Optimization Algorithms used in Neural Networks

Guoxin Sui, Yanru Qu

2017/11/08

Optimization Algorithms

- 1st order optimization
  - The gradient tells us whether the objective is decreasing or increasing at a point, which gives a **tangent line** on the error surface
  - The gradient of a function produces a **Vector Field**
  - A gradient is represented by a **Jacobian** Matrix

- 2nd order optimization
  - Use the 2nd order derivative to optimize the objective, which provides a **quadratic surface** which touch the **curvature** of the error surface
  - The 2nd order gradient is represented by a **Hessian** Matrix

Gradient Descent and Newton's Method

- Gradient Descent: update parameters along the steepest descent direction

$$\theta_{k+1} = \theta_k - \eta \nabla_\theta J(\theta_k)$$

- Newton's Method: estimate a sequence of optima (no learning rate) through quadratic curves, using Tylor Expansion

$$J(\theta + \Delta) = J(\theta) + G(\theta)\Delta + \Delta^T H(\theta)\Delta + o(\Delta^2)$$

$$G(\theta)\Delta + \Delta^T H(\theta)\Delta = 0$$

$$\theta_{k+1} = \theta_k - H^{-1}(\theta_k)G(\theta_k)$$

Gradient Descent and Newton's Method

- Gradient Descent: update parameters along the steepest descent direction

$$\theta_{k+1} = \theta_k - \eta \nabla_\theta J(\theta_k)$$

<- eta

- Newton's Method: estimate a sequence of optima (no learning rate) through quadratic curves, using Tylor Expansion

$$J(\theta + \Delta) = J(\theta) + G(\theta)\Delta + \Delta^T H(\theta)\Delta + o(\Delta^2)$$

$$G(\theta)\Delta + \Delta^T H(\theta)\Delta = 0$$

$$\theta_{k+1} = \theta_k - H^{-1}(\theta_k)G(\theta_k)$$

<- no eta

Second Order Convergence

- Assume x0 is close to x*, Hessian of x* is not singular, and Hessian around x* is k-Lipschitz continuous

$$
\begin{aligned}
\|x_{k+1} - x^*\| &= \|x_k - x^* - h(x_k)^{-1} g(x_k)\| \\
&= \|h(x_k)^{-1}(g(x_k) - h(x_k)(x_k - x^*))\| \\
&= \|h(x_k)^{-1}(g(x_k) - g(x^*) - h(x_k)(x_k - x^*))\| \\
&\leq \|h(x_k)^{-1}\| * \|g(x_k) - g(x^*) - h(x_k)(x_k - x^*)\| \\
&\leq \|h(x_k)^{-1}\| * k\|x_k - x^*\|^2 \leq \frac{k}{\lambda_{min}}\|x_k - x^*\|^2
\end{aligned}
$$

Quasi-Newton Method

- Construct a **positive definite symmetric matrix** to approximate **Hessian** (or inverse Hessian) instead of 2nd order derivatives
- Quasi-Newton condition

$$g_{k+1} - g_k \approx H_{k+1} * (x_{k+1} - x_k)$$

$$s_k = x_{k+1} - x_k, y_k = g_{k+1} - g_k$$

$$y_k \approx H_{k+1} * s_k, s_k \approx H_{k+1}^{-1} * y_k$$

$$y_k = B_{k+1} * s_k, s_k = D_{k+1} * y_k$$

Davindon-Fletcher-Powell

$$D_{k+1} = D_k + \Delta D_k$$

$$\Delta D_k = \alpha u u^T + \beta v v^T$$

$$s_k = D_k y_k + (\alpha u^T y_k) u + (\beta v^T y_k) v$$

$$\text{set: } \alpha u^T y_k = 1, \beta v^T y_k = -1 \qquad \alpha = \frac{1}{u^T y_k}, \beta = \frac{1}{v^T y_k}$$

$$u - v = s_k - D_k y_k$$

$$\text{set: } u = s_k, v = D_k y_k$$

$$D_{k+1} = D_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{D_k y_k y_k^T D_k}{y_k^T D_k y_k} \qquad (D_0 = I)$$

Broyden-Fletcher-Goldfarb-Shanno

- Similar to DFP, but approximate Hessian directly
- Theoretical guarantee for convergence
- Need to store an N*N matrix: Limited-memory BFGS (L-BFGS)

