Shuffled Linear Regression with Erroneous Observations

Samer S. Saab
Electrical and Computer Engineering
Lebanese American University
Byblos, Lebanon
ssaab@lau.edu.lb

Khaled Kamal Saab
Electrical Engineering
Stanford University
Palo Alto, CA
ksaab@stanford.edu

Samer S. Saab, Jr.
Electrical Engineering
Pennsylvania State University
State College, PA
sys5880@psu.edu

***Abstract*—Linear regression with shuffled labels is the problem of performing a linear regression fit on datasets whose labels are unknowingly shuffled with respect to their inputs. Such a problem relates to different applications such as genome sequence assembly, sampling and reconstruction of spatial fields, and communication networks. Existing methods are either applicable only to data with limited observation errors, work only for partially shuffled data, sensitive to initialization, and/or work only with small dimensions. This paper tackles this problem in its full generality using stochastic approximation, which is based on a first-order permutation-invariant constraint. We propose an optimal recursive algorithm that updates the estimate from the underdetermined function that is based on that permutation-invariant constraint. The proposed algorithm aims for per-iteration minimization of the mean square estimate error. Although our algorithm is sensitive to initialization errors, to the best of our knowledge, the resulting method is the first working solution for arbitrary large dimensions and arbitrary large observation errors while its computation throughput appears insignificant. Numerical simulations show that our method with shuffled datasets can outperform the ordinary least squares method without shuffling. We also consider a batch process to this problem where the datasets are independently available. The solution we propose is independent of initialization but requires that number of such datasets to be at least equal to the dimension of the unknown vector.**

***Index Terms*— Linear regression with shuffled labels, Shuffled linear regression, linear regression without correspondences, unlabelled sensing, stochastic approximation**

I. Introduction

A linear regression setting can simply be described as follows: Given a regressor $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ and an observed dependent variable $\overline{\boldsymbol{y}} \in \mathbb{R}^n$ such that

$$\overline{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\varepsilon}$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is unknown, and $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ is considered as an unknown disturbance term, which is usually random and may be due to erroneous observation. The problem is to estimate $\boldsymbol{w}$. Whenever $\boldsymbol{X}$ is full-column rank and elements of $\boldsymbol{X}$ and $\boldsymbol{\varepsilon}$ are uncorrelated, then one solution is the commonly used ordinary least squares, which is given $\widehat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\overline{\boldsymbol{y}}$.

In this paper we address the problem of shuffled linear regression where the elements of the observation $\overline{\boldsymbol{y}}$ is unknowingly shuffled, that is, the mutual ordering between $\boldsymbol{X}\boldsymbol{w}$ and $\overline{\boldsymbol{y}}$ is unknown. This problem is also known as linear regression with shuffled data, linear regression without correspondences, unlabeled sensing or permuted linear model [1]. The setting of the shuffled linear regression is presented as follows. Consider the following equation

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{X}\boldsymbol{w} + \boldsymbol{e} \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{w} \in \mathbb{R}^d$, $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, $\boldsymbol{M}$ is an unknown permutation matrix, and $\boldsymbol{e} \in \mathbb{R}^n$ is an additive error. There are many applications that directly relates to (1) such as in genome sequence assembly [2], sampling and reconstruction of spatial fields [3], multi-target tracking [4], and Internet-Of-Things networks [5]. In addition, this problem also relates to weakly-supervised machine learning [6]. However, this class of problem is an NP-hard problem [7], makes linear regression task considerably harder even in absence of observation errors. Nonetheless, using arguments from coding theory, it is shown that if $\boldsymbol{X}$ is iid Gaussian, it is possible to recover every $\boldsymbol{w}$ uniquely with probability 1 if and only if $n \geq 2d$ [8]. The work in [8] indicates that there is a possible recovery of $\boldsymbol{w}$ but do not deal with designing an estimation algorithm. One approach to solve this problem (with $\boldsymbol{e} = \boldsymbol{0}$) is by using brute force. That is, for each possible permutation $\boldsymbol{M}$ among the $n!$ permutations of the n entries of $\boldsymbol{y}$; check whether the linear system $\boldsymbol{M}^T\boldsymbol{y} = \boldsymbol{M}\boldsymbol{w}$ is consistent to solve, which is prohibitively complex algorithm in large dimensional problems, and of complexity $O(d^2(n+1)!)$ [1]. Other method is proposed with lesser complexity but may most likely fails in presence of any substantial observation errors [9]. A lower bound of the Signal-to-Noise Ratio (SNR) is also established in [9], below which any estimation would lead to a large estimation error. On the other hand, it is shown [10] that only if the SNR exceeds a certain threshold, then the estimation coincides with high probability with the Maximum Likelihood Estimator (MLE), $\left(\widehat{\boldsymbol{M}}_{ML}^T, \widehat{\boldsymbol{w}}_{ML}\right) = \underset{\boldsymbol{M}^T,\boldsymbol{w}}{\operatorname{argmin}} \|\boldsymbol{M}^T\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2$. The work in [6] tackles solving this MLE problem by an alternating minimization mechanism. Given an estimate $\widehat{\boldsymbol{w}}$, compute $\widehat{\boldsymbol{M}}^T$ by using the Metropolis-Hastings sampling technique on a

978-1-7281-1151-3/19/$31.00 ©2019 IEEE

Markov Chain defined over the set of permutation, then sort $\boldsymbol{y}$ accordingly, and estimate $\widehat{\boldsymbol{w}}$ using the ordinary least squares technique. This process is repeated several times. As mentioned in [1], the approach in [6] seems to be predominant but is very sensitive to initialization and generally works only for partially shuffled data. Another effective approach, e.g., used in [1] and [11], is based on an algebraic geometric approach, which uses symmetric polynomials to extract permutation-invariant constraints that $\boldsymbol{w}$ must satisfy. In particular, $\sum_{i=1}^{n}(\boldsymbol{x}_i\boldsymbol{w})^k = \sum_{j=1}^{n} y_j^k, k = 1,2,\dots,d$, where $\boldsymbol{x}_i$ is the i^{th} row of $\boldsymbol{X}$ and y_j is the j^{th} element of $\boldsymbol{y}$. This leads to a polynomial system of d equations with d unknowns. The proposed algorithmic solution of solving the polynomial set employs its most appropriate root as initialization to the Expectation-Maximization algorithm [1]. This approach yielded an efficient solution for small values of d. However, this polynomial-based approach may never be able to tackle large values of d, because even if the nominal solution, $\boldsymbol{w}$, is used with $\boldsymbol{e} = 0$, the computation of the polynomial, e.g., the term $\sum_{i=1}^{n}(\boldsymbol{x}_i\boldsymbol{w})^d - \sum_{j=1}^{n} y_j$, can diverge only due to numerical errors.

