Optimization

Seyoung Yun

References

- http://lear.inrialpes.fr/workshop/osl2013/slides/osl2013_bach.pdf
- https://github.com/abursuc/dldiy-practicals/blob/master/slides/Slide_October19.pdf
- http://cs231n.github.io/optimization-1/
- http://ruder.io/optimizing-gradient-descent/

Training Algorithm

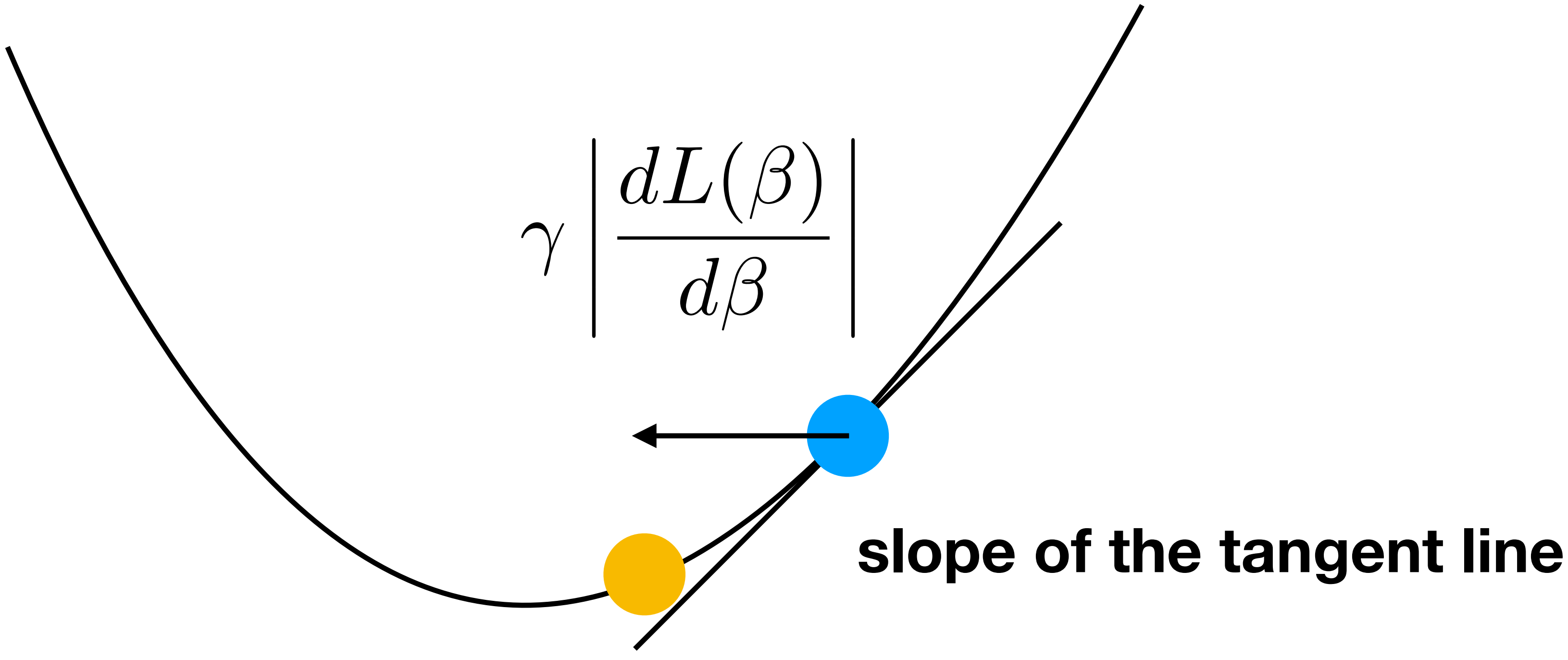
- Loss Function

$$L(\beta) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(Y^{(i)}, \widehat{Y}^{(i)}(\beta)\right)$$

- How to optimize it?
  - Gradient Descent: $\beta(t+1) = \beta(t) - \gamma(t)\nabla L(\beta(t))$
  - Some variants of GD
  - Stochastic gradient descent and its variants

