Distributed Gradient Descent

Sai Ganesh Nagarajan, Assistant Professor @ IMADA

AI801 Lecture on 04.05.2026

University of
Southern Denmark

Special Topics Outline

- Last week: How to optimise against the unknown?
  - Notion of regret minimisation
  - Online Gradient Descent (OGD)
  - Example: Online Support Vector Machines
  - Stochastic Gradient Descent via OGD
- Three main topics:
  - Online Gradient Descent (April 27th)
  - **Distributed Gradient Descent (May 4th)**
  - Communication-Efficient Gradient Descent (May 11th)
- Today: How to optimize in large-scale distributed systems?

Regret Minimization

Regret $\longrightarrow$ $$R(T) = \underbrace{\sum_{t=0}^{T-1} h_t(x_t)}_{\text{Total loss actually obtained during the T rounds.}} - \underbrace{\min_{x \in \mathcal{X}} \sum_{t=0}^{T-1} h_t(x)}_{\text{Total loss of the best fixed point (in hindsight)}}$$

Player's goal is to use $\{x_t\}_{t=0}^{T-1}$ such that: $\lim_{T \to \infty} \frac{R(T)}{T} = 0$

No-regret Algorithm: Update rules for x_t, that satisfy $R(T) = o(T)$.

Online Gradient Descent

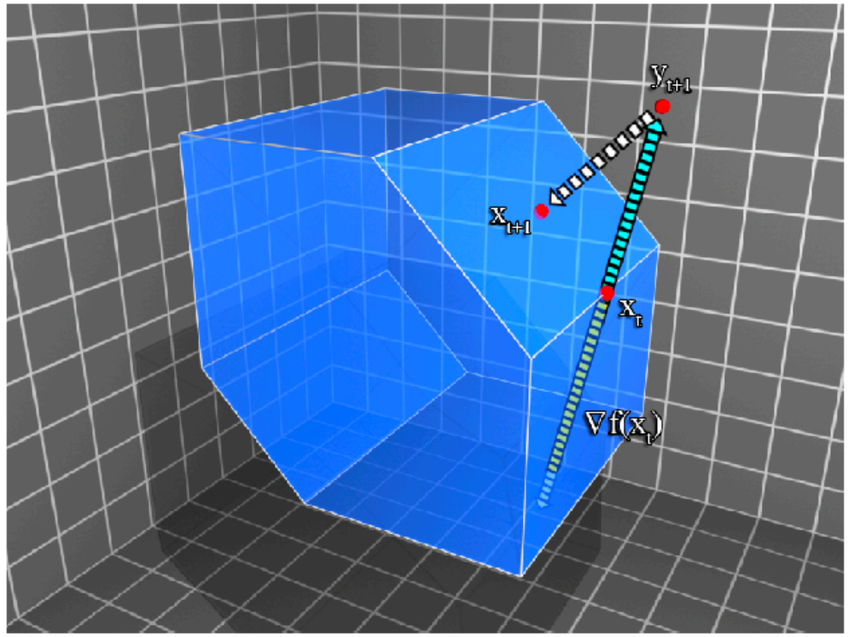
Initialize with $x_0 \in \mathcal{X}$

For $t = 0$ to $T - 1$ do,

$$y_{t+1} = x_t - \eta_t \nabla h_t(x_t)$$

$$x_{t+1} = \Pi_{\mathcal{X}}(y_{t+1})$$

Return x_T



$$\Pi_{\mathcal{X}}(y) := \text{argmin}_{x \in \mathcal{X}} \|x - y\|_2^2$$

OGD satisfies $R_T = o(T)$, with appropriate selection of $\{\eta_t\}$ and assuming uniformly bounded norm of gradients.

