Globally-convergent Iteratively Reweighted Least Squares for Robust Regression Problems

Bhaskar Mukhoty* **Govind Gopakumar***[‡] **Prateek Jain**[†] **Purushottam Kar***
{bhaskarm,govindg,purushot}@cse.iitk.ac.in prajain@microsoft.com
*IIT Kanpur [†]Microsoft Research India

Abstract

We provide the first global model recovery results for the IRLS (iteratively reweighted least squares) heuristic for robust regression problems. IRLS is known to offer excellent performance, despite bad initializations and data corruption, for several parameter estimation problems. Existing analyses of IRLS frequently require careful initialization, thus offering only local convergence guarantees. We remedy this by proposing augmentations to the basic IRLS routine that not only offer guaranteed global recovery, but in practice also outperform state-of-the-art algorithms for robust regression. Our routines are more immune to hyperparameter misspecification in basic regression tasks, as well as applied tasks such as linear-armed bandit problems. Our theoretical analyses rely on a novel extension of the notions of strong convexity and smoothness to *weighted strong convexity and smoothness*, and establishing that sub-Gaussian designs offer bounded *weighted condition numbers*. These notions may be useful in analyzing other algorithms as well.

1 Introduction

Suppose there exists an unknown gold model $\mathbf{w}^*$ and we are given n data points $(\mathbf{x}_i, y_i)_{i=1}^n$ with d-dimensional covariates $\mathbf{x}_i \in \mathbb{R}^d$ and the real-valued responses y_i generated as $y_i = \mathbf{x}_i^\top \mathbf{w}^*$. However, for an unknown set of $k < n$ data points $i_1, \ldots i_k$, the responses get corrupted i.e. we instead receive $y_{i_j} = \mathbf{x}_{i_j}^\top \mathbf{w}^* + b_{i_j}$ where $b_{i_j} \in \mathbb{R}$ is the corruption. Given the complete set of clean and corrupted data points $(\mathbf{x}_i, y_i)_{i=1}^n$, can we recover the gold model $\mathbf{w}^*$?

[‡]Work done as a master's student at IIT Kanpur.

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. Copyright 2019 by the author(s).

This is the classical robust regression problem that has become increasingly relevant to machine learning and statistical estimation techniques which frequently encounter situations where data is not trustworthy. Works exist in settings where test data is corrupted in order to fool a model that was learnt on clean data [17], as well as the more challenging setting, on which we focus, where the training data presented to the algorithm is itself corrupted [9, 11, 16].

We will seek to offer reliable model recovery despite the presence of (possibly maliciously) corrupted data in the training set. Settings which present corrupted data to learning algorithms include relatively innocuous instances of erasures and missing data, improperly or mistakenly attributed data, transient or temporary changes in user-behavior patterns, as well as deliberate and malicious attempts to derail recommendation systems and other decision-making systems using malware, click-bots and other fraudulent techniques.

Despite being a well established field, given the early seminal contributions of Huber [18] and Tukey [27], robust statistics and algorithms have received renewed interest given the threat to modern machine learning techniques. Of the several techniques that have been proposed for robust learning problems, one heuristic, namely the iteratively reweighted least squares (IRLS), remains a practitioner's favorite owing to its ease of use and excellent performance. The IRLS technique has been effectively adapted to several problems, including sparse recovery, and robust regression. The work of [26] shows that certain biological dynamical systems can be modeled upon the IRLS principle as well.

Our Contributions We offer several advances in the understanding and application of the IRLS method. In particular, we provide the first global model recovery guarantee for IRLS for robust regression - our contributions are distinguished in the context of existing analyses for IRLS in §2. We also propose algorithmic augmentations, in particular a fast gradient-based variant, to the basic IRLS heuristic which offer superior performance compared to existing state-of-the-art robust algorithms in terms of speed, as well as resilience to misspecified hyperparameters. We demonstrate this in the standard linear regression setting, as well as an applied setting, namely linear-armed bandits.

2 Related Work

Two lines of work directly relate to our contributions: 1) robust algorithms for regression and other learning problems, and 2) works that analyze (variants of) the IRLS heuristic in various settings. We review both, as well as distinguish our contributions, below.

Robust Learning Algorithms: Work on robust statistics dates back several decades [18, 27] and is too vast to be reviewed in detail. Recent years have seen interest in scalable algorithms for classification [16], principal component analysis [9], and moment estimation [14]. Within the specific problem of robust regression, two broad lines of work exist:

Covariate (feature) corruption: Results in this setting usually either give only weak guarantees, or else severely constrain data. e.g., [11, 24] allow only a $\mathcal{O}\left(1/\sqrt{d}\right)$ fraction of data to be corrupted, d being the ambient dimensionality, whereas [15, 21] only admit covariates drawn from a Gaussian distribution.

Response (label) corruption: Variants within this setting arise based on the power of the adversary introducing the corruptions, the fraction of data points that can be corrupted, restrictions on the choice of covariates, and scalability of the algorithms. Table 1 summarizes these traits for a selection of algorithms. We refer the reader to [6, 15] for other references.

IRLS Variants and Analyses: The IRLS heuristic has been successfully applied to several problems including sparse recovery [4, 13], facility location problems [8] (via the Weiszfeld procedure), and optimizing various robust cost functions, such as the L_q and Huber loss functions [2, 7, 12, 25].

Some of these works are not directly relevant to robust regression as they either operate with uncorrupted data [8], or else assume that the noise is Gaussian [4, 13]. Convergence guarantees for IRLS are common in these benign settings. To handle adversarial corruptions, it is common to use IRLS to optimize a *robust cost function* F such as L_q or Huber loss, in the anticipation that the model so obtained, say $\hat{\mathbf{w}} = \arg\min F(\mathbf{w}; \{(\mathbf{x}_i, y_i)\})$, will ensure $\hat{\mathbf{w}} \approx \mathbf{w}^*$.

However, none of these works actually ensure such a result i.e. $\hat{\mathbf{w}} \approx \mathbf{w}^*$. Some works [7, 12, 25] operate with cost functions that are convex (e.g. L_q for $q \in [1, 2]$) and simply show that IRLS approaches small cost function values. Other approaches [2] do work with non-convex cost functions, but then offer only monotonicity guarantees and no global convergence guarantees.