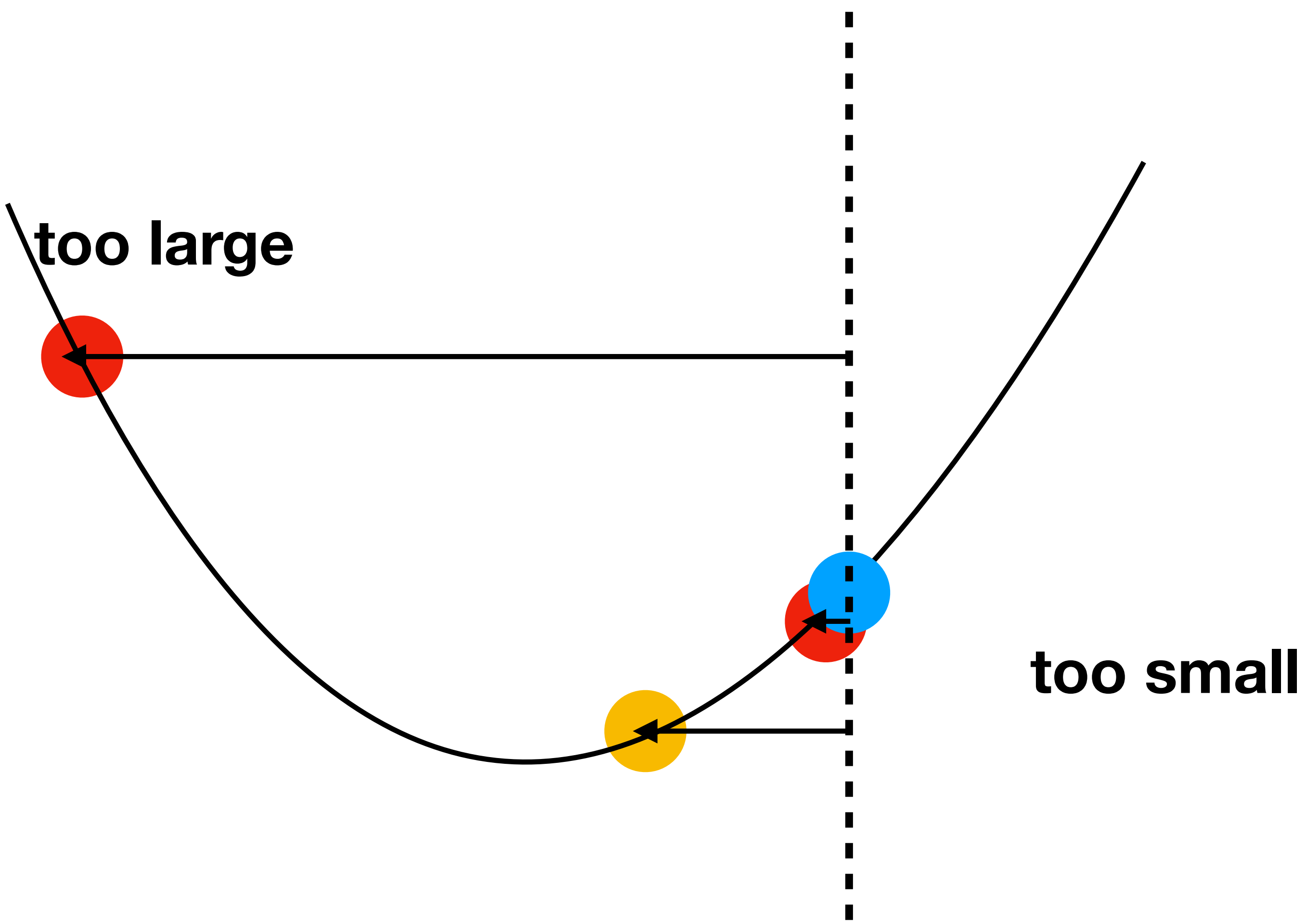
Gradient

- 1-D example
  - Gradient: $$\frac{dL(\beta)}{d\beta} = \lim_{h \to 0} \frac{L(\beta + h) - L(\beta)}{h}$$



