Role of Momentum in Byzantine Resilience

of Distributed Learning

Nirupam Gupta

EPFL

Workshop on **Principles of Distributed Learning (PODL)** at **PODC'22**


“All models are wrong, but some are useful.”
– George Box ?

Machine Learning

Machine Learning

DATA

Machine Learning

DATA

Machine Learning

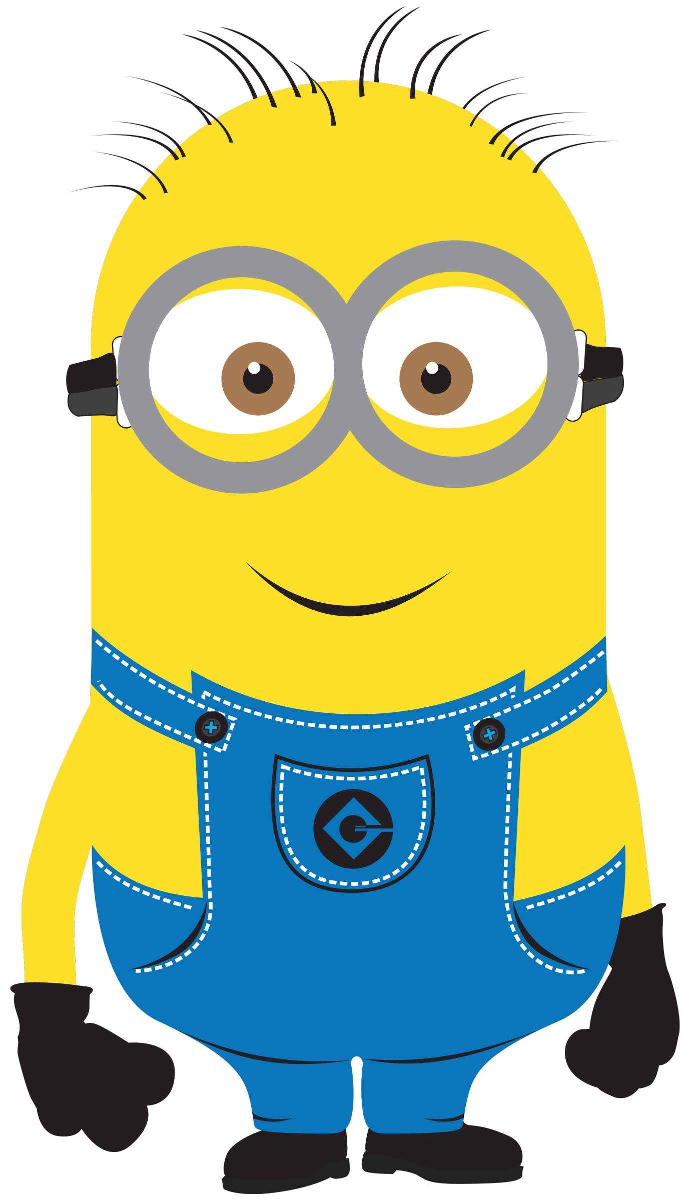
DATA

AMATEUR MACHINE

Machine Learning

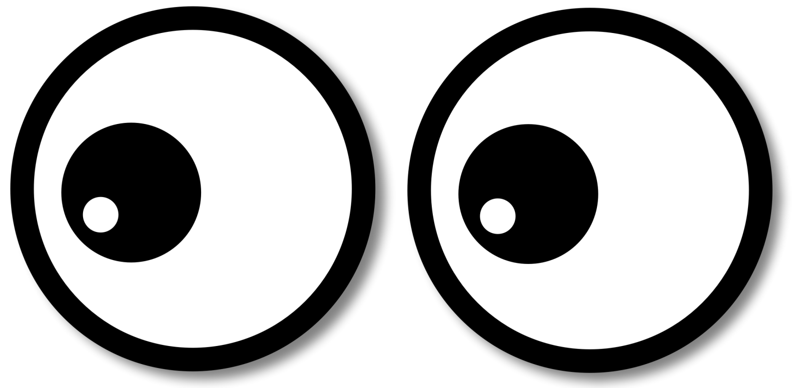
DATA

AMATEUR MACHINE

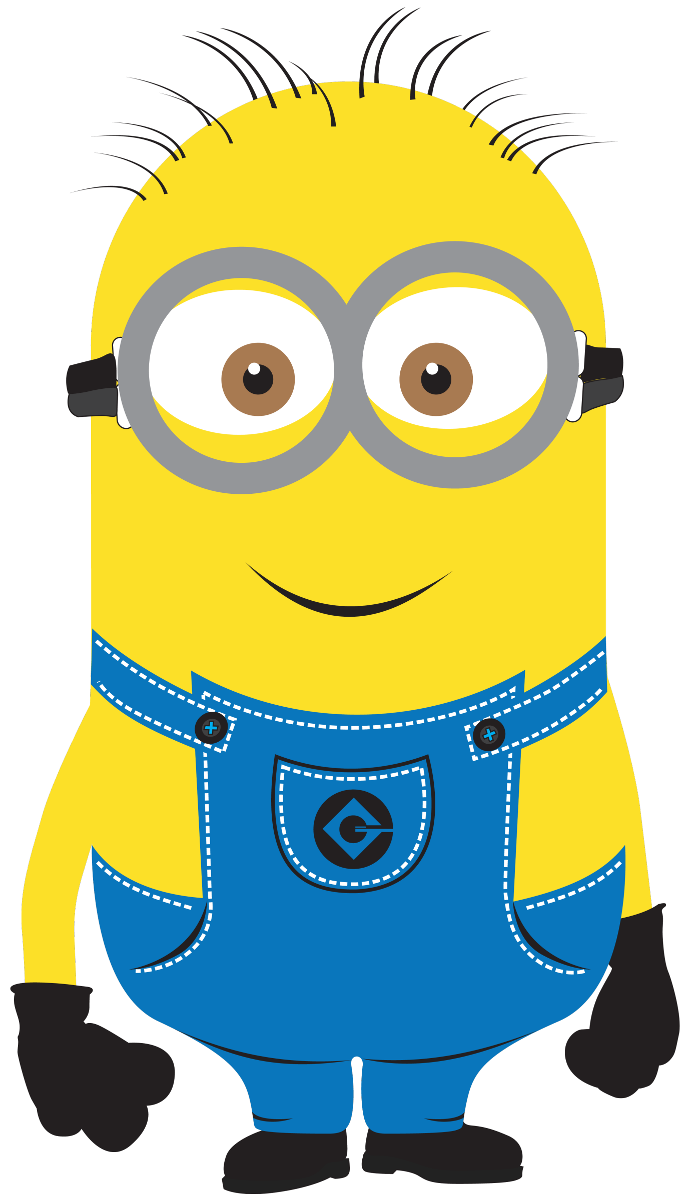


Machine Learning

DATA



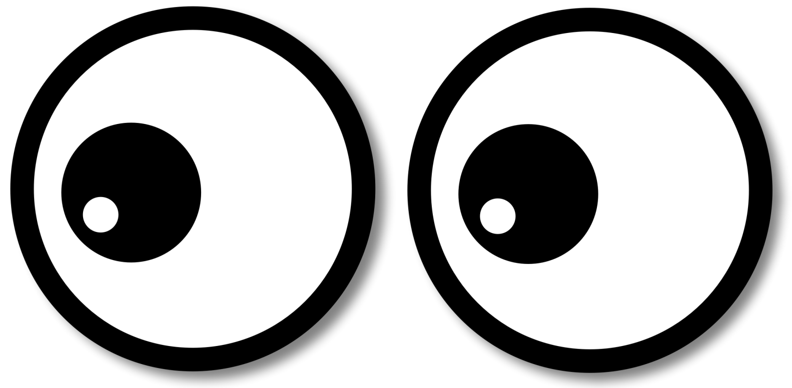
AMATEUR MACHINE



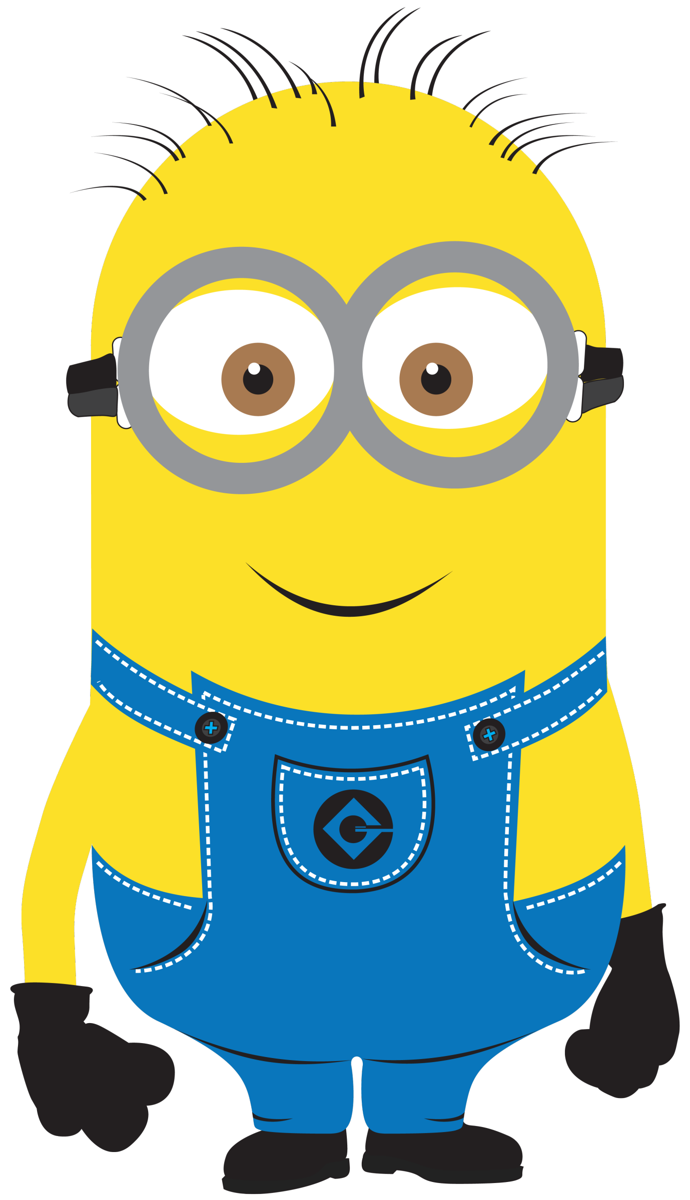
Machine Learning

DATA

WATCH & LEARN



AMATEUR MACHINE

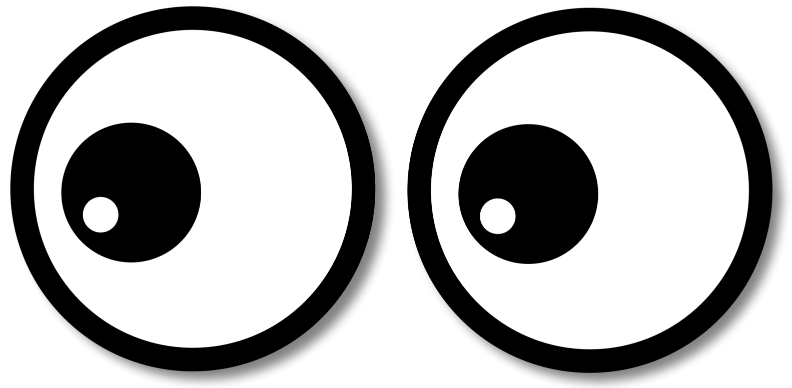


Machine Learning

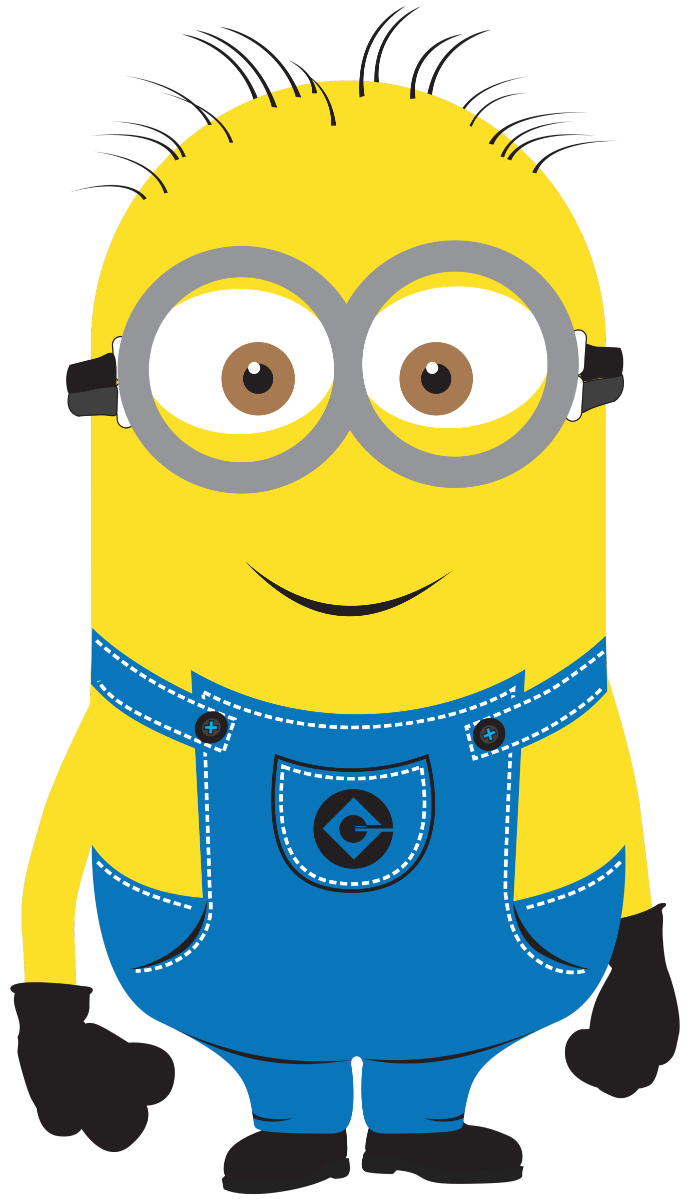
DATA

$\mathcal{D}$

WATCH & LEARN



AMATEUR MACHINE

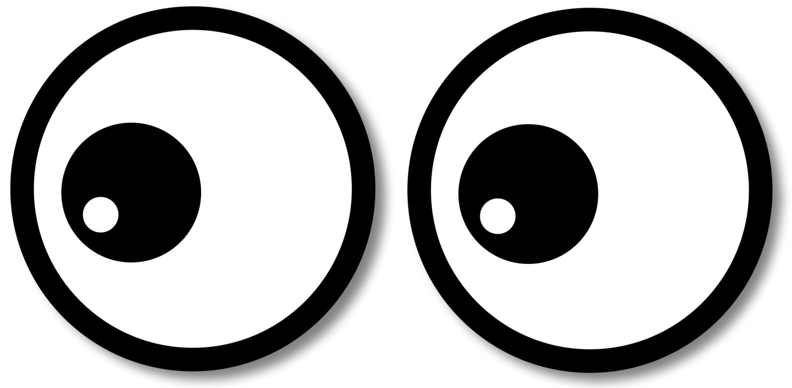


Machine Learning

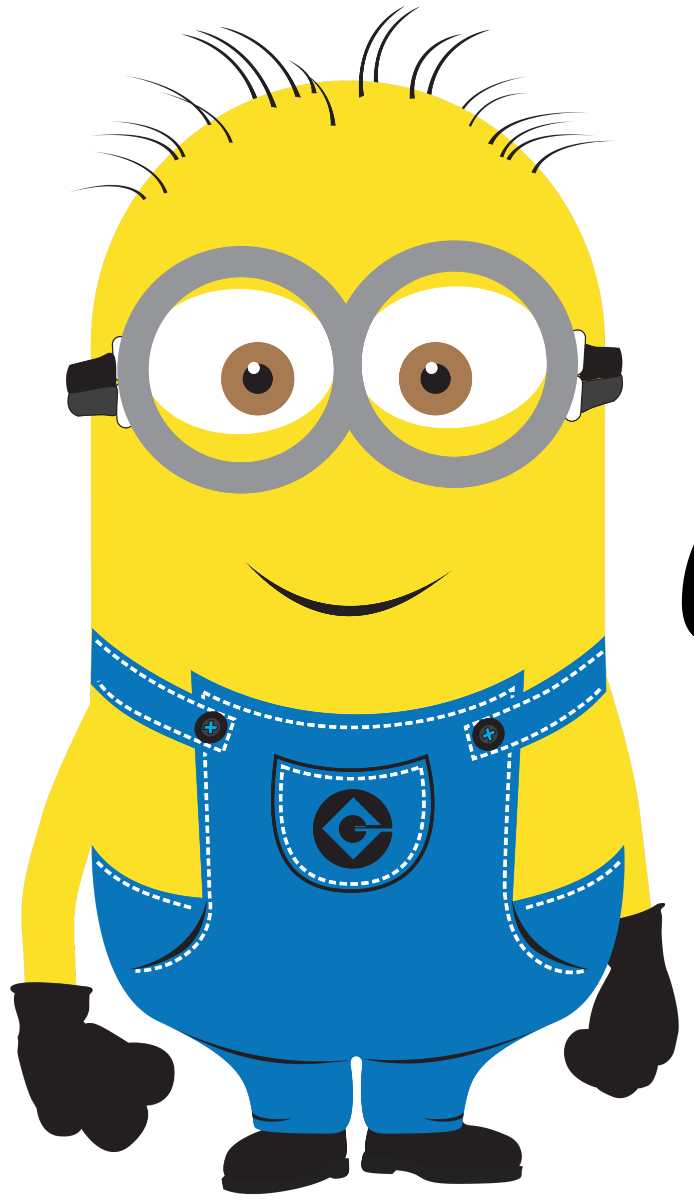
DATA

$\mathscr{D}$

WATCH & LEARN



AMATEUR MACHINE



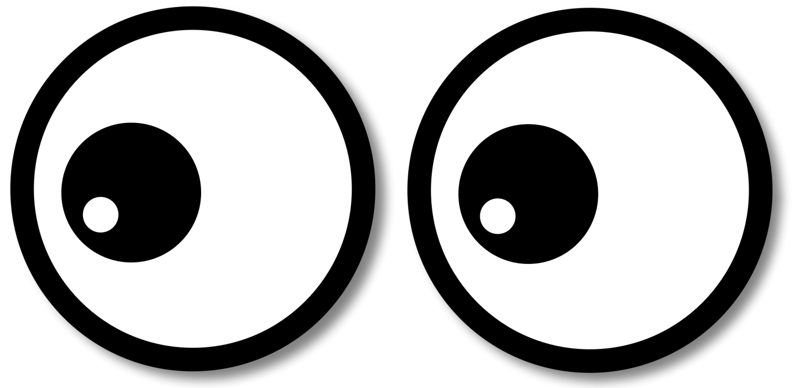
θ

Machine Learning

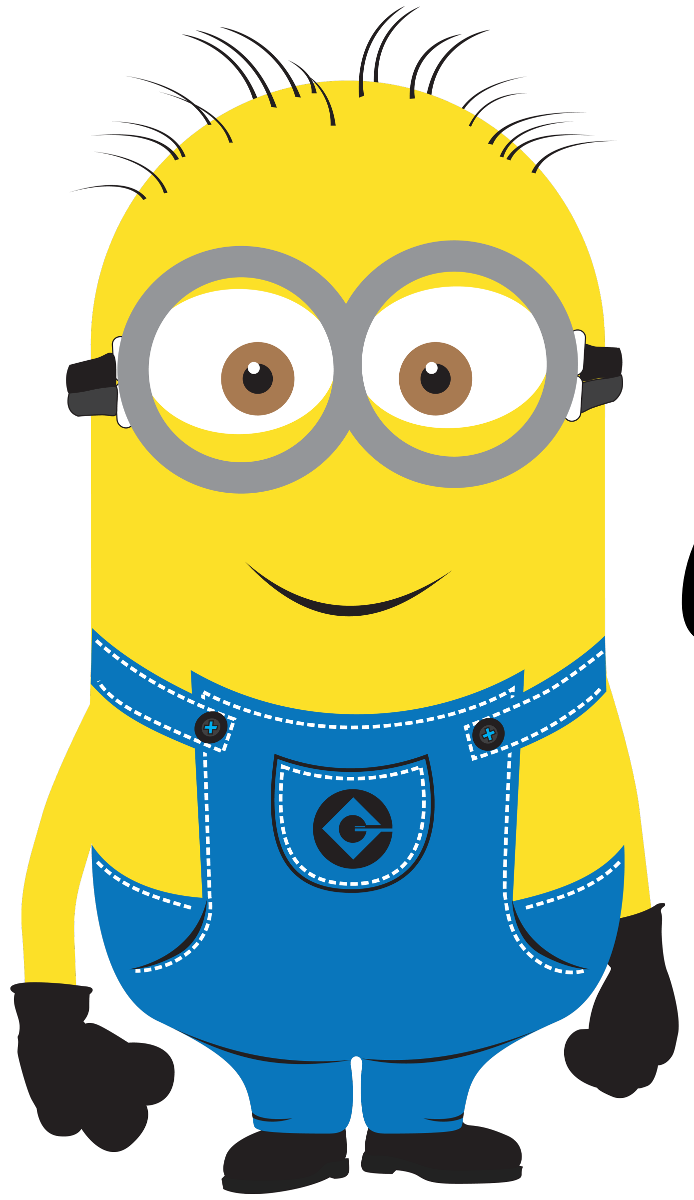
DATA

$\mathscr{D}$

WATCH & LEARN



AMATEUR MACHINE



θ

SOLVE

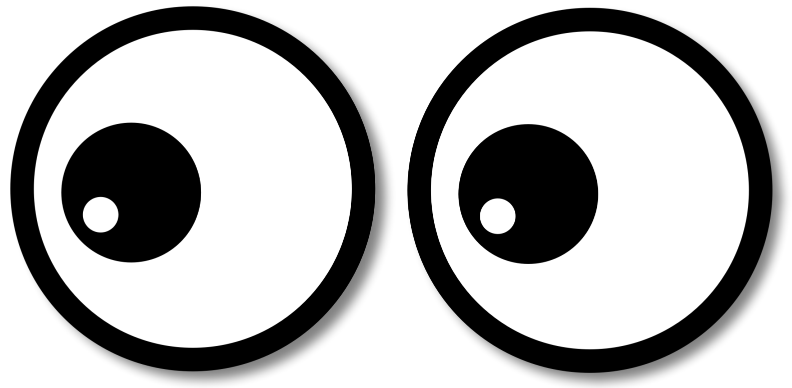
$$\theta^* \leftarrow \arg\min_{\theta} Q(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}} q(\theta, x)$$

Machine Learning

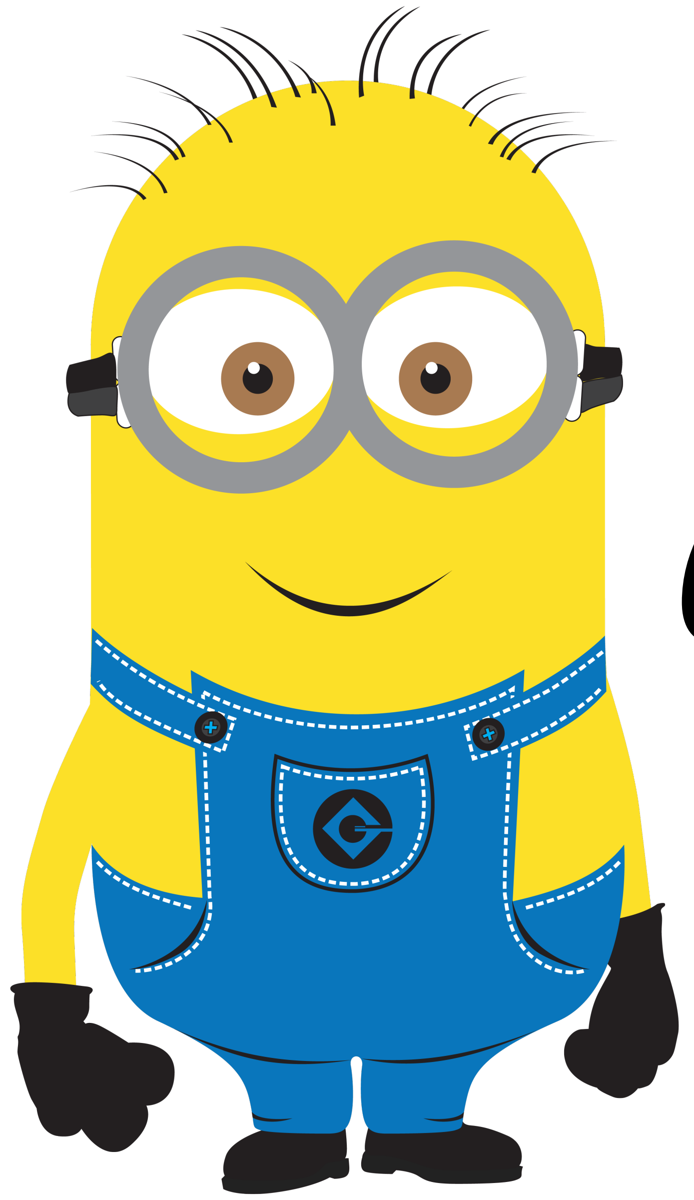
DATA

$\mathscr{D}$

WATCH & LEARN



AMATEUR MACHINE



θ

SOLVE

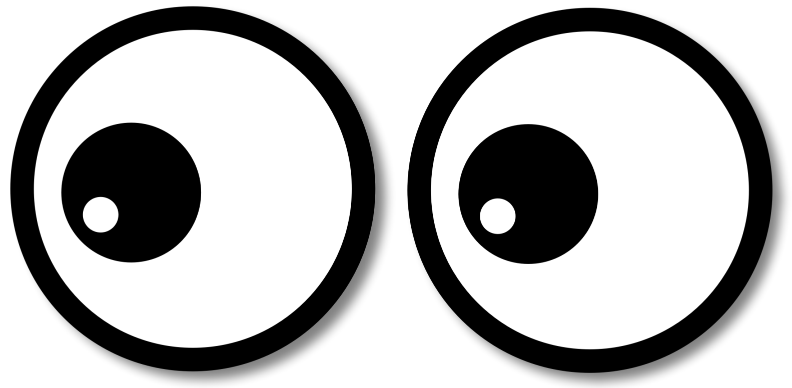
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}} q\,(\theta, x)$$

Machine Learning

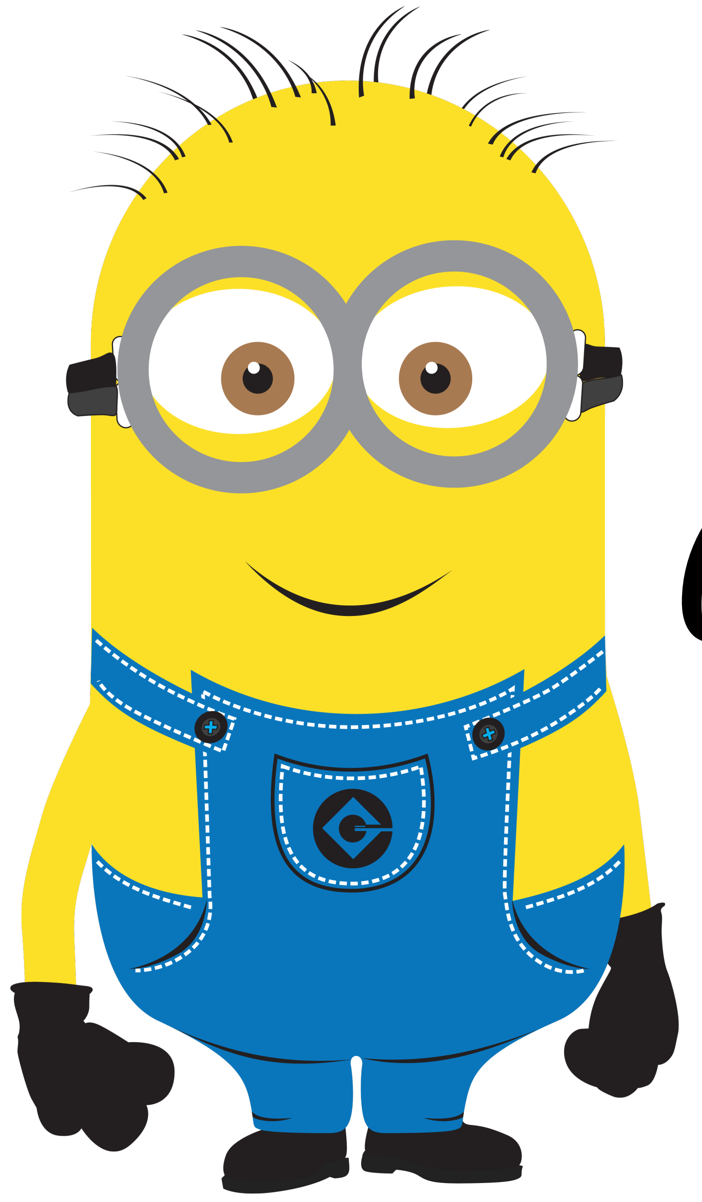
DATA

$\mathscr{D}$

WATCH & LEARN



AMATEUR MACHINE



θ

SOLVE

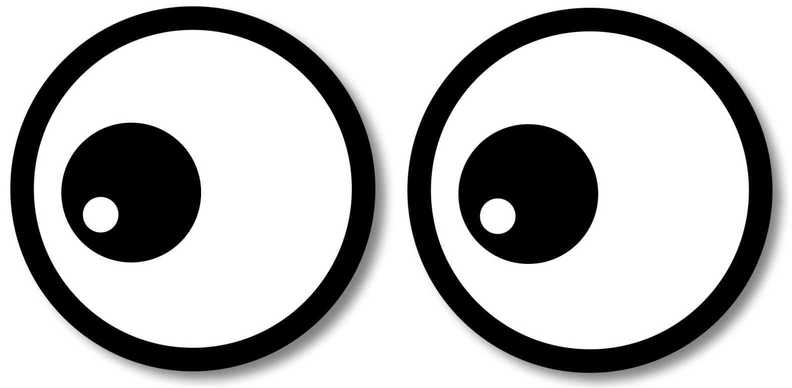
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}} q(\theta, x)$$

Machine Learning

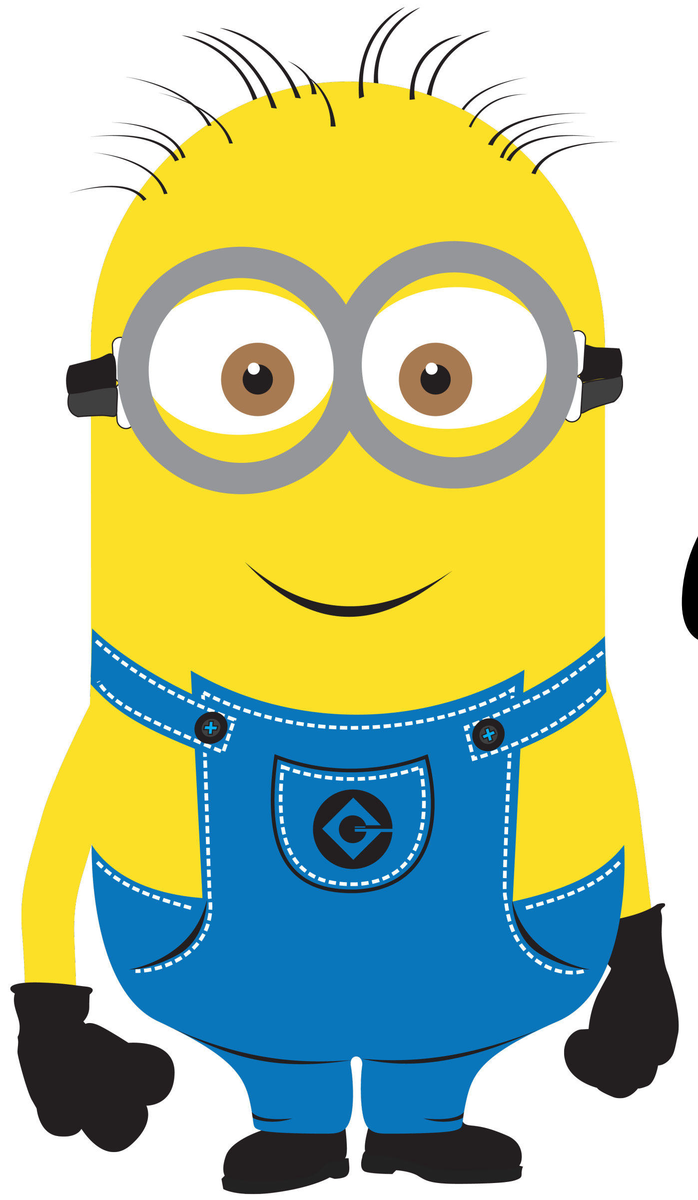
DATA

$\mathscr{D}$

WATCH & LEARN



AMATEUR MACHINE



θ

Loss per data-point

SOLVE

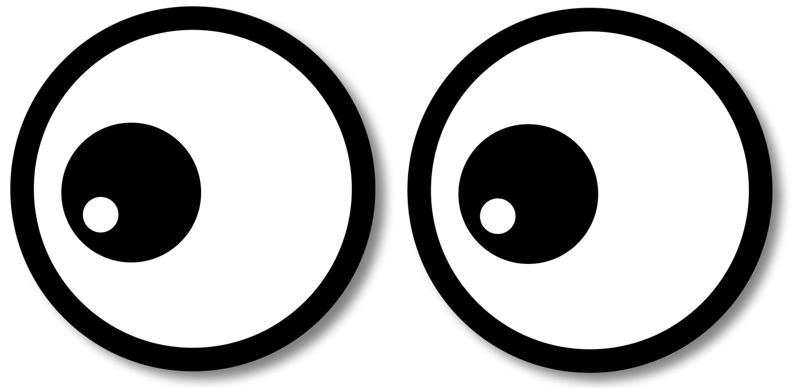
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathbb{E}_{x\sim\mathscr{D}} \, q\,(\theta, x)$$

Machine Learning

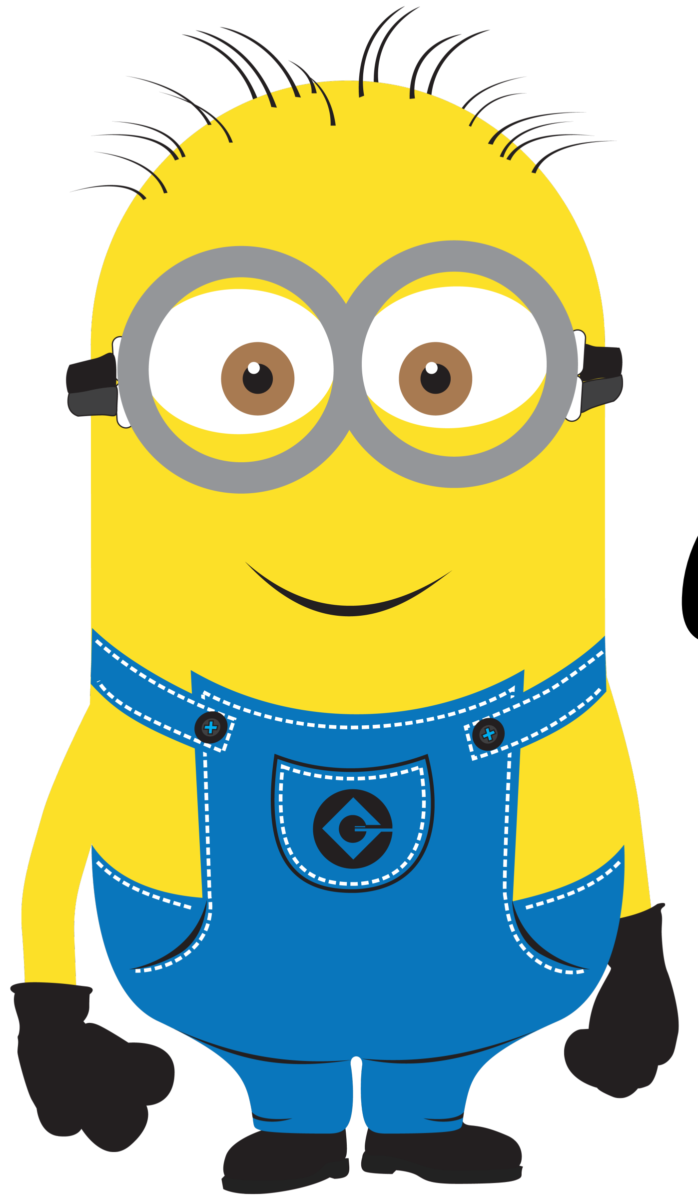
DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE



θ

Loss per data-point

SOLVE

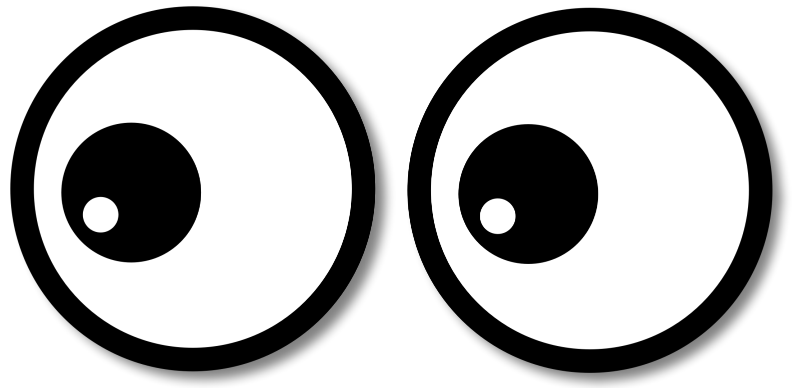
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathbb{E}_{x \sim \mathscr{D}} \, q\,(\theta, x)$$

Machine Learning

DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE

Loss per data-point

SOLVE

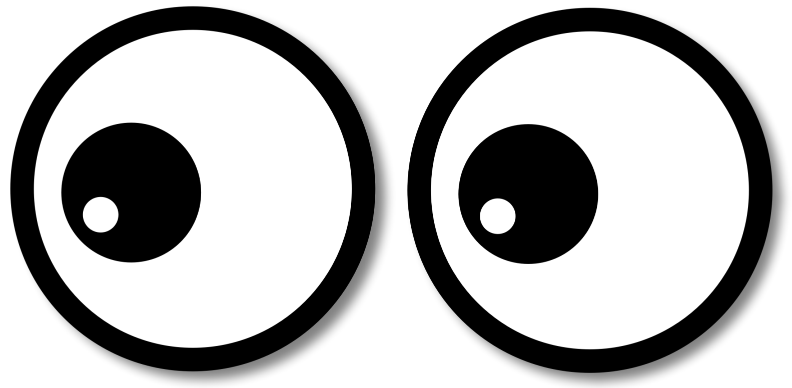
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}} q(\theta, x)$$

Machine Learning

DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE

θ^*

Loss per data-point

SOLVE

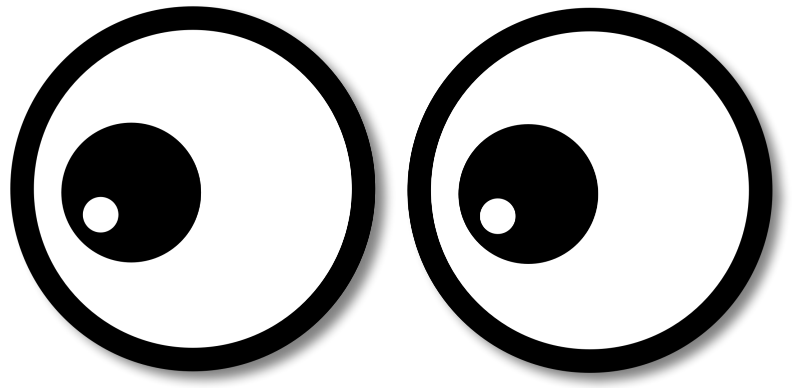
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathop{\mathbb{E}}_{x\sim\mathscr{D}} q\,(\theta, x)$$

Machine Learning

DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE

θ^*

Loss per data-point

SOLVE

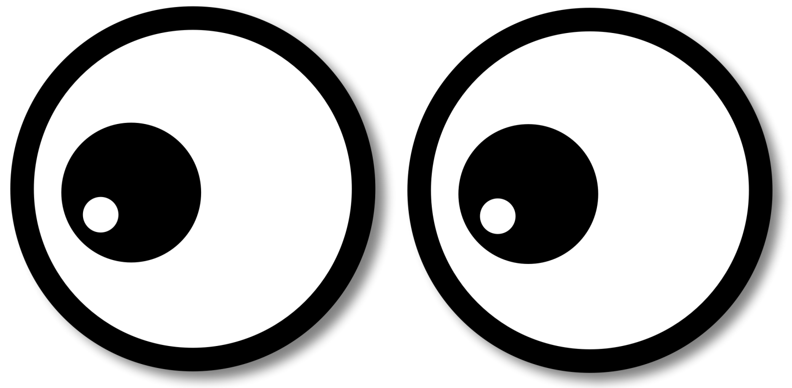
$$\theta^* \leftarrow \arg\min_{\theta} \; Q(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}} q(\theta, x)$$

Machine Learning

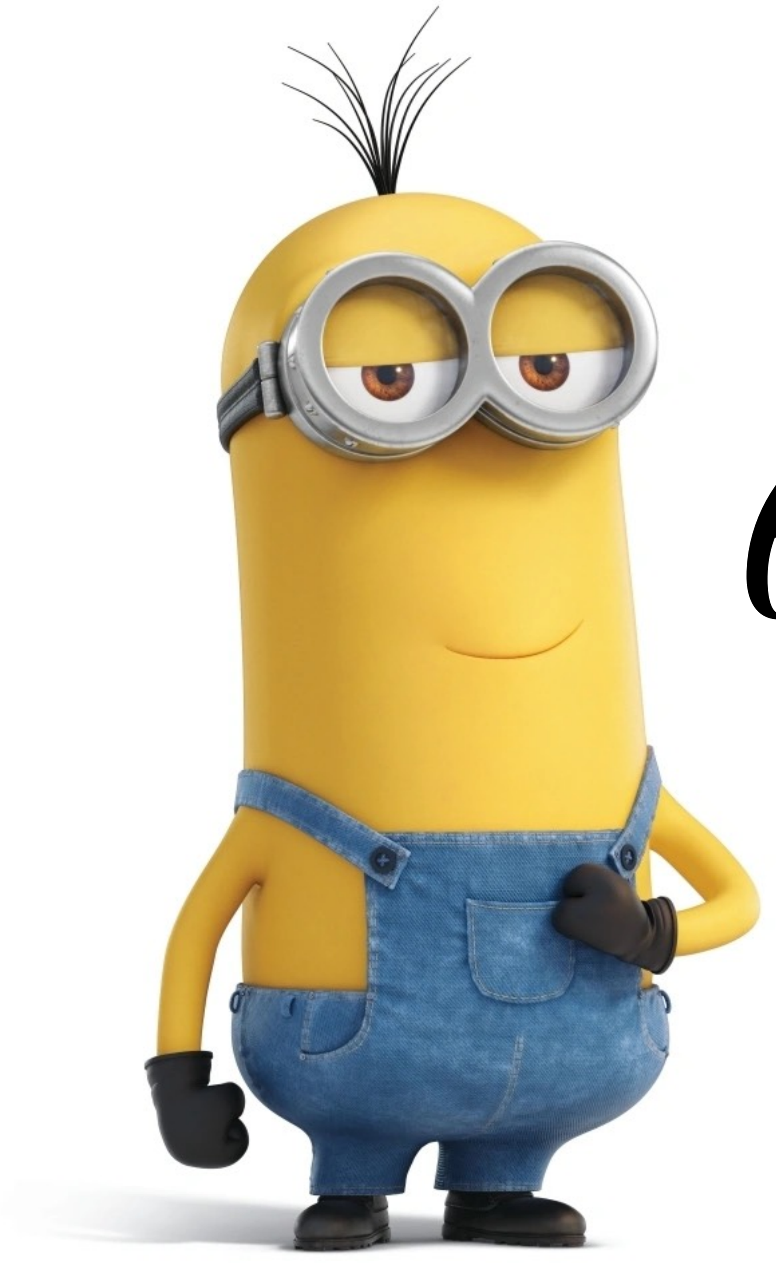
DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE



θ^*

Loss per data-point

SOLVE

$$\theta^* \leftarrow \arg\min_{\theta} Q(\theta) := \mathbb{E}_{x \sim \mathscr{D}} q(\theta, x)$$

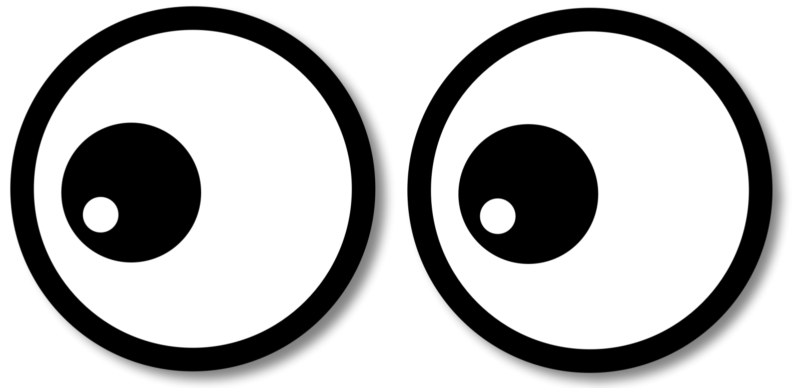
Relax

Machine Learning

DATA

$\mathscr{D}$

WATCH & LEARN



EXPERT MACHINE

θ^*

Loss per data-point

SOLVE

$$\theta^* \leftarrow \arg\min_{\theta} Q(\theta) := \mathbb{E}_{x \sim \mathscr{D}} q(\theta, x)$$

Relax

MORE REASONABLE GOAL

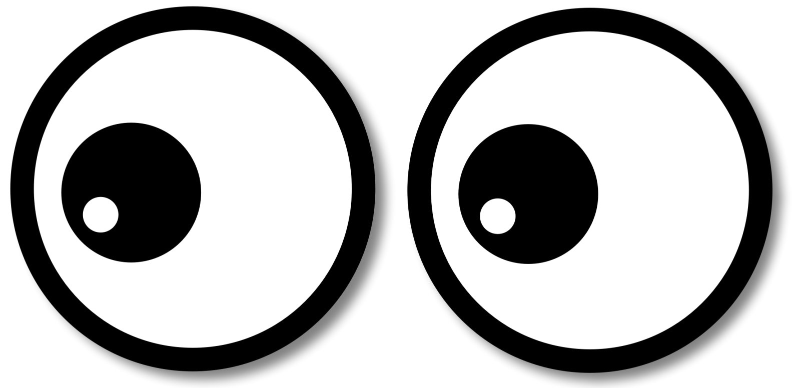
$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

Machine Learning

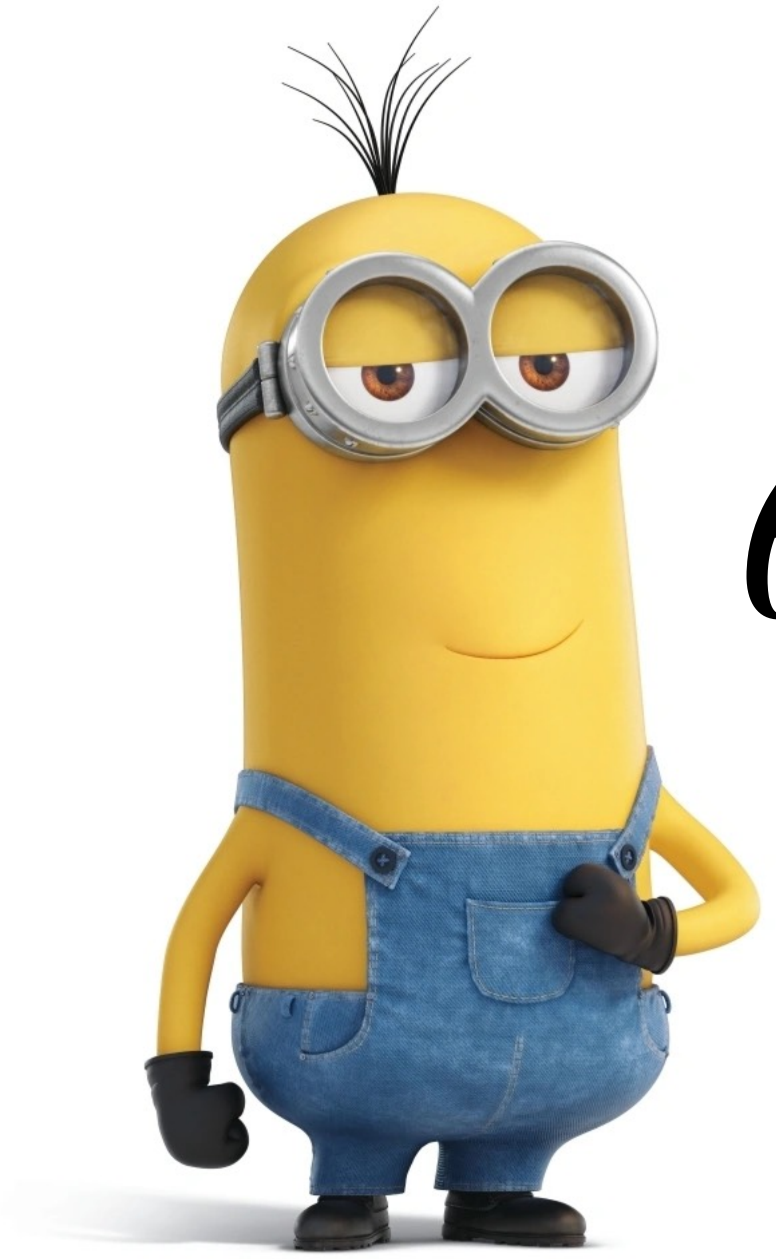
DATA

$\mathscr{D}$

WATCH & LEARN



"EXPERT" MACHINE



θ^*

Loss per data-point

SOLVE

$$\theta^* \leftarrow \arg\min_{\theta} Q(\theta) := \mathbb{E}_{x \sim \mathscr{D}}\, q(\theta, x)$$

Relax →

MORE REASONABLE GOAL

$$\theta^* \in \left(\theta \;;\; \nabla Q(\theta) = 0\right)$$

Distributed Learning

Distributed Learning

DATA IS GROWING

Distributed Learning

DATA IS GROWING

Distributed Learning

DATA IS GROWING

SO ARE THE MODELS

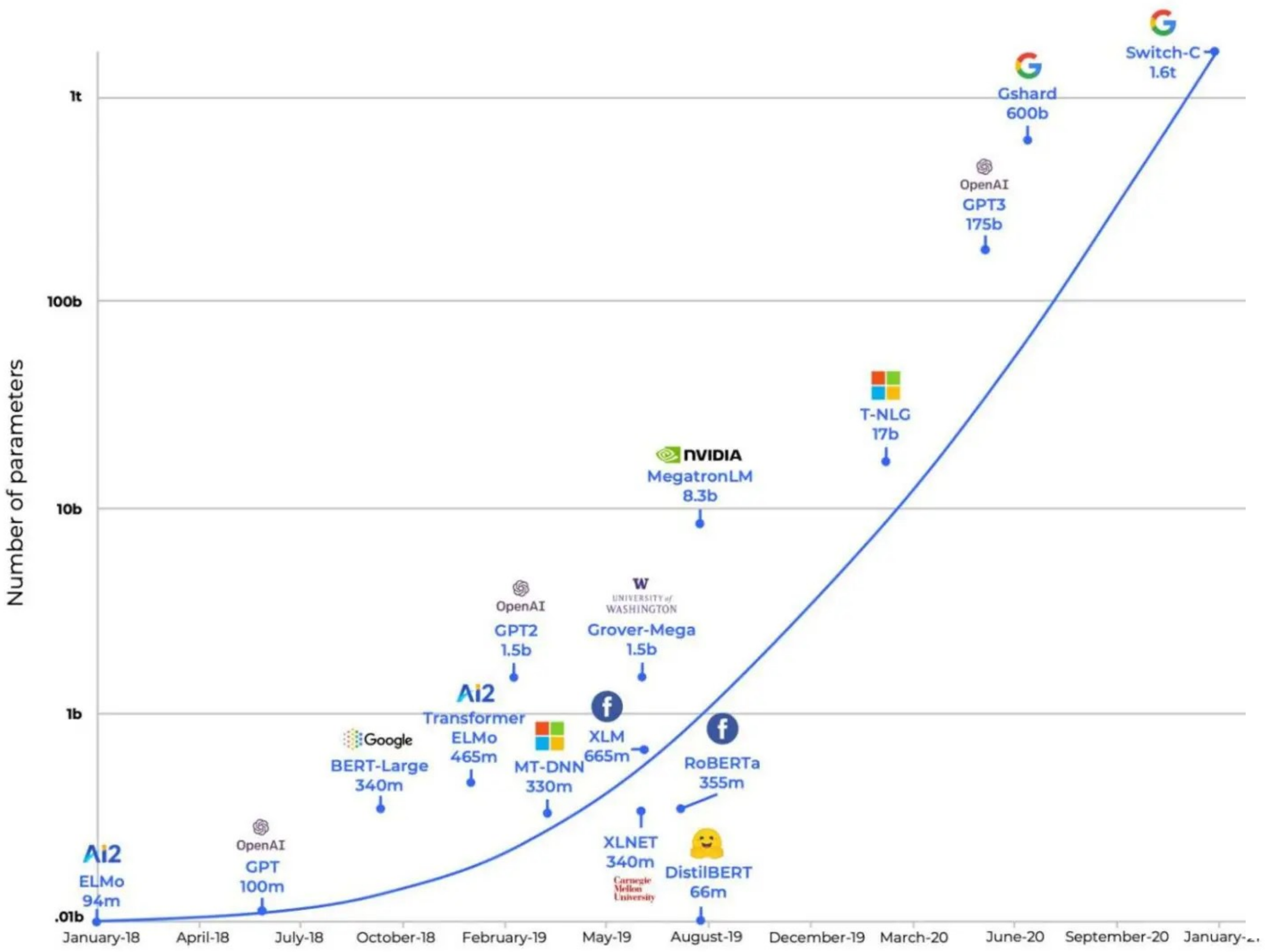
Distributed Learning

DATA IS GROWING

SO ARE THE MODELS

Distributed Learning

DATA IS GROWING

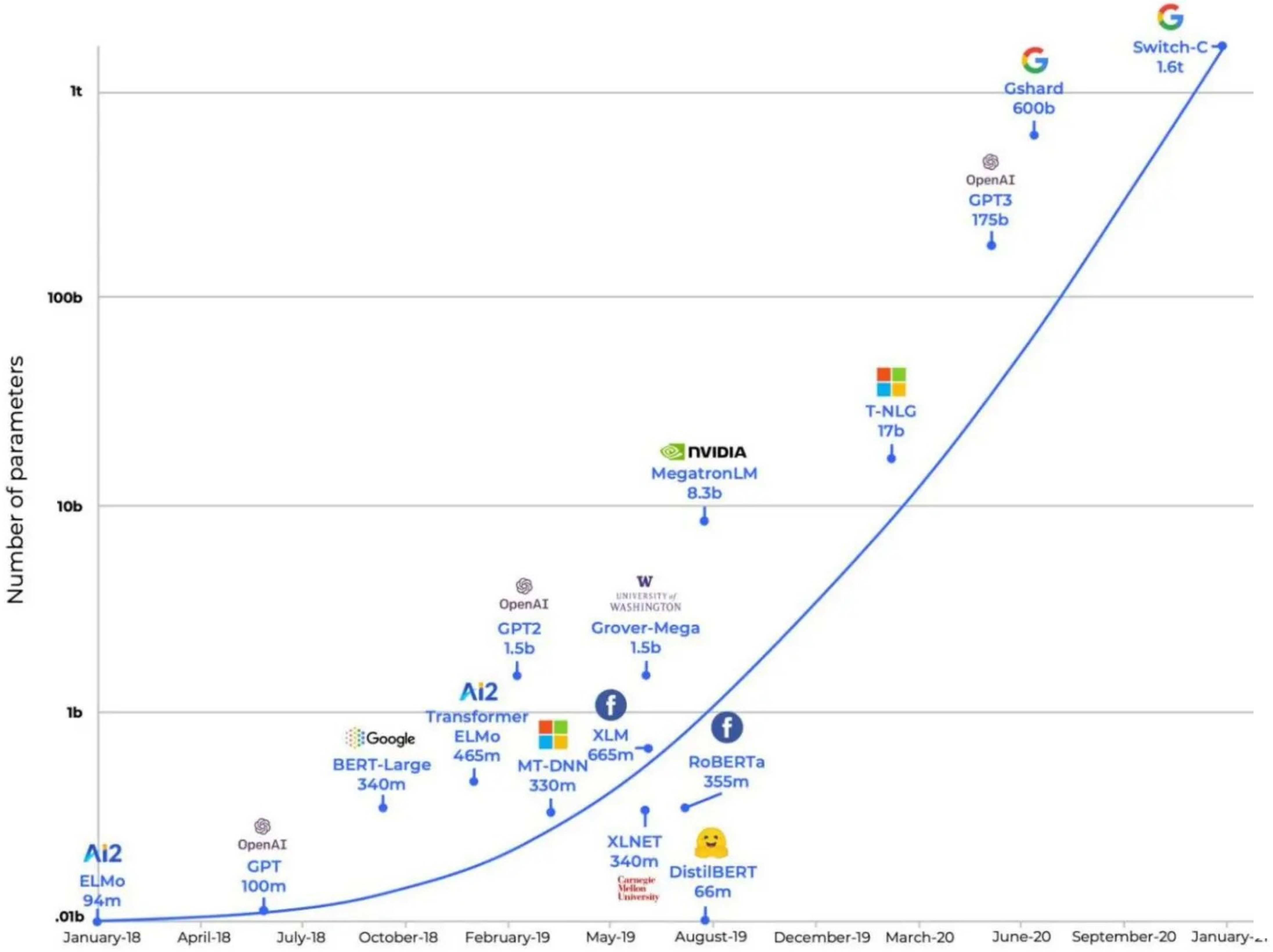


SO ARE THE MODELS

Distributed Learning

DATA IS GROWING

NEED MANY MACHINES TO TRAIN

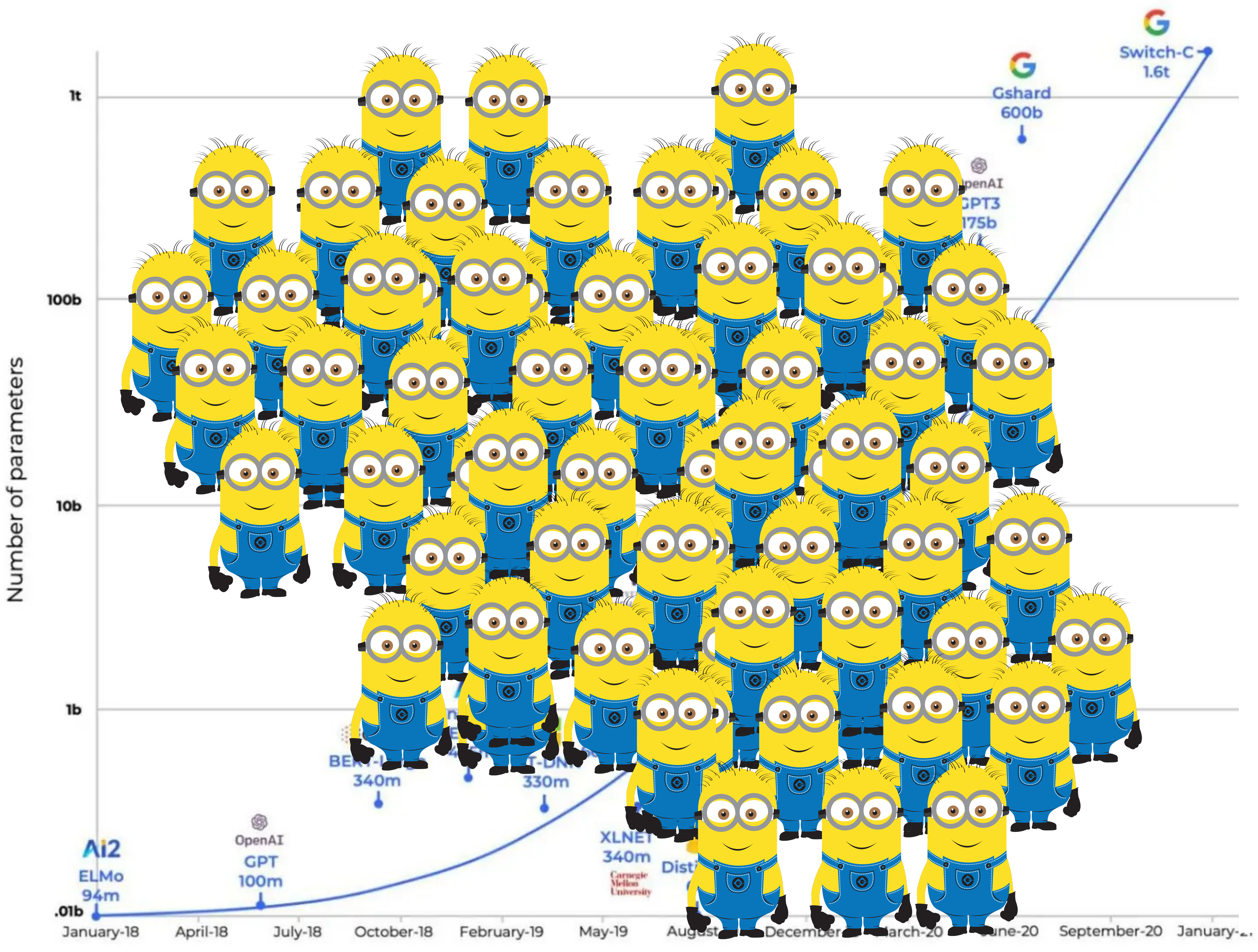


SO ARE THE MODELS

Distributed Learning

DATA IS GROWING

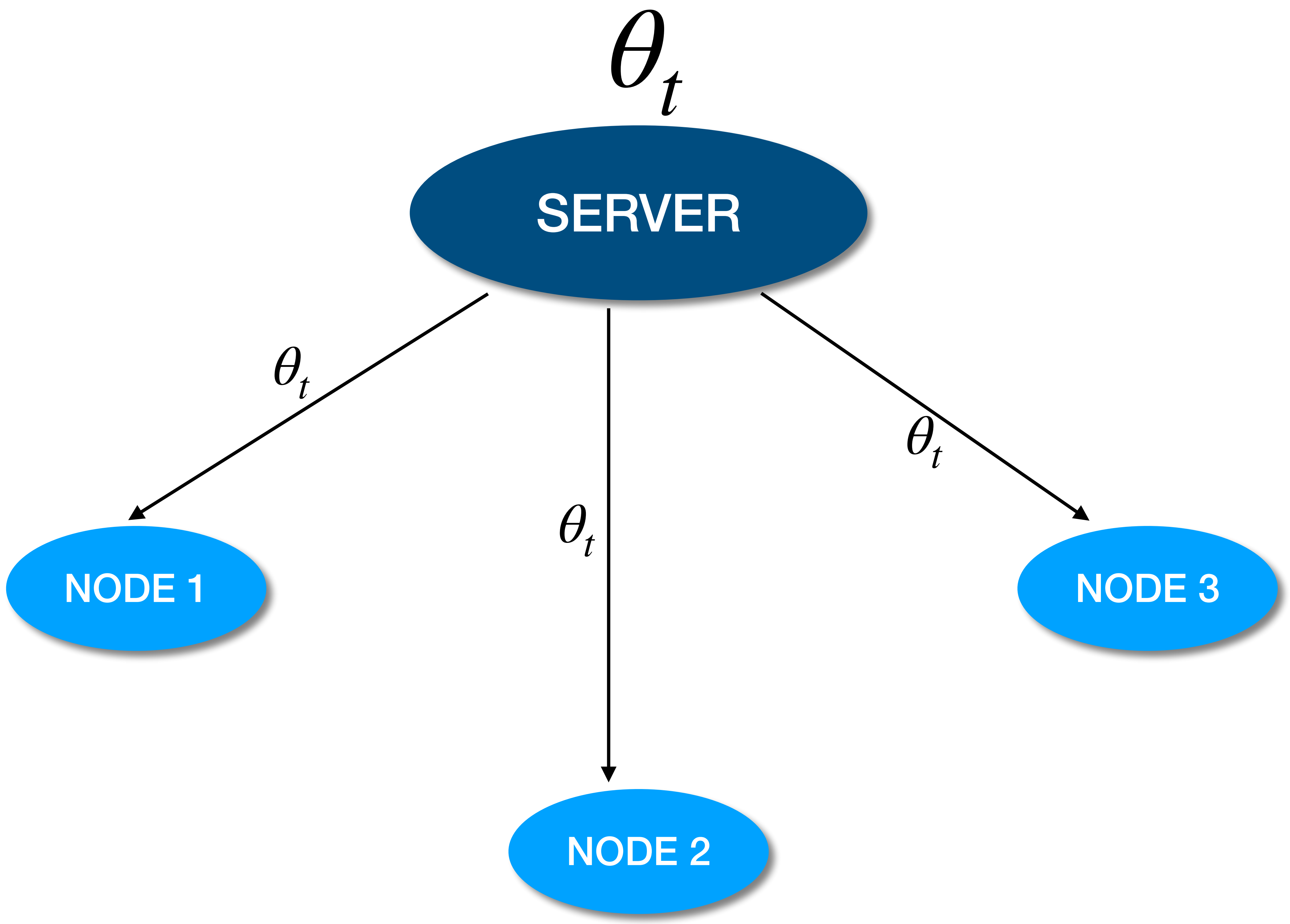
NEED MANY MACHINES TO TRAIN



SO ARE THE MODELS

Distributed SGD

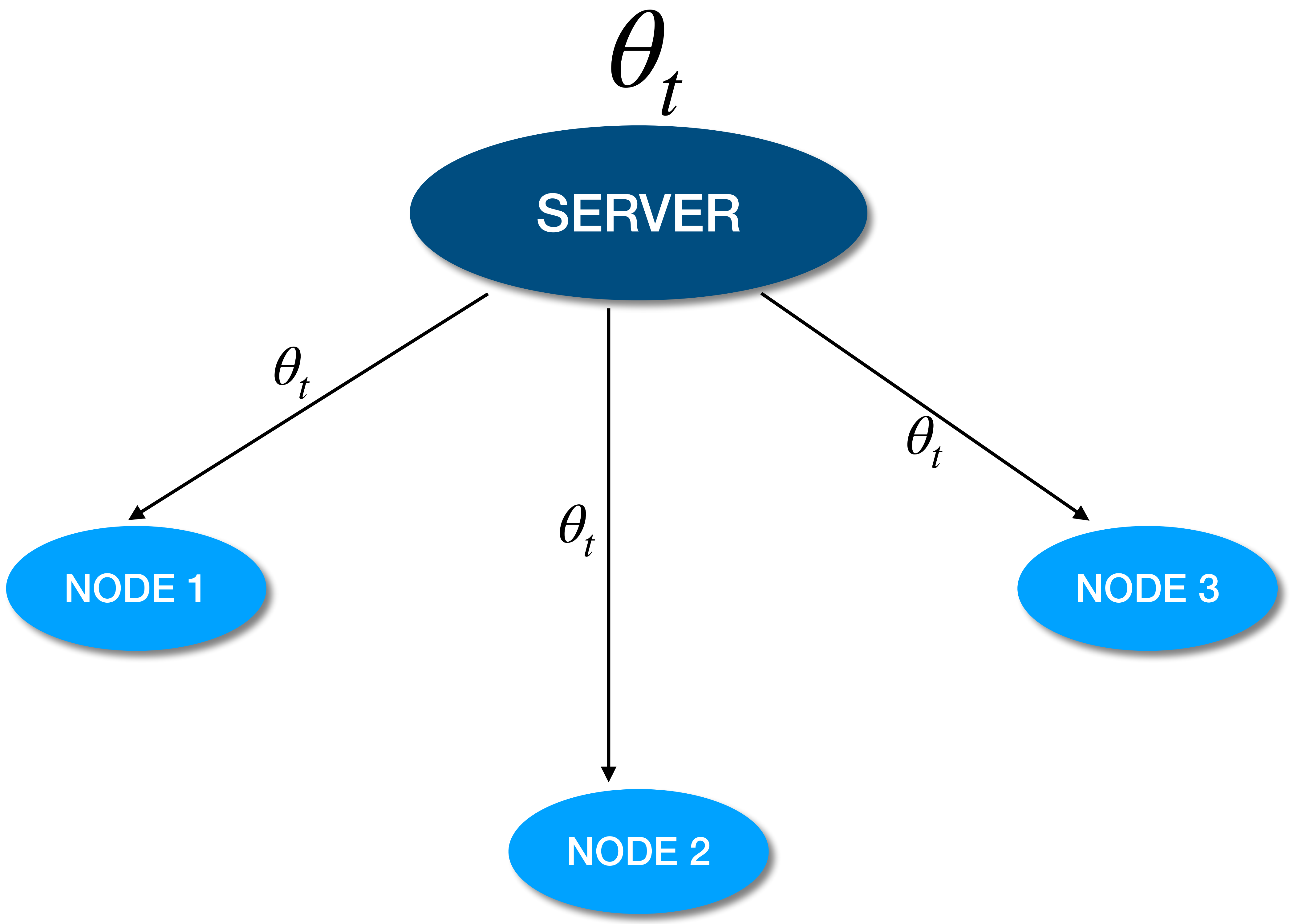
Divides the workload per machine by the total size of the system



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

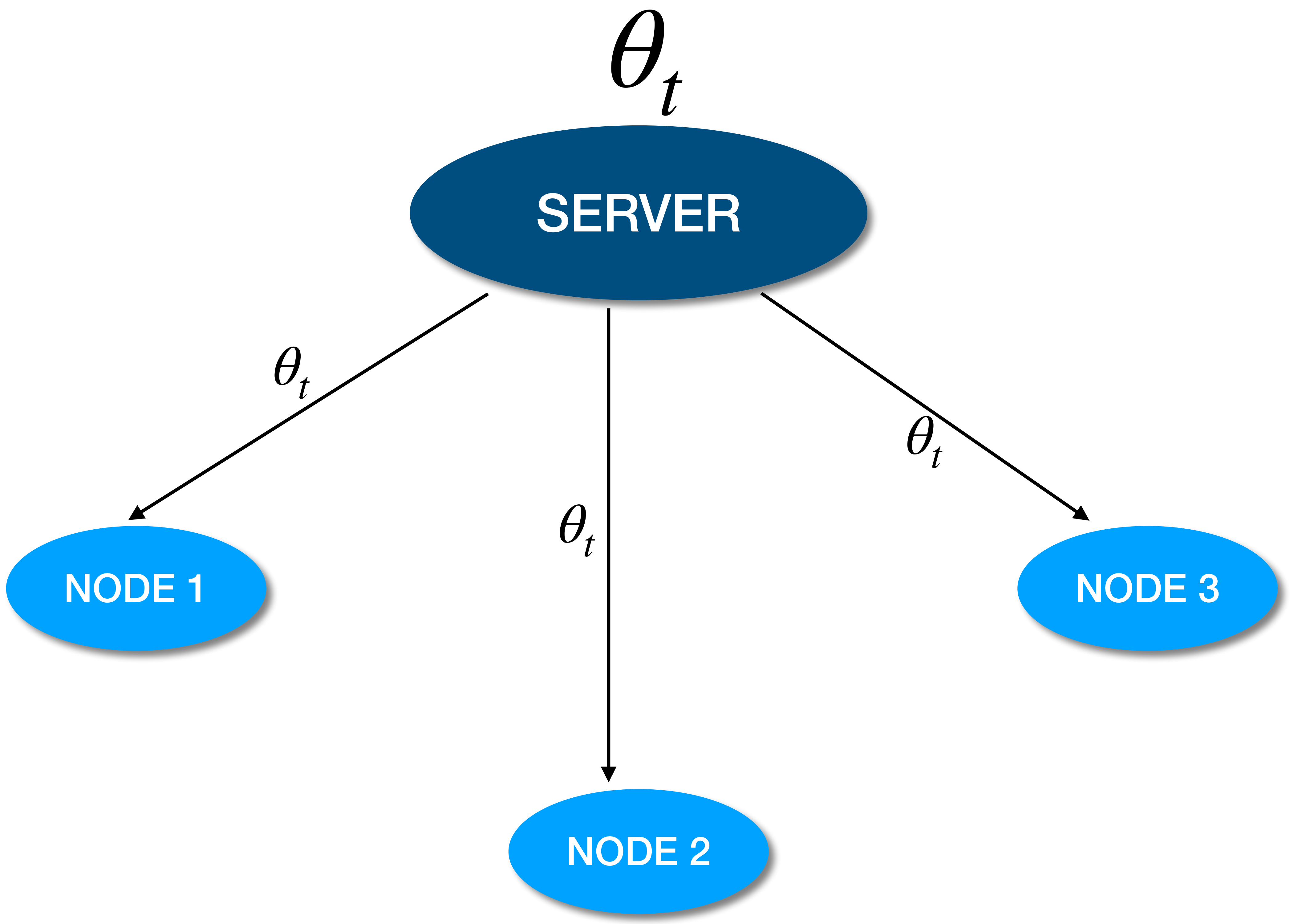


Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ;$$

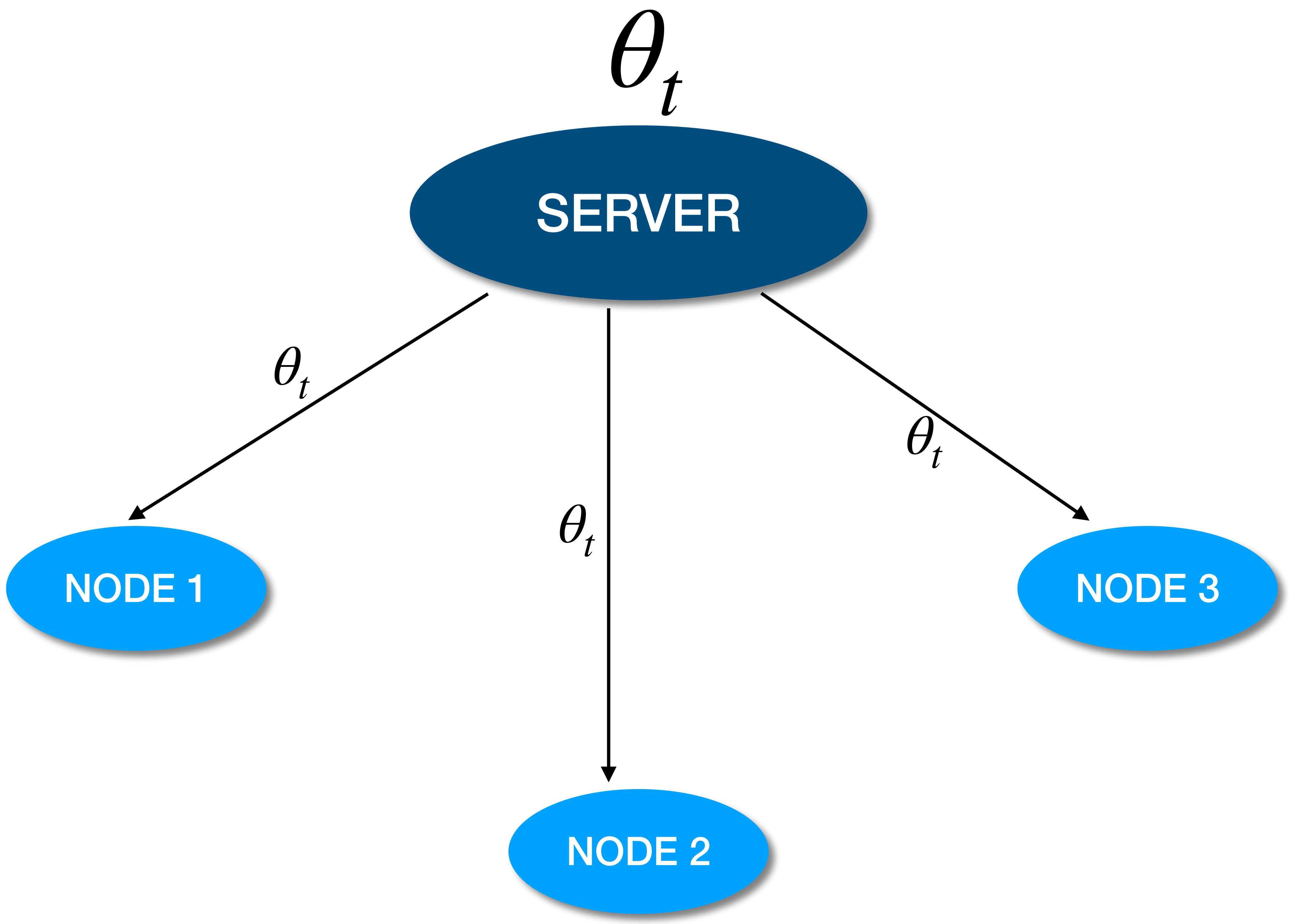


Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ;$$

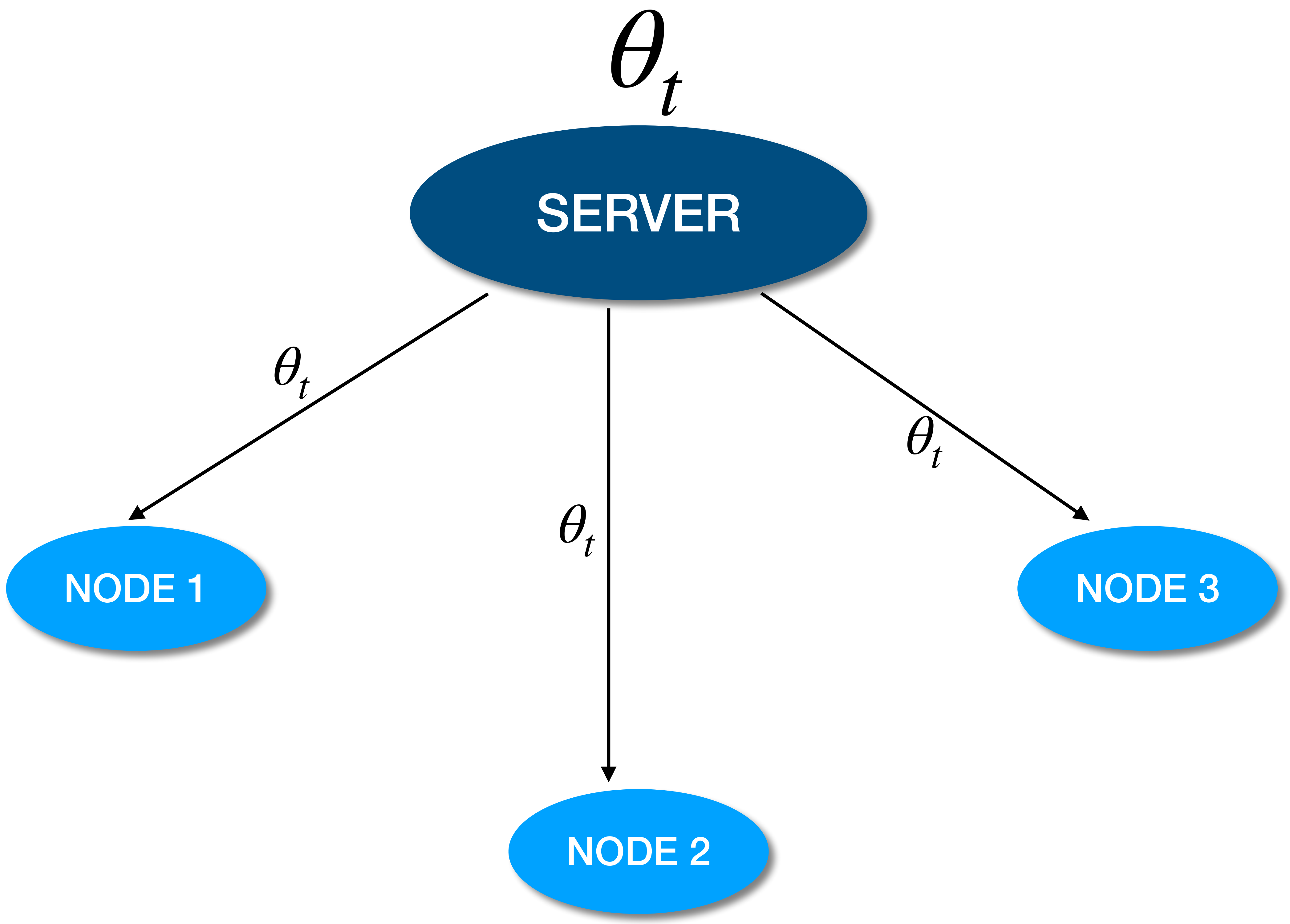


Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ;$$



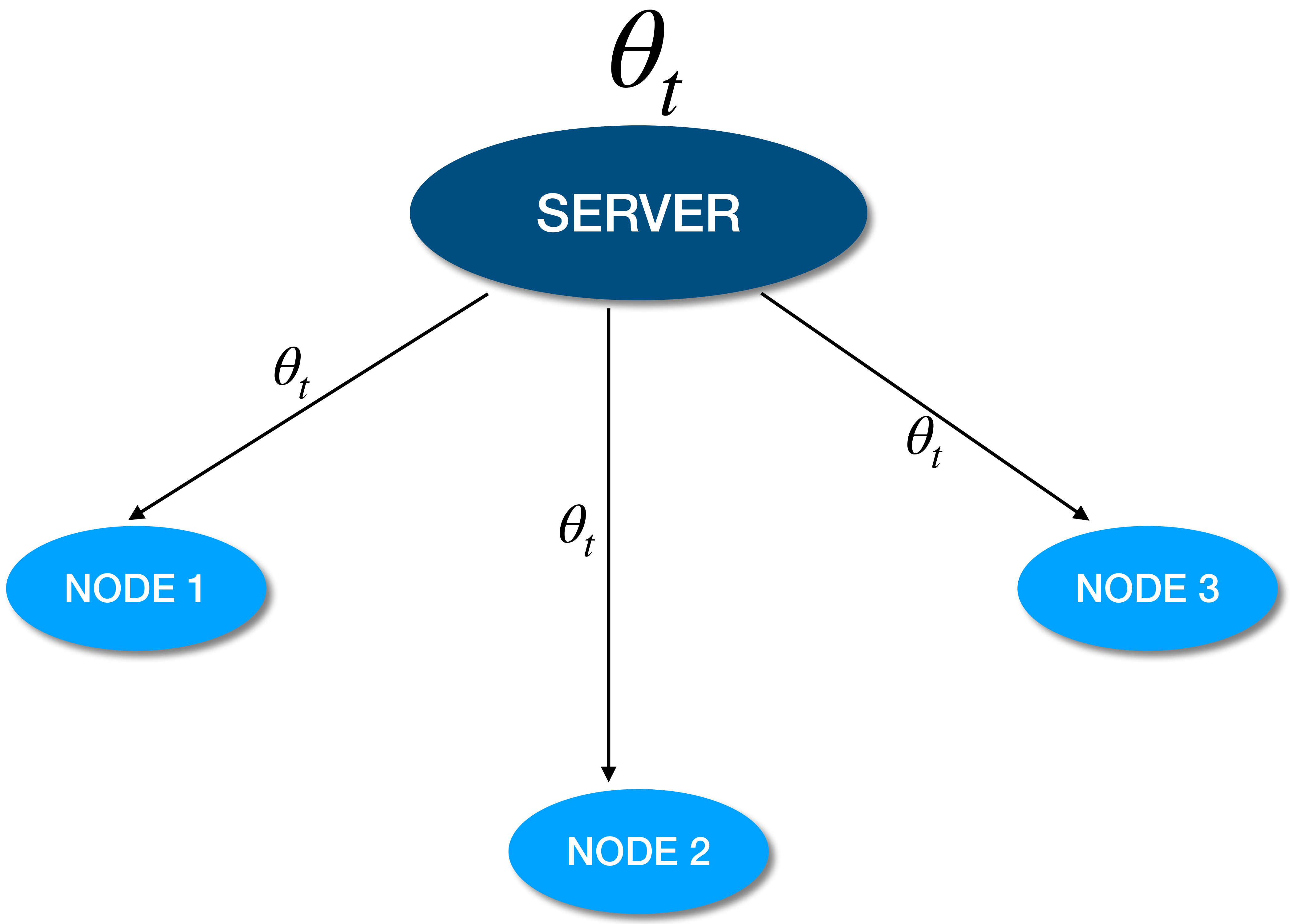
Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ;$$

True gradient



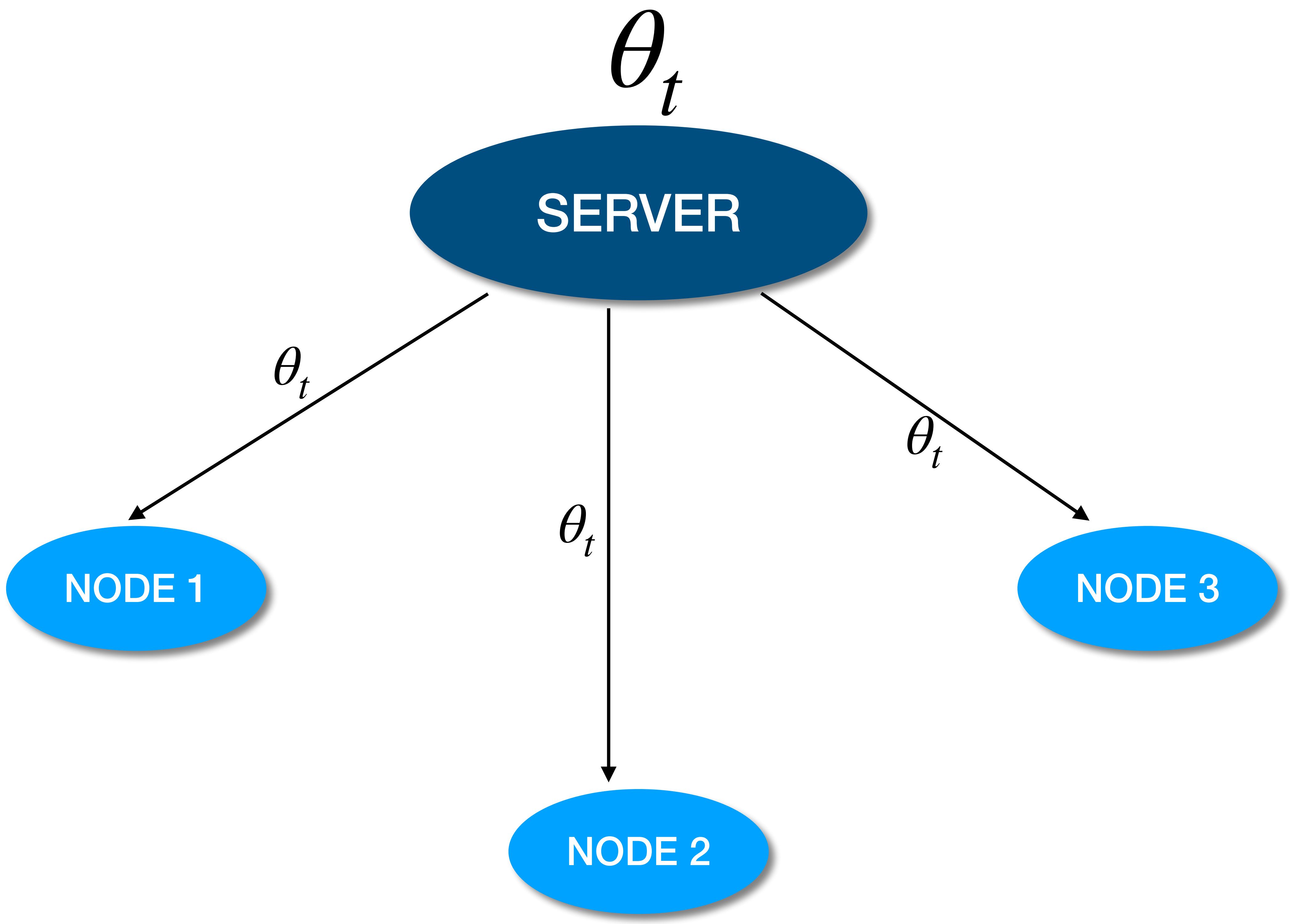
Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

True gradient



Distributed SGD

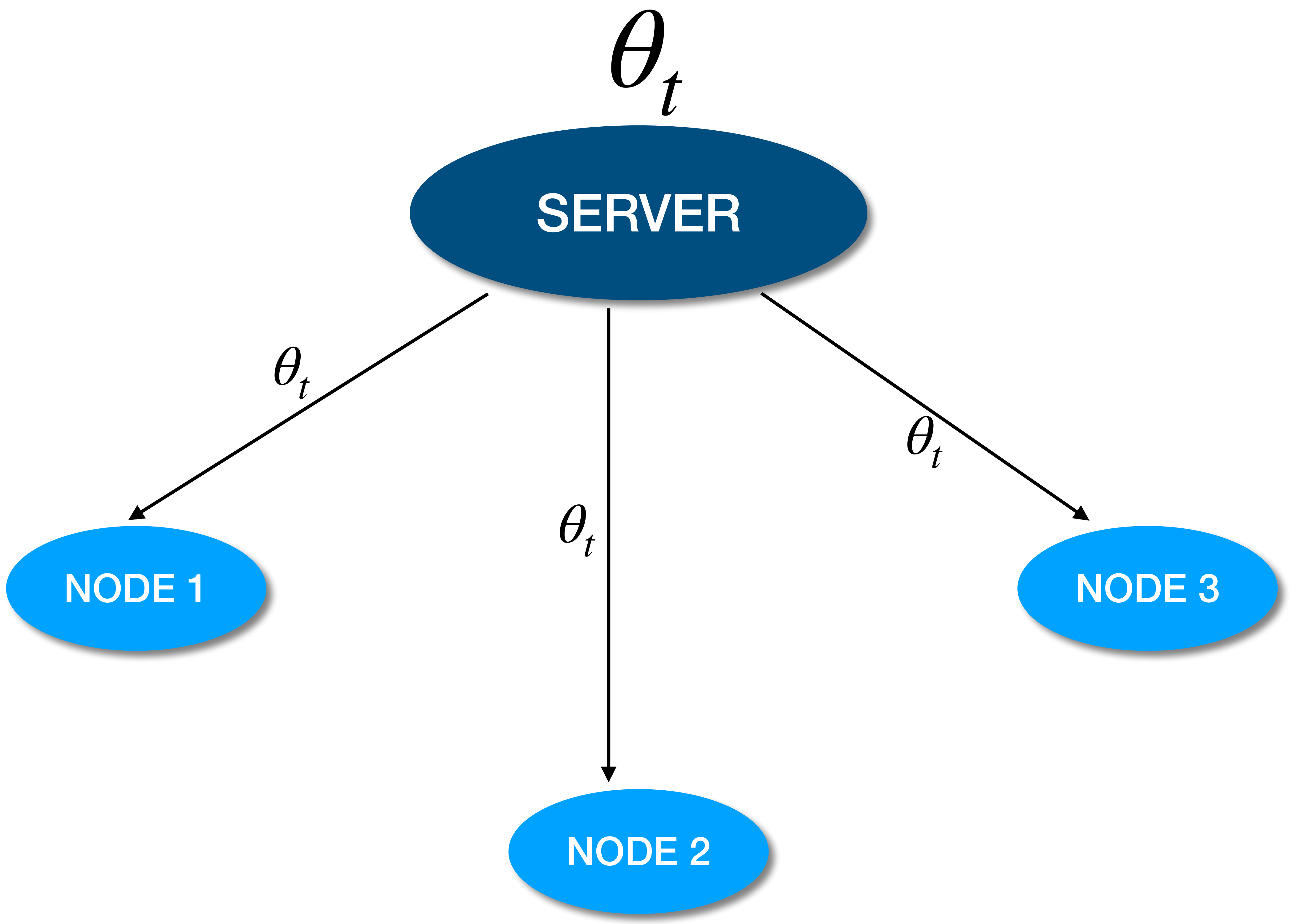
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

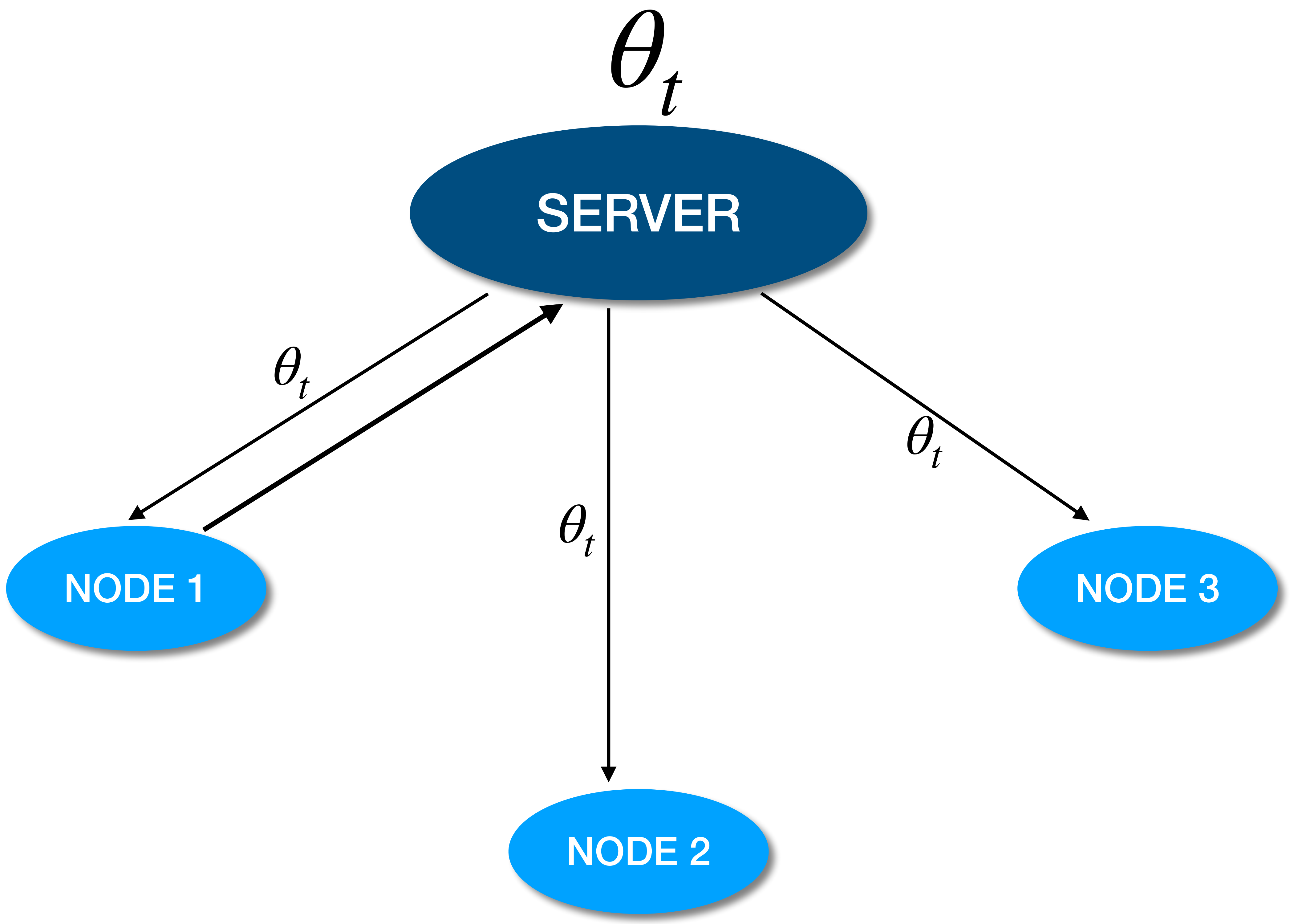
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

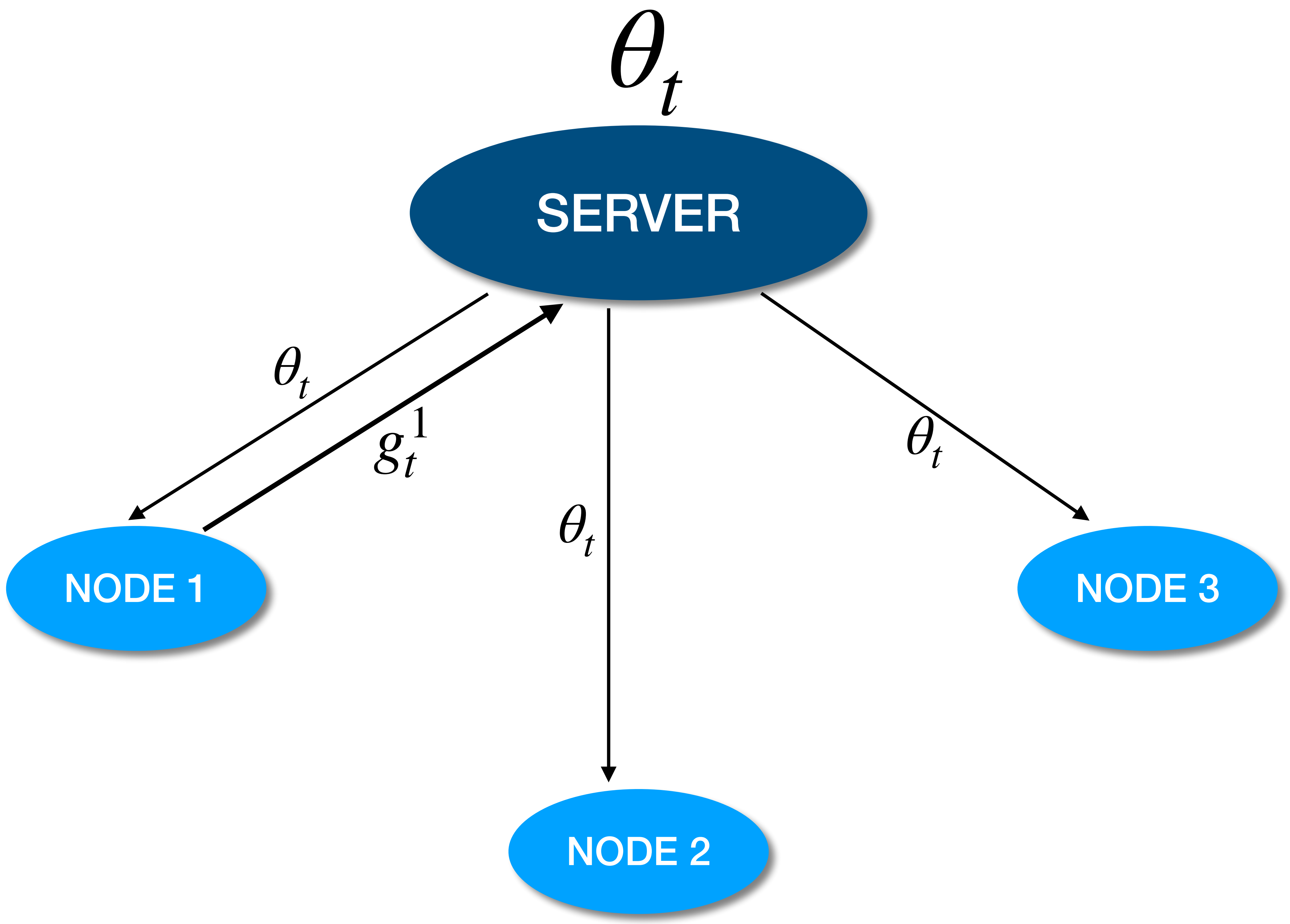
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

