

First Provably Optimal Asynchronous SGD for Homogeneous and Heterogeneous Data

Artavazd Maranjyan

<https://artomaranjyan.github.io>

AMCS/STAT Graduate Seminar

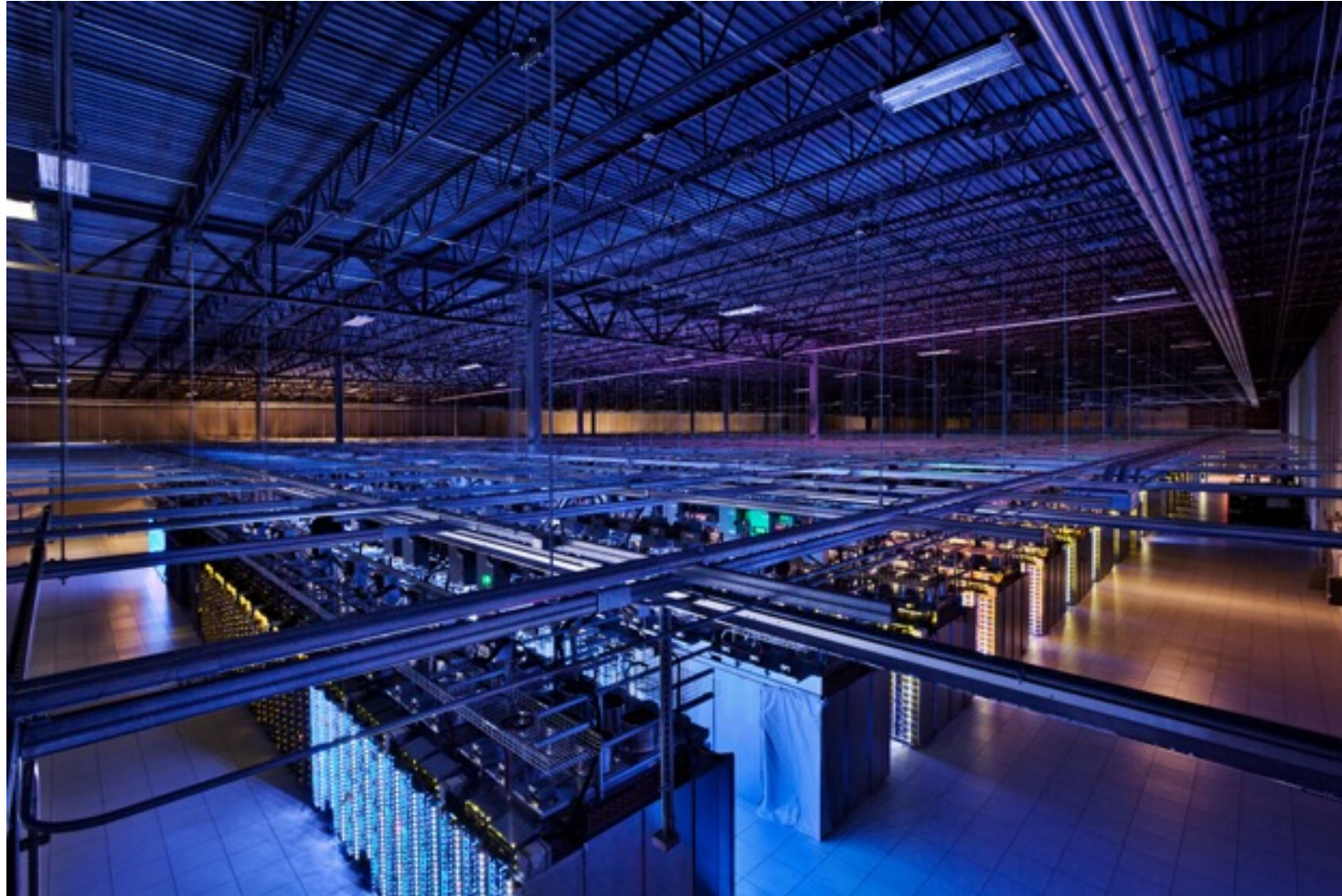
13 November 2025

King Abdullah University of
Science and Technology



جامعة الملك عبد الله
للعلوم والتقنية

Behind ChatGPT: Thousands of GPUs in Huge Data Centers



Outline of the Talk

Problem setup

Outline of the Talk

Problem setup

Asynchronous SGD

Outline of the Talk

Problem setup

Asynchronous SGD

Homogeneous Data Setting

Outline of the Talk

Problem setup

Asynchronous SGD

Homogeneous Data Setting

Heterogenous Data Setting

Training = Mathematical Optimization

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \{ f(x) := \mathbb{E}_{\xi \sim \mathcal{D}} [f(x; \xi)] \}$$

Model parameters
we want to find

The distribution of
the training dataset

$$f(x; \xi) := \text{Loss} (\text{Model} (x; \xi^{\text{input}}) , \xi^{\text{label}})$$

Distributed Learning



$\mathcal{D}_1$

$$f_1(x) := \mathbb{E}_{\xi_1 \sim \mathcal{D}_1} [f_1(x; \xi_1)]$$



$\mathcal{D}_2$

$$f_2(x) := \mathbb{E}_{\xi_2 \sim \mathcal{D}_2} [f_2(x; \xi_2)]$$



$\mathcal{D}_3$

$$f_3(x) := \mathbb{E}_{\xi_3 \sim \mathcal{D}_3} [f_3(x; \xi_3)]$$

Homogenous Data Setting

$$\mathcal{D}_1 = \mathcal{D}_2 = \dots = \mathcal{D}_n = \mathcal{D}$$

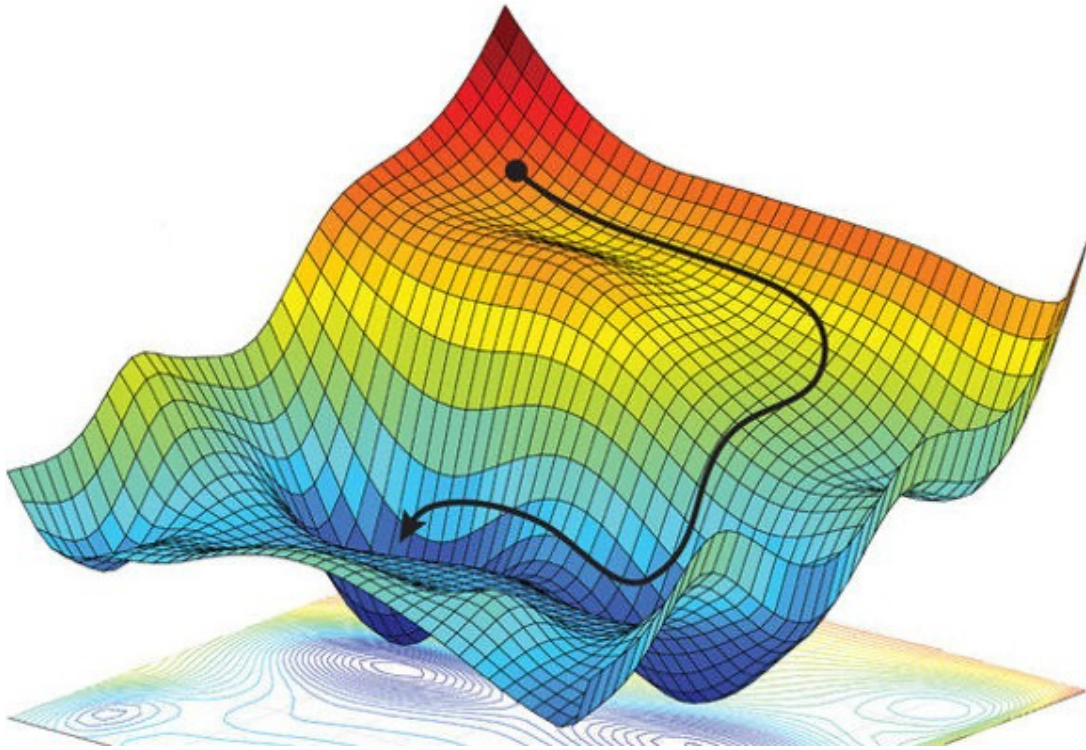
$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \{ f(x) := \mathbb{E}_{\xi \in \mathcal{D}} [f(x; \xi)] \}$$

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \left\{ f(x) := \frac{1}{n} \sum_{i=1}^n f_i(x) \right\}$$



Server

A common method in ML is Stochastic Gradient Descent (SGD)



Stepsize / Learning rate

$$x^{k+1} = x^k - \gamma g(x^k)$$

Unbiased gradient estimator, i.e.

$$\mathbb{E} [g(x^k)] = \nabla f(x^k)$$

$$g(x^k) = \frac{1}{n} \sum_{i=1}^n \nabla f_i(x^k; \xi_i^k)$$

Compute heterogeneous system



$$\nabla f_1(x; \xi_1)$$



$$\nabla f_2(x; \xi_2)$$



$$\nabla f_3(x; \xi_3)$$

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i} [\nabla f_i(x; \xi_i)] = \nabla f_i(x)$$

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i} [\|\nabla f_i(x; \xi_i) - \nabla f_i(x)\|_2^2] \leq \sigma^2$$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L \|x - y\|_2$$



Server

Compute heterogeneous system



$$\nabla f_1(x; \xi_1)$$

Compute time = τ_1

τ_1



$$\nabla f_2(x; \xi_2)$$

Compute time = τ_2

τ_2



$$\nabla f_3(x; \xi_3)$$

Compute time = τ_3

τ_3

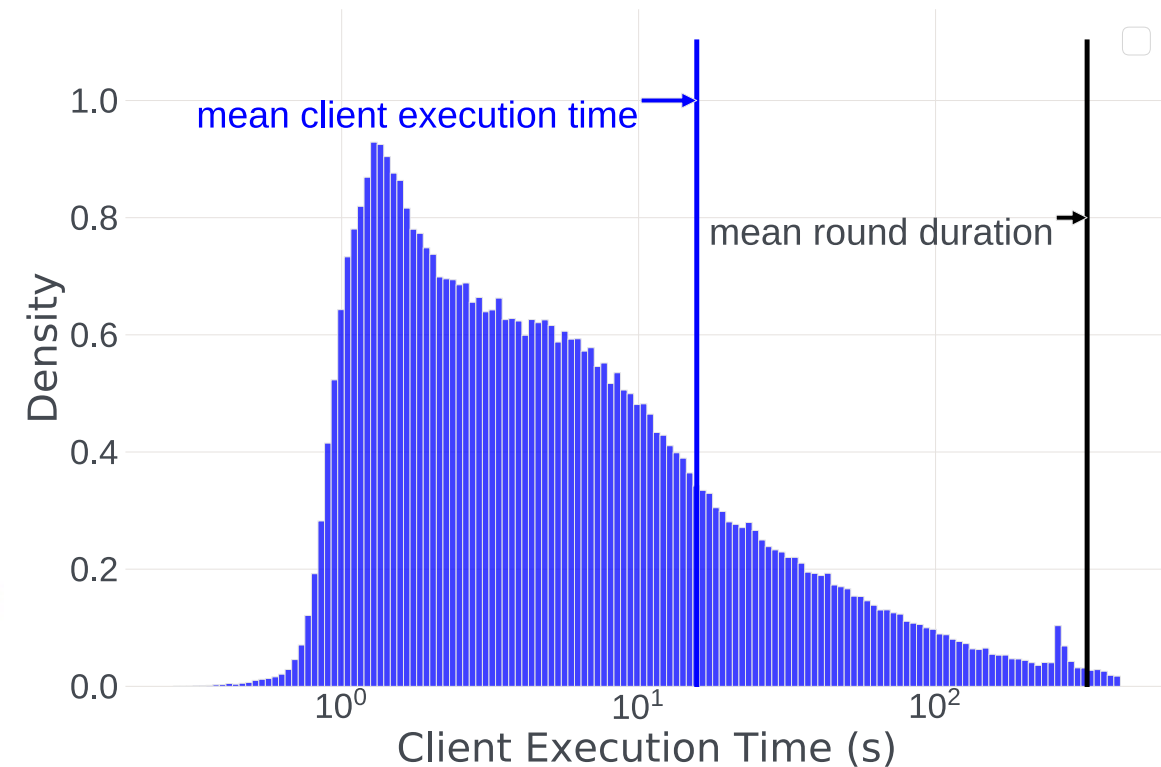
Compute time:
Fixed
Random
Arbitrarily changing (adversarial)



Server

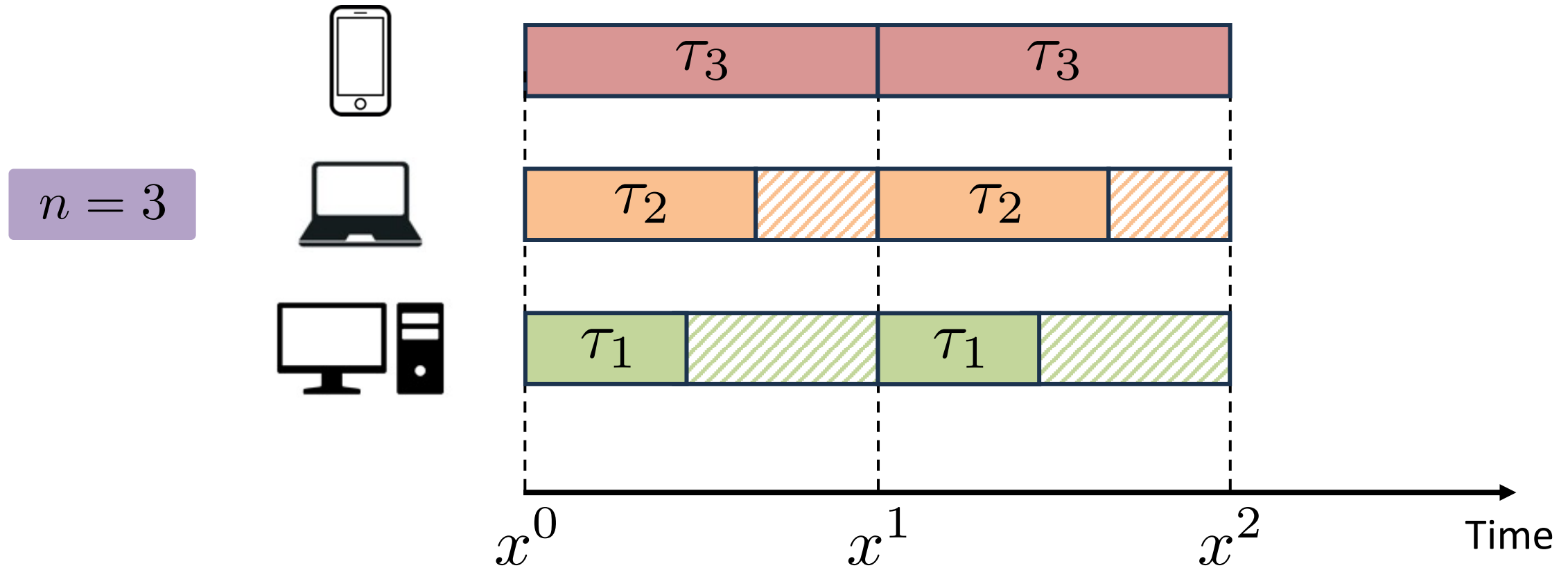
Client Compute Heterogeneity is Real

An LSTM-based language model
on nearly 100 million Android phones.



