# On the generalization of learning algorithms that do not converge

Nisha Chandramoorthy[†], Andreas Loukas[a], Khashayar Gatmiry and Stefanie Jegelka

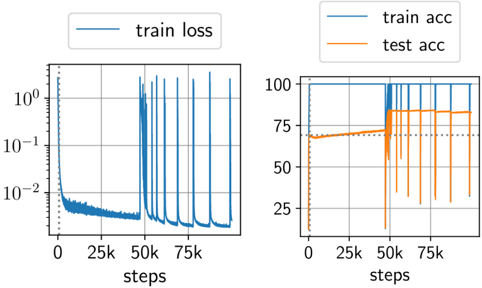
Massachusetts Institute of Technology, [†] *nishac@mit.edu*, [a] Prescient Design, Genentech Roche

https://arxiv.org/abs/2208.07951

October 19, 2022

# Non-converging optimization

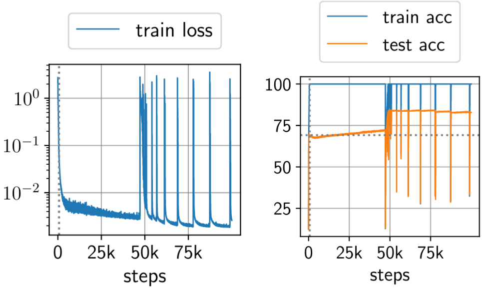
What happens in training beyond the stopping point?




Courtesy: [Lyu Li Arora 2022]. Recent interest [Kong and Tao 2021, Cohen et al 2021, Lobacheva et al 2021, Zhang Li Sra Jadbabaie 2022] in non-converging training algorithms

# Non-converging optimization

What happens in training beyond the stopping point?




Courtesy: [Lyu Li Arora 2022]. Recent interest [Kong and Tao 2021, Cohen et al 2021, Lobacheva et al 2021, Zhang Li Sra Jadbabaie 2022] in non-converging training algorithms

(Q1) How can we *define* and *study* the generalization properties of a non-converging learning algorithm?

(Q2) Can the statistical/ergodic properties of the algorithm *predict* its generalization performance?

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$
- $S$ is a set of $n$ training samples $z_1, \cdots, z_n$ iid according to $\mathcal{D}$

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$
- $S$ is a set of $n$ training samples $z_1, \cdots, z_n$ iid according to $\mathcal{D}$
- $L_S(w) = (1/n) \sum_{z \in S} \ell(z, w)$ is the sample average of the loss $\ell(\cdot, \cdot)$.

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$
- $S$ is a set of $n$ training samples $z_1, \cdots, z_n$ iid according to $\mathcal{D}$
- $L_S(w) = (1/n) \sum_{z \in S} \ell(z, w)$ is the sample average of the loss $\ell(\cdot, \cdot)$.
- $\hat{\nabla} L_S(w_t)$ is the estimate of the weight space gradient of $L_S$.

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$
- $S$ is a set of $n$ training samples $z_1, \cdots, z_n$ iid according to $\mathcal{D}$
- $L_S(w) = (1/n) \sum_{z \in S} \ell(z, w)$ is the sample average of the loss $\ell(\cdot, \cdot)$.
- $\hat{\nabla} L_S(w_t)$ is the estimate of the weight space gradient of $L_S$.

In general, deterministic/stochastic nonlinear dynamics on compact set. No guarantee of convergence to fixed points.

# SGD/GD dynamics on weight space:

$$w_{t+1} = w_t - \eta \hat{\nabla} L_S(w_t),$$

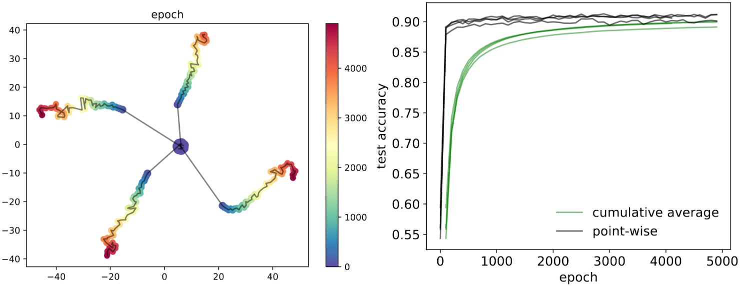
where

- $w_t \in M$ are the weights at time $t \in \mathbb{Z}^+$
- $S$ is a set of $n$ training samples $z_1, \cdots, z_n$ iid according to $\mathcal{D}$
- $L_S(w) = (1/n) \sum_{z \in S} \ell(z, w)$ is the sample average of the loss $\ell(\cdot, \cdot)$.
- $\hat{\nabla} L_S(w_t)$ is the estimate of the weight space gradient of $L_S$.

In general, deterministic/stochastic nonlinear dynamics on compact set. No guarantee of convergence to fixed points. There exist multiple invariant, ergodic distributions on weight space, $M$.

