it seems to be difficult to describe iterations or sequences of iterations in a compact form in order to avoid a too wasteful union bound.

Flip heuristic for Max-Cut. Etscheid and Röglin [33] have recently shown that the smoothed number of iterations that the flip heuristic needs is bounded by a polynomial in $n^{\log n}$ and the perturbation parameter ϕ, where n is the number of nodes of the graph.

More general, we observe that the running-time of the flip heuristic is pseudo-polynomial. For integer programming problems, it is known that every problem that can be solved in pseudo-polynomial time can also be solved in smoothed polynomial time [8]. It would be interesting to see if something similar holds for local search heuristics, i.e., if every local search algorithm with pseudo-polynomial running-time has also smoothed polynomial running-time.

Approximation ratios with initialization. The existing results about smoothed approximation ratios of local search algorithms compare the worst local optimum to the global optimal solution [31, 42]. However, the performance of local search heuristics relies heavily on a good initialization. The 2-opt heuristic is no exception, and the smoothed guarantees for the approximation ratio are easily beaten by choosing the initial tour with a constant-factor approximation algorithm.

Consequently, an obvious open problem is to take into account clever initializations when analyzing the approximation ratios of local search algorithms.

References

1. Andoni, A., Krauthgamer, R.: The smoothed complexity of edit distance. ACM Transactions on Algorithms 8(4), 44:1–44:25 (2012)
2. Arthur, D., Manthey, B., Röglin, H.: Smoothed analysis of the k-means method. Journal of the ACM 58(5) (2011)
3. Arthur, D., Vassilvitskii, S.: Worst-case and smoothed analysis of the ICP algorithm, with an application to the k-means method. SIAM Journal on Computing 39(2), 766–782 (2009)
4. Banderier, C., Beier, R., Mehlhorn, K.: Smoothed analysis of three combinatorial problems. In: Rovan, B., Vojtás, P. (eds.) Proc. of the 28th Int. Symp. on Mathematical Foundations of Computer Science (MFCS). Lecture Notes in Computer Science, vol. 2747, pp. 198–207. Springer (2003)
5. Becchetti, L., Leonardi, S., Marchetti-Spaccamela, A., Schäfer, G., Vredeveld, T.: Average case and smoothed competitive analysis of the multilevel feedback algorithm. Mathematics of Operations Research 31(1), 85–108 (2006)
6. Beier, R., Röglin, H., Vöcking, B.: The smoothed number of Pareto optimal solutions in bicriteria integer optimization. In: Fischetti, M., Williamson, D.P. (eds.) Proc. of the 12th Int. Conf. on Integer Programming and Combinatorial Optimization (IPCO). Lecture Notes in Computer Science, vol. 4513, pp. 53–67. Springer (2007)
7. Beier, R., Vöcking, B.: Random knapsack in expected polynomial time. Journal of Computer and System Sciences 69(3), 306–329 (2004)
8. Beier, R., Vöcking, B.: Typical properties of winners and losers in discrete optimization. SIAM Journal on Computing 35(4), 855–881 (2006)
9. de Berg, M., Haverkort, H.J., Tsirogiannis, C.P.: Visibility maps of realistic terrains have linear smoothed complexity. Journal of Computational Geometry 1(1), 57–71 (2010)
10. Berger, A., Röglin, H., van der Zwaan, R.: Internet routing between autonomous systems: Fast algorithms for path trading. Discrete Applied Mathematics 185, 8–17 (2015)
11. Bläser, M., Manthey, B.: Smoothed complexity theory. ACM Transactions on Computation Theory (to appear)
12. Bläser, M., Manthey, B., Rao, B.V.R.: Smoothed analysis of partitioning algorithms for Euclidean functionals. Algorithmica 66(2), 397–418 (2013)
13. Blum, A.L., Dunagan, J.D.: Smoothed analysis of the perceptron algorithm for linear programming. In: Proc. of the 13th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA). pp. 905–914. SIAM (2002)

