

# Daniel A. Spielman

## Research Overview

My research over the past decade has focused on problems at the intersection of Algorithms and Combinatorics with Optimization, Scientific Computation and Engineering. In my work, I develop mathematically rigorous analyses of algorithms and heuristics used to solve fundamental computational problems. I then try to use these analyses to obtain insight into how these problems can be better solved in practice.

In this overview, I describe my work in the areas of Smoothed Analysis, Combinatorial Scientific Computing, and Error-Correcting Codes. My work on Error-Correcting Codes has led to four patents, and has been extended and applied many times. I have spent much of the Winter of 2005 implementing my latest combinatorial scientific algorithm: a solver for certain families of linear equations. The experimental results we have obtained so far indicated that it may be the fastest solver for a large fragment of these families. We still do not know if Smoothed Analysis, the most ambitious and theoretical of my analyses, will lead to improvements in practice.

## Contents

<b>1</b>	<b>Smoothed Analysis</b>	<b>1</b>
1.1	What Smoothed Analysis is and Why I do it . . . . .	1
1.2	Papers on Smoothed Analysis . . . . .	3
<b>2</b>	<b>Combinatorial Scientific Computing</b>	<b>5</b>
2.1	Linear System Solvers . . . . .	5
2.2	Mesh and Graph Partitioning . . . . .	7
2.3	Mesh Generation . . . . .	8
<b>3</b>	<b>Error-Correcting Codes</b>	<b>9</b>
3.1	Codes Approaching Capacity . . . . .	9
3.2	Linear-Time Codes . . . . .	11

## 1 Smoothed Analysis

### 1.1 What Smoothed Analysis is and Why I do it

Shang-Hua Teng and I introduced smoothed analysis to provide a means of explaining the practical success of algorithms and heuristics that have poor worst-case behavior and for which average-case analysis was unconvincing. The problem of explaining the success of heuristics that “work in practice” has long plagued theoreticians. Many of these have poor worst-case complexity. While one may gain some insight into their performance by demonstrating that they have low average-case complexity, this analysis may be unconvincing as an average-case analysis is dominated by the performance of an algorithm on random

inputs, and these may fail to resemble the inputs actually encountered in practice. On this point, Condon, Edelsbrunner, Emerson, Fortnow, Haber, Karp, Leivant, Lipton, Lynch, Parberry, Papadimitriou, Rabin, Rosenberg, Royer, Savage, Selman, Smith, Tardos, and Vitter wrote (“Challenges for Theory of Computing: Report of an NSF-Sponsored Workshop on Research in Theoretical Computer Science” SIGACT News, 1999)

*While theoretical work on models of computation and methods for analyzing algorithms has had enormous payoffs, we are not done. In many situations, simple algorithms do well. Take for example the Simplex algorithm for linear programming, or the success of simulated annealing on certain supposedly intractable problems. We don't understand why! It is apparent that worst-case analysis does not provide useful insights on the performance of many algorithms on real data. Our methods for measuring the performance of algorithms and heuristics and our models of computation need to be further developed and refined . . . Developing means for predicting the performance of algorithms and heuristics on real data and on real computers is a grand challenge in algorithms.*

While we do not expect smoothed analysis to explain the performance of all such heuristics, we believe it can help explain the performance of many.

Smoothed analysis is a hybrid of worst-case and average-case analyses that inherits advantages of both. The smoothed complexity of an algorithm is the maximum over its inputs of the expected running time of the algorithm under slight random perturbations of that input. The smoothed complexity is then measured as a function of both the input length and the magnitude of the perturbations. If an algorithm has low smoothed complexity, then it should perform well on most inputs in *every* neighborhood of inputs. Smoothed analysis makes sense for algorithms whose inputs are subject to slight amounts of noise in their low-order digits, which is typically the case if they are derived from measurements of real-world phenomena. The framework of smoothed analysis can be made more and more convincing by restricting the families of perturbations to more closely model the noise one would actually expect in a particular problem domain.

We expect that smoothed analysis can be used to both improve estimates on the gross order of the running times of algorithms, say from exponential to polynomial, as well as on finer aspects of the asymptotics, such as the degree of the polynomial or the constants out front. We hope that, through smoothed analysis, theorists may find ways to appreciate heuristics they may have previously rejected. Moreover, we hope that smoothed analysis may inspire the design of successful algorithms that might have been rejected for having poor worst-case complexity.

