Reducing Noise in GAN training with Variance Reduced Extragradient

Tatjana Chavdarova

Research intern at Mila,
PhD student at Idiap research institute & EPFL

Thirty-third Conference on Neural Information Processing Systems, Vancouver, Canada

December 12, 2019

Reducing Noise in GAN Training with Variance Reduced Extragradient

Tatjana Chavdarova *

Gauthier Gidel *

François Fleuret

Simon Lacoste-Julien

* Equal contribution

Single Objective Vs. Two-Objective optimization

- Standard supervised learning:

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$$

- GANs [Goodfellow et al., 2014]: Different optimization problem (***minimax***).

Single Objective Vs. Two-Objective optimization

- Standard supervised learning:

$$\min_{\theta} \mathcal{L}(\theta)$$

- GANs [Goodfellow et al., 2014]: Different optimization problem (***minimax***).

$$\min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D)$$

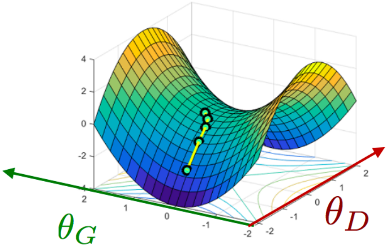


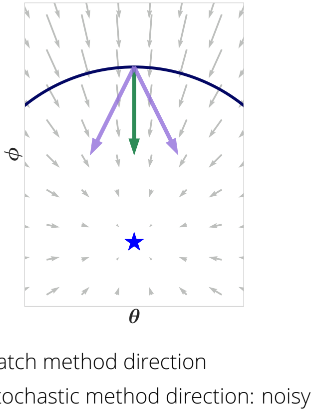
Image source: Vaishnavh Nagarajan

Terminology: "Noise"–noisy gradient estimates

Induced by stochasticity

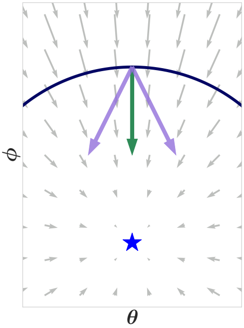
- Using sub-samples (mini-batches) of the full dataset to update the parameters
- Variance Reduced (VR) Gradient: optimization methods that reduce such noise

Minimization: Single-objective

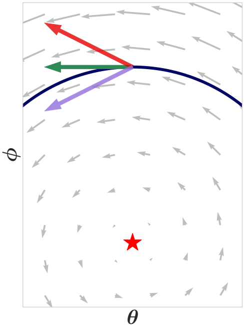


MOTIVATION: VARIANCE REDUCTION FOR GAMES

- INTUITIVELY: **MINIMIZATION** *VS.* **GAME** (NOISE FROM STOCHASTIC GRADIENT)
- EMPIRICALLY:



Minimization
Noisy gradient: "approximately" correct



Game
Noisy gradient: sometimes "opposite"

Motivation: Variance Reduction for Games

- Intuitively: **Minimization** *vs.* **Game** (Noise from Stochastic gradient)
- Empirically:
  - **BigGAN** [Brock et al., 2019]: "Increased batch size significantly improves performances"
  - Empirically tuned hyper-parameters of Adam [Kingma and Ba, 2015] which effectively use solely the variance reduction term

Variance Reduced Gradient Methods

Variance Reduced Estimate of the Gradient

Based on the finite sum assumption: $\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})$.
Epoch based algorithm:

- Save the full gradient $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}})$ and the snapshot $\boldsymbol{\omega}^{\mathcal{S}}$.
- For one epoch use the update rule:

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \eta\Big[\underbrace{\nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})}_{\text{Stochastic gradient}} \underbrace{+\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}) - \nabla\mathcal{L}\left(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}\right)}_{\text{correction using saved past iterate}}\Big]$$

- Requires 2 stochastic gradients (at the current point and at the snapshot).
- If $\boldsymbol{\omega}^{\mathcal{S}}$ is close to $\boldsymbol{\omega}$ $\rightarrow$ close to full batch gradient $\rightarrow$ small variance.
- Full batch gradient is expensive but tractable, *e.g.*, compute it once per pass.

Variance Reduced Estimate of the Gradient

Based on the finite sum assumption: $\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega})$.
Epoch based algorithm:

- Save the full gradient $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega}^{\mathcal{S}})$ and the snapshot $\boldsymbol{\omega}^{\mathcal{S}}$.
- For one epoch use the update rule:

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \eta\Big[\underbrace{\nabla\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega})}_{\text{Stochastic gradient}} \underbrace{+ \frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega}^{\mathcal{S}}) - \nabla\mathcal{L}\left(\boldsymbol{x}_i, \boldsymbol{\omega}^{\mathcal{S}}\right)}_{\text{correction using saved past iterate}} \Big]$$

- Requires 2 stochastic gradients (at the current point and at the snapshot).
- If $\boldsymbol{\omega}^{\mathcal{S}}$ is close to $\boldsymbol{\omega}$ $\rightarrow$ close to full batch gradient $\rightarrow$ small variance.
- Full batch gradient is expensive but tractable, *e.g.*, compute it once per pass.