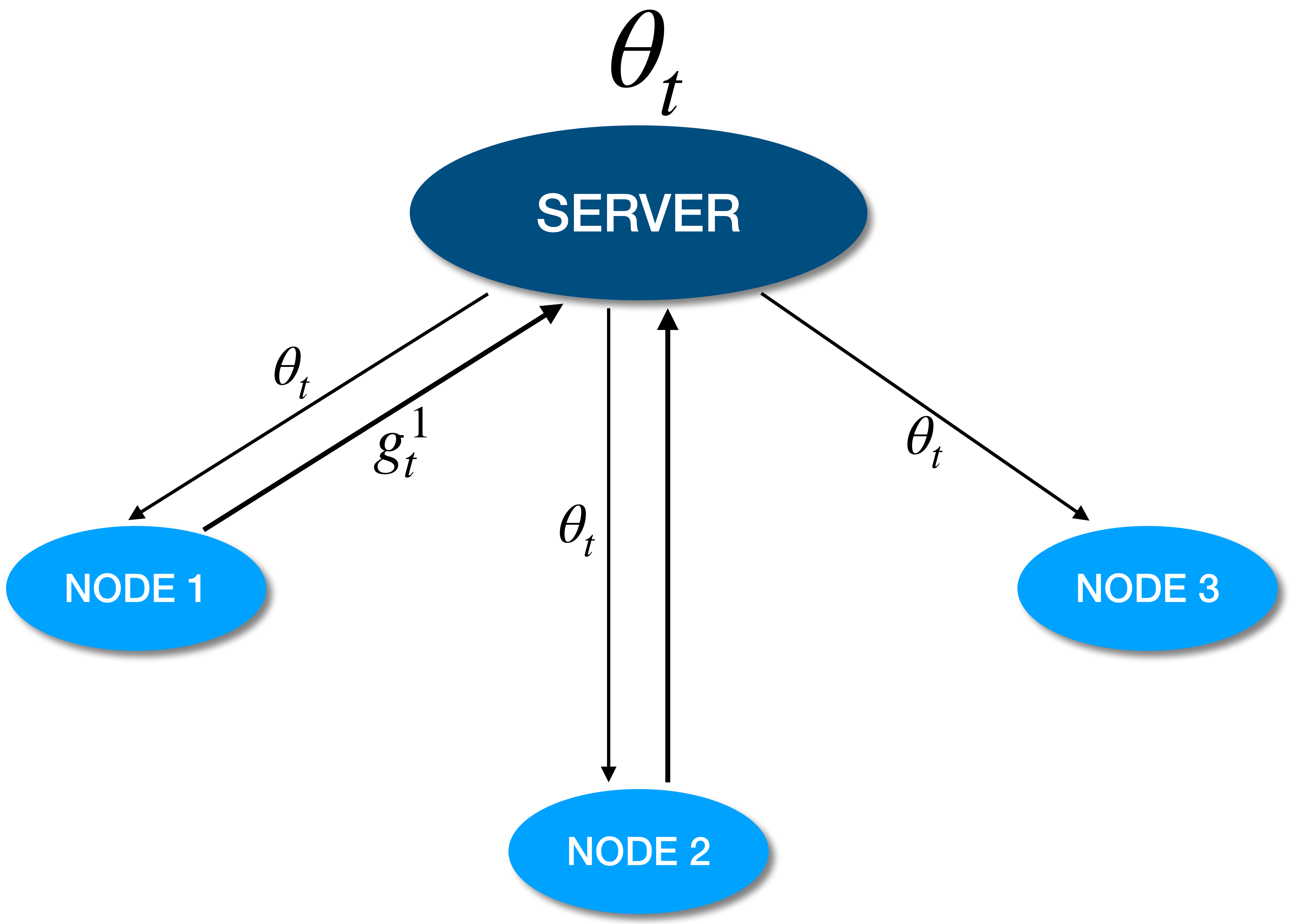
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

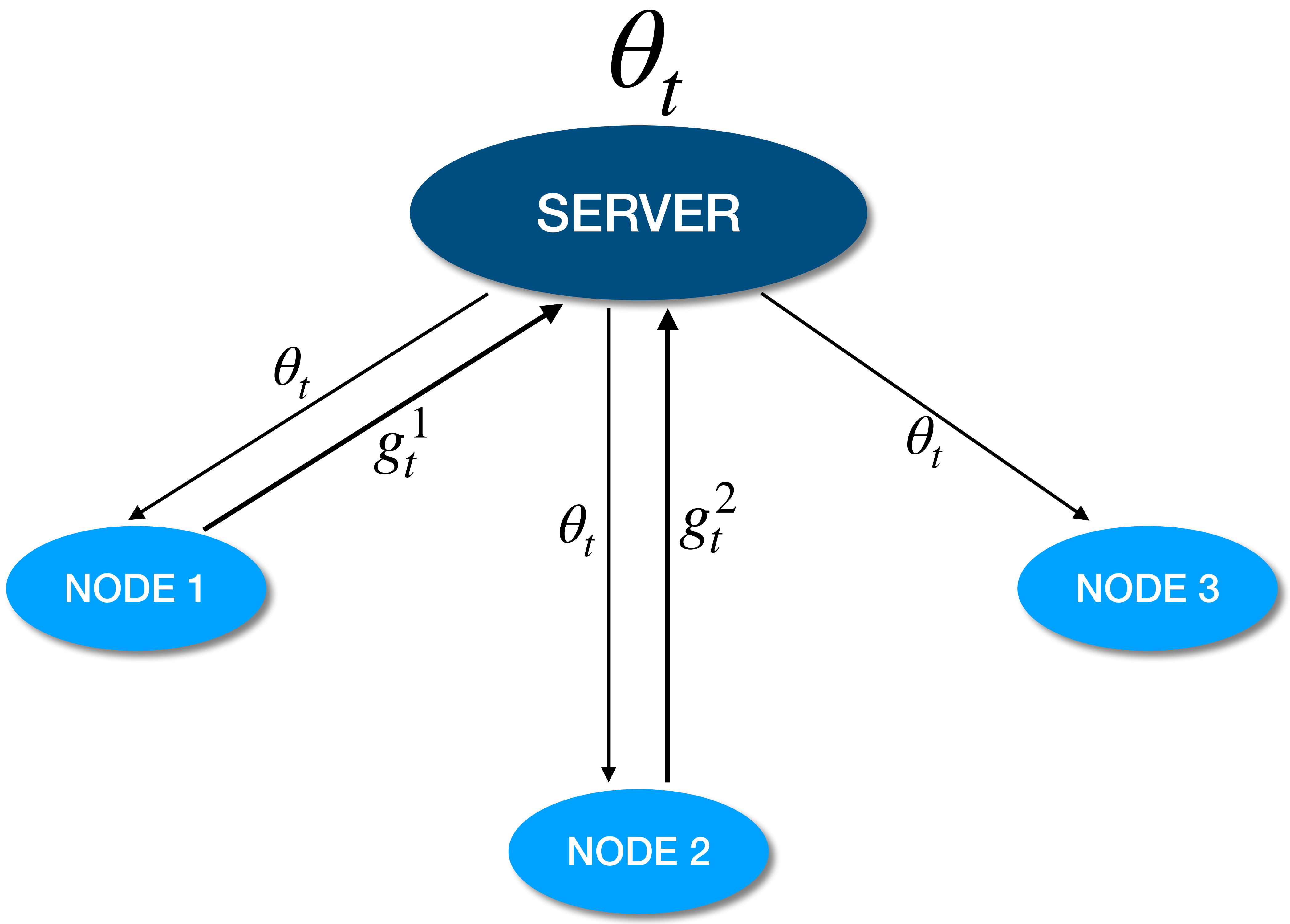
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

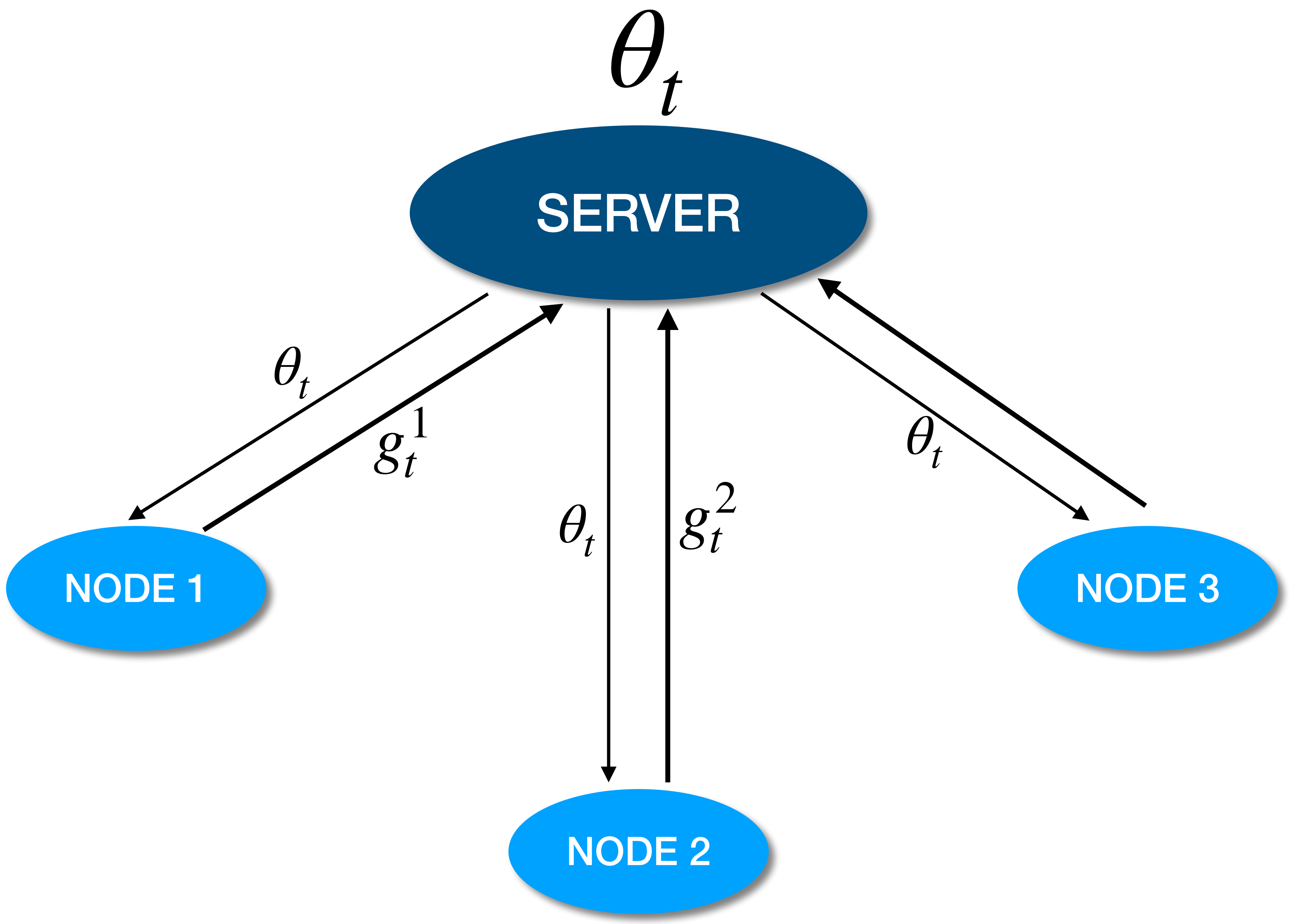
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

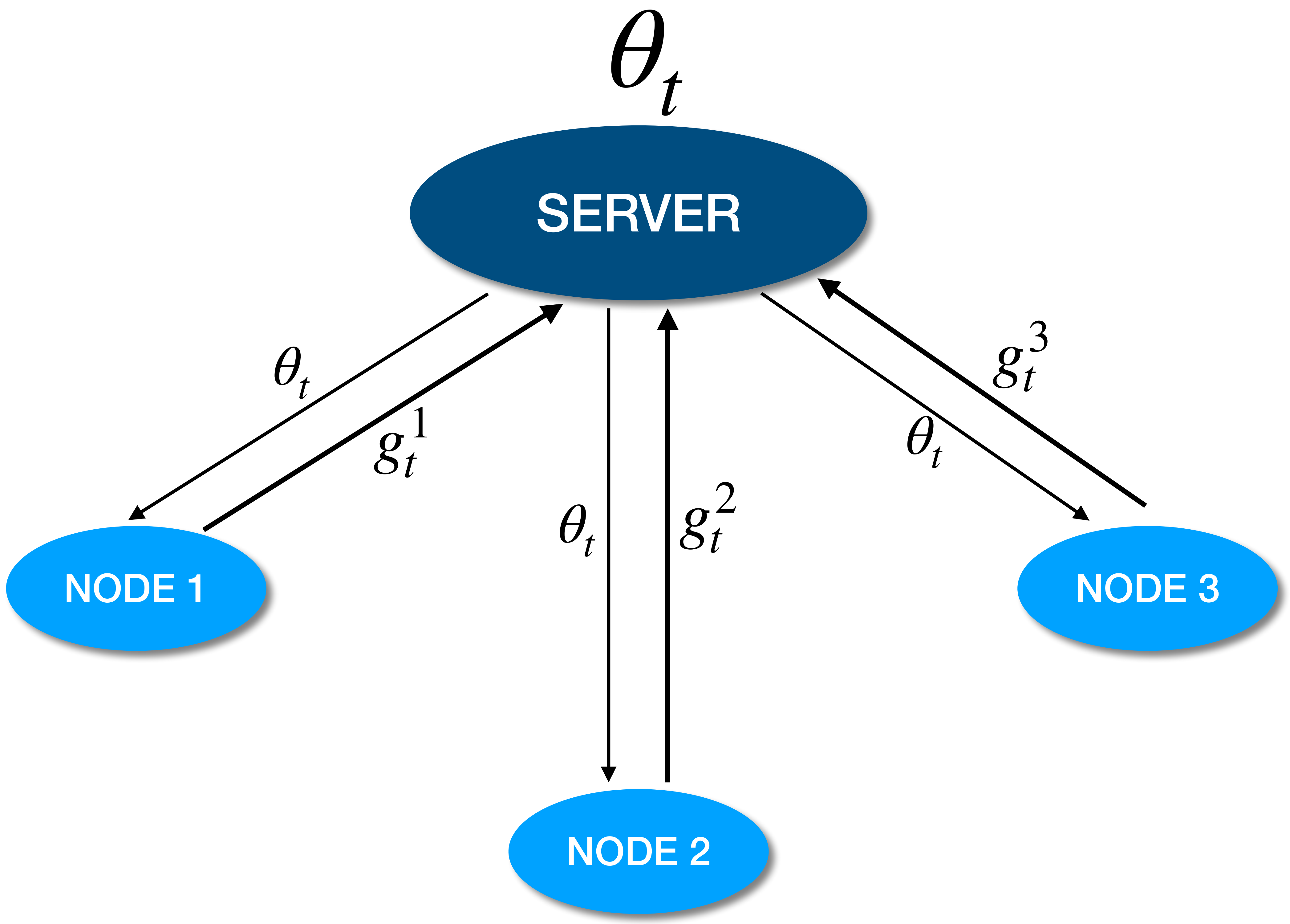
Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient



Distributed SGD

Divides the workload per machine by the total size of the system

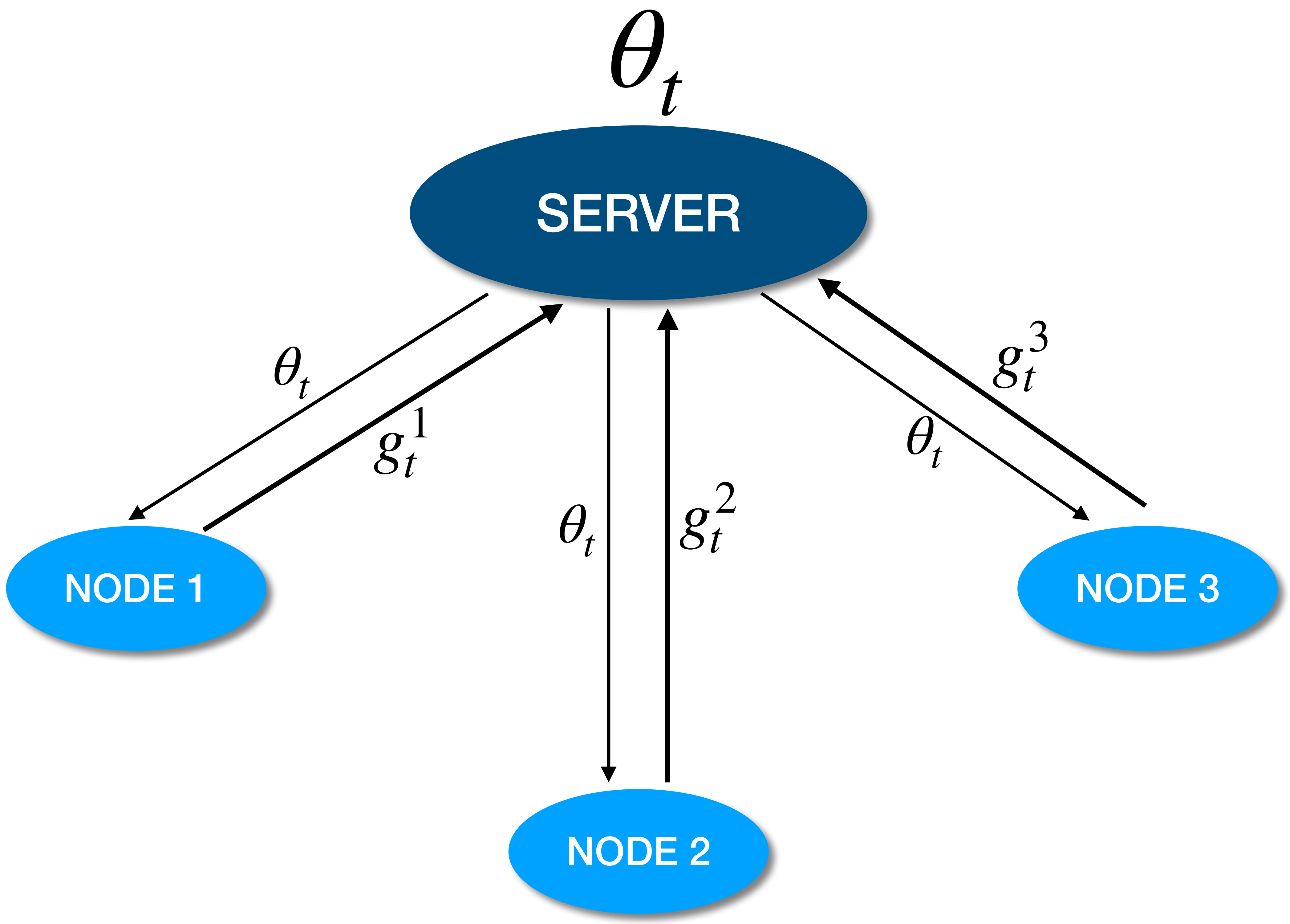
Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$



Distributed SGD

Divides the workload per machine by the total size of the system

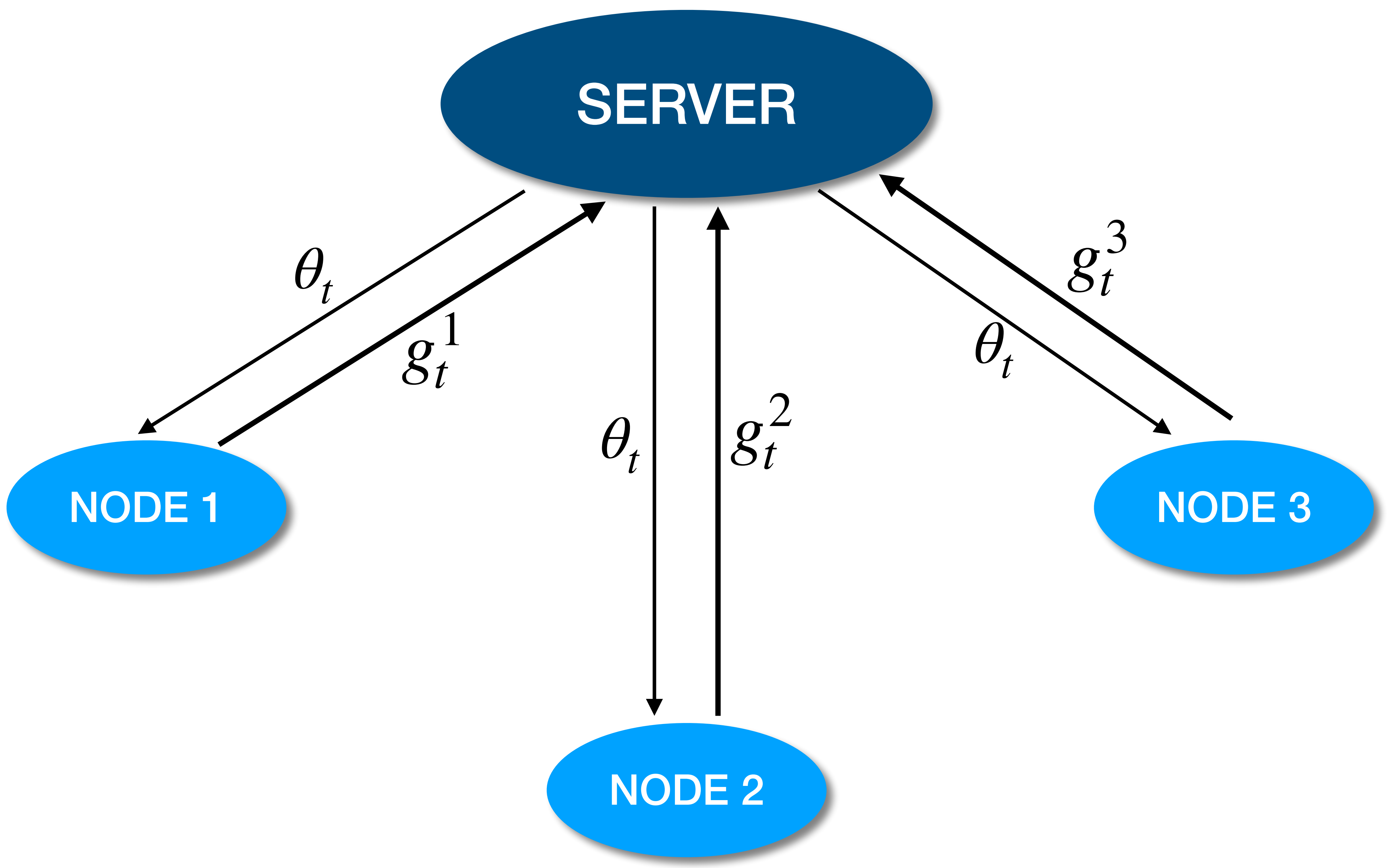
Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

True gradient

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$



Distributed SGD

Divides the workload per machine by the total size of the system

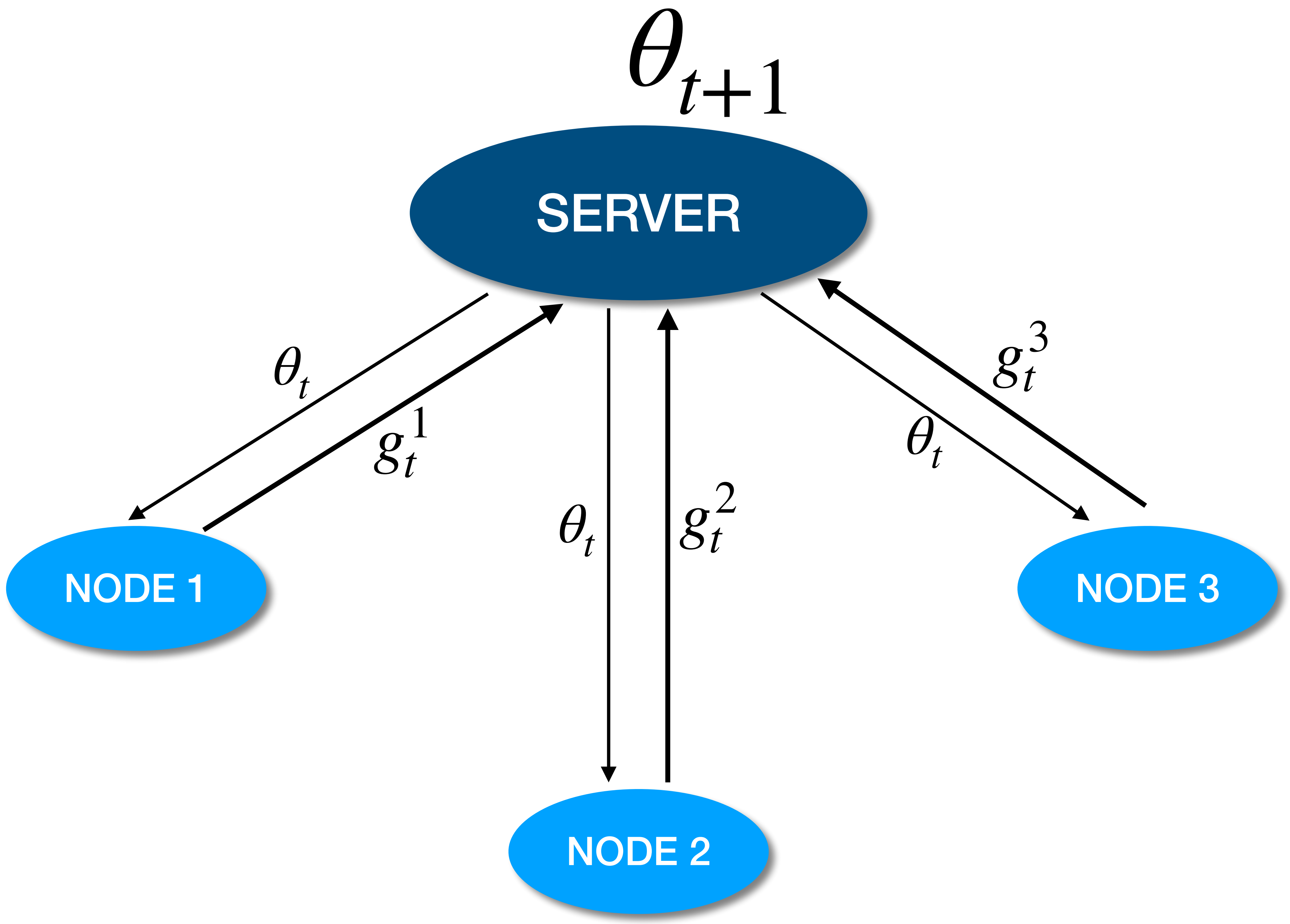
Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

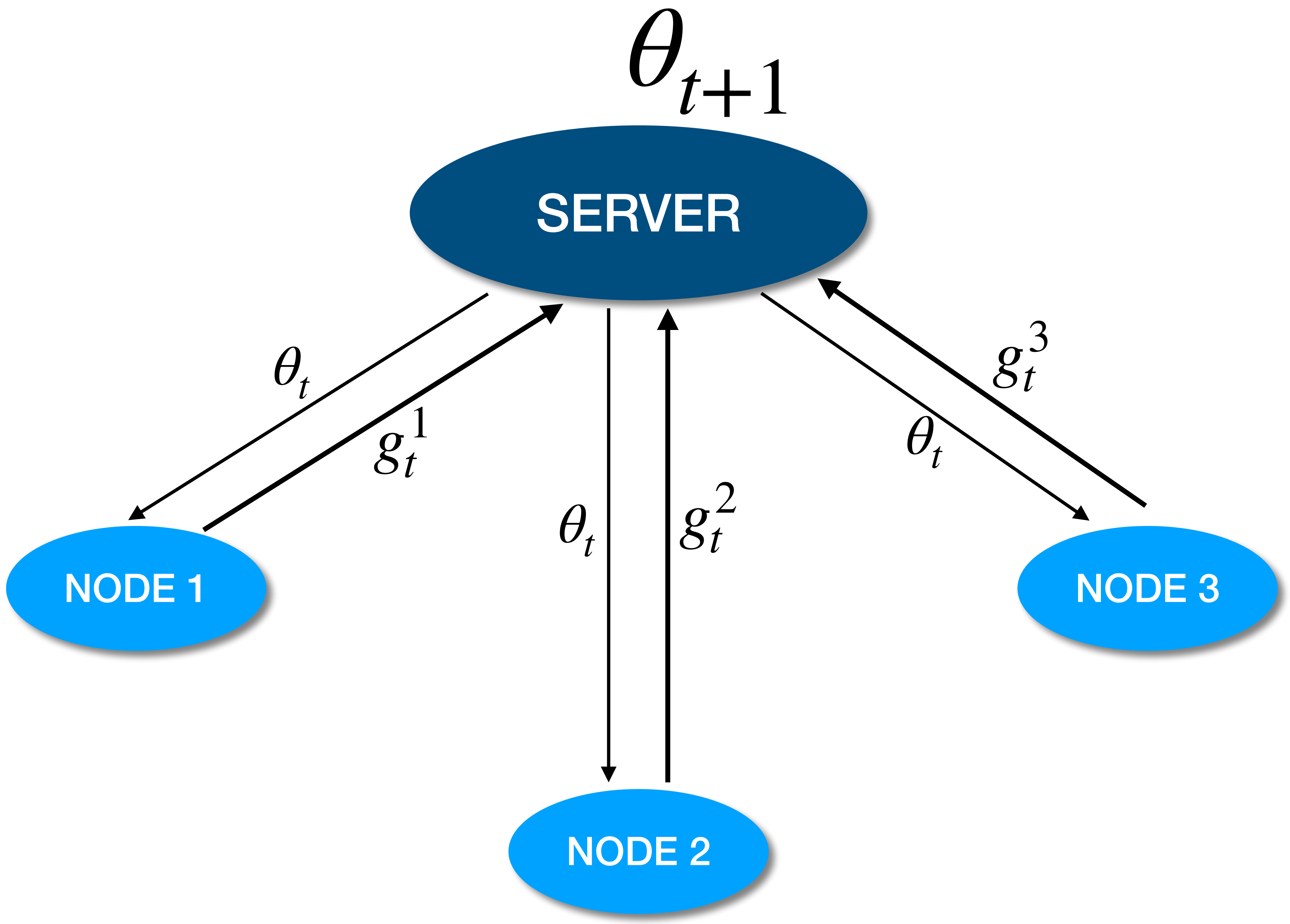
$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, \hat{g}_t$



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

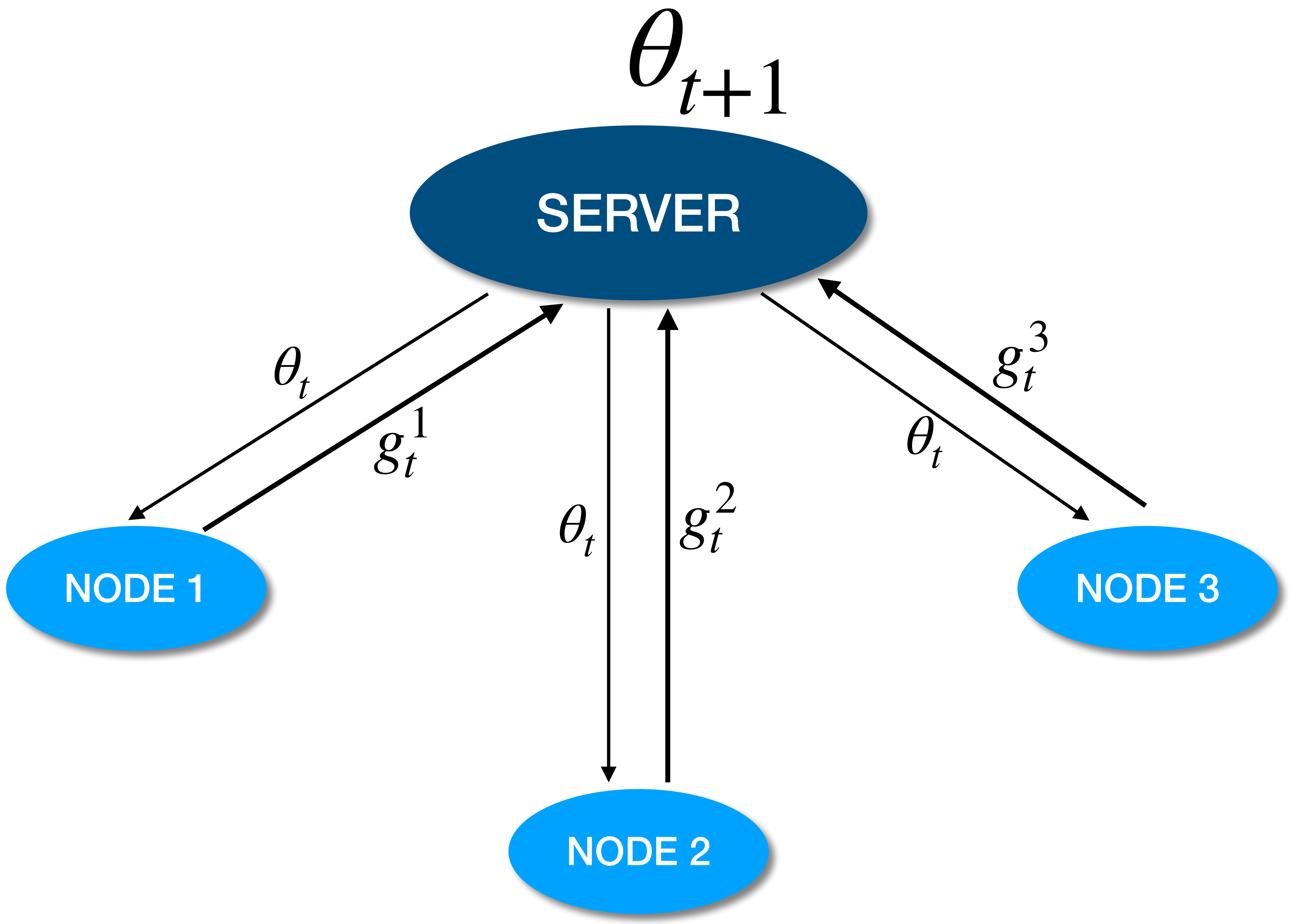
$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

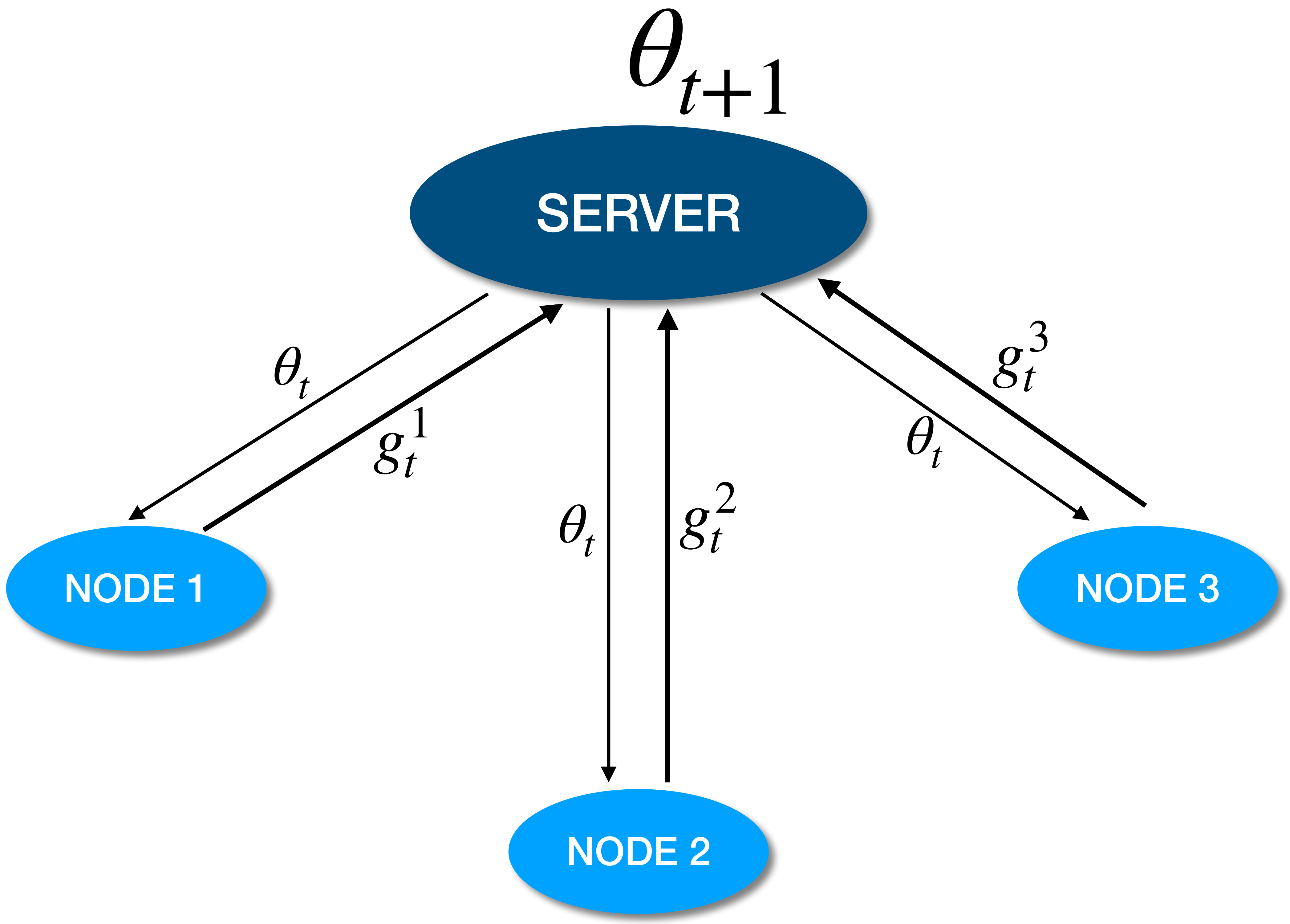
$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

True gradient

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

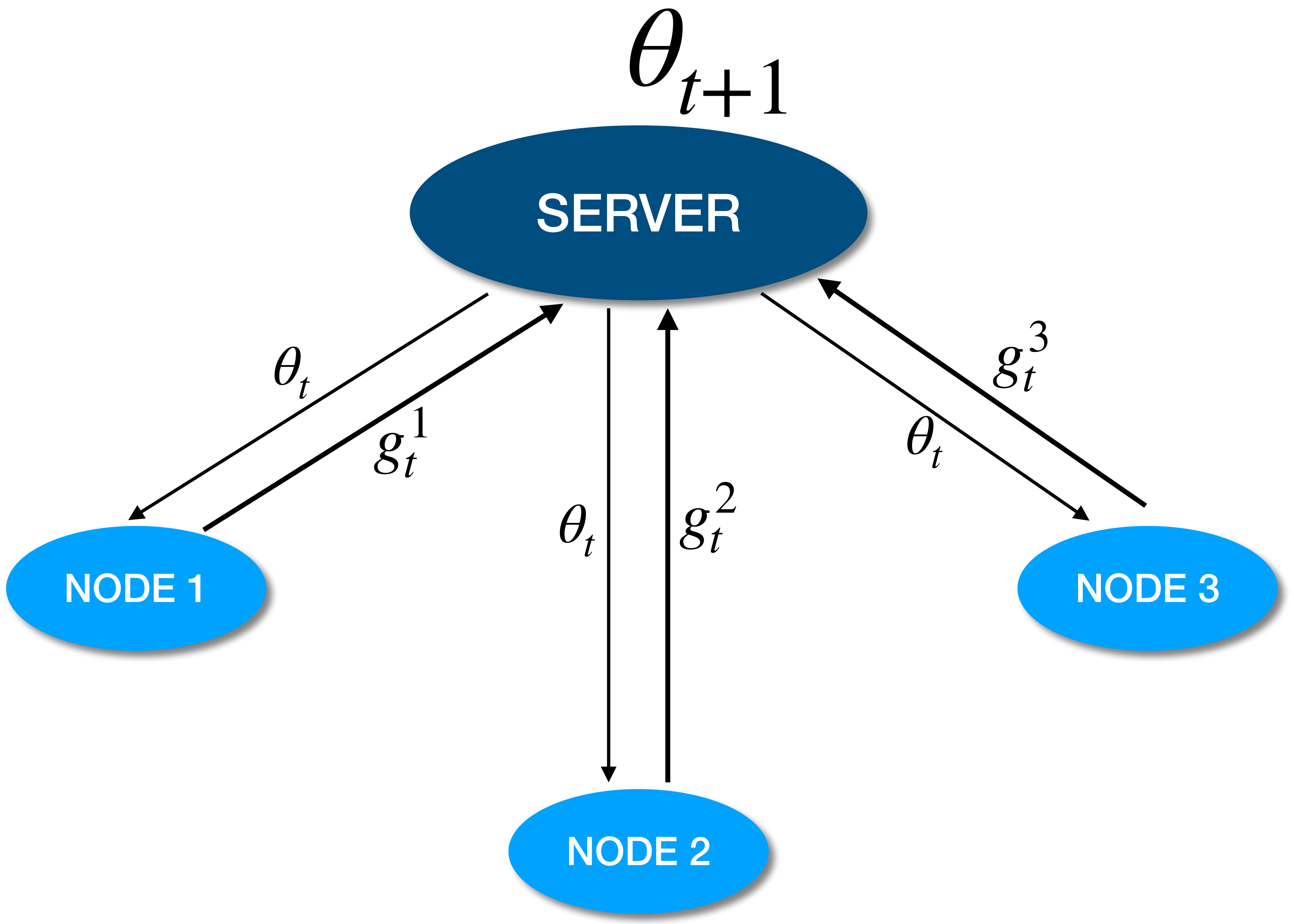
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$



Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

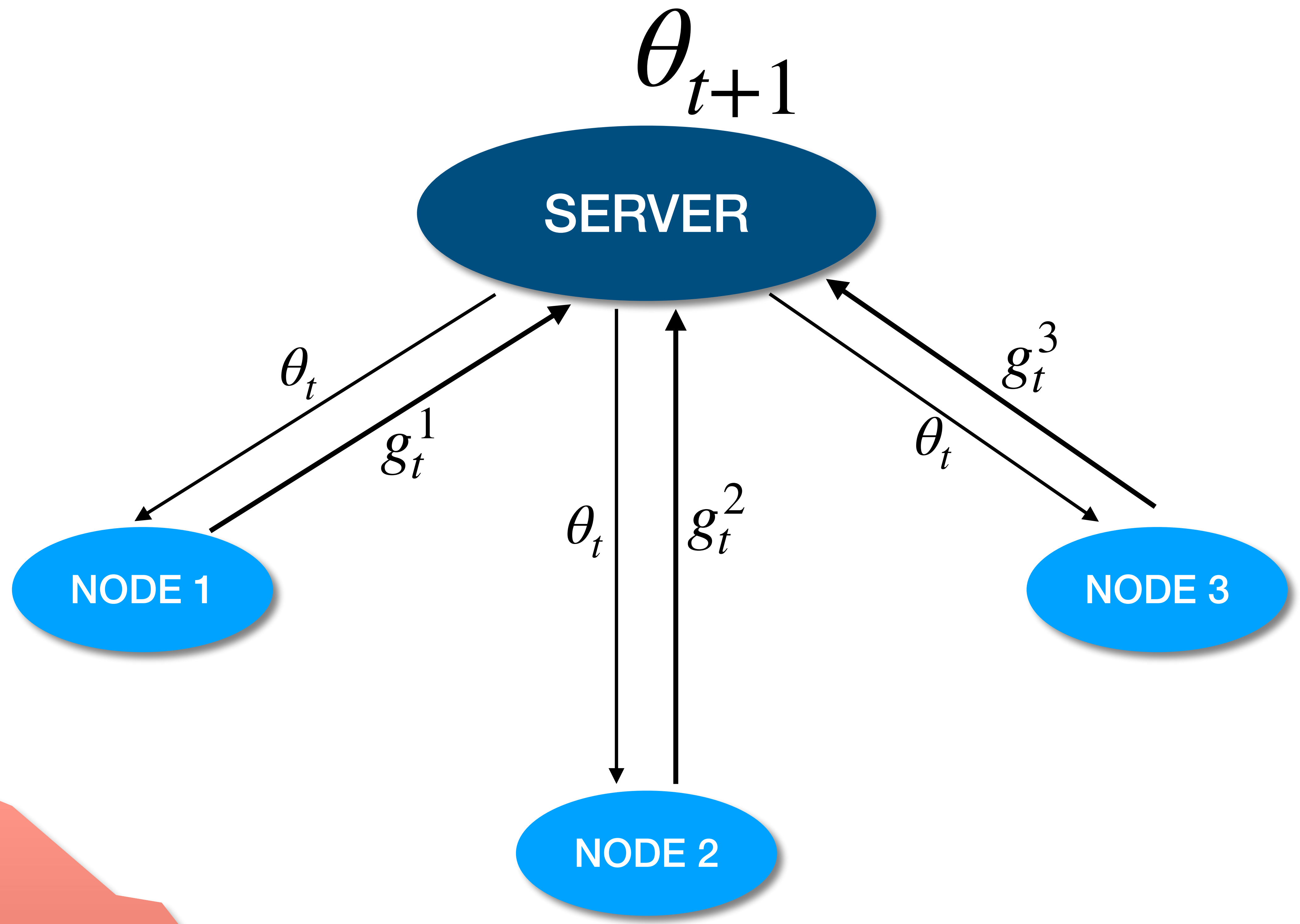
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \le \sigma^2 < \infty$$

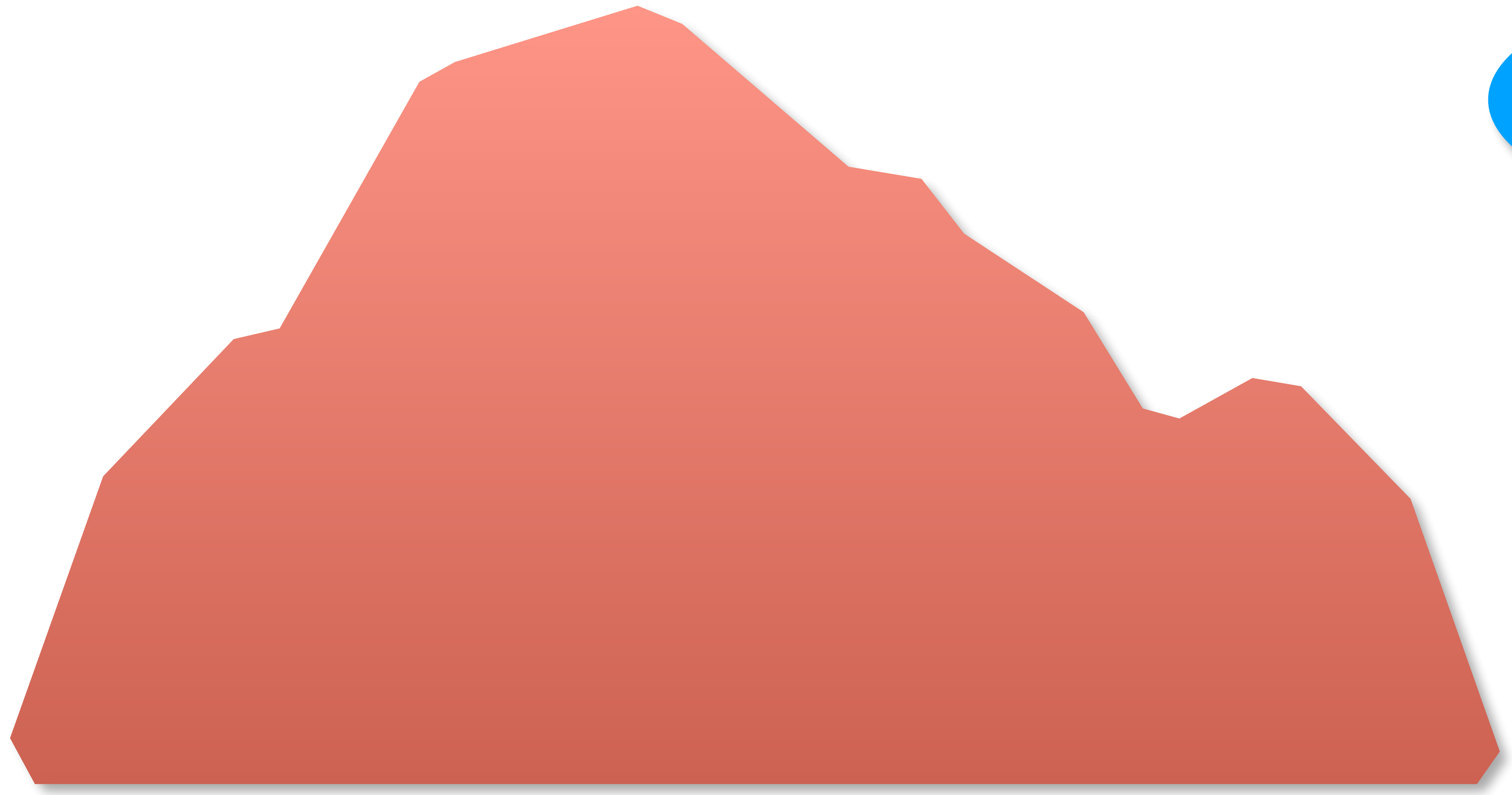
True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

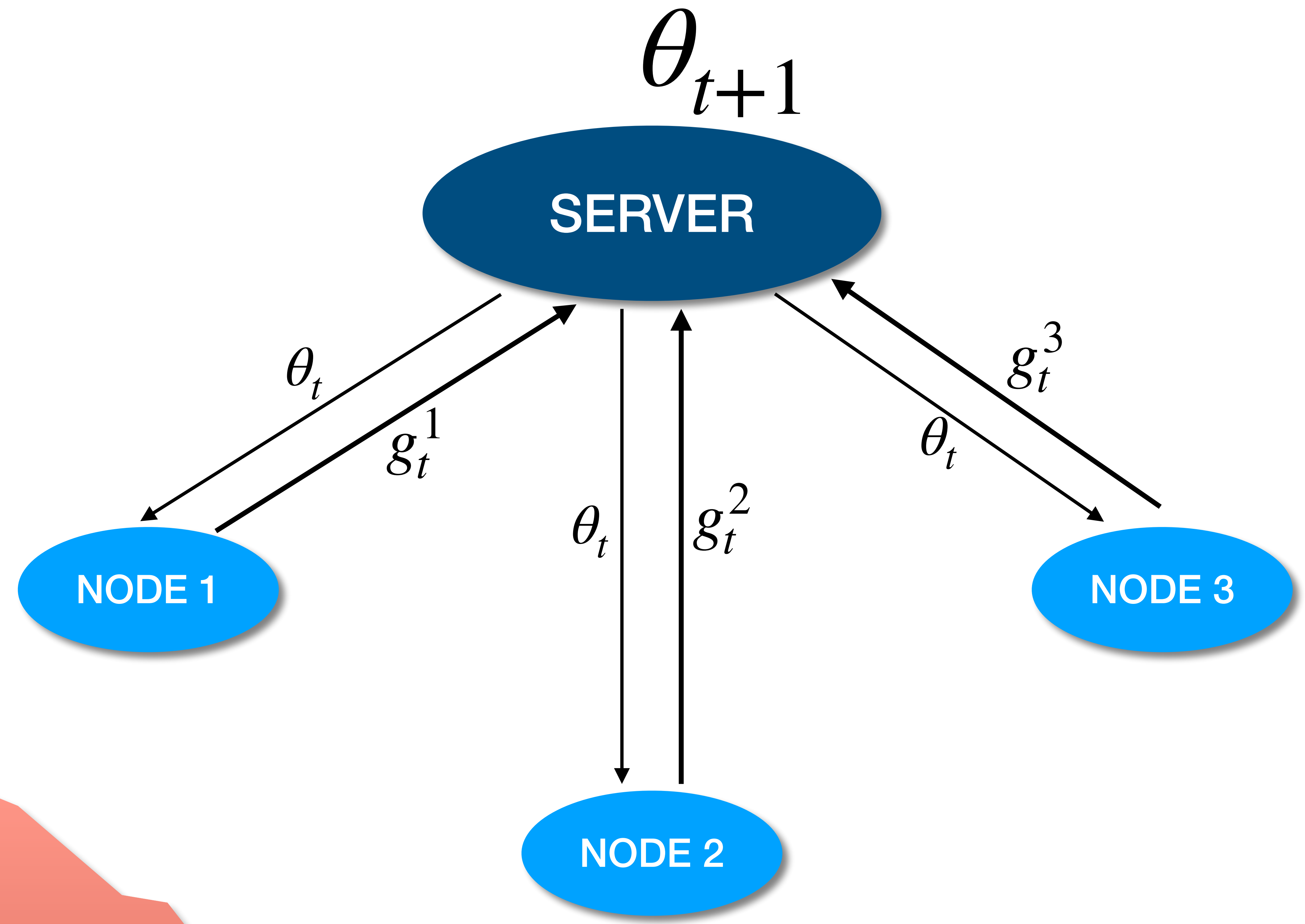
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

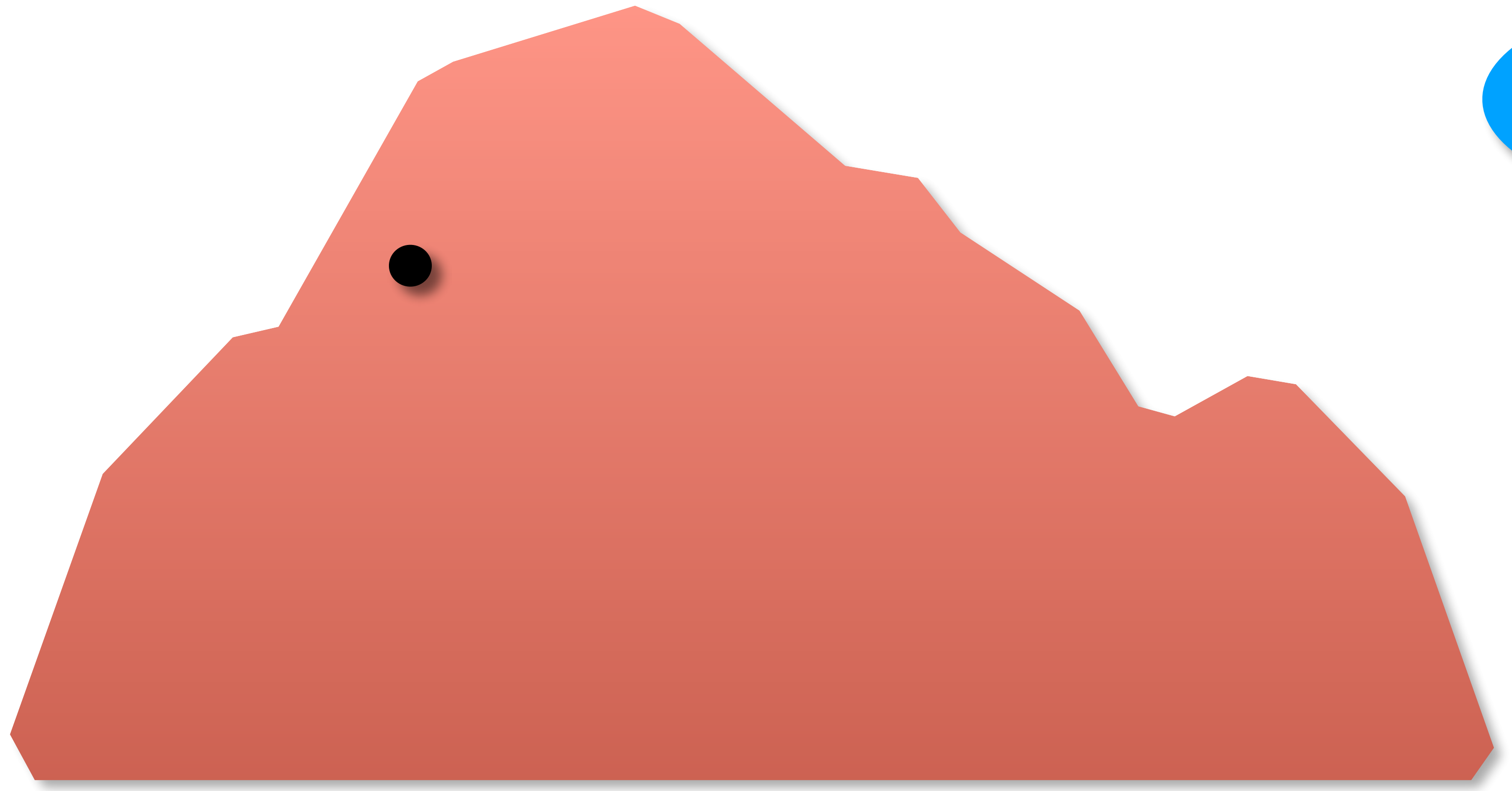
True gradient

Server averages the gradients, $\widehat{g}_t = \dfrac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

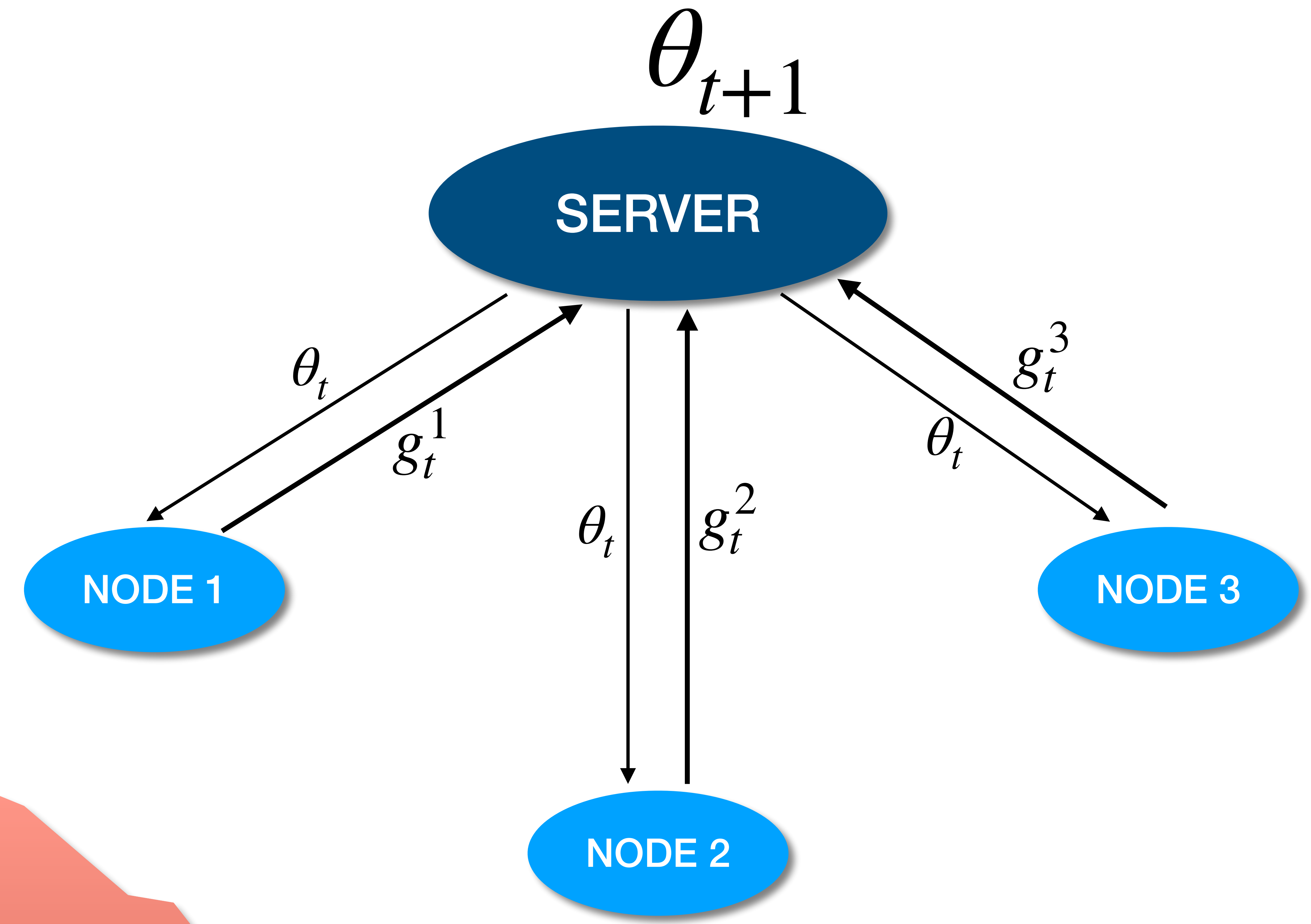
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

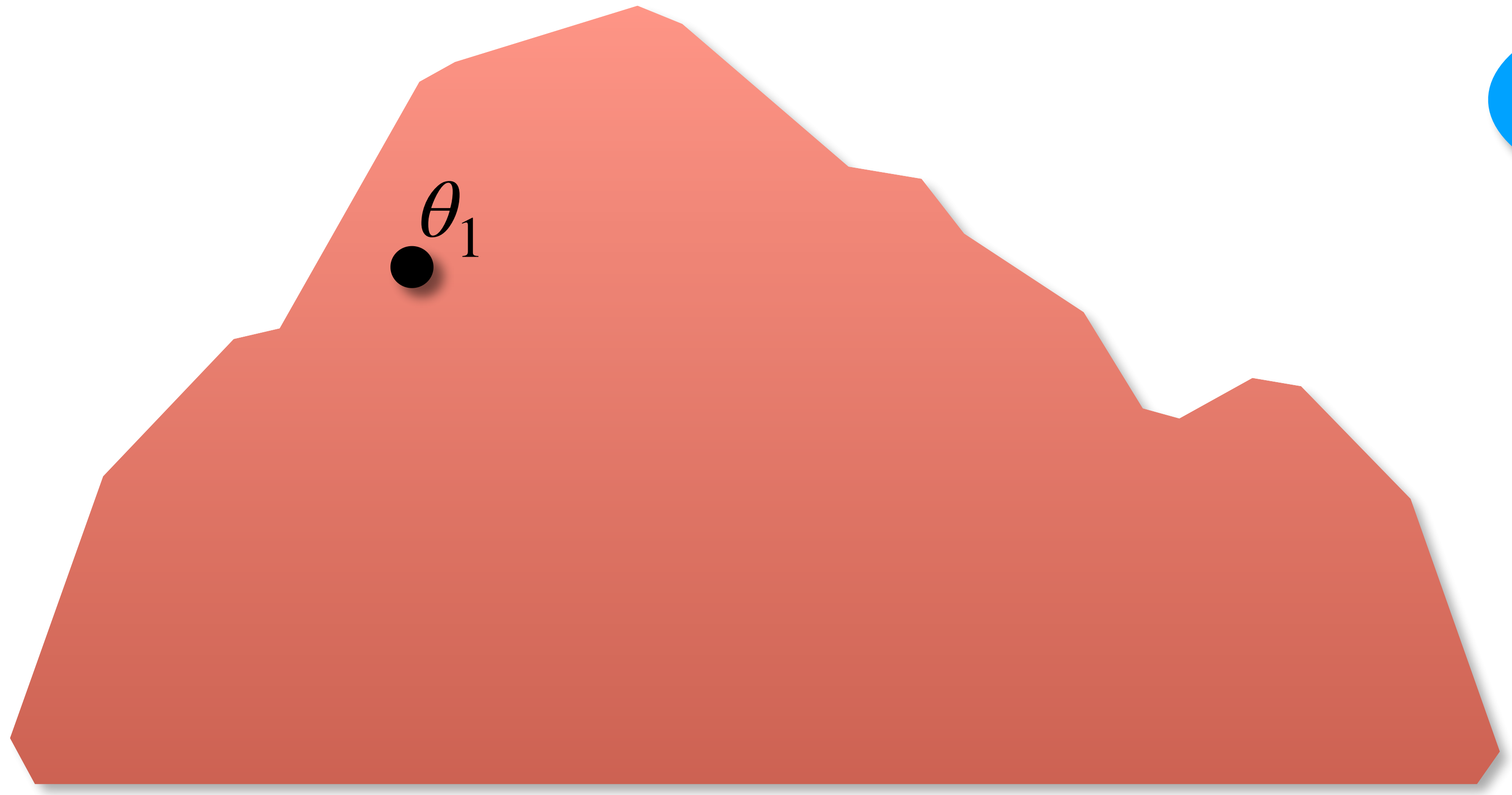
True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

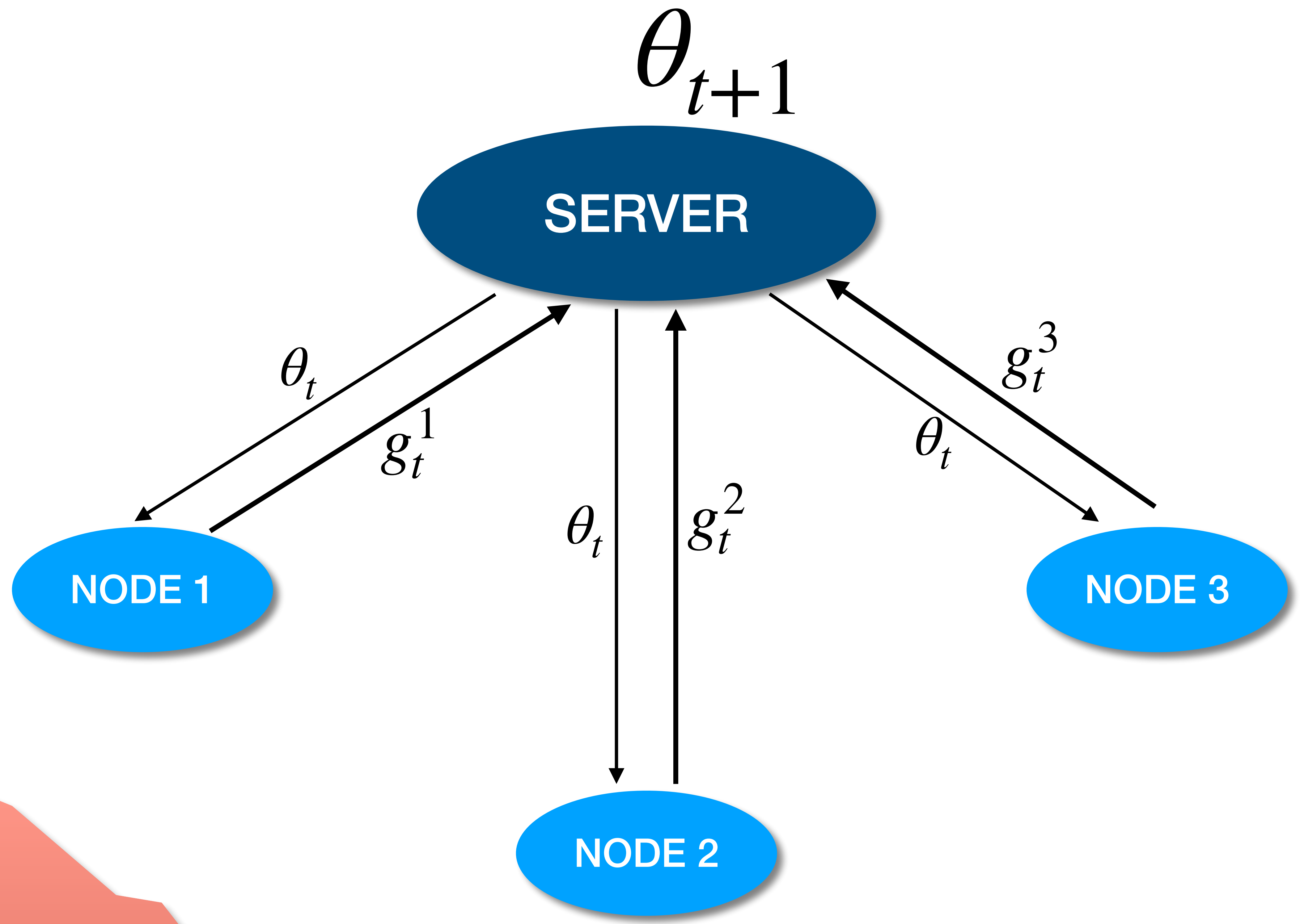
True gradient

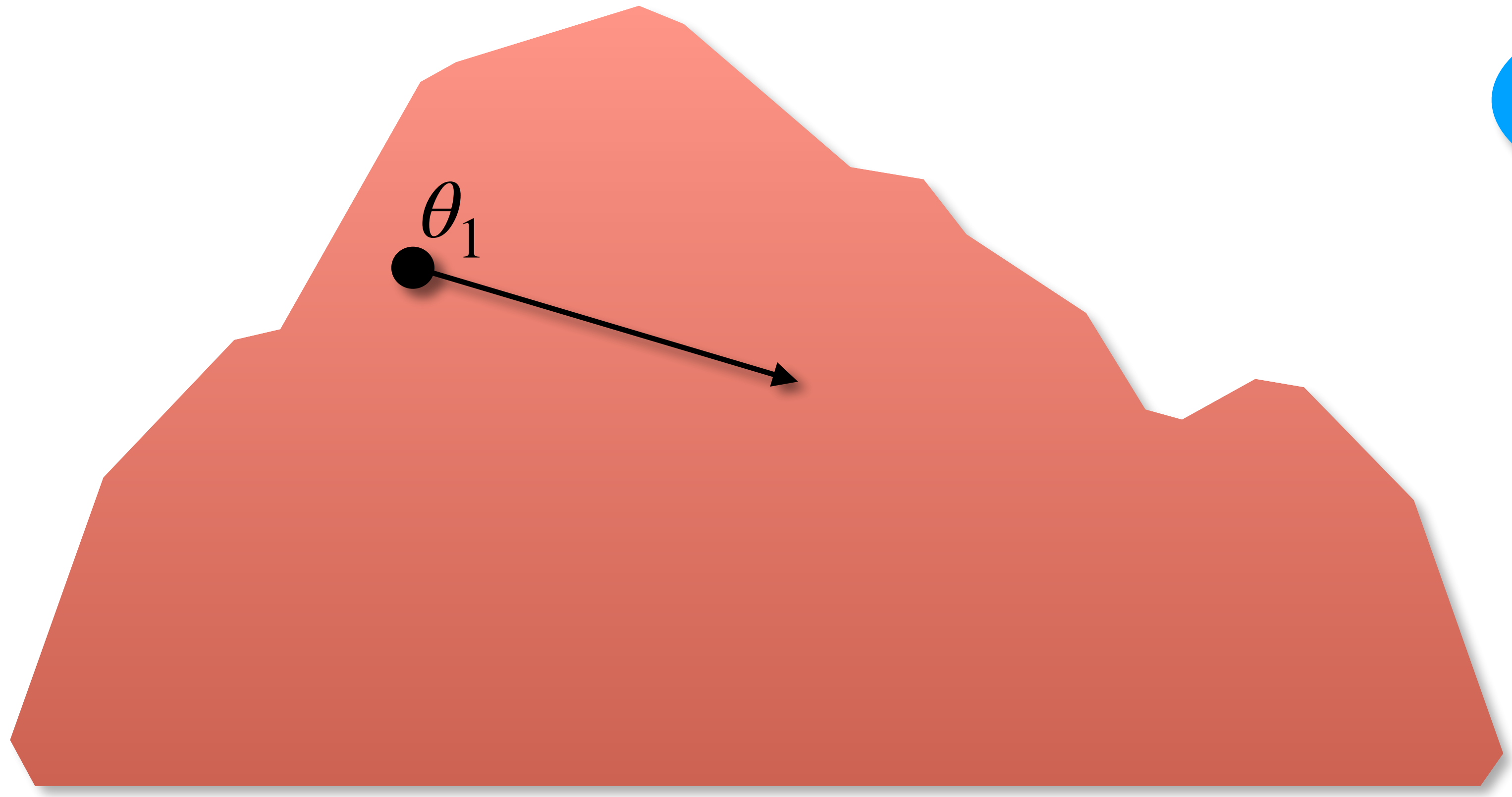
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

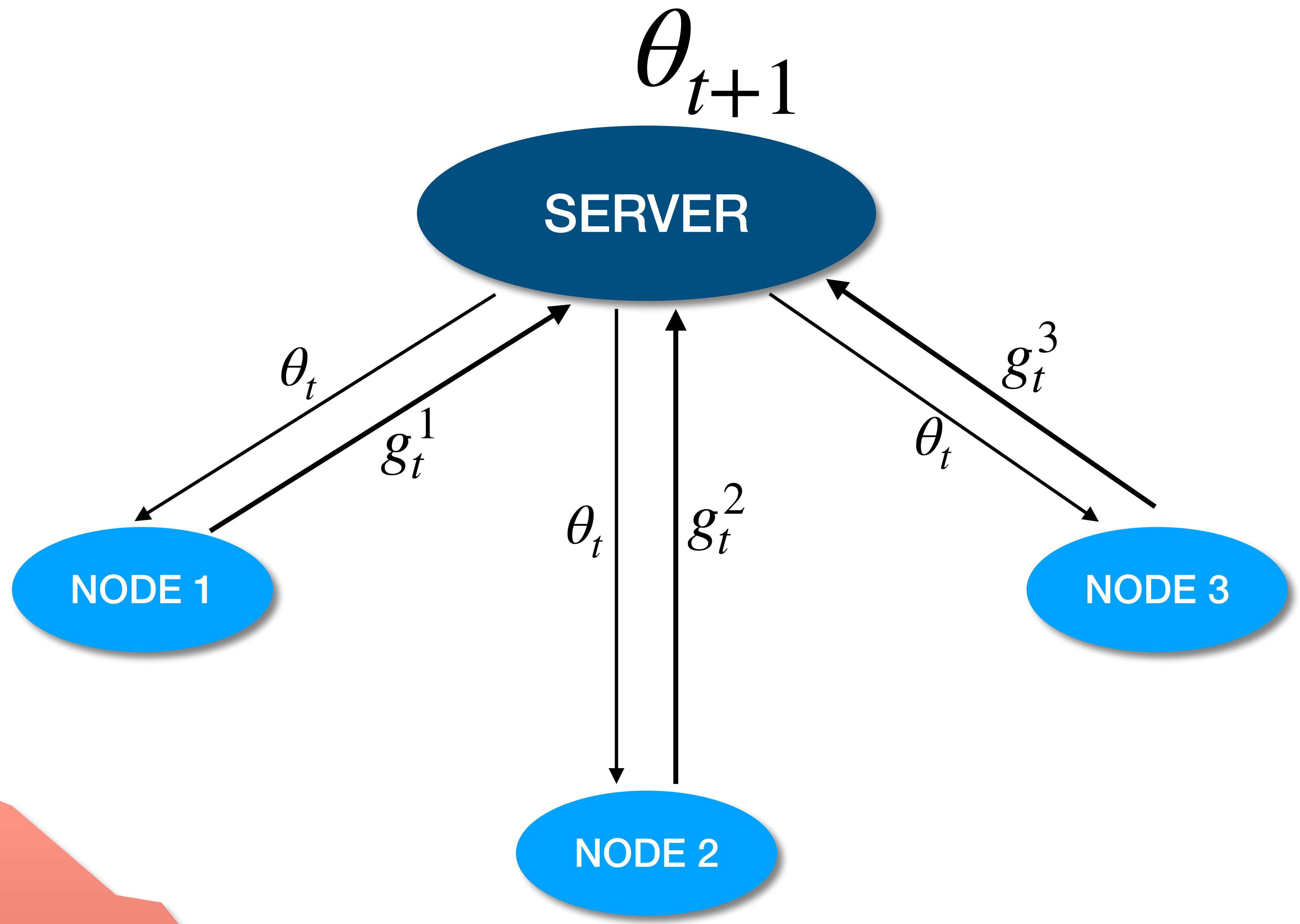
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

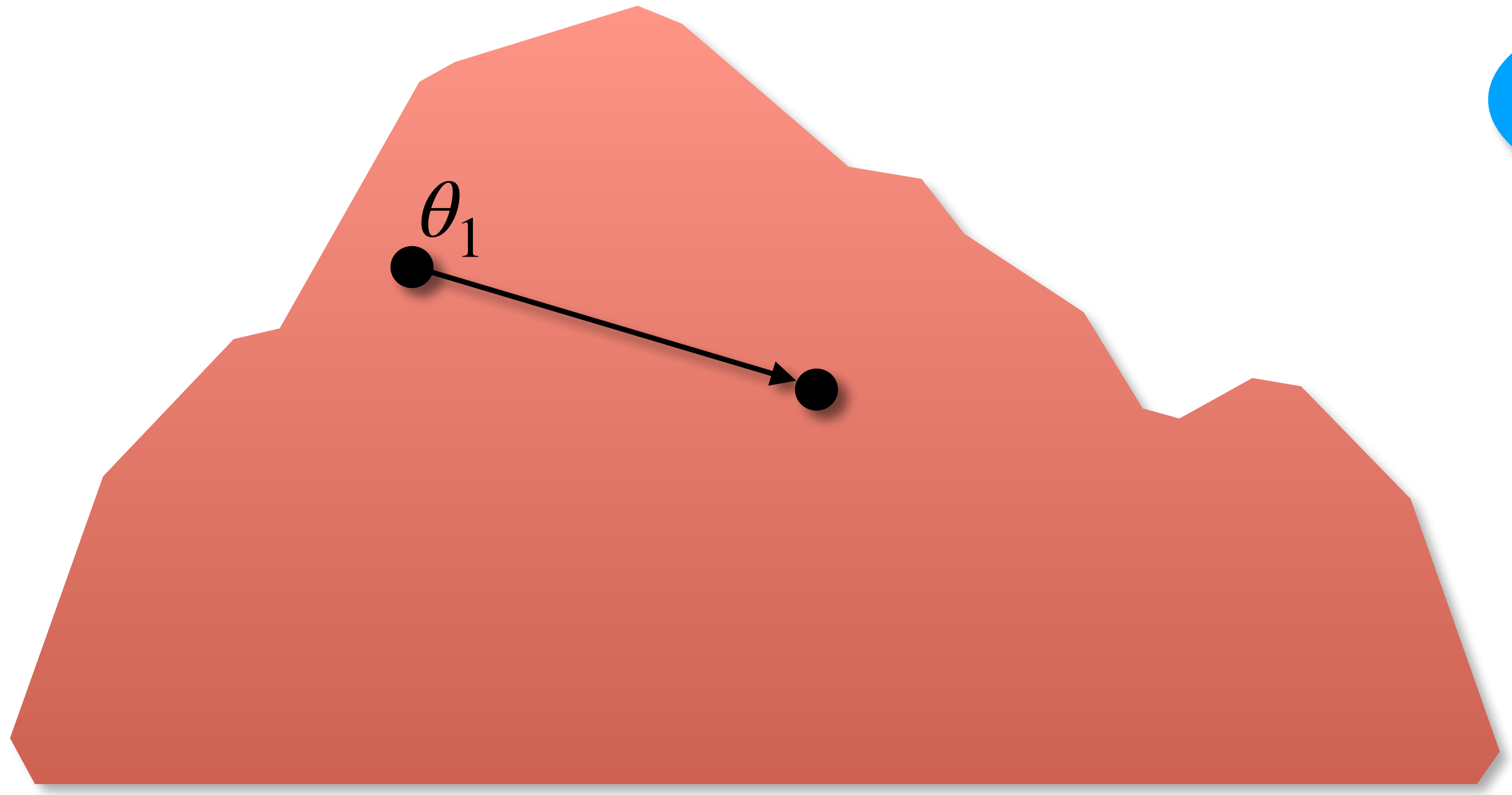
True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

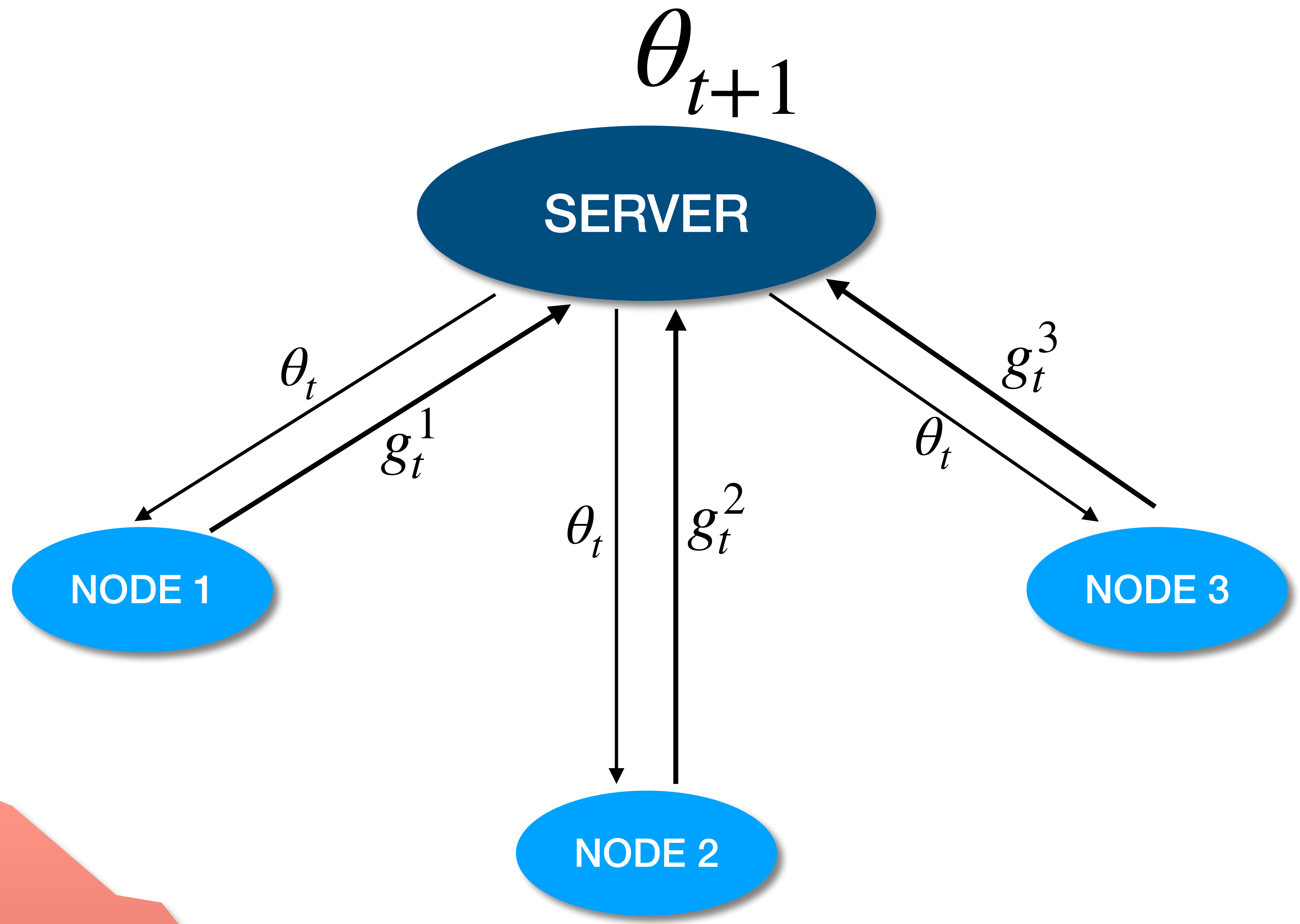
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

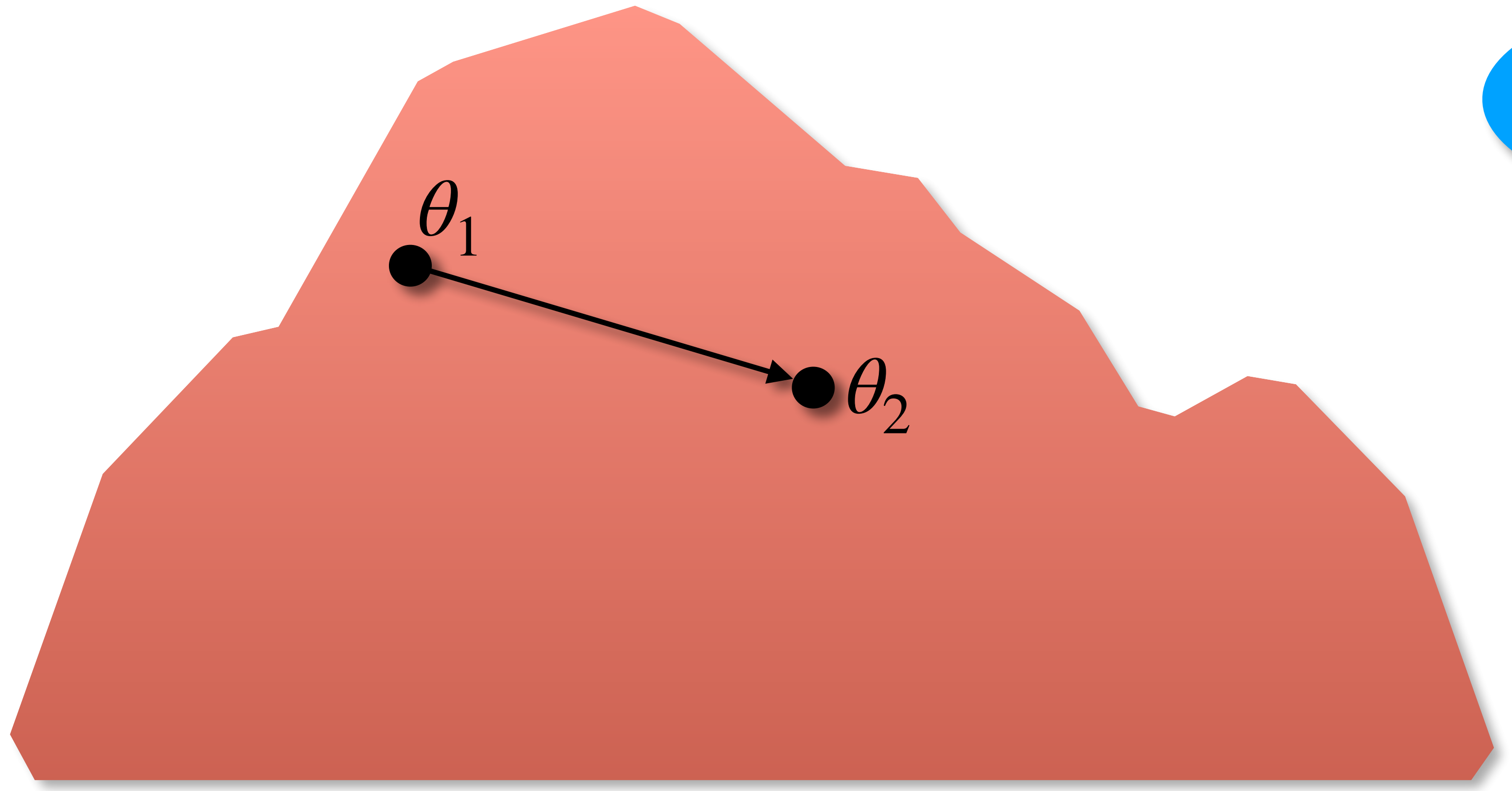
True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

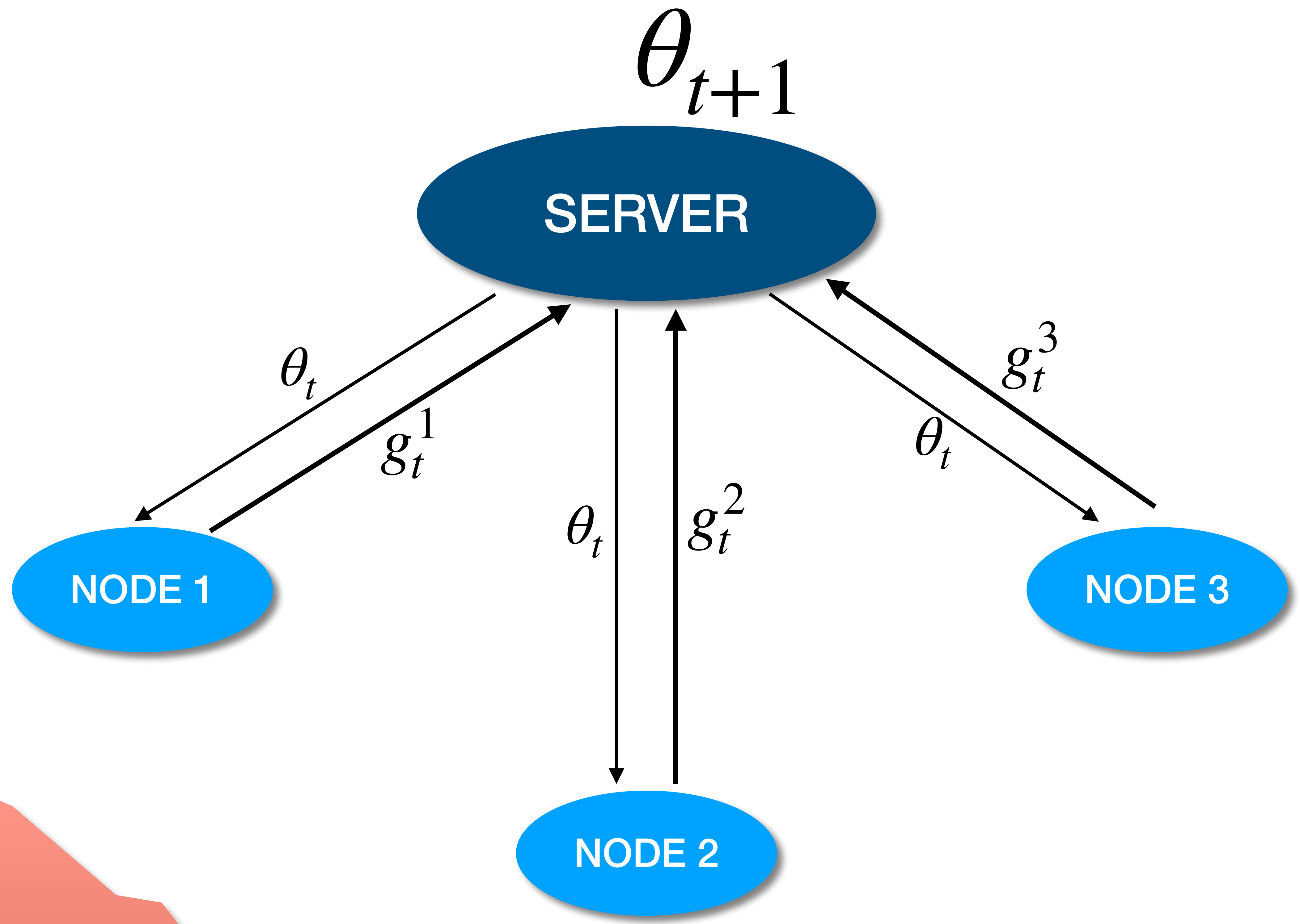
True gradient

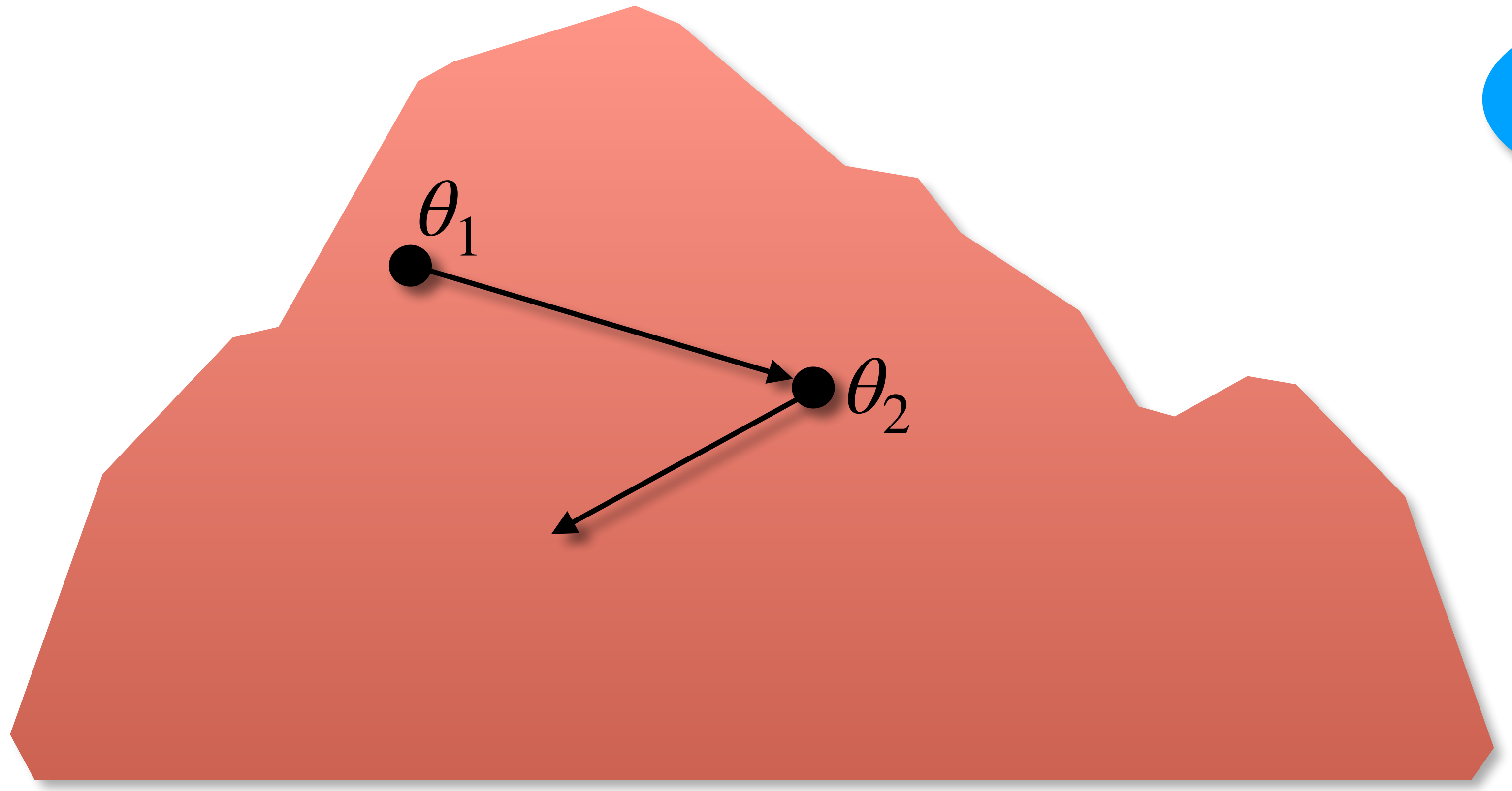
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