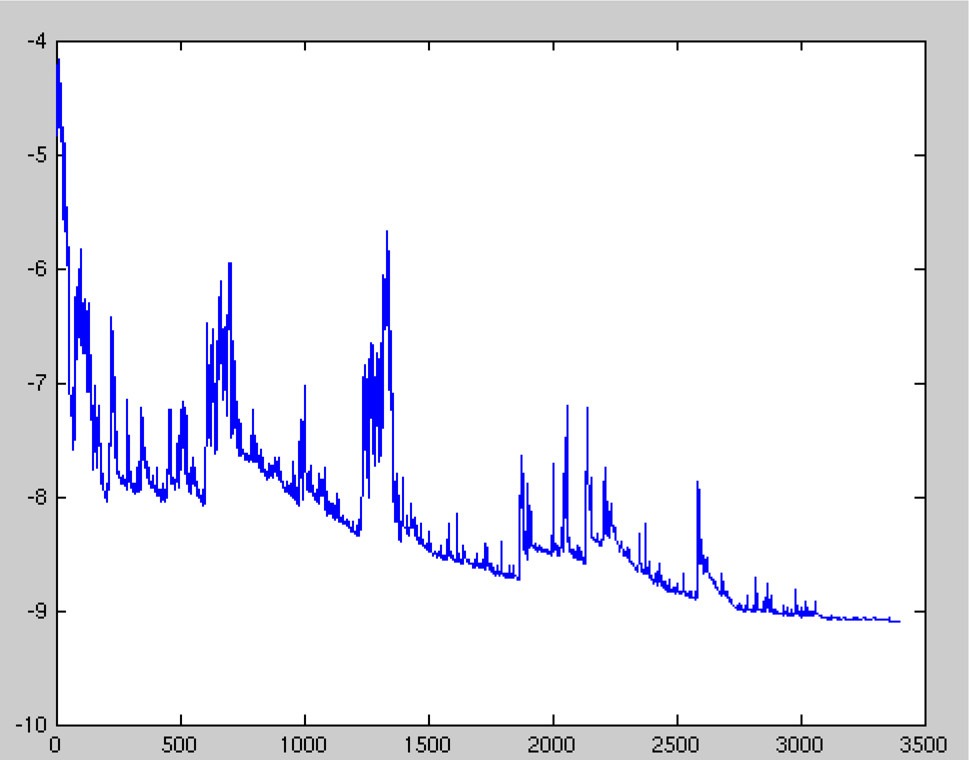
Why not Second Order Optimization

- Complexity: Gradient Descent $O(n)$, Quasi-Newton $O(n^2)$, Newton $O(n^3)$, where n is the amount of parameters
- Cramér-Rao bound states that generalization error cannot decrease faster than $O(1/k)$ in strongly convex problems
- Robustness, e.g., numerical stability

First Order Optimization

- We usually use SGD to refer mini-batch gradient descent

SGD fluctuation



Challenges of First Order Optimization

- Difficult to choose a constant learning rate
- Learning rate schedules are defined in advance thus unable to adapt to dataset’s characteristics
- Different features have different frequencies, we might not want to update all of them to the same extent
- Escape from local minima and saddle points

Momentum (MOM)

- Algo:

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta)$$
$$\theta = \theta - v_t$$

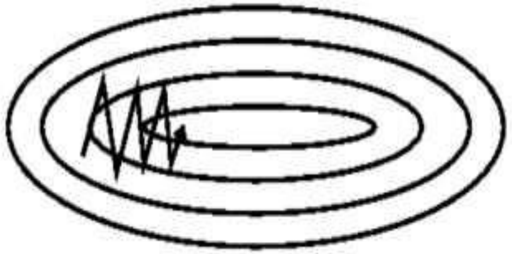


Image 2: SGD without momentum

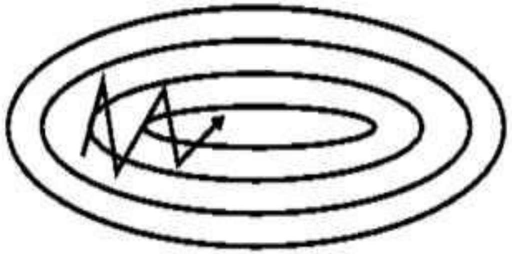


Image 3: SGD with momentum

- Physical Analogy:

$$\frac{d\theta}{dt} = -\eta \nabla_\theta J(\theta)$$

$$m\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} = -\nabla_\theta J(\theta)$$

$$m\frac{\theta_{t+\Delta t} + \theta_{t-\Delta t} - 2\theta_t}{\Delta t^2} + \mu\frac{\theta_{t+\Delta t} - \theta_t}{\Delta t} = -\nabla_\theta J(\theta)$$

$$\theta_{t+\Delta t} - \theta_t = \frac{m}{m+\mu\Delta t}(\theta_t - \theta_{t-\Delta t}) - \frac{(\Delta t)^2}{m+\mu\Delta t}\nabla_\theta J(\theta)$$

