

Ringmaster ASGD: The First Asynchronous SGD with Optimal Time Complexity

Artavazd Maranjyan

AMCS/STAT Graduate Seminar
KAUST, 27 February 2025



جامعة الملك عبد الله
للعلوم والتقنية
King Abdullah University of
Science and Technology



Ringmaster ASGD: The First Asynchronous SGD with Optimal Time Complexity

Problem setup

Optimization objective

Heterogenous system

Method (SGD)

Different ways of parallelizing SGD

Synchronized approaches

Asynchronous SGD

Problems of ASGD

Ringmaster ASGD



The core optimization problem in Machine Learning (and beyond)

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

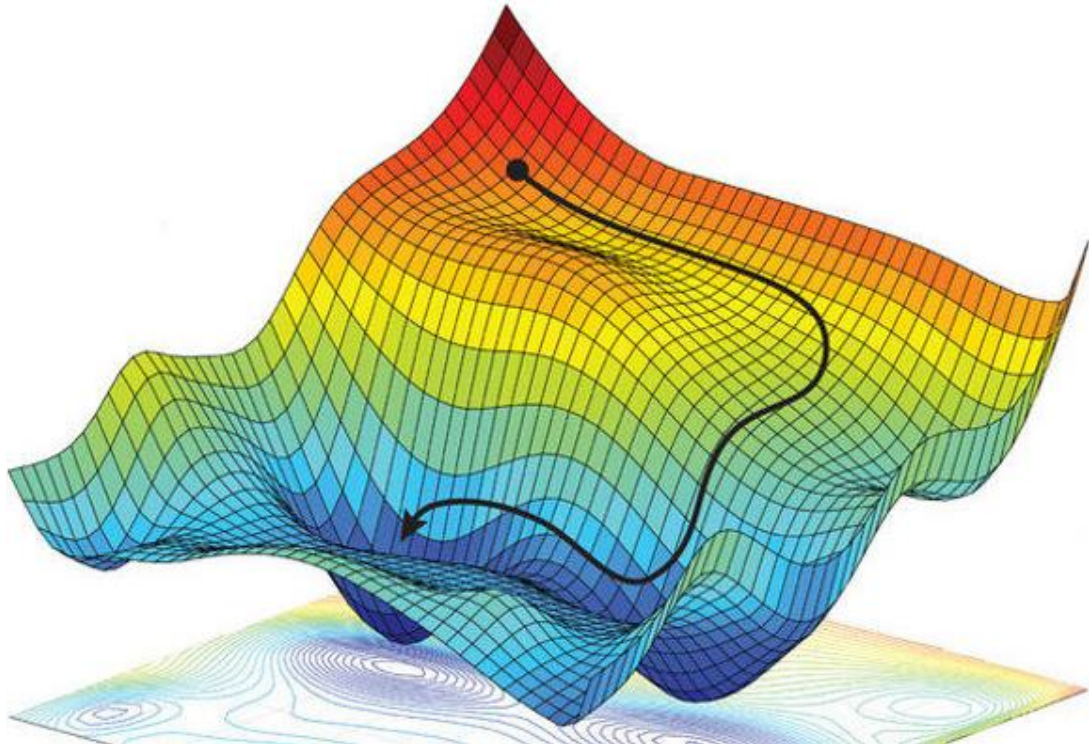
Loss of a data sample ξ

The distribution of the training dataset

$$\mathcal{D} = \text{Uniform}([m])$$

$$\frac{1}{m} \sum_{i=1}^m f(x; \xi_i)$$

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, e.g.,

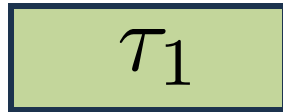
$$\begin{aligned} & \nabla f(x^k; \xi^k) \\ & \frac{1}{B} \sum_{i=1}^B \nabla f(x^k; \xi_i^k) \end{aligned}$$

How to parallelize SGD in heterogeneous systems?



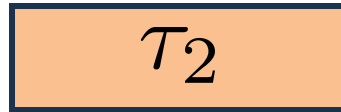
$$\nabla f(x; \xi)$$

Compute time = τ_1



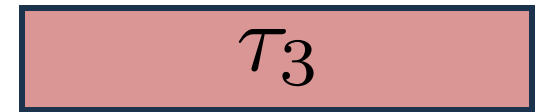
$$\nabla f(x; \xi)$$

Compute time = τ_2



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

Compute time = τ_3



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

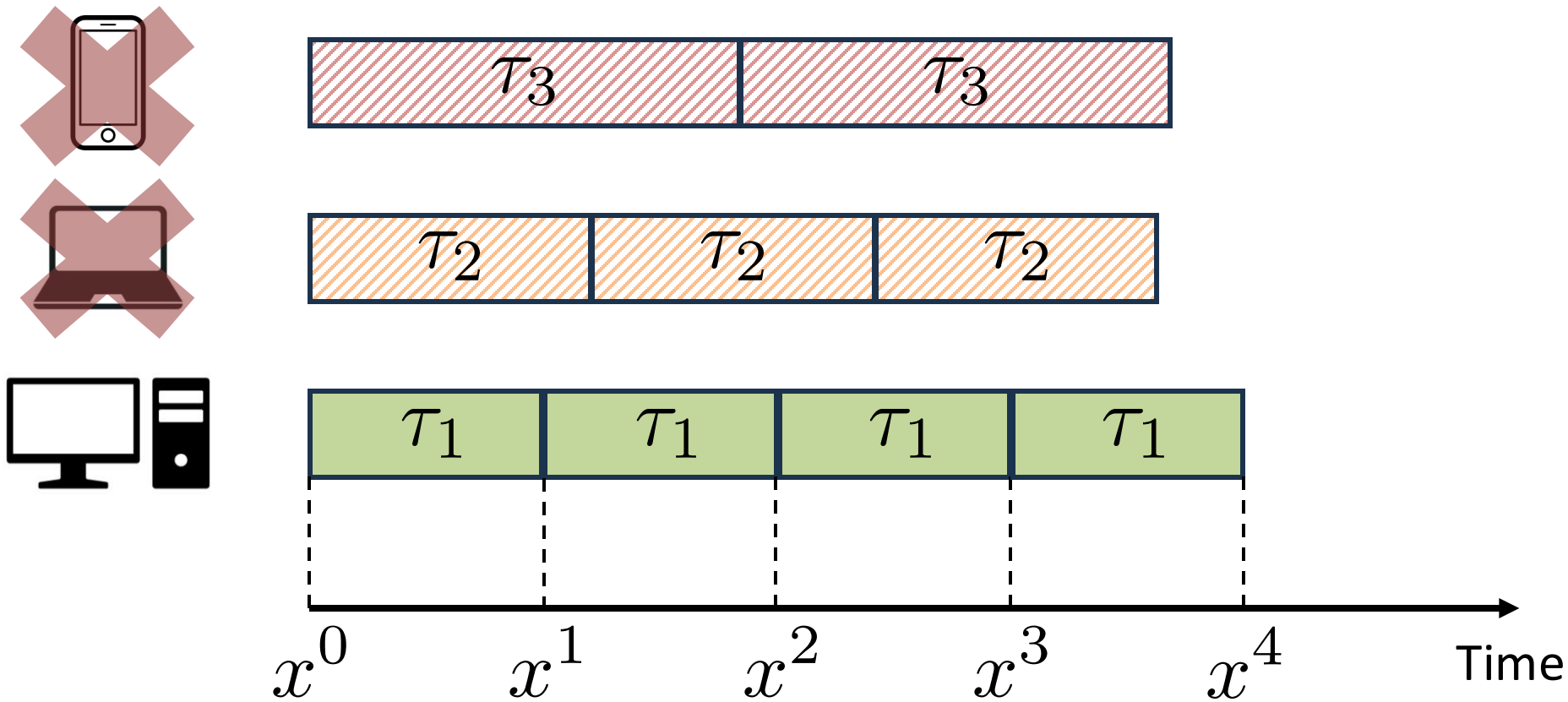


Server

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

How to construct?

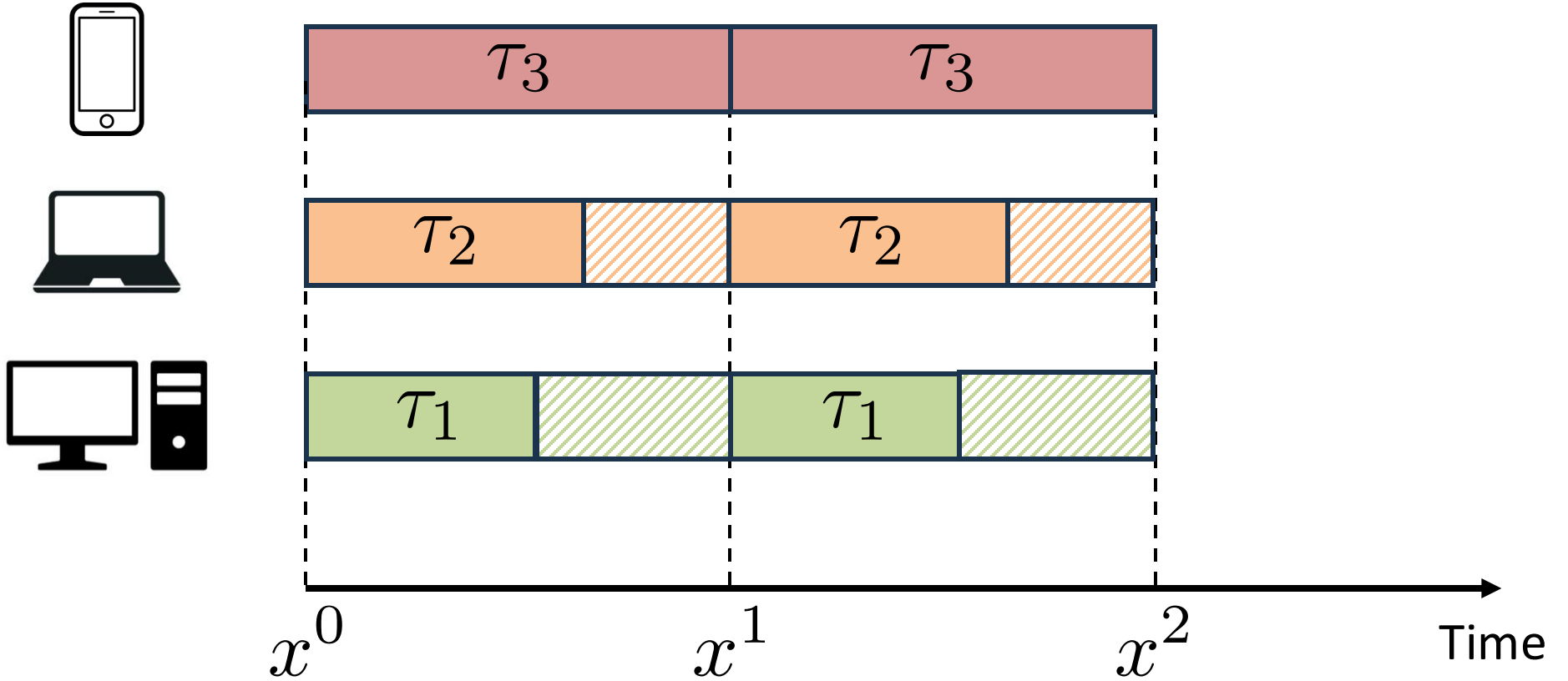
Hero SGD: The fastest worker does it all



