

Iterative Supervised Learning for Regression with Constraints

Tejaswi K. C. and Taeyoung Lee

Abstract—Regression in supervised learning often requires the enforcement of constraints to ensure that the trained models are consistent with the underlying structures of the input and output data. This paper presents an iterative procedure to perform regression under arbitrary constraints. It is achieved by alternating between a learning step and a constraint enforcement step, to which an affine extension function is incorporated. We show this leads to a contraction mapping under mild assumptions, from which the convergence is guaranteed analytically. The presented proof of convergence in regression with constraints is the unique contribution of this paper. Furthermore, numerical experiments illustrate improvements in the trained model in terms of the quality of regression, the satisfaction of constraints, and also the stability in training, when compared to other existing algorithms.

I. INTRODUCTION

Enforcing constraints on supervised learning is critical when the underlying structures of the data should be respected in the trained model, or when it is required to overcome a bias in the data set. For instance, in [1], constraints caused by length, angle, or collision are studied with projection when predicting the motion of a physical system with neural networks. In [2], fairness with respect to protected features, such as race or gender, is addressed in socially sensitive decision making. Further, it has been illustrated by [3] that the performance of deep learning can be improved by integrating the domain knowledge in the form of constraints. As such, imposing constraints is desirable in injecting our prior knowledge of the model, which is encoded indirectly in the data, to supervised learning explicitly.

One of the common techniques to implement constraints is augmenting the loss function with an additional penalty on the violation of the constraints, as presented by [4] and [5]. On the other hand, constraints have also been implemented directly as hard constraints that should be satisfied strictly. Imposing hard constraints on deep neural network is presented by [6] after customizing large-scale optimization techniques. Alternatively, [7] handles output label restrictions through a Lagrangian based formulation. Both of these approaches based on additional regularization terms or hard constrained optimization involve the process of actively adjusting model parameters in training. In other words, the possibly conflicting goals of regression and constraint enforcement should be addressed simultaneously. This may hinder the efficiency of the training procedure, while making it susceptible to various numerical issues.

Recently, an iterative procedure has been proposed by [8], where the constraints are enforced by adjusting the

target, instead of manipulating the model parameters directly, thereby addressing the aforementioned challenges. The desirable feature is that any supervised learning technique developed without constraint consideration can be adopted in conjunction with nonlinear constrained optimization tools. However, this approach is heuristic in the sense that there is no analytical assurance for convergence through iterations, while its performance is illustrated with several numerical examples. In fact, it is challenging to present a convergence property in any supervised learning with constraints.

The main objective of this paper is to establish a certain convergence guarantee in regression with constraints. Towards this end, we utilize the procedure of adjusting the target to satisfy constraints. More specifically, the ideal target is projected to the intersection between the set of possible outputs from the chosen model and the set of feasible outputs. Then the model parameters are optimized to the adjusted target, and these two steps are repeated. The proposed approach is motivated by the alternating projections [9], [10] and Dykstra's algorithm [11]. In particular, the two steps of iterations in adjusting the target, and in training the model are considered as certain projection operators, from which convergence is established by the Banach fixed point theorem [12].

The desirable feature is that we have a certain assurance of convergence in regression with constraints. Another interesting feature is its general formulation: as discussed above, this framework can be integrated with any supervised learning technique. And, it further addresses the challenges of adjusting the model parameters to the satisfaction of the constraints, while performing regression simultaneously. One downside is that we cannot enforce the constraint strictly as hard constraints, but there is a design parameter that provides a trade-off between regression and constraint satisfaction.

Numerical experiments demonstrate that the proposed approach improves the regression performance in the similar level of constraint violation. More importantly, it exhibits more consistent results over five-fold validations. As such, the proposed convergence proof is actually beneficial in numerical implementations.

