# RIFLE: Imputation and Robust Inference from Low Order Marginals

**Sina Baharlouei** [1] **Kelechi Ogudu** [2] **Peng Dai** [1] **Sze-chuan Suen** [1] **Meisam Razaviyayn** [1]

## Abstract

The ubiquity of missing values in real-world datasets poses a challenge for statistical inference and can prevent similar datasets from being analyzed in the same study, precluding many existing datasets from being used for new analyses. While an extensive collection of packages and algorithms have been developed for data imputation, the overwhelming majority perform poorly if there are many missing values and low sample sizes, which are unfortunately common characteristics in empirical data. Such low-accuracy estimations adversely affect the performance of downstream statistical models. We develop a statistical inference framework for *predicting the target variable in the presence of missing data without imputation*. Our framework, RIFLE (Robust InFerence via Low-order moment Estimations), estimates low-order moments of the underlying data distribution with corresponding confidence intervals to learn a distributionally robust model. We specialize our framework to ridge linear regression, where the resulting min-max problem is efficiently solved by applying the alternating direction method of multipliers (ADMM) on the dual problem. This framework can also be adapted to impute missing data. We compare RIFLE with state-of-the-art approaches (including MICE, Amelia, MissForest, KNN-imputer, MIDA, and Mean Imputer) in numerical experiments. Our experiments demonstrate that RIFLE outperforms other benchmark algorithms when the percentage of missing values is high and/or when the number of data points is relatively small. RIFLE is publicly available[1].

[1]Industrial and Systems Engineering Department, University of Southern California [2]Computer Science Department, University of Southern California. Correspondence to: Sina Baharlouei <baharlou@usc.edu>.

*ICML 2023 Workshop on Duality for Modern Machine Learning*, Honolulu, Hawaii, USA. Copyright 2023 by the author(s).

[1]https://github.com/optimization-for-data-driven-science/RIFLE

## 1. Introduction

Machine learning algorithms have shown promise when applied to various problems, including healthcare, finance, social data analysis, image processing, and speech recognition. However, this success mainly relied on the availability of large-scale, high-quality datasets, which may be scarce in many practical problems, especially in medical and health applications (Pedersen et al.; Sterne et al., 2009; Beaulieu-Jones et al., 2018). Moreover, many experiments and datasets suffer from the small sample size in such applications. Despite the availability of a small number of data points in each study, an increasingly large number of datasets are publicly available. To fully and effectively utilize information on related research questions from diverse datasets, information across various datasets (e.g., different questionnaires from multiple hospitals with overlapping questions) must be combined in a reliable fashion. After appending these datasets together, the obtained dataset can contain large blocks of missing values, as they may not share the same features (Figure 1).

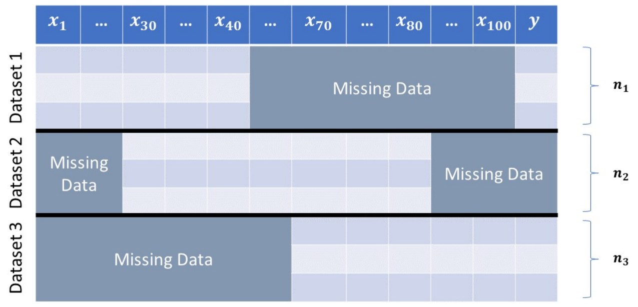

*Figure 1.* Consider the problem of predicting the trait $y$ from feature vector $(\mathbf{x}_1, \ldots, \mathbf{x}_{100})$. Suppose that we have access to three data sets: The first dataset includes the measurements of $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{40}, y)$ for $n_1$ individuals. The second dataset collects data from another $n_2$ individuals by measuring $(\mathbf{x}_{30}, \ldots, \mathbf{x}_{80})$ with no measurements of the target variable $y$ in it; and the third dataset contains the measurements from the variables $(\mathbf{x}_{70}, \ldots, \mathbf{x}_{100}, y)$ for $n_3$ number of individuals. How one should learn the predictor $\hat{y} = h(\mathbf{x}_1, \ldots, \mathbf{x}_{100})$ from these three datasets?

**Related Works:** There are three general approaches for handling missing values in classification and regression tasks. A Naive method is to remove the rows containing missing entries. However, such an approach is not an option when the percentage of missingness in a dataset is high. For instance, as demonstrated in Figure 1, the entire dataset will be discarded if we eliminate the rows with at least one missing entry.

The most common approach for handling missing values in a learning task is to impute them in a pre-processing stage. The general idea behind data imputation approaches is that the missing values can be predicted using the other available data points and correlated features. Imputation algorithms cover a wide range of methods, including imputing missing entries with the columns means Little & Rubin (2019, Chapter 3) (or median), least-square and linear regression-based methods (Raghunathan et al., 2001; Kim et al., 2005; Zhang et al., 2008; Cai et al., 2006; Buuren & Groothuis-Oudshoorn, 2010), matrix completion and expectation maximization approaches Dempster et al. (1977); Ghahramani & Jordan (1994); Honaker et al. (2011), KNN based (Troyanskaya et al., 2001), Tree based methods (Stekhoven & Bühlmann, 2012; Xia et al., 2017), and methods using different neural network structures. Appendix A presents a comprehensive review of these methods.

The imputation of data allows practitioners to run standard statistical algorithms requiring complete data. However, the prediction model's performance can be highly reliant on the accuracy of the imputer. High error rates in the prediction of missing values by the imputer can lead to the catastrophic performance of the downstream statistical methods executed on the imputed data.

Another class of methods for inference in the presence of missing values relies on robust optimization over the uncertainty sets on missing entries. Shivaswamy et al. (2006) and Xu et al. (2009) adopt robust optimization to learn the parameters of a support vector machine model. They consider uncertainty sets for the missing entries in the dataset and solve a min-max problem over those sets. The obtained classifiers are robust to the uncertainty of missing entries within the uncertainty regions. In contrast to the imputation-based approaches, the robust classification formulation does not carry the imputation error to the classification phase. However, finding appropriate intervals for each missing entry is challenging, and it is unclear how to determine the uncertainty range in many real datasets. Moreover, their proposed algorithms are limited to the SVM classifier.

In this paper, we propose RIFLE (Robust InFerence via Low-order moment Estimations) for the direct inference of a target variable based on a set of features containing missing values. The proposed framework does not require the data to be imputed in a pre-processing stage. However, it can also be used as a pre-processing tool for imputing data. The main idea of the proposed framework is to estimate the first and second-order moments of the data and their confidence intervals by bootstrapping on the available

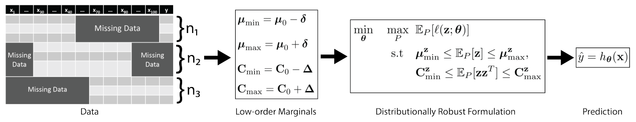
$x_1$ … $x_{30}$ … $x_{40}$ … $x_{70}$ … $x_{80}$ … $x_{100}$ y
Missing Data
Missing Data
Missing Data
Missing Data
$n_1$
$n_2$
$n_3$
Data
$\boldsymbol{\mu}_{\min} = \boldsymbol{\mu}_0 - \boldsymbol{\delta}$
$\boldsymbol{\mu}_{\max} = \boldsymbol{\mu}_0 + \boldsymbol{\delta}$
$\mathbf{C}_{\min} = \mathbf{C}_0 - \boldsymbol{\Delta}$
$\mathbf{C}_{\max} = \mathbf{C}_0 + \boldsymbol{\Delta}$
Low-order Marginals
$\min_{\boldsymbol{\theta}} \max_{P} \mathbb{E}_P[\ell(\mathbf{z};\boldsymbol{\theta})]$
s.t $\boldsymbol{\mu}^{\mathbf{z}}_{\min} \le \mathbb{E}_P[\mathbf{z}] \le \boldsymbol{\mu}^{\mathbf{z}}_{\max}$,
$\mathbf{C}^{\mathbf{z}}_{\min} \le \mathbb{E}_P[\mathbf{z}\mathbf{z}^T] \le \mathbf{C}^{\mathbf{z}}_{\max}$
Distributionally Robust Formulation
$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x})$
Prediction

*Figure 2.* Prediction of the target variable without imputation. RIFLE estimates confidence intervals for low-order (first and second-order) marginals from the input data containing missing values. Then, it solves a distributionally robust problem over the set of all distributions whose low-order marginals are within the estimated confidence intervals.

data matrix entries. Then, RIFLE finds the optimal parameters of the statistical model for the worst-case distribution with the low-order moments (mean and variance) within the estimated confidence intervals (See Figure 2). Compared to Shivaswamy et al. (2006); Xu et al. (2009) we estimate uncertainty regions for the low-order marginals using the Bootstrap technique. Furthermore, our framework is not restricted to any particular machine learning model, such as support vector machines (Xu et al., 2009).

## 2. Robust Inference via Estimating Low-order Moments

RIFLE is based on a distributionally robust optimization (DRO) framework over low-order marginals. Assume that $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ follows a joint probability distribution $P^*$. A standard approach for predicting the target variable $y$ given the input vector $\mathbf{x}$ is to find the parameter $\boldsymbol{\theta}$ that minimizes the population risk with respect to a given loss function $\ell$:

$$\min_{\boldsymbol{\theta}} \quad \mathbb{E}_{(\mathbf{x},y)\sim P^*}\Big[\ell\Big(\mathbf{x}, y; \boldsymbol{\theta}\Big)\Big]. \tag{1}$$

Since the underlying distribution of data is rarely available in practice, the above problem cannot be directly solved. The most common approach for approximating (1) is to minimize the empirical risk with respect to $n$ given i.i.d samples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ drawn from the joint distribution $P^*$:

$$\min_{\boldsymbol{\theta}} \quad \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{x}_i, y_i; \boldsymbol{\theta}).$$

The above empirical risk formulation assumes that all entries of $\mathbf{x}_i$ and $y_i$ are available. Thus, to utilize the empirical risk minimization (ERM) framework in the presence of missing values, one can either remove or impute the missing data points in a pre-processing stage. Training via robust optimization is a natural alternative in the presence of missing data. Shivaswamy et al. (2006); Xu et al. (2009) suggest the following optimization problem that minimizes the loss function for the worst-case scenario over the defined uncertainty sets per data points:

$$\min_{\boldsymbol{\theta}} \quad \max_{\{\boldsymbol{\delta}_i \in \mathcal{N}_i\}_{i=1}^n} \quad \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{x}_i - \boldsymbol{\delta}_i, y_i; \boldsymbol{\theta}), \tag{2}$$

where $\mathcal{N}_i$ represents the uncertainty region of data point $i$. Shivaswamy et al. (2006) obtains the uncertainty sets by assuming a known distribution on the missing entries of datasets. The main issue in their approach is that the constraints defined on data points are totally uncorrelated. Xu et al. (2009) on the other hand defines $\mathcal{N}_i$ as a "box" constraint around the data point $i$ such that they can be linearly correlated. For this specific case, they show that solving the corresponding robust optimization problem is equivalent to minimizing a regularized reformulation of the original loss function. Such an approach has several limitations: First, it can only handle a few special cases (SVM loss with linearly correlated perturbations on data points). Furthermore, Xu et al. (2009) is primarily designed for handling outliers and contaminated data. Thus, they do not offer any mechanism for the initial estimation of $\mathbf{x}_i$ when several vector entries are missing. In this work, we instead take a *distributionally robust* approach by consiering uncertainty on the distribution of the data instead of defining uncertainty set for each data point. In particular, we aim to fit the best parameters of a statistical learning model for the worst distribution in a given uncertainty set by solving:

$$\min_{\boldsymbol{\theta}} \quad \max_{P \in \mathcal{P}} \quad \mathbb{E}_{(\mathbf{x},y)\sim P}[\ell(\mathbf{x}, y; \boldsymbol{\theta})], \tag{3}$$

where $\mathcal{P}$ is an uncertainty set over the underlying distribution of data. A key observation is that defining the uncertainty set $\mathcal{P}$ in (3) is easier and computationally more efficient than defining the uncertainty sets $\{\mathcal{N}_i\}_{i=1}^n$ in (2). In particular, the uncertainty set $\mathcal{P}$ can be obtained naturally by estimating low-order moments of data distribution using only available entries. To explain this idea and to simplify the notations, let $\mathbf{z} = (\mathbf{x}, y)$, $\bar{\boldsymbol{\mu}}^{\mathbf{z}} \triangleq \mathbb{E}[\mathbf{z}]$, and $\bar{\mathbf{C}}^{\mathbf{z}} \triangleq \mathbb{E}[\mathbf{z}\mathbf{z}^T]$. While $\bar{\boldsymbol{\mu}}^{\mathbf{z}}$ and $\bar{\mathbf{C}}^{\mathbf{z}}$ are typically not known exactly, one can estimate them (within certain confidence intervals) from the available data by simply ignoring missing entries. Moreover, we can estimate the confidence intervals via bootstraping. Particularly, we can estimate $\boldsymbol{\mu}^{\mathbf{z}}_{\min}, \boldsymbol{\mu}^{\mathbf{z}}_{\max}, \mathbf{C}^{\mathbf{z}}_{\min}$, and $\mathbf{C}^{\mathbf{z}}_{\max}$ from data such that $\boldsymbol{\mu}^{\mathbf{z}}_{\min} \leq \bar{\boldsymbol{\mu}}^{\mathbf{z}} \leq \boldsymbol{\mu}^{\mathbf{z}}_{\max}$ and $\mathbf{C}^{\mathbf{z}}_{\min} \leq \bar{\mathbf{C}}^{\mathbf{z}} \leq \mathbf{C}^{\mathbf{z}}_{\max}$ with high probability (where the inequalities for matrices and vectors denote component-wise relations). In Appendix B, we show how a bootstrapping strategy can be used to obtain the confidence intervals described above. Given these estimated confidence intervals from data, (3) can be reformulated as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \max_{P} \ \mathbb{E}_P[\ell(\mathbf{z}; \boldsymbol{\theta})] \\ & \text{s.t.} \quad \boldsymbol{\mu}^{\mathbf{z}}_{\min} \leq \mathbb{E}_P[\mathbf{z}] \leq \boldsymbol{\mu}^{\mathbf{z}}_{\max}, \\ & \qquad\ \mathbf{C}^{\mathbf{z}}_{\min} \leq \mathbb{E}_P[\mathbf{z}\mathbf{z}^T] \leq \mathbf{C}^{\mathbf{z}}_{\max}. \end{aligned} \tag{4}$$