$$x^{k+1} = x^k - \gamma \nabla f(x^k; \xi^k)$$

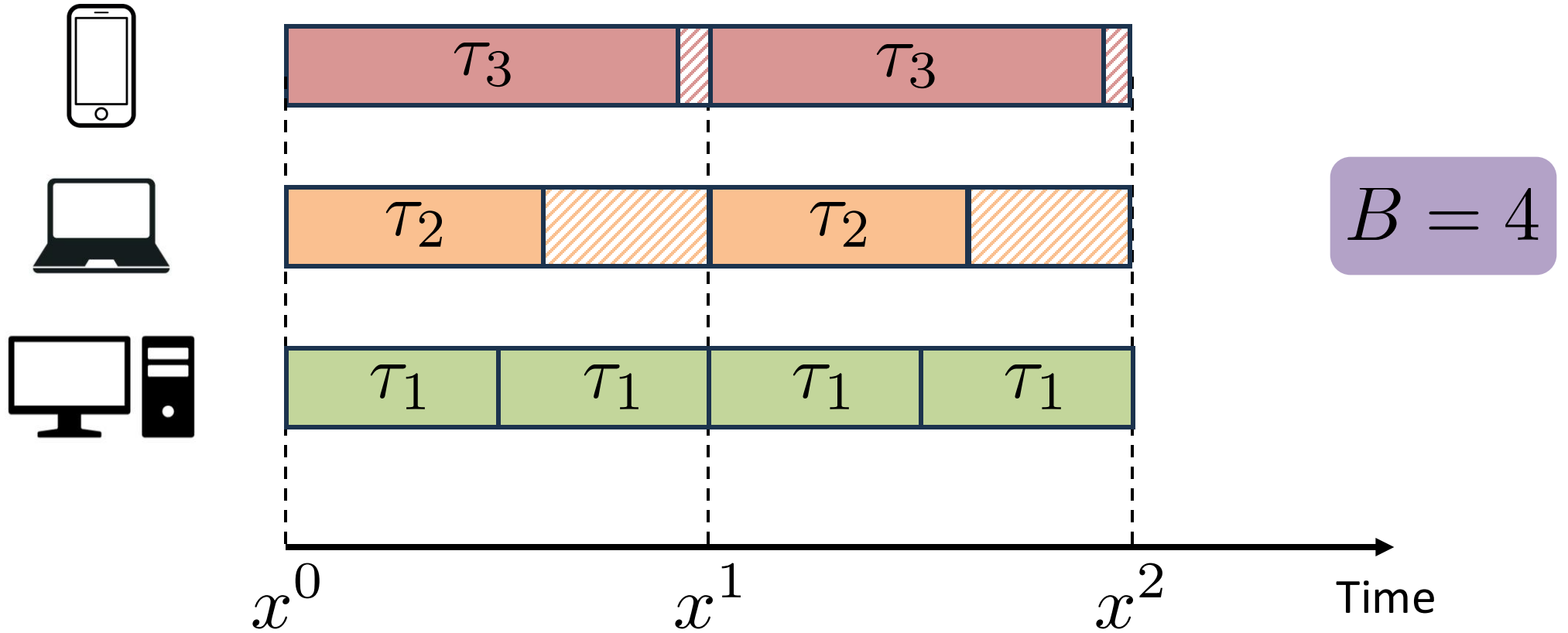
Minibatch SGD: Each worker does one job only

$n = 3$



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

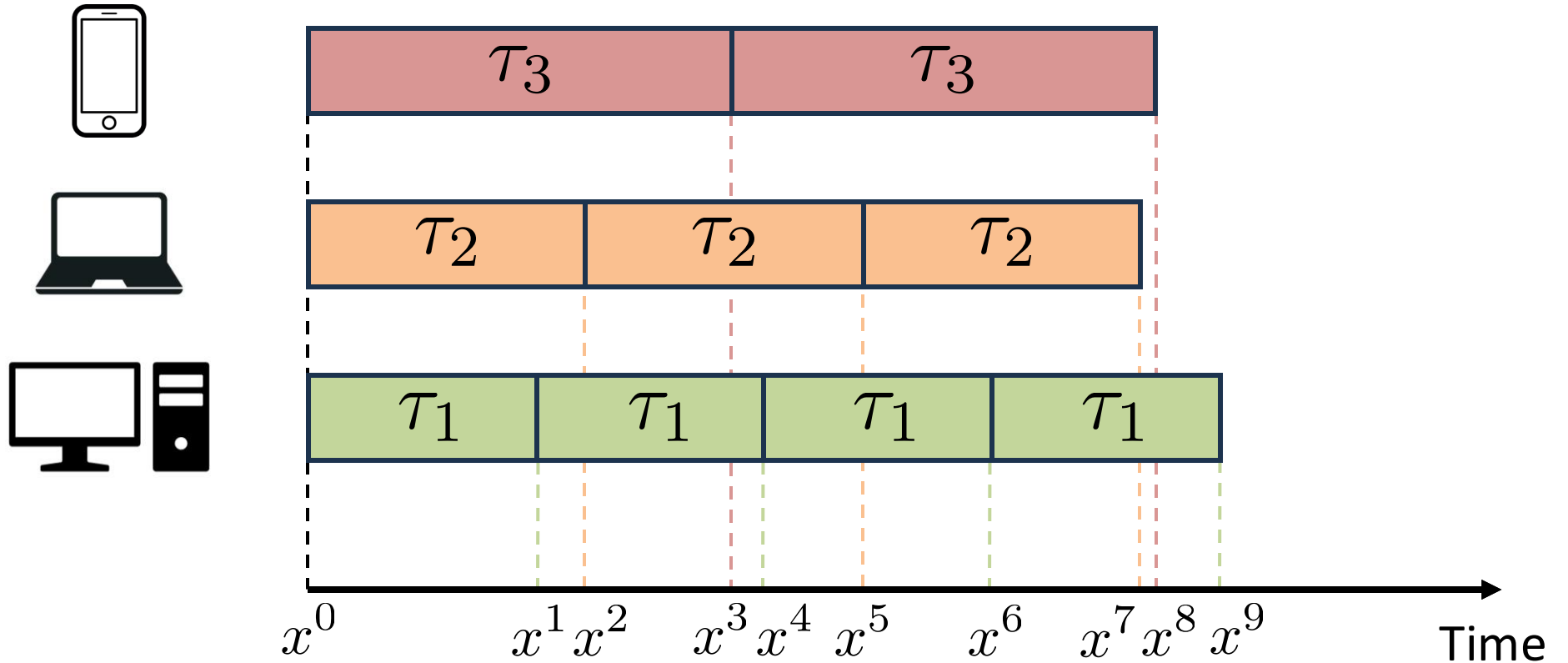
Rennala SGD: Asynchronous batch collection



