Clarke Differential

Clarke Differential

Definition: Given $f : \mathbb{R}^d \to \mathbb{R}$, for every x, the Clark differential is defined as
$\partial f(x) \triangleq$ conv $\left(\{ s \in \mathbb{R}^d : \exists \{x_i\}_{i=1}^{\infty} \to x, \{ \nabla f(x_i) \}_{i=1}^{\infty} \to s \} \right)$.
The elements in the subdifferential set are subgradients.

When does Clarke differential exists

Definition (Locally Lipschitz): $f : \mathbb{R}^d \to \mathbb{R}$ is locally Lipchitz if $\forall x \in \mathbb{R}^d$, there exists a neighborhood S of x, such that f is Lipchitz in S.

Positive Homogeneity

Definition: $f : \mathbb{R}^d \to \mathbb{R}$ is positive homogeneous of degree L if $f(\alpha x) = \alpha^L f(x)$ for any $\alpha \geq 0$.

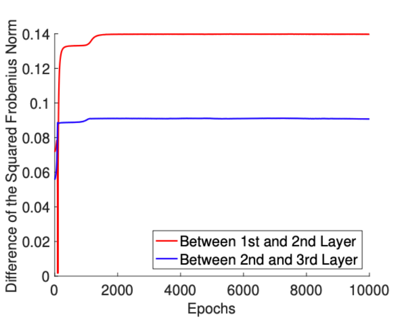
Positive Homogeneity

Positive Homogeneity

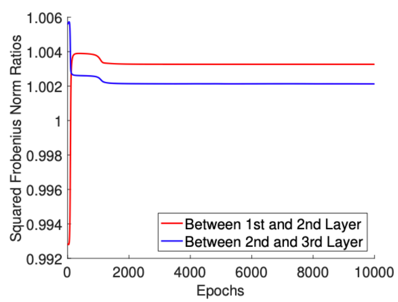
Positive Homogeneity and Clark Differential

Lemma: Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is Locally Lipschitz and L-positively homogeneous. For any $x \in \mathbb{R}^d$ and $s \in \partial f(x)$, we have $\langle s, x \rangle = Lf(x)$.

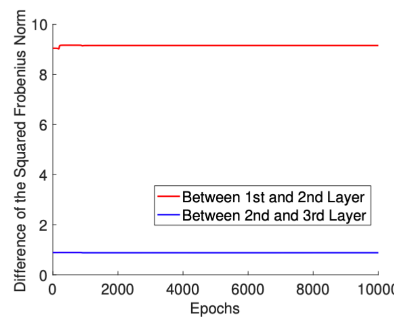
Norm Preservation



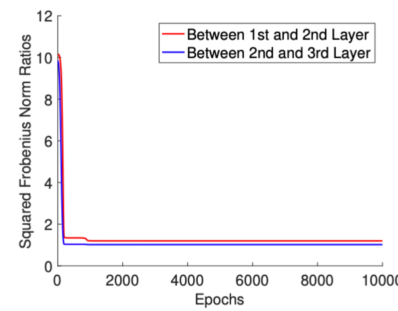
(a) Balanced initialization, squared norm differences.



(b) Balanced initialization, squared norm ratios.



(c) Unbalanced Initialization, squared norm differences.



(d) Unbalanced initialization, squared norm ratios.

Gradient flow and gradient inclusion

Discrete-time dynamics can be complex. Let's use continuous-time dynamics to simplify:

Gradient flow: $x_{t+1} = x_t - \eta \nabla f(x_t) \Rightarrow \frac{x(t)}{dt} = - \nabla f(x(t))$

Gradient inclusion: $\frac{dx(t)}{dt} \in \partial f(x(t))$

Norm preservation by gradient inclusion

Theorem (Du, Hu, Lee '18) Suppose $\alpha > 0$, $f(x; (W_{H+1}, \ldots, \alpha W_i, \ldots, W_1)) = \alpha f(x, (W_{H+1}, \ldots, W_1))$, I.e., predictions are 1-homogeneous in each layer. Then for every pair of layers $(i, j) \in [H+1] \times [H+1]$, the gradient inclusion maintains: for all $t \geq 0$,

$$\frac{1}{2}\|W_h(t)\|_F^2 - \frac{1}{2}\|W_h(0)\|_F^2 = \frac{1}{2}\|W_h(t)\|_F^2 - \frac{1}{2}\|W_{h'}(0)\|_F^2.$$