Naïve Minibatch SGD:

Each worker does one job only

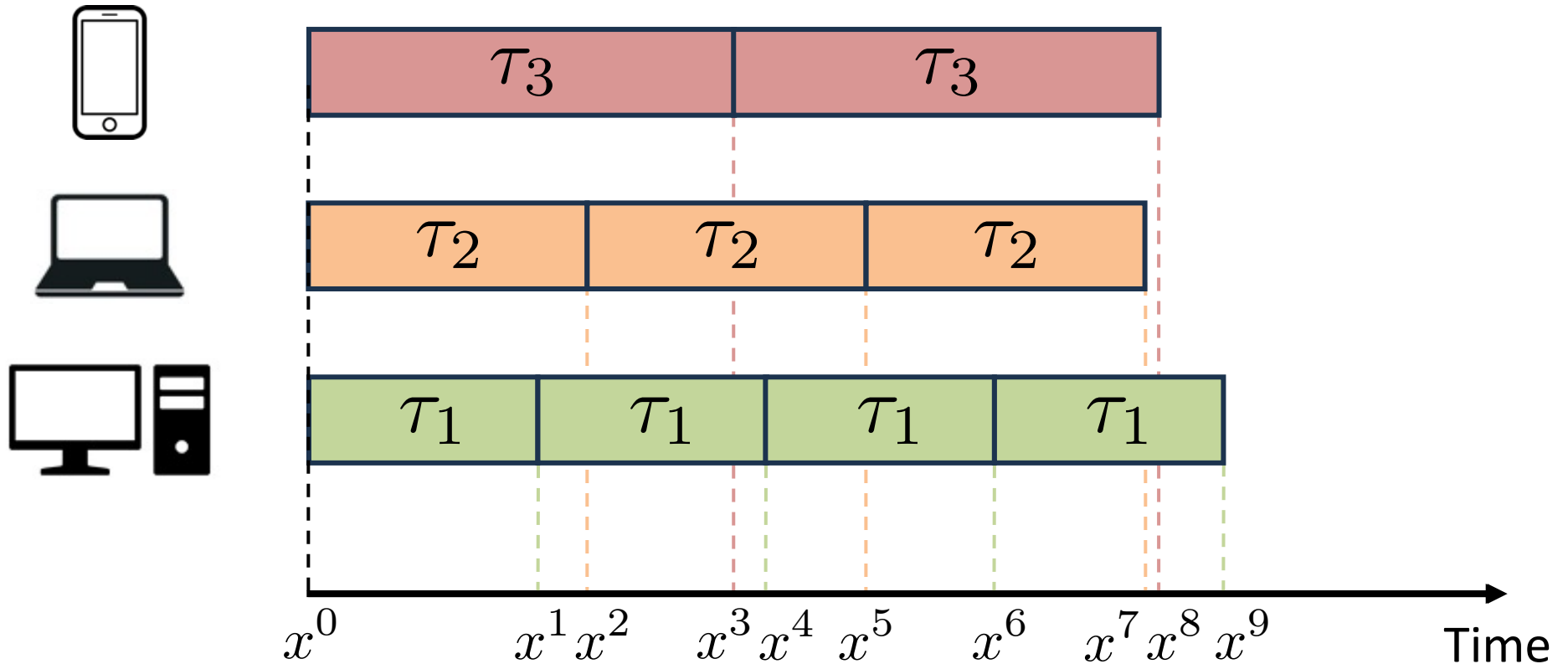


$$x^{k+1} = x^k - \gamma \frac{1}{n} \sum_{i=1}^n \nabla f_i(x^k; \xi_i^k)$$



Asynchronous SGD

Asynchronous SGD: Remove the synchronization



The first Asynchronous SGD

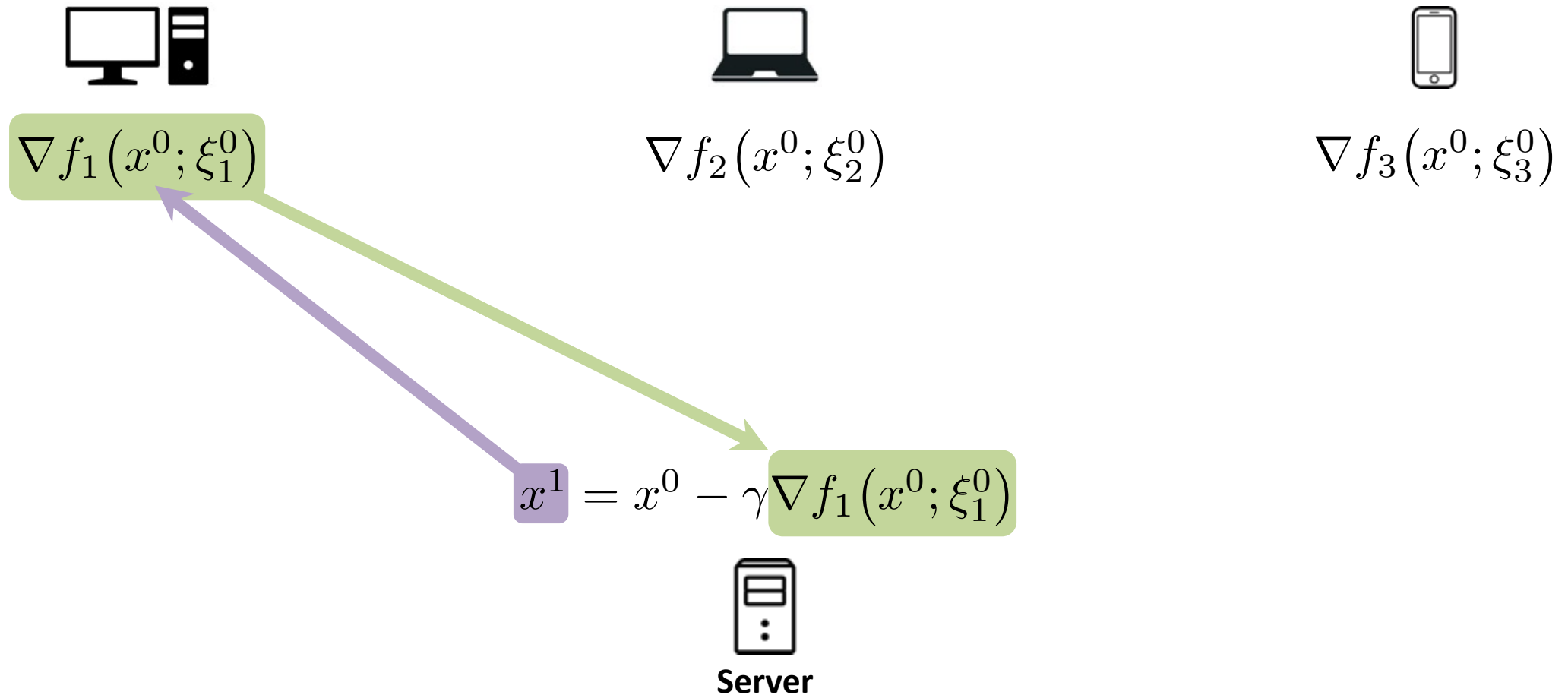


Benjamin Recht, Christopher Re, Stephen Wright, Feng Niu (2011)

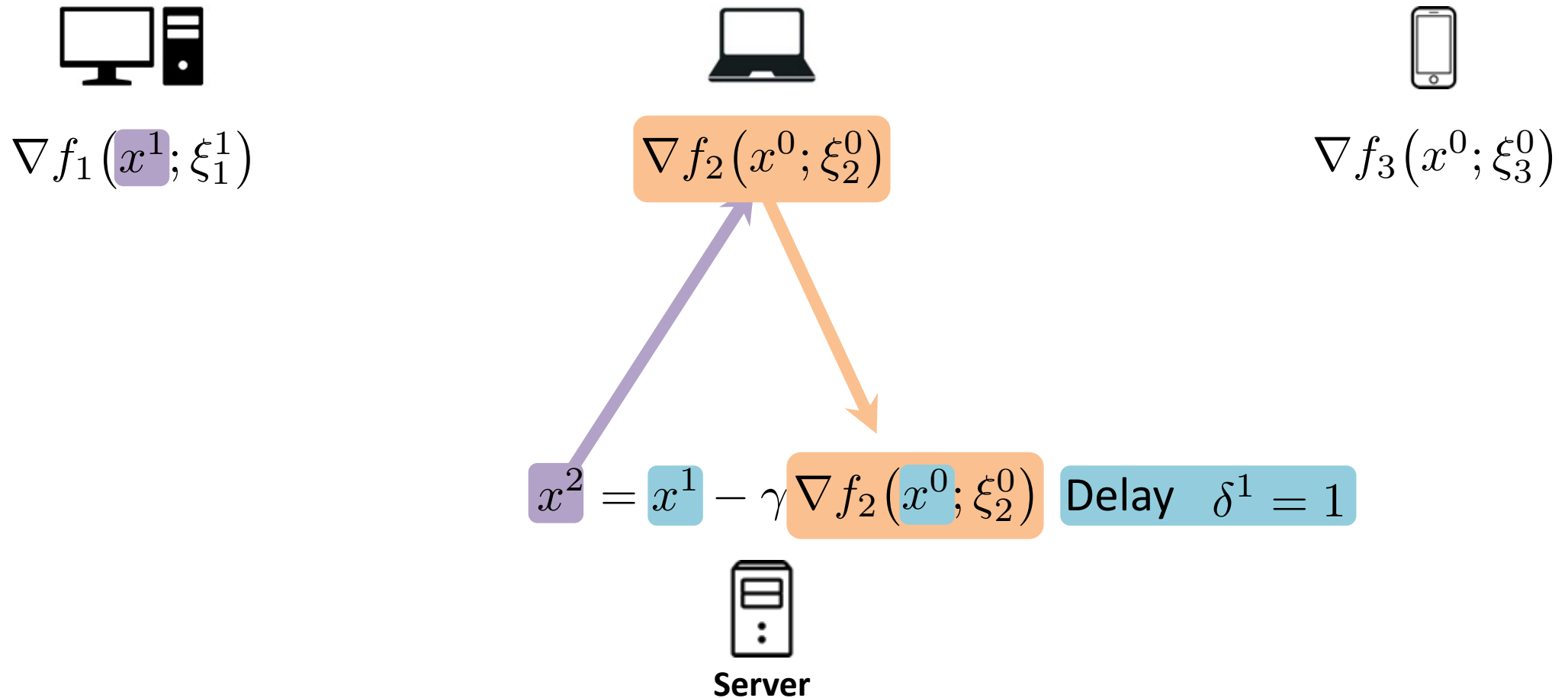
HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent

NeurIPS Test of Time Award, 2020

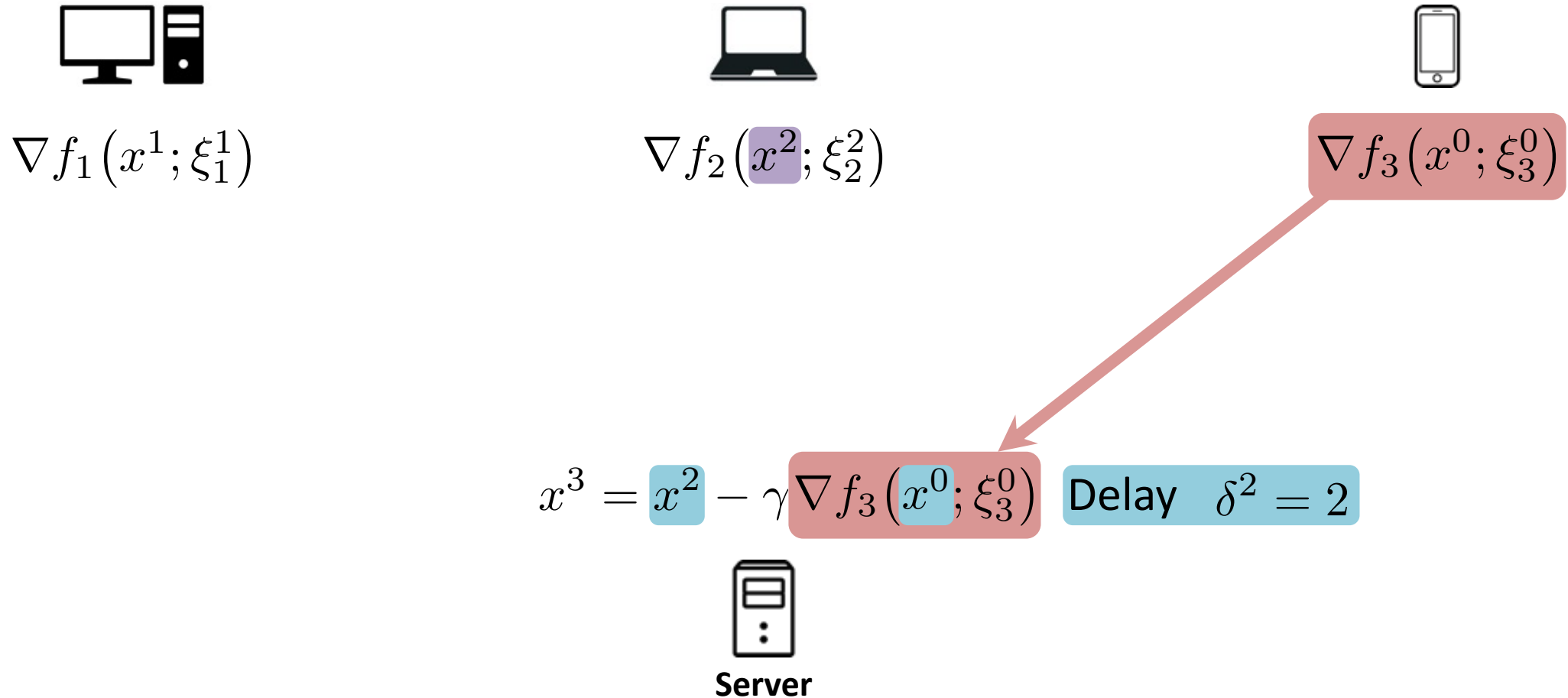
Updates of Asynchronous SGD has delayed stochastic gradients



Updates of Asynchronous SGD has delayed stochastic gradients



Updates of Asynchronous SGD has delayed stochastic gradients



Updates of Asynchronous SGD has delayed stochastic gradients

Delay of worker i^k at iteration k

$$x^{k+1} = x^k - \gamma \nabla f_{i^k} \left(x^{k-\delta_{i^k}^k}; \xi^{k-\delta_{i^k}^k} \right)$$

Uses a gradient from a single worker

Similarity assumption

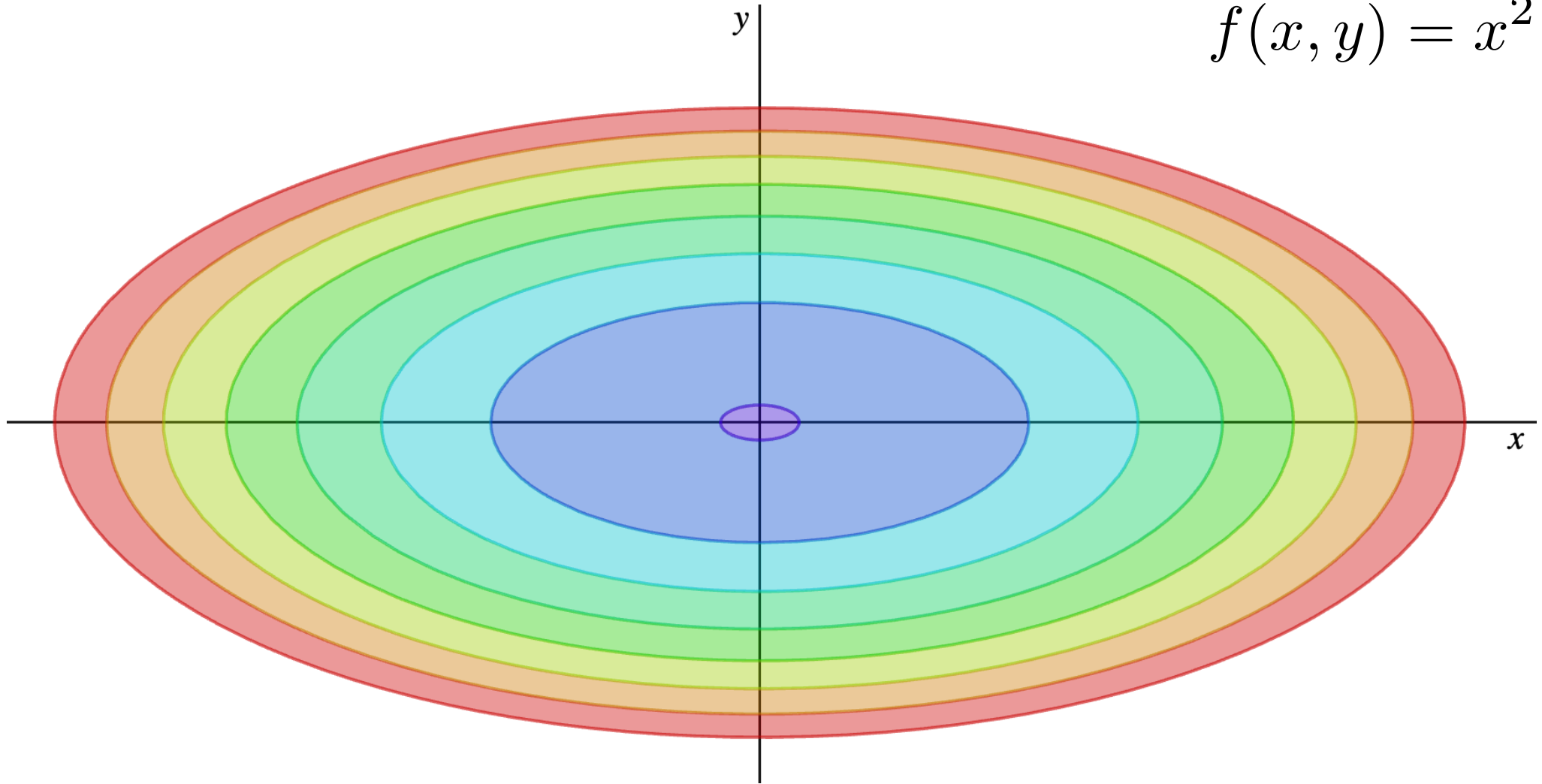
$$\|\nabla f_i(x) - \nabla f(x)\| \leq \zeta$$