Motivation - Private/Personalized Learning

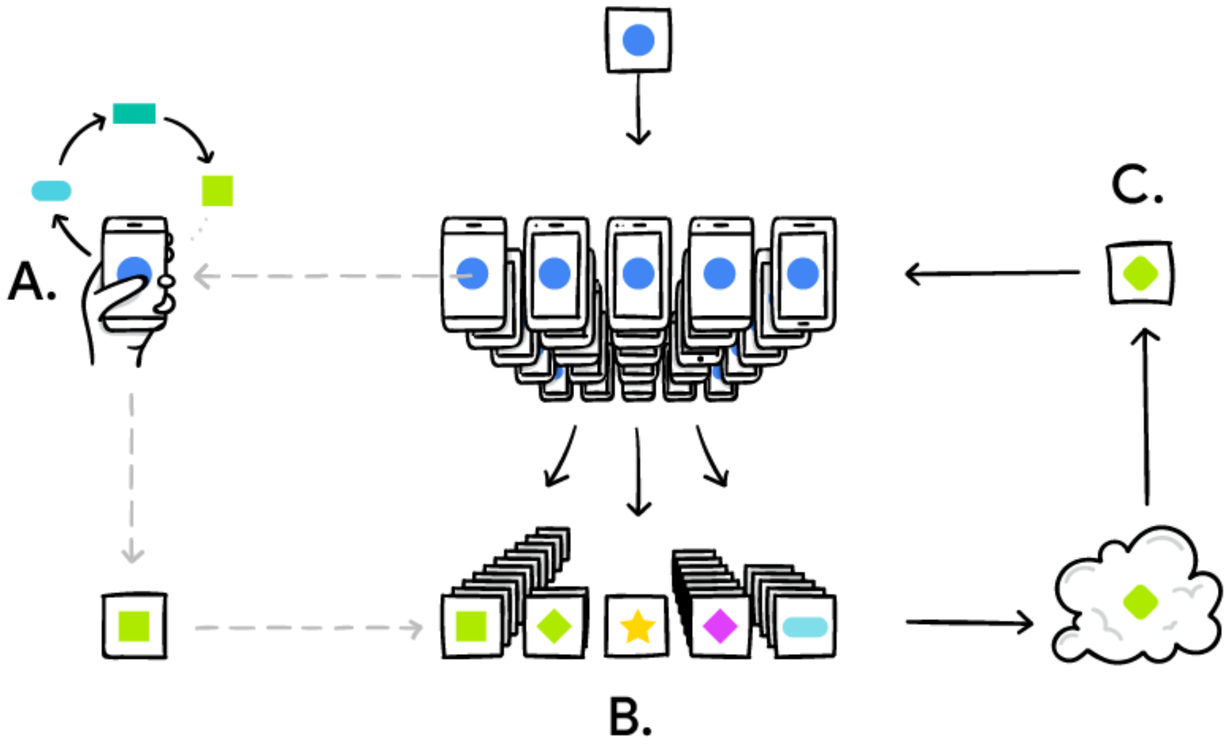


Image source

Mobile Keyboard Predictions

Motivation - Recent Compute Trends

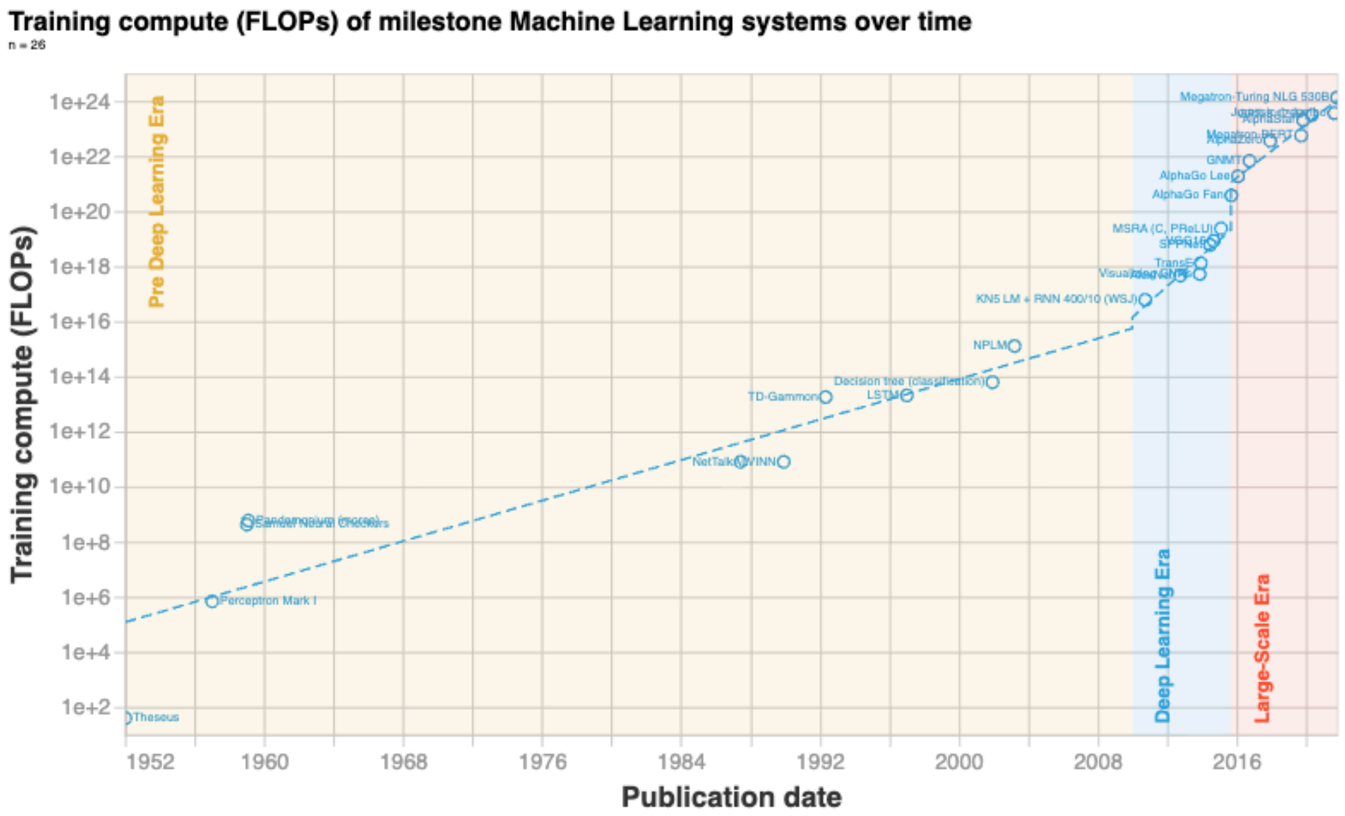


Image source

Motivation - Recent Compute Trends

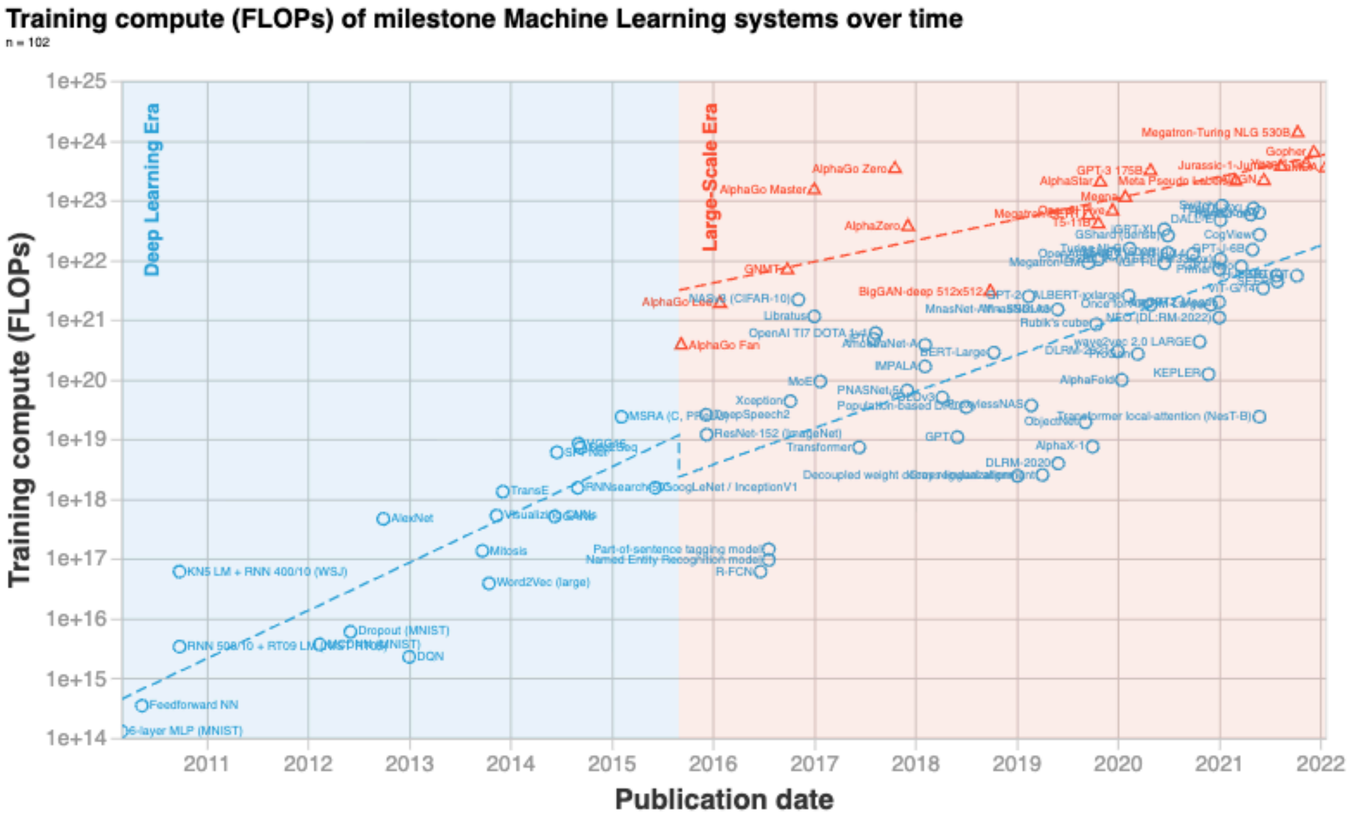


Image source

Parallel Paradigms

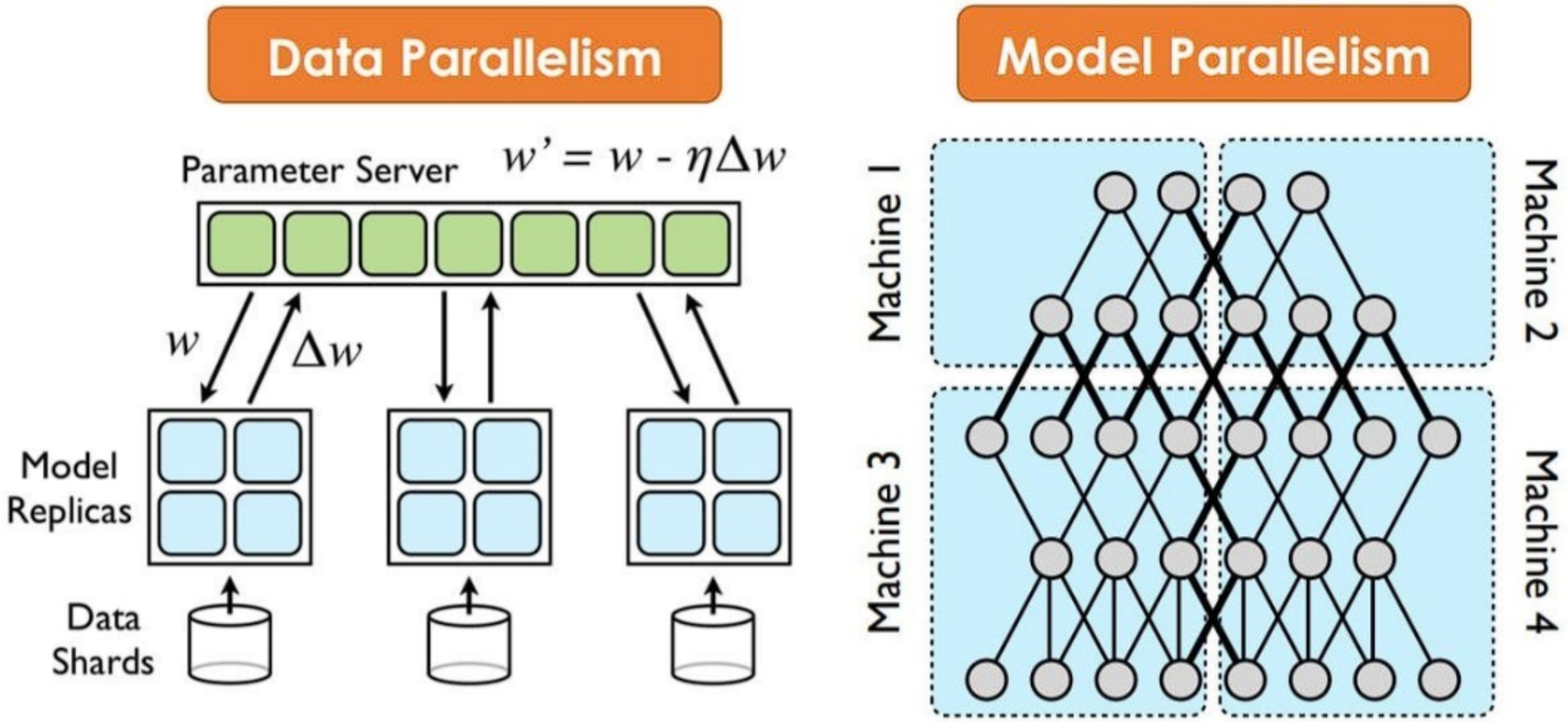


Image source

Gradient Descent

Initialize with x_0

For $t = 0$ to $T - 1$ do,

$$x_{t+1} = x_t - \eta_t \nabla h(x_t)$$

Return $\bar{x}_T = \text{argmin}_{t=0,1,\ldots,T-1}\{h(x_t)\}$

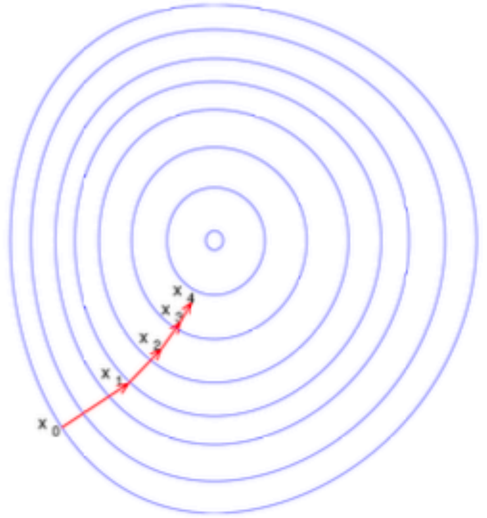


Figure 2.3: Iterates of the GD algorithm

Here we return the **best-iterate**, sometimes, we might consider,

the **last-iterate** x_T or an **average iterate** $\frac{1}{T}\sum_{t=0}^{T-1} x_t$

Stochastic Gradient Descent

Initialize with x_0

For $t = 0$ to $T - 1$ do,

