Communication-Efficient Gradient Coding for Straggler Mitigation in Distributed Learning

Swanand Kadhe
UC Berkeley

Joint work with
O. Ozan Koyluoglu
Kannan Ramchandran

ISIT '20
June 21-26, 2020
Supervised Learning

Task e.g., classification

Data: \{ (x_i, y_i) \}_{i=1}^{M}

Goal is to find:

Model \( w \) e.g., neural network

- Empirical Risk Minimization (ERM) problem

\[ \arg \min_w \frac{1}{M} \sum_{i=1}^{M} \ell(x_i, y_i; w), \]

- Gradient descent is the standard workhorse
Distributed Learning

- Empirical Risk Minimization (ERM) problem

\[
\min_w \frac{1}{M} \sum_{i=1}^{M} \ell(x_i, y_i; w),
\]

- \( w \in \mathbb{R}^d \): model parameters
- \( D = \{(x_i, y_i)\}_{i=1}^{M} \): dataset of \( M \) samples
- \( \ell(x_i, y_i; w) \): loss function

- Samples are distributed across machines
- Synchronous gradient descent is the standard workhorse

\[
g_i = \sum_{j \in D_i} \nabla \ell(x_j, y_j; w)
\]

\[
g_1 + g_2 + g_3 + g_4
\]

[Chen et al. '16]
Distributed Learning: Challenges

Results on Swiss National Supercomputer [Alistarh et al. ’18]

Time to train ResNet-152 on ImageNet

[Alistarh et al. ’18]
Distributed Learning: Challenges

Results on Swiss National Supercomputer [Alistarh et al. ’18]

Time to train ResNet-152 on ImageNet

- Gains due to parallelization are often limited due to
  - Stragglers
  - Communication overheads

[Alistarh et al. ’18]
Distributed Learning: Serverless Systems

Serverless Systems

- Emerging platform expected to dominate cloud computing
  - AWS Lambda, Google Cloud Functions, Windows Azure Functions
- The challenges become more daunting!
  - Capability to invoke tens of thousands of nodes: more stragglers
  - Stateless nodes communicate via shared memory: communication is costlier

Stragglers in AWS Lambda

[Gupta et al. '18]
Gradient Coding

- Coding theoretic framework for mitigating stragglers
- [Tandon et al. '16, Dutta et al. '16, Halbawi et al. '17, Raviv et al. '18, Ozfatura et al. 19, Reisizadeh et al. 19]

\[ g_1 + \frac{g_2}{2} \quad \frac{g_2}{2} - g_3 \quad g_3 + \frac{g_4}{2} \quad g_4 - 2g_1 \]

Parameter Server

\[ g_1 + g_2 + g_3 + g_4 \]
Gradient Coding

- Coding theoretic framework for mitigating stragglers
- [Tandon et al. '16, Dutta et al. '16, Halbawi et al. '17, Raviv et al. '18, Ozfatura et al. 19, Reisizadeh et al. 19]

**Diagram:**
- Parameter Server
- $W_1$, $W_2$, $W_3$, $W_4$
- $D_1$, $D_2$, $D_3$, $D_4$
- $g_1 + \frac{g_2}{2}$, $\frac{g_2}{2} - g_3$, $g_3 + \frac{g_4}{2}$, $g_4 - 2g_1$
- $g_1 + g_2 + g_3 + g_4$

**Communication cost still is a bottleneck!**
- e.g., ResNet: 150 layers with millions of parameters
- How to reduce communication cost and mitigate stragglers?
Communication-Efficient Gradient Coding

- Mitigate stragglers and reduce communication overhead
  - Coding across the coordinates of a gradient
  - Can be thought off as vector-linear codes!