We bridge this gap by presenting a much stronger analysis of IRLS that guarantees *global recovery* of the gold model $\mathbf{w}^*$ under mild conditions. Key to our proof technique is a novel concept that extends the basic notions of strong convexity and strong smoothness to *weighted* versions of the same, as well as a guarantee that Gaussian and sub-Gaussian designs have bounded *weighted condition numbers*. These results may be of independent interest in analyzing other algorithms.

3 Notation

Bold lower-case Latin letters $\mathbf{x}, \mathbf{y}$ denote vectors. $\mathbf{x}_i$ denotes the i^{th} coordinate of the vector $\mathbf{x}$. Upper case Latin letters A, X denote matrices. For a vector $\mathbf{v} \in \mathbb{R}^n$ and set $S \subset [n]$, $\mathbf{v}_S$ denotes the vector with $(\mathbf{v}_S)_i = \mathbf{v}_i$ for $i \in S$ and $(\mathbf{v}_S)_j = 0$ for $j \notin S$. Similarly, for any matrix $A \in \mathbb{R}^{d \times n}$ and any set $S \subset [n]$, A_S denotes the matrix in which columns $i \in S$ in A_S are identical to those in A and columns $j \notin S$ are filled with zeros.

$\lambda_{\min}(X)$ and $\lambda_{\max}(X)$ denote, respectively, the smallest and largest eigenvalues of a square symmetric matrix X. $\mathcal{B}_2(\mathbf{v}, r) := \left\{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{v}\|_2 \le r\right\}$ denotes the ball of radius r centered at $\mathbf{v}$. S^{d-1} denotes the surface of the unit sphere in d dimensions. We use the shorthand $\mathcal{B}_2(r) := \mathcal{B}_2(\mathbf{0}, r)$.

4 Problem Formulation

Given n data points $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, let $R_X := \max_{i \in [n]} \|\mathbf{x}_i\|_2$ be the maximum Euclidean length of any covariate, $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ be the covariate matrix, and $\mathbf{y} = [y_1, \ldots, y_n]^\top \in \mathbb{R}^n$ the response vector. Assume that the covariates are generated as $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim \mathcal{D}$ from an unknown distribution $\mathcal{D}$ with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and sub-Gaussian norm [28] $\|\mathcal{D}\|_{\Psi_2} \le R$. $\mathbf{w}^* \in \mathbb{R}^d$ will be the gold model with $R_W := \|\mathbf{w}^*\|_2$.

Noise Model: Given the data covariates and the gold model, the responses are generated as $\mathbf{y} = X^\top \mathbf{w}^* + \mathbf{b}$ where $\mathbf{b} = [b_1, \ldots, b_n]$ is the vector of corruptions. We make the standard assumption that $\|\mathbf{b}\|_0 \le \alpha \cdot n$. Let $B := \text{supp}(\mathbf{b})$ denote the "bad" points which suffer corruption i.e. $\mathbf{b}_j \neq 0$ for $j \in B$ (note that $|B| \le \alpha \cdot n$) and $G = [n] \backslash B$ denote the "good" points where $\mathbf{b}_i = 0$ and thus $y_i = \mathbf{x}_i^\top \mathbf{w}^*$ for $i \in G$. To avoid clutter, we abuse notation to denote $G := |G|$ and $B := |B|$. The largest value of the corruption fraction α that an algorithm can tolerate is known as its *breakdown point*.

Table 1: Algorithms for the Robust Regression problem (corrupted responses). †Please see §4 for details. Algorithms able to tolerate adaptive (as opposed to oblivious) adversaries are more resilient. A more robust algorithm can handle larger α. Sub-Gaussian covariates offer a much more flexible model than (isotropic) Gaussian covariates.

Paper	Adversary Model†	Breakdown point†	Covariate Model	Technique
Bhatia et. al. 2015 [6]	Adaptive	$\alpha \geq \Omega(1)$	sub-Gaussian	Hard Thresholding (fast)
Chen & Dalalyan 2010 [10]	Adaptive	$\alpha \geq \Omega(1)$	sub-Gaussian	SOCP (slow)
Wright & Ma 2010 [29]	Oblivious	$\alpha \to 1$	Isotropic Gaussian	L_1 regularization (slow)
This Paper	**Adaptive**	$\boldsymbol{\alpha \geq \Omega(1)}$	**sub-Gaussian**	**Reweighting (fast)**

Adversary Model: We will work with a partially adaptive adversary which is compelled to choose locations of the corruptions $\text{supp}(\mathbf{b}) = B$ before any data covariates have been generated or $\mathbf{w}^*$ is revealed. However, the adversary may fill in the corruption values at those locations with knowledge of $\mathbf{w}^*$ and X. Our results can be extended to a *fully adaptive adversary* that choose $\text{supp}(\mathbf{b})$ after looking at $\mathbf{w}^*$ and X as well, but at a cost of a smaller breakdown point α.

Key to our analyses are the notions of *weighted strong convexity and smoothness* which we define below. These definitions reflect the fact that IRLS solves *weighted* regression problems iteratively.

Definition 1 (WSC/WSS). *We say that a covariate matrix $X \in \mathbb{R}^{d\times n}$ offers* weighted strong convexity *(WSC) at level λ_S (resp.* weighted strong smoothness *(WSS) at level Λ_S), with respect to a diagonal weight matrix $S = diag(\mathbf{s}) \in \mathbb{R}^{n\times n}$ where $\mathbf{s}_i \geq 0, i \in [n]$, if*

$$\lambda_S \leq \lambda_{\min}(XSX^\top) \leq \lambda_{\max}(XSX^\top) \leq \Lambda_S$$

5 Proposed Methods

IRLS solves the robust regression problem by repeatedly alternating between the following two steps

1. **Reweighing**: Given a model $\hat{\mathbf{w}}$, assign every data point a weight s_i inversely proportional to its residual w.r.t. $\hat{\mathbf{w}}$ i.e. set $\mathbf{s}_i = \frac{1}{|\mathbf{x}_i^\top\hat{\mathbf{w}} - y_i|}$.
2. **Weighted Least Squares**: Solve a weighted least squares problem $\min_{\mathbf{w}} \sum_{i=1}^n \mathbf{s}_i(y_i - \mathbf{x}_i^\top\mathbf{w})^2$ with above weights to obtain a new model $\mathbf{w}^+ = (XSX^\top)^{-1}XS\mathbf{y}$ where $S = \text{diag}(\mathbf{s})$.