Server

Asynchronous SGD can get wild:
delays can degrade performance

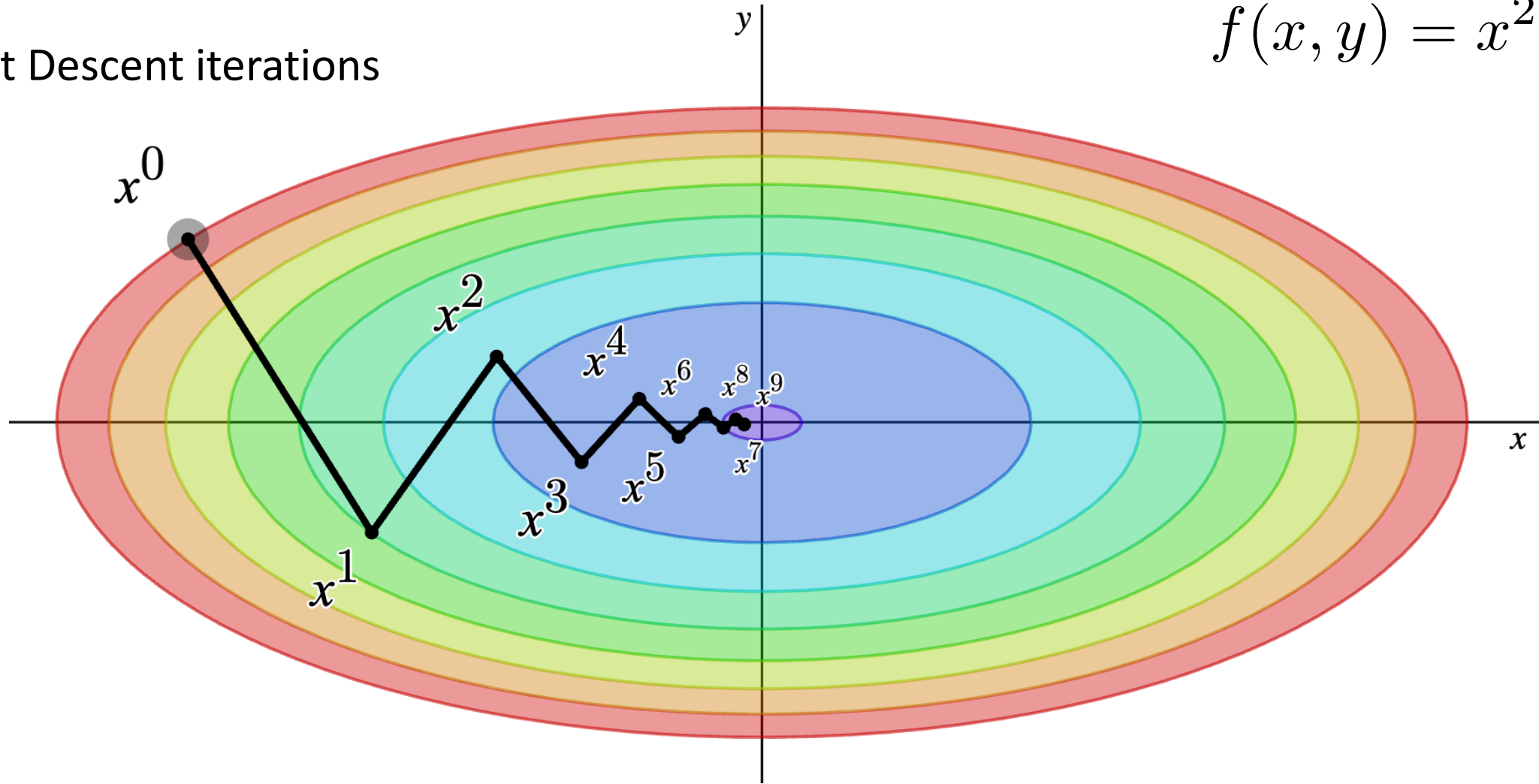
$$f(x, y) = x^2 + 5y^2$$



Asynchronous SGD can get wild: delays can degrade performance

Gradient Descent iterations

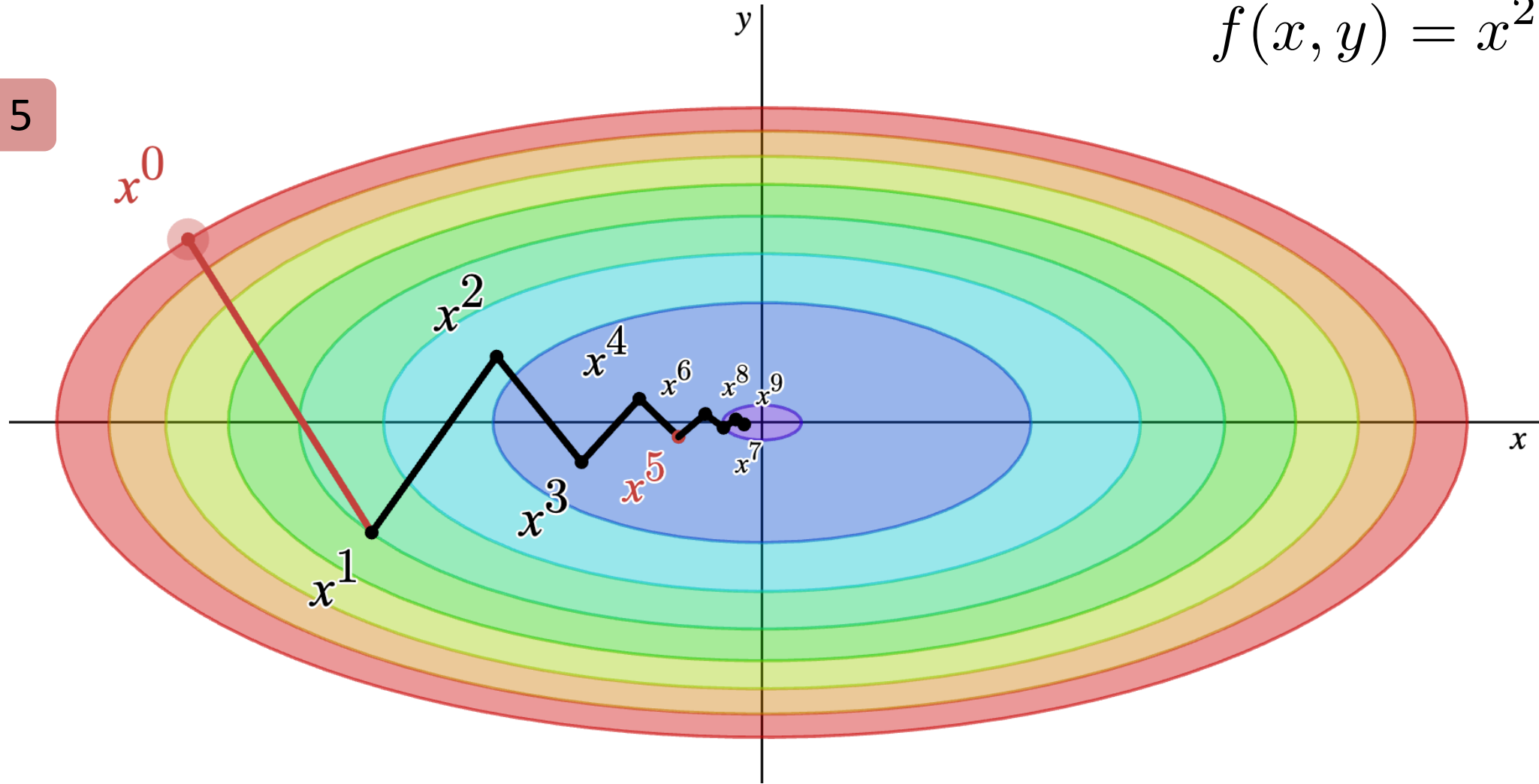
$$f(x, y) = x^2 + 5y^2$$



Asynchronous SGD can get wild: delays can degrade performance

$$f(x, y) = x^2 + 5y^2$$

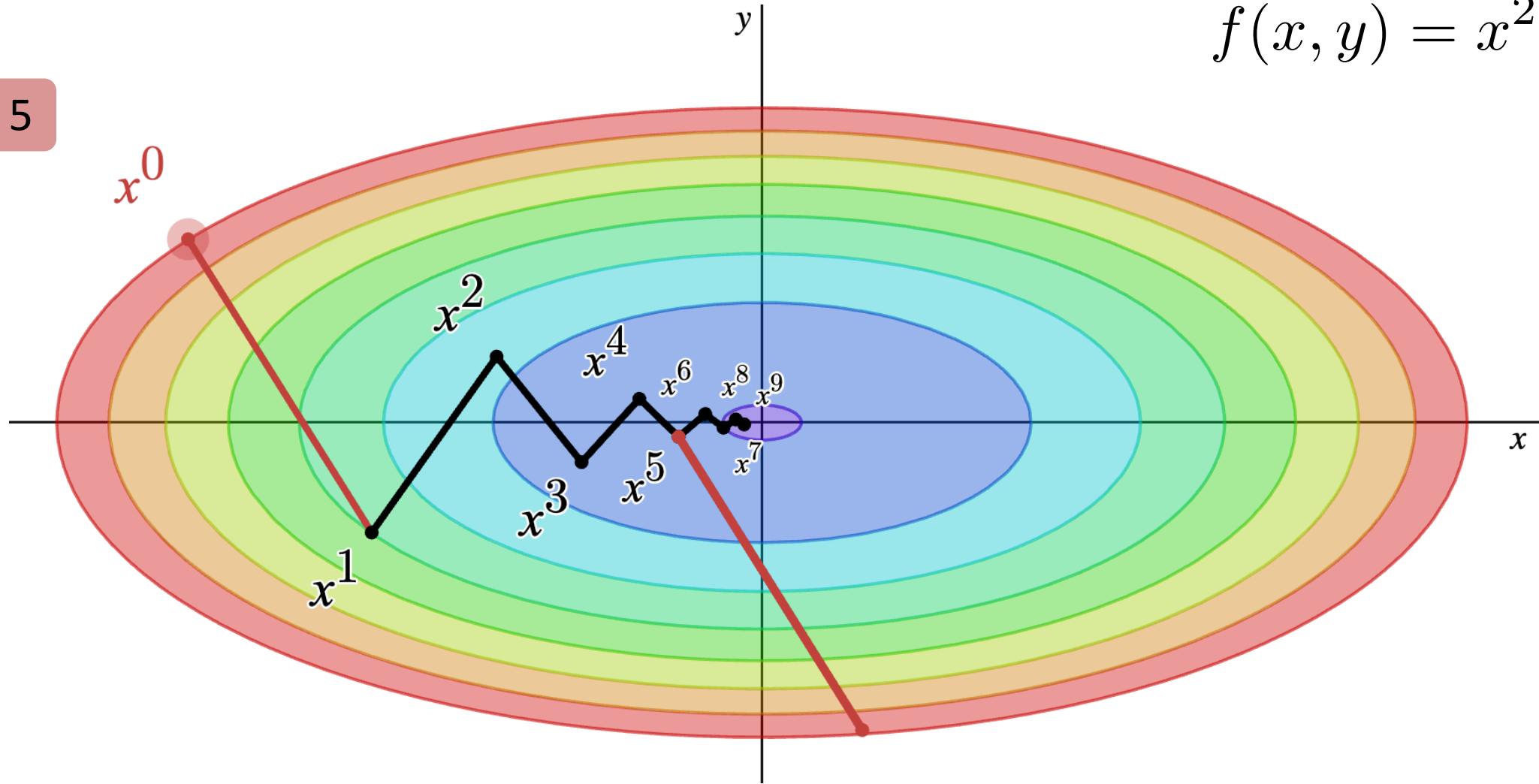
Delay = 5

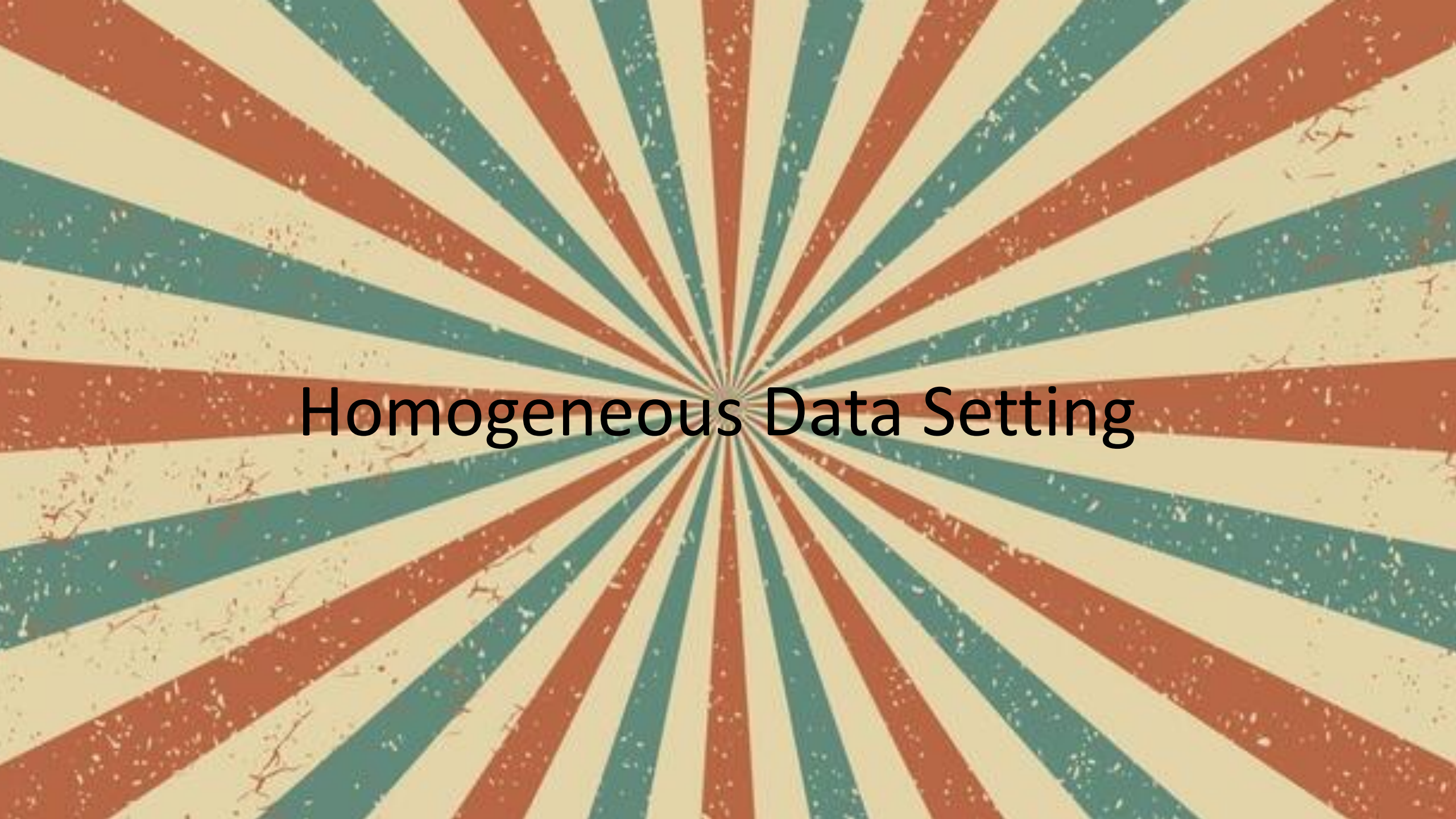


Asynchronous SGD can get wild: delays can degrade performance

$$f(x, y) = x^2 + 5y^2$$

Delay = 5





Homogeneous Data Setting

Homogenous Data Setting



$\mathcal{D}_1$

$$f_1(x) := \mathbb{E}_{\xi_1 \sim \mathcal{D}_1} [f_1(x; \xi_1)]$$



$\mathcal{D}_2$

$$f_2(x) := \mathbb{E}_{\xi_2 \sim \mathcal{D}_2} [f_2(x; \xi_2)]$$



$\mathcal{D}_3$

$$f_3(x) := \mathbb{E}_{\xi_3 \sim \mathcal{D}_3} [f_3(x; \xi_3)]$$

Homogenous Regime

$$\mathcal{D}_1 = \mathcal{D}_2 = \dots = \mathcal{D}_n = \mathcal{D}$$

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \{ f(x) := \mathbb{E}_{\xi \in \mathcal{D}} [f(x; \xi)] \}$$



Server

Homogenous Data Setting



$\mathcal{D}$

$$f(x) := \mathbb{E}_{\xi \sim \mathcal{D}} [f(x; \xi)]$$

$$\nabla f(x; \xi)$$



$\mathcal{D}$

$$f(x) := \mathbb{E}_{\xi \sim \mathcal{D}} [f(x; \xi)]$$

$$\nabla f(x; \xi)$$



$\mathcal{D}$

$$f(x) := \mathbb{E}_{\xi \sim \mathcal{D}} [f(x; \xi)]$$

$$\nabla f(x; \xi)$$

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \{ f(x) := \mathbb{E}_{\xi \in \mathcal{D}} [f(x; \xi)] \}$$



Server

Ringmaster ASGD

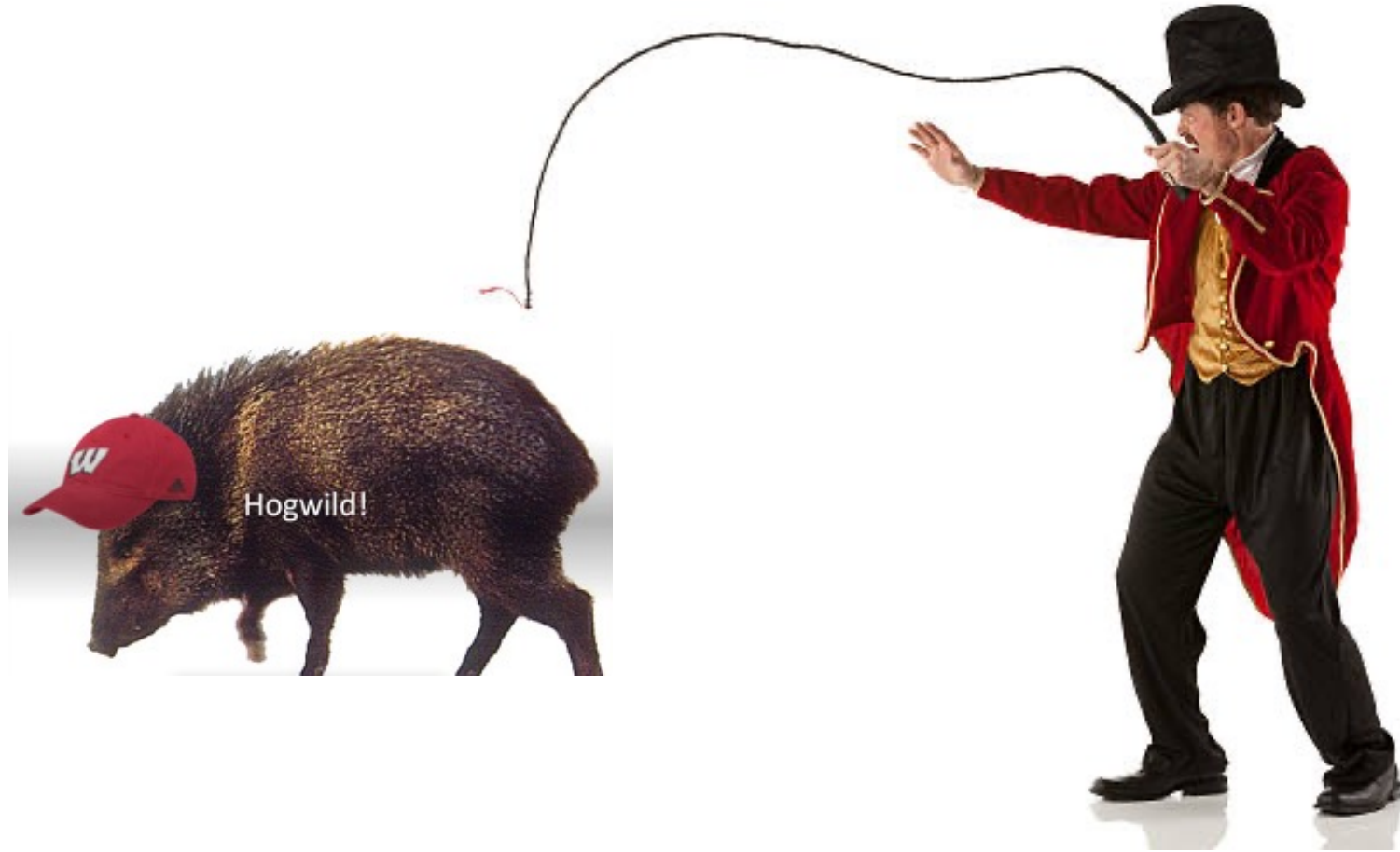
Artavazd Maranjyan, Alexander Tyurin, and Peter Richtárik.