$$x_{t+1} = x_t - \eta_t \tilde{\nabla} h(x_t)$$

Return $\bar{x}_T = \frac{1}{T} \sum_{t=0}^{T-1} x_t$

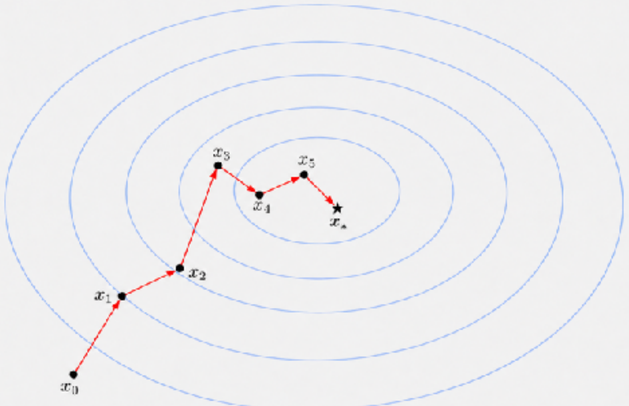


Figure: Iterates of the SGD algorithm

Note: $\tilde{\nabla} h(x_t)$ is a **noisy** gradient at x_t!

Stochastic Gradient Descent in ML

Suppose $x_i \in \mathcal{X}$ is a feature vector and $y_i \in \{-1, 1\}$ is the label such that $\{(x_i, y_i)\}$ are **i.i.d samples** from a fixed (but unknown) distribution. Let there be N data points.

Empirical Loss: $h_N(w) := \frac{1}{N} \sum_{i=0}^{N-1} h(w; (x_i, y_i))$

Noisy Gradient at w_t on sample $j \in [N]$: $\nabla h(w_t; (x_j, y_j))$

Note: $\nabla h(w_t; (x_j, y_j)) \neq \nabla h_N(w_t)$

Stochastic Gradient Descent in ML

A1: Unbiased gradient estimate: $\mathbb{E}[\,\tilde{\nabla} h_N(w_t) \,|\, w_t] = \nabla h_N(w_t)$