II. ITERATIVE LEARNING WITH CONSTRAINTS

A. Problem Formulation

Consider a regression problem where we should predict the ideal output $y \in \mathbb{R}^n$ given the inputs $X \in \mathbb{R}^{n \times d}$. Here n denotes the number of points in the dataset, and d corresponds to the number of features in each data point. Let the model for supervised learning be denoted by $\hat{y} = f(X, \theta)$, where $\theta \in \mathbb{R}^p$ are the model parameters and $\hat{y} \in \mathbb{R}^n$ is the output predicted by the current model parameters. The goal of regression is to identify the optimal model parameters θ^* that minimize a given

Mechanical and Aerospace Engineering, The George Washington University, Washington DC 20052 ktejaswi999@gmail.com, tylee@gwu.edu
The research was partially supported by NSF under the grants CNS-1837382, CMMI-1760928 and by AFOSR FA9550-18-1-0288.

loss function, $L(y, \hat{y})$, $L : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$. In addition, we enforce constraints on the predicted output so that it belongs to a feasible set denoted by $C \subset \mathbb{R}^n$, i.e., $\hat{y} \in C$. Thus, the optimization problem for regression with constraints can be formulated as

$$\theta^* = \arg \min_{\theta} \{L(y, \hat{y}) \mid \hat{y} = f(X, \theta), \text{ and } \hat{y} \in C\}.$$

Alternatively, this can be reorganized into an optimization on the output space as

$$z = \arg \min_{\hat{y}} \{L(\hat{y}, y) \mid \hat{y} \in B \cap C\}, \quad (1)$$

$$\theta^* = \arg \min_{\theta} \{L(z, \hat{y}) \mid \hat{y} = f(X, \theta), \theta \in \mathbb{R}^p\}. \quad (2)$$

where $B = \{\hat{y} \mid \hat{y} = f(X, \theta), \theta \in \mathbb{R}^p\}$ is the set of all possible outputs under the current model. In other words, (1) is to find an alternative optimal target $z \in \mathbb{R}^n$ that is closest to the ideal target y under the restriction of the given constraint and the model bias. Next, in (2), the model parameters are optimized such that the predicted output matches to the optimal target z , not the ideal target y . The intriguing feature is that the supervised learning in (2) corresponds to the usual supervised learning without constraints, as the constraints are enforced indirectly through (1). As such, any supervised learning scheme can be utilized for (2). For (1), standard tools in nonlinear constrained optimization can be applied.

B. Iterative Learning Algorithm with Constraints

In [8], this problem is tackled by a clever combination of two iterations, which is verified by various numerical examples. But it might be heuristic in the sense that no convergence property is established. Here we propose the following alternative iterative scheme for (1) and (2), summarized by Algorithm 1, which provides a certain convergence property in regression. Here, α, β are non-negative parameters in the adjustment step, and N_i is the total number of iterations of this procedure.

Algorithm 1 Regression with constraints

Input: $y \in \mathbb{R}^n, \{\alpha, \beta\} \in \mathbb{R}, N_i \in \mathbb{Z}$

1. $\hat{y}^1 = \arg \min_{\hat{y}} \{L(\hat{y}, y) \mid \hat{y} \in B\}$ # Initial training
2. **for** $i = 1$ **to** $N_i - 1$ **do**
3. **if** $\hat{y}^i \notin C$ **then**
4. $z^i = \arg \min_z \{L(z, (1 - \alpha)y + \alpha\hat{y}^i) \mid z \in C\}$
Infeasible adjustment
5. **else**
6. $z^i = \arg \min_z \{L(z, y) \mid L(z, \hat{y}^i) \leq \beta, z \in C\}$
Feasible adjustment
7. **end if**
8. $\hat{y}^{i+1} = \arg \min_{\hat{y}} \{L(\hat{y}, z^i) \mid \hat{y} \in B\}$
Unconstrained training
9. **end for**

Output: \hat{y}^{N_i}

In the first step of initial training, supervised learning is performed without considering the constraint. The next iterations are composed of two parts of target adjustment and unconstrained training, and the target adjustment step has two sub-cases depending on the output of the previous step. In particular, the most critical step is when the output of the trained model does not satisfy the constraint. In the step 4, denoted by *infeasible adjustment*, the target is adjusted to minimize $L(z, (1 - \alpha)y + \alpha\hat{y}^i)$. That is, we find a feasible target $z \in C$ that is closest to a point on the line connecting y and \hat{y} in terms of the loss function.

Next, when the output of the trained model satisfies the constraint, in the step of *feasible adjustment*, the target z is moved closer to the original target y within a ball of radius β measured in terms of the loss. Finally, the model is trained with the adjusted target, and the whole procedure is repeated.