- Each worker splits the $d$-dimensional gradients into $m$ parts
- Communication saving by a factor of $m$

\[
\begin{align*}
\bar{g}_i &= \begin{bmatrix}
\end{align*}
\[
\begin{bmatrix}
\end{bmatrix}
\begin{align*}
\tilde{g}_1 &= g_1(1) + g_2(1) - g_2(2) \\
\tilde{g}_i &= \begin{bmatrix}
\end{bmatrix}
\begin{align*}
g_1 + g_2 + g_3
\end{align*}
\end{align*}
\]

[Ye-Abbe '18]
Ye-Abbe Coding Scheme

\[ g_{\text{mat}} = \text{Gradients reshaped as a } \frac{d}{m} \times mn \text{ matrix} \]

\[
\begin{bmatrix}
g_1 \\
g_2 \\
\vdots \\
g_n
\end{bmatrix} \rightarrow \begin{bmatrix}
gradien( )ec(+r arranged as a ma(rix \\
gmat
gradient vector arranged as a matrix
\end{bmatrix}
\]
Ye-Abbe Coding Scheme

\[ g_{\text{mat}} = \text{Gradients reshaped as a } \frac{d}{m} \times mn \text{ matrix} \]

\[ \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix} \rightarrow \begin{bmatrix} g_{\text{mat}} \end{bmatrix} \]

gradient vector arranged as a matrix

\[ \text{Structure of encoding:} \]

\[ \text{Encoding} = \begin{bmatrix} g_{\text{mat}} \end{bmatrix} \]

\[ \text{Vandermonde or Random Gaussian} \]

\[ \text{specially designed matrix} \]
Ye-Abbe Coding Scheme

\[ \mathbf{g_{mat}} = \text{Gradients reshaped as a} \frac{d}{m} \times mn \text{ matrix} \]

\[
\begin{bmatrix}
\mathbf{g_1} \\
\vdots \\
\mathbf{g_n}
\end{bmatrix}
\rightarrow
\mathbf{g_{mat}}
\]

\text{gradient vector arranged as a matrix}

\textbf{Structure of decoding:}

\text{Decoding} = \begin{bmatrix} \mathbf{g_{mat}} \end{bmatrix} \begin{bmatrix} -1 \\
\text{Vandermonde or Random Gaussian} \\
\text{specially designed matrix} \end{bmatrix} \]
Ye-Abbe Coding Scheme

Structure of decoding:

\[
\text{Decoding} = \begin{bmatrix}
g_{\text{mat}} \\
\end{bmatrix} \begin{bmatrix}
\text{Vanderm} \\
\text{Random Ga} \\
\end{bmatrix}^{-1}
\]

Drawbacks:

- Inverting Vandermonde matrix becomes numerically unstable for \( n \geq 25 \)
  - Gradient descent diverges
- Inverting Random Gaussian matrix incurs heavy computation cost \( O(n^3) \)
  - Training slows down
Our Contributions

Develop a new framework for communication-efficient gradient coding
Our Contributions

Develop a new framework for communication-efficient gradient coding

- Enables us to leverage any linear code to design the encoding and decoding functions

- Gracefully trades off the straggler threshold and communication overhead for smaller decoding complexity and higher numerical stability

- LDPC codes: $O(n)$ decoding complexity

- Systematic MDS codes: higher numerical stability

- Experimentally demonstrate on Amazon EC2 that the proposed scheme reduces the average iteration time by 16%
Our Contributions

Develop a new framework for communication-efficient gradient coding

- Enables us to leverage any linear code to design the encoding and decoding functions
- Gracefully trades off the straggler threshold and communication overhead for smaller decoding complexity and higher numerical stability
  - LDPC codes: $O(\eta)$ decoding complexity
  - Systematic MDS codes: higher numerical stability

Experimentally demonstrate on Amazon EC2 that the proposed scheme reduces the average iteration time by 16%
Our Contributions

- Develop a new framework for communication-efficient gradient coding
  - Enables us to leverage any linear code to design the encoding and decoding functions
  - Gracefully trades off the straggler threshold and communication overhead for smaller decoding complexity and higher numerical stability
    - LDPC codes: $O(\eta)$ decoding complexity
    - Systematic MDS codes: higher numerical stability
  - Experimentally demonstrate on Amazon EC2 that the proposed scheme reduces the average iteration time by 16%
Proposed Framework

Fractional Repetition Placement Scheme:

- Divide $n$ workers into groups of size $N$ each
- Given $k$ data parts $\{D_1, D_2, \cdots, D_k\}$, each worker in group $i$ is assigned

$$D^{(i)} = \{D_{(i-1)l+1}, D_{(i-1)l+2}, \cdots, D_{il}\}$$

$n = 8, N = 4, k = 4, l = 2$
Proposed Framework

Fractional Repetition Placement Scheme:

- Divide \( n \) workers into groups of size \( N \) each.
- Given \( k \) data parts \( \{D_1, D_2, \ldots, D_k\} \), each worker in group \( i \) is assigned

\[
D^{(i)} = \{D_{(i-1)l+1}, D_{(i-1)l+2}, \ldots, D_{il}\}
\]

\[
\begin{array}{cccccccc}
W_1 & W_2 & W_3 & W_4 & W_5 & W_6 & W_7 & W_8 \\
D_1 & D_1 & D_1 & D_1 & D_3 & D_3 & D_3 & D_3 \\
D_2 & D_2 & D_2 & D_2 & D_4 & D_4 & D_4 & D_4 \\
\end{array}
\]

\( n = 8, N = 4, k = 4, l = 2 \)

- Assume \( d = 4 \) dimensional gradients, i.e.,

\[
g_i = \begin{bmatrix} g_i(1) \\ g_i(2) \\ g_i(3) \\ g_i(4) \end{bmatrix}
\]
Proposed Framework

Encoding Scheme (Worker Side):

1. Each worker computes the sum of partial gradients on assigned data

\[
g^{(1)} = \begin{bmatrix} g_1^{(1)} \\ g_1^{(2)} \\ g_1^{(3)} \\ g_1^{(4)} \end{bmatrix} + \begin{bmatrix} g_2^{(1)} \\ g_2^{(2)} \\ g_2^{(3)} \\ g_2^{(4)} \end{bmatrix}
\]

\[
g^{(2)} = \begin{bmatrix} g_3^{(1)} \\ g_3^{(2)} \\ g_3^{(3)} \\ g_3^{(4)} \end{bmatrix} + \begin{bmatrix} g_4^{(1)} \\ g_4^{(2)} \\ g_4^{(3)} \\ g_4^{(4)} \end{bmatrix}
\]
Proposed Framework

Encoding Scheme (Worker Side):

1. Each worker computes the sum of partial gradients on assigned data

![Matrix Diagram]

- Group 1:
  - $g^{(1)}_{\text{mat}} = \begin{bmatrix} g^{(1)}_{1(1)} & g^{(1)}_{1(3)} \\ g^{(1)}_{1(2)} & g^{(1)}_{1(4)} \end{bmatrix}$

- Group 2:
  - $g^{(2)}_{\text{mat}} = \begin{bmatrix} g^{(2)}_{1(1)} & g^{(2)}_{1(3)} \\ g^{(2)}_{1(2)} & g^{(2)}_{1(4)} \end{bmatrix}$

2. Arranges the d-dimensional vector $g^{(i)}$ as a $\frac{d}{m} \times m$ matrix $g^{(i)}_{\text{mat}}$

Group 1 workers: $g^{(1)}_{\text{mat}} = \begin{bmatrix} g^{(1)}_{1(1)} & g^{(1)}_{1(3)} \\ g^{(1)}_{1(2)} & g^{(1)}_{1(4)} \end{bmatrix}$
Proposed Framework

Encoding Scheme (Worker Side):

- Consider an \([N, K]\) code \(\mathcal{C}\) with a generator matrix \(G\)

\([4, 2]\) code: \(G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}\)
Proposed Framework

Encoding Scheme (Worker Side):

- Consider an \([N, K]\) code \(C\) with a generator matrix \(G\)

\[
\begin{bmatrix}
1 & 0 & 1 & 1 \\
0 & 1 & 1 & 2
\end{bmatrix}
\]

\([4, 2]\) code: \(G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}\)