Gao & Kleywegt (2017) utilize the distributionally robust optimization as (3) over the set of positive semi-definite (PSD) cones for robust inference under uncertainty. While their formulation considers $\ell_2$ balls for the constraints on low order moments of the data, we use $\ell_\infty$ constraints that are computationally more natural in the presence of missing entries when combined with bootstrapping. Furthermore, while it can be applied to general convex losses, their method relies on the ellipsoid and the existence of oracles for performing the steps of the ellipsoid method, which is not applicable in modern high-dimensional problems. Moreover, they assume concavity in data (the existence of some oracle to return the worst-case data points) that is practically unavailable even in convex loss functions (including linear regression in our work).

## 3. Robust Linear Regression in the Presence of Missing Values

Let us specialize our framework to the ridge linear regression model. In the absence of missing data, ridge regression finds optimal regressor parameter $\boldsymbol{\theta}$ by solving

$$\min_{\boldsymbol{\theta}} \quad \|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_2^2,$$

or equivalently by minimizing:

$$\boldsymbol{\theta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\theta} - 2\boldsymbol{\theta}^T\mathbf{X}^T\mathbf{y} + \lambda\|\boldsymbol{\theta}\|_2^2. \tag{5}$$

Thus, having the second-order moments of the data $\mathbf{C} = \mathbf{X}^T\mathbf{X}$ and $\mathbf{b} = \mathbf{X}^T\mathbf{y}$ is sufficient for finding the optimal solution. In other words, it suffices to compute the inner product of any two column vectors $\mathbf{a}_i$, $\mathbf{a}_j$ of $\mathbf{X}$, and the inner product of any column $\mathbf{a}_i$ of $\mathbf{X}$ with vector $\mathbf{y}$. Since the matrix $\mathbf{X}$ and vector $\mathbf{y}$ are not fully observed due to the existence of missing values, one can use the same approach as (12) to compute the point estimators $\mathbf{C}_0$ and $\mathbf{b}_0$. These point estimators can be highly inaccurate, especially when the number of non-missing rows for two given columns is small. In addition, if the pattern of missing entries does not follow the MCAR assumption, the point estimators are not unbiased estimators of $\mathbf{C}$ and $\mathbf{b}$.

### 3.1. A Distributionally Robust Formulation of Linear Regression

As we mentioned above, to solve the linear regression problem, we only need to estimate the second-order moments of the data ($\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}^T\mathbf{y}$). Thus, the distributionally robust formulation described in (4) is equivalent to the following optimization problem for the linear regression model:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \max_{\mathbf{C},\mathbf{b}} \ & \boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} - 2\mathbf{b}^T\boldsymbol{\theta} + \lambda\|\boldsymbol{\theta}\|_2^2 \\ \text{s.t.} \quad & \mathbf{C}_0 - c\boldsymbol{\Delta} \leq \mathbf{C} \leq \mathbf{C}_0 + c\boldsymbol{\Delta}, \\ & \mathbf{b}_0 - c\boldsymbol{\delta} \leq \mathbf{b} \leq \mathbf{b}_0 + c\boldsymbol{\delta}, \\ & \mathbf{C} \succeq 0, \end{aligned} \tag{6}$$

where the last constraint guarantees that the covariance matrix is positive and semi-definite. We dicuss the procedure of estimating the confidence intervals ($\mathbf{b}_0, \mathbf{C}_0, \boldsymbol{\delta}$, and $\boldsymbol{\Delta}$) in Appendix B.

### 3.2. RIFLE for Ridge Linear Regression

Since the objective function in 6 is convex in $\boldsymbol{\theta}$ (ridge regression) and concave in $\mathbf{b}$ and $\mathbf{C}$ (linear), the minimization and maximization sub-problems are interchangeable (Sion et al., 1958). Thus, we can equivalently rewrite Problem (6) as:

$$
\begin{aligned}
\max_{\mathbf{C},\mathbf{b}} \; & g(\mathbf{C},\mathbf{b}) \\
\text{s.t.} \; & \mathbf{C}_0 - c\boldsymbol{\Delta} \leq \mathbf{C} \leq \mathbf{C}_0 + c\boldsymbol{\Delta}, \\
& \mathbf{b}_0 - c\boldsymbol{\delta}_0 \leq \mathbf{b} \leq \mathbf{b}_0 + c\boldsymbol{\delta}, \\
& \mathbf{C} \succeq 0,
\end{aligned}
\tag{7}
$$

where $g(\mathbf{b},\mathbf{C}) = \min_{\boldsymbol{\theta}} \boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} - 2\mathbf{b}^T\boldsymbol{\theta} + \lambda\|\boldsymbol{\theta}\|^2$. Function $g$ can be computed in closed-form given any pair of $(\mathbf{C},\mathbf{b})$ by $\boldsymbol{\theta} = (\mathbf{C}+\lambda\mathbf{I})^{-1}\mathbf{b}$. Based on that, we can apply projected gradient ascent to function $g$ to find an optimal solution of (7) as described in Algorithm 1. At each iteration of the algorithm, we first perform one step of projected gradient ascent on matrix $\mathbf{C}$ and vector $\mathbf{b}$; then we update $\boldsymbol{\theta}$ in closed-form for the obtained $\mathbf{C}$ and $\mathbf{b}$. We use $\mathbf{C}_0$ and $\mathbf{b}_0$ obtained from Equation (12) (see Appendix B) as the initialization points for $\mathbf{C}$ and $\mathbf{b}$ respectively. The projection of $\mathbf{b}$ to the box contraint $\mathbf{b}_0 - c\boldsymbol{\delta} \leq \mathbf{b} \leq \mathbf{b}_0 + c\boldsymbol{\delta}$ can be done entriwise and has the following closed-form

**Algorithm 1** Projection Gradient Ascent on function $g(\mathbf{C},\mathbf{b})$ in the Presence of Missing Values

```
1: Input: $\mathbf{C}_0, \mathbf{b}_0, \boldsymbol{\Delta}, \boldsymbol{\delta}, T$
2: Initialize: $\mathbf{C} = \mathbf{C}_0, \mathbf{b} = \mathbf{b}_0$.
3: for $i = 1, \ldots, T$ do
4:     Update $\mathbf{C} = \Pi_{\boldsymbol{\Delta}+}\left[\mathbf{C} + \alpha\boldsymbol{\theta}\boldsymbol{\theta}^T\right]$
5:     Update $\mathbf{b} = \Pi_{\boldsymbol{\delta}}(\mathbf{b} - 2\alpha\boldsymbol{\theta})$
6:     Set $\boldsymbol{\theta} = (\mathbf{C}+\lambda\mathbf{I})^{-1}\mathbf{b}$
7: end for
```

$$
\Pi_{\delta}(\mathbf{b}_i) = \begin{cases} \mathbf{b}_i & \text{if} \quad \mathbf{b}_{0i} - c\boldsymbol{\delta}_i \leq \mathbf{b}_i \leq \mathbf{b}_{0i} + c\boldsymbol{\delta}_i, \\ \mathbf{b}_{0i} - c\boldsymbol{\delta}_i & \text{if} \quad \mathbf{b}_i \leq \mathbf{b}_{0i} - c\boldsymbol{\delta}_i, \\ \mathbf{b}_i + c\boldsymbol{\delta}_i & \text{if} \quad \mathbf{b}_{0i} + c\boldsymbol{\delta}_i \leq \mathbf{b}_i. \end{cases}
$$

**Theorem 1.** *Assume that $D = \|\mathbf{C}_0 - \mathbf{C}^*\|_F^2 + \|\mathbf{b}_0 - \mathbf{b}^*\|_2^2$. Then Algorithm 1 computes an $\epsilon$-stationary solution of the objective function in (7) in $\mathcal{O}\left(\frac{LD}{\epsilon}\right)$ iterations, where $L = \frac{2}{\lambda}$.*

*Proof.* The proof is relegated to Appendix F. □

In Appendix C, we show how using the acceleration method of Nesterov can improve the convergence rate of Algorithm 1 to $\mathcal{O}\left(\sqrt{\frac{LD}{\epsilon}}\right)$. A technical issue of Algorithm 1 and its accelerated version presented in Appendix C is that applying the simultaneous projection of $\mathbf{C}$ to both box constraints and the set of positive semidefinite matrices ($\Pi_{\boldsymbol{\Delta}+}[\mathbf{C}]$) is challenging and cannot be done in closed-form. A more resilient approach to handle both constraints simultaneously is to write the dual problem of the maximization problem and handle the resulting constrained minimization problem with the Alternating Direction Method of Multipliers (ADMM).

## 4. Solving the Dual Problem of the Robust Ridge Linear Regression via ADMM

The Alternating Direction Method of Multipliers (ADMM) is a popular algorithm for efficiently solving linearly constrained optimization problems (Gabay & Mercier, 1976; Hong et al., 2016). It has been extensively applied to large-scale optimization problems in machine learning and statistical inference in recent years (Assländer et al., 2018; Zhang et al., 2018). Consider the following optimization problem consisting of two blocks of variables $\mathbf{x}$ and $\mathbf{y}$ that are linearly coupled:

$$
\begin{aligned}
\min_{\mathbf{w},\mathbf{z}} \; & f(\mathbf{w}) + g(\mathbf{z}) \\
\text{s.t.} \; & \mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{z} = \mathbf{c},
\end{aligned}
\tag{8}
$$

The augmented Lagrangian of the above problem can be written as:

$$
\begin{aligned}
\min_{\mathbf{w},\mathbf{z}} \quad f(\mathbf{w}) + g(\mathbf{z}) & + \langle \mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{z} - \mathbf{c}, \boldsymbol{\lambda} \rangle \\
& + \frac{\rho}{2}\|\mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{z} - \mathbf{c}\|^2,
\end{aligned}
\tag{9}
$$

ADMM schema updates the primal and dual variables iteratively as presented in Algorithm 2.

**Algorithm 2** Schema of ADMM algorithm Applied to the Augmented Lagrangian Problem

```
1: for $t = 1, \ldots, T$ do
2:     $\mathbf{w}^{t+1} = \text{argmin}_{\mathbf{w}} f(\mathbf{w}) + \langle \mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{z}^t - \mathbf{c}, \boldsymbol{\lambda} \rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{z}^t - \mathbf{c}\|^2$
3:     $\mathbf{z}^{t+1} = \text{argmin}_{\mathbf{z}} f(\mathbf{w}^{t+1}) + \langle \mathbf{A}\mathbf{w}^{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c}, \boldsymbol{\lambda} \rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{w}^{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c}\|^2$
4:     $\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \rho(\mathbf{A}\mathbf{w}^{t+1} + \mathbf{B}\mathbf{z}^{t+1} - \mathbf{c})$
5: end for
```

As we mentioned earlier, simultaneous projection of $\mathbf{C}$ to the set of positive semi-definite matrices and the box constraint $\mathbf{C}_{\min} \leq \mathbf{C} \leq \mathbf{C}_{\max}$ in Algorithm 1 is computationally expensive. Moreover, careful step-size tuning is necessary to avoid inconsistency and guarantee convergence in that algorithm.