**Ringmaster ASGD: The first Asynchronous SGD
with optimal time complexity**

International Conference on Machine Learning, 2025



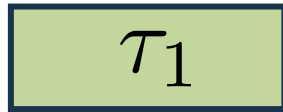
Asynchronous SGD is too wild:
Ringmaster ASGD *tames* it



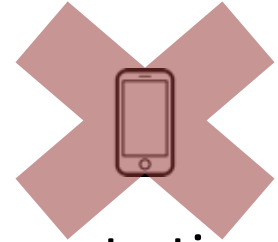
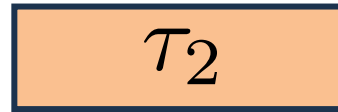
Naive approach: Remove slow workers



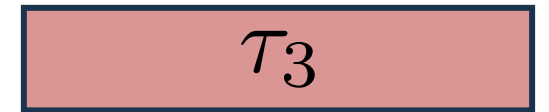
Compute time = τ_1



Compute time = τ_2



Compute time = τ_3



Server

Ringmaster ASGD:

Have a threshold on delays



If: $\delta^k < R$

$$x^{k+1} = x^k - \gamma \nabla f \left(x^{k-\delta^k}; \xi_i^{k-\delta^k} \right)$$

Else: Ignore the gradient and send the current point x^k to the worker



Server

$$\nabla f \left(x^{k-\delta^k}; \xi_i^{k-\delta^k} \right)$$



Ringmaster ASGD outperforms existing baselines

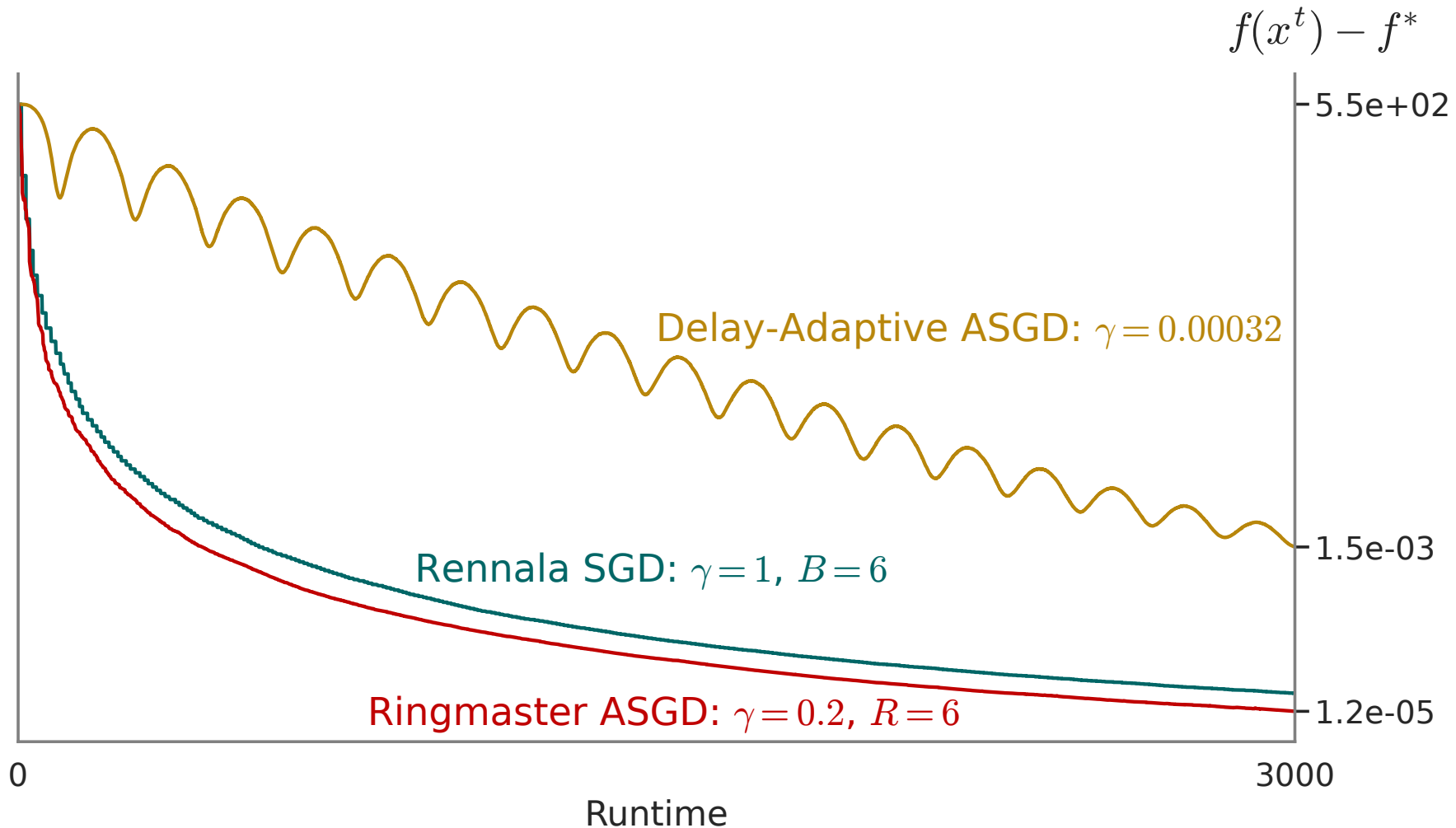
$$f(x) = \frac{1}{2}x^\top Ax - b^\top x$$

$$d = 1729$$

$$n = 6174$$

$$\tau_i = i + |\eta_i|$$

$$\eta_i \sim \mathcal{N}(0, i)$$





Heterogenous Data Setting

Heterogenous Data Setting



$\mathcal{D}_1$

$$f_1(x) := \mathbb{E}_{\xi_1 \sim \mathcal{D}_1} [f_1(x; \xi_1)]$$



$\mathcal{D}_2$

$$f_2(x) := \mathbb{E}_{\xi_2 \sim \mathcal{D}_2} [f_2(x; \xi_2)]$$



$\mathcal{D}_3$

$$f_3(x) := \mathbb{E}_{\xi_3 \sim \mathcal{D}_3} [f_3(x; \xi_3)]$$

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \left\{ f(x) := \frac{1}{n} \sum_{i=1}^n f_i(x) \right\}$$



Server

Can not Ignore Gradients

Delay of worker i^k at iteration k

$$x^{k+1} = x^k - \gamma \nabla f_{i^k} \left(x^{k-\delta_{i^k}^k}; \xi^{k-\delta_{i^k}^k} \right)$$

Uses a gradient from a single worker

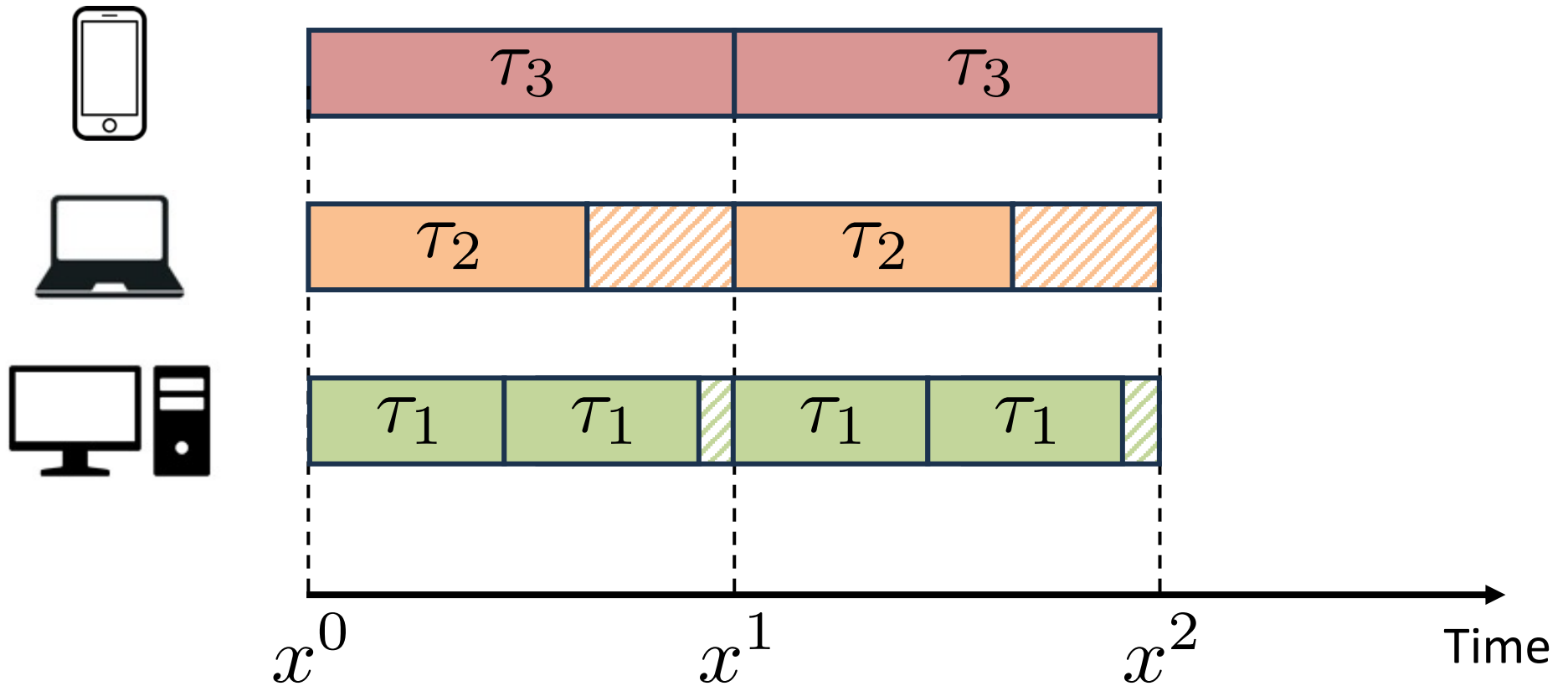
Similarity assumption

$$\|\nabla f_i(x) - \nabla f(x)\| \leq \zeta$$



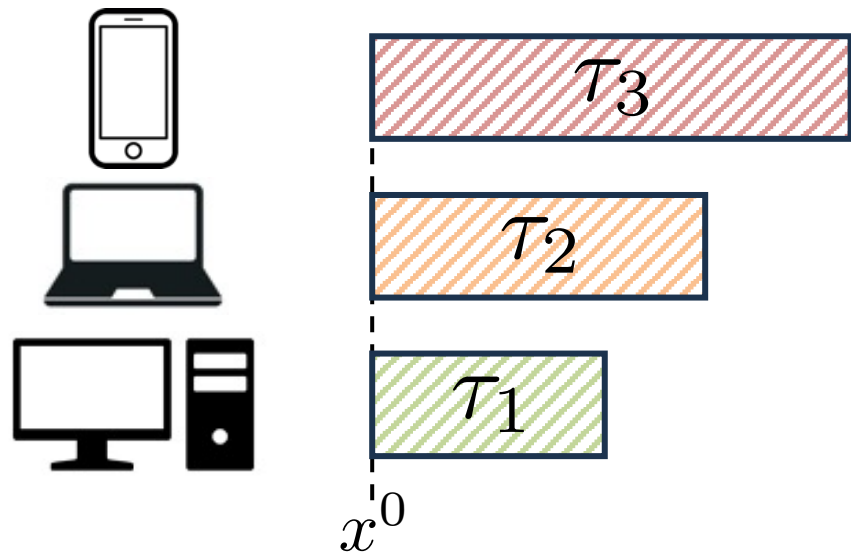
Server

Back to Synchrony: Malenia SGD



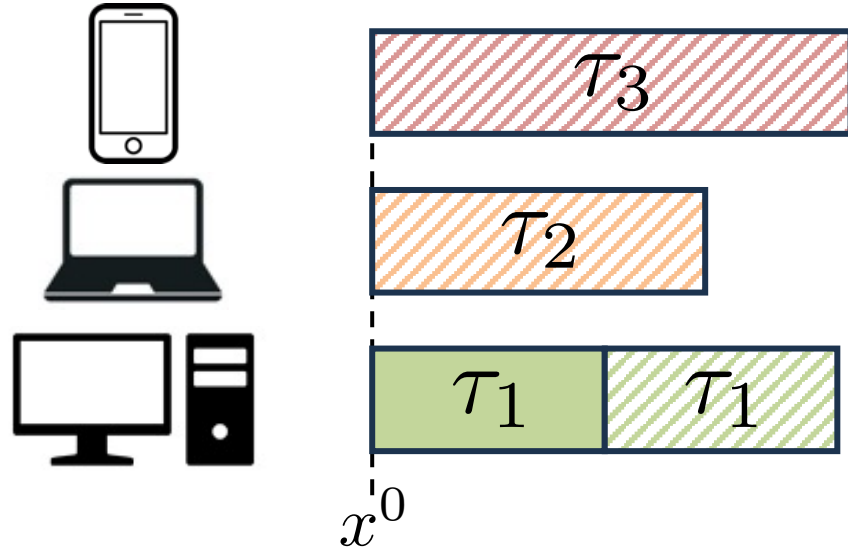
$$x^{k+1} = x^k - \gamma \frac{1}{n} \sum_{i=1}^n \frac{1}{b_i^k} \sum_{j=1}^{b_i^k} \nabla f_i(x^k; \xi_i^{k,j})$$

Malenia SGD with Gradient Table



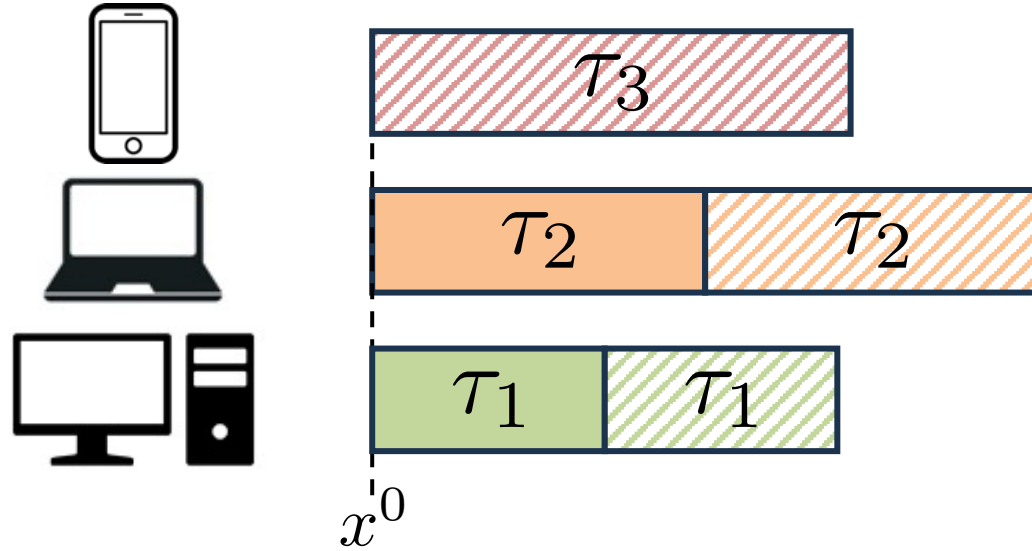
$$b \quad 0 \quad 0 \quad 0$$
$$\left[\begin{array}{c} \\ \\ \\ \end{array} \right]$$

Malenia SGD with Gradient Table



$$b \quad \begin{matrix} 1 & 0 & 0 \\ \left[\begin{array}{c} G_1 \end{array} \right] \end{matrix}$$
$$G_1 = \nabla f_1 \left(x^0; \xi_1^{0,1} \right)$$

