2018 5th International Workshop on Compressed Sensing applied to Radar, Multimodal Sensing, and Imaging (CoSeRa)

Are good local minima wide in sparse recovery?

Michael Moeller and Otmar Loffeld
University of Siegen
Hölderlinstraße 3, 57076 Siegen

Jürgen Gall
University of Bonn
Endenicher Allee 19a, 53115 Bonn

Felix Krahmer
Technical University of Munich
Boltzmannstr. 3, 85748 Garching b. München

***Abstract*—The idea of compressed sensing is to exploit representations in suitable (overcomplete) dictionaries that allow to recover signals far beyond the Nyquist rate provided that they admit a sparse representation in the respective dictionary. The latter gives rise to the sparse recovery problem of finding the best sparse linear approximation of given data in a given generating system. In this paper we analyze the iterative hard thresholding (IHT) algorithm as one of the most popular greedy methods for solving the sparse recovery problem, and demonstrate that systematically perturbing the IHT algorithm by adding noise to intermediate iterates yields improved results. Further improvements can be obtained by entirely rephrasing the problem as a parametric deep-learning-type of optimization problem. By introducing perturbations via dropout, we demonstrate to significantly outperform the classical IHT algorithm, obtaining 3 to 6 times lower average objective errors.**

I. Introduction

A. Sparse Recovery

In many applications, the number of possible measurements is limited by physical or financial constraints on the measuring system. The field of compressed sensing has demonstrated that taking significantly fewer (linear) measurements than the number of unknowns still allows for exact recovery when assuming sparsity in a suitable (overcomplete) dictionary. By combining the measurement matrix and the dictionary into a single linear operator $A \in \mathbb{R}^{m\times n}$, the sparse recovery problem can be stated as the solution of the minimization problem

$$\min_u \|Au - f\|^2 \qquad \text{s.t.} \quad |u|_0 \leq s, \tag{1}$$

for $f \in \mathbb{R}^m$ being measured data, $|u|_0$ denoting the number of nonzero entries in u, and s being a known (or estimated) level of sparsity of the desired solution.

Although (1) is NP-hard in general (see e.g. [1]), several greedy algorithms as well as convex relaxation approaches guarantee to find the global minimizer provided that A meets certain regularity conditions such as the restricted isometry property (RIP), and often yield faithful approximate solution far beyond the theoretical guarantees, see [2] for details.

In general, a popular approach for solving constrained optimization problems of the form

$$\min_u E(u) \qquad \text{s.t.} \qquad u \in M, \tag{2}$$

for some set M and an energy E whose gradient ∇E is L-Lipschitz continuous, is to iteratively minimize a majorizer of E, giving rise to the so-called *gradient projection* algorithm

$$u^{k+1} \in \arg\min_{u\in M} E(u^k) + \langle \nabla E(u^k), u - u^k\rangle + \frac{1}{2\tau}\|u-u^k\|_2^2,$$

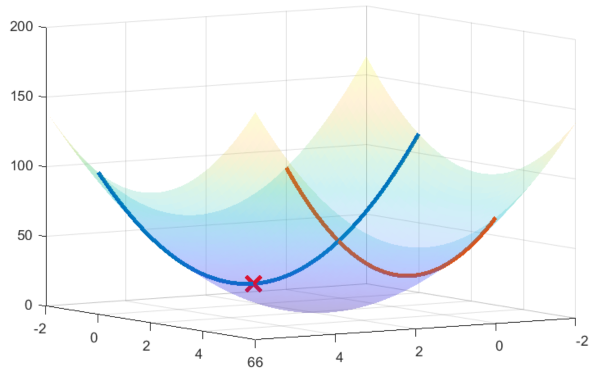


Fig. 1. Illustrating the energy landscape of (1) for $m = 2$ and $s = 1$: The semi-transparent convex surface represents $\|Au - f\|^2$, which, however, is restricted to a feasible set of one-sparse solutions represented by the union of the orange and blue curves. While the red cross illustrates the global minimum, one can see that there is a local minimum on the orange curve, too. In this paper we investigate possible strategies to avoid getting stuck on such (inferior) local minima.

for $\tau \in]0, \frac{1}{L}[$. Convergence follows from the majorization-minimization framework under weak assumptions, see e.g. [3], [4], [5].

