Sparse Convex Optimization via Adaptively Regularized Hard Thresholding

Kyriakos Axiotis 1  Maxim Sviridenko 2

Abstract
The goal of Sparse Convex Optimization is to optimize a convex function \( f \) under a sparsity constraint \( s \leq s^* \gamma \), where \( s^* \) is the target number of non-zero entries in a feasible solution (sparsity) and \( \gamma \geq 1 \) is an approximation factor. There has been a lot of work to analyze the sparsity guarantees of various algorithms (LASSO, Orthogonal Matching Pursuit (OMP), Iterative Hard Thresholding (IHT)) in terms of the Restricted Condition Number \( \kappa \). The best known algorithms guarantee to find an approximate solution of value \( f(x^*) + \epsilon \) with the sparsity bound of

\[
\gamma = O \left( \kappa \min \left\{ \log \left( \frac{(x^*)^2}{\delta^2} \right), \frac{1}{\epsilon} \right\} \right),
\]

where \( x^* \) is the target solution. We present a new Adaptively Regularized Hard Thresholding (ARHT) algorithm that makes significant progress on this problem by bringing the bound down to \( \gamma = O(\kappa) \), which has been shown to be tight for a general class of algorithms including LASSO, OMP, and IHT. This is achieved without significant sacrifice in the runtime efficiency compared to the fastest known algorithms. We also provide a new analysis of OMP with Replacement (OMPR) for general \( f \), under the condition \( s > s^* \frac{\kappa^2}{\pi^2} \), which yields Compressed Sensing bounds under the Restricted Isometry Property (RIP). When compared to other Compressed Sensing approaches, it has the advantage of providing a strong tradeoff between the RIP condition and the solution sparsity, while working for any general function \( f \) that meets the RIP condition.

1. Introduction
Sparse Convex Optimization is the problem of optimizing a convex objective, while constraining the sparsity of the solution (its number of non-zero entries). Variants and special cases of this problem have been studied for many years, and there have been countless applications in Machine Learning, Signal Processing, and Statistics. In Machine Learning it is used to regularize models by enforcing parameter sparsity, since a sparse set of parameters often leads to better model generalization. Furthermore, in a lot of large scale applications the number of parameters of a trained model is a significant factor in computational efficiency, thus improved sparsity can lead to improved time and memory performance. In applied statistics, a single extra feature translates to a real cost from increasing the number of samples. In Compressed Sensing, finding a sparse solution to a Linear Regression problem can be used to significantly reduce the sample size for the recovery of a target signal. In the context of these applications, decreasing sparsity by even a small amount while not increasing the accuracy can have a significant impact.

Sparse Optimization Given a function \( f : \mathbb{R}^n \to \mathbb{R} \) and any \( s^* \)-sparse (unknown) target solution \( x^* \), the Sparse Optimization problem is to find an \( s \)-sparse solution \( x \), i.e. a solution with at most \( s \) non-zero entries, such that \( f(x) \leq f(x^*) + \epsilon \) and \( s \leq s^* \gamma \), where \( \epsilon > 0 \) is a desired accuracy and \( \gamma \geq 1 \) is an approximation factor for the target sparsity. Even if \( f \) is a convex function, the sparsity constraint makes this problem non-convex, and it has been shown that it is an intractable problem, even when \( \gamma = O \left( \frac{2^{s^*}}{n} \right) \) and \( f \) is the Linear Regression objective (Natarajan, 1995; Foster et al., 2015). However, this worst-case behavior is not observed in practice, and so a large body of work has been devoted to the analysis of algorithms under the assumption that the restricted condition number \( \kappa_{s+s^*} = \frac{\rho_{s+s^*}}{\rho_{s+s^*}} \) (or just \( \kappa = \frac{\rho_{s+s^*}}{\rho_{s+s^*}} \)) of \( f \) is bounded (Natarajan, 1995; Shalev-Shwartz et al., 2010; Zhang, 2011; Bahmani et al., 2013; Liu et al., 2014; Jain et al., 2014; Yuan et al., 2016; Shen & Li, 2017a;b; Jain et al., 2014; Somani et al., 2018). Note: Here, \( \rho_{s+s^*} \) is the maximum smoothness constant of any restriction of \( f \) on an \((s+s^*)\)-sparse subset of coordinates and \( \rho_{s+s^*} \) is the minimum strong convexity constant of any restriction of \( f \) on an \((s+s^*)\)-sparse subset of coordinates.

The first algorithm for this problem, often called Orthogonal Matching Pursuit (OMP) or Greedy, was analyzed by

---

1MIT 2Yahoo! Research. Correspondence to: Kyriakos Axiotis <kaxiotis@mit.edu>, Maxim Sviridenko <sviri@verizonmedia.com>.

Optimal Constant (RGOC) at level \( N \) (Natarajan, 1995) for Linear Regression, and subsequently for general \( f \) by (Shai et al., 2010), obtaining the guarantee that the sparsity of the returned solution is \( O\left(s^* \kappa \log \frac{\|f(x^*)\|}{\|f(x)\|-\|f(x^*)\|}\right) \). In applications where having low sparsity is crucial, the dependence of sparsity on the required accuracy \( \epsilon \) is undesirable. The question of whether this dependence can be removed was answered positively (Shai et al., 2010; Jain et al., 2014) giving a sparsity guarantee of \( O(s^* \kappa^2) \). As remarked in (Shai et al., 2010), this bound sacrifices the linear dependence on \( \kappa \), while removing the dependence on \( \epsilon \) and \( f(x^0) - f(x^*) \).

Since then, there has been some work on improving these results by introducing non-trivial assumptions, such as the target solution \( x^* \) being close to globally optimal. More specifically, (Zhang, 2011) defines the Restricted Gradient Optimal Constant (RGOC) at level \( s^* \), \( \zeta^* \) (or just \( \zeta^* \)) as the \( \ell_2 \) norm of the top-\( s^* \) elements in \( \nabla f(x^*) \) and analyzes an algorithm that gives sparsity \( s = O(s^* \kappa \log (s^* \kappa)) \), and such that \( f(x) \leq f(x^*) + O(\zeta^*/\rho^*) \). (Somani et al., 2018) strengthens this bound to \( f(x) \leq f(x^*) + O(\zeta^*/\rho^*) \) with sparsity \( s = O(s^* \kappa \log \kappa) \). However, this means that \( f(x) \) might be much larger than \( f(x^*) + \epsilon \) in general. To the best of our knowledge, no improvement has been made over the \( O\left(s^* \min \left\{ \kappa, \frac{\|\nabla f(x^*)\|}{\epsilon}, \kappa^2 \right\}\right) \) bound in the general case.