Step Size

$\gamma \left| \frac{dL(\beta)}{d\beta} \right|$: the length of the update



2D cases

- Gradient : $\nabla L(\beta) = \begin{bmatrix} \frac{\partial L(\beta)}{\partial \beta_1} \\ \frac{\partial L(\beta)}{\partial \beta_2} \end{bmatrix}$

- Gradient Descent

- Zigzag: why? it is very hard to find a step size than is good for multi dimensional cases (vertical vs. horizontal)

Gradient descent vs Newton

$$\beta(t+1) = \beta(t) - \gamma(t)\nabla L(\beta(t))$$ **vs.**

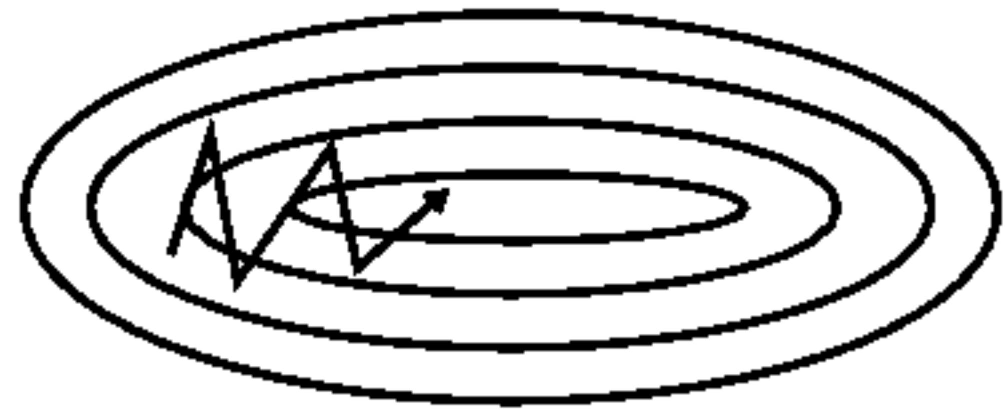
Newton

$$\beta(t+1) = \beta(t) - \gamma\nabla^2 L(\beta(t))^{-1}\nabla L(\beta(t))$$

- Newton converges faster to local minima
    - The inverse Hessian matrix control the step size adaptively
- No one want to compute a Hessian (or worst: inverse it)

Momentum

GD



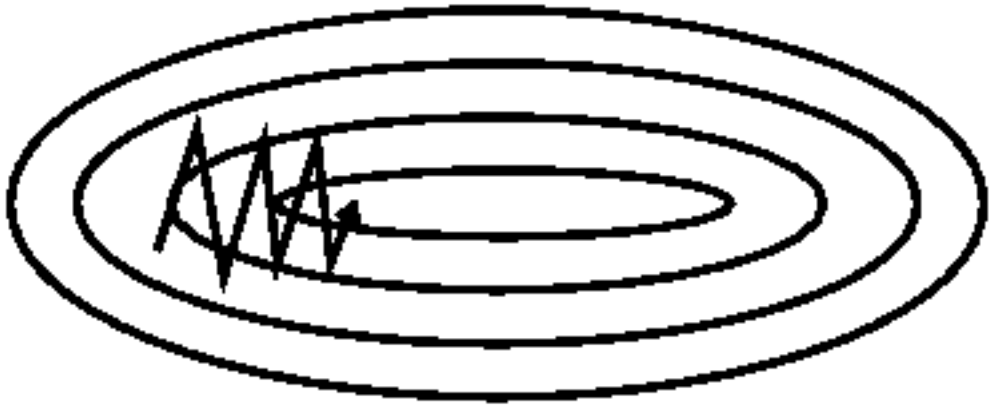
with momentum

momentum:

$$\Delta\beta(t+1) = \eta\Delta\beta(t) + \gamma\nabla f(\beta)$$

$$\beta(t+1) = \beta(t) - \Delta\beta(t+1)$$

AdaGrad



AdaGrad:

$$\beta(t+1) = \beta(t) - \frac{\gamma}{\sqrt{G_t + \varepsilon}} \nabla L(\beta(t))$$