The intuition behind this procedure is that corrupted points are likely to suffer large residuals and hence get downweighted. Given that this procedure runs the risk of divide-by-zero errors and numerical precision issues, it is common to truncate weights by employing a *truncation parameter* M while assigning weights[1] to the points i.e. $\mathbf{s}_i = \min\left\{\frac{1}{|\mathbf{x}_i^\top\hat{\mathbf{w}} - y_i|}, M\right\}$. However, it is suboptimal to rely on any single truncation value M. To see why, take a hypothetical example where the adversary introduces corruptions using a *fake model* $\tilde{\mathbf{w}}$ as $b_i = \mathbf{x}_i^\top(\tilde{\mathbf{w}} - \mathbf{w}^*)$ (i.e. $y_i = \mathbf{x}_i^\top\tilde{\mathbf{w}}$) for all $i \in B$.

Situation 1: If we set M to a small value (aggressive truncation), then no data point can ever hope to get a large weight. However, convergence to $\mathbf{w}^*$ is assured only when points in G receive really large weights in comparison to points in B. Setting a small value of M thus prevents IRLS from recovering $\mathbf{w}^*$ accurately.

Situation 2: If we always use a large value of M (lax truncation) and are unlucky enough to initialize IRLS close to $\tilde{\mathbf{w}}$, then points in the set B will initially have very small residuals, hence receive large weights (which the large value of M will allow) whereas points in the set G will receive comparatively smaller weights. This will cause IRLS to gravitate towards $\tilde{\mathbf{w}}$. This example precludes any hope of a global convergence guarantee and forces us to do careful initialization.

The above limitations of IRLS are well corroborated by experiments (see §8). To remedy this, we propose the STIR algorithm in Algorithm 1. STIR executes IRLS, but in *stages*, with initial stages employing aggressive truncation with a small value of M and later stages successively relaxing the truncation.

The advantage of the above augmentation is that even if we have an unfortunate initialization, e.g. we start at $\tilde{\mathbf{w}}$ itself, the (initially) aggressive truncation will prevent bad points from getting large weights whereas good points, being in majority, even though receiving relatively smaller weights, will still prevent STIR from latching onto $\tilde{\mathbf{w}}$ and hopefully attract the procedure towards the gold model $\mathbf{w}^*$. Subsequent stages, where truncation is relaxed, will allow good points to be given large weights, thus differentiating them from bad points. This would force STIR towards $\mathbf{w}^*$.

Algorithm 2 presents STIR-GD, a gradient version of STIR, that replaces weighted least squares by a much cheaper gradient step. This benefits large datasets, where solving weighted least squares repeatedly may be prohibitive. We note that although *stagewise* IRLS procedures have been proposed in literature [7], previous works neither give model recovery guarantees, nor offer scalable gradient versions of IRLS.

[1]Literature often cites a *regularization* procedure that sets $\mathbf{s}_i = \frac{1}{\max\{|\mathbf{x}_i^\top\hat{\mathbf{w}} - y_i|, \delta\}}$ given a parameter δ. Setting $\delta = \frac{1}{M}$ shows truncation to be equivalent to regularization.

Algorithm 1 STIR- Stagewise-Truncated IRLS

```
Input: Data $X, \mathbf{y}$, initial truncation $M_1$, increment $\eta > 1$
Output: A model $\mathbf{w}$
 1: $\mathbf{w}^1 \leftarrow \mathbf{0}$
 2: for $T = 1, 2, \ldots, K-1$ do
 3:     $\mathbf{w}^{T,1} \leftarrow \mathbf{w}^T$
 4:     $t \leftarrow 1$
 5:     while $\left\|\mathbf{w}^{T,t+1} - \mathbf{w}^{T,t}\right\|_2 > \frac{2}{\eta M_T}$ do
 6:         $\mathbf{r}^t \leftarrow X^\top \mathbf{w}^{t,1} - \mathbf{y}$
 7:         $S^t \leftarrow \mathrm{diag}(\mathbf{s}^t)$,    $\mathbf{s}_i^t \leftarrow \min\left\{\frac{1}{|\mathbf{r}_i^t|}, M_T\right\}$
 8:         $\mathbf{w}^{T,t+1} \leftarrow (XS^tX^\top)^{-1}XS^t\mathbf{y}$
 9:         $t \leftarrow t + 1$
10:     end while
11:     $\mathbf{w}^{T+1} \leftarrow \mathbf{w}^{T,t+1}$
12:     $M_{T+1} \leftarrow \eta \cdot M_T$
13: end for
14: return $\mathbf{w}^K$
```

Algorithm 2 STIR-GD: STIR-Gradient Descent

```
Input: Data $X, \mathbf{y}$, initial truncation $M_1$, increment $\eta > 1$,
    step length $C$
Output: A model $\mathbf{w}$
 8:         $\mathbf{w}^{T,t+1} \leftarrow \mathbf{w}^{T,t} - \frac{2C}{M_T n} \cdot XS^t\mathbf{r}^t$
        //Rest of steps 1-14 remain same as in STIR
```

6 IRLS is Majorization-minimization on a Scaled Huber Loss

Before presenting a convergence analysis for STIR, we point out a curious link between IRLS, STIR and the Huber loss function. We note that our observation may be folklore. The Huber loss is widely used in robust regression applications [2, 7, 12, 25], particularly those used in situations with heavy tailed noise.

$$h_\epsilon(x) = \begin{cases} \frac{1}{2}x^2 & |x| \leq \epsilon \\ \epsilon\,|x| - \frac{1}{2}\epsilon^2 & |x| > \epsilon \end{cases}$$

The function smoothly transitions from quadratic behavior close to the origin, to linear far from the origin. Now consider the following loss function

$$f_\epsilon(x) = \begin{cases} \frac{1}{2}\left(\frac{x^2}{\epsilon} + \epsilon\right) & |x| \leq \epsilon \\ |x| & |x| > \epsilon \end{cases}$$