# Convergence of loss time-averages

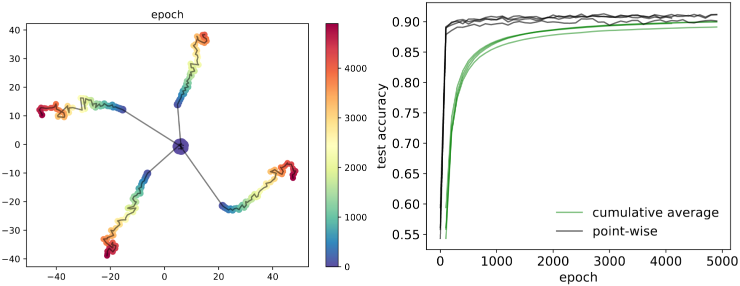
**Assumption 1:** For almost every $w_0$ and every $z$, time-average of $\ell(z, \cdot)$ converges to a constant $\langle \ell_z \rangle_S$, independent of $w_0$.




Orbits of four different initializations of a VGG16 training with SGD.

# Convergence of loss time-averages

**Assumption 1:** For almost every $w_0$ and every $z$, time-average of $\ell(z, \cdot)$ converges to a constant $\langle \ell_z \rangle_S$, independent of $w_0$.




Orbits of four different initializations of a VGG16 training with SGD.

Assumption allows us to extend algorithmic stability to *statistical algorithmic stability* (SAS).

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_{z} \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_{z} \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

**Statistical Algorithmic Stability (SAS):** We say an algorithm is SAS with coefficient $\beta$ if

$$\beta := \sup_{z} \sup_{S,S'} |\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}|.$$

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_{z} \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

**Statistical Algorithmic Stability (SAS):** We say an algorithm is SAS with coefficient $\beta$ if

$$\beta := \sup_{z} \sup_{S,S'} |\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}|.$$

The higher the value of $\beta$, the lower the statistical stability.

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_{z} \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

**Statistical Algorithmic Stability (SAS):** We say an algorithm is SAS with coefficient $\beta$ if

$$\beta := \sup_{z} \sup_{S,S'} |\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}|.$$

The higher the value of $\beta$, the lower the statistical stability. Unlike classical algorithmic stability, SAS

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_z \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

**Statistical Algorithmic Stability (SAS):** We say an algorithm is SAS with coefficient $\beta$ if

$$\beta := \sup_z \sup_{S,S'} |\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}|.$$

The higher the value of $\beta$, the lower the statistical stability. Unlike classical algorithmic stability, SAS

- applies to non-converging learning algorithms

# Statistical Algorithmic Stability

**Classical algorithmic stability [Bousquet and Elisseeff 2002]:**

$$\beta := \sup_z \sup_{S,S'} |\ell(z, w_S^*) - \ell(z, w_{S'}^*)|.$$

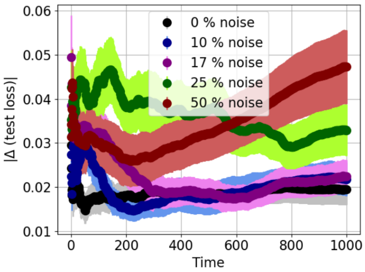
**Statistical Algorithmic Stability (SAS):** We say an algorithm is SAS with coefficient $\beta$ if

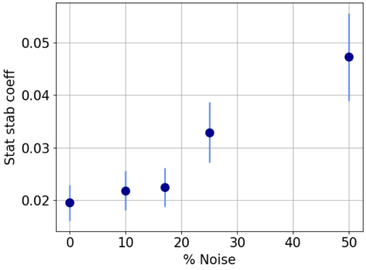
$$\beta := \sup_z \sup_{S,S'} |\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}|.$$

The higher the value of $\beta$, the lower the statistical stability. Unlike classical algorithmic stability, SAS

- applies to non-converging learning algorithms
- is constant on network function/parameter space

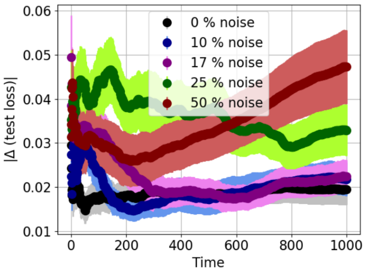
# Numerical approximation of β for SGD on VGG16 model trained on CIFAR10



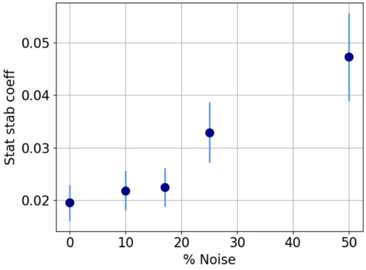



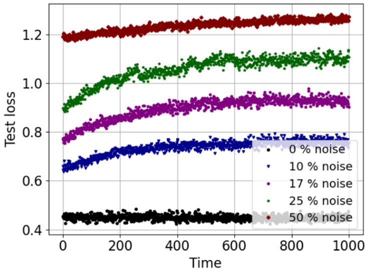

Noisy CIFAR10 labels.
Anticlockwise: Sample mean over 45 $(S, S')$ pairs, with error bars, of time-averaged test loss difference. Lower bound on β with error bars computed as sample mean. Test loss vs. time (epoch).