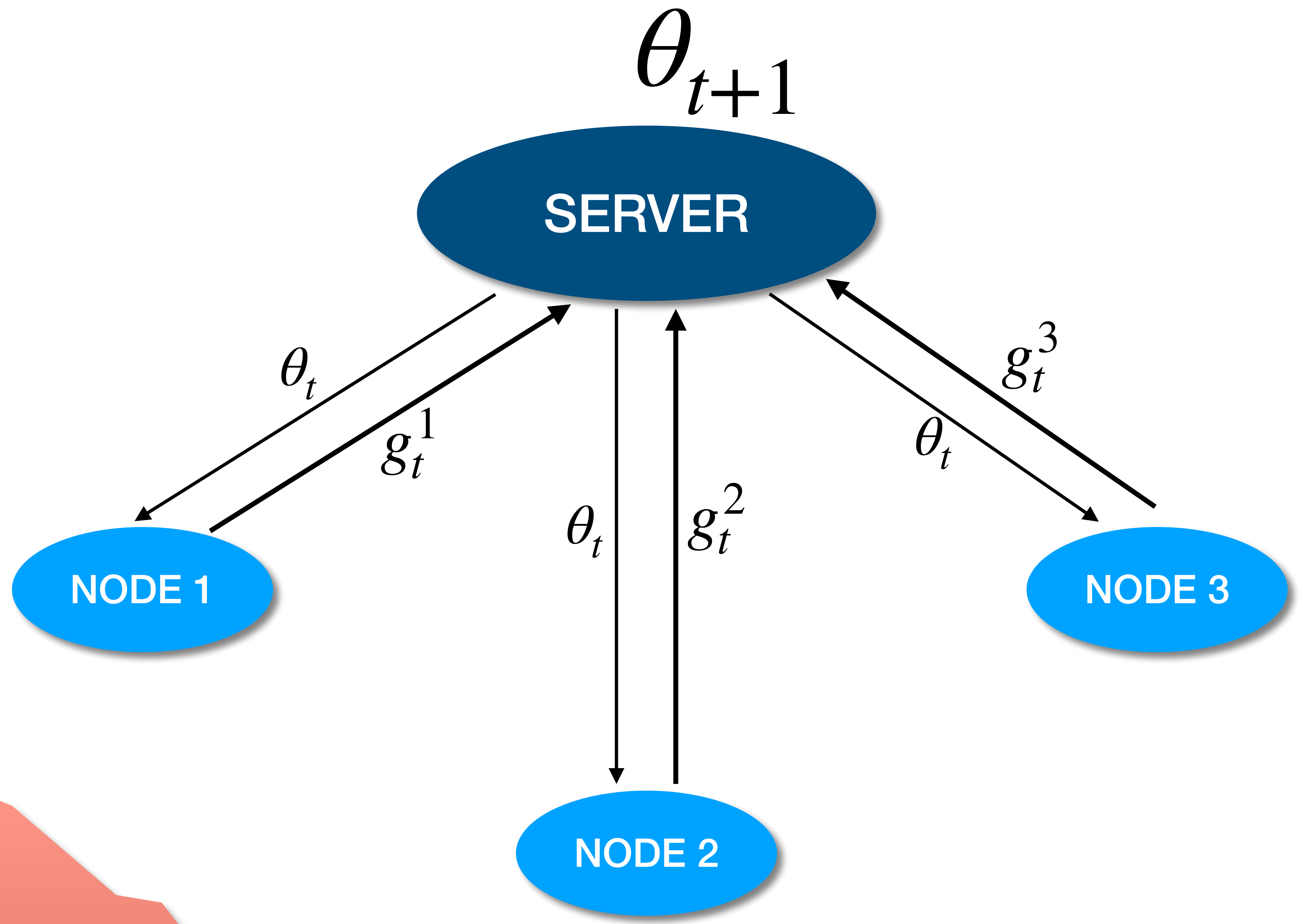
True gradient

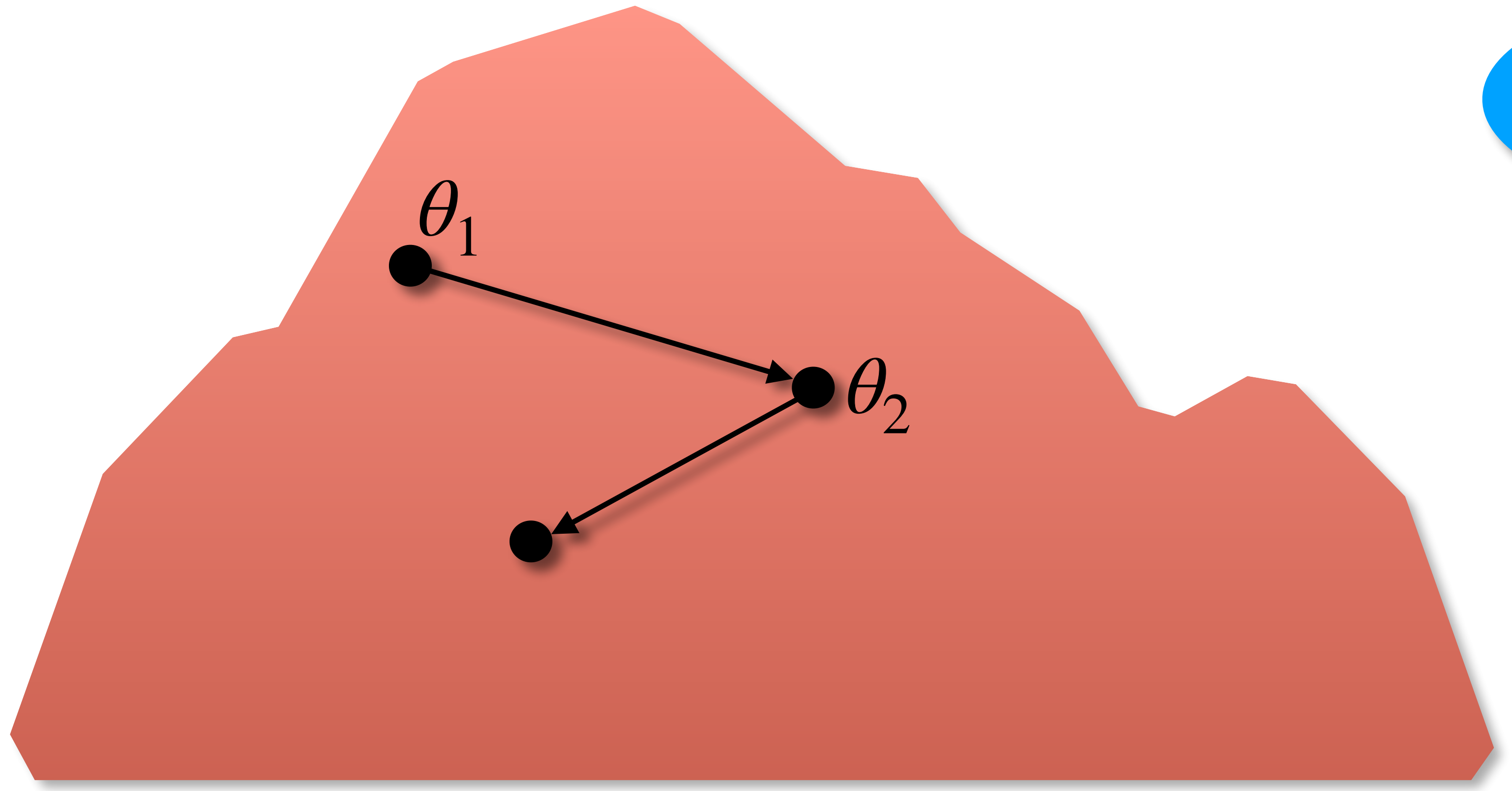
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

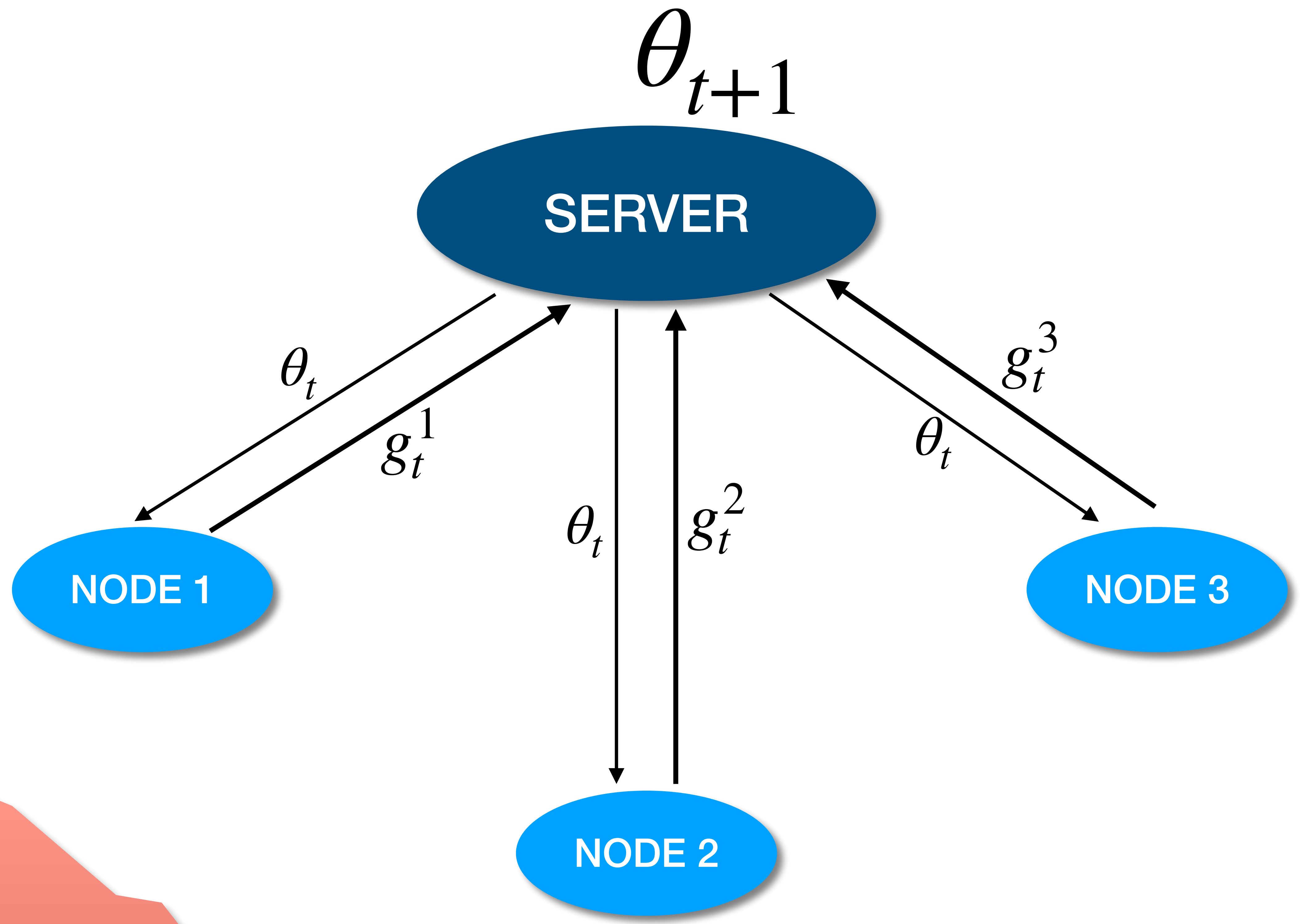
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

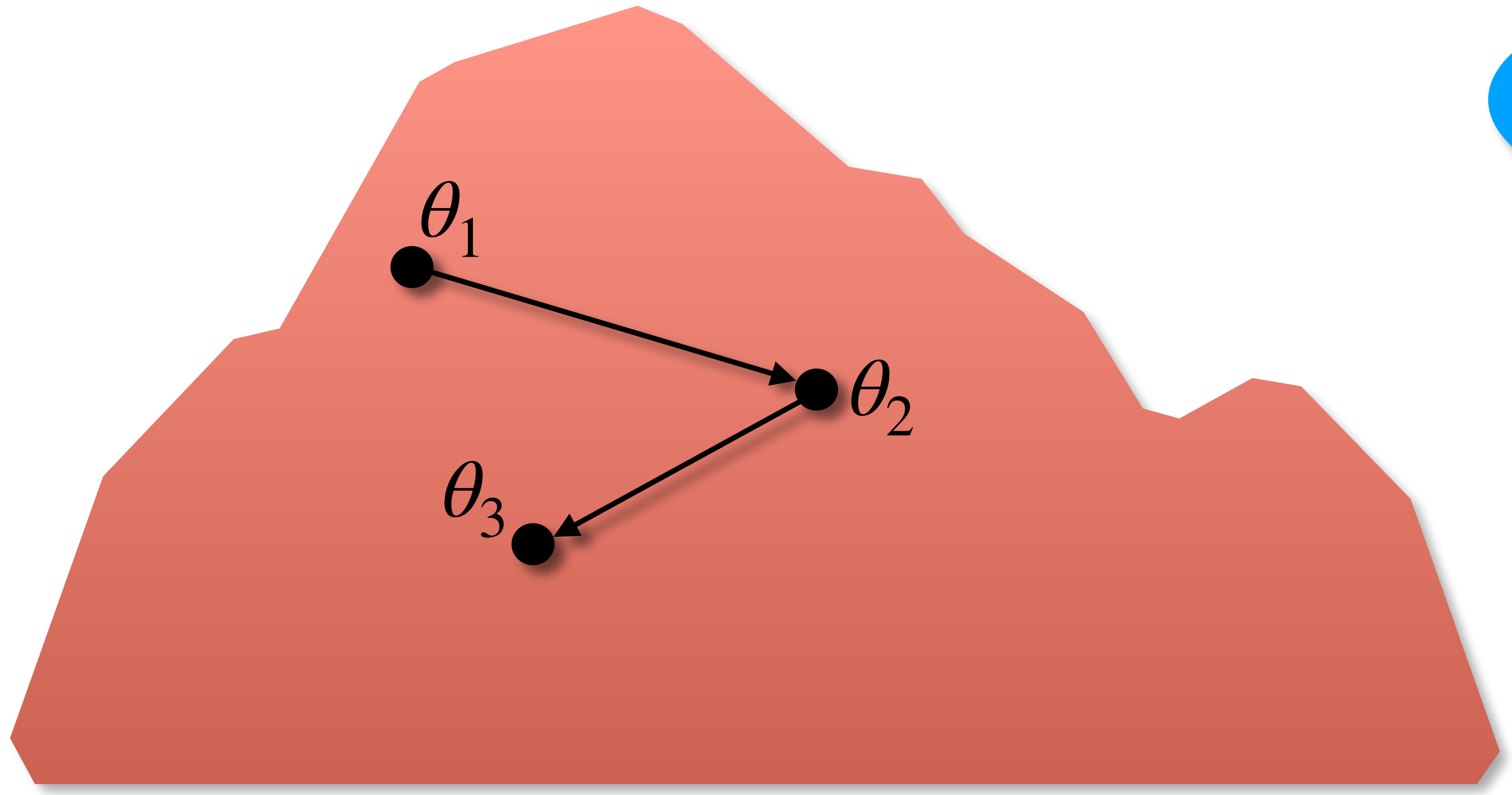
True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

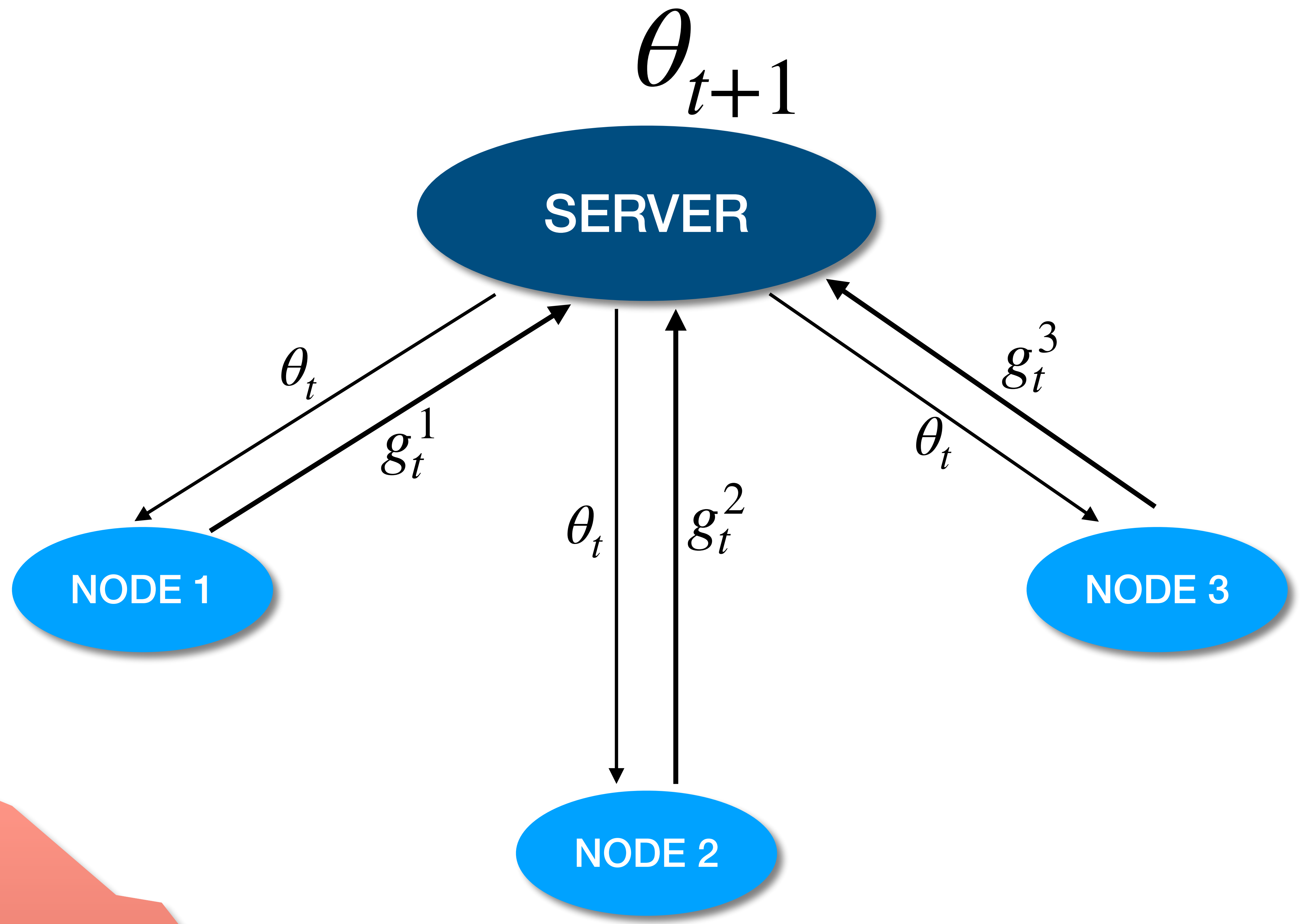
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

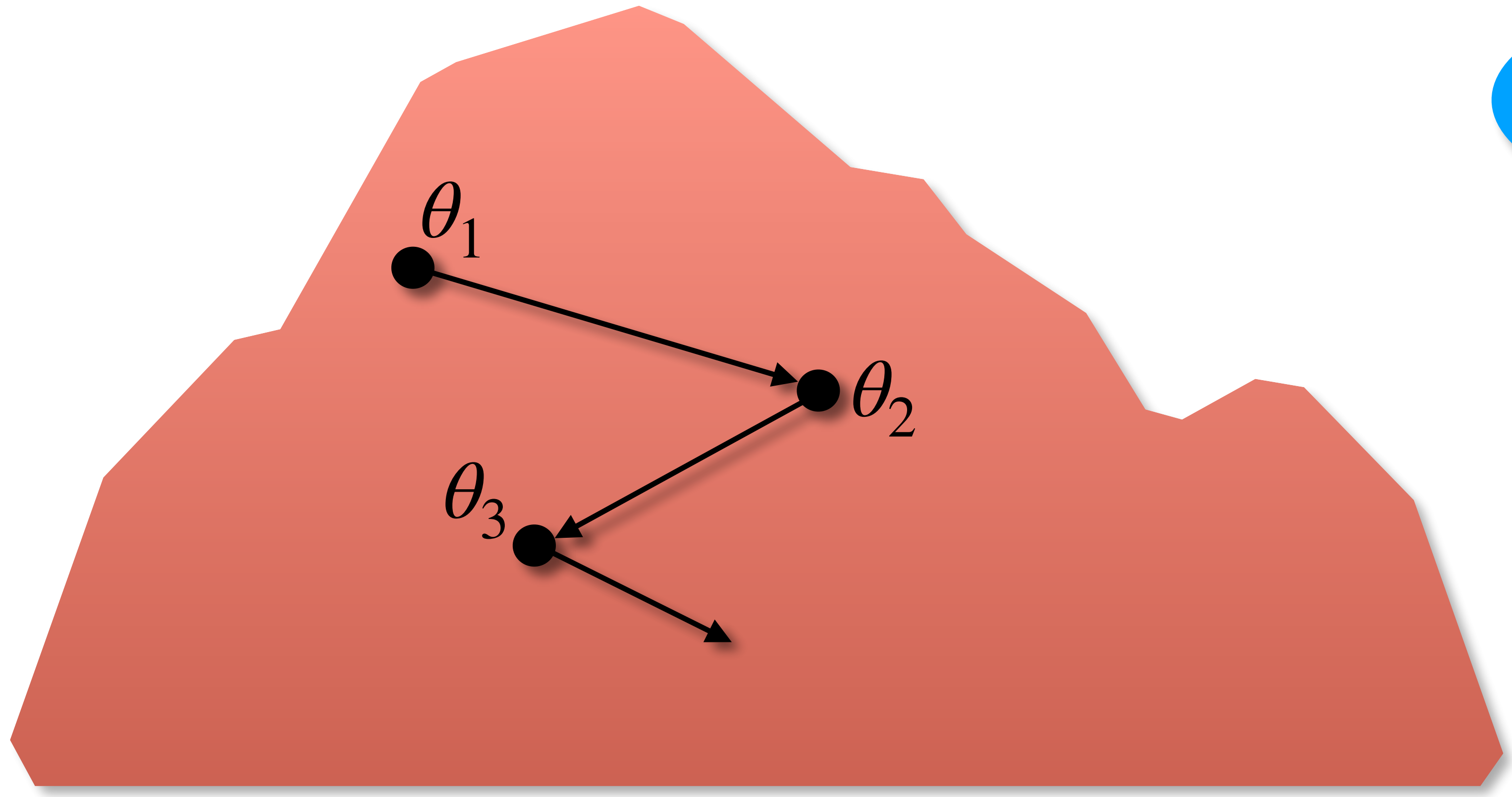
True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

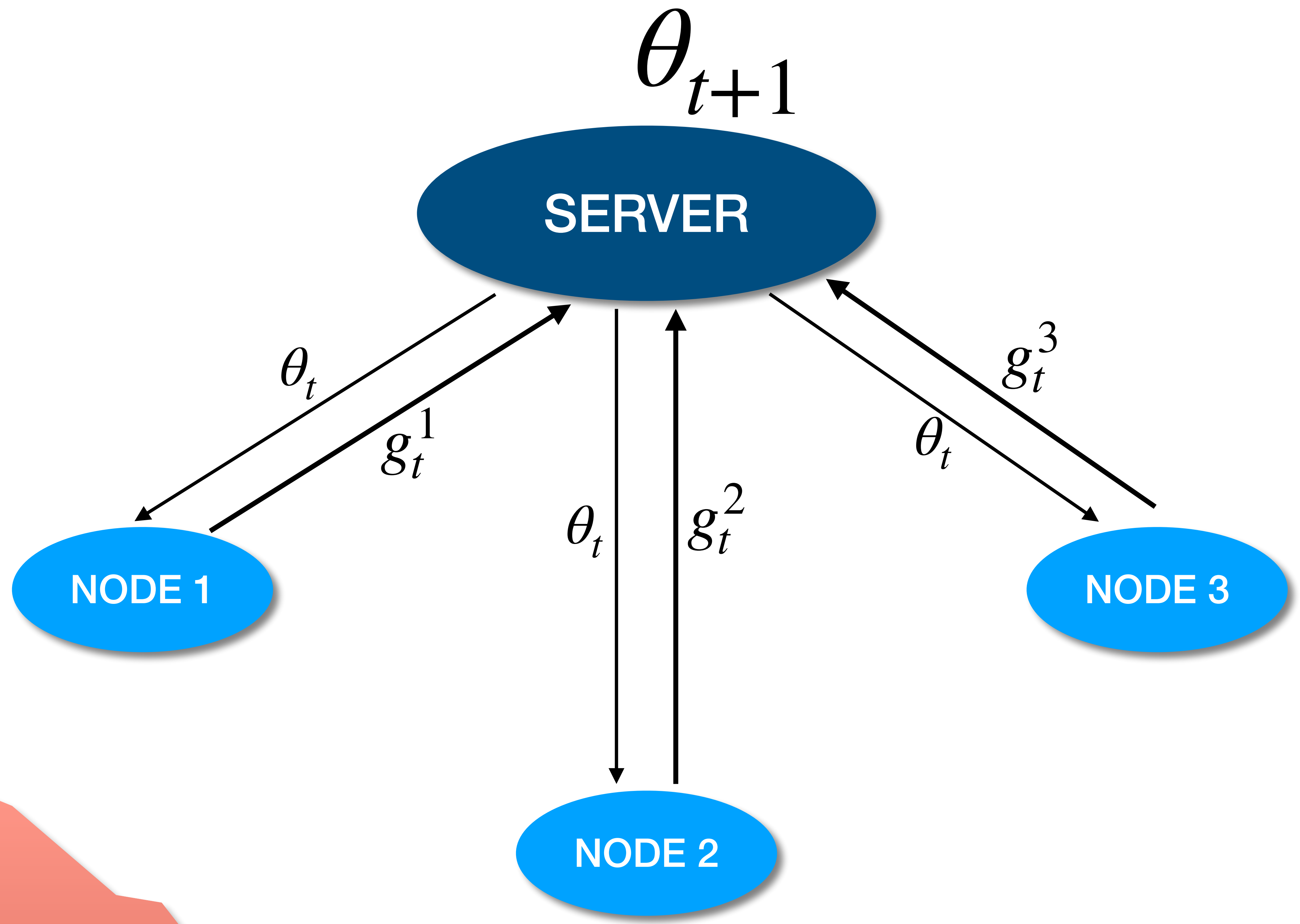
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

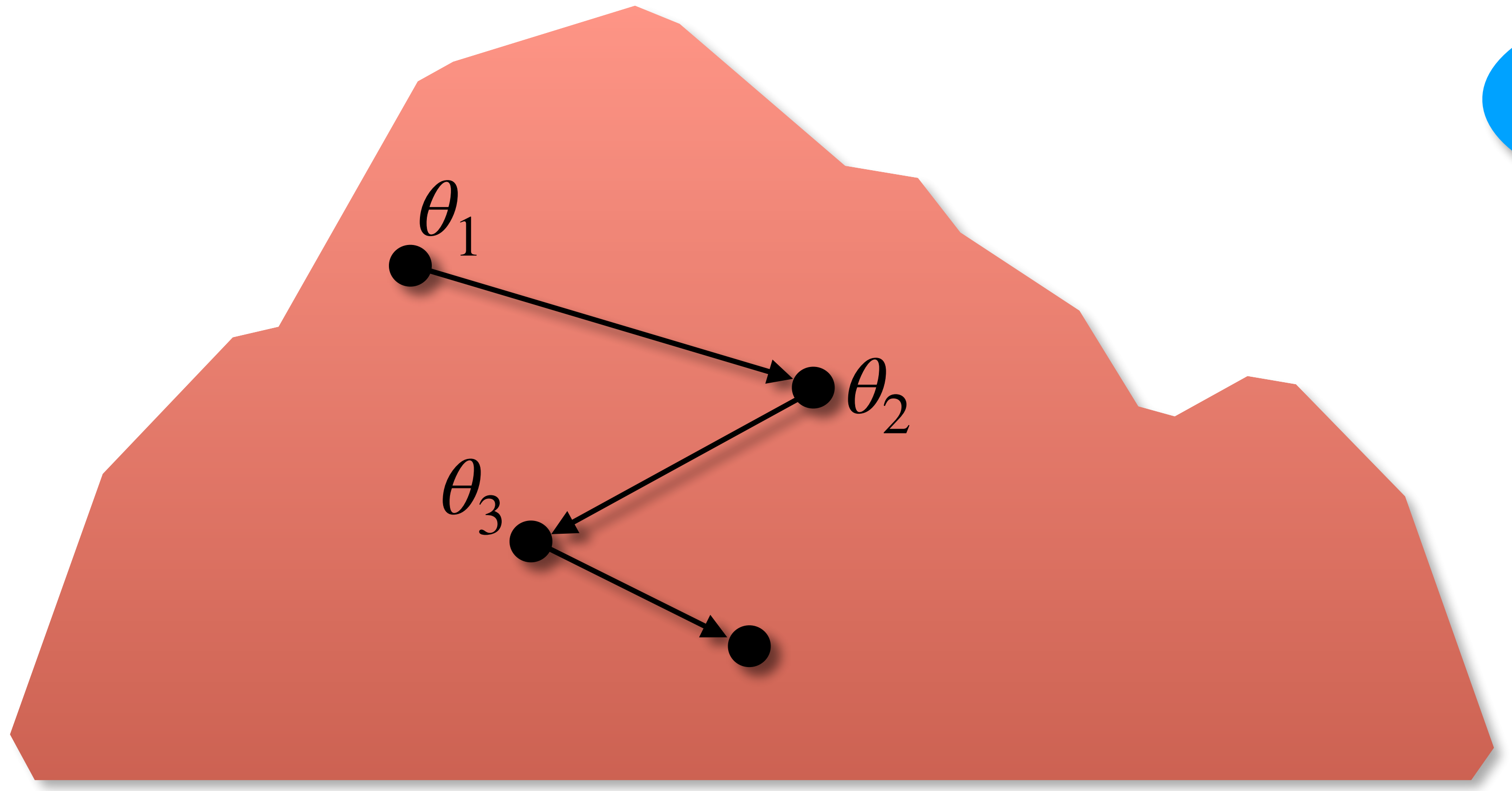
True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

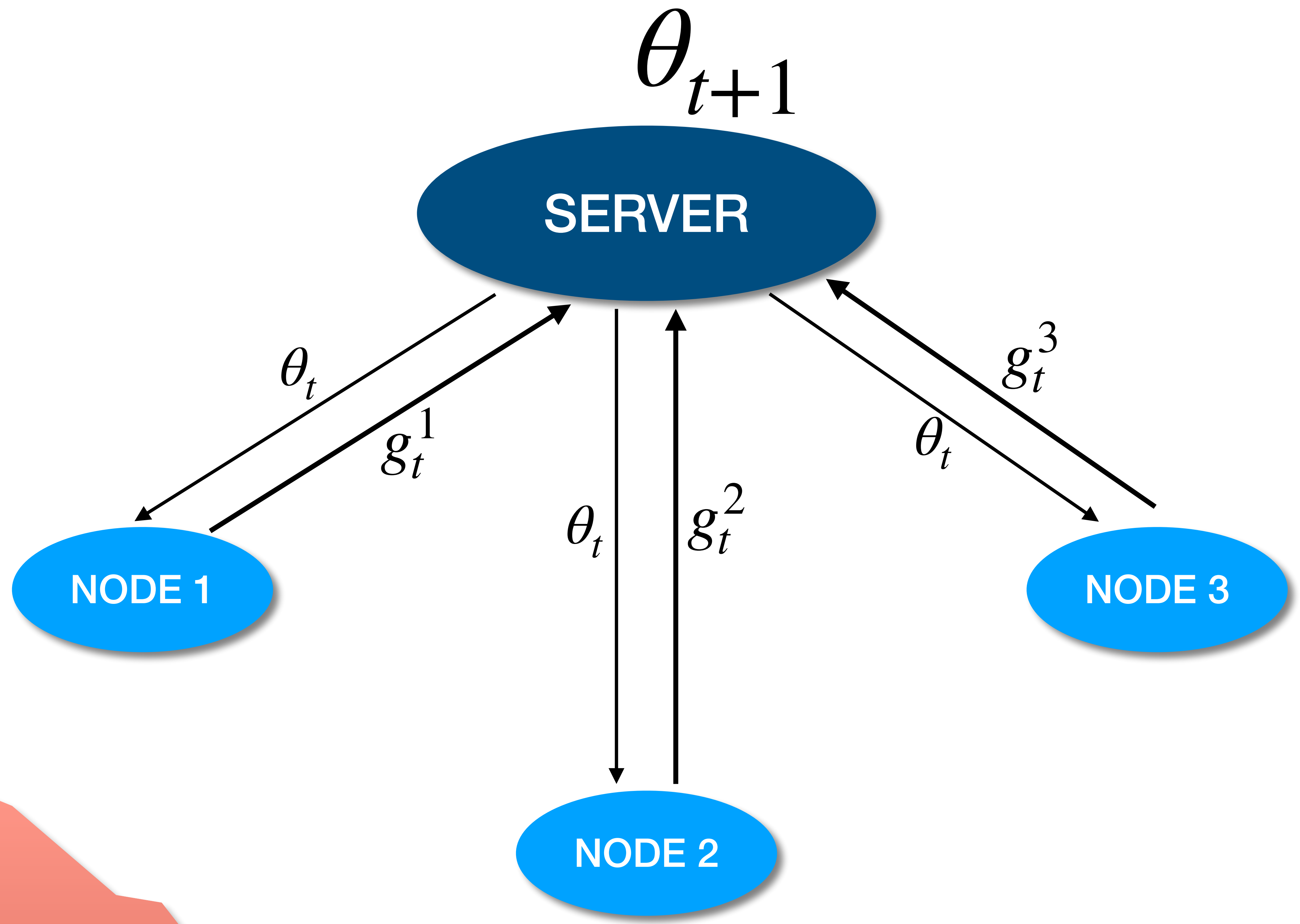
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

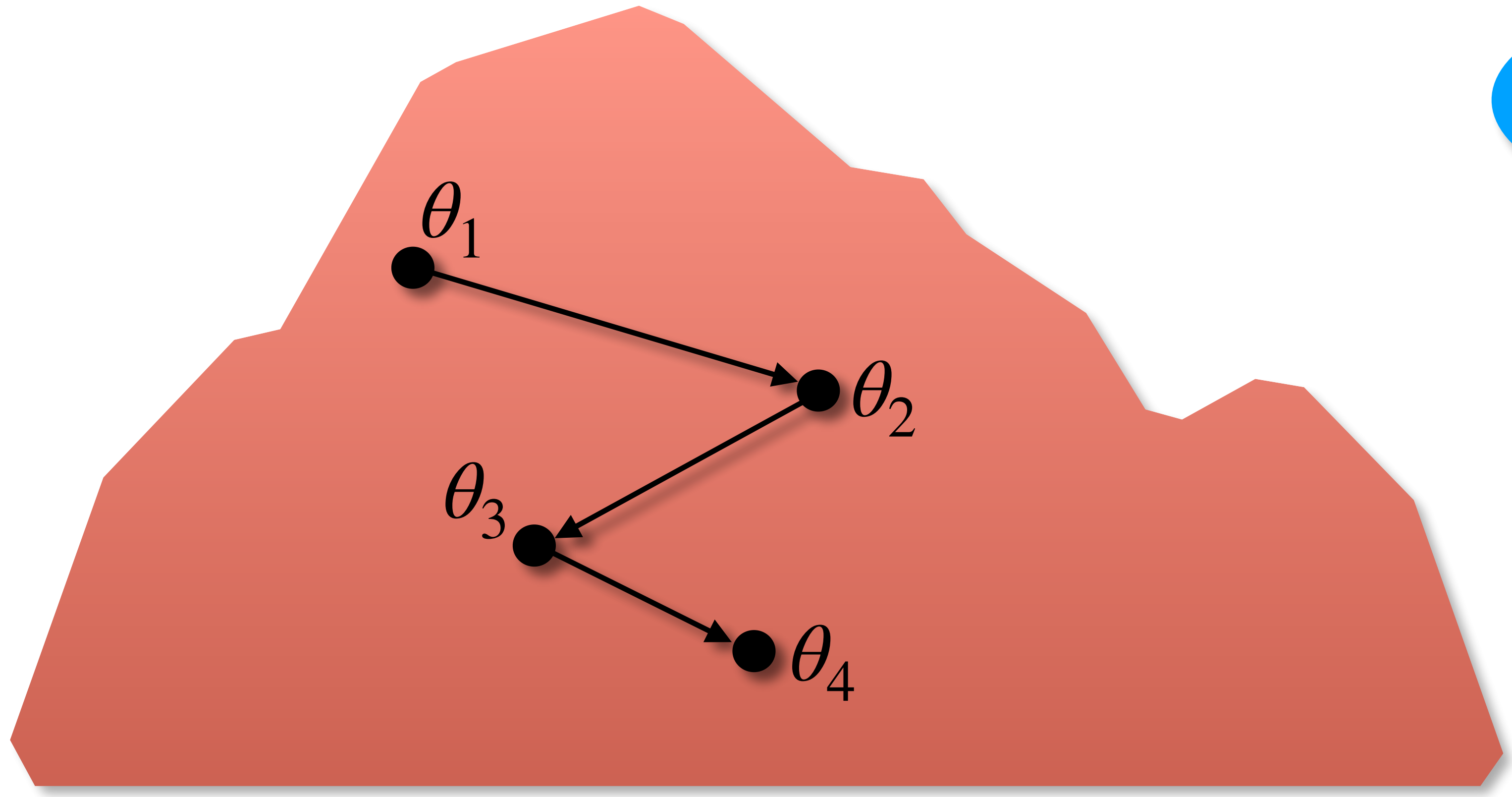
True gradient

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

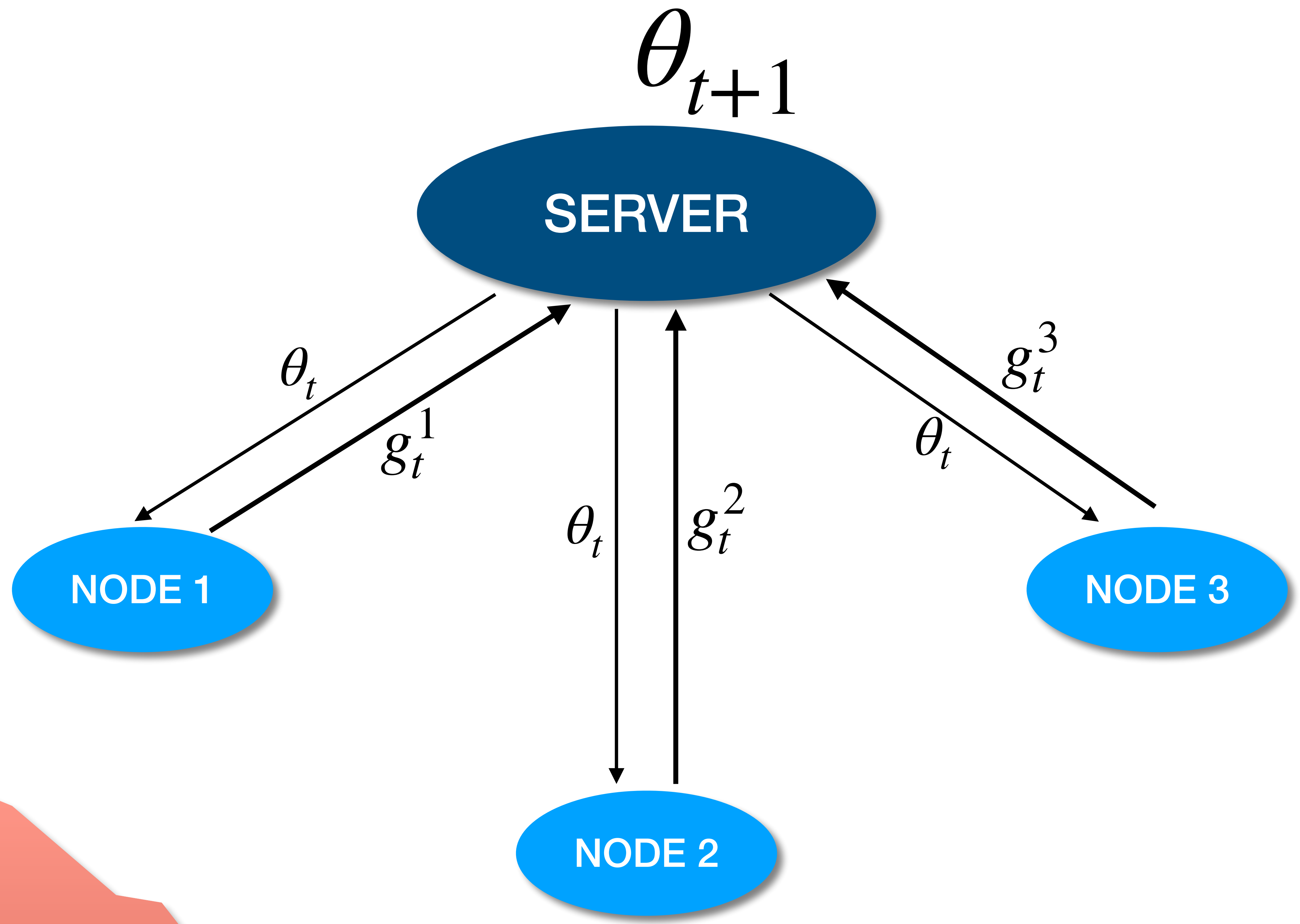
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

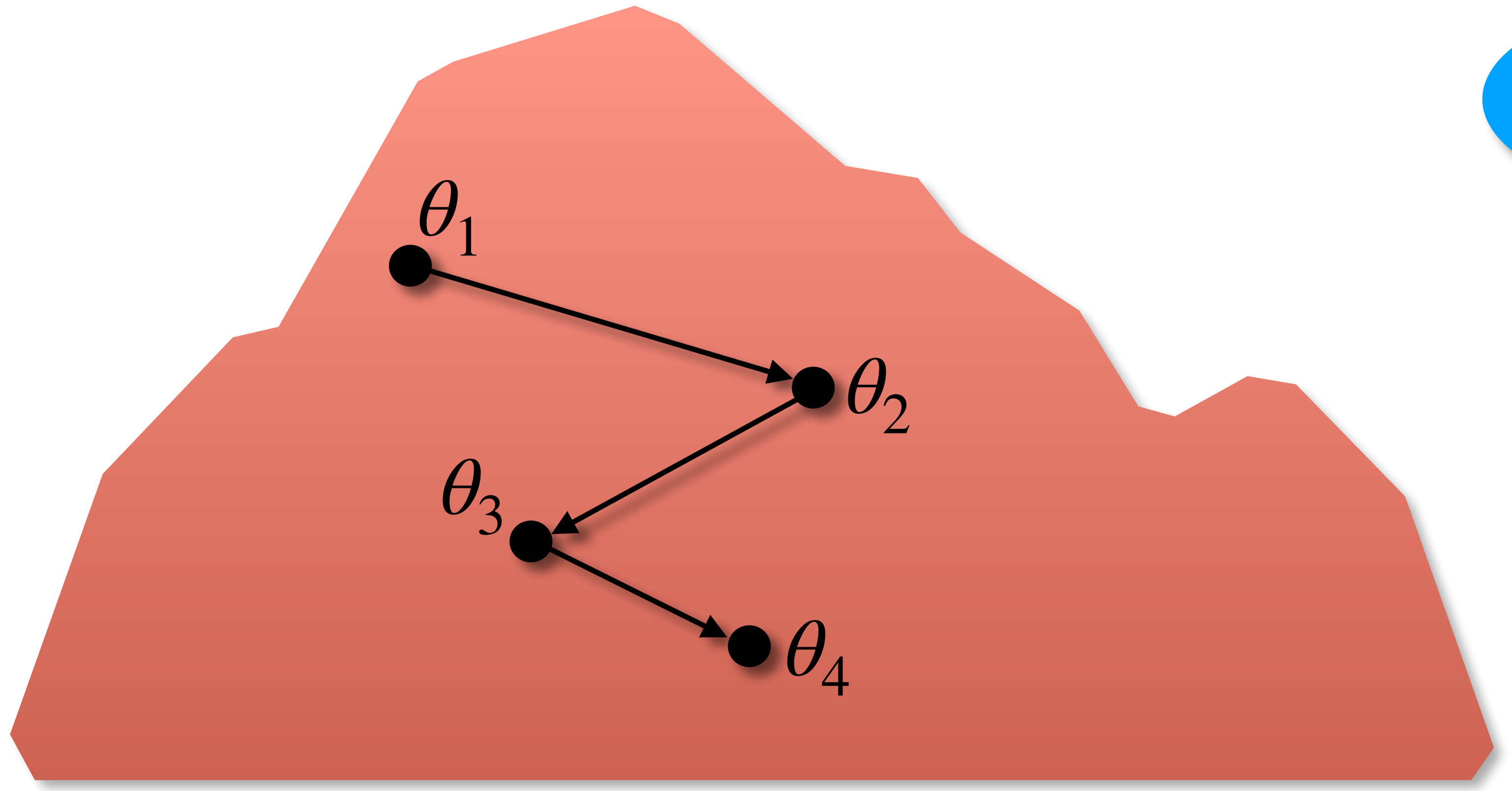
True gradient

Server averages the gradients, $\hat{g}_t = \frac{1}{n} \sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

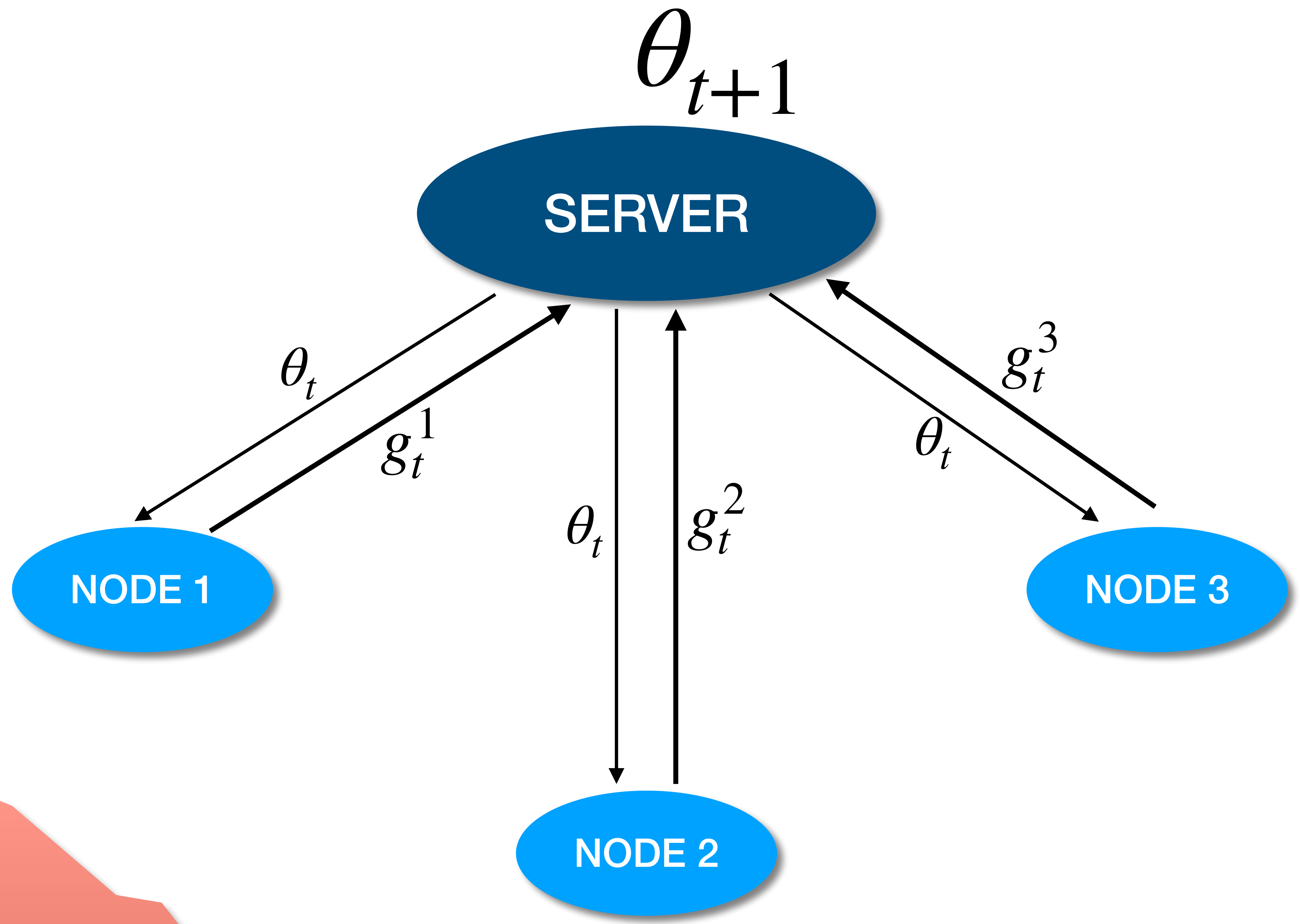
True gradient

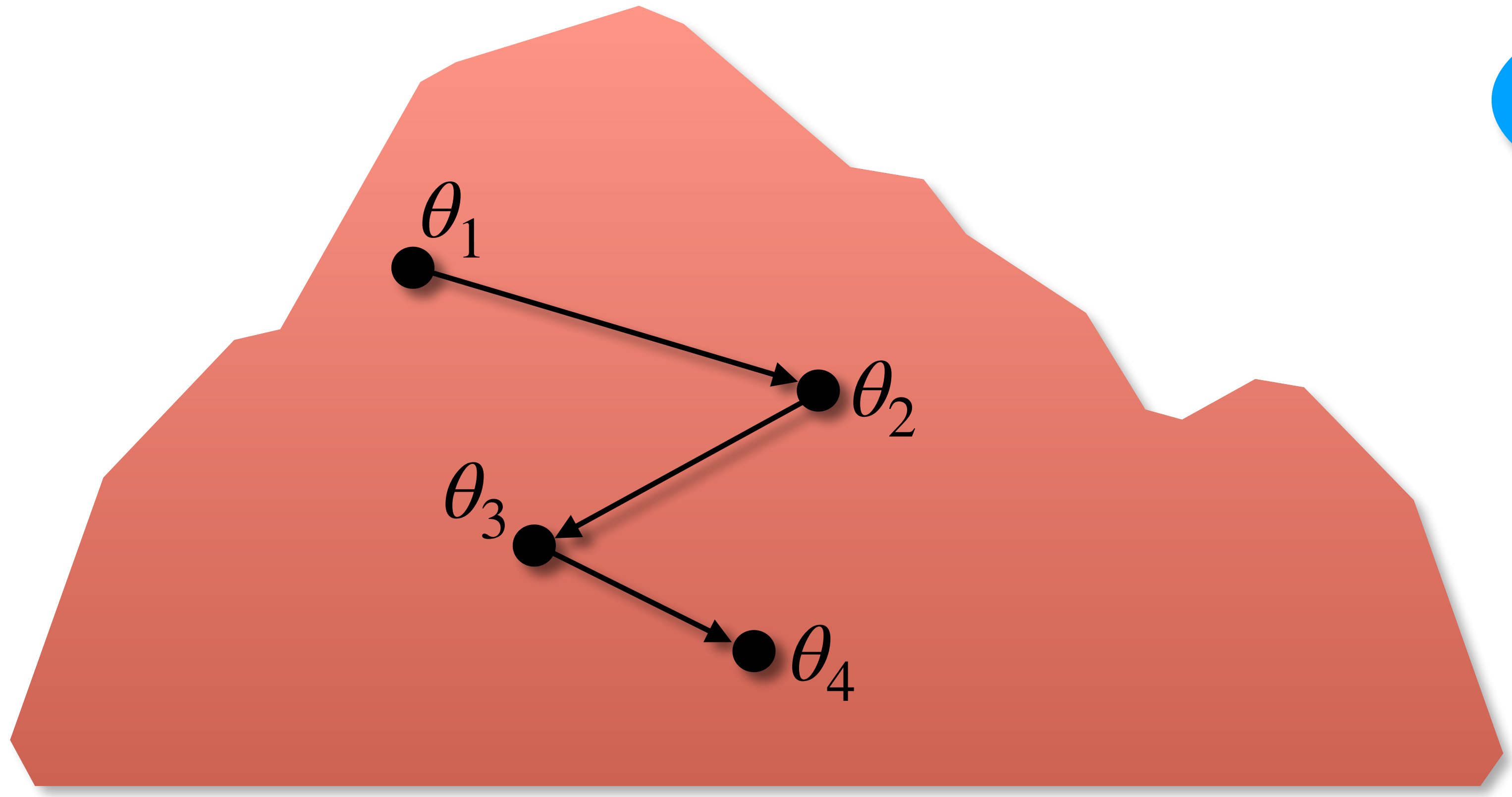
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

True gradient

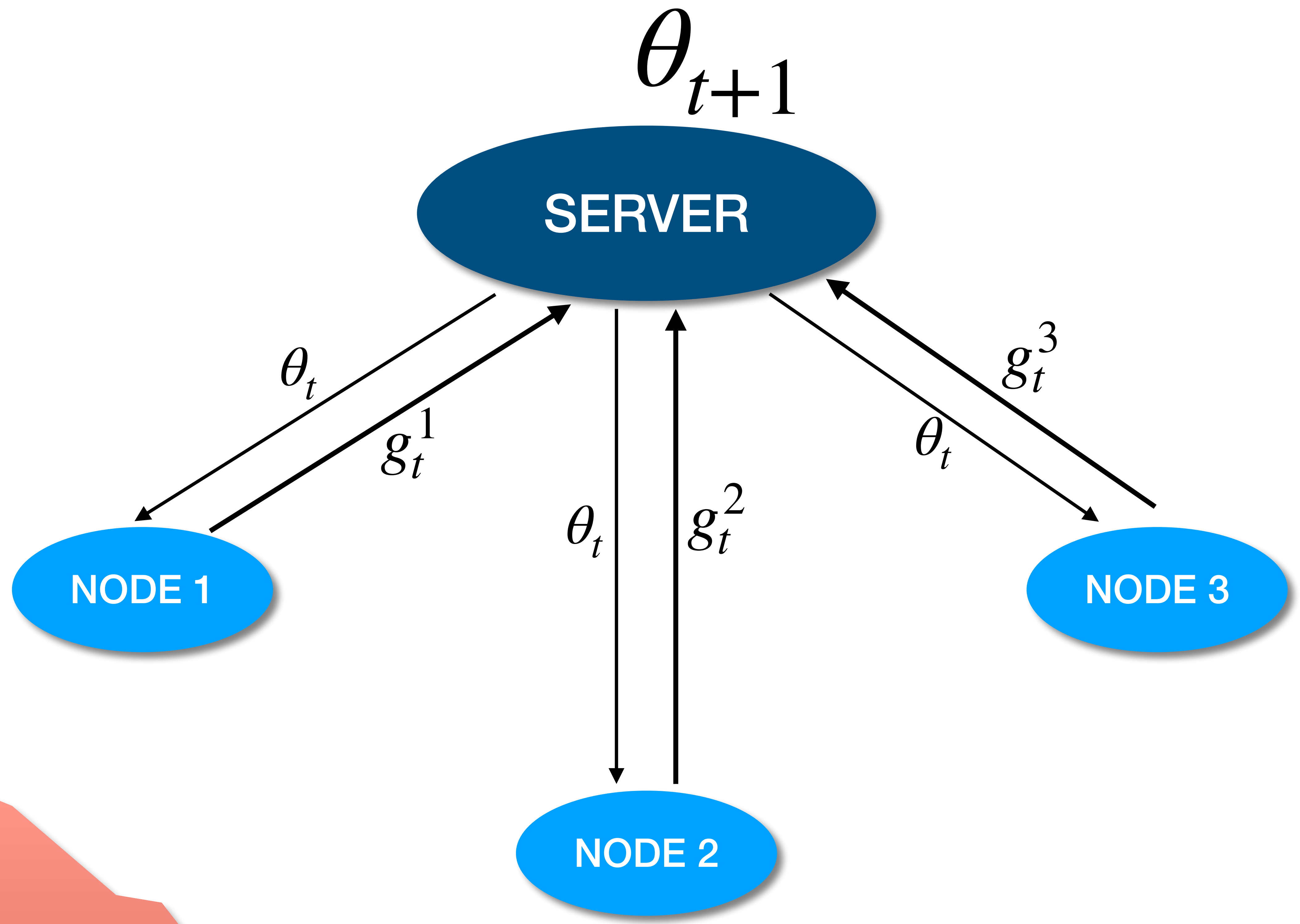
$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \le \sigma^2 < \infty$$

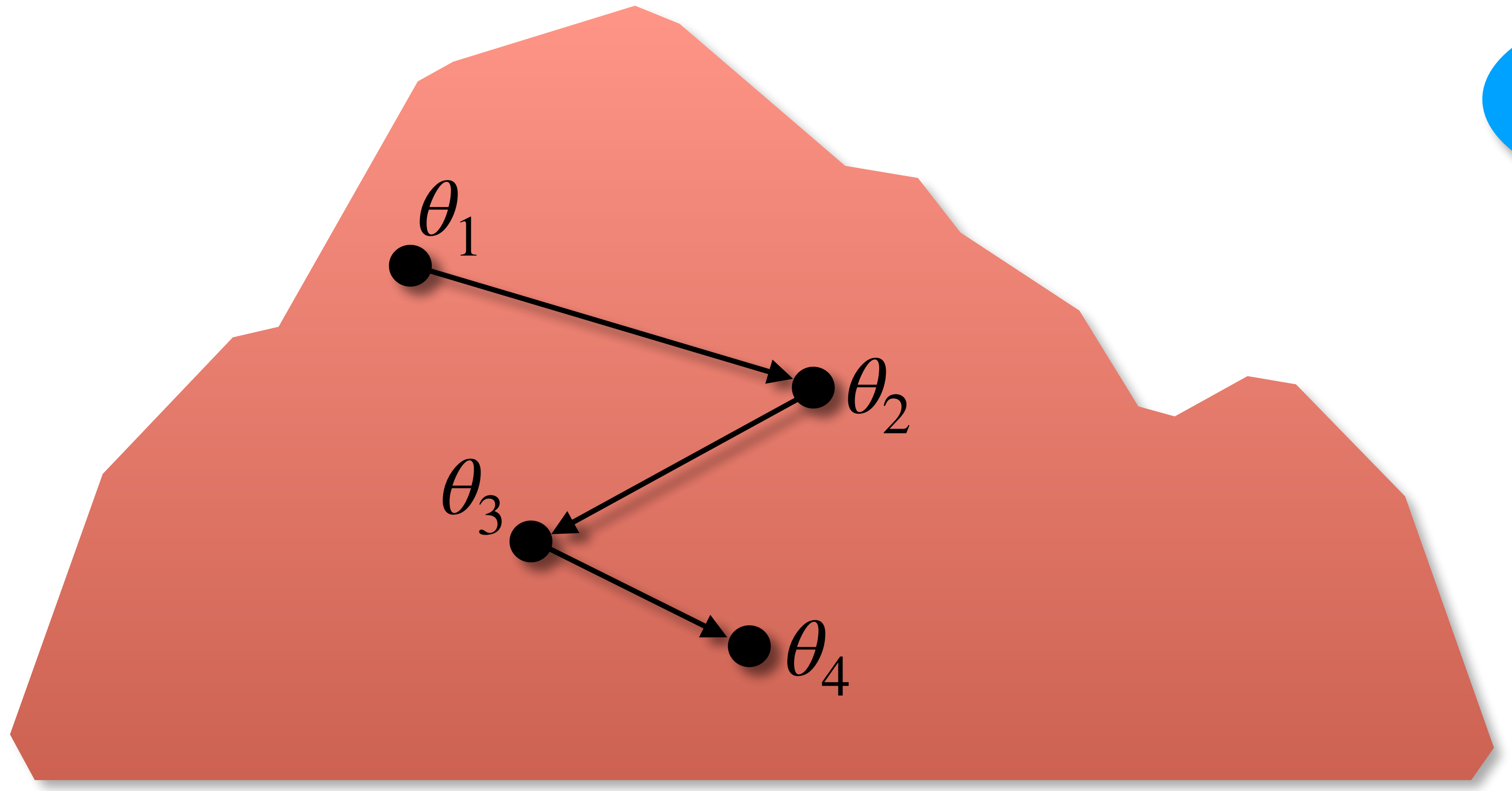
reduces variance by factor of n

Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

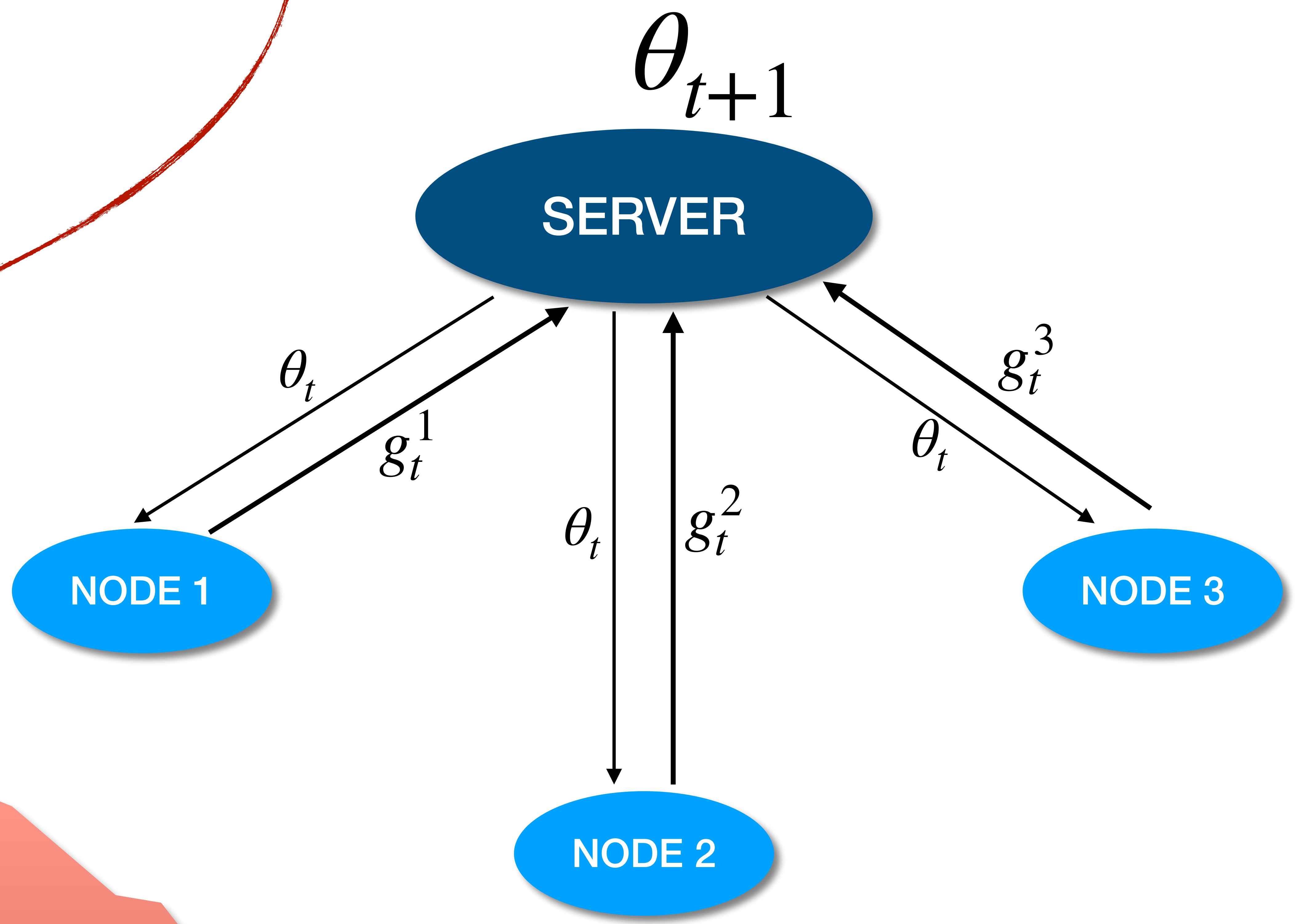
True gradient

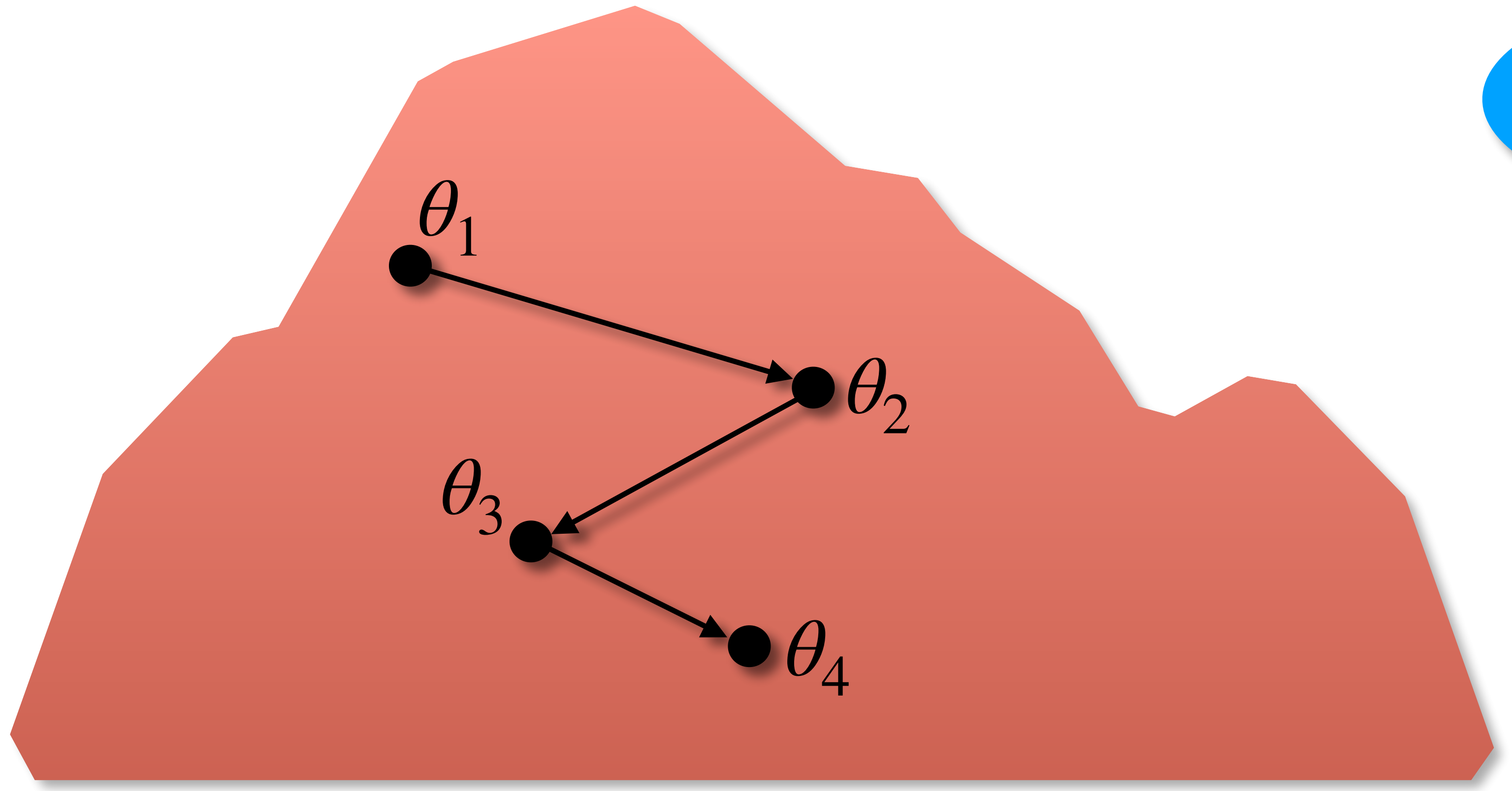
reduces variance by factor of n

Server averages the gradients, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \widehat{g}_t$





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

True gradient

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

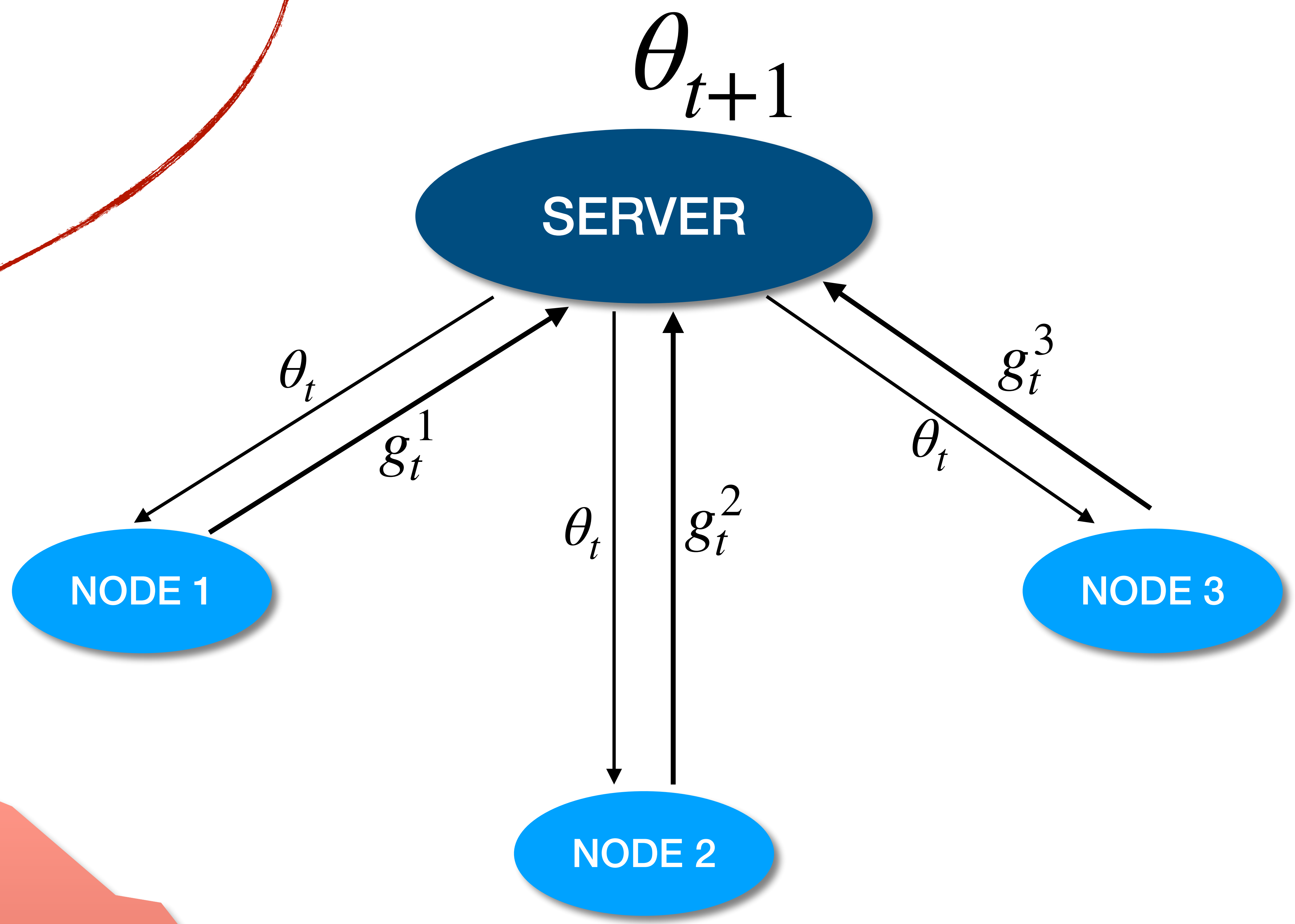
reduces variance by factor of n

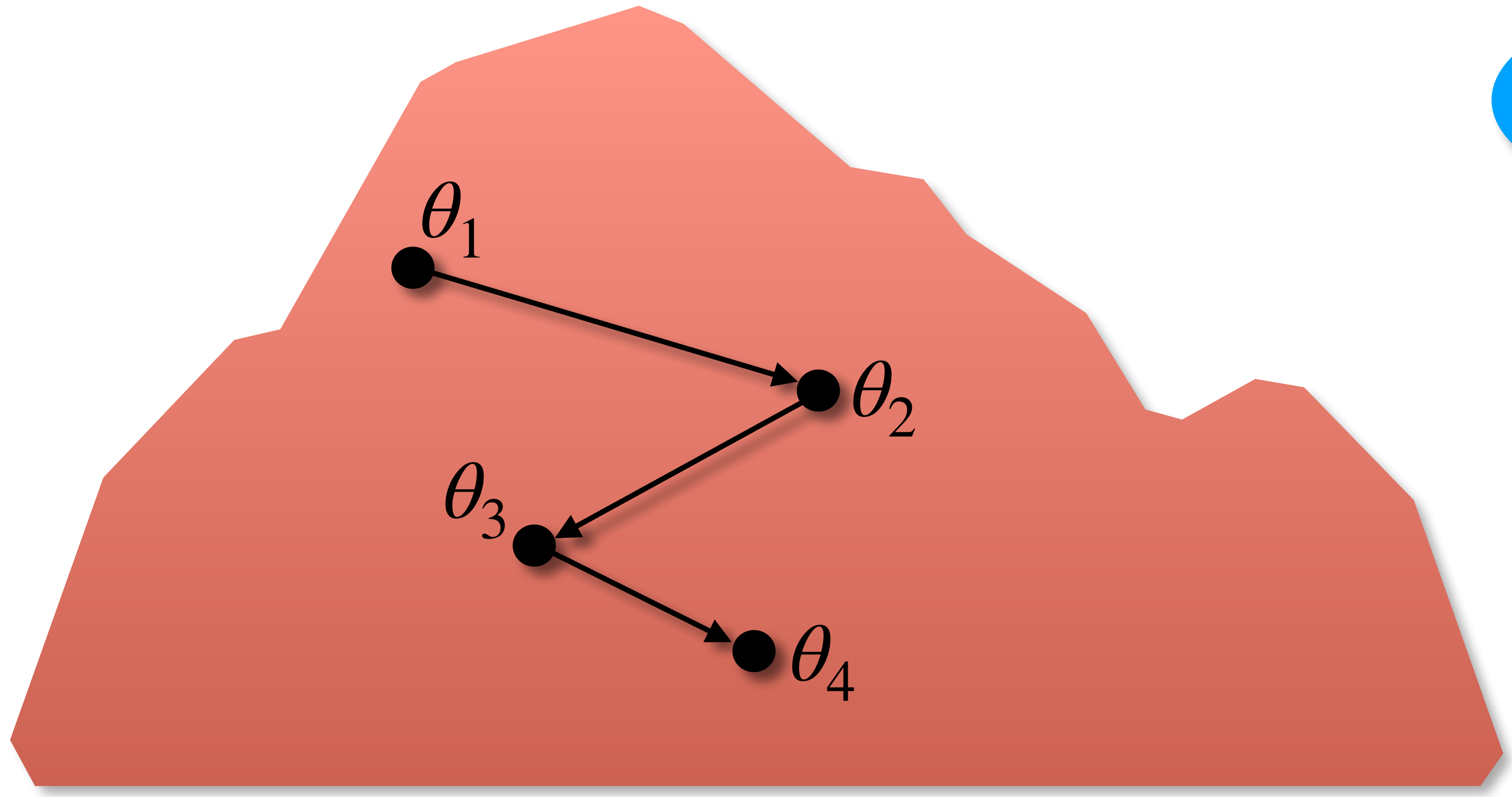
Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$

Upon T iterations





Distributed SGD

Divides the workload per machine by the total size of the system

Nodes query *stochastic gradients* with bounded variance

$$g_t^i = \nabla Q(\theta_t) + u_t^i \quad ; \quad u_t^i \sim \mathcal{U}_t$$

$$\mathbb{E}\left[\mathcal{U}_t\right] = 0 \quad ; \quad \mathbb{E}\left[\|\mathcal{U}_t\|^2\right] \leq \sigma^2 < \infty$$

True gradient

reduces variance by factor of n

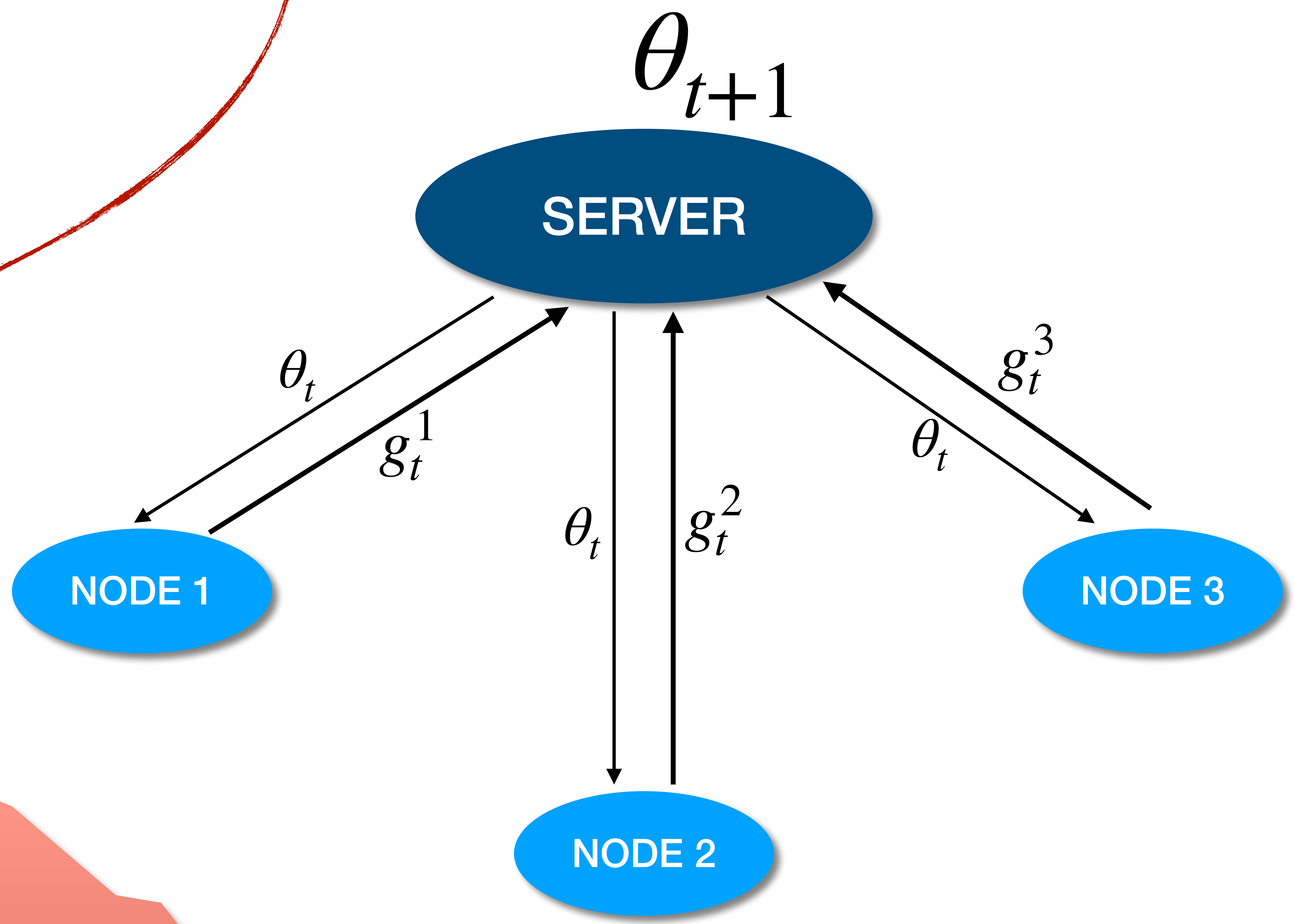
Server averages the gradients, $\hat{g}_t = \frac{1}{n}\sum_i g_t^i$

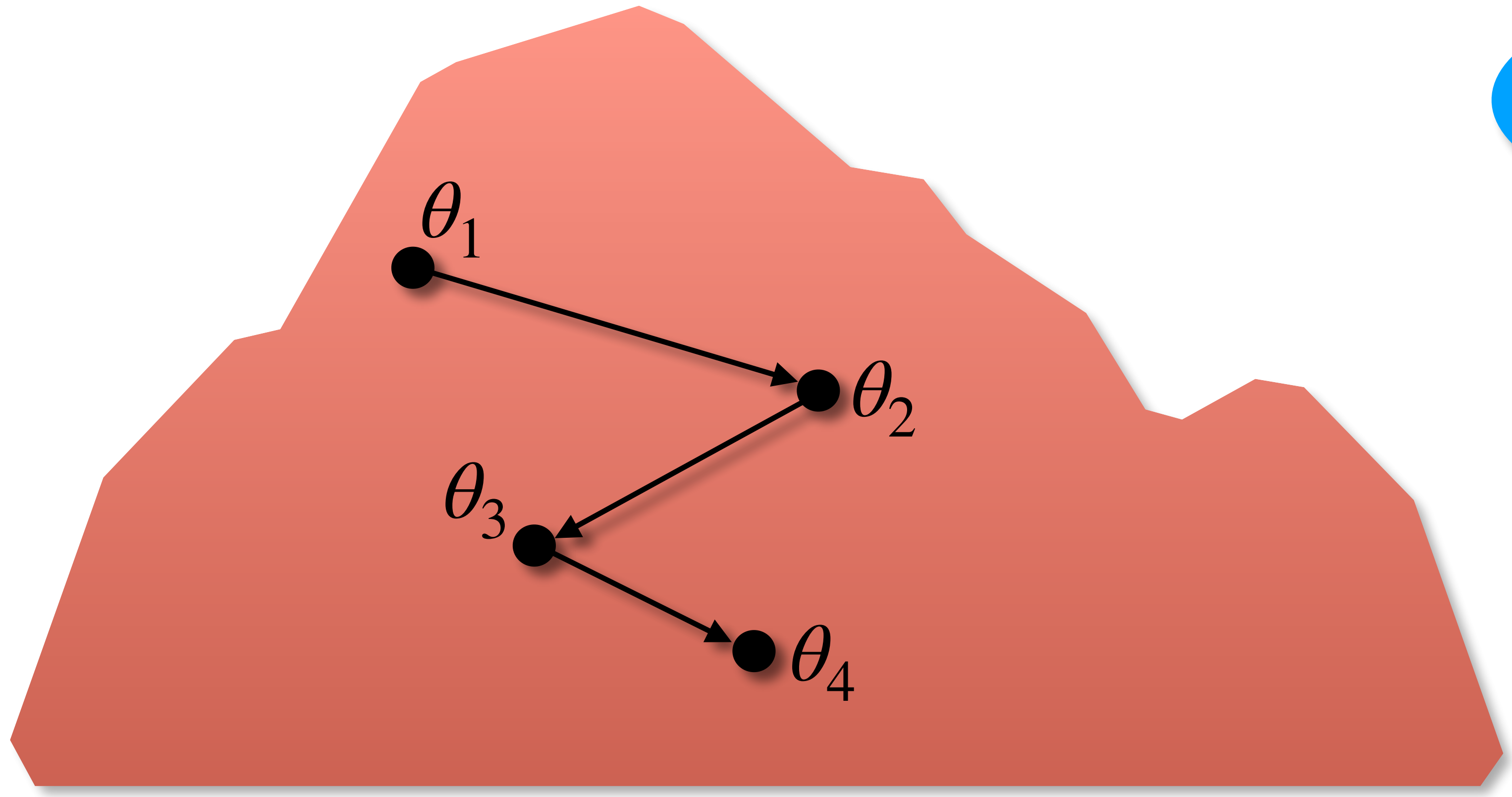
Learning rate

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \hat{g}_t$

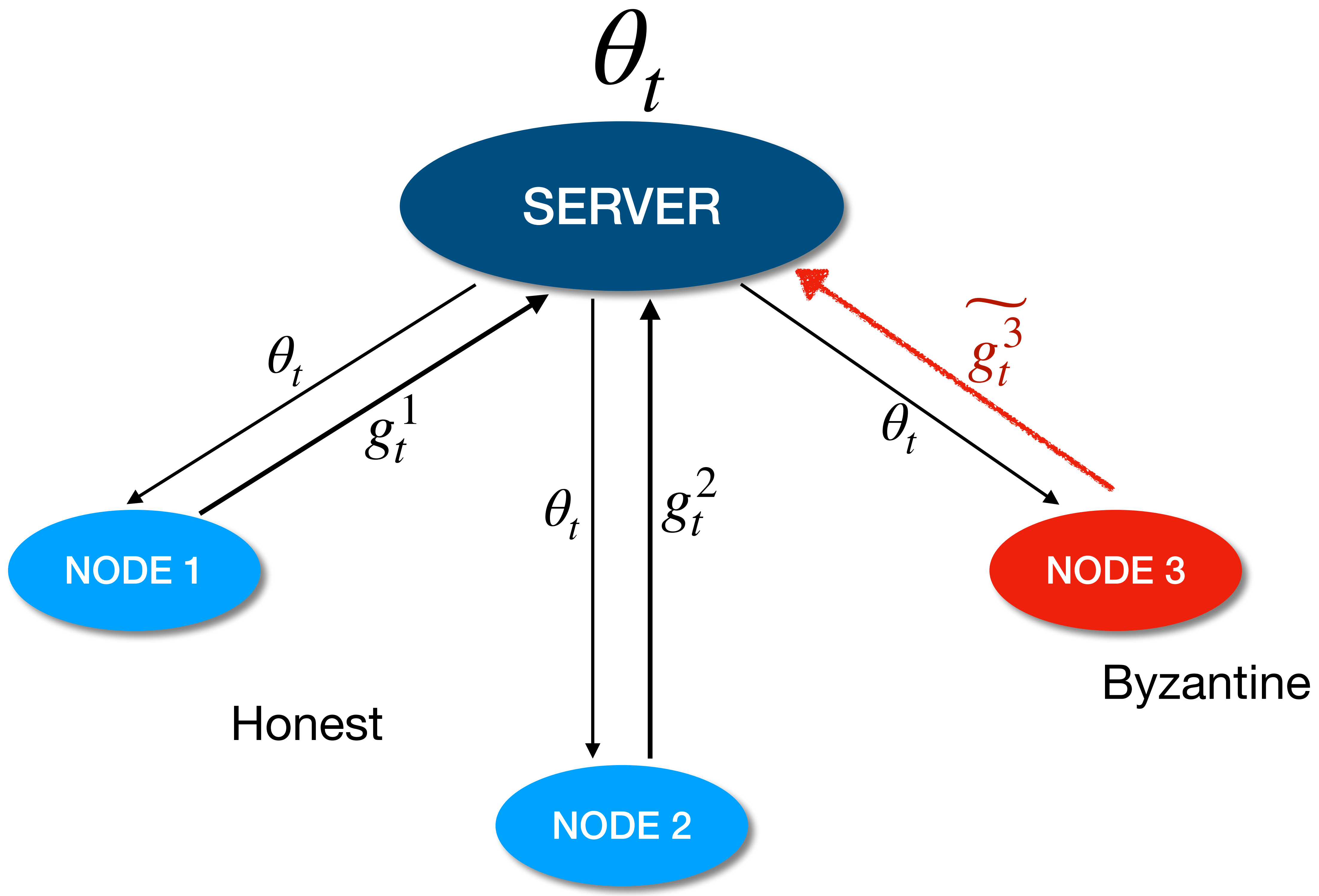
Upon T iterations

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\|\nabla Q(\theta_t)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\frac{\sigma^2}{nT}}\right)$$



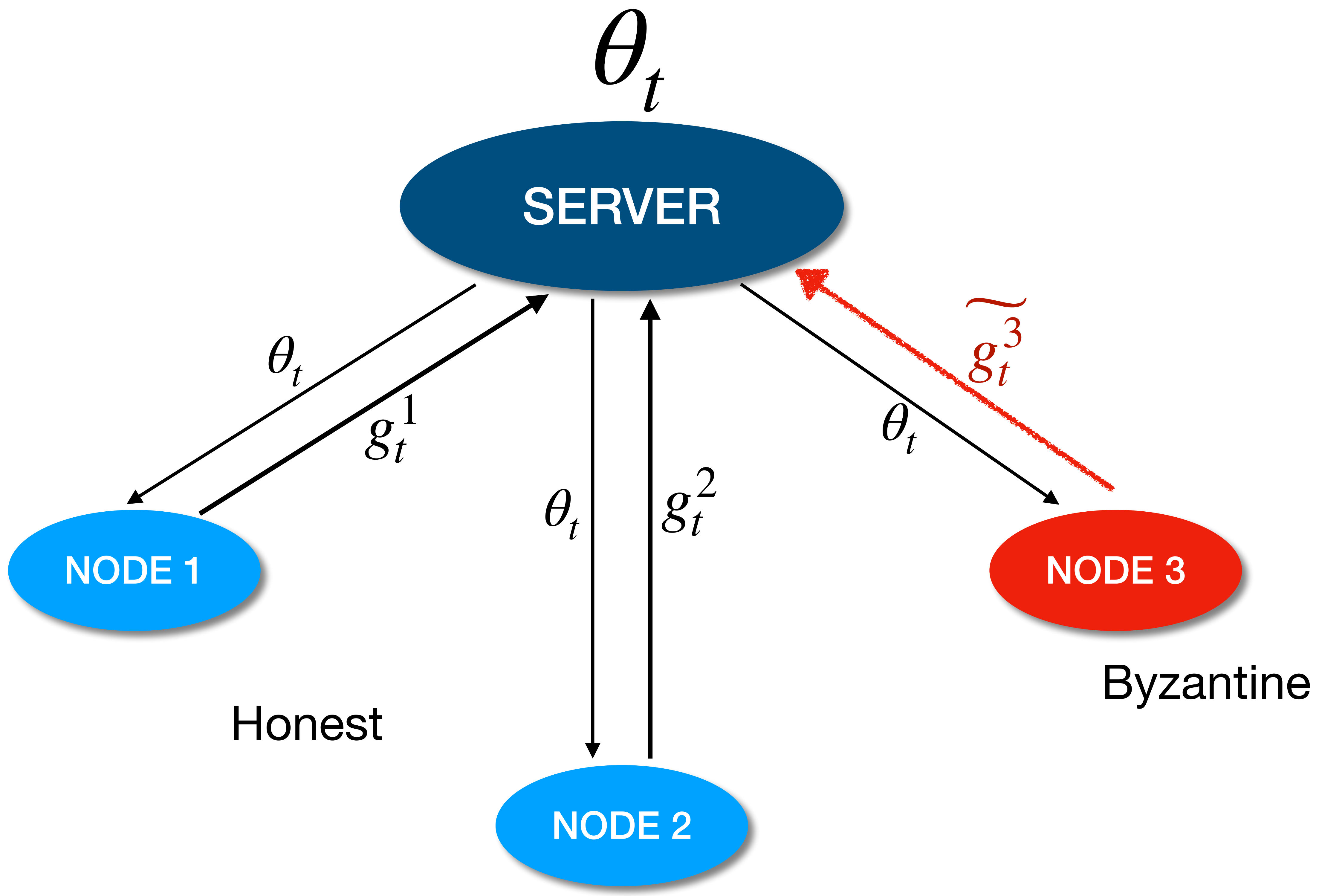


Byzantine Resilience in Distributed Learning



Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

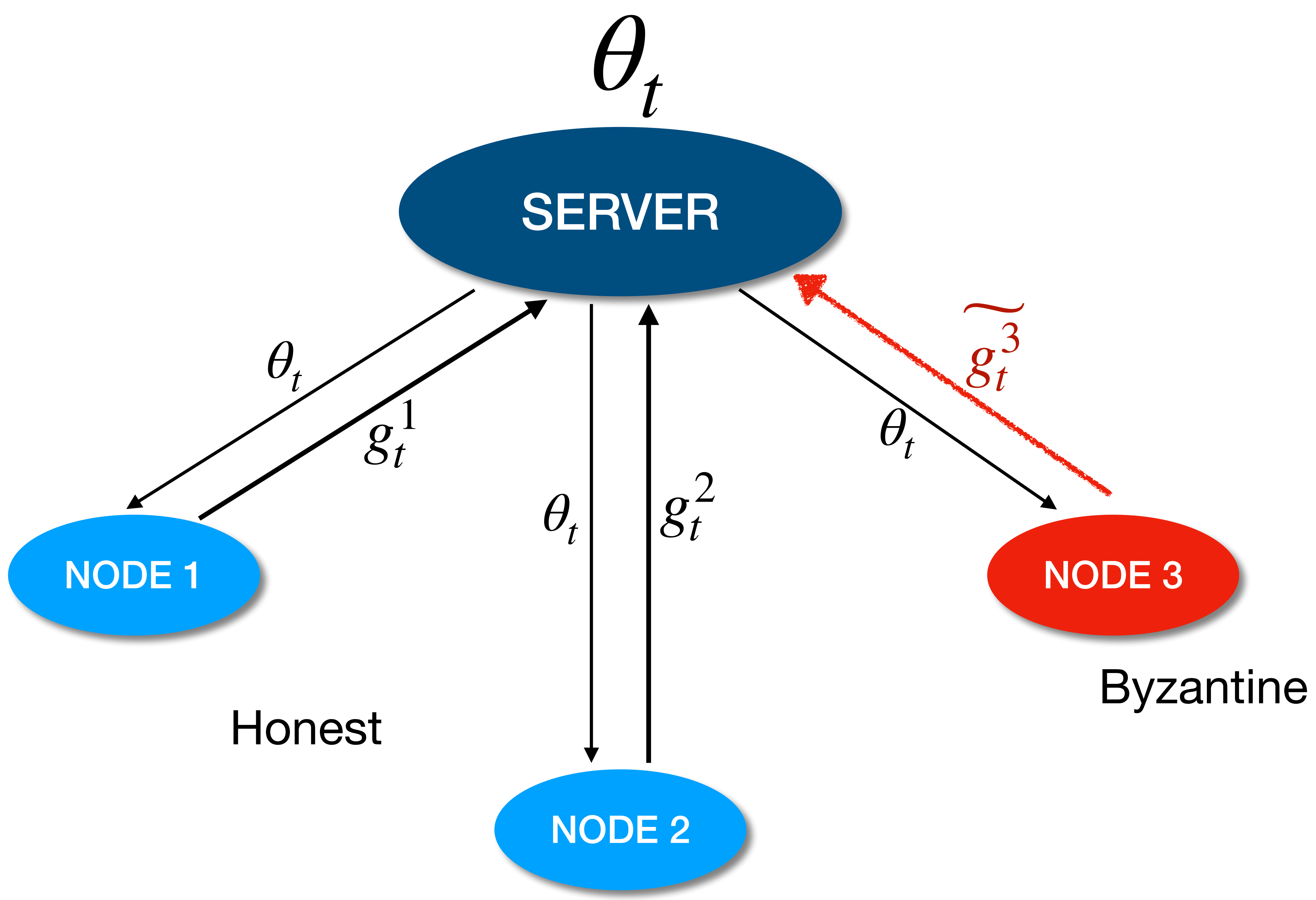


Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$



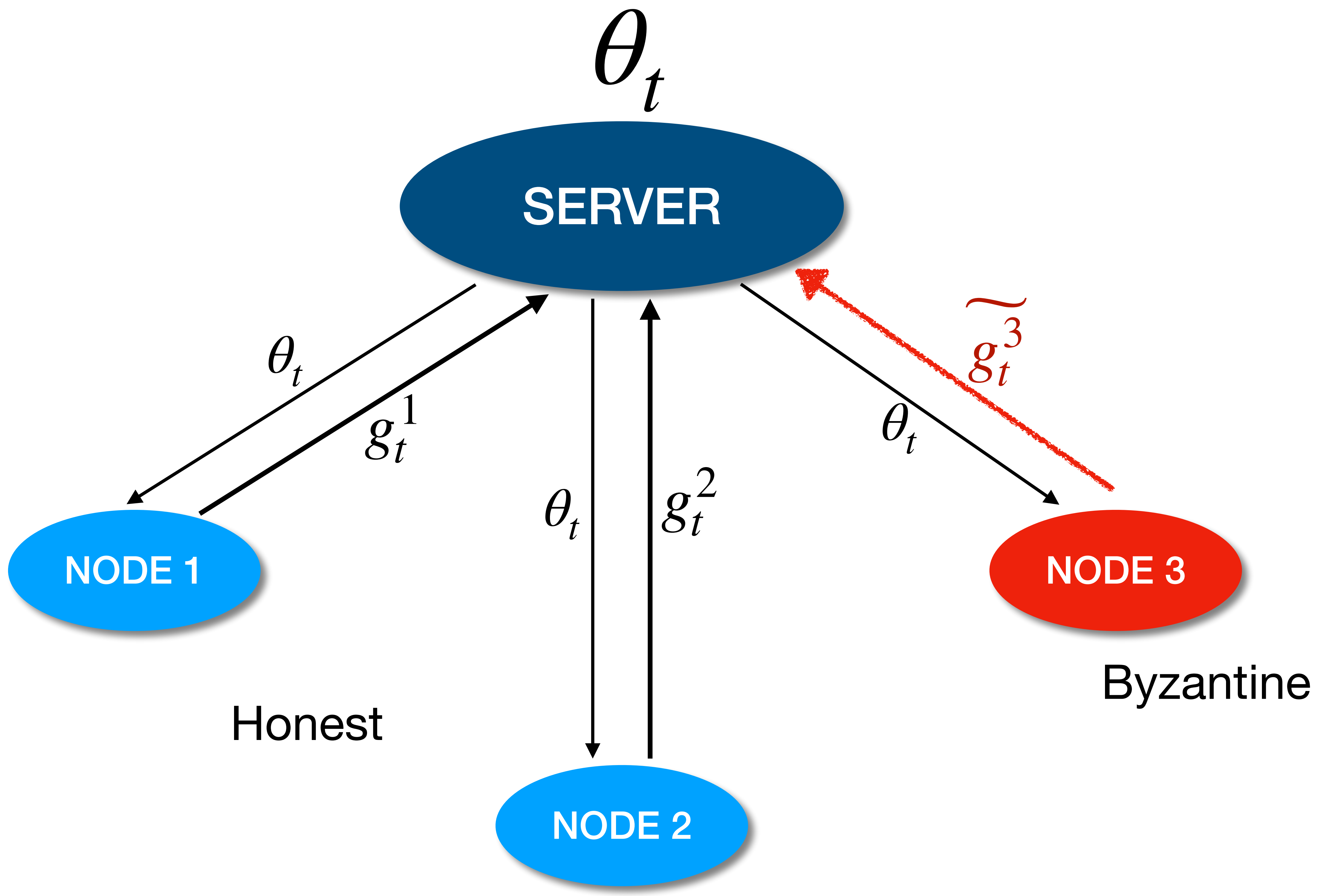
Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*