It is easily seen that $f_\epsilon(x) = \frac{h_\epsilon(x)}{\epsilon} + \frac{\epsilon}{2}$ and thus, $f_\epsilon()$ is simply a scaled (and translated) version of the Huber loss function, as well as that $|x| \leq f_\epsilon(x) \leq |x| + \frac{\epsilon}{2}$. Now, for any $a \in \mathbb{R}, \epsilon > 0$, consider the function

$$g_\epsilon(x; a) := \frac{1}{2}\left(\frac{x^2}{\max\{|a|, \epsilon\}} + \max\{|a|, \epsilon\}\right)$$

Given a model $\mathbf{w}^0$ and data $(\mathbf{x}_i, y_i)_{i=1}^n$, denote

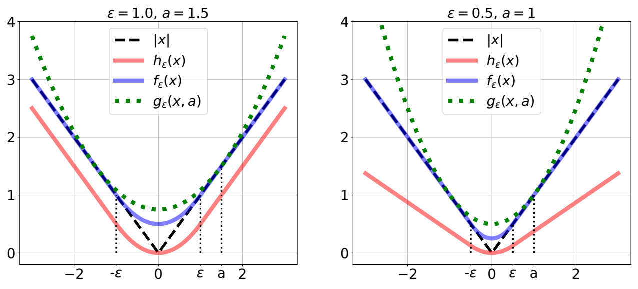


Figure 1: A depiction of Huber $h_\epsilon()$, scaled Huber $f_\epsilon()$ loss functions, and its majorizer $g_\epsilon()$ for various ϵ.

$$\ell_\epsilon(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^n f_\epsilon\left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)$$

$$\wp_\epsilon(\mathbf{w}; \mathbf{w}^0) := \sum_{i=1}^n g_\epsilon\left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i; \langle \mathbf{w}^0, \mathbf{x}_i\rangle - y_i\right)$$

The following observations are key (see Appendix A).

1. $\wp_\epsilon(\cdot; \mathbf{w}^0)$ is a majorizer for $\ell_\epsilon(\cdot)$ at $\mathbf{w}^0, \forall\epsilon > 0$ i.e. $\wp_\epsilon(\mathbf{w}; \mathbf{w}^0) \geq \ell_\epsilon(\mathbf{w}), \forall\mathbf{w}$ but $\wp_\epsilon(\mathbf{w}^0; \mathbf{w}^0) = \ell_\epsilon(\mathbf{w}^0)$
2. If the current model is $\mathbf{w}^0$ then M-truncated IRLS minimizes $\wp_{\frac{1}{M}}(\mathbf{w}; \mathbf{w}^0)$ to obtain the next model.
3. $\nabla\wp_\epsilon(\mathbf{w}^0; \mathbf{w}^0) = \nabla\ell_\epsilon(\mathbf{w}^0)$.

Thus, IRLS can be seen as performing majorization-minimization [23] on the scaled Huber loss $\ell_\epsilon(\cdot)$. The reweighing step effectively constructs the majorizer function $\wp_\epsilon(\cdot, \mathbf{w}^0)$ over which the least squares step then performs minimization. Point 3 above shows that STIR-GD can be effectively seen as performing gradient descent with respect to $\ell_\epsilon(\mathbf{w}^0)$.

This also allows us to interpret the stages of STIR as using scaled Huber losses with successively smaller values of ϵ (point 2 above shows that STIR sets $\epsilon = \frac{1}{M}$). Note that in the limit $\epsilon \to 0$, $\ell_\epsilon(\cdot)$ approaches the absolute error function, and thus, in the limit $M \to \infty$, STIR ends up optimizing the absolute error function. STIR-GD can be seen as simply replacing the minimization steps with a gradient descent step.

7 Convergence Analysis

In this section, we establish that both STIR and STIR-GD enjoy a linear rate of convergence, as well as a breakdown point $\alpha \geq \Omega(1)$. Theorem 1 summarizes the results. It is notable that STIR and STIR-GD offer a breakdown point of greater than $\frac{1}{5.25}$ (for Gaussian covariates – see below for details), which is far superior to those offered by recent works such as [6, 5] which offer breakdown points of $\approx \frac{1}{60}$ and $\frac{1}{10000}$ respectively (again for Gaussian covariates).

Theorem 1. *Suppose we have n data points with the covariates $\mathbf{x}_i$ sampled from a sub-Gaussian distribution $\mathcal{D}$ and an α fraction of the data points are corrupted. If STIR (or STIR-GD) is initialized at an (arbitrary) point $\mathbf{w}^0$, with an initial truncation that satisfies $M_1 \leq \frac{1}{\|\mathbf{w}^0 - \mathbf{w}^*\|_2}$, and executed with an increment $\eta > 1$ such that we have $\alpha \leq \frac{c}{2.88\eta + c}$, where $c > 0$ is a constant that depends only on $\mathcal{D}$, then for any $\epsilon > 0$, with probability at least $1 - \exp(-\tilde{\Omega}(n))$, after $K = \mathcal{O}\left(\log \frac{1}{M_1 \epsilon}\right)$ stages, we must have $\left\|\mathbf{w}^K - \mathbf{w}^*\right\|_2 \leq \epsilon$. Moreover, each stage consists of only $\mathcal{O}(1)$ iterations.*

Global Convergence Note that the above result allows initialization at any location $\mathbf{w}^0$, so long as the accompanying value M_1 is small enough i.e. $M_1 \leq \frac{1}{\|\mathbf{w}^0 - \mathbf{w}^*\|_2}$ which can be ensured using a simple binary search (see §8 for details on parameter setting). In particular, if an estimated upper-bound $\|\mathbf{w}^*\|_2 \leq W$ is available, then we can set $\mathbf{w}^0 = \mathbf{0}$ and set $M_1 = \frac{1}{W}$.

Given this parameter convergence result, we can also establish that STIR and STIR-GD offer linear convergence guarantees with respect to the Huber and absolute loss functions as well. We refer the reader to Appendix C.2 for details.

Breakdown Point Both STIR and STIR-GD enjoy a breakdown point of $\alpha \leq \frac{c}{2.88\eta + c}$ where η is chosen by us and c is a distribution dependent constant. Bounds on this constant are established for several interesting distributions in Appendix D.1. In particular, for the Gaussian distribution $\mathcal{N}(\mathbf{0}, I_d)$, we have $c \geq 0.68$ which, for values of $\eta \to 1$, endow STIR and STIR-GD with a breakdown point of greater than $\frac{1}{5.25}$.