Malenia SGD with Gradient Table

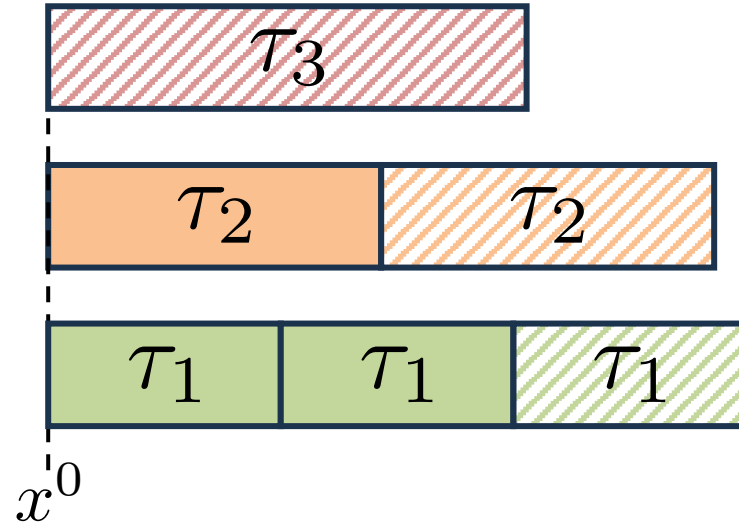


b 1 1 0

$$\begin{bmatrix} \text{green line} & \text{orange line} \\ G_1 & G_2 \end{bmatrix}$$

$$G_2 = \nabla f_2 \left(x^0; \xi_2^{0,1} \right)$$

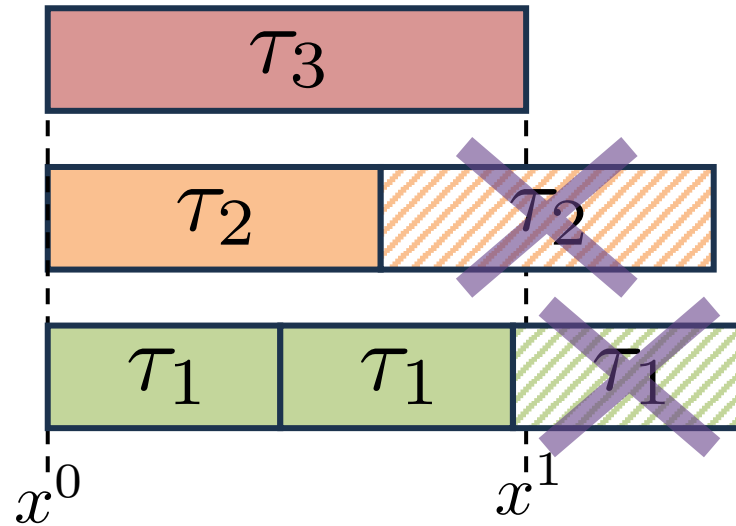
Malenia SGD with Gradient Table



$$\begin{matrix} b & 2 & 1 & 0 \\ \left[\begin{array}{c} \text{green line} \\ G_1 \\ \text{green line} \end{array} & \begin{array}{c} \text{orange line} \\ G_2 \\ \text{orange line} \end{array} \right] \end{matrix}$$

$$G_1 = \sum_{j=1}^2 \nabla f_1 \left(x^0; \xi_1^{0,j} \right)$$

Malenia SGD with Gradient Table



Discard!

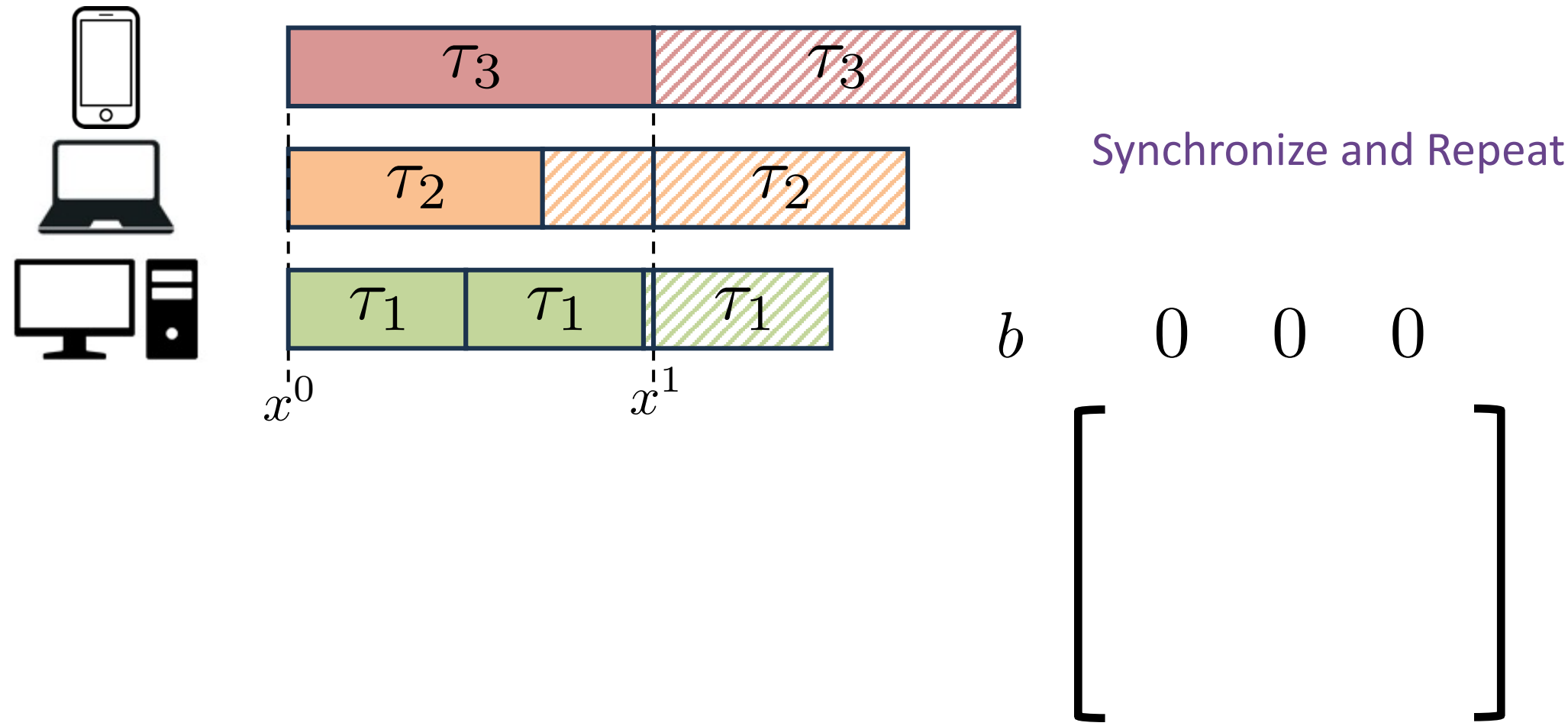
$$x^1 = x^0 - \gamma \frac{1}{n} \sum_{i=1}^n \frac{G_i}{b_i}$$

$$b \quad \begin{matrix} 2 & 1 & 1 \end{matrix}$$

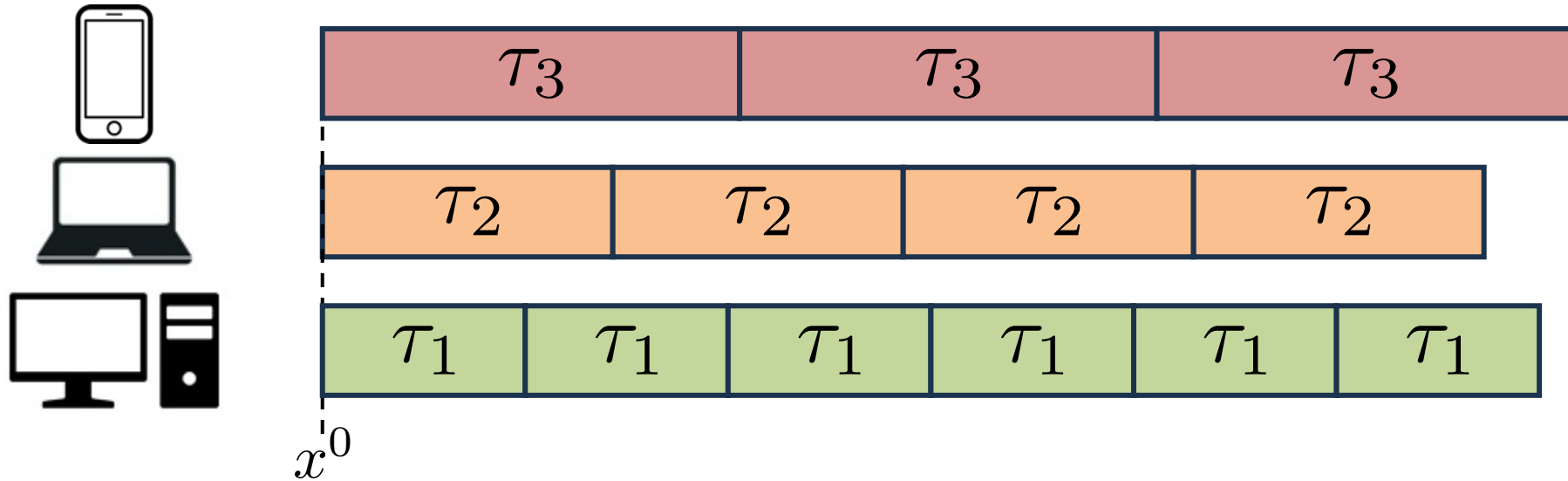
$$\left[\begin{array}{c|c|c} \text{green} & \text{orange} & \text{red} \\ G_1 & G_2 & G_3 \end{array} \right]$$

$$G_3 = \nabla f_3 \left(x^0; \xi_3^{0,1} \right)$$

Malenia SGD with Gradient Table



Can we achieve optimal time complexity with asynchronous methods?



Yes — **Ringleader ASGD** can: Make It Less Wild

Artavazd Maranjyan and Peter Richtárik.

Ringleader ASGD: The first Asynchronous SGD
with optimal time complexity under data heterogeneity.
arXiv:2509.22860, 2025



Yes — Ringleader ASGD can: Make It Less Wild

Phase 1 (collect)

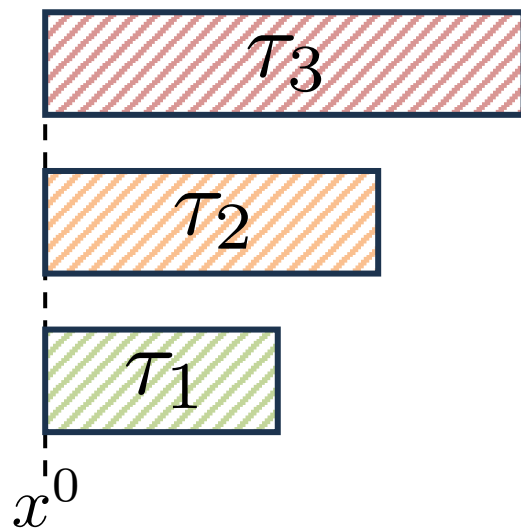
Wait to receive at least one
gradient from each worker

Phase 2 (update)

Perform a single update per worker

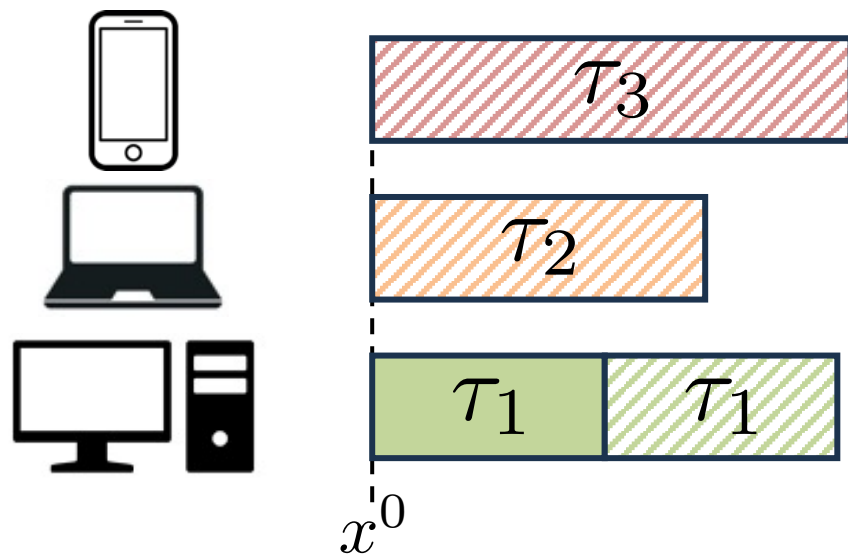


Ringleader ASGD: Phase 1 (collect)



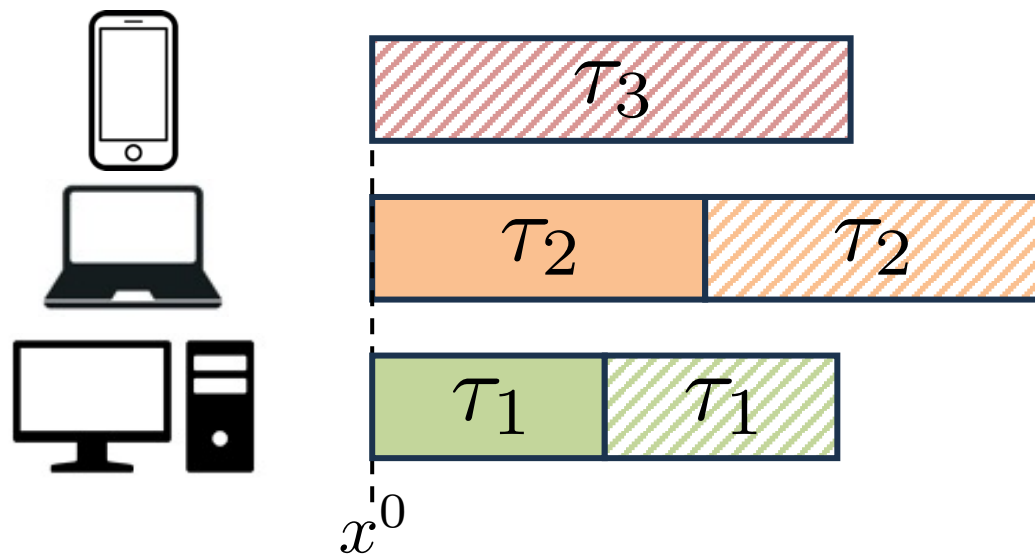
$$\begin{bmatrix} b & 0 & 0 & 0 \end{bmatrix}$$

Ringleader ASGD: Phase 1 (collect)



$$b \quad \begin{bmatrix} 1 & 0 & 0 \\ G_1 \end{bmatrix}$$
$$G_1 = \nabla f_1 \left(x^0; \xi_1^{0,1} \right)$$

Ringleader ASGD: Phase 1 (collect)

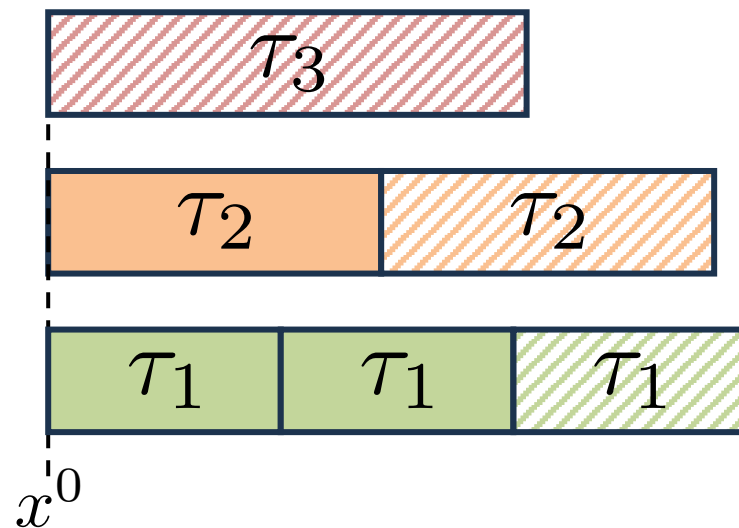


b 1 1 0

$$\begin{bmatrix} \text{green line} & \text{orange line} \\ G_1 & G_2 \end{bmatrix}$$

$$G_2 = \nabla f_2 \left(x^0; \xi_2^{0,1} \right)$$

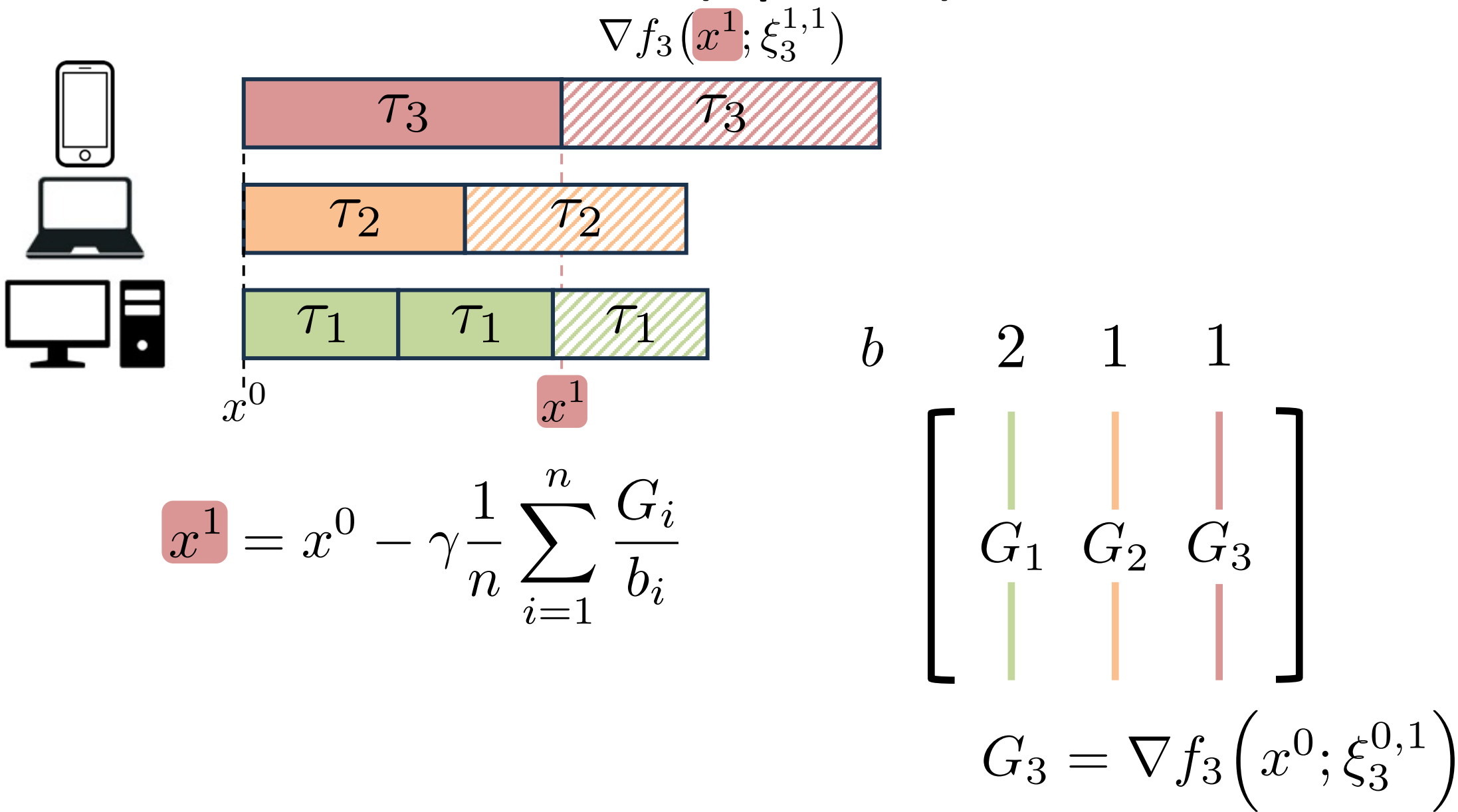
Ringleader ASGD: Phase 1 (collect)