Interestingly, although the set $M := \{u \in \mathbb{R}^m \mid |u|_0 \leq s\}$ is not convex, a (possibly not unique) projection can still be computed efficiently by simply keeping the s largest entries in magnitude and setting the remaining entries to zero. Denoting this *hard thresholding* operation by H_s, and applying the gradient projection algorithm to (1) yields the *iterative hard thresholding* (IHT) algorithm [6], [7],

$$u^{k+1} = H_s(u^k - \tau A^T(Au^k - f)). \tag{3}$$

Despite its simplicity, the IHT algorithm is one of the most popular sparse recovery algorithms, and is among the greedy algorithms with the least restrictive assumptions for exact recovery ([2, p. 29]).

In general, we have to expect the energy landscape of (1) to have many local minima on the feasible set, as we have illustrated for the simple case of $m = 2$ and $s = 1$ in Figure 1. Therefore, an interesting question is if there are systematic strategies that tend to avoid getting stuck in bad local minima?

Of course, one cannot expect a positive answer to this question for arbitrary nonconvex problems. However, let us assume/conjecture that good local minima tend to be wide for (1) in many practically relevant settings, e.g. for random measurement matrices. As a consequence of the majorization-minimization framework, the IHT algorithm monotonically decreases the objective and provably converges to a local minimizer, see [6]. If one perturbs the locally optimal solution several times and the IHT algorithm converges to the same locally optimal solution, one can conjecture to be in a wide local minimizer of the energy landscape, which - according to our assumption - has a comparably low energy. Vice versa, perturbing a locally optimal solution in a narrow minimum would likely lead to a different solution when continuing to optimize with IHT.

With the above conjecture and intuition in mind, this manuscript investigates the behavior of two different strategies for obtaining perturbed IHT algorithms - one straight-forward algorithm that adds Gaussian noise to the IHT solution after a fixed number of iterations, and one algorithm that parameterizes the solution in terms of the IHT updates themself and adapts the idea of dropout [8] from the world of deep learning.

After summarizing some related work in Section II, we present the perturbed IHT methods in Sections III-A and III-B. We demonstrate, in Section IV, that the variants indeed exhibit systematically better performance as compared to the classical IHT and draw (preliminary) conclusions in Section V.

II. Related Work

A. Sparse recovery

As the central problem in the field of compressed sensing, the solution of (1) has been studied in detail in the literature. While the exact solution to (1) is known to be NP-hard to compute in general [1], many greedy strategies, e.g. orthogonal matching pursuit (OMP) [9], CoSaMP [10], the IHT algorithm [6] discussed in the introduction, or hard thresholding pursuit (HTP) [11], as well as ℓ^1 minimizing convex relaxation approaches [12] are known to yield exact sparse recovery under certain conditions usually related to the restricted isometry property (RIP), see [2] for an overview. Recent work on partial hard thresholding [13] provides a novel framework including new algorithms and nicely summarizes some recent exact recovery results.

In this work we are less interested in exact recovery guarantees, but rather ask the question how close one can get to global minimizers of the nonconvex problem (1) in practice. For this purpose we focus on perturbations of the IHT algorithm. Since we not only consider the straight-forward approach of adding noise to the iterates, but also a parametric approach based on deep learning ideas, we briefly summarize the main necessary concepts.

B. Deep learning

The core idea of deep learning is to use a parametric function $\mathcal{N}(x;\theta)$ that maps the input x to the desired output. The *network* $\mathcal{N}$ ideally has to be parameterized by the *weights* θ in such a way that all desired outputs, but as few (undesirable) other elements as possible, lie in the range of the network.

