Notation and Conventions

General

- If $f = g + h$ and g is differentiable but h is non-differentiable, overload the ∇ operator by defining $\nabla f \coloneqq \nabla g$.
- $\|\cdot\|$: Euclidean norm
- $\angle(v, w)$: the angle between vectors v and w

Non-Federated Sparse Linear Regression

- (X, Y): training dataset
- λ: L1 regularization parameter
- β: model
- $f(\beta) \coloneqq f(X, Y, \lambda, \beta)$: objective function of sparse linear regression
- $\beta^* \coloneqq \arg\min_\beta f(\beta)$: optimal model
- β^s: model at s-th iteration
- η: step-size
- $S_{\lambda\eta}(\cdot)$: soft-thresholding operator

Federated Sparse Linear Regression

- K: number of clients
- G: communication graph
- $\mathrm{ne}(i)$: neighborhood of i-th client
- $K_i \coloneqq 1 + |\mathrm{ne}(i)|$: the number of neighbors of the i-th client including itself
- Let the index i range over $\{1, \ldots, K\}$, and the index j range over $\{i\} \cup \mathrm{ne}(i)$.
- (X_i, Y_i): i-th private training dataset
- $(X, Y) \coloneqq \bigcup_i (X_i, Y_i)$: joint training dataset
- $f_i(\beta) \coloneqq f(X_i, Y_i, \lambda, \beta)$: i-th objective function
- β_i: i-th local model

1 Problem Statement

The most basic problem is "sparse linear regression", which is well understood. This problem is generalized by "federated sparse linear regression", which is in turn generalized by the main problem, "federated sparse linear regression with poisoning attacks".

1.1 (Non-Federated) Sparse Linear Regression

The most basic setting is the non-federated setting, in which a single client possesses a training dataset (X, Y) and wants to find an optimal model $\beta^* := \arg\min_\beta f(X, Y, \lambda, \beta)$. For conciseness we can suppress X and Y and simply write the objective function as $f(\beta)$. The objective function $f(\beta)$ is the objective function for sparse linear regression.

1.2 Federated Sparse Linear Regression

More generally, in the federated setting there are K clients, each possessing a private training dataset (X_i, Y_i). Define the joint training dataset $(X, Y) = \bigcup_i (X_i, Y_i)$. The clients want to find an optimal model $\beta^* = \arg\min_\beta f(\beta)$ (the objective function depends on the joint training dataset). To collaborate, the clients broadcast messages to their neighbors on a graph G. These messages must not include private training datasets, but can include models and gradients.

1.3 Federated Sparse Linear Regression with Poisoning Attacks

Even more generally, assume that all clients are either benign or adversarial. Benign clients want to recover the optimal model as before, but adversaries will intentionally broadcast incorrect models and/or gradients to their neighbors in order to prevent convergence.

2 Methods

Sparse linear regression is solved by "proximal gradient descent". We examine how this method can be generalized to "federated sparse linear regression", and generalized again to "federated sparse linear regression robust to poisoning attacks".

2.1 Proximal Gradient Descent

All algorithms to follow will generalize proximal gradient descent, which solves non-federated sparse linear regression and has the following steps:

Gradient (G): $\beta_i \leftarrow \beta_i - \eta \nabla f_i(\beta_i)$.

Threshold (T): $\beta_i \leftarrow S_{\lambda\eta}(\beta_i)$.

2.2 Federated Proximal Gradient Descent

The first generalization of proximal gradient descent is to the federated setting. A federated method must include an aggregation step:

Consensus (C): $\beta_i \leftarrow \beta_i + \frac{\eta}{K_i}\sum(\beta_j - \beta_i)$

Aggregate (A): $\beta_i \leftarrow \beta_i - \frac{\eta}{K_i}\sum \nabla f_j(\beta_j)$

Aggregate Plus (A+): $\beta_i \leftarrow \beta_i - \frac{\eta}{K_i}\sum \nabla f_j(\beta_i)$.

Aggregation steps can be performed one after the other (sequentially) or at the same time (simultaneously). For now we will ignore simultaneous aggregation steps involving A+.