C. Convergence Property

Now we present a convergence property of Algorithm 1, which has motivated the proposed form of the objective function in the infeasible adjustment step.

Denote norms on the Euclidean space by $\|\cdot\| : \mathbb{R}^n \rightarrow \mathbb{R}$, with the L_2 and the L_1 norms represented by $\|\cdot\|_2$, and $\|\cdot\|_1$, respectively. A projection operator $P_{Z,L} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ on the set Z with respect to the loss L is defined as

$$P_{Z,L}(x) = \arg \min_z \{L(z, x) \mid z \in Z\}. \quad (3)$$

In other words, $x \in \mathbb{R}^n$ is projected to $z \in Z$ such that the distance between x, z is minimized in terms of the loss L .

Consider a convex subset $Z \subseteq X$ of a finite-dimensional normed vector space $(X, \|\cdot\|)$. There exists a unique projection $P_{Z,\|\cdot\|}(x) \in Z$ for each $x \in X$ such that

$$\|x - P_{Z,\|\cdot\|}(x)\| = \inf \{\|x - z\| \mid z \in Z\}$$

if the underlying geometric constraint is satisfied (see [13, Proposition 3.2]). That is, $P_{Z,\|\cdot\|}$ should not be contained in some non-degenerate line segment of ∂Z which is parallel to some non-degenerate line segment in the boundary of the unit $\|\cdot\|$ ball.

Assumption 1. We make the following assumptions.

- The sets B and C are convex.
- The projection operator in B and C is Lipschitz, i.e., there exists a norm $\|\cdot\|$ and $K > 0$ such that $\|P_{A,L}(x) - P_{A,L}(y)\| \leq K \|x - y\|$ for all $x, y \in \mathbb{R}^n$ where $A = B$ or $A = C$.

When the loss function in the projection (3) is MSE, the above two statements are actually equivalent [13].

Now we are concerned with convergence of the sequence, $(\hat{y}^i) \in B$ generated after the training step. In other words, we wish to show that $\hat{y}^i \rightarrow \bar{y}$ as $i \rightarrow \infty$ for some $\bar{y} \in \mathbb{R}^n$. The convergence of Algorithm 1 is established as follows.

Theorem 1. Suppose $\alpha < 1/K^2$, where $K \geq 1$ is the Lipschitz constant introduced in Assumption 1. The iterations of Algorithm 1 has a unique fixed point in B , which is the

limit of the sequence (\hat{y}^i) for an initial $\hat{y}^1 \in B$, when β is sufficiently small.

Proof. When $\beta \rightarrow 0$, Algorithm 1 iterates between the infeasible adjustment step and unconstrained training, and it can be written as

- Affine extension: $y^\alpha = (1 - \alpha)y + \alpha\hat{y}^i$
- Adjustment: $z^i = P_{C,L}(y^\alpha) = \arg \min_z \{L(z, (1 - \alpha)y + \alpha\hat{y}^i) \mid z \in C\}$
- Learning: $\hat{y}^{i+1} = P_{B,L}(z^i) = \arg \min_{\hat{y}} \{L(\hat{y}, z^i) \mid \hat{y} \in B\}$

Therefore, Algorithm 1 corresponds to a concatenation of two projections as

$$\hat{y}^{i+1} = P_{B,L}(P_{C,L}(h(\hat{y}^i))), \quad (4)$$

where $h : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is the affine extension function defined as $h(\hat{y}^i) = (1 - \alpha)y + \alpha\hat{y}^i$.

Consider any two points $\hat{y}^1, \hat{y}^2 \in B$. We have

$$\begin{aligned} & \|P_{B,L}(P_{C,L}(h(\hat{y}^1))) - P_{B,L}(P_{C,L}(h(\hat{y}^2)))\| \\ & \leq K \|P_{C,L}(h(\hat{y}^1)) - P_{C,L}(h(\hat{y}^2))\| \\ & \leq K^2 \|h(\hat{y}^1) - h(\hat{y}^2)\| \\ & \leq K^2 \alpha \|\hat{y}^1 - \hat{y}^2\| \end{aligned}$$

If $K^2\alpha < 1$, then each iteration is a contraction mapping on B with the metric induced by this norm, $d(x, y) = \|y - x\|$. Since (B, d) is a complete metric space, the series of iterations has a unique fixed point $\bar{y} = f(\bar{y})$ according to the Banach fixed point theorem [12]. Moreover, the sequence $\{\hat{y}^1, \hat{y}^2, \dots\}$ converges to \bar{y} for any $\hat{y}^1 \in B$. \square

In short, the Lipschitz properties of Assumption 1 ensures that each iteration is a contraction. The critical question is how we can ensure the Lipschitz property of the projection operators.