Another line of work studies a maximization version of the sparse convex optimization problem as well as its generalizations for matroid constraints (Altschuler et al., 2016; Elenberg et al., 2017; Chen et al., 2018).

### Sparse Solution and Support Recovery

Often, as is the case in Compressed Sensing, one needs a guarantee on the closeness of the solution \( x \) to the target solution \( x^* \) in absolute terms, rather than in terms of the value of \( f \). The goal is usually either to recover (a superset of) the target support, or to ensure that the returned solution is close to the target solution in \( \ell_2 \) norm. The results for this problem either assume a constant upper bound on the Restricted Isometry Property (RIP) constant \( \delta_r := \frac{s^* - 1}{s^* + 1} \) for some \( r \) (RIP-based recovery), or that \( x^* \) is close to being a global optimum (RIP-free recovery). This problem has been extensively studied and is an active research area in the vast Compressed Sensing literature. See also the survey by (Boche et al., 2015).

In the seminal papers of (Candes & Tao, 2005; Candes et al., 2006; Donoho, 2006; Candes, 2008) it was shown that for the Linear regression problem when \( \delta_{3s^*} < \sqrt{2} - 1 \approx 0.41 \), the LASSO algorithm (Tibshirani, 1996) can recover a solution with \( \|x - x^*\|_2 \leq C f(x^*) \), where \( C \) is a constant depending only on \( \delta_{3s^*} \) and \( f(x^*) = \frac{1}{2} \|Ax^* - b\|_2^2 \) is the error of the target solution\(^1\). Since then, a multitude of results of similar flavor have appeared, either giving related guarantees for the LASSO algorithm while improving the RIP upper bound (Foucart & Lai, 2009; Cai et al., 2009; Foucart, 2010; Cai et al., 2010; Mo & Li, 2011; Anderson & Strömberg, 2014) which culminate in a bound of \( \delta_{3s^*} < 0.6248 \), or showing that similar guarantees can be obtained by greedy algorithms under more restricted RIP conditions, but that are typically faster than LASSO (Needell & Vershynin, 2009; 2010; Needell & Tropp, 2009; Blumensath & Davies, 2009; Jain et al., 2011; Foucart, 2011; 2012). See also the comprehensive surveys (Foucart & Rauhut, 2017; Mousavi et al., 2019).

(Needell & Tropp, 2009) presents a greedy algorithm called CoSaMP and shows that for Linear Regression it achieves a bound in the form of (Candes, 2008) while having a more efficient implementation. Their method works for the more restricted RIP upper bound of \( \delta_{3s^*} < 0.025 \), or \( \delta_{4s^*} < 0.4782 \) as improved by (Foucart & Rauhut, 2017). (Blumensath & Davies, 2009) proves that another greedy algorithm called Iterative Hard Thresholding (IHT) achieves a similar bound to that of CoSaMP for Linear Regression, with the condition \( \delta_{6s^*} < 0.067 \), which is improved to \( \delta_{6s^*} < 0.5641 \) by (Jain et al., 2011) and to \( \delta_{3s^*} < 0.5774 \) by (Foucart, 2011). The RIP-free line of research has shown that strong results can be achieved without a RIP upper bound, given that the target solution is sufficiently close to being a global optimum. These results typically require that \( s \) is significantly larger than \( s^* \). In particular, (Zhang, 2011) shows that if \( \zeta \) is the RGOC of \( f \) it can be guaranteed that \( \|x - x^*\|_2 \leq \sqrt{1 + \frac{\zeta}{\rho^*}} (\text{or } \sqrt{1 + \frac{\zeta}{\rho^*}} \text{ with a slightly tighter analysis}) \). (Somani et al., 2018) strengthens this bound to \( \left(1 + \frac{\zeta}{\rho^*}\right) \leq \frac{1}{\rho^*} \). Furthermore, it has been shown that as long as a “Signal-to-Noise” condition holds, one can actually recover a superset of the target support. Typically the condition is a lower bound on \( |x_{\text{min}}^*| \), the minimum magnitude non-zero entry of the target solution. Different lower bounds that have been devised include \( \Omega\left(\frac{\sqrt{s^* + s}\|\nabla f(x^*)\|_{\infty}}{\rho^{s^* + s}}\right) \) (Jain et al., 2014), which was later improved to \( \Omega\left(\sqrt{\frac{\|x^*\|_{\infty}}{\rho}}\right) \), where \( \pi^* \) is an optimal \( s \)-sparse solution (Yuan et al., 2016). Finally, (Somani et al., 2018) improves the sparsity bound to \( O(s^* \kappa \log (s^* \kappa)) \) in the statistical setting and (Shen & Li, 2017b) shows that the sparsity can be brought down to \( s = s^* + O(\kappa^2) \) if a stronger lower bound of \( \Omega\left(\sqrt{\frac{\kappa}{\rho}}\right) \) is assumed.

\(^1\)Even though (Natarajan, 1995) states a less general result, this is what is implicitly proven.
1.1. Our work

In this work we present a new algorithm called Adapitively Regularized Hard Thresholding (ARHT), that closes the longstanding gap between the $O\left(s^* \kappa f(x^0) - f(x^*)\right)$ and $O\left(s^* \kappa^2\right)$ bounds by getting a sparsity of $O\left(s^* \kappa\right)$ and thus achieving the best of both worlds. As (Foster et al., 2015) shows that for a general class of algorithms (including greedy algorithms like OMP, IHT as well as LASSO) the linear dependence on $\kappa$ is necessary even for the special case of Sparse Regression, our result is tight for this class of algorithms. In the supplementary material we briefly describe this example and also state a conjecture that it can be turned into an inapproximability result. Furthermore, there we also show that the $O(s^* \kappa^2)$ sparsity bound is tight for OMPR, thus highlighting the importance of regularization in our method. Our algorithm is efficient, as it requires roughly $O\left(s \log^3 \frac{f(x^0) - f(x^*)}{\varepsilon}\right)$ iterations, each of which includes one function minimization in a restricted support of size $s$ and is simple to describe and implement. Furthermore, it directly implies non-trivial results in the area of Compressed Sensing.

