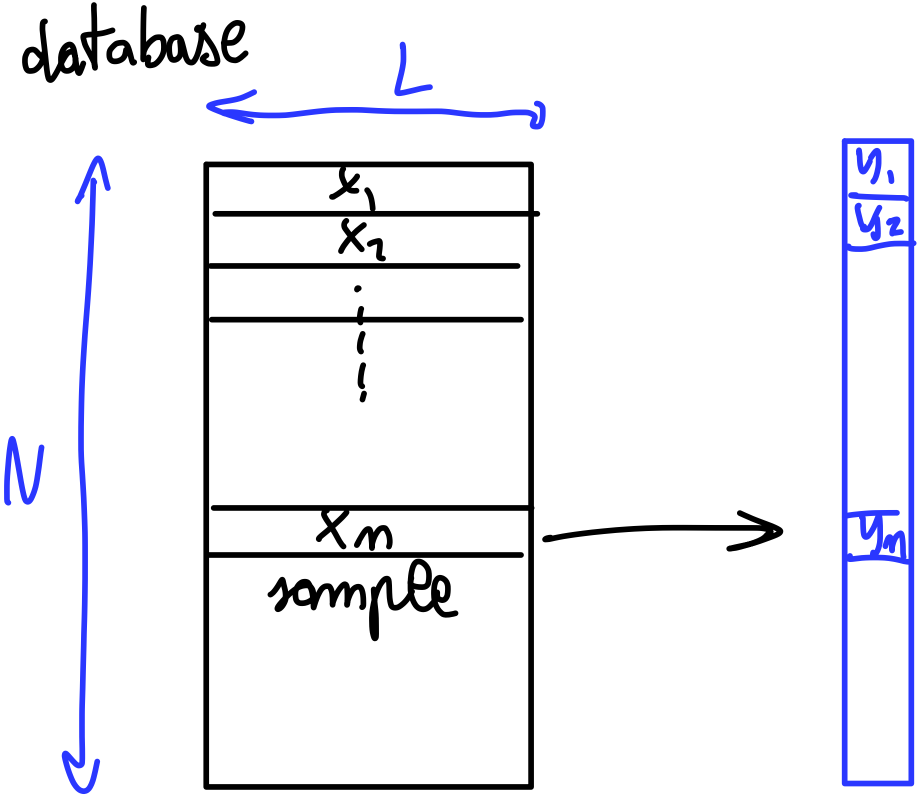
Optimization techniques

Data analysis

Machine Learning (ML)

Physics

tools of Ph. can be useful for ML



label, feature

(supervised data)

x : data

y : labels

θ : parameters

z : hidden variables

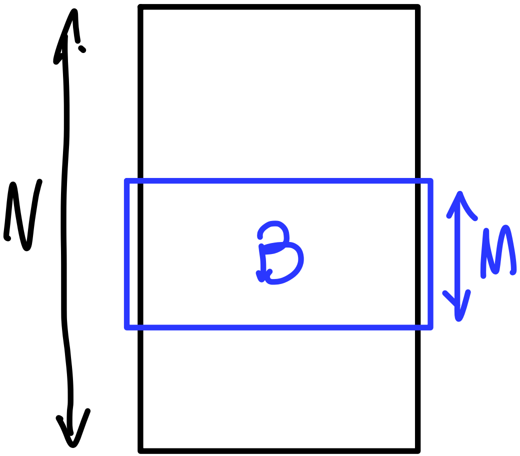
$$\boxed{x_n} \longrightarrow \boxed{\begin{matrix}\text{model} \\ \theta\end{matrix}} \longrightarrow \hat{y}(x_n)$$

prediction
(depends on θ)

Cost function $E(x_n, y_n, \theta)$ (also: $C, L, \ldots$)
(loss)