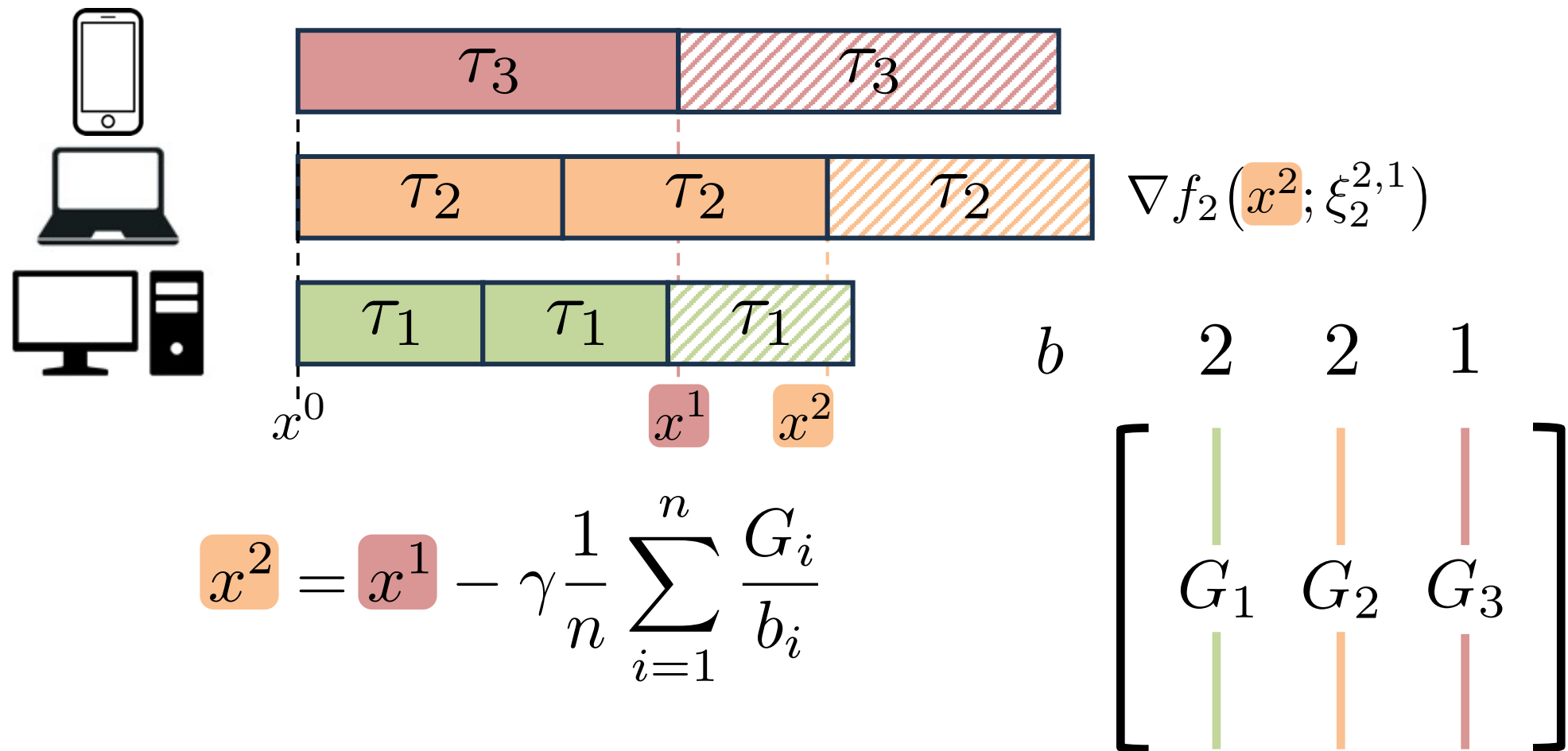
$$\begin{matrix} b & 2 & 1 & 0 \\ \left[\begin{array}{c} \text{green line} \\ G_1 \\ \text{green line} \end{array} & \begin{array}{c} \text{orange line} \\ G_2 \\ \text{orange line} \end{array} & & \right] \end{matrix}$$

$$G_1 = \sum_{j=1}^2 \nabla f_1 \left(x^0; \xi_1^{0,j} \right)$$

Ringleader ASGD: Phase 2 (update)

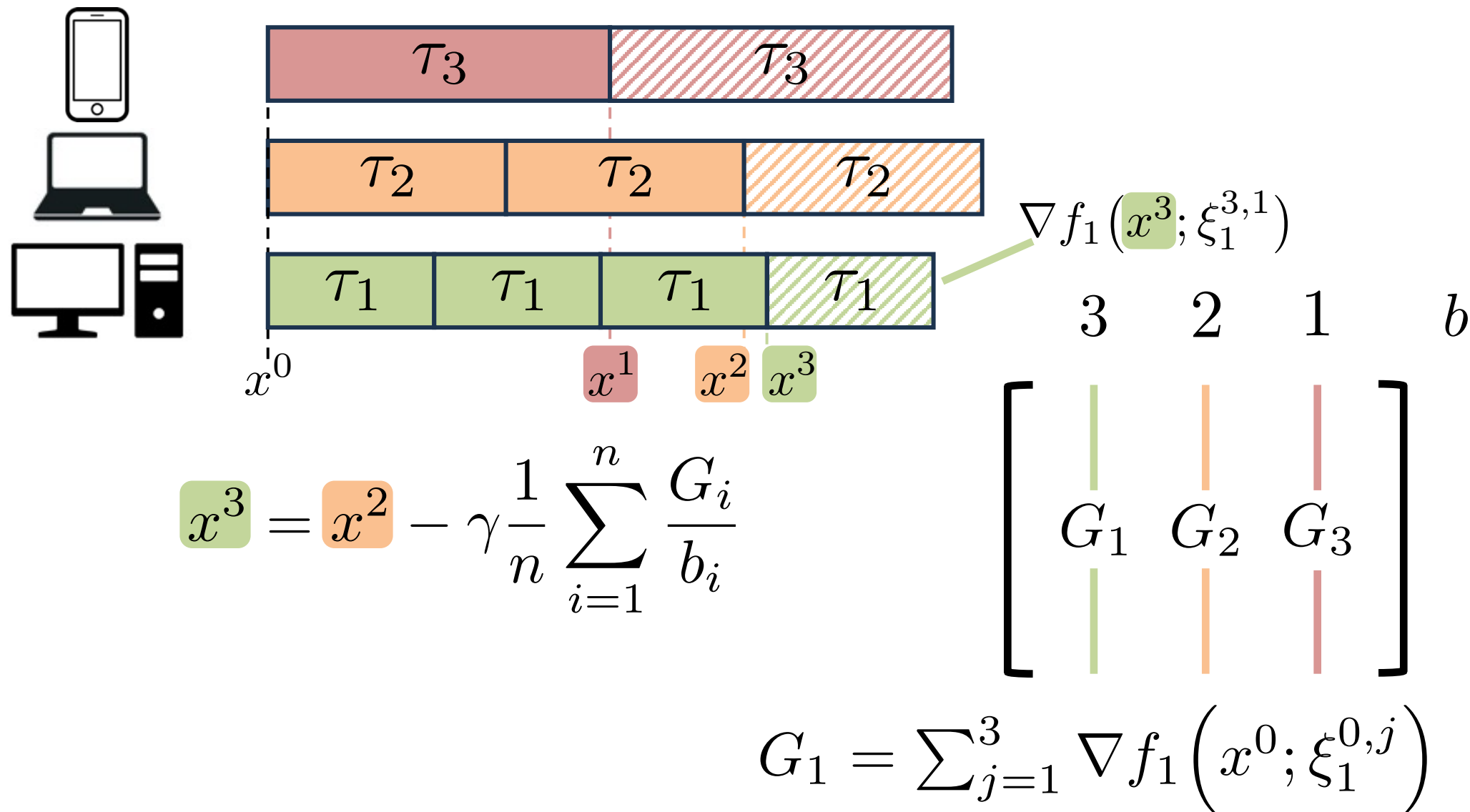


Ringleader ASGD: Phase 2 (update)

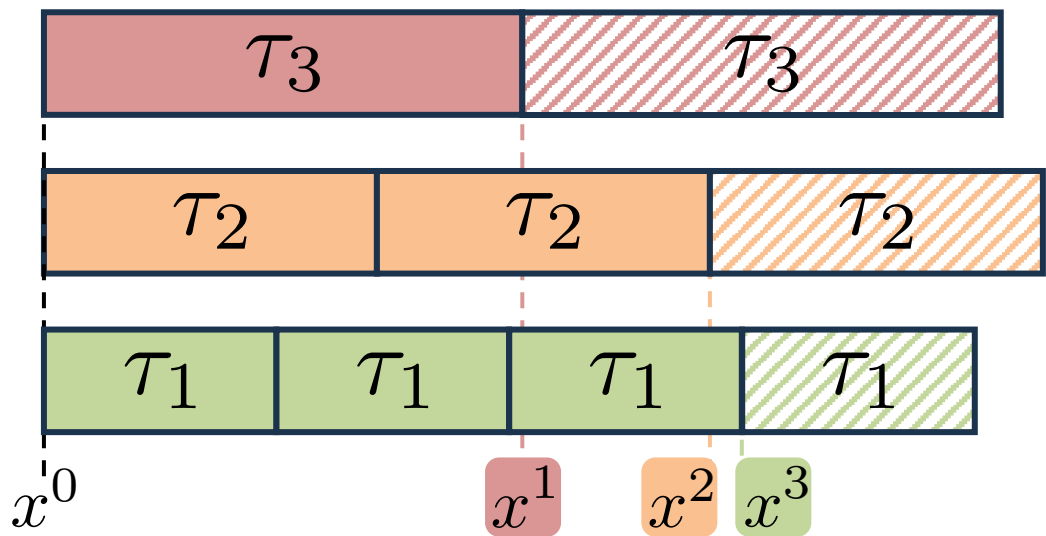


$$G_2 = \sum_{j=1}^2 \nabla f_2 \left(x^0; \xi_2^{0,j} \right)$$

Ringleader ASGD: Phase 2 (update)

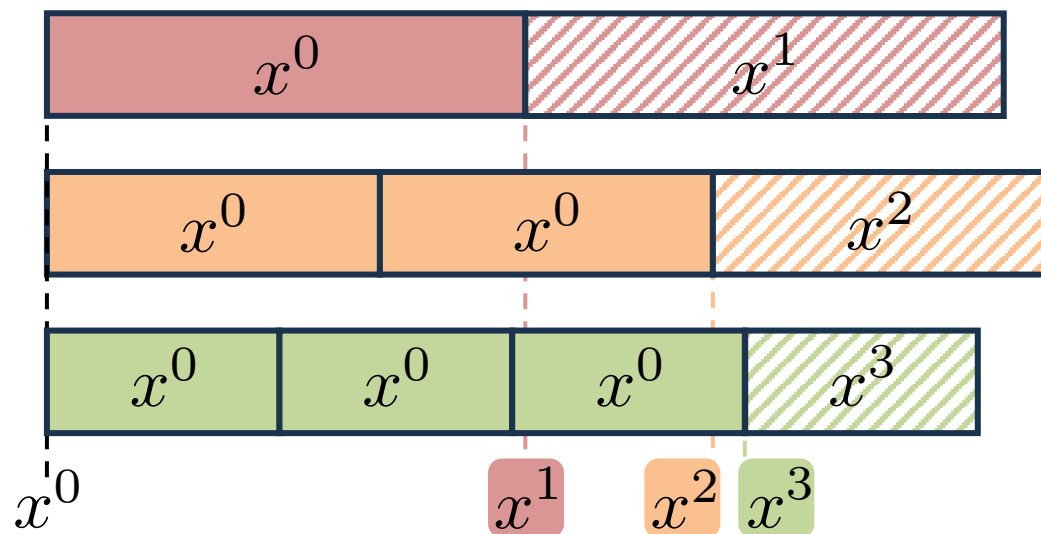


Ringleader ASGD: Phase 1 (collect)



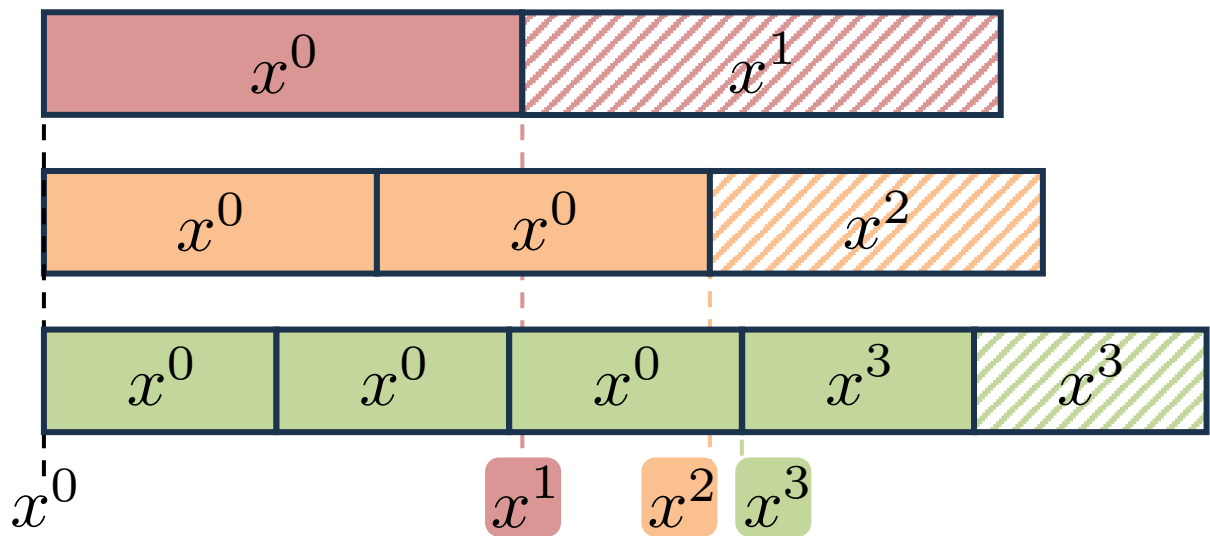
$$\begin{bmatrix} 0 & 0 & 0 & b \end{bmatrix}$$

Ringleader ASGD: Phase 1 (collect)



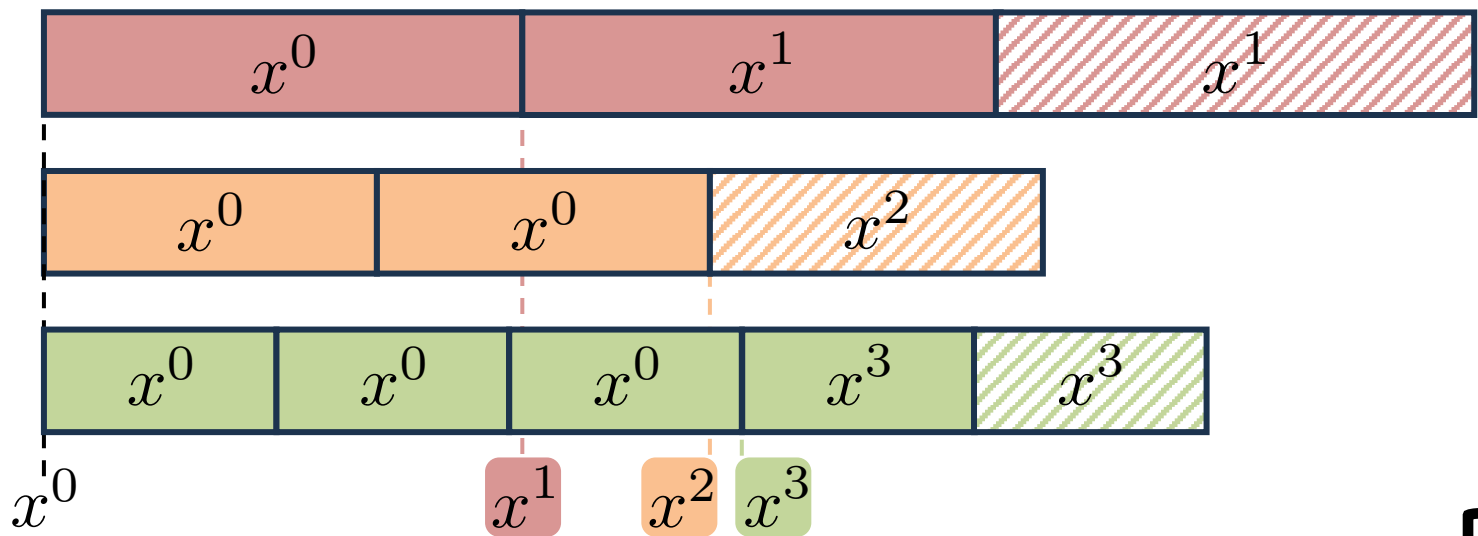
$$\begin{bmatrix} 0 & 0 & 0 & b \end{bmatrix}$$