## 1.2 Papers on Smoothed Analysis

**“Smoothed Analysis of Algorithms: Why The Simplex Algorithm Usually Takes Polynomial Time,”** *Journal of the ACM*, Vol 51 (3), pp. 385 - 463, 2004. With S.-H. Teng.

We prove that the smoothed complexity of a two-phase shadow-vertex simplex method is polynomial in the dimensions of its input and the standard deviation of Gaussian perturbations. We note that an examination of lower bounds on most pivot rules for the simplex method reveals that one could not hope for a much better analysis: none of these rules can have smoothed complexity both polynomial in the dimension and sub-polynomial in the standard deviation. The shadow-vertex pivot rule is the same one analyzed by Borgwardt in his average-case analysis of the simplex method, and similar to that used in the other average-case analyses.

**“Smoothed Analysis of the Renegar’s Condition Number for Linear Programming,”** Available at <http://arxiv.org/abs/cs.DS/0302011>. Submitted to *Mathematical Programming, Series A*. With S.-H. Teng and J. Dunagan.

For every linear program, we show that the slight random relative perturbation of that linear program has small condition number with high probability. A consequence of our analysis is that the smoothed value of the logarithm of the condition number of a linear program is  $O(\log nd/\sigma)$ . Since the condition number bounds the running time of many algorithms for linear programming, this may help explain their observed fast convergence.

**“Smoothed analysis of termination of linear programming algorithms,”** *Mathematical Programming, Series B*, Vol 97, pp. 375-404, 2003. With S.-H. Teng.

We perform a smoothed analysis of a termination phase for linear programming algorithms. That is, given a perturbed linear program, we bound how close one needs to be to the optimal solution before one can find it exactly. By combining this analysis with the smoothed analysis of Renegar’s condition number by Dunagan, Spielman and Teng (above) we show that the smoothed complexity of interior-point algorithms for linear programming is  $O(m^3 \log(m/\sigma))$ . In contrast, the best known bound on the worst-case complexity of linear programming is  $O(m^3 L)$ , where  $L$  could be as large as  $m$ .

**“Smoothed Analysis of the Condition Numbers and Growth Factors of Matrices,”** Available at <http://arxiv.org/cs.NA/0310022> Submitted to *SIAM Journal on Matrix Analysis and Applications*. With A. Sankar and S.-H. Teng.

Let  $A$  be the slight random relative perturbation of an arbitrary matrix. We prove that it is unlikely that  $A$  has large condition number. Using this result, we prove it is not too likely that  $A$  has large growth factor under Gaussian elimination without pivoting. By combining these results, we bound the smoothed precision needed by Gaussian elimination without pivoting. Our results, when specialized to the average-case, improve upon the average-case analysis of Gaussian elimination without pivoting performed by Yeung and Chan (*SIAM J. Matrix Anal. Appl.*, 1997), and our analysis of the growth in  $U$  agrees with their experimental observations.

**“Smoothed Analysis of Partial Pivoting,”** In preparation. Mostly appears in the thesis of A. Sankar, “Smoothed Analysis of Gaussian Elimination”. With A. Sankar and S.-H. Teng.

Most implementations of Gaussian elimination for dense matrices employ the “partial-pivoting” heuristic. This heuristic is designed to prevent the appearance of large entries during the elimination, which could lead to inaccurate answers due to roundoff error. While the “complete-pivoting” heuristic is known to always solve this problem, partial pivoting is much preferred because it is twice as fast, and almost always seems to work in practice. This is in spite of the fact that it has exponentially poor worst-case behavior. Despite many efforts, partial-pivoting has eluded theoretical analysis. In this paper we will show that, when applied to the random perturbation of any matrix, it is highly unlikely that partial pivoting will generate large entries, and that this probability decreases quite sharply, with the probability of entries greater than  $x$  decreasing like  $2^{-\log^2 x}$ . In this way, we hope to provide a theoretical justification for the observation that one can get away with using this heuristic in practice.

## 2 Combinatorial Scientific Computing

Combinatorial Scientific Computing is the name given to the interdisciplinary field in which one applies combinatorial algorithms to problems in computational science and engineering. My research in this area has focused on the solution of systems of linear equations, mesh generation, and mesh partitioning. These problems arise in many scientific problems, and they all arise in the application of the finite element method.

The most significant of my works in this area, which is listed first, applies a new graph partitioning algorithm in the design of a linear system solver. The graph partitioning algorithm has many other applications. For example, it enables one to quickly find a cluster around a vertex of a massive graph, if one exists, *without looking at the whole graph*. In fact, it runs in time proportional to the size of the cluster it outputs.