Variance Reduced Estimate of the Gradient

Based on the finite sum assumption: $\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})$.
Epoch based algorithm:

- Save the full gradient $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}})$ and the snapshot $\boldsymbol{\omega}^{\mathcal{S}}$.
- For one epoch use the update rule:

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \eta\Big[\underbrace{\nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})}_{\text{Stochastic gradient}} + \underbrace{\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}) - \nabla\mathcal{L}\left(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}\right)}_{\text{correction using saved past iterate}}\Big]$$

- Requires 2 stochastic gradients (at the current point and at the snapshot).
- If $\boldsymbol{\omega}^{\mathcal{S}}$ is close to $\boldsymbol{\omega}$ $\rightarrow$ close to full batch gradient $\rightarrow$ small variance.
- Full batch gradient is expensive but tractable, *e.g.*, compute it once per pass.

Variance Reduced Estimate of the Gradient

Based on the finite sum assumption: $\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})$.
Epoch based algorithm:

- Save the full gradient $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}})$ and the snapshot $\boldsymbol{\omega}^{\mathcal{S}}$.
- For one epoch use the update rule:

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \eta\Big[\underbrace{\nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega})}_{\text{Stochastic gradient}} \underbrace{+\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}) - \nabla\mathcal{L}\left(\boldsymbol{x}_i,\boldsymbol{\omega}^{\mathcal{S}}\right)}_{\text{correction using saved past iterate}}\Big]$$

- Requires 2 stochastic gradients (at the current point and at the snapshot).
- If $\boldsymbol{\omega}^{\mathcal{S}}$ is close to $\boldsymbol{\omega}$ $\rightarrow$ close to full batch gradient $\rightarrow$ small variance.
- Full batch gradient is expensive but tractable, *e.g.*, compute it once per pass.

Extragradient

Idea: *Anticipate what the next player would do*

Two players $\boldsymbol{\theta}$, $\boldsymbol{\varphi}$, and a "lookahead step" at $t+\frac{1}{2}$:

$$\text{Extrapolation:} \begin{cases} \boldsymbol{\theta}_{t+1/2} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_G(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \boldsymbol{\varphi}_{t+1/2} = \boldsymbol{\varphi}_t - \eta \nabla_{\boldsymbol{\varphi}} \mathcal{L}_D(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{cases}$$

$$\text{Update:} \begin{cases} \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_G(\boldsymbol{\theta}_{t+1/2}, \boldsymbol{\varphi}_{t+1/2}) \\ \boldsymbol{\varphi}_{t+1} = \boldsymbol{\varphi}_t - \eta \nabla_{\boldsymbol{\varphi}} \mathcal{L}_D(\boldsymbol{\theta}_{t+1/2}, \boldsymbol{\varphi}_{t+1/2}) \end{cases}$$

SVRE: Stochastic Variance-Reduced Extragradient

SVRE: Variance reduction + Extragradient

Pseudo–algorithm

1. Save snapshot $\boldsymbol{\omega}^{\mathcal{S}} \leftarrow \boldsymbol{\omega}_t$ and compute $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega}^{\mathcal{S}})$.
2. For i in $1, \ldots,$ `epoch_length`:
   - 2.1 Compute $\omega_{t+\frac{1}{2}}$ with variance reduced gradients at ω_t.
   - 2.2 Compute ω_{t+1} with variance reduced gradients at $\omega_{t+\frac{1}{2}}$.
   - 2.3 $t \leftarrow t + 1$
3. Repeat until convergence.

SVRE: Variance reduction + Extragradient

Pseudo–algorithm

1. Save snapshot $\omega^{\mathcal{S}} \leftarrow \omega_t$ and compute $\frac{1}{n}\sum_i \nabla \mathcal{L}(x_i, \omega^{\mathcal{S}})$.
2. For i in $1, \ldots,$ `epoch_length`:
   2.1 Compute $\omega_{t+\frac{1}{2}}$ with variance reduced gradients at ω_t.
   2.2 Compute ω_{t+1} with variance reduced gradients at $\omega_{t+\frac{1}{2}}$.
   2.3 $t \leftarrow t+1$
3. Repeat until convergence.