$$x^{k+1} = x^k - \gamma \frac{1}{B} \sum_{j=1}^B \nabla f(x^k; \xi_j^k)$$

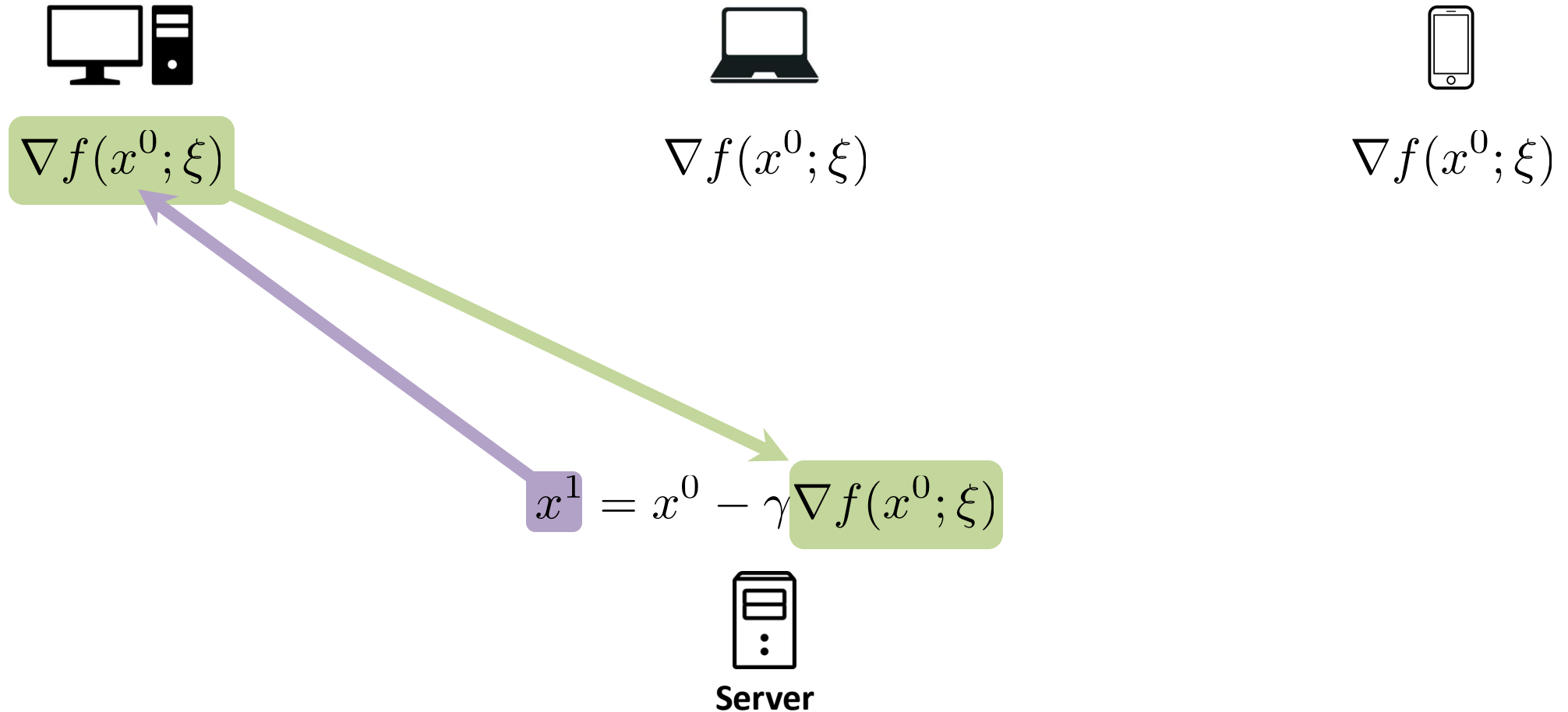
Asynchronous SGD

Remove the synchronization

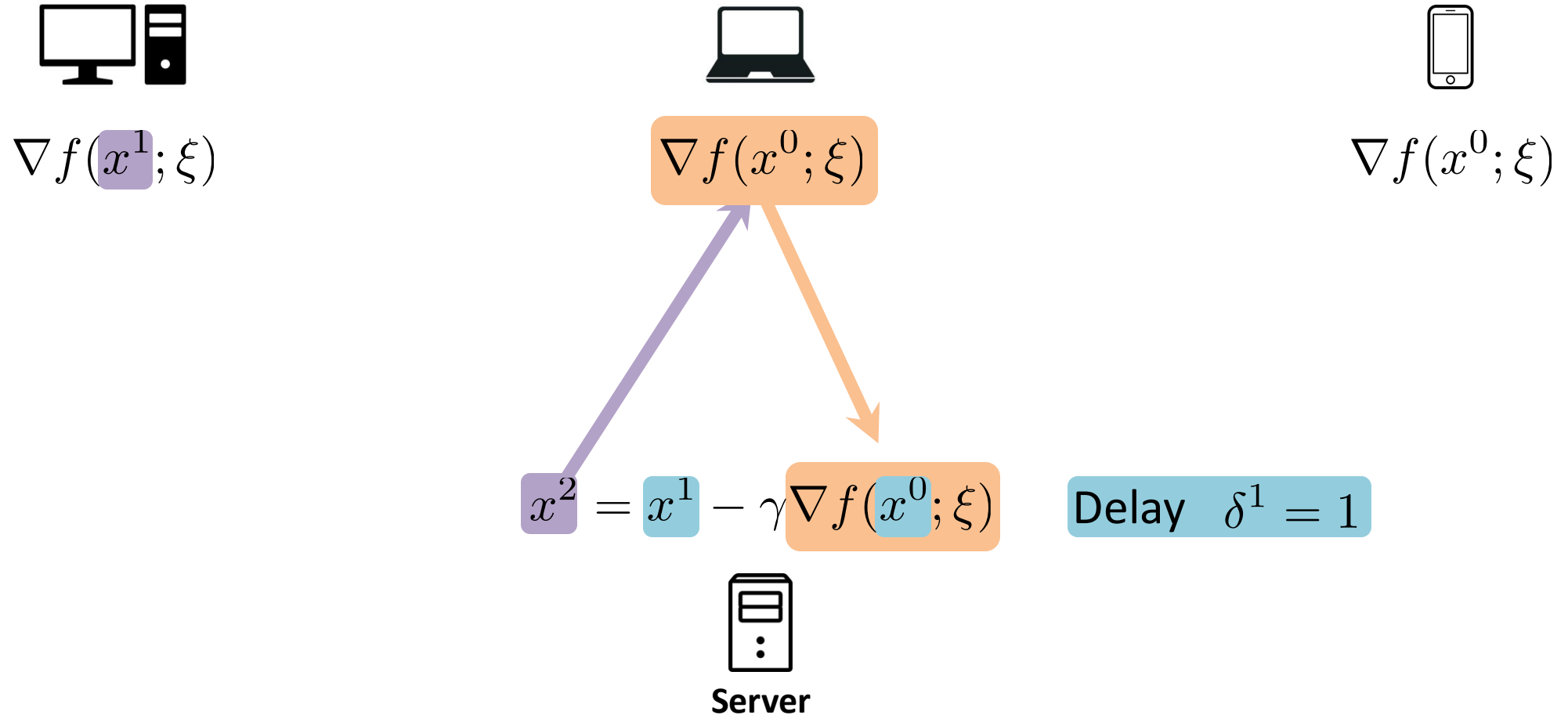


$$x^{k+1} = x^k - \gamma g(x^k)$$


Updates of Asynchronous SGD has delayed stochastic gradients




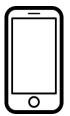
Updates of Asynchronous SGD has delayed stochastic gradients



Updates of Asynchronous SGD has delayed stochastic gradients


$$\nabla f(x^1; \xi)$$


$$\nabla f(x^2; \xi)$$


$$\nabla f(x^0; \xi)$$

$$x^3 = x^2 - \gamma \nabla f(x^0; \xi) \quad \text{Delay } \delta^2 = 2$$



Server

Updates of Asynchronous SGD has delayed stochastic gradients



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

Delay δ^k



Server

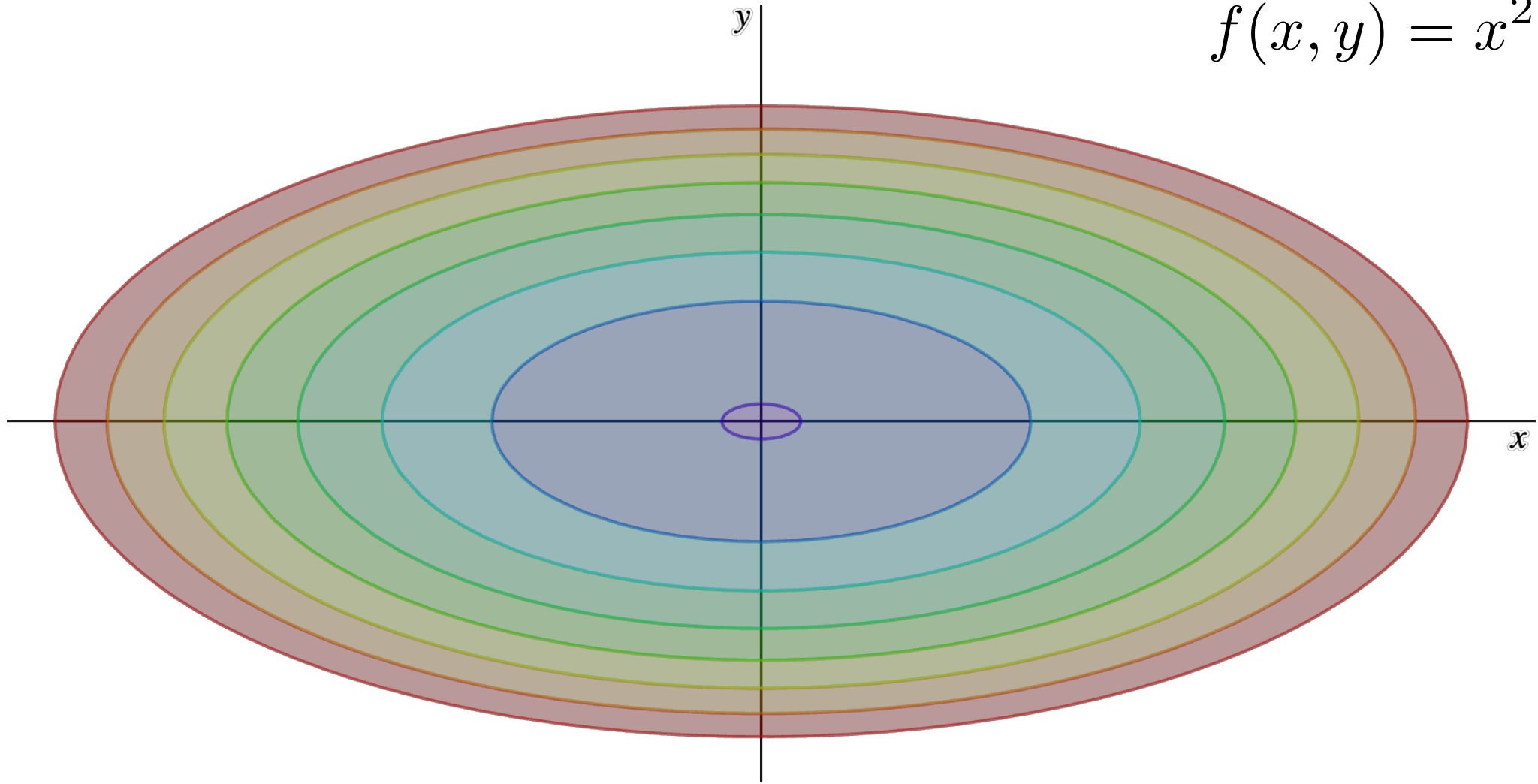


Niu, et al. (2011).