Existing methods are either applicable only to data with limited observation errors, work only for partially shuffled data, sensitive to initialization, and/or limited to smaller dimensions, e.g., partially shuffled data.

In this paper we propose an optimal recursive algorithm that updates the estimate of $\boldsymbol{w}$, $\widehat{\boldsymbol{w}}_k$, that attempts to find zeroes of the underdetermined function, $f(\boldsymbol{w}) = \sum_{i=1}^{n}(\boldsymbol{x}_i\boldsymbol{w})^d - \sum_{j=1}^{n} y_j$. $f(\boldsymbol{w})$ is associated with the first-order permutation-invariant constraint. This recursive algorithm is driven by

$$\hat{f}(\widehat{\boldsymbol{w}}_k) = \sum_{i=1}^{n} \boldsymbol{x}_i\widehat{\boldsymbol{w}}_k - \sum_{j=1}^{n} y_j, \tag{2}$$

and assumes zero-mean white observation errors with arbitrary covariance norm. The optimality is in the sense of minimizing the covariance of the estimation error, $\boldsymbol{w} - \widehat{\boldsymbol{w}}_k$. The approach is inspired by the work in [12]. We show that the variance of $f(\widehat{\boldsymbol{w}}_k) = \sum_{i=1}^{n} \boldsymbol{x}_i\widehat{\boldsymbol{w}}_k - \sum_{j=1}^{n}(\boldsymbol{y} - \boldsymbol{M}^T\boldsymbol{e})_j$ convergences to zero regardless the size of the covariance of $\boldsymbol{e}$. Since this approach solves an underdetermined equation, then it is sensitive to initialization errors. We propose an initial guess $\widehat{\boldsymbol{w}}_0$ that assumes small variations in elements of $\boldsymbol{w}$. We numerically compare the performance of our proposed algorithm with the ordinary least squares *without shuffling* while considering high dimensions of $\boldsymbol{w}$ large observation errors. Furthermore, we consider a variation of the problem, where the datasets $\boldsymbol{X}$ and $\boldsymbol{y}$ in (1) are available, not necessarily possessing the same length, n. We show that if d of such datasets are provided, then an estimate of $\boldsymbol{w}$ can be effectively evaluated without the need of initialization. In addition, we show that the influence of observation errors can be reduced using averaging.

The rest of the paper is organized as follows. The proposed recursive algorithm and its convergence are presented in Section II. We formulate the batch process and propose a solution in Section III. We provide illustrative examples in Section IV. The proposed initial guess and our examples are also included in Section IV. Finally, a conclusion is given in Section V.

Notations: Throughout this paper, non-bold lower-case letters are used to denote scalars, bold lower-case letters denote vectors, and upper-case bold letters denote matrices, unless defined otherwise. We denote by $\boldsymbol{I}$ the identity matrix, $\boldsymbol{0}$ the zero matrix, and $\overline{\mathbf{1}}_k = [1 \quad 1 \quad \cdots \quad 1] \in \mathbb{R}^k$. $\mathbb{E}[.]$ the expectation operator, and $tr(.)$ is the trace operator. $\lambda(\boldsymbol{M})$ is the eigenvalues of $\boldsymbol{M}$. $\boldsymbol{M} \succ \boldsymbol{0}$ and $\boldsymbol{M} \succcurlyeq \boldsymbol{0}$ denote the cases that $\boldsymbol{M}$ is positive and semi-positive definite matrix, respectively.

II. Proposed recursive algorithm

We consider the following:

$$f(\boldsymbol{w}) = \overline{\mathbf{1}}_n\big(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} - \boldsymbol{e}_p\big) \tag{3}$$

where $\boldsymbol{e}_p = \boldsymbol{M}^T\boldsymbol{e}$. The above equality is based on the facts that $\overline{\mathbf{1}}_n\boldsymbol{M} = \overline{\mathbf{1}}_n\boldsymbol{M}^T = \overline{\mathbf{1}}_n$ and $\boldsymbol{M}^T\boldsymbol{M} = \boldsymbol{I}$. The gradient, $\boldsymbol{g} \in \mathbb{R}^{1\times d}$, of $f(\boldsymbol{w})$, is $\boldsymbol{g} = -\overline{\mathbf{1}}_n\boldsymbol{X}$. We propose the following update algorithm

$$\widehat{\boldsymbol{w}}_{k+1} = \widehat{\boldsymbol{w}}_k - \boldsymbol{g}^T\kappa_k\hat{f}(\widehat{\boldsymbol{w}}_k) \tag{4}$$

where $\hat{f}(\widehat{\boldsymbol{w}}_k) = \overline{\mathbf{1}}_n(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{w}}_k)$, and the step size or gain $\kappa_k \in \mathbb{R}$. The recursive algorithm (4), which is inspired by the algorithm proposed in [12], yields

$$\hat{f}(\widehat{\boldsymbol{w}}_{k+1}) = \overline{\mathbf{1}}_n\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{w}}_k + \boldsymbol{X}\boldsymbol{g}^T\kappa_k\hat{f}(\widehat{\boldsymbol{w}}_k)\right)$$

Since $\overline{\mathbf{1}}_n\boldsymbol{X} = -\boldsymbol{g}$, then the above leads to $\hat{f}(\widehat{\boldsymbol{w}}_{k+1}) = \overline{\mathbf{1}}_n(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{w}}_k) - \boldsymbol{g}\boldsymbol{g}^T\kappa_k\hat{f}(\widehat{\boldsymbol{w}}_k)$ or

$$\hat{f}(\widehat{\boldsymbol{w}}_{k+1}) = (1 - \boldsymbol{g}\boldsymbol{g}^T\kappa_k)\hat{f}(\widehat{\boldsymbol{w}}_k) \tag{5}$$