A2: Variance of the gradient to be bounded: $\mathbb{E}[\,\|\tilde{\nabla} h_N(w_t)\|_2^2 |\, w_t] \leq G^2$

Suppose we pick a sample $j \in [N]$ uniformly at random (u.a.r) from $[N]$:

$$\mathbb{E}[\,\nabla h(w_t; (x_j, y_j)) \,|\, w_t] = \frac{1}{N} \sum_{i=0}^{N-1} \nabla h(w_t; (x_i, y_i)) = \nabla h_N(w_t)$$

Initialize with w_0

For $t = 0$ to $T - 1$ do,

Pick sample $j \in [N]$ u.a.r

$$w_{t+1} = w_t - \eta_t \nabla h(w_t; (x_j, y_j))$$

Note: For strongly convex functions, it suffices to return the last-iterate, i.e., w_T

Return $\bar{w}_T = \frac{1}{T} \sum_{t=0}^{T-1} w_t$

Convergence Rates of SGD

	General (Return average-iterate)	Strongly convex (return last-iterate)
SGD	$\frac{1}{\sqrt{T}}$	$\frac{1}{\alpha T}$

SGD converges to a minimizer in the expectation, i.e., $\mathbb{E}[h(\bar{x}_T)] - h(x^*) \leq O(r(T))$, under an appropriate choice of the sequence of step-sizes $\{\eta_t\}$.