7.1 Proof Outline - the Peeling Strategy

Given the stage-wise nature of our algorithms STIR and STIR-GD, we employ a *peeling*-based proof strategy that is a departure from the techniques used by previous results such as [6, 10, 29].

Our proof partitions the model space into *annular peels* centered at the gold model $\mathbf{w}^*$ (see Figure 2). The outermost peel has a radius of $\frac{1}{M_1}$, and successive inner peels have radii that are an η factor smaller i.e. the subsequent peels have radii $\frac{1}{\eta M_1}, \frac{1}{\eta^2 M_1}, \frac{1}{\eta^3 M_1}, \ldots$. Note that by setting $M_1 \leq \frac{1}{\|\mathbf{w}^0 - \mathbf{w}^*\|_2}$, STIR is guaranteed to reside inside the outermost peel in the beginning.

We then inductively show (see Lemmata 8 and 9) that once we are inside a certain peel, say $\|\mathbf{w} - \mathbf{w}^*\|_2 \leq \frac{1}{\eta^K M}$, and if the WSC/WSS properties hold with appropriate constants (see Appendix D), then if we execute $(\eta^K M)$-truncated IRLS for a constant number of iterations, we are guaranteed to obtain a model, say $\mathbf{w}^+$, that ensures $\|\mathbf{w}^+ - \mathbf{w}^*\|_2 \leq \frac{1}{\eta^{K+1} M}$.

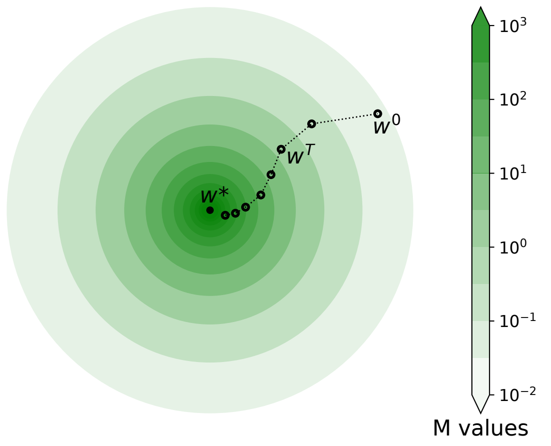


Figure 2: A depiction of the peeling process. The STIR procedure starts off far away from $\mathbf{w}^*$ and using a small value of M. In successive stages, it enters closer peels around $\mathbf{w}^*$ and also begins using larger values of M.

This implies that we have entered the next inner peel. We can now set the truncation level to $\eta^{K+1}M$ and continue the process. Note that this is exactly the algorithmic step performed by STIR/STIR-GD (see Algorithm 1, line 12) to start a new stage. Due to lack of space, all complete proofs are given in the appendices.

7.2 Establishing WSC/WSS

A central result required for the peeling strategy to work, is ensuring that our covariates satisfy the WSC/WSS properties (that we introduced in §4) with respect to the weights assigned to data points by the STIR and STIR-GD algorithms. We show that for covariates drawn from sub-Gaussian distributions, this is indeed true (see Appendix D).

The use of such *design properties* is quite common in literature e.g., restricted strong convexity/smoothness (RSC/RSS) [13] in sparse recovery, and subset strong convexity/smoothness (SSC/SSS) [6] in robust regression. It is also common to use results on extremal singular values of random matrices [28], to show that sub-Gaussian covariates satisfy RSC/RSS [3] and SSC/SSS [6], with high probability.

However, doing so in our case is not as straightforward. The reason for this is that whereas the RSC/RSS and SSC/SSS properties are defined purely in terms of the data covariates, the WSC/WSS properties also incorporate data weights. Moreover, these weights are neither constant, nor independent of the data, but rather are assigned and repeatedly updated in a stage-wise manner by an algorithm such as IRLS or STIR.

Since our proofs will require the WSC/WSS properties to hold with respect to *all* weight assignments made during the entire execution of the algorithms, a direct application of classical techniques [28] fails. Such techniques could have succeeded only if the data weights were to be constant or else independent of the data.

To overcome this challenge, we establish WSC/WSS properties for sub-Gaussian covariates in a *peel-wise* manner using a careful uniform convergence bound. The number of peels is no more than $\mathcal{O}\left(\log\frac{1}{\epsilon}\right)$ since each peel corresponds to a stage of the algorithm and $\mathcal{O}\left(\log\frac{1}{\epsilon}\right)$ is the number of stages required to achieve an ϵ-accurate solution (see Theorem 1), which then allows us to take a union bound over all peels.

Within each peel, a careful uniform convergence bound is employed over all models within that peel in order to establish WSC/WSS. Note that our results present a novel extension of the existing notions of SSC/SSS since we can recover SSC/SSS as a special case of WSC/WSS where the weights are simply zero or unity.

7.3 Corruptions and Dense Noise

So far we have looked at an idealized setting where the responses are either completely clean $y_i = \mathbf{x}_i^\top \mathbf{w}^*$ for $i \in G$ or else corrupted $y_j = \mathbf{x}_j^\top \mathbf{w}^* + \mathbf{b}_j$ for $j \in B$. We now look at a more realistic setting where even the "good" points experience sub-Gaussian noise. We will now assume that our data is generated as $\mathbf{y} = X^\top \mathbf{w}^* + \mathbf{b} + \boldsymbol{\epsilon}$ where, as before $\|\mathbf{b}\|_0 \le \alpha \cdot n$, but we additionally have $\boldsymbol{\epsilon} \sim \mathcal{D}_\varepsilon$ where $\mathcal{D}_\varepsilon$ is a σ-sub-Gaussian distribution with zero mean and real support [2].

We will denote $B := \text{supp}(\mathbf{b})$ and $G := [n] \setminus B$, as before. Our covariates will continue to be sampled from an R-sub-Gaussian distribution $\mathcal{D}$ with support over $\mathbb{R}^d$. Even in this setting, we can ensure a model recovery result with a linear rate of convergence.