Define $\boldsymbol{w}^*$ is such that $f(\boldsymbol{w}^*)|_{\boldsymbol{e}_p=0} = 0$ or by extracting the observation errors from $\boldsymbol{y}$, we have $\overline{\mathbf{1}}_n(\boldsymbol{M}^T\boldsymbol{y} - \boldsymbol{M}^T\boldsymbol{e}) = \overline{\mathbf{1}}_n\big(\boldsymbol{y} - \boldsymbol{e}_p\big) = \overline{\mathbf{1}}_n\boldsymbol{X}\boldsymbol{w}^*$ or $\overline{\mathbf{1}}_n\boldsymbol{y} = \overline{\mathbf{1}}_n\boldsymbol{X}\boldsymbol{w}^* + \overline{\mathbf{1}}_n\boldsymbol{e}_p$. It is important to reiterate that for $d > 1$, $\boldsymbol{w}^*$ is not unique. That is, there exist infinite $\boldsymbol{w}^*$ such that $f(\boldsymbol{w}^*)|_{\boldsymbol{e}_p=0} = 0$. Let the estimate error, $\boldsymbol{\delta}_k \triangleq \boldsymbol{w}^* - \widehat{\boldsymbol{w}}_k$. Using (4), we have $\boldsymbol{\delta}_{k+1} = \boldsymbol{w}^* - \widehat{\boldsymbol{w}}_k + \boldsymbol{g}^T\kappa_k\hat{f}(\widehat{\boldsymbol{w}}_k)$ or $\boldsymbol{\delta}_{k+1} = \boldsymbol{\delta}_k + \boldsymbol{g}^T\kappa_k\hat{f}(\widehat{\boldsymbol{w}}_k)$. Inserting $\overline{\mathbf{1}}_n\boldsymbol{y} = \overline{\mathbf{1}}_n\boldsymbol{X}\boldsymbol{w}^* + \overline{\mathbf{1}}_n\boldsymbol{e}_p$ into $\hat{f}(\widehat{\boldsymbol{w}}_k) = \overline{\mathbf{1}}_n(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{w}}_k)$, we get $\hat{f}(\widehat{\boldsymbol{w}}_k) = \overline{\mathbf{1}}_n\boldsymbol{X}\boldsymbol{\delta}_k + \overline{\mathbf{1}}_n\boldsymbol{e}_p$. Thus, $\boldsymbol{\delta}_{k+1} = \boldsymbol{\delta}_k + \boldsymbol{g}^T\kappa_k\overline{\mathbf{1}}_n\boldsymbol{X}\boldsymbol{\delta}_k + \boldsymbol{g}^T\kappa_k\overline{\mathbf{1}}_n\boldsymbol{e}_p$, and with $\boldsymbol{g} = -\overline{\mathbf{1}}_n\boldsymbol{X}$, we obtain

$$\boldsymbol{\delta}_{k+1} = (\boldsymbol{I} - \boldsymbol{g}^T\kappa_k\boldsymbol{g})\boldsymbol{\delta}_k + \boldsymbol{g}^T\kappa_k\overline{\mathbf{1}}_n\boldsymbol{e}_p \tag{6}$$

Since $\mathbb{E}[\boldsymbol{e}] = \mathbf{0} \Rightarrow \mathbb{E}[\boldsymbol{e}_p] = \mathbf{0}$ and (6) is a linear recursive equation, then $\mathbb{E}[\boldsymbol{\delta}_k] = \mathbf{0}$ provided that $\mathbb{E}[\boldsymbol{\delta}_0] = \mathbf{0}$. We define the covariance of $\boldsymbol{\delta}_k$ as $\boldsymbol{P}_k \triangleq \mathbb{E}[\boldsymbol{\delta}_k \boldsymbol{\delta}_k^T]$ and $\boldsymbol{\Phi}_k \triangleq \boldsymbol{I} - \boldsymbol{g}^T \kappa_k \boldsymbol{g}$. We assume:

(A1) $\overline{\mathbf{1}}_n \boldsymbol{X} \neq \mathbf{0}$

(A2) $\mathbb{E}[\boldsymbol{\delta}_0] = \mathbf{0},\ \boldsymbol{P}_0 \succ \mathbf{0}, R = \mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{e}_p \boldsymbol{e}_p^T \overline{\mathbf{1}}_n^T] > 0$ and $\mathbb{E}[\boldsymbol{\delta}_0 \boldsymbol{e}_p^T] = \mathbf{0}$.

Theorem 1. Let (3) satisfy Assumptions (A1)–(A2) and the algorithm (6) be applied. The gain κ_k that minimizes the mean-square of $\boldsymbol{\delta}_k$ at each k^{th} instant is given in the following recursive algorithm $\forall k \in \mathbb{N}$,

$$\boldsymbol{P}_{k+1} = \boldsymbol{\Phi}_k \boldsymbol{P}_k \boldsymbol{\Phi}_k^T + \boldsymbol{g}^T \kappa_k R \kappa_k \boldsymbol{g} \tag{7}$$

$$\kappa_k = (\boldsymbol{g}\boldsymbol{g}^T)^{-1} \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R)^{-1} \tag{8}$$

Proof of Theorem 1. Making use of (A2), (6) leads to

$$\boldsymbol{P}_{k+1} = \boldsymbol{\Phi}_k \boldsymbol{P}_k \boldsymbol{\Phi}_k^T + \boldsymbol{g}^T \kappa_k R \kappa_k \boldsymbol{g}$$

Expanding (7), we obtain

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_k - \boldsymbol{P}_k \boldsymbol{g}^T \kappa_k \boldsymbol{g} - \boldsymbol{g}^T \kappa_k \boldsymbol{g} \boldsymbol{P}_k + \boldsymbol{g}^T \kappa_k \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T \kappa_k \boldsymbol{g} + \boldsymbol{g}^T \kappa_k R \kappa_k \boldsymbol{g}$$