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

Asynchronous SGD can get wild:
delays can degrade performance

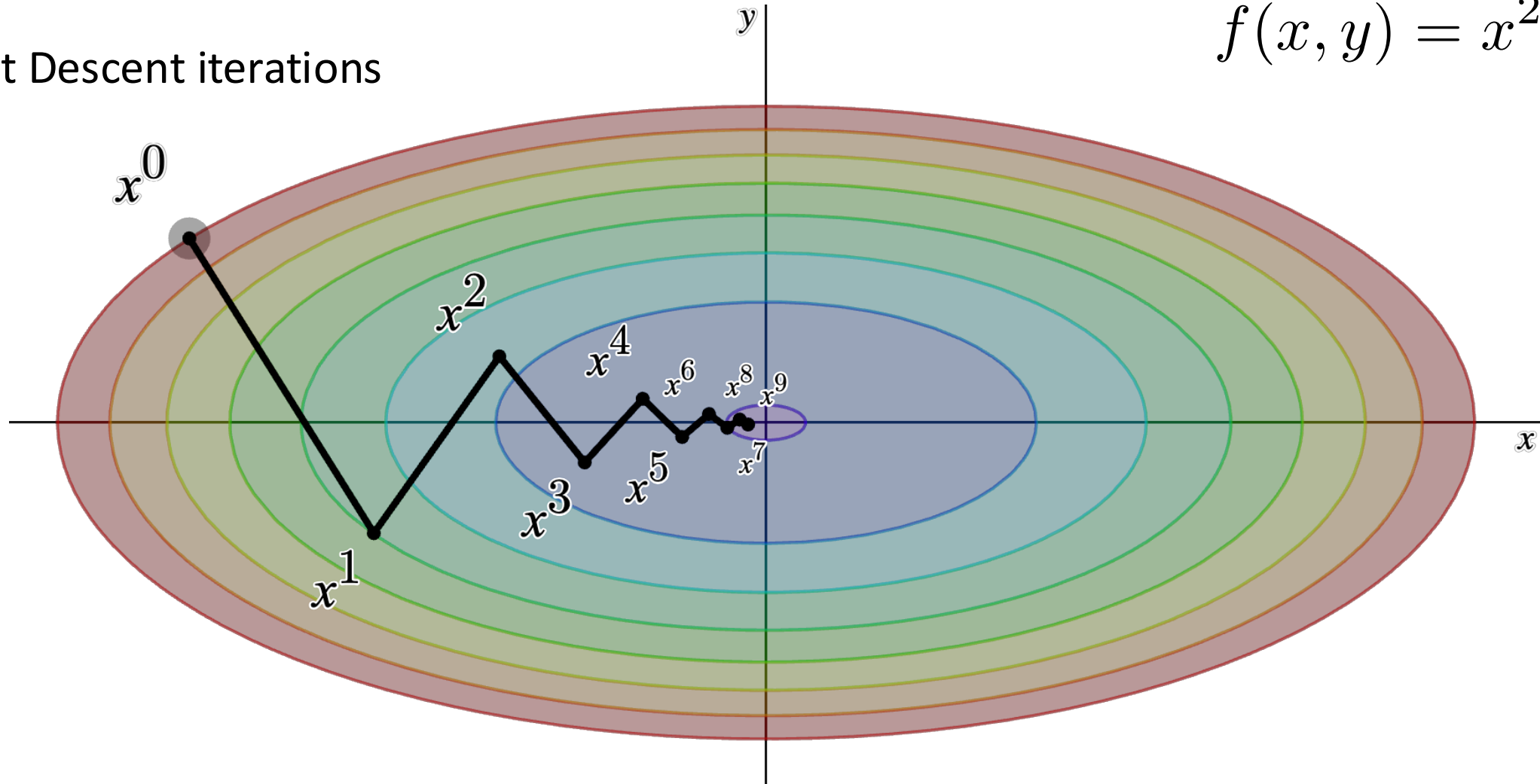
$$f(x, y) = x^2 + 5y^2$$



Asynchronous SGD can get wild: delays can degrade performance

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

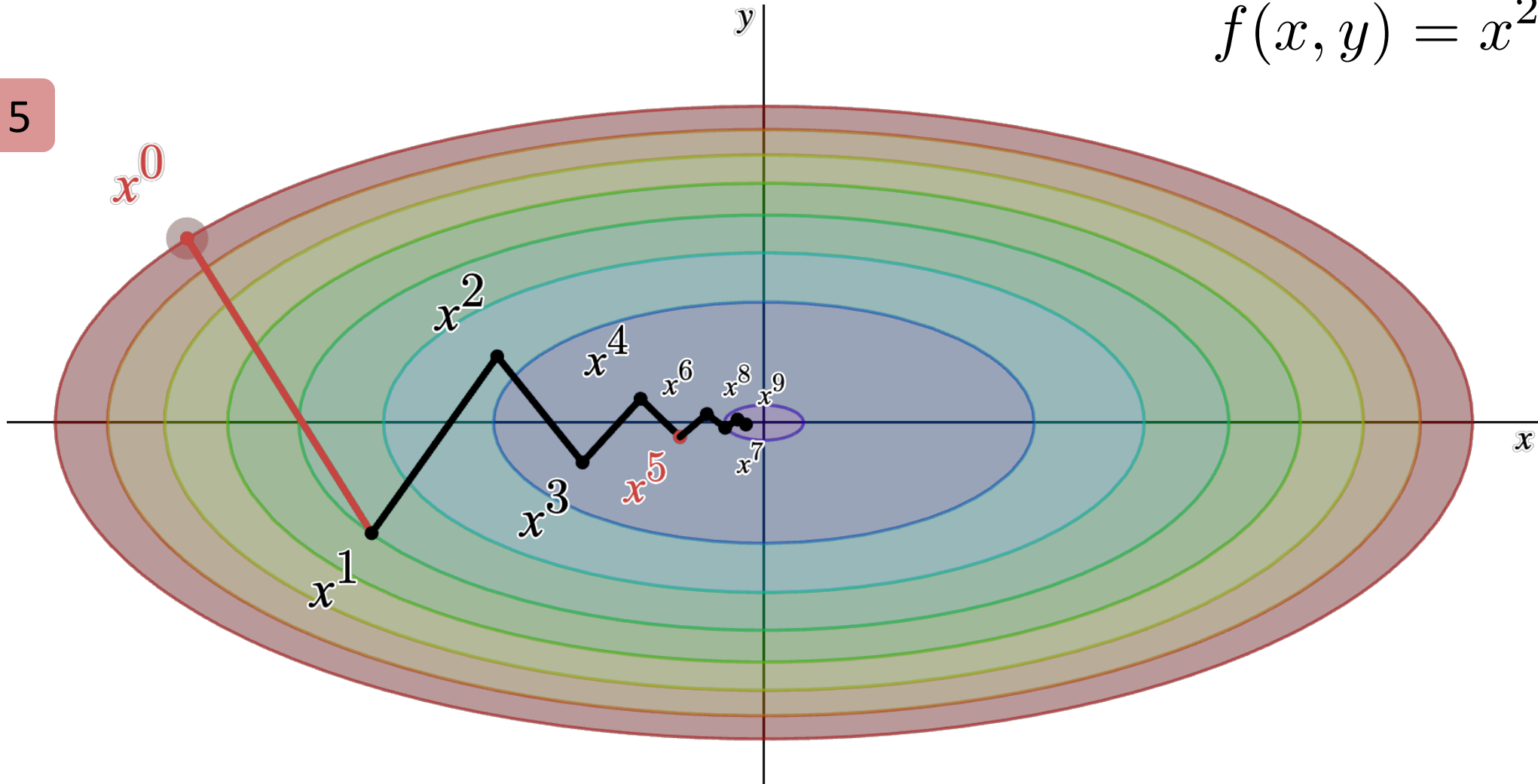
Gradient Descent iterations