We also provide a new analysis of OMPR (Jain et al., 2011) and show that under the condition that $s > s^* \frac{\varepsilon^2}{\kappa^2}$, or equivalently under the RIP condition $\delta_{s+s^*} < 2 \sqrt{\frac{\kappa^2 - s}{\kappa^2 + s}}$, it is possible to approximately minimize the function $f$ up to some error depending on the RIP constant and the closeness of $x^*$ to global optimality. More specifically, we show that for any $\varepsilon > 0$ OMPR returns a solution $x$ such that

$$f(x) \leq f(x^*) + \varepsilon + C_1 (f(x^*) - f(x^{opt}))$$

where $x^{opt}$ is the globally optimal solution, as well as

$$\|x - x^*\|_2^2 \leq \varepsilon + C_2 (f(x^*) - f(x^{opt}))$$

where $C_1, C_2$ are constants that only depend on $\frac{s}{\kappa}$ and $\delta_{s+s^*}$. An important feature of our approach is that it provides a meaningful tradeoff between the RIP constant upper bound and the sparsity of the solution, even when the sparsity $s$ is arbitrarily close to $s^*$. In other words, one can relax the RIP condition at the expense of increasing the sparsity of the returned solution. Furthermore, our analysis applies to general functions with bounded RIP constant.

Experiments with real data suggest that ARHT and a variant of OMPR which we call Exhaustive Local Search achieve promising performance in recovering sparse solutions.

1.2. Comparison to previous work

Sparse Optimization Our Algorithm 2 (ARHT) returns a solution with $s = O(s^* \kappa)$ without any additional assumptions, thus significantly improving over the bound $O\left(s^* \kappa^2\right)$ that was known in previous work. This proves that neither any dependence on the required solution accuracy $\varepsilon$, nor the quadratic dependence on the condition number $\kappa$ is necessary. Furthermore, no assumption on the function or the target solution is required to achieve this bound. Importantly, previous results imply that our bound is tight up to constants for a general class of algorithms, including Greedy-type algorithms and LASSO (Foster et al., 2015).

Sparse Solution Recovery In Corollary 3.5, we show that the improved guarantees of Theorem 3.3 immediately imply that ARHT gives a bound of $\|x - x^\ast\|_2 \leq (2 + \theta) \frac{\varepsilon}{\kappa}$ for any $\theta > 0$, where $\theta$ is the Restricted Gradient Optimal Constant. This improves the constant factor in front of the corresponding results of (Zhang, 2011; Somani et al., 2018).

As we saw, our Theorem 3.7 directly implies that OMPR gives an upper bound on $\|x - x^\ast\|_2^2$ of the same form as the RIP-based bounds in previous work, under the condition $\delta_{s+s^*} < 2 \sqrt{\frac{\kappa^2 - s}{\kappa^2 + s}}$. While previous results either concentrate on the case $s = s^*$, or $s \gg s^*$, our analysis provides a way to trade off increased sparsity for a more relaxed RIP bound, allowing for a whole family of RIP conditions where $s$ is arbitrarily close to $s^*$. Specifically, if we set $s = s^*$ our work implies recovery for $\delta_{2s^*} < \frac{1}{4} \approx 0.33$, which matches the best known bound for any greedy algorithm (Jain et al., 2011), although it is a stricter condition than the $\delta_{2s^*} < 0.62$ required by LASSO (Foucart & Rauhut, 2017).

Table 1 contains a few such RIP bounds implied by our analysis and shows that it readily surpasses the bounds for Subspace Pursuit $\delta_{3s^*} < 0.35$, CoSaMP $\delta_{4s^*} < 0.48$, and OMP $\delta_{31s^*} < 0.33$ (Jain et al., 2011; Zhang, 2011). Another important feature compared to previous work is that all our guarantees are not restricted to Linear Regression and are true for any function $f$, as long as it satisfies the required RIP condition, which makes the result more general.

<table>
<thead>
<tr>
<th>$s$</th>
<th>RIP CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s^*$</td>
<td>$\delta_{2s^*} &lt; 0.33$</td>
</tr>
<tr>
<td>$2s^*$</td>
<td>$\delta_{3s^*} &lt; 0.47$</td>
</tr>
<tr>
<td>$3s^*$</td>
<td>$\delta_{4s^*} &lt; 0.55$</td>
</tr>
<tr>
<td>$30s^*$</td>
<td>$\delta_{31s^*} &lt; 0.83$</td>
</tr>
</tbody>
</table>

Sparse Support Recovery Corollary 3.6 shows that as a direct consequence of our work, the condition $|x^*_{\text{min}}| > \frac{\varepsilon}{\kappa}$ suffices for our algorithm to recover a superset of the support with size $s = O(s^* \kappa)$. Compared to (Jain et al., 2014), we improve both the size of the superset, as well as
the condition, since \( \sqrt{s} \frac{\|\nabla f(x^*)\|_\infty}{\rho} \geq \sqrt{s} \frac{\kappa}{\rho} = \Omega \left( \frac{\kappa}{\rho} \right) \).

Compared to (Shen & Li, 2017b), the bounds on the superset size are incomparable in general, but our \( |x^*_{\text{min}}| \) condition is more relaxed, since \( \sqrt{n}/\rho = \Omega(1/n) \). Finally, compared to (Yuan et al., 2016) we have a stricter lower bound on \( |x^*_{\text{min}}| \), but with a better bound on the superset size \( O(s^*\kappa^2) \) instead of \( O(s^*\kappa^2) \). Although not explicitly stated, (Zhang, 2011; Somani et al., 2018) also give a similar lower bound \( \sqrt{1 + \frac{10}{\rho}} \frac{\kappa}{\rho} \) which we improve by a constant factor.