Collecting terms

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_k - \boldsymbol{P}_k \boldsymbol{g}^T \kappa_k \boldsymbol{g} - \boldsymbol{g}^T \kappa_k \boldsymbol{g} \boldsymbol{P}_k + \boldsymbol{g}^T \kappa_k (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R) \kappa_k \boldsymbol{g}$$

Since $\boldsymbol{P}_{k+1}$ is positive definite, then in order to minimize $\mathrm{tr}(\boldsymbol{P}_{k+1})$ with respect to κ_k, we set $\partial \mathrm{tr}(\boldsymbol{P}_{k+1}) / \partial \kappa_k \equiv \mathbf{0}$ at each iteration,

$$\frac{\partial \mathrm{tr}(\boldsymbol{P}_{k+1})}{\partial \kappa_k} = 2 \boldsymbol{g} \boldsymbol{g}^T \kappa_k (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R) - 2 \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T \equiv \mathbf{0}$$

Thus, $\kappa_k = (\boldsymbol{g}\boldsymbol{g}^T)^{-1} \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R)^{-1}$. □

Corollary 1. $\lim_{k \to \infty} \hat{f}(\widehat{\boldsymbol{w}}_k) = 0$ and $\lim_{k \to \infty} \kappa_k = 0$.

Proof. Inserting (8) in (5), we have $\hat{f}(\widehat{\boldsymbol{w}}_{k+1}) = \gamma_k \hat{f}(\widehat{\boldsymbol{w}}_k)$ where $\gamma_k \triangleq 1 - \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R)^{-1}$. Thus, $\hat{f}(\widehat{\boldsymbol{w}}_k) = \gamma_k^k \hat{f}(\widehat{\boldsymbol{w}}_0)$. Since $\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T > 0$ and $R > 0$, then $0 < \gamma_k < 1$, $\forall k$ and $\lim_{k \to \infty} \hat{f}(\widehat{\boldsymbol{w}}_k) = 0$. Equation (5) implies that $\lim_{k \to \infty} (1 - \boldsymbol{g}\boldsymbol{g}^T \kappa_k) = 0$ and since $\boldsymbol{g}\boldsymbol{g}^T \neq 0$, $\lim_{k \to \infty} \kappa_k = 0$. □

Remark 1. In applications, (A1) and (A2) may not be exactly satisfied. Thus, $\boldsymbol{P}_k$ (7) would not be optimal and could be larger than the actual error covariance matrix. To offset such an issue and subsequently not to drive κ_k to zero too fast, it is recommended to use instead of (7)

$$\boldsymbol{P}_{k+1} = \boldsymbol{\Phi}_k \boldsymbol{P}_k \boldsymbol{\Phi}_k^T + \boldsymbol{g}^T \kappa_k R \kappa_k \boldsymbol{g} + \alpha_P \boldsymbol{I} \tag{9}$$

where it is recommended to have the tuning parameter, α_P, satisfying $0 \leq \alpha_P \ll 1$. □

We next consider $f(\widehat{\boldsymbol{w}}_k) = \overline{\mathbf{1}}_n (\boldsymbol{y} - \boldsymbol{X} \widehat{\boldsymbol{w}}_k - \boldsymbol{e}_p) = \hat{f}(\widehat{\boldsymbol{w}}_k) - \overline{\mathbf{1}}_n \boldsymbol{e}_p$, and using (4), we have

$$f(\widehat{\boldsymbol{w}}_{k+1}) = \overline{\mathbf{1}}_n (\boldsymbol{y} - \boldsymbol{X} \widehat{\boldsymbol{w}}_k + \boldsymbol{X} \boldsymbol{g}^T \kappa_k \hat{f}(\widehat{\boldsymbol{w}}_k) - \boldsymbol{e}_p)$$

Inserting $\hat{f}(\widehat{\boldsymbol{w}}_k) = f(\widehat{\boldsymbol{w}}_k) + \overline{\mathbf{1}}_n \boldsymbol{e}_p$, we obtain

$$f(\widehat{\boldsymbol{w}}_{k+1}) = \overline{\mathbf{1}}_n (\boldsymbol{y} - \boldsymbol{X} \widehat{\boldsymbol{w}}_k + \boldsymbol{X} \boldsymbol{g}^T \kappa_k f(\widehat{\boldsymbol{w}}_k) + \boldsymbol{X} \boldsymbol{g}^T \kappa_k \overline{\mathbf{1}}_n \boldsymbol{e}_p - \boldsymbol{e}_p)$$

Using (3) and $\boldsymbol{g} = -\overline{\mathbf{1}}_n \boldsymbol{X}$, $f(\widehat{\boldsymbol{w}}_k) = \overline{\mathbf{1}}_n (\boldsymbol{y} - \boldsymbol{X} \widehat{\boldsymbol{w}}_k - \boldsymbol{e}_p)$, and collecting terms yield

$$f(\widehat{\boldsymbol{w}}_{k+1}) = (1 - \boldsymbol{g}\boldsymbol{g}^T \kappa_k) f(\widehat{\boldsymbol{w}}_k) - \boldsymbol{g}\boldsymbol{g}^T \kappa_k \overline{\mathbf{1}}_n \boldsymbol{e}_p \tag{10}$$

Corollary 2. If $\mathbb{E}[f(\widehat{\boldsymbol{w}}_0)] = 0$, then the covariance of $f(\widehat{\boldsymbol{w}}_k)$, $Q_k \triangleq \mathbb{E}[f^2(\widehat{\boldsymbol{w}}_k)]$, tends to zero as $k \to \infty$.