Corollary 1. *The convergence of Algorithm 1 is guaranteed as described in Theorem 1 for the following loss functions.*

- For the mean squared error (MSE) given by $L(z, y) = \frac{1}{n} \sum_{k=1}^n (z_k - y_k)^2$, the algorithm converges for the parameter $\alpha \in [0, 1)$.
- For the mean absolute error (MAE) given by $L(z, y) = \frac{1}{n} \sum_{k=1}^n |z_k - y_k|$, the algorithm converges for the parameter $\alpha \in [0, 0.25)$.

Proof. For MSE, the projection in (3) corresponds to

$$\begin{aligned} P_{Z,L}(y) &= \arg \min_z \left\{ \frac{1}{n} \sum_{k=1}^n (z_k - y_k)^2 \mid z \in Z \right\} \\ &= \arg \min_z \left\{ \|z - y\|_2^2 \mid z \in Z \right\}, \end{aligned}$$

which is equal to minimization with respect to the standard Euclidean norm, $\|\cdot\|_2$. The proximity map for a closed convex set in the Hilbert space with Euclidean inner product satisfies the condition $\|Px - Py\| \leq \|x - y\|$ [9], [13]. Since its Lipschitz constant is $K = 1$, according to Theorem 1, Algorithm 1 converges for $\alpha \in [0, 1)$.

Next, for the MAE loss, the projection becomes

$$P_{Z,L}(y) = \arg \min_z \left\{ \frac{1}{n} \sum_{k=1}^n |z_k - y_k| \mid z \in Z \right\}$$

which is optimization with respect to the L_1 norm, $\|\cdot\|_1$. It is shown by [14] that the Lipschitz constant is $K = 2$ with respect to the L_1 norm. Hence convergence is guaranteed if $\alpha \in [0, 0.25)$. \square

Corollary 1 is the main result of this paper establishing the convergence of iterative algorithm for regression with constraints. The key idea which facilitated it is selection of the objective function of the infeasible adjustment as $L(z, (1 - \alpha)y + \alpha\hat{y}^i)$. Interestingly, for a specific choice of the loss function, namely mean squared error (MSE) loss, it is equivalent to the form of $L(z, y) + \frac{1}{\alpha}L(z, \hat{y})$ as described below.

Remark 1. *If the loss function is mean squared error, the procedure in Algorithm 1 and the Moving Targets algorithm in [8] are equivalent after adjusting the parameter α .*

Proof. In the infeasible adjustment case of the Moving Targets, a vector is obtained that considers the original label, y and the current prediction, \hat{y} separately in the form of $L(z, y) + \frac{1}{\alpha}L(z, \hat{y})$. Since the main difference between the two algorithms is in this formulation, we compare the corresponding optimization problems. With $L(z, y) = (1/n) \sum_{k=1}^n (z_k - y_k)^2$ as the MSE loss, Algorithm 1 addresses

$$z_a = \arg \min_z \left\{ \sum_{k=1}^n (z_k - (1 - \alpha_a)y_k - \alpha_a\hat{y}_k)^2 \mid z \in C \right\}$$

Whereas the master step from Moving Targets is,

$$z_m = \arg \min_z \left\{ \sum_{k=1}^n (z_k - y_k)^2 + \frac{1}{\alpha_m} \sum_{k=1}^n (z_k - \hat{y}_k)^2 \mid z \in C \right\}$$

Here, the subscript a represents our algorithm while variables with m as the subscript are from Moving Targets. The objective functions in z_a, z_m differ only by a scale if,

$$\alpha_a(\alpha_m + 1) = 1. \quad (5)$$

Hence, the solutions that are obtained from them will be identical, i.e., $z_a = z_m$. \square

Next, we show that the proposed algorithm further exhibits improved numerical properties in several examples, beyond providing mathematical assurance.

