Research Statement

Sophie Huiberts

February 23, 2026

1 Introduction

My work focuses on linear programming (LP), a type of mathematical optimization problem. LP is a fundamental method with broad application in academia and industry, including for modern energy systems, logistics, manufacturing and digital infrastructure. For that reason, this subject is part of many graduate and undergraduate curricula in mathematics, economics and computer science. In LP, we are given linear (in)equality constraints in a number of real variables, along with a linear objective function, and we are tasked with finding values for the variables such that all constraints are satisfied and the objective value of the solution is optimal. One example of a linear program is

$$\begin{aligned} &\text{maximize } c^\top x \\ &\text{subject to } Ax \leq b, \end{aligned} \tag{1}$$

where $c \in \mathbb{R}^d, b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times d}$ are given. The task is to compute a vector $x \in \mathbb{R}^d$ which satisfies the system of inequality constraints $Ax \leq b$ (feasibility) and, among those satisfying vectors, find one for which the objective value $c^\top x$ is maximal (optimality).

To solve linear programs in practice with good accuracy, two classes of algorithms are used: interior-point methods and simplex methods. Interior-point methods are often more efficient when solving linear programs from scratch. Simplex methods are much more efficient when solving linear programs from a *warm start*. Warm starting benefits apply for most algorithmic technology built on top of LP, including state-of-the-art approaches to solving mixed-integer linear programming (MILP) problems. Because of these complementary functions, all LP software includes both types of algorithms.

My own research is primarily focused on simplex methods. These algorithms operate through a sequence of *pivot steps*, each one consisting of a few solves of systems of linear equations. Simplex methods are reported to be highly efficient, with almost eight decades of practical experience indicating that the required number of pivot steps is linear in the size of the input. The cause of this efficient behavior is unknown. My work uses methods from theoretical computer science to better understand this phenomenon. By developing innovative analytical techniques, I aim to produce actionable insight for algorithm designers and implementers.

Past Work Besides studying the simplex method, I have done theoretical work on many other facets of modern MILP software. This includes interior-point methods [DHNV23], branch-and-bound tree search [BDHK24], integrality gaps [BDHT22], and cutting-plane methods [DHHW23].

2 Theory

On a high level, the simplex method works as follows. The algorithm possesses a *feasible basis* of the linear program. Each pivot step moves the algorithm to an adjacent feasible basis. These pivot steps improve the basis' objective value while maintaining feasibility. When a provably optimal basis is found, the algorithm terminates.

In theory, a geometric interpretation is made. The set of all vectors satisfying the constraints are considered as a convex polyhedron called the *feasible set*. Any feasible basis then maps onto a vertex of this set, and every vertex has at least one associated feasible basis. In this interpretation, the simplex method traces a path consisting of pairwise adjacent vertices. This makes it possible to analyze the simplex method using geometric principles.

Since the 1970's, many different variants of the simplex method have been mathematically proven to require exponentially many pivot steps under the *worst-case complexity* model. There is an apparent contradiction between the observed near-linear number of pivot steps and the proven exponentially large number of pivot steps. In order to develop a better theoretical understanding of the number of pivot steps, theory must go beyond the worst-case model. For decades, the leading analytical frameworks in this spirit were variants of average-case analysis. In these models, assumptions are made that the data A, b, c in (1) are drawn from particular probability distributions. The most advanced average-case analysis model is known as *smoothed analysis* and was introduced by Spielman and Teng [ST04]. In smoothed analysis, we assume that the constraint data A and b are subject to small independent Gaussian random noise. One can then prove that the expected running time is bounded from above by a function of the standard deviation. This model was, up until recently, the state of the art.

Past Work I have done extensive work on the simplex method in the past. My research has covered all major analysis models: worst case analysis [BDHM26], average case analysis [BDGHL26], smoothed analysis [DH20; DH21; HLZ23; BH25], and by-the-book analysis [BBHK26]. In the smoothed analysis model I proved the first non-trivial lower bounds and, together with my PhD student Eleon Bach, I gave an algorithm whose running time provably matches this lower bound in the noise parameter. By-the-book analysis is the successor to smoothed analysis. More information about this framework can be found in the next section.

3 Computation

Implementations of the simplex method do not obey the strict theoretical separation between feasible and infeasible vectors. Partially, this is the result of unavoidable complications that occur when implementing real-valued algorithms using floating-point arithmetic. But, in the case of high-performance implementations of the simplex method, there is more to it.