Proof of Corollary 2. Inserting (8) in (10), we have

$$f(\widehat{\boldsymbol{w}}_{k+1}) = \gamma_k f(\widehat{\boldsymbol{w}}_k) - (1 - \gamma_k) \overline{\mathbf{1}}_n \boldsymbol{e}_p$$

where $\gamma_k \triangleq 1 - \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T (\boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T + R)^{-1}$. We define $Q_k \triangleq \mathbb{E}[f(\widehat{\boldsymbol{w}}_k) f^T(\widehat{\boldsymbol{w}}_k)]$. Thus,

$$Q_{k+1} = \gamma_k^2 Q_k + (1 - \gamma_k)^2 R$$

Iterating Q_k, we get

$$Q_k = \bar{Q}_{0,k} + \bar{Q}_k \tag{11}$$

where $\bar{Q}_{0,k} = \left(\prod_{i=0}^{k-1} \gamma_{k-1-i}\right)^2 Q_0$ and $\bar{Q}_k = \sum_{i=0}^{k-1} \left(\prod_{j=1}^{k-1-i} \gamma_{k-j}\right)^2 (1 - \gamma_i)^2 R$, where $\prod_{i=k+1}^{k} \beta_i = 1$. We have $0 < \gamma_k < 1$, $\forall k$ and if $\lim_{k \to \infty} \gamma_k = 1$, then $\lim_{k \to \infty} \boldsymbol{g} \boldsymbol{P}_k \boldsymbol{g}^T = 0$ and since $\boldsymbol{P}_k \succ 0$, this implies $\lim_{k \to \infty} \boldsymbol{P}_k = \mathbf{0}$, which means the ultimate problem is solved. Therefore, we assume that $\lim_{k \to \infty} \gamma_k < 1$. Therefore, $\lim_{k \to \infty} \bar{Q}_{0,k} = 0$. Next, we show that $\lim_{k \to \infty} \bar{Q}_k = 0$ by showing that every term in the series converges to zero.

Denote $\Omega_{i,k} \triangleq \left(\prod_{j=1}^{k-1-i} \gamma_{k-j}\right)^2 (1 - \gamma_i)^2 R$. Consider $\lim_{k \to \infty} \sum_{i=0}^{k-1} \Omega_{i,k} = \sum_{i=0}^{\infty} \lim_{k \to \infty} \Omega_{i,k} = \sum_{i=0}^{l<\infty} \lim_{k \to \infty} \Omega_{i,k} +$ {the infinitely many terms in the series corresponding to $l \to \infty$}. The infinitely many terms are zero since $\lim_{k \to \infty} \Omega_{i,k} = 0$, and for the same reason we have $\sum_{i=0}^{l<\infty} \lim_{k \to \infty} \Omega_{i,k} = 0$. □

Remark 2. Corollary 2 assumes that $\mathbb{E}[f(\widehat{\boldsymbol{w}}_0)] = 0$ or $\mathbb{E}[f(\widehat{\boldsymbol{w}}_0)] = \left(\mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{y}] - \mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{X} \widehat{\boldsymbol{w}}_0] - \mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{e}_p]\right) = 0$. Since $\mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{e}_p] = 0$, then $\mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{y}] = \mathbb{E}[\overline{\mathbf{1}}_n \boldsymbol{X} \widehat{\boldsymbol{w}}_0]$. One direction for selecting $\widehat{\boldsymbol{w}}_0$ is proposed in example of the subsequent section. □

Many techniques used in the proofs are inspired by the work done in [13], [14] and [15].

III. A batch process consideration

The entire dataset in an application may involve having different independent subdatasets satisfying (1) as follows

$$\boldsymbol{y}_l = \boldsymbol{M}_l \boldsymbol{X}_l \boldsymbol{w} + \boldsymbol{e}_l \tag{12}$$

where $\boldsymbol{y}_l \in \mathbb{R}^{n_l}$, $\boldsymbol{w} \in \mathbb{R}^d, \boldsymbol{X}_l \in \mathbb{R}^{n_l \times d}$, $\boldsymbol{M}_l$ is an unknown permutation matrix, and $\boldsymbol{e}_l \in \mathbb{R}^{n_l}$ is an additive error, where $l \geq d$ and $n_l \geq 1$.

Since $\overline{\mathbf{1}}_{n_l} \boldsymbol{y}_l = \overline{\mathbf{1}}_{n_l} \boldsymbol{X}_l \boldsymbol{w} + \overline{\mathbf{1}}_{n_l} \boldsymbol{e}_l$, then the latter can be written as $S_l^{\boldsymbol{y}} = \begin{bmatrix} S_l^{\boldsymbol{x}_1} & S_l^{\boldsymbol{x}_2} & \cdots & S_l^{\boldsymbol{x}_d} \end{bmatrix} \boldsymbol{w} + S_l^{\boldsymbol{e}}$, where $S_l^z = \sum_{i=1}^{n_l} z_i$, and, $\boldsymbol{x}_j$, in $S_l^{\boldsymbol{x}_j}$ is the j^{th} column of $\boldsymbol{X}_l$.

$$\boldsymbol{\xi} = \boldsymbol{\Xi} \boldsymbol{w} + \boldsymbol{\epsilon} \tag{13}$$

where the i^{th} element of $\boldsymbol{\xi} \in \mathbb{R}^l$ and $\boldsymbol{e} \in \mathbb{R}^l$ is $S_i^{\boldsymbol{y}}$ and $S_i^{\boldsymbol{e}}$, respectively, and j^{th} row of $\boldsymbol{\Xi} \in \mathbb{R}^{l \times d}$ is $\begin{bmatrix} S_l^{\boldsymbol{x}_1} & S_l^{\boldsymbol{x}_2} & \cdots & S_l^{\boldsymbol{x}_d} \end{bmatrix}$. If $\boldsymbol{\Xi}$ is a full column-rank, then by using ordinary least squares, we can have an estimate independent of initialization

$$\widehat{\boldsymbol{w}} = (\boldsymbol{\Xi}^T \boldsymbol{\Xi})^{-1} \boldsymbol{\Xi}^T \boldsymbol{\xi} \tag{14}$$