e.g. $E = \frac{1}{2}\left(\hat{y}(x) - y\right)^2$

Minibatch B : subset of data, of size $M \ll N$



cost function

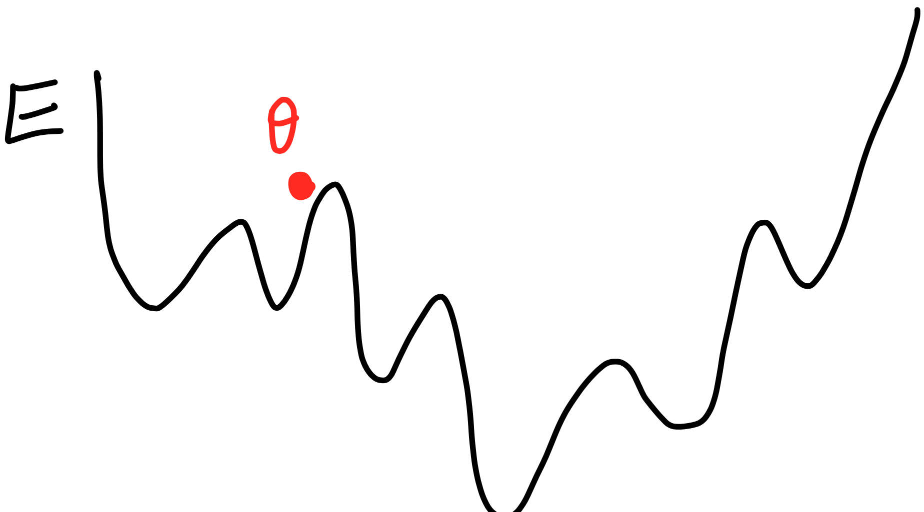
$$\rightarrow \quad E_B = \sum_{n \in B} E(x_n, y_n, \theta)$$

- faster
- introduces "noise", $E_B \neq E$ of whole database

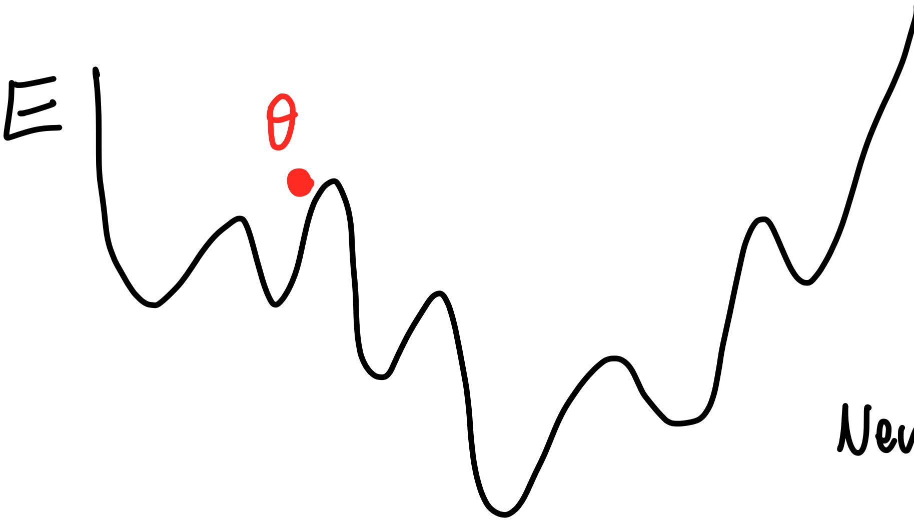
Aim: minimize E $\Longleftrightarrow$ energy minimization

Aim: minimize E $\Longleftrightarrow$ energy minimization

by changing parameter(s) θ



Newton ?



momentum

$$p = \frac{d}{dt}\theta \equiv \dot{\theta}$$

Newton's eq.

$$\begin{cases} m\,\dot{p} = -\dfrac{\partial E}{\partial \theta} \\ \dot{\theta} = p \end{cases}$$

mass . acceler.

minus gradient of pot. energy

⚠ Does not stop at minimum

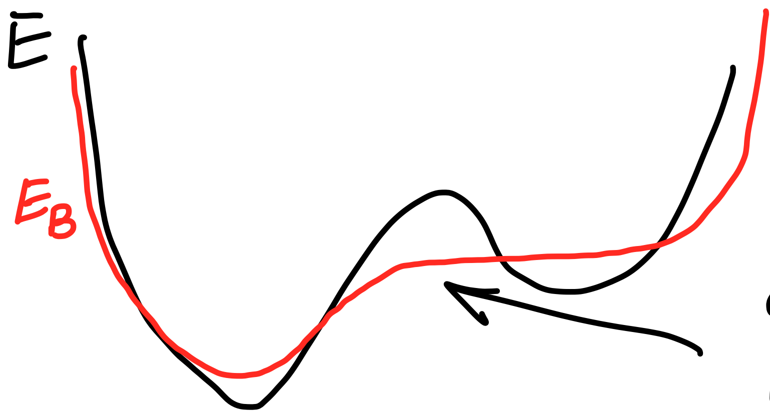
Langevin : Newton + friction + noise

$$\begin{cases} m\dot{p} = -\frac{\partial E}{\partial \theta} - \phi p + \xi \\ \dot{\theta} = p \end{cases}$$