An alternative approach for solving Problem (6) that avoids removing the PSD constraint in the implementation of Algorithm 1 is to solve the dual of the inner maximization problem. Since the maximization problem is concave with respect to $\mathbf{C}$ and $\mathbf{b}$, and the relative interior of the feasible set of constraints is non-empty, the duality gap is zero. Hence, instead of solving the inner maximization problem, we can solve its dual which is a minimization problem. Theorem 2 describes the dual problem of the inner maximization problem in (6). Thus, Problem (6) can be alternatively formulated as a minimization problem rather than a min-max problem. We can solve such a constrained minimization problem efficiently via the ADMM algorithm. As we will show, the ADMM algorithm applied to the dual problem does not need tuning of step size or applying simultaneous projections to the box constraints and positive semi-definite (PSD) constraints.

**Theorem 2.** *(Dual Problem) Problem* (6) *can be equivalently formulated as:*

$$\begin{aligned}\min_{\boldsymbol{\theta},\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}} \quad & -\langle \mathbf{b}_{\min}, \mathbf{d}\rangle + \langle \mathbf{b}_{\max}, \mathbf{e}\rangle \\ & -\langle \mathbf{C}_{\min}, \mathbf{A}\rangle + \langle \mathbf{C}_{\max}, \mathbf{B}\rangle + \lambda\|\boldsymbol{\theta}\|^2 \\ \text{s.t.} \quad & -\boldsymbol{\theta}\boldsymbol{\theta}^T - \mathbf{A} + \mathbf{B} - \mathbf{H} = 0, \\ & 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e} = 0, \\ & \mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{e} \geq 0, \\ & \mathbf{H} \succeq 0.\end{aligned}$$

*Proof.* The proof is relegated to Appendix F. □

To apply the ADMM method to the dual problem, we need to divide the optimization variables into two blocks as in (8) to solve both sub-problems in Algorithm 2 efficiently. To do so, first, we introduce the auxiliary variables $\mathbf{d}^{'}, \mathbf{e}^{'}, \boldsymbol{\theta}^{'}, \mathbf{A}^{'}$ and $\mathbf{B}^{'}$ to the dual problem. Also, let $\mathbf{G} = \mathbf{H} + \boldsymbol{\theta}^{'}\boldsymbol{\theta}^{'T}$. Therefore, Problem (2) is equivalent to:

$$\begin{aligned}\min_{\boldsymbol{\theta},\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}} \quad & -\langle \mathbf{b}_{\min}, \mathbf{d}\rangle + \langle \mathbf{b}_{\max}, \mathbf{e}\rangle \\ & -\langle \mathbf{C}_{\min}, \mathbf{A}\rangle + \langle \mathbf{C}_{\max}, \mathbf{B}\rangle + \lambda\|\boldsymbol{\theta}\|^2 \\ \text{s.t.} \quad & \mathbf{B} - \mathbf{A} = \mathbf{G}, \\ & 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e} = 0, \\ & \mathbf{A} = \mathbf{A}^{'}, \mathbf{B} = \mathbf{B}^{'}, \\ & \mathbf{d} = \mathbf{d}^{'}, \mathbf{e} = \mathbf{e}^{'}, \boldsymbol{\theta} = \boldsymbol{\theta}^{'}, \\ & \mathbf{A}^{'}, \mathbf{B}^{'}, \mathbf{d}^{'}, \mathbf{e}^{'} \geq 0, \\ & \mathbf{G} \succeq \boldsymbol{\theta}^{'}\boldsymbol{\theta}^{'T}.\end{aligned} \tag{10}$$

Since handling both constraints on $\boldsymbol{\theta}$ in Problem (2) is difficult, we interchange $\boldsymbol{\theta}$ with $\boldsymbol{\theta}^{'}$ in the first constraint. Moreover, the non-negativity constraints on $\mathbf{A}, \mathbf{B}, \mathbf{d}$ and $\mathbf{e}$ are exchanged with non-negativity constraints on $\mathbf{A}^{'}, \mathbf{B}^{'}, \mathbf{d}^{'}$ and $\mathbf{e}^{'}$. Now, we show how to apply ADMM schema to Problem (10) to obtain Algorithm 5. As we discussed earlier, we consider two separate blocks of variables $\mathbf{w} = (\boldsymbol{\theta}, \mathbf{d}, \mathbf{e}, \mathbf{G}, \mathbf{B}^{'}, \mathbf{A}^{'})$ and $\mathbf{z} = (\mathbf{d}^{'}, \mathbf{e}^{'}, \boldsymbol{\theta}^{'}, \mathbf{B}, \mathbf{A})$. Assigning $\boldsymbol{\Gamma}, \boldsymbol{\eta}, \mathbf{M}_A, \mathbf{M}_B, \boldsymbol{\mu}_d, \boldsymbol{\mu}_e$, and $\boldsymbol{\mu}_\theta$ to the constraints of Problem (10) in order, we can write the corresponding augmented Lagrangian function as:

$$\begin{aligned}\min_{\substack{\boldsymbol{\theta},\boldsymbol{\theta}^{'},\mathbf{A},\mathbf{A}^{'},\mathbf{B},\mathbf{B}^{'} \\ \mathbf{d},\mathbf{d}^{'},\mathbf{e},\mathbf{e}^{'},\mathbf{G}}} \quad & -\langle \mathbf{b}_{\min}, \mathbf{d}\rangle + \langle \mathbf{b}_{\max}, \mathbf{e}\rangle \\ & -\langle \mathbf{C}_{\min}, \mathbf{A}\rangle + \langle \mathbf{C}_{\max}, \mathbf{B}\rangle + \lambda\|\boldsymbol{\theta}\|^2 \\ & +\langle \mathbf{A} - \mathbf{A}^{'}, \mathbf{M}_A\rangle + \frac{\rho}{2}\|\mathbf{A} - \mathbf{A}^{'}\|_F^2 \\ & +\langle \mathbf{B} - \mathbf{B}^{'}, \mathbf{M}_B\rangle + \frac{\rho}{2}\|\mathbf{B} - \mathbf{B}^{'}\|_F^2 \\ & +\langle \mathbf{d} - \mathbf{d}^{'}, \boldsymbol{\mu}_d\rangle + \frac{\rho}{2}\|\mathbf{d} - \mathbf{d}^{'}\|^2 \\ & +\langle \mathbf{e} - \mathbf{e}^{'}, \boldsymbol{\mu}_e\rangle + \frac{\rho}{2}\|\mathbf{e} - \mathbf{e}^{'}\|^2 \\ & +\langle \boldsymbol{\theta} - \boldsymbol{\theta}^{'}, \boldsymbol{\mu}_\theta\rangle + \frac{\rho}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}^{'}\|^2 \\ & +\langle 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e}, \boldsymbol{\eta}\rangle + \frac{\rho}{2}\|2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e}\|^2 \\ & +\langle \mathbf{B} - \mathbf{A} - \mathbf{G}, \boldsymbol{\Gamma}\rangle + \frac{\rho}{2}\|\mathbf{B} - \mathbf{A} - \mathbf{G}\|_F^2 \\ \text{s.t.} \quad & \mathbf{A}^{'}, \mathbf{B}^{'}, \mathbf{d}^{'}, \mathbf{e}^{'} \geq 0, \\ & \mathbf{G} \succeq \boldsymbol{\theta}^{'}\boldsymbol{\theta}^{'T},\end{aligned} \tag{11}$$

At each iteration of the ADMM algorithm, the parameters of one block are fixed, and the optimization problem is solved with respect to the parameters of the other block. The details of the aforementioned ADMM procedure for optimizing (11) and its convergence analysis are presented in Appendix D.

**Remark 3.** *The optimal solution obtained from the ADMM algorithm can be different from the one given by Algorithm 1 because we remove the positive semi-definite constraint on* $\mathbf{C}$ *in the latter. We investigate the difference between solutions of two algorithms in three cases: First, we generate a small positive semi-definite matrix* $\mathbf{C}^*$ *and the matrix of confidence intervals* ($\boldsymbol{\Delta}$) *as follows:*

$$\mathbf{C}^* = \begin{bmatrix} 97 & 40 & 92 \\ 40 & 17 & 38 \\ 92 & 38 & 88 \end{bmatrix}, \quad \boldsymbol{\Delta} = \begin{bmatrix} 0.2 & 0.3 & 0.2 \\ 0.3 & 0.1 & 0.2 \\ 0.1 & 0.3 & 0.1 \end{bmatrix}.$$

*Moreover, let* $\mathbf{b}^*$ *and* $\boldsymbol{\delta}$ *are generated as follows:*

$$\mathbf{b}^* = \begin{bmatrix} 6.65 \\ 8.97 \\ 5.40 \end{bmatrix}, \quad \boldsymbol{\delta} = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \end{bmatrix}.$$

*Initializing both algorithms with a random matrix within* $\mathbf{C}_{\min} = \mathbf{C}^* - \boldsymbol{\Delta}$ *and* $\mathbf{C}_{\max} = \mathbf{C}^* + \boldsymbol{\Delta}$, *and a random vector within* $\mathbf{b}_{\min} = \mathbf{b}^* - \boldsymbol{\delta}$ *and* $\mathbf{b}_{\max} = \mathbf{b}^* + \boldsymbol{\delta}$, *ADMM algorithm returns a different solution from Algorithm 1. Besides, the difference in the performance of algorithms during the test phase can be observed in the experiments on synthetic datasets depicted in Figure 5 as well, especially when the number of samples is smaller.*

## 5. Experiments

In this section, we evaluate the performance of RIFLE on a diverse set of inference tasks in the presence of missing values. We compare RIFLE's performance to several state-of-the-art approaches for data imputation on synthetic and real-world datasets. The experiments are designed in a manner that the sensitivity of the model to factors such as the number of samples, data dimension, types, and proportion of missing values can be evaluated. The description of all datasets used in the experiments can be found in Appendix G.

### 5.1. Generating MCAR and MNAR Missing Values

To evaluate RIFLE and other state-of-the-art imputation approaches, we need to have access to the ground-truth values of the missing entries. Hence, we artificially mask a proportion of available data entries and predict them with different imputation methods. A method performs better than others if the predicted missing entries are closer to the ground-truth values. To measure the performance of RIFLE and the existing approaches on a regression task for a given test dataset consisting of $N$ data points, we use normalized root mean squared error (NRMSE), defined as:

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

where $y_i$, $\hat{y}_i$, and $\bar{y}$ represent the true value of the $i$-th data point, the predicted value of the $i$-th data point, and the average of true values of data points, respectively. In all experiments, generated missing entries follow either a missing completely at random (MCAR) or a missing not at random (MNAR) pattern. A discussion on the procedure of generating these patterns can be found in Appendix E.

### 5.2. Tuning Hyper-parameters of RIFLE

The hyper-parameter $c$ in (6) controls the robustness of the model by adjusting the size of confidence intervals. This parameter is tuned by performing a cross-validation procedure over the set $\{0.1, 0.25, 0.5, 1, 2, 5, 10, 20, 50, 100\}$, and the one with the lowest NMRSE is chosen. The default value in the implementation is $c = 1$ since it consistently performs well over different experiments. Furthermore, $\lambda$, the hyper-parameter for the ridge regression regularizer, is tuned by choosing $20\%$ of the data as the validation set from the set $\{0.01, 0.1, 0.5, 1, 2, 5, 10, 20, 50\}$. To tune $K$, the number of bootstrap samples for estimating the confidence intervals, we tried $10, 20, 50$, and $100$. No significant difference is observed in terms of the test performance for the above values.

Furthermore, we tune the hyper-parameters of the competing packages as follows. For KNN-Imputer (Troyanskaya et al., 2001), we try $\{2, 10, 20, 50\}$ for the number of neighbors ($K$) and pick the one with the highest performance. For MICE (Buuren & Groothuis-Oudshoorn, 2010) and Amelia (Honaker et al., 2011), we generate $5$ different imputed data and pick the one with the highest performance on the test data. MissForest has multiple hyper-parameters. We keep the criterion as "MSE" since our performance evaluation measure is NRMSE. Moreover, we tune the number of iterations and number of estimations (number of trees) by checking values from $\{5, 10, 20\}$ and $\{50, 100, 200\}$, respectively. We do not change the structure of the neural networks for MIDA (Gondara & Wang, 2018) and GAIN (Yoon et al., 2018), and the default versions are performed for imputing datasets.

### 5.3. RIFLE Convergence

We presented three algorithms for solving the robust linear regression problem formulated in (6): Projected gradient ascent (Algorithm 1, Nesterov acceleration method (Algorithm 4), and Alternating Direction Method of Multipliers (ADMM) (Algorithm 5) applied on the dual problem. We established the convergence rate of the projected gradient ascent and Nesterov acceleration methods in Theorem ,1 and Theorem ,5 respectively. To investigate the convergence of the ADMM algorithm and its dependence on $\rho$, we perform Algorithm 5 on the Super Conductivity dataset (Description in Appendix G) with 30% MCAR missing values. Figure 3 demonstrates the convergence of the ADMM algorithm for multiple values of $\rho$ applied to the Super Conductivity dataset as described above. As can be observed, decreasing the value of $\rho$ accelerates the ADMM convergence to the optimal value. Note that for $\rho = 0.2$, the objective function is smaller than the final value in the first few iterations. The reason is that for those iterations, the solution is not feasible (as observed in the right figure). The final solution is the optimal **feasible** solution.