Asynchronous SGD can get wild: delays can degrade performance

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

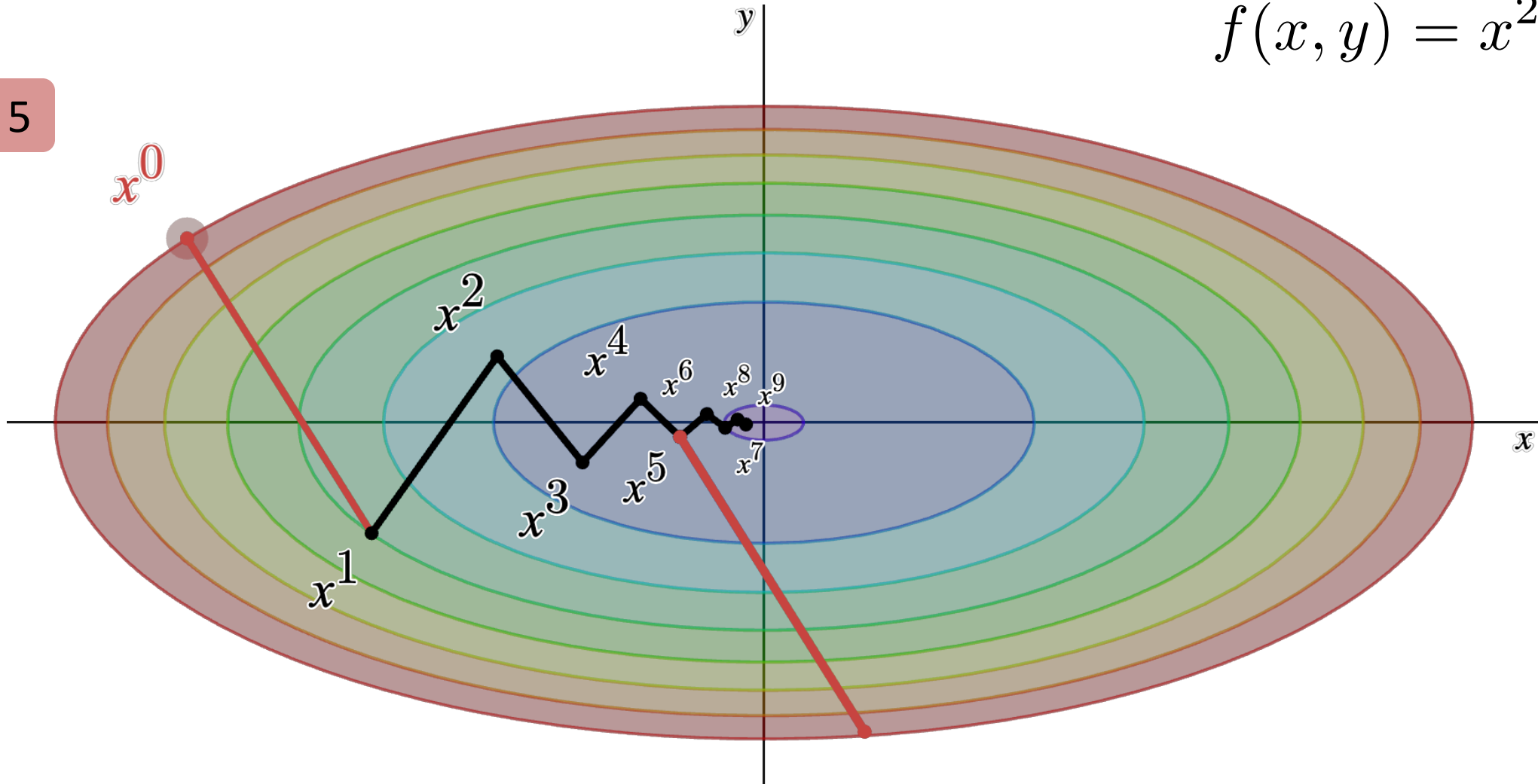
Delay = 5



Asynchronous SGD can get wild: delays can degrade performance

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

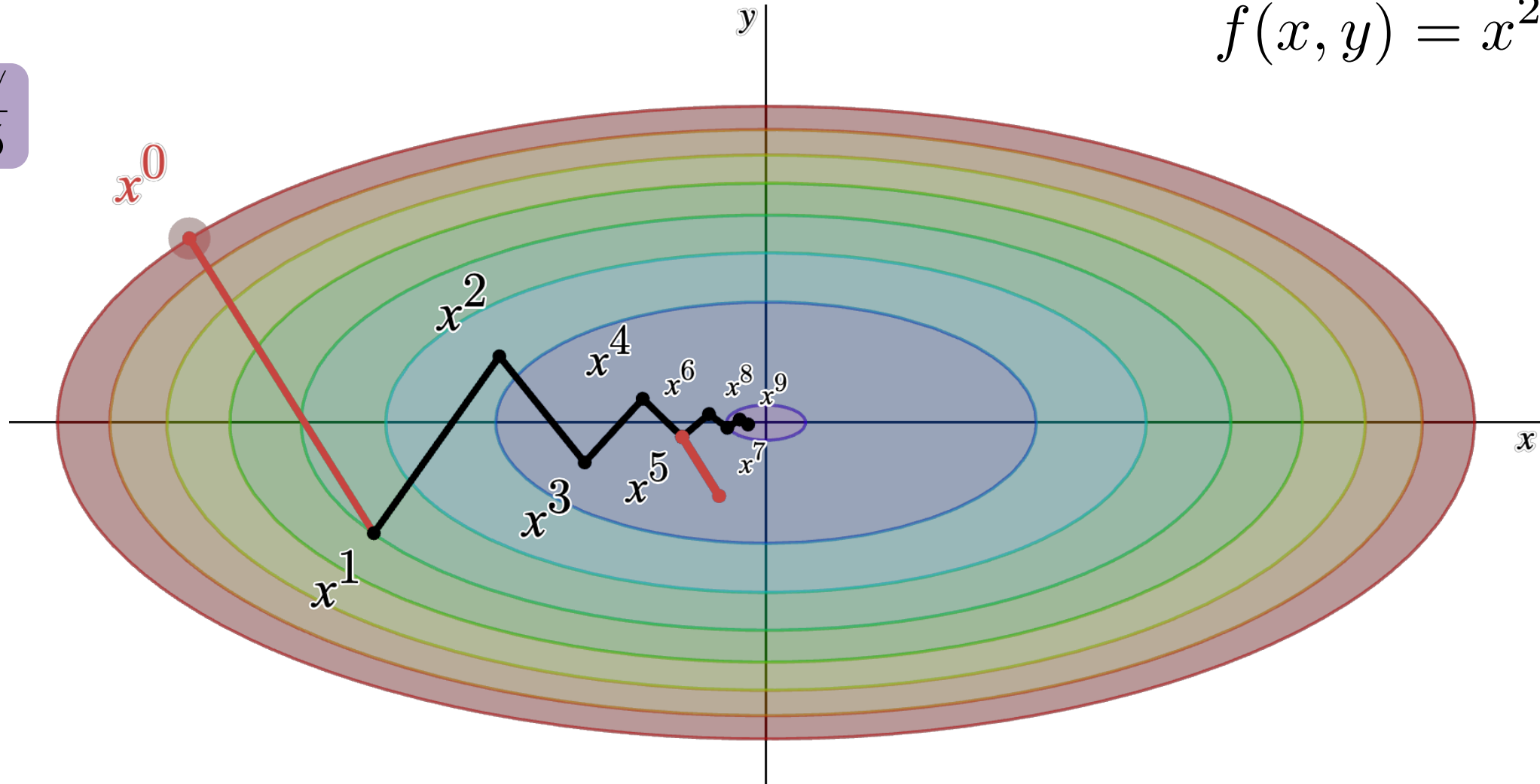
Delay = 5



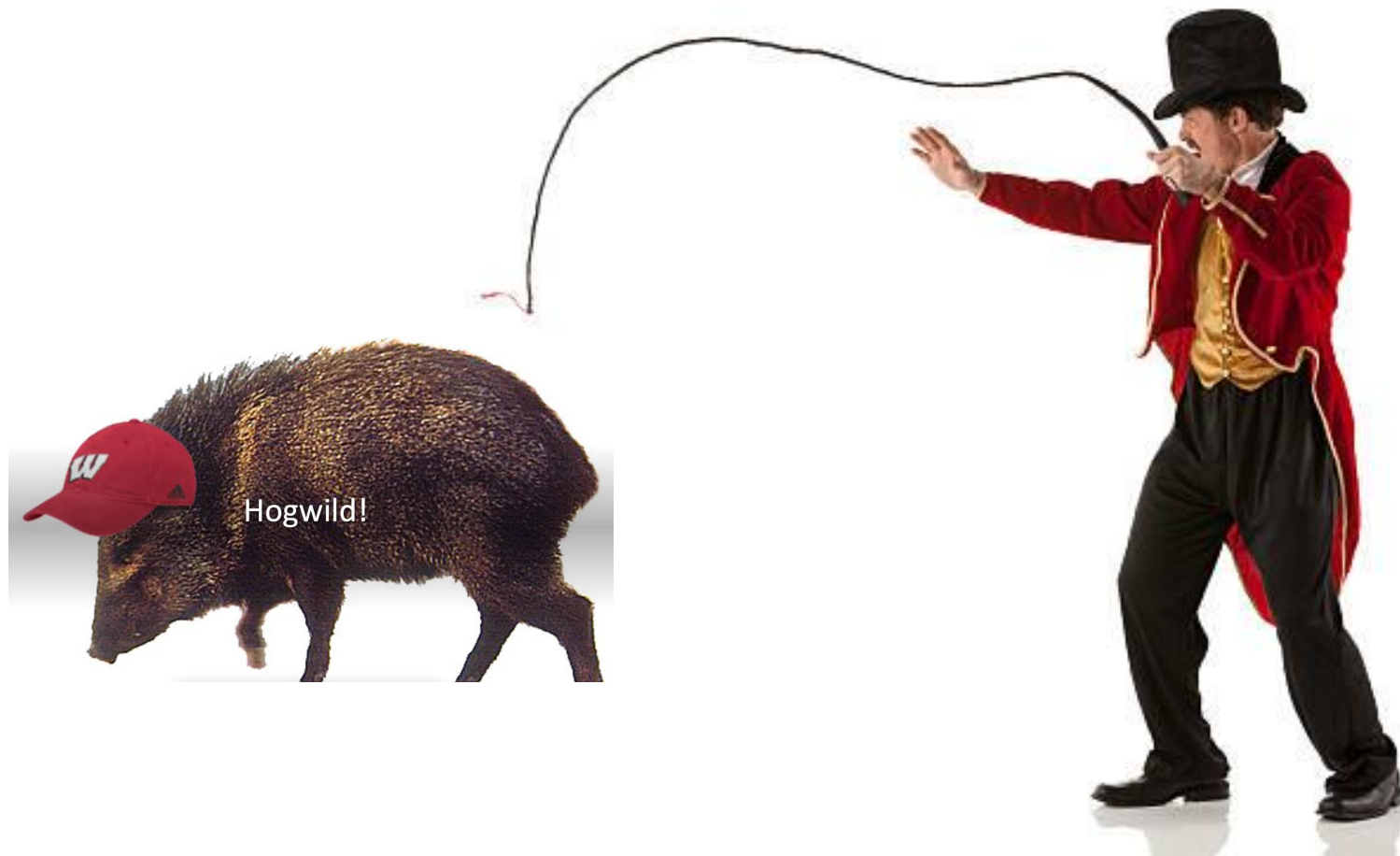
How to fix this?
Make the stepsize smaller