Optimization Methods for Deep Learning

Gradient descent for non-convex optimization

Decsent Lemma: Let $f : \mathbb{R}^d \to \mathbb{R}$ be twice differentiable, and $\|\nabla^2 f\|_2 \le \beta$. Then setting the learning rate $\eta = 1/\beta$, and applying gradient descent, $x_{t+1} = x_t - \eta \nabla f(x_t)$, we have:

$$f(x_t) - f(x_{t+1}) \ge \frac{1}{2\beta} \|\nabla f(x_t)\|_2^2.$$

Converging to stationary points

Theorem: In $T = O(\frac{\beta}{\epsilon^2})$ iterations, we have $\|\nabla f(x)\|_2 \leq \epsilon$.

Gradient Descent for Quadratic Functions

Problem: $\min_x \frac{1}{2} x^\top A x$ with $A \in \mathbb{R}^{d\times d}$ being positive-definite.

Theorem: Let $\lambda_{\max}$ and $\lambda_{\min}$ be the largest and the smallest eigenvalues of A. If we set $\eta \le \frac{1}{\lambda_{\max}}$, we have

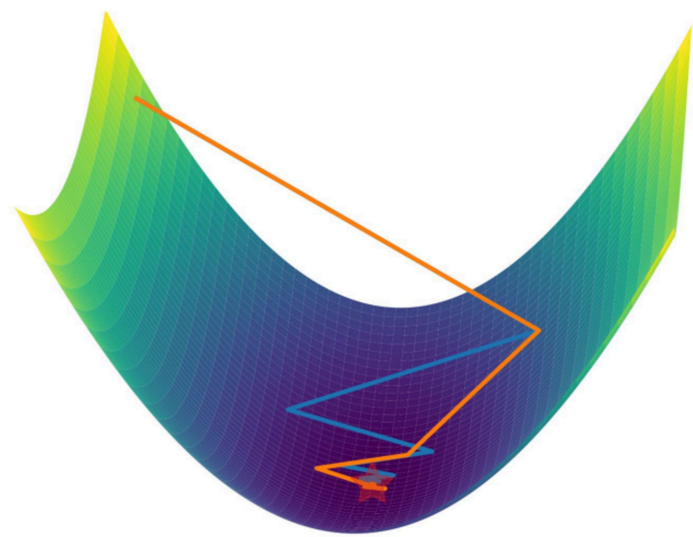
$$\|x_t\|_2 \le \left(1 - \eta\lambda_{\min}\right)^t \|x_0\|_2$$

Momentum: Heavy-Ball Method (Polyak '64)

Problem: $\min_x f(x)$

Method: $v_{t+1} = -\nabla f(x_t) + \beta v_t$

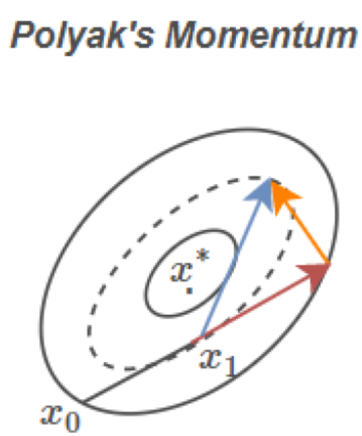
$x_{t+1} = x_t + \eta v_{t+1}$

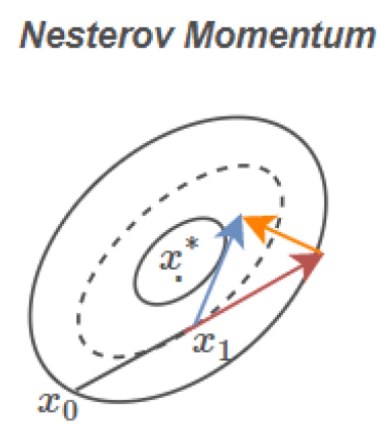


Momentum: Nesterov Acceleration (Nesterov ’89)