$$\gamma = \frac{m}{m+\mu\Delta t}$$

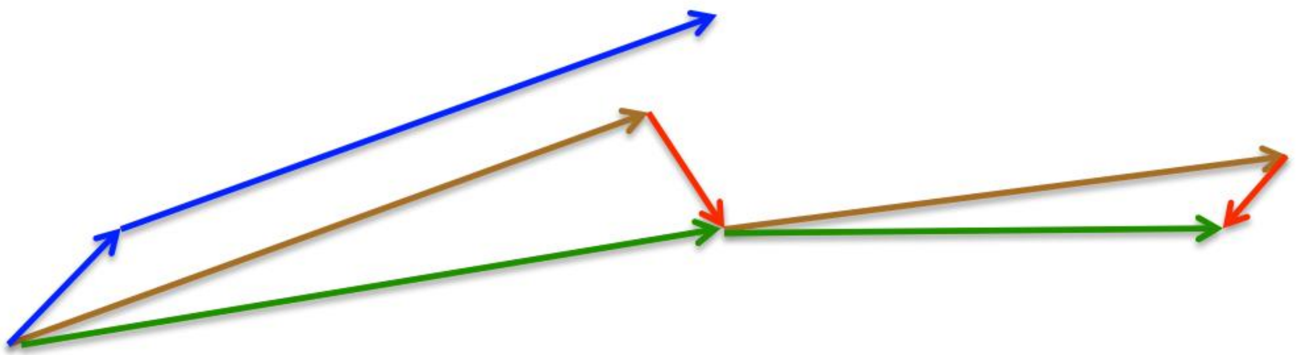
$$\eta = -\frac{(\Delta t)^2}{m+\mu\Delta t}$$

Nesterov Accelerated Gradient (NAG)

- Algo :

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1})$$
$$= \gamma v_{t-1} + \eta \nabla_\theta J(\theta) + \eta \left(\nabla_\theta J(\theta - \gamma v_{t-1}) - \nabla_\theta J(\theta)\right)$$
$$\theta = \theta - v_t$$

- Vector presentation :