### 2.1 Linear System Solvers

**“Nearly linear time algorithms for graph partitioning, graph sparsification, and solving linear systems”** Extended abstract in *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pp. 81–90, 2004. Full version available at <http://arxiv.org/abs/cs.DS/0310051> With S.-H. Teng.

We design preconditioners for symmetric diagonally-dominant linear systems that enable their solution in time nearly-linear in their number of non-zero entries. In particular, we solve systems with  $m$  non-zeros to accuracy  $\epsilon$  in time  $m \log^{O(1)} m \log(1/\epsilon)$ . If the systems are planar, we can reduce the time to  $O(m(\log m \log \log m)^2 \log(1/\epsilon))$ .

Our algorithm for constructing the preconditioners makes use of two novel algorithmic tools that could be important in their own right. The first is a nearly-linear time algorithm that takes as input any weighted graph  $A$  and outputs a weighted graph  $B$  with at most  $O(n \log^{O(1)} n)$  edges so that  $B$  is a good spectral approximation of  $A$ : the Laplacian matrices of these graphs,  $L_A$  and  $L_B$  satisfy

$$\forall x, \quad x^T L_B x \leq x^T L_A x \leq (1 + \alpha) x^T L_B x,$$

for any  $\alpha > 0$ . In turn, this graph  $B$  is obtained from a fast algorithm for the following approximation version of the graph partitioning problem: on input  $\phi$  and a graph  $G$ , output a cut of isoperimetric number at most  $\phi^{1/3} \log^{O(1)} n$  separating at least  $2/3$  as many nodes as the best cut of isoperimetric number  $\phi$ . That is, it attempts to find the most balanced cut of isoperimetric number close to  $\phi$ . This partitioning algorithm is in turn based on an algorithm that finds clusters in time proportional to their size.

For the latest experimental results, see my talk from CS&E 2005, <http://www-math.mit.edu/~spielman/TALKS/cse.ppt>.

**“Solving Sparse, Symmetric, Diagonally-Dominant Linear Systems in Time  $O(m^{1.31})$ ”** *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, 2003. Most recent version available at <http://arxiv.org/cs.DS/0310036>. With S.-H. Teng.

Extending the work of Vaidya, we present a fast algorithm for constructing provably good preconditioners for symmetric, diagonally-dominant linear systems. This was improved by the previously listed paper.

**“Lower-Stretch Spanning Trees”** To appear in *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, 2005. With M. Elkin, Y. Emek and S.-H. Teng. Available at <http://arxiv.org/abs/cs.DS/0411064>

Low-stretch spanning trees play a fundamental role in our linear system solvers. In this paper, we show how to construct in nearly-linear time a spanning tree in any graph with average stretch  $O(\log^2 n \log \log n)$ . This improves upon the previous best bound of  $2^{O(\sqrt{\log n \log \log n})}$  by Alon, Karp, Peleg and West (SIAM J. Comp. 1995), and immediately implies a similar improvement in the running time of our linear system solvers.

## 2.2 Mesh and Graph Partitioning

The problem of graph partitioning is to divide a graph into a few pieces of roughly equal size while cutting as few edges as possible. In our papers on graph partitioning, we have analyzed popular algorithms for graph partitioning, as well as introduced the first provably good algorithm for domain decomposition in planar graphs.

**“Spectral Partitioning Works: Planar Graphs and Finite-Element Meshes,”** *Proceedings of the 35th Annual IEEE Conference on Foundations of Computer Science*, pp. 96–105, 1996. With S.-H. Teng.

Spectral partitioning methods use an eigenvector of the Laplacian matrix of a graph to find a small separator in the graph. These methods are important components of many scientific numerical algorithms and have been demonstrated by experiment to work extremely well. In this paper, we show that spectral partitioning methods work well on bounded-degree planar graphs and well-shaped meshes—the classes of graphs to which they are usually applied. While naive spectral bisection does not necessarily work, we prove that spectral partitioning techniques can be used to produce separators whose ratio of edges cut to vertices removed is small. The heart of our analysis is an upper bound on the second-smallest eigenvalues of the Laplacian matrices of these graphs: we prove a bound of  $O(1/n)$  for bounded-degree planar graphs and  $O(1/n^{2/d})$  for well-shaped  $d$ -dimensional meshes.