14. Brunsch, T., Cornelissen, K., Manthey, B., Röglin, H.: Smoothed analysis of belief propagation for minimum-cost flow and matching. Journal of Graph Algorithms and Applications 17(6), 647–670 (2013)
15. Brunsch, T., Cornelissen, K., Manthey, B., Röglin, H., Rösner, C.: Smoothed analysis of the successive shortest path algorithm. Computing Research Repository 1501.05493 [cs.DS], arXiv (2015), a preliminary version has been presented at SODA 2013
16. Brunsch, T., Goyal, N., Rademacher, L., Röglin, H.: Lower bounds for the average and smoothed number of pareto-optima. Theory of Computing 10(10), 237–256 (2014)
17. Brunsch, T., Röglin, H.: Improved smoothed analysis of multiobjective optimization. Journal of the ACM 62(1), 4 (2015)
18. Brunsch, T., Röglin, H., Rutten, C., Vredeveld, T.: Smoothed performance guarantees for local search. Mathematical Programming 146(1–2), 185–218 (2014)
19. Bürgisser, P., Cucker, F., Lotz, M.: Smoothed analysis of complex conic condition numbers. Journal de Mathématiques Pures et Appliquées 86(4), 293 – 309 (2006)
20. Bürgisser, P., Cucker, F., Lotz, M.: The probability that a slightly perturbed numerical analysis problem is difficult. Mathematics of Computation 77(263), 1559–1583 (2008)
21. Chandra, B., Karloff, H., Tovey, C.: New results on the old k-opt algorithm for the traveling salesman problem. SIAM Journal on Computing 28(6), 1998–2029 (1999)
22. Chaudhuri, S., Koltun, V.: Smoothed analysis of probabilistic roadmaps. Computational Geometry 42(8), 731–747 (2009)
23. Chen, X., Deng, X., Teng, S.H.: Settling the complexity of computing two-player Nash equilibria. Journal of the ACM 56(3) (2009)
24. Coja-Oghlan, A., Feige, U., Frieze, A.M., Krivelevich, M., Vilenchik, D.: On smoothed k-CNF formulas and the Walksat algorithm. In: Proc. of the 20th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA). pp. 451–460. SIAM (2009)
25. Cornelissen, K., Manthey, B.: Smoothed analysis of the minimum-mean cycle canceling algorithm and the network simplex algorithm. In: Proc. of the 21st Ann. Int. Computing and Combinatorics Conf. (COCOON). Lecture Notes in Computer Science, Springer (2015), to appear
26. Curticapean, R., Künnemann, M.: A quantization framework for smoothed analysis of euclidean optimization problems. In: Bodlaender, H.L., Italiano, G.F. (eds.) Proc. of the 21st Ann. European Symp. on Algorithms (ESA). Lecture Notes in Computer Science, vol. 8125, pp. 349–360. Springer (2013)
27. Damerow, V., Manthey, B., auf der Heide, F.M., Räcke, H., Scheideler, C., Sohler, C., Tantau, T.: Smoothed analysis of left-to-right maxima with applications. ACM Transactions on Algorithms 8(3) (2012)
28. Deshpande, A., Spielman, D.A.: Improved smoothed analysis of the shadow vertex simplex method. In: Proc. of the 46th Ann. IEEE Symp. on Foundations of Computer Science (FOCS). pp. 349–356. IEEE Computer Society (2005)
29. Dunagan, J., Spielman, D.A., Teng, S.H.: Smoothed analysis of condition numbers and complexity implications for linear programming. Mathematical Programming 126(2), 315–350 (2011)
30. Elsässer, R., Tscheuschner, T.: Settling the complexity of local max-cut (almost) completely. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) Proc. of the 38th Int. Coll. on Automata, Languages and Programming (ICALP). Lecture Notes in Computer Science, vol. 6755, pp. 171–182. Springer (2011)
31. Englert, M., Röglin, H., Vöcking, B.: Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP. Algorithmica 68(1), 190–264 (2014)