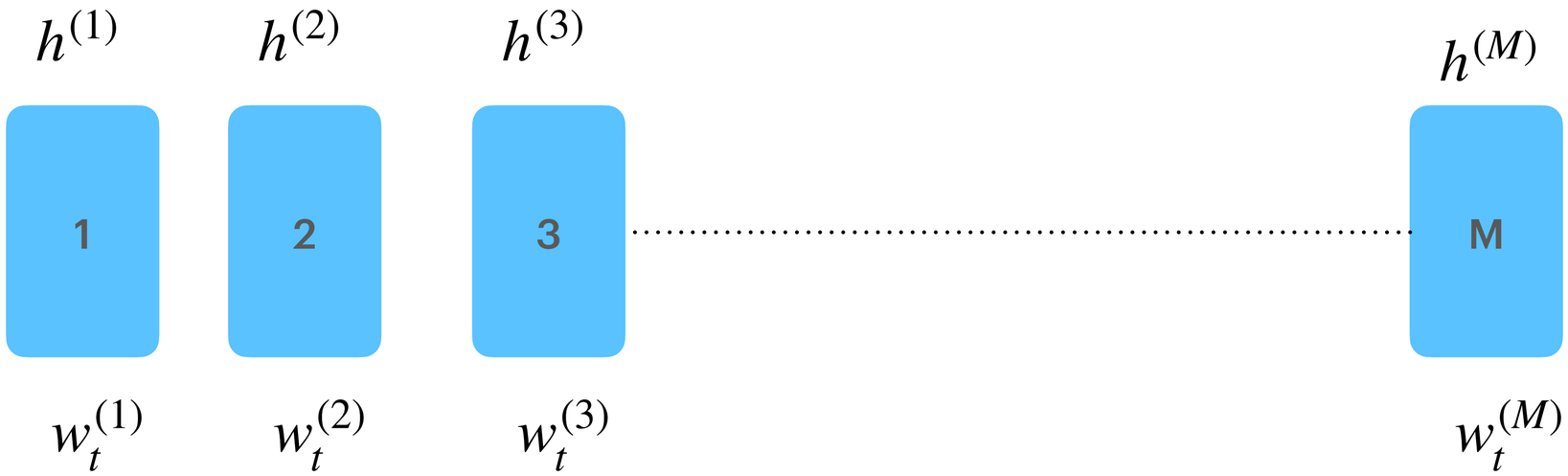
Convergence Rates of SGD

	General (Return average-iterate)	Strongly convex (return last-iterate)
SGD	$\frac{1}{\sqrt{T}}$	$\frac{1}{\alpha T}$

With additional variance reduction methods, one can get faster convergence for well-conditioned (smooth + strongly-convex) problems matching GD!

SGD converges to a minimizer in the expectation, i.e., $\mathbb{E}[h(\bar{x}_T)] - h(x^*) \leq O(r(T))$, under an appropriate choice of the sequence of step-sizes $\{\eta_t\}$.