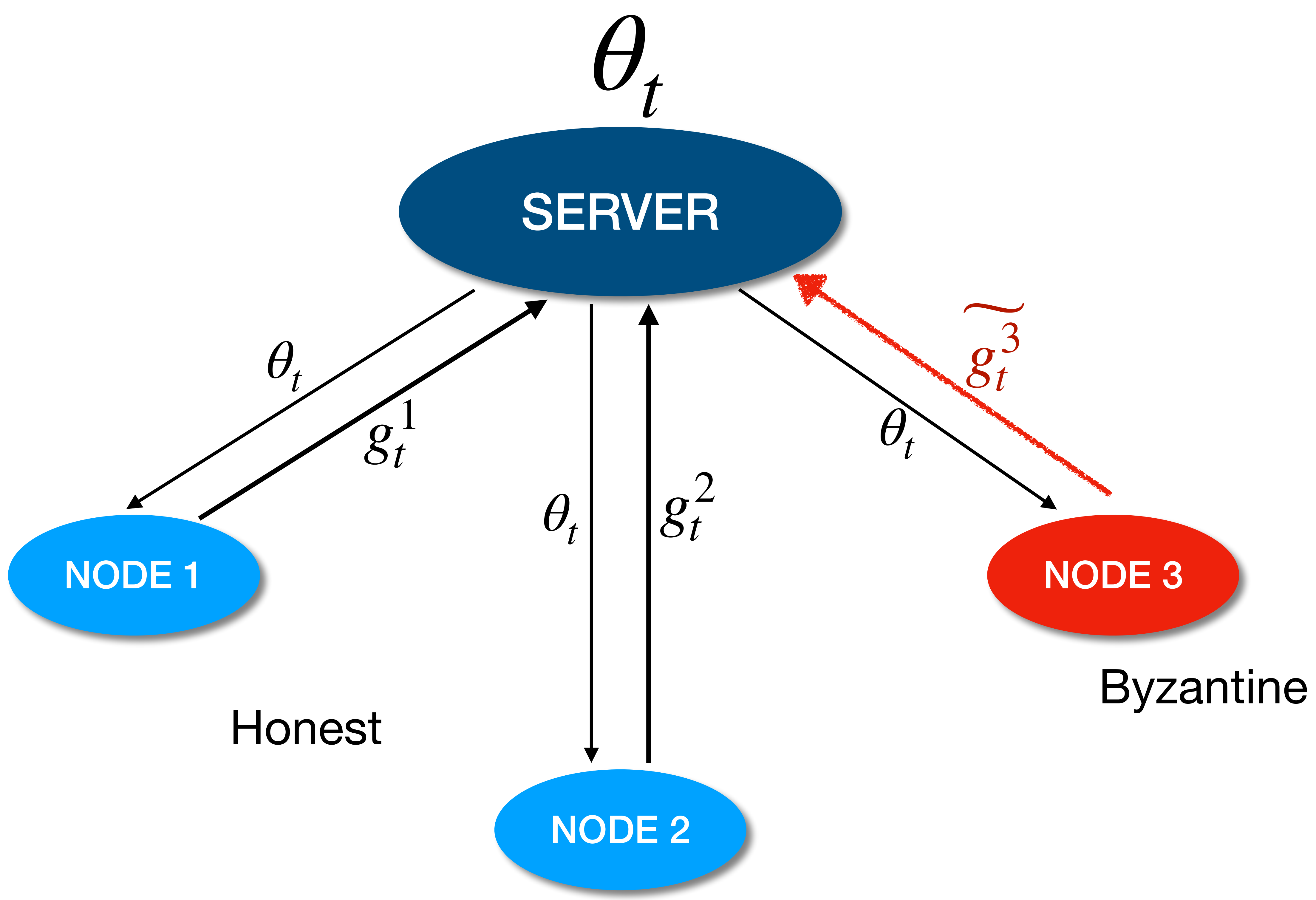
Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*



* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Byzantine Resilience in Distributed Learning

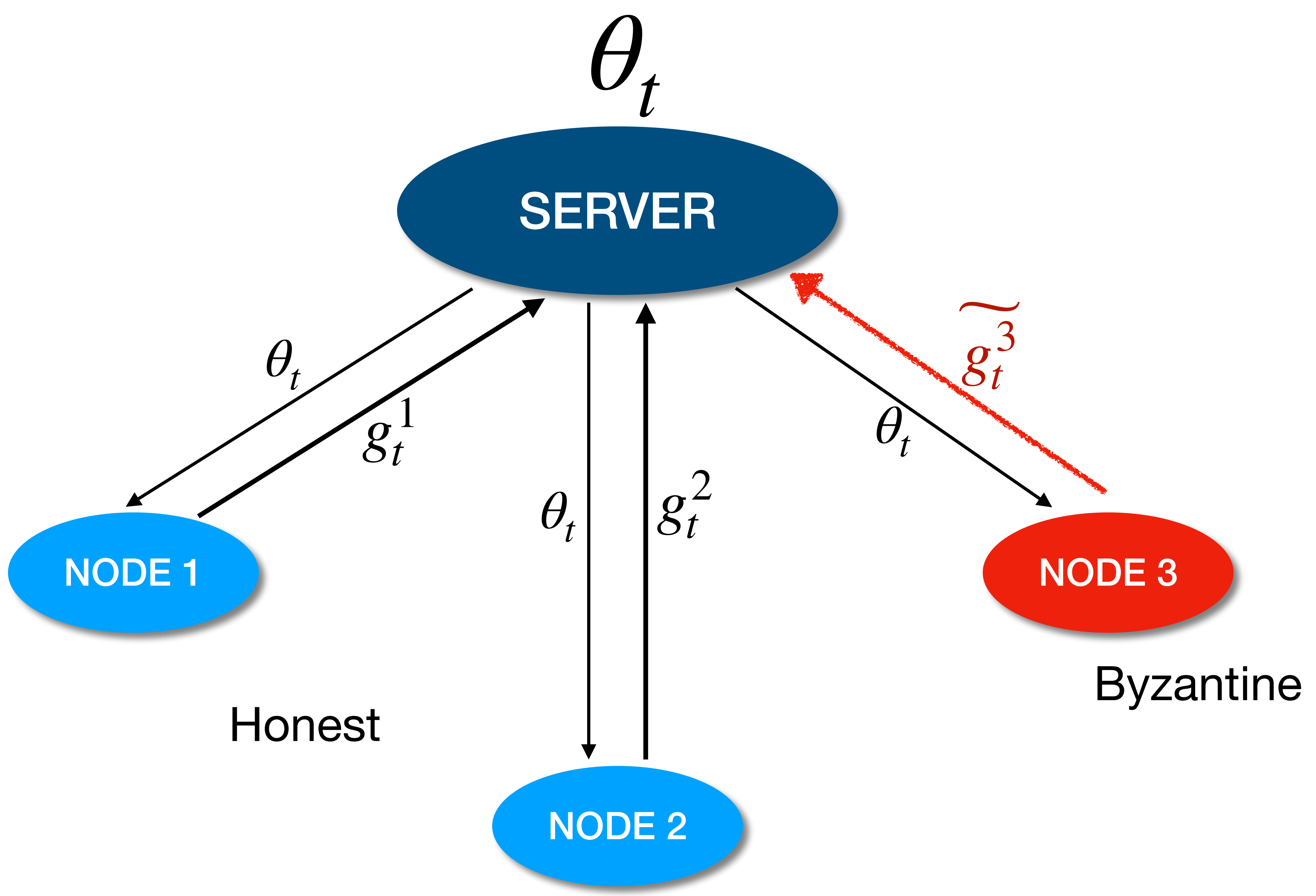
f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*

(f, ϵ)**-Resilience**



* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

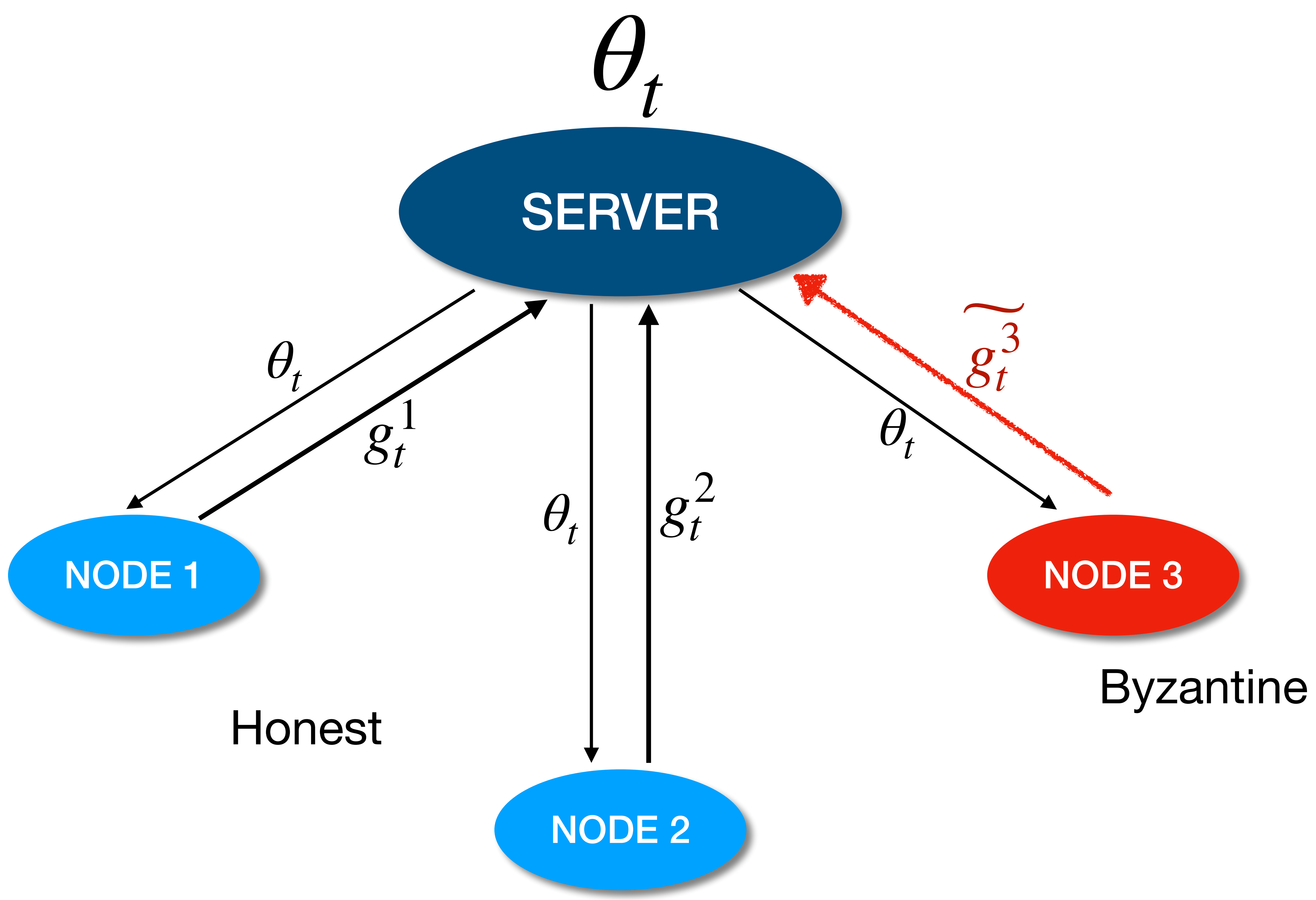
$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*

(f, ϵ)-Resilience

An algorithm that allows the honest nodes to compute $\widehat{\theta}$ such that

$$\|\nabla Q\left(\widehat{\theta}\right)\|^2 \leq \epsilon$$



* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

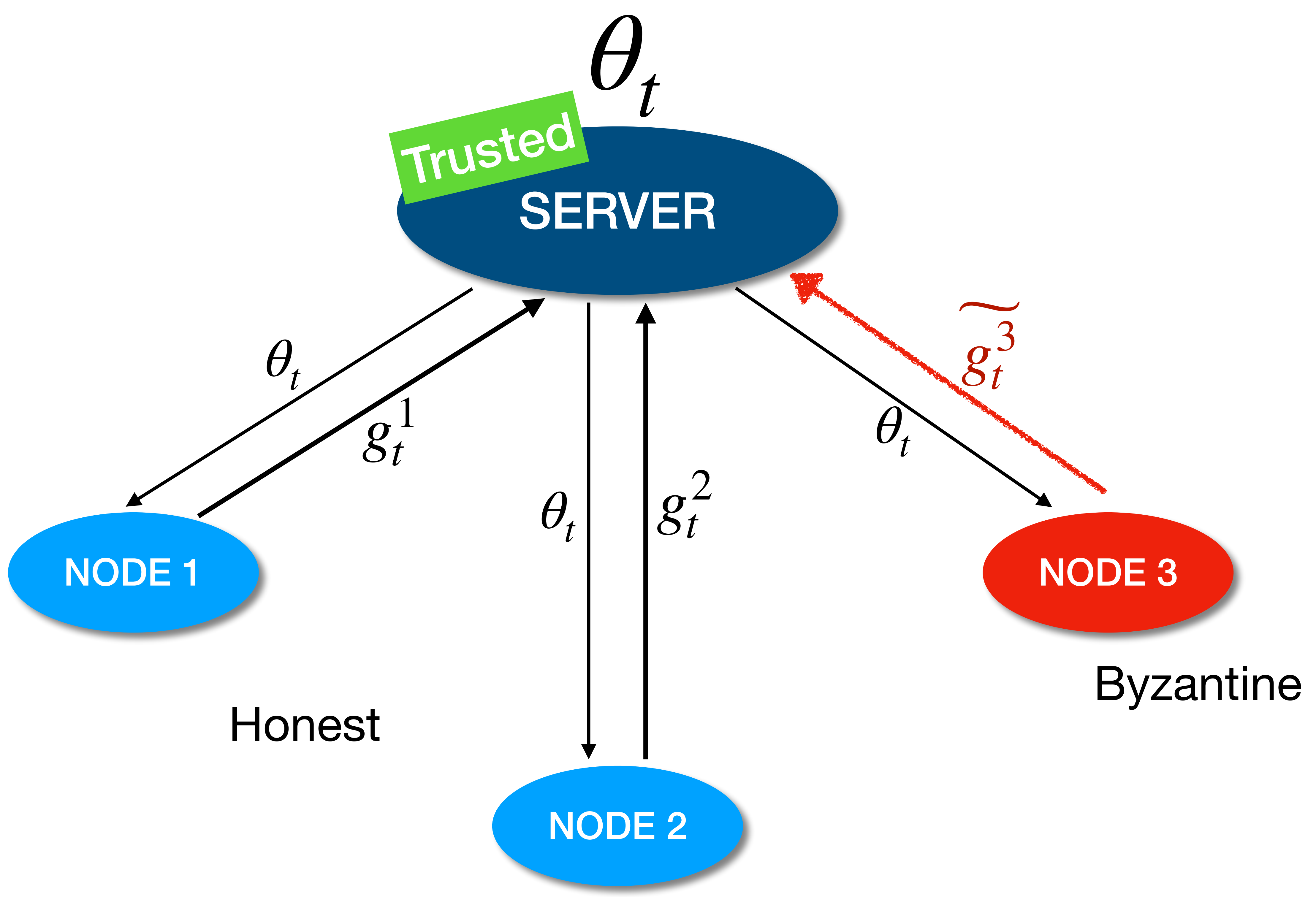
$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*

(f, ϵ)-Resilience

An algorithm that allows the honest nodes to compute $\hat{\theta}$ such that

$$\|\nabla Q\left(\hat{\theta}\right)\|^2 \leq \epsilon$$



* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

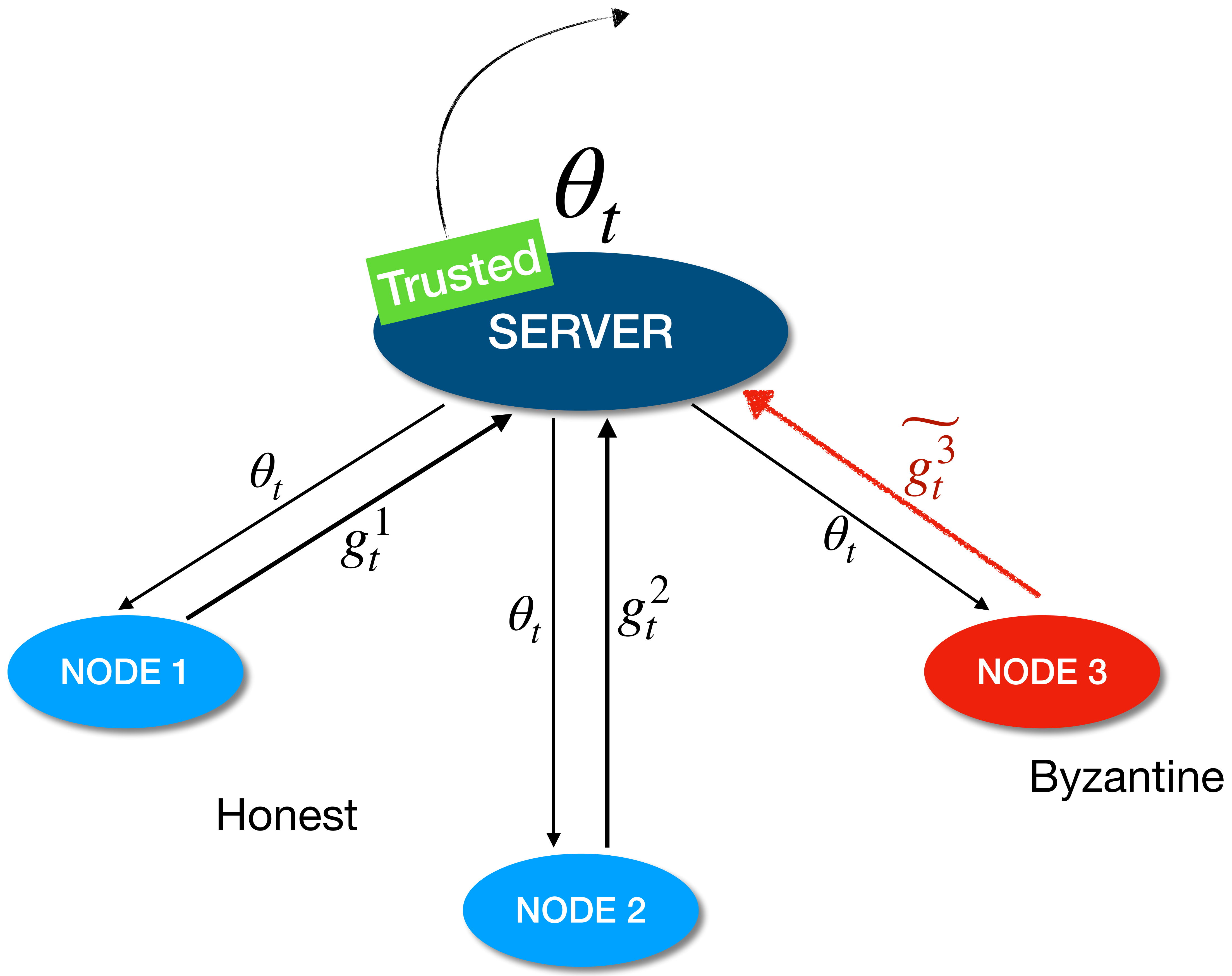
$$\theta^* \in \left(\theta \;;\; \nabla Q(\theta) = 0\right)$$

Feasible as nodes are assumed
to sample data from the same distribution*

(f, ϵ)-Resilience

An algorithm that allows the honest nodes to compute $\widehat{\theta}$ such that

$$\|\nabla Q\left(\widehat{\theta}\right)\|^2 \leq \epsilon$$



* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Byzantine Resilience in Distributed Learning

f out of n nodes are Byzantine faulty

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \,;\; \nabla Q(\theta) = 0\right)$$

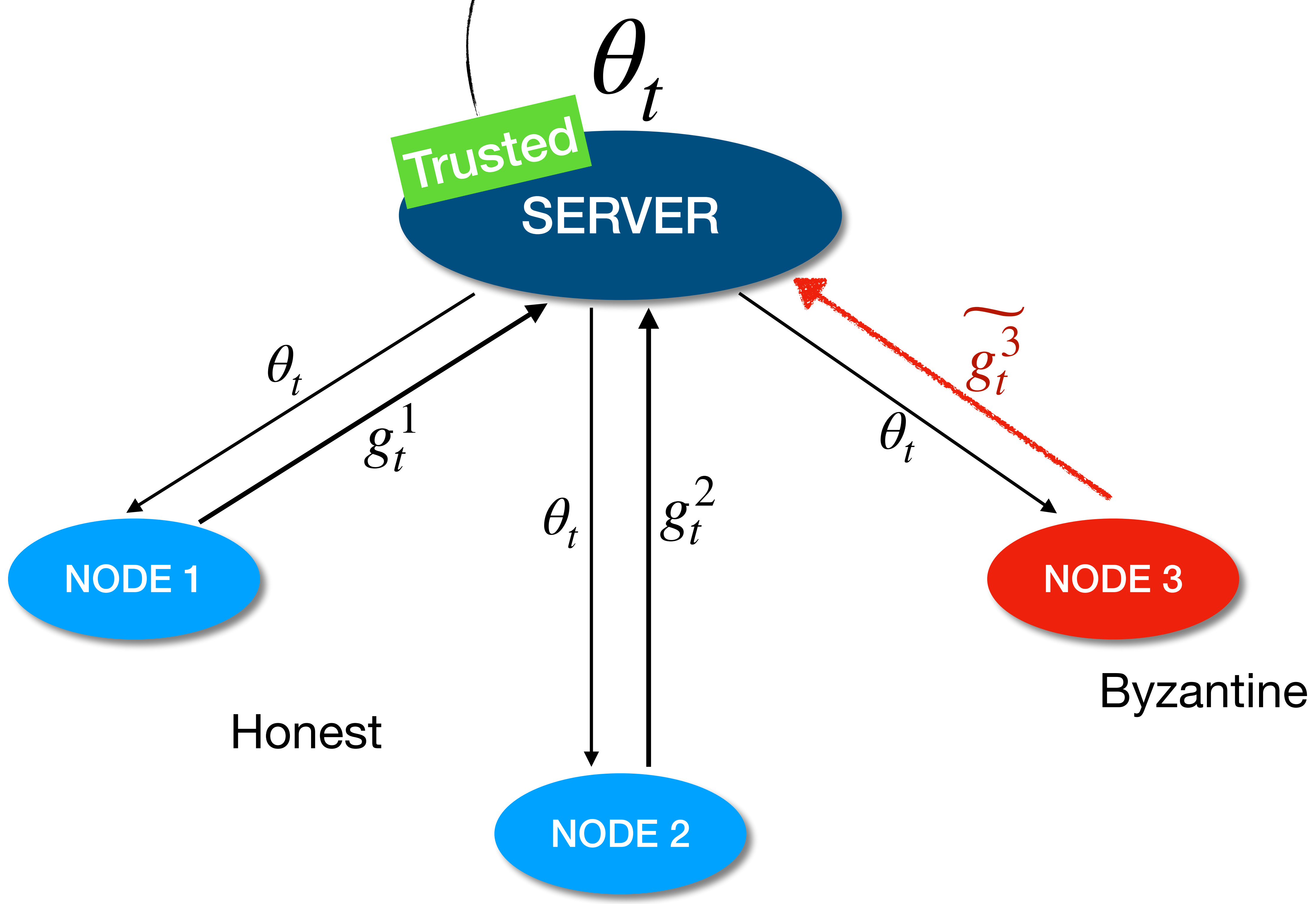
Feasible as nodes are assumed to sample data from the same distribution*

(f, ϵ)-Resilience

An algorithm that allows the honest nodes to compute $\hat{\theta}$ such that

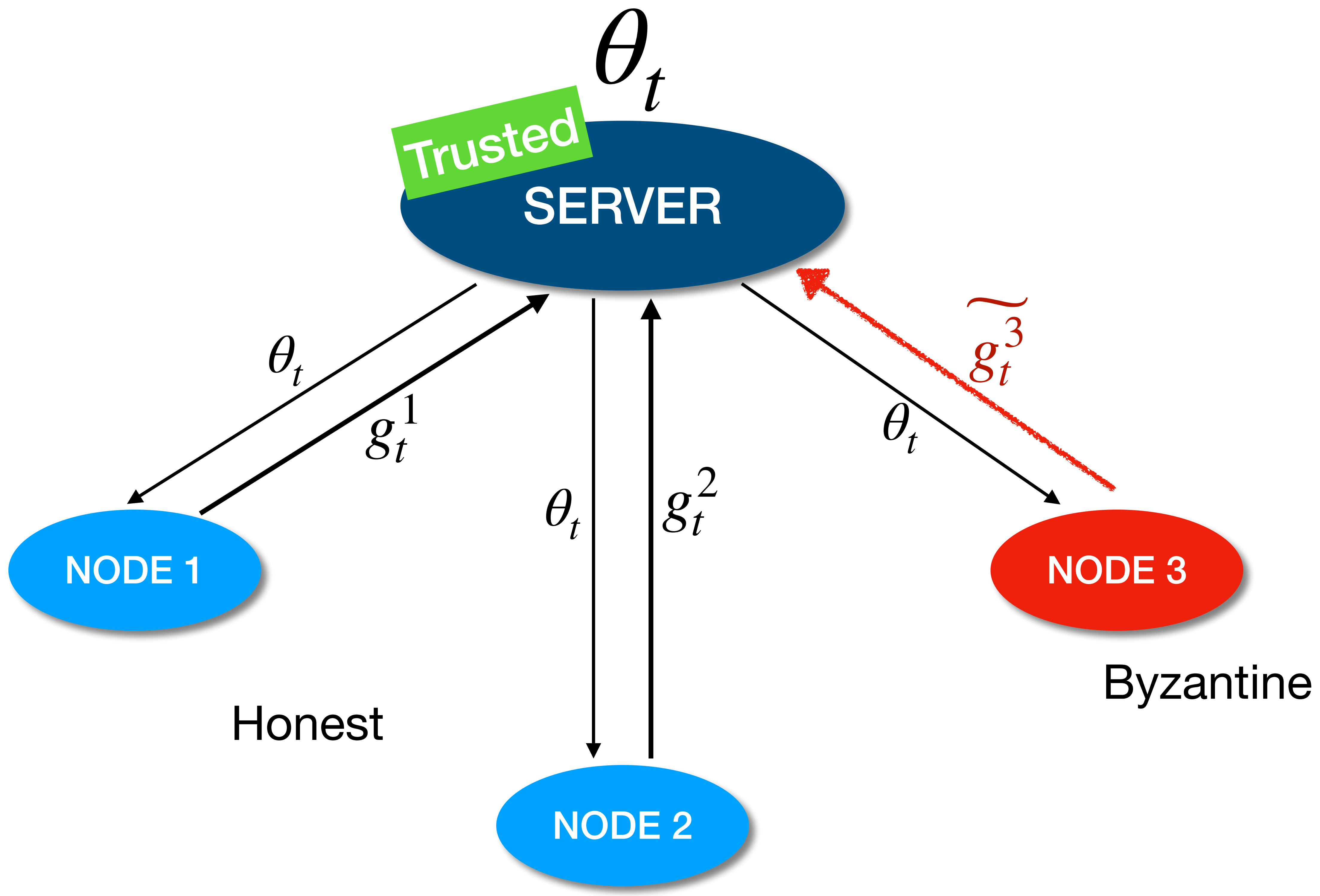
$$\|\nabla Q\left(\hat{\theta}\right)\|^2 \leq \epsilon$$

obtain insights that can be applied to non-trusted server settings



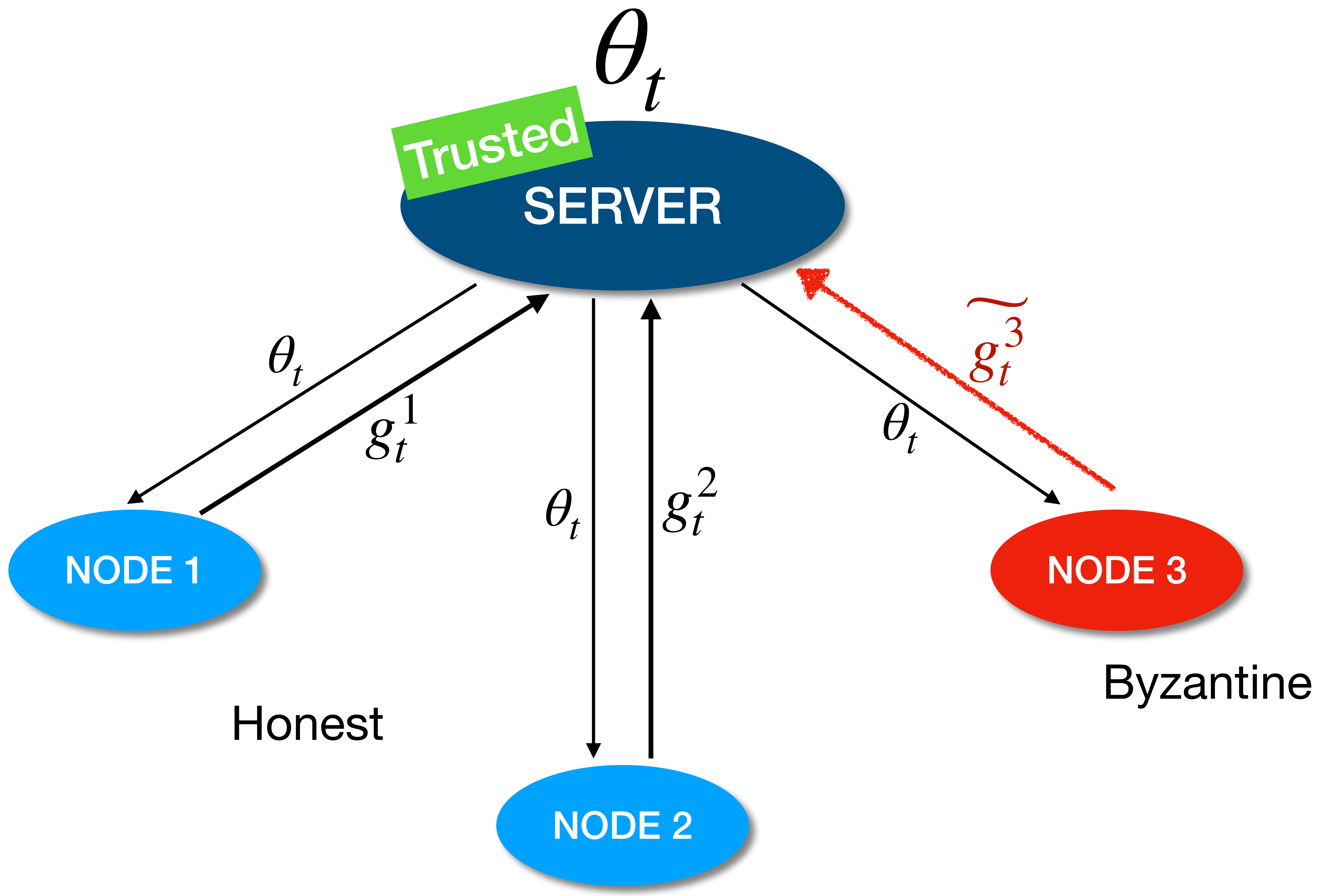
* In general, when data distributions differ, we require $2f$-*redundancy* in data (Gupta and Vaidya, 2021)

Making D-SGD Byzantine Resilient



Making D-SGD Byzantine Resilient

Classical D-SGD - Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \ \widehat{g}_t$

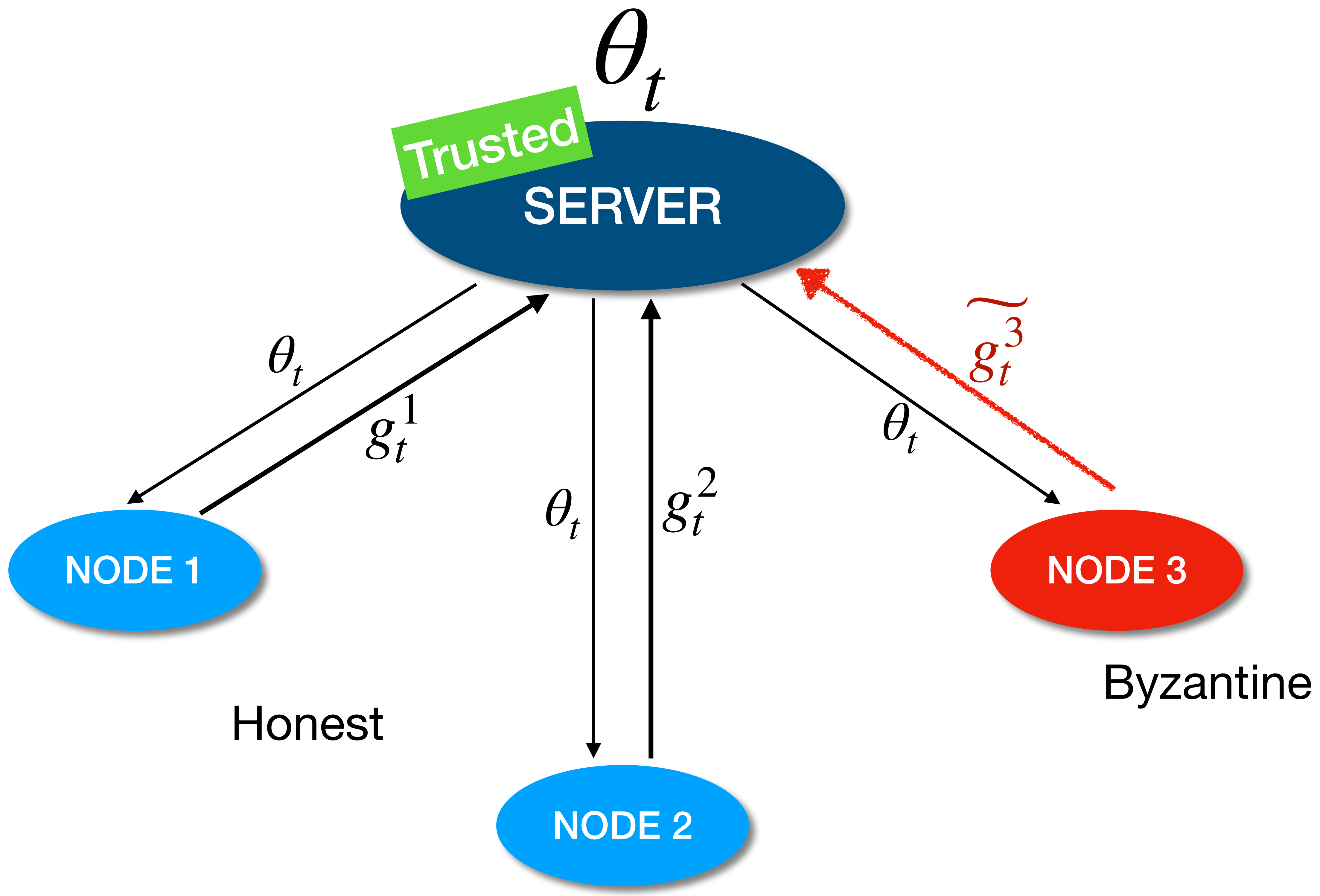


Making D-SGD Byzantine Resilient

Classical D-SGD - Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \; \widehat{g}_t$

We **cannot use**, for updates, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

(or any other linear function)



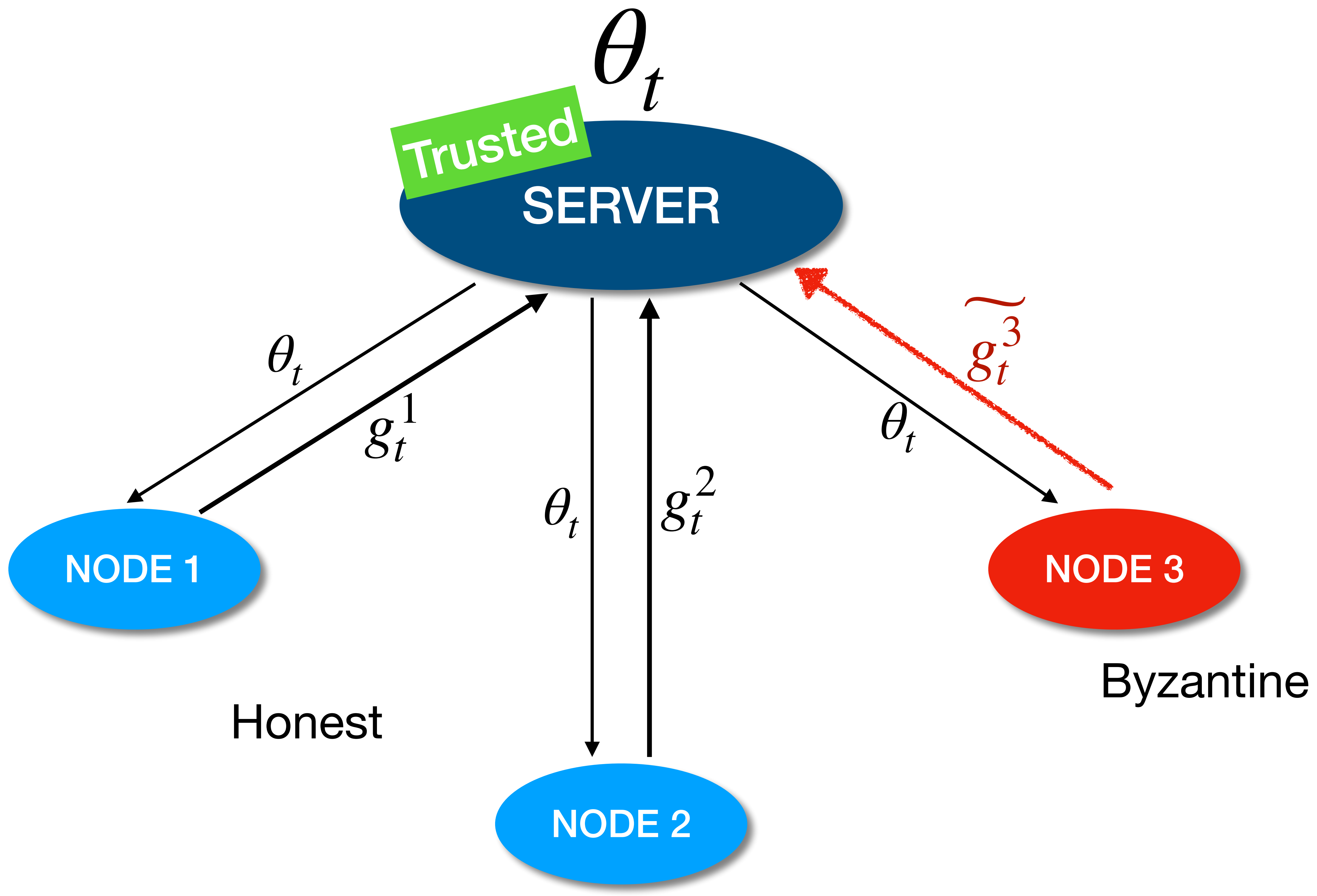
Making D-SGD Byzantine Resilient

Classical D-SGD - Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \; \widehat{g}_t$

We **cannot use**, for updates, $\widehat{g}_t = \dfrac{1}{n} \sum_i g_t^i$

(or any other linear function)

Use a **"robust aggregation"** $R_t = F\left(g_t^1, \ldots, g_t^n\right)^*$



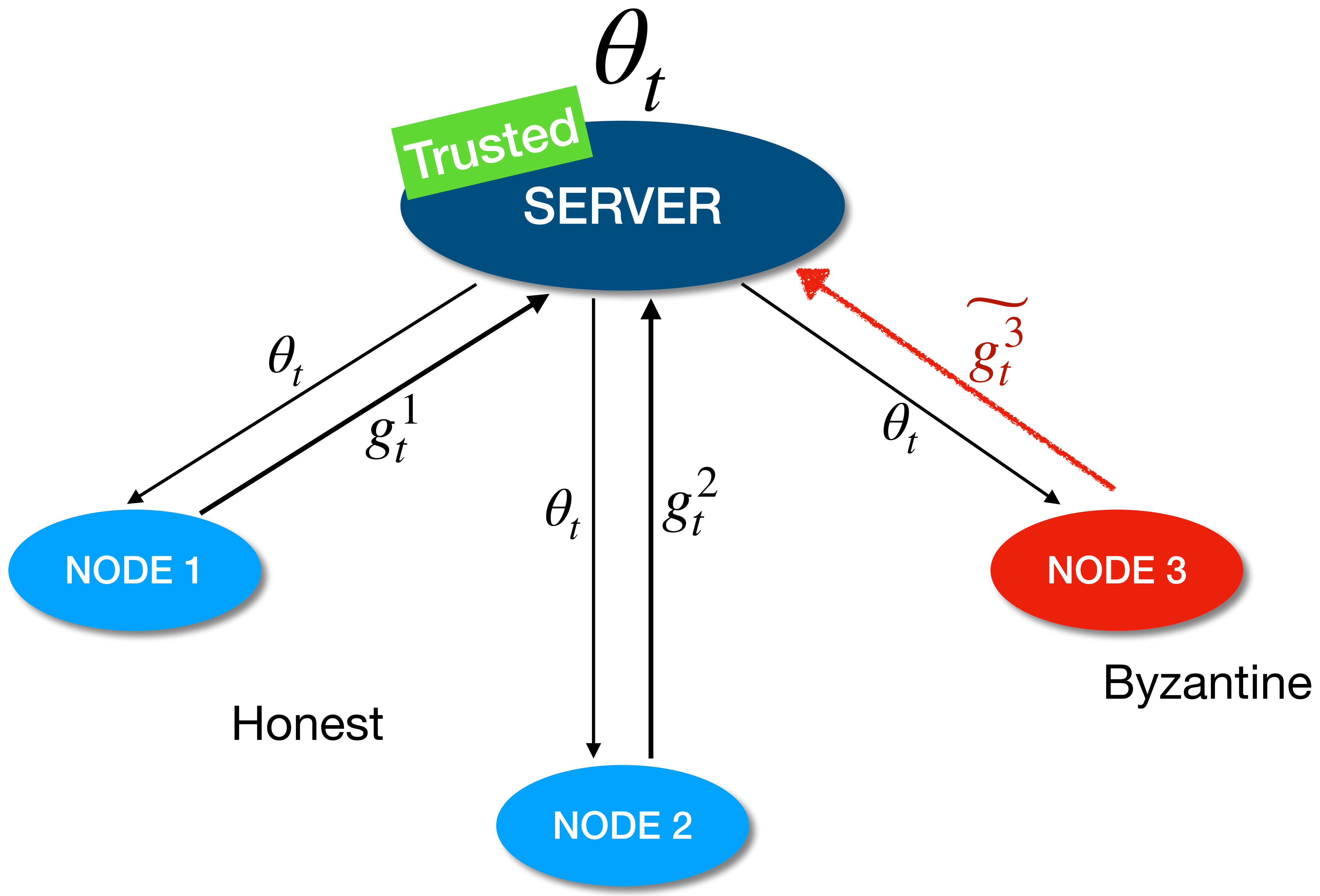
Making D-SGD Byzantine Resilient

Classical D-SGD - Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \; \widehat{g}_t$

We **cannot use**, for updates, $\widehat{g}_t = \frac{1}{n} \sum_i g_t^i$

(or any other linear function)

Use a **"robust aggregation"** $R_t = F\left(g_t^1, \ldots, g_t^n\right)^*$



* As the identity of Byzantine nodes is unknown we also denote their gradients by g

Making D-SGD Byzantine Resilient

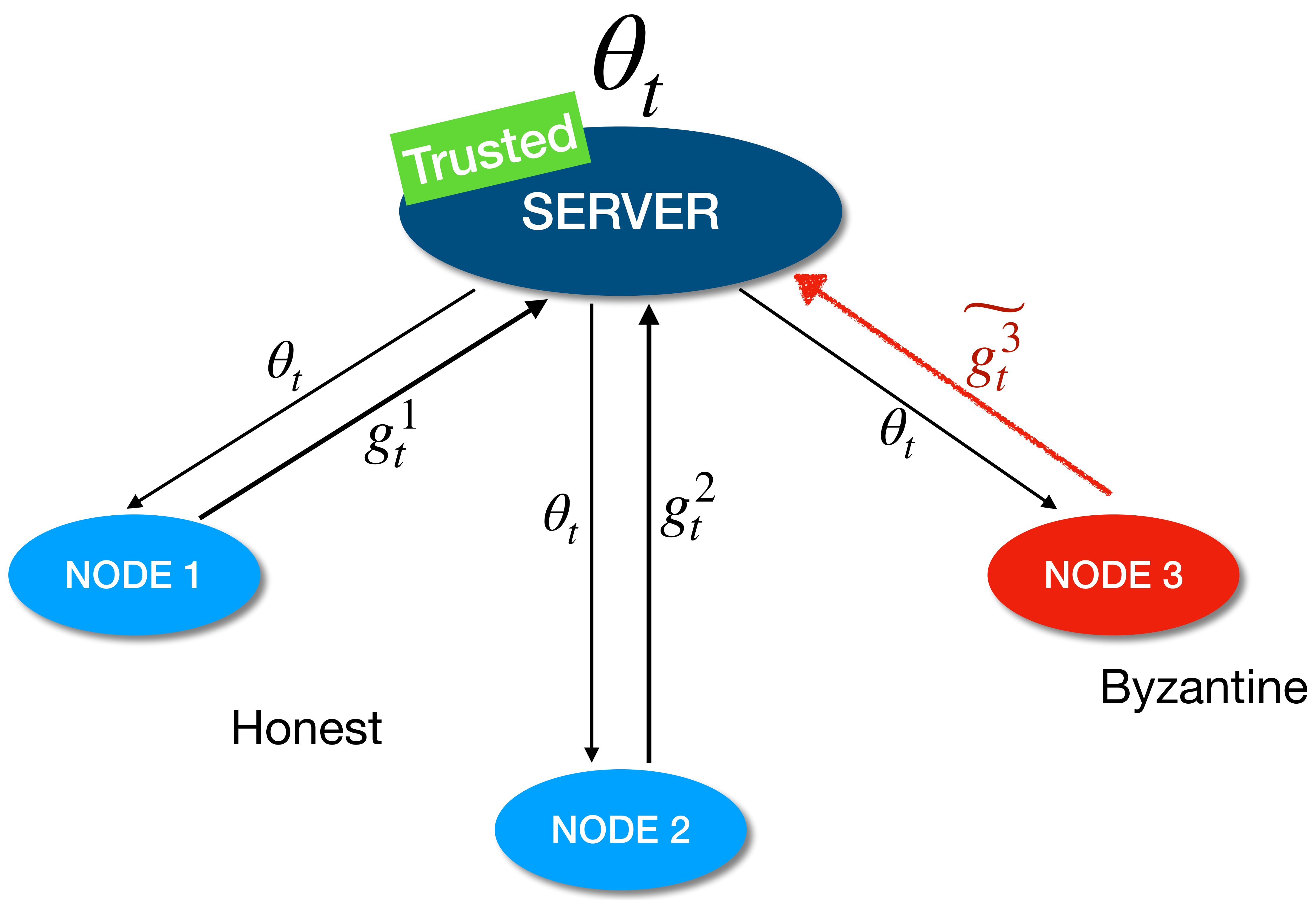
Classical D-SGD - Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \; \widehat{g}_t$

We **cannot use**, for updates, $\widehat{g}_t = \frac{1}{n}\sum_i g_t^i$

(or any other linear function)

Use a **"robust aggregation"** $R_t = F\left(g_t^1, \ldots, g_t^n\right)^*$

Server updates $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \; R_t$



* As the identity of Byzantine nodes is unknown we also denote their gradients by g

What is a Robust Aggregation ?

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):

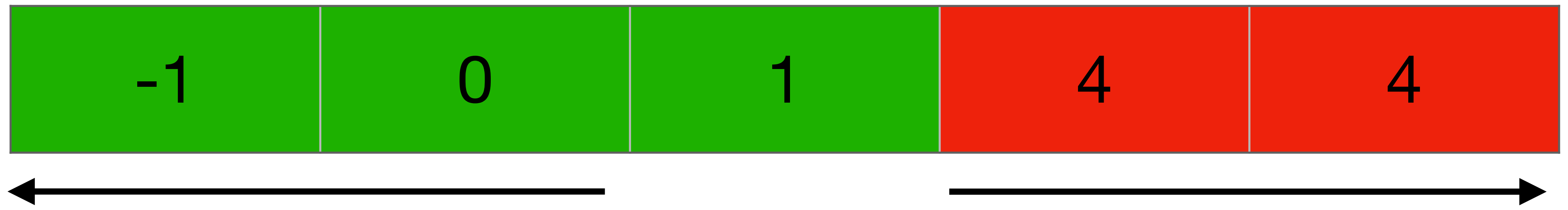
-1	0	1	4	4

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):

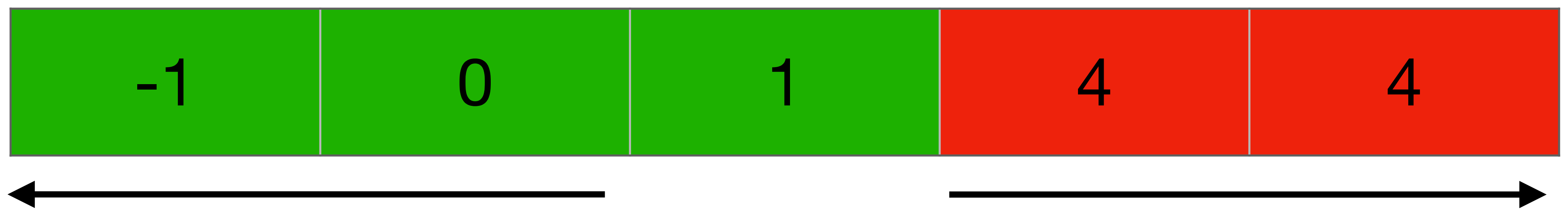


What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):



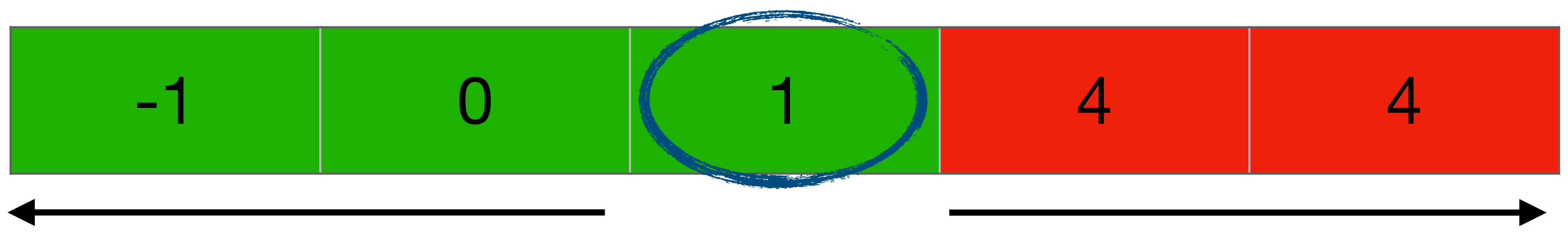
Eliminate f extreme values
Average the remaining

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):



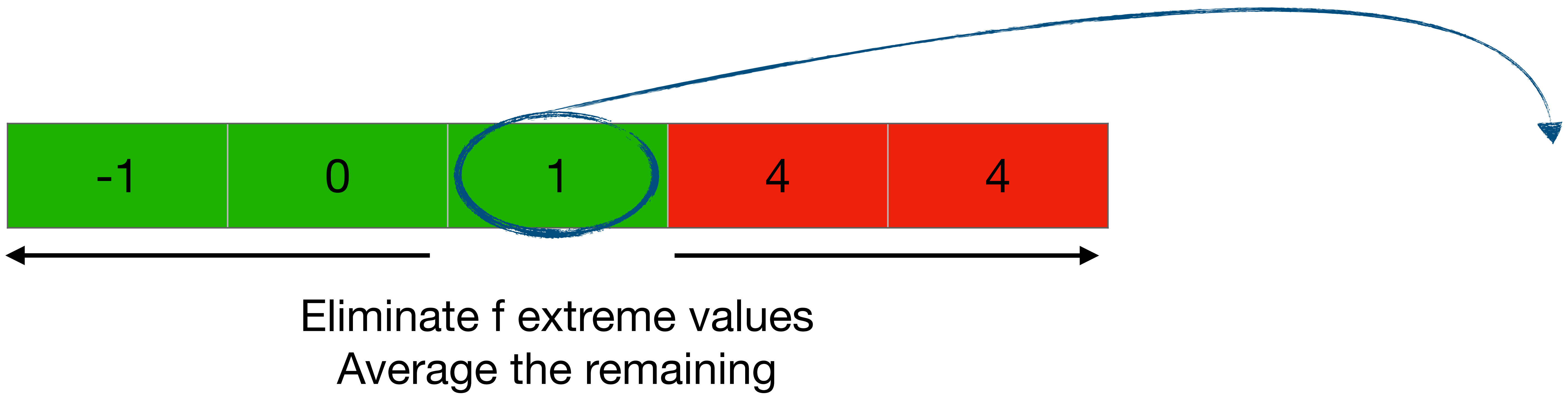
Eliminate f extreme values
Average the remaining

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):

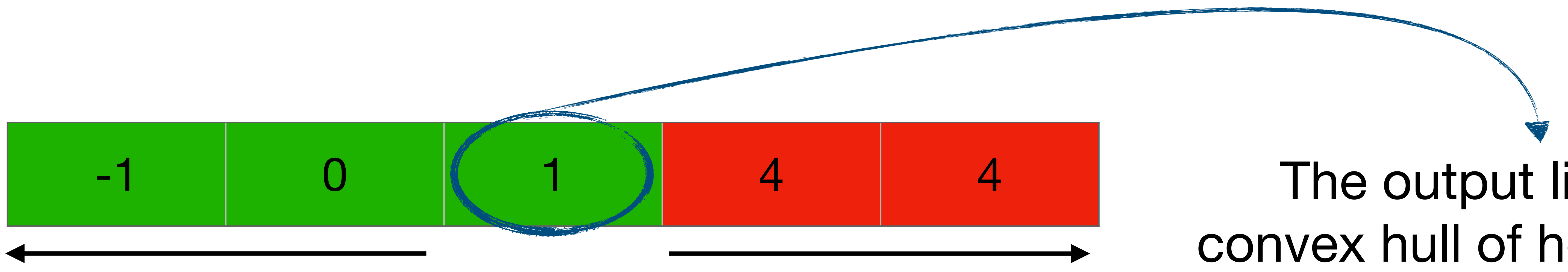


What is a Robust Aggregation ?

Reproduces the impact of Byzantine inputs

Examples: $n = 5$ and $f = 2$

Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

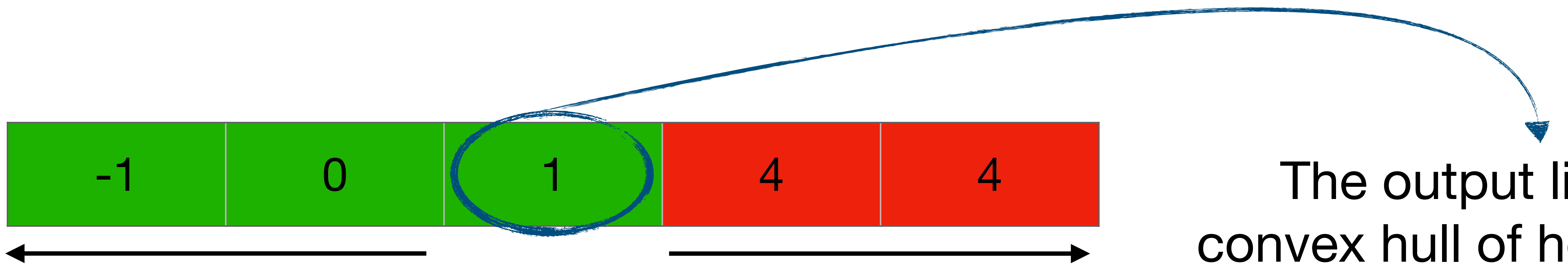
Eliminate f extreme values
Average the remaining

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

Trimmed Mean (TM):

-1	0	1	4	4

Eliminate f extreme values
Average the remaining

The output lies in the convex hull of honest inputs

Minimum Diameter Averaging* (MDA):

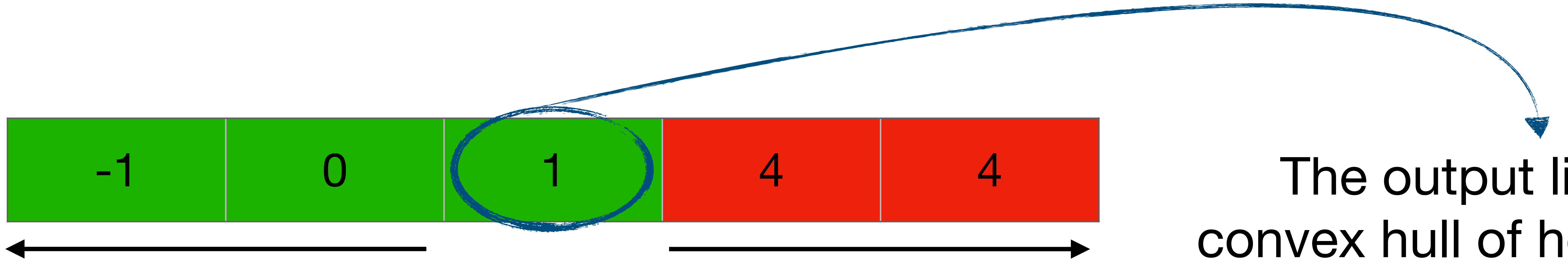
-1	0	1	4	4

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

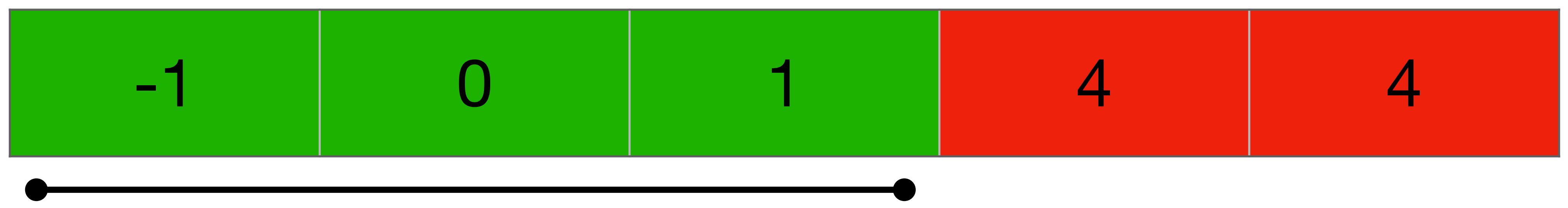
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):

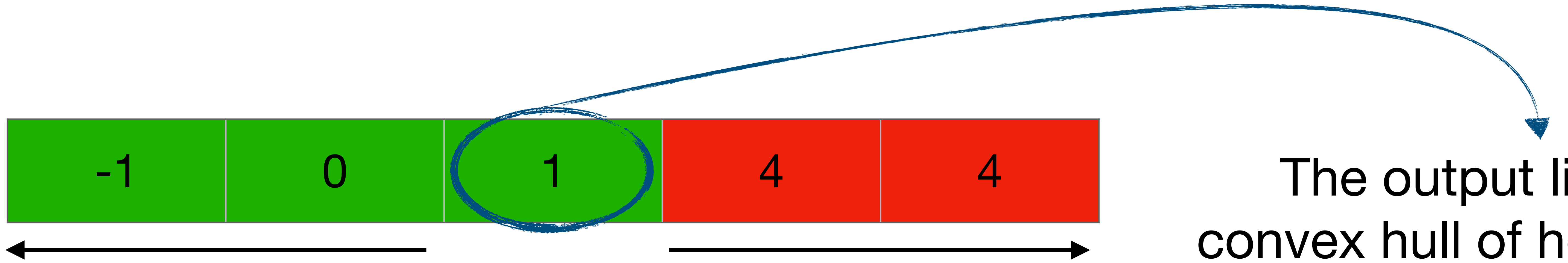


What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

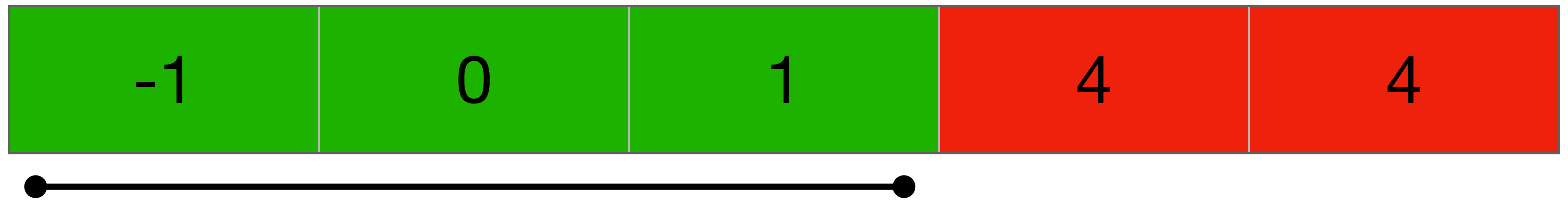
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):



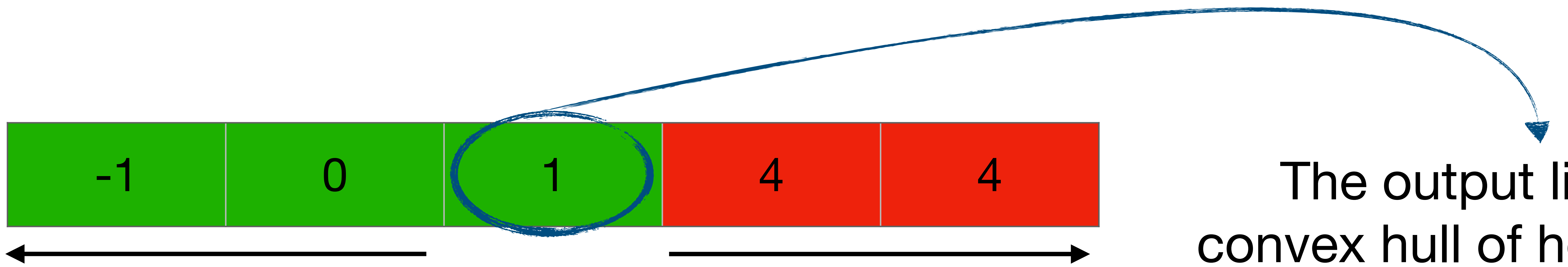
Average n-f values
with smallest diameter

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

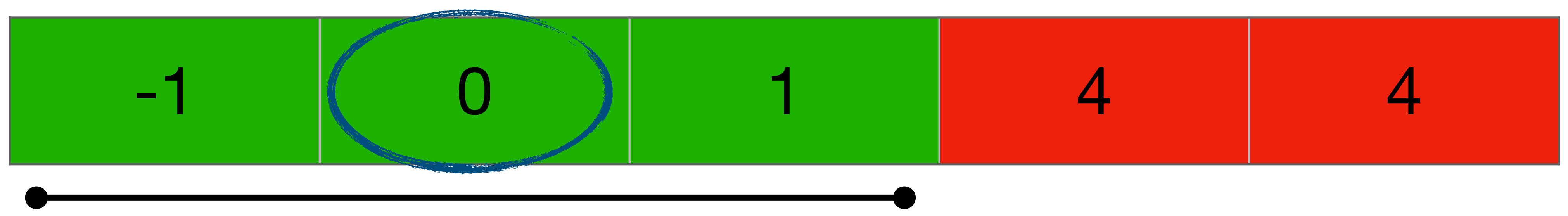
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):



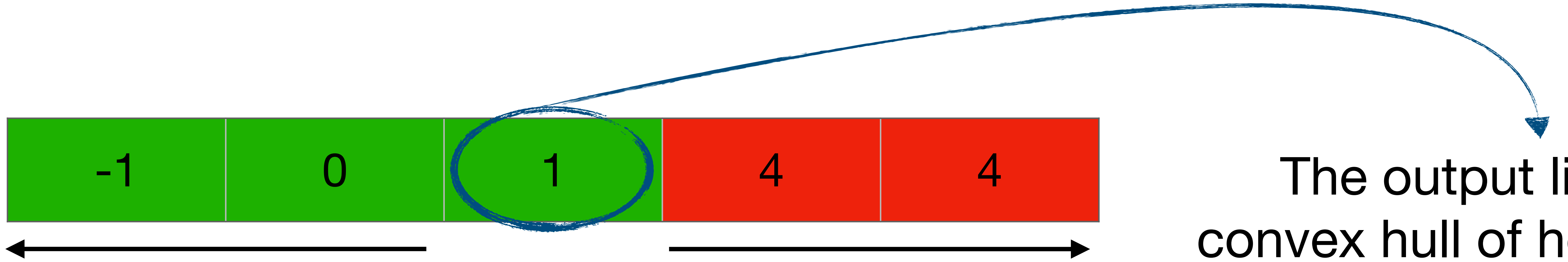
Average n-f values
with smallest diameter

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

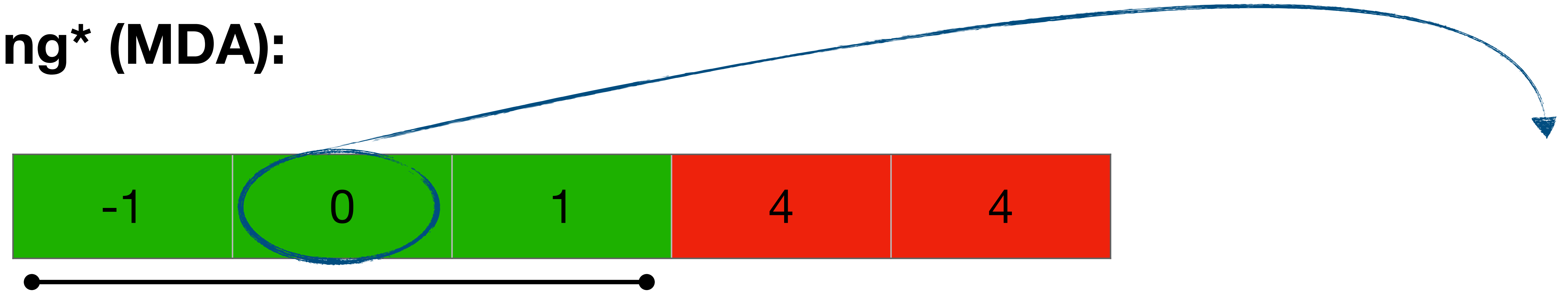
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):



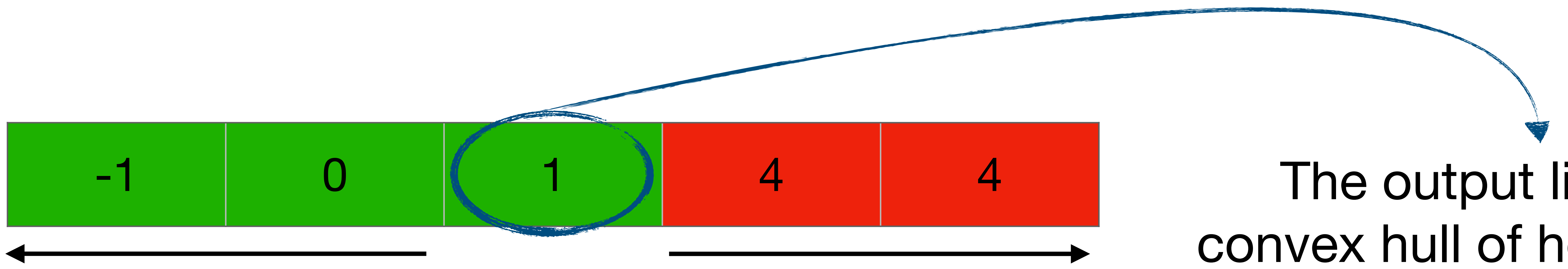
Average n-f values
with smallest diameter

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

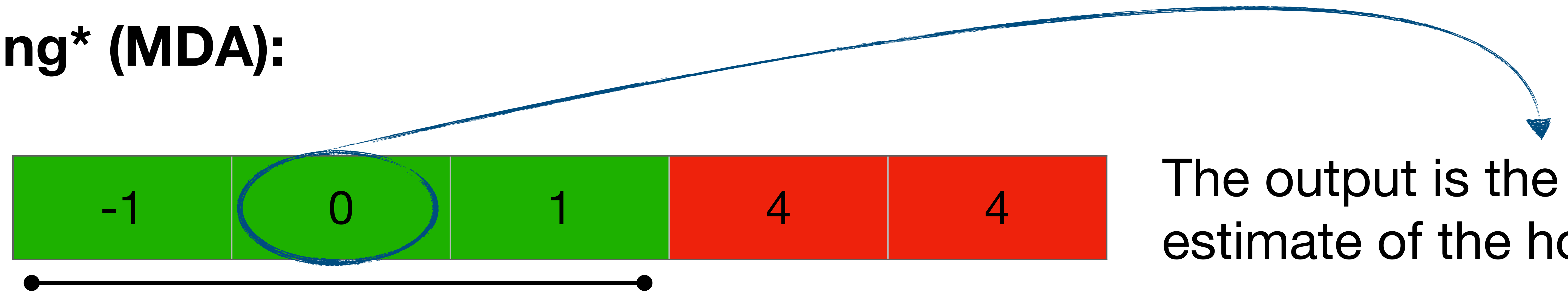
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):



The output is the best possible estimate of the honest average

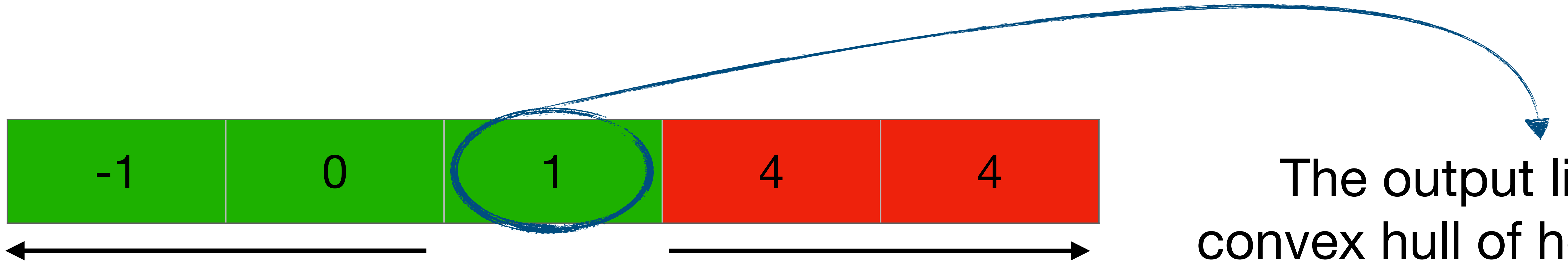
Average n-f values
with smallest diameter

What is a Robust Aggregation ?

Reduces the impact of Byzantine inputs

Examples: n = 5 and f = 2

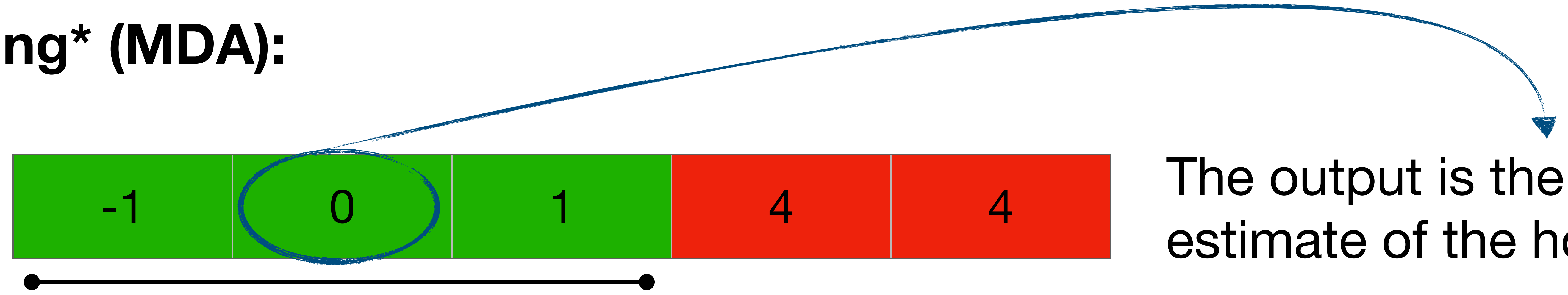
Trimmed Mean (TM):



The output lies in the convex hull of honest inputs

Eliminate f extreme values
Average the remaining

Minimum Diameter Averaging* (MDA):



The output is the best possible estimate of the honest average

Average n-f values
with smallest diameter

* Very costly in high-dimension

Does Robust Aggregation Suffice

Does Robust Aggregation Suffice

It depends

Does Robust Aggregation Suffice

It depends

Aggregation Rule/Scheme	Additional Assumption
Trimmed Mean *(Yin et al., 2018)*	Sub-exponential stochasticity
Geometric Median *(Chen et al., 2017)*	Sub-exponential stochasticity
Krum/multi-Krum *(Blanchard et al., 2017)*	Vanishing variance
Monitoring temporal averages of gradients *(Alistarh et al., 2018)*	(a.s.) Absolutely boundedness

Does Robust Aggregation Suffice

It depends

Aggregation Rule/Scheme	Additional Assumption
Trimmed Mean *(Yin et al., 2018)*	Sub-exponential stochasticity
Geometric Median *(Chen et al., 2017)*	Sub-exponential stochasticity
Krum/multi-Krum *(Blanchard et al., 2017)*	Vanishing variance
Monitoring temporal averages of gradients *(Alistarh et al., 2018)*	(a.s.) Absolutely boundedness

Most give resilience only against a small fraction of Byzantines << 1/2

Vulnerable due to Stringent Assumptions

Vulnerable due to Stringent Assumptions

* MNIST classification task

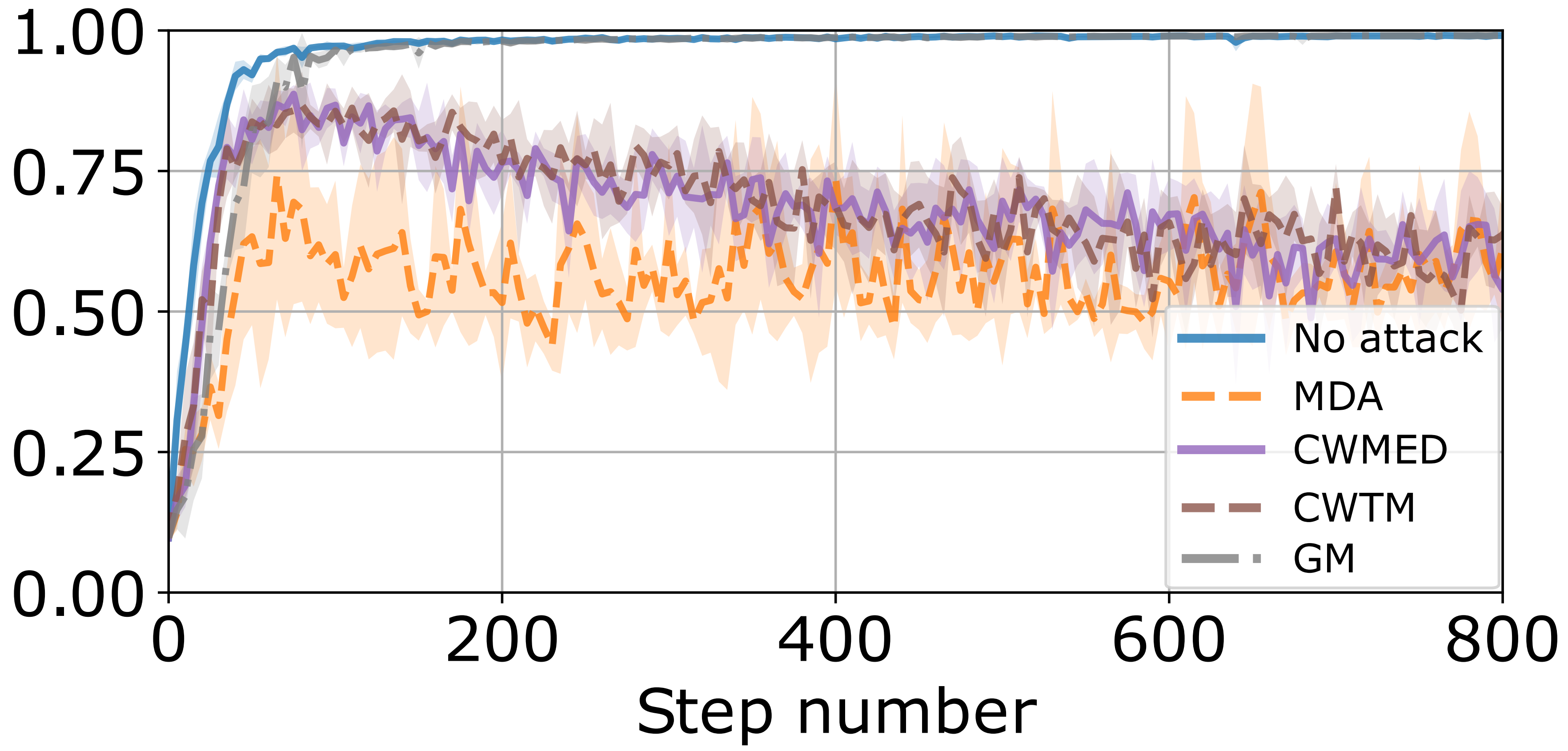
Vulnerable due to Stringent Assumptions

5 out of 25 nodes are Byzantine faulty

* MNIST classification task

Vulnerable due to Stringent Assumptions

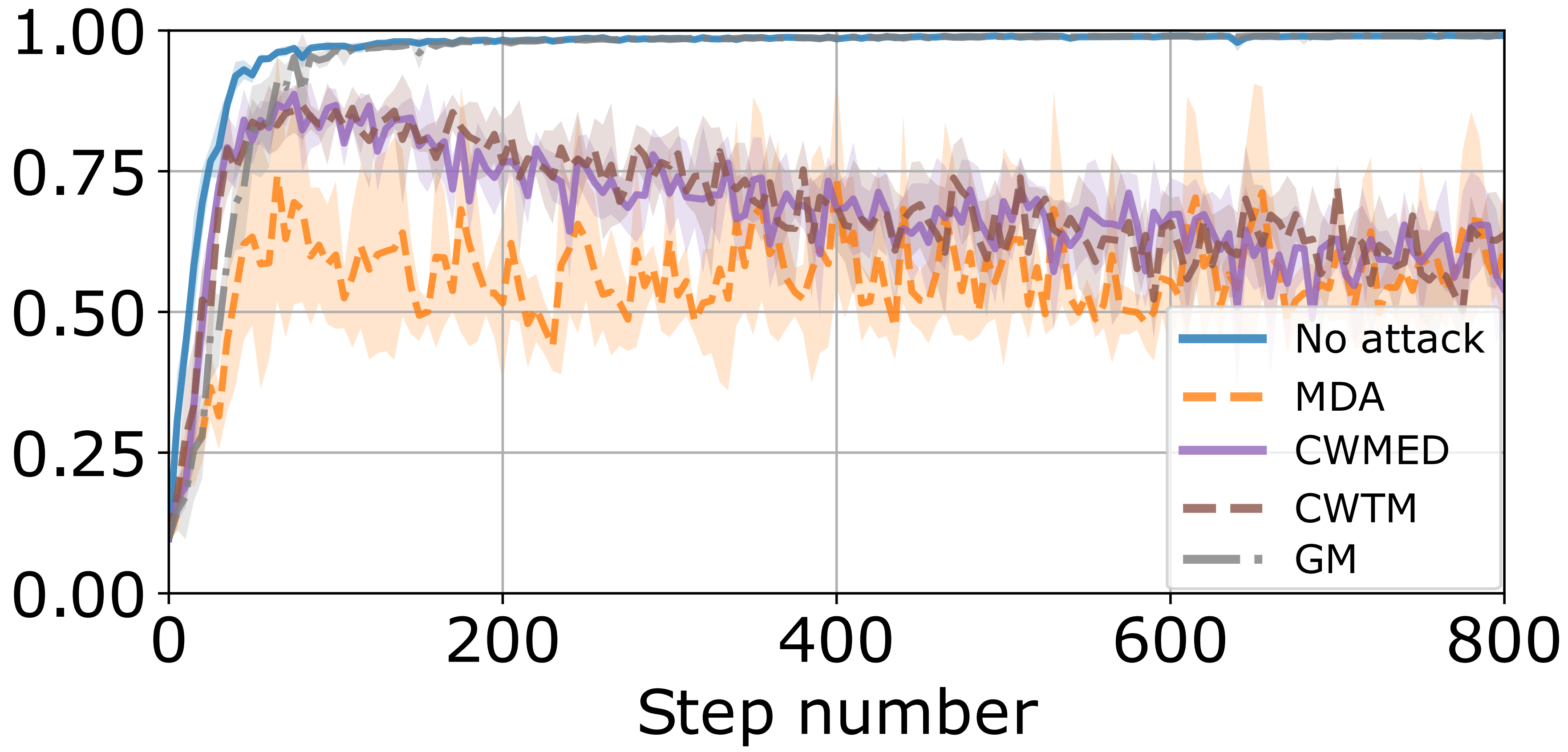
5 out of 25 nodes are Byzantine faulty



* MNIST classification task

Vulnerable due to Stringent Assumptions

5 out of 25 nodes are Byzantine faulty

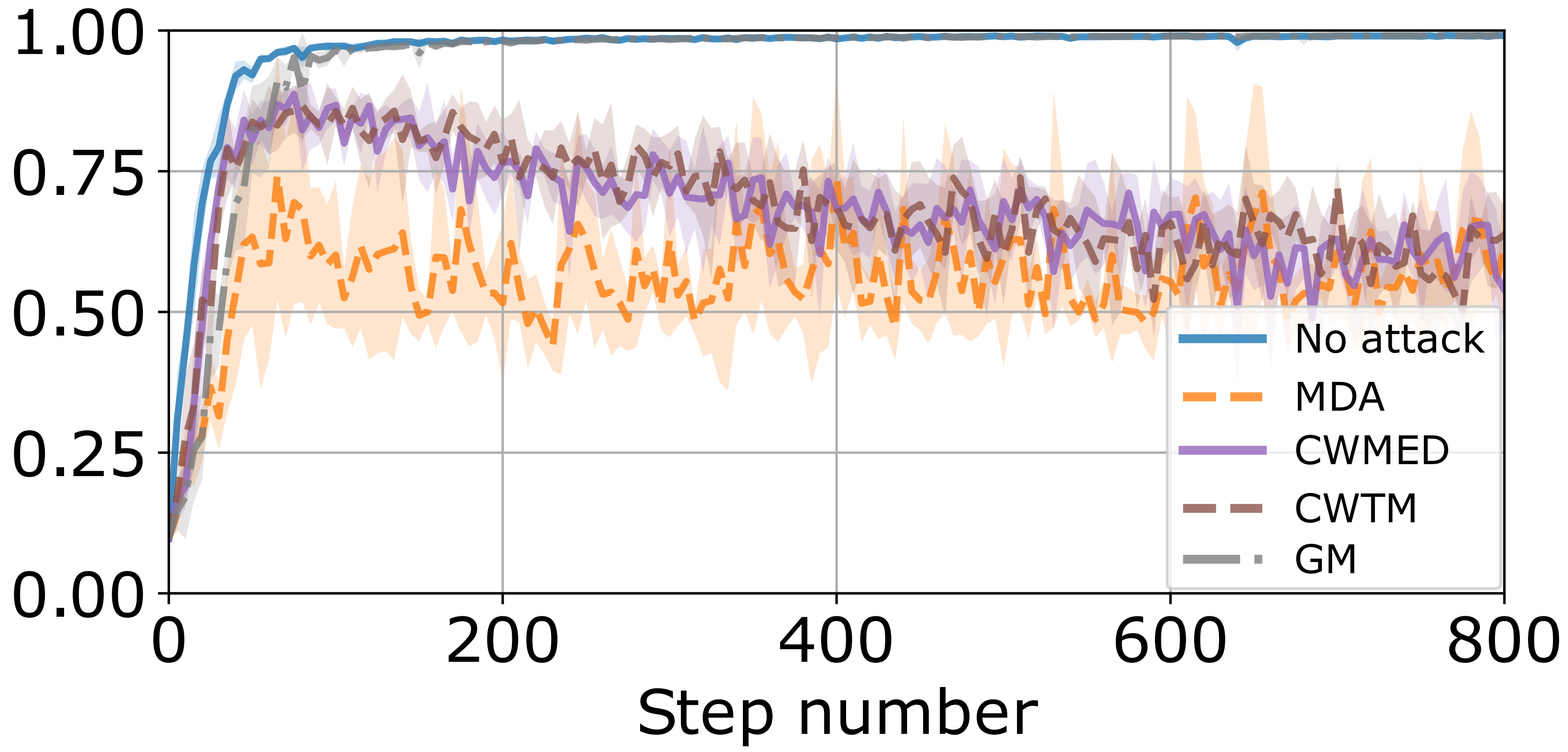


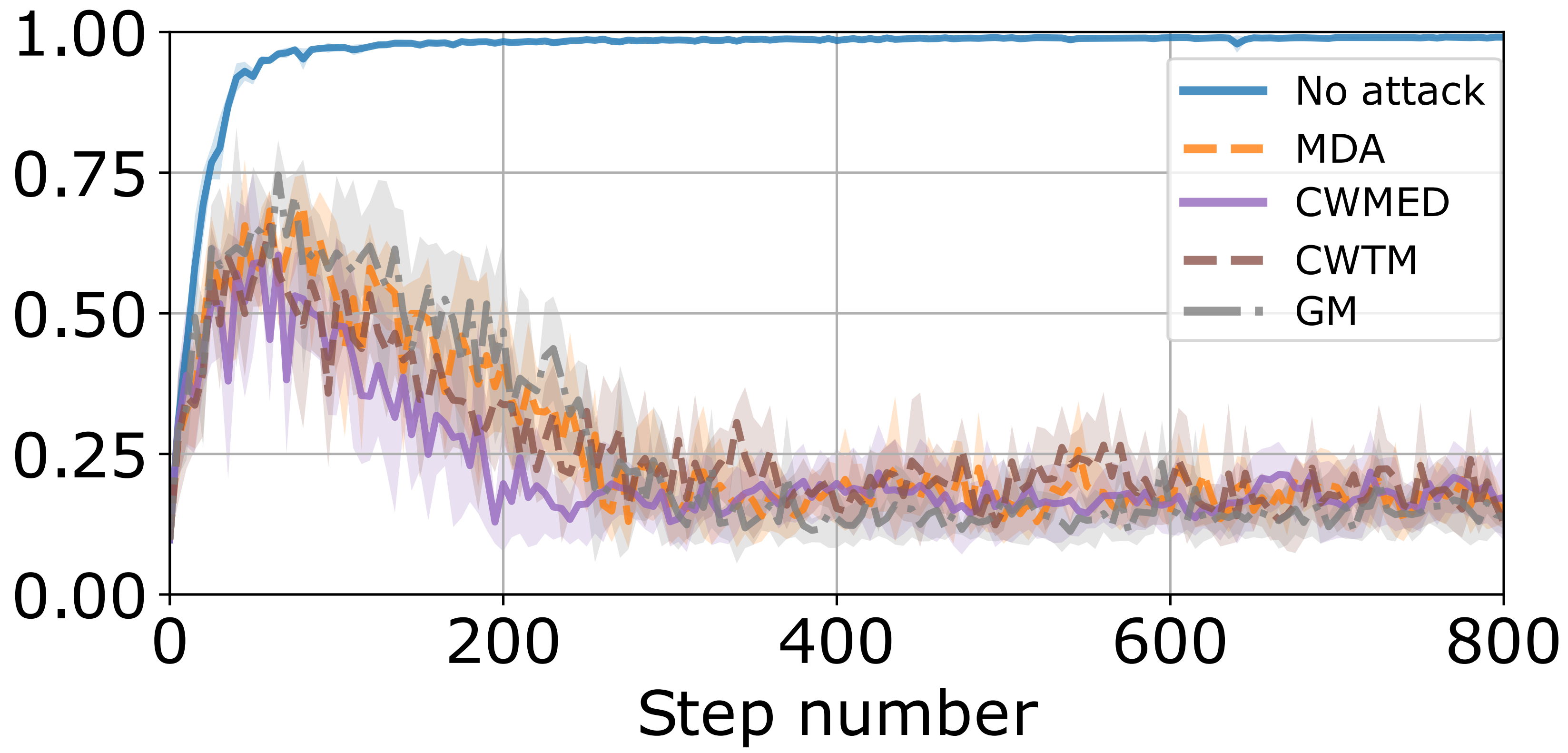
Label-flipping

* MNIST classification task

Vulnerable due to Stringent Assumptions

5 out of 25 nodes are Byzantine faulty



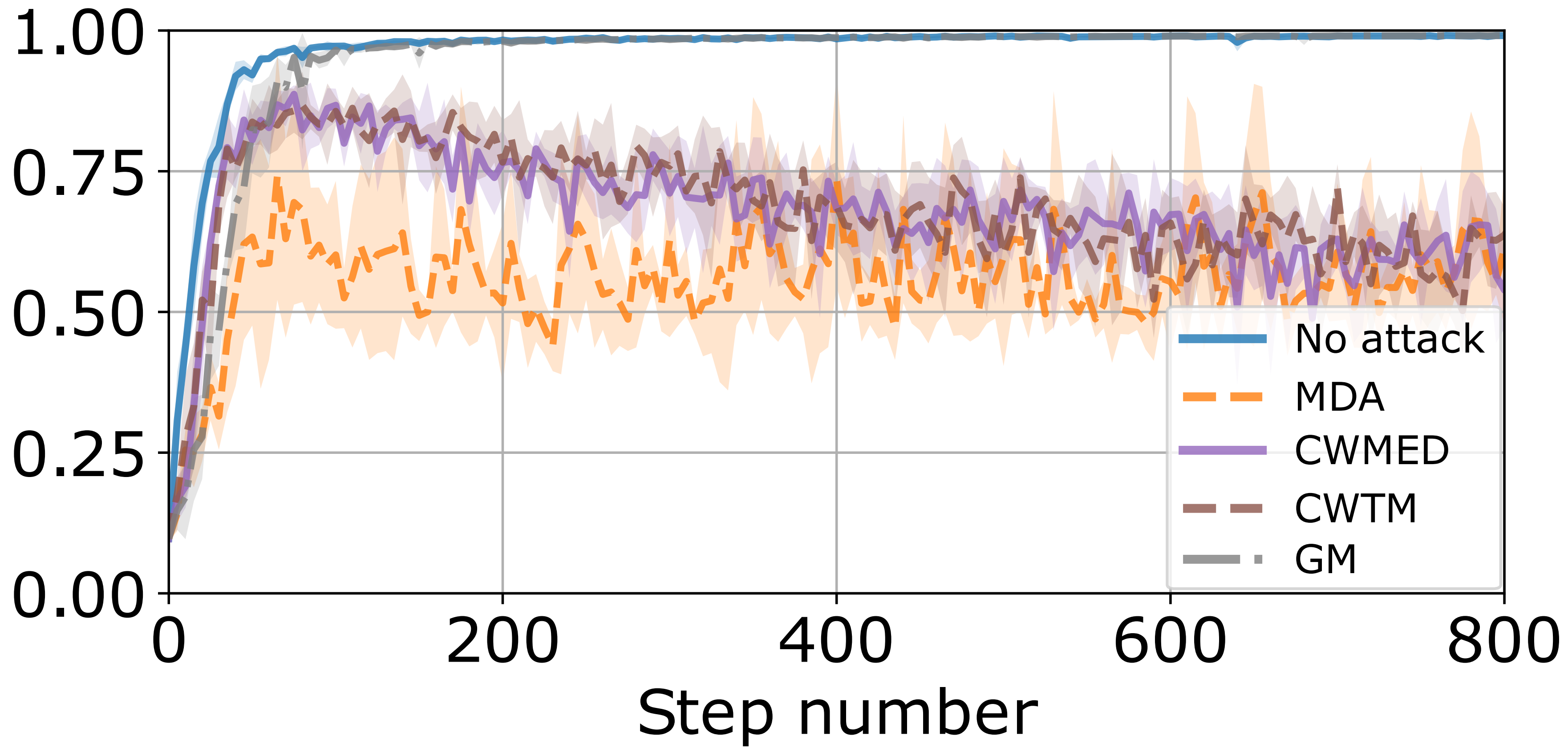


Label-flipping

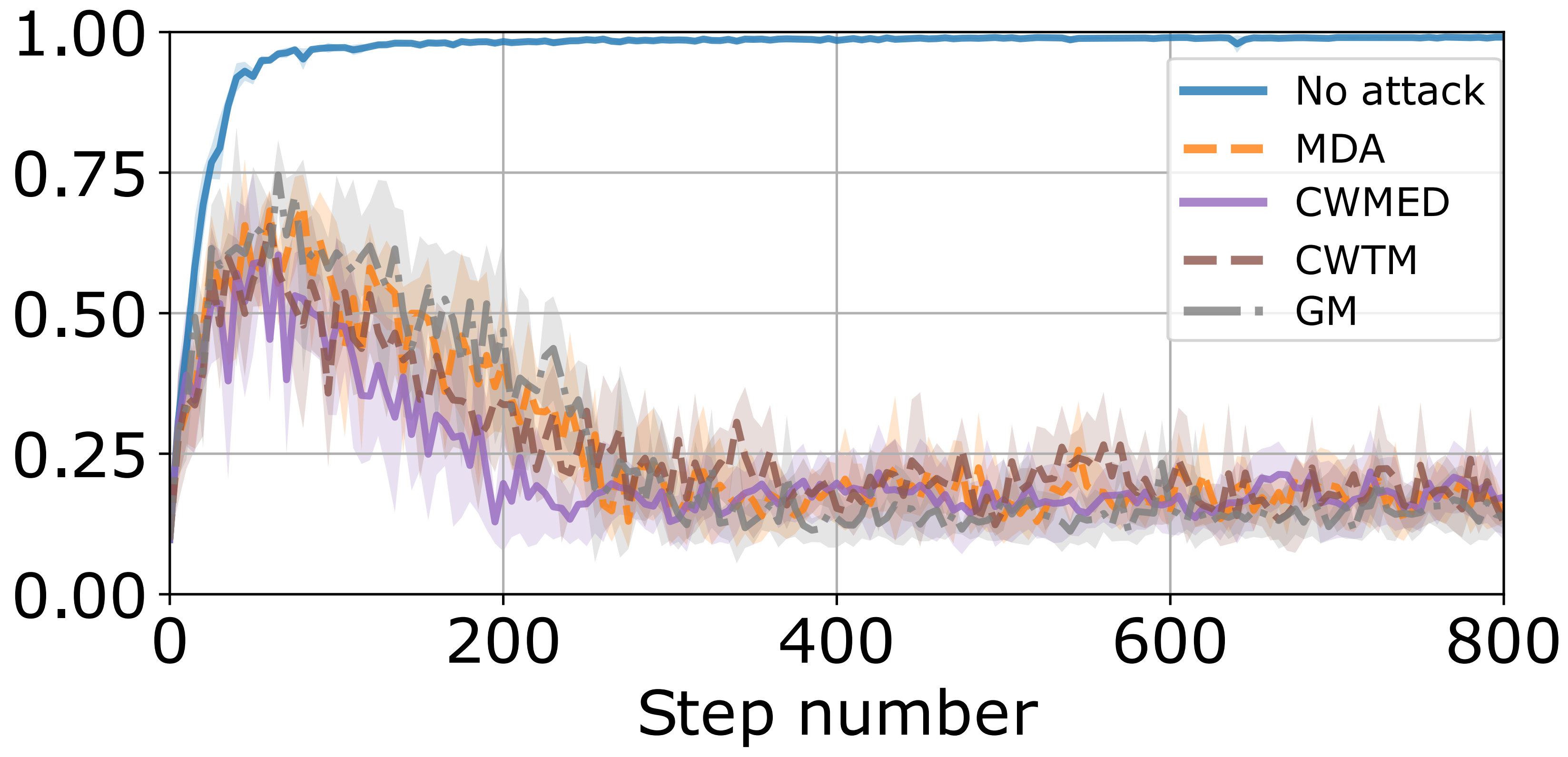
* MNIST classification task

Vulnerable due to Stringent Assumptions

5 out of 25 nodes are Byzantine faulty



Label-flipping

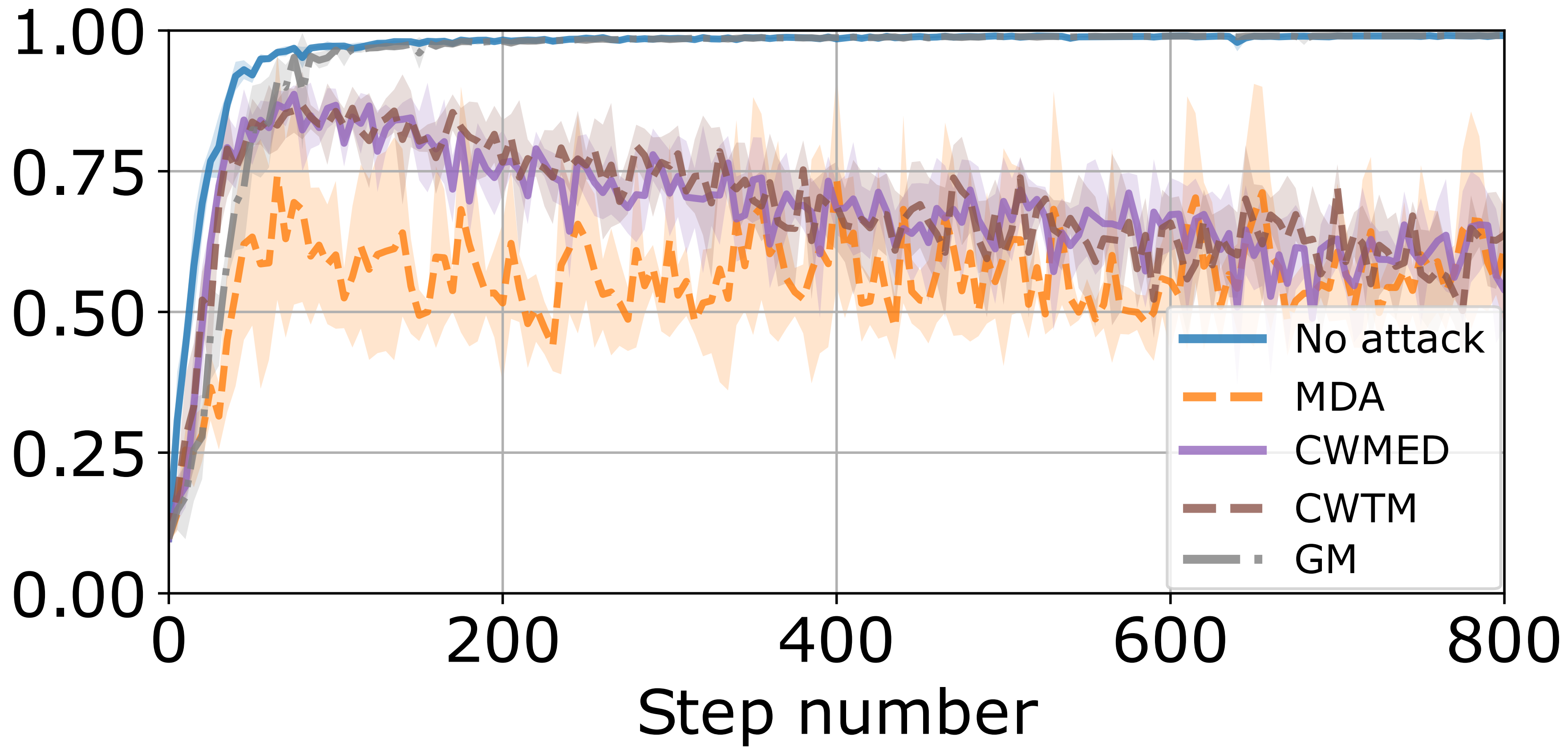


Little is enough *(Baruch et al., 2019)*

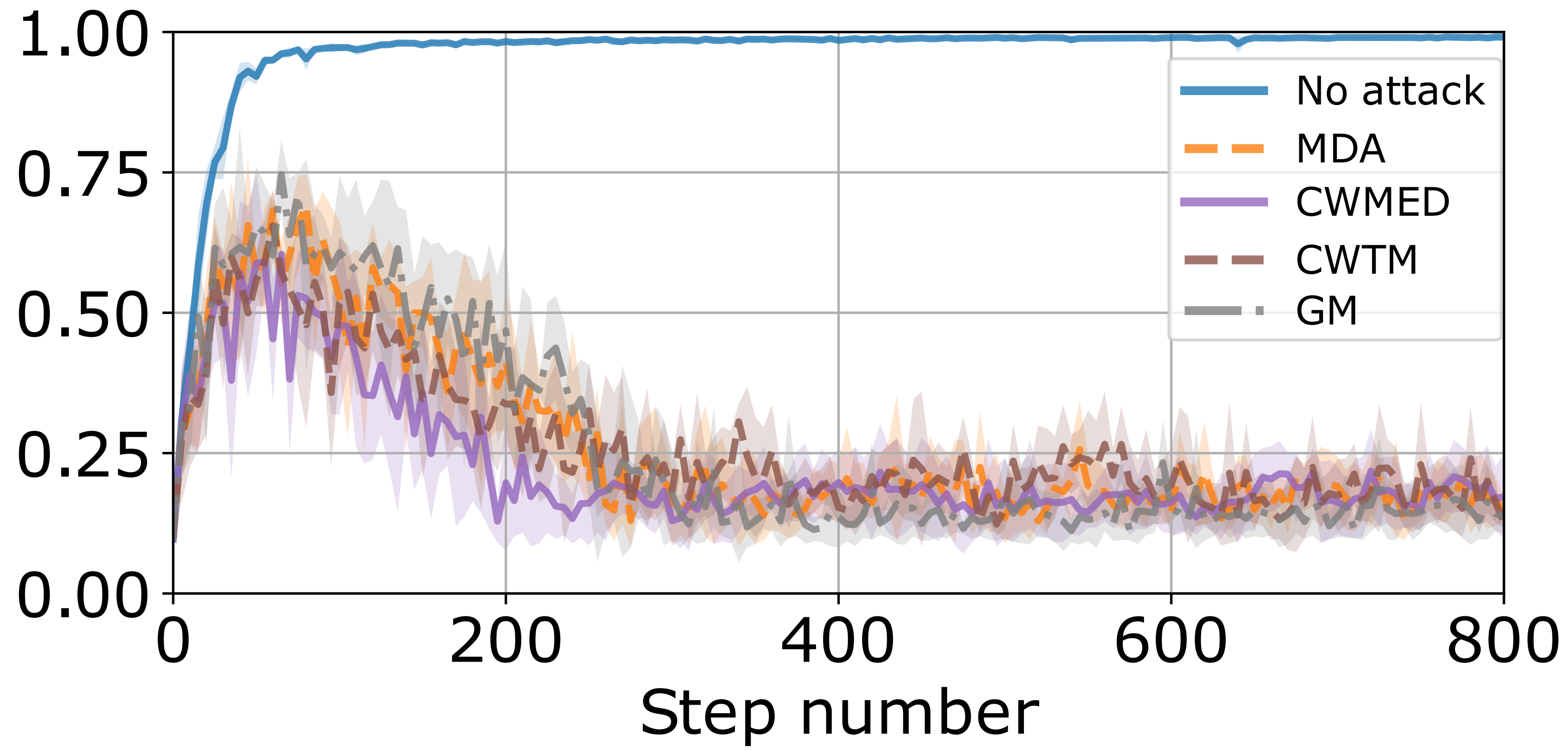
* MNIST classification task

Vulnerable due to Stringent Assumptions

5 out of 25 nodes are Byzantine faulty



Label-flipping



Little is enough *(Baruch et al., 2019)*

Memoryless robust aggregation need not be sufficient *(Karimireddy et al., 2021)*

* MNIST classification task

What is the Problem?