III. NUMERICAL SIMULATION

A. Constrained Learning Problem

We evaluate the performance of the proposed algorithm with various datasets, parameter values, and loss functions. First, we underscore that this section is meant to be an exercise in understanding an algorithmic procedure, and the resulting output is supposed to be interpreted as purely technical results. The type of constraints considered in this section is called fairness constraints in socially sensitive decision making (see

TABLE I
PARAMETERS

Weight on \hat{y} w.r.t y	α	
	Algorithm 1, α_a	Moving Targets, α_m
Less	0.1	9
Equal	0.5	1
More	0.9	1/9

[15]), which is measured in the form of Disparate Impact Discrimination Index as

$$DIDI^r(z) = \sum_{p \in P} \sum_{v \in D_p} \left| \frac{1}{n} \sum_{i=1}^n z_i - \frac{1}{|X_{p,v}|} \sum_{i \in X_{p,v}} z_i \right| \leq \epsilon. \quad (6)$$

Here D_p is the set of values for the p -th protected feature, such as gender or disability, from the set P , and $X_{p,v}$ represents the inputs whose p -th feature has value v . Roughly speaking, it represents the difference between the mean of the output and the mean conditioned by the protected feature, and the higher DIDI, the more the dataset suffers from disparate impact. The constraint on the DIDI value, ϵ , is taken to be a fraction (0.2) of the DIDI value for the training set.

Three different datasets are considered for this regression problem with fairness constraints:

- `student` dataset ($n = 649$ points, $d = 33$ attributes) for Portuguese class from the UCI repository which has been used to predict secondary school student performance in [16]. We are going to protect the feature, `sex`, and try to estimate the final grade of each student, `G3`, after removing unrelated features.
- `crime` dataset also from the UC Irvine Machine Learning repository [17] which has $n = 2,215$, $d = 147$. Since the target variable is `violentPerPop` representing per capita violent crimes, we want to impose fairness constraints w.r.t. the protected feature `race`.
- `blackfriday` dataset which is available online at [18]. Considering computational challenges, we select a sample of data from the start with size, $n = 50,000$, among the original training data ($n \approx 550,000$, $d = 12$). Here, the goal is to estimate the amount of money spent, `Purchase`, while ensuring that the predictions are fair with respect to the protected feature, `Gender`. A new attribute, `Product_ID_Count`, which is the value count of `Product_ID` is introduced since it represents the number of times a product has been purchased.

All the categorical features in the data are encoded into an integer array using an Ordinal Encoder. Finally, obtained values are normalized to be between 0 and 1 to ensure balanced regression.

Next the values of parameters α, β are chosen as follows. For the parameter α , three different values are chosen, and the corresponding values of α_m are calculated from (5) in Remark 1. These are listed at Table I where as α_a is increased, more weight is assigned to \hat{y} that has been adjusted for the constraint, compared with the original target y . Therefore,

there is more emphasis on the satisfaction of the constraint. We choose $\beta = 0.1$ according to the empirical study presented in [8].

For the machine learning model of regression, a gradient boosted tree is chosen as it ensures repeatability. It also achieves higher accuracy as well as better constraint satisfaction. To study the convergence property, the algorithms are executed for the total of $N_i = 30$ iterations. A five-fold cross validation is performed to obtain a reliable estimate of performance as well as the standard deviation. We utilize a computer with Intel i7 CPU and Nvidia GK107 GPU with 16 GB RAM. To solve the optimization problems in the adjustment step of Algorithm 1, we utilize the IBM software CPLEX [19] for MSE and MAE. Additionally, we consider the mean Huber loss (MHL) $L(z, y) = \frac{1}{n} \sum_{k=1}^n g(z_k - y_k)$, where

$$g(x) = \begin{cases} x^2, & |x| \leq M \\ 2M|x| - M^2, & |x| > M \end{cases} \quad (7)$$

with $M = 0.1$, which is implemented by CVXPY [20].