Parallel SGD



$$h(w) = \frac{1}{M} \sum_{k=1}^{M} \mu_k h^{(k)}(w)$$

Parallelized SGD

Initialize with $w_0^{(k)} = w_0$

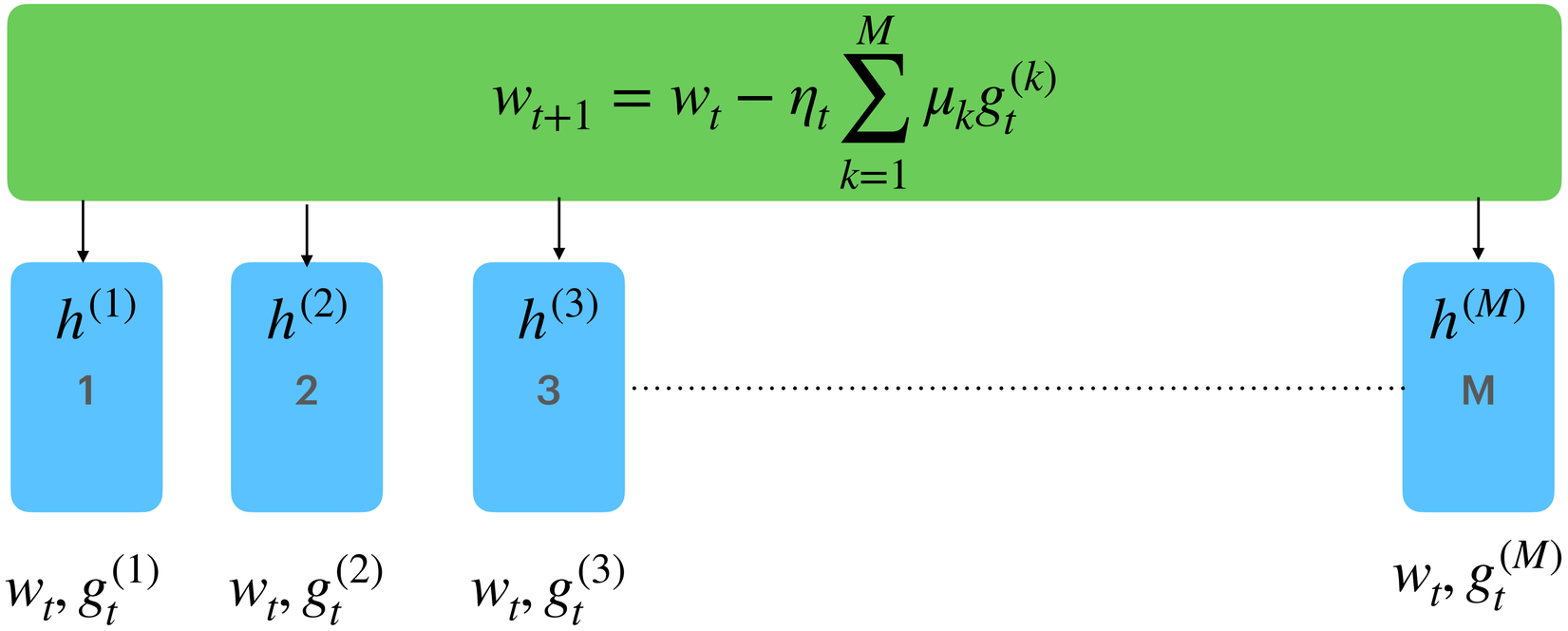
For $k = 1$ to M in parallel do,

$$w_T^{(k)} = \text{SGD}(h^{(k)}, w_0; \eta^{(k)})$$

Return $\bar{w}_T = \frac{1}{M} \sum_{k=1}^{M} w_T^{(k)}$

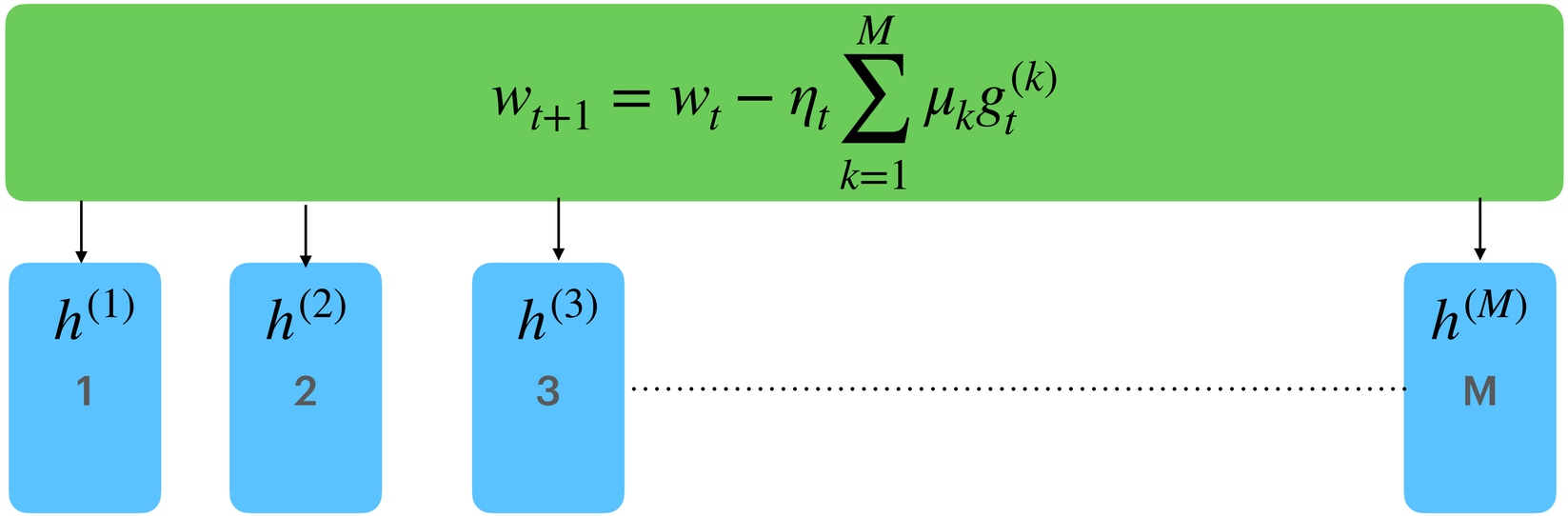
Note: Only one round of communication at the end.

Synchronous SGD



$$h(w) = \sum_{k=1}^{M} \mu_k h^{(k)}(w)$$

Synchronous SGD/Federated SGD



Gradients here are local and can be from a sample or a mini-batch.

$$h(w) = \sum_{k=1}^{M} \mu_k h^{(k)}(w)$$

Note: Every round involves communication from nodes to a server or between all nodes to synchronize their weights at every step and do the gradient updates.

Convergence Rates of Synchronised SGD

	General (Return average-iterate)	Strongly convex (return last-iterate)
SGD	$\frac{1}{\sqrt{MT}}$	$\frac{1}{\alpha MT}$

SGD converges to a minimizer in the expectation, i.e., $\mathbb{E}[h(\bar{x}_T)] - h(x^*) \leq O(r(T))$, under an appropriate choice of the sequence of step-sizes $\{\eta_t\}$.

Convergence Rates of Synchronised SGD

	General (Return average-iterate)	Strongly convex (return last-iterate)
Synch SGD/FedSGD	$\frac{1}{\sqrt{MT}}$	$\frac{1}{\alpha MT}$

Average gradients from M workers, helps reduces the variance, by the factor leading to improved convergence rates. But there is a clear trade-off between the communication overhead!

SGD converges to a minimizer in the expectation, i.e., $\mathbb{E}[h(\bar{x}_T)] - h(x^*) \leq O(r(T))$, under an appropriate choice of the sequence of step-sizes $\{\eta_t\}$.

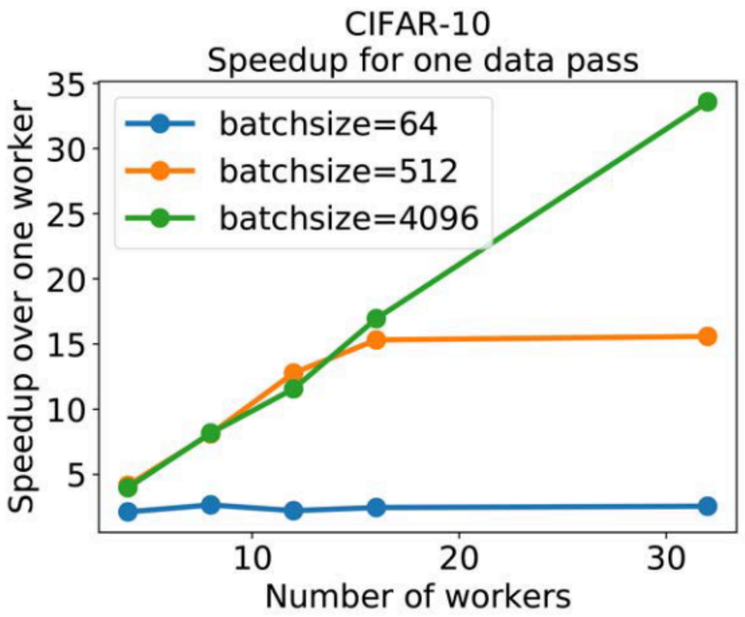
Variance Reduction?