Compared to any theoretical description, a number of deliberate changes are made during implementation. These changes are similar across different software packages and mostly date back to scientific literature from between 1960-1980. These implementation techniques deliberately disregard the strict definition of feasibility, invalidating the geometric interpretation of the algorithm. In exchange, the running time of the simplex method is greatly reduced. These implementation details are varied and involve all parts of the code, but are most prominent in the pivot selection (e.g., perturbations, Harris' ratio test, scaling) and determination of the current solution vector (e.g., shifting, basis refactorization, stalling detection). Although these techniques have been successfully used in practice for decades, they had never been theoretically studied. I interviewed developers of all leading MILP software packages and found that they strongly disagreed with each other about how these techniques caused their associated speedups.

I believe that these implementation techniques are of theoretical interest, and that at least some number of them can provably speed up the simplex method. This, then, constitutes the ground on which I am building a modern theoretical analysis of simplex methods. This theory will use these unexplored implementation techniques to formulate new mathematical assumptions. Additional assumptions can be formulated on the basis of observations on the context in which simplex methods are used. Using these assumptions, it becomes possible to prove new running time guarantees. By being firmly rooted in observation and implementation, such theoretical work can inform the developers of MILP software during future development and evaluation. This work has already started. The initial phase is being supported by my ANR grant "Towards Testable Theories of Linear Programming" (2024-2028).

The pilot study [BBHK26] for this research line is scheduled to appear in STOC 2026, and is the first theoretical paper giving provable running time guarantees from practical observations. Out of the large

spectrum of possible implementation techniques and contextual assumptions, three types were used in the pilot study.

1. Assumptions on scaling and choice of units by the LP user, which are based on instructions in MILP software user manuals and measurements of benchmark MILP data.

2. Assumptions on the existence of *feasibility tolerances*, which are taken from the technical specifications in the same user manuals.

3. Finally, an algorithmic technique known as *bound perturbation*, which is implemented in all open-source and commercial codes.

Under these three types of assumptions, my collaborators and I showed that the number of pivot steps of a simplex method is provably bounded from above. This is a theoretical paper whose conclusion supports the observation that these techniques speed up the simplex method.

Developers of MILP solvers are already exploring how our analysis of the bound perturbations can speed up their solver.

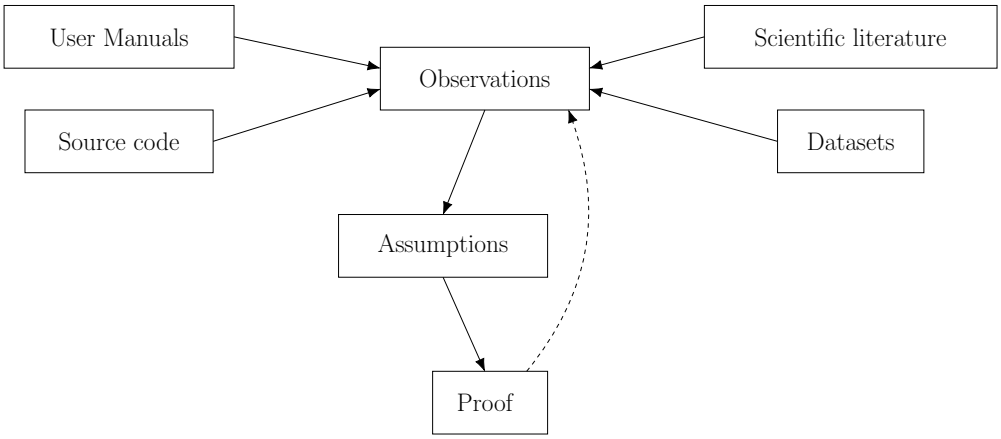

Figure 1: Schematic depiction of by-the-book analysis. Observations are used to formulate assumptions, which are then used to prove theorems. After the theorem is proven, it may inform additional observations.

This framework, where we used source code, user manuals and data sets to formulate mathematical assumptions for algorithm analysis, is new. See Figure 1 for a schematic depiction of the framework's process. We call our framework "by-the-book analysis". It is premised on the idea that the theoretician is allowed to make any number of additional assumptions they need, so long as these assumptions are given strong justifications. Importantly, all these justifications must be derived from real-world observation.

Future work Now that the viability of by-the-book analysis has been demonstrated, there remains much theoretical work to be done. As mentioned above, most implementation techniques and phenomena have not yet been studied from a theoretical angle, despite practitioners having long observed their positive effects on running times. The most theoretically promising techniques and phenomena include automatic re-scaling, Harris' ratio test, (hyper)sparsity, basis re-factorization, stalling detection and primal-dual infeasibility repair. I plan to lead studies into each of these, using all available analysis models including worst-case, average-case and by-the-book analysis.

One such project is underway, where we expect to prove that a combination of scaling, perturbation and primal-dual infeasibility repair can yield a simplex method with polynomial worst-case complexity.