B. Numerical Results

Table II presents the results for varying loss functions, datasets and α values. Performance is measured through the regression coefficient R^2 , and the ratio \mathcal{C} of DIDI (6) of the predicted output to training data. We also compare between our algorithm (denoted by A) and the Moving Targets (M) for the corresponding α values from Table I. According to Remark 1, both methods are equivalent for MSE. For the `blackfriday` dataset, two cases of α are left out since they could not be solved with the available computing resources. The trade-off between constraint satisfaction and regression is well reflected in Table II: as α_a is increased, \mathcal{C} decreases at the cost of reduced R^2 .

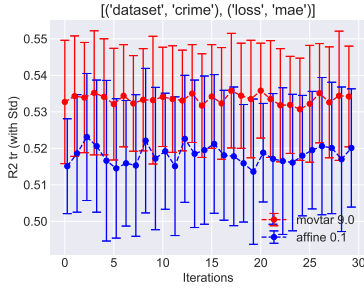
Next, the **bold** fonts in Table II represent the cases for which our algorithm performs better than Moving Targets in a statistically meaningful manner, and the *italic* fonts represent the opposite case. The statistical importance is assumed to occur when $|\mu_a - \mu_m| \geq \sigma_a + \sigma_m$, i.e., the difference between the mean figures is greater than the sum of their standard deviations. It can be observed that our procedure performs better in terms of both R^2 and \mathcal{C} in more cases for both `crime` and `blackfriday` datasets. For the `student` dataset, which is the smallest one ($n = 649$), the results are mostly comparable.

Beyond the regression results summarized by Table II, the advantages of the proposed approach are well illustrated by investigating the learning process. Figures 1 and 2 presents the evolution of R^2 and \mathcal{C} over iterations for `crime` and `blackfriday` data, respectively. When α_a is small (0.1), both the algorithms yield very similar results as seen in Figures 1.(a) and (d).

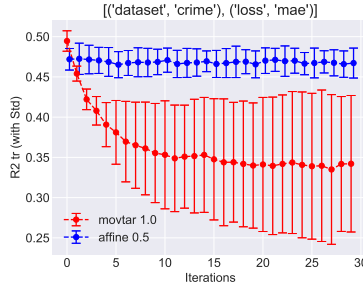
However, once α_a is increased to 0.5 and 0.9 for more emphasis on constraint satisfaction, the proposed Algorithm 1 performs noticeably better. As shown in Figures 1.(b) and (e), and also in Figures 2.(a) and (c), the proposed approach yields a greater R^2 with a lower \mathcal{C} . Next, in Figures 1.(c) and (f), and

TABLE II
PERFORMANCE AFTER $N_i = 30$ ITERATIONS; SHOWN AS *mean (std)* OF 5 FOLDS

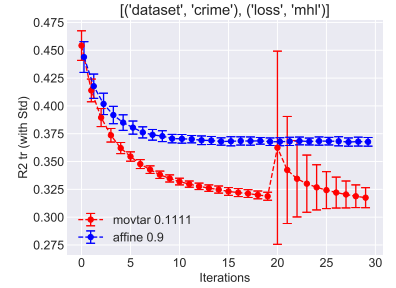
Loss	α_a		crime		student		blackfriday	
			A	M	A	M	A	M
MSE	0.1	R^2	.550 (.013)		.921 (.010)		.645 (.002)	
		C	.262 (.013)		.325 (.044)		.567 (.016)	
	0.5	R^2	.494 (.012)		.908 (.012)		.620 (.001)	
		C	.237 (.006)		.289 (.027)		.511 (.026)	
	0.9	R^2	.368 (.004)		.881 (.022)		.481 (.003)	
		C	.216 (.004)		.241 (.006)		.358 (.013)	
MAE	0.1	R^2	.520 (.016)	.534 (.014)	.891 (.018)	.888 (.027)	.645 (.003)	.647 (.002)
		C	.260 (.014)	.280 (.007)	.331 (.054)	.318 (.041)	.582 (.018)	.577 (.017)
	0.5	R^2	.467 (.019)	.342 (.085)	.874 (.019)	.883 (.029)	.624 (.002)	.590 (.003)
		C	.239 (.013)	.265 (.005)	.333 (.054)	.327 (.026)	.478 (.018)	.577 (.028)
	0.9	R^2	.383 (.062)	.359 (.043)	.799 (.051)	.785 (.047)	.502 (.002)	.295 (.005)
		C	.220 (.011)	.215 (.007)	.278 (.026)	.212 (.021)	.256 (.010)	.219 (.006)
MHL	0.1	R^2	.530 (.013)	.534 (.013)	.921 (.011)	.923 (.010)	—	—
		C	.272 (.008)	.276 (.012)	.326 (.048)	.311 (.049)	—	—
	0.5	R^2	.493 (.010)	.489 (.013)	.907 (.013)	.900 (.015)	.620 (.001)	.611 (.003)
		C	.248 (.007)	.258 (.006)	.289 (.035)	.291 (.044)	.511 (.026)	.509 (.025)
	0.9	R^2	.368 (.004)	.318 (.009)	.882 (.022)	.860 (.024)	—	—
		C	.217 (.002)	.207 (.004)	.241 (.006)	.232 (.010)	—	—