**“Min-Max Boundary Domain Decomposition,”** *Theoretical Computer Science*, pp. 253-266, Volume 261, Issue 2, 2001. With M. Kiwi and S.-H. Teng.

Motivated by domain decomposition, one of the most effective and popular parallel numerical techniques, we study the following min-max boundary multi-way partitioning problem: Given a graph  $G$  and a positive integer  $k$ , we would like to divide  $G$  into  $k$  subgraphs  $G_1, \dots, G_k$  (by removing edges) such that (i)  $|G_i| = \Theta(n/k)$  for all  $i$ ; and (ii) the maximum boundary size of any subgraph (the set of edges connecting it with other subgraphs) is minimized. We present an  $O(n \log k)$  time algorithm that, for every well-shaped  $d$ -dimensional mesh  $G$ , produces such a decomposition in which each subgraph has at most  $O(n/k)^{1-1/d}$  boundary edges.

**“Disk Packings and Planar Separators,”** *Proceedings of the 12th Annual ACM Symposium on Computational Geometry*, pp. 349-358, 1996. With S.-H. Teng.

We prove that the geometric separator algorithm of Miller, Teng, Thurston, and Vavasis finds a  $3/4$ -separator of size  $1.84\sqrt{n}$  in every  $n$  node planar graph, thereby demonstrating that this algorithm is the equal of the combinatorial algorithm of Alon, Seymour and Thomas.

## 2.3 Mesh Generation

**“Parallel Delaunay Refinement: Algorithms and Analyses”** *Proceedings of the 11th International Meshing Roundtable*, pp. 205-217, 2002. With S.-H. Teng and A. Ungor. Submitted to the *International Journal of Computational Geometry & Applications*.

We analyze the complexity of natural parallelizations of Delaunay refinement methods for mesh generation. The parallelizations employ a simple strategy: at each iteration, they choose a set of “independent” points to insert into the domain, and then update the Delaunay triangulation. We show that such a set of independent points can be constructed efficiently in parallel and that the number of iterations needed is  $O(\log^2(L/s))$ , where  $L$  is the diameter of the domain, and  $s$  is the smallest edge in the output mesh. In addition, we show that the insertion of each independent set of points can be realized sequentially by Ruppert’s method in two dimensions and Shewchuk’s in three dimensions. These are the first provably  $\text{polylog}(L/s)$  parallel time Delaunay meshing algorithms that generate well-shaped meshes of size optimal to within a constant.

**“Parallel Delaunay Refinement with Off-Centers”** *Proceedings of the 10th International EURO-PAR Conference, Lecture Notes in Computer Science 3149*, pp. 812–819, 2004. With S.-H. Teng and A. Ungor.

We improve upon the previous paper by inserting off-centers instead of centers of triangles. We prove that Delaunay refinement with off-centers takes only  $O(\log(L/s))$  parallel iterations, where  $L$  is the diameter of the domain, and  $s$  is the smallest edge in the initial triangulation.

## 3 Error-Correcting Codes

Error-correcting codes are the means by which we compensate for interference in communication. The goals of my research in error-correcting codes have been to develop codes that are quickly encodable and decodable, and which allow communication at rates approaching the capacity of the communication channel. The most famous of these papers, “Improved Low-Density Parity-Check Codes Using Irregular Graphs” shared the the 2002 Information Theory Society Paper Award. It demonstrated that irregular low-density parity-check codes could perform much better than their more common regular cousins on the additive white Gaussian noise channel. This extended the result of our paper “Efficient Erasure Correcting Codes”, which introduced irregular low-density parity-check codes, and proved that they could approach the capacity of the binary erasure channel—the natural model for packet loss in internet traffic. Irregular low-density parity-check codes have had many applications, including the recent DVB-S2 (Digital Video Broadcasting—Satellite v2) standard.

### 3.1 Codes Approaching Capacity

Shannon proved that every communication channel has a maximum rate, called capacity, at which one can communicate reliably over the channel. By reliable communication, Shannon meant that with high probability, one could communicate without error. In our work on constructing error-correcting codes that approach capacity, we’ve analyzed the behavior of the belief propagation decoding heuristic and its discretizations on low-density parity-check codes. Our analyses enabled us to design particular distributions of graphs that induce codes on which the heuristic has superior performance.

**“Efficient Erasure Correcting Codes,”** *IEEE Transactions on Information Theory*, 47(2), pp. 569-584, Feb. 2001. With M. G. Luby, M. Mitzenmacher and M. A. Shokrollahi.