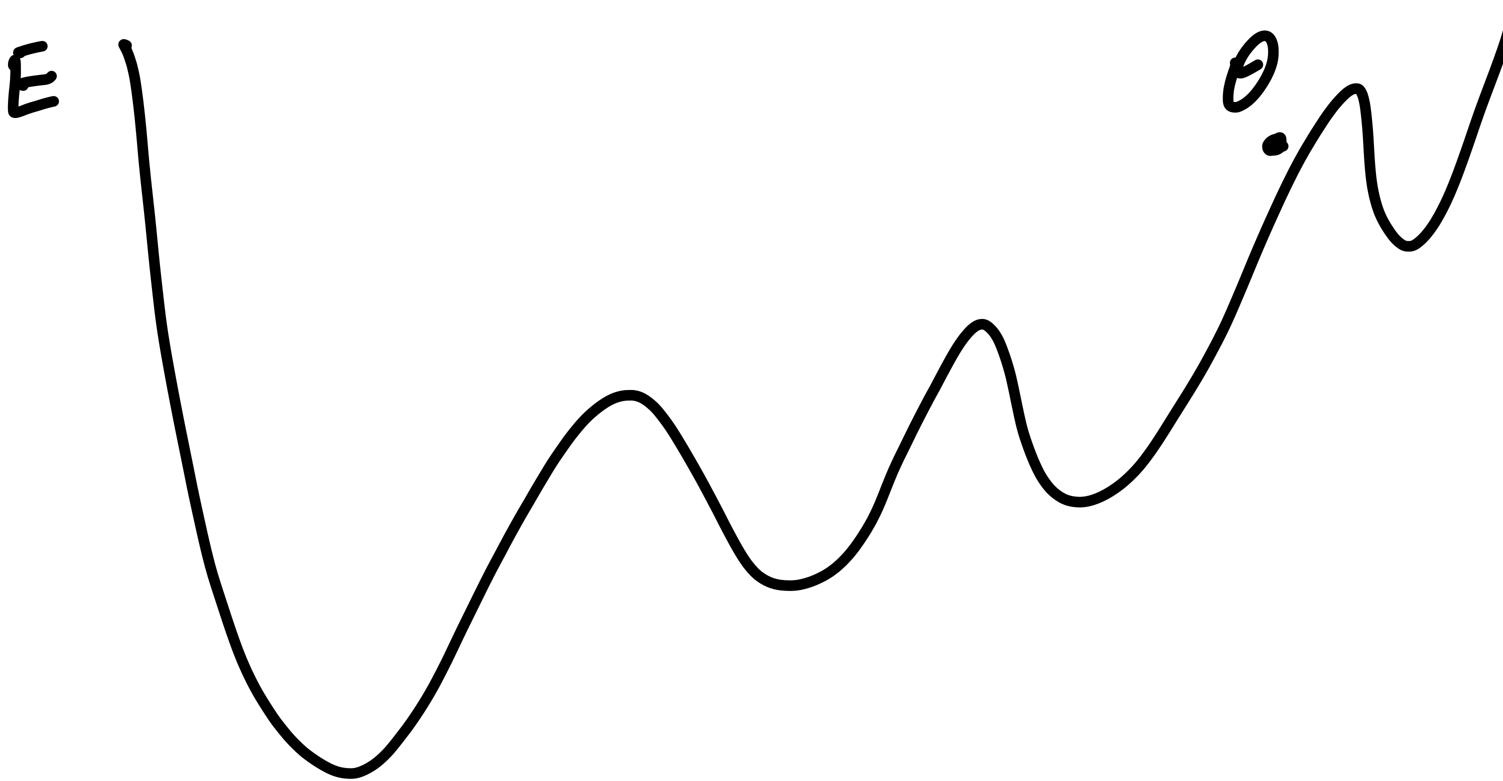
friction coeff.

noise : using minibatch

$$\sim \frac{\partial}{\partial \theta}\left(E_B - E\right)$$



difference $E_B - E$ may be useful for overtaking local barriers


E
θ

"Vanilla" Gradient descent

"Stochastic" GD if using minibatches

---

algorithm: discrete update at "time" $t = 0, 1, 2, \ldots$

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta E(\theta_t)$$

learning rate $\eta \ll 1$

"Vanilla" Gradient descent

$$\boxed{\theta_{t+1} = \theta_t - \eta \nabla_\theta E(\theta_t)} \quad (1)$$

corresponds to overdamped dynamics (no momentum

$m = 0$

$$\theta_{t+1} - \theta_t = \dot{\theta} \, (\phi \, \Delta t) = - \eta \nabla_\theta E$$