$$\gamma = \frac{\gamma}{5}$$

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



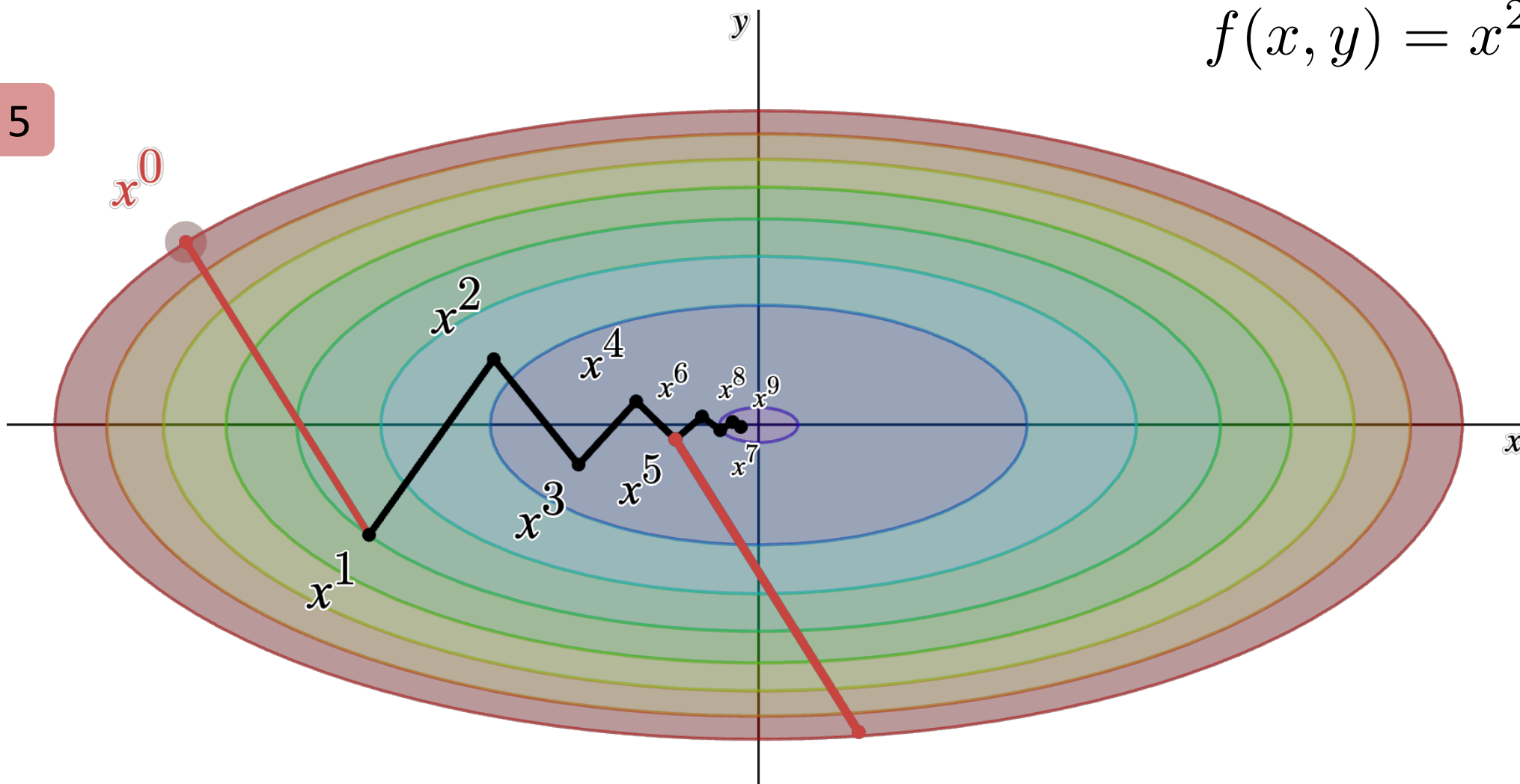
Asynchronous SGD is too wild:
Ringmaster ASGD *tames* it