In this paper, we developed the Tornado Codes, a family of codes that approach the capacity of the erasure channel and that can be encoded and decoded in linear time. These codes have exceedingly fast software implementations and should be useful for compensating for packet loss in internet traffic.

We derive our codes by analyzing the performance of a discretization of Gallager’s decoding algorithm for low-density parity-check codes that is suitable for erasures. We then design families of irregular graphs that induce low-density parity-check codes on which this algorithm has superior performance. While these codes are quadratic-time encodable, we derive linear-time encodable codes by combining them in a novel recurrence that does not damage their rate or decoding threshold.

**“Improved Low-Density Parity-Check Codes Using Irregular Graphs,”** *IEEE Transactions on Information Theory*, 47(2), pp. 585-598, Feb. 2001. With M. G. Luby, M. Mitzenmacher and M. A. Shokrollahi.

In this work, we extend the techniques of our “Efficient Erasure-Correcting Codes” paper to analyze the performance of low-density parity-check codes correcting errors.

Our first result is the experimental demonstration that, under belief propagation decoding, low-density parity-check codes derived from certain irregular graphs have performance superior to the traditional regular low-density parity-check codes. In some cases, our results come very close to reported results for Turbo Codes, suggesting that variations of irregular low-density parity-check codes may be able to match or beat Turbo Code performance.

We also study hard decision decoding, a discrete weakening of belief propagation. For hard decision decoding, we demonstrate how to prove performance guarantees for irregular low-density parity-check codes, extending the original work of Gallager. We also provide efficient methods for finding good irregular graphs that induce codes that perform well under the hard decision decoding algorithms.

Richardson, Shokrollahi and Urbanke (IEEE TIT '01) improved on our results by introducing the Density Evolution framework for analyzing the behavior of irregular low-density parity-check codes. Together, our papers shared the 2002 Information Theory Society Paper Award.

**“Finding Good LDPC Codes,”** in Proceedings of the *Thirty-Sixth Allerton Conference on Communications, Control, and Computing*, pp. 211–219, 1998.

In this work, we heuristically extended the techniques of the previous paper. Using these heuristics, we constructed good distributions of irregular low-density parity-check (LDPC) codes. Experimental results demonstrated that these were the first binary codes to have lower bit error rates than Turbo Codes at very low signal-to-noise ratios.

**“The Minimum Distance of Turbo-Like Codes,”** submitted to *IEEE Transactions on Information Theory*, 2003.

Available at <http://math.mit.edu/~spielman/Research/mindist.html>.

With L. Bazzi and M. Mahdian.

The low-complexity and near-capacity performance of Turbo codes has led to a revolution in coding theory. The most famous casualty of the revolution has been the idea that good codes should have high minimum distance: the most useful Turbo codes have been observed to have low minimum distance. In this work, we provide general conditions under which many constructions of turbo-like codes, including families of serially-concatenated Turbo-like codes, and Repeat-Accumulate (RA) codes, must be asymptotically bad. We also present a simple family of depth-3 serially concatenated Turbo-like codes that are asymptotically good.

### 3.2 Linear-Time Codes

The papers “Expander Codes” and “Linear-Time Encodable and Decodable Error-Correcting Codes” formed the bulk of my thesis, which won the 1995 ACM Doctoral Dissertation Award. The latter of these papers also won the Best Student Paper Award at the 27th ACM STOC conference.

“Expander Codes,” *IEEE Transactions on Information Theory*, 42(6), pp. 1710-1722, Nov. 1996. With M. Sipser.

We present a new class of asymptotically good, linear error-correcting codes based upon expander graphs. They form a sub-class of Gallager’s low-density parity-check codes. These codes have linear-time sequential decoding algorithms that can correct a constant fraction of worst-case errors. The decoding algorithms have parallel variants that take logarithmic-time and use a linear number of processors. We present both randomized and explicit constructions for some of these codes.

**“Linear-Time Encodable and Decodable Error-Correcting Codes,”** *IEEE Transactions on Information Theory*, 42(6), pp. 1723-1731, Nov. 1996.

We present the first linear-time encodable codes with linear-time decoding algorithms that can recover from a constant fraction of error. The encoding and decoding algorithms can also be implemented in logarithmic parallel time with a linear number of processors. The construction combines low-density generator matrix variants of the Expander Codes in a recursion analogous to Pippenger’s construction of superconcentrators. We present both randomized and explicit constructions of these codes.