3. Workers in group \(i\) send \(\tilde{g}^{(i)} = g^{(i)}_{\text{mat}} G\)

\[
g^{(i)}_{\text{mat}} = \begin{bmatrix} g^{(i)}(1) & g^{(i)}(3) \\ g^{(i)}(2) & g^{(i)}(4) \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}
\]

\[
\begin{array}{c}
W_1 \\
D_1 \\
D_2 \\
g^{(1)}(1) \\
g^{(1)}(2)
\end{array}
\begin{array}{c}
W_2 \\
D_1 \\
D_2 \\
g^{(1)}(3) \\
g^{(1)}(4)
\end{array}
\begin{array}{c}
W_3 \\
D_1 \\
D_2 \\
g^{(1)}(1) + g^{(1)}(3) \\
g^{(1)}(2) + g^{(1)}(4)
\end{array}
\begin{array}{c}
W_4 \\
D_1 \\
D_2 \\
g^{(1)}(1) + 2g^{(1)}(3) \\
g^{(1)}(2) + 2g^{(1)}(4)
\end{array}
\]
Proposed Framework

Decoding Scheme (Server Side):

1. Suppose there are $t$ non-stragglers in group $i$
2. Server solves for $g^{(i)}_{\text{mat}}$ by performing erasure decoding for code $C$

$\begin{array}{l}
\tilde{g}^{(1)}_3 = g^{(1)}_{(1)} + g^{(1)}_{(3)} g^{(1)}_{()} + g^{(1)}_{()} \\
\tilde{g}^{(1)}_4 = g^{(1)}_{(1)} + 2 g^{(1)}_{()} \\
\end{array}$
Proposed Framework

Decoding Scheme (Server Side):

1. Suppose there are $t$ non-stragglers in group $i$

2. Server solves for $g_{mat}^{(i)}$ by performing erasure decoding for code $C$
   
   ▶ Equivalent to solving
   
   $\begin{bmatrix} \tilde{g}_{i_1}^{(i)} & \tilde{g}_{i_2}^{(i)} & \cdots & \tilde{g}_{i_t}^{(i)} \end{bmatrix} = g_{mat}^{(i)} G^{(i)}$, 

   where $G^{(i)}$ is $K \times t$ submatrix of $G$ corresponding to non-stragglers
Proposed Framework

Decoding Scheme (Server Side):

1. Suppose there are $t$ non-stragglers in group $i$

2. Server solves for $g_{\text{mat}}^{(i)}$ by performing erasure decoding for code $C$

   - Equivalent to solving
     
     $\begin{bmatrix} \tilde{g}_{i_1}^{(i)} & \tilde{g}_{i_2}^{(i)} & \cdots & \tilde{g}_{i_t}^{(i)} \end{bmatrix} = g_{\text{mat}}^{(i)} G^{(i)}$, 

     where $G^{(i)}$ is $K \times t$ submatrix of $G$ corresponding to non-stragglers.

\[ \tilde{g}_3^{(1)} = \begin{bmatrix} g^{(1)}(1) + g^{(1)}(3) \\ g^{(1)}(2) + g^{(1)}(4) \end{bmatrix}, \quad \tilde{g}_4^{(1)} = \begin{bmatrix} g^{(1)}(1) + 2g^{(1)}(3) \\ g^{(1)}(2) + 2g^{(1)}(4) \end{bmatrix} \]
Proposed Framework

Decoding Scheme (Server Side):

1. Suppose there are $t$ non-stragglers in group $i$

2. Server solves for $g_{m_{at}}^{(i)}$ by performing erasure decoding for code $C$

   - Equivalent to solving
     \[
     \begin{bmatrix}
     \tilde{g}_{i_1}^{(i)} & \tilde{g}_{i_2}^{(i)} & \cdots & \tilde{g}_{i_t}^{(i)}
     \end{bmatrix}
     = g_{m_{at}}^{(i)} G^{(i)},
     \]

   where $G^{(i)}$ is $K \times t$ submatrix of $G$ corresponding to non-stragglers

\[
\begin{bmatrix}
W_1 & W_2 & W_3 & W_4
\end{bmatrix}
\begin{bmatrix}
D_1

D_2

D_1

D_2

D_1

D_2

\end{bmatrix}
\]