The simplest and most generic architectures are *fully connected networks*, for which $\mathcal{N}$ is given by a composition

$$\mathcal{N}(x;\theta) = \phi(\sigma(\ldots\phi(\sigma(\phi(x;\theta_1));\theta_2)\ldots);\theta_L) \tag{4}$$

of affine linear transfer functions

$$\phi(x;\theta_i) = \theta_i^W x + \theta_i^b, \tag{5}$$

and nonlinearities σ, e.g. rectified linear units $\sigma(x) = \max(x,0)$. The overall parameters θ consist of the parameters θ_i of each *layer*, which themselves typically divide into a weight matrix θ_i^W and a *bias* θ_i^b.

Once an appropriate architecture for the network $\mathcal{N}$ has been chosen, the *training* consists of an optimization problem

$$\hat{\theta} \in \arg\min_{\theta} \sum_{i\in\text{training set}} \mathcal{L}(\mathcal{N}(x_i;\theta), y_i), \tag{6}$$

in which one tries to determine parameters $\hat{\theta}$ that yield the lowest loss $\mathcal{L}$ on the *training data*, typically consisting of pairs (x_i, y_i) of inputs x_i and desired outputs y_i of the network.

We refer the reader to [14] for a more detailed introduction.

To obtain a sufficiently expressive network many researchers have turned to deeply nested functions $\mathcal{N}$ with such a large number of weights θ, that the training data can often be fitted almost perfectly. In order to regularize the training (6) of the network and prevent *overfitting*, a very successful strategy is to introduce a *dropout layer* [8], i.e. a random perturbation, into the training. More precisely, a dropout layer sets a certain fraction of the entries of an input vector to zero at random positions, and leaves the others untouched. This strategy has proven to be very efficient in preventing overfitting, and avoiding narrow local minima in the energy landscape of the training (6), which makes it an interesting approach for our purposes.

Finally, a common strategy for learning approaches inspired by variational or partial differential equation based approaches is to roll-out suitable algorithms and treat certain parts of the latter as learnable parameters of the resulting network – see e.g. [15], [16], [17] for details.

C. Dithering

Another area where artificially introducing perturbations has proven useful is in analog-to-digital conversion, in this context this approach is known under the name of dithering. See for example [18] for a recent analysis of its benefits.

III. Perturbing IHT

As an additional motivation for a perturbed IHT, we do the following test in 2D, i.e. for $n = m = 2$ and $s = 1$. We draw a random measurement matrix A with columns of norm two, and random data f with norm one. We run the IHT algorithm from 81×81 different initial positions in $[-1,1]^2$ with step size $\tau = 0.05/\|A\|^2$. Figure 2 illustrates the information such a run contains: One (typically) obtains two minima with corresponding regions in which one has to start to converge to the respective point. The conjecture the perturbed IHT algorithms are based on is that the suboptimal local minimum is closer to the region that converges to the global minimum than the global is to the region which converges to the suboptimal local one. The latter would make a perturbed algorithm more systematically more successful. While we picked a case where our assumption is met for illustration purposes in Figure 2, we ran the above random setting 1000 times, found 890 settings with two local minima in $[-1,1]$ and computed the average distances discussed above: Indeed the average distance of the global minimum to the local region

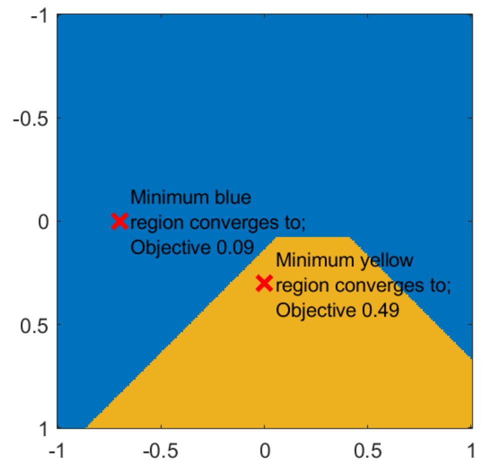


Fig. 2. Running IHT in a simple 2D setting often yields cases in which the region of starting points that converge to the global minimum is significantly closer to the suboptimal local minimum than the global minimum is to the region of points from which IHT converges to the suboptimal local minimum.

was 0.254 while the distance of the suboptimal local minimum to the global region was 0.144. Of course, the above test is very simplistic and we had to pick a relatively small τ in order to arrive local minima more often. Nevertheless, we hope for the intuition to carry over to the significantly more difficult recovery problems in higher dimensions. The following section discusses two variants of perturbed IHT algorithms.