SVRE: Variance reduction + Extragradient

Pseudo–algorithm

1. Save snapshot $\omega^{\mathcal{S}} \leftarrow \omega_t$ and compute $\frac{1}{n}\sum_i \nabla \mathcal{L}(\boldsymbol{x}_i, \omega^{\mathcal{S}})$.
2. For i in $1, \dots,$ `epoch_length`:
   2.1 Compute $\boldsymbol{\omega}_{t+\frac{1}{2}}$ with variance reduced gradients at $\boldsymbol{\omega}_t$.
   2.2 Compute ω_{t+1} with variance reduced gradients at $\omega_{t+\frac{1}{2}}$.
   2.3 $t \leftarrow t + 1$
3. Repeat until convergence.

SVRE: Variance reduction + Extragradient

Pseudo–algorithm

1. Save snapshot $\omega^{\mathcal{S}} \leftarrow \omega_t$ and compute $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i, \omega^{\mathcal{S}})$.
2. For i in $1, \ldots,$ `epoch_length`:
   2.1 Compute $\boldsymbol{\omega}_{t+\frac{1}{2}}$ with variance reduced gradients at $\boldsymbol{\omega}_t$.
   2.2 Compute $\boldsymbol{\omega}_{t+1}$ with variance reduced gradients at $\boldsymbol{\omega}_{t+\frac{1}{2}}$.
   2.3 $t \leftarrow t+1$
3. Repeat until convergence.

SVRE: Variance reduction + Extragradient

Pseudo–algorithm

1. Save snapshot $\omega^{\mathcal{S}} \leftarrow \omega_t$ and compute $\frac{1}{n}\sum_i \nabla \mathcal{L}(\boldsymbol{x}_i, \omega^{\mathcal{S}})$.
2. For i in $1, \ldots,$ `epoch_length`:
   2.1 Compute $\boldsymbol{\omega}_{t+\frac{1}{2}}$ with variance reduced gradients at $\boldsymbol{\omega}_t$.
   2.2 Compute $\boldsymbol{\omega}_{t+1}$ with variance reduced gradients at $\boldsymbol{\omega}_{t+\frac{1}{2}}$.
   2.3 $t \leftarrow t+1$
3. Repeat until convergence.

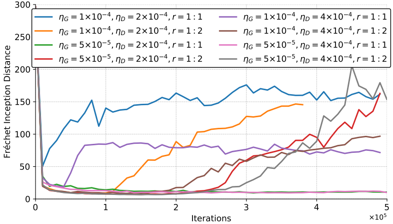
SVRE: Variance reduction + Extragradient
Pseudo–algorithm

1. Save snapshot $\boldsymbol{\omega}^{\mathcal{S}} \leftarrow \boldsymbol{\omega}_t$ and compute $\frac{1}{n}\sum_i \nabla\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{\omega}^{\mathcal{S}})$.
2. For i in $1, \ldots,$ `epoch_length`:
   2.1 Compute $\omega_{t+\frac{1}{2}}$ with variance reduced gradients at ω_t.
   2.2 Compute ω_{t+1} with variance reduced gradients at $\omega_{t+\frac{1}{2}}$.
   2.3 $t \leftarrow t+1$
3. Repeat until convergence.

SVRE yields the fastest convergence rate for strongly convex stochastic game optimization in the literature.

SVRE: EXPERIMENTS

Stochastic baseline

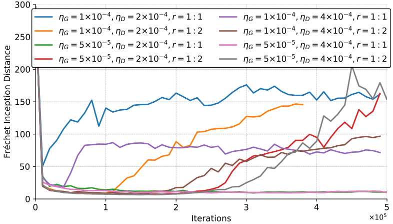


– Always diverges.
– Many hyperparameters ($\eta_G, \eta_D, \beta_1, \gamma, r$).
+ if convergence $\rightarrow$ fast

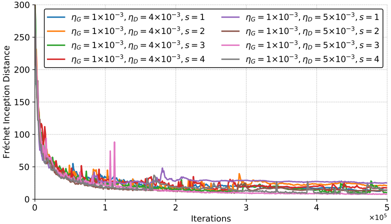
Experiments

SVRE yields stable GAN optimization

Stochastic baseline



SVRE



– Always diverges.

– Many hyperparameters ($\eta_G, \eta_D, \beta_1, \gamma, r$).

+ if convergence $\rightarrow$ fast

+ Does not diverge.

+ fewer hyperparameters (omits β_1, γ, r)

– slower for very deep nets.

SVRE: Takeaways

SVRE: Takeaways

- Controlling variance is more critical for games (could be reason behind success of *Adam* on GANs)
- SVRE: combines Extragradient and variance reduction
- Best convergence rate (under some assumptions) for large class of games
- Good stability properties

Thanks!

REFERENCES I

A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In ICLR, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In NIPS, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In ICLR, 2015.