Ringleader ASGD: Phase 1 (collect)



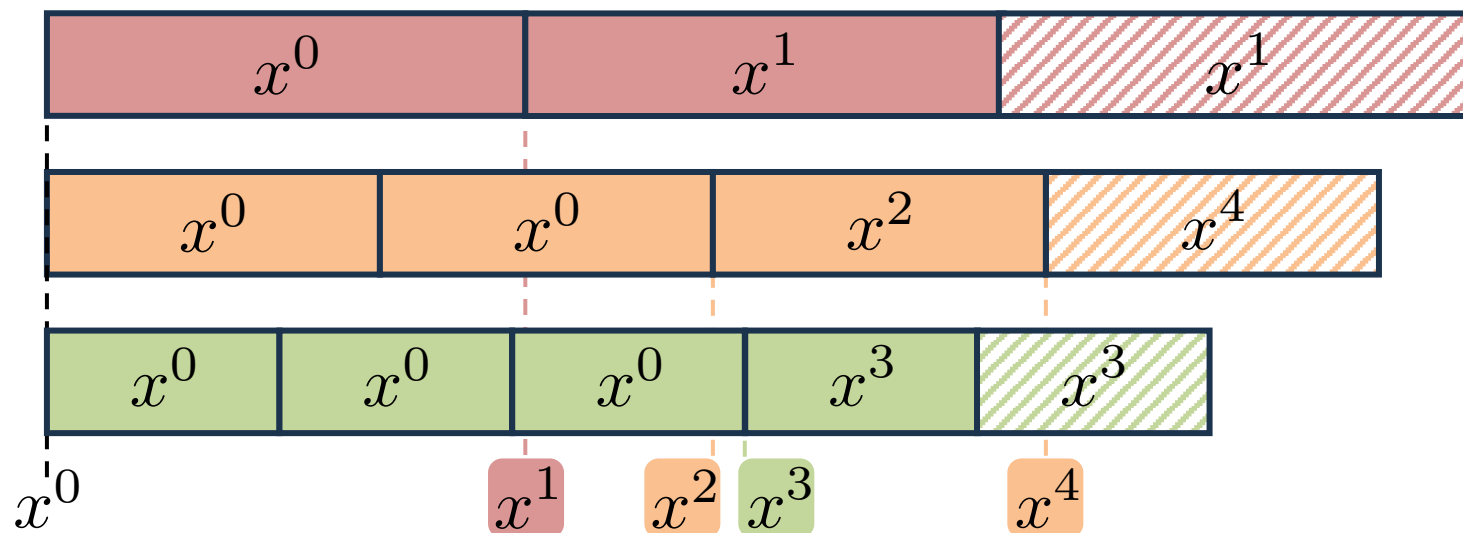
$$\begin{bmatrix} 1 & 0 & 0 & b \\ G_1 \end{bmatrix}$$
$$G_1 = \nabla f_1 \left(x^3; \xi_1^{3,1} \right)$$

Ringleader ASGD: Phase 1 (collect)



$$\begin{bmatrix} 1 & 0 & 1 & b \\ G_1 & & G_3 & \end{bmatrix}$$
$$G_3 = \nabla f_3 \left(x^1; \xi_3^{1,1} \right)$$

Ringleader ASGD: Phase 2 (update)

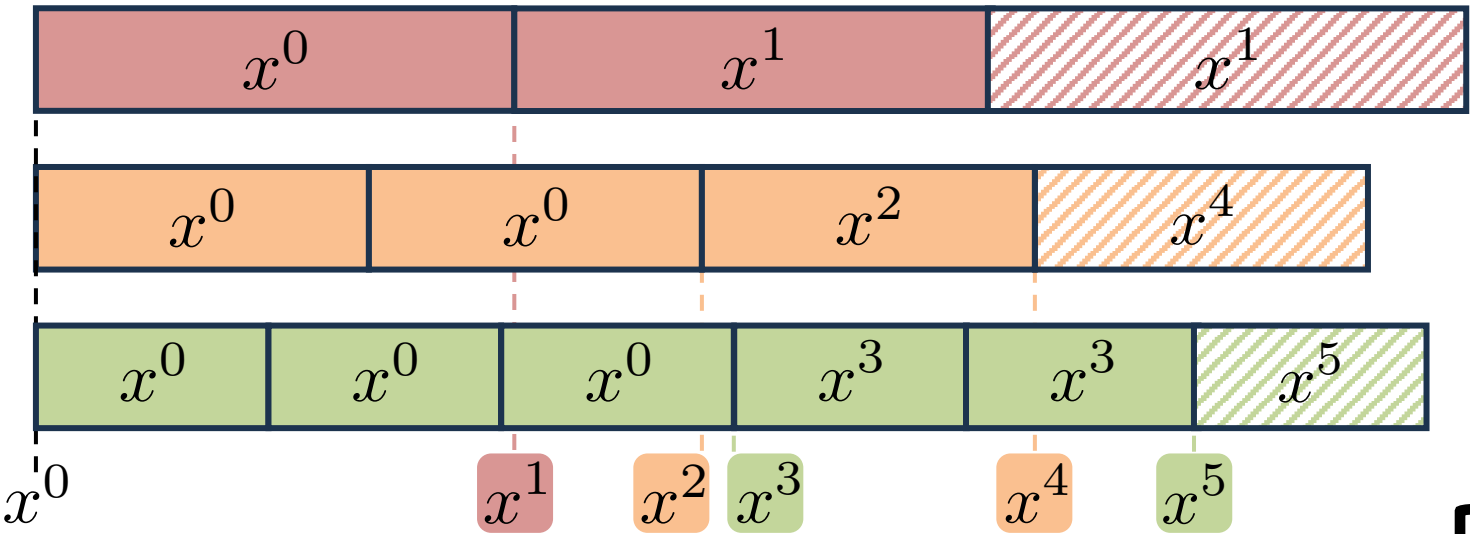


$$x^4 = x^3 - \gamma \frac{1}{n} \sum_{i=1}^n \frac{G_i}{b_i}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ G_1 & G_2 & G_3 \end{bmatrix} \quad b$$

$$G_2 = \nabla f_2 \left(x^2; \xi_2^{2,1} \right)$$

Ringleader ASGD: Phase 2 (update)



$$x^5 = x^4 - \gamma \frac{1}{n} \sum_{i=1}^n \frac{G_i}{b_i}$$

2

1

1

b

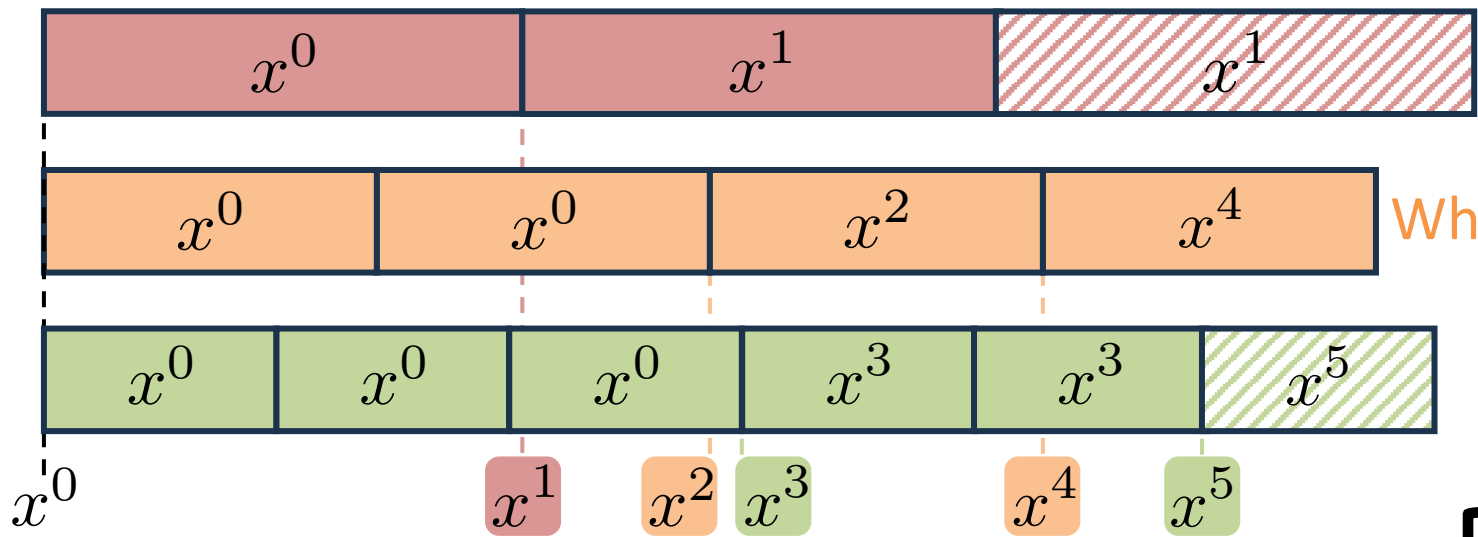
G_1

G_2

G_3

$$G_1 = \sum_{j=1}^2 \nabla f_1 \left(x^3; \xi_1^{3,j} \right)$$

Ringleader ASGD: Phase 2 (update)



What to do with this gradient?

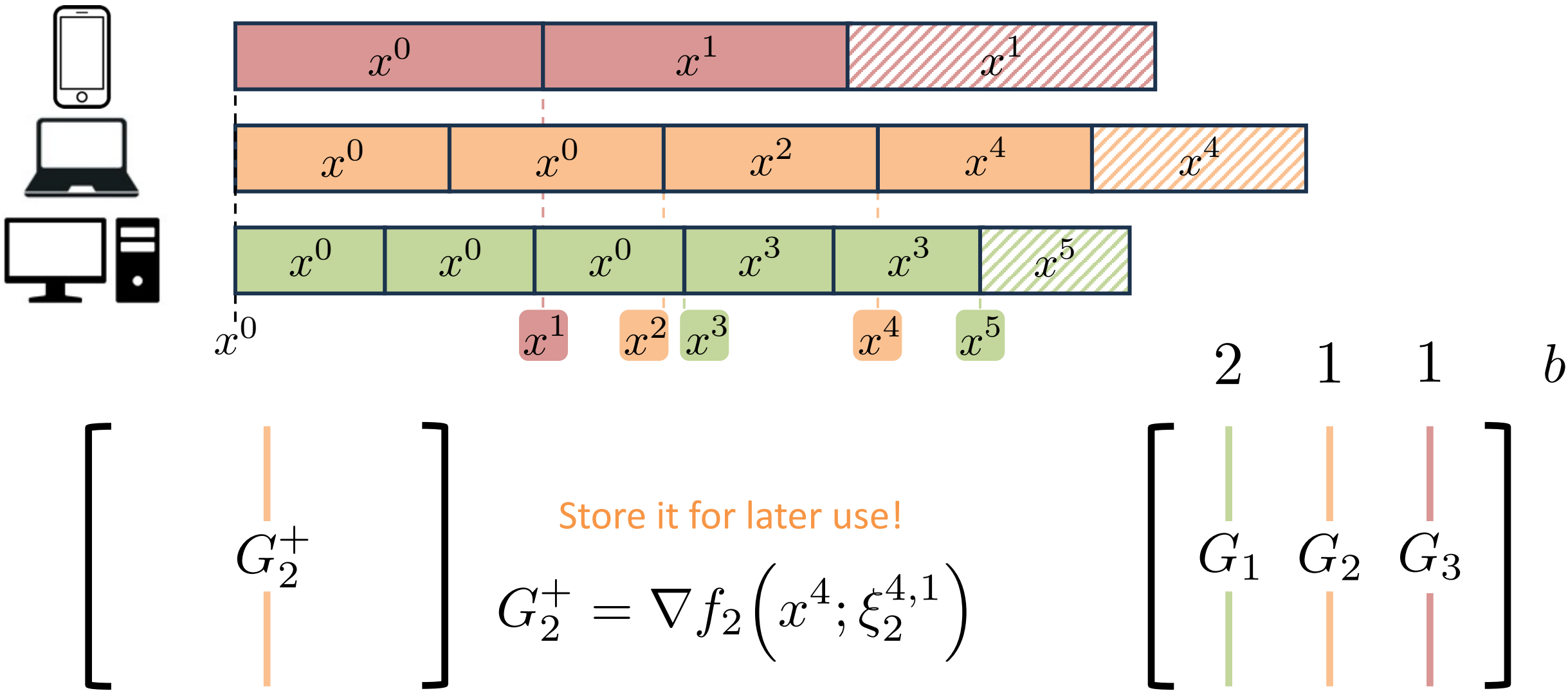
$$\left[\begin{array}{c} \text{orange line} \\ G_2^+ \end{array} \right]$$

Store it for later use!

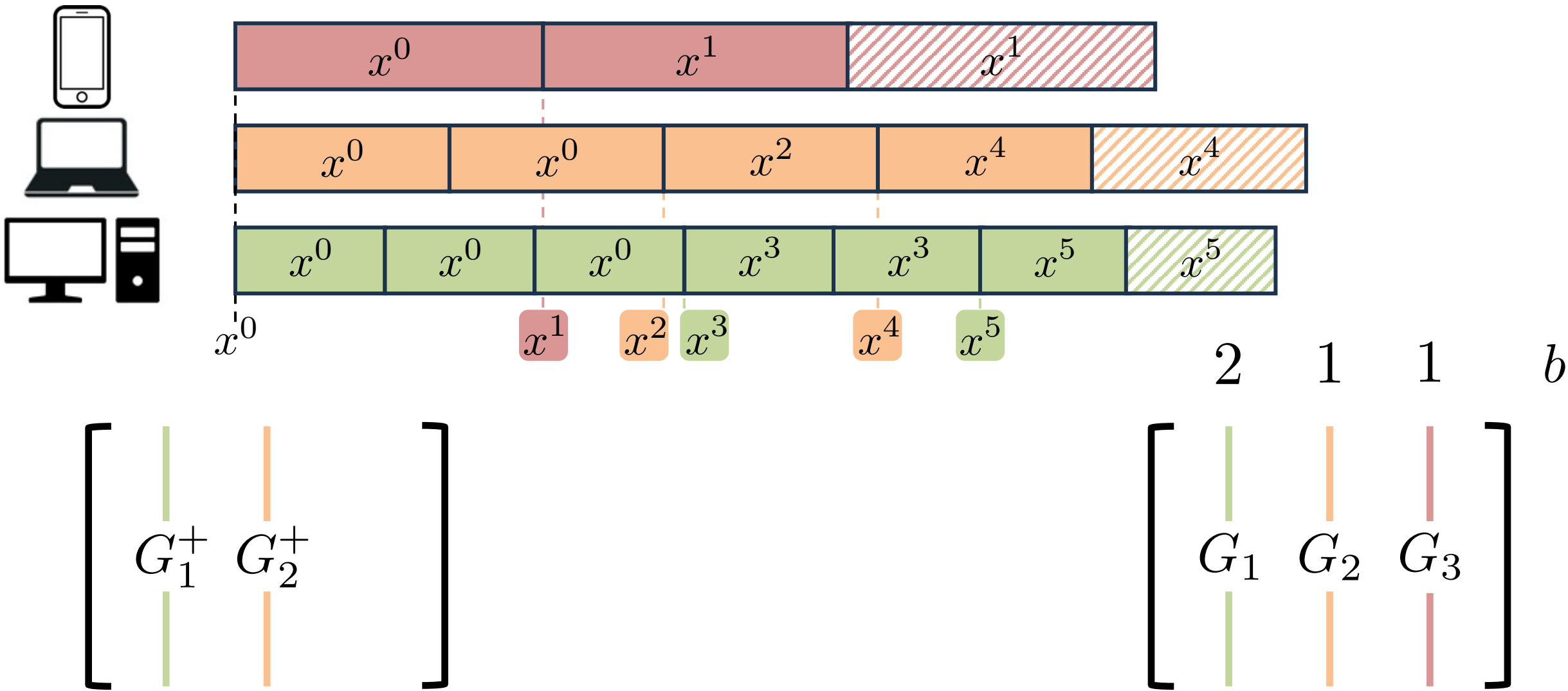
$$G_2^+ = \nabla f_2 \left(x^4; \xi_2^{4,1} \right)$$

$$\left[\begin{array}{ccc} 2 & 1 & 1 & b \\ \text{green line} & \text{orange line} & \text{red line} & \\ G_1 & G_2 & G_3 & \end{array} \right]$$