$$G_{t+1} = G_t + (\nabla L(\beta(t)))^2$$

element-wise

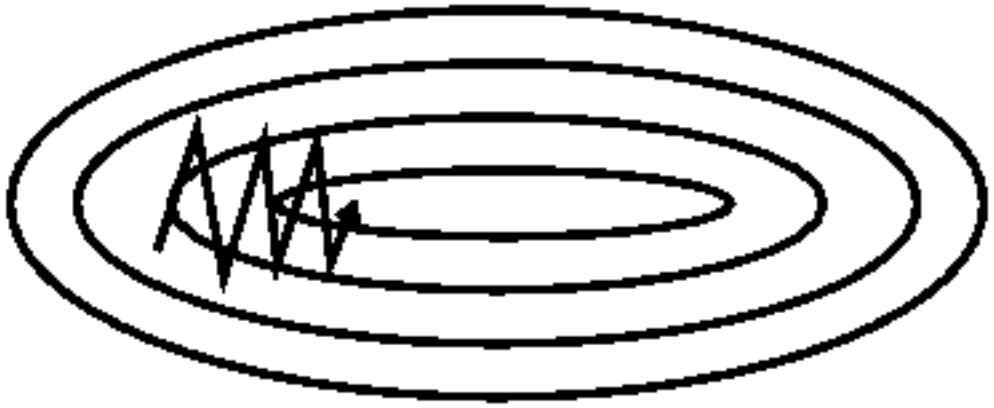
RMSProp

RMSProp:

$$G_t = \eta G_{t-1} + (1-\eta)(\nabla L(\beta(t)))^2$$

$$\beta(t+1) = \beta(t) - \frac{\gamma}{\sqrt{G_t + \varepsilon}} \nabla L(\beta(t))$$

Adam



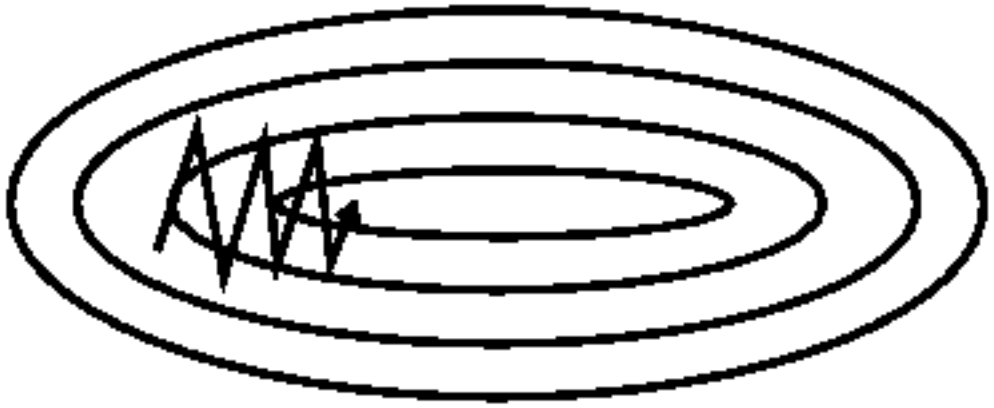
Adam:

$$M_t = \eta_1 M_{t-1} + (1 - \eta_1) \nabla L(\beta(t))$$

$$G_t = \eta_2 G_{t-1} + (1 - \eta_2) (\nabla L(\beta(t)))^2$$

$$\beta(t+1) = \beta(t) - \frac{\gamma}{\sqrt{G_t + \varepsilon}} M_t$$

Adam



Adam:

$$M_t = \eta_1 M_{t-1} + (1-\eta_1)\nabla L(\beta(t))$$

$$G_t = \eta_2 G_{t-1} + (1-\eta_2)(\nabla L(\beta(t)))^2$$

$$\hat{M}_t = \frac{M_t}{1-\eta_1^t} \qquad \hat{G}_t = \frac{G_t}{1-\eta_2^t}$$

$$\beta(t+1) = \beta(t) - \frac{\gamma}{\sqrt{G_t + \varepsilon}} M_t$$

Optimization

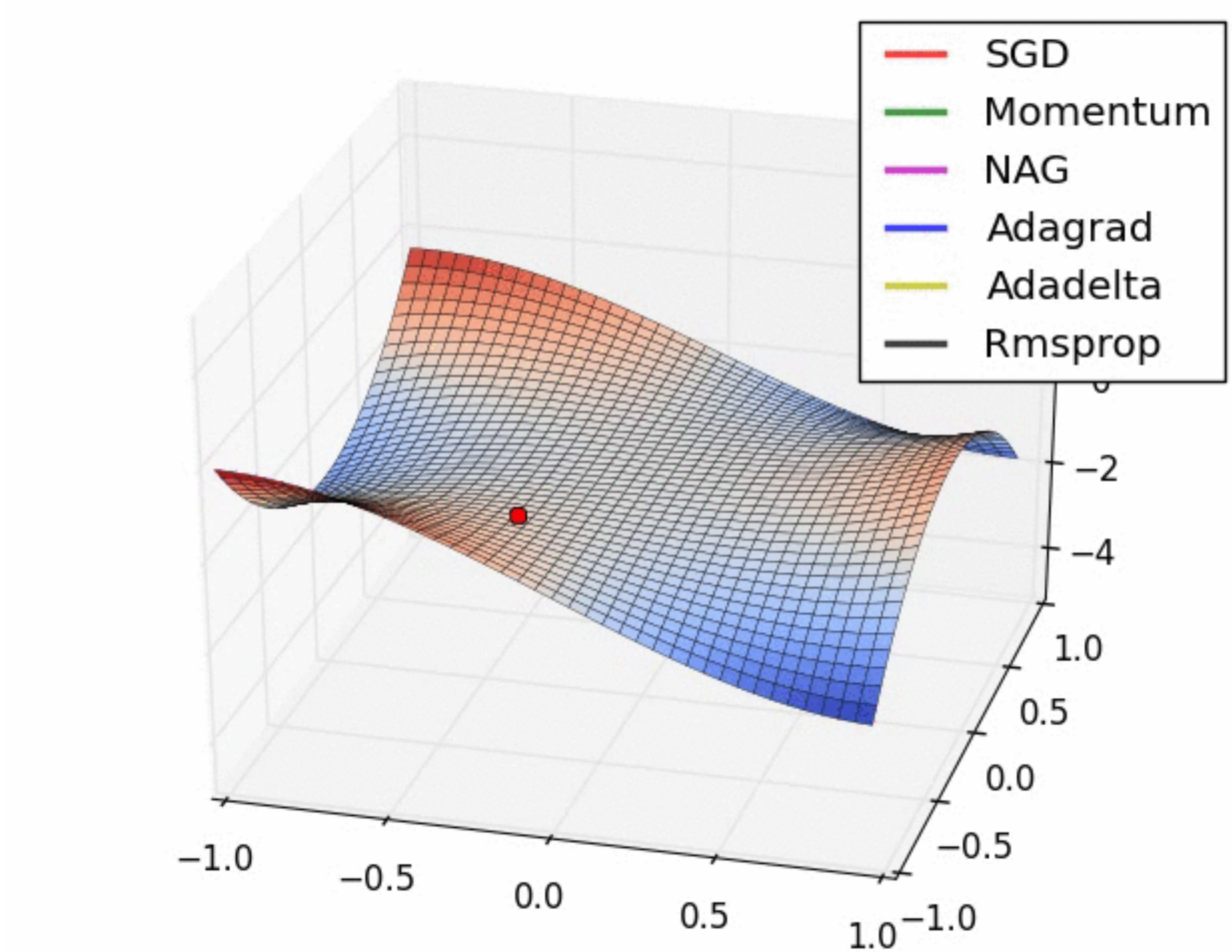