Of note, If the elements of $\boldsymbol{X}_l$ are real and iid, then $S_j^{\boldsymbol{x}_1}$ is real and iid and (since $l \geq d$) $\boldsymbol{\Xi}$ is full-column rank with probability 1. In case $\boldsymbol{e} \cong \mathbf{0}$, then the least of amount of data required to uniquely recover $\boldsymbol{w}$ is whenever $l = d$ and $n_l = 1$, where in this case $\widehat{\boldsymbol{w}} = \boldsymbol{\Xi}^{-1} \boldsymbol{\xi}$. If $n_l = 1$, then the solution becomes the one of ordinary least squares without shuffling. However, whenever $\boldsymbol{e} \ncong \mathbf{0}$ and n_l is not large enough and in order to make the estimate (14) less sensitive to observation errors, we need $l \gtrsim 2d$. The accuracy of $\widehat{\boldsymbol{w}}$ depends on the variance of $\boldsymbol{e}_l$. In what follows, we propose a method that can reduce the effect of $\boldsymbol{e}_l$ on $\widehat{\boldsymbol{w}}$ with $l \geq 2d$. For each d independent set of observations with any length $n_l \geq 1$, we have (12) a k^{th} estimate

$$\widehat{\boldsymbol{w}}_k = (\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\xi}_k \tag{15}$$

We consider the nominal value to be the one associated with (13) using any of the k^{th} estimates while compensating for observation errors, that is

$$\boldsymbol{w} = (\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\xi}_k - \boldsymbol{\epsilon}_k^{\boldsymbol{\Xi}} \tag{16}$$

where $\boldsymbol{\epsilon}_k^{\boldsymbol{\Xi}} = (\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\epsilon}_k$. Since $\boldsymbol{X}$ and $\boldsymbol{e}$ are uncorrelated, then $\boldsymbol{\Xi}_k$ or $(\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T$ and $\boldsymbol{\epsilon}_k$ are as well. Thus, $\mathbb{E}[(\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\epsilon}_k] = \mathbb{E}[(\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T] \mathbb{E}[\boldsymbol{\epsilon}_k]$ and since $\mathbb{E}[\boldsymbol{e}_k] = 0$, then $\mathbb{E}[\boldsymbol{\epsilon}_k] = 0$ and $\mathbb{E}[\boldsymbol{\epsilon}_k^{\boldsymbol{\Xi}}] = 0$. In addition, since $\boldsymbol{w}$ is deterministic, then $\boldsymbol{w} = \mathbb{E}[\boldsymbol{w}] = \mathbb{E}[(\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\xi}_k] - \mathbb{E}[\boldsymbol{\epsilon}_k^{\boldsymbol{\Xi}}]$ or $\boldsymbol{w} = \mathbb{E}[(\boldsymbol{\Xi}_k^T \boldsymbol{\Xi}_k)^{-1} \boldsymbol{\Xi}_k^T \boldsymbol{\xi}_k]$. Consequently, $\boldsymbol{w} = \mathbb{E}[\widehat{\boldsymbol{w}}_k]$. Without loss of generality, we assume $l = md$, where the integer $m \gg 2$, then

$$\boldsymbol{w} \approx \frac{1}{m} \sum_{i=1}^{m} \widehat{\boldsymbol{w}}_k \tag{17}$$

IV. Numerical simulations

In this section, we provide two examples. The first example illustrates the performance of the proposed algorithm in (4), (7) and (8). Based on the datasets under consideration, we provide for $\widehat{\boldsymbol{w}}_0$, The second example illustrates the proposed approach while considering a batch process presented in Section III.

Example 1

The choice of initial guess is thought to be rather critical for any practical estimation of such underdetermined system of equations (1). However, in order to show the capability of the proposed algorithm in (4), (7) and (8) rejecting observation errors, in our example we set all the elements of the actual $\boldsymbol{w}$ to 1 in the first 3 scenarios and consider fluctuations in the elements of $\boldsymbol{w}$ in the fourth. We assume that the error, $\boldsymbol{e}$, is uniformly distributed, whose entries are drawn from $\mathcal{U}(0, \sigma^2)$; here the mean and variance of the uniformly distributed random variable are equal to 0 and σ^2, respectively. In this example we draw the elements of $\boldsymbol{X}$ from $\mathcal{U}(0,1)$. The permutation matrix, $\boldsymbol{M}$, is chosen at random from the set of all $n \times n$ permutation matrices.

Our initial guess, $\widehat{\boldsymbol{w}}_{\boldsymbol{0}}$, is based on the following: Equation (1) implies that $\sum_{i=1}^{n} (\boldsymbol{y})_i = \sum_{i=1}^{n} (\boldsymbol{M}\boldsymbol{X}\boldsymbol{w})_i + \sum_{i=1}^{n} (\boldsymbol{e})_i$ where the operator $(.)_i$ is the i^{th} element of its argument vector. Since $\boldsymbol{M}$ is a permutation matrix, then $\sum_{i=1}^{n} (\boldsymbol{M}\boldsymbol{X}\boldsymbol{w})_i = \sum_{i=1}^{n} (\boldsymbol{X}\boldsymbol{w})_i$. Thus, $\frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{y})_i = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{X}\boldsymbol{w})_i + \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{e})_i$. Since $\mathbb{E}[\boldsymbol{e}] = 0 \Rightarrow \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{e})_i = 0$, then for $n \gg 1$, we have $\frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{e})_i \cong 0$ and $\frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{y})_i \cong \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{X}\boldsymbol{w})_i$. Consequently, we set our initial guess

$$\widehat{\boldsymbol{w}}_0 \equiv \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{X}^{\#})_i \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{y})_i \, \overline{\mathbf{1}}_d^T \tag{18}$$

where $\boldsymbol{X}^{\#} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T$. If $\widehat{\boldsymbol{w}}_0 \equiv \frac{1}{d}\sum_{i=1}^{d} (\boldsymbol{X}^{\#}\boldsymbol{y})_i \, \overline{\mathbf{1}}_d^T$ is used instead of (18), then similar results are obtained. This initial guess can fail if the fluctuations of the elements in the columns of $\boldsymbol{X}$ or $\boldsymbol{w}$ around their non-zero means is large. However, it is adopted to illustrate the robustness of the proposed algorithm to high levels of observation errors and high dimensionality of $\boldsymbol{w}$.