Theorem 2. *Suppose we have n data points with the covariates $\mathbf{x}_i$ sampled from a sub-Gaussian distribution $\mathcal{D}$ and an α fraction of the data points are corrupted with the rest subjected to sub-Gaussian noise sampled from a distribution $\mathcal{D}_\varepsilon$ with sub-Gaussian norm σ. If STIR (or STIR-GD) is initialized at an (arbitrary) point $\mathbf{w}^0$, with an initial truncation that satisfies $M_1 \le \frac{1}{\|\mathbf{w}^0 - \mathbf{w}^*\|_2}$, and executed with an increment $\eta > 1$ such that we have $\alpha \le \frac{c_\varepsilon}{5.85\eta + c_\varepsilon}$, where $c_\varepsilon > 0$ is a constant that depends only on the distributions $\mathcal{D}$ and $\mathcal{D}_\varepsilon$, then with probability at least $1 - \exp(-\tilde{\Omega}(n))$, after $K = \mathcal{O}\left(\log\frac{1}{M_1\sigma}\right)$ stages, each of which has only $\mathcal{O}(1)$ iterations, we must have $\left\|\mathbf{w}^K - \mathbf{w}^*\right\|_2 \le \mathcal{O}(\sigma)$.*

We refer the reader to Appendix E for the full proof.

[2]We can tolerate noise with non-zero mean as well, by using a simple pairing trick which has a side effect of at most doubling the corruption rate α

Global Convergence This result also allows arbitrary initialization so long as we set $M_1 \le \frac{1}{\|\mathbf{w}^0 - \mathbf{w}^*\|_2}$. However, note that this result only guarantees a convergence to $\left\|\mathbf{w}^{K,1} - \mathbf{w}^*\right\|_2 \le \mathcal{O}(\sigma)$ and thus, does not ensure a consistent solution. We refer the reader to the proof of Theorem 2 in Appendix E for a discussion on this result. We also note that our results or our algorithms, do not require the knowledge of the noise parameter σ.

Breakdown Point For Gaussian covariates i.e. $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$, Gaussian noise i.e. $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \sigma^2)$, we have $c \ge 0.52$ (see Appendix E), and for $\eta \to 1$ this gives STIR and STIR-GD with a breakdown point of $\frac{1}{12.25}$.

8 Experiments

In this section, we report results of a variety of experiments comparing STIR and STIR-GD to other robust learning algorithms. These experiments were performed over two learning settings, namely robust linear regression and robust linear-armed bandit problems.

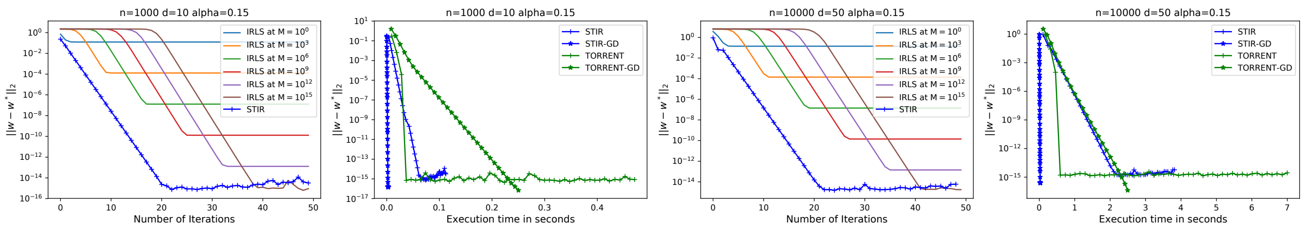
Parameter and Adversary Setting Algorithms considered in this section require only scalar parameters to be specified (α for TORRENT, step length for TORRENT-GD, η and M_1 for STIR, and step length C for STIR-GD), all which were tuned via a fine grid search using a held-out validation set. In particular, a binary search was found to suffice for setting M_1. For all experiments, the adversary was made to introduce corruptions using a fake model as described in §5. All algorithms were initialized at the fake model itself to test their behavior under adversarial initialization.

8.1 Robust Regression Experiments

We executed STIR and STIR-GD on linear regression problems with response corruption as described in §4.

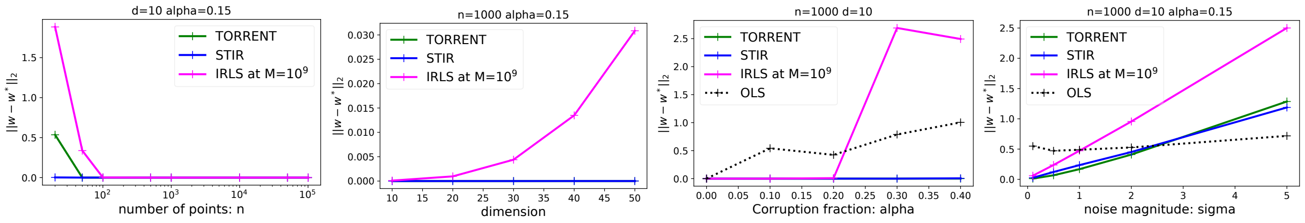
Algorithms: We compared STIR and STIR-GD with the TORRENT algorithm [6], its faster gradient version TORRENT-GD, the classical IRLS algorithm with various fixed values of the truncation parameter, and the standard OLS (Ordinary Least Squares) algorithm. We do not compare to some other state-of-the-art algorithms for robust regression, such as L_1 minimization techniques and extended Lasso since [6] establishes that TORRENT outperforms all of them.

Data: The covariate dimensionality and the number of data points are mentioned with each plot. All covariates were generated from a normal distribution. The gold and fake models were chosen as two independently sampled unit vectors. The set of "bad" data points was chosen randomly and the fake model was used to introduce corruptions, as in Section 5.



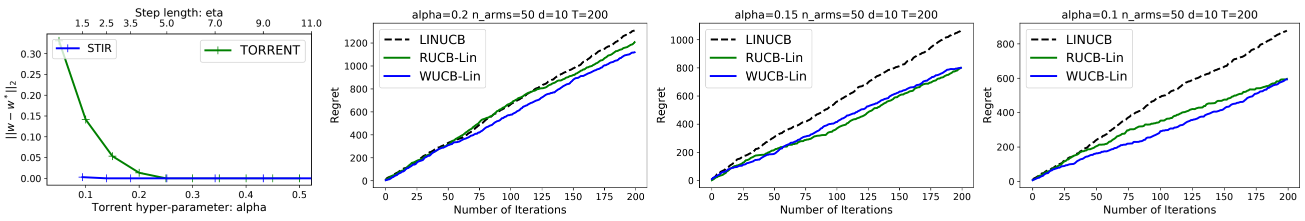
(a) STIR vs IRLS with fixed M (b) STIR vs TORRENT (c) STIR vs IRLS with fixed M (d) STIR vs TORRENT

Figure 3: All y-axes are in log-scale. Figs (a) and (c) use different data dimensionalities and number of data points and compare STIR to when IRLS is executed with various fixed values of the truncation parameter M. It is clear that no fixed value performs well. For small fixed values $M \approx 10^0$, IRLS converges rapidly but to poor models. For large fixed values $M \approx 10^{12}$, IRLS gets stuck at the fake model and takes long to converge. On the other hand, although STIR was initialized with $M_1 = 0$ for this experiment, it adaptively increases its truncation parameter to offer far better convergence than IRLS with any fixed value of M. Figs (b) and (d) compare STIR and STIR-GD with TORRENT and TORRENT-GD. In all cases, STIR-GD offers the fastest convergence.