(AC): $\beta_i \leftarrow \beta_i - \frac{\eta}{2K_i}\sum (\nabla f_j(\beta_j) - (\beta_j - \beta_i))$

(GC): $\beta_i \leftarrow \beta_i - \frac{\eta}{2K_i}(K_i\nabla f_i(\beta_i) - \sum(\beta_j - \beta_i))$.

To enable aggregation steps, we also require broadcasting steps. All aggregation steps except for A+ require a single broadcast (B). A+ actually requires two rounds of broadcasting (B+).

Broadcast (B): $i \xrightarrow{m\subseteq\{\beta_i, \nabla f_i(\beta_i)\}} j$

Broadcast Plus (B+): $i \xrightarrow{\{\beta_i\}} j \xrightarrow{\{\nabla f_j(\beta_i)\}} i$

We have an alphabet of G, T, C, A, A+, (AC), (GC), B, and B+. The letters G, C, A, A+, (AC), and (GC) are "aggregation steps" (think of G as a degenerate case) and the letters B and B+ are "broadcasting steps". All of the methods for federated proximal gradient descent are strings consisting of letters of the alphabet. We need grammatical rules determining which strings are "well-formed". These rules will yield a list of methods worth investigating, and will be informed by common sense assumptions.

Assumption: Output must be sparse.

1. T occurs exactly once and occurs last.

Assumption: No more gradients than necessary.

2. A+ is banned.
3. Exactly one of A and G occurs.

Assumption: No more broadcast rounds than necessary.

4. If A or C occurs, then B occurs exactly once and occurs directly before the earliest A or C.

5. C occurs at most once.

   Assumption: Don't use information that is out of date.

6. AC and CA are banned.

Thus, the complete list of methods: G, A, (AC), CG, GC, (GC).

Name	Rule	Cost	"Informativeness"
G	$\beta_i \leftarrow \beta_i - \eta \nabla f_i(\beta_i)$	Low	Low
C	$\beta_i \leftarrow \beta_i + \frac{\eta}{K_i} \sum (\beta_j - \beta_i)$	Medium	Medium
A	$\beta_i \leftarrow \beta_i - \frac{\eta}{K_i} \sum \nabla f_j(\beta_j)$	Medium	Medium
A+	$\beta_i \leftarrow \beta_i - \frac{\eta}{K_i} \sum \nabla f_j(\beta_i)$	High	High

2.3 Federated Proximal Gradient Descent Robust to Poisoning Attacks

To generalize further to the setting with poisoning attacks, we can take the aggregation scheme (i.e., A, (AC), etc.) and turn it into a weighted sum rather than a regular sum. The weight given to the term from client j should depend on the amount that client i "trusts" client j. How trust is computed depends on the information available to client i.

For example, suppose that client i sees β_i^s, β_i^{s-1}, β_j^s, and β_j^{s-1}, that is, its own local model during the current and previous iteration, as well as the local model of client j during the current and previous iteration. It would not be suspicious if the models of client i and client j approach each other over time. Therefore, we can check

$$\|\beta_i^s - \beta_j^s\| < \|\beta_i^{s-1} - \beta_j^{s-1}\|.$$

Any method including C already involves broadcasting models, and can compute this with no additional broadcasts.

Suppose alternatively that client i sees $\nabla f_i(\beta)$ and $\nabla f_j(\beta)$. It would not be suspicious if the angle between these gradients is small. Therefore, we can check

$$\cos(\angle(\nabla f_i(\beta), \nabla f_j(\beta))).$$

To ensure that the angle between gradients should be small, the two gradients must be evaluated at the same model. If both gradients are evaluated at β_j, then client i has

Argument	Rule	Use Case
Models	$\|\beta_i^s - \beta_j^s\| < \|\beta_i^{s-1} - \beta_j^{s-1}\|$	C
Gradients	$\cos(\angle(\nabla f_i(\beta_j), \nabla f_j(\beta_j)))$	A
Gradients	$\cos(\angle(\nabla f_i(\beta_i), \nabla f_j(\beta_i)))$	A+

3 The Written Qualifying Exam

WRITE ABOUT HOW THE WQE FITS INTO THIS FRAMEWORK