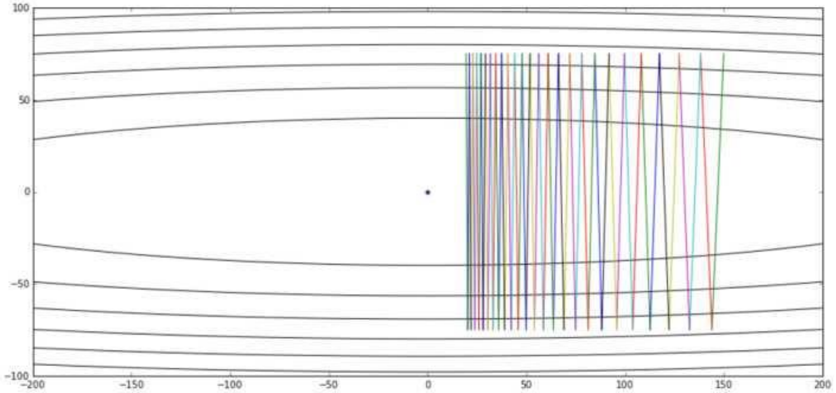


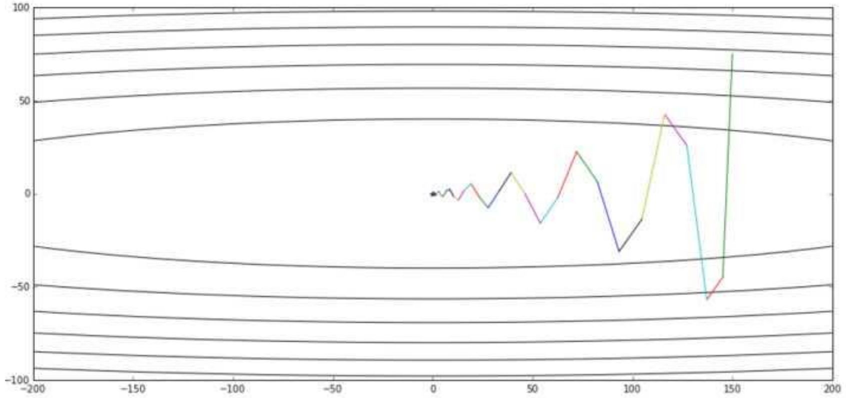
Momentum and Nesterov

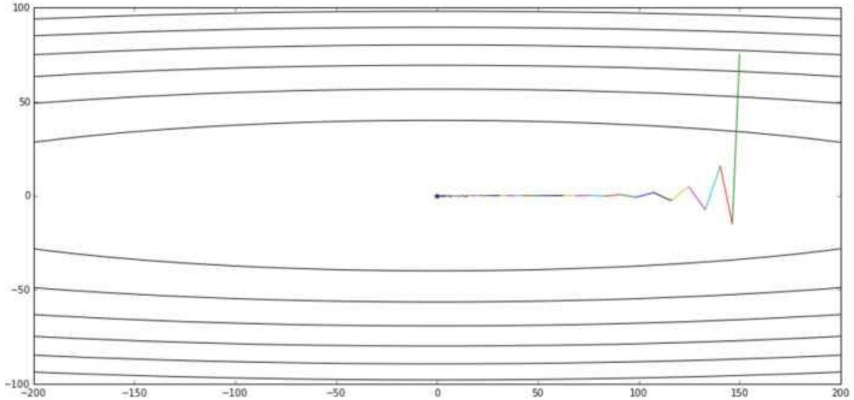
- NAG v.s. MOM (z = x^2 + 50y^2)

$$\begin{cases} \hat{\theta}_t & \triangleq \theta_t - \gamma v_t \\ \hat{v}_t & \triangleq (\frac{\gamma}{\eta})^2 v_{t-1} + (\frac{\gamma}{\eta} + 1)\nabla_\theta J(\theta)(\theta_{t-1} - \gamma v_{t-1}) \end{cases}$$

$$\begin{cases} \hat{v}_t & = \gamma \hat{v}_{t-1} + \eta \nabla_\theta J(\hat{\theta}_{t-1}) + \gamma[\nabla_\theta J(\hat{\theta}_{t-1}) - \nabla_\theta J(\hat{\theta}_{t-2})] \\ \hat{\theta} & = \hat{\theta} - \hat{v}_t \end{cases}$$







Adagrad

- Algo:

$$\begin{cases} g_{t,i} &= \nabla_\theta J(\theta_i) \\ \theta_{t+1,i} &= \theta_{t,i} - \eta g_{t,i} \end{cases} \Rightarrow \begin{cases} g_{t,i} &= \nabla_\theta J(\theta_i) \\ G_{t,ii} &= \sum_{\tau=1}^{t} g_{\tau,i}^2 \\ \theta_{t+1,i} &= \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii}+\epsilon}} g_{t,i} \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{G_t+\epsilon}} \odot g_t \end{cases}$$

- Generalization of GD:

$$\begin{cases} x_{t+1} = \prod_\chi (x_t - \eta g_t) = argmin||x - (x_t - \eta g_t)||_2^2 \\ ||.||_A = \sqrt{\langle ., A. \rangle} \\ g = \left[g_1, g_2 \dots g_t \right] \\ G_t = \sum_{\tau=1}^{t} g_\tau g_\tau^T \end{cases}$$

$$\begin{cases} x_{t+1} = \prod_\chi^{G_t^{1/2}} (x_t - \eta G_t^{-1/2} g_t) \\ x_{t+1} = \prod_\chi^{diag(G_t)^{1/2}} (x_t - \eta diag(G_t)^{-1/2} g_t) \end{cases}$$

Adadelta

- Accumulate over window

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t, \text{ or } \theta_{t+1} = \theta_t - \frac{\eta}{RMS(g)_t} g_t$$

<- eps

- Unit correction

$$\Delta\theta = \frac{\partial J / \partial \theta}{\partial^2 J / \partial \theta^2} \Rightarrow \frac{1}{\partial^2 J / \partial \theta^2} g_t = \frac{\Delta\theta}{\partial J / \partial \theta} g_t$$

$$\Delta\theta = -\frac{RMS(\Delta\theta)_{t-1}}{RMS(g)_t} g_t, \text{Matthew Zeiler, 2012}$$

<- eta

$$(\Delta\theta_t = -\frac{1}{|diag(H_t)|} \frac{E[g_{t-w:t}]^2}{E[g_{t-w:t}^2]} g_t, \text{Schaul, Zhang, LeCun 2012})$$

RMSProp

- A special case of Adadelta, Hinton, unpublished

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2, \gamma = 0.9$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{E[g^2]_t + \epsilon} g_t, \eta = 0.001$$

<- eta

Adaptive Moment Estimation

- AdaGrad + RMSProp

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, m_0 = 0$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, v_0 = 0$$

