Corner Gradient Descent

Provable acceleration of power-law convergence of SGD[1]

Dmitry Yarotsky

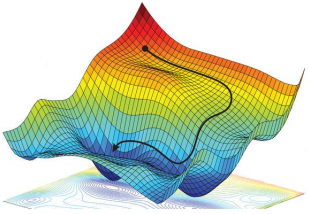
ICLR 2026

[1] https://openreview.net/forum?id=nOXCfldhD9 (ICLR 2026)

Gradient descent

Basic algorithm for learning (optimizing) neural networks and other predictive models

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla L(\mathbf{w}_t)$$



(https://reconsider.news/2018/05/09/ai-researchers-allege-machine-learning-alchemy/)

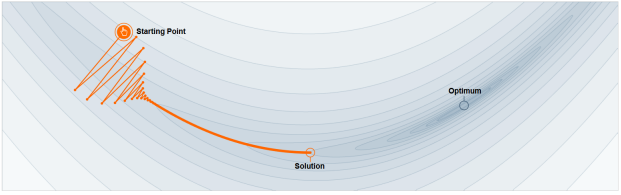
Stochastic Gradient Descent (SGD): instead of exact loss $L(\mathbf{w}_t)$ use its estimates $L_{B_t}(\mathbf{w}_t)$ computed over mini-batches B_t

Heavy Ball[2]

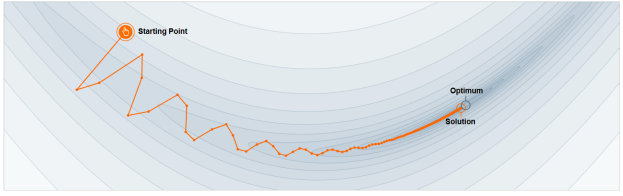
(S)GD can be accelerated by exploiting "momentum vector" $\mathbf{u}_t$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{u}_{t+1}, \quad \mathbf{u}_{t+1} = \alpha \nabla L(\mathbf{w}_t) + \beta \mathbf{u}_t$$

Without momentum



With momentum



(https://distill.pub/2017/momentum/)

[2] Polyak (1964)

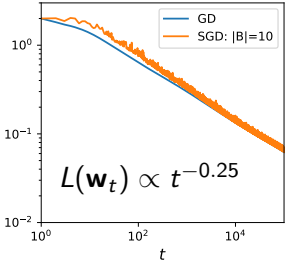
Power laws

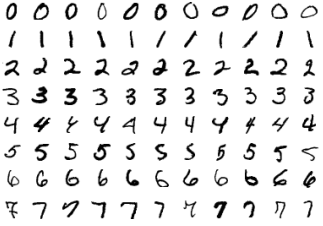
Optimization of neural networks: a high-dimensional and ill-conditioned problem

Theoretically and practically, optimization trajectories in these problems are often well described by **power laws**:

$$L(\mathbf{w}_t) \propto t^{-\zeta}$$

For example, for MNIST hand-written digit classification $\zeta \approx 0.25$



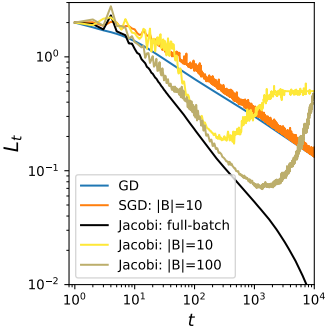


Acceleration of convergence exponents

For **non-stochastic** GD, momentum with a Jacobi schedule $\beta_t \sim 1 - \frac{const}{t}$ allows to **double** the convergence exponent[3]:

$$L(\mathbf{w}_t) = O(t^{-\mathbf{2}\zeta})$$

But in **Stochastic** GD such acceleration only works for a limited number of iterations; after that optimization **diverges**



The challenge: Can we achieve a stable acceleration for Stochastic GD?

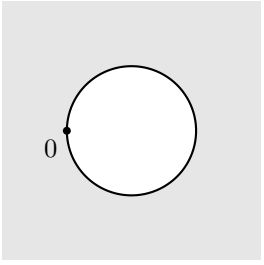
A solution: Corner Gradient Descent

[3] Nemirovskiy & Polyak (1984), Brakhage (1987)

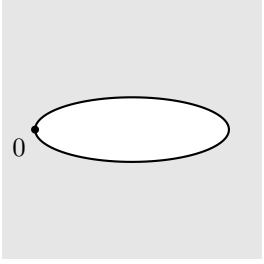
Generalized (S)GD's as contours in $\mathbb{C}$

Stationary generalizations of (S)GD with arbitrary linear memory can be identified with **contours** $\gamma \subset \mathbb{C}$ through **frequency response function** Ψ: $\gamma = \Psi(\{|z| = 1\})$

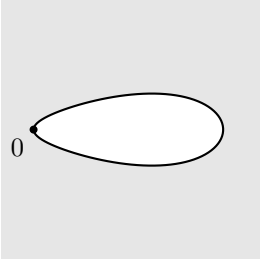
Plain (S)GD:
a circle



Heavy Ball:
an ellipse

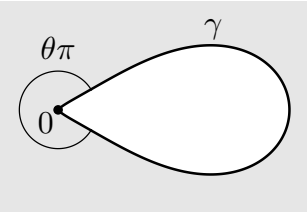


General memory-1:
a Zhukovsky airfoil



Corner algorithms

Correspond to contours with external angle $\theta\pi$



Accelerate convergence exponent of **non-stochastic** GD by the factor θ:

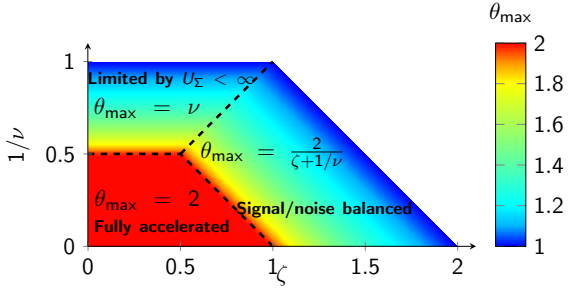
$$L(\mathbf{w}_t) \propto t^{-\zeta} \quad \rightsquigarrow \quad L(\mathbf{w}_t) \propto t^{-\theta\zeta}$$

Maximum acceleration is $\theta = 2$ because the full angle is 2π

Corner algorithms in mini-batch stochastic setting

But corner algorithms also **amplify sampling noise**

For tasks with power-law spectral data, maximum acceleration $\theta_{\max}$ is obtained by balancing deterministic acceleration and noise amplification



Finite-memory algorithms

Ideal corner algorithms require **infinite memory**, but can be efficiently approximated by finite-memory algorithms thanks to fast rational approximations of power functions[4]:

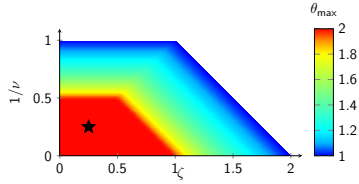
$$\sup_{\Re(z)>0,|z|<1} \left| z^\theta - \frac{P(z)}{Q(z)} \right| = O\left(e^{-c\sqrt{\deg(PQ)}}\right)$$

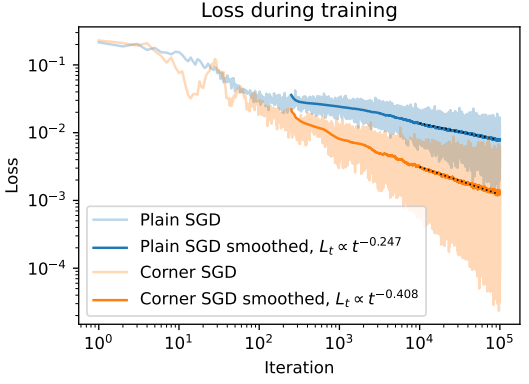
[4] Newman (1964), Gopal & Trefethen (2019)

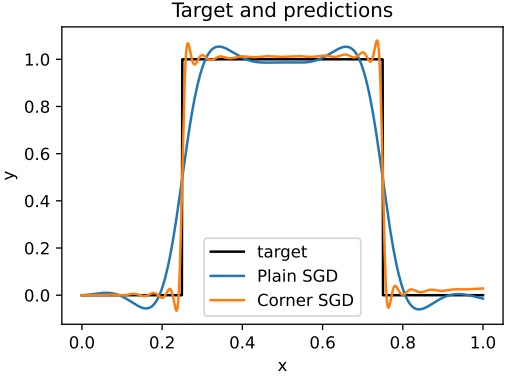
Example 1[5]: Indicator function $\mathbf{1}_{[1/4,3/4]}(x)$

ReLU NN with one hidden layer; only output layer is trained

Theory: $\zeta = \frac{1}{4}$; feasible accelerations up to $\theta_{\max} = 2$





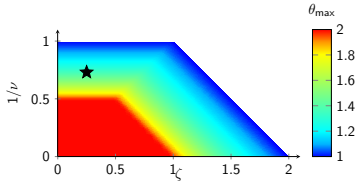


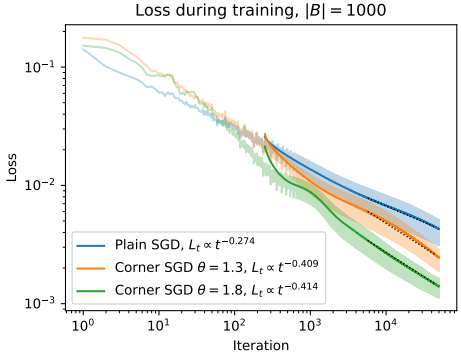
[5] Demo: https://github.com/yarotsky/corner-gradient-descent

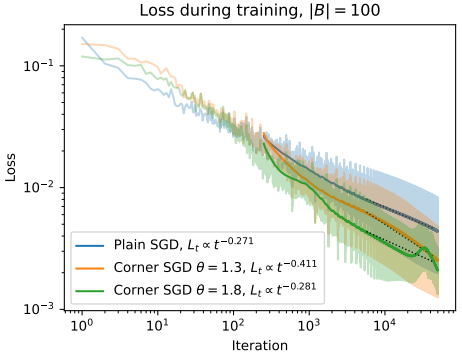
Example 2: classifier MNIST

NN with one hidden layer, $\zeta \approx 0.25$

Theory: acceleration $\theta_{\text{max}} \approx 1.35$







Train and test accuracy