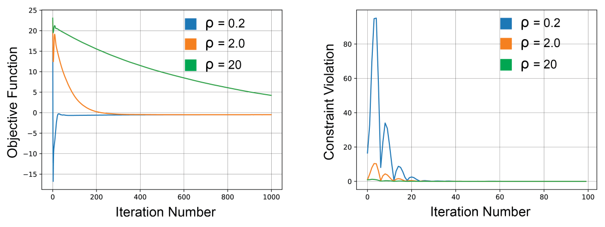

*Figure 3.* Convergence of ADMM algorithm to the optimal solution of Problem (2) for different values of $\rho$. The left plot measures the objective function of Problem (2) per iteration (without considering the constraints), while the right plot demonstrates the constraint violation of the algorithm per iteration. The constraint violation can be measured by adding all regularization terms in the augmented Lagrangian function formulated in Problem (11).

In the next experiment, we compare the three proposed algorithms in terms of the number of iterations required to reach a certain level of test accuracy on the Super Conductivity dataset. The number of training samples is 1000, containing 40% of MCAR missing values on both input features and the target variable. The test dataset contains 2000 samples. As depicted in Figure 4, ADMM and Nesterov's algorithms require less number of iterations to reach the $\epsilon$-optimal solution compared to Algorithm 1. However, the cost per iteration of the ADMM algorithm (Algorithm 5) is higher than the Nesterov acceleration and Algorithm 1.

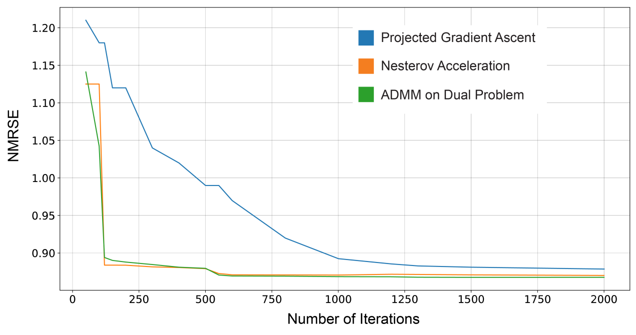

*Figure 4.* The performance of the Nesterov acceleration method, projected gradient ascent, and ADMM on the Super Conductivity dataset vs. the number of iterations.

### 5.4. RIFLE Consistency

In Figure 5, we investigate the consistency of RIFLE on synthetic datasets with different proportions of missing values. The synthetic data has 50 input features following a jointly normal distribution with the mean whose entries are randomly chosen from the interval $(-100, 100)$. Moreover, the covariance matrix equals $\Sigma = SS^T$ where $S$ elements are randomly picked from $(-1, 1)$. The dimension of $S$ is $50 \times 20$. The target variable is a linear function of input features added to a mean zero normal noise with a standard deviation of $0.01$. As depicted in Figure 5, RIFLE requires fewer samples to recover the ground-truth parameters of the model compared to MissForest, KNN Imputer, Expectation Maximization (Dempster et al., 1977), and MICE. Amelia's performance is significantly good since the predictors have a joint normal distribution and the linear underlying model. Note that by increasing the number of samples, the NRMSE of our framework converges to $0.01$, which is the standard deviation of the zero-mean Gaussian noise added to each target value (the dashed line).

### 5.5. Data Imputation via RIFLE

As explained in Section 3, while the primary goal of RIFLE is to learn a robust regression model in the presence of missing values, it can also be used as an imputation tool. We run RIFLE and several state-of-the-art approaches on five datasets from the UCI repository (Dua & Graff, 2017) (Spam, Housing, Clouds, Breast Cancer, and Parkinson datasets) with different proportions of MCAR missing values (the description of the datasets can be found in Appendix G). Then, we compute the NMRSE of imputed entries. Table 1 (see Appendix I) shows the performance of RIFLE compared to other approaches for the datasets

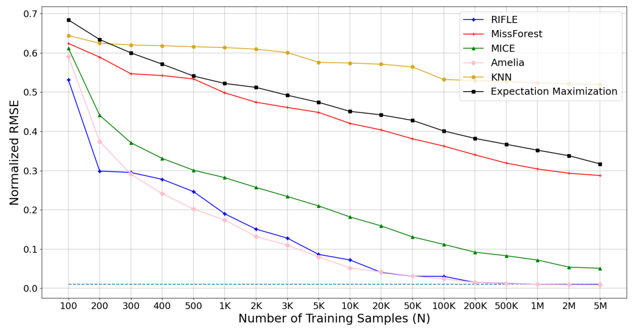

*Figure 5.* Comparing the consistency of RIFLE, MissForest, KNN Imputer, MICE, Amelia, and Expectation Maximization methods on a synthetic dataset containing $40\%$ of missing values.

where the proportion of missing values are relatively high $\left(\frac{n(1-p)}{d} \approx \mathcal{O}(1)\right)$. RIFLE outperforms these methods in almost all cases and performs slightly better than MissForest, which uses a highly non-linear model (random forest) to impute missing values.

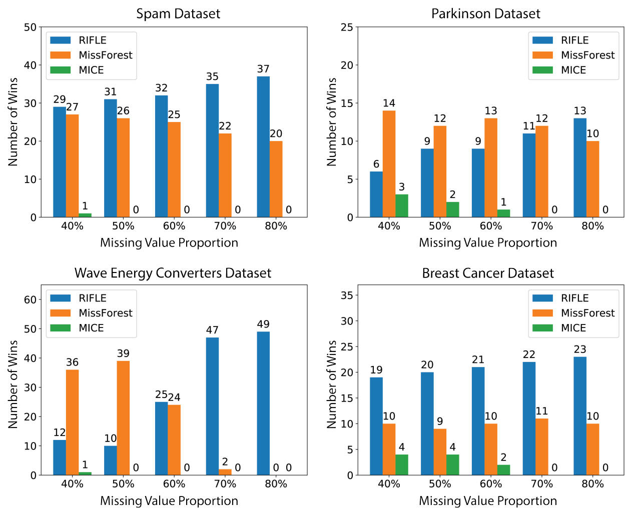

*Figure 6.* Performance Comparison of RIFLE, MICE, and MissForest on four UCI datasets: Parkinson, Spam, Wave Energy Converter, and Breast Cancer. For each dataset, we count the number of features that each method outperforms the others.

### 5.6. Sensitivity of RIFLE to the Number of Samples and Proportion of Missing Values

In this section, we analyze the sensitivity of RIFLE and other state-of-the-art approaches to the number of samples and the proportion of missing values. In the experiment in Figure 6, we create 5 datasets containing $40\%$, $50\%$, $60\%$, $70\%$, and $80\%$ of MCAR missing values, respectively, for four real datasets (Spam, Parkinson, Wave Energy Converter, and Breast Cancer) from UCI Repository (Dua & Graff, 2017) (the description of the datasets can be found in Appendix G).

Given a feature in a dataset containing missing values, we say an imputer wins that feature if the imputation error in terms of NRMSE for that imputer is less than the error of the other imputers. Figure 6 reports the number of features won by each imputer on the created datasets described above. As we observe, the number of wins for RIFLE increases as we increase the proportion of missing values. This observation shows that the sensitivity of RIFLE as an imputer to the proportion of missing values is less than MissForest and MICE in general.

Figure 6 does not show how the NRMSE of one imputer is changed when the proportion of missing values is increased. Next, we analyze the sensitivity of RIFLE and several imputers to change in missing value proportions. Fixing the proportion of missing values, we generate 10 random datasets containing missing values in random locations on the Drive dataset (the description of datasets is available in Appendix G). We impute the missing values for each dataset with RIFLE, MissForest, Mean Imputation, and MICE. Figure 7 shows the average and the standard

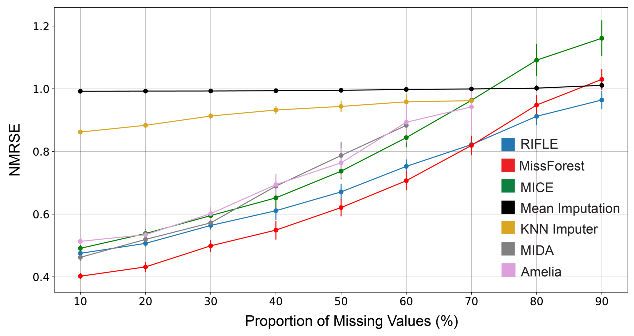

*Figure 7.* Sensitivity of RIFLE, MissForest, Amelia, KNN Imputer, MIDA, and Mean Imputer to the percentage of missing values on the Drive dataset. Increasing the percentage of missing value entries degrades the benchmarks' performance compared to RIFLE. KNN-imputer implementation cannot be executed on datasets containing 80% (or more) missing entries. Moreover, Amelia and MIDA do not converge to a solution when the percentage of missing value entries is higher than 70%.

deviation of these 4 imputers' performances for different proportions of missing values (10% to 90%). Figure 7 depicts the sensitivity of MissForest and RIFLE to the proportion of missing values in the Drive dataset. We select 400 data points for each experiment with different proportions of missing values (from 10% to 90%) and report the average NRMSE of imputed entries. Finally, in Figure 8, we have evaluated RIFLE and other methods on the BlogFeedback dataset (see Appendix G) containing 40% missing values. The results show that RIFLE's performance is less sensitive to decreasing the number of samples.

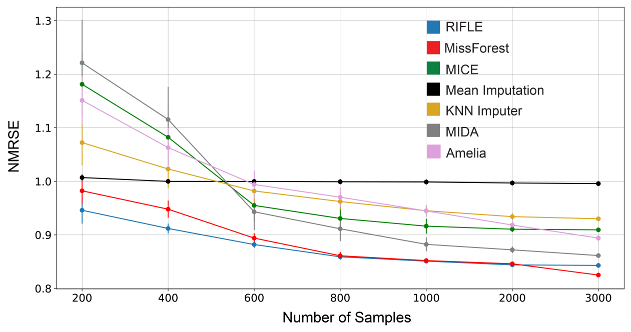

*Figure 8.* Sensitivity of RIFLE, MissForest, MICE, Amelia, Mean Imputer, KNN Imputer, and MIDA to the number of samples for the imputations of Blog Feedback dataset containing 40% of MCAR missing values. When the number of samples is limited, RIFLE outperforms other methods, and its performance is very close to the non-linear imputer MissForest for larger samples.

## 5.7. Performance Comparison on Real Datasets

In this section, we compare the performance of RIFLE to several state-of-the-art approaches, including MICE (Buuren & Groothuis-Oudshoorn, 2010), Amelia (Honaker et al., 2011), MissForest (Stekhoven & Bühlmann, 2012), KNN Imputer (Raghunathan et al., 2001), and MIDA (Gondara & Wang, 2018). There are two primary ways to do this. One method to predict a continuous target variable in a dataset with many missing values is first to impute the missing data with a state-of-the-art package, then run a linear regression. An alternative approach is to directly learn the target variable, as we discussed in Section 3.

Table 2 compares the performance of mean imputation, MICE, MIDA, MissForest, and KNN to that of RIFLE on three datasets: NHANES, Blog Feedback, and superconductivity. Both Blog Feedback and Superconductivity datasets contain 30% of MNAR missing values generated by Algorithm 6, with 10000 and 20000 training samples, respectively. The description of the NHANES data and its distribution of missing values can be found in Appendix G.

**Conclusion:** In this paper, we proposed a distributionally robust optimization framework over the distributions with the low-order marginals within the estimated confidence intervals for inference and imputation of datasets in the presence of missing values. We developed algorithms for robust ridge linear regression with convergence guarantees. In particular, we apply the alternating direction method of multipliers (ADMM) on the dual of the original min-max problem. The resulting algorithm is convergent to the original problem's optimal solution. The performance of the method is evaluated on synthetic and real datasets with different numbers of samples, dimensions, missing value proportions, and types of missing values. In most experiments, RIFLE consistently outperforms other existing methods.

## References

Assländer, J., Cloos, M. A., Knoll, F., Sodickson, D. K., Hennig, J., and Lattanzi, R. Low rank alternating direction method of multipliers reconstruction for mr fingerprinting. *Magnetic resonance in medicine*, 79(1):83–96, 2018.

Barazandeh, B. and Razaviyayn, M. Solving non-convex non-differentiable min-max games using proximal gradient method. *arXiv preprint arXiv:2003.08093*, 2020.

Beaulieu-Jones, B. K., Lavage, D. R., Snyder, J. W., Moore, J. H., Pendergrass, S. A., and Bauer, C. R. Characterizing and managing missing structured data in electronic health records: data analysis. *JMIR medical informatics*, 6(1): e11, 2018.

Bertsimas, D., Pawlowski, C., and Zhuo, Y. D. From predictive methods to missing data imputation: an optimization approach. *The Journal of Machine Learning Research*, 18(1):7133–7171, 2017.

Bubeck, S. Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980*, 2014.

Buuren, S. v. and Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pp. 1–68, 2010.