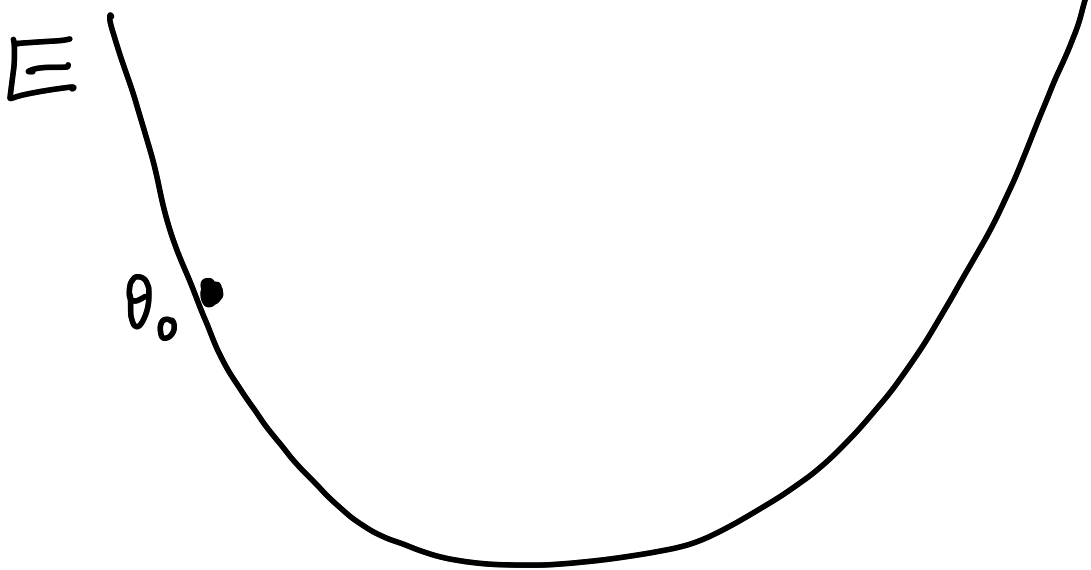
short time interval

$\Rightarrow \eta \sim \phi \, \Delta t$

ϕ: friction coeff.