# Numerical approximation of β for SGD on VGG16 model trained on CIFAR10




Noisy CIFAR10 labels.
Anticlockwise: Sample mean over 45 $(S, S')$ pairs, with error bars, of time-averaged test loss difference.
Lower bound on β with error bars computed as sample mean. Test loss vs. time (epoch).



# Generalization of a non-converging algorithm

- Training error, $\hat{R}_S := (1/n) \sum_{z \in S} \langle \ell_z \rangle_S$

# Generalization of a non-converging algorithm

- Training error, $\hat{R}_S := (1/n) \sum_{z \in S} \langle \ell_z \rangle_S$
- Test/generalization error, $R_S := \mathbb{E}_{z \sim \mathcal{D}} \langle \ell_z \rangle_S$.

# Generalization of a non-converging algorithm

- Training error, $\hat{R}_S := (1/n)\sum_{z\in S}\langle \ell_z \rangle_S$
- Test/generalization error, $R_S := \mathbb{E}_{z\sim\mathcal{D}}\langle \ell_z \rangle_S$.

> Theorem 1 **(SAS implies generalization)** For an algorithm with SAS coefficient $\beta$ and large # of samples $n$, the *generalization gap* = $R_S - \hat{R}_S = \mathcal{O}(\beta\sqrt{n})$ with high probability.

# Generalization of a non-converging algorithm

- Training error, $\hat{R}_S := (1/n) \sum_{z \in S} \langle \ell_z \rangle_S$
- Test/generalization error, $R_S := \mathbb{E}_{z \sim \mathcal{D}} \langle \ell_z \rangle_S$.

Theorem 1 **(SAS implies generalization)** For an algorithm with SAS coefficient $\beta$ and large # of samples $n$, the *generalization gap* = $R_S - \hat{R}_S = \mathcal{O}(\beta\sqrt{n})$ with high probability.

Smaller $\beta \equiv$ more SAS $\implies$ better generalization

# Generalization of a non-converging algorithm

- Training error, $\hat{R}_S := (1/n) \sum_{z \in S} \langle \ell_z \rangle_S$
- Test/generalization error, $R_S := \mathbb{E}_{z \sim \mathcal{D}} \langle \ell_z \rangle_S$.

Theorem 1 **(SAS implies generalization)** For an algorithm with SAS coefficient $\beta$ and large # of samples $n$, the *generalization gap* = $R_S - \hat{R}_S = \mathcal{O}(\beta\sqrt{n})$ with high probability.

Smaller $\beta \equiv$ more SAS $\implies$ better generalization

https://arxiv.org/abs/2208.07951

# Predicting generalization gap from timeseries data

Theorem 2 (**Slower convergence of loss statistics implies larger** $\beta$) Let $\lambda$ be the slowest mixing rate of the transition operators on loss space. Then, the corresponding training algorithm with $n$ samples has SAS coefficient

$$\beta = \mathcal{O}\left(\frac{1}{n}\frac{L_D}{1-\lambda}\right),$$

where $L_D = \sup_w \mathrm{Lip}(\nabla \ell(\cdot, w))$

# Predicting generalization gap from timeseries data

Theorem 2 (**Slower convergence of loss statistics implies larger** $\beta$) Let $\lambda$ be the slowest mixing rate of the transition operators on loss space. Then, the corresponding training algorithm with $n$ samples has SAS coefficient

$$\beta = \mathcal{O}\left(\frac{1}{n}\frac{L_D}{1-\lambda}\right),$$

where $L_D = \sup_w \mathrm{Lip}(\nabla \ell(\cdot, w))$

- use perturbation theory of Markov operators to explain SAS
- as a proxy for the mixing of the loss process, we use auto-correlations in the loss timeseries.

# Predicting generalization gap from timeseries data

Theorem 2 (**Slower convergence of loss statistics implies larger** $\beta$) Let $\lambda$ be the slowest mixing rate of the transition operators on loss space. Then, the corresponding training algorithm with $n$ samples has SAS coefficient

$$\beta = \mathcal{O}\left(\frac{1}{n}\frac{L_D}{1-\lambda}\right),$$

where $L_D = \sup_w \mathrm{Lip}(\nabla \ell(\cdot, w))$

- use perturbation theory of Markov operators to explain SAS
- as a proxy for the mixing of the loss process, we use auto-correlations in the loss timeseries.

https://arxiv.org/abs/2208.07951