Cai, Z., Heydari, M., and Lin, G. Iterated local least squares microarray missing value imputation. *Journal of bioinformatics and computational biology*, 4(05):935–957, 2006.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Gabay, D. and Mercier, B. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & mathematics with applications*, 2(1):17–40, 1976.

Gao, R. and Kleywegt, A. J. Distributionally robust stochastic optimization with dependence structure. *arXiv preprint arXiv:1701.04200*, 2017.

Ghahramani, Z. and Jordan, M. I. Supervised learning from incomplete data via an em approach. In *Advances in neural information processing systems*, pp. 120–127, 1994.

Gondara, L. and Wang, K. MIDA: Multiple imputation using denoising autoencoders. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 260–272. Springer, 2018.

He, B. and Yuan, X. On non-ergodic convergence rate of douglas–rachford alternating direction method of multipliers. *Numerische Mathematik*, 130(3):567–577, 2015.

Honaker, J., King, G., and Blackwell, M. Amelia ii: A program for missing data. *Journal of statistical software*, 45(7):1–47, 2011.

Hong, M., Luo, Z.-Q., and Razaviyayn, M. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

Kim, H., Golub, G. H., and Park, H. Missing value estimation for dna microarray gene expression data: local least squares imputation. *Bioinformatics*, 21(2):187–198, 2005.

Little, R. J. and Rubin, D. B. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

Nesterov, Y. E. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pp. 543–547, 1983.

Pedersen, A., Mikkelsen, E., Cronin-Fenton, D., Kristensen, N., Pham, T., Pedersen, L., et al. Missing data and multiple imputation in clinical epidemiological research. clin epidemiol. 2017; 9: 157-66.

Raghunathan, T. E., Lepkowski, J. M., Van Hoewyk, J., Solenberger, P., et al. A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey methodology*, 27(1):85–96, 2001.

Schafer, J. L. *Analysis of incomplete multivariate data*. Chapman and Hall/CRC, 1997.

Shivaswamy, P. K., Bhattacharyya, C., and Smola, A. J. Second order cone programming approaches for handling missing and uncertain data. *Journal of Machine Learning Research*, 7(Jul):1283–1314, 2006.

Sion, M. et al. On general minimax theorems. *Pacific Journal of mathematics*, 8(1):171–176, 1958.

Stekhoven, D. J. and Bühlmann, P. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Sterne, J. A., White, I. R., Carlin, J. B., Spratt, M., Royston, P., Kenward, M. G., Wood, A. M., and Carpenter, J. R. Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj*, 338: b2393, 2009.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

Xia, J., Zhang, S., Cai, G., Li, L., Pan, Q., Yan, J., and Ning, G. Adjusted weight voting algorithm for random forests in handling missing values. *Pattern Recognition*, 69:52–60, 2017.

Xu, H., Caramanis, C., and Mannor, S. Robustness and regularization of support vector machines. *Journal of machine learning research*, 10(7), 2009.

Yoon, J., Jordon, J., and Van Der Schaar, M. Gain: Missing data imputation using generative adversarial nets. *arXiv preprint arXiv:1806.02920*, 2018.

Zhang, T., Ye, S., Zhang, K., Tang, J., Wen, W., Fardad, M., and Wang, Y. A systematic dnn weight pruning framework using alternating direction method of multipliers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 184–199, 2018.

Zhang, X., Song, X., Wang, H., and Zhang, H. Sequential local least squares imputation estimating missing value of microarray data. *Computers in biology and medicine*, 38(10):1112–1120, 2008.

## A. A Review of Missing Value Imputation Methods in the Literature

The fundamental idea behind many data imputation approaches is that the missing values can be predicted based on the available data of other data points and correlated features. One of the most straightforward imputation techniques is to replace missing values by the mean (or median) of that feature calculated from what data is available see Little & Rubin (2019, Chapter 3). However, this naïve approach ignores the correlation between features and does not preserve the variance of features. Another class of imputers has been developed based on the least-square methods (Raghunathan et al., 2001; Kim et al., 2005; Zhang et al., 2008; Cai et al., 2006). Raghunathan et al. (2001) learns a linear model with multivariate Gaussian noise for the feature with the least missing entries. It repeats the same procedure on the updated data to impute the next feature with the least missing entries until all features are completely imputed. One drawback of this approach is that the error from the imputation of previous features can be propagated to subsequent features. To impute entries of a given feature in a dataset, Kim et al. (2005) learns several univariate regression models that consider that feature as the response. Then it takes the average of these predictions as the final value of imputation. This approach fails to learn the correlations involving more than two features.

Many more complex algorithms have been developed for imputation, although many are sensitive to initial assumptions and may not converge. For instance, KNN-Imputer imputes a missing feature of a data point by taking the mean value of the $K$ closest complete data points (Troyanskaya et al., 2001). MissForest, on the other hand, imputes the missing values of each feature by learning a random forest classifier using other training data features (Stekhoven & Bühlmann, 2012). MissForest does not need to assume that all features are continuous (Honaker et al., 2011) or categorical (Schafer, 1997). However, both KNN-imputer and MissForest do not guarantee statistical or computational convergence for their algorithms. Moreover, when the proportion of missing values is high, both are likely to have a severe drop in performance, as demonstrated in Section 5. The Expectation Maximization (EM) algorithm is another popular approach that learns the parameters of a prior distribution on the data using available values based on the EM algorithm of Dempster et al. (1977); see also Ghahramani & Jordan (1994) and Honaker et al. (2011). The EM algorithm is also used in Amelia, which fits a jointly normal distribution to the data using EM and the bootstrap technique (Honaker et al., 2011). While Amelia demonstrates a superior performance on datasets following a normal distribution, it is highly sensitive to the violation of the normality assumption (as discussed in Bertsimas et al. (2017)). Ghahramani & Jordan (1994) adopt the EM algorithm to learn a joint Bernoulli distribution for the categorical data and a joint Gaussian distribution for the continuous variables independently. While those algorithms can be viewed as inference methods based on low-order estimates of moments, they do not consider *uncertainty* in such low-order moments estimates. By contrast, our framework utilizes *robust optimization* to consider the uncertainty around the estimated moments. Moreover, our optimization procedure for imputation and prediction is guaranteed to converge despite some of the algorithms mentioned above.

Another popular method for data imputation is multiple imputations by chained equations (MICE). MICE learns a parametric distribution for each feature conditional on the remaining features. For instance, it assumes that the current target variable is a linear function of other features with a zero-mean Gaussian noise. Each feature can have its distinct distribution and parameters (e.g., Poisson regression, logistic regression). Based on the learned parameters of conditional distributions, MICE can generate one or more imputed datasets (Buuren & Groothuis-Oudshoorn, 2010). More recently, several neural network-based imputers have been proposed. GAIN (Generative Adversarial Imputation Network) learns a generative adversarial network based on the available data and then imputes the missing values using the trained generator (Yoon et al., 2018). One advantage of GAIN over other existing GAN imputers is that it does not need a complete dataset during the training phase. MIDA (Multiple Imputation using Denoising Autoencoders) is an auto-encoder-based approach that trains a denoising auto-encoder on the available data considering the missing entries as noise. Similar to other neural network-based methods, these algorithms suffer from their black-box nature. They are challenging to interpret/explain, making them unpopular in mission-critical healthcare approaches. In addition, no statistical or computational guarantees are provided for these algorithms.

Bertsimas et al. (2017) formulates the imputation task as a constrained optimization problem where the constraints are determined by the underlying classification model such as KNN ($k$-nearest neighbors), SVM (Support Vector Machine), and Decision Trees. Their general framework is non-convex, and the authors relax the optimization for each choice of the cost function using first-order methods. The block coordinate descent algorithm then optimizes the relaxed problem. They show the convergence and accuracy of their proposed algorithm numerically, while a theoretical analysis that guarantees the algorithm's convergence is absent in their work.

## B. Estimating Confidence Intervals of Low-order Moments

In this section, we explain the methodology of estimating confidence intervals for $\mathbb{E}[\mathbf{z}_i]$ and $\mathbb{E}[\mathbf{z}_i\mathbf{z}_j]$. Let $\mathbf{X}^{n\times d}$ and $\mathbf{y}$ be the data matrix and target variables for $n$ given data points respectively whose entries are in $\tilde{\mathbb{R}} = \mathbb{R} \cup \{*\}$, where $*$ symbol represents a missing entry. Moreover, assume that $\mathbf{a}_i$ represents the $i$-th column (feature) of matrix $\mathbf{X}$. We define:

$$\tilde{\mathbf{a}}_i(k) = \begin{cases} \mathbf{a}_i(k) & \text{if } \mathbf{a}_i(k) \neq * \\ 0 & \text{if } \mathbf{a}_i(k) = * \end{cases}$$

Thus, $\tilde{\mathbf{a}}$ is obtained by replacing the missing values with 0. We estimate the confidence intervals for the mean and covariance of features using multiple bootstrap samples on the available data. Let $\mathbf{C}_0[i][j]$ and $\mathbf{\Delta}_0[i][j]$ be the center and the radius of the confidence interval for $\mathbf{C}[i][j]$, respectively. We compute the center of the confidence interval for $\mathbf{C}[i][j]$ as follows:

$$\mathbf{C}_0[i][j] = \frac{1}{m_{ij}} \tilde{\mathbf{a}}_i^T \tilde{\mathbf{a}}_j \tag{12}$$

where $m_i = |\{k : \mathbf{a}_i(k) \neq *\}|$ and $m_{ij} = |\{k : \mathbf{a}_i(k) \neq *, \mathbf{a}_j(k) \neq *\}|$. This estimator is obtained from the rows where both features are available. More precisely, let $\mathbf{M}$ be the mask of the input data matrix $\mathbf{X}$ defined as:

$$\mathbf{M}_{ij} = \begin{cases} 0, & \text{if } \mathbf{X}_{ij} \text{ is missing,} \\ 1, & \text{otherwise.} \end{cases}$$

Assume that $m_{ij} = (\mathbf{M}^T\mathbf{M})_{ij}$, which is the number of rows in the dataset where both features $i$ and $j$ are available. To estimate the confidence intervals for $\mathbf{C}_{ij}$, we use Algorithm 3. First, we select multiple $(K)$ samples of size $N = m_{ij}$ from the rows where both features are available. Each one of these samples with size $m_{ij}$ is obtained by applying a bootstrap sampler (sampling with replacement) on the $m_{ij}$ rows where both features are available. Then, we compute the second-order moment of two features for each sample.

To find the radius of confidence intervals for each given pair $(i, j)$ of features, we choose $k$ different bootstrap samples with length $n$ on the rows where both features $i$ and $j$ are available. Then, we compute $\mathbf{C}_0[i][j]$ of two features in each bootstrap sample. The standard deviation of these estimations determines the radius of the corresponding confidence interval. Algorithm 3 summarizes the required steps for computing the confidence interval radius for the $ij$-th entry of covariance matrix $\mathbf{\Delta}$. Note that the confidence intervals for $\boldsymbol{\mu}$ can be computed similarly. Having $\mathbf{C}_0$ and $\mathbf{\Delta}$, the confidence interval

**Algorithm 3** Estimating Confidence Interval Length $\mathbf{\Delta}_{ij}$ for Feature $i$ and Feature $j$.

```
1: Input: $K$ : Number of bootstrap estimations
2: for $t = 1, \ldots, K$ do
3:     Pick $n$ samples with replacement from the rows where both $i$-th and $j$-th are available.
       Let $(\hat{X}_{i1}, \hat{X}_{j1}), \ldots, (\hat{X}_{in}, \hat{X}_{jn})$ be the $i$-th and $j$-th features of the selected samples
4:     $C_t = \frac{1}{n} \sum_{r=1}^{n} \hat{X}_{ir}\hat{X}_{jr}$
5: end for
6: $\mathbf{\Delta}_{ij} = \text{std}(C_1, C_2, \ldots, C_K)$
```

for the matrix $\mathbf{C}$ is computed as follows:

$$\mathbf{C}_{\min} = \mathbf{C}_0 - c\mathbf{\Delta}$$
$$\mathbf{C}_{\max} = \mathbf{C}_0 + c\mathbf{\Delta},$$

Computing $\mathbf{b}_{\min}$ and $\mathbf{b}_{\max}$ can be done in the same manner. The hyper-parameter $c$ is defined to control the robustness of the model by tuning the length of confidence intervals. A larger $c$ corresponds to bigger confidence intervals and, thus, a more robust estimator. On the other hand, large values for $c$ lead to very large confidence intervals that can adversely affect the performance of the trained model.