(a) Variation with dataset size (b) Variation with dimension (c) Variation with corruption (d) Variation with white noise

Figure 4: The figures compare STIR, TORRENT, IRLS, and OLS for convergence behavior. OLS exceeds the figure boundaries and hence not visible in Figs (a) and (b). Fig (a) examines the effect of varying the training set size. Note that the x-axis is in log-scale. IRLS performs poorly with very few data points but STIR and TORRENT continue to offer good convergence. Fig (b) shows that IRLS worsens with increasing dimensionality whereas STIR and TORRENT remain stable. Fig (c) explores the affect of increasing the fraction of corrupted points. Both OLS and IRLS show considerable worsening with increasing fraction of corruptions. Finally, Fig (d) explores the hybrid noise model discussed in Section 7.3 (Figs (a)-(c) had no white noise). Here, IRLS performs the worst of all. However, once the noise variance goes beyond a point, TORRENT and STIR start losing the distinction between good and bad points and the naive OLS starts outperforming them.



(a) Misspecifying parameters (b) Underreporting α as 0.15 (c) Exact α to TORRENT (d) Overreporting α as 0.15

Figure 5: The figures compare STIR and TORRENT with respect to hyperparameter misspecification. STIR was initialized at $\mathbf{w}^0 = \mathbf{0}$ in these experiments. For Fig (a), 25% data was corrupted but TORRENT was given various values of its hyperparameter α (denoting the fraction of corrupted points) as indicated. STIR was also given various values of its own hyperparameter η in a wide range. TORRENT is very susceptible to hyperparameter misspecification and degrades heavily when not given a proper value whereas STIR is much more stable with respect to its hyperparameter. For Figs (b), (c), (d), respectively 20%, 15% and 10% of the data was corrupted and linear-armed bandit algorithms that use OLS (LINUCB), TORRENT (RUCB-Lin) and STIR (WUCB-Lin) were executed. For Figs (b), (c), (d), TORRENT was always given a hyperparameter value $\alpha = 0.15$. Note that this is appropriate for Fig (c) where actually 15% data was corrupted but not for Figs (b) and (d). TORRENT performs comparably to STIR if provided the true value of α, as in Fig (c) but its performance degrades if we give a value smaller than true value, such as in Fig (b) or a larger value, such as in Fig (d).

Algorithm 3 WUCB-Lin: Weighted UCB for Linear Contextual Bandits

```
Input: Upper bounds $\sigma_0$ (on sub-Gaussian norm of noise distribution), $B$ (on magnitude of corruption), $\alpha_0$ (on fraction of corrupted points), initial truncation $M_1$, increment rate $\eta$
1: for $t = 1, 2, \ldots, T$ do
2:     Receive set of arms $A_t$
3:     Play arm $\hat{\mathbf{x}}^t = \arg\max_{\mathbf{x}\in A_t, \mathbf{w}\in C_{t-1}} \langle \mathbf{x}, \mathbf{w}\rangle$
4:     Receive reward $r_t$
5:     $(\hat{\mathbf{w}}^t, S^t) \leftarrow$ STIR $\left(\{\hat{\mathbf{x}}^\tau, r_\tau\}_{\tau=1}^t, M_1, \eta\right)$
                                    //Denote $S^t = \mathtt{diag}(s_1^t, s_2^t, \ldots, s_t^t)$
6:     $V^t \leftarrow \sum_{\tau\le t} s_\tau^t \hat{\mathbf{x}}^\tau (\hat{\mathbf{x}}^\tau)^\top$, $X^t \leftarrow \left[\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \ldots, \hat{\mathbf{x}}^t\right]$
7:     $\bar{\mathbf{w}}^t \leftarrow (V^t)^{-1} X^t S^t \mathbf{y}$
8:     $C_t \leftarrow \{\mathbf{w} : \left\|\mathbf{w} - \bar{\mathbf{w}}^t\right\|_{V^t} \le \sigma_0\sqrt{d\log T} + \alpha_0 BT\}$
9: end for
```

8.2 Robust Linear Bandit Experiments

As linear-armed bandit algorithms [1] utilize regression routines internally, recent works have explored the possibility of using robust regression algorithms to target cases when arm-pulls are corrupted, for example [19] that uses TORRENT itself to develop corruption-tolerant bandit learning algorithms.

Algorithm 3 presents WUCB-Lin, an adaptation of STIR to linear bandit settings. We refer the reader to Appendix F for details of the algorithm. WUCB-Lin roughly follows the popular *Optimism-in-the-face-of-uncertainty* (OFUL) principle while selecting arms to pull at various time instants.

However, since we know some of the arm pulls generated corrupted rewards, instead of applying the OFUL principle blindly, WUCB-Lin invokes STIR and obtains not only an estimate of the reward generating model, but also a set of weights on previous arm pulls which indicate which pulls were corrupted and which pulls were clean. WUCB-Lin then uses these weights to form a *weighted confidence set* (Algorithm 3, line 6) that is further utilized in applying the OFUL principle to decide future arm pulls (Algorithm 3, line 3).

Algorithms and Data: We compare WUCB-Lin with LINUCB that uses the simple OLS estimator, as well as the RUCB-Lin algorithm from [19]. We refer the reader to Appendix F for details of the problem setting.

8.3 Discussion on Experiments

Figures 3, 4 and 5 present graphs with the outcomes of the experiments. Although the respective captions in the figures detail the observed behaviours of various algorithms considered therein, here we point out some broad inferences.