A. Adding noise

The first (straight-forward) way to perturb the IHT algorithm is to add noise after a fixed number of iterations. For this purpose, we run the IHT algorithm for 600 iterations before adding zero-mean Gaussian noise with standard deviation $\sigma = 0.025$ to its outcome, which is subsequently used as the input to another 600 IHT iterations. We stop the process after the fifth run of the IHT algorithm. While this procedure can surely be optimized in terms of the hyperparameters, the number of times the system is perturbed, and the tracking of the best solution among all reinitializations, the goal of our work is to see if this simple perturbation strategy allows to demonstrate a systematic improvement over the plain IHT algorithm (which we run for $5 \cdot 600 = 3000$ iterations for the sake of fairness in our numerical experiments). We refer to this variant as the *noisy IHT algorithm.*

B. Optimizing the algorithm with dropout

An entirely different approach is roll out the IHT algorithm and treat the updates themselves as learnable parameters: Note that L iterations of the IHT algorithm can be written as

$$u^L = H_s(\phi(\dots H_s(\phi(H_s(\phi(u^0;\theta_1));\theta_2))\dots;\theta_L)), \qquad (7)$$

for ϕ denoting the linear transfer functions defined in equation (5), and the parameters θ_i taking the specific form

$$\theta_i^W = Id - \tau A^T A, \quad \theta_i^b = \tau A^T f, \quad \forall i \in \{1,\dots,L\}. \qquad (8)$$

In full analogy to (4) one could now treat $u^L =: \mathcal{N}(u^0;\theta)$ as a neural network and try to optimize

$$\min_\theta \|A\mathcal{N}(u^0;\theta) - f\|_2^2. \qquad (9)$$

Note, however, that using (8) as an initialization for the weights along with a deep network leads to the initial parametrization yielding a local minimizer such that gradient based weight optimization can never progress. Moreover, each layer of the parametrization (7) has $n(n+1)$-many free parameters, making it expensive to optimize deep architectures.

We therefore fix the first 3000 iterations to the noisy IHT algorithm and merely use two further IHT iterations as our parametrization. To avoid the problem of starting at a local minimum, we pick up the perturbation idea and introduce an intermediate dropout layer. More precisely, we consider the architecture

$$\mathcal{N}(u^0;\theta) = H_s(\phi(H_s(\text{drop}(\phi(u^0;\theta^1);5\%));\theta^2)), \qquad (10)$$

use the result of the noisy IHT algorithm as u^0, initialize according to (8), and optimize (9). The layer drop$(x;5\%)$ denotes randomly setting 5% of the entries of x to zero and is only used during the optimization, not for the final prediction. We refer the reader to [8] for details on such a *dropout.*

For the optimization itself we utilize a subgradient-descent-type algorithm with momentum as commonly used in learning applications. Similar to the way rectified linear or max-pooling units are optimized in the deep-learning literature, we ignore the set of non-differentiable points in the hard-thresholding operator H_s and use