**Remark 4.** *Since the computation of confidence intervals for different entries of the covariance matrix are independent of each other, they can be computed in parallel. In particular, if $\gamma$ cores are available, $d/\gamma$ features (columns of the covariance matrix) can be assigned to each one of the available cores.*

## C. Solving Robust Ridge Regression with the Optimal Convergence Rate

The convergence rate of Algorithm 1 to the optimal solution of Problem (6) can be slow in practice since the algorithm requires to do a matrix inversion for updating $\boldsymbol{\theta}$ and applying the box constraint to $\mathbf{C}$ and $\mathbf{b}$ at each iteration. While we update the minimization problem in closed-form with respect to $\boldsymbol{\theta}$, we can speed up the convergence rate of the maximization problem by applying Nesterov's acceleration method to function $g(\mathbf{b}, \mathbf{C})$ in (7). Since function $g$ is the minimum of convex functions, its gradient with respect to $\mathbf{C}$ and $\mathbf{b}$ can be computed using Danskin's theorem. Algorithm 4 describes the steps to optimize Problem (7) using Nesterov's acceleration method.

**Algorithm 4** Applying the Nesterov's Acceleration Method to Robust Linear Regression

```
 1: $\mathbf{C}_0, \mathbf{b}_0, \boldsymbol{\Delta}, \boldsymbol{\delta}, T$
 2: **Initialize**: $\mathbf{C}_1 = \mathbf{C}_0, \mathbf{b}_1 = \mathbf{b}_0, \gamma_0 = 0, \gamma_1 = 1.$
 3: **for** $i = 1, \ldots, T$ **do**
 4:     $\gamma_{i+1} = \frac{1+\sqrt{1+4\gamma_i^2}}{2}$
 5:     $Y_{\mathbf{C}_i} = \mathbf{C}_i + \frac{\gamma_i - 1}{\gamma_{i+1}}(\mathbf{C}_i - \mathbf{C}_{i-1})$
 6:     $\mathbf{C}_{i+1} = \Pi_{\Delta+}\Big(Y_{\mathbf{C}_i} + \frac{1}{L}\boldsymbol{\theta}\boldsymbol{\theta}^T\Big)$
 7:     $Y_{\mathbf{b}_i} = \mathbf{b}_i + \frac{\gamma_i - 1}{\gamma_{i+1}}(\mathbf{b}_i - \mathbf{b}_{i-1})$
 8:     $\mathbf{b}_{i+1} = \Pi_{\boldsymbol{\delta}}(Y_{\mathbf{b}_i} - \frac{2\boldsymbol{\theta}}{L})$
 9:     Set $\boldsymbol{\theta} = (\mathbf{C}_{i+1} + \lambda I)^{-1}\mathbf{b}_{i+1}$
10: **end for**
```

**Theorem 5.** *Assume that $D = \|\mathbf{C}_0 - \mathbf{C}^*\|_F^2 + \|\mathbf{b}_0 - \mathbf{b}^*\|_2^2$. Then Algorithm 4 computes an $\epsilon$-stationary solution of the objective function in (7) in $\mathcal{O}\Big(\sqrt{\frac{LD}{\epsilon}}\Big)$ iterations, where $L = \frac{2}{\lambda}$.*

*Proof.* The proof is relegated to Appendix F. □

## D. ADMM Algorithm For Solving the Dual Problem

For the simplicity of presentation, assume that $\mathbf{c}_1^t = \mathbf{b}_{\min} - \boldsymbol{\mu}_d^t + \rho\mathbf{d}'^t + \boldsymbol{\eta}^t$, $\mathbf{c}_2^t = -\mathbf{b}_{\max} - \boldsymbol{\mu}_e^t + \rho\mathbf{e}'^t - \boldsymbol{\eta}^t$, $\mathbf{c}_3^t = -\boldsymbol{\mu}_{\boldsymbol{\theta}} + \rho\boldsymbol{\theta}'^t - 2\boldsymbol{\eta}^t$, $\mathbf{D}_1^t = \rho\mathbf{A}'^t - \rho\mathbf{G}^t + \boldsymbol{\Gamma}^t - \mathbf{M}_A^t + \mathbf{C}_{\min}$, and $\mathbf{D}_2^t = \rho\mathbf{B}'^t + \rho\mathbf{G}^t - \boldsymbol{\Gamma}^t - \mathbf{M}_B^t - \mathbf{C}_{\max}$. Algorithm 5 describes the ADMM algorithm applied to Problem (10). At each iteration of the ADMM algorithm, the parameters of one block are fixed, and the optimization problem is solved with respect to the parameters of the other block. For the simplicity of presentation, let $\mathbf{c}_1^t = \rho\boldsymbol{\theta}'^t - \boldsymbol{\mu}_{\boldsymbol{\theta}}^t - 2\boldsymbol{\eta}^t$, $\mathbf{c}_2^t = \rho\mathbf{d}'^t - \boldsymbol{\mu}_{\mathbf{d}}^t - \mathbf{b}_{\min} + \boldsymbol{\eta}^t$, $\mathbf{c}_3^t = \rho\mathbf{e}'^t - \boldsymbol{\mu}_e^t + \mathbf{b}_{\max} - \boldsymbol{\eta}^t$, $\mathbf{D}_1^t = \rho\mathbf{A}'^t - \rho\mathbf{G}^t + \boldsymbol{\Gamma}^t - \mathbf{M}_A^t + \mathbf{C}_{\min}$, and $\mathbf{D}_2^t = \rho\mathbf{B}'^t + \rho\mathbf{G}^t - \boldsymbol{\Gamma}^t - \mathbf{M}_B^t - \mathbf{C}_{\max}$.

**Corollary 6.** *If the feasible set of Problem (6) has non-empty interior, then Algorithm 5 converges to an $\epsilon$-optimal solution of Problem (10) in $\mathcal{O}(\frac{1}{\epsilon})$ iterations.*

*Proof.* Since the inner maximization problem, in (6) is convex, and its feasible interior set is not empty, the duality gap is zero by Slater's condition. Thus, according to Theorem 6.1 in He & Yuan (2015), Algorithm 5 converges to an optimal solution of the primal-dual problem with a linear rate. Moreover, the sequence of constraint residuals converges to zero with a linear rate as well. □

We have two non-trivial problems containing positive semi-definite constraints. The sub-problem with respect to $\mathbf{G}$ can be written as:

$$\begin{aligned} &\min_{\mathbf{G}} \ \langle \mathbf{B}^t - \mathbf{A}^t - \mathbf{G}, \boldsymbol{\Gamma}^t \rangle + \frac{\rho}{2}\|\mathbf{B}^t - \mathbf{A}^t - \mathbf{G}\|_F^2 \\ &\text{s.t.} \ \ \mathbf{G} \succeq \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT}, \end{aligned} \tag{13}$$

By completing the square, and changing the variable $\mathbf{G}' = \mathbf{G} - \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT}$, equivalently we require to solve the following

**Algorithm 5** Applying ADMM to the Dual Problem of Robust Linear Regression

```
1: Given: $\mathbf{b}_{\min}, \mathbf{b}_{\max}, \mathbf{C}_{\min}, \mathbf{C}_{\max}, \lambda, \rho$
2: Initialize: $\mathbf{C}_1 = \mathbf{C}_0, \mathbf{b}_1 = \mathbf{b}_0, \gamma_0 = 0, \gamma_1 = 1.$
3: for $t = 0, \dots, T$ do
4:     $\boldsymbol{\theta}^{t+1} = \frac{1}{6\lambda+7\rho}(2\mathbf{c}_1^t - 2\mathbf{c}_2^t - 3\mathbf{c}_3^t)$
5:     $\mathbf{d}^{t+1} = \frac{1}{6\lambda+7\rho}(\frac{6\rho+4\lambda}{\rho}\mathbf{c}_1^t + \frac{4\rho+4\lambda}{\rho}\mathbf{c}_2^t + 2\mathbf{c}_3^t)$
6:     $\mathbf{e}^{t+1} = \frac{2}{6\lambda+7\rho}(\frac{\rho+2\lambda}{\rho}\mathbf{c}_1^t + \frac{3\rho+2\lambda}{\rho}\mathbf{c}_2^t - \mathbf{c}_3^t)$
7:     $\mathbf{A}'^{t+1} = \max(\mathbf{A}^t + \frac{\mathbf{M}_A^t}{\rho}, 0)$
8:     $\mathbf{B}'^{t+1} = \max(\mathbf{B}^t + \frac{\mathbf{M}_B^t}{\rho}, 0)$
9:     $\mathbf{G}^{t+1} = [\mathbf{B}^t - \mathbf{A}^t + \frac{\boldsymbol{\Gamma}^t}{\rho} - \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT}]_+ + \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT}$
10:    $\mathbf{d}'^{t+1} = \max(\mathbf{d}^t + \frac{\boldsymbol{\mu}_d^t}{\rho}, 0)$
11:    $\mathbf{e}'^{t+1} = \max(\mathbf{e}^t + \frac{\boldsymbol{\mu}_e^t}{\rho}, 0)$
12:    $\boldsymbol{\theta}'^{t+1} = \text{argmin}_{\boldsymbol{\theta}'} \quad \|\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}'\|^2 + \langle \boldsymbol{\mu}_{\boldsymbol{\theta}}^t, \boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}' \rangle \quad \text{s.t. } \mathbf{G}^{t+1} \succeq \boldsymbol{\theta}'^T\boldsymbol{\theta}'$
13:    $\mathbf{A}^{t+1} = \frac{1}{3\rho}(2\mathbf{D}_1^t + \mathbf{D}_2^t)$
14:    $\mathbf{B}^{t+1} = \frac{1}{3\rho}(\mathbf{D}_1^t + 2\mathbf{D}_2^t)$
15:    $\mathbf{M}_A^{t+1} = \mathbf{M}_A^t + \rho(\mathbf{A}^{t+1} - \mathbf{A}'^{t+1})$
16:    $\mathbf{M}_B^{t+1} = \mathbf{M}_B^t + \rho(\mathbf{B}^{t+1} - \mathbf{B}'^{t+1})$
17:    $\boldsymbol{\mu}_d^{t+1} = \boldsymbol{\mu}_d^t + \rho(\mathbf{d}^{t+1} - \mathbf{d}'^{t+1})$
18:    $\boldsymbol{\mu}_e^{t+1} = \boldsymbol{\mu}_e^t + \rho(\mathbf{e}^{t+1} - \mathbf{e}'^{t+1})$
19:    $\boldsymbol{\mu}_{\boldsymbol{\theta}}^{t+1} = \boldsymbol{\mu}_\theta^t + \rho(\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}'^{t+1})$
20:    $\boldsymbol{\eta}^{t+1} = \boldsymbol{\eta}^t + \rho(2\boldsymbol{\theta}^{t+1} - \mathbf{d}^{t+1} + \mathbf{e}^{t+1})$
21:    $\boldsymbol{\Gamma}^{t+1} = \boldsymbol{\Gamma}^t + \rho(\mathbf{B}^{t+1} - \mathbf{A}^{t+1} - \mathbf{G}^{t+1})$
22: end for
```

problem:

$$\begin{aligned} \min_{\mathbf{G}'} \ & \frac{\rho}{2}\|\mathbf{G}' - (\mathbf{B}^t - \mathbf{A}^t - \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT} + \frac{\boldsymbol{\Gamma}^t}{\rho})\|_F^2 \\ \text{s.t.} \ & \boldsymbol{G}' \succeq 0, \end{aligned} \tag{14}$$

Thus, $\mathbf{G}'^* = [\mathbf{B}^t - \mathbf{A}^t + \frac{\boldsymbol{\Gamma}^t}{\rho} - \boldsymbol{\theta}'^t\boldsymbol{\theta}'^{tT}]_+$, where $[\mathbf{A}]_+$ is the projection to the set of PSD matrices, which can be done by setting the negative eigenvalues of $\mathbf{A}$ in its singular value decomposition to zero.

The other non-trivial sub-problem in Algorithm (5) is the minimization with respect to $\boldsymbol{\theta}'$ (Line 10). By completing the square, it can be equivalently formulated as:

$$\begin{aligned} \min_{\boldsymbol{\theta}'} \ & \|\boldsymbol{\theta}' - (\boldsymbol{\theta}^{t+1} + \frac{\boldsymbol{\mu}_{\boldsymbol{\theta}}^t}{\rho})\|_2^2 \\ \text{s.t.} \ & \mathbf{G}^{t+1} \succeq \boldsymbol{\theta}'\boldsymbol{\theta}'^T, \end{aligned} \tag{15}$$

Let $\mathbf{G} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$ be the singular value decomposition of the matrix $\mathbf{G}$ where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of the matrix G. Set $\boldsymbol{\alpha} = \boldsymbol{\theta}^{t+1} + \frac{\boldsymbol{\mu}_{\boldsymbol{\theta}}^t}{2}$. Since $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$, we have:

$$\|\boldsymbol{U}^T\boldsymbol{\theta} - \boldsymbol{U}^T\boldsymbol{\alpha}\|^2 = \|\boldsymbol{\theta} - \boldsymbol{\alpha}\|_2^2$$

Set $\boldsymbol{\beta} = \boldsymbol{U}^T\boldsymbol{\theta}'$, then Problem (15) can be reformulated as:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \ & \|\boldsymbol{\beta} - \boldsymbol{U}^T\boldsymbol{\alpha}\|_2^2 \\ \text{s.t.} \ & \boldsymbol{\beta}\boldsymbol{\beta}^T \preceq \boldsymbol{\Lambda}. \end{aligned} \tag{16}$$

Note that the constraint of the above optimization problem is equivalent to the following:

$$\boldsymbol{\beta}\boldsymbol{\beta}^T \preceq \boldsymbol{\Lambda} \Leftrightarrow \begin{bmatrix} 1 & \boldsymbol{\beta}^T \\ \boldsymbol{\beta} & \boldsymbol{\Lambda} \end{bmatrix} \succeq 0 \Leftrightarrow \boldsymbol{\beta}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{\beta} \leq 1 \Leftrightarrow \sum_{i=1}^{n} \frac{\boldsymbol{\beta}_i^2}{\lambda_i} \leq 1,$$

where $\lambda_i = \boldsymbol{\Lambda}_{ii}$ Since the block matrix is symmetric, using Schur Complement, it is positive semi-definite if and only if $\boldsymbol{\Lambda}$ is positive semi-definite and $1 - \boldsymbol{\beta}^t \boldsymbol{\Lambda}^{-1} \boldsymbol{\beta} \geq 0$ (the third inequality above).