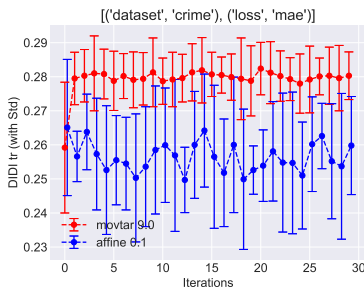
(a) R^2 for $\alpha_a = 0.1$, using MAE



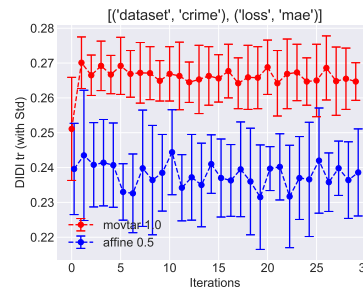
(b) R^2 for $\alpha_a = 0.5$, using MAE



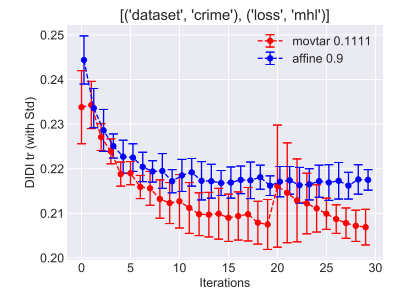
(c) R^2 for $\alpha_a = 0.9$, using MHL



(d) C for $\alpha_a = 0.1$, using MAE



(e) C for $\alpha_a = 0.5$, using MAE



(f) C for $\alpha_a = 0.9$, using MHL

Fig. 1. Comparison of our algorithm (blue) vs Moving Targets (red) for *crime* dataset; error bars represent standard deviation

in Figures 2.(b) and (d), it exhibits greater values of R^2 while being comparable in terms of the constraint satisfaction. More importantly, the proposed approach displays more uniform performances over five-fold validation as the standard deviation is much lower, for example as illustrated by Figures 1.(b), 1.(c), and 2.(c). This suggests that the presented proof of convergence is in fact beneficial in numerical implementations as well, and the improved numerical properties in iterations may be more important for the scalability of regression and the complexity of constraints.

IV. CONCLUSIONS

We have proposed an iterative algorithm for regression with constraints, composed of feasible/infeasible adjustments and training. A convergence guarantee is also provided with an affine extension function in the infeasible adjustment step. Later, the results of numerical experiments are presented with varying datasets and parameter values. The proposed convergence proof in supervised learning with constraints is the unique contribution, and it is further shown that the