A1: Unbiased gradient estimate: $\mathbb{E}[\,\tilde{\nabla} h_N(w_t) \,|\, w_t] = \nabla h_N(w_t)$

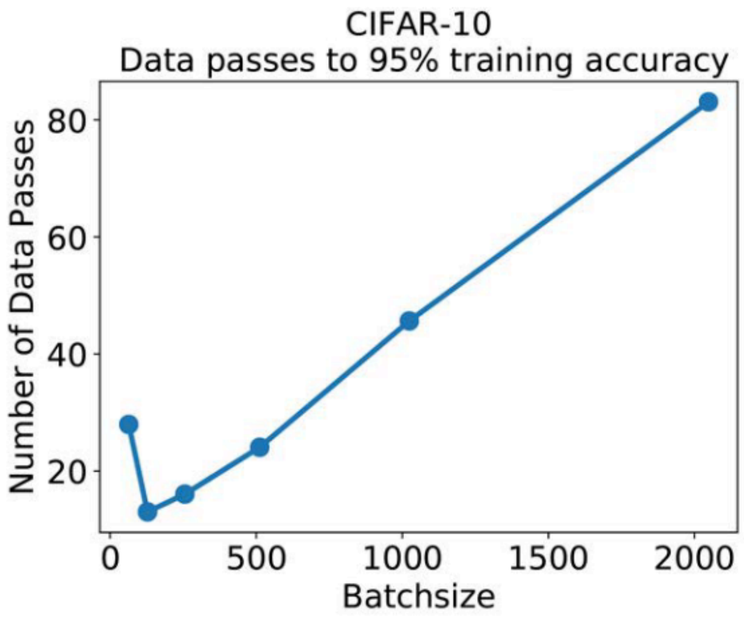
A2: Variance of the gradient to be bounded: $\mathbb{E}[\|\tilde{\nabla} h_N(w_t)\|_2^2 | \, w_t] \leq G^2$

Why?

Gradient Diversity



(a)



(b)

Tradeoff in batch size: Computation/communication vs accuracy

How to adapt batch size for large-scale data?

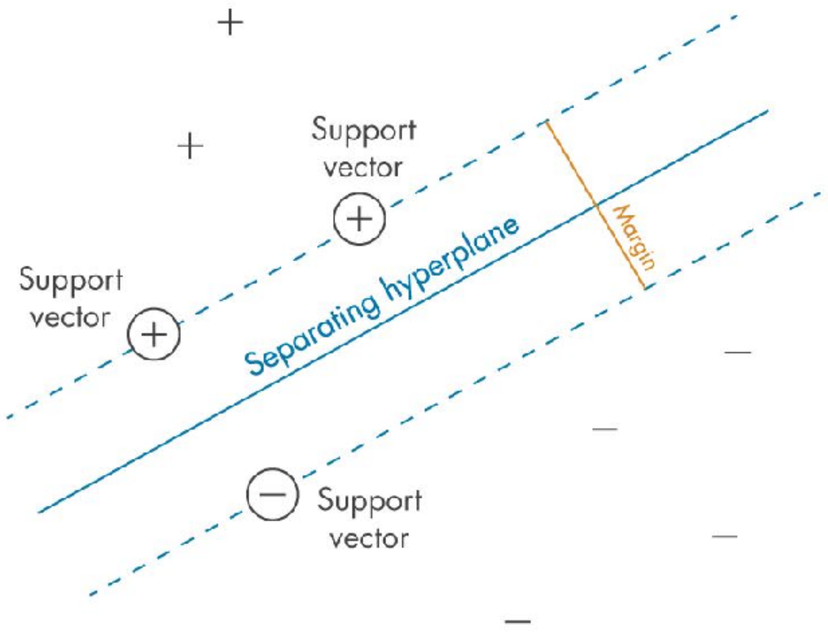
$$\Delta_{\mathcal{S}}(w) = \frac{\sum_{i=1}^{n} \|\nabla f_i(w)\|^2}{\left\| \sum_{i=1}^{n} \nabla f_i(w) \right\|^2} = \frac{\sum_{i=1}^{n} \|\nabla f_i(w)\|^2}{\sum_{i=1}^{n} \|\nabla f_i(w)\|^2 + \sum_{i \neq j} \left\langle \nabla f_i(w), \nabla f_j(w) \right\rangle} .$$