Set $\boldsymbol{\gamma} = \boldsymbol{U}^T \boldsymbol{\alpha}$, then we can write Problem (16) as:

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \ \|\boldsymbol{\beta} - \boldsymbol{\gamma}\|_2^2 \\ &\text{s.t.} \ \ \textstyle\sum_{i=1}^n \frac{\boldsymbol{\beta}_i^2}{\lambda_i} \leq 1, \end{aligned} \tag{17}$$

It can be easily shown that the optimal solution has the form $\boldsymbol{\beta}_i^* = \frac{\gamma_i}{1+\frac{\mu^*}{\lambda_i}}$, where $\boldsymbol{\mu}^*$ is the optimal Lagrangian multiplier corresponding to the constraint of Problem (17). The bisection algorithm can efficiently obtain the optimal Lagrangian multiplier. Having $\boldsymbol{\beta}^*$, the optimal $\boldsymbol{\theta}$ can be computed by solving the linear equation $\boldsymbol{U}^T \boldsymbol{\theta}^* = \boldsymbol{\beta}^*$.

## E. Generating Missing Values Patterns in Numerical Experiments

In this appendix, we define MCAR and MNAR patterns and discuss how to generate them in a given dataset. Formally, the distribution of missing values in a dataset follows a missing completely at random (MCAR) pattern if the probability of having a missing value for a given entry is constant, independent of other available and missing entries. On the other hand, a dataset follows a Missing At Random (MAR) pattern if the missingness of each entry only depends on the available data of other features. Finally, if the distribution of missing values does not follow an MCAR or MAR pattern, we call it missing not at random (MNAR).

To generate the MCAR pattern on a given dataset, we fix a constant probability $0 < p < 1$ and make each data entry unavailable with the probability of $p$. On the other hand, the generation of the MNAR pattern is based on the idea that if the value of an entry is farther from the mean of its corresponding feature, then the probability of missingness for that entry is larger.

The generation of the MNAR pattern is based on the idea that if the value of an entry is farther from the mean of its corresponding feature, then the probability of missingness for that entry is larger. Algorithm 6 describes the procedure of generating MNAR missing values for a given column of a dataset:

**Algorithm 6** Generating MNAR Pattern for a Given Column of a Dataset

```
1: Input: $x_1, x_2, \dots, x_n$: The entries of the current column in the dataset, $a, b$: Hyper-parameters controlling the
   percentage of missing values
2: Initialize: Set $\mu = \frac{1}{n}\sum_{i=1}^n x_i$ and $\sigma^2 = \frac{1}{n}\sum_{i=1}^n x_i^2 - \mu^2$.
3: for $i = 1, \dots, n$ do
4:     $x_i' = \frac{x_i - \mu}{\sigma}$
5:     $p_i = F(a|x_i'| + b)$
6:     Set $x_i = *$ with probability of $p_i$
7: end for
```

Note that $F$ in the above algorithm is the cumulative distribution function of a standard Gaussian random variable. $a$ and $b$ control the percentage of missing values in the given column. As $a$ and $b$ increase, the probability of having more missing values is higher. Since the availability of each data entry depends on its value, the generated missing pattern is missing not at random (MNAR).

## F. Proof of Lemmas and Theorems

In this appendix, we prove all lemmas and theorems presented in the article.

First, we prove the following lemma that is useful in several convergence proofs:

**Lemma 7.** *The Lipschitz constant of the gradient of the function $g$ defined in Problem (7) is equal to $L = \frac{2}{\lambda}$.*

*Proof.* Since, the problem is convex in $\boldsymbol{\theta}$ and concave in $\mathbf{C}$ and $\mathbf{b}$, we have:

$$\min_{\boldsymbol{\theta}} \quad \max_{\mathbf{C},\mathbf{b}} \quad \boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} - 2\mathbf{b}^T\boldsymbol{\theta} + \lambda\|\boldsymbol{\theta}\|^2 = -\min_{\mathbf{C},\mathbf{b}} \quad \max_{\boldsymbol{\theta}} \quad -\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} + 2\mathbf{b}^T\boldsymbol{\theta} - \lambda\|\boldsymbol{\theta}\|^2$$

The new objective function is convex in $\mathbf{C}$ and $\mathbf{b}$ and strongly concave with respect to $\boldsymbol{\theta}$. According to Lemma 1 in Barazandeh & Razaviyayn (2020), $g^{'} = -g = \max_{\boldsymbol{\theta}} -\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} + 2\mathbf{b}^T\boldsymbol{\theta} - \lambda\|\boldsymbol{\theta}\|^2$ is Lipschitz continuous with the Lipschitz constant equal to:

$$L_g = L_{g'} = L_{11} + \frac{L_{12}}{\sigma},$$

where $\sigma$ is the strong-concavity modulus of $-\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta}+2\mathbf{b}^T\boldsymbol{\theta}-\lambda\|\boldsymbol{\theta}\|^2$. Assume that $h(\boldsymbol{\theta},\mathbf{C},\mathbf{b}) \triangleq -\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta}+2\mathbf{b}^T\boldsymbol{\theta}-\lambda\|\boldsymbol{\theta}\|^2$ and $L_{11}$, $L_{12}$ are defined as follows:

$$\|\nabla_{\mathbf{b},\mathbf{C}} h(\boldsymbol{\theta},\mathbf{b}_1,\mathbf{C}_1) - \nabla_{\mathbf{b},\mathbf{C}} h(\boldsymbol{\theta},\mathbf{b}_2,\mathbf{C}_2)\| \leq L_{11}\|(\mathbf{C}_1,\mathbf{b}_1) - (\mathbf{C}_2,\mathbf{b}_2)\|$$
$$\|\nabla_{\theta} h(\boldsymbol{\theta},\mathbf{b}_1,\mathbf{C}_1) - \nabla_{\boldsymbol{\theta}} h(\boldsymbol{\theta},\mathbf{b}_2,\mathbf{C}_2)\| \leq L_{12}\|(\mathbf{C}_1,\mathbf{b}_1) - (\mathbf{C}_2,\mathbf{b}_2)\|$$

It is easy to see that $L_{11} = 0$ and $L_{12} = 2\max\|\boldsymbol{\theta}\|_2 \leq 2$, which completes the proof. □

**Proof of Theorem 1**: Since the set of feasible solutions for $\mathbf{b}$ and $\mathbf{C}$ defines a compact set, and function $g$ is a concave function with respect to $\mathbf{b}$ and $\mathbf{C}$, the projected gradient ascent algorithm converges to the global maximizer of $g$ in $T = \mathcal{O}(\frac{LD^2}{\epsilon})$ iterations (Bubeck, 2014), where $D = \|\mathbf{C}_0 - \mathbf{C}^*\|_F^2 + \|\mathbf{b}_0 - \mathbf{b}^*\|_2^2$ and $L$ is the Lipschitz constant of function $g$, which is equal to $\frac{2}{\lambda}$ according to Lemma 7.

**Proof of Theorem 2**: First, note that if we multiply the objective function by $-1$, Problem (6) can be equivalently formulated as:

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & \min_{\mathbf{C},\mathbf{b}} -\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} + 2\mathbf{b}^T\boldsymbol{\theta} - \lambda\|\boldsymbol{\theta}\|_2^2 \\ \text{s.t.} \quad & \quad -\mathbf{C} + \mathbf{C}_{\min} \leq 0, \\ & \quad \mathbf{C} - \mathbf{C}_{\max} \leq 0, \\ & \quad -\mathbf{b} + \mathbf{b}_{\min} \leq 0, \\ & \quad \mathbf{b} - \mathbf{b}_{\max} \leq 0, \\ & \quad -\mathbf{C} \preceq 0 \end{aligned} \tag{18}$$

If we assign $\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{e}, \mathbf{H}$ to the constraints respectively, then the Lagrangian function can be written as:

$$\begin{aligned} L(\mathbf{C},\mathbf{b},\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}) = & -\boldsymbol{\theta}^T\mathbf{C}\boldsymbol{\theta} + 2\mathbf{b}^T\boldsymbol{\theta} + \langle\mathbf{A}, -\mathbf{C}+\mathbf{C}_{\min}\rangle \\ & + \langle\mathbf{B}, \mathbf{C}-\mathbf{C}_{\max}\rangle + \langle\mathbf{d}, -\mathbf{b}+\mathbf{b}_{\min}\rangle \\ & + \langle\mathbf{e}, \mathbf{b}-\mathbf{b}_{\max}\rangle - \langle\mathbf{C},\mathbf{H}\rangle - \lambda\|\boldsymbol{\theta}\|_2^2, \end{aligned} \tag{19}$$

The dual problem is defined as:

$$\max_{\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}} \min_{\mathbf{C},\mathbf{b}} L(\mathbf{C},\mathbf{b},\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}) \tag{20}$$

The minimization of $L$ takes the following form:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{b}} \; & \langle\mathbf{C}, -\boldsymbol{\theta}\boldsymbol{\theta}^T - \mathbf{A} + \mathbf{B} - \mathbf{H}\rangle + \langle\mathbf{b}, 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e}\rangle - \lambda\|\boldsymbol{\theta}\|_2^2 \\ & -\langle\mathbf{B},\mathbf{C}_{\max}\rangle + \langle\mathbf{A},\mathbf{C}_{\min}\rangle - \langle\mathbf{e},\mathbf{b}_{\max}\rangle + \langle\mathbf{d},\mathbf{b}_{\min}\rangle, \end{aligned} \tag{21}$$

To avoid $-\infty$ value for the above minimization problem, it is required to set $-\boldsymbol{\theta}\boldsymbol{\theta}^T - \mathbf{A} + \mathbf{B} - \mathbf{H}$ and $2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e}$ to zero. Thus the dual problem of (18) is formulated as:

$$\begin{aligned} \max_{\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}} \quad & \mathbf{b}_{\min}^T\mathbf{d} - \mathbf{b}_{\max}^T\mathbf{e} + \langle \mathbf{C}_{\min}, \mathbf{A}\rangle - \langle \mathbf{C}_{\max}, \mathbf{B}\rangle - \lambda\|\boldsymbol{\theta}\|_2^2 \\ \text{s.t.} \quad & -\boldsymbol{\theta}\boldsymbol{\theta}^T - \mathbf{A} + \mathbf{B} - \mathbf{H} = 0, \\ & 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e} = 0, \\ & \mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{e} \geq 0, \\ & \mathbf{H} \succeq 0 \end{aligned} \tag{22}$$

Since the duality gap is zero, Problem (6) can be equivalently formulated as:

$$\begin{aligned} \max_{\boldsymbol{\theta},\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{e},\mathbf{H}} \quad & \mathbf{b}_{\min}^T\mathbf{d} - \mathbf{b}_{\max}^T\mathbf{e} + \langle \mathbf{C}_{\min}, \mathbf{A}\rangle - \langle \mathbf{C}_{\max}, \mathbf{B}\rangle - \lambda\|\boldsymbol{\theta}\|^2 \\ \text{s.t.} \quad & -\boldsymbol{\theta}\boldsymbol{\theta}^T - \mathbf{A} + \mathbf{B} - \mathbf{H} = 0, \\ & 2\boldsymbol{\theta} - \mathbf{d} + \mathbf{e} = 0, \\ & \mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{e} \geq 0, \\ & \mathbf{H} \succeq 0. \end{aligned} \tag{23}$$

We can multiply the objective function by $-1$ and change the maximization to minimization, which gives the dual problem described in (2).

**Proof of Theorem** 5 Algorithm 4 applies the projected Nesterov acceleration method on the concave function $g$. As proved in Nesterov (1983), the rate of convergence of this method conforms to the lower bound of first-order oracles for the general convex minimization (concave maximization) problems, which is $\mathcal{O}(\sqrt{\frac{LD^2}{\epsilon}})$. We compute the Lipschitz constant $L$ that appeared in the iteration complexity bound by Lemma 7.