- Bias correction

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

$$\hat{v}_t = v_t / (1-\beta_2^t)$$

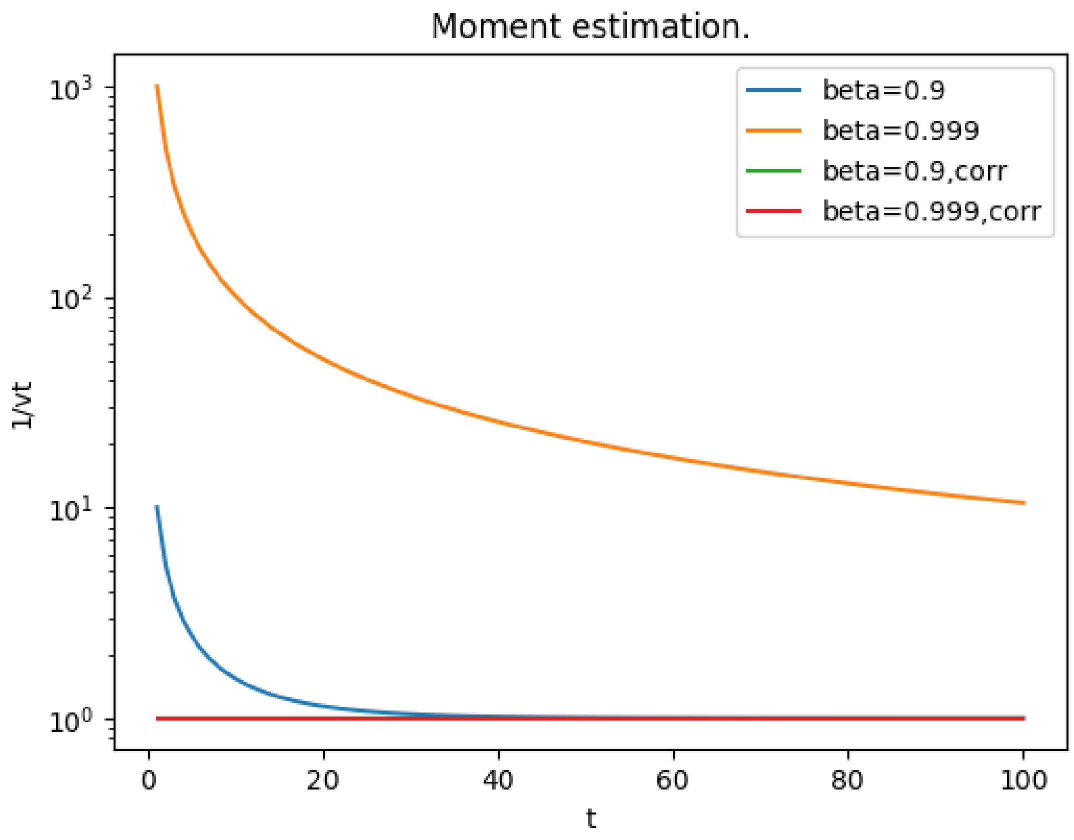
$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \text{Kingma, 2014}$$

$$\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e-8$$

$$v_t = (1-\beta_2) \sum_{i=1}^{t} \beta_2^{t-i} g_i^2$$

$$\begin{aligned} E[v_t] &= E[(1-\beta_2) \sum_{i=1}^{t} \beta_2^{t-i} g_i^2] \\ &= E[g^2](1-\beta_2) \sum_{i=1}^{t} \beta_2^{t-i} + \delta \\ &= E[g^2](1-\beta_2^t) + \delta \end{aligned}$$

Moment Estimation



Adam Convergence

- An online learning framework, Zinkevich, 2003

convex cost functions $f_1(\theta), f_2(\theta), \ldots, f_T(\theta)$

optimize regret $$R(T) = \sum_{t=1}^{T} [f_t(\theta_t) - f_t(\theta^*)]$$

- Adam has regret bound $R(T) = O(\sqrt{T})$
- Convergence $$\lim_{T \to \infty} \frac{R(T)}{T} = 0$$

AdaMax and Nadam

- AdaMax: a variant of Adam

$$v_t = \beta_2^p v_{t-1} + (1 - \beta_2^p)|g_t|^p$$ <- large p-norm is unstable

$$u_t = \lim_{p \to \infty} (v_t)^{1/p} = \max(\beta_2 u_{t-1}, |g_t|)$$ <- no bias

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{u_t}, \text{Kingma, 2014}$$

- Nadam: Adam + Nesterov, Timothy, 2016

Conclusion

```
SGD -> Momentum -> Nesterov
     \                  v
      \     Adam ->  Nadam
       v    ^    ^
     AdaGrad   -> RMSProp -> AdaDelta
                               ^
Newton -> Quasi-Newton _____|
```

Thanks!