Group 1

\[
\tilde{g}_3^{(1)} = \begin{bmatrix}
g^{(1)}(1) + g^{(1)}(3) \\
g^{(1)}(2) + g^{(1)}(4)
\end{bmatrix} \quad \tilde{g}_4^{(1)} = \begin{bmatrix}
g^{(1)}(1) + 2g^{(1)}(3) \\
g^{(1)}(2) + 2g^{(1)}(4)
\end{bmatrix}
\]

\[
\begin{bmatrix}
\tilde{g}_3^{(1)} & \tilde{g}_4^{(1)}
\end{bmatrix}
= \begin{bmatrix}
g^{(1)}(1) & g^{(1)}(3) \\
g^{(1)}(2) & g^{(1)}(4)
\end{bmatrix}
\begin{bmatrix}
1 & 1 \\
1 & 2
\end{bmatrix}
\]
How to Choose the Underlying Code?

The underlying code dictates the trade-off between stragglers, communication, and computation

Theorem: The proposed framework using an \([N, K, \delta]\) code achieves the triple \((l = \frac{kN}{n}, m = K, s = \delta - 1)\)
Achieving Optimality

Three dimensional trade-off between $l$, $m$, and $s$ [Ye-Abbe '18]:

$$\frac{l}{k} \geq \frac{s+m}{n}$$

Corollary: The proposed framework using an $[s + m, m]$ MDS code achieves optimal trade-off $\frac{l}{k} = \frac{s+m}{n}$

Remark: Setting $m = 1$ reduces to the first gradient code in [Tandon et al. '16]. Our framework generalizes fractional repetition gradient code in [Tandon et al. '16] to achieve communication savings.
Corollary: Suppose \( s = m = O(n) \). The proposed framework using an \([N, K]\)-LDPC code can tolerate a random set of \( \frac{N-K}{\eta} \) stragglers with \( O(n) \) decoding complexity.

Here \( \eta > 1 \) is the threshold obtained using the density evolution.
Corollary: Suppose $s = m = O(n)$. The proposed framework using an $[N, K]$-LDPC code can tolerate a random set of $\frac{N-K}{\eta}$ stragglers with $O(n)$ decoding complexity.

Here $\eta > 1$ is the threshold obtained using the density evolution.

- Number of workers $n = 30,000$
- Communication savings $m = 5,000$
- Straggler tolerance $s = 4,000$
Trading Off Optimality for Decoding Efficiency

Corollary: Suppose \( s = m = O(n) \). The proposed framework using an \([N, K]\)-LDPC code can tolerate a random set of \( \frac{N-K}{\eta} \) stragglers with \( O(n) \) decoding complexity.

Here \( \eta > 1 \) is the threshold obtained using the density evolution.

- Number of workers \( n = 30,000 \)
- Communication savings \( m = 5,000 \)
- Straggler tolerance \( s = 4,000 \)
- \([10k, 5k]\)-MDS code
  - Tolerates any 5k stragglers
  - Decoding cost = \( O(10^9) \)
Corollary: Suppose \( s = m = O(n) \). The proposed framework using an \([N, K]\)-LDPC code can tolerate a random set of \( \frac{N-K}{\eta} \) stragglers with \( O(n) \) decoding complexity.

Here \( \eta > 1 \) is the threshold obtained using the density evolution.

- Number of workers \( n = 30,000 \)
- Communication savings \( m = 5,000 \)
- Straggler tolerance \( s = 4,000 \)
- \([10k, 5k]\)-MDS code
  - Tolerates any 5k stragglers
  - Decoding cost = \( O(10^9) \)
- \([10k, 5k]\)-LDPC code from \((3, 6)\) ensemble
  - Tolerates random 4294 stragglers
  - Decoding cost = \( O(10^3) \)
Trading Off Optimality for Numerical Stability

Straggler-threshold under numerical stability constraint [Ye-Abbe ’18]: number of stragglers \( s_\kappa \) that can be tolerated such that the condition number of any matrix involved in the decoding is upper bounded by \( \kappa \)
Trading Off Optimality for Numerical Stability