The smaller the delay, the better the gradient

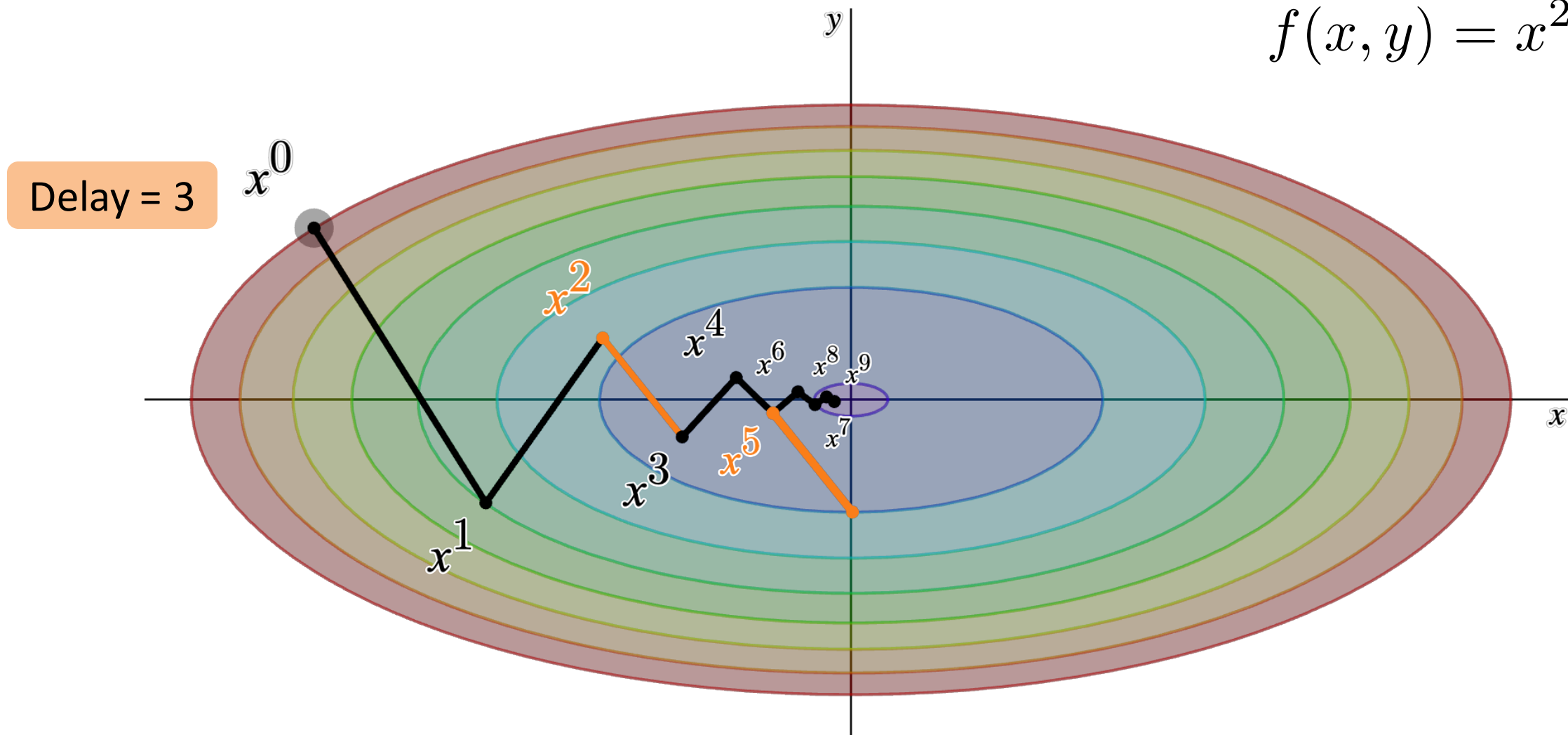
$$f(x, y) = x^2 + 5y^2$$

Delay = 5



The smaller the delay,
the better the gradient

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



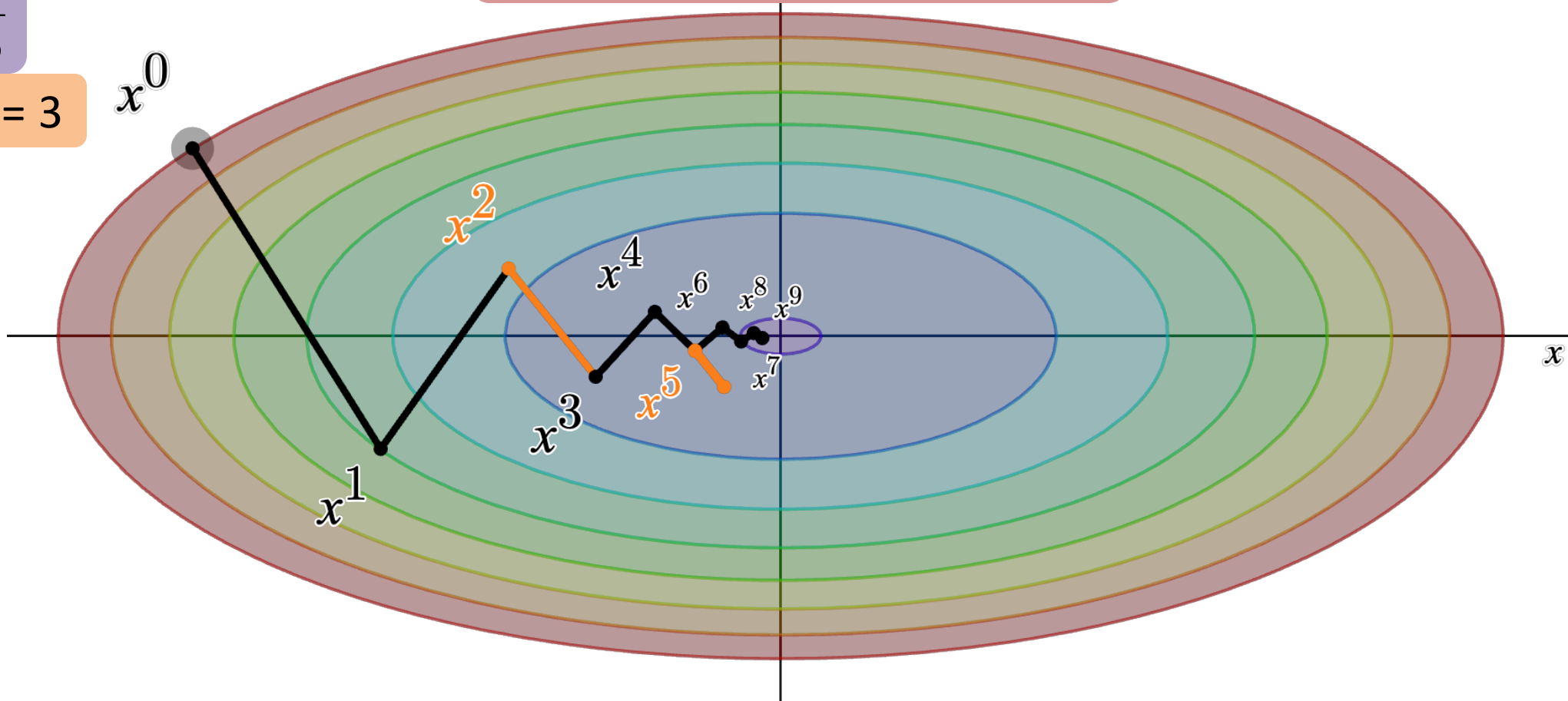
The smaller the delay, the better the gradient

How can we reduce the delay?

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

$$\gamma = \frac{\gamma}{5}$$

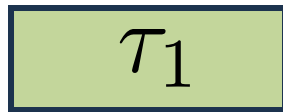
Delay = 3



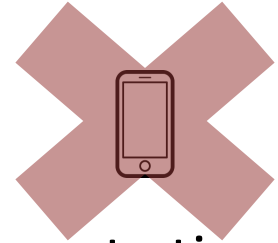
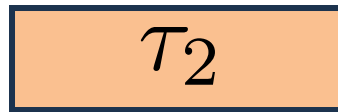
Naive approach: Remove slow workers



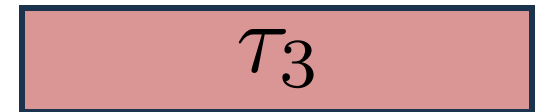
Compute time = τ_1



Compute time = τ_2



Compute time = τ_3



Server

Naive approach: Remove slow workers

Use only the first

$$m_{\star} = \arg \min_{m \in [n]} \left\{ \left(\frac{1}{m} \sum_{i=1}^m \frac{1}{\tau_i} \right)^{-1} \left(1 + \frac{\sigma^2}{m\varepsilon} \right) \right\}$$

fastest workers

$$\mathbb{E} [\|\nabla f(x; \xi) - \nabla f(x)\|^2] \leq \sigma^2$$

$$\mathbb{E} [\|\nabla f(x)\|^2] \leq \varepsilon$$

Problem: τ_i -s may be unknown and dynamic

Ringmaster ASGD: Have a threshold on delays



If: $\delta^k < R$

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

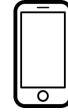
Else: Ignore the gradient and send the current point x^k to the worker



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



Ringmaster ASGD: Have a threshold on delays



How to choose the delay threshold R

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

$$\nabla f \left(x^k; \xi_i^k \right)$$



Server

Certain threshold choices in Ringmaster ASGD recover previous methods

$$R = \max \left\{ 1, \left\lceil \frac{\sigma^2}{\varepsilon} \right\rceil \right\}$$

$R = 1$
Hero SGD

Sweet spot

$R = \infty$
HOGWILD!



Theoretical results validate our intuition

$$\mathcal{O}\left(\frac{R}{\varepsilon} + \frac{\sigma^2}{\varepsilon^2}\right)$$

Number of iterations

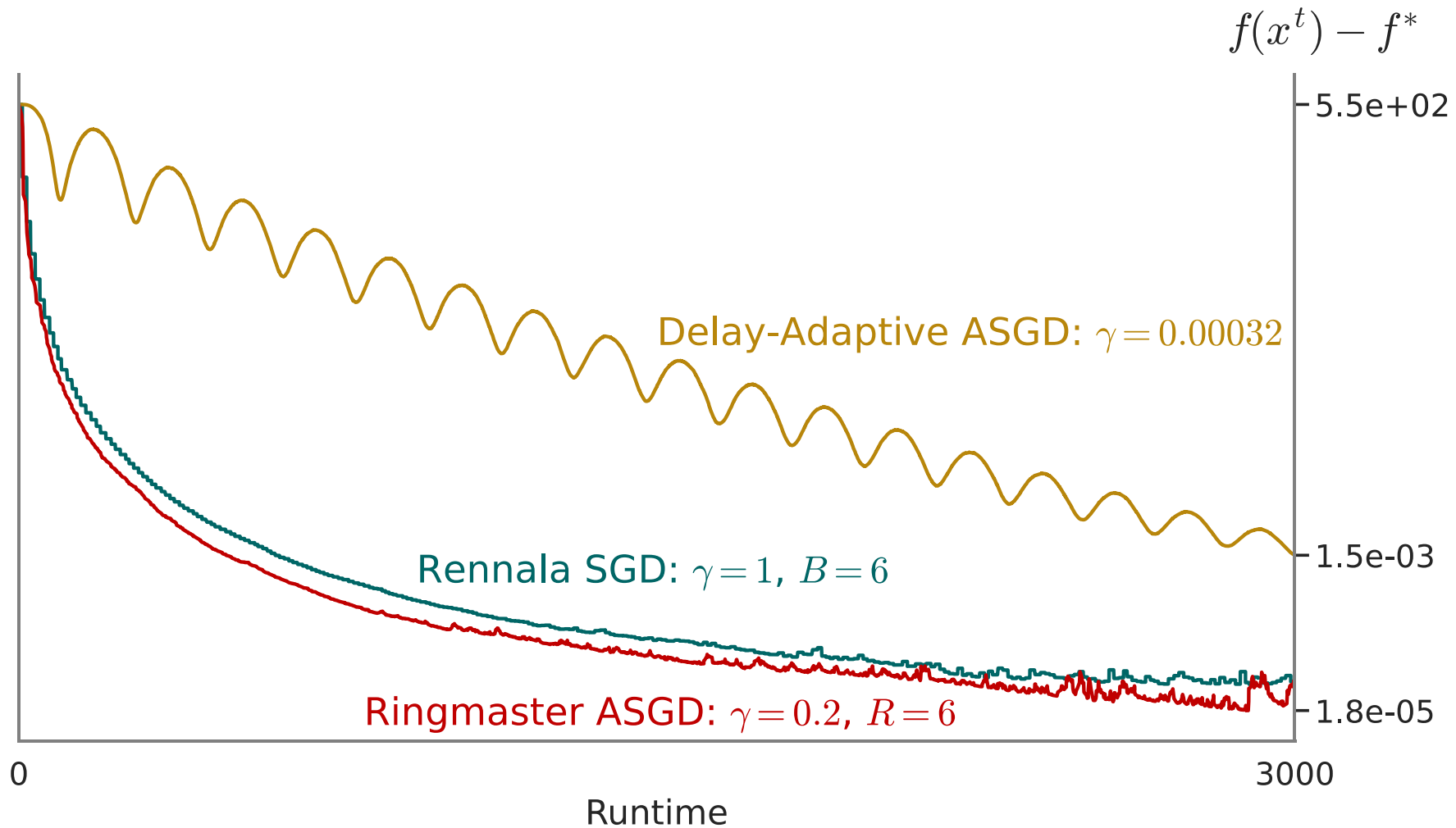
$$\mathcal{O}\left(\min_{m \in [n]} \left[\left(\frac{1}{m} \sum_{i=1}^m \frac{1}{\tau_i} \right)^{-1} \left(\frac{1}{\varepsilon} + \frac{\sigma^2}{m\varepsilon^2} \right) \right]\right)$$