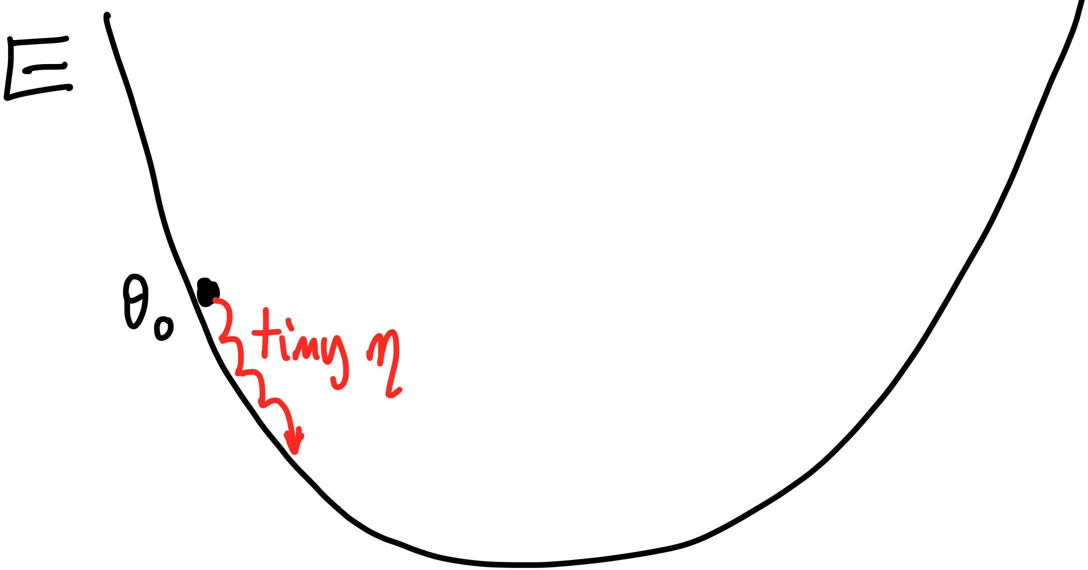
"Vanilla" Gradient descent

$$\boxed{\theta_{t+1} = \theta_t - \eta \nabla_\theta E(\theta_t)} \quad (1)$$



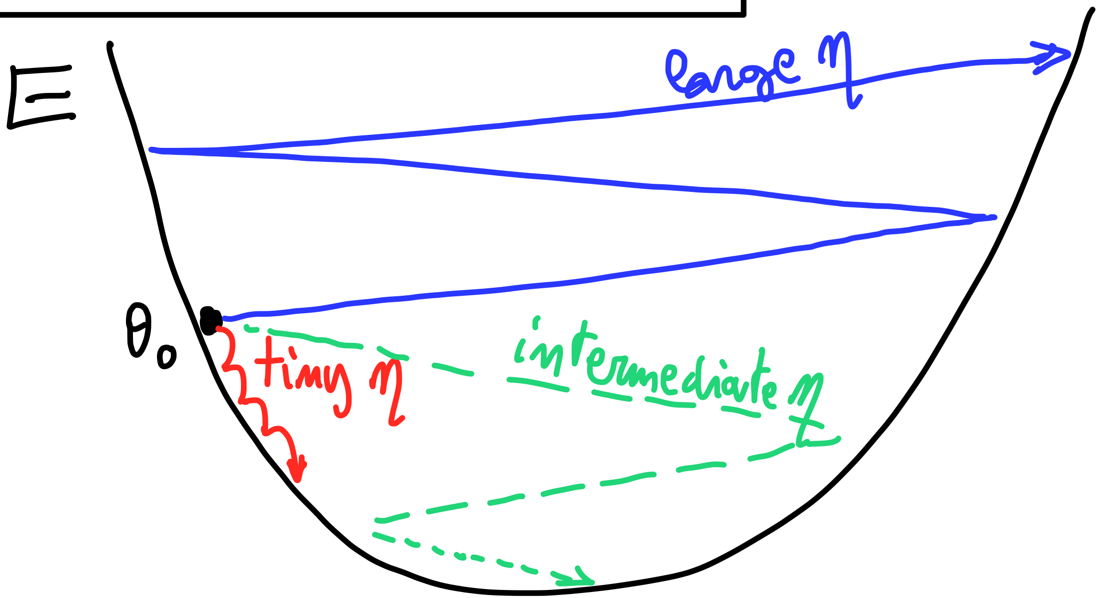
"Vanilla" Gradient descent

$$\boxed{\theta_{t+1} = \theta_t - \eta \nabla_\theta E(\theta_t)} \quad (1)$$



"Vanilla" Gradient descent

$$\boxed{\theta_{t+1} = \theta_t - \eta \nabla_\theta E(\theta_t)} \quad (1)$$



Momentum : $-v \Longleftrightarrow p \cdot \Delta t$

$$(2)\begin{cases} v_t = \gamma\, v_{t-1} + \eta\, \nabla_\theta E(\theta_t) \\ \theta_{t+1} = \theta_t - v_t \end{cases}$$