We set the parameters used in the recursive algorithm (8) and (9) as follows: $\boldsymbol{P}_0 = \boldsymbol{I}_d$, $\alpha_P = 0.1$, and $R = 2n\sigma^2$. The adequate value for $R = n\sigma^2$; however, an erroneous model is used to show the robustness of the algorithm. It turns out that both values result almost in the same convergence rate, which is about 12 iterations to drive $\left|\hat{f}(\widehat{\boldsymbol{w}}_k)\right| < 10^{-6}$. We compare the estimation of our proposed algorithm with those weights estimated by ordinary least squares before shuffling the labels, that is, $\widehat{\boldsymbol{w}}_0^{LS} = \boldsymbol{X}^{\#} \boldsymbol{M}^T \boldsymbol{y}$. For each point in the

experiments we conduct 1,000 different run and take the average of the obtained relative errors, $\frac{\|\boldsymbol{w}-\widehat{\boldsymbol{w}}\|_2}{\|\boldsymbol{w}\|_2}$.

Scenario 1.

We hold $d = 50$ and $n = 2d$ while varying the standard deviation of the observation error, σ, between 0 and 5. The relative errors are shown in Fig. 1. We find that the relative error increases from 6.0×10^{-4} to 4.7×10^{-3} using the proposed algorithm; and increases from 4.6×10^{-14} to 6.9×10^{-1} using least squares without shuffling. This shows the ability of the proposed algorithm rejecting erroneous observations with significantly large values.

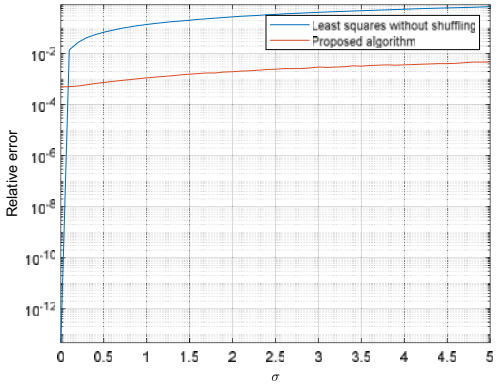

Fig. 1. Example 1, *Scenario* 1: $d = 50$ and $n = 2d$.

Scenario 2.

We hold $\sigma = 1$ and vary d from 2 to 100 with $n = 2d$. The corresponding results are depicted in Fig. 2. The relative error decreases from 1.5×10^{-1} to 4.4×10^{-4} using the proposed algorithm; and decreases from 5×10^{-1} to 1×10^{-1} using least squares without shuffling. A significantly better improvement in performance is obtained using the proposed algorithm when increasing the dimension of the weight.

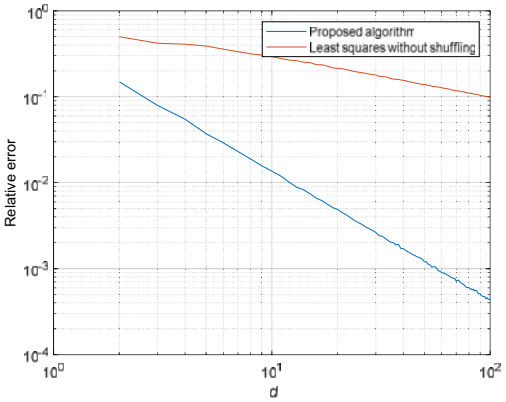

Fig. 2. Example 1, *Scenario* 2: $\sigma = 1$ and $n = 2d$.

Scenario 3.

We hold $d = 50$ and $\sigma = 1$ while varying n while 100 to 1,000. The results of the relative errors are shown in Fig. 3. The relative error decreases from 1.2×10^{-3} to 3.2×10^{-4} using the proposed algorithm; and decreases from 1.4×10^{-1} to 3.2×10^{-2} using least squares without shuffling, which is almost the same rate of improvement for both schemes when n is increased.

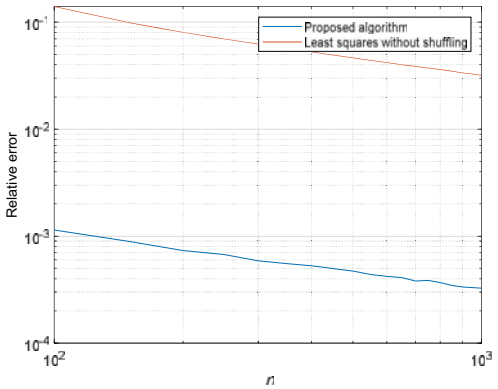

Fig. 3. Example 1, *Scenario* 3: $d = 50$ and $\sigma = 1$.

Scenario 4.

We set the elements of $\boldsymbol{w}$, $w_i = 1 + \delta w_i$, where δw_i is drawn from $\mathcal{U}(0, \delta\omega^2)$. That means that $\max(w_i) - \min(w_i) = \sqrt{12}\delta\omega$ or $-\sqrt{3}\,\delta\omega \le \delta w_i \le \sqrt{3}\,\delta\omega$. We set $n = 2d$. We also use $\widehat{\boldsymbol{w}}_{\boldsymbol{0}}$ given in (18). We run for different combinations of $d \in \{10,50\}$ and $\sigma \in \{1,5\}$. We also compare performance with the ordinary least squares without shuffling. The corresponding results are depicted in Fig. 4. Unlike least squares results, the proposed algorithm shows very much unaffected by the size of observation errors and the dimension of $\boldsymbol{w}$. However, the performance of the proposed algorithm is sensitive to initialization and the initial guess (18) is sensitive to the fluctuations in the elements of $\boldsymbol{w}$.

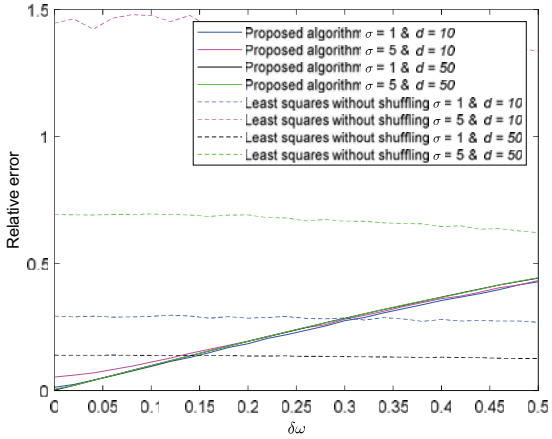

Fig. 4. Example 1, *Scenario* 4: $n = 2d$, $w_i = 1 + \delta w_i$, where δw_i is drawn from $\mathcal{U}(0, \delta\omega^2)$.

