Heuristic Optimality Checks for Noise-Aware Sparse Recovery by ℓ_1-Minimization

Christoph Brauer and Dirk A. Lorenz
Institute for Analysis and Algebra, TU Braunschweig
Pockelsstraße 14, D-38106 Braunschweig, Germany

Andreas M. Tillmann
Research Group Optimization, TU Darmstadt
Dolivostr. 15, D-64293 Darmstadt, Germany

***Abstract*—During recent years, ℓ_1-norm minimization has become a standard approach for the task of finding sparse exact or approximate solutions to underdetermined linear equation systems, with a broad range of specialized solvers to choose from. In this work, we present an algorithmic tool we call *Heuristic Optimality Check* (HOC) which attempts to construct an optimal point directly, given a support estimate and sign pattern. Our general scheme can easily be adapted to a variety of ℓ_1-minimization problems prominent in sparse recovery and compressed sensing. We provide numerical experiments showing that our HOC techniques can indeed often improve the solution speed and accuracy of existing ℓ_1-minimization methods in the presence of sparse solutions.**

I. Introduction & Motivation

Some of the most popular approaches for sparse reconstruction from incomplete linear measurements are included in the general class of ℓ_1-minimization problems

$$\min \|\boldsymbol{x}\|_1 \quad \text{s.t.} \quad \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\| \leq \delta, \tag{P$_1$}$$

where $\delta \geq 0$ is an estimate of measurement noise, and $\|\cdot\|$ is some norm (chosen with respect to the noise model). By now, it is well-known that under many different conditions, solving (P$_1$) also solves the generally NP-hard problem of finding the sparsest-possible solution under the given constraints, or at least leads to desirable error bounds. An overview of such results, and also of several specialized solution methods for ℓ_1-problems, can be found, e.g., in [1].

To further exploit solution sparsity, a *Heuristic Optimality Check (HOC)* for the noise-free (basis pursuit) problem was introduced in [2] (see also [3]). Very briefly, given a (not necessarily primal feasible) vector $\boldsymbol{x}$ with approximate support S, the HOC scheme tries to construct an optimal primal-dual pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ such that $\hat{\boldsymbol{x}}$ has support $\hat{S} \subseteq S$. Via approximate calculations and post-verification instead of forced fulfillment of certain constraints to speed up the incurred computations, executing HOC at certain iteration intervals of a basis pursuit solver was then demonstrated to often allow for "jumping" to the optimal sparse solution long before the respective ℓ_1-solver terminates on its own, thereby improving both solution speed and accuracy.

II. Heuristic Optimality Checks for ℓ_1-Minimization

The foundation of our HOC routine is a well-known primal-dual characterization of optimality: For $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ with $\operatorname{rank}(\boldsymbol{A}) = m \leq n$, strong duality holds for (P$_1$) and its dual problem

$$\max -\boldsymbol{b}^\top\boldsymbol{y} - \delta\|\boldsymbol{y}\|_* \quad \text{s.t.} \quad \|\boldsymbol{A}^\top\boldsymbol{y}\|_\infty \leq 1, \tag{D$_1$}$$

where $\|\cdot\|_*$ denotes the dual norm to $\|\cdot\|$.

Proposition 1: If $\|\boldsymbol{b}\| > \delta$, then a tuple $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ is a primal-dual optimal pair for (P$_1$) and (D$_1$) if and only if $-\boldsymbol{A}^\top\hat{\boldsymbol{y}} \in \partial\|\hat{\boldsymbol{x}}\|_1$ and $\boldsymbol{A}\hat{\boldsymbol{x}} - \boldsymbol{b} \in \delta\partial\|\hat{\boldsymbol{y}}\|_*$, or equivalently, $\|\boldsymbol{A}\hat{\boldsymbol{x}}-\boldsymbol{b}\| = \delta$, $\|\boldsymbol{A}^\top\hat{\boldsymbol{y}}\|_\infty = 1$ and $\|\hat{\boldsymbol{x}}\|_1 + \delta\|\hat{\boldsymbol{y}}\|_* + \boldsymbol{b}^\top\hat{\boldsymbol{y}} = 0$.
This can be shown using Lagrange duality (cf., e.g., [4]).

The HOC scheme then proceeds as shown in Algorithm 1. It can

Algorithm 1 Heuristic Optimality Check (HOC) for (P$_1$).