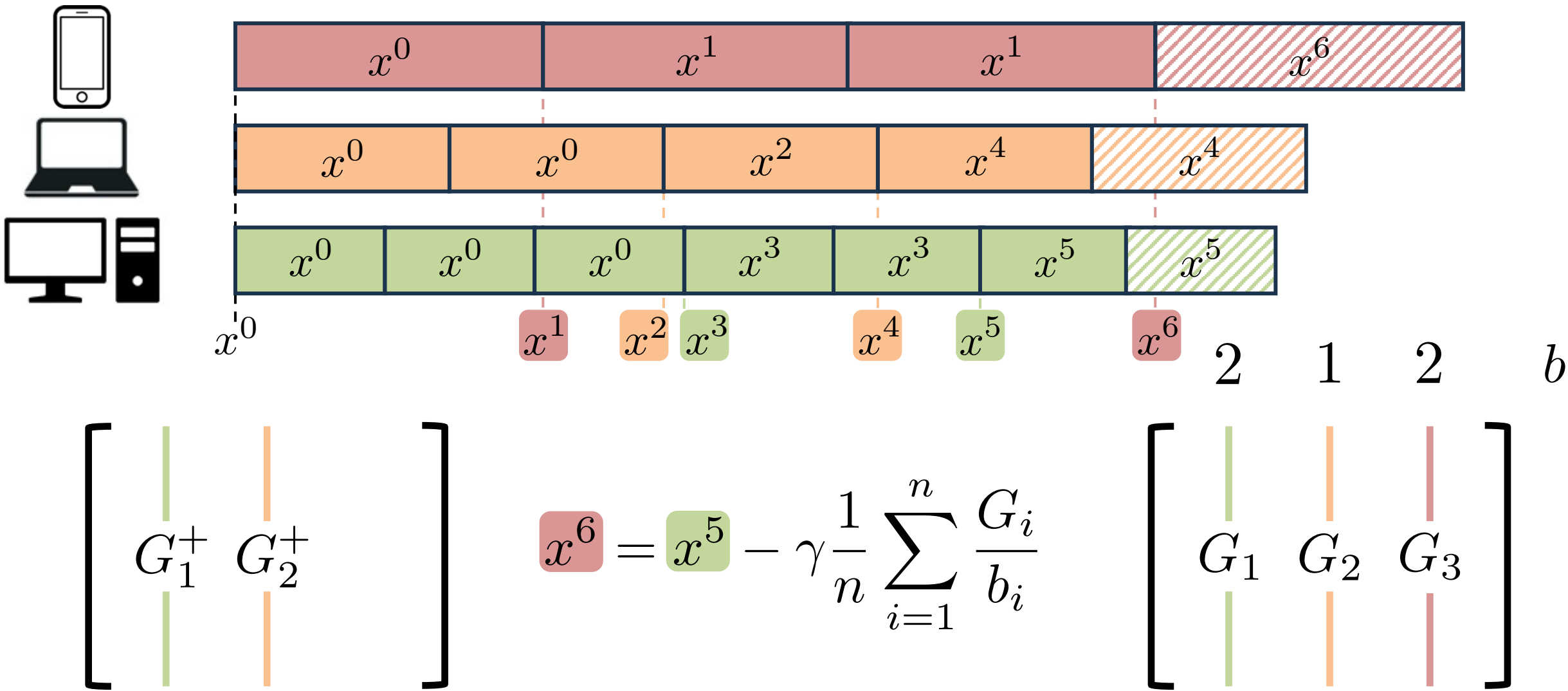
Ringleader ASGD: Phase 2 (update)



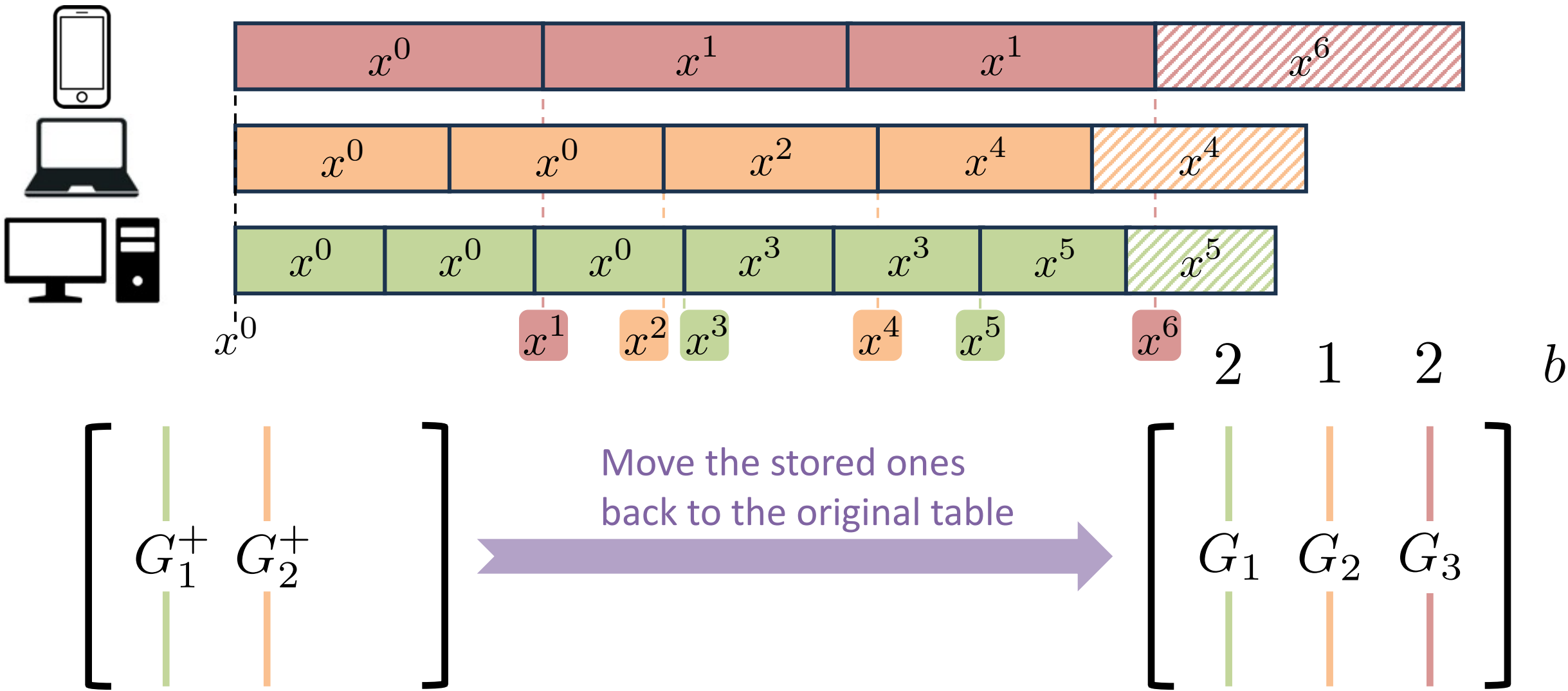
Ringleader ASGD: Phase 2 (update)



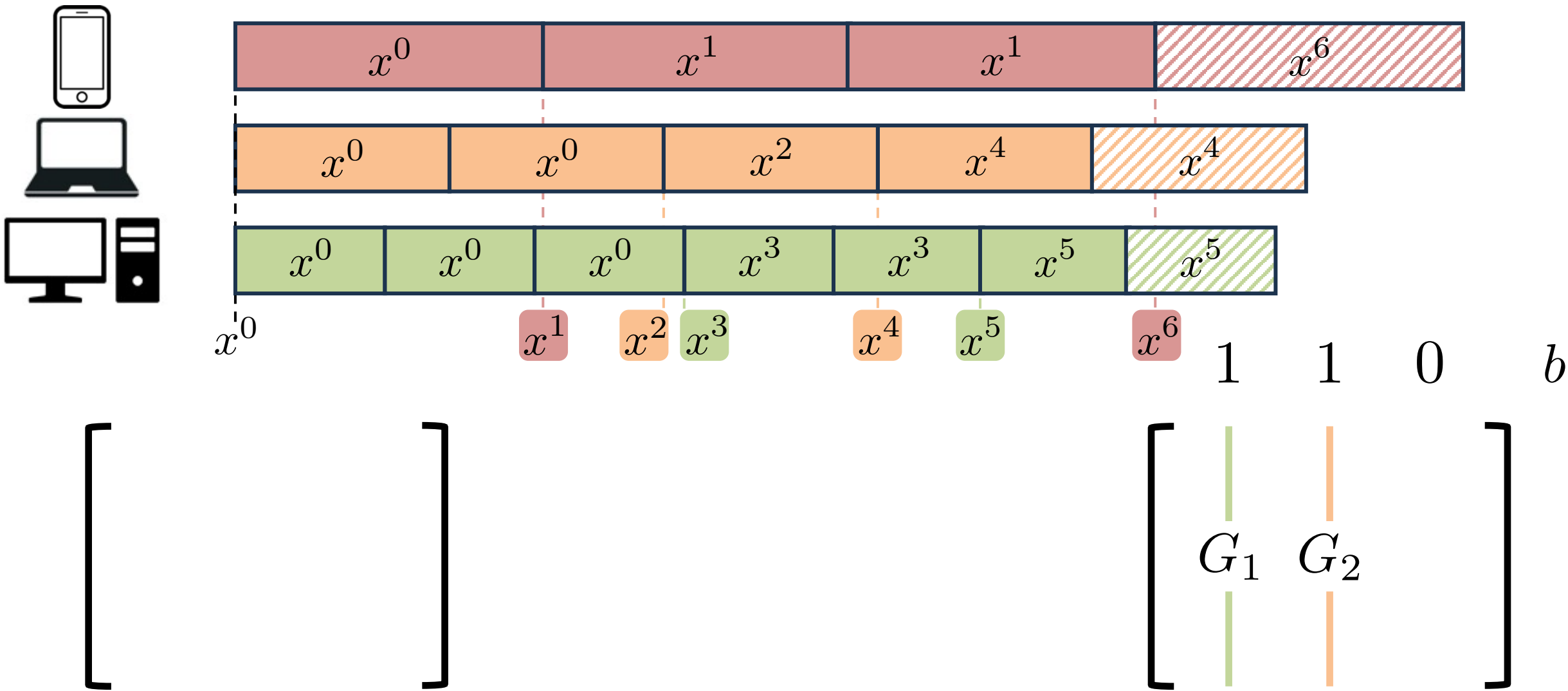
Ringleader ASGD: Phase 2 (update)



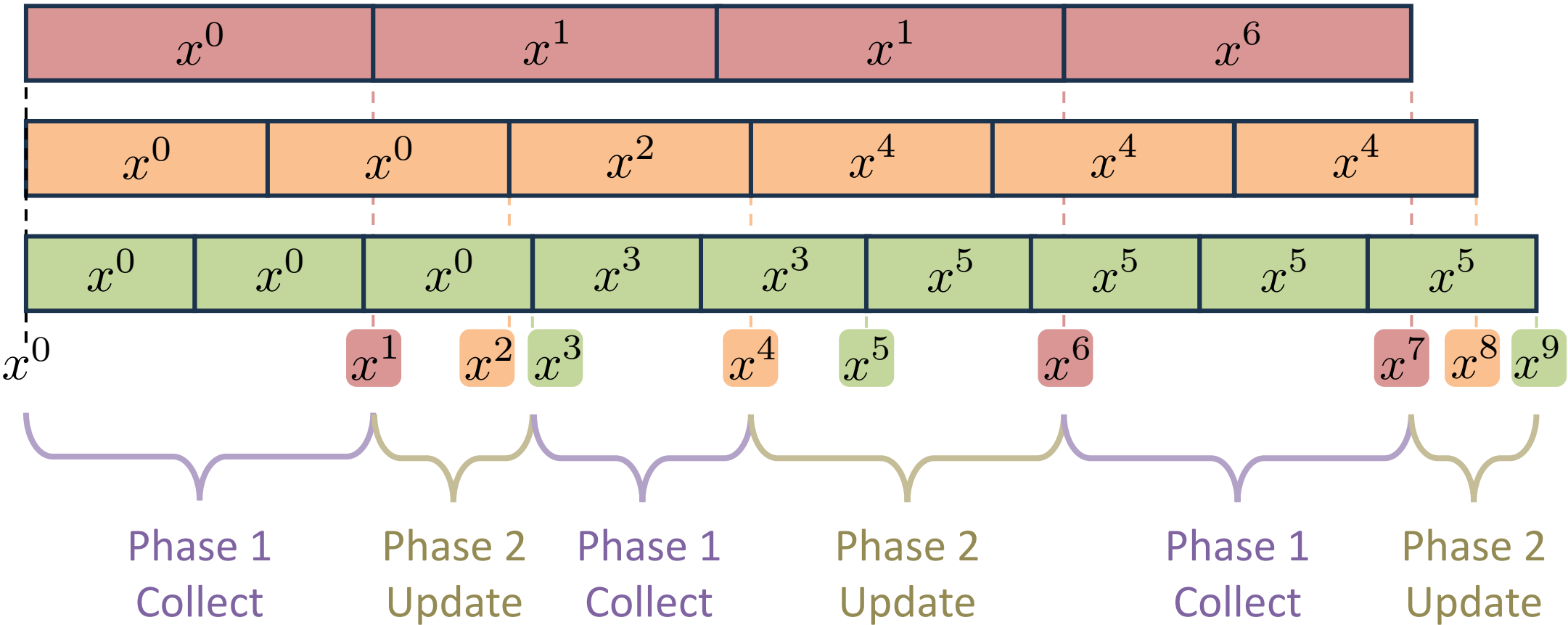
Ringleader ASGD: Prepare for the Next Round



Ringleader ASGD: Start with Phase 1

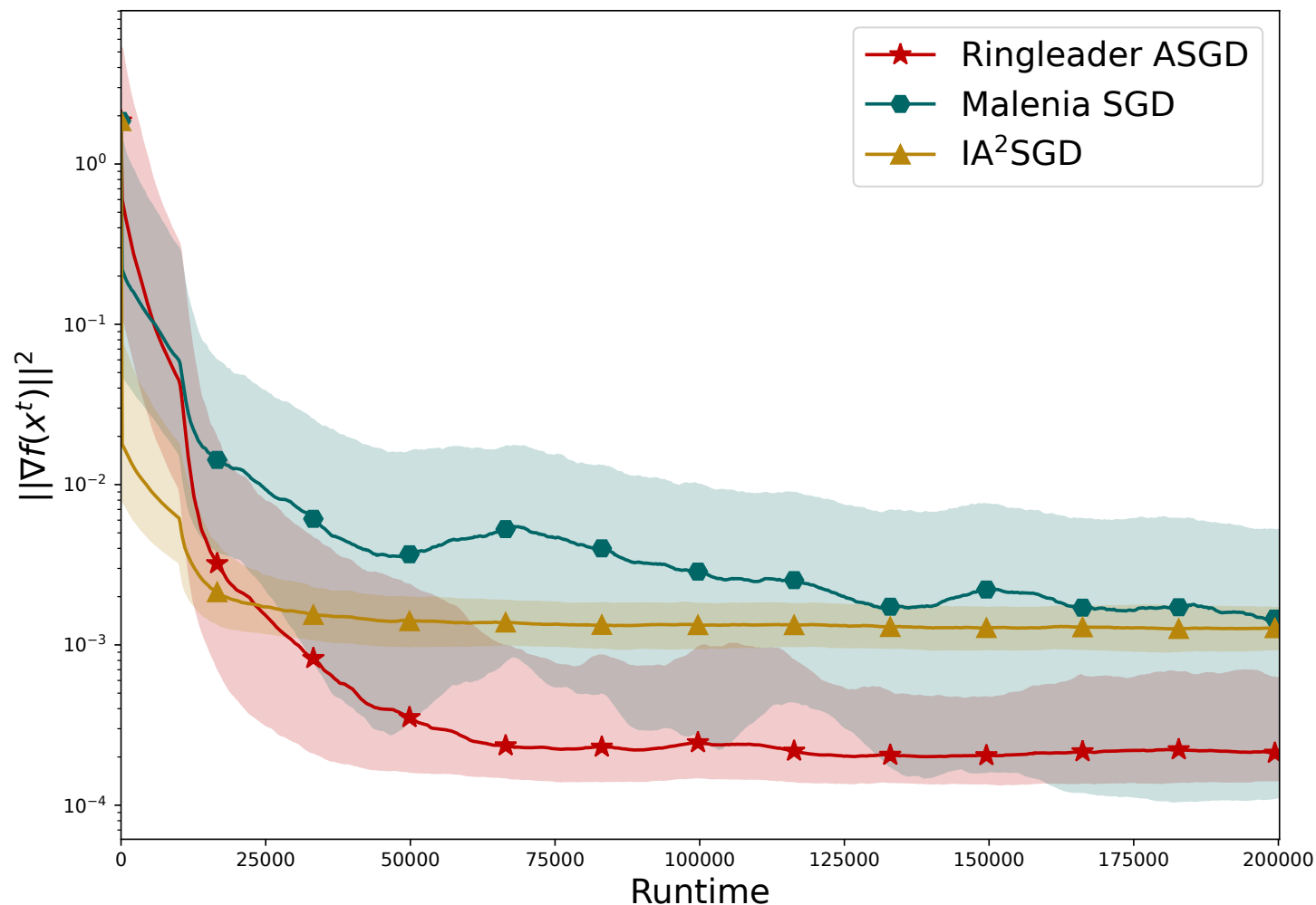


Ringleader ASGD



$$\delta_i^k \leq 2n - 2$$

Ringleader ASGD outperforms existing baselines



Two-layer MLP

MNIST

$n = 100$

$\tau_i = i + |\eta_i|$

$\eta_i \sim \mathcal{N}(0, i)$



KHALAS

More on my PhD Defense
December 4, 8:30 am