$$(\nabla H_s(x))_i = (|(H_s(x))_i| > 0) \cdot x_i$$

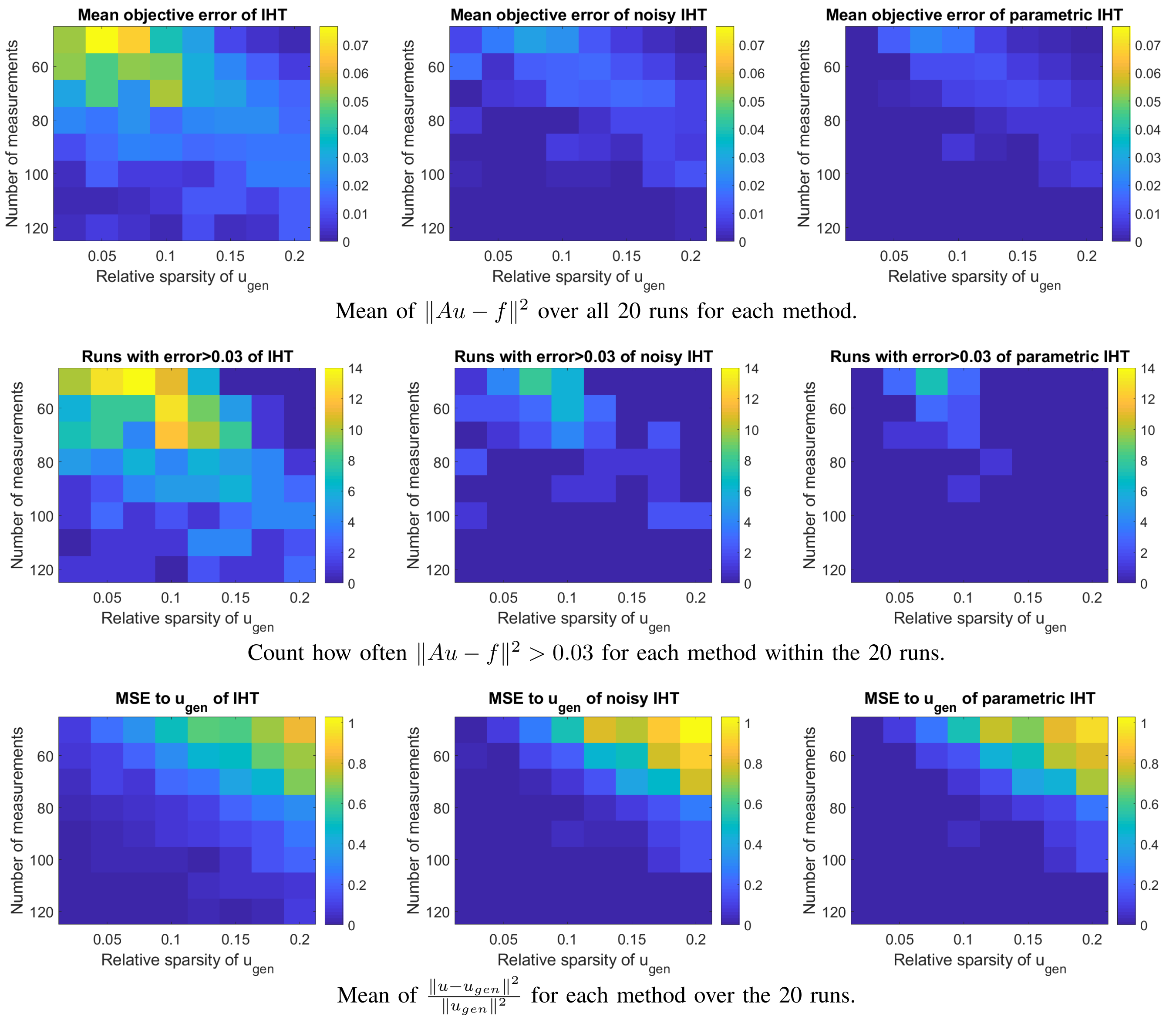
as a (sub-)derivative. Practically, we use Matlab's deep learning framework, set the momentum to 0.9, the step size (learning rate) to 10^{-4}, and run the algorithm for 2000 iterations. Despite the lack of convergence guarantee for such an algorithm, it successfully reduced the objective value in all test cases. We refer to this approach as the *parametric IHT method.*

Note that the parametric IHT method is not a learning based approach to sparse recovery, but merely uses a particular parametrization of the solution space which exploits some concepts borrowed from the deep learning literature. Similar ideas of regularization by reparametrization have recently been made for image reconstruction in [19], where – similar to the idea of (9) – a deep network based architecture is used to parameterize natural images.

It is worth pointing out that the bias of the last linear transfer function is of the same size as the desired solution already, such that (10) represents a significant overparametrization, which on the other hand also appears to be the reason why the method outperforms classical optimization algorithms such as IHT significantly, as we will see in the next section. Similar to other areas of nonconvex optimization, e.g. [20], increasing the dimensionality of the underlying problem seems to help in avoiding local minima even before entering a setting of convex lifting approaches.