Image by Alec Radford

Optimization

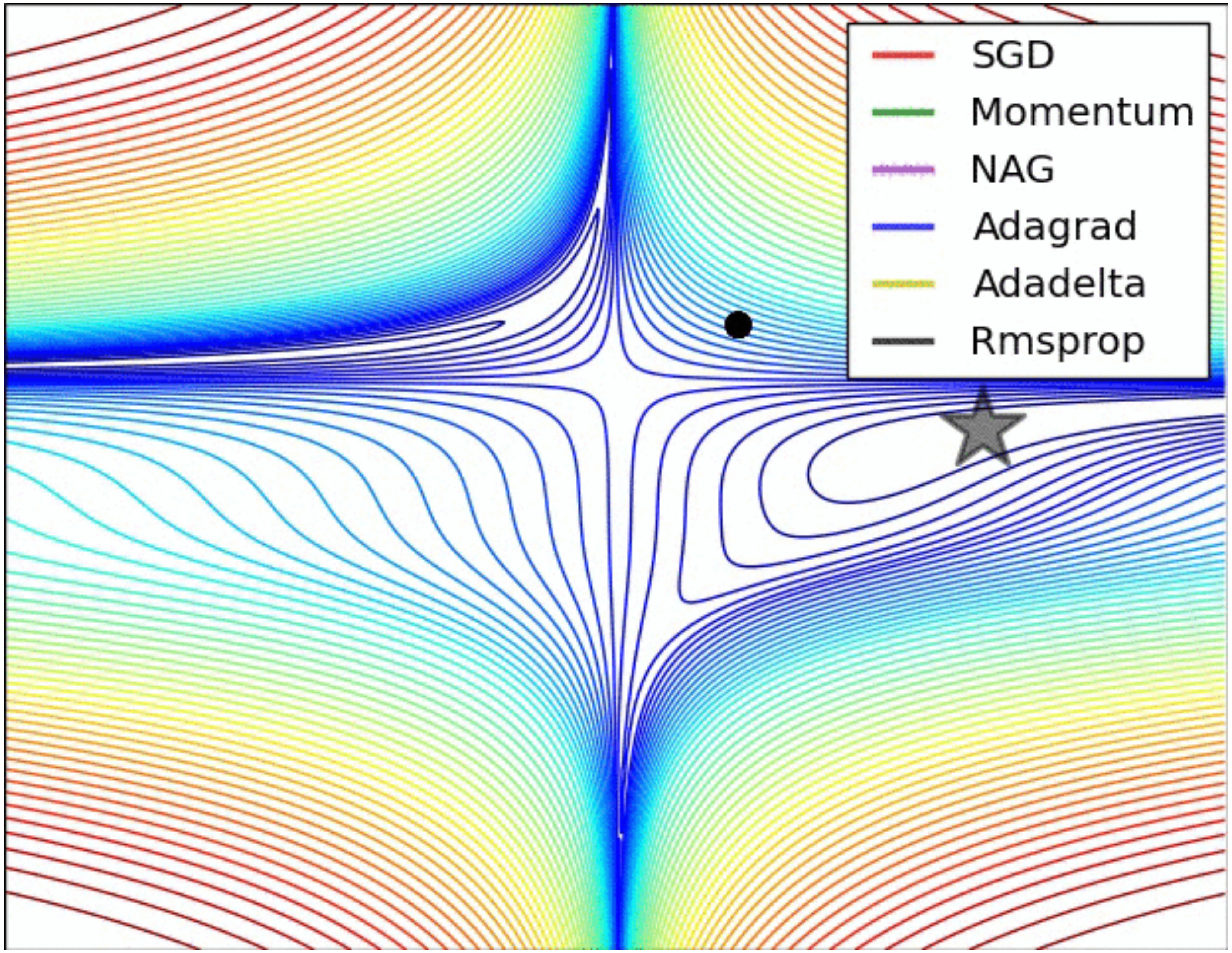


Image by Alec Radford

Batch vs mini-Batch vs SGD

- Batch Gradient Descent

$$\beta \leftarrow \beta - \eta \nabla L(\beta)$$

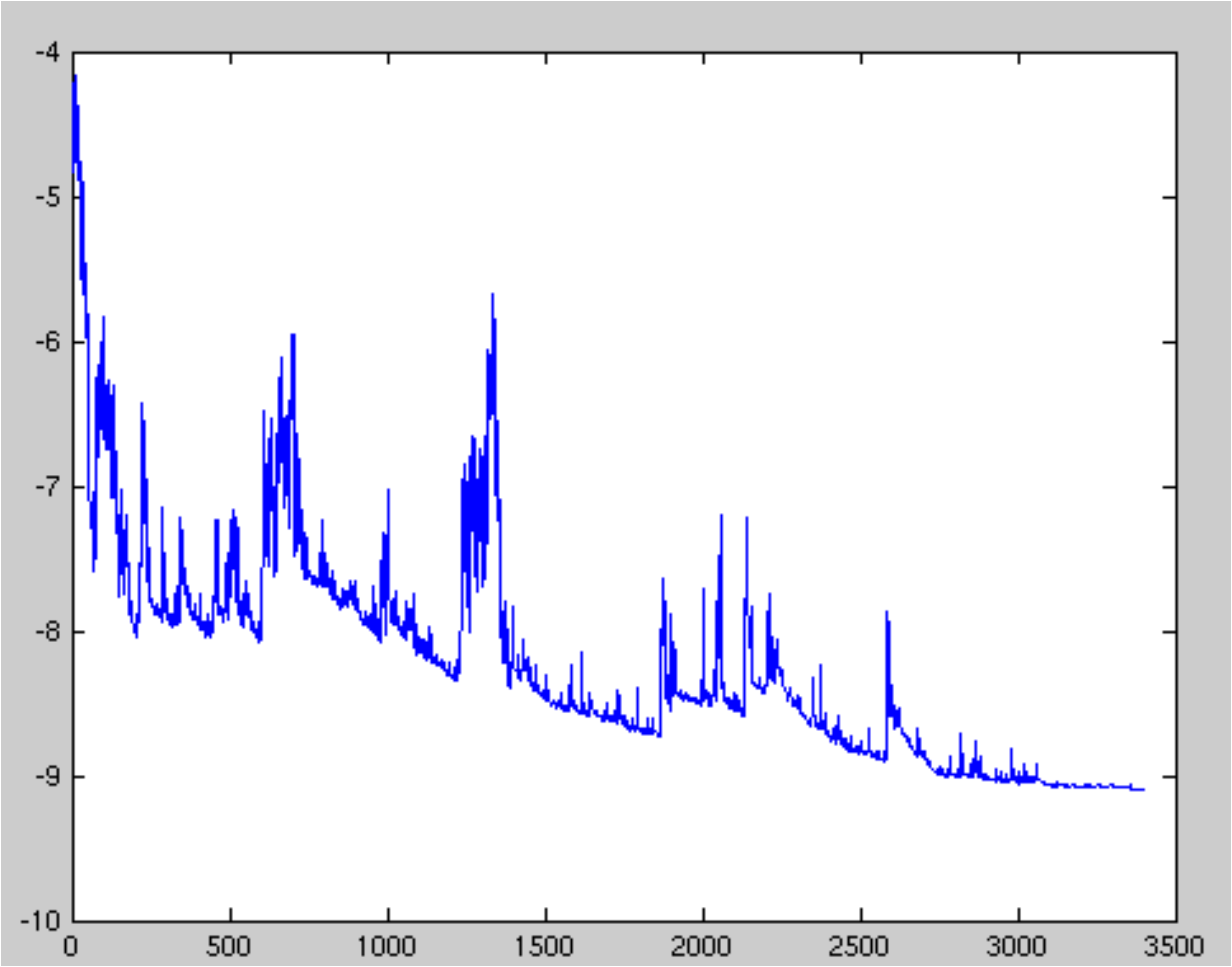
- Mini-batch Gradient Descent

$$\beta \leftarrow \beta - \eta \nabla \left(\sum_{\tau=1}^{B} \ell(Y^{(i_\tau)}, \widehat{Y}^{(i_\tau)}(\beta)) \right)$$

- Stochastic Gradient Descent

$$\beta \leftarrow \beta - \eta \nabla \ell(Y^{(i)}, \widehat{Y}^{(i)}(\beta))$$

SGD fluctuation



Learning rate decay

- Now, eta is a function of the number of iterations, epochs,