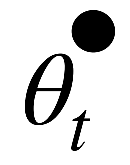
Non-trivial variance of stochastic gradients

What is the Problem?

Non-trivial variance of stochastic gradients

What is the Problem?

Non-trivial variance of stochastic gradients

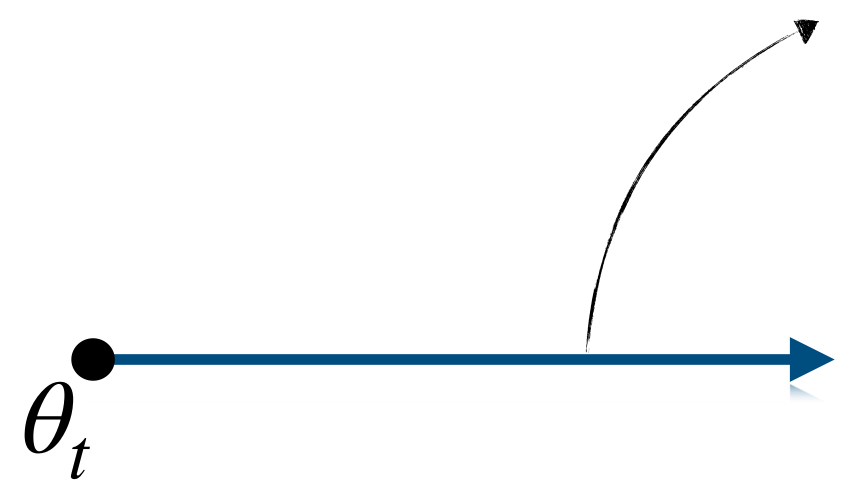


What is the Problem?

Non-trivial variance of stochastic gradients

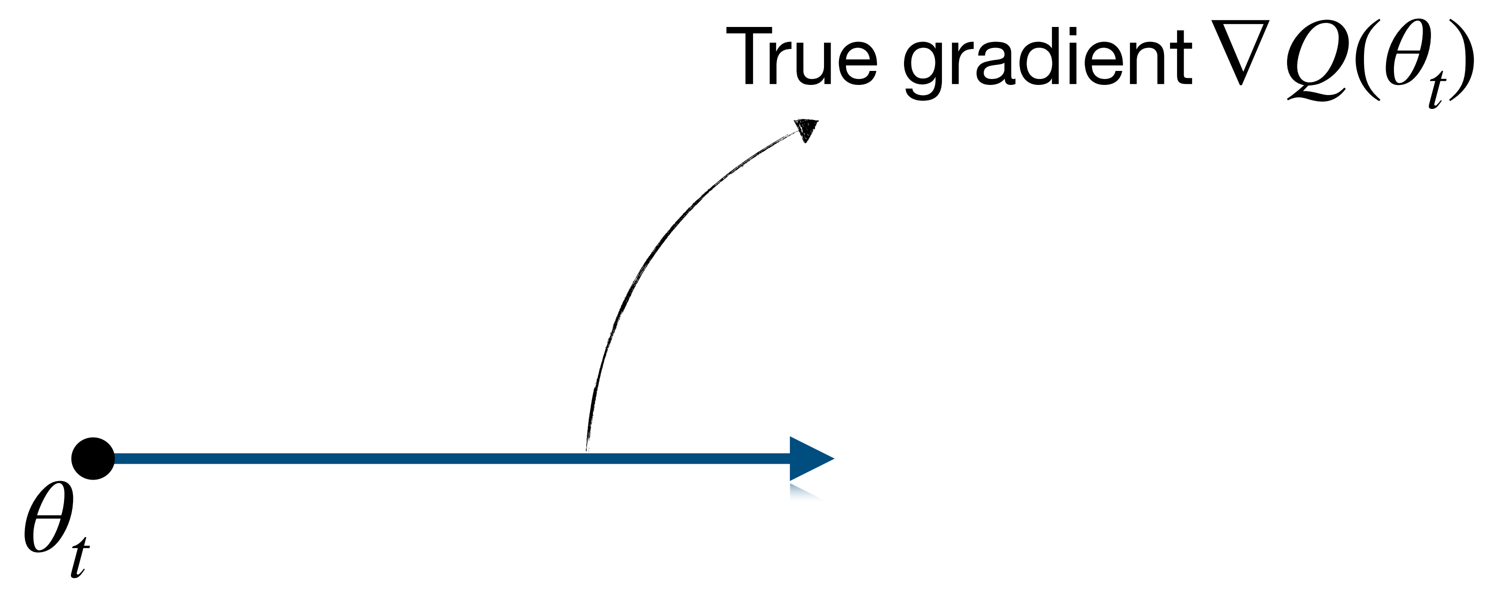
What is the Problem?

Non-trivial variance of stochastic gradients



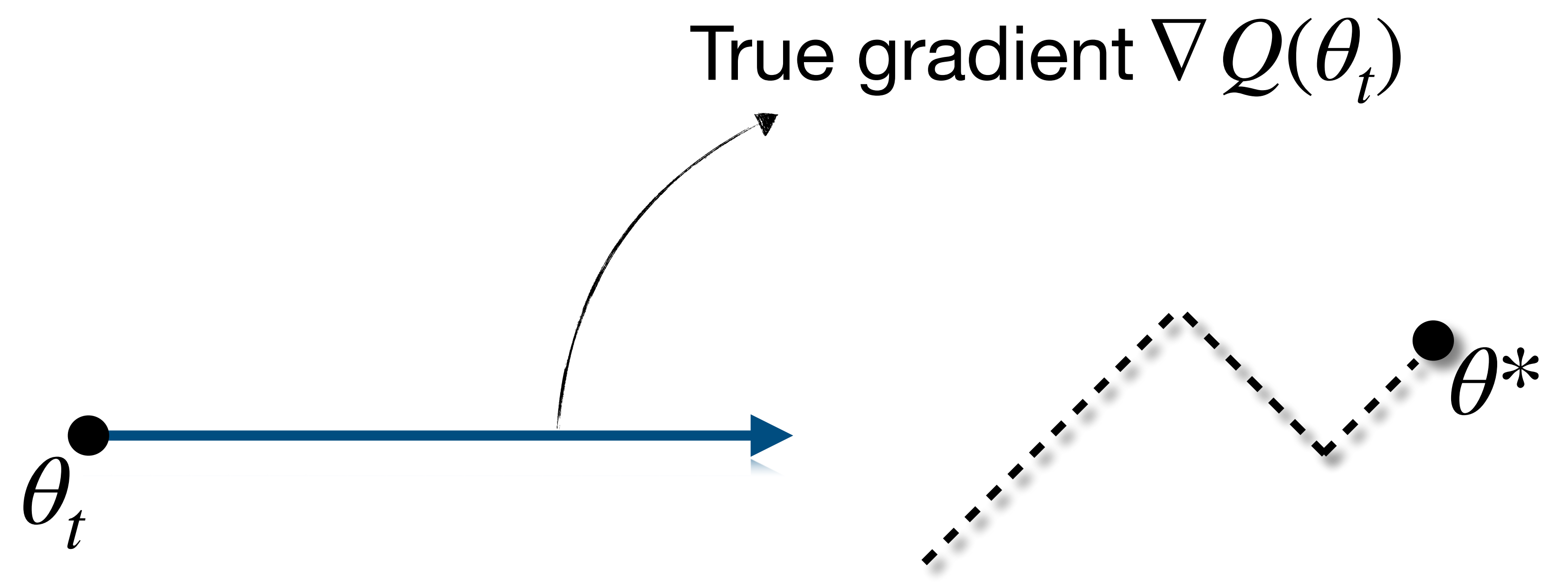
What is the Problem?

Non-trivial variance of stochastic gradients



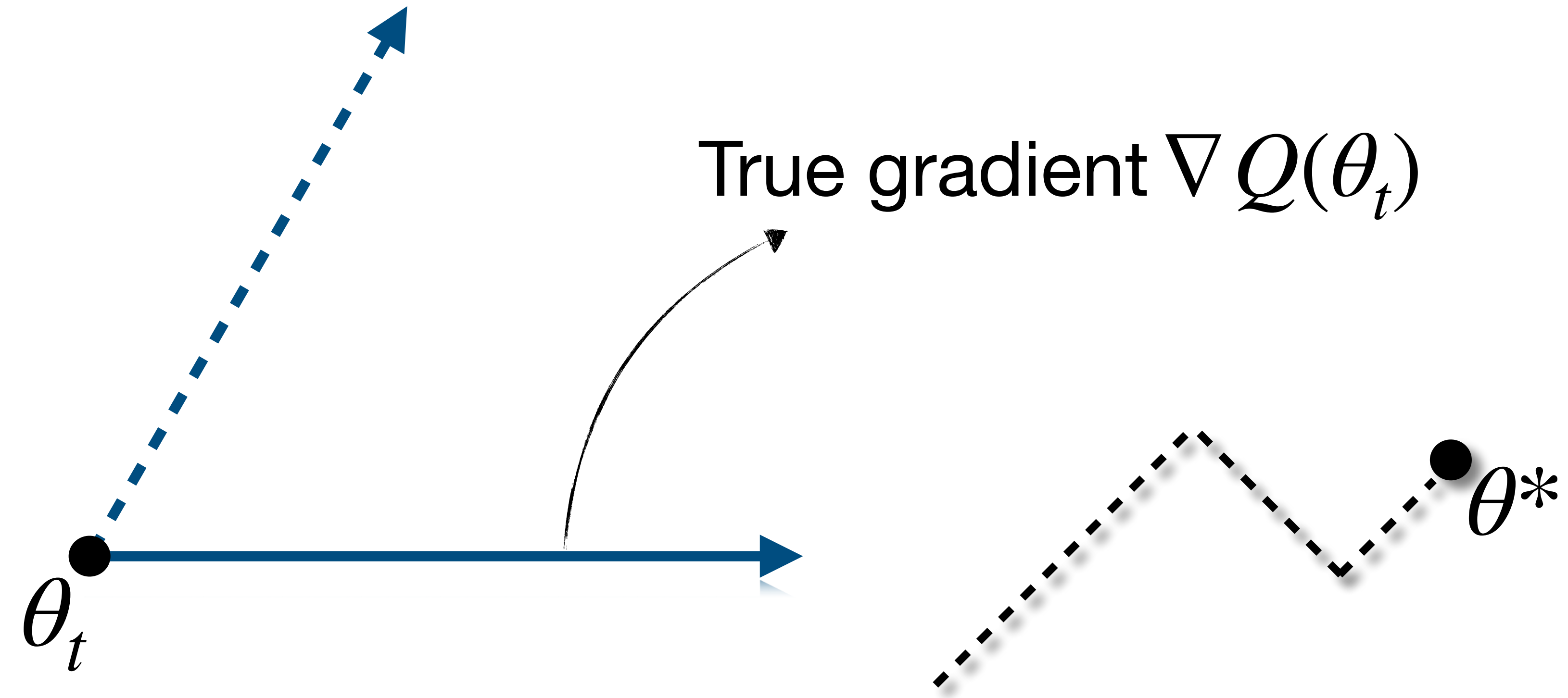
What is the Problem?

Non-trivial variance of stochastic gradients



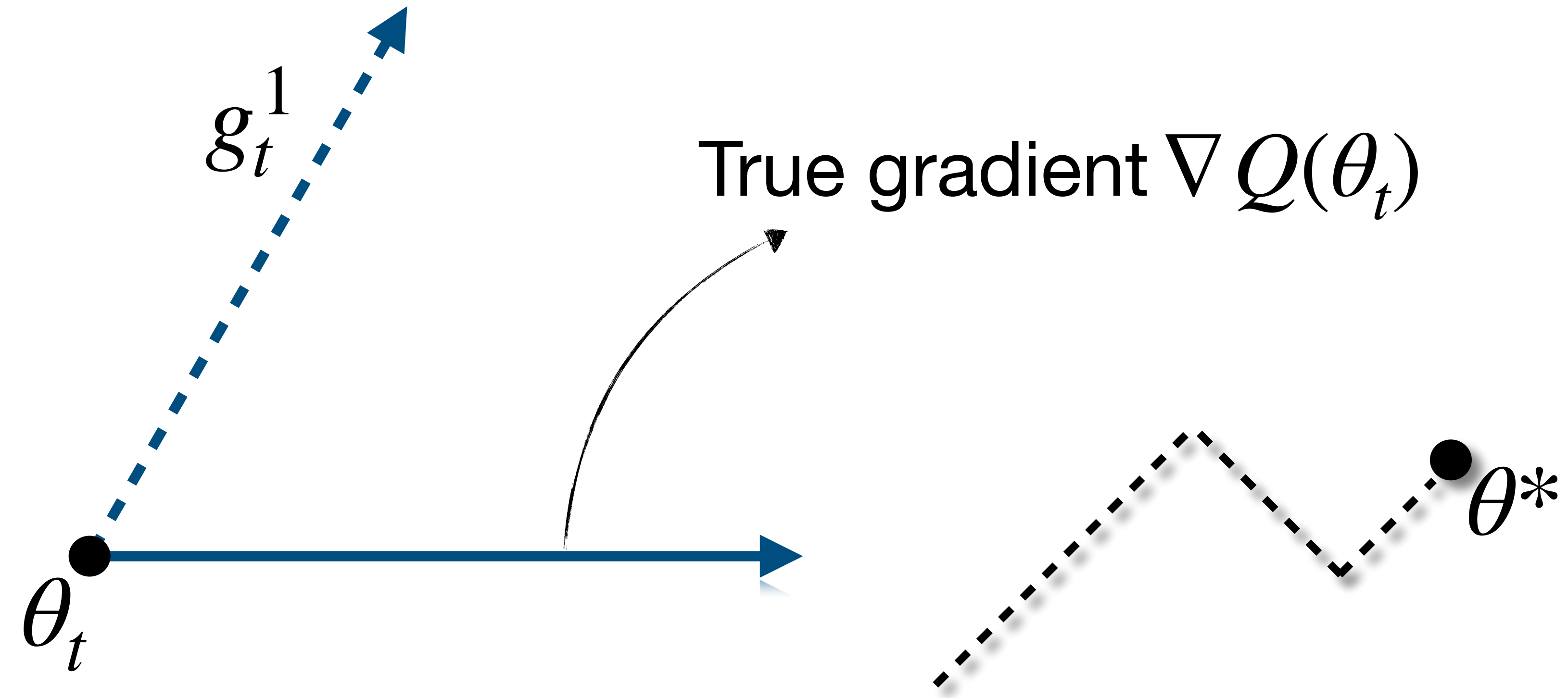
What is the Problem?

Non-trivial variance of stochastic gradients



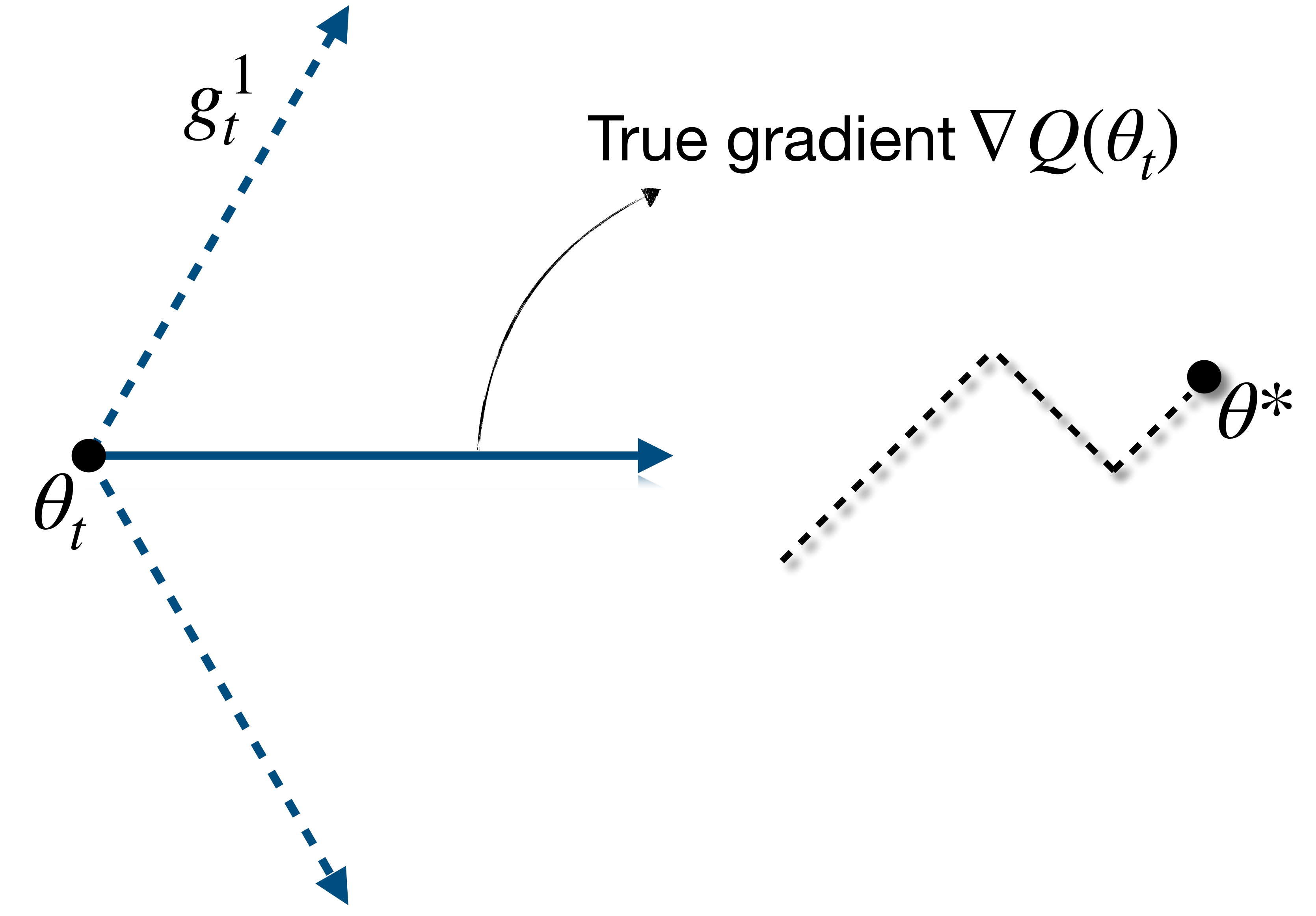
What is the Problem?

Non-trivial variance of stochastic gradients



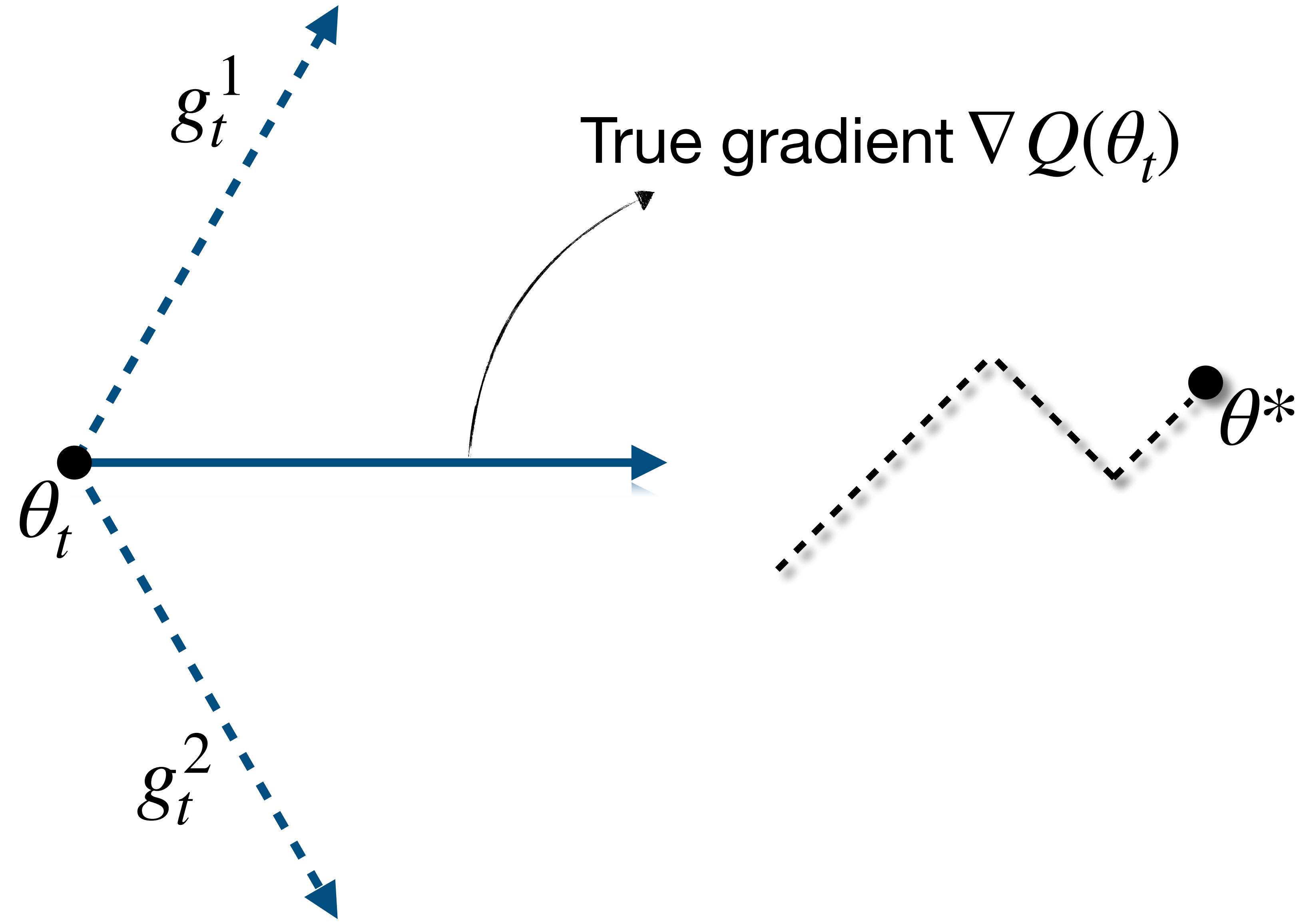
What is the Problem?

Non-trivial variance of stochastic gradients



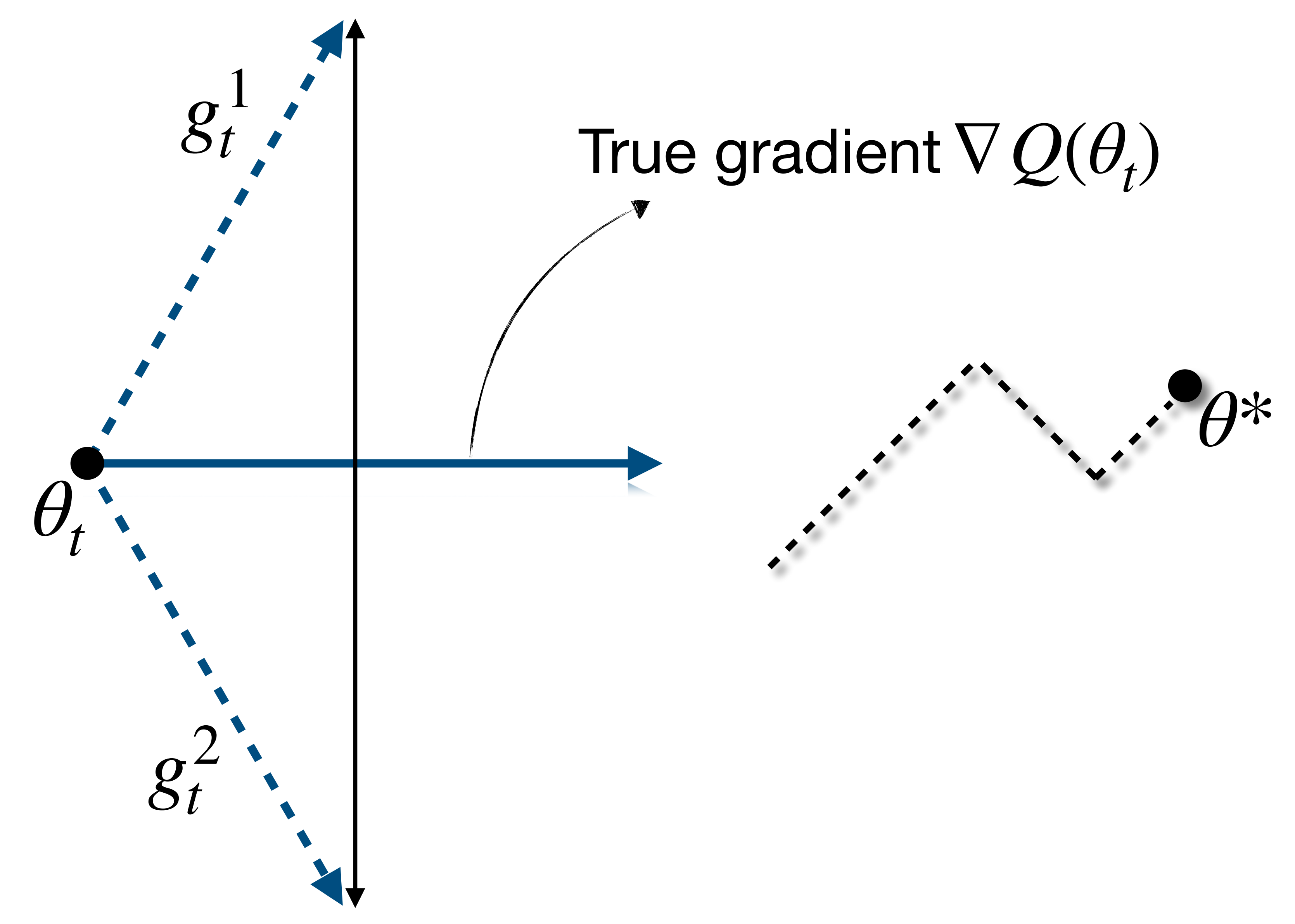
What is the Problem?

Non-trivial variance of stochastic gradients



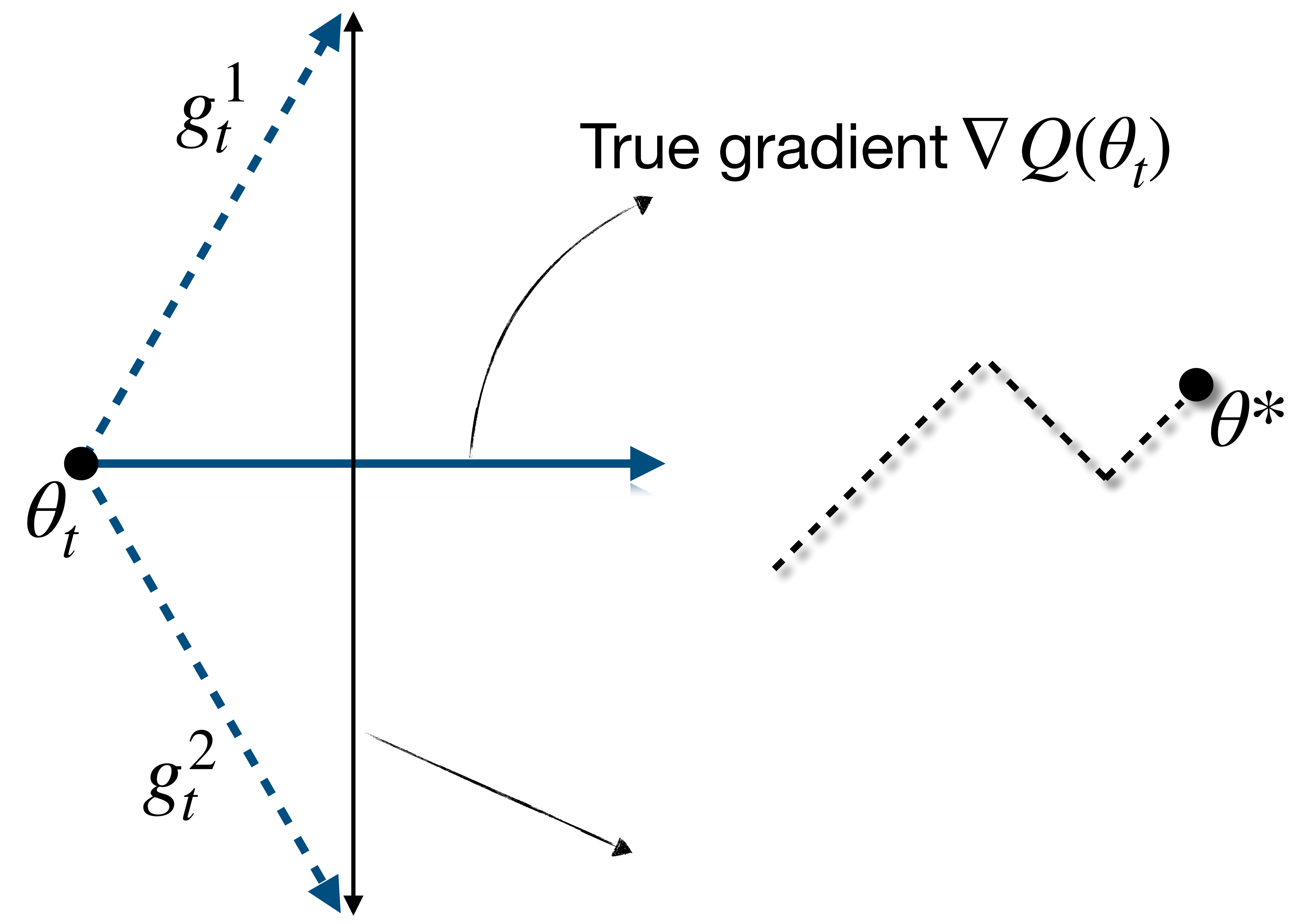
What is the Problem?

Non-trivial variance of stochastic gradients



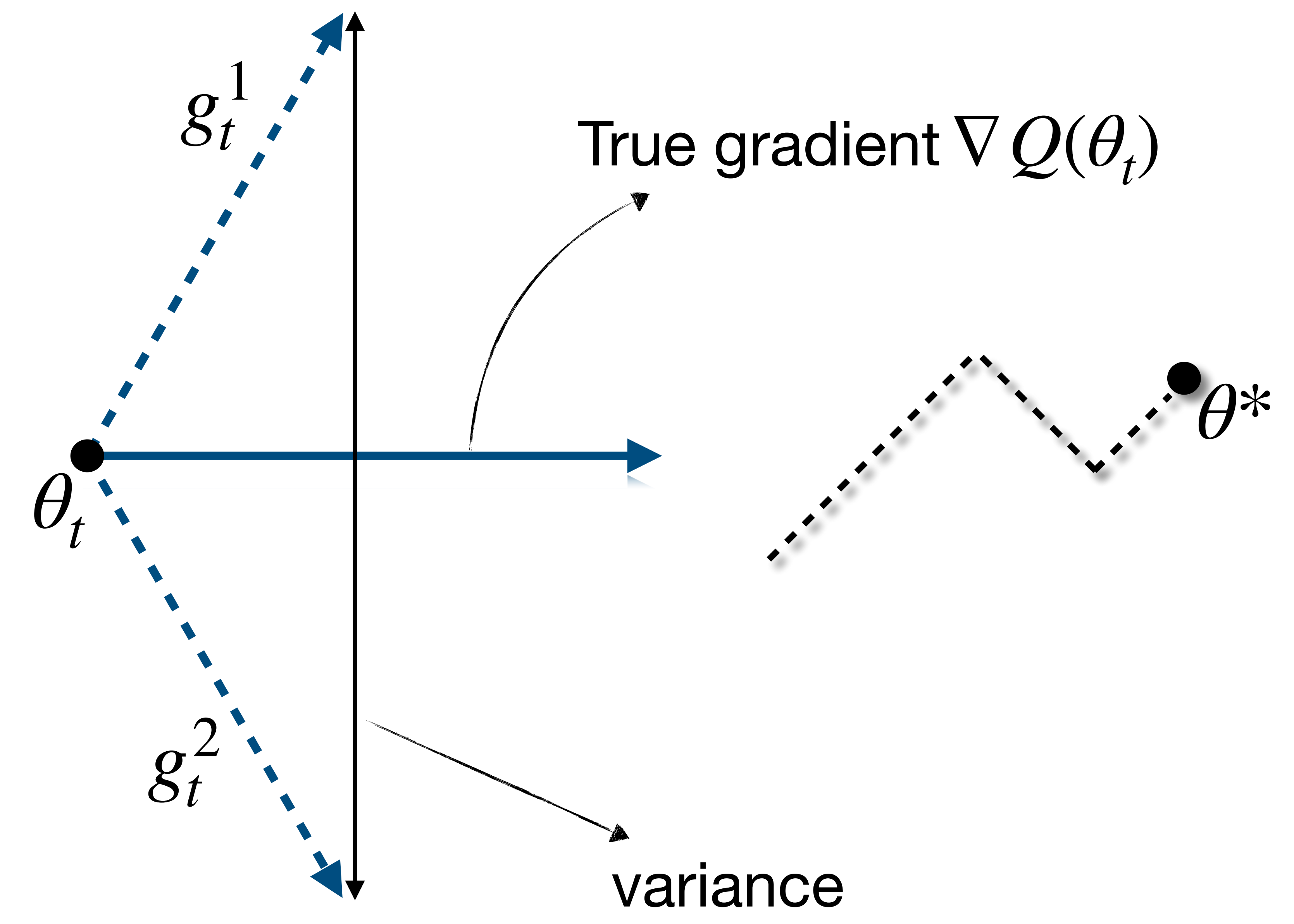
What is the Problem?

Non-trivial variance of stochastic gradients



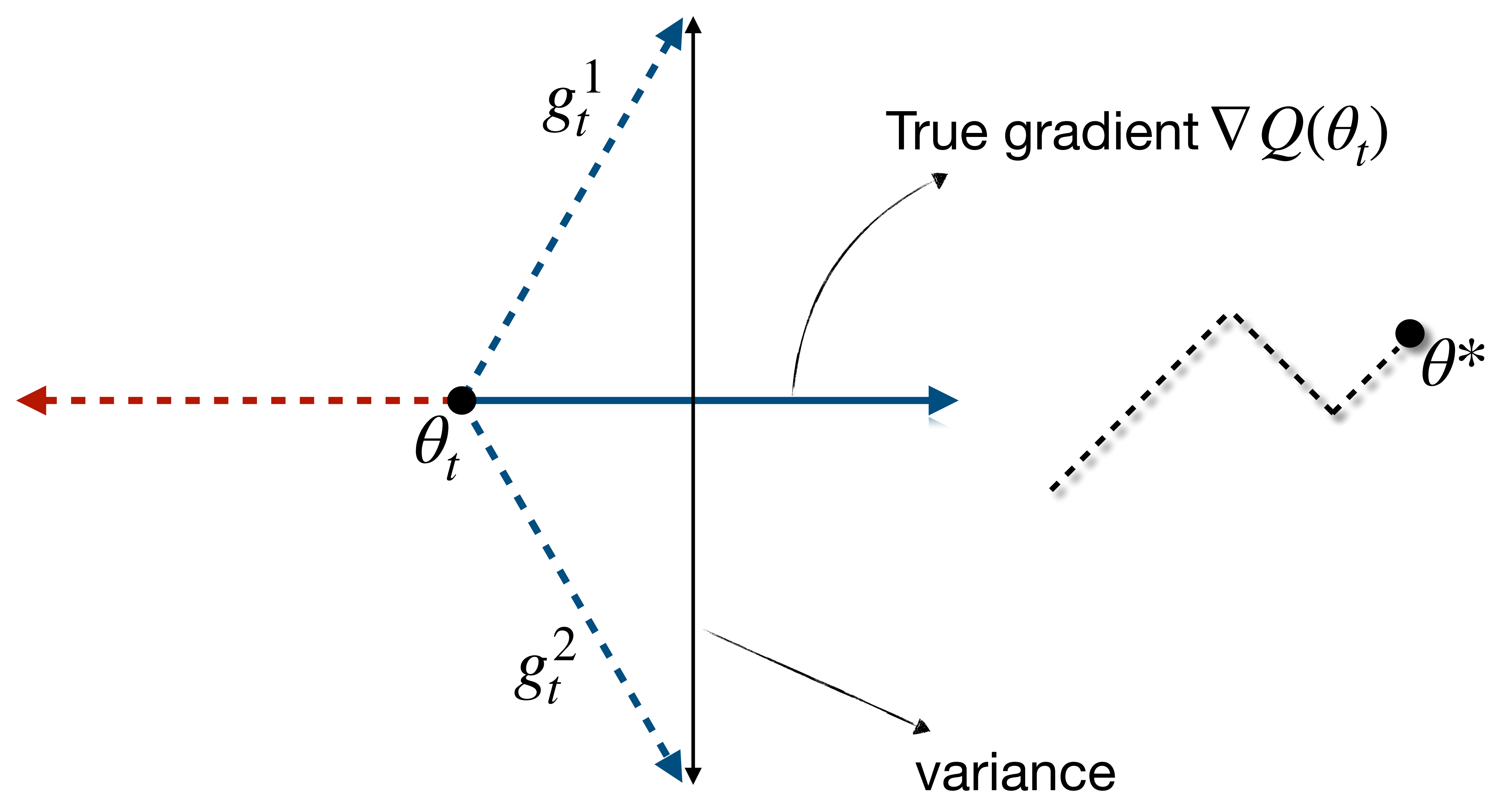
What is the Problem?

Non-trivial variance of stochastic gradients



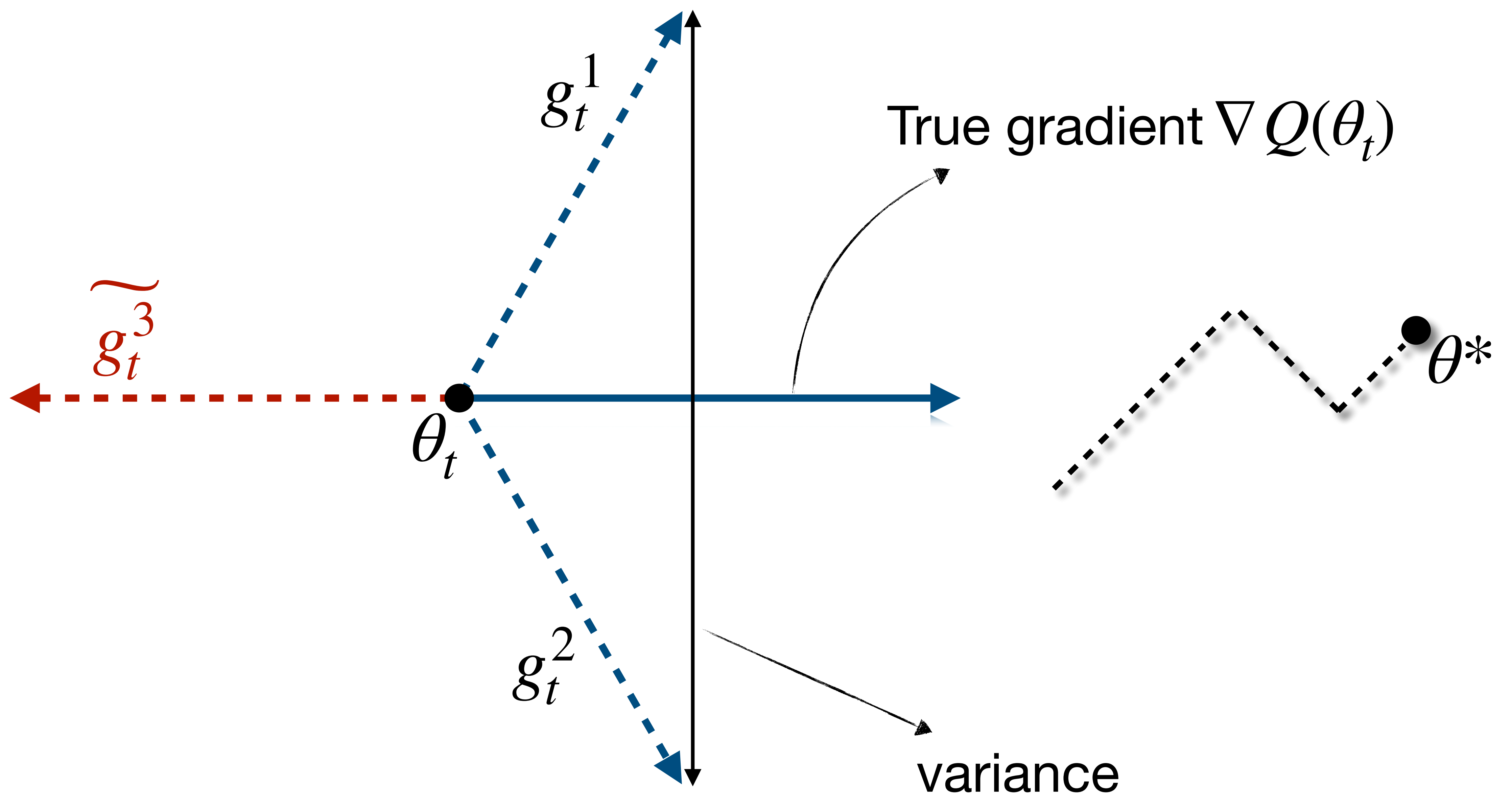
What is the Problem?

Non-trivial variance of stochastic gradients



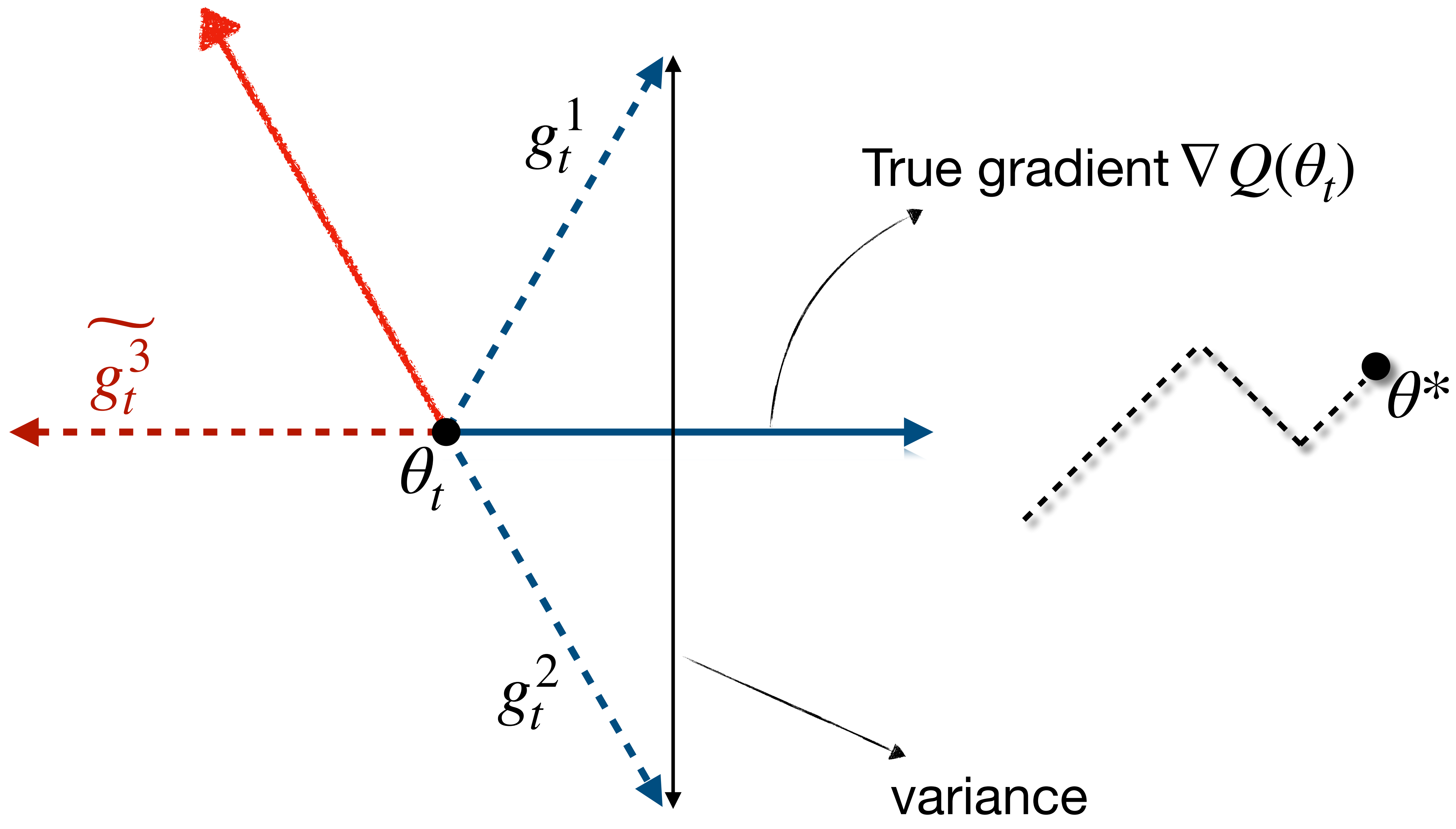
What is the Problem?

Non-trivial variance of stochastic gradients



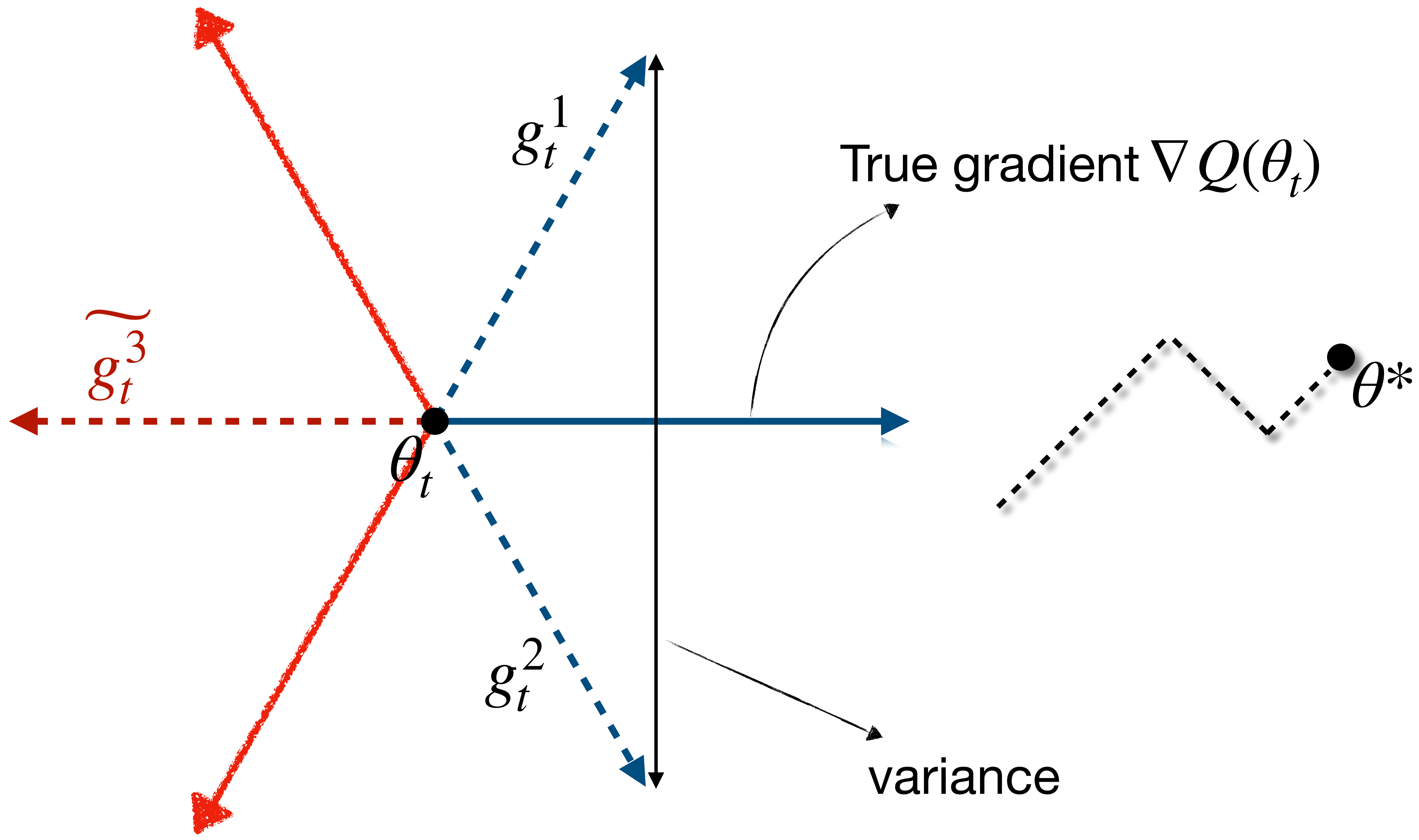
What is the Problem?

Non-trivial variance of stochastic gradients



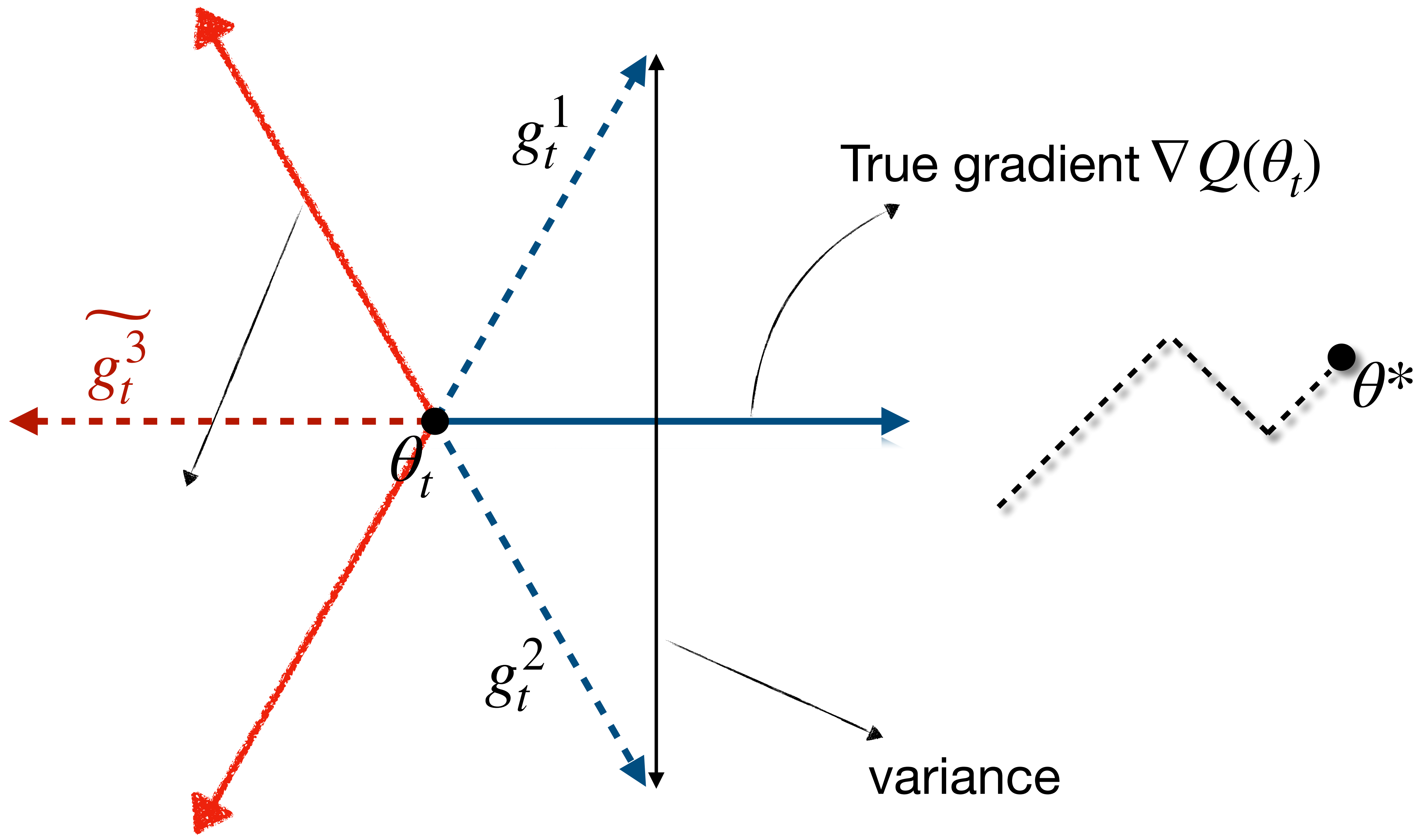
What is the Problem?

Non-trivial variance of stochastic gradients



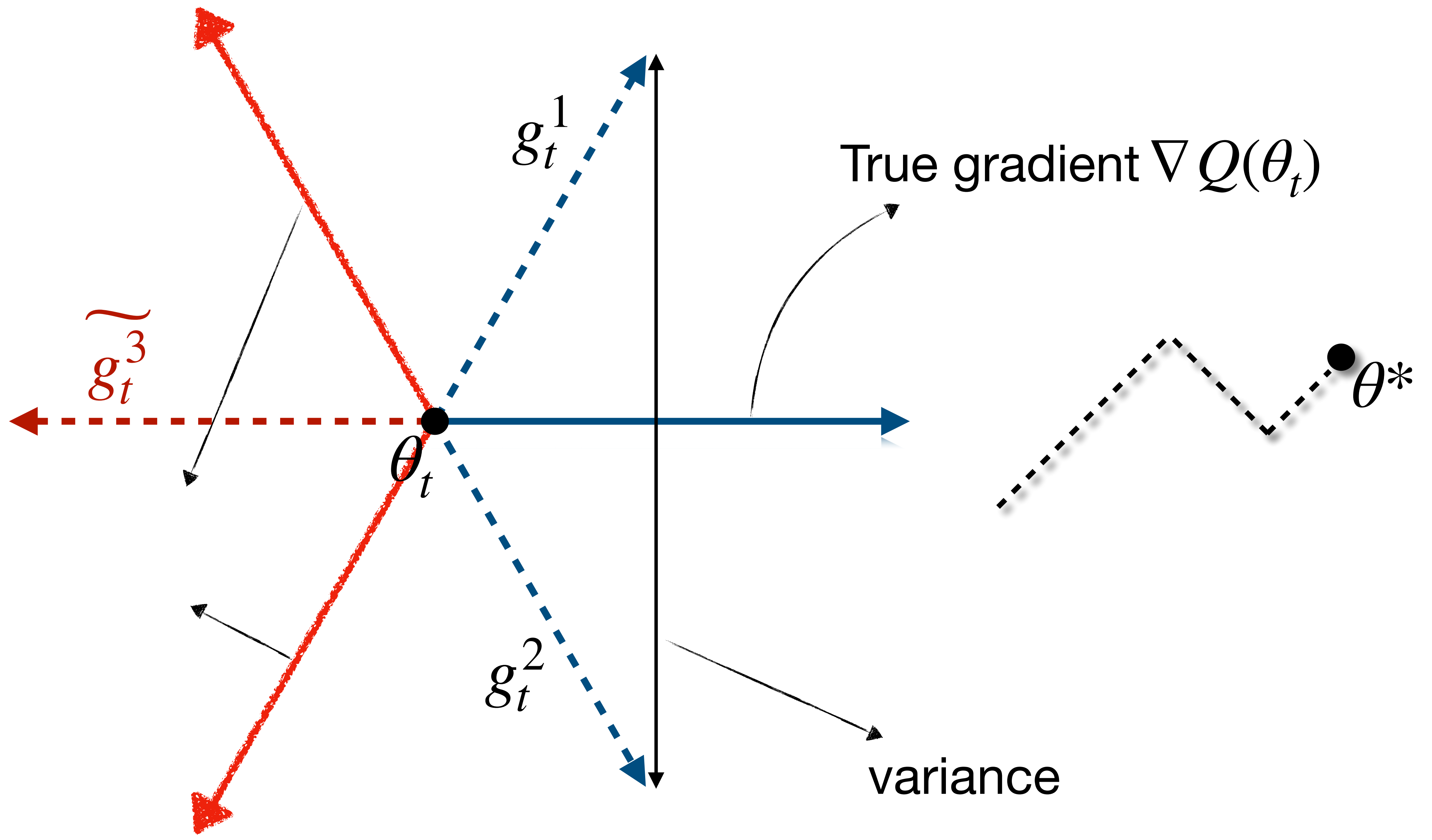
What is the Problem?

Non-trivial variance of stochastic gradients



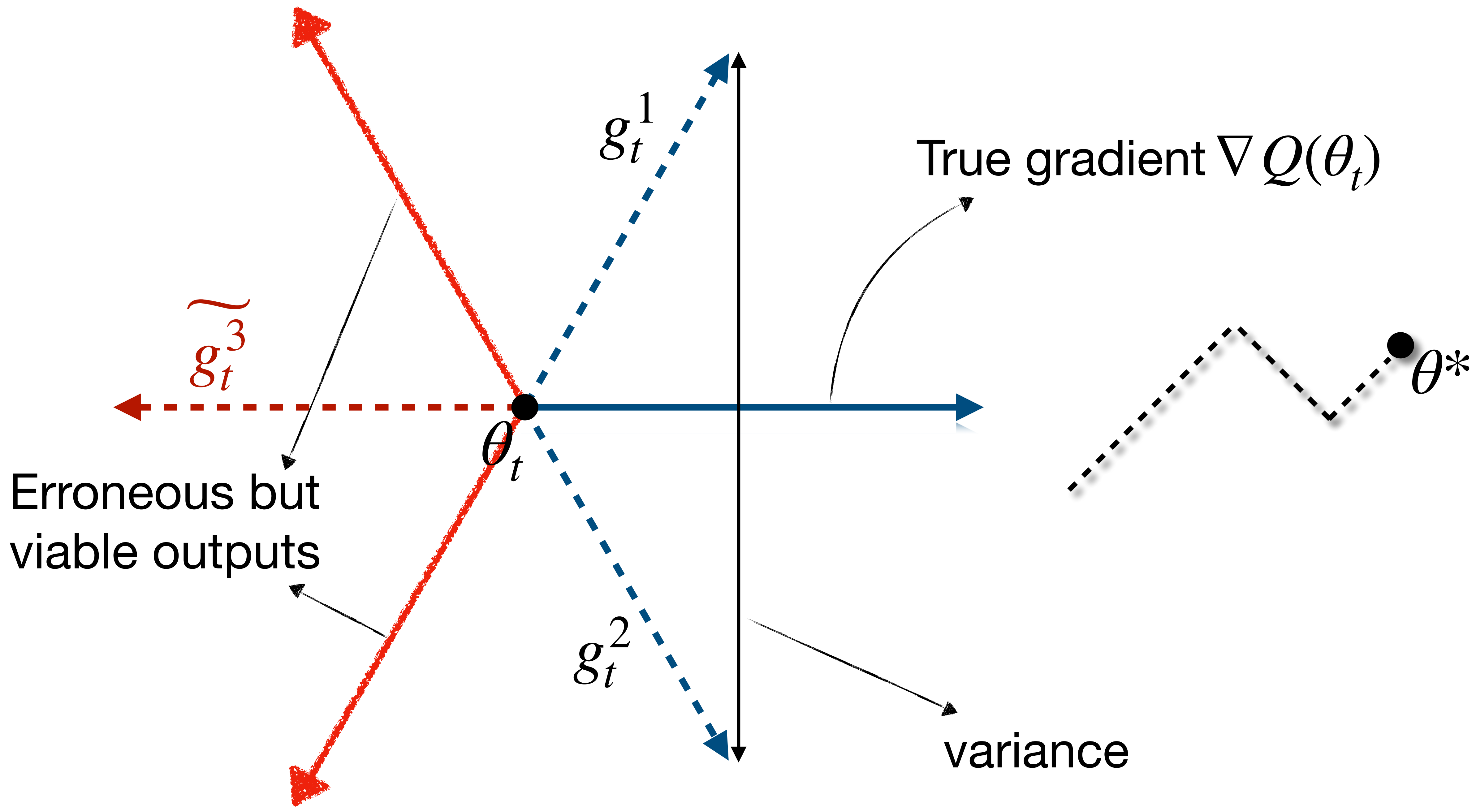
What is the Problem?

Non-trivial variance of stochastic gradients



What is the Problem?

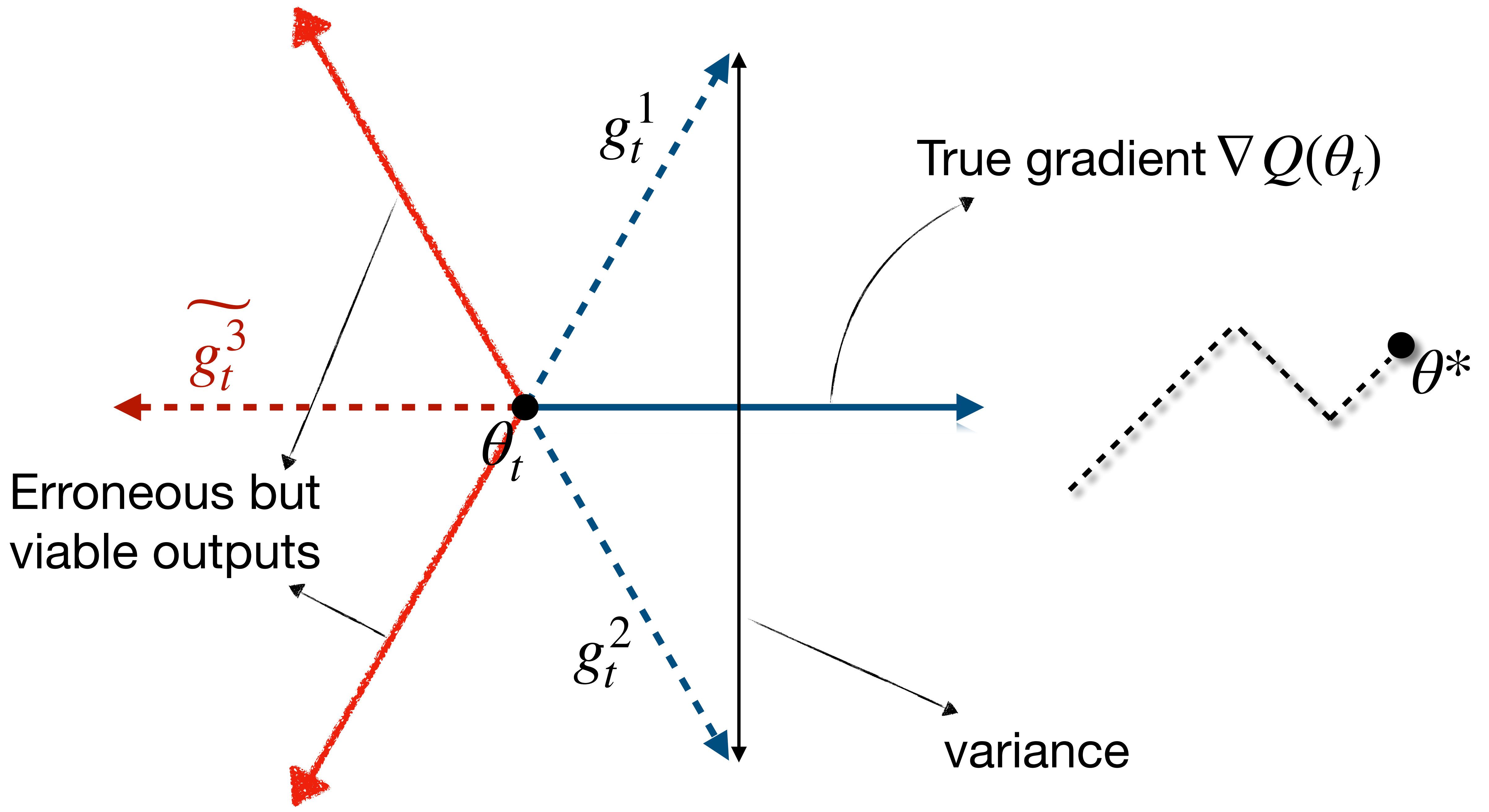
Non-trivial variance of stochastic gradients



What is the Problem?

Non-trivial variance of stochastic gradients

The worst-case guarantees
are quite bad

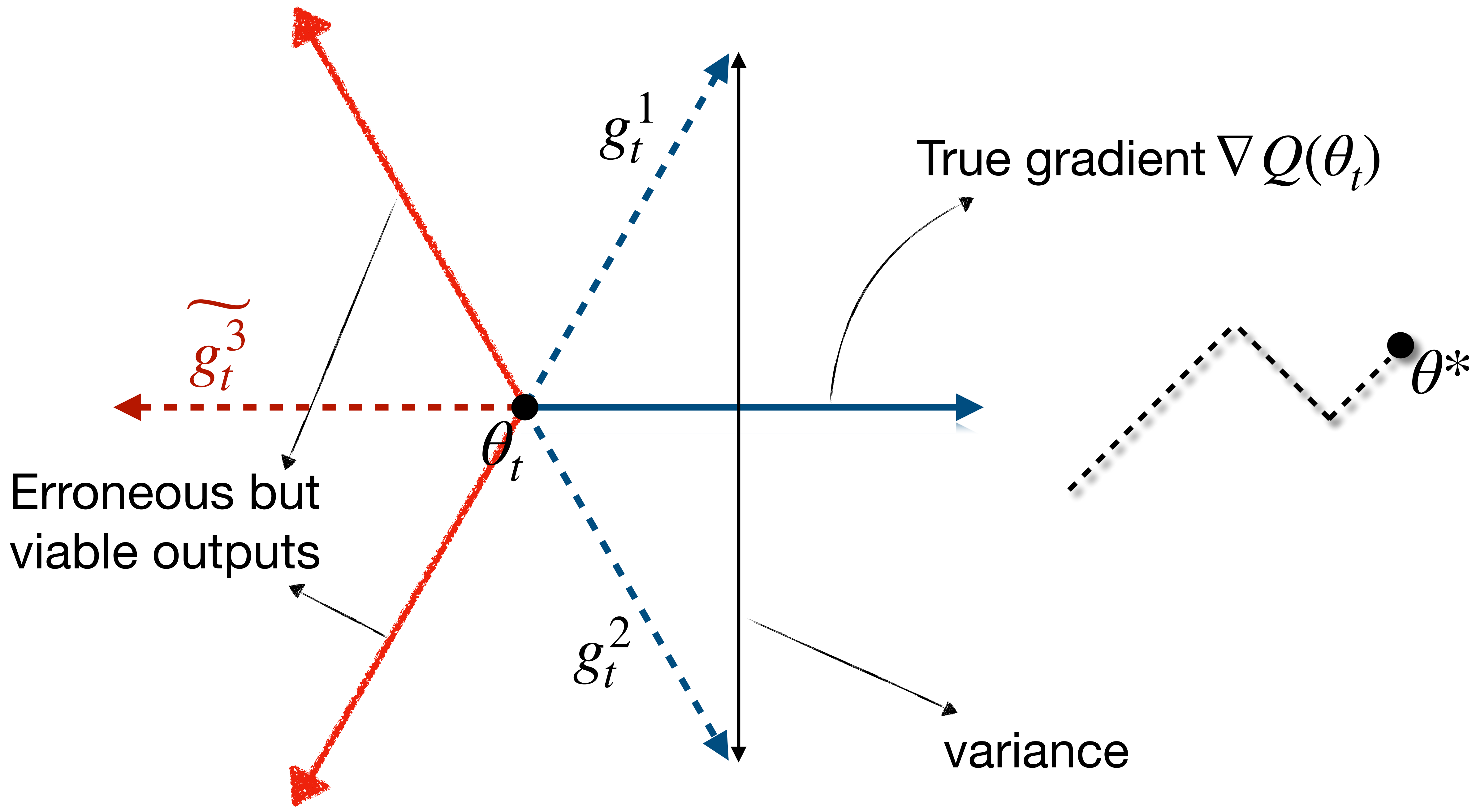


What is the Problem?

Non-trivial variance of stochastic gradients

The worst-case guarantees are quite bad

Variance reduction schemes -



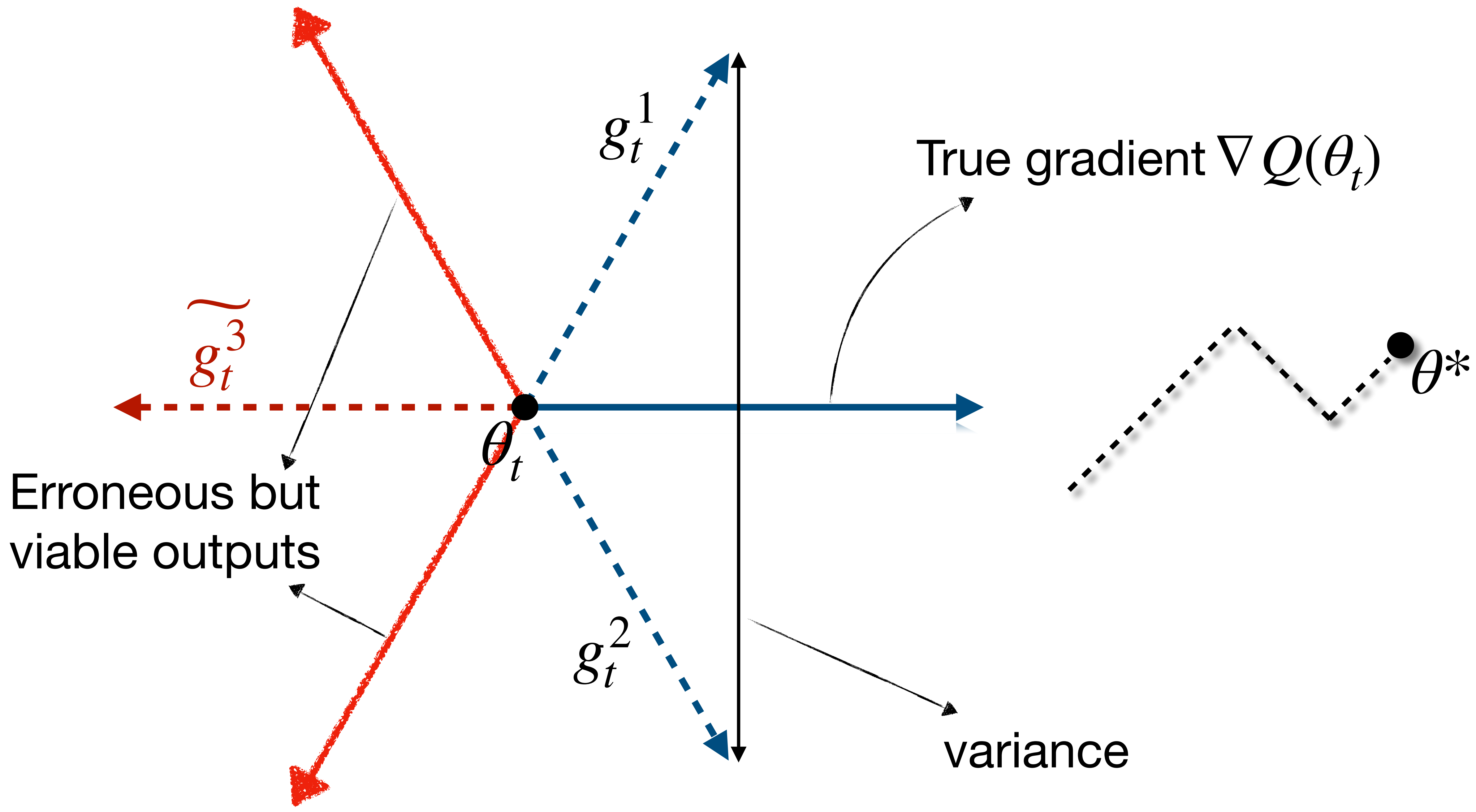
What is the Problem?

Non-trivial variance of stochastic gradients

The worst-case guarantees
are quite bad

Variance reduction schemes -

Computationally **expensive** (querying multiple gradients)



What is the Problem?

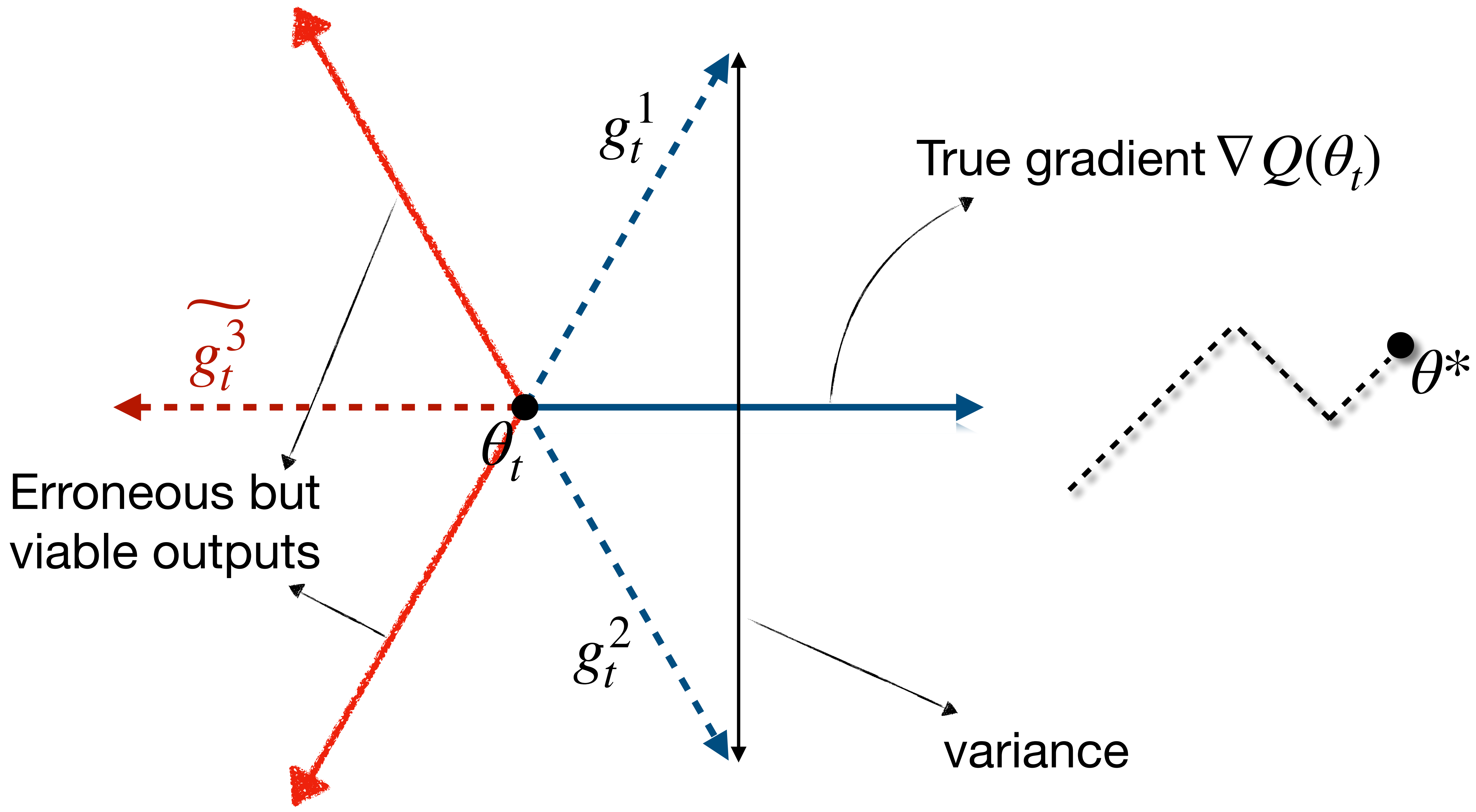
Non-trivial variance of stochastic gradients

The worst-case guarantees
are quite bad

Variance reduction schemes -

Computationally **expensive** (querying multiple gradients)

Leads to **bias in gradient estimate**



What is the Problem?

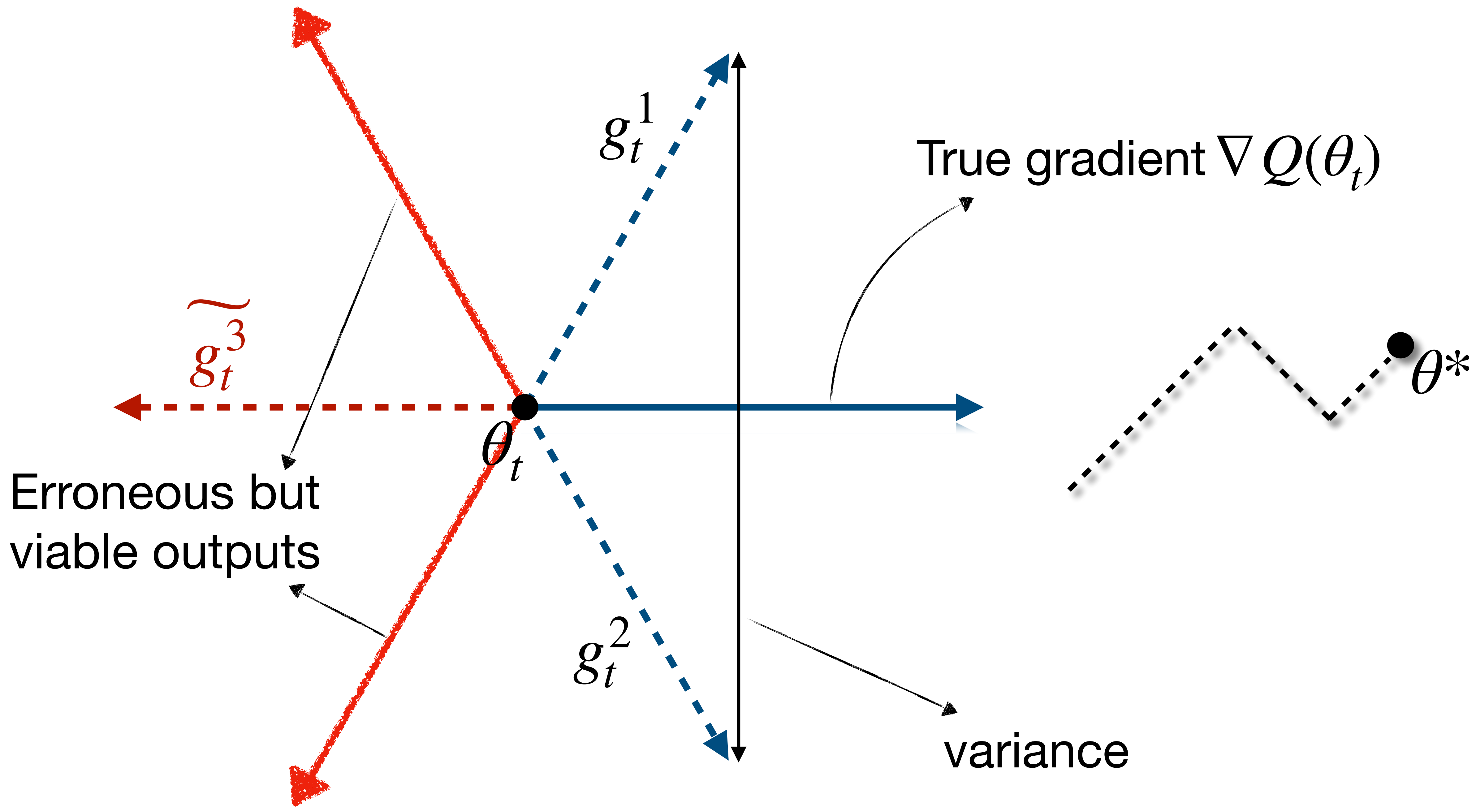
Non-trivial variance of stochastic gradients

The worst-case guarantees
are quite bad

Variance reduction schemes -

Computationally **expensive** (querying multiple gradients)

Leads to **bias in gradient estimate**



What is the Problem?

Non-trivial variance of stochastic gradients

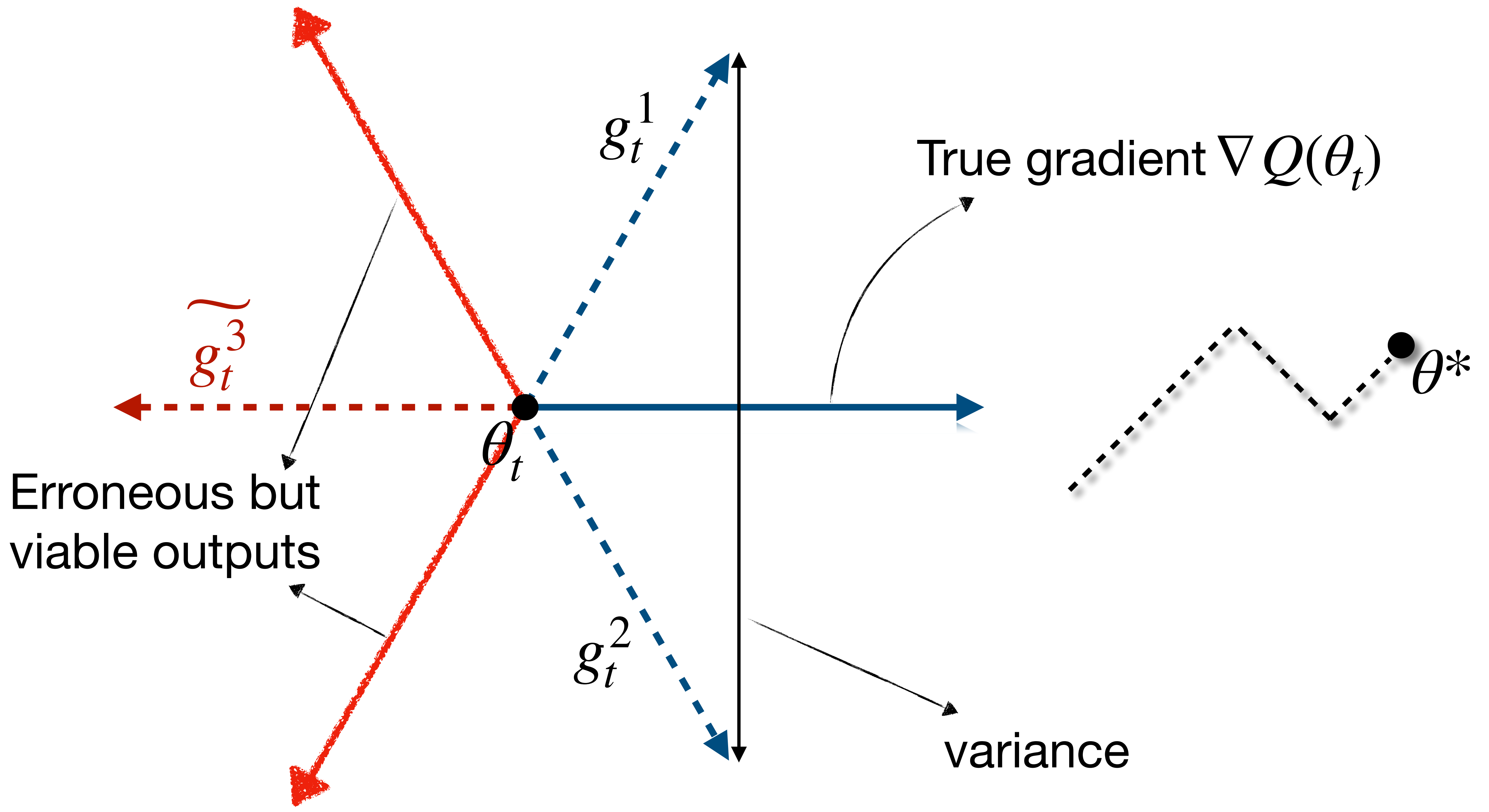
The worst-case guarantees
are quite bad

Variance reduction schemes -

Computationally **expensive** (querying multiple gradients)

Leads to **bias in gradient estimate**

Threatens the convergence



What is the Problem?

Non-trivial variance of stochastic gradients

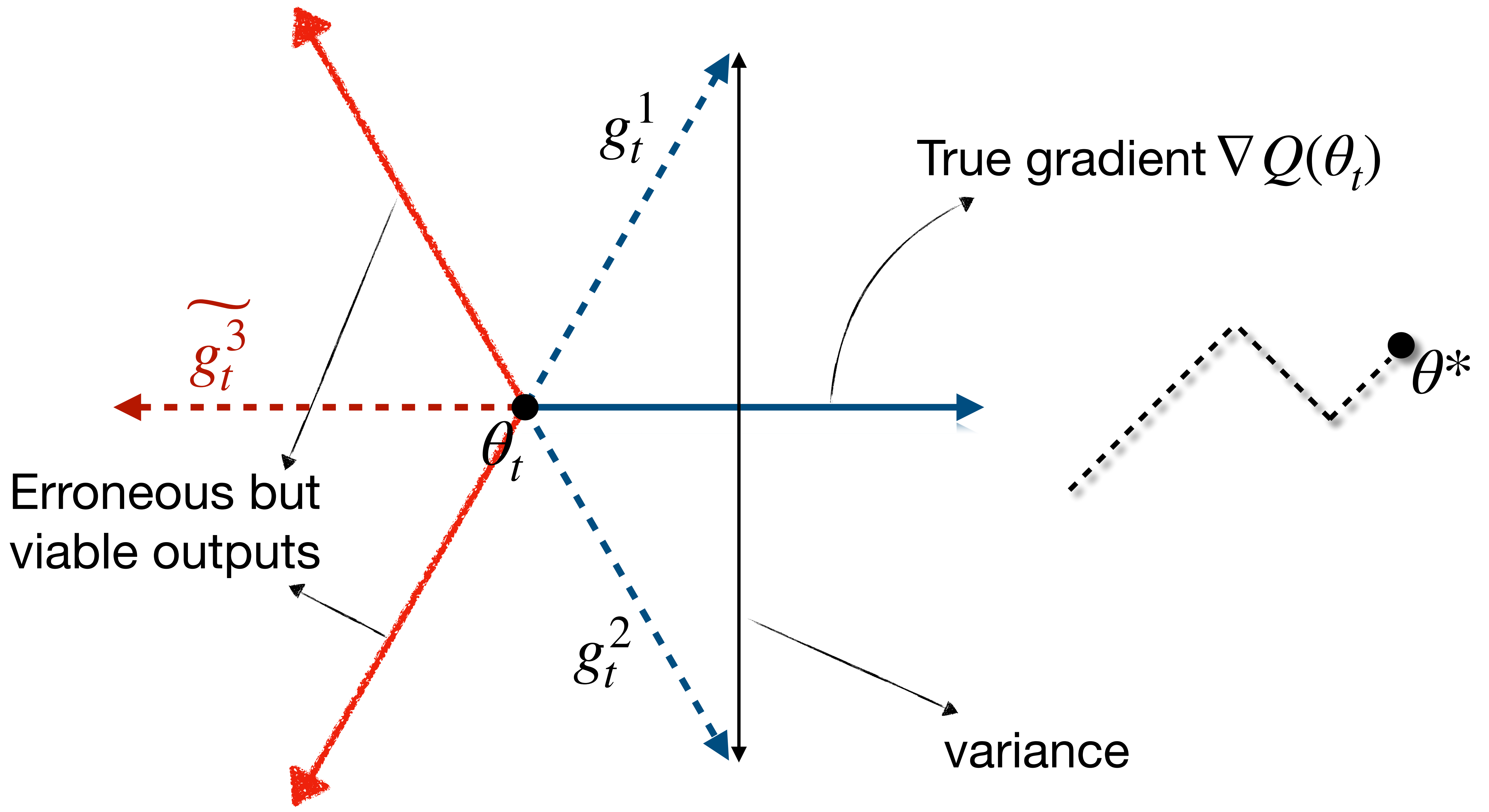
The worst-case guarantees
are quite bad

Variance reduction schemes -

Computationally **expensive** (querying multiple gradients)

Leads to **bias in gradient estimate**

Threatens the convergence

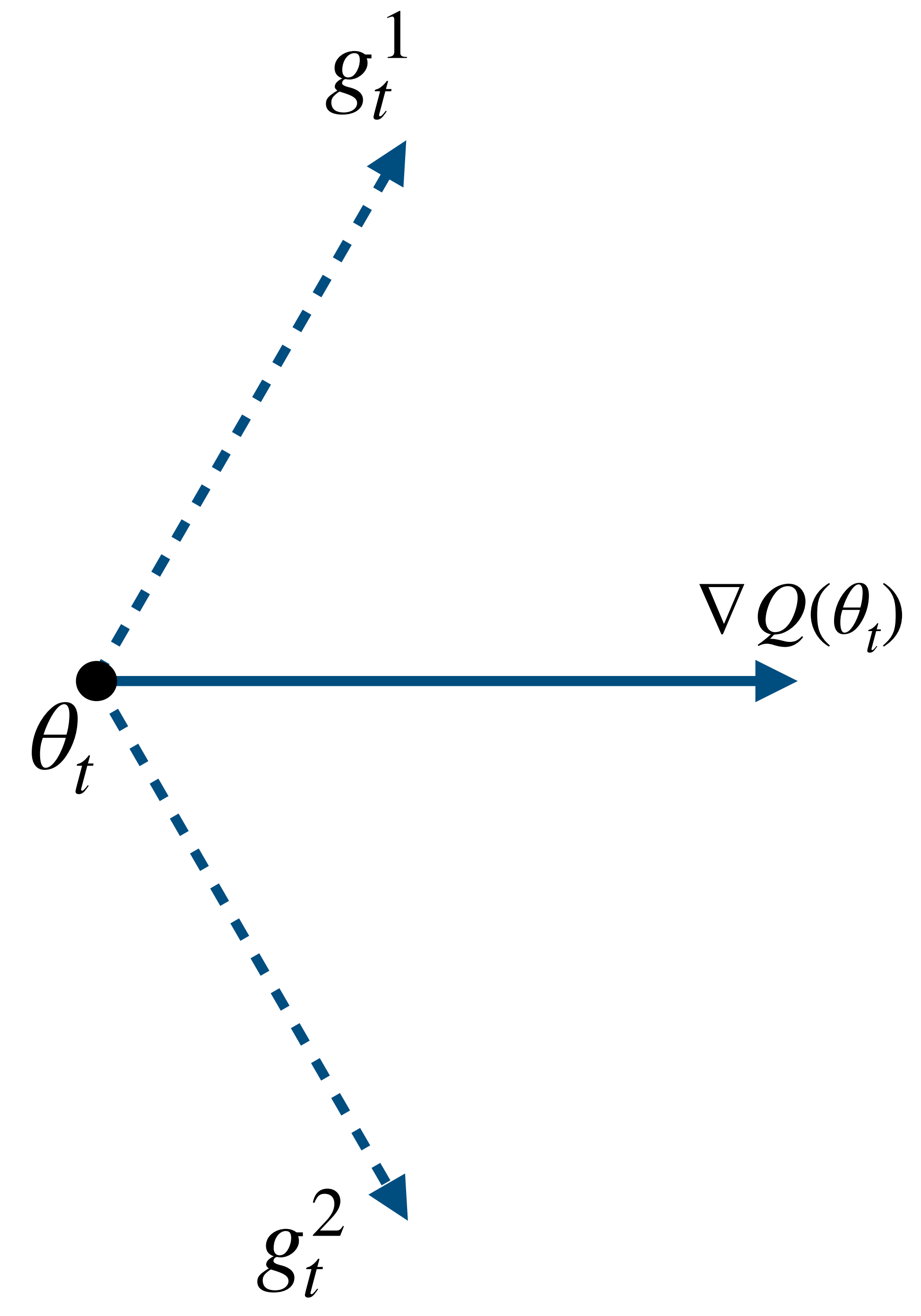


“With every mistake, me must surely be learning …

While my *GPU* gently weeps.”