1. STIR-GD offers much faster convergence as compared to TORRENT or TORRENT-GD.

2. No single value of the truncation parameter M ensures a good performance with IRLS. A stage-wise implementation with continuously updated truncation parameters, as STIR offers, is necessary for rapid and assuredly global convergence.

3. TORRENT requires an estimate of the fraction of corrupted points as a hyperparameter and is extremely susceptible to misspecification in this value. STIR and STIR-GD on the other hand are much more resilient to misspecifications of their own hyperparameters.

9 Conclusion and Future Work

In this work we presented STIR, a stage-wise algorithm that makes simple and efficient modifications, including a gradient-based implementation STIR-GD, to the well-known IRLS heuristic to obtain the first global convergence results for robust regression. These algorithms offer not only theoretically superior results to state-of-the-art algorithms such as TORRENT but are empirically faster and more immune to hyperparameter mis-specification.

Our theoretical results are superior to those of previous works in terms of offering a better breakdown point, and are based on a novel notion of weighted strong convexity. Working with this new notion of strong convexity required us to develop the peeling proof technique which is novel in robust regression literature and may be of independent interest in analyzing other iterative algorithms.

Several avenues of future work exist. It would be interesting to examine other weighing functions (IRLS and STIR use the inverse of the residual) for robust regression. It is likely that any reasonable decreasing function of residuals should suffice. It would also be interesting to derive formal regret bounds for the WUCB-Lin algorithm and see how they compare to the regret bounds of the RUCB-Lin algorithm from [19].

Acknowledgements

The authors thank the anonymous reviewers for several helpful comments on a previous version of the paper. B.M. thanks the Research-I Foundation for a travel grant. P.K. is supported by the Deep Singh and Daljeet Kaur Faculty Fellowship and the Research-I Foundation at IIT Kanpur, and thanks Microsoft Research India and Tower Research for research grants.

References

[1] Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Improved Algorithms for Linear Stochastic Bandits. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

[2] Khurrum Aftab and Richard Hartley. Convergence of Iteratively Re-weighted Least Squares to Robust M-Estimators. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[3] Alekh Agarwal, Sahand N. Negahban, and Martin J. Wainwright. Fast global convergence of gradient methods for high-dimensional statistical recovery. *Annals of Statistics*, 40(5):2452–2482, 2012.

[4] Demba Ba, Behtash Babadi, Patrick L. Purdon, and Emery N. Brown. Convergence and Stability of Iteratively Re-weighted Least Squares Algorithms. *IEEE Transactions on Signal Processing*, 62(1):183–195, 2013.

[5] Kush Bhatia, Prateek Jain, Parameswaran Kamalaruban, and Purushottam Kar. Consistent Robust Regression. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[6] Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust Regression via Hard Thresholding. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[7] Nicolai Bissantz, Lutz Dümbgen, Axel Munk, and Bernd Stratmann. Convergence Analysis of Generalized Iteratively Reweighted Least Squares Algorithms on Convex Function Spaces. *SIAM Journal of Optimization*, 19(4):1828–1845, 2009.

[8] Jack Brimberg and Robert F. Love. Global Convergence of a Generalized Iterative Procedure for the Minisum Location Problem with lp Distances. *Operations Research*, 41(6):1010–1176, 1993.

[9] Emmanuel J. Candès, Xiaodong Li, and John Wright. Robust Principal Component Analysis? *Journal of the ACM*, 58(1):1–37, 2009.

[10] Yin Chen and Arnak S. Dalalyan. Fused sparsity and robust estimation for linear models with unknown variance. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS)*, 2012.

[11] Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust Sparse Regression under Adversarial Corruption. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

[12] A. K. Cline. Rate of Convergence of Lawson's Algorithm. *Mathematics of Computation*, 26(117):167–176, 1972.

[13] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C. Sinan Güntürk. Iteratively Reweighted Least Squares Minimization for Sparse Recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

[14] Ilias Diakonikolas, Gautam Kamath, Daniel M. Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robustly Learning a Gaussian: Getting Optimal Error, Efficiently. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2683–2702, 2018.

[15] Ilias Diakonikolas, Weihao Kong, and Alistair Stewart. Efficient Algorithms and Lower Bounds for Robust Linear Regression. In *30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

[16] Jiashi Feng, Huan Xu, Shie Mannor, and Shuicheng Yan. Robust Logistic Regression and Classification. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

[17] Ian Goodfellow, Patrick McDaniel, and Nicolas Papernot. Making Machine Learning Robust Against Adversarial Inputs. *Communications of the ACM*, 61(7):56–66, 2018.

[18] Peter J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[19] Sayash Kapoor, Kumar Kshitij Patel, and Purushottam Kar. Corruption-tolerant bandit learning. Machine Learning (to appear) https://doi.org/10.1007/s10994-018-5758-5, 2018.

[20] Lihong Li, Wei Chu, John Langford, and Robert Schapire. A Contextual-Bandit Approach to Personalized News Article Recommendation. In *Proceedings of the 19th International World Wide Web Conference (WWW)*, 2010.

[21] Liu Liu, Yanyao Shen, Tianyang Li, and Constantine Caramanis. High Dimensional Robust Sparse Regression. arXiv:1805.11643v1 [cs.LG], 2018.

[22] Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 114–122, 2018.

[23] Julien Mairal. Incremental Majorization-Minimization Optimization with Application to Large-Scale Machine Learning. *SIAM Journal of Optimization*, 25(2):829–855, 2015.

[24] Brian McWilliams, Gabriel Krummenacher, Mario Lucic, and Joachim M. Buhmann. Fast and Robust Least Squares Estimation in Corrupted Linear Models. In *28th Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

[25] M. R. Osborne. *Finite Algorithms in Optimization and Data Analysis*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons, 1985.

[26] Damian Straszak and Nisheeth K. Vishnoi. IRLS and Slime Mold: Equivalence and Convergence. arXiv:1601.02712 [cs.DS], 2016.

[27] John W. Tukey. A Survey of Sampling from Contaminated Distributions. *Contributions to Probability and Statistics*, 2:448–485, 1960.

[28] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

[29] John Wright and Yi Ma. Dense Error Correction via ℓ^1 Minimization. *IEEE Transactions on Information Theory*, 56(7):3540–3560, 2010.