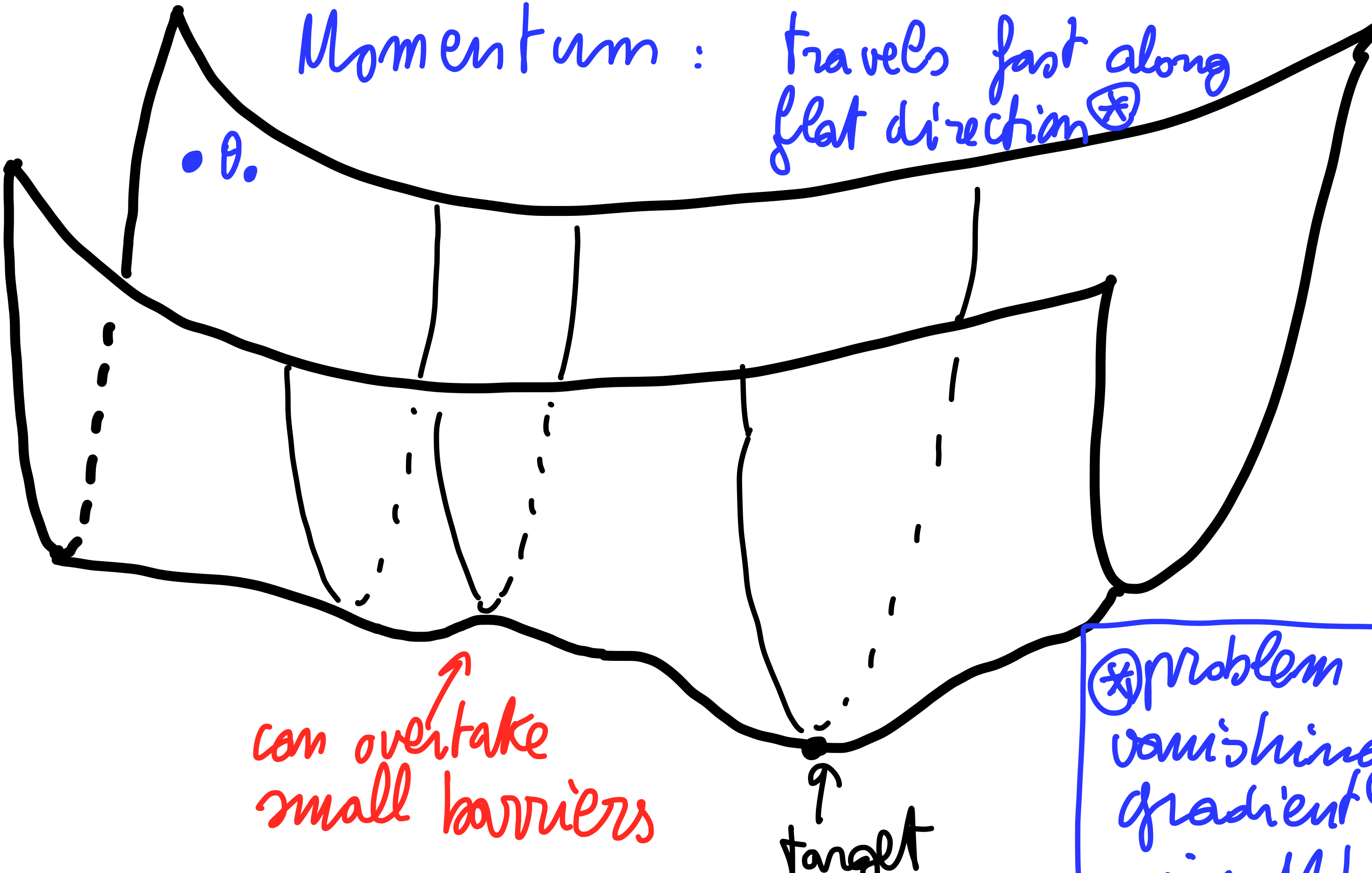
γ keeps memory of v, being $0 < \gamma < 1$
(usually $\gamma = 0.9$ or 0.99)

$(1-\gamma)$ is related to friction coeff. ϕ

Momentum : Nesterov Accelerated Gradient (NAG)

$$(3)\begin{cases} v_t = \gamma\, v_{t-1} + \eta\, \nabla_\theta E(\theta_t - \gamma v_t) \\ \theta_{t+1} = \theta_t - v_t \end{cases}$$