$$B_{\mathcal{S}}(w) = n \, \Delta_{\mathcal{S}}(w) .$$

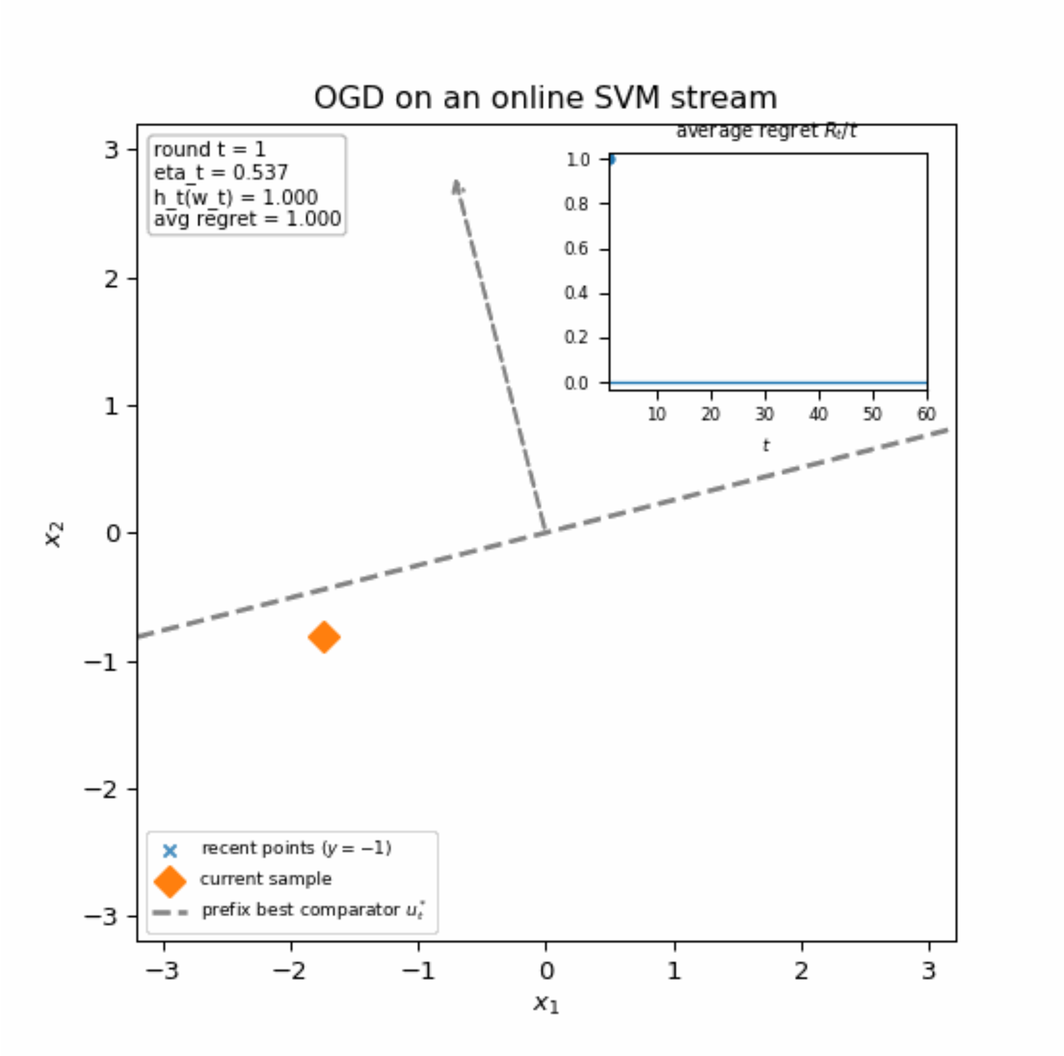
SVM in Machine Learning

Suppose $x_i \in \mathcal{X}$ is a feature vector and $y_i \in \{-1, 1\}$ is the label such that $\{(x_i, y_i)\}$ are **i.i.d samples** from a fixed (but unknown) distribution. Let there be N data points.

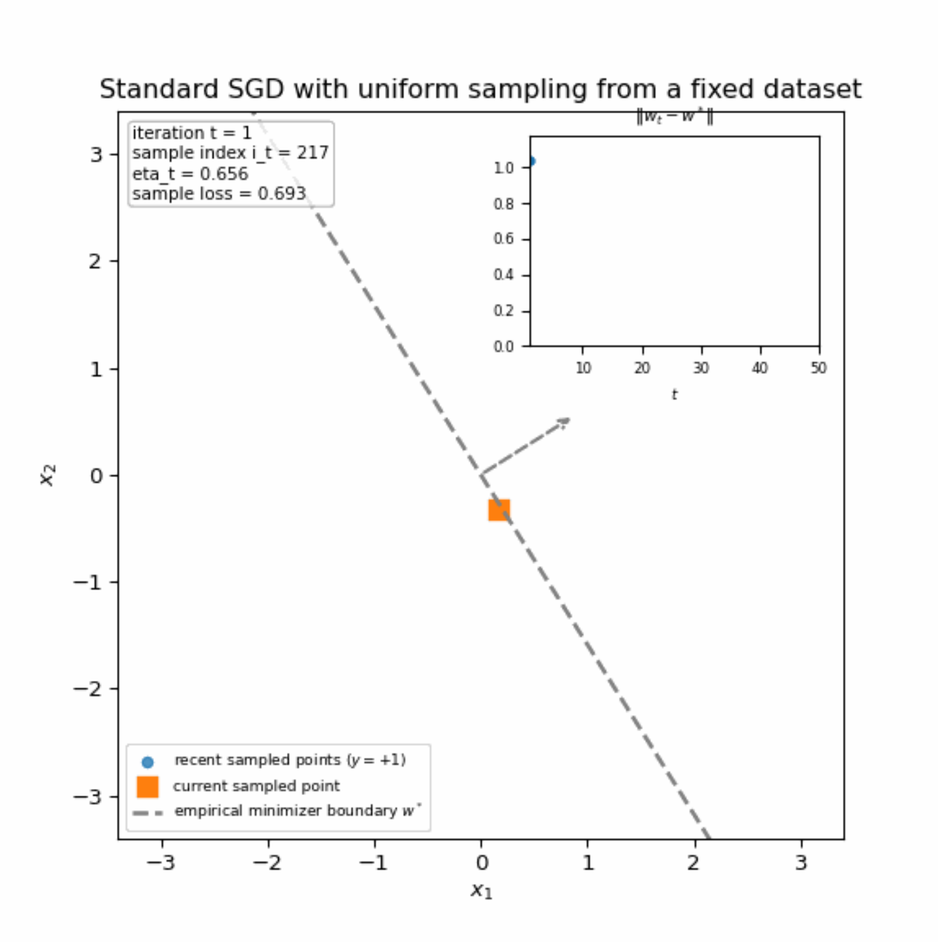
Goal: Find **a separating hyperplane** that minimizes the average hinge-loss: $h_N(w) = \frac{1}{N} \sum_{i=0}^{N-1} h(w; (x_i, y_i))$.



Online SVM OGD



SVM SGD


Standard SGD with uniform sampling from a fixed dataset
iteration t = 1
sample index i_t = 217
eta_t = 0.656
sample loss = 0.693
$\|w_t - w^*\|$
t
recent sampled points (y = +1)
current sampled point
empirical minimizer boundary w^*
x_1
x_2

Summary

- We saw motivation to look at distributed gradient descent.
- We looked at different methods of parallel and synchronous SGD.
- Tradeoff between communication and variance reduction/speed-up in computation.
- SVM revisited.
- Next class: Communication-efficient SGD