**Runtime Discussion** ARHT has the advantage of being very simple to implement in practice. The runtime of Algorithm 2 (ARHT) is comparable to that of the most efficient greedy algorithms (e.g. OMP/OMPR), as it requires a single function minimization per iteration.

**Naming Conventions** The algorithm that we call Orthogonal Matching Pursuit (OMP), is also known as “Greedy” (Natarajan, 1995), “Fully Corrective Forward Greedy Selection” or just “Forward Selection”. What we call Orthogonal Matching Pursuit with Replacement (OMPR) (Jain et al., 2011) is also known by various other names. It is referenced in (Shalev-Shwartz et al., 2010) as a simpler variant of their “Fully Corrective Forward Greedy Selection with Replacement” algorithm, or just Forward Selection with Replacement, or “Partial Hard Thresholding with parameter \( r = 1 \)” (PHT(\( r \)) where \( r = 1 \)) (Jain et al., 2017) which is a generalization of Iterative Hard Thresholding. Finally, what we call Exhaustive Local Search is essentially a variant of “Orthogonal Least Squares” that includes replacement steps, and also appears in (Shalev-Shwartz et al., 2010) as “Fully Corrective Forward Greedy Selection with Replacement”, or just “Forward Stepwise Selection with Replacement”. See also (Blumensath & Davies, 2007) regarding naming conventions.

**Remark 1.1.** Most of the results in the literature either only apply to, or are only presented for the Linear Regression problem. Since all of our results apply to general function minimization, we present them as such.

## 2. Preliminaries

**Remark 2.1.** An addendum to this section can be found in the Supplementary Material.

We denote \([i] := \{1, 2, \ldots, i\} \). For any \( x \in \mathbb{R}^n \) and \( R \subseteq [n] \), we define \( x_R \in \mathbb{R}^R \) as \( (x_R)_i = \begin{cases} x_i & i \in R \\ 0 & \text{otherwise} \end{cases} \)

Additionally, for any differentiable function \( f : \mathbb{R}^n \to \mathbb{R} \) with gradient \( \nabla f(x) \), we will denote by \( \nabla_R f(x) \) the restriction of \( \nabla f(x) \) to \( R \), i.e. \( (\nabla f(x))_R \).

In Lemma 2.2 we state a standard martingale concentration inequality that we will use. See also (Chung & Lu, 2006) for more on martingales.

**Lemma 2.2** (Martingale concentration inequality (Special case of Theorem 6.5 in (Chung & Lu, 2006))). Let \( Y_0 = 0, Y_1, \ldots, Y_n \) be a martingale with respect to the sequence \( i_1, \ldots, i_n \), such that

\[
\text{Var}(Y_k \mid i_1, \ldots, i_{k-1}) \leq \sigma^2
\]

and

\[
Y_{k-1} - Y_k \leq M
\]

for all \( k \in [n] \), then for any \( \lambda > 0 \),

\[
\Pr[|Y_n| > \lambda] = e^{-\lambda^2/(2(\sigma^2 + M\lambda/3))}
\]

**Definition 2.3.** For any \( x \in \mathbb{R}^n \), we denote the support of \( x \) by \( \text{supp}(x) = \{i : x_i \neq 0\} \).

**Definition 2.4** (Restricted Condition Number). Given a differentiable function \( f \), the Restricted \( \ell_2 \)-Smoothness (RSC) constant, or just Restricted Smoothness constant, of \( f \) at sparsity level \( s \) is the minimum \( \rho_s^+ \in \mathbb{R} \) such that

\[
f(y) \leq f(x) + \nabla f(x)^\top(y - x) + \frac{\rho_s^+}{2} \|y - x\|_2^2
\]

for any \( y, x \in \mathbb{R}^n \) with \( |\text{supp}(y - x)| \leq s \). Similarly, the Restricted \( \ell_2 \)-Strong Convexity (RSC) constant, or just Restricted Strong Convexity constant, of \( f \) at sparsity level \( s \) is the maximum \( \rho_s^- \in \mathbb{R} \) such that

\[
f(y) \geq f(x) + \nabla f(x)^\top(y - x) + \frac{\rho_s^-}{2} \|y - x\|_2^2
\]

for any \( x, y \in \mathbb{R}^n \) with \( |\text{supp}(y - x)| \leq s \). Given that \( \rho_s^+, \rho_s^- \geq 0 \), the Restricted Condition Number of \( f \) at sparsity level \( s \) is defined as \( \kappa_s = \rho_s^+/\rho_s^- \). We will also make use of \( \bar{\kappa}_s = \rho_2^+/\rho_s^- \) which is at most \( \kappa_s \) as long as \( s \geq 2 \).

**Definition 2.5** (Restricted Isometry Property (RIP)). We say that a differentiable function \( f \) has the Restricted Isometry Property at sparsity level \( s \) if \( \rho_s^+, \rho_s^- > 0 \), and the RIP constant of \( f \) at sparsity level \( s \) is then defined as \( \delta_s = \frac{\kappa_{s-1}}{\kappa_s+1}^3 \).

**Definition 2.6** (Restricted Gradient Optimal Constant (RGOCC)). Given a differentiable function \( f \) and a “target” solution \( x^* \), the Restricted Gradient Optimal Constant (Zhang, 2011) at sparsity level \( s \) is the minimum \( \zeta_s \in \mathbb{R}_+ \) such that

\[
\|\nabla f(x^*) - y\| \leq \zeta_s \|y\|_2
\]

for all \( s \)-sparse \( y \). Setting \( y = \nabla_S f(x^*) \) for some set \( S \) with \( |S| \leq s \), this implies that \( \zeta_s \geq \|\nabla_S f(x^*)\| \). An alternative definition is that \( \zeta_s \) is the \( \ell_2 \) norm of the \( s \) elements of \( \nabla f(x^*) \) with highest absolute value.

\(^3\)We note that this is a more general definition than the one usually given for quadratic functions (i.e. Linear Regression).
We eliminate this discrepancy by running the algorithm on elements \( R \) large on elements \( S \). Thus, we have \( f(x) \leq f(x^\ast) \).