32. Etscheid, M.: Performance guarantees for scheduling algorithms under perturbed machine speeds. Discrete Applied Mathematics (to appear)
33. Etscheid, M., Röglin, H.: Smoothed analysis of local search for the maximum-cut problem. In: Proc. of the 25th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA). pp. 882–889. SIAM (2014)
34. Feige, U.: Refuting smoothed 3CNF formulas. In: Proc. of the 48th Ann. IEEE Symp. on Foundations of Computer Science (FOCS). pp. 407–417. IEEE Computer Society (2007)
35. Flaxman, A.D., Frieze, A.M.: The diameter of randomly perturbed digraphs and some applications. Random Structures and Algorithms 30(4), 484–504 (2007)
36. Fouz, M., Kufleitner, M., Manthey, B., Zeini Jahromi, N.: On smoothed analysis of quicksort and Hoare's find. Algorithmica 62(3-4), 879–905 (2012)
37. Friedrich, T., Sauerwald, T., Vilenchik, D.: Smoothed analysis of balancing networks. Random Structures and Algorithms 39(1), 115–138 (2011)
38. Kalai, A.T., Samorodnitsky, A., Teng, S.H.: Learning and smoothed analysis. In: Proc. of the 50th Ann. IEEE Symp. on Foundations of Computer Science (FOCS). pp. 395–404. IEEE Computer Society (2009)
39. Karger, D., Onak, K.: Polynomial approximation schemes for smoothed and random instances of multidimensional packing problems. In: Proc. of the 18th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA). pp. 1207–1216. SIAM (2007)
40. Kelner, J.A., Nikolova, E.: On the hardness and smoothed complexity of quasiconcave minimization. In: Proc. of the 48th Ann. IEEE Symp. on Foundations of Computer Science (FOCS). pp. 472–482 (2007)
41. Krivelevich, M., Sudakov, B., Tetali, P.: On smoothed analysis in dense graphs and formulas. Random Structures and Algorithms 29(2), 180–193 (2006)
42. Künnemann, M., Manthey, B.: Towards understanding the smoothed approximation ratio of the 2-opt heuristic. In: Proc. of the 42nd Int. Coll. on Automata, Languages and Programming (ICALP). Lecture Notes in Computer Science, Springer (2015), to appear
43. Lin, S., Kernighan, B.W.: An effective heuristic for the traveling-salesman problem. Operations Research 21(2), 498–516 (1973)
44. Manthey, B., Plociennik, K.: Approximating independent set in perturbed graphs. Discrete Applied Mathematics 161(12), 1761–1768 (2013)
45. Manthey, B., Reischuk, R.: Smoothed analysis of binary search trees. Theoretical Computer Science 378(3), 292–315 (2007)
46. Manthey, B., Röglin, H.: Smoothed analysis: Analysis of algorithms beyond worst case. it – Information Technology 53(6), 280–286 (2011)
47. Manthey, B., Röglin, H.: Worst-case and smoothed analysis of k-means clustering with Bregman divergences. Journal of Computational Geometry 4(1), 94–132 (2013)
48. Manthey, B., Veenstra, R.: Smoothed analysis of the 2-Opt heuristic for the TSP: Polynomial bounds for Gaussian noise. In: Cai, L., Cheng, S.W., Lam, T.W. (eds.) Proc. of the 24th Ann. Int. Symp. on Algorithms and Computation (ISAAC). Lecture Notes in Computer Science, vol. 8283, pp. 579–589. Springer (2013)
49. Moitra, A., O'Donnell, R.: Pareto optimal solutions for smoothed analysts. SIAM Journal on Computing 41(5), 1266–1284 (2012)
50. Röglin, H., Teng, S.H.: Smoothed analysis of multiobjective optimization. In: Proc. of the 50th Ann. IEEE Symp. on Foundations of Computer Science (FOCS). pp. 681–690. IEEE Computer Society (2009)
51. Röglin, H., Vöcking, B.: Smoothed analysis of integer programming. Mathematical Programming 110(1), 21–56 (2007)

52. Sankar, A., Spielman, D.A., Teng, S.H.: Smoothed analysis of the condition numbers and growth factors of matrices. SIAM Journal on Matrix Analysis and Applications 28(2), 446–476 (2006)
53. Schäfer, G., Sivadasan, N.: Topology matters: Smoothed competitiveness of metrical task systems. Theoretical Computer Science 241(1–3), 216–246 (2005)
54. Spielman, D.A., Teng, S.H.: Smoothed analysis: Motivation and discrete models. In: Dehne, F., Sack, J.R., Smid, M. (eds.) Proc. of the 8th Workshop on Algorithms and Data Structures (WADS). Lecture Notes in Computer Science, vol. 2748, pp. 256–270. Springer (2003)
55. Spielman, D.A., Teng, S.H.: Smoothed analysis of termination of linear programming algorithms. Mathematical Programming, Series B 97(1–2), 375–404 (2003)
56. Spielman, D.A., Teng, S.H.: Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. Journal of the ACM 51(3), 385–463 (2004)
57. Spielman, D.A., Teng, S.H.: Smoothed analysis: An attempt to explain the behavior of algorithms in practice. Communications of the ACM 52(10), 76–84 (2009)
58. Tao, T., Vu, V.H.: Smooth analysis of the condition number and the least singular value. Mathematics of Computation 79(272), 2333–2352 (2010)
59. Vershynin, R.: Beyond Hirsch conjecture: Walks on random polytopes and smoothed complexity of the simplex method. SIAM Journal on Computing 39(2), 646–678 (2009)