Example 2

In this example, we illustrate the performance of the proposed batch process approach proposed in Section

III. We draw the elements of $\boldsymbol{X}_l$ from $\mathcal{U}(0,1)$. The permutation matrix, $\boldsymbol{M}_l$, is chosen at random from the set of all $n_l \times n_l$ permutation matrices. We set the elements of $\boldsymbol{w}$, $w_i = 1 + \delta w_i$, where δw_i is drawn from $\mathcal{U}(0, \delta\omega^2)$. We hold $d = 50$ and $\sigma = 0.1$ for $\delta\omega \in \{0,0.5,1,5\}$, and the estimate (15), we use $n_l = d$ and $l = 2d$. We consider $k \in \{1,2,\dots,10\}$ in (17). For each point in the experiments we conduct 1,000 different run and take the average of the corresponding relative error. The results are depicted in Fig. 5. As expected, the relative error decreases as k increases or $\delta\omega$ increases.

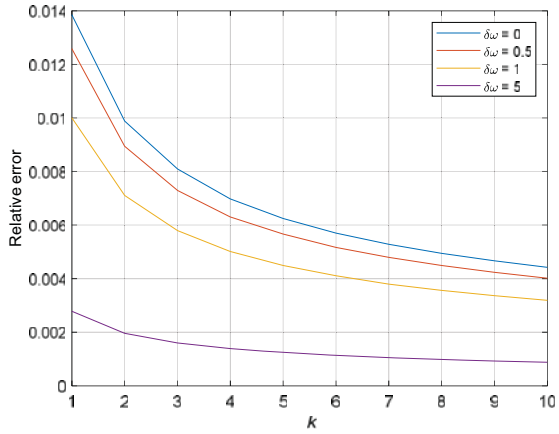

Fig. 5. Example 2. $d = 50, \sigma = 0.1, n_l = d$, and $l = 2d$.

V. Conclusion

This paper tackled shuffled linear regression problem using stochastic approximation. The proposed recursive algorithm aimed for per-iteration minimization of the mean square estimate error. Although our algorithm turned out to be sensitive to initialization errors, the algorithm is considered as the first working solution for arbitrary large dimensions and arbitrary large observation errors. Numerical simulations have shown that our method with shuffled datasets can outperform the proposed approach in [11] and the ordinary least squares method without shuffling in presence of substantial observation errors. Further work could address its sensitivity to initialization while comparing performance with the existing art work such as the ones in [1] and [11], which can broaden its domain to different applications. For example, based on the different classes of datasets, one may be able to devise different initialization. In addition to the mentioned work, we considered a problem where at least d different independent datasets are available. The proposed initialization-independent solution has been shown to be simple, effective, accurate, and can easily deal with high dimensions.

References

[1] M. C. Tsakiris, L. Peng, A. Conca, L. Kneip, Y. Shi, and H. Choi, "An Algebraic-Geometric Approach to Shuffled Linear Regression," *arXiv preprint arXiv:1810.05440*. 2018 Oct 12.

[2] X. Huang and A. Madan, "Cap3: A dna sequence assembly program," *Genome Research*, vol. 9, no. 9, pp. 868–877, 09 1999.

[3] J. Unnikrishnan and M. Vetterli, "Sampling and reconstruction of spatial fields using mobile sensors," *IEEE Transactions on Signal Processing*, vol. 61, no. 9, pp. 2328–2340, May 2013.

[4] A. B. Poore and S. Gadaleta, "Some assignment problems arising from multiple target tracking," *Mathematical and Computer Modelling*, vol. 43, no. 9, pp. 1074 – 1091, 2006.

[5] X. Song, H. Choi, and Y. Shi, "Permuted linear model for header-free communication via symmetric polynomials," *in International Symposium on Information Theory* (ISIT), 2018.

[6] A. Abid and J. Zou, "Stochastic em for shuffled linear regression," *arXiv preprint arXiv:1804.00681v1*, 2018.

[7] A. Pananjady, M. J. Wainwright, and T. A. Courtade, "Linear regression with shuffled data: Statistical and computational limits of permutation recovery," in *IEEE Transactions on Information Theory*, vol. 64, no. 5, pp. 3286–3300, 2018.

[8] J. Unnikrishnan, S. Haghighatshoar, and M. Vetterli, "Unlabeled sensing with random linear measurements," *IEEE Transactions on Information Theory*, vol. 64, no. 5, pp. 3237–3253, 2018.

[9] D. J. Hsu, K. Shi, and X. Sun, "Linear regression without correspondence," *in Advances in Neural Information Processing Systems* (NIPS), pp. 1531–1540, 2017.

[10] A. Pananjady, M. J. Wainwright, and T. A. Courtade, "Linear regression with shuffled data: Statistical and computational limits of permutation recovery," *IEEE Transactions on Information Theory*, vol. 64, no. 5, pp. 3286–3300, 2018.

[11] A. Abid, A. Poon, and J. Zou, "Linear regression with shuffled labels," *arXiv preprint arXiv:1705.01342v2*, 2017.

[12] K. K. Saab and S. S. Saab, Jr, "A stochastic Newton's method with noisy function measurements", *IEEE Signal Process Letters*, vol. 23, no. 3, pp. 361-365, 2016.

[13] S. S. Saab and P. Ghanem, "A Multivariable Stochastic Tracking Controller for Robot Manipulators without Joint Velocities," *IEEE Transactions on Automatic Control*, vol. 63, no. 8, pp. 2481-2495, Aug. 2018.

[14] S. S. Saab, "An optimal stochastic multivariable PID controller: a direct output tracking approach," *International Journal of Control*. Accepted author version posted online: 07 Aug 2017.

[15] S. S. Saab, "Development of multivariable PID controller gains in presence of measurement noise," *International Journal of Control*, vol. 90, no. 12, pp., 2692-2710, 2017.