IV. Numerical Results

We investigate the behavior of the IHT, the noisy IHT, and the parametric IHT algorithms by generating matrices $A \in \mathbb{R}^{m\times n}$ with entries sampled from a Gaussian distribution, using a fixed $n = 200$, and varying $m \in \{50, 60, 70, \dots, 120\}$. We generate data $f = Au_{gen}$ for u_{gen} having $s = \text{round}(\mu \cdot n)$ many nonzero entries also drawn from a Gaussian distribution.



Mean of $\|Au - f\|^2$ over all 20 runs for each method.

Count how often $\|Au - f\|^2 > 0.03$ for each method within the 20 runs.

Mean of $\frac{\|u - u_{gen}\|^2}{\|u_{gen}\|^2}$ for each method over the 20 runs.

Fig. 3. Comparison of the average objective error $\|Au - f\|^2$ (first row), the number of times the objective error exceeded 0.03 among 20 runs (middle row), and the relative difference $\|u - u_{gen}\|^2/\|u_{gen}\|^2$ to the element that generated the data $f = Au_{gen}$.

We vary the relative sparsity $\mu \in \{0.025, 0.05, 0.075, \dots, 0.2\}$, and finally normalize to $\|f\|_2 = 1$. We then run each of the algorithms with the number $s = |u_{gen}|_0$ of nonzero entries we used to generate the data with, and measure their performance in terms of the objective error $\|Au - f\|^2$. To avoid the issue of possibly having taken too few iterations, we solve one final least-squares problem on the support of the solution returned by each of the algorithms.

The first row of figure 3 shows the average objective error over 20 realizations of the IHT, the noisy IHT, and the parametric IHT approaches. The second row shows the count how often (among the 20 overall runs) the objective error exceeded a value of 0.03, and, for the sake of interpretability, the third row shows the normalized difference to the generating element $\|u - u_{gen}\|^2/\|u_{gen}\|^2$ for each of the three methods.

We conducted similar experiments with measurement matrices A arising from a Bernoulli-sampling, and subsampled discrete cosine transform matrices. Since the results were qualitatively similar, we are merely showing the Gaussian case for the sake of brevity.

As an additional summary, Table I shows the average objective error of all three methods over all experiments of Figure 3. As we can see, both perturbations of the IHT algorithm led to significant improvements over the classical IHT with the overall average objective error being improved by a factor of 3.4 by the noisy IHT and by a factor of 6 by the parametric IHT. While the number $2n(n+1)$ of free parameters of the parametric IHT approach is notably larger than the n parameters of the original unknown (leading to a runtime of up to 7-8 seconds on a CPU in the above test), the resulting optimization with dropout is significantly less prone to getting stuck in local minima and yields surprisingly large improvements.

TABLE I. AVERAGE OBJECTIVE ERROR $\|Au - f\|^2$ OF THE CLASSICAL IHT, THE NOISY IHT, AND THE PARAMETRIC IHT ALGORITHMS, OVER ALL RELATIVE SPARSITIES AND ALL NUMBER OF MEASUREMENTS FROM FIGURE 3.

Method	IHT	Noisy IHT	Param. IHT
Avg. $100 \cdot \|Au - f\|^2$	2.0	0.59	0.33

As we can see in Figure 3, a small support of u_{gen} along with a small number of measurements caused relatively large objective errors of the classical IHT, which could be improved significantly by its perturbed variants. Since the histogram graphs (second row in Figure 3) look quite similar to the mean objective errors of the first row, we can conclude that the errors of IHT are not just based on a single unlucky random realization, but yield systematic problems with up to 14 failures to reduces the objective below 0.03.