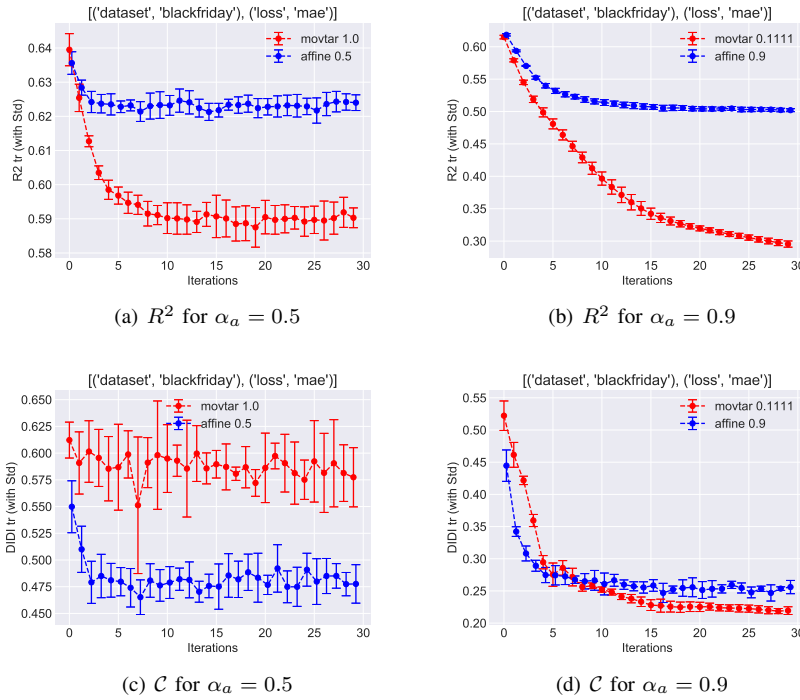


Fig. 2. Comparison of our algorithm (blue) vs Moving Targets (red) for `blackfriday` dataset using MAE; error bars represent standard deviation

performances in all of the aspects of regression, constraint satisfaction and training stability are improved over the existing techniques. For future direction, we aim to study a convergence guarantee in more generic, non-Lipschitz conditions, and even for classification setups.

REFERENCES

- [1] S. Yang, X. He, and B. Zhu, "Learning physical constraints with neural projections," *arXiv preprint arXiv:2006.12745*, 2020.
- [2] R. Berk, H. Heidari, S. Jabbari, M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth, "A convex framework for fair regression," *arXiv preprint arXiv:1706.02409*, 2017.
- [3] A. Borghesi, F. Baldo, and M. Milano, "Improving deep learning models via constraint-based domain knowledge: a brief survey," *arXiv preprint arXiv:2005.10691*, 2020.
- [4] S. V. Mehta, J. Y. Lee, and J. Carbonell, "Towards semi-supervised learning for deep semantic role labeling," *arXiv preprint arXiv:1808.09543*, 2018.
- [5] M. Diligenti, S. Roychowdhury, and M. Gori, "Integrating prior knowledge into deep learning," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2017, pp. 920–923.
- [6] P. Márquez-Neila, M. Salzmann, and P. Fua, "Imposing hard constraints on deep networks: Promises and limitations," *arXiv preprint arXiv:1706.02025*, 2017.
- [7] Y. Nandwani, A. Pathak, P. Singla, *et al.*, "A primal dual formulation for deep learning with constraints," 2019.
- [8] F. Detassis, M. Lombardi, and M. Milano, "Teaching the old dog new tricks: Supervised learning with constraints," *arXiv preprint arXiv:2002.10766*, 2020.
- [9] W. Cheney and A. A. Goldstein, "Proximity maps for convex sets," *Proceedings of the American Mathematical Society*, vol. 10, no. 3, pp. 448–450, 1959.
- [10] S. Boyd and J. Dattorro, "Alternating projections," *EE392o, Stanford University*, 2003.
- [11] H. H. Bauschke and J. M. Borwein, "Dykstra's alternating projection algorithm for two sets," *Journal of Approximation Theory*, vol. 79, no. 3, pp. 418–443, 1994.
- [12] K. Ciesielski *et al.*, "On Stefan Banach and some of his results," *Banach Journal of Mathematical Analysis*, vol. 1, no. 1, pp. 1–10, 2007.
- [13] V. Balestro, H. Martini, and R. Teixeira, "Convex analysis in normed spaces and metric projections onto convex bodies," *arXiv preprint arXiv:1908.08742*, 2019.
- [14] D. G. De Figueiredo and L. Karlovitz, "On the radial projection in normed spaces," in *Djairo G. de Figueiredo-Selected Papers*. Springer, 1967, pp. 11–15.
- [15] S. Aghaei, M. J. Azizi, and P. Vayanos, "Learning optimal and fair decision trees for non-discriminative decision-making," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1418–1426.
- [16] P. Cortez and A. M. G. Silva, "Using data mining to predict secondary school student performance," 2008.
- [17] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [18] Black friday dataset. [Online]. Available: <https://www.kaggle.com/sdolezel/black-friday>
- [19] IBM ILOG CPLEX, "V12. 1: User's manual for CPLEX," *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.
- [20] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.