The long-term goal is to theoretically analyze the effect of Harris' ratio test. Out of all implementation techniques mentioned, this variant on the traditional ratio test is most broadly agreed to improve performance

under all circumstances. While all developers agree that Harris' test is necessary for any high-performance simplex method, there is no agreement on why it speeds up the simplex method.

The future scientific study of the practical behavior of the simplex method will require both theoretical and experimental work. Computational experiments are necessary to evaluate the quality of theoretical results. In turn, computational experiments benefit from theory when determining what to measure. However, computational simplex method research has seen little academic activity in the past decades. This is both a detriment to scientific research and has contributed to a global talent shortage in industry. Now that theory is opening up new avenues for experiments, I plan to more actively work towards a revival of computational simplex method research. In the longer term, I see potential to build up more computational research infrastructure, including a modern research-ready simplex method implementation and a public benchmark dataset.

Besides the simplex method, I intend to study the by-the-book analysis framework in its own right. Using this framework might provide new insights into the performance of other algorithms, including many algorithms which have been previously subjected to smoothed analysis. Moreover, applying our framework to other algorithms and algorithmic problems will help to identify the framework's strengths and weaknesses. The first step to build these collaborations will be the Dagstuhl workshop I organize in July 2026 on analysis of algorithms beyond the worst case.

4 Application

I believe that the best theoretical results exist in context, and so I have taken an interest in the application context that (mixed integer) linear programming exists within. Over the past 1.5 years, I have started laying the groundwork for a large project within this space. The project is to find out which government organizations make use of MILP, and to then obtain their MILP instances through Freedom Of Information (FOI) requests. This would allow the creation of a new large dataset of MILP instances.

Currently, publicly available MILP instances are all shared voluntarily. Due to most MILPs containing sensitive industrial operations information, few MILP real-world instances have been made available in the public dataset MIPLIB [Gle+21]. This makes it hard for academics to achieve the level of performance testing that commercial businesses have access to, and has lead to a situation where we are dependent on commercial solver developers in order to know which algorithms are effective and which are not. A new high-quality collection of MILPs would be of great value to the computational MILP research community.

FOI legislation exists within most democratic countries, where citizens and non-citizens alike may request access to existing government documents. Unless specific grounds for refusal are met, the documents must be provided to the requester. These laws are primarily a tool used by journalists and activists, but I aim to explore their function for computer science research. Due to the binding nature of these laws, I expect FOI to be a successful mechanism to obtain more MILP instances.

My first FOI request was to the Dutch Ministry of Defense. Per my request, they informed me of what they use MILP for, which software package they use, how they chose that particular package, and how much they pay for the software [Min24]. In this FOI request I did not yet ask for MILP instances. In the Netherlands, computer code and log files are subject to FOI requests. Thus I believe that it is likely that requesting MILP instances from government agencies will succeed to a meaningful degree, especially since MILP is used both in daily operations (e.g., [Min24]) and in policy making (e.g., [WH15; KBSGEHM25]) in many different agencies.

If successful, the outcome of this project would be a large set of MILP instances with trusted provenance. These instances would be known to have been used for real-world practical problems. Some fraction of instances might even be part of a time series, solving a recurring optimization problem that happens periodically with small variations. Current computational research into repeated solving of similar instances use synthetic variations, because organic time series are not available [BBGBDMPT23]. I hope that the FOI method can fill this gap.

Initially I plan to execute my FOI requests to Dutch governmental bodies. This is mainly based on my personal familiarity with the country making it easier to write effective FOI requests and get early results. After developing a replicable approach in the Netherlands, I want to develop this project into a large multinational collaboration.

References

[BBGBDMPT23] Suresh Bolusani, Mathieu Besançon, Ambros Gleixner, Timo Berthold, Claudia D'Ambrosio, Gonzalo Muñoz, Joseph Paat, and Dimitri Thomopulos. *The MIP Workshop 2023 Computational Competition on Reoptimization*. 2023. arXiv: 2311.14834.

[BBHK26] Eleon Bach, Alexander E. Black, Sophie Huiberts, and Sean Kafer. *Beyond Smoothed Analysis: Analyzing the Simplex Method by the Book*. To appear in STOC. 2026. arXiv: 2510.21613.

[BDGHL26] Gilles Bonnet, Daniel Dadush, Uri Grupel, Sophie Huiberts, and Galyna Livshyts. "Asymptotic Bounds on the Combinatorial Diameter of Random Polytopes". In: *Discrete & Computational Geometry* (Feb. 2026). Preliminary version in SoCG 2022. ISSN: 1432-0444. DOI: 10.1007/s00454-025-00814-6. arXiv: 2112.13027.

[BDHK24] Sander Borst, Daniel Dadush, Sophie Huiberts, and Danish Kashaev. "A nearly optimal randomized algorithm for explorable heap selection". In: *Mathematical Programming* 210.1–2 (Nov. 2024). Preliminary version in IPCO 2023, pp. 75–96. ISSN: 1436-4646. DOI: 10.1007/s10107-024-02145-5. arXiv: 2210.05982.