## G. Dataset Descriptions

In this section, we introduce the datasets used in Section 5 to evaluate the performance of RIFLE. Except for the NHANES, all other datasets contain no missing values. For those datasets, we generate MCAR and MNAR missing values artificially (for MNAR patterns, we apply Algorithm 6 to the datasets).

- **NHANES**: The percentage of missing values varies for different features of the NHANES dataset. There are two sources of missing values in NHANES data: Missing entries during data collection and missing entries resulting from merging different datasets in the NHANES collection. On average, approximately 20% of data is missing.
- **Super Conductivity**[2]: Super Conductivity datasets contains 21263 samples describing superconductors and their relevant features (81 attributes). All features are continuous, and the assigned task is to predict the critical temperature based on the given 81 features. We have used this dataset in experiments summarized in Figure 3, Figure 4, and Table 2.
- **BlogFeedback**[3]: BlogFeedback data is a collection of 280 features extracted from HTML-documents of the blog posts. The assigned task is to predict the number of comments in the upcoming 24 hours based on the features of more than $60K$ data training data points. The test dataset is fixed and is originally separated from the training data. The dataset is used in experiments described in Table 2.
- **Breast Cancer**(Prognostic)[4]: The dataset consists of 34 features and 198 instances. Each record represents follow-up data for one breast cancer case collected in 1984. We have done several experiments to impute the MCAR missing values generated artificially with different proportions. The results are depicted in Table 1 and Figure 6.
- **Parkinson**[5]: The dataset describes a range of biomedical voice recording from 31 people, 23 with Parkinson's disease (PD). The assigned task is to discriminate healthy people from those with PD. There are 193 records and 23 features in the dataset. The dataset is processed similarly to the Breast Cancer dataset and used in the same experiments.

[2]`https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data`
[3]`https://archive.ics.uci.edu/ml/datasets/BlogFeedback`
[4]`https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Prognostic)`
[5]`https://archive.ics.uci.edu/ml/datasets/parkinsons`

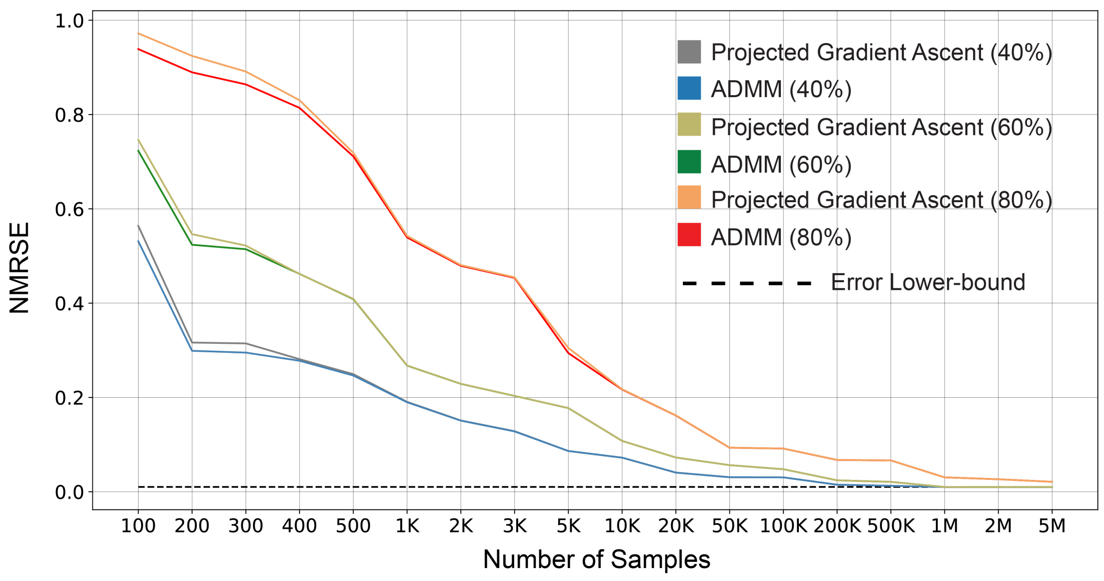

*Figure 9.* Consistency of ADMM (Algorithm 5) and Projected Gradient Ascent on function $g$ (Algorithm 1) on the synthetic datasets with 40%, 60% and 80% missing values.

- **Spam Base**[6]: The dataset consists of 4601 instances and 57 attributes. The assigned classification task is to predict whether the email is spam. To evaluate different imputation methods, we randomly mask a proportion of data entries and impute them with different approaches. The results are depicted in Table 1 and Figure 6.
- **Boston Housing**[7]: Boston Housing dataset contains 506 instances and 14 columns. We generate random missing entries with different proportions and impute them with RIFLE and several state-of-the-art approaches. The results are demonstrated in Table 1 and Figure 6.
- **Cloud**[8]: The dataset has 1024 instances and 10 features extracted from clouds images. We use this dataset in experiments depicted in Table 1 with 70% artificial MCAR missing values.
- **Wave Energy Converters**[9]: We sample a subset of 3000 instances with 49 features from the original Wave Energy Converter dataset. We have executed several imputation methods on the dataset, and the results are shown in Figure 6.
- **Sensorless Drive Diagnosis**[10]: The 49 continuous features in this dataset are extracted from electric current drive signals, and the associated classification task is to determine the condition of device's motor. We choose different random samples with size 400 to run experiments (imputation) in Figure 7.

## H. Further Discussion on the Consistency of RIFLE

The three developed algorithms in Section 3 for solving robust ridge regression are all consistent. To show this, we have generated a synthetic dataset with 50 input features following a jointly normal distribution. As observed in Figure 9, by increasing the number of samples, the NRMSE of all three algorithms converges to 0.01, which is the standard deviation of the zero-mean Gaussian noise added to each target value (the dashed line). The pattern can be observed for different percentages of missing values.

[6]https://archive.ics.uci.edu/ml/datasets/spambase
[7]https://www.kaggle.com/c/boston-housing
[8]https://archive.ics.uci.edu/ml/datasets/Cloud
[9]https://archive.ics.uci.edu/ml/datasets/Wave+Energy+Converters
[10]https://archive.ics.uci.edu/ml/datasets/dataset+for+sensorless+drive+diagnosis

## I. Imputation Table

| Dataset Name | RIFLE | QRIFLE | MICE | Amelia | GAIN | MissForest | MIDA | EM |
|---|---|---|---|---|---|---|---|---|
| Spam (30%) | $\underline{0.87}$ ±0.009 | **0.82** ±0.009 | 1.23 ±0.012 | 1.26 ±0.007 | 0.91 ±0.005 | 0.90 ±0.013 | 0.97 ±0.008 | 0.94 ± 0.004 |
| Spam (50%) | $\underline{0.90}$ ±0.013 | **0.86** ±0.014 | 1.29 ±0.018 | 1.33 ±0.024 | 0.93 ±0.015 | 0.92 ±0.011 | 0.99 ±0.011 | 0.97 ± 0.008 |
| Spam (70%) | $\underline{0.92}$ ±0.017 | **0.91** ±0.019 | 1.32 ±0.028 | 1.37 ±0.032 | 0.97 ±0.014 | 0.95 ±0.016 | 0.99 ±0.018 | 0.98 ± 0.017 |
| Housing (30%) | 0.86 ±0.015 | 0.89 ±0.018 | 1.03 ±0.024 | 1.02 ±0.016 | **0.82** ±0.015 | $\underline{0.84}$ ±0.018 | 0.93 ±0.025 | 0.95 ± 0.011 |
| Housing (50%) | **0.88** ±0.021 | 0.90 ±0.024 | 1.14 ±0.029 | 1.09 ±0.027 | **0.88** ±0.019 | **0.88** ±0.018 | 0.98 ±0.029 | 0.96 ± 0.016 |
| Housing (70%) | **0.92** ±0.026 | 0.95 ±0.028 | 1.22 ±0.036 | 1.18 ±0.038 | 0.95 ±0.027 | $\underline{0.93}$ ±0.024 | 1.02 ±0.037 | 0.98 ± 0.017 |
| Clouds (30%) | 0.81 ±0.018 | 0.79 ±0.019 | 0.98 ±0.024 | 1.04 ±0.027 | $\underline{0.76}$ ±0.021 | **0.71** ±0.011 | 0.83 ±0.022 | 0.86 ± 0.013 |
| Clouds (50%) | 0.84 ±0.026 | 0.84 ±0.028 | 1.10 ±0.041 | 1.13 ±0.046 | $\underline{0.82}$ ±0.027 | **0.75** ±0.023 | 0.88 ±0.033 | 0.89 ± 0.018 |
| Clouds (70%) | $\underline{0.87}$ ±0.029 | 0.90 ±0.033 | 1.16 ±0.044 | 1.19 ±0.048 | 0.89 ±0.035 | **0.81** ±0.031 | 0.93 ±0.044 | 0.92 ± 0.023 |
| Breast Cancer (30%) | **0.52** ±0.023 | $\underline{0.54}$ ±0.027 | 0.74 ±0.031 | 0.81 ±0.032 | 0.58 ±0.024 | 0.55 ±0.016 | 0.70 ±0.026 | 0.67 ± 0.014 |
| Breast Cancer (50%) | **0.56** ±0.026 | 0.59 ±0.027 | 0.79 ±0.029 | 0.85 ±0.033 | 0.64 ±0.025 | $\underline{0.59}$ ±0.022 | 0.76 ±0.035 | 0.69 ± 0.022 |
| Breast Cancer (70%) | **0.59** ±0.031 | 0.65 ±0.034 | 0.86 ±0.042 | 0.92 ±0.044 | 0.70 ±0.037 | $\underline{0.63}$ ±0.028 | 0.82 ±0.035 | 0.67 ± 0.014 |
| Parkinson (30%) | 0.57 ±0.016 | 0.55 ±0.016 | 0.71 ±0.019 | 0.67 ±0.021 | **0.53** ±0.015 | $\underline{0.54}$ ±0.010 | 0.62 ±0.017 | 0.64 ± 0.011 |
| Parkinson (50%) | $\underline{0.62}$ ±0.022 | 0.64 ±0.025 | 0.77 ±0.029 | 0.74 ±0.034 | **0.61** ±0.022 | 0.65 ±0.014 | 0.71 ±0.027 | 0.69 ± 0.022 |
| Parkinson (70%) | **0.67** ±0.027 | 0.74 ±0.033 | 0.85 ±0.038 | 0.82 ±0.037 | $\underline{0.69}$ ±0.031 | 0.73 ±0.022 | 0.78 ±0.038 | 0.75 ± 0.029 |

*Table 1.* Performance comparison of RIFLE, QRIFLE (Quadratic RIFLE), and state-of-the-art methods on several UCI datasets. We applied to impute methods on three different missing-value proportions for each dataset. The best imputer is highlighted with bold font, and the second-best imputer is underlined. Each experiment is done 5 times, and the average and the standard deviation of performances are reported.

| Methods | Datasets | | |
|---|---|---|---|
| | Super Conductivity | Blog Feedback | NHANES |
| Regression on Complete Data | 0.4601 | 0.7432 | 0.6287 |
| **RIFLE** | **0.4873** ± 0.0036 | 0.8326 ± 0.0085 | **0.6304** ± 0.0027 |
| Mean Imputer + Regression | 0.6114 ± 0.0006 | 0.9235 ± 0.0003 | 0.6329 ± 0.0008 |
| MICE + Regression | 0.5078 ± 0.0124 | 0.8507 ± 0.0325 | 0.6612 ± 0.0282 |
| EM + Regression | 0.5172 ± 0.0162 | 0.8631 ± 0.0117 | 0.6392 ± 0.0122 |
| MIDA Imputer + Regression | 0.5213 ± 0.0274 | 0.8394 ± 0.0342 | 0.6542 ± 0.0164 |
| MissForest | 0.4925 ± 0.0073 | **0.8191** ± 0.0083 | 0.6365 ± 0.0094 |
| KNN Imputer | 0.5438 ± 0.0193 | 0.8828 ± 0.0124 | 0.6427 ± 0.0135 |

*Table 2.* Normalized RMSE of RIFLE and several state-of-the-art Methods on Superconductivity, blog feedback, and NHANES datasets. The first two datasets contain $30\%$ Missing Not At Random (MNAR) missing values in the training phase generated by Algorithm 6. Each method applied 5 times to each dataset, and the result is reported as the average performance ± standard deviation of experiments in terms of NRMSE.

Since MICE and MIDA cannot predict values during the test phase without data imputation, we use them in a pre-processing stage to impute the data. Then we apply the linear regression to the imputed dataset. On the other hand, RIFLE, KNN imputer, and MissForest can predict the target variable without imputing the training dataset. Table 2 shows that RIFLE outperforms all other state-of-the-art approaches executed on the three mentioned datasets. In particular, RIFLE outperforms MissForest, while the underlying model used by RIFLE is simpler (linear) compared to the nonlinear random forest model utilized by Missforest.