Note that the goal of this work is to investigate the minimization of $\|Au - f\|$ under sparsity constraints rather than the exact recovery of the generating element u_{gen}. Nevertheless, we included the normalized distance to u_{gen} for the sake of completeness in the last row of Figure 3. We would, however, like to point out that settings with few measurements and large supports, e.g. $m = 50$ and $s = 40$ (upper right part of the plots in the last row of Figure 3), naturally imply that we can neither expect u_{gen} to be the unique element to meet $f = Au$, nor the sparsest. This means that although the noisy and parametric IHT gave very high values of $\|u - u_{gen}\|^2 / \|u_{gen}\|^2$ in such cases, they do provide desirable solutions with almost no objective errors. Moreover, differing from u_{gen} in only one or two components can already lead to comparably high objective errors in the case of few measurements as we can see in the IHT graphs in the upper left. Nevertheless, in settings of sparse u_{gen} and moderately many measurements, the last row of Figure 3 also indicates that the perturbed IHT algorithm have a higher success rate in recovering u_{gen}.

Future research will try to give some theoretical explanations for the behavior we observed in the the numerical studies above. In particular, we will consider the case of noisy measurements and investigate strategies to quantify by what amount the IHT algorithm should ideally be perturbed. Furthermore, an interesting question is the optimal design of the network-like parametrization.

V. CONCLUSION

In this paper we have considered the nonconvex optimization problem of minimizing a quadratic objective subject to a constraint on the maximum number of nonzero entries of the unknown. Based on the assertion that good local minima are wide, we compared the IHT algorithm with a simple modification that perturbs the iterates with noise, as well as with a reparametrization of the solution in a network-like architecture which was perturbed by incorporating dropout. It was shown that the noisy IHT yields significant improvements over the plain IHT algorithm, while the parametric IHT approach involving dropout can further improve the noisy IHT results by a large margin. In summary, our findings support the hypothesis that good local minima are wide for randomly sampled measurement matrices and exact data, and demonstrate that a dropout-based parametrization of the solution space is an excellent approach for exploiting this property.

REFERENCES

[1] Y. Chen and M. Wang, "Hardness of approximation for sparse optimization with l_0 norm," 2016, preprint at https://pdfs.semanticscholar.org/9fb3/d78128e33d057b0399948f882babafa3eb31.pdf.

[2] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Birkhäuser Basel, 2013.

[3] D. Hunter and K. Lange, "A tutorial on mm algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004. [Online]. Available: https://doi.org/10.1198/0003130042836

[4] J. Mairal, "Optimization with first-order surrogate functions," in *ICML*, 2013, pp. III–783–III–791.

[5] Y. Sun, P. Babu, and D. Palomar, "Majorization-minimization algorithms in signal processing," *IEEE Transactions on Signal Processing*, vol. 65, pp. 794–816, 2017.

[6] T. Blumensath and M. Davies, "Iterative thresholding for sparse approximations," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 629–654, Dec 2008.

[7] ——, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265 – 274, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520309000384

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[9] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec 1993.

[10] D. Needell and J. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301 – 321, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520308000638

[11] S. Foucart, "Hard thresholding pursuit: An algorithm for compressive sensing," *SIAM Journal on Numerical Analysis*, vol. 49, no. 6, pp. 2543–2563, 2011. [Online]. Available: https://doi.org/10.1137/100806278

[12] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998. [Online]. Available: https://doi.org/10.1137/S1064827596304010

[13] P. Jain, A. Tewari, and I. S. Dhillon, "Partial hard thresholding," *IEEE Transactions on Information Theory*, vol. 63, no. 5, pp. 3029–3038, May 2017.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[15] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr, "Conditional random fields as recurrent neural networks," in *International Conference on Computer Vision (ICCV)*, 2015.

[16] Y. Chen, W. Yu, and T. Pock, "On learning optimized reaction diffusion processes for effective image restoration," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 5261–5269.

[17] A. Richard and J. Gall, "A bag-of-words equivalent recurrent neural network for action recognition," *Computer Vision and Image Understanding*, vol. 156, pp. 79–91, 2017.

[18] L. J. C. Xu, "Quantized compressive sensing with rip matrices: The benefit of dithering," preprint available online at https://arxiv.org/abs/1801.05870.

[19] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," preprint available online at https://arxiv.org/abs/1711.10925.

[20] C. Zach, "Robust bundle adjustment revisited," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 772–787. [Online]. Available: https://doi.org/10.1007/978-3-319-10602-1_50