Momentum: travels fast along flat direction ⊛



⊛ problem of vanishing gradient in ML ⚠

⚠ problem of diverging gradient as well

↓

reducing learning rate η is safer but less effective

Methods using 2^nd moment of gradient

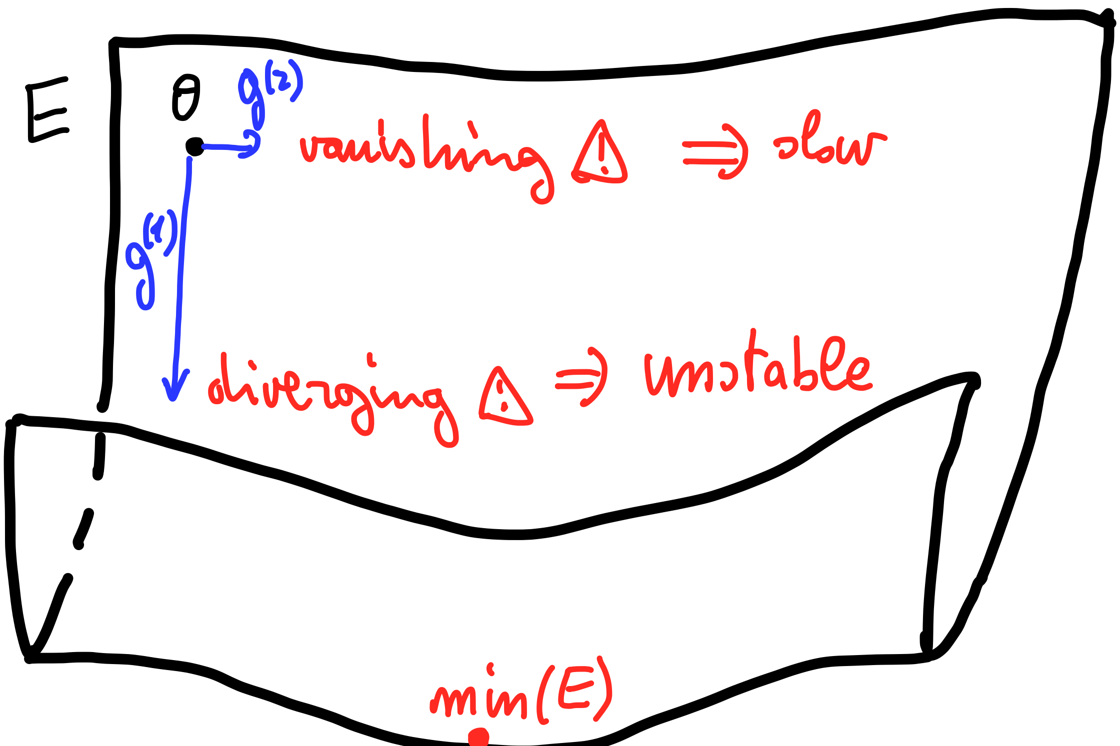
vector of parameters $\theta = (\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(D)})$

$$g^{(i)} = \frac{\partial E}{\partial \theta^{(i)}} \longrightarrow |g^{(i)}|^2$$

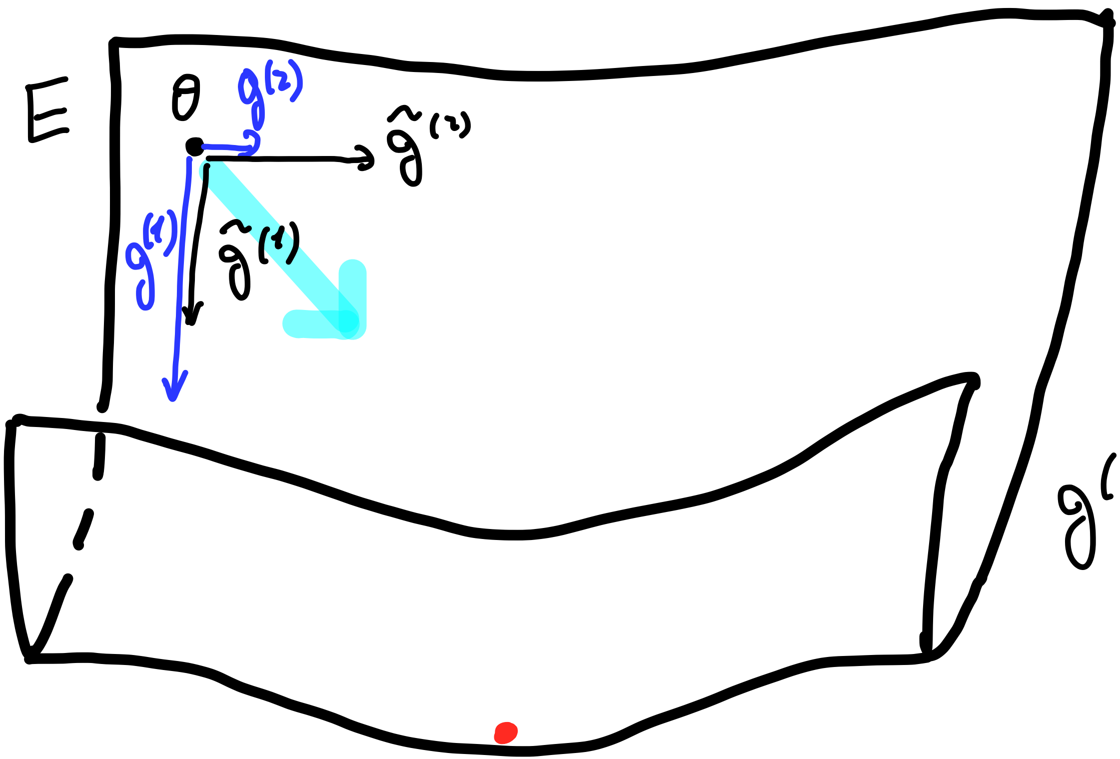
for 2^nd moment

used to rescale gradient (sort of self tuning learning rate)