3. Theoretical Results

In this section we will call the current solution \( x \) and the target solution \( x^\ast \), with respective supports \( S \) and \( S^\ast \) and sizes \( s = |S| \) and \( s^\ast = |S^\ast| \). The results in this section are stated in terms of \( \hat{\kappa} \), but since \( \hat{\kappa} = \frac{\rho_1^2}{\rho_{s+s^\ast}} \leq \frac{\rho_1^2}{\rho_{s+s^\ast}} = \kappa \), the same statements where \( \hat{\kappa} \) is replaced by \( \kappa \) automatically follow.

3.1. Adaptively Regularized Hard Thresholding (ARHT)

Our algorithm is essentially a Hard Thresholding algorithm (and more specifically OMP, or also known as PHT(1)) with the crucial novelty that it is applied on an adaptively regularized objective function.

Hard thresholding algorithms maintain a solution \( x \) supported on \( S \subseteq [n] \), which they iteratively update by inserting new elements into the support set \( S \) and removing the same number of elements from it, in order to preserve the sparsity of \( x \). More specifically, OMP makes one insertion and one removal in each iteration. In order to evaluate the element \( i \) to be inserted into \( S \), OMP uses the fact that, because of smoothness, \( \frac{(\nabla_i f(x))^2}{2 \rho_i^2} \) is a lower bound on the decrease of \( f(x) \) caused by inserting \( i \) into the support, and therefore picks \( i \) to maximize \( |\nabla_i f(x)| \). Similarly, in order to evaluate the element \( j \) to be removed from \( S \), OMP uses the fact that \( \frac{\rho_2^2}{2} x_j^2 \) upper bounds the increase of \( f(x) \) caused by setting \( x_j = 0 \), and therefore picks \( j \) to minimize \( |x_j| \). However, the real worth of \( j \) might be as small as \( \frac{\rho_2^2}{2} x_j^2 \), so the upper bound can be loose by a factor of \( \frac{\rho_1^2}{\rho_2^2} \), so the upper bound can be loose by a factor of \( \hat{\kappa} = \kappa \).

We eliminate this discrepancy by running the algorithm on the regularized function \( g(z) := f(z) + \frac{\rho_2^2}{2} \|z\|_2^2 \). As the restricted condition number of \( g \) is now \( O(1) \), the real worth of a removal candidate \( j \) matches the upper bound up to a constant factor.

However, even though \( g \) is now well conditioned, the analysis can only guarantee the quality of the solution in terms of the original objective \( f \) if the regularization is not applied on elements \( S^\ast \), i.e. \( \frac{\rho_2^2}{2} \|x_{S \setminus S^\ast}\|_2^2 \) for some sufficiently large \( R \subseteq [n] \); if this is the case, a solution with sparsity \( O(s^\ast \hat{\kappa}) \) can be recovered. Unfortunately, there is no way of knowing a priori which elements not to regularize, as this is equivalent to finding the target solution. As a result, the algorithm can get trapped in local minima, which are defined as states in which one iteration of the algorithm does not decrease \( g(x) \), even though \( x \) is a suboptimal solution in terms of \( f \) (i.e. \( f(x) > f(x^\ast) \)).

The main contribution of this work is to characterize such local minima and devise a procedure that is able to successfully escape them, thus allowing \( x \) to converge to a desired solution for the original objective.

When \( x \) has significant \( \ell_2^2 \) mass in the target support, the regularization term \( \frac{\rho_2^2}{2} \|x\|_2^2 \) might penalize the target solution too much, leading to a Type 2 iteration. In this case, we use random sampling to detect an element in the optimal support and unregularize it. This procedure escapes all local minima, thus leading to a bound in the total number of Type 2 iterations.

More concretely, we show that if at some iteration of the algorithm the value of \( g(x) \) does not decrease sufficiently (Type 2 iteration), then roughly at least a \( \frac{1}{4} \)-fraction of the \( \ell_2^2 \) mass of \( x \) lies in the target support \( S^\ast \). We exploit this property by sampling an element \( i \) proportional to \( x_i^2 \) and removing its corresponding term from the regularizer (unregularizing it). We show that with constant probability this will happen at most \( O(s^\ast \hat{\kappa}) \) times, as after this all the elements in \( S^\ast \) will have been unregularized.

The core algorithm is presented in Algorithm 1. The full algorithm additionally requires some standard routines like binary search and is presented in Algorithm 2.

Let \( R \subseteq [n] \) be the set of currently regularized elements. The following invariant is a crucial ingredient for bringing the sparsity from \( O(s^\ast \hat{\kappa}) \) down to \( O(s^\ast \hat{\kappa}) \), and we intend to enforce it at all times. It essentially states that there will always be enough elements in the current solution that are being regularized.

**Invariant 3.1.**

\[ |R \cap S| \geq s^\ast \max\{1, 8\hat{\kappa}\} \]

To give some intuition on this, ARHT owes its improved \( \hat{\kappa} \) dependence on the regularizer \( \frac{\rho_2^2}{2} \|x\|_2^2 \). However, during the algorithm, some elements are being unregularized. Our analysis requires that the current solution support always contains \( \Omega(s^\ast \hat{\kappa}) \) regularized elements, which is what Invariant 3.1 states.

In the following, we will let \( \text{opt} \) denote a guess on the target value \( f(x^\ast) \). Also, \( x^0 \) will denote the initial solution, which is an \( S^0 \)-restricted minimizer an arbitrary set \( S^0 \subseteq [n] \) with \( |S^0| = s \). In Algorithm 1, \( S^0 \) is defined explicitly as \( S \), however in practice one might want to pick a better initial set (e.g. returned by running OMP).
We begin with a lemma analyzing Algorithm 1, which is the core of our algorithm.

**Lemma 3.2.** If $s \geq s^* \max\{4\kappa + 7, 12\kappa + 6\} + \text{opt} \geq f(x^*)$, with probability at least 0.2 ARHT_core($s$, opt, $\epsilon$) returns an $s$-sparse solution $x$ such that $f(x) \leq \text{opt} + \epsilon$.