Input: $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\delta \geq 0$, $\boldsymbol{x} \in \mathbb{R}^n$ with approx. supp. S
- $\hat{\boldsymbol{y}} \leftarrow$ approximate solution to $-\boldsymbol{A}_S^\top\boldsymbol{w} = \operatorname{sign}(\boldsymbol{x}_S)$
- **if** $\|\boldsymbol{A}^\top\hat{\boldsymbol{y}}\|_\infty \approx 1$ **then**
  - $\hat{\boldsymbol{x}}_S \leftarrow$ approx. solution to $\boldsymbol{A}_S\boldsymbol{z} \in \boldsymbol{b} + \delta\partial\|\hat{\boldsymbol{y}}\|_*$, $\hat{\boldsymbol{x}}_{S^c} \leftarrow 0$
  - **if** $\|\boldsymbol{A}\hat{\boldsymbol{x}} - \boldsymbol{b}\| \approx \delta$ **then**
    - **if** $(\|\hat{\boldsymbol{x}}\|_1 + \delta\|\hat{\boldsymbol{y}}\|_* + \boldsymbol{b}^\top\hat{\boldsymbol{y}})/\|\hat{\boldsymbol{x}}\|_1 \approx 0$ **then**
      - **return** (approximately) optimal primal-dual pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$

easily be verified that, were we to carry out all computations and comparisons exactly, a pair $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ returned by HOC would indeed be optimal. Although this needs not hold if the exactness requirements are relaxed, we observed that HOC hardly ever makes false-positive optimality claims. Performed repeatedly during a run of some ℓ_1-solver, the hope is that HOC can thus lead to early termination.

III. Numerical Experiments

It is important to note that HOC is independent of the solution algorithm and can, consequently, be intregrated into virtually any ℓ_1-solver implementation. We did so for several problem classes (with the most common ℓ_p-norm-constraints with $p \in \{1, 2, \infty\}$) and solvers and empirically evaluated the impact of HOC on various test instances (with known sparse optima). The results clearly indicate that using HOC indeed often improves the solution speed and accuracy. Space limitations disallow us to go into much detail here; we outline some results in the following.

Fig. 1 illustrates the HOC concept, demonstrating a run of the incremental subgradient method from [5] applied to (P$_1$) with ℓ_∞-constraints without and with HOC – one can see that by employing HOC, the iteration number can be reduced to less than half (the runtime improvement here was roughly 62%).

Even for very fast solvers such as SPGL1 (cf. [6]), HOC can achieve improvements: Fig. 2 shows results of extensive experiments, in terms of distance of the computed points $\bar{\boldsymbol{x}}$ to the known optimal points $\boldsymbol{x}^*$ versus running time without and with HOC. Clearly, HOC often leads to both early termination and better accuracy; in case HOC is not successful, the overhead introduced by its integration is negligibly small.

Similar, or even better, results can be achieved for interior-point methods: For ℓ_1-Magic [7] (applied to $\min\{\|\boldsymbol{x}\|_1 : \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2 \leq \delta\}$), average speed-ups between 17.5% and 25% can be achieved by running HOC in *every* iteration. (The overhead introduced by HOC is again very small in unsuccessful cases, since interior-point methods typically perform only a few iterations but of much larger iteration complexity than in first-order methods.) The HOC scheme for ℓ_2-norm constraint ℓ_1-minimization can also be adapted to the related ℓ_1-regularized least-squares problem $\min \lambda\|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$; Fig. 3 shows the positive effect of this HOC variant employed in

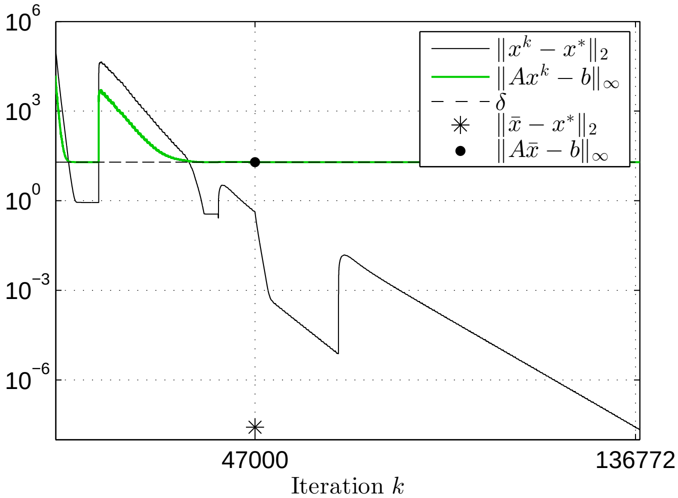

Fig. 1. Example for HOC efficiency in the incremental subgradient method of [5], applied to (P_1) with ℓ_∞-constraints; $\delta \approx 20$, $m = 512$, $n = 1024$, $\|\boldsymbol{x}^*\|_0 = 28$. Solution $\bar{\boldsymbol{x}}$ produced executing HOC every 1000 iterations.

each iteration of the SolveBP/PDCO solver from SparseLab [8]. It is worth mentioning that the improvement persists, even if slightly less prominent, when using a more typical, smaller value of the regularization parameter λ on the same set of test instances (e.g., average speed-up is around 1.52% when $\lambda = 0.01$).

IV. Conclusion

We provide a generalized HOC scheme for the problem class (P_1). Empirical results for several problem types (in particular, those with ℓ_p-norm constraints for $p \in \{1, 2, \infty\}$) indicate that the HOC idea also works well in this noise-aware setting. Moreover, our theoretical and algorithmical tools can also be exploited to generate test instances with known (sparse) optima for (P_1), and may be extended to ℓ_1-regularized least-squares as well as ℓ_1-analysis models that minimize $\|\boldsymbol{Bx}\|_1$ with $\boldsymbol{B} \in \mathbb{R}^{q \times n}$ $(q \geq n)$ instead of just $\|\boldsymbol{x}\|_1$.