[BDHM26] Eleon Bach, Yann Disser, Sophie Huiberts, and Nils Mosis. "An Unconditional Lower Bound for the Active-Set Method in Convex Quadratic Maximization". In: *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Jan. 2026, pp. 314–327. ISBN: 9781611978971. DOI: 10.1137/1.9781611978971.14. arXiv: 2507.16648.

[BDHT22] Sander Borst, Daniel Dadush, Sophie Huiberts, and Samarth Tiwari. "On the integrality gap of binary integer programs with Gaussian data". In: *Mathematical Programming* (2022). Preliminary version in IPCO 2021. DOI: 10.1007/s10107-022-01828-1. arXiv: 2012.08346.

[BH25] Eleon Bach and Sophie Huiberts. "Optimal Smoothed Analysis of the Simplex Method". In: *Proceedings of the 66th Annual Symposium on Foundations of Computer Science (FOCS)*. 2025. arXiv: 2504.04197.

[DH20] Daniel Dadush and Sophie Huiberts. "A Friendly Smoothed Analysis of the Simplex Method". In: *SIAM Journal on Computing* 49.5 (Jan. 2020), STOC18–449–499. DOI: 10.1137/18m1197205. arXiv: 1711.05667.

[DH21] Daniel Dadush and Sophie Huiberts. "Smoothed Analysis of the Simplex Method". In: *Beyond the Worst-Case Analysis of Algorithms*. Ed. by Tim Roughgarden. Cambridge University Press, 2021, pp. 309–333. DOI: 10.1017/9781108637435.019.

[DHHW23] Daniel Dadush, Christopher Hojny, Sophie Huiberts, and Stefan Weltge. "A simple method for convex optimization in the oracle model". In: *Mathematical Programming* (Aug. 2023). Preliminary version in IPCO 2022. DOI: 10.1007/s10107-023-02005-8. arXiv: 2011.08557.

[DHNV23] Daniel Dadush, Sophie Huiberts, Bento Natura, and László A. Végh. "A scaling-invariant algorithm for linear programming whose running time depends only on the constraint matrix". In: *Mathematical Programming* (Apr. 2023). Preliminary version in ACM STOC 2020. DOI: 10.1007/s10107-023-01956-2. arXiv: 1912.06252.

[Gle+21] Ambros Gleixner, Gregor Hendel, Gerald Gamrath, Tobias Achterberg, Michael Bastubbe, Timo Berthold, Philipp M. Christophel, Kati Jarck, Thorsten Koch, Jeff Linderoth, Marco Lübbecke, Hans D. Mittelmann, Derya Ozyurt, Ted K. Ralphs, Domenico Salvagnin, and Yuji Shinano. "MIPLIB 2017: Data-Driven Compilation of the 6th Mixed-Integer Programming Library". In: *Mathematical Programming Computation* (2021). DOI: 10.1007/s12532-020-00194-3.

[HLZ23] Sophie Huiberts, Yin Tat Lee, and Xinzhi Zhang. "Upper and Lower Bounds on the Smoothed Complexity of the Simplex Method". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*. June 2023. DOI: 10.1145/3564246.3585124. arXiv: 2211.11860.

[KBSGEHM25] Aart Kooiman, Rebeka Beres, Martin Scheepers, Noelia Martín Gregorio, Leonard Eblé, Rudi Hakvoort, and Jos Meeuwsen. *Systeemkostenanalyse kernenergie*. 2025. URL: https://www.tweedekamer.nl/kamerstukken/brieven_regering/detail?id=2025D44440&did=2025D44440.

[Min24] Ministerie van Defensie. *Woo-besluit inclusief lijst van documenten en bijlagen over aanbesteding van en besluitvorming over aanschaf van (mixed integer) lineaire software*. 2024. URL: https://www.rijksoverheid.nl/ministeries/ministerie-van-defensie/documenten/woo-besluiten/2024/11/07/woo-besluit-inclusief-lijst-van-documenten-en-bijlagen-over-aanbesteding-van-en-besluitvorming-over-aanschaf-van-mixed-integer-lineaire-software.

[ST04] Daniel A Spielman and Shang-Hua Teng. "Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time". In: *Journal of the ACM (JACM)* 51.3 (2004), pp. 385–463. DOI: 10.1145/990308.990310.

[WH15] C.P.A. van Wagenberg and R. Hoste. *Effecten van een verbod op het gebruik van genetisch gemodificeerde soja als veevoedergrondstof*. 2015. URL: https://www.tweedekamer.nl/kamerstukken/brieven_regering/detail?id=2015D49986&did=2015D49986.