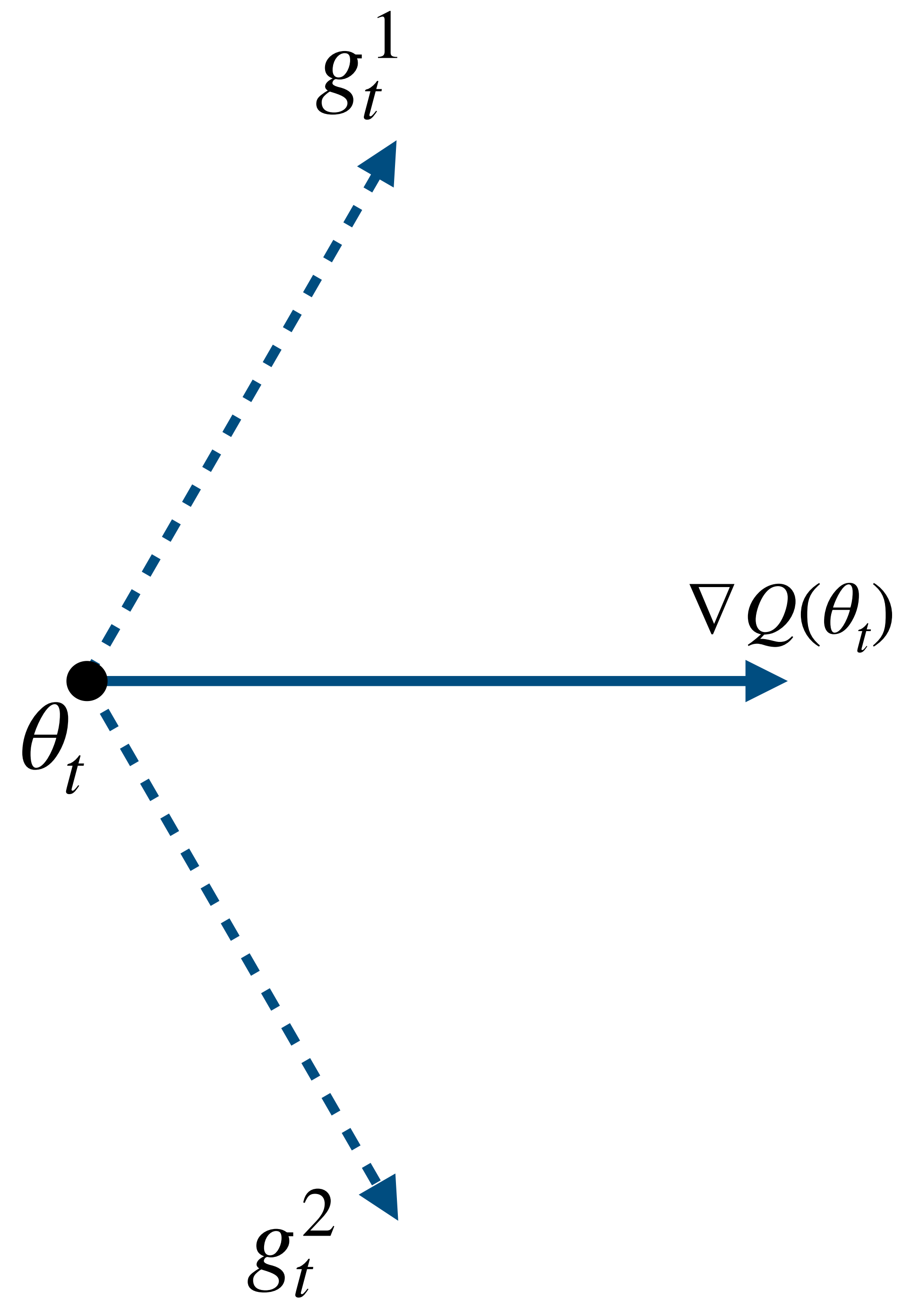
- George Harrison

Local Gradient Momentum



Local Gradient Momentum

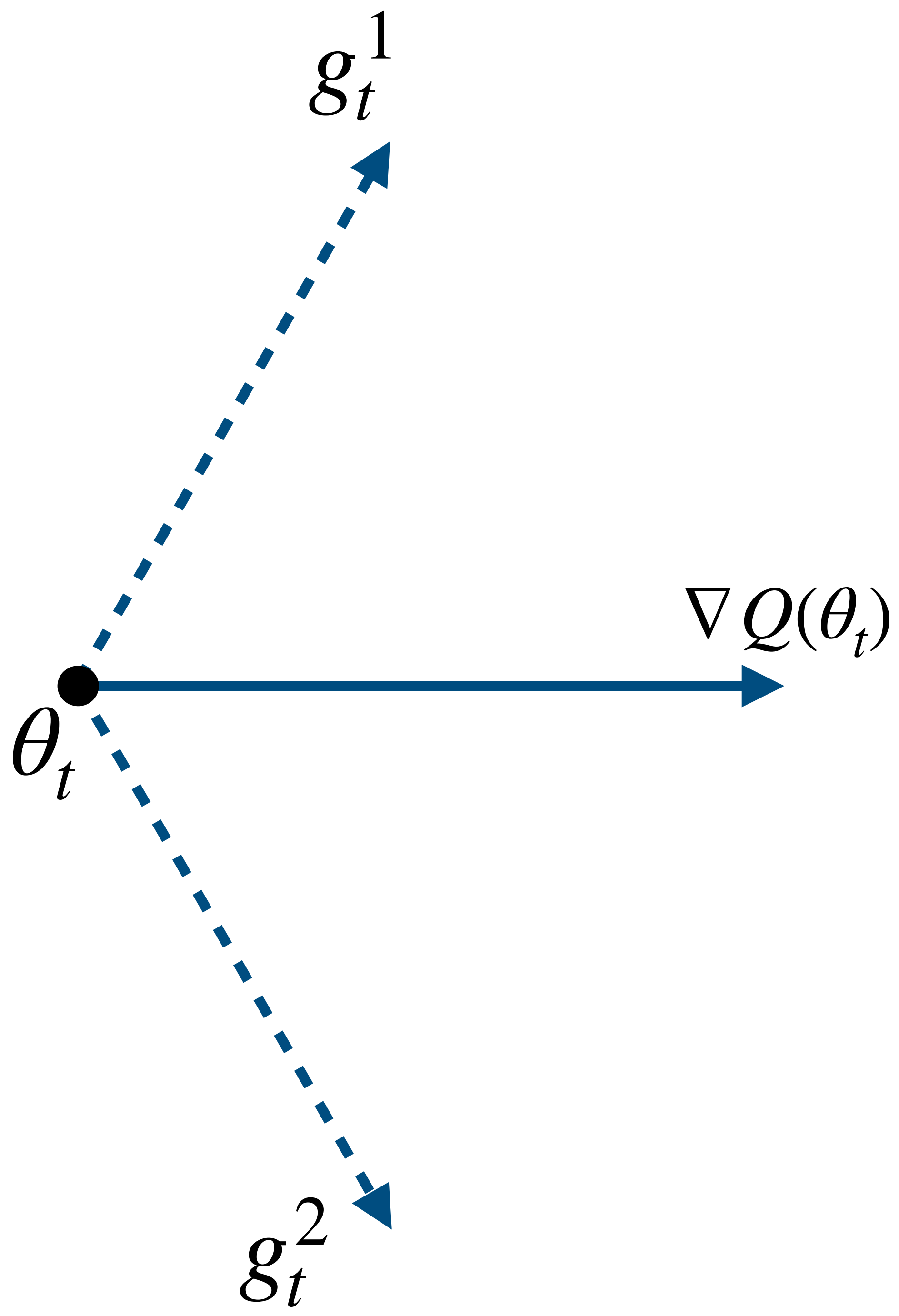
Nodes compute and **send momentums** of their gradients,



Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

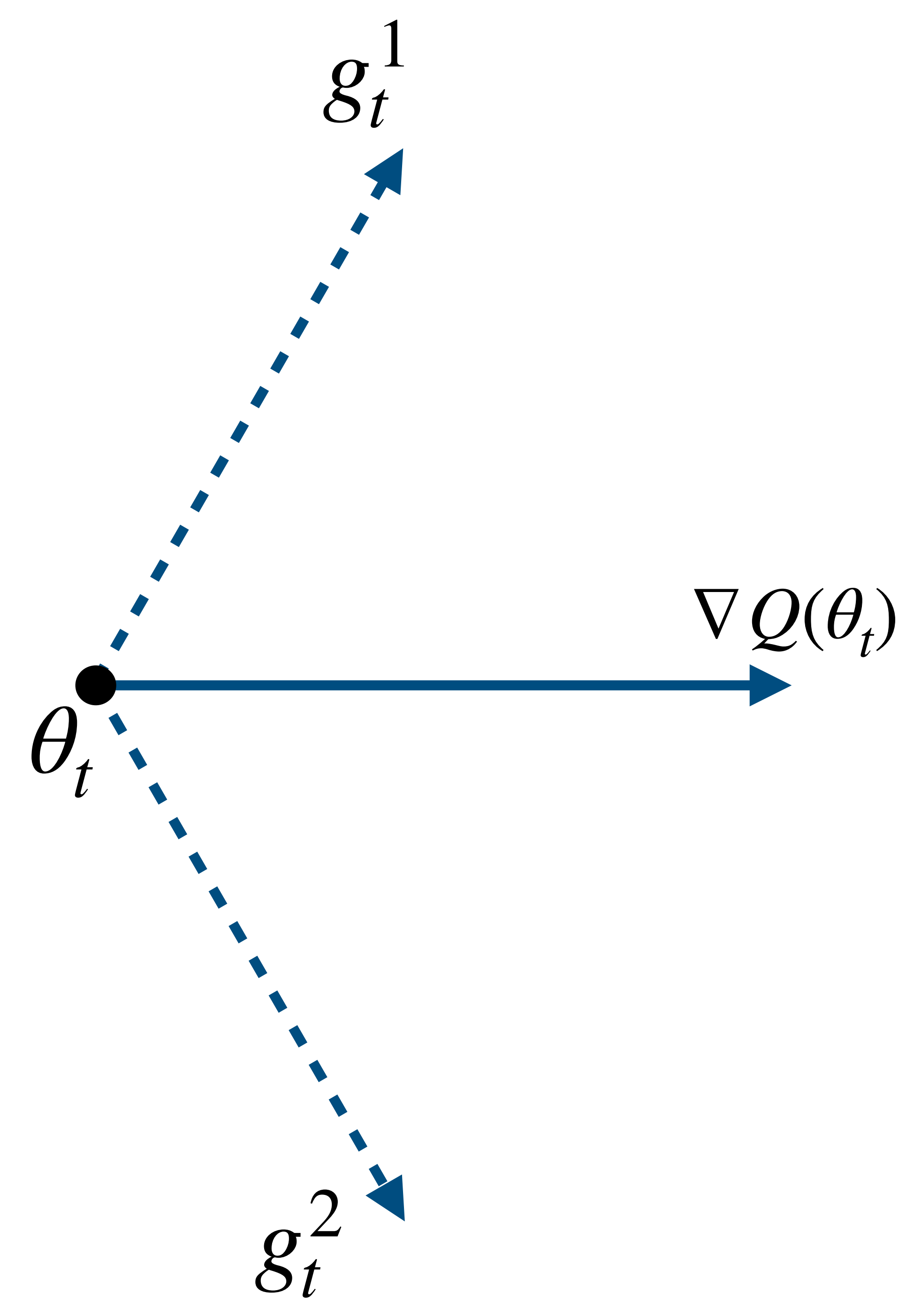


Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1 - \beta)\, g_t^i$$

$\beta \in [0, 1)$ is referred as the momentum coefficient

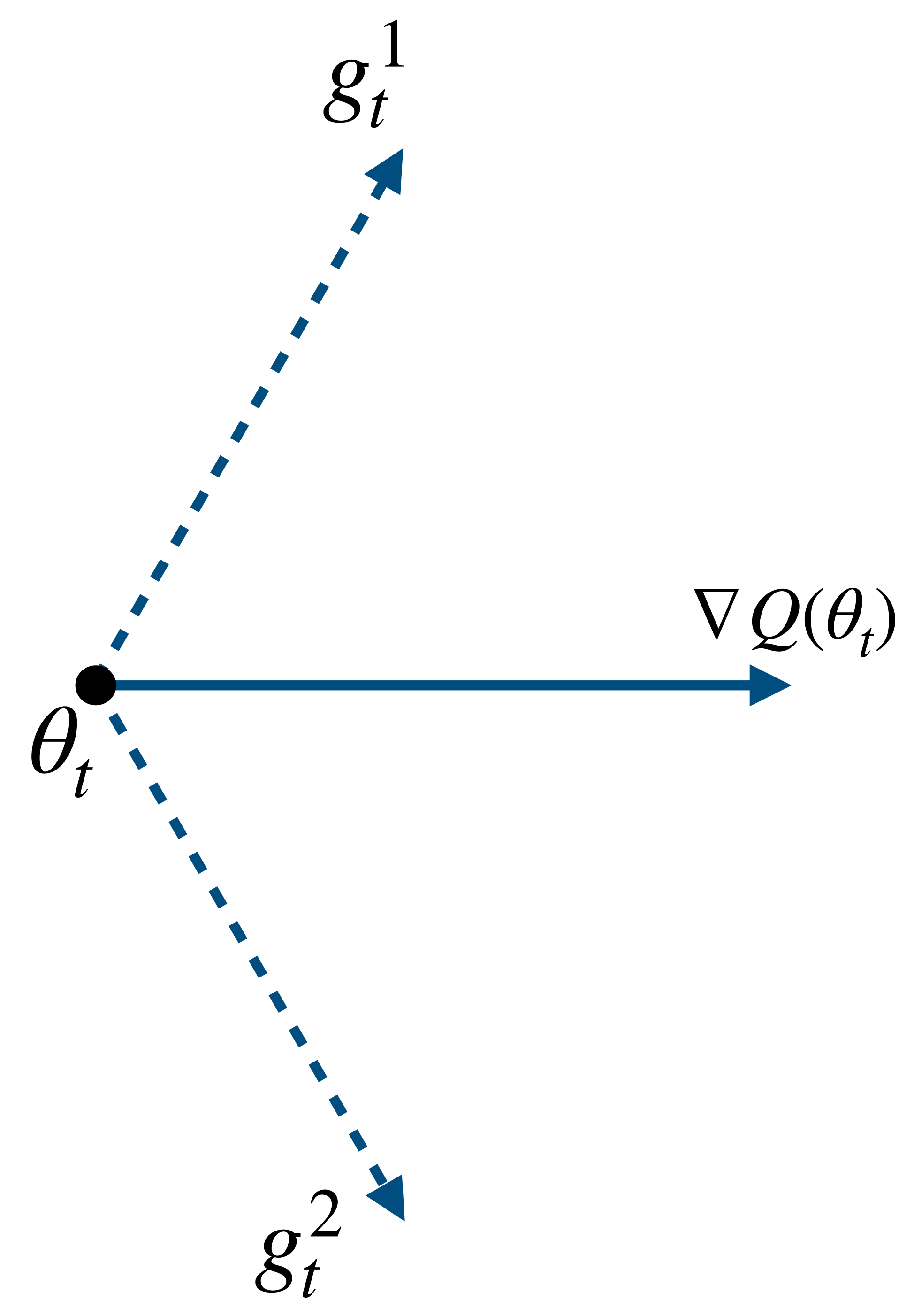


Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

$\beta \in [0, 1)$ is referred as the momentum coefficient

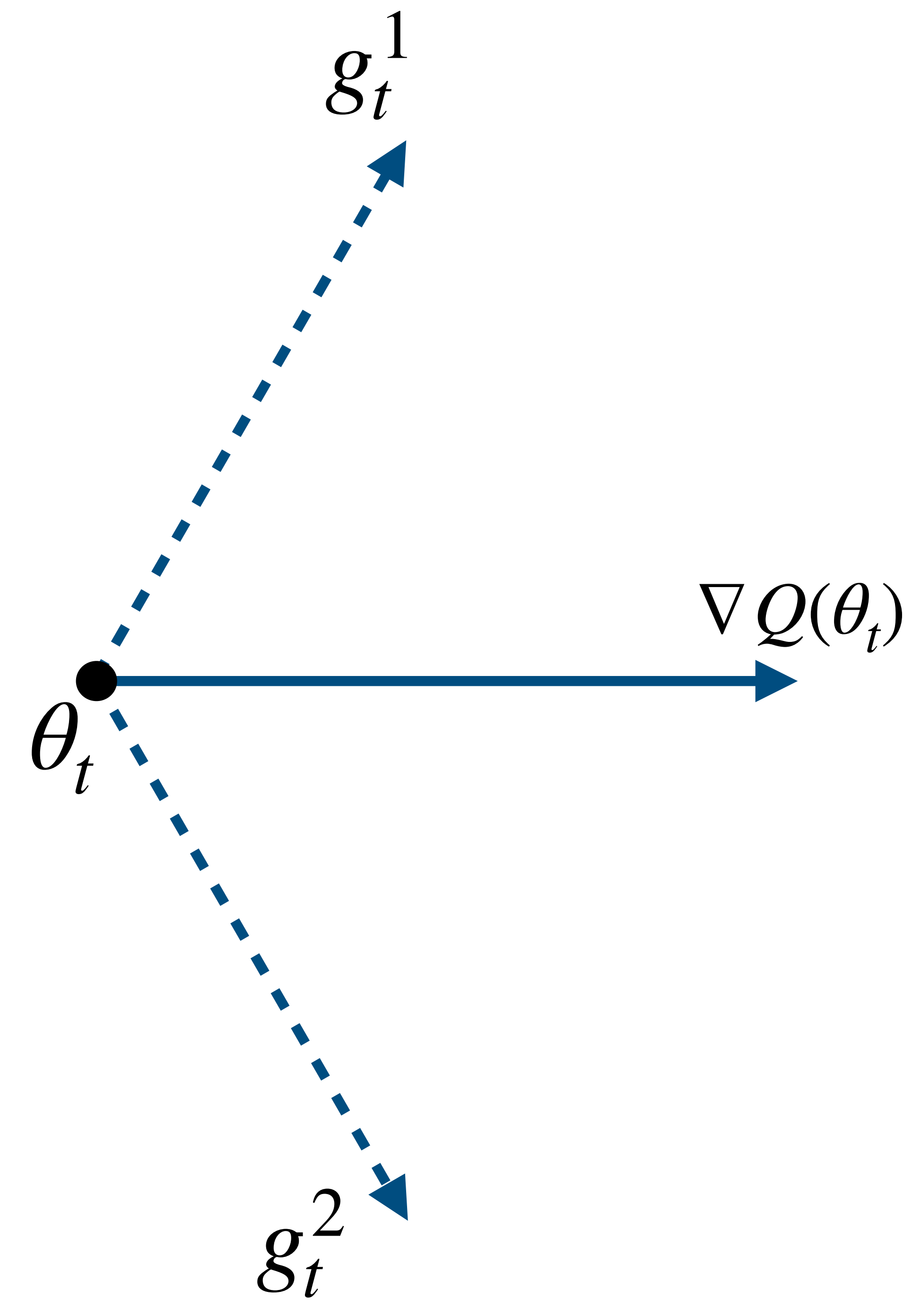


Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

$\beta \in [0, 1)$ is referred as the momentum coefficient



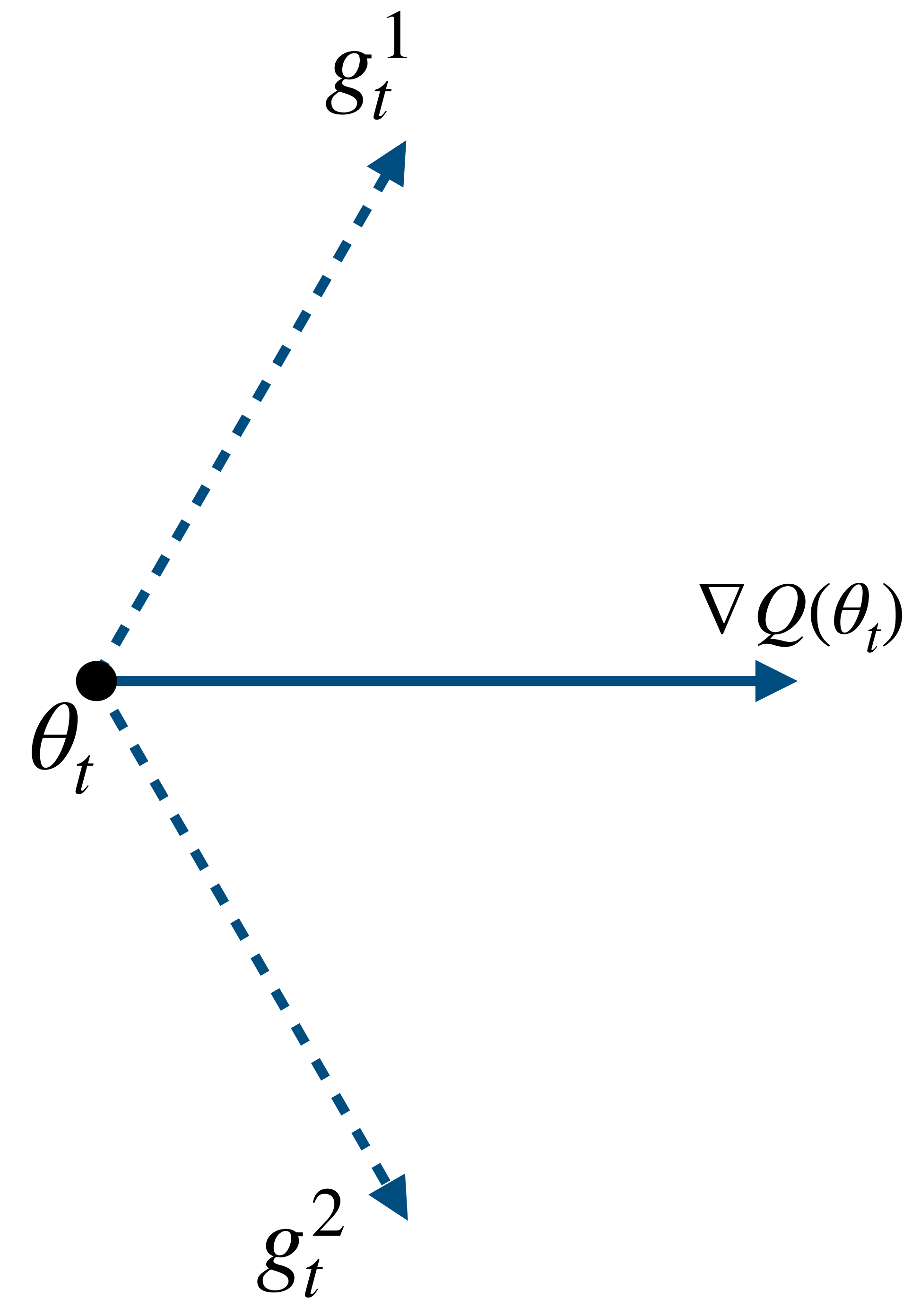
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



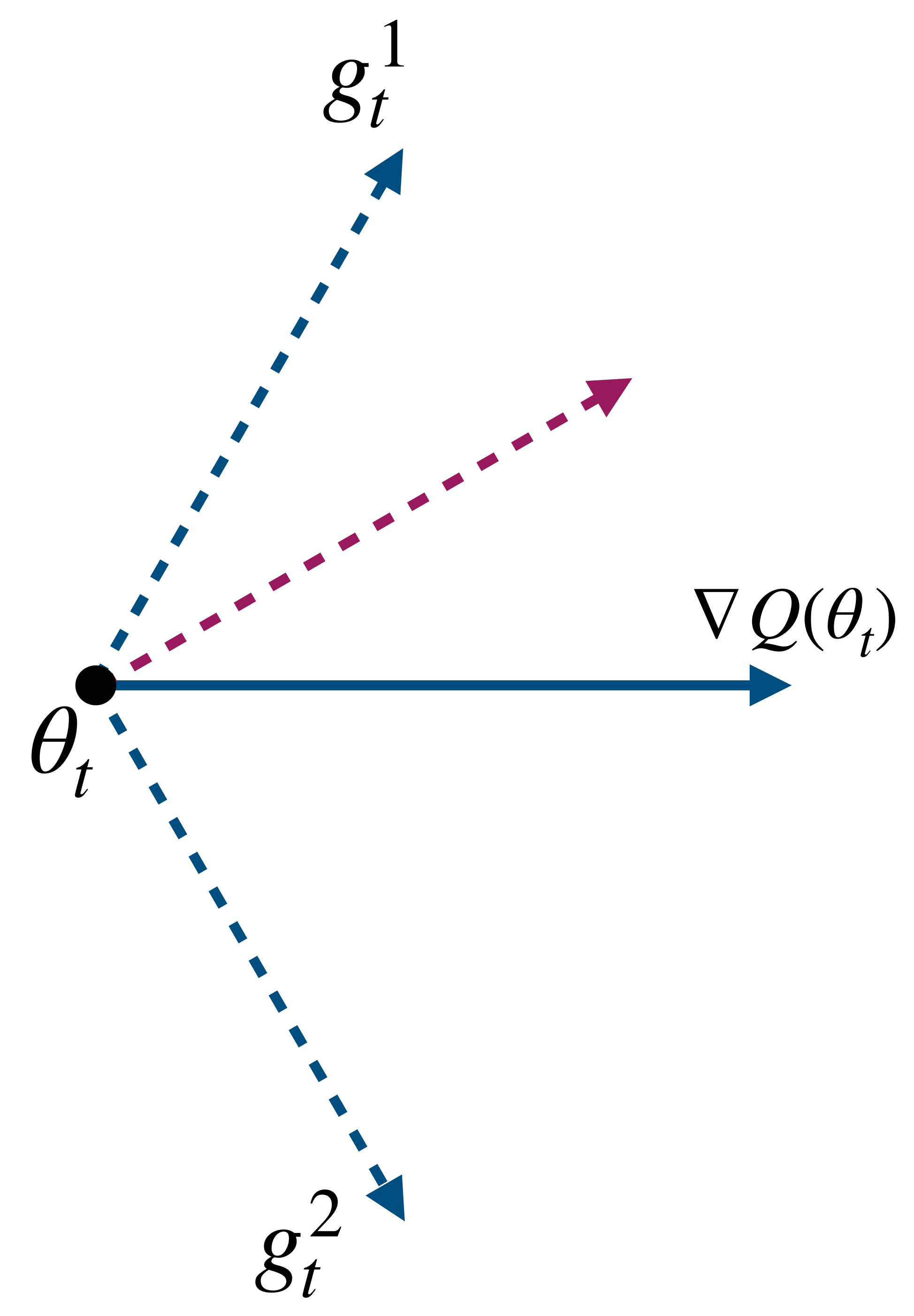
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



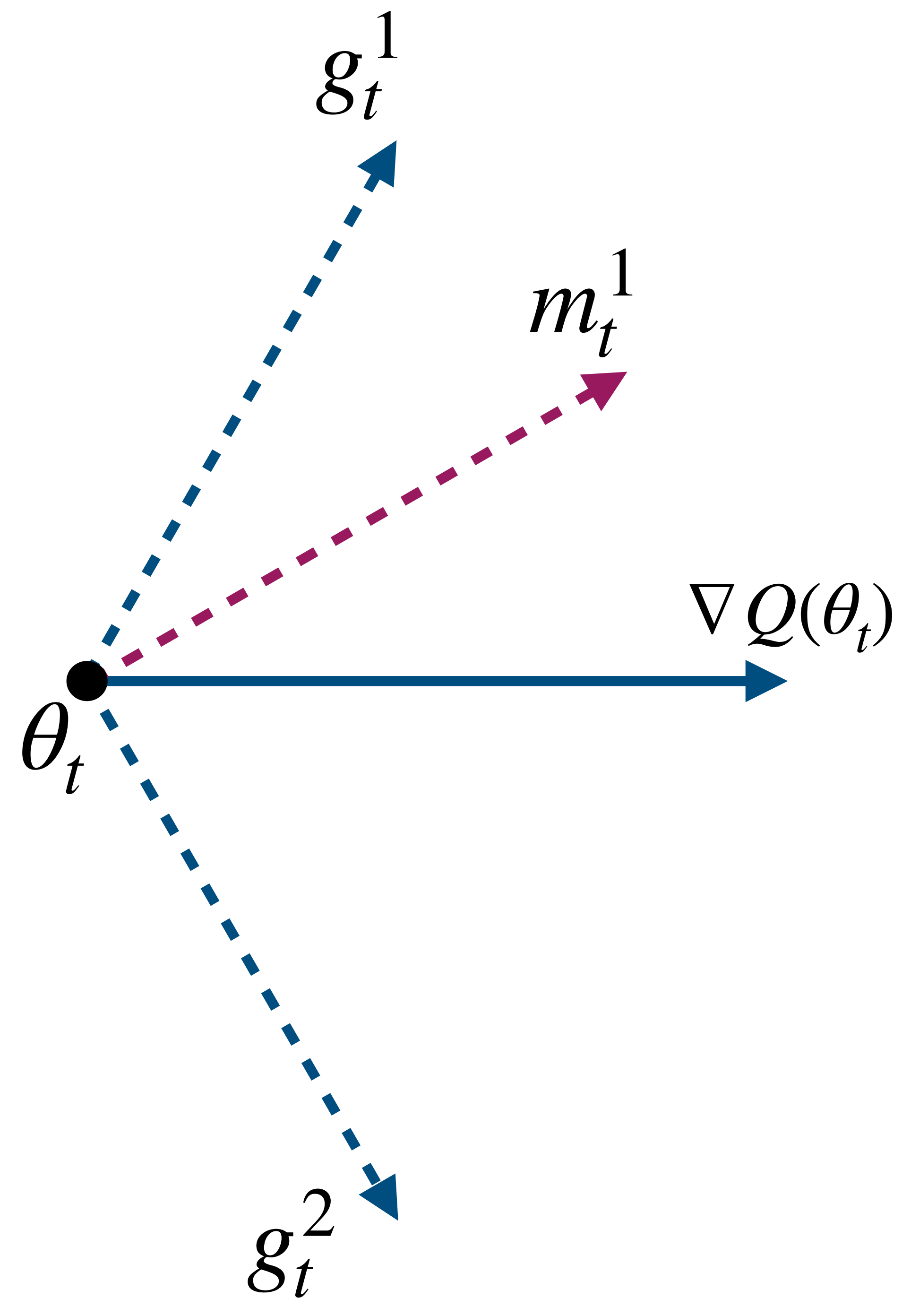
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



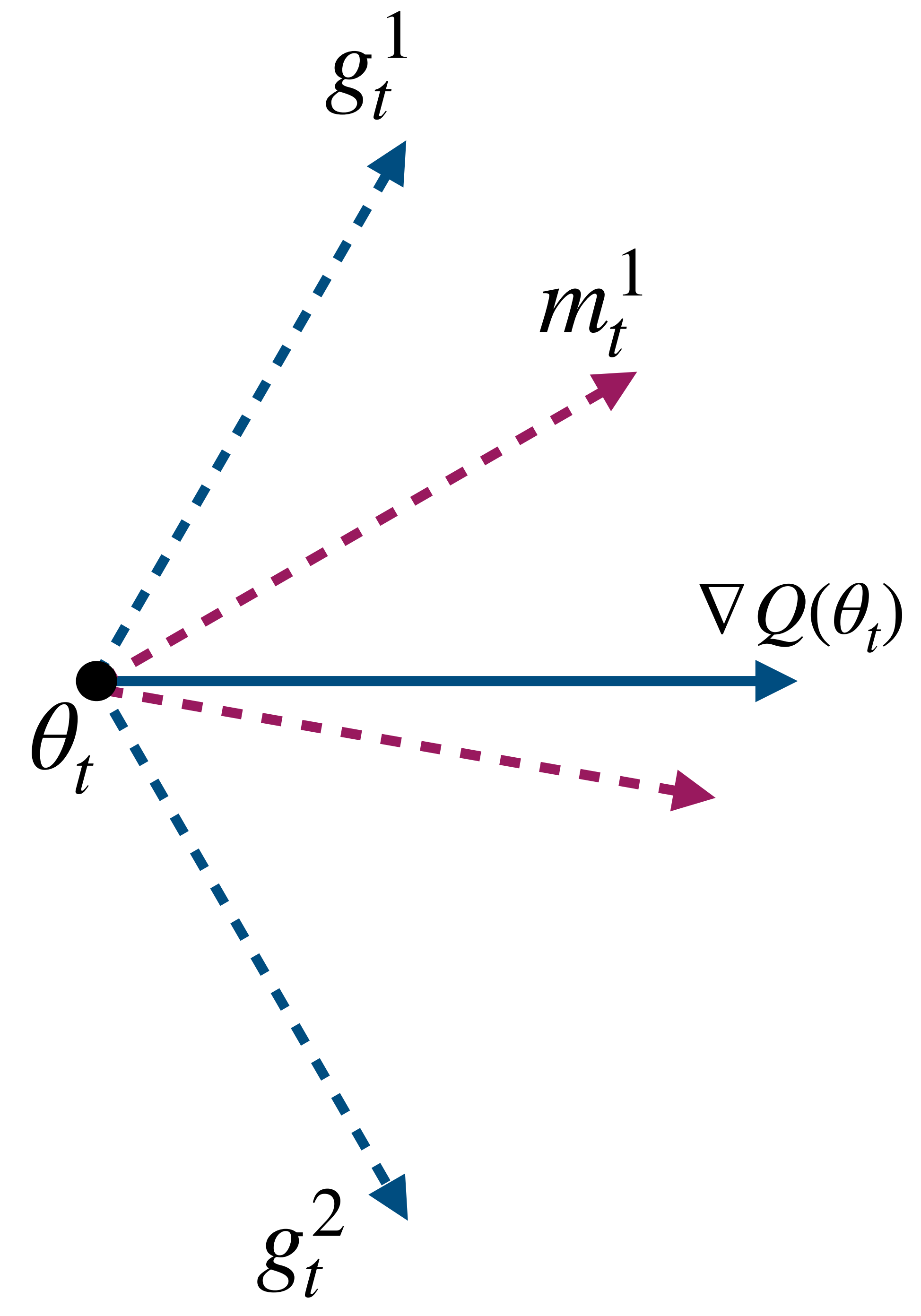
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1 - \beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



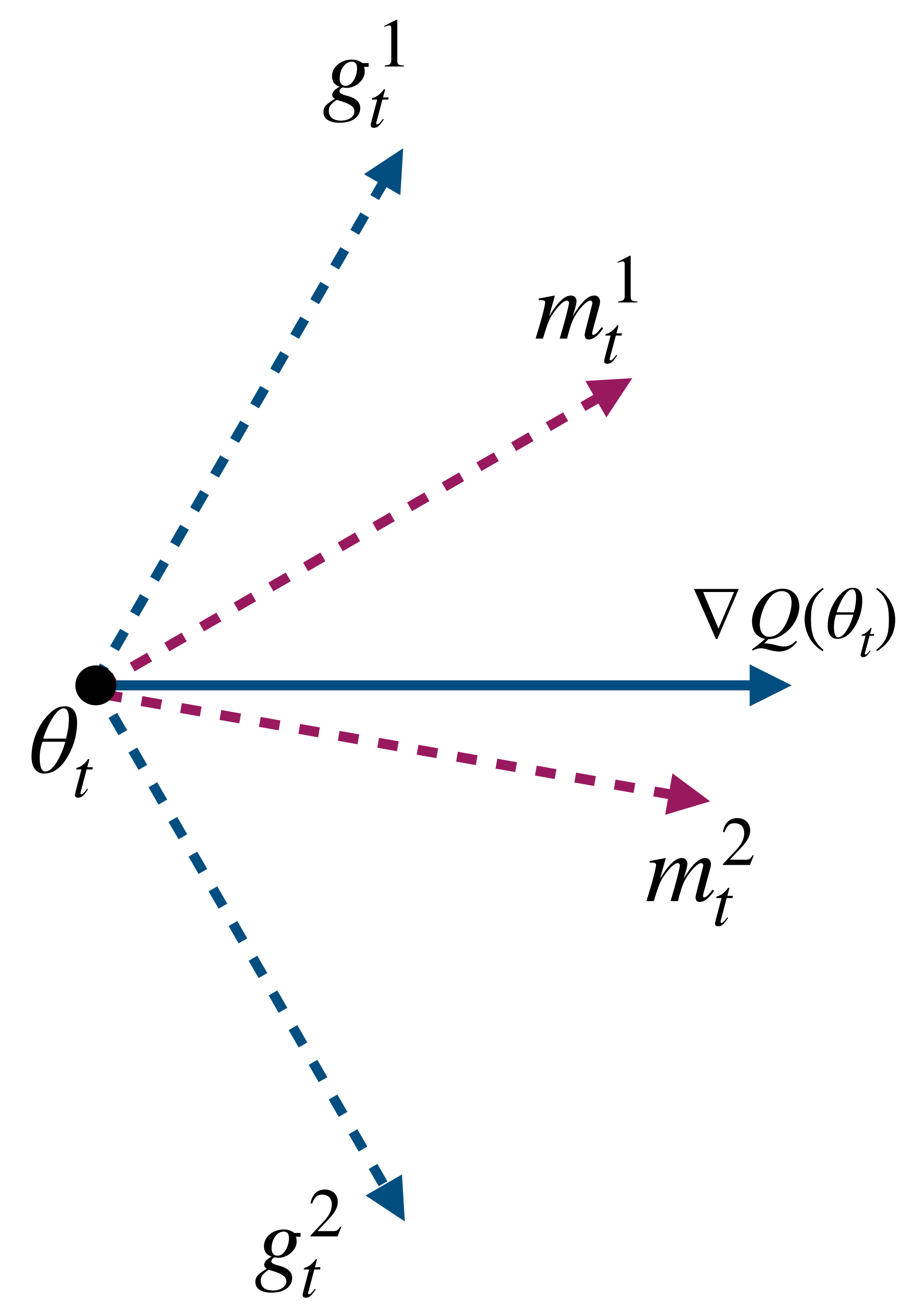
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1 - \beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



Local Gradient Momentum

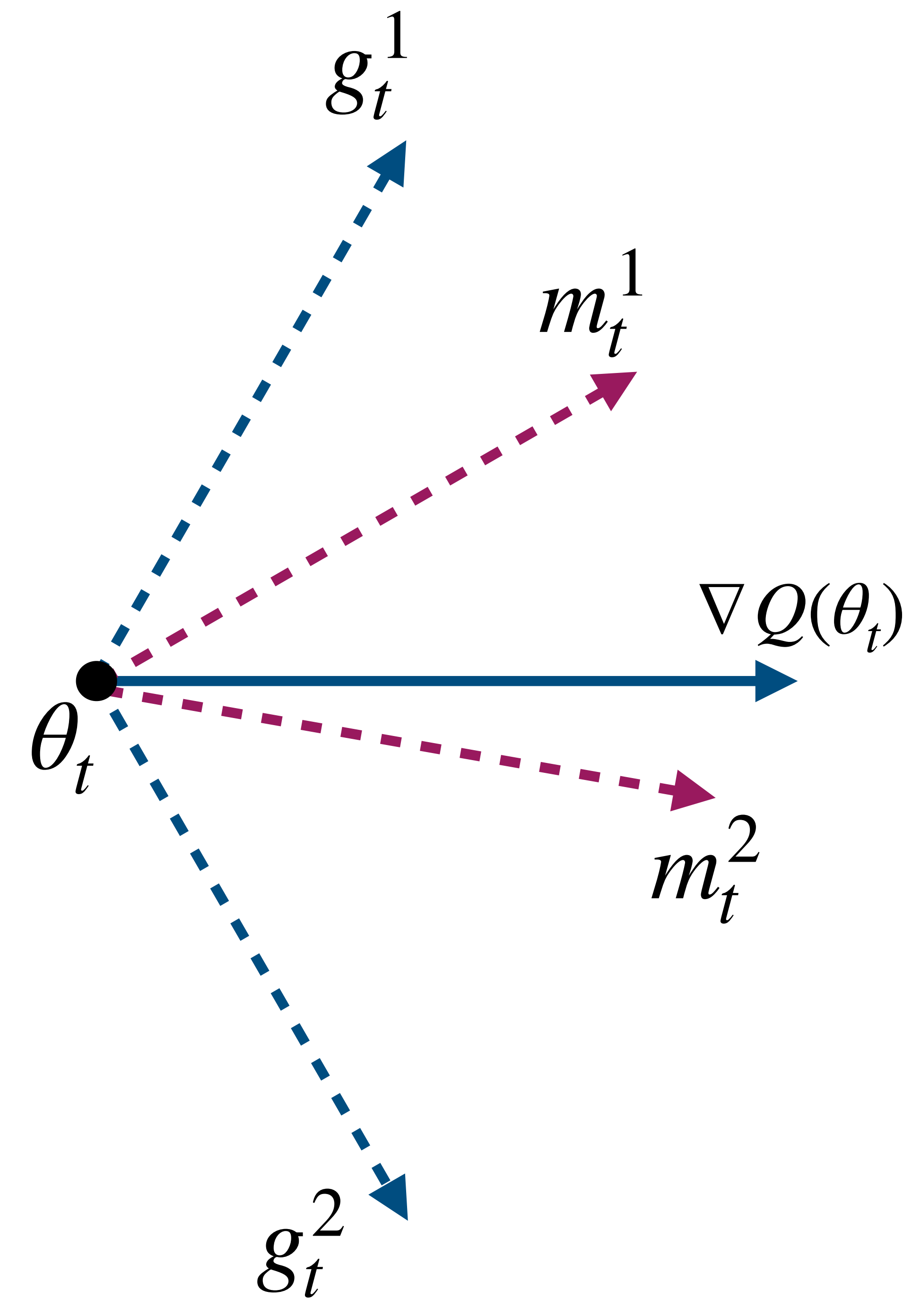
Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient

Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$



Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

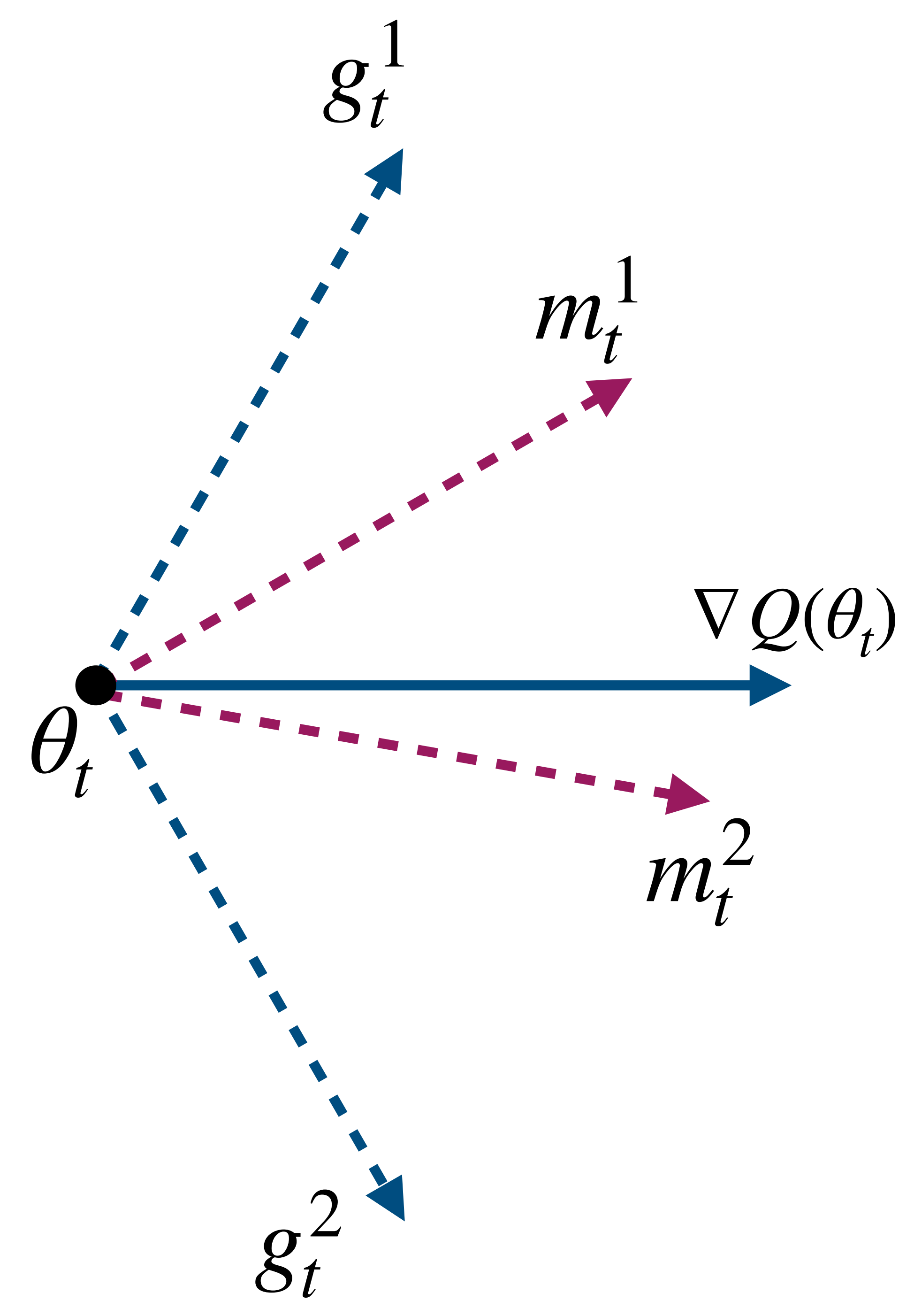
$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient

Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)



Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

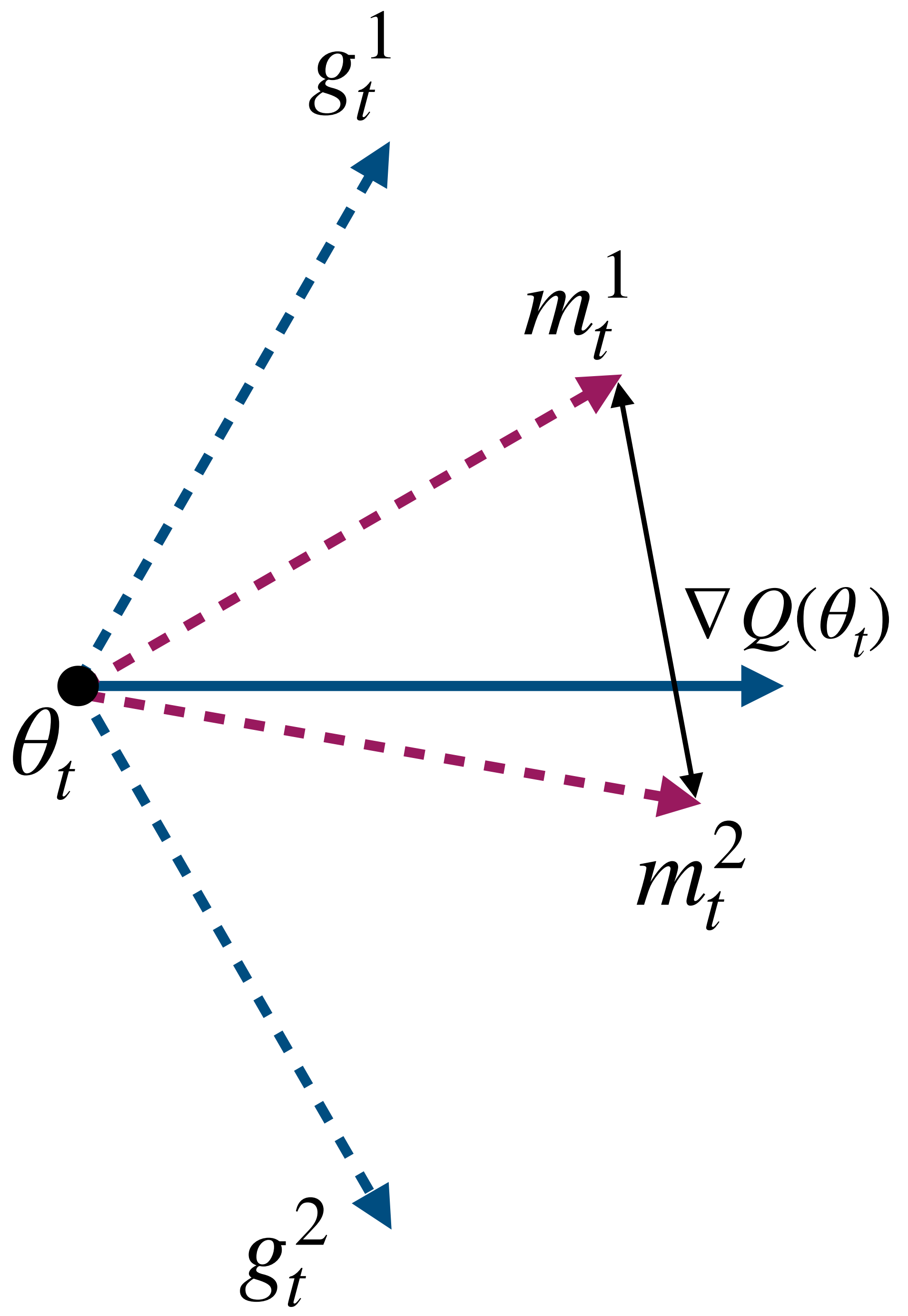
$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient

Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)



Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

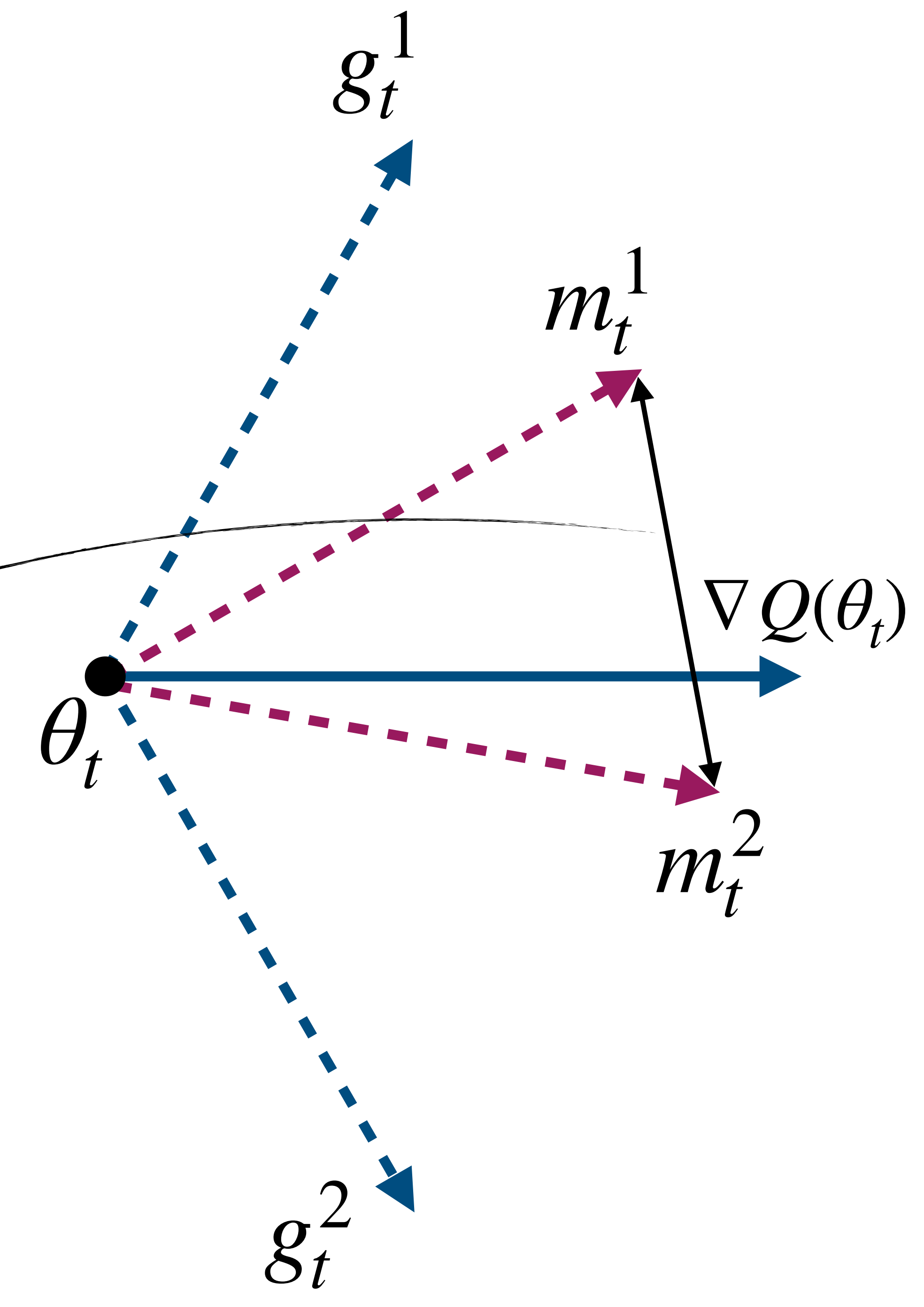
$$m_t^i = \beta m_{t-1}^i + (1 - \beta) g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient

Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)



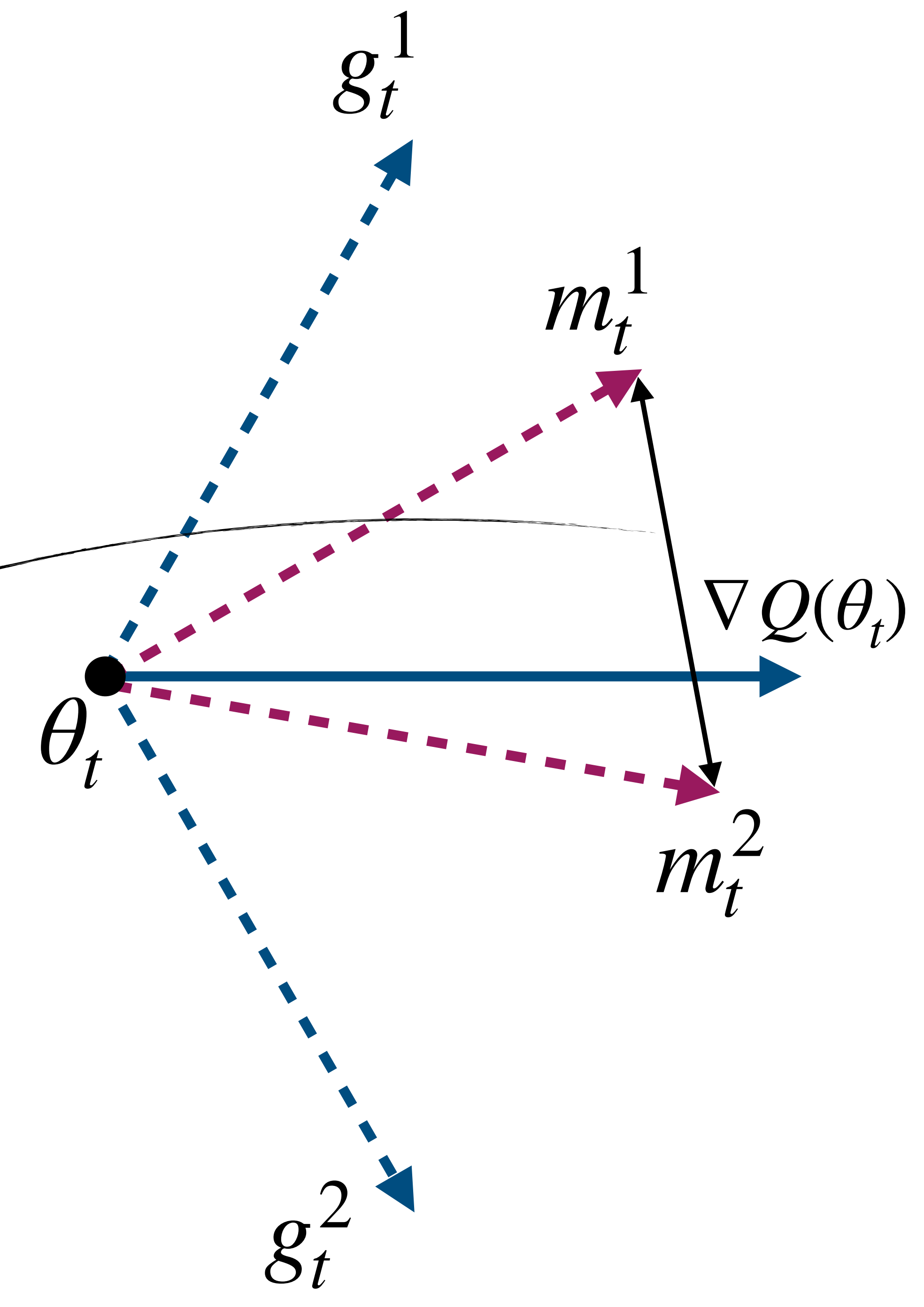
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)

The variance of the momentums can be reduced -

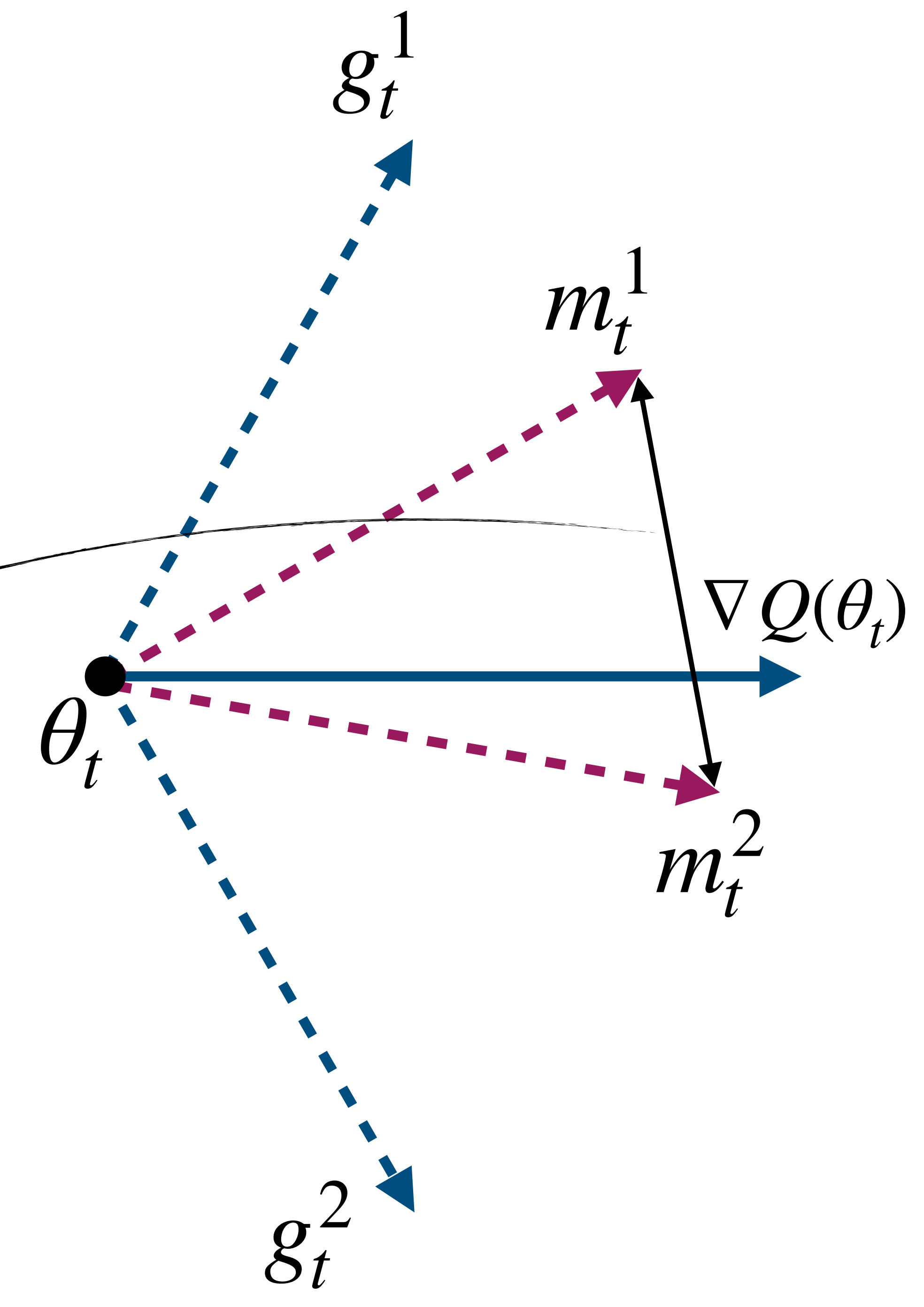
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)

The variance of the momentums can be reduced -

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq (1-\beta)\, c\sigma^2 \,, \quad \forall \;\; \textbf{honest} \;\; i, j$$

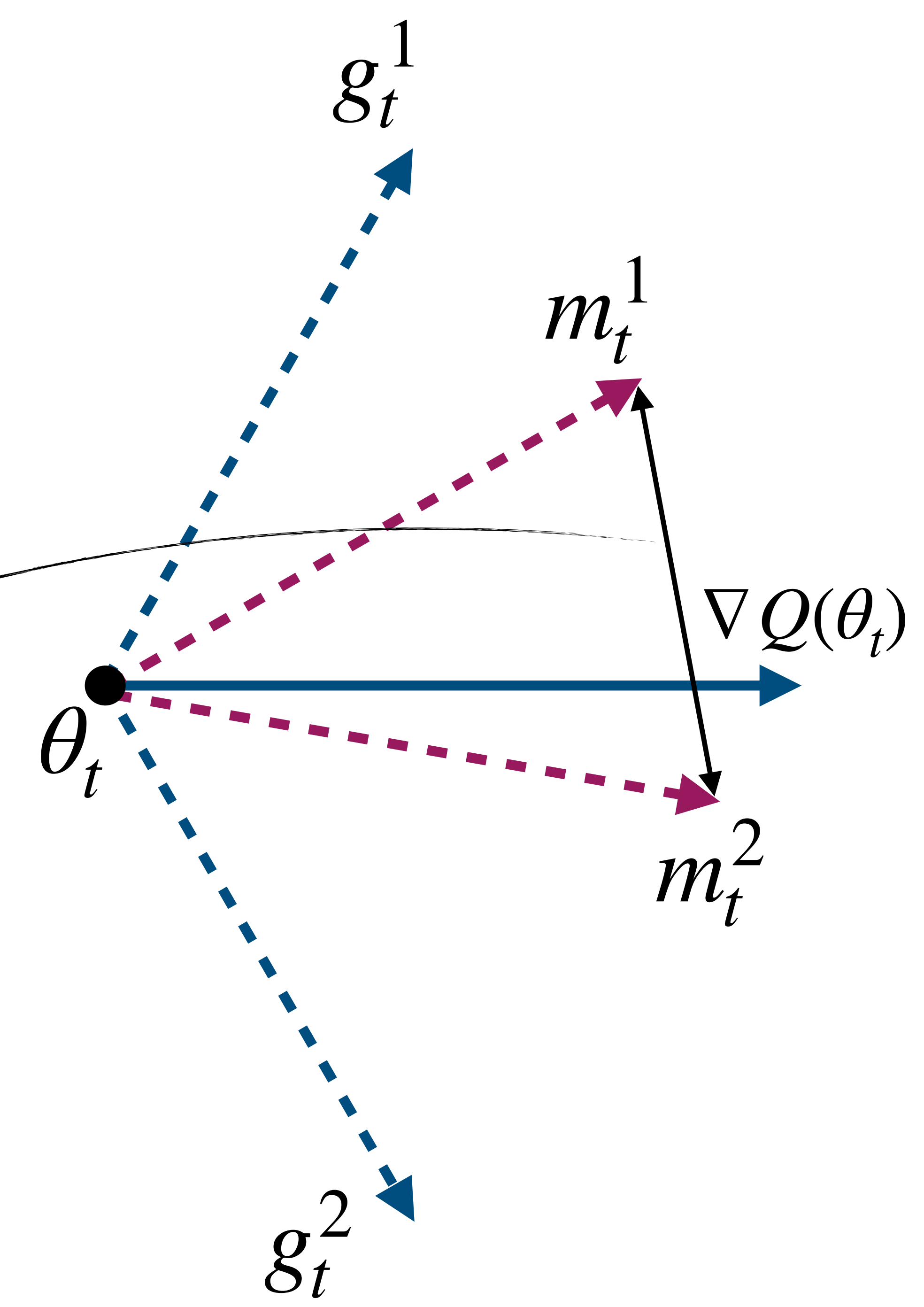
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)

The variance of the momentums can be reduced -

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq (1-\beta)\, c\sigma^2 , \quad \forall \ \textbf{honest} \ \ i, j$$

Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

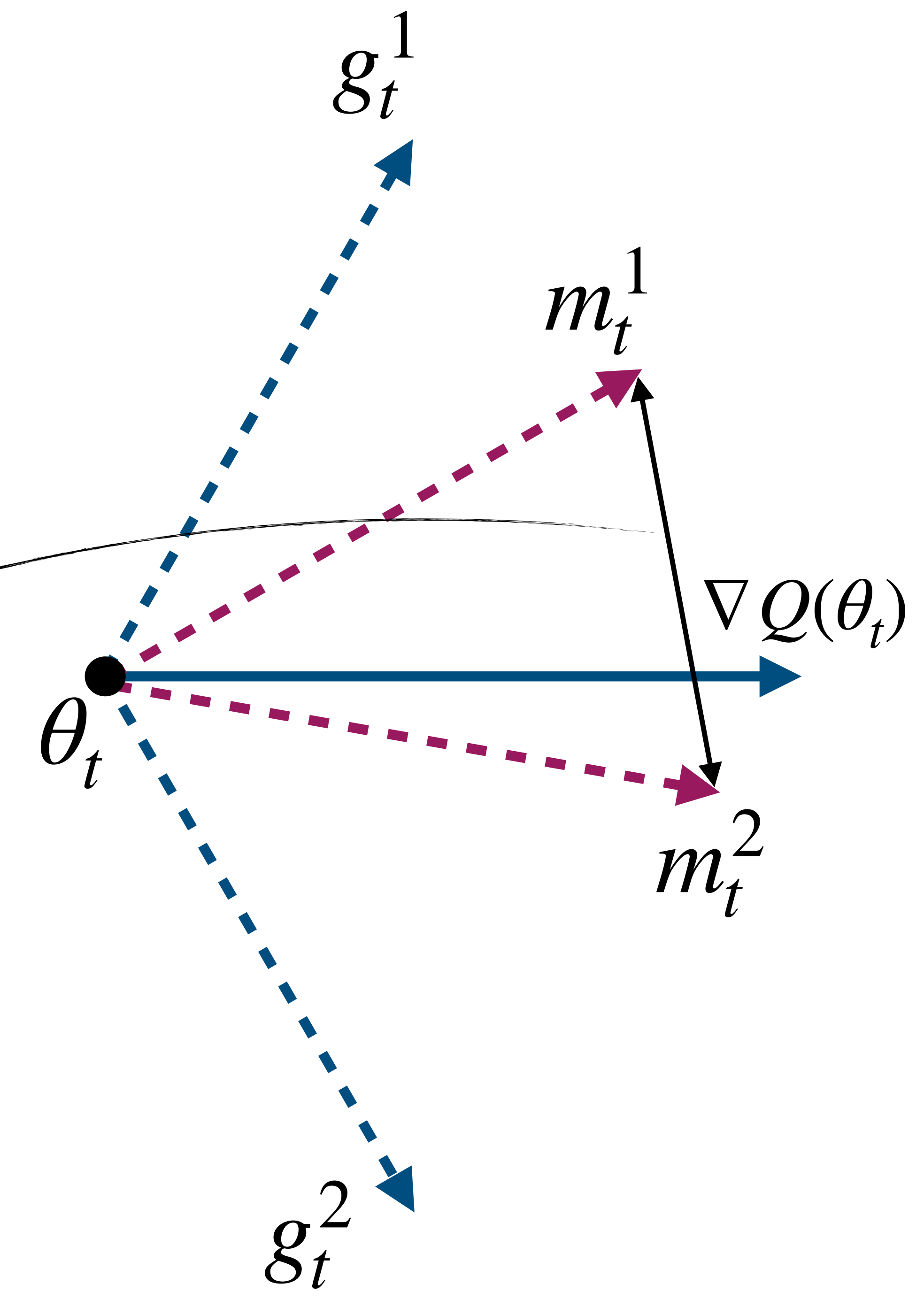
$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient

Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

(Rest remains the same)



The variance of the momentums can be reduced -

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq (1-\beta)\, c\sigma^2\,, \quad \forall \ \textbf{honest} \ \ i, j$$

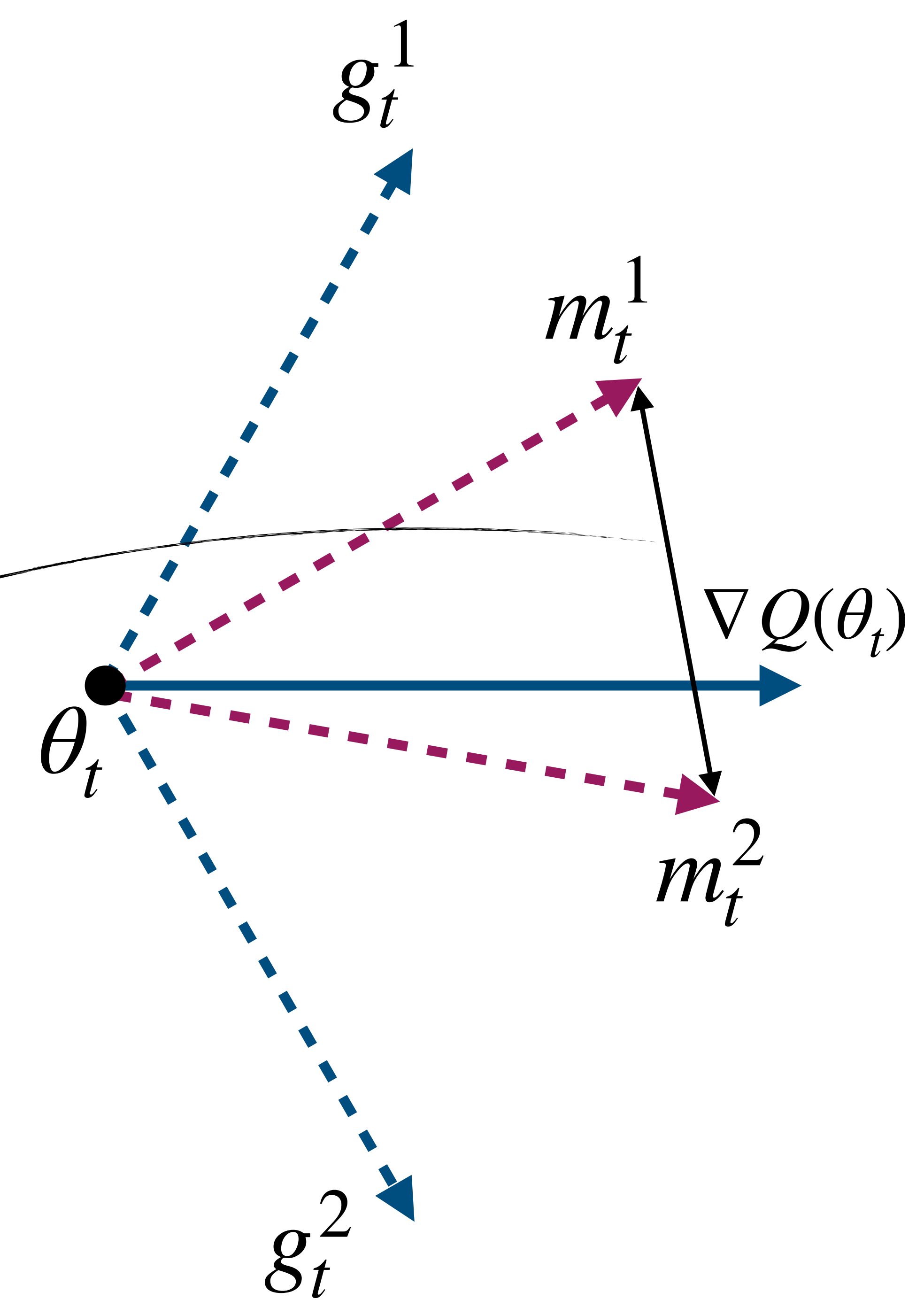
Local Gradient Momentum

Nodes compute and **send momentums** of their gradients,

$$m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$$

Past gradients

$\beta \in [0, 1)$ is referred as the momentum coefficient



Server aggregates the momentums $R_t = F\left(m_t^1, \ldots, m_t^n\right)$

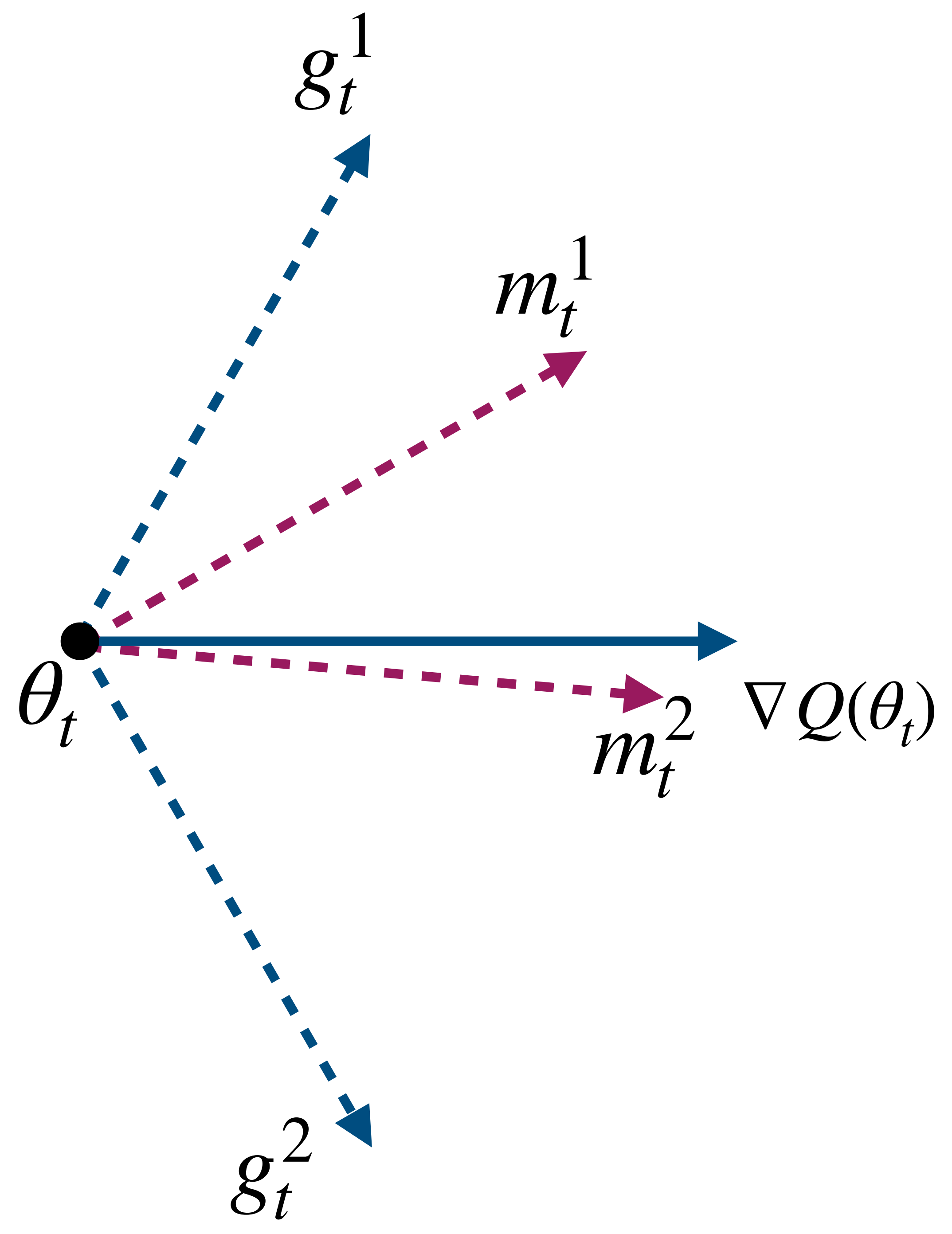
(Rest remains the same)

The variance of the momentums can be reduced -

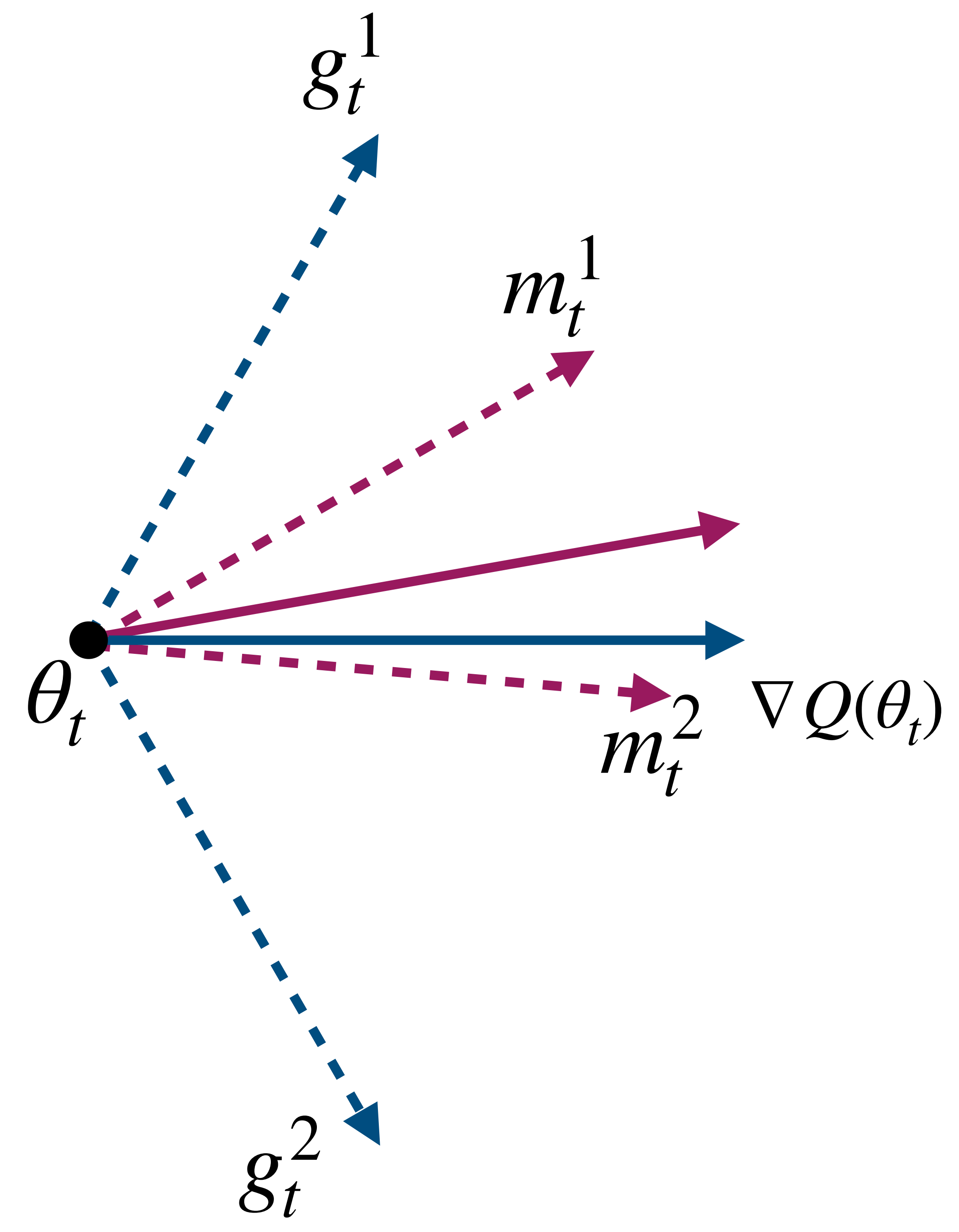
$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq (1-\beta)\, c\sigma^2 \;, \quad \forall \;\textbf{honest}\;\; i, j$$

Ideally large

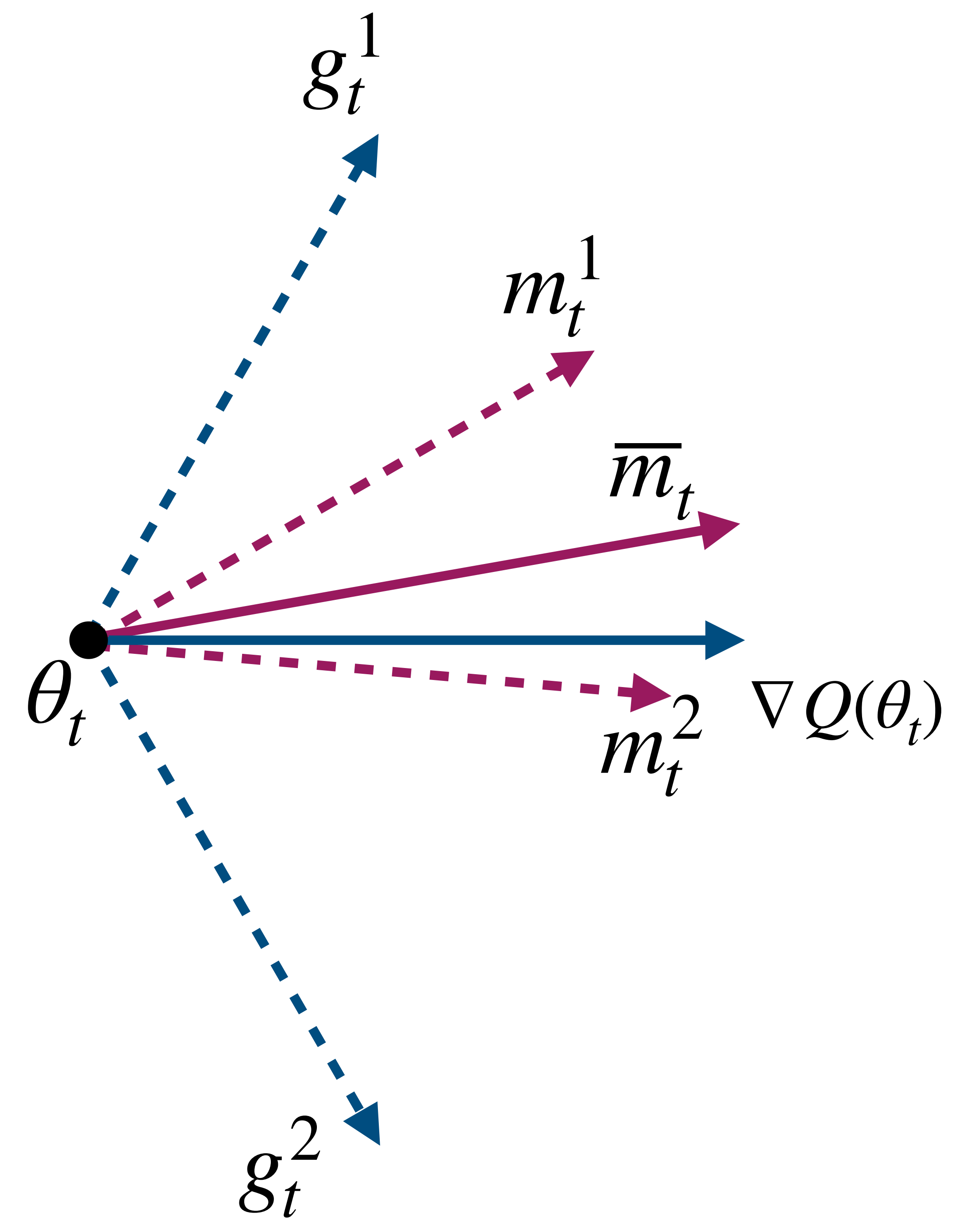
Caveat of Momentum



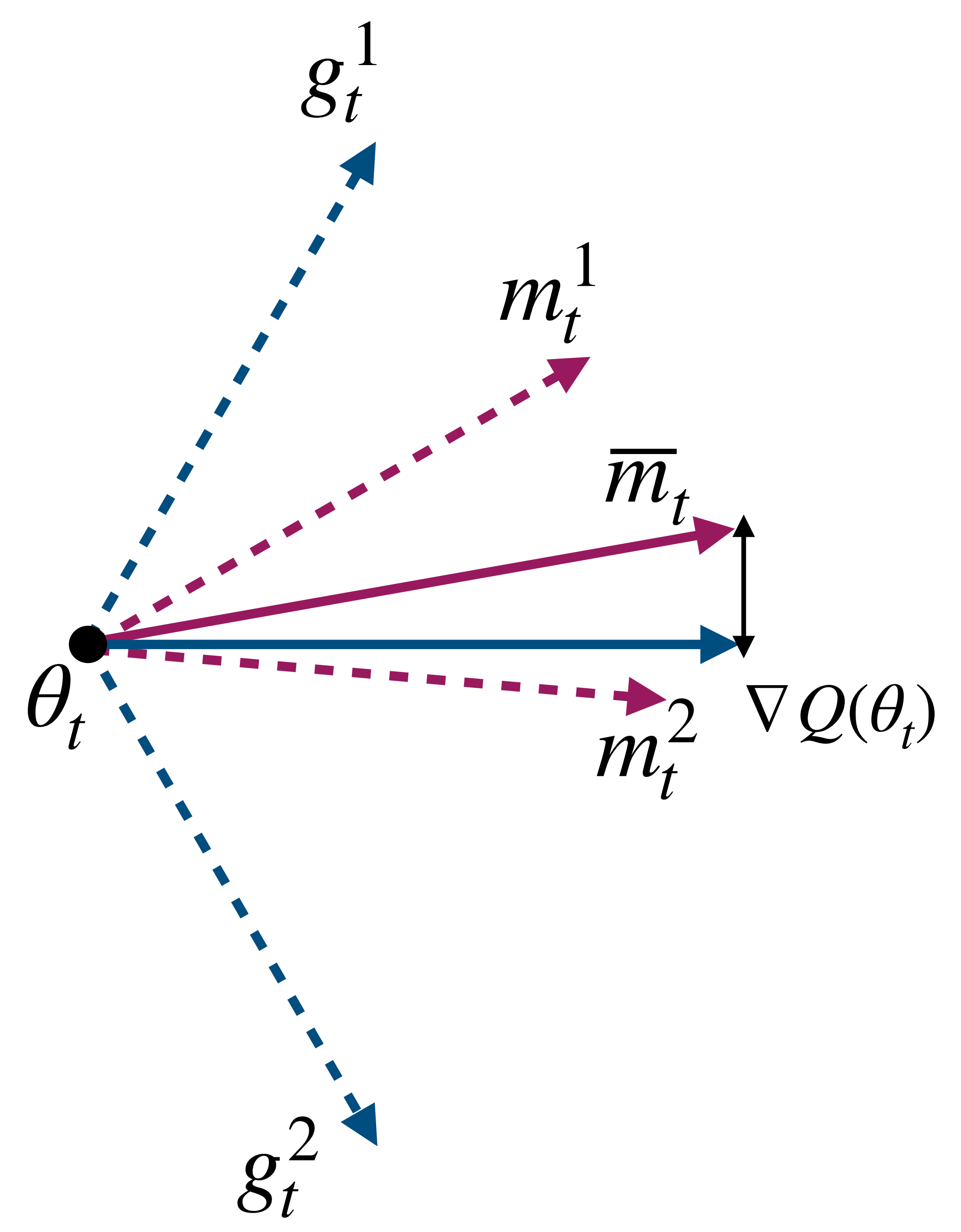
Caveat of Momentum



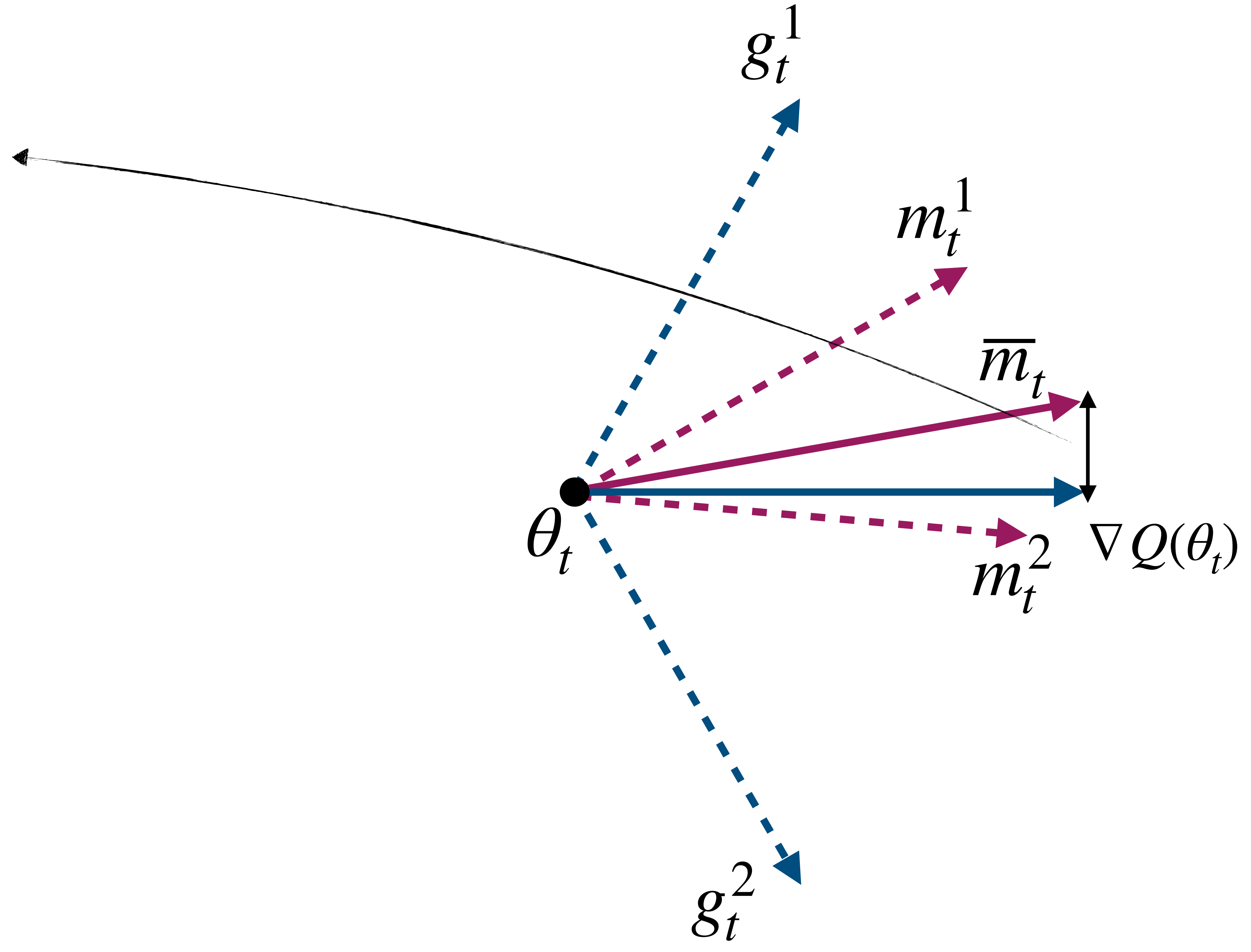
Caveat of Momentum



Caveat of Momentum

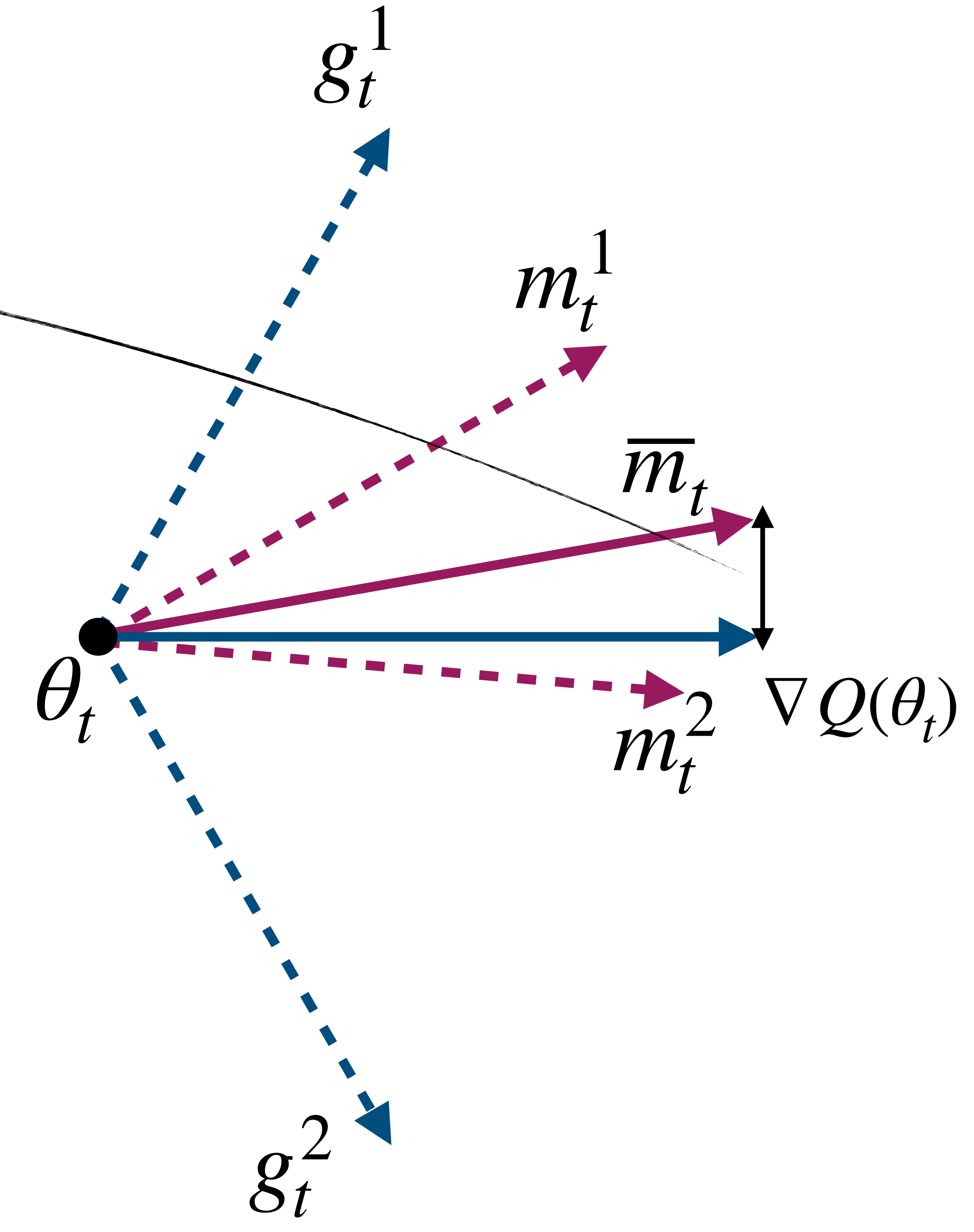


Caveat of Momentum



Caveat of Momentum

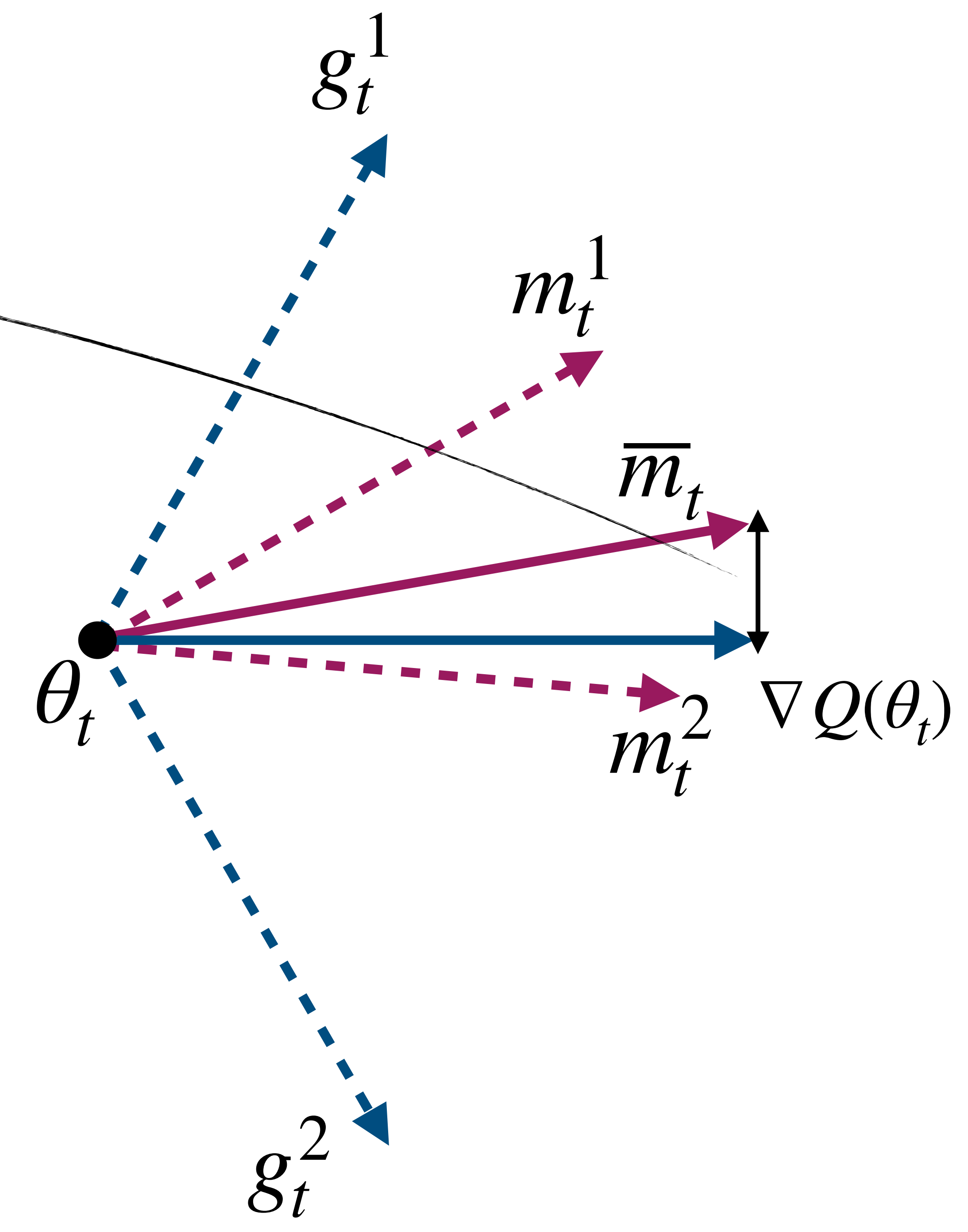
Local momentums are no longer unbiased gradient estimators



Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

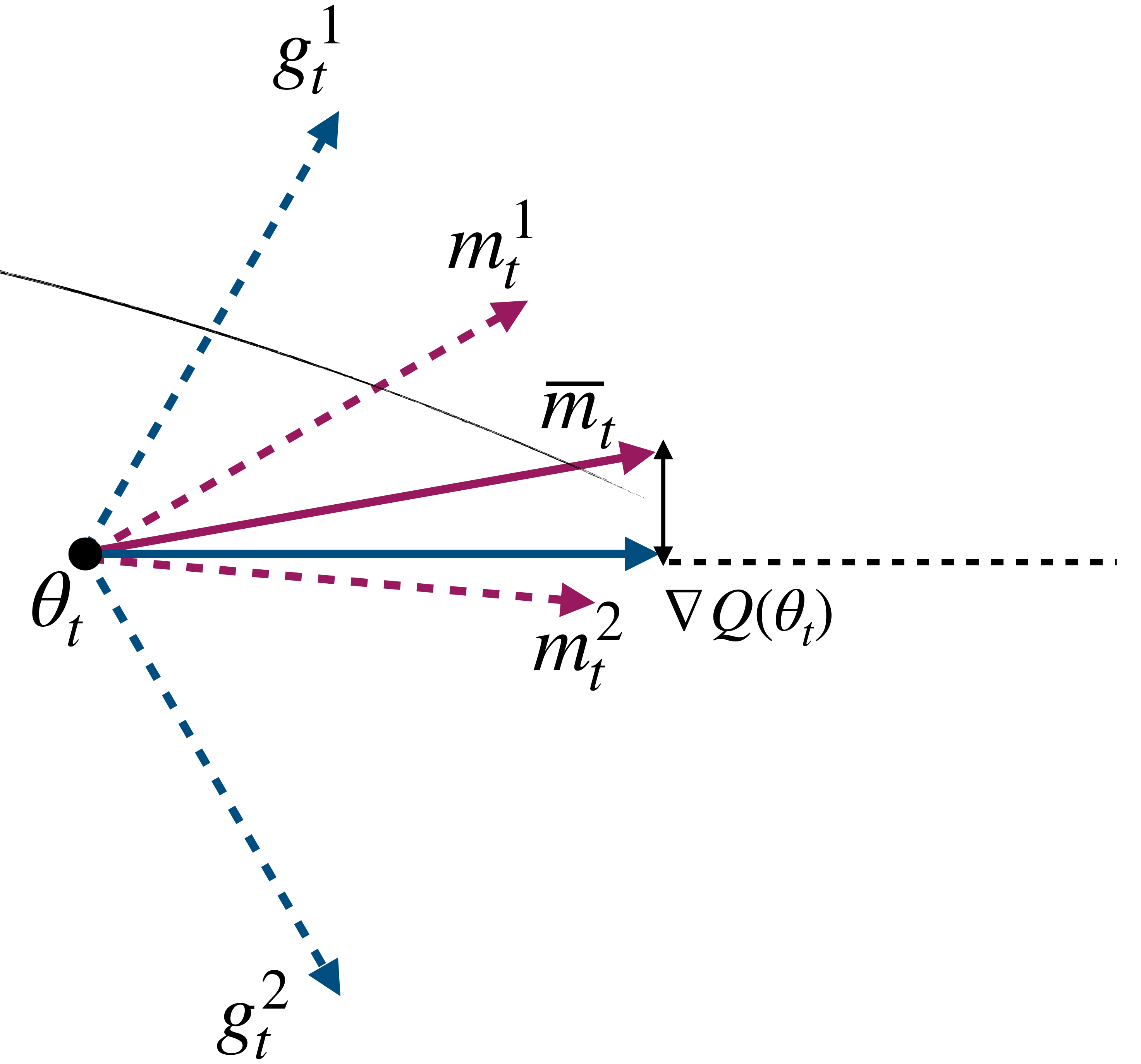
$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2\,\mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2\frac{\sigma^2}{n-f} + \epsilon(t)$$



Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

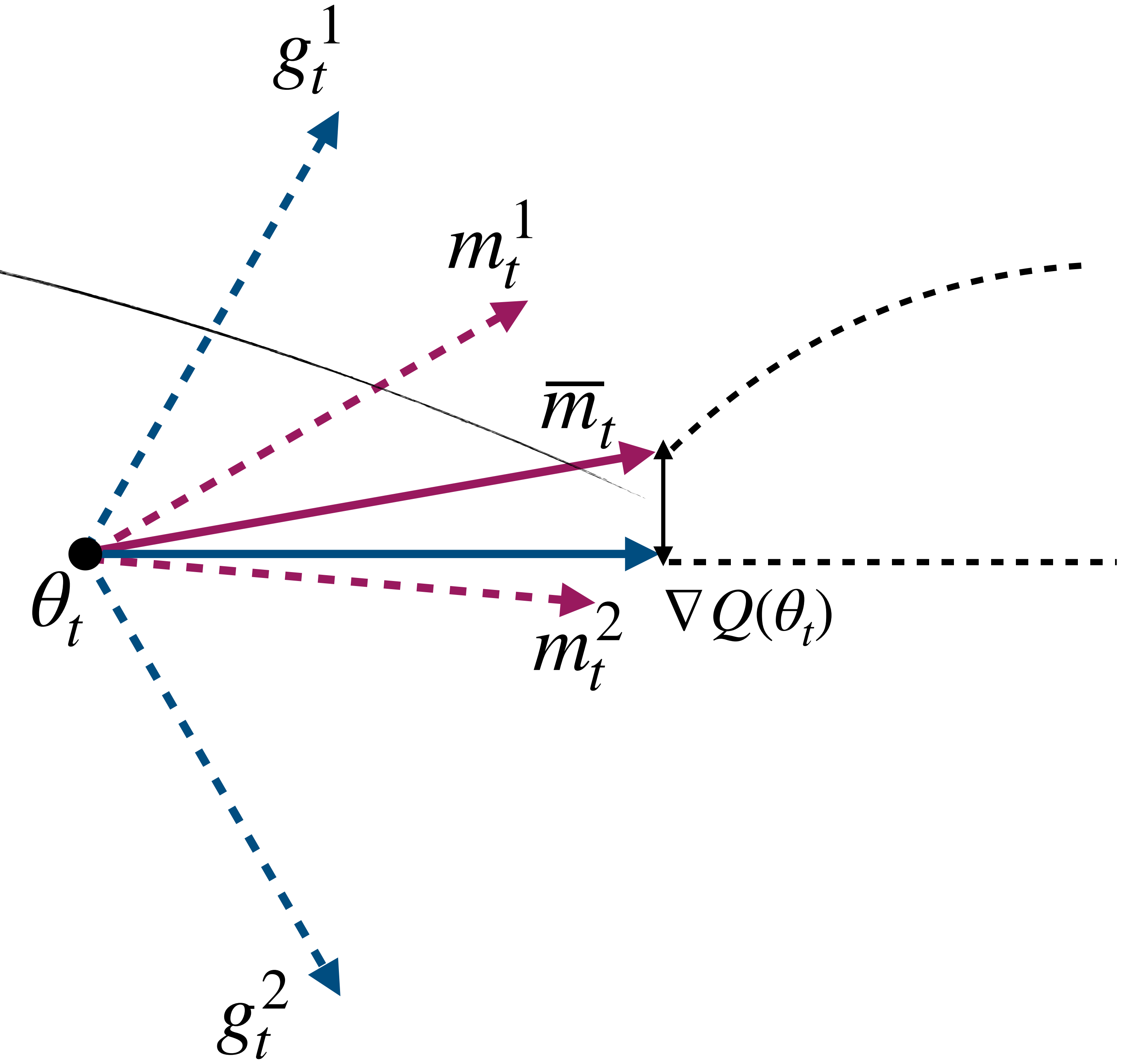
$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \, \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$



Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

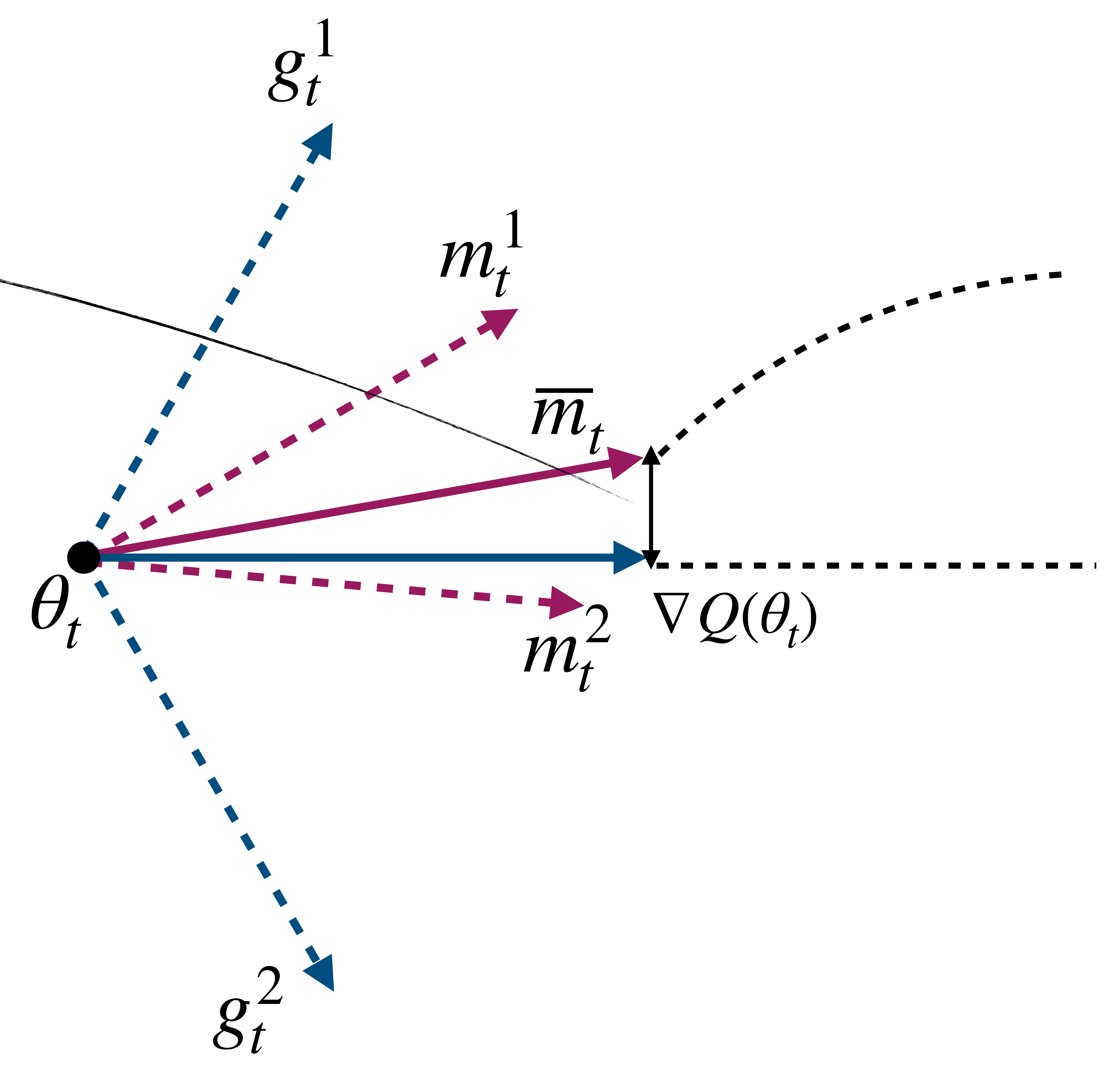
$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2\,\mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2\frac{\sigma^2}{n-f} + \epsilon(t)$$



Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

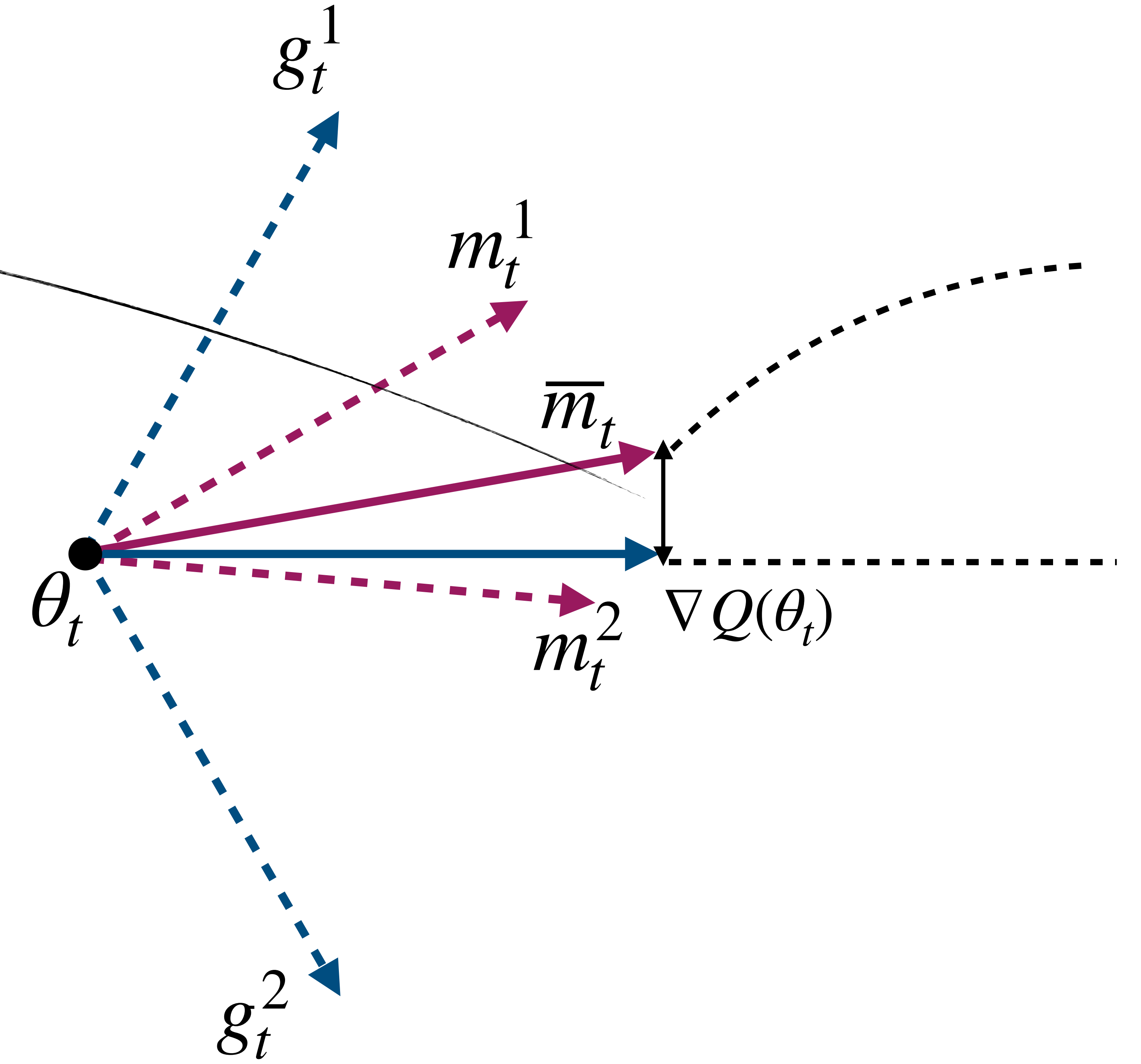
$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$



Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$

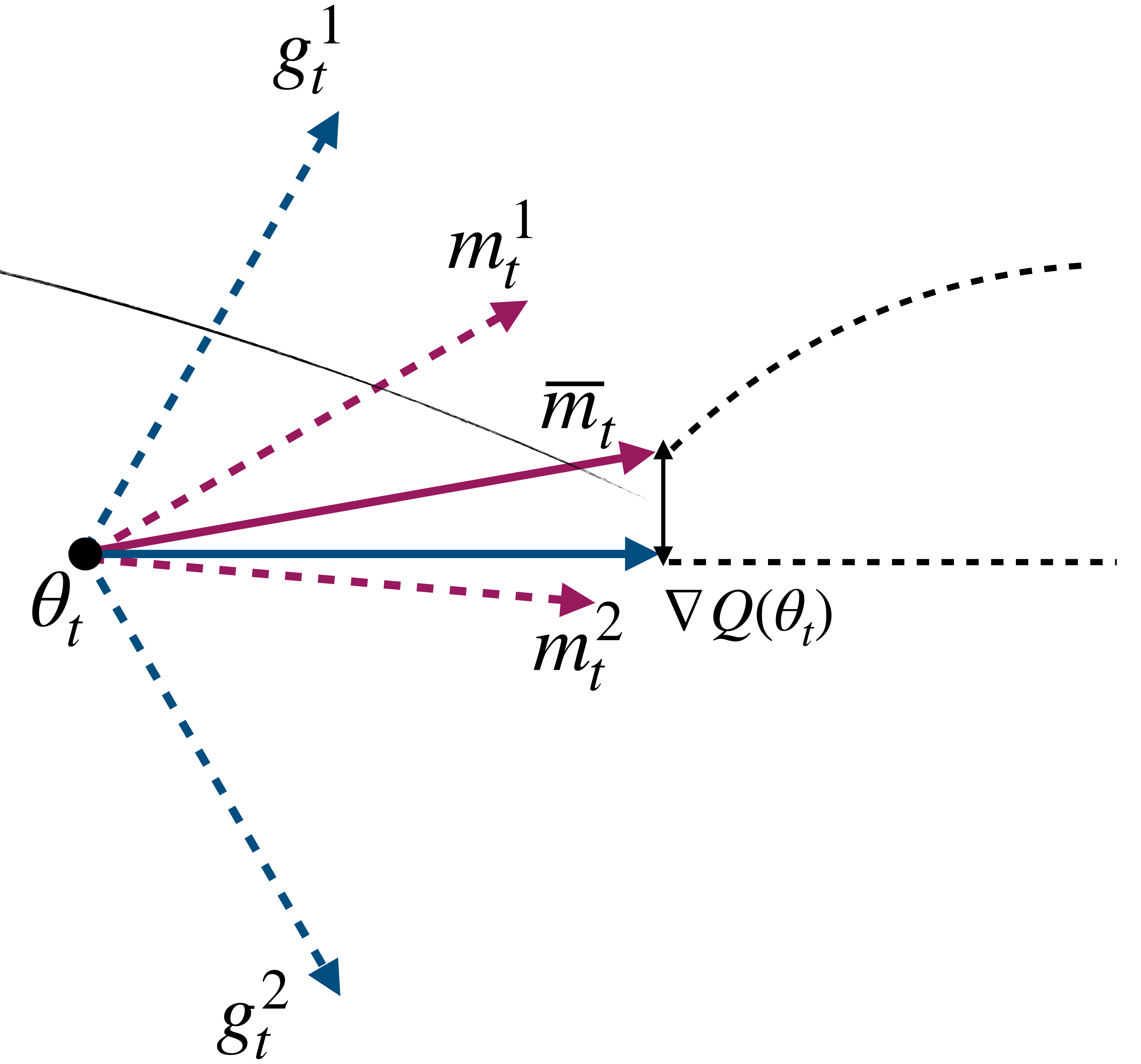


Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$

Ideally small

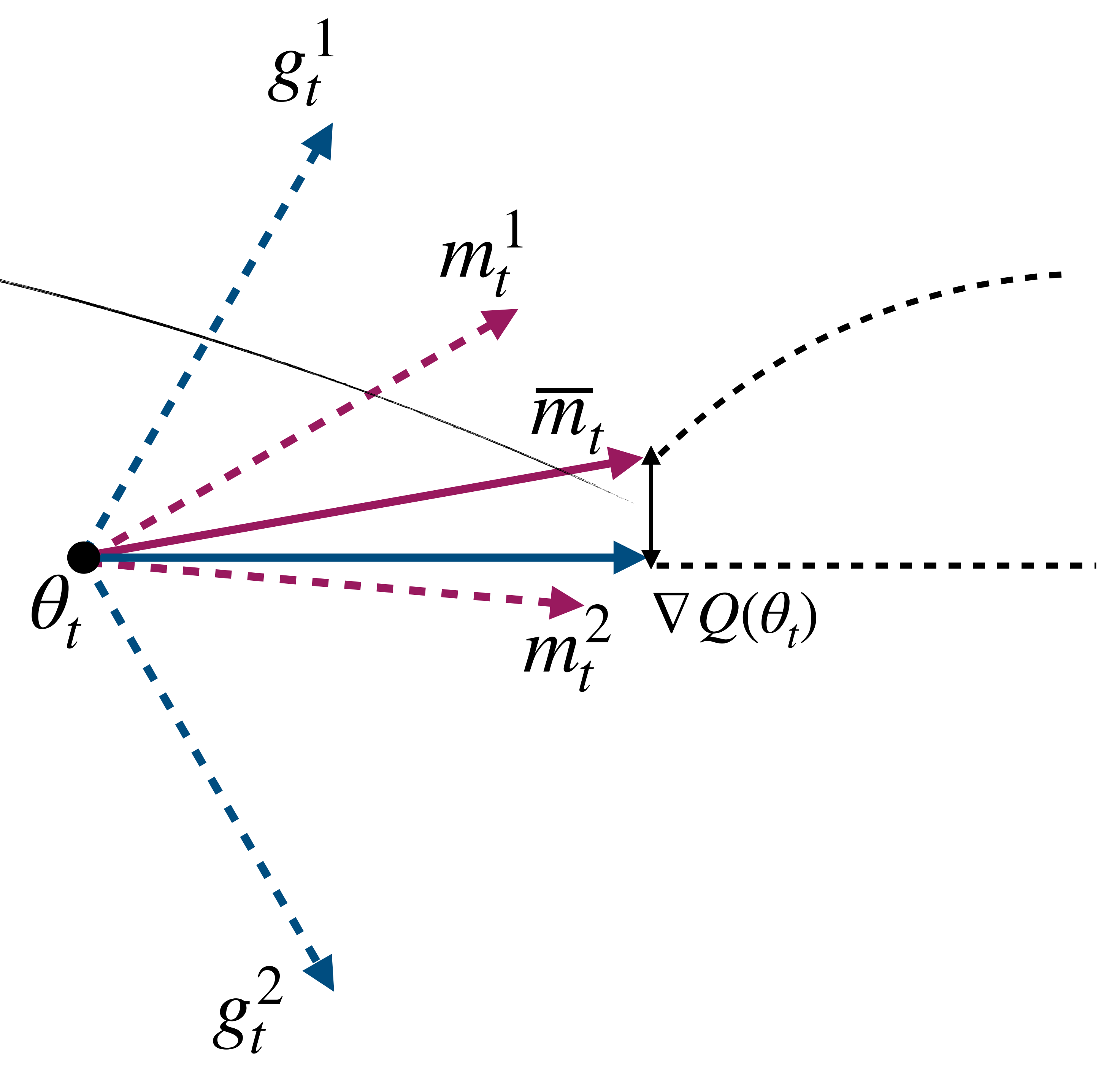


Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$

Ideally small



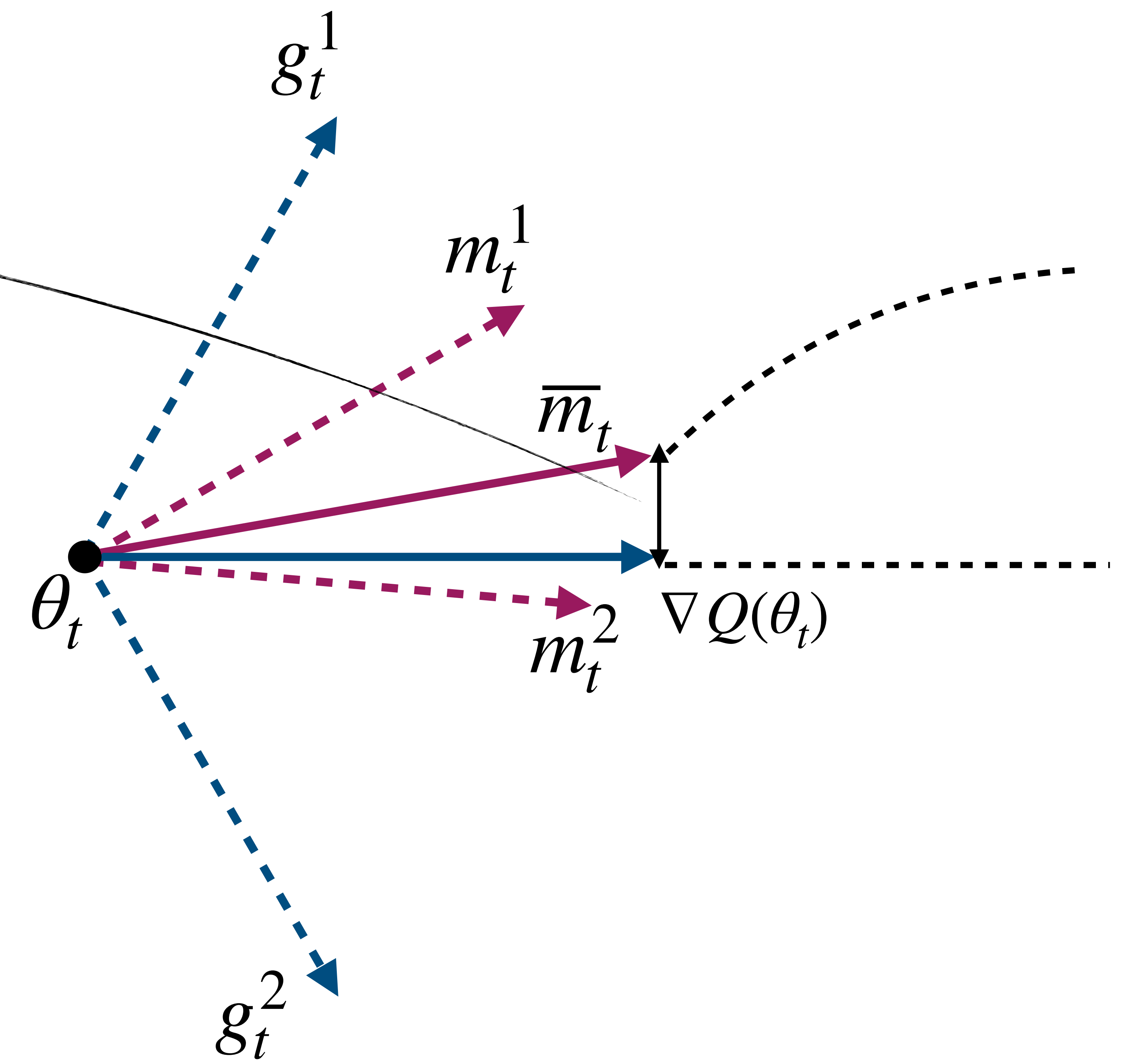
!! Good News !!

Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$

Ideally small



!! Good News !!

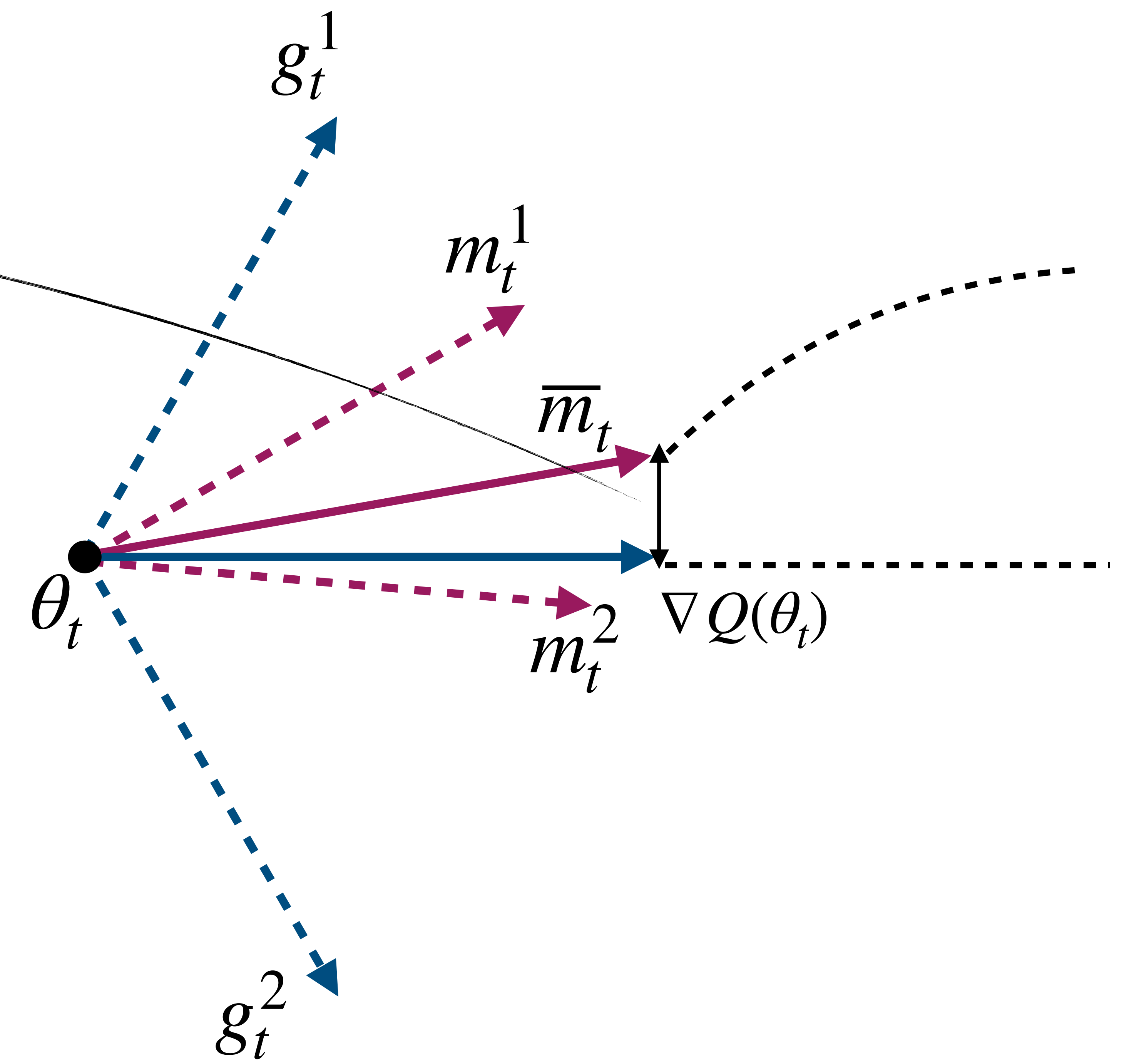
IF the **momentum** coefficient is **chosen wisely**

Caveat of Momentum

Local momentums are no longer unbiased gradient estimators

$$\mathbb{E}\left[\|\overline{m}_t - \nabla Q\left(\theta_t\right)\|^2\right] \leq \beta^2 \mathbb{E}\left[\|\overline{m}_{t-1} - \nabla Q\left(\theta_{t-1}\right)\|^2\right] + (1-\beta)^2 \frac{\sigma^2}{n-f} + \epsilon(t)$$

Ideally small



!! Good News !!

IF the **momentum** coefficient is **chosen wisely**

THEN the **good** (reduced variance)
Outweighs the bad (biased gradient estimation)

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \bar{x}_S\| \leq \lambda \max_{i,j \in S}\|x_i - x_j\|$$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)***-resilient averaging (RESA)*** -

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)***-resilient averaging (RESA)*** -

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)***-resilient averaging (RESA) -***

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\| F(x_1, \ldots, x_n) - \bar{x}_S \| \leq \lambda \max_{i,j \in S} \| x_i - x_j \|$$

Avg. of $\{x_i\}_{i \in S}$

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

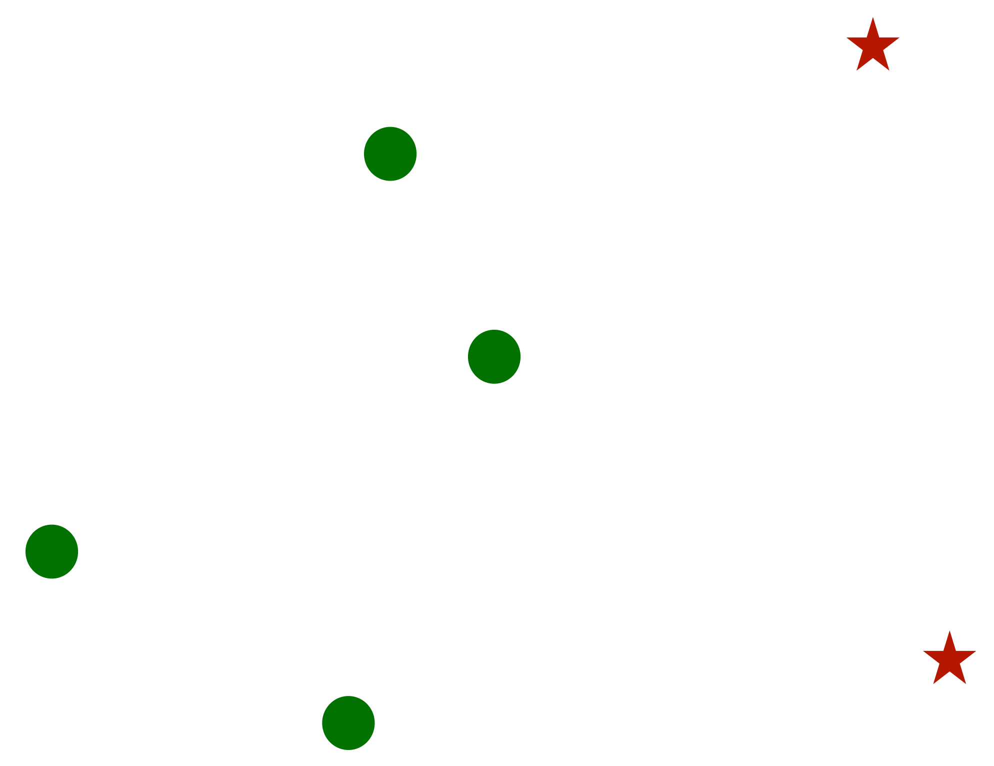
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

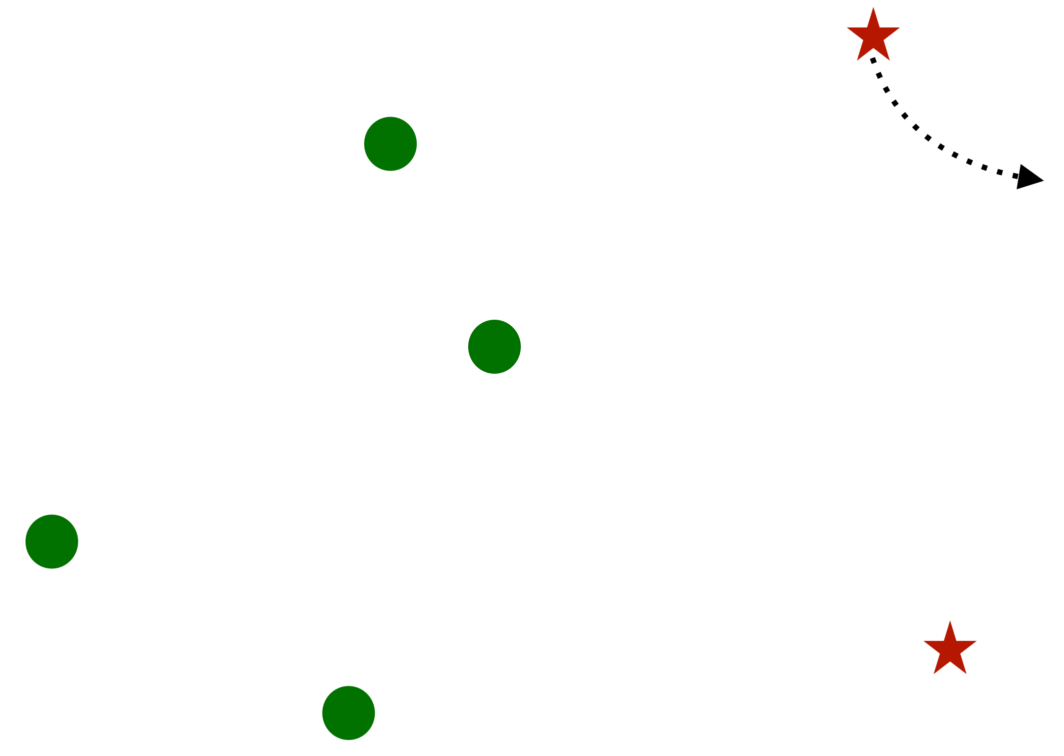
$\exists \ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall \ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\| F\left(x_1, \ldots, x_n\right) - \overline{x}_S \| \leq \lambda \max_{i,j \in S} \| x_i - x_j \|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

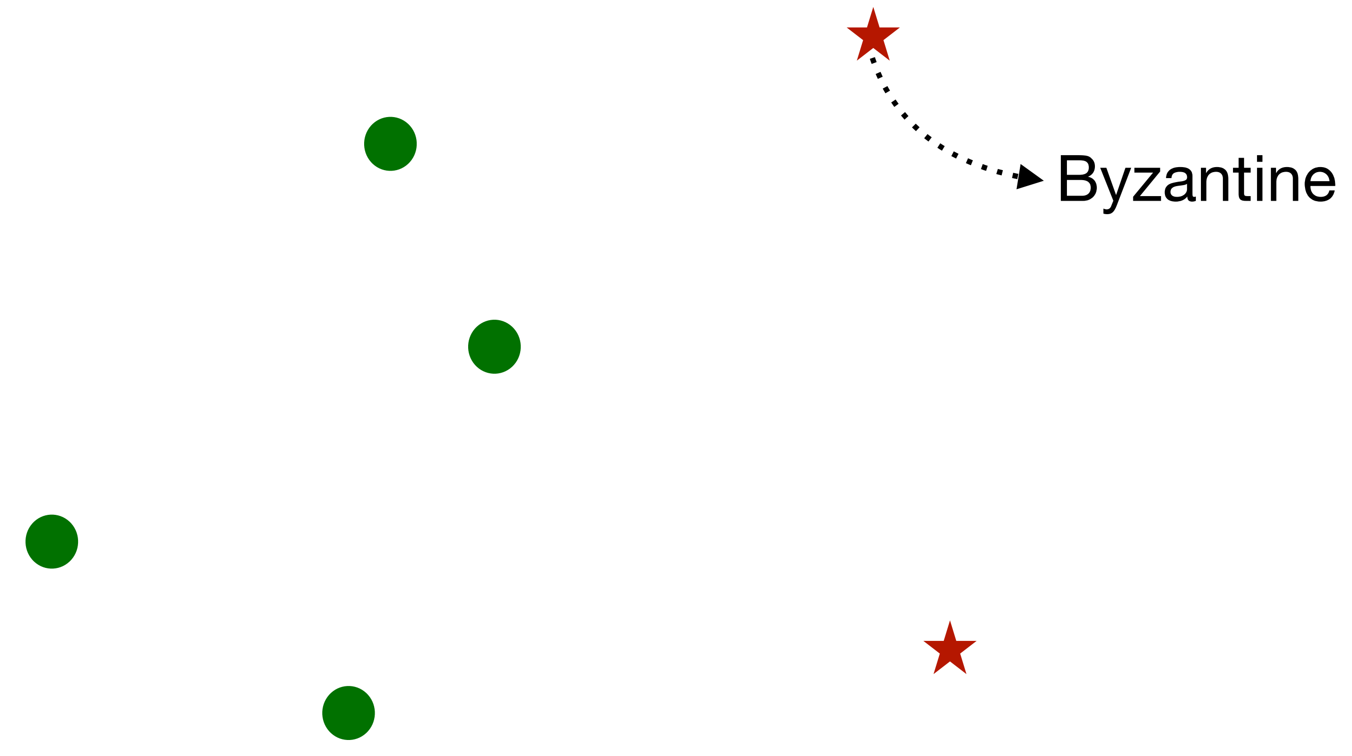
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n-f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

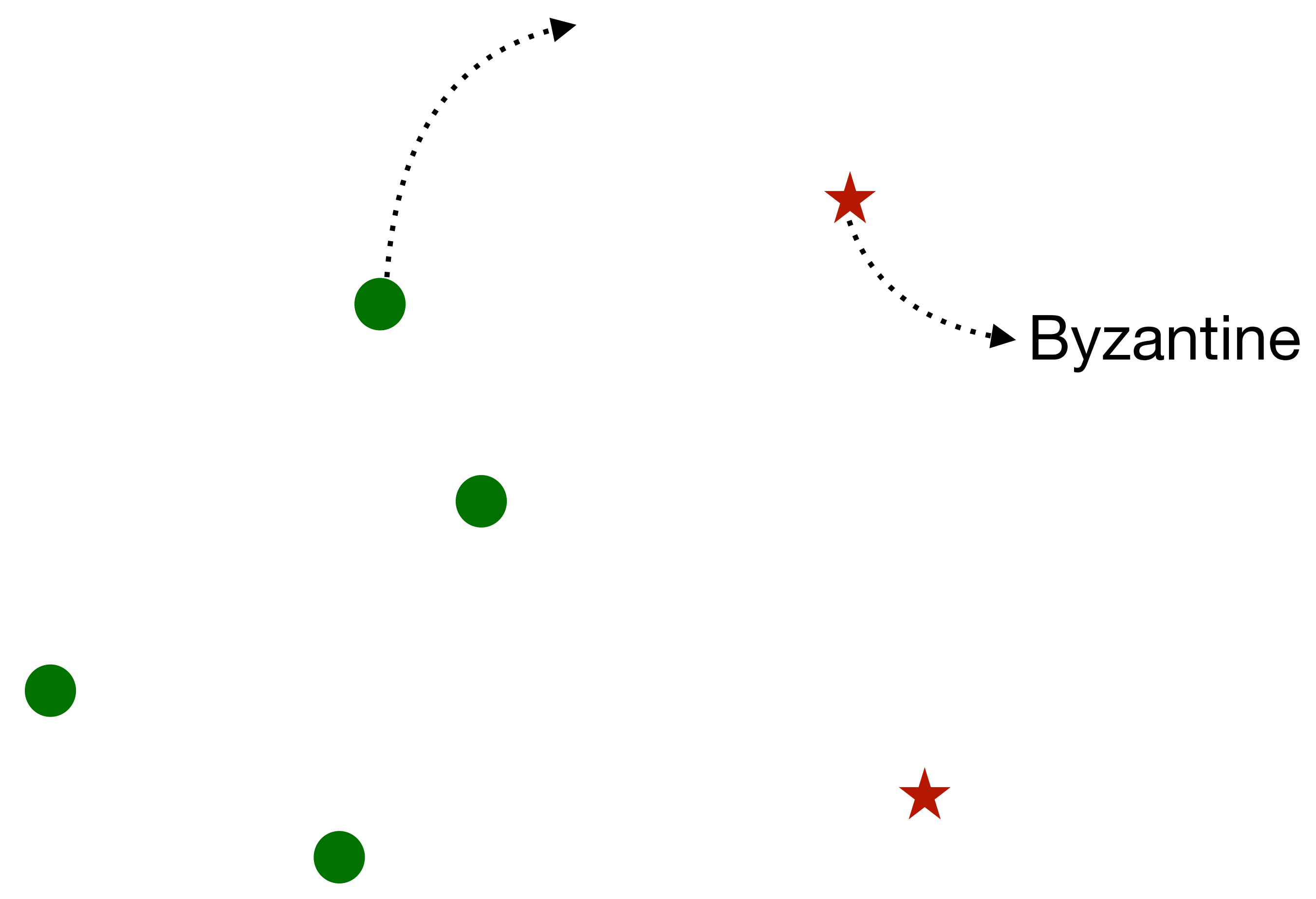
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

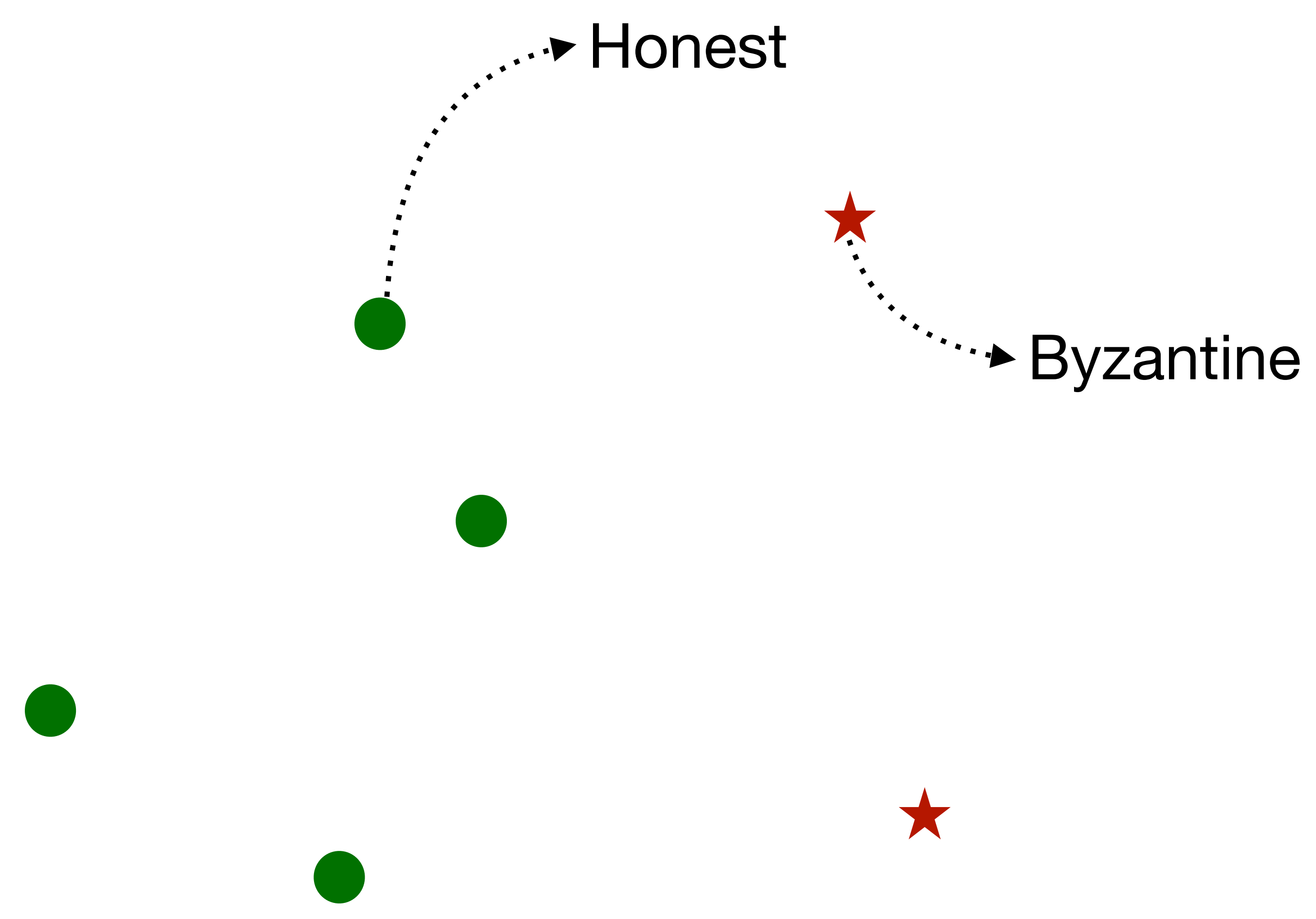
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

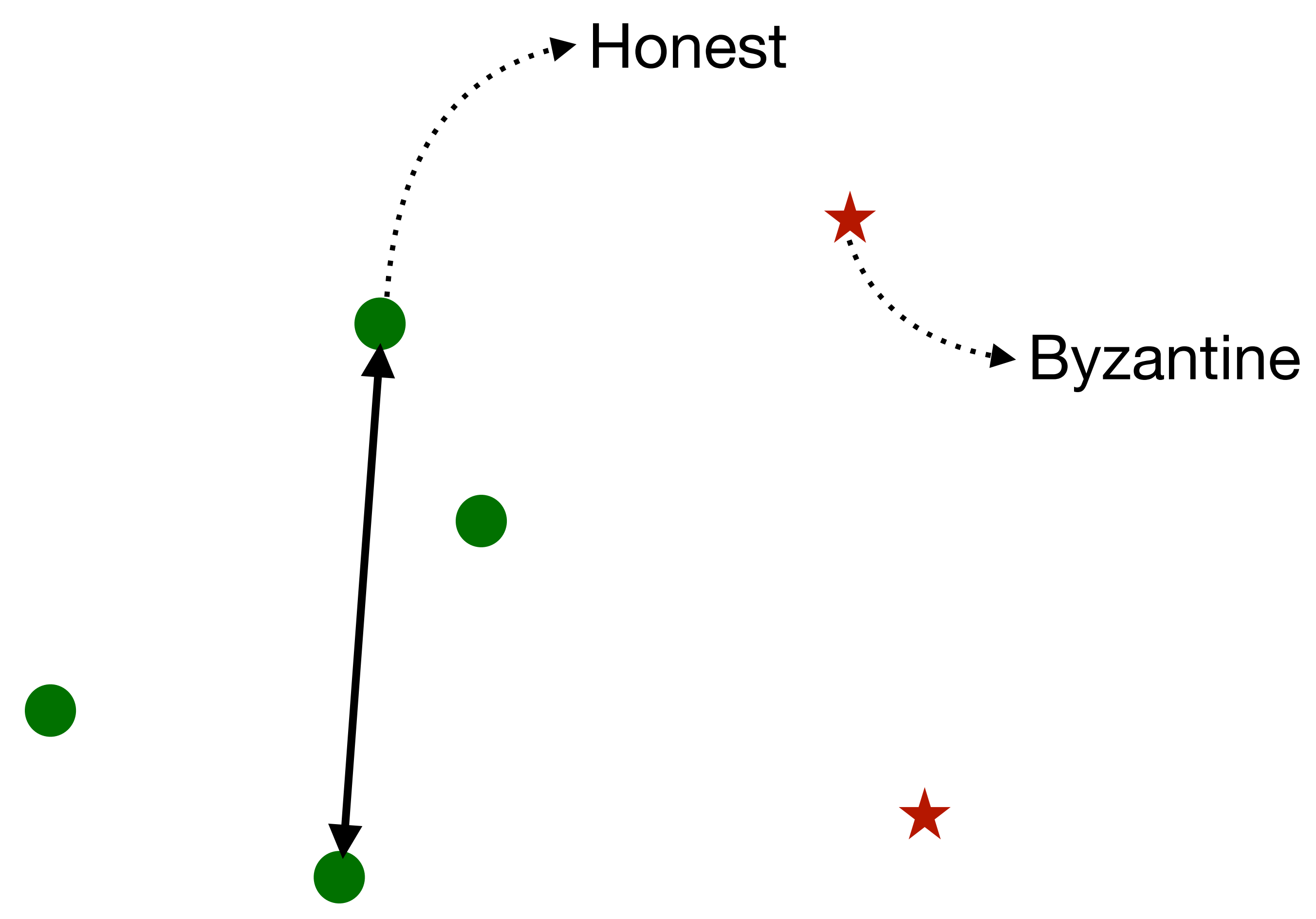
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

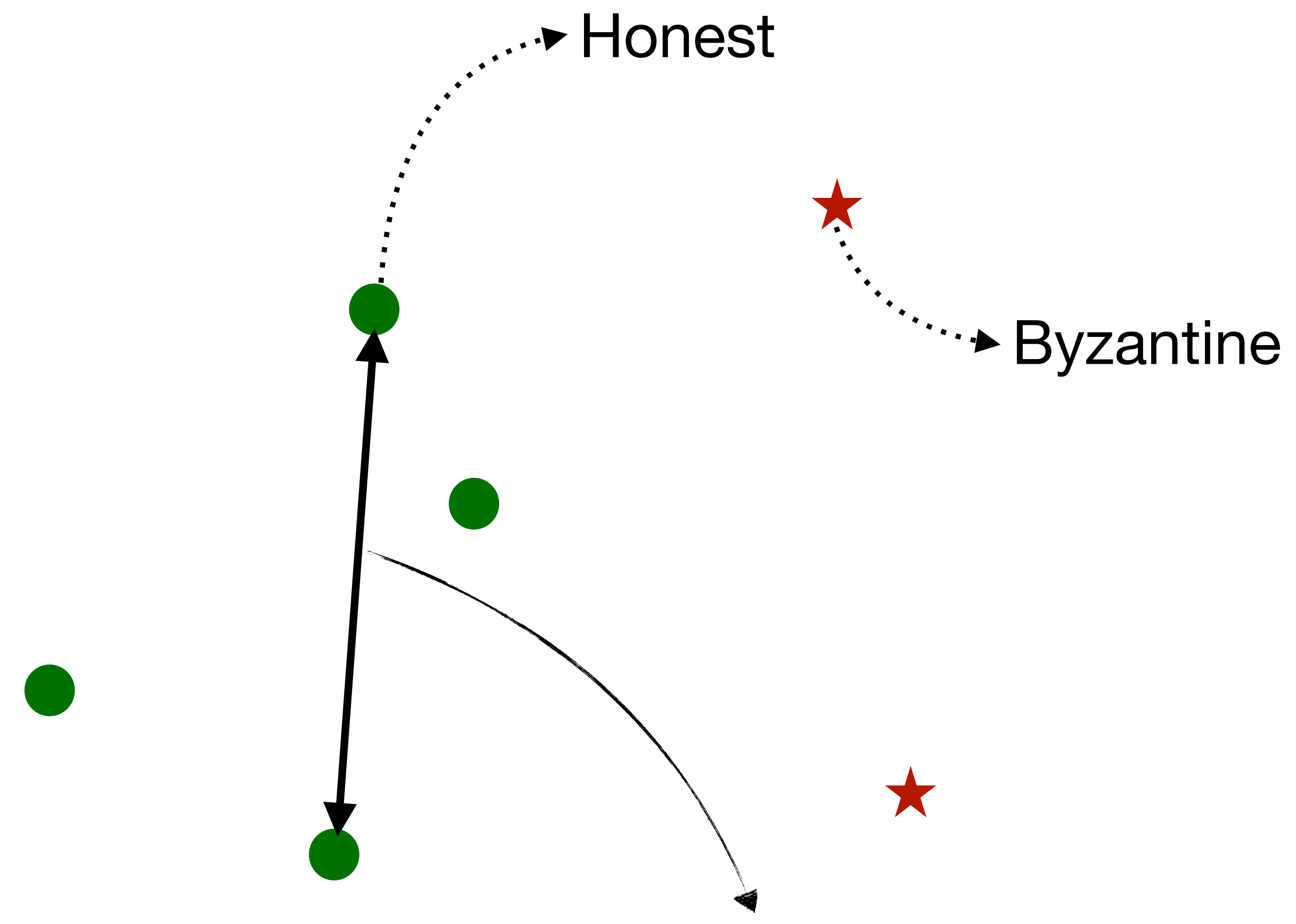
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

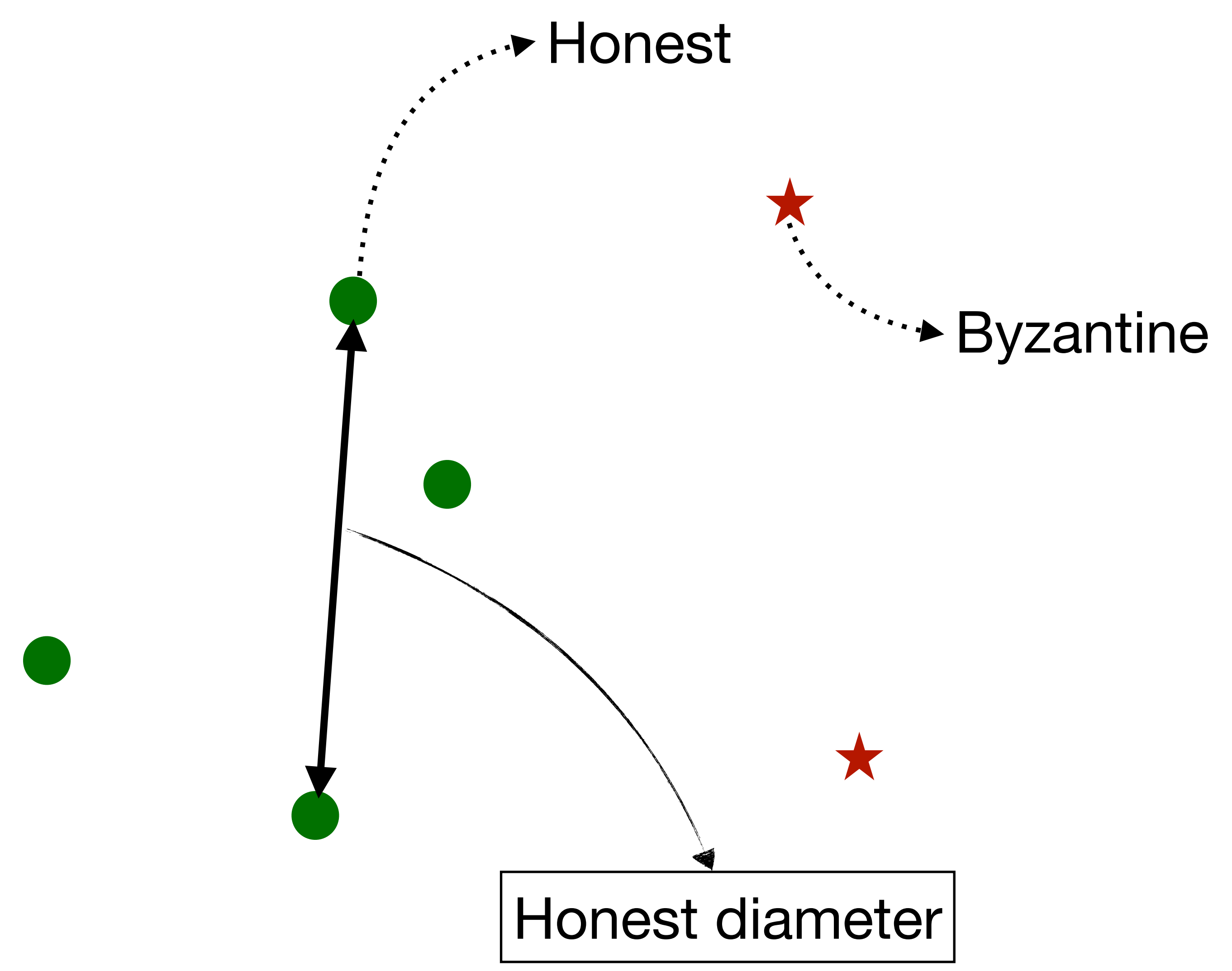
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

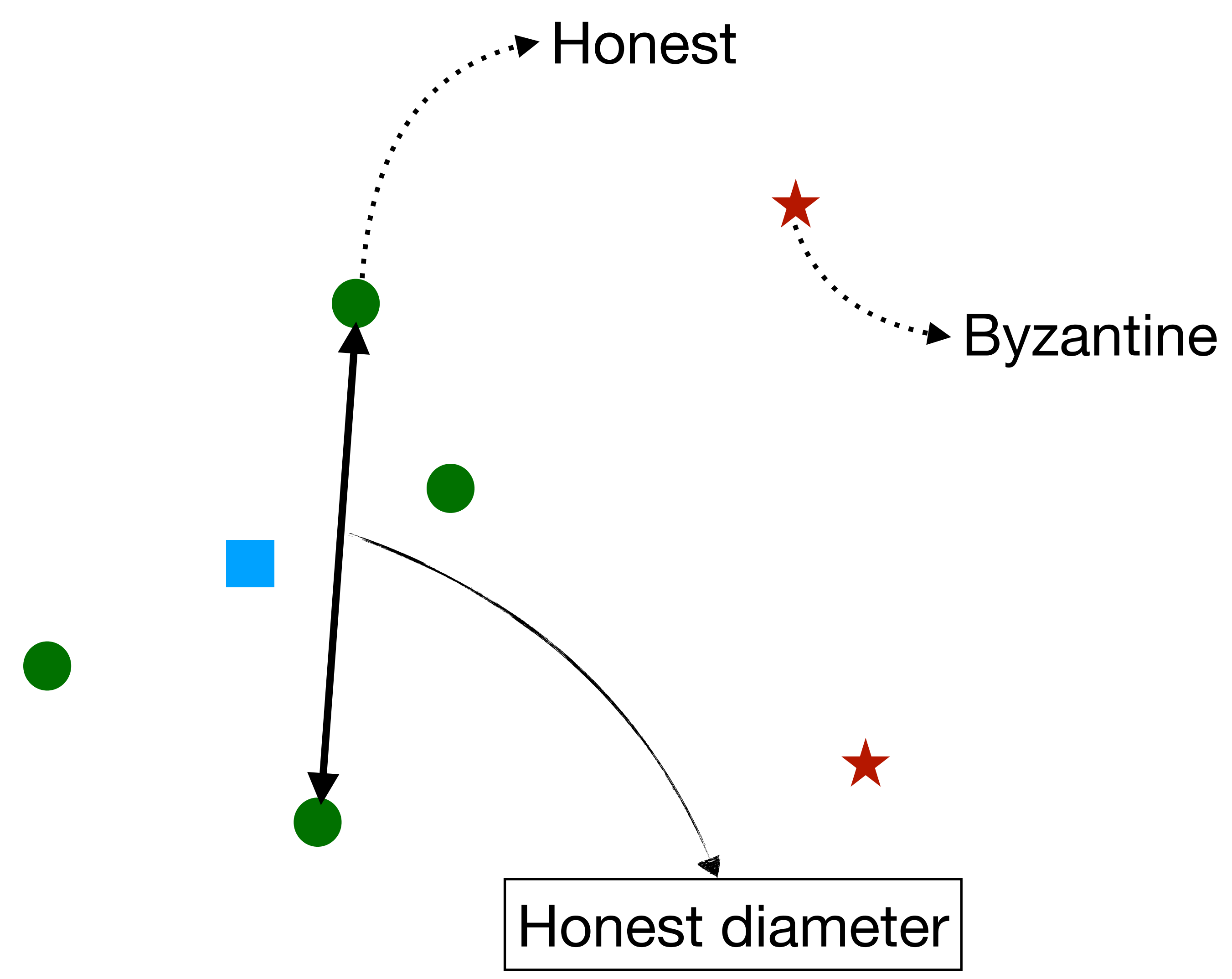
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

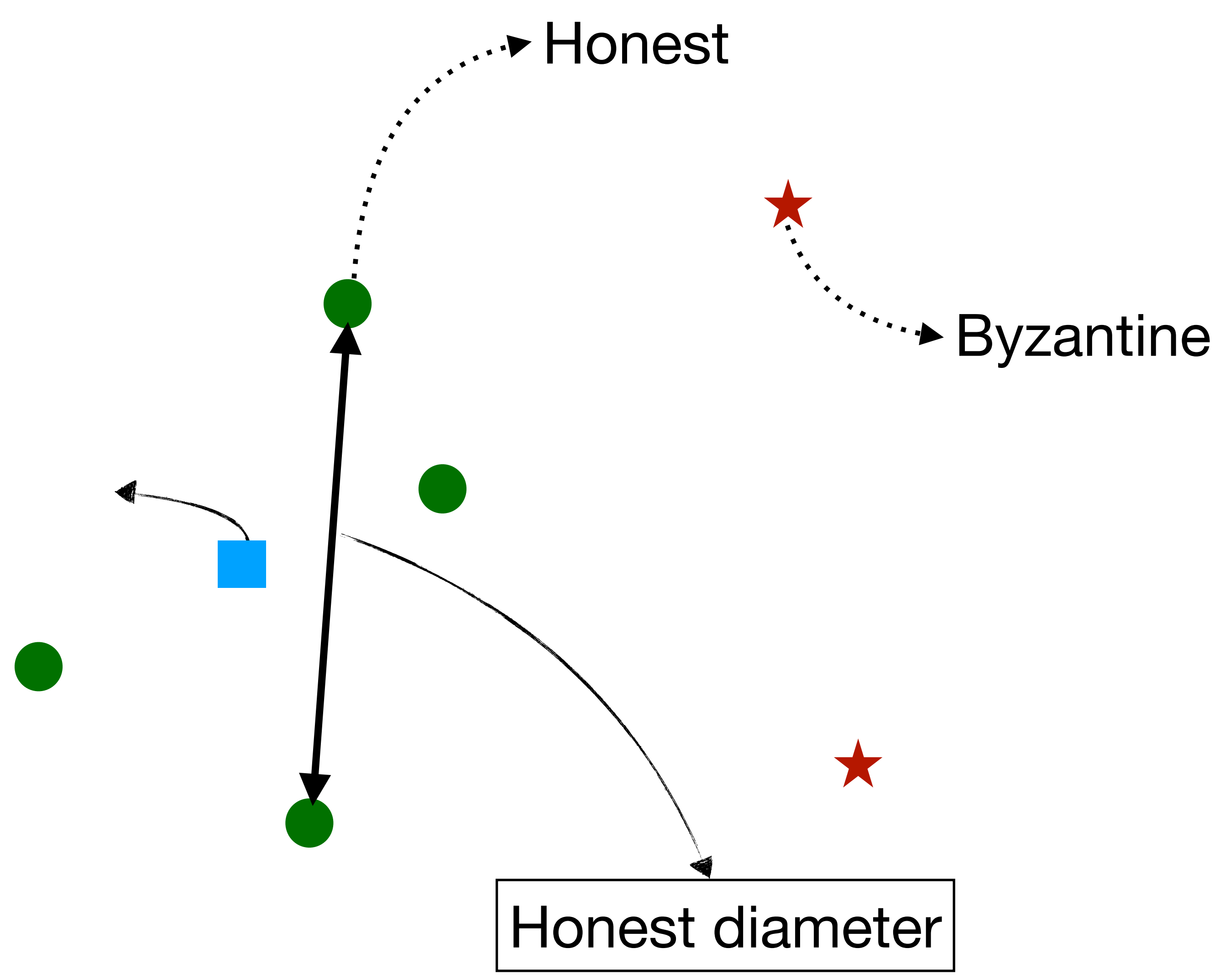
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

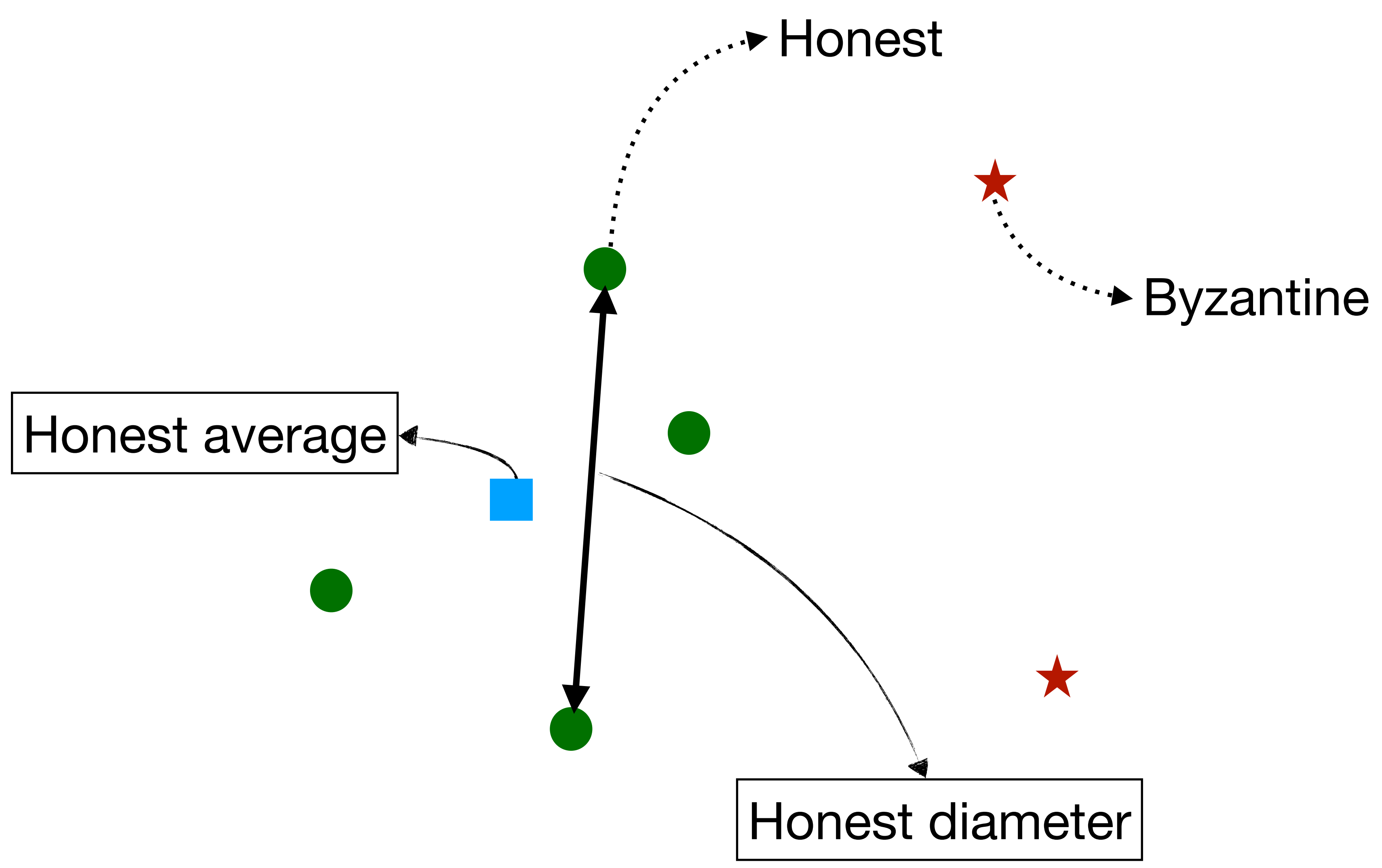
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\left\| F\left(x_1, \ldots, x_n\right) - \overline{x}_S \right\| \leq \lambda \max_{i,j \in S} \| x_i - x_j \|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

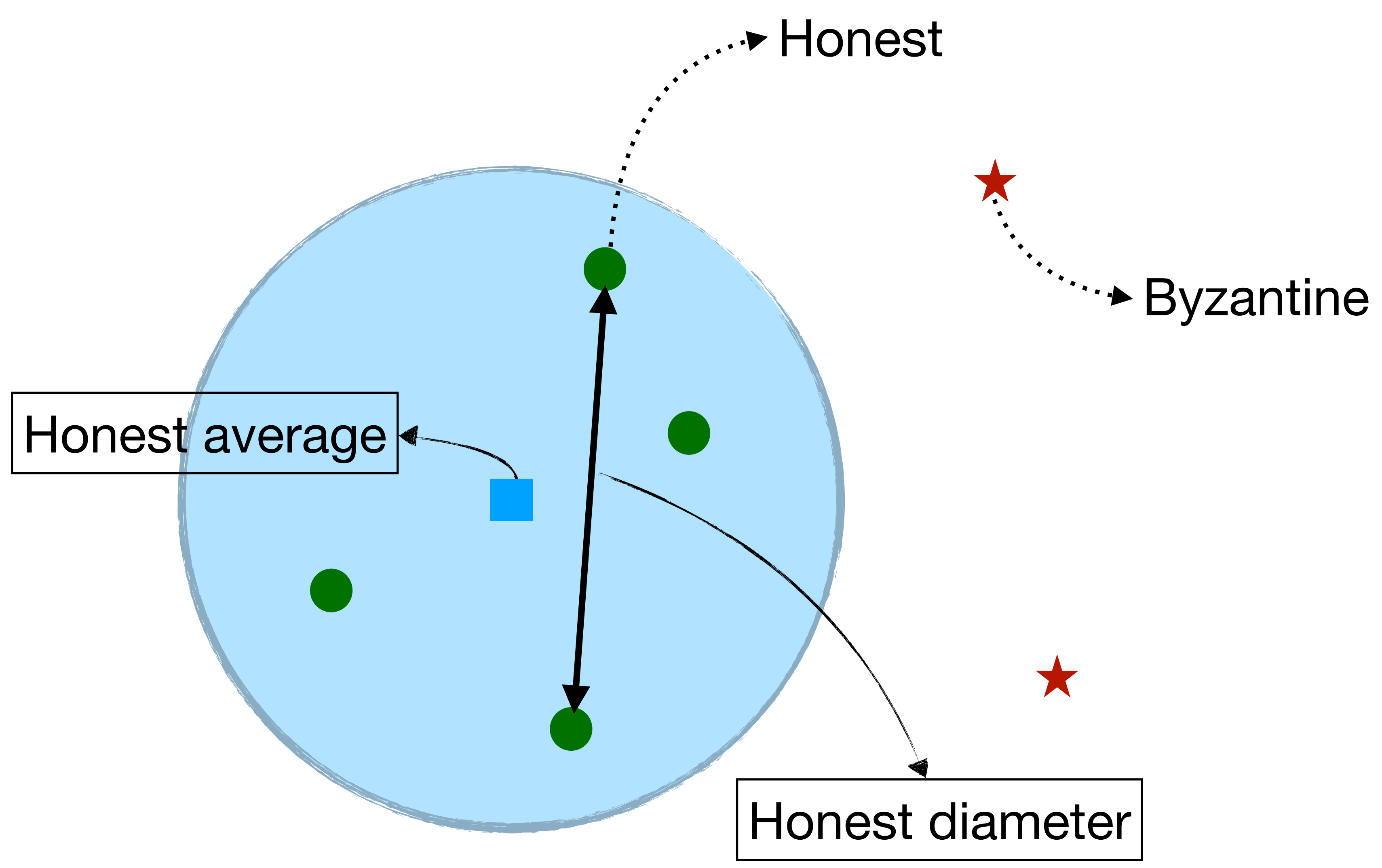
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

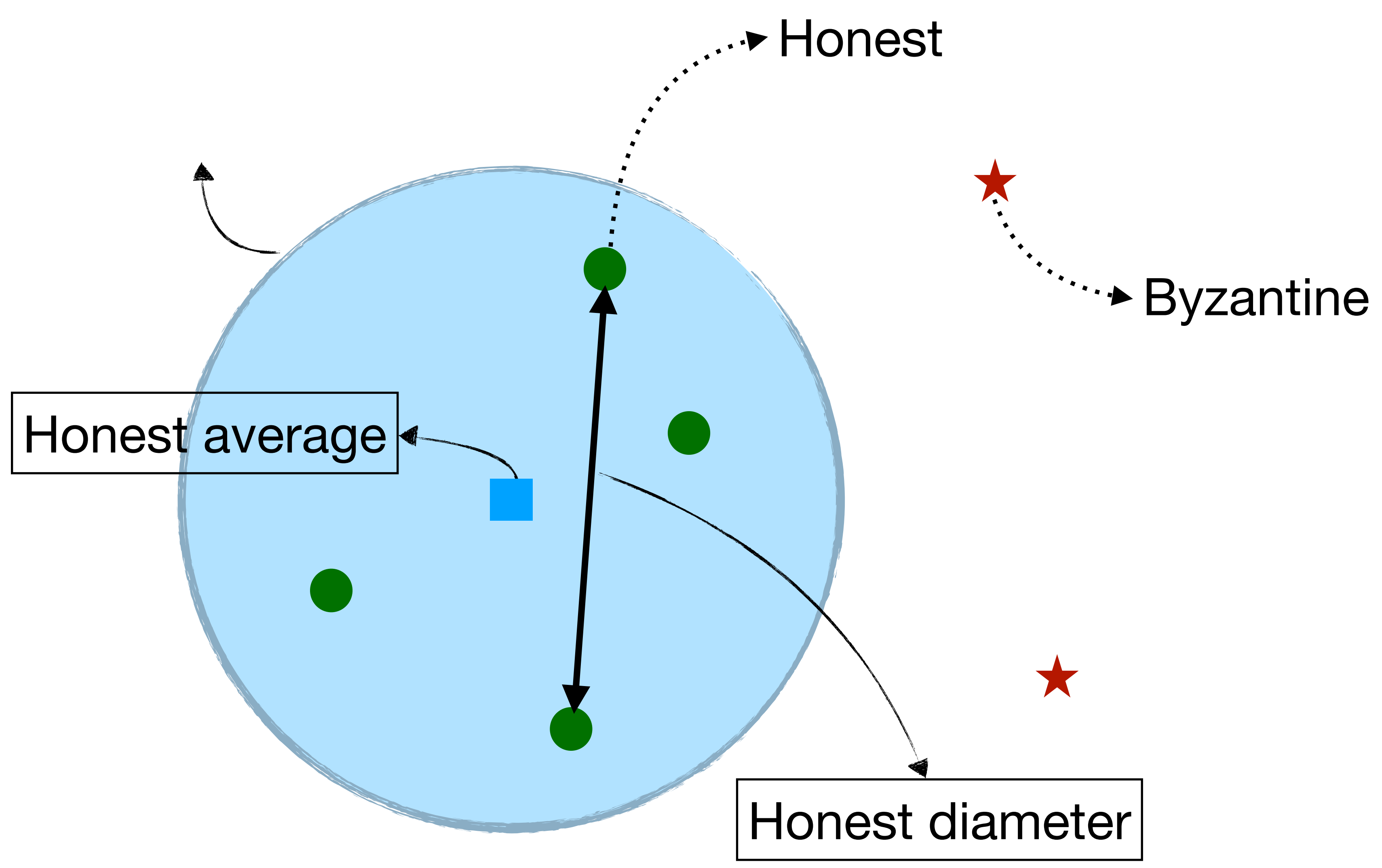
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \bar{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

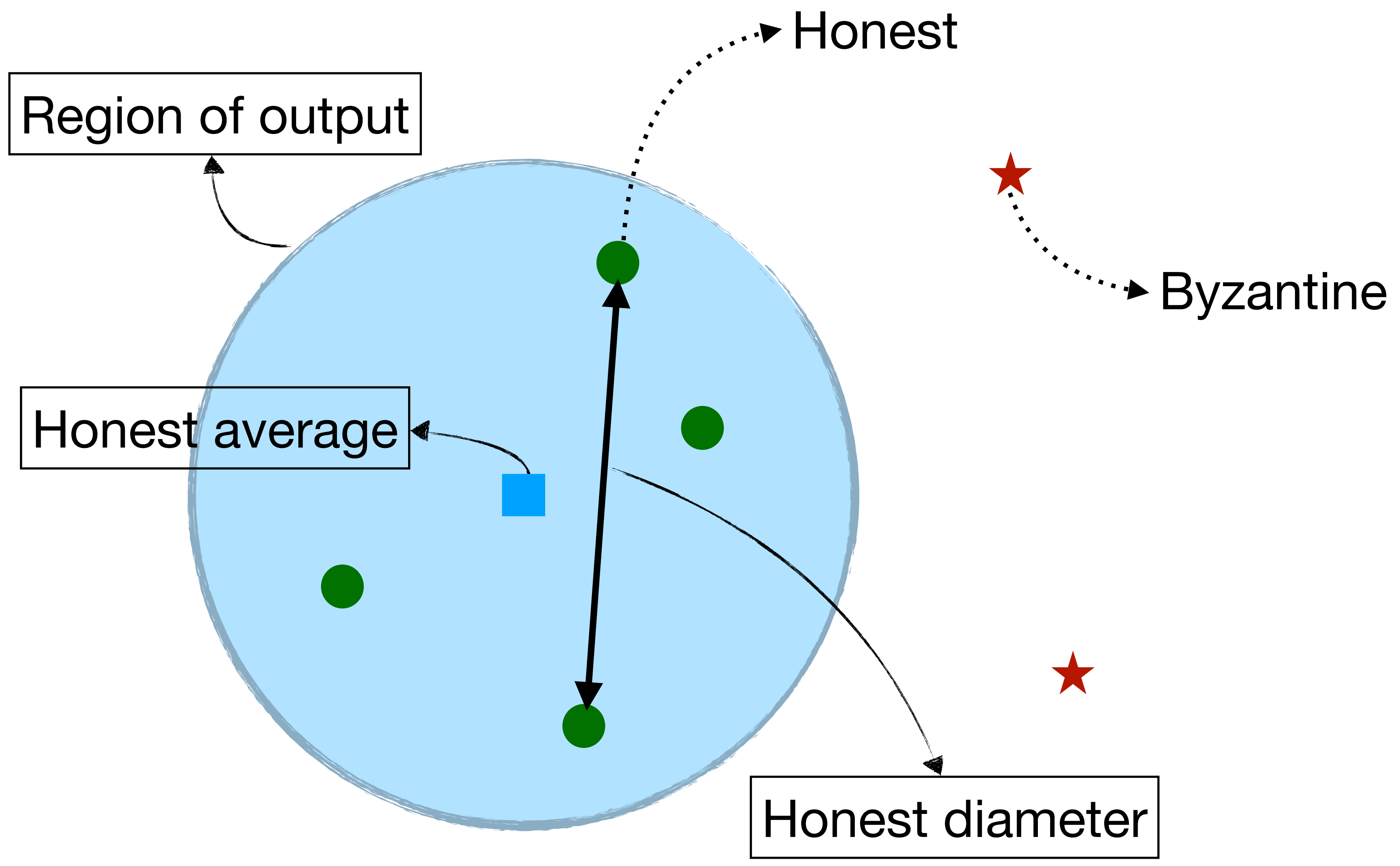
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

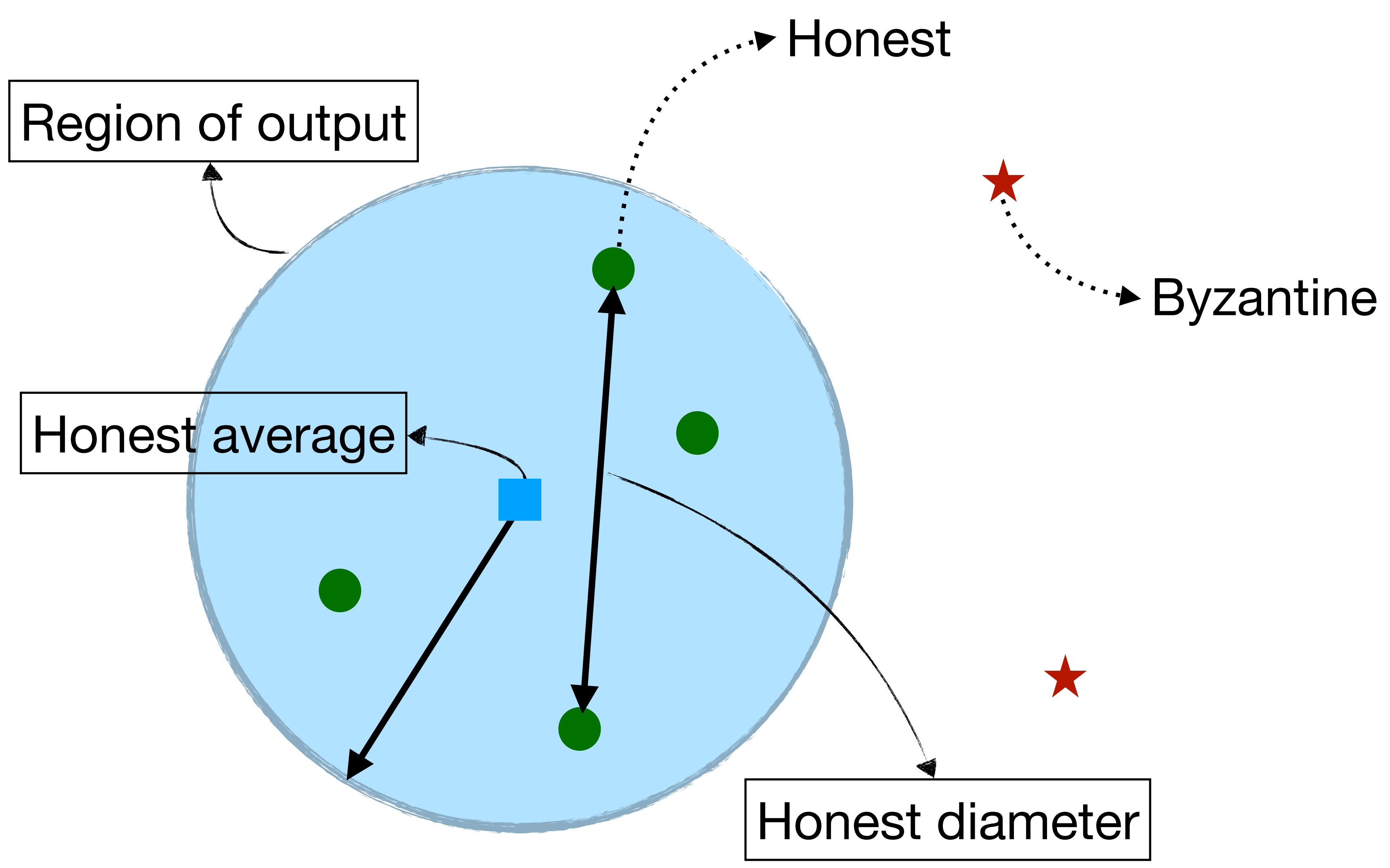
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1, \ldots, n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

***(f, λ)-resilient averaging (RESA)* -**

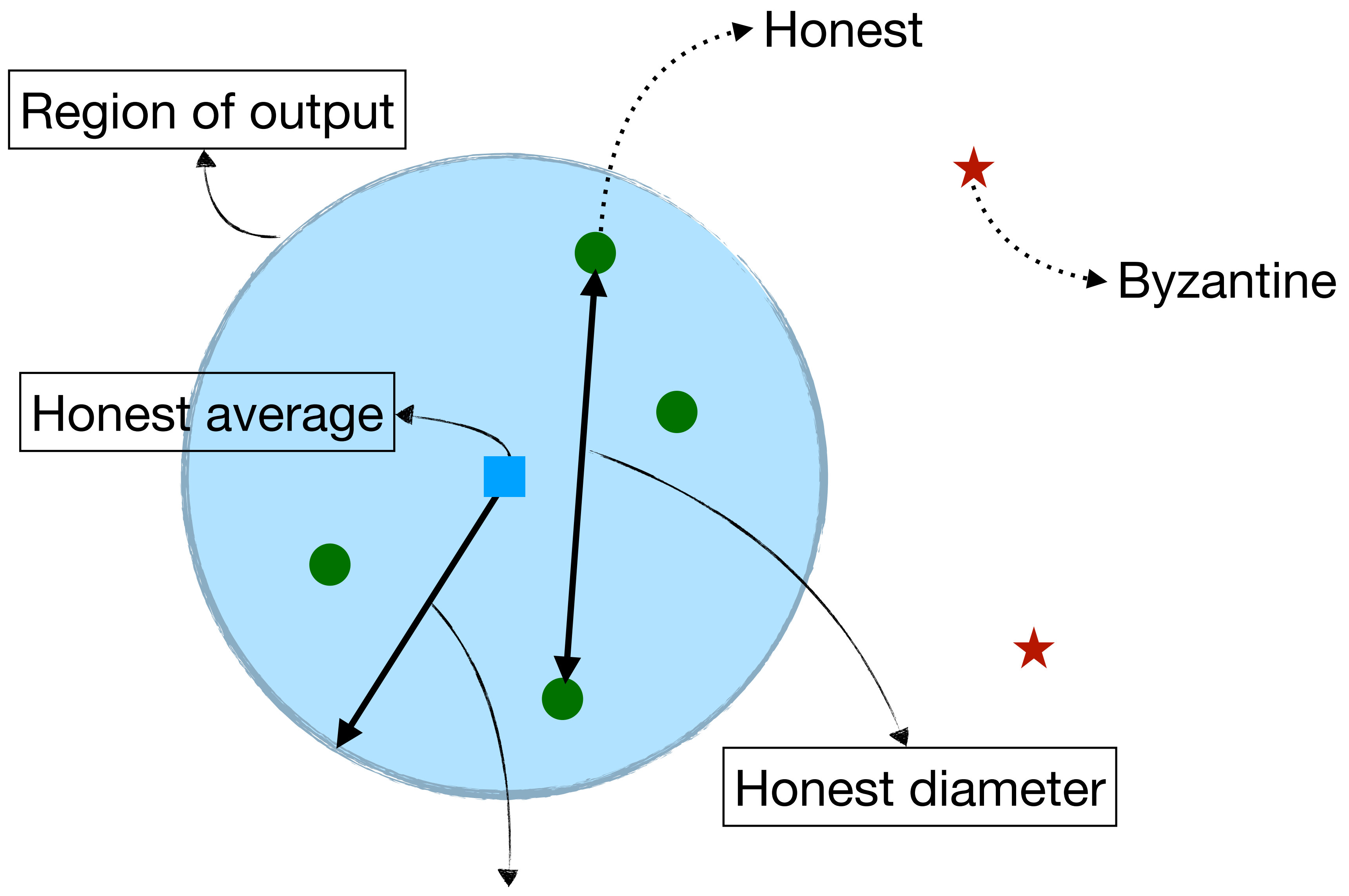
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

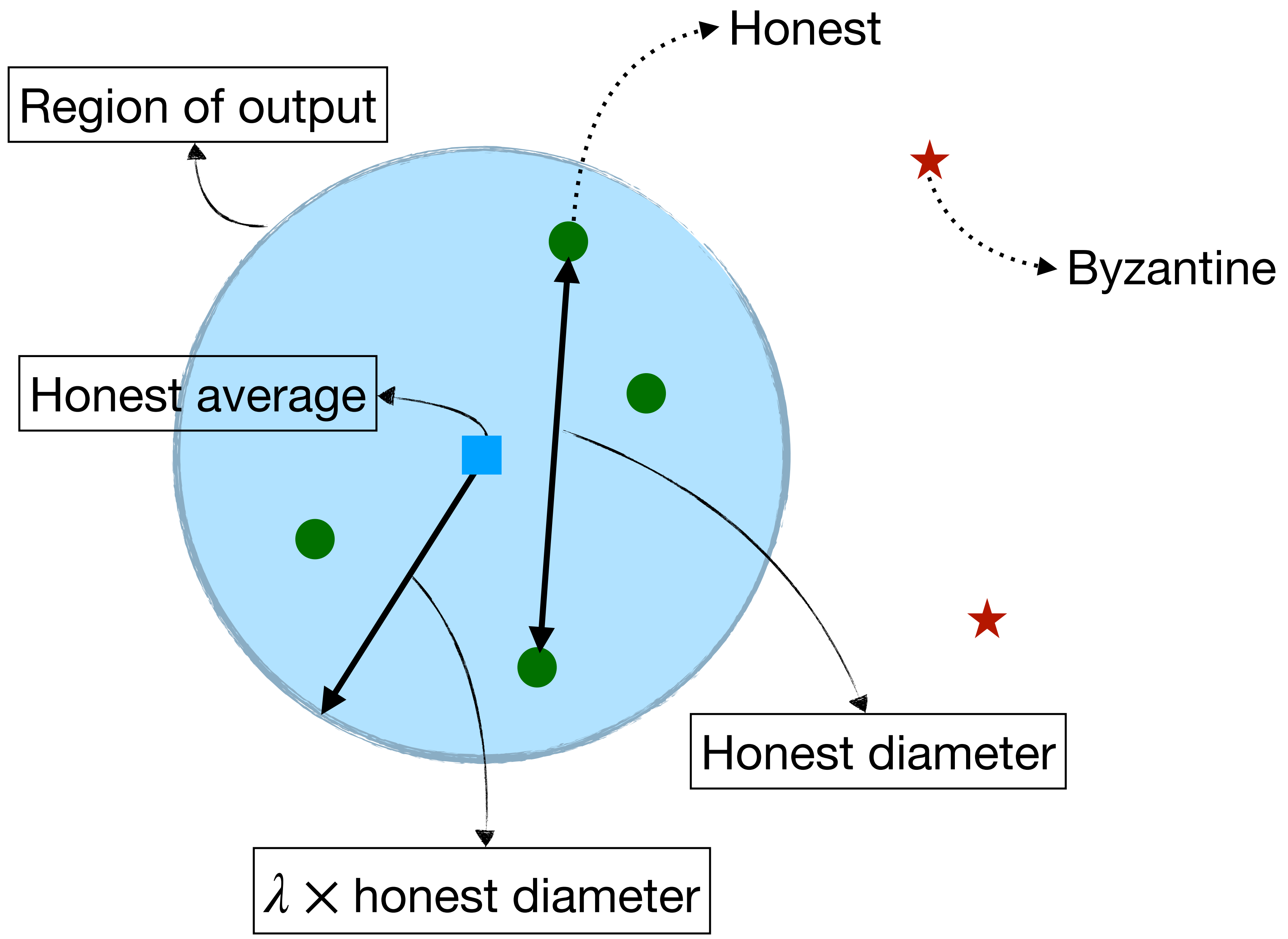
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F(x_1, \ldots, x_n) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

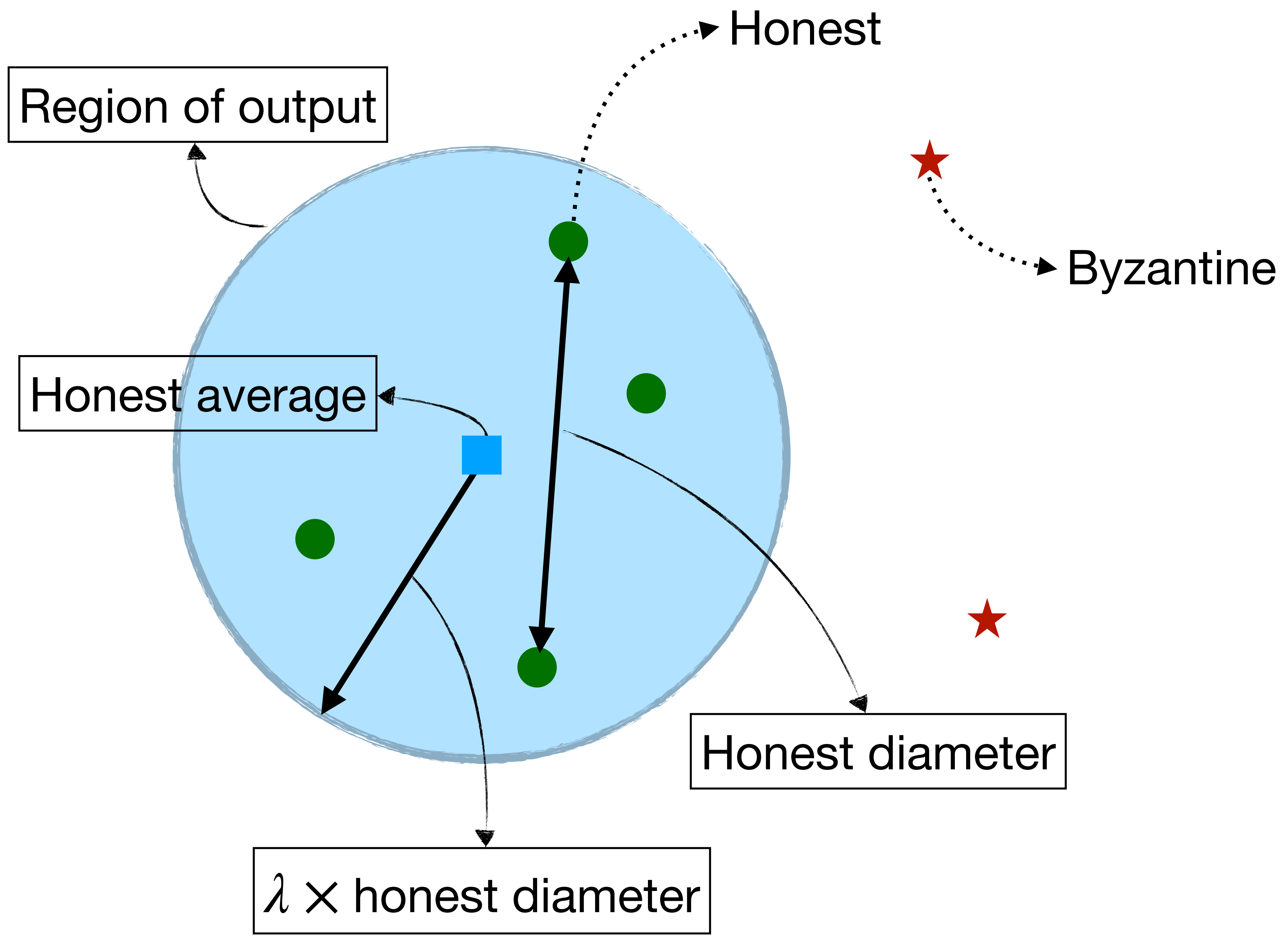
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n-f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



If all honest values are equal then RESA outputs that value.

Resilient Averaging (RESA)

Robustness property of F - **key** to optimally utilize Momentum

(f, λ)-resilient averaging (RESA) -

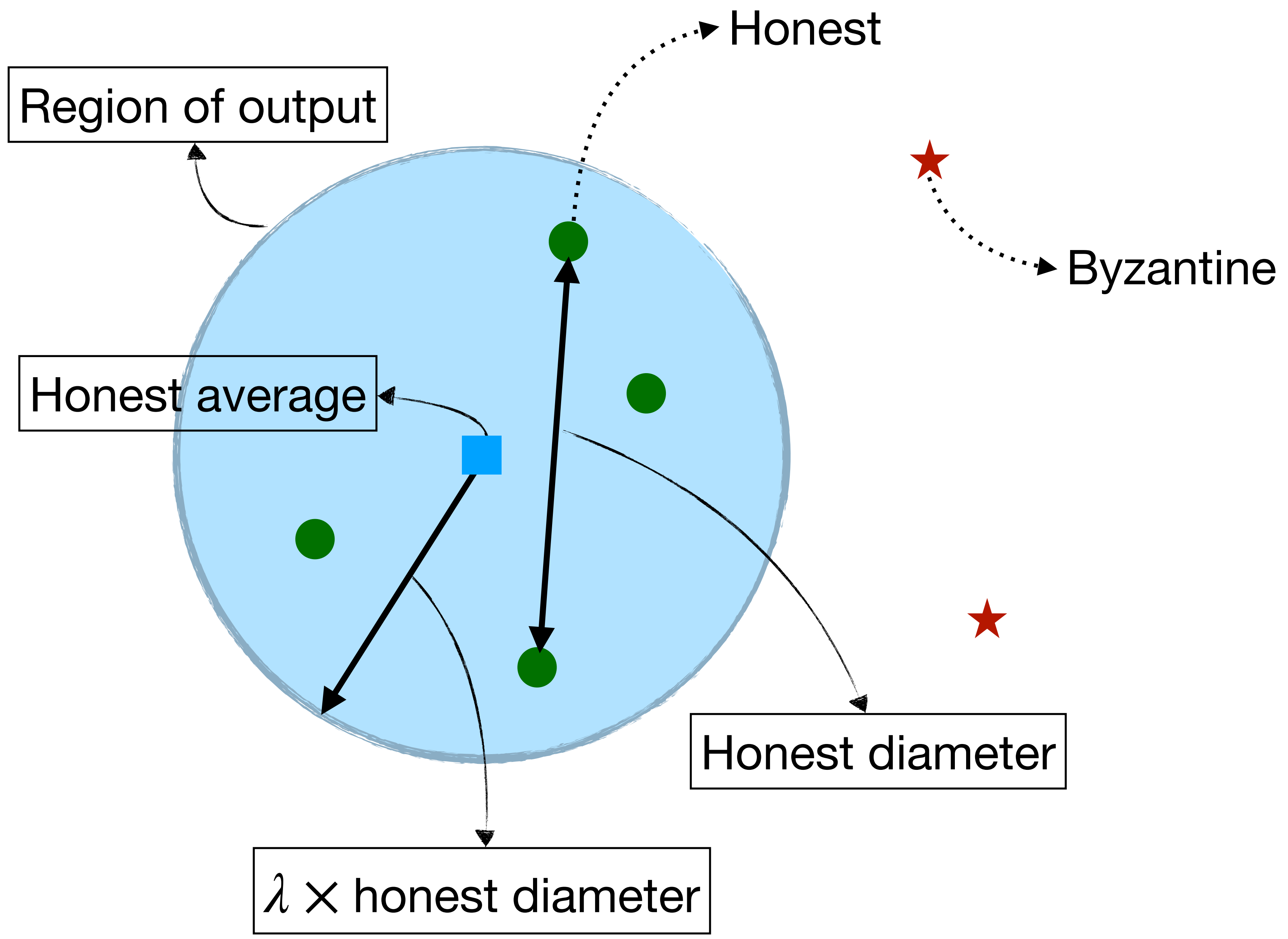
$\exists\ \lambda \geq 0$ s.t. $\forall$ set of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

and $\forall\ S \subset \{1,\ldots,n\}$ of size $n - f$,

$$\|F\left(x_1, \ldots, x_n\right) - \overline{x}_S\| \leq \lambda \max_{i,j \in S} \|x_i - x_j\|$$

Avg. of $\{x_i\}_{i \in S}$

Resilience coefficient



Sanity check

If all honest values are equal then RESA outputs that value.