Problem: $\min_x f(x)$

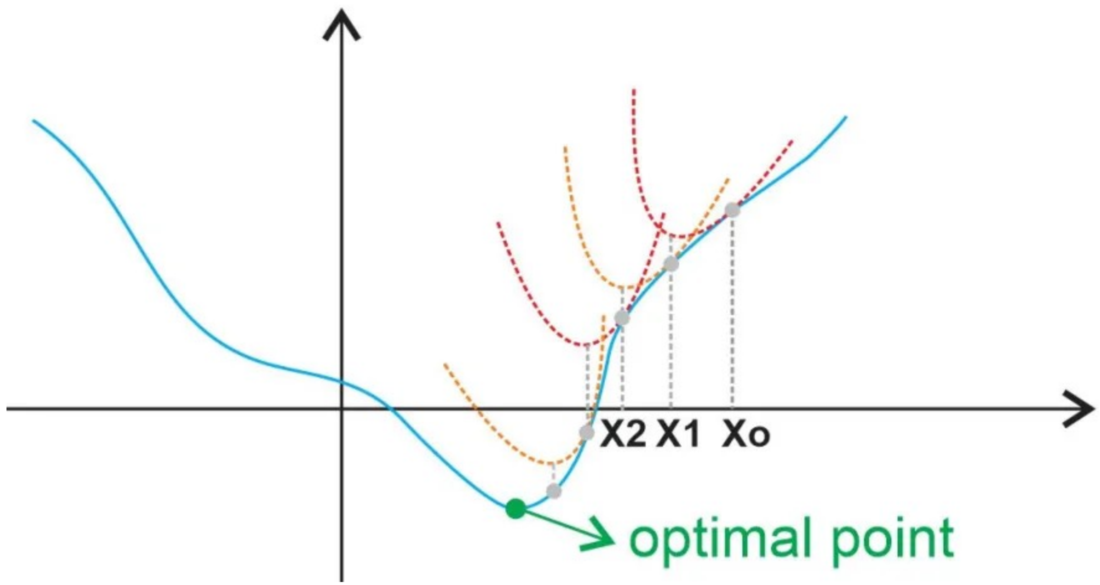
Method:
$$v_{t+1} = -\nabla f(x_t + \beta v_t) + \beta v_t$$
$$x_{t+1} = x_t + \eta v_{t+1}$$





Newton's Method

Newton's Method: $x_{t+1} = x_t - \eta(\nabla^2 f(x_t))^{-1} \nabla f(x_t)$



AdaGrad (Duchi et al. '11)

Newton Method: $x_{t+1} = x_t - \eta(\nabla^2 f(x_t))^{-1} \nabla f(x_t)$

AdaGrad: separate learning rate for every parameter

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t),\ (G_t)_{ii} = \sqrt{\sum_{j=1}^{t-1} \left(\nabla f(x_t)_i \right)^2}$$

RMSProp (Hinton et al. ’12)

AdaGrad: separate learning rate for every parameter

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t), \; (G_t)_{ii} = \sqrt{\sum_{j=1}^{t-1} \left(\nabla f(x_t)_i \right)^2}$$

RMSProp: exponential weighting of gradient norms

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t),$$
$$(G_{t+1})_{ii} = \beta (G_t)_{ii} + (1 - \beta)(\nabla f(x_t)_i)^2$$

AdaDelta (Zeiler '12)

RMSProp:

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t),$$
$$(G_{t+1})_{ii} = \beta(G_t)_{ii} + (1 - \beta)(\nabla f(x_t)_i)^2$$

AdaDelta:

$$x_{t+1} = x_t - \eta \Delta x_t,$$
$$\Delta x_t = \sqrt{u_t + \epsilon} \cdot (G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t)$$
$$(G_{t+1})_{ii} = \rho(G_t)_{ii} + (1 - \rho)(\nabla f(x_t)_i)^2,$$
$$u_{t+1} = \rho u_t + (1 - \rho)\|\Delta x_t\|_2^2$$

Adam (Kingma & Ba ’14)

Momentum:

$v_{t+1} = -\nabla f(x_t) + \beta v_t, x_{t+1} = x_t + \eta v_{t+1}$

RMSProp: exponential weighting of gradient norms

$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t),$

$(G_t)_{ii} = \beta(G_t)_{ii} + (1-\beta)(\nabla f(x_t)_i)^2$

Adam

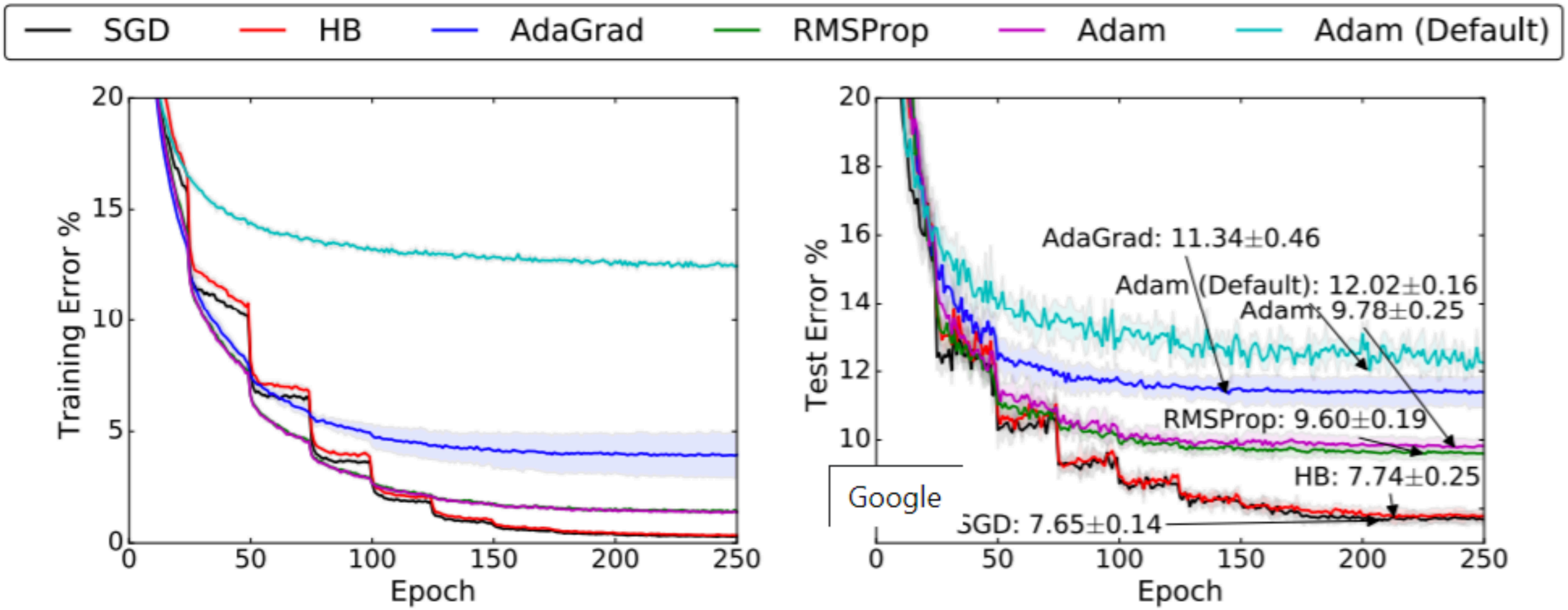
$v_{t+1} = \beta_1 v_t + (1-\beta_1) \nabla f(x_t)$

$(G_{t+1})_{ii} = \beta_2(G_t)_{ii} + (1-\beta_2)(\nabla f(x_t)_i)^2$

$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1/2} v_{t+1}$

Default choice nowadays.

Are these actually useful



(a) CIFAR-10 (Train) **(b)** CIFAR-10 (Test)

Figure 1: Training (left) and top-1 test error (right) on CIFAR-10. The annotations indicate where the best performance is attained for each method. The shading represents ± one standard deviation computed across five runs from random initial starting points. In all cases, adaptive methods are performing worse on both train and test than non-adaptive methods.

Wilson, Roelofs, Stern, Srebro, Recht '18