In other words, as long as opt \geq f(x^*), a solution of value \leq opt + \epsilon will be found. As the value opt is not known a priori, we perform binary search on it, as described in Algorithm 2. Furthermore, the probability of success in the previous lemma can be boosted by repeating multiple times.

The following theorem encapsulates the main result of this section.

**Theorem 3.3.** Given a function $f$ and an (unknown) $s^*$-sparse solution $x^*$, with probability at least $1 - \frac{1}{n}$ Algorithm 2 returns an $s$-sparse solution $x$ with $f(x) \leq f(x^*) + \epsilon$, as long as $s \geq s^* \max\{4\kappa + 7, 12\kappa + 6\}$. The number of iterations is $O\left(s \log^2 \frac{f(\tilde{0}) - B}{\epsilon} \log \left(n \log \frac{f(\tilde{0}) - B}{\epsilon}\right)\right)$ where $B = \min f(x)$.

The following corollary that bounds the total runtime can be immediately extracted. Note that in practice the total runtime heavily depends on the choice of $f$, and it can often be improved for various special cases (e.g. linear regression).

**Corollary 3.4 (Theorem 3.3 runtime).** If we denote by $G$ the time needed to compute $\nabla f$ and by $M$ the time to minimize $f$ in a restricted subset of $[n]$ of size $s$, the total runtime of Algorithm 2 is $O \left((G + M)s \log^2 \frac{f(\tilde{0}) - B}{\epsilon} \log \left(n \log \frac{f(\tilde{0}) - B}{\epsilon}\right)\right)$. If gradient descent is used for the implementation of the inner optimization problem, then $M = O \left(G\kappa \log \frac{f(\tilde{0}) - B}{\epsilon}\right)$ and so the total runtime can be bounded by $O \left(Gs\kappa \log^3 \frac{f(\tilde{0}) - B}{\epsilon} \log \left(n \log \frac{f(\tilde{0}) - B}{\epsilon}\right)\right)$. 

---

**Algorithm 1 ARHT core routine**

1: function ARHT_core($s$, opt, $\epsilon$)  
2: function to be minimized $f : \mathbb{R}^n \rightarrow \mathbb{R}$  
3: target sparsity $s$  
4: target value opt (current guess for the optimal value)  
5: target error $\epsilon$  
6: Define $g_{R}(x) := f(x) + \frac{x^2}{2} \|x\|_2^2$ for all $R \subseteq [n]$.  
7: $R^0 \leftarrow [n]$  
8: $S^0 \leftarrow [s]$  
9: $x^0 \leftarrow \arg\min_{\supp(x) \subseteq S^0} g_{R}(x)$  
10: $T = 2s \log g_{R}(x^0) - \min f(x)$ (number of iterations)  
11: for $t = 0 \ldots T - 1$ do  
12: if $\min_{\supp(x) \subseteq S^t} f(x) \leq \text{opt}$ then  
13: return $\arg\min_{\supp(x) \subseteq S^t} f(x)$  
14: end if  
15: $i \leftarrow \arg\max_{i \in [n]} |\nabla_i g_{R}(x^t)|$  
16: $j \leftarrow \arg\min_{j \in S^t} |x_j|$  
17: $S^{t+1} \leftarrow S^t \cup \{i\} \setminus \{j\}$  
18: $x^{t+1} \leftarrow \arg\min_{\supp(x) \subseteq S^{t+1}} g_{R}(x)$  
19: if $g_{R}(x^t) - g_{R}(x^{t+1}) < \frac{1}{2} (g_{R}(x^t) - \text{opt})$ then  
20: $S^{t+1} \leftarrow S^t$  
21: Sample $i \in R^t$ proportional to $(x_i^t)^2$  
22: $R^{t+1} \leftarrow R^t \setminus \{i\}$  
23: $x^{t+1} \leftarrow \arg\min_{\supp(x) \subseteq S^{t+1}} g_{R^{t+1}}(x)$  
24: end if  
25: end for  
26: return $x^T$

27: end function

**Algorithm 2 ARHT**

1: function ARHT_robust($s$, opt, $\epsilon$, $B$)  
2: function to be minimized $f : \mathbb{R}^n \rightarrow \mathbb{R}$  
3: lower bound on target value $B$  
4: $x^\text{ret} \leftarrow 0$  
5: for $z = 1 \ldots 5 \log \left(6n \log \frac{f(\tilde{0}) - B}{\epsilon}\right)$ do  
6: $x \leftarrow \text{ARHT_core}(s, \text{opt}, \epsilon)$  
7: if $f(x) < f(x^\text{ret})$ then  
8: $x^\text{ret} \leftarrow x$  
9: end if  
10: end for  
11: return $x^\text{ret}$

12: end function

13: function ARHT($s$, $\epsilon$)  
14: function to be minimized $f : \mathbb{R}^n \rightarrow \mathbb{R}$  
15: target sparsity $s$  
16: target error $\epsilon$  
17: $B \leftarrow \min f(x)$  
18: $l \leftarrow B$  
19: $b \leftarrow 0$  
20: $r \leftarrow f(b)$  
21: while $r - l > \epsilon$ do  
22: $m \leftarrow \frac{l + r}{2}$  
23: $x \leftarrow \text{ARHT_robust}(s, m, \epsilon/3, B)$  
24: if $f(x) > m + \epsilon/3$ then  
25: $l \leftarrow m$  
26: end else  
27: $b \leftarrow x$  
28: $r \leftarrow f(x)$  
29: end if  
30: end while  
31: return $b$  
32: end function

---
As the first corollary of the above theorem, we show that it directly implies solution recovery bounds similar to those of \cite{Zhang2011}, while also improving the recovery bound by a constant factor.