Time complexity

non-decreasing

decreasing

Ringmaster ASGD outperforms existing baselines



Recap of what we have covered

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



$\nabla f(x; \xi)$

Compute time = τ_1

τ_1



$\nabla f(x; \xi)$

Compute time = τ_2

τ_2



$\nabla f(x; \xi)$

Compute time = τ_3

τ_3

Problem setup

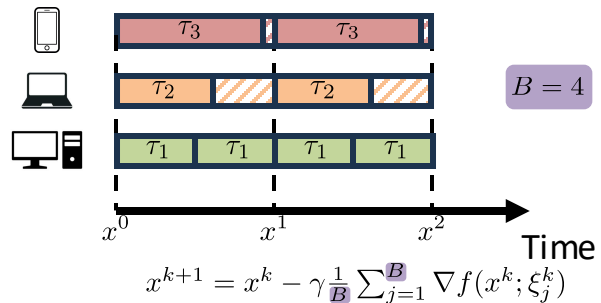
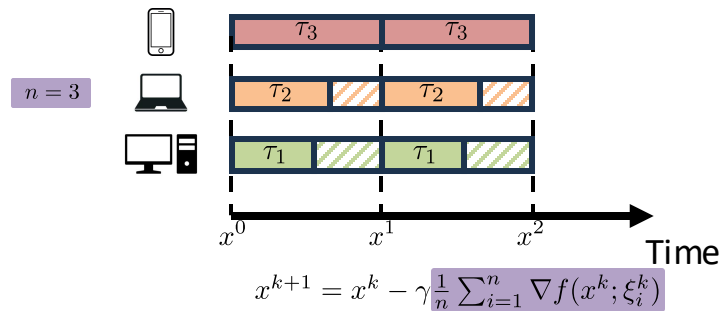
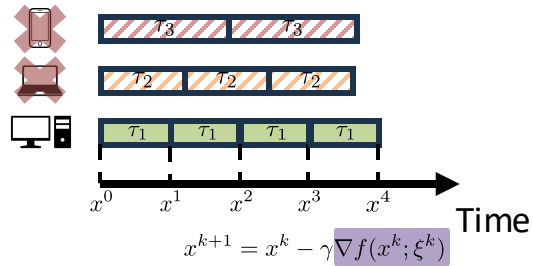
Optimization objective

Heterogenous system

Method (SGD)

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

Recap of what we have covered



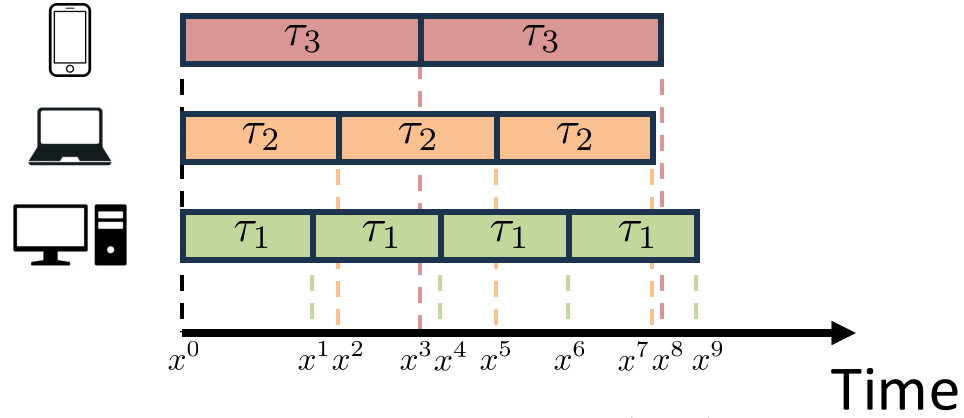
Problem setup

- Optimization objective
- Heterogenous system
- Method (SGD)

Different ways of parallelizing SGD

Synchronized approaches

Recap of what we have covered



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

Problem setup

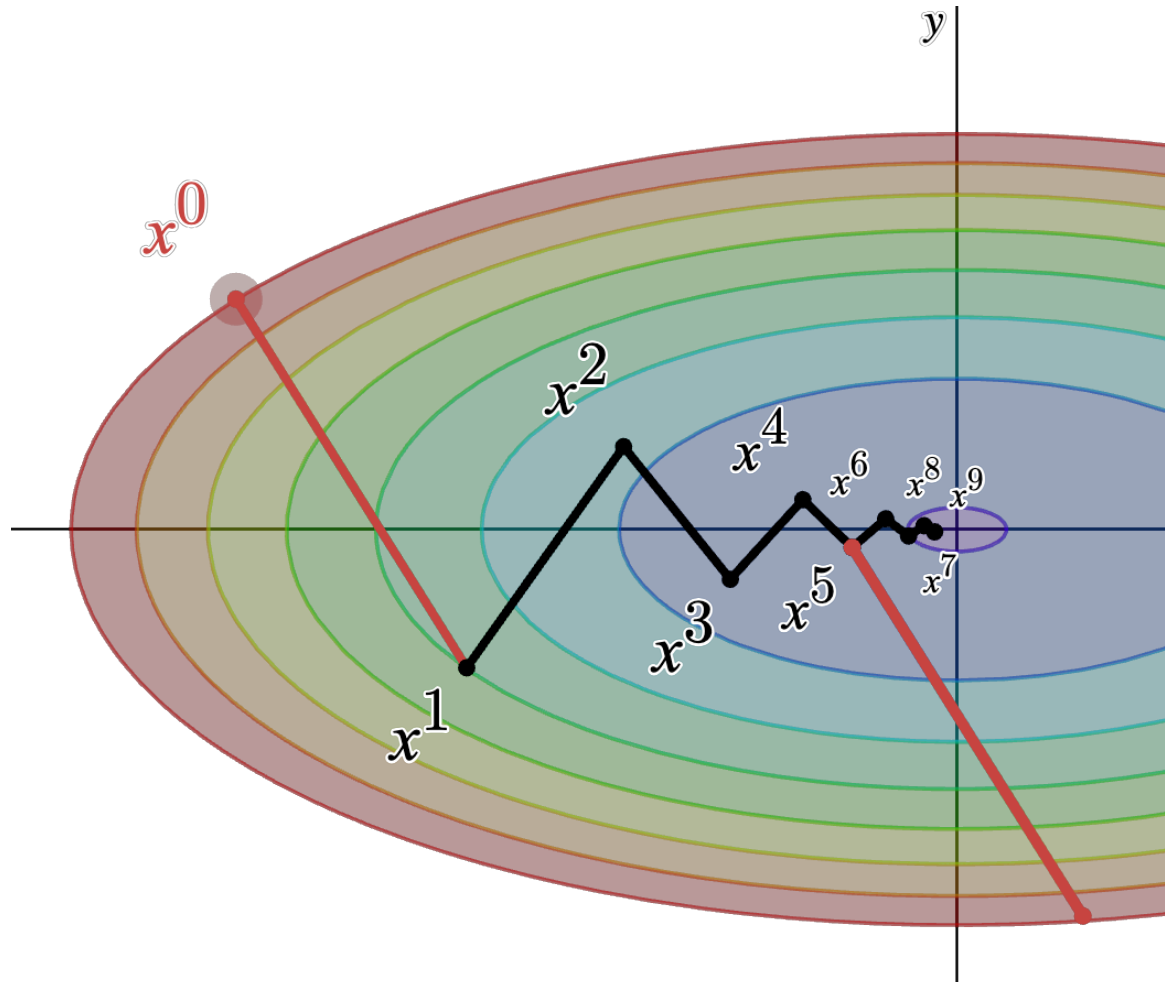
- Optimization objective
- Heterogenous system
- Method (SGD)

Different ways of parallelizing SGD

- Synchronized approaches
- Asynchronous SGD



Recap of what we have covered



Problem setup

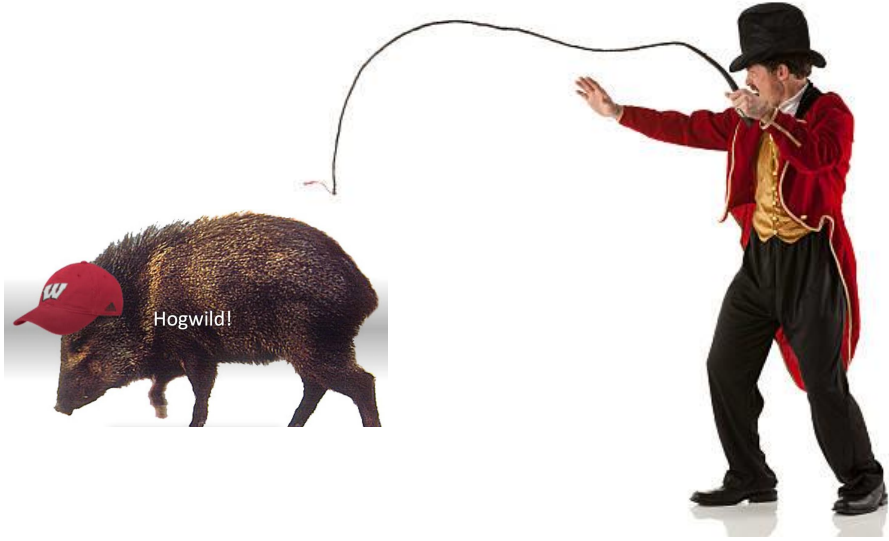
- Optimization objective
- Heterogenous system
- Method (SGD)

Different ways of parallelizing SGD

- Synchronized approaches
- Asynchronous SGD

Problems of ASGD

Recap of what we have covered



$$\nabla f(x^k; \xi_i^k)$$

If: $\delta^k < R$

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

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



Server

Problem setup

- Optimization objective
- Heterogenous system
- Method (SGD)

Different ways of parallelizing SGD

- Synchronized approaches
- Asynchronous SGD

Problems of ASGD

Ringmaster ASGD



Alexander Tyurin
Skoltech



Peter Richtárik
KAUST

Closely related papers

Artavazd Maranjyan, Omar Shaikh Omar, Peter Richtárik (2024)

MindFlayer: Efficient asynchronous parallel SGD in the presence of heterogeneous and random worker compute times

Artavazd Maranjyan, El Mehdi Saad, Peter Richtarik, and Francesco Orabona (2025)

ATA: Adaptive Task Allocation for Efficient Resource Management in Distributed Machine Learning



Hogwild!