Resilience Coefficients of some Aggregation Rules

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ					

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$				

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \frac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$			

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$		

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1 + \frac{n-f}{2\sqrt{n(n-2f)}}$	$1 + \sqrt{\frac{n-f}{n-2f}}$	

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$	$1+\sqrt{\frac{n-f}{n-2f}}$	$\frac{n}{2\sqrt{n-f}}$

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$	$1+\sqrt{\frac{n-f}{n-2f}}$	$\frac{n}{2\sqrt{n-f}}$

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \frac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$	$1+\sqrt{\frac{n-f}{n-2f}}$	$\frac{n}{2\sqrt{n-f}}$

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$	$1+\sqrt{\frac{n-f}{n-2f}}$	$\frac{n}{2\sqrt{n-f}}$

Order optimal

Resilience Coefficients of some Aggregation Rules

Smaller the λ better the Byzantine resilience

Lower bound: $\lambda \geq \dfrac{f}{n-f}$

Aggregation:	MDA	TM	Geometric Median	Krum	Median
λ	$\frac{2f}{n-f}$	$\frac{f}{2\sqrt{n-f}}$	$1+\frac{n-f}{2\sqrt{n(n-2f)}}$	$1+\sqrt{\frac{n-f}{n-2f}}$	$\frac{n}{2\sqrt{n-f}}$

Order optimal

The values hold for any $f < n/2$

Byzantine Resilience Guarantee

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta \, m_{t-1}^i + (1-\beta)\, g_t^i$ for each honest node i

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta \, m_{t-1}^i + (1 - \beta) \, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta m_{t-1}^i + (1-\beta)\, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

(f, λ)-**RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\widehat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t\, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

(f, λ)-**RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\widehat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

$$\mathbb{E}\left[\|\nabla Q\left(\widehat{\theta}\right)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta \, m_{t-1}^i + (1-\beta) \, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

(f, λ)-**RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\widehat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

$$\mathbb{E}\left[\|\nabla Q\left(\widehat{\theta}\right)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

$(f, \epsilon)-$Resilience

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t\, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta\, m_{t-1}^i + (1-\beta)\, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

(f, λ)**-RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\hat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

$$\mathbb{E}\left[\|\nabla Q\left(\hat{\theta}\right)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

(f, ϵ)–Resilience

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta \, m_{t-1}^i + (1-\beta)\, g_t^i$ for each honest node i

Upon T iterations of RESAM

(f, λ)**-RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\widehat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

$$\mathbb{E}\left[\|\nabla Q\left(\widehat{\theta}\right)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

(f, ϵ)–Resilience

Byzantine Resilience Guarantee

Server updates: $\theta_{t+1} \longleftarrow \theta_t - \gamma_t \, R_t$

where $R_t = F\left(m_t^1, \ldots, m_t^n\right)$ and $m_t^i = \beta \, m_{t-1}^i + (1-\beta) \, g_t^i$ for each honest node i

Upon T iterations of **RESAM**

(f, λ)-**RESA** aggregation rule F + **M**omentum with $\beta = \sqrt{1 - c/T}$

Let $\widehat{\theta}$ be chosen uniformly from $\theta_1, \ldots, \theta_T$ then

$$\mathbb{E}\left[\|\nabla Q\left(\widehat{\theta}\right)\|^2\right] \leq \epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

(f, ϵ)–Resilience

Compares overall efficiency for different aggregation rules

Empirical Evidence

Empirical Evidence

* MNIST classification task

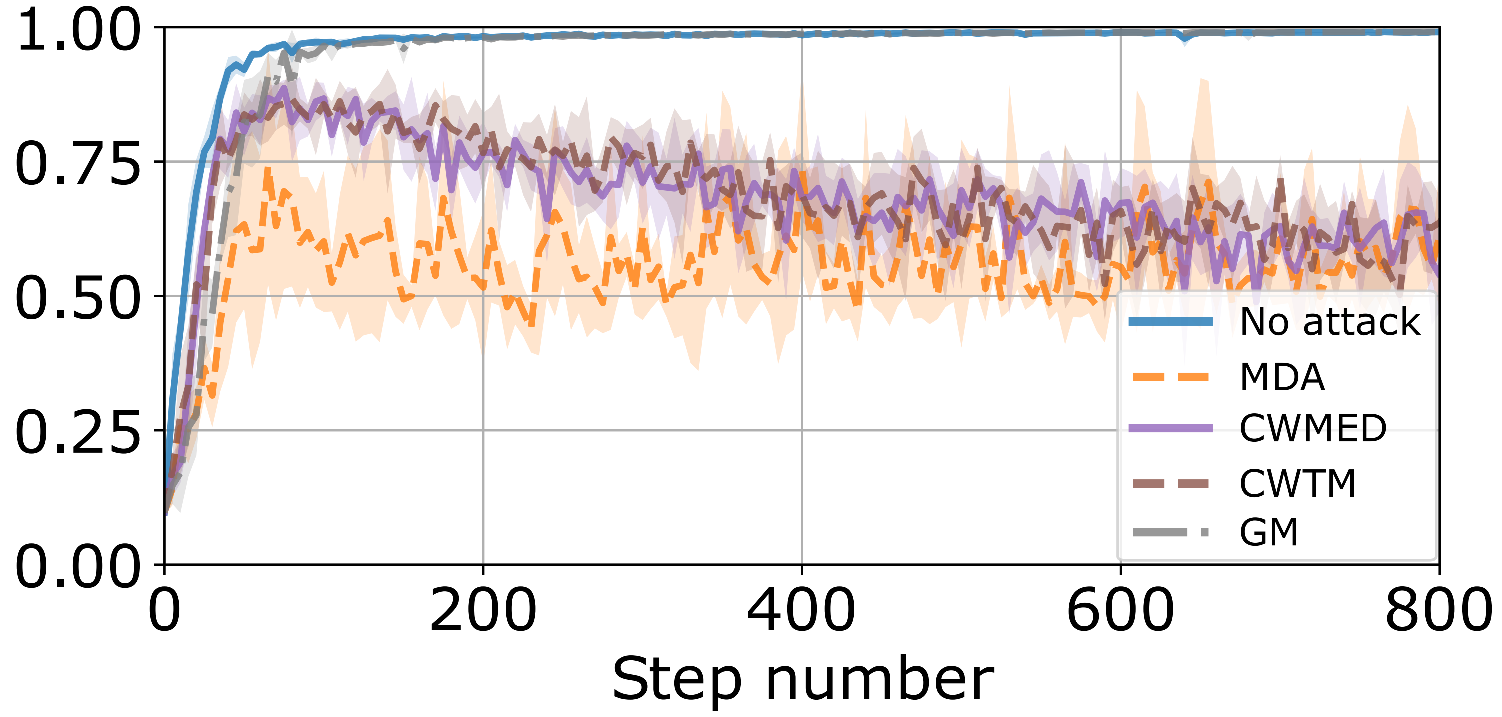
Empirical Evidence

Without Momentum

* MNIST classification task

Empirical Evidence

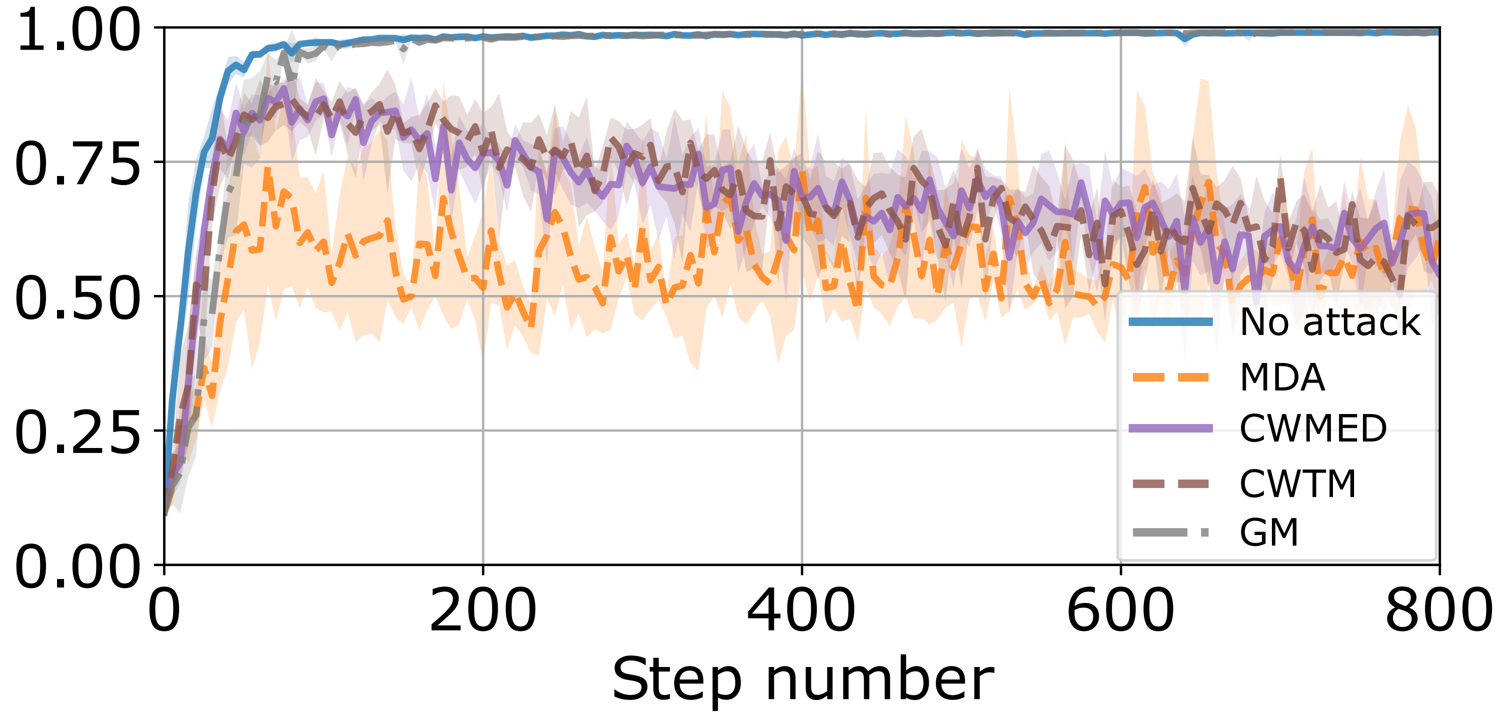
Without Momentum



* MNIST classification task

Empirical Evidence

Without Momentum

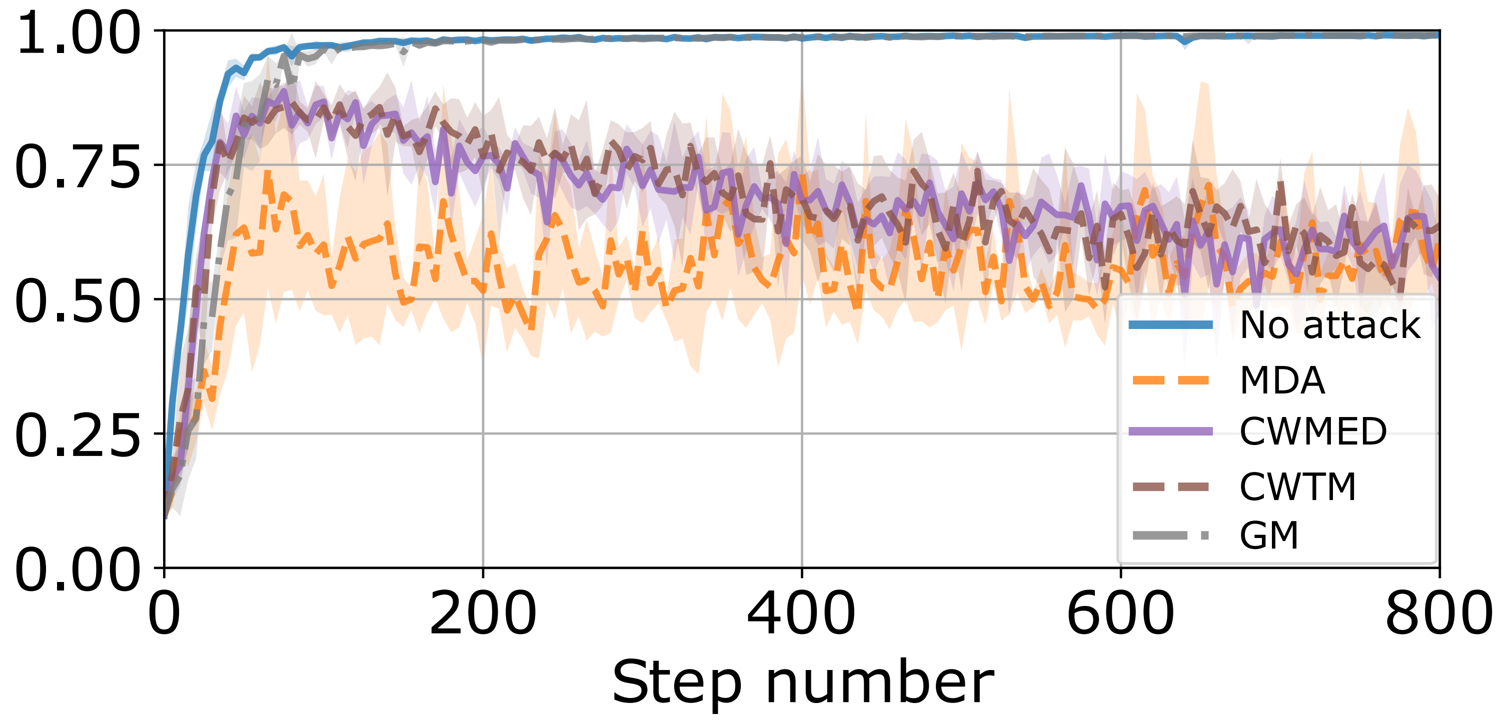


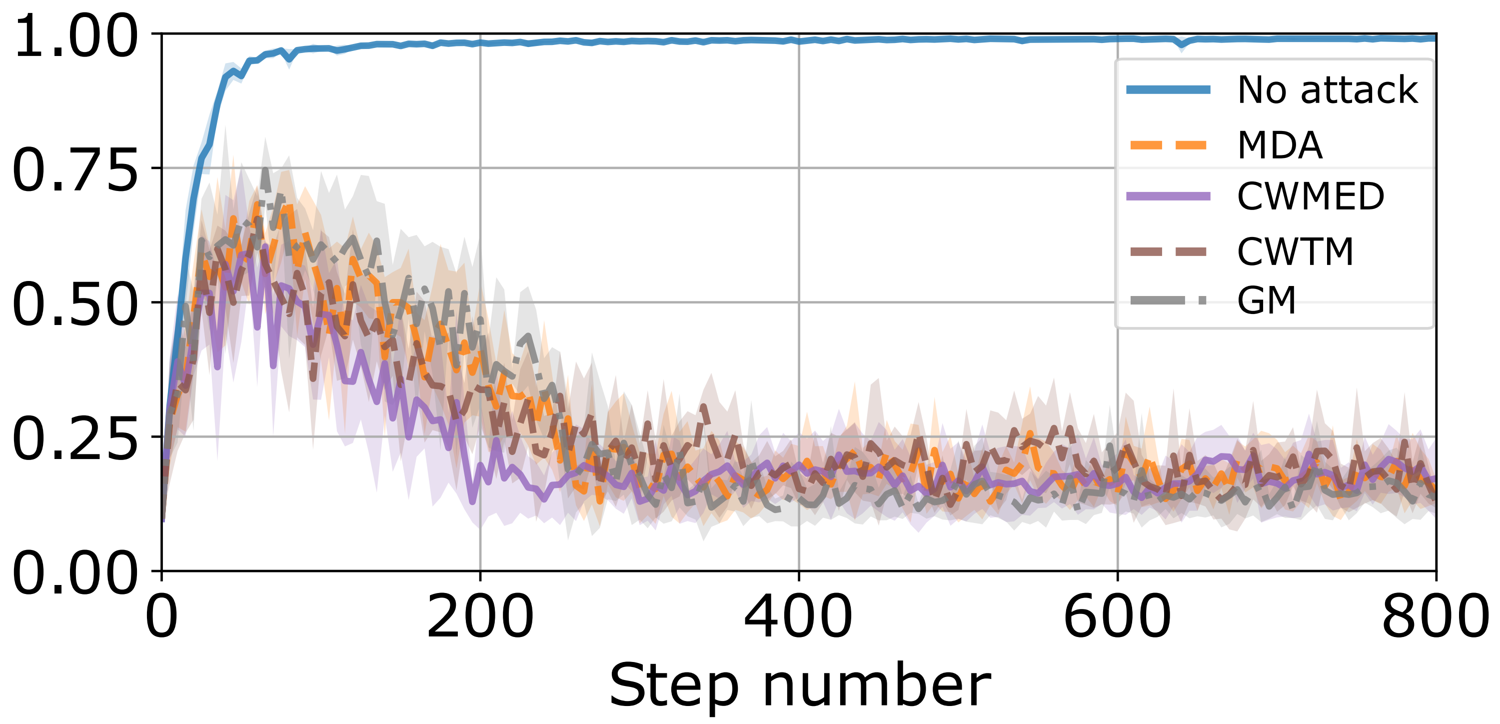
Label-flipping

* MNIST classification task

Empirical Evidence

Without Momentum



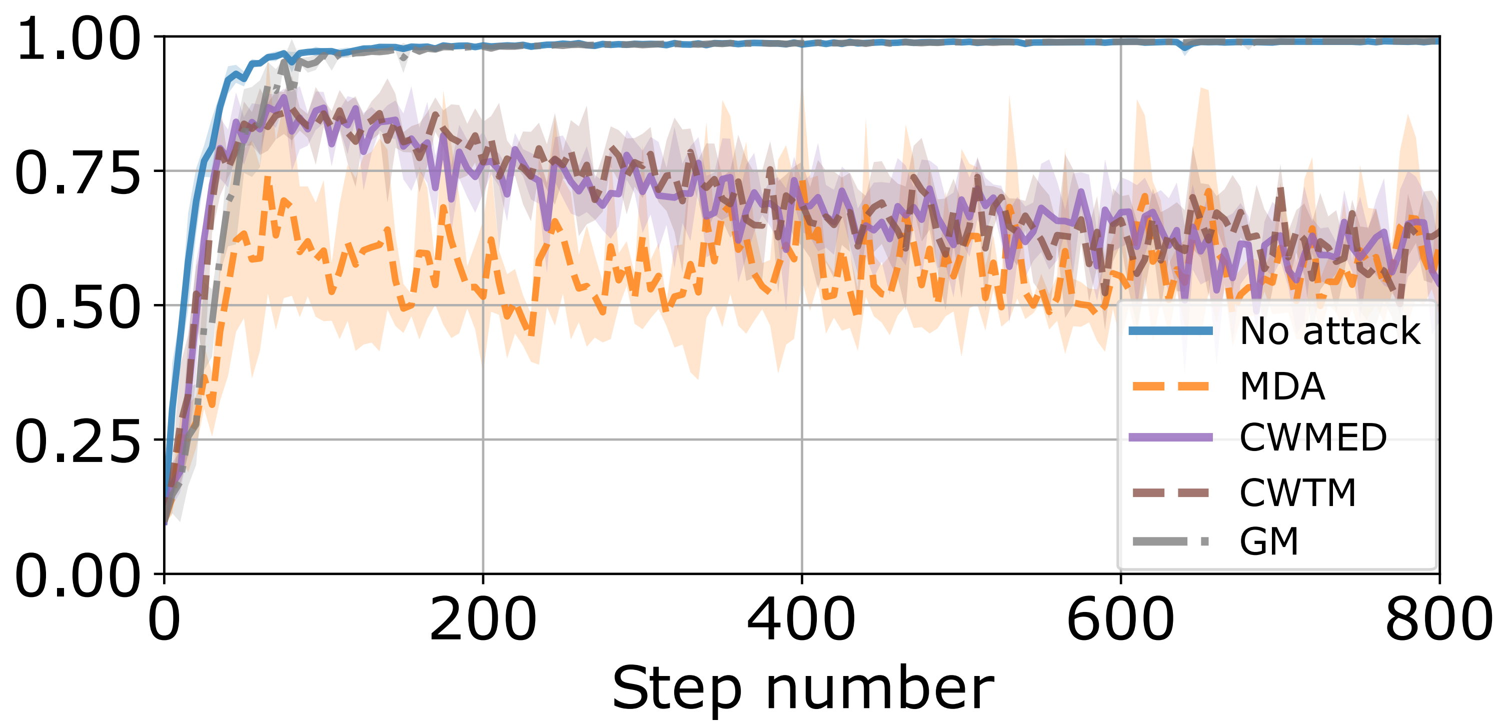


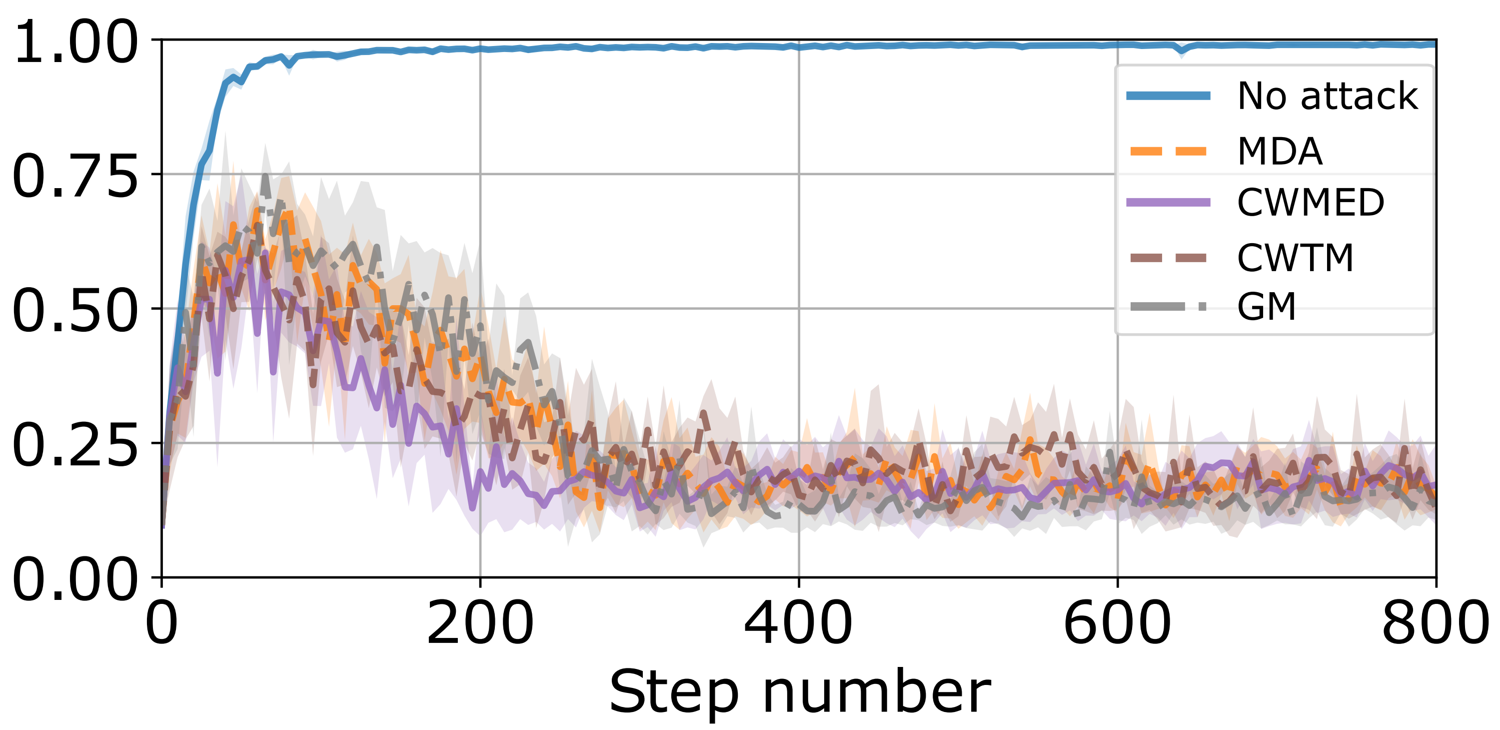
Label-flipping

* MNIST classification task

Empirical Evidence

Without Momentum





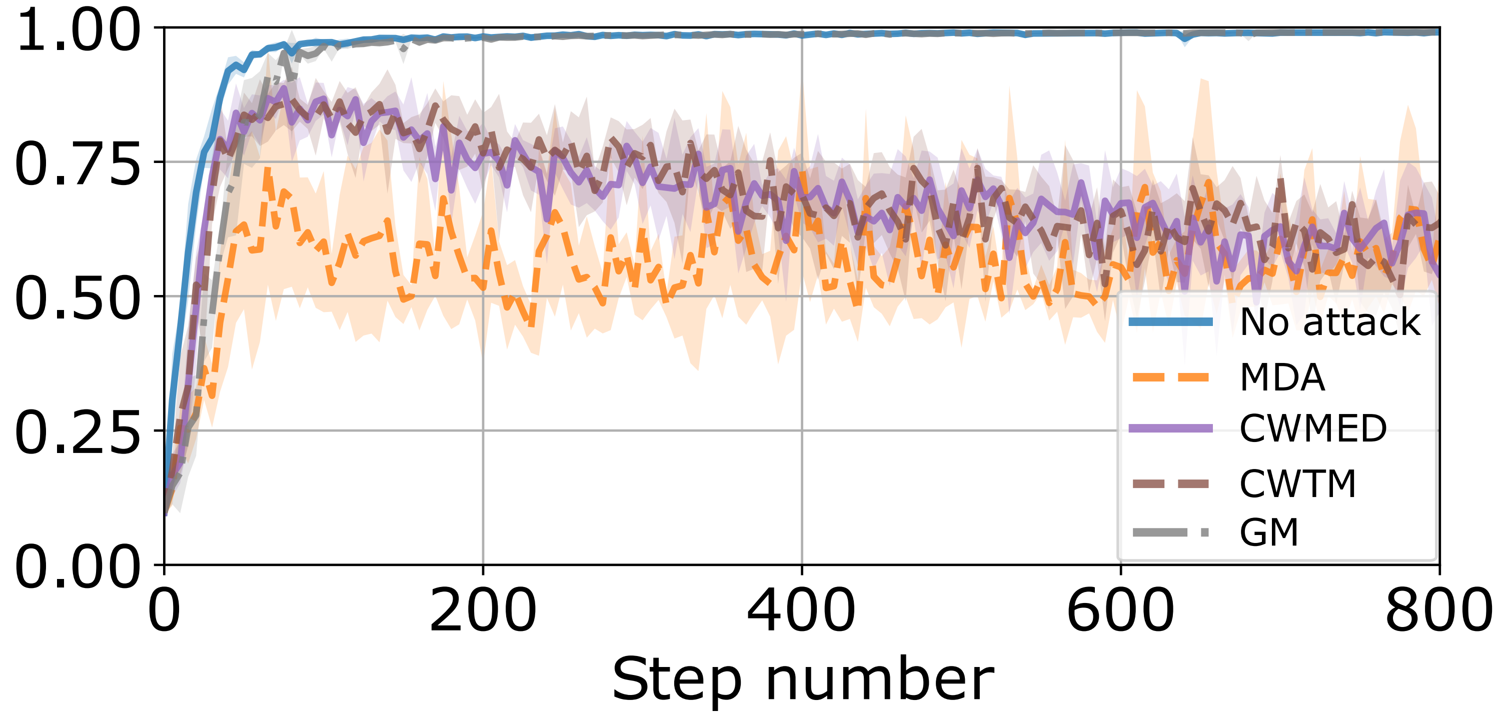
Label-flipping

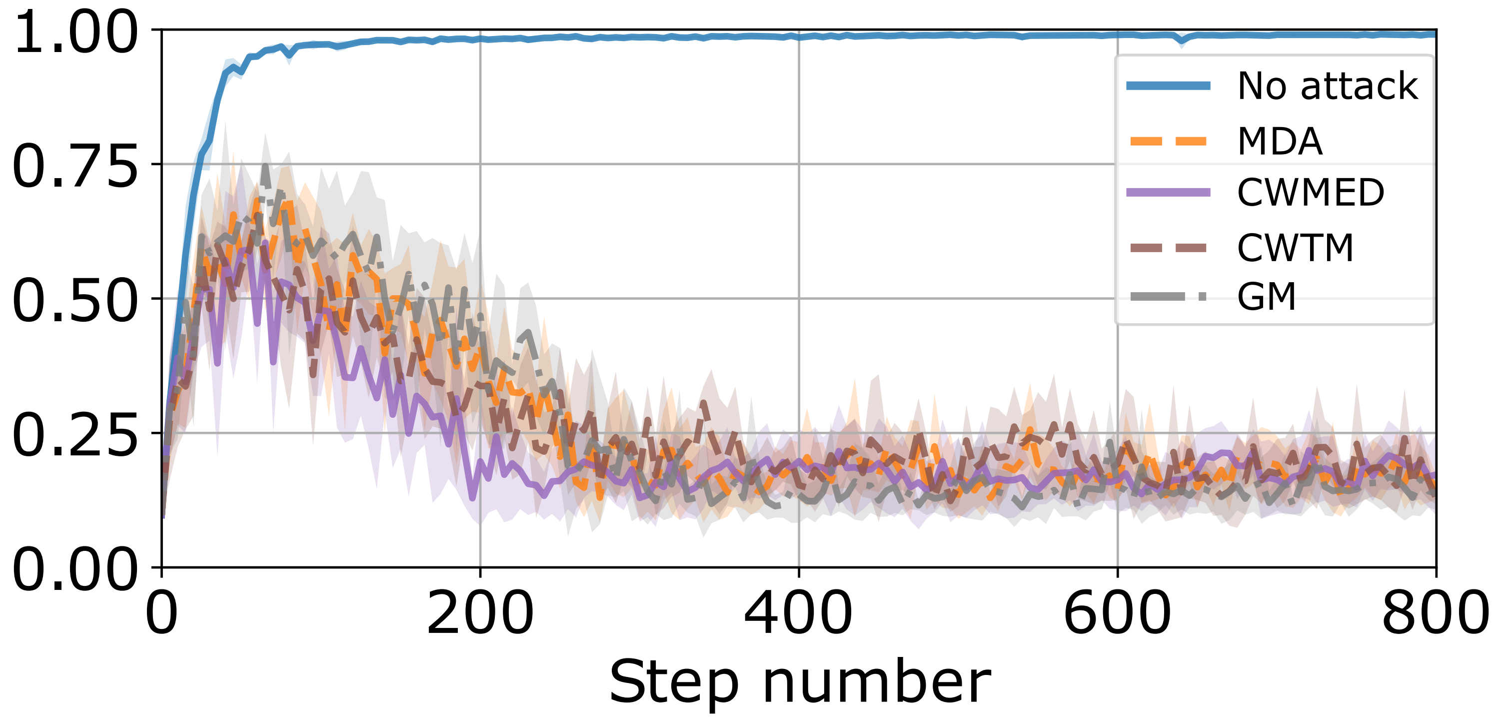
Little is enough *(Baruch et al., 2019)*

* MNIST classification task

Empirical Evidence

Without Momentum





With Momentum

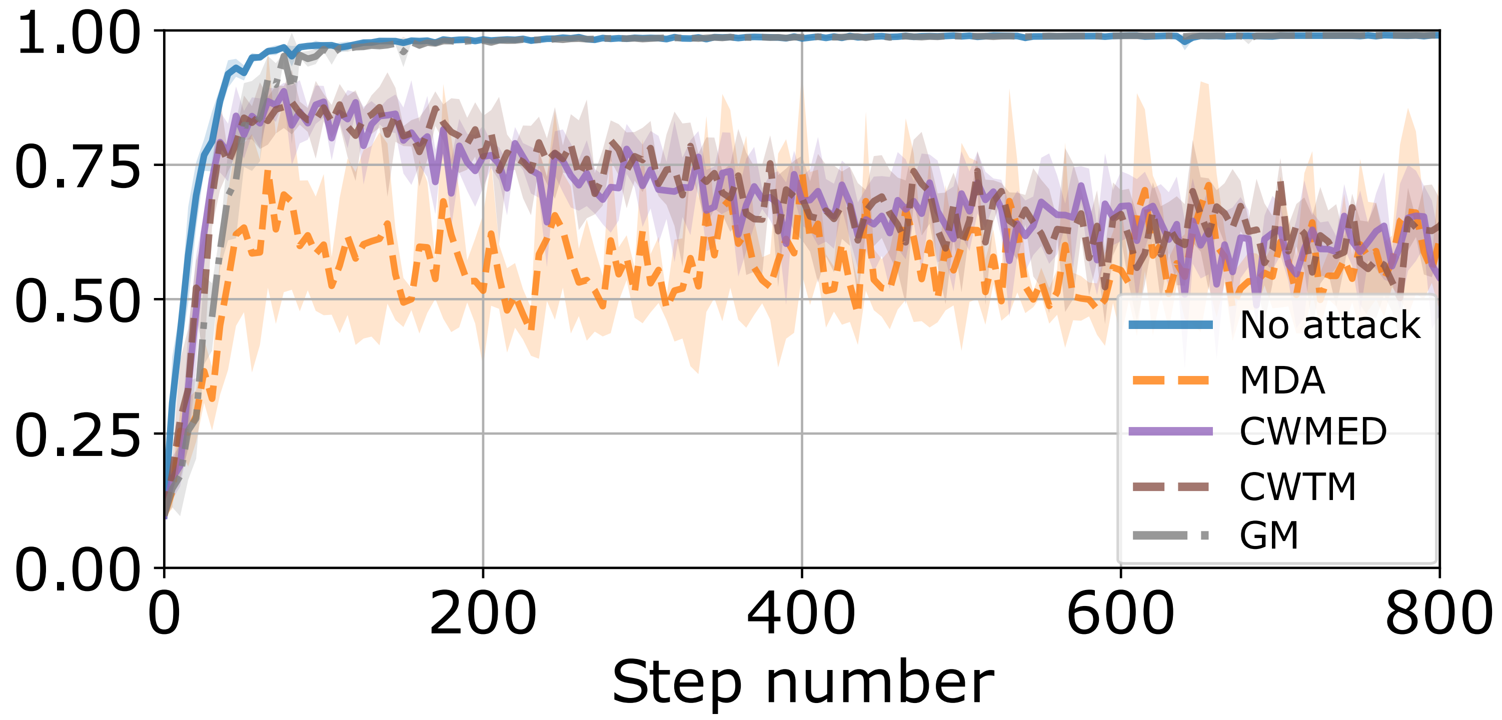
Label-flipping

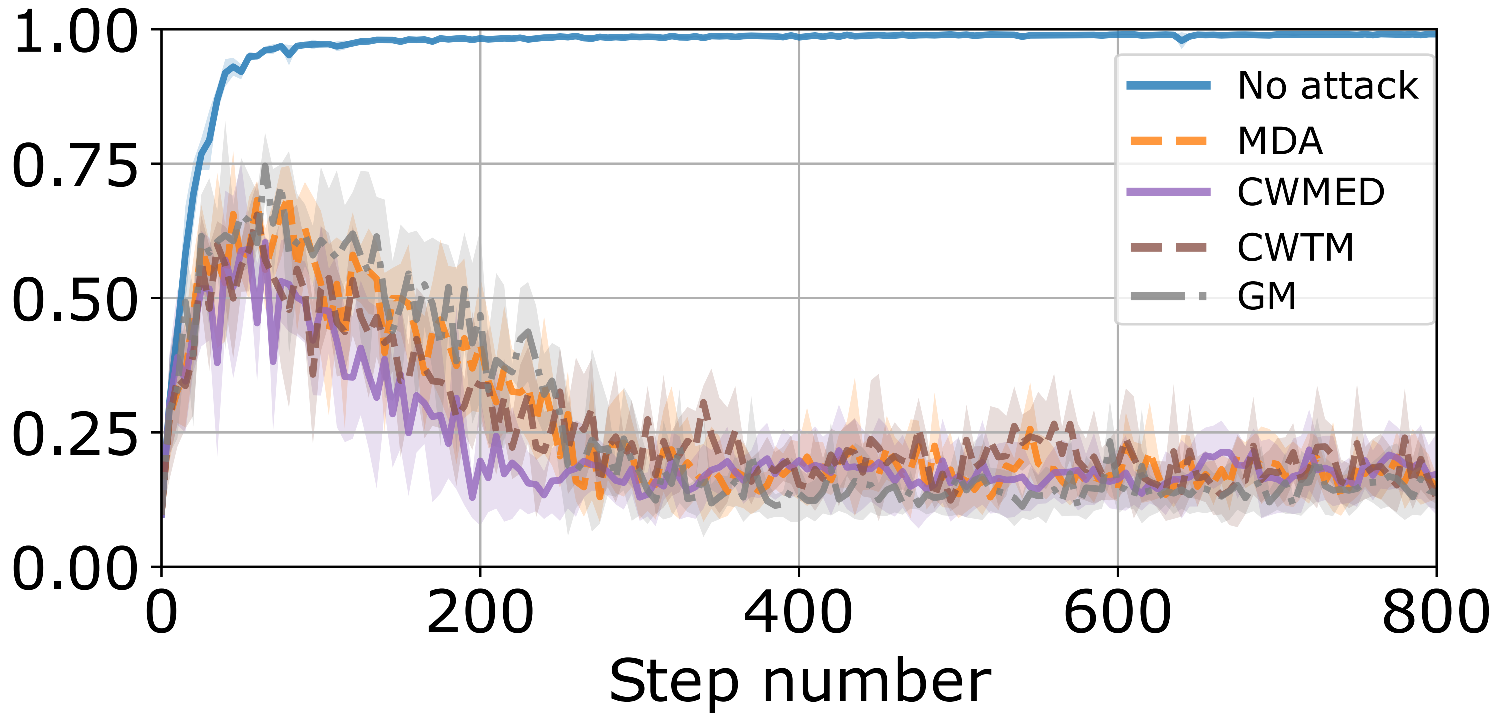
Little is enough *(Baruch et al., 2019)*

* MNIST classification task

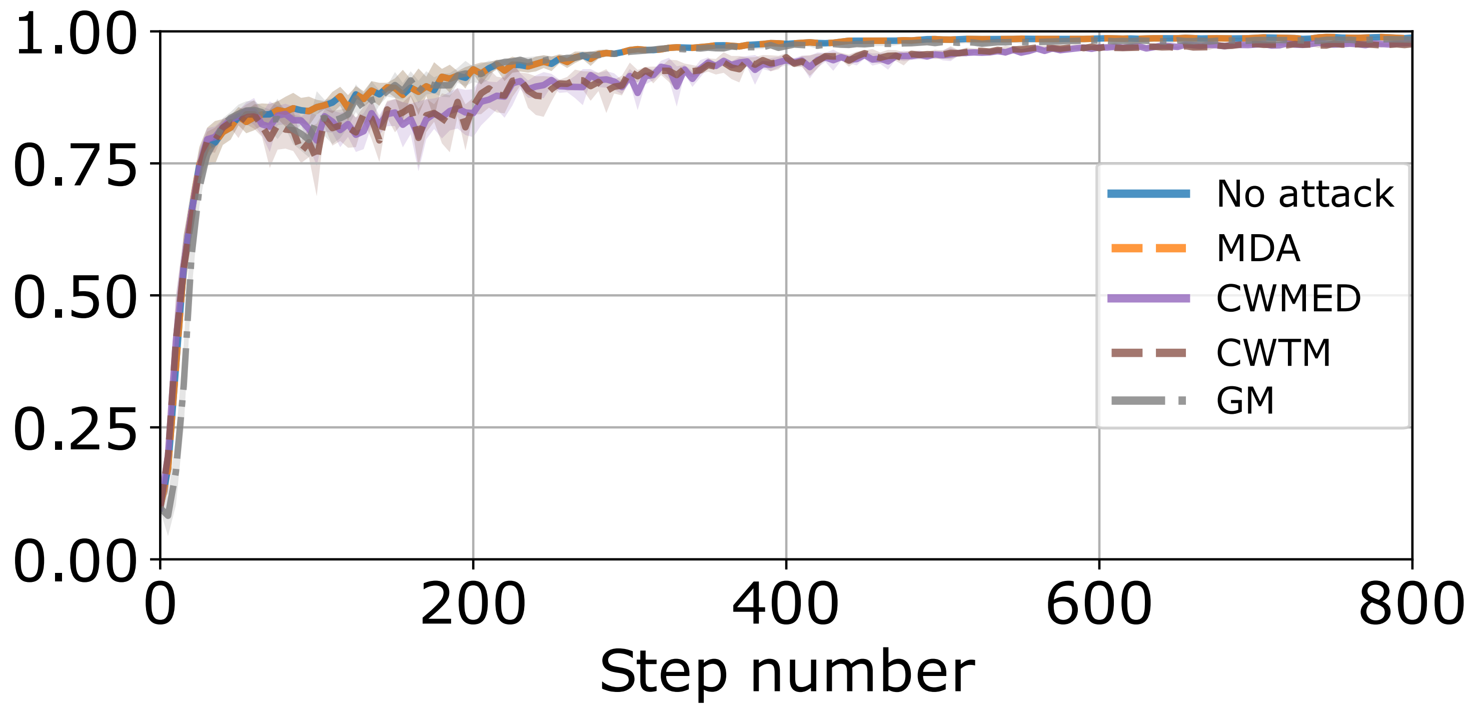
Empirical Evidence

Without Momentum





With Momentum



Label-flipping

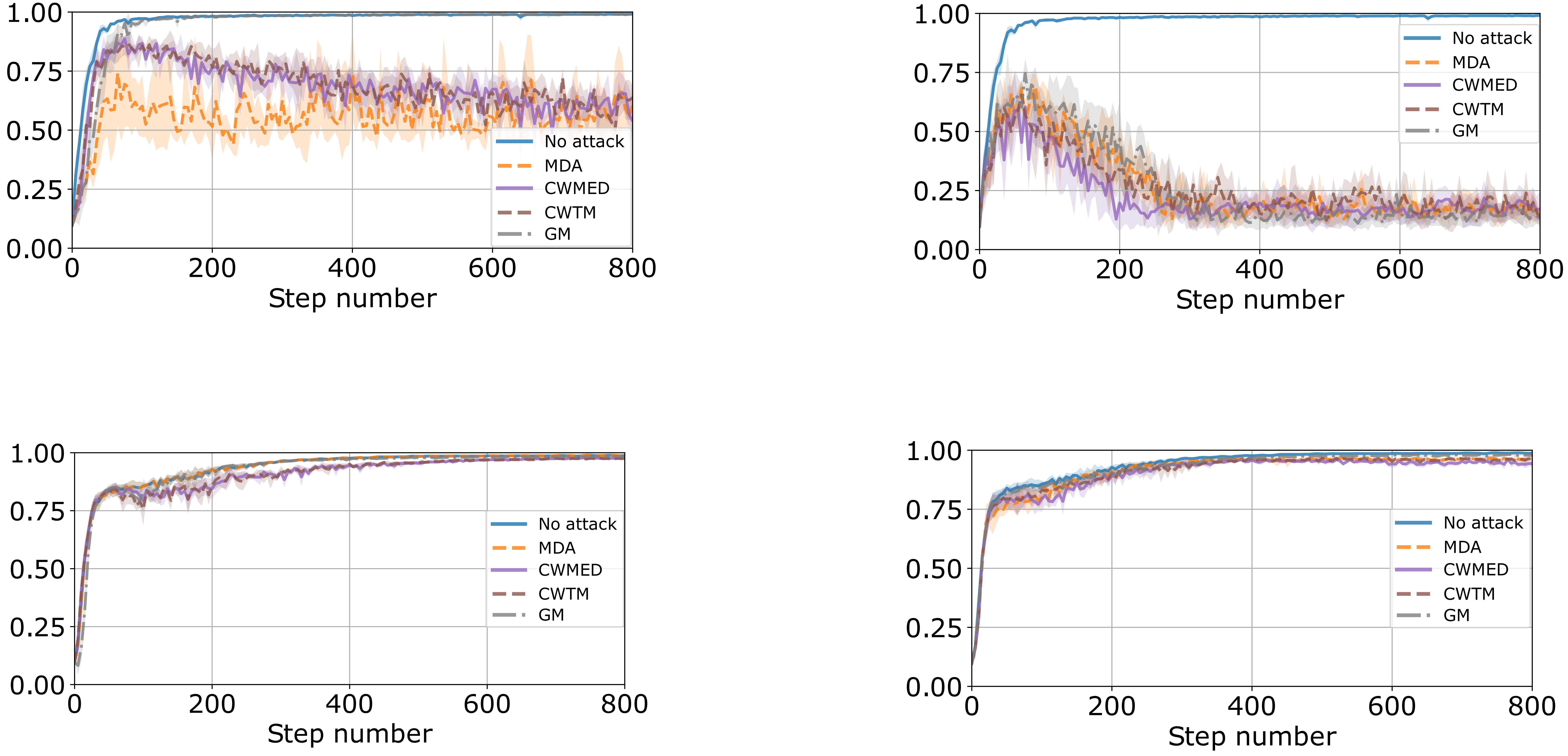
Little is enough *(Baruch et al., 2019)*

* MNIST classification task

Empirical Evidence

Without Momentum

With Momentum



Label-flipping

Little is enough *(Baruch et al., 2019)*

* MNIST classification task

Efficiency + Byzantine Resilience ?

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \ ; \ \nabla Q(\theta) = 0\right)$$

+

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \;;\; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

(f, ϵ)−Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \;;\; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

(f, ϵ)−Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

(f, ϵ)−Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

(f, ϵ)−Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA	TM	Geometric Median	Krum*	Median
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \big(\theta\ ;\ \nabla Q(\theta) = 0\big)$$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA	TM	Geometric Median	Krum*	Median
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

➕

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA ✓	TM ✗	Geometric Median ✗	Krum* ✗	Median ✗
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA ✓	TM ✗	Geometric Median ✗	Krum* ✗	Median ✗
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

(f, ϵ)−Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA ✓	TM ✗	Geometric Median ✗	Krum* ✗	Median ✗
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Median based rules

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta \; ; \; \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

Aggregation:	MDA ✓	TM ✗	Geometric Median ✗	Krum* ✗	Median ✗
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Averaging Component

Median based rules

Efficiency + Byzantine Resilience ?

CAN THE SERVER STILL OBTAIN

$$\theta^* \in \left(\theta\ ;\ \nabla Q(\theta) = 0\right)$$

+

FRACTIONAL WORKLOAD PER NODE

$(f, \epsilon)-$Resilience

$$\epsilon \in \mathcal{O}\left(\sqrt{\left(\frac{1}{n-f} + \lambda^2(n-f)\right)\frac{\sigma^2}{T}}\right)$$

Additional workload

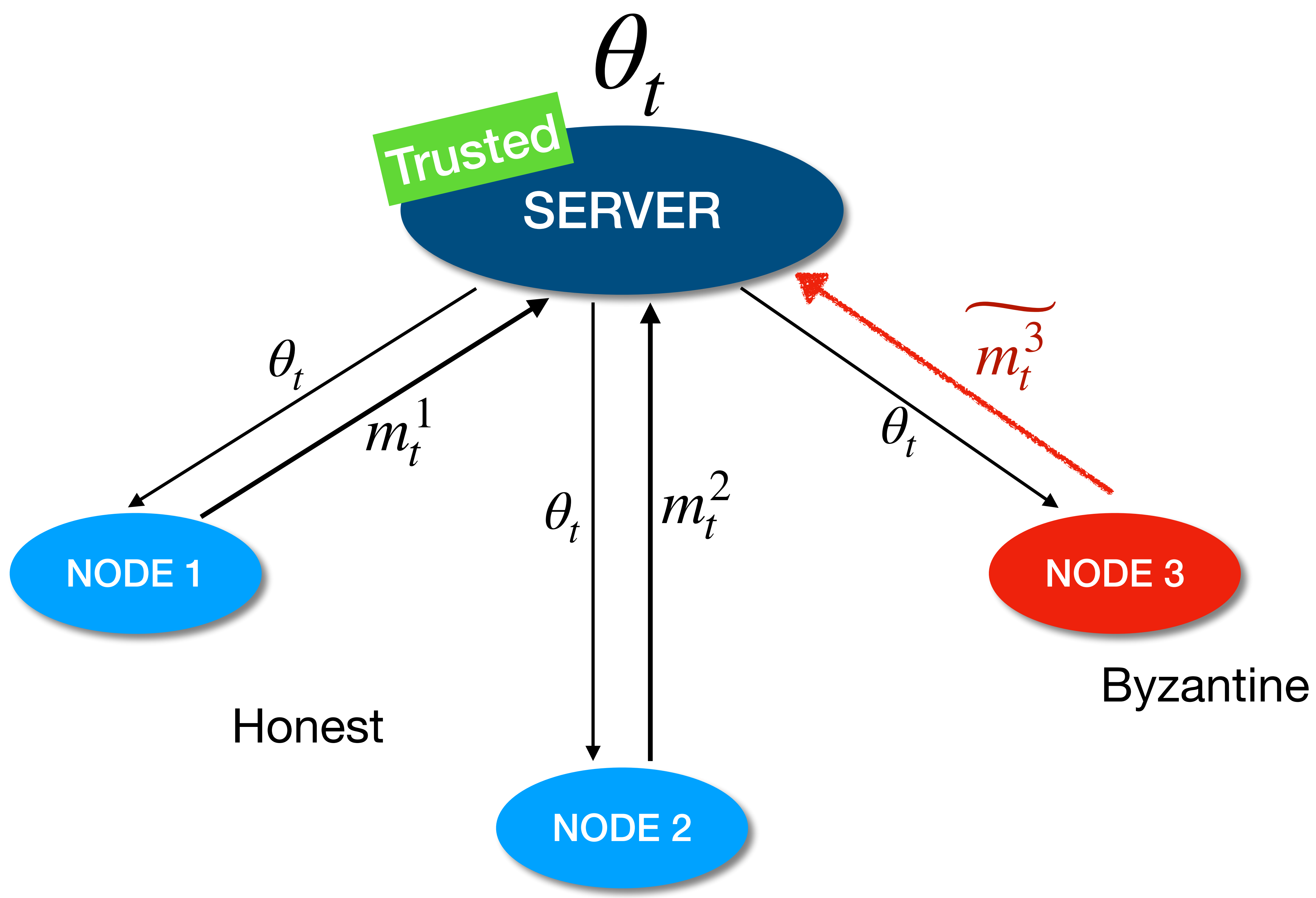
Aggregation:	MDA ✓	TM ✗	Geometric Median ✗	Krum* ✗	Median ✗
$\lambda^2(n-f)$	$\frac{f^2}{n-f}$	f^2	$\frac{(n-f)^3}{(n-2f)n}$	$\frac{(n-f)^2}{n-2f}$	n^2

Averaging Component

Median based rules

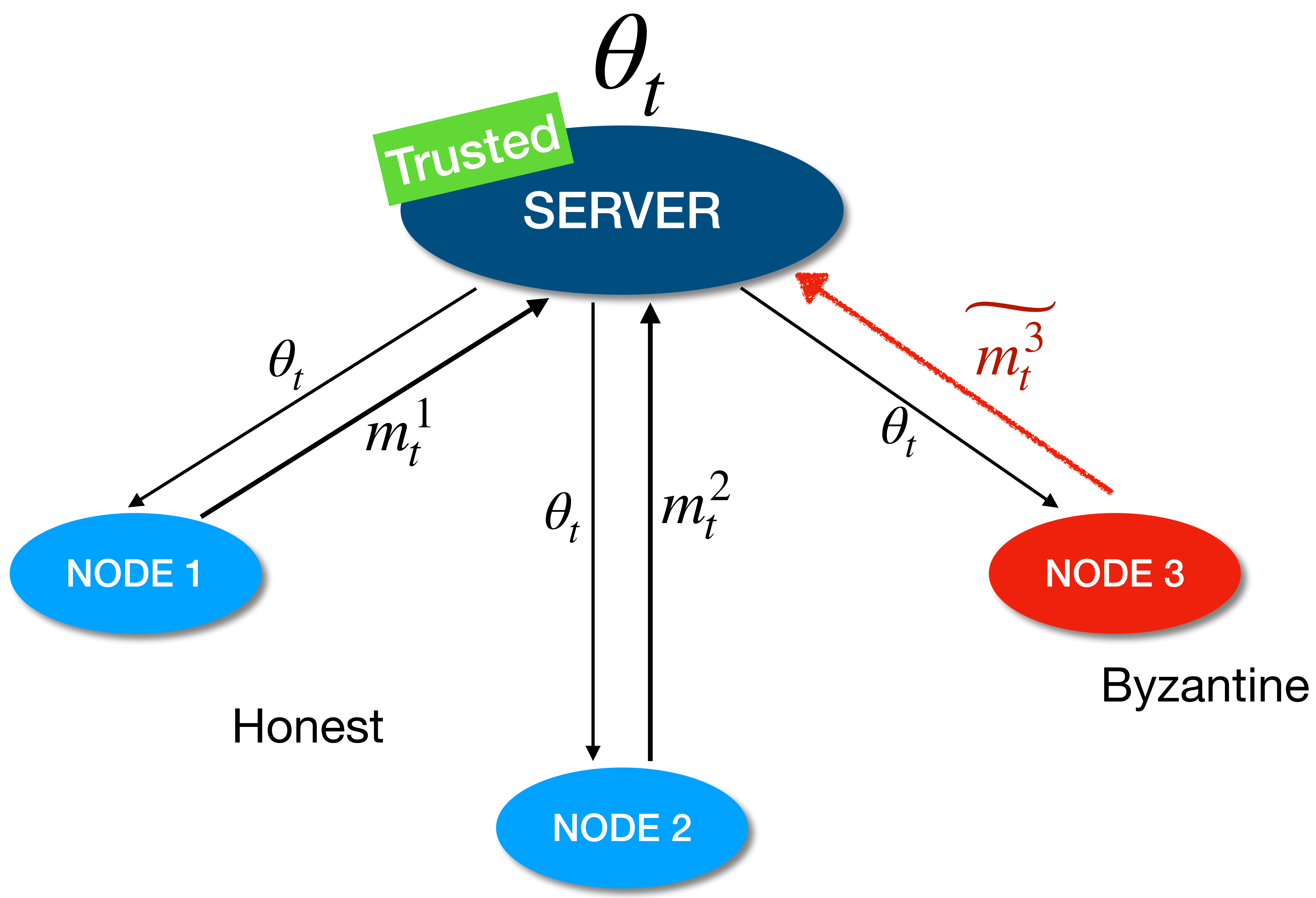
* An extension of Krum, called multi-Krum, uses an averaging component to obtain guarantees comparable to MDA, but is computationally cheaper.

!!Caution!! Momentum may NOT Help Always



!!Caution!! Momentum may NOT Help Always

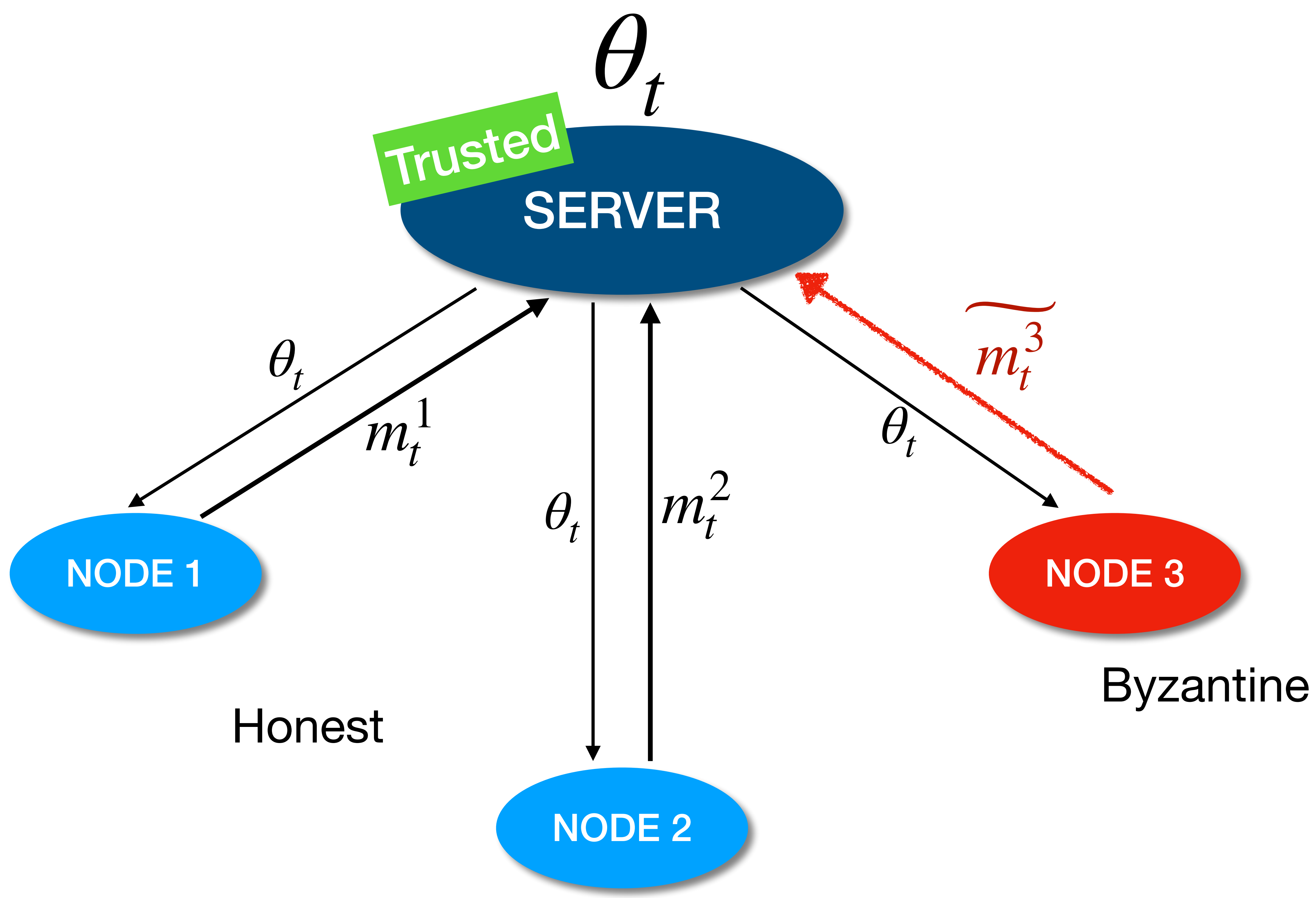
When **data distributions** across nodes **are "very" different**



!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

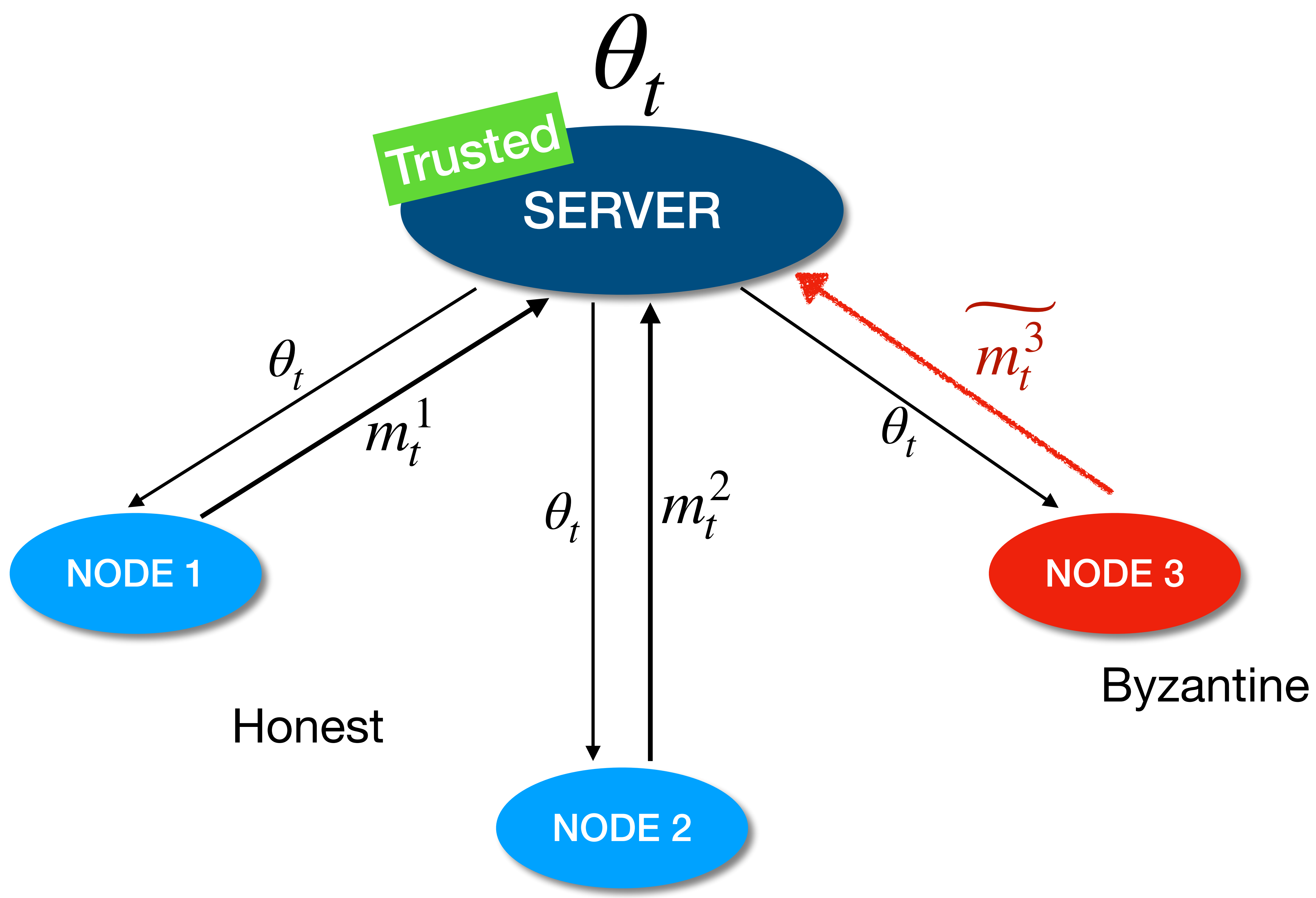


!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathbb{E}_{x \sim \mathcal{D}_i} q(\theta, x) \;\; ;$$

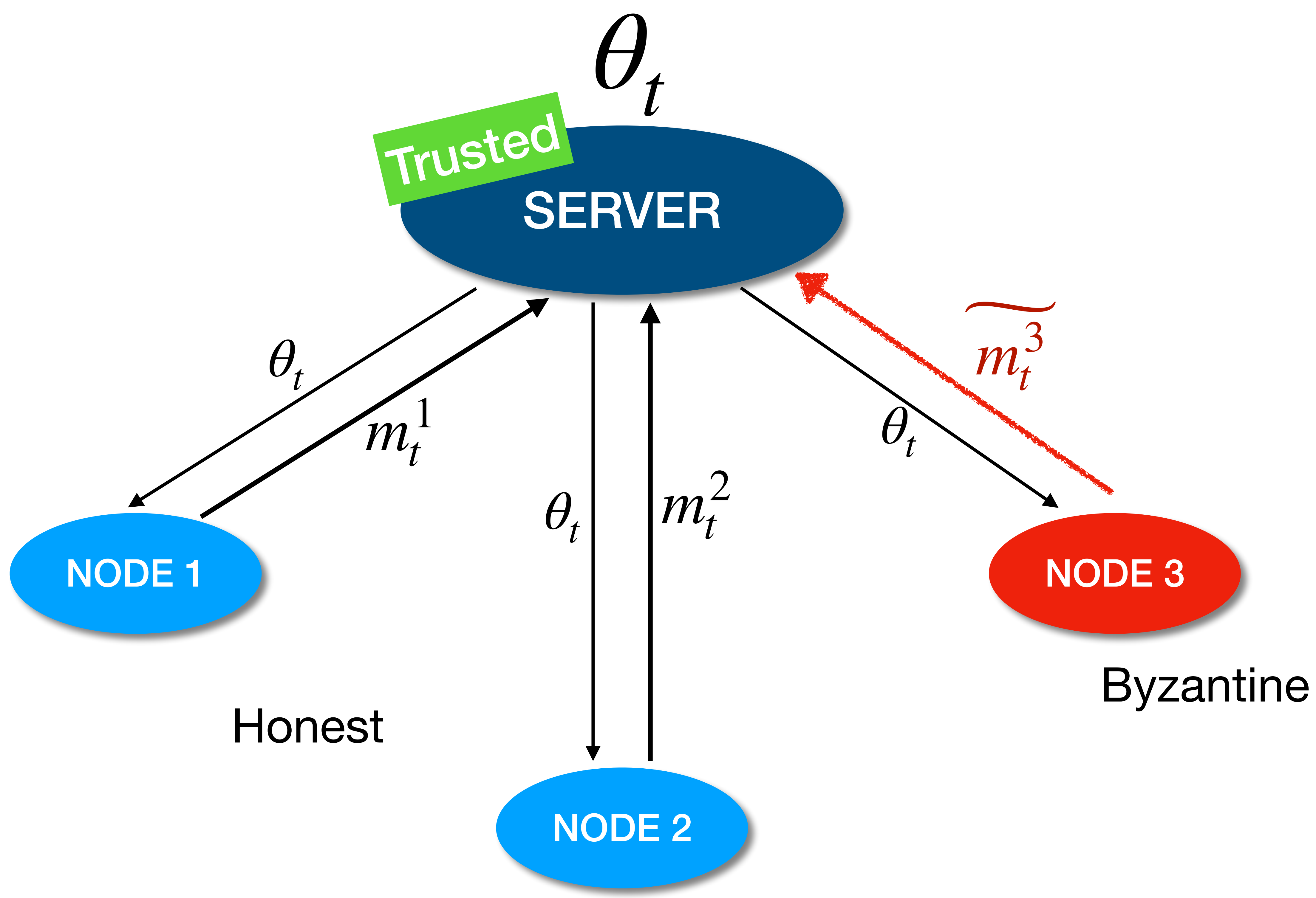


!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}_i} q(\theta, x) \quad ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$



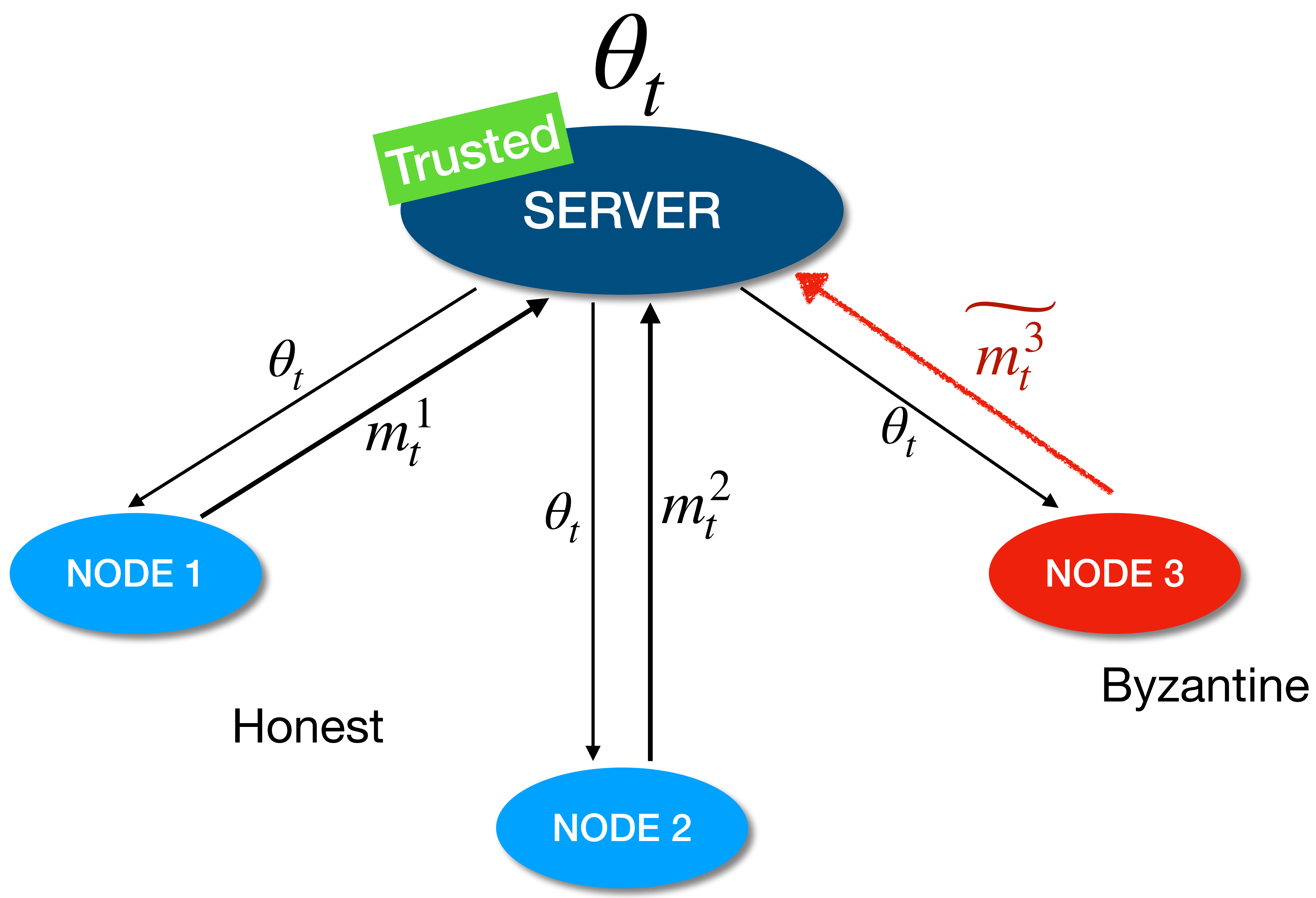
!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathcal{D}_i} q(\theta, x) \ ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute



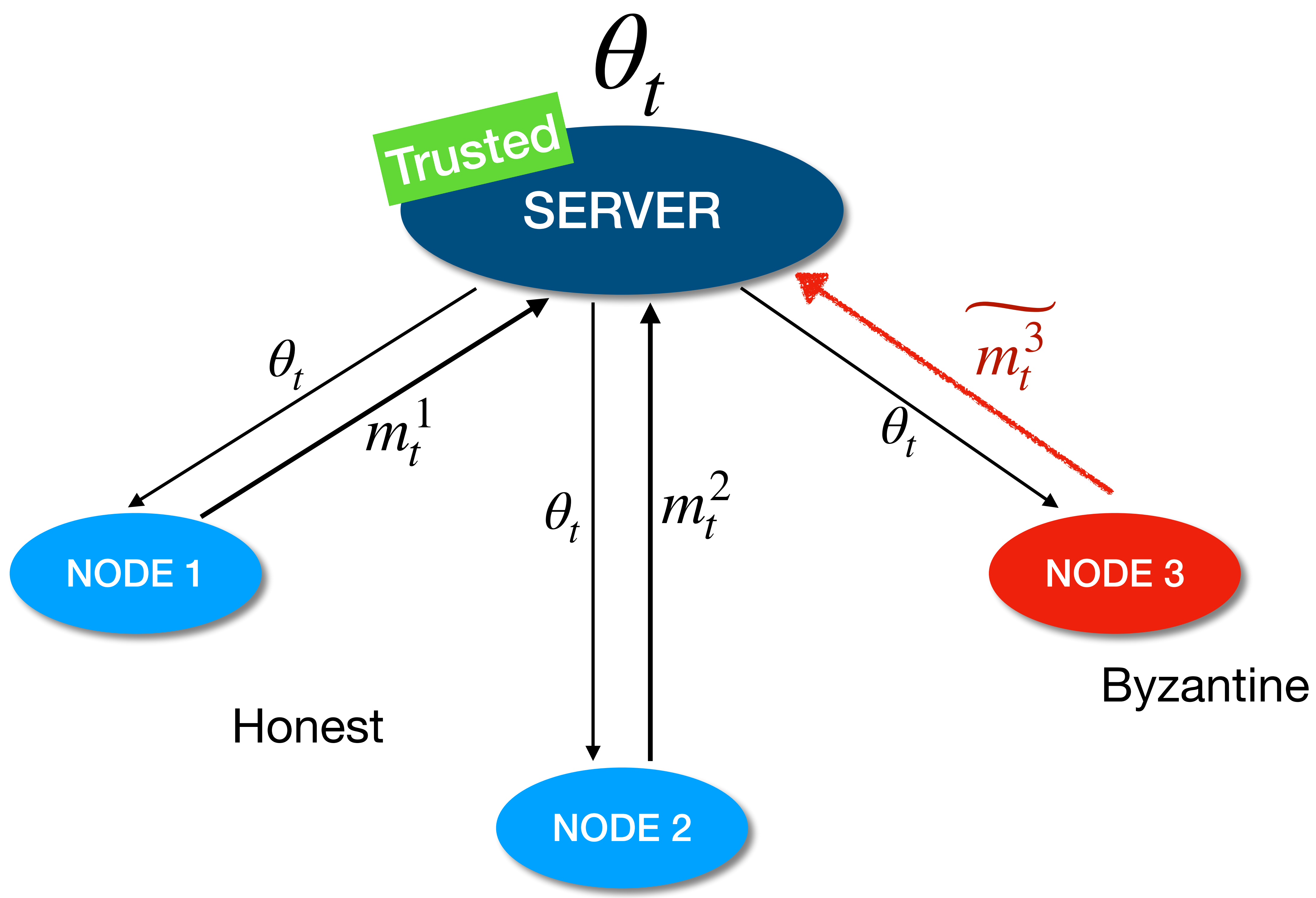
!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathcal{D}_i} q(\theta, x) \;\; ; \;\; g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$



!!Caution!! Momentum may NOT Help Always

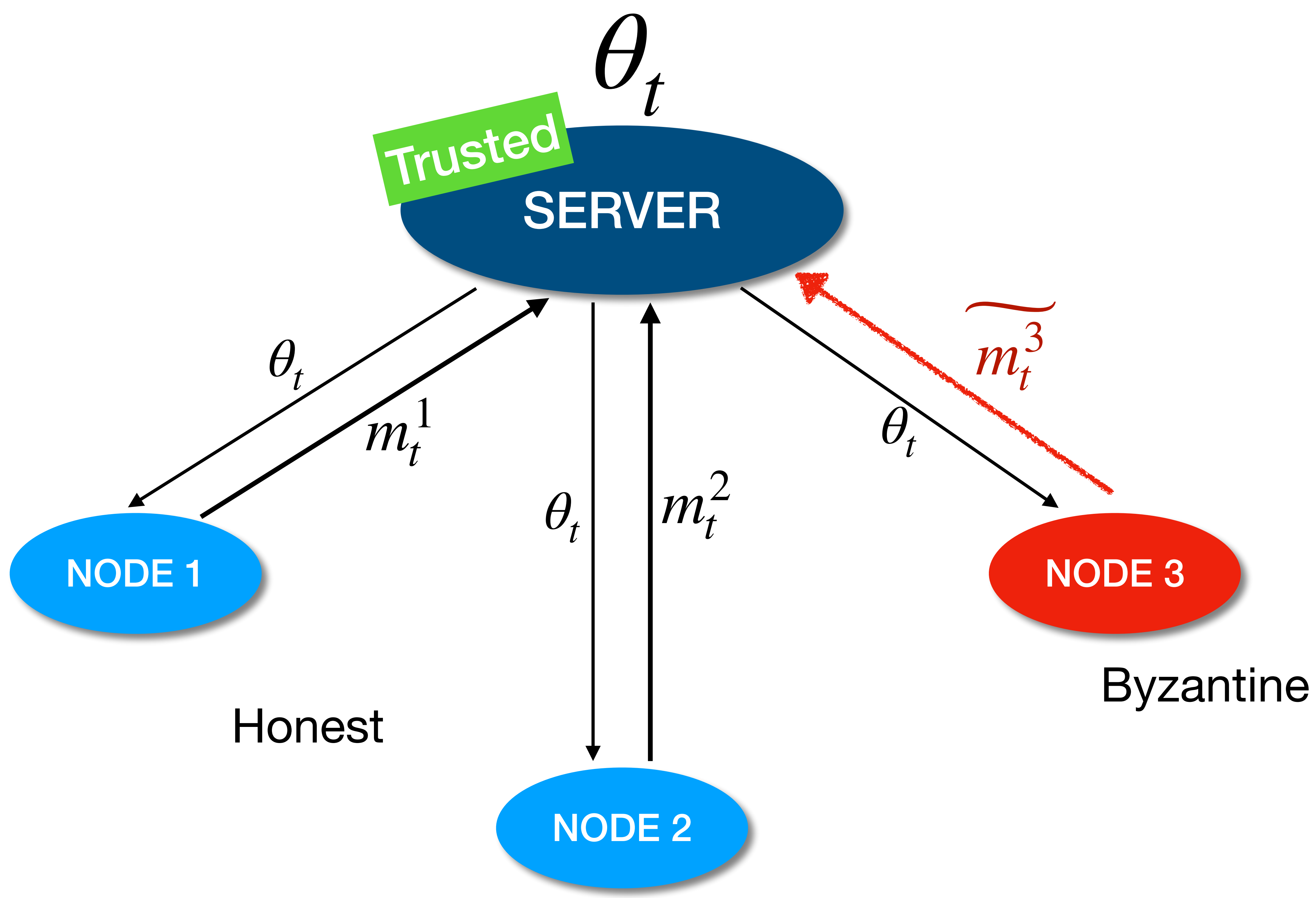
When **data distributions** across nodes **are “very” different**

For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathcal{D}_i} q(\theta, x) \; ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$

Momentum accumulates the heterogeneity



!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

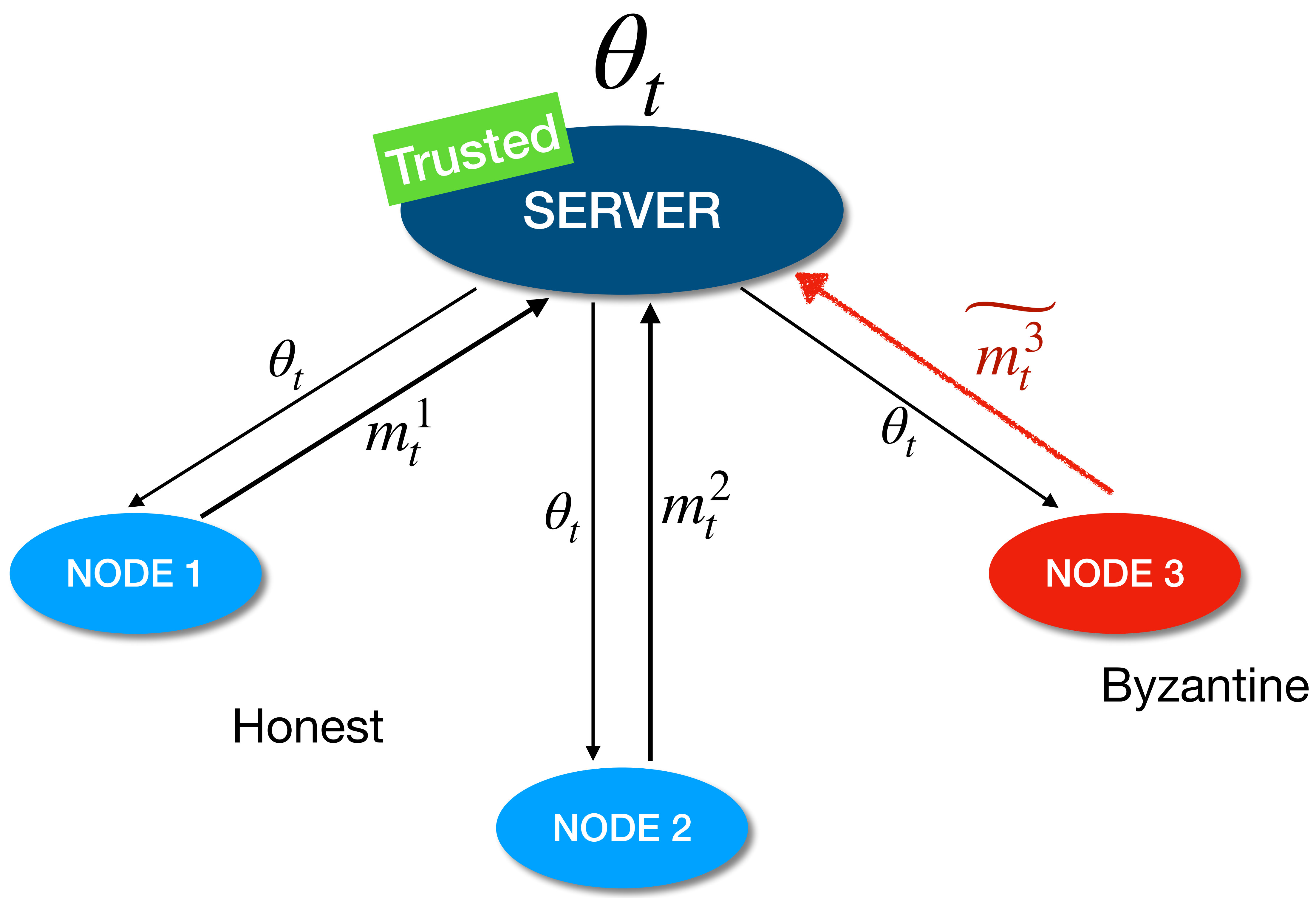
For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}_i} q(\theta, x) \; ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$

Momentum accumulates the heterogeneity

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2 \right] \leq c_1(1-\beta)\sigma^2 + c_2 \sum_{k=1}^{t} \|\nabla Q_i(\theta_k) - \nabla Q_j(\theta_k)\|^2$$



!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

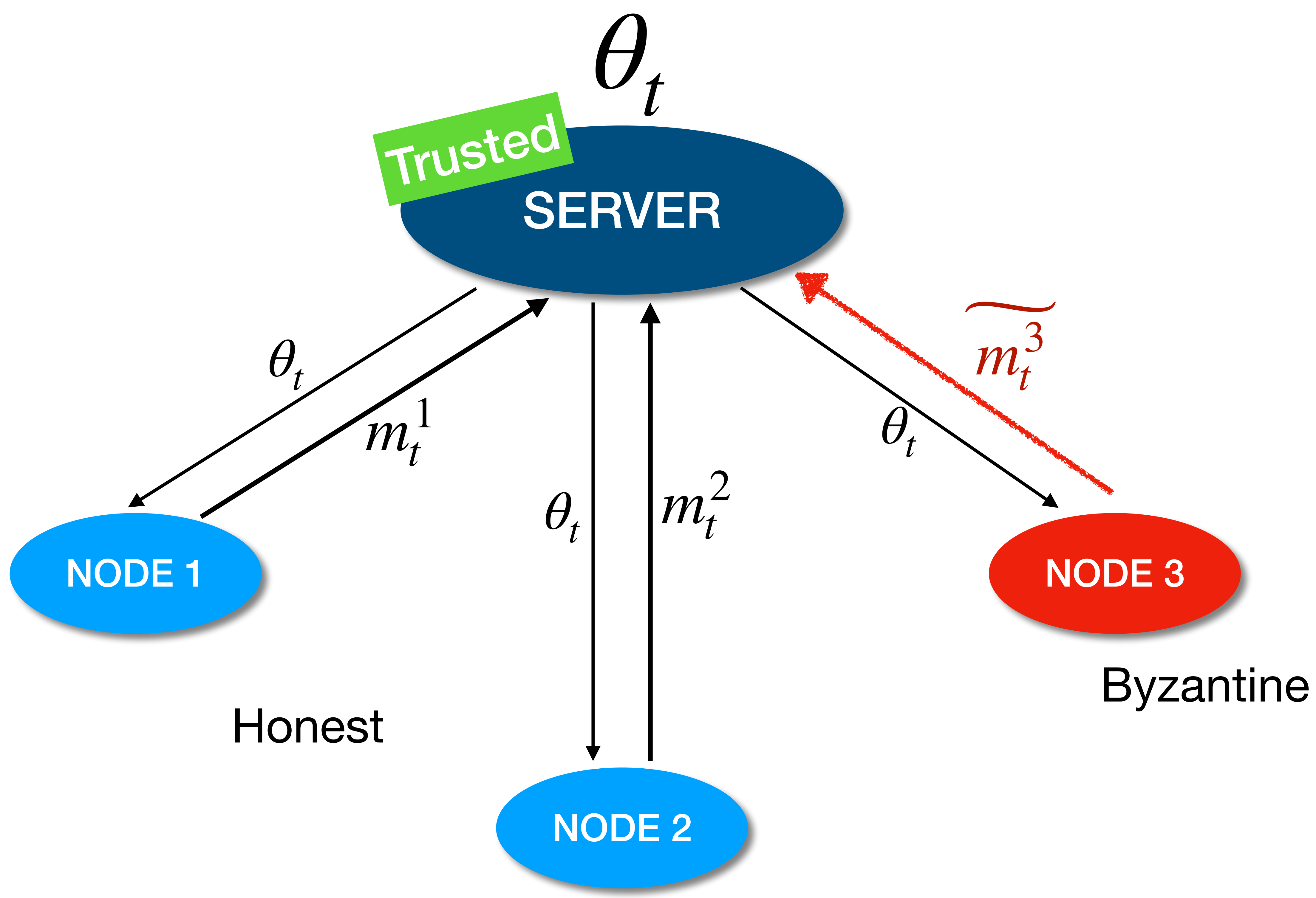
For each honest node :

$$Q_i(\theta) := \mathbb{E}_{x\sim\mathcal{D}_i}\, q\,(\theta, x) \;\; ; \;\; g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$

Momentum accumulates the heterogeneity

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq c_1(1-\beta)\sigma^2 + c_2 \sum_{k=1}^{t} \|\nabla Q_i(\theta_k) - \nabla Q_j(\theta_k)\|^2$$



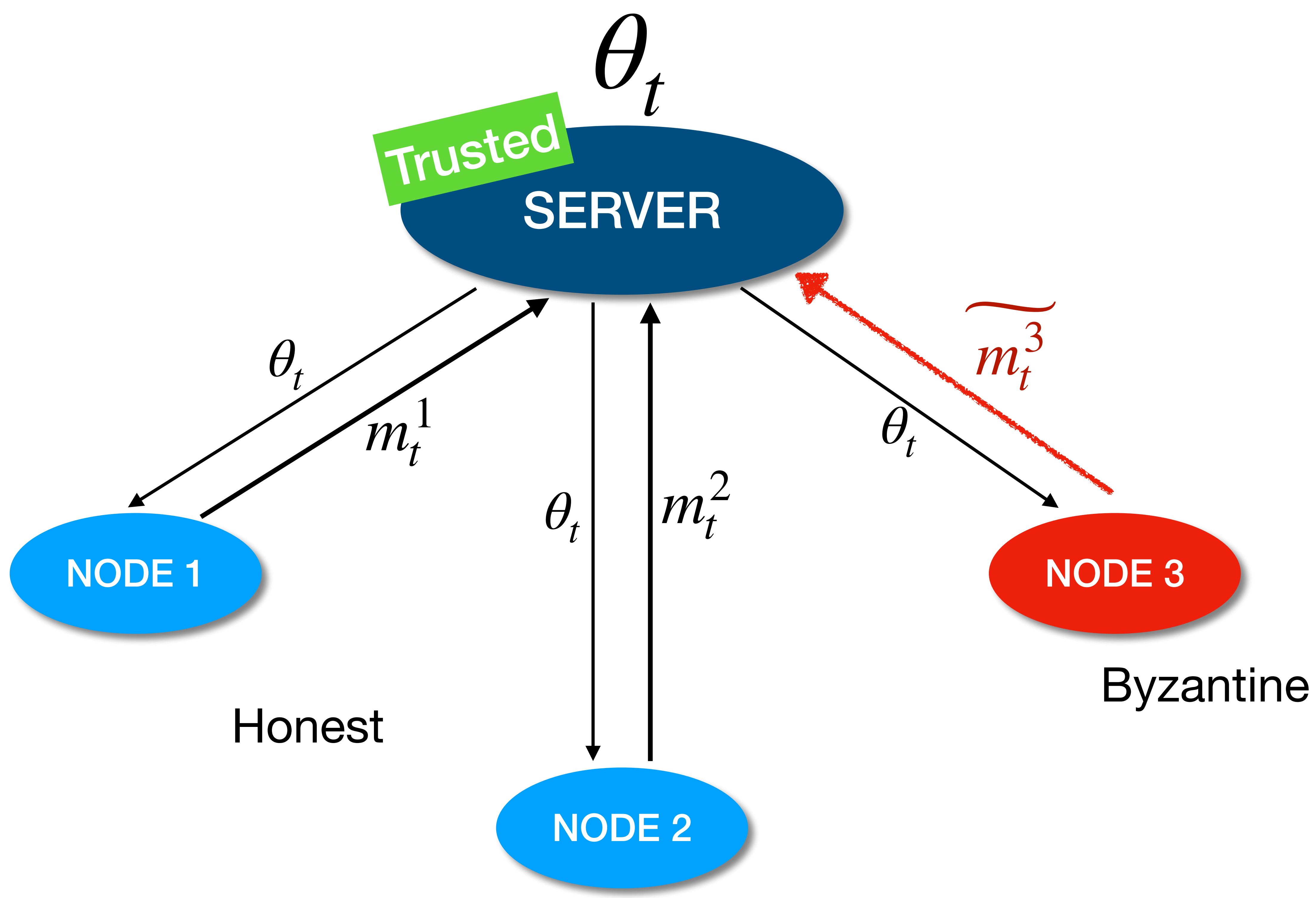
!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathop{\mathbb{E}}_{x \sim \mathscr{D}_i} q(\theta, x) \;\; ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$



Momentum accumulates the heterogeneity

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2 \right] \leq c_1(1-\beta)\sigma^2 + c_2 \sum_{k=1}^{t} \|\nabla Q_i(\theta_k) - \nabla Q_j(\theta_k)\|^2$$

On the other hand,

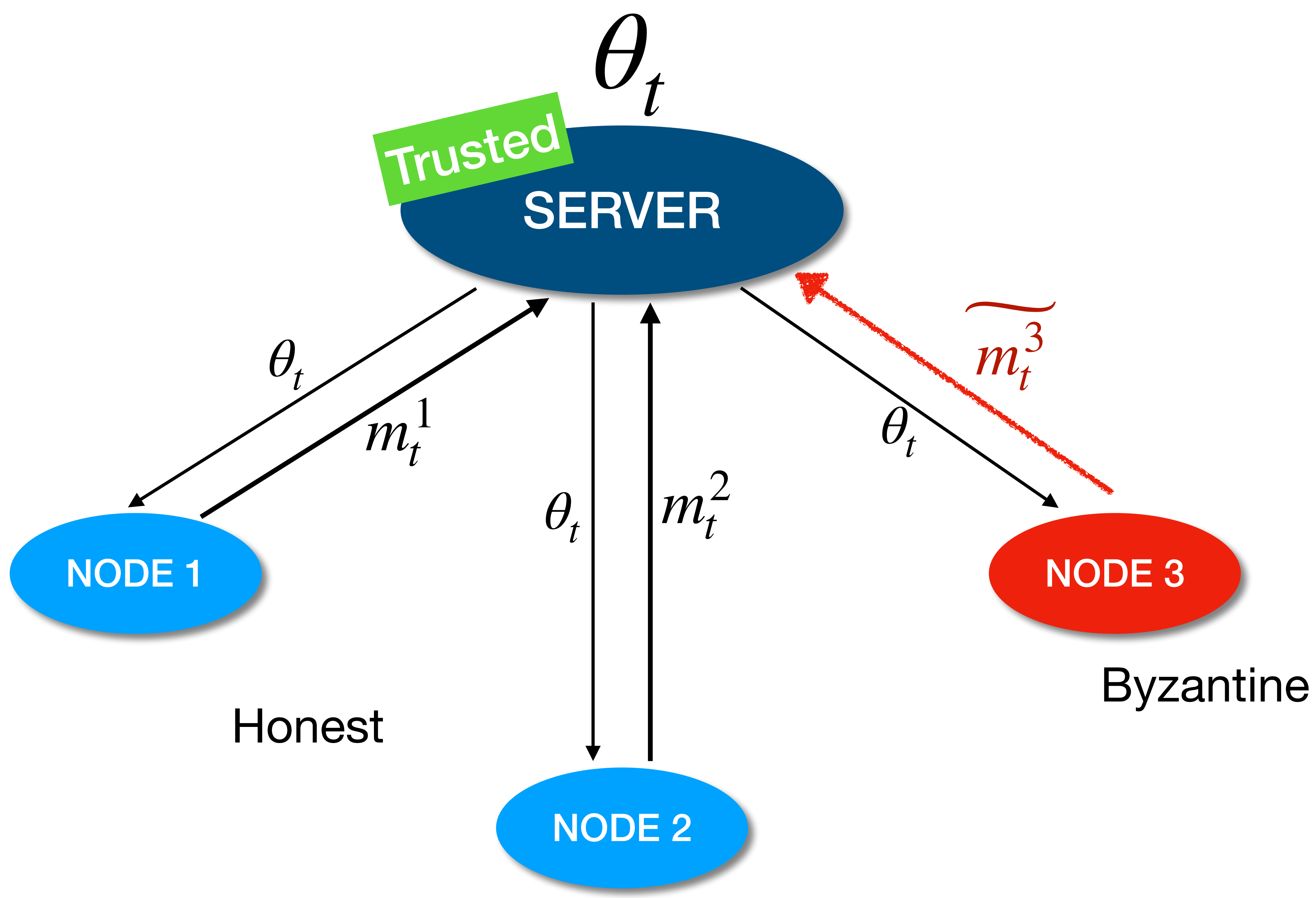
!!Caution!! Momentum may NOT Help Always

When **data distributions** across nodes **are "very" different**

For each honest node :

$$Q_i(\theta) := \mathbb{E}_{x \sim \mathcal{D}_i} q(\theta, x) \; ; \quad g_t^i = \nabla Q_i(\theta_t) + u_t^i$$

The goal is to compute $\theta^* \in \left(\theta \; ; \; \frac{1}{H} \sum_{i \in H} \nabla Q_i(\theta) = 0 \right)$



Momentum accumulates the heterogeneity

$$\mathbb{E}\left[\|m_t^i - m_t^j\|^2\right] \leq c_1(1-\beta)\sigma^2 + c_2 \sum_{k=1}^{t} \|\nabla Q_i(\theta_k) - \nabla Q_j(\theta_k)\|^2$$

On the other hand, $\mathbb{E}\left[\|g_t^i - g_t^j\|^2\right] \leq c_1 \sigma^2 + c_2 \|\nabla Q_i(\theta_k) - \nabla Q_j(\theta_k)\|^2$

What’s Next?!

What’s Next?!

Study the impact of momentum with **heterogeneity**.

What’s Next?!

Study the impact of momentum with **heterogeneity**.

Explore **other variance reduction strategies**, e.g., MVR*.

What's Next?!

Study the impact of momentum with **heterogeneity**.

Explore **other variance reduction strategies**, e.g., MVR*.

* Momentum-based Variance Reduction (MVR) has optimal convergence rate for a first-order stochastic algorithm *(Cutkosky and Orabona, 2019)*

What's Next?!

Study the impact of momentum with **heterogeneity**.

Explore **other variance reduction strategies**, e.g., MVR*.

Improve the robustness condition to guarantee *efficiency*

* Momentum-based Variance Reduction (MVR) has optimal convergence rate for a first-order stochastic algorithm *(Cutkosky and Orabona, 2019)*

What's Next?!

Study the impact of momentum with **heterogeneity**.

Explore **other variance reduction strategies**, e.g., MVR*.

Improve the robustness condition to guarantee *efficiency*

Does use of local momentum improve **privacy**?

* Momentum-based Variance Reduction (MVR) has optimal convergence rate for a first-order stochastic algorithm *(Cutkosky and Orabona, 2019)*

Thanks to ...

Readings

Farhadkhani, Sadegh; Rachid Guerraoui; Nirupam Gupta; Rafael Pinot and John Stephan.
Byzantine Machine Learning Made Easy by Resilient Averaging of Momentums. *ICML 2022.*

El-Mhamdi El-Mahdi; Rachid Guerraoui and Sébastien Rouault.
Distributed momentum for byzantine-resilient learning. *ICLR 2021.*

Karimireddy Sai Praneeth; Lie He and Martin Jaggi.
Learning from history for byzantine robust optimization. *ICML 2021.*

Momentum for Byzantine resilience

Alistarh Dan; Zeyuan Allen-Zhu and Jerry Li.
Byzantine stochastic gradient descent. *NeurIPS 2018.*

El-Mhamdi El-Mahdi; Guerraoui Rachid, and Sébastien Rouault.
The hidden vulnerability of distributed learning in Byzantium. *ICML 2018.*

Chen Yudong; Lili Su and Jiaming Xu.
Distributed statistical machine learning in adversarial settings: Byzantine gradient descent.
Proceedings of the ACM on Measurement and Analysis of Computing Systems 2017.

Notable prior work On Byzantine resilience

Thank You