**Corollary 3.5 (Solution recovery).** Given a function \( f \) and an (unknown) \( s^* \)-sparse solution \( x^* \), such that the Restricted Gradient Optimal Constant at sparsity level \( s \) is \( \zeta \), i.e.

\[
|\langle \nabla f(x^*), y \rangle| \leq \zeta \|y\|_2
\]

for all \( s \)-sparse \( y \) and as long as

\[
s \geq s^* \max \{4\tilde{\kappa} + 7, 12\tilde{\kappa} + 6\}
\]

Algorithm 2 ensures that

\[
f(x) \leq f(x^*) + \epsilon
\]

and

\[
\|x - x^*\|_2 \leq \frac{\zeta}{\rho^{-}} \left( 1 + \sqrt{1 + 2\epsilon \rho^{-}} \right)
\]

For any \( \theta > 0 \) and \( \epsilon \leq \frac{\zeta^2}{\rho^{-}} \theta(1 + \frac{\theta}{2}) \), this implies that

\[
\|x - x^*\|_2 \leq (2 + \theta) \frac{\zeta}{\rho^{-}}
\]

The next corollary shows that our Theorem 3.3 can be also used to obtain support recovery results under a “Signal-to-Noise” condition given as a lower bound to \( |x^*_{\min}| \).

**Corollary 3.6 (Support recovery).** As long as

\[
s \geq s^* \max \{4\tilde{\kappa} + 7, 12\tilde{\kappa} + 6\}
\]

and \( |x^*_{\min}| > \frac{\epsilon}{\rho^{-}} \), Algorithm 2 with \( \epsilon < -\frac{1}{2\rho^{-}}\zeta^2 + \frac{1}{2\rho^{-}} (x^*_{\min})^2 \) returns a solution \( x \) with support \( S \) such that \( S^* \subseteq S \).

### 3.2. Analysis of Orthogonal Matching Pursuit with Replacement (OMPR)

The OMPR algorithm was first described (under a different name) in \cite{Shalev-Shwartz2010}. It is an extension of OMP but after each iteration some element is removed from \( S^t \) so that the sparsity remains the same. It is described in Algorithm 3.

When \( x \) (with support \( S \)) and \( x^* \) (with support \( S^* \)) are clear from context, we will also define a solution

\[
\bar{x} = \arg\min_{\supp(z) \subseteq S \cup S^*} f(z)
\]

**Algorithm 3** Orthogonal Matching Pursuit with Replacement

1. **function** OMPR(\( s \))
2. \( f \) to be minimized \( f: \mathbb{R}^n \to \mathbb{R} \)
3. output sparsity \( s \)
4. \( S^0 \leftarrow [s] \)
5. \( x^0 \leftarrow \arg\min \{ f(x) \mid \supp(x) \subseteq S^0 \} \)
6. \( t \leftarrow 0 \)
7. while true do
8. \( i \leftarrow \arg\max \{ |\nabla_i f(x^t)| \mid i \in [n] \setminus S^t \} \)
9. \( j \leftarrow \arg\min \{ |x_j^t| \mid j \in S^t \} \)
10. \( S^{t+1} \leftarrow S^t \cup \{i\} \setminus \{j\} \)
11. \( x^{t+1} \leftarrow \arg\min \{ f(x) \mid \supp(x) \subseteq S^{t+1} \} \)
12. if \( f(x^{t+1}) \geq f(x^t) \) then
13. break
14. end if
15. \( t \leftarrow t + 1 \)
16. end while
17. \( T \leftarrow t \)
18. return \( x^T \)
19. end function

Furthermore, we will denote by \( \bar{x}^\star \) the optimal \((s + s^*)\)-sparse solution, i.e.

\[
\bar{x}^\star = \arg\min_{|\supp(z)| \leq s + s^*} f(z)
\]

By definition, for any \( x, x^\star \) the following chain of inequalities holds

\[
\min_{z \in \mathbb{R}^n} f(z) \leq f(\bar{x}^\star) \leq f(\bar{x}) \leq \min \{ f(x), f(x^\star) \}
\]

The following theorem is the main result of this section. Its strength lies in its generality, and various useful corollaries can be directly extracted from it.

**Theorem 3.7.** Given a function \( f \), an (unknown) \( s^* \)-sparse solution \( x^* \), a desired solution sparsity level \( s \), and error parameters \( \epsilon > 0 \) and \( 0 < \theta < 1 \), Algorithm 3 returns an \( s \)-sparse solution \( x \) such that

- If \( \tilde{\kappa} \sqrt{\frac{s}{s^*}} \leq 1 \), then
  \[
f(x) \leq f(x^*) + \epsilon \in O \left( \sqrt{ss^*} \log \frac{f(x^0) - f(x^*)}{\epsilon} \right) \text{ iterations.}
\]
- If \( 1 < \tilde{\kappa} \sqrt{\frac{s}{s^*}} < 2 - \theta \), then
  \[
f(x) \leq f(x^*) + B
\]

where

\[
B = \epsilon + \frac{4(1 - \theta)}{2 - \tilde{\kappa} \sqrt{\frac{s}{s^*}}} \left( f(x^*) - f(\bar{x}^\star) \right)
\]
Sparse Convex Optimization via Adaptively Regularized Hard Thresholding

in $O\left(\frac{\sqrt{\kappa}}{\theta} \log \frac{f(x^0) - f(x^*)}{\epsilon}\right)$ iterations.

The first corollary states that in the “noiseless” case (i.e. when the target solution is globally optimal), the returned solution can reach arbitrarily close to the target solution:

**Corollary 3.8** (Noiseless case). If $\kappa \sqrt{\frac{2}{\epsilon}} < 2$ and $x^*$ is a globally optimal solution, i.e. $f(x^*) = \min_z f(z)$, Algorithm 3 returns a solution with

$$f(x) \leq f(x^*) + \epsilon$$

in $O\left(\frac{\sqrt{\kappa} \sqrt{2}}{2 - \kappa \sqrt{2}} \log \frac{f(x^0) - f(x^*)}{\epsilon}\right)$ iterations.

**Proof.** We apply Theorem 3.7 with $\theta = \frac{1}{2} \left(2 - \kappa \sqrt{\frac{2}{\epsilon}}\right)$.

The following result is in the usual form of sparse recovery results, which provide a bound on $\|x - x^*\|_2$ given a RIP constant upper bound. It provides a tradeoff between the RIP constant and the sparsity of the returned solution.