Methods using 2^{nd} moment of gradient



Methods using 2nd moment of gradient



crude solution: set all $g^{(i)} = 1$

$$g^{(i)} \rightarrow \tilde{g}^{(i)} = \frac{g^{(i)}}{\sqrt{|g^{(i)}|^2}}$$

RMS prop

(4)

$$g^{(i)} = \frac{\partial}{\partial \theta^{(i)}} E(\theta_t)$$

$$s^{(i)} = \beta\, s_{t-1}^{(i)} + (1-\beta)\, |g_t^{(i)}|^2$$

$$\theta_{t+1}^{(i)} = \theta_t^{(i)} - \eta \frac{g^{(i)}}{\sqrt{s_t^{(i)} + \varepsilon}}$$

typical

$\beta = 0.9$

memory $\sim \beta$

$\varepsilon = 10^{-8}$

avoids division by zero

NOTE: NO momentum

RMS prop

(4)

$$g^{(i)} = \frac{\partial}{\partial \theta^{(i)}} E(\theta_t)$$

$$s^{(i)} = \beta\, s_{t-1}^{(i)} + (1-\beta)\, |g_t^{(i)}|^2$$

$$\theta_{t+1}^{(i)} = \theta_t^{(i)} - \eta \frac{g^{(i)}}{\sqrt{s_t^{(i)} + \varepsilon}}$$

hence $\theta_t^{(i)}$ updated with rescaled $g^{(i)}$ taking into account recent values of its 2^{nd} moment

initial t's $\Rightarrow$ ~ crude approx

ADAM

$$g_t^{(i)} = \frac{\partial}{\partial \theta^{(i)}} E(\theta_t)$$

(5)

$$m_t^{(i)} = \beta_1 m_{t-1}^{(i)} + (1-\beta_1)\, g_t^{(i)}$$

$$s_t^{(i)} = \beta_2 s_{t-1}^{(i)} + (1-\beta_2)\, s_t^{(i)}$$

$\beta_1, \beta_2 \simeq 0.9$
0.99

ADAM

(5)

$$g_t^{(i)} = \frac{\partial}{\partial \theta^{(i)}} E(\theta_t)$$

$$m_t^{(i)} = \beta_1 \, m_{t-1}^{(i)} + (1-\beta_1) \, g_t^{(i)}$$

$$s_t^{(i)} = \beta_2 \, s_{t-1}^{(i)} + (1-\beta_2) \, s_t^{(i)}$$

at small t
they amplify
m, s
($\beta_1, \beta_2 < 1$)

$$\hat{m}_t^{(i)} = \frac{1}{1-(\beta_1)^t} \, m_t^{(i)}$$

$$\hat{s}_t^{(i)} = \frac{1}{1-(\beta_2)^t} \, s_t^{(i)}$$

ADAM

$$g_t^{(i)} = \frac{\partial}{\partial \theta^{(i)}} E(\theta_t)$$

$$m_t^{(i)} = \beta_1 \, m_{t-1}^{(i)} + (1-\beta_1) \, g_t^{(i)}$$

(5)

$$s_t^{(i)} = \beta_2 \, s_{t-1}^{(i)} + (1-\beta_2) \, s_t^{(i)}$$

$$\hat{m}_t^{(i)} = \frac{1}{1-(\beta_1)^t} \, m_t^{(i)}$$

$$\hat{s}_t^{(i)} = \frac{1}{1-(\beta_2)^t} \, s_t^{(i)}$$

Note: no momentum

$$\theta_{t+1}^{(i)} = \theta_t^{(i)} - \eta \frac{\hat{m}_t^{(i)}}{\sqrt{\hat{s}_t^{(i)} + \varepsilon}}$$

Final comments

- Stocasticity from
  - minibatches
  - (added noise)
- Physics: useful but not rigorous
- ADAM is unstable
  $\downarrow$
  ADAmax cures it

New methods? very active field

Final comments

- Stocasticity from
  - minibatches
  - (added noise)

- Physics: useful but not rigorous

- ADAM is unstable
  $\downarrow$
  <u>ADA max</u> cures it

- for $\theta = (\theta^{(1)}, \dots, \theta^{(100000)})$

  landscape of "energy" $E(\theta)$ contains many interconnected valleys