Straggler-threshold under numerical stability constraint [Ye-Abbe ’18]: number of stragglers $s_\kappa$ that can be tolerated such that the condition number of any matrix involved in the decoding is upper bounded by $\kappa$

**Theorem:** For our proposed framework using an $[s + m, m]$ MDS code with a random Gaussian generator matrix, it holds w.h.p., that

$$s_\kappa \geq s + m - \arg \min_{t \in [m, s+m]} f_{m,s,\kappa}(t),$$

where $f_{m,s,\kappa}(t) = \frac{1}{\sqrt{2\pi}} \binom{m+s}{t} \left( \frac{C t}{\kappa(t-m+1)} \right)^{t-m+1}$ and $C \leq 6.414$ is a constant.
Trading Off Optimality for Numerical Stability

Straggler-threshold under numerical stability constraint [Ye-Abbe ’18]:

number of stragglers \( s_\kappa \) that can be tolerated such that the condition number of any matrix involved in the decoding is upper bounded by \( \kappa \)

Theorem: For our proposed framework using an \([s + m, m]\) MDS code with a random Gaussian generator matrix, it holds w.h.p., that

\[
s_\kappa \geq s + m - \arg \min_{t \in [m, s+m]} f_{m,s,\kappa}(t),
\]

where \( f_{m,s,\kappa}(t) = \frac{1}{\sqrt{2\pi}} \left( \frac{m+s}{t} \right) \left( \frac{C t}{\kappa(t-m+1)} \right)^{t-m+1} \)

and \( C \leq 6.414 \) is a constant.

Proof idea:

- Condition number of an \( u \times v \) random Gaussian matrix satisfies [Chen-Dongarra ’05]: \( \Pr \left( \frac{\text{cond}(M)}{v/(|v-u|+1)} > x \right) \leq \sqrt{2\pi} \left( \frac{C}{x} \right)^{|v-u|+1} \)
- Union bound over all possible non-stragglers
Trading Off Optimality for Numerical Stability

[m + s, m]-MDS code vs. Ye-Abbe code for $\kappa = 1000$
(both using random Gaussian matrix)

<table>
<thead>
<tr>
<th>Regime</th>
<th>n</th>
<th>s</th>
<th>m</th>
<th>Ye – Abbe</th>
<th>MDS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Small n</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~ 10%stragglers</td>
<td>60</td>
<td>8</td>
<td>2</td>
<td>→</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>~ 20%stragglers</td>
<td>60</td>
<td>13</td>
<td>12</td>
<td>→</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11</td>
</tr>
<tr>
<td><strong>Large n</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~ 10%stragglers</td>
<td>1000</td>
<td>90</td>
<td>10</td>
<td>→</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>78</td>
</tr>
<tr>
<td>~ 20%stragglers</td>
<td>1000</td>
<td>190</td>
<td>10</td>
<td>→</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>172</td>
</tr>
</tbody>
</table>
Experimental Evaluation on Amazon EC2

- Logistic regression model on Amazon Employee Access dataset
- Number of samples = 26,210  Model dimension = 241,915
- Setup:
  - c3.8xlarge instance as the server
  - n = 60 t2.micro instances as workers

Proposed framework (CommFR-GC) reduces average per iteration time by 16%
Experimental Evaluation on Amazon EC2

- Logistic regression model on Amazon Employee Access dataset
- Number of samples = 26,210   Model dimension = 241,915
- Setup:
  - c3.8xlarge instance as the server
  - n = 60 t2.micro instances as workers

![Graph showing AUC over time for different strategies]

Proposed framework (CommFR-GC) achieves the target generalization faster
Summary and Future Directions

- Communication-efficient gradient coding framework based on fractional repetition
  - Any linear code can be used to design encoding/decoding functions
  - Gracefully trades off straggler threshold and communication overhead for smaller decoding complexity and higher numerical stability
  - Reduces average per iteration time by 16%

Future Directions

- Code constructions using orthogonal polynomials like Chebyshev polynomials inspired by [Fahim-Cadambe ’19]?
- Joint “source-channel” coding?