**Corollary 3.9** ($\ell_2$ solution recovery). Given any parameters $\epsilon > 0$ and $0 < \theta < 1$, the returned solution $x$ of Algorithm 3 will satisfy

$$\|x - x^*\|_2^2 \leq \epsilon + C \left(f(x) - \min_z f(z)\right)$$

as long as

$$\delta_{s+s^*} < \frac{(2 - \theta) \sqrt{\frac{2}{\epsilon}} - 1}{(2 - \theta) \sqrt{\frac{2}{\epsilon}} + 1}$$

where $C$ is a constant that depends only on $\theta$, $\delta_{s+s^*}$, and $\frac{s^*}{s}$.

In particular, for $s = s^*$, the above lemma implies recovery under the condition $\delta_{2s^*} < \frac{1}{2}$.

4. Experiments

**Remark 4.1.** More details about the datasets used for evaluation can be found in the Supplementary material.

In this section we evaluate the training performance of different algorithms in the tasks of Linear Regression and Logistic Regression. More specifically, for each algorithm we are interested in how the loss over the training set (the quality of the solution) evolves as a function of the the sparsity of the solution, i.e. the number of non-zeros.

The algorithms that we will consider are **LASSO**, **Orthogonal Matching Pursuit (OMP)**, **Orthogonal Matching Pursuit with Replacement (OMPR)**, **Adaptively Regularized Hard Thresholding (ARHT)** (Algorithm 2), and **Exhaustive Local Search** (which is a version of OMPR that examines all possible insertions/removals in each iteration). We run our experiments on publicly available regression and binary classification datasets, out of which we have presented those on which the algorithms have significantly different performance between each other. In some of the other datasets that we tested, we observed that all algorithms had similar performance. The results are presented in Figure 1 and Figure 2. Another relevant class of algorithms that we considered was $\ell_p$ Approximate Message Passing algorithms (Donoho et al., 2009; Zheng et al., 2017). Brief experiments showed its performance in terms of sparsity for $p \leq 0.5$ to be promising (on par with OMPR and ARHT although these had much faster runtimes), however a detailed comparison is left for future work.

In both types of objectives (linear and logistic) we include an intercept term, which is present in all solutions (i.e. it is always counted as $+1$ in the sparsity of the solution). For consistency, all greedy algorithms (OMPR, ARHT, Exhaustive Local Search) are initialized with the OMP solution of the same sparsity.

The experiments make it clear that Exhaustive Local Search outperforms the other algorithms. However, ARHT also has promising performance and it might be preferred because of better computational efficiency. As a general conclusion, however, both Exhaustive Local Search and ARHT offer an advantage compared to OMP and OMPR. As a limitation, we observe that ARHT has inconsistent performance in some cases, oscillating between the Exhaustive Local Search and OMPR solutions.

In $O\left(\frac{\sqrt{\kappa}}{\theta} \log \frac{f(x^0) - f(x^*)}{\epsilon}\right)$ iterations.

The following result is in the usual form of sparse recovery results, which provide a bound on $\|x - x^*\|_2$ given a RIP constant upper bound. It provides a tradeoff between the RIP constant and the sparsity of the returned solution.

**Corollary 3.9** ($\ell_2$ solution recovery). Given any parameters $\epsilon > 0$ and $0 < \theta < 1$, the returned solution $x$ of Algorithm 3 will satisfy

$$\|x - x^*\|_2^2 \leq \epsilon + C \left(f(x) - \min_z f(z)\right)$$

as long as

$$\delta_{s+s^*} < \frac{(2 - \theta) \sqrt{\frac{2}{\epsilon}} - 1}{(2 - \theta) \sqrt{\frac{2}{\epsilon}} + 1}$$

where $C$ is a constant that depends only on $\theta$, $\delta_{s+s^*}$, and $\frac{s^*}{s}$.

In particular, for $s = s^*$, the above lemma implies recovery under the condition $\delta_{2s^*} < \frac{1}{2}$.

4. Experiments

**Remark 4.1.** More details about the datasets used for evaluation can be found in the Supplementary material.

In this section we evaluate the training performance of different algorithms in the tasks of Linear Regression and Logistic Regression. More specifically, for each algorithm we are interested in how the loss over the training set (the quality of the solution) evolves as a function of the the sparsity of the solution, i.e. the number of non-zeros.

The algorithms that we will consider are **LASSO**, **Orthogonal Matching Pursuit (OMP)**, **Orthogonal Matching Pursuit with Replacement (OMPR)**, **Adaptively Regularized Hard Thresholding (ARHT)** (Algorithm 2), and **Exhaustive Local Search** (which is a version of OMPR that examines all possible insertions/removals in each iteration). We run our experiments on publicly available regression and binary classification datasets, out of which we have presented those on which the algorithms have significantly different performance between each other. In some of the other datasets that we tested, we observed that all algorithms had similar performance. The results are presented in Figure 1 and Figure 2. Another relevant class of algorithms that we considered was $\ell_p$ Approximate Message Passing algorithms (Donoho et al., 2009; Zheng et al., 2017). Brief experiments showed its performance in terms of sparsity for $p \leq 0.5$ to be promising (on par with OMPR and ARHT although these had much faster runtimes), however a detailed comparison is left for future work.

In both types of objectives (linear and logistic) we include an intercept term, which is present in all solutions (i.e. it is always counted as $+1$ in the sparsity of the solution). For consistency, all greedy algorithms (OMPR, ARHT, Exhaustive Local Search) are initialized with the OMP solution of the same sparsity.

The experiments make it clear that Exhaustive Local Search outperforms the other algorithms. However, ARHT also has promising performance and it might be preferred because of better computational efficiency. As a general conclusion, however, both Exhaustive Local Search and ARHT offer an advantage compared to OMP and OMPR. As a limitation, we observe that ARHT has inconsistent performance in some cases, oscillating between the Exhaustive Local Search and OMPR solutions.
Figure 1. Comparison of different algorithms in the Regression datasets cal_housing and year using the Linear Regression loss.

Figure 2. Comparison of different algorithms in the Binary classification datasets census and kddcup04_bio using the Logistic Regression loss.

References


Blumensath, T. and Davies, M. E. On the difference between orthogonal matching pursuit and orthogonal least squares. 2007.