References

[1] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*, ser. Applied and Harmonic Analysis. New York, NY, USA: Springer, 2013.

[2] D. A. Lorenz, M. E. Pfetsch, and A. M. Tillmann, “Solving Basis Pursuit: Heuristic Optimality Check and Solver Comparison,” *ACM T. Math. Software*, vol. 41, no. 2, article no. 8, 2015.

[3] A. M. Tillmann, *Computational Aspects of Compressed Sensing*. Doctoral Dissertation, TU Darmstadt, 2013.

[4] A. Ruszczyński, *Nonlinear Optimization*. Princeton, NJ, USA: Princeton University Press, 2006.

[5] E. S. Helou Neto and A. R. De Pierro, “Incremental Subgradient Algorithms for Constrained Convex Optimization: A Unified Framework and New Methods,” *SIAM J. Optim.*, vol. 20, no. 3, pp. 1547–1572, 2009.

[6] E. van den Berg and M. P. Friedlander, “Probing the Pareto Frontier for Basis Pursuit Solutions,” *SIAM J. Sci. Optim.*, vol. 31, no. 2, pp. 890–912, 2008.

[7] ℓ_1-Magic webpage, http://statweb.stanford.edu/~candes/l1magic/ [retrieved 05/27/15].

[8] SparseLab webpage, https://sparselab.stanford.edu/ [retrieved 05/27/15].

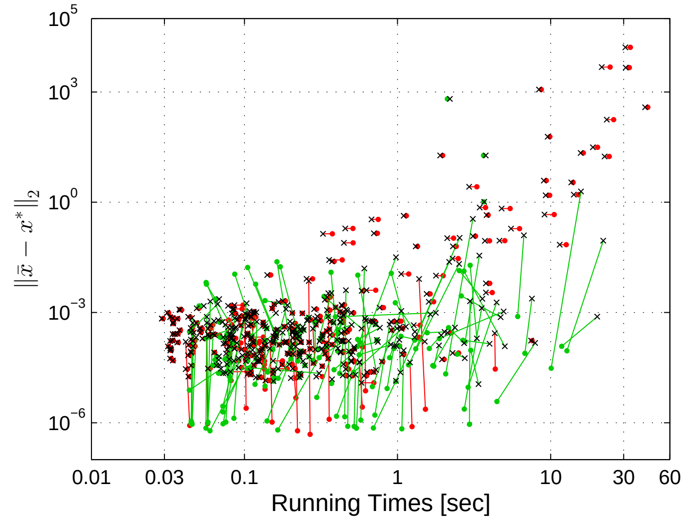

Fig. 2. HOC impact within SPGL1 [6] applied to (P_1) with ℓ_2-constraints; 444 test instances from [3] with average $\delta \approx 0.1$, (m, n) ranging from $(512, 1024)$ to $(1024, 8192)$, with varying solution sparsities. Crossmarks represent results without HOC, dots those with HOC called every $\lfloor m/20 \rfloor$ iterations; results for the same instances are connected by lines (green/red: faster/slower with HOC). Overall average speed-up: 3%.

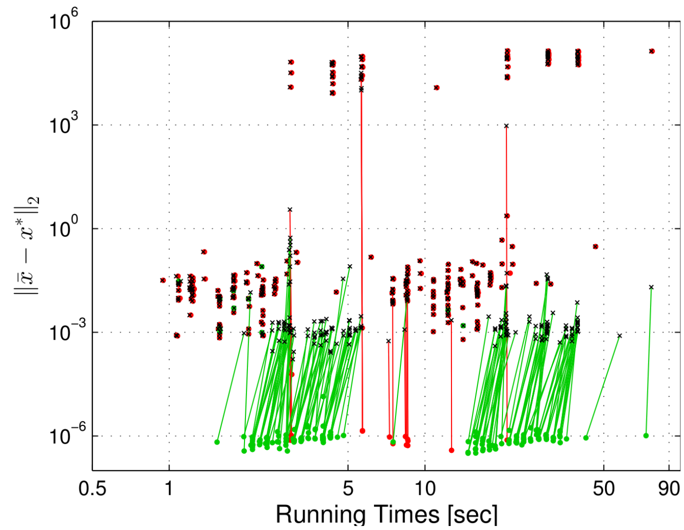

Fig. 3. HOC impact within SolveBP/PDCO applied to $\min \lambda\|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2$; 444 test instances from [3] with average $\lambda = 10$, (m, n) ranging from $(512, 1024)$ to $(1024, 8192)$, with varying solution sparsities. Crossmarks represent results without HOC, dots those with HOC called in every iteration; results for the same instances are connected by lines (green/red: faster/slower with HOC). Overall average speed-up: 5.67%.